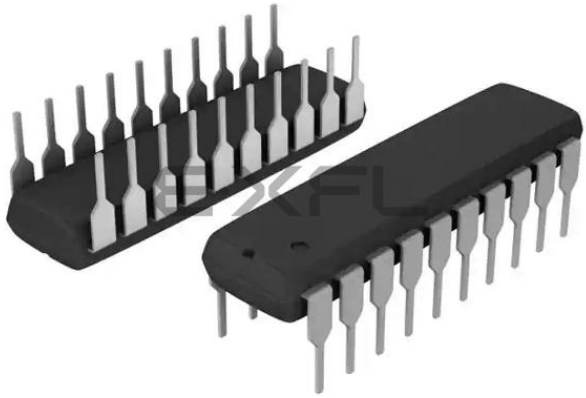

**Welcome to E-XFL.COM**

## 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 | 5MHz |
| Connectivity | IrDA, UART/USART |
| Peripherals | Brown-out Detect/Reset, LED, POR, PWM, WDT |
| Number of I/O | 16 |
| Program Memory Size | 2KB (2K x 8) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 512 x 8 |
| Voltage - Supply (Vcc/Vdd) | 2.7V ~ 3.6V |
| Data Converters | A/D 7x10b |
| Oscillator Type | Internal |
| Operating Temperature | 0°C ~ 70°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/z8f0223ph005sc |

# CPU and Peripheral Overview

## eZ8 CPU Features

The eZ8 CPU, Zilog's latest 8-bit central processing unit (CPU), meets the continuing demand for faster and 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.
- Software stack allows much greater depth in subroutine calls and interrupts than hardware stacks.
- Compatible with existing Z8 code.
- Expanded internal Register File allows access of up to 4 KB.
- New instructions improve execution efficiency for code developed using higher-level programming languages, including C.
- Pipelined instruction fetch and execution.
- New instructions for improved performance including BIT, BSWAP, BTJ, CPC, LDC, LDCI, LEA, MULT, and SRL.
- New instructions support 12-bit linear addressing of the Register file.
- Up to 10 MIPS operation.
- C-Compiler friendly.
- 2 to 9 clock cycles per instruction.

For more information on eZ8 CPU, refer to *eZ8 CPU Core User Manual (UM0128)* available for download at www.zilog.com.

## General-Purpose I/O

Z8 Encore! XP F0823 Series features 6 to 24 port pins (Ports A–C) for general-purpose I/O (GPIO). The number of GPIO pins available is a function of package. Each pin is individually programmable. 5 V tolerant input pins are available on all I/Os on 8-pin devices, most I/Os on other package types.

## Flash Controller

The Flash Controller programs and erases Flash memory. The Flash Controller supports protection against accidental program and erasure, as well as factory serialization and read protection.

**Table 8. Register File Address Map (Continued)**

| Address (Hex) | Register Description | Mnemonic | Reset (Hex) | Page No |
|---|---|---|---|---|
| F91–FBF | Reserved | — | XX | |
| **Interrupt Controller** | | | | |
| FC0 | Interrupt Request 0 | IRQ0 | 00 | 58 |
| FC1 | IRQ0 Enable High Bit | IRQ0ENH | 00 | 60 |
| FC2 | IRQ0 Enable Low Bit | IRQ0ENL | 00 | 61 |
| FC3 | Interrupt Request 1 | IRQ1 | 00 | 59 |
| FC4 | IRQ1 Enable High Bit | IRQ1ENH | 00 | 62 |
| FC5 | IRQ1 Enable Low Bit | IRQ1ENL | 00 | 62 |
| FC6 | Interrupt Request 2 | IRQ2 | 00 | 60 |
| FC7 | IRQ2 Enable High Bit | IRQ2ENH | 00 | 63 |
| FC8 | IRQ2 Enable Low Bit | IRQ2ENL | 00 | 63 |
| FC9–FCC | Reserved | — | XX | |
| FCD | Interrupt Edge Select | IRQES | 00 | 64 |
| FCE | Shared Interrupt Select | IRQSS | 00 | 64 |
| FCF | Interrupt Control | IRQCTL | 00 | 65 |
| **GPIO Port A** | | | | |
| FD0 | Port A Address | PAADDR | 00 | 43 |
| FD1 | Port A Control | PACTL | 00 | 45 |
| FD2 | Port A Input Data | PAIN | XX | 45 |
| FD3 | Port A Output Data | PAOUT | 00 | 45 |
| **GPIO Port B** | | | | |
| FD4 | Port B Address | PBADDR | 00 | 43 |
| FD5 | Port B Control | PBCTL | 00 | 45 |
| FD6 | Port B Input Data | PBIN | XX | 45 |
| FD7 | Port B Output Data | PBOUT | 00 | 45 |
| **GPIO Port C** | | | | |
| FD8 | Port C Address | PCADDR | 00 | 43 |
| FD9 | Port C Control | PCCTL | 00 | 45 |
| FDA | Port C Input Data | PCIN | XX | 45 |
| FDB | Port C Output Data | PCOUT | 00 | 45 |
| FDC–FEF | Reserved | — | XX | |
| **Watchdog Timer (WDT)** | | | | |
| FF0 | Reset Status | RSTSTAT | XX | 90 |
| | Watchdog Timer Control | WDTCTL | XX | 90 |
| FF1 | Watchdog Timer Reload Upper Byte | WDTU | FF | 91 |

