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	Active
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	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	64-LQFP
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f4822ar020sg

Table 5. Z8 Encore! XP F64xx Series Program Memory Maps (Continued)

Program Memory Address (Hex)	Function
0006-0007	Illegal Instruction Trap
0008-0037	Interrupt Vectors*
0038-BFFF	Program Memory
Z8F642x Products	
0000-0001	Option Bits
0002-0003	Reset Vector
0004-0005	WDT Interrupt Vector
0006-0007	Illegal Instruction Trap
0008-0037	Interrupt Vectors*
0038-FFFF	Program Memory

Note: *See Table 23 on page 48 for a list of the interrupt vectors.

Data Memory

The Z8 Encore! XP F64xx Series does not use the eZ8 CPU's 64KB data memory address space.

Information Area

Table 6 describes the Z8 Encore! XP F64xx Series' Information Area. This 512-byte Information Area is accessed by setting bit 7 of the Page Select Register to 1. When access is enabled, the Information Area is mapped into program memory and overlays the 512 bytes at addresses FE00H to FFFFH. When the Information Area access is enabled, execution of the LDC and LDCI instructions from these program memory addresses return the Information Area data rather than the program memory data. Reads of these addresses through the On-Chip Debugger also returns the Information Area data. Execution of code from these addresses continues to correctly use program memory. Access to the Information Area is read-only.

Table 7. Z8 Encore! XP F64xx Series Register File Address Map (Continued)

Address (Hex)	Register Description	Mnemonic	Reset (Hex)	Page
DMA 0 (continued)				
FB2	DMA0 End/Start Address High Nibble	DMA0H	XX	155
FB3	DMA0 Start Address Low Byte	DMA0START	XX	156
FB4	DMA0 End Address Low Byte	DMA0END	XX	156
DMA 1				
FB8	DMA1 Control	DMA1CTL	00	153
FB9	DMA1 I/O Address	DMA1IO	XX	154
FBA	DMA1 End/Start Address High Nibble	DMA1H	XX	155
FBB	DMA1 Start Address Low Byte	DMA1START	XX	156
FBC	DMA1 End Address Low Byte	DMA1END	XX	156
DMA ADC				
FBD	DMA_ADC Address	DMAA_ADDR	XX	157
FBE	DMA_ADC Control	DMAACTL	00	158
FBF	DMA_ADC Status	DMAASTAT	00	159
Interrupt Controller				
FC0	Interrupt Request 0	IRQ0	00	51
FC1	IRQ0 Enable High Bit	IRQ0ENH	00	55
FC2	IRQ0 Enable Low Bit	IRQ0ENL	00	55
FC3	Interrupt Request 1	IRQ1	00	53
FC4	IRQ1 Enable High Bit	IRQ1ENH	00	56
FC5	IRQ1 Enable Low Bit	IRQ1ENL	00	56
FC6	Interrupt Request 2	IRQ2	00	54
FC7	IRQ2 Enable High Bit	IRQ2ENH	00	58
FC8	IRQ2 Enable Low Bit	IRQ2ENL	00	58
FC9–FCC	Reserved	—	XX	
FCD	Interrupt Edge Select	IRQES	00	60
FCE	Interrupt Port Select	IRQPS	00	60
FCF	Interrupt Control	IRQCTL	00	61
GPIO Port A				
FD0	Port A Address	PAADDR	00	40
FD1	Port A Control	PACTL	00	41
FD2	Port A Input Data	PAIN	XX	46

Note: XX = Undefined.

Architecture

Figure 11 displays a block diagram of the interrupt controller.

