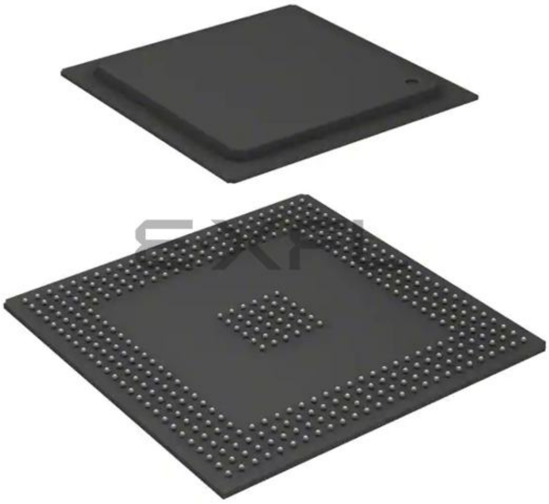
# NXP USA Inc. - MPC564MVR56R2 Datasheet



## What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

## Applications of "Embedded - Microcontrollers"

| Details | |
|---|---|
| Product Status | Obsolete |
| Core Processor | PowerPC |
| Core Size | 32-Bit Single-Core |
| Speed | 56MHz |
| Connectivity | CANbus, EBI/EMI, SCI, SPI, UART/USART |
| Peripherals | POR, PWM, WDT |
| Number of I/O | 56 |
| Program Memory Size | 512KB (512K x 8) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 32K x 8 |
| Voltage - Supply (Vcc/Vdd) | 2.5V ~ 2.7V |
| Data Converters | A/D 32x10b |
| Oscillator Type | External |
| Operating Temperature | -40°C ~ 125°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 388-BBGA |
| Supplier Device Package | 388-PBGA (27x27) |
| Purchase URL | https://www.e-xfl.com/product-detail/nxp-semiconductors/mpc564mvr56r2 |

# Contents

**Paragraph Number** **Title** **Page Number**

- A specified CR field can be the explicit result of an integer compare instruction.

Instructions are provided to test individual CR bits.

#### 3.7.4.1 Condition Register CR0 Field Definition

In most integer instructions, when the CR is set to reflect the result of the operation (that is, when Rc = 1), and for addic., andi., and andis., the first three bits of CR0 are set by an algebraic comparison of the result to zero; the fourth bit of CR0 is copied from XER[SO]. For integer instructions, CR0[0:3] are set to reflect the result as a signed quantity. The EQ bit reflects the result as an unsigned quantity or bit string.

The CR0 bits are interpreted as shown in Table 3-7. If any portion of the result (the 32-bit value placed into the destination register) is undefined, the value placed in the first three bits of CR0 is undefined.

**Table 3-7. Bit Settings for CR0 Field of CR**

| CR0 Bit | Description |
|---|---|
| 0 | Negative (LT). This bit is set when the result is negative. |
| 1 | Positive (GT). This bit is set when the result is positive (and not zero). |
| 2 | Zero (EQ). This bit is set when the result is zero. |
| 3 | Summary overflow (SO). This is a copy of the final state of XER[SO] at the completion of the instruction. |

#### 3.7.4.2 Condition Register CR1 Field Definition

In all floating-point instructions when the CR is set to reflect the result of the operation (that is, when Rc = 1), the CR1 field (bits 4 to 7 of the CR) is copied from FPSCR[0:3] to indicate the floating-point exception status. For more information about the FPSCR, see Section 3.7.3, “Floating-Point Status and Control Register (FPSCR).” The bit settings for the CR1 field are shown in Table 3-8.

**Table 3-8. Bit Settings for CR1 Field of CR**

| CR1 Bit | Description |
|---|---|
| 0 | Floating-point exception (FX). This is a copy of the final state of FPSCR[FX] at the completion of the instruction. |
| 1 | Floating-point enabled exception (FEX).This is a copy of the final state of FPSCR[FEX] at the completion of the instruction. |
| 2 | Floating-point invalid exception (VX).This is a copy of the final state of FPSCR[VX] at the completion of the instruction. |
| 3 | Floating-point overflow exception (OX).This is a copy of the final state of FPSCR[OX] at the completion of the instruction. |

#### 3.7.4.3 Condition Register CR*n* Field — Compare Instruction

When a specified CR field is set by a compare instruction, the bits of the specified field are interpreted as shown in Table 3-9. A condition register field can also be accessed by the mfcr, mcrf, and mtcrf instructions.

Table 3-10. Integer Exception Register Bit Descriptions

| Bits | Name | Description |
|---|---|---|
| 2 | CA | Carry (CA). In general, the carry bit is set to indicate that a carry out of bit 0 occurred during execution of an instruction. Add carrying, subtract from carrying, add extended, and subtract from extended instructions set CA if there is a carry out of bit 0, and clear it otherwise. The CA bit is not altered by compare instructions or other instructions that cannot carry, except that shift right algebraic instructions set the CA bit to indicate whether any ‘1’ bits have been shifted out of a negative quantity. |
| 3:24 | — | Reserved |
| 25:31 | BYTES | This field specifies the number of bytes to be transferred by a Load String Word Indexed (lswx) or Store String Word Indexed (stswx) instruction. |

## 3.7.6 Link Register (LR)

The link register (LR), SPR 8, supplies the branch target address for the branch conditional to link register (bclrx) instruction, and can be used to hold the logical address of the instruction that follows a branch and link instruction.

Note that although the two least-significant bits can accept any values written to them, they are ignored when the LR is used as an address.

Both conditional and unconditional branch instructions include the option of placing the effective address of the instruction after the branch instruction in the LR. This is done regardless of whether the branch is taken.

| | MSB<br>0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | LSB<br>31 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Branch Address | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset | Unchanged | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Addr | SPR 8 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Figure 3-9. Link Register (LR)

## 3.7.7 Count Register (CTR)

The count register (CTR), SPR 9, is used to hold a loop count that can be decremented during execution of branch instructions with an appropriately coded BO field. If the value in CTR is 0 before being decremented, it is –1 afterward. The count register provides the branch target address for the branch conditional to count register (bcctrx) instructio

| | MSB<br>0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | LSB<br>31 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Loop Count | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset | Unchanged | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Addr | SPR 9 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Figure 3-10. Count Register (CTR)

**Table 6-2. SGPIO Configuration**

| SGPIO Group Name | Individual Pin Control | Direction Control | Available When SC = 00 (32-bit Port Size Mode) | Available When SC = 01 (16-bit Port Size Mode) | Available When SC = 10 (Single-Chip Mode with Trace) | Available When SC = 11 (Single-Chip Mode) |
|---|---|---|---|---|---|---|
| SGPIOD[0:7] | | GDDR0 | | | X | X |
| SGPIOD[8:15] | | GDDR1 | | | X | X |
| SGPIOD[16:23] | | GDDR2 | | X | X | X |
| SGPIOD[24:31] | X | SDDRD[23:31] | | X | X | X |
| SGPIOC[0:7][1] | X | SDDRC[0:7] | | | | |
| SGPIOA[8:15] | | GDDR3 | | | | X |
| SGPIOA[16:23] | | GDDR4 | | | | X |
| SGPIOA[24:31] | | GDDR5 | | | | X |

[1] SGPIOC[0:7] is selected according to GPC and MLRC fields in SIUMCR. See Section 6.2.2.1.1, “SIU Module Configuration Register (SIUMCR).”

