Zilog - EZ80L92AZ020EC00TR Datasheet

Welcome to E-XFL.COM

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	eZ80
Core Size	8-Bit
Speed	20MHz
Connectivity	EBI/EMI, I²C, IrDA, SPI, UART/USART
Peripherals	DMA, WDT
Number of I/O	24
Program Memory Size	-
Program Memory Type	ROMless
EEPROM Size	-
RAM Size	-
Voltage - Supply (Vcc/Vdd)	3V ~ 3.6V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	100-LQFP
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/ez80l92az020ec00tr

Table 1. 100-Pin LQFP Pin Identification of eZ80L92 MCU (Continued)

Pin No	Symbol	Function	Signal Direction	Description
43	V_{DD}	Power Supply		Power Supply.
44	V_{SS}	Ground		Ground.
45	IORQ	Input/Output Request	Bidirectional, Active Low	$\overline{IORQ}$ indicates that the CPU is accessing a location in I/O space. $\overline{RD}$ and $\overline{WR}$ indicate the type of access. The eZ80L92 MCU does not drive this line during RESET. It is an input in bus acknowledge cycles.
46	MREQ	Memory Request	Bidirectional, Active Low	$\overline{MREQ}$ Low indicates that the CPU is accessing a location in memory. The $\overline{RD}$, $\overline{WR}$, and $\overline{INSTRD}$ signals indicate the type of access. The eZ80L92 MCU does not drive this line during RESET. It is an input in bus acknowledge cycles.
47	RD	Read	Output, Active Low	$\overline{RD}$ Low indicates that the eZ80L92 MCU is reading from the current address location. This pin is tristated during bus acknowledge cycles.
48	WR	Write	Output, Active Low	$\overline{WR}$ indicates that the CPU is writing to the current address location. This pin is tristated during bus acknowledge cycles.
49	INSTRD	Instruction Read Indicator	Output, Active Low	$\overline{INSTRD}$ (with $\overline{MREQ}$ and $\overline{RD}$) indicates the eZ80L92 MCU is fetching an instruction from memory. This pin is tristated during bus acknowledge cycles.
50	WAIT	WAIT Request	Input, Active Low	Driving the $\overline{WAIT}$ pin Low forces the CPU to wait additional clock cycles for an external peripheral or external memory to complete its Read or Write operation.
51	RESET	Reset	Schmitt Trigger Input, Active Low	This signal is used to initialize the eZ80L92 MCU. This input must be Low for a minimum of 3 system clock cycles, and must be held Low until the clock is stable. This input includes a Schmitt trigger to allow RC rise times.

Table 1. 100-Pin LQFP Pin Identification of eZ80L92 MCU (Continued)

Pin No	Symbol	Function	Signal Direction	Description
63	TCK	JTAG Test Clock	Input	JTAG and ZDI clock input.
64	TRIGOUT	JTAG Test Trigger Output	Output	Active High trigger event indicator.
65	TDI	JTAG Test Data In	Bidirectional	JTAG data input pin. Functions as ZDI data I/O pin when JTAG is disabled.
66	TDO	JTAG Test Data Out	Output	JTAG data output pin.
67	V_{DD}	Power Supply		Power Supply.
68	PD0	GPIO Port D	Bidirectional	This pin can be used for GPIO. It can be individually programmed as input or output and can also be used individually as an interrupt input. Each Port D pin, when programmed as output, can be selected to be an open-drain or open-source output. Port D is multiplexed with one UART.
	TxD0	UART Transmit Data	Output	This pin is used by the UART to transmit asynchronous serial data. This signal is multiplexed with PD0.
	IR_TXD	IrDA Transmit Data	Output	This pin is used by the IrDA encoder/decoder to transmit serial data. This signal is multiplexed with PD0.
69	PD1	GPIO Port D	Bidirectional	This pin can be used for GPIO. It can be individually programmed as input or output and can also be used individually as an interrupt input. Each Port D pin, when programmed as output, can be selected to be an open-drain or open-source output. Port D is multiplexed with one UART.
	RxD0	Receive Data	Input	This pin is used by the UART to receive asynchronous serial data. This signal is multiplexed with PD1.
	IR_RXD	IrDA Receive Data	Input	This pin is used by the IrDA encoder/decoder to receive serial data. This signal is multiplexed with PD1.

