



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	8KB (8K 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	Surface Mount
Package / Case	20-SSOP (0.209", 5.30mm Width)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f0813hh005sg

Flash Code Protection Against Accidental Program and Erasure	137
Byte Programming	139
Page Erase	139
Mass Erase	139
Flash Controller Bypass	140
Flash Controller Behavior in DEBUG Mode	140
Flash Control Register Definitions	141
Flash Control Register	141
Flash Status Register	142
Flash Page Select Register	142
Flash Sector Protect Register	144
Flash Frequency High and Low Byte Registers	144
Flash Option Bits	146
Operation	146
Option Bit Configuration By Reset	146
Option Bit Types	147
Reading the Flash Information Page	148
Flash Option Bit Control Register Definitions	148
Trim Bit Address Register	148
Trim Bit Data Register	149
Flash Option Bit Address Space	149
Trim Bit Address Space	151
Zilog Calibration Data	152
ADC Calibration Data	153
Serialization Data	154
Randomized Lot Identifier	154
On-Chip Debugger	156
Architecture	156
Operation	157
OCD Interface	157
DEBUG Mode	158
OCD Data Format	159
OCD Autobaud Detector/Generator	159
OCD Serial Errors	160
OCD Unlock Sequence (8-Pin Devices Only)	161
Breakpoints	161
Runtime Counter	161
On-Chip Debugger Commands	162
On-Chip Debugger Control Register Definitions	166
OCD Control Register	166

OCD Status Register	168
Oscillator Control	169
Operation	169
System Clock Selection	169
Clock Failure Detection and Recovery	170
Oscillator Control Register Definitions	171
Internal Precision Oscillator	173
Operation	173
eZ8 CPU Instruction Set	174
Assembly Language Programming Introduction	174
Assembly Language Syntax	175
eZ8 CPU Instruction Notation	176
eZ8 CPU Instruction Classes	178
eZ8 CPU Instruction Summary	182
Opcode Maps	192
Electrical Characteristics	196
Absolute Maximum Ratings	196
DC Characteristics	197
AC Characteristics	200
On-Chip Peripheral AC and DC Electrical Characteristics	201
General Purpose I/O Port Input Data Sample Timing	204
General Purpose I/O Port Output Timing	206
On-Chip Debugger Timing	207
UART Timing	208
Packaging	210
Ordering Information	211
Part Number Suffix Designations	220
Index	221
Customer Support	230

CPU and Peripheral Overview

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 about the eZ8 CPU, refer to the [eZ8 CPU Core User Manual \(UM0128\)](#) available for download at www.zilog.com.

General-Purpose I/O

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. 5V-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.

HALT Mode

Executing the eZ8 CPU's HALT instruction places the device into HALT Mode, which powers down the CPU but leaves all other peripherals active. In HALT Mode, the operating characteristics are:

- Primary oscillator is enabled and continues to operate
- System clock is enabled and continues to operate
- eZ8 CPU is stopped
- Program counter stops incrementing
- Watchdog Timer's internal RC oscillator continues to operate
- If enabled, the Watchdog Timer continues to operate
- All other on-chip peripherals continue to operate

The eZ8 CPU can be brought out of HALT Mode by any of the following operations:

- Interrupt
- Watchdog Timer time-out (interrupt or reset)
- Power-On Reset
- Voltage Brown-Out reset
- External $\overline{\text{RESET}}$ pin assertion

To minimize current in HALT Mode, all GPIO pins that are configured as inputs must be driven to one of the supply rails (V_{CC} or GND).

Peripheral-Level Power Control

In addition to the STOP and HALT modes, it is possible to disable each peripheral on each of the Z8 Encore! XP F0823 Series devices. Disabling a given peripheral minimizes its power consumption.

Power Control Register Definitions

The following sections describe the power control registers.

Power Control Register 0

Each bit of the following registers disables a peripheral block, either by gating its system clock input or by removing power from the block.

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

Table 14. Power Control Register 0 (PWRCTL0)

Bit	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
Address	F80H							

Bit	Description
[7]	Reserved This bit is reserved and must be programmed to 1.
[6:5]	Reserved These bits are reserved and must be programmed to 00.
[4] VBO	Voltage Brown-Out 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.
[3]	Reserved This bit is reserved and must be programmed to 0.
[2] ADC	Analog-to-Digital Converter Disable 0 = Analog-to-Digital Converter enabled. 1 = Analog-to-Digital Converter disabled.
[1] COMP	Comparator Disable 0 = Comparator is enabled. 1 = Comparator is disabled.
[0]	Reserved This bit is reserved and must be programmed to 0.