Figure 6-2 illustrates the functionality of the SGPIO.

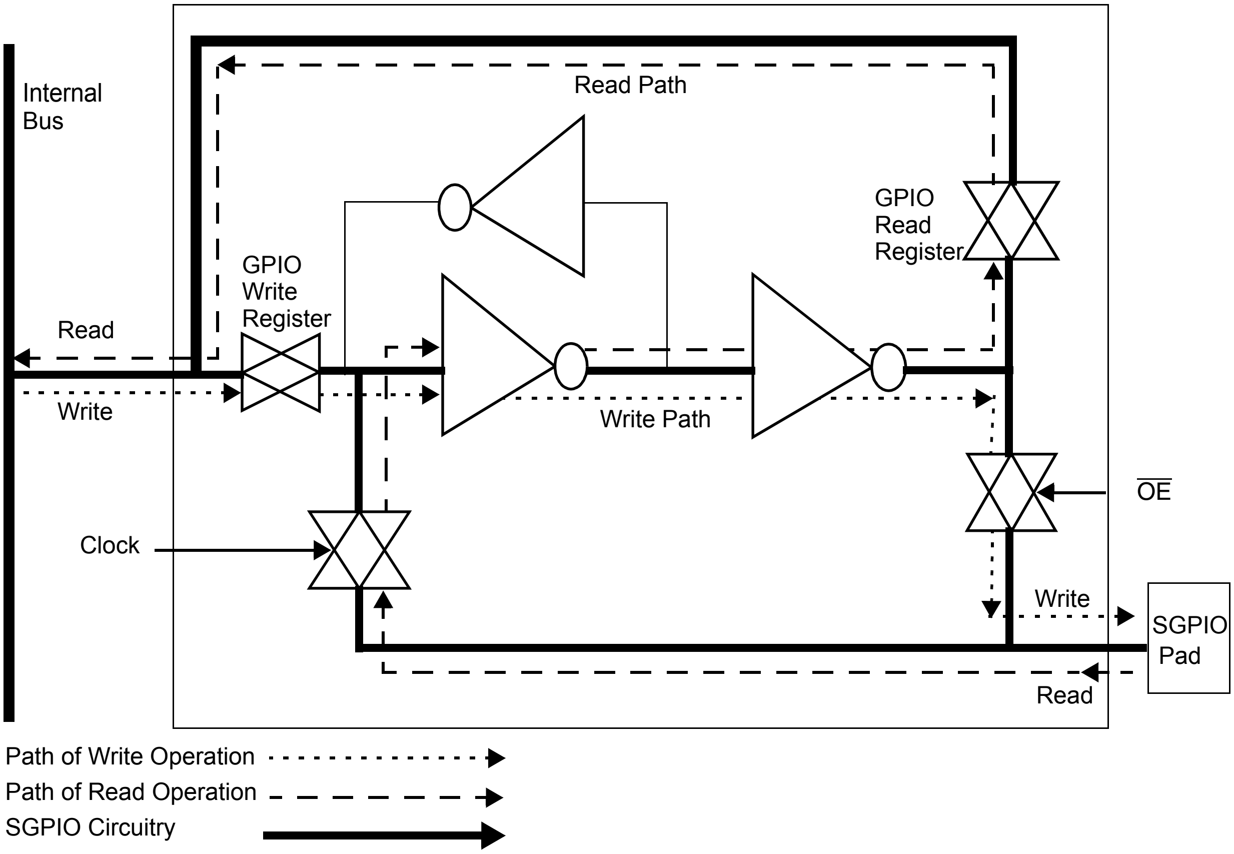


**Figure 6-2. Circuit Paths of Reading and Writing to SGPIO**

- XFC — External filter capacitor. XFC connects to the off-chip capacitor for the PLL filter. One terminal of the capacitor is connected to XFC, and the other terminal is connected to VDDSYN.
  - The off-chip capacitor must have the following values:
    - $0 < MF + 1 < 4 (1130 \times (MF + 1) - 80)$ pF
    - $MF + 1 \geq 42100 \times (MF + 1)$ pF

    Where MF = the value stored on MF[0:11]. This is one less than the desired frequency multiplication.

## 8.3 System Clock During PLL Loss of Lock

At reset, until the SPLL is locked, the SPLL output clock is disabled.

During normal operation (once the PLL has locked), either the oscillator or an external clock source is generating the system clock. In this case, if loss of lock is detected and the LOLRE (loss of lock reset enable) bit in the PLPRCR is cleared, the system clock source continues to function as the PLL's output clock. The USIU timers can operate with the input clock to the PLL, so that these timers are not affected by the PLL loss of lock. Software can use these timers to measure the loss-of-lock period. If the timer reaches the user-preset software criterion, the MPC561/MPC563 can switch to the backup clock by setting the switch to backup clock (STBUC) bit in the SCCR, provided the limp mode enable (LME) bit in the SCCR is set.

If loss of lock is detected during normal operation, assertion of $\overline{\text{HRESET}}$ (for example, if LOLRE is set) disables the PLL output clock until the lock condition is met. During hard reset, the STBUC bit is set as long as the PLL lock condition is not met and clears when the PLL is locked. If STBUC and LME are both set, the system clock switches to the backup clock (BUCLK), and the chip operates in limp mode until STBUC is cleared.

Every change in the lock status of the PLL can generate a maskable interrupt.

**NOTE**

When the VCO is the system clock source, chip operation is unpredictable while the PLL is unlocked. Note further that a switch to the backup clock is possible only if the LME bit in the SCCR is set.

## 8.4 Low-Power Divider

The output of the PLL is sent to a low-power divider block. (In limp mode the BUCLK is sent to a low-power divider block.) This block generates all other clocks in normal operation, but has the ability to divide the output frequency of the VCO before it generates the general system clocks sent to the rest of the MPC561/MPC563. The PLL VCOOUT is always divided by at least two.

The purpose of the low-power divider block is to allow reduction and restoration of the operating frequencies of different sections of the MPC561/MPC563 without losing the PLL lock. Using the low-power divider block, full chip operation can still be obtained, but at a lower frequency. This is called gear mode. The selection and speed of gear mode can be changed at any time, with changes occurring immediately.

sub-queue CCWs, all of the samples are guaranteed to have been taken during the same scan pass. However, a high trigger event rate for queue 1 can prohibit the completion of queue 2. If this occurs, the software may choose to begin execution of queue 2 with the aborted CCW entry.

