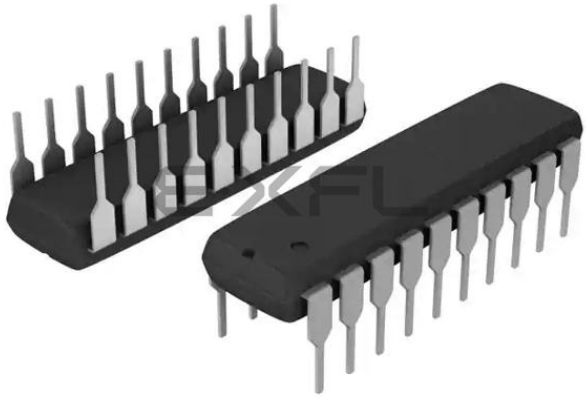

Welcome to E-XFL.COM

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	-40°C ~ 105°C (TA)
Mounting Type	Through Hole
Package / Case	20-DIP (0.300", 7.62mm)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f0811ph020ec

Warning: DO NOT USE IN LIFE SUPPORT

LIFE SUPPORT POLICY

ZILOG'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS PRIOR WRITTEN APPROVAL OF THE PRESIDENT AND GENERAL COUNSEL OF ZILOG CORPORATION.

As used herein

Life support devices or systems are devices which (a) are intended for surgical implant into the body, or (b) support or sustain life and whose failure to perform when properly used in accordance with instructions for use provided in the labeling can be reasonably expected to result in a significant injury to the user. A critical component is any component in a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system or to affect its safety or effectiveness.

Document Disclaimer

 Information in this publication concerning the devices, applications, or technology described is intended to suggest possible uses and may be superseded. ZILOG, INC. DOES NOT ASSUME LIABILITY FOR OR PROVIDE A REPRESENTATION OF ACCURACY OF THE INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED IN THIS DOCUMENT. ZILOG ALSO DOES NOT ASSUME LIABILITY FOR INTELLECTUAL PROPERTY INFRINGEMENT RELATED IN ANY MANNER TO USE OF INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED HEREIN OR OTHERWISE. The information contained within this document has been verified according to the general principles of electrical and mechanical engineering.

Revision History

Each instance in Revision History reflects a change to this document from its previous revision. For more details, refer to the corresponding pages and appropriate links in the table below.

Date	Revision Level	Description	Page Number
May 2008	17	Removed Flash Microcontrollers from the title throughout the document.	All
February 2008	16	Updated the flag status for BCLR, BIT, and BSET in Table 126.	219
December 2007	15	Updated Zilog logo, Zilog text, Disclaimer section, and implemented style guide. Updated Z8 Encore! 8K Series to Z8 Encore! XP F0822 Series Flash Microcontrollers throughout the document.	All
June 2007 and August 2007	13 and 14	No Changes.	All
December 2006	12	Updated Ordering Information and minor edits done.	All

Table of Contents

Table 3. Signal Descriptions (Continued)

Signal Mnemonic	I/O	Description
SPI Controller		
$\overline{SS}$	I/O	**Slave Select**—This signal can be an output or an input. If the Z8 Encore! XP® is the SPI Master, this pin can be configured as the Slave Select output. If the Z8 Encore! XP is the SPI Slave, this pin is the input slave select. It is multiplexed with a GPIO pin.
SCK	I/O	**SPI Serial Clock**—The SPI Master supplies this pin. If the Z8 Encore! XP is the SPI Master, this pin is the output. If the Z8 Encore! XP is the SPI Slave, this pin is the input. It is multiplexed with a GPIO pin.
MOSI	I/O	**Master-Out/Slave-In**—This signal is the data output from the SPI Master device and the data input to the SPI Slave device. It is multiplexed with a GPIO pin.
MISO	I/O	**Master-In/Slave-Out**—This pin is the data input to the SPI Master device and the data output from the SPI Slave device. It is multiplexed with a GPIO pin.
UART Controllers		
TXD0	O	**Transmit Data**—This signal is the transmit output from the UART and IrDA. The TXD signals are multiplexed with GPIO pins.
RXD0	I	**Receive Data**—This signal is the receiver input for the UART and IrDA. The RXD signals are multiplexed with GPIO pins.
$\overline{CTS0}$	I	**Clear To Send**—This signal is control inputs for the UART. The $\overline{CTS}$ signals are multiplexed with GPIO pins.
DE0	O	**Driver Enable**—This signal allows automatic control of external RS-485 drivers. This signal is approximately the inverse of the TXE (Transmit Empty) bit in the UART Status 0 Register. The DE signal can be used to ensure the external RS-485 driver is enabled when data is transmitted by the UART.
Timers		
T0OUT / T1OUT	O	**Timer Output 0–1**—These signals are output pins from the timers. The Timer Output signals are multiplexed with GPIO pins.
T0IN / T1IN	I	**Timer Input 0–1**—These signals are used as the Capture, Gating and Counter inputs. The Timer Input signals are multiplexed with GPIO pins.
Analog		
ANA[4:0]	I	**Analog Input**—These signals are inputs to the Analog-to-Digital Converter (ADC). The ADC analog inputs are multiplexed with GPIO pins.
VREF	I	**Analog-to-Digital Converter reference voltage input**—As an output, the VREF signal is not recommended for use as a reference voltage for external devices. If the ADC is configured to use the internal reference voltage generator, this pin should be left unconnected or capacitively coupled to analog ground (AVSS).