Table 17. Port Alternate Function Mapping (Non 8-Pin Parts) (Continued)

Port	Pin	Mnemonic	Alternate Function Description	Alternate Function Set Register AFS1
Port B ³	PB03	Reserved		AFS1[0]: 0
		ANA0	ADC Analog Input	AFS1[0]: 1
	PB1	Reserved		AFS1[1]: 0
		ANA1	ADC Analog Input	AFS1[1]: 1
	PB2	Reserved		AFS1[2]: 0
		ANA2	ADC Analog Input	AFS1[2]: 1
	PB3	CLKIN	External Clock Input	AFS1[3]: 0
		ANA3	ADC Analog Input	AFS1[3]: 1
	PB4	Reserved		AFS1[4]: 0
		ANA7	ADC Analog Input	AFS1[4]: 1
	PB5	Reserved		AFS1[5]: 0
		V _{REF} ⁴	ADC Voltage Reference	AFS1[5]: 1
	PB6	Reserved		AFS1[6]: 0
		Reserved		AFS1[6]: 1
	PB7	Reserved		AFS1[7]: 0
		Reserved		AFS1[7]: 1

Notes:

1. 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 the [Port A–C Alternate Function Subregisters](#) section on page 43 automatically enables the associated alternate function.
2. Whether PA0/PA6 take on the timer input or timer output complement function depends on the timer configuration as described in the [Timer Pin Signal Operation](#) section on page 83.
3. Because there are at most two choices of alternate function for any pin of Port B, the Alternate Function Set register AFS2 is implemented but not used to select the function. Also, alternate function selection as described in the [Port A–C Alternate Function Subregisters](#) section on page 43 must also be enabled.
4. V_{REF} is available on PB5 in 28-pin products only.
5. Because there are at most two choices of alternate function for any pin of Port C, the Alternate Function Set register AFS2 is implemented but not used to select the function. Also, Alternate Function selection as described in the [Port A–C Alternate Function Subregisters](#) section on page 43 must also be enabled.
6. V_{REF} is available on PC2 in 20-pin parts only.

Port A–C Data Direction Subregisters

The Port A–C Data Direction Subregister is accessed through the Port A–C Control Register by writing 01H to the Port A–C Address Register; see Table 22.

Table 22. Port A–C Data Direction Subregisters (PxDD)

Bit	7	6	5	4	3	2	1	0
Field	DD7	DD6	DD5	DD4	DD3	DD2	DD1	DD0
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	If 01H in Port A–C Address Register, accessible through the Port A–C Control Register.							

Bit	Description
[7:0]	Data Direction
DDx	<p>These bits control the direction of the associated port pin. Port Alternate Function operation overrides the Data Direction register setting.</p> <p>0 = Output. Data in the Port A–C Output Data Register is driven onto the port pin.</p> <p>1 = Input. The port pin is sampled and the value written into the Port A–C Input Data Register.</p> <p>The output driver is tristated.</p>

Note: x indicates the specific GPIO port pin number (7–0).

Port A–C Alternate Function Subregisters

The Port A–C Alternate Function Subregister (Table 23) is accessed through the Port A–C Control Register by writing 02H to the Port A–C Address Register. The Port A–C Alternate Function subregisters enable the alternate function selection on pins. If disabled, pins function as GPIO. If enabled, select one of four alternate functions using alternate function set subregisters 1 and 2 as described in the [Port A–C Alternate Function Set 1 Subregisters](#) section on page 48 and the [Port A–C Alternate Function Set 2 Subregisters](#) section on page 49. See the [GPIO Alternate Functions](#) section on page 34 to determine the alternate function associated with each port pin.

! Caution: Do not enable alternate functions for GPIO port pins for which there is no associated alternate function. Failure to follow this guideline can result in unpredictable operation.

