



Welcome to E-XFL.COM

What is "[Embedded - Microcontrollers](#)"?

"[Embedded - Microcontrollers](#)" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "[Embedded - Microcontrollers](#)"

Details

Product Status	Obsolete
Core Processor	eZ8
Core Size	8-Bit
Speed	20MHz
Connectivity	IrDA, UART/USART
Peripherals	Brown-out Detect/Reset, LED, LVD, POR, PWM, Temp Sensor, WDT
Number of I/O	23
Program Memory Size	4KB (4K x 8)
Program Memory Type	FLASH
EEPROM Size	128 x 8
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 3.6V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Through Hole
Package / Case	28-DIP (0.600", 15.24mm)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f042apj020sc

Hardware Overflow	123
Automatic Powerdown	123
Single-Shot Conversion	123
Continuous Conversion	124
Interrupts	125
Calibration and Compensation	125
ADC Compensation Details	127
Input Buffer Stage	129
ADC Control Register Definitions	130
ADC Control Register 0	130
ADC Control/Status Register 1	132
ADC Data High Byte Register	132
ADC Data Low Byte Register	133
Low Power Operational Amplifier	134
Overview	134
Operation	134
Comparator	135
Operation	135
Comparator Control Register Definitions	136
Temperature Sensor	139
Temperature Sensor Operation	139
Flash Memory	141
Architecture	141
Flash Information Area	142
Operation	143
Flash Operation Timing Using the Flash Frequency Registers	145
Flash Code Protection Against External Access	145
Flash Code Protection Against Accidental Program and Erasure	145
Byte Programming	147
Page Erase	147
Mass Erase	147
Flash Controller Bypass	148
Flash Controller Behavior in DEBUG Mode	148
Flash Control Register Definitions	149
Flash Control Register	149
Flash Status Register	150
Flash Page Select Register	150

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 more code-efficient microcontrollers. The eZ8 CPU executes a superset of the original Z8[®] instruction set. The features of eZ8 CPU 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.

10-Bit Analog-to-Digital Converter

The optional analog-to-digital converter (ADC) converts an analog input signal to a 10-bit binary number. The ADC accepts inputs from eight different analog input pins in both single-ended and differential modes. The ADC also features a unity gain buffer when high input impedance is required.

U0RXI—UART 0 Receiver Interrupt Request

0 = No interrupt request is pending for the UART 0 receiver.

1 = An interrupt request from the UART 0 receiver is awaiting service.

U0TXI—UART 0 Transmitter Interrupt Request

0 = No interrupt request is pending for the UART 0 transmitter.

1 = An interrupt request from the UART 0 transmitter is awaiting service.

ADCI—ADC Interrupt Request

0 = No interrupt request is pending for the analog-to-digital Converter.

1 = An interrupt request from the Analog-to-Digital Converter is awaiting service.

Interrupt Request 1 Register

The Interrupt Request 1 (IRQ1) register ([Table 34](#)) stores interrupt requests for both vectored and polled interrupts. When a request is presented to the interrupt controller, the corresponding bit in the IRQ1 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 1 register to determine if any interrupt requests are pending.

Table 34. Interrupt Request 1 Register (IRQ1)

BITS	7	6	5	4	3	2	1	0
FIELD	PA7VI	PA6CI	PA5I	PA4I	PA3I	PA2I	PA1I	PA0I
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	FC3H							

PA7VI—Port A Pin 7 or LVD Interrupt Request

0 = No interrupt request is pending for GPIO Port A or LVD.

1 = An interrupt request from GPIO Port A or LVD.

PA6CI—Port A Pin 6 or Comparator Interrupt Request

0 = No interrupt request is pending for GPIO Port A or Comparator.

1 = An interrupt request from GPIO Port A or Comparator.

PAxI—Port A Pin x Interrupt Request

0 = No interrupt request is pending for GPIO Port A pin x.

1 = An interrupt request from GPIO Port A pin x is awaiting service.

where x indicates the specific GPIO Port pin number (0–5).