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

Port A–C Output Data Register

The Port A–C Output Data Register (Table 23) controls the output data to the pins.

Table 23. Port A–C Output Data Register (P*x*OUT)

BITS	7	6	5	4	3	2	1	0
FIELD	POUT7	POUT6	POUT5	POUT4	POUT3	POUT2	POUT1	POUT0
RESET	0							
R/W	R/W							
ADDR	FD3H, FD7H, FDBH							

POUT[7:0]—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.

Table 29. IRQ0 Enable High Bit Register (IRQ0ENH)

BITS	7	6	5	4	3	2	1	0
FIELD	Reserved	T1ENH	T0ENH	U0RENH	U0TENH	I2CENH	SPIENH	ADCENH
RESET	0							
R/W	R/W							
ADDR	FC1H							

Reserved—Must be 0
T1ENH—Timer 1 Interrupt Request Enable High Bit
T0ENH—Timer 0 Interrupt Request Enable High Bit
U0RENH—UART 0 Receive Interrupt Request Enable High Bit
U0TENH—UART 0 Transmit Interrupt Request Enable High Bit
I2CENH—I^2C Interrupt Request Enable High Bit
SPIENH—SPI Interrupt Request Enable High Bit
ADCENH—ADC Interrupt Request Enable High Bit

Table 30. IRQ0 Enable Low Bit Register (IRQ0ENL)

BITS	7	6	5	4	3	2	1	0
FIELD	Reserved	T1ENL	T0ENL	U0RENL	U0TENL	I2CENL	SPIENL	ADCENL
RESET	0							
R/W	R/W							
ADDR	FC2H							

Reserved—Must be 0
T1ENL—Timer 1 Interrupt Request Enable Low Bit
T0ENL—Timer 0 Interrupt Request Enable Low Bit
U0RENL—UART 0 Receive Interrupt Request Enable Low Bit
U0TENL—UART 0 Transmit Interrupt Request Enable Low Bit
I2CENL—I^2C Interrupt Request Enable Low Bit
SPIENL—SPI Interrupt Request Enable Low Bit
ADCENL—ADC Interrupt Request Enable Low Bit

IRQ1 Enable High and Low Bit Registers

Table 31 describes the priority control for IRQ1. The IRQ1 Enable High and Low Bit Registers (Table 32 and Table 33) form a priority encoded enabling for interrupts in the Interrupt Request 1 Register. Priority is generated by setting bits in each register.

Timers

Z8 Encore! XP® F0822 Series products contain up to two 16-bit reloadable timers that can be used for timing, event counting, or generation of pulse-width modulated signals. The timer features include:

- 16-bit reload counter.
- Programmable prescaler with prescale values from 1 to 128.
- PWM output generation.
- Capture and compare capability.
- External input pin for timer input, clock gating, or capture signal. External input pin signal frequency is limited to a maximum of one-fourth the system clock frequency.
- Timer output pin.
- Timer interrupt.

In addition to the timers described in this chapter, the Baud Rate Generators for any unused UART, SPI, or I^2C peripherals can also be used to provide basic timing functionality. See the respective serial communication peripheral chapters for information on using the Baud Rate Generators as timers.

Architecture

Figure 10 displays the architecture of the timers.

