TOSHIBA

32 bit TX System RISC
TX19 Family
TMP19A71CYFG／UG
TMP19A71FYFG／UG

Not Recommended for New Design

Rev 2.0（Feb.2007）

TOSHIBA CORPORATION

Contents

Not Recommended for New Design

32-Bit RISC Microprocessor TX19 Family

TMP19A71FYFG/FYUG/CYFG/CYUG

1. Features

The TX19A core processor contained in the TMP19A71 is a family of high-performance 32-bit microprocessors that offers the speed of a 32-bit RISC solution with the added advantage of a significantly reduced code size of a 16-bit architecture. The instruction set of the TX19A includes the high-performance MIPS32ISA, and is enhanced by the MIPS16e-TX™ Application-Specific Extensions (ASE) based on the highly code-efficient MIPS16eISA of MIPS Technologies, Inc. and with added instructions by Toshiba.

The TMP19A71 is built on a TX19A core processor and contains a selection of intelligent peripherals. It is suitable for low-voltage and low-power applications.

The TMP19A71 has the following features:

(1) TX19A core processor (For details, refer to the TX19A Architecture manual.)

1) Two instruction set architecture (ISA) modes: 16-bit ISA for code density and 32-bit ISA for speed
   - The 16-bit ISA is object-code compatible with the code-efficient MIPS16e™ASE.
   - The 32-bit ISA is object-code compatible with the high-performance TX39 Family.

2) Combines high performance with low power consumption.
   - High performance
     - Single clock cycle execution (except for save, restore, jump/branch instructions)
     - 3-operand computational instructions for high instruction throughput
     - 5-stage pipeline
     - On-chip high-speed memory
     - DSP function: Executes 32-bit multiply-accumulate operations (32-bit x 32-bit + 64-bit = 64-bit) in a single clock cycle.
   - Low power consumption
     - Optimized design using a low-power cell library

Not Recommended for New Design

060116EBP

• The information contained herein is subject to change without notice. 021023_D
• TOSHIBA is continually working to improve the quality and reliability of its products. Nevertheless, semiconductor devices in general can malfunction or fail due to their inherent electrical sensitivity and vulnerability to physical stress. It is the responsibility of the buyer, when utilizing TOSHIBA products, to comply with the standards of safety in making a safe design for the entire system, and to avoid situations in which a malfunction or failure of such TOSHIBA products could cause loss of human life, bodily injury or damage to property.
In developing your designs, please ensure that TOSHIBA products are used within specified operating ranges as set forth in the most recent TOSHIBA products specifications. Also, please keep in mind the precautions and conditions set forth in the "Handling Guide for Semiconductor Devices," or "TOSHIBA Semiconductor Reliability Handbook" etc. 021023_A
• The TOSHIBA products listed in this document are intended for usage in general electronics applications (computer, personal equipment, office equipment, measuring equipment, industrial robotics, domestic appliances, etc.). These TOSHIBA products are neither intended nor warranted for usage in equipment that requires extraordinarily high quality and/or reliability or a malfunctionor failure of which may cause loss of human life or bodily injury ("Unintended Usage"). Unintended Usage include atomic energy control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, combustion control instruments, medical instruments, all types of safety devices, etc. Unintended Usage of TOSHIBA products listed in this document shall be made at the customer's own risk. 021023_B
• The products described in this document shall not be used or embedded to any downstream products of which manufacture, use and/or sale are prohibited under any applicable laws and regulations. 060106_Q
• The information contained herein is presented only as a guide for the applications of our products. No responsibility is assumed by TOSHIBA for any infringements of patents or other rights of the third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of TOSHIBA or others. 021023_C
• The products described in this document are subject to the foreign exchange and foreign trade laws. 021023_E
• For a discussion of how the reliability of microcontrollers can be predicted, please refer to Section 1.3 of the chapter entitled Quality and Reliability Assurance/Handling Precautions. 030619_S

- Programmable standby modes in which processor clocks are stopped

3) Fast interrupt response suitable for real-time control

- Distinct starting locations for each interrupt service routine
- Automatically generated vectors for each interrupt source
- Automatic updates of the interrupt mask level

(2) On-chip program memory and data memory

Product	On-Chip ROM	On-Chip RAM
TMP19A71FYFG/UG	256 Kbytes Flash ROM	10 Kbytes
TMP19A71CYFG/UG	256 Kbytes Mask ROM	10 Kbytes

- ROM correction logic (8 words x 8 blocks)

(3) 8-channel DMA controller

- Interrupt- or software-triggered
- Transfer destination: On-chip memory, on-chip peripherals

(4) 4-channel 16-bit timer

- 16-Bit Interval Timer mode
- 16-Bit Event Counter mode
- 16-Bit PPG output
- Input capture

(5) 4-channel general-purpose serial interface

- Either UART mode or Synchronous mode can be selected for 2 channels; the other 2 channels are UART only.
- 50% duty cycle generation (for UART mode only)

(6) 2-channel 3-phase PWM generation (PMD)

- Generating 3-phase PWM with a resolution of 35.7 ns (at IMCLK = 28 MHz)
- Dead time insertion
- 3-phase PWM generation disabled under abnormal condition
- Two channels can be started synchronously.

(7) 1-channel ABZ encoder

- Supporting incremental encoder
- Rotation direction detection circuit
- Absolute position detection circuit
- Position comparison circuit
- On-chip noise filter

(8) 19-channel 10-bit AD converter (with internal sample and hold)

- High-speed conversion (min: 2.36 μs)
- Input voltage range: 0 V to 3.3 V
- External trigger supported
- Fixed-Channel or Channel Scan mode
- Single Conversion or Continuous Conversion mode

Not Recommended for New Design

- High-Priority Conversion mode
- AD conversion monitoring
- PMD mode

(9) 1-channel watchdog timer

(10) Interrupt sources

- 2 CPU interrupts: Software interrupt (within the co-processor)
- 37 internal interrupts: 7 priority levels (excluding the watchdog timer interrupt)
- 11 external interrupts: 7 priority levels (excluding the NMI interrupt)

(11) 75-pin input/output ports

(12) Standby modes

- Three standby modes: DOZE, HALT, STOP

(13) Clock generator

- On-chip PLL (x 16)
- Clock gear: Divides the high-speed clock to 1/2, 1/4 or 1/8.

(14) Endian

- Little-endian fixed

(15) Power voltage

- Peripheral I/O: Vcc3 = 3.3V ±0.3 V (TMP19A71FYFG/UG, TMP19A71CYFG/UG)
- Internal: Vcc2 = 2.5V ±0.2 V (MP19A71FYFG/UG)
- Internal: VccC15 = 1.5V ±0.15 V (TMP19A71CYFG/UG)

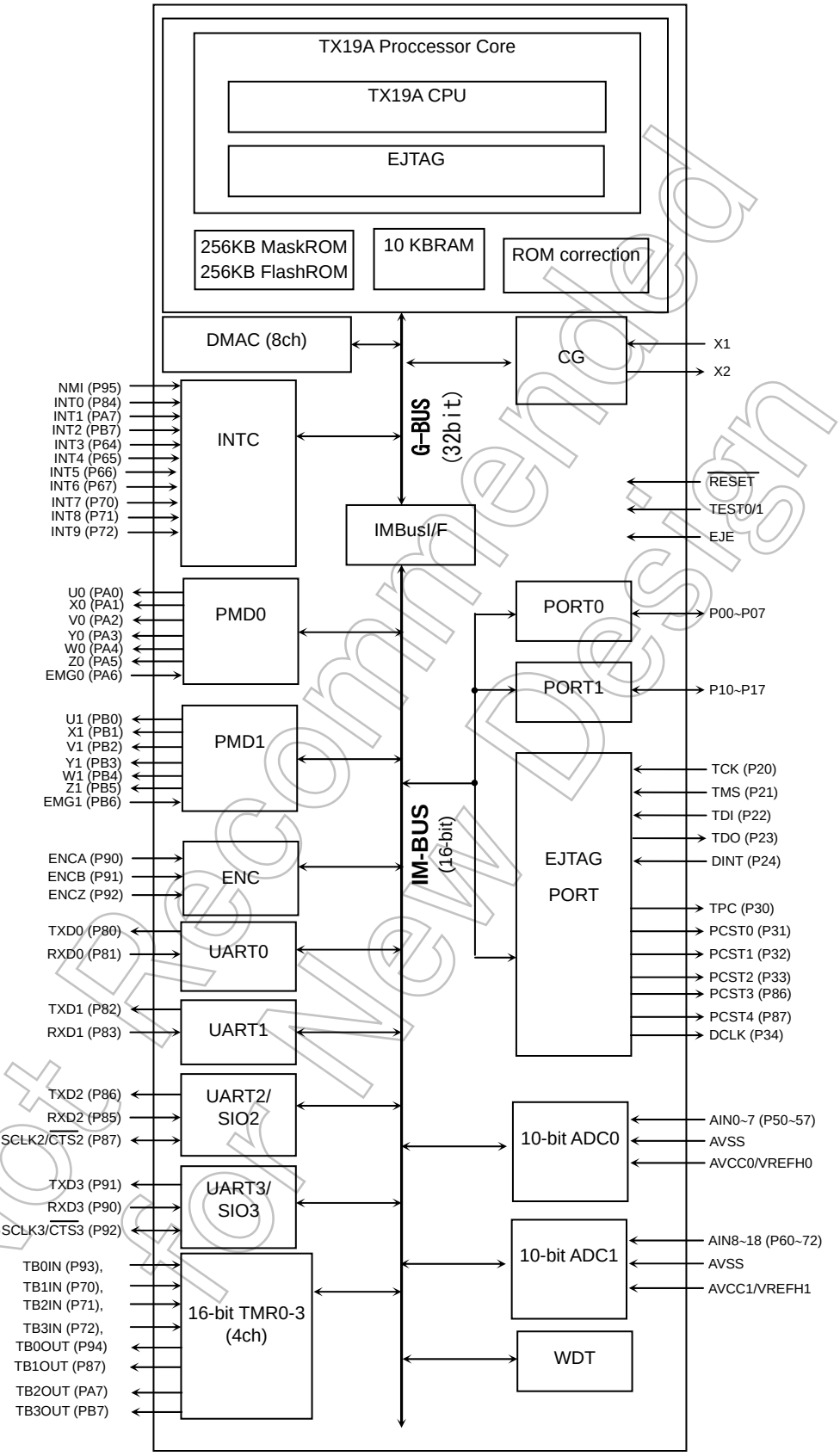
(16) Operating frequency

- 56 MHz (Vcc2 = 2.5V ±0.2 V: TMP19A71FYFG/UG)
- 56 MHz (Vcc15 = 1.5V ±0.15 V: TMP19A71CYFG/UG)

(17) Package

- P-LQFP100-1414-0.50F (14mm × 14mm, 0.5-mm pitch): TMP19A71FYUG/CYUG
- P-QFP100-1420-0.65A (14mm × 20mm, 0.65-mm pitch): TMP19A71FYFG/CYFG

Not Recommended for New Design



( ): Default function after reset

Figure 1.1 TMP19A71 Block Diagram

Not Recommended for New Design

2. Pin Assignments and Pin Functions

This section contains pin assignments for the TMP19A71 as well as brief descriptions of the TMP19A71 input and output signals.

2.1 TMP19A71CYFG/UG Pin Assignments

Figure 2.1 shows the pin assignments of the TMP19A71CYUG.

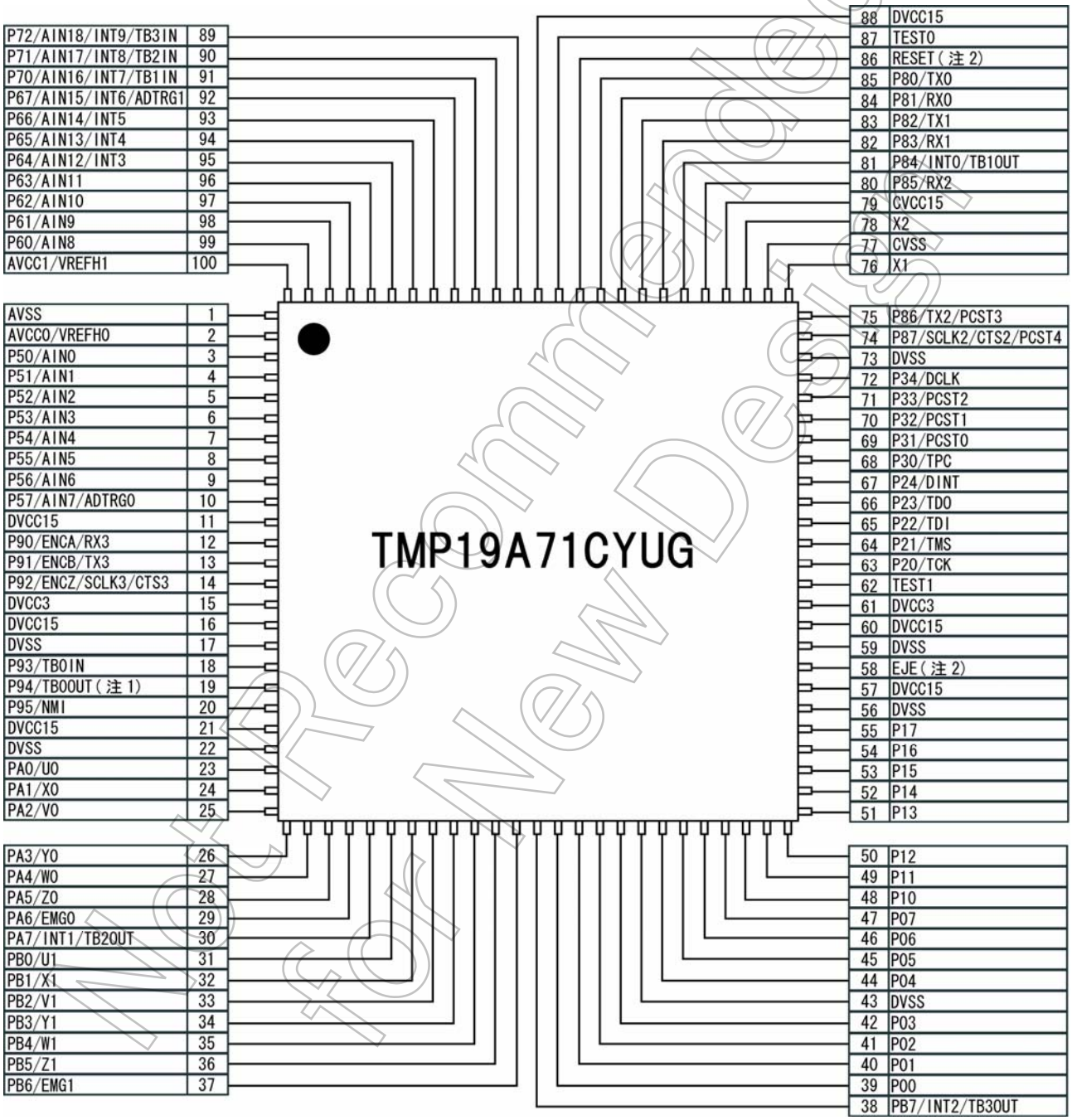


Figure 2.1 TMP19A71CYUG Pin Assignments (100-pin LQFP)

Note 1: This pin should be set to High during a reset sequence.

Note 2: These signals are Low active.

Not Recommended for New Design

Figure 2.2 shows the pin assignments of the TMP19A71CYFG.

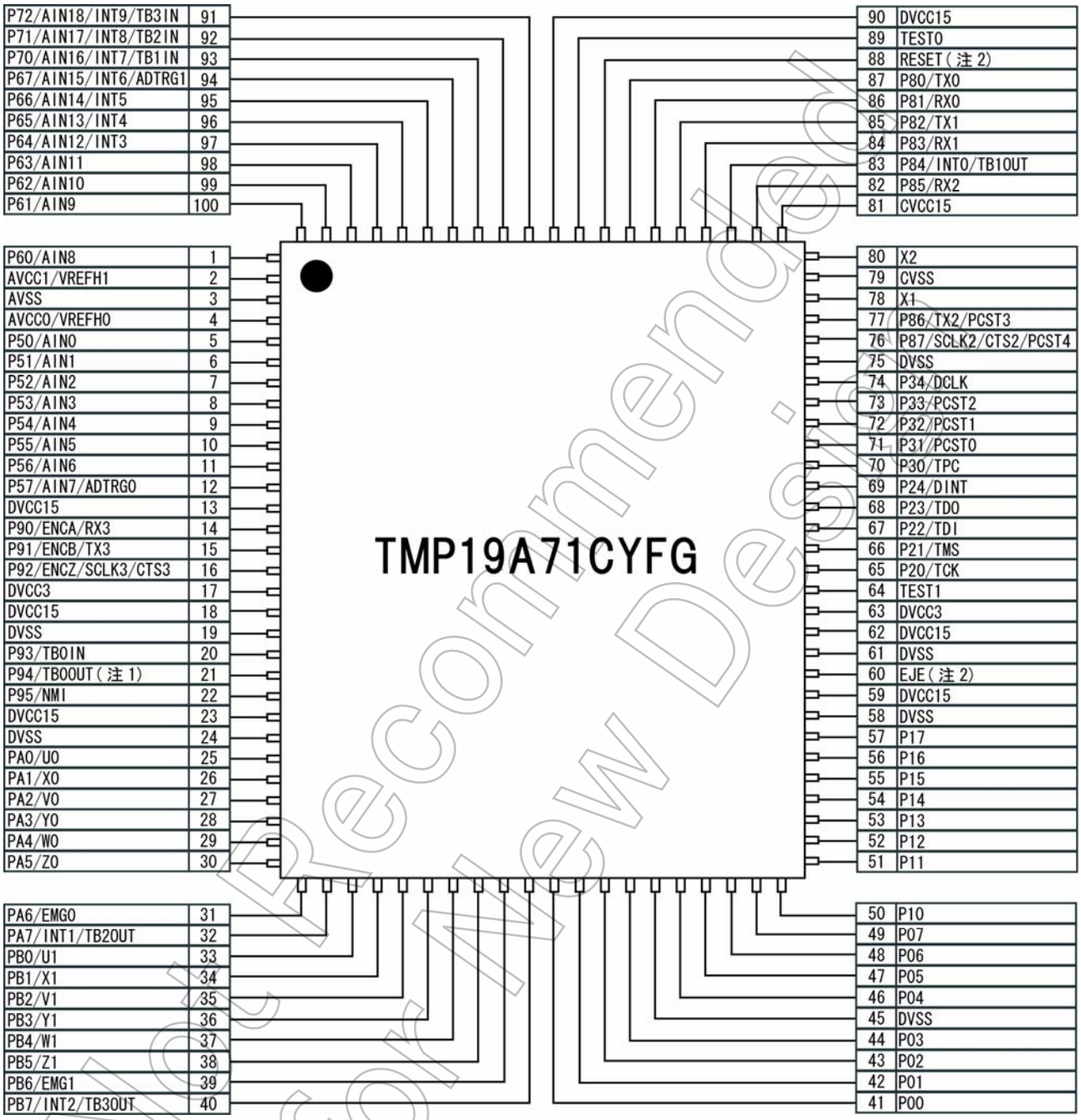


Figure 2.2 TMP19A71CYFG Pin Assignments (100-pin QFP)

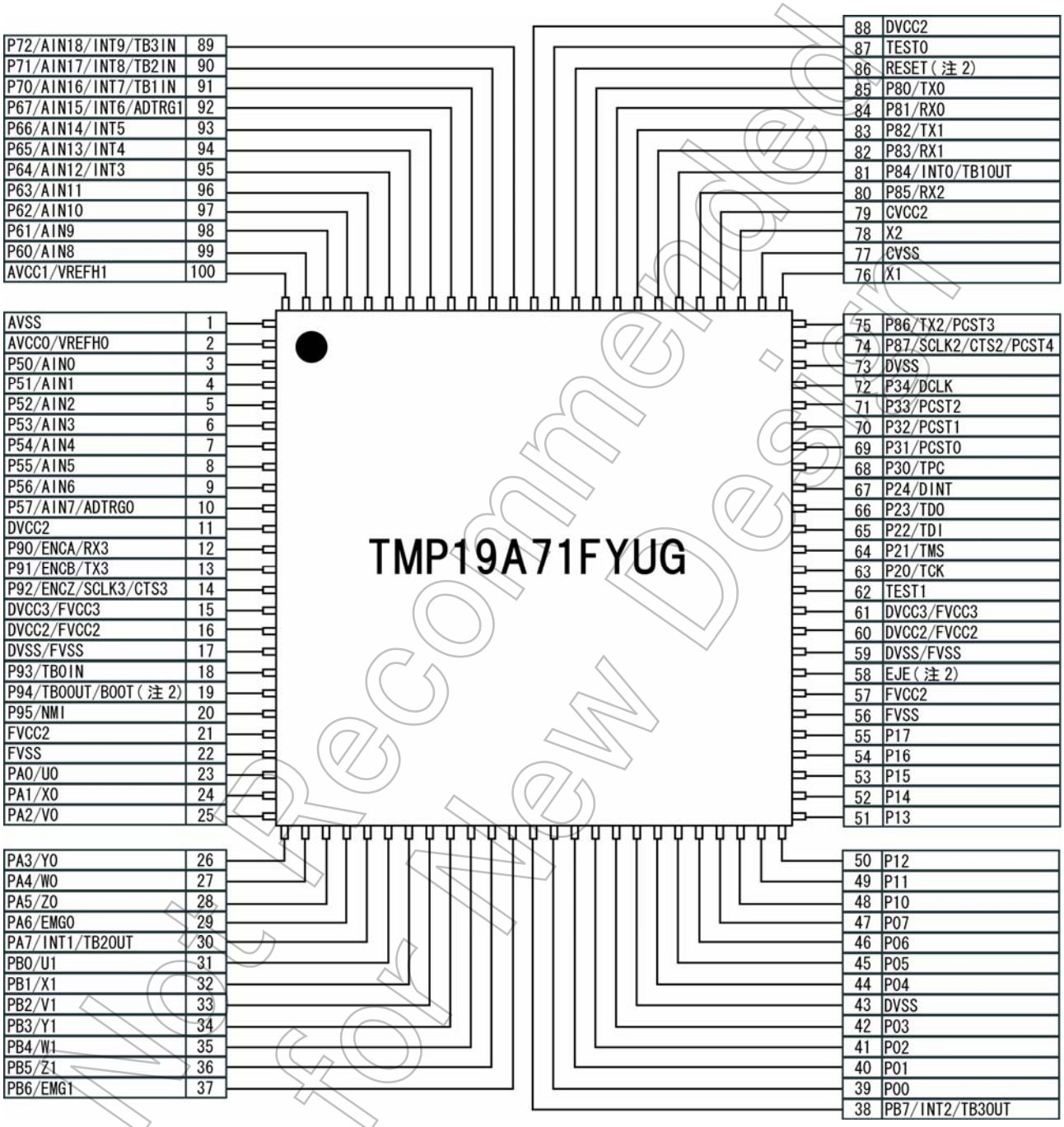
Not Recommended for New Design

Note 1: This pin should be set to High during a reset sequence.

Note 2: These signals are Low active.

2.2 TMP19A71FYFG/UG Pin Assignments

Figure 2.3 shows the pin assignments of the TMP19A71FYUG.



Not Recommended for New Design

Figure 2.3 TMP19A71FYUG Pin Assignments (100-pin LQFP)

Note 1: This pin should be set to High during a reset sequence.

Note 2: These signals are Low active.

Figure 2.4 shows the pin assignments of the TMP19A71FYFG.

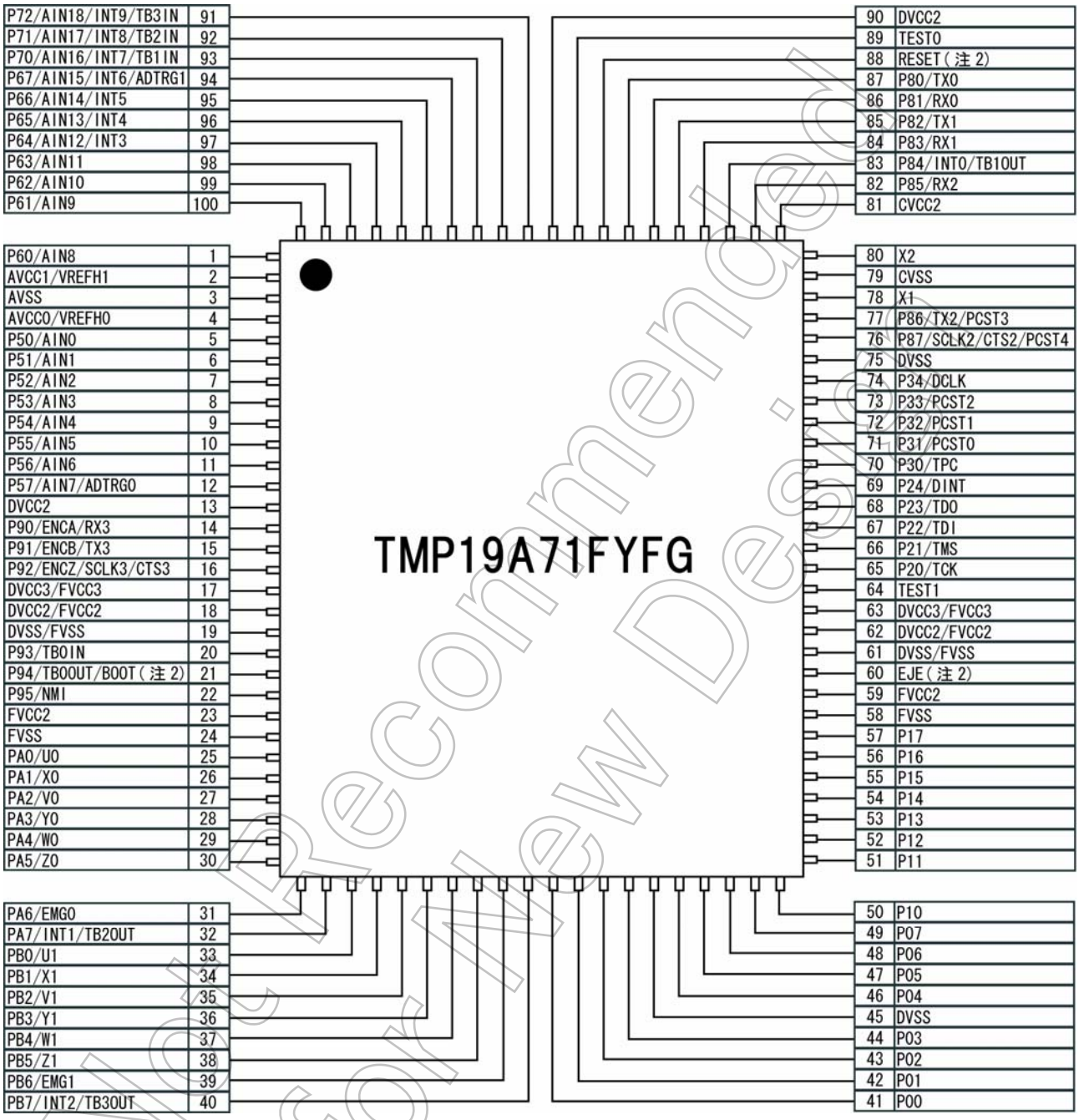


Figure 2.4 TMP19A71FYFG Pin Assignments (100-pin QFP)

Not Recommended for New Design

Note 1: This signal must be set to High during a reset sequence.

Note 2: These signals are Low active.

2.3 Pin Names and Functions

Table 2.3.1 lists the input and output pins of the TMP19A71, including alternate pin names and functions for multi-function pins.

Table 2.3.1 Pin Names and Functions (1/3)

Pin Name	Number of Pins	Type	Function
P00 to P07	8	Input/Output	Port 0: Individually programmable as input or output
P10 to P17	8	Input/Output	Port 1: Individually programmable as input or output
P20 TCK	1	Input/Output Input	Port 20: Programmable as input or output EJTAG pin (Schmitt-triggered input)
P21 TMS	1	Input/Output Input	Port 21: Programmable as input or output EJTAG pin (Schmitt-triggered input)
P22 TDI	1	Input/Output Input	Port 22: Programmable as input or output EJTAG pin (Schmitt-triggered input)
P23 TDO	1	Input/Output Output	Port 23: Programmable as input or output EJTAG pin
P24 DINT	1	Input/Output Input	Port 24: Programmable as input or output EJTAG pin (Schmitt-triggered input)
P30 TPC	1	Input/Output Output	Port 30: Programmable as input or output EJTAG pin
P31 PCST0	1	Input/Output Output	Port 31: Programmable as input or output EJTAG pin
P32 PCST1	1	Input/Output Output	Port 32: Programmable as input or output EJTAG pin
P33 PCST2	1	Input/Output Output	Port 33: Programmable as input or output EJTAG pin
P34 DCLK	1	Input/Output Output	Port 34: Programmable as input or output EJTAG pin
P50 to P57 AN0 to AN7	8	Input Input	Port 5: Input-only Analog Input: Input to the AD converter
P60 to P63 AN8 to AN11	4	Input Input	Port 60 to 63: Input-only Analog Input: Input to the AD converter
P64 to P67 AN12 to AN15 INT3 to INT6	4	Input/Output Input Input	Port 64 to 67: Programmable as Schmitt-triggered input or output Analog Input: Input to the AD converter External interrupt pins: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive
P70 AN16 INT7 TB1IN	1	Input/Output Input Input Input	Port 70: Programmable as Schmitt-triggered input or output Analog Input: Input to the AD converter External Interrupt 7: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive 16-Bit Timer Input: Input to 16-bit timer 1
P71 AN17 INT8 TB2IN	1	Input/Output Input Input Input	Port 71: Programmable as Schmitt-triggered input or output Analog Input: Input to the AD converter External Interrupt 8: Programmable to be high-level, low-level, rising-edge or falling edge sensitive 16-bit Timer 2 Input: Input to 16-bit timer 2
P72 AN18 INT9 TB3IN	1	Input/Output Input Input Input	Port 72: Programmable as Schmitt-triggered input or output Analog Input: Input to the AD converter External Interrupt 9: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive 16-Bit Timer 3 Input: Input to 16-bit timer 3

Not Recommended for New Design

Table 2.3.2 Pin Names and Functions (2/3)

Pin Name	Number of Pins	Type	Function
P80	1	Input/Output	Port 80: Programmable as input or open-drain output
TX0		Output	Serial Transmit Data 0
P81	1	Input/Output	Port 81: Programmable as input or output
RX0		Input	Serial Receive Data 0
P82	1	Input/Output	Port 82: Programmable as input or open-drain output
TX1		Output	Serial Transmit Data 1
P83	1	Input/Output	Port 83: Programmable as input or output
RX1		Input	Serial Receive Data 1
P84	1	Input/Output	Port 84: Programmable as Schmitt-triggered input or output
INT0		Input	External interrupt pin
TB1OUT		Output	16-Bit Timer 1 Output: Output from 16-bit timer 1
P85	1	Input/Output	Port 85: Programmable as input or output
RX2		Input	Serial Receive Data 2
P86	1	Input/Output	Port 86: Programmable as input or open-drain output
TX2		Output	Serial Transmit Data 2
PCST3		Output	EJTAG pin
P87	1	Input/Output	Port 87: Programmable as Schmitt-triggered input or open-drain output
SCLK2		Input/Output	Serial Clock Input/Output 2
CTS2		Output	Serial Clear-to-Send 2
PCST4		Output	EJTAG pin
P90	1	Input/Output	Port 90: Programmable as Schmitt-triggered input or output
ENCA		Input	Encoder A-phase input pin
RX3		Input	Serial Receive Data 3
P91	1	Input/Output	Port 91: Programmable as Schmitt-triggered input or output
ENCB		Input	Encoder B-phase input pin
TX3		Output	Serial Transmit Data 3
P92	1	Input/Output	Port 92: Programmable as Schmitt-triggered input or output
ENCZ		Input	Encoder Z-phase input pin
SCLK2		Input/Output	Serial Clock Input/Output 3
CTS2		Output	Serial Clear-to-Send 3
P93	1	Input/Output	Port 93: Programmable as Schmitt-triggered input or output
TB0IN		Input	16-Bit Timer 0 Input: Input to 16-bit timer 0 and emergency stop input pin
P94	1	Input/Output	Port 94: Programmable as input or output
TB0OUT		Output	16-Bit Timer 0 Output: Output from 16-bit timer 0
BOOT (Note)			Single boot mode set pin: Should be set to Low to start up in Boot mode.
P95	1	Input/Output	Port 95: Programmable as Schmitt-triggered input or output
NMI		Input	Nonmaskable Interrupt Request: Programmable to be rising-edge or falling edge sensitive
PA0	1	Input/Output	Port A0: Programmable as input or output
U0		Output	PMD0: U-phase output
PA1	1	Input/Output	Port A1: Programmable as input or output
X0		Output	PMD0: X-phase output
PA2	1	Input/Output	Port A2: Programmable as input or output
V0		Output	PMD0: V-phase output
PA3	1	Input/Output	Port A3: Programmable as input or output
Y0		Output	PMD0: Y-phase output
PA4	1	Input/Output	Port A4: Programmable as input or output
W0		Output	PMD0: W-phase output
PA5	1	Input/Output	Port A5: Programmable as input or output
Z0		Output	PMD0: Z-phase output

Not Recommended for New Design

Table 2.3.3 Pin Names and Functions (3/3)

Pin Name	Number of Pins	Type	Function
PA6 EMG0	1	Input/Output Input	Port A6: Programmable as Schmitt-triggered input or output PMD0: Emergency stop input pin
PA7 INT1 TB2OUT	1	Input/Output Input Output	Port A7: Programmable as Schmitt-triggered input or output Interrupt Request 1: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive 16-Bit Timer 2 Output: Output from 16-bit timer 2
PB0 U1	1	Input/Output Output	Port B0: Programmable as input or output PMD1: U-phase output
PB1 X1	1	Input/Output Output	Port B1: Programmable as input or output PMD1: X-phase output
PB2 V1	1	Input/Output Output	Port B2: Programmable as input or output PMD1: V-phase output
PB3 Y1	1	Input/Output Output	Port B3: Programmable as input or output PMD1: Y-phase output
PB4 W1	1	Input/Output Output	Port B4: Programmable as input or output PMD1: W-phase output
PB5 Z1	1	Input/Output Output	Port B5: Programmable as input or output PMD1: Z-phase output
PB6 EMG1	1	Input/Output Input	Port B6: Programmable as Schmitt-triggered input or output PMD1: Emergency stop input pin
PB7 INT2 TB3OUT	1	Input/Output Input Output	Port B7: Programmable as Schmitt-triggered input or output Interrupt Request 2: Programmable to be high-level, low-level, rising-edge or falling edge sensitive 16-Bit Timer 3 Output: Output from 16-bit timer 3
AVSS	1	–	Ground pin (0 V) for the AD converter
AVCC0 /VREFH0	1	–	3.3-V power supply pin for the AD converter 0 Input pin for high reference voltage for the AD converter (Shared with the above pin)
AVCC1 /VREFH1	1	–	3.3-V power supply pin for the AD converter 1 Input pin for high reference voltage for the AD converter (Shared with the above pin)
EJE	1	Input	EJTAG Enable (Low active)
RESET	1	Input	Reset: Initialize LSI (Schmitt-triggered input with internal pull-up register, low active)
TEST0	1	–	Test pin: This pin should be tied to logic 0.
TEST1	1	–	Test pin: This pin should be tied to logic 0.
X1/X2	2	Input/Output	Connection pins for a resonator
Power Supply and Ground Pins for the Mask-Version Product			
CVCC15	1	–	1.5-V power supply pin for the oscillator
CVSS	1	–	Ground pin (0 V) for the oscillator
DVCC3	2	–	3.3-V power supply pin
DVCC15	6	–	1.5-V power supply pin
DVSS	6	–	Ground pin (0 V)
Power Supply and Ground Pins for the Flash-Version Product			
CVCC2	1	–	2.5-V power supply pin for the oscillator
CVSS	1	–	Ground pin (0 V) for the oscillator
FVCC3	(2)	–	3.3-V power supply pin for flash macro (Shared with DVCC3)
FVCC2	2	–	2.5-V power supply pin for flash macro
FVSS	2	–	Ground pin (0 V) for flash macro
DVCC3	2	–	3.3-V power supply pin
DVCC2	4	–	2.5-V power supply pin
DVSS	4	–	Ground pin (0V)

Note: This pin should be fixed to High in a mask-version product.

Not Recommended for New Design

3. Core Processor

The TMP19A71 contains a high-performance 32-bit core processor called the TX19A. For a detailed description of the core processor, refer to the TX19A Architecture manual.

The functions unique to the TMP19A71 not covered in the architecture manual are described below.

Note: All references to register addresses in the following description assume that the TMP19A71 is operating in Little-Endian mode.

3.1 Power-Up Sequence

To power up the TMP19A71, we recommend that the core power supply (2.5 V in a flash-version product and 1.5 V in a mask-version product) be turned on first.

3.2 Reset Operation

To reset the TMP19A71, $\overline{\text{RESET}}$ must be asserted for at least a specified period of time, as shown in Table 3.2.1, after the power supply voltage has stabilized. This time period is required to initialize internal circuits. If this requirement is not satisfied, the TMP19A71 may not operate properly due to improper initialization of internal circuits. The incorporated program begins executing 30 usec after $\overline{\text{RESET}}$ is released.

Table 3.2.1 Reset Input Time

Reset Timing	Equation (sec)	Required External Reset Input Time
Flash-version device: At power-on, and second and subsequent resets (CLKMISC.MSFR = 0)	Fixed	1 msec after power supply has stabilized
Flash-version device: Second and subsequent resets (CLKMISC.MSFR = 1) Mask-version device	32/X1	4.6 us (at 7MHz/) or 6.4 us (at 5 MHz) after oscillation has stabilized

Note: When oscillation is started, oscillation stabilization time and PLL lock-up time are additionally required.

The following occur as a result of a reset:

- The System Control Coprocessor (CP0) registers within the TX19A core processor are initialized. For details, refer to the TX19A Architecture manual.
- The Reset exception is taken. Program control is transferred to the exception handler at a predefined address. This predefined location is called an exception vector, which directly indicates the start of the actual exception handler routine. The Reset exception is always vectored to virtual address 0xBFC0_0000 (which is the same as for the Nonmaskable Interrupt exception).
- All on-chip I/O peripheral registers are initialized.
- All port pins, including those multiplexed with on-chip peripheral functions, are configured as either general-purpose inputs or general-purpose outputs.

Note 1: The TMP19A71 must be powered up with $\overline{\text{RESET}}$ asserted. The reset state should not be terminated until after the power supply voltage stablizes within the valid operating range.

Note 2: There is a possibility that on-chip RAM locations accessed and general-purpose registers of the selected bank may be corrupted during a reset.

Not Recommended for New Design

3.3 Start-Up Routine

The following explains a standard start-up routine. Write a start-up routine according to the requirements of your program.

1. Enable the shadow register sets

   Set the SSD bit of the SSCR register (CP0 register) to 0 to enable the shadow register sets.

2. Set the global pointer r28 (GP) and the stack pointer r29 (SP)

   Set the initial values in r28 and r29 as required. When the shadow register sets are used, it is necessary to set r29 separately for shadow register set 0 and shadow register sets 1 to 7.

3. Set the CP0 Status register

   In the CP0 Status register, set the CU0 bit (CP0 usability) to 1, the BEV bit (bootstrap exception vector) to 1, and the IM[4:2] field (interrupt mask) to 1, as required.

4. Set the CP0 Cause register

   Set the IV bit (interrupt vector) in the CP0 Cause register to 1, as required.

5. Set the block decode registers

   It is necessary to set the block decode registers to change the data read method according to whether the flash-version or mask-version device is used. If this setting is not made, internal ROM data cannot be read correctly. The B0DCR and B0DLR registers should be accessed from block 0, and the B1DCR and B1DLR registers should be accessed from block 1.

   (Programming examples)

   By using instructions stored at 0xBFC0_0000 to 0xBFC1_FFFF (0x0000_0000 to 0x0001_FFFF):

   B0DCR（0xFFFF_E530） <-- 0x00

   B0DLR（0xFFFF_E534） <-- 0x3D

   By using instructions stored at 0xBFC2_0000 to 0xBFC3_FFFF (0x0002_0000 to 0x0003_FFFF):

   B1DCR（0xFFFF_E538） <-- 0x00

   B1DLR（0xFFFF_E53C） <-- 0x3D

Not Recommended for New Design

Block 0 Decode Control Register

B0DCR (0xFFFF_E530)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	—	—	B0DECEN
Read/Write								R/W
Reset Value	0	0	0	0	0	0	0	1
Function								1: Flash version 0: Mask version

Note 1: In the mask-version device, the B0DECEN bit is not initialized by a WDT reset; it is initialized by an external reset.

Note 2: In the flash-version device, the B0DECEN bit is not initialized by a normal reset; it is initialized by a power-on reset.

Note 3: The B0DCR should be accessed by an instruction stored in block 0 (0xBFC0_0000 to 0xBFC1_FFFF or 0x0000_0000 to 0x0001_FFFF).

Block 0 Decode Lock Register

B0DLR (0xFFFF_E534)

	7	6	5	4	3	2	1	0
Bit Symbol	—							
Read/Write	W							
Reset Value	—	—	—	—	—	—	—	—
Function	The value written in the B0DLR.B0DECEN bit takes effect by writing 0x3D in this register.							

Note: The B0DLR should be accessed by an instruction stored in block 0 (0xBFC0_0000 to 0xBFC1_FFFF or 0x0000_0000 to 0x0001_FFFF).

Block 1 Decode Control Register

B1DCR (0xFFFF_E538)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	—	—	B1DECEN
Read/Write								R/W
Reset Value	0	0	0	0	0	0	0	1
Function								1: Flash version 0: Mask version

Note 1: In the mask-version device, the B1DECEN bit is not initialized by a WDT reset; it is initialized by an external reset.

Note 2: In the flash-version product, the B1DECEN bit is not initialized by a normal reset; it is initialized by a power-on reset.

Note 3: The B1DCR should be accessed by an instruction stored in block 1 (0xBFC2_0000 to 0xBFC3_FFFF or 0x0002_0000 to 0x0003_FFFF).

Block 1 Decode Lock Register

B1DLR (0xFFFF_E53C)

	7	6	5	4	3	2	1	0
Bit Symbol	—							
Read/Write	W							
Reset Value	—	—	—	—	—	—	—	—
Function	The value written in the B1DLR.B1DECEN bit takes effect by writing 0x3D in this register.							

Note: The B1DLR should be accessed by an instruction stored in block 1 (0xBFC2_0000 to 0xBFC3_FFFF or 0x0002_0000 to 0x0003_FFFF).

Not Recommended for New Design

3.4 Bus Cycles

In a processor using pipelining like the TX19A core processor, performance is greatly influenced by pipeline hazards. To improve performance, therefore, due consideration must be given to pipeline hazards related to bus cycles. The TX19A core processor controls bus cycles asynchronous to the pipeline (non-blocking loads, etc.) to prevent degradation in performance due to pipeline hazards.

In addition, taking account of DMA transfers triggered by external sources, it is extremely difficult to control bus cycles by software. The TX19A core processor is provided with the SYNC instruction for synchronization of bus cycles. The SYNC instruction stalls execution of the next instruction until all instructions generating bus cycles (including the write buffer) have been completed.

The following gives considerations related to bus cycles through explaining how to use the SYNC instruction. Please note that the following considerations may not apply and other considerations may be required depending on the system.

For a detailed description of the write buffer and bus cycles, refer to the TX19A Architecture manual.

3.4.1 Bus Cycle Execution Time

Table 3.4.1 shows the number of clock cycles required for completing the bus cycle of a load or store instruction. Since the start timing of each bus cycle varies depending on the write buffer and bus states, the values shown in this table may not always apply.

Table 3.4.1 Number of Clock Cycles for Completing Bus Cycles

	1bit/8 bits (byte)	16 bits (half word)	32 bits (word)
On-chip ROM	2 clk (fsys): operand	2 clk (fsys): operand (1 clk (fsys): instruction)	2 clk (fsys): operand (1 clk (fsys): instruction)
On-chip RAM	1 clk (fsys)	1 clk (fsys)	1 clk (fsys)
G-bus (CG/IRC/DMAC)	CPU: 3 to 4 clk (fsys) DMAC: 4 clk (fsys)	CPU: 3 to 4 clk (fsys) DMAC: 4 clk (fsys)	CPU: 3 to 4 clk (fsys) DMAC: 4 clk (fsys)
IM-bus (I/O registers other than G-bus) (IMCLK: 28 MHz)	CPU: 4 to 5 clk (IMCLK) DMAC: 4 to 5 clk (IMCLK)	CPU: 4 to 5 clk (IMCLK) DMAC: 4 to 5 clk (IMCLK)	CPU: 4 to 5 clk (IMCLK) DMAC: 4 to 5 clk (IMCLK)

Not Recommended for New Design

3.4.2 When Using Instructions Executed Asynchronous to Bus Cycles

Table 3.4.2 lists the co-processor and special-purpose instructions that are executed independent of bus cycles to enable and disable interrupts and to enter standby mode.

Table 3.4.2 State Transition Instructions Not Requiring Bus Cycles

	Operation
EI	Interrupts are enabled 2 clock cycles after the EI instruction is executed (E stage).
DI	Interrupts are disabled immediately after the DI instruction is executed (E stage). (The status change is reflected in the CP0 register after 2 clock cycles).
MTC0	Writes to the CP0 registers take effect 2 clock cycles after the MTC0 instruction is executed (E stage). (Only the interrupt disable setting takes effect immediately.)
WAIT	Standby mode is entered 2 clock cycles after the WAIT instruction is executed.

To execute these instructions, caution must be exercised on preceding bus cycles. The following examples show possible problems.

```
Example 1: Enabling interrupts after clearing an interrupt source
(Problem example)
lui     r27, hi(ICLR)
sh      r26, lo(ICLR)(r27)      ; Clear interrupt source.
mtc0    r29, IER                ; Enable interrupts.
nop
nop                             ; Interrupts are actually enabled.
```

In the above example, the MTC0 instruction may be executed before the preceding bus cycle is completed so that interrupts are enabled before the interrupt source is cleared as intended. This problem can be avoided by inserting the SYNC instruction before the MTC0 instruction, as shown below.

```
(Workaround example)
lui     r27, hi(ICLR)
sh      r26, lo(ICLR)(r27)      ; Clear interrupt source.
sync                            ; Stall the next instruction until the interrupt source is cleared.
mtc0    r29, IER                ; Enable interrupts.
nop
nop                             ; Interrupts are actually enabled.
```

Not Recommended for New Design

Example 2: Exiting standby mode

(Problem example)

```
ori     r26, r0 , 0x0d
lui     r27, hi(TB0RUN)
sb      r26, lo(TB0RUN)(r27)        ; Bit 0(TRUN) = 1(timer start)
wait                                ; Enter standby mode.
nop
```

This is an example of exiting standby mode when the timer reaches the specified time. If the WAIT instruction is executed before the preceding bus cycle is completed, standby mode may be entered before the timer is set, making it impossible to exit standby mode. This problem can be avoided by inserting the SYNC instruction before the WAIT instruction so that the WAIT instruction is stalled until the timer starts counting, as shown below.

(Workaround example)

```
ori     r26, r0 , 0x0d
lui     r27, hi(TB0RUN)
sb      r26, lo(TB0RUN)(r27)        ; Bit 0(TRUN)=1 (timer start)
sync                                ; Stall until the timer starts counting.
wait                                ; Enter standby mode.
nop
```

Generally speaking, it is not possible to predict when a bus cycle completes. Therefore, we do not recommend using the NOP instruction instead of the SYNC instruction in the above examples for waiting for completion of the preceding bus cycle.

Not Recommended for New Design

3.4.3 When an Memory Area Is Modified

Is it also necessary to exercise caution on bus cycles when a memory area is modified through the ROM correction function or an external bus interface. The following shows an example of execution entering an area that is modified by ROM correction immediately after the ROM correction setting has been made.

Note: The TMP19A71 does not contain an external bus interface.

Example 3: Executing the ROM correction target area after the ROM correction setting has been made

(Problem example)

```
lui     r26, hi(NG_AREA)
addiu   r26, r26, lo(NG_AREA)       ; Set the address of NG_AREA to be replaced.
lui     r27, hi(ADDREG0)
sw      r26, lo(ADDREG0)(r27)       ; Replace NG_AREA with 0xFFFFBF00-.
NG_AREA:                            ; Replaced area
nop
nop
```

In the above example, execution enters the memory area to be replaced immediately after the ROM correction setting is made. Although instructions are executed sequentially here, this situation may also occur with a jump or branch instruction. It is not normally possible to know in advance the area to be replaced with the ROM correction function. Therefore, the SYNC instruction should be inserted after an instruction for setting ROM correction. In this way, the area to be replaced with the ROM correction function will not be executed until the relevant processing is completed.

(Workaround example)

```
lui     r26, hi(NG_AREA)
addiu   r26, r26, lo(NG_AREA)       ; Set the address of NG_AREA to be replaced.
lui     r27, hi(ADDREG0)
sw      r26, lo(ADDREG0)(r27)       ; Replace NG_AREA with 0xFFFFBF00-.
sync                                ; Stall until ROM correction setting is completed.
NG_AREA:                            ; Replaced area
nop
nop
```

Not Recommended for New Design

3.4.4 When the SYNC Instruction Is Invalidated by an Interrupt

Even if the SYNC instruction is inserted to prevent possible problems as described in the above examples, the SYNC instruction may be invalidated by an interrupt. The following shows such a case occurring in the above example 2 (exiting standby mode).

Example 4: An interrupt invalidating the SYNC instruction

```
(Problem example)
ori     r26, r0, 0x0d
lui     r27, hi(TB0RUN)
sb      r26, lo(TB0RUN)(r27)        ; Bit0 (TRUN) = 1 (timer start)
sync                                ; Stall until the timer starts counting.
; Omitted                           ; <---An interrupt occurs here. ----
lui     r27, hi(TB0RUN)
lb      r26, lo(TB0RUN)(r27)        ; Save TB0RUN on the stack.
sb      r0 , lo(TB0RUN)(r27)        ; Bit 0 (TRUN) = 0 (timer stop)
 (Required processing)
sb      r26, lo(TB0RUN)(r27)        ; Restore TB0RUN (timer restart).
ERET
; Omitted                           ; <---End of interrupt service routine----
wait                                ; Enter standby mode
nop
```

This problem can be avoided by inserting the SYNC instruction at the end of the interrupt service routine (immediately before the ERET instruction).

Not Recommended for New Design

(Workaround example)

```
ori     r26, r0, 0x0d
lui     r27, hi(TB0RUN)
sb      r26, lo(TB0RUN)(r27)        ; Bit0 (TRUN) = 1 (timer start)
sync                                ; Stall until the timer starts counting.
; Omitted                           ; <---An interrupt occurs here. ----
lui r27, hi(TB0RUN)
lb      r26, lo(TB0RUN)(r27)        ; Save TB0RUN on the stack.
sb      r0 , lo(TB0RUN)(r27)        ; Bit 0 (TRUN) = 0 (timer stop)
 (Required processing)
sb      r26, lo(TB0RUN)(r27)        ; Restore TB0RUN (timer restart).
sync                                ; Stall until the bus cycle of interrupt service routine completes.
; Omitted                           ; <---End of interrupt service routine----
wait                                ; Enter standby mode.
nop
```

Not Recommended for New Design

3.4.5 Write Buffer

3.4.5.1 TMP19A71 Write Buffer

The TMP19A71 contains a four-entry FIFO write buffer. Each pipeline stage is basically executed in a single clock cycle. However, a write bus cycle accessing an area other than on-chip memory may require more than one clock cycle. The write buffer is provided to accommodate such speed variations so that program execution can achieve higher performance.

With the TMP19A71 write buffer, a read bus cycle (load instruction) is always stalled until the write buffer becomes empty regardless of the addresses to be accessed by store and load instructions (see Figure 3.4.1). Therefore, bus cycles are always generated in accordance with the program execution sequence.

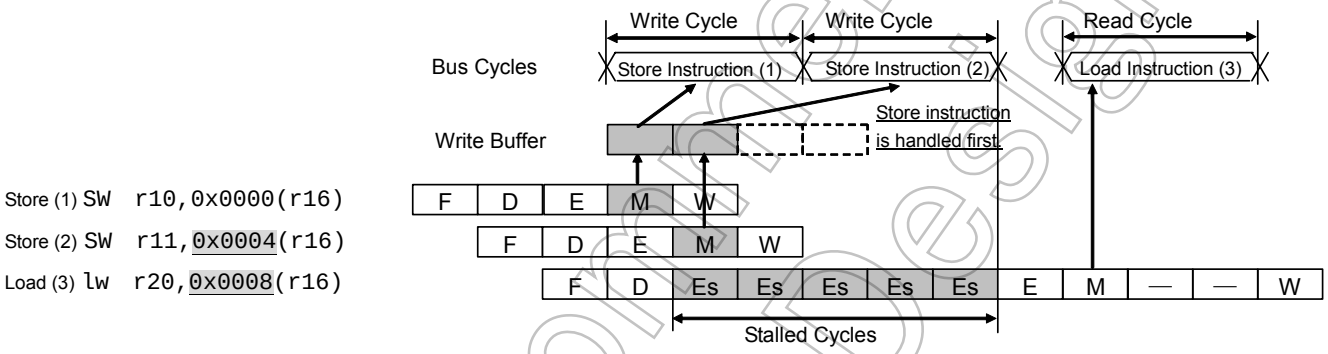


Figure 3.4.1 TMP19A71 Write Buffer Operation

Not Recommended for New Design

3.4.5.2 TMP19A70 Write Buffer (For Reference)

With the TMP19A70 write buffer, a load instruction may be executed before the immediately preceding store instruction. In an example shown in Figure 3.4.2, the target address of the third load instruction is different from the target address of the second store instruction that is queued up in the write buffer. In this case, the read bus cycle of the load instruction is processed before the write bus cycle of the store instruction in the write buffer. (If the second and third instructions have the same target address, the load instruction is stalled until the store instruction is completed.)

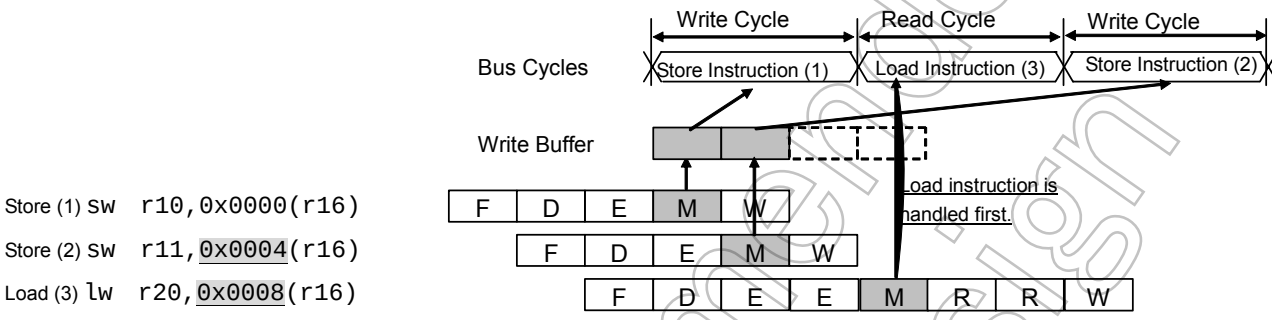


Figure 3.4.2 TMP19A70 Write Buffer Operation (with Different Target Addresses)

The following example shows a possible problem case with the TMP19A70 write buffer for reference.

```
Example: Reading Port 0 (TMP19A70)
(Problem example)
sb      r0 , P0IER      ; Enable Port 0 input.
lb      r10, P0D        ; Read Port 0.
```

In this example, the write buffer may cause the instruction for reading Port 0 to be executed before Port 0 is enabled. If this happens, the port output value will be read from Port 0. This problem can be avoided by inserting the SYNC instruction before the load instruction, as shown below, to stall the load instruction until Port 0 input is enabled.

```
(Workaround example)
sb      r0 , P0IER      ; Enable Port 0 input.
sync                    ; Stall until Port 0 input is enabled.
lb      r10, P0D        ; Read Port 0.
```

Not Recommended for New Design

3.4.6 Limitations on Accessing Special-Function Registers (SFRs)

Read-modify or read-modify-write instructions must be used with caution on SFRs that include write-only bits or bits that are cleared by a read.

3.4.6.1 SFRs Requiring Extra Caution

(1) Registers including write-only bits

If a read-modify-write instruction is executed on a register including write-only bits with undefined read values, the write operation may not be performed as expected because the value read from each write-only bit cannot be guaranteed.

(2) Registers including bits cleared by a read

If a read-modify or read-modify-write instruction is executed on a register including bits that are cleared by a read, the read operation may unintentionally clear these bits.

SFRs requiring extra caution are listed in the table below.

Table 3.4.3 SFRs Requiring Extra Caution

Functional Unit	Register	Write-Only Bits	Bits Cleared by Read
CG	CLKACT	Included	Not included
	CLKSPD	Included	Not included
IRC	ILEV	Included	Not included
	ICLR	Included	Not included
DMAC	DCR	Included	Not included
	CCRn	Included	Not included
TMRB	TBnMOD	Included	Not included
SIO	SCnMOD2	Included	Not included
	SCnCR	Not included	Included
	SCnBUF	Included	Not included
	SCnFRC	Included	Not included
	SCnFTC	Included	Not included
ADC	ADNRESn	Not included	Included
	ADCHPRn	Not included	Included
	ADPRES0	Not included	Included
PMD	EMGRELn	Included	Not included
	EMGCRn	Included	Not included
ABZ encoder	ENTNCR	Included	Not included
WDT	WDCR	Included	Not included
Flash	SEQMOD	Included	Not included

Not Recommended for New Design

3.4.6.2 Bit Manipulation Instructions Requiring Extra Caution

The bit manipulation instructions listed in the table below are read-modify or read-modify-write instructions that must not be used on the SFRs listed in Table 3.4.3. If these instructions are used to access the said SFRs, unexpected operation may result.

Table 3.4.4 Read-Modify/Read-Modify-Write Instructions

Instruction Name	Access Length	Operation Type
Bit Test (BTST)	8 bits	Read → Modify
Bit Extract (BEXT)	8 bits	Read → Modify
Bit Clear (BCLR)	8 bits	Read → Modify → Write
Bit Set (BSET)	8 bits	Read → Modify → Write
Bit Insert (BINS)	8 bits	Read → Modify → Write
Add Immediate to Memory Word (ADDMIU)	32 bits	Read → Modify → Write

3.4.6.3 Considerations for Access Length Discrepancy

The TX19A core handles bit manipulation instructions by using the access length shown in Table 3.4.4 and internally realizing 1-bit accesses in a pseudo manner. Therefore, if bit manipulation instructions are used on the SFRs shown in Table 3.4.3, the correct results may not be obtained.

This problem can be avoided by using the _rbi modifier that is provided in Toshiba's C compiler for inhibiting bit manipulation instructions. For details, refer to the instruction manual of the C compiler.

3.4.6.4 Considerations for Using the C Compiler

If bit fields are used in the SFRs shown in Table 3.4.3, the C compiler may generate bit manipulation instructions or read-modify or read-modify-write instructions of 8-bit or larger quantity.

Toshiba's C compiler provides the _rbi modifier that can be used for inhibiting bit manipulation instructions on specified SFRs. For details, refer to the instruction manual of the C compiler.

Not Recommended for New Design

4. Memory Map

Figure 4.1.1 shows memory assignment for the TMP19A71.

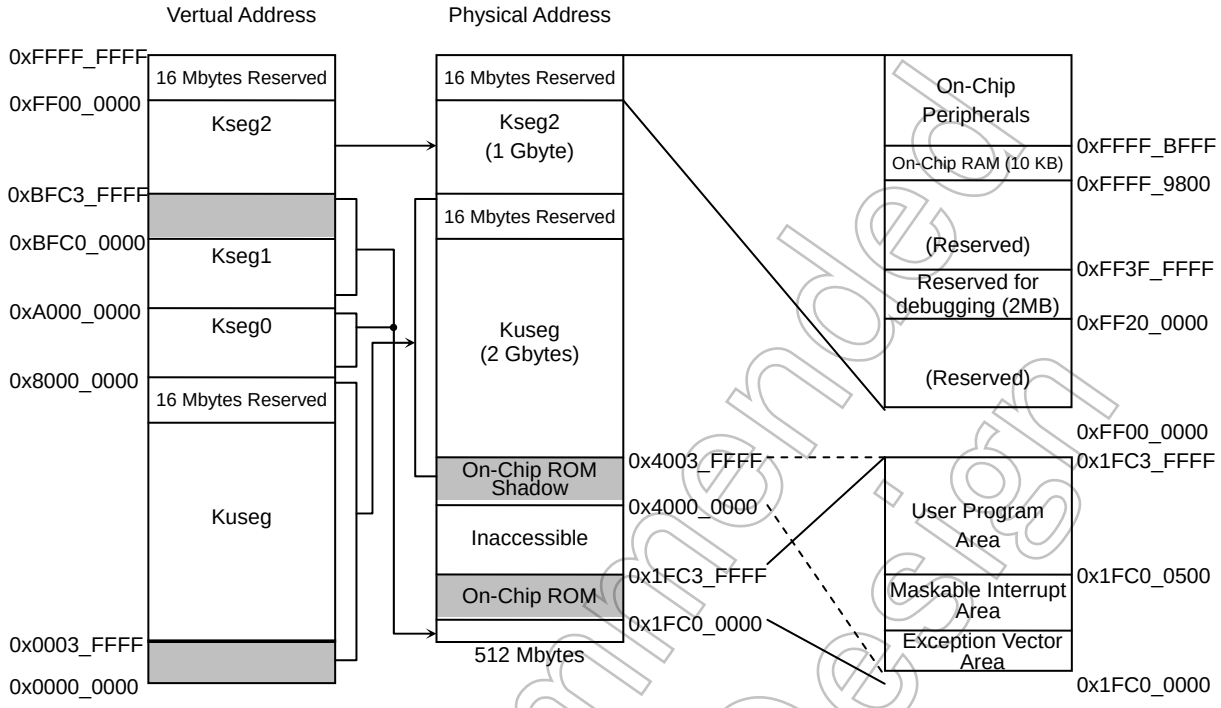


Figure 4.1.1 Memory Map

Note 1: The on-chip 256-Kbyte ROM is mapped to virtual addresses from 0x0000_0000 through 0x0003_FFFF or 0xBFC0_0000 through 0xBFC3_FFFF.
The on-chip 10-Kbyte RAM is mapped to virtual addresses from 0xFFFF_9800 through 0xFFFF_BFFF.

Note 2: Since the physical address space from 0xFFFF_4000 through 0XFFFF_BFFF is reserved as the RAM area, do not access the region except that within which RAM is located.

Note 3: The on-chip ROM is located in a linear address space beginning at physical address 0x0000_0000 or 0xBFC0_0000. All types of exceptions are vectored to the on-chip ROM when the BEV bit of the System Control Coprocessor's Status register is set to the default value of 1. (When BEV = 0, not all exception vectors reside in contiguous locations.) When external memory is used, the BEV bit can be cleared to 0.

Using the 0x0000_0000 ± 32KB virtual address space helps to improve code efficiency. The virtual address space beginning at 0x0000_0000 is a shadow of the on-chip memory beginning at 0xBFC0_0000, and references to this space are rerouted to the on-chip ROM.

Examples: 32-bit ISA

- **Accessing the 0x0000_0000 ± 32KB space**

```
LW      r2, lo (_t) (r0)        ; (r2)←Data of 0x0000_xxxx
          ↑
        Accessed with a single instruction
```

- **Accessing other locations**

```
LUI     r3, hi (_f)             ; ← Upper 16 bits of address are loaded into r3.
LW      r2, lo (_f) (r3)        ; Lower 16 bits of address must be added to upper 16 bits.
```

Note 4: No instruction should be placed in the last four words of the physical address space because the instruction prefetch circuit will access a location beyond the on-chip ROM area.
- **0xBFC3_FFF0 through 0xBFC3_FFFF of 256-Kbyte on-chip ROM**

Note 5: The TMP19A71 is always operated in the Kernal mode. The User mode should not be used.

Not Recommended for New Design

5. Clock/Standby Control

5.1 Standby Control

The TMP19A71 provides support for several levels of power reduction. While in NORMAL mode, setting the RP bit in the System Control Coprocessor (CP0)'s Status register and then executing the WAIT instruction cause the TMP19A71 to enter one of the standby modes—IDLE (Halt, Doze) or STOP—as specified by the SS field of the CLKSPD register.

The characteristics of IDLE and STOP modes are as follows:

IDLE: In IDLE mode, the TX19A core processor stops.

IDLE mode can be exited by a hardware interrupt, a nonmaskable interrupt (NMI) or a reset. The latter two include those triggered by the watchdog timer. If the level of a wakeup interrupt set in the ILxx field of the IMRxx register is lower than the mask level set in the CMASK field of the ILEV register, the TMP19A71 does not wake up from IDLE mode. If the interrupt level is higher than the mask level, the TMP19A71 returns to NORMAL mode and then services the interrupt.

Note 1: In Halt mode, the TMP19A71 freezes the TX19A core processor, preserving the pipeline state. In Halt mode, the TMP19A71 ignores any external bus requests; so it continues to assume bus mastership.

Note 2: In Doze mode, the TMP19A71 freezes the TX19A core processor, preserving the pipeline state. In Doze mode, the TMP19A71 recognizes external bus requests.

STOP: In STOP mode, the whole TMP19A71 stops.

STOP mode can be exited by INT0 to INT3, NMI or a reset. The latter two do not include those triggered by the watchdog timer.

When INT0 to INT3 are used for waking up from STOP mode, set CLKW0.W0WE = 1 for INT0 and CLKINTx.IxKI = 1 for INT1 to INT3. If one of these interrupts occurs and the interrupt level set in the IMRxx.ILxx field is higher than the mask level set in the ILEV.CMASK field, the TMP19A71 returns to NORMAL mode and then services the interrupt.

The interrupt level of INT0 to INT3, when used for exiting STOP mode, should be set to a value higher than the mask level.

Not Recommended for New Design

(1) TMP19A71 operation in NORMAL and standby modes

Table 5.1.1 TMP19A71 Operation in NORMAL and Standby Modes

Operating Mode	Operating Status
NORMAL	The TX19A core processor and on-chip peripherals operate at frequencies specified in the CG block.
IDLE (Halt)	The processor and DMAC operations stop; other on-chip peripherals are active.
IDLE (Doze)	The processor stops; on-chip peripherals including DMAC are active.
STOP	All processor and peripheral operations stop completely.

(2) Clock generation operation in NORMAL and standby modes

Table 5.1.2 Block Generation Operation in NORMAL and Standby Modes

Clock Source	Mode	Oscillator	Clock Supply to Peripherals	Clock Supply to CPU
External Crystal	NORMAL	On	On	On
	IDLE (Halt)	On	On	Off
	IDLE (Doze)	On	On	Off
	STOP	Off	Off	Off

On: Operating, or clock supplied

Off: Stopped, or clock not supplied

(3) Processor and peripheral block operation in standby modes

Table 5.1.3 Processor and Peripheral Blocks in Standby Modes

Circuit Block	Clock Source	IDLE (Doze)	IDLE (Halt)	STOP
TX19A Processor Core	fsys	Off	Off	Off
DMAC		On	Off	Off
INTC		On	On	Off (Note 1)
CG		On	On	Off (Note 1)
WDT	IMCLK	On	On	Off (Note 2)
I/O Ports		On	On	On (Note 3)

Note 1: In STOP mode, clock supply is stopped but INT0 to INT3 can be used to wake up from STOP mode. After STOP mode is exited, the INTC accepts the interrupt request.

Note 2: The WDT stops operating in STOP mode. The WDT counter value is not cleared after STOP mode is exited.

Note 3: I/O ports are not automatically disabled upon entering IDLE or STOP mode. To reduce power consumption, I/O ports should be disabled before entering IDLE or STOP mode.

Not Recommended for New Design

5.2 Clock Source Block Diagram

5.2.1 Block Diagram

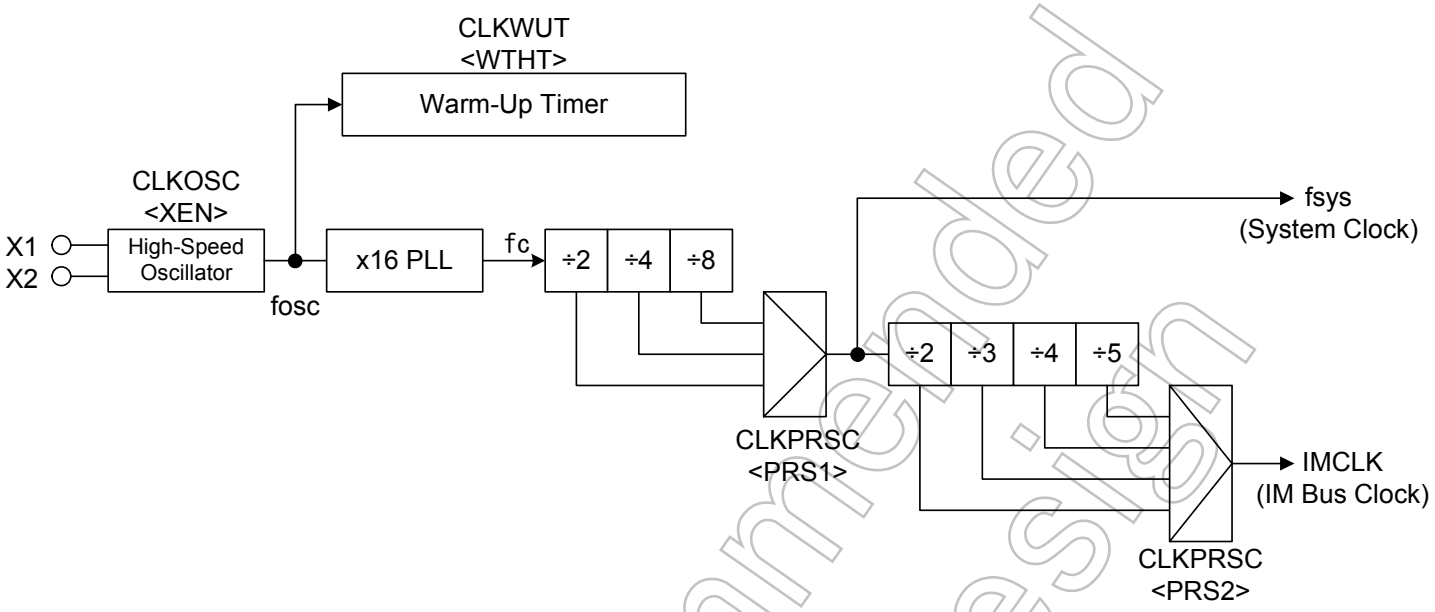


Figure 5.2.1 Clock Source Block Diagram

5.3 Clock Generator (CG) Registers

5.3.1 Register Map

Table 5.3.1 shows the register map of the clock generator. All registers other than the CLKACT are 8 bits wide, but registers at consecutive addresses can be accessed as a 16- or 32-bit quantity. When accessing more than one register at a time, be careful not to include any reserved area. For information about reserved areas, see "18. I/O Register Summary".

Table 5.3.1 Clock Generator Registers

Address	Number of Bits	Mnemonic	Register Name
0xFFFF_D300	16	CLKACT	Clock generator activate register
0xFFFF_D304	8	CLKOSC	Oscillator setting register
0xFFFF_D305	8	CLKWUT	Warm-up setting register
0xFFFF_D306	8	CLKSPD	Mode switch register
0xFFFF_D307	8	CLKPRSC	Clock gear control register
0xFFFF_D30D	8	CLKMISC	Clock generator setting register
0xFFFF_D310	8	CLKNMI	NMI setting register
0xFFFF_D312	8	CLKW0	INT0 setting register 0
0xFFFF_D31A	8	CLKINT0	INT0 setting register 1
0xFFFF_D31B	8	CLKINT1	INT1 setting register
0xFFFF_D31C	8	CLKINT2	INT2 setting register
0xFFFF_D31D	8	CLKINT3	INT3 setting register

Note: The settings made in these CG registers take effect by writing 0x5A5A and then 0xF0F0 consecutively in the CLKACT register within 64 system clock cycles after the settings are made. It this time limit is not observed, the settings will not take effect.

Not Recommended for New Design

5.3.2 Register Description

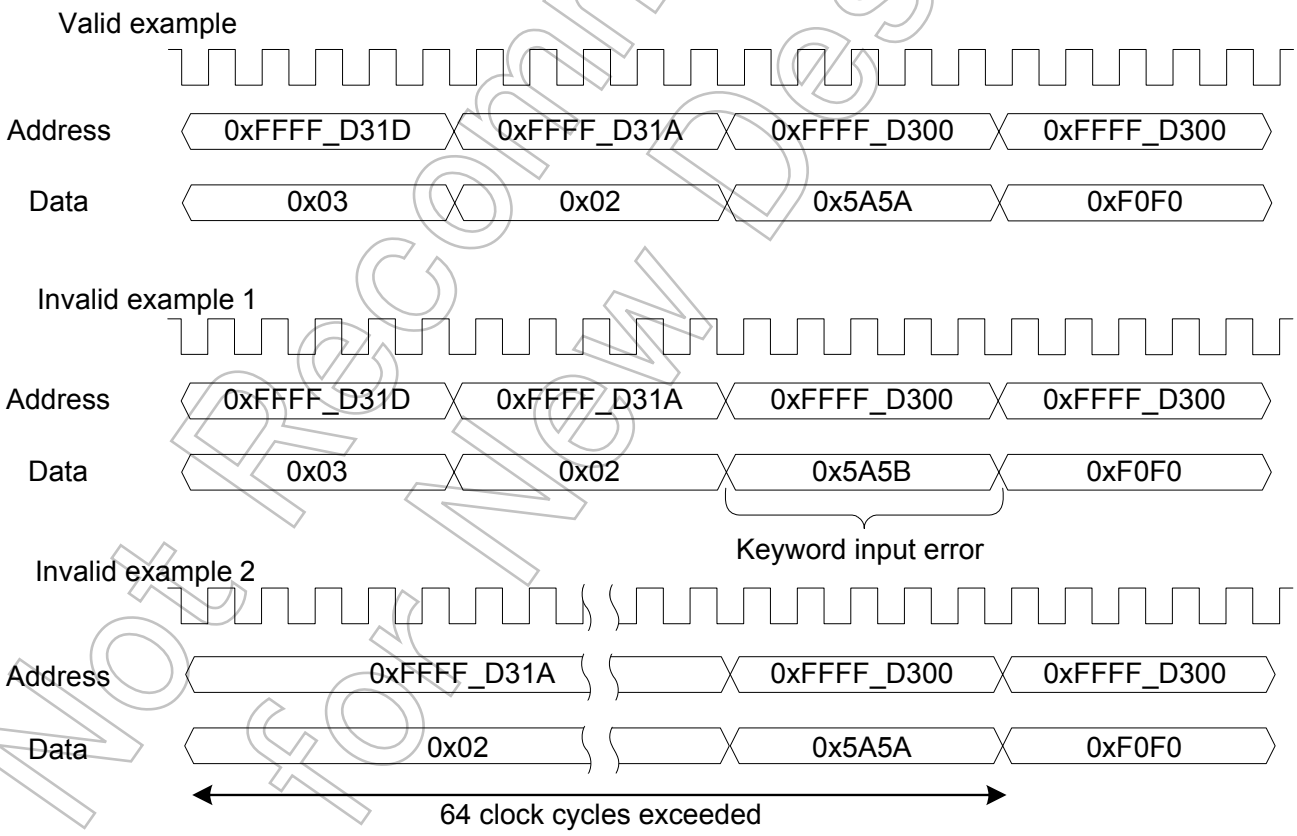
Clock Generator Activate Register

CLKACT (0xFFFF_D300)

	7	6	5	4	3	2	1	0
Bit Symbol	ACT							
Read/Write	W							
Reset Value	0	0	0	0	0	0	0	0
Function								
	15	14	13	12	11	10	9	8
Bit Symbol	ACT							
Read/Write	W							
Reset Value	0	0	0	0	0	0	0	0
Function	The settings made in the CG registers take effect by writing 0x5A5A and then 0xF0F0 consecutively in this register within 64 system clock cycles after the settings are made.							

Note 1: This register must be accessed as a 16-bit quantity; bit manipulation instructions cannot be used.

Note 2: The settings made in the CG registers take effect by writing 0x5A5A and then 0xF0F0 consecutively in this register within 64 system clock cycles after the settings are made. If this time limit is not observed, the settings will not take effect.



Not Recommended for New Design

Figure 5.3.1 Example of How to Use the Clock Generator Activate Register

Oscillator Setting Register

CLKOSC (0xFFFF_D304)

	7	6	5	4	3	2	1	0
Bit Symbol	XEN	–	RXEN	–	DRVH	–	–	–
Read/Write	R/W							
Reset Value	1	0	1	0	0	0	0	0
Function	Oscillator 0: Disable 1: Enable	Must be set to 0.	Oscillator after exiting STOP mode 0: Disable 1: Enable	Must be set to 0.	Oscillator AMP capability 0:Normal 1: Low	Must be set to 0.		

Warm-Up Setting Register

CLKWUT (0xFFFF_D305)

	7	6	5	4	3	2	1	0
Bit Symbol	WTHD	WTHW	WTHT		–	–	–	–
Read/Write	R	R/W	R/W		R	R/W		
Reset Value	1	1	11		0	1	1	1
Function	Warm-up end flag 0: Warming up 1: Complete	Warm-up operation enable 0: No warm-up 1: Enable warm-up operation	Oscillator warm-up time 00:2^8 clock cycles 01:2^12 clock cycles 10:2^14 clock cycles 11:2^16 clock cycles					

Note 1: The warm-up time set in the WTHT field is counted using the fosc clock.

Note 2: When the WTHW bit is set to 1, the warm-up time set in the WTHT field is automatically inserted before clock oscillation is started. At power-on, if a reset state is released without waiting for 2^16 clock cycles, the internal circuits may not be initialized properly.

Note 3: During the warm-up period, no clock is supplied to the internal circuits.

Not Recommended for New Design

Mode Switch Register

CLKSPD (0xFFFF_D306)

	7	6	5	4	3	2	1	0
Bit Symbol	—	SS		—	—	—	—	—
Read/Write	R/W	W		R/W		R		
Reset Value	1	00		1	0	0	0	0
Function	Must be set to 1.	Standby mode (Note 1) 00: NORMAL mode 01: STOP mode 10: Reserved 11: IDLE (Halt) mode		Must be set to 1.	Must be set to 0.			

Note 1: The CLKSPD.SS field selects the standby mode in combination with the RP bit of CP0's Status register, as shown in the table below. The X mark indicates that the WAIT instruction cannot be used in that mode.

	CLKSPD.SS	Halt RP=0	Doze RP=1
NORMAL	00	X	X
STOP	01	STOP	X
Reserved	10	X	X
IDLE	11	Halt	Doze

Note 2: Each time the TMP19A71 is placed in a standby mode, set the CLKSPD.SS field before executing the WAIT instruction. The WAIT instruction should not be executed successively.

Note 3: To set the CLKSPD.SS field to a value other than 00, be sure to set 0x5A5A and 0xF0F0 to the CLKACT register exclusively to enable the CLKSPD.SS setting. If other clock generator registers are set at the same time, the settings may not be reflected correctly.

Note 4: This register does not support bit manipulation instructions.

Not Recommended for New Design

Clock Gear Control Register

CLKPRSC (0xFFFF_D307)	7	6	5	4	3	2	1	0
Bit Symbol	PRS1		PRS2			—	—	—
Read/Write	R/W					R		
Reset Value	00		000			0	0	0
Function	System clock (fsys) 00: 1/2 frequency 01: 1/4 frequency 10: 1/8 frequency 11:Reserved		IMCLK clock 000: 1/2 frequency 010: 1/3 frequency 100: 1/4 frequency 110: 1/5 frequency Others: Reserved					

Note: Before changing the system clock setting, make sure that all peripheral functions are stopped.

5.3.3 Interrupt Registers

NMI Setting Register

CLKNMI (0xFFFF_D310)	7	6	5	4	3	2	1	0
Bit Symbol	NMISEN		—	—	—	—	—	NMIBE
Read/Write	R/W		—	—	—	—	—	R
Reset Value	00		0	0	0	0	0	0
Function	NMI sensitivity 00: Prohibited 11: Both edges 01: Rising edge 10: Falling edge							CLKNMI setting enable 0: Enable 1: Disable

Note 1: Setting this register causes the NMIBE bit to be set to 1, disabling any subsequent writes to this register until a reset is applied.

Note 2: To use NMI, appropriate settings must be made in the relevant port registers. For details, see "8. I/O Ports".

INT0 Setting Register 0

CLKW0 (0xFFFF_D312)	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	W0WE	—	—	—
Read/Write	R/W					R		
Reset Value	0	0	0	0	0	0	0	0
Function	Must be set to 0.	Must be set to 0.	Must be set to 0.	Must be set to 0.	INT0 interrupt type 0: Typical interrupt 1: Wake-up signaling			

Note: The W0WE bit must be set to 1 to use INT0 as the wake-up signaling to take the TMP19A71 out of STOP mode.

Not Recommended for New Design

INT0 Setting Register 1

CLKINT0 (0xFFFF_D31A)

	7	6	5	4	3	2	1	0
Bit Symbol	I0SEN			—	—	—	—	—
Read/Write	R/W			R				
Reset Value	000			0	0	0	0	0
Function	INT0 sensitivity 001: Rising edge 010: Falling edge 011: Both edges 101: High level 110: Low level Others: Disable							

INT1 Setting Register

CLKINT1 (0xFFFF_D31B)

	7	6	5	4	3	2	1	0
Bit Symbol	I1SEN			—	I1KI	—	—	—
Read/Write	R/W			R	R/W	R		
Reset Value	000			0	0	0	0	0
Function	INT1 sensitivity 001: Rising edge 010: Falling edge 011: Both edges 101: High level 110: Low level Others: Disable				INT1 interrupt type 0: Typical interrupt 1: Wake-up signaling			

Note: The I1KI bit must be set to 1 to use INT1 as the wake-up signaling to take the TMP19A71 out of STOP mode.

INT2 Setting Register

CLKINT2 (0xFFFF_D31C)

	7	6	5	4	3	2	1	0
Bit Symbol	I2SEN			—	I2KI	—	—	—
Read/Write	R/W			R	R/W	R		
Reset Value	000			0	0	0	0	0
Function	INT2 sensitivity 001: Rising edge 010: Falling edge 011: Both edges 101: High level 110: Low level Others: Disable				INT2 interrupt type 0: Typical interrupt 1: Wake-up signaling			

Note: The I2KI bit must be set to 1 to use INT2 as the wake-up signaling to take the TMP19A71 out of STOP mode.

Not Recommended for New Design

INT3 Setting Register

CLKINT3 (0xFFFF_D31D)

	7	6	5	4	3	2	1	0
Bit Symbol	I3SEN			–	I3KI	–	–	–
Read/Write	R/W			R	R/W	R		
Reset Value	000			0	0	0	0	0
Function	INT3 sensitivity 001: Rising edge 010: Falling edge 011: Both edges 101: High level 110: Low level Others: Disable				INT3 interrupt type 0: Typical interrupt 1: Wake-up signaling			

Note: The I3KI bit must be set to 1 to use INT3 as the wake-up signaling to take the TMP19A71 out of STOP mode.

Not Recommended for New Design

5.3.4 Reset Registers

Clock Generator Setting Register (Mask-Version Product)

CLKMISC (0xFFFF_D30D)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	MSWDR	—	—	MSNMI		MSBC
Read/Write	R/W					R		
Reset Value	0	0	0	0	0	00		0
Function			WDT reset flag 0: No WDT reset 1: WDT reset occurred	Must be set to 0.	Must be set to 0.	NMI source flag 00: External pin 01: WDT 10: Bus error (store)		CG access flag 0: Access enabled 1: Access disabled

Note 1: Bits 7 to 5 of the CLKMISC register are not initialized by a WDT reset; they are initialized by an external reset.

Note 2: The MSWDR bit is not initialized by a WDT reset; it is initialized by an external reset. To clear this bit after a WDT reset occurred, it must be programmed to 0.

Note 3: The MSBC bit indicates whether or not new settings can be made to the CG registers. When MSBC = 1, the settings in the CG registers are in the middle of being changed after the CLKACT register is set. The MSBC bit must be cleared to 0 before new values can be written to the CG registers.

Clock Generator Setting Register (Flash-Version Product)

CLKMISC (0xFFFF_D30D)

	7	6	5	4	3	2	1	0
Bit Symbol	MSCW	MSFR	MSWDR	—	—	MSNMI		MSBC
Read/Write	R/W					R		
Reset Value	0	0	0	0	0	00		0
Function	Reset type 0: Power-on reset 1: Normal reset	Flash reset by WDT or external reset 0: Enable 1: Disable	WDT reset flag 0: No WDT reset 1: WDT reset occurred	Must be set to 0.	Must be set to 0.	NMI source flag 00: External pin 01: WDT 10: Bus error (store)		CG access flag 0: Access enabled 1: Access disabled

Note 1: Bits 7 to 5 of the CLKMISC register are not initialized by a normal reset; they are initialized by a power-on reset.

Note 2: The MSWDR bit is not initialized by a normal reset; it is initialized only by a power-on reset. To clear this bit after a WDT reset occurred, it must be programmed to 0.

Note 3: The MSCW bit is not initialized by a normal reset; it is initialized only by a power-on reset. This bit can be used as a flag to indicate whether a power-on or normal reset occurred by programming this bit to 1 after a power-on reset. This bit is not automatically set to 1 by a normal reset.

Note 4: The MSBC bit indicates whether or not new settings can be made to the CG registers. When MSBC = 1, the settings in the CG registers are in the middle of being changed after the CLKACT register is set. The MSBC bit must be cleared to 0 before new values can be written to the CG registers.

Note 5: When the MSFR bit is set to 1, the Flash ROM is not initialized by an external or WDT reset. To program or erase the Flash ROM, this bit should be set to 0.

Not Recommended for New Design

6. Watchdog Timer (WDT)

The TMP19A71 contains a watchdog timer (WDT). The WDT is used to regain control of the system in the event of software system lockups due to spurious noises, etc. When a watchdog timer time-out occurs, the WDT generates a nonmaskable interrupt (NMI) or a reset exception to the TX19A core processor.

6.1 Operational Overview

The WDT can be programmed to generate a reset or NMI upon time-out. When NMI is selected, a reset occurs upon counter overflow.

6.1.1 Generating an NMI (WDMOD.RESCR = 0)

If the WDT counter is not cleared within the time-out period set in the WDMOD.FTP field, the WDT generates an NMI upon time-out. Then, the WDT continues counting. If the 23-bit binary counter is not cleared before it overflows (about 300 ms with IMCLK = 28 MHz), the WDT generates a reset exception. This causes the WDT to be initialized and start counting again with the default setting.

Note: After an NMI occurs, save necessary data on the stack and wait for an overflow reset.

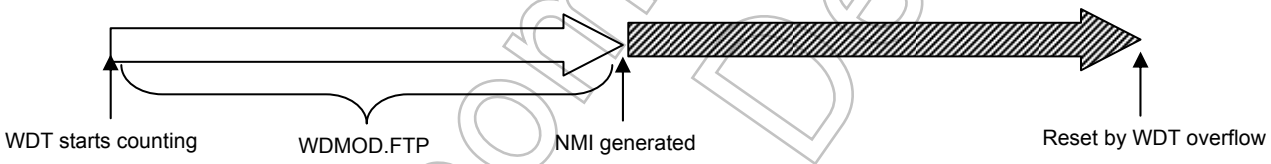


Figure 6.1.1 WDT Operation when WDMOD.RESCR=0

6.1.2 Generating a Reset (WDMOD.RESCR = 1)

If the WDT counter is not cleared within the time-out period set in the WDMOD.FTP field, the WDT generates a reset exception upon time-out. A reset exception causes the WDT to be initialized and start counting again with the default setting.

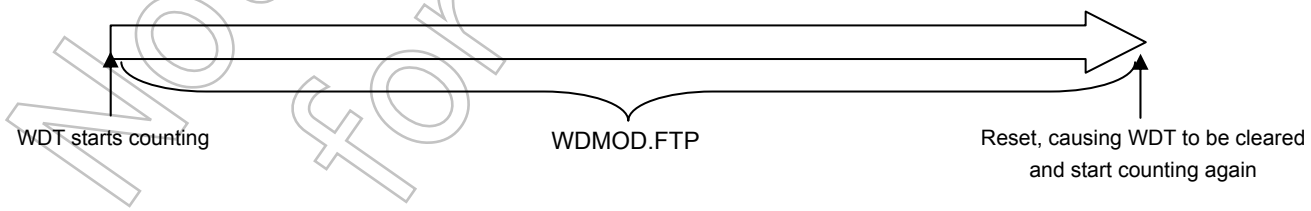


Figure 6.1.2 WDT Operation when WDMOD.RESCR=1

Not Recommended for New Design

6.2 Register Description

The WDT is controlled by two control registers (WDMOD, WDCR) and a counter (WDCNT), as shown in Table 6.2.1.

Table 6.2.1 WDT Register Map

Address	Number of Bits	Mnemonic	Register Name
0xFFFF_C830	16（8）	WDMOD (L)	Watchdog Timer Mode Register (Low)
0xFFFF_C831	8	(WDMODH)	(Watchdog Timer Mode Register High)
0xFFFF_C834	8	WDCR	Watchdog Timer Control Register
0xFFFF_C838	16	WDCNT	Watchdog Timer Count Register

Note: Although the WDMOD register is a 16-bit register, the lower 8 bits (WDMODL) and upper 8 bits (WDMODH) can be accessed separately.

6.2.1 Watchdog Timer Mode Register (WDMOD)

Watchdog Timer Mode Register

WDMOD(L) (0xFFFF_C830)

	7	6	5	4	3	2	1	0
Bit Symbol	—	FTP			—	WDEN	—	RESCR
Read/Write	R	R/W						R/W
Reset Value	0	010			0	1	0	0
Function	Can be read as 0.	Time-out period 000: 2^12 (about 0.15 ms at IMCLK=28 MHz) 001: 2^13 (about. 0.29 ms at IMCLK=28 MHZ) 010: 2^14 (about 0.59 ms at IMCLK=28 MHz) 011: 2^15 (about 1.2 ms at IMCLK=28 MHz) 100: 2^16 (about 2.3 ms at IMCLK=28 MHz) 101: 2^19 (about 18.7 ms at IMCLK=28 MHz) 110: 2^21 (about 74.9 ms at IMCLK=28 MHz) 111: 2^22 (about 150 ms at IMCLK=28 MHz)				Must be set to 0.	WDT enable 0: Disable 1: Enable	Must be set to 0.	Reset type 0: NMI upon time-out 1: Reset exception upon time-out

(WDMODH) (0xFFFF_C831)

	15	14	13	12	11	10	9	8
Bit Symbol	—	—	—	—	—	—		
Read/Write	R/W			R		R/W		
Reset Value	0	0	0	0	0	000		
Function	Must be set to 0.	Must be set to 0.	Must be set to 0.	Can be read as 0.	Can be read as 0.			

Note: Do not change bits other than the WDEN bit while the WDT is operating.

Not Recommended for New Design

(1) First time-out period (WDMOD.FTP)

This 3-bit field determines the duration of the WDT time-out interval. Upon reset, the FTP field is initialized to 010. Possible time-out intervals are shown in the register table.

(2) WDT enable (WDMOD.WDEN)

Upon reset, the WDEN bit is set to 1, enabling the WDT. To disable the WDT, the clearing of the WDEN bit must be followed by a write of a special disable code (B1H) to the WDCR register. This prevents a "lost" program from disabling the WDT operation. The WDT can be re-enabled simply by setting the WDEN bit.

(3) WDT reset (WDMOD.RESCR)

When RESCR=1, a reset exception is generated and the WDT is initialized upon WDT time-out. When RESCR=0, an NMI is generated upon WDT time-out and then a reset exception is generated upon counter overflow.

Not Recommended for New Design

6.2.2 Watchdog Timer Control Register (WDCR)

This register is used to disable the WDT and to clear the WDT binary counter.

Watchdog Timer Control Register

WDCR (0xFFFF_C834)

	7	6	5	4	3	2	1	0
Bit Symbol	—							
Read/Write	W							
Reset Value	—							
Function	B1H : WDT disable code 4E H: WDT clear-count code							

→ WDT disable and clear -count

0xB1	Disable code
0x4E	Clear-count code
Others	Invalid

Note: This register does not support bit manipulation instructions.

- Disabling the WDT

The WDT can be disabled by clearing the WDMOD.WDEN to 0 and then writing the disable code (B1H) to the WDCR register. At this time, the counter value is maintained. Before enabling the WDT again, clear the counter by writing the clear-count code (4EH).

```
WDMODL ← – – – – – 0 – –    Clear the WDEN bit to 0.
WDCR   ← 1 0 1 1 0 0 0 1    Write the disable code (B1H) to the WDCR.
```

- Enabling the WDT

The WDT can be enabled simply by setting the WDEN bit in the WDMOD to 1.

- Clearing the WDT counter

Writing the clear-count code (4EH) to the WDCR resets the binary counter to 0. The counting process begins again.

```
WDCR   ← 0 1 0 0 1 1 1 0    Write the clear-count code (4EH) to the WDCR.
```

Watchdog Counter Register

WDCNT (0xFFFF_C838)

	7	6	5	4	3	2	1	0
Bit Symbol	—							
Read/Write	R							
Reset Value	0							
Function								
	15	14	13	12	11	10	9	8
Bit Symbol	—							
Read/Write	R							
Reset Value	0							
Function	Bits 22 to 7 of the WDT counter value can be read.							

Not Recommended for New Design

7. Exceptions/Interrupts

7.1 Overview

TMP19A71 has exceptions of 15 types including nonmaskable interrupt (NMI) and 49 maskable interrupt sources as listed below.

- Gereral Exceptions
  - Reset exception
  - Nonmaskable Interrupt (NMI) exception
  - Address Error exception (Instruction Fetch)
  - Address Error exception (Load/Store)
  - Bus Error exception (Instruction Fetch)
  - Bus Error exception (Data Access)
  - Coprocessor Unusable exception
  - Reserved Instruction exception
  - Integer Overflow exception
  - Trap exception
  - System Call exception
  - Breakpoint exception
- Debug Exceptions
  - Single Step exception
  - Debug Breakpoint exception
- Interrupts
  - Maskable software interrupts (2 sources)
  - Maskable hardware interrupts (37 internal sources and 10 external sources)

TMP19A71 can process not only interrupt requests from on-chip peripheral hardware and external sources but also exceptions forcibly as measures of notification of error conditions arising in execution of general instructions.

By using the register bank called "shadow register set" newly implemented in the TX19A processor core, it is now unnecessary to save the general-purpose register (GPR) contents elsewhere upon interrupt response thus leading to very fast interrupt response.

Interrupt requests can be nested according to programmable priority of seven levels. It is also possible to mask interrupt requests of priority levels lower than the specified mask level.

Not Recommended for New Design

7.2 Exception Vectors

An exception vector address is the entry address of a routine that handles an exeption. Reset and Nonmaksable Interrupt exceptions are vectored to address 0xBFC0_0000. A debug exception is vectored to 0xBFC0_0480 when the EJTAG ProbEn signal is 0 and 0xFF20_0200 when the EJTAG ProbEn signal is 1 according to the internal signal value of ProbEn.Values of other exceptions may be various depending on the BEV bit of the Status register and the IV bit of the Cause register belonging to the system control coprocessor (CP0).

Table 7.2.1 Exception Vector Table (Virtual Addresses)

Exception Type	BEV=0	BEV=1
Reset, NMI	0xBFC0_0000	0xBFC0_0000
Debug exception (En=0)	0xBFC0_0480	0xBFC0_0480
Debug exception (En=1)	0xFF20_0200	0xFF20_0200
Interrupt (IV=0)	0x8000_0180	0xBFC0_0380
Interrupt (IV=1)	0x8000_0200	0xBFC0_0400
Other general exceptions	0x8000_0180	0xBFC0_0380

Note 1 : When exception vector addresses reside in the on-chip ROM, the BEV bit of the CP0 Status register must be set to 1. TMP19A71 has no external bus interface, so Status.BEV=0 is not allowed.

Note 2 : To assign different exception vector addresses for interrupts and other general exceptions, set the IV bit of the CP0 Cause register to 1.

7.3 Reset Exception

A Reset exception occurs when an external reset pin is driven low or the WDT counts to its reset value. As a Reset exception occurs, on-chip peripheral registers (Note 1) and CP0 registers are initialied, and a control jumps to the exception vector address 0xBFC0_0000. Upon a Reset exception, the PC value is stored in the CP0 ErrorEPC register.

When a Reset exception occurs, the ERL bit of the CP0 Status register is set to 1, disabling interrupts. To use interrupts, the ERL bit must be cleared to 0 in the startup routine (reset exception handler) or by other means.

For a detailed description of Reset exception handling, refer to the chapter Exception Handling Reset Exception in the *32-Bit TX19 System RISC TX19 Family Architecture* manual.

Note 1 : In the flash-version product, some on-chip peripheral registers are not initialized by a Reset exception; these registers are initialized only by the internal power-on reset signal that is generated at power-on.

Note 2 : In the mask-version product, some on-chip registers are not initialized by a Reset exception caused by the WDT; these registers are initialized only by a Reset exception via an external reset pin.

Not Recommended for New Design

7.4 Nonmaskable Interrupt (NMI)

A Nonmaskable Interrupt (NMI) occurs when an external NMI pin is asserted as specified by the NMISEN field of the CLKNMI register; the WDT counts to the NMI value; or the bus error area is accessed by a store access including DMA transfer when MODECR<BERCTL>=0. When a NMI occurs, the ERL and NMI bits of the CP0 Status register are set to 1 and a control jumps to the exception vector address 0xBFC0_0000.

The PC value at the time of an NMI is stored in the CP0 ErrorEPC register. However, if a bus error occurs during a store instruction, a NMI exception is generated asynchronously to the instruction execution timing and the PC is stored not at the instruction that caused the NMI but at the instruction that is being executed when the NMI is generated.

Upon NMI generation, when Shadow Register Set is enabled, SSCR <CSS> will be overwritten by the value of SSCR <PSS> but the register bank will not be switched because the value of SSCR <CSS> is not updated. The reason why only the SSCR <PSS> value is updated is because it is necessary to prevent the register bank from being changed when SSCR <PSS> is overwritten by the value of SSCR <CSS> due to an ERET instruction executed upon returning from NMI.

The cause of NMI generation can be determined by NMIFLG <WDT> and <WBER> of CG.

A reset initializes the NMI pin (P95) as a general-purposed port. To use the NMI pin, it is necessary to set the P9FR15 bit of the Port 9 Function Register 1 (P9FR1) and the NMISEN field of the CLKNMI register.

For a detailed description of NMI handling, refer to the chapter "Exception Handling Nonmaskable Interrupts" of the separate volume, *TX19A Core Architecture*.

7.5 General Exceptions (other than Reset Exception/NMI)

A general exception occurs when a specific instruction such as the SYSCALL instruction is executed or an error condition such as an illegal instruction fetch is detected. When a general exception occurs with the Status.BEV bit set to 1, control jumps to the exception vector address 0xBFC0_380. The cause of a general exception can be determined by the ExCode field of the CP0 Cause register.

The PC value at the time of a general exception is stored in the CP0 EPC register. However, a Bus Error exception (data access) is generated asynchronously to the instruction execution timing so that the PC is stored not at the instruction that caused the exception but at the instruction that is being executed when the exception is generated. Upon a general exception, when the shadow register set is enabled, SSCR <CSS> will be overwritten by the value of SSCR <PSS> but the register bank will not be switched because the value of SSCR <CSS> is not updated. The reason why only the SSCR <PSS> value is updated is because it is necessary to prevent the register bank from being changed when SSCR <PSS> is overwritten by the value of SSCR <CSS> due to an ERET instruction executed upon returning from the exception.

The illegal address that caused an Address Error exception (instruction fetch, load, store) or Bus Error exception (instruction fetch, data access) is stored in the CP0 BadVAddr register.

For a detailed description of general exception handling, refer to the chapter "Exception Handling" of the separate volume, *TX19A Core Architecture*.

Note 1 : No Address Error exception (load, store) occurs during DMA transfer. In this case, error conditions can be detected by the configuration error flag (the Conf bit of the CSRx register) in the DMAC.

Note 2 : A Bus Error exception (data access) occurs during a load instruction or a load access by DMA transfer.

Not Recommended for New Design

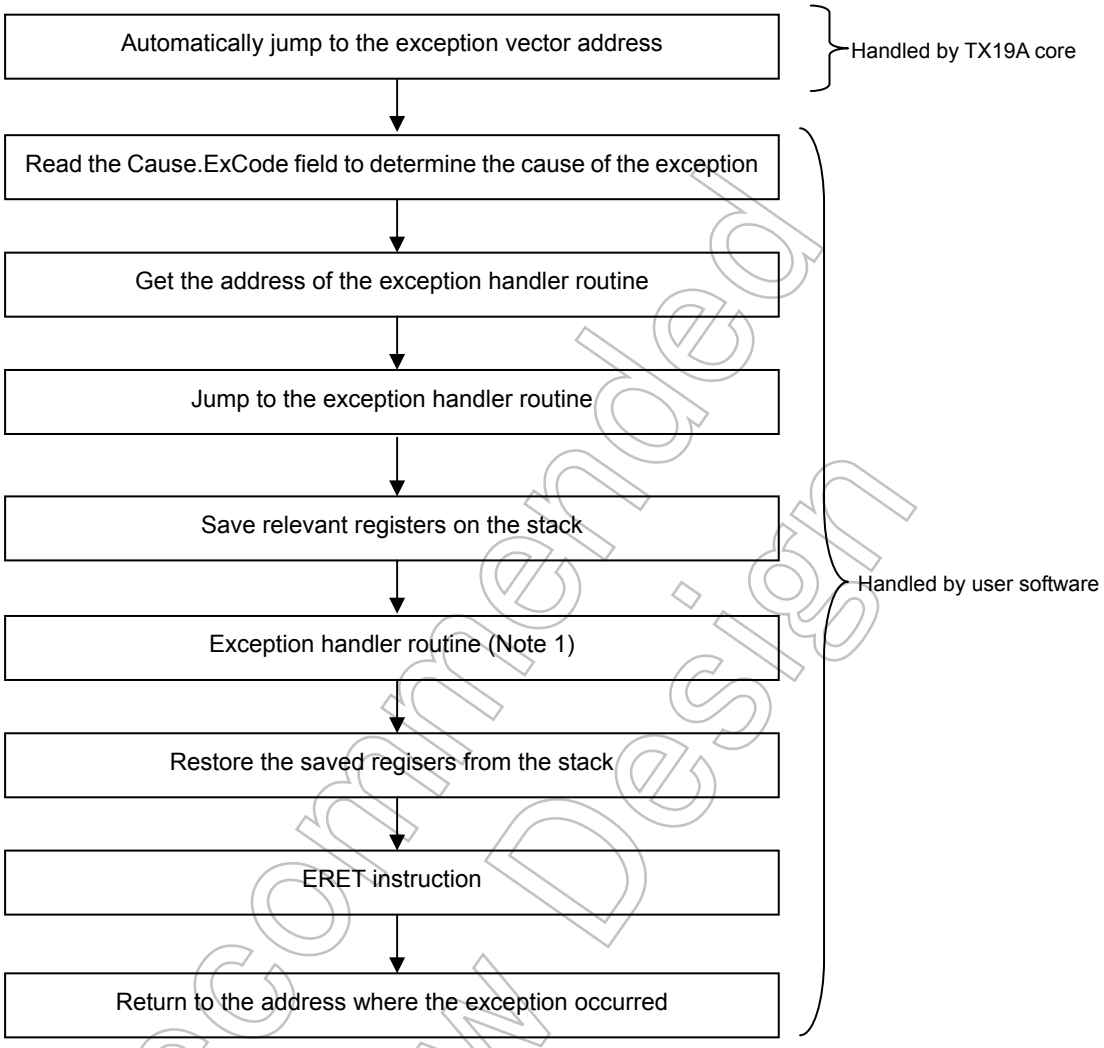


Figure 7.5.1 General Exception Operation (Exceptions other than Reset or NMI)

Note 1 : General exceptions (i.e. exceptions other than Reset exception or NMI) excluding Trap, System Call, and Breakpoint exceptions indicate error conditions; they are normally handled by a reset routine.

Note 2 : For general exceptions (i.e. exceptions other than Reset exception or NMI) excluding Bus Error exception (instruction fetch, data access), the PC value is stored in the EPC register as the instruction that caused the exception. Therefore, if the ERET instruction is executed to resume execution from the saved PC address, the same exception may occur again.

Not Recommended for New Design

7.6 Debug Exceptions

Debug exceptions include Single-step and Debug Breakpoint exceptions. These exceptions are not normally used in user programs.

Also enabling the shadow register set will not be effective in debug exceptions.

For a detailed description of debug exception handling, refer to the chapter "Exception Handling Debug Exception" of the separate volume, *TX19 Core Architecture*.

7.7 Maskable Software Interrupts

The TMP19A71 provides two sources of maskable software interrupts (hereafter referred to as software interrupts). Each software interrupt can be generated by setting the corresponding bit in the IP[1:0] field of the CP0 Cause register.

A software interrupt is accepted, at the fastest, 3 clock cycles after the IP[1:0] field of the CP0 Cause register is set.

Software interrupt requests are accepted when all the following conditions are met:

- The IM[1:0] field of the CP0 Status register is set to 1.
- The IE bit of the CP0 Status register is set to 1.
- The ERL and EXL bits of the CP0 Status register are cleared to 0.

Each software interrupt can be masked by clearing the corresponding bit in the IM[1:0] field of the CP0 Status register. If a software interrupt and a hardware interrupt occur simultaneously, the hardware interrupt is given higher priority.

Upon software interrupts, when Shadow Register Set is enabled, SSCR <CSS> will be overwritten by the value of SSCR <PSS> but the register bank will not be switched because the value of SSCR <CSS> is not updated. The reason why only the SSCR <PSS> value is updated is because it is necessary to prevent the register bank from being changed when SSCR <PSS> is overwritten by the value of SSCR <CSS> due to an ERET instruction executed upon returning from the software interrupt. Software interrupts are processed in a process flow shown in Figure 7.7.1.

Note: Software interrupts are different from Software Set interrupts which are generated as maskable hardware interrupts to be described hereinafter. A hardware interrupt generation caused by setting the EIM00 field of the IMR00 register to 01 is called Software Set.

Not Recommended for New Design

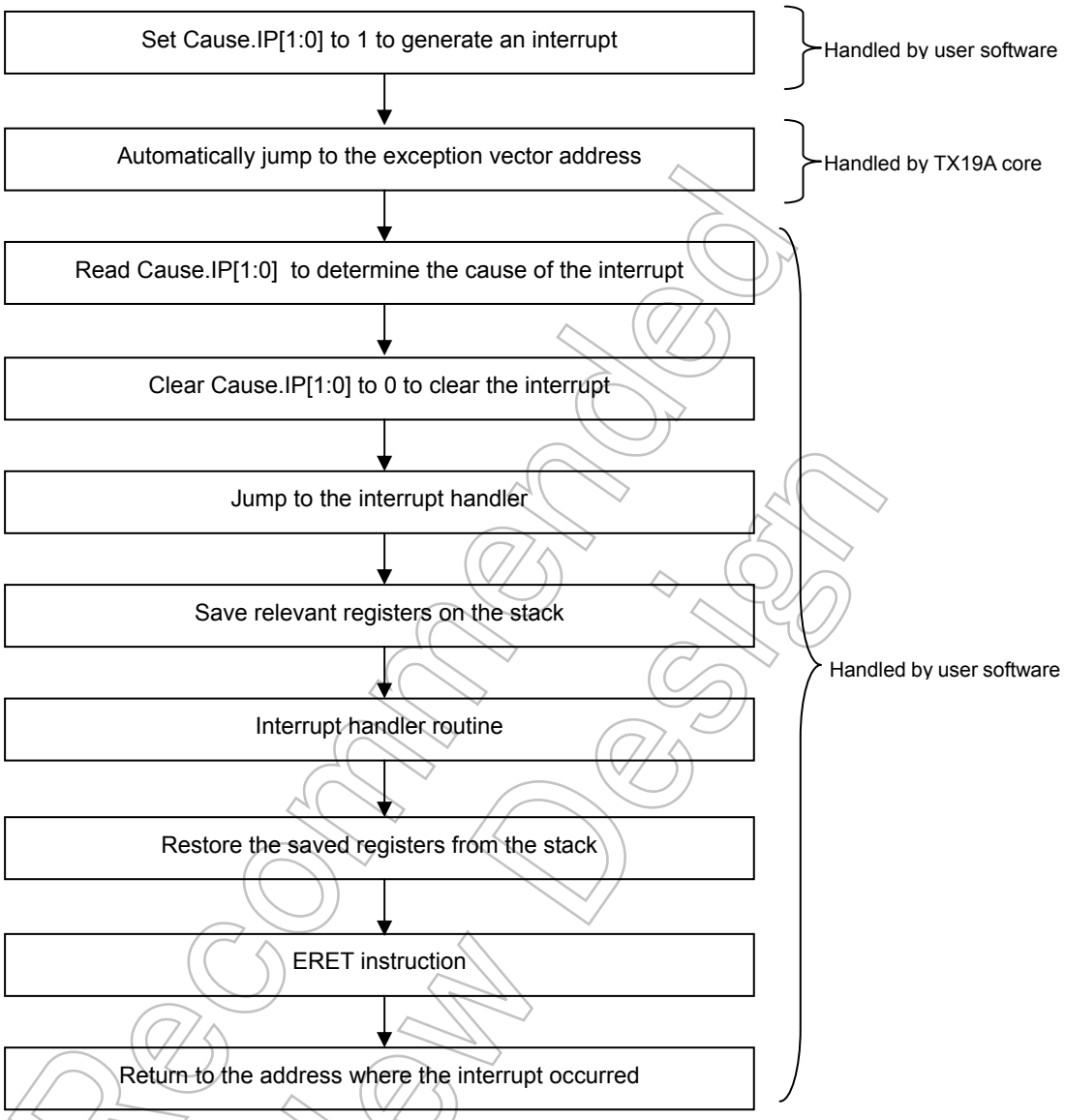


Figure 7.7.1 Example of Softwre Interrupt Operation

Note: A software interrupt is accepted, at the fastest, 3 clock cycles after the interrupt is enabled, and the PC at this moment is stored in the EPC register.

Not Recommended for New Design

7.8 Maskable Hardware Interrupts

7.8.1 Features

A maskable hardware interrupt (hereinafter referred to as hardware interrupt) is interrupt request of 47 sources that can set the seven interrupt levels of priority order individually with an interrupt controller (INTC).

Hardware interrupt requests are accepted when all the following conditions are met:

- The IM[4:2] field of the CP0 Status register is set to 1.
- The IE bit of the CP0 Status register is set to 1.
- The ERL and EXL bits of the CP0 Status register are cleared to 0.

If two or more interrupt occur simultaneously, interrupt requests are accepted according to their priority levels. If interrupt requests of the same interrupt level occur simultaneously, the interrupt is accepted in ascending order starting with that of the smallest number (see Table 7.8.1).

When a hardware interrupt request is accepted, the EXL bit of the CP0 Status register is set to 1 to disable interrupts, and the CMASK field of the ILEV register is automatically updated to the interrupt level of the accepted interrupt request. The IE bit of the CP0 Status register remains as has been set when an interrupt request is accepted.

In hardware interrupts processing, each interrupt level is associated with a register bank called Shadow Register Set. When an interrupt request is accepted, the register bank is switched to the one whose number is the same number of corresponding interrupt level. Through this mechanism, there is no need for user program to save the general-purposed register (GPR) contents elsewhere upon interrupt response, thus a faster interrupt response is ensured. To use the Shadow Register Set, the SSD bit of the CP0 SSCR register must be cleared to 0.

Once an interrupt request is accepted, further interrupt requests can be nested by clearing the EXL bit of the CP0 Status register to 0 to enable interrupts. At this time, the CMASK bit of the ILEV register of INTC is updated to the priority level whose interrupt request has been set, thus allows only interrupt requests with higher priority levels than the one it has been accepting. For details about interrupt nesting, refer to 7.8.9 Setting Example of Nesting Interrupt.

Using the CMASK bit of the ILEV register enables masking an interrupt request of lower priority level than the masking level to a programmable.

All interrupt requests can be used for triggering DMA transfer.

Detailed operation of hardware interrupts is provided below. Also, refer to the chapter Exception Handling Maskable Interrupts (Interrupts) of the separate volume, *TX19 Core Architecture*.

Not Recommended for New Design

7.8.2 Hardware Interrupt Sources

Table 7.8.1 Hardware Interrupt Sources (1/2)

Interrupt Number	IVR[8 : 0]	Interrupt Name	Interrupt Source	IMR
0	0x000	Software set	Set IMR00.EIM00 to 01	IMR00
1	0x004	INT0	INT0 pin	(IMR01)
2	0x008	Reserved	---	(IMR02)
3	0x00C	Reserved	---	(IMR03)
4	0x010	Reserved	---	IMR04
5	0x014	Reserved	---	(IMR05)
6	0x018	INT1	INT1 pin	(IMR06)
7	0x01C	INT2	INT2 pin	(IMR07)
8	0x020	INT3	INT3 pin	IMR08
9	0x024	Reserved	---	(IMR09)
10	0x028	Reserved	---	(IMR10)
11	0x02C	Reserved	---	(IMR11)
12	0x030	Reserved	---	IMR12
13	0x034	Reserved	---	(IMR13)
14	0x038	Reserved	---	(IMR14)
15	0x03C	Reserved	---	(IMR15)
16	0x040	Reserved	---	IMR16
17	0x044	Reserved	---	(IMR17)
18	0x048	Reserved	---	(IMR18)
19	0x04C	Reserved	---	(IMR19)
20	0x050	INTPMD0	PMD0 count register (MDCNT0) match	IMR20
21	0x054	INTPMD1	PMD1 count register (MDCNT1) match	(IMR21)
22	0x058	INTEMG0	PMD0 EMG input (PA6)	(IMR22)
23	0x05C	INTEMG1	PMD1 EMG input (PB6)	(IMR23)
24	0x060	INTENC	Encoder match	IMR24
25	0x064	INTTBCOM00	TB0REG0 match/TB0CNT overflow	(IMR25)
26	0x068	INTTBCOM01	TB0REG1 match	(IMR26)
27	0x06C	INTTBCOM10	TB1REG0 match/TB1CNT overflow	(IMR27)
28	0x070	INTTBCOM11	TB1REG1 match	IMR28
29	0x074	INTTBCOM20	TB2REG0 match/TB2CNT overflow	(IMR29)
30	0x078	INTTBCOM21	TB2REG1 match	(IMR30)
31	0x07C	INTTBCOM30	TB3REG0 match/TB3CNT overflow	(IMR31)
32	0x080	INTTBCOM31	TB3REG1 match	IMR32
33	0x084	INTTBE0	TMRB0 EMG input (P93)	(IMR33)
34	0x088	Reserved	---	(IMR34)
35	0x08C	Reserved	---	(IMR35)
36	0x090	Reserved	---	IMR36
37	0x094	Reserved	---	(IMR37)
38	0x098	Reserved	---	(IMR38)
39	0x09C	Reserved	---	(IMR39)
40	0x0A0	Reserved	---	IMR40
41	0x0A4	Reserved	---	(IMR41)
42	0x0A8	Reserved	---	(IMR42)
43	0x0AC	Reserved	---	(IMR43)
44	0x0B0	Reserved	---	IMR44
45	0x0B4	Reserved	---	(IMR45)
46	0x0B8	Reserved	---	(IMR46)
47	0x0BC	Reserved	---	(IMR47)
48	0x0C0	INTTX0	UART0 transmit complete	IMR48
49	0x0C4	INTRX0	UART0 receive complete	(IMR49)
50	0x0C8	INTTX1	UART1 transmit complete	(IMR50)
51	0x0CC	INTRX1	UART1 receive complete	(IMR51)
52	0x0D0	INTTX2	SIO2/UART2 transmit complete	IMR52
53	0x0D4	INTRX2	SIO2/UART2 receive complete	(IMR53)
54	0x0D8	INTTX3	SIO3/UART3 transmit complete	(IMR54)
55	0x0DC	INTRX3	SIO3/UART3 receive complete	(IMR55)

Not Recommended for New Design

Table 7.8.2 Hardware Interrupt Sources (2/2)

Interrupt Number	IVR[8 : 0]	Interrupt Name	Interrupt Source	IMR
56	0x0E0	INTDMA0	DMA0 transfer complete	IMR56
57	0x0E4	INTDMA1	DMA1tranfer complete	(IMR57)
58	0x0E8	INTDMA2	DMA2 transfer complete	(IMR58)
59	0x0EC	INTDMA3	DMA3 transfer complete	(IMR59)
60	0x0F0	INTDMA4	DMA4 transfer complete	IMR60
61	0x0F4	INTDMA5	DMA5 transfer complete	(IMR61)
62	0x0F8	INTDMA6	DMA6 transfer complete	(IMR62)
63	0x0FC	INTDMA7	DMA7 transfer complete	(IMR63)
64	0x100	Reserved	---	IMR64
65	0x104	Reserved	---	(IMR65)
66	0x108	Reserved	---	(IMR66)
67	0x10C	Reserved	---	(IMR67)
68	0x110	INTAD0	ADC0 conversion complete	IMR68
69	0x114	INTADHP0	ADC0 highest-priority conversion complete	(IMR69)
70	0x118	INTADM0	ADC0 conversion value compare	(IMR70)
71	0x11C	INTAD1	ADC1 conversion complete	(IMR71)
72	0x120	INTADHP1	ADC1 highest-priority conversion complete	IMR72
73	0x124	INTADM1	ADC1 converson value compare	(IMR73)
74	0x128	INT4	INT4 pin	(IMR74)
75	0x12C	INT5	INT5 pin	(IMR75)
76	0x130	INT6	INT6 pin	IMR76
77	0x134	INT7	INT7 pin	(IMR77)
78	0x138	INT8	INT8 pin	(IMR78)
79	0x13C	INT9	INT9 pin	(IMR79)
80	0x140	Reserved	---	IMR80
81	0x144	Reserved	---	(IMR81)
82	0x148	Reserved	---	(IMR82)
83	0x14C	Reserved	---	(IMR83)
84	0x150	INTTBCAP00	TB0CAP1 capture	IMR84
85	0x154	INTTBCAP01	TB0CAP0 capture	(IMR85)
86	0x158	INTTBCAP10	TB1CAP1 capture	(IMR86)
87	0x15C	INTTBCAP11	TB1CAP0 capture	(IMR87)
88	0x160	INTTBCAP20	TB2CAP1 capture	IMR88
89	0x164	INTTBCAP21	TB2CAP0 capture	(IMR89)
90	0x168	INTTBCAP30	TB3CAP1 capture	(IMR90)
91	0x16C	INTTBCAP31	TB3CAP0 capture	(IMR91)
92	0x170	Reserved	---	IMR92
93	0x174	Reserved	---	(IMR93)
94	0x178	Reserved	---	(IMR94)
95	0x17C	Reserved	---	(IMR95)

Note1: Although IMRxx is a 32-bit register, it is accessible by 8-bit or 16-bit one. i.e. making IMR00 be IMR00/IMR01/IMR02/IMR03 enables 8-bit access.

