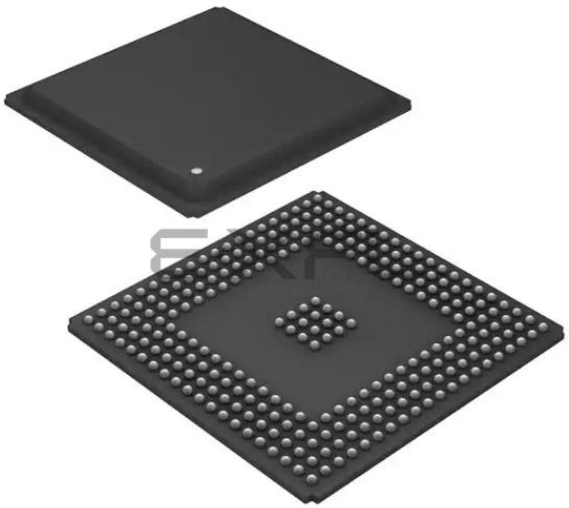


**Welcome to E-XFL.COM**

## Understanding Embedded - Microprocessors

Embedded microprocessors are specialized computing chips designed to perform specific tasks within an embedded system. Unlike general-purpose microprocessors found in personal computers, embedded microprocessors are tailored for dedicated functions within larger systems, offering optimized performance, efficiency, and reliability. These microprocessors are integral to the operation of countless electronic devices, providing the computational power necessary for controlling processes, handling data, and managing communications.

## Applications of Embedded - Microprocessors

Embedded microprocessors are utilized across a broad spectrum of applications, making them indispensable in

| Details | |
|---|---|
| Product Status | Not For New Designs |
| Core Processor | PowerPC G2_LE |
| Number of Cores/Bus Width | 1 Core, 32-Bit |
| Speed | 400MHz |
| Co-Processors/DSP | - |
| RAM Controllers | DDR, SDRAM |
| Graphics Acceleration | No |
| Display & Interface Controllers | - |
| Ethernet | 10/100Mbps (1) |
| SATA | - |
| USB | USB 1.1 (2) |
| Voltage - I/O | 2.5V, 3.3V |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Security Features | - |
| Package / Case | 272-BBGA |
| Supplier Device Package | 272-PBGA (27x27) |
| Purchase URL | https://www.e-xfl.com/product-detail/nxp-semiconductors/mpc5200cvr400 |

  - CODEC interface for Soft Modem, Master/Slave CODEC Mode, $I^2S$ and AC97
  - Full duplex SPI mode
  - IrDA mode from 2400 bps to 4 Mbps
- Fast Ethernet Controller (FEC)
  - Supports 100Mbps IEEE 802.3 MII, 10 Mbps IEEE 802.3 MII, 10 Mbps 7-wire interface
- Universal Serial Bus Controller (USB)
  - USB Revision 1.1 Host
  - Open Host Controller Interface (OHCI)
  - Integrated USB Hub, with two ports.
- Two Inter-Integrated Circuit Interfaces ($I^2C$)
- Serial Peripheral Interface (SPI)
- Dual CAN 2.0 A/B Controller (MSCAN)
  - —Freescale Scalable Controller Area Network (FSCAN) architecture
  - Implementation of version 2.0A/B CAN protocol
  - Standard and extended data frames
- J1850 Byte Data Link Controller (BDLC)
  - J1850 Class B data communication network interface compatible and ISO compatible for low speed (<125 kbps) serial data communications in automotive applications.
  - Supports 4X mode, 41.6 kbps
  - In-frame response (IFR) types 0, 1, 2, and 3 supported
- Systems level features
  - Interrupt Controller supports four external interrupt request lines and 47 internal interrupt sources
  - GPIO/Timer functions
    - Up to 56 total GPIO pins (depending on functional multiplexing selections) that support a variety of interrupt/WakeUp capabilities.
    - Eight GPIO pins with timer capability supporting input capture, output compare, and pulse width modulation (PWM) functions
  - Real-time Clock with one-second resolution
  - Systems Protection (watch dog timer, bus monitor)
  - Individual control of functional block clock sources
  - Power management: Nap, Doze, Sleep, Deep Sleep modes
  - Support of WakeUp from low power modes by different sources (GPIO, RTC, CAN)

**NOTE**

Beware of changing the values on the pins of the reset configuration word after the deassertion of $\overline{\text{PORRESET}}$. This may cause problems because it may change the internal clock ratios and so extend the PLL locking process.

## 3.3.4 External Interrupts

The MPC5200 provides three different kinds of external interrupts:

- Four IRQ interrupts
- Eight GPIO interrupts with simple interrupt capability (not available in power-down mode)
- Eight WakeUp interrupts (special GPIO pins)

The propagation of these three kinds of interrupts to the core is shown in the following graphic:

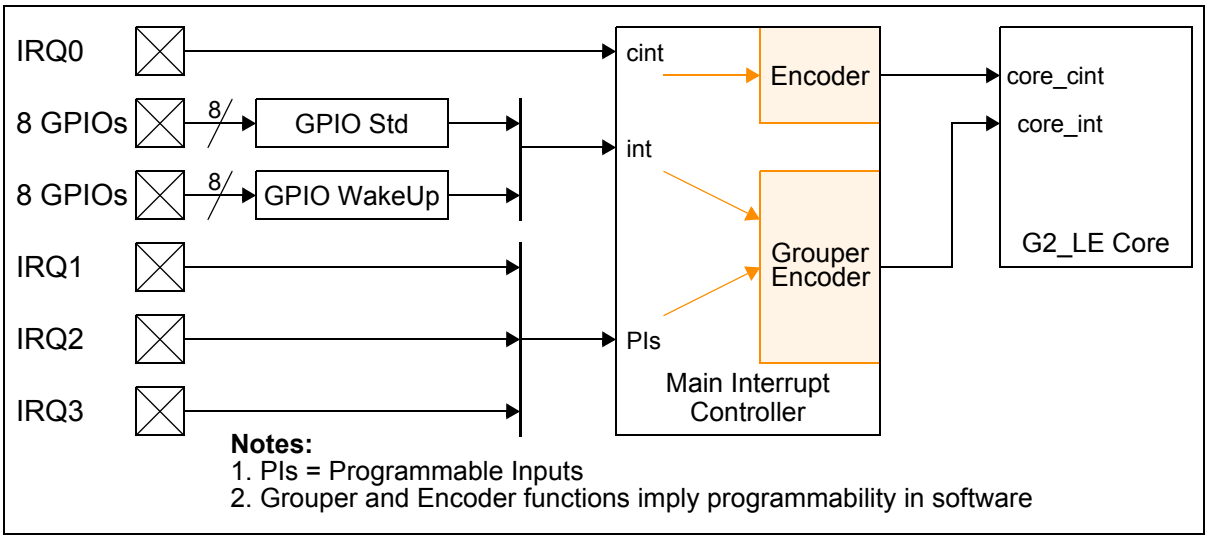



**Figure 4. External interrupt scheme**

Due to synchronization, prioritization, and mapping of external interrupt sources, the propagation of external interrupts to the core processor is delayed by several IP_CLK clock cycles. The following table specifies the interrupt latencies in IP_CLK cycles. The IP_CLK frequency is programmable in the Clock Distribution Module (see Note Table 16).

**Table 16. External interrupt latencies**

| Interrupt Type | Pin Name | Clock Cycles | Reference Clock | Core Interrupt | SpecID |
|---|---|---|---|---|---|
| Interrupt Requests | IRQ0 | 10 | IP_CLK | critical (cint) | A4.1 |
| | IRQ0 | 10 | IP_CLK | normal (int) | A4.2 |
| | IRQ1 | 10 | IP_CLK | normal (int) | A4.3 |
| | IRQ2 | 10 | IP_CLK | normal (int) | A4.5 |
| | IRQ3 | 10 | IP_CLK | normal (int) | A4.6 |