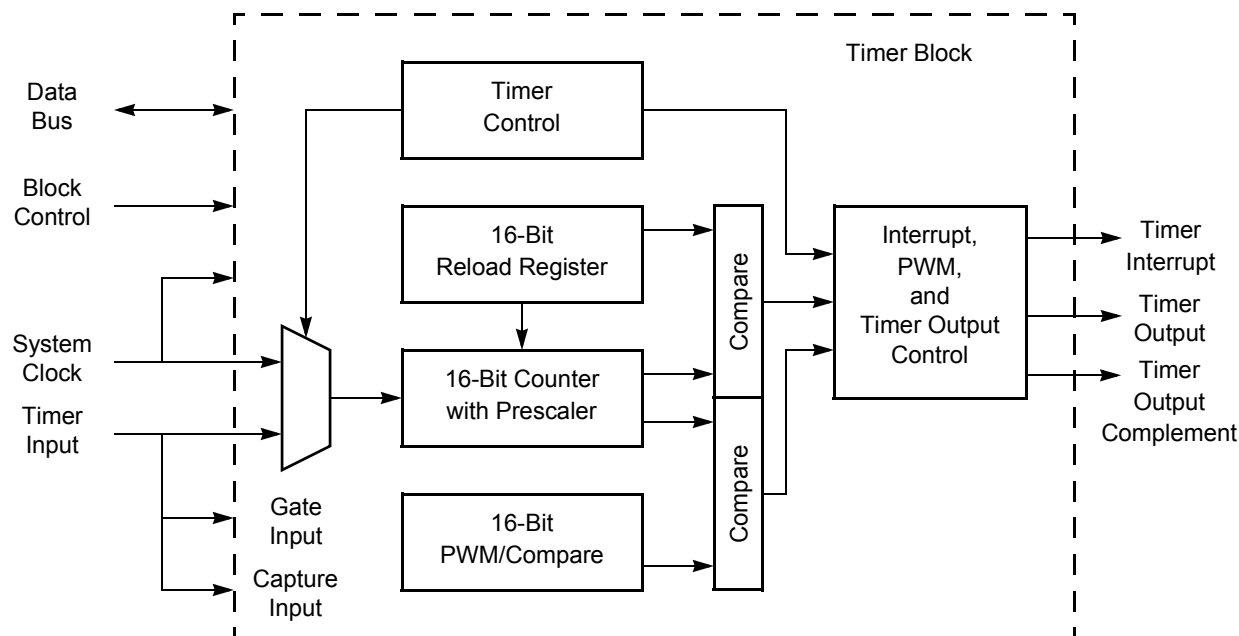


Figure 9. Timer Block Diagram

Operation

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

Timer Operating Modes

The timers can be configured to operate in the following modes:

ONE-SHOT Mode

In ONE-SHOT mode, the timer counts up to the 16-bit Reload value stored in the Timer Reload High and Low byte registers. The timer input is the system clock. Upon reaching the Reload value, the timer generates an interrupt and the count value in the Timer High and Low Byte registers is reset to 0001H. The timer is automatically disabled and stops counting.