Note2: Reserved is a reserved area for expansion. It is recommended to set the same value as initial, "0x00" to IMR register of a reserved area.

Not Recommended for New Design

7.8.3 Detection of Interrupt Requests

An interrupt request detection varies by a source as shown in Table 7.8.3. All interrupt requests, after being detected, are sent to the INTC for priority arbitration and then sent to the TX19A core processor, as illustrated in Figure 7.8.1. For a detection level that can be used by each interrupt source, refer to Table 7.8.5.

Table 7.8.3 Detecting Part of Interrupt Request

Interrupt Type	Detecting Part	Interrupt Notification Route
(1) External pin interrupt INT0 to INT3	CG	PortT → CG (detection) → INTC (arbitration) → TX19A core
(2) External pin interrupt INT4 to INT9	INTC	Port → INTC (detection/arbitration) → TX19A core
(3) Emergency stop interrupt INTEMGx	Port	Pprt (detection) → PMD → INTC (arbitration) → TX19A core
(4) Emergency stop interrupt INTTBE0	Port	Port (detection) →INTC (arbitration) → TX19A core
(5) Other interrupts	INTC	Peripheral hardware →INTC (detection/arbitration) → TX19A core

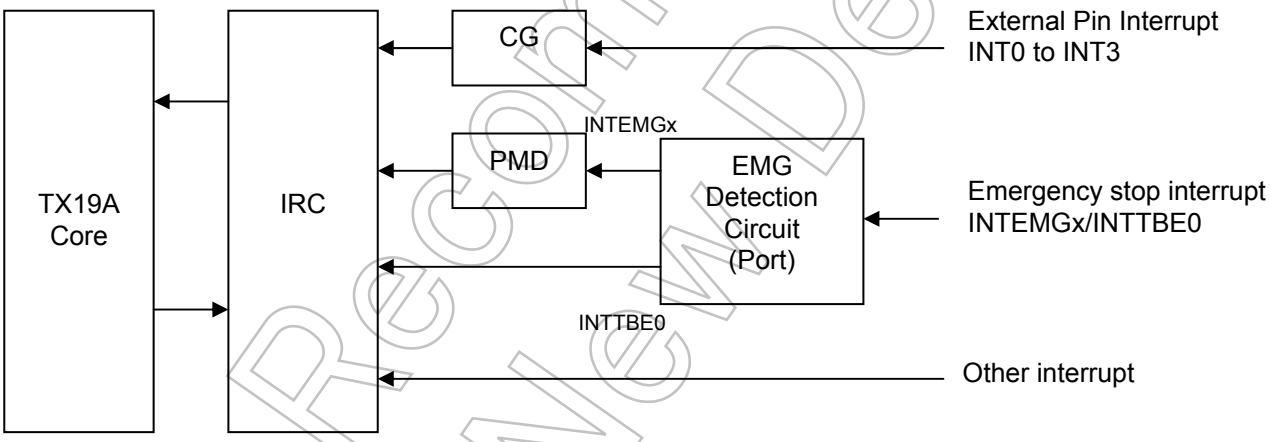


Figure 7.8.1 Notification Route of Interrupt

Not Recommended for New Design

7.8.4 Interrupt Arbitration

1. Seven levels of interrupt priority

   The INTC can set seven levels of interrupt priority individually for each interrupt source. The ILxx field of the IMRxx register is used to set priority of each interrupt source. The larger the number of interrupt level is set, the higher the priority becomes. When the value is "000" (interrupt level = 0), the source does not eneble the interrupt. And, the source of an interrupt level 0 is not stored.

2. Interrupt level notification

   When an interrupt request occurs, the INTC compares the priority level of the request interrupt with the mask level set in the CMASK field of the ILEV register. When an interrupt request has a higher priority level than that of the mask level, the INTC sends the interrupt request to the TX19A core processor.

   If two or more interrupt requests occur simultaneously, the INTC sends the interrupt request in accordance with the established priorities. If two or more interrupt requests having the same priority level occur simultaneously, the INTC sneds the interrupt request in ascending order starting from the smallest number (see Table 7.8.1).

   If another interrupt request is made from the same interrupt source before the previous interrupt request is cleared, the INTC ignores the second interrupt request.

3. INTC Register Update

   When TX19A core accepts an interrupt request, its priority level is stored in the CMASK field of the ILEV register and the corresponding vector value is set to the IVR register. CMASK/IVR once set is not updated until IVR is read or sent to the core even though an interrupt request of higher level occurs.

Note: Before changing the ILEV value, be sure to read the IVR value. If the ILEV value is changed without reading the IVR value, an unexpected interrupt may occur.

7.8.5 Hardware Interrupt Operation

When a hardware interrupt is generated, TX19A core performs the following operations and a control jumps to the exception vector address according to the BEV bit of the CP0 Status register and the IV bit of the CP0 Cause register (see Table 7.2.1).

1. The EXL bit of the CP0 Status register is set to 1.
2. The PC value upon an interrupt generation is stored in the CP0 EPC register.
3. When Shadow Register Set is enabled (CP0 register SSCR<SSD> =0), CP0register SSCR<CSS/PSS> is updated, thus a register bank of the same number as an interrupt level becomes effective.
4. The CMASK and PMASKx fields of the ILEV register of the INTC are updated to set the interrupt mask level to the priority level of the accepted interrupt.
5. Bits 0 to 8 of the IVR register of the INTC are set to the value corresponding to the accepted interrupt as shown in Table 7.8.1.

Not Recommended for New Design

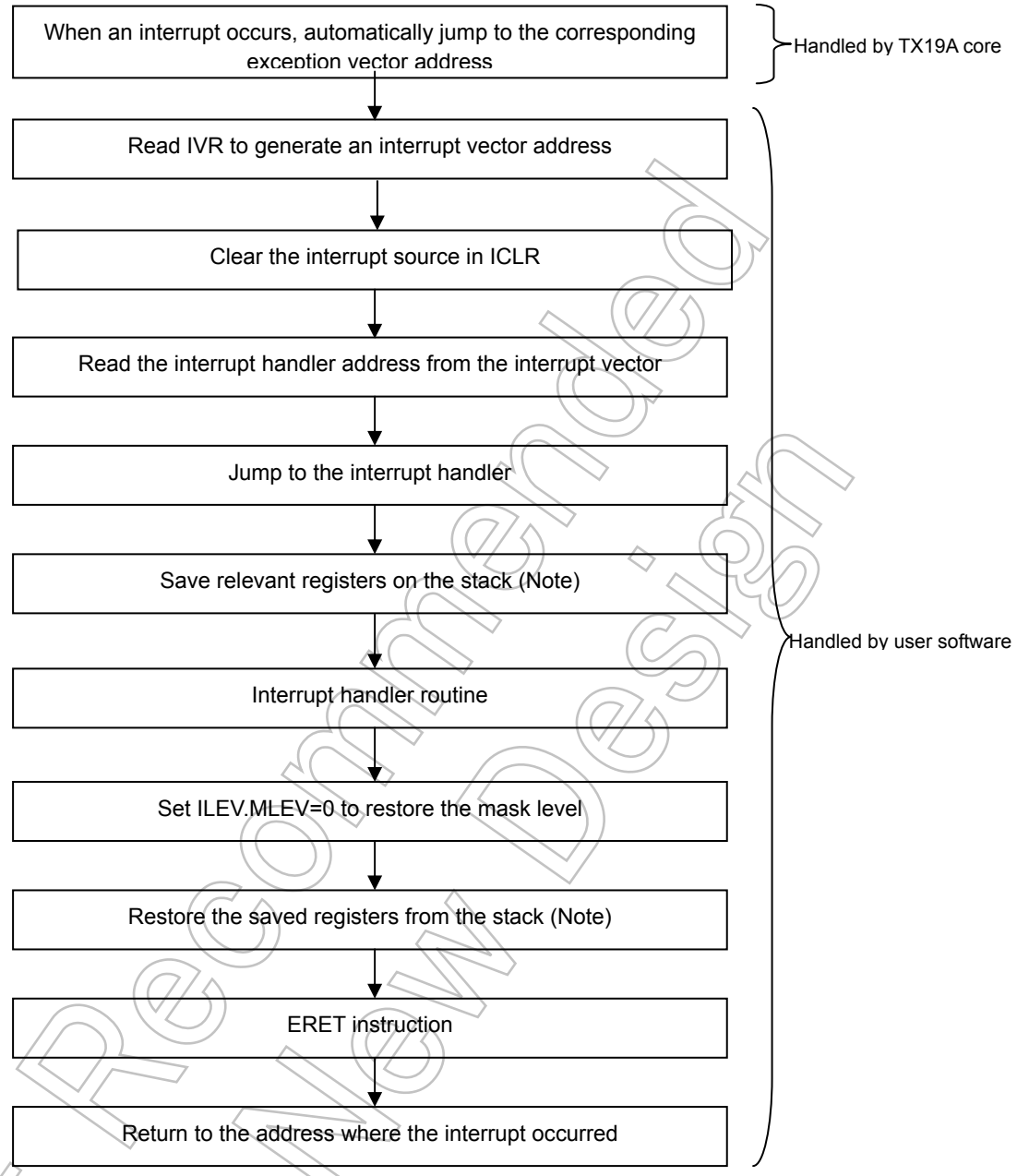


Figure 7.8.4 Basic Operation Sample of Hardware Interrupt

Note: TX19A core can automatically save the most part of a general-purposed register by using Shadow Register Set (CP0 register SSCR<SSD>=0).

Not Recommended for New Design

7.8.6 Interrupt Initial Settings

In Section 7.8.6.1, the initial settings common to all interrupts regardless of sources and in Section 7.8.6.2, the initial settings specific to each interrupt source are described, both as necessary settings before using interrupts.

7.8.6.1 Initial Settings Common to All Interrupts

The following settings must be made in order to use interrupts.

1. Set the IM[4:2] field of the CP0 Status register to 111.
2. Set the base address of the interrupt vector table in bits 9 to 31 of the INTC IVR register.
3. Set an interrupt handler address for a respective interrupt source to the address, the sum of a base address of interrupt vector table and IVR[8:0] by interrupt source.

Programming example for the above 1.: Using exeption vector address 0xBFC00400

```
lui     r2,0x1040          ; CU0=1 ,BEV =1 (r2 =0x1040_xxxx)
addiu   r2,r2,0x1C00       ; IM4,IM3,IM2 =1 (r2 =0x1040_1C00)
mtc0    r2,r12
```

Programming example for the above 2.: Using VectorTable as a label of the interrupt vector table

```
lui     r3,hi(VectorTable)
addiu   r3,r3,lo(VectorTable)   ; r3 = VectorTable address
lui     r2,hi(IVR)              ; r2 =0xFFFF_xxxx (upper 16 bits of address in IVR)
sw      r3,lo(IVR)(r2)          ; Set VectorTable address in IVR[31:9]
```

Programing example for the above 3.: Using address 0xBFC20000 as a base address of the interrupt vector table

```
_VectorTable section code isa32 abs=0xBFC20000
VectorTable:
        dw      _SWINT          ; 0 --- software interrupt
        dw      _INT0           ; 1 --- INT0
        dw      _RESEARVED      ; 2 --- Reserved
        dw      _RESEARVED      ; 3 --- Reserved
        dw      _RESEARVED      ; 4 --- Reserved
        dw      _RESEARVED      ; 5 --- Reserved
        dw      _INT1           ; 6 --- INT1
        dw      _INT2           ; 7 --- INT2
        dw      _INT3           ; 8 --- INT3
        dw      _RESEARVED      ; 9 --- Reserved
```

Note: These examples assume the use of a Toshiba assembler. When using a third-party assembler, modify them as necessary to avoid syntax errors.

Not Recommended for New Design

7.8.6.2 Initial Settings Specific to Each Interrupt Source

The registers that must be set for using an interrupt varies by sources shown below:

Table 7.8.5 Interrupt Detection and Setting Register

Interrupt Type	Setting Regiser	Supported Interrupt Sensitivity Settings
(1) External pin interrupts INT0 to INT3	PxIER (Port) PxFR (Port) CLKINTx (CG) IMRxx (INTC)	Programmable as low level, high level, falling edge, or rising edge sensitive through the IxSEN field of the CLKINTx register in the CG. In the INTC, the EIMxx field of the IMRxx register must be set to falling edge or low level according to the setting made in the CG.
(2) External pin interrupts INT4 to INT9	PxIER (Port) PxFR (Port) IMRxx (INTC)	Programmable as low level, high level, falling edge, or rising edge sensitive through the EIMxx field of the IMRxx register in the INTC.
(3) Emergency stop interrupts INTEMGx	PxIER (Port) PxFR (Port) PxECR (Port) EMGCRx (PMD) IMRxx (INTC)	Programmable as low level, high level, falling edge, or rising edge sensitive through the ERMx field of the PxECR register in the port unit. In the INTC, the EIMxx field of the IMRxx register must be set to falling edge.
(4) Emergency stop interrupt INTTBE0	P9IER (Port) P9FR2 (Port) P9ECR (Port) IMR33 (INTC)	Programmable as low level, high level, falling edge, or rising edge sensitive through the ERM9 field of the P9ECR register in the port unit. In the INTC, the EIM33 field of the IMR33 register must be set to falling edge or low level.
(5) Other interrupts	IMRxx (INTC)	Must always be set as falling edge sensitive.

Note: In level detection, a value is checked at internal clock timing each time. An edge is detected by comparing a previous value with a current value at internal clock timing.

1. External Pin Interrupts, INT0 to INT3
   - In the port unit, set the PxIER register to enable input (see 7. Port Function).
   - In the port unit, set INT0 to INT3 as the pin function to the PxFR register (see 7. Port Function).
   - In the CG, set Interrupt Sensitivity in the IxSEN field of the CLKINTx register (see 5.3.3 Interrupt Registers).
   - In the CG, set Enable/Disable of Standby Cancel in the IxKI bit of the CLKINTx register (see 5.3.3 Interrupt Registers).
   - In the INTC, set the EIMxx field of the IMRxx register to specify the sensitivity of the interrupt signal sent from the CG. When rising/falling edge is selected in the CLKINTx.IxSEN, set 10 to the IMRxx.EIMxx to select falling edge. When high/low level is selected in the CLKINTx.IxSEN, set 00 to the IMRxx.EIMxx to select low level (see 7.8.10 Register).

Note 1: To write to the CLKINTx register, it is necessary to write 0x5A5A and then 0xF0F0 in the CGACT register.

Note 2: To initialize an interrupt, follow the interrupt detection route indicated in Table 7.8.3 and make the interrupt enable with the CP0 register. If any different setting order is used, an unexpected interrupt may be generated. So, be sure to clear interrupt sources before setting interrupt enable. Similarly, to disable an interrupt, make the interrupt disable with the CP0 register and then set the registers accordingly in the reverse order of interrupt detection route.

Not Recommended for New Design

・Setting example: Using the external pin interrupt INT3 for waking up from STOP mode (rising edge)

```
Status<IE> =”0”                          ; Disable interrupts
P6IER<P6IER4> =”0”                       ; Enable port input
P6FR<P6FR4> =”1”                         ; Configure port as INT3
CLKINT3<I3SEN> =”010”                    ; Set INT3 as falling edge sensitive
CLKINT3<I3KI> =”1”                       ; Set INT3 as STOP wakeup signal
CLKACT =”0x5A5A”→“0xF0F0”                ; Enable CG register settings
IMR08<EIM08> =”10”                       ; Set INT3 as falling edge sensitive
ICLR<IV> =”0x020”                        ; Clear INT3 interrupt request
IMR08<IL08> =”101”                       ; Set  INT3 interrupt level to 5
ILEV<MLEV>/<CMASK> =”1”/”xxx”            ; Set mask level to ”xxx” (set simultaneously with ILEV<MLEV>)
SYNC instruction                         ; Stall  until interrupt settings take effect
Status<IE> =”1”                          ; Enable interrupts
```

・Setting example: Using the external pin interrupt INT3 for making it disable

```
Status<IE> =”0”                          ; Disable interrupts
IMR08<IL08> =”000”                       ; Disable INT3 interrupt
ICLR<IV> =”0x020”                        ; Clear INT3 interrupt request
```

2. External Pin Interrupts, INT4 to INT9
   - In the port unit, set the PxIER register to enable input (see 7. Port Function).
   - In the port unit, set INT4 to INT9 as the pin function to the PxFR register (see 7. Port Function).
   - In the INTC, set the EIMxx field of the IMRxx register to specify the sensitivity of the interrupt signal (see 7.8.10 Register).

Note 1: To initialize an interrupt, follow the interrupt detection route indicated in Table 7.8.3 and make the interrupt enable with the CP0 register. If any different setting order is used, an unexpected interrupt may be generated. So, be sure to clear interrupt sources before setting interrupt enable. Similarly, to disable an interrupt, make the interrupt disable with the CP0 register and then set the registers accordingly in the reverse order of interrupt detection route.

・Setting example: Using the external pin interrupt INT4 as H level

```
Status<IE> =”0”                          ; Disable interrupts
P6IER<P6IER5> =”0”                       ; Enable port input
P6FR<P6FR5> =”1”                         ; Configure port as INT4
IMR74<EIM74> =”01”                       ; Set INT4 as high level sinsitive
ICLR<IV> =”0x020”                        ; Clear INT4 interrupt request
IMR74<IL74> =”010”                       ; Set INT4 interrupt level to 2
ILEV<MLEV>/<CMASK> =”1”/”xxx”            ; Set mask level to ”xxx" (set simultaneously with ILEV<MLEV>)
SYNC instruction                         ; Stall until interrupt settings take effect
Status<IE> =”1”                          ; Enable interrupts
```

Not Recommended for New Design

3. Interrupt Halted, INTEMG0/INTEMG1

For detailed setting example, refer to the section 7.12 Usage Note of EMG Input Pin (PA6/PB6).

- In the port unit, set the ERMx field of PxECR register to be sensitive (see 7. Port Function).
- In the port unit, set Input Enable to the PxIER register (see 7. Port Function).
- In the port unit, set EMGx to the pin function of PxFR register (see 7. Port Function).
- In the PMD, set 1 to the EMGEN field of the EMGCRx register (see 12.3.4 EMG Protection Circuit).
- Set 10 to IMRxx<EIMxx> of INTC (see 7.8.10 Register).

Note 1: To set PxECR of a port, set 0x55 to PxECLR of the port first and then 0xAA.

Note 2: To initialize an interrupt, enable the interrupt in CP0 register after setting it by following the interrupt detection routine as shown in Table 7.8.3. If the setting order varies, an unexpected interrupt may be generated or unexpected transfer of EMG state may be made. When setting an interrupt to Enable, the interrupt sources and EMG state must be cleared to 0. Also be sure to set an interrupt in reverse order of the detection routine after disabling an interrupt in CP0 register when disabling an interrupt.

4. Interrupt Halted, INTTBE0

For detailed setting example, refer to the section 7.9.1 Usage Note of EMG Input Pin (P.93).

- In the port unit, set the ERM9 field of the P9ECR register to be sensitive (see 7. Port Function).
- In the port unit, set Input Enable to the port of P9IER register (see 7. Port Function).
- In the port unit, set EMG Input to the pin function of P9FR register (see 7. Port Function).
- Set 10 to IMR33<EIM33> of INTC (see 7.8.10 Register).

Note 1: To set PxECR of a port, set 0x55 to PxECLR of the port first and then 0xAA.

Note 2: To initialize an interrupt, enable the interrupt in CP0 register after setting it by following the interrupt detection routine as shown in Table 7.8.3. If the setting order varies, an unexpected interrupt may be generated or unexpected transfer of EMG state may be made. When setting an interrupt to Enable, the interrupt sources and EMG state must be cleared to 0. Also be sure to set an interrupt in reverse order of the detection routine after disabling an interrupt in CP0 register when disabling an interrupt.

5. Other Hardware Interrupt

- Set the peripheral hardware to use.
- Set 10 to IMRxx<EIMxx> of INTC (see 7.8.10 Register).

Note 1: To initialize an interrupt, enable the interrupt in CP0 register after setting INTC. To disable an interrupt, set INTC after disabling it in the CP0 register.

Not Recommended for New Design

7.8.7 Enabling/Disabling Interrupts

Here, it is described the procedure of enabling and disabling of interrupt being programmed.

7.8.7.1 Enabling Interrupts

To enable interrupts, all the following three conditions must be satisfied in addition to the settings described in 7.8.6 Interrupt Initial Settings:

- The ERL bit of the CP0 Status register is cleared to 0.
- The EXL bit of the CP0 Status register is cleared to 0.
- The IE bit of the CP0 Status register is set to 1.

When an instruction which makes these settings is executed, interrupts are enabled and the register setting takes effect after two clock cycles. The IE bit of the CP0 Status register can be set to 1 in the following four ways:

- Set the IE bit of the CP0 Status register to 1 using the MTC0 instruction of 32-bit ISA.
- Set the CP0 IER register to a value other than 0 using the MTC0 instruction of 32-bit ISA (see Note 1.)
- Set the IE bit of the CP0 Status register to 1 using the MTC0 instruction of 16-bit ISA.
- Execute the EI instruction of 16-bit ISA (see Note 2.)

Note 1: It is recommended to use this measure when enabling an interrupt for 32-bit ISA because of the code efficiency. In Toshiba's C compiler, too, this instruction is executed for __EI() intrinsic function of 32-bit ISA.

Note 2: It is recommended to use this measure when enabling an interrupt for 16-bit ISA because of the code efficiency. In Toshiba's C compiler, too, this instruction is executed for __EI() intrinsic function of 16-bit ISA.

Note 3: Of the above four methods, we recommend using the second or fourth because of smaller code size and faster execution.

Not Recommended for New Design

7.8.7.2 Disabling Interrupts

Interrupts are disabled if any of the following three conditions is satisfied. When interrupts are disabled in this way, interrupt requests from interrupt sources that have been enabled in the initial setting (see 7.8.6 Interrupt Initial Settings) remain pending. Note that the TMP19A71 does not latch interrupt requests from interrupt sources whose level is set to 0.

- The ERL bit of the CP0 Status register is set to 1.
- The EXL bit of the CP0 Status register is set to 1.
- The IE bit of the CP0 Status register is cleared to 0.

Execution of an instruction which makes these settings immediately disables interrupts and the register setting takes effect after two clock cycles. The ERL and EXL bits of the CP0 Status registrer are automatically set when an interrupt or exception occurs, and are automatically cleared when the ERET instruction is executed. Therefore, for disabling interrupts, we recommend using the third method, i.e., clearing the IE bit of the CP0 Status register to 0. For how to disable interrupts when interrupt nesting is used, see 7.8.9 Setting Example of Nesting Interrupt. The IE bit of the CP0 Status register can be cleared to 0 in the following four ways:

- Clear the IE bit of the CP0 Status register to 0 using the MTC0 instruction of 32-bit ISA.
- Clear the CP0 IER register to 0 using the MTC0 istruction of 32-bit ISA (see Note 1).
- Clear the IE bit of the CP0 Status register to 0 using the MTC0 instruction of 16-bit ISA.
- Execute the DI instruction of 16-bit ISA (see Note 2).

Note 1: It is recommended to use this measure when disabling an interrupt for 32-bit ISA because of the code efficiency. In Toshiba's C compiler, too, this instruction is executed for __DI() intrinsic function of 32-bit ISA.

Note 2: It is recommended to use this measure when disabling an interrupt for 16-bit ISA because of the code efficiency. In Toshiba's C compiler, too, this instruction is executed for __DI() intrinsic function of 16-bit ISA.

Note 3: Of the above four methods, we recommend using the second or fourth because of smaller code size and faster execution.

To disable individual source of interrupt that has been enabled once after its level is set with IMRxx<ILxx> of INTCb (IMRxx<ILxx> ="000"), set Staus<ERL/EXL/EI> of CP0 register by following the example shown below, and then disable an interrupt source after disabling the interrupt.

Programming example for disabling interrupt sources individually

```
mtc0    r0, IER          ; Disable interrupts (Clear Status<IE> to 0)
sb      r0, IMRxx        ; Disable interrupt sources
sync                     ; Stall until writing becomes effective
mtc0    r29, IER         ; Enable interrupts (Set Status<IE> to 1)
```

Note1: This programming example is of the time when using Toshiba's assembler. When the third-party assembler is used, programming error may occur. The program should be changed according to an assembler to use.

Not Recommended for New Design

7.8.8 Interrupt Handling

Here, the detailed operation is described based on the basic flow of Figure 7.8.4.

7.8.8.1 Interrupt Response and Restore

1. Interrupt Accepted by Hardware

After an interrupt request arbitration, INTC sets the interrupt vector and interrupt level of the interrupt request accepted to IVR and ILEV<CMASK>, respectively, to notify the TX19A processor core of the interrupt level. When the interrupt level is notified, the TX19A processor core sets 1 to Status <EXL> of the CP0 register to disable interrupts and saves the PC value at the interrupt generation to EPC. If Shadow Register Set is enabled (CP0 register SSCR <SSD> = 0), the processor core sets the interrupt level to SSCR <CSS> of the CP0 register and switches the register bank.

When an interrupt is accepted, any ongoing execution is suspended and it automatically jumps to the exception vector address (for interrupts). Figure 7.8.2 shows the sequence of accepting interrupts.

Not Recommended for New Design

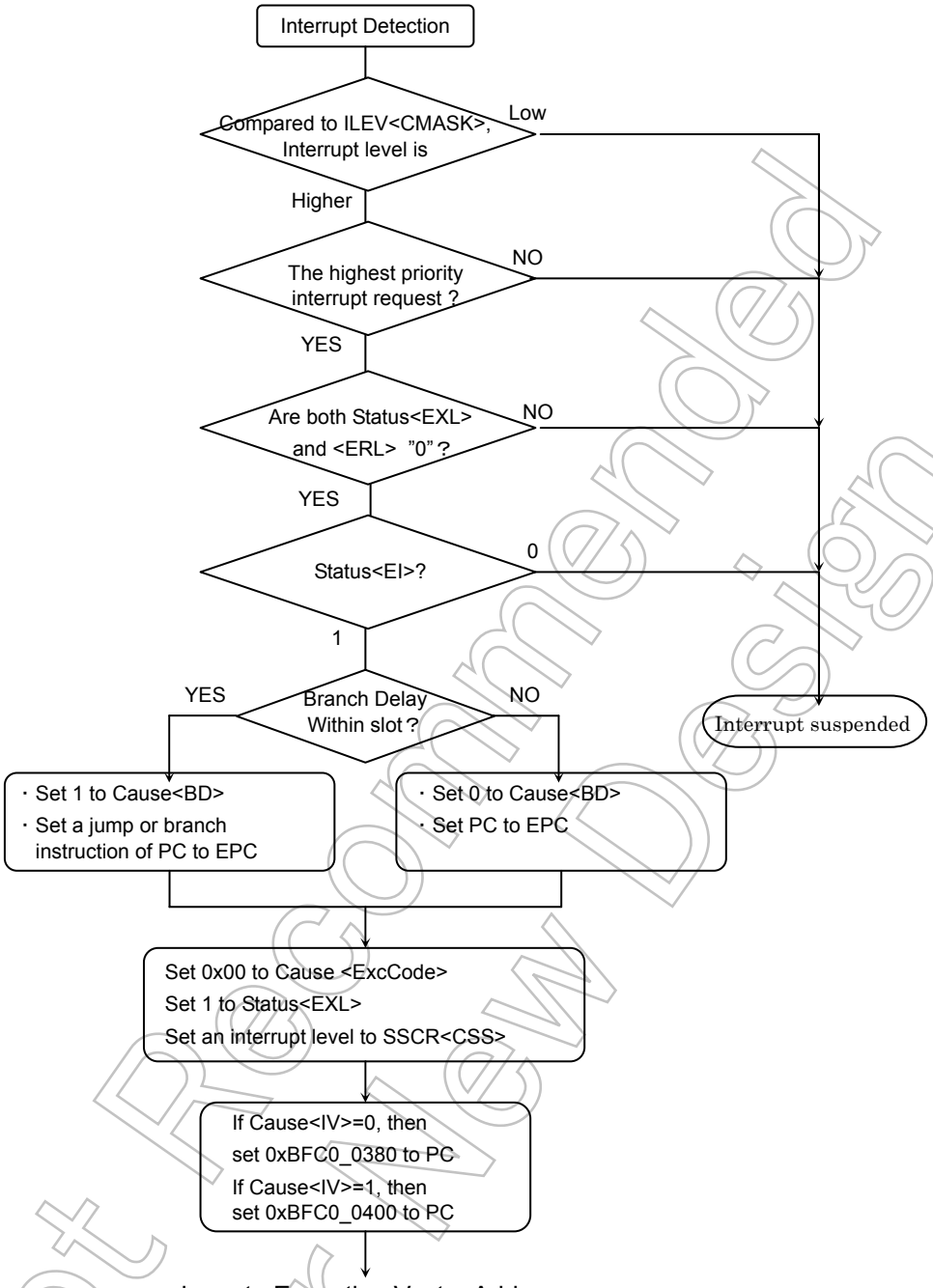


Figure 7.8.2 Sequence of Interrupt Accepted by Hardware

Not Recommended for New Design

2. Process Necessary for Exception Handler

After an interrupt request is accepted, it automatically jumps to the exception handler in which the interrupt vector address is read from INTC IVR, and the user program generates the address of the interrupt handler. As in the example statements presented in Section 7.8.6 Interrupt Initial Setting, an interrupt vector base address is set in the range of IVR[31:8], thus the IVR value becomes the interrupt vector address.

After reading the INTC IVR value, an interrupt source is cleared. If the interrupt source is cleared before IVR is read, no correct value can be read because the IVR value is also cleared.

Programming example of exception handlers: when exception vector address (interrupt) is 0xBFC0_0400

```
VECTOR_INT section code isa32 abs=0xBFC00400
__InterruptVector:
lui     r26,hi(IVR)
lw      r26,lo(IVR)(r26)        ; Read interrupt vector address from IVR
lui     r27,hi(ICLR)
sh      r26,lo(ICLR)(r27)       ; Clear interrupt request
lw      r26,0(r26)              ; Read interrupt handler address from interrupt vector
jr      r26                     ; Jump to interrupt handler
nop
```

Note 1: This programming example is of the case Toshiba's assembler is used. When the third-party assembler is used, syntax error may occur. Program should be changed according to an assembler to use.

3. Process Necessary for Interrupt Handler

Typical tasks of the interrupt handler are to save appropriate registers and to process interrupts. If the shadow register set is enabled (CP0 register SSCR <SSD> = 0), the general-purposed register values other than r26, r27, r28, and r29 (Shadow Register Set number 1 to 7) are automatically saved, thus user program doesn't need to save them. Refer to the separate volume, *TX19A Core Architecture* for details of general-purposed registers that are to be saved.

Generally, registers other than general-purposed registers are dependent on user programs. The Status, EPC, SSCR, HI, LO, Cause, and Config values of the CP0 register shall be saved as appropriate. Clearing Status<EXL>to 0 after the saving process, nesting interrupts can be used by enabling interrupts.

Note 1: Since general exceptions are accepted even when interrupts are disabled, it is recommended to save general-purposed registers and CP0 register that may be rewritten by general exceptions even when nesting interrupts is not to be used.

Not Recommended for New Design

Setting example necessary for interrupt handler

```
SSCR→ Save on the stack          ; Saving SSCR values (as appropriate)
EPC →Save on the stack           ; Saving EPC values (as appropriate)
Status →Save on the stack        ; Saving Status values (as appropriate)
NOP instruction                  ; Stall before the execution of ERET instruction
NOP instruction                  ; Stall before the execution of ERET instruction
Status<EXL> ="0"                 ; Interrupt enabled (only when nesting interrupts)
```

Note 1: After rewriting SSCR of CP0 register, wait for two instructions to allow for register bank switching and then access to the register.

4. Restore From Interrupt Handler

To restore from an interrupt handler to the main process, restore the register saved at the head of the interrupt handler and set 0 to INTC ILEV<MLEV> to clear the interrupt mask level. By executing the ERET instruction after all the restorings are completed, Status<EXL> of the CP0 register is cleared to 0 and the EPC address is restored in PC for resuming the main process. When Shadow Register Set is sensitive (CP0 register SSCR <SSD> = 0), SSCR<CSS> is updated by the ERET instruction, and the previous number of Shadow Register Set is restored automatically, thus the general-purposed registers saved in the register bank is also automatically restored.

If nesting interrupts are used, it is necessary to set 1 to Status<EXL> of the CP0 register before restoring to disable interrupts.

Setting example of restoring from interrupt handler

```
Status<EXL> ="1"                 ; Interrupt disabled (only when nesting interrupts)
ILEV<MLEV> ="0"                  ; Restore the mask level by one
SYNC instruction                 ; Stall until the mask level is restored
SSCR ←saved SSCR                 ; Restore SSCR values (as appropriate)
NOP instruction                  ; Stall until SSCR is switched
NOP instruction                  ; Stall until SSCR is switched
EPC ←saved EPC                   ; Restore EPC values (as appropriate)
Status ←saved Status             ; Restore Status values (as appropriate)
NOP instruction                  ; Stall  before executing ERET instruction
NOP instruction                  ; Stall  before executing ERET instruction
ERET instruction                 ; Status<EXL> ="0", PC ←EPC, SSCR<CSS> ←SSCR<PSS>
NOP instruction                  ; Stall after ERET instruction (only for TMP19A70)
```

Note 1: After rewriting SSCR of CP0 register, wait for two instructions to allow for register bank switching and then access to the register.

Note 2: Do not access CP0 register two instructions prior to the execution of ERET instruction.

Note 3: After ERET instruction execution, NOP instruction must be set (only for TMP19A70).

Not Recommended for New Design

7.8.9 Setting Example of Nesting Interrupt

Nesting interrupt is the processing of the interrupt request of higher priority during the processing of some other interrupts. TMP19A71 can perform nesting interrupt because INTC arbitrates the priority of interrupts. When an interrupt request is accepted, ILEV<CMASK> of INTC is automatically updated to the interrupt level of the interrupt accepted, so that it can be arbitrated according to the priority preset by the user program.

1. Additional processes required for nesting interrupts

When an interrupt is accepted, 1 is set to the Status<EXL> of the CP0 register, and interrupt becomes disabled. In order to allow nesting interrupts, it is necessary to save the registers that could be overwritten by the second and the following interrupts before enabling the nesting interrupt process. For this purpose, in addition to the typical exception handler and interrupt handler processes, save the following registers before setting 0 to Status<EXL> of the CP0 register and then enable interrupts.

CP0 registers that must be saved:

- EPC
- SSCR

Note1: Some of the registers are automatically saved and restored by using interrupt functions of Toshiba's C compilier. For details, refer to the additional document of TX19 Toshiba C compiler, *TX19A C Compiler Reference*.

2. Additional restoration required for nesting interrupts

Before restoring registers in the restoration from interrupts, it is necessary to disable interrupts in the way described in 7.8.7.2 Interrupt Disabled. This is to prevent a restored register value from being corrupted by nesting interrupts. The ERET instruction automatically clears Status <EXL> of the CP0 register to 0. Therefore, by setting 1 to Status<EXL> of the CP0 register to disable interrupts in the restoration, it is possible to restore automatically from the interrupt which is in interrupts enabled state.

3. Proper use of Status <EXL> and Status <IE>

While there is no significant distinction between the Status<EXL> and Status<IE> parameters, Status<EXL> is automatically set to 1 upon interrupt generation and cleared to 0 by the ERET instruction automatically. In saving and restoring register values at the top and end, where interrupts have to be disabled, Status<EXL> controlled by hardware is normally used. Status<IE> is used for other general interrupt enabled/disabled control functions.

A control flow of interrupt enabled/disabled is described in Section 7.8.9.1 Interrupt Control for Nesting Interrupt.

Not Recommended for New Design

7.8.9.1 Interrupt Control for Nesting Interrupt

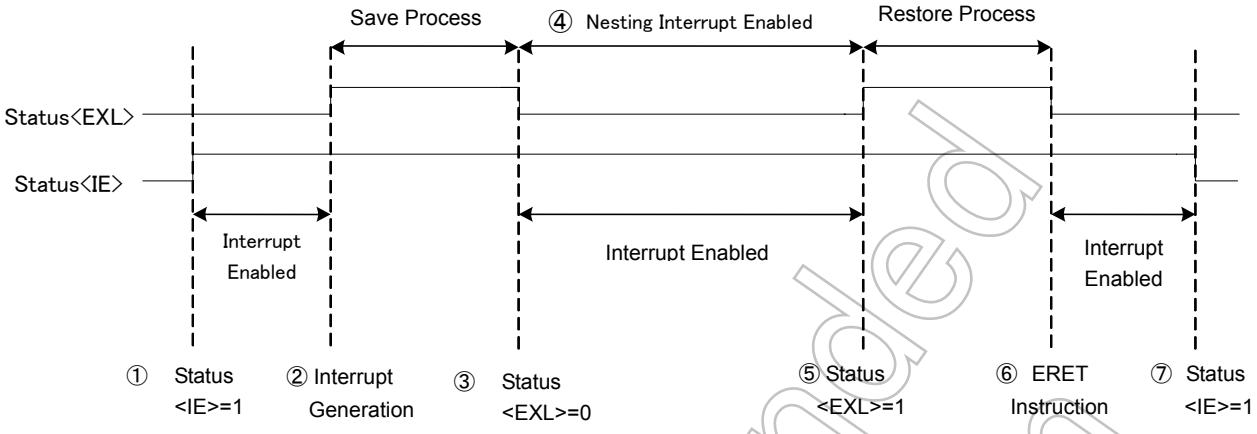


Figure 7.8.3 Interrupt Enabled/Disabled of Nesting Interrupt Control

1. Status<IE>=1

Enabling interrupts becomes possible by setting 1 to Status<IE> of CP0 register in the condition that Status<EXL> of CP0 register is 0. This process shall be optionally set by software as appropriate.

2. Interrupt Generation

As interrupts be generated, 1 is automatically set to Status<EXL> of CP0 register, and the interrupt becomes disabled. This is processed automatically by hardware.

3. Status<EXL>=0

To enable nesting interrupts, it is necessary to enable interrupts by setting 0 to Status <EXL> of the CP0 register after saving relevant registers. If interrupts are made enabled before saving registers, a higher priority level interrupt may corrupt the register data. This process shall be optionally set by software as appropriate.

4. Nesting Interrupt Enabled

It is an enabled interval of nesting interrupts. The interrupts of higher level than the current interrupt level (ILEV<CMASK>) are accepted. To disable interrupts in this interval, set 0 to Status<IE> of CP0 register.

5. Status<EXL>=1

If nesting interrupts are made enabled, it is necessary to to disable interrupts by setting 1 to Status <EXL> of the CP0 register before restoring relevant register values. If registers are saved before disabling interrupts, a higher priority level interrupt may corrupt the register data. This process shall be optionally set by software as appropriate.

6. ERET Instruction

It is the instrucition to restore the state before an interrupt generation. If this instruction is executed while Status<EXL> of the CP0 register is set to 1, 0 is automatically set to the Status<EXL>, and interrupt becomes enabled (provided that 1 is set to Status<IE> of the CP0 register).

7. Status<IE>=0

Disabling interrupts is possible by setting 0 to Status<IE> of CP0 register. This process shall be optionally set by software as appropriate.

Not Recommended for New Design

7.8.10 Register

7.8.10.1 Register Map

Table 7.8.6 INTC Register Map

Address	Mnemonic	Register Name	Corresponding Interrupt Number
0xFFFF_D000	IMR00	Interrupt Mode Control Register 00	0 - 3
0xFFFF_D004	IMR04	Interrupt Mode Control Register 04	4 - 7
0xFFFF_D008	IMR08	Interrupt Mode Control Register 08	8 - 11
0xFFFF_D00C	IMR12	Interrupt Mode Control Register 12	12 - 15
0xFFFF_D010	IMR16	Interrupt Mode Control Register 16	16 - 19
0xFFFF_D014	IMR20	Interrupt Mode Control Register 20	20 - 23
0xFFFF_D018	IMR24	Interrupt Mode Control Register 24	24 - 27
0xFFFF_D01C	IMR28	Interrupt Mode Control Register 28	28 - 31
0xFFFF_D020	IMR32	Interrupt Mode Control Register 32	32 - 35
0xFFFF_D024	IMR36	Interrupt Mode Control Register 36	36 - 39
0xFFFF_D028	IMR40	Interrupt Mode Control Register 40	40 - 43
0xFFFF_D02C	IMR44	Interrupt Mode Control Register 44	44 - 47
0xFFFF_D030	IMR48	Interrupt Mode Control Register 48	48 - 51
0xFFFF_D034	IMR52	Interrupt Mode Control Register 52	52 - 55
0xFFFF_D038	IMR56	Interrupt Mode Control Register 56	56 - 59
0xFFFF_D03C	IMR60	Interrupt Mode Control Register 60	60 - 63
0xFFFF_D040	IMR64	Interrupt Mode Control Register 64	64 - 67
0xFFFF_D044	IMR68	Interrupt Mode Control Register 68	68 - 71
0xFFFF_D048	IMR72	Interrupt Mode Control Register 72	72 - 75
0xFFFF_D04C	IMR76	Interrupt Mode Control Register 76	76 - 79
0xFFFF_D050	IMR80	Interrupt Mode Control Register 80	80 - 83
0xFFFF_D054	IMR84	Interrupt Mode Control Register 84	84 - 87
0xFFFF_D058	IMR88	Interrupt Mode Control Register 88	88 - 91
0xFFFF_D05C	IMR92	Interrupt Mode Control Register 92	92 - 95
0xFFFF_D080	IVR	Interrupt Vector Register	All (0 - 95)
0xFFFF_D084	ICLR	Interrupt Request Clear Register	All (0 - 95)
0xFFFF_D088	ILEV	Interrupt Mask Level Register	All (0 - 95)

Note 1: While an interrupt mode control register (IMRxx) is 32-bit register, it is accesible by 16-bit and 8-bit ones.

Note 2: The interrupt number to which Reserved is set in Table 7.8.1 Hardware Interrupt Sources is a reserved area for expansion. 0, the same value as initial value shall be set to interrupt mode control registers (IMRxx) of relevant interrupt number.

Not Recommended for New Design

7.8.10.2 Interrupt Vector Register (IVR)

IVR is the register indicating an interrupt vector address of interrupt source generated. When an interrupt request is accepted, the corresponding values to Table 7.8.1 is set to IVR[8:2]. IVR[31:9] are the bits readable and writable. By setting a base address of interrupt vecter, an interrupt vector address can be generated easily only by reading IVR.

Interrupt Vector Register

IVR (0xFFFF_D080)

	7	6	5	4	3	2	1	0
Bit Symbol	IVR7	IVR6	IVR5	IVR4	IVR3	IVR2	–	–
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0
Function	A vector of interrupt source being generated is set.							
	15	14	13	12	11	10	9	8
Bit Symbol	–	–	–	–	–	–	–	IVR8
Read/Write	R/W							R
Reset Value	0	0	0	0	0	0	0	0
Function								A vector of interrupt source being generated is set.
	23	22	21	20	19	18	17	16
Bit Symbol	–	–	–	–	–	–	–	–
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								
	31	30	29	28	27	26	25	24
Bit Symbol	–	–	–	–	–	–	–	–
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								

Not Recommended for New Design

7.8.10.3 Interrupt Level Register (ILEV)

ILEV is the register that controls a level notifying interrupt requests fromINTC to TX19A processor core.

Those under the interrupt level ILEV<CMASK> are suspended. The top of the priority is 7 and the lowest is 1. Note that any interrupt of the interrupt level 0 is not suspended.

When an interrupt is generated, its interrupt level is stored in <CMASK>, and any previously stored values are incremented in mask levels such that the previous CMASK is saved in PMASK0, PMASK0 in PMASK1, and so on. To write newly a value of <CMASK>, write <CMASK> as set 1 to <MLEV>. No value of <PMASKx> can be rewritten.

When 0 is set to <MLEV>, the interrupt mask level in the register shifts back to the previous state such that PMASK0 is moved to CMASK, PMASK1 to PMASK0, and so on. To <PMASK6>, 000 is set. To restore from an interrupt, set 0 to <MLEV> before executing the ERET instruction. <MLEV> always can read 0.

Interrupt Level Register

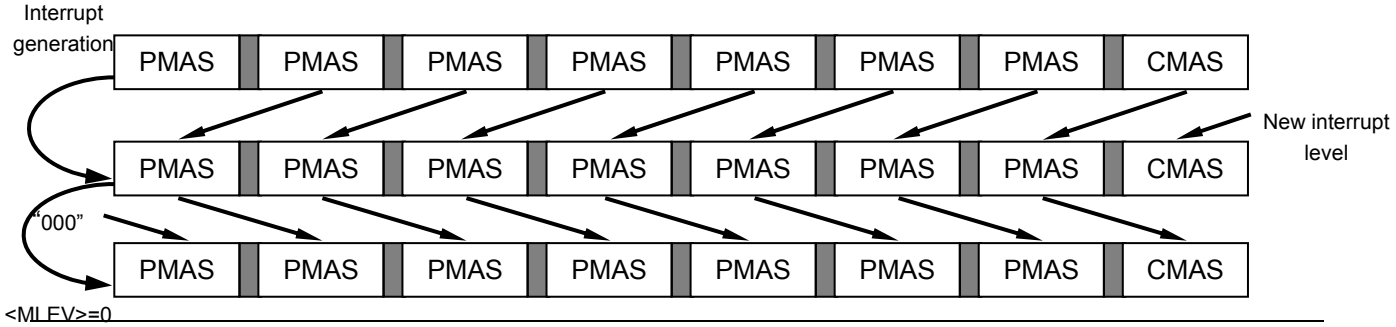
ILEV (0xFFFF_D088)

	7	6	5	4	3	2	1	0
Bit Symbol	—	PMASK0			—	CMASK		
Read/Write	R					R/W		
Reset Value	0	000			0	000		
Function		Interrupt mask level (previous) 0				Interrupt mask level (current)		
	15	14	13	12	11	10	9	8
Bit Symbol	—	PMASK2			—	PMASK1		
Read/Write	R							
Reset Value	0	000			0	000		
Function		Interrupt mask level (previous) 2				Interrupt mask level (previous)1		
	23	22	21	20	19	18	17	16
Bit Symbol	—	PMASK4			—	PMASK3		
Read/Write	R							
Reset Value	0	000			0	000		
Function		Interrupt mask level (previous) 4				Interrupt mask level (previous) 3		
	31	30	29	28	27	26	25	24
Bit Symbol	MLEV	PMASK6			—	PMASK5		
Read/Write	W	R						
Reset Value	0	000			0	000		
Function	0:Mask level restored 1:CMASK changed	Interrupt mask level (previous) 6				Interrupt mask level (previous) 5		

Note 1: This register must be accessed as a 32-bit quantity.

Note 2: Before changing the ILEV value, be sure to read the IVR value. If the ILEV value is changed without reading the IVR value, an unexpected interrupt may be generated.

Note 3: This register does not support bit manipulation instructions.



Not Recommended for New Design

7.8.10.4 Interrupt Mode Control Registers (IMRxx)

IMRxx consists of:

<ILxx>: determines the interrupt level by sources

<DMxx>: set to starting souces of DMA transfer

<EIMXX>: determines Sensitivity of interrupt request

The interrupt numbers to which Reserved is set in Table 7.8.1 Hardware Interrupt Sources are reserved area for expansion. 0, the same as the initial value shall be set to IMRxx of relevant interrupt numbers.

This register can access in the quantity of 16-/8-/1-bit by deviding IMR00 (32 bits) by 8 bits into IMR00/IMR01/IMR02/IMR03.

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR00	Bit Symbol	—	EIM00		DM00	—	IL00		
(0xFFFF_D000)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Interrupt request Setting this field to 01 generates an interrupt.		DMAC trigger 0: Disable 1:Enable interrupt number 0 as DMAC trigger		When DM00 = 0 Interrupt number 0 (software set) priority level 000: Interrupt disabled 001-111: 1-7 When DM00 = 1 DMAC channel select 000-111: 0-7		
		15	14	13	12	11	10	9	8
(IMR01)	Bit Symbol	—	EIM01		DM01	—	IL01		
(0xFFFF_D001)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Sensitivity of interrupt requests is set. When Sensitivity in CG is Edge, 10 shall be set, and when is Level, 00 shall be set.		DMAC trigger 0: Disable 1: Enable interrupt number 1 as DMAC trigger		When DM01 = 0 Interrupt number 1 (INT0) priority level 000: Interrupt disabled 001-111: 1-7 When DM01 = 1 DMAC channel select 000-111: 0-7		
		23	22	21	20	19	18	17	16
(IMR02)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D002)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		31	30	29	28	27	26	25	24
(IMR03)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D003)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 000.		Must be set as 0.		Must be set as 000.		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR04	Bit Symbol	—	—		—	—	—		
(0xFFFF_D004)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		15	14	13	12	11	10	9	8
(IMR05)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D005)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		23	22	21	20	19	18	17	16
(IMR06)	Bit Symbol	—	EIM06		DM06	—	IL06		
(0xFFFF_D006)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Sensitivity of interrupt requests is set. When Sensitivity in CG is Edge, 10 shall be set, and when is Level, 00 shall be set.		DMAC trigger 0: Disable 1: Enable interrupt number 6 as DMAC trigger		When DM06 = 0 Interrupt number 6 (INT1) priority level 000: Interrupt disabled 001-111: 1-7 When DM06 = 1 DMAC channel select 000-111: 0-7		
		31	30	29	28	27	26	25	24
(IMR07)	Bit Symbol	—	EIM07		DM07	—	IL07		
(0xFFFF_D007)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Sensitivity of interrupt requests is set. When Sensitivity in CG is Edge, 10 shall be set, and when is Level, 00 shall be set.		DMAC trigger 0:Disable 1: Enable interrupt number 7 as DMAC trigger		When DM07 = 0 Interrupt number 7 (INT2) priority level 000: Interrupt disabled 001-111: 1-7 When DM07 = 1 DMAC channel select 000-111: 0-7		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR08	bit Symbol	—	EIM08		DM08	—	IL08		
(0xFFFF_D008)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Sensitivity of interrupt requests is set. When Sensitivity in CG is Edge, 10 shall be set, and when is Level, 00 shall be set.		DMAC trigger 0: Disable 1:Enable interrupt number 8 as DMAC trigger		When DM08 = 0 Interrupt number 8 (INT3) priority level 000: Interrupt disabled 001-111: 1-7 DM08 = 1 DMAC channel select 000-111: 0-7		
		15	14	13	12	11	10	9	8
(IMR09)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D009)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		23	22	21	20	19	18	17	16
(IMR10)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D00A)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		31	30	29	28	27	26	25	24
(IMR11)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D00B)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR12	Bit Symbol	—	—		—	—	—		
(0xFFFF_D00C)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		15	14	13	12	11	10	9	8
(IMR13)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D00D)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		23	22	21	20	19	18	17	16
(IMR14)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D00E)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		31	30	29	28	27	26	25	24
(IMR15)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D00F)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR16	Bit Symbol	—	—		—	—	—		
(0xFFFF_D010)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		15	14	13	12	11	10	9	8
(IMR17)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D011)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		23	22	21	20	19	18	17	16
(IMR18)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D012)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		31	30	29	28	27	26	25	24
(IMR19)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D013)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR20	Bit Symbol	—	EIM20		DM20	—	IL20		
(0xFFFF_D014)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0:Disable 1: Enable interrupt number 20 as DMAC trigger		When DM20 = 0 Interrupt number 20 (INTPMD0) priority level 000: Interrupt disabled 001-111: 1-7 When DM20 = 1 DMAC channel select 000-111: 0-7		
		15	14	13	12	11	10	9	8
(IMR21)	Git Symbol	—	EIM21		DM21	—	IL21		
(0xFFFF_D015)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 21 as DMAC trigger		When DM21 = 0 Interrupt number 21 (INTPMD1) priority level 000: Interrupt disabled 001-111: 1-7 When DM21 = 1 DMAC channel select 000-111: 0-7		
		23	22	21	20	19	18	17	16
(IMR22)	Bit Symbol	—	EIM22		DM22	—	IL22		
(0xFFFF_D016)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0:Disable 1: Enable interrupt number 22 as DMAC trigger		When DM22 = 0 Interrupt number 22 (INTEMG0) priority level 000: Interrupt disabled 001-111: 1-7 When DM22 = 1 DMAC channel select 000-111: 0-7		
		31	30	29	28	27	26	25	24
(IMR23)	Bit Symbol	—	EIM23		DM23	—	IL23		
(0xFFFF_D017)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 23 as DMAC trigger		When DM23 = 0 Interrupt number 23 (INTEMG0) priority level 000: Interrupt disabled 001-111: 1-7 When DM23 = 1 DMAC channel select 000-111: 0-7		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR24	Bit Symbol	—	EIM24		DM24	—	IL24		
(0xFFFF_D018)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 24 as DMAC trigger		When DM24 = 0 Interrupt number 24 (INTENC) priority level 000: Interrupt disabled 001-111: 1-7 When DM24 = 1 DMAC channel select 000-111: 0-7		
		15	14	13	12	11	10	9	8
(IMR25)	Bit Symbol	—	EIM25		DM25	—	IL25		
(0xFFFF_D019)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 25 as DMAC trigger		When DM25 = 0 Interrupt number 25 (INTTBCOM00) priority level 000: Interrupt disabled 001-111: 1-7 When DM25 = 1 DMAC channel select 000-111: 0-7		
		23	22	21	20	19	18	17	16
(IMR26)	Bit Symbol	—	EIM26		DM26	—	IL26		
(0xFFFF_D01A)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 26 as DMAC trigger		When DM26 = 0 Interrupt number 26 (INTTBCOM01) priority level 000: Interrupt disabled 001-111: 1-7 When DM26 = 1 DMAC channel select 000-111: 0-7		
		31	30	29	28	27	26	25	24
(IMR27)	Bit Symbol	—	EIM27		DM27	—	IL27		
(0xFFFF_D01B)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable intrrupt number 27 as DMAC trigger		When DM27 = 0 Interrupt number 27 (INTTBCOM10) priority level 000: Interrupt disabled 001-111: 1-7 When DM27 = 1 DMAC channel select 000-111: 0-7		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR28	Bit Symbol	—	EIM28		DM28	—	IL28		
(0xFFFF_D01C)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable intrrupt number 28 as DMAC trigger		When DM28 = 0 Interrupt number 28 (INTTBCOM11) priority level 000: Interrupt disabled 001-111: 1-7 When DM28 = 1 DMAC channel select 000-111: 0-7		
		15	14	13	12	11	10	9	8
(IMR29)	Bit Symbol	—	EIM29		DM29	—	IL29		
(0xFFFF_D01D)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 29 as DMAC trigger		When DM29 = 0 Interrupt number 29 (INTTBCOM20) priority level 000: Interrupt disabled 001-111: 1-7 When DM29 = 1 DMAC channel select 000-111: 0-7		
		23	22	21	20	19	18	17	16
(IMR30)	Bit Symbol	—	EIM30		DM30	—	IL30		
(0xFFFF_D01E)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 30 as DMAC trigger		When DM30 = 0 Interrupt number 30 (INTTBCOM21) priority level 000: Interrupt disabled 001-111: 1-7 When DM30 = 1 DMAC channel select 000-111: 0-7		
		31	30	29	28	27	26	25	24
(IMR31)	Bit Symbol	—	EIM31		DM31	—	IL31		
(0xFFFF_D01F)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 31 as DMAC trigger		When DM31 = 0 Interrupt number 31 (INTTBCOM30) priority level 000: Interrupt disabled 001-111: 1-7 When DM31 = 1 DMAC channel select 000-111: 0-7		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR32	Bit Symbol	—	EIM32		DM32	—	IL32		
(0xFFFF_D020)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Funcion		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 32 as DMAC trigger		When DM32 = 0 Interrupt number 32 (INTTBCOM31) priority level 000: Interrupt disabled 001-111: 1-7 When DM32 = 1 DMAC channel select 000-111: 0-7		
		15	14	13	12	11	10	9	8
(IMR33)	Bit Symbol	—	EIM33		DM33	—	IL33		
(0xFFFF_D021)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 33 as DMAC trigger		When DM33 = 0 Interrupt number (INTTBE0) priority level 000: Interrupt disabled 001-111: 1-7 When DM33 = 1 DMAC channel select 000-111: 0-7		
		23	22	21	20	19	18	17	16
(IMR34)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D022)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		31	30	29	28	27	26	25	24
(IMR35)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D023)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR36	Bit Symbol	—	—		—	—	—		
(0xFFFF_D024)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		15	14	13	12	11	10	9	8
(IMR37)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D025)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		23	22	21	20	19	18	17	16
(IMR38)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D026)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		31	30	29	28	27	26	25	24
(IMR39)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D027)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR40	Bit Symbol	—	—		—	—	—		
(0xFFFF_D028)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		15	14	13	12	11	10	9	8
(IMR41)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D029)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		23	22	21	20	19	18	17	16
(IMR42)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D02A)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		31	30	29	28	27	26	25	24
(IMR43)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D02B)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR44	Bit Symbol	—	—		—	—	—		
(0xFFFF_D02C)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		15	14	13	12	11	10	9	8
(IMR45)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D02D)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		23	22	21	20	19	18	17	16
(IMR46)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D02E)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		31	30	29	28	27	26	25	24
(IMR47)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D02F)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR48	Bit Symbol	—	EIM48		DM48	—	IL48		
(0xFFFF_D030)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 48 as DMAC trigger		When DM48 = 0 Intrrupt number 48 (INTTX0) priority level 000: Interrupt disabled 001-111: 1-7 When DM48 = 1 DMAC channel select 000-111: 0-7		
		15	14	13	12	11	10	9	8
(IMR49)	Bit Symbol	—	EIM49		DM49	—	IL49		
(0xFFFF_D031)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 49 as DMAC trigger		When DM49 = 0 Interrupt number 49(INTRX0) priority level 000: Interrupt disabled 001-111: 1-7 When DM49 = 1 DMAC channel select 000-111: 0-7		
		23	22	21	20	19	18	17	16
(IMR50)	Bit Symbol	—	EIM50		DM50	—	IL50		
(0xFFFF_D032)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 50 as DMAC trigger		When DM50 = 0 Interrupt number 50 (INTTX1) priority level 000: Interrupt disabled 001-111: 1-7 When DM50 = 1 DMAC channel select 000-111: 0-7		
		31	30	29	28	27	26	25	24
(IMR51)	Bit Symbol	—	EIM51		DM51	—	IL51		
(0xFFFF_D033)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 51 as DMAC trigger		When DM51 = 0 Interrupt number 51(INTRX1) priority level 000: Interrupt disabled 001-111: 1-7 When DM51 = 1 DMAC channel select 000-111: 0-7		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR52	Bit Symbol	—	EIM52		DM52	—	IL52		
(0xFFFF_D034)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 52 as DMAC trigger		When DM52 = 0 Intrrupt number 52 (INTTX2) priority level 000: Interrupt disabled 001-111: 1-7 When DM52 = 1 DMAC channel select 000-111: 0-7		
		15	14	13	12	11	10	9	8
(IMR53)	Bit Symbol	—	EIM53		DM53	—	IL53		
(0xFFFF_D035)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 53 as DMAC trigger		When DM53 = 0 Interrupt number 53 (INTRX2) priority level 000: Interrupt disabled 001-111: 1-7 When DM53 = DMAC channel select 000-111: 0-7		
		23	22	21	20	19	18	17	16
(IMR54)	Bit Symbol	—	EIM54		DM54	—	IL54		
(0xFFFF_D036)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 54 as DMAC trigger		When DM54 = 0 Interrupt number 54 (INTTX3) priority level 000: Interrupt disabled 001-111: 1-7 When DM54 = 1 DMAC channel select 000-111: 0-7		
		31	30	29	28	27	26	25	24
(IMR55)	Bit Symbol	—	EIM55		DM55	—	IL55		
(0xFFFF_D037)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 55 as DMAC trigger		When DM55 = 0 Interrupt number 55 (INTRX3) priority level 000: Interrupt disabled 001-111: 1-7 When DM55 = 1 DMAC channel select 000-111: 0-7		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR56	Bit Symbol	—	EIM56		DM56	—	IL56		
(0xFFFF_D038)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 56 as DMAC trigger		When DM56 = 0 Interrupt number 56 (INTDMA0) peiority level 000: Interrupt disabled 001-111: 1-7 When DM56 = 1 DMAC channel select 000-111: 0-7		
		15	14	13	12	11	10	9	8
(IMR57)	Bit Symbol	—	EIM57		DM57	—	IL57		
(0xFFFF_D039)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 57 as DMAC trigger		When DM57 = 0 Interrupt number 57 (INTDMA1) priority level 000: Interrupt disabled 001-111: 1-7 When DM57 = 1 DMAC channel select 000-111: 0-7		
		23	22	21	20	19	18	17	16
(IMR58)	Bit Symbol	—	EIM58		DM58	—	IL58		
(0xFFFF_D03A)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 58 as DMAC trigger		When DM58 = 0 Interrupt number 58 （INTDMA2） priority level 000: Interrupt disabled 001-111: 1-7 When DM58 = 1 DMAC channel select 000-111: 0-7		
		31	30	29	28	27	26	25	24
(IMR59)	Bit Symbol	—	EIM59		DM59	—	IL59		
(0xFFFF_D03B)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 59 as DMAC trigger		When DM59 = 0 Interrupt number 59 (INTDMA3) priority level 000: Interrupt disabled 001-111: 1-7 When DM59 = 1 DMAC channel select 000-111: 0-7		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR60	Bit Symbol	—	EIM60		DM60	—	IL60		
(0xFFFF_D03C)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 60 as DMAC trigger		When DM60 = 0 Interrupt number 60 (INTDMA4) priority level 000: Interrupt disabled 001-111: 1-7 When DM60 = 1 DMAC channel select 000-111: 0-7		
		15	14	13	12	11	10	9	8
(IMR61)	Bit Symbol	—	EIM61		DM61	—	IL61		
(0xFFFF_D03D)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 61 as DMAC trigger		When DM61 = 0 Interrupt number 61(INTDMA5) priority level 000: Interrupt disabled 001-111: 1-7 When DM61 = 1 DMAC channel select 000-111: 0-7		
		23	22	21	20	19	18	17	16
(IMR62)	Bit Symbol	—	EIM62		DM62	—	IL62		
(0xFFFF_D03E)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Funcion		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 62 as DMAC trigger		When DM62 = 0 Interrupt number 62 (INTDMA6) priority level 000: Interrupt disabled 001-111: 1-7 When DM62 = 1 DMAC channel select 000-111: 0-7		
		31	30	29	28	27	26	25	24
(IMR63)	Bit Symbol	—	EIM63		DM63	—	IL63		
(0xFFFF_D03F)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 63 as DMAC trigger		When DM63 = 0 Interrupt number 63 (INTDMA7) priority level 000: Interrupt disabled 001-111: 1-7 When DM63 = 1 DMAC channel select 000-111: 0-7		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR64	Bit Symbol	—	—		—	—	—		
(0xFFFF_D040)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		15	14	13	12	11	10	9	8
(IMR65)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D041)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		23	22	21	20	19	18	17	16
(IMR66)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D042)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0		Must be set as 000.		
		31	30	29	28	27	26	25	24
(IMR67)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D043)	Read/Write	R	R/W			R	R/W		
	Reset Vaue	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR68	bit Symbol	—	EIM68		DM68	—	IL68		
(0xFFFF_D044)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 68 as DMAC trigger		When DM68 = 0 Interrupt number 68 (INTAD0) interrupt level 000: Interrupt disabled 001-111: 1-7 When DM68 = 1 DMAC channel select 000-111: 0-7		
		15	14	13	12	11	10	9	8
(IMR69)	Bit Symbol	—	EIM69		DM69	—	IL69		
(0xFFFF_D045)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 69 as DMAC trigger		When DM69 = 0 Interrupt number 69 (INTADHP0) priority level 000: Interrupt disable 001-111: 1-7 When DM69 = 1 DMAC channel select 000-111: 0-7		
		23	22	21	20	19	18	17	16
(IMR70)	Bit Symbol	—	EI70		DM70	—	IL70		
(0xFFFF_D046)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 70 as DMAC trigger		When DM70 = 0 Interrupt number 70 (INTADM0) peiority level 000: Interrupt disabled 001-111: 1-7 When DM70 = 1 DMAC channel select 000-111: 0-7		
		31	30	29	28	27	26	25	24
(IMR71)	Bit Symbol	—	EIM71		DM71	—	IL71		
(0xFFFF_D047)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 71 as DMAC trigger		When DM71 = 0 Interrupt number 71 (INTAD1) priority level 000: Interrupt disabled 001-111: 1-7 When DM71 = 1 DMAC channel select 000-111: 0-7		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR72	Bit Symbol	—	EIM72		DM72	—	IL72		
(0xFFFF_D048)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 72 as DMAC trigger		When DM72 = 0 Interrupt number 72 (INTADHP1) priority level 000: Interrupt disabled 001-111: 1-7 When DM72 = 1 DMAC channel select 000-111: 0-7		
		15	14	13	12	11	10	9	8
(IMR73)	Bit Symbol	—	EIM73		DM73	—	IL73		
(0xFFFF_D049)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1:Enable interrupt number 73 as DMAC trigger		When DM73 = 0 Interrupt number 73 (INTADM1) priority level 000: Interrupt disabled 001-111: 1-7 When DM73 = 1 DMAC channel select 000-111: 0-7		
		23	22	21	20	19	18	17	16
(IMR74)	Bit Symbol	—	EI74		DM74	—	IL74		
(0xFFFF_D04A)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 00: Level “L” 01: Level “H” 10: Rising edge 11: Falling edge		DMAC trigger 0: Disable 1: Enable interrupt number 74 as DMAC trigger		When DM74 = 0 Interrupt number 74 (INT4) priority level 000: Interrupt disabled 001-111: 1 1-7 When DM74 = 1 DMAC channel select 000-111: 0-7		
		31	30	29	28	27	26	25	24
(IMR75)	Bit Symbol	—	EIM75		DM75	—	IL75		
(0xFFFF_D04B)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 00: Level “L” 01: Level “H” 10: Rising edge 11: Falling edge		DMAC trigger 0: Disable 1: Enable interrupt number 75 as DMAC trigger		When DM75 = 0 Interrupt number 75 (INT5) priority level 000: Interrupt disabled 001-111: 1-7 When DM75 = 1 DMAC channel select 000-111: 0-7		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR76	Bit Symbol	—	EI76		DM76	—	IL76		
(0xFFFF_D04C)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 00: Level “L” 01: Level “H” 10: Rising edge 11: Falling edge		DMAC trigger 0: Disable 1: Enable interrupt number 76 as DMAC trigger		When DM76 = 0 Interrupt number 76 (INT6) priority level 000: Interrupt disabled 001-111: 1-7 When DM76 = 1 DMAC channel select 000-111: 0-7		
		15	14	13	12	11	10	9	8
(IMR77)	Bit Symbol	—	EI77		DM77	—	IL77		
(0xFFFF_D04D)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 00: Level “L” 01: Level “H” 10: Rising edge 11: Falling edge		DMAC trigger 0:Disable 1: Enable interrupt number 77 as DMAC trigger		When DM77 = 0 Interrupt number 77 (INT7) priorityl evel 000: Interrupt disabled 001-111: 1-7 When DM77 = 1 DMAC channel select 000-111: 0-7		
		23	22	21	20	19	18	17	16
(IMR78)	Bit Symbol	—	EI78		DM78	—	IL78		
(0xFFFF_D04E)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 00: Level “L” 01: Level “H” 10: Rising edge 11: Falling edge		DMAC trigger 0: Disable 1: Enable interrupt number 78 as DMAC trigger		When DM78 = 0 Interrupt number 78 (INT8) priority level 000: Interrupt disabled 001-111: 1-7 When DM78 = 1 DMAC channel select 000-111: 0-7		
		31	30	29	28	27	26	25	24
(IMR79)	Bit Symbol	—	EI79		DM79	—	IL79		
(0xFFFF_D04F)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 00: Level “L” 01: Level “H” 10: Rising edge 11: Falling edge		DMAC trigger 0: Disable 1: Enable interrupt number 79 as DMAC trigger		When DM79 = 0 Interrupt number 79 (INT9) priority level 000: Interrupt disabled 001-111: 1-7 When DM79 = 1 DMAC channel select 000-111: 0-7		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR80	Bit Symbol	—	—		—	—	—		
(0xFFFF_D050)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		15	14	13	12	11	10	9	8
(IMR81)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D051)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		23	22	21	20	19	18	17	16
(IMR82)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D052)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		31	30	29	28	27	26	25	24
(IMR83)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D053)	Read/Write	R	R/W			R	R/W		
	Reset Vaue	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR84	Bit Symbol	—	EIM84		DM84	—	IL84		
(0xFFFF_D054)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 84 as DMAC trigger		When DM84 = 0 Interrupt number 84 (INTTBCAP00) priority level 000: Interrupt disabled 001-111: 1-7 When DM84 = 1 DMAC channel select 000-111: 0-7		
		15	14	13	12	11	10	9	8
(IMR85)	Bit Symbol	—	EIM85		DM85	—	IL85		
(0xFFFF_D055)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 85 as DMAC trigger		When DM85 = 0 Interrupt number 85 (INTTBCAP01) priority level 000: Interrupt disabled 001-111: 1-7 When DM85 = 1 DMAC channel select 000-111: 0-7		
		23	22	21	20	19	18	17	16
(IMR86)	Bit Symbol	—	EIM86		DM86	—	IL86		
(0xFFFF_D056)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 86 as DMAC trigger		When DM86 = 0 Interrupt number 86 (INTTBCAP10) priority level 000: Interrupt disabled 001-111: 1-7 When DM86 = 1 DMAC channel select 000-111: 0-7		
		31	30	29	28	27	26	25	24
(IMR87)	Bit Symbol	—	EIM87		DM87	—	IL87		
(0xFFFF_D057)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 87 as DMAC trigger		When DM87 = 0 Interrupt number 87 (INTTBCAP11) priority level 000: Interrupt disabled 001-111: 1-7 When DM87 = 1 DMAC channel select 000-111: 0-7		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR88	bit Symbol	—	EIM88		DM88	—	IL88		
(0xFFFF_D058)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 88 as DMAC trigger		When DM88 = 0 Interrupt number 88 (INTTBCAP20) priority level 000: Interrupt disabled 001-111: 1-7 When DM88 = 1 DMAC channel select 000-111: 0-7		
		15	14	13	12	11	10	9	8
(IMR89)	Bit Symbol	—	EIM89		DM89	—	IL89		
(0xFFFF_D059)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 89 as DMAC trigger		When DM89 = 0 Interrupt number 89 (INTTBCAP21) priority level 000: Interrupt disabled 001-111: 1-7 When DM89 = 1 DMAC channel select 000-111: 0-7		
		23	22	21	20	19	18	17	16
(IMR90)	Bit Symbol	—	EIM90		DM90	—	IL90		
(0xFFFF_D05A)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 90 as DMAC trigger		When DM90 = 0 Interrupt number 90 (INTTBCAP30) priority level 000: Interrupt disabled 001-111: 1-7 When DM90 = 1 DMAC channel select 000-111: 0-7		
		31	30	29	28	27	26	25	24
(IMR91)	Bit Symbol	—	EIM91		DM91	—	IL91		
(0xFFFF_D05B)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Set Sensitivity of interrupt request. 10 must be set to it.		DMAC trigger 0: Disable 1: Enable interrupt number 91 as DMAC trigger		When DM91 = 0 Interrupt number 91 (INTTBCAP31) priority level 000: Interrupt disabled 001-111: 1-7 When DM91 = 1 DMAC channel select 000-111: 0-7		

Not Recommended for New Design

Interrupt Mode Control Registers

		7	6	5	4	3	2	1	0
IMR92	Bit Symbol	—	—		—	—	—		
(0xFFFF_D05C)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		15	14	13	12	11	10	9	8
(IMR93)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D05D)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		23	22	21	20	19	18	17	16
(IMR94)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D05E)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		
		31	30	29	28	27	26	25	24
(IMR95)	Bit Symbol	—	—		—	—	—		
(0xFFFF_D05F)	Read/Write	R	R/W			R	R/W		
	Reset Value	0	00		0	0	000		
	Function		Must be set as 00.		Must be set as 0.		Must be set as 000.		

Not Recommended for New Design

7.8.10.5 Interrupt Request Clear Register (ICLR)

By setting IVR[8:0] of interrupt source whose request is desired to clear to ICLR, an interrupt request suspended can be cleared. As an interrupt request is cleared, IVR values also are cleared, thus no determination of interrupt sources can be made. Interrupt requests must never be cleared before reading IVR values.

Interrupt Request Clear Register

ICLR (0xFFFF_D084)

	7	6	5	4	3	2	1	0
Bit Symbol	IV							
Read/Write	W							
Reset Value	—	—	—	—	—	—	—	—
Function	Set the values in IVR[8:0] of sources to the interrupts whose request is desired to clear.							
	15	14	13	12	11	10	9	8
Bit Symbol	—	—	—	—	—	—	—	IV
Read/Write	R							W
Reset Value	0	0	0	0	0	0	0	—
Function								

Note 1: This register must be accessed in 16 bits.

Note 2: Regardless of Sensitivity setting of IMRxx<EIMxx> of INTC, which may be level "H"/"L" or rising/faling edge, interrupt request shall be cleared to retain its interrupt source.

Note 3: This register is not accessible with any bit manipulation instruction.

Note 4: No external transfer request caused by interrupt sources of DMAC is cleared. An external transfer request once accepted is not cancelled until DMA transfer is executed. Therefore, to clear unnecessary external transfer request, DMA transfer execution, disabling interrupt in IMRxx<ILxx> before accepting, or cancelling a starting source of DMAC in IMRxx<DMxx> is required.

Not Recommended for New Design

7.8.10.6 Mode Control Register（MODECR）

Bus Error exceptions are not generated by store instructions or write accesses by the DMAC. By setting a 0 in the BERCTL bit of the MODECR, a NMI can be generated when the bus error area is accessed by a store instruction or a write access by the DMAC.

Mode Control Register

MODECR (0xFFFF_D400)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	—	—	—
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0
Function								
	15	14	13	12	11	10	9	8
Bit Symbol	—	—	—	—	—	—	—	—
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0
Function								
	23	22	21	20	19	18	17	16
Bit Symbol	—	—	—	—	—	—	—	BERCTL
Read/Write	R					R/W		
Reset Value	0	0	0	0	0	1	1	1
Function						Must be set as 1.	Must be set as 1.	Bus error by store access 0: NMI generated 1: NMI not generated
	31	30	29	28	27	26	25	24
Bit Symbol	—	—	—	—	—	—	—	—
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0
Function								

Note: This register must be accessed as a 32-bit quantity.

Not Recommended for New Design

7.9 Usage Note of Interrupt

Cautions and warnings upon using interrupts are described here. A user program must be programmed, meeting the requirements below.

7.9.1 TX19A Processor Core

- Since TMP19A71 has no external bus interface, no interrupt can be used by setting 0 to Status<BEV> of CP0 register.
- Exceptions cannot be disabled. Note that some of them have two types of instructions whose differences are only Generated Exception or Non-generated. Use them as usage.
- Software Sets of software interrupt and hardware interrupt sources are different interrupt source.
- Place two NOP instructions immediately after rewriting SSCR of CP0 register because it takes two clocks to change a register bank.
- When the interrupt requests of the same level are accepted simultaneously by changing ILEV<CMASK>, it is necessary to save in user program since register banks do not switch.
- IER of CP0 register is only accessible from 32-bit ISA.
- Stack pointers (r29) needs to be set twice since they are distinguished as Shadow Register Set number 0 and Shadow Register Set number from 1 to7. Using Shadow Register Set number 1 by setting 1 to SSCR<CSS> in main processing is the way to use a common stack pointer. In this meshod, it is necessary to save in user program because no register bank is switched even if an interrupt of level 1 is accepted.
- If an ERET instruction is executed while interrupts are disabled by setting 1 to Status<ERL> of the CP0 register, it restores ErrorEPC of CP0 register in main processing as a restoring address. Since TX19A processor core saves the interrupt restoring address in EPC, it is necessary to be careful with disabling interrupts in Status<ERL>.
- Do not execuse ERET instruction within two clocks after accessing Status, ErrorEPC, EPC, or SSCR of CP0 register.
- When disabling an interrupt by setting Status<ERL/EXL/IE> of CP0 register, the interrupt becomes disabled at the instruction execution point (Stage E) while the value set to the register becomes effective two clocks later.
- When enabling an interrupt by setting Status<ERL/EXL/IE> of CP0 register, it becomes enabled two clocks after the instruction execution point (Stage E), and the value set to the register also becomes effective two clocks after the instruction execution point (Stage E).
- TMP19A71 has two types of register number: r9 (SEL6) which is accessible with 32-bit ISA only and r22 (SEL0) which is asscessible with 32-bit/16-bit ISA. In both cases, it turns out to be the same result. To use the register number r9 (SEL6) with Toshiba's C compiler, specify -tx19_sscr9 as a compiling option. For details, refer to the additional documents of Toshiba C compiler, *TX19A C Compilier Reference*.

Not Recommended for New Design

7.9.2 INTC

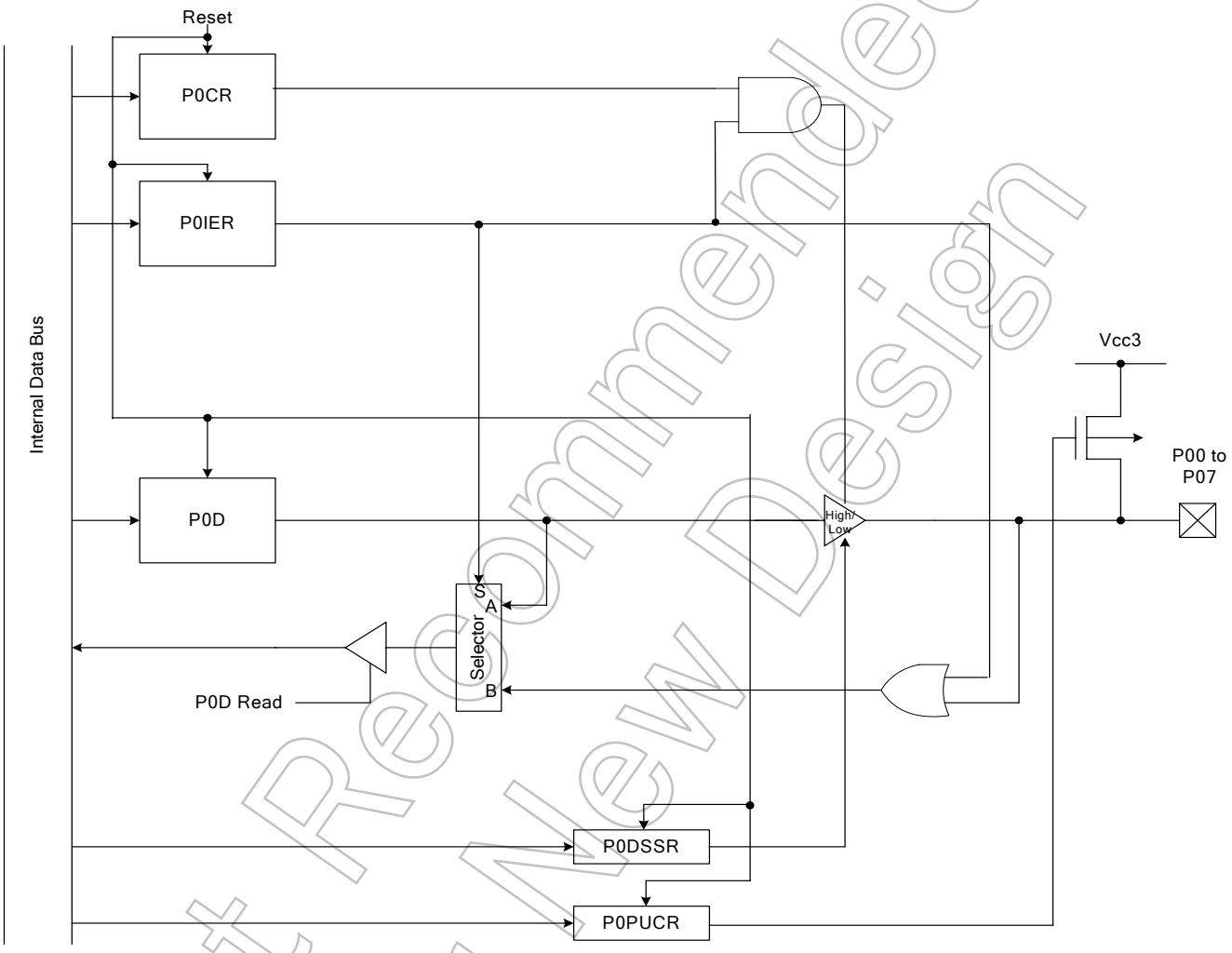
- When there are two or more interrupt requests of the same level, the acceptance is made on a priority basis from the sources of the smallest interrupt number.
- Interrupt sources of level 0 is not suspended.
- To disable an interrupt source (interrupt level 0) individually, disable it in Interrupt Disabled state.
- Initial values of IMRxx<EIMxx> of INTC and setting value may be different.
- ILEV of INTC must be accessed in 32-bit quantity.
- ICLR of INTC must be accessed in 16-bit quantity.
- When an interrupt request is cleared in ICLR before reading IVR value of INTC, IVR value is cleared and interrupt sources cannot be distinguished.
- To enable an interrupt, it must be set in the detection order (from outside to inside) and to disable it, in reverse of the detection order (from inside to outside). If not, unexpected interrupt may be generated or unexpected transfer of EMG state may occur. To prevent such cases, interrupt sources or EMG state must be cleared before enabling interrupts.
- To rewrite ILEV<CMASK> values of INTC, set 1 to <MLEV> simultaneously.

Not Recommended for New Design

8. I/O Ports

8.1 Port 0 (P00 to P07)

Port 0 pins can be individually programmed to function as discrete general-purpose I/O pins.



Note: The selectors in the figure output input A when S=1 and input B when S=0.

Figure 8.1.1 Port 0 (P00 to P07)

Not Recommended for New Design

Port 0 Register

P0D (0xFFFF_C000)

	7	6	5	4	3	2	1	0
Bit Symbol	P0D7	P0D6	P0D5	P0D4	P0D3	P0D2	P0D1	P0D0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	Port 0 output data (Output latch)							

Note: When P0IER=0, the port state can be read from this register.

Port 0 Control Register

P0CR (0xFFFF_C004)

	7	6	5	4	3	2	1	0
Bit Symbol	P0CR7	P0CR6	P0CR5	P0CR4	P0CR3	P0CR2	P0CR1	P0CR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Output disabled 1: Output enabled							

Port 0 Input Enable Register

P0IER (0xFFFF_C008)

	7	6	5	4	3	2	1	0
Bit Symbol	P0IER7	P0IER6	P0IER5	P0IER4	P0IER3	P0IER2	P0IER1	P0IER0
Read/Write	R/W							
Reset Value	1	1	1	1	1	1	1	1
Function	0: Input enabled 1: Input disabled							

Port 0 Drive Strength Register

P0DSSR (0xFFFF_C00C)

	7	6	5	4	3	2	1	0
Bit Symbol	P0DSSR7	P0DSSR6	P0DSSR5	P0DSSR4	P0DSSR3	P0DSSR2	P0DSSR1	P0DSSR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Low drive capability 1: High drive capability							

Note: The current flowing through ports should not exceed the maximum rating.

Port 0 Pull-Up Control Register

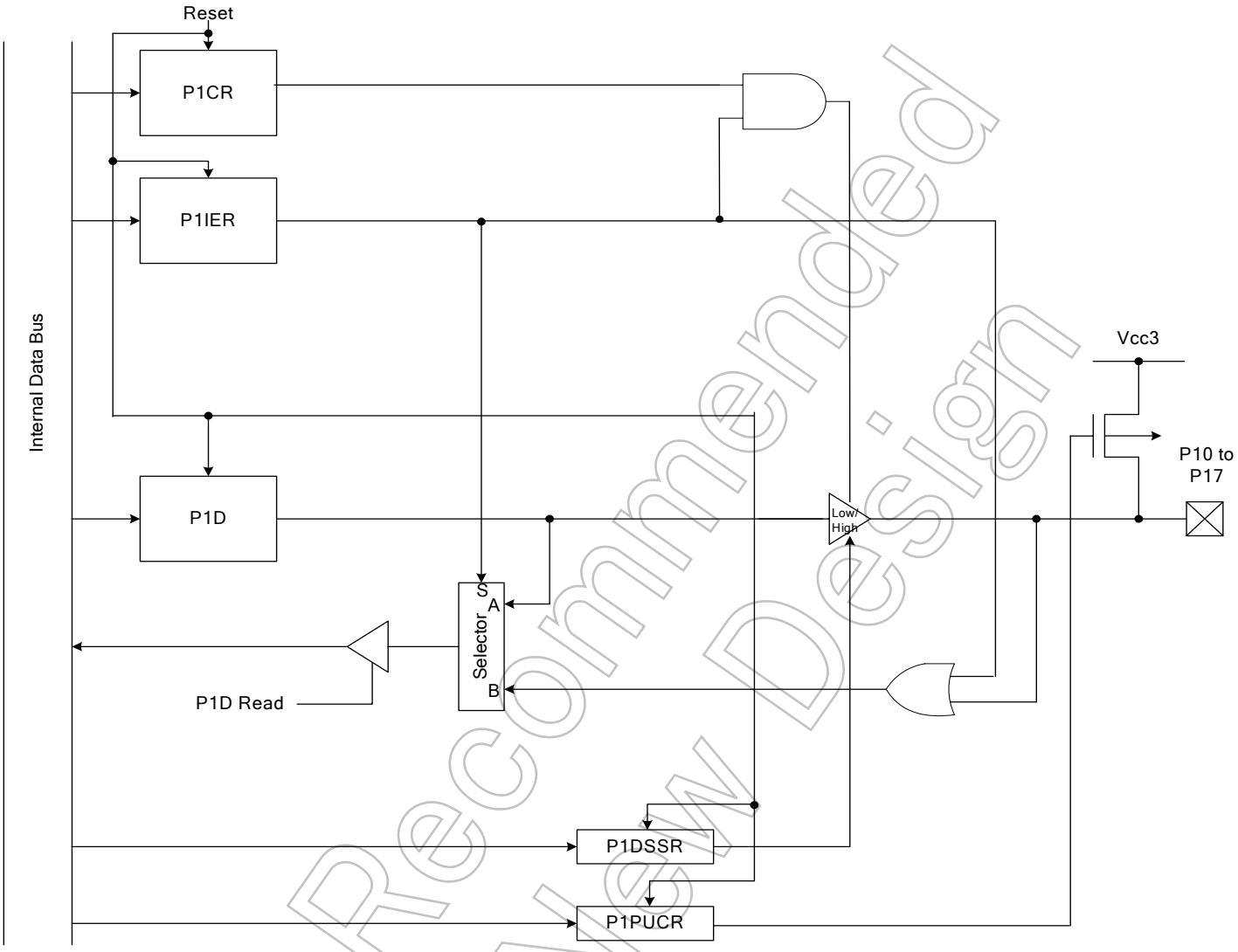
P0PUCR (0xFFFF_C014)

	7	6	5	4	3	2	1	0
Bit Symbol	P0PUCR7	P0PUCR6	P0PUCR5	P0PUCR4	P0PUCR3	P0PUCR2	P0PUCR1	P0PUCR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Pull-up disabled 1: Pull-up enabled							

Not Recommended for New Design

8.2 Port 1 (P10 to P17)

Eight Port 1 pins can be individually programmed to function as discrete general-purpose I/O pins.



Note: The selectors in the figure output input A when S=1 and input B when S=0.

Figure 8.2.1 Port 1 (P10 to P17)

Not Recommended for New Design

Port 1 Register

P1D (0xFFFF_C040)

	7	6	5	4	3	2	1	0
Bit Symbol	P1D7	P1D6	P1D5	P1D4	P1D3	P1D2	P1D1	P1D0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	Port 1 output data (Output latch)							

Note: When P1IER=0, the port state can be read from this register.

Port 1 Control Register

P1CR (0xFFFF_C044)

	7	6	5	4	3	2	1	0
Bit Symbol	P1CR7	P1CR6	P1CR5	P1CR4	P1CR3	P1CR2	P1CR1	P1CR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Output disabled 1: Output enabled							

Port 1 Input Enable Register

P1IER (0xFFFF_C048)

	7	6	5	4	3	2	1	0
Bit Symbol	P1IER7	P1IER6	P1IER5	P1IER4	P1IER3	P1IER2	P1IER1	P1IER0
Read/Write	R/W							
Reset Value	1	1	1	1	1	1	1	1
Function	0: Input enabled 1: Input disabled							

Port 1 Drive Strength Register

P1DSSR (0xFFFF_C04C)

	7	6	5	4	3	2	1	0
Bit Symbol	P1DSSR7	P1DSSR6	P1DSSR5	P1DSSR4	P1DSSR3	P1DSSR2	P1DSSR1	P1DSSR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Low drive capability 1: High drive capability							

Note: The current flowing through ports should not exceed the maximum ratings for each port pin and for all the port pins.

Port 1 Pull-Up Control Register

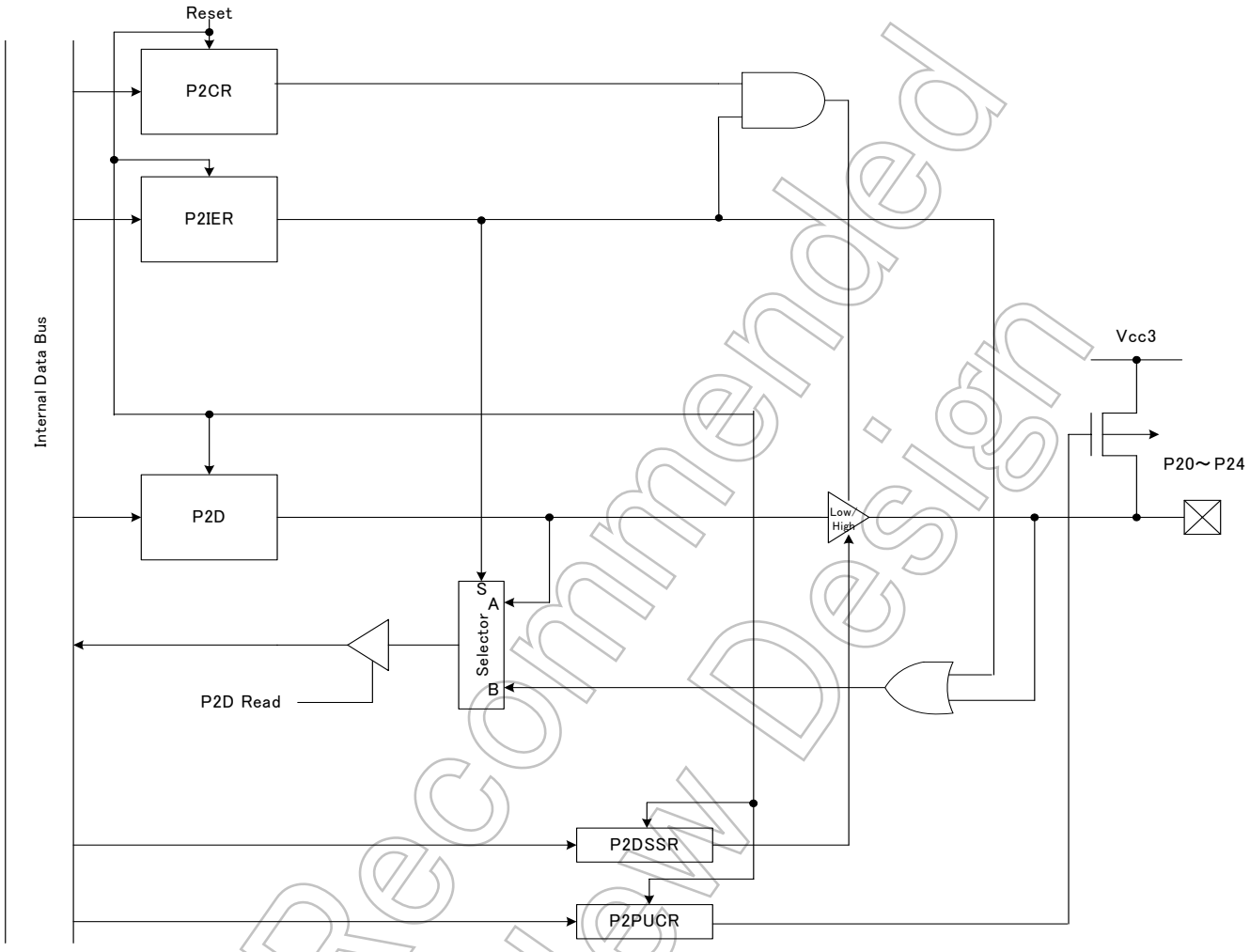
P1PUCR (0xFFFF_C054)

	7	6	5	4	3	2	1	0
Bit Symbol	P1PUCR7	P1PUCR6	P1PUCR5	P1PUCR4	P1PUCR3	P1PUCR2	P1PUCR1	P1PUCR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Pull-up disabled 1: Pull-up enabled							

Not Recommended for New Design

8.3 Port 2 (P20 to P24)

Five Port 2 pins can be individually programmed to function as discrete general-purpose I/O pins.



Note: The selectors in the figure output input A when S=1 and input B when S=0.

Figure 8.3.1 Port 2 (P20 to P24)

Not Recommended for New Design

Port 2 Register

P2D (0xFFFF_C080)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	P2D4	P2D3	P2D2	P2D1	P2D0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	Port 2 output data (Output latch)							

Note: When P2IER=0, the port state can be read from this register.

Port 2 Control Register

P2CR (0xFFFF_C084)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	P2CR4	P2CR3	P2CR2	P2CR1	P2CR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Output disabled 1: Output enabled							

Port 2 Input Enable Register

P2IER (0xFFFF_C088)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	P2IER4	P2IER3	P2IER2	P2IER1	P2IER0
Read/Write	R/W							
Reset Value	0	0	0	1	1	1	1	1
Function	0: Input enabled 1: Input disabled							

Port 2 Drive Strength Register

P2DSSR (0xFFFF_C08C)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	P2DSSR4	P2DSSR3	P2DSSR2	P2DSSR1	P2DSSR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Low drive capability 1: High drive capability							

Note: The current flowing through ports should not exceed the maximum ratings for each port pin and for all the port pins.

Port 2 Pull-Up Control Register

P2PUCR (0xFFFF_C094)

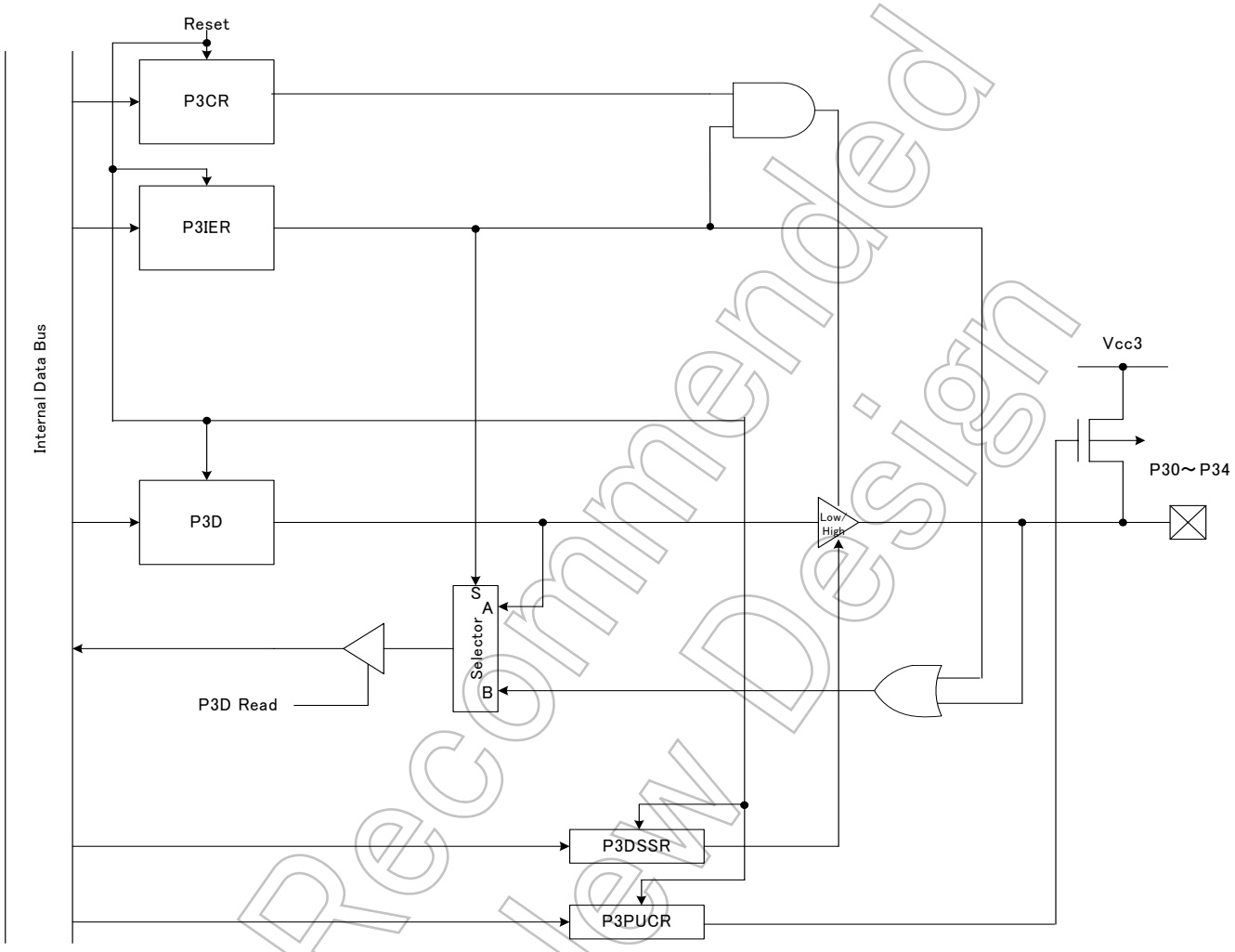
	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	P2PUCR4	P2PUCR3	P2PUCR2	P2PUCR1	P2PUCR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Pull-up disabled 1: Pull-up enabled							

Note: In DSU (EJTAG) mode, Port 2 pins function as DSU control pins and the P2D, P2CR, P2IER, P2DDSR and P2PUCR are invalid.

Not Recommended for New Design

8.4 Port 3 (P30 to P34)

Five Port 3 pins can be individually programmed to function as discrete general-purpose I/O pins. Figure 8.4.1 shows the configuration of Port 3 when not used in DSU (EJTAG) mode.



Note: The selectors in the figure output input A when S=1 and input B when S=0.

Figure 8.4.1 Port 3 (P30 to P34)

Not Recommended for New Design

Port 3 Register

		7	6	5	4	3	2	1	0
P3D	Bit Symbol	—	—	—	P3D4	P3D3	P3D2	P3D1	P3D0
(0xFFFF_C0C0)	Read/Write	R/W							
	Reset Value	0	0	0	0	0	0	0	0
	Function				Port 3 output data (Output latch)				

Note: When P3IER=0, the port state can be read from this register.

Port 3 Control Register

		7	6	5	4	3	2	1	0
P3CR	Bit Symbol	—	—	—	P3CR4	P3CR3	P3CR2	P3CR1	P3CR0
(0xFFFF_C0C4)	Read/Write	R/W							
	Reset Value	0	0	0	0	0	0	0	0
	Function				0: Output disabled 1: Output enabled				

Port 3 Input Enable Register

		7	6	5	4	3	2	1	0
P3IER	Bit Symbol	—	—	—	P3IER4	P3IER3	P3IER2	P3IER1	P3IER0
(0xFFFF_C0C8)	Read/Write	R/W							
	Reset Value	0	0	0	1	1	1	1	1
	Function				0: Input enabled 1: Input disabled				

Port 3 Drive Strength Register

		7	6	5	4	3	2	1	0
P3DSSR	Bit Symbol	—	—	—	P3DSSR4	P3DSSR3	P3DSSR2	P3DSSR1	P3DSSR0
(0xFFFF_C0CC)	Read/Write	R/W							
	Reset Value	0	0	0	0	0	0	0	0
	Function				0: Low drive capability 1: High drive capability				

Note: The current flowing through ports should not exceed the maximum ratings for each port pin and for all the port pins.

Port 3 Pull-Up Control Register

		7	6	5	4	3	2	1	0
P3PUCR	Bit Symbol	—	—	—	P3PUCR4	P3PUCR3	P3PUCR2	P3PUCR1	P3PUCR0
(0xFFFF_C0D4)	Read/Write	R/W							
	Reset Value	0	0	0	0	0	0	0	0
	Function				0: Pull-up disabled 1: Pull-up enabled				

Note: In Level-1 DSU (EJTAG) mode, Port 3 pins function as DSU control pins and the P3D, P3CR, P3IER, P3DSSR and P3PUCR are invalid.

Not Recommended for New Design

8.5 Port 5 (P50 to P57)

Eight Port 5 pins are input-only pins that can also function as the analog input pins of the AD converter (ADC).

Note 1: As Port 5 uses AVCC0 as its I/O power source, it must be connected with the 3.3 V source even if ADC0 is not used.

Note 2: When Port 5 is not used as analog input pins, the AD conversion accuracy of ADC0 may deteriorate by a few LSBs. Be sure to check that this poses no problem on your system.

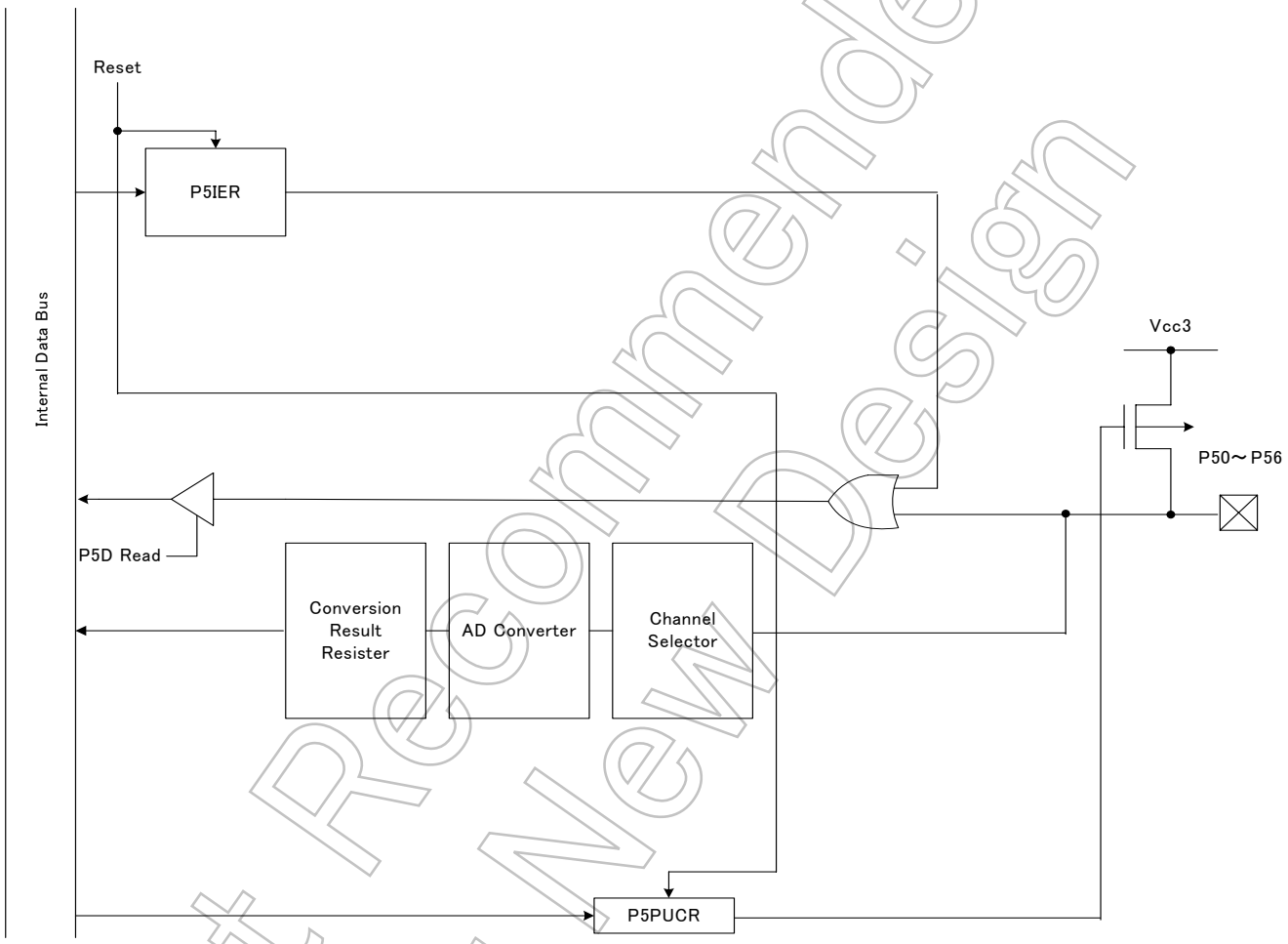


Figure 8.5.1 Port 5 (P50 to P56)

Not Recommended for New Design

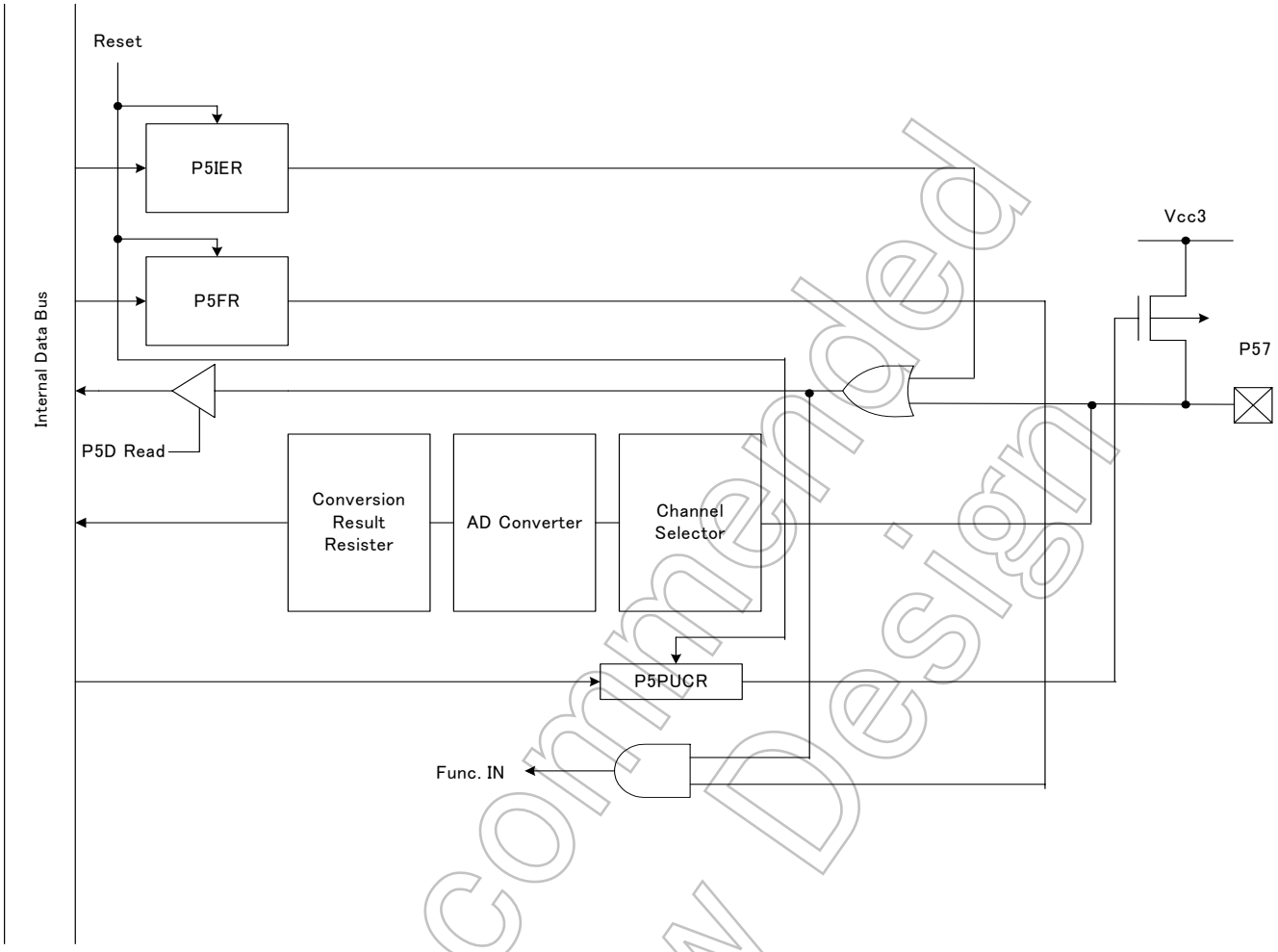


Figure 8.5.2 Port 5 (P57)

Not Recommended for New Design

Port 5 Register

P5D (0xFFFF_C140)

	7	6	5	4	3	2	1	0
Bit Symbol	P5D7	P5D6	P5D5	P5D4	P5D3	P5D2	P5D1	P5D0
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0
Function	Port 5 input data							

Note: When P5IER=0, the port state can be read from this register.

Port 5 Input Enable Register

P5IER (0xFFFF_C148)

	7	6	5	4	3	2	1	0
Bit Symbol	P5IER7	P5IER6	P5IER5	P5IER4	P5IER3	P5IER2	P5IER1	P5IER0
Read/Write	R/W							
Reset Value	1	1	1	1	1	1	1	1
Function	0: Input enabled 1: Input disabled							

Port 5 Pull-Up Control Register

P5PUCR (0xFFFF_C154)

	7	6	5	4	3	2	1	0
Bit Symbol	P5PUCR7	P5PUCR6	P5PUCR5	P5PUCR4	P5PUCR3	P5PUCR2	P5PUCR1	P5PUCR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Pull-up disabled 1: Pull-up enabled							

Port 5 Function Register

P5FR (0xFFFF_C158)

	7	6	5	4	3	2	1	0
Bit Symbol	P5FR7	—	—	—	—	—	—	—
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Port/AD input 1:ADTRG0							

Not Recommended for New Design

8.6 Port 6 (P60 to P67)

The lower 4 bits are input-only pins, and the upper 4 bits can be individually programmed to function as discrete general-purpose I/O pins shared with the analog input pins of the AD converter (ADC).

Note 1: As Port 6 uses AVCC1 as its I/O power source, it must be connected to the 3.3 V source even if ADC1 is not used.

Note 2: When Port 6 is not used as analog input pins, the AD conversion accuracy of ADC1 may deteriorate by a few LSBs. When Port 6 is used as an output port, this may result in a noticeable deterioration in AD conversion accuracy which may exceed the worst conditions presented in the AD conversion characteristics later in this manual. Be sure to check that this poses no problem on your system.