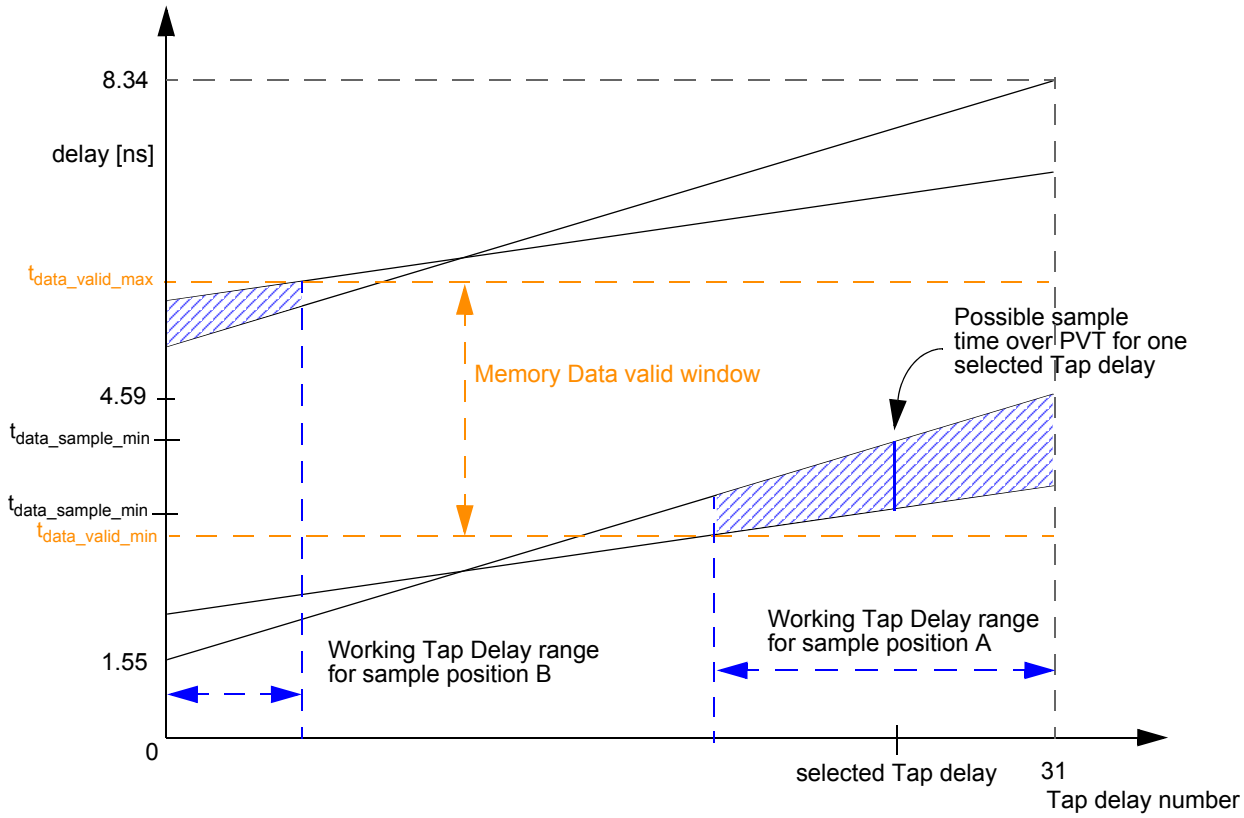



**Figure 8. Read Data sample window depend on the number of Tap delay**

The position of the $t_{data\_valid}$ window is depend on the clock / data flight time on the board. The MDQS signal indicate if the read data are valid. If the controller is not able to detect a valid MDQS signal on the sample time (sample position A) then the controller will look for valid MDQS / data on the next edge of the MEM_CLK signal (sample position B). Depend on the board travel time, different working tap delay configurations are possible. For a fast connection the data will be sampled with the next edge of MEM_CLK, this shows Figure 8, sample position A. With a longer connection maybe two sample positions are possible. Figure 8 shows a example with two working sample position (A and B). With a bigger board delay only sample position B will be possible.

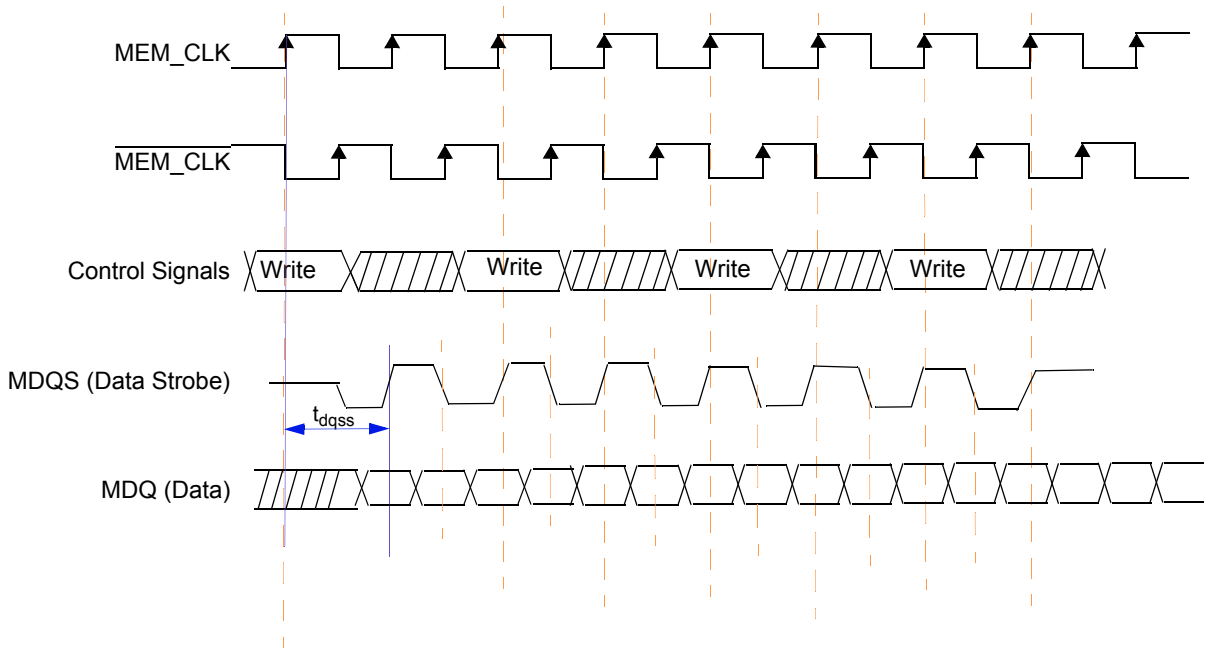
The equation below shows how to calculate the upper and lower limit. The right Tap delay number is selected, when the possible max and min sample timing is within the memory data valid window.

- $t_{data\_sample\_max}$ = max((1.55 + TapNum * 0.095), (1.74 + TapNum * 0.045))
- $t_{data\_sample\_min}$ = min((1.55 + TapNum * 0.095), (1.74 + TapNum * 0.045))

### 3.3.5.4 Memory Interface Timing-DDR SDRAM Write Command

**Table 21. DDR SDRAM Memory Write Timing**

| Sym | Description | Min | Max | Units | SpecID |
|---|---|---|---|---|---|
| $t_{mem\_clk}$ | MEM_CLK period | 7.5 | — | ns | A5.20 |
| $t_{DQSS}$ | Delay from write command to first rising edge of MDQS | — | $t_{mem\_clk}$+0.4 | ns | A5.21 |




NOTE: Control Signals signals are composed of RAS, CAS, $\overline{\text{MEM\_WE}}$, $\overline{\text{MEM\_CS}}$, $\overline{\text{MEM\_CS1}}$ and CLK_EN

**Figure 9. DDR SDRAM Memory Write Timing**

## 3.3.6 PCI

The PCI interface on the MPC5200 is designed to PCI Version 2.2 and supports 33-MHz and 66-MHz PCI operations. See the PCI Local Bus Specification [4]; the component section specifies the electrical and timing parameters for PCI components with the intent that components connect directly together whether on the planar or an expansion board, without any external buffers or other “glue logic.” Parameters apply at the package pins, not at expansion board edge connectors.

The MPC5200 is always the source of the PCI CLK. The clock waveform must be delivered to each 33-MHz or 66-MHz PCI component in the system. Figure 10 shows the clock waveform and required measurement points for 3.3 V signaling environments. Table 22 summarizes the clock specifications.

**Table 23. PCI Timing Parameters**

| Sym | Description | 66 MHz | | 33 MHz | | Units | Notes | SpecID |
|---|---|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | | | |
| $T_{val}$ | CLK to Signal Valid Delay - bused signals | 2 | 6 | 2 | 11 | ns | 1,2,3 | A6.5 |
| $T_{val}$(ptp) | CLK to Signal Valid Delay - point to point | 2 | 6 | 2 | 12 | ns | 1,2,3 | A6.6 |
| $T_{on}$ | Float to Active Delay | 2 | | 2 | | ns | 1 | A6.7 |
| $T_{off}$ | Active to Float Delay | | 14 | | 28 | ns | 1 | A6.8 |
| $T_{su}$ | Input Setup Time to CLK - bused signals | 3 | | 7 | | ns | 3,4 | A6.9 |
| $T_{su}$(ptp) | Input Setup Time to CLK - point to point | 5 | | 10,12 | | ns | 3,4 | A6.10 |
| $T_{h}$ | Input Hold Time from CLK | 0 | | 0 | | ns | 4 | A6.11 |

NOTES:
1. See the timing measurement conditions in the PCI Local Bus Specification [4]. It is important that all driven signal transitions drive to their Voh or Vol level within one Tcyc.
2. Minimum times are measured at the package pin with the load circuit, and maximum times are measured with the load circuit as shown in the PCI Local Bus Specification [4].
3. REQ# and GNT# are point-to-point signals and have different input setup times than do bused signals. GNT# and REQ# have a setup of 5 ns at 66 MHz. All other signals are bused.
4. See the timing measurement conditions in the PCI Local Bus Specification [4].

For Measurement and Test Conditions, see the PCI Local Bus Specification [4].