**Note:** *This register is only reset during a Power-On Reset sequence. Other System Reset events do not affect it.*

**Table 13. Power Control Register 0 (PWRCTL0)**

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| FIELD | Reserved | Reserved | | VBO | Reserved | ADC | COMP | Reserved |
| RESET | 1 | 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 | F80H | | | | | | | |

Reserved—Must be 1

Reserved—Must be 0

VBO—Voltage Brownout Detector Disable
This bit and the VBO_AO Flash option bit must both enable the VBO for the VBO to be active.
0 = VBO enabled
1 = VBO disabled

ADC—Analog-to-Digital Converter Disable
0 = Analog-to-Digital Converter enabled
1 = Analog-to-Digital Converter disabled

COMP—Comparator Disable
0 = Comparator is enabled
1 = Comparator is disabled

Reserved—Must be 0

tions as a GPIO pin. If it is not present, the debug feature is disabled until/unless another reset event occurs. For more details, see On-Chip Debugger on page 151.

## Crystal Oscillator Override

For systems using a crystal oscillator, PA0 and PA1 are used to connect the crystal. When the crystal oscillator is enabled (see Oscillator Control Register Definitions on page 167), the GPIO settings are overridden and PA0 and PA1 are disabled.

## 5 V Tolerance

All six I/O pins on the 8-pin devices are 5 V-tolerant, unless the programmable pull-ups are enabled. If the pull-ups are enabled and inputs higher than $V_{DD}$ are applied to these parts, excessive current flows through those pull-up devices and can damage the chip.

**Note:** *In the 20- and 28-pin versions of this device, any pin which shares functionality with an ADC, crystal or comparator port is not 5 V-tolerant, including PA[1:0], PB[5:0], and PC[2:0]. All other signal pins are 5 V-tolerant, and can safely handle inputs higher than $V_{DD}$ even with the pull-ups enabled.*

## External Clock Setup

For systems using an external TTL drive, PB3 is the clock source for 20- and 28-pin devices. In this case, configure PB3 for alternate function CLKIN. Write the Oscillator Control Register (see Oscillator Control Register Definitions on page 167) such that the external oscillator is selected as the system clock. For 8-pin devices use PA1 instead of PB3.

**Table 15. Port Alternate Function Mapping (Non 8-Pin Parts)**

| Port | Pin | Mnemonic | Alternate Function Description | Alternate Function Set Register AFS1 |
|---|---|---|---|---|
| **Port A** | PA0 | T0IN/$\overline{\text{T0OUT}}$* | Timer 0 Input/Timer 0 Output Complement | N/A |
| | | Reserved | | |
| | PA1 | T0OUT | Timer 0 Output | |
| | | Reserved | | |
| | PA2 | DE0 | UART 0 Driver Enable | |
| | | Reserved | | |
| | PA3 | $\overline{\text{CTS0}}$ | UART 0 Clear to Send | |
| | | Reserved | | |
| | PA4 | RXD0/IRRX0 | UART 0 / IrDA 0 Receive Data | |
| | | Reserved | | |
| | PA5 | TXD0/IRTX0 | UART 0 / IrDA 0 Transmit Data | |
| | | Reserved | | |
| | PA6 | T1IN/$\overline{\text{T1OUT}}$* | Timer 1 Input/Timer 1 Output Complement | |
| | | Reserved | | |
| | PA7 | T1OUT | Timer 1 Output | |
| | | Reserved | | |

**Note:** Because there is only a single alternate function for each Port A pin, the Alternate Function Set registers are not implemented for Port A. Enabling alternate function selections as described in Port A–C Alternate Function Sub-Registers automatically enables the associated alternate function.

