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, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	46
Program Memory Size	48KB (48K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	4K x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 3.6V
Data Converters	A/D 12x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	68-LCC (J-Lead)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f4802vs020ec00tr

Use of All Uppercase Letters

The use of all uppercase letters designates the names of states and commands.

- Example 1: The bus is considered BUSY after the Start condition.
- Example 2: A START command triggers the processing of the initialization sequence.

Bit Numbering

Bits are numbered from *0* to *n–1* where *n* indicates the total number of bits. For example, the 8 bits of a register are numbered from 0 to 7.

Safeguards

It is important that all users understand the following safety terms, which are defined here.

Caution: Indicates a procedure or file may become corrupted if the user does not follow directions.

Trademarks

Block Diagram

Figure 55 illustrates the block diagram of the architecture of the Z8 Encore!™.

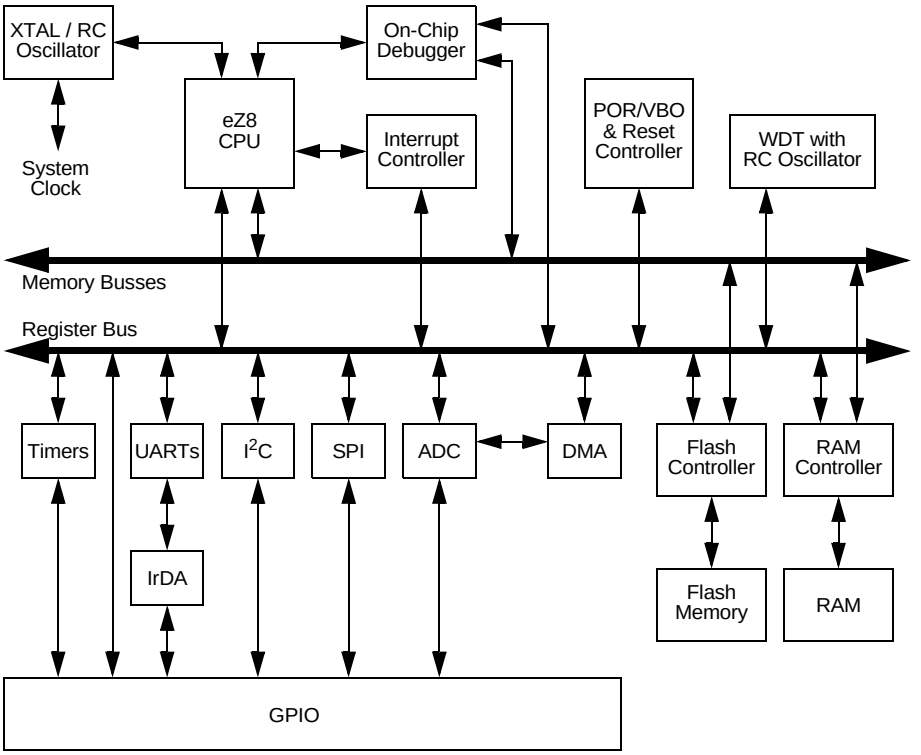


Figure 55. Z8 Encore!® Block Diagram

CPU and Peripheral Overview

eZ8 CPU Features

The eZ8, ZiLOG's latest 8-bit Central Processing Unit (CPU), meets the continuing demand for faster and more code-efficient microcontrollers. The eZ8 CPU executes a superset of the original Z8 instruction set. The eZ8 CPU features include:

- Direct register-to-register architecture allows each register to function as an accumulator, improving execution time and decreasing the required program memory

Power-On Reset

The Z8F640x family products contain an internal Power-On Reset (POR) circuit. The POR circuit monitors the supply voltage and holds the device in the Reset state until the supply voltage reaches a safe operating level. After the supply voltage exceeds the POR voltage threshold (V_{POR}), the POR Counter is enabled and counts 514 cycles of the Watch-Dog Timer oscillator. After the POR counter times out, the XTAL Counter is enabled to count a total of 16 system clock pulses. The Z8F640x family device is held in the Reset state until both the POR Counter and XTAL counter have timed out. After the device exits the Power-On Reset state, the eZ8 CPU fetches the Reset vector. Following Power-On Reset, the POR status bit in the Watch-Dog Timer Control (WDTCTL) register is set to 1.