Operation

The timers are 16-bit up-counters. Minimum time-out delay is set by loading the value 0001H into the Timer Reload High and Low Byte registers and setting the prescale value to 1. Maximum time-out delay is set by loading the value 0000H into the Timer Reload High and Low Byte registers and setting the prescale value to 128. If the Timer reaches FFFFH, the timer rolls over to 0000H and continues counting.

COMPARE Mode
When the timer is disabled, the Timer Output signal is set to the value of this bit. When the timer is enabled, the Timer Output signal is complemented upon timer Reload.

GATED Mode
0 = Timer counts when the Timer Input signal is High (1) and interrupts are generated on the falling edge of the Timer Input.
1 = Timer counts when the Timer Input signal is Low (0) and interrupts are generated on the rising edge of the Timer Input.

CAPTURE/COMPARE Mode
0 = Counting is started on the first rising edge of the Timer Input signal. The current count is captured on subsequent rising edges of the Timer Input signal.
1 = Counting is started on the first falling edge of the Timer Input signal. The current count is captured on subsequent falling edges of the Timer Input signal.

PRES—Prescale value
The timer input clock is divided by 2^{PRES}, where PRES is set from 0 to 7. The prescaler is reset each time the Timer is disabled. This insures proper clock division each time the Timer is restarted.
000 = Divide by 1
001 = Divide by 2
010 = Divide by 4
011 = Divide by 8
100 = Divide by 16
101 = Divide by 32
110 = Divide by 64
111 = Divide by 128

TMODE—Timer Mode
000 = ONE-SHOT mode
001 = CONTINUOUS mode
010 = COUNTER mode
011 = PWM mode
100 = CAPTURE mode
101 = COMPARE mode
110 = GATED mode
111 = CAPTURE/COMPARE mode

Receiving Data Using Interrupt-Driven Method

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

1. Write to the UART Baud Rate High and Low Byte Registers to set the required 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 required 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 desired.
   – Set the Multiprocessor Mode Select (MPEN) to enable MULTIPROCESSOR mode.
   – Set the Multiprocessor Mode Bits, MPMD[1:0], to select the required 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 required, 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 ISR performs the following:

1. Check the UART Status 0 Register to determine the source of the interrupt-error, break, or received data.
2. If the interrupt was due to data available, read the data from the UART Receive Data Register. If operating in MULTIPROCESSOR (9-bit) mode, further actions may be required depending on the Multiprocessor Mode bits MPMD[1:0].
3. Clear the UART Receiver Interrupt in the applicable Interrupt Request Register.
4. Execute the IRET instruction to return from the ISR and await more data.

Reserved—Must be 0

NEWFRM—Status bit denoting the start of a new frame. Reading the UART Receive Data Register resets this bit to 0.
0 = The current byte is not the first data byte of a new frame.
1 = The current byte is the first data byte of a new frame.

MPRX—Multiprocessor Receive
Returns the value of the last multiprocessor bit received. Reading from the UART Receive Data Register resets this bit to 0.

UART Control 0 and Control 1 Registers

The UART Control 0 and Control 1 registers (Table 56 and Table 57 on page 104) configure the properties of the UART's transmit and receive operations. The UART Control Registers must not been written while the UART is enabled.

Table 56. UART Control 0 Register (U0CTL0)

BITS	7	6	5	4	3	2	1	0
FIELD	TEN	REN	CTSE	PEN	PSEL	SBRK	STOP	LBEN
RESET	0							
R/W	R/W							
ADDR	F42H							

TEN—Transmit Enable
This bit enables or disables the transmitter. The enable is also controlled by the $\overline{\text{CTS}}$ signal and the CTSE bit. If the $\overline{\text{CTS}}$ signal is low and the CTSE bit is 1, the transmitter is enabled.
0 = Transmitter disabled.
1 = Transmitter enabled.

REN—Receive Enable
This bit enables or disables the receiver.
0 = Receiver disabled.
1 = Receiver enabled.

CTSE—CTS Enable
0 = The $\overline{\text{CTS}}$ signal has no effect on the transmitter.
1 = The UART recognizes the $\overline{\text{CTS}}$ signal as an enable control from the transmitter.

PEN—Parity Enable
This bit enables or disables parity. Even or odd is determined by the PSEL bit. This bit is overridden by the MPEN bit.
0 = Parity is disabled.
1 = The transmitter sends data with an additional parity bit and the receiver receives an additional parity bit.

SPI Control Register

The SPI Control Register configures the SPI for transmit and receive operations.

