**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 | eZ8 |
| Core Size | 8-Bit |
| Speed | 20MHz |
| Connectivity | I²C, IrDA, SPI, UART/USART |
| Peripherals | Brown-out Detect/Reset, POR, PWM, WDT |
| Number of I/O | 19 |
| Program Memory Size | 4KB (4K x 8) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 1K x 8 |
| Voltage - Supply (Vcc/Vdd) | 2.7V ~ 3.6V |
| Data Converters | - |
| Oscillator Type | Internal |
| Operating Temperature | 0°C ~ 70°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 28-SOIC (0.295", 7.50mm Width) |
| Supplier Device Package | - |
| Purchase URL | https://www.e-xfl.com/product-detail/zilog/z8f0412sj020sc00tr |

Operation . . . . . 109
Transmitting IrDA Data . . . . . 110
Receiving IrDA Data . . . . . 111
Infrared Endec Control Register Definitions . . . . . 112
**Serial Peripheral Interface . . . . . 113**
Architecture . . . . . 113
Operation . . . . . 114
SPI Signals . . . . . 115
SPI Clock Phase and Polarity Control . . . . . 116
Multi-Master Operation . . . . . 118
Slave Operation . . . . . 118
Error Detection . . . . . 119
SPI Interrupts . . . . . 119
SPI Baud Rate Generator . . . . . 120
SPI Control Register Definitions . . . . . 121
SPI Data Register . . . . . 121
SPI Control Register . . . . . 122
SPI Status Register . . . . . 123
SPI Mode Register . . . . . 124
SPI Diagnostic State Register . . . . . 125
SPI Baud Rate High and Low Byte Registers . . . . . 125
**I2C Controller . . . . . 127**
Architecture . . . . . 127
Operation . . . . . 128
SDA and SCL Signals . . . . . 128
$I^2C$ Interrupts . . . . . 128
Software Control of I2C Transactions . . . . . 129
Start and Stop Conditions . . . . . 130
Master Write and Read Transactions . . . . . 130
Address Only Transaction with a 7-bit Address . . . . . 131
Write Transaction with a 7-Bit Address . . . . . 132
Address Only Transaction with a 10-bit Address . . . . . 133
Write Transaction with a 10-Bit Address . . . . . 134
Read Transaction with a 7-Bit Address . . . . . 136
Read Transaction with a 10-Bit Address . . . . . 137
I2C Control Register Definitions . . . . . 139
I2C Data Register . . . . . 139
I2C Status Register . . . . . 140
I2C Control Register . . . . . 141
I2C Baud Rate High and Low Byte Registers . . . . . 143
I2C Diagnostic State Register . . . . . 143

### Braces

The curly braces { }, indicate a single register or bus created by concatenating some combination of smaller registers, buses, or individual bits.

- **Example:** The 12-bit register address {0H, RP[7:4], R1[3:0]} is composed of a 4-bit hexadecimal value (0H) and two 4-bit register values taken from the Register Pointer (RP) and Working Register R1. 0H is the most significant nibble (4-bit value) of the 12-bit register, and R1[3:0] is the least significant nibble of the 12-bit register.

### Parentheses

The parentheses ( ), indicate an indirect register address lookup.

- **Example:** (R1) is the memory location referenced by the address contained in the Working Register R1.

### Parentheses/Bracket Combinations

The parentheses ( ), indicate an indirect register address lookup and the square brackets, [ ], indicate a register or bus.

- **Example:** Assume PC[15:0] contains the value 1234h. (PC [15:0]) then refers to the contents of the memory location at address 1234h.

### Use of the Words *Set*, *Reset* and *Clear*

The word *set* implies that a register bit or a condition contains a logical 1. The words re*set* or *clear* imply that a register bit or a condition contains a logical 0. When either of these terms is followed by a number, the word *logical* cannot be included; however, it is implied.

### Notation for Bits and Similar Registers

A field of bits within a register is designated as: Register[*n*:*n*].

- **Example:** ADDR[15:0] refers to bits 15 through bit 0 of the Address.

### Use of the Terms *LSB*, *MSB*, *lsb*, and *msb*

In this document, the terms *LSB* and *MSB,* when appearing in upper case, mean *least significant byte* and *most significant byte,* respectively. The lowercase forms, *lsb* and *msb*, mean *least significant bit* and *most significant bit*, respectively.

### Use of Initial Uppercase Letters

Initial uppercase letters designate settings and conditions in general text.

- **Example 1**: The receiver forces the SCL line to Low.
- **Example 2:** The Master generates a STOP condition to abort the transfer.

| Abbreviations/ Acronyms | Expansion |
|---|---|
| PDIP | Plastic Dual Inline Package |
| SOIC | Small Outline Integrated Circuit |
| SSOP | Small Shrink Outline Package |
| PC | Program Counter |
| IRQ | Interrupt Request |

## Block Diagram

Figure 1 displays the block diagram of the architecture of Z8 Encore! XP® F0822 Series devices.

