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	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	8KB (8K 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	-40°C ~ 105°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/z8f0812sj020eg

Program Memory

The eZ8 CPU supports 64KB of Program memory address space. Z8 Encore! XP® F0822 Series contain 4KB to 8KB on-chip Flash in the Program memory address space, depending on the device. Reading from Program memory addresses outside the available Flash addresses returns FFH. Writing to unimplemented Program memory addresses produces no effect. Table 5 describes the Program memory Maps for Z8 Encore! XP® F0822 Series devices.

Table 5. Z8 Encore! XP® F0822 Series Program Memory Maps

Program Memory Address (Hex)	Function
Z8F082x and Z8F081x Products	
0000-0001	Option Bits
0002-0003	Reset Vector
0004-0005	WDT Interrupt Vector
0006-0007	Illegal Instruction Trap
0008-0037	Interrupt Vectors*
0038-1FFF	Program Memory
Z8F042x and Z8F041x Products	
0000-0001	Option Bits
0002-0003	Reset Vector
0004-0005	WDT Interrupt Vector
0006-0007	Illegal Instruction Trap
0008-0037	Interrupt Vectors*
0038-0FFF	Program Memory

Note: *See Table 24 on page 41 for a list of the interrupt vectors.

Data Memory

Z8 Encore! XP® F0822 Series does not use the eZ8 CPU's 64KB Data Memory address space.

Information Area

Table 6 describes the Z8 Encore! XP® F0822 Series Information Area. This 512-byte Information Area is accessed by setting bit 7 of the Page Select Register to 1. When access is enabled, the Information Area is mapped into the Program memory and overlays the 512 bytes at addresses FE00H to FFFFH. When the Information Area access is enabled, all

Port A–C High Drive Enable Subregisters

The Port A–C High Drive Enable Subregister, shown in Table 19, is accessed through the Port A–C Control Register by writing 04H to the Port A–C Address Register. Setting the bits in the Port A–C High Drive Enable subregisters to 1 configures the specified port pins for high-output current drive operation. The Port A–C High Drive Enable Subregister affects the pins directly and, as a result, alternate functions are also affected.

Table 19. Port A–C High Drive Enable Subregisters

Bit	7	6	5	4	3	2	1	0
Field	PHDE7	PHDE6	PHDE5	PHDE4	PHDE3	PHDE2	PHDE1	PHDE0
RESET	0							
R/W	R/W							
Address	See footnote.							

Note: If 04H is written to the Port A–C Address Register, then it is accessible via the Port A–C Control Register.

Bit	Description
[7:0] PHDEx	**Port High Drive Enabled** 0 = The port pin is configured for standard-output current drive. 1 = The port pin is configured for high-output current drive.

Note: x indicates register bits in the range [7:0].

Port A–C Output Data Register

The Port A–C Output Data Register, shown in Table 23, controls the output data to the pins.

Table 23. Port A–C Output Data Register (P*x*OUT)

Bit	7	6	5	4	3	2	1	0
Field	POUT7	POUT6	POUT5	POUT4	POUT3	POUT2	POUT1	POUT0
RESET	0							
R/W	R/W							
Address	FD3H, FD7H, FDBH							

Bit	Description
[7:0] PxOUT	**Port Output Data** These bits contain the data to be driven to the port pins. The values are only driven if the corresponding pin is configured as an output and the pin is not configured for alternate function operation. 0 = Drive a logical 0 (Low). 1 = Drive a logical 1 (High). This High value is not driven if the drain has been disabled by setting the corresponding Port Output Control Register bit to 1.

Note: x indicates register bits in the range [7:0].

Architecture

Figure 9 displays a block diagram of the interrupt controller.