### 3.3.7.2 Burst Mode

**Table 25. Burst Mode Timing**

| Sym | Description | Min | Max | Units | Notes | SpecID |
|---|---|---|---|---|---|---|
| $t_{CSA}$ | PCI CLK to CS assertion | - | 1.8 | ns | | A7.20 |
| $t_{CSN}$ | PCI CLK to CS negation | - | 1.8 | ns | | A7.21 |
| $t_1$ | CS pulse width | $(1+WS+4^{LB}*2*(32/DS))*t_{PCIck}$ | $(1+WS+4^{LB}*2*(32/DS))*t_{PCIck}$ | ns | 1,2 | A7.22 |
| $t_2$ | ADDR valid before CS assertion | $t_{IPBIck}$ | $t_{PCIck}$ | ns | | A7.23 |
| $t_3$ | ADDR hold after CS negation | - | -0.7 | ns | | A7.24 |
| $t_4$ | OE assertion before CS assertion | - | 0.4 | ns | | A7.25 |
| $t_5$ | OE negation before CS negation | - | 0.4 | ns | | A7.26 |
| $t_6$ | RW valid before CS assertion | $t_{PCIck}$ | - | ns | | A7.27 |
| $t_7$ | RW hold after CS negation | $t_{PCIck}$ | - | ns | | A7.28 |
| $t_8$ | DATA setup before rising edge of PCI | 1.8 | - | ns | | A7.29 |
| $t_9$ | DATA hold after rising edge of PCI | 0 | - | ns | | A7.30 |
| $t_{10}$ | DATA hold after CS negation | 0 | $(DC+1)*t_{PCIck}$ | ns | | A7.31 |
| $t_{11}$ | ACK assertion after CS assertion | - | $(WS+1)*t_{PCIck}$ | ns | | A7.32 |
| $t_{12}$ | ACK negation before CS negation | - | 0.6 | ns | 3 | A7.33 |
| $t_{13}$ | ACK pulse width | $4^{LB}*2*(32/DS)*t_{PCIck}$ | $4^{LB}*2*(32/DS)*t_{PCIck}$ | ns | 2,3 | A7.34 |
| $t_{14}$ | CS assertion after TS assertion | - | 0.8 | ns | | A7.35 |
| $t_{15}$ | TS pulse width | $t_{PCIck}$ | $t_{PCIck}$ | ns | | A7.36 |

NOTES:
1. Wait States (WS) can be programmed in the Chip Select X Register, Bit field WaitP and WaitX. It can be specified 0 - 65535.
2. Example:
   Long Burst is used, this means the CS related BERx and SLB bits of the Chip Select Burst Control Register are set and a burst on the internal XLB is executed. => LB = 1
   Data bus width is 8 bit. => DS = 8
   => $4^1*2*(32/8) = 32$ => ACK is asserted for 32 PCI cycles to transfer one cache line.
   Wait State is set to 10. => WS = 10
   1+10+32 = 43 => CS is asserted for 43 PCI cycles.
3. ACK is output and indicates the burst.

### 3.3.7.3 MUXed Mode

**Table 26. MUXed Mode Timing**

| Sym | Description | Min | Max | Units | Notes | SpecID |
|---|---|---|---|---|---|---|
| $t_{CSA}$ | PCI CLK to CS assertion | - | 1.8 | ns | | A7.15 |
| $t_{CSN}$ | PCI CLK to CS negation | - | 1.8 | ns | | A7.16 |
| $t_{ALEA}$ | PCI CLK to ALE assertion | - | 1 | ns | | A7.16 |
| $t_1$ | ALE assertion before Address, Bank, TSIZ assertion | - | 0.8 | ns | | A7.17 |
| $t_2$ | CS assertion before Address, Bank, TSIZ negation | - | 0.7 | ns | | A7.18 |
| $t_3$ | CS assertion before Data wr valid | - | 0.7 | ns | | A7.19 |
| $t_4$ | Data wr hold after CS negation | $t_{IPBIck}$ | - | ns | | A7.20 |
| $t_5$ | Data rd setup before CS negation | 2.8 | - | ns | | A7.21 |
| $t_6$ | Data rd hold after CS negation | 0 | (DC+1)*$t_{PCIck}$ | ns | 1 | A7.22 |
| $t_7$ | ALE pulse width | - | $t_{PCIck}$ | ns | | A7.23 |
| $t_{TSA}$ | CS assertion after TS assertion | - | 0.8 | ns | | A7.24 |
| $t_8$ | TS pulse width | - | $t_{PCIck}$ | ns | | A7.24 |
| $t_9$ | CS pulse width | (2+WS)*$t_{PCIck}$ | (2+WS)*$t_{PCIck}$ | ns | | A7.25 |
| $t_{OEA}$ | OE assertion before CS assertion | - | 0.4 | ns | | A7.26 |
| $t_{OEN}$ | OE negation before CS negation | - | 0.4 | ns | | A7.27 |
| $t_{10}$ | RW assertion before ALE assertion | $t_{IPBIck}$ | - | ns | | A7.26 |
| $t_{11}$ | RW negation after CS negation | - | $t_{PCIck}$ | ns | | A7.27 |
| $t_{12}$ | ACK assertion after CS assertion | $t_{IPBIck}$ | - | ns | 2 | A7.28 |
| $t_{13}$ | ACK negation after CS negation | - | $t_{PCIck}$ | ns | 2 | A7.28 |

Note:
1. ACK can shorten the CS pulse width.
   Wait States (WS) can be programmed in the Chip Select X Register, Bit field WaitP and WaitX. It can be specified 0 - 65535.
2. ACK is input and can be used to shorten the CS pulse width.