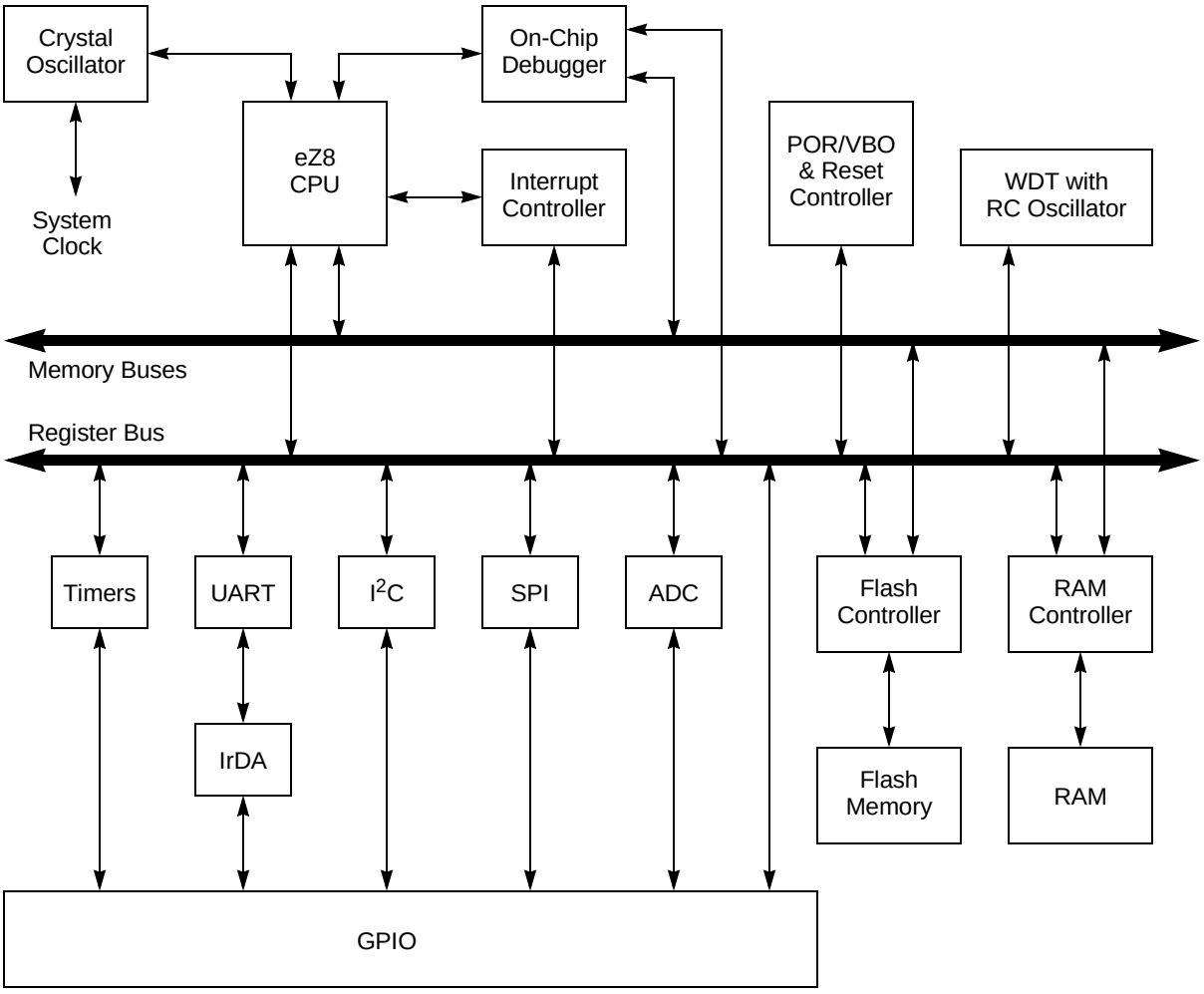



**Figure 1. Z8 Encore! XP® F0822 Series Block Diagram**

## CPU and Peripheral Overview

### eZ8 CPU Features

Zilog's latest eZ8 8-bit CPU, meets the continuing demand for faster and more code-efficient microcontrollers. The eZ8 CPU executes a superset of the original Z8® instruction set.

Interrupt Request 0
IRQ0 (FC0H - Read/Write)

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|

- D0: ADC Interrupt Request
- D1: SPI Interrupt Request
- D2: I2C Interrupt Request
- D3: UART 0 Transmitter Interrupt
- D4: UART 0 Receiver Interrupt
- D5: Timer 0 Interrupt Request
- D6: Timer 1 Interrupt Request
- D7: Reserved

For all of the above peripherals:
0 = Peripheral IRQ is not pending
1 = Peripheral IRQ is awaiting service

IRQ0 Enable High Bit
IRQ0ENH (FC1H - Read/Write)

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|

- D0: ADC IRQ Enable Hit Bit
- D1: SPI IRQ Enable High Bit
- D2: I2C IRQ Enable High Bit
- D3: UART 0 Transmitter IRQ
- D4: UART 0 Receiver IRQ Enable
- D5: Timer 0 IRQ Enable High Bit
- D6: Timer 1 IRQ Enable High Bit
- D7: Reserved

### External Pin Reset

The $\overline{\text{RESET}}$ pin contains a Schmitt-triggered input, an internal pull-up, an analog filter, and a digital filter to reject noise. After the $\overline{\text{RESET}}$ pin is asserted for at least 4 system clock cycles, the device progresses through the System Reset sequence. While the $\overline{\text{RESET}}$ input pin is asserted Low, Z8 Encore! XP F0822 Series device continues to be held in the Reset state. If the $\overline{\text{RESET}}$ pin is held Low beyond the System Reset time-out, the device exits the Reset state immediately following $\overline{\text{RESET}}$ pin deassertion. Following a System Reset initiated by the external $\overline{\text{RESET}}$ pin, the EXT status bit in the Watchdog Timer Control Register (WDTCTL) is set to 1.