Table 64. SPI Control Register (SPICTL)

BITS	7	6	5	4	3	2	1	0
FIELD	IRQE	STR	BIRQ	PHASE	CLKPOL	WOR	MMEN	SPIEN
RESET	0							
R/W	R/W							
ADDR	F61H							

IRQE—Interrupt Request Enable
0 = SPI interrupts are disabled. No interrupt requests are sent to the Interrupt Controller.
1 = SPI interrupts are enabled. Interrupt requests are sent to the Interrupt Controller.

STR—Start an SPI Interrupt Request
0 = No effect.
1 = Setting this bit to 1 also sets the IRQ bit in the SPI Status Register to 1. Setting this bit forces the SPI to send an interrupt request to the Interrupt Control. This bit can be used by software for a function similar to transmit buffer empty in a UART. Writing a 1 to the IRQ bit in the SPI Status Register clears this bit to 0.

BIRQ—BRG Timer Interrupt Request
If the SPI is enabled, this bit has no effect. If the SPI is disabled:
0 = BRG timer function is disabled.
1 = BRG timer function and time-out interrupt are enabled.

PHASE—Phase Select
Sets the phase relationship of the data to the clock. For more information on operation of the PHASE bit, see SPI Clock Phase and Polarity Control on page 116.

CLKPOL—Clock Polarity
0 = SCK idles Low (0).
1 = SCK idle High (1).

WOR—Wire-OR (Open-Drain) Mode Enabled
0 = SPI signal pins not configured for open-drain.
1 = All four SPI signal pins (SCK, $\overline{SS}$, MISO, MOSI) configured for open-drain function. This setting is typically used for multi-master and/or multi-slave configurations.

MMEN—SPI MASTER Mode Enable
0 = SPI configured in SLAVE mode.
1 = SPI configured in MASTER mode.

SPIEN—SPI Enable
0 = SPI disabled.
1 = SPI enabled.

Follow the steps below for a transmit operation on a 10-bit addressed slave:

1. Software asserts the IEN bit in the I2C Control Register.
2. Software asserts the TXI bit of the I2C Control Register to enable Transmit interrupts.
3. The I2C interrupt asserts because the I2C Data Register is empty.
4. Software responds to the TDRE interrupt by writing the first slave address byte to the I2C Data Register. The least-significant bit must be 0 for the write operation.
5. Software asserts the START bit of the I2C Control Register.
6. The I2C Controller sends the START condition to the I2C Slave.
7. The I2C Controller loads the I2C Shift register with the contents of the I2C Data Register.
8. After one bit of address is shifted out by the SDA signal, the Transmit Interrupt is asserted.
9. Software responds by writing the second byte of address into the contents of the I2C Data Register.
10. The I2C Controller shifts the rest of the first byte of address and write bit out the SDA signal.
11. If the I2C Slave acknowledges the first address byte by pulling the SDA signal low during the next high period of SCL, the I2C Controller sets the ACK bit in the I2C Status register. Continue with step 12.

    If the slave does not acknowledge the first address byte, the I2C Controller sets the NCKI bit and clears the ACK bit in the I2C 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).
12. The I2C Controller loads the I2C Shift register with the contents of the I2C Data Register.
13. The I2C Controller shifts the second address byte out the SDA signal. After the first bit has been sent, the Transmit Interrupt is asserted.
14. Software responds by writing a data byte to the I2C Data Register.
15. The I2C Controller completes shifting the contents of the shift register on the SDA signal.
16. If the I2C Slave sends an acknowledge by pulling the SDA signal low during the next high period of SCL, the I2C Controller sets the ACK bit in the I2C Status register. Continue with step 17.

    If the slave does not acknowledge the second address byte or one of the data bytes, the

Table 75. I²C Diagnostic State Register (I2CDST)

BITS	7	6	5	4	3	2	1	0
FIELD	SCLIN	SDAIN	STPCNT	TXRXSTATE				
RESET	X		0					
R/W	R							
ADDR	F55H							

SCLIN—Value of Serial Clock input signal
SDAIN—Value of the Serial Data input signal
STPCNT—Value of the internal Stop Count control signal
TXRXSTATE—Value of the internal I²C state machine