Table 1. 100-Pin LQFP Pin Identification of eZ80L92 MCU (Continued)

Pin No	Symbol	Function	Signal Direction	Description
70	PD2	GPIO Port D	Bidirectional	This pin can be used for GPIO. It can be individually programmed as input or output and can also be used individually as an interrupt input. Each Port D pin, when programmed as output, can be selected to be an open-drain or open-source output. Port D is multiplexed with one UART.
	RTS0	Request to Send	Output, Active Low	Modem control signal from UART. This signal is multiplexed with PD2.
71	PD3	GPIO Port D	Bidirectional	This pin can be used for GPIO. It can be individually programmed as input or output and can also be used individually as an interrupt input. Each Port D pin, when programmed as output, can be selected to be an open-drain or open-source output. Port D is multiplexed with one UART.
	CTS0	Clear to Send	Input, Active Low	Modem status signal to the UART. This signal is multiplexed with PD3.
72	PD4	GPIO Port D	Bidirectional	This pin can be used for GPIO. It can be individually programmed as input or output and can also be used individually as an interrupt input. Each Port D pin, when programmed as output, can be selected to be an open-drain or open-source output. Port D is multiplexed with one UART.
	DTR0	Data Terminal Ready	Output, Active Low	Modem control signal to the UART. This signal is multiplexed with PD4.
73	PD5	GPIO Port D	Bidirectional	This pin can be used for GPIO. It can be individually programmed as input or output and can also be used individually as an interrupt input. Each Port D pin, when programmed as output, can be selected to be an open-drain or open-source output. Port D is multiplexed with one UART.
	DSR0	Data Set Ready	Input, Active Low	Modem status signal to the UART. This signal is multiplexed with PD5.

Table 1. 100-Pin LQFP Pin Identification of eZ80L92 MCU (Continued)

Pin No	Symbol	Function	Signal Direction	Description
74	PD6	GPIO Port D	Bidirectional	This pin can be used for GPIO. It can be individually programmed as input or output and can also be used individually as an interrupt input. Each Port D pin, when programmed as output, can be selected to be an open-drain or open-source output. Port D is multiplexed with one UART.
	DCD0	Data Carrier Detect	Input, Active Low	Modem status signal to the UART. This signal is multiplexed with PD6.
75	PD7	GPIO Port D	Bidirectional	This pin can be used for GPIO. It can be individually programmed as input or output and can also be used individually as an interrupt input. Each Port D pin, when programmed as output, can be selected to be an open-drain or open-source output. Port D is multiplexed with one UART.
	RI0	Ring Indicator	Input, Active Low	Modem status signal to the UART. This signal is multiplexed with PD7.
76	PC0	GPIO Port C	Bidirectional	This pin can be used for GPIO. It can be individually programmed as input or output and can also be used individually as an interrupt input. Each Port C pin, when programmed as output, can be selected to be an open-drain or open-source output. Port C is multiplexed with one UART.
	TxD1	Transmit Data	Output	This pin is used by the UART to transmit asynchronous serial data. This signal is multiplexed with PC0.

Reset

RESET Operation

The RESET controller within the eZ80L92 MCU provides a consistent system reset (RESET) function for all type of resets that may affect the system. Following four events can cause a RESET:

- External $\overline{\text{RESET}}$ pin assertion.
- Watchdog Timer (WDT) time-out when configured to generate a RESET.
- Real time clock alarm with eZ80 CPU in low-power SLEEP mode.
- Execution of a Debug RESET command.

During RESET, an internal RESET mode timer holds the system in RESET for 257 system clock (SCLK) cycles. The RESET mode timer begins incrementing on the next rising edge of SCLK following deactivation of all RESET events ($\overline{\text{RESET}}$ pin, WDT, real time clock, and Debugger)

Note: You must determine if 257 SCLK cycles provides sufficient time for the primary crystal oscillator to stabilize.

RESET, through the external $\overline{\text{RESET}}$ pin, must always be executed following application of power (V_{DD} ramp). Without the RESET, following power-up, proper operation of the eZ80L92 cannot be guaranteed.

Port x Data Direction Registers

In addition to the other GPIO Control Registers, the Port *x* Data Direction Registers (see Table 8) control the operating modes of the GPIO port pins. See Table 6.

Table 8. Port x Data Direction Registers (PB_DDR = 009Bh, PC_DDR = 009Fh, PD_DDR = 00A3h)

Bit	7	6	5	4	3	2	1	0
Reset	1	1	1	1	1	1	1	1
CPU Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Note: R/W = Read/Write.								

Port x Alternate Register 1

In addition to other GPIO Control Registers, the Port *x* Alternate Register 1 (see Table 9) control the operating modes of the GPIO port pins. See Table 6.

Table 9. Port x Alternate Registers 1 (PB_ALT1 = 009Ch, PC_ALT1 = 00A0h, PD_ALT1 = 00A4h)

Bit	7	6	5	4	3	2	1	0
Reset	0	0	0	0	0	0	0	0
CPU Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Note: R/W = Read/Write.								