Port A–C Stop Mode Recovery Source Enable Subregisters

The Port A–C Stop Mode Recovery Source Enable Subregister (Table 26) 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 subregisters 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 26. Port A–C Stop Mode Recovery Source Enable Subregisters (PSMREx)

Bit	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
Address	If 05H in Port A–C Address Register, accessible through the Port A–C Control Register							

Bit	Description
[7:0]	Port Stop Mode Recovery Source Enabled.
PSMREx	0 = The Port pin is not configured as a Stop Mode Recovery source. Transitions on this pin during STOP Mode do not initiate Stop Mode Recovery. 1 = The Port pin is configured as a Stop Mode Recovery source. Any logic transition on this pin during STOP Mode initiates Stop Mode Recovery.

Note: x indicates the specific GPIO port pin number (7–0).

Bit	Description (Continued)
[4] 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.
[3] 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.
[2:1]	Reserved These bits are reserved and must be programmed to 00.
[0] ADC1	ADC Interrupt Request 0 = No interrupt request is pending for the ADC. 1 = An interrupt request from the ADC is awaiting service.

Interrupt Request 1 Register

The Interrupt Request 1 (IRQ1) register (Table 37) 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 reads the Interrupt Request 1 Register to determine if any interrupt requests are pending.

Table 37. Interrupt Request 1 Register (IRQ1)

Bit	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
Address	FC3H							

Bit	Description
[7] PA7VI	Port A7 Interrupt Request 0 = No interrupt request is pending for GPIO Port A. 1 = An interrupt request from GPIO Port A.
[6] PA6CI	Port A6 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.
[5:0] 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.

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

PWM SINGLE OUTPUT Mode

In PWM SINGLE OUTPUT Mode, the timer outputs a PWM output signal through a GPIO port pin. The timer input is the system clock. The timer first counts up to the 16-bit PWM match value stored in the Timer PWM High and Low Byte registers. When the timer count value matches the PWM value, the Timer Output toggles. The timer continues counting until it reaches the reload 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.

If the **TPOL** bit in the Timer Control Register is set to 1, the Timer Output signal begins as a High (1) and transitions to a Low (0) when the timer value matches the PWM value. The Timer Output signal returns to a High (1) after the timer reaches the reload value and is reset to 0001H.

If the **TPOL** bit in the Timer Control Register is set to 0, the Timer Output signal begins as a Low (0) and transitions to a High (1) when the timer value matches the PWM value. The Timer Output signal returns to a Low (0) after the timer reaches the reload value and is reset to 0001H.

Observe the following steps to configure a timer for PWM Single Output mode and initiating the PWM operation:

1. Write to the Timer Control Register to:
 - Disable the timer
 - Configure the timer for PWM Mode
 - Set the prescale value
 - Set the initial logic level (High or Low) and PWM High/Low transition for the Timer Output alternate function
2. Write to the Timer High and Low Byte registers to set the starting count value (typically 0001H); this write only affects the first pass in PWM Mode. After the first timer reset in PWM Mode, counting always begins at the reset value of 0001H.
3. Write to the PWM High and Low Byte registers to set the PWM value.
4. Write to the Timer Reload High and Low Byte registers to set the reload value (PWM period). The reload value must be greater than the PWM value.
5. If appropriate, enable the timer interrupt and set the timer interrupt priority by writing to the relevant interrupt registers.
6. Configure the associated GPIO port pin for the Timer Output alternate function.
7. Write to the Timer Control Register to enable the timer and initiate counting.

The PWM period is represented by the following equation:

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

CAPTURE/COMPARE Mode

In CAPTURE/COMPARE Mode, the timer begins counting on the first external Timer Input transition. The acceptable transition (rising edge or falling edge) is set by the TPOL bit in the Timer Control Register. The timer input is the system clock.

Every subsequent acceptable transition (after the first) of the Timer Input signal captures the current count value. The capture value is written to the Timer PWM High and Low Byte registers. When the capture event occurs, an interrupt is generated, 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 caused by 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 compare 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 because of an input capture event.

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

1. Write to the Timer Control Register to:
 - Disable the timer
 - Configure the timer for CAPTURE/COMPARE Mode
 - Set the prescale value
 - Set the capture edge (rising or falling) for the Timer Input
2. Write to the Timer High and Low Byte registers to set the starting count value (typically 0001H).
3. Write to the Timer Reload High and Low Byte registers to set the compare value.
4. Enable the timer interrupt, if appropriate, and set the timer interrupt priority by writing to the relevant interrupt registers. By default, the timer interrupt are generated for both 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.
5. Configure the associated GPIO port pin for the Timer Input alternate function.
6. Write to the Timer Control Register to enable the timer.