### On-Chip Debugger Initiated Reset

A POR is initiated using the OCD by setting the RST bit in the OCD Control Register. The OCD block is not reset but the rest of the chip goes through a normal system reset. The RST bit automatically clears during the system reset. Following the system reset, the POR bit in the WDT Control Register is set.

## Stop Mode Recovery

STOP mode is entered by execution of a STOP instruction by the eZ8 CPU. For detailed information on STOP mode, see Low-Power Modes on page 45. During Stop Mode Recovery, the device is held in reset for 66 cycles of the WDT oscillator followed by 16 cycles of the system clock. Stop Mode Recovery only affects the contents of the WDT Control Register and does not affect any other values in the Register File, including the Stack Pointer, Register Pointer, Flags, Peripheral Control Registers, and General-Purpose RAM.

The eZ8 CPU fetches the Reset vector at Program Memory addresses 0002H and 0003H and loads that value into the Program Counter. Program execution begins at the Reset vector address. Following Stop Mode Recovery, the STOP bit in the WDT Control Register is set to 1. Table 10 lists the Stop Mode Recovery sources and resulting actions. The text following provides more detailed information on each of the Stop Mode Recovery sources.

**Table 10. Stop Mode Recovery Sources and Resulting Action**

| Operating Mode | Stop Mode Recovery Source | Action |
|---|---|---|
| STOP mode | WDT time-out when configured for Reset | Stop Mode Recovery |
| | WDT time-out when configured for interrupt | Stop Mode Recovery followed by interrupt (if interrupts are enabled) |
| | Data transition on any GPIO Port pin enabled as a Stop Mode Recovery source | Stop Mode Recovery |

# Interrupt Controller

The interrupt controller on Z8 Encore! XP® F0822 Series products prioritizes the interrupt requests from the on-chip peripherals and the GPIO port pins. The features of the interrupt controller include the following:

- 19 unique interrupt vectors:
  - 12 GPIO port pin interrupt sources.
  - 7 On-chip peripheral interrupt sources.
- Flexible GPIO interrupts:
  - 8 selectable rising and falling edge GPIO interrupts.
  - 4 dual-edge interrupts.
- Three levels of individually programmable interrupt priority.
- WDT is configured to generate an interrupt.

Interrupt Requests (IRQs) allow peripheral devices to suspend CPU operation in an orderly manner and force the CPU to start an Interrupt Service Routine (ISR). Usually this ISR is involved with the exchange of data, status information, or control information between the CPU and the interrupting peripheral. When the service routine is completed, the CPU returns to the operation from which it was interrupted.

The eZ8 CPU supports both vectored and polled interrupt handling. For polled interrupts, the interrupt control has no effect on operation. For more information on interrupt servicing, refer to *eZ8 CPU Core User Manual (UM0128)* available for download at www.zilog.com.

## Interrupt Vector Listing

Table 24 lists all the interrupts available in order of priority. The interrupt vector is stored with the most significant byte (MSB) at the even Program Memory address and the least significant byte (LSB) at the following odd Program Memory address.

**Table 24. Interrupt Vectors in Order of Priority**

| Priority | Program Memory Vector Address | Interrupt Source |
|---|---|---|
| **Highest** | 0002H | Reset (not an interrupt) |
| | 0004H | WDT (see Watchdog Timer on page 83) |
| | 0006H | Illegal Instruction Trap (not an interrupt) |

## Architecture

Figure 9 displays a block diagram of the interrupt controller.

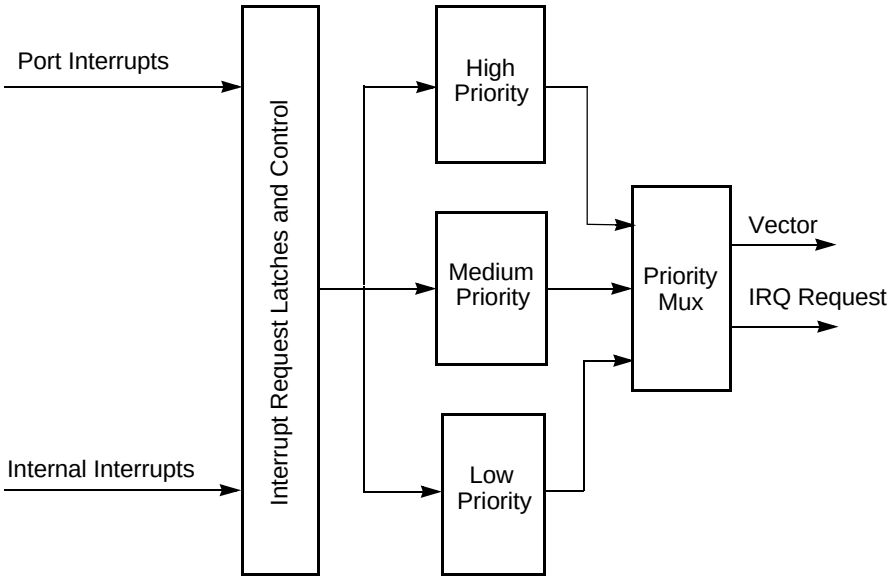



**Figure 9. Interrupt Controller Block Diagram**

## Operation

### Master Interrupt Enable

The master interrupt enable bit (IRQE) in the Interrupt Control Register globally enables and disables interrupts.

Interrupts are globally enabled by any of the following actions:

- Execution of an EI (Enable Interrupt) instruction.
- Execution of an IRET (Return from Interrupt) instruction.
- Writing a 1 to the IRQE bit in the Interrupt Control Register.

