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	5MHz
Connectivity	IrDA, UART/USART
Peripherals	Brown-out Detect/Reset, LED, POR, PWM, WDT
Number of I/O	16
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	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/z8f0113hh005sg

returns FFH. Writing to these unimplemented Program Memory addresses produces no effect. Table 6 describes the Program Memory maps for the Z8 Encore! XP F0823 Series products.

Table 6. Z8 Encore! XP F0823 Series Program Memory Maps

Program Memory Address (Hex)	Function
Z8F0823 and Z8F0813 Products	
0000–0001	Flash Option Bits
0002–0003	Reset Vector
0004–0005	WDT Interrupt Vector
0006–0007	Illegal Instruction Trap
0008–0037	Interrupt Vectors*
0038–003D	Oscillator Fail Traps*
003E–0FFF	Program Memory
Z8F0423 and Z8F0413 Products	
0000–0001	Flash Option Bits
0002–0003	Reset Vector
0004–0005	WDT Interrupt Vector
0006–0007	Illegal Instruction Trap
0008–0037	Interrupt Vectors*
0038–003D	Oscillator Fail Traps*
003E–0FFF	Program Memory
Z8F0223 and Z8F0213 Products	
0000–0001	Flash Option Bits
0002–0003	Reset Vector
0004–0005	WDT Interrupt Vector
0006–0007	Illegal Instruction Trap
0008–0037	Interrupt Vectors*
0038–003D	Oscillator Fail Traps*
003E–07FF	Program Memory

Note: *See the Trap and Interrupt Vectors in Order of Priority section on page 55 for a list of the interrupt vectors and traps.

Bit	Description (Continued)
[4] 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. For POR/Stop Mode Recover event values, please see Table 13.
[3:0]	**Reserved** These bits are reserved and must be programmed to 0000 when read.

Table 13. POR Indicator Values

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

Table 18. GPIO Port Registers and Subregisters (Continued)

Port Register Mnemonic	Port Register Name
P*x*HDE	High Drive Enable.
P*x*SMRE	Stop Mode Recovery Source Enable.
P*x*PUE	Pull-up Enable.
PxAFS1	Alternate Function Set 1.
PxAFS2	Alternate Function Set 2.

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 19).

Table 19. Port A–C GPIO Address Registers (P*x*ADDR)

Bit	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
Address	FD0H, FD4H, FD8H							

Bit	Description
[7:0] PADDR	**Port Address** The Port Address selects one of the subregisters accessible through the Port Control Register. See Table 20 for each subregister function.

Table 20. PADDR[7:0] Subregister Functions

PADDR[7:0]	Port Control Subregister 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.

Port A–C Output Data Register

The Port A–C Output Data Register (Table 31) controls the output data to the pins.

Table 31. 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	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
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). 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 the specific GPIO port pin number (7–0).	

LED Drive Enable Register

The LED Drive Enable Register, shown in Table 32, activates the controlled current drive. The Alternate Function Register has no control over the LED function; therefore, setting the Alternate Function Register to select the LED function is not required. LEDEN bits [7:0] correspond to Port C bits [7:0], respectively.

Table 32. LED Drive Enable (LEDEN)

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	LEDEN[7:0]							
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
Address	F82H							

Bit	**Description**
[7:0] LEDEN	**LED Drive Enable** These bits determine which Port C pins are connected to an internal current sink. 0 = Tristate the Port C pin. 1= Connect controlled current sink to the Port C pin.

Bit	Description (Continued)
[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.
[2:1]	**Reserved** These bits are reserved and must be programmed to 00.
[0] ADCI	**ADC Interrupt Request** 0 = No interrupt request is pending for the ADC. 1 = An interrupt request from the ADC is awaiting service.

Interrupt Request 1 Register

The Interrupt Request 1 (IRQ1) register (Table 37) 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 37. Interrupt Request 1 Register (IRQ1)

Bit	**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
Address	FC3H							

Bit	Description
[7] PA7VI	**Port A7 Interrupt Request** 0 = No interrupt request is pending for GPIO Port A. 1 = An interrupt request from GPIO Port A.
[6] 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.
[5:0] 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.

Note: x indicates the specific GPIO Port pin number (0–5).