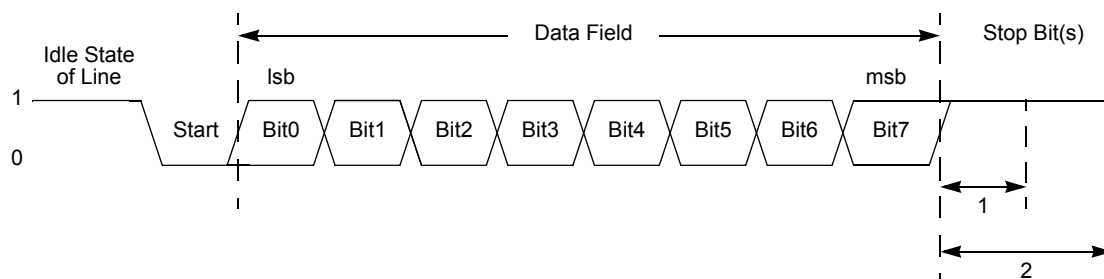


Figure 11. UART Asynchronous Data Format without Parity

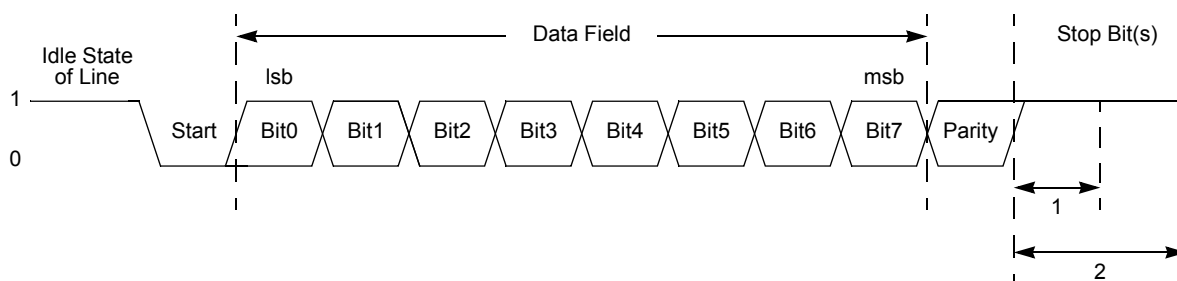


Figure 12. UART Asynchronous Data Format with Parity

Transmitting Data using the Polled Method

Follow the steps below to transmit data using the polled method of operation:

1. Write to the UART Baud Rate High and Low Byte registers to set the required baud rate.
2. Enable the UART pin functions by configuring the associated GPIO Port pins for alternate function operation.
3. Write to the UART Control 1 register, if MULTIPROCESSOR mode is appropriate, to enable MULTIPROCESSOR (9-bit) mode functions.
4. Set the Multiprocessor Mode Select (MPEN) bit to enable MULTIPROCESSOR mode.
5. Write to the UART Control 0 register to:
 - Set the transmit enable bit (TEN) to enable the UART for data transmission.
 - Set the parity enable bit (PEN), if parity is appropriate and MULTIPROCESSOR mode is not enabled, and select either even or odd parity (PSEL).
 - Set or clear the CTSE bit to enable or disable control from the remote receiver using the $\overline{\text{CTS}}$ pin.

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:
2. Set the Multiprocessor Bit Transmitter (MPBT) if sending an address byte, clear it if sending a data byte.
3. 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.
4. Clear the UART Transmit interrupt bit in the applicable Interrupt Request register.
5. 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

Follow the steps below 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 5](#). 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].
7. Return to [Step 4](#) to receive additional data.

Compensation Steps:

1. Correct for Offset

ADC MSB	ADC LSB
---------	---------

-

Offset MSB	Offset LSB
------------	------------

=

#1 MSB	#1 LSB
--------	--------

2. Take absolute value of the offset corrected ADC value *if negative*—the gain correction factor is computed assuming positive numbers, with sign restoration afterward.

#2 MSB	#2 LSB
--------	--------

Also take absolute value of the gain correction word *if negative*.

AGain MSB	AGain LSB
-----------	-----------

3. Multiply by Gain Correction Word. If in DIFFERENTIAL mode, there are two gain correction values: one for positive ADC values, another for negative ADC values. Based on the sign of #2, use the appropriate Gain Correction Word.

#2 MSB	#2 LSB
--------	--------

*

AGain MSB	AGain LSB
-----------	-----------

=

#3	#3	#3	#3
----	----	----	----

4. Round the result and discard the least significant two bytes (this is equivalent to dividing by 2^{16}).

#3	#3	#3	#3
----	----	----	----

-

0x00	0x00	0x80	0x00
------	------	------	------

=

#4 MSB	#4 LSB
--------	--------

5. Determine sign of the gain correction factor using the sign bits from [Step 2](#). If the offset corrected ADC value AND the gain correction word have the same sign, then the factor is positive and is left unchanged. If they have differing signs, then the factor is negative and must be multiplied by -1.

