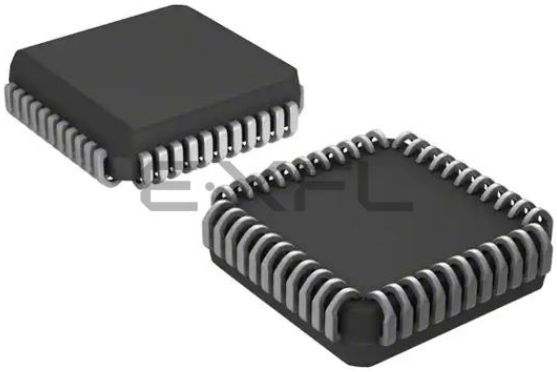

**Welcome to E-XFL.COM**

## 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 | 31 |
| Program Memory Size | 16KB (16K x 8) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 2K x 8 |
| Voltage - Supply (Vcc/Vdd) | 3V ~ 3.6V |
| Data Converters | A/D 8x10b |
| Oscillator Type | Internal |
| Operating Temperature | 0°C ~ 70°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 44-LCC (J-Lead) |
| Supplier Device Package | - |
| Purchase URL | https://www.e-xfl.com/product-detail/zilog/z8f1621vn020sg |

# Introduction

Zilog's Z8 Encore! XP F64xx Series MCU family of products are a line of Zilog microcontroller products based upon the 8-bit eZ8 CPU. The Z8 Encore! XP F64xx Series adds Flash memory to Zilog's extensive line of 8-bit microcontrollers. The Flash in-circuit programming capability allows for faster development time and program changes in the field. The new eZ8 CPU is upward-compatible with existing Z8 instructions. The rich-peripheral set of the Z8 Encore! XP F64xx Series makes it suitable for a variety of applications including motor control, security systems, home appliances, personal electronic devices, and sensors.

## Features

The features of Z8 Encore! XP F64xx Series include:

- 20MHz eZ8 CPU
- Up to 64KB Flash with in-circuit programming capability
- Up to 4KB register RAM
- 12-channel, 10-bit Analog-to-Digital Converter (ADC)
- Two full-duplex 9-bit UARTs with bus transceiver Driver Enable control
- Inter-integrated circuit ($I^2C$)
- Serial Peripheral Interface (SPI)
- Two Infrared Data Association (IrDA)-compliant infrared encoder/decoders
- Up to four 16-bit timers with capture, compare and PWM capability
- Watchdog Timer (WDT) with internal RC oscillator
- Three-channel DMA
- Up to 60 input/output (I/O) pins
- 24 interrupts with configurable priority
- On-Chip Debugger
- Voltage Brown-Out (VBO) Protection
- Power-On Reset (POR)
- Operating voltage of 3.0V to 3.6V with 5V-tolerant inputs
- 0°C to +70°C, –40°C to +105°C, and –40°C to +125°C operating temperature ranges

# Part Selection Guide

Table 1 identifies the basic features and package styles available for each device within the Z8 Encore! XP product line.

**Table 1. Z8 Encore! XP F64xx Series Part Selection Guide**

| Part Number | Flash (KB) | RAM (KB) | I/O | 16-bit Timers with PWM | ADC Inputs | UARTs with IrDA | $I^2C$ | SPI | 40-/ 44-Pin Package | 64/68-Pin Package | 80-Pin Package |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Z8F1621 | 16 | 2 | 31 | 3 | 8 | 2 | 1 | 1 | X | | |
| Z8F1622 | 16 | 2 | 46 | 4 | 12 | 2 | 1 | 1 | | X | |
| Z8F2421 | 24 | 2 | 31 | 3 | 8 | 2 | 1 | 1 | X | | |
| Z8F2422 | 24 | 2 | 46 | 4 | 12 | 2 | 1 | 1 | | X | |
| Z8F3221 | 32 | 2 | 31 | 3 | 8 | 2 | 1 | 1 | X | | |
| Z8F3222 | 32 | 2 | 46 | 4 | 12 | 2 | 1 | 1 | | X | |
| Z8F4821 | 48 | 4 | 31 | 3 | 8 | 2 | 1 | 1 | X | | |
| Z8F4822 | 48 | 4 | 46 | 4 | 12 | 2 | 1 | 1 | | X | |
| Z8F4823 | 48 | 4 | 60 | 4 | 12 | 2 | 1 | 1 | | | X |
| Z8F6421 | 64 | 4 | 31 | 3 | 8 | 2 | 1 | 1 | X | | |
| Z8F6422 | 64 | 4 | 46 | 4 | 12 | 2 | 1 | 1 | | X | |
| Z8F6423 | 64 | 4 | 60 | 4 | 12 | 2 | 1 | 1 | | | X |

Note: For die form sales, contact your local Zilog Sales Office.

