



Welcome to [E-XFL.COM](https://www.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	5MHz
Connectivity	IrDA, UART/USART
Peripherals	Brown-out Detect/Reset, LED, POR, PWM, WDT
Number of I/O	24
Program Memory Size	1KB (1K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	256 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-SSOP (0.173", 4.40mm Width)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f0113hj005ec

Internal Precision Oscillator

The internal precision oscillator (IPO) is a trimmable clock source that requires no external components.

10-Bit Analog-to-Digital Converter

The optional analog-to-digital converter (ADC) converts an analog input signal to a 10-bit binary number. The ADC accepts inputs from eight different analog input pins in both single-ended and differential modes.

Analog Comparator

The analog comparator compares the signal at an input pin with either an internal programmable voltage reference or a second input pin. The comparator output can be used to drive either an output pin or to generate an interrupt.

Universal Asynchronous Receiver/Transmitter

The UART is full-duplex and capable of handling asynchronous data transfers. The UART supports 8- and 9-bit data modes and selectable parity. The UART also supports multi-drop address processing in hardware. The UART baud rate generator can be configured and used as a basic 16-bit timer.

Timers

Two enhanced 16-bit reloadable timers can be used for timing/counting events or for motor control operations. These timers provide a 16-bit programmable reload counter and operate in ONE-SHOT, CONTINUOUS, GATED, CAPTURE, CAPTURE RESTART, COMPARE, CAPTURE AND COMPARE, PWM SINGLE OUTPUT, and PWM DUAL OUTPUT modes.

Interrupt Controller

Z8 Encore! XP[®] F0823 Series products support up to 20 interrupts. These interrupts consist of eight internal peripheral interrupts and 12 general-purpose I/O pin interrupt sources. The interrupts have three levels of programmable interrupt priority.

Table 8. Register File Address Map (Continued)

Address (Hex)	Register Description	Mnemonic	Reset (Hex)	Page No
FF2	Watchdog Timer Reload High Byte	WDTH	FF	91
FF3	Watchdog Timer Reload Low Byte	WDTL	FF	91
FF4–FF5	Reserved	—	XX	
Trim Bit Control				
FF6	Trim Bit Address	TRMADR	00	143
FF7	Trim Data	TRMDR	XX	144
Flash Memory Controller				
FF8	Flash Control	FCTL	00	137
FF8	Flash Status	FSTAT	00	137
FF9	Flash Page Select	FPS	00	138
	Flash Sector Protect	FPROT	00	139
FFA	Flash Programming Frequency High Byte	FFREQH	00	140
FFB	Flash Programming Frequency Low Byte	FFREQL	00	140
eZ8 CPU				
FFC	Flags	—	XX	Refer to eZ8 CPU Core User Manual (UM0128)
FFD	Register Pointer	RP	XX	
FFE	Stack Pointer High Byte	SPH	XX	
FFF	Stack Pointer Low Byte	SPL	XX	
XX=Undefined				

STOP—Stop Mode Recovery Indicator

If this bit is set to 1, a Stop Mode Recovery is occurred. If the STOP and WDT bits are both set to 1, the Stop Mode Recovery occurred because of 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 from a change in an input pin also resets this bit. Reading this register resets this bit. This read must occur before clearing the WDT interrupt.

EXT—External Reset Indicator

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

Reserved—0 when read

► **Note:** *This register is only reset during a Power-On Reset sequence. Other System Reset events do not affect it.*

Table 13. Power Control Register 0 (PWRCTL0)

BITS	7	6	5	4	3	2	1	0
FIELD	Reserved	Reserved		VBO	Reserved	ADC	COMP	Reserved
RESET	1	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
ADDR	F80H							

Reserved—Must be 1

Reserved—Must be 0

VBO—Voltage Brownout Detector Disable

This bit and the VBO_AO Flash option bit must both enable the VBO for the VBO to be active.

0 = VBO enabled

1 = VBO disabled

ADC—Analog-to-Digital Converter Disable

0 = Analog-to-Digital Converter enabled

1 = Analog-to-Digital Converter disabled

COMP—Comparator Disable

0 = Comparator is enabled

1 = Comparator is disabled

Reserved—Must be 0

Port A–C Address Registers

The Port A–C Address registers select the GPIO Port functionality accessible through the Port A–C Control registers. The Port A–C Address and Control registers combine to provide access to all GPIO Port controls (Table 18).

Table 18. Port A–C GPIO Address Registers (PxADDR)

BITS	7	6	5	4	3	2	1	0
FIELD	PADDR[7:0]							
RESET	00H							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
ADDR	FD0H, FD4H, FD8H							

PADDR[7:0]—Port Address

The Port Address selects one of the sub-registers accessible through the Port Control register.

