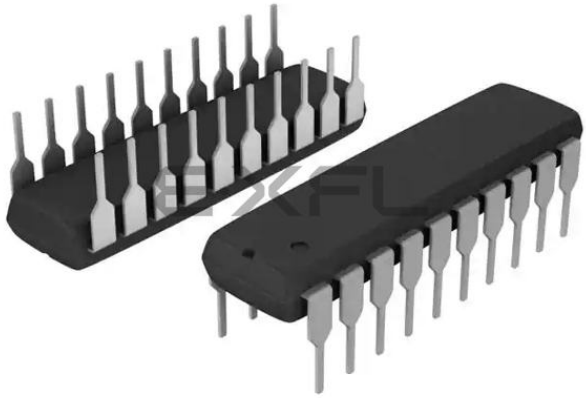


Welcome to E-XFL.COM

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, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	11
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	A/D 2x10b
Oscillator Type	Internal
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Through Hole
Package / Case	20-DIP (0.300", 7.62mm)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f0421ph020sc

Warning: DO NOT USE IN LIFE SUPPORT

LIFE SUPPORT POLICY

ZILOG'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS PRIOR WRITTEN APPROVAL OF THE PRESIDENT AND GENERAL COUNSEL OF ZILOG CORPORATION.

As used herein

Life support devices or systems are devices which (a) are intended for surgical implant into the body, or (b) support or sustain life and whose failure to perform when properly used in accordance with instructions for use provided in the labeling can be reasonably expected to result in a significant injury to the user. A critical component is any component in a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system or to affect its safety or effectiveness.

Document Disclaimer

 Information in this publication concerning the devices, applications, or technology described is intended to suggest possible uses and may be superseded. ZILOG, INC. DOES NOT ASSUME LIABILITY FOR OR PROVIDE A REPRESENTATION OF ACCURACY OF THE INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED IN THIS DOCUMENT. ZILOG ALSO DOES NOT ASSUME LIABILITY FOR INTELLECTUAL PROPERTY INFRINGEMENT RELATED IN ANY MANNER TO USE OF INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED HEREIN OR OTHERWISE. The information contained within this document has been verified according to the general principles of electrical and mechanical engineering.

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.

Table 7. Register File Address Map (Continued)

Address (Hex)	Register Description	Mnemonic	Reset (Hex)	Page No
FCE	Reserved	—	00	
FCF	Interrupt Control	IRQCTL	00	67
GPIO Port A				
FD0	Port A Address	PAADDR	00	50
FD1	Port A Control	PACTL	00	51
FD2	Port A Input Data	PAIN	XX	54
FD3	Port A Output Data	PAOUT	00	55
GPIO Port B				
FD4	Port B Address	PBADDR	00	50
FD5	Port B Control	PBCTL	00	51
FD6	Port B Input Data	PBIN	XX	54
FD7	Port B Output Data	PBOUT	00	55
GPIO Port C				
FD8	Port C Address	PCADDR	00	50
FD9	Port C Control	PCCTL	00	51
FDA	Port C Input Data	PCIN	XX	54
FDB	Port C Output Data	PCOUT	00	55
FDC-FEF	Reserved	—	XX	
Watchdog Timer (WDT)				
FF0	Watchdog Timer Control	WDTCTL	XXX00000b	86
FF1	Watchdog Timer Reload Upper Byte	WDTU	FF	87
FF2	Watchdog Timer Reload High Byte	WDTH	FF	87
FF3	Watchdog Timer Reload Low Byte	WDTL	FF	87
FF4-FF7	Reserved	—	XX	
Flash Memory Controller				
FF8	Flash Control	FCTL	00	159
FF8	Flash Status	FSTAT	00	160
FF9	Page Select	FPS	00	160
FF9 (if enabled)	Flash Sector Protect	FPROT	00	161
FFA	Flash Programming Frequency High Byte	FFREQH	00	161
FFB	Flash Programming Frequency Low Byte	FFREQL	00	161
Read-Only Memory				
FF8	Reserved	—	XX	
FF9	Page Select	RPS	00	160
FFA-FFB	Reserved	—	XX	
eZ8 CPU				

