Zilog - Z8F083AHH020EG Datasheet

Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

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

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	eZ8
Core Size	8-Bit
Speed	20MHz
Connectivity	-
Peripherals	Brown-out Detect/Reset, LED, POR, PWM, WDT
Number of I/O	17
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	256 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 3.6V
Data Converters	A/D 7x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	20-SSOP (0.209", 5.30mm Width)
Supplier Device Package	20-SSOP
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f083ahh020eg

Block Diagram

Figure 1 displays a block diagram of the Z8 Encore! F083A Series architecture.

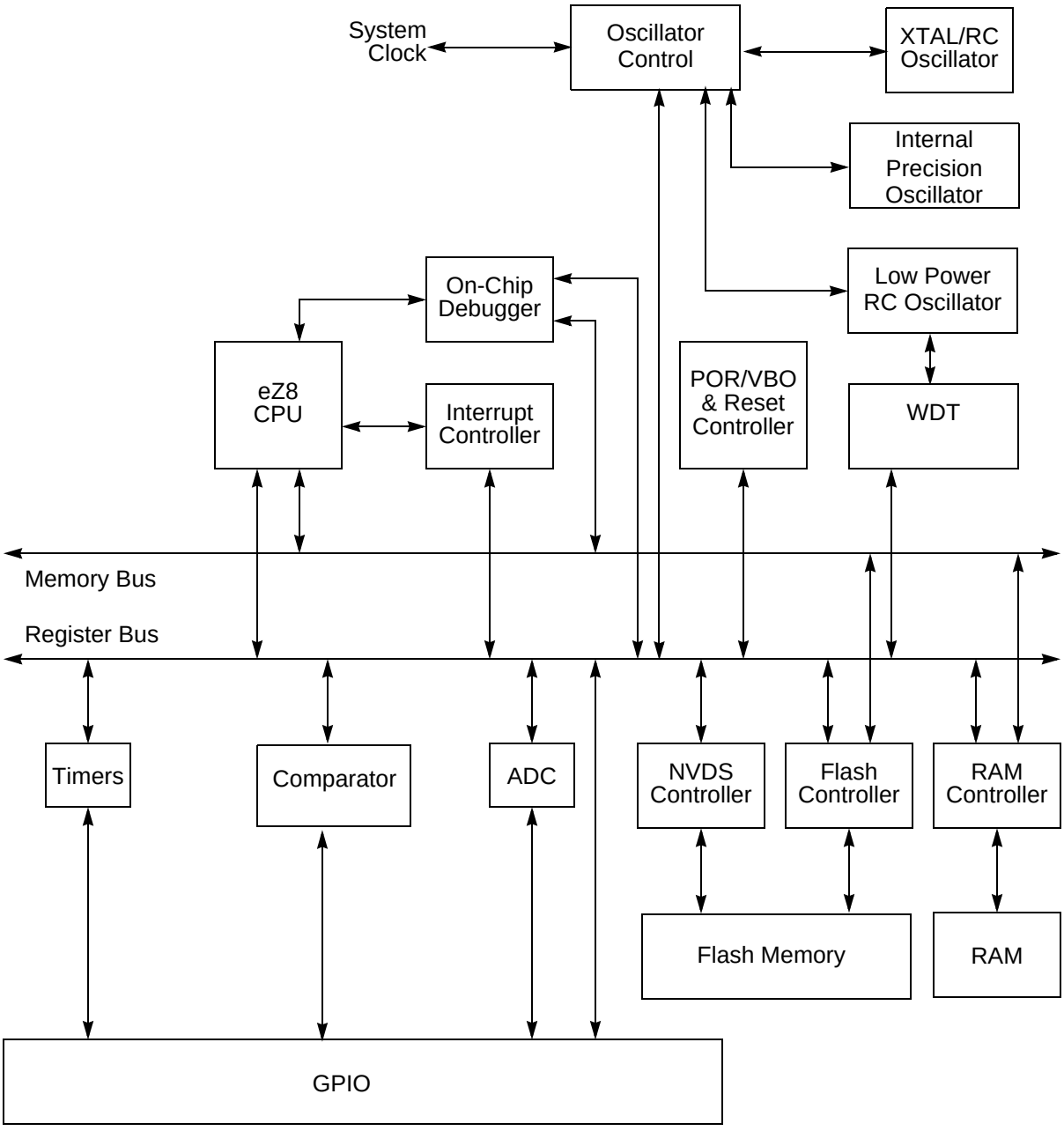

Figure 1. Z8 Encore! F083A Series Block Diagram

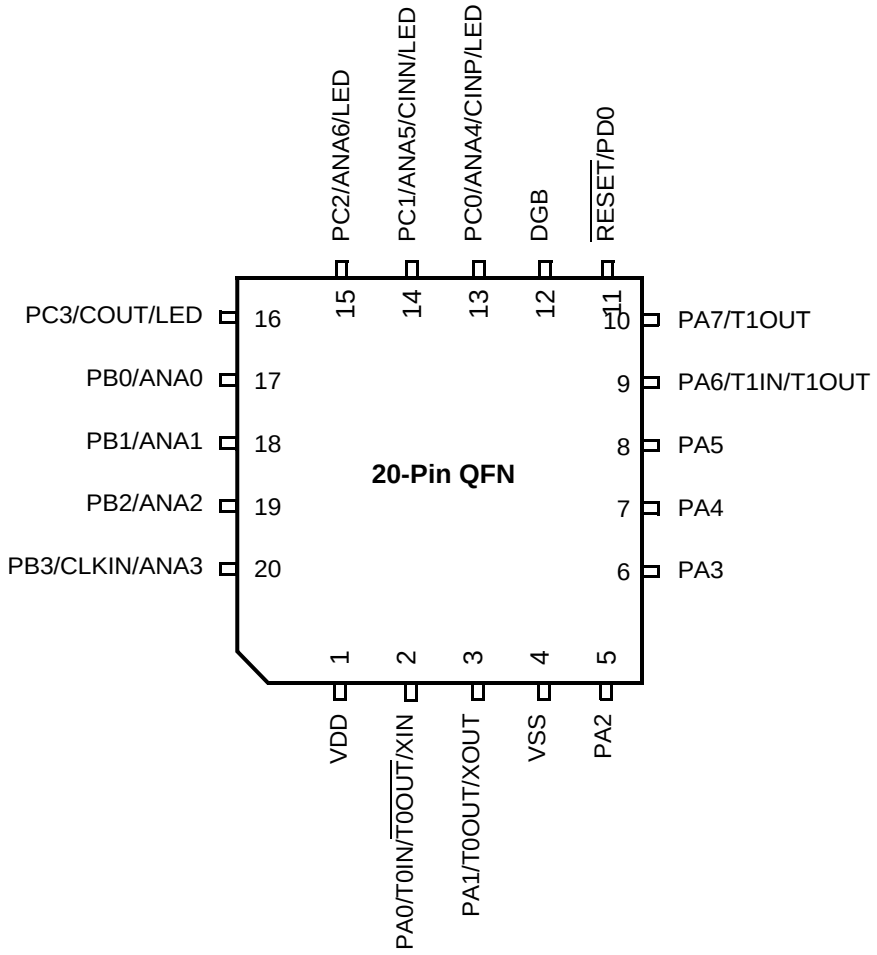

Figure 4. Z8F083A Series in 20-Pin QFN Package

Reset and Stop Mode Recovery

The reset controller in the Z8 Encore! F083A Series controls Reset and Stop Mode Recovery operations. In a typical operation, the following events cause a Reset:

- Power-On Reset
- Voltage Brown-Out
- Watchdog Timer time-out (when configured by the WDT_RES Flash option bit to initiate a reset)
- External $\overline{\text{RESET}}$ pin assertion (when the alternate Reset function is enabled by the GPIO register)
- On-Chip Debugger initiated reset (OCDCTL[0] set to 1)

When the device is in STOP Mode, a Stop Mode Recovery is initiated by either of the following:

- WDT time-out
- GPIO port input pin transition on an enabled Stop Mode Recovery source

The VBO circuitry on the device generates the VBO reset when the supply voltage drops below a minimum safe level.