- Software can change the queue operating mode to disabled mode. Any conversion in progress for that queue is aborted. Putting a queue into the disabled mode does not power down the converter.
- Software can change the queue operating mode to another valid mode. Any conversion in progress for that queue is aborted. The queue restarts at the beginning of the queue, once an appropriate trigger event occurs.
- For low power operation, software can set the stop mode bit to prepare the module for a loss of clocks. The QADC64E aborts any conversion in progress when the stop mode is entered.
- When the freeze enable bit is set by software and the IMB3 internal FREEZE line is asserted, the QADC64E freezes at the end of the conversion in progress. When internal FREEZE is negated, the QADC64E resumes queue execution beginning with the next CCW entry. Refer to Section 13.5.7, “Configuration and Control Using the IMB3 Interface” for more information.

| | MSB | | | | | | | | | | | | | | | LSB |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| Field | — | | | | | | P | BYP | IST | | CHAN[5:0] | | | | | |
| Reset | Unaffected | | | | | | | | | | | | | | | |
| Addr | 0x30 4A00 – 0x30 4A7F, 0x30 4E00 – 0x30 4E7F | | | | | | | | | | | | | | | |

**Figure 13-16. Conversion Command Word Table (CCW)**

**Table 13-18. CCW Bit Descriptions**

| Bits | Name | Description |
|---|---|---|
| 0:5 | — | Reserved |
| 6 | P | Pause. The pause bit allows the creation of sub-queues within queue 1 and queue 2. The QADC64E performs the conversion specified by the CCW with the pause bit set, and then the queue enters the pause state. Another trigger event causes execution to continue from the pause to the next CCW.<br>0 Do not enter the pause state after execution of the current CCW.<br>1 Enter the pause state after execution of the current CCW. |
| 7 | BYP | Sample amplifier bypass. Setting BYP enables the amplifier bypass mode for a conversion, and subsequently changes the timing. Refer to Section 13.4.1.2, “Amplifier Bypass Mode Conversion Timing,” for more information.<br>0 Amplifier bypass mode disabled.<br>1 Amplifier bypass mode enabled. |

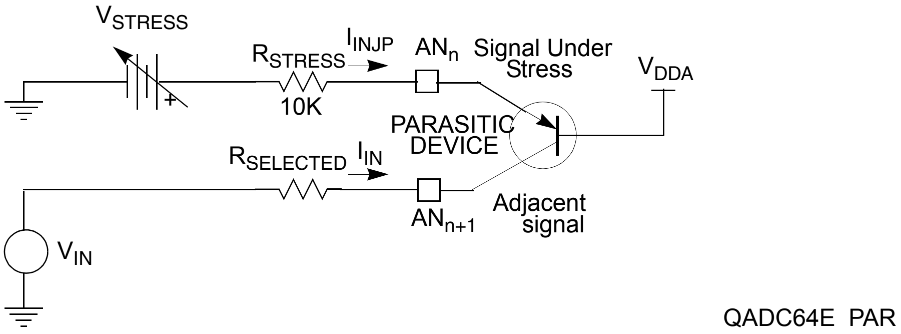


**Figure 13-55. Input Signal Subjected to Positive Stress**

The current into the signal ($I_{INJN}$ or $I_{INJP}$) under negative or positive stress is determined by the following equations:

$$I_{INJN} = \frac{-(V_{STRESS} - V_{BE})}{R_{STRESS}} \qquad \textit{Eqn. 13-2}$$

$$I_{INJP} = \frac{V_{STRESS} - V_{EB} - V_{DDA}}{R_{STRESS}} \qquad \textit{Eqn. 13-3}$$

where:

$V_{STRESS}$ = Adjustable voltage source

$V_{EB}$ = Parasitic PNP emitter/base voltage
(refer to $V_{NEGCLAMP}$ in Appendix F, “Electrical Characteristics”)

$V_{BE}$ = Parasitic NPN base/emitter voltage
(refer to $V_{NEGCLAMP}$ in Appendix F, “Electrical Characteristics”)

$R_{STRESS}$ = Source impedance
(10-kΩ resistor in Figure 13-54 and Figure 13-55 on stressed channel)

$R_{SELECTED}$ = Source impedance on channel selected for conversion

The current into ($I_{IN}$) the neighboring signal is determined by the $K_N$ (current coupling ratio) of the parasitic bipolar transistor ($K_N << 1$). The $I_{IN}$ can be expressed by the following equation:

$I_{IN} = - K_N * I_{INJ}$

where $I_{INJ}$ is either $I_{INJN}$ or $I_{INJP}$.

A method for minimizing the impact of stress conditions on the QADC64E is to strategically allocate QADC64E inputs so that the lower accuracy inputs are adjacent to the inputs most likely to see stress conditions.

Also, suitable source impedances should be selected to meet design goals and minimize the effect of stress conditions.

- Programmable Transfer Delay — from 0.6 µs to 0.3 µs (at 28 MHz)
- Programmable Queue Pointer
- Continuous Transfer Mode — up to 256 bits
- Optional on-chip expanded QSPI chip selects

Standard SCI features are listed below, followed by a list of additional features offered.

Standard SCI two-wire system features:

- Standard nonreturn-to-zero (NRZ) mark/space format
- Advanced error detection mechanism (detects noise duration up to 1/16 of a bit-time)
- Full-duplex operation
- Software selectable word length (8- or 9-bit words)
- Separate transmitter and receiver enable bits
- May be interrupt driven
- Four separate interrupt enable bits
- Two independent operating SCI modules

Standard SCI receiver features:

- Receiver wakeup function (idle or address mark bit)
- Idle-line detect
- Framing, noise, and overrun error detect
- Receive data register full flag

Standard SCI transmitter features:

- Transmit data register empty flag
- Transmit complete flag
- Send break

QSMCM-additional SCI features:

- 13-bit programmable baud-rate modulus counter
- Even/odd parity generation and detection
- Two idle-line detect modes
- Receiver active flag

QSMCM-enhanced SCI features:

- 16 register receive buffer on one SCI
- 16 register transmit buffer on one SCI

### 15.2.1 MPC561/MPC563 QSMCM Details

The QSMCM module has an identical function to the MPC555. The MUXing of the pins is controlled by the QPAPCS3 bit in the QSMCM pin assignment register (PQSPAR).

The operation of the receiver bit processor is shown in Figure 15-30. This example demonstrates the search for a valid start bit and the synchronization procedure as outlined above. The possibilities of noise durations greater than one bit-time are not considered in this examples.