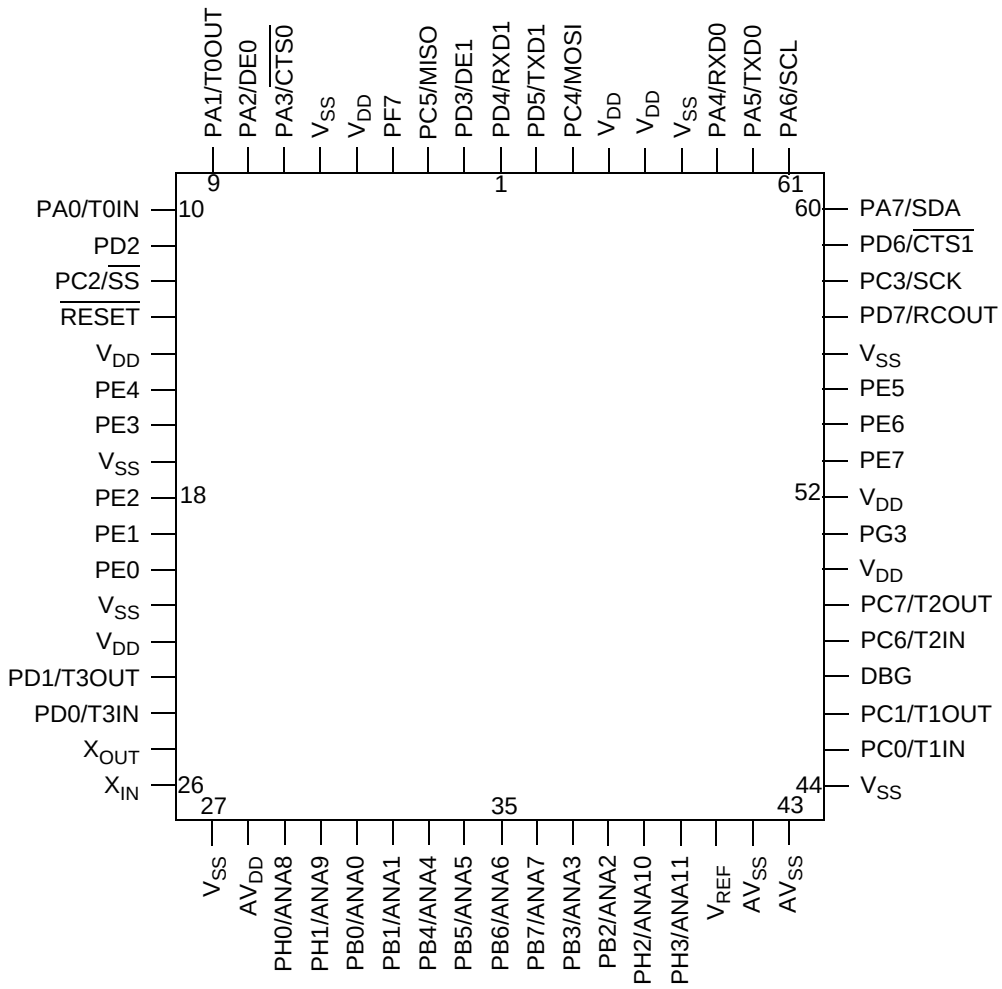


**Figure 6. Z8 Encore! XP F64xx Series in 68-Pin Plastic Leaded Chip Carrier (PLCC)**

### Watchdog Timer Reset

If the device is in normal or HALT Mode, the Watchdog Timer can initiate a system reset at time-out if the WDT_RES option bit is set to 1. This capability is the default (unprogrammed) setting of the WDT_RES option bit. The WDT status bit in the WDT Control Register is set to signify that the reset was initiated by the Watchdog Timer.

