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	Active
Core Processor	eZ80
Core Size	8-Bit
Speed	50MHz
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/ez80l92az050eg

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

Pin No	Symbol	Function	Signal Direction	Description
80	PC4	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.
	DTR1	Data Terminal Ready	Output, Active Low	Modem control signal to the UART. This signal is multiplexed with PC4.
81	PC5	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.
	DSR1	Data Set Ready	Input, Active Low	Modem status signal to the UART. This signal is multiplexed with PC5.
82	PC6	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.
	DCD1	Data Carrier Detect	Input, Active Low	Modem status signal to the UART. This signal is multiplexed with PC6.
83	PC7	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.
	RI1	Ring Indicator	Input, Active Low	Modem status signal to the UART. This signal is multiplexed with PC7.

Table 3. Register Map (Continued)

Address (hex)	Mnemonic	Name	Reset (hex)	CPU Access	Page No
00E9	RTC_AMIN	RTC Alarm Minutes Register	XX	R/W	99
00EA	RTC_AHRS	RTC Alarm Hours Register	XX	R/W	100
00EB	RTC_ADOW	RTC Alarm Day-of-the-Week Register	0X	R/W	101
00EC	RTC_ACTRL	RTC Alarm Control Register	00	R/W	102
00ED	RTC_CTRL	RTC Control Register[4]	x0xxxx00b/ x0xxxx10b	R/W	103
Chip Select Bus Mode Control					
00F0	CS0_BMC	Chip Select 0 Bus Mode Control Register	02h	R/W	71
00F1	CS1_BMC	Chip Select 1 Bus Mode Control Register	02h	R/W	71
00F2	CS2_BMC	Chip Select 2 Bus Mode Control Register	02h	R/W	71
00F3	CS3_BMC	Chip Select 3 Bus Mode Control Register	02h	R/W	71

Notes

1. After an external pin reset, the Watchdog Timer Control register is reset to 00h. After a Watchdog Timer time-out reset, the Watchdog Timer Control register is reset to 20h.
2. When the CPU reads this register, the current sampled value of the port is read.
3. Read-only if RTC is locked; Read/Write if RTC is unlocked.
4. After an external pin reset or a WDT reset, the RTC Control register is reset to x0xxxx00b. After an RTC Alarm Sleep-Mode Recovery reset, the RTC Control register is reset to x0xxxx10b.

Signal timing for Intel® Bus Mode with multiplexed address and data is illustrated for a Read operation in Figure 12 and for a Write operation in Figure 13. In the figures, each Intel® Bus Mode state is 2 eZ80 system clock cycles in duration. Figure 12 and Figure 13 also illustrate the assertion of one WAIT state (T_{WAIT}) by the selected peripheral.