Reset Types

The Z8 Encore! F083A Series provides different types of Reset operation. Stop Mode Recovery is considered as a form of reset. Table 9 lists the types of reset and their operating characteristics. The system reset is longer, if the external crystal oscillator is enabled by the Flash option bits, allowing additional time for oscillator start-up.

Table 9. Reset and Stop Mode Recovery Characteristics and Latency

	Reset Characteristics and Latency		
Reset Type	**Control Registers**	**eZ8 CPU**	**Reset Latency (Delay)**
System Reset	Reset (as applicable)	Reset	About 66 internal precision oscillator cycles.
System Reset with Crystal Oscillator Enabled	Reset (as applicable)	Reset	About 5000 internal precision oscillator cycles.
Stop Mode Recovery	Unaffected, except WDT_CTL and OSC_CTL registers	Reset	About 66 internal precision oscillator cycles.
Stop Mode Recovery with crystal oscillator enabled	Unaffected, except WDT_CTL and OSC_CTL registers	Reset	About 5000 internal precision oscillator cycles.

During a system Reset or Stop Mode Recovery, the Z8 Encore! F083A Series device is held in reset for about 66 cycles of the internal precision oscillator. If the crystal oscillator is enabled in the Flash option bits, the reset period is increased to about 5000 IPO cycles. When a reset occurs because of a low voltage condition or POR, the reset delay is measured from the time the supply voltage first exceeds the POR level (discussed later in this chapter). If the external pin reset remains asserted at the end of the reset period, the device remains in reset until the pin is deasserted.

At the beginning of reset, all GPIO pins are configured as inputs with pull-up resistor disabled, except PD0 which is shared with the reset pin. On Reset, the Port D0 pin is configured as a bidirectional open-drain reset. This pin is internally driven low during port reset, after which the user code reconfigures this pin as a general purpose output.

During reset, the eZ8 CPU and on-chip peripherals are idle; however, the on-chip crystal oscillator and Watchdog Timer Oscillator continues to run.

On reset, control registers within the Register File that have a defined reset value are loaded with their reset values. Other control registers (including the Stack Pointer, Register Pointer and Flags) and general purpose RAM are undefined following the reset. The eZ8 CPU fetches the reset vector at program memory addresses `0002H` and `0003H` and loads that value into the program counter. Program execution begins at the reset vector address.

Because the control registers are reinitialized by a system reset, the system clock after reset is always the IPO. User software must reconfigure the oscillator control block, to enable and select the correct system clock source.

Reset Sources

Table 10 lists the possible sources of a system reset.

Table 10. Reset Sources and Resulting Reset Type

Operating Mode	Reset Source	Special Conditions
NORMAL or HALT modes	Power-On Reset / Voltage Brown-Out.	Reset delay begins after supply voltage exceeds POR level.
	WDT time-out when configured for reset.	None.
	$\overline{\text{RESET}}$ pin assertion.	All reset pulses less than four system clocks in width are ignored.
	On-Chip Debugger initiated reset (OCDCTL[0] set to 1).	System, except the On-Chip Debugger is unaffected by the reset.
STOP Mode	Power-On Reset / Voltage Brown-Out.	Reset delay begins after supply voltage exceeds POR level.
	$\overline{\text{RESET}}$ pin assertion.	All reset pulses less than 12 ns are ignored.
	DBG pin driven Low.	None.

Power-On Reset

Each device in the Z8 Encore! F083A Series contains an internal POR circuit. The POR circuit monitors the digital supply voltage and holds the device in the Reset state until the digital supply voltage reaches a safe operating level. After the supply voltage exceeds the POR voltage threshold (V_{POR}), the device is held in the Reset state until the POR counter has timed out. If the crystal oscillator is enabled by the option bits, the time-out is longer.

