

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	4KB (4K 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	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/z8f0413ph005sg

Table of Contents

Revision History	iii
List of Figures	x
List of Tables	xii
Overview	1
Features	1
Part Selection Guide	2
Block Diagram	3
CPU and Peripheral Overview	4
General-Purpose I/O	4
Flash Controller	4
Internal Precision Oscillator	5
10-Bit Analog-to-Digital Converter	5
Analog Comparator	5
Universal Asynchronous Receiver/Transmitter	5
Timers	5
Interrupt Controller	5
Reset Controller	5
On-Chip Debugger	6
Pin Description	7
Available Packages	7
Pin Configurations	7
Signal Descriptions	9
Pin Characteristics	11
Address Space	13
Register File	13
Program Memory	13
Data Memory	15
Flash Information Area	15
Register Map	16
Reset and Stop Mode Recovery	21
Reset Types	21
Reset Sources	23
Power-On Reset	23
Voltage Brown-Out Reset	24
Watchdog Timer Reset	25

Overview

Zilog's Z8 Encore! XP microcontroller unit (MCU) family of products are the first Zilog microcontroller products based on the 8-bit eZ8 CPU core. Z8 Encore! XP F0823 Series products expand upon 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 Z8 Encore! XP F0823 Series makes it suitable for a variety of applications including motor control, security systems, home appliances, personal electronic devices, and sensors.

Features

The key features of Z8 Encore! XP F0823 Series include:

- 5MHz eZ8 CPU
- 1 KB, 2KB, 4KB, or 8KB Flash memory with in-circuit programming capability
- 256B, 512B, or 1 KB register RAM
- 6 to 24 I/O pins depending upon package
- Internal precision oscillator (IPO)
- Full-duplex UART
- The universal asynchronous receiver/transmitter (UART) baud rate generator (BRG) can be configured and used as a basic 16-bit timer
- Infrared data association (IrDA)-compliant infrared encoder/decoders, integrated with UART
- Two enhanced 16-bit timers with capture, compare, and PWM capability
- Watchdog Timer (WDT) with dedicated internal RC oscillator
- On-Chip Debugger (OCD)
- Optional 8-channel, 10-bit Analog-to-Digital Converter (ADC)
- On-Chip analog comparator
- Up to 20 vectored interrupts
- Direct LED drive with programmable drive strengths
- Voltage Brown-Out (VBO) protection
- Power-On Reset (POR)

Low-Power Modes

Z8 Encore! XP F0823 Series products contain power-saving features. The highest level of power reduction is provided by the STOP Mode, in which nearly all device functions are powered down. The next lower level of power reduction is provided by the HALT Mode, in which the CPU is powered down.

Further power savings can be implemented by disabling individual peripheral blocks while in ACTIVE mode (defined as being in neither STOP nor HALT Mode).

STOP Mode

Executing the eZ8 CPU's Stop instruction places the device into STOP Mode, powering down all peripherals except the Voltage Brown-Out detector, and the Watchdog Timer. These two blocks may also be disabled for additional power savings. In STOP Mode, the operating characteristics are:

- Primary crystal oscillator and internal precision oscillator are stopped; X_{IN} and X_{OUT} (if previously enabled) are disabled, and PA0/PA1 revert to the states programmed by the GPIO registers
- System clock is stopped
- eZ8 CPU is stopped
- Program counter (PC) stops incrementing
- Watchdog Timer's internal RC oscillator continues to operate if enabled by the Oscillator Control Register
- If enabled, the Watchdog Timer logic continues to operate
- If enabled for operation in STOP Mode by the associated Flash Option Bit, the Voltage Brown-Out protection circuit continues to operate
- All other on-chip peripherals are idle

To minimize current in STOP Mode, all GPIO pins that are configured as digital inputs must be driven to one of the supply rails (V_{CC} or GND). Additionally, any GPIOs configured as outputs must also be driven to one of the supply rails. The device can be brought out of STOP Mode using Stop Mode Recovery. For more information about Stop Mode Recovery, see the [Reset and Stop Mode Recovery](#) chapter on page 21.