Port x Alternate Register 2

In addition to other GPIO Control Registers, the Port *x* Alternate Register 2 (see Table 10) control the operating modes of the GPIO port pins. See Table 6.

Table 10. Port x Alternate Registers 2 (PB_ALT2 = 009Dh, PC_ALT2 = 00A1h, PD_ALT2 = 00A5h)

Bit	7	6	5	4	3	2	1	0
Reset	0	0	0	0	0	0	0	0
CPU Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Note: R/W = Read/Write.								

Memory Chip Select Priority

A lower-numbered Chip Select is given priority over a higher-numbered Chip Select. For example, if the address space of Chip Select 0 overlaps the Chip Select 1 address space, Chip Select 0 is active.

RESET States

On RESET, Chip Select 0 is active for all addresses, as its Lower Bound register resets to 00h and its Upper Bound register resets to FFh. All of the other Chip Select Lower and Upper Bound registers reset to 00h.

Memory Chip Select Example

The use of Memory Chip Selects is illustrated in Figure 4. The associated control register values are listed in Table 13. In this example, all 4 Chip Selects are enabled and configured for memory addresses. Also, CS1 overlaps with CS0. As CS0 has the higher than CS1, CS1 is not active for much of its defined address space.

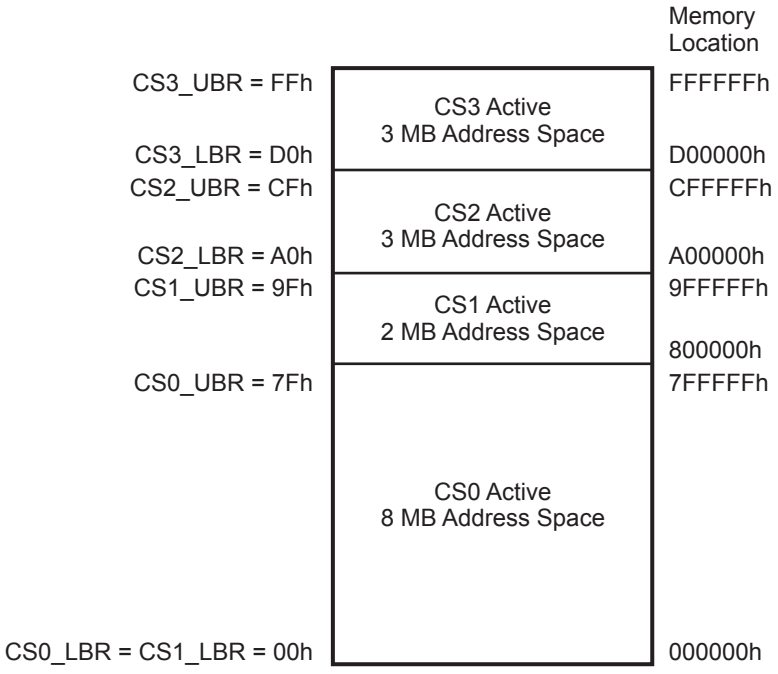

Figure 4. Memory Chip Select Example

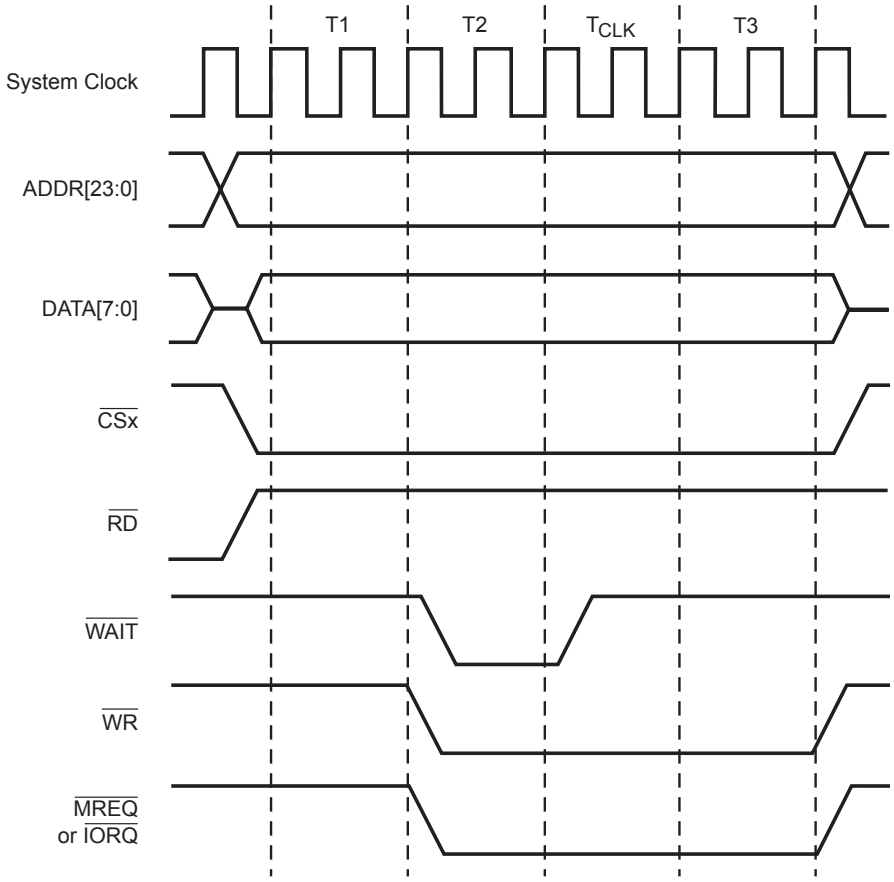

Figure 8. Z80® Bus Mode Write Timing Example

Intel Bus Mode

Chip selects configured for Intel Bus Mode modify the eZ80 bus signals to duplicate a four-state memory transfer similar to that found on Intel-style microprocessors. The bus signals and eZ80L92 pins are mapped as illustrated in Figure 9. In Intel Bus Mode, the user can select either multiplexed or non-multiplexed address and data buses. In non-multiplexed operation, the address and data buses are separate. In multiplexed operation, the lower byte of the address, ADDR[7:0], also appears on the data bus, DATA[7:0], during State T1 of the Intel Bus Mode cycle. During multiplexed operation, the lower byte of the address bus also appears on the address bus in addition to the data bus.

Table 16. Intel® Bus Mode Read States (Separate Address and Data Buses) (Continued)

STATE T3	During State T3, no bus signals are altered. If the external ReadY ($\overline{WAIT}$) pin is driven Low at least one eZ80 system clock cycle prior to the beginning of State T3, additional WAIT states (T_{WAIT}) are asserted until the ReadY pin is driven High.
STATE T4	The CPU latches the Read data at the beginning of State T4. The CPU deasserts the $\overline{RD}$ signal and completes the Intel Bus Mode cycle.