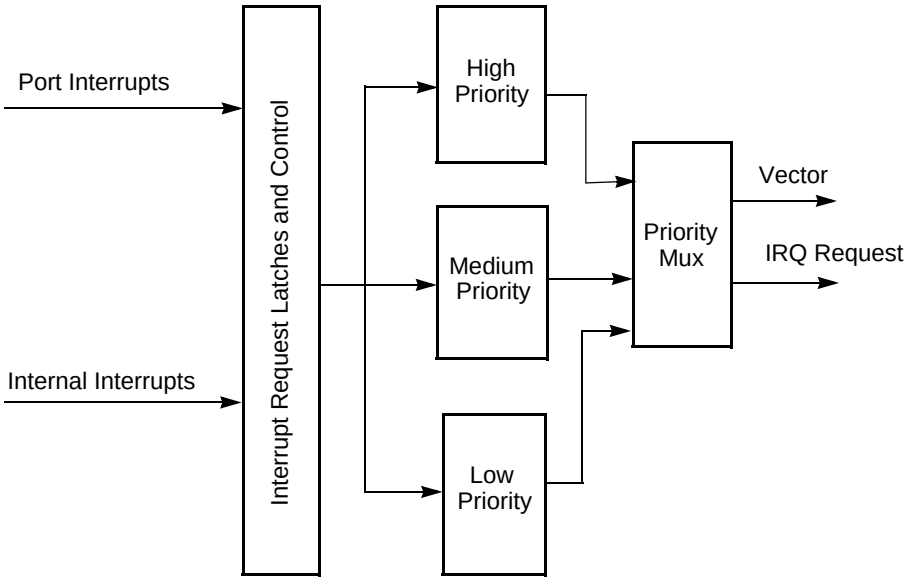

Figure 11. Interrupt Controller Block Diagram

Operation

This section describes the operational aspects of the following functions.

Master Interrupt Enable: see page 49

Interrupt Vectors and Priority: see page 50

Interrupt Assertion: see page 50

Software Interrupt Assertion: see page 51

Master Interrupt Enable

The master interrupt enable bit (IRQE) in the Interrupt Control Register globally enables and disables interrupts.

Interrupts are globally enabled by any of the following actions:

- Executing an Enable Interrupt (EI) instruction
- Executing an Return from Interrupt (IRET) instruction

Bit	Description
[7] TEN	**Timer Enable** 0 = Timer is disabled. 1 = Timer enabled to count.
[6] TPOL	**Timer Input/Output Polarity** Operation of this bit is a function of the current operating mode of the timer. **ONE-SHOT 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. **CONTINUOUS 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. **COUNTER 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. 0 = Count occurs on the rising edge of the timer input signal. 1 = Count occurs on the falling edge of the timer input signal. **PWM Mode** 0 = timer output is forced Low (0) when the timer is disabled. When enabled, the timer output is forced High (1) upon PWM count match and forced Low (0) upon reload. 1 = timer output is forced High (1) when the timer is disabled. When enabled, the timer output is forced Low (0) upon PWM count match and forced High (1) upon reload. **CAPTURE Mode** 0 = Count is captured on the rising edge of the timer input signal. 1 = Count is captured on the falling edge of the timer input signal. **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. **Caution:** When the timer output alternate function TxOUT on a GPIO port pin is enabled, TxOUT will change to whatever state the TPOL bit is in. The timer does not need to be enabled for that to happen. Also, the Port Data Direction Subregister is not needed to be set to output on TxOUT. Changing the TPOL bit with the timer enabled and running does not immediately change the TxOUT.

UART Interrupts

The UART features separate interrupts for the transmitter and the receiver. In addition, when the UART primary functionality is disabled, the Baud Rate Generator can also function as a basic timer with interrupt capability.

Transmitter Interrupts

The transmitter generates a single interrupt when the Transmit Data Register Empty bit (TDRE) is set to 1. This indicates that the transmitter is ready to accept new data for transmission. The TDRE interrupt occurs after the Transmit Shift Register has shifted the first bit of data out. At this point, the Transmit Data Register can be written with the next character to send. This provides 7 bit-periods of latency to load the Transmit Data Register before the Transmit Shift Register completes shifting the current character. Writing to the UART Transmit Data Register clears the TDRE bit to 0.

Receiver Interrupts

The receiver generates an interrupt when any of the following events occurs:

- A data byte has been received and is available in the UART Receive Data Register. This interrupt can be disabled independent of the other receiver interrupt sources. The received data interrupt occurs once the receive character has been received and placed in the Receive Data Register. Software must respond to this received data available condition before the next character is completely received to avoid an overrun error.

Note: In MULTIPROCESSOR Mode (MPEN = 1), the receive data interrupts are dependent on the multiprocessor configuration and the most recent address byte.

- A break is received
- An overrun is detected
- A data framing error is detected

UART Overrun Errors

When an overrun error condition occurs the UART prevents overwriting of the valid data currently in the Receive Data Register. The Break Detect and Overrun status bits are not displayed until after the valid data has been read.