Interrupt Controller

The interrupt controller on the Z8 Encore! XP F0823 Series products prioritizes the interrupt requests from the on-chip peripherals and the GPIO port pins. The features of interrupt controller include:

- 20 unique interrupt vectors
 - 12 GPIO port pin interrupt sources (two are shared)
 - 8 on-chip peripheral interrupt sources (two are shared)
- Flexible GPIO interrupts
 - Eight selectable rising and falling edge GPIO interrupts
 - Four dual-edge interrupts
- Three levels of individually programmable interrupt priority
- Watchdog Timer can be configured to generate an interrupt

Interrupt requests (IRQs) allow peripheral devices to suspend CPU operation in an orderly manner and force the CPU to start an interrupt service routine (ISR). Usually this interrupt service routine is involved with the exchange of data, status information, or control information between the CPU and the interrupting peripheral. When the service routine is completed, the CPU returns to the operation from which it was interrupted.

The eZ8 CPU supports both vectored and polled interrupt handling. For polled interrupts, the interrupt controller has no effect on operation. For more information about interrupt servicing by the eZ8 CPU, refer to the [eZ8 CPU Core User Manual \(UM0128\)](#) available for download at www.zilog.com.

Interrupt Vector Listing

Table 35 lists all of the interrupts available in order of priority. The interrupt vector is stored with the most-significant byte (MSB) at the even Program Memory address and the least-significant byte (LSB) at the following odd Program Memory address.

► **Note:** Some port interrupts are not available on the 8- and 20-pin packages. The ADC interrupt is unavailable on devices not containing an ADC.

Interrupt Request 2 Register

The Interrupt Request 2 (IRQ2) register (Table 38) stores interrupt requests for both vectored and polled interrupts. When a request is presented to the interrupt controller, the corresponding bit in the IRQ2 Register becomes 1. If interrupts are globally enabled (vectored interrupts), the interrupt controller passes an interrupt request to the eZ8 CPU. If interrupts are globally disabled (polled interrupts), the eZ8 CPU can read the Interrupt Request 2 Register to determine if any interrupt requests are pending.

Table 38. Interrupt Request 2 Register (IRQ2)

Bit	7	6	5	4	3	2	1	0
Field	Reserved				PC3I	PC2I	PC1I	PC0I
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
Address	FC6H							

Bit	Description
[7:4]	Reserved These bits are reserved and must be programmed to 0000.
[3:0] PCxI	Port C Pin x Interrupt Request 0 = No interrupt request is pending for GPIO Port C pin x. 1 = An interrupt request from GPIO Port C pin x is awaiting service.

Note: x indicates the specific GPIO Port C pin number (3–0).

IRQ0 Enable High and Low Bit Registers

Table 39 describes the priority control for IRQ0. The IRQ0 Enable High and Low Bit registers (Table 40 and Table 41) form a priority-encoded enabling for interrupts in the Interrupt Request 0 Register. Priority is generated by setting bits in each register.

Table 39. IRQ0 Enable and Priority Encoding

IRQ0ENH[x]	IRQ0ENL[x]	Priority	Description
0	0	Disabled	Disabled
0	1	Level 1	Low
1	0	Level 2	Nominal
1	1	Level 3	High

Note: where x indicates the register bits from 0–7.

input capture and reload events. If appropriate, configure the timer interrupt to be generated only at the input capture event or the reload event by setting TICONFIG field of the TxCTL1 Register.

6. Configure the associated GPIO port pin for the Timer Input alternate function.
7. Write to the Timer Control Register to enable the timer and initiate counting.

In CAPTURE Mode, the elapsed time from timer start to capture event can be calculated using the following equation:

$$\text{Capture Elapsed Time (s)} = \frac{(\text{Capture Value} - \text{Start Value}) \times \text{Prescale}}{\text{System Clock Frequency (Hz)}}$$

CAPTURE RESTART Mode

In CAPTURE RESTART Mode, the current timer count value is recorded when the acceptable 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 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 count value in the Timer High and Low Byte registers is reset to 0001H and counting resumes. The INPCAP bit in TxCTL1 Register is set to indicate the timer interrupt is because of an input capture event.

If no capture event occurs, the timer counts up to the 16-bit compare value stored in the Timer Reload High and Low Byte registers. Upon reaching the reload value, the timer generates an interrupt, the count value in the Timer High and Low Byte registers is reset to 0001H and counting resumes. The INPCAP bit in TxCTL1 Register is cleared to indicate the timer interrupt is not caused by an input capture event.