After the Z8 Encore! F083A Series device exits the POR state, the eZ8 CPU fetches the reset vector. Following the POR, the POR status bit in the Reset Status (RSTSTAT) Register is set to 1.

Figure 6 displays POR operation. For POR threshold voltage (V_{POR}), see the Electrical Characteristics chapter on page 184.

Stop Mode Recovery Using the External $\overline{\text{RESET}}$ Pin

When the Z8 Encore! F083A Series device is in STOP Mode and the external $\overline{\text{RESET}}$ pin is driven low, a system reset occurs. Because of a glitch filter operating on the $\overline{\text{RESET}}$ pin, the low pulse must be greater than the minimum width specified about 12 ns or it is ignored. The EXT bit in the Reset Status (RSTSTAT) Register is set.

Debug Pin Driven Low

Debug reset is initiated when the On-Chip Debugger detects any of the following error conditions on the DBG pin:

- Serial break (a minimum of nine continuous bits low)
- Framing error (received STOP bit is low)
- Transmit collision (OCD and host simultaneous transmission detected by the OCD)

When the Z8F083 is in STOP Mode, the debug reset will cause a system reset. The On-Chip Debugger block is not reset, but the remainder of the chip goes through a normal system reset. The POR bit in the reset (RSTSTAT) Register is set to 1.

Reset Register Definitions

The following sections define the Reset registers.

Reset Status Register

The Reset Status (RSTSTAT) Register detailed in Table 12 is a read-only register that indicates the source of the most recent Reset event, a Stop Mode Recovery event or a WDT time-out event. Reading this register resets the upper four bits to 0.

This register shares its address with the Watchdog Timer Control Register, which is write-only.

Port A–D Control Registers

The Port A–D control registers, shown in Table 20, set GPIO port operation. The value in the corresponding Port A–D Address Register determines which subregister is read from or written to by a Port A–D Control Register transaction.

Table 20. Port A–D Control Registers (P*x*CTL)

Bit	7	6	5	4	3	2	1	0
Field	PCTL							
RESET	00H							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Address	FD1H, FD5H, FD9H, FDDH							

Bit	Description
[7:0] PCTL	**Port Control** The Port Control Register provides access to all subregisters that configure GPIO port operation.

Port A–D Data Direction Subregisters

The Port A–D Data Direction Subregister, shown in Table 21, is accessed through the Port A–D Control Register by writing 01H to the Port A–D Address Register.

Table 21. Port A–D 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–D Address Register, accessible through the Port A–D Control Register							

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

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

Interrupt Request 0 Register

The Interrupt Request 0 (IRQ0) Register, shown in Table 35, stores the interrupt requests for both vectored and polled interrupts. When a request is sent to the Interrupt Controller, the corresponding bit in the IRQ0 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 0 Register to determine if any interrupt requests are pending.

Table 35. Interrupt Request 0 Register (IRQ0)

Bit	7	6	5	4	3	2	1	0
Field	Reserved	T1I	T0I	Reserved				ADCI
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	FC0H							

Bit	Description
[7]	**Reserved** This bit is reserved and must be programmed to 0.
[6] T1I	**Timer 1 Interrupt Request** 0 = No interrupt request is pending for Timer 1. 1 = An interrupt request from timer 1 is awaiting service.
[5] T0I	**Timer 0 Interrupt Request** 0 = No interrupt request is pending for Timer 0. 1 = An interrupt request from timer 0 is awaiting service.
[4:1]	**Reserved** These bits are reserved and must be programmed to 0000.
[0] ADCI	**ADC Interrupt Request** 0 = No interrupt request is pending for the ADC. 1 = An interrupt request from the ADC is awaiting service.