* Whether PA0/PA6 take on the timer input or timer output complement function depends on the timer configuration as described in Timer Pin Signal Operation on page 79.

0 = The drains are enabled for any output mode (unless overridden by the alternate function).
1 = The drain of the associated pin is disabled (open-drain mode).

### Port A–C High Drive Enable Sub-Registers

The Port A–C High Drive Enable sub-register (Table 23) is accessed through the Port A–C Control register by writing 04H to the Port A–C Address register. Setting the bits in the Port A–C High Drive Enable sub-registers to 1 configures the specified port pins for high current output drive operation. The Port A–C High Drive Enable sub-register affects the pins directly and, as a result, alternate functions are also affected.

**Table 23. Port A–C High Drive Enable Sub-Registers (PxHDE)**

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| FIELD | PHDE7 | PHDE6 | PHDE5 | PHDE4 | PHDE3 | PHDE2 | PHDE1 | PHDE0 |
| 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 04H in Port A–C Address Register, accessible through the Port A–C Control Register | | | | | | | |

PHDE[7:0]—Port High Drive Enabled.
0 = The Port pin is configured for standard output current drive.
1 = The Port pin is configured for high output current drive.

### Port A–C Stop Mode Recovery Source Enable Sub-Registers

The Port A–C Stop Mode Recovery Source Enable sub-register (Table 24) is accessed through the Port A–C Control register by writing 05H to the Port A–C Address register. Setting the bits in the Port A–C Stop Mode Recovery Source Enable sub-registers to 1 configures the specified Port pins as a Stop Mode Recovery source. During STOP mode, any logic transition on a Port pin enabled as a Stop Mode Recovery source initiates Stop Mode Recovery.

**Table 24. Port A–C Stop Mode Recovery Source Enable Sub-Registers (PxSMRE)**

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| FIELD | PSMRE7 | PSMRE6 | PSMRE5 | PSMRE4 | PSMRE3 | PSMRE2 | PSMRE1 | PSMRE0 |
| 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 05H in Port A–C Address Register, accessible through the Port A–C Control Register | | | | | | | |

**Table 26. Port A–C Alternate Function Set 1 Sub-Registers (PxAFS1)**

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| FIELD | PAFS17 | PAFS16 | PAFS15 | PAFS14 | PAFS13 | PAFS12 | PAFS11 | PAFS10 |
| RESET | 00H (all ports of 20/28 pin devices); 04H (Port A of 8-pin device) | | | | | | | |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | If 07H in Port A–C Address Register, accessible through the Port A–C Control Register | | | | | | | |

PAFS1[7:0]—Port Alternate Function Set to 1
0 = Port Alternate Function selected as defined in Table 14 (see GPIO Alternate Functions on page 36).
1 = Port Alternate Function selected as defined in Table 14 (see GPIO Alternate Functions on page 36).

### Port A–C Alternate Function Set 2 Sub-Registers

The Port A–C Alternate Function Set 2 sub-register (Table 27) is accessed through the Port A–C Control register by writing 08H to the Port A–C Address register. The Alternate Function Set 2 sub-registers selects the alternate function available at a port pin. Alternate Functions selected by setting or clearing bits of this register is defined in Table 14 in the section GPIO Alternate Functions on page 36.

**Table 27. Port A–C Alternate Function Set 2 Sub-Registers (PxAFS2)**

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| FIELD | PAFS27 | PAFS26 | PAFS25 | PAFS24 | PAFS23 | PAFS22 | PAFS21 | PAFS20 |
| RESET | 00H (all ports of 20/28 pin devices); 04H (Port A of 8-pin device) | | | | | | | |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | If 08H in Port A–C Address Register, accessible through the Port A–C Control Register | | | | | | | |

PAFS2[7:0]—Port Alternate Function Set 2
0 = Port Alternate Function selected as defined in Table 14 (see GPIO Alternate Functions on page 36).
1 = Port Alternate Function selected as defined in Table 14.