Observe the following steps to configure a timer for CAPTURE RESTART Mode and initiating the count:

1. Write to the Timer Control Register to:
 - Disable the timer
 - Configure the timer for CAPTURE RESTART Mode; setting the mode also involves writing to TMODEHI bit in TxCTL1 Register
 - 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.

- Set or clear CTSE to enable or disable control from the remote receiver using the CTS pin.

8. Execute an EI instruction to enable interrupts.

The UART is now configured for interrupt-driven data transmission. Because the UART Transmit Data Register is empty, an interrupt is generated immediately. When the UART Transmit interrupt is detected, the associated interrupt service routine (ISR) performs the following:

1. Write the UART Control 1 Register to select the multiprocessor bit for the byte to be transmitted:
 Set the Multiprocessor Bit Transmitter (MPBT) if sending an address byte, clear it if sending a data byte.
2. Write the data byte to the UART Transmit Data Register. The transmitter automatically transfers the data to the Transmit Shift register and transmits the data.
3. Clear the UART Transmit interrupt bit in the applicable Interrupt Request register.
4. Execute the IRET instruction to return from the interrupt-service routine and wait for the Transmit Data Register to again become empty.

Receiving Data Using the Polled Method

Observe the following steps to configure the UART for polled data reception:

1. Write to the UART Baud Rate High and Low Byte registers to set an acceptable baud rate for the incoming data stream.
2. Enable the UART pin functions by configuring the associated GPIO port pins for alternate function operation.
3. Write to the UART Control 1 Register to enable MULTIPROCESSOR Mode functions, if appropriate.
4. Write to the UART Control 0 Register to:
 - Set the receive enable bit (REN) to enable the UART for data reception
 - Enable parity, if appropriate and if Multiprocessor mode is not enabled, and select either even or odd parity
5. Check the RDA bit in the UART Status 0 Register to determine if the Receive Data Register contains a valid data byte (indicated by a 1). If RDA is set to 1 to indicate available data, continue to [Step 6](#). If the Receive Data Register is empty (indicated by a 0), continue to monitor the RDA bit awaiting reception of the valid data.
6. Read 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].

$$\left(\frac{1}{\text{Baud Rate (Hz)}}\right) \leq \text{DE to Start Bit Setup Time (s)} \leq \left(\frac{2}{\text{Baud Rate (Hz)}}\right)$$

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

UART Address Compare Register

The UART Address Compare Register stores the multinode network address of the UART. When the MPMD[1] bit of UART Control Register 0 is set, all incoming address bytes are compared to the value stored in the Address Compare Register. Receive interrupts and RDA assertions only occur in the event of a match.

Table 70. UART Address Compare Register (U0ADDR)

Bit	7	6	5	4	3	2	1	0
Field	COMP_ADDR							
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
Address	F45H							

Bit **Description**

[7:0] **Compare Address**

COMP_ADDR This 8-bit value is compared to incoming address bytes.

UART Baud Rate High and Low Byte Registers

The UART Baud Rate High and Low Byte registers (Table 71 and Table 72) combine to create a 16-bit baud rate divisor value (BRG[15:0]) that sets the data transmission rate (baud rate) of the UART.

Table 71. UART Baud Rate High Byte Register (U0BRH)

Bit	7	6	5	4	3	2	1	0
Field	BRH							
RESET	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Address	F46H							

Table 72. UART Baud Rate Low Byte Register (U0BRL)

Bit	7	6	5	4	3	2	1	0
Field	BRL							
RESET	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Address	F47H							

The window remains open until the count again reaches 8 (that is, 24 baud clock periods since the previous pulse was detected), giving the endec a sampling window of minus four baud rate clocks to plus eight baud rate clocks around the expected time of an incoming pulse. If an incoming pulse is detected inside this window this process is repeated. If the incoming data is a logical 1 (no pulse), the endec returns to the initial state and waits for the next falling edge. As each falling edge is detected, the endec clock counter is reset, resynchronizing the endec to the incoming signal, allowing the endec to tolerate jitter and baud rate errors in the incoming datastream. Resynchronizing the endec does not alter the operation of the UART, which ultimately receives the data. The UART is only synchronized to the incoming data stream when a Start bit is received.

Infrared Encoder/Decoder Control Register Definitions

All infrared endec configuration and status information is set by the UART control registers as defined in the Universal Asynchronous Receiver/Transmitter chapter on page 97.