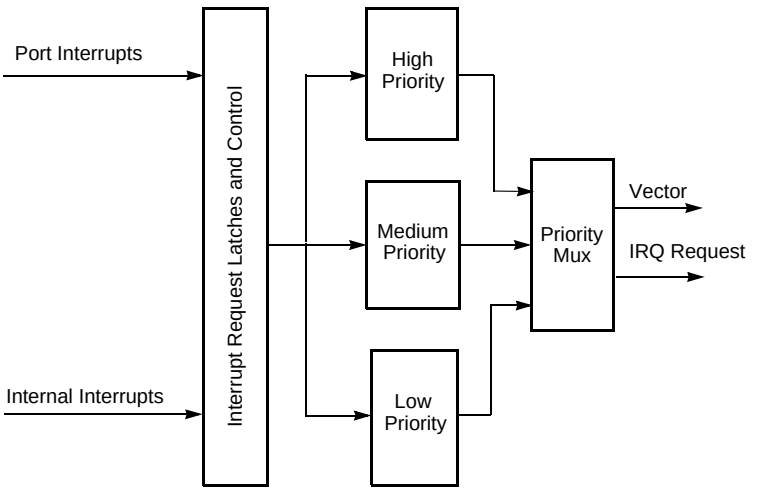

Figure 9. Interrupt Controller Block Diagram

Operation

This section describes the operational aspects of the following functions.

Master Interrupt Enable: see page 42

Interrupt Vectors and Priority: see page 43

Interrupt Assertion: see page 43

Software Interrupt Assertion: see page 44

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 Enable Interrupt (EI) instruction
- Execution of an Return from Interrupt (IRET) instruction

Interrupt Control Register Definitions

For all interrupts other than the WDT interrupt, the Interrupt Control registers enable individual interrupts, set interrupt priorities, and indicate interrupt requests.

Interrupt Request 0 Register

The Interrupt Request 0 (IRQ0) Register, shown in Table 25, stores the interrupt requests for both vectored and polled interrupts. When a request is presented to the interrupt controller, the corresponding bit in the IRQ0 Register becomes 1. If interrupts are globally enabled (vectored interrupts), the interrupt controller passes an interrupt request to the eZ8 CPU. If interrupts are globally disabled (polled interrupts), the eZ8 CPU reads the IRQ0 Register to determine if any interrupt requests are pending.

Table 25. Interrupt Request 0 Register (IRQ0)

Bit	7	6	5	4	3	2	1	0
Field	Reserved	T1I	T0I	U0RXI	U0TXI	I2CI	SPII	ADCI
RESET	0							
R/W	R/W							
Address	FC0H							

Bit	Description
[7]	**Reserved** This bit is reserved and must be programmed to 0.
[6] T1I	**Timer 1 Interrupt Request** 0 = No interrupt request is pending for Timer 1. 1 = An interrupt request from Timer 1 is awaiting service.
[5] T0I	**Timer 0 Interrupt Request** 0 = No interrupt request is pending for Timer 0. 1 = An interrupt request from Timer 0 is awaiting service.
[4] U0RXI	**UART 0 Receiver Interrupt Request** 0 = No interrupt request is pending for the UART 0 receiver. 1 = An interrupt request from the UART 0 receiver is awaiting service.
[3] U0TXI	**UART 0 Transmitter Interrupt Request** 0 = No interrupt request is pending for the UART 0 transmitter. 1 = An interrupt request from the UART 0 transmitter is awaiting service.
[2] I2CI	**I^2C Interrupt Request** 0 = No interrupt request is pending for the I^2C. 1 = An interrupt request from the I^2C is awaiting service.

SPI Interrupts

When SPI interrupts are enabled, the SPI generates an interrupt after character transmission/reception completes in both Master and Slave modes. A character is defined to be 1 through 8 bits by the NUMBITS field in the SPI Mode Register. In SLAVE Mode it is not necessary for $\overline{SS}$ to deassert between characters to generate the interrupt. The SPI in SLAVE Mode also generates an interrupt if the $\overline{SS}$ signal deasserts prior to transfer of all of the bits in a character (see the previous paragraph). Writing a 1 to the IRQ bit in the SPI Status Register clears the pending SPI interrupt request. The IRQ bit must be cleared to 0 by the ISR to generate future interrupts. To start the transfer process, an SPI interrupt can be forced by software writing a 1 to the STR bit in the SPICTL Register.

If the SPI is disabled, an SPI interrupt can be generated by a BRG time-out. This timer function must be enabled by setting the BIRQ bit in the SPICTL Register. This BRG time-out does not set the IRQ bit in the SPISTAT Register, just the SPI interrupt bit in the interrupt controller.