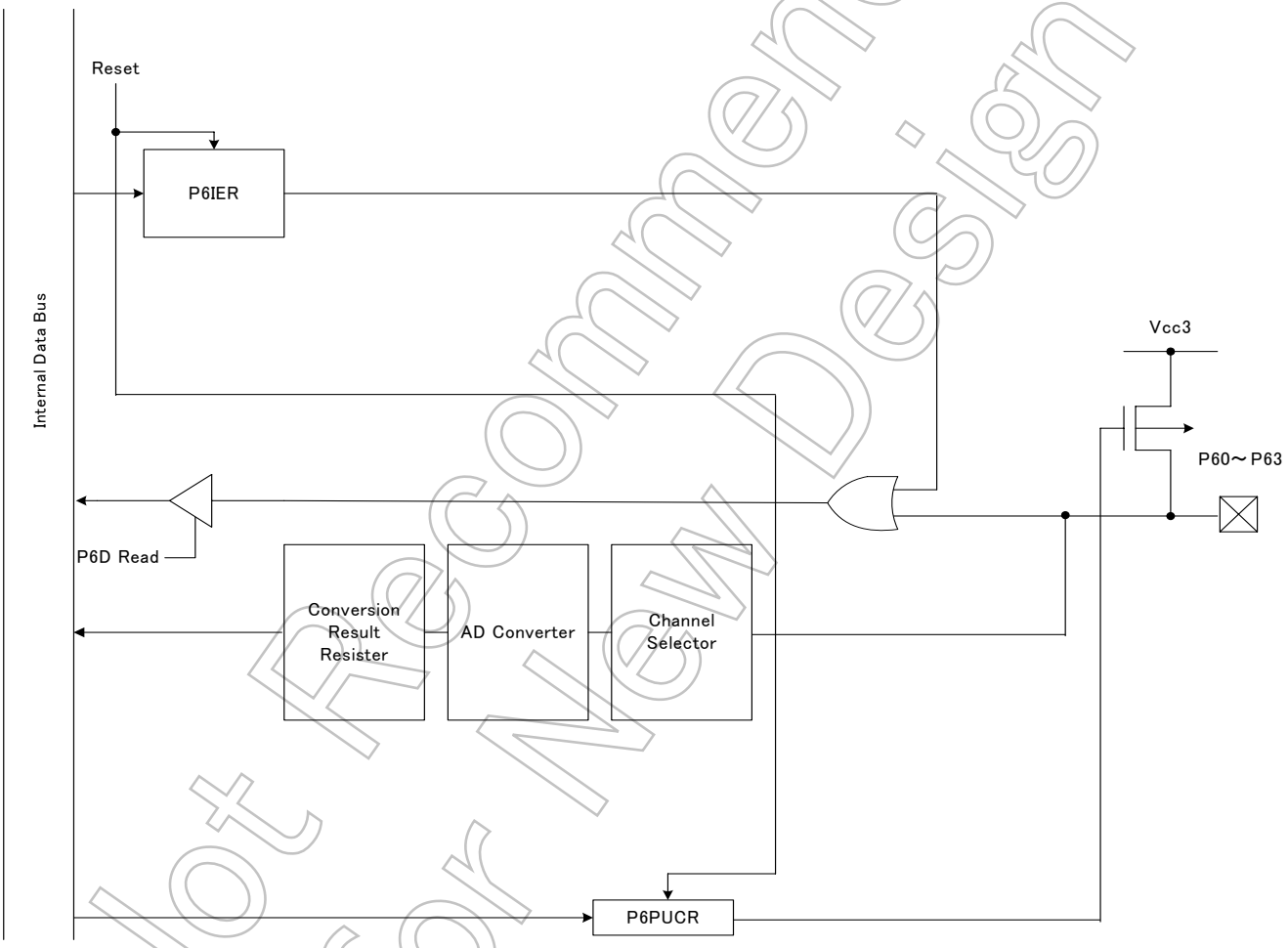
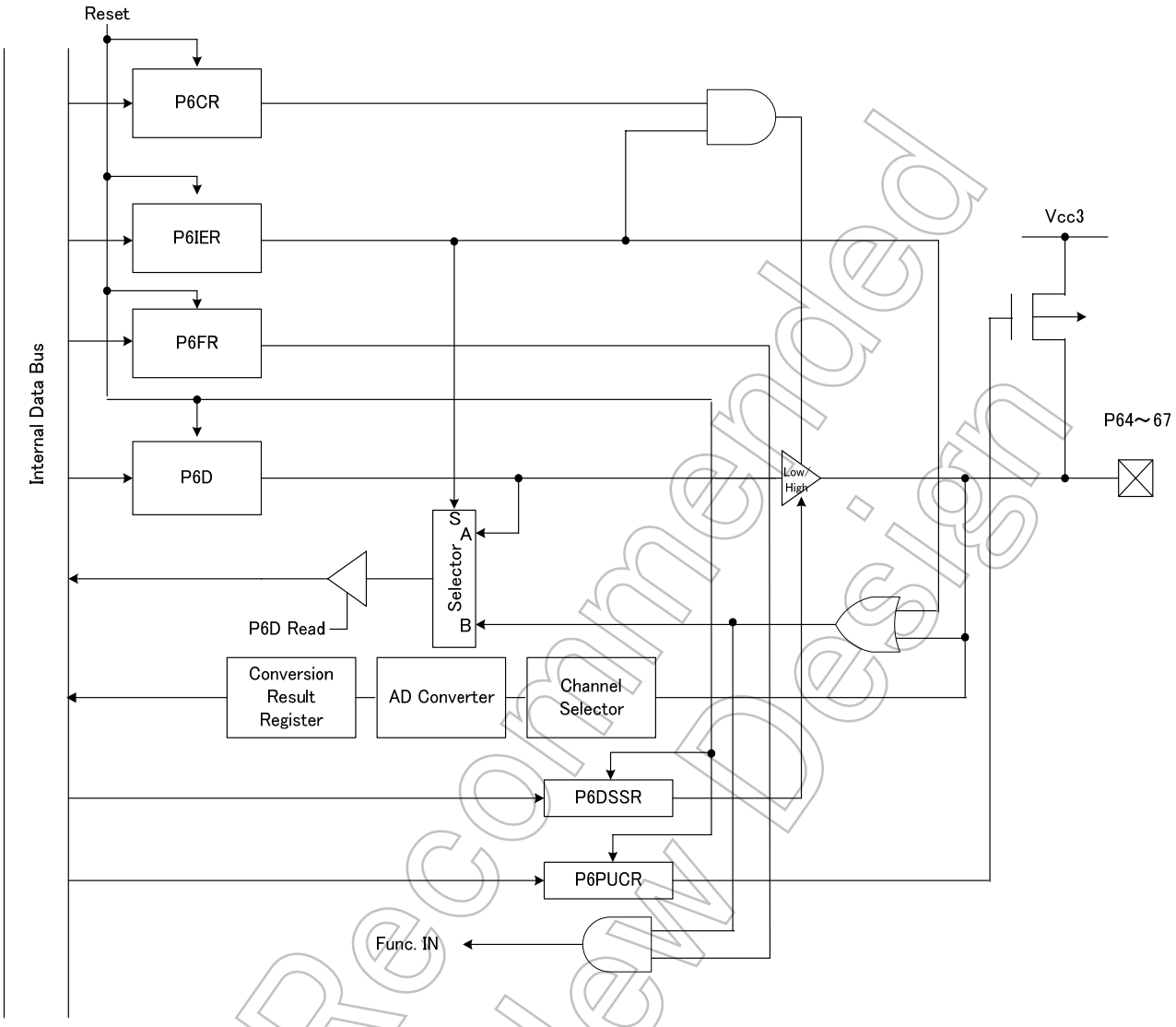


Figure 8.6.1 Port 6 (P60 to P63)

Not Recommended for New Design



Note: **The selectors in the figure output input A when S=1 and input B when S=0.**

Figure 8.6.2 Port 6 (P64 to P67)

Not Recommended for New Design

Port 6 Register

		7	6	5	4	3	2	1	0
P6D	Bit Symbol	P6D7	P6D6	P6D5	P6D4	P6D3	P6D2	P6D1	P6D0
(0xFFFF_C180)	Read/Write	R/W				R			
	Reset Value	0	0	0	0	0	0	0	0
	Function	Port 6 output data (Output latch)				Port 6 input data			

Note: When P6IER=0, the port state can be read from this register.

Port 6 Control Register

		7	6	5	4	3	2	1	0
P6CR	Bit Symbol	P6CR7	P6CR6	P6CR5	P6CR4	—	—	—	—
(0xFFFF_C184)	Read/Write	R/W							
	Reset Value	0	0	0	0	0	0	0	0
	Function	0: Output disabled 1: Output enabled							

Port 6 Input Enable Register

		7	6	5	4	3	2	1	0
P6IER	Bit Symbol	P6IER7	P6IER6	P6IER5	P6IER4	P6IER3	P6IER2	P6IER1	P6IER0
(0xFFFF_C188)	Read/Write	R/W							
	Reset Value	1	1	1	1	1	1	1	1
	Function	0: Input enabled 1: Input disabled							

Port 6 Drive Strength Register

		7	6	5	4	3	2	1	0
P6DSSR	Bit Symbol	P6DSSR7	P6DSSR6	P6DSSR5	P6DSSR4	—	—	—	—
(0xFFFF_C18C)	Read/Write	R/W							
	Reset Value	0	0	0	0	0	0	0	0
	Function	0: Low drive capability 1: High drive capability							

Note: The current flowing through ports should not exceed the maximum ratings for each port pin and for all the port pins.

Port 6 Pull-Up Control Register

		7	6	5	4	3	2	1	0
P6PUCR	Bit Symbol	P6PUCR7	P6PUCR6	P6PUCR5	P6PUCR4	P6PUCR3	P6PUCR2	P6PUCR1	P6PUCR0
(0xFFFF_C194)	Read/Write	R/W							
	Reset Value	0	0	0	0	0	0	0	0
	Function	0: Pull-up disabled 1: Pull-up enabled							

Not Recommended for New Design

Port 6 Function Register

		7	6	5	4	3	2	1	0
P6FR	Bit Symbol	P6FR7	P6FR6	P6FR5	P6FR4	—	—	—	—
(0xFFFF_C198)	Read/Write	R/W							
	Reset Value	0	0	0	0	0	0	0	0
	Function	0:Port/AD input 1:ADTRG1 /INT6	0:Port/AD input 1:INT5	0:Port/AD input 1:INT4	0:Port/AD input 1:INT3				

Note: When the P6FR is set to 1 (port or AD input) with P6CR=1 (output enabled), the output values of this register become undefined.

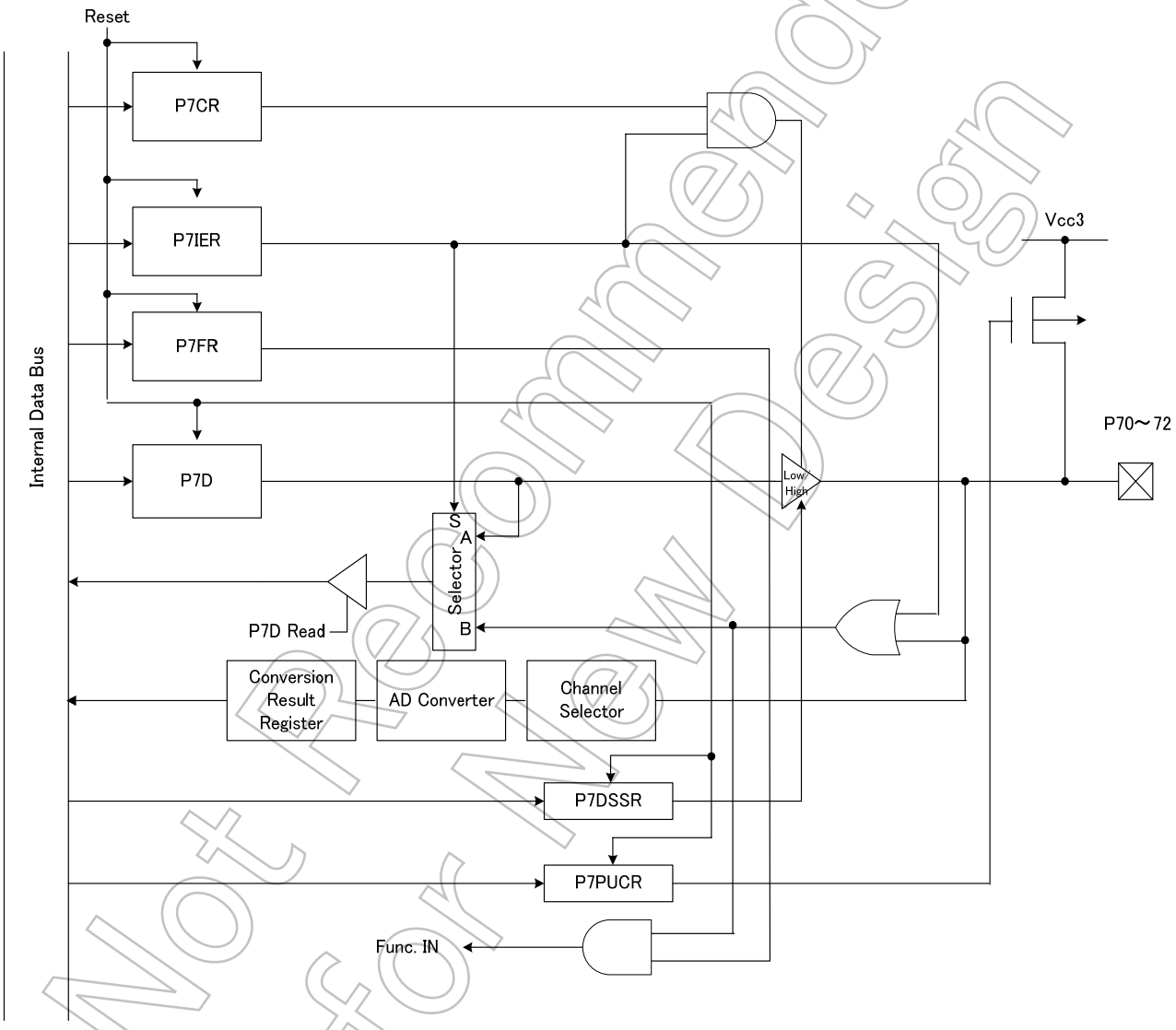
Not Recommended for New Design

8.7 Port 7 (P70 to P72)

Three Port 7 pins can be individually programmed to function as discrete general-purpose I/O pins shared with the analog input pins of the AD converter (ADC).

Note 1: As Port 7 uses AVCC1 as its I/O power source, it must be connected to the 3.3 V source even if ADC1 is not used.

Note 2: When Port 7 is not used as analog input pins, the AD conversion accuracy of ADC1 may deteriorate by a few LSBs. When Port 7 is used as an output port, this may result in a noticeable deterioration in AD conversion accuracy which may exceed the worst conditions presented in the AD conversion characteristics later in this manual. Be sure to check that this poses no problem on your system.



Note: The selectors in the figure output input A when S=1 and input B when S=0.

Figure 8.7.1 Port 7 (P70 to P72)

Not Recommended for New Design

Port 7 Register

P7D (0xFFFF_C1C0)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	P7D2	P7D1	P7D0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function						Port 7 output data (Output latch)		

Note: When P7IER=0, the port state can be read from this register.

Port 7 Control Register

P7CR (0xFFFF_C1C4)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	P7CR2	P7CR1	P7CR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function						0: Output disabled 1: Output enabled		

Port 7 Input Enable Register

P7IER (0xFFFF_C1C8)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	P7IER2	P7IER1	P7IER0
Read/Write	R/W							
Reset Value	0	0	0	0	0	1	1	1
Function						0: Input enabled 1: Input disabled		

Port 7 Drive Strength Register

P7DSSR (0xFFFF_C1CC)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	P7DSSR2	P7DSSR1	P7DSSR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function						0: Low drive capability 1: High drive capability		

Note: The current flowing through ports should not exceed the maximum ratings for each port pin and for all the port pins.

Port 7 Pull-Up Control Register

P7PUCR (0xFFFF_C1D4)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	P7PUCR2	P7PUCR1	P7PUCR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function						0: Pull-up disabled 1: Pull-up enabled		

Not Recommended for New Design

Port 7 Function Register

		7	6	5	4	3	2	1	0
P7FR1	Bit Symbol	—	—	—	—	—	P7FR12	P7FR11	P7FR10
(0xFFFF_C1D8)	Read/Write	R/W							
	Reset Value	0	0	0	0	0	0	0	0
	Function						0:Port/AD input 1:INT9	0:Port/AD input 1:INT8	0:Port/AD input 1:INT7

Note: When the P7FR is set to 1 (port or AD input) with P7CR=1 (output enabled), the output values of this register become undefined.

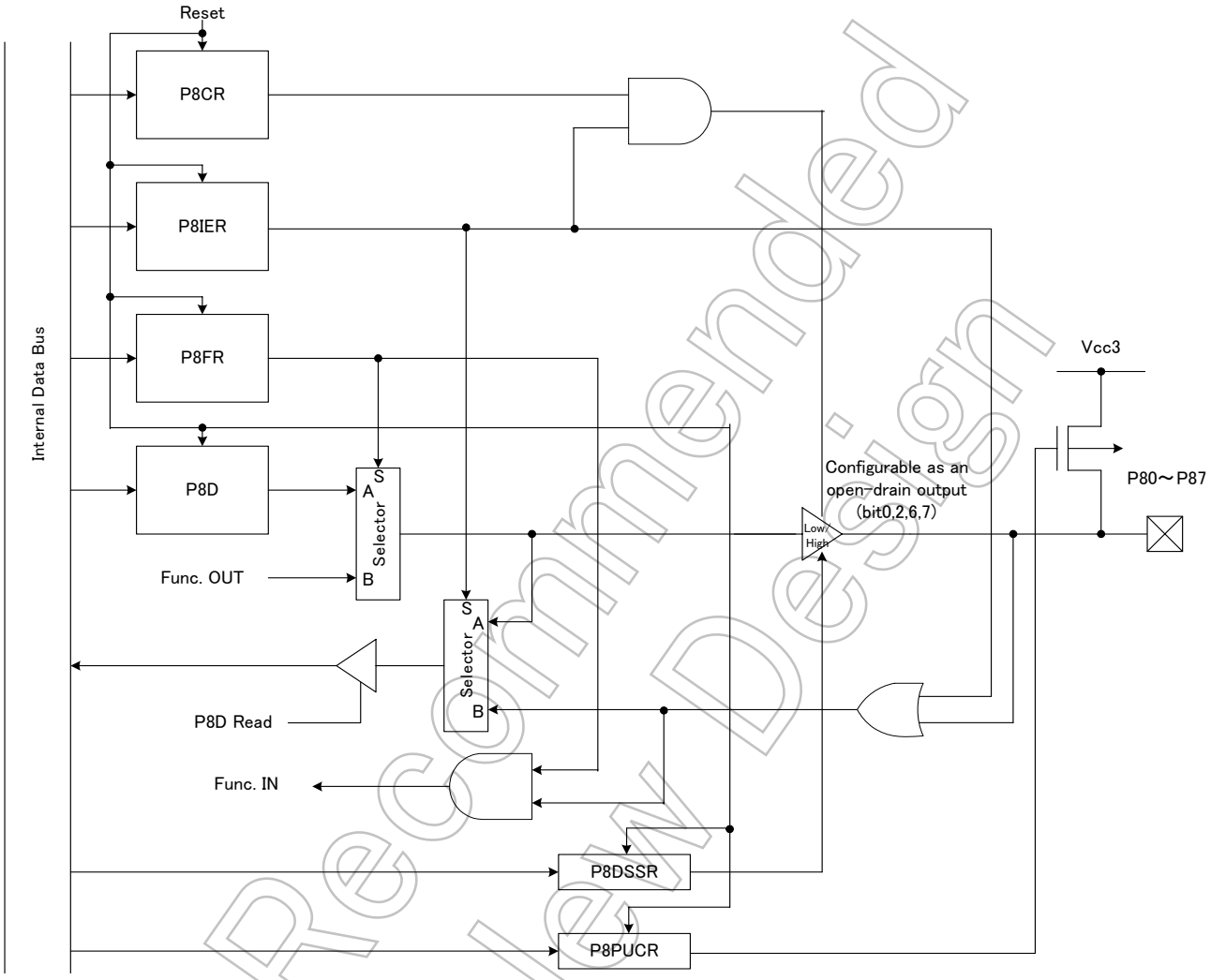
Port 7 Function Register

		7	6	5	4	3	2	1	0
P7FR2	Bit Symbol	—	—	—	—	—	P7FR22	P7FR21	P7FR20
(0xFFFF_C1DC)	Read/Write	R/W							
	Reset Value	0	0	0	0	0	0	0	0
	Function						0:Port/AD input 1:TB3IN	0:Port/AD input 1:TB2IN	0:Port/AD input 1:TB1IN

Not Recommended for New Design

8.8 Port 8 (P80 to P87)

Eight Port 8 pins can be individually programmed to function as discrete general-purpose I/O pins.



Note: The selectors in the figure output input A when S=1 and input B when S=0.

Figure 8.8.1 Port 8 (P80 to P87)

Not Recommended for New Design

Port 8 Register

P8D (0xFFFF_C200)

	7	6	5	4	3	2	1	0
Bit Symbol	P8D7	P8D6	P8D5	P8D4	P8D3	P8D2	P8D1	P8D0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	Port 8 output data (Output latch)							

Note: When P8IER=0, the port state can be read from this register.

Port 8 Control Register

P8CR (0xFFFF_C204)

	7	6	5	4	3	2	1	0
Bit Symbol	P8CR7	P8CR6	P8CR5	P8CR4	P8CR3	P8CR2	P8CR1	P8CR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Output disabled 1: Output enabled							

Port 8 Input Enable Register

P8IER (0xFFFF_C208)

	7	6	5	4	3	2	1	0
Bit Symbol	P8IER7	P8IER6	P8IER5	P8IER4	P8IER3	P8IER2	P8IER1	P8IER0
Read/Write	R/W							
Reset Value	1	1	1	1	1	1	1	1
Function	0: Input enabled 1: Input disabled							

Port 8 Drive Strength Register

P8DSSR (0xFFFF_C20C)

	7	6	5	4	3	2	1	0
Bit Symbol	P8DSSR7	P8DSSR6	P8DSSR5	P8DSSR4	P8DSSR3	P8DSSR2	P8DSSR1	P8DSSR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Low drive capability 1: High drive capability							

Note: The current flowing through ports should not exceed the maximum ratings for each port pin and for all the port pins.

Port 8 Open-Drain Control Register

P8ODCR (0xFFFF_C210)

	7	6	5	4	3	2	1	0
Bit Symbol	P8ODCR7	P8ODCR6	—	—	—	P8ODCR2	—	P8ODCR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Open-drain disabled 1: Open-drain enabled							

Port 8 Pull-Up Control Register

P8PUCR (0xFFFF_C214)

	7	6	5	4	3	2	1	0
Bit Symbol	P8PUCR7	P8PUCR6	P8PUCR5	P8PUCR4	P8PUCR3	P8PUCR2	P8PUCR1	P8PUCR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Pull-up disabled 1: Pull-up enabled							

Note: In Level-1 DSU (EJTAG) mode, P86 and P87 function as DSU control pins and the P8D, P8CR, P8IER, P8DSSR, P8ODCR and P8PUCR are invalid.

Not Recommended for New Design

Port 8 Function Register 1

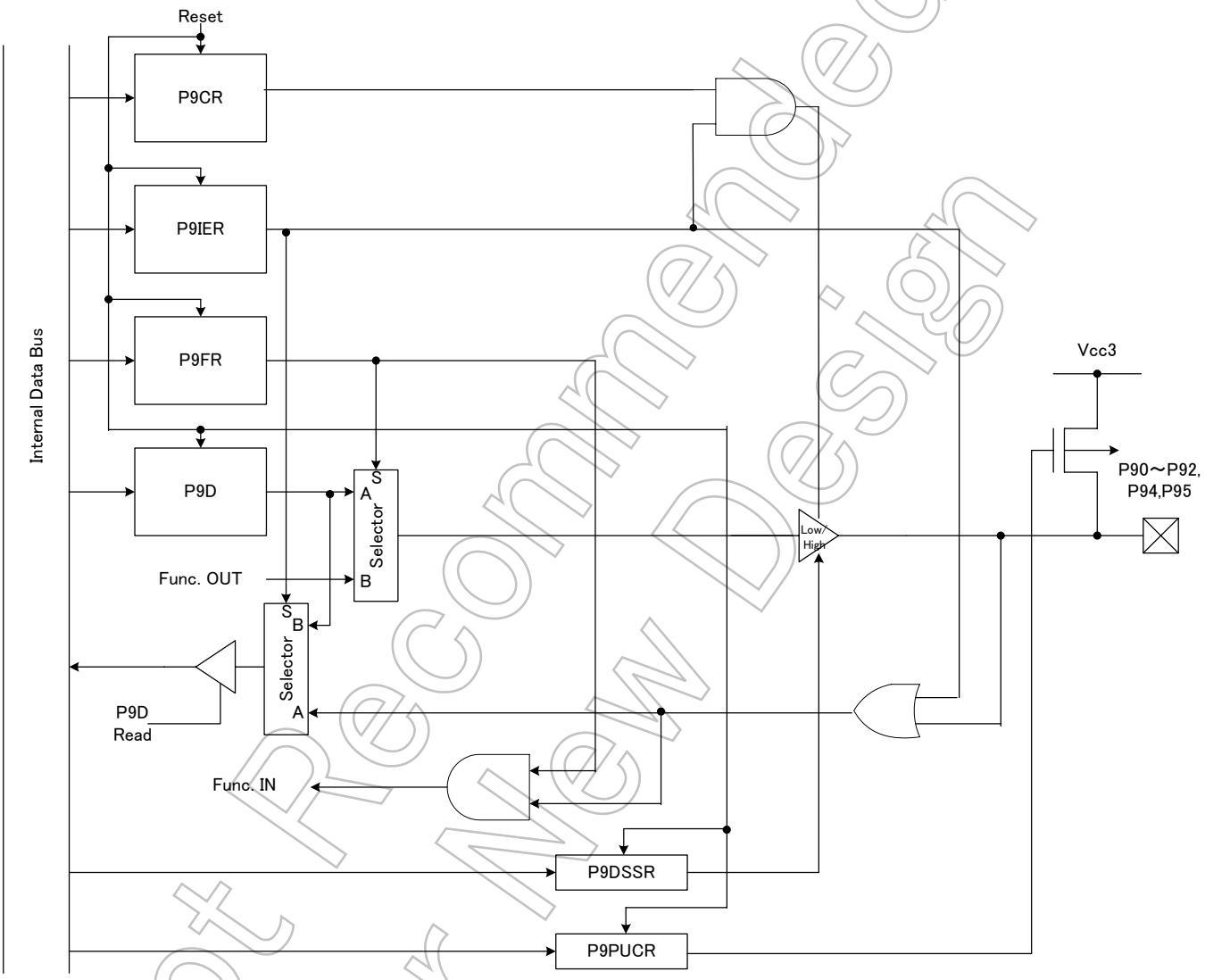
		7	6	5	4	3	2	1	0
P8FR	Bit Symbol	P8FR17	P8FR16	P8FR15	P8FR14	P8FR13	P8FR12	P8FR11	P8FR10
(0xFFFF_C218)	Read/Write	R/W							
	Reset Value	0	0	0	0	0	0	0	0
	Function	0:Port 1:SCLK2 /CTS2	0:Port 1:TX2	0:Port 1:RX2	0:Port 1:TB1OUT /INT0	0:Port 1:RX1	0:Port 1:TX1	0:Port 1:RX0	0:Port 1:TX0

Note: When the P8FR is set to 1 (port input) with P8CR=1 (output enabled), the output values of P81, P83, P84, P85 and P87 become undefined.

Not Recommended for New Design

8.9 Port 9 (P90 to P95)

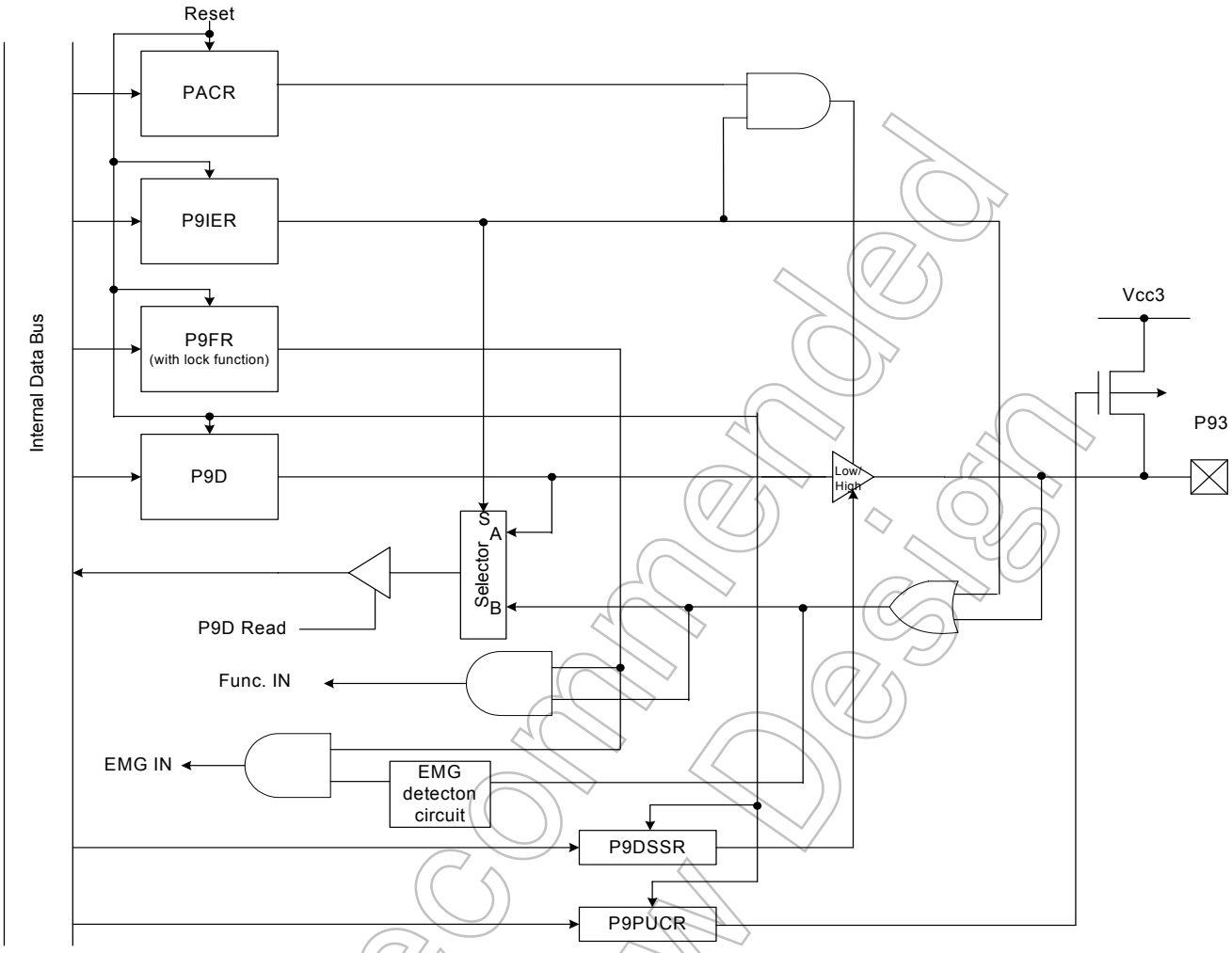
Six Port 9 pins can be individually programmed to function as discrete general-purpose I/O pins. P93 is shared with the emergency stop signal input pin (EMG pin) of TMRB0, and set as a general-purpose port after reset. P93 can be used as the EMG pin by setting the P9FR2.P9FR23 bit which is protected with the lock function. Likewise, P95 is shared with the NMI pin, and set as a general-purpose port after reset. P95 can be used as the NMI pin by setting the P9FR1.P9FR15 bit which is protected with the lock function.



Note: The selectors in the figure output input A when S=1 and input B when S=0.

Figure 8.9.1 Port 9 (P90 to P92, P94, P95)

Not Recommended for New Design



Note: The selectors in the figure output input A when S=1 and input B when S=0.

Figure 8.9.2 Port 9 (P93)

Not Recommended for New Design

Port 9 Register

P9D (0xFFFF_C240)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	P9D5	P9D4	P9D3	P9D2	P9D1	P9D0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	Port 9 output data (Output latch)							

Note: When P9IER=0, the port state can be read from this register.

Port 9 Control Register

P9CR (0xFFFF_C244)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	P9CR5	P9CR4	P9CR3	P9CR2	P9CR1	P9CR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Output disabled 1: Output enabled							

Port 9 Input Enable Register

P9IER (0xFFFF_C248)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	P9IER5	P9IER4	P9IER3	P9IER2	P9IER1	P9IER0
Read/Write	R/W							
Reset Value	0	0	1	1	1	1	1	1
Function	0: Input enabled 1: Input disabled							

Port 9 Drive Strength Register

P9DSSR (0xFFFF_C24C)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	P9DSSR5	P9DSSR4	P9DSSR3	P9DSSR2	P9DSSR1	P9DSSR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0:Low drive capability 1: High drive capability							

Note: The current flowing through ports should not exceed the maximum ratings for each port pin and for all the port pins.

Port 9 Pull-Up Control Register

P9PUCR (0xFFFF_C254)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	P9PUCR5	P9PUCR4	P9PUCR3	P9PUCR2	P9PUCR1	P9PUCR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Pull-up disabled 1: Pull-up enabled							

Note: P94 is designated as the BOOT pin. To start up the device in BOOT mode (see the chapter on Flash memory), P94 should be set to 0 during a reset sequence. To start up the device in NORMAL mode, P94 should be set to 1 during a reset sequence.

Not Recommended for New Design

Port 9 Function Register 1

P9FR1 (0xFFFF_C258)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	P9FR15	P9FR14	P9FR13	P9FR12	P9FR11	P9FR10
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function			0:Port 1:NMI (with lock function)	0:Port 1:TB0OUT	0:Port 1:TB0IN	0:Port 1:ENCZ	0:Pprt 1:ENCB	0:Port 1:ENCA

P9FR15 is a register bit with the lock function. Writing a value to this bit requires writing 0x55 and then 0xAA to the P9ECLR register. Once these values are written, the P9ECLR remains in effect until a write to a Port 9 register with the lock function is completed.

Port 9 Function Register 2

P9FR2 (0xFFFF_C25C)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	P9FR23	P9FR22	P9FR21	P9FR20
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function					0:Port 1:EMG input (with lock function)	0:Port 1:SCLK3 /CTS3	0:Port 1:TX3	0:Port 1:RX3

P9FR23 is a register bit with the lock function. Writing a value to this register requires writing 0x55 and then 0xAA to the P9ECLR register. Once these values are written, the P9ECLR remains in effect until a write to a Port 9 register with the lock function is completed.
Setting the P9FR23 bit to 1 prohibits writes to other registers related to P93.

Port 9 EMG Control Register

P9ECR (0xFFFF_C260)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	ERM		EMGF	EMGE	—	—
Read/Write	R/W				R	R/W		
Reset Value	0	0	0	0	0	0	0	0
Function			EMG sensitivity 00: Low level 01: High level 10: Falling edge 11: Rising edge (with lock function)		EMG condition flag 0: Normal condition 1: EMG condition	EMG condition clear 1: Clear EMG condition This bit is read as 0. (with lock function)		

Not Recommended for New Design

Port 9 EMG Clear Register

		7	6	5	4	3	2	1	0
P9ECLR	Bit Symbol	—							
(0xFFFF_C264)	Read/Write	W							
	Reset Value	—							
	Function	Writing 0x55 and then 0xAA to this register allows a single write to a register with the lock function.							

Note 1: Setting both P9FR13 and P9FR23 to 1 results in undefined behavior.

Note 2: When the P9FR is set to 1 (port input) with P9CR=1 (output enabled), the output values of P90, P91, P92, P93 and P95 become undefined.

8.9.1 Notes on Using the Emergency Stop Signal Input Pin (P93)

8.9.1.1 Port Operation in the EMG Condition

When P93 set as the EMG pin is asserted, output is disabled on P94 and an INTTBE0 interrupt is generated in Port 9, as shown in Table 8.9.1. As the EMG detection circuit operates independently of the 16-bit timer, the 16-bit timer continues to operate normally even in case of emergency.

Table 8.9.1 Port Operation in the EMG Condition

P93	P94	INTTBE0
Normal	PWM/PORT output	Not generated
EMG	Hi-z	Generated

Not Recommended for New Design

8.9.1.2 Register Settings for P93

When P93 is set as the EMG pin (P9FR2.P9FR23=1), other registers related to P93 (i.e., P9CR3, P9IER3, P9DSSR3, P9PUCR3, P9FR13) cannot be changed. Clearing the P9FR23 bit to 0 enables writes to these registers again.

Table 8.9.2 shows the register settings for P93 according to the selected function.

Table 8.9.2 Register Settings for P93

	General-purpose I/O port	TB0IN	EMG pin
P9CR.P9CR3	X	0	0 (Note)
P9IER.P9IER3	X	0	0 (Note)
P9DSSR.P9DSSR3	X	X	X (Note)
P9PUCR.P9PUCR3	X	X	0 (Note)
P9FR1.P9FR13	0	1	0
P9FR2.P9FR23	0	0	1

Note: Must be set before the P9FR2.P9FR23 bit is set.

General procedure for setting P93 as the EMG pin (falling edge sensitive)

```
P9ECLR=0x55→0xAA                ; release lock
P9ECR<ERM>=10                   ; falling edge sensitive
P9CR<P9CR3>=0                   ; disable output
P9IER<P9IER3>=0                 ; enable input
P9PUCR<PAPUCR6>=0               ; disable pull-up
P9ECLR=0x55→0xAA                ; release lock
P9ECR<EMGE>=1                   ; clear EMG condition
P9ECLR=0x55→ 0xAA               ; release lock
P9FR2<P9FR23>=1                 ; set P93 as EMG pin
IMR33<EIM33>=10
ICLR<IV>=0x084                  ; clear INTTBE0
IMR33<IL33[2:0]>= 111           ; set INTTBE0 interrupt level to 7 (or any level)
```

General procedure for clearing the EMG condition (edge sensitive)

(* When the EMG pin is set as edge sensitive, make sure that P93 is inactive before clearing the EMG condition.)

```
P9ECLR=0x55→0xAA                ; release lock
P9ECR<EMGE>=1                   ; clear EMG condition
```

Procedure for returning P93 to a general-purpose port

```
IMR33<IL33[2:0]>= 000           ; disable INTTBE0 interrupt
ICLR<IV>=0x084                  ; clear INTTBE0
P9ECLR=0x55→ 0xAA               ; release lock
P9FR23<P9FR23>=0                ; set P93 as a general-purpose port
```

8.9.1.3 Sensitivity-Related Considerations

(1) Level sensitive

When the EMG pin is set as level sensitive, the EMG condition is held (P9ECR.EMGF=1) only while the EMG pin is active. Therefore, there is no need to clear the EMG condition by setting the P9ECR.EMGE bit to 1.

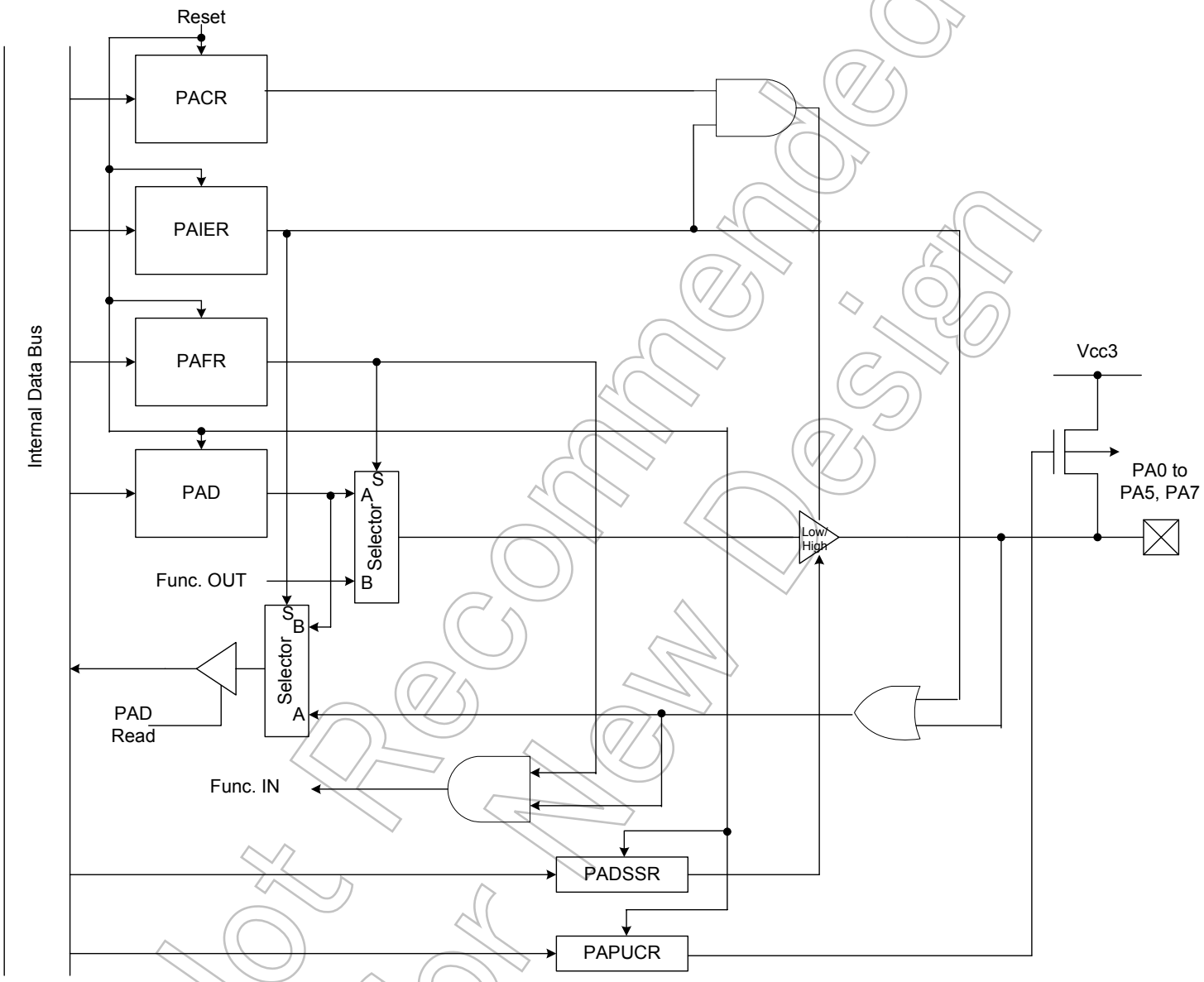
(2) Edge sensitive

When the EMG pin is set as edge sensitive, be sure to check that the EMG pin is inactive before making an EMG condition setting.

Not Recommended for New Design

8.10 Port A (PA0 to PA7)

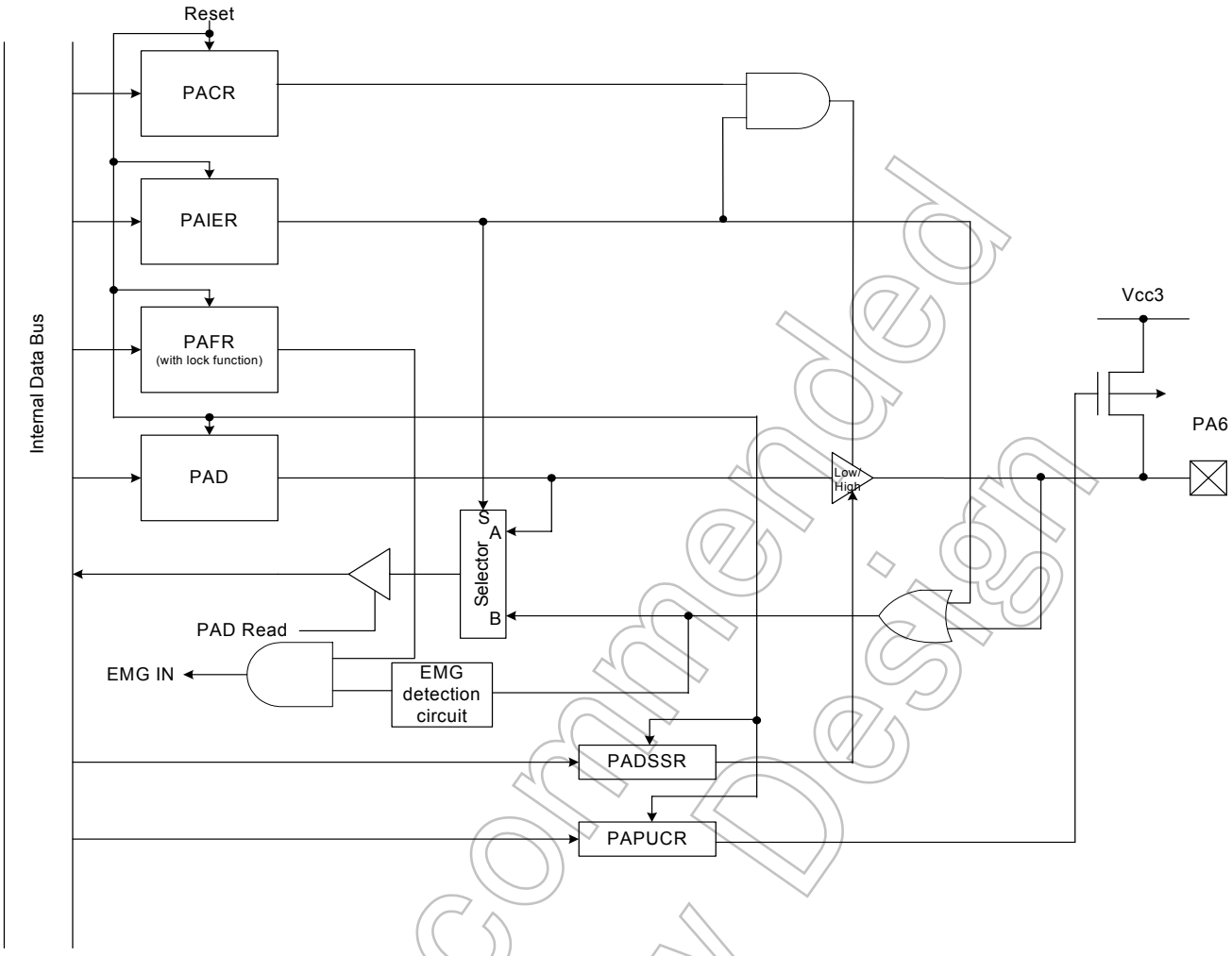
Eight Port A pins can be individually programmed to function as discrete general-purpose I/O pins. PA6 is shared with the emergency stop signal input pin (EMG0 pin) of PMD0, and set as a general-purpose port after reset. PA6 can be used as the EMG0 pin by setting the PAFR.PAFR6 bit which is protected with the lock function.



Note: The selectors in the figure output input A when S=1 and input B when S=0.

Figure 8.10.1 Port A (PA0 to PA5, PA7)

Not Recommended for New Design



Note: The selectors in the figure output input A when S=1 and input B when S=0.

Figure 8.10.2 Port A (PA6)

Not Recommended for New Design

Port A Register

PAD (0xFFFF_C280)

	7	6	5	4	3	2	1	0
Bit Symbol	PAD7	PAD6	PAD5	PAD4	PAD3	PAD2	PAD1	PAD0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	Port A output data (Output latch)							

Note: When PAIER=0, the port state can be read from this register.

Port A Control Register

PACR (0xFFFF_C284)

	7	6	5	4	3	2	1	0
Bit Symbol	PACR7	PACR6	PACR5	PACR4	PACR3	PACR2	PACR1	PACR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Output disabled 1: Output enabled							

Port A Input Enable Register

PAIER (0xFFFF_C288)

	7	6	5	4	3	2	1	0
Bit Symbol	PAIER7	PAIER6	PAIER5	PAIER4	PAIER3	PAIER2	PAIER1	PAIER0
Read/Write	R/W							
Reset Value	1	1	1	1	1	1	1	1
Function	0: Input enabled 1: Input disabled							

Port A Drive Strength Register

PADSSR (0xFFFF_C28C)

	7	6	5	4	3	2	1	0
Bit Symbol	PADSSR7	PADSSR6	PADSSR5	PADSSR4	PADSSR3	PADSSR2	PADSSR1	PADSSR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Low drive capability 1: High drive capability							

Note: The current flowing through ports should not exceed the maximum ratings for each port pin and for all the port pins.

Port A Pull-Up Control Register

PAPUCR (0xFFFF_C294)

	7	6	5	4	3	2	1	0
Bit Symbol	PAPUCR7	PAPUCR6	PAPUCR5	PAPUCR4	PAPUCR3	PAPUCR2	PAPUCR1	PAPUCR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Pull-up disabled 1: Pull-up enabled							

Not Recommended for New Design

Port A Function Register

PAFR (0xFFFF_C298)

	7	6	5	4	3	2	1	0
Bit Symbol	PAFR7	PAFR6	PAFR5	PAFR4	PAFR3	PAFR2	PAFR1	PAFR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0:Port 1:TB2OUT/ INT1	0:Port 1:EMG0 (with lock function)	0:Port 1:Z0	0:Port 1:W0	0:Port 1:Y0	0:Port 1:V0	0:Port 1:X0	0:Port 1:U0

When the PAFR6 bit is set to 1, PA6 is used as the EMG0 pin. The PAFR6 bit has the lock function, and writing a value to this bit requires writing 0x55 and then 0xAA to the PAECLR register. Once these values are written, the PAECLR register remains in effect until a write to a Port A register with the lock function is completed.
Setting the PAFR6 bit to 1 prohibits writes to other registers related to PA6.

Port A EMG Control Register

PAECR (0xFFFF_C29C)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	ERMA		EMGFA	EMGEA	—	—
Read/Write	R/W				R	R/W		
Reset Value	0	0	0	0	0	0	0	0
Function			EMG sensitivity 00: Low level 01: High level 10: Falling edge 11: Rising edge (with lock function)		EMG condition flag 0: Normal condition 1: EMG condition	EMG condition clear 1: Clear EMG condition This bit is read as 0. (with lock function))		

Port A EMG Clear Register

PAECLR (0xFFFF_C2A0)

	7	6	5	4	3	2	1	0
Bit Symbol	—							
Read/Write	W							
Reset Value	—							
Function	Writing 0x55 and then 0xAA to this register allows a single write to a Port A register with the lock function.							

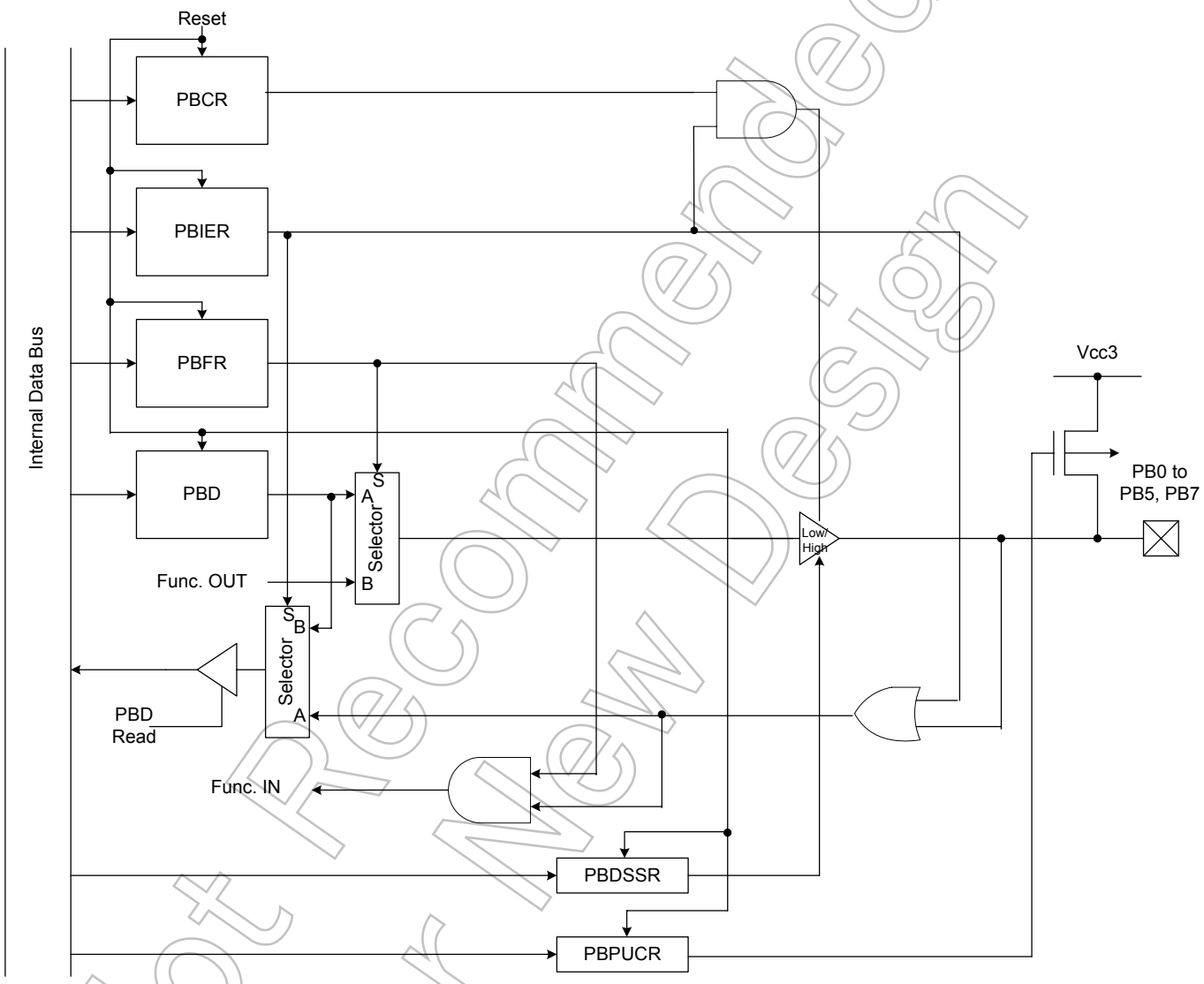
Note: When the PAFR is set to 1 (port input) with PACR=1 (output enabled), the output values of PA6 and PA7 become undefined.

For details, see 8.12 Notes on Using the Emergency Stop Signal Input Pins (PA6, PB6).

Not Recommended for New Design

8.11 Port B (PB0 to PB7)

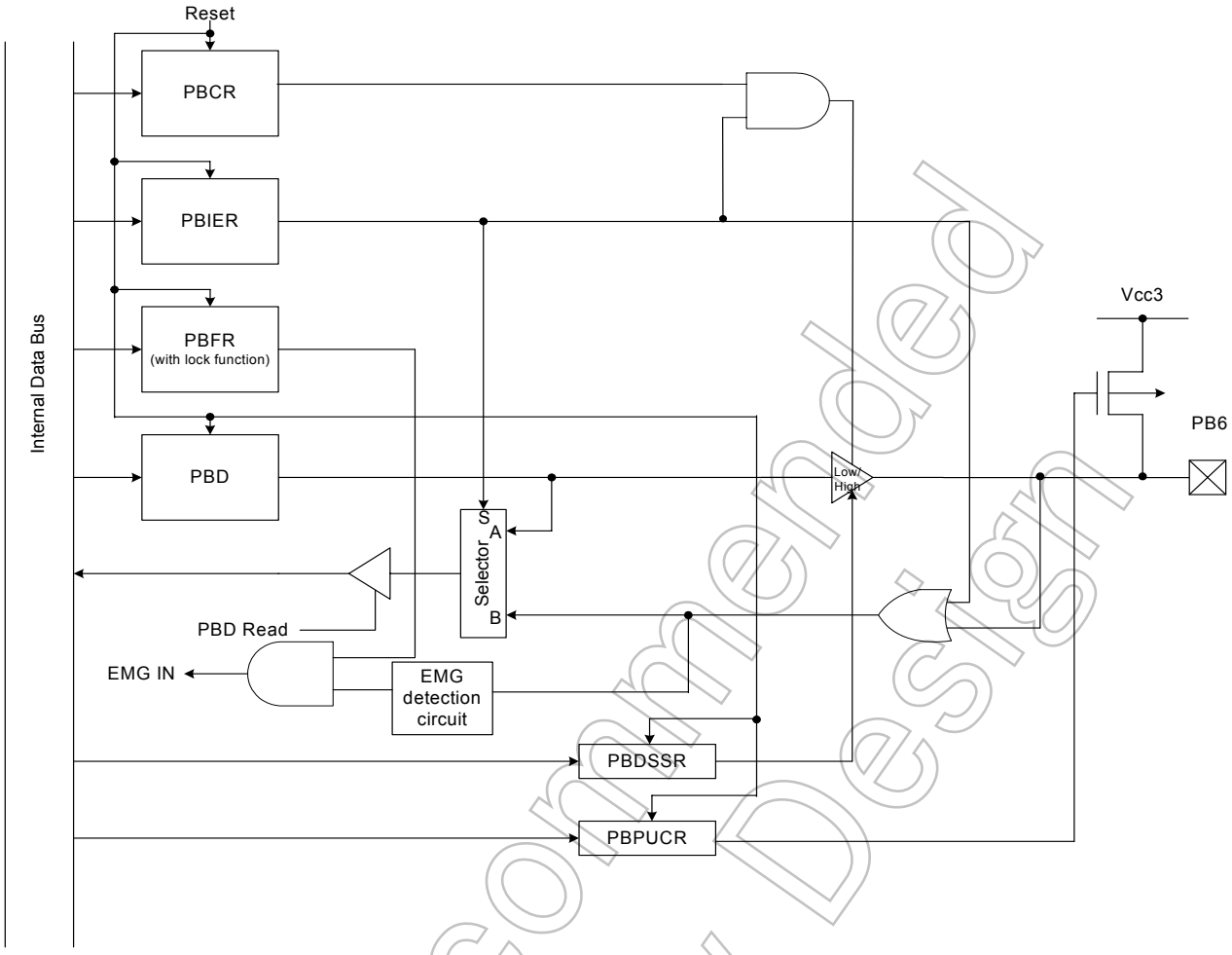
Eight Port B pins can be individually programmed to function as discrete general-purpose I/O pins. PB6 is shared with the emergency stop signal input pin (EMG1 pin) of PMD1, and set as a general-purpose port after reset. PB6 can be used as the EMG1 pin by setting the PBFR.PBFR6 bit which is protected with the lock function.



Not Recommended for New Design

Note: The selectors in the figure output input A when S=1 and input B when S=0.

Figure 8.11.1 Port B (PB0 to PB5, PB7)



Note: The selectors in the figure output input A when S=1 and input B when S=0.

Figure 8.11.2 Port B (P86)

Not Recommended for New Design

Port B Register

PBD (0xFFFF_C2C0)

	7	6	5	4	3	2	1	0
Bit Symbol	PBD7	PBD6	PBD5	PBD4	PBD3	PBD2	PBD1	PBD0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	Port B output data (Output latch)							

Note: When PBIER=0, the port state can be read from this register.

Port B Control Register

PBCR (0xFFFF_C2C4)

	7	6	5	4	3	2	1	0
Bit Symbol	PBCR7	PBCR6	PBCR5	PBCR4	PBCR3	PBCR2	PBCR1	PBCR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Output disabled 1: Output enabled							

Port B Input Enable Register

PBIER (0xFFFF_C2C8)

	7	6	5	4	3	2	1	0
Bit Symbol	PBIER7	PBIER6	PBIER5	PBIER4	PBIER3	PBIER2	PBIER1	PBIER0
Read/Write	R/W							
Reset Value	1	1	1	1	1	1	1	1
Function	0: Input enabled 1: Input disabled							

Port B Drive Strength Register

PBDSSR (0xFFFF_C2CC)

	7	6	5	4	3	2	1	0
Bit Symbol	PBDSSR7	PBDSSR6	PBDSSR5	PBDSSR4	PBDSSR3	PBDSSR2	PBDSSR1	PBDSSR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Low drive capability 1: High drive capability							

Note: The current flowing through ports should not exceed the maximum ratings for each port pin and for all the port pins.

Port B Pull-Up Control Register

PBPUCR (0xFFFF_C2D4)

	7	6	5	4	3	2	1	0
Bit Symbol	PBPUCR7	PBPUCR6	PBPUCR5	PBPUCR4	PBPUCR3	PBPUCR2	PBPUCR1	PBPUCR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0: Pull-up disabled 1: Pull-up enabled							

Not Recommended for New Design

Port B Function Register

PBFR (0xFFFF_C2D8)	7	6	5	4	3	2	1	0
Bit Symbol	PBFR7	PBFR6	PBFR5	PBFR4	PBFR3	PBFR2	PBFR1	PBFR0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	0:Port 1:TB3OUT/ INT2	0:Port 1:EMG1 (with lock function)	0:Port 1:Z1	0:Port 1:W1	0:Port 1:Y1	0:Port 1:V1	0:Port 1:X1	0:Port 1:U1

When the PBFR6 bit is set to 1, PB6 is used as the EMG1 pin. The PBFR6 bit has the lock function, and writing to this bit requires writing 0x55 and then 0xAA to the PBECLR register. Once these values are written, the PBECLR register remains in effect until a write to a Port B register with the lock function is completed.
Setting the PBFR6 bit to 1 prohibits writes to other registers related to PB6.

Port B EMG Control Register

PBECR (0xFFFF_C2DC)	7	6	5	4	3	2	1	0
Bit Symbol	—	—	ERMB		EMGFB	EMGEB	—	—
Read/Write	R/W				R	R/W		
Reset Value	0	0	0	0	0	0	0	0
Function			EMG sensitivity 00: Low level 01: High level 10: Falling edge 11: Rising edge (with lock function)		EMG condition flag 0: Normal condition 1: EMG condition	EMG condition clear 1: Clear EMG condition This bit is read as 0. (with lock function)		

Port B EMG Clear Register

PBECLR (0xFFFF_C2E0)	7	6	5	4	3	2	1	0
Bit Symbol	—							
Read/Write	W							
Reset Value	—							
Function	Writing 0x55 and then 0xAA to this register allows a single write to a Port B register with the lock function.							

Note: When the PBFR is set to 1 (port input) with PBCR=1 (output enabled), the output values of PB6 and PB7 become undefined.

For details, see 8.12 Notes on Using the Emergency Stop Signal Input Pins (PA6, PB6).

Not Recommended for New Design

8.12 Notes on Using the Emergency Stop Signal Input Pins (PA6, PB6)

8.12.1 Block Diagram of the EMG Detection Circuit

Note: The following descriptions for PA[6:0] (PMD0) also apply to PB[6:0] (PMD1), unless otherwise noted.

When PA6 is set as the emergency stop signal input pin (EMG0 pin), an EMG input activates the EMG detection circuit of PMD0 and forcefully disables output on PA[5:0] even if these pins are not set for PMD0 output. The EMG detection circuit of PMD0 is enabled by setting the EMGCR0.EMGEN bit to 1 in addition to setting PA6 as the EMG0 pin.
Figure 8.12.1 shows a block diagram of the EMG detection circuit.

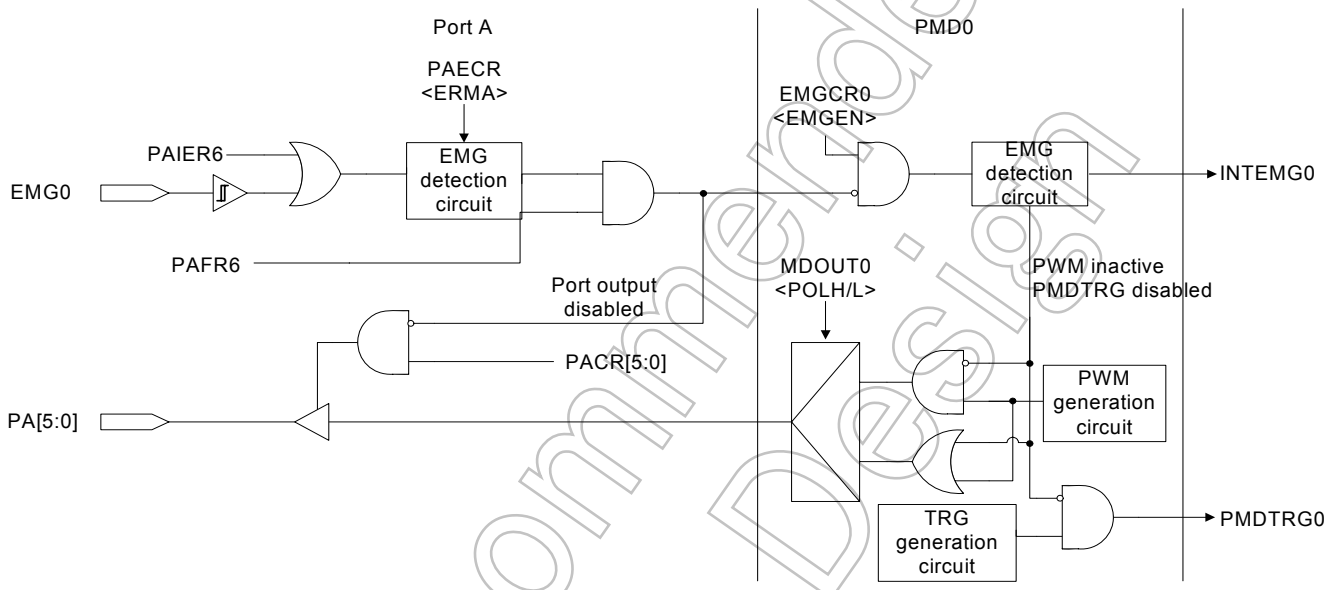


Figure 8.12.1 Block Diagram of EMG Detection Circuit

8.12.2 Operations in the EMG Condition

When Port A and PMD are put in the EMG condition, the following operations are performed.

- In Port A, output is disabled on PA[5:0].
- In PMD, PWM output is made inactive, ADC start trigger (PMDTRG) is disabled, and an INTEMG interrupt is generated.

Table 8.12.1 shows a summary of operations in the EMG condition for PMD and Port A which operate independently of each other.

Table 8.12.1 PMD and Port A Operations in the EMG Condition

PMD	Port A	PA[5:0]	PMDTRG	INTEMG
Normal	Normal	PWM/Port output	Trigger enabled	Interrupt not generated
EMG	Normal	Inactive	Trigger disabled	Interrupt generated (Note 1)
Normal	EMG	Hi-z	Trigger enabled	Interrupt not generated
EMG	EMG	Hi-z	Trigger disabled	Interrupt generated

Note: If PA6 is not set as the EMG0 pin, no EMG input will be accepted and thus PMD will not be put in the EMG condition. The combination of PMD=EMG and Port A=normal occurs only when the EMG condition is cleared in Port A when PMD=EMG and Port A=EMG.

Not Recommended for New Design

8.12.3 Register Settings for PA6

When PA6 is set as the EMG0 pin (PAFR.PAFR6=1), other registers related to PA6 (i.e., PACR6, PAIER6, PADSSR6, PAPUCR6) cannot be changed. Clearing the PAFR6 bit to 0 enables writes to these registers again. Table 8.12.2 shows the register settings for PA6 according to the selected function.

Table 8.12.2 Register Settings for PA6

	General-purpose I/O port	EMG pin
PACR.PACR6	X	0(Note)
PAIER.PAIER6	X	0(Note)
PADSSR.PADSSR6	X	X(Note)
PAPUCR.PAPUCR6	X	0(Note)
PAFR.PAFR6	0	1

Note: Must be set before the PAFR.PAFR6 bit is set.

General procedure for setting PA6 as the EMG0 pin (falling edge sensitive)

```
PAECLR=0x55→0xAA                    ; release lock
PAECR<ERMA>=10                      ; falling edge sensitive
PACR<PACR6>=0                       ; disable output
PAIER<PAIER6>=0                     ; enable input
PAPUCR<PAPUCR6>=0                   ; disable pull-up
PAECLR=0x55→0xAA                    ; release lock
PAECR<EMGEA>=1                      ; clear EMG condition (must be set separately from setting edge sensitivity)
PAECLR=0x55→0xAA                    ; release lock
PAFR<PAFR6>=1                       ; set PA6 as EMG0
IMR22<EIM22>=10
ICLR<IV>=0x058                      ; clear INTEMG0
IMR22<IL22[2:0]>=111                ; set  INTEMG0 interrupt level to 7 (or any level)
```

General procedure for clearing the EMG condition (edge sensitive)

```
PAECLR=0x55→0xAA                    ; release lock
PAECR<EMGE>=1                       ; clear EMG condition in Port A
MDOUT0<WOC,VOC,UOC>=000000          ; make PWM output inactive (through PMD register)
EMGCR0<EMGRS>=1                     ; clear EMG condition in PMD (through PMD register)
MDOUT0<WOC,VOC,UOC>=xxxxxx          ; set PWM output as desired (through PMD register)
```

Note 1: When EMG0 is set as edge sensitive, be sure to check that EMG0 is inactive before clearing the EMG condition.

Note 2: If the EMG condition is cleared only in PMD and not in Port A, PMD is put in the EMG condition again and an INTEMG0 interrupt is generated.

General procedure for clearing the EMG condition (level sensitive)

```
MDOUT0<WOC,VOC,UOC>=000000          ; make PWM output inactive (through PMD register)
EMGCR0<EMGRS>=1                     ; clear EMG condition in PMD (through PMD register)
MDOUT0<WOC,VOC,UOC>=xxxxxx          ; set PWM output as desired (through PMD register)
```

Note 1: When EMG0 is set as level sensitive, Port A is put in the EMG condition only while EMG0 is active. Thus, the EMG condition should be cleared only in PMD.

Note 2: If the EMG condition is cleared only in PMD and not in Port A, PMD is put in the EMG condition again and an INTEMG0 interrupt is generated.

Procedure for returning PA6 to a general-purpose port

```
IMR22<IL22[2:0]>= 000               ; disable INTEMG0 interrupt
ICLR<IV>=0x058                      ; clear INTEMG0
PAECLR=0x55→0xAA                    ; release lock
PAFR<PAFR6>=0                       ; set PA6 as a general-purpose port
```

Not Recommended for New Design

When EMG0 is set as level sensitive, Port A is put in the EMG condition only while EMG0 is active (PAECR.EMGFA=1). Thus, there is no need to clear the EMG condition in Port A by setting PAECR.EMGEA to 1. However, when the EMG detection circuit is enabled in PMD, the EMG condition must be cleared in PMD by setting EMGCR1.EMGRS to 1 after making sure that EMG0 is inactive.

When EMG0 is set as edge sensitive, make sure that EMG0 is inactive before making an EMG condition setting.

8.12.4 Difference between P93 (TB0IN) and PA6 (EMG0)/PB6 (EMG1)

P93 can also be used as the EMG pin. The main difference between P93 and PA6/PB6 is that Port 9 generates an EMG interrupt as shown in Figure 8.12.2. In the case of PA6/PB6, when the EMG function is disabled in PMD (EMGCR.EMGEN=0), no EMG interrupt (INTEMGx) is generated whereas P93 causes an EMG interrupt (INTTBE0) to be generated as soon as Port 9 is put in the EMG condition.

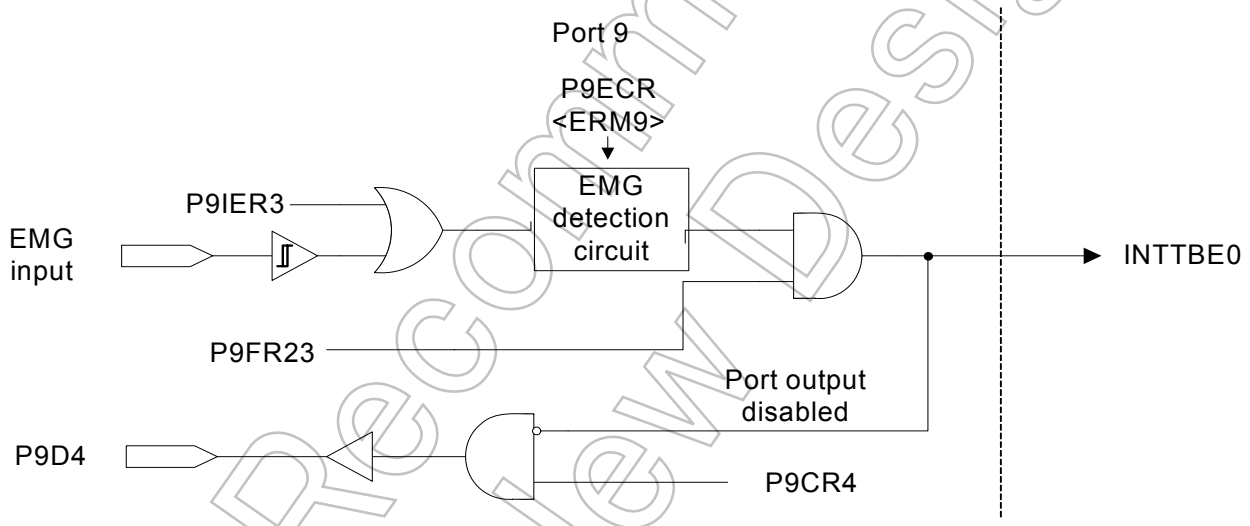


Figure 8.12.2 P93 (EMG pin) Block Diagram

Not Recommended for New Design

9. Debug Support Unit (DSU)

TMP19A71 is supplied with DSU (Debug Support Unit) mode. This function makes a subset of ports be DSU control pins.

The DSU mode has two types; Lv.1 (12-pin mode) and Lv.0 (5-pin mode). Using 12 control pins, Lv.1 provides more powerful debug function than Lv.0 does. The mode can be used like selsecting Lv.1 in the first stage of debug operation where it needs larger debug information, and Lv.0 in the last stage of debug operation since Lv.0 has less pin restriction.

9.1 DSU (EJTAG) Mode Setting

To set the DSU mode, L must be set to EJE of an external pin that is in reset cycle, and then TMP19A71 becomes in DSU (EJTAG) mode when it is started up with the DSU level, DSU-PROBE first. If DSU-PROBE is not connected, it starts from Lv.0.

Note 1: DSU disabled must be released for the Mask version.

9.1.1 Pin Status Upon the DSU (EJTAG) Mode Startup

When TMP19A71 starts in DSU (EJTAG) mode, a specific pin register automatically changes into DSU control pin regardless of its setting. In addition, as a read value of register, a set value can be read.

9.1.2 Motor Breakage Prevention

TMP19A71 has a mechanism that automatically turns its moter output OFF (RxCRn=0) to prevent the motor breakage upon the BREAK execution (including OneSTEP execution) in the DSU mode.

Intended ports are P94(TB0OUT), PA[5:0](PMD0), and PB[5:0](PMD1). Their PxCrn becomes 0 (output of a prescribed bit n of PORTx disabled) only when they are set to the motor control outputs (TB0OUT, PMD0, and PMD1). To resume the motor control, 1 is to be set to PxCRn. The motor control output, however, does not restart when it is started after changing the port setting in IDE. The port must be set during the programming.

Not Recommended for New Design

9.2 Pin Status in Reset Cycle

9.2.1 Pins Whose Status Change According to Mode; Normal and DSU

Table 9.2.1 shows the status change upon resetting of each pin. Even when a pin is not connected to DSU-PROBE in DSU mode, its status becomes the same as in DSU mode shown in Table 9.2.1.

Table 9.2.1 Pin Status in Reset Cycle

Pin	Normal Mode (EJE="H")	DSU Mode (EJE="L")
P20(TCK)	Hi-z	Hi-z(TCK)
P21(TMS)	Hi-z	Hi-z(TMS)
P22(TDI)	Hi-z	Hi-z(TDI)
P23(TDO) (注 2)	Hi-z	Undefined(TDO)
P24(DINT)	Hi-z	Hi-z(DINT)
P30(TPC)	Hi-z	Hi-z
P31(PCST0)	Hi-z	Hi-z
P32(PCST1)	Hi-z	Hi-z
P33(PCST2)	Hi-z	Hi-z
P34(DCLK)	Hi-z	Hi-z
P86(TX2/PCST3)	Hi-z	Hi-z
P87(SCLK2/CTS2/PCST4)	Hi-z	Hi-z
P94(TB0OUT/BOOT)	External "H" fixed *Note 1	External "H" fixed *Note 1
Other general-purposed I/O port	Hi-z	Hi-z
EJE	External "H" fixed	External "L" fixed
RESET	External "L" fixed	External "L" fixed
TEST0	External "L" fixed	External "L" fixed
TEST1	External "L" fixed	External "L" fixed

Note 1: These pins must be fixed externally until the reset is released.

Note 2: Even during the reset, the behavior of P23(TDO) shall be unstable until its internal current becomes stable.

Not Recommended for New Design

9.2.2 Pin Status Upon the Connection to DSU-PROBE

Upon the connection of DSU-PROBE, an output value of a port changes until the connection is completed. Since, as for pins used in Lv.1, there is only changes in output values of a pin to be used but no change in switching timing, here DCLK(P34) is described.

9.2.2.1 DSU-PROBE Connection (Lv.1)

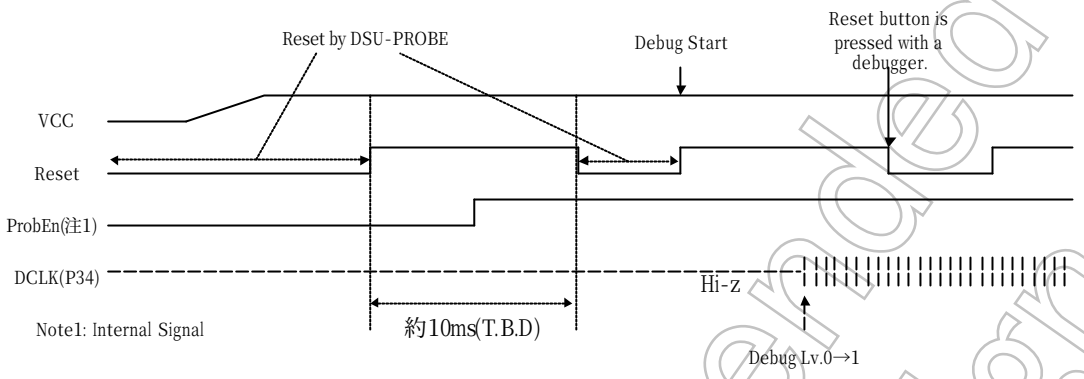


Figure 9.2.1 shows, in the connection in Lv.1, DSU-PROBE sets 1 to ProbEn of an internal register after the second reset being performed that follows the power supply. When the second reset is released, a DSU control pin used in Lv.1 mode switches to the one for DSU control and starts communication with DSU-PROBE.

Note1: For the first reset releasing cycle, refer to the operation manual of DSU-PROBE you are using.

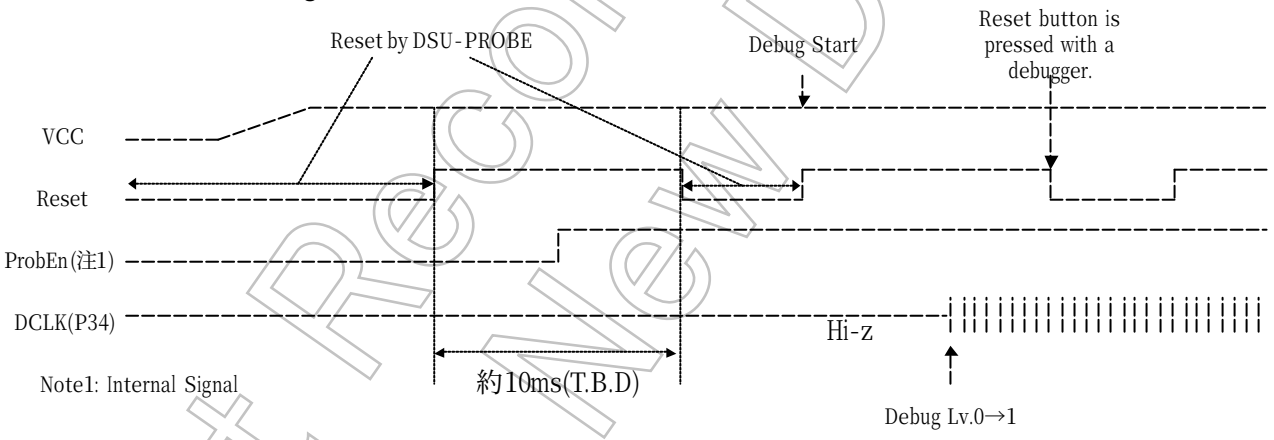


Figure 9.2.1 DSU-PROBE Connection (Lv.1)

9.2.2.2 DSU-PROBE Connection (Lv.0)

As Figure 9.2.2 DSU-PROBE Connection (Lv.0) upon the connection in Lv.0 mode, DSU-PROBE sets 1 to Proben of an internal register after the second reset being performed that follows the power supply. By setting 0 to EJE, DSU control pin to be used in Lv.0 mode behaves as the one for DSU control immediately after t he power supply.

Note1: For the first reset releasing cycle, refer to the operation manual of DSU-PROBE you are using.

Not Recommended for New Design

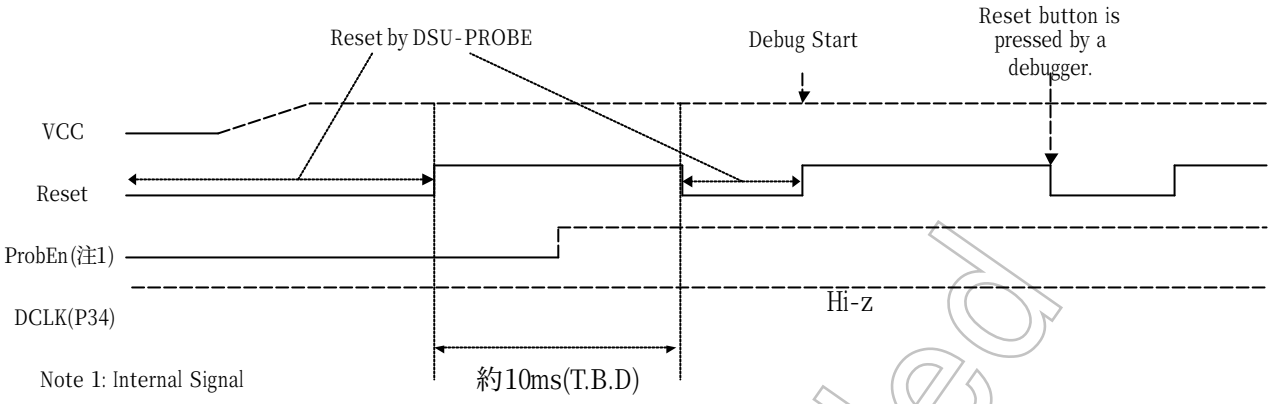


Figure 9.2.2 DSU-PROBE Connection (Lv.0)

Not Recommended for New Design

9.2.3 DSU-PROBE Disabled

This functions when debugging by using DSU-PROBE. It is an I/F exclusive for connecting to DSU-PROBE. For details of debug utilizing DSU-PROBE, refer to the operation manual of DSU-PROBE you are using. Here, DSU-PROBE Enabled/Disabled in DSU (EJTAG) mode is described.

1. DSU-PROBE Enabled/Disabled

   This device can debug by using DSU-PROBE on borad. Therefore, it has the function that disables use of DSU-PROBE (hereinafter referred to as DSU Disabled), which allows no third party to read data of incorporated flash easily. Validating the DSU Disabled makes it impossible to use DSU-PROBE.

2. DSU Disabled (Disabling debug that uses DSU-PROBE)

   User can validate the writer security functin of flash itself by issuing the protect commands described later to all the two blocks of the flash upon the program debug completion. In this condition, even if a reading is tryed by using a writer, data of incorporated flash cannot be read. Debug is impossible by using DSU-BROBE after its power is turned off unless DSU Disabled is set upon the next powering and DSU Disabled is released.

3. DSU Enabled (Enabling debug that uses DSU-PROBE)

   DSU Disabled is fail-safe to prevent any accidental release caused with such as runaway. To release DSU Disabled, 0 must be set to the DSU security mode register, SEQMOD<DSUOFF>, and the security code “0x0000_00C5” must be written in the DSU security control register, SEQCNT. Then the debug using DSU-PROBE becomes active. The security function becomes active again by setting 1 to SEQMOD<DSUOFF> without turning off the power and writing “0x0000_00C5” in SEQCNT.

4. Initialization of SEQMOD<DSUOFF>

   Flash products are not initialized by the normal reset. They are initialized only by supplying power (Power-On Reset).

   Mask products are not initialized by the reset with WDT. They are initialized by an external reset.

Not Recommended for New Design

SEQMOD (0xFFFF_E510)

	31	30	29	28	27	26	25	24
Bit Symbol	-	-	-	-	-	-	-	-
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0
Function								
	23	22	21	20	19	18	17	16
Bit Symbol	-	-	-	-	-	-	-	-
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0
Function								
	15	14	13	12	11	10	9	8
Bit Symbol	-	-	-	-	-	-	-	-
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0
Function								
	7	6	5	4	3	2	1	0
Bit Symbol	-	-	-	-	-	-	-	DSUOFF
Read/Write	R							R/W
Reset Value	0	0	0	0	0	0	0	1
Function								1: DSU Disabled 0: DSU Enabled

Note 1: This register must be accessed by 32-bit system. It is not accessible with any bit operation instruction.

SEQCNT (0xFFFF_E514)

	31	30	29	28	27	26	25	24
Bit Symbol	-	-	-	-	-	-	-	-
Read/Write	W							
Reset Value	-	-	-	-	-	-	-	-
Function	Must be written as 0x0000_00C5.							
	23	22	21	20	19	18	17	16
Bit Symbol	-	-	-	-	-	-	-	-
Read/Write	W							
Reset Value	-	-	-	-	-	-	-	-
Function	Must be written as 0x0000_00C5.							
	15	14	13	12	11	10	9	8
Bit Symbol	-	-	-	-	-	-	-	-
Read/Write	W							
Reset Value	-	-	-	-	-	-	-	-
Function	Must be written as 0x0000_00C5.							
	7	6	5	4	3	2	1	0
Bit Symbol	-	-	-	-	-	-	-	-
Read/Write	W							
Reset Value	-	-	-	-	-	-	-	-
Function	Must be written as 0x0000_00C5.							

Note 1: This register must be accessed by 32-bit system. It is not accessible with any bit operation instruction.

Not Recommended for New Design

5. Example of Use by User

Example of how to use DSU-PROBE using this function is shown below.

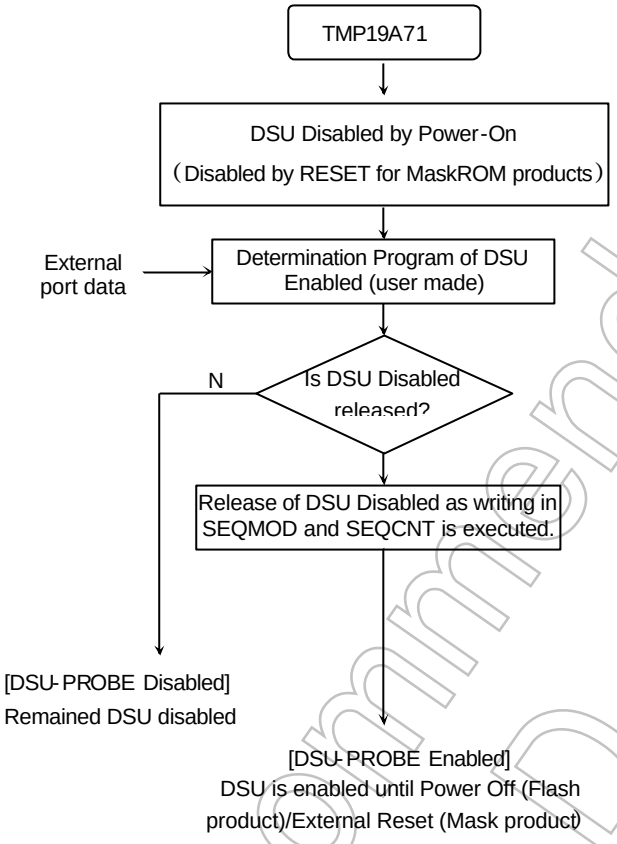


Figure 9.2.3 Example Use of DSU Disabled

Not Recommended for New Design

10. DMA Controller (DMAC)

The TMP19A71 contains an eight-channel DMA controller (DMAC).

10.1 Features

The DMAC has the following features:

(1) Eight independent DMA channels

(2) Transfer requests:
Internal transfer requests: Software initiated
External transfer requests: Interrupt signals from on-chip I/O peripherals and external interrupt pins

(3) Dual-address mode

(4) Memory-to-memory, memory-to-I/O, and I/O-to-memory transfers

(5) Transfer width:

- Memory: 32-bit
- I/O peripherals: 8-, 16-, and 32-bit

(6) Address pointers can increment, decrement or remain constant. The user can program the bit positions at which address increment or decrement occurs.

(7) Fixed channel priority

Not Recommended for New Design

10.2 Implementation

10.2.1 On-Chip DMAC Interface

Figure 10.2.1 shows how the DMAC is internally connected with the TX19A core processor and the Interrupt Controller (INTC).

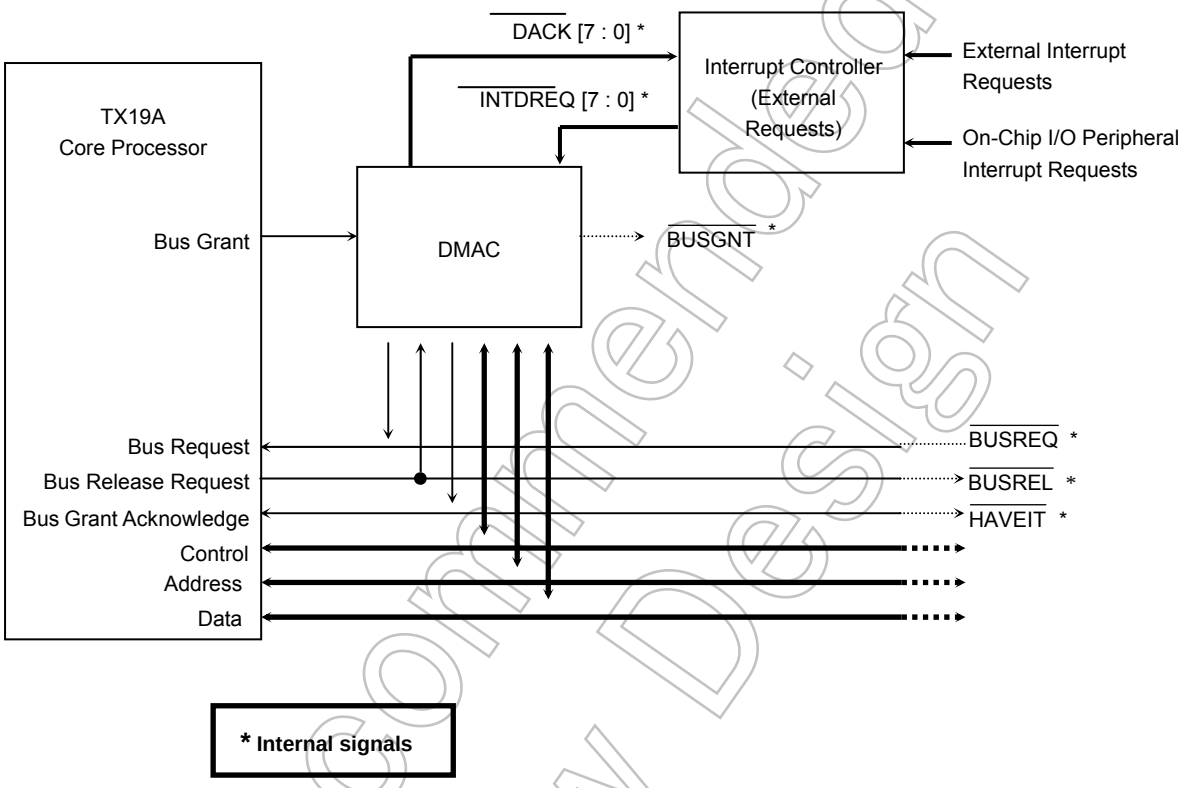


Figure 10.2.1 DMAC Connections within the TMP19A71

The DMAC provides eight independently programmable channels. With each DMA channel, there are two associated signals: a DMA request ($\overline{\text{INTDREQ}}$n) and a DMA acknowledge ($\overline{\text{DACK}}$n), where n is a channel number from 0 to 7. Channel priority is fixed. Channel 0 has the highest priority, and Channel 7 has the lowest priority.

The TX19A core processor has a snoop function. The snoop function releases the TX19A core processor's data bus to the DMAC, enabling the DMAC to access the internal ROM and internal RAM connected with the TX19A core processor. The DMAC can select whether or not to use this snoop function. For details, see "10.2.3 Snoop Function".

The DMAC can use two types of bus request: SREQ and GREQ. GREQ is used when the snoop function is not used, and SREQ is used when the snoop function is used. SREQ has higher priority than GREQ.

Note: In debug mode (CP0's Debug.DM=1), peripheral functions cannot be accessed properly with SREQ. In debug mode, do not use SREQ to access peripheral functions.

Not Recommended for New Design

10.2.2 DMAC Block

The DMAC block diagram is shown in Figure 10.2.2.

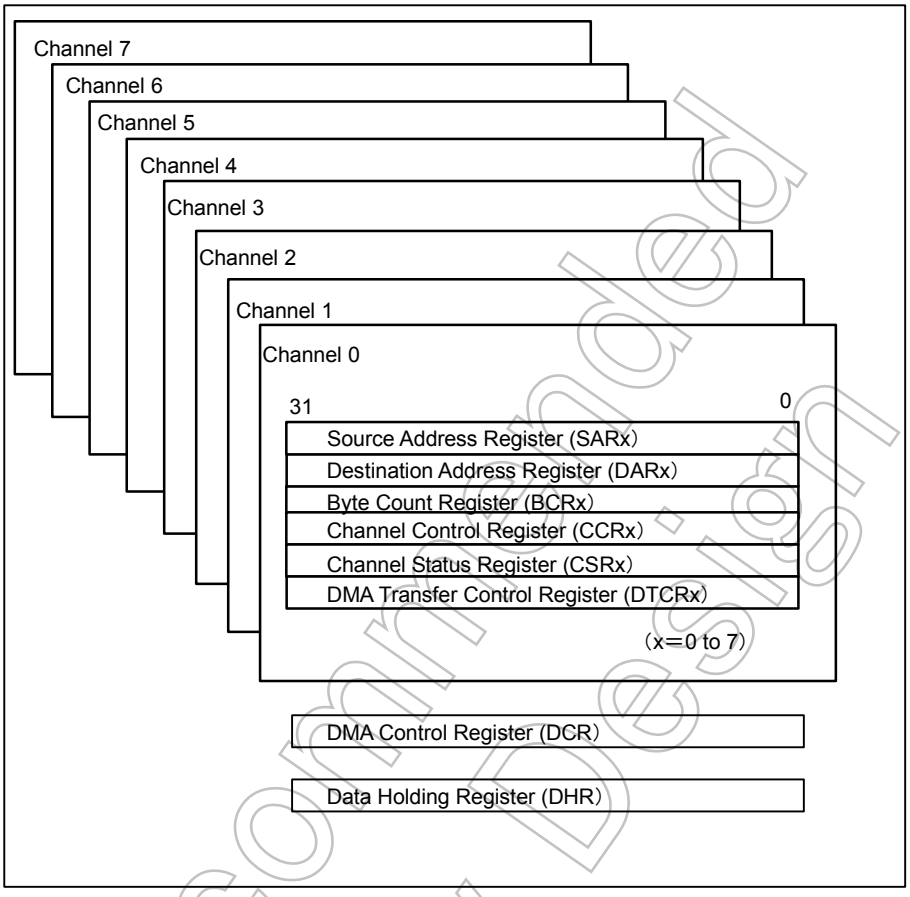


Figure 10.2.2 DMAC Block Diagram

10.2.3 Snoop Function

The TX19A core processor has the snoop function, which releases the TX19A core processor's data bus to the DMAC. When the snoop function is used, the TX19A core processor stops operating until the DMAC relinquishes the bus. The snoop function enables the DMAC to access the internal RAM and internal ROM so that these locations can be specified as source and destination addresses.

When the snoop function is not used, the DMAC cannot access the internal RAM and internal ROM. However, even when the snoop function is not used, the G-Bus is released to the DMAC. If the TX19A core processor tries to access memory or I/O through the G-Bus, pipeline operation will be stalled until the DMAC relinquishes bus mastership.

Note: When the snoop function is not used, the TX19A core processor does not release the data bus to the DMAC. In this case, if an internal RAM or ROM location is specified as a DMA source or destination address, no acknowledge signal will be returned for the bus request from the DMAC and bus operation will be locked.

Not Recommended for New Design

10.2.4 Register Description

The DMAC has fifty 32-bit registers, as listed in Table 10.2.1 .

Table 10.2.1 DMAC Register Map (1/2)

Address	Symbol	Register Name
0xFFFF_D600	CCR0	Channel Control Register (Channel 0)
0xFFFF_D604	CSR0	Channel Status Register (Channel 0)
0xFFFF_D608	SAR0	Source Address Register (Channel 0)
0xFFFF_D60C	DAR0	Destination Address Register (Channel 0)
0xFFFF_D610	BCR0	Byte Count Register (Channel 0)
0xFFFF_D618	DTCR0	DMA Transfer Control Register (Channel 0)
0xFFFF_D620	CCR1	Channel Control Register (Channel 1)
0xFFFF_D624	CSR1	Channel Status Register (Channel 1)
0xFFFF_D628	SAR1	Source Address Register (Channel 1)
0xFFFF_D62C	DAR1	Destination Address Register (Channel 1)
0xFFFF_D630	BCR1	Byte Count Register (Channel 1)
0xFFFF_D638	DTCR1	DMA Transfer Control Register (Channel 1)
0xFFFF_D640	CCR2	Channel Control Register (Channel 2)
0xFFFF_D644	CSR2	Channel Status Register (Channel 2)
0xFFFF_D648	SAR2	Source Address Register (Channel 2)
0xFFFF_D64C	DAR2	Destination Address Register (Channel 2)
0xFFFF_D650	BCR2	Byte Count Register (Channel 2)
0xFFFF_D658	DTCR2	DMA Transfer Control Register (Channel 2)
0xFFFF_D660	CCR3	Channel Control Register (Channel 3)
0xFFFF_D664	CSR3	Channel Status Register (Channel 3)
0xFFFF_D668	SAR3	Source Address Register (Channel 3)
0xFFFF_D66C	DAR3	Destination Address Register (Channel 3)
0xFFFF_D670	BCR3	Byte Count Register (Channel 3)
0xFFFF_D678	DTCR3	DMA Transfer Control Register (Channel 3)
0xFFFF_D680	CCR4	Channel Control Register (Channel 4)
0xFFFF_D684	CSR4	Channel Status Register (Channel 4)
0xFFFF_D688	SAR4	Source Address Register (Channel 4)
0xFFFF_D68C	DAR4	Destination Address Register (Channel 4)
0xFFFF_D690	BCR4	Byte Count Register (Channel 4)
0xFFFF_D698	DTCR4	DMA Transfer Control Register (Channel 4)
0xFFFF_D6A0	CCR5	Channel Control Register (Channel 5)
0xFFFF_D6A4	CSR5	Channel Status Register (Channel 5)
0xFFFF_D6A8	SAR5	Source Address Register (Channel 5)
0xFFFF_D6AC	DAR5	Destination Address Register (Channel 5)
0xFFFF_D6B0	BCR5	Byte Count Register (Channel 5)
0xFFFF_D6B8	DTCR5	DMA Transfer Control Register (Channel 5)
0xFFFF_D6C0	CCR6	Channel Control Register (Channel 6)
0xFFFF_D6C4	CSR6	Channel Status Register (Channel 6)
0xFFFF_D6C8	SAR6	Source Address Register (Channel 6)
0xFFFF_D6CC	DAR6	Destination Address Register (Channel 6)
0xFFFF_D6D0	BCR6	Byte Count Register (Channel 6)
0xFFFF_D6D8	DTCR6	DMA Transfer Control Register (Channel 6)

Not Recommended for New Design

Table 10.2.2 DMAC Register Map (2/2)

Address	Symbol	Register Name
0xFFFF_D6E0	CCR7	Channel Control Register (Channel 7)
0xFFFF_D6E4	CSR7	Channel Status Register (Channel 7)
0xFFFF_D6E8	SAR7	Source Address Register (Channel 7)
0xFFFF_D6EC	DAR7	Destination Address Register (Channel 7)
0xFFFF_D6F0	BCR7	Byte Count Register (Channel 7)
0xFFFF_D6F8	DTCR7	DMA Transfer Control Register (Channel 7)
0xFFFF_D700	DCR	DMA Control Register (DMAC)
0xFFFF_D704	Reserved	
0xFFFF_D70C	DHR	Data Holding Register (DMAC)

Note: Although the DMAC registers are 32-bit wide, they can be accessed in 8-bit or 16-bit units. For example, the CCR0[31:0] register can be divided into four 8-bit registers: CCR0[7:0]=CCR0LL, CCR0[15:8]=CCR0LH, CCR0[23:16]=CCR0HL and CCR0[31:24]=CCR0HH. For details, see "18. I/O Register Summary".

Not Recommended for New Design

There are basically no functional differences among the eight DMAC channels. In the following register descriptions, only DMAC0 is explained.

10.2.5 DMA Control Register (DCR)

DCR (0xFFFF_D700)

	7	6	5	4	3	2	1	0
Bit Symbol	Rst7	Rst6	Rst5	Rst4	Rst3	Rst2	Rst1	Rst0
Read/Write	W	W	W	W	W	W	W	W
Reset Value	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit Symbol	-							
Read/Write	R							
Reset Value	0x00							
	23	22	21	20	19	18	17	16
Bit Symbol	-							
Read/Write	R							
Reset Value	0x00							
	31	30	29	28	27	26	25	24
Bit Symbol	Rstall	-						
Read/Write	W	R						
Reset Value	0	0	0	0	0	0	0	0

Bit	Mnemonic	Field Name	Description
31	Rstall	Reset All	Performs a software reset of the DMAC. When the Rstall bit is set to 1, all the DMAC internal registers are initialized to their reset values. Any transfer requests are removed and all the eight DMA channels are put in Idle state. 0: Don't care 1: Reset the DMAC.
7	Rst7	Reset 7	Performs a software reset of DMAC Channel 7. When the Rst7 bit is set to 1, all the DMAC Channel 7 internal registers are initialized to their reset values. Any transfer requests for Channel 7 are removed and Channel 7 is put in Idle state. 0: Don't care 1: Reset DMAC Channel 7.
6	Rst6	Reset 6	Performs a software reset for DMAC Channel 6. When the Rst6 bit is set to 1, all the DMAC Channel 6 internal registers are initialized to their reset values. Any transfer requests for Channel 6 are removed and Channel 6 is put in Idle state. 0: Don't care 1: Reset DMAC Channel 6.
5	Rst5	Reset 5	Performs a software reset for DMAC Channel 5. When the Rst5 bit is set to 1, all the DMAC Channel 5 internal registers are initialized to their reset values. Any transfer requests for Channel 5 are removed and Channel 5 is put in Idle state. 0: Don't care 1: Reset DMAC Channel 5.

Not Recommended for New Design

Bit	Mnemonic	Field Name	Description
4	Rst4	Reset 4	Performs a software reset of DMAC Channel 4. When the Rst4 bit is set to 1, all the DMAC Channel 4 internal registers are initialized to their reset values. Any transfer requests for Channel 4 are removed and Channel 4 is put in Idle state. 0: Don't care 1: Reset DMAC Channel 4.
3	Rst3	Reset 3	Performs a software reset of DMAC Channel 3. When the Rst3 bit is set to 1, all the DMAC Channel 3 internal registers are initialized to their reset values. Any transfer requests for Channel 3 are removed and Channel 3 is put in Idle state. 0: Don't care 1: Reset DMAC Channel 3.
2	Rst2	Reset 2	Performs a software reset of DMAC Channel 2. When the Rst2 bit is set to 1, al the DMAC Channel 2 internal registers are initialized to their reset values. Any transfer requests for Channel 2 are removed and Channel 2 is put in Idle state. 0: Don't care 1: Reset DMAC Channel 2.
1	Rst1	Reset 1	Performs a software reset of DMAC Channel 1. When the Rst1 bit is set to 1, all the DMAC Channel 1 internal registers are initialized to their reset values. Any transfer requests for Channel 1 are removed and Channel 1 is put in Idle state. 0: Don't care 1: Reset DMAC Channel 1.
0	Rst0	Reset 0	Performs a software reset of DMAC Channel 0. When the Rst0 bit is set to 1, all the DMAC Channel 0 internal registers are initialized to their reset values. Any transfer requests for Channel 0 are removed and Channel 0 is put in Idle state. 0: Don't care 1: Reset DMAC Channel 0.

Note 1: If a software reset command is written to the DCR register immediately after the completion of the transfer cycle of a DMA transaction, the DMA-done interrupt will not be cleared. In this case, the software reset only initializes channel registers and other settings.

Note 2: Do not issue a software reset command to the DCR register via a DMA transfer.

Note 3: This register does not support bit manipulation instructions.

Not Recommended for New Design

10.2.6 Channel Control Register (CCR0)

CCR0 (0xFFFF_D600)

	7	6	5	4	3	2	1	0
Bit Symbol	SAC	DIO	DAC		TrSiz		DPS	
Read/Write	R/W	R/W	R/W		R/W		R/W	
Reset Value	0	0	00		00		00	
	15	14	13	12	11	10	9	8
Bit Symbol	-	ExR	-	-	-	-	STIO	SAC
Read/Write	R/W	R/W	R/W				R/W	R/W
Reset Value	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit Symbol	NIEn	AbIEn	-	-	-	-	-	-
Read/Write	R/W	R/W	R/W	R/W			R/W	R/W
Reset Value	1	1	1	0	0	0	1	0
	31	30	29	28	27	26	25	24
Bit Symbol	Str	-	-	-	-	-	-	-
Read/Write	W	R						W
Reset Value	0	0	0	0	0	0	0	-

Bit	Mnemonic	Field Name	Description
31	Str	Channel Start	(Reset value: −) Enables the corresponding DMA channel. Setting this bit to 1 puts the DMA channel in Ready state. DMA transfer starts as soon as a transfer request is received. Only a write of 1 is valid, and a write of 0 has no effect. This bit is always read as 0. 1: Enable the DMA channel.
24	—	(Reserved)	This bit is reserved. It must always be written as 0.
23	NIE0	Normal Completion Interrupt Enable	(Reset value: 1) 1: Enable interrupts on normal conversion completion. 0: Disable interrupts on normal conversion completion.
22	AbIE0	Abnormal Completion Interrupt Enable	(Reset value: 1) 1: Enable interrupts on abnormal conversion completion. 0: Disable interrupts on abnormal conversion completion.
21	—	(Reserved)	**This bit is reserved. It is reset to 1, but must always be written as 0.**
20 : 18	—	(Reserved)	This bit is reserved. It must always be written as 0.
17	—	(Reserved)	**This bit is reserved. It is reset to 1, but must always be written as 0.**
16 : 15	—	(Reserved)	This bit is reserved. It must always be written as 0.
14	ExR	External Request Mode	(Reset value: 0) Specifies a transfer request mode. 1: External transfer request (Interrupt request) 0: Internal transfer request (Software start)
13	—	(Reserved)	This bit is reserved. It must always be written as 0.

Not Recommended for New Design

Bit	Mnemonic	Field Name	Description
12	—	(Reserved)	**This bit is reserved. It is reset to 0, but must always be written as 1.**
11	SReq	Snoop request	(Reset value: 0) Specifies whether or not to use the snoop function. When the snoop is used, the TX19A core processor releases the data bus to the DMAC. 1: Use the snoop function. (SREQ) 0: Do not use the snoop function. (GREQ)
10	RelEn	Release Request Enable	(Reset value: 0) Specifies whether or not to respond to a bus release request from the TX19A core processor. This bit is valid only when GREQ is used. When SREQ is used, the TX19A core processor cannot issue a bus release request. 1: Respond to a bus release request from the TX19A core processor when the DMAC has bus mastership. When the TX19A core processor issues a bus release request, the DMAC relinquishes the bus upon completion of the current bus operation. 0: Do not respond to a bus release request from the TX19A core processor.
9	STIO	Source I/O	(Reset value: 0) Specifies the type of the source device. 1: I/O device 0: Memory
8 : 7	SAC	Source Address Count	(Reset value: 00) Specifies the manner in which the source address changes after each cycle. 1x: Fixed 01: Decremented 00: Incremented
6	DIO	Destination I/O	(Reset value: 0) Specifies the type of the destination device. 1: I/O device 0: Memory
5 : 4	DeAC	Destination Address Count	(Reset value: 00) Specifies the manner in which the destination address changes after each cycle. 1x: Fixed 01: Decremented 00: Incremented
3 : 2	TrSiz	Transfer Size	(Reset value: 00) Specifies the amount of data to be transferred in response to a DMA request. 11: 8 bits (1 byte) 10: 16 bits (2 bytes) 0x: 32 bits (4 bytes)
1 : 0	DPS	Device Port Size	(Reset value: 00) Specifies the bus width of the I/O device specified as a source or destination device. 11: 8 bits (1 byte) 10: 16 bits (2 bytes) 0x: 32 bits (4 bytes)

Not Recommended for New Design

Note1: The CCRn register must be programmed before placing the DMAC in Ready state.

Note 2: The DPS field has no meaning or effect on memory-to-memory transfers.

Note 3: When CCRn.DIO=1 (I/O device), do not specify the internal RAM or CG/IRC registers as a destination device.

Note 4: This register does not support bit manipulation instructions.

10.2.7 Channel Status Register (CSR0)

CSR0 (0xFFFF_D604)

	7	6	5	4	3	2	1	0
Bit Symbol	-	-	-	-	-	-	-	-
Read/Write	R					R/W		
Reset Value	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit Symbol	-	-	-	-	-	-	-	-
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit Symbol	NC	AbC	-	BES	BED	Conf	-	-
Read/Write	R/W			R				
Reset Value	0	0	0	0	0	0	0	0
	31	30	29	28	27	26	25	24
Bit Symbol	Act	-	-	-	-	-	-	-
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0

Bit	Mnemonic	Field Name	Description
31	Act	Channel Active	(Reset value: 0) Indicates whether or not the DMA channel is in Ready state. 1: The DMA channel is in Ready state. 0: The DMA channel is not in Ready state.
23	NC	Normal Completion	(Reset value: 0) If set, the DMA channel has terminated by normal completion. If the NIE0 bit in the CCR0 register is set, an interrupt is generated. The NC bit is cleared by writing a 0 to it. Clearing the NC bit causes the interrupt to be cleared. The NC bit must be cleared to prior to starting the next transfer. An attempt to set the Str bit in the CCE0 when NC=1 will cause an error. A write of 1 has no effect on this bit. 1: The DMA channel has terminated by normal completion. 0: The DMA channel has not terminated by normal completion.

Not Recommended for New Design

Bit	Mnemonic	Field Name	Description
22	AbC	Abnormal completion	(Reset value: 0) If set, the DMA channel has terminated with an error. If the AbIE0 bit in the CCR0 register is set, an interrupt is generated. The AbC bit can be cleared by writing a 0 to it. Clearing the AbC bit causes the interrupt to be cleared and the BES, BED and Conf bits to be also cleared. The AbC bit must be cleared prior to starting the next transfer. An attempt to set the Str bit in the CCR0 when AbC=1 will cause an error. A write of 1 has no effect on this bit. 1: The DMA channel has terminated with an error. 0: The DMA channel has not terminated with an error.
21	—	(Reserved)	This bit is reserved. It must always be written as 0.
20	BES	Source Bus Error	(Reset value: 0) 1: A bus error has occurred during the source read cycle. 0: A bus error has not occurred during the source read cycle.
19	BED	Destination Bus Error	(Reset value: 0) 1: A bus error has occurred during the destination write cycle. 0: A bus error has not occurred during the destination write cycle.
18	Conf	Configuration Error	(Reset value: 0) 1: A configuration error is present. 0: No configuration error is present.
2 : 0	—	(Reserved)	This bit is reserved. It must always be written as 0.

Not Recommended for New Design

10.2.8 Source Address Register (SAR0)

SAR0 (0xFFFF_D608)

	7	6	5	4	3	2	1	0
Bit Symbol	SAddr							
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit Symbol	SAddr							
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit Symbol	SAddr							
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
	31	30	29	28	27	26	25	24
Bit Symbol	SAddr							
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0

Bit	Mnemonic	Field Name	Description
31 : 0	SAddr	Source Address	(Reset value: −) Contains the **physical address** of the source device. The address changes as programmed in the SAC and TrSiz fields in the CCR0 and the SACM field in the DTCR0.

Not Recommended for New Design

10.2.9 Destination Address Register (DAR0)

DAR0 (0xFFFF_D60C)

	7	6	5	4	3	2	1	0
Bit Symbol	DAddr							
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit Symbol	DAddr							
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit Symbol	DAddr							
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
	31	30	29	28	27	26	25	24
Bit Symbol	DAddr							
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0

Bit	Mnemonic	Field Name	Description
31 : 0	DAddr	Destination Address	(Reset value: −) Contains the **physical address** of the destination device. The address changes as programmed in the DAC and TrSiz fields in the CCR0 and the DACM field in the DTCR0.

Not Recommended for New Design

10.2.10 Byte Count Register (BCR0)

BCR0 (0xFFFF_D610)

	7	6	5	4	3	2	1	0
Bit Symbol	BC							
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit Symbol	BC							
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit Symbol	BC							
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
	31	30	29	28	27	26	25	24
Bit Symbol	-	-	-	-	-	-	-	-
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0

Bit	Mnemonic	Field Name	Description
23 : 0	BC	Byte Count	(Reset value: −) Contains the number of bytes left to transfer on the DMA channel. The count is decremented by 1, 2 or 4 (as determined by the TrSiz field in the CCR0 register) for each successful transfer.

Not Recommended for New Design

10.2.11 DMA Transfer Control Register (DTCR0)

DTCR0 (0xFFFF_D618)

	7	6	5	4	3	2	1	0
Bit Symbol	-	-	DACM			SACM		
Read/Write	R		R/W					
Reset Value	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit Symbol	-	-	-	-	-	-	-	-
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit Symbol	-	-	-	-	-	-	-	-
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0
	31	30	29	28	27	26	25	24
Bit Symbol	-	-	-	-	-	-	-	-
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0

Bit	Mnemonic	Field Name	Description
5 : 3	DACM	Destination Address Count Mode	Selects the manner in which the destination address is incremented or decremented. 000: Counting begins with bit 0 of the DAR0. 001: Counting begins with bit 4 of the DAR0. 010: Counting begins with bit 8 of the DAR0. 011: Counting begins with bit 12 of the DAR0. 100: Counting begins with bit 16 of the DAR0. 101: Reserved 110: Reserved 111: Reserved
2 : 0	SACM	Source Address Count Mode	Selects the manner in which the source address is incremented or decremented. 000: Counting begins with bit 0 of the SAR0. 001: Counting begins with bit 4 of the SAR0. 010: Counting begins with bit 8 of the SAR0. 011: Counting begins with bit 12 of the SAR0. 100: Counting begins with bit 16 of the SAR0. 101: Reserved 110: Reserved 111: Reserved

Not Recommended for New Design

10.2.12 Data Holding Register (DHR)

DHR (0xFFFF_D70C)

	7	6	5	4	3	2	1	0
Bit Symbol	DOT							
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
Bit Symbol	DOT							
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit Symbol	DOT							
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
	31	30	29	28	27	26	25	24
Bit Symbol	DOT							
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0

Bit	Mnemonic	Field Name	Description
31 : 0	DOT	Data on Transfer	(Reset value: -) Contains data read from the source address during a dual-address operation.

Not Recommended for New Design

10.3 Operation

This section describes the operation of the DMAC.

10.3.1 Overview

The DMAC is a high-speed 32-bit DMA controller used to quickly move large blocks of data between I/O peripherals and memory without intervention of the TX19A core processor.

(1) Devices supported for the source and destination

The DMAC handles data transfers from memory to memory and between memory and I/O peripherals. The device from which data is transferred is referred to as a source device, and the device to which data is transferred is referred to as a destination device. Both memory and I/O peripherals can be a source or destination device. The DMAC supports data transfers from memory to I/O peripherals, from I/O peripherals to memory, and from memory to memory, but not from I/O peripherals to I/O peripherals.

DMA protocols for memory and I/O peripherals differ in accessing an I/O peripheral. To access an I/O peripheral, the DMAC asserts the $\overline{\text{DACKn}}$ (n = channel number) signal to indicate that data is being transferred in response to a previous transfer request. Because each DMA channel has only one $\overline{\text{DACKn}}$ signal, the DMAC cannot handle data transfers between two I/O peripherals.

Interrupt requests can be programmed to be a trigger to initiate a DMA process instead of requesting an interrupt to the TX19A core processor. If so programmed, the Interrupt Controller (INTC) forwards a DMA request to the DMAC. The DMA request coming from the INTC is cleared when the INTC receives a $\overline{\text{DACKn}}$ from the DMAC. Consequently, a DMA request for a transfer to/from an I/O peripheral is cleared after each DMA bus cycle (i.e., every time the number of bytes programmed into the CCRn.TrSiz field is transferred). On the other hand, during memory-to-memory transfer, the $\overline{\text{DACKn}}$ signal is not asserted until the byte count register (BCRn) reaches zero. Therefore, memory-to-memory transfer can continuously move large blocks of data in response to a single DMA request.

The TMP19A71 on-chip I/O peripherals are handled as memory. For example, data transfers between the TMP19A71 on-chip I/O peripheral and on-chip memory is discontinued after every DMA bus cycle. Nonetheless, until the BCRn register reaches zero, the DMAC remains in Ready state to wait for the next transfer request. Data transfer is continued until the byte count register (BCRn) reaches zero.

Not Recommended for New Design

(2) Exchanging bus mastership (bus arbitration)

In response to a DMA request, the DMAC issues a bus request to the TX19A core processor. When the DMAC receives a bus grant signal from the TX19A core processor, it assumes bus mastership to service the DMA request.

The DMAC can select whether or not to use the snoop function in requesting bas masterhip to the TX19A core processor. The snoop function releases the TX19A core processor's data bus to the DMAC. This selection is made for each channel by programming the SReq bit in the CCRn register.

The TX19A core processor may generate a bus release request to the DMAC. Whether or not to respond to a bus release request from the TX19A core processor is specified for each channel in the ReIEn bit in the CCRn register. The setting of this bit is valid only when the snoop function is not used (GREQ). When the snoop function is used (SREQ), the TX19A core processor cannot generate a bus release request signal.

The DMAC relinquishes the bus to the TX19A core processor when there is no pending DMA request to be serviced.

Note1: The $\overline{\text{NMI}}$ interrupt is left pending while the DMAC has control of the bus.

Note 2: Do not place the TMP19A71 in Halt mode while the DMAC is operating.

(3) Transfer request generation

Each DMA channel supports two types of request generation methods: internal and external. Internal requests are those generated within the DMAC. The DMA channel is started as soon as the Str bit in the CCRn register is set. The channel immediately requests the bus and begins transferring data.

If a channel is programmed for external request and the Str bit is set, the $\overline{\text{INTDREQ}}$n signal asserted by the INTC causes the channel to request the bus and begin a transfer. The DMAC can be programmed to recognize a transfer request with the low level of the $\overline{\text{INTDREQ}}$n signal.

(4) Data transfer mode

The TMP19A71 DMAC supports dual-address transfers, but not single-address transfers.

The dual-address mode allows data to be transferred from memory to memory and between memory and an I/O peripheral. In this mode, the DMAC explicitly addresses both the source and destination devices. The DMAC also generates a $\overline{\text{DACK}}$n signal when accessing an I/O peripheral. In dual-address mode, a transfer takes place in two DMA bus cycles: a source read cycle and a destination write cycle. In the source read cycle, the data being transferred is read from the source address and put into the DMAC internal Data Holding Register (DHR). In the destination write cycle, the DMAC writes data in the DHR to a destination address.

Not Recommended for New Design

(5) DMA channel operation

The DMAC has eight independent DMA channels 0 to 7. Setting the Start (Str) bit in the CCRn (n = channel number) enables a particular channel and puts it in Ready state.

When a DMA request is detected in any of the channels in Ready state, the DMAC arbitrates for the bus and begins a transfer. When no DMA request is pending, the DMAC relinquishes the bus to the TX19A core processor and returns to Ready state. The channel can terminate by normal completion or from an error of a bus cycle. When a channel terminates, that channel is put in Idle state. Interrupts can be generated by error termination or by normal channel termination.

Figure 10.3.1 shows general state transitions of a DMA channel.