Table 73. ADC Data High Byte Register (ADCD_H)

BITS	7	6	5	4	3	2	1	0
FIELD	ADCDH							
RESET	X	X	X	X	X	X	X	X
R/W	R	R	R	R	R	R	R	R
ADDR	F72H							
X = Undefined.								

ADCDH—ADC Data High Byte

This byte contains the upper eight bits of the ADC output. These bits are not valid during a single-shot conversion. During a continuous conversion, the most recent conversion output is held in this register. These bits are undefined after a Reset.

ADC Data Low Byte Register

The ADC Data Low Byte (ADCD_L) register contains the lower bits of the ADC output as well as an overflow status bit. The output is a 13-bit two's complement value. During a single-shot conversion, this value is invalid. Access to the ADC Data Low Byte register is read-only. Reading the ADC Data High Byte register latches data in the ADC Low Bits register.

Table 74. ADC Data Low Byte Register (ADCD_L)

BITS	7	6	5	4	3	2	1	0
FIELD	ADCDL					Reserved		OVF
RESET	X	X	X	X	X	X	X	X
R/W	R	R	R	R	R	R	R	R
ADDR	F73H							
X = Undefined.								

ADCDL—ADC Data Low Bits

These bits are the least significant five bits of the 13-bits of the ADC output. These bits are undefined after a Reset.

Reserved—Must be undefined.

OVF—Overflow Status

0= A hardware overflow did not occur in the ADC for the current sample.

1= A hardware overflow did occur in the ADC for the current sample, therefore the current sample is invalid.

Randomized Lot Identification Bits

As an optional feature, Zilog is able to provide a factory-programmed random lot identifier. With this feature, all devices in a given production lot are programmed with the same random number. This random number is uniquely regenerated for each successive production lot and is not likely to be repeated.

The randomized lot identifier is a 32 byte binary value, stored in the Flash information page (see [Reading the Flash Information Page](#) on page 155 and [Randomized Lot Identifier](#) on page 166 for more details) and is unaffected by mass erasure of the device's Flash memory.

Reading the Flash Information Page

The following code example shows how to read data from the Flash information area.

```
; get value at info address 60 (FE60h)

ldx FPS, #%80 ; enable access to flash info page

ld R0, #%FE
ld R1, #%60
ldc R2, @RR0 ; R2 now contains the calibration value
```

Flash Option Bit Control Register Definitions

Trim Bit Address Register

The Trim Bit Address (TRMADR) register contains the target address for an access to the trim option bits ([Table 84](#)).

Table 84. Trim Bit Address Register (TRMADR)

BITS	7	6	5	4	3	2	1	0
FIELD	TRMADR - Trim Bit Address (00H to 1FH)							
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	FF6H							

Temperature Sensor Calibration Data

Table 95. Temperature Sensor Calibration High Byte at 003A (TSCALH)

BITS	7	6	5	4	3	2	1	0
FIELD	TSCALH							
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
ADDR	Information Page Memory 003A							
Note: U = Unchanged by Reset. R/W = Read/Write.								

TSCALH – Temperature Sensor Calibration High Byte

The TSCALH and TSCALL bytes combine to form the 12-bit temperature sensor offset calibration value. For more details, see [Temperature Sensor Operation](#) on page 139.

Table 96. Temperature Sensor Calibration Low Byte at 003B (TSCALL)

BITS	7	6	5	4	3	2	1	0
FIELD	TSCALL							
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
ADDR	Information Page Memory 003B							
Note: U = Unchanged by Reset. R/W = Read/Write.								

TSCALL – Temperature Sensor Calibration Low Byte

The TSCALH and TSCALL bytes combine to form the 12-bit temperature sensor offset calibration value. For usage details, see [Temperature Sensor Operation](#) on page 139.