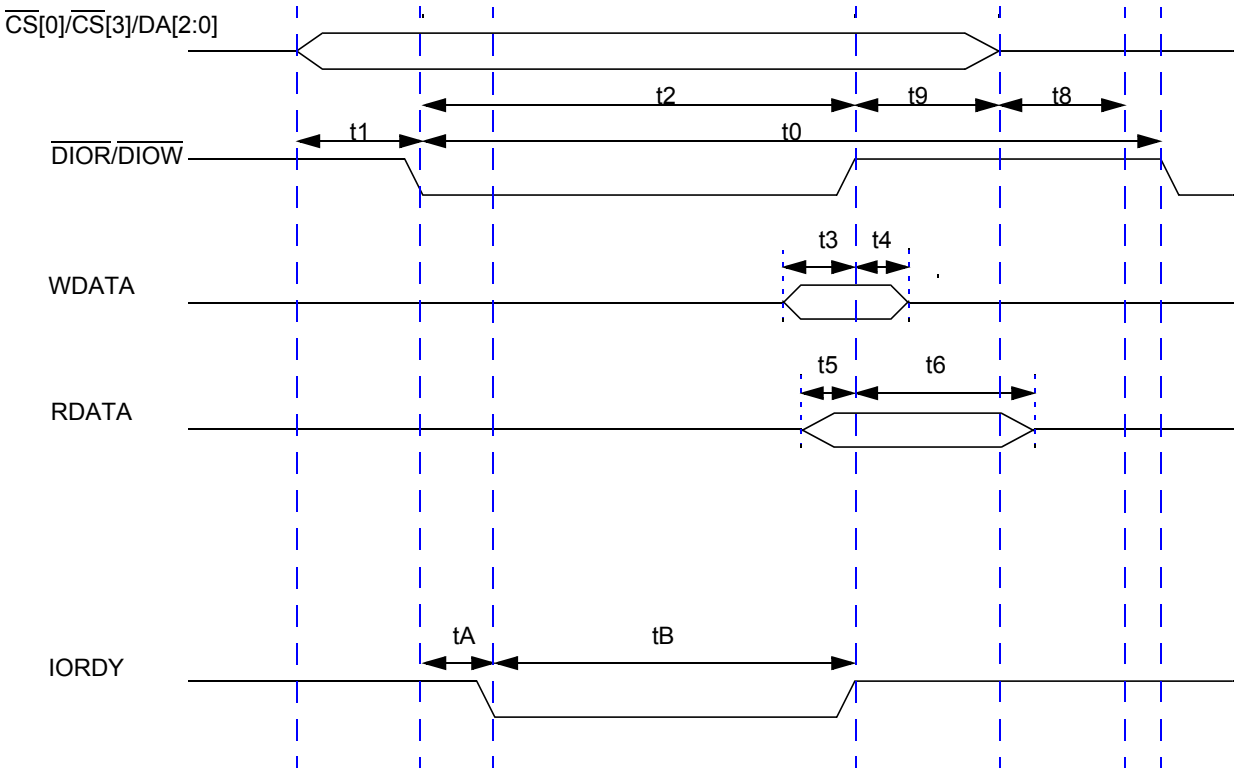



**Figure 15. PIO Mode Timing**

**Table 28. Multiword DMA Timing Specifications**

| | Multiword DMA Timing Parameters | Min/Max | Mode 0(ns) | Mode 1(ns) | Mode 2(ns) | SpecID |
|---|---|---|---|---|---|---|
| t0 | Cycle Time | min | 480 | 150 | 120 | A8.12 |
| tC | $\overline{\text{DMACK}}$ to DMARQ delay | max | — | — | — | A8.13 |
| tD | $\overline{\text{DIOR}}$/$\overline{\text{DIOW}}$ pulse width (16-bit) | min | 215 | 80 | 70 | A8.14 |
| tE | $\overline{\text{DIOR}}$ data access | max | 150 | 60 | 50 | A8.15 |
| tG | $\overline{\text{DIOR}}$/$\overline{\text{DIOW}}$ data setup | min | 100 | 30 | 20 | A8.16 |
| tF | $\overline{\text{DIOR}}$ data hold | min | 5 | 5 | 5 | A8.17 |
| tH | $\overline{\text{DIOW}}$ data hold | min | 20 | 15 | 10 | A8.18 |
| tI | DMACK to $\overline{\text{DIOR}}$/$\overline{\text{DIOW}}$ setup | min | 0 | 0 | 0 | A8.19 |
| tJ | $\overline{\text{DIOR}}$/$\overline{\text{DIOW}}$ to $\overline{\text{DMACK}}$ hold | min | 20 | 5 | 5 | A8.20 |
| tKr | $\overline{\text{DIOR}}$ negated pulse width | min | 50 | 50 | 25 | A8.21 |
| tKw | $\overline{\text{DIOW}}$ negated pulse width | min | 215 | 50 | 25 | A8.22 |
| tLr | $\overline{\text{DIOR}}$ to DMARQ delay | max | 120 | 40 | 35 | A8.23 |
| tLw | $\overline{\text{DIOW}}$ to DMARQ delay | max | 40 | 40 | 35 | A8.24 |