XX=Undefined

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

Table 29. IRQ0 Enable High Bit Register (IRQ0ENH)

BITS	7	6	5	4	3	2	1	0
FIELD	Reserved	T1ENH	T0ENH	U0RENH	U0TENH	I2CENH	SPIENH	ADCENH
RESET	0							
R/W	R/W							
ADDR	FC1H							

Reserved—Must be 0
T1ENH—Timer 1 Interrupt Request Enable High Bit
T0ENH—Timer 0 Interrupt Request Enable High Bit
U0RENH—UART 0 Receive Interrupt Request Enable High Bit
U0TENH—UART 0 Transmit Interrupt Request Enable High Bit
I2CENH—I^2C Interrupt Request Enable High Bit
SPIENH—SPI Interrupt Request Enable High Bit
ADCENH—ADC Interrupt Request Enable High Bit

Table 30. IRQ0 Enable Low Bit Register (IRQ0ENL)

BITS	7	6	5	4	3	2	1	0
FIELD	Reserved	T1ENL	T0ENL	U0RENL	U0TENL	I2CENL	SPIENL	ADCENL
RESET	0							
R/W	R/W							
ADDR	FC2H							

Reserved—Must be 0
T1ENL—Timer 1 Interrupt Request Enable Low Bit
T0ENL—Timer 0 Interrupt Request Enable Low Bit
U0RENL—UART 0 Receive Interrupt Request Enable Low Bit
U0TENL—UART 0 Transmit Interrupt Request Enable Low Bit
I2CENL—I^2C Interrupt Request Enable Low Bit
SPIENL—SPI Interrupt Request Enable Low Bit
ADCENL—ADC Interrupt Request Enable Low Bit

IRQ1 Enable High and Low Bit Registers

Table 31 describes the priority control for IRQ1. The IRQ1 Enable High and Low Bit Registers (Table 32 and Table 33) form a priority encoded enabling for interrupts in the Interrupt Request 1 Register. Priority is generated by setting bits in each register.

Timers

Z8 Encore! XP® F0822 Series products contain up to two 16-bit reloadable timers that can be used for timing, event counting, or generation of pulse-width modulated signals. The timer features include:

- 16-bit reload counter.
- Programmable prescaler with prescale values from 1 to 128.
- PWM output generation.
- Capture and compare capability.
- External input pin for timer input, clock gating, or capture signal. External input pin signal frequency is limited to a maximum of one-fourth the system clock frequency.
- Timer output pin.
- Timer interrupt.

In addition to the timers described in this chapter, the Baud Rate Generators for any unused UART, SPI, or I^2C peripherals can also be used to provide basic timing functionality. See the respective serial communication peripheral chapters for information on using the Baud Rate Generators as timers.

Architecture

Figure 10 displays the architecture of the timers.

Operation

The timers are 16-bit up-counters. Minimum time-out delay is set by loading the value 0001H into the Timer Reload High and Low Byte registers and setting the prescale value to 1. Maximum time-out delay is set by loading the value 0000H into the Timer Reload High and Low Byte registers and setting the prescale value to 128. If the Timer reaches FFFFH, the timer rolls over to 0000H and continues counting.

All three Watchdog Timer Reload Registers must be written in this order. There must be no other register writes between each of these operations. If a register write occurs, the lock state machine resets and no further writes occur unless the sequence is restarted. The value in the Watchdog Timer Reload Registers is loaded into the counter when the WDT is first enabled and every time a WDT instruction is executed.

Watchdog Timer Control Register Definitions

Watchdog Timer Control Register

The Watchdog Timer Control Register (WDTCTL), detailed in Table 48, is a Read-Only Register that indicates the source of the most recent Reset event, a Stop Mode Recovery event, and a WDT time-out. Reading this register resets the upper four bits to 0.