After the valid data has been read, the UART Status 0 Register is updated to indicate the overrun condition (and Break Detect, if applicable). The RDA bit is set to 1 to indicate that the Receive Data Register contains a data byte. However, because the overrun error occurred, this byte may not contain valid data and should be ignored. The BRKD bit indicates if the overrun was caused by a break condition on the line. After reading the status

UART Control 0 and Control 1 Registers

The UART Control 0 and Control 1 Registers, shown in Tables 57 and 58, 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 57. UART Control 0 Register (U*x*CTL0)

Bit	7	6	5	4	3	2	1	0
Field	TEN	REN	CTSE	PEN	PSEL	SBRK	STOP	LBEN
RESET	0							
R/W	R/W							
Address	F42H and F4AH							

Bit	Description
[7] 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.
[6] REN	**Receive Enable** This bit enables or disables the receiver. 0 = Receiver disabled. 1 = Receiver enabled.
[5] 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.
[4] PEN	**Parity Enable** This bit enables or disables parity. Even or odd is determined by the PSEL bit. It 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.
[3] PSEL	**Parity Select** 0 = Even parity is transmitted and expected on all received data. 1 = Odd parity is transmitted and expected on all received data.
[2] SBRK	**Send Break** This bit pauses or breaks data transmission. Sending a break interrupts any transmission in progress, so ensure that the transmitter has finished sending data before setting this bit. 0 = No break is sent. 1 = The output of the transmitter is zero.

Serial Peripheral Interface

The Serial Peripheral Interface is a synchronous interface allowing several SPI-type devices to be interconnected. SPI-compatible devices include EEPROMs, Analog-to-Digital Converters, and ISDN devices. Features of the SPI include:

- Full-duplex, synchronous, character-oriented communication
- Four-wire interface
- Data transfers rates up to a maximum of one-half the system clock frequency
- Error detection
- Dedicated Baud Rate Generator

Architecture

The SPI may be configured as either a Master (in single or multimaster systems) or a Slave as displayed in Figures 22 through 24.

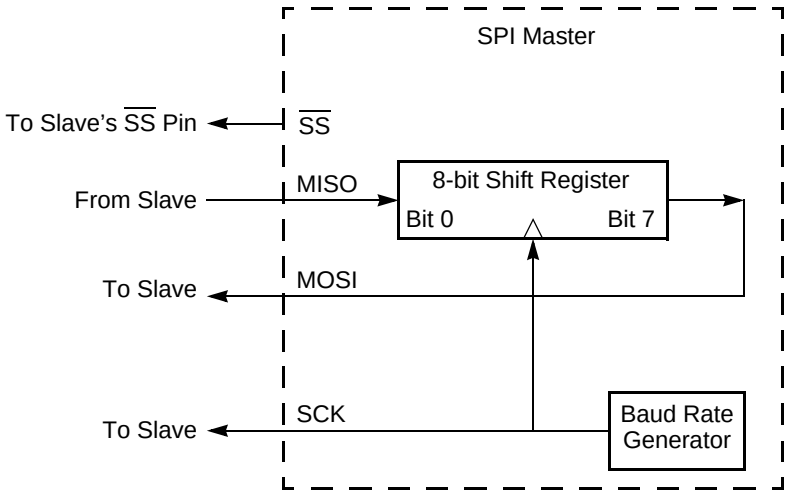

Figure 22. SPI Configured as a Master in a Single-Master, Single-Slave System

DMA*x* Start/Current Address Low Byte Register

The DMA*x* Start/Current Address Low Byte Register, shown in Table 81, in conjunction with the DMA*x* Address High Nibble Register, shown in Table 80, forms a 12-bit Start/Current Address. Writes to this register set the Start Address for DMA operations. Each time the DMA completes a data transfer, the 12-bit Start/Current Address increments by either 1 (single-byte transfer) or 2 (two-byte word transfer). Reads from this register return the low byte of the current address to be used for the next DMA data transfer.

Table 81. DMA*x* Start/Current Address Low Byte Register (DMA*x*START)

Bit	7	6	5	4	3	2	1	0
Field	DMA_START							
RESET	X							
R/W	R/W							
Address	FB3H, FBBH							

Bit	Description
[7:0] DMA_START	**DMA*x* Start/Current Address Low** These bits, with the four lower bits of the DMA*x*_H Register, form the 12-bit Start/Current address. The full 12-bit address is provided by {DMA_START_H[3:0], DMA_START[7:0]}.