Figure 62 illustrates Power-On Reset operation. Refer to the **Electrical Characteristics** chapter for the POR threshold voltage (V_{POR}).

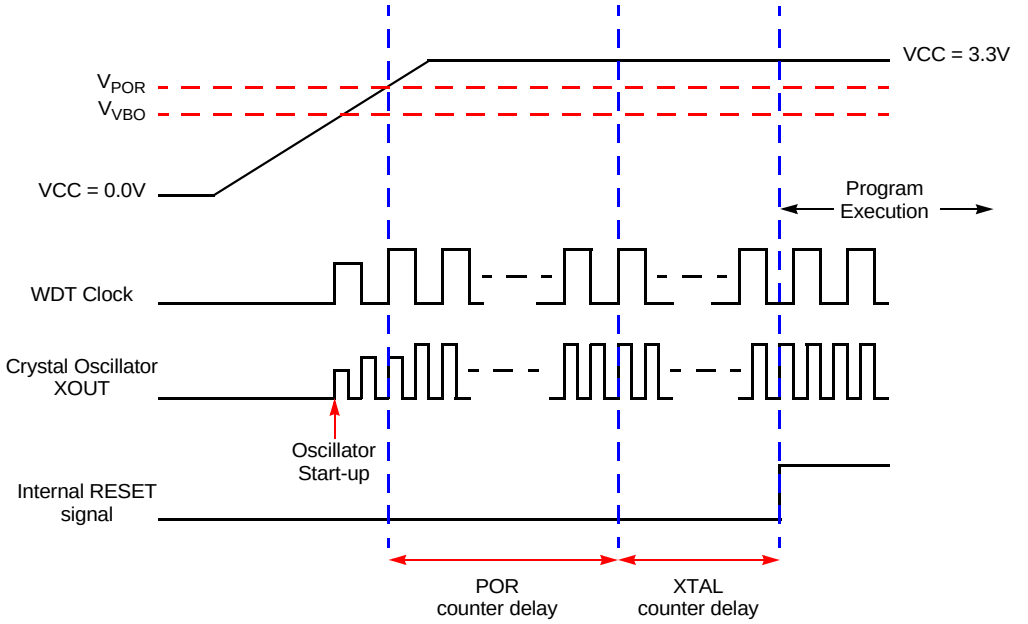


Figure 62. Power-On Reset Operation (not to scale)

Voltage Brown-Out Reset

The devices in the Z8F640x family provide low Voltage Brown-Out (VBO) protection. The VBO circuit senses when the supply voltage drops to an unsafe level (below the VBO

Low-Power Modes

Overview

The Z8F640x family products contain power-saving features. The highest level of power reduction is provided by Stop mode. The next level of power reduction is provided by the Halt mode.

Stop Mode

Execution of the eZ8 CPU's STOP instruction places the Z8F640x family device into Stop mode. In Stop mode, the operating characteristics are:

- Primary crystal oscillator is stopped
- System clock is stopped
- eZ8 CPU is stopped
- Program counter (PC) stops incrementing
- Watch-Dog Timer's internal RC oscillator continues to operate
- If enabled, the Watch-Dog Timer continues to operate
- All other on-chip peripherals are idle

To minimize current in Stop mode, all GPIO pins that are configured as digital inputs must be driven to one of the supply rails (V_{CC} or GND). The Z8F640x family device can be brought out of Stop mode using Stop Mode Recovery. For more information on STOP Mode Recovery refer to the **Reset and Stop Mode Recovery** chapter.

Halt Mode

Execution of the eZ8 CPU's HALT instruction places the Z8F640x family device into Halt mode. In Halt mode, the operating characteristics are:

- Primary crystal oscillator is enabled and continues to operate
- System clock is enabled and continues to operate
- eZ8 CPU is idle
- Program counter (PC) stops incrementing

Port A-H Data Direction Sub-Registers

The Port A-H Data Direction sub-register is accessed through the Port A-H Control register by writing 01H to the Port A-H Address register (Table 15).