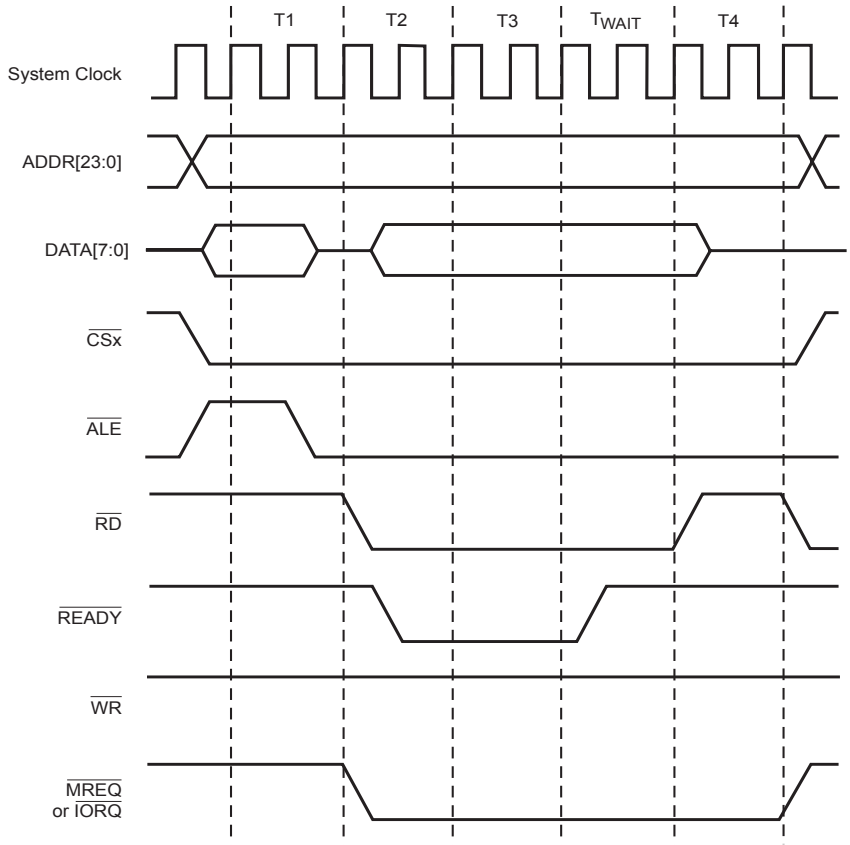

Figure 12. Intel® Bus Mode Read Timing Example (Multiplexed Address and Data Bus)

Table 20. Motorola Bus Mode Read States (Continued)

STATE S6	During state S6, data from the external peripheral device is driven onto the data bus.
STATE S7	On the rising edge of the clock entering state S7, the CPU latches data from the addressed peripheral device and deasserts $\overline{AS}$ and $\overline{DS}$. The peripheral device deasserts $\overline{DTACK}$ at this time.

The eight states for a Write operation in Motorola Bus Mode are described in Table 21.

Table 21. Motorola Bus Mode Write States

STATE S0	The Write cycle starts in S0. The CPU drives R/$\overline{W}$ High (if a preceding Write cycle leaves R/$\overline{W}$ Low).
STATE S1	Entering S1, the CPU drives a valid address on the address bus.
STATE S2	On the rising edge of S2, the CPU asserts $\overline{AS}$ and drives R/$\overline{W}$ Low.
STATE S3	During S3, the data bus is driven out of the high-impedance state as the data to be written is placed on the bus.
STATE S4	At the rising edge of S4, the CPU asserts $\overline{DS}$. The CPU waits for a cycle termination signal $\overline{DTACK}$ ($\overline{WAIT}$). If the termination signal is not asserted at least one full CPU clock period prior to the rising clock edge at the end of S4, the CPU inserts WAIT (T_{WAIT}) states until $\overline{DTACK}$ is asserted. Each WAIT state is a full bus mode cycle.
STATE S5	During S5, no bus signals are altered.
STATE S6	During S6, no bus signals are altered.
STATE S7	Upon entering S7, the CPU deasserts $\overline{AS}$ and $\overline{DS}$. As the clock rises at the end of S7, the CPU drives R/$\overline{W}$ High. The peripheral device deasserts $\overline{DTACK}$ at this time.

enabled) can be active. For I/O Chip Selects, this register defines the address to which ADDR[15:8] is compared to generate an I/O Chip Select. All Chip Select lower bound registers reset to 00h.

Table 22. Chip Select x Lower Bound Registers (CS0_LBR = 00A8h, CS1_LBR = 00ABh, CS2_LBR = 00AEh, CS3_LBR = 00B1h)

Bit	7	6	5	4	3	2	1	0
CS0_LBR Reset	0	0	0	0	0	0	0	0
CS1_LBR Reset	0	0	0	0	0	0	0	0
CS2_LBR Reset	0	0	0	0	0	0	0	0
CS3_LBR 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] CSx_LBR	00h–FFh	For Memory Chip Selects (CSx_IO = 0) This byte specifies the lower bound of the Chip Select address range. The upper byte of the address bus, ADDR[23:16], is compared to the values contained in these registers for determining whether a Memory Chip Select signal must be generated.
		For I/O Chip Selects (CSx_IO = 1) This byte specifies the Chip Select address value. ADDR[15:8] is compared to the values contained in these registers for determining whether an I/O Chip Select signal must be generated.

of the timer is achieved by loading 0001h. Maximum duration is achieved by loading 0000h, because the timer first rolls over to FFFFh and then continues counting down to 0000h.

The time-out period of the PRT is returned by the following equation:

$$\text{PRT Time-Out Period} = \frac{\text{Clock Divider Ratio x Reload Value}}{\text{System Clock Frequency}}$$

To calculate the time-out period with the above equation when using an initial value of 0000h, enter a reload value of 65536 (FFFFh + 1).

Minimum time-out duration is 4 times longer than the input clock period and is generated by setting the clock divider ratio to 1:4 and the reload value to 0001h. Maximum time-out duration is 2^{24} (16,777,216) times longer than the input clock period and is generated by setting the clock divider ratio to 1:256 and the reload value to 0000h.