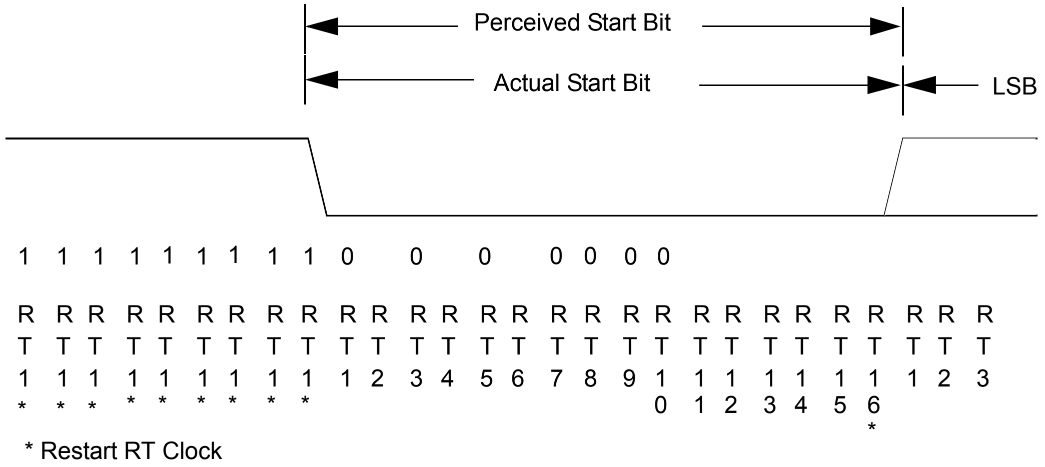

Perceived Start Bit
Actual Start Bit
LSB
1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0
RT1* RT1* RT1* RT1* RT1* RT1* RT1* RT1* RT1* RT1 RT2 RT3 RT4 RT5 RT6 RT7 RT8 RT9 RT10 RT11 RT12 RT13 RT14 RT15 RT16* RT1 RT2 RT3
* Restart RT Clock

**Figure 15-30. Start Search Example**

### 15.7.7.8 Receiver Functional Operation

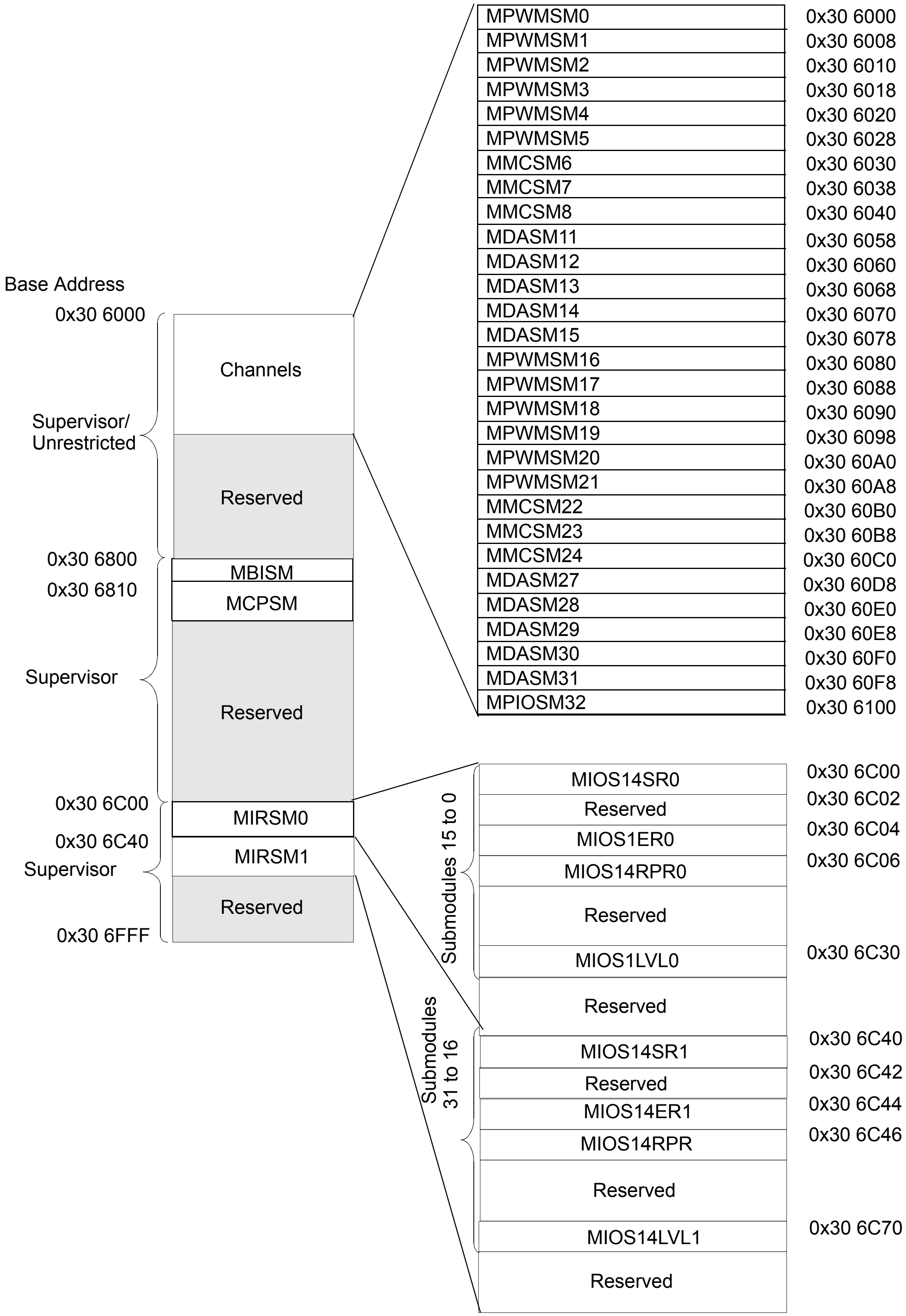
The RE bit in SCCxR1 enables (RE = 1) and disables (RE = 0) the receiver. The receiver contains a receive serial shifter and a parallel receive data register (RDRx) located in the SCI data register (SCxDR). The serial shifter cannot be directly accessed by the CPU. The receiver is double-buffered, allowing data to be held in the RDRx while other data is shifted in.

Receiver bit processor logic drives a state machine that determines the logic level for each bit-time. This state machine controls when the bit processor logic is to sample the RXD pin and also controls when data is to be passed to the receive serial shifter. A receive time clock is used to control sampling and synchronization. Data is shifted into the receive serial shifter according to the most recent synchronization of the receive time clock with the incoming data stream. From this point on, data movement is synchronized with the MCU IMB3 clock. Operation of the receiver state machine is detailed in the *Queued Serial Module Reference Manual*.

The number of bits shifted in by the receiver depends on the serial format. However, all frames must end with at least one stop bit. When the stop bit is received, the frame is considered to be complete, and the received data in the serial shifter is transferred to the RDRx. The receiver data register flag (RDRF) is set when the data is transferred.

The stop bit is always a logic one. If a logic zero is sensed during this bit-time, the FE flag in SCxSR is set. A framing error is usually caused by mismatched baud rates between the receiver and transmitter or by a significant burst of noise. Note that a framing error is not always detected; the data in the expected stop bit-time may happen to be a logic one.

Noise errors, parity errors, and framing errors can be detected while a data stream is being received. Although error conditions are detected as bits are received, the noise flag (NF), the parity flag (PF), and the framing error (FE) flag in SCxSR are not set until data is transferred from the serial shifter to the RDRx.