## Port A–C Input Data Registers

Reading from the Port A–C Input Data registers (Table 28) returns the sampled values from the corresponding port pins. The Port A–C Input Data registers are read-only. The value returned for any unused ports is 0. Unused ports include those missing on the 8- and 28-pin packages, as well as those missing on the ADC-enabled 28-pin packages.

### COMPARATOR COUNTER Mode

In COMPARATOR COUNTER mode, the timer counts input transitions from the analog comparator output. The TPOL bit in the Timer Control Register selects whether the count occurs on the rising edge or the falling edge of the comparator output signal. In COMPARATOR COUNTER mode, the prescaler is disabled.

**Caution:** *The frequency of the comparator output signal must not exceed one-fourth the system clock frequency.*

After reaching the Reload value stored in the Timer Reload High and Low Byte registers, 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) at timer Reload.

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

1. Write to the Timer Control register to:
   - Disable the timer.
   - Configure the timer for COMPARATOR COUNTER mode.
   - Select either the rising edge or falling edge of the comparator output signal for the count. This also sets the initial logic level (High or Low) for the Timer Output alternate function. However, the Timer Output function is not required to be enabled.
2. Write to the Timer High and Low Byte registers to set the starting count value. This action only affects the first pass in COMPARATOR COUNTER mode. After the first timer Reload in COMPARATOR COUNTER mode, counting always begins at the reset value of 0001H. Generally, in COMPARATOR COUNTER mode the Timer High and Low Byte registers must be written with the value 0001H.
3. Write to the Timer Reload High and Low Byte registers to set the Reload value.
4. If appropriate, 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.

In COMPARATOR COUNTER mode, the number of comparator output transitions since the timer start is given by the following equation:

Comparator Output Transitions = Current Count Value – Start Value

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
This field along with the TMODEHI bit in TxCTL0 register determines the operating mode of the timer. TMODEHI is the most significant bit of the Timer mode selection value.

0000 = ONE-SHOT mode
0001 = CONTINUOUS mode
0010 = COUNTER mode
0011 = PWM SINGLE OUTPUT mode
0100 = CAPTURE mode
0101 = COMPARE mode
0110 = GATED mode
0111 = CAPTURE/COMPARE mode
1000 = PWM DUAL OUTPUT mode
1001 = CAPTURE RESTART mode
1010 = COMPARATOR COUNTER Mode

### 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. The Transmit Data register can now be written with the next character to send. This action 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 occurs:

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

**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 cannot contain valid data and must be ignored. The BRKD bit indicates if the overrun was caused by a break condition on the line. After reading the status byte indicating an overrun error, the Receive Data register must be read again to clear the error bits is the UART Status 0 register. Updates to the Receive Data register occur only when the next data word is received.

### UART Data and Error Handling Procedure

Figure 15 displays the recommended procedure for use in UART receiver interrupt service routines.

Reserved—0 when read

FSTAT—Flash Controller Status
000000 = Flash Controller locked
000001 = First unlock command received (73H written)
000010 = Second unlock command received (8CH written)
000011 = Flash Controller unlocked

000100 = Sector protect register selected

001xxx = Program operation in progress
010xxx = Page erase operation in progress
100xxx = Mass erase operation in progress

## Flash Page Select Register

The Flash Page Select (FPS) register shares address space with the Flash Sector Protect Register. Unless the Flash controller is unlocked and written with 5EH, writes to this address target the Flash Page Select Register.

The register is used to select one of the eight available Flash memory pages to be programmed or erased. Each Flash Page contains 512 bytes of Flash memory. During a Page Erase operation, all Flash memory having addresses with the most significant 7-bits given by FPS[6:0] are chosen for program/erase operation.

**Table 81. Flash Page Select Register (FPS)**

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| FIELD | INFO_EN | PAGE | | | | | | |
| 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 | FF9H | | | | | | | |

INFO_EN—Information Area Enable

0 = Information Area us not selected

1 = Information Area is selected. The Information Area is mapped into the Program Memory address space at addresses FE00H through FFFFH.