### External Pin Reset

The $\overline{\text{RESET}}$ pin has a Schmitt-triggered input, an internal pull-up, an analog filter and a digital filter to reject noise. Once the $\overline{\text{RESET}}$ pin is asserted for at least 4 system clock cycles, the devices progress through the system reset sequence. While the $\overline{\text{RESET}}$ input pin is asserted Low, the Z8 Encore! XP F64xx Series devices continue to be held in the Reset state. If the $\overline{\text{RESET}}$ pin is held Low beyond the system reset time-out, the devices exit the Reset state immediately following $\overline{\text{RESET}}$ pin deassertion. Following a system reset initiated by the external $\overline{\text{RESET}}$ pin, the EXT status bit in the Watchdog Timer Control (WDTCTL) Register is set to 1.

### On-Chip Debugger Initiated Reset

A Power-On Reset can be initiated using the On-Chip Debugger by setting the RST bit in the OCD Control Register. The On-Chip Debugger block is not reset but the rest of the chip goes through a normal system reset. The RST bit automatically clears during the system reset. Following the system reset the POR bit in the WDT Control Register is set.

## Stop Mode Recovery

STOP Mode is entered by the eZ8 executing a stop instruction. For detailed STOP Mode information, see the Low-Power Modes chapter on page 34. During Stop Mode Recovery, the devices are held in reset for 66 cycles of the Watchdog Timer oscillator followed by 16 cycles of the system clock. Stop Mode Recovery only affects the contents of the Watchdog Timer Control Register. Stop Mode Recovery does not affect any other values in the Register File, including the Stack Pointer, Register Pointer, Flags, peripheral control registers, and general-purpose RAM.

The eZ8 CPU fetches the Reset vector at program memory addresses 0002H and 0003H and loads that value into the Program Counter. Program execution begins at the Reset vector address. Following Stop Mode Recovery, the stop bit in the Watchdog Timer Control Register is set to 1. Table 10 lists the Stop Mode Recovery sources and resulting actions.

**Table 12. Port Alternate Function Mapping (Continued)**

| Port | Pin | Mnemonic | Alternate Function Description |
|---|---|---|---|
| **Port F** | PF[7:0] | N/A | No alternate functions |
| **Port G** | PG[7:0] | N/A | No alternate functions |
| **Port H** | PH0 | ANA8 | ADC analog input 8 |
| | PH1 | ANA9 | ADC analog input 9 |
| | PH2 | ANA10 | ADC analog input 10 |
| | PH3 | ANA11 | ADC analog input 11 |

# GPIO Interrupts

Many of the GPIO port pins can be used as interrupt sources. Some port pins may be configured to generate an interrupt request on either the rising edge or falling edge of the pin input signal. Other port pin interrupts generate an interrupt when any edge occurs (both rising and falling). For more information about interrupts using the GPIO pins, see the Interrupt Controller chapter on page 47.

# GPIO Control Register Definitions

Four registers for each Port provide access to GPIO control, input data, and output data. Table 13 lists these Port registers. Use the Port A–H Address and Control registers together to provide access to subregisters for Port configuration and control.

**Table 13. GPIO Port Registers and Subregisters**

| Port Register Mnemonic | Port Register Name |
|---|---|
| P*x*ADDR | Port A–H Address Register (selects subregisters) |
| P*x*CTL | Port A–H Control Register (provides access to subregisters) |
| P*x*IN | Port A–H Input Data Register |
| P*x*OUT | Port A–H Output Data Register |
| **Port Subregister Mnemonic** | **Port Register Name** |
| P*x*DD | Data Direction |
| P*x*AF | Alternate Function |
| P*x*OC | Output Control (Open-Drain) |
| PxDD | High Drive Enable |
| P*x*SMRE | Stop Mode Recovery Source Enable |

## Port A–H High Drive Enable Subregisters

The Port A–H High Drive Enable Subregister, shown in Table 19, is accessed through the Port A–H Control Register by writing 04H to the Port A–H Address Register. Setting the bits in the Port A–H High Drive Enable subregisters to 1 configures the specified port pins for high-current output drive operation. The Port A–H High Drive Enable Subregister affects the pins directly and, as a result, alternate functions are also affected.