Base Address
0x30 6000
Supervisor/
Unrestricted
Channels
Reserved
0x30 6800
MBISM
0x30 6810
MCPSM
Supervisor
Reserved
0x30 6C00
MIRSM0
0x30 6C40
MIRSM1
Supervisor
Reserved
0x30 6FFF
MPWMSM0 0x30 6000
MPWMSM1 0x30 6008
MPWMSM2 0x30 6010
MPWMSM3 0x30 6018
MPWMSM4 0x30 6020
MPWMSM5 0x30 6028
MMCSM6 0x30 6030
MMCSM7 0x30 6038
MMCSM8 0x30 6040
MDASM11 0x30 6058
MDASM12 0x30 6060
MDASM13 0x30 6068
MDASM14 0x30 6070
MDASM15 0x30 6078
MPWMSM16 0x30 6080
MPWMSM17 0x30 6088
MPWMSM18 0x30 6090
MPWMSM19 0x30 6098
MPWMSM20 0x30 60A0
MPWMSM21 0x30 60A8
MMCSM22 0x30 60B0
MMCSM23 0x30 60B8
MMCSM24 0x30 60C0
MDASM27 0x30 60D8
MDASM28 0x30 60E0
MDASM29 0x30 60E8
MDASM30 0x30 60F0
MDASM31 0x30 60F8
MPIOSM32 0x30 6100
Submodules 15 to 0
MIOS14SR0 0x30 6C00
Reserved 0x30 6C02
MIOS1ER0 0x30 6C04
MIOS14RPR0 0x30 6C06
Reserved
MIOS1LVL0 0x30 6C30
Reserved
Submodules
31 to 16
MIOS14SR1 0x30 6C40
Reserved 0x30 6C42
MIOS14ER1 0x30 6C44
MIOS14RPR 0x30 6C46
Reserved
MIOS14LVL1 0x30 6C70
Reserved

**NOTE**

> If a submodule in a group of 16 cannot generate interrupts, then its corresponding flag bit in the status register is inactive and is read as zero.

When an event occurs in a submodule that activates a flag line, the corresponding flag bit in the status register is set. The status register is read/write, but a flag bit can be reset only if it has previously been read as a one. Writing a “one” to a flag bit has no effect. When the software intends to clear only one flag bit within a status register, the software must write an all-ones 16-bit value except for the bit position to be cleared which is a zero.

The enable register is initialized by the software to indicate whether each interrupt request is enabled for the levels defined in the ICS.

**NOTE**

> In the case of multiple requests levels implementation in the same MIOS14, it is possible to enable interrupts at more than one different levels for the same submodule. It is the responsibility of the software to manage this.

Each bit in the IRQ pending register is the result of a logical “AND” between the corresponding bits in the status and in the enable registers. If a flag bit is set and the level enable bit is also set, then the IRQ pending bit is set, and the information is transferred to the interrupt control section that is in charge of sending the corresponding level to the CPU. The IRQ pending register is read only.

**NOTE**

> When the enable bit is not set for a particular submodule, the corresponding status register bit is still set when the corresponding flag is set. This allows the traditional software approach of polling the flag bits to see which ones are set. The status register makes flag polling easy, since up to 16 flag bits are contained in one register.

The submodule number of an interrupting source defines the corresponding MIRSM number and the bit position in the status registers. To find the MIRSM number and bit position of an interrupting source, proceed as follow:

1. Divide the interrupting submodule number by 16
2. The integer result of the division gives the MIRSM number
3. The reminder of the division gives the bit position

## 17.12.3 MIRSM0 Interrupt Registers

### 17.12.3.1 Interrupt Status Register (MIOS14SR0)

This register contains the flag bits that are raised when the submodules generate an interrupt. Each bit corresponds to a given submodule.

## 19.4.10 Channel Interrupt Status Register (CISR)

The channel interrupt status register (CISR) contains one interrupt status flag per channel. Time functions specify via microcode when an interrupt flag is set. Setting a flag causes the TPU3 to make an interrupt service request if the corresponding CIER bit is set. To clear a status flag, read CISR, then write a zero to the appropriate bit.

**NOTE**

CISR is the only TPU3 register that can be accessed on a byte basis.

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MSB | | | | | | | | | | | | | | | LSB |
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| Field | CH 15 | CH 14 | CH 13 | CH 12 | CH 11 | CH 10 | CH 9 | CH 8 | CH 7 | CH 6 | CH 5 | CH 4 | CH 3 | CH 2 | CH 1 | CH 0 |
| $\overline{\text{SRESET}}$ | 0000_0000_0000_0000 | | | | | | | | | | | | | | | |
| Addr | 0x30 4020 (TPU_A), 0x30 4420 (TPU_B) | | | | | | | | | | | | | | | |

**Figure 19-20. CISR — Channel Interrupt Status Register**

**Table 19-17. CISR Bit Descriptions**

| Bits | Name | Description |
|---|---|---|
| 0:15 | CH[15:0] | Channel interrupt status<br>0 Channel interrupt not asserted<br>1 Channel interrupt asserted |

## 19.4.11 TPU3 Module Configuration Register 2 (TPUMCR2)

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MSB | | | | | | | | | | | | | | | LSB |
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| Field | — | | | | | | | DIV2 | SOFTRST | ETBANK | | FPSCK | | | T2CF | DTPU |
| $\overline{\text{SRESET}}$ | 0000_0000_0000_0000 | | | | | | | | | | | | | | | |
| Addr | 0x30 4028 (TPU_A), 0x30 4428 (TPU_B) | | | | | | | | | | | | | | | |

**Figure 19-21. TPUMCR2 — TPU Module Configuration Register 2**

**Table 19-18. TPUMCR2 Bit Descriptions**

| Bits | Name | Description |
|---|---|---|
| 0:6 | — | Reserved |
| 7 | DIV2 | Divide by 2 control. When asserted, the DIV2 bit, along with the TCR1P bit and the PSCK bit in the TPUMCR, determines the rate of the TCR1 counter in the TPU3. If set, the TCR1 counter increments at a rate of two system clocks. If negated, TCR1 increments at the rate determined by control bits in the TCR1P and PSCK fields.<br>0 TCR1 increments at rate determined by control bits in the TCR1P and PSCK fields of the TPUMCR register<br>1 Causes TCR1 counter to increment at a rate of the system clock divided by two |