PAGE—Page Select
This 7-bit field identifies the Flash memory page for Page Erase and page unlocking. Program Memory Address[15:9] = PAGE[6:0]. For the Z8F04x3 devices, the upper 4 bits must always be 0. For the Z8F02x3 devices, the upper 5 bits must always be 0. For the Z8F01x3 devices, the upper 6 bits must always be 0.

# On-Chip Debugger

Z8 Encore! XP® F0823 Series devices contain an integrated On-Chip Debugger (OCD) that provides advanced debugging features that include:

- Single pin interface
- Reading and writing of the register file
- Reading and writing of program and data memory
- Setting of breakpoints and watchpoints
- Executing eZ8 CPU instructions
- Debug pin sharing with general-purpose input-output function to maximize the pins available

## Architecture

The on-chip debugger consists of four primary functional blocks: transmitter, receiver, auto-baud detector/generator, and debug controller. Figure 22 displays the architecture of the OCD.

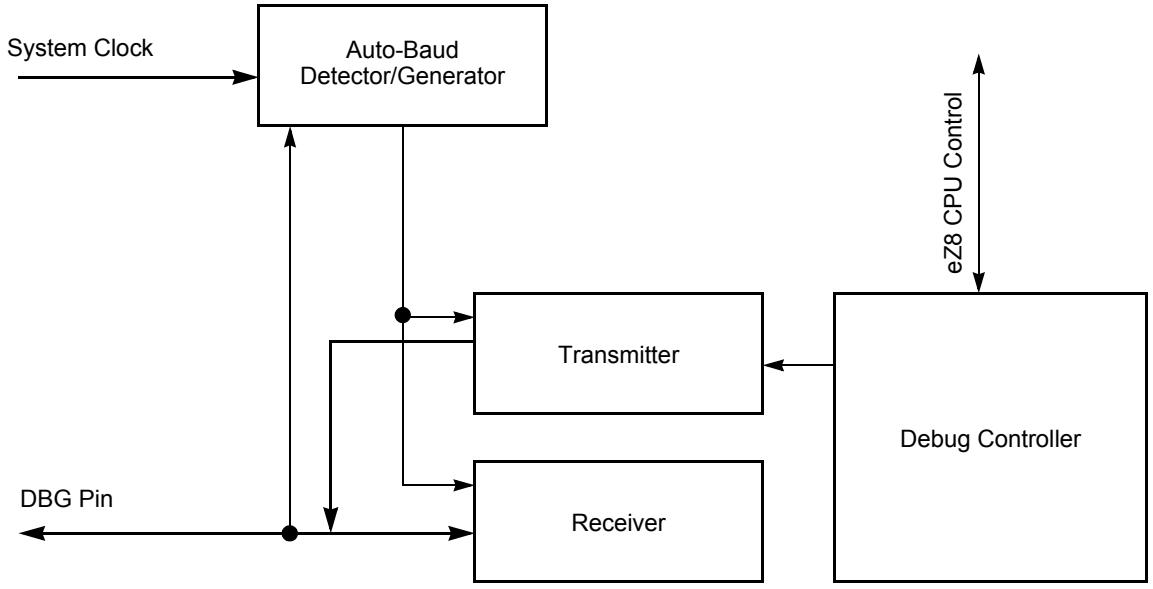


**Figure 22. On-Chip Debugger Block Diagram**

# Oscillator Control

Z8 Encore! XP® F0823 Series devices uses three possible clocking schemes, each user-selectable:

- On-chip precision trimmed RC oscillator
- External clock drive
- On-chip low power Watchdog Timer oscillator

In addition, Z8 Encore! XP F0823 Series devices contain clock failure detection and recovery circuitry, allowing continued operation despite a failure of the primary oscillator.

## Operation

This chapter discusses the logic used to select the system clock and handle primary oscillator failures. A description of the specific operation of each oscillator is outlined elsewhere in this document.

### System Clock Selection

The oscillator control block selects from the available clocks. Table 101 details each clock source and its usage.