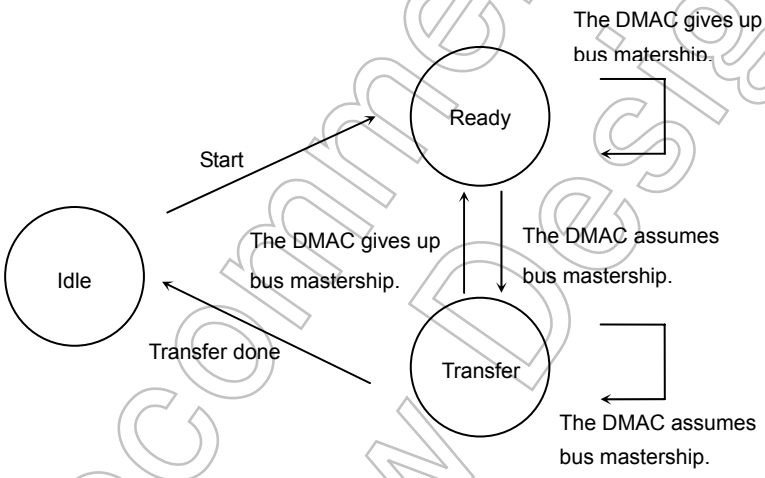


Figure 10.3.1 DMA Channel State Transitions

Not Recommended for New Design

(6) Summary of transfer modes

The DMAC can perform data transfers according to the combination of mode settings, as shown in the table below.

Table 10.3.1 DMAC Mode Combinations

Transfer Request	Edge/Level	Address Mode	Data Flow
Internal (Software)	—	Dual	Memory-to-memory
External (Interrupt)	Low level (INTDREQn)		Memory-to-memory
			Memory- to-I/O
			I/O-to-memory

(7) Address change options

Address pointers can increment, decrement or remain constant. The SAC and DAC fields in the CCRn respectively select address change directions for the Source Address Register (SARn) and the Destination Address Register (DARn). While memory addresses can be programmed to increment, decrement or remain constant, I/O addresses must be programmed to remain constant. When an I/O peripheral is selected as the source or destination device, the SAC or DAC field in the CCRn must be set to 1x (address fixed).

The SACM and DACM fields in the DTCRn provides options to program bit positions at which the source and destination addresses are incremented or decremented after each transfer. The bit position can be bit 0, 4, 8, 12, or 16. Use of bit 0 is the regular increment/decrement mode in which the address changes by 1, 2, or 4, according to the setting of the CCRn.TrSiz field. When bit 4, 8, 12 or 16 is selected, the specified bit of the address changes by 1 regardless of the CCRn.TrSiz field.

Two examples of how increment/decrement modes affect address changes are shown below.

Example 1: When address bit 0 is selected in the SACM field and address bit 4 is selected in the DACM field

SAC: Programmed to increment the source address
DAC: Programmed to increment the destination address
TrSiz: Programmed to a transfer size of 32 bits
Source address: 0xA000_1000
Destination address: 0xB000_0000
SACM: 000→Bit 0 is the source address bit at which address increment occurs.
DACM: 001→Bit 4 is the destination address bit at which address increment occurs.

	Source	Destination
1st transfer	0xA000_1000	0xB000_0000
2nd transfer	0xA000_1004	0xB000_0010
3rd transfer	0xA000_1008	0xB000_0020
4th transfer	0xA000_100C	0xB000_0030
	…	…

Not Recommended for New Design

Example 2: When address bit 8 is selected in the SACM field and address bit 0 is selected in the DACM field

SAC: Programmed to decrement the source address
DAC: Programmed to decrement the destination address
TrSiz: Programmed to a transfer size of 16 bits
Source address: 0xA000_0000
Destination address: 0xB000_0000
SACM: 000→Bit 8 is the source address bit at which address increment occurs.
DACM: 001→Bit 0 is the destination address bit at which address increment occurs.

	Source	Destination
1st transfer	0xA000_0000	0xB000_0000
2nd transfer	0x9FFF_FF00	0xAFFF_FFFE
3rd transfer	0x9FFF_FE00	0xAFFF_FFFC
4th transfer	0x9FFF_FD00	0xAFFF_FFFA
	…	…

10.3.2 Transfer Request Generation

A DMA request must be issued for the DMAC to initiate a data transfer. Each DMA channel in the DMAC supports two types of request generation method: internal and external. In either request generation mode, once a DMA channel is started, a DMA request causes the DMAC to arbitrate for the bus and begin transferring data.

- Internal request generation

  A channel is programmed for internal request by clearing the ExR bit in the CCRn. In internal request generation mode, a transfer request is generated as soon as the Str bit in the CCRn is set.

  An internally generated request keeps a transfer request pending until the transfer is complete. If no transition to a higher-priority DMA channel or a bus master occurs, the channel will use 100% of the available bus bandwidth to transfer all data continuously.

  Internally generated requests support only memory-to-memory transfer.

- External request generation

  A channel is programmed for external request by setting the ExR bit in the CCRn. In external request generation mode, setting the Str bit in the CCRn puts the channel in Ready sate. While in Ready state, assertion of the $\overline{\text{INTDREQ}}$n signal (where n is the channel number) coming from the Interrupt Controller (INTC) causes a transfer request to be generated. Externally generated requests support data transfers from memory to memory and between memory and an I/O peripheral.

  The TMP19A71 can recognize a transfer request with the low level of $\overline{\text{INTDREQ}}$n.

  The transfer size, i.e., the amount of data to be transferred in response to a transfer request, is programmed in the TrSiz field in the CCRn. The transfer size can be 32 bits, 16 bits or 8 bits.

  Transfer request generation by $\overline{\text{INTDREQ}}$n is described in detail below.

Not Recommended for New Design

(1) Transfer request coming from the INTC

A transfer request is removed by assertion of the $\overline{\text{DACK}}$n signal (where n is the channel number). $\overline{\text{DACK}}$n is asserted: 1) when an I/O peripheral bus cycle has completed and 2) when the Byte Count Register (BCRn) has reached zero in memory-to-memory transfer. Consequently, a memory-to-I/O or I/O-to-memory transfer request terminates after one DMA bus cycle completes, whereas memory-to-memory transfer can continuously move large blocks of data in response to a single DMA request.

The INTC might clear $\overline{\text{INTDREQ}}$n before the DMAC accepts it and begins a data transfer. It must be noted that, even if that happens, a DMA bus cycle might be executed after the interrupt request has been cleared.

Not Recommended for New Design

10.3.3 DMA Address Modes

DMA transfer is generally performed in either of two address modes: dual-address mode and single-address mode. In dual-address mode, both the source and destination devices are explicitly addressed. In single-address mode, only either the source device or the destination device is explicitly addressed. The TMP19A71, however, supports dual-address mode only.

In dual-address mode, two bus transfers occur: a read from the source device and a write to the destination device. In the source read cycle, data is read from the source address and placed in the DMAC internal Data Holding Register (DHR). Then, in the destination write cycle, the data held in the DHR is written to the destination address.

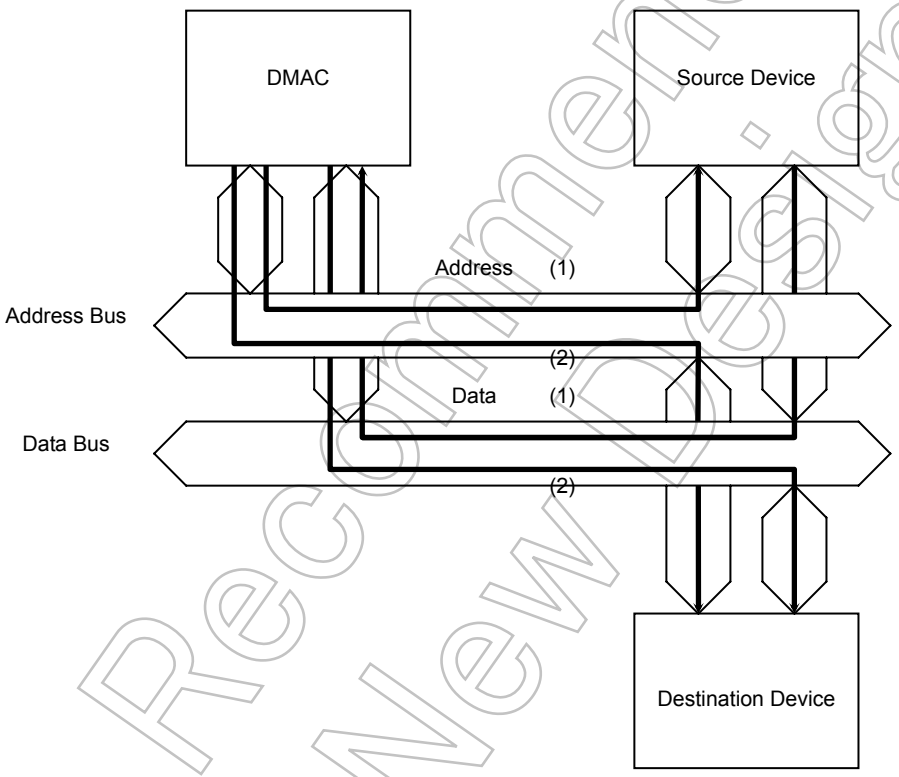


Figure 10.3.32 Dual-Address Transfer Mode

The transfer size programmed into the CCRn.TrSiz field determines the amount of data that is transferred from a source device in response to a DMA request. The transfer size can be 32 bits, 16 bits or 8 bits.

The internal DHR is a 32-bit register that serves as a buffer for the data being transferred from a source device to a destination device during dual-address mode.

Memory accesses occur in a manner to fulfill the CCRn.TrSiz setting.

Memory-to-I/O and I/O-to-memory DMA transfers are governed by the setting of the CCRn.DPS field in addition to the setting of CCRn.TrSiz. The DPS field defines the port size of a source or destination I/O peripheral. The I/O port size can be 32 bits, 16 bits or 8 bits.

Not Recommended for New Design

If the transfer size is equal to the I/O port size, an I/O access takes a single read or single write cycle. If the I/O port size is less than the programmed transfer size, the internal 32-bit DHR serves as a buffer for the data being transferred. For example, assume that the transfer size is programmed to 32 bits. If the source I/O port size is 8 bits and the destination memory width is 32 bits, then four 8-bit read cycles occur, followed by a 32-bit write cycle. The 32 bits of data are buffered in the DHR until the destination write cycle occurs.

Source and destination addresses can be programmed to increment or decrement after each transfer. The BRCn is decremented by TrSiz for each data transfer. It is forbidden to program the device port size (DPS) to a value greater than the DMA transfer size (TrSiz). The relationships between TrSiz and DPS are summarized below.

Table 10.3.2 DMA Transfer Sizes and Device Port Sizes (in Dual-Address Mode)

TrSiz	DPS	Number of I/O Bus Cycles
0x (32 bits)	0x (32 bits)	1
0x (32 bits)	10 (16 bits)	2
0x (32 bits)	11 (8 bits)	4
10 (16 bits)	0x (32 bits)	Setting prohibited
10 (16 bits)	10 (16 bits)	1
10 (16 bits)	11 (8 bits)	2
11 (8 bits)	0x (32 bits)	Setting prohibited
11 (8 bits)	10 (16 bits)	Setting prohibited
11 (8 bits)	11 (8 bits)	1

Not Recommended for New Design

10.3.4 DMA Channel Operation

Each DMA channel is started by setting the Str bit in the CCRn to 1. Once started, the DMAC checks the channel setups for configuration errors. If no configuration error is present, the channel enters Ready state.

When a DMA request is detected while in Ready state, the DMAC arbitrates for the bus and begins transferring data.

The channel can terminate by normal completion or from an error. The state of termination is indicated in the CSRn.

Channel startup

A DMA channel is started by setting the Str bit in the CCRn.

Once started, the DMAC checks the channel setups for configuration errors. If a configuration error is detected, the channel terminates abnormally. If no configuration error is present, the channel enters Ready state. Once a channel enters Ready state, the Act bit in the CSRn is set to 1.

If the channel is programmed for internal requests, the channel requests the bus and starts transferring data immediately. If the channel is programmed for external requests, $\overline{\text{INTDREQ}}$n must be asserted before the channel requests the bus.

Channel termination

A DMA channel can terminate by normal completion or from an error. The status of a DMA operation can be determined by reading the CSRn.

A channel terminates abnormally if an attempt is made to set the Str bit in the CCRn when the NC or AbC bit in the CSRn is set.

Normal termination

A DMA channel terminates by normal completion in the following case. Normal completion always occurs at the boundary of transfers programmed into the CCRn. TrSize field.

- Data transfers have terminated, with the BCRn decremented to 0.

Abnormal termination

The following summarizes the cases in which a DMA channel terminates from an error.

- Configuration errors

  A configuration error results when the channel initialization contains inconsistencies or errors. A configuration error is reported before any data transfer takes place; therefore, in case of a configuration error, the SARn, DARn and BCRn remain unaltered. When a DMA channel has terminated from a configuration error, the AbC and Conf bits in the CSRn are set. A configuration error occurs for the following cases:

  – Both the SIO and DIO bits in the CCRn are set to 1.
  – The CCRn.Str bit is set to 1 when the NC or AbC bit in the CSRn is set to 1.
  – The BCRn contains a value that is not an integer multiple of the transfer size programmed into the CCRn.TrSiz field.
  – The SARn or DARn contains a value that is not an integer multiple of the

Not Recommended for New Design

transfer size programmed into the CCRn.TrSiz field.

– The CCRn.TrSiz and CCRn.DPS fields contain illegal combinations.

– The CCRn.Str bit is set to 1 when the BCRn contains a value of zero.

- Bus errors

When a DMA channel has terminated from a bus error, the AbC bit and the BES or the BED bit in the CSRn are set.

– A bus error has been reported during a source read or destination write cycle.

Note: The contents of the BCRn, SARn and DARn are not guaranteed when a channel has terminated due to a bus error. Chapter 18 lists the reserved addresses that, if accessed, cause a bus error.

10.3.5 DMA Channel Priority

The DMAC provides a fixed priority for the eight channels, with channel 0 always having the highest priority and channel 7 the lowest. For example, when transfer requests occur on channels 0 and 1 simultaneously, the channel 0 request is serviced first. The channel 1 request is left pending. In order for the channel 1 request to be serviced, it must be maintained until data transfer completes on channel 0. Remember that the internally generated request is kept until the servicing of the request is finished. External transfer requests come from the Interrupt Controller (INTC). The INTC can program any interrupts to be used as a DMA trigger instead of as an interrupt request. If such an interrupt is programmed to be edge-sensitive, the INTC internally maintains a transfer request. However, a level-sensitive interrupt is not held in the INTC; thus the interrupt request signal must remain asserted until the servicing of the DMA request begins.

A higher-priority channel always gets the attention of the DMAC. If a transfer request occurs on channel 0 while a request on channel 1 is being serviced, the servicing of the channel 1 request is suspended temporarily in order to service the channel 0 request first. After the channel 0 request has been serviced, channel 1 resumes the remaining data transfer.

Channel transitions take place at the boundary of a transfer size programmed for the current channel being serviced; that is, after all data in the DHR are written to a destination.

Interrupts

The DMAC can generate an interrupt request (INTDMAn) to the TX19A core processor upon completion of a channel operation: either by normal channel termination or by abnormal termination of a bus cycle.

- Normal completion interrupt

When a channel operation terminates by normal completion, the NC bit in the CSRn is set to 1. At this time, if the NIEn bit in the CCRn is set, an interrupt request is generated to the TX19A core processor.

- Abnormal completion interrupt

When a channel operation terminates abnormally, the AbC bit in the CSRn register is set to 1. At this time, if the AbIEn bit in the CCRn is set, an interrupt request is generated to the TX19A core processor.

Not Recommended for New Design

10.4 DMA Transfer Timing

All DMAC operations are synchronous to the rising edges of the internal system clock.

10.4.1 Dual-Address Mode

- Memory-to-memory transfer

  Figure 10.4.1 shows a DMA cycle from one external 16-bit memory to another, with the transfer size programmed to 16 bits. A block of data is transferred until the BCRn register reaches 0.

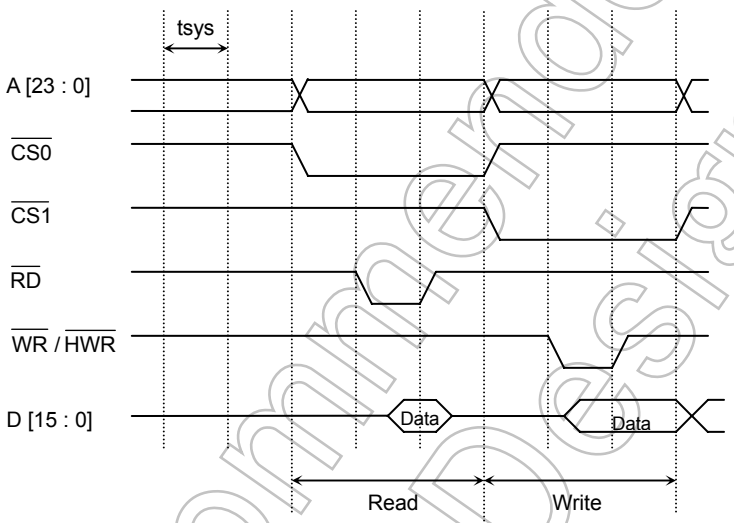


Figure 10.4.1 Memory-to-Memory Transfer (Dual-Address Mode)

- Memory-to-I/O transfer

  Figure 10.4.2 shows a DMA cycle from a 16-bit memory to an 8-bit I/O peripheral, with the transfer size programmed to 16 bits.

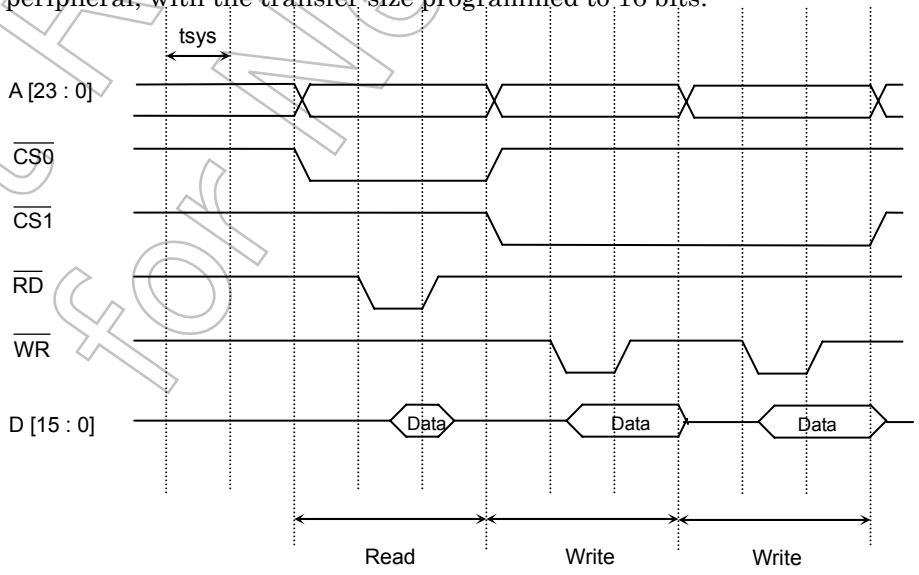


Figure 10.4.2 Memory-to-I/O Transfer (Dual-Address Mode)

Not Recommended for New Design

- I/O-to-memory transfer

  Figure 10.4.3 shows a DMA cycle from an 8-bit I/O peripheral to a 16-bit memory, with the transfer size programmed to 16 bits.

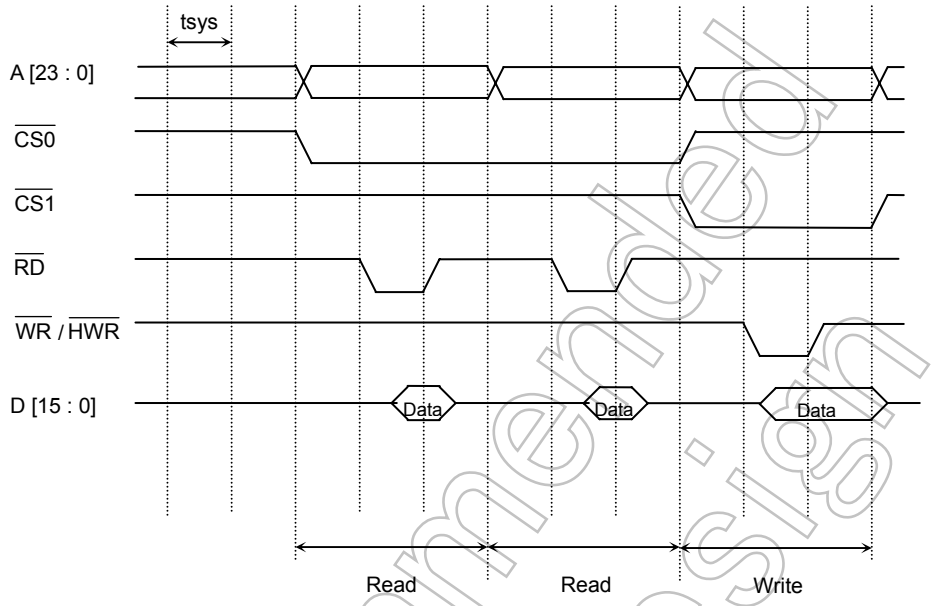


Figure 10.4.3 I/O-to-Memory Transfer (Dual-Address Mode)

Not Recommended for New Design

10.4.2 Programming Example

The following illustrates the programming required to transfer data from an SIO receive buffer (SC1BUF) to the on-chip RAM.

(1) DMAC settings:

- DMA channel used: Channel 0
- Source address: SC1BUF
- Destination address: 0xFFFF_9800 (physical address)
- Number of bytes transferred: 256

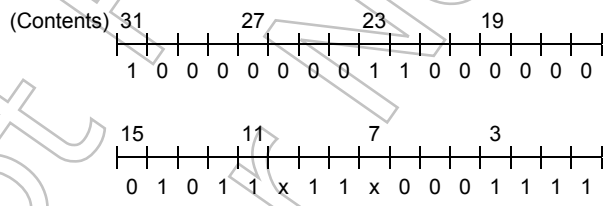
(2) SIO settings:

- Data format: 8 bits, UART
- SIO channel used: Channel 1
- Transfer rate: 9600 bps

DMA channel 0 is used for the transfer. The SIO1 receive interrupt is used as a trigger to start the DMA channel 0.

(3) DMA channel 0 settings:

```
DCR     ←   0x8000_0000              /* Reset DMAC * /
IMR56   ←   15          7       0
            xxxx, xxxx, x100, x100   /* Interrupt level = 4 (arbitrary) * /
ICLR    ←   0xe0                     /* IVR [8:0] * /
DTCR0   ←   0x0000_0000              /* DACM = 000 * /
                 ....                /* SACM = 000 * /
SAR0    ←   0xFFFF_F208              /* Physical address of SC1BUF */
DAR0    ←   0xFFFF_9800              /* Physical address of destination */
BCR0    ←   0x0000_00FF              /* 256 (Number of bytes to be transferred) /
CCR0    ←   0x80C0_5B0F
```



(4) SIO channel 1 settings:

```
IMR51   ←   31          15
            xxxx, xxxx, x101, x000   /* Use INTRX1 as a DMA trigger and select DMA ch.0 * /
ICLR    ←   0xCC                     /* IVR [8:0]; clear INTRX1 * /
SC1MOD0 ←   0x29                     /* UART mode, 8-bit data format * /
SC1CR   ←   0x00
BR1CR   ←   0xB5                     /* @IMCLK = 28 MHz (approx. 9615 bps) */
BR1ADD  ←   0x05                     /* Baud rate generator divisor */
```

Not Recommended for New Design

11. 16-Bit Timer/Event Counters (TMRBs)

The TMP19A71 has a 16-bit timer/event counter consisting of four identical channels (TMRB0 to TMRB3). Each channel has the following three basic operating modes:

- 16-Bit Interval Timer mode
- 16-Bit Event Counter mode
- 16-Bit Programmable Pulse Generation (PPG) mode

Each channel has capture capability, which enables the following operations:

- Pulse width measurement
- One-shot pulse generation from an external trigger pulse

Figure 11.1.1 shows a block diagram of the TMRB0.

The main components of a TMRBn block are a 16-bit up-counter, two 16-bit timer registers (one of which is double-buffered), two 16-bit capture registers, two comparators, capture control logic and timer flip-flop logic.

Each of the four channels (TMRB0 to TMRB3) is independently programmable and functionally equivalent except for the differences shown in Table 11.1.1. In the sections that follow, any references to the TMRB0 also apply to other channels.

Table 11.1.1 Pins and Registers for the TMRB0 to TMRB3

Specifications \ Channel		TMRB0	TMRB1	TMRB2	TMRB3
External Pins	External Clock/ Capture Trigger Input	TB0IN (shared with P93)	TB1IN (shared with P70)	TB2IN (shared with P71)	TB3IN (shared with P72)
	Timer Flip-Flop Output	TB0OUT (shared with P94)	TB1OUT (shared with P84)	TB2OUT(shared with PA7)	TB3OUT (shared with PB7)
Registers (Addresses)	Timer Run Register	TB0RUN (0xFFFF_C700)	TB1RUN (0xFFFF_C720)	TB2RUN (0xFFFF_C740)	TB3RUN (0xFFFF_C760)
	Timer Mode Register	TB0MOD (0xFFFF_C704)	TB1MOD (0xFFFF_C724)	TB2MOD (0xFFFF_C744)	TB3MOD (0xFFFF_C764)
	Timer Flip-Flop Control Register	TB0FF (0xFFFF_C708)	TB1FF (0xFFFF_C728)	TB2FF (0xFFFF_C748)	TB3FF (0xFFFF_C768)
	Timer Registers	TB0REG0 (0xFFFF_C70C) TB0REG1 (0xFFFF_C710)	TB1REG0 (0xFFFF_C72C) TB1REG1 (0xFFFF_C730)	TB2REG0 (0xFFFF_C74C) TB2REG1 (0xFFFF_C750)	TB3REG0 (0xFFFF_C76C) TB3REG1 (0xFFFF_C770)
	Capture Registers	TB0CP0 (0xFFFF_C714) TB0CP1 (0xFFFF_C718)	TB1CP0 (0xFFFF_C734) TB1CP1 (0xFFFF_C738)	TB2CP0 (0xFFFF_C754) TB2CP1 (0xFFFF_C758)	TB3CP0 (0xFFFF_C774) TB3CP1 (0xFFFF_C778)
	Counter	TB0CNT (0XFFFF_C71C)	TB1CNT (0XFFFF_C73C)	TB2CNT (0XFFFF_C75C)	TB3CNT (0XFFFF_C77C)

Not Recommended for New Design

11.1 Block Diagram

Figure 11.1.1 shows a block diagram of the 16-bit timer/event counter (TMRB0).

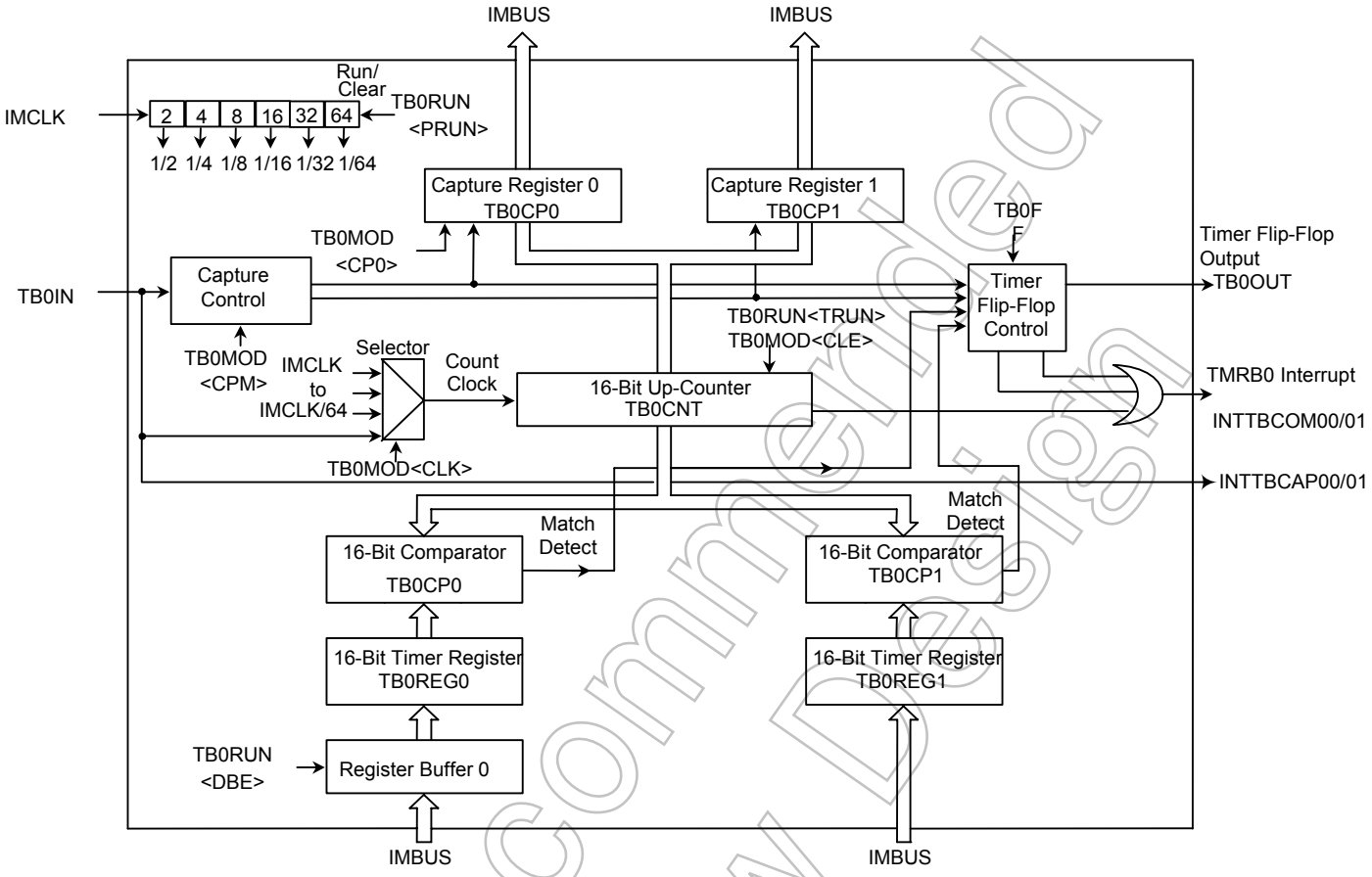


Figure 11.1.1 TMRB0 Block Diagram

Not Recommended for New Design

11.2 Timer Components

(1) Prescaler

The TMRB0 has a 6-bit prescaler that slows the rate of a clocking source to the counter. The prescaler clock source is the IMCLK selected by the PRS2 field in the CLKPRSC register within the clock generator. The prescaler output clock can be selected from IMCLK, IMCLK/2, IMCLK/4, IMCLK/8, IMCLK/16, IMCLK/32 and IMCLK/64 by programming the CLK field in the TB0MOD register.

(2) Up-Counter (TB0CNT)

The TMRB0 contains a 16-bit up-counter, which is driven by the clock selected by the CLK field in the TB0MOD register.

The clock input to the TB0CNT can be selected from seven prescaler outputs (IMCLK, IMCLK/2, IMCLK/4, IMCLK/8, IMCLK/16, IMCLK/32 and IMCLK/64) or the external clock applied to the TB0IN pin. The RUN bit in the TB0RUN register is used to start the TB0CNT and to stop and clear the TB0CNT. The TB0CNT is cleared to 0000H, if so enabled, when it reaches the value in the TB0REG0 or TB0REG1 register. This clearing can be enabled and disabled by the CLE bit in the TB0MOD register.

If the clearing is disabled, the TB0CNT acts as a free-running counter.

If the overflow interrupt is enabled in the OFI bit in the TB0RUN register, an interrupt (INTTBCOM00) is generated upon a counter overflow.

Not Recommended for New Design

(3) Timer Registers (TB0REG0, TB0REG1)

Each timer channel has two 16-bit registers containing a time constant. When the up-counter reaches the timer constant value in each timer register, the associated comparator block generates a match-detect signal.

Each of the timer registers (TB0REG0, TB0REG1) can be written with a halfword-load instruction. Although it is also possible to use a series of two byte-load instructions, be sure to use a halfword-load instruction while the TB0CNT is counting to prevent an erroneous match detect when only the first byte-load instruction has been executed. To write to the timer register while the TB0CNT is counting and double-buffering is disabled, the write timing must be managed by software.

One of the two timer registers, TB0REG0, is double-buffered. The double-buffering function can be enabled and disabled through the programming of the DBE bit in the TB0RUN register: 0 = disable, 1=enable. If double-buffering is enabled, the TB0REG0 latches a new time constant from the register buffer 0. This takes place when a match is detected between the TB0CNT and the TB0REG1.

Upon reset, the contents of the TB0REG0 and TB0REG1 are cleared to zero; thus, they must be loaded with valid values before the timer can be used. A reset clears the TB0RUN.DBE bit to 0, disabling the double-buffering function. To use this function, the TB0RUN.DBE bit must be set to 1 after loading the TB0REG0 and TB0REG1with time constants. When TB0RUN.DBE=1, the next time constant can be written to the register buffer.

The TB0REG0 and the corresponding register buffer are mapped to the same address (0xFFFF_C70C). When TB0RUN.DBE=0, a time constant value is written to both the TB0REG0 and the register buffer. When TB0RUN.DBE=1, a time constant value is written only to the register buffer. Therefore, the double-buffering function should be disabled when writing an initial time constant to each timer register.

(4) Capture Registers (TB0CP0, TB0CP1)

The capture registers are 16-bit registers used to latch the value of the up-counter (TB0CNT). Each of the capture registers can be read with a halfword-load instruction. Although it is also possible to use a series of two byte-load instructions, it is recommended to use a halfword-load instruction while the timer is counting because the register value may be updated before the second byte-load instruction is executed.

The CPM field in the TB0MOD register is used to select the timing for latching the TB0CNT value to the TB0CP0 and TB0CP1.

Furthermore, an up-counter value can be captured under software control: a write of 0 to the TB0MOD.CP0 bit causes the current TB0CNT value to be latched into the TB0CP0. To use the capture capability, the prescaler must be running (i.e., TB0RUN.PRUN=1).

Not Recommended for New Design

(5) Comparators (TB0CMP0, TB0CMP1)

The TMRB0 contains two 16-bit comparators. The TB0CMP0 block compares the output of the up-counter (TB0CNT) with a time constant value in the TB0REG0. The TB0CMP1 block compares the output of the TB0CNT with a time constant value in the TB0REG1. When a match is detected, an interrupt (INTTBCOM0x) is generated.

The TB0CMP0 does not detect a match when the TB0REG0 value is 0000H whereas the TB0CMP1 detects a match when TB0REG1=0000H. To use the match detect function of the TB0CMP1, setting TB0MOD.CLE=1 or TB0FF.INVC1=1 is required. However, if TB0REG1 is set to 0000H with TB0MOD.CLE=1, undefined operation will result.

(6) Timer Flip-Flop (TB0FF)

The timer flip-flop (TB0FF) is toggled, if so enabled, upon assertion of match-detect signals from the comparators and latch signals from the capture control logic. The toggling of the TB0FF can be enabled and disabled through the programming of the INVL1, INVL0, INVC1, INVC0, and MOD bits in the TB0FF register.

Upon reset, the TB0FF is cleared to 0. A write of 00 to the MOD field in the TB0FF causes the TB0FF to be toggled to the opposite value; a write of 01 to this field sets the TB0FFto 1; and a write of 10 to this field clears the TB0FF to 0.

The value of the TB0FF can be driven onto the TB0OUT pin, which is multiplexed with P94. The Port 9 registers (P9CR, P9FR1) must be programmed to configure the TB0OUT/P94 pin as an output from the TB0FF. After reset, the TB0OUT pin outputs 0 until the TB0FF.MOD field is set.

Not Recommended for New Design

11.3 Register Description

As shown in Table 11.3.1, the main components of the TMRBn block are a 16-bit up-counter, two 16-bit timer registers (one of which is double-buffered), two 16-bit capture registers, two comparators, capture control logic and timer flip-flop control logic. The 11-byte registers provide control over the operating modes and timer flip-flops.

Table 11.3.1 TMRB Register Map (1/2)

Address	Bits	Mnemonic	Register Name
0xFFFF_C700	8	TB0RUN	TMRB0 Run Register
0xFFFF_C704	16(8)	TB0MOD(L)	TMRB0 Mode Register (Low)
0xFFFF_C705	8	TB0MODH	TMRB0 Mode Register High
0xFFFF_C708	8	TB0FF	TMRB0 Flip-Flop Control Register
0xFFFF_C70C	16	TB0REG0	TMRB0 Compare Register 0
0xFFFF_C710	16	TB0REG1	TMRB0 Compare Register 1
0xFFFF_C714	16	TB0CP0	TMRB0 Capture Register 0
0xFFFF_C718	16	TB0CP1	TMRB0 Capture Register 1
0xFFFF_C71C	16	TB0CNT	TMRB0 Counter Register
0xFFFF_C720	8	TB1RUN	TMRB1 Run Register
0xFFFF_C724	16(8)	TB1MOD(L)	TMRB1 Mode Register (Low)
0xFFFF_C725	8	TB1MODH	TMRB1 Mode Register High
0xFFFF_C728	8	TB1FF	TMRB1 Flip-Flop Control Register
0xFFFF_C72C	16	TB1REG0	TMRB1 Compare Register 0
0xFFFF_C730	16	TB1REG1	TMRB1 Compare Register 1
0xFFFF_C734	16	TB1CP0	TMRB1 Capture Register 0
0xFFFF_C73C	16	TB1CNT	TMRB1 Counter Register

Not Recommended for New Design

Table 11.3.2 TMRB Register Map (2/2)

Address	Bits	Mnemonic	Register Name
0xFFFF_C740	8	TB2RUN	TMRB2 Run Register
0xFFFF_C744	16(8)	TB2MOD(L)	TMRB2 Mode Register (Low)
0xFFFF_C745	8	TB2MODH	TMRB2 Mode Register High
0xFFFF_C748	8	TB2FF	TMRB2 Flip-Flop Control Register
0xFFFF_C74C	16	TB2REG0	TMRB2 Compare Register 0
0xFFFF_C750	16	TB2REG1	TMRB2 Compare Register 1
0xFFFF_C754	16	TB2CP0	TMRB2 Capture Register 0
0xFFFF_C75C	16	TB2CNT	TMRB2 Counter Register
0xFFFF_C760	8	TB3RUN	TMRB3 Run Register
0xFFFF_C764	16(8)	TB3MOD(L)	TMRB3 Mode Register (Low)
0xFFFF_C765	8	TB3MODH	TMRB3 Mode Register High
0xFFFF_C768	8	TB3FF	TMRB3 Flip-Flop Control Register
0xFFFF_C76C	16	TB3REG0	TMRB3 Compare Register 0
0xFFFF_C770	16	TB3REG1	TMRB3 Compare Register 1
0xFFFF_C774	16	TB3CP0	TMRB3 Capture Register 0
0xFFFF_C77C	16	TB3CNT	TMRB3 Counter Register

Note 1: Although the TBxMOD is a 16-bit register, it can be accessed as two 8-bit registers: TBxMODL (low) and TBxMODH (high).

Note 2: The TBxCP0 and TBxCP1 can be read by two byte-load instructions. However, we recommend using a halfword-load instruction while the timer is counting as the register value may be updated between two byte-load instructions.

Note 3: The TB0REG0 and TB0REG1 can be written by two byte-load instructions. However, we recommend using a halfword-load instruction as a match with TB0CNT may be erroneously detected when only the first byte has been written.

Not Recommended for New Design

TMRB0 Run Register

TB0RUN (0xFFFF_C700)

	7	6	5	4	3	2	1	0
Bit Symbol	DBE	—	TRGSEL	CSSEL	IDL	PRUN	OFI	TRUN
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	Double-buffer 0: Disable 1: Enable	Must be set as 0.	External trigger 0: Rising edge 1: Falling edge	Counter start 0: Software start 1: External trigger	TMRB0 operation 0: Stop & keep counter value 1:Normal operation	Prescaler start 0: Stop and clear 1:Run	Overflow interrupt 0: Disable 1: Enable	Timer start 0: Stop and clear 1:Run

Note 1: The difference between stopping the timer by setting IDL=0 and TRUN=0 is that IDL=0 preserves the TBxCNT value whereas TRUN=0 clears the TBxCNT value.

Note 2: When the CSSEL bit is set to 1, the TB0CNT starts counting triggered by the TB0IN pin input as specified in the TRGSEL bit. To start counting by the external trigger signal, the TRUN bit must be set to 1. If TRUN=0, the counter remains stopped and cleared as in the case of software start.

Note 3: Once the counter is started by an external trigger, the trigger is kept internally. To accept a next external trigger, it is necessary to clear and stop the counter by clearing the TRUN bit to 0 and then to set TRUN=1 again. Any external triggers accepted before the TRUN bit is cleared to 0 are ignored.

Not Recommended for New Design

TMRB0 Mode Register

TB0MOD(L) (0xFFFF_C704)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	CPM		CLE	CLK		
Read/Write	W		R/W					
Reset Value	—	—	0	0	0	000		
Function			Capture timing 00: Disable 10: Latches the counter value into TB0CP0 at rising edges of TB0IN and generates INTTBCAP01. Latches the counter value into TB0CP1 at falling edges of TB0IN and generates INTTBCA00. Others: Reserved		Up-counter clear control 0:Clearing disabled 1: Clears up-counter upon a match with TB0REG1	Clock source 000: TB0IN pin input (TMRB0 only) 001: IMCLK 010: IMCLK/2 011: IMCLK/4 100: IMCLK/8 101: IMCLK/16 110: IMCLK/32 111: IMCLK/64		

TB0MODH (0xFFFF_C705)

	15	14	13	12	11	10	9	8
Bit Symbol	—	—	—	—	—	—	—	CP0
Read/Write	R						R/W	W
Reset Value	0	0	0	0	0	0	0	1
Function							Must be set as 0.	Software capture control 0: Software capture 1:Don't care This bit is always read as 1.

Note: This register does not support bit manipulation instructions.

Not Recommended for New Design

TMRB0 Flip-Flop Control Register

TB0FF (0xFFFF_C708)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	INVL1	INVL0	INVC1	INVC0	MOD	
Read/Write	W		R/W				W	
Reset Value	—	—	0	0	0	0	11	
Function			When the up-counter value is latched into TB0CP1 0: Toggle-trigger disabled 1:Toggle-trigger enabled	When the up-counter value is latched into TB0CP0 0: Toggle-trigger disabled 1:Toggle-trigger enabled	When the up-counter value reaches TB0REG1 0: Toggle-trigger disabled 1: Toggle-trigger enabled	When the up-counter value reaches TB0REG0 0: Toggle-trigger disabled 1:Toggle-trigger enabled	Flip-flop control 00: Toggles TB0OUT (software toggle) 01: Sets TB0OUT to 1 10: Clears TB0OUT to 0 11: Don't care This field is always read as 11.	

TMRB0 Compare Register 0

TB0REG0 (0xFFFF_C70C)

	7	6	5	4	3	2	1	0
Bit Symbol	CMP0							
Read/Write	R/W							
Reset Value	0x00							
Function	When double-buffering is enabled, this register stores the value used for the second comparison.							

	15	14	13	12	11	10	9	8
Bit Symbol	CMP0							
Read/Write	R/W							
Reset Value	0x00							
Function								

Note 1: The TB0CMP0 does not detect a match when TB0REG0=0x0000.

Note 2: To use the INTTBCOM0x interrupt, capture operation must be disabled by setting TB0MOD.CPM=00. When TB0MOD.CPM is set to a value other than 00, no interrupt is generated. However, match detection is performed so that the output on the TB0OUT pin can be toggled.

TMRB0 Compare Register 1

TB0REG1 (0xFFFF_C710)

	7	6	5	4	3	2	1	0
Bit Symbol	CMP1							
Read/Write	R/W							
Reset Value	0x00							
Function	This register stores the value used for comparison.							

	15	14	13	12	11	10	9	8
Bit Symbol	CMP1							
Read/Write	R/W							
Reset Value	0x00							
Function								

Note 1: The TB0CMP1 detects a match even when TB0REG1=0x0000.

Note 2: Match detection by the TB0CMP1 requires setting TB0MOD.CLE=1 or TB0FF.INV1=1.

Not Recommended for New Design

TMRB0 Capture Register 0

TB0CP0 (0xFFFF_C714)

	7	6	5	4	3	2	1	0
Bit Symbol	CP0							
Read/Write	R							
Reset Value	0x00							
Function	Capture value 0 of the up-counter (Low)							

	15	14	13	12	11	10	9	8
Bit Symbol	CP0							
Read/Write	R							
Reset Value	0x00							
Function	Capture value 0 of the up-counter (High)							

TMRB0 Capture Register 1

TB0CP1 (0xFFFF_C718)

	7	6	5	4	3	2	1	0
Bit Symbol	CP1							
Read/Write	R							
Reset Value	0x00							
Function	Capture value 1 of the up-counter (Low)							

	15	14	13	12	11	10	9	8
Bit Symbol	CP1							
Read/Write	R							
Reset Value	0x00							
Function	Capture value 1 of the up-counter (High)							

TMRB0 Counter Register

TB0CNT (0xFFFF_C71C)

	7	6	5	4	3	2	1	0
Bit Symbol	CNT							
Read/Write	R							
Reset Value	0x00							
Function	Count value of the up-counter (Low)							

	15	14	13	12	11	10	9	8
Bit Symbol	CNT							
Read/Write	R							
Reset Value	0x00							
Function	Count value of the up-counter (High)							

Not Recommended for New Design

11.4 Operating Modes

The 16-bit timer has the following operation modes:

(A) 16-Bit Interval Timer mode

(B) 16-Bit Event Counter mode

(C) 16-Bit Programmable Pulse Generation (PPG) mode

The TMRB0 has the capture capability used to latch the value of the counter. The capture capability allows:

(D) Pulse width measurement

(E) One-shot pulse generation using an external trigger pulse

11.4.1 16-Bit Interval Timer Mode

To accomplish periodic interrupt generation, the interval time is set in the TB0REG1 register, and the INTTBCOM01 interrupt is enabled.

Example: Setting the 20 μs interval timer (IMCLK: 28 MHz) using INTTBCOM01

```
1.  TB0RUN = 0x00;                  // Stop timer 0
2.  IMR25 = 0x00;                   // Disable INTTBCOM00
    IMR26 = 0x41;                   // Enable INTTBCOM01
3.  TB0FF = 0x0A;                   // INVC1=1, FF=0
    TB0MOD = 0x010A;                // Select prescaler (IMCLK/2)
    TB0REG1 = 0x0118;               // Set interval time
4.  TB0RUN = 0x0D;                  // Start timer
```

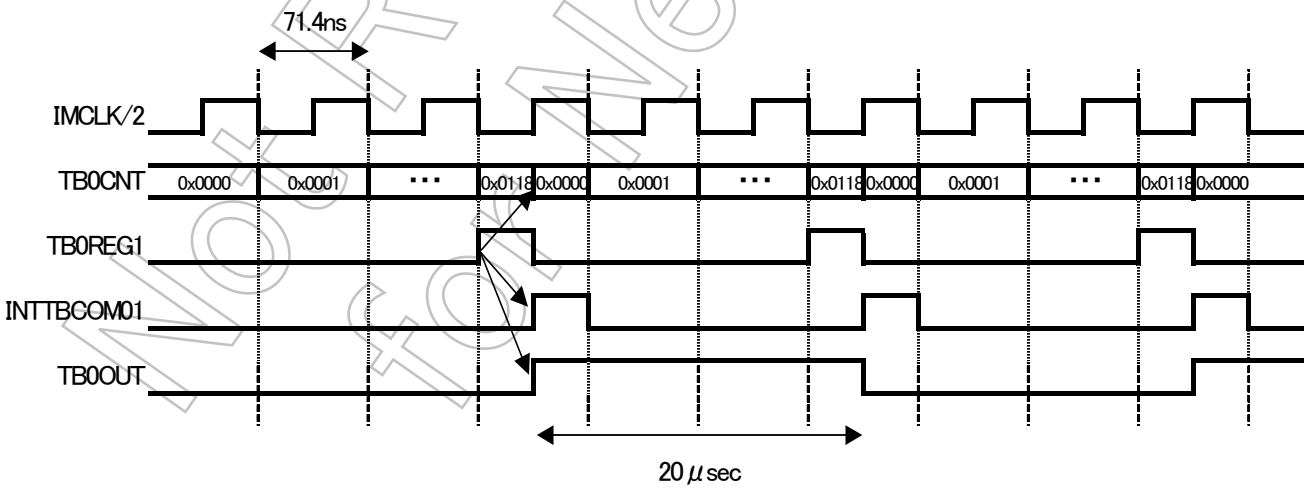


Figure 11.4.1 16-Bit Interval Timer Mode

Not Recommended for New Design

11.4.2 16-Bit Event Counter Mode

This mode is used to count events by interpreting the rising edges of the external counter clock (TB0IN) as events.

The up-counter counts up on each rising edge of the TB0IN pin input. The counter value can be latched into a capture register under software control. To determine the number of events (i.e., cycles) counted, the value in the capture register must be read.

Example: Setting the event counter

```
1.    TB0RUN = 0x00;          // Stop timer 0
2.    IMR84 = 0x41;           // Enable INTTBCAP00
      IMR85 = 0x00;           // Disable INTTBCAP01
3.    TB0FF = 0x03;           // Disable trigger
      TB0MOD = 0x0124;        // Select external time, IMCLK/8
      TB0REG1 = 0x0050;       // Set interval time
4.    TB0RUN = 0x0D;          // Start timer
```

Not Recommended for New Design

11.4.3 16-Bit Programmable Pulse Generation (PPG) Mode

The 16-Bit PPG mode can be used to generate a square wave with any frequency and duty cycle. The pulse can be high-going or low-going, as determined by the initial setting of the timer flip-flop (TB0FF).

A square wave is generated by toggling the timer flip-flop (TB0FF) every time the up-counter (TB0CNT) reaches the value in each timer register (TB0REG0, TB0REG1). The square-wave output is driven to the TB0OUT pin. In this mode, the following relationship must be satisfied:

(TB0REG0 value) < (TB0REG1 value)

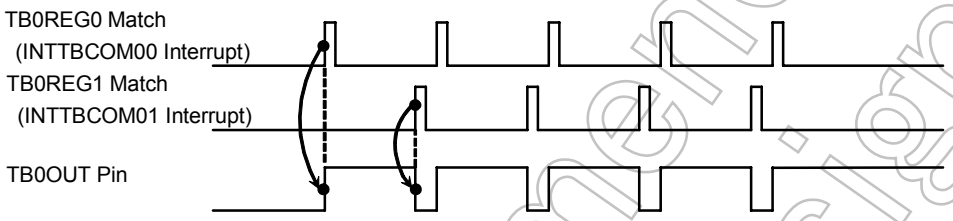


Figure 11.4.2 PPG Output Waveform

If the double-buffering function is enabled, the TB0REG0 value can be changed dynamically by writing a new value into the register buffer. Upon a match between the TB0REG1 and the TB0CNT, the TB0REG0 latches a new value from the register buffer. The TB0REG0 can be loaded with a new value upon every match thus making it easy to generate a square wave with virtually any duty cycle.

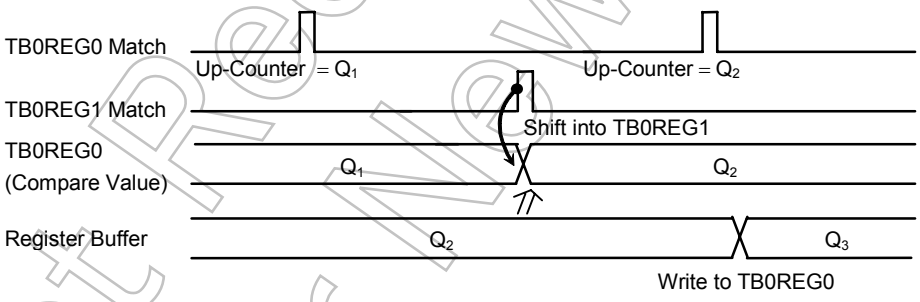


Figure 11.4.3 Register Buffer Operation

Not Recommended for New Design

Example: Setting the event counter with double-buffering

```
1.  TB0RUN = 0x00;                  // Stop timer 0
2.  TB0REG0 = 0x0050;               // Set interval time
    TB0REG1 = 0x0080;
3.  TB0RUN = 0x80;                  // Enable double buffer
4.  TB0FF = 0x0E;                   // Initialize flip-flop
    TB0MOD = 0x010D;                // Select prescaler (IMCLK/16)
5.  P9FR1 = 0x10;                   // P94 TB0OUT
    P9CR = 0x10;                    // P94 output enable
6.  TB0RUN = 0x8D;                  // Start timer
```

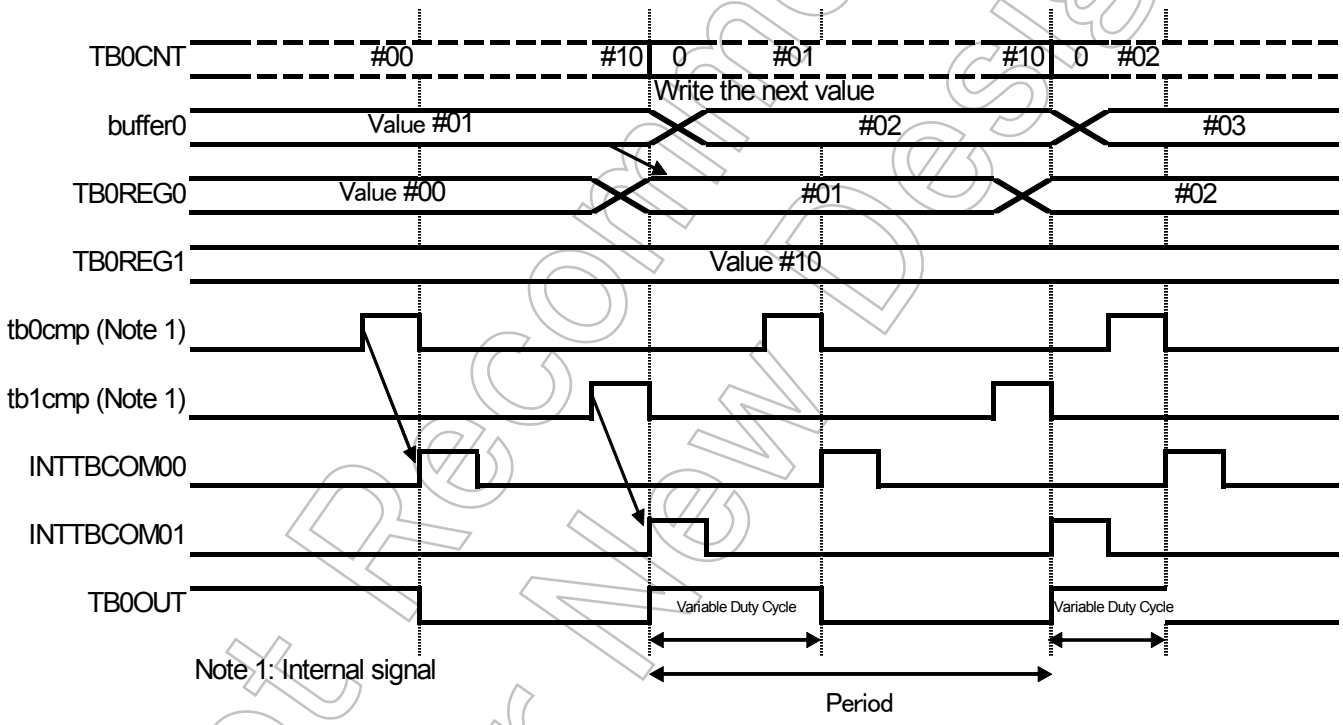


Figure 11.4.4 Programmable Pulse Generation (PPG) Mode

Not Recommended for New Design

11.4.4 Pulse Width Measurement

The capture function can be used to measure the pulse width of an external clock. The external clock is applied to the TB0IN pin. The up-counter (TB0CNT) is programmed to operate as a free-running counter, clocked by one of the prescaler outputs. The capture function is used to latch the TB0CNT value into the capture registers (TB0CP0, TB0CP1) at the clock rising edge and at the next clock falling edge, respectively. The Interrupt Controller (INTC) should be programmed to generate the INTTBCAP00 interrupt at the falling edge of the TB0IN input.

Multiplying the counter clock period by the difference between the values captured into the TB0CP0 and TB0CP1 gives the high pulse width of the TB0IN0 clock.

For example, if the prescaler output clock has a period of 0.5 μs and the difference between the TB0CP0 and TB0CP1 is 100, the high pulse width is calculated as 0.5 μs x 100 = 50 μs.

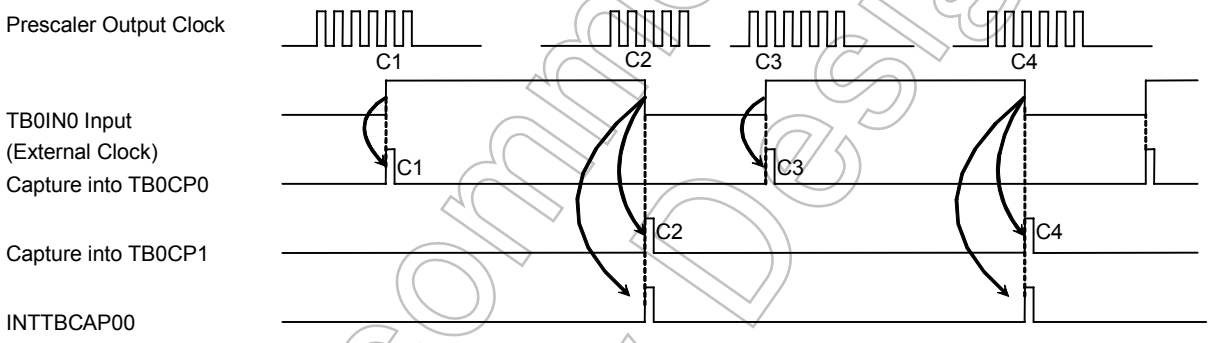


Figure 11.4.5 Pulse Width Measurement

The low pulse width of the external clock can be measured by setting the INTTBCAP01 interrupt to be generated on the rising edge of the TB0IN pin, and multiplying the difference between the TB0CP1 value at C2 and the TB0CP0 value at C3 by the prescaler output clock period. If no edge input occurs on the TB0IN pin, this can be detected by a counter overflow.

Not Recommended for New Design

11.4.5 One-Shot Pulse Generation Using an External Trigger Pulse

The TMRBn can be used to produce a one-time pulse as follows.

(1) The 16-bit up-counter (TB0CNT) is programmed to function as a free-running counter, clocked by one of the prescaler outputs. The TB0IN pin is used as an active-high external trigger pulse input for latching the counter value into the capture register (TB0CP0).

(2) The Interrupt Controller (INTC) must be programmed to generate an INTTBCAP01 interrupt upon detection of a rising edge on the TB0IN pin. The TB0REG0 is loaded with the sum of the TB0CP0 value (c) and the pulse delay (d)—i.e., (c) + (d). The TB0REG1 is loaded with the sum of the TB0REG0 value and the pulse width (p)—i.e., (c) + (d) + (p).

(3) Next, the INVC0 and INVC1 bits in the timer flip-flop control register (TB0FF) are set to 11, so that the timer flip-flop (TB0FF) will toggle when a match is detected between the TB0CNT and the TB0REG0 and between the TB0CNT and the TB0REG1. With the TB0FF toggled twice, a one-shop pulse is produced. Upon a match between the TB0CNT and the TB0REG1, the TMRB0 generates the INTTBCOM01 interrupt, which must disable the toggle trigger for the TB0FF.

Figure 11.4.6 depicts one-shot pulse generation, with annotations showing (c), (d) and (p).

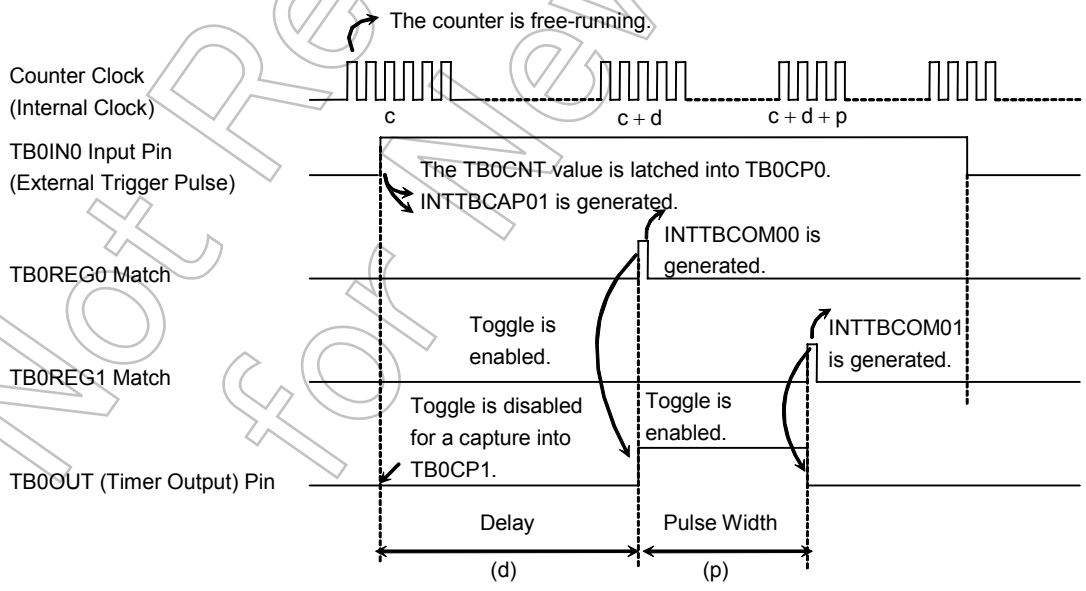


Figure 11.4.6 One-Shop Pulse Generation (with a Delay)

Not Recommended for New Design

Example: Generating a one-shot pulse with a width of 2 ms and a delay of 3 ms on assertion of an external trigger pulse on the TB0IN pin

Clocking conditions:

System clock: 56 MHz

Prescaler clock: IMCLK/2 (IMCLK = fsys/2)

Settings in the main routine

		7	6	5	4	3	2	1	0	
										Place the counter in free-running mode.
TB0MOD	←	–	–	1	0	0	0	1	0	Select IMCLK/2 as the counter clock source.
										Latch TB0CNT value into TB0CP0 at rising edges of the TB0IN input.
TB0FF	←	–	–	0	0	0	0	1	0	Clear TB0FF0 to 0.
										Disable the toggle trigger for TB0FF0.
P9IER	←	–	–	–	1	–	–	–	–	Configure the P94 pin as TB0OUT.
P9CR	←	–	–	–	1	–	–	–	–	
P9FR1	←	–	–	–	1	–	–	–	–	
IMR85	←	X	1	0	0	X	1	0	0	Enable INTTBCAP01 and disable INTTBCOM00.
IMR25	←	X	1	0	0	X	0	0	0	
TB0RUN	←	–	0	X	X	1	1	X	1	Start TMRB0.

Settings in INTTBCAP01

TB0REG0	←	TB0CP0 + 3ms/(IMCLK/2)								
TB0REG1	←	TB0REG0 + 2ms/(IMCLK/2)								
TB0FF	←	–	–	–	–	1	1	1	1	Enable the TB0FF0 toggle trigger for TB0REG0 and TB0REG1 matches.
IMR25	←	X	1	0	0	X	1	0	0	Enable INTTBCOM00.

Settings in INTTBCOM01

TB0FF	←	–	–	–	–	0	0	1	1	Disable the TB0FF0 toggle trigger for TB0REG0 and TB0REG1 matches.
IMR25	←	X	1	0	0	X	0	0	0	Disable INTTBCOM00.

X: Don't care, –: No change

If no delay is necessary, enable the TB0FF toggle trigger for a capture of the TB0CNT value into the TB0CP0. Use the INTTBCAP01 interrupt to load the TB0REG1 with a sum of the TB0CP0 value (c) and the pulse width (p) and to enable the TB0FF toggle trigger for a match between the TB0CNT and TB0REG1 values. A match generates the INTTBCOM1 interrupt, which then is to disable the TB0FF toggle trigger.

Not Recommended for New Design

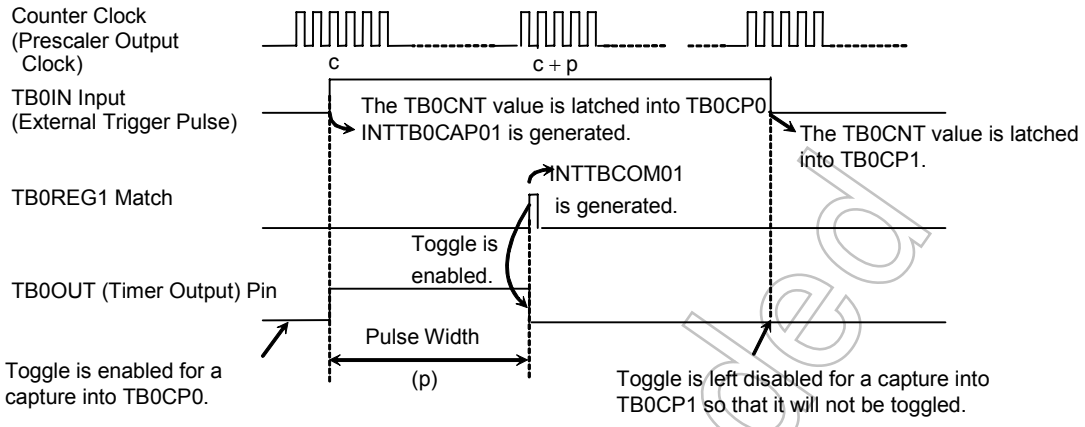


Figure 11.4.7 One-Shot Pulse Generation (without a Delay)

Not Recommended for New Design

11.4.6 One-Shot Pulse Generation Using an External Count Start Trigger

Using an external count start trigger enables one-shot pulse generation with a shorter delay.

(1) The 16-bit up-counter (TB0CNT) is programmed to count up on the rising edge of the TB0IN pin (TB0RUN.TREGSEL=1, TB0RUN.CSSEL=1). The TB0REG0 is loaded with the pulse delay (d), and the TB0REG1 is loaded with the sum of the TB0REG0 value (d) and the pulse width (p)—i.e., (d) + (p).

(2) The TB0CNT is programmed to start counting on the rising edge of the external trigger pulse.

(3) Next, the INVC0 and INVC1 bits in the timer flip-flop control register (TB0FF) are set to 11, so that the timer flip-flop (TB0FF) will toggle when a match is detected between the TB0CNT and the TB0REG0 and between the TB0CNT and the TB0REG1. With the TB0FF toggled twice, a one-shot pulse is produced. Upon a match between the TB0CNT and the TB0REG1, the TMRB0 generates the INTTBCOM01 interrupt, which must disable the toggle trigger for the TB0FF.

Figure 11.4.8 depicts one-shot pulse generation, with annotations showing (d) and (p).

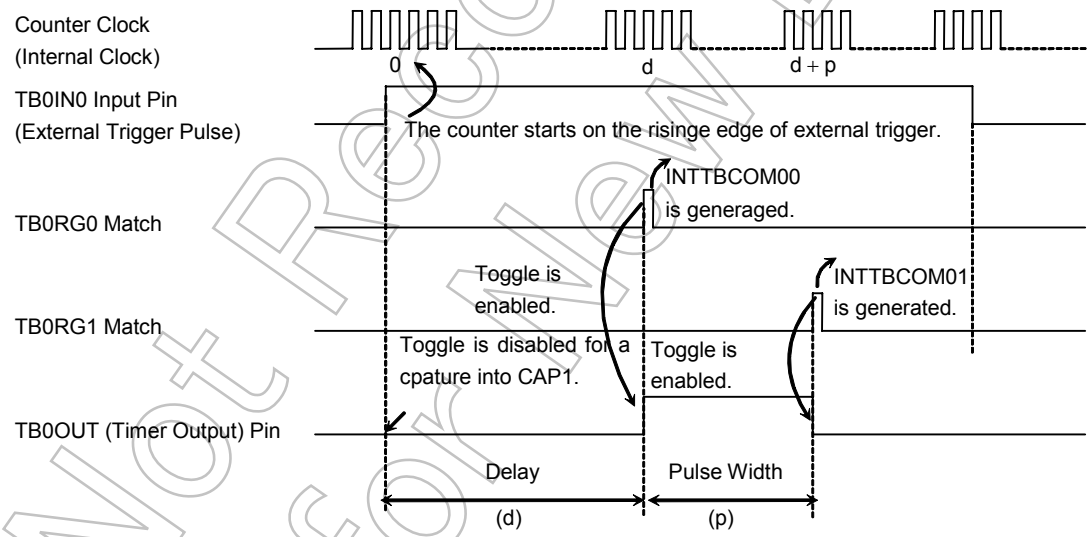


Figure 11.4.8 One-Shot Pulse Generation Using an External Count Start Trigger (with a Delay)

Not Recommended for New Design

12. Serial I/O (SIO)

12.1 Overview

The TMP19A71 contains four channels of serial I/O (SIO0 to SIO3). The SIO2 and SIO3 can be used in UART mode (asynchronous) and I/O Interface mode (synchronous). The SIO0 and SIO1 only support UART mode. The SIO0 and SIO1 do not have the SCLK and CTS pins; thus an external clock cannot be used as a UART transfer clock in these channels.

- I/O Interface mode —— Mode 0: Transmits/receives a serial clock (SCLK) as well as data streams for a synchronous clock mode of operation
- UART mode
  - Mode 1: 7 data bits
  - Mode 2: 8 data bits
  - Mode 3: 9 data bits

In Mode 1 and Mode 2, each frame can include a parity bit. In Mode 3, the wake-up feature is available for multidrop applications in which a master station is connected to several slave stations through a serial link. Figure 12.2.1 shows a block diagram of the SIO2.

The main components of an SIO channel are a clock prescaler, a serial clock generator, a receive buffer, a receive controller, a transmit buffer and a transmit controller. Each SIO channel is independently programmable and functionally equivalent. In the following sections, any references to the SIO2 also apply to the other channels unless otherwise noted.

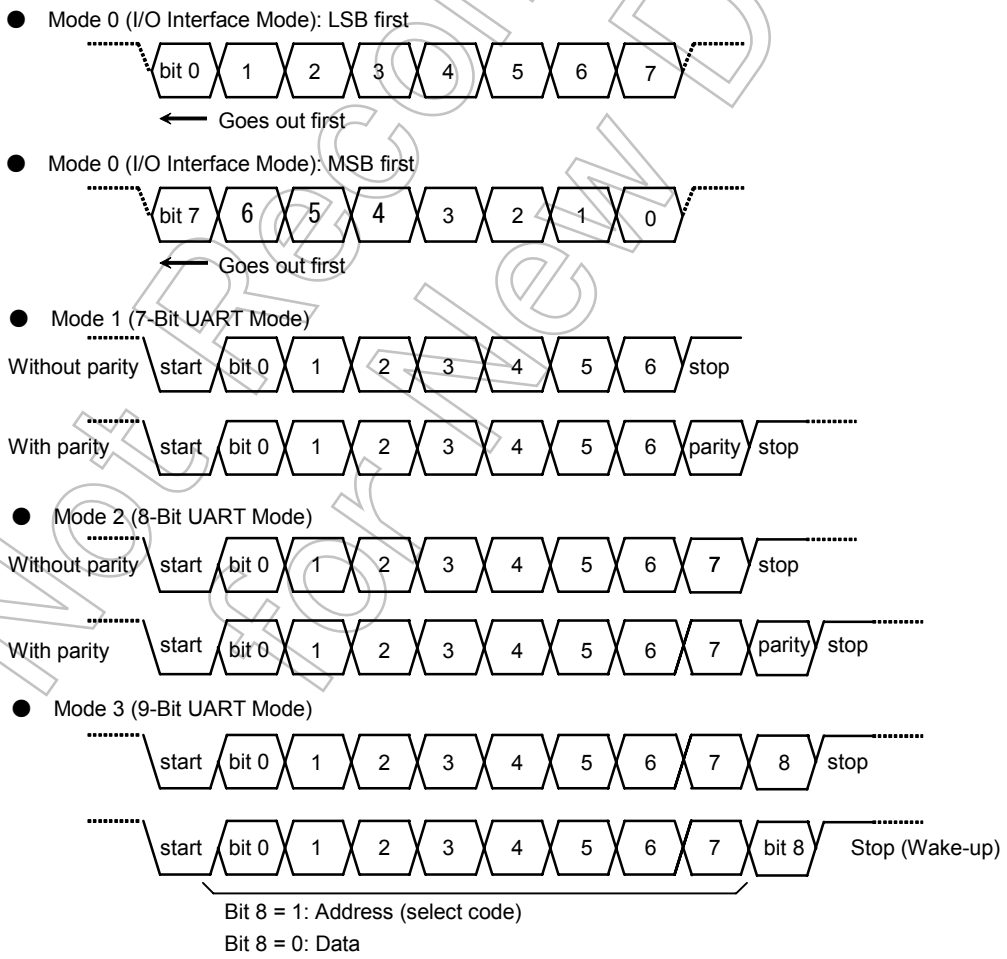


Figure 12.1.1 Data Formats

Not Recommended for New Design

12.2 Block Diagram (SIO2)

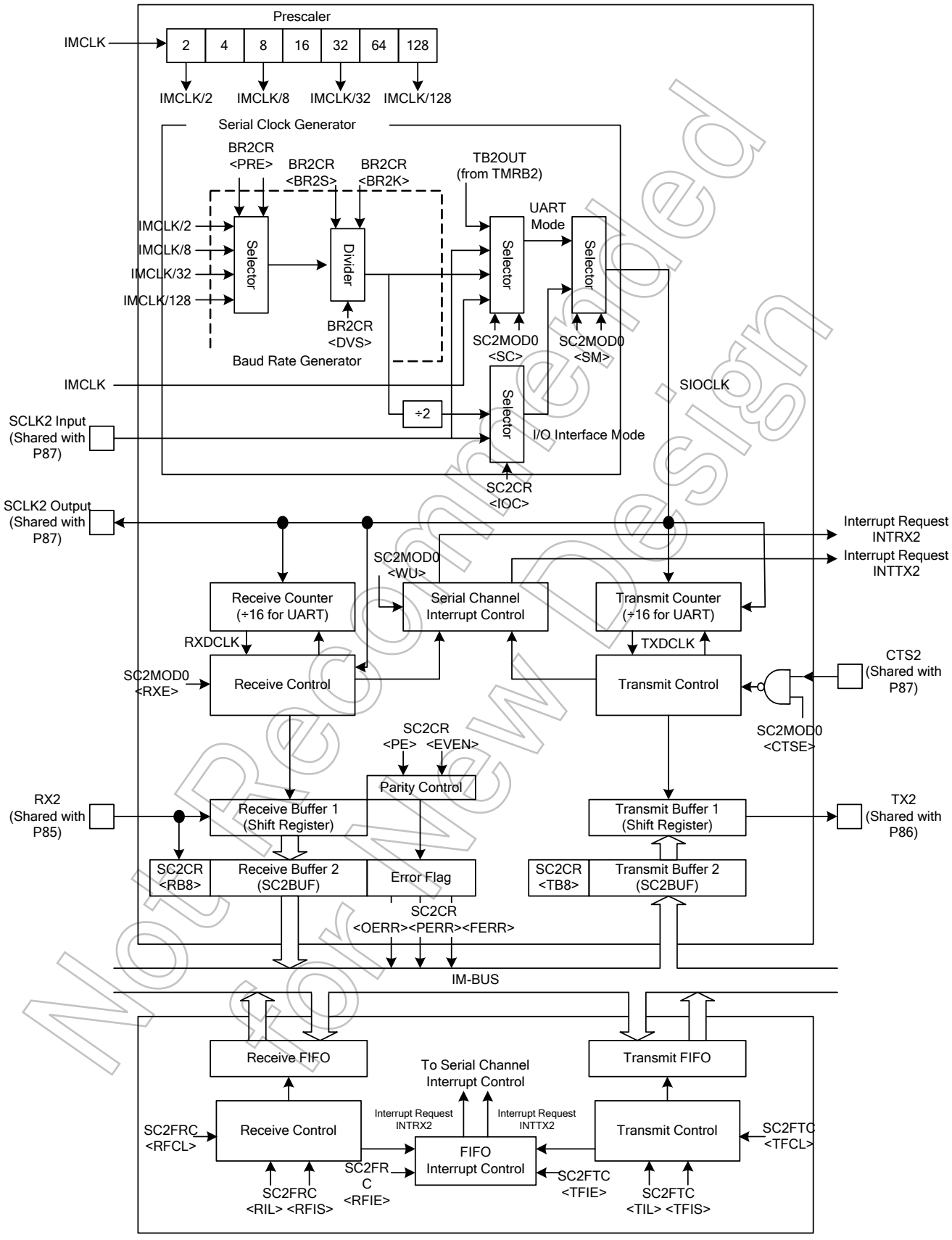


Figure 12.2.1 SIO2 Block Diagram

Not Recommended for New Design

12.3 SIO Components (SIO2)

12.3.1 Prescaler

The SIO2 has a 7-bit prescaler that slows the rate of a clocking source to the serial clock generator. The prescaler clock source (IMCLK) can be programmed in the PRS2 bit of the CLKPRSC located within the clock generator.

The prescaler can output four types of clocks to the baud rate generator: IMCLK/2, IMCLK/8、IMCLK/32 and IMCLK/128.

The serial clock is selectable from several clocks; the prescaler is only enabled when the baud rate generator output clock is selected as a serial clock. Table 12.3.1 shows prescaler output clock resolutions.

Table 12.3.1 Prescaler Output Clock Resolutions

fc = 12 MHz (PLL output clock)

Clock Gear Value CLKPRSC.PRS1	IMCLK Selection CLKPSC.PRS2	Prescaler Output Clock Resolution			
		IMCLK/2	IMCLK/8	IMCLK/32	IMCLK/128
00 (fc/2)	000 (fsys/2)	fc/8 (71.4 ns)	fc/32 (0.29 μs)	fc/128 (1.1 μs)	fc/512 (4.6 μs)
	010 (fsys/3)	fc/12 (107 ns)	fc/48 (0.43 μs)	fc/192 (1.7 μs)	fc/768 (6.9 μs)
	100 (fsys/4)	fc/16 (143 ns)	fc/64 (0.57 μs)	fc/256 (2.3 μs)	fc/1024 (9.1 μs)
	110 (fsys/5)	fc/20 (178 ns)	fc/80 (0.71 μs)	fc/320 (2.9 μs)	fc/1280 (11.4 μs)
01 (fc/4)	000 (fsys/2)	fc/16 (143 ns)	fc/64 (0.57 μs)	fc/256 (2.3 μs)	fc/1024 (9.1 μs)
	010 (fsys/3)	fc/24 (187 ns)	fc/96 (0.86 μs)	fc/384 (3.4 μs)	fc/1524 (13.7 μs)
	100 (fsys/4)	fc/32 (286 ns)	fc/128 (1.1 μs)	fc/512 (4.6 μs)	fc/2048 (18.3 μs)
	110 (fsys/5)	fc/40 (357 ns)	fc/160 (1.43 μs)	fc/640 (5.7 μs)	fc/2560 (22.9 μs)
10 (fc/8)	000 (fsys/2)	fc/32 (0.29 μs)	fc/128 (1.1 μs)	fc/512 (4.6 μs)	fc/2048 (18.3 μs)
	010 (fsys/3)	fc/48 (0.43 μs)	fc/192 (1.7 μs)	fc/768 (6.9 μs)	fc/3048 (17.4 μs)
	100 (fsys/4)	fc/64 (0.57 μs)	fc/256 (2.3 μs)	fc/1024 (9.1 μs)	fc/4096 (36.6 μs)
	110 (fsys/5)	fc/80 (0.71 μs)	fc/320 (2.9 μs)	fc/1280 (11.4 μs)	fc/5120 (45.8 μs)

Note: Do not change the clock gear value while the SIO is operating.

Not Recommended for New Design

12.3.2 Baud Rate Generator

The frequency used to transmit and receive data through the SIO2 is derived from the baud rate generator. The clock source for the baud rate generator can be selected from the 7-bit prescaler outputs (IMCLK/2, IMCLK/8, IMCLK/32, IMCLK/128) through the programming of the PRE bit in the BR2CR.

The baud rate generator contains a clock divider that can divide the selected clock by N (N = 1 to 16) or N+(16-K)/16 (N = 2 to 15, K = 1 to 15). The clock divisor is programmed into the DVS and BR2S bits in the BR2CR and the BR2K bit in the BR2ADD.

- I/O Interface mode

I/O Interface mode cannot utilize the N + (16 – K)/16 clock division function. The DVS bit in the BR2CR must be cleared to 0.

- UART mode

1) When BR2CR.DVS=0

When the BR2CR.DVS bit is cleared, the BR2ADD.BR2K field has no meaning or effect. In this case, the baud rate generator input clock is divided down by a value of N (1 to 16) programmed in the BR2CR.BR2S field.

2) When BR2CR.DVS=1

Setting the BR2CR.DVS bit to 1 enables the N + (16 – K)/16 division function. The baud rate generator input clock is divided down according to the value of N (2 to 15) programmed in the BR2CR.BR2S field and the value of K (1 to 15) programmed in the BR2ADD.BR2K field.

Note: Setting N to 1 or 16 disables the N + (16−K)/16 clock division function. When N = 1 or 16, the BR2CR.DVS bit must be cleared to 0.

- Baud rate calculations

1) I/O Interface mode

$$\text{Baud rate} = \frac{\text{Baud rate generator input clock}}{\text{Baud rate generator divisor}} \div 2$$

When the clock input to the baud rate benerator is IMCLK/2 (14 MHz) and the baud rate generator divisor is set to 2, the maximum baud rate is 3.5 Mbps.

Not Recommended for New Design

2) UART mode

$$\text{Baud rate} = \frac{\text{Baud rate generator input clock}}{\text{Baud rate generator divisor}} \div 16$$

When the clock input to the baud rate generator is IMCLK/2 (14 MHz), the maximum baud rate is 875 Kbps.

The baud rate generator can be bypassed if the user wants to use the IMCLK clock as a serial clock. In this case, the maximum baud rate is 1.75 Mbps (at IMCLK = 28 MHz).

- Calculation examples

1) Integral division (divide-by-N)

IMCLK = 28 MHz

Baud rate generator input clock: IMCLK/8

Clock divisor N (BR2CR.BR2S) = 4

BR2CR.DVS = 0

Clocking conditions { System clock : 56 MHz; IMCLK : 28 MHz (divide-by-2) }

$$\text{Baud rate} = \frac{\text{IMCLK}/8}{4} \div 16$$

$= 28 \times 10^6 \div 8 \div 4 \div 16 = 54.7$ (Kbps)

Note: Clearing the BR2CR.DVS bit to 0 disables the N + (16 - K)/16 clock division function. At this time, the BR2ADD.BR2K field is ignored.

2) N + (16 - K)/16 clock division (UART mode only)

IMCLK = 28 MHz

Baud rate generator input clock: IMCLK/32

N (BR2CR.BR2S) = 5

K (BR2ADD.BR2K) = 5

BR2CR.DVS = 1

Clocking conditions { System clock : 56 MHz; IMCLK : 28 MHz (divide-by-2) }

$$\text{Baud rate} = \frac{\text{IMCLK}/32}{5 + \frac{(16-5)}{16}} \div 16$$

$$= 28 \times 10^6 \div 32 \div \left(5 + \frac{11}{16}\right) \div 16 = 9615 \text{ (bps)}$$

Not Recommended for New Design

The SIO2 can use an external clock as a serial clock, bypassing the baud rate generator. When an external clock is used, the baud rate is determined as shown below.

- Using an external clock as a serial clock

1) I/O Interface mode

Baud rate = external clock input

When double-buffering is used, the external clock period must be greater than 12/fsys. Therefore, when fsys = 56 MHz, the maximum baud rate is 4.7 Mbps (56 ÷ 12).

When double-buffering is not used, the exernal clock period must be greater than 16/fsys. Therefore, when fsys = 56 MHz, the maximum baud rate is 3.5 Mbps (56 ÷ 16).

2) UART mode

Baud rate = external clock input ÷ 16

The external clock input must be greater than or equal to 4/fsys. Therefore, when fsys = 56 MHz, the maximum baud rate is 875 Kbps (56 ÷ 4 ÷ 16).

Table 12.3.2 and Table 12.3.3 show baud rate setting examples in UART mode.

Not Recommended for New Design

Table 12.3.2 UART Baud Rate Selection

Logic Baud Rate (bps)	Generated Baud Rate (bps)	Prescaler	Divisor N	Correction Value K	Error (%)
1200	1202	IMCLK/128	11	10	0.16
2400	2404	IMCLK/128	5	5	0.16
4800	4808	IMCLK/32	11	10	0.16
9600	9615	IMCLK/32	5	5	0.16
14400	14403	IMCLK/8	15	13	0.02
19200	19231	IMCLK/8	11	10	0.16
28800	28689	IMCLK/8	7	6	0.39
31250	31250	IMCLK/8	7	None	0
38400	38462	IMCLK/8	5	5	0.16
57600	57613	IMCLK/2	15	13	0.02
115200	115702	IMCLK/2	7	7	0.44
230400	229508	IMCLK/2	3	3	0.39

Note 1: This table assumes: fsys = 56 MHz, IMCLK = fsys/2 (28 MHz).

Note 2: When a baud rate slower than 600 bps is used, the input clock must be TMRB2.

Table 12.3.3 UART Baud Rate Selection

TB2REG1 values when the TMRB2 timer trigger output (Internal TB2OUT) is used

(TMRB2 input clock = IMCLK/4)

IMCLK / Baud rate (bps)	28 MHz	20 MHz	14 MHz	10 MHz	7 MHz
100	4375	3125	2188	1563	1094
150	2916	2084	1458	1042	730
200	2188	1563	1094	781	547
300	1458	1042	729	521	365
400	1094	781	547	391	273
500	875	625	438	313	219
600	729	521	365	260	182

When the timer TMRB2 is used to generate a serial clock, the baud rate is determined by the following equation:

$$\text{Baud rate} = \frac{\text{IMCLK}}{\text{TB2REG1} \times \underline{4} \times 16}$$

(When the TMRB2 clock source is IMCLK/4)

Note: In I/O Interface mode, the SIO2 and SIO3 cannot utilize the trigger output signal (internal) from the timer TMRB2 as a serial clock.

Not Recommended for New Design

12.3.3 Serial Clock Generator

This block generates a basic clock that controls the transmit and receive operations.

- I/O Interface mode

  If the SCLK2 pin is configured as an output by clearing the SC2CR.IOC bit to 0, the output clock from the baud rate generator is divided by two to generate the basic clock.

  If the SCLK2 pin is configured as an input by setting the SC2CR.IOC bit to 1, the external SCLK2 clock is used as the basic clock; the SC2CR.SCLKS bit determines the active clock edge.

- UART mode

  The basic clock (SIOCLK) is selected from a clock produced by the baud rate generator, the system clock (IMCLK/2), the internal output signal from the timer TMRB2, and the external SCLK2 clock, according to the setting of the SC2MOD0.SC field.

12.3.4 Receive Counter

The receive counter is a 4-bit binary up-counter used in UART mode. This counter is clocked by SIOCLK. The receiver utilizes 16 clocks for each received bit, and oversamples each bit three times around their center (with 7th to 9th clocks). The value of a bit is determined by voting logic which takes the value of the majority of three samples.

12.3.5 Receive Controller

- I/O Interface mode

  If the SCLK2 pin is configured as an output by clearing the SC2CR.IOC bit to 0, the receive controller samples the RX2 input at the rising edge of the shift clock driven out from the SCLK2 pin.

  If the SCLK2 pin is configured as an input by setting the SC2CR.IOC bit to 1, the receive controller samples the RX2 input at either the rising or falling edge of the SCLK2 clock, as programmed in the SC2CR.SCLKS bit.

- UART mode

  The receive controller uses 16 clocks for receiving the start bit. It samples the 7th to 9th clocks to determine by voting logic whether or not the correct start bit is received. Receive operation is started upon reception of the correct start bit.

12.3.6 Receive Buffer

The receive buffer is double-buffered to prevent overrun errors. Received data is serially shifted bit by bit into Receive Buffer 1. When a whole frame is loaded into Receive Buffer 1, it is transferred to Receive Buffer 2 (SC2BUF), and the INTRX2 is generated. At this time, the Receive Buffer Full flag (SC2MOD2.RBFLL) is set to 1, indicatig that Receive Buffer 2 contains valid data.

The TX19A core processor reads a frame from Receive Buffer 2 (SC2BUF), causing the Receive Buffer Full flag (SC2MOD2.RBFLL) to be cleared to 0. Receive Buffer 1

Not Recommended for New Design

can accept a new frame before the TX19A core processor picks up the previous frame in Receive Buffer 2 (SC2BUF).

If the SCLK2 pin is configured as an output in I/O Interface mode, Receive Buffer 2 (SC2BUF) can be enabled or disabled by programming the WBUF bit in the SC2MOD2. Disabling Receive Buffer 2 (double-buffering) enables handshaking during data transfer; the SIO2 stops outputting the SCLK2 clock every time a single frame has been transmitted. In this case, the TX19A core processor reads a frame from Receive Buffer 1, causing the output of the SCLK2 clock to be restarted. If Receive Buffer 2 (double-buffering) is enabled, a received frame is transferred from Receive Buffer 1 to Receive Buffer 2. Once a next frame is received resulting in both Receive Buffers 1 and 2 containing valid data, the SIO2 stops outputting the SCLK2 clock. When the TX19A core processor reads a frame from Receive Buffer 2, the frame stored in Receive Buffer 1 is transferred to Receive Buffer 2, causing a receive-done interrupt (INTRX2) to occur and the SIO2 to restart outputting the SCLK2 clock. Consequently, no overrun error occurs if the SCLK2 pin is configured as an output in I/O Interface mode, regardless of the setting of the SC2MOD2.WBUF bit.

Note: In SCLK output mode, the OEER flag in the SC2CR has no meaning; it is read as undefined. When exiting SCLK output mode, first read the SC2CR to initialize this flag.

In other operating modes, Receive Buffer 2 is always enabled to improve performance during continuous transfer. However, the TX19A core processor must read Receive Buffer 2 (SC2BUF) before Receive Buffer 1 is filled with a new frame. Otherwise, an overrun error occurs, causing the frame previously stored in Receive Buffer 1 to be lost. Even in that case, the contents of Receive Buffer 2 and the SC2CR.RB8 bit are preserved.

The SC2CR.RB8 bit holds the parity bit in 8-Bit UART mode and the most significant bit in 9-Bit UART mode.

In 9-Bit UART mode, the receiver wake-up feature can be enabled for slave controllers by setting the SC2MOD0.WU bit to 1. The receiver generates the INTRX2 interrupt only when the SC2CR.RB8 bit is set to 1.

Not Recommended for New Design

12.3.7 Transmit Counter

The transmit counter is a 4-bit binary up-counter used in UART mode. Like the receive counter, the transmit counter is also clocked by SIOCLK. The transmitter generates a transmit clock (TXDCLK) pulse every 16 SIOCLK pulses.

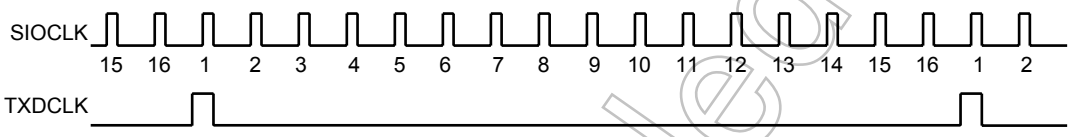


Figure 12.3.1 Transmit Clock Generation

12.3.8 Transmit Controller

- I/O Interface mode

  If the SCLK2 pin is configured as an output by clearing the SC2CR.IOC bit to 0, the transmit controller shifts out each bit in the transmit buffer to the TX2 pin at the falling edge of the shift clock driven out on the SCLK2 pin.

  If the SCLK2 pin is configured as an input by setting the SC2CR.IOC bit to 1, the transmit controller shifts out each bit in the transmit buffer to the TX2 pin at either the rising or falling edge of the SCLK2 input, as programmed in the SC2CR.SCLKS bit.

- UART mode

  Once the TX19A core processor loads a frame into the transmit buffer, the transmit controller begins transmission at the next falling edge of TXDCLK, producing a transmit shift clock.

Not Recommended for New Design

Handshaking (SIO2 and SIO3 only)

The SIO2 has a clear-to-send (CTS2) pin. If the CTS operation is enabled, a frame can be transmitted only when the CTS2 input is low. This feature can be used for flow control to prevent overrun errors in the receiver. The SC2MOD0.CTSE bit enables and disables the CTS operation.

If the CTS2 pin goes high in the middle of a transmission, the transmit controller stops transmission upon completion of the current frame until CTS2 goes low again. If so enabled, the transmit controller generates the INTTX2 interrupt to notify the TX19A core processor that the transmit buffer is empty. After the next frame is loaded into the transmit buffer, the transmit controller remains in an idle state until it detects CTS2 going low.

Although the SIO2 does not have the $\overline{\text{RTS}}$ pin, any general-purpose port pins can serve as the $\overline{\text{RTS}}$ pin. The receiving device uses the $\overline{\text{RTS}}$ output to control the CTS2 input of the transmitting device. Once the receiving device has received a frame, $\overline{\text{RTS}}$ should be set to high in the receive-done interrupt handler to temporarily stop the transmitting device from sending the next frame. This way, the user can easily implement a two-way handshake protocol.

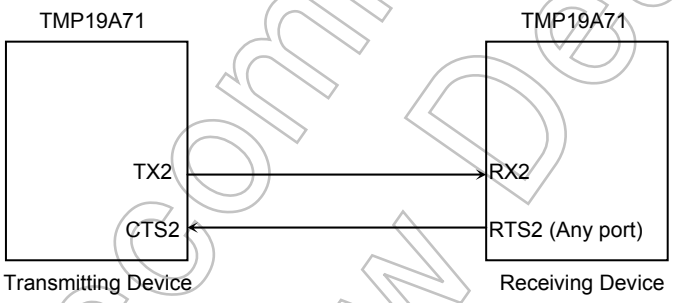
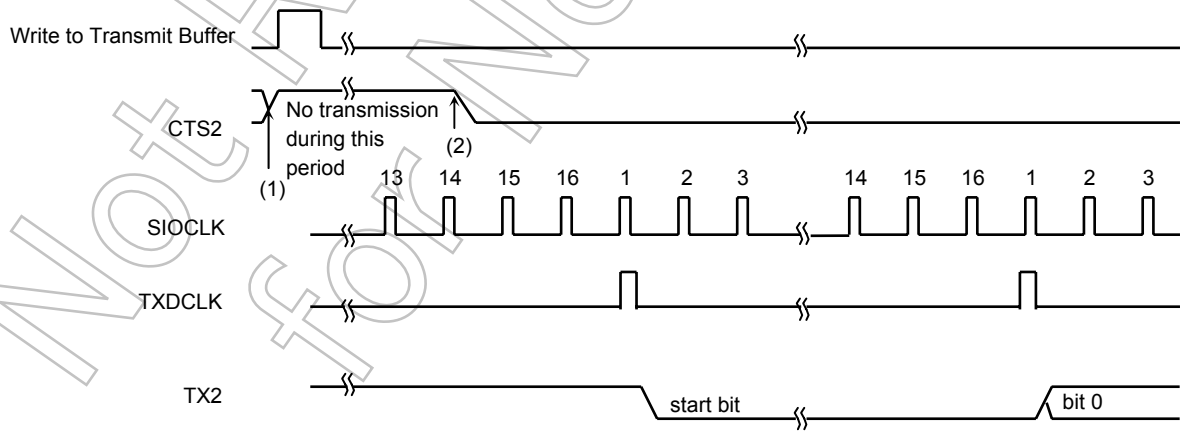


Figure 12.3.2 Handshaking Signals



Note 1: If the CTS2 signal goes high in the middle of a transmission, the transmitter sotps transmission after the current frame has been sent.

Note 2: The transmitter starts transmission at the first falling edge of the TXDCLK clock after the CTS2 signal goes low.

Figure 12.3.3 Clear-To-Send (CTS) Signal Timing

Not Recommended for New Design

12.3.9 Generating a Waveform with a 50% Duty Cycle

When the UART bit in the SC2MOD1 is set to 1, the UART output and the internal transmit signal are ORed, as shown in Figure 12.3.4. When the baud rate generator divisor is set to a value of N in UART mode, a waveform with a 50% duty cycle is generated. The duty ratio varies when the N+ (16-K)/16 clock division function is used.

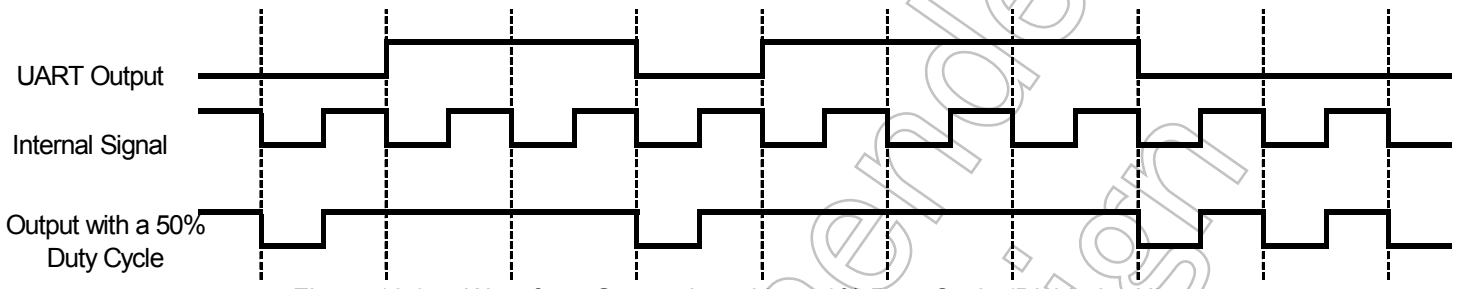


Figure 12.3.4 Waveform Generation with a 50% Duty Cycle (Divide-by-N)

12.3.10 Accuracy of Waveform Generation with a 50% Duty Cycle

(A) When the baud rate generator divisor is set to a value of N

A waveform with a 50% duty cycle is generated.

(B) When the N + (16 - K)/16 clock division function is used

The duty ratio is calculated as the ratio of low width to high width as shown below.

K = 0 to 8 : (K × N) + (8 - K) × (N + 1): 8 × (N + 1)

K = 8 to 16 : 8 × N: (K - 8) × N + (16 - K) × (N + 1)

The largest deviation occurs when K = 8 and N = 1. In this case, the ratio of low width to high width is 8:16 (33%:67%).

Example: Generating 9600 bps by using the N + (16 – K)/16 clock division function

System clock : fsys = 56 MHz (IMCLK = 28 MHz)

Input clock : IMCLK/32 = 875 KHz

Baud rate : 9615 bps (N = 5, K = 5)

Duty ratio : Low:High = 43:48 = 47.25%:52.75%

Not Recommended for New Design

12.3.11 Transmit Buffer

The transmit buffer is double-buffered. Double-buffering can be enabled or disabled by programming the WBUF bit in the SC2MOD2. If double-buffering is enabled, a frame is first written to Transmit Buffer 2 (SC2BUF) and then transferred to Transmit Buffer 1 (shift register), causing the INTTX2 interrupt to occur and the Transmit Buffer Empty flag (SC2MOD2.TBEMP) to be set. This flag indicates that Transmit Buffer 2 is empty and a next transmit frame can be written. Writing a next frame to Transmit Buffer 2 clears the TBEMP flag.

When the SCLK2 pin is configured as an input in I/O Interface mode, an underrun error occurs upon completion of transmitting a frame from Transmit Buffer 1 if a next frame is not written to Transmit Buffer 2 before the clock pulse for the next frame is input. An underrun error is indicated by the parity/underrun flag (PERR) in the SC2CR. When the SCLK2 pin is configured as an output in I/O Interface mode, the SIO2 stops outputting the SCLK2 clock after transmitting a frame which has been transferred from Transmit Buffer 2 to Transmit Buffer 1. In this mode, therefore, no underrun error occurs.

Note: When the SCLK2 pin is configured as an output in I/O Interface mode, the PERR flag in the SC2CR has no meaning; it is read as undefined. When exiting SCLK output mode, first read the SC2CR to initialize this flag.

If double-buffering is disabled, the TX19A core processor writes a transmit frame to Transmit Buffer 1. The INTTX2 interrupt is generated upon completion of transmission.

If handshaking is required, Transmit Buffer 2 must be disabled by clearing the WBUF bit in the SC2MOD2. For continuous transmission without handshaking, Transmit Buffer 2 can be enabled by setting the WBUF bit to improve performance. When double-buffering is not used, do not write to Transmit Buffer 1 while a frame is being transmitted.

12.3.12 Parity Controller

For transmit operations, setting the SC2CR.PE bit to 1 enables parity generation in 7- and 8-Bit UART modes. The SC2CR.EVEN bit selects either even or odd parity.

If enabled, the parity controller automatically generates parity for the frame in the transmit buffer (SC2BUF). In 7-Bit UART mode, the TB7 bit in the SC2BUF holds the parity bit. In 8-Bit UART mode, the TB8 bit in the SC2MOD holds the parity bit. The parity bit is set after the frame has been transmitted. The SC2CR.PE and SC2CR.EVEN bits must be programmed prior to a write to the transmit buffer.

For receive operations, the parity controller automatically computes the expected parity when a frame in Receive Buffer 1 is transferred to Receive Buffer 2 (SC2BUF). The received parity bit is compared to the SC2BUF.RB7 bit in 7-Bit UART mode and to the SC2CR.RB8 bit in 8-Bit UART mode. If a frame is received with incorrect parity, the SC2CR.PERR bit is set.

In I/O Interface mode, the SC2CR.PERR bit indicates an underrun error rather than a parity error.

Not Recommended for New Design

12.3.13 Error Flags

The SIO2 has the following three error flags for improved data reception reliability.

1. Overrun error: SC2CR.OERR

   In UART and I/O Interface mode, an overrun error is reported with the OERR bit set to 1 if all bits of a new frame are received before the current frame is read from the receive buffer. Reading the flag causes it to be cleared. Note that an overrun error can only be cleared by reading the receive buffer or executing a software reset using the SC2MOD2.SWRST.

   When the SCLK2 pin is configured as an output in I/O Interface mode, however, no overrun error occurs so that the OERR flag has no meaning and is read as undefined.

2. Parity error/underrun error: SC2CR.PERR

   In UART mode, this flag indicates whether a parity error has occurred. A parity error is reported when the parity bit attached to a received frame does not match the expected parity computed from the frame. Reading the flag causes it to be cleared.

   In I/O Interface mode, this flag indicates whether an underrun error has occurred, only when double-buffering (Transmit Buffer 2) is enabled (SC2MOD2.WBUF = 1) with the SCLK2 pin configured as an input. An underrun error is reported upon completion of transmitting a frame from Transmit Buffer 1 if a next frame is not written to Transmit Buffer 2 before the clock pulse for the next frame is input. When the SCLK2 pin is configured as an output, no underrun error occurs so that the PERR flag nas no meaning and is read as undefined. Reading the flag causes it to be cleared.

3. Framing error: SC2CR.FERR

   In UART mode, this flag indicates whether a framing error has occurred. A framing error is reported if a 0 is detected where a stop bit was expected. (The middle three of the 16 samples are used to determine the bit value.) Reading the flag causes it to be cleared. During reception, only a single stop bit is detected regardless of the setting of the SBLEN bit in the SC2MOD2.

Not Recommended for New Design

Table 12.3.4 Error Flags

Operating Mode	Error Flag	Function
UART	OERR	Overrun error flag
	PERR	Parity error flag
	FERR	Framing error flag
I/O Interface (SCLK Input)	OERR	Overrun error flag
	PERR	Underrun error flag (WBUF=1)
		Fixed to 0 (WBUF=0)
	FERR	Fixed to 0
I/O Interface (SCLK Output)	OERR	Undefined
	PERR	Undefined
	FERR	Fixed to 0

12.3.14 Bit Transfer Sequence

The DRCHG bit in Serial Mode Control Register 2 (SC2MOD2) determines whether the most significant bit (MSB) or least significant bit (LSB) is transmitted first in I/O Interface mode. The setting of the DRCHG bit cannot be modified while the SIO is transferring data.

12.3.15 Stop Bit Length

The SBLEN bit in the SC2MOD2 determines the number of stop bits (1 or 2) used in UART mode.

Not Recommended for New Design

12.3.16 Status Flag

The RBFLL bit in the SC2MOD2 indicates whether Receive Buffer 2 is full when double-buffering is enabled (SC2MOD2.WBUF = 1). It is set to 1 once a received frame is transferred from Receive Buffer 1 to Receive Buffer 2. The RBFLL bit is cleared to 0 when the TX19A core processor or DMAC reads data from Receive Buffer 2. When WBUF = 0, the RBFLL bit has no meaning; it should not be used as a status flag. The TBEMP bit in the SC2MOD2 indicates whether Transmit Buffer 2 is empty when double-buffering is enabled (SC2MOD2.WBUF = 1). It is set to 1 once a transmit frame is transferred from Transmit Buffer 2 to Transmit Buffer 1 (shift register). The TBEMP bit is cleared to 0 when the TX19A core processor or DMAC stores data in Transmit Buffer 2. When WBUF = 0, the TBEMP bit has no meaning; it should not be used as a status flag.

12.3.17 Transmit/Receive Buffer Configuration

Table 12.3.5 Transmit/Receive Buffer Configuration

		WBUF = 0	WBUF = 1
UART	Transmit	Single	Double
	Receive	Double	Double
I/O Interface (SCLK Input)	Transmit	Single	Double
	Receive	Double	Double
I/O Interface (SCLK Output)	Transmit	Single	Double
	Receive	Single	Double

Not Recommended for New Design

12.3.18 Transmit/Receive FIFO Buffers

As shown in Figure 12.3.5 and Figure 12.3.6, a total of 16 bytes of FIFO buffer is available both in UART mode (excluding 9-Bit UART mode) and I/O Interface mode. When the FIFO buffer is used for both transmit and receive operations, 8 bytes are assigned to each. When the FIFO buffer is required for only transmit or receive, all the 16 bytes can be used as transmit or receive buffers.

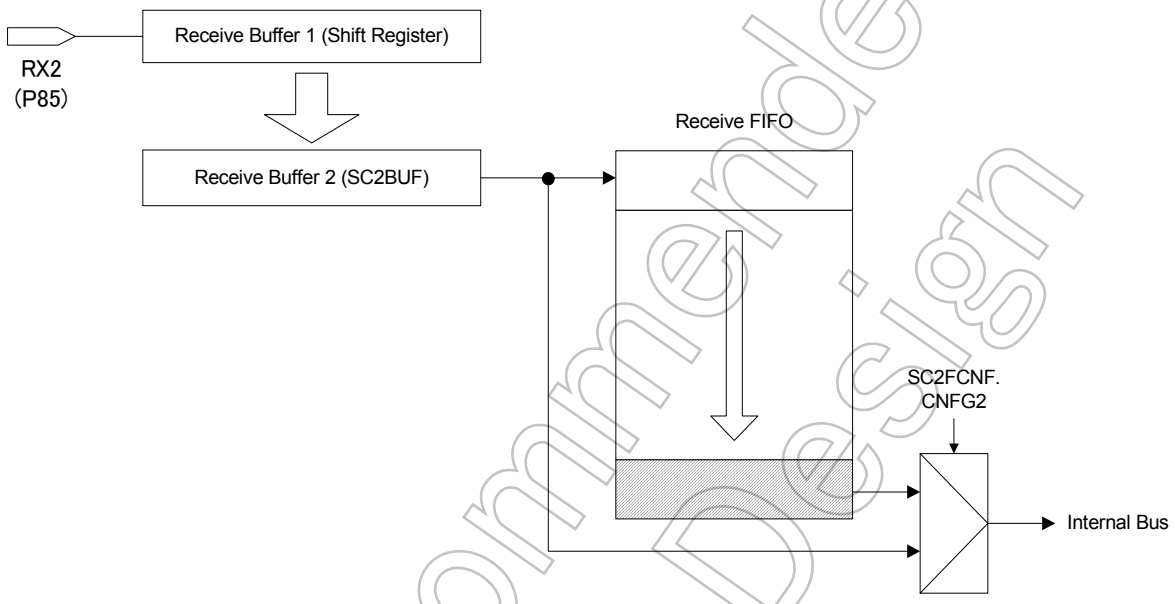


Figure 12.3.5 Receive FIFO Block Diagram

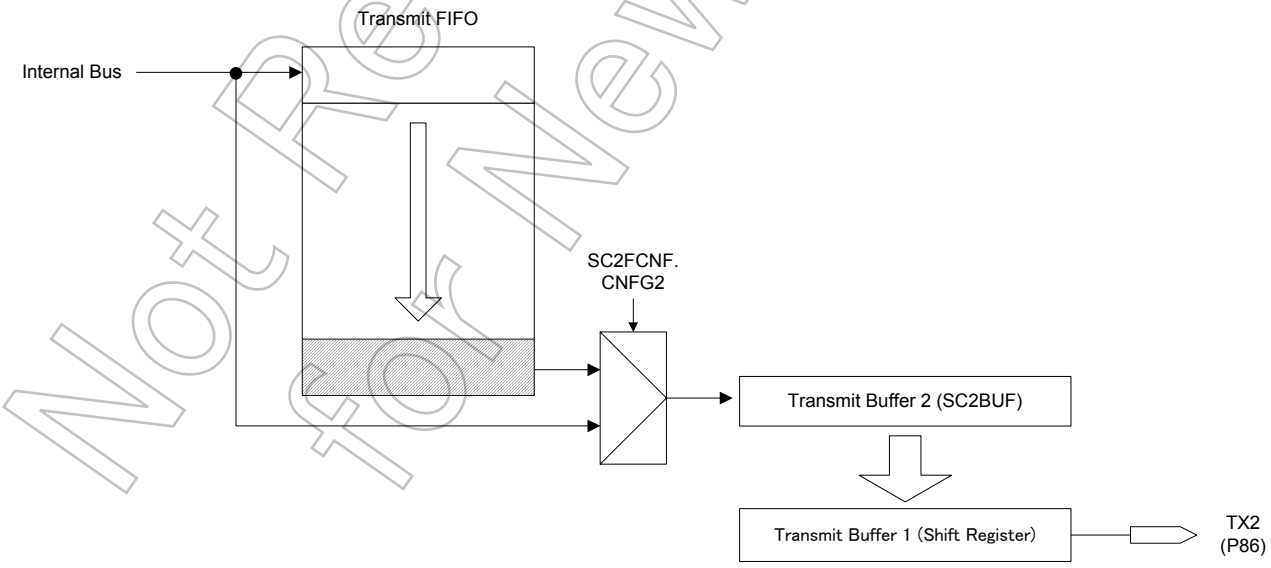


Figure 12.3.6 Transmit FIFO Block Diagram

Not Recommended for New Design

In SCLK Output mode (in I/O Interface mode), writing data in the transmit buffer starts a transmission in half-duplex mode. If the transmit buffer contains no data, transmit operation is halted. Setting the RXE bit in the SC2MOD0 to 1 starts receive operation in half-duplex mode. Receive operation can be stopped by clearing the SC2MOD0.RXE bit to 0 before reading the last frame. When the FIFO buffer is enabled, the following sequence must be executed to stop receive operation in half-duplex mode.

1. After receiving the last frame but one, disable the receive FIFO.
2. After receiving the last frame, disable receive operation by clearing the SC2MOD0.RXE bit.
3. Enable the receive FIFO with the same conditions as before.
   (When the transmit FIFO is enabled, it should be kept enabled.)
4. Read the data in the FIFO.
5. Disable the receive FIFO.
6. Read the last frame.

Operation in full-duplex mode is the same as transmit operation in half-duplex mode. The received data must be read before a next transmit frame has been written.

Note 1: When the transmit FIFO is used, do not access registers other than the SC2BUF, SC2FRS, and SC2FTS.

Note 2: When the receive FIFO is used, do not access the SC2CR or write to the SC2FRS.

Note 3: Do not write to the transmit FIFO when it is full. Before writing to the transmit FIFO, check the number of bytes stored in the transmit FIFO by using the SC2FTS.TLVL field.

Note 4: Do not read from the receive FIFO when it is empty. Before reading the receive FIFO, check the number of bytes stored in the receive FIFO by using the SC2FRS.RLVL field.

Not Recommended for New Design

12.3.19 Signal Generation Timing

(1) I/O Interface mode

Table 12.3.6 Signal Generation Timing in I/O Interface Mode

Receive operation

Interrupt (WBUF = 0)	SCLK Output Mode	Immediately after the rising edge of the last SCLK pulse
	SCLK Input Mode	Immediately after the rising or falling edge of the last SCLK pulse, as programmed
Interrupt (WBUF = 1)	SCLK Output Mode	Immediately after the rising edge of the last SCLK pulse (i.e., immediately after the frame is transferred to Receive Buffer 2) or immediately after the frame is read from Receive Buffer 2
	SCLK Input Mode	Immediately after the rising or falling edge of the last SCLK pulse, as programmed (i.e., immediately after the frame is transferred to Receive Buffer 2)
Overrun Error	SCLK Input Mode	Immediately after the rising or falling edge of the last SCLK pulse, as programmed

Transmit operation

Interrupt (WBUF = 0)	SCLK Output Mode	Immediately after the rising edge of the last SCLK pulse
	SCLK Input Mode	Immediately after the rising or falling edge of the last SCLK pulse, as programmed
Interrupt (WBUF = 1)	SCLK Output Mode	Immediately after the rising edge of the last SCLK pulse or immediately after the frame is transferred to Transmit Buffer 1
	SCLK Input Mode	Immediately after the rising or falling edge of the last SCLK pulse, as programmed or immediately after the frame is transferred to Transmit Buffer 1
Underrun error (WBUF=1)	SCLK Input Mode	Immediately after the rising or falling edge of the next SCLK pulse, as programmed

Note 1: Do not modify any control registers while data is being transmitted or received (receive operation is enabled).

Note 2: Do not disable receive operation (SC2MOD0.RXE = 0) while data is being received.

Not Recommended for New Design

(2) UART mode

Table 12.3.7 Signal Generation Timing in UART Mode

Receive operation

Mode	9 Data Bits	8 Data Bits with Parity	8 Data Bits with No Parity, 7 Data Bits with Parity 7 Data Bits wih No Parity
Interrupt	Middle of the first stop bit	Middle of the first stop bit	Middle of the first stop bit
Framing Error	Middle of the stop bit	Middle of the stop bit	Middle of the stop bit
Parity Error	—	Middle of the last bit (i.e., parity bit)	Middle of the last bit (i.e., parity bit)
Overrun Error	Middle of the stop bit	Middle of the stop bit	Middle of the stop bit

Transmit operation

Mode	9 Data Bits	8 Data Bits with Parity	8 Data Bits with No Parity, 7 Data Bits with Parity, 7 Data Bits with No Parity
Interrupt (WBUF = 0)	Simultaneously with transferring the stop bit	Simultaneously with transferring the stop bit	Simultaneously with transferring the stop bit
Interrupt (WBUF = 1)	Immediately after the frame is transferred to Transmit Buffer 1 (i.e., simultaneously with transferring the start bit)	Immediately after the frame is transferred to Transmit Buffer 1 (i.e., simultaneously with transferring the start bit)	Immediately after the frame is transferred to Transmit Buffer 1 (i.e., simultaneously with transferring the start bit)

Note 1: Do not modify any control registers while data is being transmitted or received (or receive operation is enabled).

Note 2: Do not disable receive operation (SC2MOD0.RXE = 0) while data is being received.

Note 3: The "middle" in the above table means the 9th bit of SIOCLK.

Not Recommended for New Design

12.4 Register Descripsion (Only channel 2 registers are described.)

Table 12.4.1 SIO Register Map

Address	Bits	Mnemonic	Register Name
0xFFFF_C480	8	SC0MOD0	Serial 0 Mode Control Register 0
0xFFFF_C481	8	SC0MOD1	Serial 0 Mode Control Register 1
0xFFFF_C484	8	SC0CR	Serial 0 Control Register
0xFFFF_C485	8	SC0MOD2	Serial 0 Mode Control Register 2
0xFFFF_C488	8	BR0CR	Baud Rate Generator Control Register (SIO0)
0xFFFF_C489	8	BR0ADD	Baud Rate Generator Additional Control Register (SIO0)
0xFFFF_C490	8	SC0BUF	Serial 0 Transmit/Receive Buffer Register
0xFFFF_C494	8	SC0FCNF	Serial 0 FIFO Configuration Register
0xFFFF_C498	8	SC0FTC	Serial 0 FIFO Transmit Control Register
0xFFFF_C499	8	SC0FRC	Serial 0 FIFO Receive Control Register
0xFFFF_C49C	8	SC0FTS	Serial 0 FIFO Transmit Status Register
0xFFFF_C49D	8	SC0FRS	Serial 0 FIFO Receive Status Register
0xFFFF_C4A0	8	SC1MOD0	Serial 1 Mode Control Register 0
0xFFFF_C4A1	8	SC1MOD1	Serial 1 Mode Control Register 1
0xFFFF_C4A4	8	SC1CR	Serial 1 Control Register
0xFFFF_C4A5	8	SC1MOD2	Serial 1 Mode Control Register 2
0xFFFF_C4A8	8	BR1CR	Baud Rate Generator Control Register (SIO1)
0xFFFF_C4A9	8	BR1ADD	Baud Rate Generator Additional Control Register (SIO1)
0xFFFF_C4B0	8	SC1BUF	Serial 1 Transmit/Receive Buffer Register
0xFFFF_C4B4	8	SC1FCNF	Serial 1 FIFO Configuration Register
0xFFFF_C4B8	8	SC1FTC	Serial 1 FIFO Transmit Control Register
0xFFFF_C4B9	8	SC1FRC	Serial 1 FIFO Receive Control Register
0xFFFF_C4BC	8	SC1FTS	Serial 1 FIFO Transmit Status Register
0xFFFF_C4BD	8	SC1FRS	Serial 1 FIFO Receive Status Register

Note: Although these registers are 8-bit wide, two registers at consecutive addresses can be accessed simultaneously with a 16-bit access instruction.