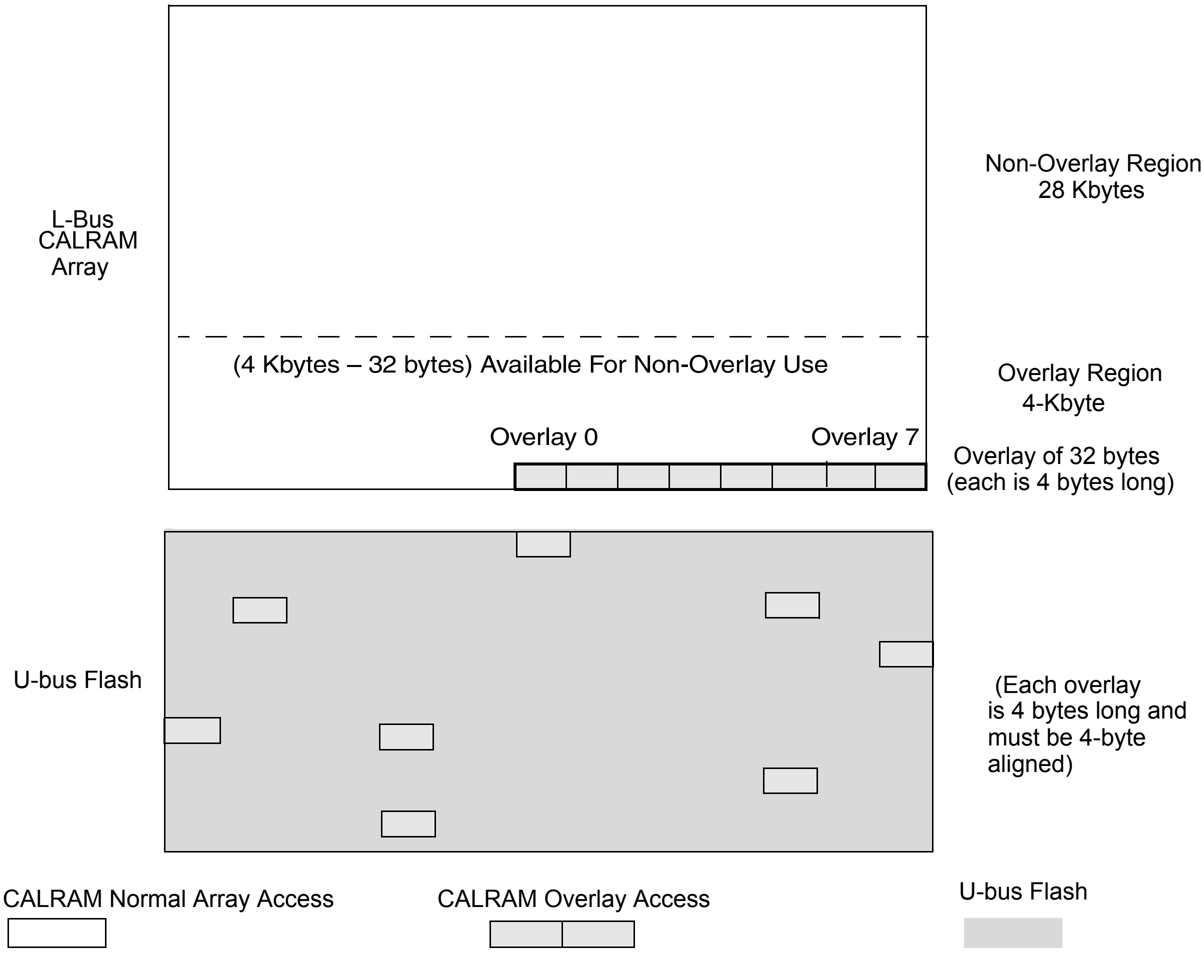


**Figure 22-7. CALRAM Module Overlay Map of Flash (CLPS = 1)**

Figure 22-8 shows the overlay regions when the CLPS bit is set for CALRAM in MPC561/MPC563.

| | MSB | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| Field | Ownership Trace Register | | | | | | | | | | | | | | | |
| $\overline{\text{SRESET}}$ | 0000_0000_0000_0000 | | | | | | | | | | | | | | | |
| Addr | 0x38 002C | | | | | | | | | | | | | | | |

| | | | | | | | | | | | | | | | | LSB |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Field | Ownership Trace Register | | | | | | | | | | | | | | | |
| $\overline{\text{SRESET}}$ | 0000_0000_0000_0000 | | | | | | | | | | | | | | | |

Figure 22-12. CALRAM Ownership Trace Register (CRAM_OTR)

economical interface (three pins) that allows the development system to operate in a lower frequency than the frequency of the CPU. Note that it is also possible to debug the CPU using monitor debugger software, for more information refer to Section 23.5, “Software Monitor Debugger Support.”

Debug mode is a state where the CPU fetches all instructions from the development port. In addition, when in debug mode, data can be read from the development port and written to the development port. This allows memory and registers to be read and modified by a development tool (emulator) connected to the development port.

For protection purposes, two possible working modes are defined: debug mode enable and debug mode disable. These working modes are selected only during reset. For more information refer to Section 23.3.1.1, “Debug Mode Enable vs. Debug Mode Disable.”

The user can work in debug mode starting from reset or the CPU can be programmed to enter debug mode as a result of a predefined list of events. These events include all possible interrupts and exceptions in the CPU system, including the internal breakpoints, together with two levels of development port requests (masked and non-masked) and one peripheral breakpoint request that can be generated by any one of the peripherals of the system (including internal and external modules). Each event can be programmed either to be treated as a regular interrupt that causes the machine to branch to its interrupt vector, or to be treated as a special interrupt that causes debug mode entry.

When in debug mode an rfi instruction will return the machine to its regular work mode. The debugger tool should issue an isync instruction to the debug port prior to any other instructions when the CPU enters debug mode after running code.

The relationship between the debug mode logic to the rest of the CPU chip is shown in Figure 23-5.

### 23.6.11 I-Bus Support Control Register (ICTRL)

| | MSB<br>0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | CTA | | | CTB | | | CTC | | | CTD | | | IWP0 | | IWP1 | |
| $\overline{\text{SRESET}}$ | 0000_0000_0000_0000 | | | | | | | | | | | | | | | |

| | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | LSB<br>31 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | IWP2 | | IWP3 | | SIWP0 EN | SIWP1 EN | SIWP2 EN | SIWP3 EN | DIWP0 EN | DIWP1 EN | DIWP2 EN | DIWP3 EN | IFM | ISCT_SER[1] | | |
| $\overline{\text{SRESET}}$ | 0000_0000_0000_0000 | | | | | | | | | | | | | | | |
| Addr | SPR 158 | | | | | | | | | | | | | | | |

**Figure 23-24. I-Bus Support Control Register (ICTRL)**

[1] Changing the instruction show cycle programming starts to take effect only from the second instruction after the actual mtspr to ICTRL.

If the processor aborts a fetch of the target of a direct branch (due to an exception), the target is not always visible on the external pins. Program trace is not affected by this phenomenon.