TXRXSTATE	State Description
0_0000	Idle State
0_0001	START State
0_0010	Send/Receive data bit 7
0_0011	Send/Receive data bit 6
0_0100	Send/Receive data bit 5
0_0101	Send/Receive data bit 4
0_0110	Send/Receive data bit 3
0_0111	Send/Receive data bit 2
0_1000	Send/Receive data bit 1
0_1001	Send/Receive data bit 0
0_1010	Data Acknowledge State
0_1011	Second half of data Acknowledge State used only for not acknowledge
0_1100	First part of STOP state
0_1101	Second part of STOP state
0_1110	10-bit addressing: Acknowledge State for 2nd address byte 7-bit addressing: Address Acknowledge State
0_1111	10-bit address: Bit 0 (Least significant bit) of 2nd address byte 7-bit address: Bit 0 (Least significant bit) (R/W) of address byte
1_0000	10-bit addressing: Bit 7 (Most significant bit) of 1st address byte
1_0001	10-bit addressing: Bit 6 of 1st address byte
1_0010	10-bit addressing: Bit 5 of 1st address byte
1_0011	10-bit addressing: Bit 4 of 1st address byte
1_0100	10-bit addressing: Bit 3 of 1st address byte
1_0101	10-bit addressing: Bit 2 of 1st address byte
1_0110	10-bit addressing: Bit 1 of 1st address byte

Operation

Automatic Power-Down

If the ADC is idle (no conversions in progress) for 160 consecutive system clock cycles, portions of the ADC are automatically powered-down. From this power-down state, the ADC requires 40 system clock cycles to power-up. The ADC powers up when a conversion is requested using the ADC Control Register.

Single-Shot Conversion

When configured for single-shot conversion, the ADC performs a single analog-to-digital conversion on the selected analog input channel. After completion of the conversion, the ADC shuts down. Follow the steps below for setting up the ADC and initiating a single-shot conversion:

1. Enable the desired analog inputs by configuring the GPIO pins for alternate function. This configuration disables the digital input and output drivers.
2. Write to the ADC Control Register to configure the ADC and begin the conversion. The bit fields in the ADC Control Register is written simultaneously:
   – Write to the ANAIN[3:0] field to select one of the 5 analog input sources.
   – Clear CONT to 0 to select a single-shot conversion.
   – Write to the VREF bit to enable or disable the internal voltage reference generator.
   – Set CEN to 1 to start the conversion.
3. CEN remains 1 while the conversion is in progress. A single-shot conversion requires 5129 system clock cycles to complete. If a single-shot conversion is requested from an ADC powered-down state, the ADC uses 40 additional clock cycles to power-up before beginning the 5129 cycle conversion.
4. When the conversion is complete, the ADC control logic performs the following operations:
   – 10-bit data result written to {ADCD_H[7:0], ADCD_L[7:6]}.
   – CEN resets to 0 to indicate the conversion is complete.
   – An interrupt request is sent to the Interrupt Controller.
5. If the ADC remains idle for 160 consecutive system clock cycles, it is automatically powered-down.

Continuous Conversion

When configured for continuous conversion, the ADC continuously performs an analog-to-digital conversion on the selected analog input. Each new data value over-writes the previous value stored in the ADC Data Registers. An interrupt is generated after each conversion.

Figure 43 displays the typical current consumption in HALT mode while operating at 25 ºC versus the system clock frequency. All GPIO pins are configured as outputs and driven High.

