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	Surface Mount
Package / Case	20-SSOP (0.209", 5.30mm Width)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f0421hh020sc

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

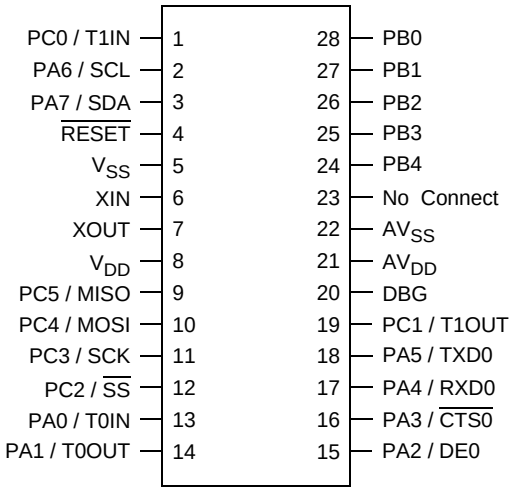


Figure 5. Z8F0812 and Z8F0412 in 28-Pin SOIC and PDIP Packages

Signal Descriptions

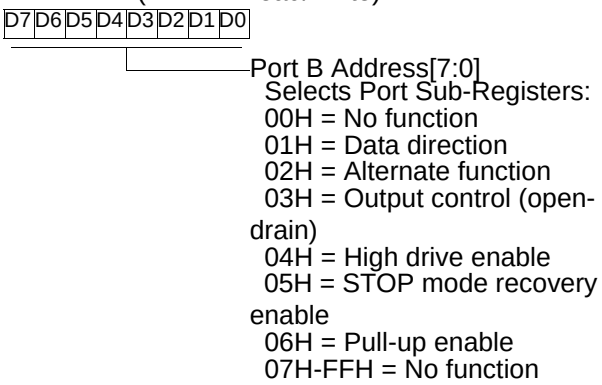
Table 3 describes Z8 Encore! XP® F0822 Series signals. See Pin Configurations on page 7 to determine the signals available for the specific package styles

.

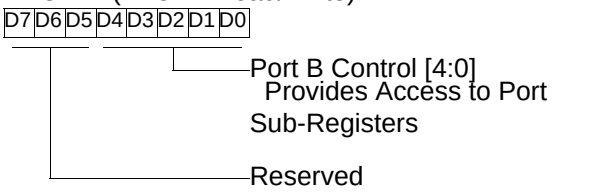
Table 3. Signal Descriptions

Signal Mnemonic	I/O	Description
General-Purpose I/O Ports A-H		
PA[7:0]	I/O	**Port C**—These pins are used for general-purpose I/O and supports 5 V-tolerant inputs.
PB[4:0]	I/O	**Port B**—These pins are used for general-purpose I/O.
PC[5:0]	I/O	**Port C**—These pins are used for general-purpose I/O and support 5 V-tolerant inputs.
I^2C Controller		
SCL	I/O	**Serial Clock**—This open-drain pin clocks data transfers in accordance with the I^2C standard protocol. This pin is multiplexed with a GPIO pin. When the GPIO pin is configured for alternate function to enable the SCL function, this pin is open-drain.
SDA	I/O	**Serial Data**—This open-drain pin transfers data between the I^2C and a slave. This pin is multiplexed with a GPIO pin. When the GPIO pin is configured for alternate function to enable the SDA function, this pin is open-drain.

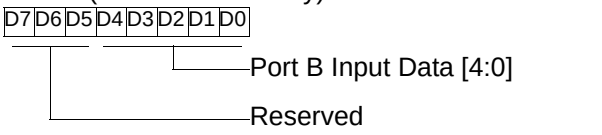
Port B Address
PBADDR (FD4H - Read/Write)



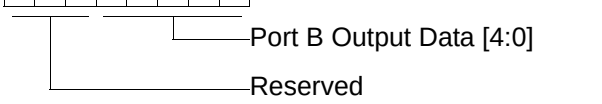
Port B Control
PBCTL (FD5H - Read/Write)