SPI Baud Rate Generator

In SPI MASTER Mode, the BRG creates a lower frequency serial clock (SCK) for data transmission synchronization between the Master and the external Slave. The input to the BRG is the system clock. The SPI Baud Rate High and Low Byte registers combine to form a 16-bit reload value, BRG[15:0], for the SPI Baud Rate Generator. The SPI baud rate is calculated using the following equation:

$$\text{SPI Baud Rate (bits/s)} = \frac{\text{System Clock Frequency (Hz)}}{2 \times \text{BRG[15:0]}}$$

The minimum baud rate is obtained by setting BRG[15:0] to 0000H for a clock divisor value of (2 x 65536 = 131072).

When the SPI is disabled, BRG functions as a basic 16-bit timer with interrupt upon time-out. Observe the following procedure to configure BRG as a timer with interrupt upon time-out:

1. Disable the SPI by clearing the SPIEN bit in the SPI Control Register to 0.
2. Load the appropriate 16-bit count value into the SPI Baud Rate High and Low Byte registers.
3. Enable BRG timer function and associated interrupt by setting the BIRQ bit in the SPI Control Register to 1.

When configured as a general-purpose timer, the interrupt interval is calculated using the following equation:

Interrupt Interval (s) = System Clock Period (s) ×BRG[15:0]]

Start and Stop Conditions

The Master (I^2C) drives all Start and Stop signals and initiates all transactions. To start a transaction, the I^2C Controller generates a start condition by pulling the SDA signal Low while SCL is High. To complete a transaction, the I^2C Controller generates a Stop condition by creating a Low-to-High transition of the SDA signal while the SCL signal is High. The start and stop bits in the I^2C Control Register control the sending of start and stop conditions. A Master is also allowed to end one transaction and begin a new one by issuing a restart. This restart issuance is accomplished by setting the start bit at the end of a transaction rather than setting the stop bit.

Note: The start condition is not sent until the start bit is set and data has been written to the I^2C Data Register.

Master Write and Read Transactions

The following sections provide Zilog's recommended procedure for performing I^2C write and read transactions from the I^2C Controller (Master) to slave I^2C devices. In general, software should rely on the TDRE, RDRF and NCKI bits of the status register (these bits generate interrupts) to initiate software actions. When using interrupts or DMA, the TXI bit is set to start each transaction and cleared at the end of each transaction to eliminate a *trailing* transmit interrupt.

Caution: Caution should be used in using the ACK status bit within a transaction because it is difficult for software to tell when it is updated by hardware.

When writing data to a slave, the I^2C pauses at the beginning of the Acknowledge cycle if the data register has not been written with the next value to be sent (TDRE bit in the I^2C Status Register equal to 1). In this scenario where software is not keeping up with the I^2C bus (TDRE asserted longer than one byte time), the Acknowledge clock cycle for byte *n* is delayed until the data register is written with byte *n+1*, and appears to be grouped with the data clock cycles for byte *n+1*. If either the start or stop bit is set, the I^2C does not pause prior to the Acknowledge cycle because no additional data is sent.

When a Not Acknowledge condition is received during a write (either during the address or data phases), the I^2C Controller generates the Not Acknowledge interrupt (NCKI = 1) and pause until either the stop or start bit is set. Unless the Not Acknowledge was received on the last byte, the data register will already have been written with the next address or data byte to send. In this case the FLUSH bit of the control register should be set at the same time the stop or start bit is set to remove the stale transmit data and enable subsequent transmit interrupts.

12. The I2C Controller loads the contents of the I2C Shift Register with the contents of the I2C Data Register.
13. The I2C Controller shifts the data out of using the SDA signal. After the first bit is sent, the transmit interrupt is asserted.
14. If more bytes remain to be sent, return to Step 9.
15. Software responds by setting the stop bit of the I2C Control Register (or start bit to initiate a new transaction). In the STOP case, software clears the TXI bit of the I2C Control Register at the same time.
16. The I2C Controller completes transmission of the data on the SDA signal.
17. The slave can either Acknowledge or Not Acknowledge the last byte. Because either the stop or start bit is already set, the NCKI interrupt does not occur.
18. The I2C Controller sends the stop (or restart) condition to the I2C bus. The stop or start bit is cleared.