**Table 19. Port A–H High Drive Enable Subregisters**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | PHDE7 | PHDE6 | PHDE5 | PHDE4 | PHDE3 | PHDE2 | PHDE1 | PHDE0 |
| RESET | 0 | | | | | | | |
| R/W | R/W | | | | | | | |
| Address | See note. | | | | | | | |

Note: If a 04H exists in the Port A–H Address Register, it is accessible through the Port A–H Control Register.

| Bit | Description |
|---|---|
| [7:0]<br>PHDEx | **Port High Drive Enabled**<br>0 = The Port pin is configured for standard output current drive.<br>1 = The Port pin is configured for high output current drive. |

Note: x indicates register bits in the range [7:0].

If TPOL is set to 0, the ratio of the PWM output High time to the total period is calculated using the following equation:

$$\text{PWM Output High Time Ratio (\%)} = \frac{\text{Reload Value} - \text{PWM Value}}{\text{Reload Value}} \times 100$$

If TPOL is set to 1, the ratio of the PWM output High time to the total period is calculated using the following equation:

$$\text{PWM Output High Time Ratio (\%)} = \frac{\text{PWM Value}}{\text{Reload Value}} \times 100$$

## CAPTURE Mode

In CAPTURE Mode, the current timer count value is recorded when the appropriate external timer input transition occurs. The Capture count value is written to the Timer PWM High and Low Byte Registers. The timer input is the system clock. The TPOL bit in the Timer Control 1 Register determines if the Capture occurs on a rising edge or a falling edge of the timer input signal. When the capture event occurs, an interrupt is generated and the timer continues counting.

The timer continues counting up to the 16-bit reload value stored in the Timer Reload High and Low Byte registers. Upon reaching the reload value, the timer generates an interrupt and continues counting.

Observe the following procedure for configuring a timer for CAPTURE Mode and initiating the count:

1. Write to the Timer Control 1 Register to:
   - Disable the timer
   - Configure the timer for CAPTURE Mode
   - Set the prescale value
   - Set the Capture edge (rising or falling) for the timer input
2. Write to the Timer High and Low Byte registers to set the starting count value (typically `0001H`).
3. Write to the Timer Reload High and Low Byte registers to set the reload value.
4. Clear the Timer PWM High and Low Byte registers to `0000H`. This allows the software to determine if interrupts were generated by either a capture event or a reload. If the PWM High and Low Byte registers still contain `0000H` after the interrupt, then the interrupt was generated by a reload.
5. If appropriate, enable the timer interrupt and set the timer interrupt priority by writing to the relevant interrupt registers.

ister. This operation allows accurate reads of the full 16-bit timer count value while enabled. When the timers are not enabled, a read from the Timer Low Byte Register returns the actual value in the counter.

### Timer Output Signal Operation

A timer output is a GPIO port pin alternate function. Generally, the timer output is toggled every time the counter is reloaded.

## Timer Control Register Definitions

This section defines the features of the following Timer Control registers.

Timer 0–3 High and Low Byte Registers: see page 72

Timer Reload High and Low Byte Registers: see page 74

Timer 0–3 PWM High and Low Byte Registers: see page 75

Timer 0–3 Control 0 Registers: see page 76

Timer 0–3 Control 1 Registers: see page 77

Timers 0–2 are available in all packages. Timer 3 is only available in 64-, 68- and 80-pin packages.

### Timer 0–3 High and Low Byte Registers

The Timer 0–3 High and Low Byte (TxH and TxL) registers, shown in Tables 39 and 40, contain the current 16-bit timer count value. When the timer is enabled, a read from TxH causes the value in TxL to be stored in a temporary holding register. A read from TMRL always returns this temporary register when the timers are enabled. When the timer is disabled, reads from the TMRL read the register directly.

Writing to the Timer High and Low Byte registers while the timer is enabled is not recommended. There are no temporary holding registers available for write operations, so simultaneous 16-bit writes are not possible. If either the Timer High or Low Byte registers are written during counting, the 8-bit written value is placed in the counter (High or Low Byte) at the next clock edge. The counter continues counting from the new value.

Timer 3 is unavailable in 44-pin packages.

**Table 47. Watchdog Timer Approximate Time-Out Delays**

| WDT Reload Value (Hex) | WDT Reload Value (Decimal) | Approximate Time-Out Delay (with 10kHz typical WDT Oscillator Frequency) | |
|---|---|---|---|
| | | Typical | Description |
| 000004 | 4 | 400µs | Minimum time-out delay |
| FFFFFF | 16,777,215 | 1677.5s | Maximum time-out delay |

## Watchdog Timer Refresh

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

When the Z8 Encore! XP F64xx Series devices are operating in DEBUG Mode (through the On-Chip Debugger), the Watchdog Timer is continuously refreshed to prevent spurious Watchdog Timer time-outs.