Writing the 55H, AAH unlock sequence to the Watchdog Timer Control Register (WDTCTL) address unlocks the three Watchdog Timer Reload Byte registers (WDTU, WDTH, and WDTL) to allow changes to the time-out period. These write operations to the WDTCTL address produce no effect on the bits in the WDTCTL. The locking mechanism prevents spurious writes to the Reload registers.

Table 48. Watchdog Timer Control Register (WDTCTL)

BITS	7	6	5	4	3	2	1	0
FIELD	POR	STOP	WDT	EXT	Reserved			
RESET	See descriptions below			0				
R/W	R							
ADDR	FF0H							

Reset or Stop Mode Recovery Event	POR	STOP	WDT	EXT
Power-On Reset	1	0	0	0
Reset through $\overline{\text{RESET}}$ pin assertion	0	0	0	1
Reset through WDT time-out	0	0	1	0
Reset through the OCD (OCTCTL[1] set to 1)	1	0	0	0
Reset from STOP Mode through the DBG Pin driven Low	1	0	0	0
Stop Mode Recovery through GPIO pin transition	0	1	0	0
Stop Mode Recovery through WDT time-out	0	1	1	0

POR—Power-On Reset Indicator
If this bit is set to 1, a POR event occurred. This bit is reset to 0, if a WDT time-out or Stop Mode Recovery occurs. This bit is also reset to 0, when the register is read.

Universal Asynchronous Receiver/Transmitter

The Universal Asynchronous Receiver/Transmitter (UART) is a full-duplex communication channel capable of handling asynchronous data transfers. The UART uses a single 8-bit data mode with selectable parity. Features of the UART include:

- 8-bit asynchronous data transfer
- Selectable even- and odd-parity generation and checking
- Option of one or two STOP bits
- Separate transmit and receive interrupts
- Framing, parity, overrun, and break detection
- Separate transmit and receive enables
- 16-bit Baud Rate Generator
- Selectable Multiprocessor (9-bit) mode with three configurable interrupt schemes
- BRG timer mode
- Driver Enable output for external bus transceivers

Architecture

The UART consists of three primary functional blocks: Transmitter, Receiver, and Baud Rate Generator. The UART's transmitter and receiver functions independently, but use the same baud rate and data format. Figure11 on page 90 displays the UART architecture.

Clear To Send Operation

The CTS pin, if enabled by the CTSE bit of the UART Control 0 register, performs flow control on the outgoing transmit datastream. The Clear To Send ($\overline{CTS}$) input pin is sampled one system clock before beginning any new character transmission. To delay transmission of the next data character, an external receiver must deassert $\overline{CTS}$ at least one system clock cycle before a new data transmission begins. For multiple character transmissions, this would be done during STOP bit transmission. If $\overline{CTS}$ deasserts in the middle of a character transmission, the current character is sent completely.

Multiprocessor (9-bit) Mode

The UART has a MULTIPROCESSOR (9-bit) mode that uses an extra (9th) bit for selective communication when a number of processors share a common UART bus. In MULTIPROCESSOR mode (also referred to as 9-bit mode), the multiprocessor bit is transmitted following the 8-bits of data and immediately preceding the STOP bit(s) as displayed in Figure 14. The character format is as displayed in Figure 14.