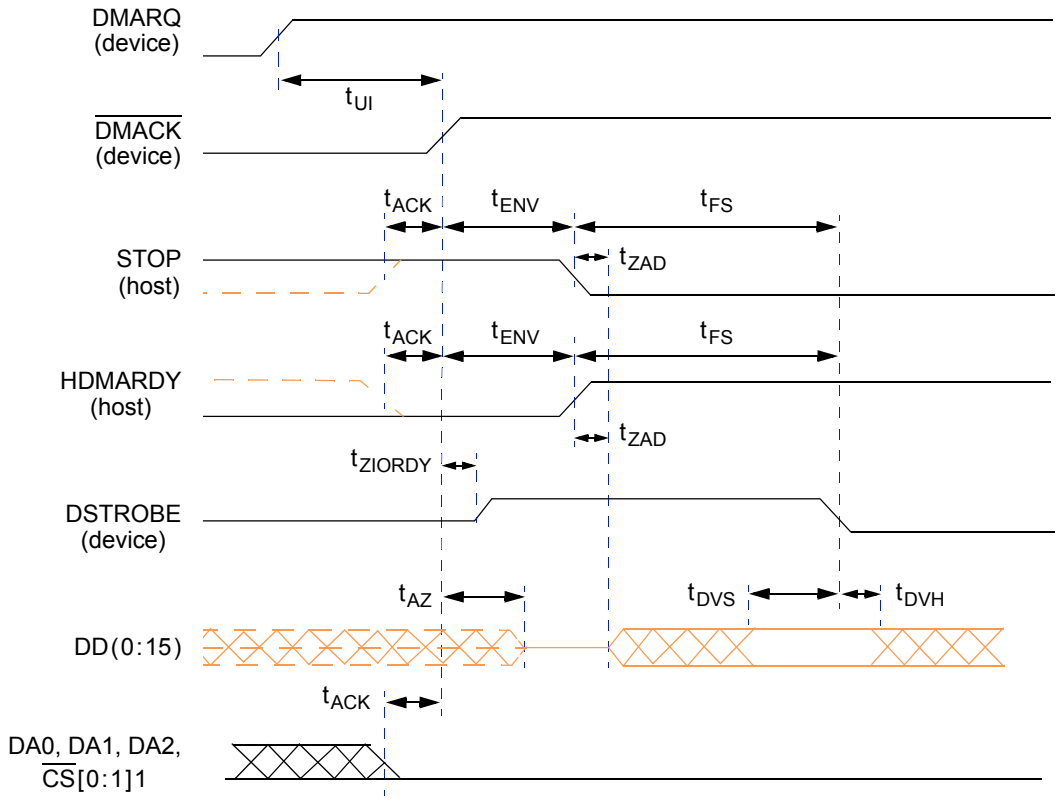



**Figure 17. Timing Diagram—Initiating an Ultra DMA Data In Burst**

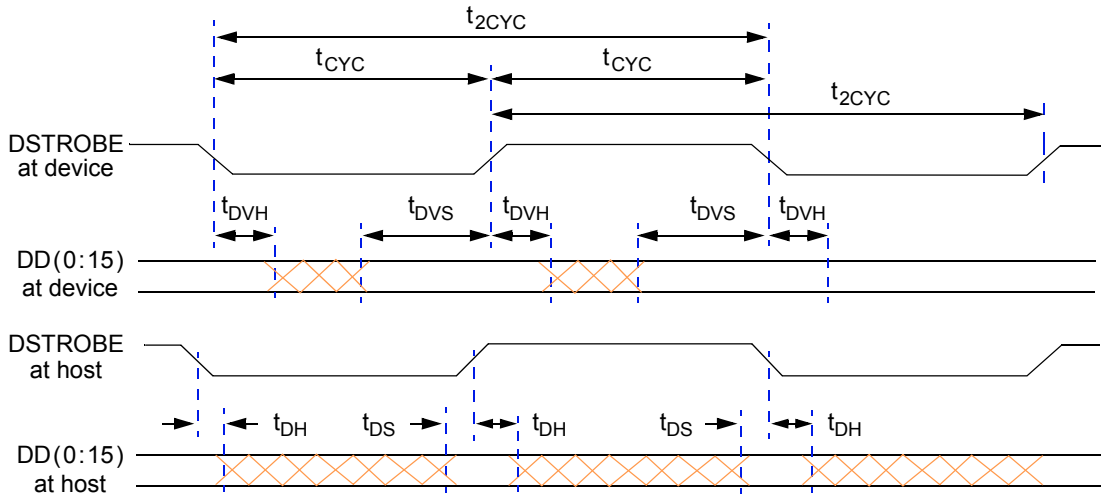



**Figure 18. Timing Diagram—Sustained Ultra DMA Data In Burst**

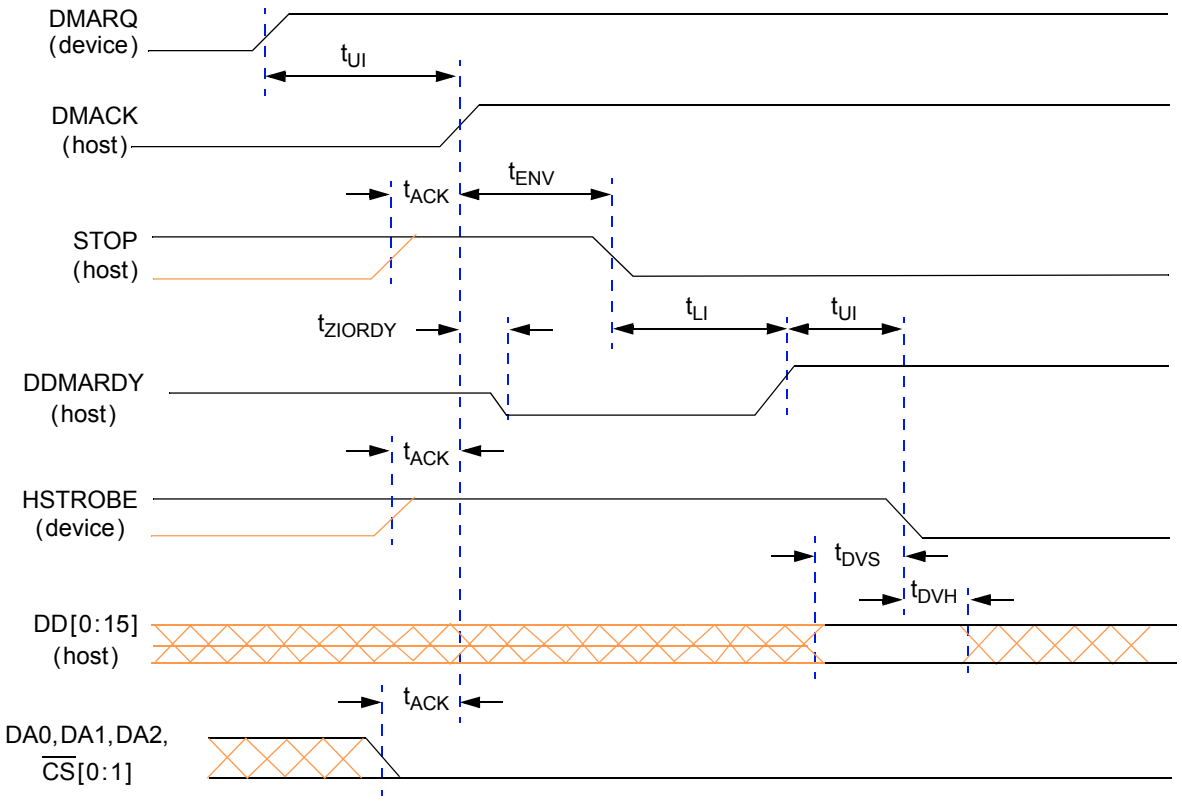



**Figure 22. Timing Diagram—Initiating an Ultra DMA Data Out Burst**

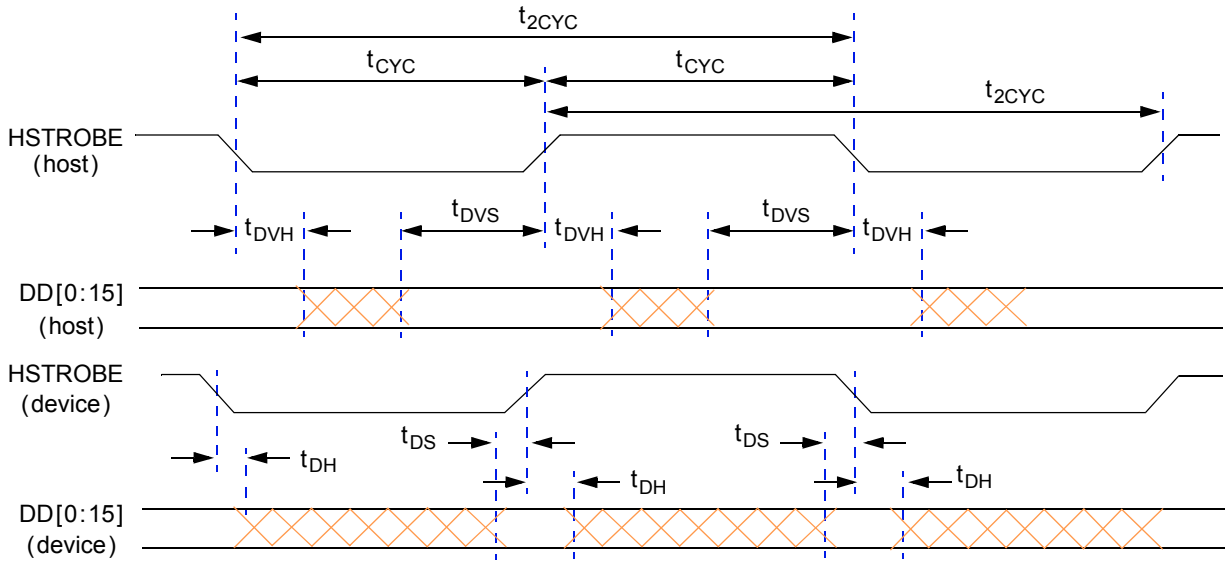



**Figure 23. Timing Diagram—Sustained Ultra DMA Data Out Burst**

## 3.3.9 Ethernet

### AC Test Timing Conditions:

- Output Loading
  All Outputs: 25 pF

**Table 31. MII Rx Signal Timing**

| Sym | Description | Min | Max | Unit | SpecID |
|---|---|---|---|---|---|
| M1 | RXD[3:0], RX_DV, RX_ER to RX_CLK setup | 10 | — | ns | A9.1 |
| M2 | RX_CLK to RXD[3:0], RX_DV, RX_ER hold | 10 | — | ns | A9.2 |
| M3 | RX_CLK pulse width high | 35% | 65% | RX_CLK Period[1] | A9.3 |
| M4 | RX_CLK pulse width low | 35% | 65% | RX_CLK Period[1] | A9.4 |

NOTES:
1 RX_CLK shall have a frequency of 25% of data rate of the received signal. See the IEEE 802.3 Specification [6].

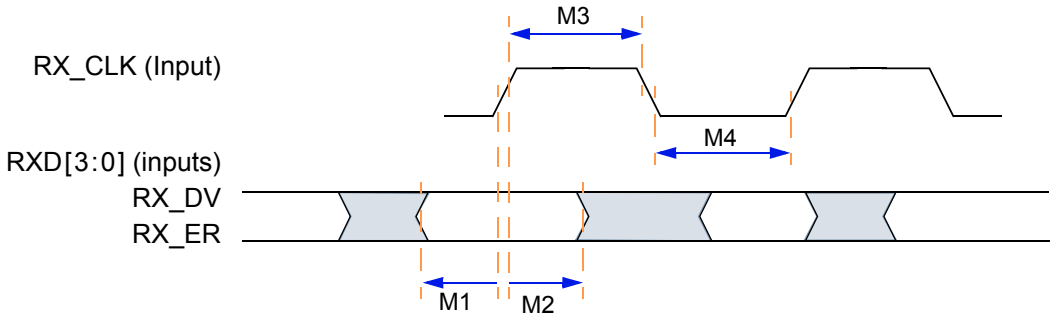



**Figure 28. Ethernet Timing Diagram—MII Rx Signal**

**Table 41. I²C Output Timing Specifications—SCL and SDA**

| Sym | Description | Min | Max | Units | SpecID |
|---|---|---|---|---|---|
| 1[1] | Start condition hold time | 6 | — | IP-Bus Cycle[3] | A13.8 |
| 2[1] | Clock low period | 10 | — | IP-Bus Cycle[3] | A13.9 |
| 3[2] | SCL/SDA rise time | — | 7.9 | ns | A13.10 |
| 4[1] | Data hold time | 7 | — | IP-Bus Cycle[3] | A13.11 |
| 5[1] | SCL/SDA fall time | — | 7.9 | ns | A13.12 |
| 6[1] | Clock high time | 10 | — | IP-Bus Cycle[3] | A13.13 |
| 7[1] | Data setup time | 2 | — | IP-Bus Cycle[3] | A13.14 |
| 8[1] | Start condition setup time (for repeated start condition only) | 20 | — | IP-Bus Cycle[3] | A13.15 |
| 9[1] | Stop condition setup time | 10 | — | IP-Bus Cycle[3] | A13.16 |

NOTES:
1. Programming IFDR with the maximum frequency (IFDR=0x20) results in the minimum output timings listed. The I²C interface is designed to scale the data transition time, moving it to the middle of the SCL low period. The actual position is affected by the prescale and division values programmed in IFDR.
2. Because SCL and SDA are open-drain-type outputs, which the processor can only actively drive low, the time SCL or SDA takes to reach a high level depends on external signal capacitance and pull-up resistor values
3. Inter Peripheral Clock is defined in the MPC5200 User Manual [1].

**NOTE**

Output timing was specified at a nominal 50 pF load.

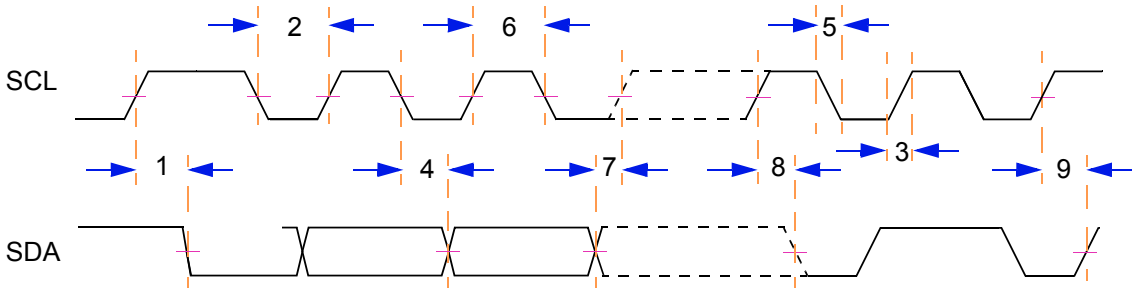



**Figure 37. Timing Diagram—I²C Input/Output**

## 3.3.14 J1850

See the MPC5200 User Manual [1].