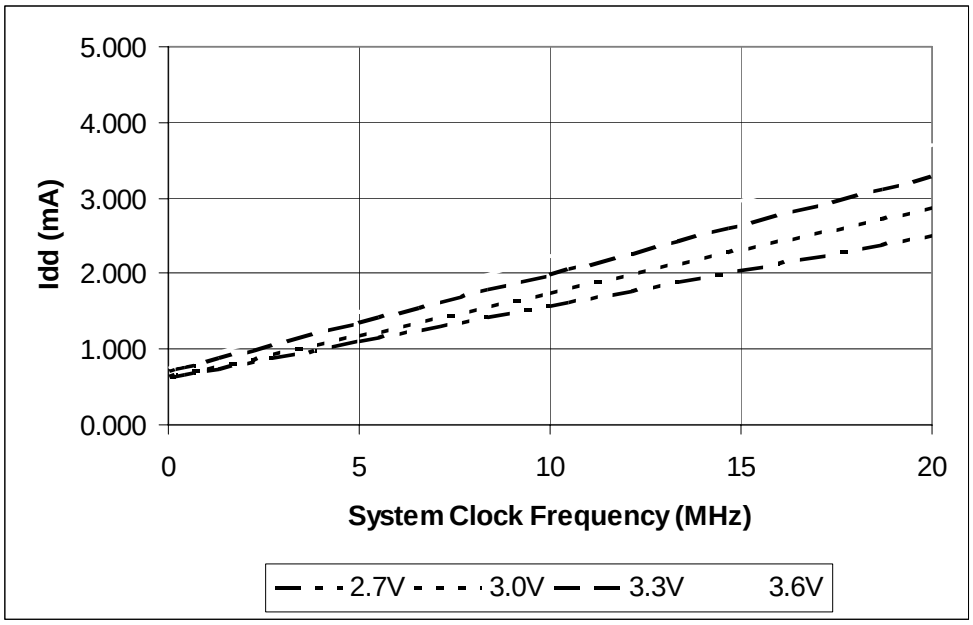

Figure 43. Typical HALT Mode I_{DD} Versus System Clock Frequency

Table 101. Flash Memory Electrical Characteristics and Timing (Continued)

Parameter	V_{DD} = 2.7 - 3.6V T_A = -40 °C to 105 °C Minimum	Typical	Maximum	Units	Notes
Data Retention	100	–	–	years	25 °C
Endurance	10,000	–	–	cycles	Program / erase cycles

Table 102 provides information on the Reset and Stop Mode Recovery pin timing and Table 103 provides information on the Watchdog Timer Electrical Characteristics and Timing.

Table 102. Reset and Stop Mode Recovery Pin Timing

Symbol	Parameter	T_A = -40 °C to 105 °C Minimum	Typical	Maximum	Units	Conditions
T_{RESET}	$\overline{\text{Reset}}$ pin assertion to initiate a System Reset	4	–	–	T_{CLK}	Not in STOP mode. T_{CLK} = System Clock period.
T_{SMR}	Stop Mode Recovery pin Pulse Rejection Period	10	20	40	ns	$\overline{\text{RESET}}$, DBG and GPIO pins configured as SMR sources.

[1] When using the external RC oscillator mode, the oscillator can stop oscillating if the power supply drops below 2.7 V, but before the power supply drops to the voltage brown-out threshold. The oscillator will resume oscillation as soon as the supply voltage exceeds 2.7 V.

Table 103. Watchdog Timer Electrical Characteristics and Timing

Symbol	Parameter	V_{DD} = 2.7–3.6 V T_A = -40 °C to 105 °C Minimum	Typical	Maximum	Units	Conditions
F_{WDT}	WDT Oscillator Frequency	5	10	20	kHz	
I_{WDT}	WDT Oscillator Current including internal RC oscillator	–	< 1	5	µA	

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

Table 116. Additional Symbols

Symbol	Definition
dst	Destination Operand
src	Source Operand
@	Indirect Address Prefix
SP	Stack Pointer
PC	Program Counter
FLAGS	Flags Register
RP	Register Pointer
#	Immediate Operand Prefix
B	Binary Number Suffix
%	Hexadecimal Number Prefix
H	Hexadecimal Number Suffix

Assignment of a value is indicated by an arrow. For example,

dst ← dst + src

indicates the source data is added to the destination data and the result is stored in the destination location.

eZ8 CPU Instruction Classes

eZ8 CPU instructions are divided functionally into the following groups:

- Arithmetic
- Bit Manipulation
- Block Transfer
- CPU Control
- Load
- Logical
- Program Control
- Rotate and Shift

Tables 118 through Table 125 on page 218 contain the instructions belonging to each group and the number of operands required for each instruction. Some instructions appear in more than one table as these instruction can be considered as a subset of more than one category. Within these tables, the source operand is identified as ’src’, the destination operand is ’dst’ and a condition code is ’cc’.

Table 118. Arithmetic Instructions

Mnemonic	Operands	Instruction
ADC	dst, src	Add with Carry
ADCX	dst, src	Add with Carry using Extended Addressing
ADD	dst, src	Add
ADDX	dst, src	Add using Extended Addressing
CP	dst, src	Compare
CPC	dst, src	Compare with Carry
CPCX	dst, src	Compare with Carry using Extended Addressing
CPX	dst, src	Compare using Extended Addressing
DA	dst	Decimal Adjust
DEC	dst	Decrement
DECW	dst	Decrement Word
INC	dst	Increment
INCW	dst	Increment Word
MULT	dst	Multiply