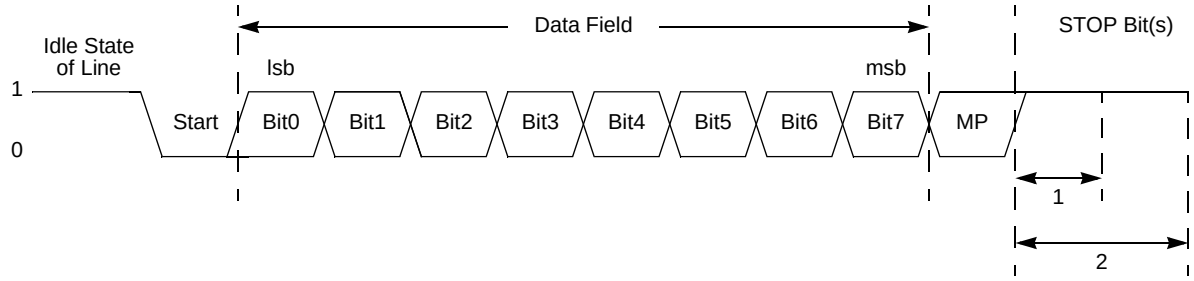


Figure 14. UART Asynchronous MULTIPROCESSOR Mode Data Format

In MULTIPROCESSOR (9-bit) mode, the Parity bit location (9th bit) becomes the Multiprocessor control bit. The UART Control 1 and Status 1 Registers provide Multiprocessor (9-bit) mode control and status information. If an automatic address matching scheme is enabled, the UART Address Compare Register holds the network address of the device.

MULTIPROCESSOR (9-bit) Mode Receive Interrupts

When multiprocessor mode is enabled, the UART only processes frames addressed to it. The determination of whether a frame of data is addressed to the UART can be made in hardware, software, or combination of the two depending on the multiprocessor configuration bits. In general, the address compare feature reduces the load on the CPU, because it does not need to access the UART when it receives data directed to other devices on the

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.

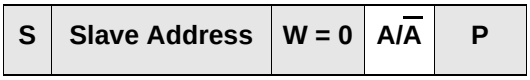


Figure 26. 7-Bit Address Only Transaction Format

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.

Write Transaction with a 7-Bit Address

Figure 27 displays the data transfer format for a 7-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	W = 0	A	Data	A	Data	A	Data	A/$\overline{A}$	P/S

Figure 27. 7-Bit Addressed Slave Data Transfer Format

Follow the steps below for a transmit operation 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.
4. Software responds to the TDRE bit by writing a 7-bit Slave address plus write bit (=0) to the I²C Data Register.
5. Software asserts the START bit of the I²C Control Register.
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. After one bit of address has been shifted out by the SDA signal, the Transmit Interrupt is asserted (TDRE = 1).
9. Software responds by writing the transmit data into the I²C Data Register.
10. The I²C Controller shifts the rest of the address and write bit out by 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 Not Acknowledge interrupt occurs (NCKI bit is set in the Status register, ACK bit is cleared). Software responds 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 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.

OCDCNTR Register

The OCD contains a multipurpose 16-bit Counter Register. It can be used for the following:

- Count system clock cycles between Breakpoints.
- Generate a BRK when it counts down to zero.
- Generate a BRK when its value matches the Program Counter.

When configured as a counter, the OCDCNTR register starts counting when the OCD leaves DEBUG mode and stops counting when it enters DEBUG mode again or when it reaches the maximum count of FFFFH. The OCDCNTR register automatically resets itself to 0000H when the OCD exits DEBUG mode if it is configured to count clock cycles between breakpoints.

Caution: *The OCDCNTR register is used by many of the OCD commands. It counts the number of bytes for the register and memory read/write commands. It holds the residual value when generating the CRC. Therefore, if the OCDCNTR is being used to generate a BRK, its value should be written as a last step before leaving DEBUG mode.*

Since this register is overwritten by various OCD commands, it should only be used to generate temporary breakpoints, such as stepping over CALL instructions or running to a specific instruction and stopping.

On-Chip Debugger Commands

The host communicates to the OCD by sending OCD commands using the DBG interface. During normal operation, only a subset of the OCD commands are available. In DEBUG mode, all OCD commands become available unless the user code and control registers are protected by programming the Read Protect Option Bit (RP). The Read Protect Option Bit prevents the code in memory from being read out of the Z8 Encore! XP F0822 Series products. When this option is enabled, several of the OCD commands are disabled. Table 93 on page 177 contains a summary of the OCD commands. Each OCD command is described further in the bulleted list. It also lists the commands that operate when the device is not in DEBUG mode (normal operation) and those commands that are disabled by programming the Read Protect Option Bit.