SINGLE PASS Mode

In SINGLE PASS mode, when the end-of-count value, 0000h, is reached, counting halts, the timer is disabled, and the PRT_EN bit resets to 0. To restart the timer, the CPU must reenable the timer by setting the PRT_EN bit to 1. An example of a PRT operating in SINGLE PASS mode is illustrated in Figure 19. Timer register information is indicated in Table 29.

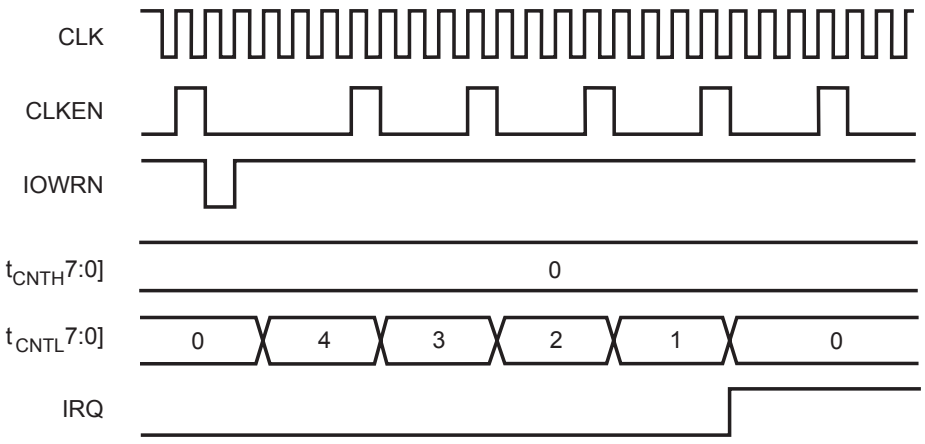

Figure 19. PRT SINGLE PASS Mode Operation Example

Real Time Clock Year Register

This register contains the current year count. See Table 44.

Table 44. Real Time Clock Year Register (RTC_YR = 00E6h)

Bit	7	6	5	4	3	2	1	0
Reset	X	X	X	X	X	X	X	X
CPU Access	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*
Note: X = Unchanged by RESET; R/W* = Read-only if RTC locked, Read/Write if RTC unlocked.								

Binary-Coded-Decimal Operation (BCD_EN = 1)		
Bit Position	**Value**	**Description**
[7:4] TENS_YR	0–9	The tens digit of the current year count.
[3:0] YR	0–9	The ones digit of the current year count.

Binary Operation (BCD_EN = 0)		
Bit Position	**Value**	**Description**
[7:0] YR	00h–63h	The current year count.

Real Time Clock Alarm Day-of-the-Week Register

This register contains the alarm day-of-the-week value. See Table 49.

Table 49. Real Time Clock Alarm Day-of-the-Week Register (RTC_ADOW = 00EBh)

Bit	7	6	5	4	3	2	1	0
Reset	0	0	0	0	X	X	X	X
CPU Access	R	R	R	R	R/W*	R/W*	R/W*	R/W*

Note: X = Unchanged by RESET; R = Read Only; R/W* = Read-only if RTC locked, Read/Write if RTC unlocked.

Binary-Coded-Decimal Operation (BCD_EN = 1)

Bit Position	Value	Description
[7:4]	0000	Reserved.
[3:0] ADOW	1-7	The alarm day-of-the-week.value.

Binary Operation (BCD_EN = 0)

Bit Position	Value	Description
[7:4]	0000	Reserved.
[3:0] ADOW	01h–07h	The alarm day-of-the-week value.

rupts are enabled (except for the transmit interrupt) and the application is ready to use the module for transmission/reception.

Data Transfers—Transmit. To transmit data, the application enables the transmit interrupt. An interrupt is immediately expected in response. The application reads the UARTx_IIR register and determines that the interrupt occurs due to an empty UARTx_THR register. When the application determines this occurrence, the application writes the transmit data bytes to the UARTx_THR register. The number of bytes that the application writes depends on whether or not the FIFO is enabled. If the FIFO is enabled, the application can write 16 bytes at a time. If not, the application can write one byte at a time. As a result of the first Write, the interrupt is deactivated. The processor then waits for the next interrupt. When the interrupt is raised by the UART module, the processor repeats the same process until it exhausts all of the data for transmission.

To control and check the modem status, the application sets up the modem by writing to the UARTx_MCTL register and reading the UARTx_MCTL register before starting the process mentioned above.