Table 53. Timer 0–1 Reload High Byte Register (TxRH)

Bit	7	6	5	4	3	2	1	0
Field	TRH							
RESET	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Address	F02H, F0AH							

Table 54. Timer 0–1 Reload Low Byte Register (TxRL)

Bit	7	6	5	4	3	2	1	0
Field	TRL							
RESET	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Address	F03H, F0BH							

Bit	Description
[7]	
[6]	
[5]	
[4]	
[3]	
[2]	
[1]	
[0]	

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.

Watchdog Timer Refresh

When first enabled, the WDT is loaded with the value in the Watchdog Timer Reload registers. The Watchdog Timer counts down to `000000H` unless a `WDT` instruction is executed by the eZ8 CPU. Execution of the `WDT` instruction causes the down counter to be reloaded with the WDT reload value stored in the Watchdog Timer Reload registers. Counting resumes following the reload operation.

When Z8 Encore! XP F0823 Series devices are operating in DEBUG Mode (using the OCD), the Watchdog Timer is continuously refreshed to prevent any Watchdog Timer time-outs.

Watchdog Timer Time-Out Response

The Watchdog Timer times out when the counter reaches `000000H`. A time-out of the Watchdog Timer generates either an interrupt or a system reset. The WDT_RES Flash Option Bit determines the time-out response of the Watchdog Timer. For information about programming of the WDT_RES Flash Option Bit, see **the** Flash Option Bits chapter on page 146.

WDT Interrupt in Normal Operation

If configured to generate an interrupt when a time-out occurs, the Watchdog Timer issues an interrupt request to the interrupt controller and sets the `WDT` status bit in the Watchdog Timer Control Register. If interrupts are enabled, the eZ8 CPU responds to the interrupt request by fetching the Watchdog Timer interrupt vector and executing code from the vector address. After time-out and interrupt generation, the Watchdog Timer counter rolls over to its maximum value of `FFFFFH` and continues counting. The Watchdog Timer counter is not automatically returned to its Reload Value.

The Reset Status Register (see the Reset Status Register section on page 28) must be read before clearing the WDT interrupt. This read clears the WDT time-out Flag and prevents further WDT interrupts for immediately occurring.

WDT Interrupt in STOP Mode

If configured to generate an interrupt when a time-out occurs and F0823 Series are in STOP Mode, the Watchdog Timer automatically initiates a Stop Mode Recovery and generates an interrupt request. Both the WDT status bit and the `STOP` bit in the Watchdog Timer Control Register are set to 1 following a WDT time-out in STOP Mode. For more information about Stop Mode Recovery, see **the** Reset and Stop Mode Recovery chapter on page 21.

If interrupts are enabled, following completion of the Stop Mode Recovery the eZ8 CPU responds to the interrupt request by fetching the Watchdog Timer interrupt vector and executing code from the vector address.