Table 93. On-Chip Debugger Commands

Debug Command	Command Byte	Enabled when NOT in DEBUG mode?	Disabled by Read Protect Option Bit
Read OCD Revision	00H	Yes	-
Write OCD Counter Register	01H	-	-
Read OCD Status Register	02H	Yes	-
Read OCD Counter Register	03H	-	-
Write OCD Control Register	04H	Yes	Cannot clear DBGMODE bit
Read OCD Control Register	05H	Yes	-
Write Program Counter	06H	-	Disabled
Read Program Counter	07H	-	Disabled
Write Register	08H	-	Only writes of the peripheral control registers at address F00H-FFH are allowed. Additionally, only the Mass Erase command is allowed to be written to the Flash Control Register.
Read Register	09H	-	Only reads of the peripheral control registers at address F00H-FFH are allowed.
Write Program Memory	0AH	-	Disabled
Read Program Memory	0BH	-	Disabled
Write Data Memory	0CH	-	Disabled
Read Data Memory	0DH	-	Disabled
Read Program Memory CRC	0EH	-	-
Reserved	0FH	-	-
Step Instruction	10H	-	Disabled
Stuff Instruction	11H	-	Disabled

DC Characteristics

Table 97 lists the DC characteristics of the Z8 Encore! XP® F0822 Series products. All voltages are referenced to V_{SS}, the primary system ground.

Table 97. DC Characteristics

Symbol	Parameter	T_A = -40 °C to 105 °C			Units	Conditions
		Minimum	Typical	Maximum		
V_{DD}	Supply Voltage	2.7	–	3.6	V	
V_{IL1}	Low Level Input Voltage	-0.3	–	0.3*V_{DD}	V	For all input pins except $\overline{\text{RESET}}$, DBG, and XIN.
V_{IL2}	Low Level Input Voltage	-0.3	–	0.2*V_{DD}	V	For $\overline{\text{RESET}}$, DBG, and XIN.
V_{IH1}	High Level Input Voltage	0.7*V_{DD}	–	5.5	V	Ports A and C pins when their programmable pull-ups are disabled.
V_{IH2}	High Level Input Voltage	0.7*V_{DD}	–	V_{DD}+0.3	V	Port B pins. Ports A and C pins when their programmable pull-ups are enabled.
V_{IH3}	High Level Input Voltage	0.8*V_{DD}	–	V_{DD}+0.3	V	$\overline{\text{RESET}}$, DBG, and XIN pins.
V_{OL1}	Low Level Output Voltage	–	–	0.4	V	I_{OL} = 2 mA; VDD = 3.0 V High Output Drive disabled.
V_{OH1}	High Level Output Voltage	2.4	–	–	V	I_{OH} = -2 mA; VDD = 3.0 V High Output Drive disabled.
V_{OL2}	Low Level Output Voltage High Drive	–	–	0.6	V	I_{OL} = 20 mA; VDD = 3.3 V High Output Drive enabled T_A = -40 °C to +70 °C
V_{OH2}	High Level Output Voltage High Drive	2.4	–	–	V	I_{OH} = -20 mA; VDD = 3.3 V High Output Drive enabled; T_A = -40 °C to +70 °C
V_{OL3}	Low Level Output Voltage High Drive	–	–	0.6	V	I_{OL} = 15mA; VDD = 3.3 V High Output Drive enabled; T_A = +70 °C to +105 °C
V_{OH3}	High Level Output Voltage High Drive	2.4	–	–	V	I_{OH} = 15 mA; VDD = 3.3 V High Output Drive enabled; T_A = +70 °C to +105 °C

Assembly Language Syntax

For proper instruction execution, eZ8 CPU assembly language syntax requires that the operands be written as ‘destination, source’. After assembly, the object code usually has the operands in the order ’source, destination’, but ordering is opcode-dependent. The following instruction examples illustrate the format of some basic assembly instructions and the resulting object code produced by the assembler. This binary format must be followed by users that prefer manual program coding or intend to implement their own assembler.