Address-Only Transaction with a 10-Bit Address

In situations in which software must determine if a slave with a 10-bit address is responding without sending or receiving data, a transaction is performed which only consists of an address phase. Figure 28 displays this *address only* transaction to determine if a slave with a 10-bit address will acknowledge.

As an example, this transaction is used after a write has been executed to an EEPROM to determine when the EEPROM completes its internal write operation and is again responding to I2C transactions. If the slave does not acknowledge, the transaction is repeated until the slave is able to acknowledge.

S	Slave Address 1st Seven Bits	W = 0	A/A	Slave Address 2nd Byte	A/A

Figure 28. 10-Bit Address Only Transaction Format

Observe the following procedure for an address-only transaction to a 10-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 interrupt by writing the first slave address byte. The least-significant bit must be 0 for the write operation.
5. Software asserts the start bit of the I2C Control Register.

Observe the following procedure to setup the Flash Sector Protect Register from user code:

1. Write 00H to the Flash Control Register to reset the Flash Controller.
2. Write 5EH to the Flash Control Register to select the Flash Sector Protect Register.
3. Read and/or write the Flash Sector Protect Register which is now at Register File address FF9H.
4. Write 00H to the Flash Control Register to return the Flash Controller to its reset state.

Flash Write Protection Option Bit

The Flash Write Protect option bit can block all program and erase operations from user code. For more information, see the Option Bits chapter on page 155.

Byte Programming

When the Flash Controller is unlocked, writes to Flash memory from user code programs a byte into the Flash if the address is located in the unlocked page. An erased Flash byte contains all 1s (FFH). The programming operation is used to change bits from 1 to 0. To change a Flash bit (or multiple bits) from zero to one requires a Page Erase or Mass Erase operation.

Byte programming is accomplished using the eZ8 CPU's LDC or LDCI instructions. Refer to the eZ8 CPU Core User Manual (UM0128) for a description of the LDC and LDCI instructions.

While the Flash Controller programs the contents of Flash memory, the eZ8 CPU idles but the system clock and on-chip peripherals continue to operate. Interrupts that occur when a programming operation is in progress are serviced after the programming operation is complete. To exit programming mode and lock the Flash Controller, write 00H to the Flash Control Register.

User code cannot program Flash memory on a page that is located in a protected sector. When user code writes memory locations, only addresses located in the unlocked page are programmed. Memory writes outside of the unlocked page are ignored.

Caution: Each memory location must not be programmed more than twice before an erase occurs.

Observe the following procedure to program the Flash from user code:

1. Write 00H to the Flash Control Register to reset the Flash Controller.
2. Write the page of memory to be programmed to the Page Select Register.
3. Write the first unlock command 73H to the Flash Control Register.

```
DBG ← 12H
DBG ← 1-5 byte opcode
```

On-Chip Debugger Control Register Definitions

This section describes the features of the On-Chip Debugger Control and Status registers.

OCD Control Register

The OCD Control Register, shown in Table 94, controls the state of the OCD. This register enters or exits DEBUG Mode and enables the BRK instruction. It can also reset the Z8 Encore! XP® F0822 Series device.

A *reset and stop* function can be achieved by writing 81H to this register. A *reset and go* function can be achieved by writing 41H to this register. If the device is in DEBUG Mode, a *run* function can be implemented by writing 40H to this register.

Table 94. OCD Control Register (OCDCTL)