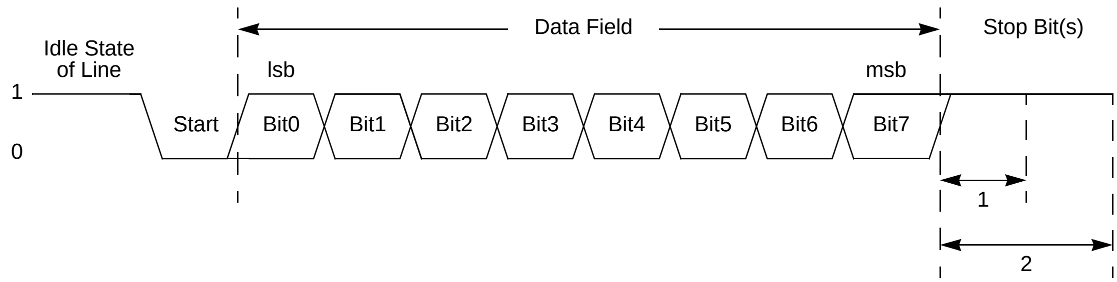

Figure 11. UART Asynchronous Data Format without Parity

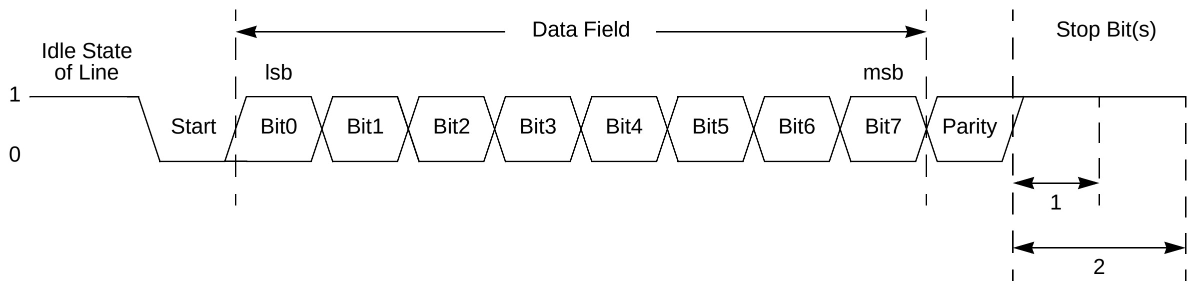

Figure 12. UART Asynchronous Data Format with Parity

Transmitting Data Using the Polled Method

Observe the following steps to transmit data using the polled method of operation:

1. Write to the UART Baud Rate High and Low Byte registers to set the required baud rate.
2. Enable the UART pin functions by configuring the associated GPIO port pins for alternate function operation.
3. Write to the UART Control 1 Register, if MULTIPROCESSOR Mode is appropriate, to enable MULTIPROCESSOR (9-bit) Mode functions.
4. Set the Multiprocessor Mode Select (`MPEN`) bit to enable MULTIPROCESSOR Mode.
5. Write to the UART Control 0 Register to:
   – Set the transmit enable bit (`TEN`) to enable the UART for data transmission
   – Set the parity enable bit (`PEN`), if parity is appropriate and MULTIPROCESSOR Mode is not enabled, and select either even or odd parity (`PSEL`)

7. Return to Step 4 to receive additional data.

Receiving Data Using the Interrupt-Driven Method

The UART Receiver interrupt indicates the availability of new data (as well as error conditions). Observe the following steps 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:

Bit	Description (Continued)
[1] STOP	**Stop Bit Select** 0 = The transmitter sends one stop bit. 1 = The transmitter sends two stop bits.
[0] LBEN	**Loop Back Enable** 0 = Normal operation. 1 = All transmitted data is looped back to the receiver.

Table 69. UART Control 1 Register (U0CTL1)

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	MPMD[1]	MPEN	MPMD[0]	MPBT	DEPOL	BRGCTL	RDAIRQ	IREN
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
Address	F43H							

Bit	Description
[7,5] MPMD[1:0]	**MULTIPROCESSOR Mode** If MULTIPROCESSOR (9-bit) Mode is enabled. 00 = The UART generates an interrupt request on all received bytes (data and address). 01 = The UART generates an interrupt request only on received address bytes. 10 = The UART generates an interrupt request when a received address byte matches the value stored in the Address Compare Register and on all successive data bytes until an address mismatch occurs. 11 = The UART generates an interrupt request on all received data bytes for which the most recent address byte matched the value in the Address Compare Register.
[6] MPEN	**MULTIPROCESSOR (9-bit) Enable** This bit is used to enable MULTIPROCESSOR (9-bit) Mode. 0 = Disable MULTIPROCESSOR (9-bit) Mode. 1 = Enable MULTIPROCESSOR (9-bit) Mode.
[4] MPBT	**Multiprocessor Bit Transmit** This bit is applicable only when MULTIPROCESSOR (9-bit) Mode is enabled. The 9th bit is used by the receiving device to determine if the data byte contains address or data information. 0 = Send a 0 in the multiprocessor bit location of the data stream (data byte). 1 = Send a 1 in the multiprocessor bit location of the data stream (address byte).
[3] DEPOL	**Driver Enable Polarity** 0 = DE signal is Active High. 1 = DE signal is Active Low.

The UART data rate is calculated using the following equation:

$$\text{UART Baud Rate (bits/s)} = \frac{\text{System Clock Frequency (Hz)}}{16 \times \text{UART Baud Rate Divisor Value}}$$

For a given UART data rate, calculate the integer baud rate divisor value using the following equation:

$$\text{UART Baud Rate Divisor Value (BRG)} = \text{Round}\left(\frac{\text{System Clock Frequency (Hz)}}{16 \times \text{UART Data Rate (bits/s)}}\right)$$

The baud rate error relative to the acceptable baud rate is calculated using the following equation:

$$\text{UART Baud Rate Error (\%)} = 100 \times \left(\frac{\text{Actual Data Rate} - \text{Desired Data Rate}}{\text{Desired Data Rate}}\right)$$

For reliable communication, the UART baud rate error must never exceed five percent. Table 73 provides information about data rate errors for a 5.5296 MHz System Clock.