1. Checks the UART Status 0 Register to determine the source of the interrupt - error, break, or received data.
2. Reads the data from the UART Receive Data Register if the interrupt was because of data available. If operating in MULTIPROCESSOR (9-bit) Mode, further actions may be required depending on the MULTIPROCESSOR Mode bits MPMD[1:0].
3. Clears the UART Receiver interrupt in the applicable Interrupt Request register.
4. Executes the IRET instruction to return from the interrupt-service routine and await more data.

Clear To Send ($\overline{\text{CTS}}$) Operation

The CTS pin, if enabled by the CTSE bit of the UART Control 0 Register, performs flow control on the outgoing transmit datastream. The Clear To Send ($\overline{\text{CTS}}$) input pin is sampled one system clock before beginning any new character transmission. To delay transmission of the next data character, an external receiver must deassert $\overline{\text{CTS}}$ at least one system clock cycle before a new data transmission begins. For multiple character transmissions, this action is typically performed during Stop Bit transmission. If $\overline{\text{CTS}}$ deasserts in the middle of a character transmission, the current character is sent completely.

MULTIPROCESSOR (9-Bit) Mode

The UART has a MULTIPROCESSOR (9-bit) Mode that uses an extra (9th) bit for selective communication when a number of processors share a common UART bus. In MULTIPROCESSOR Mode (also referred to as 9-bit mode), the multiprocessor bit (MP) is transmitted immediately following the 8-bits of data and immediately preceding the Stop bit(s) as displayed in Figure 13. The character format is given below:

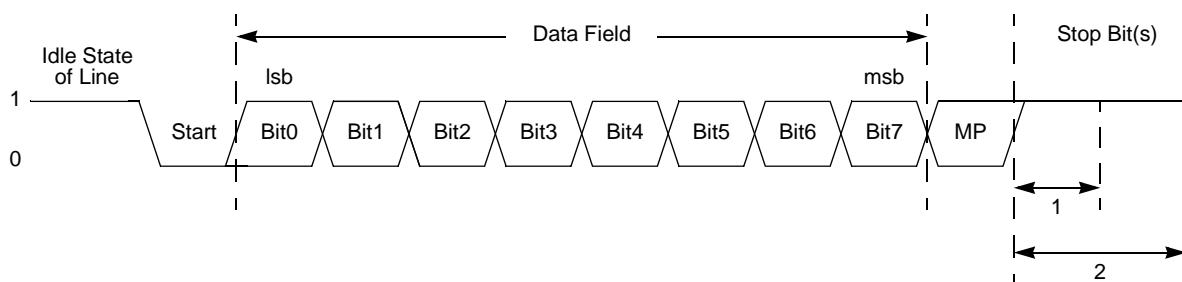


Figure 13. UART Asynchronous MULTIPROCESSOR Mode Data Format

In MULTIPROCESSOR (9-bit) Mode, the parity bit location (9th bit) becomes the Multiprocessor control bit. The UART Control 1 and Status 1 registers provide MULTIPROCESSOR (9-bit) Mode control and status information. If an automatic address matching

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.

Software Compensation Procedure

The value read from the ADC high and low byte registers are uncompensated. The user mode software must apply gain and offset correction to this uncompensated value for maximum accuracy. The following formula yields the compensated value:

$$ADC_{comp} = (ADC_{uncomp} - OFFCAL) + ((ADC_{uncomp} - OFFCAL) * GAINCAL) / 2$$

where GAINCAL is the gain calibration byte, OFFCAL is the offset calibration byte and ADC_{uncomp} is the uncompensated value read from the ADC. The OFFCAL value is in two's complement format, as are the compensated and uncompensated ADC values.

► **Note:** The offset compensation is performed first, followed by the gain compensation. One bit of resolution is lost because of rounding on both the offset and gain computations. As a result the ADC registers read back 13 bits: 1 sign bit, two calibration bits lost to rounding and 10 data bits. Also note that in the second term, the multiplication must be performed before the division by 2^{16} . Otherwise, the second term evaluates to zero incorrectly.