Bit	7	6	5	4	3	2	1	0
Field	DBGMODE	BRKEN	DBGACK	BRKLOOP	BRKPC	BRKZRO	Reserved	RST
RESET	0							
R/W	R/W			R				R/W

Bit	Description
[7] DBGMODE	**Debug Mode** Setting this bit to 1 causes the device to enter DEBUG Mode. When in DEBUG Mode, the eZ8 CPU stops fetching new instructions. Clearing this bit causes the eZ8 CPU to start running again. This bit is automatically set when a BRK instruction is decoded and breakpoints are enabled. If the Read Protect option bit is enabled, this bit can only be cleared by resetting the device, it cannot be written to 0. 0 = The Z8 Encore! XP® F0822 Series device is operating in NORMAL Mode. 1 = The Z8 Encore! XP® F0822 Series device is in DEBUG Mode.
[6] BRKEN	**Breakpoint Enable** This bit controls the behavior of the BRK instruction (Op Code 00H). By default, breakpoints are disabled and the BRK instruction behaves like an NOP instruction. If this bit is set to 1 and a BRK instruction is decoded, the OCD takes action dependent upon the BRKLOOP bit. 0 = BRK instruction is disabled. 1 = BRK instruction is enabled.
[5] DBGACK	**Debug Acknowledge** This bit enables the debug acknowledge feature. If this bit is set to 1, then the OCD sends an Debug Acknowledge character (FFH) to the host when a Breakpoint occurs. 0 = Debug Acknowledge is disabled. 1 = Debug Acknowledge is enabled.

Table 98. DC Characteristics (Continued)

		T_A = –40°C to 105°C				
Symbol	**Parameter**	**Minimum**	**Typical**	**Maximum**	**Units**	**Conditions**
I_{TL}	Tri-State Leakage Current	–5	–	+5	µA	V_{DD} = 3.6 V
C_{PAD}	GPIO Port Pad Capacitance	–	8.0[2]	–	pF	
C_{XIN}	X_{IN} Pad Capacitance	–	8.0[2]	–	pF	
C_{XOUT}	X_{OUT} Pad Capacitance	–	9.5[2]	–	pF	
I_{PU1}	Weak Pull-up Current	9	20	50	µA	V_{DD} = 2.7–3.6V T_A = 0°C to +70°C
I_{PU2}	Weak Pull-up Current	7	20	75	µA	V_{DD} = 2.7–3.6V T_A = –40°C to +105°C

Note: [1] This condition excludes all pins that have on-chip pull-ups, when driven Low.
Note: [2] These values are provided for design guidance only and are not tested in production.

Figure 41 displays the typical active mode current consumption while operating at 25ºC, 3.3V, plotted opposite the system clock frequency. All GPIO pins are configured as outputs and driven High.

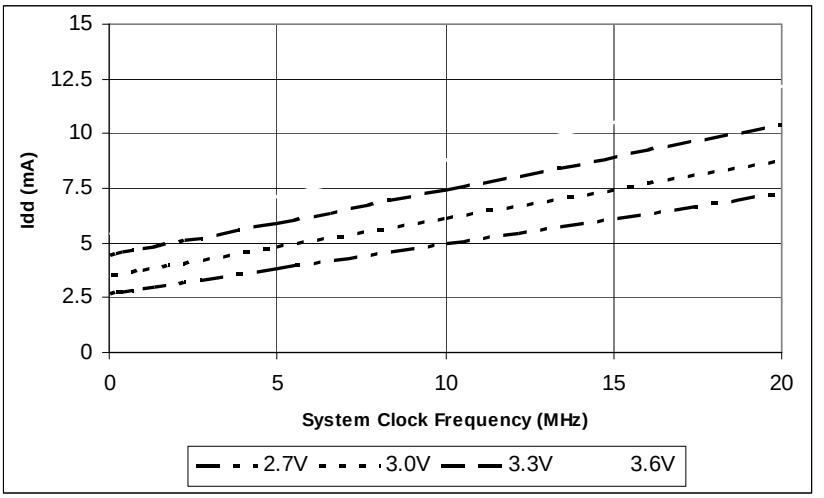

Figure 41. Typical Active Mode I_{DD} vs. System Clock Frequency

General Purpose I/O Port Output Timing

Figure 49 and Table 107 provide timing information for GPIO port pins.