## Watchdog Timer Time-Out Response

The Watchdog Timer times out when the counter reaches 000000H. A time-out of the Watchdog Timer generates either an interrupt or a Reset. The WDT_RES option bit determines the time-out response of the Watchdog Timer. For information about programming of the WDT_RES option bit, see the Option Bits chapter on page 180.

### WDT Interrupt in Normal Operation

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

### WDT Interrupt in STOP Mode

If configured to generate an interrupt when a time-out occurs and the Z8 Encore! XP F64xx Series devices are in STOP Mode, the Watchdog Timer automatically initiates a Stop Mode Recovery and generates an interrupt request. Both the WDT status bit and the stop bit in the Watchdog Timer Control Register are set to 1 following WDT time-out in

In MULTIPROCESSOR (9-Bit) Mode, the parity bit location (9th bit) becomes the MULTIPROCESSOR control bit. The UART Control 1 and Status 1 registers provide MULTIPROCESSOR (9-Bit) Mode control and status information. If an automatic address matching scheme is enabled, the UART Address Compare Register holds the network address of the device.

### MULTIPROCESSOR (9-bit) Mode Receive Interrupts

When MULTIPROCESSOR Mode is enabled, the UART only processes frames addressed to it. The determination of whether a frame of data is addressed to the UART can be made in hardware, software or some combination of the two, depending on the multiprocessor configuration bits. In general, the address compare feature reduces the load on the CPU, since it does not need to access the UART when it receives data directed to other devices on the multinode network. The following three MULTIPROCESSOR modes are available in hardware:

- Interrupt on all address bytes
- Interrupt on matched address bytes and correctly framed data bytes
- Interrupt only on correctly framed data bytes

These modes are selected with MPMD[1:0] in the UART Control 1 Register. For all MULTIPROCESSOR modes, bit MPEN of the UART Control 1 Register must be set to 1.

The first scheme is enabled by writing 01b to MPMD[1:0]. In this mode, all incoming address bytes cause an interrupt, while data bytes never cause an interrupt. The interrupt service routine must manually check the address byte that caused triggered the interrupt. If it matches the UART address, the software clears MPMD[0]. At this point, each new incoming byte interrupts the CPU. The software is then responsible for determining the end of the frame. It checks for end-of-frame by reading the MPRX bit of the UART Status 1 Register for each incoming byte. If MPRX=1, a new frame has begun. If the address of this new frame is different from the UART's address, then set MPMD[0] to 1 causing the UART interrupts to go inactive until the next address byte. If the new frame's address matches the UART's, the data in the new frame is processed as well.

The second scheme is enabled by setting MPMD[1:0] to 10b and writing the UART's address into the UART Address Compare Register. This mode introduces more hardware control, interrupting only on frames that match the UART's address. When an incoming address byte does not match the UART's address, it is ignored. All successive data bytes in this frame are also ignored. When a matching address byte occurs, an interrupt is issued and further interrupts now occur on each successive data byte. The first data byte in the frame contains the NEWFRM=1 in the UART Status 1 Register. When the next address byte occurs, the hardware compares it to the UART's address. If there is a match, the interrupts continue sand the NEWFRM bit is set for the first byte of the new frame. If there is no match, then the UART ignores all incoming bytes until the next address match.

## UART Status 0 Register

The UART Status 0 Register, shown in Table 55, identifies the current UART operating configuration and status.