Interrupts are globally disabled by any of the following actions:

- Execution of a DI (Disable Interrupt) instruction.
- eZ8 CPU acknowledgement of an interrupt service request from the interrupt controller.
- Writing a 0 to the IRQE bit in the Interrupt Control Register.
- Reset.
- Execution of a Trap instruction.
- Illegal Instruction trap.

**TH and TL—Timer High and Low Bytes**
These 2 bytes, {TMRH[7:0], TMRL[7:0]}, contain the current 16-bit timer count value.

## Timer Reload High and Low Byte Registers

The Timer 0–1 Reload High and Low Byte (TxRH and TxRL) Registers (Table 41) store a 16-bit reload value, {TRH[7:0], TRL[7:0]}. Values written to the Timer Reload High Byte register are stored in a temporary holding register. When a write to the Timer Reload Low Byte Register occurs, the temporary holding register value is written to the Timer High Byte Register. This operation allows simultaneous updates of the 16-bit Timer Reload value.

In COMPARE mode, the Timer Reload High and Low Byte Registers store the 16-bit Compare value.

**Table 41. Timer 0–1 Reload High Byte Register (TxRH)**

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| FIELD | TRH | | | | | | | |
| RESET | 1 | | | | | | | |
| R/W | R/W | | | | | | | |
| ADDR | F02H, F0AH | | | | | | | |

**Table 42. Timer 0–1 Reload Low Byte Register (TxRL)**

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| FIELD | TRL | | | | | | | |
| RESET | 1 | | | | | | | |
| R/W | R/W | | | | | | | |
| ADDR | F03H, F0BH | | | | | | | |

**TRH and TRL—Timer Reload Register High and Low**
These two bytes form the 16-bit Reload value, {TRH[7:0], TRL[7:0]}. This value sets the maximum count value which initiates a timer reload to 0001H. In COMPARE mode, these two bytes form the 16-bit Compare value.

## Timer 0–1 PWM High and Low Byte Registers

The Timer 0–1 PWM High and Low Byte (TxPWMH and TxPWML) registers (Table 43 and Table 44) are used for Pulse-Width Modulator (PWM) operations. These registers also store the Capture values for the CAPTURE and CAPTURE/COMPARE modes.

**STOP—Stop Mode Recovery Indicator**
If this bit is set to 1, a Stop Mode Recovery occurred. If the STOP and WDT bits are both set to 1, the Stop Mode Recovery occurred due to a WDT time-out. If the STOP bit is 1 and the WDT bit is 0, the Stop Mode Recovery was not caused by a WDT time-out. This bit is reset by a POR or a WDT time-out that occurred while not in STOP mode. Reading this register also resets this bit.

**WDT—Watchdog Timer Time-Out Indicator**
If this bit is set to 1, a WDT time-out occurred. A POR resets this pin. A Stop Mode Recovery due a change in an input pin also resets this bit. Reading this register resets this bit.

**EXT—External Reset Indicator**
If this bit is set to 1, a Reset initiated by the external $\overline{\text{RESET}}$ pin occurred. A POR or a Stop Mode Recovery from a change in an input pin resets this bit. Reading this register resets this bit.

**Reserved**
These bits are reserved and must be 0.

## Watchdog Timer Reload Upper, High and Low Byte Registers

The Watchdog Timer Reload Upper, High and Low Byte (WDTU, WDTH, WDTL) Registers (Table 49 through Table 51) form the 24-bit reload value that is loaded into the WDT, when a WDT instruction executes. The 24-bit reload value is {WDTU[7:0], WDTH[7:0], WDTL[7:0]}. Writing to these registers sets the required Reload Value. Reading from these registers returns the current WDT count value.

**! Caution:** *The 24-bit WDT Reload Value must not be set to a value less than* 000004H.

**Table 49. Watchdog Timer Reload Upper Byte Register (WDTU)**

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| FIELD | WDTU | | | | | | | |
| RESET | 1 | | | | | | | |
| R/W | R/W* | | | | | | | |
| ADDR | FF1H | | | | | | | |
| R/W*—Read returns the current WDT count value. Write sets the desired Reload Value. | | | | | | | | |

**WDTU—WDT Reload Upper Byte**
Most significant byte (MSB), Bits[23:16], of the 24-bit WDT reload value.

multi-node network. The following MULTIPROCESSOR modes are available in hardware:

- Interrupt on all address bytes.
- Interrupt on matched address bytes and correctly framed data bytes.
- Interrupt only on correctly framed data bytes.

These modes are selected with MPMD[1:0] in the UART Control 1 Register. For all MULTIPROCESSOR modes, bit MPEN of the UART Control 1 Register must be set to 1.

The first scheme is enabled by writing 01b to MPMD[1:0]. In this mode, all incoming address bytes cause an interrupt, while data bytes never cause an interrupt. The ISR must manually check the address byte that caused triggered the interrupt. If it matches the UART address, the software should clear MPMD[0]. At this point, each new incoming byte interrupts the CPU. The software is then responsible for determining the end-of-frame. It checks for the end-of-frame by reading the MPRX bit of the UART Status 1 Register for each incoming byte. If MPRX=1, then a new frame begins. If the address of this new frame is different from the UART's address, then MPMD[0] must be set to 1 causing the UART interrupts to go inactive until the next address byte. If the new frame's address matches the UART's address, then the data in the new frame should be processed as well.