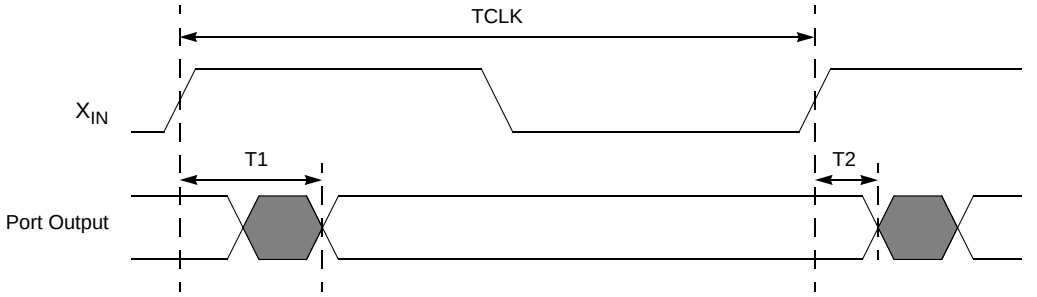

Figure 49. GPIO Port Output Timing

Table 107. GPIO Port Output Timing

		Delay (ns)	
Parameter	**Abbreviation**	**Minimum**	**Maximum**
GPIO Port Pins			
T_1	X_{IN} Rise to Port Output Valid Delay	–	15
T_2	X_{IN} Rise to Port Output Hold Time	2	–

SPI MASTER Mode Timing

Figure 51 and Table 109 provide timing information for SPI MASTER Mode pins. Timing is shown with SCK rising edge used to source MOSI output data, SCK falling edge used to sample MISO input data. Timing on the SS output pin(s) is controlled by software.

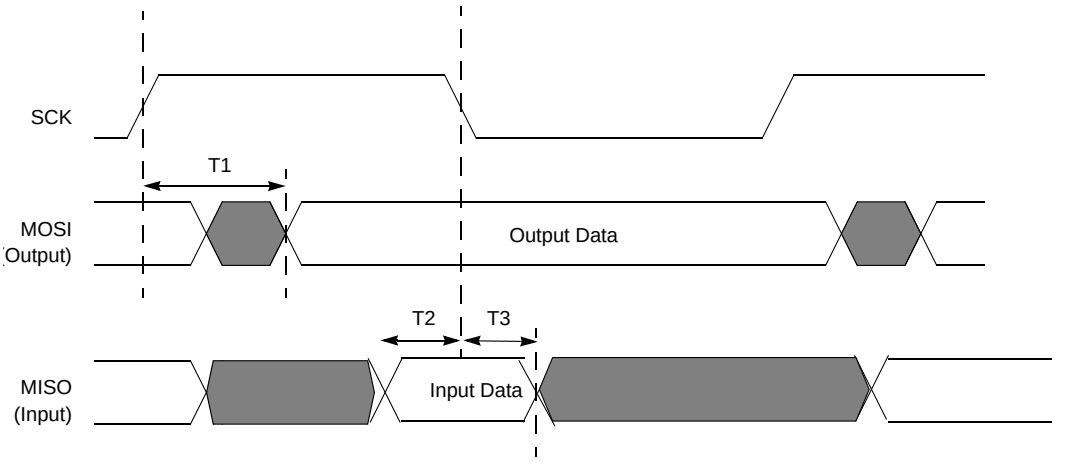

Figure 51. SPI MASTER Mode Timing

Table 109. SPI MASTER Mode Timing

		Delay (ns)	
Parameter	**Abbreviation**	**Minimum**	**Maximum**
SPI MASTER			
T_1	SCK Rise to MOSI output Valid Delay	–5	+5
T_2	MISO input to SCK (receive edge) Setup Time	20	
T_3	MISO input to SCK (receive edge) Hold Time	0	

SPI SLAVE Mode Timing

Figure 52 and Table 110 provide timing information for the SPI SLAVE Mode pins. Timing is shown with SCK rising edge used to source MISO output data, SCK falling edge used to sample MOSI input data.