! **Caution:** To prevent spurious signals during IrDA data transmission, set the IREN bit in the UART Control 1 Register to 1 to enable the endec before enabling the GPIO port alternate function for the corresponding pin.

Table 74. ADC Control Register 0 (ADCCTL0)

Bit	7	6	5	4	3	2	1	0
Field	CEN	REFSELL	REFEXT	CONT	ANAIN[3:0]			
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
Address	F70H							

Bit	Description
[7] CEN	<p>Conversion Enable</p> <p>0 = Conversion is complete. Writing a 0 produces no effect. The ADC automatically clears this bit to 0 when a conversion is complete.</p> <p>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.</p>
[6] REFSELL	<p>Voltage Reference Level Select Low Bit</p> <p>In conjunction with the High bit (REFSELH) in ADC Control/Status Register 1, this determines the level of the internal voltage reference; the following details the effects of {REFSELH, REFSELL}. This reference is independent of the Comparator reference.</p> <p>00 = Internal Reference Disabled, reference comes from external pin.</p> <p>01 = Internal Reference set to 1.0V.</p> <p>10 = Internal Reference set to 2.0V (default).</p>
[5] REFEXT	<p>External Reference Select</p> <p>0 = External reference buffer is disabled; V_{REF} pin is available for GPIO functions.</p> <p>1 = The internal ADC reference is buffered and connected to the V_{REF} pin.</p>
[4] CONT	<p>Continuous Conversion</p> <p>0 = Single-shot conversion. ADC data is output once at completion of the 5129 system clock cycles.</p> <p>1 = Continuous conversion. ADC data updated every 256 system clock cycles.</p>

Trim Bit Data Register

The Trim Bid Data (TRMDR) register contains the read or write data for access to the trim option bits.

Table 88. Trim Bit Data Register (TRMDR)

Bit	7	6	5	4	3	2	1	0
Field	TRMDR: Trim Bit Data							
RESET	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Address	FF7H							

Flash Option Bit Address Space

The first two bytes of Flash program memory at addresses 0000H and 0001H are reserved for the user-programmable Flash option bits.

Table 89. Flash Option Bits at Program Memory Address 0000H

Bit	7	6	5	4	3	2	1	0
Field	WDT_RES	WDT_AO	Reserved		VBO_AO	FRP	Reserved	FWP
RESET	U	U	U	U	U	U	U	U
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Address	Program Memory 0000H							
Note: U = Unchanged by Reset. R/W = Read/Write.								

Bit	Description
[7] WDT_RES	Watchdog Timer Reset 0 = Watchdog Timer time-out generates an interrupt request. Interrupts must be globally enabled for the eZ8 CPU to acknowledge the interrupt request. 1 = Watchdog Timer time-out causes a system reset. This setting is the default for unprogrammed (erased) Flash.
[6] WDT_AO	Watchdog Timer Always ON 0 = Watchdog Timer is automatically enabled upon application of system power. Watchdog Timer can not be disabled. 1 = Watchdog Timer is enabled upon execution of the WDT instruction. Once enabled, the Watchdog Timer can only be disabled by a Reset or Stop Mode Recovery. This setting is the default for unprogrammed (erased) Flash.
[5:4]	Reserved These bits are reserved and must be programmed to 11 during writes, and to 11 when read.

On-Chip Debugger

Z8 Encore! XP F0823 Series devices contain an integrated On-Chip Debugger (OCD) which 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.