Observe the following steps to configure a timer for PWM DUAL OUTPUT Mode and for initiating the PWM operation are:

1. Write to the Timer Control Register to:
   - Disable the timer
   - Configure the timer for PWM DUAL OUTPUT Mode. Setting the mode also involves writing to TMODEHI bit in the TxCTL1 Register
   - 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 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 PWM Control Register to set the PWM deadband delay value. The deadband delay must be less than the duration of the positive phase of the PWM signal (as defined by the PWM High and Low Byte registers). It must also be less than the duration of the negative phase of the PWM signal (as defined by the difference between the PWM registers and the timer reload registers).
5. 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.
6. If appropriate, enable the timer interrupt and set the timer interrupt priority by writing to the relevant interrupt registers.
7. Configure the associated GPIO port pin for the timer output and timer output complement alternate functions. The timer output complement function is shared with the timer input function for both timers. Setting the timer mode to dual PWM, will automatically switch the function from timer-in to timer-out complement.
8. Write to the Timer Control Register to enable the timer and initiate counting.

The PWM period is represented by the following equation:

$$\text{PWM Period (s)} = \frac{\text{Reload Value} \times \text{Prescale}}{\text{System Clock Frequency (Hz)}}$$

If an initial starting value other than `0001H` is loaded into the Timer High and Low Byte registers, the ONE-SHOT Mode equation determines the first PWM time-out period.

If TPOL bit is set to 0, the ratio of the PWM output high time to the total period is represented by:

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. Counting begins on the first appropriate transition of the timer input signal. No interrupt is generated by the first edge.

In CAPTURE/COMPARE Mode, the elapsed time from timer start to capture event is calculated using the following equation:

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

Reading the Timer Count Values

The current count value in the timers are read (i.e., enabled) while counting. This capability has no effect on Timer operation. When the timer is enabled and the Timer High Byte Register is read, the contents of the Timer Low Byte Register are placed in a holding register. A subsequent read from the Timer Low Byte Register returns the value in the holding register. This operation allows accurate reads of the full 16-bit timer count value when enabled. When the timers are not enabled, a read from the Timer Low Byte Register returns the actual value in the counter.

Timer Pin Signal Operation

Timer output is a GPIO port pin alternate function. The timer output is toggled every time the counter is reloaded.

The timer input is used as a selectable counting source. It shares the same pin as the complementary timer output. When selected by the GPIO alternate function registers, this pin functions as a timer input in all modes except for the DUAL PWM OUTPUT Mode. For this mode, there is no timer input available.

Timer Control Register Definitions

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

Timer 0–1 High and Low Byte Registers: see page 84

Timer Reload High and Low Byte Registers: see page 84

Timer 0–1 PWM High and Low Byte Registers: see page 86

Timer 0–1 Control Registers: see page 87

Timer 0–1 High and Low Byte Registers

The Timer 0–1 High and Low Byte (TxH and TxL) registers, shown in Tables 50 and 51, contain the current 16-bit timer count value. When the timer is enabled, a read from TxH causes the value in TxL to be stored in a temporary holding register. A read from TxL always returns this temporary register content when the timer is enabled; however, when the timer is disabled, a read from the TxL reads the TxL Register content directly.

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

Table 50. Timer 0–1 High Byte Register (TxH)

Bit	7	6	5	4	3	2	1	0
Field	TH							
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	F00H, F08H							

Table 51. Timer 0–1 Low Byte Register (TxL)

Bit	7	6	5	4	3	2	1	0
Field	TL							
RESET	0	0	0	0	0	0	0	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Address	F01H, F09H							

Bit	Description
[7:0] TH, TL	**Timer High and Low Bytes** These 2 bytes, {TH[7:0], TL[7:0]}, contain the current 16-bit timer count value.

Timer Reload High and Low Byte Registers

The timer 0–1 reload High and Low Byte (TxRH and TxRL) registers, shown in Tables 52 and 53, store a 16-bit reload value, {TRH[7:0], TRL[7:0]}. Values written to the timer reload high byte register are stored in a temporary holding register. When a write to the timer reload low byte register occurs, the temporary holding register value is written to the Timer High Byte Register. This operation allows simultaneous updates of the 16-bit timer

Timer 0–1 PWM High and Low Byte Registers

The Timer 0–1 PWM High and Low Byte (TxPWMH and TxPWML) registers, shown in Tables 54 and 55, control PWM operations. These registers also store the capture values for the CAPTURE and CAPTURE/COMPARE modes.