Not Recommended for New Design

Address	Bits	Mnemonic	Register Name
0xFFFF_C4C0	8	SC2MOD0	Serial 2 Mode Control Register 0
0xFFFF_C4C1	8	SC2MOD1	Serial 2 Mode Control Register 1
0xFFFF_C4C4	8	SC2CR	Serial 2 Control Register
0xFFFF_C4C5	8	SC2MOD2	Serial 2 Mode Control Register 2
0xFFFF_C4C8	8	BR2CR	Baud Rate Generator Control Register (SIO2)
0xFFFF_C4C9	8	BR2ADD	Baud Rate Generator Additional Control Register (SIO2)
0xFFFF_C4D0	8	SC2BUF	Serial 2 Transmit/Receive Buffer Register
0xFFFF_C4D4	8	SC2FCNF	Serial 2 FIFO Configuration Register
0xFFFF_C4D8	8	SC2FTC	Serial 2 FIFO Transmit Control Register
0xFFFF_C4D9	8	SC2FRC	Serial 2 FIFO Receive Control Register
0xFFFF_C4DC	8	SC2FTS	Serial 2 FIFO Transmit Status Register
0xFFFF_C4DD	8	SC2FRS	Serial 2 FIFO Receive Status Register
0xFFFF_C4E0	8	SC3MOD0	Serial 3 Mode Control Register 0
0xFFFF_C4E1	8	SC3MOD1	Serial 3 Mode Control Register 1
0xFFFF_C4E4	8	SC3CR	Serial 3 Control Register
0xFFFF_C4E5	8	SC3MOD2	Serial 3 Mode Control Register 2
0xFFFF_C4E8	8	BR3CR	Baud Rate Generator Control Register (SIO3)
0xFFFF_C4E9	8	BR3ADD	Baud Rate Generator Additional Control Register (SIO3)
0xFFFF_C4E0	8	SC3BUF	Serial 3 Transmit/Receive Buffer Register
0xFFFF_C4F4	8	SC3FCNF	Serial 3 FIFO Configuration Register
0xFFFF_C4F8	8	SC3FTC	Serial 3 FIFO Transmit Control Register
0xFFFF_C4F9	8	SC3FRC	Serial 3 FIFO Receive Control Register
0xFFFF_C4FC	8	SC3FTS	Serial 3 FIFO Transmit Status Register
0xFFFF_C4FD	8	SC3FRS	Serial 3 FIFO Receive Status Register

Note: Although these registers are 8-bit wide, two registers at consecutive addresses can be accessed simultaneously with a 16-bit access instruction.

Not Recommended for New Design

Serial 2 Mode Control Register 0

SC2MOD0 (0xFFFF_C4C0)

	7	6	5	4	3	2	1	0
Bit Symbol	TB8	CTSE	RXE	WU	SM		SC	
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	Bit 8 of a transmitted character	Handshake control 0: Disable CTS operation 1: Enable CTS operation	Receive control 0: Disable 1: Enable	Wake-up function 0: Disable 1: Enable	Serial transfer mode 00: I/O Interface mode (for SIO2, SIO3 only) 01: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode		Serial clock (for UART) 00: Timer TB2OUT 01: Baud rate generator 10: Internal clock (IMCLK) 11: External clock (SCLK2 input) (for SIO2, SIO3 only)	

Wake-up function

	9-bit UART mode	Other modes
0	Interrupt on every received frame	Don't care
1	Interrupt only when RB8 = 1	

Handshake ($\overline{\text{CTS}}$) control (for SIO2, SIO3 only)

0	Disable (Accept data streams at all times)
1	Enable

Note 1: In I/O Interface mode, the Serial Control Register (SC2CR) is used to select a serial clock.

Note 2: Like the SIO2, the SIO0, SIO1 and SIO3 allows use of the timer TB2OUT as a serial clock.

Note 3: The SC2MOD0, SC2MOD1 and SC2MOD2 registers must be set with the RXE bit cleared to 0. After setting these registers, set the RXE bit to 1.

Note 4: During transmit operation in half-duplex mode (SC2MOD1.FDPX=0) in I/O Interface mode (SC2MOD0.SM=00), do not set the RXE bit to 1.

Note 5: The TB8 bit is not double-buffered. Before writing to this bit, make sure that double-buffering is disabled and no transmit operation is in progress.

Not Recommended for New Design

Serial 2 Mode Control Register 1

SC2MOD1 (0xFFFF_C4C1)

	7	6	5	4	3	2	1	0
Bit Symbol	—	FDPX	—	UART	—	—	—	—
Read/Write	R/W	R/W	R/W	R/W	W			
Reset Value	0	0	0	0	0	0	0	0
Function		Sync method 0: Half duplex 1: Full duplex		UART output 0: Normal 1: 50% duty cycle				

Note: When the N + (16 - K)/16 clock division function is used, the duty ratio varies with the value of K.

Not Recommended for New Design

Serial 2 Mode Control Register 2

SC2MOD2 (0xFFFF_C4C5)

	7	6	5	4	3	2	1	0
Bit Symbol	TBEMP	RBFLL	TXRUN	SBLEN	DRCHG	WBUF	SWRST	
Read/Write	R			R/W			W	
Reset Value	1	0	0	0	0	0	00	
Function	Transmit buffer empty flag 0: Full 1: Empty	Receive buffer full flag 0: Empty 1: Full	Transmit-in-progress flag 0: Stopped 1: Trans-mitting	Number of stop bits 0: 1 bit 1: 2 bits	Bit sequence 0: LSB first 1: MSB first	Double-buffering 0: Disable 1: Enable	Software reset A write of 10 followed by a write of 01	

Symbol	Function
SWRST	A write of 10 followed by a write of 01 to this field resets the module, thus initializing the RXE bit in the SC2MOD0, the TBEMP, RBFLL and TXRUN bits in the SC2MOD2, the OERR, PERR and FERR bis in the SC2CR, and the internal circuits.
WBUF	Enables or disables double-buffering for transmit (SCLK output or input) or receive (SCLK output) operation in I/O Interface mode and transmit operation in UART mode. For any other modes of operation, double-buffering is always enabled.
DRCHG	Specifies the bit transfer sequence in I/O Interface mode. In UART mode, the LSB is always transferred first.
SBLEN	Specifies the number of transmit stop bits in UART mode. For receive operation, a single stop bit is used regardless of the setting of this bit.
TXRUN	A status flag indicating whether transmit shift operation is in progress. When this bit is set to 1, transmit operation is in progress. When this bit is cleared to 0, transmit operation is completed (if TBEMP = 1) or the trnamit buffer contains a next frame and is ready for transmission (if TBEMP= 0).
RBFLL	A flag indicating whether Receive Buffer 2 is full. The RBFLL bit is set to 1 once a received frame is transferred from Receive Buffer 1 to Receive Buffer 2. It is cleared when the frame is read from Receive Buffer 2. When double-buffering is disabled, the RBFLL bit has no meaning.
TBEMP	A flag indicating whether Transmit Buffer 2 is empty. The TBEMP bit is set to 1 once a frame is transferred from Transmit Buffer 2 to Transmit Buffer 1. It is cleared when a next frame is written to Transmit Buffer 2. When double-buffering is disabled, the TBEMP bit has no meaning.

Note 1: If the module needs to be reset while it is transmitting data, two consecutive software reset sequences (i.e., 10, 01, 10, 01) must be executed.

Note 2: This register does not support bit manipulation instructions.

Not Recommended for New Design

Serial 2 Control Register

SC2CR
(0xFFFF_C4C4)

	7	6	5	4	3	2	1	0
Bit Symbol	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
Read/Write	R	R/W		R (Cleared when read)			R/W	
Reset Value	0	0	0	0	0	0	0	0
Function	Bit 8 of a received character	Parity type 0: Odd 1: Even	Parity 0: Enable 1: Disable	1: Error has occurred.			0: SCLK2 ↑ 1: SCLK2 ↓	0: Baud rate generator 1: SCLK2 input
				Overrun	Parity/ underrun	Framing		

→ Input clock in I/O Interface mode (for CH2, CH3 only)

0	Baud rate generator
1	SCLK2 pin input

→ Active edge for SCLK2 input (for CH2, CH3 only)

0	Data is transmitted/received on the SCLK2 rising edge.
1	Data is transmitted/received on the SCLK2 falling edge.

→ Framing error flag
→ Parity error/underrun error flag
→ Overrun error flag

} These bits are cleared to 0 when read.

→ Patiy type

0	Odd parity
1	Even parity

Note 1: All error flags are cleared to 0 when read.

Note 2: This register does not support bit manipulation instructions.

Note 3: The SC2CR.FERR bit should not be polled; instead, it should be read in the INTRX2 interrupt routine before the receive buffer is read. For details, see the example in "12.5.3 8-Bit UART Mode".

Not Recommended for New Design

Baud Rate Generator Control Register

BR2CR (0XFFFF_C4C8)

	7	6	5	4	3	2	1	0
Bit Symbol	−	DVS	PRE		BR2S			
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	Must be written as 0.	N + (16 − K)/16 division function 0: Disable 1: Enable	00: IMCLK/2 01: IMCLK/8 10: IMCLK/32 11: IMCLK/128		Clock divisor value N			

Clock source for baud rate generator

00	Internal clock IMCLK/2
01	Internal clock IMCLK/8
10	Internal clock IMCLK/32
11	Internal clock IMCLK/128

Baud Rate Generator Additional Control Register

BR2ADD (0xFFFF_C4C9)

	7	6	5	4	3	2	1	0
Bit Symbol					BR2K			
Read/Write					R/W			
Reset Value	0	0	0	0	0	0	0	0
Function					Value K in N + (16 − K)/16			

Clock divisor value for the baud rate generator

BR2ADD.BR2K \ BR2CR.BR2S	BR2CR.DVS= 1		BR2CR.DVS= 0
	0000 (N = 16) to 0001 (N = 1)	0010 (N = 2) to 1111 (N = 15)	0001 (N = 1) (UART only) to 1111 (N = 15) 0000 (N = 16)
0000	Prohibited	Prohibited	Divide by N
0001(K = 1) to 1111(K = 15)	Prohibited	Divide by $N + \frac{(16-K)}{16}$	Divide by N

Note 1: The baud rate generator divisor cannot be set to 1 in UART mode if the N + (16 – K)/16 clock division function is enabled. In I/O Interface mode, do not set the baud rate generator divisor to 1; setting the divisor to 1 will cause incorrect operation.

Note 2: To use the N + (16 – K)/16 clock division function, the value of K must be programmed in the BR2ADD.BR2K field before setting the BR2CR.DVS bit to 1. However, the N + (16 – K)/16 clock division function is not usable when BR2CR.BR2S = 0000 (N = 16) or 0001 (N = 1).

Note 3: The N + (16 – K)/16 clock division function can only be used in UART mode. In I/O Interface mode, it must be disabled by clearing the BR2CR.DVS bit to 0.

Not Recommended for New Design

Serial Transmit/Receive Buffer Register

SC2BUF (0xFFFF_C4D0)

	7	6	5	4	3	2	1	0
Bit Symbol	TB							
Read/Write	W							
Reset Value	0	0	0	0	0	0	0	0
Function	Transmit buffer							

	7	6	5	4	3	2	1	0
bit Symbol	RB							
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0
Function	Receive buffer							

Note: In I/O Interface mode (SC2MOD0.SM = 0), do not write to the transmit buffer during receive operation in half-duplex mode (SC2MOD1.FDPX = 0).

Not Recommended for New Design

Serial 2 FIFO Configuration Register

SC2FCNF (0xFFFF_C4D4)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	—	CNFG	
Read/Write	R						R/W	
Reset Value	0	0	0	0	0	0	00	
Function							FIFO 00: Disable 01: Transmit (16 bytes) 10: Receive (16 bytes) 11: Transmit/Receive (8 bytes each)	

Note: For continuous transmit/receive operations using the FIFO, double-buffering must be enabled (SC2MOD2.WBUF=1).

Serial 2 FIFO Receive Control Register

SC2FRC (0xFFFF_C4D9)

	7	6	5	4	3	2	1	0
Bit Symbol	RIL				RFIS	RFCL	RFIE	—
Read/Write	R/W					W	R/W	R
Reset Value	0000				0	0	0	0
Function	Interrupt level 0000: Interrupt is generated when the receive FIFO reaches 1 byte. 0001: Interrupt is generated when the receive FIFO reaches 2 bytes. 0010: Interrupt is generated when the receive FIFO reaches 3 bytes. 0011: Interrupt is generated when the receive FIFO reachs 4 bytes. … 1111: Interrupt is generated when the receive FIFO reaches 16 bytes. Setting has no effect when CNFG=00/01. The most significant bit must be cleared to 0 when CNFG=11.				Interrupt condition 0: Interrupt generated only when RIL=RLVL 1: Interrupt generated when RIL≦RLVL	FIFO clear Setting this bit to 1 clears FIFO value. (This bit is always read as 0.)	Receive FIFO interrupt 0: Disable 1: Enable	

RFIS: When RFIS=0, a receive FIFO interrupt is generated only when the number of bytes stored in the receive FIFO set in the SC2FRS.RLVL matches the interrupt generation level set in the SC2FRC.RIL.
When RFIS=1, a receive FIFO interrupt is generated when the number of bytes stored in the receive FIFO set in the SC2FRS.RLVL is equal to or greater than the interrupt generation level set in the SC2FRC.RIL.

Note: This register does not support bit manipulation instructions.

Not Recommended for New Design

Serial 2 FIFO Transmit Control Register

SC2FTC (0xFFFF_C4D8)

	7	6	5	4	3	2	1	0
Bit Symbol	TIL				TFIS	TFCL	TFIE	—
Read/Write	R/W					W	R/W	R
Reset Value	0111				0	0	0	0
Function	Interrupt level 0000: Interrupt is generated when the transmit FIFO reaches 1 byte. 0001: Interrupt is generated when the transmit FIFO reaches 2 bytes. 0010: Interrupt is generated when the transmit FIFO reaches 3 byes. 0011: Interrupt is generated when the transmit FIFO reaches 4 bytes. … 1111: Interrupt is generated when the transmit FIFO reaches 16 bytes. Setting has no effect when CNFG=00/10. The most significant bit must be cleared to 0 when CNFG=11.				Interrupt condition 0: Interrupt generated only when TIL=TLVL. 1: Interrupt generated when TIL≧TLVL	FIFO clear Setting this bit to 1 clears the FIFO value. (This bit is always read as 0.)	Transmit FIFO interrupt 0：Disable 1: Enable	

TFIS: When TFIS=0, a transmit FIFO interrupt is generated only when the number of bytes stored in the transmit FIFO set in the SC2FTS.TLVL matches the interrupt generation level set in the SC2FTC.TIL.
When TFIS=1, a transmit FIFO interrupt is generated when the number of bytes stored in the transmit FIFO set in the SC2FTS.TLVL is equal to or smaller than the interrupt generation level set in the SC2FTC.TIL.

Note: This register does not support bit manipulation instructions.

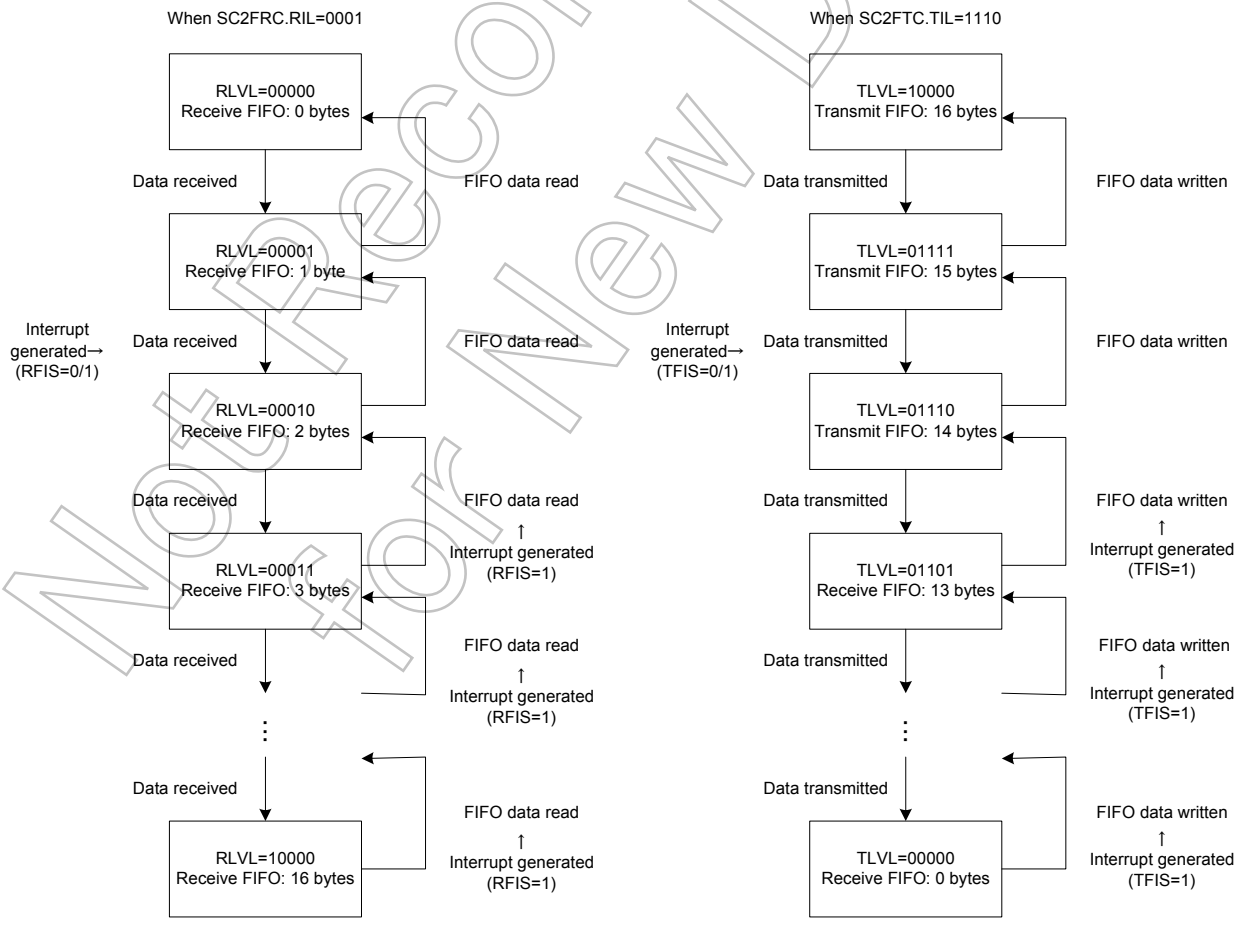


Figure 12.4.1 Example of Interrupt Generaton Timing when Using FIFO

Not Recommended for New Design

Serial 2 FIFO Receive Status Register

SC2FRS (0xFFFF_C4DD)

	7	6	5	4	3	2	1	0
Bit Symbol	RUR	—	—	RLVL				
Read/Write	R							
Reset Value	0	0	0	00000				
Function	This bit is set to 1 when the receive FIFO is full.	Can be read as 0.	Can be read as 0.	Receive FIFO byte count 00000: 0 bytes 00001: 1 byte 00010: 2 bytes 00011: 3 bytes … 10000: 16 bytes				

RUR: The RUR bit is set to 1 if an attempt to store a new value is made when the receive FIFO is already full. This bit is cleared to 0 when it is read while the receive FIFO buffer is not full.

Note: This register does not support bit manipulation instructions.

Serial 2 FIFO Transmit Status Register

SC2FTS (0xFFFF_C4DC)

	7	6	5	4	3	2	1	0
Bit Symbol	TUR	—	—	TLVL				
Read/Write	R							
Reset Value	1	0	0	00000				
Function	This bit is set to 1 when the transmit FIFO is empty.	Can be read as 0.	Can be read as 0.	Transmit FIFO byte count 00000: 0 bytes 00001: 1 byte 00010: 2 bytes 00011: 3 bytes … 10000: 16 bytes				

TUR: The TUR bit is set to 1 when the transmit FIFO becomes empty. When the first byte is stored in the transmit FOFO, it is immediately transferred to the transfer buffer (SC2BUF), causing the transmit FIFO to become empty and the TUR bit to be set to 1. This bit is automatically cleared to 0 when data is written to the transmit FIFO.

Not Recommended for New Design

12.5 Operating Modes

12.5.1 I/O Interface Mode

I/O Interface mode utilizes a synchronization clock (SCLK), which can be configured for either Output mode in which the SCLK clock is driven out from the TMP19A71 or Input mode in which the SCLK clock is supplied externally.

(1) Transmit operation (half-duplex)

SCLK Output mode

When transmit double-buffering is disabled (SC2MOD2.WBUF = 0) in SCLK Output mode, each time the TX19A core processor writes a frame to the transmit buffer, the 8 bits of the frame are shifted out on the TXD2 pin, and the synchronization clock is driven out from the SCLK2 pin. When all the bits have been shifted out, the INTTX2 interrupt is generated.

When transmit double-buffering is enabled (SC2MOD2.WBUF = 1), a frame is transferred from Transmit Buffer 2 to Transmit Buffer 1 (shift register) once the TX19A core processor writes the frame to Transmit Buffer 2 when no data is being transmitted or the last frame in Transmit Buffer 1 has been sent. At this time, the transmit buffer empty flag (SC2MOD2.TBEMP) is set to 1 and the INTTX2 interrupt is generated. If there is no data to be transferred from Transmit Buffer 2 to Transmit Buffer 1, however, the INTTX2 interrupt is not generated and SCLK2 output is stopped.

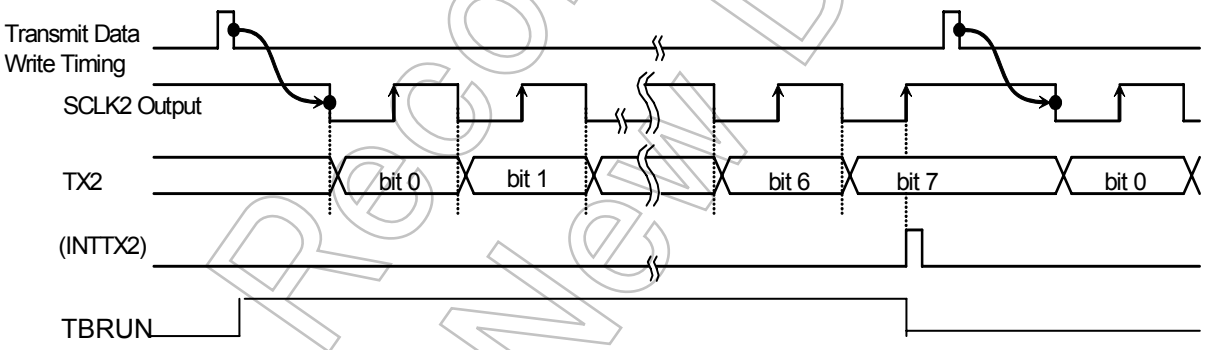


Figure 12.5.1 Transmit Operation in I/O Interface Mode
(SCLK Output mode, double-buffering disabled)

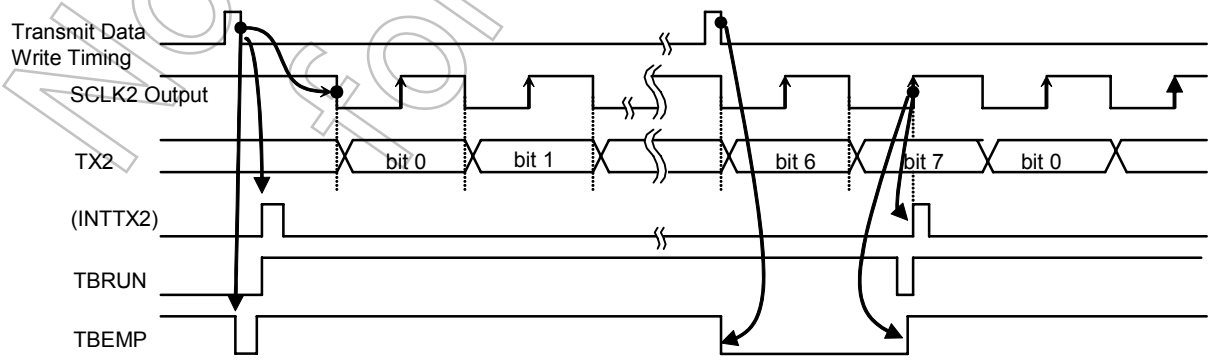


Figure 12.5.2 Transmit Operation in I/O Interface Mode

(SCLK Output mode, double-buffer enabled, data in Transmit Buffer 2)

Not Recommended for New Design

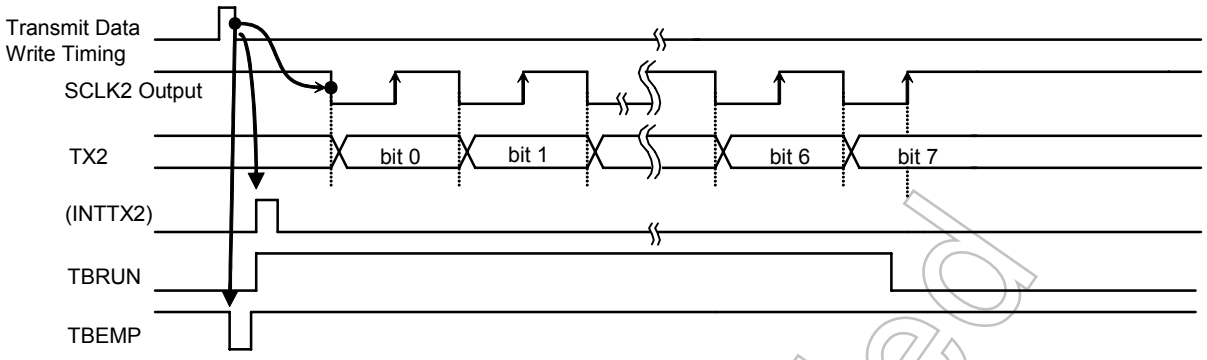


Figure 12.5.3 Transmit Operation in I/O Interface mode

(SCLK Output mode, double-buffering enabled, no data in Transmit Buffer 2)

SCLK Input mode

When transmit double-buffering is disabled (SC2MOD2.WBUF = 0) in SCLK Input mode, the 8 bits of a frame in the transmit buffer are shifted out on the TX2 pin when the SCLK2 input becomes active (i.e., the first rising or falling edge, as programmed) with transmit data written in the transmit buffer. The TX19A core processor must load a next frame into the transmit buffer by point A (shown in Figure 12.5.4).

When transmit double-buffering is enabled (SC2MOD2.WBUF = 1), a frame is transferred from Transmit Buffer 2 to Transmit Buffer 1 (shift register) once the TX19A core processor writes the frame to Transmit Buffer 2 before the SCLK2 input becomes active or once the last frame in Transmit Buffer 1 has been sent. At this time, the transmit buffer empty flag (SC2MOD2.TBEMP) is set to 1 and the INTTX2 interrupt is generated. If the SCLK2 input becomes active before a frame is written to Transmit Buffer 2, an underrun error occurs and 8 bits of dummy data (0xFF) are sent although the internal bit counter starts counting.

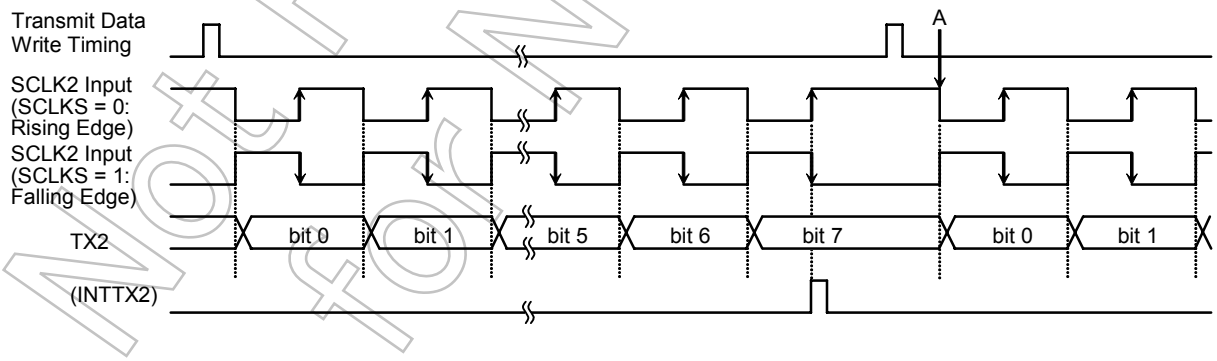


Figure 12.5.4 Transmit Operation in I/O Interface Mode
(SCLK Input mode, double-bufferig disabled)

Not Recommended for New Design

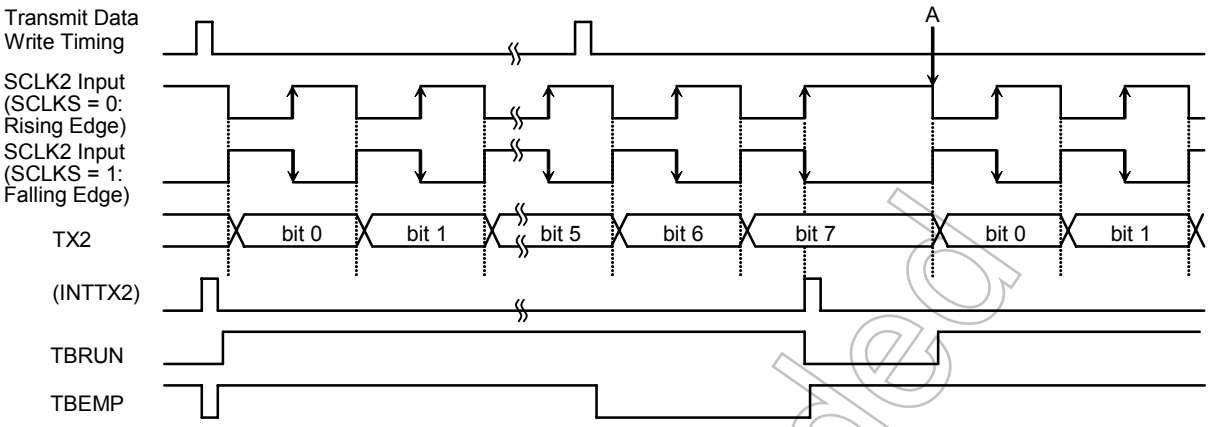


Figure 12.5.5 Transmit Operation in I/O Interface Mode
(SCLK Input mode, double-buffering enabled, data in Transmit Buffer 2)

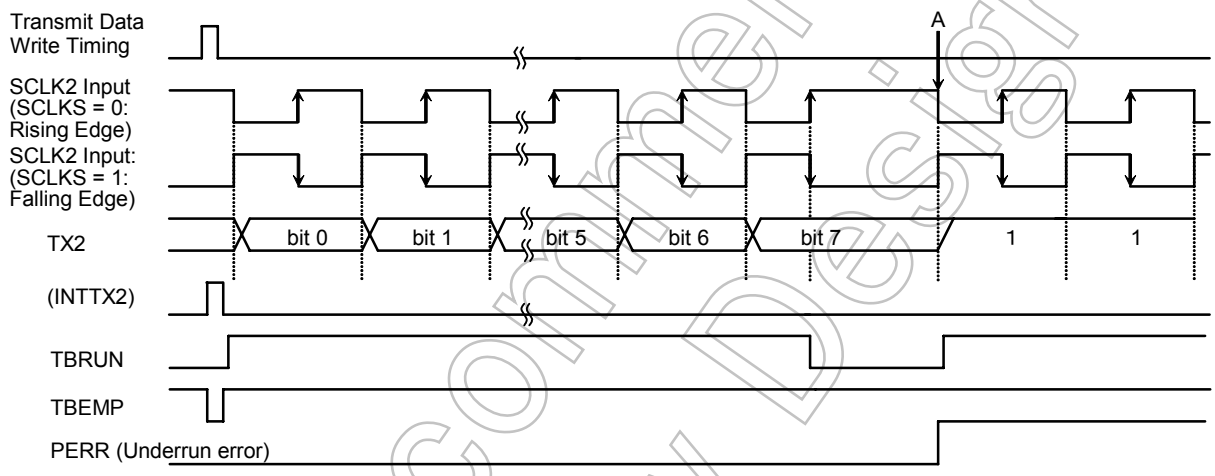


Figure 12.5.6 Transmit Operation in I/O Interface Mode
(SCLK Input mode, double-buffering enabled, no data in Transmit Buffer 2)

Not Recommended for New Design

(2) Receive operation (half-duplex)

SCLK Output mode

When receive double-buffering is disabled (SC2MOD2.WBUF = 0) in SCLK Output mode, each time the TX19A core processor picks up the frame in Receive Buffer 1, the synchronization clock is driven out from the SCLK2 pin to shift the next frame into Receive Buffer 1. When a whole-8-bit frame has been received in Receive Buffer 1, the INTRX2 interrupt is generated.

The SCLK output is initiated by setting the SC2MOD0.RXE bit to 1. When receive double-buffering is enabled (SC2MOD2.WBUF = 1), the frame received first is transferred to Receive Buffer 2 and then a next frame is received into Receive Buffer 1. Once a frame is transferred from Receive Buffer 1 to Receive Buffer 2, the Receive Buffer Full flag (SC2MOD2.RBFULL) is set to 1 and the INTRX2 interrupt is generated.

After a frame has been transferred to Receive Buffer 2, the TX19A core processor or DMAC should read it before all 8 bits of a next frame are received. Otherwise, the INTRX2 interrupt is not generated and SCLK2 output is stopped. If the TX19A core processor or DMAC subsequently reads the frame in Receive Buffer 2 in this state, the next frame is transferred from Receive Buffer 1 to Receive Buffer 2, generating the INTRX2 interrupt to restart receive operation.

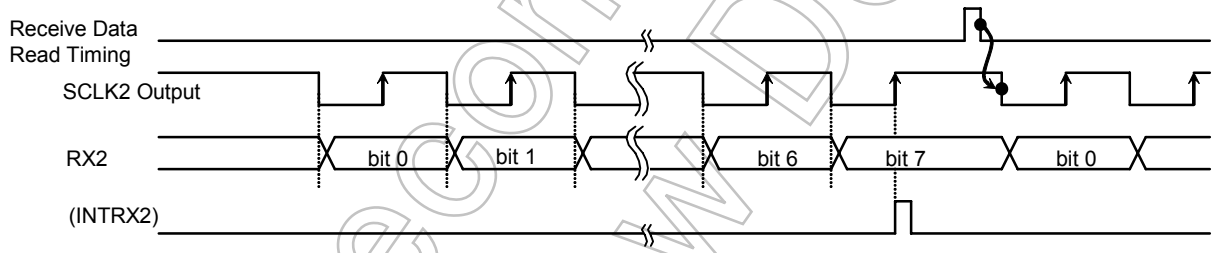


Figure 12.5.7 Receive Operation in I/O Interface Mode

(SCLK Output mode, double-buffering disabled)

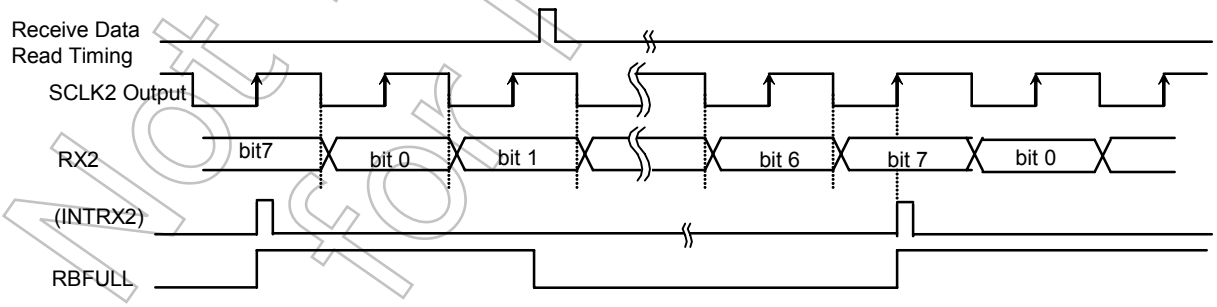


Figure 12.5.8 Receive Operation in I/O Interface Mode

(SCLK Output mode, double-buffering enabled, reading Receive Buffer 2)

Not Recommended for New Design

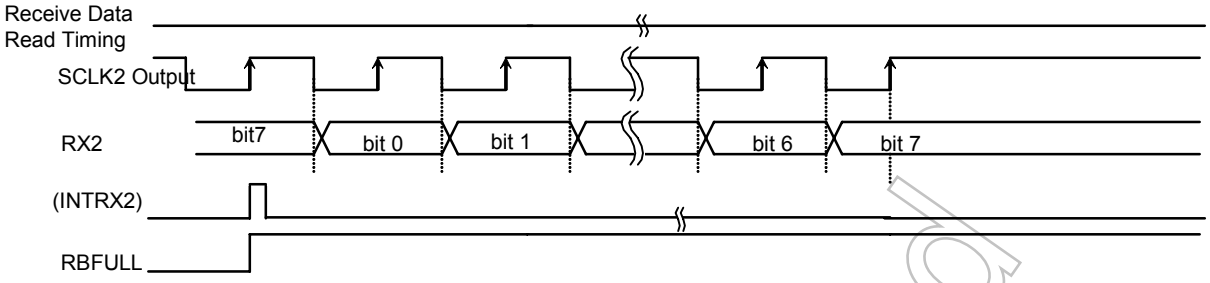


Figure 12.5.9 Receive Operation in I/O Interface Mode

(SCLK Output mode, double-buffering enabled, not reading Receive Buffer 2)

SCLK Input mode

In SCLK Input mode, receive double-buffering is always enabled. A received frame is transferred to Receive Buffer 2 so that a next frame can be received continuously into Receive Buffer 1.

The INTRX2 interrupt is generated every time a frame is transferred from Receive Buffer 1 to Receive Buffer 2.

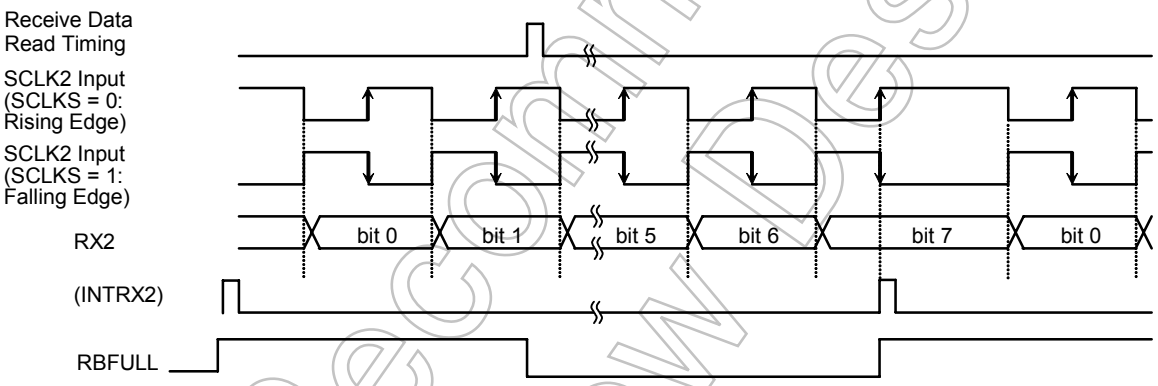


Figure 12.5.10 Receive Operation in I/O Interface Mode

(SCLK Input mode, reading Receive Buffer 2)

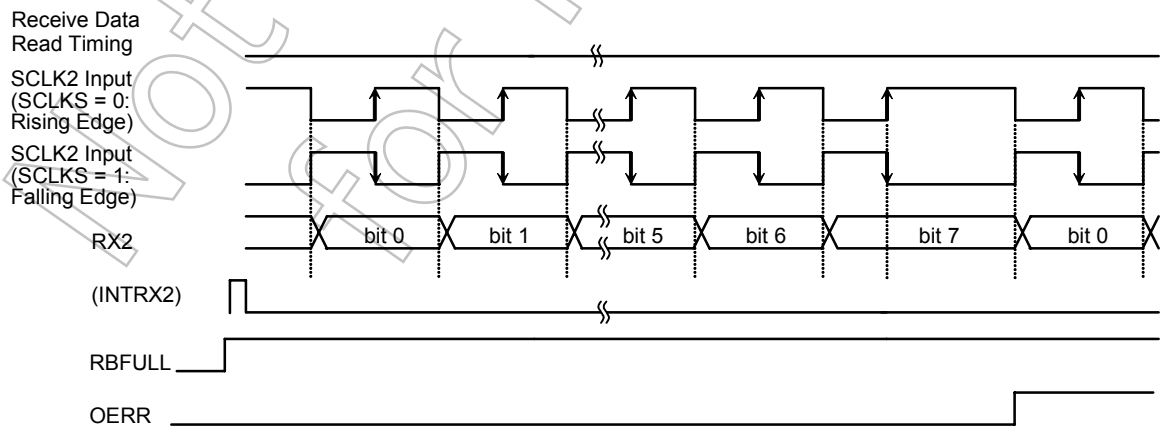


Figure 12.5.11 Receive Operation in I/O Interface Mode

(SCLK Input mode, not reading Receive Buffer 2)

Not Recommended for New Design

Note: To perform receive operation, the SC2MOD0.RXE bit must be set to 1 in both SCLK Input and SCLK Output modes.

(3) Transmit/receive operation (full-duplex)

Setting the SC2MOD1.FDPX2 bit to 1 enables full-duplex communication.

SCLK Output mode

When transmit/receive double-bufferig is disabled (SC2MOD2.WBUF = 0) in SCLK Output mode, each time the TX19A core processor writes a frame to the transmit buffer, the synchronization clock is driven out from the SCLK2 pin to shift an 8-bit frame into Receive Buffer 1, generating the INTRX2 interrupt. At the same time, the frame written to the transmit buffer is shifted out on the TX2 pin. When all the bits have been shifted out, the transmit-done interrupt (INTTX2) is generated and SCLK2 output is stopped. When the TX19A core processor subsequently picks up the frame in the receive buffer and writes a next frame to the transmit buffer, next transmit/receive operation starts, regardless of whether the TX19A core processor first reads the receive buffer or writes data to the transmit buffer.

When transmit/receive double-buffering is enabled (SC2MOD2.WBUF = 1), each time the TX19A core processor writes a frame to Transmit Buffer 2, the synchronization clock is driven out from the SCLK2 pin to shift an 8-bit frame into Receive Buffer 1; it is then transferred to Receive Buffer 2, generating the INTRX2 interrupt. At the same time, the frame stored in Transmit Buffer 1 is shifted out on the TXD2 pin. When all the bits have been shifted out, the transmit-done interrupt (INTTX2) is generated and the next frame is transferred from Transmit Buffer 2 to Transmit Buffer 1. Durnig the above sequence, SCLK output is stopped if Transmit Buffer 2 becomes empty (SC2MOD2.TBEMP = 1) or if Receive Buffer 2 still contains data (SC2MOD2.RBFULL = 1). When the TX19A core processor suqsequently picks up the frame in Receive Buffer 2 and writes a next frame to Transmit Buffer 2, SCLK2 output is restarted so that next transmit/receive operation starts.

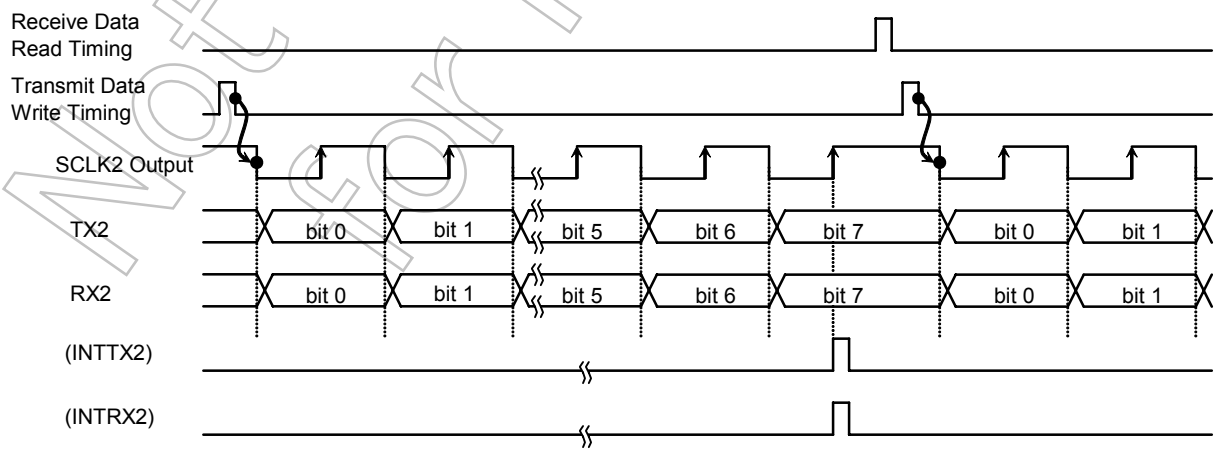


Figure 12.5.12 Transmit/Receive Operation in I/O Interface Mode
(SCLK Output mode, double-buffering disabled)

Not Recommended for New Design

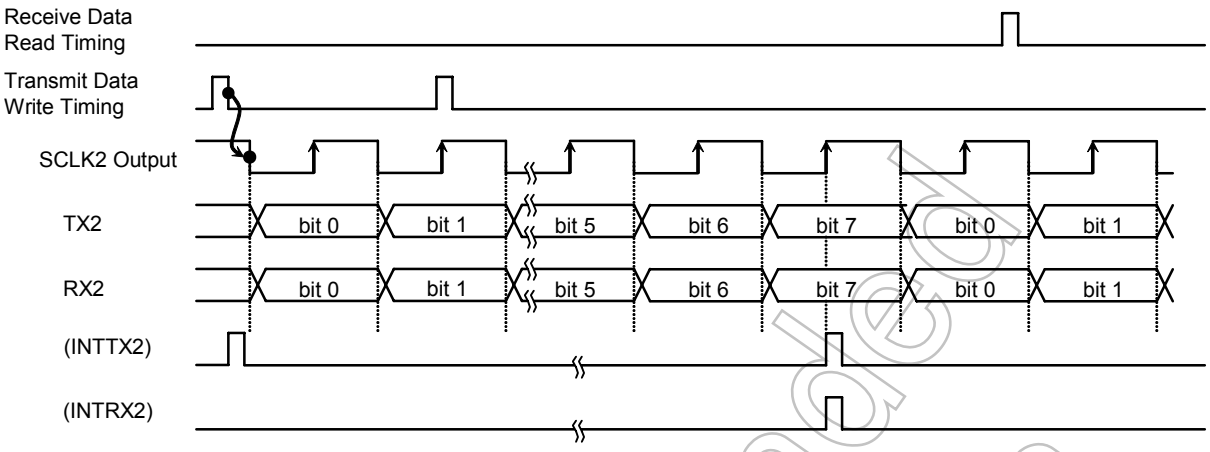


Figure 12.5.13 Transmit/Receive Operation in I/O Interface Mode
(SCLK Output mode, double-buffering enabled, no next data)

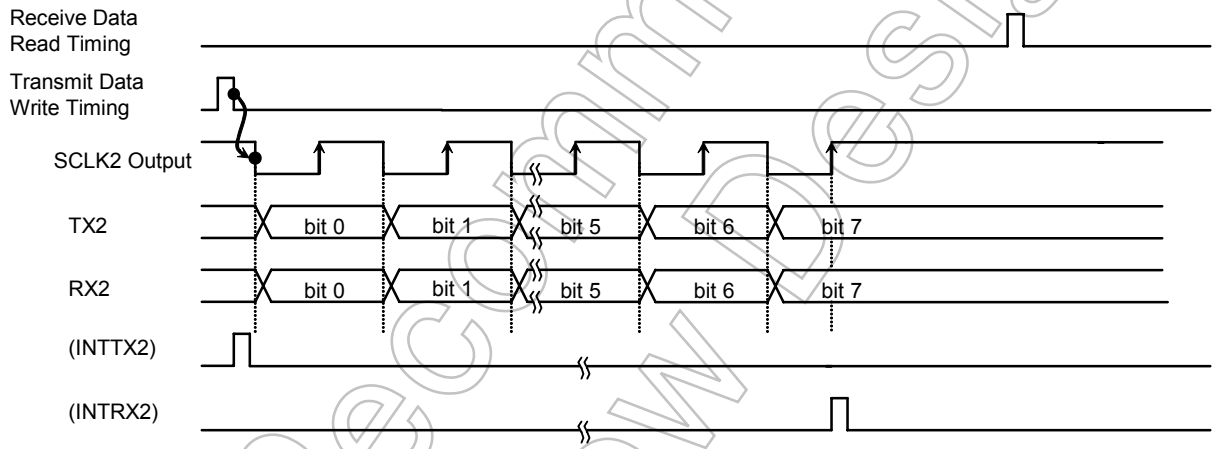


Figure 12.5.14 Transmit/Receive Operation in I/O Interface Mode
(SCLK Output mode, double-buffering enabled, continuous transfer)

Not Recommended for New Design

SCLK Input mode

When transmit double-buffering is disabled (SC2MOD2. WBUF = 0) in SCLK Input mode (receive double-buffering is always enabled in this mode), the TX19A core processor must write a frame to the transmit buffer before the SCLK2 input becomes avtive. The 8 bits of a frame in the transmit buffer are shifted out on the TX2 pin, and the 8 bits of a received frame are shifted into Receive Buffer 1, synchronous to the programmed edge of the SCLK2 input. When all the bits have been shifted out, the transmit-done interrupt (INTTX2) is generated. When all the bits have been received, the frame is transferred from Receive Buffer 1 to Receive Buffer 2, generating the INTRX2 interrupt. The TX19A core processor must load a next frame into the transmit buffer before the SCLK signal for the next frame is input (i.e., by point A shown in Figure 12.5.15 below). The TX19A core processor must also pick up the frame in Receive Buffer 2 before a next frame has been received.

When transmit/receive double-buffering is enabled (SC2MOD2.WBUF = 1), a frame is transferred from Transmit Buffer 2 to Transmit Buffer 1 once the last frame in Transmit Buffer 1 has been sent. At this time, the INTTX2 interrupt is generated. When the 8-bit frame, received in parallel with transmission, has been shifted into Receive Buffer 1, it is transferred to Receive Buffer 2, generating the INTRX2 interrupt. When the SCLK2 is subsequently activated, the frame stored in Transmit Buffer 1 is shifted out while a next frame is received into Receive Buffer 1. If the TX19A core processor does not read the frame from Receive Buffer 2 before the last bit of a next frame is received, an overrun error occurs. If the TX19A core processor does not write a frame to Transmit Buffer 2 before the SCLK2 input is subsequently activated, an underrun error occurs.

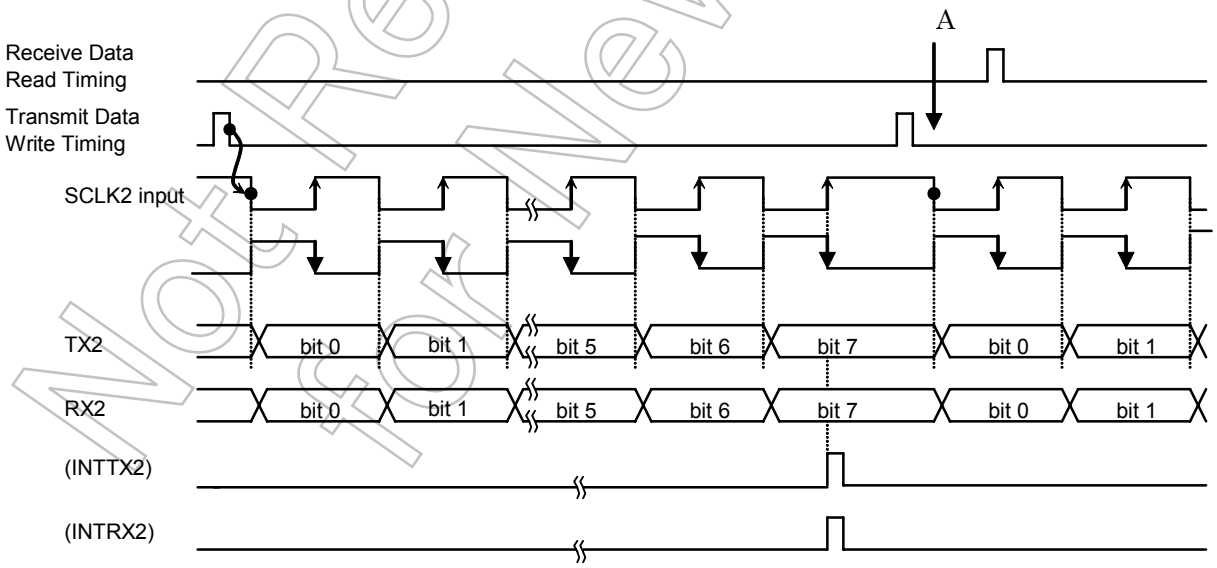


Figure 12.5.15 Transmit/Receive Operation in I/O Interface Mode
(SCLK Input mode, double-buffering disabled)

Not Recommended for New Design

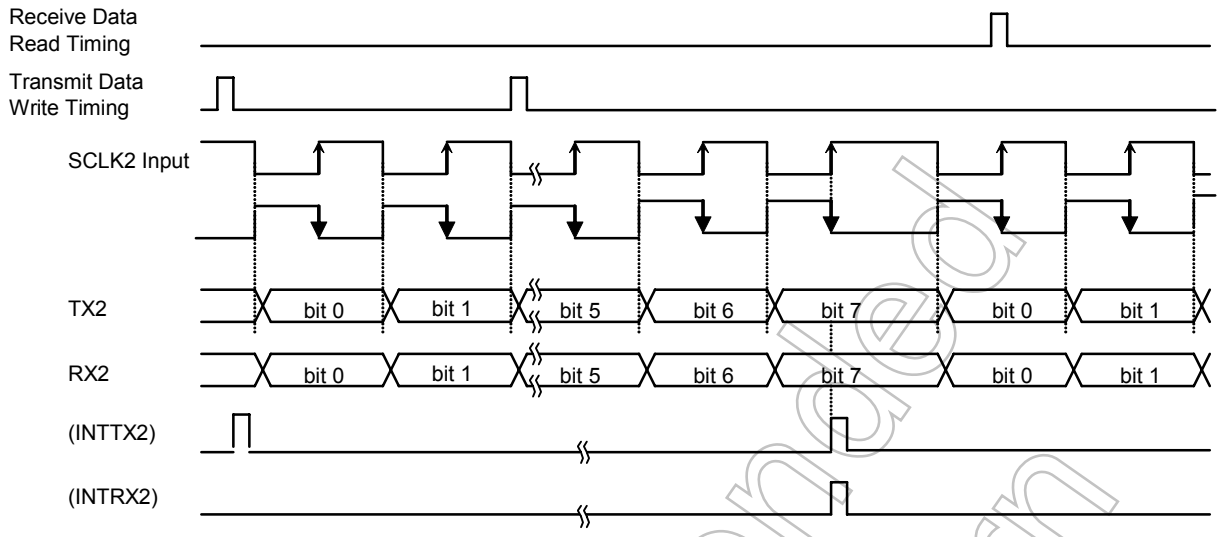


Figure 12.5.16 Transmit/Receive Operation in I/O Interface Mode
(SCLK Input mode, double-buffering enabled, no error occurred)

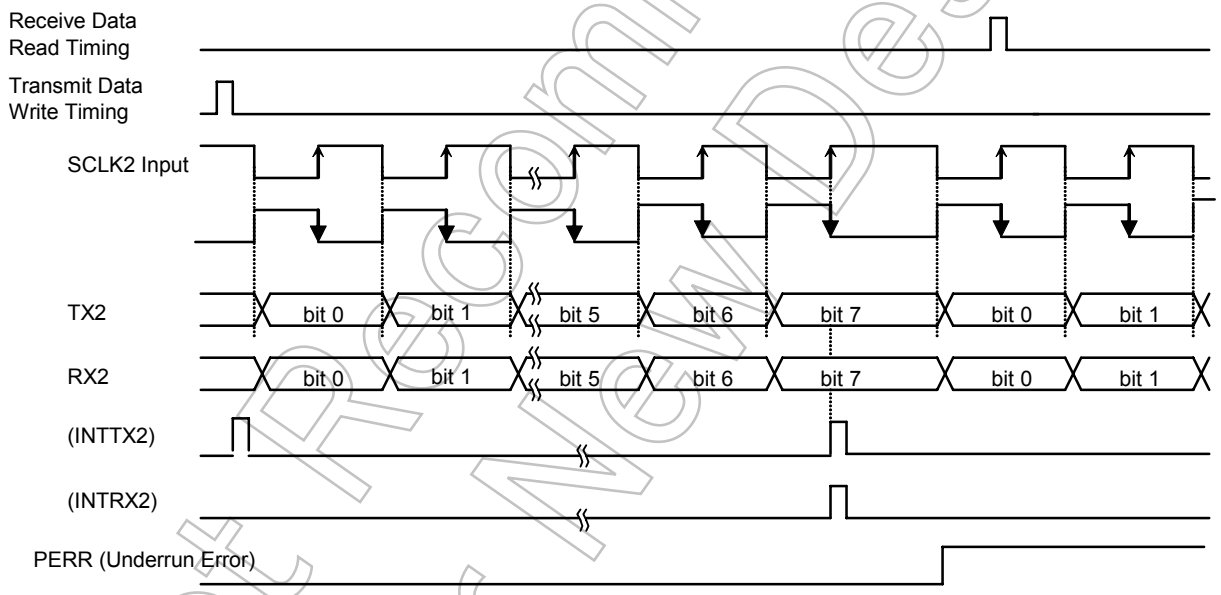


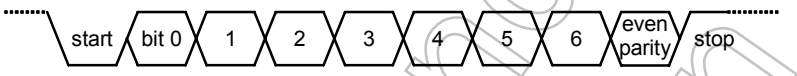
Figure 12.5.17 Transmit/Receive Operation in I/O Interface Mode
(SCLK Input mode, double-buffering enabled, an underrun error occurred)

Not Recommended for New Design

12.5.2 7-Bit UART Mode

Setting the SM field in the SC2MOD0 to 01 puts the SIO2 in 7-Bit UART mode. In this mode, the parity bit can be added to the transmitted frame, and the receiver can perform a parity check on incoming data. Parity can be enabled and disabled through the programming of the PE bit in the SC2CR. When the PE bit is set to 1 to enable parity, the SC2CR.EVEN bit selects even or odd parity. The SBLEN bit in the SC2MOD2 specifies the number of stop bits.

Example: Transmitting 7-bit UART frames with an even-parity bit



Clocking conditions:

System clock:	: 56 MHz
IMBUS clock	: 1/2 (28 MHz)
Prescaler clock	: IMCLK/32
Transfer rate	: 4800 bps (fsys = 56 MHz)

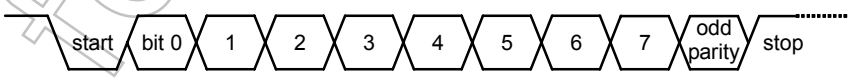
	7	6	5	4	3	2	1	0	
P8CR ←	–	1	–	–	–	–	–	–	Configure the P86 pin as TX2.
P8FR1 ←	–	1	–	–	–	–	–	–	
SC2MOD0 ←	X	0	–	X	0	1	0	1	Select 7-Bit UART mode.
SC2CR ←	X	1	1	X	X	X	0	0	Select even parity.
BR2CR ←	0	1	1	0	1	0	X	1	N=11, and K is valid. IMCLK/32
BR2ADD ←	0	0	0	0	1	0	1	0	Set the transfer rate to 4800 bps. (K = 10)
IMR52 ←	–	1	0	0	–	1	0	0	Enable the INTTX2 interrupt and set its priority level to 4.
SC2BUF ←	X	*	*	*	*	*	*	*	Load the transmit buffer with a frame.

Note: X = Don't care, –: No change

12.5.3 8-Bit UART Mode

Setting the SM field in the SC2MOD0 to 10 puts the SIO2 in 8-Bit UART mode. In this mode, the parity bit can be added to the transmitted frame, and the receiver can perform a parity check on incoming data. Parity can be enabled and disabled through the programming of the PE bit in the SC2CR. When the PE bit is set to 1 to enable parity, the SC2CR.EVEN bit selects even or odd parity.

Example: Transmitting 8-bit UART frames with an odd-parity bit



Clocking conditions

System clock	: 56 MHz
High-speed clock gear	: 1/2 (28 MHz)
Prescaler clock	: IMCLK/32
Transfer rate	: 9600 bps (fsys = 56 MHz)

Not Recommended for New Design

- Settings in the main routine

```
                  7 6 5 4 3 2 1 0
P8IER     ← – – 0 – – – – –   }  Configure the P85 pin as RX2.
P8FR1     ← – – 1 – – – – –   }
SC2MOD0   ← – 0 0 X 1 0 0 1      Select 8-Bit UART mode.
SC2CR     ← X 0 1 X X X X X      Select odd parity.
BR2CR     ← 0 1 1 0 0 1 0 1      N=5, and K is valid.
BR2ADD    ← 0 0 0 0 0 1 0 1      Set the transfer rate to 9600 bps. (K = 5)
IMR53     ← – 1 0 0 – 1 0 0      Enable the INTRX2 interrupt and set its priority level to 4.
SC2MOD0   ← – – 1 X – – – –      Enable reception.
```

- Example of interrupt routine processing

```
ICLR  ← 0 1 1 0 1 0 1 0 0   Clear the interrupt request.
Reg.  ← SC2CR AND 0x1C      }
if Reg. ≠ 0 then error      }  Check for errors.
Reg.  ← SC2BUF              Read received data.
End of interrupt processing
```

Note: X = Don't care, –: No change

Not Recommended for New Design

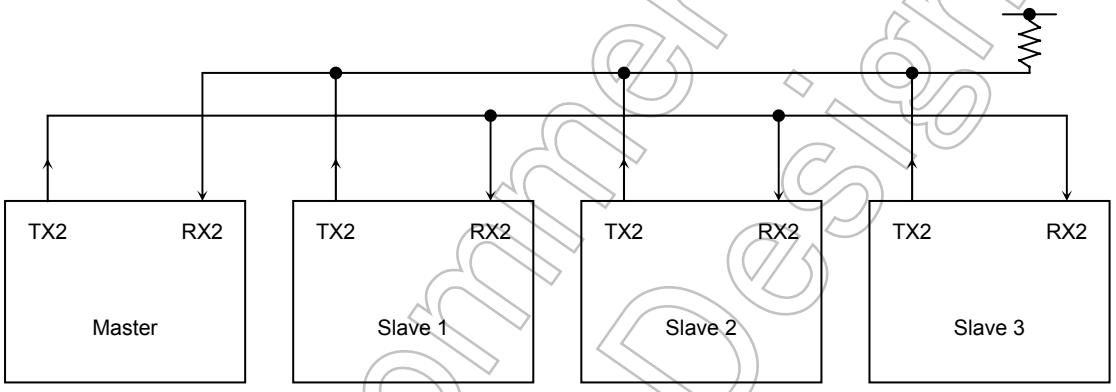
12.5.4 9-Bit UART Mode

Setting the SM field in the SC2MOD0 to 11 puts the SIO2 in 9-Bit UART mode. In this mode, the parity bit cannot be used and must be disabled by clearing the SC2CR.PE bit to 0.

For transmit operation, the most-significant bit (9th bit) is stored in the TB8 bit in the SC2MOD0. For receive operation, the most-significant bit is stored in the RB8 bit in the SC2CR. Reads and writes from and to the transmit and receive buffers must be done with the most-significant bit first, followed by the SC2BUF. The SBLEN bit in the SC2MOD2 specifies the number of stop bits.

Wake-up feature

In 9-Bit UART mode, the wake-up feature can be enabled for slave controllers by setting the WU bit in the SC2MOD0 to 1. When this feature is enabled, the INTRX2 interrupt is generated only when SC2CR.RB8 = 1.



Note: The TX2 pin of a slave controller must be configured as an open-drain output by programming the Port 8 Open-Drain Control Register (P8ODCR).

Figure 12.5.18 Serial Link Using the Wake-Up Feature

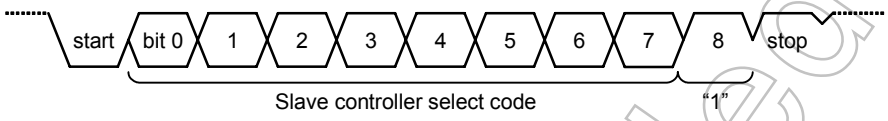
Not Recommended for New Design

Protocol

(1) Put all the master and slave controllers in 9-Bit UART mode.

(2) Enable the receiver in each slave controller by setting the SC2MOD0.WU bit to 1.

(3) The master controller transmits an 8-bit address frame (i.e., select code) that identifies a slave controller. The most-significant bit (TB8) of an address frame is a 1.

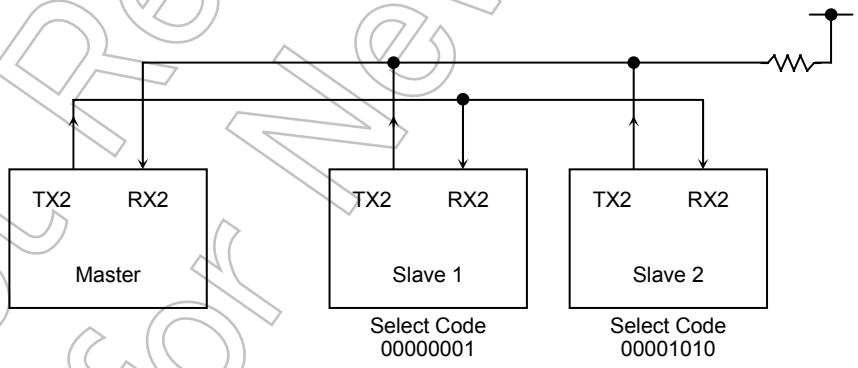


(4) Each slave controller compares the received address to its station address and clears the WU bit to 0 if they match.

(5) The master controller transmits a block of data to the selected salve controller (with SC2MOD.WU = 0). The most-significant bit (TB8) of a data frame is a 0.

(6) Slave controllers not addressed (with SC2MOD.WU = 1) continue to monitor the data stream but discard any frames with the most-significant bit (RB8) cleared to 0. Thus, the receive-done interrupt (INTRX2) is not generated. The addressed slave controller (with SC2MOD.WU = 0) can transmit data to the mater controller to notify that is has successfully received the message.

Example: Connecting a master controller with two slave controllers through a serial link using the IMCLK/2 clock as a serial clock



Not Recommended for New Design

3) Master controller setings

Main routine

```
P8IER   ← – 1 0 – – – – –   } Configure the P86 pin as TX2 and the P85 pin as RX2.
P8CR    ← – 1 0 – – – – –   }
P8FR1   ← – 1 1 – – – – –   }
IMR53   ← – 1 0 0 – 1 0 1   Enable INTRX2 and set its interrupt level to 5.
IMR52   ← – 1 0 0 – 1 0 0   Enable INTTX2 and set its interrupt level to 4.
SC2MOD0 ← 1 0 1 0 1 1 1 0   Select 9-Bit UART mode and select IMCLK as a serial clock.
SC2BUF  ← 0 0 0 0 0 0 0 1   Load the select code for slave 1.
```

Interrupt routine (INTTX2)

```
ICLR    ← 0 1 1 0 1 0 0 0 0   Clear the interrupt request.
SC2MOD0 ← 0 – – – – – – –     Clear the TB8 bit to 0.
SC2BUF  ← * * * * * * * *     Load transmit data.
    End of interrupt processing
```

4) Slave controller settings

Main routine

```
P8IER   ← – 1 0 – – – – –   }
P8CR    ← – 1 0 – – – – –   } Set the P86 pin as TX2 (open-drain output) and the P85 pin as RX2.
P8FR1   ← – 1 1 – – – – –   }
P8ODCR  ← – 1 – – – – – –   }
IMR53   ← – – 1 1 0 1 1 0   Enable INTTX2 and INTRX2.
IMR52   ← – – 1 1 0 1 0 1   }
SC2MOD0 ← 0 0 1 1 1 1 1 0   Select 9-Bit UART mode, select IMCLK as a serial clock and set the
                            WU bit to 1.
```

Interrupt routine (INTRX2)

```
ICLR    ← 0 1 1 0 1 0 1 0 0   Clear the interrupt request.
Reg.    ← SC2CR AND 0x1C      }
if Reg. ≠ 0 then error        } Check for errors.
Reg.    ← SC2BUF              }
if  Reg. = Select code        }
Then                          }
SC2MOD0 ← – – – 0 – – – –     Clear the WU bit to 0.
```

Not Recommended for New Design

13. Analog-to-Digital Converters (ADCs)

The TMP19A71 contains two 10-bit successive-approximation analog-to-digital converters (ADCs). Both ADCs have two modes; Normal mode and PMD mode. While Normal mode supports typical AD conversion with two 8-channel inputs, PMD mode is specifically designed for AD conversion for motor control. In PMD mode, the ADCs have 8-channel and 11-channel inputs. The two ADCs can be programmed to operate independently, and the operating mode can be separately selected for each of the ADCs.

13.1 Features

13.1.1 Normal Mode

(1) Two 8-channel, 10-bit AD converters are available. Each channel has a separate conversion result register.
(2) The two AD converters can be independently programmed for Fixed-Channel or Channel Scan mode.
(3) The two AD converters can be independently programmed for Single Conversion or Continuous Conversion mode.
(4) The INTAD0/1 interrupt is generated upon completion of a conversion. The interrupt interval is selectable.
(5) Setting register enables starting of an AD conversion under the following conditions:
- TMRB interrupt (INTTB1)
- External trigger input (ADTRG0/1)
- Software trigger (ADSFT0)

(6) The highest-priority conversion can interrupt the ongoing conversion in Channel Scan and Fixed-Channel Continuous Conversion modes (The highest-priority conversion can only initiated by software).
(7) The INTADHP0/1 interrupt is generated upon completion of the highest-priority conversion.
(8) AD conversions can be monitored via the Busy and Overrun flags.
(9) In Channel Scan Continuous Conversion mode, the interval between conversions can be selected.
(10) The conversion result can be compared to the two compare registers. The user can select whether or not to generate an interrupt when the conversion result equals the compare register.

13.1.2 PMD Mode

(1) Two 10-bit AD converters are available. One has 8 conversion result registers, and the other has 11 conversion result registers.
(2) The conversion enable, input channel and PMD timing trigger can be programmed independently for each conversion result register.
(3) Conversions are started in ascending order from the smallest-numbered enabled conversion result register.
(4) The conversion interval can be increased by a maximum of 255 times the PMD trigger interval.

Not Recommended for New Design

13.2 Register Description

Each of the two ADCs contains a group of registers for both Normal mode and PMD mode as shown in Table 13.2.1.

Table 13.2.1 ADC Register Map (1/3)

Normal Mode (ADC0)

Address	Bits	Mnemonic	Register Name
0xFFFFC900	16	ADNRES0	AD Normal Mode Result Register 0
0xFFFFC904	16	ADNRES1	AD Normal Mode Result Register 1
0xFFFFC908	16	ADNRES2	AD Normal Mode Result Register 2
0xFFFFC90C	16	ADNRES3	AD Normal Mode Result Register 3
0xFFFFC910	16	ADNRES4	AD Normal Mode Result Register 4
0xFFFFC914	16	ADNRES5	AD Normal Mode Result Register 5
0xFFFFC918	16	ADNRES6	AD Normal Mode Result Register 6
0xFFFFC91C	16	ADNRES7	AD Normal Mode Result Register 7
0xFFFFC920	16	ADCHPR0	Highest-Priority Conversion Result Register (ADC0)
0xFFFFC924	16(8)	ADNMOD0 (L)	AD Normal Mode Control Register (Low) (ADC0)
0xFFFFC925	8	ADNMOD0H	AD Normal Mode Control Register High (ADC0)
0xFFFFC928	8	ADNCLK0	AD Normal Mode Clock Control Register (ADC0)
0xFFFFC92C	16(8)	CMPCTL0 (L)	AD Monitor Control Register (Low) (ADC0)
0xFFFFC92C	8	CMPCTL0H	AD Monitor Control Register High (ADC0)
0xFFFFC930	8	ADCHPC0	Highest-Priority Conversion Control Register (ADC0)
0xFFFFC934	16	ADCMP00	AD Compare Register 0(ADC0)
0xFFFFC938	16	ADCMP01	AD Compare Register 1 (ADC0)
0xFFFFC93C	16	ADCBASN0	AD Normal Mode Basic Setting Register (ADC0)
0xFFFFC940	8	ADCSTART0	AD Software Start Register (ADC0)

Not Recommended for New Design

Table 13.2.2 ADC Register Map (2/3)

Normal Mode (ADC1)

Address	Bits	Mnemonic	Register Name
0xFFFFC980	16	ADNRES8	AD Normal Mode Result Register 8
0xFFFFC984	16	ADNRES9	AD Normal Mode Result Register 9
0xFFFFC988	16	ADNRES10	AD Normal Mode Result Register 10
0xFFFFC98C	16	ADNRES11	AD Normal Mode Result Register 11
0xFFFFC990	16	ADNRES12	AD Normal Mode Result Register 12
0xFFFFC994	16	ADNRES13	AD Normal Mode Result Register 13
0xFFFFC998	16	ADNRES14	AD Normal Mode Result Register 14
0xFFFFC99C	16	ADNRES15	AD Normal Mode Result Register 15
0xFFFFC9A0	16	ADCHPR1	Highest-Priority Conversion Result Register (ADC1)
0xFFFFC9A4	16(8)	ADNMOD1 (L)	AD Normal Mode Control Register (Low) (ADC1)
0xFFFFC9A5	8	ADNMOD1H	AD Normal Mode Control Register High (ADC1)
0xFFFFC9A8	8	ADNCLK1	AD Normal Mode Clock Control Register (ADC1)
0xFFFFC9AC	16(8)	CMPCTL1 (L)	AD Monitor Control Register (Low) (ADC1)
0xFFFFC9AC	8	CMPCTL1H	AD Monitor Control Register High (ADC1)
0xFFFFC9B0	8	ADCHPC1	Highest-Priority Conversion Control Register (ADC1)
0xFFFFC9B4	16	ADCMP10	AD Compare Register 0(ADC1)
0xFFFFC9B8	16	ADCMP11	AD Compare Register 1 (ADC1)
0xFFFFC9BC	16	ADCBASN1	A/D Normal Mode Basic Setting Register (ADC1)
0xFFFFC9C0	8	ADCSTART1	AD Software Start Register (ADC1)

PMD Mode (ADC0)

Address	Bits	Mnemonic	Register Name
0xFFFFCD00	16	ADPRES0	AD PMD Mode Result Register 0
0xFFFFCD04	16	ADPRES1	AD PMD Mode Result Register 1
0xFFFFCD08	16	ADPRES2	AD PMD Mode Result Register 2
0xFFFFCD0C	16	ADPRES3	AD PMD Mode Result Register 3
0xFFFFCD10	16	ADPRES4	AD PMD Mode Result Register 4
0xFFFFCD14	16	ADPRES5	AD PMD Mode Result Register 5
0xFFFFCD18	16	ADPRES6	AD PMD Mode Result Register 6
0xFFFFCD1C	16	ADPRES7	AD PMD Mode Result Register 7
0xFFFFCD40	16(8)	ADCSETT00 (L)	AD Input Timing Trigger Register 0 (Low) (ADC0)
0xFFFFCD41	8	ADCSETT00H	AD Input Timing Trigger Register 0 High (ADC0)
0xFFFFCD48	16(8)	ADCSET00 (L)	AD Input Port Select Register 0 (Low) (ADC0)
0xFFFFCD49	8	ADCSET00H	AD Input Port Select Register 0 High (ADC0)
0xFFFFCD4C	16(8)	ADCSET01 (L)	AD Input Port Select Register 1 (Low) (ADC0)
0xFFFFCD4D	8	ADCSET01H	AD Input Port Select Register 1 High (ADC0)
0cFFFFCD58	8	ADPCLK0	AD PMD Mode Clock Control Register (ADC0)
0xFFFFCD5C	8	ADPMOD00	AD PMD Mode Control Register 0 (ADC0)
0xFFFFCD60	16(8)	ADPMOD01 (L)	AD PMD Mode Control Register 1 (Low) (ADC0)
0xFFFFCD61	8	ADPMOD01H	AD PMD Mode Control Register 1 High (ADC0)
0xFFFFCD64	16(8)	ADCNE0 (L)	A/D Count Enable Register (Low) (ADC0)
0xFFFFCD65	8	ADCNE0H	A/D Count Enable Register High (ADC0)
0xFFFFCD68	8	ADCNT0	A/D Conversion Count Setting Register (ADC0)
0xFFFFCD6C	16	ADCBASP0	A/DPMD Mode Basic Setting Register (ADC0)
0xFFFFCD70	8	ADMODSEL0	AD Mode Control Register (ADC0)

Not Recommended for New Design

Table 13.2.3 ADC Register Map(3/3)

PMD Mode (ADC1)

Address	Bits	Mnemonic	Register Name
0xFFFFCD80	16	ADPRES8	AD PMD Mode Result Register 8
0xFFFFCD84	16	ADPRES9	AD PMD Mode Result Register 9
0xFFFFCD88	16	ADPRES10	AD PMD Mode Result Register 10
0xFFFFCD8C	16	ADPRES11	AD PMD Mode Result Register 11
0xFFFFCD90	16	ADPRES12	AD PMD Mode Result Register 12
0xFFFFCD94	16	ADPRES13	AD PMD Mode Result Register 13
0xFFFFCD98	16	ADPRES14	AD PMD Mode Result Register 14
0xFFFFCD9C	16	ADPRES15	AD PMD Mode Result Register 15
0xFFFFCDA0	16	ADPRES16	AD PMD Mode Result Register 16
0xFFFFCDA4	16	ADPRES17	AD PMD Mode Result Register 17
0xFFFFCDA8	16	ADPRES18	AD PMD Mode Result Register 18
0xFFFFCDC0	16(8)	ADCSETT10 (L)	AD Input Timing Trigger Register 0 (Low) (ADC1)
0xFFFFCDC1	8	ADCSETT10H	AD Input Timing Trigger Register 0 High (ADC1)
0xFFFFCDC4	8	ADCSETT11	AD Input Timing Trigger Register 1 (ADC1)
0xFFFFCDC8	16(8)	ADCSET10 (L)	AD Input Port Select Register 0 (Low) (ADC1)
0xFFFFCDC9	8	ADCSET10H	AD Input Port Select Register 0 High (ADC1)
0xFFFFCDCC	16(8)	ADCSET11 (L)	AD Input Port Select Register 1 (Low) (ADC1)
0xFFFFCDCD	8	ADCSET11H	AD Input Port Select Register 1 High (ADC1)
0xFFFFCDD0	16(8)	ADCSET12 (L)	AD Input Port Select Register 2 (Low) (ADC1)
0xFFFFCDD1	8	ADCSET12H	AD Input Port Select Register 2 High (ADC1)
0xFFFFCDD8	8	ADPCLK1	AD PMD Mode Clock Control Register (ADC1)
0xFFFFCDDC	8	ADPMOD10	AD PMD Mode Control Register 0 (ADC1)
0xFFFFCDE0	16(8)	ADPMOD11 (L)	AD PMD Mode Control Register 1 (Low) (ADC1)
0xFFFFCDE1	8	ADPMOD11H	AD PMD Mode Control Register 1 High (ADC1)
0xFFFFCDE4	16(8)	ADCNE1 (L)	A/D Counting Conversion Enable Register (Low) (ADC1)
0xFFFFCDE5	8	ADCNE1H	A/D Counting Conversion Enable Register High (ADC1)
0xFFFFCDE8	8	ADCNT1	A/D Conversion Count Setting Register (ADC1)
0xFFFFCDEC	16	ADCBASP1	A/DPMD Mode Basic Setting Register (ADC1)
0xFFFFCDF0	8	ADMODSEL1	AD Mode Control Register (ADC1)

Note 1: Some of 16-bit ADC registers is accessible in 8-bit system by dividing higher 8bits and lower 8 bits. i.e. ADNMOD0 becomes accessible in 8-bit system by making it be ADNMOD0L/ADNMOD0H.

Not Recommended for New Design

The MODSEL bit in the ADMODSEL0 register is used to select either Normal mode (MODSEL = 0) or PMD mode (MODSEL = 1). Normal mode provides the AD monitor and highest-priority conversion features. PMD mode is synchronous to the trigger inputs from a programmable motor driver (PMD). In Normal mode, the two ADCs are basically functionally equivalent; in the following description any references to ADC0 also apply to ADC1.

AD Mode Control Register (ADC0)

ADMODSEL0 (0xFFFF_CD70)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	—	VREFON	MODSEL
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function							VREF control 0: OFF 1: ON	ADC conversion mode 0: Normal mode 1: PMD mode

Note 1: The MDOSEL bit must not be changed during an AD conversion. If it is changed, operation cannot be guaranteed.

Note 2: Registers other than those for the selected conversion mode must not be programmed. Before programming registers, the MODSEL bit must be programmed to select Normal or PMD mode.

Note 3: The VREFON bit must be set 3 us before an AD conversion is started to ensure the stable internal reference voltage. If an AD conversion is started with VREFON= 0 or before the internal reference voltage has stabilized, conversion accuracy cannot be guaranteed.

Note 4: The VREFON bit is automatically set to 1 after an AD conversion is started. However, conversion accuracy cannot be guaranteed until the reference voltage has stabilized (see Note 3).

Not Recommended for New Design

13.3 Normal Mode (ADMODSEL0.MODSEL=0)

AD Normal Mode Control Register (Low) (ADC0)

ADNMOD0(L) (0xFFFF_C924)

	7	6	3	4	3	2	1	0
Bit Symbol	—	ADCH			LAT	ITM	REP	SCAN
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	Must be set to 0.	Analog input channel select			Latency 0: None 1: Wait until the result register is read.	Interrupt in Fixed-Channel Continuous Conversion mode	Continuous conversion mode 0: Single 1: Continu-ous	Channel scan mode 0:Fixed-Channel 1: Channel Scan

Analog Input Channel Select

SCAN / ADCH0 [2:0]	0 Fixed-Channel Mode	1 Channel Scan Mode
000	AIN0	AIN0
001	AIN1	AIN0 to AIN1
010	AIN2	AIN0 to AIN2
011	AIN3	AIN0 to AIN3
100	AIN4	AIN0 to AIN4
101	AIN5	AIN0 to AIN5
110	AIN6	AIN0 to AIN6
111	AIN7	AIN0 to AIN7

Interrupt in Fixed-Channel Continuous Conversion Mode

	Fixed-Channel Continuous Conversion Mode SCAN=0, REP=1
0	Generate an interrupt when a single conversion has been completed.
1	Generate an interrupt when a sequence of four conversions has been completed.

In ADC1, the input channels correspond as follows:
AIN0→AIN8
AIN1→AIN9
AIN2→AIN10
AIN3→AIN11
AIN4→AIN12
AIN5→AIN13
AIN6→AIN14
AIN7→AIN15

Note 1: AIN7 pin (AIN15 pin for ADC1) may be used as ADTRG0 input pin. Therefore, when ADTRG0 is used in ADNMOD0<TSEL>= "10", do not set to ADNMOD0<ADCH>="111," and when ADTRG1 is used in ADNMOD1<TSEL>="11," do not set to ADNMOD1<ADCH>="111."

Note 2: ADNMOD0<LAT> setting becomes effective only when it is in Continuous Conversion mode. When ADNMOD 0<LAT>="1" is set, the next conversion does not start until the reading of the register stored at the end is finished.

For example, when ADNMOD0<ADCH>="101," ADNMOD0<LAT>="1," ADNMOD0<REP>="1," and ADNMOD0<SCAN>="1," the next conversion does not start until reading of the result for ADNRES5 after Channel Scan conversion has finished. When ADNRES5 is read prior to ADNRES0 – 4, the next conversion starts as ADNRES5 starts to be read. And, when ADNMOD0<ADCH>="101," ADNMOD0<LAT>="1," ADNMOD0<ITM>="1," ADNMOD0<REP>="1," and ADNMOD0<SCAN>="0," the next conversion does not start until the results are stored four times in ADNRES5 and are read out.

Not Recommended for New Design

AD Normal Mode Control Register (High) (ADC0)

(ADNMOD0H) (0xFFFF_C925)

	15	14	13	12	11	10	9	8
Bit Symbol	—	—	—	—	—	TRGE	TSEL	
Read/Write	R			R/W				
Reset Value	0	0	0	0	0	0	0	0
Function				Must be set to 0.	Must be set to 0.	Normal mode conversion trigger 0: Software 1: Hardware	Hardware trigger source 00: Reserved 01: INTTBCOM11 10: ADTRG0 11: ADTRG1	

Note 1: When <TRGE>="1" is set, too, it can be started up by software.

Note 2: ADC1 also can select INTTBCOM11 as a hardware starting source by setting ADNMOD1<TSEL>=01.

AD Software Start Register (ADC0)

ADCSTART0 (0xFFFF_C940)

	7	6	5	4	3	2	1	0
bit Symbol	BUSY	EOS	—	—	—	—	—	ADSFT
Read/Write	R	R/W	R					W
Reset Value	0	0	0	0	0	0	0	0
Function	Normal mode AD conversion busy flag 0: Idle 1: Busy	Conversion Complete flag 0: Don't care 1:Complet-ed Write a 0 to clear this bit.						AD conversion start 0:Don't care 1:Start This bit is always read as 0.

Note: The BUSY bit indicates whether or not an AD conversion is in progress. Use the EOS bit to check whether or not an AD conversion has completed.

A/D Normal Mode Basic Setting Register (ADC0)

ADCBASN0 (0xFFFF_C93C)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	AZSEL	—	—
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	Must be set to 0.	Must be set to 0.	Must be set to 0.	Must be set to 0.	Must be set to 0.	Sample Hold time 1: 6 clocks 0: 12 clocks	Must be set to 0.	Must be set to 0.
	15	14	13	12	11	10	9	8
Bit Symbol	—	—	—	—	—	—	—	—
Read/Write	R/W							
Reset Value	0	0	0	1	0	0	0	0
Function	Must be set to 0.	Must be set to 0.	Must be set to 0.	Must be set to 1.	Must be set to 0.	Must be set to 0.	Must be set to 0.	Must be set to 0.

Note: The time taken for the conversion of ADC is derived from the equation, (the number of clocks selected in <AZSEL>+27 clocks)/ADCLK.

Not Recommended for New Design

Highest-Priority Conversion Control Register (ADC0)

ADCHPC0 (0xFFFF_C930)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	HBSY	HPRQ	HPCH		
Read/Write	R				R/W			
Reset Value	0	0	0	0	0	0	0	0
Function				Highest-Priority AD conversion busy flag 0: Completed 1: Busy	Highest-Priority conversion request 0: Don't care 1: Start highest-priority conversion	Highest-priority channel select		

Highest-Priority Analog Input Channel Select

SCAN / ADCH [2:0]	0 Fixed-Channel Mode
000	AIN0
001	AIN1
010	AIN2
011	AIN3
100	AIN4
101	AIN5
110	AIN6
111	AIN7

In ADC1, the input channels correspond as follows:
AIN0→AIN8
AIN1→AIN9
AIN2→AIN10
AIN3→AIN11
AIN4→AIN12
AIN5→AIN13
AIN6→AIN14
AIN7→AIN15

Note: AIN7 pin (AIN15 pin for ADC1) may be used as ADTRG0 input pin. Therefore, when ADTRG0 is used in ADNMOD0<TSEL>= "10," do not set to ADNMOD0<ADCH>="111," and when ADTRG1 is used in ADNMOD1<TSEL>="11," do not set to ADNMOD1<ADCH>="111."

Not Recommended for New Design

There are 16 conversion result registers numbered from 0 to 15, which are all identical. The following is a description of register 0.

AD Normal Mode Conversion Result Register 0

ADNRES0 (0xFFFF_C91C)

	7	6	5	4	3	2	1	0
Bit Symbol	ADR							
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0
Function	Lower 8 bits of an AD conversion result							
	15	14	13	12	11	10	9	8
Bit Symbol	VAL	OVR	—	—	—	—	ADR	
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0
Function	Conversion result store flag 1: Stored	Overrun flag 0: No overrun 1: Overrun					Upper 2 bits of an AD conversion result	

Note 1: This register must be accessed as a 16-bit or larger quantity. If it is accessed as an 8-bit quantity, operation cannot be guaranteed.

Note 2: Bit 15 is an AD conversion result flag ADNRES0<VAL>. 1 is set to this when AD conversion value is stored, and it is cleared to 0 when the ADNRES0 is read.

Note 3: Bit 14 is an Overrun flag ADNRES0<OVR>. 1 is set to this when it is overwritten before reading the conversion result register ADNRES0. This bit is cleared to 0 when a new conversion result is stored in ADNRES0 with VAL=0.

Note 4: This register does not support bit manipulation instructions.

Highest-Priority Conversion Result Register (ADC0)

ADCHPR0 (0xFFFF_C9020)

	7	6	5	4	3	2	1	0
Bit Symbol	ADR							
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0
Function	Lower 8 bits of an AD conversion result							
	15	14	13	12	11	10	9	8
Bit Symbol	VAL	OVR	—	—	—	—	ADR	
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0
Function	Conversion result store flag 1: Stored	Overrun flag 0: No overrun 1: Overrun					Upper 2 bits of an AD conversion result	

Note 1: This register must be accessed as a 16-bit or larger quantity. If it is accessed as an 8-bit quantity, operation cannot be guaranteed.

Note 2: Bit 15 is an AD conversion result flag ADNRES0<VAL>. 1 is set to this when AD conversion value is stored, and it is cleared to 0 when the ADNRES0 is read.

Note 3: Bit 14 is an Overrun flag ADNRES0<OVR>. 1 is set to this when it is overwritten before reading the conversion result register ADNRES0. This bit is cleared to 0 when a new conversion result is stored in ADNRES0 with VAL=0.

Note 4: This register does not support bit manipulation instructions.

Not Recommended for New Design

AD Normal Mode Clock Control Register 1

ADNCLK0 (0xFFFF_C928)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	ADCCK		
Read/Write	R					R/W		
Reset Value	0	0	0	0	0	0	0	0
Function						Prescaler clock output select 001: IMCLK/2 010: IMCLK/4 011: IMCLK/8 100: IMCLK/16 Other: IMCLK		

Note 1: AD conversion is performed at the clock frequency selected in this register. To assure conversion accuracy, however, the conversion clock frequency must be 14 MHz or slower (which results in a conversion time of 2.36 μs or longer with 6-clock sample hold).

Note 2: The conversion clock must not be changed while AD conversion is in progress. Wait at least 2 ADCLK clocks after AD conversion has completed before changing the conversion clock.

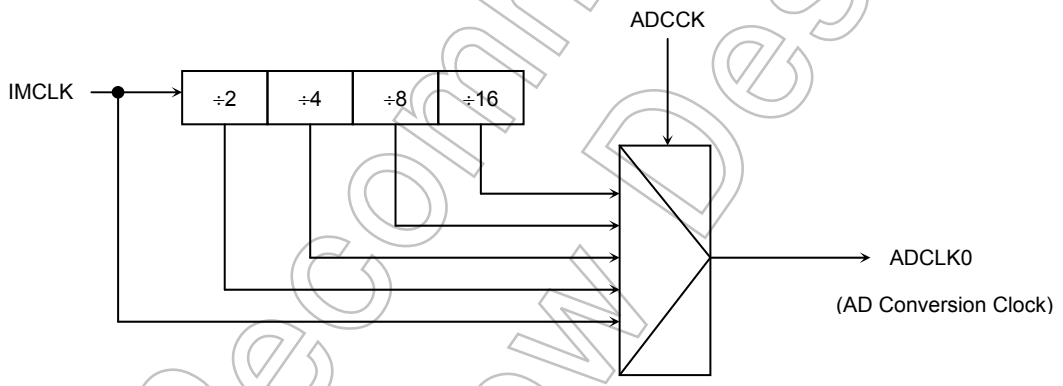


Figure 13.3.1 Clock Control Circuit

Not Recommended for New Design

A/D Monitor Control Register (ADC0)

CMPCTL0(L)
(0xFFFF_C92C)

	7	6	5	4	3	2	1	0
Bt Symbol	—	—	CMCH0			CMOP0	IRQEN0	CMCAP0
Read/Write	R		R/W					
Reset Value	0	0	0	0	0	0	0	0
Function			AD input channel 0 to be compared			A/D Monitor Interrupt setting is 0. 0:Under Compare Register 1:More than Compare Register	A/D Monitor Interrupt setting is 0. 0: disable 1: enable	A/D Monitor Interrupt flag is 0. 0:Monitoring Non-generated interrupt 1:Monitoring Generated Interrupt

CNCH	AD Input Channel To Be Compared
000	AIN0
001	AIN1
010	AIN2
011	AIN3
100	AIN4
101	AIN5
110	AIN6
111	AIN7

In ADC1, the input channels correspond as follows:
AIN0→AIN8
AIN1→AIN9
AIN2→AIN10
AIN3→AIN11
AIN4→AIN12
AIN5→AIN13
AIN6→AIN14
AIN7→AIN15

AD Monitor Control Register (ADC0)

(CMPCTL0H)
(0xFFFF_C92D)

	15	14	13	12	11	10	9	8
Bit Symbol	—	—	CMCH1			CMOP1	IRQEN1	CMCAP1
Read/Write	R		R/W					
Reset Value	0	0	0	0	0	0	0	0
Function			AD input channel 1 to be compared			A/D Monitor Interrupt setting is 1. 0:Under Compare Register 1:More than Compare Register	A/D Monitor Interrupt setting is 1. 0: disable 1: enable	A/D Monitor Interrupt flag is 1. 0:Monitoring Non-generated interrupt 1:Monitoring Generated Interrupt

CNCH	AD Input Channel To Be Compared
000	AIN0
001	AIN1
010	AIN2
011	AIN3
100	AIN4
101	AIN5
110	AIN6
111	AIN7

In ADC1, the input channels correspond as follows:
AIN0→AIN8
AIN1→AIN9
AIN2→AIN10
AIN3→AIN11
AIN4→AIN12
AIN5→AIN13
AIN6→AIN14
AIN7→AIN15

Not Recommended for New Design

Note: CMCAPx is cleared to 0 by writing data in "1" or in ADCMPxx. Because interrupt requests are continuously sent until this register is cleared, it must be cleared within the Monitor Interrupt routine.

Not Recommended for New Design

A/D Conversion Result Compare Register (ADC0)

ADCMP00 (0xFFFF_C934)

	7	6	5	4	3	2	1	0
Bit Symbol	ADR0							
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	A/D conversion result compare value 0 is stored.							
	15	14	13	12	11	10	9	8
Bit Symbol	—	—	—	—	—	—	ADR0	
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0
Function								

A/D Conversion Result Compare Register (ADC0)

ADCMP01 (0xFFFF_C938)

	7	6	5	4	3	2	1	0
Bit Symbol	ADR1							
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	A/D conversion result compare value 1 is stored.							
	15	14	13	12	11	10	9	8
Bit Symbol	—	—	—	—	—	—	ADR1	
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0
Function								

Not Recommended for New Design

13.4.1 Operation (Normal Mode)

13.4.1.1 Analog Reference Voltage

Clearing the VREFON bit in the ADMODSEL0 turns off the switch between the VREFH and FREFL pins. Once the VREFON bit is cleared, the internal reference voltage requires a recovery time of 3 μs to stabilize after the VREFON bit is again set to 1. Before starting an AD conversion, therefore, be sure to wait for 3 μs after setting the VREFON bit to 1. If an AD conversion is started before this stabilization period has elapsed, conversion accuracy cannot be guaranteed.

13.4.1.2 Selecting an Analog Input Channel(s)

Selection of the analog input channel(s) to be used varies according to the operating mode of the AD converter.

(1) Normal AD Conversion

- When an analog input is used in Channel Fixed mode (ADNMOD0<SCAN>="0")

  Among the analog input from AIN0 to AIN 7, select one channel according to the ADNMOD0<ADCH> setting.

- When an analog input is used in Channel Scan mode (ADNMOD0<SCAN>="1")

  Select one Scan mode among the eight types Scan modes according to the ADNMOD0<ADCH> setting.

(2) Highest Priority AD Conversion

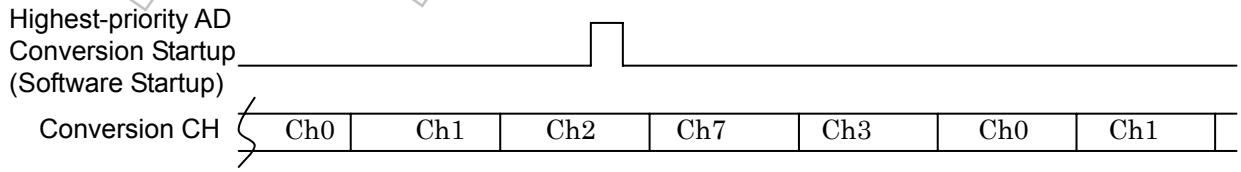
Among the analog input from AIN0 to AIN 7, select one channel according to the A/DCHPC0<HPCH> setting.

After resetting, ADNMOD0<SCAN> is initialized to 0, and ADNMOD0<ADCH>to 0000, which makes the selection be in processing, and Channel Fixed mode input of AIN0 pin is selected. Note that pins not used as analog input channels can be used as normal ports (a part of them are for input only), however, the conversion accuracy may get worse.

When the highest-priority AD conversion is started during the normal AD conversion, the highest-priority one starts in a break, the normal AD conversion resumes at the end of the highest-priority one is over.

Example:

When the highest-priority conversion of AIN7 is started in ADCHP0<HPCH>=111 during the Continuous Scan conversion of the Channel from AIN0 to AIN3 in ADNMOD0<REP:SCAN>=11 and ADNMOD0<ADCH>=00011.



Not Recommended for New Design

13.3.1.1 Starting of AD Conversion

AD conversion has two types; Normal AD conversion and Highest-Priority AD conversion. A Normal AD conversion starts up in software by setting 1 to ADCSTART0<ADSFT>. Also, the Highest-Priority conversion starts up in software by setting 1 to ADCHPC0<HPRQ>. A For Normal AD conversion, one operating mode among the four operating modes specified by ADNMOD0<REP:SCAN>. An operating mode of the Highest-Priority conversion is Channel-Fixed Single conversion only. A Normal AD conversion can start up with hardware starting source selected by ADNMOD0<TSEL> by setting 1 to ADNMOD0<TRGE>. When this bit is "10/11," a Normal AD conversion starts up at the rising edge of ADTRG0 pin, and when "01," it starts up with INTTBCOM11 of the timer from 16-bit quantity. It can start in software even if the startup in hardware becomes enabled.

When a Normal AD conversion starts, 1 is set to an AD conversion Busy flag (ADCSTART0<BUSY>) indicating that the conversion is in progress. When the Highest-Priority AD conversion starts, 1 is set to an AD conversion Busy flag (ADCHPC0<HBSY>). At this time, a Busy flag for Normal AD conversion retains the value before a starting of the Highest-Priority AD conversion. A conversion end flag ADCSTART0<EOS> for Normal AD conversion, also, retains the value before a starting of the Highest-Priority AD conversion. Since ADCSTART0<BUSY> is a flag showing the conversion operation, it has an interval of being 0 between the conversions such as those in the Continuous conversion mode. When palling the end of conversion, ADCSTART0<EOS> must not be used.

When 1 is set to ADCHPC0<HPRQ> during a Normal AD conversion, the Highest-Priority AD conversion starts upon the storage of result register of the ongoing conversion, and AD conversion (Channel-Fixed Single conversion) of the cannel specified by ADCHPC0<HPCH> starts. When this result is stored in the result register ADCHPR0, a Normal AD conversion resumes operation from the part suspended.

13.3.1.2 Restart

A Normal AD conversion restarts when 1 is set to ADCSTART0<ADSFT0> during the Channel-Fixed Normal Conversion, or it is started with hardware source. At the time of restarting, a Normal AD conversion performed till then starts conversion after a lapse of conversion time, however, the result that has been converted at the moment of restarting is not stored. Restarting does clear neither the flags of <OVRx> nor <VALx>.

Note 1: When Continuous conversion is in process, stop it first (ADNMOD0<REP>=0) and restart after all of the conversions ended.

Note 2: When Channel Scan mode conversion is in process, restart after all of the conversions ended.

13.3.1.3 Stop Repeat

Changing the ADNMOD0 bit in <REP> from 1 to 0 enables the stop of repeating after the Continuous conversion made one-cycle repeat. In Channel-Fixed Continuous conversion mode (interrupts after four conversions), as an interrupt generates after conversion is performed four times, the Continuous conversion stops. In Channel Scan Continuous conversion mode, after the conversions performed as much as specified number of channels, the Continuous conversion stops as an interrupt is generated.

Not Recommended for New Design

13.3.1.4 AD Conversion Mode and Interrupt in the End of AD Conversion

The normal mode has the four operating modes shown in Table 13.3.1. Normal AD Conversion can select a mode according to the ADNMOD0<REP:SCAN> setting while the Highest-Priority AD conversion can select only Channel-Fixed single conversion regardless of the ADNMOD0<REP:SCAN> setting.

Table 13.3.1 Relations of AD Conversion Mode, Interrupt Generation Timing, and Flag Behavior

Conversion Mode	Interrupt Generation Timing	EOS Set Timing (Note 1)	BUSY (after an interrupt has generated)	ADNMOD0		
				ITM	REP	SCAN
Fixed-Channel Single Conversion	After a conversion ends	After a conversion ends	0	———	0	0
Channel Scan Single Conversion	After a Scan conversion ends	After a Scan conversion ends	0	———	0	1
Fixed-Channel Continuous Conversion	Every time of conversion	After a conversion ends.	1	0	1	0
	Every fur times of conversion	After a conversion ends four times	1	1		
Channel Scan Continuous Conversion	Every time of Scan conversion	After a Scan conversion ends.	1	———	1	1

Note 1: Write 0 and clear EOS.

(1) Normal AD Conversion

ADNMOD0<REP:SCAN> selects an operating mode. As an AD conversion starts, 1 is set to ADCSTART0<BUSY>. After a specified AD conversion ends, 1 is set to ADCSTART0<EOS> that indicates the AD conversion ended, and then an AD conversion end interrupt (INTAD0) generates. <BUSY> is cleared to 0 as <EOS> is set when <REP>="0." There are timings to be 0 at the intervals of each channel conversion when <REP>="1."

a) Fixed-Channel Single Conversion Mode

This mode is selected by programming the REP and SCAN bits in the ADNMOD0 register to 00. In this mode, the ADC performs a single conversion on a single selected channel. When a conversion is completed, the ADC sets 1 to the ADCSTART0.EOS bit, clears the ADCSTART0.BUSY bit in 0 and generates the INTAD0 interrupt. The EOS bit must be cleared by writing 0.

b) Channel Scan Single Conversion Mode

This mode is selected by programming the REP and SCAN bits in the ADNMOD0 register to 01. In this mode, the ADC performs a single conversion on each selected group of channels. When a single conversion sequence is completed, the ADC sets 1 to the ADCSTART0.EOS bit, clears the ADCSTART0.BUSY bit in 0 and generates the INTAD0 interrupt. The EOS bit must be cleared by writing 0.

Not Recommended for New Design

c) Fixed-Channel Continuous Conversion Mode

This mode is selected by programming the REP and SCAN bits in the ADNMOD0 register to 10. In this mode, the ADC repeatedly converts a single selected channel. When a conversion process is completed, the ADC sets 1 to the ADCSTART0.EOS bit. A generation timing of an interrupt request is selectable according to the ADNMOD0<ITM> setting. The setting timing of EOS is associated with the timing of an interrupt.

The EOS bit must be cleared by writing 0.

When ITM=0, the ADC generates an interrupt request every time a conversion ends. In this case, the conversion result is stored in the corresponding conversion result register to a selected channel, which makes 1 be set to the EOS bit.

When ITM=1, the ADC generates an interrupt after every four conversions completed. The conversion result is stored in the corresponding conversion result register to a selected channel. After the result of the fourth conversion is stored, 1 is set to the EOS bit. And then, a conversion starts again. The EOS bit must be cleared by writing 0.

ADNMOD0<LAT> setting can make the next conversion of the conversion in Continuous conversion mode wait until the result register is read.

When ITM=0, the time taken until a conversion starts after a previous conversion ends is controlled, and when ITM=1, the time taken until a conversion starts after previous conversion ends four times is controlled.

d) Channel Scan Continuous Conversion Mode

This mode is selected by programming the REP and SCAN bits in the ADNMOD0 register to 11. In this mode, the ADC repeatedly converts the selected group of channels. Every time a Scan conversion ends, 1 is set to ADNMOD0<EOS>, and an interrupt request of INTAD0 generates. ADNMOD0<BUSY> has a timing that becomes 0 at the interval of each channel conversion. The EOS bit must be cleared by writing 0.

To stop the operation of conversion of Continuous conversion modes, described in c) and d), 0 shall be written to ADNMOD0<REP>. The mode ends as an ongoing conversion ends, and ADNMOD0<BUSY> is cleared to 0.

Stop AD conversions and set 0 to ADMODSEL0<VREFON> before transferring to a Stop mode. The electricity is carried even it is in the Stop mode unless the transfer is made stopping a conversion. If the transfer is made with an AD conversion in performing, the result come out after the releasing of the Stop mode is not guaranteed.

(2) The Highest-Priority AD Conversion

ADNMOD0<REP, SCAN> setting has no effect on the Highest-Priority AD conversion. Its operation mode is Channel-Fixed Single conversion mode only. When the starting condition is met, the Highest-Priority AD conversion of a specified channel in ADCHPC0<HPCH> is performed only one time. As the conversion ends, an interrupt of the Highest-Priority AD conversion end generates, and ADCHPC0<HBSY> is cleared to 0.

Not Recommended for New Design

13.3.1.5 Highest-Priority Conversion Mode

The Highest-Priority AD conversion can interrupt a Normal AD conversion. The Highest-Priority AD conversion can start by setting 1 to ADCHPC0<HPRQ>. If the Highest-Priority AD conversion starts during the Normal AD conversion, AD conversion result during the conversion process is stored in a result register, and after that, a channel specified by ADCHPC0<HPCH>is single-converted. That result is stored in ADCHPR0, and the Highest-Priority AD conversion interrupt generates. Then, the Normal AD conversion resumes from the part continued from the previous time. If the Highest-Priority AD conversion restarts during the Highest-Priority AD conversion process, the conversion in process is finished, and then the Highest-Priority Conversion starts newly.

For example, if the Continuous conversion of the channels from AIN0 to AIN7 is active, and 1 is set to <HPRQ> while AIN3 is converted, the channel specified by <HPCH> is converted as soon as the AIN3 conversion ends, and the result is stored in ADCHPR0, and then the Continuous conversion restarts from AIN4.

13.3.1.6 AD Monitoring

Each AD converter has two AD monitoring functions and can compare a conversion value and two setting value simultaneously. When 1 is set to CMPCTL0<IREQEN0>, an AD monitoring is enabled. When the contents of a conversion result register specified by CMPCTL0<CMCH0> is more than or under the value of compare register (it is specified by <CMOP0>), AD monitoring interrupt (INTADM0) generates. CMPCTL0<CMCAP0> can determine which setting condition is met. Also, this comparing operates every time a result is stored in the relevant conversion result register, and as the condition is met, an interrupt generates. Note that since a register assigned to AD monitoring is not read in software in general, the overrun flag ADNRES0<OVR> and the conversion result flag ADNRES0<VAL> are always set. Therefore, to use the AD monitoring, do not use the flag of a relevant conversion result register.

13.3.1.7 AD Conversion Time

One AD conversion takes 27 clocks excluding sampling clocks. In ADCBASN0<AZSEL>,6 or 12 clocks can be selected as sampling clocks, thus the sum of AD conversion clocks may be 33 or 39. ADNCLK0<ADCCK> selects an AD conversion clocks among the AD pre-scaler output; IMCLK, IMCLK/2, IMCLK/4, IMCLK/8, and IMCLK/16.To assure the accuracy, it is necessary to set the AD conversion clock less than 14MHz, i.e. under 2.36μs (if Sample Hold is 6 clocks).

13.3.1.8 Storage and Read of AD Conversion Result

AD conversion results are stored in the result register of a Normal AD conversion (from ADNRES0 to ADNRES7). Correspondence of result registers and analog input channels are the same in any operating mode if they are in normal mode. For example, the result of AIN0 conversion is always stored in the ADNRES0 register.

Table 13.3.2 shows the correspondence of analog input channels and AD conversion result registers.

Not Recommended for New Design

Table 13.3.2 Analog Input Channel and AD Conversion Register

Analog Input Channel	Conversion Result Register
AIN0	ADNRES0
AIN1	ADNRES1
AIN2	ADNRES2
AIN3	ADNRES3
AIN4	ADNRES4
AIN5	ADNRES5
AIN6	ADNRES6
AIN7	ADNRES7

13.3.1.9 Data Polling

To process an AD conversion result by polling data without using any interrupt, ADNMOD0<EOS> is to be polled. When this flag is set, a conversion result is stored in a predetermined AD conversion result register. Therefore AD conversion result register must be read after checking the set. To detect an overrun at the time, the conversion result register must be read in 16-bit system. If the result were <OVR>=0 and <VAL>=1, a conversion result not overwritten would be gained.

Not Recommended for New Design

13.4.1 PMD Mode (MODSEL=1)

In PMD mode, the ADC performs AD conversions synchronous to a PMD trigger. The PMD trigger can be selected from three types: PMDTRG00 to PMDTRG02.

The ADC0 has 8 conversion result registers while the ADC1 has 11 conversion result registers. For each of these result registers, an analog input port and a PMD trigger can be programmed separately, and AD conversions can be enabled and disabled separately for each register.

Also, each of ADC unit has a counter, and the two cycles; every time and the specified number can be set to the counter. As all the programs enabled are converted completely, an ADC interrupt generates.

AD Input Timing Trigger Register (ADC0)

		7	6	5	4	3	2	1	0
ADCSETTOO(L) (0xFFFF_CD40)	Bit Symbol	ADST3		ADST2		ADST1		ADST0	
	Read/Write	R/W							
	Reset Value	0	0	0	0	0	0	0	0
	Function	Input timing trigger for result register 3 00: PMDTRG00 01: PMDTRG01 10: PMDTRG02 11: Reserved		Input timing trigger for result register 2 00: PMDTRG00 01: PMDTRG01 10: PMDTRG02 11: Reserved		Input timing trigger for result register 1 00: PMDTRG00 01: PMDTRG01 10: PMDTRG02 11: Reserved		Input timing trigger for result register 0 00: PMDTRG00 01: PMDTRG01 10: PMDTRG02 11: Reserved	

		15	14	13	12	11	10	9	8
(ADCSETTOOH) (0xFFFF_CD41)	Bit Symbol	ADST7		ADST6		ADST5		ADST4	
	Read/Write	R/W							
	Reset Value	0	0	0	0	0	0	0	0
	Function	Input timing trigger for result register 7 00: PMDTRG00 01: PMDTRG01 10: PMDTRG02 11: Reserved		Input timing trigger for result register 6 00: PMDTRG00 01: PMDTRG01 10: PMDTRG02 11: Reserved		Input timing trigger for result register 5 00: PMDTRG00 01: PMDTRG01 10: PMDTRG02 11: Reserved		Input timing trigger for result register 4 00: PMDTRG00 01: PMDTRG01 10: PMDTRG02 11: Reserved	

Not Recommended for New Design

AD Input Timing Trigger Register 0 (ADC1)

ADCSETT1O(L) (0xFFFF_CDC0)

	7	6	5	4	3	2	1	0
Bit Symbol	ADST11		ADST10		ADST9		ADST8	
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	Input timing trigger for result register 11 00: PMDTRG10 01: PMDTRG11 10: PMDTRG12 11: Reserved		Input timing trigger for result register 10 00: PMDTRG10 01: PMDTRG11 10: PMDTRG12 11: Reserved		Input timing triffer for result register 9 00: PMDTRG10 01: PMDTRG11 10: PMDTRG12 11: Reserved		Input timing triffer for result register 8 00: PMDTRG10 01: PMDTRG11 10: PMDTRG12 11: Reserved	

(ADCSETT1OH) (0xFFFF_CDC1)

	15	14	13	12	11	10	9	8
Bit Symbol	ADST15		ADST14		ADST13		ADST12	
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	Input timing trigger for result register 15 00: PMDTRG10 01: PMDTRG11 10: PMDTRG12 11: Reserved		Input timing trigger for result register 14 00: PMDTRG10 01: PMDTRG11 10: PMDTRG12 11: Reserved		Input timing trigger for result register 13 00: PMDTRG10 01: PMDTRG11 10: PMDTRG12 11: Reserved		Input timing triggr for result register 12 00: PMDTRG10 01: PMDTRG11 10: PMDTRG12 11: Reserved	

AD Input Timing Trigger Register 1(ADC1)

ADCSETT11 (0xFFFF_CDC4)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	ADST18		ADST17		ADST16	
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function			Input timing trigger for result register 10 00: PMDTRG10 01: PMDTRG11 10: PMDTRG12 11: Reserved		Input timing trigger for result register 9 00: PMDTRG10 01: PMDTRG11 10: PMDTRG12 11: Reserved		Input timing trigger for result register 8 00: PMDTRG10 01: PMDTRG11 10: PMDTRG12 11: Reserved	

Not Recommended for New Design

AD Input Port Select Register 0 (ADC0)

		7	6	5	4	3	2	1	0
ADCSET0O(L) (0xFFFF_CD48)	Bit Symbol	—	ADSI1			—	ADSI0		
	Read/Write	R/W							
	Reset Value	0	0	0	0	0	0	0	0
	Function	Must be written as 0.	Input port for result register 1 000: AIN0 001: AIN1 010: AIN2 011: AIN3 100: AIN4 101: AIN5 110: AIN6 111: AIN7			Must be written as 0.	Input port for result register 0 000: AIN0 001: AIN1 010: AIN2 011: AIN3 100: AIN4 101: AIN5 110: AIN6 111: AIN7		
		15	14	13	12	11	10	9	8
(ADCSET0OH) (0xFFFF_CD49)	Bit Symbol	—	ADSI3			—	ADSI2		
	Read/Write	R/W							
	Reset Value	0	0	0	0	0	0	0	0
	Function	Must be written as 0.	Input port for result register 3 000: AIN0 001: AIN1 010: AIN2 011: AIN3 100: AIN4 101: AIN5 110: AIN6 111: AIN7			Must be written as 0.	Input port for result register 2 000: AIN0 001: AIN1 010: AIN2 011: AIN3 100: AIN4 101: AIN5 110: AIN6 111: AIN7		

AD Input Port Select Register 1 (ADC0)

		7	6	5	4	3	2	1	0
ADCSET01(L) (0xFFFF_CD4C)	Bit Symbol	—	ADSI5			—	ADSI4		
	Read/Write	R/W							
	Reset Value	0	0	0	0	0	0	0	0
	Function	Must be written as 0.	Input port for result register 5. 000: AIN0 001: AIN1 010: AIN2 011: AIN3 100: AIN4 101: AIN5 110: AIN6 111: AIN7			Must be written as 0.	Input port for result register 4 000: AIN0 001: AIN1 010: AIN2 011: AIN3 100: AIN4 101: AIN5 110: AIN6 111: AIN7		
		15	14	13	12	11	10	9	8
(ADCSET01H) (0xFFFF_CD4D)	Bit Symbol	—	ADSI7			—	ADSI6		
	Read/Write	R/W							
	Reset Value	0	0	0	0	0	0	0	0
	Function	Must be written as 0.	Input port for result register 7 000: AIN0 001: AIN1 010: AIN2 011: AIN3 100: AIN4 101: AIN5 110: AIN6 111: AIN7			Must be written as 0.	Input port for result register 6 000: AIN0 001: AIN1 010: AIN2 011: AIN3 100: AIN4 101: AIN5 110: AIN6 111: AIN7		

Not Recommended for New Design

AD Input Port Select Register 0(ADC1)

ADCSET1O(L) (0xFFFF_CDC8)

	7	6	5	4	3	2	1	0
Bit Symbol	ADSI9				ADSI8			
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	Input port for result register 9 0000: AIN8 0001: AIN9 0010: AIN10 0011: AIN11 0100: AIN12 0101: AIN13 0110: AIN14 0111: AIN15 1000: AIN16 1001: AIN17 1010: AIN18 Other: Reserved				Input port for result register 8 0000: AIN8 0001: AIN9 0010: AIN10 0011: AIN11 0100: AIN12 0101: AIN13 0110: AIN14 0111: AIN15 1000: AIN16 1001: AIN17 1010: AIN18 Other: Reserved			

(ADCSET1OH) (0xFFFF_CDC9)

	15	14	13	12	11	10	9	8
Bit Symbol	ADSI11				ADSI10			
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	Input port for result register 11 0000: AIN8 0001: AIN9 0010: AIN10 0011: AIN11 0100: AIN12 0101: AIN13 0110: AIN14 0111: AIN15 1000: AIN16 1001: AIN17 1010: AIN18 Other: Reserved				Input port for result register 10 0000: AIN8 0001: AIN9 0010: AIN10 0011: AIN11 0100: AIN12 0101: AIN13 0110: AIN14 0111: AIN15 1000: AIN16 1001: AIN17 1010: AIN18 Other: Reserved			

Not Recommended for New Design

AD Input Port Select Register 1 (ADC1)

ADCSET11(L) (0xFFFF_CDCC)

	7	6	5	4	3	2	1	0
Bit Symbol	ADSI13				ADSI12			
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	Input port for result register 13 0000: AIN8 0001: AIN9 0010: AIN10 0011: AIN11 0100: AIN12 0101: AIN13 0110: AIN14 0111: AIN15 1000: AIN16 1001: AIN17 1010: AIN18 Other: Reserved				Input port for result register 12 0000: AIN8 0001: AIN9 0010: AIN10 0011: AIN11 0100: AIN12 0101: AIN13 0110: AIN14 0111: AIN15 1000: AIN16 1001: AIN17 1010: AIN18 Other: Reserved			

(ADCSET11H) (0xFFFF_CDCD)

	15	14	13	12	11	10	9	8
Bit Symbol	ADSI15				ADSI14			
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	Input port for result register 15 0000: AIN8 0001: AIN9 0010: AIN10 0011: AIN11 0100: AIN12 0101: AIN13 0110: AIN14 0111: AIN15 1000: AIN16 1001: AIN17 1010: AIN18 Other: Reserved				Input port for result register 14 0000: AIN8 0001: AIN9 0010: AIN10 0011: AIN11 0100: AIN12 0101: AIN13 0110: AIN14 0111: AIN15 1000: AIN16 1001: AIN17 1010: AIN18 Other: Reserved			

Not Recommended for New Design

AD Input Port Select Register 2 (ADC1)

ADCSET12(L) (0xFFFF_CDD0)

	7	6	5	4	3	2	1	0
Bit Symbol	ADSI17				ADSI16			
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	Input port for result register 17 0000: AIN8 0001: AIN9 0010: AIN10 0011: AIN11 0100: AIN12 0101: AIN13 0110: AIN14 0111: AIN15 1000: AIN16 1001: AIN17 1010: AIN18 Other: Reserved				Input port for result register 16 0000: AIN8 0001: AIN9 0010: AIN10 0011: AIN11 0100: AIN12 0101: AIN13 0110: AIN14 0111: AIN15 1000: AIN16 1001: AIN17 1010: AIN18 Other: Reserved			

(ADCSET12H) (0xFFFF_CDD1)

	15	14	13	12	11	10	9	8
Bit Symbol	—	—	—	—	ADSI18			
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function					Input port for result register 18 0000: AIN8 0001: AIN9 0010: AIN10 0011: AIN11 0100: AIN12 0101: AIN13 0110: AIN14 0111: AIN15 1000: AIN16 1001: AIN17 1010: AIN18 Other: Reserved			

Not Recommended for New Design

AD PMD Mode Control Register 0 (ADC0)

ADPMOD00 (0xFFFF_CD5C)

	7	6	5	4	3	2	1	0
Bit Symbol	ADF0	—	—	—	—	—	—	ADEN0
Read/Write	R							R/W
Reset Value	0	0	0	0	0	0	0	0
Function	Conversion Complete flag 0: Completed 1: In progress or not started yet							AD conversion 0: Disable 1: Enable

(ADPMOD10 used by ADC1 also applies to these contents.)