The second scheme is enabled by setting MPMD[1:0] to 10b and writing the UART's address into the UART Address Compare Register. This mode introduces more hardware control, interrupting only on frames that match the UART's address. When an incoming address byte does not match the UART's address, it is ignored. All successive data bytes in this frame are also ignored. When a matching address byte occurs, an interrupt is issued and further interrupts occur on each successive data byte. The first data byte in the frame contains the NEWFRM=1 in the UART Status 1 Register. When the next address byte occurs, the hardware compares it to the UART's address. If there is a match, the interrupts continue and the NEWFRM bit is set for the first byte of the new frame. If there is no match, then the UART ignores all incoming bytes until the next address match.

The third scheme is enabled by setting MPMD[1:0] to 11b and by writing the UART's address into the UART Address Compare Register. This mode is identical to the second scheme, except that there are no interrupts on address bytes. The first data byte of each frame is still accompanied by a NEWFRM assertion.

## External Driver Enable

The UART provides a Driver Enable (DE) signal for off-chip bus transceivers. This feature reduces the software overhead associated with using a GPIO pin to control the transceiver when communicating on a multi-transceiver bus, such as RS-485.

Driver Enable is an active High signal that envelopes the entire transmitted data frame including parity and STOP bits as displayed in Figure15 on page 97. The Driver Enable signal asserts when a byte is written to the UART Transmit Data Register. The Driver

When reading data from the slave, the I2C pauses after the data Acknowledge cycle until the receive interrupt is serviced and the RDRF bit of the status register is cleared by reading the I2C Data Register. Once the I2C Data Register has been read, the I2C reads the next data byte.

## Address Only Transaction with a 7-bit Address

In the situation where software determines if a slave with a 7-bit address is responding without sending or receiving data, a transaction can be done which only consists of an address phase. Figure 26 on page 131 displays this "address only" transaction to determine if a slave with a 7-bit address will acknowledge. As an example, this transaction can be used after a "write" has been done to a EEPROM to determine when the EEPROM completes its internal write operation and is once again responding to I2C transactions. If the slave does not Acknowledge, the transaction is repeated until the slave does Acknowledge.

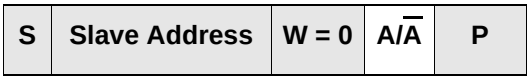



**Figure 26. 7-Bit Address Only Transaction Format**

Follow the steps below for an address only transaction to a 7-bit addressed slave:

1. Software asserts the IEN bit in the I2C Control Register.
2. Software asserts the TXI bit of the I2C Control Register to enable Transmit interrupts.
3. The I2C interrupt asserts, because the I2C Data Register is empty (TDRE = 1)
4. Software responds to the TDRE bit by writing a 7-bit Slave address plus write bit (=0) to the I2C Data Register. As an alternative this could be a read operation instead of a write operation.
5. Software sets the START and STOP bits of the I2C Control Register and clears the TXI bit.
6. The I2C Controller sends the START condition to the I2C Slave.
7. The I2C Controller loads the I2C Shift register with the contents of the I2C Data Register.
8. Software polls the STOP bit of the I2C Control Register. Hardware deasserts the STOP bit when the address only transaction is completed.
9. Software checks the ACK bit of the I2C Status Register. If the slave acknowledged, the ACK bit is equal to 1. If the slave does not acknowledge, the ACK bit is equal to 0. The NCKI interrupt does not occur in the not acknowledge case because the STOP bit was set.

9. Software responds by writing the second byte of address into the contents of the I²C Data Register.
10. The I²C Controller shifts the rest of the first byte of address and write bit out the SDA signal.
11. If the I²C Slave sends an acknowledge by pulling the SDA signal low during the next high period of SCL the I²C Controller sets the ACK bit in the I²C Status register. Continue with step 12.

    If the slave does not acknowledge the first address byte, the I²C Controller sets the NCKI bit and clears the ACK bit in the I²C Status register. Software response to the Not Acknowledge interrupt by setting the STOP and FLUSH bits and clearing the TXI bit. The I2C Controller sends the STOP condition on the bus and clears the STOP and NCKI bits. The transaction is complete (ignore following steps).
12. The I²C Controller loads the I²C Shift register with the contents of the I²C Data Register (2nd byte of address).
13. The I²C Controller shifts the second address byte out the SDA signal. After the first bit has been sent, the Transmit Interrupt is asserted.
14. Software responds by setting the STOP bit in the I²C Control Register. The TXI bit can be cleared at the same time.
15. Software polls the STOP bit of the I²C Control Register. Hardware deasserts the STOP bit when the transaction is completed (STOP condition has been sent).
16. Software checks the ACK bit of the I²C Status register. If the slave acknowledged, the ACK bit is equal to 1. If the slave does not acknowledge, the ACK bit is equal to 0. The NCKI interrupt do not occur because the STOP bit was set.

## Write Transaction with a 10-Bit Address

Figure 29 displays the data transfer format for a 10-bit addressed slave. Shaded regions indicate data transferred from the I²C Controller to slaves and unshaded regions indicate data transferred from the slaves to the I²C Controller.

| S | Slave Address 1st 7 bits | W = 0 | A | Slave Address 2nd Byte | A | Data | A | Data | $\overline{A}$/A | P/S |
|---|---|---|---|---|---|---|---|---|---|---|

**Figure 29. 10-Bit Addressed Slave Data Transfer Format**

The first seven bits transmitted in the first byte are 11110XX. The two bits XX are the two most-significant bits of the 10-bit address. The lowest bit of the first byte transferred is the read/write control bit (=0). The transmit operation is carried out in the same manner as 7-bit addressing.

set, this bit is reset by the I2C Controller after a STOP condition is sent or by deasserting the IEN bit. If this bit is 1, it cannot be cleared to 0 by writing to the register.