Table 54. Timer 0–1 PWM High Byte Register (TxPWMH)

Bit	7	6	5	4	3	2	1	0
Field	PWMH							
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	F04H, F0CH							

Table 55. Timer 0–1 PWM Low Byte Register (TxPWML)

Bit	7	6	5	4	3	2	1	0
Field	PWML							
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	F05H, F0DH							

Bit	Description
[7:0] PWMH, PWML	**Pulse width modulator High and Low Bytes** These two bytes, {PWMH[7:0], PWML[7:0]}, form a 16-bit value that is compared to the current 16-bit timer count. When a match occurs, the PWM output changes state. The PWM output value is set by the TPOL bit in the Timer Control Register (TxCTL1). The TxPWMH and TxPWML registers also store the 16-bit captured timer value when operating in capture or CAPTURE/COMPARE modes.

Flash Frequency High and Low Byte Registers

The Flash Frequency High and Low Byte registers, shown in Tables 77 and 78, combine to form a 16-bit value, FFREQ, to control timing for Flash Program and Erase operations. The 16-bit binary Flash frequency value must contain the system clock frequency (in kHz) and is calculated using the following equation.

$$FFREQ[15:0] = \{FFREQH[7:0],FFREQL[7:0]\} = \frac{\text{System Clock Frequency}}{1000}$$

Caution: Flash programming and erasure is not supported for system clock frequencies below 10 kHz or above 20 MHz. The Flash frequency High and Low Byte registers must be loaded with the correct value to ensure proper operation of the device.

Table 77. Flash Frequency High Byte Register (FFREQH)

Bit	7	6	5	4	3	2	1	0
Field	FFREQH							
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	FFAH							

Bit	Description
[7:0] FFREQH	**Flash Frequency High Byte** The high byte of the 16-bit Flash frequency value.

Table 78. Flash Frequency Low Byte Register (FFREQL)

Bit	7	6	5	4	3	2	1	0
Field	FFREQL							
RESET	0							
R/W	R/W							
Address	FFBH							

Bit	Description
[7:0] FFREQL	**Flash Frequency Low Byte** The low byte of the 16-bit Flash frequency value.

Note: The bit values used in Table 88 are set at the factory; no calibration is required.

Table 89. VBO Trim Definition

VBO_TRIM	Trigger Voltage Level
000	1.7
001	1.6
101	2.2
110	2.0
100	2.4
111	1.8

The F083A Series' on-chip Flash memory only guarantees write operations with a voltage supply over 2.7V. Write operations below 2.7V may cause unpredictable results.

Trim Bit Address 0006H

Table 90. Trim Option Bits at 0006H (TCLKFLT)

Bit	7	6	5	4	3	2	1	0
Field	DivBy4	Reserved	DlyCtl1	DlyCtl2	DlyCtl3	Reserved	FilterSel1	FilterSel0
RESET	0	1	0	0	0	1	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Address	Information Page Memory 0026H							

Note: U = Unchanged by Reset; R/W = Read/Write.

Bit	Description
[7] DivBy4	**Output Frequency Selection** 0 = Output frequency is input frequency. 1 = Output frequency is 1/4 of the input frequency.
[6]	**Reserved** This bit is reserved and must be programmed to 1.
[5:3] DlyCtl*x*	**Delay Control** Filtered 3-bit pulse width selection. For 3.3V operation, see Table 91.

Notes: x indicates bit values 3–1; y indicates bit values 1–0.

If the OCD receives a serial break (nine or more continuous bits low), the autobaud detector/generator resets. Reconfigure the autobaud detector/generator by sending 80H.