During Write operations with separate address and data buses, the Intel Bus Mode employs 4 states (T1, T2, T3, and T4) as described in Table 17.

Table 17. Intel® Bus Mode Write States (Separate Address and Data Buses)

STATE T1	The Write cycle begins in State T1. The CPU drives the address onto the address bus, the associated Chip Select signal is asserted, and the data is driven onto the data bus. The CPU drives the ALE signal High at the beginning of T1. During the middle of T1, the CPU drives ALE Low to facilitate the latching of the address.
STATE T2	During State T2, the CPU asserts the $\overline{WR}$ signal. Depending on the instruction, either the $\overline{MREQ}$ or $\overline{IORQ}$ signal is asserted.
STATE T3	During State T3, no bus signals are altered. If the external ReadY ($\overline{WAIT}$) pin is driven Low at least one eZ80 system clock cycle prior to the beginning of State T3, additional WAIT states (T_{WAIT}) are asserted until the ReadY pin is driven High.
STATE T4	The CPU deasserts the $\overline{WR}$ signal at the beginning of State T4. The CPU holds the data and address buses through the end of T4. The bus cycle is completed at the end of T4.

Intel Bus Mode timing is illustrated for a Read operation in Figure 10 and for a Write operation in Figure 11. If the ReadY signal (external $\overline{WAIT}$ pin) is driven Low prior to the beginning of State T3, additional WAIT states (T_{WAIT}) are asserted until the ReadY signal is driven High. The Intel Bus Mode states can be configured for 2 to 15 eZ80 system clock cycles. In the figures, each Intel® Bus Mode state is 2 eZ80 system clock cycles in duration. Figure 10 and Figure 11 also illustrate the assertion of one WAIT state (T_{WAIT}) by the selected peripheral.

Programmable Reload Timers

The eZ80L92 MCU features six Programmable Reload Timers (PRT). Each PRT contains a 16-bit downcounter and a 16-bit reload register. In addition, each PRT features a clock prescaler with four selectable taps for CLK ÷ 4, CLK ÷ 16, CLK ÷ 64, and CLK ÷ 256. Each timer can be individually enabled to operate in either SINGLE PASS or CONTINUOUS mode. The timer can be programmed to start, stop, restart from the current value, or restart from the initial value, and generate interrupts to the CPU.

Four of the Programmable Reload Timers (timers 0–3) feature a selectable clock source input. The input for these timers can be either the system clock or the Real-Time Clock (RTC) source. Timers 0–3 can also be used for event counting, with their inputs received from a GPIO port pin. Output from timers 4 and 5 can be directed to a GPIO port pin.