**BIRQ—Baud Rate Generator Interrupt Request**
This bit allows the I2C Controller to be used as an additional timer when the I2C Controller is disabled. This bit is ignored when the I2C Controller is enabled.
1 = An interrupt occurs every time the BRG counts down to one.
0 = No BRG interrupt occurs.

**TXI—Enable TDRE interrupts**
This bit enables the transmit interrupt when the I2C Data Register is empty (TDRE = 1).
1 = Transmit Interrupt (and DMA transmit request) is enabled.
0 = Transmit Interrupt (and DMA transmit request) is disabled.

**NAK—Send NAK**
This bit sends a Not Acknowledge condition after the next byte of data is read from the I2C Slave. Once asserted, it is deasserted after a Not Acknowledge is sent or the IEN bit is deasserted. If this bit is 1, it cannot be cleared to 0 by writing to the register.

**FLUSH—Flush Data**
Setting this bit to 1 clears the I2C Data Register and sets the TDRE bit to 1. This bit allows flushing of the I2C Data Register when a Not Acknowledge interrupt is received after the data has been sent to the I2C Data Register. Reading this bit always returns 0.

**FILTEN—I2C Signal Filter Enable**
This bit enables low-pass digital filters on the SDA and SCL input signals. These filters reject any input pulse with periods less than a full system clock cycle. The filters introduce a 3-system clock cycle latency on the inputs.
1 = low-pass filters are enabled.
0 = low-pass filters are disabled.

```
DBG ← 04H
DBG ← OCDCTL[7:0]
```

- **Read OCD Control Register (05H)**—The Read OCD Control Register command reads the value of the OCDCTL register.

```
DBG ← 05H
DBG → OCDCTL[7:0]
```

- **Write Program Counter (06H)**—The Write Program Counter command writes the data that follows to the eZ8 CPU's Program Counter. If the device is not in DEBUG mode or if the Read Protect Option Bit is enabled, the Program Counter values are discarded.

```
DBG ← 06H
DBG ← ProgramCounter[15:8]
DBG ← ProgramCounter[7:0]
```

- **Read Program Counter (07H)**—The Read Program Counter command reads the value in the eZ8 CPU's Program Counter. If the device is not in DEBUG mode or if the Read Protect Option Bit is enabled, this command returns FFFFH.

```
DBG ← 07H
DBG → ProgramCounter[15:8]
DBG → ProgramCounter[7:0]
```

- **Write Register (08H)**—The Write Register command writes data to the Register File. Data can be written 1-256 bytes at a time (256 bytes can be written by setting size to zero). If the device is not in DEBUG mode, the address and data values are discarded. If the Read Protect Option Bit is enabled, then only writes to the Flash Control Registers are allowed and all other register write data values are discarded.

```
DBG ← 08H
DBG ← {4'h0,Register Address[11:8]}
DBG ← Register Address[7:0]
DBG ← Size[7:0]
DBG ← 1-256 data bytes
```

- **Read Register (09H)**—The Read Register command reads data from the Register File. Data can be read 1-256 bytes at a time (256 bytes can be read by setting size to zero). Reading peripheral control registers through the OCD does not effect peripheral operation. For example, register bits that are normally cleared upon a read operation will not be effected (WDTSTAT register is affected by OCD read register operation). If the device is not in DEBUG mode or if the Read Protect Option Bit is enabled, this command returns FFH for all the data values.

```
DBG ← 09H
DBG ← {4'h0,Register Address[11:8]
DBG ← Register Address[7:0]
DBG ← Size[7:0]
DBG → 1-256 data bytes
```

## I²C Timing

Figure 53 and Table 110 provide timing information for I²C pins.