DMA*x* End Address Low Byte Register

The DMA*x* End Address Low Byte Register, shown in Table 82, forms a 12-bit End Address.

Table 82. DMA*x* End Address Low Byte Register (DMA*x*END)

Bit	7	6	5	4	3	2	1	0
Field	DMA_END							
RESET	X							
R/W	R/W							
Address	FB4H, FBCH							

Bit	Description
[7] DMA_END	**DMA*x* End Address Low** These bits, with the four upper bits of the DMA*x*_H Register, form a 12-bit address. This address is the ending location of the DMA*x* transfer. The full 12-bit address is provided by {DMA_END_H[3:0], DMA_END[7:0]}.

Bit	Description (Continued)
[2] IRQA	**DMA_ADC Interrupt Request Indicator** This bit is automatically reset to 0 each time a read from this register occurs. 0 = DMA_ADC is not the source of the interrupt from the DMA Controller. 1 = DMA_ADC completed transfer of data from the last ADC analog input and generated an interrupt.
[1] IRQ1	**DMA1 Interrupt Request Indicator** This bit is automatically reset to 0 each time a read from this register occurs. 0 = DMA1 is not the source of the interrupt from the DMA Controller. 1 = DMA1 completed transfer of data to/from the End Address and generated an interrupt.
[0] IRQ0	**DMA0 Interrupt Request Indicator** This bit is automatically reset to 0 each time a read from this register occurs. 0 = DMA0 is not the source of the interrupt from the DMA Controller. 1 = DMA0 completed transfer of data to/from the End Address and generated an interrupt.

On-Chip Debugger

The Z8 Encore! XP F64xx Series products contain an integrated On-Chip Debugger (OCD) that provides advanced debugging features including:

- Reading and writing of the Register File
- Reading and writing of Program and Data memory
- Setting of breakpoints
- Execution of eZ8 CPU instructions

Architecture

The On-Chip Debugger consists of four primary functional blocks: transmitter, receiver, autobaud generator, and debug controller. Figure 36 displays the architecture of the On-Chip Debugger.

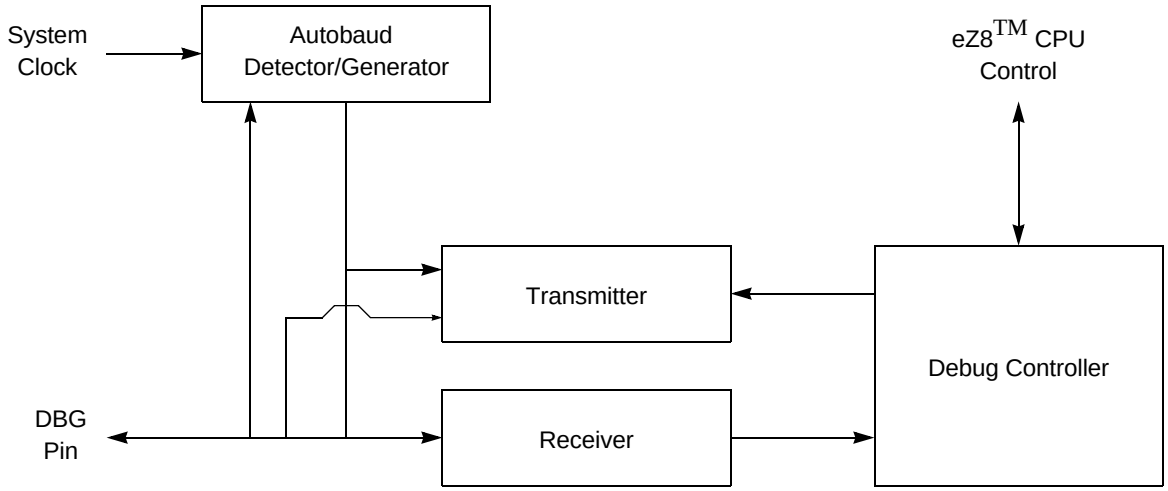

Figure 36. On-Chip Debugger Block Diagram

- Voltage Brown-Out reset
- Asserting the $\overline{\text{RESET}}$ pin Low to initiate a Reset
- Driving the DBG pin Low while the device is in STOP Mode initiates a system reset

OCD Data Format

The OCD interface uses the asynchronous data format defined for RS-232. Each character is transmitted as 1 start bit, 8 data bits (least significant bit first), and 1 stop bit, as shown in Figure 39.