Table 15. Port A-H Data Direction Sub-Registers

BITS	7	6	5	4	3	2	1	0
FIELD	DD7	DD6	DD5	DD4	DD3	DD2	DD1	DD0
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
ADDR	If 01H in Port A-H Address Register, accessible via Port A-H Control Register							

DD[7:0]—Data Direction
These bits control the direction of the associated port pin. Port Alternate Function operation overrides the Data Direction register setting.
0 = Output. Data in the Port A-H Output Data register is driven onto the port pin.
1 = Input. The port pin is sampled and the value written into the Port A-H Input Data Register. The output driver is tri-stated.

Port A-H Alternate Function Sub-Registers

The Port A-H Alternate Function sub-register (Table 16) is accessed through the Port A-H Control register by writing 02H to the Port A-H Address register. The Port A-H Alternate Function sub-registers select the alternate functions for the selected pins. Refer to the **GPIO Alternate Functions** section to determine the alternate function associated with each port pin.

Caution: Do not enable alternate function for GPIO port pins which do not have an associated alternate function. Failure to follow this guideline may result in unpredictable operation.

Table 16. Port A-H Alternate Function Sub-Registers

BITS	7	6	5	4	3	2	1	0
FIELD	AF7	AF6	AF5	AF4	AF3	AF2	AF1	AF0
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	If 02H in Port A-H Address Register, accessible via Port A-H Control Register							

out, first set the TPOL bit in the Timer Control Register to the start value before beginning One-Shot mode. Then, after starting the timer, set TPOL to the opposite bit value.

The steps for configuring a timer for One-Shot mode and initiating the count are as follows:

1. Write to the Timer Control register to:
   – Disable the timer
   – Configure the timer for One-Shot 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.
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:

$$\textbf{One-Shot Mode Time-Out Period (s)} = \frac{(\textbf{Reload Value} - \textbf{Start Value}) \times \textbf{Prescale}}{\textbf{System Clock Frequency (Hz)}}$$

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.

The steps for configuring a timer for Continuous mode and initiating the count are as follows:

1. Write to the Timer Control register to:
   – Disable the timer
   – Configure the timer for Continuous mode.
   – Set the prescale value.

Watch-Dog Timer

Overview

The Watch-Dog Timer (WDT) helps protect against corrupt or unreliable software, power faults, and other system-level problems which may place the Z8 Encore!® into unsuitable operating states. The Watch-Dog Timer includes the following features:

- On-chip RC oscillator
- A selectable time-out response: Short Reset or interrupt
- 24-bit programmable time-out value

Operation

The Watch-Dog Timer (WDT) is a retriggerable one-shot timer that resets or interrupts the Z8F640x family device when the WDT reaches its terminal count. The Watch-Dog Timer uses its own dedicated on-chip RC oscillator as its clock source. The Watch-Dog Timer has only two modes of operation—on and off. Once enabled, it always counts and must be refreshed to prevent a time-out. An enable can be performed by executing the WDT instruction or by setting the WDT_AO Option Bit. The WDT_AO bit enables the Watch-Dog Timer to operate all the time, even if a WDT instruction has not been executed.

The Watch-Dog Timer is a 24-bit reloadable downcounter that uses three 8-bit registers in the eZ8 CPU register space to set the reload value. The nominal WDT time-out period is given by the following equation:

$$\text{WDT Time-out Period (ms)} = \frac{\text{WDT Reload Value}}{50}$$

where the WDT reload value is the decimal value of the 24-bit value given by {WDTU[7:0], WDTH[7:0], WDTL[7:0]} and the typical Watch-Dog Timer RC oscillator frequency is 50kHz. The Watch-Dog Timer cannot be refreshed once it reaches 000002H. The WDT Reload Value must not be set to values below 000004H. Table 45 provides

Reserved
These bits are reserved and must be 0.

Watch-Dog Timer Reload Upper, High and Low Byte Registers

The Watch-Dog Timer Reload Upper, High and Low Byte (WDTU, WDTH, WDTL) registers (Tables 47 through 49) form the 24-bit reload value that is loaded into the Watch-Dog Timer when a WDT instruction executes. The 24-bit reload value is {WDTU[7:0], WDTH[7:0], WDTL[7:0]. Writing to these registers sets the desired Reload Value. Reading from these registers returns the current Watch-Dog Timer count value.

Caution: The 24-bit WDT Reload Value must not be set to a value less than 000004H or unpredictable behavior may result.