OCD Serial Errors

The On-Chip Debugger detects any of the following error conditions on the DBG pin:

- Serial break (a minimum of nine continuous bits Low)
- Framing error (received Stop bit is Low)
- Transmit collision (simultaneous transmission by OCD and host detected by the OCD)

When the OCD detects one of these errors, it aborts any command currently in progress, transmits a four character long serial break back to the host and resets the autobaud detector/generator. A framing error or transmit collision may be caused by the host sending a serial break to the OCD. As a result of the open-drain nature of the interface, returning a serial break back to the host only extends the length of the serial break if the host releases the serial break early.

The host transmits a serial break on the DBG pin when first connecting to the Z8 Encore! F083A Series devices or when recovering from an error. A serial break from the host resets the autobaud generator/detector, but does not reset the OCD Control Register. A serial break leaves the device in DEBUG Mode, if that is the current mode. The OCD is held in reset until the end of the serial break when the DBG pin returns high. Because of the open-drain nature of the DBG pin, the host sends a serial break to the OCD even if the OCD is transmitting a character.

Breakpoints

Execution breakpoints are generated using the BRK instruction (Opcode 00H). When the eZ8 CPU decodes a BRK instruction, it signals the On-Chip Debugger. If breakpoints are enabled, the OCD enters DEBUG Mode and idles the eZ8 CPU. If breakpoints are not enabled, the OCD ignores the BRK signal and the BRK instruction operates as an NOP instruction.

Breakpoints in Flash Memory

The BRK instruction is Opcode 00H, which corresponds to the fully programmed state of a byte in Flash memory. To implement a breakpoint, write 00H to the required break address overwriting the current instruction. To remove a breakpoint, the corresponding page of Flash memory must be erased and reprogrammed with the original data.

eZ8 CPU Instruction Set

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

Assembly Language Programming Introduction: see page 162

Assembly Language Syntax: see page 163

eZ8 CPU Instruction Notation: see page 164

eZ8 CPU Instruction Classes: see page 166

eZ8 CPU Instruction Summary: see page 171

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 (op codes and operands) to represent the instructions themselves. The op codes 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 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 given in the following example.

Table 114. eZ8 CPU Instruction Summary (Continued)

Assembly Mnemonic	Symbolic Operation	Address Mode dst	Address Mode src	Op Code(s) (Hex)	Flags C	Z	S	V	D	H	Fetch Cycles	Instr. Cycles
AND dst, src	dst ← dst AND src	r	r	52	–	*	*	0	–	–	2	3
		r	Ir	53							2	4
		R	R	54							3	3
		R	IR	55							3	4
		R	IM	56							3	3
		IR	IM	57							3	4
ANDX dst, src	dst ← dst AND src	ER	ER	58	–	*	*	0	–	–	4	3
		ER	IM	59							4	3
ATM	Block all interrupt and DMA requests during execution of the next 3 instructions			2F	–	–	–	–	–	–	1	2
BCLR bit, dst	dst[bit] ← 0	r		E2	–	*	*	0	–	–	2	2
BIT p, bit, dst	dst[bit] ← p	r		E2	–	*	*	0	–	–	2	2
BRK	Debugger Break			00	–	–	–	–	–	–	1	1
BSET bit, dst	dst[bit] ← 1	r		E2	–	*	*	0	–	–	2	2
BSWAP dst	dst[7:0] ← dst[0:7]	R		D5	X	*	*	0	–	–	2	2
BTJ p, bit, src, dst	if src[bit] = p PC ← PC + X		r	F6	–	–	–	–	–	–	3	3
			Ir	F7							3	4
BTJNZ bit, src, dst	if src[bit] = 1 PC ← PC + X		r	F6	–	–	–	–	–	–	3	3
			Ir	F7							3	4
BTJZ bit, src, dst	if src[bit] = 0 PC ← PC + X		r	F6	–	–	–	–	–	–	3	3
			Ir	F7							3	4

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

GPIO Port Output Timing

Figure 31 and Table 126 provide timing information for the GPIO port pins.

