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	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 3.6V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	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/z8f0811hh020sc

Interrupt Request 0
IRQ0 (FC0H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

- D0 — ADC Interrupt Request
- D1 — SPI Interrupt Request
- D2 — I2C Interrupt Request
- D3 — UART 0 Transmitter Interrupt
- D4 — UART 0 Receiver Interrupt
- D5 — Timer 0 Interrupt Request
- D6 — Timer 1 Interrupt Request
- D7 — Reserved

For all of the above peripherals:
0 = Peripheral IRQ is not pending
1 = Peripheral IRQ is awaiting service

IRQ0 Enable High Bit
IRQ0ENH (FC1H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

- D0 — ADC IRQ Enable Hit Bit
- D1 — SPI IRQ Enable High Bit
- D2 — I2C IRQ Enable High Bit
- D3 — UART 0 Transmitter IRQ
- D4 — UART 0 Receiver IRQ Enable
- D5 — Timer 0 IRQ Enable High Bit
- D6 — Timer 1 IRQ Enable High Bit
- D7 — Reserved

Interrupt Request 2
IRQ2 (FC6H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Port C Pin Interrupt Request
0 = IRQ from corresponding pin [3:0] is not pending
1 = IRQ from corresponding pin [3:0] is awaiting service

Reserved

IRQ2 Enable High Bit
IRQ2ENH (FC7H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Port C Pin IRQ Enable High

Reserved

IRQ2 Enable Low Bit
IRQ2ENL (FC8H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Port C Pin IRQ Enable Low

Reserved

Interrupt Control
IRQCTL (FCFH - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Reserved

Interrupt Request Enable
0 = Interrupts are disabled
1 = Interrupts are enabled

Register Pointer
RP (FFDH- Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Working Register Group

Stack Pointer High Byte
SPH (FFEH - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Stack Pointer [15:8]

Stack Pointer Low Byte
SPL (FFFH - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Stack Pointer [7:0]

Table 24. Interrupt Vectors in Order of Priority (Continued)

Priority	Program Memory Vector Address	Interrupt Source
	0008H	Reserved
	000AH	Timer 1
	000CH	Timer 0
	000EH	UART 0 receiver
	0010H	UART 0 transmitter
	0012H	I^2C
	0014H	SPI
	0016H	ADC
	0018H	Port A7, rising or falling input edge
	001AH	Port A6, rising or falling input edge
	001CH	Port A5, rising or falling input edge
	001EH	Port A4, rising or falling input edge
	0020H	Port A3, rising or falling input edge
	0022H	Port A2, rising or falling input edge
	0024H	Port A1, rising or falling input edge
	0026H	Port A0, rising or falling input edge
	0028H	Reserved
	002AH	Reserved
	002CH	Reserved
	002EH	Reserved
	0030H	Port C3, both input edges
	0032H	Port C2, both input edges
	0034H	Port C1, both input edges
Lowest	0036H	Port C0, both input edges

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

  - Set the prescale value
  - If using the Timer Output alternate function, set the initial output level (High or Low).
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 Reload value.
4. If desired, enable the timer interrupt 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 ONE-SHOT mode, the system clock always provides the timer input. The timer period is given by the following equation:

$$\text{ONE-SHOT Mode Time-Out Period (s)} = \frac{(\text{Reload Value} - \text{Start Value})\text{xPrescale}}{\text{System Clock Frequency (Hz)}}$$

CONTINUOUS Mode

In CONTINUOUS mode, 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. Upon 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. 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 timer Reload.

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

1. Write to the Timer Control Register to:
   - Disable the timer
   - Configure the timer for CONTINUOUS mode
   - Set the prescale value.
   - If using the Timer Output alternate function, set the initial output level (High or Low).
2. Write to the Timer High and Low Byte registers to set the starting count value (usually `0001H`). This only affects the first pass in CONTINUOUS mode. After the first timer Reload in CONTINUOUS mode, counting always begins at the reset value of `0001H`.
3. Write to the Timer Reload High and Low Byte Registers to set the Reload value.
4. If desired, enable the timer interrupt 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.

When the UART is disabled, the BRG functions as a basic 16-bit timer with interrupt on time-out. Follow the steps below to configure the BRG as a timer with interrupt on time-out:

1. Disable the UART by clearing the `REN` and `TEN` bits in the UART Control 0 Register to 0.
2. Load the desired 16-bit count value into the UART Baud Rate High and Low Byte Registers.
3. Enable the BRG timer function and associated interrupt by setting the `BKGCTL` bit in the UART Control 1 Register to 1.