! **Caution:** Although the ADC can be used without the gain and offset compensation, it does exhibit non-unity gain. Designing the ADC with sub-unity gain reduces noise across the ADC range but requires the ADC results to be scaled by a factor of 8/7.

ADC Control Register Definitions

The following sections define the ADC Control registers.

ADC Control Register 0

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

Option Bit Types

This section describes the five types of Flash option bits offered in the F083A Series.

User Option Bits

The user option bits are contained in the first two bytes of program memory. Access to these bits has been provided because these locations contain application-specific device configurations. The information contained here is lost when page 0 in program memory is erased.

Trim Option Bits

The trim option bits are contained in a Flash memory information page. These bits are factory programmed values required to optimize the operation of onboard analog circuitry and cannot be permanently altered. Program memory may be erased without endangering these values. It is possible to alter working values of these bits by accessing the Trim Bit Address and Data Registers, but these working values are lost after a power loss or any other reset event.

There are 32 bytes of trim data. To modify one of these values the user code must first write a value between 00H and 1FH into the Trim Bit Address Register. The next write to the Trim Bit Data Register changes the working value of the target trim data byte.

Reading the trim data requires the user code to write a value between 00H and 1FH into the Trim Bit Address Register. The next read from the Trim Bit Data Register returns the working value of the target trim data byte.

► **Note:** The trim address range is from information address 20–3F only. The remainder of the information page is not accessible through the trim bit address and data registers.

Calibration Option Bits

The calibration option bits are also contained in the information page. These bits are factory programmed values intended for use in software correcting the device's analog performance. To read these values, the user code must employ the LDC instruction to access the information area of the address space as defined in the [Flash Information Area](#) section on page 15.

Serialization Bits

As an optional feature, Zilog is able to provide factory-programmed serialization. For serialized products, the individual devices are programmed with unique serial numbers. These serial numbers are binary values, four bytes in length. The numbers increase in size with each device, but gaps in the serial sequence may exist.

Bit	Description (Continued)
[3] VBO_AO	Voltage Brown-Out Protection Always ON 0 = Voltage Brown-Out Protection can be disabled in STOP Mode to reduce total power consumption. For the block to be disabled, the power control register bit must also be written (see the Power Control Register 0 section on page 31). 1 = Voltage Brown-Out Protection is always enabled including during STOP Mode. This setting is the default for unprogrammed (erased) Flash.
[2] FRP	Flash Read Protect 0 = User program code is inaccessible. Limited control features are available through the On-Chip Debugger. 1 = User program code is accessible. All On-Chip Debugger commands are enabled. This setting is the default for unprogrammed (erased) Flash.
[1]	Reserved This bit is reserved and must be programmed to 1.
[0] FWP	Flash Write Protect This Option Bit provides Flash Program Memory protection: 0 = Programming and erasure disabled for all of Flash Program Memory. Programming, Page Erase, and Mass Erase through User Code is disabled. Mass Erase is available using the On-Chip Debugger. 1 = Programming, Page Erase, and Mass Erase are enabled for all of Flash program memory.

Table 90. Flash Options Bits at Program Memory Address 0001H

Bit	7	6	5	4	3	2	1	0
Field	Reserved			XTLDIS	Reserved			
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 0001H							
Note: U = Unchanged by Reset. R/W = Read/Write.								

Operation

The following section describes the operation of the OCD.

OCD Interface

The OCD uses the DBG pin for communication with an external host. This one-pin interface is a bidirectional 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 F0823 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 23 and Figure 24. The recommended method is the buffered implementation depicted in Figure 24. The DBG pin has an internal pull-up resistor which is sufficient for some applications (for more details about the pull-up current, see the [Electrical Characteristics](#) chapter on page 196). For OCD operation at higher data rates or in noisy systems, Zilog recommends an external pull-up resistor.

! Caution: For operation of the OCD, 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.