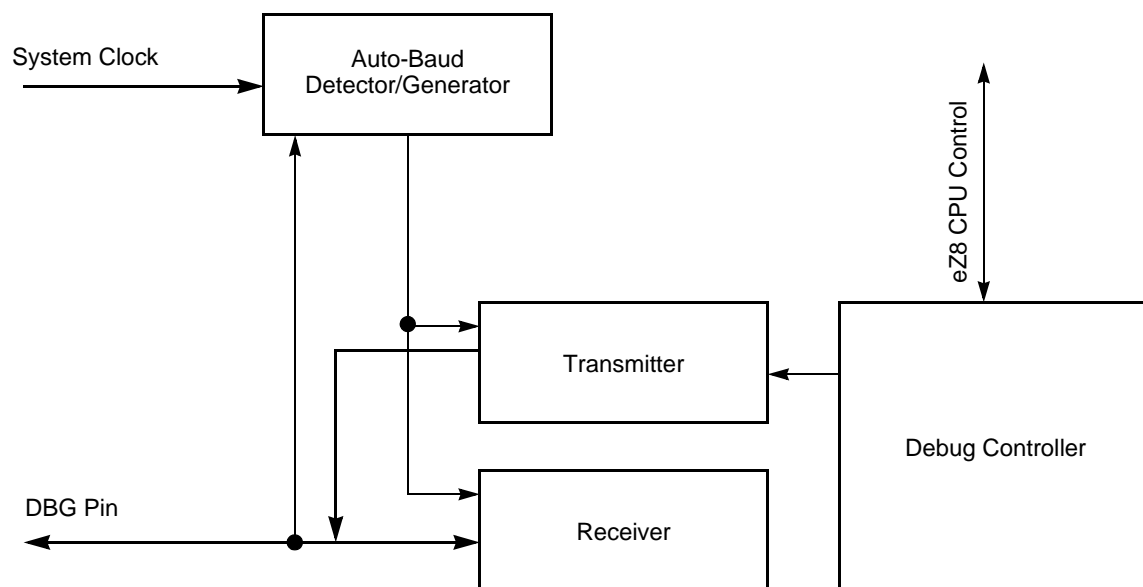


Figure 22. On-Chip Debugger Block Diagram

Table 105. Oscillator Control Register (OSCCTL)

Bit	7	6	5	4	3	2	1	0
Field	INTEN	Reserved	WDTEN	POFEN	WDFEN	SCKSEL		
RESET	1	0	1	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Address	F86H							

Bit	Description
[7] INTEN	Internal Precision Oscillator Enable 1 = Internal precision oscillator is enabled. 0 = Internal precision oscillator is disabled.
[6]	Reserved This bit is reserved and must be programmed to 0 during writes and to 0 when read.
[5] WDTEN	Watchdog Timer Oscillator Enable 1 = Watchdog Timer oscillator is enabled. 0 = Watchdog Timer oscillator is disabled.
[4] POFEN	Primary Oscillator Failure Detection Enable 1 = Failure detection and recovery of primary oscillator is enabled. 0 = Failure detection and recovery of primary oscillator is disabled.
[3] WDFEN	Watchdog Timer Oscillator Failure Detection Enable 1 = Failure detection of Watchdog Timer oscillator is enabled. 0 = Failure detection of Watchdog Timer oscillator is disabled.
[2:0] SCKSEL	System Clock Oscillator Select 000 = Internal precision oscillator functions as system clock at 5.53MHz. 001 = Internal precision oscillator functions as system clock at 32kHz. 010 = Reserved. 011 = Watchdog Timer oscillator functions as system clock. 100 = External clock signal on PB3 functions as system clock. 101 = Reserved. 110 = Reserved. 111 = Reserved.

Internal Precision Oscillator

The internal precision oscillator (IPO) is designed for use without external components. You can either manually trim the oscillator for a non-standard frequency or use the automatic factory-trimmed version to achieve a 5.53MHz frequency. The features of IPO include:

- On-chip RC oscillator that does not require external components
- Output frequency of either 5.53MHz or 32.8kHz (contains both a fast and a slow mode)
- Trimming possible through Flash option bits with user override
- Elimination of crystals or ceramic resonators in applications where high timing accuracy is not required

Operation

An 8-bit trimming register, incorporated into the design, compensates for absolute variation of oscillator frequency. Once trimmed the oscillator frequency is stable and does not require subsequent calibration. Trimming is performed during manufacturing and is not necessary for you to repeat unless a frequency other than 5.53MHz (fast mode) or 32.8kHz (slow mode) is required. This trimming is done at +30°C and a supply voltage of 3.3 V, so accuracy of this operating point is optimal.

Power down this block for minimum system power. By default, the oscillator is configured through the Flash Option bits. However, the user code can override these trim values, as described in the [Trim Bit Address Space](#) section on page 151.

Select one of the two frequencies for the oscillator: 5.53MHz and 32.8kHz, using the OSCSEL bits in the [Oscillator Control](#) chapter on page 169.