Data Transfers—Receive. The receiver is always enabled, and it continually checks for the start bit on the RxD input signal. When an interrupt is raised by the UART module, the application reads the UARTx_IIR register and determines the cause for the interrupt. If the cause is a line status interrupt, the application reads the UARTx_LSR register, reads the data byte and then can discard the byte or take other appropriate action. If the interrupt is caused by a receive-data-ready condition, the application alternately reads the UARTx_LSR and UARTx_RBR registers and removes all of the received data bytes. It reads the UARTx_LSR register before reading the UARTx_RBR register to determine that there is no error in the received data.

To control and check modem status, the application sets up the modem by writing to the UARTx_MCTL register and reading the UARTx_MSR register before starting the process mentioned above.

Poll Mode Transfers. When interrupts are disabled, all data transfers are referred to as poll mode transfers. In poll mode transfers, the application must continually poll the UARTx_LSR register to transmit or receive data without enabling the interrupts. The same holds true for the UARTx_MSR register. If the interrupts are not enabled, the data in the UARTx_IIR register cannot be used to determine the cause of interrupt.

Baud Rate Generator

The Baud Rate Generator consists of a 16-bit downcounter, two registers, and associated decoding logic. The initial value of the Baud Rate Generator is defined by the two BRG Divisor Latch registers, {UARTx_BRG_H, UARTx_BRG_L}. At the rising edge of each system clock, the BRG decrements until it reaches the value `0001h`. On the next system clock rising edge, the BRG reloads the initial value from {UARTx_BRG_H,

UARTx_BRG_L) and outputs a pulse to indicate the end-of-count. Calculate the UART data rate with the following equation:

$$\text{UART Data Rate (bps)} = \frac{\text{System Clock Frequency}}{\text{16 x (UART Baud Rate Generator Divisor)}}$$

Upon RESET, the 16-bit BRG divisor value resets to 0002h. A minimum BRG divisor value of 0001h is also valid, and effectively bypasses the BRG. A software Write to either the Low- or High-byte registers for the BRG Divisor Latch causes both the Low and High bytes to load into the BRG counter, and causes the count to restart.

The divisor registers can only be accessed if bit 7 of the UART Line Control register (UART*x*_LCTL) is set to 1. After reset, this bit is reset to 0.

Recommended Usage of the Baud Rate Generator

The following is the normal sequence of operations that should occur after the eZ80L92 MCU is powered on to configure the Baud Rate Generator:

- Set UARTx_LCTL[7] to 1 to enable access of the BRG divisor registers
- Program the UARTx_BRG_L and UARTx_BRG_H registers
- Clear UARTx_LCTL[7] to 0 to disable access of the BRG divisor registers

BRG Control Registers

UART Baud Rate Generator Registers—Low and High Bytes

The registers hold the Low and High bytes of the 16-bit divisor count loaded by the processor for UART baud rate generation. The 16-bit clock divisor value is returned by {UART*x*_BRG_H, UART*x*_BRG_L}, where *x* is either 0 or 1 to identify the two available UART devices. On RESET, the 16-bit BRG divisor value resets to 0002h. The initial 16-bit divisor value must be between 0002h and FFFFh as the values 0000h and 0001h are invalid, and proper operation is not guaranteed. As a result, the minimum BRG clock divisor ratio is 2.

A Write to either the Low- or High-byte registers for the BRG Divisor Latch causes both bytes to be loaded into the BRG counter. The count is then restarted.

Bit 7 of the associated UART Line Control register (UART*x*_LCTL) must be set to 1 to access this register. See Table 52 and Table 53. For more information, see UART Line Control Registers (UARTx_LCTL) on page 115.

Note: The UARTx_BRG_L registers share the same address space with the UARTx_RBR and UARTx_THR registers. The UARTx_BRG_H registers share the same address space with the UARTx_IER registers. Bit 7 of the associated UART Line Control register (UARTx_LCTL) must be set to 1 to enable access to the BRG registers.

Table 52. UART Baud Rate Generator Registers—Low Byte (UART0_BRG_L = 00C0h, UART1_BRG_L = 00D0h)

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

Bit Position	Value	Description
[7:0] UARTx_BRG_L	00h–FFh	These bits represent the Low byte of the 16-bit Baud Rate Generator divider value. The complete BRG divisor value is returned by {UARTx_BRG_H, UARTx_BRG_L}.

Table 53. UART Baud Rate Generator Registers—High Byte (UART0_BRG_H = 00C1h, UART1_BRG_H = 00D1h)

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 = Read only; R/W = Read/Write.								