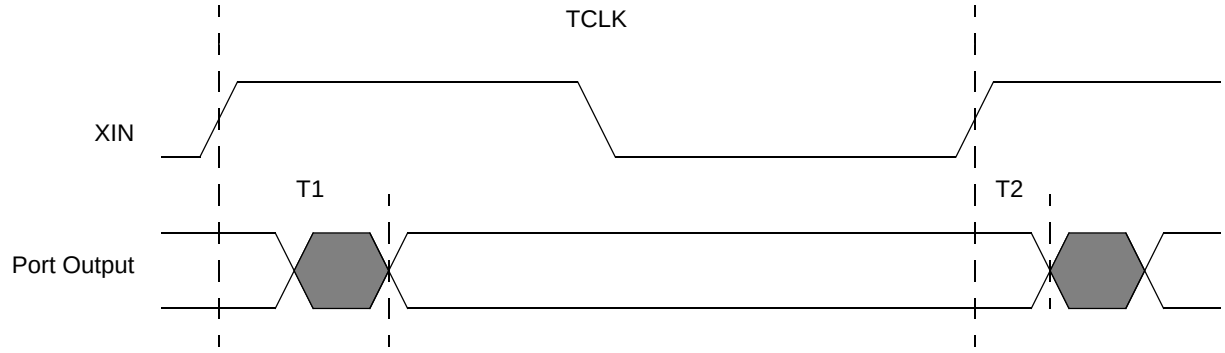

Figure 31. GPIO Port Output Timing

Table 126. GPIO Port Output Timing

		Delay (ns)	
Parameter	**Abbreviation**	**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	–

Hex Address: F09

Table 140. Timer 1 Low Byte Register (T1L)

Bit	7	6	5	4	3	2	1	0
Field	TL							
RESET	0	0	0	0	0	0	0	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Address	F09H							

Hex Address: F0A

Table 141. Timer 1 Reload High Byte Register (T1RH)

Bit	7	6	5	4	3	2	1	0
Field	TRH							
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	F0AH							

Hex Address: F0B

Table 142. Timer 1 Reload Low Byte Register (T1RL)

Bit	7	6	5	4	3	2	1	0
Field	TRL							
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	F0BH							

Hex Address: F0C

Table 143. Timer 1 PWM High Byte Register (T1PWMH)

Bit	7	6	5	4	3	2	1	0
Field	PWMH							
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	F0CH							

G

gated mode 89
general-purpose I/O 33
GPIO 4, 33
  alternate functions 34
  architecture 33
  control register definitions 39
  input data sample timing 194
  interrupts 39
  port A-C pull-up enable sub-registers 46, 47, 48
  port A-H address registers 40
  port A-H alternate function sub-registers 42
  port A-H control registers 41
  port A-H data direction sub-registers 41
  port A-H high drive enable sub-registers 44
  port A-H input data registers 49
  port A-H output control sub-registers 43
  port A-H output data registers 50, 51
  port A-H stop mode recovery sub-registers 45
  port availability by device 33
  port input timing 194
  port output timing 195

H

H 165
HALT 168
halt mode 31, 168
hexadecimal number prefix/suffix 165

I

I2C 4
IM 164
immediate data 164
immediate operand prefix 165
INC 166
increment 166
increment word 166
INCW 166
indexed 165
indirect address prefix 165
indirect register 164
indirect register pair 164
indirect working register 164
indirect working register pair 164
instruction set, ez8 CPU 162
instructions
  ADC 166
  ADCX 166
  ADD 166
  ADDX 166
  AND 169
  ANDX 169
  arithmetic 166
  BCLR 167
  BIT 167
  bit manipulation 167
  block transfer 167
  BRK 169
  BSET 167
  BSWAP 167, 170
  BTJ 169
  BTJNZ 166, 169
  BTJZ 169
  CALL 169
  CCF 167, 168
  CLR 168
  COM 169
  CP 166
  CPC 166
  CPCX 166
  CPU control 168
  CPX 166
  DA 166
  DEC 166
  DECW 166
  DI 168
  DJNZ 169
  EI 168
  HALT 168
  INC 166
  INCW 166
  IRET 169
  JP 169
  LD 168
  LDC 168