Assembly Language Source Program Example

```
JP START      ; Everything after the semicolon is a comment.
START:        ; A label called 'START'. The first instruction (JP START) in this
              ; example causes program execution to jump to the point within the
              ; program where the START label occurs.

LD R4, R7     ; A Load (LD) instruction with two operands. The first operand,
              ; Working Register R4, is the destination. The second operand,
              ; Working Register R7, is the source. The contents of R7 is
              ; written into R4.

LD 234H, #01  ; Another Load (LD) instruction with two operands.
              ; The first operand, Extended Mode Register Address 234H,
              ; identifies the destination. The second operand, Immediate Data
              ; value 01H, is the source. The value 01H is written into the
              ; Register at address 234H.
```

Assembly Language Syntax

For proper instruction execution, eZ8 CPU assembly language syntax requires that the operands be written as 'destination, source'. After assembly, the object code usually has the operands in the order 'source, destination', but ordering is opcode-dependent. The following instruction examples illustrate the format of some basic assembly instructions and the resulting object code produced by the assembler. You must follow this binary format if you prefer manual program coding or intend to implement your own assembler.

Example 1

If the contents of registers 43H and 08H are added and the result is stored in 43H, the assembly syntax and resulting object code is shown in Table 106.


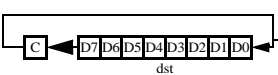
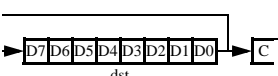
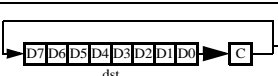
Table 106. Assembly Language Syntax Example 1

Assembly Language Code	ADD	43H,	08H	(ADD dst, src)
Object Code	04	08	43	(OPC src, dst)

Example 2

In general, when an instruction format requires an 8-bit register address, that address can specify any register location in the range 0–255 or, using Escaped Mode Addressing, a Working Register R0–R15. If the contents of Register 43H and Working Register R8 are added and the result is stored in 43H, the assembly syntax and resulting object code is shown in Table 107.

Table 118. eZ8 CPU Instruction Summary (Continued)

Assembly Mnemonic	Symbolic Operation	Address Mode		Opcode(s) (Hex)	Flags						Fetch Cycles	Instr. Cycles
		dst	src		C	Z	S	V	D	H		
POPX dst	dst ← @SP SP ← SP + 1	ER		D8	–	–	–	–	–	–	3	2
PUSH src	SP ← SP – 1 @SP ← src	R		70	–	–	–	–	–	–	2	2
		IR		71							2	3
		IM		IF70							3	2
PUSHX src	SP ← SP – 1 @SP ← src	ER		C8	–	–	–	–	–	–	3	2
RCF	C ← 0			CF	0	–	–	–	–	–	1	2
RET	PC ← @SP SP ← SP + 2			AF	–	–	–	–	–	–	1	4
RL dst		R		90	*	*	*	*	–	–	2	2
		IR		91							2	3
RLC dst		R		10	*	*	*	*	–	–	2	2
		IR		11							2	3
RR dst		R		E0	*	*	*	*	–	–	2	2
		IR		E1							2	3
RRC dst		R		C0	*	*	*	*	–	–	2	2
		IR		C1							2	3

Note: Flags Notation:

* = Value is a function of the result of the operation.

– = Unaffected.

X = Undefined.

0 = Reset to 0.

1 = Set to 1.