Each of the six PRTs available on the eZ80L92 can be controlled individually. They do not share the same counters, reload registers, control registers, or interrupt signals.
A simplified block diagram of a programmable reload timer is illustrated in Figure 18.

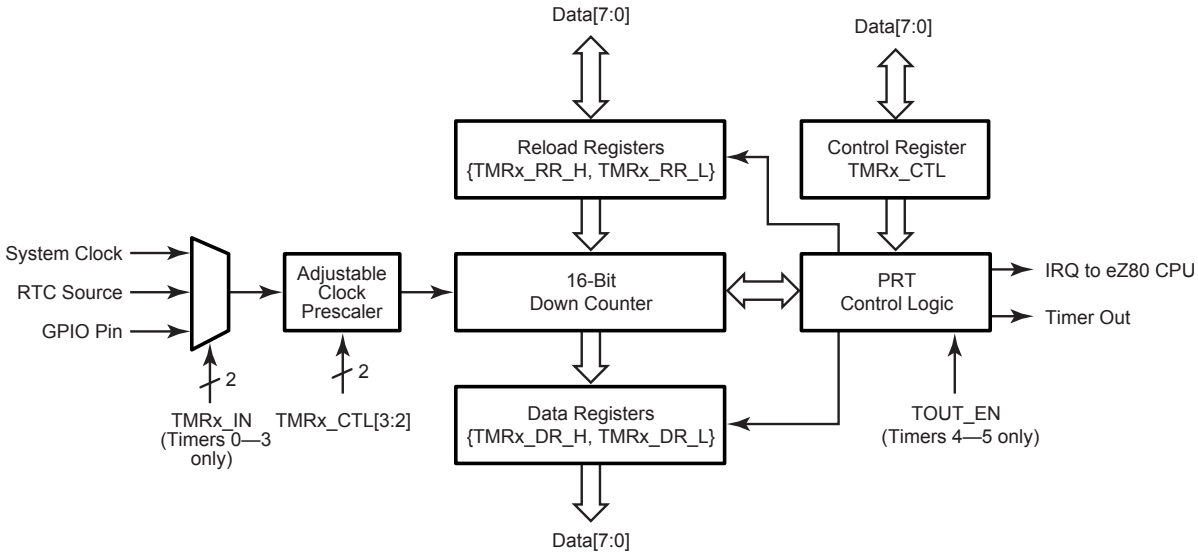

Figure 18. Programmable Reload Timer Block Diagram

Programmable Reload Timer Operation

Setting Timer Duration

There are three factors to consider when determining Programmable Reload Timer duration—clock frequency, clock divider ratio, and initial count value. Minimum duration

The third source of a receiver interrupt is a line status error, indicating an error in byte reception. This error may result from:

- Incorrect received parity.
- Incorrect framing; that is, the stop bit is not detected by receiver at the end of the byte.
- Receiver over run condition.
- A BREAK condition being detected on the receive data input.

An interrupt due to one of the above conditions is cleared when the UARTx_LSR register is read. In FIFO mode, a line status interrupt is generated only after the received byte with an error reaches the top of the FIFO and is ready to be read.

A line status interrupt is activated (provided this interrupt is enabled) as long as the Read pointer of the receiver FIFO points to the location of the FIFO that contains a byte with the error. The interrupt is immediately cleared when the UARTx_LSR register is read. The ERR bit of the UARTx_LSR register is active as long as an erroneous byte is present in the receiver FIFO.

UART Modem Status Interrupt

The modem status interrupt is generated if there is any change in state of the modem status inputs to the UART. This interrupt is cleared when the processor reads the UARTx_MSR register.

UART Recommended Usage

The following is the standard sequence of events that occur in the eZ80L92 MCU using the UART. A description of each follows.

1. Module reset.
2. Control transfers to configure UART operation.
3. Data transfers.

Module Reset. Upon reset, all internal registers are set to their default values. All command status registers are programmed with their default values, and the FIFOs are flushed.

Control Transfers. Based on the requirements of the application, the data transfer baud rate is determined and the BRG is configured to generate a 16X clock frequency. Interrupts are disabled and the communication control parameters are programmed in the UARTx_LCTL register. The FIFO configuration is determined and the receive trigger levels are set in the UARTx_FCTL register. The status registers, UARTx_LSR and UARTx_MSR, are read, and ensure that none of the interrupt sources are active. The inter-

SPI Signals

The four basic SPI signals are:

- MISO (Master-In/Slave-Out)
- MOSI (Master-Out/Slave-In)
- SCK (SPI Serial Clock)
- $\overline{SS}$ (Slave Select)

The following section describes the SPI signals. Each signal is described in both MASTER and SLAVE modes.