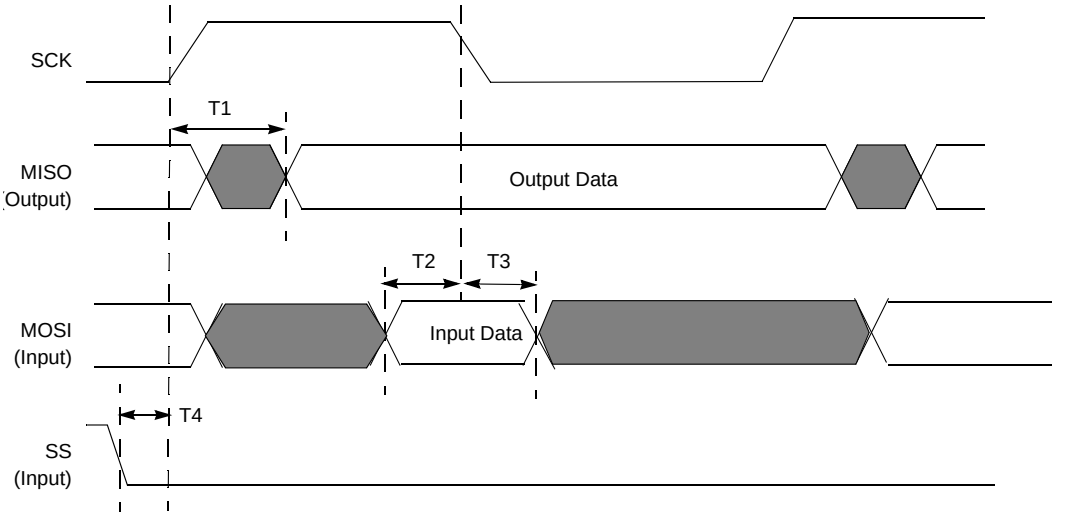

Figure 52. SPI SLAVE Mode Timing

Table 110. SPI SLAVE Mode Timing

		Delay (ns)	
Parameter	**Abbreviation**	**Minimum**	**Maximum**
SPI SLAVE			
T_1	SCK (transmit edge) to MISO output Valid Delay	2 * X_{IN} period	3 * X_{IN} period + 20ns
T_2	MOSI input to SCK (receive edge) Setup Time	0	
T_3	MOSI input to SCK (receive edge) Hold Time	3 * X_{IN} period	
T_4	SS input assertion to SCK setup	1 * X_{IN} period	

Table 115. Assembly Language Syntax Example 2

Assembly Language Code	ADD	43H,	R8	(ADD dst, src)
Object Code	04	E8	43	(OPC src, dst)

The register file size varies, depending on device type. See the device-specific Z8 Encore! XP Product Specification to determine the exact register file range available.

eZ8 CPU Instruction Notation

In the eZ8 CPU Instruction Summary and Description sections, the operands, condition codes, status flags, and address modes are represented by a notational shorthand that is described in Table 116.

Table 116. Notational Shorthand

Notation	Description	Operand	Range
b	Bit	b	b represents a value from 0 to 7 (000B to 111B).
cc	Condition Code	—	See the Condition Codes overview in the eZ8 CPU User Manual.
DA	Direct Address	Addrs	Addrs. represents a number in the range of 0000H to FFFFH
ER	Extended Addressing Register	Reg	Reg. represents a number in the range of 000H to FFFH
IM	Immediate Data	#Data	Data is a number between 00H to FFH
Ir	Indirect Working Register	@Rn	n = 0–15
IR	Indirect Register	@Reg	Reg. represents a number in the range of 00H to FFH
Irr	Indirect Working Register Pair	@RRp	p = 0, 2, 4, 6, 8, 10, 12, or 14
IRR	Indirect Register Pair	@Reg	*Reg* represents an even number in the range 00H to FEH
p	Polarity	p	Polarity is a single bit binary value of either 0B or 1B.
r	Working Register	Rn	n = 0 – 15
R	Register	Reg	*Reg* represents a number in the range of 00H to FFH