Watchdog Timer Calibration Data

Table 97. Watchdog Calibration High Byte at 007EH (WDTCALH)

BITS	7	6	5	4	3	2	1	0
FIELD	WDTCALH							
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
ADDR	Information Page Memory 007EH							
Note: U = Unchanged by Reset. R/W = Read/Write.								

Non-Volatile Data Storage

The Z8 Encore! XP[®] F082A Series devices contain a non-volatile data storage (NVDS) element of up to 128 bytes. This memory can perform over 100,000 write cycles.

Operation

The NVDS is implemented by special purpose Zilog[®] software stored in areas of program memory, which are not user-accessible. These special-purpose routines use the Flash memory to store the data. The routines incorporate a dynamic addressing scheme to maximize the write/erase endurance of the Flash.

► **Note:** *Different members of the Z8 Encore! XP F082A Series feature multiple NVDS array sizes. See [Z8 Encore! XP[®] F082A Series Family Part Selection Guide](#) on page 3 for details. Also the members containing 8 KB of Flash memory do not include the NVDS feature.*

NVDS Code Interface

Two routines are required to access the NVDS: a write routine and a read routine. Both of these routines are accessed with a CALL instruction to a pre-defined address outside of the user-accessible program memory. Both the NVDS address and data are single-byte values. Because these routines disturb the working register set, user code must ensure that any required working register values are preserved by pushing them onto the stack or by changing the working register pointer just prior to NVDS execution.

During both read and write accesses to the NVDS, interrupt service is NOT disabled. Any interrupts that occur during the NVDS execution must take care not to disturb the working register and existing stack contents or else the array may become corrupted. Disabling interrupts before executing NVDS operations is recommended.

Use of the NVDS requires 15 bytes of available stack space. Also, the contents of the working register set are overwritten.

For correct NVDS operation, the Flash Frequency Registers must be programmed based on the system clock frequency (see [Flash Operation Timing Using the Flash Frequency Registers](#) on page 145).

Byte Write

To write a byte to the NVDS array, the user code must first push the address, then the data byte onto the stack. The user code issues a CALL instruction to the address of the byte-write routine (0x10B3). At the return from the sub-routine, the write status byte

Operation

OCD Interface

The on-chip debugger uses the DBG pin for communication with an external host. This one-pin interface is a bi-directional, open-drain interface that transmits and receives data. Data transmission is half-duplex, in that transmit and receive cannot occur simultaneously. The serial data on the DBG pin is sent using the standard asynchronous data format defined in RS-232. This pin creates an interface from the Z8 Encore! XP[®] F082A Series products to the serial port of a host PC using minimal external hardware. Two different methods for connecting the DBG pin to an RS-232 interface are displayed in [Figure 24](#) and [Figure 25](#). The recommended method is the buffered implementation displayed in [Figure 25](#). The DBG pin has a internal pull-up resistor which is sufficient for some applications (for more details on the pull-up current, see [Electrical Characteristics](#) on page 221). For OCD operation at higher data rates or in noisy systems, an external pull-up resistor is recommended.

**Caution:**

For operation of the on-chip debugger, all power pins (V_{DD} and AV_{DD}) must be supplied with power, and all ground pins (V_{SS} and AV_{SS}) must be properly grounded. The DBG pin is open-drain and may require an external pull-up resistor to ensure proper operation.

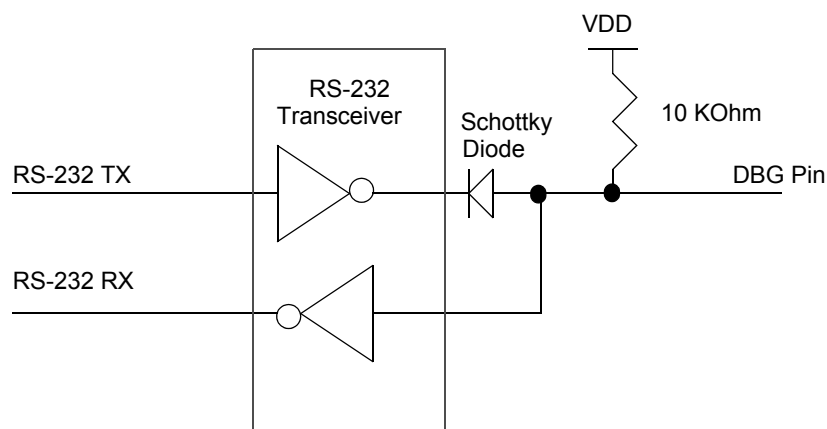


Figure 24. Interfacing the On-Chip Debugger's DBG Pin with an RS-232 Interface (1)