Master-In, Slave-Out

The Master-In/Slave-Out (MISO) pin is configured as an input in a master device and as an output in a slave device. It is one of the two lines that transfer serial data, with the most significant bit sent first. The MISO pin of a slave device is placed in a high-impedance state if the slave is not selected. When the SPI is not enabled, this signal is in a high-impedance state.

Master-Out, Slave-In

The Master-Out/Slave-In (MOSI) pin is configured as an output in a master device and as an input in a slave device. It is one of the two lines that transfer serial data, with the most significant bit sent first. When the SPI is not enabled, this signal is in a high-impedance state.

Slave Select

The active Low Slave Select ($\overline{SS}$) input signal is used to select the SPI as a slave device. It must be Low prior to all data communication and must stay Low for the duration of the data transfer.

The $\overline{SS}$ input signal must be High for the SPI to operate as a master device. If the $\overline{SS}$ signal goes Low, a Mode Fault error flag (MODF) is set in the SPI_SR register. See the SPI Status Register (SPI_SR) on page 136 for more information.

When the clock phase (CPHA) is set to 0, the shift clock is the logical OR of $\overline{SS}$ with SCK. In this clock phase mode, $\overline{SS}$ must go High between successive characters in an SPI message. When CPHA is set to 1, $\overline{SS}$ can remain Low for several SPI characters. In cases where there is only one SPI slave, its $\overline{SS}$ line could be tied Low as long as CPHA is set to 1. See the SPI Control Register (SPI_CTL) on page 135 for more information about CPHA.

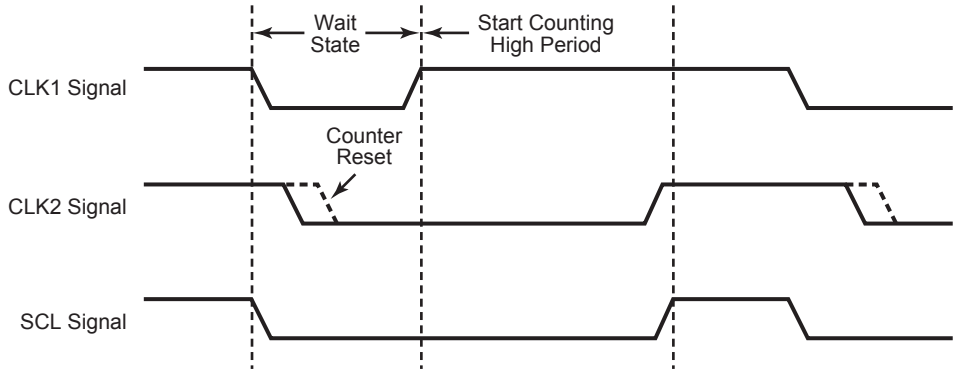

Figure 34. Clock Synchronization In I²C Protocol

Arbitration

A master may start a transfer only if the bus is free. Two or more masters may generate a START condition within the minimum hold time of the START condition which results in a defined START condition to the bus. Arbitration takes place on the SDA line, while the SCL line is at the High level, in such a way that the master which transmits a High level, while another master is transmitting a Low level switches off its data output stage because the level on the bus doesn't correspond to its own level.

Arbitration can continue for many bits. Its first stage is comparison of the address bits. If the masters are each trying to address the same device, arbitration continues with comparison of the data. Because address and data information about the I²C bus is used for arbitration, no information is lost during this process. A master which loses the arbitration can generate clock pulses until the end of the byte in which it loses the arbitration.

If a master also incorporates a slave function and it loses arbitration during the addressing stage, it's possible that the winning master is trying to address it. The losing master must switch over immediately to its slave-receiver mode. Figure 34 illustrates the arbitration procedure for two masters. Of course, more may be involved (depending on how many masters are connected to the bus). The moment there is a difference between the internal data level of the master generating DATA 1 and the actual level on the SDA line, its data output is switched off, which means that a High output level is then connected to the bus. As a result, the data transfer initiated by the winning master is not affected. Because control of the I²C bus is decided solely on the address and data sent by competing masters, there is no central master, nor any order of priority on the bus.

Special attention must be paid if, during a serial transfer, the arbitration procedure is still in progress at the moment when a repeated START condition or a STOP condition is transmitted to the I²C bus. If it is possible for such a situation to occur, the masters involved must send this repeated START condition or STOP condition at the same position in the format frame. In other words, arbitration is not allowed between:

- A repeated START condition and a data bit.
- A STOP condition and a data bit.
- A repeated START condition and a STOP condition.

Clock Synchronization for Handshake

The Clock synchronizing mechanism can function as a handshake, enabling receivers to cope with fast data transfers, on either a byte or bit level. The byte level allows a device to receive a byte of data at a fast rate, but allows the device more time to store the received byte or to prepare another byte for transmission. Slaves hold the SCL line Low after reception and acknowledge the byte, forcing the master into a wait state until the slave is ready for the next byte transfer in a handshake procedure.