Bit Position	Value	Description
[7:0] UARTx_BRG_H	00h–FFh	These bits represent the High byte of the 16-bit Baud Rate Generator divider value. The complete BRG divisor value is returned by {UARTx_BRG_H, UARTx_BRG_L}.

UART Registers

After a RESET, all UART registers are set to their default values. Any Writes to unused registers or register bits are ignored and Reads return a value of 0. For compatibility with future revisions, unused bits within a register must be written with a value of 0. Read/

Write attributes, reset conditions, and bit descriptions of all of the UART registers are provided in this section.

UART Transmit Holding Registers

If less than eight bits are programmed for transmission, the lower bits of the byte written to this register are selected for transmission. The transmit FIFO is mapped at this address. You can write up to 16 bytes for transmission at one time to this address if the FIFO is enabled by the application. If the FIFO is disabled, this buffer is only one byte deep.

These registers share the same address space as the UARTx_RBR and UARTx_BRG_L registers. See Table 54.

Table 54. UART Transmit Holding Registers (UART0_THR = 00C0h, UART1_THR = 00D0h)

Bit	7	6	5	4	3	2	1	0
Reset	X	X	X	X	X	X	X	X
CPU Access	W	W	W	W	W	W	W	W
Note: W = Write only.								

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

UART Receive Buffer Registers

The bits in this register reflect the data received. If less than eight bits are programmed for receive, the lower bits of the byte reflect the bits received whereas upper unused bits are 0. The receive FIFO is mapped at this address. If the FIFO is disabled, this buffer is only one byte deep.

These registers share the same address space as the UARTx_THR and UARTx_BRG_L registers. See Table 55.

Bit Position	Value	Description
6 TEMT	0	Transmit holding register/FIFO is not empty or transmit shift register is not empty or transmitter is not idle.
	1	Transmit holding register/FIFO and transmit shift register are empty; and the transmitter is idle. This bit cannot be set to 1 during the BREAK condition. This bit only becomes 1 after the BREAK command is removed.
5 THRE	0	Transmit holding register/FIFO is not empty.
	1	Transmit holding register/FIFO is empty. This bit cannot be set to 1 during the BREAK condition. This bit only becomes 1 after the BREAK command is removed.
4 BI	0	Receiver does not detect a BREAK condition. This bit is reset to 0 when the UARTx_LSR register is read.
	1	Receiver detects a BREAK condition on the receive input line. This bit is 1 if the duration of BREAK condition on the receive data is longer than one character transmission time, the time depends on the programming of the UARTx_LSR register. In case of FIFO only one null character is loaded into the receiver FIFO with the framing error. The framing error is revealed to the eZ80 whenever that particular data is read from the receiver FIFO.
3 FE	0	No framing error detected for character at the top of the FIFO. This bit is reset to 0 when the UARTx_LSR register is read.
	1	Framing error detected for the character at the top of the FIFO. This bit is set to 1 when the stop bit following the data/parity bit is logic 0.
2 PE	0	The received character at the top of the FIFO does not contain a parity error. This bit is reset to 0 when the UARTx_LSR register is read.
	1	The received character at the top of the FIFO contains a parity error.

Bit Position	Value	Description
7 DCD	0–1	Data Carrier Detect In NORMAL mode, this bit reflects the inverted state of the $\overline{\text{DCDx}}$ input pin. In LOOP BACK mode, this bit reflects the value of the UARTx_MCTL[3] = out2.
6 RI	0–1	Ring Indicator In NORMAL mode, this bit reflects the inverted state of the $\overline{\text{RIx}}$ input pin. In LOOP BACK mode, this bit reflects the value of the UARTx_MCTL[2] = out1.
5 DSR	0–1	Data Set Ready In NORMAL mode, this bit reflects the inverted state of the $\overline{\text{DSRx}}$ input pin. In LOOP BACK mode, this bit reflects the value of the UARTx_MCTL[0] = DTR.
4 CTS	0–1	Clear to Send In NORMAL mode, this bit reflects the inverted state of the $\overline{\text{CTSx}}$ input pin. In LOOP BACK mode, this bit reflects the value of the UARTx_MCTL[1] = RTS.
3 DDCD	0–1	Delta Status Change of $\overline{\text{DCD}}$ This bit is set to 1 whenever the $\overline{\text{DCDx}}$ pin changes state. This bit is reset to 0 when the UARTx_MSR register is read.
2 TERI	0–1	Trailing Edge Change on $\overline{\text{RI}}$. This bit is set to 1 whenever a falling edge is detected on the $\overline{\text{RIx}}$ pin. This bit is reset to 0 when the UARTx_MSR register is read.
1 DDSR	0–1	Delta Status Change of $\overline{\text{DSR}}$ This bit is set to 1 whenever the $\overline{\text{DSRx}}$ pin changes state. This bit is reset to 0 when the UARTx_MSR register is read.
0 DCTS	0–1	Delta Status Change of $\overline{\text{CTS}}$ This bit is set to 1 whenever the $\overline{\text{CTSx}}$ pin changes state. This bit is reset to 0 when the UARTx_MSR register is read.