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

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

UART Control Register Definitions

The UART Control Registers support the UART and the associated Infrared Encoder/Decoders. See Infrared Encoder/Decoder on page 109 for more information on the infrared operation.

UART Transmit Data Register

Data bytes written to the UART Transmit Data Register (Table 52) are shifted out on the TXD*x* pin. The Write-only UART Transmit Data Register shares a Register File address with the Read-only UART Receive Data Register.

Table 52. UART Transmit Data Register (U0TXD)

BITS	7	6	5	4	3	2	1	0
FIELD	TXD							
RESET	X	X	X	X	X	X	X	X
R/W	W	W	W	W	W	W	W	W
ADDR	F40H							

TXD—Transmit Data
UART transmitter data byte to be shifted out through the TXD*x* pin.

MPBT—Multiprocessor Bit Transmit
This bit is applicable only when Multiprocessor (9-bit) mode is enabled.
0 = Send a 0 in the multiprocessor bit location of the data stream (9th bit).
1 = Send a 1 in the multiprocessor bit location of the data stream (9th bit).

DEPOL—Driver Enable Polarity
0 = DE signal is Active High.
1 = DE signal is Active Low.

BRGCTL—Baud Rate Control
This bit causes different UART behavior depending on whether the UART receiver is enabled (REN = 1 in the UART Control 0 Register).

When the UART receiver is not enabled, this bit determines whether the BRG will issue interrupts.
0 = Reads from the Baud Rate High and Low Byte registers return the BRG Reload Value
1 = The BRG generates a receive interrupt when it counts down to zero. Reads from the Baud Rate High and Low Byte registers return the current BRG count value.

When the UART receiver is enabled, this bit allows reads from the Baud Rate Registers to return the BRG count value instead of the Reload Value.
0 = Reads from the Baud Rate High and Low Byte registers return the BRG Reload Value.
1 = Reads from the Baud Rate High and Low Byte registers return the current BRG count value. Unlike the Timers, there is no mechanism to latch the High Byte when the Low Byte is read.

$\overline{\text{RDAIRQ}}$—Receive Data Interrupt $\overline{\text{Enable}}$
0 = Received data and receiver errors generates an interrupt request to the Interrupt Controller.
1 = Received data does not generate an interrupt request to the Interrupt Controller. Only receiver errors generate an interrupt request.

IREN—Infrared Encoder/Decoder Enable
0 = Infrared Encoder/Decoder is disabled. UART operates normally operation.
1 = Infrared Encoder/Decoder is enabled. The UART transmits and receives data through the Infrared Encoder/Decoder.

UART Address Compare Register

The UART Address Compare register stores the multi-node network address of the UART. When the MPMD[1] bit of UART Control Register 0 is set, all incoming address bytes will be compared to the value stored in the Address Compare register. Receive interrupts and RDA assertions will only occur in the event of a match.

of minus four baud rate clocks to plus eight baud rate clocks around the expected time of an incoming pulse. If an incoming pulse is detected inside this window this process is repeated. If the incoming data is a logical 1 (no pulse), the Endec returns to the initial state and waits for the next falling edge. As each falling edge is detected, the Endec clock counter is reset, resynchronizing the Endec to the incoming signal. This procedure allows the Endec to tolerate jitter and baud rate errors in the incoming data stream. Resynchronizing the Endec does not alter the operation of the UART, which ultimately receives the data. The UART is only synchronized to the incoming data stream when a Start bit is received.

Infrared Endec Control Register Definitions

All Infrared Endec configuration and status information is set by the UART control registers as defined in UART Control Register Definitions on page 100.

! Caution: *To prevent spurious signals during IrDA data transmission, set the* `IREN` *bit in the UART Control 1 register to 1 to enable the Infrared Endec before enabling the GPIO Port alternate function for the corresponding pin.*

Operation

The I^{2}C Controller operates in MASTER mode to transmit and receive data. Only a single master is supported. Arbitration between two masters must be accomplished in software. I^{2}C supports the following operations:

- Master transmits to a 7-bit Slave
- Master transmits to a 10-bit Slave
- Master receives from a 7-bit Slave
- Master receives from a 10-bit Slave

SDA and SCL Signals

I^{2}C sends all addresses, data and acknowledge signals over the SDA line, most-significant bit first. SCL is the common clock for the I^{2}C Controller. When the SDA and SCL pin alternate functions are selected for their respective GPIO ports, the pins are automatically configured for open-drain operation.