**Table 101. Oscillator Configuration and Selection**

| Clock Source | Characteristics | Required Setup |
|---|---|---|
| Internal Precision RC Oscillator | • 32.8 kHz or 5.53 MHz<br>• ± 4% accuracy when trimmed<br>• No external components required | • Unlock and write Oscillator Control Register (OSCCTL) to enable and select oscillator at either 5.53 MHz or 32.8 kHz |
| External Clock Drive | • 0 to 20 MHz<br>• Accuracy dependent on external clock source | • Write GPIO registers to configure PB3 pin for external clock function<br>• Unlock and write OSCCTL to select external system clock<br>• Apply external clock signal to GPIO |
| Internal Watchdog Timer Oscillator | • 10 kHz nominal<br>• ± 40% accuracy; no external components required<br>• Very Low power consumption | • Enable WDT if not enabled and wait until WDT Oscillator is operating.<br>• Unlock and write Oscillator Control Register (OSCCTL) to enable and select oscillator |

**Table 114. Rotate and Shift Instructions (Continued)**

| Mnemonic | Operands | Instruction |
|---|---|---|
| SRA | dst | Shift Right Arithmetic |
| SRL | dst | Shift Right Logical |
| SWAP | dst | Swap Nibbles |

# eZ8 CPU Instruction Summary

Table 115 summarizes the eZ8 CPU instructions. The table identifies the addressing modes employed by the instruction, the effect upon the Flags register, the number of CPU clock cycles required for the instruction fetch, and the number of CPU clock cycles required for the instruction execution.

**Table 115. eZ8 CPU Instruction Summary**

| Assembly Mnemonic | Symbolic Operation | Address Mode dst | Address Mode src | Opcode(s) (Hex) | Flags C | Z | S | V | D | H | Fetch Cycles | Instr. Cycles |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ADC dst, src | dst ← dst + src + C | r | r | 12 | * | * | * | * | 0 | * | 2 | 3 |
| | | r | Ir | 13 | | | | | | | 2 | 4 |
| | | R | R | 14 | | | | | | | 3 | 3 |
| | | R | IR | 15 | | | | | | | 3 | 4 |
| | | R | IM | 16 | | | | | | | 3 | 3 |
| | | IR | IM | 17 | | | | | | | 3 | 4 |
| ADCX dst, src | dst ← dst + src + C | ER | ER | 18 | * | * | * | * | 0 | * | 4 | 3 |
| | | ER | IM | 19 | | | | | | | 4 | 3 |
| ADD dst, src | dst ← dst + src | r | r | 02 | * | * | * | * | 0 | * | 2 | 3 |
| | | r | Ir | 03 | | | | | | | 2 | 4 |
| | | R | R | 04 | | | | | | | 3 | 3 |
| | | R | IR | 05 | | | | | | | 3 | 4 |
| | | R | IM | 06 | | | | | | | 3 | 3 |
| | | IR | IM | 07 | | | | | | | 3 | 4 |
| ADDX dst, src | dst ← dst + src | ER | ER | 08 | * | * | * | * | 0 | * | 4 | 3 |
| | | ER | IM | 09 | | | | | | | 4 | 3 |
| Flags Notation: | * = Value is a function of the result of the operation.<br>– = Unaffected<br>X = Undefined | | | | 0 = Reset to 0<br>1 = Set to 1 | | | | | | | |