Note 1: <ADF> must be 1 as the starting condition, and it becomes 0 when all the conversions enabled are completed.

Note 2: If 0 is set to <ADEN>=0 while the conversion is in progress, the value is set to the result register after the ongoing channel conversion ends, and the operation stops. The next conversion starts not from the part where it stopped but from a channel of the very beginning.

Not Recommended for New Design

AD PMD Mode Control Register 1 (used in ADC0)

		7	6	5	4	3	2	1	0
ADPMOD01(L) (0xFFFF_CD60)	Bit Symbol	ADPE7	ADPE6	ADPE5	ADPE4	ADPE3	ADPE2	ADPE1	ADPE0
	Read/Write	R/W							
	Reset Value	0	0	0	0	0	0	0	0
	Function	Result register 7 enable 0: Disable 1: Enable	Result register 6 enable 0: Disable 1: Enable	Result register 5 enable 0: Disable 1: Enable	Result register 4 enable 0: Disable 1: Enable	Result register 3 enable 0: Disable 1: Enable	Result register 2 enable 0: Disable 1: Enable	Result register 1 enable 0: Disable 1: Enable	Result register 0 enable 0: Disable 1: Enable
		15	14	13	12	11	10	9	8
(ADPMOD01H) (0xFFFF_CD61)	Bit Symbol	—	—	—	—	—	—	—	—
	Read/Write	R/W							
	Reset Value	0	0	0	0	0	0	0	0
	Function	Must be written as 0.	Must be written as 0.	Must be written as 0.	Must be written as 0.	Must be written as 0.	Must be written as 0.	Must be written as 0.	Must be written as 0.

AD PMD Mode Control Register 1 (used in ADC1)

		7	6	5	4	3	2	1	0
ADPMOD11(L) (0xFFFF_CDE0)	Bit Symbol	ADPE15	ADPE14	ADPE13	ADPE12	ADPE11	ADPE10	ADPE9	ADPE8
	Read/Write	R/W							
	Reset Value	0	0	0	0	0	0	0	0
	Function	Result register 15 enable 0: Disable 1: Enable	Result register 14 enable 0: Disable 1: Enable	Result register 13 enable 0: Disable 1: Enable	Result register 12 enable 0: Disable 1: Enable	Result register 11 enable 0: Disable 1: Enable	Result register 10 enable 0: Disable 1: Enable	Result register 9 enable 0: Disable 1: Enable	Result register 8 enable 0: Disable 1: Enable
		15	14	13	12	11	10	9	8
(ADPMOD11H) (0xFFFF_CDE1)	Bit Symbol	—	—	—	—	—	ADPE18	ADPE17	ADPE16
	Read/Write	R/W							
	Reset Value	0	0	0	0	0	0	0	0
	Function	Must be written as 0.	Must be written as 0.	Must be written as 0.	Must be written as 0.	Must be written as 0.	Result register 18 enable 0: Disable 1: Enable	Result register 17 enable 0: Disable 1: Enable	Result register 16 enable 0: Disable 1: Enable

Not Recommended for New Design

AD Count Enable Register 0

ADCNE0(L) (0xFFFF_CD64)

	7	6	5	4	3	2	1	0
Bit Symbol	ADCNE7	ADCNE6	ADCNE5	ADCNE4	ADCNE3	ADCNE2	ADCNE1	ADCNE0
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	Convert the conversion register 7 after the count 0:Always enable 1:Enable after the count	Convert the conversion register 6 after the count 0:Always enable 1:Enable after the count	Convert the conversion register 5 after the count 0:Always enable 1:Enable after the count	Convert the conversion register 4 after the count 0:Always enable 1:Enable after the count	Convert the conversion register 3 after the count 0:Always enable 1:Enable after the count	Convert the conversion register 2 after the count 0:Always enable 1:Enable after the count	Convert the conversion register 1 after the count 0:Always enable 1:Enable after the count	Convert the conversion register 0 after the count 0:Always enable 1:Enable after the count

(ADCNE0H) (0xFFFF_CD65)

	15	14	13	12	11	10	9	8
Bit Symbol	—	—	—	—	—	—	—	—
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								

AD Count Enable Register 1

ADCNE1(L) (0xFFFF_CDE4)

	7	6	5	4	3	2	1	0
Bit Symbol	ADCNE15	ADCNE14	ADCNE13	ADCNE12	ADCNE11	ADCNE10	ADCNE9	ADCNE8
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	Convert the conversion register 15 after the count 0:Always enable 1:Enable after the count	Convert the conversion register 14 after the count 0:Always enable 1:Enable after the count	Convert the conversion register 13 after the count 0:Always enable 1:Enable after the count	Convert the conversion register 12 after the count 0:Always enable 1:Enable after the count	Convert the conversion register 11 after the count 0:Always enable 1:Enable after the count	Convert the conversion register 10 after the count 0:Always enable 1:Enable after the count	Convert the conversion register 9 after the count 0:Always enable 1:Enable after the count	Convert the conversion register 8 after the count 0:Always enable 1:Enable after the count

ADCNE1(H) (0xFFFF_CDE5)

	15	14	13	12	11	10	9	8
Bit Symbol	—	—	—	—	—	ADCNE18	ADCNE17	ADCNE16
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function						Convert the conversion register 18 after the count 0:Always enable 1:Enable after the count	Convert the conversion register 17 after the count 0:Always enable 1:Enable after the count	Convert the conversion register 16 after the count 0:Always enable 1:Enable after the count

Note 1: At least one channel must always be converted in the unit ADC1. If all the channels are converted after counting, the conversion is not skipped properly.

Not Recommended for New Design

AD Conversion Count Setting Register 0 (used in ADC0)

ADCNT0 (0xFFFF_CD68)

	7	6	5	4	3	2	1	0
Bit Symbol	CMPCNT							
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	When the cycle value becomes the same value as the value set here, a channel set after counting is also converted.							

(ADCNT1 used by ADC1 applies to these contents.)

Example of Actual Operation

Setting

ADPMOD01="1011_0111" : Conversion Enabled(CH0,1,2,4,5,7)

ADCNE0 ="1111_0000" : Always Convert (CH0,1,2,3) or after counting (CH4,5,6,7)

ADCNT0 ="0000_1111" : Count Value(16 counts)

Note 1: These are premised on that the amount of PMDTRG capable of converting all the channels within a certain cycle is available.

1) A channel changes from CH0 to CH1, and CH1 to CH2, then INTAD0 generates.
2) The count value decrements by one.
3) Goes to the step 1) if the count value is not 0, and to 4) if the value is 0 (after 16 cycles of conversion).
4) A channel changes from CH0 to CH1, CH1 to CH2, and so on till becomes CH7, then INTAD0 generates.
5) Loads the register value set as count set value.
6) Goes back to the step 1).

AD PMD Mode Basic Setting Register (ADC0)

ADCBASP0 (0xFFFF_CD6C)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	AZSEL	—	—
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function	Must be written as 0.	Must be written as 0.	Must be written as 0.	Must be written as 0.	Must be written as 0.	Sample hold time 1:6 clocks 0:12 clocks	Must be written as 0.	Must be written as 0.
	15	14	13	12	11	10	9	8
Bit Symbol	—	—	—	—	—	—	—	—
Read/Write	R/W							
Reset Value	0	0	0	1	0	0	0	0
Function	Must be written as 0.	Must be written as 0.	Must be written as 0.	Must be written as 1.	Must be written as 0.	Must be written as 0.	Must be written as 0.	Must be written as 0.

(ADCBASP1 used by ADC1 applies to these contents.)

Note 1: The conversion time of ADC is derived from the equation; (the number of clocks selected in <AZSEL>+27 clocks)/ADCLK.

Not Recommended for New Design

AD PMD Mode Clock Control Register 0 (used by ADC0)

ADPCLK0 (0xFFFF_CD58)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	ADPCK		
Read/Write	R					R/W		
Reset Value	0	0	0	0	0	0	0	0
Function						Prescaler clock output select 000: IMCLK 001: IMCLK /2 010: IMCLK /4 011: IMCLK /8 100: IMCLK /16 101: fsys Other: Reserved		

(ADPCLK1 used by ADC1 applies to these contents.)

Note 1: ADC conversions are executed with clocks selected by above-mentioned registers. To guarantee the accuracy, it is necessary to select a conversion clock to make the conversion time less than 36 μs (less than 14 MHz in an AD clock).

Note 2: A conversion clock must not be changed during an AD conversion in progress. More than two clocks of ADCLK after the conversion stops, it must be changed.

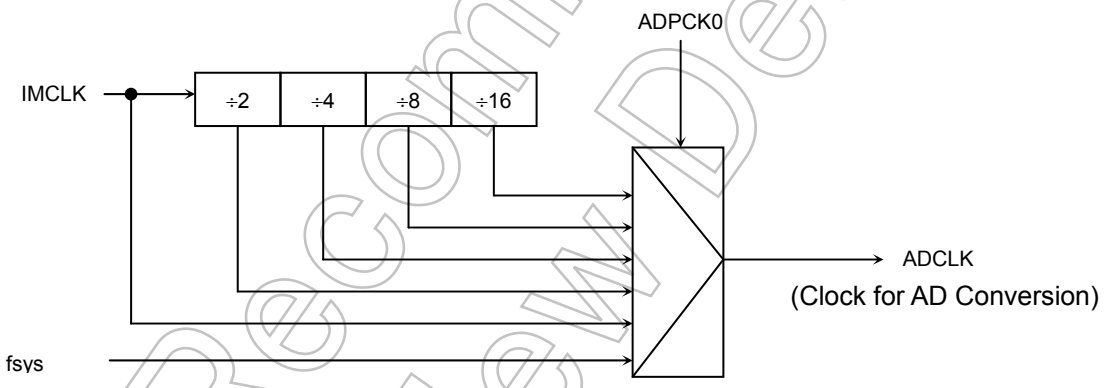


Figure 13.3.2 Clock Control Circuit

Not Recommended for New Design

There are the same registers from 0 to 18 as the result registers.

Here the register 0 is described.

AD PMD Mode Result Register 0

ADPRES0 (0xFFFF_CD00)

	7	6	5	4	3	2	1	0
Bit Symbol	ADR07	ADR06	ADR05	ADR04	ADR03	ADR02	ADR01	ADR00
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0
Function	Lower 8 bits of an AD conversion result							
	15	14	13	12	11	10	9	8
Bit Symbol	VAL	OVR	—	—	—	—	ADR09	ADR08
Read/Write	R							
Reset Value	0	0	0	0	0	0	0	0
Function	Result store flag 1: Stored	Overrun flag 0: No overrun 1: Overrun					Upper 2 bits of an AD conversion result	

Note 1: To access this register, the system of more than 16-bit is to be used. When accessed by 8-bit system, the operation shall not be guaranteed.

Note 2: Bit 15 is an AD conversion result flag <VAL>. When an AD conversion value is stored, 1 is set to it. When this register (ADPRES) is read, it is cleared to 0.

Note 3: Bit 14 is an overrun flag<OVR>. 1 is set to this flag when a conversion result is overwritten before reading the conversion result register (ADPRES). It is cleared to 0 with a flag reading.

Note 4: This register is not accessible with any bit operation instruction.

Not Recommended for New Design

13.4.2 Operation (PMD Mode)

13.4.2.1 Analog Reference Voltage

By writing 0 to the ADMODSEL0<VREFON> bit, a switch between VREFH and VREFL can be turned off. To start an AD conversion, 1 must be written to the <VREFON> bit, and then it must be waited for more than 3 μs until an internal reference voltage is stabilized. The conversion accuracy when the conversion is started waiting for less than 3 μs shall not be guaranteed.

13.4.2.2 Basic Operation

In PMD mode, a conversion result register becomes the reference of AD conversions. Each conversion result register sets Conversion Enabled (ADPMOD01), Conversion Trigger (ADCSETT00), and Input Ports (ADCSET0x). Setting 1 to ADPMOD00<ADEN> causes the wait state of a conversion trigger.

As a conversion trigger from PMD is accepted, AD conversion is executed in ascending order from the conversion result register of the smallest number set in the conversion enabled.

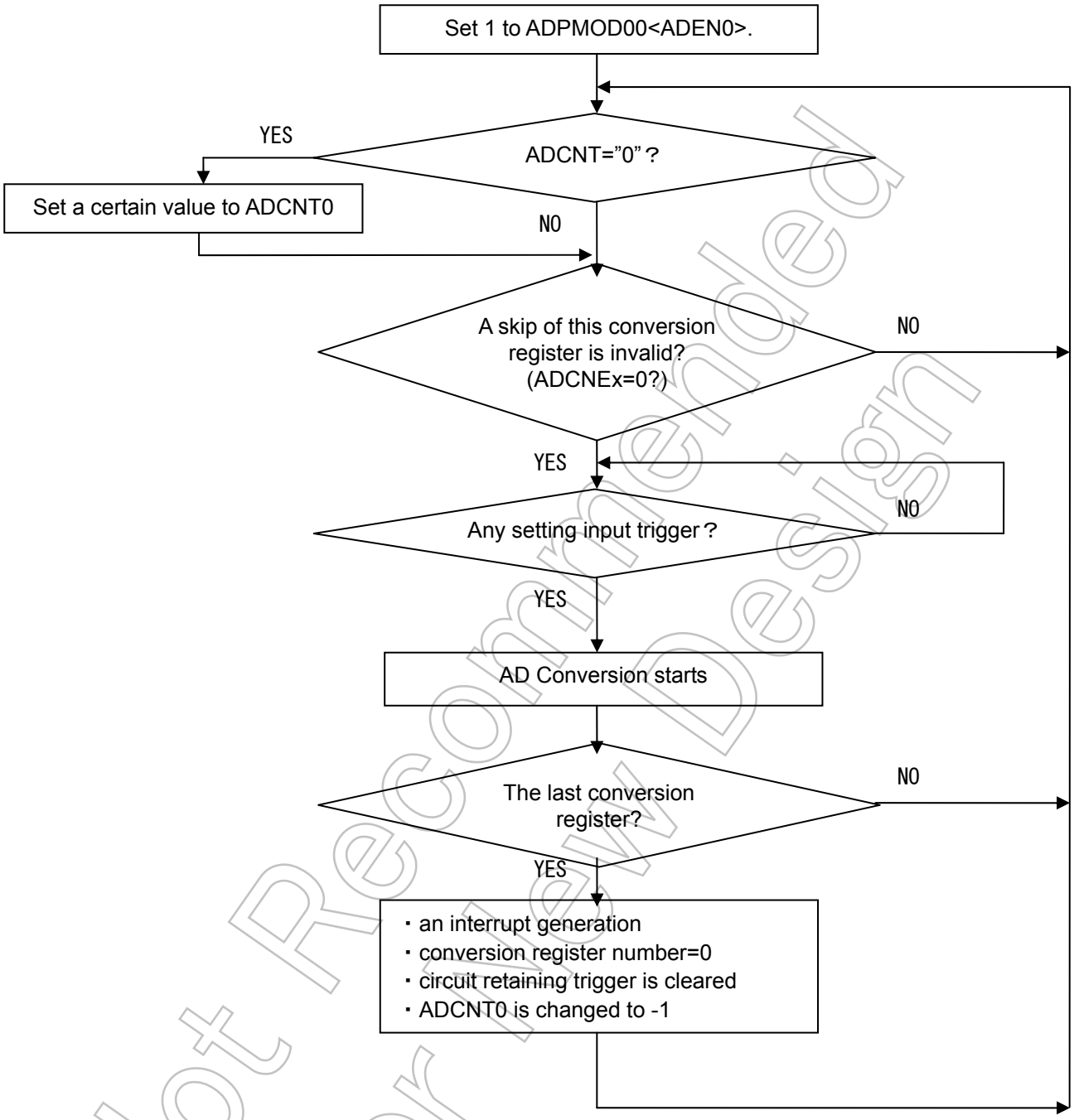
An accepted conversion trigger is retained inside until the conversions of a whole unit are completed. When a conversion result register to be executed next is set to the trigger already accepted, a conversion starts immediately.

13.4.2.3 AD Conversion Counting

In PMD mode, there is an AD conversion count function that enables a skip a conversion trigger (PMDTRG) of a specific result register for the set number of times. By using this function, both a result register whose conversion is desired in every cycle and a result register whose conversion cycle may delay can be controlled in the same unit. The skip, however, is not possible for all the result registers whose conversions are enabled in the unit.

Figure 13.3.3 shows an AD conversion operation in PMD mode.

Not Recommended for New Design



Not Recommended for New Design

Figure 13.3.3 PMD Mode Operation

13.4.2.4 AD Conversion Time

The number of clocks of AD conversion for one cycle is 27 excluding sampling clocks. Since the ADCBASP0<AZSEL> bit can select a sampling clock from 6 or 12 clocks, the sum of AD conversion clocks becomes 33 or 39. ADPCLK0<ADPCK> selects an AD conversion clock among an AD prescaler output fsys, IMCLK, IMCLK/2, IMCLK/4, IMCLK/8, and IMCLK/16. To ensure its accuracy, AD conversion clock must be set less than14MHz, i.e. more than 2.36 μs of AD conversion time (when Sample Hold is 6 clocks).

13.4.2.5 Data Polling

To process an AD conversion result by polling data without using any interrupt, ADPMOD00<ADF> should be polled. When this flag is cleared to 0, a conversion result is stored in a prescribed AD conversion result register. Therefore the result resister must be read after checking the set. To detect any overrun at this time, a conversion result register must be read in 16-bit system. If the result came out as <OVR>=0 and <VAL>=1, a conversion result that was not overwritten would be gained.

Not Recommended for New Design

13.5 Operating Timing

In PMD trigger mode, setting ADPMOD00<ADEN> enables the acceptance of a PMD trigger, and inputting PMDTRG00/01 starts a converting operation. After all the program conversions end, ADF is cleared as an interrupt request is output, and then the next input of PMDTRG00/01 is waited. When ADEN is cleared, ADF is cleared without waiting for the end of all the program conversions.

Inputting of the same PMD trigger whose conversion is in process is ignored, while a different input is retained. A program conversion of a different PMD trigger is processed continuously immediately after the one of the previous trigger.

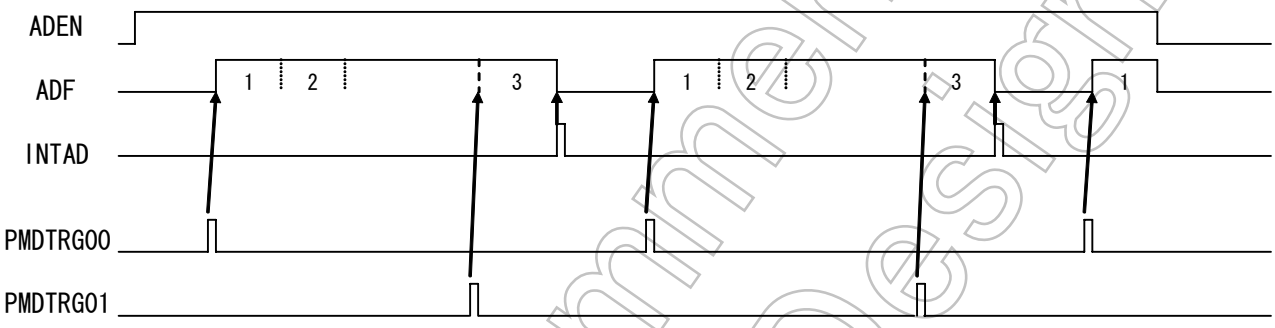


Figure 13.5.1 Timing Chart 1 in PMD Mode

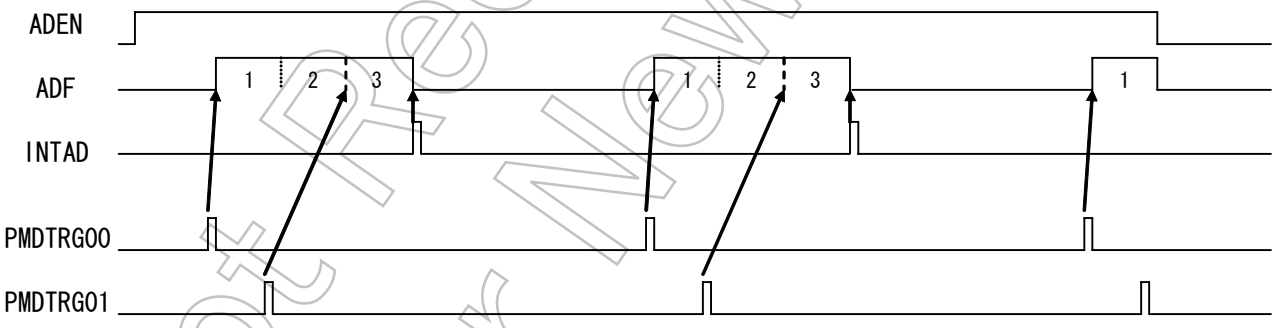


Figure 13.5.2 Timing Chart 2 in PMD Mode

Not Recommended for New Design

13.6 Example of Use

13.6.1 PWM Peak Synchronization (Read once)

Example of Use: Connect a U-phase current CT output to AIN0, and V-phase current CT output to AIN1. A conversion is processed at the PWM carrier peaks (PWM counter=MDPRD). A result of AIN0 is stored in ADPRES0, and a result of AIN1 to ADPRES1.

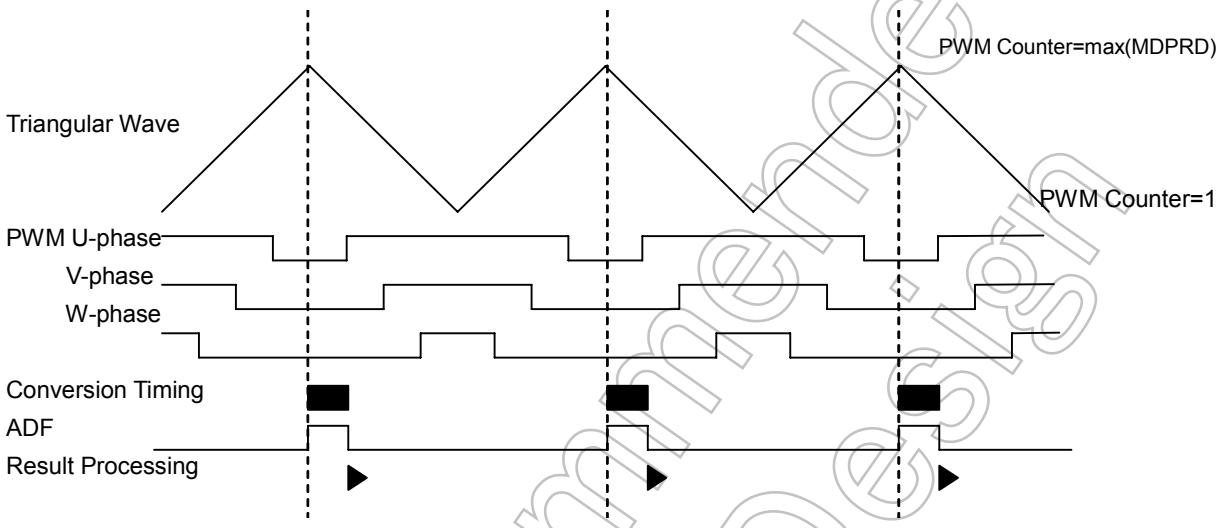


Figure 13.6.1 AD Converter Example of Use 1 Timing

Setting

- ADMODSEL0 = **** **** **** ***1: PMD mode
- ADPMOD00 = **** **** **** ***1 : ADC enabled
- ADPMOD01 = 0000 0000 0000 0011:Select result register 1 or 2
- ADCSET00= **** **** 0001 0000 : Select AIN
- ADCSETT00= **** **** **** 0000 : Select PMDTRG0
- TRGCR0 = **** **** **** *100 : PMD trigger setting

Operation

- At the first PWM carrier peak after 1 is set to ADEN, a conversion starts. ADF becomes 1. It, however, does not start when a triangular wave (PWM counter) is in idle state.
- A conversion is processed in ascending order from the smallest program number (conversion register number).
  Program 0 converts with AIN0 as an input and put the result in ADPRES0.
  Program 1 converts with AIN1 as an input and put the result in ADPRES1.
- As ADF becomes 0, an interrupt INTAD0 generates.

Result Processing

Read ADPRES0 and ADPRES1 after checking if ADF is 0 or in the interrupt processing of finishing all AD conversions, use them as a U-phase and V-phase currents.

Not Recommended for New Design

13.6.2 PWM Peak Synchronization (Read once)

Example of Use: Connect U-phase current to AIN1, V-phase current to AIN2, and W-phase current to IN3. The conversion is supposed to be performed at the peak of a triangular wave (PWM counter=max). The result of AIN1 is store in ADPRES0 and 3, the result of AIN2 in ADPRES1 and 4, and the result of AIN3 in ADPRES2 and 5.

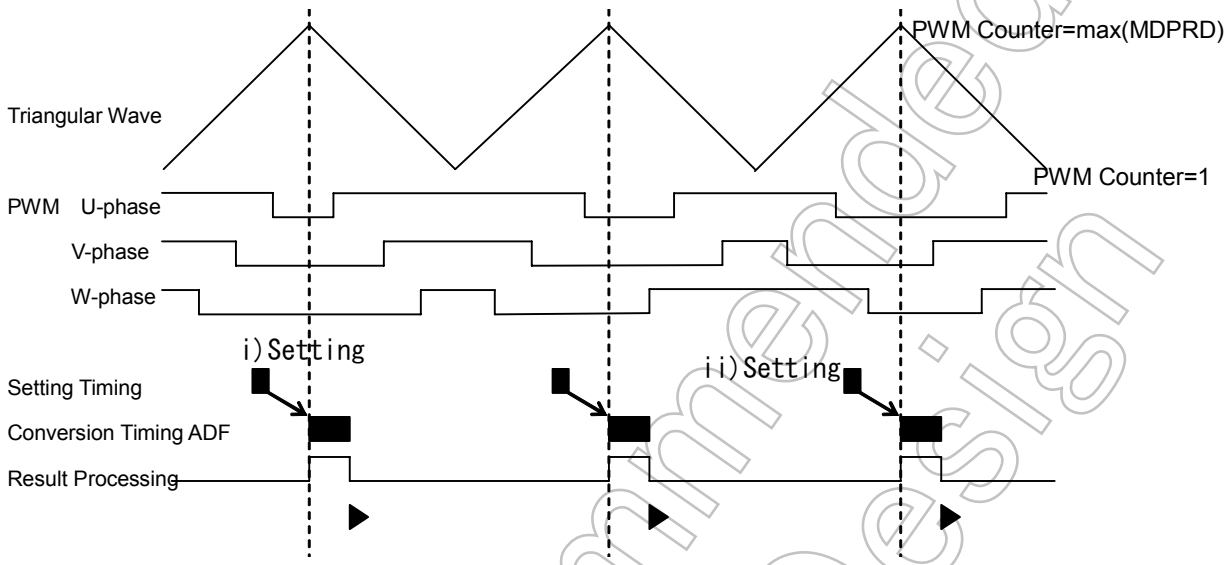


Figure 13.6.2 AD Converter Example of Use 2 Timing

i) Setting

- ADMODSEL0 = **** **** **** ***1: PMD mode
- ADPMOD00 = **** **** **** ***1 : ADC enabled
- ADPMOD01 = 0000 0000 0011 1111: Select result register 0,1,2,3,4,or 5
- ADCSET00= 0001 0011 0010 0001 : Select AIN
- ADCSET01= **** **** 0011 0010 : Select AIN
- ADCSETT00= **** 0000 0000 0000: Select PMDTRG0
- TRGCR0 = **** **** **** *100 : PMD trigger setting

i) Operation

- A conversion starts at the first peak of triangular wave after 1 is set to ADEN. ADF becomes 1.
- Program 0 converts with AIN1 as an input and put the result in ADPRES0.
- Program 1 converts with AIN2 as an input and put the result in ADPRES1.
- Program 2 converts with AIN3 as an input and put the result in ADPRES2.
- Program 3 converts with AIN1 as an input and put the result in ADPRES3.
- Program 4 converts with AIN2 as an input and put the result in ADPRES4.
- Program 5 converts with AIN3 as an input and put the result in ADPRES5.
- Since the settings of ADPE7 and 6 are off, all programs end. ADF becomes 0.

Not Recommended for New Design

13.6.3 Synchronization to an Optional Timing of PWM Cycle (Read twice)

Example of Use: Connect the DC-shunt output to AIN1. A single shunt system executes conversions at the timing where all U, V, W are other than H or L. Output data is to be updated every PWM cycle, and a current is to be detected every PWM cycle.

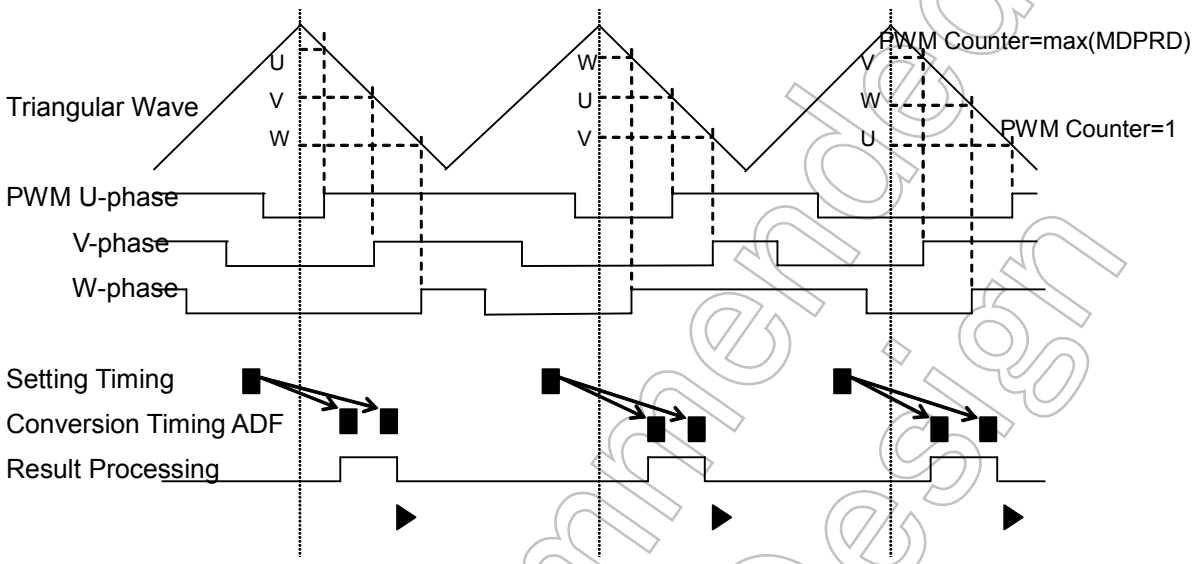


Figure 13.6.3 AD Converter Example of Use 3 Timing

i) Setting

- ADMODSEL0 = **** **** **** ***1: PMD mode
- ADPMOD00 = **** **** **** ***1: ADC enabled
- ADPMOD01 = 0000 0000 0000 1111: Select a conversion register0, 1, 2, or 3
- ADCSET00= 0001 0001 0001 0001 : Select AIN
- ADCSETT00= **** **** 0101 0000: Select PMDTRG0 or 1
- TRGCR0 = **** **** **00 1001 : PMD trigger setting
- TRGCMP00 = Any of CMPU-CMPV: PMD trigger timing setting
- TRGCMP01 = Any of CMPV-CMPW: PMD trigger timing setting

i) Operation

- TRG0MD and TRG1MD are 001 after 1 is set to ADEN, and TRGCMP00 and 01 have effect at the max of a PWM counter.
- A conversion starts when PWM counter= TRGCMP00. ADF becomes 1. Program 0 and 1 with PMDTRG0 selected convert with AIN1 as an input and put the results in ADPRES0 and 1 each.
- A conversion starts when PWM counter=TRGCMP01. Program 2 and 3 with PMDTRG1 selected convert with AIN1 as an input and put the results in ADPRES2 and 3 each.
- Since the settings of AD7-4 are off, the program ends here. ADF becomes 0.

i) Result Processing

Iu=(ADPRES0+ADPRES1)/2 −Iw=(ADPRES2+ADPRES3)/2

ii) Setting

- ADMODSEL0 = **** **** **** ***1: PMD mode
- ADPMOD00 = **** **** **** ***1: ADC enabled
- ADPMOD01 = 0000 0000 0000 1111:Program 0, 1, 2, and 3 enabled
- ADCSET00= 0001 0001 0001 0001 : Select AIN
- ADCSETT00= **** **** 0101 0000: Select PMDTRG0 or 1

Not Recommended for New Design

・TRGCR0 = **** **** **00 1001 : PMD trigger setting
・TRGCMP00 = Any of CMPW – CMPU: PMD trigger timing setting
・TRGCMP01 = Any of CMPU – CMPV: PMD trigger timing setting

ii) Operation

・Since TRG0MD/TRG1MD=001 TRGCMP00 and 01 have effect where PWM counter is the max.
・A conversion starts when PWM counter=TRGCMP00. ADF becomes 1. Program 0 and 1 with PMDTRG0 selected converts with AIN1 as an input and put the result in ADPRES0 and 1 each.
・ A conversion starts when PWM counter=TRGCMP01. Program 2 and 3 with PMDTRG1 selected convert with AIN1 as an input and put the results in ADPRES2 and 3 each.
・Since the settings of AD7-4 are off, the program ends here. ADF becomes 0.

ii) Result Processing

－Iw=(ADPRES0+ADPRES1)/2 Iv=(ADPRES2+ADPRES3)/2

Not Recommended for New Design

14. Motor Control Circuit (PMD: Programmable Motor Driver)

The TMP19A71 contains a two-channel programmable motor driver (PMD). In addition to a 3-phase waveform generation circuit, the PMD also has a sync sampling signal generation circuit for sampling operations of the AD converter. By implementing these functions by hardware, the load on software can be reduced, and vector control of brushless DC motors can easily be implemented.

14.1 Functional Block Diagram

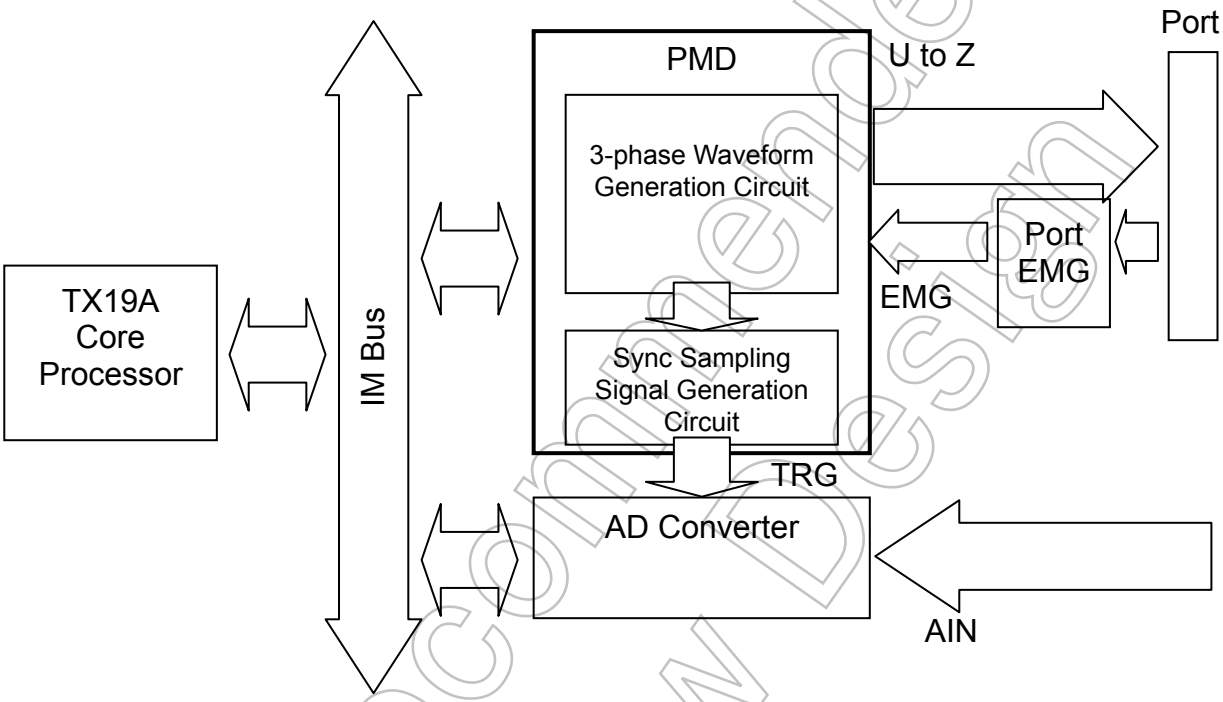


Figure 14.1.1 PMD Functional Block Diagram

Not Recommended for New Design

14.2 PMD Registers

Table 14.2.1 PMD Register Map

Address	Bits	Mnemonic	Register Name
0xFFFF_C300	16	MDCR0	PMD0 Control Register
0xFFFF_C304	16	MDCNT0	PMD0 Count Register
0xFFFF_C308	16	MDPRD0	PMD0 Period Register
0xFFFF_C30C	16	CMPU0	PMD0 Compare Register U
0xFFFF_C310	16	CMPV0	PMD0 Compare Register V
0xFFFF_C314	16	CMPW0	PMD0 Compare Register W
0xFFFF_C318	16	MDOUT0	PMD0 Output Register
0xFFFF_C31C	16	EMGREL0	EMG0 Release Register
0xFFFF_C320	16	EMGCR0	EMG0 Control Register
0xFFFF_C324	16	TRGCR0	Trigger Control Register (PMD0)
0xFFFF_C328	16	TRGCMP00	Trigger Compare 0 Register (PMD0)
0xFFFF_C32C	16	TRGCMP01	Trigger Compare 1 Register (PMD0)
0xFFFF_C330	16	TRGCMP02	Trigger Compare 2 Register (PMD0)
0xFFFF_C340	16	MDCR1	PMD1 Control Register
0xFFFF_C344	16	MDCNT1	PMD1 Count Register
0xFFFF_C348	16	MDPRD1	PMD1 Period Register
0xFFFF_C34C	16	CMPU1	PMD1 Compare Register U
0xFFFF_C350	16	CMPV1	PMD1 Compare Register V
0xFFFF_C354	16	CMPW1	PMD1 Compare Register W
0xFFFF_C358	16	MDOUT1	PMD1 Output Register
0xFFFF_C35C	16	EMGREL1	EMG1 Release Register
0xFFFF_C360	16	EMGCR1	EMG1 Control Register
0xFFFF_C364	16	TRGCR1	Trigger Control Register (PMD1)
0xFFFF_C368	16	TRGCMP10	Trigger Compare 0 Register (PMD1)
0xFFFF_C36C	16	TRGCMP11	Trigger Compare 1 Register (PMD1)
0xFFFF_C370	16	TRGCMP12	Trigger Compare 2 Register (PMD1)

Note: These registers must be accessed as a 16-bit quantity, unless otherwise noted. These registers do not support bit manipulation instructions.

Not Recommended for New Design

14.3 PMD Components

The two PMD channels are, essentially, functionally equivalent so that only PMD0 is explained here.

14.3.1 Three-Phase Waveform Generation Circuit

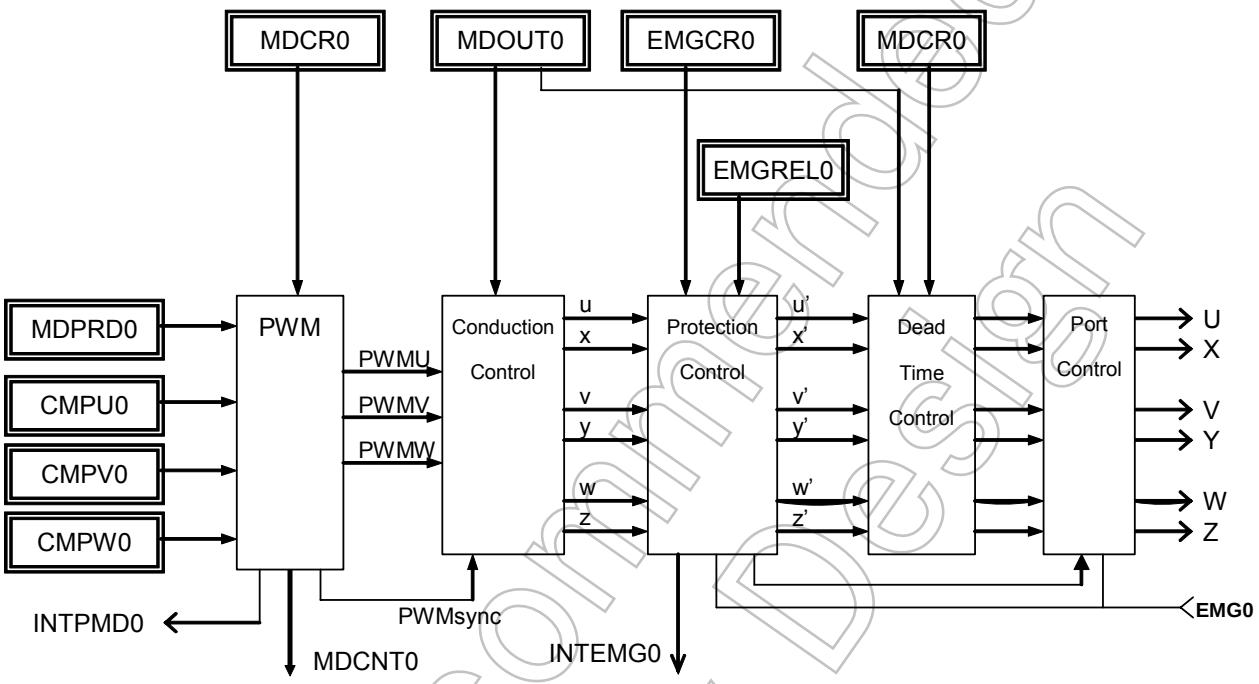


Figure 14.3.1 Three-Phase Waveform Generation Circuit

The 3-phase waveform generation circuit consists of a pulse width modulation (PWM) circuit, a conduction control circuit, an EMG protection (emergency stop) circuit and a dead time control circuit. The pulse width modulation circuit generates independent 3-phase PWM waveforms with the same PWM carrier wave. The conduction control circuit determines the output pattern for each of the upper and lower sides of the U, V and W phases. The EMG protection circuit enables emergency output stop by EMG0 input. The dead time control circuit prevents a short circuit which may occur when the upper side and lower side are switched.

Not Recommended for New Design

14.3.2 Pulse Width Modulation Circuit (PWM Waveform Generation Unit)

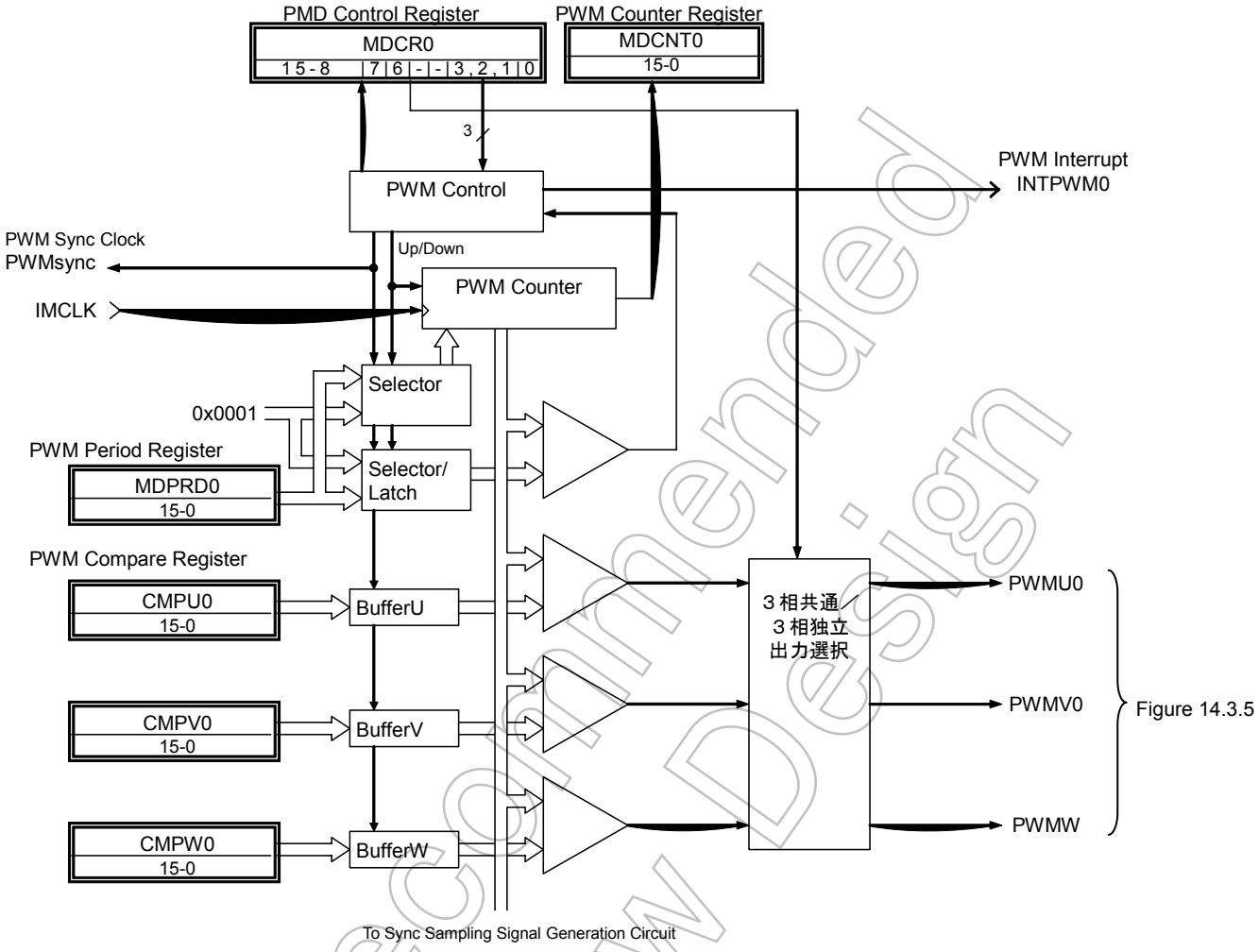


Figure 14.3.2 Pulse Width Modulation Circuit

The pulse width modulation circuit has a 16-bit up-/down-counter (PWM counter) and generates PWM carrier waves with a resolution of 35.7 ns (IMCLK = 28 MHz). The PWM carrier wave mode can be selected from the following two modes:

PWM Mode 0 : Edge PWM (Sawtooth wave modulation)
PWM Mode 1 : Center PWM (Triangular wave modulation)

The MDPRD0 register is used to specify the PWM period. The MDPRD0 is double-buffered and the comparator input is updated at every PWM period or at every half PWM period.

Sawtooth wave PWM : MDPRD0 register value = $\frac{\text{IMCLK [Hz]}}{\text{PWM frequency [Hz]}}$

Triangular wave PWM : MDPRD0 register value = $\frac{\text{IMCLK [Hz]}}{\text{PWM frequency x 2 [Hz]}}$

Not Recommended for New Design

The pulse width modulation circuit compares the PWM compare registers of the 3 phases (CMPU0, CMPV0, CMPW0) and the carrier wave generated by the PWM counter (MDCNT0) to determine which is larger to generate PWM waveforms with the desired duty.

The PWM compare register of each phase has a compare register (double-buffer structure). The PWM compare register value is loaded into the corresponding compare register at every PWM period (when the internal counter value matches the MDPRD0 value). It is also possible to update the compare register at every half PWM period.

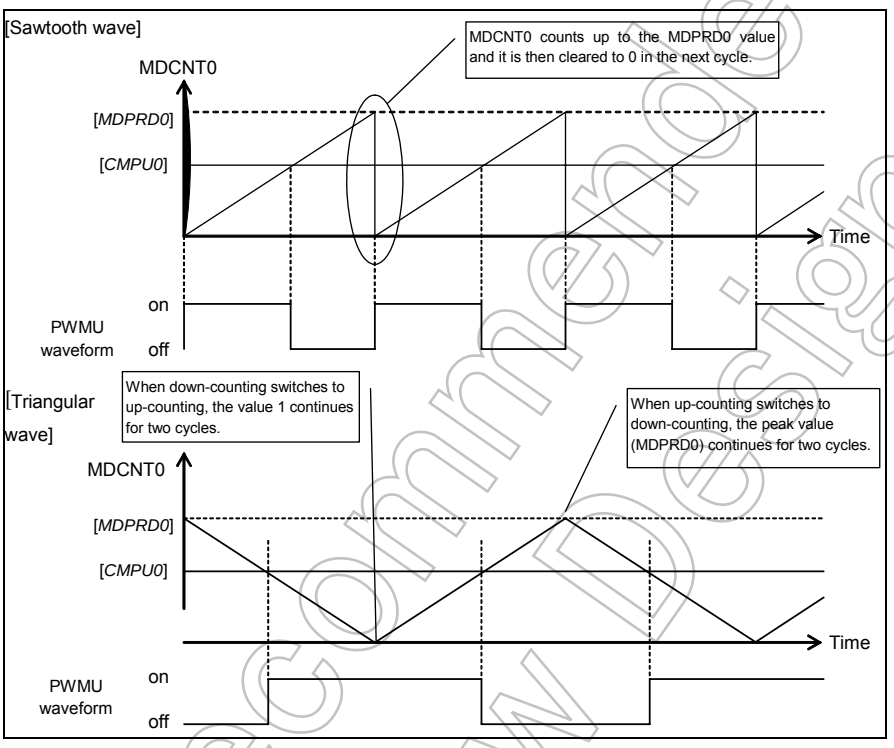


Figure 14.3.3 PWM Waveforms

Three-phase PWM waveforms can be generated in the following two modes:

(1) 3-phase independent mode:
Each of the PWM compare registers for the three phases is set independently to generate independent PWM waveforms for each phase. This mode is used to generate drive waveforms such as sinusoidal waves.

(2) 3-phase common mode:
Only the U-phase PWM compare register is set to generate identical PWM waveforms for all the three phases. This mode is used for rectangular wave drive of brushless DC motors.

Not Recommended for New Design

The pulse width modulation circuit generates PWM interrupt requests in synchronization with PWM waveforms. The PWM interrupt period can be set to half a PWM period, one PWM period, two PWM periods, or four PWM periods.

When the PWM interrupt period is set to two or four PWM periods, the first interrupt after the counter is started occurs at any timing in the specified period. For example, if an interrupt is to be generated at every four PWM periods, the first interrupt will be generated any time during the first to fourth PWM periods with the second and subsequent interrupts generated at every fourth PWM period.

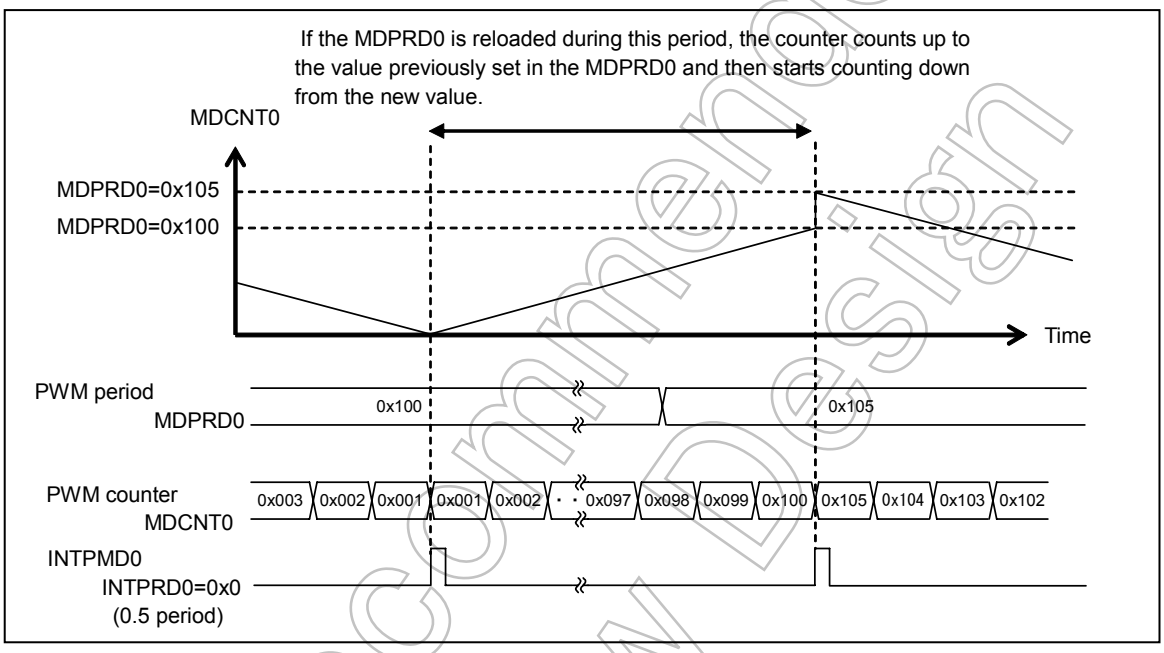


Figure 14.3.4 MDPRD0 Reload Timing (Triangular Wave, Interrupt at Every 0.5 PWM Period)

Not Recommended for New Design

PMD0 Control Register

MDCR0 (0xFFFF_C300)

	7	6	5	4	3	2	1	0
Bit Symbol	UPDWN	SYNCEN	DTYMD	PINT	INTPRD		PWMMD	PWMEN
Read/Write	R	R/W	R/W	R/W	R/W		R/W	R/W
Reset Value	0	0	0	0	0		0	0
Function								

	15	14	13	12	11	10	9	8
Bit Symbol	DTR							
Read/Write	R/W							
Reset Value	0x00							

Symbol	Name	Function
UPDWN	PWM counter flag	0: Up-counting 1: Down-counting
SYNCEN	PMD synchronized start	0: Disable synchronized start 1: Enable synchronized start
DTYMD	Duty mode	0: 3-phase common mode 1: 3-phase independent mode
PINT	PWM interrupt timing	0: Interrupt request when PWM counter = 1 1: Interrupt request when PWM counter = MDPRD [PWM counter = MDPRD when edge mode is selected (PWMMD=0). PWM counter = 1 or MDPRD when 0.5 PWM period is selected (INTPRD=00).]
INTPRD	PWM interrupt period	00: Interrupt request at every 0.5 PWM period (PWM mode1: triangular waves only) 01: Interrupt request at every PWM period 10: Interrupt request at every 2 PWM periods 11: Interrupt request at every 4 PWM periods
PWMMD	PWM mode	0: PWM Mode 0: edge PWM (sawtooth waves) 1: PWM Mode 1: center PWM (triangular waves)
PWMEN	PWM counter start	0: Stop & clear 1: Start

Note: The settings in the MDCR0 register must be changed while the PWMEN bit is 0. It is also not allowed to change the MDCR0 settings at the same time as writing to the PWMEN bit to start or stop the PWM counter.

Not Recommended for New Design

PMD0 Count Register

MDCNT0
(0xFFFF_C304)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit Symbol	MDCNT															
Read/Write	R															
Reset Value	0x0000															
Function	PWM counter value: 0x0001 to 0xFFFF															

PMD0 Period Register

MDPRD0
(0xFFFF_C308)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit Symbol	MDPRD															
Read/Write	R/W															
Reset Value	0x0000															
Function	PWM carrier wave period (Must be within 0x0010 to 0xFFFF.)															

Note: This register is double-buffered; the value written to this register takes effect when MDCNT0 = MDPRD0.

PMD0 Compare Registers (U, V, W)

CMPU0
(0xFFFF_C30C)
CMPV0
(0xFFFF_C310)
CMPW0
(0xFFFF_C314)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit Symbol	CMPU															
	CMPV															
	CMPW															
Read/Write	R/W															
Reset Value	0x0000															

CMPU0	PWM Compare U Register	0x0000 to 0xFFFF: U-phase pulse width duty
CMPV0	PWM Compare V Register	0x0000 to 0xFFFF: V-phase pulse width duty
CMPW0	PWM Compare W Register	0x0000 to 0xFFF: W-phase pulse width duty

Note 1 : When CMPx0=0, a 0%-duty cycle waveform is generated. When CMPx0 ≧ MDPRD0, a 100%-duty cycle waveform is generated.

Note 2 : These registers are double-buffered; the values written to these registers take effect when MDCNT0 = MDPRD0.

Not Recommended for New Design

Detailed Description of the PMD0 Registers

Symbol	Name	Function
UPDWN	PWM counter flag	Indicates whether the PWM counter is up-counting or down-counting. When edge PWM is selected, this bit is always read as 0.
SYNCEN	PMD synchronized start	Enables the PMD synchronized start function.
DTYMD	Duty mode	Selects whether to make duty setting independently for each phase or to use the CMPU register setting for all three phases.
PINT	PWM interrupt timing	Selects whether to generate an interrupt when PWM counter equals 1 or the MDPRD value.
INTPRD	PWM interrupt period	Selects the PWM interrupt period from 0.5 PWM period, one PWM period, two PWM periods and four PWM periods. If this bit is changed during operation, an interrupt may occur at that time.
PWMMD	PWM mode	Selects PWM Mode 0 (edge PWM, sawtooth wave) or PMW Mode 1 (center PWM, triangular wave).
PWMEN	Waveform generation circuit enable/disable	When this bit is cleared to stop and clear the PWM counter, output ports become high-impedance. Before setting this bit to 1 to start the PWM counter, it is necessary to set all the other bits in the MDCR0 register. While this bit is set to 1, do not change the MDCR0 settings other than the PWMEN bit.
MDCNT	PWM counter	A 16-bit counter for reading the PWM period count value.
MDPRD	PWM period	A 16-bit register for specifying the PWM period. This register is double-buffered and can be changed even when the PWM counter is counting. The buffer is loaded at every PWM period. (That is, when the PWM counter matches the MDPRD value. When 0.5 PWM period is selected, loading is performed when the PWM counter matches 1 or MDPRD0.) See Figure 14.3.4.
CMPU CMPV CMPW	PWM pulse width	16-bit compare registers for determining the output pulse width of U, V and W phases. These registers are double-buffered. Pulse width is determined by comparing the buffer and the PWM counter to evaluate which is larger. When CMPx0 = 0, a 0% duty-cycle waveform is generated. When CMPx0 >= MDPRD0, a 100% duty-cycle waveform is generated. (To be loaded when the PWM counter matches the MDPRD value. When 0.5 period is selected, loading is performed when the PWM counter matches 1 or MDPRD.)

Setting the SYNCEN bit of the MDCR0 register to 1 enables the PMD synchronized start function. When the PWMEN bit of the MDCR0 is set to 1 with the PMD synchronized start function enabled, PMD0 is put on standby for starting as soon as the PWMEN bit of the MDCR1 is set to 1 to start PMD1. By setting the SYNCEN and PWMEN bits simultaneously, PMD0 can be put on standby for synchronized start.

When the synchronized start function is enabled for both PMD0 and PMD1 (MDCR0.SYNCEN=1, MDCR1.SYNCEN=1), the channel that is enabled first (MDCRx.PWMEN bit=1) is put on standby and starts operating as soon as the other channel is enabled.

Even when the synchronized start function is enabled, registers are set independently for each channel. To operate PMD0 and PMD1 with the same conditions, it is necessary to set the PMD0 and PMD1 registers identically.

Example: To start PMD0 in synchronization with PMD1

```
MDCR0 = 0y********_*1*****1          ; SYNCEN=1 (Enable the synchronized start function.)
                                     ; PWMEN=1 (Put PMD0 on standby.)
MDCR1 = 0y********_*******1          ; PWMEN=1 (Start PMD1.)
```

Not Recommended for New Design

14.3.3 Conduction Control Circuit

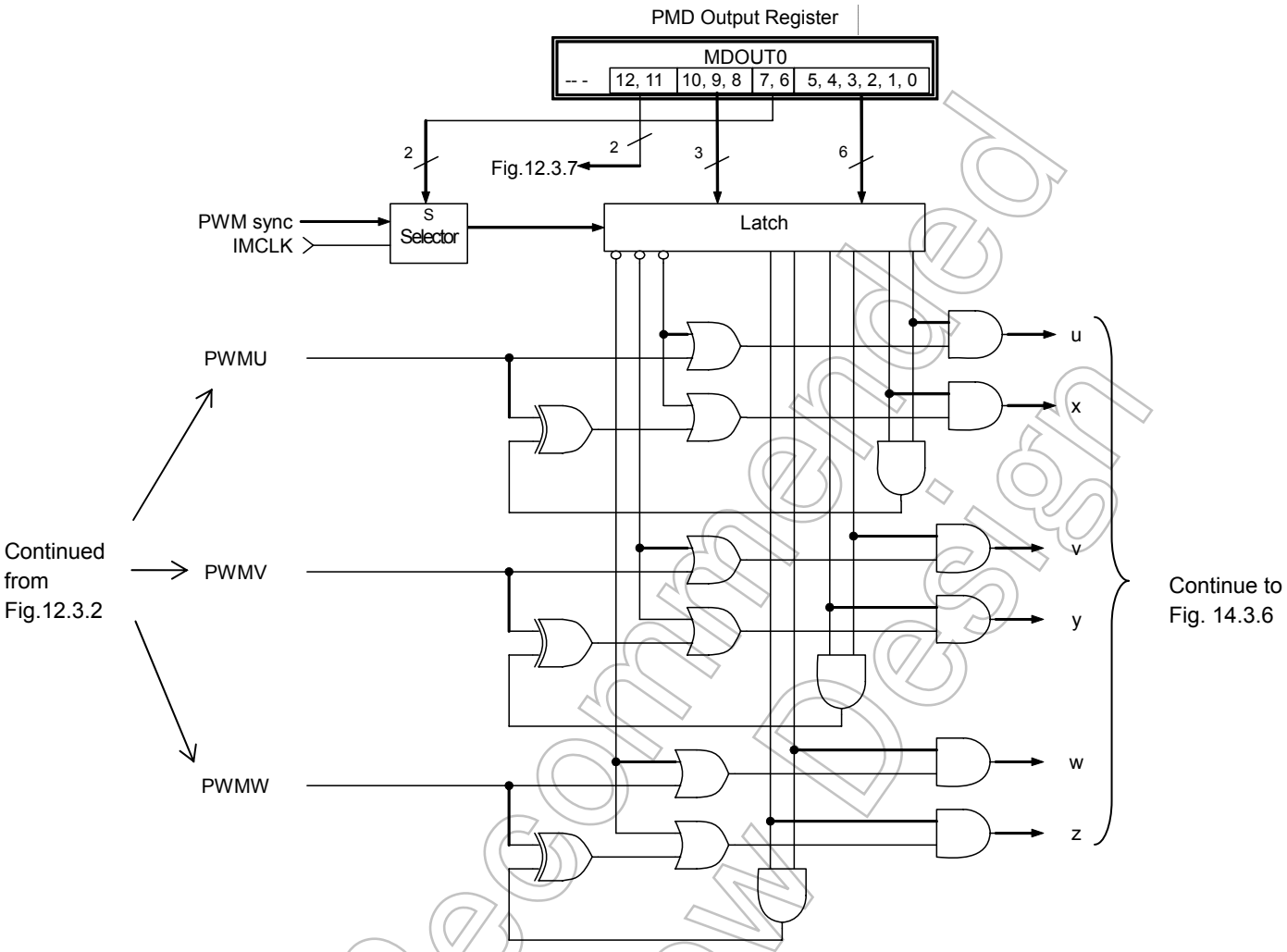


Figure 14.3.5 Conduction Control Circuit

The conduction control circuit performs output port control according to the settings made in the PMD output register (MDOUT0). The MDOUT0 register bits are divided into two parts: settings for the synchronization signal for port output and settings for port output. The latter part is double-buffered and update timing can be set as synchronous or asynchronous to PWM.

The output settings for six port lines are made independently for each of the upper and lower phases through the POLH and POLL bits of the MDOUT0 register. In addition, the UOC, VOC and WOC bits of the MDOUT0 register are used to select PWM or H/L output for each of the U, V, and W phases. When PWM output is selected, PWM waveforms are output. When H/L output is selected, output is fixed to either a high or low level. Table 14.3.1 shows a summary of U-phase port outputs according to port output and polarity settings in the MDOUT0.

Not Recommended for New Design

PMD0 Output Register

MDOUT0 (0xFFFF_C318)

	7	6	5	4	3	2	1	0
Bit Symbol	PSYNS		WOC		VOC		UOC	
Read/Write	R/W		R/W		R/W		R/W	
Reset Value	0		0		0		0	
	15	14	13	12	11	10	9	8
Bit Symbol	―	―	―	POLH	POLL	WPWM	VPWM	UPWM
Read/Write	R	R	R	R/W	R/W	R/W	R/W	R/W
Reset Value	0	0	0	0	0	0	0	0

Symbol	Name	Function
POLH	Upper phase port polarity	0: Low active 1: High active
POLL	Lower phase port polarity	0: Low active 1: High active
WPWM	W-phase PWM output	0: H/L output 1: PWM waveform output
VPWM	V-phase PWM output	0: H/L output 1: PWM waveform output
UPWM	U-phase PWM output	0: H/L output 1: PWM waveform output
PSYNCS	MDOUT transfer timing	00: Async to PWM 01: Load when PWM counter = 1 10: Load when PWM counter = MDPRD 11: Load when PWM counter = 1 or MDPRD
WOC	W-phase output control	See Table 14.3.1.
VOC	V-phase output control	
UOC	U-phase output control	

Note 1: Before changing the POLH, POLL and PSYNCS bits of the MDOUT0 register, make sure that the MDCR0.PWMEN bit is cleared to 0.

Note 2: The xPWM and xOC bits of the MDOUT0 register are double-buffered; the values written to these bits take effect according to the timing selected in the MDOUT0.PSYNCS field.

Not Recommended for New Design

Table 14.3.1 Summary of U-Phase Port Outputs according to the UOC and UPWM Settings

Polarity: Active high (POLH, POLL=1)

MDOUT<UOC>		MDOUT0<UPWM>			
		0: H/L output		1: PWM output	
Bit 1	Bit 0	U output	X output	U output	X output
0	0	L	L	$\overline{PWM}$	PWM
0	1	L	H	L	PWM
1	0	H	L	PWM	L
1	1	H	H	PWM	$\overline{PWM}$

Polarity: Active low (POLH, POLL=0)

MDOUT<UOC>		MDOUT0<UPWM>			
		0: H/L output		1: PWM output	
Bit 1	Bit 0	U output	X output	U output	X output
0	0	H	H	PWM	$\overline{PWM}$
0	1	H	L	H	$\overline{PWM}$
1	0	L	H	$\overline{PWM}$	H
1	1	L	L	$\overline{PWM}$	PWM

The VOC and VPWM bits and the WOC and WPWM bits should be set for the V phase and the W phase, respectively, as shown in the above table.

Name	Symbol	Function
Output port polarity	POLL, POLH	Select the output port polarity for the upper and lower phases. Set these bits when MDCR0.PWMEN=0.
Port output sync setting	PSYNCS	Select port output timing for the U, V and W phases. Select either MDCNT0 (PMD0 compare register) peak/bottom sync or async.
U-, V-, W-phase output control	UOC, VOC, WOC, UPWM, VPWM, WPWM	Specify port output settings for the U, V and W phases (see Table 14.3.1).

Not Recommended for New Design

14.3.4 EMG Protection Circuit

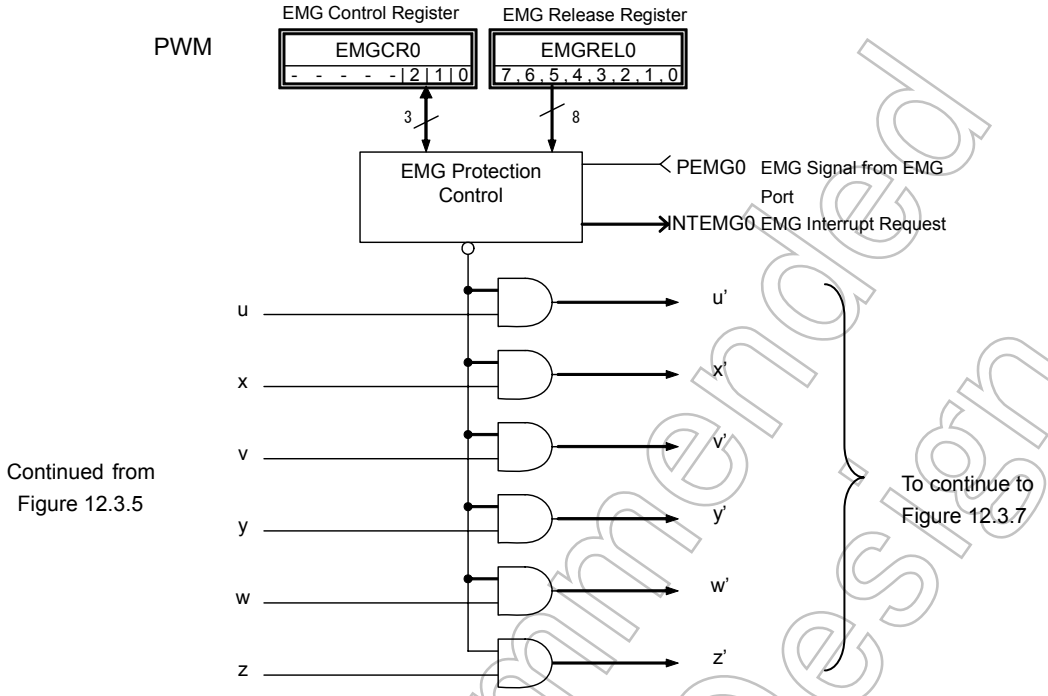


Figure 14.3.6 EMG Protection Circuit

The EMG protection circuit is activated when the EMG input from the EMG0 (PA6) pin becomes the active state specified in the port A EMG control register (PAECR). When the PA6 pin is not configured as an EMG input pin, the EMG protection circuit does not function.

The EMG protection circuit offers an emergency stop mechanism: when the EMG input is asserted, an EMG interrupt request (INTEMG0) is generated and the PMDTRG0 output to the AD converter is disabled.

When only the PMD is protected with the EMG input pin enabled, all six port output lines output inactive signals.

EMG protection is set through the EMG control register (EMGCR0). A read value of 1 in the EMGST bit of the EMGCR0 indicates that the EMG protection circuit is active. In this state, EMG protection can be released by setting all the port output lines inactive (MDOUT[10:0]= 00000000000) and then setting the EMGRS bit of the EMGCR0 to 1.

To disable the EMG protection function, the following sequence of operations must be performed consecutively. This sequence becomes invalid if it is interrupted by any operation on the EMGCR0 or EMGREL0 register before it is completed.

1) Write 0x5A in the EMGREL0 register.
2) Write 0xA5 in the EMGREL0 register.
3) Clear the EMGEN bit of the EMGCR0 register to 0.

If protection is released in the EMG protection circuit while the EMG input pin is asserted, protection is applied again. For details about the port settings related to the EMG protection function, see section 8.12 Notes on Using the EMG Input Pins (PA6, PA8).

Not Recommended for New Design

EMG0 Release Register

EMGREL0
(0xFFFF_C31C)

	7	6	5	4	3	2	1	0
Bit Symbol	EMGREL							
Read/Write	W							
Reset Value	0x00							
Function	The EMG protection circuit can be disabled by writing 0x5A and 0xA5 in this order to this register and then clearing EMGCR0.EMGEN bit to 0.							

EMG0 Control Register

EMGCR0
(0xFFFF_C320)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	EMGST	EMGRS	EMGEN
Read/Write	R	R	R	R	R	R	W	R/W
Reset Value	0	0	0	0	0	0	0	1

Symbol	Name	Function
EMGST	EMG protection state	0: —— 1: Protected
EMGRS	EMG protection release	0: —— 1: Release protection
EMGEN	EMG protection circuit enable/disable	0: Disable 1: Enable

Detailed Description of the EMG Control Register

Name	Symbol	Function
EMG protection state	EMGST	The EMG protection state can be known by reading this bit.
EMG protection release	EMGRS	EMG protection can be released by setting MDOUT[10:0] to 00000000000 and then setting the EMGRS bit to 1.
EMG protection circuit enable/disable	EMGEN	The EMG protection circuit is enabled by setting this bit to 1. In the initial state, the EMG protection circuit is enabled. To disable this circuit, write 0x5A and 0xA5 in this order to the EMGREL0 register and then clear the EMGEN bit to 0.

Not Recommended for New Design

14.3.5 Dead Time Control Circuit

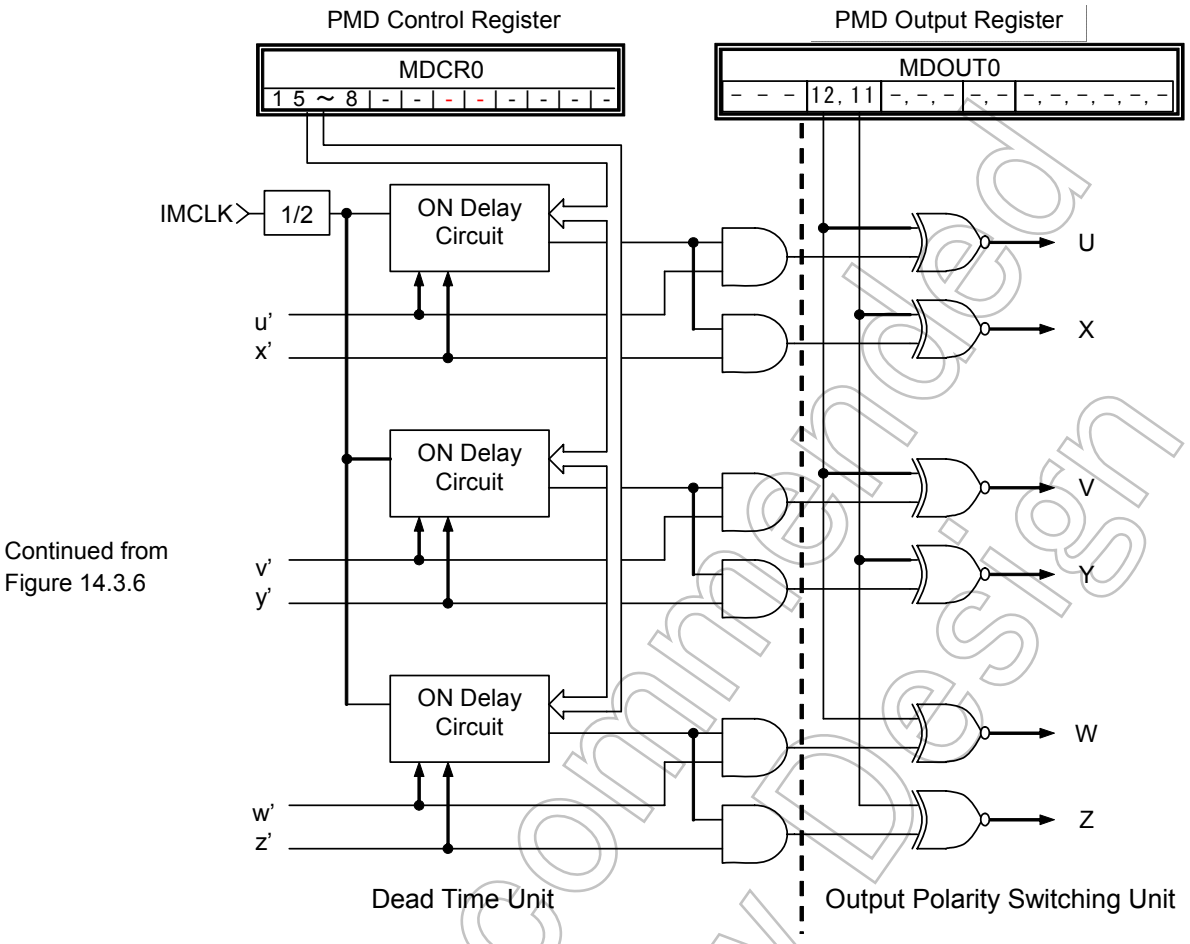


Figure 14.3.7 Dead Time Control Circuit

The dead time control circuit consists of a dead time unit and an output polarity switching unit.

For each of the U, V, and W phases, the ON delay circuit introduces a delay (dead time) when the upper and lower phases are switched to prevent a short circuit. The dead time is set to the DTR field in the MDCR0 register as an 8-bit value with a resolution of 71.4 ns (at IMCLK = 28 MHz). No delay time is inserted when DTR=0x00.

The output polarity switching unit allows the polarity (active high or active low) of the upper and lower phases to be independently set through the POLH and POLL bits of the MDOUT0 register.

PMD0 Control Register

MDCR0 (0xFFFF_C300)

	7	6	5	4	3	2	1	0
Bit Symbol	UPDWN	-	DTYMD	PINT	INTPRD		PWMMD	PWMEN
Read/Write	R	R/W	R/W	R/W	R/W		R/W	R/W
Reset Value	0	0	0	0	0		0	0
Function		Must be set to 0.						

	15	14	13	12	11	10	9	8
Bit Symbol	DTR							
Read/Write	R/W							
Reset Value	0x00							
Function	Dead time: 71.4 ns × 8 bits (max. 18.2 μs) IMCLK = 28 MHz							

Not Recommended for New Design

14.3.6 Sync Sampling Signal Generation Circuit

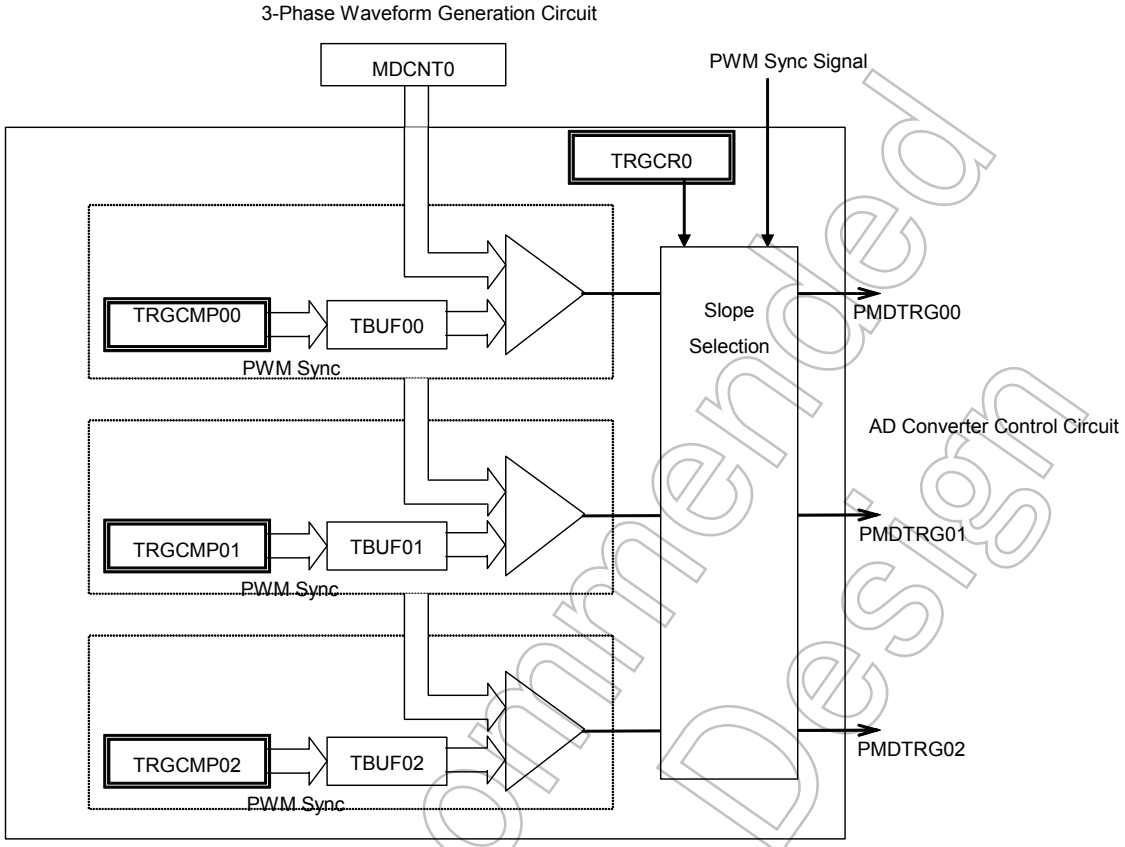


Figure 14.3.8 Sync Sampling Signal Generation Circuit

The sync sampling signal generation circuit generates trigger signals for starting ADC sampling in synchronization with PWM. The ADC trigger signal (PMDTRG0) is generated by a match between the MDCNT0 and TRGCMP0. The signal generation timing can be selected from up-count match, down-count match, and up-/down-count match. When the edge PWM mode is selected, the ADC trigger signal is generated on an up-count match. When PWM output is disabled (MDCR0.PWMEN=0) or EMG protection is applied, trigger output is also disabled.

Not Recommended for New Design

Trigger Control Register (PMD0)

TRGCR0
(0xFFFF_C324)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	—	—	TRG2MD			TRG1MD			TRG0MD		
Read/Write	R	R	R	R	R	R	R	R/W			R/W			R/W		
Reset Value	0	0	0	0	0	0	0	000			000			000		

Symbol	Name	Function
TRG2MD	PMDTRG2 mode setting	000: Trigger output disabled 001: Trigger output on down-count match 010: Trigger output on up-count match 011: Trigger output on up-/down-count match 100: Trigger output at PWM carrier peak 101: Trigger output at PWM carrier bottom 110: Trigger output at PWM carrier peak/bottom 111: ——
TRG1MD	PMDTRG1 mode setting	
TRG0MD	PMDTRG0 mode setting	

Note: The TRG0MD, TRG1MD and TRG2MD fields must be set while MDCR0.PWMEN=0.

Trigger Compare Registers (PMD0)

TRGCMP02
(0xFFFF_C330)
TRGCMP01
(0xFFFF_C32C)
TRGCMP00
(0xFFFF_C328)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit Symbol	—															
Read/Write	R/W															
Reset Value	0x0000															
Function	PMDTRG0 is output on a match between this register and MDCNT0.															

Note1: The trigger compare registers must be set to satisfy the following conditions:

1 < RGCMP00, TRGCMP01, TRGCMP02 <MDPRD0

Note 2: These registers are double-buffered; the values written to these registers take effect at the following timings:

TRGxMD=001: The register values take effect when MDCNT0=MDPRD0.

TRGxMD=010: The register values take effect when MDCNT0=0.

TRGxMD=011: The register values take effect when MDCNT0=MDPRD0 or 0.

Not Recommended for New Design

15. Encoder Input Circuit

15.1 Functional Overview

(1) Allows direct input of the incremental encoder signal.
(2) Contains a x4 multiplier circuit and a rotation direction control circuit.
(3) Contains an absolute position detection counter.
(4) Generates an interrupt on a match with the encoder pulse position set value.
(5) Incorporates noise filters in the signal input part.

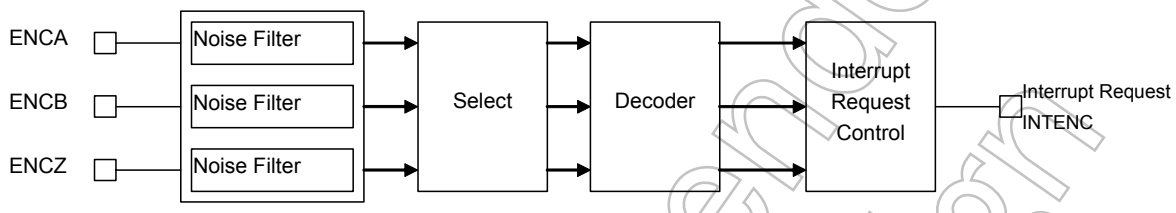


Figure 15.1.1 Block Diagram of the Encoder Input Circuit

Note: Unless otherwise specified, the registers for the encoder input circuit must be accessed as a 16-bit quantity. Bit manipulation instructions cannot be used on these registers.

Not Recommended for New Design

15.2 Register Description

Encoder Input Control Register

ENTNCR (0xFFFF_C400)

	7	6	5	4	3	2	1	0
Bit Symbol	ZEN	CUNEN	NR1	NR0	ENCAP	—	—	—
Read/Write	R/W							
Reset Value	0	0	00		0	0	0	0
	15	14	13	12	11	10	9	8
Bit Symbol	ENCLR	U/D	ZDET	—	—	—	—	—
Read/Write	W	R						
Reset Value	0	0	0	0	0	0	0	0

Symbol	Name	Function
ENCLR	Encoder pulse counter clear	Writing a 1 and then a 0 to this bit clears the encoder counter.
U/D	Encoder rotation direction	1: CW 0: CCW
ZDET	Phase Z detection state	1: Phase Z is detected. 0: The CUNEN bit is set to 1 or a reset is applied.
ZEN	Counter clear by phase Z	1: Enable 0: Disable
CUNEN	Encoder pulse counter enable	1: Enable 0: Disable
NR[1:0]	Noise filter	00: No filter 01: Eliminate pulses of less than 31/IMCLK as noise 10: Eliminate pulses of less than 63/IMCLK as noise 11: Eliminate pulses of les than 127/IMCLK as noise
ENCAP	Encoder interrupt request enable	1: Enable 0: Disable

Note 1: The ENTNCR register must be set after all the other relevant registers have been set.

Note 2: Do not change the settings in this register other the ENCLR and CUNEN bits while the encoder counter is operating.

Note 3: If the CUNEN bit is cleared to 0 when ENCNT=ENINT, the INTENC interrupt is generated.

Note 4: To re-enable the encoder counter after disabling it by CUNEN=0, clear the counter value by using the ENCLR bit.

Not Recommended for New Design

Table 15.2.1 Detailed Description of the Encoder Input Control Register

Name	Symbol	Function
Encoder pulse counter clear	ENCLR	Writing a 1 and then a 0 to this bit clears the encoder counter to 0. Then, the counter starts counting again.
Encoder rotation direction	U/D	When the motor is rotating in CW direction (phase A of the incremental encoder signal is 90 degrees ahead of phase B), this bit is set to 1. When the motor is rotating in CCW direction (phase A is 90 degrees behind phase B), this bit is cleared to 0.
Z phase detection state	ZDET	This bit is cleared to 0 when a 1 is written to the CUNEN bit and at reset. It is set to 1 on the next ZDETECT—the signal to be output on the rising edge (CW direction) or falling edge (CCW direction) of the incremental encoder signal phase Z. (This bit is independent of the ZEN value.)
Counter clear by phase Z	ZEN	When the motor is rotating in CW direction, this bit is cleared to 0 on the rising edge of phase Z (ZDETECT). When the motor is rotating in CCW direction, this bit is cleared to 0 on the falling edge of phase Z (ZDETECT). If ENCLK (a clock obtained by multiplying the phase A and phase B signals by 4) and ZDETECT coincide with each other, the counter is cleared to 0 without counting.
Encoder pulse counter enable	CUNEN	When CUNEN=1, the ZDET bit is cleared to 0 and the encoder counter (ENCNT) is enabled. When CUNEN=0, the encoder counter is disabled.
Noise filter	NR1,0	00: No filter 01: Eliminate pulses of less than 31/IMCLK as noise (1.11 µs IMCLK = 28 MHz) 10: Eliminate pulses of less than 63/IMCLK as noise (2.25 µs IMCLK = 28 MHz) 11: Eliminate pulses of less than 127/IMCLK as noise (4.54 µs IMCLK = 28 MHz)
Encoder interrupt request	ENCAP	ENCAP=1 enables interrupt request signal generation. ENCAP=0 disables interrupt request signal generation.

Not Recommended for New Design

Encoder Counter Reload Register

		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENRELOAD	Bit Symbol	–															
(0xFFFF_C404)	Read/Write	R/W															
	Reset Value	0x0000															
	Function	Encoder counter target value: 0x0000 to 0xFFFF (input pulse count multiplied by 4) When phase Z is used: Set the number of pulses required for one rotation When phase Z is not used: Set the number of pulses required for one rotation minus one															

When the encoder counter (ENCNT) is up-counting, the counter is zero-cleared on the next ENCLK timing after the count value reaches the ENRELOAD value. When the encoder counter (ENCNT) is down-counting, it is re-loaded with the ENRELOAD value on the next ENCLK timing after the counter value reaches 0.

Encoder Compare Register

		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENINT	Bit Symbol	–															
(0xFFFF_C408)	Read/Write	R/W															
	Reset Value	0x0000															
	Function	Interrupt request generation position: 0x0000 to 0xFFFF When the encoder counter matches the value set in this register, an interrupt request is generated.															

When the encoder counter (ENCNT) value reaches the ENINT value, an interrupt request (INTENC) is generated. When ZEN=1, however, an interrupt request is not generated until the ZDET bit is set to 1.

Encoder Counter

		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENCNT	Bit Symbol	–															
(0xFFFF_C40C)	Read/Write	R															
	Reset Value	0x0000															
	Function	0x0000 to 0xFFFF: Up-/down counter that counts encoder pulses															

When the motor is rotating in CW direction, the encoder counter is an up-counter to be zero-cleared on the next ENCLK timing after the count value reaches the ENRELOAD value. When the motor is rotating in CCW direction, the encoder counter is a down-counter to be re-loaded with the ENRELOAD value on the next ENCLK timing after the count value reaches 0.

Not Recommended for New Design

15.3 Operation

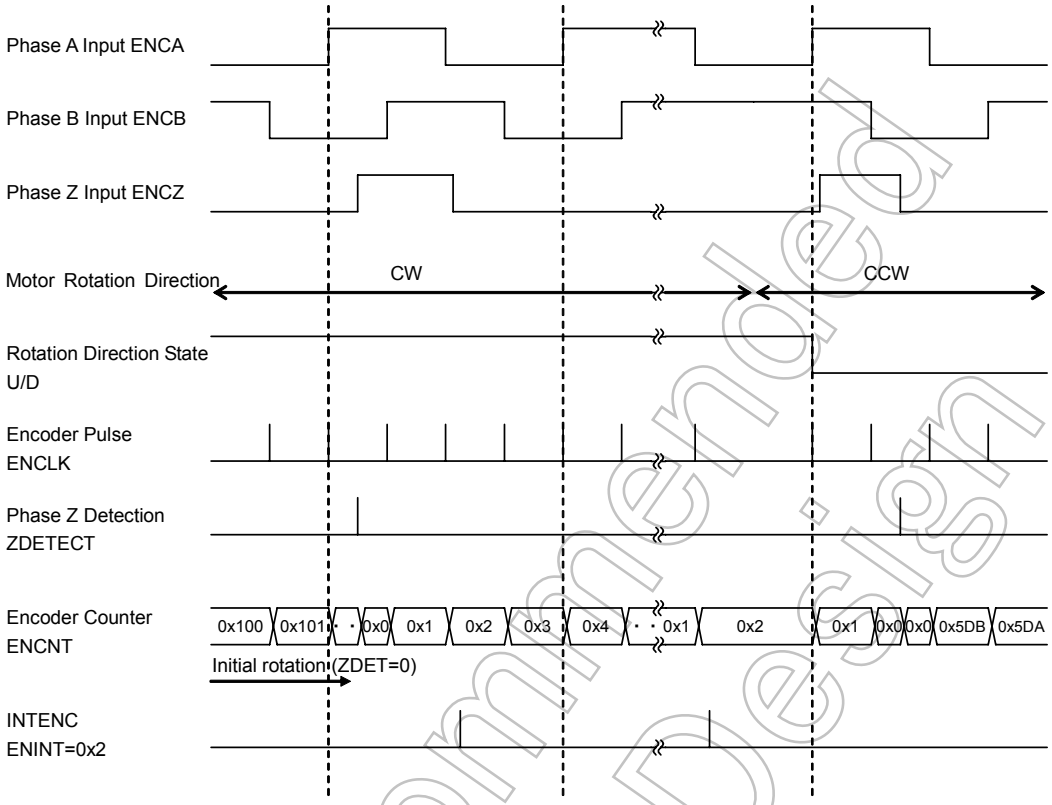


Figure 15.3.1 Encoder Input Circuit Timing Chart (1) (ZEN=1, ENRELOAD=0x05DB)

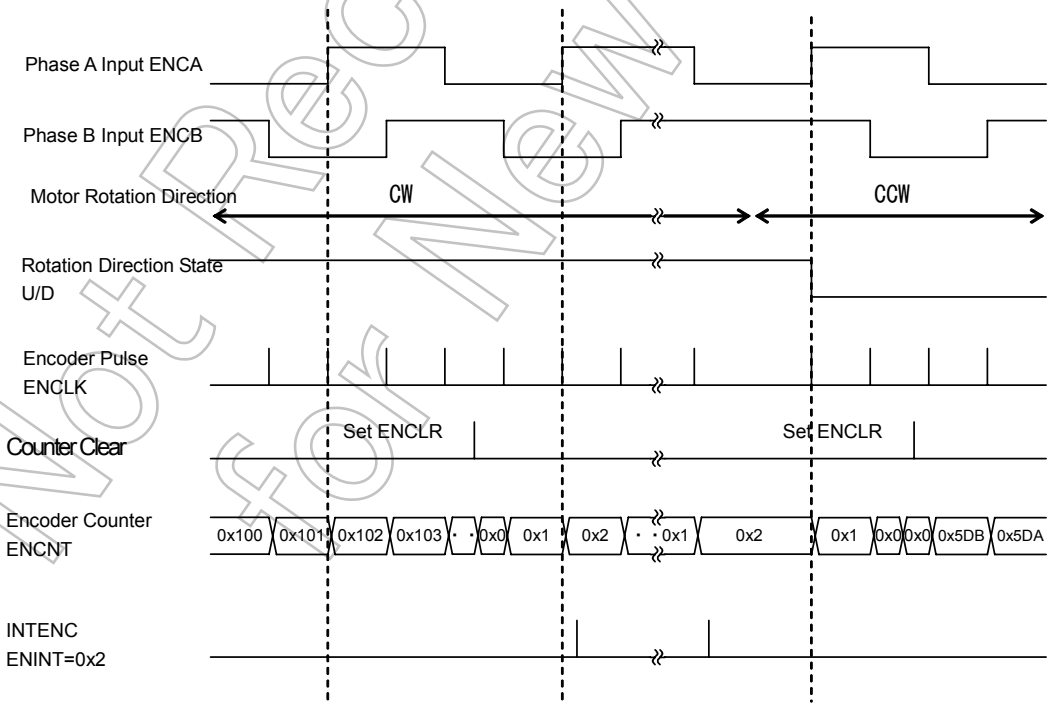


Figure 15.3.2 Encoder Input Circuit Timing Chart (2) (ZEN=0, ENRELOAD=0x5DB)

Not Recommended for New Design

(1) Each incremental encoder signal is connected to phase A, phase B, and phase Z, respectively. This signal is multiplied by four for being counted.
(2) When the motor is rotating in CW direction (phase A is 90 degrees ahead of phase B), the encoder counter operates as an up-counter. When the count value reaches the ENERELOAD value, the counter is zero-cleared on the next ENCLK timing for counting up.
(3) When the motor is rotating in CCW direction (phase A is 90 degrees behind phase B), the encoder counter operates as a down-counter. When the count value reaches 0, the counter is re-loaded with the ENRELOAD value on the next ENCLK timing for counting down.
(4) When ZEN=1, the encoder counter is zero-cleared on the rising edge of phase Z (ZDETECT) while the motor is rotating in CW direction and on the falling edge of phase Z (ZDETECT) in the case of CCW direction. When ENCK and ZDETECT coincide with each other, no count operation is performed and the counter is zero-cleared.
(5) When a 0 is written to the ENCLR bit, the counter is zero-cleared.
(6) An interrupt request can be generated when the counter value reaches the ENINT value. When ZEN=1, however, an interrupt request cannot be generated until the ZDET bit is set to 1.
(7) The ZDET bit is zero-cleared when a 1 is written to the CUNEN bit and at reset, and it is set to 1 on the next ZDETECT timing (regardless of the ZEN value).
(8) The U/D bit is set to 1 when CW rotation is detected and to 0 when CCW rotation is detected.

Not Recommended for New Design

15.4 How to Use the Encoder Input Circuit

15.4.1 Using the Encoder Interrupt Request

(1) Set the encoder pulse count.

Assuming that the encoder requires 1200 pulses for one rotation, the pulse count (after being multiplied by 4) is set as follows:

ENRELOAD=1200x4=0x12C0

0	0	0	1	0	0	1	0	1	0	1	1	1	1	1	1

(2) Set the encoder compare register value.

For generating an interrupt request at counter value = 1000 ($03E8_H$), the encoder compare register is set as follows:

ENINT=0x03E8

0	0	0	0	0	0	1	1	1	1	1	0	1	0	0	0

(3) Enable the encoder counter, interrupt request, and Z-phase detection.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENTNCR	-	-	-	-	-	-	-	-	1	1	-	-	1	-	-	-

(4) Operation

1. As the encoder rotates, the encoder decoder outputs x4 pulses (ENCLK) and rotation direction (U/D).
2. The encoder counter counts ENCLK pulses. Whether to count up or down is determined by rotation direction.
3. When phase Z is detected, the encoder counter is cleared. In the case of up-counting, when the count value matches the ENRELOAD register value, the counter is zero-cleared on the next ENCLK timing. In the case of down-counting, when the count value reaches 0, the counter is re-loaded with the ENRELOAD register value on the next ENCLK timing.
4. Up to the first Z-phase detection, the encoder counter value indicates not the absolute position of the actual encoder but the relative position from count start time. When phase Z is detected, the ZDET is set to 1 and the counter value now indicates the absolute position.
5. When a match occurs between the encoder compare register (ENINT) value and the encoder count value and the ZDET is set to 1, an ENINT interrupt request is generated. This interrupt request is generated regardless whether the counter is counting up or down.

Not Recommended for New Design

16. ROM Correction

This chapter describes the ROM correction function supported by the TMP19A71.

Note: The registers for the ROM correction function must be accessed as a 32-bit quantity. Bit manipulation instructions cannot be used on these registers.

16.1 Features

- Up to eight 8-word sequences of data can be replaced.
- When the physical address stored in an address register (ADDREGn) matches the program counter (PC) value or the address generated by the DMAC (the lower five bits of the address are "don't care"), the data at the specified address in the on-chip ROM is replaced with the data from the RAM area corresponding to the address register.
- Writing an address to an address register causes ROM correction for the address to be enabled automatically. A reset is required to disable the ROM correction function.
- A correction requiring the replacement of more than eight words can be performed by replacing the ROM data with an instruction code which makes a branch to a specified location in the RAM area which contains substitution data.

16.2 Operation

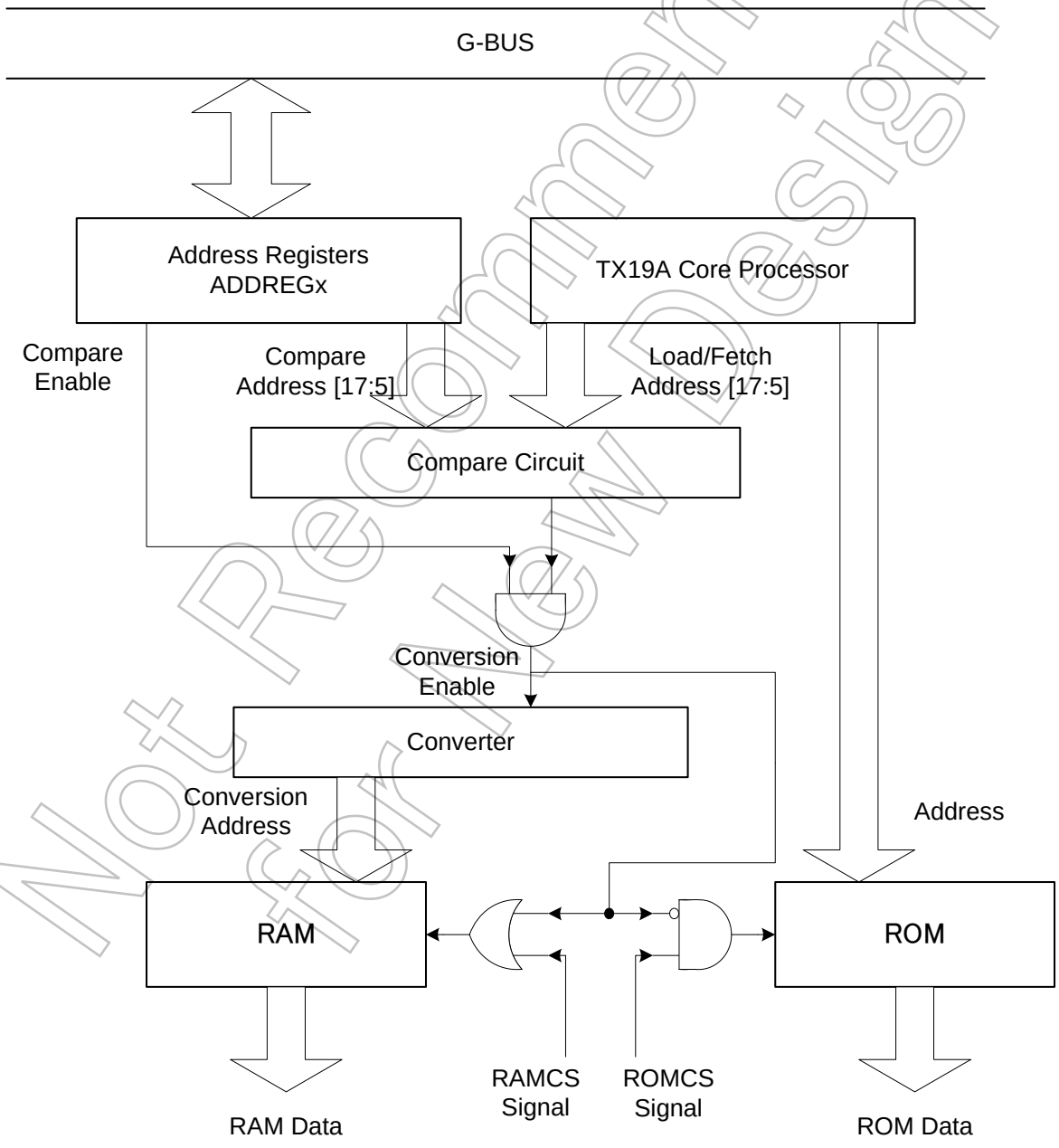
To correct data in a ROM area (or a projected ROM area), store the physical address of the area in an address register (ADDREGn). Store the substitution data in the RAM area corresponding to the address register. Writing an address to an address register causes ROM correction for the address to be enabled automatically. Upon reset, the ROM correction function is disabled. If the initial routine executed upon reset is used to correct ROM data, write a physical address to the relevant address register after a reset is released. The address registers to which addresses are written are enabled for ROM correction. When the stored address matches the PC value (if the TX19A core processor owns the bus) or the source or destination address issued by the DMAC (if the DMAC owns the bus), the data at the specified address in the ROM is replaced with the data stored in the corresponding RAM area. For example, storing addresses in the ADDREG0 and ADDREG3 enables correction for the respective ROM areas, so that the ROM correction circuit block constantly monitors the PC and DMAC-issued addresses for a match with a specified address and, if a match is detected, replaces data, while ignoring the ADDREG2 and ADDREG4 to ADDREG7. Each address register has bits 31:5 although only bits 17:5 are used for address comparison, in order to simplify the circuit. A match detected in the ROM correction circuit is internally ANDed with the ROMCS signal, which indicates a specified ROM address block, to determine an exact match. ROM addresses specified for correction must be located on eight-word boundaries, i.e., the lower five bits are 0. In other words, ROM data is always replaced in 32-byte units. If only part of 32 bytes needs to be replaced, substitution RAM data corresponding to the other bytes must be the same as the current data in the corresponding ROM addresses.

Not Recommended for New Design

The following table shows the relationship between the address registers and RAM areas.

Table 16.2.1 Relationship between the ADDREGn Registers and RAM Areas

Address Register	RAM Area
ADDREG0	0xFFFF_BF00 to 0xFFFF_BF1F
ADDREG1	0xFFFF_BF20 to 0xFFFF_BF3F
ADDREG2	0xFFFF_BF40 to 0xFFFF_BF5F
ADDREG3	0xFFFF_BF60 to 0xFFFF_BF7F
ADDREG4	0xFFFF_BF80 to 0xFFFF_BF9F
ADDREG5	0xFFFF_BFA0 to 0xFFFF_BFBF
ADDREG6	0xFFFF_BFC0 to 0xFFFF_BFDF
ADDREG7	0xFFFF_BFE0 to 0xFFFF_BFFF



Not Recommended for New Design

Figure 16.2.1 ROM Correction Block Diagram

16.3 Registers

(1) Address Registers

ADDREG0 (0xFFFF_E540)

	7	6	5	4	3	2	1	0
Bit Symbol	ADD07	ADD06	ADD05	–	–	–	–	–
Read/Write	R/W							
Reset Value	0	0	0	1	1	1	1	1
Function								
	15	14	13	12	11	10	9	8
Bit Symbol	ADD015	ADD014	ADD013	ADD012	ADD011	ADD010	ADD09	ADD08
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								
	23	22	21	20	19	18	17	16
Bit Symbol	ADD023	ADD022	ADD021	Add020	ADD019	ADD018	ADD017	ADD016
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								
	31	30	29	28	27	26	25	24
Bit Symbol	ADD031	ADD030	ADD029	ADD028	ADD027	ADD026	ADD025	ADD024
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								

ADDREG1 (0xFFFF_E544)

	7	6	5	4	3	2	1	0
Bit Symbol	ADD17	ADD16	ADD15	–	–	–	–	–
Read/Write	R/W							
Reset Value	0	0	0	1	1	1	1	1
Function								
	15	14	13	12	11	10	9	8
Bit Symbol	ADD115	ADD114	ADD113	ADD112	ADD111	ADD110	ADD19	ADD18
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								
	23	22	21	20	19	18	17	16
Bit Symbol	ADD123	ADD122	ADD121	ADD120	ADD119	ADD118	ADD117	ADD116
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								
	31	30	29	28	27	26	25	24
Bit Symbol	ADD131	ADD130	ADD129	ADD128	ADD127	ADD126	ADD125	ADD124
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								

Not Recommended for New Design

ADDREG2 (0xFFFF_E548)

	7	6	5	4	3	2	1	0
Bit Symbol	ADD27	ADD26	ADD25	–	–	–	–	–
Read/Write	R/W							
Reset Value	0	0	0	1	1	1	1	1
Function								
	15	14	13	12	11	10	9	8
Bit Symbol	ADD215	ADD214	ADD213	ADD212	ADD211	ADD210	ADD29	ADD28
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								
	23	22	21	20	19	18	17	16
Bit Symbol	ADD223	ADD222	ADD221	ADD220	ADD219	ADD218	ADD217	ADD216
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								
	31	30	29	28	27	26	25	24
Bit Symbol	ADD231	ADD230	ADD229	ADD228	ADD227	ADD226	ADD225	ADD224
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								

ADDREG3 (0xFFFF_E54C)

	7	6	5	4	3	2	1	0
Bit Symbol	ADD37	ADD36	ADD35	–	–	–	–	–
Read/Write	R/W							
Reset Value	0	0	0	1	1	1	1	1
Function								
	15	14	13	12	11	10	9	8
Bit Symbol	ADD315	ADD314	ADD313	ADD312	ADD311	ADD310	ADD39	ADD38
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								
	23	22	21	20	19	18	17	16
Bit Symbol	ADD323	ADD322	ADD321	Add320	ADD319	ADD318	ADD317	ADD316
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								
	31	30	29	28	27	26	25	24
Bit Symbol	ADD331	ADD330	ADD329	ADD328	ADD327	ADD326	ADD325	ADD324
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								

Not Recommended for New Design

ADDREG4 (0xFFFF_E550)

	7	6	5	4	3	2	1	0
Bit Symbol	ADD47	ADD46	ADD45	–	–	–	–	–
Read/Write	R/W							
Reset Value	0	0	0	1	1	1	1	1
Function								
	15	14	13	12	11	10	9	8
Bit Symbol	ADD415	ADD414	ADD413	ADD412	ADD411	ADD410	ADD49	ADD48
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								
	23	22	21	20	19	18	17	16
Bit Symbol	ADD423	ADD422	ADD421	ADD420	ADD419	ADD418	ADD417	ADD416
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								
	31	30	29	28	27	26	25	24
Bit Symbol	ADD431	ADD430	ADD429	ADD428	ADD427	ADD426	ADD425	ADD424
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								

ADDREG5 (0xFFFF_E554)

	7	6	5	4	3	2	1	0
Bit Symbol	ADD57	ADD56	ADD55	–	–	–	–	–
Read/Write	R/W							
Reset Value	0	0	0	1	1	1	1	1
Function								
	15	14	13	12	11	10	9	8
Bit Symbol	ADD515	ADD514	ADD513	ADD512	ADD511	ADD510	ADD59	ADD58
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								
	23	22	21	20	19	18	17	16
Bit Symbol	ADD523	ADD522	ADD521	ADD520	ADD519	ADD518	ADD517	ADD516
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								
	31	30	29	28	27	26	25	24
Bit Symbol	ADD531	ADD530	ADD529	ADD528	ADD527	ADD526	ADD525	ADD524
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								

Not Recommended for New Design

ADDREG6 (0xFFFF_E558)

	7	6	5	4	3	2	1	0
Bit Symbol	ADD67	ADD66	ADD65	–	–	–	–	–
Read/Write	R/W							
Reset Value	0	0	0	1	1	1	1	1
Function								
	15	14	13	12	11	10	9	8
Bit Symbol	ADD615	ADD614	ADD613	ADD612	ADD611	ADD610	ADD69	ADD68
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								
	23	22	21	20	19	18	17	16
Bit Symbol	ADD623	ADD622	ADD621	ADD620	ADD619	ADD618	ADD617	ADD616
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								
	31	30	29	28	27	26	25	24
Bit Symbol	ADD631	ADD630	ADD629	ADD628	ADD627	ADD626	ADD625	ADD624
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								

ADDREG7 (0xFFFF_E55C)

	7	6	5	4	3	2	1	0
Bit Symbol	ADD77	ADD76	ADD75	–	–	–	–	–
Read/Write	R/W							
Reset Value	0	0	0	1	1	1	1	1
Function								
	15	14	13	12	11	10	9	8
Bit Symbol	ADD715	ADD714	ADD713	ADD712	ADD711	ADD710	ADD79	ADD78
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								
	23	22	21	20	19	18	17	16
Bit Symbol	ADD723	ADD722	ADD721	ADD720	ADD719	ADD718	ADD717	ADD716
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								
	31	30	29	28	27	26	25	24
Bit Symbol	ADD731	ADD730	ADD729	ADD728	ADD727	ADD726	ADD725	ADD724
Read/Write	R/W							
Reset Value	0	0	0	0	0	0	0	0
Function								

Note: DMA transfer cannot be performed to an address register. DMA transfer can be performed to a data area in the RAM which contains substitution data. The ROM correction function can be used when either the TX19A core processor or DMAC owns the bus.

Not Recommended for New Design

17. Flash Memory

This chapter describes the hardware configuration and operation of the flash memory contained in the TMP19A71.

17.1 Overview

17.1.1 Features

1) Memory capacity

The TMP19A71 contains 2 Mbits (256 Kbytes) of flash memory, which is divided into two 128-Kbyte blocks. Each block can be independently protected from program and erase operations. While the TX19A core processor can access the flash memory through a full 32-bit data bus, an external flash programmer can only access the flash memory through a 16-bit data bus.

2) Program and erase times

Chip program time (including verify): 5 seconds (typ.)

Chip erase time (including verify): 20 seconds (typ.)

Note: These program and erase times are typical values not including data transfer overhead. The actual chip program and erase times depend on the programming method used.

3) Programming modes

The TMP19A71 flash memory can be programmed while mounted on a user board (On-Board Programming mode) or by using an EPROM programmer (Programmer mode).

- On-Board Programming modes
  - 1) User Boot mode
    A user-created programming algorithm can be used.
  - 2) Single Boot mode
    A Toshiba-defined serial interface protocol is used.
- Programmer mode
  A general-purpose programmer can be used. (T. B. D)

5) Programming method

Programming operations of the TMP19A71 flash memory are controlled by commands except for a few functions. The TMP19A71 contains a command sequencer which recognizes programming commands and automatically executes corresponding sequences of operation. This feature eliminates the need for the user to code complex program and erase sequences.

The TMP19A71 provides an anti-programmer security feature for protecting the on-chip flash memory from being read by programming equipment. The TMP19A71 also allows the user to protect individual blocks of the flash memory from program or erase operations. This block protection feature is implemented by software; the hardware method (high voltage application) is not supported. The anti-programmer security feature is automatically enabled when both of the two blocks are placed under protection. When the Unprotect command is executed, the flash memory is automatically erased before block protection is lifted to ensure data security.