PADDR[7:0]	Port Control Sub-register Accessible Using the Port A–C Control Registers
00H	No function. Provides some protection against accidental Port reconfiguration.
01H	Data Direction.
02H	Alternate Function.
03H	Output Control (Open-Drain).
04H	High Drive Enable.
05H	Stop Mode Recovery Source Enable.
06H	Pull-up Enable.
07H	Alternate Function Set 1.
08H	Alternate Function Set 2.
09H–FFH	No function.

Port A–C Control Registers

The Port A–C Control registers set the GPIO port operation. The value in the corresponding Port A–C Address register determines which sub-register is read from or written to by a Port A–C Control register transaction (Table 19).

Table 33. Trap and Interrupt Vectors in Order of Priority

Priority	Program Memory Vector Address	Interrupt or Trap Source
Highest	0002H	Reset (not an interrupt)
	0004H	Watchdog Timer (see Watchdog Timer on page 87)
	003AH	Primary Oscillator Fail Trap (not an interrupt)
	003CH	Watchdog Timer Oscillator Fail Trap (not an interrupt)
	0006H	Illegal Instruction Trap (not an interrupt)
	0008H	Reserved
	000AH	Timer 1
	000CH	Timer 0
	000EH	UART 0 receiver
	0010H	UART 0 transmitter
	0012H	Reserved
	0014H	Reserved
	0016H	ADC
	0018H	Port A Pin 7, selectable rising or falling input edge
	001AH	Port A Pin 6, selectable rising or falling input edge or Comparator Output
	001CH	Port A Pin 5, selectable rising or falling input edge
	001EH	Port A Pin 4, selectable rising or falling input edge
	0020H	Port A Pin 3 or Port D Pin 3, selectable rising or falling input edge
	0022H	Port A Pin 2 or Port D Pin 2, selectable rising or falling input edge
	0024H	Port A Pin 1, selectable rising or falling input edge
	0026H	Port A Pin 0, selectable rising or falling input edge
	0028H	Reserved
	002AH	Reserved
	002CH	Reserved
	002EH	Reserved
	0030H	Port C Pin 3, both input edges
	0032H	Port C Pin 2, both input edges
	0034H	Port C Pin 1, both input edges

Interrupt Request 1 Register

The Interrupt Request 1 (IRQ1) register (Table 35) stores interrupt requests for both vectored and polled interrupts. When a request is presented to the interrupt controller, the corresponding bit in the IRQ1 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 Interrupt Request 1 register to determine if any interrupt requests are pending.

Table 35. Interrupt Request 1 Register (IRQ1)

BITS	7	6	5	4	3	2	1	0
FIELD	PA7VI	PA6CI	PA5I	PA4I	PA3I	PA2I	PA1I	PA0I
RESET	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
ADDR	FC3H							

PA7VI—Port A7 Interrupt Request

0 = No interrupt request is pending for GPIO Port A

1 = An interrupt request from GPIO Port A

PA6CI—Port A6 or Comparator Interrupt Request

0 = No interrupt request is pending for GPIO Port A or Comparator

1 = An interrupt request from GPIO Port A or Comparator

PAxI—Port A Pin x Interrupt Request

0 = No interrupt request is pending for GPIO Port A pin x

1 = An interrupt request from GPIO Port A pin x is awaiting service

where x indicates the specific GPIO Port pin number (0–5)

Interrupt Request 2 Register

The Interrupt Request 2 (IRQ2) register (Table 36) stores interrupt requests for both vectored and polled interrupts. When a request is presented to the interrupt controller, the corresponding bit in the IRQ2 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 can read the Interrupt Request 2 register to determine if any interrupt requests are pending.

timer value is not reset to 0001H). Also, if the Timer Output alternate function is enabled, the Timer Output pin changes state (from Low to High or from High to Low) upon Compare.

If the Timer reaches FFFFH, the timer rolls over to 0000H and continue counting. Follow the steps below to configure a timer for COMPARE mode and to initiate the count:

1. Write to the Timer Control register to:
 - Disable the timer.
 - Configure the timer for Compare mode.
 - Set the prescale value.
 - Set the initial logic level (High or Low) for the Timer Output alternate function, if appropriate.
2. Write to the Timer High and Low Byte registers to set the starting count value.
3. Write to the Timer Reload High and Low Byte registers to set the Compare value.
4. Enable the timer interrupt, if appropriate, and set the timer interrupt priority by writing to the relevant interrupt registers.
5. If using the Timer Output function, configure the associated GPIO port pin for the Timer Output alternate function.
6. Write to the Timer Control register to enable the timer and initiate counting.