Table 47. Watch-Dog Timer Reload Upper Byte Register (WDTU)

BITS	7	6	5	4	3	2	1	0
FIELD	WDTU							
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*
ADDR	FF1H							

R/W* - Read returns the current WDT count value. Write sets the desired Reload Value.

WDTU—WDT Reload Upper Byte

Most significant byte (MSB), Bits[23:16], of the 24-bit WDT reload value.

Table 48. Watch-Dog Timer Reload High Byte Register (WDTH)

BITS	7	6	5	4	3	2	1	0
FIELD	WDTH							
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*
ADDR	FF2H							

R/W* - Read returns the current WDT count value. Write sets the desired Reload Value.

WDTH—WDT Reload High Byte

Table 58. UART Baud Rates

20.0 MHz System Clock

Desired Rate (kHz)	BRG Divisor (Decimal)	Actual Rate (kHz)	Error (%)
1250.0	1	1250.0	0.00
625.0	2	625.0	0.00
250.0	5	250.0	0.00
115.2	11	113.6	-1.36
57.6	22	56.8	-1.36
38.4	33	37.9	-1.36
19.2	65	19.2	0.16
9.60	130	9.62	0.16
4.80	260	4.81	0.16
2.40	521	2.40	-0.03
1.20	1042	1.20	-0.03
0.60	2083	0.60	0.02
0.30	4167	0.30	-0.01

18.432 MHz System Clock

Desired Rate (kHz)	BRG Divisor (Decimal)	Actual Rate (kHz)	Error (%)
1250.0	1	1152.0	-7.84%
625.0	2	576.0	-7.84%
250.0	5	230.4	-7.84%
115.2	10	115.2	0.00
57.6	20	57.6	0.00
38.4	30	38.4	0.00
19.2	60	19.2	0.00
9.60	120	9.60	0.00
4.80	240	4.80	0.00
2.40	480	2.40	0.00
1.20	960	1.20	0.00
0.60	1920	0.60	0.00
0.30	3840	0.30	0.00

16.667 MHz System Clock

Desired Rate (kHz)	BRG Divisor (Decimal)	Actual Rate (kHz)	Error (%)
1250.0	1	1041.69	-16.67
625.0	2	520.8	-16.67
250.0	4	260.4	4.17
115.2	9	115.7	0.47
57.6	18	57.87	0.47
38.4	27	38.6	0.47
19.2	54	19.3	0.47
9.60	109	9.56	-0.45
4.80	217	4.80	-0.83
2.40	434	2.40	0.01
1.20	868	1.20	0.01
0.60	1736	0.60	0.01
0.30	3472	0.30	0.01

11.0592 MHz System Clock

Desired Rate (kHz)	BRG Divisor (Decimal)	Actual Rate (kHz)	Error (%)
1250.0	N/A	N/A	N/A
625.0	1	691.2	10.59
250.0	3	230.4	-7.84
115.2	6	115.2	0.00
57.6	12	57.6	0.00
38.4	18	38.4	0.00
19.2	36	19.2	0.00
9.60	72	9.60	0.00
4.80	144	4.80	0.00
2.40	288	2.40	0.00
1.20	576	1.20	0.00
0.60	1152	0.60	0.00
0.30	2304	0.30	0.00

Operation

When the Infrared Endec is enabled, the transmit data from the associated on-chip UART is encoded as digital signals in accordance with the IrDA standard and output to the infrared transceiver via the TXD pin. Likewise, data received from the infrared transceiver is passed to the Infrared Endec via the RXD pin, decoded by the Infrared Endec, and then passed to the UART. Communication is half-duplex, which means simultaneous data transmission and reception is not allowed.

The baud rate is set by the UART's Baud Rate Generator and supports IrDA standard baud rates from 9600 baud to 115.2 kbaud. Higher baud rates are possible, but do not meet IrDA specifications. The UART must be enabled to use the Infrared Endec. The Infrared Endec data rate is calculated using the following equation:

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

Transmitting IrDA Data

The data to be transmitted using the infrared transceiver is first sent to the UART. The UART's transmit signal (TXD) and baud rate clock are used by the IrDA to generate the modulation signal (IR_TXD) that drives the infrared transceiver. Each UART/Infrared data bit is 16-clocks wide. If the data to be transmitted is 1, the IR_TXD signal remains low for the full 16-clock period. If the data to be transmitted is 0, a 3-clock high pulse is output following a 7-clock low period. After the 3-clock high pulse, a 6-clock low pulse is output to complete the full 16-clock data period. Figure 72 illustrates IrDA data transmission. When the Infrared Endec is enabled, the UART's TXD signal is internal to the Z8F640x family device while the IR_TXD signal is output through the TXD pin.