Table 73. UART Baud Rates

5.5296 MHz System Clock			
Acceptable Rate (kHz)	**BRG Divisor (Decimal)**	**Actual Rate (kHz)**	**Error (%)**
1250.0	N/A	N/A	N/A
625.0	N/A	N/A	N/A
250.0	1	345.6	38.24
115.2	3	115.2	0.00
57.6	6	57.6	0.00
38.4	9	38.4	0.00
19.2	18	19.2	0.00
9.60	36	9.60	0.00
4.80	72	4.80	0.00
2.40	144	2.40	0.00
1.20	288	1.20	0.00
0.60	576	0.60	0.00
0.30	1152	0.30	0.00

Software Compensation Procedure

The value read from the ADC high and low byte registers are uncompensated. The user mode software must apply gain and offset correction to this uncompensated value for maximum accuracy. The following formula yields the compensated value:

$$ADC_{comp} = (ADC_{uncomp} - OFFCAL) + ((ADC_{uncomp} - OFFCAL)*GAINCAL)/2^{16}$$

where GAINCAL is the gain calibration byte, OFFCAL is the offset calibration byte and ADC_{uncomp} is the uncompensated value read from the ADC. The OFFCAL value is in two's complement format, as are the compensated and uncompensated ADC values.

Note: The offset compensation is performed first, followed by the gain compensation. One bit of resolution is lost because of rounding on both the offset and gain computations. As a result the ADC registers read back 13 bits: 1 sign bit, two calibration bits lost to rounding and 10 data bits. Also note that in the second term, the multiplication must be performed before the division by 2^{16}. Otherwise, the second term evaluates to zero incorrectly.

Caution: Although the ADC can be used without the gain and offset compensation, it does exhibit non-unity gain. Designing the ADC with sub-unity gain reduces noise across the ADC range but requires the ADC results to be scaled by a factor of 8/7.

ADC Control Register Definitions

The following sections define the ADC Control registers.

ADC Control Register 0

The ADC Control Register selects the analog input channel and initiates the analog-to-digital conversion.

Flash Status Register

The Flash Status Register indicates the current state of the Flash Controller. This register can be read at any time. The read-only Flash Status Register shares its Register File address with the write-only Flash Control Register.

Table 82. Flash Status Register (FSTAT)

Bit	7	6	5	4	3	2	1	0
Field	Reserved		FSTAT					
RESET	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R
Address	FF8H							

Bit	Description
[7:6]	**Reserved** These bits are reserved and must be programmed to 0 when read.
[5:0] 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.

In the following list of OCD Commands, data and commands sent from the host to the OCD are identified by `'DBG ← Command/Data'`. Data sent from the OCD back to the host is identified by `'DBG → Data'`.

Read OCD Revision (00H). The Read OCD Revision command determines the version of the OCD. If OCD commands are added, removed, or changed, this revision number changes.

```
DBG ← 00H
DBG → OCDRev[15:8] (Major revision number)
DBG → OCDRev[7:0] (Minor revision number)
```

Read OCD Status Register (02H). The Read OCD Status Register command reads the OCDSTAT Register.

```
DBG ← 02H
DBG → OCDSTAT[7:0]
```

Read Runtime Counter (03H). The Runtime Counter counts system clock cycles in between breakpoints. The 16-bit Runtime Counter counts up from `0000H` and stops at the maximum count of `FFFFH`. The Runtime Counter is overwritten during the Write Memory, Read Memory, Write Register, Read Register, Read Memory CRC, Step Instruction, Stuff Instruction, and Execute Instruction commands.

```
DBG ← 03H
DBG → RuntimeCounter[15:8]
DBG → RuntimeCounter[7:0]
```