In COMPARE mode, the system clock always provides the timer input. The Compare time can be calculated by the following equation:

$$\text{COMPARE Mode Time (s)} = \frac{(\text{Compare Value} - \text{Start Value}) \times \text{Prescale}}{\text{System Clock Frequency (Hz)}}$$

GATED Mode

In GATED mode, the timer counts only when the Timer Input signal is in its active state (asserted), as determined by the TPOL bit in the Timer Control register. When the Timer Input signal is asserted, counting begins. A timer interrupt is generated when the Timer Input signal is deasserted or a timer Reload occurs. To determine if a Timer Input signal deassertion generated the interrupt, read the associated GPIO input value and compare to the value stored in the TPOL bit.

The timer counts up to the 16-bit Reload value stored in the Timer Reload High and Low Byte registers. The timer input is the system clock. When reaching the Reload value, the timer generates an interrupt, the count value in the Timer High and Low Byte registers is reset to 0001H and counting resumes (assuming the Timer Input signal remains asserted). Also, if the Timer Output alternate function is enabled, the Timer Output pin changes state (from Low to High or from High to Low) at timer reset.

Receiving Data using the Interrupt-Driven Method

The UART Receiver interrupt indicates the availability of new data (as well as error conditions). Follow the steps below to configure the UART receiver for interrupt-driven operation:

1. Write to the UART Baud Rate High and Low Byte registers to set the acceptable baud rate.
2. Enable the UART pin functions by configuring the associated GPIO port pins for alternate function operation.
3. Execute a `DI` instruction to disable interrupts.
4. Write to the Interrupt control registers to enable the UART Receiver interrupt and set the acceptable priority.
5. Clear the UART Receiver interrupt in the applicable Interrupt Request register.
6. Write to the UART Control 1 Register to enable Multiprocessor (9-bit) mode functions, if appropriate.
 - Set the Multiprocessor Mode Select (`MPEN`) to Enable `MULTIPROCESSOR` mode
 - Set the Multiprocessor Mode Bits, `MPMD[1:0]`, to select the acceptable address matching scheme
 - Configure the UART to interrupt on received data and errors or errors only (interrupt on errors only is unlikely to be useful for Z8 Encore! XP devices without a DMA block)
7. Write the device address to the Address Compare Register (automatic `MULTIPROCESSOR` modes only).
8. Write to the UART Control 0 register to:
 - Set the receive enable bit (`REN`) to enable the UART for data reception
 - Enable parity, if appropriate and if multiprocessor mode is not enabled, and select either even or odd parity
9. Execute an `EI` instruction to enable interrupts.

The UART is now configured for interrupt-driven data reception. When the UART Receiver interrupt is detected, the associated interrupt service routine (ISR) performs the following:

1. Checks the UART Status 0 register to determine the source of the interrupt - error, break, or received data.
2. Reads the data from the UART Receive Data register if the interrupt was because of data available. If operating in `MULTIPROCESSOR` (9-bit) mode, further actions may be required depending on the `MULTIPROCESSOR` mode bits `MPMD[1:0]`.

value 63H to the Flash Control register initiates the Mass Erase operation. While the Flash Controller executes the Mass Erase operation, the eZ8 CPU idles but the system clock and on-chip peripherals continue to operate. Using the On-Chip Debugger, poll the Flash Status register to determine when the Mass Erase operation is complete. When the Mass Erase is complete, the Flash Controller returns to its locked state.

Flash Controller Bypass

The Flash Controller can be bypassed and the control signals for the Flash memory brought out to the GPIO pins. Bypassing the Flash Controller allows faster Row Programming algorithms by controlling the Flash programming signals directly.

Row programming is recommended for gang programming applications and large volume customers who do not require in-circuit initial programming of the Flash memory. Page Erase operations are also supported when the Flash Controller is bypassed.

For more information on bypassing the Flash Controller, refer to *Third-Party Flash Programming Support for Z8 Encore! (AN0117)* available for download at www.zilog.com.

Flash Controller Behavior in DEBUG Mode

The following changes in behavior of the Flash Controller occur when the Flash Controller is accessed using the On-Chip Debugger:

- The Flash Write Protect option bit is ignored
- The Flash Sector Protect register is ignored for programming and erase operations
- Programming operations are not limited to the page selected in the Page Select register
- Bits in the Flash Sector Protect register can be written to one or zero
- The second write of the Page Select register to unlock the Flash Controller is not necessary
- The Page Select register can be written when the Flash Controller is unlocked
- The Mass Erase command is enabled through the Flash Control register

! **Caution:** *For security reasons, Flash controller allows only a single page to be opened for write/erase. When writing multiple Flash pages, the Flash controller must go through the unlock sequence again to select another page.*

Reserved—0 when read

FSTAT—Flash Controller Status

000000 = Flash Controller locked

000001 = First unlock command received (73H written)

000010 = Second unlock command received (8CH written)

000011 = Flash Controller unlocked

000100 = Sector protect register selected

001xxx = Program operation in progress

010xxx = Page erase operation in progress

100xxx = Mass erase operation in progress

Flash Page Select Register

The Flash Page Select (FPS) register shares address space with the Flash Sector Protect Register. Unless the Flash controller is unlocked and written with 5EH, writes to this address target the Flash Page Select Register.