Error Detection

The SPI contains error detection logic to support SPI communication protocols and recognize when communication errors have occurred. The SPI Status register indicates when a data transmission error has been detected.

Overrun (Write Collision)

An overrun error (write collision) indicates a write to the SPI Data register was attempted while a data transfer is in progress. An overrun sets the OVR bit in the SPI Status register to 1. Writing a 1 to OVR clears this error flag.

Mode Fault (Multi-Master Collision)

A mode fault indicates when more than one Master is trying to communicate at the same time (a multi-master collision). The mode fault is detected when the enabled Master's $\overline{SS}$ pin is asserted. A mode fault sets the COL bit in the SPI Status register to 1. Writing a 1 to COL clears this error flag.

SPI Interrupts

When SPI interrupts are enabled, the SPI generates an interrupt after data transmission. The SPI in Master mode generates an interrupt after a character has been sent. A character can be defined to be 1 through 8 bits by the NUMBITS field in the SPI Mode register. The SPI in Slave mode generates an interrupt when the $\overline{SS}$ signal deasserts to indicate completion of the data transfer. Writing a 1 to the IRQ bit in the SPI Status Register clears the pending interrupt request. If the SPI is disabled, an SPI interrupt can be generated by a Baud Rate Generator time-out.

SPI Baud Rate Generator

In SPI Master mode, the Baud Rate Generator creates a lower frequency serial clock (SCK) for data transmission synchronization between the Master and the external Slave. The input to the Baud Rate Generator is the system clock. The SPI Baud Rate High and Low Byte registers combine to form a 16-bit reload value, BRG[15:0], for the SPI Baud Rate Generator. The reload value must be greater than or equal to 0002H for SPI operation (maximum baud rate is system clock frequency divided by 4). The SPI baud rate is calculated using the following equation:

$$\text{SPI Baud Rate (bits/s)} = \frac{\text{System Clock Frequency (Hz)}}{2 \times \text{BRG[15:0]}}$$

When the SPI is disabled, the Baud Rate Generator can function as a basic 16-bit timer with interrupt on time-out. To configure the Baud Rate Generator as a timer with interrupt on time-out, complete the following procedure:

I²C Control Register Definitions

I²C Data Register

The I²C Data register holds the data that is to be loaded into the I²C Shift register during a write to a slave. This register also holds data that is loaded from the I²C Shift register during a read from a slave. The I²C Shift is not accessible in the Register File address space, but is used only to buffer incoming and outgoing data.

Table 66. I²C Data Register (I2CDATA)

BITS	7	6	5	4	3	2	1	0
FIELD	DATA							
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	F50H							

I²C Status Register

The Read-only I²C Status register indicates the status of the I²C Controller.

Table 67. I²C Status Register (I2CSTAT)

BITS	7	6	5	4	3	2	1	0
FIELD	TDRE	RDRF	ACK	10B	RD	TAS	DSS	NCKI
RESET	1	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R
ADDR	F51H							

TDRE—Transmit Data Register Empty
When the I²C Controller is enabled, this bit is 1 when the I²C Data register is empty. When active, this bit causes the I²C Controller to generate an interrupt, except when the I²C Controller is shifting in data during the reception of a byte or when shifting an address and the RD bit is set. This bit and the interrupt are cleared by writing to the I²CD register.

RDRF—Receive Data Register Full
This bit is set active high when the I²C Controller is enabled and the I²C Controller has

I²C Baud Rate High and Low Byte Registers

The I²C Baud Rate High and Low Byte registers combine to form a 16-bit reload value, BRG[15:0], for the I²C Baud Rate Generator. The I²C baud rate is calculated using the following equation:

$$\text{I2C Baud Rate (bits/s)} = \frac{\text{System Clock Frequency (Hz)}}{4 \times \text{BRG[15:0]}}$$

.

Table 69. I²C Baud Rate High Byte Register (I2CBRH)

BITS	7	6	5	4	3	2	1	0
FIELD	BRH							
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
ADDR	F53H							

BRH = I²C Baud Rate High Byte
Most significant byte, BRG[15:8], of the I²C Baud Rate Generator's reload value.