Not Recommended for New Design

Table 17.1.1 Modified/Deleted Auto Programming Features

Available Auto Programming Features	Modified/Deleted Features
• Auto Program • Auto Chip Erase • Auto Block Erase • Auto Multi-Block Erase • Data Polling	Modified: Block protection is available only under software control. Deleted: Erase Resume/Suspend mode

17.1.2 Block Diagram

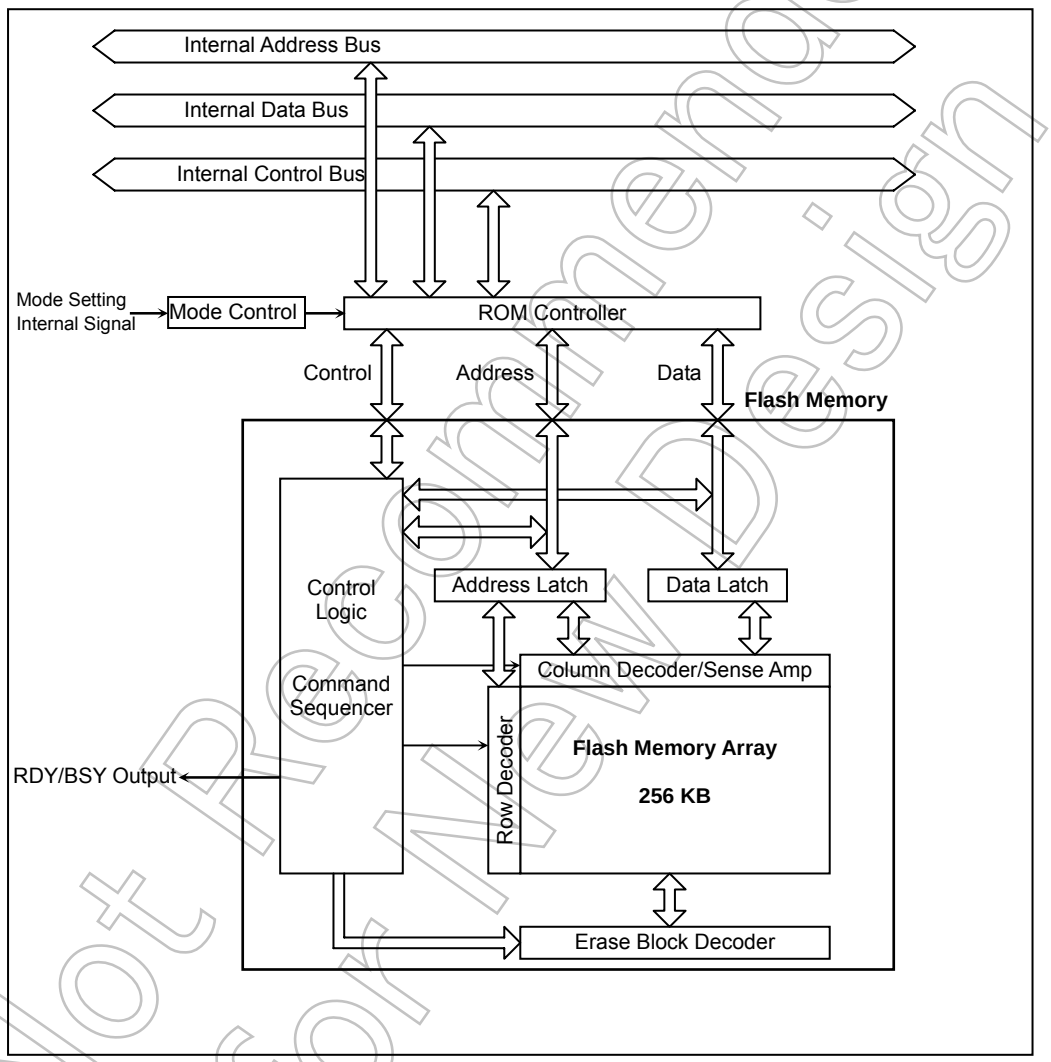


Figure 17.1.1 Flash Memory Block Diagram

Not Recommended for New Design

17.2 Operating Modes

The TMP19A71 offers a total of four operating modes as shown in the table below.

Table 17.2.1 Operating Modes

Operating Mode		Description
Single-Chip Mode		After a reset, the TX19A core processor executes out of the on-chip flash memory.
	Normal Mode	Single-Chip mode is further divided into Normal mode in which the user application executes and User Boot mode which allows for re-programming of the flash memory while the TMP19A71 is installed on a printed circuit board.
	User Boot Mode	The user can freely define how to switch between Normal mode and User Boot mode. For example, the logic state on Port 00 can be used to determine whether to put the flash memory in Normal mode or User Boot mode. In this case, the user must include a routine in the application program to test the state of that port.
Single Boot Mode		After a reset, the TX19A core processor executes out of the on-chip boot ROM (which is a mask ROM). The boot ROM contains a routine to aid users in performing on-board programming of the flash memory via a serial port of the TMP19A71. The serial port is connected to an external host which transfers new data according to a prescribed protocol.
Programmer Mode		This mode allows for re-programming of the flash memory with a general-purpose EPROM programmer. Use the programmer and programming adaptor recommended by Toshiba.

The on-chip flash memory can be programmed in one of the following three modes: User Boot mode, Single Boot mode and Programmer mode. Of these modes, User Boot mode and Single Boot mode allow the flash memory to be programmed while the TMP19A71 is mounted on a printed circuit board. These two modes are collectively referred to as on-board programming modes.

Not Recommended for New Design

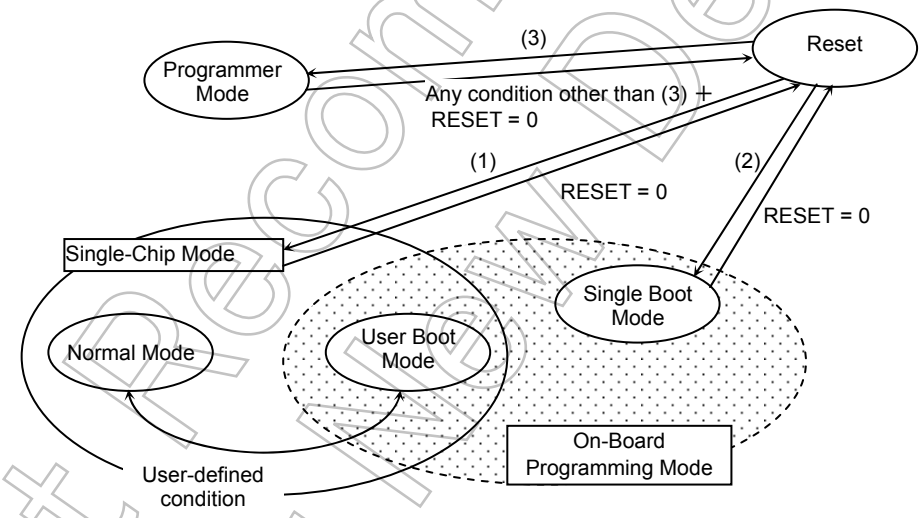
The logic states on the TEST0, P90 to P93 and P94 (BOOT) pins during a reset sequence determine the mode of operation for the flash memory, as shown in Table 17.2.2. After the reset state is released, P94 (BOOT) and P90 to P93 can be configured as general-purpose I/O pins.

After a reset, the TX19A core processor operates in compliance with the selected mode. When Programmer mode is selected, however, the RESET pin must be held at logic 0. The input pins listed in Table 17.2.2 must remain stable once the flash memory is put in a given mode of operation.

Table 17.2.2 Modes of Operation

	Operating Mode	Input Pins							
		RESET	P90	P91	P92	P93	BOOT	TEST0	TEST1
(1)	Single-Chip Mode	0 → 1	(Note)	(Note)	(Note)	(Note)	1	0	0
(2)	Single Boot Mode	0 → 1	(Note)	(Note)	(Note)	(Note)	0	0	0
(3)	Programmer Mode	0	1	1	0	0	(Note)	1	0

Note: Don't care. The pins must be held at 0 or 1.



Parenthesized numbers indicate that the relevant pins are at the logic states shown in Table 17.2.2.

Figure 17.2.1 Mode Transitions

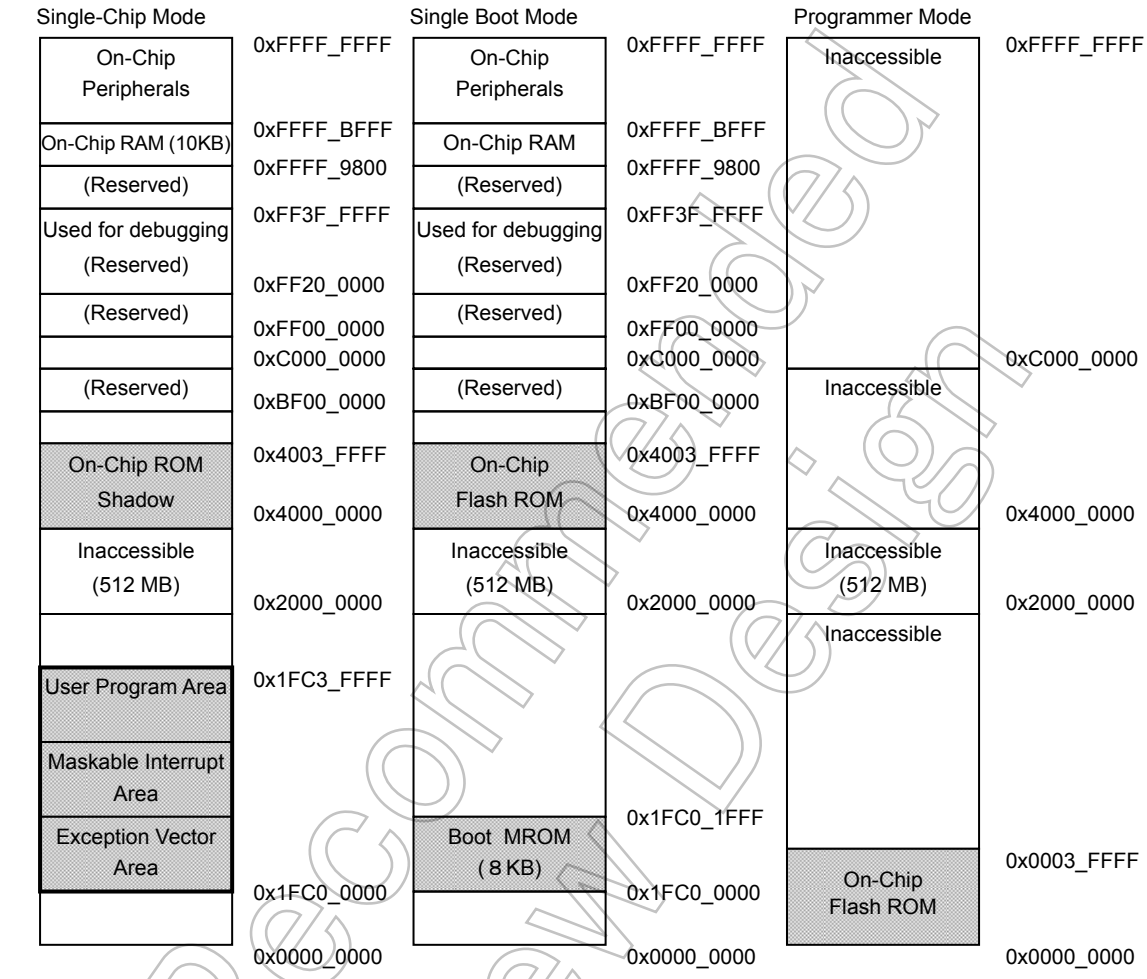
17.2.1 Reset Operation

To reset the TMP19A71, the RESET input must be kept at logic 0 at least for 10 ms after power-up.

Not Recommended for New Design

17.2.2 Memory Maps

The memory map for the TMP19A71 varies according to the mode of operation selected for the on-chip flash memory, as shown below.



Note: The addresses shown above are physical addresses.

Figure 17.2.2 TMP19A71 Memory Maps

When the TMP19A71 is started in Single Boot mode, the boot ROM (mask ROM) is mapped to an 8-Kbyte area starting from the reset vector (0x1FC0_0000), and the flash memory is mapped from 0x4000_0000.

When the TMP19A71 is started in Single-Chip mode, the flash memory is mapped from the reset vector, and the flash memory shadow is mapped from 0x4000_0000.

The following descriptions use virtual addresses, unless otherwise noted.

Not Recommended for New Design

As shown in Figure 17.2.3, the TMP19A71 flash memory is comprised of two 128-Kbyte blocks.

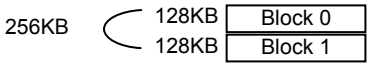


Figure 17.2.3 Flash Memory Block Architecture

Table 17.2.3 Block Addresses Based on Mode Setting

	Single-Chip mode	Single Boot mode	Programmer mode
Block 0	0xBFC0_0000 to 0xBFC 1 _FFFF (Shadow: 0x0000_0000 to 0x0001_FFFF)	0x0000_0000 to 0x000 1 _FFFF	0x0000_0000 to 0x0001_FFFF
Block 1	0xBFC2_0000 to 0xBFC3_FFFF (Shadow: 0x0002_0000 to 0x0003_FFFF)	0x0002_0000 to 0x0003_FFFF	0x0002_0000 to 0x0003_FFFF

17.2.3 Block Protection

The TMP19A71 flash memory is comprised of two 128-Kbyte blocks. To protect stored data from any program and erase operations, each block has a protect bit, which can be set by executing the Block Protect command sequence. Blocks in protection mode are protected from even the Chip Erase and Multi-Block Erase commands; these commands erase only unprotected blocks. Since protection status is stored in flash memory cells, it is retained if the chip is powered off. When both blocks are protected, the data stored in them cannot be read in Programmer mode, which provides a security feature (hereinafter referred to as the anti-programmer security feature).

17.2.4 Security Features When the TX19A Core Processor Is Active

Table 17.2.4 shows the security features available when the TX19A core processor is active. BLKA indicates Block 0 (at addresses 0xBFC0_0000 to 0xBFC1_FFFF), and BLKB indicates Block 1 (at addresses 0xBFC2_0000 to 0xBFC3_FFFF).

In Programmer mode in which a general-purpose EPROM programmer is used, the security features available differ from those shown in the table below.

Table 17.2.4 Security Features of the TX19A Core Processor

DSU Function	Enabled				Disabled			
BLKA Write Protect	OFF		ON		OFF		ON	
BLKB Write Protect	OFF	ON	OFF	ON	OFF	ON	OFF	ON
Use of DSU	Yes	Yes	Yes	--	--	--	--	No
BLKA Read	Yes	Yes	Yes	--	--	--	--	Yes
BLKB Read	Yes	Yes	Yes	--	--	--	--	Yes
BLKA Program (Write)	Yes	Yes	No	--	--	--	--	No
BLKB Program (Write)	Yes	No	Yes	--	--	--	--	No
BLKA Erase	Yes	Yes	×	--	--	--	--	No
BLKB Erase	Yes	No	Yes	--	--	--	--	No
Chip Erase	Yes	Yes*1	Yes*1	--	--	--	--	No
BLKA Protect	Yes	Yes	Yes	--	--	--	--	Yes
BLKB Protect	Yes	Yes	Yes	--	--	--		Yes
Unprotect (All Blocks)	Yes	Yes	Yes	--	--	--	--	Yes
ID Read/Protect Verify	Yes	Yes	Yes	--	--	--	--	Yes

Yes: Can be used
No: Cannot be used
--: Not supported
*1: The Chip Erase command erases only unprotected blocks.

Not Recommended for New Design

DSU (EJTAG)-Probe Interface

The DSU-probe interface is used solely for software debugging using an external DSU-probe unit. Consult the DUS-probe operation manual for the details on debugging using the DSU-probe. When the TMP19A71 is in DUS (EJTAG) mode, the on-chip flash memory provides a security feature.

(1) Permitting and prohibiting the use of a DSU-probe

The TMP19A71 supports on-board debugging while it is installed on a printed circuit board. The TMP19A71 provides a feature to prohibit the use of a DSU-probe to prevent intrusive access to the flash memory. In DSU Prohibit mode, a DSU-probe is denied access to the entirety of the flash memory.

(2) DSU Prohibit mode (Disabling debugging with a DSU-probe)

Once program debugging is completed, write the Protect command to both blocks. This turns on the anti-programmer security feature. While the flash memory is in the secure state, a DSU-probe cannot read its contents. When the chip is powered off and powered on again, the flash memory is put in DSU Prohibit mode, which disables debugging using a DSU-probe until the flash memory exits DSU Prohibit mode.

(3) DSU Permit mode (Enabling debugging with a DSU-probe)

The flash memory can only be brought out of DSU Prohibit mode by clearing the DSUOFF bit in the SEQMOD to 0 and then writing a special code (0x0000_00C5) to the DSU Security Control (SEQCNT) register. This prevents runaway software from inadvertently turning off the DSU Prohibit feature. When the flash memory exits DSU Prohibit mode, the DSU interface is enabled. The flash memory can be secured again by setting the DSUOFF bit in the SEQMOD to 1 and writing 0x0000_00C5 to the SEQCNT while the chip is powered.

DSU Security Mode Register

SEQMOD (0xFFFF_E510)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	—	—	—	DSUOFF
Read/Write	R	R	R	R	R	R	R	R/W
Reset Value	0	0	0	0	0	0	0	1
Function								1: DSU disabled 0: DSU enabled
	15	14	13	12	11	10	9	8
Bit Symbol	—	—	—	—	—	—	—	—
Read/Write	R	R	R	R	R	R	R	R
Reset Value	0	0	0	0	0	0	0	0
Function								
	23	22	21	20	19	18	17	16
Bit Symbol	—	—	—	—	—	—	—	—
Read/Write	R	R	R	R	R	R	R	R
Reset Value	0	0	0	0	0	0	0	0
Function								
	31	30	29	28	27	26	25	24
Bit Symbol	—	—	—	—	—	—	—	—
Read/Write	R	R	R	R	R	R	R	R
Reset Value	0	0	0	0	0	0	0	0
Function								

Note 1: The setting of the DSUOFF bit takes effect after the SEQCNT register is set.
Note 2: This register must be accessed as a 32-bit quantity. Bits 1 to 31 are read as 0.
Note 3: In the flash-version device, this register is initialized by a power-on reset.
Note 4: This register does not support bit manipulation instructions.

Not Recommended for New Design

DSU Security Control Register

SEQCNT (0xFFFF_E514)

	7	6	5	4	3	2	1	0
Bit Symbol	—							
Read/Write	W							
Reset Value	—							
Function	Must be written as 0x0000_00C5.							
	15	14	13	12	11	10	9	8
Bit Symbol	—							
Read/Write	W							
Reset Value	—							
Function	Must be written as 0x0000_00C5.							
	23	22	21	20	19	18	17	16
bit Symbol	—							
Read/Write	W							
Reset Value	—							
Function	Must be written as 0x0000_00C5.							
	31	30	29	28	27	26	25	24
Bit Symbol	—							
Read/Write	W							
Reset Value	—							
Function	Must be written as 0x0000_00C5.							

Note 1: This register must be accessed as a 32-bit quantity.

Note 2: This register does not support bit manipulation instructions.

(4) Application example

The following flowchart shows an example of how to use the security feature with a DSU-probe.

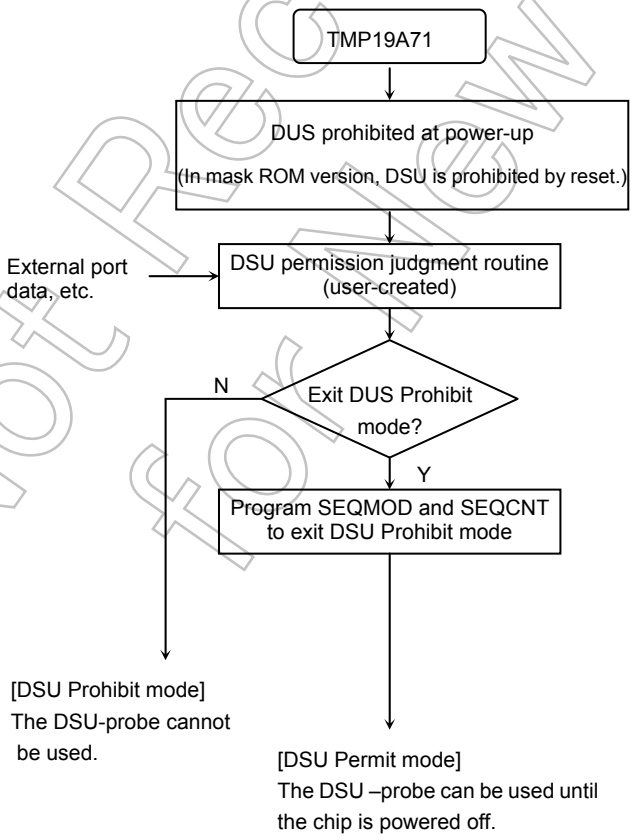


Figure 17.2.4 Using the DSU Prohibit Feature

Not Recommended for New Design

17.3 On-Board Programming Mode

On-board programming allows re-programming of the flash memory while the TMP19A71 is soldered on a printed circuit board. The TMP19A71 provides two types of on-programming mode. In Single Boot mode, new data comes from a serial port under control of a Toshiba-provided routine in the boot ROM. User Boot mode allows you to create an algorithm of your own for flash memory erase and program operations.

The TMP19A71 flash memory provides an anti-programmer security feature to prevent intrusive access to the flash memory while in Programmer mode. This security feature can be enabled upon completion of on-board programming to protect ROM data from being read by third parties.

17.3.1 User Boot Mode (Single-Chip Mode)

User Boot mode allows you to create a programming algorithm of your own. User Boot mode is one of the two submodes in Single-Chip mode; the other submode is Normal mode in which the TX19A core processor executes the user application. To re-program the flash memory, the mode of operation must be switched from Normal mode to User Boot mode. The user application code must include a mode judgment routine.

The user must define the conditions for mode switching, based on the logic states on I/O ports of the TMP19A71. Additionally, the user must incorporate a programming algorithm into the user application code. After User Boot mode is entered, this programming algorithm is copied into the on-chip RAM to re-program the flash memory.

The flash memory cannot be read while it is being erased or programmed. While the flash memory is being erased or programmed, all interrupts including nonmaskable interrupts must be disabled. Once re-programming is complete, it is recommended to protect flash memory blocks as required from accidental corruption.

The pages that follow describe the general procedures for two cases where the programming routine is: a) stored within the TMP19A71 flash memory, and b) loaded from an external controller. For a detailed description of program and erase operations, see Section 17.4 On-Board Programming and Erasure.

Not Recommended for New Design

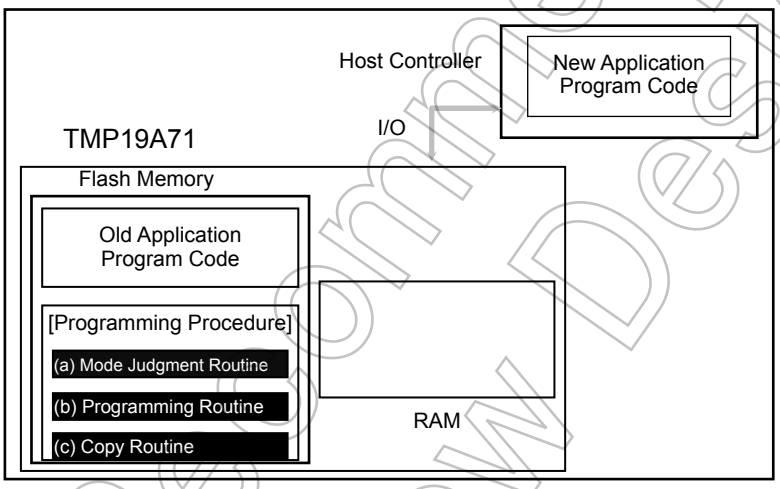
User Boot Mode

(1-A) Method 1: Storing a programming routine in the flash memory

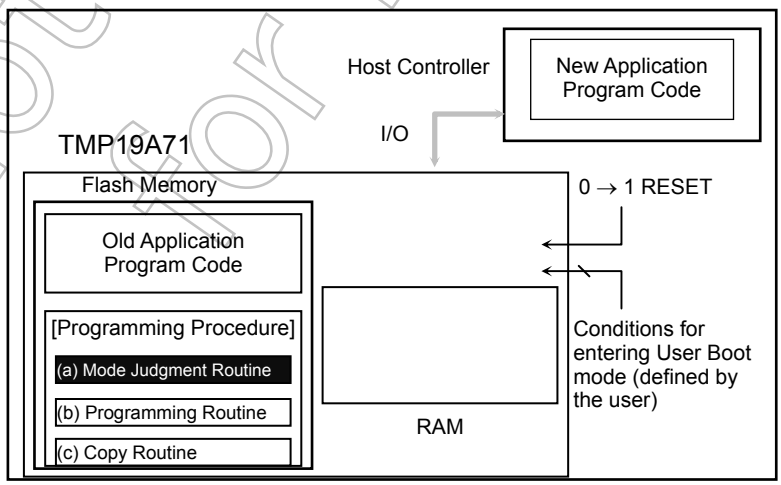
(1) Determine the conditions (e.g., pin states) required for the flash memory to enter User Boot mode and the I/O bus to be used to transfer new program code. Create hardware and software accordingly. Before installing the TMP19A71 on a printed circuit board, write the following program routines into a flash block using programming equipment.

(a) Mode judgment routine: Code to determine whether or not to switch to User Boot mode
(b) Programming routine: Code to download new program code from a host controller and re-program the flash memory
(c) Copy routine: Code to copy the flash programming routine from the TMP19A71 flash memory to the TMP19A71 on-chip RAM

Routines (a), (b) and (c) are collectively called the programming procedure.

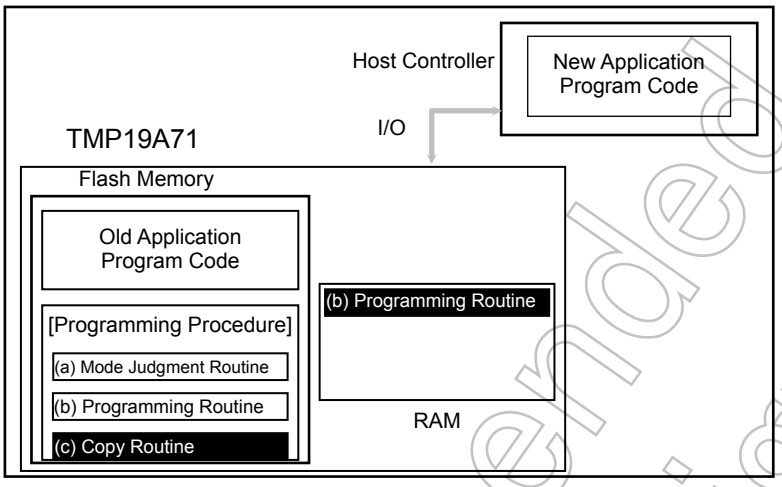


(2) The mode judgment routine written in one of the blocks in the flash memory determines whether to put the TMP19A71 flash memory in User Boot mode. If the specified conditions are met, the flash memory enters User Boot mode.

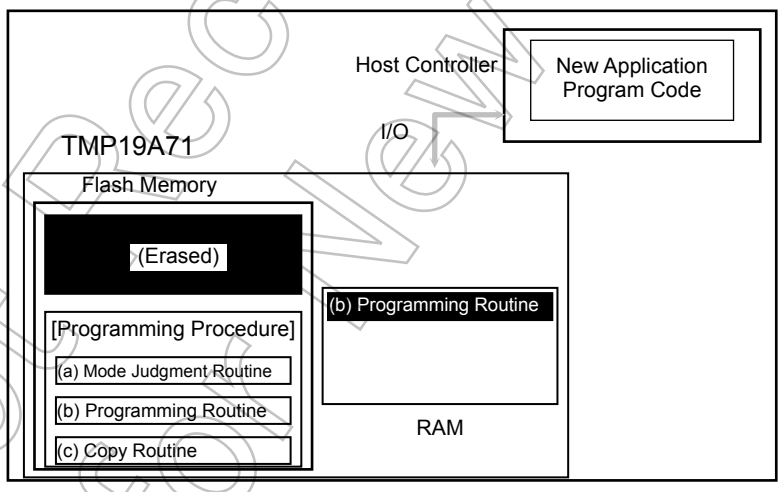


Not Recommended for New Design

(3) Once User Boot mode is entered, execute the copy routine to copy the flash programming routine to the TMP19A71 on-chip RAM.

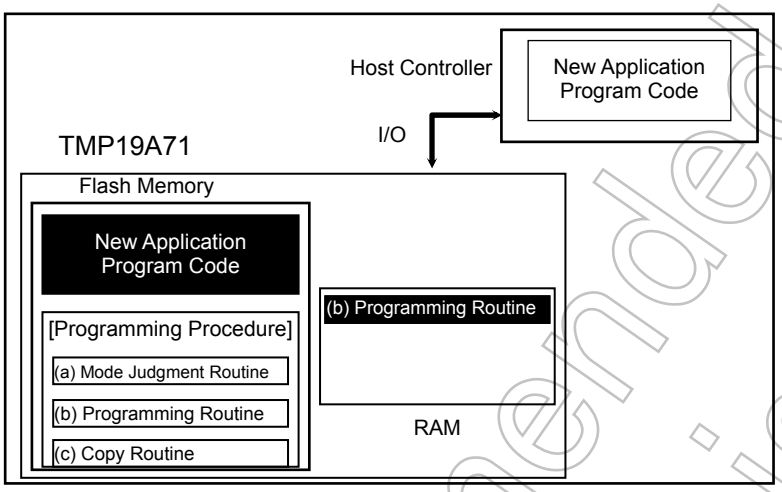


(4) Jump program execution to the flash programming routine in the on-chip RAM to erase the flash block containing the old application program code. All interrupts including RESET and NMI must be disabled until new application program code has been written to the flash memory.

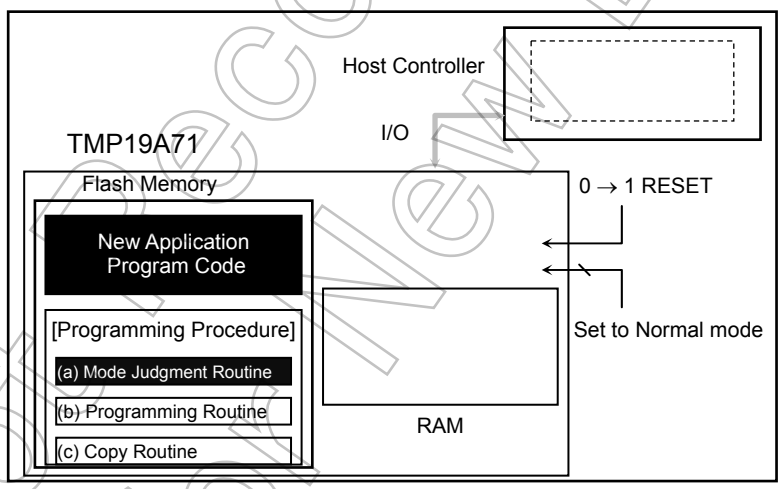


Not Recommended for New Design

(5) Continue executing the flash programming routine to download new application program code from the host controller and program it into the erased flash block. Once programming is complete, turn on the protection of that flash block.



(6) Drive the $\overline{\text{RESET}}$ pin low to reset the TMP19A71 and enter Normal mode or jump to an arbitrary address to execute the new user application program.



Not Recommended for New Design

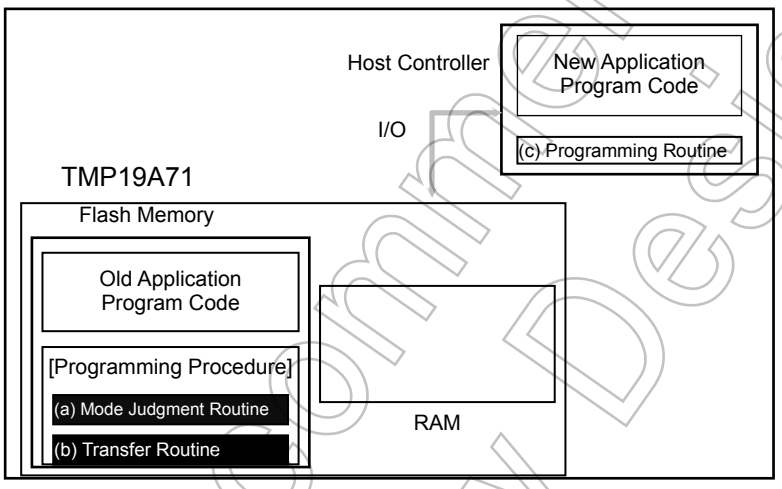
(1-B) Method 2: Transferring a programming routine from an external host

(1) Determine the conditions (e.g., pin states) required for the flash memory to enter User Boot mode and the I/O bus to be used to transfer new program code. Create hardware and software accordingly. Before installing the TMP19A71 on a printed circuit board, write the following program routines into a flash block using programming equipment.

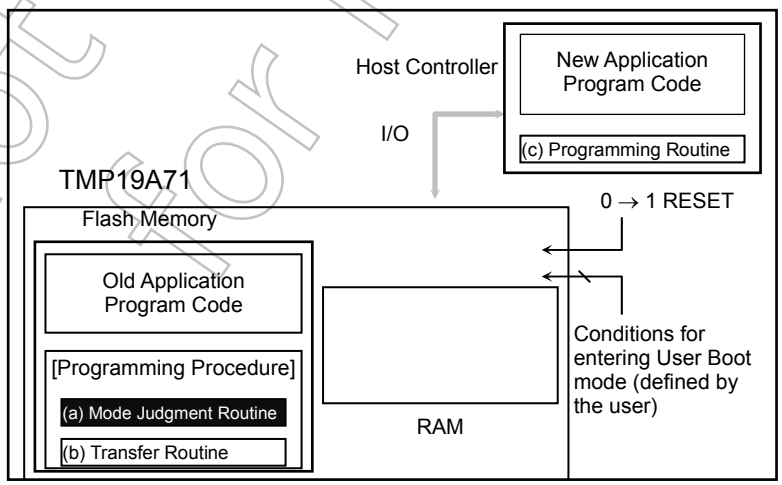
(a) Mode judgment routine: Code to determine whether or not to switch to User Boot mode
(b) Transfer routine: Code to download new program code from a host controller

Routines (a) and (b) are collectively called the programming procedure.

Also, prepare the following routine on the host controller.
(c) Programming routine: Code to re-program the flash memory

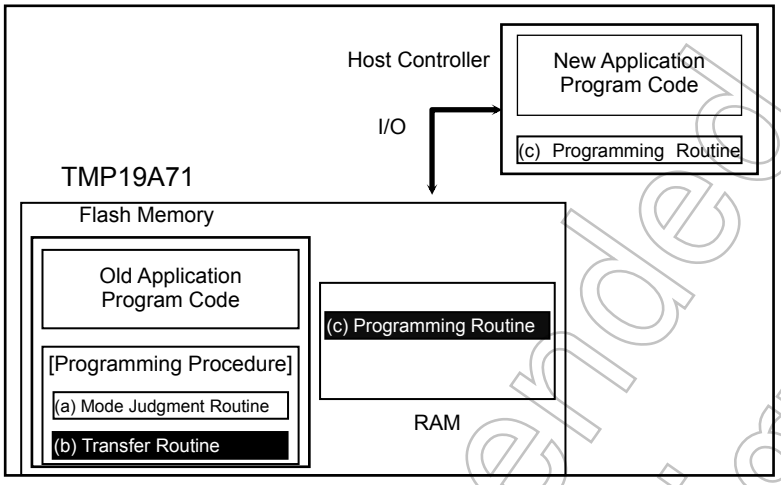


(2) The mode judgment routine written in one of the blocks in the flash memory determines whether to put the TMP19A71 flash memory in User Boot mode. If the specified conditions are met, the flash memory enters User Boot mode.

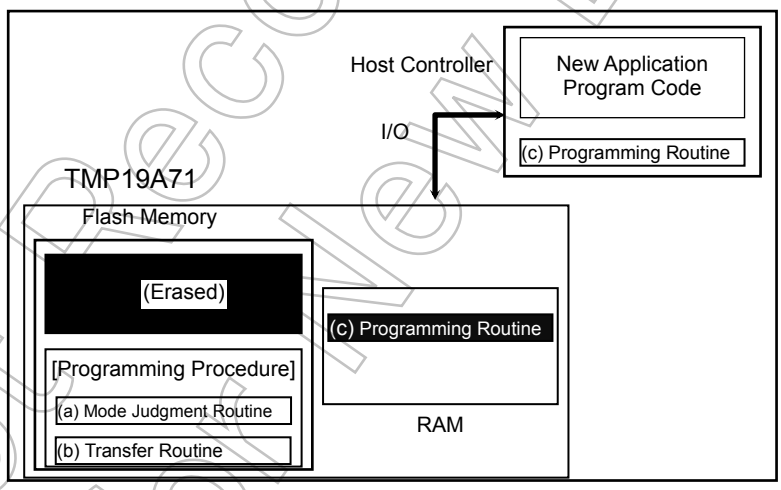


Not Recommended for New Design

(3) Once User Boot mode is entered, execute the transfer routine to download the flash programming routine from the host controller to the TMP19A71 on-chip RAM.

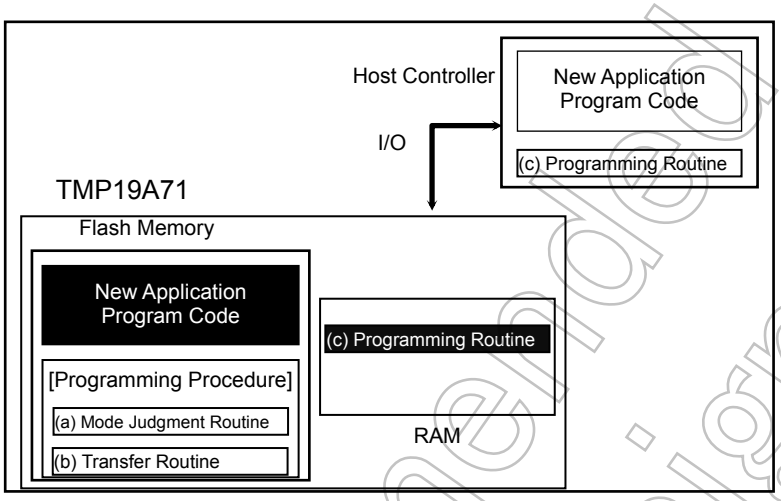


(4) Jump program execution to the flash programming routine in the on-chip RAM to erase a flash block containing the old application program code. All interrupts including RESET and NMI must be disabled until new application program code has been written to the flash memory.

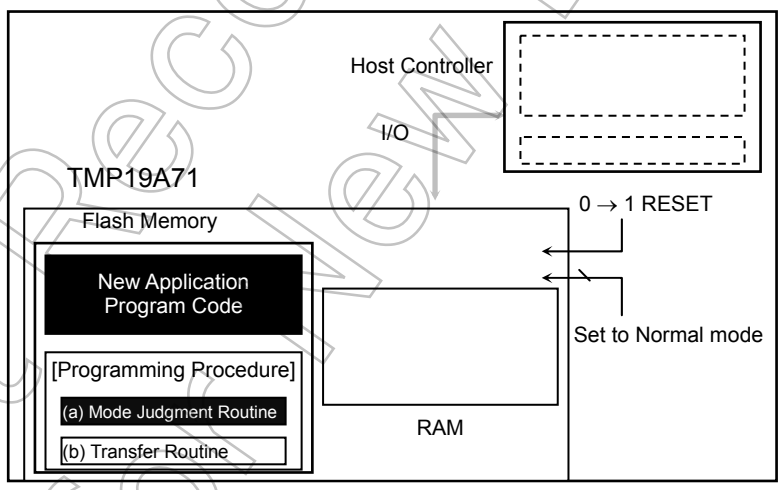


Not Recommended for New Design

(5) Continue executing the flash programming routine to download new application program code and program it into the erased flash block. Once programming is complete, turn on the protection on that flash block.



(6) Drive the $\overline{\text{RESET}}$ pin low to reset the TMP19A71 and enter Normal mode or jump to an arbitrary address to execute the new user application program.



Not Recommended for New Design

17.3.2 Single Boot Mode

In Single Boot mode, the flash memory can be re-programmed by using a program contained in the TMP19A71 on-chip boot ROM. This boot ROM is a masked ROM. When Single Boot mode is selected upon reset, the boot ROM is mapped to the address region including the interrupt vector table while the flash memory is mapped to an address region different from it (see Figure 17.2.2).

Single Boot mode allows for serial programming of the flash memory. The SIO (SIO2) of the TMP19A71 is connected to an external host controller. Via this serial link, a programming routine is downloaded from the host controller to the TMP19A71 on-chip RAM. Then, the flash memory is re-programmed by executing the programming routine. The host sends out both commands and programming data to re-program the flash memory.

Communications between the SIO2 and the host must follow the prescribed protocol described later. To secure the contents of the flash memory, password verification is performed before a programming routine is downloaded into the on-chip RAM. If password verification fails, the transfer of a programming routine itself is aborted. All interrupts including nonmaskable interrupts must be disabled while the boot program is executed.

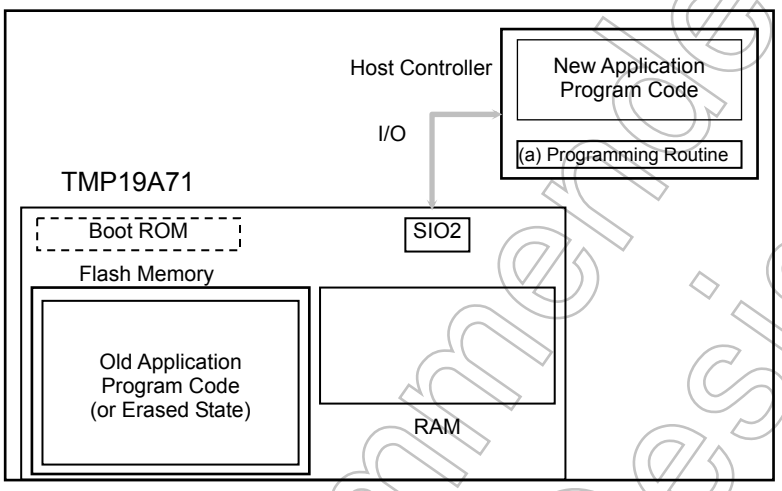
Once re-programming is complete, it is recommended to protect flash memory blocks as required from accidental corruption during subsequent operation in Single-Chip (Normal) mode. For a detailed description of erase and program operations, see section 17.4 On-Board Programming and Erasure.

Not Recommended for New Design

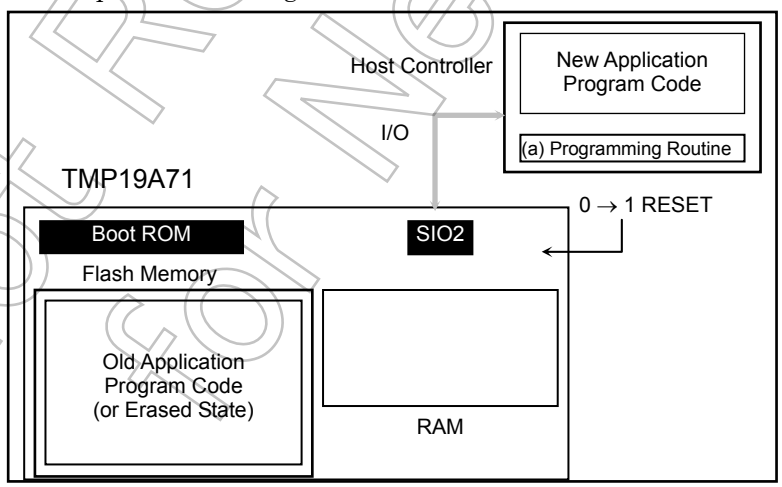
Single Boot Mode

(2-A) General procedure: Using a programming algorithm in the on-chip boot ROM

(1) The flash block containing the older version of the program code need not be erased before executing the programming routine. Since a programming routine and programming data are transferred via the UART2 or SIO2, the UART 2 or SIO2 pins must be connected to a host controller. Prepare a programming routine on the host controller. (The SIO2 is used here.)

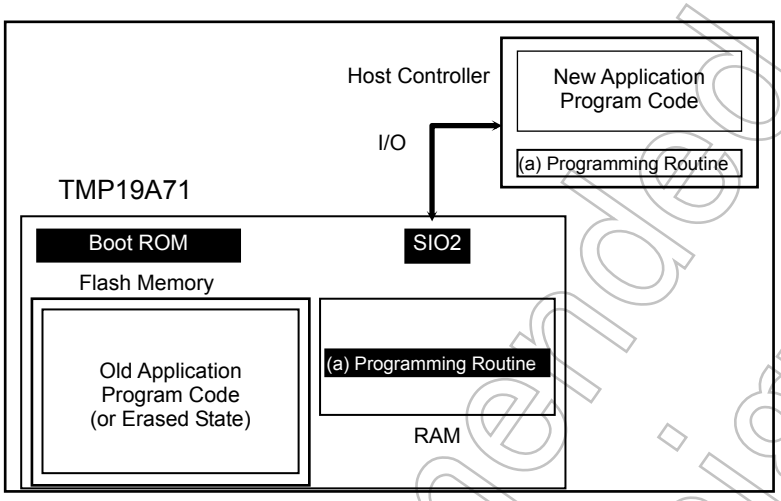


(2) Reset the TMP19A71 with the mode setting pins held at appropriate logic values for re-booting from the on-chip boot ROM. In Single Boot mode, the 12-byte password transferred from the host controller is first compared to the password stored in the user application program in the flash memory. (If the flash memory has already been erased, password verification is performed using the erased data.)

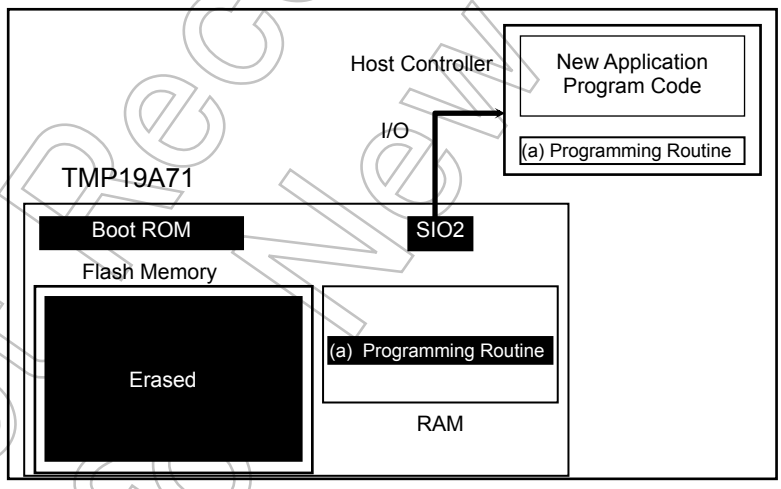


Not Recommended for New Design

(3) If password verification is successful, the boot program loads the programming routine from the host controller into the on-chip RAM. The programming routine must be stored in the address range of 0xFFFF_9800 to 0xFFFF_AFFF.



(4) The TX19A core processor jumps to the programming routine in the on-chip RAM to erase the flash block containing the old application program code. (The Block Erase or Chip Erase command may be used.)

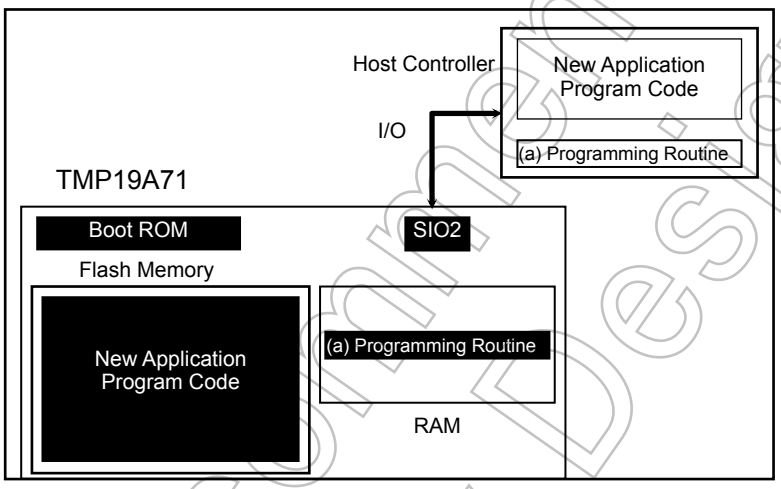


Not Recommended for New Design

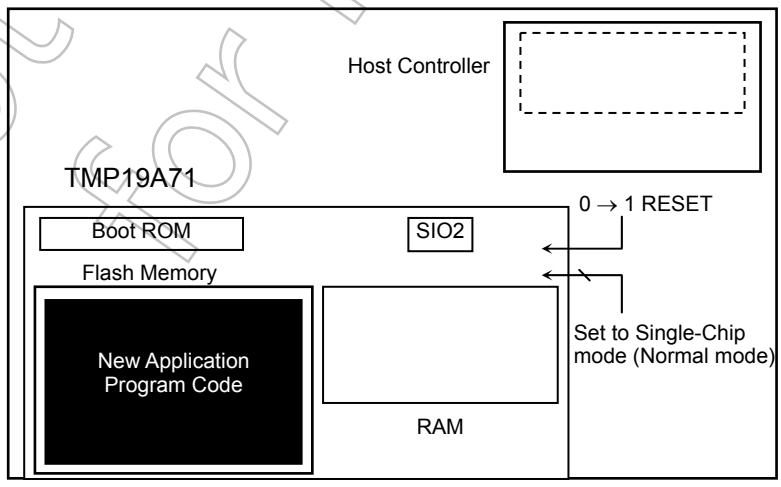
(5) Next, the programming routine loads new application program code from the host controller into the erased flash block. Once programming is complete, the flash block is placed under protection.

In the example below, new program code comes from the same host controller via the same SIO channel as for the programming routine. However, once the programming routine has begun to execute, the transfer path and the source of the transfer can be changed as desired. Create board hardware and a programming routine to suit your particular needs.

Note: The boot ROM provides no vector area for all maskable interrupts and NMI. While the programming routine is executed, no exception should be allowed to occur.



(6) When the flash memory has been programmed, power off the board and disconnect the cable connecting the host controller. Then, turn on the power again and re-boot the TMP19A71 in Single-Chip mode (Normal mode) to execute the new application program.



Not Recommended for New Design

(1) Connection examples in Single Boot mode

In Single Boot mode, serial transfer is used to re-program the flash memory while the TMP19A71 is installed on a printed circuit board. In this mode, the SIO (channel 2) of the TMP19A71 is connected to a host controller (programming tool). The host controller issues commands to the target board to perform programming operations. Figure 17.3.1 and Figure 17.3.2 show examples of host-to-target connection.

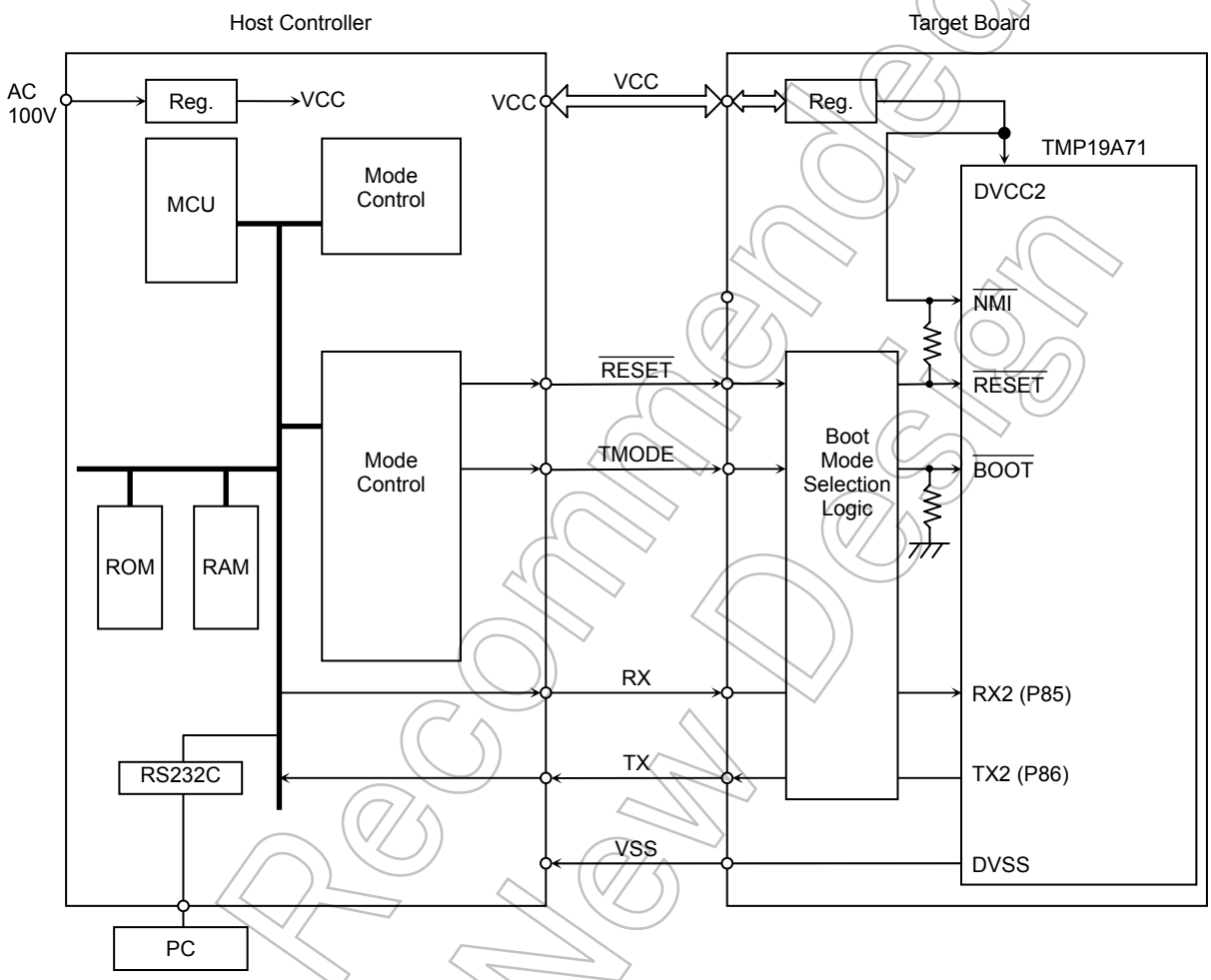


Figure 17.3.1 Example of Connection between a Host Controller and a Target Board in Single Boot Mode (when the SIO2 is configured for UART mode)

Not Recommended for New Design

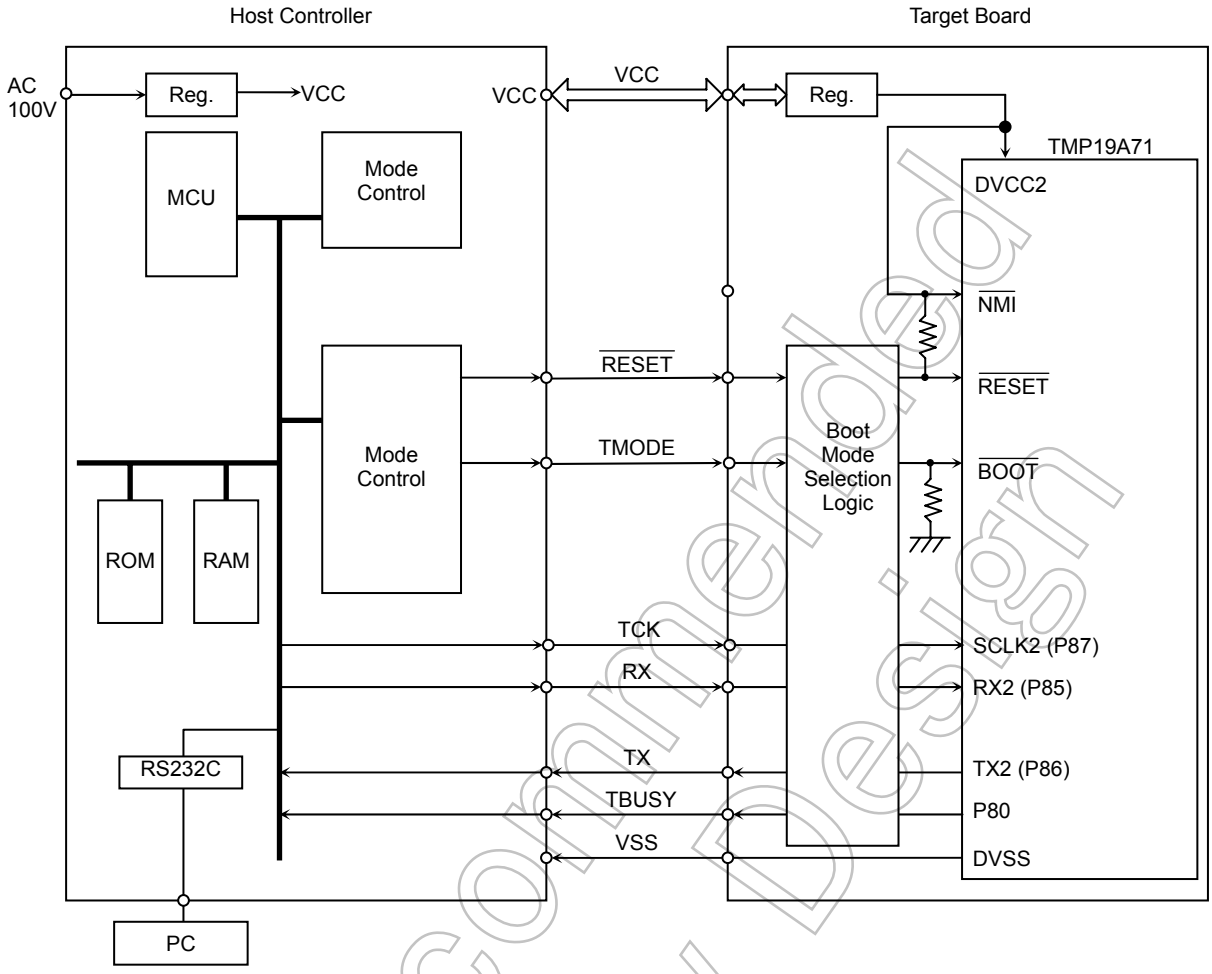


Figure 17.3.2 Example of Connection between a Host Controller and a Target Board in Single Boot Mode (when the SIO2 is configured for I/O Interface mode)

Not Recommended for New Design

(2) Configuring for Single Boot mode

To perform on-board programming, boot up the TMP19A71 in Single Boot mode, as shown below.

TEST0 = 0

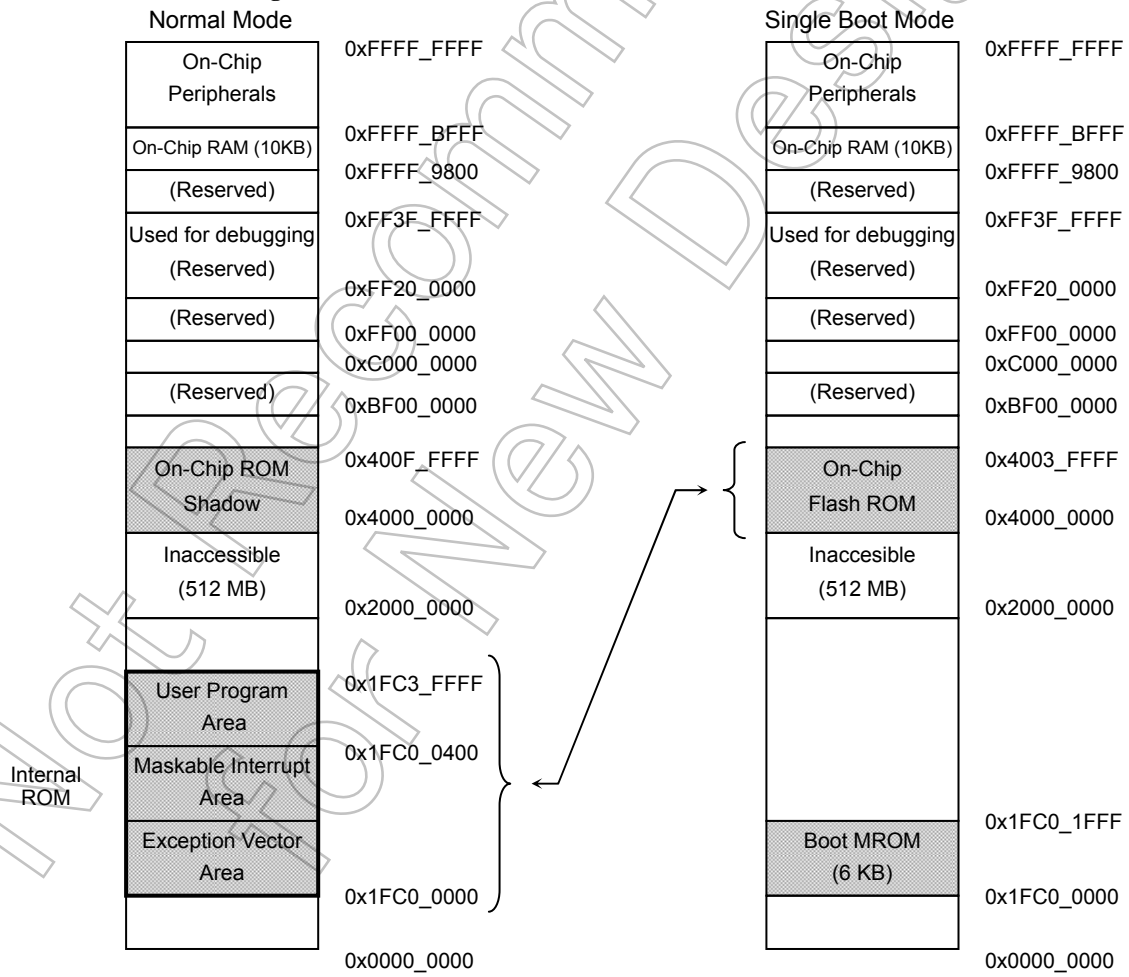
BOOT = 0

RESET = 0 → 1

While the RESET pin is held at logic 0, set the TEST0 and BOOT (P94) pins at the logic values shown above. Then, driving the RESET pin high boots up the TMP19A71 in Single Boot mode.

(3) Memory map

Figure 17.3.3 shows a comparison of the memory maps in Single-Chip (Normal) mode and Single Boot mode. In Single Boot mode, the on-chip flash memory is mapped to physical addresses 0x4000_0000 through 0x400F_FFFF and virtual addresses 0x0000_0000 through 0x000F_FFFF. The on-chip boot ROM (mask ROM) is mapped to physical addresses 0x1FC0_0000 through 0x1FC0_1FFF.



Note: The addresses shown above are physical addresses.

Figure 17.3.3 Memory Maps for Normal and Single Boot Modes

Not Recommended for New Design

(4) Interface Specifications

In Single Boot mode, an SIO channel is used for communications with a programming controller. Both UART (asynchronous) and I/O Interface (synchronous) modes are supported. To perform on-board programming, the interface specifications shown below must also be set on the controller side.

- UART mode

  Communications channel : UART2
  Transfer mode : UART (asynchronous) mode, full-duplex
  Data length : 8 bits
  Parity bit : None
  STOP bit : 1 bit
  Baud rate : Up to 437.5 kbps (at 56 Hz)
  Up to 312.5 kbps (at 40 Hz)
  Others : LSB first, SC2MOD2.WBUF=0

- I/O Interface mode

  Communications channel : SIO2
  Transfer mode : I/O Interface mode, half-duplex
  Synchronization signal (SCLK2) SCLK input mode (SC2CR = 0x01)
  Handshaking signal : P80 configured as an output
  Baud rate : Up to 3 Mbps (at 56 Hz)
  Up to 2.5 Mbps (at 40 MHz)
  Others : LSB first, SC2MOD2.WBUF = 0

Table 17.3.1 Required Pin Connections

Pin		Interface	
		UART Mode	I/O Interface Mode
Power Supply Pins	DVCC3	Required	Required
	DVSS	Required	Required
Mode Setting Pin	$\overline{\text{BOOT}}$	Required	Required
Reset Pin	$\overline{\text{RESET}}$	Required	Required
Communications Pins	TX2	Required	Required
	RX2	Required	Required
	SCLK2	Not Required	Required (Input Mode)
	P80	Not Required	Required (Output Mode)

(5) Data Transfer Format

Table 17.3.2 lists the commands to be issued from the host controller to the target board in Single Boot mode. Tables 17.3.3 to 17.3.5 illustrate the sequence of two-way communications that should occur in response to each command.

Table 17.3.2 Single Boot Mode Commands

Code	Command
0x10	RAM Transfer
0x20	Show Flash Memory Sum
0x30	Show Product Information

Not Recommended for New Design

Table 17.3.3 Transfer Format for the RAM Transfer Command

	Byte	Data Transferred from the Controller to the TMP19A71	Baud Rate	Data Transferred from the TMP19A71 to the Controller
Boot ROM	1st byte	Serial operation mode and baud rate For UART mode 0x86 For I/O Interface mode (Note 2) 0x30	Specified baud rate (Note 1)	—
	2nd byte	—		ACK for the serial operation mode byte For UART mode Normal acknowledge 0x86 (The boot program aborts if the baud rate cannot be set correctly.) For I/O Interface mode Normal acknowledge 0x30
	3rd byte	Command code (0x10)		—
	4th byte	—		ACK for the command code byte (Note 3) Normal acknowledge 0x30 Negative acknowledge 0x11 Communication error 0x18
	5th byte to 16th byte	Password sequence (12 bytes) (0x0000_0474 to 0x0000_047F)		—
	17th byte	Checksum value for 5th to 16th bytes (Note 4)		—
	18th byte	—		ACK for the checksum byte (Note 3) Normal acknowledge 0x10 Negative acknowledge 0x11 Communication error 0x18
	19th byte	RAM storage start address 31 to 24 (Note 5)		—
	20th byte	RAM storage start address 23 to 16 (Note 5)		—
	21st byte	RAM storage start address 15 to 8 (Note 5)		—
	22nd byte	RAM storage start address 7 to 0 (Note 5)		—
	23rd byte	RAM storage byte count 15 to 8 (Note 5)		—
	24th byte	RAM storage byte count 7 to 0 (Note 5)		—
	25th byte	Checksum value for 19th to 24th bytes (Note 4)		—
	26th byte	—		ACK for the checksum byte (Note 3) Normal acknowledge 0x10 Negative acknowledge 0x11 Communication error 0x18
	27th byte to mth byte	RAM storage data		—
	(m + 1)th byte	Checksum value for 27th to mth bytes (Note 4)		—
	(m + 2)th byte	—		ACK for the checksum byte (Note 3) Normal acknowledge 0x10 Negative acknowledge 0x11 Communication error 0x18
RAM	(m + 3)th byte	—		Jump to RAM storage start address

Note 1: In I/O Interface mode, the baud rate for transferring the 1st and 2nd bytes must be 1/16 of the specified baud rate.

Note 2: In I/O interface mode, a waveform that allows the serial operation mode to be determined should be generated.

Note 3: In case of any negative acknowledgment, the boot program returns to a state in which it waits for a command code (3rd byte). In I/O Interface mode, no acknowledgment is returned for a communicaion error.

Note 4: The checksum value is obtained by adding all the bytes of transmitted data together, dropping the carries, and taking the two's complement of the total sum.

Note 5: The 19th to 24th bytes must be within the RAM address range 0xFFFF_9800 to 0xFFFF_AFFF.

Not Recommended for New Design

Table 17.3.4 Transfer Format for the Show Flash Memory SUM Command

	Byte	Data Transferred from the Controller to the TMP19A71	Baud Rate	Data Transferred from the TMP19A71 to the Controller
Boot ROM	1st byte	Serial operation mode and baud rate For UART mode 0x86 For I/O Interface mode (Note 2) 0x30	Specified baud rate (Note 1)	—
	2nd byte	—		ACK for the serial operation mode byte For UART mode Normal acknowledge 0x86 (The boot program aborts if the baud rate cannot be set correctly.) For I/O Interface mode Normal acknowledge 0x30
	3rd byte	Command code (0x20)		—
	4th byte	—		ACK for the command code byte (Note 3) Normal acknowledge 0x20 Negative acknowledge 0x21 Communication error 0x28
	5th byte	—		SUM (upper byte)
	6th byte	—		SUM (lower byte)
	7th byte	—		Checksum value for 5th and 6th bytes (Note 4)
	8th byte	(Wait for the next command code.)		—

Note 1: In I/O Interface mode, the baud rate for transferring the 1st and 2nd bytes must be 1/16 of the specified baud rate.

Note 2: In I/O Interface mode, a waveform that allows the serial operation mode to be determined should be generated.

Note 3: In case of any negative acknowledgment, the boot program returns to a state in which it waits for a command code (3rd byte). In I/O Interface mode, no acknowledgment is returned for a communication error.

Note 4: The checksum value is obtained by adding all the bytes of transmitted data together, dropping the carries, and taking the two's complement of the total sum.

Note 5: The SUM value is the lower 16 bits of a value obtained by adding all the bytes in the flash memory together.
SUM (high) = SUM[15 : 8], SUM (low) = SUM[7 : 0]

Not Recommended for New Design

Table 17.3.5 Transfer Format for the Show Product Information Command (1/2)

	Byte	Data Transferred from the Controller to the TMP19A71	Baud Rate	Data Transferred from the TMP19A71 to the Controller
Boot ROM	1st byte	Serial operation mode and baud rate For UART mode 0x86 For I/O Interface mode (Note 2) 0x30	Specified baud rate (Note 1)	—
	2nd byte	—		ACK for the serial operation mode byte For UART mode Normal acknowledge 0x86 (The boot program aborts if the baud rate cannot be set correctly.) For I/O Interface mode Normal acknowledge 0x30
	3rd byte	Command code (0x30)		—
	4th byte	—		ACK for the command code byte (Note 3) Normal acknowledge 0x30 Negative acknowledge 0x31 Communication error 0x38
	5th byte	—		Flash memory data (at address 0x0000_0470)
	6th byte	—		Flash memory data (at address 0x0000_0471)
	7th byte	—		Flash memory data (at address 0x0000_0472)
	8th byte	—		Flash memory data (at address 0x0000_0473)
	9th byte to 20th byte	—		Product name (12-byte ASCII code) 'TX19A71FY' + 0x20, 0x20, 0x20 from the 9th byte
	21st byte to 24th byte	—		Password comparison start address (4 bytes) 0x74, 0x04, 0x00, 0x00 from the 21st byte
	25th byte to 28th byte	—		RAM start address (4 bytes) 0x00,0x98, 0xFF, 0xFF from the 25th byte
	29th byte to 32nd byte	—		Dummy data (4 bytes) 0xFF, 0xA7, 0xFF, 0xFF from the 29th byte
	33rd byte to 36th byte	—		RAM end address (4 bytes) 0xFF, 0xBF, 0xFF, 0xFF from the 33rd byte
	37th byte to 40th byte	—		Dummy data (4 bytes) 0x00, 0xA8, 0xFF, 0xFF from the 37th byte
	41st byte to 44th byte	—		Dummy data (4 bytes) 0xFF, 0xAF, 0xFF, 0xFF from the 41st byte
	45th byte to 46th byte	—		Fuse information (2 bytes) 0x00, 0x00 from the 45th byte
	47th byte to 50th byte	—		Flash memory start address (4 bytes) 0x00, 0x00, 0x00, 0x00 from the 47th byte
	51st byte to 54th byte	—		Flash memory end address (4 bytes) 0xFF, 0xFF, 0x03, 0x00 from the 51st byte
	55th byte to 56th byte	—		Flash memory block count (2 bytes) 0x02, 0x00 from the 55th byte
	57th byte to 60th byte	—		Start address of a group of the same-size flash blocks (4 bytes) 0x00, 0x00, 0x00, 0x00 from the 57th byte

Not Recommended for New Design

Table 17.3.5 Transfer Format for the Show Device Information Command (2/2)

	Byte	Data Transferred from the Controller to the TMP19A71	Baud Rate	Data Transferred from the TMP19A71 to the Controller
Root ROM	61st byte to 64th byte	—		Size (in half words) of the same-size flash blocks (4 bytes) 0x00, 0x00, 0x01, 0x00 from the 61st byte
	65th byte	—		Number of flash blocks of the same size (1 byte) 0x02
	66th byte	—		Checksum value for the 5th to 65th bytes (Note 4)
	67th byte	(Wait for the next command code.)		—

Note 1: In I/O Interface mode, the baud rate for transferring the 1st and 2nd bytes must be 1/16 of the specified baud rate.

Note 2: In I/O Interface mode, a waveform that allows the serial operation mode to be determined should be generated.

Note 3: In case of any negative acknowledgment, the boot program returns to a state in which it waits for a command code (3rd byte). In I/O Interface mode, no acknowledgment is returned for a communication error.

Note 4: The checksum value is obtained by adding all the bytes of transmitted data together, dropping the carries, and taking the two's complement of the total sum.

Not Recommended for New Design

(6) Overview of the boot program commands

When Single Boot mode is selected, the boot program is automatically executed on startup. The boot program offers the following three commands (each command is explained in detail in the subsections that follow):

1. RAM Transfer command
   The RAM Transfer command stores program code transferred from a host controller to the on-chip RAM and executes the program once the transfer is successfully completed. The maximum program size is 6 Kbytes (0xFFFF_9800 to 0xFFFF_AFFF).

   The RAM Transfer command enables the user to control on-board programming of the flash memory in a unique manner by providing the means for downloading a user-created programming routine. The programming routine must use the flash memory command sequences described in section 17.4 On-Board Programming and Erasure.

   Before initiating a transfer, the RAM Transfer command checks a password sequence coming from the controller against the password stored in the flash memory. If they do not match, the RAM Transfer command aborts.

2. Show Flash Memory Sum command
   The Show Flash Memory Sum command adds the contents of the flash memory and returns the lower 16 bits of the result. The boot program does not provide a command to read out the contents of the flash memory. Instead, the Show Flash Memory Sum command can be used for software revision management.

3. Show Product Information command
   The Show Product Information command provides product information, such as the product name and on-chip memory configuration, stored at addresses 0x0000_0470 to 0x0000_0473 in the flash memory. In addition to the Show Flash Memory Sum command, this command can be used for software revision management.

Not Recommended for New Design

1) RAM Transfer command (see Table 17.3.3)

1. The 1st byte specifies which one of the two serial operation modes is used. For a detailed description of how the serial operation mode is determined, see subsection 5) Determination of a serial operation mode. If it is determined as UART mode, the boot program then checks if the SIO2 is programmable to the baud rate at which the 1st byte was transferred. The 1st byte is transferred with receive operation disabled (SC2MOD.RXE = 0).

   - To communicate in UART mode
     The controller sends 86H to the target board in UART data format at the desired baud rate. If the serial operation mode is determined as UART, then the boot program checks if the SIO2 can be programmed to the baud rate at which the 1st byte was received. The time (number of instruction execution cycles) required for this operation varies with the operating frequency and baud rate used. If the desired baud rate cannot be used, the boot program aborts, disabling any subsequent communication. In this case, a reset is required. If necessary, the controller should set a time limit for transferring the 2nd byte as appropriate to the baud rate used.

   - To communicate in I/O Interface mode
     The controller sends the 1st byte to the target board to generate a waveform that allows the serial operation mode to be determined as I/O Interface mode. (For details, see subsection 5) Determination of a serial operation mode.) When the boot program determines that communications can be performed in I/O Interface mode at the desired baud rate, the handshake pin is driven high. If the high level is not detected on the handshake pin within the expected time period, this indicates that the boot program has aborted after determining that the SIO2 is not programmable to the desired baud rate. In this case, a reset is required. The command sequence must be started again from the 1st byte at another baud rate.
     In I/O Interface mode, the TX19A core processor sees the serial receive pin as if it were a general-purpose input port and monitors its logic transitions. If the baud rate of incoming data is high or the chip's operating frequency is high, the TX19A core processor may not be able to keep up with the speed of logic transitions. To prevent such situations, the 1st and 2nd bytes must be transferred at 1/16 of the desired baud rate; then the boot program calculates 16 times that as the desired baud rate. At this time, the transmission of one byte should be completed before an overflow occurs in the TMRB0 (see Figure 17.3.6).
     When the serial operation mode is determined as I/O Interface mode, the SIO2 is configured for SCLK Input mode. Beginning with the 3rd byte, the controller must ensure that the AC timing restrictions are satisfied at the selected baud rate. In I/O Interface mode, the boot program does not check the receive error flag; thus there is no such thing as error acknowledge (bit 3) (18H).

2. The 2nd byte, transmitted from the target board to the controller, is an acknowledge response to the 1st byte. The boot program echoes back the first byte: 86H for UART mode and 30H for I/O Interface mode.

   - UART mode
     If the SIO2 can be programmed to the baud rate at which the 1st byte was

Not Recommended for New Design

transferred, the boot program programs the BR2CR and BR2ADD registers and sends back 86H to the controller as an acknowledge. If the SIO2 is not programmable at that baud rate, the boot program simply aborts with no error indication. Following the 1st byte, the controller should allow for a time-out period. If it does not receive 86H within the allotted time-out period, the controller should give up the communication. The boot program sets the RXE bit in the SC2MOD to 1 to enable reception before loading the transmit buffer with 86H.

- I/O Interface mode
  The boot program programs the SC2MOD and SC2CR registers to configure the SIO2 in I/O Interface mode and writes 30H to the SC2BUF. Then, the SIO2 waits for the SCLK2 signal to come from the controller. Following the transmission of the 1st byte, the controller must wait for the rising edge of the handshaking pin before sending the SCLK clock. This must be done at 1/16 of the desired baud rate. If the controller receives 30H as the 2nd byte, this should be taken as the go-ahead. The controller must then deliver the 3rd byte to the target board at a rate equal to the desired baud rate. The boot program sets the SC2MOD.RXE bit to 1 before 30H is written to the transmit buffer. In I/O Interface mode, receive errors are not checked.

3. The 3rd byte, which the target board receives from the controller, is a command. The code for the RAM Transfer command is 10H.

4. The 4th byte, transmitted from the target board to the controller, is an acknowledge response to the 3rd byte. Before sending back the acknowledge response, the boot program checks for a receive error. If there was a receive error, the boot program transmits x8H and returns to the state in which it waits for a command again. In this case, the upper four bits of the acknowledge response are undefined—they hold the same values as the upper four bits of the preceding command.

   If the 3rd byte is equal to any of the command codes listed in Table 17.3.2, the boot program echoes it back to the controller. When the RAM Transfer command was received, the boot program echoes back a value of 10H and then branches to the RAM Transfer routine. Once this branch is taken, a password check is done. Password checking is detailed later in this subsection. If the 3rd byte is not a valid command, the boot program sends back 11H to the controller to indicate a command error and returns to the state in which it waits for a command. In this case, the upper four bits of the response are undefined—they hold the same values as the upper four bits of the preceding command.

5. The 5th to 16th bytes, which the target board receives from the controller, are a 12-byte password. The 5th byte is compared to the contents of address 0x0000_0474 in the flash memory; the 6th byte is compared to the contents of address 0x0000_0475 in the flash memory; likewise, the 16th byte is compared to the contents of address 0x0000_047F in the flash memory.

6. The 17th byte is a checksum value for the password sequence (5th to 16th bytes). To calculate the checksum value for the 12-byte password, add the 12 bytes together, drop the carries and take the two's complement of the total sum. Transmit this

Not Recommended for New Design

checksum value from the controller to the target board. The checksum calculation is described in detail later in this section.

7. The 18th byte, transmitted from the target board to the controller, is an acknowledge response to the 5th to 17th bytes. First, the RAM Transfer routine checks for a receive error in the 5th to 17th bytes. If there was a receive error, the boot program sends back 18H and returns to the state in which it waits for a command (i.e., 3rd byte) again. In this case, the upper four bits of the acknowledge response are the same as those of the preceding command (i.e., all 1s).

   Next, the RAM Transfer routine performs the checksum operation to ensure data integrity. Adding the 5th to 17th bytes together must result in zero (with the carries dropped). If it is not zero, the RAM Transfer routine sends back 11H to the controller to indicate a checksum error and returns to the state in which it waits for a command (i.e., the 3rd byte) again.

   Finally, the RAM Transfer routine examines the result of the password check. If a password error is determined, the RAM Transfer routine sends back 11H to the controller and returns to the state in which it waits for a command (i.e., the 3rd byte) again.

   For data sequences that can be used as a password, see subsection 6) Password.

   When all the above checks have been successful, the RAM transfer routine returns a normal acknowledge response (10H) to the controller.

8. The 19th to 22nd bytes, which the target board receives from the controller, indicate the start address of the RAM region where subsequent data should be stored. The 19th byte corresponds to bits 31 to 24 of the address, and the 22nd byte corresponds to bits 7 to 0 of the address.

9. The 23rd and 24th bytes, which the target board receives from the controller, indicate the number of bytes that will be transferred from the controller to be stored in the RAM. The 23rd byte corresponds to bits 15 to 8 of the transfer byte count, and the 24th byte corresponds to bits 7 to 0 of the transfer byte count.

10. The 25th byte is a checksum value for the 19th to 24th bytes. To calculate the checksum value, add all these bytes together, drop the carries and take the two's complement of the total sum. Transmit this checksum value from the controller to the target board. The checksum calculation is described later in this section.

11. The 26th byte, transmitted from the target board from the controller, is an acknowledge response to the 19th to 25th bytes. First, the RAM Transfer routine checks for a receive error in the 19th to 25th bytes. If there was a receive error, the RAM Transfer routine sends back 18H and returns to the state in which it waits for a command (i.e., the 3rd byte). In this case, the upper four bits of the acknowledge response are the same as those of the preceding command (i.e., all 1s).

    Next, the RAM Transfer routine performs the checksum operation to ensure data integrity. Adding the 19th to 25th bytes together must result in zero (with the

Not Recommended for New Design

carries dropped). If it is not zero, the RAM transfer routine sends back 11H to the controller to indicate a checksum error and returns to the state in which it waits for a command (i.e., the 3rd byte).

- The RAM storage addresses must be within the range of 0xFFFF_9800 to 0xFFFF_AFFF. Although the boot program does not perform address checks, RAM transfer may not be performed properly if other RAM address locations are specified as they are used in the program.

When the above checks have been successful, the RAM Transfer routine returns a normal acknowledge response (10H) to the controller.

12. The 27th to m'th bytes from the controller are stored in the on-chip RAM of the TMP19A71. Storage begins at the address specified by the 19th to 22nd bytes and continues for the number of bytes specified by the 23rd and 24th bytes.

13. The (m + 1)th byte is a checksum value. To calculate the checksum value, add the 27th to m'th bytes together, drop the carries and take the two's complement of the total sum. Transmit this checksum value from the controller to the target board. The checksum calculation is described in detail later in this section.

14. The (m + 2)th byte is an acknowledge response to the 27th to (m+1)th bytes. First, the RAM Transfer routine checks for a receive error in the 27th to (m+1)th bytes. If there was a receive error, the RAM Transfer routine sends back 18H and returns to the state in which it waits for a command (i.e., the 3rd byte) again. In this case, the upper four bits of the acknowledge response are the same as those of the preceding command (i.e., all 1s).
    Next, the RAM Transfer routine performs the checksum operation to ensure data integrity. Adding the 27th to (m+1)th bytes together must result in zero (with carries dropped). If it is not zero, the RAM Transfer routine sends back 11H to the controller to indicate a checksum error and returns to the state in which it waits for a command (i.e., the 3rd byte) again. If all the above checks are successful, the RAM Transfer routine returns a normal acknowledge response (10H) to the controller.

15. If the (m + 2)th byte was a normal acknowledge response, a branch is made to the address specified by the 19th to 22nd bytes in 32-bit ISA mode.

Not Recommended for New Design

2) Show Flash Memory Sum command (see Table 17.3.4)

1. The processing of the 1st and 2nd bytes are the same as for the RAM Transfer command.

2. The 3rd byte, which the target board receives from the controller, is a command. The code for the Show Flash Memory Sum command is 20H.

3. The 4th byte, transmitted from the target board to the controller, is an acknowledge response to the 3rd byte. The processing of the 4th byte is the same as for the RAM Transfer command except that the command code is 20H.

4. The Show Flash Memory Sum routine adds all the bytes of the flash memory together. The 5th to 6th bytes, transmitted from the target board to the controller, indicate the upper and lower bytes of this total sum, respectively. The sum calculation is described in detail later in this section.

5. The 7th byte is a checksum value for the 5th and 6th bytes. To calculate the checksum value, add the 5th and 6th bytes together, drop the carry and take the two's complement of the sum. Transmit this checksum value from the controller to the target board.

6. The 8th byte is the next command code.

Not Recommended for New Design

3) Show Product Information Command (see Table 17.3.5)

1. The processing of the 1st and 2nd bytes are the same as for the RAM Transfer command.

2. The 3rd byte, which the target board receives from the controller, is a command. The code for the Show Product Information command is 30H.

3. The 4th byte, transmitted from the target board to the controller, is an acknowledge response to the 3rd byte. The processing of the 4th byte is the same as for the RAM Transfer command except that the command data is 30H.

4. The 5th to 8th bytes, transmitted from the target board to the controller, are the data read from addresses 0x0000_0470 to 0x0000_0473 in the flash memory. Software revision management is possible by storing software information such as an ID in these locations.

5. The 9th to 20th bytes, transmitted from the target board to the controller, indicate the product name, which is 'TX19A71FY' in ASCII code followed by 20H, 20H, and 20H (12 bytes).

6. The 21st to 24th bytes, transmitted from the target board to the controller, indicate the start address of the flash memory area containing the password (74H, 04H, 00H , 00H).

7. The 25th to 28th bytes, transmitted from the target board to the controller, indicate the start address of the on-chip RAM (00H, 98H, FFH, FFH).

8. The 29th to 32nd bytes, transmitted from the target board to the controller, are dummy data (FFH, A7H, FFH, FFH).

9. The 33rd to 36th bytes, transmitted from the target board to the controller, indicate the end address of the on-chip RAM (FFH, FFH, BFH, FFH).

10. The 37th to 40th bytes, transmitted from the target board to the controller, are 00H, A8, FFH and FFH.
    The 41st to 44th bytes, transmitted from the target board to the controller, are FFH, AFH, FFH and FFH.

11. The 45th and 46th bytes, transmitted from the target board to the controller, indicate whether the security and protect bits are available and whether the flash memory is divided into blocks. Bit 0 indicates the presence or absence of the security bit; it is 0 if the security bit is available. Bit 1 indicates the presence or absence of the protect bits; it is 0 if the protect bits are available. If bit 2 is 0, it indicates that the flash memory is divided into blocks. The remaining bits are undefined. The 45th and 46th bytes are 00H, 00H.

12. The 47th to 50th bytes, transmitted from the target board to the controller, indicate the start address of the on-chip flash memory (00H, 00H, 00H, 00H).

Not Recommended for New Design

13. The 51st to 54th bytes, transmitted from the target board to the controller, indicate the end address of the on-chip flash memory (FFH, FFH, 03H, 00H).

14. The 55th and 56th bytes, transmitted from the target board to the controller, indicate the number of flash blocks available (02H, 00H).

15. The 57th to 92nd bytes, transmitted from the target board to the controller, contain information about the flash blocks. Flash blocks of the same size are treated as a group. Information about the flash blocks indicate the start address of a group, the size of the blocks in that group (in half words) and the number of the blocks in that group.

    The 57th to 65th bytes are the information about 128-Kbyte blocks (Block 0 and Block 1). See Table 17.3.5 for the values of bytes transmitted.

16. The 66th byte, transmitted from the target board to the controller, is a checksum value for the 5th to 65th bytes. The checksum value is calculated by adding all these bytes together, dropping the carries and taking the two's complement of the total sum.

17. The 67th byte is the next command code.

Not Recommended for New Design

4) Acknowledge responses

The boot program returns to the controller specific codes to notify processing states. Table 17.3.6 to Table 17.3.8 show the values of possible acknowledge responses to the received data. The upper four bits of the acknowledge response are equal to those of the command being executed. Bit 3 of the code indicates a receive error. Bit 0 indicates an invalid command error, a checksum error or a password error. Bit 1 and bit 2 are always 0. Receive error checking is not performed in I/O Interface mode.

Table 17.3.6 ACK Response to the Serial Operation Mode Bytes

Return Value	Meaning
0x86	The SIO can be configured to operate in UART mode. (Note)
0x30	The SIO can be configured to operate in I/O Interface mode.

Note: If the serial operation mode is determined as UART, the boot program checks if the SIO can be programmed to the baud rate at which the operation mode byte was transferred. If the baud rate is not possible, the boot program aborts without sendig back any response.

Table 17.3.7 ACK Response to the Command Byte

Return Value	Meaning
0xN8 (Note)	A receive error occurred while a command code was being received.
0xN1 (Note)	An undefined command code was received. (Reception was completed normally.)
0x10	The RAM Transfer command was received.
0x20	The Show Flash Memory Sum command was received.
0x30	The Show Product Information command was received.

Note: The four high-order bits of the ACK response are the same as those of the preceding command code.

Table 17.3.8 ACK Response to the Checksum Byte

Return Value	Meaning
0xN8 (Note)	A receive error occurred.
0xN1 (Note)	A checksum or password error occurred.
0xN0 (Note)	The checksum was correct.

Note: The four high-order bits of the ACK response are the four high-order bits of the preceding command code. They are 1 if a password error occurred.

Not Recommended for New Design

5) Determination of a serial operation mode

The first byte from the controller determines the serial operation mode. To use UART mode for communications between the controller and the target board, the controller must first send a value of 86H at a desired baud rate to the target board. To use I/O Interface mode, the controller must send the first byte with a waveform satisfying tAB > tCD (30H is sent here) at 1/16 of the desired baud rate. Figure 17.3.4 shows the waveforms for the first byte.

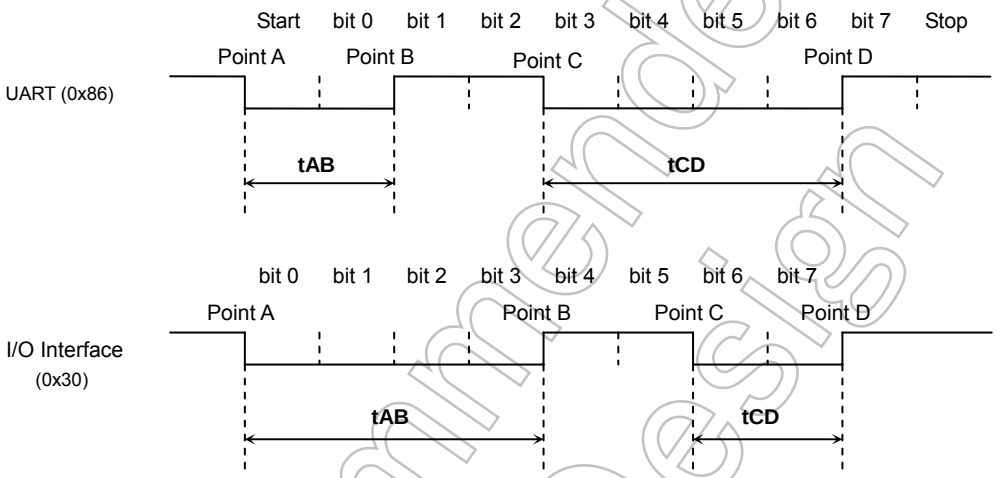


Figure 17.3.4 Serial Operation Mode Byte

After the reset state is released, the boot program monitors the first serial byte from the controller with the SIO reception disabled, and calculates the intervals of tAB, tAC and tAD shown in Figure 17.3.4. Figure 17.3.5 shows a flowchart describing the steps to determine the intervals of tAB, tAC and tAD. As shown in the flowchart, the boot program captures timer counts each time a logic transition occurs in the first serial byte. Consequently, the calculated tAB, tAC and tAD intervals are bound to have slight errors. If the transfer goes at a high baud rate, the TX19A core processor might not be able to keep up with the speed of logic transitions at the serial receive pin. In particular, I/O Interface mode is more prone to this problem since its baud rate is generally much higher than that for UART mode. To avoid such a situation, the controller should send the first serial byte at 1/16 of the desired baud rate.

The flowchart in Figure 17.3.6 shows how the boot program distinguishes between UART and I/O Interface modes. If the length of tAB is equal to or less than the length of tCD, the serial operation mode is determined as UART mode. If the length of tAB is greater than tCD, the serial operation mode is determined as I/O Interface mode. Bear in mind that if the baud rate is too high or the timer operating frequency is too low, the timer resolution will be coarse, relative to the intervals between logic transitions. This becomes a problem due to inherent errors caused by the way in which timer counts are captured by software; consequently the boot program might not be able to determine the serial operation mode correctly. (Serial operation mode settings must be made again in the programming routine.)

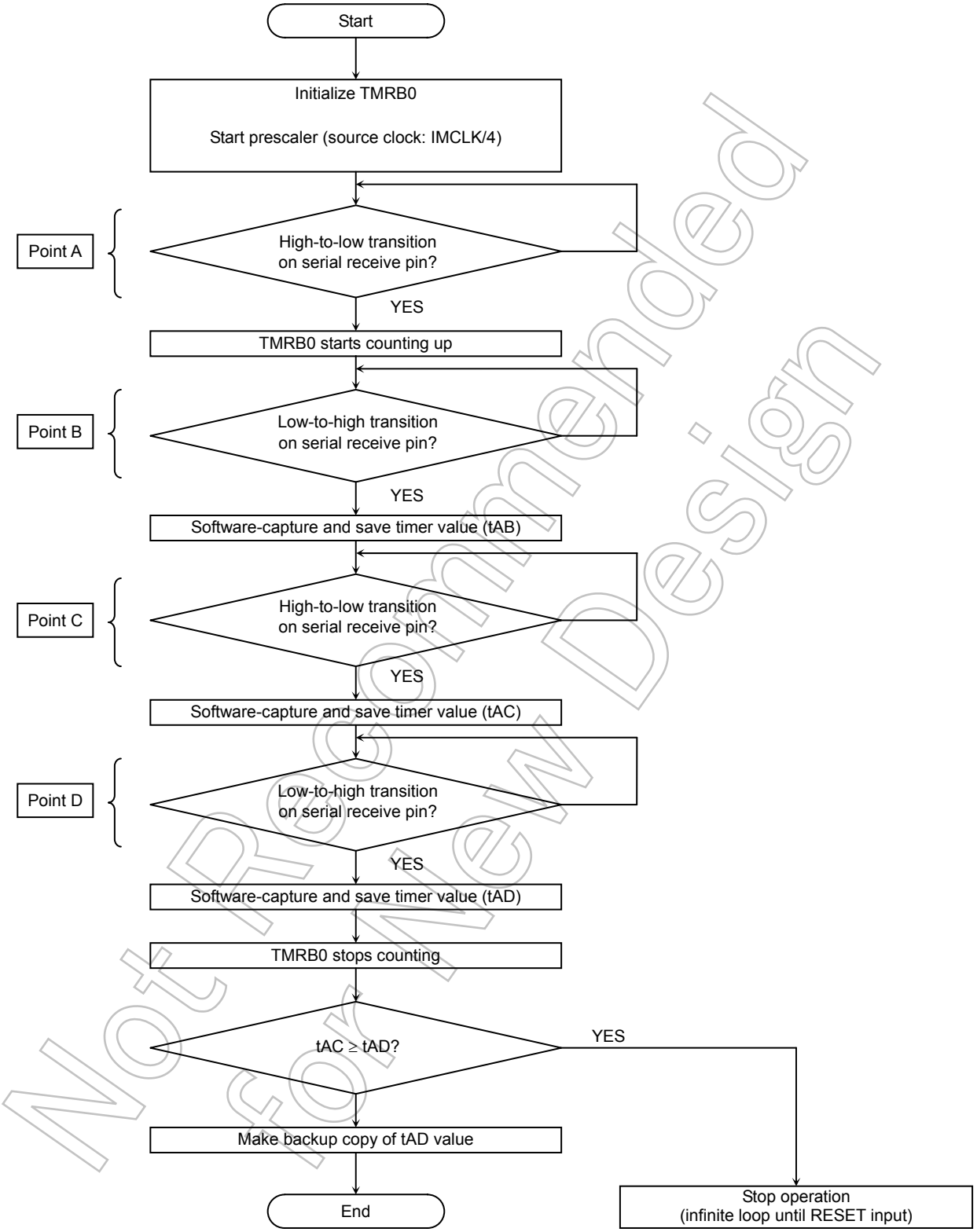
For example, the serial operation mode may be determined to be I/O Interface mode when the intended mode is UART mode. To avoid such a situation, when UART mode is utilized, the controller should allow for a time-out period within which it expects to receive an echo-back (86H) from the target board. The controller should give up the communication if it fails to get that echo-back within the allotted time. When I/O Interface mode is utilized, once the first

Not Recommended for New Design

serial byte has been transmitted, the controller should send the SCLK clock after a certain idle time to get an acknowledge response. If the received acknowledge response is not 30H, the controller should give up further communications.

To select I/O Interface mode, the data in the 1st byte need not be 30H so long as tAB > tCD is satisfied. The 1st byte may be 91H, A1H, or B1H; these values generate a falling edge at point A and point C and a rising edge at point B and point D. When tAB > tCD is satisfied and I/O Interface mode is determined, 30H is transmitted as the 2nd byte (even if the first byte is not 30H). (Here it is assumed that 30H is transmitted as the 1st byte to select I/O Interface mode.)

Not Recommended for New Design



Not Recommended for New Design

Figure 17.3.5 Serial Operation Mode Byte Reception Flow

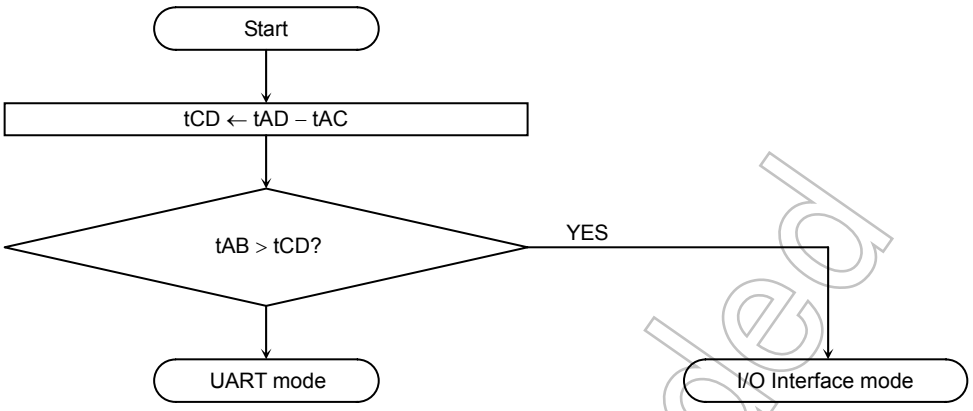


Figure 17.3.6 Serial Operation Mode Determination Flow

6) Password

The RAM Transfer command (10H) causes the boot program to perform a password check. Following an echo-back of the command, the boot program checks the contents of the 12-byte password area (0x0000_0474 to 0x0000_047F) within the flash memory. As shown in Figure 17.3.7, if all these address locations contain the same bytes of data other than FFH, a password area error occurs. In this case, the boot program returns an error acknowledge (11H) in response to the checksum byte (the 17th byte) regardless the result of password check. The only exception is when the password area is all 00H and the data at the first flash memory address (0x0000_0000 in Single Boot mode) is 0x0000_0000. In this case, 12 bytes of 00H in the password area do not cause a password area error.

The password sequence received from the controller (5th to 16th bytes) is compared to the password stored in the flash memory. Table 17.3.9 shows how they are compared byte by byte. All of the 12 bytes must match to pass the password check. Otherwise, a password error occurs, which causes the boot program to return an error acknowledge in response to the checksum byte (the 17th byte).

Not Recommended for New Design

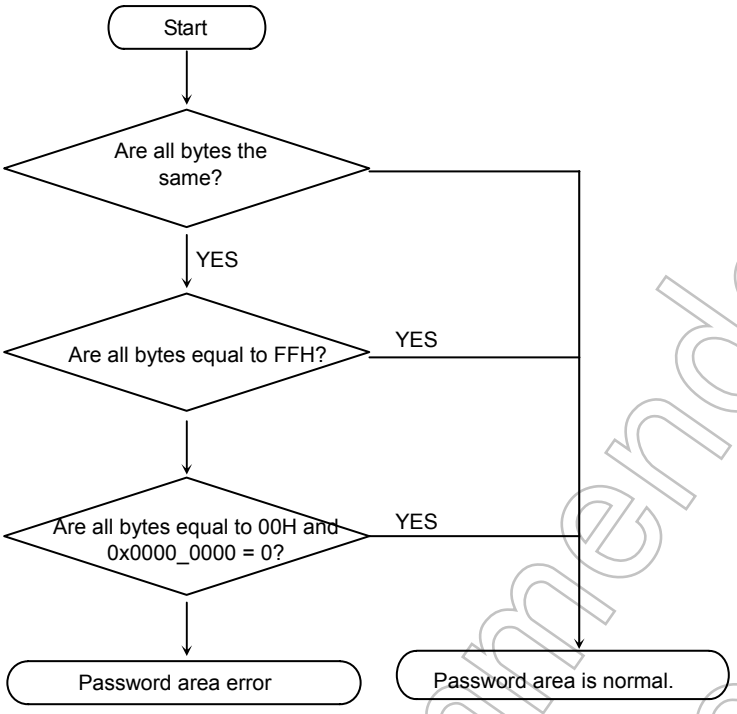


Figure 17.3.7 Password Area Check Flow

Table 17.3.9 Relationship between Received Bytes and Flash Memory Locations

Received Byte	Compared Flash Memory Data
5th byte	Address 0x0000_0474
6th byte	Address 0x0000_0475
7th byte	Address 0x0000_0476
8th byte	Address 0x0000_0477
9th byte	Address 0x0000_0478
10th byte	Address 0x0000_0479
11th byte	Address 0x0000_047A
12th byte	Address 0x0000_047B
13th byte	Address 0x0000_047C
14th byte	Address 0x0000_047D
15th byte	Address 0x0000_047E
16th byte	Address 0x0000_047F

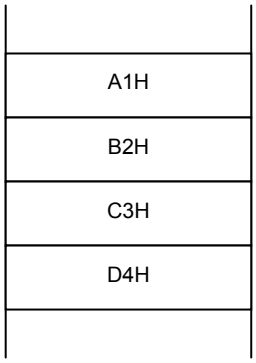
Note: We recommend setting data other than 0 at address 0x0000_0000 for security reasons.

Not Recommended for New Design

7) Calculation of the Show Flash Memory Sum command

The Show Flash Memory Sum command adds all 256 Kbytes of the flash memory together and provides the total sum as a word quantity. The sum is sent to the controller with the upper 8 bits first, followed by the lower 8 bits.

Example:



In the interests of simplicity, assume the depth of the flash memory is four locations. Then the sum of the four bytes is calculated as:

A1H + B2H + C3H + D4H = 02EAH

Hence, 02H is first sent to the controller, followed by EAH.

8) Checksum calculation

The checksum byte for a series of bytes of data is calculated by adding the bytes together, dropping the carries, and taking the two's complement of the total sum. The Show Flash Memory Sum command and the Show Product Information command perform the checksum calculation. The controller must perform the same checksum operation in transmitting checksum bytes.

Example:

Assume the Show Flash Memory Sum command provides the high- and low-order bytes of the sum as E5H and F6H. To calculate the checksum for a series of E5H and F6H:

Add the bytes together.

E5H + F6H = 1DBH

Drop the carry, and then take the two's complement of the sum. The result is the checksum byte.

0 – DBH = 25H

Not Recommended for New Design

(7) General boot program flowchart

Figure 17.3.8 shows an overall flowchart of the boot program.

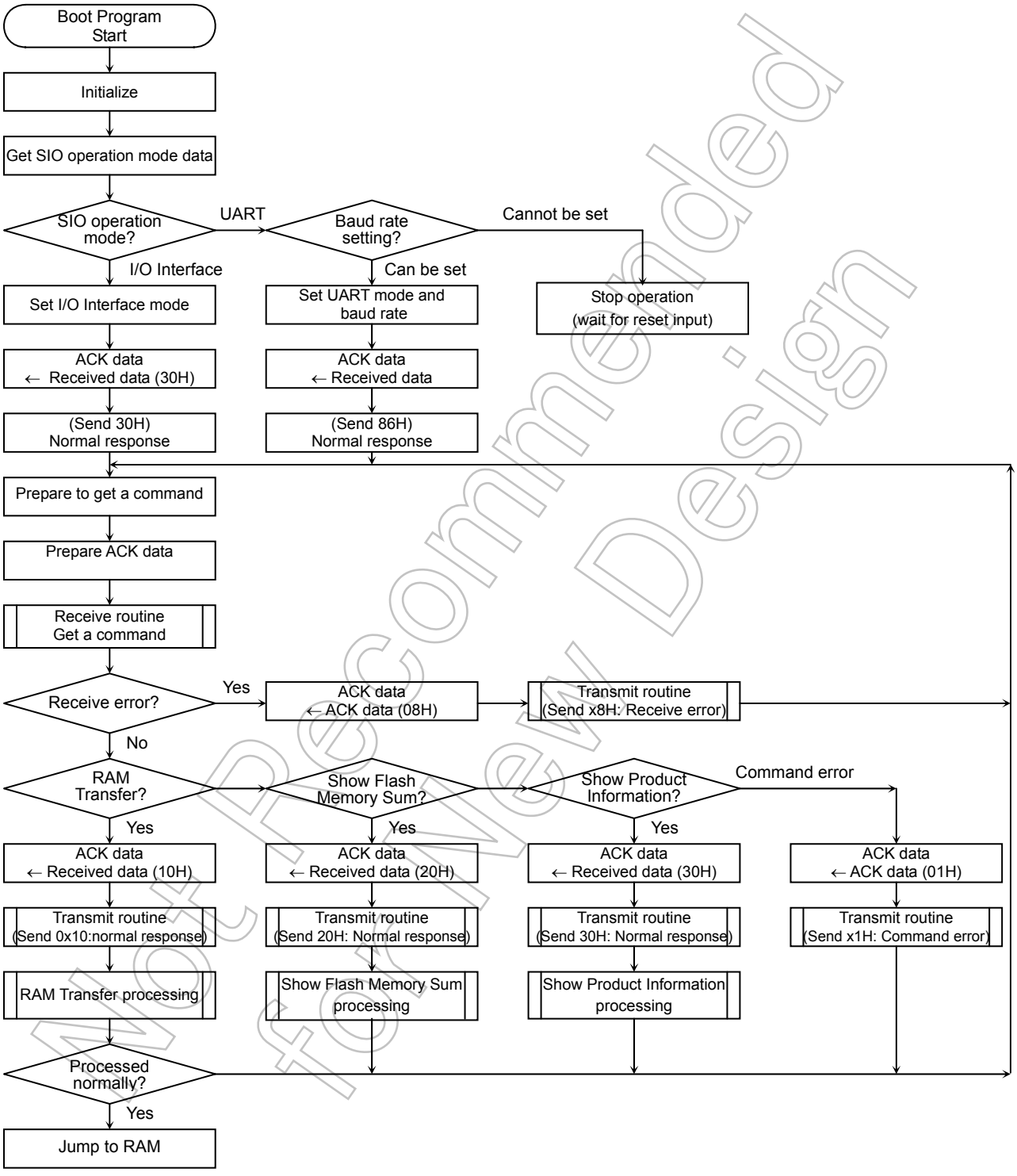


Figure 17.3.8 Overall Boot Program Flow

Not Recommended for New Design

(8) Handshake operation in I/O Interface mode

<Receive operation>

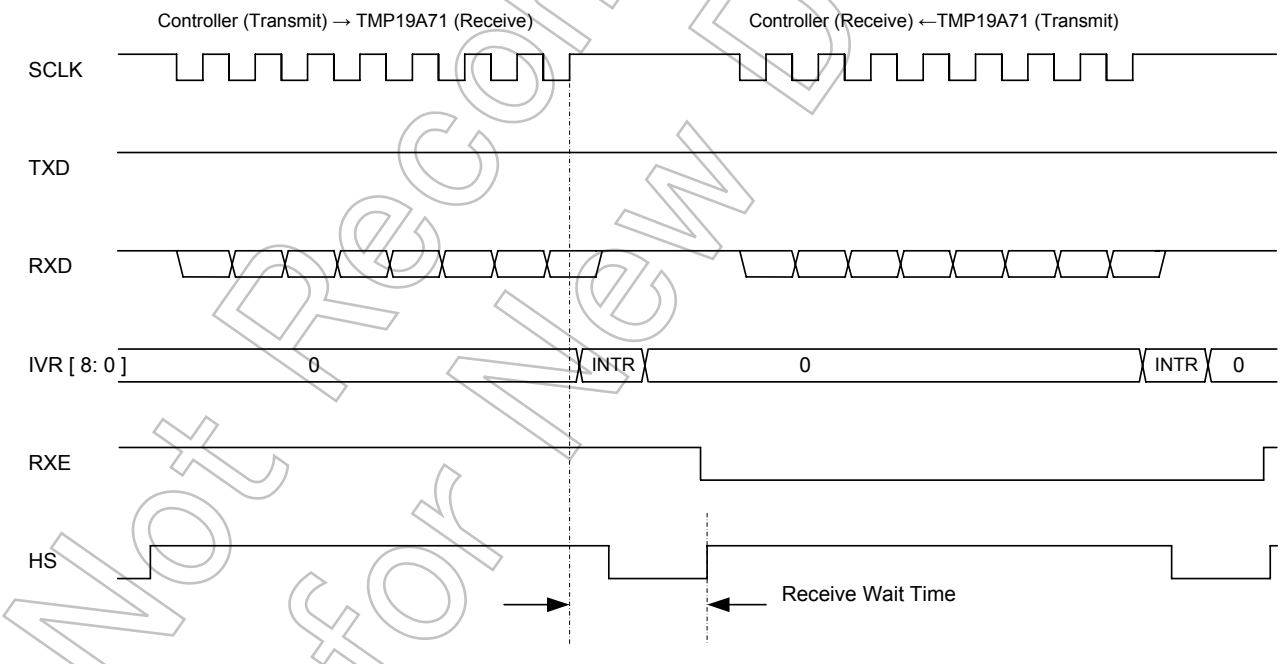
1) After setting SC2MOD0.RXE=1, the TMP19A71 drives the P80 pin high and waits for the SCLK2 signal for receiving data.

2) After receiving one byte and generating a receive-done interrupt request, the TMP19A71 performs the following sequence of operations:
   - Drives the P80 pin low to notify the controller that new data transfer cannot be performed.
   - Processes the received data (RAM storage, checksum verification, sum verification, etc.) and then clears the receive-done interrupt request.
   - Then, drives the P80 pin high to notify the controller that it is ready to receive new data and waits for the SCLK2 signal for the next receive operation.

   In Figure 17.3.9, the receive wait time is defined as the period between the rising edge of bit 7 of SCLK2 and the next rising edge of the P80 pin.

3) The next transfer operation should be initiated after a low-to-high transition is confirmed on the P80 pin. (The controller must optimize the transmit wait time for each transfer format.)

Note: The receive wait time to be inserted after each receive operation varies depending on the operating frequency and baud rate used and the processing to be performed on received data (checksum verification, RAM storage, password area check, password data check, etc.).



SCLK = SCLK2, TXD = TX2, RXD = RX2, INTT = INTRX2,
RXE = SC2MOD0.RXE, HS = P80

Figure 17.3.9 Handshake Waveform and Receive Wait Time

Not Recommended for New Design

<Transmit operation>

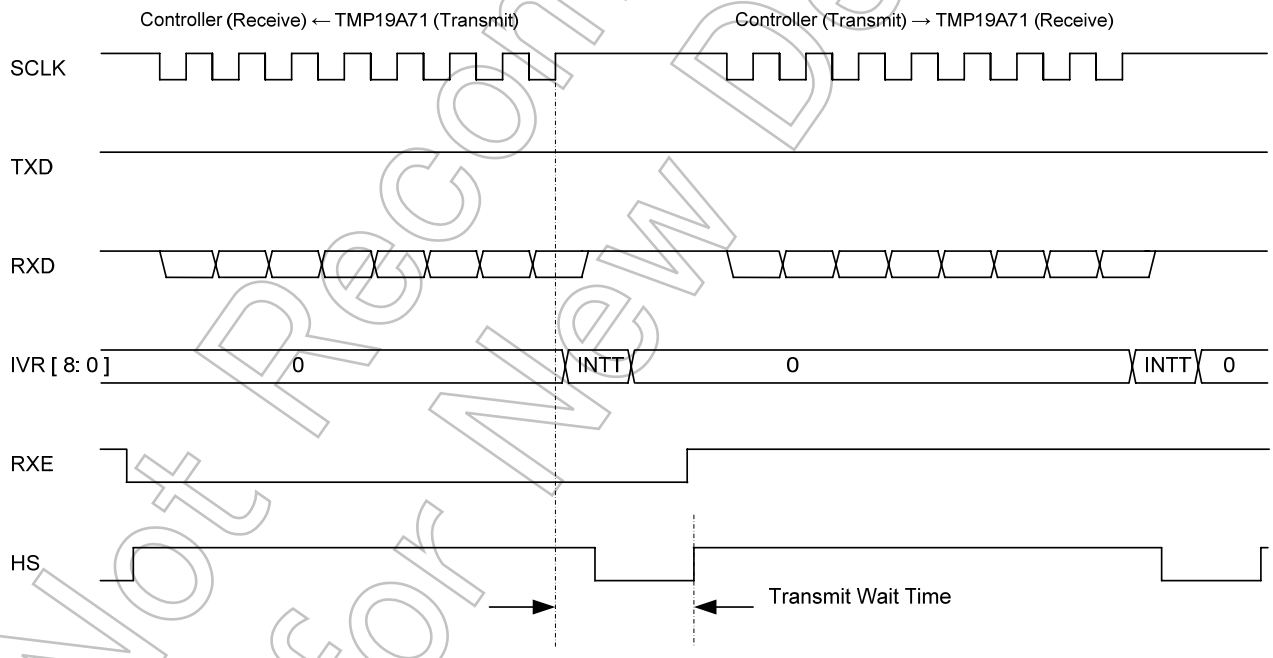
1) After setting SC2MOD0.RXE=0 and setting transmit data in the SC2BUF, the TMP19A71 drives the P80 pin high and waits for the SCLK2 signal for transmitting data.

2) After transmitting one byte and generating a transmit-done interrupt request, the TMP19A71 performs the following sequence of operations:
   - Drives the P80 pin low to notify the controller that new data transfer cannot be performed.
   - Performs required processing on the transmitted data (checksum verification, sum verification, etc.) and then clears the transmit-done interrupt request.
   - Then, sets SC2MOD0.RXE to 1 and drives the P80 pin high to notify the controller that it is ready to receive new data.

   In Figure 17.3.10, the transmit wait time is defined as the period between the rising edge of bit 7 of SCLK2 and the next rising edge of the P80 pin.

① The next transfer operation should be initiated after a low-to-high transition is confirmed on the P80 pin. (The controller must optimize the transmit wait time for each transfer format.)

Note: The transmit wait time to be inserted after each transmit operation varies depending on the operating frequency and baud rate used and the processing to be performed on transfer data (SC2MOD0.RXE setting, checksum verification, RAM storage, password verification, etc.).



SCLK = SCLK2, TXD = TX2, RXD = RX2, INTR = INTTX2, INTT = INTRX2,
RXE = SC2MOD0.RXE, HS = P80

Figure 17.3.10 Handshake Waveform and Transmit Wait Time

Not Recommended for New Design

(9)Supplementary explanation on determination of a serial operation mode
(Transmit waveforms and handshaking in I/O Interface mode)

As explained in subsection 5) Determination of a serial operation mode, each operation command must send a waveform satisfying tAB>tCD to select I/O Interface mode. At this time, point A and point C should be programmed to be falling edges and point B and point D to be rising edges. If 30H is transmitted as the 1st byte, the boot program must create at least 1 bit of high level after sending 30H so that a rising edge occurs at point D (see Figure 17.3.11 below). This measure is not needed if the data to be sent as the 1st byte includes a rising edge at point D (e.g. 91H, D9H). In either case, the serial receive pin must be driven high before entering the procedure for determining a serial operation mode. (We recommend setting the serial receive pin to high upon reset.)

In I/O Interface mode, the serial receive pin functions as a general-purpose input port for receiving the 1st byte. The timing at which I/O Interface mode is determined is after a rising edge at point D, at which point the port pin used for handshaking goes high even if this occurs before bit 7 is transmitted. (If UART mode is determined, no output is made on the handshake pin.) As shown in Figure 17.3.11 below, the position of point D varies with the data transmitted as the 1st byte. After the 1st byte has been transferred, the controller can initiate the transfer of the 2nd byte after the handshake pin goes high, which indicates that the serial operation mode has been determined.