Write Memory, Read Memory, Write Register, Read Register, Read Memory CRC, Step Instruction, Stuff Instruction, and Execute Instruction commands.

```
DBG ← 03H
DBG → RuntimeCounter[15:8]
DBG → RuntimeCounter[7:0]
```

- **Write OCD Control Register (04H)**—The Write OCD Control Register command writes the data that follows to the OCDCTL register. When the Flash Read Protect Option Bit is enabled, the DBGMODE bit (OCDCTL[7]) can only be set to 1, it cannot be cleared to 0 and the only method of returning the device to normal operating mode is to reset the device.

```
DBG ← 04H
DBG ← OCDCTL[7:0]
```

- **Read OCD Control Register (05H)**—The Read OCD Control Register command reads the value of the OCDCTL register.

```
DBG ← 05H
DBG → OCDCTL[7:0]
```

- **Write Program Counter (06H)**—The Write Program Counter command writes the data that follows to the eZ8 CPU's Program Counter (PC). If the device is not in DEBUG mode or if the Flash Read Protect Option bit is enabled, the Program Counter (PC) values are discarded.

```
DBG ← 06H
DBG ← ProgramCounter[15:8]
DBG ← ProgramCounter[7:0]
```

- **Read Program Counter (07H)**—The Read Program Counter command reads the value in the eZ8 CPU's Program Counter (PC). If the device is not in DEBUG mode or if the Flash Read Protect Option bit is enabled, this command returns FFFFH.

```
DBG ← 07H
DBG → ProgramCounter[15:8]
DBG → ProgramCounter[7:0]
```

- **Write Register (08H)**—The Write Register command writes data to the Register File. Data can be written 1–256 bytes at a time (256 bytes can be written by setting size to 0). If the device is not in DEBUG mode, the address and data values are discarded. If the Flash Read Protect Option bit is enabled, only writes to the Flash Control Registers are allowed and all other register write data values are discarded.

```
DBG ← 08H
DBG ← {4'h0, Register Address[11:8]}
DBG ← Register Address[7:0]
DBG ← Size[7:0]
DBG ← 1-256 data bytes
```

Table 108. Oscillator Configuration and Selection

Clock Source	Characteristics	Required Setup
Internal Precision RC Oscillator	<ul style="list-style-type: none"> • 32.8 kHz or 5.53 MHz • High accuracy • No external components required 	<ul style="list-style-type: none"> • Unlock and write Oscillator Control Register (OSCCTL) to enable and select oscillator at either 5.53 MHz or 32.8 kHz
External Crystal/Resonator	<ul style="list-style-type: none"> • 32 kHz to 20 MHz • Very high accuracy (dependent on crystal or resonator used) • Requires external components 	<ul style="list-style-type: none"> • Configure Flash option bits for correct external oscillator mode • Unlock and write OSCCTL to enable crystal oscillator, wait for it to stabilize and select as system clock (if the XTLDIS option bit has been de-asserted, no waiting is required)
External RC Oscillator	<ul style="list-style-type: none"> • 32 kHz to 4 MHz • Accuracy dependent on external components 	<ul style="list-style-type: none"> • Configure Flash option bits for correct external oscillator mode • Unlock and write OSCCTL to enable crystal oscillator and select as system clock
External Clock Drive	<ul style="list-style-type: none"> • 0 to 20 MHz • Accuracy dependent on external clock source 	<ul style="list-style-type: none"> • Write GPIO registers to configure PB3 pin for external clock function • Unlock and write OSCCTL to select external system clock • Apply external clock signal to GPIO
Internal Watchdog Timer Oscillator	<ul style="list-style-type: none"> • 10 kHz nominal • Low accuracy; no external components required • Very low power consumption 	<ul style="list-style-type: none"> • Enable WDT if not enabled and wait until WDT Oscillator is operating. • Unlock and write Oscillator Control Register (OSCCTL) to enable and select oscillator