| Upper Nibble (Hex) \ Lower Nibble (Hex) | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1.1 **BRK** | 2.2 **SRP** IM | 2.3 **ADD** r1,r2 | 2.4 **ADD** r1,Ir2 | 3.3 **ADD** R2,R1 | 3.4 **ADD** IR2,R1 | 3.3 **ADD** R1,IM | 3.4 **ADD** IR1,IM | 4.3 **ADDX** ER2,ER1 | 4.3 **ADDX** IM,ER1 | 2.3 **DJNZ** r1,X | 2.2 **JR** cc,X | 2.2 **LD** r1,IM | 3.2 **JP** cc,DA | 1.2 **INC** r1 | 1.2 **NOP** |
| 1 | 2.2 **RLC** R1 | 2.3 **RLC** IR1 | 2.3 **ADC** r1,r2 | 2.4 **ADC** r1,Ir2 | 3.3 **ADC** R2,R1 | 3.4 **ADC** IR2,R1 | 3.3 **ADC** R1,IM | 3.4 **ADC** IR1,IM | 4.3 **ADCX** ER2,ER1 | 4.3 **ADCX** IM,ER1 | | | | | | See 2nd Opcode Map |
| 2 | 2.2 **INC** R1 | 2.3 **INC** IR1 | 2.3 **SUB** r1,r2 | 2.4 **SUB** r1,Ir2 | 3.3 **SUB** R2,R1 | 3.4 **SUB** IR2,R1 | 3.3 **SUB** R1,IM | 3.4 **SUB** IR1,IM | 4.3 **SUBX** ER2,ER1 | 4.3 **SUBX** IM,ER1 | | | | | | 1, 2 ATM |
| 3 | 2.2 **DEC** R1 | 2.3 **DEC** IR1 | 2.3 **SBC** r1,r2 | 2.4 **SBC** r1,Ir2 | 3.3 **SBC** R2,R1 | 3.4 **SBC** IR2,R1 | 3.3 **SBC** R1,IM | 3.4 **SBC** IR1,IM | 4.3 **SBCX** ER2,ER1 | 4.3 **SBCX** IM,ER1 | | | | | | |
| 4 | 2.2 **DA** R1 | 2.3 **DA** IR1 | 2.3 **OR** r1,r2 | 2.4 **OR** r1,Ir2 | 3.3 **OR** R2,R1 | 3.4 **OR** IR2,R1 | 3.3 **OR** R1,IM | 3.4 **OR** IR1,IM | 4.3 **ORX** ER2,ER1 | 4.3 **ORX** IM,ER1 | | | | | | |
| 5 | 2.2 **POP** R1 | 2.3 **POP** IR1 | 2.3 **AND** r1,r2 | 2.4 **AND** r1,Ir2 | 3.3 **AND** R2,R1 | 3.4 **AND** IR2,R1 | 3.3 **AND** R1,IM | 3.4 **AND** IR1,IM | 4.3 **ANDX** ER2,ER1 | 4.3 **ANDX** IM,ER1 | | | | | | 1.2 **WDT** |
| 6 | 2.2 **COM** R1 | 2.3 **COM** IR1 | 2.3 **TCM** r1,r2 | 2.4 **TCM** r1,Ir2 | 3.3 **TCM** R2,R1 | 3.4 **TCM** IR2,R1 | 3.3 **TCM** R1,IM | 3.4 **TCM** IR1,IM | 4.3 **TCMX** ER2,ER1 | 4.3 **TCMX** IM,ER1 | | | | | | 1.2 **STOP** |
| 7 | 2.2 **PUSH** R2 | 2.3 **PUSH** IR2 | 2.3 **TM** r1,r2 | 2.4 **TM** r1,Ir2 | 3.3 **TM** R2,R1 | 3.4 **TM** IR2,R1 | 3.3 **TM** R1,IM | 3.4 **TM** IR1,IM | 4.3 **TMX** ER2,ER1 | 4.3 **TMX** IM,ER1 | | | | | | 1.2 **HALT** |
| 8 | 2.5 **DECW** RR1 | 2.6 **DECW** IRR1 | 2.5 **LDE** r1,Irr2 | 2.9 **LDEI** Ir1,Irr2 | 3.2 **LDX** r1,ER2 | 3.3 **LDX** Ir1,ER2 | 3.4 **LDX** IRR2,R1 | 3.5 **LDX** IRR2,IR1 | 3.4 **LDX** r1,rr2,X | 3.4 **LDX** rr1,r2,X | | | | | | 1.2 **DI** |
| 9 | 2.2 **RL** R1 | 2.3 **RL** IR1 | 2.5 **LDE** r2,Irr1 | 2.9 **LDEI** Ir2,Irr1 | 3.2 **LDX** r2,ER1 | 3.3 **LDX** Ir2,ER1 | 3.4 **LDX** R2,IRR1 | 3.5 **LDX** IR2,IRR1 | 3.3 **LEA** r1,r2,X | 3.5 **LEA** rr1,rr2,X | | | | | | 1.2 **EI** |
| A | 2.5 **INCW** RR1 | 2.6 **INCW** IRR1 | 2.3 **CP** r1,r2 | 2.4 **CP** r1,Ir2 | 3.3 **CP** R2,R1 | 3.4 **CP** IR2,R1 | 3.3 **CP** R1,IM | 3.4 **CP** IR1,IM | 4.3 **CPX** ER2,ER1 | 4.3 **CPX** IM,ER1 | | | | | | 1.4 **RET** |
| B | 2.2 **CLR** R1 | 2.3 **CLR** IR1 | 2.3 **XOR** r1,r2 | 2.4 **XOR** r1,Ir2 | 3.3 **XOR** R2,R1 | 3.4 **XOR** IR2,R1 | 3.3 **XOR** R1,IM | 3.4 **XOR** IR1,IM | 4.3 **XORX** ER2,ER1 | 4.3 **XORX** IM,ER1 | | | | | | 1.5 **IRET** |
| C | 2.2 **RRC** R1 | 2.3 **RRC** IR1 | 2.5 **LDC** r1,Irr2 | 2.9 **LDCI** Ir1,Irr2 | 2.3 **JP** IRR1 | 2.9 **LDC** Ir1,Irr2 | | 3.4 **LD** r1,r2,X | 3.2 **PUSHX** ER2 | | | | | | | 1.2 **RCF** |
| D | 2.2 **SRA** R1 | 2.3 **SRA** IR1 | 2.5 **LDC** r2,Irr1 | 2.9 **LDCI** Ir2,Irr1 | 2.6 **CALL** IRR1 | 2.2 **BSWAP** R1 | 3.3 **CALL** DA | 3.4 **LD** r2,r1,X | 3.2 **POPX** ER1 | | | | | | | 1.2 **SCF** |
| E | 2.2 **RR** R1 | 2.3 **RR** IR1 | 2.2 **BIT** p,b,r1 | 2.3 **LD** r1,Ir2 | 3.2 **LD** R2,R1 | 3.3 **LD** IR2,R1 | 3.2 **LD** R1,IM | 3.3 **LD** IR1,IM | 4.2 **LDX** ER2,ER1 | 4.2 **LDX** IM,ER1 | | | | | | 1.2 **CCF** |
| F | 2.2 **SWAP** R1 | 2.3 **SWAP** IR1 | 2.6 **TRAP** Vector | 2.3 **LD** Ir1,r2 | 2.8 **MULT** RR1 | 3.3 **LD** R2,IR1 | 3.3 **BTJ** p,b,r1,X | 3.4 **BTJ** p,b,Ir1,X | | | | | | | | |