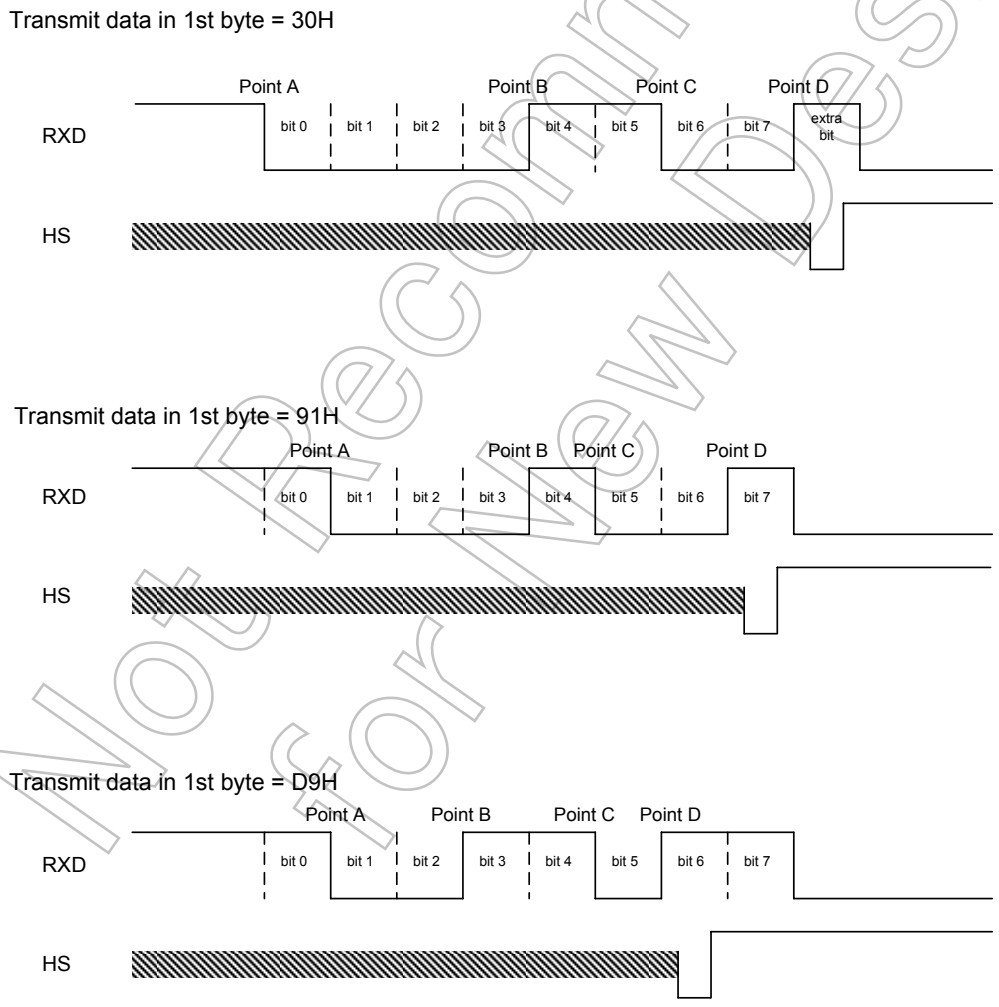


Figure 17.3.11 Supplementary Explanation on Determination of a Serial Operation Mode

Not Recommended for New Design

(10) Recommended baud rates for the boot program

Table 17.3.10 Recommended Baud Rates in Single Boot Mode (UART Mode)

		437.5kbps	345.6kbps	115.2kbps	57.6kbps	38.4kbps	19.2kbps
56MHz	PRSC	0	0	0	0	1	2
	N	2	4	7	15	11	5
	K	0	15	6	13	10	5
40MHz	PRSC	−	−	0	0	0	1
	N	−	−	9	10	16	15
	K	−	−	8	2	12	12

PRSC = BR2CR[5:4], N = BR2CR [3:0], K = BR2ADD[3:0]

Note 1: While the serial operation mode is being determined, the baud rate is determined by measuring the waveform with a timer. Capture value errors generated by the prescaler source clock and division errors that occur in the serial mode determination flow may make it impossible to set the baud rate.

Note 2: The above table shows expected combinations of frequency and baud rate settings, and other combinations may be used.

Note 3: Due to the specifications of the boot program, the fastest baud rate setting allowed is BR2CR=02H. The controller must communicate at a baud rate equal to or slower than this setting.

Table 17.3.11 Recommended Baud Rate Values in Single Boot Mode (I/O Interface Mode)

Operating Frequency (MHz)	Baud Rate in I/O Interface Mode (bps)					
56	3M	2.5M	1.25M	1M	500K	250K
40	2M	1.25M	1M	500K	250K	125K

Note: When I/O Interface mode is selected, a wait time is generated after transmitting/receiving every single byte.

Not Recommended for New Design

(11) Other considerations for using the boot program

- The on-chip boot ROM inserts one wait state in executing each instruction. In Single Boot mode, the pipeline operation for executing each instruction takes twice as long as in the case of Single-Chip mode.
- No wait states are inserted while the programming routine transferred to the on-chip RAM is executed on the on-chip RAM.
- The boot program updates the values of general-purpose registers upon reset.
- The boot program executes all instructions in 32-bit ISA mode. The instruction to be placed at the first address of the programming routine (i.e., RAM storage start address) must be a 32-bit ISA instruction.
- All special-purpose registers must be set in the programming routine.
- The boot program uses the general-purpose register r29 (shadow = 0) as the stack pointer. The r29 is set to 0xFFFF_BFF0 immediately after the RAM Transfer command is executed.
- Locations other than those to which the programming routine can be transferred with the RAM Transfer commend (0xFFFF_9800 to 0xFFFF_AFFF) are used in the boot program after reset.
- There are no limitations on RAM locations that can be used by the programming routine stored in the on-chip RAM.
- While the boot program is being executed, all maskable interrupts are disabled.
- If maskable interrupts are enabled during execution of the programming routine, the maskable interrupt vector of the boot program reads the Interrupt Vector Register (IVR) to obtain the vector address corresponding to the interrupt source, and then transfers control to this vector address. The value of the general-purpose register r4/r29 (SP) is updated by executing the routine shown on the following page. The boot program area does not provide vector addresses for interrupt sources and control for clearing interrupt requests. These should be provided in the programming routine including the setting of the IVR value.

Not Recommended for New Design

<Maskable interrupt vector routine in the boot program>

```
interrupts:
            addi    sp,  sp,  -4          ; Maskable interrupt vector start address
            sw      r4,  0( sp )
            mfc0    r4,  r13
            nop
            nop
            srl     r4,  r4,  2
            andi    r4,  r4,  0x1F
            bne     r4,  r0,  the_other   ;  If not a maskable interrupt, operation stops
            nop

            lw      r4,  IVR              ;  Read IVR
            lw      r4,  0( r4 )          ;  Read maskable interrupt vector address

            addi    sp,  sp,  4
            jr      r4                    ;  Jump to vector address
            nop

the_other:
            addi    sp,  sp,  4
the_other_lp:
            nop
            j       the_other_lp          ;  Operation stop loop
            nop
```

Note: After an interrupt is generated, one wait state is inserted for executing each instruction until control jumps to the vector address.

- Nonmaskable interrupts must not be used as they will cause an endless loop in the boot program. If an endless loop occurs, a reset must be applied.
- Once control jumps to the programming routine, it should not be returned to the boot program area except by the above maskable interrupt vector.
- Initial settings should be made before the procedure for determining a serial operation mode. After the reset state is released, an interval of approximately 200 instructions should be inserted before the transfer of the 1st byte. Note that the 1st byte is not transferred via a serial channel. To detect a falling edge on the serial receive pin, the serial receive pin must be set to high well in advance of the completion of the above wait interval. (We recommend setting the serial receive pin to high upon reset.)

Not Recommended for New Design

17.4 On-Board Programming and Erasure

The TMP19A71 flash memory is controlled by commands. In User Boot mode and Single Boot mode (the RAM Transfer command), the flash memory can be programmed and erased by the TX19A core processor executing software commands. It is the user's responsibility to create a program/erase routine. Because the flash memory cannot be read while it is being programmed or erased, the program/erase routine must be executed from the on-chip RAM.

17.4.1 Key Features

Program and erase operations on the TMP19A71 flash memory are in principle controlled by commands. This feature enables program and erase commands to be executed by accessing particular addresses in the flash memory. The TX19A core processor issues a command sequence to the flash memory by using the 32-bit SW instruction. Once a command sequence is written, the flash memory does not require the TX19A core processor to provide further controls or timings. The flash memory initiates the embedded program or erase algorithm automatically. The entire flash memory or one or two flash blocks can be erased at a time.

Table 17.4.1

Feature	Description
Auto Program	Programs and verifies the desired addresses word by word automatically.
Auto Chip Erase	Erases and verifies the entire memory array automatically.
Auto Block Erase	Erases and verifies all memory locations in the selected block automatically.
Auto Multi-Block Erase	Erases and verifies all memory locations in multiple selected blocks automatically.
Write Status Flags	Provides several status bits such as the Data Polling bit and Toggle bit, which can be used to determine whether a program or erase operation is complete or in progress.
Anti-Programmer Security Feature	Prevents intrusive access to the flash memory while in Programmer mode. Protecting both of the two flash blocks turns on the anti-programmer security feature. Unprotecting both the blocks turns off the anti-programmer security feature. This automatically causes the entire flash memory to be erased.
Block Protect	Disables program and erase operations in a flash block. Block-protecting both of the two flash blocks automatically turns on the anti-programmer security feature.

Bear in mind that, due to the on-chip TX19A core processor interface, the TMP19A71 uses addresses different from those of the standard flash command sequences. Programming is done word by word; thus the word (32 bits) load instruction should be used to write to the flash memory. Unless otherwise noted, the addresses in the flash memory are represented as virtual addresses.

Not Recommended for New Design

(1) Block architecture

0x0_0000 to 0x1_FFFF	128 Kbytes
0x2_0000 to 0x3_FFFF	128 Kbytes

Address bits [31:18] vary with the operation mode.

Figure 17.4.1 Flash Memory Block Architecture

(2) Interface between the TX19A core processor and the flash memory

Figure 17.4.2 illustrates the internal interface between the TX19A core processor and the flash memory in on-board programming modes. The diagram does not show the actual logic network; instead it is only a conceptual depiction of the interface between the TX19A core processor and the flash memory.

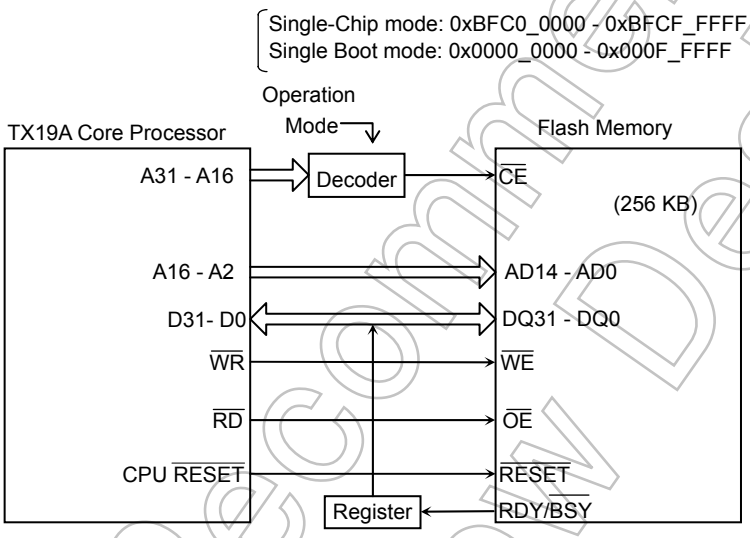


Figure 17.4.2 Internal Interface between TX19A Core Processor and Flash Memory

Not Recommended for New Design

(3) Basic operations

The TMP19A71 flash memory has the following two modes of operation:

- Read mode in which array data is read
- Embedded Operation mode in which the flash array is programmed or erased.

The flash memory enters Embedded Operation mode when a valid command sequence is executed in Read mode. In Embedded Operation mode, array data cannot be read.

1) Reading Array Data

To read array data, the flash memory must be set to Read mode. The flash memory is automatically set to Read mode upon device power-up, upon reset of the TX19A core processor, and after an embedded operation is successfully completed. If an embedded operation terminated abnormally or the flash memory is required to return to Read mode, software reset or hardware reset is used.

2) Writing Commands

The TMP19A71 flash memory is controlled by commands. A write to the command sequencer is effected by issuing a command sequence to the flash memory. The flash memory latches the provided address and data in the command sequencer and executes the required instructions (see Table 17.4.4 and Table 17.4.5).

The command sequence being written can be canceled by issuing the Read/Reset command or the Reset command (software reset). The Reset command clears the command sequencer and resets the flash memory to Read mode. Invalid command sequences also cause the flash memory to clear the command sequencer and return to Read mode.

3) Reset

- Read/Reset command, Reset command (software reset)

  The flash memory does not return to Read mode automatically if an embedded operation terminated abnormally. In this case, the Read/Reset or Reset command must be issued to put the flash memory back in Read mode. The Read/Reset or Reset command may also be written between sequence cycles of the command being written to clear the contents of the command sequencer.

- Hardware reset

  As shown in Figure 17.4.2, the flash memory has a reset pin, which is connected to the RESET signal of the TX19A core processor. When the system drives the RESET pin low or when certain events such as a watchdog timer time-out causes a reset of the TX19A core processor, the flash memory immediately terminates any operation in progress and is reset to Read mode.

  The Read/Reset and Reset commands are also tied to the RESET pin to reset the flash memory to Read mode. The embedded operation that was interrupted should be re-initiated once the flash memory is ready to accept another command sequence because data may be corrupted.

  For a description of the hardware reset operation, see section 3.1 Reset Operation. When a valid reset is achieved, the TX19A core processor reads the Reset exception vector from the flash memory and services the Reset exception.

Not Recommended for New Design

4) Auto Program command

A bit must be programmed to change its state from 1 to 0. A bit cannot be programmed from 0 back to 1. Only an erase operation can change 0 back to 1.
In User Boot mode and the RAM Transfer command of Single Boot mode, the Auto Program command programs the desired addresses in units of 32 bits (words). The Auto Program command requires four bus cycles; the program address and data are written in the fourth cycle, upon completion of which the program operation will commence. As programming is performed on a word-by-word basis, the program address must satisfy A1=A0=0.

It is not possible to program a 32-bit word if some bits have already been written. (This also applies when 1 is written in some bits. These bits must be erased before new data can be programmed.)

The Auto Program command executes a sequence of internally timed events to program the desired bits of the addressed memory word and verify that the desired bits are sufficiently programmed. The system can determine the status of the programming operation by using write status flags (see Table 17.4.3). Any commands written during the programming operation are ignored. A hardware reset immediately terminates the programming operation. The programming operation that was interrupted should be reinitiated once the flash memory is ready to accept another command sequence because data may be corrupted.

The block protection feature disables programming operation in any block. If an attempt is made to program a protected block, the Auto Program command does nothing; the flash memory returns to Read mode in approximately 3 μs after the completion of the fourth bus cycle of the command sequence.

When the embedded Auto Program algorithm is complete, the flash memory returns to Read mode.

If any failure occurs during the programming operation, the flash memory remains locked in Embedded Operation mode. The system can determine this status by using write status flags. To put the flash memory back in Read mode, use a software reset to reset the flash memory or a hardware reset to reset the whole chip. In case of a programming failure, it is recommended to replace the chip or to discontinue the use of the failing flash block.

Not Recommended for New Design

5) Auto Chip Erase command

The Auto Chip Erase command requires six bus cycles. The flash area is partitioned into two blocks, Block 0 and Block 1. The chip erase operation is performed for each individual block. After completion of the sixth bus cycle, the Auto Chip Erase operation will commence immediately. The embedded Auto Erase algorithm automatically preprograms the entire memory for an all-0 data pattern prior to the erase; then it automatically erases and verifies the entire memory for an all-1 data pattern. The system can determine the status of the chip erase operation by using write status flags (see Table 17.4.3). Any commands written during the chip erase operation are ignored. A hardware reset immediately terminates the chip erase operation. The chip erase operation that was interrupted should be re-initiated once the flash memory is ready to accept another command sequence because data may be corrupted.

The block protection feature disables erase operations in any block. The Auto Chip Erase algorithm erases the unprotected blocks and ignores the protected blocks. If both blocks are protected, the Auto Chip Erase command does nothing; the flash memory returns to Read mode in approximately 100 μs after the completion of the sixth bus cycle of the command sequence. When the embedded Auto Chip Erase algorithm is complete, the flash memory returns to Read mode.

If any failure occurs during the erase operation, the flash memory remains locked in Embedded Operation mode. The system can determine this status by using write status flags. To put the flash memory back in Read mode, use a software reset or a hardware reset to reset the flash memory or the device. In case of an erase failure, it is recommended to replace the chip or discontinue the use of the failing flash block. The failing block can be identified by the Block Erase command.

6) Auto Block Erase and Auto Multi-Block Erase Commands

The Auto Block Erase command requires six bus cycles. A time-out begins from the completion of the command sequence. After a time-out, the erase operation will commence. The embedded Auto Block Erase algorithm automatically preprograms the selected block for an all-0 data pattern, and then erases and verifies that block for an all-1 data pattern.

To erase the next block, the sixth bus cycle must be repeated; the next block address and the Auto Block Erase command must be provided within the time-out period.

Any command other than Auto Block Erase during the time-out period resets the flash memory to Read mode. The block erase time-out period is 50 μs. The system may read DQ3 to determine whether the time-out period has expired. The block erase timer begins counting upon completion of the sixth bus cycle of the Auto Block Erase command sequence. The system can determine the status of the erase operation by using write status flags (see Table 17.4.3).

Any commands written during the block erase operation are ignored. A hardware reset immediately terminates the block erase operation. The block erase operation that was interrupted should be re-initiated once the flash memory is ready to accept another command sequence because data may be corrupted.

The block protection feature disables erase operations in any block. The Auto Block Erase algorithm erases the unprotected blocks and ignores the protected blocks. If all the selected blocks are protected, the Auto Block Erase algorithm does nothing; the flash memory returns to Read mode in approximately 100 μs after the final bus cycle of the command sequence.

If any failure occurs during the erase operation, the flash memory remains locked in Embedded Operation mode. The system can determine this status by using the write status flags. To put the flash memory back in Read mode, use a software or hardware reset to reset the flash memory or the whole chip. In case of an erase failure, it is recommended to replace the chip or discontinue the use of the failing flash block. If any failure occurred during the

Not Recommended for New Design

multi-block erase operation, the failing block can be identified by running Auto Block Erase on each of the blocks selected for multi-block erasure.

7) Block Protect command

The block protection feature disables both program and erase operations in any block. After completion of the seventh bus write cycle, the RDY/BSY bit in the FLCS register is cleared to 0 during the block protect operation. Once the block protect operation is complete, this bit is set again and the flash memory automatically returns to Read mode.

If any failure occurred during the Block Protect operation, the flash memory remains locked in Embedded Operation mode with FLCS.RDY/BSY = 0. To put the flash memory back in Read mode, a software or hardware reset must be executed.

Table 17.4.2 Effects of the Program and Erase Commands on the Protected Blocks

Command	Operation
Program command on a protected block	No programming operation is performed, and the flash memory automatically returns to Read mode.
Block Erase command on a protected block	No erase operation is performed, and the flash memory automatically returns to Read mode.
Chip Erase command when all the blocks are protected	No erase operation is performed, and the flash memory automatically returns to Read mode.
Chip Erase command when any blocks are protected	Only the unprotected blocks are erased. Upon completion, the flash memory automatically returns to Read mode.
Multi-Block Erase command when any blocks are protected	Only the unprotected blocks are erased. Upon completion, the flash memory automatically returns to Read mode.

Any commands written during the Block Protect algorithm are ignored. A hardware reset immediately terminates the block protect operation. The Block Protect command that was interrupted should be re-initiated once the flash memory is ready to accept another command sequence.

8) Block Unprotect command

The Block Unprotect command requires seven bus cycles. After completion of the seventh bus write cycle, the RDY/BSY bit in the FLCS register is cleared to 0 during the block unprotect operation. Once the block unprotect operation is complete, this bit is set again and the flash memory automatically returns to Read mode.

If any failure occurred during the block unprotect operation, the flash memory remains locked in Embedded Operation mode with FLCS.RDY/BSY = 0. To put the flash memory back in Read mode, a software or hardware reset must be executed.

Any commands written during the Block Unprotect algorithm are ignored. A hardware reset immeidately terminates the block unprotect operation. In this case, the block unprotect operation must be performed again by starting with protecting all the blocks again. Use the Verify Block Protect command to verify the protect status of a block.

9) Verify Block Protect command

The Verify Block Protect command is used to verify the protect status of a block. Verify Block Protect is a four-bus-cycle operation. The address of the block to be verified is given in the fourth cycle. Any address within the block range will suffice, provided A[3:0] = 0, A4 = 1 and A6 = 0. To get correct data, a 32-bit read must be performed at least twice. Use the last read as valid data. If the selected block is protected, a value of 0x0000_0001 is returned. If the selected block is not protected, a value of 0x0000_0000 is returned. Following the fourth bus cycle, an

Not Recommended for New Design

additional block address may be read.

The Verify Block Protect command does not return the flash memory to Read mode. Either the Read/Reset command or a hardware reset is required to reset the flash memory to Read mode or to write the next command.

10) ID-READ command

The ID-READ command reads 0x0098 (fixed) as Toshiba's manufacturer's code. The flash memory address to be read is specified in the fourth bus read cycle. This address must satisfy A[4:0] = 0 and A6 = 0. To get correct data, a 32-bit read must be performed at least twice. Use the last read as valid data. By specifying an address where data other than 0x0098 is stored, the ID-READ command can be used to distinguish between the flash-version device (read value: 0x0098) and the mask-version device (read value: other than 0x0098).

Not Recommended for New Design

11) Write Operation Status

As shown in Table 17.4.3, the flash memory provides flag bits to determine the status of an embedded operation: DQ7, DQ5 and DQ3. These status bits can be read during an embedded operation in the same timing as for Read mode. The flash memory automatically returns to Read mode when an embedded operation completes. The status of an embedded operation can be checked by reading the DQ7 flag. Once an embedded operation completes, the cell data can be read from the DQ7. The DQ7 must be read after checking that an embedded operation has started (FLCS.RDY/BSY=0).

During the embedded program operation, the system must provide the program address (with A[1:0] = 0) to read valid status information. During the embedded erase operation, the system must provide an address (with A[1:0] = 0) within any of the blocks selected for erasure to read valid status information.

While an embedded operation is in progress, D[31:16] are read as 0. These bits, therefore, can be used in place of the FLCS.RDY/BSY bit in a system where D[31:16] are not normally 0. Although the FLCS register is read as undefined in the mask-version device, use of D[31:16] allows the same program to be used in the flash-version and mask-version devices.

Table 17.4.3 Write Status Flags

Status		D7 (DQ7)	D5 (DQ5)	D3 (DQ3)
Embedded operation in progress	Auto Program	DQ7 inverted	0	0
	Auto Erase (during the time-out window)	0	0	0
	Auto Erase	0	0	1
Time-out in embedded operation	Auto Program	DQ7 inverted	1	1
	Auto Erase	0	1	1

Note 1: While an embedded operation is in progress, D[31:16]=0, D[15:8]=undefined, and DQ4, DQ2, DQ1, DQ0=undefined.

Note 2: While an embedded operation is in progress, DQ7 outputs the inverted value of the programmed cell data. During the auto erase operation, DQ7 outputs 0 (erased state = 1).

Not Recommended for New Design

- DQ7 (Data Polling)

  The Data Polling bit, DQ7, indicates the status of an embedded operation. Data polling is valid after the final bus write cycle of an embedded command sequence.

  When the embedded Program algorithm is in progress, an attempt to read the flash memory will produce the complement of the data written to DQ7. Upon completion of the Program algorithm, an attempt to read the flash memory will produce the true data written to DQ7.

  When the embedded Erase algorithm is in progress, an attempt to read the flash memory will produce a 0 at the DQ7 output. Upon completion of the Erase algorithm, the flash memory will produce a 1 at the DQ7 output.

  If any failure occurs during an embedded operation, DQ7 continues to output the same value. Thus, DQ7 must always be polled in conjunction with the Exceeded Timing Limits (DQ5) flag. (see Table 17.4.3).

  The flash memory disables address latching when an embedded operation is complete. Data polling must be performed with a valid programmed address or an address within any of the unprotected blocks selected for erasure.

- DQ5 (Exceeded Timing Limits)

  DQ5 produces a 0 while the program or erase operation is in progress normally. DQ5 produces a 1 to indicate that the program or erase time has exceeded the specified internal limit. This is a failure condition that indicates the program or erase cycle was not successfully completed.

  The DQ5 failure condition also appears if the system tries to program a 1 to a location that was previously programmed to a 0. Only an erase operation can change a 0 back to a 1. In this case, the embedded Program algorithm halts the operation. Once the operation has exceeded the timing limits, DQ5 will indicate a 1. Note that this is not a device failure condition since the flash memory was used incorrectly.

  Under both these conditions, the flash memory remains locked in Embedded Operation mode. A software reset is needed to return the flash memory to Read mode.

- DQ3 (Block Erase Timer)

  After the completion of the sixth bus cycle of the Auto Block Erase command sequence, the block erase time-out window of 50 μs begins. The erase operation will begin after the time-out has expired. When the time-out is complete and the erase operation has begun, DQ3 switches from 0 to 1. If DQ3 is 0, the flash memory will accept additional Auto Block Erase commands. Each time an Auto Block Erase command is written, the time-out window is reset. DQ3 produces a 1 if an embedded operation is not successfully completed.

Not Recommended for New Design

12) Flash Control/Status Register

This is a 32-bit register for monitoring the status of the flash memory.

In Programmer mode, the RDY/BSY output is provided for the host system to monitor the status of an embedded algorithm. The TX19A core processor can poll the RDY/BSY bit in the FLCS register for the same purpose. The RDY/BSY bit is cleared to 0 when the flash memory is performing an embedded operation. The RDY/BSY bit is set to 1 when an embedded operation has completed and the flash memory is ready to accept the next command. If any failure occurs during an embedded operation, this bit remains 0. A hardware reset sets this bit to 1.

The RDY/BSY bit is cleared to 0 upon completion of the final bus write cycle of an embedded operation command, with one exception. In the case of the Auto Block Erase command, this bit is cleared after the time-out has expired. Any command is ignored while the RDY/BSY bit is cleared.

FLCS (0xFFFF_E520)

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	—	—	MROM	RDY/BSY	—	—
Read/Write	W	R	R	R/W	R	R	W	R
Reset Value	0	0	0	0	0	1	0	0
Function	Must be set to 0.			Must be set to 0.	0:Flash 1:Mask	Ready/Busy 0: Busy 1: Ready	Must be set to 0.	
	15	14	13	12	11	10	9	8
Bit Symbol	—	—	—	—	—	—	—	—
Read/Write	R	R	R	R	R	R	R	R
Reset Value	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit Symbol	—	—	—	—	—	—	—	—
Read/Write	R	R	R	R	R	R	R	R
Reset Value	0	0	0	0	0	0	0	0
	31	30	29	28	27	26	25	24
Bit Symbol	—	—	—	—	—	—	—	—
Read/Write	W	R	R	R	R	R	R	R
Reset Value	—	0	0	0	0	0	0	0

Note 1: This register must be accessed as a 32-bit quantity.
Note 2: This register does not support bit manipulation instructions.
Note 3: In the mask-version device, the MROM bit is set to 1 and any other bits are read-only with the same initial values.

Not Recommended for New Design

13) Flash Security

The TMP19A71 flash memory supports not only on-board programming but also programming using a general-purpose programmer. Therefore, the TMP19A71 flash memory provides a security feature to prevent intrusive access to the flash memory while in Programmer mode. The TMP19A71 is secured when both of the two blocks are protected, and the contents of the flash memory cannot be read by a programmer.

- Turning on the anti-programmer security feature (Disabling read accesses)

  Turning on the anti-programmer security feature disables a general-purpose programmer from reading the contents of the flash memory. To turn on this feature, once programming is complete, protect both the flash blocks. If either one of the blocks is unprotected, the anti-programmer security feature is off.

  In on-board programming modes, the TX19A core processor can read the flash memory even if the anti-programmer security feature is on. When the anti-programmer security feature is on, any reads by programming equipment will always return a half-word length value of 0x0098.

- Turning off the anti-programmer security feature (Enabling read accesses)

  The anti-programmer security feature is designed to disable reads of the flash memory by programming equipment. While the TMP19A71 is soldered on a board, the TX19A core processor can always read the flash memory, regardless of whether or not the anti-programmer security feature is on. Since the flash memory is placed under control of a user's application program in on-board operating modes, it is not easy for third parties to perform intrusive access to the flash memory. Therefore, within the confines of a board, the flash memory does not need to be secured. The anti-programmer security feature can be turned off by unprotecting both of the two flash blocks.

  Prior to turning off the anti-programmer security feature, the flash array is erased unconditionally. After the flash array is erased, the protect bit of each block is erased to turn off the anti-programmer security feature. In Single-Chip mode, block protection is lifted in a user application program. Thus, the flash memory is not erased when the anti-programmer security feature is turned off.

Not Recommended for New Design

(4) Command definitions

Table 17.4.4 On-Board Programming Mode Command Definitions

Command Sequence	Cycles Required	1st Bus Cycle (Write)		2nd Bus Cycle (Write)		3rd Bus Cycle (Write)		4th Bus Cycle (Read/Write)	
		Addr.	Data	Addr.	Data	Addr.	Data	Addr.	Data
Reset	1	0xXXXX	0x0F0						
Read/Reset	3	0x5554	0x0AA	0xAAA8	0x055	0x5554	0x0F0	RA	RD
Auto Program	4	0x5554	0x0AA	0xAAA8	0x055	0x5554	0x0A0	PA	PD
Auto Chip Erase	6	0x5554	0x0AA	0xAAA8	0x055	0x5554	0x080	0x5554	0xAA
Auto Block Erase	6	0x5554	0x0AA	0xAAA8	0x055	0x5554	0x080	0x5554	0xAA
Block Protect	7	0x5554	0x0AA	0xAAA8	0x055	0x5554	0x09A	0x5554	0xAA
Block Unprotect	7	0x5554	0x0AA	0xAAA8	0x055	0x5554	0x06A	0x5554	0xAA
ID Read/ Block Protect Verify	4	0x5554	0x0AA	0xAAA8	0x055	0x5554	0x090	IA	ID

(Continued from above)

Command Sequence	Cycles Required	5th Bus Cycle (Write)		6th Bus Cycle (Write)		7th Bus Cycle (Write)	
		Addr.	Data	Addr.	Data	Addr.	Data
Reset	1						
Read/Reset	3						
Auto Program	4						
Auto Chip Erase	6	0xAAA8	0x055	0x5554	0x010		
Auto Block Erase	6	0xAAA8	0x055	BA	0x030	BA (Note 3)	0x030
Block Protect	7	0xAAA8	0x055	0x5554	0x09A	BPA	0x09A
Block Unprotect	7	0xAAA8	0x055	0x5554	0x06A	0x5554	0x06A
ID Read/Block Protect Verify	4						

Note 1: After every bus write cycle, execute the SYNC and NOP instructions in sequence.

Note 2: In each bus write cycle, bits 16 to 19 should be set to the value coresponding to the flash memory address.

Note 3: For a multi-block erase operation, add BA in the 7th and subsequent bus write cycles.

Note 4: To operate on the flash memory, the watchdog timer must be disabled.

Not Recommended for New Design

The addresses to be provided by the TX19A core processor are shown below.

Table 17.4.5 Addresses Provided by the TX19A Core Processor

Command Address	Address: A[23:0]															
Addr.	A[23:16]	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
0xXXX0	Flash memory block	X	X	X	X	X	X	X	X	X	X	X	X	0	0	0	0
0x0000		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xAAA8		1	0	1	0	1	0	1	0	1	0	1	0	1	0	0	0
0x5554		0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	0

(5) Miscellaneous

- 0x0F0, 0x0AA, 0x055, 0x0A0, 0x080, 0x09A, 0x06A, 0x090, 0x010, 0x030

  Command sequence write data D[8:0]. To write command data, use a 32-bit (word) load (SW) instruction with D[31:9] = 0.

- RA: Read Address

  Any address in the flash memory can be specified.

- RD: Read Data

  The data at the RA (Read Address) can be read by using an 8-bit (byte) 16-bit (half-word) or 32-bit (word) load instruction.

- PA: Program Address

  Any flash memory address (A[1:0]=0) to be programmed can be specified.

- PD: Program Data

  By using a 32-bit (word) SW instruction, the PA (Program Address) can be programmed to the specified data.

- IA: ID Address

  The flash memory address (A[1:0]=0) on which the ID-READ or Verify Block Protect command is to be executed.

Table 17.4.6 IA (ID Address) Table

	A17	A6	A4	A3	ID
ID-READ (manufacturer's code)	x	0	0	0	0x0098 (fixed)
Verify Block 0 Protect	0	0	1	0	0x0000_0001 (Block 0 protected)
					0x0000_0000 (Block 0 not protected)
Verify Block 1 Protect	1				0x0000_0001 (Block 1 protected)
					0x0000_0000 (Block 1 not protected)

- ID: ID Data

  The data that indicates the result of ID-READ or Verify Block Protect executed on the IA (ID Address).

Not Recommended for New Design

- BA: Block Address

  The flash memory address (A[1:0]=0) to specify the block to be erased. For example, in User Boot mode, Block 0 can be selected by executing the LW instruction on an address in the range of 0xBFC0_0000 to 0xBFC1_FFFF (0x0000_0000~0x0001_FFFF).

- BPA: Block Protect Address

  The flash memory address (A[1:0]=0) to specify the block to be protected. For example, in User Boot mode, Block 0 can be selected by executing the LW instruction on an address in the range of 0xBFC0_0000 to 0xBFC1_FFFF (0x0000_0000~0x0001_FFFF).

Table 17.4.7 BA (Block Address) and BPA (Block Protect Address) Table

	User Boot Mode	Single Boot Mode	Size	A17
Block 0	0xBFC0_0000 to 0xBFC1_FFFF (or 0x0000_0000 to 0x0001_FFFF)	0x0000_0000 to 0x0001_FFFF	128 Kbytes	0
Block 1	0xBFC2_0000 to 0xBFC3_FFFF (or 0x0002_0000 to 0x0003_FFFF)	0x0002_0000 to 0x0003_FFFF	128 Kbytes	1

(6) Programming Examples

(a) Programming example for ID-READ

```
lui     r4,0x0000       ; r4=0x0000_xxxx
addiu   r4,r4,0x5554    ; r4=0x0000_5554
lui     r5,0x0000       ; r5=0x0000_xxxx
ori     r5,r5,0xaaa8    ; r5=0x0000_aaa8
ori     r6,r0,0x00aa    ; 1st bus write cycle
sw      r6,0(r4)        ; 1st 0x0000_5554 <-- 0x00aa
sync
nop
ori     r6,r0,0x0055    ; 2nd bus write cycle
sw      r6,0(r5)        ; 2nd 0x0000_aaa8 <-- 0x0055
sync
nop
ori     r6,r0,0x0090    ; 3rd bus write cycle
sw      r6,0(r4)        ; 3rd 0x0000_5554 <-- 0x0090
sync
nop
ori     r6,r0,0x00aa    ; 4th bus read cycle
lw      r7,0(r0)        ; 4th 0x0000_0000(IA:A6=A4=A3=A1=A0 =0) --> r7(dummy)
lw      r7,0(r0)        ; 4th 0x0000_0000(IA:A6=A4=A3=A1=A0 =0) --> r7(dummy)
lw      r7,0(r0)        ; 4th 0x0000_0000(IA:A6=A4=A3=A1=A0 =0) --> r7
sync
nop
```

Not Recommended for New Design

(b) Programming example for polling the FLCS.RDY/BSY bit

```
lui     r7,hi(FLCS)       ; r7=0xFFFF_xxxx
addiu   r7,r7,lo(FLCS)    ; r7=0xFFFF_E520 (FLCS address)
rdybsy_lp:                ; RDY/BSY poiing
lw      r6,0(r7)          ; r6 <-- FLCS
andi    r6,r6,0x04        ; Mask bits other than FLCS.RDY/BSY
beq     r6,r0,rdybsy_lp   ; Loop until FLCS.RDY/BSY=1
nop
```

(c) Programming example for erasing Block 1 and polling the write status flags

```
lui     r4,0x0002
addiu   r4,r4,0x5554      ; r4=0x0002_5554
lui     r5,0x0002
ori     r5,r5,0xaaa8      ; r5=0x0002_aaa8
ori     r6,r0,0x00aa      ; 1st bus write cycle
sw      r6,0(r4)          ; 1st 0x0002_5554 <-- 0x00aa
sync
nop
ori     r6,r0,0x0055      ; 2nd bus write cycle
sw      r6,0(r5)          ; 2nd 0x0002_aaa8 <-- 0x0055
sync
nop
ori     r6,r0,0x0080      ;3rd bus write cycle
sw      r6,0(r4)          ; 3rd 0x0002_5554 <-- 0x0080
sync
nop
ori     r6,r0,0x00aa      ; 4th bus write cycle
sw      r6,0(r4)          ; 4th 0x0002_5554 <-- 0x00aa
sync
nop
ori     r6,r0,0x0055      ; 5th bus write cycle
sw      r6,0(r5)          ; 5th 0x0002_aaa8 <-- 0x0055
sync
nop
ori     r6,r0,0x0030      ; 6th bus write cycle
sw      r6,0(r5)          ; 6th 0x0002_aaa8(A17=1) <-- 0x0030
sync                      ; Start erasing Block 1
nop

dq3_lp:                   ; Start polling the write status flags
    lw      r6,0(r5)      ; Read data at 0x0002_aaa8
```

Not Recommended for New Design

```
        andi    r7,r6,0x08          ; Mask flags other than DQ3
        beq     r7,r0,dq3_lp        ; If during the time-out (DQ3=0), go to dp3_lp
        nop
dq7_lp:
        lw      r6,0(r5)            ; Read data at 0x0002_aaa8 again
        andi    r7,r6,0x80          ; Mask flags other than DQ7
        bne     r7,r0,data_chk      ; If DQ7=1, go to data_chk
        nop
        andi    r7,r6,0x20          ; Mask flags other than MDQ5
        beq     r7,r0,dq7_lp        ; If DQ5=0 (busy), go to dq7_lp
        nop
        lw      r6,0(r5)            ; Read data at 0x0002_aaa8 again
        andi    r7,r6,0x80          ; Mask flags other than DQ7
        beq     r7,r0,erase_err     ; If DQ7=0, go to error routine
        nop
data_chk:
        nor     r7,r0,r0            ; r7=0xFFFF_FFFF
        lw      r6,0(r5)            ; Read data at 0x0002_aaa8 again
        beq     r7,r6,complete      ; If erased properly, go to end routine
        nop
erase_err:
        ori     r6,r0,0x00F0        ; Software reset
        sw      r6,0(r0)            ; 1st 0x0000_0000 <-- 0x00F0
        sync                        ; Return to Read mode
        nop

complete:       (Omitted)           ; End routine
```

Note: These programming examples assume the use of a Toshiba assembler. If a third-party assembler is used, syntax errors may occur. Change the code as necessary according to the assembler to be used.

Not Recommended for New Design

(7) Embedded Algorithms

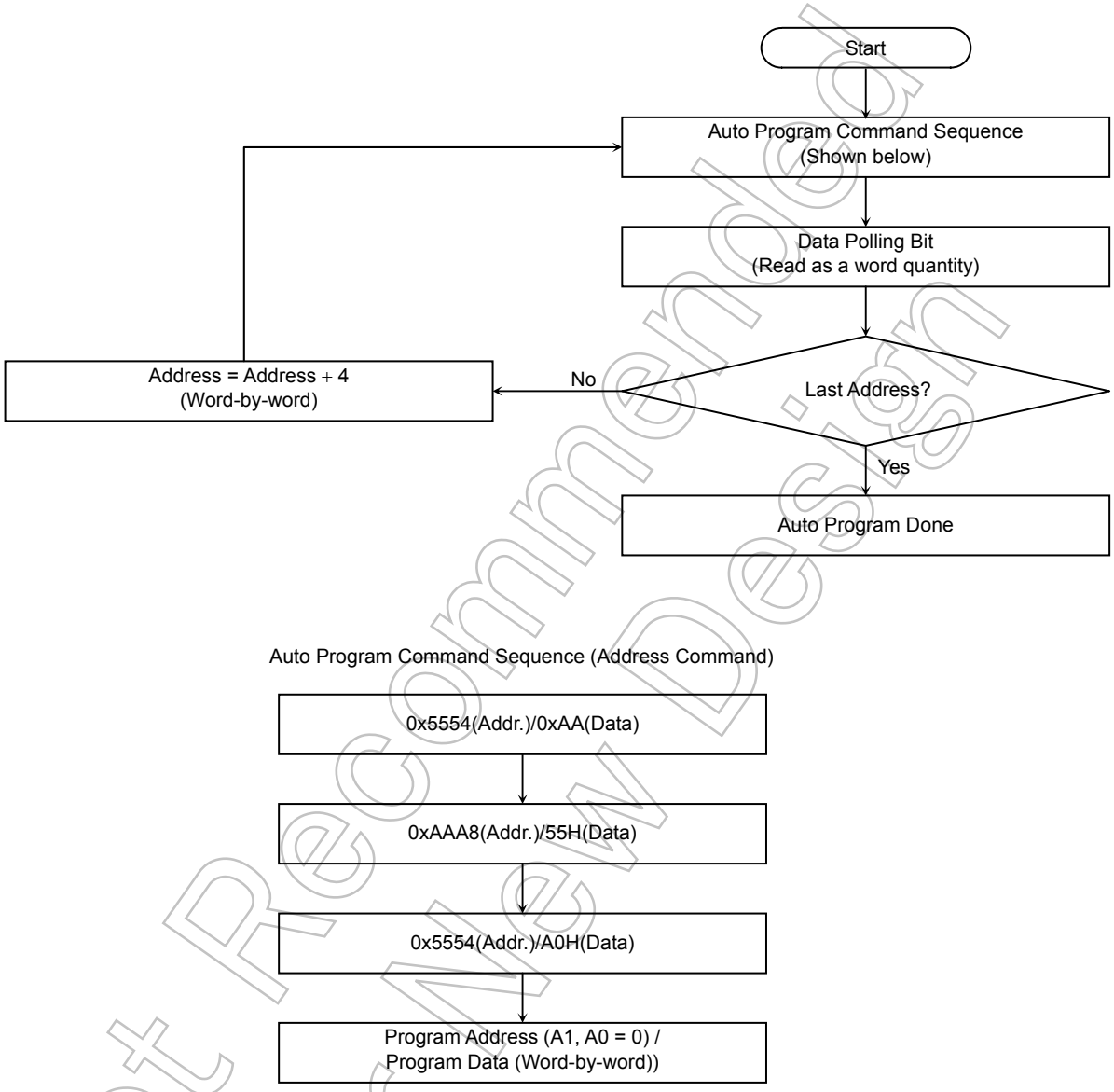
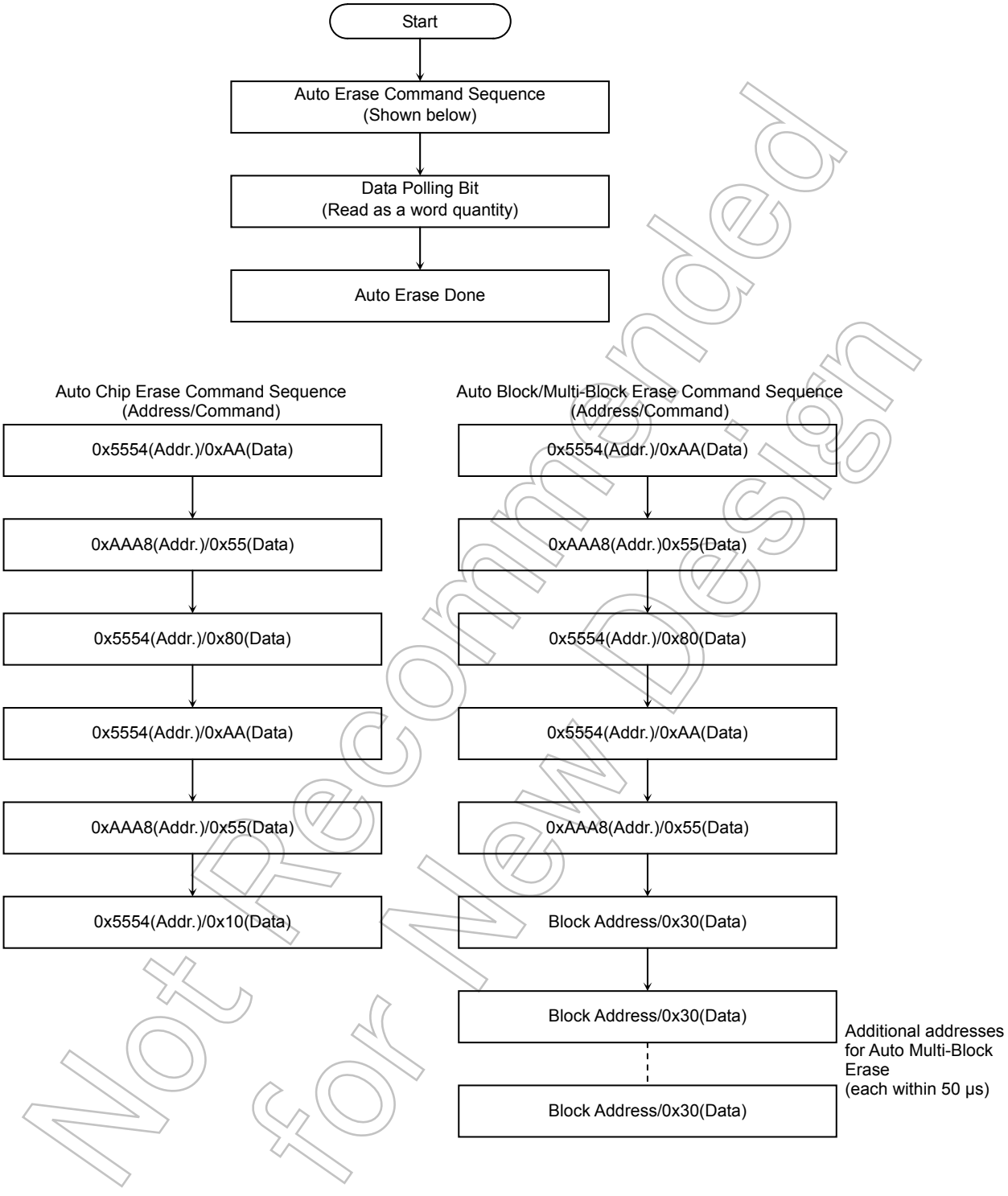


Figure 17.4.3 Auto Program Operation

Not Recommended for New Design



Not Recommended for New Design

Figure 17.4.4 Auto Erase Operations

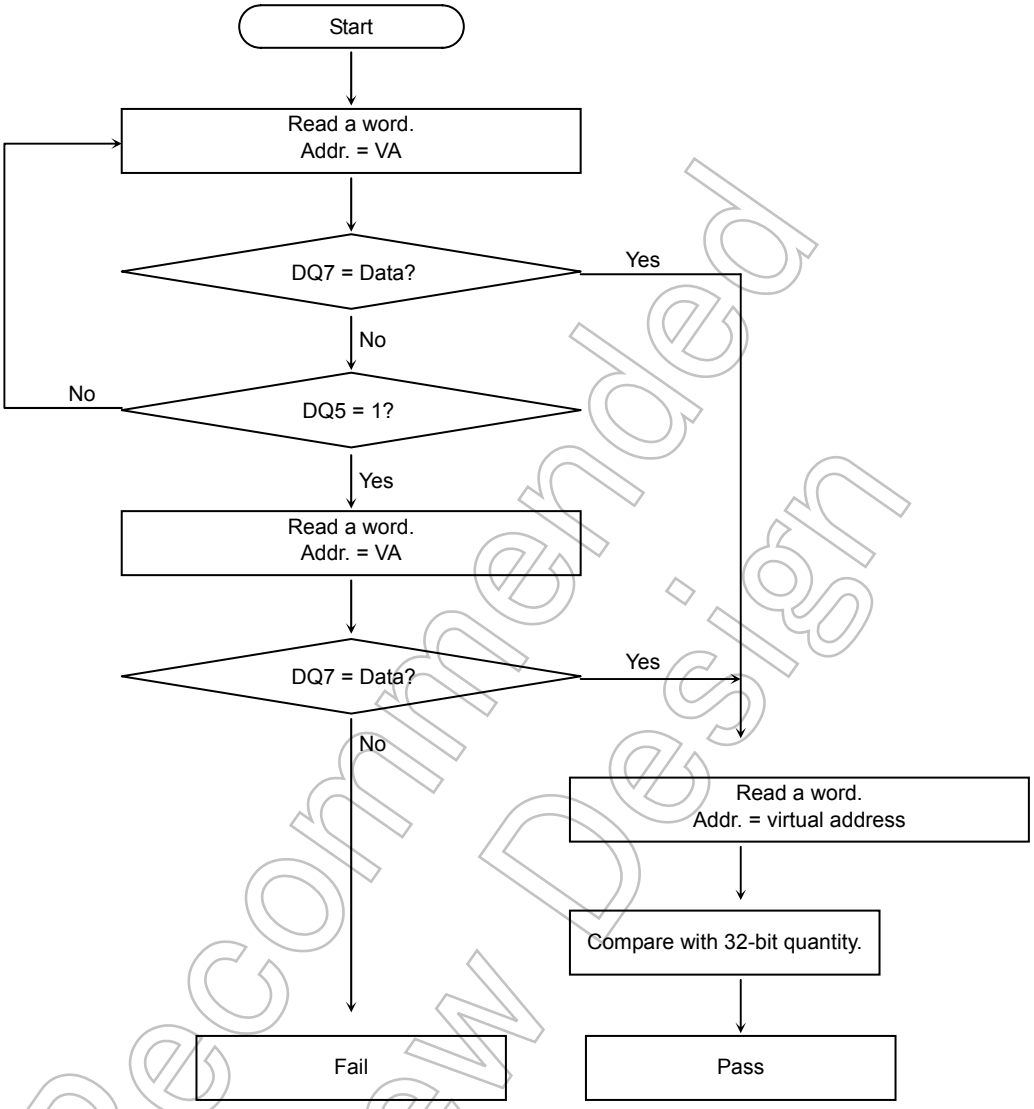


Figure 17.4.5 Data Polling (DQ7) Algorithm

Not Recommended for New Design

18. I/O Register Summary

18.1 Register Map

I/O registers occupy 16-Kbyte addresses from FFFFC000H through FFFFFFFFH.

(1) Ports
(2) Motor control circuit (PMD: Programmable Motor Driver)
(3) Encoder input circuit
(4) Serial I/O (SIO)
(5) 16-bit timer/event counter (TMRB)
(6) Watchdog timer
(7) AD converter
(8) Interrupts
(9) Clock/standby control
(10) DMA controller (DMAC)
(11) Flash memory
(12) ROM correction

Not Recommended for New Design

[1] Ports

Address	Mnemonic
FFFFC000H	P0D
1H	
2H	
3H	
4H	P0CR
5H	
6H	
7H	
8H	P0IER
9H	
AH	
BH	
CH	P0DSSR
DH	
EH	
FH	

Address	Mnemonic
FFFFC010H	Reserved
1H	
2H	
3H	
4H	P0PUCR
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC020H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC030H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC040H	P1D
1H	
2H	
3H	
4H	P1CR
5H	
6H	
7H	
8H	P1IER
9H	
AH	
BH	
CH	P1DSSR
DH	
EH	
FH	

Address	Mnemonic
FFFFC050H	Reserved
1H	
2H	
3H	
4H	P1PUCR
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC060H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC070H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC080H	P2D
1H	
2H	
3H	
4H	P2CR
5H	
6H	
7H	
8H	P2IER
9H	
AH	
BH	
CH	P2DSSR
DH	
EH	
FH	

Address	Mnemonic
FFFFC090H	Reserved
1H	
2H	
3H	
4H	P2PUCR
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC0A0H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC0B0H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Not Recommended for New Design

Address	Mnemonic
FFFFC0C0H	P3D
1H	
2H	
3H	
4H	P3CR
5H	
6H	
7H	
8H	P3IER
9H	
AH	
BH	
CH	P3DSSR
DH	
EH	
FH	

Address	Mnemonic
FFFFC0D0H	Reserved
1H	
2H	
3H	
4H	P3PUCR
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC0E0H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC0F0H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC100H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC110H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC120H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC130H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC140H	P5D
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	P5IER
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC150H	Reserved
1H	
2H	
3H	
4H	P5PUCR
5H	
6H	
7H	
8H	P5FR
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC160H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC170H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Not Recommended for New Design

Address	Mnemonic
FFFFC180H	P6D
1H	
2H	
3H	
4H	P6CR
5H	
6H	
7H	
8H	P6IER
9H	
AH	
BH	
CH	P6DSSR
DH	
EH	
FH	

Address	Mnemonic
FFFFC190H	Reserved
1H	
2H	
3H	
4H	P6PUCR
5H	
6H	
7H	
8H	P6FR
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC1A0H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC1B0H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC1C0H	P7D
1H	
2H	
3H	
4H	P7CR
5H	
6H	
7H	
8H	P7IER
9H	
AH	
BH	
CH	P7DSSR
DH	
EH	
FH	

Address	Mnemonic
FFFFC1D0H	Reserved
1H	
2H	
3H	
4H	P7PUCR
5H	
6H	
7H	
8H	P7FR1
9H	
AH	
BH	
CH	P7FR2
DH	
EH	
FH	

Address	Mnemonic
FFFFC1E0H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC1F0H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFC200H	P8D
1H	
2H	
3H	
4H	P8CR
5H	
6H	
7H	
8H	P8IER
9H	
AH	
BH	
CH	P8DSSR
DH	
EH	
FH	

Address	Mnemonic
FFFFC210H	P8ODCR
1H	
2H	
3H	
4H	P8PUCR
5H	
6H	
7H	
8H	P8FR
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC220H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC230H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Not Recommended for New Design

Address	Mnemonic
FFFFC240H	P9D
1H	
2H	
3H	
4H	P9CR
5H	
6H	
7H	
8H	P9IER
9H	
AH	
BH	
CH	P9DSSR
DH	
EH	
FH	

Address	Mnemonic
FFFFC250H	Reserved
1H	
2H	
3H	
4H	P9PUCR
5H	
6H	
7H	
8H	P9FR1
9H	
AH	
BH	
CH	P9FR2
DH	
EH	
FH	

Address	Mnemonic
FFFFC260H	P9ECR
1H	
2H	
3H	
4H	P9ECLR
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC270H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC280H	PAD
1H	
2H	
3H	
4H	PACR
5H	
6H	
7H	
8H	PAIER
9H	
AH	
BH	
CH	PADSSR
DH	
EH	
FH	

Address	Mnemonic
FFFFC290H	Reserved
1H	
2H	
3H	
4H	PAPUCR
5H	
6H	
7H	
8H	PAFR
9H	
AH	
BH	
CH	PAECR
DH	
EH	
FH	

Address	Mnemonic
FFFFC2A0H	PAECLR
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC2B0H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC2C0H	PBD
1H	
2H	
3H	
4H	PBCR
5H	
6H	
7H	
8H	PBIER
9H	
AH	
BH	
CH	PBDSSR
DH	
EH	
FH	

Address	Mnemonic
FFFFC2D0H	Reserved
1H	
2H	
3H	
4H	PBPUCR
5H	
6H	
7H	
8H	PBFR
9H	
AH	
BH	
CH	PBECR
DH	
EH	
FH	

Address	Mnemonic
FFFFC2E0H	PBECLR
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFC2F0H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Not Recommended for New Design

[2] PMD

Address	Mnemonic
FFFFC300H	MDCR0
1H	
2H	
3H	
4H	MDCNT0
5H	
6H	
7H	
8H	MDPRD0
9H	
AH	
BH	
CH	CMPU0
DH	
EH	
FH	

Address	Mnemonic
FFFFC310H	CMPV0
1H	
2H	
3H	
4H	CMPW0
5H	
6H	
7H	
8H	MDOUT0
9H	
AH	
BH	
CH	EMGREL0
DH	
EH	
FH	

Address	Mnemonic
FFFFC320H	EMGCR0
1H	
2H	
3H	
4H	TRGCR0
5H	
6H	
7H	
8H	TRGCMP00
9H	
AH	
BH	
CH	TRGCMP01
DH	
EH	
FH	

Address	Mnemonic
FFFFC330	TRGCMP02
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFC340H	MDCR1
1H	
2H	
3H	
4H	MDCNT1
5H	
6H	
7H	
8H	MDPRD1
9H	
AH	
BH	
CH	CMPU1
DH	
EH	
FH	

Address	Mnemonic
FFFFC350H	CMPV1
1H	
2H	
3H	
4H	CMPW1
5H	
6H	
7H	
8H	MDOUT1
9H	
AH	
BH	
CH	EMGREL1
DH	
EH	
FH	

Address	Mnemonic
FFFFC360H	EMGCR1
1H	
2H	
3H	
4H	TRGCR1
5H	
6H	
7H	
8H	TRGCMP10
9H	
AH	
BH	
CH	TRGCMP11
DH	
EH	
FH	

Address	Mnemonic
FFFFC370	TRGCMP12
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[3] ABZ Encoder

Address	Mnemonic
FFFFC400H	ENTNCR
1H	
2H	
3H	
4H	ENRELOAD
5H	
6H	
7H	
8H	ENINT
9H	
AH	
BH	
CH	ENCNT
DH	
EH	
FH	

Address	Mnemonic
FFFFC410H	Reserved
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFC420H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Not Recommended for New Design

[4] SIO

Address	Mnemonic
FFFFC480H	SC0MOD0
1H	SC0MOD1
2H	
3H	
4H	SC0CR
5H	SC0MOD2
6H	
7H	
8H	BR0CR
9H	BR0ADD
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFC490H	SC0BUF
1H	
2H	
3H	
4H	SC0FCNF
5H	
6H	
7H	
8H	SC0FTC
9H	SC0FRC
AH	
BH	
CH	SC0FTS
DH	SC0FRS
EH	
FH	

Address	Mnemonic
FFFFC4A0H	SC1MOD0
1H	SC1MOD1
2H	
3H	
4H	SC1CR
5H	SC1MOD2
6H	
7H	
8H	BR1CR
9H	BR1ADD
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFC4B0H	SC1BUF
1H	
2H	
3H	
4H	SC1FCNF
5H	
6H	
7H	
8H	SC1FTC
9H	SC1FRC
AH	
BH	
CH	SC1FTS
DH	SC1FRS
EH	
FH	

Address	Mnemonic
FFFFC4C0H	SC2MOD0
1H	SC2MOD1
2H	
3H	
4H	SC2CR
5H	SC2MOD2
6H	
7H	
8H	BR2CR
9H	BR2ADD
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFC4D0H	SC2BUF
1H	
2H	
3H	
4H	SC2FCNF
5H	
6H	
7H	
8H	SC2FTC
9H	SC2FRC
AH	
BH	
CH	SC2FTS
DH	SC2FRS
EH	
FH	

Address	Mnemonic
FFFFC4E0H	SC3MOD0
1H	SC3MOD1
2H	
3H	
4H	SC3CR
5H	SC3MOD2
6H	
7H	
8H	BR3CR
9H	BR3ADD
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFC4F0H	SC3BUF
1H	
2H	
3H	
4H	SC3FCNF
5H	
6H	
7H	
8H	SC3FTC
9H	SC3FRC
AH	
BH	
CH	SC3FTS
DH	SC3FRS
EH	
FH	

[5] TMRB

Address	Mnemonic
FFFFC700H	TB0RUN
1H	
2H	
3H	
4H	TB0MOD(L)
5H	(TB0MODH)
6H	
7H	
8H	TB0FF
9H	
AH	
BH	
CH	TB0REG0
DH	
EH	
FH	

Address	Mnemonic
FFFFC710H	TB0REG1
1H	
2H	
3H	
4H	TB0CP0
5H	
6H	
7H	
8H	TB0CP1
9H	
AH	
BH	
CH	TB0CNT
DH	
EH	
FH	

Address	Mnemonic
FFFFC720H	TB1RUN
1H	
2H	
3H	
4H	TB1MOD(L)
5H	(TB1MODH)
6H	
7H	
8H	TB1FF
9H	
AH	
BH	
CH	TB1REG0
DH	
EH	
FH	

Address	Mnemonic
FFFFC730H	TB1REG1
1H	
2H	
3H	
4H	TB1CP0
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	TB1CNT
DH	
EH	
FH	

Not Recommended for New Design

Address	Mnemonic
FFFFC740H	TB2RUN
1H	
2H	
3H	
4H	TB2MOD(L)
5H	(TB2MODH)
6H	
7H	
8H	TB2FF
9H	
AH	
BH	
CH	TB2REG0
DH	
EH	
FH	

Address	Mnemonic
FFFFC750H	TB2REG1
1H	
2H	
3H	
4H	TB2CP0
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	TB2CNT
DH	
EH	
FH	

Address	Mnemonic
FFFFC760H	TB3RUN
1H	
2H	
3H	
4H	TB3MOD(L)
5H	(TB3MODH)
6H	
7H	
8H	TB3FF
9H	
AH	
BH	
CH	TB3REG0
DH	
EH	
FH	

Address	Mnemonic
FFFFC770H	TB3REG1
1H	
2H	
3H	
4H	TB3CP0
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	TB3CNT
DH	
EH	
FH	

[6] WDT

Address	Mnemonic
FFFFC830H	WDMOD(L)
1H	(WDMODH)
2H	
3H	
4H	WDCR
5H	
6H	
7H	
8H	WDCNT
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[7-1] ADC (Normal Mode)

Address	Mnemonic
FFFFC900H	ADNRES0
1H	
2H	
3H	
4H	ADNRES1
5H	
6H	
7H	
8H	ADNRES2
9H	
AH	
BH	
CH	ADNRES3
DH	
EH	
FH	

Address	Mnemonic
FFFFC910H	ADNRES4
1H	
2H	
3H	
4H	ADNRES5
5H	
6H	
7H	
8H	ADNRES6
9H	
AH	
BH	
CH	ADNRES7
DH	
EH	
FH	

Address	Mnemonic
FFFFC920H	ADCHPR0
1H	
2H	
3H	
4H	ADNMOD0(L)
5H	(ADNMOD0H)
6H	
7H	
8H	ADNCLK0
9H	
AH	
BH	
CH	CMPCTL0(L)
DH	(CMPCTL0H)
EH	
FH	

Address	Mnemonic
FFFFC930H	ADCHPC0
1H	
2H	
3H	
4H	ADCMP00
5H	
6H	
7H	
8H	ADCMP01
9H	
AH	
BH	
CH	ADCBASN0
DH	
EH	
FH	

Not Recommended for New Design

Address	Mnemonic
FFFFC940H	ADCSTART0
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFC980H	ADNRES8
1H	
2H	
3H	
4H	ADNRES9
5H	
6H	
7H	
8H	ADNRES10
9H	
AH	
BH	
CH	ADNRES11
DH	
EH	
FH	

Address	Mnemonic
FFFFC990H	ADNRES12
1H	
2H	
3H	
4H	ADNRES13
5H	
6H	
7H	
8H	ADNRES14
9H	
AH	
BH	
CH	ADNRES15
DH	
EH	
FH	

Address	Mnemonic
FFFFC9A0H	ADCHPR1
1H	
2H	
3H	
4H	ADNMOD1(L)
5H	(ADNMOD1H)
6H	
7H	
8H	ADNCLK1
9H	
AH	
BH	
CH	CMPCTL1(L)
DH	(CMPCTL1H)
EH	
FH	

Address	Mnemonic
FFFFC9B0H	ADCHPC1
1H	
2H	
3H	
4H	ADCMP10
5H	
6H	
7H	
8H	ADCMP11
9H	
AH	
BH	
CH	ADCBASN1
DH	
EH	
FH	

Address	Mnemonic
FFFFC9C0H	ADCSTART1
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Not Recommended for New Design

[7-2] ADC (PMD Mode)

Address	Mnemonic
FFFFCD00H	ADPRES0
1H	
2H	
3H	
4H	ADPRES1
5H	
6H	
7H	
8H	ADPRES2
9H	
AH	
BH	
CH	ADPRES3
DH	
EH	
FH	

Address	Mnemonic
FFFFCD10H	ADPRES4
1H	
2H	
3H	
4H	ADPRES5
5H	
6H	
7H	
8H	ADPRES6
9H	
AH	
BH	
CH	ADPRES7
DH	
EH	
FH	

Address	Mnemonic
FFFFCD20H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFCD30H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFCD40H	ADCSETT00(L)
1H	(ADCSETT00H)
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	ADCSET00(L)
9H	(ADCSET00H)
AH	
BH	
CH	ADCSET01(L)
DH	(ADCSET01H)
EH	
FH	

Address	Mnemonic
FFFFCD50H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	ADPCLK0
9H	
AH	
BH	
CH	ADPMOD00
DH	
EH	
FH	

Address	Mnemonic
FFFFCD60H	ADPMOD01(L)
1H	(ADPMOD01H)
2H	
3H	
4H	ADCNE0(L)
5H	(ADCNE0H)
6H	
7H	
8H	ADCNT0
9H	
AH	
BH	
CH	ADCBASP0
DH	
EH	
FH	

Address	Mnemonic
FFFFCD70H	ADMODSEL0
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFCD80H	ADPRES8
1H	
2H	
3H	
4H	ADPRES9
5H	
6H	
7H	
8H	ADPRES10
9H	
AH	
BH	
CH	ADPRES11
DH	
EH	
FH	

Address	Mnemonic
FFFFCD90H	ADPRES12
1H	
2H	
3H	
4H	ADPRES13
5H	
6H	
7H	
8H	ADPRES14
9H	
AH	
BH	
CH	ADPRES15
DH	
EH	
FH	

Address	Mnemonic
FFFFCDA0H	ADPRES16
1H	
2H	
3H	
4H	ADPRES17
5H	
6H	
7H	
8H	ADPRES18
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFCDB0H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Not Recommended for New Design

Address	Mnemonic
FFFFCDC0H	ADCSETT10(L)
1H	(ADCSETT10H)
2H	
3H	
4H	ADCSETT11
5H	
6H	
7H	
8H	ADCSET10(L)
9H	(ADCSET10H)
AH	
BH	
CH	ADCSET11(L)
DH	(ADCSET11H)
EH	
FH	

Address	Mnemonic
FFFFCDD0H	ADCSET12(L)
1H	(ADCSET12H)
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	ADPCLK1
9H	
AH	
BH	
CH	ADPMOD10
DH	
EH	
FH	

Address	Mnemonic
FFFFCDE0H	ADPMOD11(L)
1H	(ADPMOD11H)
2H	
3H	
4H	ADCNE1(L)
5H	(ADCNE1H)
6H	
7H	
8H	ADCNT1
9H	
AH	
BH	
CH	ADCBASP1
DH	
EH	
FH	

Address	Mnemonic
FFFFCDF0H	ADMODSEL1
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[8] IRC

Address	Mnemonic
FFFFD000H	IMR00
1H	(IMR01)
2H	(IMR02)
3H	(IMR03)
4H	IMR04
5H	(IMR05)
6H	(IMR06)
7H	(IMR07)
8H	IMR08
9H	(IMR09)
AH	(IMR10)
BH	(IMR11)
CH	IMR12
DH	(IMR13)
EH	(IMR14)
FH	(IMR15)

Address	Mnemonic
FFFFD010H	IMR16
1H	(IMR17)
2H	(IMR18)
3H	(IMR19)
4H	IMR20
5H	(IMR21)
6H	(IMR22)
7H	(IMR23)
8H	IMR24
9H	(IMR25)
AH	(IMR26)
BH	(IMR27)
CH	IMR28
DH	(IMR29)
EH	(IMR30)
FH	(IMR31)

Address	Mnemonic
FFFFD020H	IMR32
1H	(IMR33)
2H	(IMR34)
3H	(IMR35)
4H	IMR36
5H	(IMR37)
6H	(IMR38)
7H	(IMR39)
8H	IMR40
9H	(IMR41)
AH	(IMR42)
BH	(IMR43)
CH	IMR44
DH	(IMR45)
EH	(IMR46)
FH	(IMR47)

Address	Mnemonic
FFFFD030H	IMR48
1H	(IMR49)
2H	(IMR50)
3H	(IMR51)
4H	IMR52
5H	(IMR53)
6H	(IMR54)
7H	(IMR55)
8H	IMR56
9H	(IMR57)
AH	(IMR58)
BH	(IMR59)
CH	IMR60
DH	(IMR61)
EH	(IMR62)
FH	(IMR63)

Address	Mnemonic
FFFFD040H	IMR64
1H	(IMR65)
2H	(IMR66)
3H	(IMR67)
4H	IMR68
5H	(IMR69)
6H	(IMR70)
7H	(IMR71)
8H	IMR72
9H	(IMR73)
AH	(IMR74)
BH	(IMR75)
CH	IMR76
DH	(IMR77)
EH	(IMR78)
FH	(IMR79)

Address	Mnemonic
FFFFD050H	IMR80
1H	(IMR81)
2H	(IMR82)
3H	(IMR83)
4H	IMR84
5H	(IMR85)
6H	(IMR86)
7H	(IMR87)
8H	IMR88
9H	(IMR89)
AH	(IMR90)
BH	(IMR91)
CH	IMR92
DH	(IMR93)
EH	(IMR94)
FH	(IMR95)

Address	Mnemonic
FFFFD080H	IVR
1H	
2H	
3H	
4H	ICLR
5H	
6H	
7H	
8H	ILEV
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFD070H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Not Recommended for New Design

[9] Clock Generator

Address	Mnemonic
FFFFD300H	CLKACT
1H	
2H	
3H	
4H	CLKOSC
5H	CLKWUT
6H	CLKSPD
7H	CLKPRSC
8H	Reserved
9H	Reserved
AH	Reserved
BH	
CH	Reserved
DH	CLKMISC
EH	
FH	

Address	Mnemonic
FFFFD310H	CLKNMI
1H	Reserved
2H	CLKW0
3H	Reserved
4H	Reserved
5H	Reserved
6H	Reserved
7H	Reserved
8H	Reserved
9H	Reserved
AH	CLKINT0
BH	CLKINT1
CH	CLKINT2
DH	CLKINT3
EH	Reserved
FH	Reserved

Address	Mnemonic
FFFFD320H	Reserved
1H	Reserved
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFD330H	Reserved
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[10] MODEC

Address	Mnemonic
FFFFD400H	MODECR
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFD410H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFD420H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFD430H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[11] DMAC

Address	Mnemonic
FFFFD600H	CCR0(LL)
1H	(CCR0LH)
2H	(CCR0HL)
3H	(CCR0HH)
4H	CSR0(LL)
5H	(CSR0LH)
6H	(CSR0HL)
7H	(CSR0HH)
8H	SAR0(LL)
9H	(SAR0LH)
AH	(SAR0HL)
BH	(SAR0HH)
CH	DAR0(LL)
DH	(DAR0LH)
EH	(DAR0HL)
FH	(DAR0HH)

Address	Mnemonic
FFFFD610H	BCR0(LL)
1H	(BCR0LH)
2H	(BCR0HL)
3H	(BCR0HH)
4H	
5H	
6H	
7H	
8H	DTCR0(LL)
9H	(DTCR0LH)
AH	(DTCR0HL)
BH	(DTCR0HH)
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFD620H	CCR1(LL)
1H	(CCR1LH)
2H	(CCR1HL)
3H	(CCR1HH)
4H	CSR1(LL)
5H	(CSR1LH)
6H	(CSR1HL)
7H	(CSR1HH)
8H	SAR1(LL)
9H	(SAR1LH)
AH	(SAR1HL)
BH	(SAR1HH)
CH	DAR1(LL)
DH	(DAR1LH)
EH	(DAR1HL)
FH	(DAR1HH)

Address	Mnemonic
FFFFD630H	BCR1(LL)
1H	(BCR1LH)
2H	(BCR1HL)
3H	(BCR1HH)
4H	
5H	
6H	
7H	
8H	DTCR1(LL)
9H	(DTCR1LH)
AH	(DTCR1HL)
BH	(DTCR1HH)
CH	
DH	
EH	
FH	

Not Recommended for New Design

Address	Mnemonic
FFFFD640H	CCR2(LL)
1H	(CCR2LH)
2H	(CCR2HL)
3H	(CCR2HH)
4H	CSR2(LL)
5H	(CSR2LH)
6H	(CSR2HL)
7H	(CSR2HH)
8H	SAR2(LL)
9H	(SAR2LH)
AH	(SAR2HL)
BH	(SAR2HH)
CH	DAR2(LL)
DH	(DAR2LH)
EH	(DAR2HL)
FH	(DAR2HH)

Address	Mnemonic
FFFFD650H	BCR2(LL)
1H	(BCR2LH)
2H	(BCR2HL)
3H	(BCR2HH)
4H	
5H	
6H	
7H	
8H	DTCR2(LL)
9H	(DTCR2LH)
AH	(DTCR2HL)
BH	(DTCR2HH)
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFD660H	CCR3(LL)
1H	(CCR3LH)
2H	(CCR3HL)
3H	(CCR3HH)
4H	CSR3(LL)
5H	(CSR3LH)
6H	(CSR3HL)
7H	(CSR3HH)
8H	SAR3(LL)
9H	(SAR3LH)
AH	(SAR3HL)
BH	(SAR3HH)
CH	DAR3(LL)
DH	(DAR3LH)
EH	(DAR3HL)
FH	(DAR3HH)

Address	Mnemonic
FFFFD670H	BCR3(LL)
1H	(BCR3LH)
2H	(BCR3HL)
3H	(BCR3HH)
4H	
5H	
6H	
7H	
8H	DTCR3(LL)
9H	(DTCR3LH)
AH	(DTCR3HL)
BH	(DTCR3HH)
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFD680H	CCR4(LL)
1H	(CCR4LH)
2H	(CCR4HL)
3H	(CCR4HH)
4H	CSR4(LL)
5H	(CSR4LH)
6H	(CSR4HL)
7H	(CSR4HH)
8H	SAR4(LL)
9H	(SAR4LH)
AH	(SAR4HL)
BH	(SAR4HH)
CH	DAR4(LL)
DH	(DAR4LH)
EH	(DAR4HL)
FH	(DAR4HH)

Address	Mnemonic
FFFFD690H	BCR4(LL)
1H	(BCR4LH)
2H	(BCR4HL)
3H	(BCR4HH)
4H	
5H	
6H	
7H	
8H	DTCR4(LL)
9H	(DTCR4LH)
AH	(DTCR4HL)
BH	(DTCR4HH)
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFD6A0H	CCR5(LL)
1H	(CCR5LH)
2H	(CCR5HL)
3H	(CCR5HH)
4H	CSR5(LL)
5H	(CSR5LH)
6H	(CSR5HL)
7H	(CSR5HH)
8H	SAR5(LL)
9H	(SAR5LH)
AH	(SAR5HL)
BH	(SAR5HH)
CH	DAR5(LL)
DH	(DAR5LH)
EH	(DAR5HL)
FH	(DAR5HH)

Address	Mnemonic
FFFFD6B0H	BCR5(LL)
1H	(BCR5LH)
2H	(BCR5HL)
3H	(BCR5HH)
4H	
5H	
6H	
7H	
8H	DTCR5(LL)
9H	(DTCR5LH)
AH	(DTCR5HL)
BH	(DTCR5HH)
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFD6C0H	CCR6(LL)
1H	(CCR6LH)
2H	(CCR6HL)
3H	(CCR6HH)
4H	CSR6(LL)
5H	(CSR6LH)
6H	(CSR6HL)
7H	(CSR6HH)
8H	SAR6(LL)
9H	(SAR6LH)
AH	(SAR6HL)
BH	(SAR6HH)
CH	DAR6(LL)
DH	(DAR6LH)
EH	(DAR6HL)
FH	(DAR6HH)

Address	Mnemonic
FFFFD6D0H	BCR6(LL)
1H	(BCR6LH)
2H	(BCR6HL)
3H	(BCR6HH)
4H	
5H	
6H	
7H	
8H	DTCR6(LL)
9H	(DTCR6LH)
AH	(DTCR6HL)
BH	(DTCR6HH)
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFD6E0H	CCR7(LL)
1H	(CCR7LH)
2H	(CCR7HL)
3H	(CCR7HH)
4H	CSR7(LL)
5H	(CSR7LH)
6H	(CSR7HL)
7H	(CSR7HH)
8H	SAR7(LL)
9H	(SAR7LH)
AH	(SAR7HL)
BH	(SAR7HH)
CH	DAR7(LL)
DH	(DAR7LH)
EH	(DAR7HL)
FH	(DAR7HH)

Address	Mnemonic
FFFFD6F0H	BCR7(LL)
1H	(BCR7LH)
2H	(BCR7HL)
3H	(BCR7HH)
4H	
5H	
6H	
7H	
8H	DTCR7(LL)
9H	(DTCR7LH)
AH	(DTCR7HL)
BH	(DTCR7HH)
CH	
DH	
EH	
FH	

Not Recommended for New Design

Address	Mnemonic
FFFFD700H	DCR(LL)
1H	(DCRLH)
2H	(DCRHL)
3H	(DCRHH)
4H	Reserved
5H	Reserved
6H	Reserved
7H	Reserved
8H	
9H	
AH	
BH	
CH	DHR(LL)
DH	(DHRLH)
EH	(DHRHL)
FH	(DHRHH)

Address	Mnemonic
FFFFD710H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFD720H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFD730H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFD740H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFD750H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFD760H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFD770H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Not Recommended for New Design

[12] Flash Memory (Flash version only; DMA not supported)

Address	Mnemonic
FFFFE510H	SEQMOD
1H	
2H	
3H	
4H	SEQCNT
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFE520H	FLCS
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFE530H	B0DCR
1H	
2H	
3H	
4H	B0DLR
5H	
6H	
7H	
8H	B1DCR
9H	
AH	
BH	
CH	B1DLR
DH	
EH	
FH	

[13] ROM Correction (DMA not supported)

Address	Mnemonic
FFFFE540H	ADDREG0
1H	
2H	
3H	
4H	ADDREG1
5H	
6H	
7H	
8H	ADDREG2
9H	
AH	
BH	
CH	ADDREG3
DH	
EH	
FH	

Address	Mnemonic
FFFFE550H	ADDREG4
1H	
2H	
3H	
4H	ADDREG5
5H	
6H	
7H	
8H	ADDREG6
9H	
AH	
BH	
CH	ADDREG7
DH	
EH	
FH	

Address	Mnemonic
FFFFE560H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Address	Mnemonic
FFFFE570H	Reserved
1H	
2H	
3H	
4H	Reserved
5H	
6H	
7H	
8H	Reserved
9H	
AH	
BH	
CH	Reserved
DH	
EH	
FH	

Not Recommended for New Design

18.2 Bus Error Area

FFFF_C000 FFFF_C0FF	Ports 0 to 3
FFFF_C100 FFFF_C13F	(Note 1)
FFFF_C140 FFFF_C2FF	Ports 5 to B
FFFF_C300 FFFF_C37F	PMD
FFFF_C380 FFFF_C3FF	(Note 1)
FFFF_C400 FFFF_C43F	ENC
FFFF_C440 FFFF_C47F	(Note 1)
FFFF_C480 FFFF_C4DF	SIO
FFFF_C4E0 FFFF_C6FF	(Note 1)
FFFF_C700 FFFF_C77F	TMRB
FFFF_C780 FFFF_C82F	(Note 1)
FFFF_C830 FFFF_C83F	WDT
FFFF_C840 FFFF_C8FF	(Note 1)
FFFF_C900 FFFF_C9FF	ADC (Normal)
FFFF_CA00 FFFF_CCFF	(Note 1)
FFFF_CD00 FFFF_CDFF	ADC (PMD)
FFFF_CE00 FFFF_CFFF	(Note 1)

FFFF_D000 FFFF_D0FF	IRC
FFFF_D080 FFFF_D2FF	(Note 1)
FFFF_D300 FFFF_D33F	CG
FFFF_D340 FFFF_D3FF	(Note 1)
FFFF_D400 FFFF_D4FF	MODEC
FFFF_D500 FFFF_D5FF	(Note 1)
FFFF_D600 FFFF_D61B	DMAC
FFFF_D61C FFFF_D61F	(Note 2)
FFFF_D620 FFFF_D63B	DMAC
FFFF_D63C FFFF_D63F	(Note 2)
FFFF_D640 FFFF_D65B	DMAC
FFFF_D65C FFFF_D65F	(Note 2)
FFFF_D660 FFFF_D67B	DMAC
FFFF_D67C FFFF_D67F	(Note 2)
FFFF_D680 FFFF_D69B	DMAC
FFFF_D69C FFFF_D69F	(Note 2)
FFFF_D6A0 FFFF_D6BB	DMAC

FFFF_D6BC FFFF_D6BF	(Note 2)
FFFF_D6C0 FFFF_D6DB	DMAC
FFFF_D6DC FFFF_D6DF	(Note 2)
FFFF_D6E0 FFFF_D6FB	DMAC
FFFF_D6FC FFFF_D6FF	(Note 2)
FFFF_D700 FFFF_D707	DMAC
FFFF_D708 FFFF_D70B	(Note 2)
FFFF_D70C FFFF_D70F	DMAC
FFFF_D710 FFFF_D7FF	(Note 2)
FFFF_D800 FFFF_E3FF	(Note 1)
FFFF_E400 FFFF_E40F	Reserved
FFFF_E410 FFFF_E47F	(Note 1)
FFFF_E480 FFFF_E48B	Reserved
FFFF_E48C FFFF_E4FF	(Note 2)
FFFF_E500 FFFF_E6FF	Flash /ROM
FFFF_E700 FFFF_FFFF	(Note 1)

Note 1: Bus error area. A store access does not cause a bus error exception, but a NMI occurs.(MODECR<BERCTL>=0)

Note 2: Bus error area, but a store access does not cause a NMI.

Not Recommended for New Design

19. Electrical Characteristics

19.1 Maximum Ratings

The letter X in equations presented in this chapter represents the fsys or IMCLK period selected by the CLKPRSC.PRS1 or PRS2 field.

Mask-Version Product

Parameter		Symbol	Rating	Unit
Supply voltage		V_{CC15} (Core)	-0.3 to 3.0	V
		V_{CC3} (I/O)	-0.3 to 3.9	
		AVCC (AD)	-0.3 to 3.6	
Input voltage		V_{IN}	-0.3 to V_{CC3} + 0.3 (Note 1) -0.3 to AVCC + 0.3 (Note 2)	V
Low-level output current	Per pin	I_{OL}	15	mA
	Total	ΣI_{OL}	80	
High-level output current	Per pin	I_{OH}	-15	
	Total	ΣI_{OH}	-50	
Power dissipation (Ta = 85°C)		PD	600	mW
Soldering temperature	10 seconds	T_{SOLDER}	260	°C
	3 seconds	T_{SOLDER}	350	°C
Average temperature		Ta ave	-20 to 65	°C
Storage temperature		T_{STG}	-65 to 150	°C
Operating temperature		T_{OPR}	-40 to 85	°C

$V_{CC}15$=DVCC15=CVCC15, $V_{CC}3$=DVCC3, AVCC=AVCCn (n=0, 1), V_{SS}=DVSS=AVSS=CVSS

Note 1: The absolute maximum rating of $V_{CC}3$ (-0.3 to 3.9 V) must not be exceeded.

Note 2: Since Ports 5 to 7 use AVCC as the power supply for each port function, the maximum rating for AVCC (-0.3 to 3.6 V) should be applied to these ports.

Note 3: Maximum ratings are limiting values of operating and environmental conditions which should not be exceeded under the worst possible conditions. The equipment manumacturer should design so that no maximum rating value is exceeded with respect to current, voltage, power dissipation, temperature, etc. Exposure to conditions beyond those listed above may cause permanent damage to the device or affect device reliability, which could increase potential risks of personal injury due to IC blowup and/or burning.

Not Recommended for New Design

Flash-Version Product

Parameter		Symbol	Rating	Unit
Supply voltage		V_{CC2} (Core)	-0.3 to 3.6	V
		V_{CC3} (I/O)	-0.3 to 3.9	
		AVCC (A/D)	-0.3 to3.6	
Input voltage		V_{IN}	-0.3 to V_{CC3} + 0.3 (Note 1) -0.3 to AVCC + 0.3 (Note 2)	V
Low-level output current	Per pin	I_{OL}	15	mA
	Total	ΣI_{OL}	80	
High-level output current	Per pin	I_{OH}	-15	
	Total	ΣI_{OH}	-50	
Power dissipation (Ta = 85°C)		PD	1000	mW
Soldering temperature	10 seconds	T_{SOLDER}	260	°C
	3 seconds	T_{SOLDER}	350	°C
Average temperature		Ta ave	-20 to 65	°C
Storage temperature		T_{STG}	-65 to 150	°C
Operating temperature	Other than Flash program/erase	T_{OPR}	-40 to 85	°C
	Flash program/erase		-0 to 60	
Flash reprogram times		N_{WE}	100	cycle

$V_{CC}2$=DVCC2=CVCC2=FVCC2, $V_{CC}3$=FVCC3=DVCC3, AVCC=AVCCn (n=0, 1),
V_{SS}=DVSS=AVSS=CVSS=FVSS

Note 1: The absolute maximum rating of $V_{CC}3$ (-0.3 to 3.9V) must not be exceeded.

Note 2: Since Ports 5 to 7 use AVCC as the power supply for each port function, the maximum rating for AVCC (-0.3 to 3.6 V) should be applied to these ports.

Note 3: Maximum ratings are limiting values of operating and environmental conditions which should not be exceeded under the worst possible conditions. The equipment manufacturer should design so that no maximum rating value is exceeded with respect to current, voltage, power dissipation, temperature, etc. Exposure to conditions beyond those listed above may cause permanent damage to the device or affect device reliability, which could increase potential risks of personal injury due to IC blowup and/or burning.

Note 4: The number of times the flash memory can be reprogrammed includes programming of nonvolatile bits in the flash ROM. Note that programming nonvolatile bits to the same value is also counted in the reprogram times.

Not Recommended for New Design

19.2 Recommended Operating Conditions

Mask-Version Product

Ta = -40 to 85 °C

Parameter		Symbol	Conditions	Min.	Typ. (Note 1)	Max.	Unit
Supply voltage DVCC15=CVCC15		DVCC15	Input clock = 4 to 7 MHz fsys = 32 to 56 MHz	1.35		1.65	V
DVCC3 CVSS=DVSS=AVSS=0V		DVCC3		3.0		3.6	
		AVCCn		3.0		3.6	
Low-level input voltage	P0, P1, P23, P3, P80-P83, P85, P86, P94, PA0-PA5, PB0-PB5	V_{IL}	3.0 V ≦ DVCC3 ≦ 3.6 V	-0.3		0.3 DVCC3	V
	P5-P63 (Used as a port)	V_{IL1}	3.0 V ≦ AVCCn (n=0, 1) ≦ 3.6 V			0.3 AVCCn	
	P20-P22, P24, P64-P67, P70-P72, P84, P87, P90-P93, P95, PA6, PA7, PB6, PB7, P95/NMI, RESET	V_{IL2}	3.0 V ≦ DVCC3 ≦ 3.6 V 3.0 V ≦ AVCCn (n=0, 1) ≦ 3.6 V			0.2 DVCC3 (0.2 AVCCn)	
	X1	V_{IL3}	1.35 V ≦ CVCC15 ≦ 1.65 V			0.1 CVCC15	
High-level input voltage	P0, P1, P23, P3, P80-P83, P85-P86, P94, PA0-PA5, PB0-PB5	V_{IH}	3.0 V ≦ DVCC3 ≦ 3.6 V	0.7 DVCC3		DVCC3	V
	P5-P63 (Used as a port)	V_{IH1}	3.0 V ≦ AVCCn (n=0, 1) ≦ 3.6 V	0.7 AVCCn		AVCCn	
	P20-P22, P24, P64-P67, P70-P72, P84, P87, P90-P93, P95, PA6, PA7, PB6, PB7, P95/NMI, RESET	V_{IH2}	3.0 V ≦ DVCC3 ≦ 3.6 V 3.0 V ≦ AVCCn (n=0, 1) ≦ 3.6 V	0.8 DVCC3 (0.8 AVCCn)		DVCC3 (AVCCn)	
	X1	V_{IH3}	1.35 V ≦ CVCC15 ≦ 1.65 V	0.9 CVCC15		CVCC15	

Note 1: Recommended operating conditions are usage conditions recommended for proper operation of the device maintaining an expected level of quality. The equipment manufacturer should design so that no recommended operating condition is exceeded with respect to supply voltage, operating temperature range, AC/DC specifications, etc. Using the device under conditions beyond those listed above may cause the device to malfunction.

Note 2: No maximum absolute rating as well as recommended operating condition must ever be exceeded.

Note 3: Since AVCCn is also used as the power supply for Ports 5 to 7, it should be connected to a power source even if the AD converter is not used.

Note 4: Unless otherwise specified, the values specified for ports also apply to functions assigned to each port.

Not Recommended for New Design

Flash-Version Product

Ta = -40 to 85 °C

Parameter		Symbol	Conditions	Min.	Typ. (Note 1)	Max.	Unit
Supply voltage DVCC2=FVCC2=CVCC2 DVCC3=FVCC3 CVSS=DVSS=FVSS= AVSS=0V		DVCC2	Input clock = 4 to 7 MHz fsys = 32 to 56 MHz	2.3		2.7	V
		DVCC3		3.0		3.6	
		AVCCn		3.0		3.6	
Low-level input voltage	P0, P1, P23, P3, P80-P83, P85, P86, P94, PA0-PA5, PB0-PB5	V_{IL}	3.0 V ≦ DVCC3 ≦ 3.6 V	-0.3		0.3 DVCC3	V
	P5-P63 (Used as a port)	V_{IL1}	3.0 V ≦ AVCCn (n=0, 1) ≦ 3.6 V			0.3 AVCCn	
	P20-P22, P24, P64-P67, P70-P72, P84, P87, P90-P93, P95, PA6, PA7, PB6, PB7, P95/NMI, RESET	V_{IL2}	3.0V ≦ DVCC3 ≦ 3.6 V 3.0 V ≦ AVCCn (n=0, 1) ≦ 3.6 V			0.2 DVCC3 (0.2 AVCCn)	
	X1	V_{IL3}	2.3 V ≦ CVCC2 ≦ 2.7 V			0.1 CVCC2	
High-level input voltage	P0, P1, P23, P3, P80-P83, P85, P86, P94, PA0-PA5, PB0-PB5	V_{IH}	3.0 V ≦ DVCC3 ≦ 3.6 V	0.7 DVCC3		DVCC3	V
	P5-P63 (Used as a port)	V_{IH1}	3.0 V ≦ AVCCn (n=0, 1) ≦ 3.6 V	0.7 AVCCn		AVCCn	
	P20-P22, P24, P64-P67, P70-P72, P84, P87, P90-P93, P95, PA6, PA7, PB6, PB7, P95/NMI, RESET	V_{IH2}	3.0 V ≦ DVCC3 ≦ 3.6 V 3.0 V ≦ AVCCn (n=0, 1) ≦ 3.6 V	0.8 DVCC3 (0.8 AVCCn)		DVCC3 (AVCCn)	
	X1	V_{IH3}	2.3 V ≦ CVCC2 ≦ 2.7 V	0.9 CVCC2		CVCC2	

Note 1: Recommended operating conditions are usage conditions recommended for proper operation of the device maintaining an expected level of quality. The equipment manufacturer should design so that no recommended operating condition is exceeded with respect to supply voltage, operating temperature range, AC/DC specifications, etc. Using the device under conditions beyond those listed above may cause the device to malfunction.

Note 2: No maximum absolute rating as well as recommended operating condition must ever be exceeded.

Note 3: Since AVCCn is also used as the power supply for Ports 5 to 7, it should be connected to a power source even if the AD converter is not used.

Note 4: Unless otherwise specified, the values specified for ports also apply to functions assigned to each port.

Not Recommended for New Design

19.3 DC Electrical Characteristics (1/2)

Mask-Version Product Ta = -40 to 85 °C

Parameter		Symbol	Conditions		Min.	Typ. (Note1)	Max.	Unit
Low-level output voltage (Note 2)	Low drive capability	V_{OL}	I_{OL} = 0.5 mA	DVCC3 ≧ 3.0 V			0.4	V
	High drive capability		I_{OL} = 2 mA	DVCC3 ≧ 3.0 V				
			I_{OL} =10 mA	DVCC3 ≧ 3.0 V			1.0	
High-level output voltage (Note 2)	Low drive capability	V_{OH}	I_{OH} = −0.5 mA	DVCC3 ≧ 3.0 V	2.4			
	High drive capability		I_{OH} = −2 mA	DVCC3 ≧ 3.0 V				
Input leakage current		I_{LI}	0.0 ≦ V_{IN} ≦ DVCC3 0.0 ≦ V_{IN} ≦ AVCCn (n=0, 1)			0.02	± 5	μA
Output leakage current		I_{LO}	0.2 ≦ V_{IN} ≦ DVCC3−0.2 0.2 ≦ V_{IN} ≦ AVCCn−0.2 (N=0, 1)			0.05	± 10	
Hysteresis (Schmitt width) P20-P22, P24, P64-P67, P70-P72, P84, P87, P90-P93, P95, PA6, A7, PB6, PB7, P95/NMI, RESET		ΔVIN	3.0V≦DVCC3≦3.6V 3.0 V ≦ AVCCn (n=0, 1) ≦ 3.6 V		0.4	0.9	1.6	V
Pull-up resistor		PUP	DVCC3 = 3.0 V to 3.6 V		40	100	185	kΩ
Pin capacitance (excluding power supply pins)		C_{IO}	fc = 1 MHz				10	pF

Note 1: Ta = 25°C, DVCC3 = 3.3 V, DVCC15 = 1.5V and AVCCn = 3.3 V, unless otherwise noted.

Note 2: The drive capability can be set to low or high in the PnDSSR register for each port.

Flash-Version Product Ta= -40 to 85 °C

Parameter		Symbol	Conditions		Min.	Typ. (Note 1)	Max.	Unit
Low-level output voltage (Note 2)	Low drive capability	V_{OL}	I_{OL} = 0.5mA	DVCC3 ≧ 3.0V			0.4	V
	High drive capability		I_{OL} = 2mA	DVCC3 ≧ 3.0V				
			I_{OL} = 10mA	DVCC3 ≧ 3.0V			1.0	
High-level output voltage (Note 2)	Low drive capability	V_{OH}	I_{OH} = −0.5mA	DVCC3 ≧ 3.0V	2.4			
	High drive capability		I_{OH} = −2mA	DVCC3 ≧ 3.0V				
Input leakage current		I_{LI}	0.0 ≦ V_{IN} ≦ DVCC3 0.0 ≦ V_{IN} ≦ AVCCn (n=0, 1)			0.02	± 5	μA
Output leakage current		I_{LO}	0.2≦V_{IN}≦DVCC3−0.2 0.2 ≦ V_{IN} ≦AVCCn−0.2 (n=0, 1)			0.05	± 10	
Hysteresis (Schmitt width) P20-P22, P24, P64-P67, P70-P72, P84, P87, P90-P93, P95, PA6, PA7, PB6, PB7, P95/NMI, RESET		ΔVIN	3.0V ≦ DVCC3 ≦ 3.6 V 3.0 V ≦ AVCCn (n=0, 1) ≦ 3.6 V		0.4	0.9	1.6	V
Pull-up resistor		PUP	DVCC3 = 3.0 V to 3.6 V		40	100	185	kΩ
Pin capacitance (excluding power supply pins)		C_{IO}	fc = 1 MHz				10	pF

Note 1: Ta=25°C, DVCC3=3.3V, DVCC2=2.5V and AVCCn=3.3V, unless otherwise noted.

Note 2: The drive capability can be set to low or high in the PnDSSR register for each port.

Not Recommended for New Design

19.4 DC Electrical Characteristics (2/2)

Mask-Version Product

DVCC15 = CVCC15=1.35 V to 1.65 V, DVCC3 = 3.0 V to 3.6 V, AVCCn = 3.0 V to 3.6 V, Ta = -40 to 85℃ (n=0, 1)

Parameter	Symbol	Conditions	Min.	Typ. (Note 1)	Max. (Note 2)	Unit
NORMAL 1.5V (Note 3)	I_{CCN15}	f_{sys}=56 MHz (Input clock = 7 MHz, PLL x16, gear ratio = 1/2)		70	90	mA
IDLE (Doze)	I_{CCD}			45	60	
IDLE (Halt)	I_{CCH}			45	60	
STOP	I_{CCSt}	DVCC15=CVCC15=1.35 to 1.65V DVCC3= 3.0 to 3.6V AVCCn=3.0 to 3.6V		3	5	mA

Note 1: Ta=25℃, DVCC3=3.3V, DVCC15=1.5V and AVCCn=3.3V, unless otherwise noted.

Note 2: Max. values are theoretical maximum values that should not be exceeded under the worst possible conditions.

Note 3: I_{CCN} (Typ) measurement conditions: Run an arithemetic program provided by Toshiba with all internal peripheral active.

Flash-Version Product

DVCC2 = CVCC2 = 2.3 V to 2.7 V, DVCC3 = 3.0 V to 3.6 V, AVCCn = 3.0 V to 3.6 V, Ta = -40 to 85℃ (n = 0, 1)

Parameter	Symbol	Conditions	Min.	Typ. (Note 1)	Max. (Note 2)	Unit
NORMAL 2.5V (Note 3)	I_{CCN2}	f_{sys} = 56 MHz (Input clock = 7 MHz, PLL x16, gear ratio = 1/2)		212	285	mA
IDLE (Doze)	I_{CCD2}			130	200	
IDLE (Halt)	I_{CCH2}			120	190	
STOP	I_{CCSt}	DVCC2 = CVCC2 = 2.3 to 2.7 V DVCC3 = 3.0 to 3.6 V AVCCn = 3.0 to 3.6 V		11	1000	μA

Note 1: Ta=25℃, DVCC3=3.3V, DVCC2=2.5V and AVCCn=3.3V, unless otherwise noted.

Note 2: Max. values are theoretical maximum values that should not be exceeded under the worst possible conditions.

Note 3: I_{CCN} (Typ) measurement conditions: Run an arithmetic program provided by Toshiba with all internal peripherals acitve.

Not Recommended for New Design

19.5 10-Bit AD Conversion Characteristics

Mask-Version Product

DVCC15 = CVCC15 = 1.35 to 1.65 V, DVCC3 = 3.0 to 3.6 V, AVCCn = VREFH = 3.0 to 3.6 V,
AVSS = DVSS = VREFL = 0 V, Ta = -40 to 85 °C

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Analog reference voltage (+) (Note 3)	VREFH		3.0		3.6	V
Analog input voltage	VAIN		AVSS		AVCCn	V
Integral nonlinearity error (Note 6)	—	AVCCn = VREFH = 3.0 to 3.6 V DVSS = AVSS AVCCn load capacitance ≧ 10 μF VREFH load capacitance ≧ 10 μF Conversion time ≧ 2.36 μs		± 1.5	± 3	LSB
Differential nonlinearity error	—			± 1	± 2	LSB
Offset error	—			± 4	± 7	LSB
Gain error	—			± 2	± 4	LSB
Relative error (Note 5)	—			4	8	LSB
Total error	—			± 4	± 7	LSB

Note 1: 1 LSB = (VREFH–VREFL)/1024 [V]

Note 2: The supply current flowing through the AVCCn pin is included in the digital supply current parameter (I_{CC}).

Note 3: The VREFHn pin is shared with the AVCCn pin.

Note 4: The above characteristics apply when the ADC input pins are not used for other functions.

Note 5: Indicates the difference between the minimum and maximum conversion errors.

Note 6: Indicates a value after offset and gain errors have been adjusted.

Not Recommended for New Design

Flash-Version Product

DVCC2 = FVCC2 = CVCC2 = 2.5 ±0.2 V, DVCC3 = 3.3 ±0.3 V, AVCCn = VREFH = 3.0 to 3.6 V,
AVSS = DVSS = VREFL = 0 V, Ta = -40 to 85 °C

Parameter	Symbol	Condition	Min	Typ	Max	Unit
Analog reference voltage (+) (Note 3)	VREFH		3.0		3.6	V
Analog input voltage	VAIN		AVSS		AVCCn	V
Integral nonlinearity error (Note 6)	—	AVCCn = VREFH = 3.0 to 3.6V DVSS = AVSS AVCCn load capacitance ≧10μF VREFH load capacitance ≧10μF Conversion time≧2.36μs		± 1.5	± 3	LSB
Differential nonlinearity error	—			± 1	± 2	LSB
Offset error	—			± 4	± 7	LSB
Gain error	—			± 2	± 4	LSB
Relative error (Note 6)	—			4	8	LSB
Total error	—			± 4	± 7	LSB

Note 1: 1 LSB = (VREFH–VREFL)/1024 [V]

Note 2: The supply current flowing through the AVCCn pin is included in the digital supply current parameter (I_{CC}).

Note 3: The VREFHn pin is shared with the AVCCn pin.

Note 4: The above characteristics apply when the ADC input pins are not used for other functions.

Note 5: Indicates the difference between the minimum and maximum conversion errors.

Note 6: Indicates a value after offset and gain errors have been adjusted.

Not Recommended for New Design

19.6 SIO Timings

Mask-Version Product

(1) I/O Interface Mode (DVCC3 = 3.3 ±0.3 V, DVCC15 = 1.5 ±0.15 V, Ta = -40 to 85 °C)

The letter x in the tables below represents the system clock fsys period which depends on the clock gear setting.

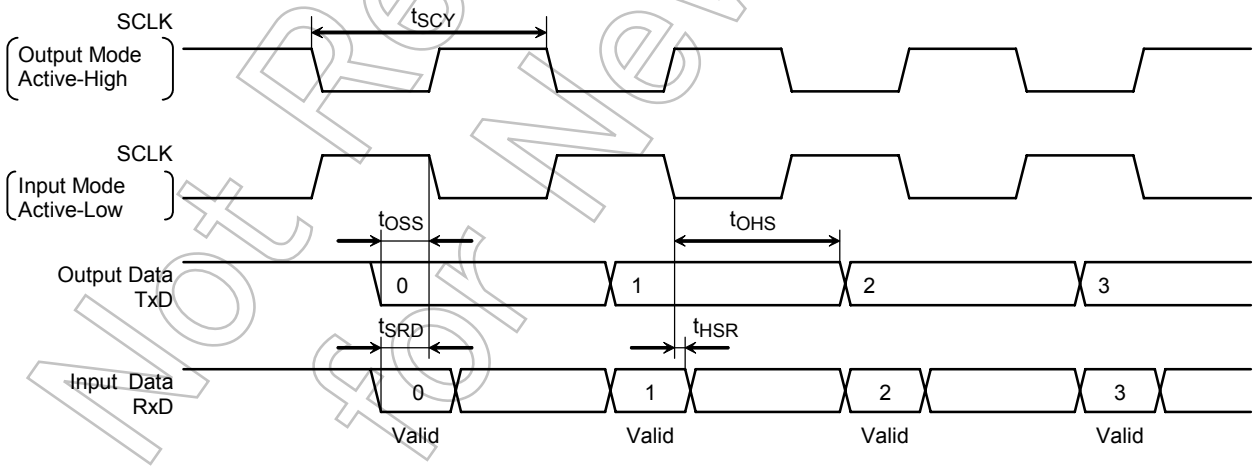
- SCLK Input mode (SIO2)

Parameter	Symbol	Equation		56 MHz		Unit
		Min	Max	Min	Max	
SCLK period	t_{SCY}	16x		286		ns
TxD data to SCLK rise or fall *	t_{OSS}	(tSCY/2) - 4x - 23		50		ns
TxD data hold after SCLK rise or fall*	t_{OHS}	(tSCY/2) + 2x		179		ns
RxD data valid to SCLK rise or fall*	t_{SRD}	2x + 8		44		ns
RxD data hold after SCLK rise or fall*	t_{HSR}	0		0		ns

*) SCLK rise or fall: Measured relative to the programmed active edge of SCLK.

- SCLK Output mode (SIO2)

Parameter	Symbol	Equation		56 MHz		Unit
		Min	Max	Min	Max	
SCLK period (programmable)	t_{SCY}	16x		286		ns
TxD data to SCLK rise	t_{OSS}	(tSCY/2) - 15		128		ns
TxD data hold after SCLK rise	t_{OHS}	(tSCY/2) - 15		128		ns
RxD data valid to SCLK rise	t_{SRD}	2X + 23		59		ns
RxD data hold after SCLK rise	t_{HSR}	0		0		ns



Note 1: Output level measurement conditions: High 0.8DVCC3 [V] / Low 0.2DVCC3 [V], CL=30 pF

Note 2: Input level measurement conditions: High 0.7DVCC3 [V] / Low 0.2DVCC3 [V]

Not Recommended for New Design

Flash-Version Product

(1) I/O Interface mode (DVCC3 = 3.3 ±0.3 V, DVCC2 = 2.5 ±0.2 V, Ta = -40 to 85 ºC)

The letter x in the tables below represents the system clock fsys period which depends on the clock gear setting.

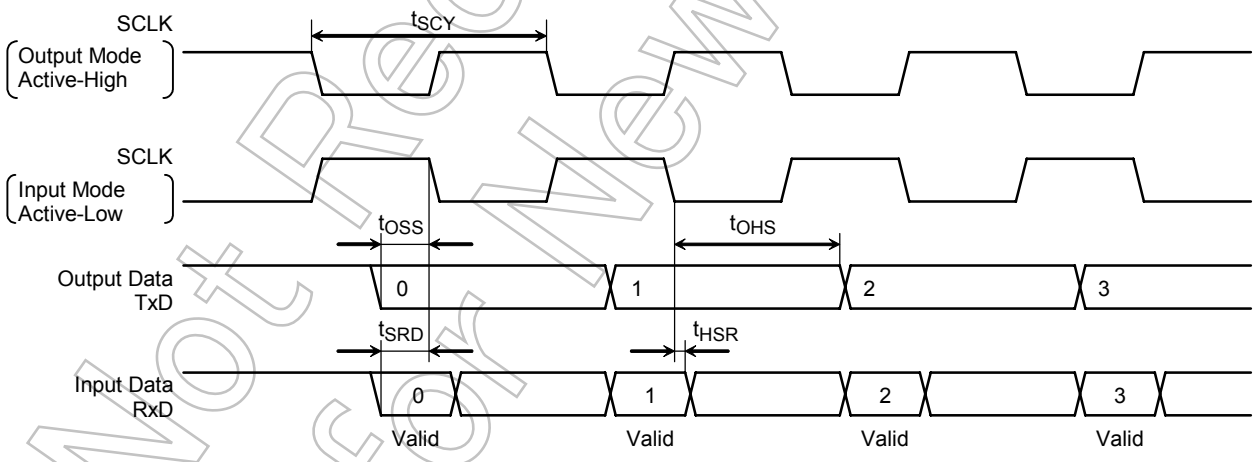
- SCLK Input mode (SIO2)

Parameter	Symbol	Equation		56 MHz		Unit
		Min	Max	Min	Max	
SCLK period	t_{SCY}	16x		286		ns
TxD data to SCLK rise or fall*	t_{OSS}	(tSCY/2) - 4x - 23		50		ns
TxD data hold after SCLK rise or fall*	t_{OHS}	(tSCY/2) + 2x		179		ns
RxD data valid to SCLK rise or fall*	t_{SRD}	2x + 8		44		ns
RxD data hold after SCLK rise or fall*	t_{HSR}	0		0		ns

*) SCLK rise or fall: Measured relative to the programmed active edge of SCLK

- SCLK Output mode (SIO2)

Parameter	Symbol	Equation		56 MHz		Unit
		Min	Max	Min	Max	
SCLK period (programmable)	t_{SCY}	16x		286		ns
TxD data to SCLK rise	t_{OSS}	(tSCY/2) – 15		128		ns
TxD data hold after SCLK rise	t_{OHS}	(tSCY/2) – 15		128		ns
RxD data valid to SCLK rise	t_{SRD}	2X + 23		59		ns
RxD data hold after SCLK rise	t_{HSR}	0		0		ns



Note 1: Output level measurement conditions: High 0.8DVCC3 [V] / Low 0.2DVCC3 [V], CL=30 pF

Note 2: Input level measurement conditions: High 0.7DVCC3 [V] / Low 0.2DVCC3 [V]

Not Recommended for New Design

19.7 Event Counter

Mask-Version and Flash-Version Products

The letter x in the table below represents the IMCLK period.

Parameter	Symbol	Equation		IMCLK = 28 MHz		Unit
		Min	Max	Min	Max	
Clock low-level pulse width	t_{VCKL}	X + 100		136		ns
Clock high-level pulse width	t_{VCKH}	X + 100		136		ns

19.8 Capture

Mask-Version and Flash-Version Products

The letter x in the table below represents the IMCLK period.

Parameter	Symbol	Equation		IMCLK = 28 MHz		Unit
		Min	Max	Min	Max	
Low-level pulse width	t_{CPL}	X + 100		136		ns
High-level pulse width	t_{CPH}	X + 100		136		ns

19.9 Interrupts (INTC)

Mask-Version and Flash-Version Products

The letter x in the table below represents the system clock fsys period.

Parameter	Symbol	Equation		fsys = 56 MHz		Unit
		Min	Max	Min	Max	
INT0 to A low-level pulse width	t_{INTAL}	X + 100		118		ns
INT0 to A high-level pulse width	t_{INTAH}	X + 100		118		ns

19.10 Interrupts (NMI, STOP wakeup interrupt)

Mask-Version and Flash-Version Products

Parameter	Symbol	Equation		fsys = 56 MHz		Unit
		Min	Max	Min	Max	
NMI, INT0 to 4 low-level pulse width	t_{INTBL}	100		100		ns
INT0 to 4 high-level pulse width	t_{INTBH}	100		100		ns

Not Recommended for New Design

19.11 ADTRG Input

Mask-Version and Flash-Version Products

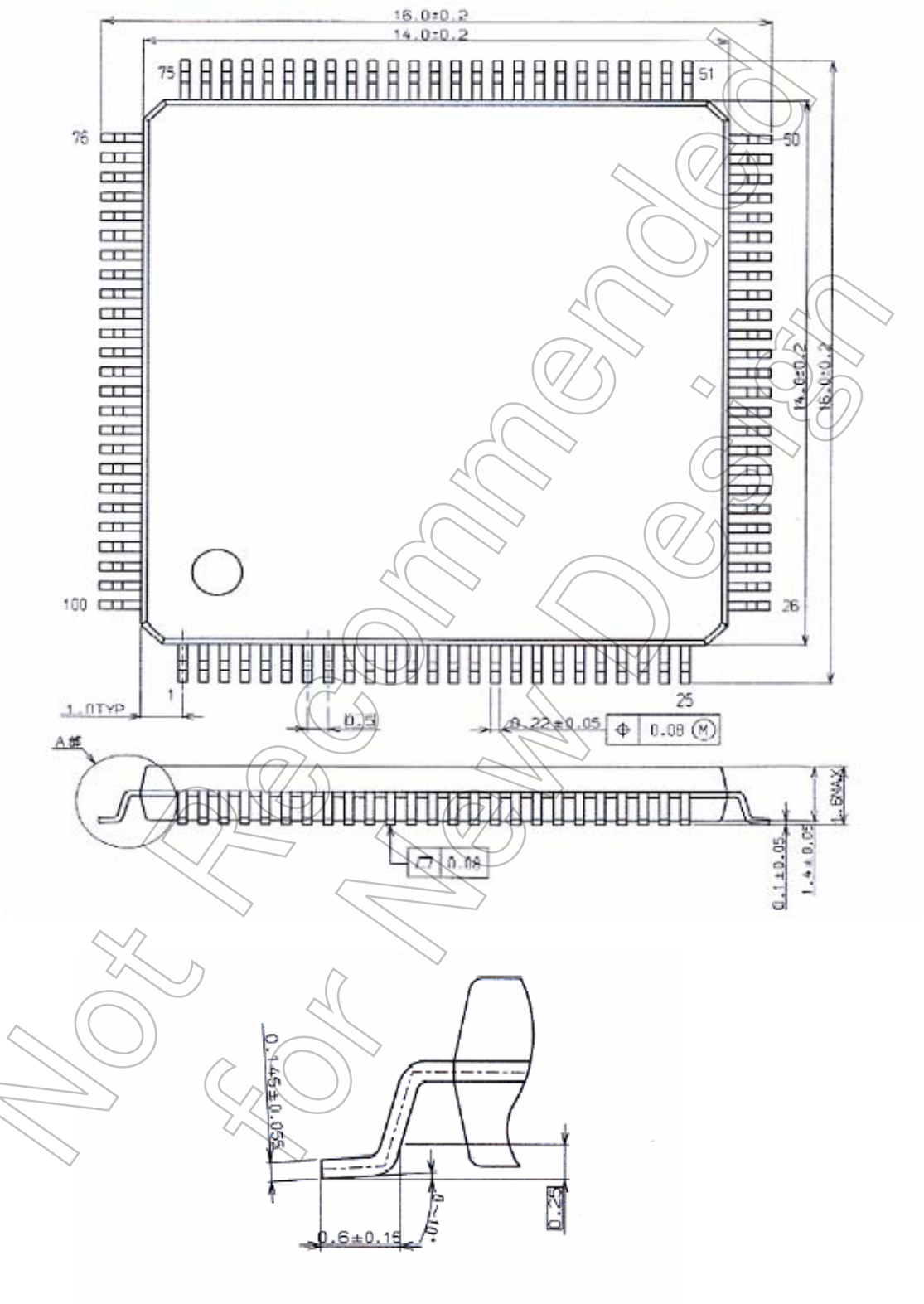
The letter x in the table below represents the IMCLK period.

Parameter	Symbol	Equation		IMCLK = 28 MHz		Unit
		Min	Max	Min	Max	
ADTRG low-level pulse width	tad_L	X + 100		136		ns
ADTRG high-level pulse width	tadh	X + 100		136		ns

Not Recommended for New Design

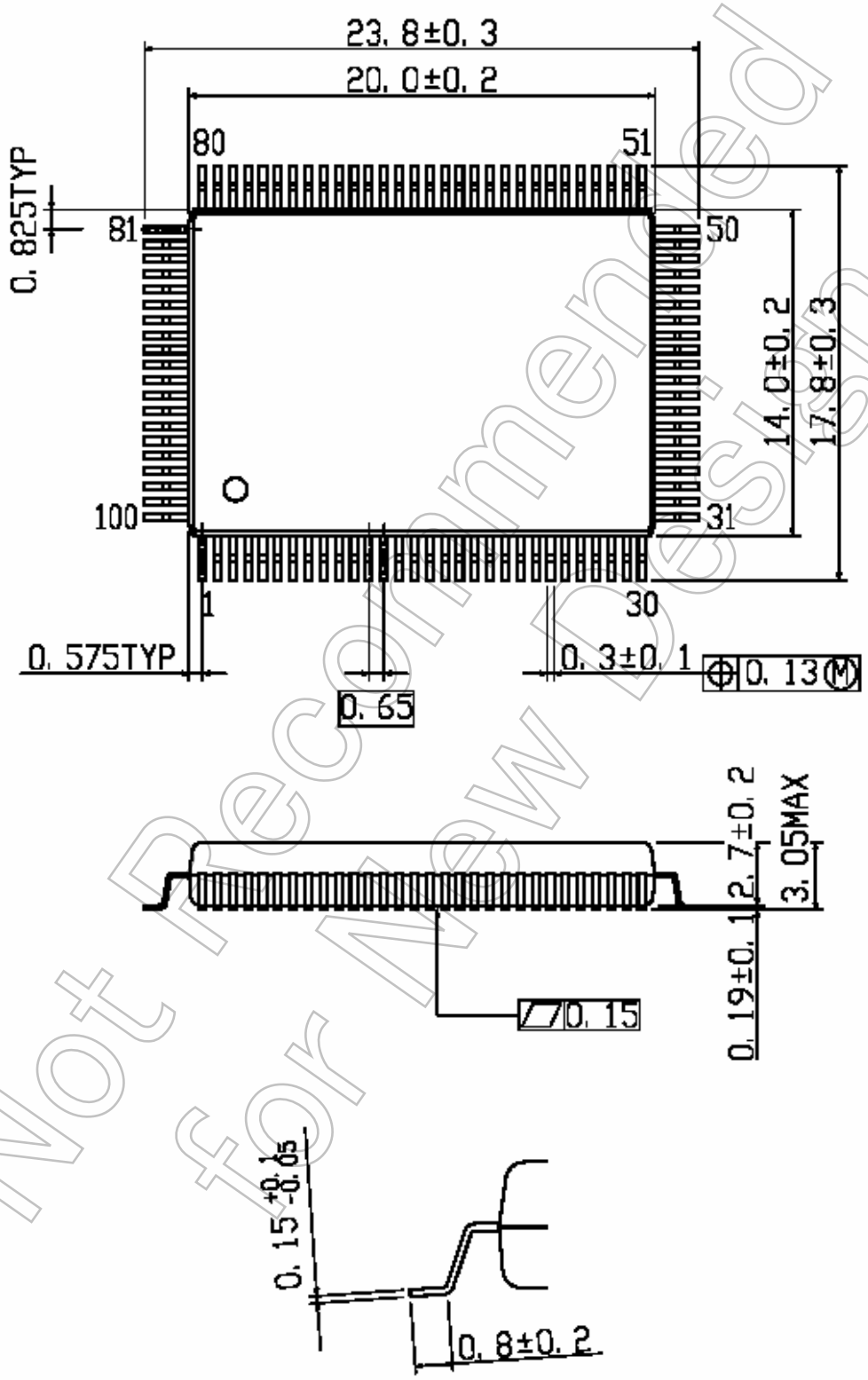
20. Package Dimensions

20.1 P-LQFP-1414-0.50F



Not Recommended for New Design

20.2 P-QFP-1420-0.65A



Not Recommended for New Design