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.

Table 75. I²C Master Transmit Status Codes

Code	I²C State	MCU Response	Next I²C Action
18h	Addr+W transmitted[1], ACK received	For a 7-bit address: write byte to DATA, clear IFLG	Transmit data byte, receive ACK
		Or set STA, clear IFLG	Transmit repeated START
		Or set STP, clear IFLG	Transmit STOP
		Or set STA & STP, clear IFLG	Transmit STOP then START
		For a 10-bit address: write extended address byte to DATA, clear IFLG	Transmit extended address byte
20h	Addr+W transmitted, ACK not received	Same as code 18h	Same as code 18h
38h	Arbitration lost	Clear IFLG	Return to idle
		Or set STA, clear IFLG	Transmit START when bus is free
68h	Arbitration lost, +W received, ACK transmitted	Clear IFLG, AAK = 0[2]	Receive data byte, transmit NACK
		Or clear IFLG, AAK = 1	Receive data byte, transmit ACK
78h	Arbitration lost, General call addr received, ACK transmitted	Same as code 68h	Same as code 68h
B0h	Arbitration lost, SLA+R received, ACK transmitted	Write byte to DATA, clear IFLG, clear AAK = 0	Transmit last byte, receive ACK
		Or write byte to DATA, clear IFLG, set AAK = 1	Transmit data byte, receive ACK

Notes:
1. W is defined as the Write bit; i.e., the lsb is cleared to 0.
2. AAK is defined as the I²C Acknowledge bit.

If 10-bit addressing is being used, then the status code is `18h` or `20h` after the first part of a 10-bit address plus the Write bit are successfully transmitted.

After this interrupt is serviced and the second part of the 10-bit address is transmitted, the I2C_SR register contains one of the codes in Table 76.

eZ80® CPU Instruction Set

Table 108 through Table 108 indicate the eZ80 CPU instructions available for use with the eZ80L92 MCU. The instructions are grouped by class. A detailed information is available in the eZ80 CPU User Manual.

Table 108. Arithmetic Instructions

Mnemonic	Instruction
ADC	Add with Carry
ADD	Add without Carry
CP	Compare with Accumulator
DAA	Decimal Adjust Accumulator
DEC	Decrement
INC	Increment
MLT	Multiply
NEG	Negate Accumulator
SBC	Subtract with Carry
SUB	Subtract without Carry

Table 108. Bit Manipulation Instructions

Mnemonic	Instruction
BIT	Bit Test
RES	Reset Bit
SET	Set Bit

Table 108. Block Transfer and Compare Instructions

Mnemonic	Instruction
CPD (CPDR)	Compare and Decrement (with Repeat)
CPI (CPIR)	Compare and Increment (with Repeat)

Table 108. Input/Output Instructions

Mnemonic	Instruction
OUT0	Output to I/O on Page 0
OUTD (OTDR)	Output to I/O and Decrement (with Repeat)
OUTD2 (OTD2R)	Output to I/O and Decrement (with Repeat)
OUTI (OTIR)	Output to I/O and Increment (with Repeat)
OUTI2 (OTI2R)	Output to I/O and Increment (with Repeat)
TSTIO	Test I/O

Table 108. Load Instructions

Mnemonic	Instruction
LD	Load
LEA	Load Effective Address
PEA	Push Effective Address
POP	Pop
PUSH	Push

Table 108. Logical Instructions

Mnemonic	Instruction
AND	Logical AND
CPL	Complement Accumulator
OR	Logical OR
TST	Test Accumulator
XOR	Logical Exclusive OR

Table 108. Processor Control Instructions

Mnemonic	Instruction
CCF	Complement Carry Flag
DI	Disable Interrupts

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.

Customer Support

For answers to technical questions about the product, documentation, or any other issues with ZiLOG's offerings, please visit ZiLOG's Knowledge Base at http://www.zilog.com/kb.

For any comments, detail technical questions, or reporting problems, please visit ZiLOG's Technical Support at http://support.zilog.com.