The register is used to select one of the eight available Flash memory pages to be programmed or erased. Each Flash Page contains 512 bytes of Flash memory. During a Page Erase operation, all Flash memory having addresses with the most significant 7-bits given by FPS[6:0] are chosen for program/erase operation.

Table 81. Flash Page Select Register (FPS)

BITS	7	6	5	4	3	2	1	0
FIELD	INFO_EN	PAGE						
RESET	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
ADDR	FF9H							

INFO_EN—Information Area Enable

0 = Information Area us not selected

1 = Information Area is selected. The Information Area is mapped into the Program Memory address space at addresses FE00H through FFFFH.

PAGE—Page Select

This 7-bit field identifies the Flash memory page for Page Erase and page unlocking. Program Memory Address[15:9] = PAGE[6:0]. For the Z8F04x3 devices, the upper 4 bits must always be 0. For the Z8F02x3 devices, the upper 5 bits must always be 0. For the Z8F01x3 devices, the upper 6 bits must always be 0.

Flash Sector Protect Register

The Flash Sector Protect (FPROT) register is shared with the Flash Page Select Register. When the Flash Control Register is written with 73H followed by 5EH, the next write to this address targets the Flash Sector Protect Register. In all other cases, it targets the Flash Page Select Register.

This register selects one of the 8 available Flash memory sectors to be protected. The reset state of each Sector Protect bit is an unprotected state. After a sector is protected by setting its corresponding register bit, it cannot be unprotected (the register bit cannot be cleared) without powering down the device.

Table 82. Flash Sector Protect Register (FPROT)

BITS	7	6	5	4	3	2	1	0
FIELD	SPROT7	SPROT6	SPROT5	SPROT4	SPROT3	SPROT2	SPROT1	SPROT0
RESET	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
ADDR	FF9H							

SPROT7-SPROT0—Sector Protection

Each bit corresponds to a 512 bytes Flash sector. For the Z8F08x3 devices, the upper 3 bits must be zero. For the Z8F04x3 devices all bits are used. For the Z8F02x3 devices, the upper 4 bits are unused. For the Z8F01x3 devices, the upper 6 bits are unused.

Flash Frequency High and Low Byte Registers

The Flash Frequency High (FFREQH) and Low Byte (FFREQL) registers combine to form a 16-bit value, FFREQ, to control timing for Flash program and erase operations. The 16-bit binary Flash Frequency value must contain the system clock frequency (in kHz) and is calculated using the following equation:

$$\text{FFREQ}[15:0] = \{\text{FFREQH}[7:0], \text{FFREQL}[7:0]\} = \frac{\text{System Clock Frequency}}{1000}$$

! Caution: *The Flash Frequency High and Low Byte registers must be loaded with the correct value to ensure proper operation of the device. Also, Flash programming and erasure is not supported for system clock frequencies below 20 kHz or above 20 MHz.*

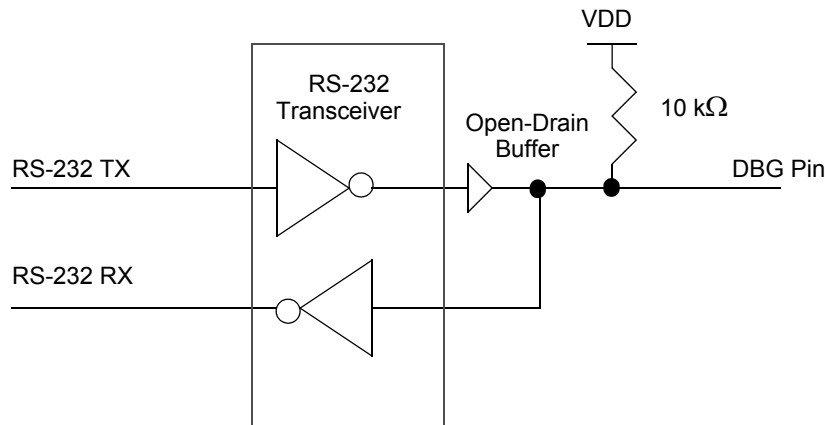


Figure 24. Interfacing the On-Chip Debugger's DBG Pin with an RS-232 Interface (2)

DEBUG Mode

The operating characteristics of the devices in DEBUG mode are:

- The eZ8 CPU fetch unit stops, idling the eZ8 CPU, unless directed by the OCD to execute specific instructions
- The system clock operates unless in STOP mode
- All enabled on-chip peripherals operate unless in STOP mode
- Automatically exits HALT mode
- Constantly refreshes the Watchdog Timer, if enabled.