Port B Input Data
PBIN (FD6H - Read Only)



Port B Output Data
PBOUT (FD7H - Read/Write)



Flash Control
FCTL (FF8H - Write Only)

D7	D6	D5	D4	D3	D2	D1	D0

Flash Command
73H = First unlock command
8CH = Second unlock command
95H = Page erase command
63H = Mass erase command
5EH = Flash Sector Protect reg select

Flash Status
FSTAT (FF8H - Read Only)

D7	D6	D5	D4	D3	D2	D1	D0

Flash Controller Status
00_0000 = Flash controller locked
00_0001 = First unlock received
00_0010 = Second unlock received
00_0011 = Flash controller unlocked
00_0100 = Flash Sector Protect register selected
00_1xxx = Programming in progress
01_0xxx = Page erase in progress
10_0xxx = Mass erase in progress

Reserved

Page Select
FPS (FF9H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Page Select [6:0]
Identifies the Flash memory page for Page Erase operation.

Information Area Enable

Table 12. Port Alternate Function Mapping (Continued)

Port	Pin	Mnemonic	Alternate Function Description
Port C	PC0	T1IN	Timer 1 Input
	PC1	T1OUT	Timer 1 Output
	PC2	$\overline{SS}$	SPI Slave Select
	PC3	SCK	SPI Serial Clock
	PC4	MOSI	SPI Master Out Slave In
	PC5	MISO	SPI Master In Slave Out

GPIO Interrupts

Many of GPIO port pins are used as interrupt sources. Some port pins are configured to generate an interrupt request on either the rising edge or falling edge of the pin input signal. Other port pin interrupts generate an interrupt when any edge occurs (both rising and falling). For more details on interrupts using the GPIO pins, see GPIO Port Pin Block Diagram on page 48.

GPIO Control Register Definitions

Four registers for each port provide access to GPIO control, input data, and output data. Table 13 lists the GPIO Port Registers and Sub-Registers. Use the Port A–C Address and Control Registers together to provide access to sub-registers for Port configuration and control.

Table 13. GPIO Port Registers and Sub-Registers

Port Register Mnemonic	Port Register Name
P*x*ADDR	Port A–C Address Register (selects sub-registers)
P*x*CTL	Port A–C Control Register (provides access to sub-registers)
P*x*IN	Port A–C Input Data Register
P*x*OUT	Port A–C Output Data Register
Port Sub-Register Mnemonic	**Port Register Name**
P*x*DD	Data Direction
P*x*AF	Alternate Function

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)

Interrupt Vectors and Priority

The interrupt controller supports three levels of interrupt priority. Level 3 is the highest priority, Level 2 is the second highest priority, and Level 1 is the lowest priority. If all the interrupts were enabled with identical interrupt priority (all as Level 2 interrupts), then interrupt priority would be assigned from highest to lowest as specified in Table 24. Level 3 interrupts always have higher priority than Level 2 interrupts which in turn always have higher priority than Level 1 interrupts. Within each interrupt priority level (Level 1, Level 2, or Level 3), priority is assigned as specified in Table 24. Reset, WDT interrupt (if enabled), and Illegal Instruction Trap always have highest priority.

Interrupt Assertion

Interrupt sources assert their interrupt requests for only a single system clock period (single pulse). When the interrupt request is acknowledged by the eZ8 CPU, the corresponding bit in the Interrupt Request Register is cleared until the next interrupt occurs. Writing a 0 to the corresponding bit in the Interrupt Request Register likewise clears the interrupt request.

Caution: *The following style of coding to clear bits in the Interrupt Request Registers is not recommended. All incoming interrupts received between execution of the first LDX command and the last LDX command is lost.*

Poor coding style resulting in lost interrupt requests:

```
LDX r0, IRQ0
AND r0, MASK
LDX IRQ0, r0
```

Note: *To avoid missing interrupts, the following style of coding to clear bits in the Interrupt Request 0 register is recommended:*

Good coding style that avoids lost interrupt requests:

```
ANDX IRQ0, MASK
```

Software Interrupt Assertion

Program code generates interrupts directly. Writing 1 to the desired bit in the Interrupt Request Register triggers an interrupt (assuming that interrupt is enabled). When the interrupt request is acknowledged by the eZ8 CPU, the bit in the Interrupt Request Register is automatically cleared to 0.