The master (I^{2}C) is responsible for driving the SCL clock signal, although the clock signal becomes skewed by a slow slave device. During the low period of the clock, the slave pulls the SCL signal Low to suspend the transaction. The master releases the clock at the end of the low period and notices that the clock remains low instead of returning to a high level. When the slave releases the clock, the I^{2}C Controller continues the transaction. All data is transferred in bytes and there is no limit to the amount of data transferred in one operation. When transmitting data or acknowledging read data from the slave, the SDA signal changes in the middle of the low period of SCL and is sampled in the middle of the high period of SCL.

I^{2}C Interrupts

The I^{2}C Controller contains four sources of interrupts—Transmit, Receive, Not Acknowledge, and Baud Rate Generator. These four interrupt sources are combined into a single interrupt request signal to the interrupt controller. The Transmit Interrupt is enabled by the IEN and TXI bits of the control register. The Receive and Not Acknowledge interrupts are enabled by the IEN bit of the control register. BRG interrupt is enabled by the BIRQ and IEN bits of the control register.

Not Acknowledge interrupts occur when a Not Acknowledge condition is received from the slave or sent by the I^{2}C Controller and neither the `START` or `STOP` bit is set. The Not Acknowledge event sets the NCKI bit of the I^{2}C Status Register and can only be cleared by setting the `START` or `STOP` bit in the I^{2}C Control Register. When this interrupt occurs, the I^{2}C Controller waits until either the STOP or START bit is set before performing any action. In an ISR, the NCKI bit should always be checked prior to servicing transmit or receive interrupt conditions because it indicates the transaction is being terminated.

Receive interrupts occur when a byte of data has been received by the I^2C Controller (Master reading data from Slave). This procedure sets the RDRF bit of the I^2C Status Register. The RDRF bit is cleared by reading the I^2C Data Register. The RDRF bit is set during the acknowledge phase. The I^2C Controller pauses after the acknowledge phase until the receive interrupt is cleared before performing any other action.

Transmit interrupts occur when the TDRE bit of the I^2C Status register sets and the TXI bit in the I^2C Control Register is set. Transmit interrupts occur under the following conditions when the Transmit Data Register is empty:

- The I^2C Controller is enabled
- The first bit of the byte of an address is shifting out and the `RD` bit of the I^2C Status register is deasserted.
- The first bit of a 10-bit address shifts out.
- The first bit of write data shifts out.

Note: *Writing to the I^2C Data Register always clears the TRDE bit to 0. When TDRE is asserted, the I^2C Controller pauses at the beginning of the Acknowledge cycle of the byte currently shifting out until the data register is written with the next value to send or the STOP or START bits are set indicating the current byte is the last one to send.*

The fourth interrupt source is the BRG. If the I^2C Controller is disabled (IEN bit in the I2CCTL Register = 0) and the BIRQ bit in the I2CCTL Register = 1,
an interrupt is generated when the BRG counts down to 1. This allows the I^2C Baud Rate Generator to be used by software as a general purpose timer when IEN = 0.

Software Control of I^2C Transactions

Software controls I^2C transactions by using the I^2C Controller interrupt, by polling the I^2C Status register or by DMA. Note that not all products include a DMA Controller.

To use interrupts, the I^2C interrupt must be enabled in the Interrupt Controller. The TXI bit in the I^2C Control Register must be set to enable transmit interrupts.

To control transactions by polling, the interrupt bits (TDRE, RDRF and NCKI) in the I^2C Status Register should be polled. The TDRE bit asserts regardless of the state of the TXI bit.

Either or both transmit and receive data movement can be controlled by the DMA Controller. The DMA Controller channel(s) must be initialized to select the I^2C transmit and receive requests. Transmit DMA requests require that the TXI bit in the I^2C Control Register be set.