**Table 55. UART Status 0 Register (U*x*STAT0)**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | RDA | PE | OE | FE | BRKD | TDRE | TXE | CTS |
| RESET | 0 | | | | | | 1 | X |
| R/W | R | | | | | | | |
| Address | F41H and F49H | | | | | | | |

| Bit | Description |
|---|---|
| [7]<br>RDA | **Receive Data Available**<br>This bit indicates that the UART Receive Data Register has received data. Reading the UART Receive Data Register clears this bit.<br>0 = The UART Receive Data Register is empty.<br>1 = There is a byte in the UART Receive Data Register. |
| [6]<br>PE | **Parity Error**<br>This bit indicates that a parity error has occurred. Reading the UART Receive Data Register clears this bit.<br>0 = No parity error occurred.<br>1 = A parity error occurred. |
| [5]<br>OE | **Overrun Error**<br>This bit indicates that an overrun error has occurred. An overrun occurs when new data is received and the UART Receive Data Register has not been read. If the RDA bit is reset to 0, then reading the UART Receive Data Register clears this bit.<br>0 = No overrun error occurred.<br>1 = An overrun error occurred. |
| [4]<br>FE | **Framing Error**<br>This bit indicates that a framing error (no stop bit following data reception) was detected. Reading the UART Receive Data Register clears this bit.<br>0 = No framing error occurred.<br>1 = A framing error occurred. |
| [3]<br>BRKD | **Break Detect**<br>This bit indicates that a break occurred. If the data bits, parity/multiprocessor bit, and stop bit(s) are all zeros then this bit is set to 1. Reading the UART Receive Data Register clears this bit.<br>0 = No break occurred.<br>1 = A break occurred. |

## 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]<br>TEN | **Transmit Enable**<br>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.<br>0 = Transmitter disabled.<br>1 = Transmitter enabled. |
| [6]<br>REN | **Receive Enable**<br>This bit enables or disables the receiver.<br>0 = Receiver disabled.<br>1 = Receiver enabled. |
| [5]<br>CTSE | **CTS Enable**<br>0 = The $\overline{\text{CTS}}$ signal has no effect on the transmitter.<br>1 = The UART recognizes the $\overline{\text{CTS}}$ signal as an enable control from the transmitter. |
| [4]<br>PEN | **Parity Enable**<br>This bit enables or disables parity. Even or odd is determined by the PSEL bit. It is overridden by the MPEN bit.<br>0 = Parity is disabled.<br>1 = The transmitter sends data with an additional parity bit and the receiver receives an additional parity bit. |
| [3]<br>PSEL | **Parity Select**<br>0 = Even parity is transmitted and expected on all received data.<br>1 = Odd parity is transmitted and expected on all received data. |
| [2]<br>SBRK | **Send Break**<br>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.<br>0 = No break is sent.<br>1 = The output of the transmitter is zero. |

### Serial Clock

The Serial Clock (SCK) synchronizes data movement both in and out of the device through its MOSI and MISO pins. In MASTER Mode, the SPI's Baud Rate Generator creates the serial clock. The Master drives the serial clock out its own SCK pin to the Slave's SCK pin. When the SPI is configured as a Slave, the SCK pin is an input and the clock signal from the Master synchronizes the data transfer between the Master and Slave devices. Slave devices ignore the SCK signal, unless the $\overline{SS}$ pin is asserted. When configured as a slave, the SPI block requires a minimum SCK period of greater than or equal to 8 times the system ($X_{IN}$) clock period.

The Master and Slave are each capable of exchanging a character of data during a sequence of NUMBITS clock cycles (see the NUMBITS field in the SPI Mode Register section on page 125). In both Master and Slave SPI devices, data is shifted on one edge of the SCK and is sampled on the opposite edge where data is stable. Edge polarity is determined by the SPI phase and polarity control.

### Slave Select

The active Low Slave Select ($\overline{SS}$) input signal selects a Slave SPI device. $\overline{SS}$ must be Low prior to all data communication to and from the Slave device. $\overline{SS}$ must stay Low for the full duration of each character transferred. The $\overline{SS}$ signal may stay Low during the transfer of multiple characters or may deassert between each character.

When the SPI is configured as the only Master in an SPI system, the $\overline{SS}$ pin can be set as either an input or an output. Other GPIO output pins can also be employed to select external SPI Slave devices.

When the SPI is configured as one Master in a multimaster SPI system, the $\overline{SS}$ pin must be set as an input. The $\overline{SS}$ input signal on the Master must be High. If the $\overline{SS}$ signal goes Low (indicating another Master is driving the SPI bus), a collision error flag is set in the SPI Status Register.

## SPI Clock Phase and Polarity Control

The SPI supports four combinations of serial clock phase and polarity using two bits in the SPI Control Register. The clock polarity bit, CLKPOL, selects an active high or active Low clock and has no effect on the transfer format. Table 63 lists the SPI Clock Phase and Polarity Operation parameters. The clock phase bit, PHASE, selects one of two fundamentally different transfer formats. For proper data transmission, the clock phase and polarity must be identical for the SPI Master and the SPI Slave. The Master always places data on the MOSI line a half-cycle before the receive clock edge (SCK signal), in order for the Slave to latch the data.