Table 70. I²C Baud Rate Low Byte Register (I2CBRL)

BITS	7	6	5	4	3	2	1	0
FIELD	BRL							
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
ADDR	F54H							

BRL = I²C Baud Rate Low Byte
Least significant byte, BRG[7:0], of the I²C Baud Rate Generator's reload value.

Analog-to-Digital Converter

Overview

The Analog-to-Digital Converter (ADC) converts an analog input signal to a 10-bit binary number. The features of the sigma-delta ADC include:

- 12 analog input sources are multiplexed with general-purpose I/O ports
- Interrupt upon conversion complete
- Internal voltage reference generator
- Direct Memory Access (DMA) controller can automatically initiate data conversion and transfer of the data from 1 to 12 of the analog inputs.

Architecture

Figure 83 illustrates the three major functional blocks (converter, analog multiplexer, and voltage reference generator) of the ADC. The ADC converts an analog input signal to its digital representation. The 12-input analog multiplexer selects one of the 12 analog input sources. The ADC requires an input reference voltage for the conversion. The voltage reference for the conversion may be input through the external VREF pin or generated internally by the voltage reference generator.

Caution: The byte at each address of the Flash memory cannot be programmed (any bits written to 0) more than twice before an erase cycle occurs.

Page Erase

The Flash memory can be erased one page (512 bytes) at a time. Page Erasing the Flash memory sets all bytes in that page to the value FFH. The Flash Page Select register identifies the page to be erased. With the Flash Controller unlocked, writing the value 95H to the Flash Control register initiates the Page Erase operation. While the Flash Controller executes the Page Erase operation, the eZ8 CPU idles but the system clock and on-chip peripherals continue to operate. The eZ8 CPU resumes operation after the Page Erase operation completes. If the Page Erase operation is performed through the On-Chip Debugger, poll the Flash Status register to determine when the Page Erase operation is complete. When the Page Erase is complete, the Flash Controller returns to its locked state.

Mass Erase

The Flash memory can also be Mass Erased using the Flash Controller. Mass Erasing the Flash memory sets all bytes to the value FFH. With the Flash Controller unlocked, writing the 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. Typically, the Flash Memory is Mass Erased using the On-Chip Debugger. Via the On-Chip Debugger, poll the Flash Status register to determine when the Mass Erase operation is complete. Although the Flash can be Mass Erased by user program code, when the Mass Erase is complete the user program code is completely erased. 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 programing is recommended for gang programming applications and large volume customers who do not require in-circuit initial programming of the Flash memory. Mass Erase and Page Erase operations are also supported when the Flash Controller is bypassed.

Please refer to the document entitled *Third-Party Flash Programming Support for Z8 Encore!*™ for more information on bypassing the Flash Controller. This document is available for download at www.zilog.com.

Figure 92 illustrates the typical current consumption in Halt mode while operating at 25ºC, 3.3V, versus the system clock frequency.