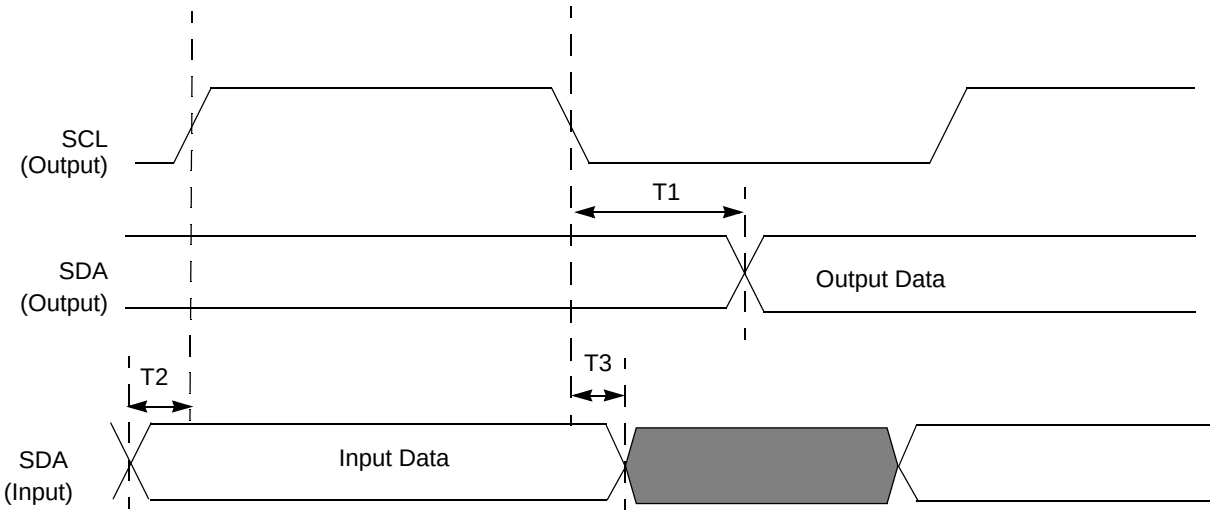



**Figure 53. I²C Timing**

**Table 110. I²C Timing**

| | | Delay (ns) | |
|---|---|---|---|
| **Parameter** | **Abbreviation** | **Minimum** | **Maximum** |
| **I²C** | | | |
| $T_1$ | SCL Fall to SDA output delay | | SCL period/4 |
| $T_2$ | SDA Input to SCL rising edge Setup Time | 0 | |
| $T_3$ | SDA Input to SCL falling edge Hold Time | 0 | |

## eZ8 CPU Instruction Classes

eZ8 CPU instructions are divided functionally into the following groups:

- Arithmetic
- Bit Manipulation
- Block Transfer
- CPU Control
- Load
- Logical
- Program Control
- Rotate and Shift

Tables 118 through Table 125 on page 218 contain the instructions belonging to each group and the number of operands required for each instruction. Some instructions appear in more than one table as these instruction can be considered as a subset of more than one category. Within these tables, the source operand is identified as ’src’, the destination operand is ’dst’ and a condition code is ’cc’.

**Table 118. Arithmetic Instructions**

| Mnemonic | Operands | Instruction |
|---|---|---|
| ADC | dst, src | Add with Carry |
| ADCX | dst, src | Add with Carry using Extended Addressing |
| ADD | dst, src | Add |
| ADDX | dst, src | Add using Extended Addressing |
| CP | dst, src | Compare |
| CPC | dst, src | Compare with Carry |
| CPCX | dst, src | Compare with Carry using Extended Addressing |
| CPX | dst, src | Compare using Extended Addressing |
| DA | dst | Decimal Adjust |
| DEC | dst | Decrement |
| DECW | dst | Decrement Word |
| INC | dst | Increment |
| INCW | dst | Increment Word |
| MULT | dst | Multiply |

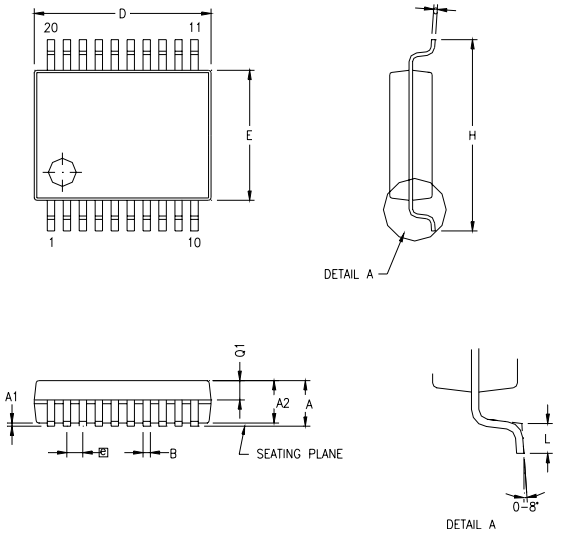
**Table 126. eZ8 CPU Instruction Summary (Continued)**

| Assembly Mnemonic | Symbolic Operation | Address Mode dst | Address Mode src | Opcode(s) (Hex) | Flags C | Z | S | V | D | H | Fetch Cycles | Instr. Cycles |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TM dst, src | dst AND src | r | r | 72 | - | * | * | 0 | - | - | 2 | 3 |
| | | r | Ir | 73 | | | | | | | 2 | 4 |
| | | R | R | 74 | | | | | | | 3 | 3 |
| | | R | IR | 75 | | | | | | | 3 | 4 |
| | | R | IM | 76 | | | | | | | 3 | 3 |
| | | IR | IM | 77 | | | | | | | 3 | 4 |
| TMX dst, src | dst AND src | ER | ER | 78 | - | * | * | 0 | - | - | 4 | 3 |
| | | ER | IM | 79 | | | | | | | 4 | 3 |
| TRAP Vector | SP ← SP – 2<br>@SP ← PC<br>SP ← SP – 1<br>@SP ← FLAGS<br>PC ← @Vector | | Vector | F2 | - | - | - | - | - | - | 2 | 6 |
| WDT | | | | 5F | - | - | - | - | - | - | 1 | 2 |
| XOR dst, src | dst ← dst XOR src | r | r | B2 | - | * | * | 0 | - | - | 2 | 3 |
| | | r | Ir | B3 | | | | | | | 2 | 4 |
| | | R | R | B4 | | | | | | | 3 | 3 |
| | | R | IR | B5 | | | | | | | 3 | 4 |
| | | R | IM | B6 | | | | | | | 3 | 3 |
| | | IR | IM | B7 | | | | | | | 3 | 4 |
| XORX dst, src | dst ← dst XOR src | ER | ER | B8 | - | * | * | 0 | - | - | 4 | 3 |
| | | ER | IM | B9 | | | | | | | 4 | 3 |

Flags Notation: * = Value is a function of the result of the operation.
- = Unaffected
X = Undefined
0 = Reset to 0
1 = Set to 1

# Packaging

Figure 60 displays the 20-pin SSOP package available for Z8 Encore! XP® F0822 Series devices.