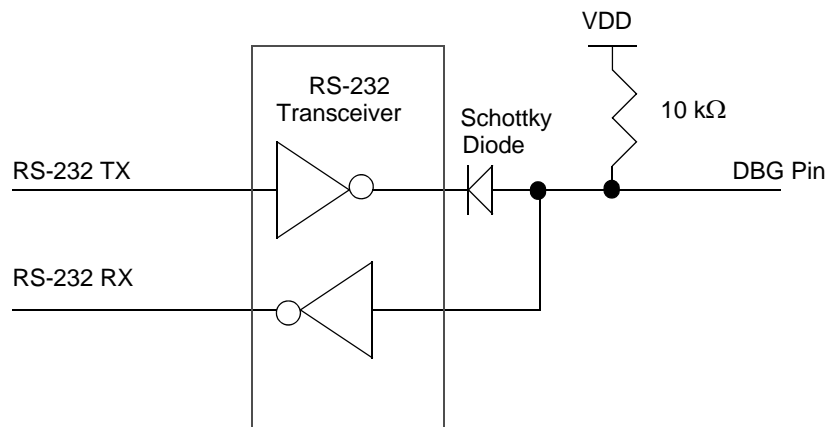


Figure 23. Interfacing the On-Chip Debugger's DBG Pin with an RS-232 Interface, # 1 of 2

OCD Status Register

The OCD Status Register reports status information about the current state of the debugger and the system.

Table 103. OCD Status Register (OCDSTAT)

Bit	7	6	5	4	3	2	1	0
Field	DBG	HALT	FRPENB	Reserved				
RESET	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R

Bit	Description
[7] DBG	Debug Status 0 = NORMAL Mode. 1 = DEBUG Mode.
[6] HALT	HALT Mode 0 = Not in HALT Mode. 1 = In HALT Mode.
[5] FRPENB	Flash Read Protect Option Bit Enable 0 = FRP bit enabled to allow disabling of many OCD commands. 1 = FRP bit has no effect.
[4:0]	Reserved These bits are reserved and must be 00000 when read.

eZ8 CPU Instruction Set

This chapter describes the following features of the eZ8 CPU instruction set:

Assembly Language Programming Introduction: see page 174

Assembly Language Syntax: see page 175

eZ8 CPU Instruction Notation: see page 176

eZ8 CPU Instruction Classes: see page 178

eZ8 CPU Instruction Summary: see page 182

Assembly Language Programming Introduction

The eZ8 CPU assembly language provides a means for writing an application program without concern for actual memory addresses or machine instruction formats. A program written in assembly language is called a source program. Assembly language allows the use of symbolic addresses to identify memory locations. It also allows mnemonic codes (opcodes and operands) to represent the instructions themselves. The opcodes identify the instruction while the operands represent memory locations, registers, or immediate data values.

Each assembly language program consists of a series of symbolic commands called *statements*. Each statement can contain labels, operations, operands, and comments.

Labels are assigned to a particular instruction step in a source program. The label identifies that step in the program as an entry point for use by other instructions.

The assembly language also includes assembler directives that supplement the machine instruction. The assembler directives, or pseudo-ops, are not translated into a machine instruction. Rather, the pseudo-ops are interpreted as directives that control or assist the assembly process.

The source program is processed (assembled) by the assembler to obtain a machine language program called the object code. The object code is executed by the eZ8 CPU. An example segment of an assembly language program is detailed in the following example.

Table 108. Notational Shorthand (Continued)

Notation	Description	Operand	Range
RA	Relative Address	X	X represents an index in the range of +127 to –128 which is an offset relative to the address of the next instruction
rr	Working Register Pair	RRp	p = 0, 2, 4, 6, 8, 10, 12, or 14.
RR	Register Pair	Reg	Reg. represents an even number in the range of 00H to FEH.
Vector	Vector Address	Vector	Vector represents a number in the range of 00H to FFH.
X	Indexed	#Index	The register or register pair to be indexed is offset by the signed Index value (#Index) in a +127 to –128 range.

Table 109 lists additional symbols that are used throughout the Instruction Summary and Instruction Set Description sections.

Table 109. Additional Symbols

Symbol	Definition
dst	Destination Operand
src	Source Operand
@	Indirect Address Prefix
SP	Stack Pointer
PC	Program Counter
FLAGS	Flags Register
RP	Register Pointer
#	Immediate Operand Prefix
B	Binary Number Suffix
%	Hexadecimal Number Prefix
H	Hexadecimal Number Suffix

Assignment of a value is indicated by an arrow, as shown in the following example.

$\text{dst} \leftarrow \text{dst} + \text{src}$

This example indicates that the source data is added to the destination data; the result is stored in the destination location.