Operating Modes

Master Transmit

In MASTER TRANSMIT mode, the I^2C transmits a number of bytes to a slave receiver.

Enter MASTER TRANSMIT mode by setting the STA bit in the I2C_CTL register to 1. The I^2C then tests the I^2C bus and transmits a START condition when the bus is free. When a START condition is transmitted, the IFLG bit is 1 and the status code in the I2C_SR register is 08h. Before this interrupt is serviced, the I2C_DR register must be loaded with either a 7-bit slave address or the first part of a 10-bit slave address, with the lsb cleared to 0 to specify TRANSMIT mode. The IFLG bit should now be cleared to 0 to prompt the transfer to continue.

After the 7-bit slave address (or the first part of a 10-bit address) plus the Write bit are transmitted, the IFLG is set again. A number of status codes are possible in the I2C_SR register. See Table 75.

Bit Position	Value	Description
[7:0] SLAX	00h–FFh	Least significant 8 bits of the 10-bit extended slave address.

I²C Data Register

This register contains the data byte/slave address to be transmitted or the data byte just received. In transmit mode, the most significant bit of the byte is transmitted first. In receive mode, the first bit received is placed in the most significant bit of the register. After each byte is transmitted, the I2C_DR register contains the byte that is present on the bus in case a lost arbitration event occurs. See Table 83.

Table 83. I²C Data Register (I2C_DR = 00CAh)

Bit	7	6	5	4	3	2	1	0
Reset	0	0	0	0	0	0	0	0
CPU Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Note: R/W = Read/Write.								

Bit Position	Value	Description
[7:0] DATA	00h–FFh	I²C data byte.

I²C Control Register

The I2C_CTL register is a control register that is used to control the interrupts and the master slave relationships on the I²C bus.

When the Interrupt Enable bit (IEN) is set to 1, the interrupt line goes High when the IFLG is set to 1. When IEN is cleared to 0, the interrupt line always remains Low.

When the Bus Enable bit (ENAB) is set to 0, the I²C bus inputs SCLx and SDAx are ignored and the I²C module does not respond to any address on the bus. When ENAB is set to 1, the I²C responds to calls to its slave address and to the general call address if the GCE bit (I2C_SAR[0]) is set to 1.

When the Master Mode Start bit (STA) is set to 1, the I²C enters MASTER mode and sends a START condition on the bus when the bus is free. If the STA bit is set to 1 when the I²C module is already in MASTER mode and one or more bytes are transmitted, then a repeated START condition is sent. If the STA bit is set to 1 when the I²C block is being

can be repeated. This allows repeated Read or Write operations without having to resend the ZDI command. A START signal must follow to initiate a new ZDI command. Figure 39 illustrates the timing for address Writes to ZDI registers.

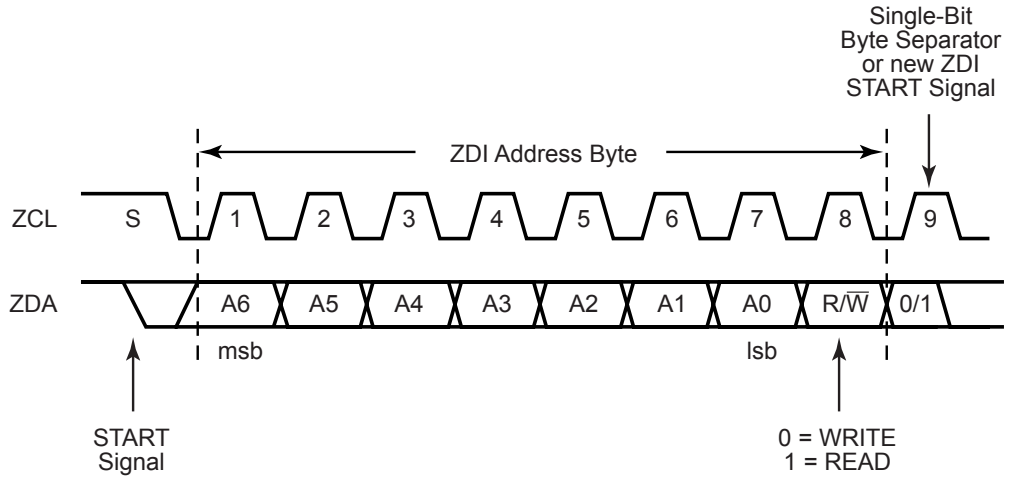

Figure 39. ZDI Address Write Timing

ZDI Write Operations

ZDI Single-Byte Write

For single-byte Write operations, the address and write control bit are first written to the ZDI block. Following the single-bit byte separator, the data is shifted into the ZDI block on the next 8 rising edges of ZCL. The master terminates activity after 8 clock cycles. Figure 40 illustrates the timing for ZDI single-byte Write operations.