Caution: *Unintentional accesses to the oscillator control register can actually stop the chip by switching to a non-functioning oscillator. To prevent this condition, the oscillator control block employs a register unlocking/locking scheme.*

OSC Control Register Unlocking/Locking

To write the oscillator control register, unlock it by making two writes to the OSCCTL register with the values E7H followed by 18H. A third write to the OSCCTL register changes the value of the actual register and returns the register to a locked state. Any other sequence of oscillator control register writes has no effect. The values written to unlock the register must be ordered correctly, but are not necessarily consecutive. It is possible to write to or read from other registers within the unlocking/locking operation.

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

SCKSEL—System Clock Oscillator Select

000 = Internal precision oscillator functions as system clock at 5.53 MHz

001 = Internal precision oscillator functions as system clock at 32 kHz

010 = Crystal oscillator or external RC oscillator functions as system clock

011 = Watchdog Timer oscillator functions as system

100 = External clock signal on PB3 functions as system clock

101 = Reserved

110 = Reserved

111 = Reserved

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. This binary format must be followed if manual program coding is preferred or if you 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:

Table 112. 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:

Table 113. Assembly Language Syntax Example 2

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

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

eZ8 CPU Instruction Notation

In the eZ8 CPU Instruction Summary and Description sections, the operands, condition codes, status flags, and address modes are represented by a notational shorthand that is described in [Table 114](#).

Table 121. Logical Instructions (Continued)

Mnemonic	Operands	Instruction
ORX	dst, src	Logical OR using Extended Addressing
XOR	dst, src	Logical Exclusive OR
XORX	dst, src	Logical Exclusive OR using Extended Addressing

Table 122. Program Control Instructions

Mnemonic	Operands	Instruction
BRK	—	On-Chip Debugger Break
BTJ	p, bit, src, DA	Bit Test and Jump
BTJNZ	bit, src, DA	Bit Test and Jump if Non-Zero
BTJZ	bit, src, DA	Bit Test and Jump if Zero
CALL	dst	Call Procedure
DJNZ	dst, src, RA	Decrement and Jump Non-Zero
IRET	—	Interrupt Return
JP	dst	Jump
JP cc	dst	Jump Conditional
JR	DA	Jump Relative
JR cc	DA	Jump Relative Conditional
RET	—	Return
TRAP	vector	Software Trap

Table 123. Rotate and Shift Instructions

Mnemonic	Operands	Instruction
BSWAP	dst	Bit Swap
RL	dst	Rotate Left
RLC	dst	Rotate Left through Carry
RR	dst	Rotate Right
RRC	dst	Rotate Right through Carry

General Purpose I/O Port Output Timing

Figure 35 and Table 140 provide timing information for GPIO Port pins.