Entering DEBUG Mode

The device enters DEBUG mode following the operations below:

- The device enters DEBUG mode after the eZ8 CPU executes a BRK (breakpoint) instruction
- If the DBG pin is held Low during the most recent clock cycle of System Reset, the part enters DEBUG mode upon exiting System Reset

► **Note:** Holding the DBG pin Low for an additional 5000 (minimum) clock cycles after reset (making sure to account for any specified frequency error if using an internal oscillator) prevents a false interpretation of an Autobaud sequence (see OCD Auto-Baud Detector/Generator on page 154).

- If the PA2/RESET pin is held Low while a 32-bit key sequence is issued to the PA0/DBG pin, the DBG feature is unlocked. After releasing PA2/RESET, it is pulled high. At this

Watchdog Timer Failure

In the event of a Watchdog Timer oscillator failure, a similar non-maskable interrupt-like event is issued. This event does not trigger an attendant clock switch-over, but alerts the CPU of the failure. After a Watchdog Timer failure, it is no longer possible to detect a primary oscillator failure. The failure detection circuitry does not function if the Watchdog Timer is used as the primary oscillator or if the Watchdog Timer oscillator has been disabled. For either of these cases, it is necessary to disable the detection circuitry by deasserting the WDFEN bit of the OSCCTL register.

The Watchdog Timer oscillator failure detection circuit counts system clocks while searching for a Watchdog Timer clock. The logic counts 8004 system clock cycles before determining that a failure has occurred. The system clock rate determines the speed at which the Watchdog Timer failure can be detected. A very slow system clock results in very slow detection times.

! **Caution:** *It is possible to disable the clock failure detection circuitry as well as all functioning clock sources. In this case, the Z8 Encore! XP F0823 Series device ceases functioning and can only be recovered by Power-On Reset.*

Oscillator Control Register Definitions

The following section provides the bit definitions for the Oscillator Control register.

Oscillator Control Register

The Oscillator Control register (OSCCTL) enables/disables the various oscillator circuits, enables/disables the failure detection/recovery circuitry and selects the primary oscillator, which becomes the system clock.

The Oscillator Control register must be unlocked before writing. Writing the two step sequence $E7H$ followed by $18H$ to the Oscillator Control Register unlocks it. The register is locked at successful completion of a register write to the OSCCTL.

Table 102. Oscillator Control Register (OSCCTL)

BITS	7	6	5	4	3	2	1	0
FIELD	INTEN	Reserved	WDTEN	POFEN	WDFEN	SCKSEL		
RESET	1	0	1	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
ADDR	F86H							