**Figure 27. First Opcode Map**

**Table 125. Analog-to-Digital Converter Electrical Characteristics and Timing**

| Symbol | Parameter | $V_{DD}$ = 3.0 V to 3.6 V<br>$T_A$ = 0 °C to +70 °C<br>(unless otherwise stated) | | | Units | Conditions |
|---|---|---|---|---|---|---|
| | | Minimum | Typical | Maximum | | |
| | Resolution | 10 | | – | bits | |
| | Differential Nonlinearity (DNL) | -1.0 | – | 1.0 | LSB[3] | External $V_{REF}$ = 2.0 V; $R_S \leftarrow$ 3.0 kΩ |
| | Integral Nonlinearity (INL) | -3.0 | – | 3.0 | LSB[3] | External $V_{REF}$ = 2.0 V; $R_S \leftarrow$ 3.0 kΩ |
| | Offset Error with Calibration | | ±1 | | LSB[3] | |
| | Absolute Accuracy with Calibration | | ±3 | | LSB[3] | |
| $V_{REF}$ | Internal Reference Voltage | 1.0<br>2.0 | 1.1<br>2.2 | 1.2<br>2.4 | V | REFSEL=01<br>REFSEL=10 |
| $V_{REF}$ | Internal Reference Variation with Temperature | | ±1.0 | | % | Temperature variation with $V_{DD}$ = 3.0 |
| $V_{REF}$ | Internal Reference Voltage Variation with $V_{DD}$ | | ±0.5 | | % | Supply voltage variation with $T_A$ = 30 °C |
| $R_{REFOUT}$ | Reference Buffer Output Impedance | | 850 | | Ω | When the internal reference is buffered and driven out to the VREF pin (REFOUT = 1) |
| | Single-Shot Conversion Time | – | 5129 | – | System clock cycles | All measurements but temperature sensor |
| | | | 10258 | | | Temperature sensor measurement |
| | Continuous Conversion Time | – | 256 | – | System clock cycles | All measurements but temperature sensor |
| | | | 512 | | | Temperature sensor measurement |
| | Signal Input Bandwidth | – | 10 | | kHz | As defined by -3 dB point |
| $R_S$ | Analog Source Impedance[4] | – | – | 10 | kΩ | In unbuffered mode |
| | | | | | | |

## D



## E



## F

## P



## R