Write OCD Control Register (04H). The Write OCD Control Register command writes the data that follows to the OCDCTL register. When the Flash Read Protect Option Bit is enabled, the DBGMODE bit (`OCDCTL`[7]) can only be set to 1, it cannot be cleared to 0 and the only method of returning the device to normal operating mode is to reset the device.

```
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 (PC). If the device is not in DEBUG Mode or if the Flash Read Protect Option bit is enabled, the Program Counter (PC) values are discarded.

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

Table 118. eZ8 CPU Instruction Summary (Continued)

Assembly Mnemonic	Symbolic Operation	Address Mode dst	Address Mode src	Opcode(s) (Hex)	Flags C	Z	S	V	D	H	Fetch Cycles	Instr. Cycles
LD dst, rc	dst ← src	r	IM	0C-FC	–	–	–	–	–	–	2	2
		r	X(r)	C7							3	3
		X(r)	r	D7							3	4
		r	Ir	E3							2	3
		R	R	E4							3	2
		R	IR	E5							3	4
		R	IM	E6							3	2
		IR	IM	E7							3	3
		Ir	r	F3							2	3
		IR	R	F5							3	3
LDC dst, src	dst ← src	r	Irr	C2	–	–	–	–	–	–	2	5
		Ir	Irr	C5							2	9
		Irr	r	D2							2	5
LDCI dst, src	dst ← src r ← r + 1 rr ← rr + 1	Ir	Irr	C3	–	–	–	–	–	–	2	9
		Irr	Ir	D3							2	9
LDE dst, src	dst ← src	r	Irr	82	–	–	–	–	–	–	2	5
		Irr	r	92							2	5
LDEI dst, src	dst ← src r ← r + 1 rr ← rr + 1	Ir	Irr	83	–	–	–	–	–	–	2	9
		Irr	Ir	93							2	9
LDWX dst, src	dst ← src	ER	ER	1FE8	–	–	–	–	–	–	5	4

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

Table 119. Opcode Map Abbreviations

Abbreviation	Description	Abbreviation	Description
b	Bit position	IRR	Indirect Register Pair
cc	Condition code	p	Polarity (0 or 1)
X	8-bit signed index or displacement	r	4-bit Working Register
DA	Destination address	R	8-bit register
ER	Extended Addressing register	r1, R1, Ir1, Irr1, IR1, rr1, RR1, IRR1, ER1	Destination address
IM	Immediate data value	r2, R2, Ir2, Irr2, IR2, rr2, RR2, IRR2, ER2	Source address
Ir	Indirect Working Register	RA	Relative
IR	Indirect register	rr	Working Register Pair
Irr	Indirect Working Register Pair	RR	Register Pair

Lower Nibble (Hex)

Upper Nibble (Hex)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	1.1 **BRK**	2.2 **SRP** IM	2.3 **ADD** r1,r2	2.4 **ADD** r1,Ir2	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,Ir2	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,Ir2	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
3	2.2 **DEC** R1	2.3 **DEC** IR1	2.3 **SBC** r1,r2	2.4 **SBC** r1,Ir2	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,Ir2	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,Ir2	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,Ir2	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,Ir2	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,Irr2	2.9 **LDEI** Ir1,Irr2	3.2 **LDX** r1,ER2	3.3 **LDX** Ir1,ER2	3.4 **LDX** IRR2,R1	3.5 **LDX** IRR2,IR1	3.4 **LDX** r1,rr2,X	3.4 **LDX** rr1,r2,X						1.2 **DI**
9	2.2 **RL** R1	2.3 **RL** IR1	2.5 **LDE** r2,Irr1	2.9 **LDEI** Ir2,Irr1	3.2 **LDX** r2,ER1	3.3 **LDX** Ir2,ER1	3.4 **LDX** R2,IRR1	3.5 **LDX** IR2,IRR1	3.3 **LEA** r1,r2,X	3.5 **LEA** rr1,rr2,X						1.2 **EI**
A	2.5 **INCW** RR1	2.6 **INCW** IRR1	2.3 **CP** r1,r2	2.4 **CP** r1,Ir2	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,Ir2	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,Irr2	2.9 **LDCI** Ir1,Irr2	2.3 **JP** IRR1	2.9 **LDC** Ir1,Irr2		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,Irr1	2.9 **LDCI** Ir2,Irr1	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,Ir2	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** Ir1,r2	2.8 **MULT** RR1	3.3 **LD** R2,IR1	3.3 **BTJ** p,b,r1,X	3.4 **BTJ** p,b,Ir1,X								

Figure 27. First Opcode Map

Lower Nibble (Hex)

Upper 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	3 ,															
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									5, 4 **LDWX** ER2,ER1							
F																

Figure 28. Second Opcode Map after 1FH

S

T