Table 129. Z8 Encore! XP F0830 Series Ordering Matrix

Part Number	Flash	RAM	I/O Lines	Interrupts	16-Bit Timers w/PWM	10-Bit A/D Channels	I^2C	SPI	UARTs with IrDA	Description
Z8F04xx with 4KB Flash, 10-Bit Analog-to-Digital Converter										
Standard Temperature: 0°C to 70°C										
Z8F0421HH020SG	4KB	1KB	11	16	2	2	1	0	1	SSOP 20-pin package
Z8F0421PH020SG	4KB	1KB	11	16	2	2	1	0	1	PDIP 20-pin package
Z8F0422SJ020SG	4KB	1KB	19	19	2	5	1	1	1	SOIC 28-pin package
Z8F0422PJ020SG	4KB	1KB	19	19	2	5	1	1	1	PDIP 28-pin package
Extended Temperature: –40°C to 105°C										
Z8F0421HH020EG	4KB	1KB	11	16	2	2	1	0	1	SSOP 20-pin package
Z8F0421PH020EG	4KB	1KB	11	16	2	2	1	0	1	PDIP 20-pin package
Z8F0422SJ020EG	4KB	1KB	19	19	2	5	1	1	1	SOIC 28-pin package
Z8F0422PJ020EG	4KB	1KB	19	19	2	5	1	1	1	PDIP 28-pin package
Z8F04xx with 4KB Flash										
Standard Temperature: 0°C to 70°C										
Z8F0411HH020SG	4KB	1KB	11	16	2	0	1	0	1	SSOP 20-pin package
Z8F0411PH020SG	4KB	1KB	11	16	2	0	1	0	1	PDIP 20-pin package
Z8F0412SJ020SG	4KB	1KB	19	19	2	0	1	1	1	SOIC 28-pin package
Z8F0412PJ020SG	4KB	1KB	19	19	2	0	1	1	1	PDIP 28-pin package
Extended Temperature: –40°C to 105°C										
Z8F0411HH020EG	4KB	1KB	11	16	2	0	1	0	1	SSOP 20-pin package
Z8F0411PH020EG	4KB	1KB	11	16	2	0	1	0	1	PDIP 20-pin package
Z8F0412SJ020EG	4KB	1KB	19	19	2	0	1	1	1	SOIC 28-pin package
Z8F0412PJ020EG	4KB	1KB	19	19	2	0	1	1	1	PDIP 28-pin package
Z8F08200100KITG	Development Kit (20- and 28-pin)									
ZUSBSC00100ZACG	USB Smart Cable Accessory Kit									
ZUSBOPTSC01ZACG	Opto-Isolated USB Smart Cable Accessory Kit									

Visit the Zilog website at http://www.zilog.com for ordering information about Z8 Encore! XP® F0822 Series development tools and accessories.

Hex Address: FD2

Table 184. Port A–C Input Data Registers (PxIN)

Bit	7	6	5	4	3	2	1	0
Field	PIN7	PIN6	PIN5	PIN4	PIN3	PIN2	PIN1	PIN0
RESET	X							
R/W	R							
Address	FD2H, FD6H, FDAH							

Hex Address: FD3

Table 185. Port A–C Output Data Register (PxOUT)

Bit	7	6	5	4	3	2	1	0
Field	POUT7	POUT6	POUT5	POUT4	POUT3	POUT2	POUT1	POUT0
RESET	0							
R/W	R/W							
Address	FD3H, FD7H, FDBH							

Hex Address: FD4

Table 186. Port A–C GPIO Address Registers (PxADDR)

Bit	7	6	5	4	3	2	1	0
Field	PADDR[7:0]							
RESET	00H							
R/W	R/W							
Address	FD0H, FD4H, FD8H							

Hex Address: FD5

Table 187. Port A–C Control Registers (PxCTL)

Bit	7	6	5	4	3	2	1	0
Field	PCTL							
RESET	00H							
R/W	R/W							
Address	FD1H, FD5H, FD9H							

Hex Address: FDA

Table 192. Port A–C Input Data Registers (PxIN)

Bit	7	6	5	4	3	2	1	0
Field	PIN7	PIN6	PIN5	PIN4	PIN3	PIN2	PIN1	PIN0
RESET	X							
R/W	R							
Address	FD2H, FD6H, FDAH							

Hex Address: FDB

Table 193. Port A–C Output Data Register (PxOUT)

Bit	7	6	5	4	3	2	1	0
Field	POUT7	POUT6	POUT5	POUT4	POUT3	POUT2	POUT1	POUT0
RESET	0							
R/W	R/W							
Address	FD3H, FD7H, FDBH							

Hex Addresses: FDC–FEF

This address range is reserved.

Watchdog Timer Control Registers

For more information about the Watchdog Timer control registers, see the Watchdog Timer Control Register Definitions section on page 73.

Hex Address: FF0

Table 194. Watchdog Timer Control Register (WDTCTL)

Bit	7	6	5	4	3	2	1	0
Field	POR	STOP	WDT	EXT	Reserved			
RESET	See Table 49 on page 74.				0			
R/W	R							
Address	FF0H							