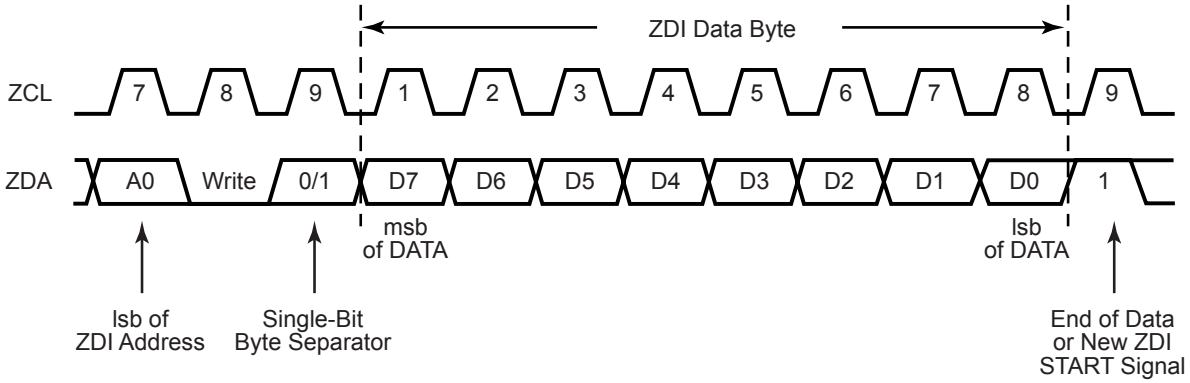

Figure 40. ZDI Single-Byte Data Write Timing

OCI Information Requests

For additional information regarding On-Chip Instrumentation, or to order OCI debug tools, please contact:

First Silicon Solutions, Inc.
5440 SW Westgate Drive, Suite 240
Portland, OR 97221
Phone: (503) 292-6730
Fax: (503) 292-5840
www.fs2.com

Table 108. Rotate and Shift Instructions

Mnemonic	Instruction
RLC	Rotate Left Circular
RLCA	Rotate Left Circular–Accumulator
RLD	Rotate Left Decimal
RR	Rotate Right
RRA	Rotate Right–Accumulator
RRC	Rotate Right Circular
RRCA	Rotate Right Circular–Accumulator
RRD	Rotate Right Decimal
SLA	Shift Left Arithmetic
SRA	Shift Right Arithmetic
SRL	Shift Right Logical

Table 124. External I/O Write Timing

		20 MHz (ns)		50 MHz (ns)	
Parameter	**Description**	**Min**	**Max**	**Min**	**Max.**
T_1	Clock Rise to ADDR Valid Delay	—	7.7	—	7.7
T_2	Clock Rise to ADDR Hold Time	2.2	—	2.2	—
T_3	Clock Fall to Output DATA Valid Delay	—	6	—	6
T_4	Clock Rise to DATA Hold Time	2.3	—	2.3	—
T_5	Clock Rise to $\overline{CSx}$ Assertion Delay	2.6	10.8	2.6	10.8
T_6	Clock Rise to $\overline{CSx}$ Deassertion Delay	2.4	8.8	2.4	8.8
T_7	Clock Rise to $\overline{IORQ}$ Assertion Delay	2.6	7.0	2.6	7.0
T_8	Clock Rise to $\overline{IORQ}$ Deassertion Delay	2.3	6.3	2.3	6.3
T_9	Clock Fall to $\overline{WR}$ Assertion Delay	1.8	4.5	1.8	4.5
T_{10}	Clock Rise to $\overline{WR}$ Deassertion Delay*	1.6	4.4	1.6	4.4
	$\overline{WR}$ Deassertion to ADDR Hold Time	0.4	—	0.4	—
	$\overline{WR}$ Deassertion to DATA Hold Time	0.5	—	0.5	—
	$\overline{WR}$ Deassertion to $\overline{CSx}$ Hold Time	1.2	—	1.2	—
	$\overline{WR}$ Deassertion to $\overline{IORQ}$ Hold Time	0.5	—	0.5	—

Note: *At the conclusion of a Write cycle, deassertion of $\overline{WR}$ always occurs before any change to ADDR, DATA, $\overline{CSx}$, or $\overline{IORQ}$. In certain applications, the deassertion of $\overline{WR}$ can be concurrent with ADDR, DATA, $\overline{CSx}$, or $\overline{MREQ}$ when buffering is used off-chip.

N

O

P