		Lower Nibble (Hex)															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Upper Nibble (Hex)	0	1.1 BRK	2.2 SRP IM	2.3 ADD r1,r2	2.4 ADD r1,l,r2	3.3 ADD R2,R1	3.4 ADD IR2,R1	3.3 ADD R1,IM	3.4 ADD IR1,IM	4.3 ADDX ER2,ER1	4.3 ADDX IM,ER1	2.3 DJNZ r1,X	2.2 JR cc,X	2.2 LD r1,IM	3.2 JP cc,DA	1.2 INC r1	1.2 NOP
	1	2.2 RLC R1	2.3 RLC IR1	2.3 ADC r1,r2	2.4 ADC r1,l,r2	3.3 ADC R2,R1	3.4 ADC IR2,R1	3.3 ADC R1,IM	3.4 ADC IR1,IM	4.3 ADCX ER2,ER1	4.3 ADCX IM,ER1						See 2nd Opcode Map
	2	2.2 INC R1	2.3 INC IR1	2.3 SUB r1,r2	2.4 SUB r1,l,r2	3.3 SUB R2,R1	3.4 SUB IR2,R1	3.3 SUB R1,IM	3.4 SUB IR1,IM	4.3 SUBX ER2,ER1	4.3 SUBX IM,ER1						1, 2 ATM
	3	2.2 DEC R1	2.3 DEC IR1	2.3 SBC r1,r2	2.4 SBC r1,l,r2	3.3 SBC R2,R1	3.4 SBC IR2,R1	3.3 SBC R1,IM	3.4 SBC IR1,IM	4.3 SBCX ER2,ER1	4.3 SBCX IM,ER1						
	4	2.2 DA R1	2.3 DA IR1	2.3 OR r1,r2	2.4 OR r1,l,r2	3.3 OR R2,R1	3.4 OR IR2,R1	3.3 OR R1,IM	3.4 OR IR1,IM	4.3 ORX ER2,ER1	4.3 ORX IM,ER1						
	5	2.2 POP R1	2.3 POP IR1	2.3 AND r1,r2	2.4 AND r1,l,r2	3.3 AND R2,R1	3.4 AND IR2,R1	3.3 AND R1,IM	3.4 AND IR1,IM	4.3 ANDX ER2,ER1	4.3 ANDX IM,ER1						1.2 WDT
	6	2.2 COM R1	2.3 COM IR1	2.3 TCM r1,r2	2.4 TCM r1,l,r2	3.3 TCM R2,R1	3.4 TCM IR2,R1	3.3 TCM R1,IM	3.4 TCM IR1,IM	4.3 TCMX ER2,ER1	4.3 TCMX IM,ER1						1.2 STOP
	7	2.2 PUSH R2	2.3 PUSH IR2	2.3 TM r1,r2	2.4 TM r1,l,r2	3.3 TM R2,R1	3.4 TM IR2,R1	3.3 TM R1,IM	3.4 TM IR1,IM	4.3 TMX ER2,ER1	4.3 TMX IM,ER1						1.2 HALT
	8	2.5 DECW RR1	2.6 DECW IRR1	2.5 LDE r1,l,r2	2.9 LDEI l,r1,l,r2	3.2 LDX r1,ER2	3.3 LDX l,r1,ER2	3.4 LDX IRR2,R1	3.5 LDX IRR2,IR1	3.4 LDX r1,r2,X	3.4 LDX rr1,r2,X						1.2 DI
	9	2.2 RL R1	2.3 RL IR1	2.5 LDE r2,l,r1	2.9 LDEI l,r2,l,r1	3.2 LDX r2,ER1	3.3 LDX l,r2,ER1	3.4 LDX R2,IRR1	3.5 LDX IR2,IRR1	3.3 LEA r1,r2,X	3.5 LEA rr1,r2,X						1.2 EI
	A	2.5 INCW RR1	2.6 INCW IRR1	2.3 CP r1,r2	2.4 CP r1,l,r2	3.3 CP R2,R1	3.4 CP IR2,R1	3.3 CP R1,IM	3.4 CP IR1,IM	4.3 CPX ER2,ER1	4.3 CPX IM,ER1						1.4 RET
	B	2.2 CLR R1	2.3 CLR IR1	2.3 XOR r1,r2	2.4 XOR r1,l,r2	3.3 XOR R2,R1	3.4 XOR IR2,R1	3.3 XOR R1,IM	3.4 XOR IR1,IM	4.3 XORX ER2,ER1	4.3 XORX IM,ER1						1.5 IRET
	C	2.2 RRC R1	2.3 RRC IR1	2.5 LDC r1,l,r2	2.9 LDCI l,r1,l,r2	2.3 JP IRR1	2.9 LDC l,r1,l,r2		3.4 LD r1,r2,X	3.2 PUSHX ER2							1.2 RCF
	D	2.2 SRA R1	2.3 SRA IR1	2.5 LDC r2,l,r1	2.9 LDCI l,r2,l,r1	2.6 CALL IRR1	2.2 BSWAP R1	3.3 CALL DA	3.4 LD r2,r1,X	3.2 POPX ER1							1.2 SCF
	E	2.2 RR R1	2.3 RR IR1	2.2 BIT p,b,r1	2.3 LD r1,l,r2	3.2 LD R2,R1	3.3 LD IR2,R1	3.2 LD R1,IM	3.3 LD IR1,IM	4.2 LDX ER2,ER1	4.2 LDX IM,ER1						1.2 CCF
	F	2.2 SWAP R1	2.3 SWAP IR1	2.6 TRAP Vector	2.3 LD l,r1,r2	2.8 MULT RR1	3.3 LD R2,IR1	3.3 BTJ p,b,r1,X	3.4 BTJ p,b,l,r1,X								

Figure 27. First Opcode Map

Packaging

Figure 34 displays the 8-pin Plastic Dual Inline Package (PDIP) available for the Z8 Encore! XP® F0823 Series devices.