Index

Numerics

10-bit ADC 4

A

absolute maximum ratings 196

AC characteristics 200

ADC 178

architecture 121

block diagram 122

continuous conversion 124

control register 126, 129

control register definitions 126

data high byte register 130

data low bits register 131

electrical characteristics and timing 203

operation 122

single-shot conversion 123

ADCCTL register 126, 129

ADCDH register 130

ADC DL register 131

ADCX 178

ADD 178

add - extended addressing 178

add with carry 178

add with carry - extended addressing 178

additional symbols 177

address space 13

ADDX 178

analog signals 10

analog-to-digital converter (ADC) 121

AND 181

ANDX 181

arithmetic instructions 178

assembly language programming 174

assembly language syntax 175

B

B 177

b 176

baud rate generator, UART 108

BCLR 179

binary number suffix 177

BIT 179

bit 176

clear 179

manipulation instructions 179

set 179

set or clear 179

swap 179

test and jump 181

test and jump if non-zero 181

test and jump if zero 181

bit jump and test if non-zero 181

bit swap 181

block diagram 3

block transfer instructions 179

BRK 181

BSET 179

BSWAP 179, 181

BTJ 181

BTJNZ 181

BTJZ 181

C

CALL procedure 181

CAPTURE mode 89

CAPTURE/COMPARE mode 89

cc 176

CCF 180

characteristics, electrical 196

clear 180

CLR 180

COM 181

COMPARE 89

compare - extended addressing 178

COMPARE mode 89

compare with carry 178

compare with carry - extended addressing 178
 complement 181
 complement carry flag 179, 180
 condition code 176
 continuous conversion (ADC) 124
 CONTINUOUS mode 88
 control register definition, UART 108
 Control Registers 13, 16
 COUNTER modes 89
 CP 178
 CPC 178
 CPCX 178
 CPU and peripheral overview 4
 CPU control instructions 180
 CPX 178
 Customer Support 230

D

DA 176, 178
 data memory 15
 DC characteristics 197
 debugger, on-chip 156
 DEC 178
 decimal adjust 178
 decrement 178
 decrement and jump non-zero 181
 decrement word 178
 DECW 178
 destination operand 177
 device, port availability 33
 DI 180
 direct address 176
 disable interrupts 180
 DJNZ 181
 dst 177

E

EI 180
 electrical characteristics 196
 ADC 203
 flash memory and timing 202
 GPIO input data sample timing 204

Watchdog Timer 202, 204
 enable interrupt 180
 ER 176
 extended addressing register 176
 external pin reset 25
 eZ8 CPU features 4
 eZ8 CPU instruction classes 178
 eZ8 CPU instruction notation 176
 eZ8 CPU instruction set 174
 eZ8 CPU instruction summary 182

F

FCTL register 141, 148, 149
 features, Z8 Encore! 1
 first opcode map 194
 FLAGS 177
 flags register 177
 flash
 controller 4
 option bit address space 149
 option bit configuration - reset 146
 program memory address 0000H 149
 program memory address 0001H 150
 flash memory 134
 arrangement 135
 byte programming 139
 code protection 137
 configurations 134
 control register definitions 141, 148
 controller bypass 140
 electrical characteristics and timing 202
 flash control register 141, 148, 149
 flash option bits 138
 flash status register 142
 flow chart 136
 frequency high and low byte registers 144
 mass erase 139
 operation 135
 operation timing 137
 page erase 139
 page select register 142, 144
 FPS register 142, 144
 FSTAT register 142

- ONE-SHOT mode 70, 88
- operating mode 70
- PWM mode 75, 76, 89
- reading the timer count values 83
- reload high and low byte registers 84
- timer control register definitions 83
- timer output signal operation 83
- timers 0-3
 - control registers 86, 87
 - high and low byte registers 83, 86
- TM 179
- TMX 179
- tools, hardware and software 220
- transmit
 - IrDA data 118
- transmitting UART data-pollled method 99
- transmitting UART dat-interrupt-driven method 100
- TRAP 181

U

- UART 4
 - architecture 97
 - baud rate generator 108
 - control register definitions 108
 - controller signals 9
 - interrupts 105
 - MULTIPROCESSOR mode 103
 - receiving data using interrupt-driven method 102
 - receiving data using the polled method 101
 - transmitting data using the interrupt-driven method 100
 - transmitting data using the polled method 99
 - x baud rate high and low registers 115
 - x control 0 and control 1 registers 112
 - x status 0 and status 1 registers 110, 111
- UxBRH register 115
- UxBRL register 115
- UxCTL0 register 112, 115
- UxCTL1 register 113
- UxRXD register 109
- UxSTAT0 register 110

- UxSTAT1 register 111
- UxTXD register 109

V

- vector 177
- Voltage Brownout reset (VBR) 24

W

- Watchdog Timer
 - approximate time-out delay 91
 - CNTL 24
 - control register 94, 171
 - electrical characteristics and timing 202, 204
 - interrupt in normal operation 92
 - interrupt in STOP mode 92
 - refresh 92, 180
 - reload unlock sequence 93
 - reload upper, high and low registers 94
 - reset 25
 - reset in normal operation 93
 - reset in STOP mode 93
 - time-out response 92
- Watchdog Timer Control Register (WDTCTL) 94
- WDTCTL register 94, 133, 172
- WDTH register 95
- WDTL register 95
- WDTU register 95
- working register 176
- working register pair 177

X

- X 177
- XOR 181
- XORX 181

Z

- Z8 Encore!
 - block diagram 3
 - features 1