Caution: *The following style of coding to generate software interrupts by setting bits in the Interrupt Request Registers is not recommended. All incoming interrupts received between execution of the first LDX command and the last LDX command is lost.*

Table 34. IRQ2 Enable and Priority Encoding

IRQ2ENH[*x*]	IRQ2ENL[*x*]	Priority	Description
0	0	Disabled	Disabled
0	1	Level 1	Low
1	0	Level 2	Nominal
1	1	Level 3	High

where *x* indicates the register bits from 0 through 7.

Table 35. IRQ2 Enable High Bit Register (IRQ2ENH)

BITS	7	6	5	4	3	2	1	0
FIELD	Reserved				C3ENH	C2ENH	C1ENH	C0ENH
RESET	0							
R/W	R/W							
ADDR	FC7H							

Reserved—Must be 0.
C3ENH—Port C3 Interrupt Request Enable High Bit
C2ENH—Port C2 Interrupt Request Enable High Bit
C1ENH—Port C1 Interrupt Request Enable High Bit
C0ENH—Port C0 Interrupt Request Enable High Bit

Table 36. IRQ2 Enable Low Bit Register (IRQ2ENL)

BITS	7	6	5	4	3	2	1	0
FIELD	Reserved				C3ENL	C2ENL	C1ENL	C0ENL
RESET	0							
R/W	R/W							
ADDR	FC8H							

Reserved—Must be 0.
C3ENL—Port C3 Interrupt Request Enable Low Bit
C2ENL—Port C2 Interrupt Request Enable Low Bit
C1ENL—Port C1 Interrupt Request Enable Low Bit
C0ENL—Port C0 Interrupt Request Enable Low Bit

3. Write to the PWM High and Low Byte registers to set the PWM value.
4. Write to the Timer Reload High and Low Byte Registers to set the Reload value (PWM period). The Reload value must be greater than the PWM value.
5. If required, enable the timer interrupt and set the timer interrupt priority by writing to the relevant interrupt registers.
6. Configure the associated GPIO port pin for the Timer Output alternate function.
7. Write to the Timer Control Register to enable the timer and initiate counting.

The PWM period is given by the following equation.

$$\text{PWM Period (s)} = \frac{\text{Reload Value} \times \text{Prescale}}{\text{System Clock Frequency (Hz)}}$$

If an initial starting value other than 0001H is loaded into the Timer High and Low Byte Registers, the ONE-SHOT mode equation is used to determine the first PWM time-out period.

If TPOL is set to 0, the ratio of the PWM output High time to the total period is given by

$$\text{PWM Output High Time Ratio (\%)} = \frac{\text{Reload Value} - \text{PWM Value}}{\text{Reload Value}} \times 100$$

If TPOL is set to 1, the ratio of the PWM output High time to the total period is given by

$$\text{PWM Output High Time Ratio (\%)} = \frac{\text{PWM Value}}{\text{Reload Value}} \times 100$$

CAPTURE Mode

In CAPTURE mode, the current timer count value is recorded when the desired external Timer Input transition occurs. The Capture count value is written to the Timer PWM High and Low Byte Registers. The timer input is the system clock. The TPOL bit in the Timer Control Register determines if the Capture occurs on a rising edge or a falling edge of the Timer Input signal. When the Capture event occurs, an interrupt is generated and the timer continues counting.

The timer continues counting up to the 16-bit Reload value stored in the Timer Reload High and Low Byte registers. Upon reaching the Reload value, the timer generates an interrupt and continues counting.

Follow the steps below for configuring a timer for CAPTURE mode and initiating the count:

1. Write to the Timer Control Register to:
   – Disable the timer

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.