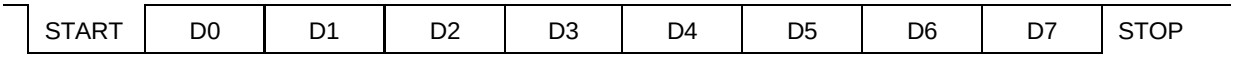

Figure 39. OCD Data Format

OCD Autobaud Detector/Generator

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

The Autobaud Detector/Generator is clocked by the system clock. The minimum baud rate is the system clock frequency divided by 512. For optimal operation, the maximum recommended baud rate is the system clock frequency divided by 8. The theoretical maximum baud rate is the system clock frequency divided by 4. This theoretical maximum is possible for low noise designs with clean signals. Table 101 lists minimum and recommended maximum baud rates for sample crystal frequencies.

Table 101. OCD Baud-Rate Limits

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

Read Data Memory (0DH). The Read Data Memory command reads from Data Memory. This command is equivalent to the LDE and LDEI instructions. Data can be read 1-65536 bytes at a time (65536 bytes can be read by setting size to zero). If the device is not in DEBUG Mode, this command returns FFH for the data.

```
DBG ← 0DH
DBG ← Data Memory Address[15:8]
DBG ← Data Memory Address[7:0]
DBG ← Size[15:8]
DBG ← Size[7:0]
DBG 1-65536 data bytes
```

Read Program Memory CRC (0EH). The Read Program Memory CRC command computes and returns the CRC (cyclic redundancy check) of program memory using the 16-bit CRC-CCITT polynomial. If the device is not in DEBUG Mode, this command returns FFFFH for the CRC value. Unlike most other OCD Read commands, there is a delay from issuing of the command until the OCD returns the data. The OCD reads program memory, calculates the CRC value, and returns the result. The delay is a function of the program memory size and is approximately equal to the system clock period multiplied by the number of bytes in program memory.

```
DBG ← 0EH
DBG CRC[15:8]
DBG CRC[7:0]
```

Step Instruction (10H). The Step Instruction command steps one assembly instruction at the current Program Counter (PC) location. If the device is not in DEBUG Mode or the Read Protect option bit is enabled, the OCD ignores this command.

```
DBG ← 10H
```

Stuff Instruction (11H). The Stuff Instruction command steps one assembly instruction and allows specification of the first byte of the instruction. The remaining 0-4 bytes of the instruction are read from program memory. This command is useful for stepping over instructions where the first byte of the instruction has been overwritten by a breakpoint. If the device is not in DEBUG Mode or the Read Protect option bit is enabled, the OCD ignores this command.

```
DBG ← 11H
DBG ← opcode[7:0]
```

Execute Instruction (12H). The Execute Instruction command allows sending an entire instruction to be executed to the eZ8 CPU. This command can also step over breakpoints. The number of bytes to send for the instruction depends on the op code. If the device is not in DEBUG Mode or the Read Protect option bit is enabled, the OCD ignores this command

```
DBG ← 12H
DBG ← 1-5 byte opcode
```

Figure 46 displays the maximum HALT Mode current consumption across the full operating temperature range of the device and plotted opposite the system clock frequency. All GPIO pins are configured as outputs and driven High.