**Table 23-26. ICTRL Bit Descriptions**

| Bits | Mnemonic | Description | Function | |
|---|---|---|---|---|
| | | | Non-compressed mode[1] | Compressed Mode[2] |
| 0:2 | CTA | Compare type of comparator A | 0xx = not active (reset value)<br>100 = equal<br>101 = less than<br>110 = greater than<br>111 = not equal | 1xx = not active<br>000 = equal (reset value)<br>001 = less than<br>010 = greater than<br>011 = not equal |
| 3:5 | CTB | Compare type of comparator B | | |
| 6:8 | CTC | Compare type of comparator C | | |
| 9:11 | CTD | Compare type of comparator D | | |
| 12:13 | IWP0 | I-bus 1st watchpoint programming | 0x = not active (reset value)<br>10 = match from comparator A<br>11 = match from comparators (A&B) | |
| 14:15 | W1 | I-bus 2nd watchpoint programming | 0x = not active (reset value)<br>10 = match from comparator B<br>11 = match from comparators (A I B) | |
| 16:17 | IWP2 | I-bus 3rd watchpoint programming | 0x = not active (reset value)<br>10 = match from comparator C<br>11 = match from comparators (C&D) | |
| 18:19 | IWP3 | I-bus 4th watchpoint programming | 0x = not active (reset value)<br>10 = match from comparator D<br>11 = match from comparators (C I D)<br>0x = not active (reset value)<br>10 = match from comparator D<br>11 = match from comparators (C I D) | |

## NOTE

The MC register is not available prior to Revision D of the MPC561 and is not available in Revision B and earlier versions of the MPC563. Prior revisions have only the default features.

| | MSB<br>7 | 6 | 5 | 4 | 3 | 2 | 1 | LSB<br>0 |
|---|---|---|---|---|---|---|---|---|
| Field | QTST | — | | | | PTM | QFM | PTSM |
| $\overline{\text{RSTI}}$ | 0000_0000 | | | | | | | |
| Address | 0x0B | | | | | | | |

**Figure 24-5. READI Mode Control (MC) Register**

**Table 24-9. MC Bit Descriptions**

| RCPU Bits | Nexus Bits | Name | Description |
|---|---|---|---|
| 0 | 7 | QTST | Enables a factory test mode for structural testing of the queue. This bit can only be written in factory test mode. When set, no trace messages are queued.<br>Users should always write this bit as a 0. |
| 5 | 2 | PTM | The Program Trace Mode (PTM) bit enables an enhanced method of program trace. This mode allows program trace to work with the ISCTL bits of the ICTRL register set to any value except 3. The value of 2 is recommended for optimal processor performance. The drawback of this mode is direct branch messages are never syncronizing so sync requests must be held until the next indirect branch.<br>0 Legacy Program Trace Mode<br>1 Enhanced Program Trace Mode |
| 6 | 1 | QFM | The Queue Flush Mode (QFM) bit selects if information in the queue is discarded or transmitted at the time of an overrun. Discarding information allows trace to resume quicker after an overrun, but makes it difficult to find the cause of the overrun.<br>0 Information in the queue is removed<br>1 Trace is stopped until the queue empties. |
| 7 | 0 | PTSM | The Program Trace Sync Mode (PTSM) indicates if the program trace messages contain the I-CNT packet.<br>0 Program trace message do not contain the I-CNT packet.<br>1 Program trace message contain the I-CNT packet. |

### 24.6.1.6 User Base Address Register (UBA)

The UBA register defines the memory map address for the OT register. Table 24-10 gives a description of the register bits.

RCPU development access can be achieved either via the READI signals or the BDM signals on the MCU. The access method is determined during READI module configuration. Figure 24-78 shows how READI and BDM signals are multiplexed for RCPU development access.

When the READI module is configured for RCPU development access, IEEE-ISTO 5001 compliant vendor-defined messages are used for transmission of data in and out of the MCU.

### NOTE

On the MPC561/MPC563 the BDM signals are shared with the READI signals. Therefore BDM access is limited to access via the Nexus vendor-defined development support messages.

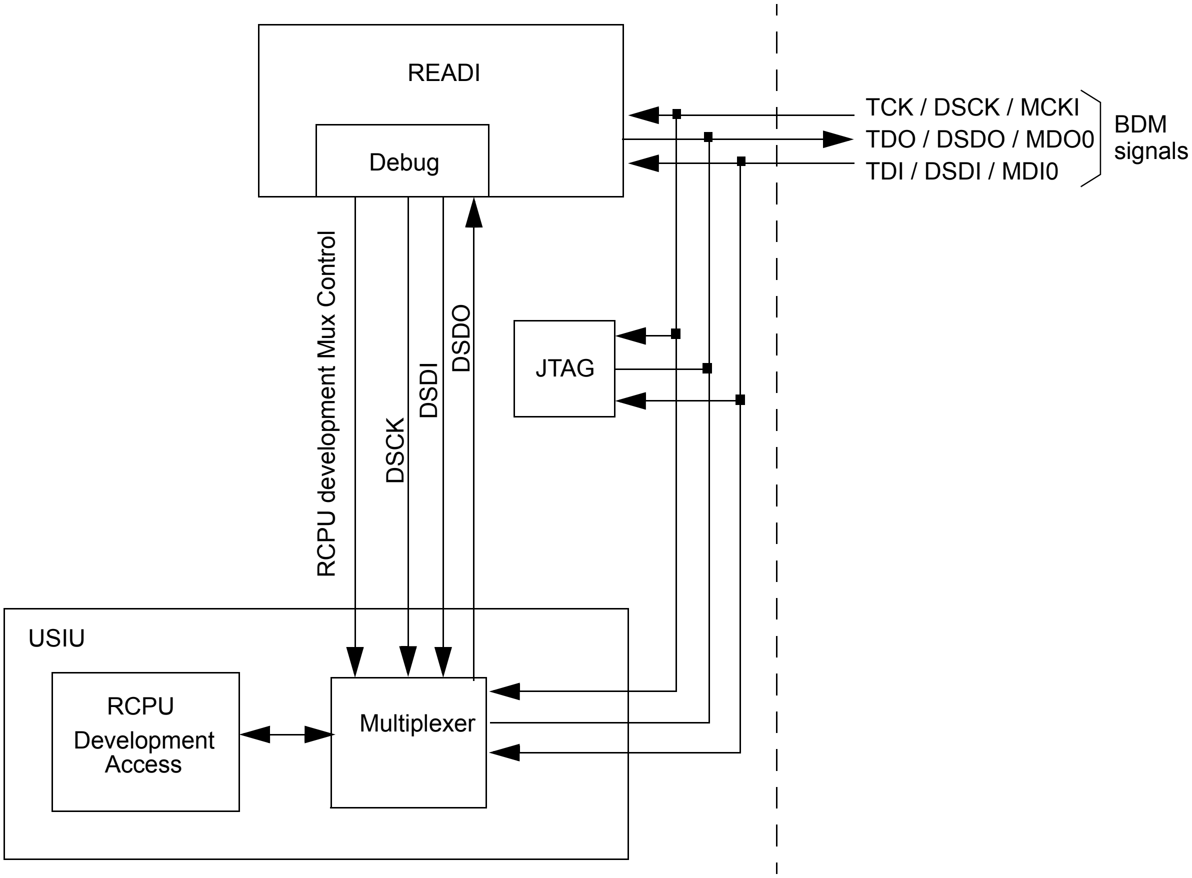


**Figure 24-78. RCPU Development Access Multiplexing between READI and BDM Signals**

## 24.14.1 RCPU Development Access Messaging

The following RCPU development access messages are used for handshaking between the device and the tool — DSDI data message, DSDO data message, and BDM status message.