## DMA Control of the ADC

The Direct Memory Access (DMA) Controller can control operation of the ADC including analog input selection and conversion enable. For more information about the DMA and configuring for ADC operations, see the Direct Memory Access Controller chapter on page 150.

# ADC Control Register Definitions

This section defines the features of the following ADC Control registers.

ADC Control Register: see page 165

ADC Data High Byte Register: see page 167

ADC Data Low Bits Register: see page 168

## ADC Control Register

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

**Table 87. ADC Control Register (ADCCTL)**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | CEN | Reserved | VREF | CONT | ANAIN[3:0] | | | |
| RESET | 0 | | 1 | 0 | | | | |
| R/W | R/W | | | | | | | |
| Address | F70H | | | | | | | |

| Bit | Description |
|---|---|
| [7]<br>CEN | **Conversion Enable**<br>0 = Conversion is complete. Writing a 0 produces no effect. The ADC automatically clears this bit to 0 when a conversion has been completed.<br>1 = Begin conversion. Writing a 1 to this bit starts a conversion. If a conversion is already in progress, the conversion restarts. This bit remains 1 until the conversion is complete. |
| [6] | **Reserved**<br>This bit is reserved and must be programmed to 0. |
| [5]<br>VREF | **Voltage Reference**<br>0 = Internal voltage reference generator enabled. The $V_{REF}$ pin should be left unconnected (or capacitively coupled to analog ground) if the internal voltage reference is selected as the ADC reference voltage.<br>1 = Internal voltage reference generator disabled. An external voltage reference must be provided through the $V_{REF}$ pin. |

## ADC Data High Byte Register

The ADC Data High Byte Register, shown in Table 88, contains the upper eight bits of the 10-bit ADC output. During a single-shot conversion, this value is invalid. Access to the ADC Data High Byte Register is read-only. The full 10-bit ADC result is provided by {ADCD_H[7:0], ADCD_L[7:6]}. Reading the ADC Data High Byte Register latches data in the ADC Low Bits Register.

**Table 88. ADC Data High Byte Register (ADCD_H)**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | ADCD_H | | | | | | | |
| RESET | X | | | | | | | |
| R/W | R | | | | | | | |
| Address | F72H | | | | | | | |

| Bit | Description |
|---|---|
| [7:0] ADCD_H | **ADC Data High Byte**<br>This byte contains the upper eight bits of the 10-bit ADC output. These bits are not valid during a single-shot conversion. During a continuous conversion, the last conversion output is held in this register. These bits are undefined after a Reset. |

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.