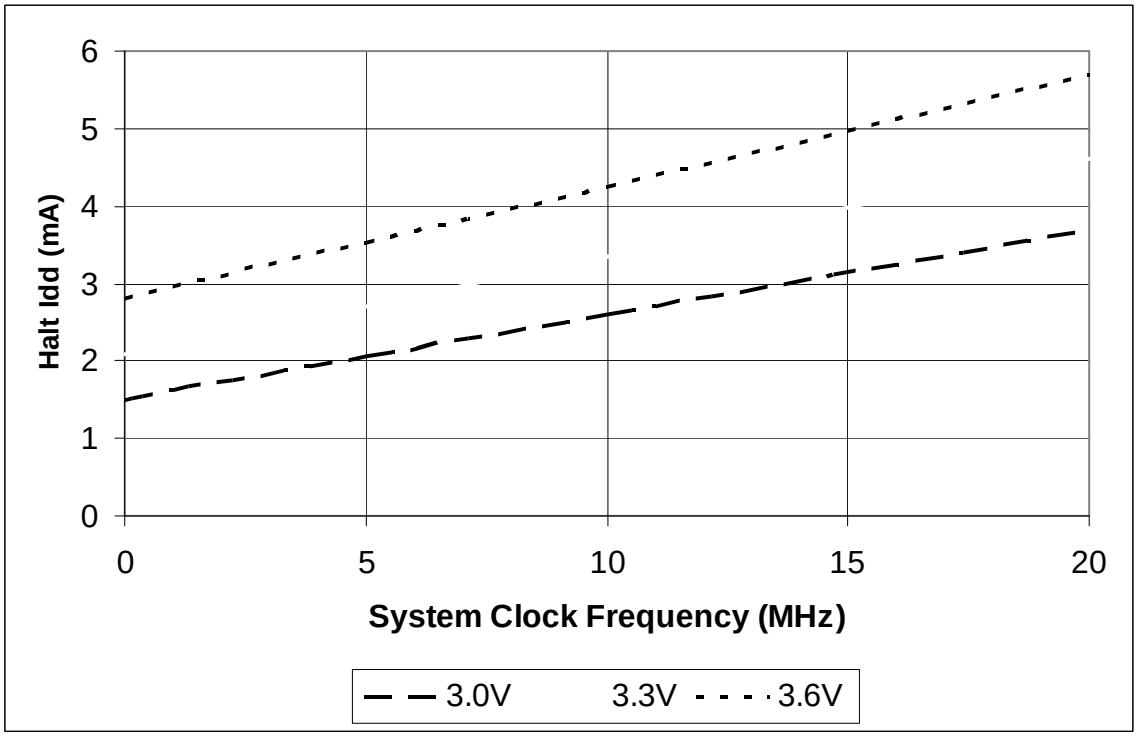

Figure 46. Maximum HALT Mode I_{CC} vs. System Clock Frequency

General-Purpose I/O Port Input Data Sample Timing

Figure 50 displays timing of the GPIO Port input sampling. Table 115 lists the GPIO port input timing.