### 24.14.1.1 DSDI Message

The DSDI message is used by the tool to download information to the RCPU.

### CONTROL BITS

| | NAME | OPTIONS | ADDRESSES |
|---|---|---|---|
| 0 1 2 3<br>▣▣▣▣ | Channel Function Select | xxxx – UART Function Number. Assigned during microcode assembly. See Table D-1 | 0x30YY0C – 0x30YY12 |
| 0 1<br>▣▣ | Host Sequence | 00 – No Parity<br>01 – No Parity<br>10 – Even Parity<br>11 – Odd Parity | 0x30YY14 – 0x30YY16 |
| 0 1<br>▣▣ | Host Service Request | 00 – Not Used<br>01 – Not Used<br>10 – Transmit<br>11 – Receive | 0x30YY18 – 0x30YY1A |
| 0 1<br>▣▣ | Channel Priority | 00 – Disabled<br>01 – Low Priority<br>10 – Medium Priority<br>11 – High Priority | 0x30YY1C – 0x30YY1E |
| 0<br>▣ | Channel Interrupt Enable | 0 – Channel Interrupt Disabled<br>1 – Channel Interrupt Enabled | 0x30YY0A |
| 0<br>□ | Channel Interrupt Status | 0 – Channel Interrupt Not Asserted<br>1 – Channel Interrupt Asserted | 0x30YY20 |

### PARAMETER RAM

| ADDRESS OFFSETS | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x30XXW0 | PARITY_TEMP | | | | | | | | | | | | | | | | Param 0 |
| 0x30XXW2 | MATCH_RATE | | | | | | | | | | | | | | | | Param 1 |
| 0x30XXW4 | PE | RE | TRANSMIT_DATA_REG | | | | | | | | | | | | | | Param 2 |
| 0x30XXW6 | DATA_SIZE | | | | | | | | | | | | | | | | Param 3 |
| 0x30XXW8 | ACTUAL_BIT_COUNT | | | | | | | | | | | | | | | | Param 4 |
| 0x30XXWA | SHIFT_REGISTER | | | | | | | | | | | | | | | | Param 5 |
| 0x30XXWC | ▓ | | | | | | | | | | | | | | | | Param 6 |
| 0x30XXWA | ▓ | | | | | | | | | | | | | | | | Param 7 |

▣= Written By RCPU

░= Written by RCPU and TPU

W = Channel Number

□= Written By TPU

▓= Unused Parameters

For address offsets: XX=41 for TPU_A, 45 for TPU_B
YY=40 for TPU_A,
44 for TPU_B
See Table 19-24 for the PRAM Address Offset Map.

**Figure D-8. UART Receiver Parameters**

**Table F-10. Bus Operation Timing (continued)**

**Note:** ($V_{DD}$ = 2.6 V ± 0.1 V, $V_{DDH}$ = 5.0 V ± 0.25 V, $T_A = T_L$ to $T_H$, 50 pF load unless noted otherwise)

| | Characteristic | 40 MHz | | 56 MHz[1] | | Unit |
|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | |
| 29 | $\overline{\text{TS}}$ valid to CLKOUT Rising Edge (Setup Time) | 7 | | 5 | | ns |
| 30 | CLKOUT Rising Edge to $\overline{\text{TS}}$ Valid (Hold Time). | 5 | | 5 | | ns |

[1] 56-MHz operation is available as an option. Some parts (without the 56-MHz option) will operate at a maximum frequency of 40 MHz.

[2] The timing for $\overline{\text{BR}}$ output is relevant when the MPC561/MPC563 is selected to work with external bus arbiter. The timing for $\overline{\text{BG}}$ output is relevant when the MPC561/MPC563 is selected to work with internal bus arbiter.

[3] The setup times required for $\overline{\text{TA}}$, $\overline{\text{TEA}}$, and $\overline{\text{BI}}$ are relevant only when they are supplied by the external device (and not the memory controller).

[4] The maximum value of spec 8 for DATA[0:31] pins must be extended by 1.1 ns if the pins have been precharged to greater than $V_{DDL}$. This is the case if an external slave device on the bus is running at the max. value of VDATAPC. This is currently specified at 3.1 V. The 1.1 ns addition to spec 8 reflects the expected timing degradation for 3.1 V.

[5] The timing 27 refers to $\overline{\text{CS}}$ when ACS = ‘00’ and to $\overline{\text{WE}}$[0:3]/$\overline{\text{BE}}$[0:3] when CSNT = ‘0’.

## NOTE

The D[0:31] input timings 17 and 18 refer to the rising edge of the CLKOUT in which the $\overline{\text{TA}}$ input signal is asserted.

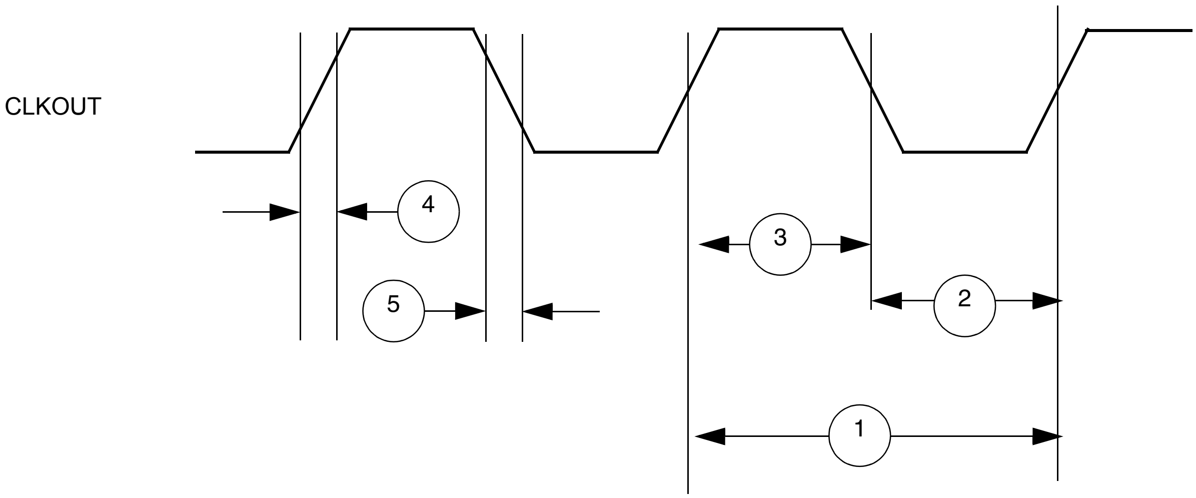


**Figure F-10. CLKOUT Pin Timing**