Table 43. Timer 0–1 PWM High Byte Register (TxPWMH)

BITS	7	6	5	4	3	2	1	0
FIELD	PWMH							
RESET	0							
R/W	R/W							
ADDR	F04H, F0CH							

Table 44. Timer 0–1 PWM Low Byte Register (TxPWML)

BITS	7	6	5	4	3	2	1	0
FIELD	PWML							
RESET	0							
R/W	R/W							
ADDR	F05H, F0DH							

PWMH and PWML—Pulse-Width Modulator High and Low Bytes
These two bytes, {PWMH[7:0], PWML[7:0]}, form a 16-bit value that is compared to the current 16-bit timer count. When a match occurs, the PWM output changes state. The PWM output value is set by the TPOL bit in the Timer Control Register (TxCTL) register.

The TxPWMH and TxPWML registers also store the 16-bit captured timer value when operating in CAPTURE or CAPTURE/COMPARE modes.

Timer 0–3 Control 0 Registers

The Timer 0–3 Control 0 (TxCTL0) registers (Table 45) allow cascading of the Timers.

Table 45. Timer 0–3 Control 0 Register (TxCTL0)

BITS	7	6	5	4	3	2	1	0
FIELD	Reserved			CSC	Reserved			
RESET	0							
R/W	R/W							
ADDR	F06H, F0EH, F16H, F1EH							

CSC—Cascade Timers
0 = Timer Input signal comes from the pin.
1 = For Timer 0, input signal is connected to Timer 1 output.
For Timer 1, input signal is connected to Timer 0 output.

TXRXSTATE	State Description
1_0111	10-bit addressing: Bit 0 (R/W) of 1st address byte
1_1000	10-bit addressing: Acknowledge state for 1st address byte
1_1001	10-bit addressing: Bit 7 of 2nd address byte 7-bit addressing: Bit 7 of address byte
1_1010	10-bit addressing: Bit 6 of 2nd address byte 7-bit addressing: Bit 6 of address byte
1_1011	10-bit addressing: Bit 5 of 2nd address byte 7-bit addressing: Bit 5 of address byte
1_1100	10-bit addressing: Bit 4 of 2nd address byte 7-bit addressing: Bit 4 of address byte
1_1101	10-bit addressing: Bit 3 of 2nd address byte 7-bit addressing: Bit 3 of address byte
1_1110	10-bit addressing: Bit 2 of 2nd address byte 7-bit addressing: Bit 2 of address byte
1_1111	10-bit addressing: Bit 1 of 2nd address byte 7-bit addressing: Bit 1 of address byte

I²C Diagnostic Control Register

The I²C Diagnostic register (Table 76) provides control over diagnostic modes. This register is a read/write register used for I²C diagnostics.

Table 76. I²C Diagnostic Control Register (I2CDIAG)

BITS	7	6	5	4	3	2	1	0
FIELD	Reserved							DIAG
RESET	0							
R/W	R							R/W
ADDR	F56H							

DIAG = Diagnostic Control Bit—Selects read back value of the Baud Rate Reload registers.

0 = Normal mode. Reading the Baud Rate High and Low Byte registers returns the baud rate reload value.

1 = Diagnostic mode. Reading the Baud Rate High and Low Byte registers returns the baud rate counter value.

OCD Auto-Baud Detector/Generator

To run over a range of baud rates (bits per second) with various system clock frequencies, the OCD contains an Auto-Baud Detector/Generator. After a reset, the OCD is idle until it receives data. The OCD requires that the first character sent from the host is the character 80H. The character 80H has eight continuous bits Low (one Start bit plus 7 data bits). The Auto-Baud Detector measures this period and sets the OCD Baud Rate Generator accordingly.

The Auto-Baud Detector/Generator is clocked by the system clock. The minimum baud rate is the system clock frequency divided by 512. For optimal operation, the maximum recommended baud rate is the system clock frequency divided by 8. The theoretical maximum baud rate is the system clock frequency divided by 4. This theoretical maximum is possible for low noise designs with clean signals. Table 92 lists minimum and recommended maximum baud rates for sample crystal frequencies.