! **Caution:** *A transmit (write) DMA operation hangs if the slave responds with a Not Acknowledge before the last byte has been sent. After receiving the Not Acknowledge, the I^2C Controller sets the NCKI bit in the Status Register and pauses until either the STOP or*

5. After the first bit has been shifted out, a Transmit Interrupt is asserted.
6. Software responds by writing the lower eight bits of address to the I^{2}C Data Register.
7. The I^{2}C Controller completes shifting of the two address bits and a 0 (write).
8. If the I^{2}C Slave acknowledges the first address byte by pulling the SDA signal low during the next high period of SCL, the I^{2}C Controller sets the `ACK` bit in the I^{2}C Status register. Continue with step 9.

   If the slave does not acknowledge the first address byte, the I^{2}C Controller sets the `NCKI` bit and clears the ACK bit in the I^{2}C Status register. 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).
9. The I^{2}C Controller loads the I^{2}C Shift register with the contents of the I^{2}C Data Register (second address byte).
10. The I^{2}C Controller shifts out the second address byte. After the first bit is shifted, the I^{2}C Controller generates a Transmit Interrupt.
11. Software responds by setting the `START` bit of the I^{2}C Control Register to generate a repeated START and by clearing the TXI bit.
12. Software responds by writing `11110B` followed by the 2-bit Slave address and a 1 (read) to the I^{2}C Data Register.
13. If only one byte is to be read, software sets the NAK bit of the I^{2}C Control Register.
14. After the I^{2}C Controller shifts out the 2nd address byte, the I^{2}C Slave sends an acknowledge by pulling the SDA signal low during the next high period of SCL, the I^{2}C Controller sets the `ACK` bit in the I^{2}C Status register. Continue with step 15.

    If the slave does not acknowledge the second address byte, the I^{2}C Controller sets the `NCKI` bit and clears the ACK bit in the I^{2}C Status register. 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 the following steps).
15. The I^{2}C Controller sends the repeated START condition.
16. The I^{2}C Controller loads the I^{2}C Shift register with the contents of the I^{2}C Data Register (third address transfer).
17. The I^{2}C Controller sends `11110B` followed by the two most significant bits of the slave read address and a 1 (read).
18. The I^{2}C Slave sends an acknowledge by pulling the SDA signal Low during the next high period of SCL.
    If the slave were to Not Acknowledge at this point (this should not happen because the slave did acknowledge the first two address bytes), software would respond by setting the STOP and FLUSH bits and clearing the TXI bit. The I2C Controller sends the

Table 97. DC Characteristics (Continued)

Symbol	Parameter	T_A = -40 °C to 105 °C			Units	Conditions
		Minimum	Typical	Maximum		
V_{RAM}	RAM Data Retention	0.7	–	–	V	
I_{IL}	Input Leakage Current	-5	–	+5	µA	V_{DD} = 3.6 V; V_{IN} = VDD or VSS[1]
I_{TL}	Tri-State Leakage Current	-5	–	+5	µA	V_{DD} = 3.6 V
C_{PAD}	GPIO Port Pad Capacitance	–	8.0[2]	–	pF	
C_{XIN}	XIN Pad Capacitance	–	8.0[2]	–	pF	
C_{XOUT}	XOUT Pad Capacitance	–	9.5[2]	–	pF	
I_{PU1}	Weak Pull-up Current	9	20	50	µA	VDD = 2.7–3.6 V. T_A = 0 °C to +70 °C
I_{PU2}	Weak Pull-up Current	7	20	75	µA	VDD = 2.7–3.6 V. T_A = -40 °C to +105 °C

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

Figure 41 on page 189 displays the typical active mode current consumption while operating at 25 ºC, 3.3 V, versus the system clock frequency. All GPIO pins are configured as outputs and driven High.

Figure 44 displays the maximum HALT mode current consumption across the full operating temperature range of the device and versus the system clock frequency. All GPIO pins are configured as outputs and driven High.