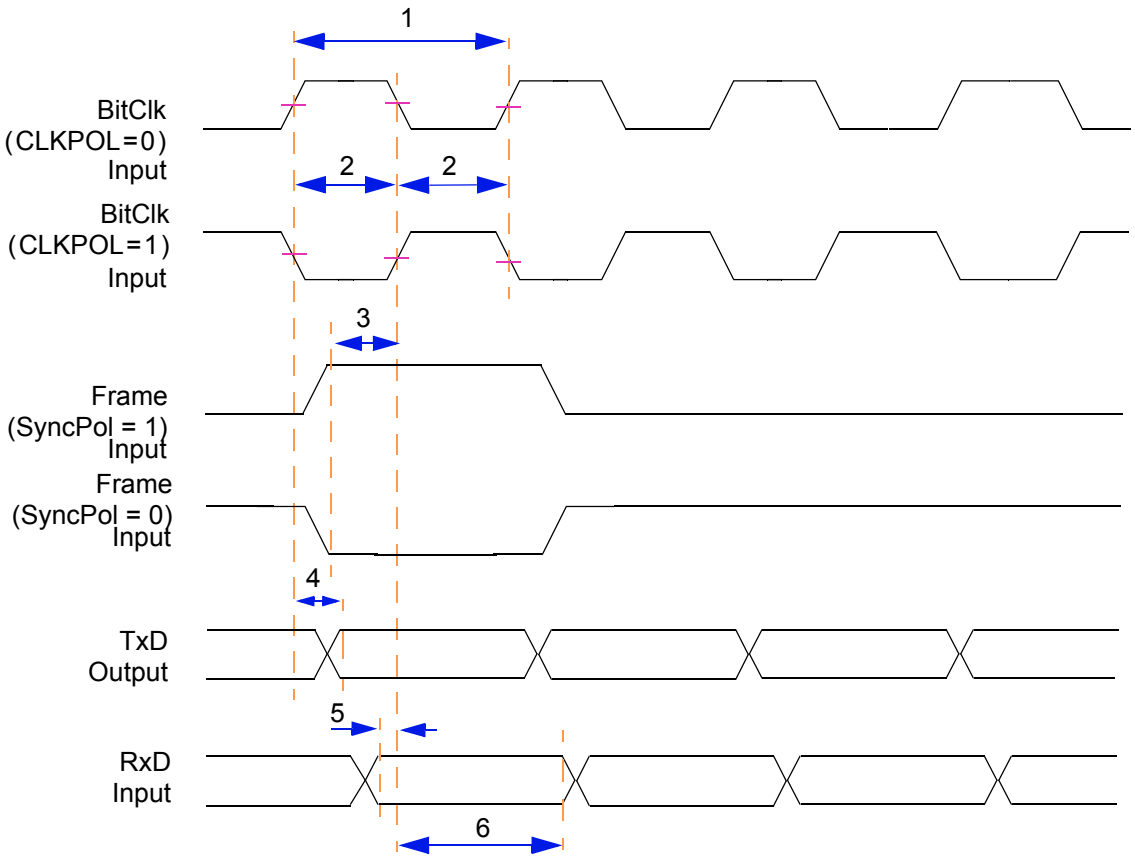



**Figure 39. Timing Diagram — 8,16, 24, and 32-bit CODEC / I²S Slave Mode**

### 3.3.15.2 AC97 Mode

**Table 44. Timing Specifications — AC97 Mode**

| Sym | Description | Min | Typ | Max | Units | SpecID |
|---|---|---|---|---|---|---|
| 1 | Bit Clock cycle time | — | 81.4 | — | ns | A15.15 |
| 2 | Clock pulse high time | — | 40.7 | — | ns | A15.16 |
| 3 | Clock pulse low time | — | 40.7 | — | ns | A15.17 |
| 4 | Frame valid after rising clock edge | — | — | 13.0 | ns | A15.18 |
| 5 | Output Data valid after rising clock edge | — | — | 14.0 | ns | A15.19 |
| 6 | Input Data setup time | 1.0 | — | — | ns | A15.20 |
| 7 | Input Data hold time | 1.0 | — | — | ns | A15.21 |

**NOTE**

Output timing was specified at a nominal 50 pF load.

### 3.3.15.4 SPI Mode

**Table 46. Timing Specifications — SPI Master Mode, Format 0 (CPHA = 0)**

| Sym | Description | Min | Max | Units | SpecID |
|---|---|---|---|---|---|
| 1 | SCK cycle time, programable in the PSC CCS register | 30.0 | — | ns | A15.26 |
| 2 | SCK pulse width, 50% SCK cycle time | 15.0 | — | ns | A15.27 |
| 3 | Slave select clock delay, programable in the PSC CCS register | 30.0 | — | ns | A15.28 |
| 4 | Output Data valid after Slave Select ($\overline{SS}$) | — | 8.9 | ns | A15.29 |
| 5 | Output Data valid after SCK | — | 8.9 | ns | A15.30 |
| 6 | Input Data setup time | 6.0 | — | ns | A15.31 |
| 7 | Input Data hold time | 1.0 | — | ns | A15.32 |
| 8 | Slave disable lag time | — | 8.9 | ns | A15.33 |
| 9 | Sequential Transfer delay, programable in the PSC CTUR / CTLR register | 15.0 | — | ns | A15.34 |
| 10 | Clock falling time | — | 7.9 | ns | A15.35 |
| 11 | Clock rising time | — | 7.9 | ns | A15.36 |

**NOTE**

Output timing was specified at a nominal 50 pF load.

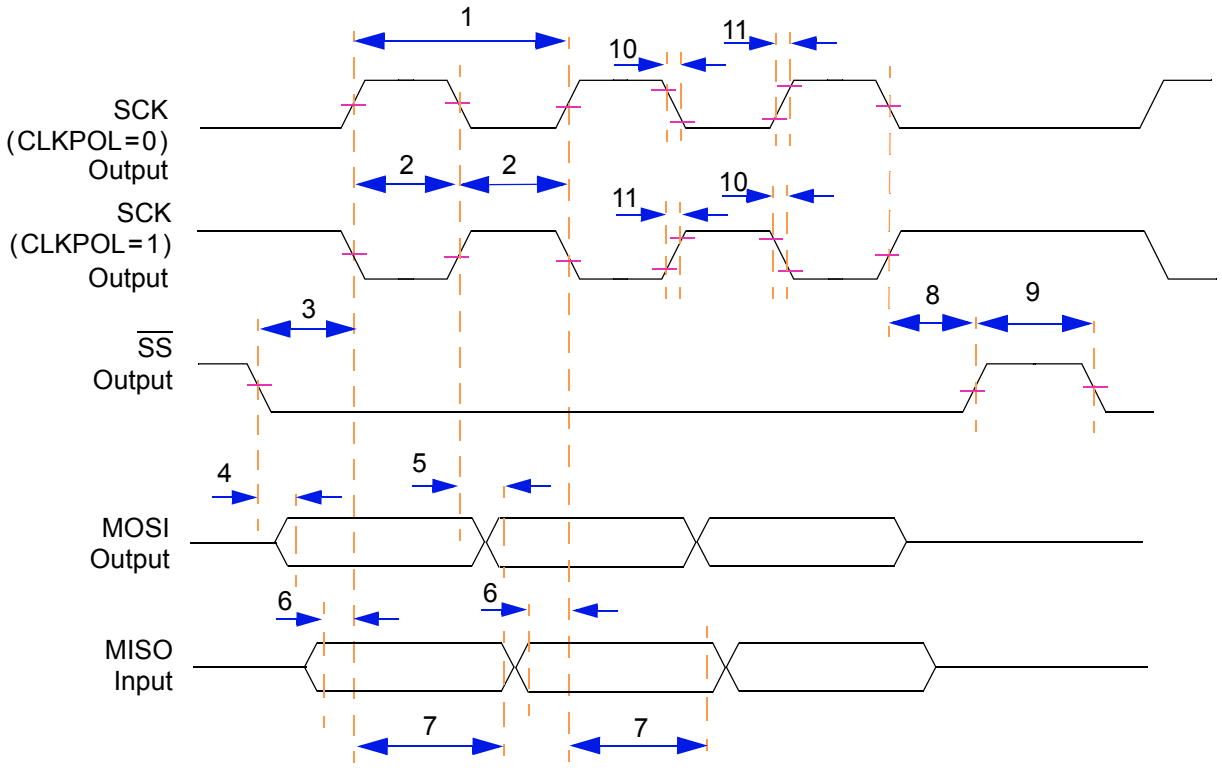



**Figure 42. Timing Diagram — SPI Master Mode, Format 0 (CPHA = 0)**

**Table 47. Timing Specifications — SPI Slave Mode, Format 0 (CPHA = 0)**

| Sym | Description | Min | Max | Units | SpecID |
|---|---|---|---|---|---|
| 1 | SCK cycle time, programable in the PSC CCS register | 30.0 | — | ns | A15.37 |
| 2 | SCK pulse width, 50% SCK cycle time | 15.0 | — | ns | A15.38 |
| 3 | Slave select clock delay | 1.0 | — | ns | A15.39 |
| 4 | Input Data setup time | 1.0 | — | ns | A15.40 |
| 5 | Input Data hold time | 1.0 | — | ns | A15.41 |
| 6 | Output data valid after $\overline{SS}$ | — | 14.0 | ns | A15.42 |
| 7 | Output data valid after SCK | — | 14.0 | ns | A15.43 |
| 8 | Slave disable lag time | 0.0 | — | ns | A15.44 |
| 9 | Minimum Sequential Transfer delay = 2 * IP Bus clock cycle time | 30.0 | — | — | A15.45 |

**NOTE**

Output timing was specified at a nominal 50 pF load.