Example 1: If the contents of Registers 43H and 08H are added and the result is stored in 43H, the assembly syntax and resulting object code is:

Table 113. Assembly Language Syntax Example 1

Assembly Language Code	ADD	43H	08H	(ADD dst, src)
Object Code	04	08	43	(OPC src, dst)

Example 2: In general, when an instruction format requires an 8-bit register address, that address can specify any register location in the range 0–255 or, using Escaped Mode Addressing, a Working Register R0–R15. If the contents of Register 43H and Working Register R8 are added and the result is stored in 43H, the assembly syntax and resulting object code is:

Table 114. Assembly Language Syntax Example 2

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

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

eZ8 CPU Instruction Notation

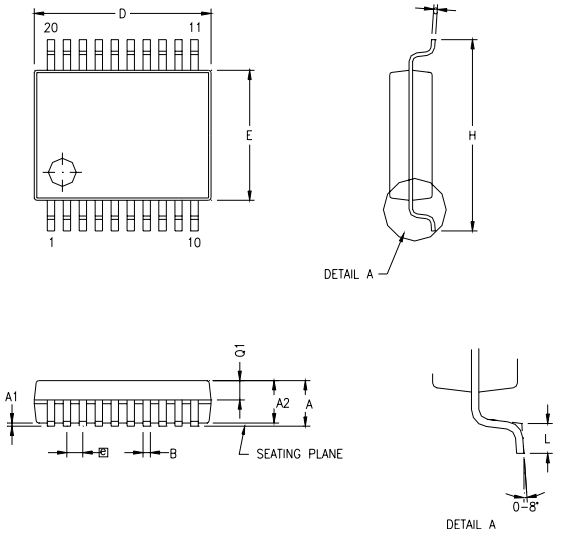
In the eZ8 CPU Instruction Summary and Description sections, the operands, condition codes, status flags, and address modes are represented by a notational shorthand that is described in Table 115 on page 211.

Upper Nibble (Hex) \ Lower Nibble (Hex)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0																
1																
2																
3																
4																
5																
6																
7																
8																
9																
A			3.3 **CPC** r1,r2	3.4 **CPC** r1,Ir2	4.3 **CPC** R2,R1	4.4 **CPC** IR2,R1	4.3 **CPC** R1,IM	4.4 **CPC** IR1,IM	5.3 **CPCX** ER2,ER1	5.3 **CPCX** IM,ER1						
B																
C	3.2 **SRL** R1	3.3 **SRL** IR1														
D																
E																
F																

Figure 59. Second Opcode Map after 1FH

Packaging

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

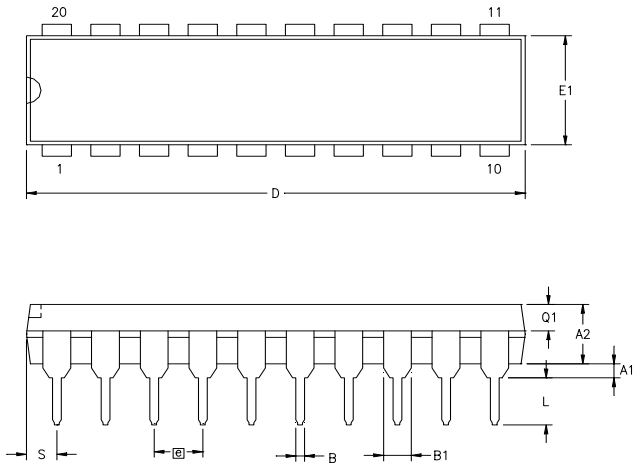


SYMBOL	MILLIMETER			INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	1.73	1.85	1.98	0.068	0.073	0.078
A1	0.05	0.13	0.21	0.002	0.005	0.008
A2	1.68	1.73	1.83	0.066	0.068	0.072
B	0.25	0.30	0.38	0.010	0.012	0.015
C	0.13	0.15	0.22	0.005	0.006	0.009
D	7.07	7.20	7.33	0.278	0.283	0.289
E	5.20	5.30	5.38	0.205	0.209	0.212
e	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)

D

E

F

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.