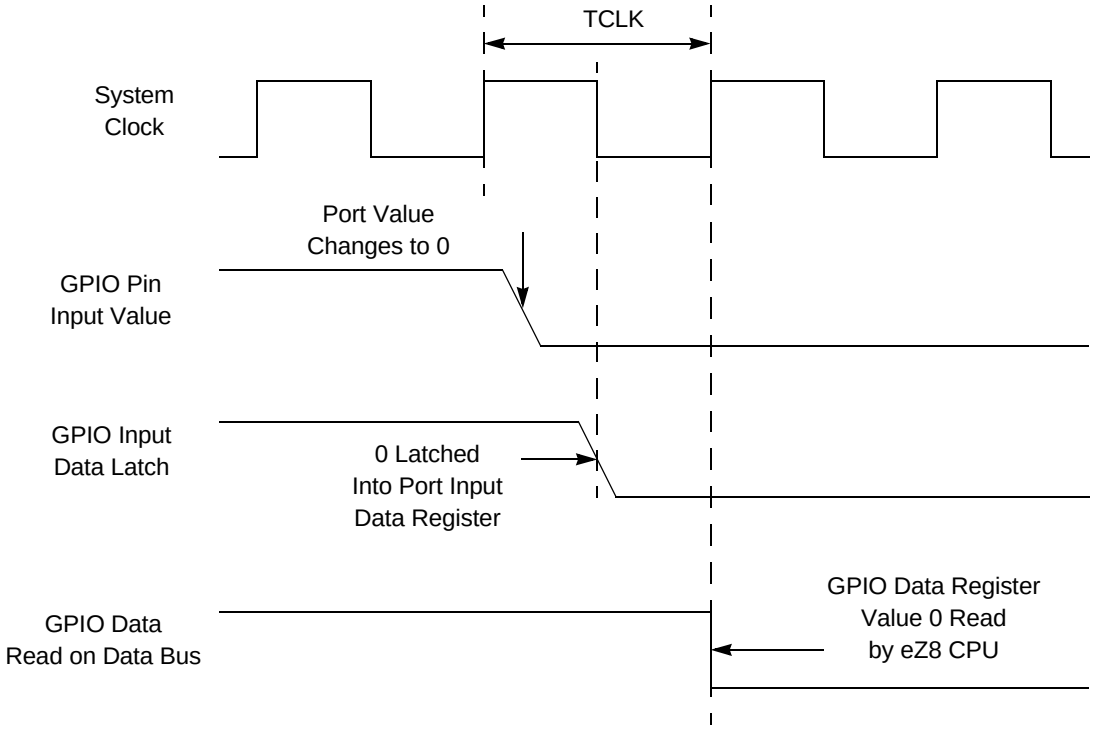

Figure 50. Port Input Sample Timing

Table 115. GPIO Port Input Timing

		Delay (ns)	
Parameter	**Abbreviation**	**Min**	**Max**
T_{S_PORT}	Port Input Transition to X_{IN} Fall Setup Time (not pictured)	5	–
T_{H_PORT}	X_{IN} Fall to Port Input Transition Hold Time (not pictured)	6	–
T_{SMR}	GPIO Port Pin Pulse Width to Insure Stop Mode Recovery (for GPIO Port pins enabled as SMR sources)	1 µs	

Table 124. Assembly Language Syntax Example 2

Assembly Language Code	ADD	43H,	R8	(ADD dst, src)
Object Code	04	E8	43	(OPC src, dst)

Refer to the device-specific Product Specification to determine the exact register file range available. The register file size varies, depending on the device type.

eZ8 CPU Instruction Notation

In the eZ8 CPU Instruction Summary and Description sections, the operands, condition codes, status flags, and address modes are represented by a notational shorthand that is described in Table 125.

Table 125. Notational Shorthand

Notation	Description	Operand	Range
b	Bit	b	b represents a value from 0 to 7 (000B to 111B).
cc	Condition Code	—	Refer to 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.

Table 131. CPU Control Instructions

Mnemonic	Operands	Instruction
ATM	—	Atomic Execution
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 132. 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
LDWX	dst, src	Load Word using Extended Addressing
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

General-Purpose Input/Output (GPIO)

For more information about these GPIO Control registers, see the GPIO Control Register Definitions section on page 39.

Hex Address: FD0

Table 229. Port A–H 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							
Address	FD0H, FD4H, FD8H, FDCH, FE0H, FE4H, FE8H, FECH							

Hex Address: FD1

Table 230. Port A–H Control Registers (P*x*CTL)

Bit	7	6	5	4	3	2	1	0
Field	PCTL							
RESET	00H							
R/W	R/W							
Address	FD1H, FD5H, FD9H, FDDH, FE1H, FE5H, FE9H, FEDH							

Hex Address: FD2

Table 231. Port A–H Input Data Registers (PxIN)

Bit	7	6	5	4	3	2	1	0
Field	PIN7	PIN6	PIN5	PIN4	PIN3	PIN2	PIN1	PIN0
RESET	X							
R/W	R							
Address	FD2H, FD6H, FDAH, FDEH, FE2H, FE6H, FEAH, FEEH							

Hex Address: FE7

Table 252. Port A–H Output Data Register (P*x*OUT)

Bit	7	6	5	4	3	2	1	0
Field	POUT7	POUT6	POUT5	POUT4	POUT3	POUT2	POUT1	POUT0
RESET	0							
R/W	R/W							
Address	FD3H, FD7H, FDBH, FDFH, FE3H, FE7H, FEBH, FEFH							

Hex Address: FE8

Table 253. Port A–H 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							
Address	FD0H, FD4H, FD8H, FDCH, FE0H, FE4H, FE8H, FECH							

Hex Address: FE9

Table 254. Port A–H Control Registers (P*x*CTL)

Bit	7	6	5	4	3	2	1	0
Field	PCTL							
RESET	00H							
R/W	R/W							
Address	FD1H, FD5H, FD9H, FDDH, FE1H, FE5H, FE9H, FEDH							

Hex Address: FEA

Table 255. Port A–H Input Data Registers (PxIN)

Bit	7	6	5	4	3	2	1	0
Field	PIN7	PIN6	PIN5	PIN4	PIN3	PIN2	PIN1	PIN0
RESET	X							
R/W	R							
Address	FD2H, FD6H, FDAH, FDEH, FE2H, FE6H, FEAH, FEEH							

T

U