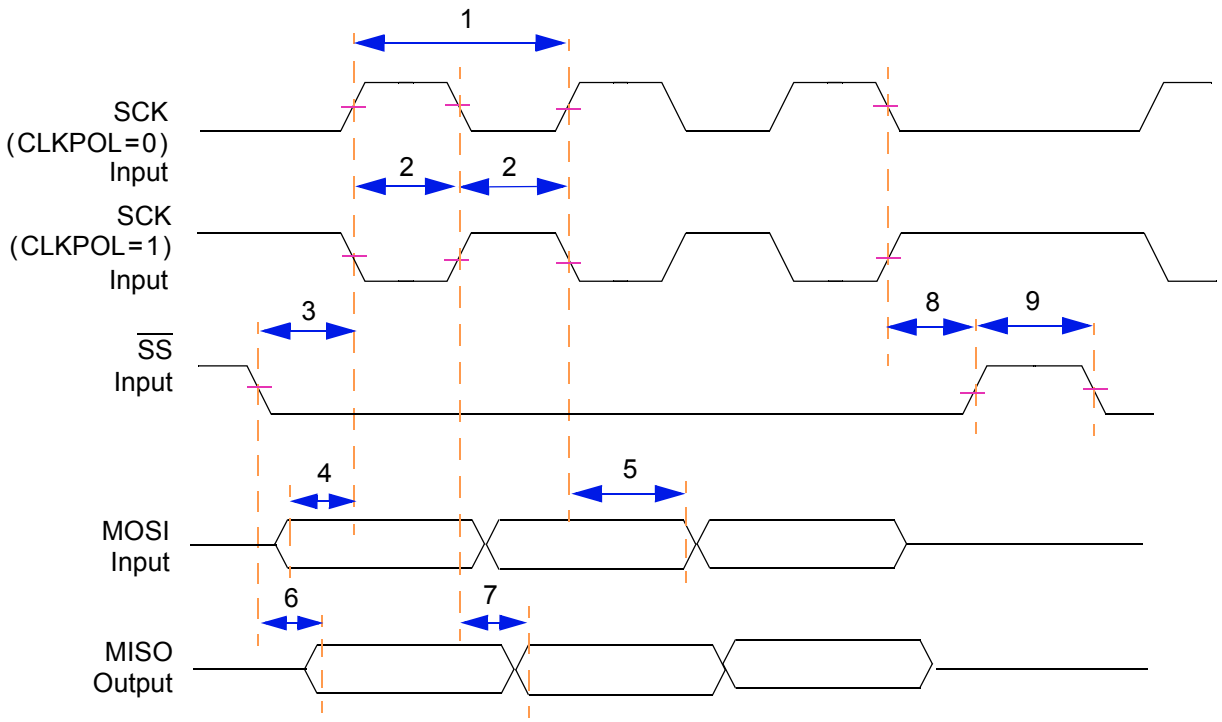



**Figure 43. Timing Diagram — SPI Slave Mode, Format 0 (CPHA = 0)**

**Table 49. Timing Specifications — SPI Slave Mode, Format 1 (CPHA = 1)**

| Sym | Description | Min | Max | Units | SpecID |
|---|---|---|---|---|---|
| 1 | SCK cycle time, programable in the PSC CCS register | 30.0 | — | ns | A15.56 |
| 2 | SCK pulse width, 50% SCK cycle time | 15.0 | — | ns | A15.57 |
| 3 | Slave select clock delay | 0.0 | — | ns | A15.58 |
| 4 | Output data valid | — | 14.0 | ns | A15.59 |
| 5 | Input Data setup time | 2.0 | — | ns | A15.60 |
| 6 | Input Data hold time | 1.0 | — | ns | A15.61 |
| 7 | Slave disable lag time | 0.0 | — | ns | A15.62 |
| 8 | Minimum Sequential Transfer delay = 2 * IP-Bus clock cycle time | 30.0 | — | ns | A15.63 |

**NOTE**

Output timing was specified at a nominal 50 pF load.

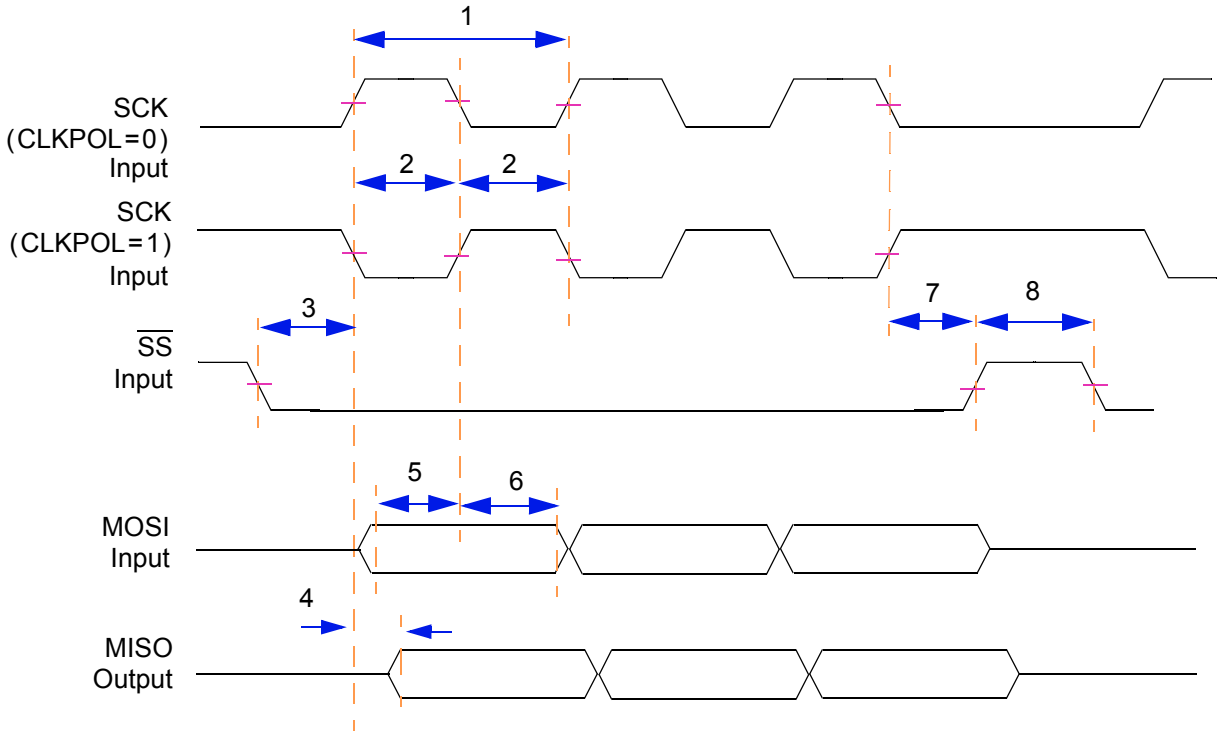



**Figure 45. Timing Diagram — SPI Slave Mode, Format 1 (CPHA = 1)**

## 4.3 Pinout Listings

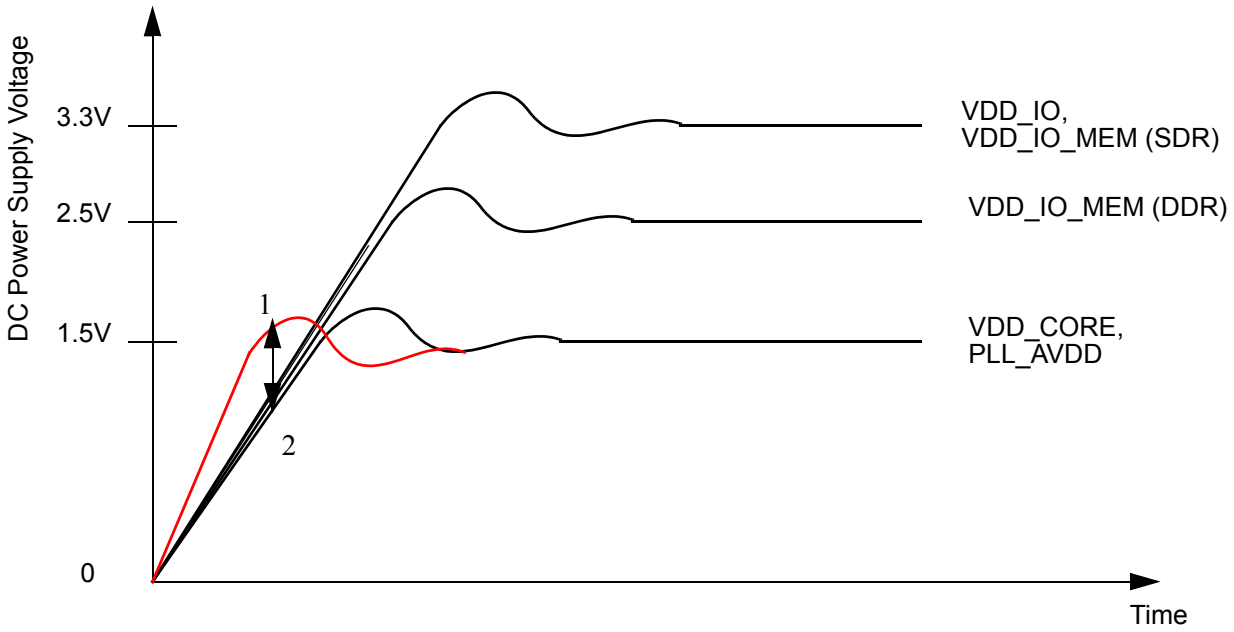
See details in the MPC5200 User Manual [1].