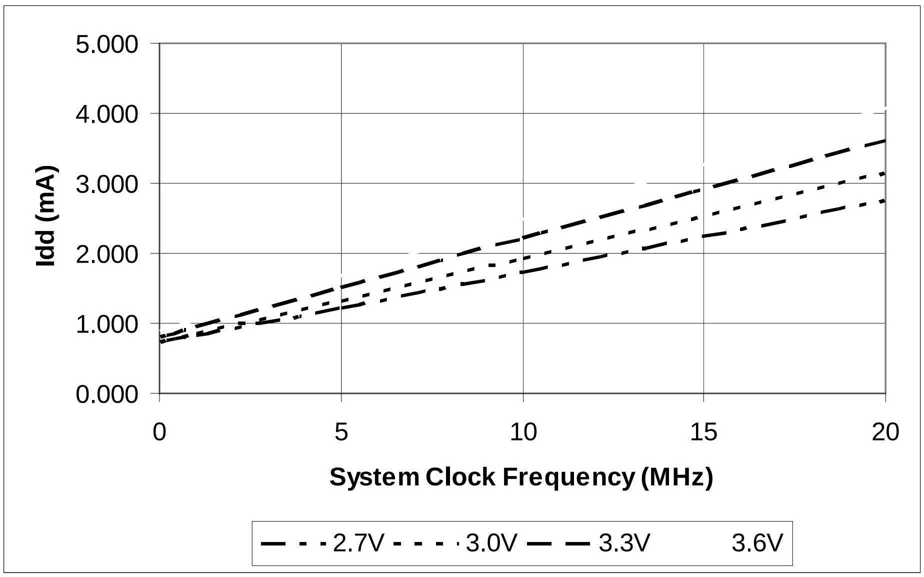


Figure 44. Maximum HALT Mode I_{CC} Versus System Clock Frequency

Table 121. CPU Control Instructions

Mnemonic	Operands	Instruction
CCF	—	Complement Carry Flag
DI	—	Disable Interrupts
EI	—	Enable Interrupts
HALT	—	HALT Mode
NOP	—	No Operation
RCF	—	Reset Carry Flag
SCF	—	Set Carry Flag
SRP	src	Set Register Pointer
STOP	—	STOP Mode
WDT	—	Watchdog Timer Refresh

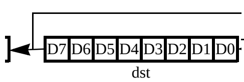
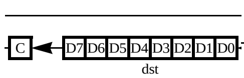
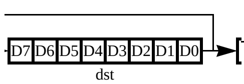
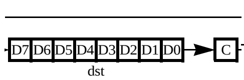
Table 122. Load Instructions

Mnemonic	Operands	Instruction
CLR	dst	Clear
LD	dst, src	Load
LDC	dst, src	Load Constant to/from Program Memory
LDCI	dst, src	Load Constant to/from Program Memory and Auto-Increment Addresses
LDE	dst, src	Load External Data to/from Data Memory
LDEI	dst, src	Load External Data to/from Data Memory and Auto-Increment Addresses
LDX	dst, src	Load using Extended Addressing
LEA	dst, X(src)	Load Effective Address
POP	dst	Pop
POPX	dst	Pop using Extended Addressing
PUSH	src	Push
PUSHX	src	Push using Extended Addressing

Table 126. eZ8 CPU Instruction Summary (Continued)

Assembly Mnemonic	Symbolic Operation	Address Mode dst	Address Mode src	Opcode(s) (Hex)	Flags C	Flags Z	Flags S	Flags V	Flags D	Flags 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

Table 126. eZ8 CPU Instruction Summary (Continued)

Assembly Mnemonic	Symbolic Operation	Address Mode dst	Address Mode src	Opcode(s) (Hex)	Flags C	Flags Z	Flags S	Flags V	Flags D	Flags H	Fetch Cycles	Instr. Cycles
POPX dst	dst ← @SP SP ← SP + 1	ER		D8	-	-	-	-	-	-	3	2
PUSH src	SP ← SP – 1 @SP ← src	R		70	-	-	-	-	-	-	2	2
		IR		71							2	3
PUSHX src	SP ← SP – 1 @SP ← src	ER		C8	-	-	-	-	-	-	3	2
RCF	C ← 0			CF	0	-	-	-	-	-	1	2
RET	PC ← @SP SP ← SP + 2			AF	-	-	-	-	-	-	1	4
RL dst		R		90	*	*	*	*	-	-	2	2
		IR		91							2	3
RLC dst		R		10	*	*	*	*	-	-	2	2
		IR		11							2	3
RR dst		R		E0	*	*	*	*	-	-	2	2
		IR		E1							2	3
RRC dst		R		C0	*	*	*	*	-	-	2	2
		IR		C1							2	3
SBC dst, src	dst ← dst – src - C	r	r	32	*	*	*	*	1	*	2	3
		r	Ir	33							2	4
		R	R	34							3	3
		R	IR	35							3	4
		R	IM	36							3	3
		IR	IM	37							3	4
SBCX dst, src	dst ← dst – src - C	ER	ER	38	*	*	*	*	1	*	4	3
		ER	IM	39							4	3