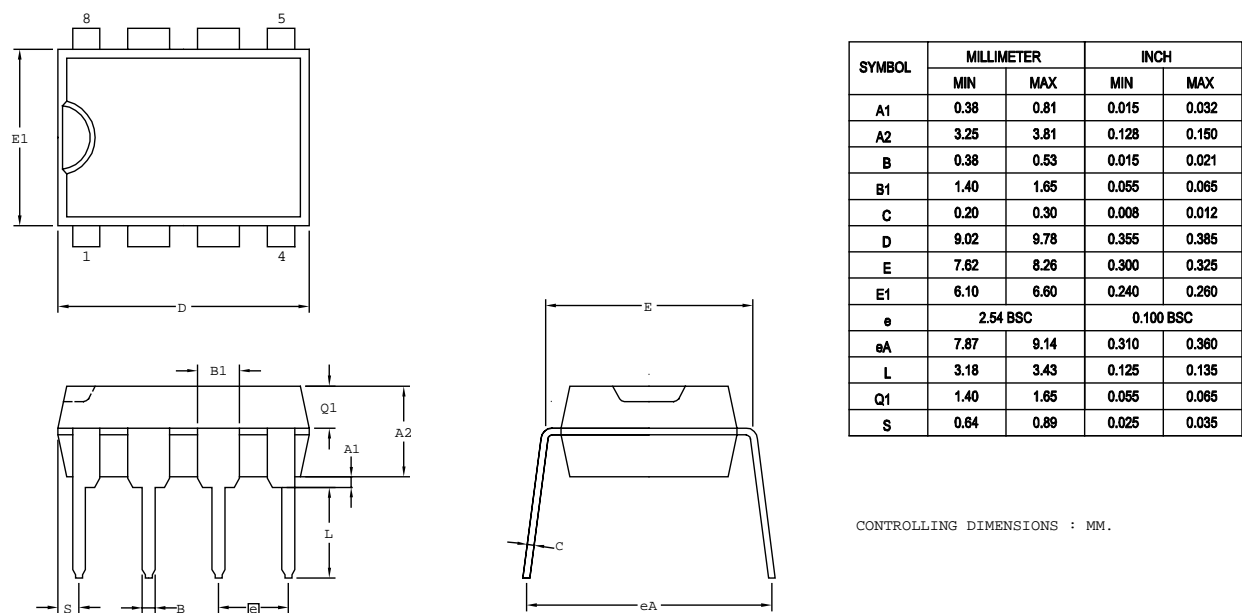
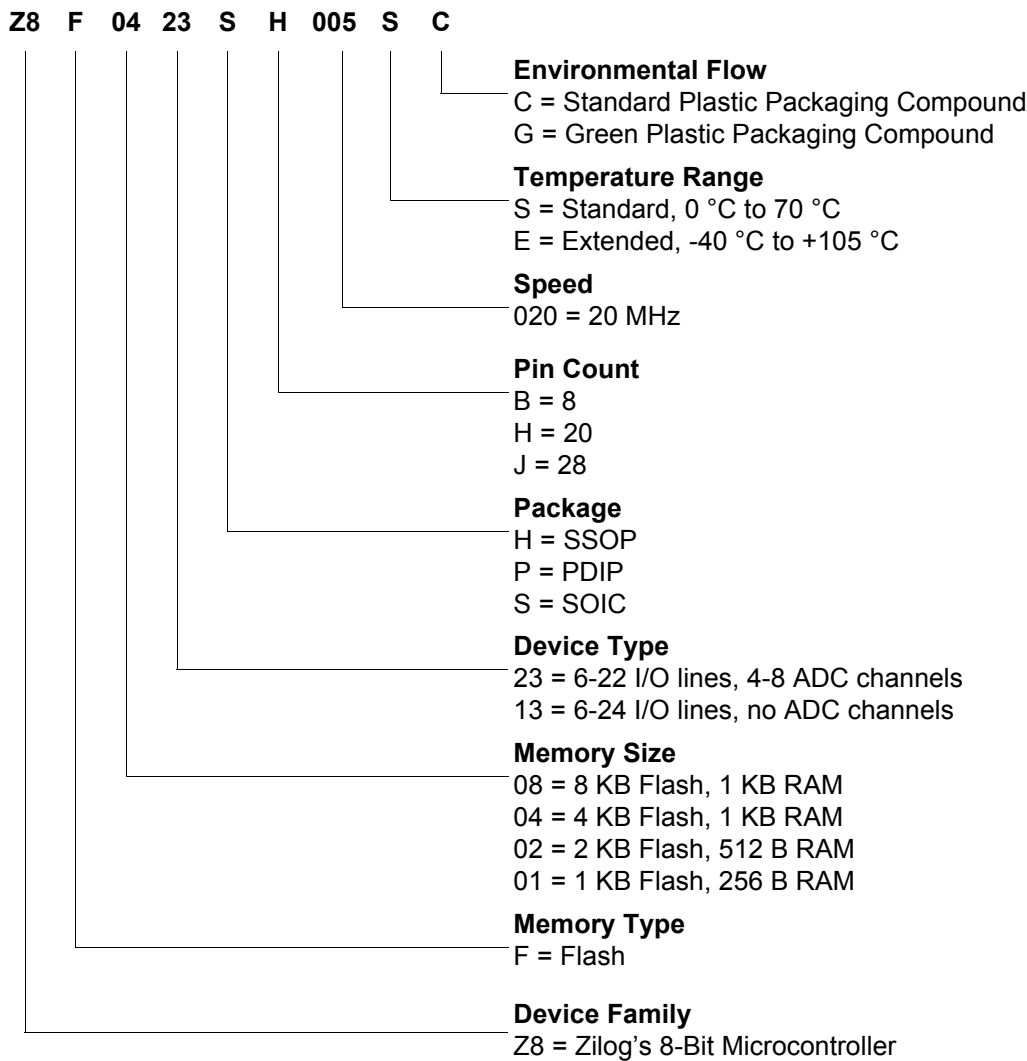