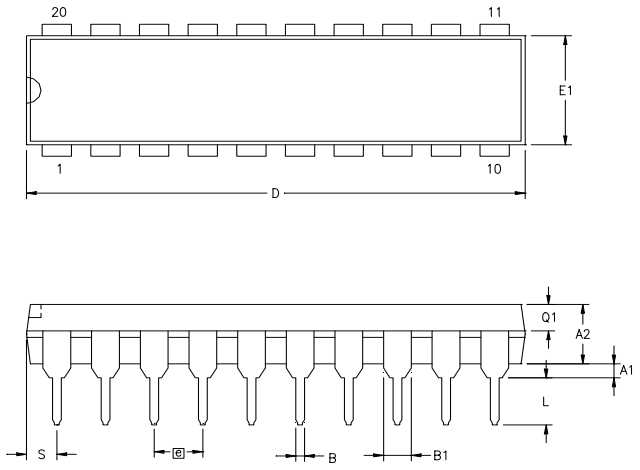



| SYMBOL | MILLIMETER | | | INCH | | |
|---|---|---|---|---|---|---|
| | MIN | NOM | MAX | MIN | NOM | MAX |
| A | 1.73 | 1.85 | 1.98 | 0.068 | 0.073 | 0.078 |
| A1 | 0.05 | 0.13 | 0.21 | 0.002 | 0.005 | 0.008 |
| A2 | 1.68 | 1.73 | 1.83 | 0.066 | 0.068 | 0.072 |
| B | 0.25 | 0.30 | 0.38 | 0.010 | 0.012 | 0.015 |
| C | 0.13 | 0.15 | 0.22 | 0.005 | 0.006 | 0.009 |
| D | 7.07 | 7.20 | 7.33 | 0.278 | 0.283 | 0.289 |
| E | 5.20 | 5.30 | 5.38 | 0.205 | 0.209 | 0.212 |
| e | 0.65 BSC | | | 0.0256 BSC | | |
| H | 7.65 | 7.80 | 7.90 | 0.301 | 0.307 | 0.311 |
| L | 0.56 | 0.75 | 0.94 | 0.022 | 0.030 | 0.037 |
| Q1 | 0.74 | 0.78 | 0.82 | 0.029 | 0.031 | 0.032 |

CONTROLLING DIMENSIONS : MM
LEADS ARE COPLANAR WITHIN .004 INCH.

**Figure 60. 20-Pin Small Shrink Outline Package (SSOP)**

Figure 61 displays the 20-pin PDIP package available for Z8 Encore! XP F0822 Series devices.




| SYMBOL | MILLIMETER | | INCH | |
|---|---|---|---|---|
| | MIN | MAX | MIN | MAX |
| A1 | 0.38 | 0.81 | .015 | .032 |
| A2 | 3.25 | 3.68 | .128 | .145 |
| B | 0.41 | 0.51 | .016 | .020 |
| B1 | 1.47 | 1.57 | .058 | .062 |
| C | 0.20 | 0.30 | .008 | .012 |
| D | 25.65 | 26.16 | 1.010 | 1.030 |
| E | 7.49 | 8.26 | .295 | .325 |
| E1 | 6.10 | 6.65 | .240 | .262 |
| e | 2.54 BSC | | .100 BSC | |
| eA | 7.87 | 9.14 | .310 | .360 |
| L | 3.18 | 3.43 | .125 | .135 |
| Q1 | 1.42 | 1.65 | .056 | .065 |
| S | 1.52 | 1.65 | .060 | .065 |

CONTROLLING DIMENSIONS : INCH

**Figure 61. 20-Pin Plastic Dual-Inline Package (PDIP)**

compare 82
compare - extended addressing 214
compare mode 82
compare with carry 214
compare with carry - extended addressing 214
complement 217
complement carry flag 215, 216
condition code 211
continuous conversion (ADC) 148
continuous mode 81
control register definition, UART 100
control register, I2C 141
counter modes 81
CP 214
CPC 214
CPCX 214
CPU and peripheral overview 3
CPU control instructions 216
CPX 214
Customer Feedback Form 251
customer feedback form 240
Customer Information 251

## D

DA 211, 214
data register, I2C 139
DC characteristics 187
debugger, on-chip 171
DEC 214
decimal adjust 214
decrement 214
  and jump non-zero 217
  word 214
DECW 214
destination operand 212
device, port availability 47
DI 216
direct address 211
disable interrupts 216
DJNZ 217
DMA controller 5
dst 212

## E

EI 216
electrical characteristics 185
  ADC 199
  flash memory and timing 196
  GPIO input data sample timing 200
  watch-dog timer 197
enable interrupt 216
ER 211
extended addressing register 211
external pin reset 43
external RC oscillator 196
eZ8
  features 3
eZ8 CPU features 3
eZ8 CPU instruction classes 214
eZ8 CPU instruction notation 210
eZ8 CPU instruction set 209
eZ8 CPU instruction summary 218

## F

FCTL register 159
features, Z8 Encore! 1
first opcode map 231
FLAGS 212
flags register 212
flash
  controller 4
  option bit address space 163
  option bit configuration - reset 163
  program memory address 0001H 165
flash memory
  arrangement 154
  byte programming 157
  code protection 156
  control register definitions 159
  controller bypass 158
  electrical characteristics and timing 196
  flash control register 159
  flash status register 160
  frequency high and low byte registers 161
  mass erase 158
  operation 155