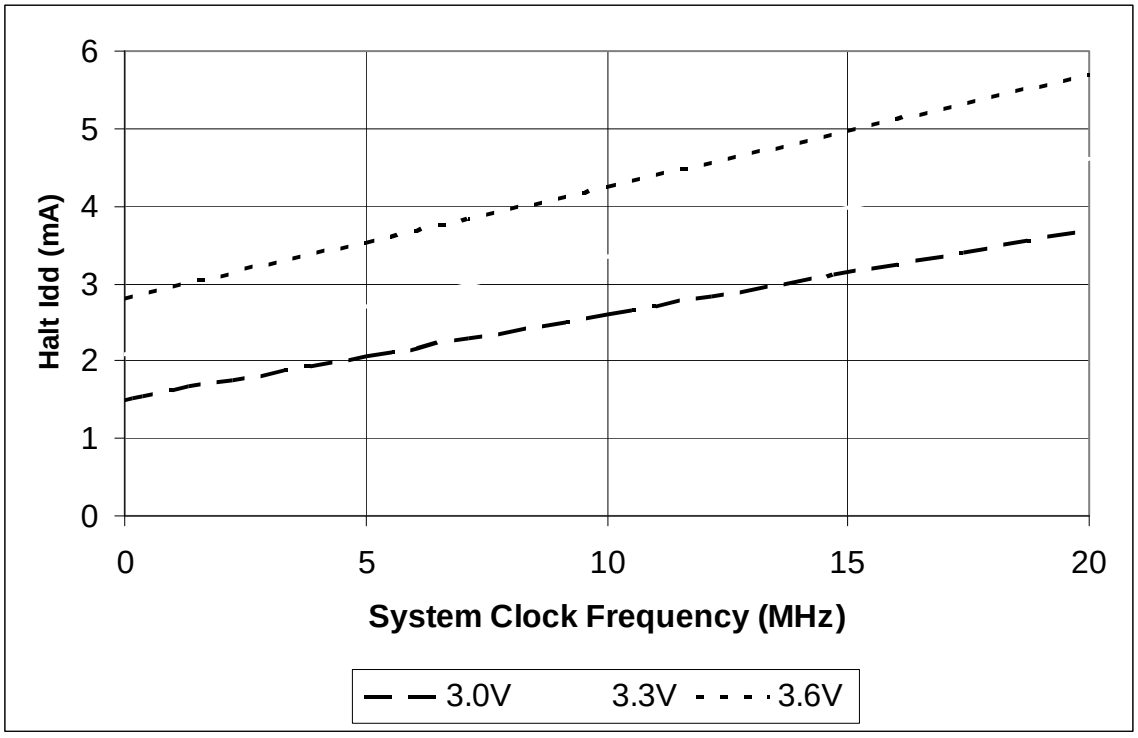


**Figure 46. Maximum HALT Mode $I_{CC}$ vs. System Clock Frequency**

**Table 136. 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DA dst | dst ← DA(dst) | R | | 40 | * | * | * | X | – | – | 2 | 2 |
| | | IR | | 41 | | | | | | | 2 | 3 |
| DEC dst | dst ← dst – 1 | R | | 30 | – | * | * | * | – | – | 2 | 2 |
| | | IR | | 31 | | | | | | | 2 | 3 |
| DECW dst | dst ← dst – 1 | RR | | 80 | – | * | * | * | – | – | 2 | 5 |
| | | IRR | | 81 | | | | | | | 2 | 6 |
| DI | IRQCTL[7] ← 0 | | | 8F | – | – | – | – | – | – | 1 | 2 |
| DJNZ dst, RA | dst ← dst – 1<br>if dst ≠ 0<br>PC ← PC + X | r | | 0A–FA | – | – | – | – | – | – | 2 | 3 |
| EI | IRQCTL[7] ← 1 | | | 9F | – | – | – | – | – | – | 1 | 2 |
| HALT | HALT Mode | | | 7F | – | – | – | – | – | – | 1 | 2 |
| INC dst | dst ← dst + 1 | R | | 20 | – | * | * | * | – | – | 2 | 2 |
| | | IR | | 21 | | | | | | | 2 | 3 |
| | | r | | 0E–FE | | | | | | | 1 | 2 |
| INCW dst | dst ← dst + 1 | RR | | A0 | – | * | * | * | – | – | 2 | 5 |
| | | IRR | | A1 | | | | | | | 2 | 6 |
| IRET | FLAGS ← @SP<br>SP ← SP + 1<br>PC ← @SP<br>SP ← SP + 2<br>IRQCTL[7] ← 1 | | | BF | * | * | * | * | * | * | 1 | 5 |
| JP dst | PC ← dst | DA | | 8D | – | – | – | – | – | – | 3 | 2 |
| | | IRR | | C4 | | | | | | | 2 | 3 |
| JP cc, dst | if cc is true<br>PC ← dst | DA | | 0D–FD | – | – | – | – | – | – | 3 | 2 |
| JR dst | PC ← PC + X | DA | | 8B | – | – | – | – | – | – | 2 | 2 |
| JR cc, dst | if cc is true<br>PC ← PC + X | DA | | 0B–FB | – | – | – | – | – | – | 2 | 2 |

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

**Table 137. Op Code Map Abbreviations (Continued)**

| Abbreviation | Description | Abbreviation | Description |
|---|---|---|---|
| IM | Immediate data value | r2, R2, Ir2, Irr2, IR2, rr2, RR2, IRR2, ER2 | Source address |
| Ir | Indirect working register | RA | Relative |
| IR | Indirect register | rr | Working register pair |
| Irr | Indirect working register pair | RR | Register pair |

### Hex Address: FDB

**Table 240. 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: FDC

**Table 241. 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: FDD

**Table 242. 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: FDE

**Table 243. 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 | | | | | | | |