Figure 34. 8-Pin Plastic Dual Inline Package (PDIP)

Part Number Suffix Designations



page erase 135
page select register 138, 139
FPS register 138, 139
FSTAT register 137

G

GATED mode 84
general-purpose I/O 35
GPIO 4, 35
 alternate functions 36
 architecture 36
 control register definitions 43
 input data sample timing 202
 interrupts 43
 port A-C pull-up enable sub-registers 48, 49
 port A-H address registers 44
 port A-H alternate function sub-registers 45
 port A-H control registers 44
 port A-H data direction sub-registers 45
 port A-H high drive enable sub-registers 47
 port A-H input data registers 49
 port A-H output control sub-registers 46
 port A-H output data registers 50
 port A-H stop mode recovery sub-registers 47
 port availability by device 35
 port input timing 203
 port output timing 204

H

H 174
HALT 176
halt mode 32, 176
hexadecimal number prefix/suffix 174

I

I²C 4
IM 173
immediate data 173
immediate operand prefix 174

INC 175
increment 175
increment word 175
INCW 175
indexed 173
indirect address prefix 174
indirect register 173
indirect register pair 173
indirect working register 173
indirect working register pair 173
infrared encoder/decoder (IrDA) 113
Instruction Set 171
instruction set, eZ8 CPU 171
instructions
 ADC 175
 ADCX 175
 ADD 175
 ADDX 175
 AND 177
 ANDX 177
 arithmetic 175
 BCLR 176
 BIT 176
 bit manipulation 176
 block transfer 176
 BRK 178
 BSET 176
 BSWAP 176, 178
 BTJ 178
 BTJNZ 178
 BTJZ 178
 CALL 178
 CCF 176
 CLR 177
 COM 177
 CP 175
 CPC 175
 CPCX 175
 CPU control 176
 CPX 175
 DA 175
 DEC 175
 DECW 175
 DI 176

DJNZ 178
EI 176
HALT 176
INC 175
INCW 175
IRET 178
JP 178
LD 177
LDC 177
LDCI 176, 177
LDE 177
LDEI 176
LDX 177
LEA 177
load 177
logical 177
MULT 175
NOP 176
OR 177
ORX 178
POP 177
POPX 177
program control 178
PUSH 177
PUSHX 177
RCF 176
RET 178
RL 178
RLC 178
rotate and shift 178
RR 178
RRC 178
SBC 175
SCF 176, 177
SRA 179
SRL 179
SRP 177
STOP 177
SUB 175
SUBX 175
SWAP 179
TCM 176
TCMX 176
TM 176

TMX 176
TRAP 178
Watchdog Timer refresh 177
XOR 178
XORX 178
instructions, eZ8 classes of 174
interrupt control register 64
interrupt controller 53
 architecture 53
 interrupt assertion types 56
 interrupt vectors and priority 56
 operation 55
 register definitions 58
 software interrupt assertion 57
interrupt edge select register 63
interrupt request 0 register 58
interrupt request 1 register 59
interrupt request 2 register 59
interrupt return 178
interrupt vector listing 53
interrupts
 UART 101
IR 173
Ir 173
IrDA
 architecture 113
 block diagram 113
 control register definitions 116
 operation 113
 receiving data 115
 transmitting data 114
IRET 178
IRQ0 enable high and low bit registers 60
IRQ1 enable high and low bit registers 61
IRQ2 enable high and low bit registers 62
IRR 173
Irr 173

J

JP 178
jump, conditional, relative, and relative conditional 178