Table 92. OCD Baud-Rate Limits

System Clock Frequency (MHz)	Recommended Maximum Baud Rate (kbps)	Minimum Baud Rate (Kbps)
20.0	2500	39.1
1.0	125.0	1.96
0.032768 (32 kHz)	4.096	0.064

If the OCD receives a Serial Break (nine or more continuous bits Low) the Auto-Baud Detector/Generator resets. The Auto-Baud Detector/Generator can then be reconfigured by sending 80H.

OCD Serial Errors

The OCD can detect any of the following error conditions on the DBG pin:

- Serial Break (a minimum of nine continuous bits Low)
- Framing Error (received STOP bit is Low)
- Transmit Collision (OCD and host simultaneous transmission detected by the OCD)

When the OCD detects one of these errors, it aborts any command currently in progress, transmits a Serial Break 4096 System Clock cycles long back to the host, and resets the Auto-Baud Detector/Generator. A Framing Error or Transmit Collision can be caused by the host sending a Serial Break to the OCD. Because of the open-drain nature of the interface, returning a Serial Break break back to the host only extends the length of the Serial Break if the host releases the Serial Break early.

```
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
```

On-Chip Peripheral AC and DC Electrical Characteristics

Table 99 provides information on the Power-On Reset and Voltage Brownout electrical characteristics.

Table 99. Power-On Reset and Voltage Brownout Electrical Characteristics and Timing

		T_A = -40 °C to 105 °C				
Symbol	**Parameter**	**Minimum**	**Typical[1]**	**Maximum**	**Units**	**Conditions**
V_{POR}	Power-On Reset Voltage Threshold	2.15	2.40	2.60	V	$V_{DD} = V_{POR}$
V_{VBO}	Voltage Brownout Reset Voltage Threshold	2.05	2.30	2.55	V	$V_{DD} = V_{VBO}$
	V_{POR} to V_{VBO} hysteresis	50	100	–	mV	
	Starting V_{DD} voltage to ensure valid POR	–	V_{SS}	–	V	
T_{ANA}	POR Analog Delay	–	50	–	µs	$V_{DD} > V_{POR}$; T_{POR} Digital Reset delay follows T_{ANA}
T_{POR}	POR Digital Delay	–	5.0	–	ms	50 WDT Oscillator cycles (10 kHz) + 16 System Clock cycles (20 MHz)
T_{VBO}	Voltage Brownout Pulse Rejection Period	–	10	–	µs	$V_{DD} < V_{VBO}$ to generate a Reset.
T_{RAMP}	Time for VDD to transition from V_{SS} to V_{POR} to ensure valid Reset	0.10	–	100	ms	

1 Data in the typical column is from characterization at 3.3 V and 25 °C. These values are provided for design guidance only and are not tested in production.

Table 125. Rotate and Shift Instructions

Mnemonic	Operands	Instruction
BSWAP	dst	Bit Swap
RL	dst	Rotate Left
RLC	dst	Rotate Left through Carry
RR	dst	Rotate Right
RRC	dst	Rotate Right through Carry
SRA	dst	Shift Right Arithmetic
SRL	dst	Shift Right Logical
SWAP	dst	Swap Nibbles

eZ8 CPU Instruction Summary

Table 126 summarizes the eZ8 CPU instructions. The table identifies the addressing modes employed by the instruction, the effect upon the Flags register, the number of CPU clock cycles required for the instruction fetch, and the number of CPU clock cycles required for the instruction execution.

Table 126. eZ8 CPU Instruction Summary

Assembly Mnemonic	Symbolic Operation	Address Mode		Opcode(s) (Hex)	Flags						Fetch Cycles	Instr. Cycles
		dst	src		C	Z	S	V	D	H		
ADC dst, src	dst ← dst + src + C	r	r	12	*	*	*	*	0	*	2	3
		r	Ir	13							2	4
		R	R	14							3	3
		R	IR	15							3	4
		R	IM	16							3	3
		IR	IM	17							3	4
ADCX dst, src	dst ← dst + src + C	ER	ER	18	*	*	*	*	0	*	4	3
		ER	IM	19							4	3

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.