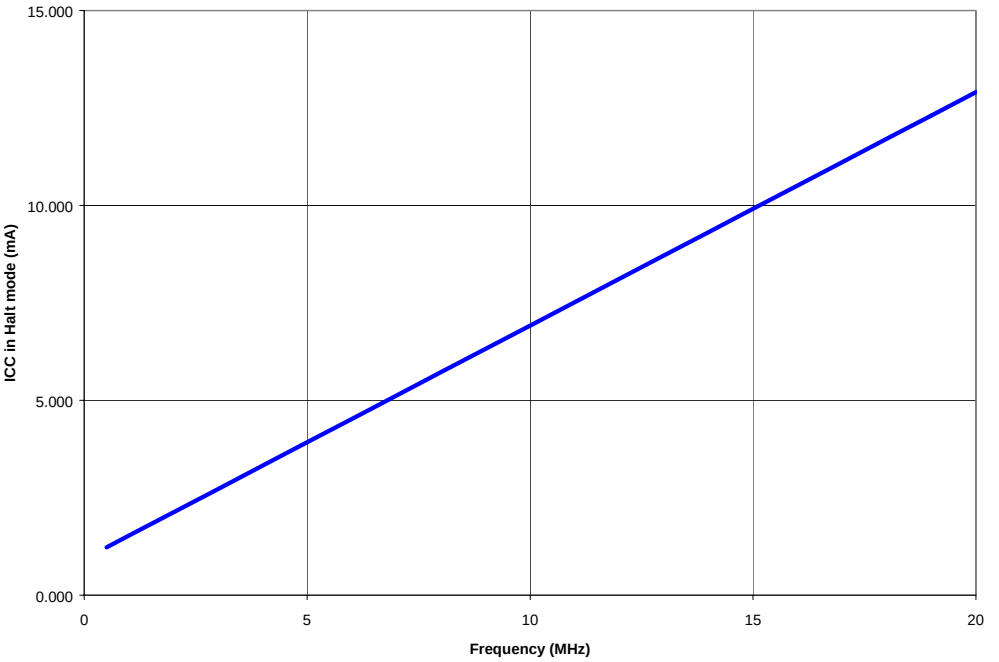


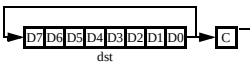
Figure 92. Nominal Halt Mode ICC Versus System Clock Frequency

Table 115. Notational Shorthand

Notation	Description	Operand	Range
b	Bit	b	b represents a value from 0 to 7 (000B to 111B).
cc	Condition Code	—	See Condition Codes overview in the eZ8 CPU User Manual.
DA	Direct Address	Addrs	Addrs. represents a number in the range of 0000H to FFFFH
ER	Extended Addressing Register	Reg	Reg. represents a number in the range of 000H to FFFH
IM	Immediate Data	#Data	Data is a number between 00H to FFH
Ir	Indirect Working Register	@Rn	n = 0 –15
IR	Indirect Register	@Reg	Reg. represents a number in the range of 00H to FFH
Irr	Indirect Working Register Pair	@RRp	p = 0, 2, 4, 6, 8, 10, 12, or 14
IRR	Indirect Register Pair	@Reg	Reg. represents an even number in the range 00H to FEH
p	Polarity	p	Polarity is a single bit binary value of either 0B or 1B.
r	Working Register	Rn	n = 0 – 15
R	Register	Reg	Reg. represents a number in the range of 00H to FFH
RA	Relative Address	X	X represents an index in the range of +127 to –128 which is an offset relative to the address of the next instruction
rr	Working Register Pair	RRp	p = 0, 2, 4, 6, 8, 10, 12, or 14
RR	Register Pair	Reg	Reg. represents an even number in the range of 00H to FEH
Vector	Vector Address	Vector	Vector represents a number in the range of 00H to FFH
X	Indexed	#Index	The register or register pair to be indexed is offset by the signed Index value (#Index) in a +127 to -128 range.

Table 116 contains additional symbols that are used throughout the Instruction Summary and Instruction Set Description sections.

Table 126. 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
POP dst	dst ← @SP SP ← SP + 1	R		50	-	-	-	-	-	-	2	2
		IR		51							2	3
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

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

Opcode Maps

Figures 101 and 102 provide information on each of the eZ8 CPU instructions. A description of the opcode map data and the abbreviations are provided in Figure 100 and Table 127.

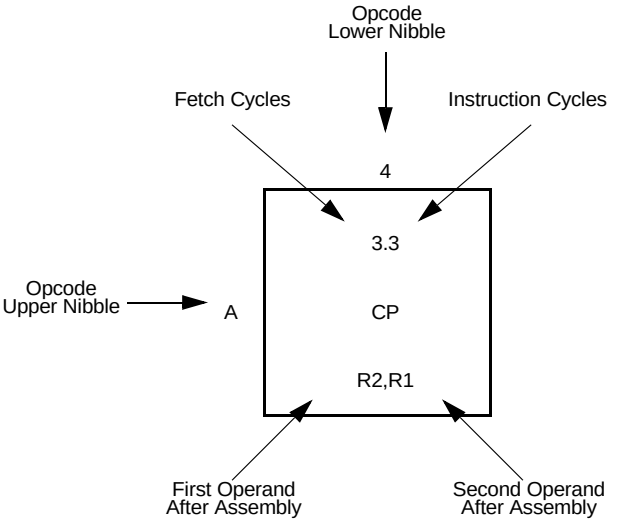


Figure 100. Opcode Map Cell Description