**Table 52. MPC5200 Pinout Listing**

| Name | Alias | Type | Power Supply | Output Driver Type | Input Type | Pull-up/ down |
|---|---|---|---|---|---|---|
| **SDRAM** | | | | | | |
| MEM_CAS | CAS | I/O | VDD_MEM_IO | DRV16_MEM | TTL | |
| MEM_CLK_EN | CLK_EN | I/O | VDD_MEM_IO | DRV16_MEM | TTL | |
| MEM_CS | | I/O | VDD_MEM_IO | DRV16_MEM | TTL | |
| MEM_DQM[3:0] | DQM | I/O | VDD_MEM_IO | DRV16_MEM | TTL | |
| MEM_MA[12:0] | MA | I/O | VDD_MEM_IO | DRV16_MEM | TTL | |
| MEM_MBA[1:0] | MBA | I/O | VDD_MEM_IO | DRV16_MEM | TTL | |
| MEM_MDQS[3:0] | MDQS | I/O | VDD_MEM_IO | DRV16_MEM | TTL | |
| MEM_MDQ[31:0] | MDQ | I/O | VDD_MEM_IO | DRV16_MEM | TTL | |
| MEM_CLK | | I/O | VDD_MEM_IO | DRV16_MEM | TTL | |
| MEM_CLK | | I/O | VDD_MEM_IO | DRV16_MEM | TTL | |
| MEM_RAS | RAS | I/O | VDD_MEM_IO | DRV16_MEM | TTL | |
| MEM_WE | | I/O | VDD_MEM_IO | DRV16_MEM | TTL | |
| **PCI** | | | | | | |
| EXT_AD[31:0] | | I/O | VDD_IO | PCI | PCI | |
| PCI_CBE_0 | | I/O | VDD_IO | PCI | PCI | |
| PCI_CBE_1 | | I/O | VDD_IO | PCI | PCI | |
| PCI_CBE_2 | | I/O | VDD_IO | PCI | PCI | |
| PCI_CBE_3 | | I/O | VDD_IO | PCI | PCI | |
| PCI_CLOCK | | I/O | VDD_IO | PCI | PCI | |
| PCI_DEVSEL | | I/O | VDD_IO | PCI | PCI | |
| PCI_FRAME | | I/O | VDD_IO | PCI | PCI | |
| PCI_GNT | | I/O | VDD_IO | DRV8 | TTL | |
| PCI_IDSEL | | I/O | VDD_IO | DRV8 | TTL | |
| PCI_IRDY | | I/O | VDD_IO | PCI | PCI | |
| PCI_PAR | | I/O | VDD_IO | PCI | PCI | |
| PCI_PERR | | I/O | VDD_IO | PCI | PCI | |
| PCI_REQ | | I/O | VDD_IO | DRV8 | TTL | |
| PCI_RESET | | I/O | VDD_IO | PCI | PCI | |
| PCI_SERR | | I/O | VDD_IO | PCI | PCI | |
| PCI_STOP | | I/O | VDD_IO | PCI | PCI | |

**Table 52. MPC5200 Pinout Listing (continued)**

| Name | Alias | Type | Power Supply | Output Driver Type | Input Type | Pull-up/ down |
|---|---|---|---|---|---|---|
| ETH_8 | RX_DV | I/O | VDD_IO | DRV4 | TTL | |
| ETH_9 | CD, RX_CLK | I/O | VDD_IO | DRV4 | Schmitt | |
| ETH_10 | CTS, COL | I/O | VDD_IO | DRV4 | TTL | |
| ETH_11 | TX_CLK | I/O | VDD_IO | DRV4 | Schmitt | |
| ETH_12 | RXD[0] | I/O | VDD_IO | DRV4 | TTL | |
| ETH_13 | USB_RXD, CTS, RXD[1] | I/O | VDD_IO | DRV4 | TTL | |
| ETH_14 | USB_RXP, UART_RX, RXD[2] | I/O | VDD_IO | DRV4 | TTL | |
| ETH_15 | USB_RXN, RX, RXD[3] | I/O | VDD_IO | DRV4 | TTL | |
| ETH_16 | USB_OVRCNT, CTS, RX_ER | I/O | VDD_IO | DRV4 | TTL | |
| ETH_17 | CD, CRS | I/O | VDD_IO | DRV4 | TTL | |
| **IRDA** | | | | | | |
| PSC6_0 | IRDA_RX, TxD | I/O | VDD_IO | DRV4 | TTL | |
| PSC6_1 | RxD | I/O | VDD_IO | DRV4 | TTL | |
| PSC6_2 | Frame, CTS | I/O | VDD_IO | DRV4 | TTL | |
| PSC6_3 | IR_USB_CLK,BitClk, RTS | I/O | VDD_IO | DRV4 | TTL | |
| **USB** | | | | | | |
| USB_0 | USB_OE | I/O | VDD_IO | DRV4 | TTL | |
| USB_1 | USB_TXN | I/O | VDD_IO | DRV4 | TTL | |
| USB_2 | USB_TXP | I/O | VDD_IO | DRV4 | TTL | |
| USB_3 | USB_RXD | I/O | VDD_IO | DRV4 | TTL | |
| USB_4 | USB_RXP | I/O | VDD_IO | DRV4 | TTL | |
| USB_5 | USB_RXN | I/O | VDD_IO | DRV4 | TTL | |
| USB_6 | USB_PRTPWR | I/O | VDD_IO | DRV4 | TTL | |
| USB_7 | USB_SPEED | I/O | VDD_IO | DRV4 | TTL | |
| USB_8 | USB_SUPEND | I/O | VDD_IO | DRV4 | TTL | |
| USB_9 | USB_OVRCNT | I/O | VDD_IO | DRV4 | TTL | |
| **$I^2C$** | | | | | | |
| I2C_0 | SCL | I/O | VDD_IO | DRV4 | Schmitt | |
| I2C_1 | SDA | I/O | VDD_IO | DRV4 | Schmitt | |
| I2C_2 | SCL | I/O | VDD_IO | DRV4 | Schmitt | |




Note:

1. VDD_CORE should not exceed VDD_IO, VDD_IO_MEM or PLL_AVDD by more than 0.4 V at any time, including power-up.
2. It is recommended that VDD_CORE/PLL_AVDD should track VDD_IO/VDD_IO_MEM up to 0.9 V then separate for completion of ramps.
3. Input voltage must not be greater than the supply voltage (VDD_IO, VDD_IO_MEM, VDD_CORE, or PLL_AVDD) by more than 0.5 V at any time, including during power-up.
4. Use 1 microsecond or slower rise time for all supplies.

**Figure 52. Supply Voltage Sequencing**

The relationship between VDD_IO_MEM and VDD_IO is non-critical during power-up and power-down sequences. Both VDD_IO_MEM (2.5 V or 3.3 V) and VDD_IO are specified relative to VDD_CORE.

### 5.1.1 Power Up Sequence

If VDD_IO/VDD_IO_MEM are powered up with the VDD_CORE at 0V, the sense circuits in the I/O pads will cause all pad output drivers connected to the VDD_IO/VDD_IO_MEM to be in a high-impedance state. There is no limit to how long after VDD_IO/VDD_IO_MEM powers up before VDD_CORE must power up. VDD_CORE should not lead the VDD_IO, VDD_IO_MEM or PLL_AVDD by more than 0.4 V during power ramp up or there will be high current in the internal ESD protection diodes. The rise times on the power supplies should be slower than 1 microsecond to avoid turning on the internal ESD protection clamp diodes.

The recommended power up sequence is as follows:

Use one microsecond or slower rise time for all supplies.

**Table 53. COP/BDM Interface Signals (continued)**

| BDM Pin # | MPC5200 I/O Pin | BDM Connector | Internal PullUp/Down | External PullUp/Down | I/O[1] |
|---|---|---|---|---|---|
| 8 | — | N/C | — | — | — |
| 7 | JTAG_TCK | tck | 100k Pull-Up | 10k Pull-Up | O |
| 6 | — | VDD[2] | — | — | — |
| 5 | See Note [3]. | halted[3] | — | — | I |
| 4 | $\overline{\text{JTAG\_TRST}}$ | trst | 100k Pull-Up | 10k Pull-Up | O |
| 3 | JTAG_TDI | tdi | 100k Pull-Up | 10k Pull-Up | O |
| 2 | See Note [4]. | qack[4] | — | — | O |
| 1 | JTAG_TDO | tdo | — | — | I |

NOTES:
1 With respect to the emulator tool's perspective:
Input is really an output from the embedded G2_LE core.
Output is really an input to the core.
2 From the board under test, power sense for chip power.
3 HALTED is not available from G2_LE core.
4 Input to the G2_LE core to enable/disable soft-stop condition during breakpoints. MPC5200 internal ties core_qack_ to GND in its normal/functional mode (always asserted).

For a board with a COP (common on-chip processor) connector, which accesses the JTAG interface and which needs to reset the JTAG module, simply wiring $\overline{\text{JTAG\_TRST}}$ and $\overline{\text{PORRESET}}$ is not recommended.

To reset the MPC5200 via the COP connector, the $\overline{\text{HRESET}}$ pin of the COP should be connected to the $\overline{\text{HRESET}}$ pin of the MPC5200. The circuitry shown in Figure 55 allows the COP to assert $\overline{\text{HRESET}}$ or $\overline{\text{JTAG\_TRST}}$ separately, while any other board sources can drive $\overline{\text{PORRESET}}$.