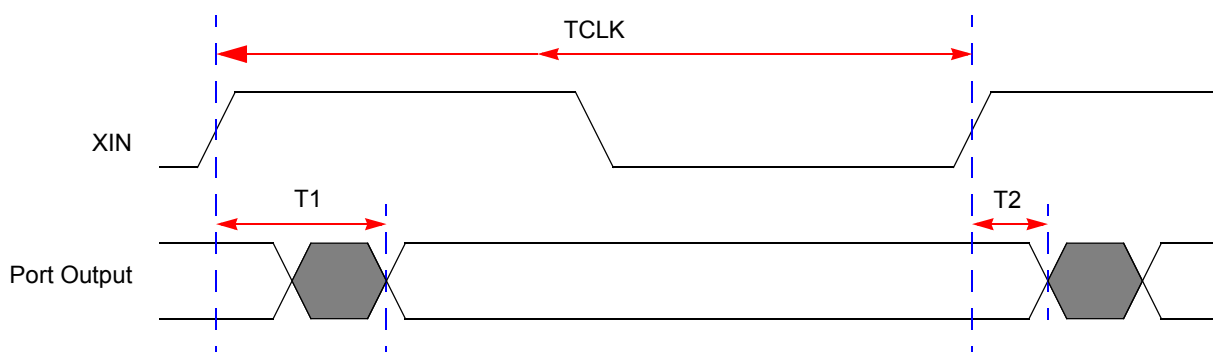


Figure 35. GPIO Port Output Timing

Table 140. GPIO Port Output Timing

Parameter	Abbreviation	Delay (ns)	
		Minimum	Maximum
GPIO Port pins			
T ₁	XIN Rise to Port Output Valid Delay	–	15
T ₂	XIN Rise to Port Output Hold Time	2	–

Ordering Information

Order the Z8 Encore! XP[®] F082A Series from Zilog[®], using the following part numbers. For more information on ordering, please consult your local Zilog sales office. The Zilog website (www.zilog.com) lists all regional offices and provides additional Z8 Encore! XP product information.

Part Number	Flash	RAM	NVDS	I/O Lines	Interrupts	16-Bit Timers w/PWM	10-Bit A/D Channels	UART with IrDA	Comparator	Temperature Sensor	Description
Z8 Encore! XP[®] F082A Series with 8 KB Flash, 10-Bit Analog-to-Digital Converter											
Standard Temperature: 0 °C to 70 °C											
Z8F082APB020SC	8 KB	1 KB	0	6	14	2	4	1	1	1	PDIP 8-pin package
Z8F082AQB020SC	8 KB	1 KB	0	6	14	2	4	1	1	1	QFN 8-pin package
Z8F082ASB020SC	8 KB	1 KB	0	6	14	2	4	1	1	1	SOIC 8-pin package
Z8F082ASH020SC	8 KB	1 KB	0	17	20	2	7	1	1	1	SOIC 20-pin package
Z8F082AHH020SC	8 KB	1 KB	0	17	20	2	7	1	1	1	SSOP 20-pin package
Z8F082APH020SC	8 KB	1 KB	0	17	20	2	7	1	1	1	PDIP 20-pin package
Z8F082ASJ020SC	8 KB	1 KB	0	23	20	2	8	1	1	1	SOIC 28-pin package
Z8F082AHJ020SC	8 KB	1 KB	0	23	20	2	8	1	1	1	SSOP 28-pin package
Z8F082APJ020SC	8 KB	1 KB	0	23	20	2	8	1	1	1	PDIP 28-pin package
Extended Temperature: -40 °C to 105 °C											
Z8F082APB020EC	8 KB	1 KB	0	6	14	2	4	1	1	1	PDIP 8-pin package
Z8F082AQB020EC	8 KB	1 KB	0	6	14	2	4	1	1	1	QFN 8-pin package
Z8F082ASB020EC	8 KB	1 KB	0	6	14	2	4	1	1	1	SOIC 8-pin package
Z8F082ASH020EC	8 KB	1 KB	0	17	20	2	7	1	1	1	SOIC 20-pin package
Z8F082AHH020EC	8 KB	1 KB	0	17	20	2	7	1	1	1	SSOP 20-pin package
Z8F082APH020EC	8 KB	1 KB	0	17	20	2	7	1	1	1	PDIP 20-pin package
Z8F082ASJ020EC	8 KB	1 KB	0	23	20	2	8	1	1	1	SOIC 28-pin package
Z8F082AHJ020EC	8 KB	1 KB	0	23	20	2	8	1	1	1	SSOP 28-pin package
Z8F082APJ020EC	8 KB	1 KB	0	23	20	2	8	1	1	1	PDIP 28-pin package
Replace C with G for Lead-Free Packaging											