

Welcome to [E-XFL.COM](https://www.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	23
Program Memory Size	4KB (4K 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 8x10b
Oscillator Type	Internal
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	28-VQFN
Supplier Device Package	-
Purchase URL	<a href="https://www.e-xfl.com/product-detail/zilog/z8f0430qj020sg">https://www.e-xfl.com/product-detail/zilog/z8f0430qj020sg</a>

# **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 Input/Output .....	4
Flash Controller .....	4
Nonvolatile Data Storage .....	5
Internal Precision Oscillator .....	5
External Crystal Oscillator .....	5
10-Bit Analog-to-Digital Converter .....	5
Analog Comparator .....	5
Timers .....	5
Interrupt Controller .....	5
Reset Controller .....	6
On-Chip Debugger .....	6
Acronyms and Expansions .....	6
Pin Description .....	7
Available Packages .....	7
Pin Configurations .....	7
Signal Descriptions .....	11
Pin Characteristics .....	13
Address Space .....	14
Register File .....	14
Program Memory .....	15
Data Memory .....	16
Flash Information Area .....	16
Register Map .....	17
Reset and Stop Mode Recovery .....	21
Reset Types .....	21
Reset Sources .....	23
Power-On Reset .....	23

► **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			VBO	Reserved	Reserved	COMP	Reserved
RESET	1	0	0	0	1	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
[7:5]	<b>Reserved</b> These registers are reserved and must be programmed to 000.
[4] VBO	<b>Voltage Brown-Out detector disable</b> This bit takes only effect when the VBO_AO Flash option bit is disabled. In STOP Mode, VBO is always disabled when the VBO_AO Flash option bit is disabled. To learn more about the VBO_AO Flash option bit function, see the <a href="#">Flash Option Bits</a> chapter on page 124. 0 = VBO enabled. 1 = VBO disabled.
[3]	<b>Reserved</b> This bit is reserved and must be programmed to 1.
[2]	<b>Reserved</b> This bit is reserved and must be programmed to 0.
[1] COMP	<b>Comparator Disable</b> 0 = Comparator is enabled. 1 = Comparator is disabled.
[0]	<b>Reserved</b> This bit is reserved and must be programmed to 0.

# General Purpose Input/Output

The Z8 Encore! F0830 Series products support a maximum of 25 port pins (Ports A–D) for General Purpose Input/Output (GPIO) operations. Each port contains control and data registers. The GPIO control registers determine data direction, open-drain, output drive current, programmable pull-ups, Stop Mode Recovery functionality and alternate pin functions. Each port pin is individually programmable. In addition, the Port C pins are capable of direct LED drive at programmable drive strengths.

## GPIO Port Availability by Device

Table 15 lists the port pins available with each device and package type.

**Table 15. Port Availability by Device and Package Type**

Devices	Package	10-Bit ADC	Port A	Port B	Port C	Port D	Total I/O
Z8F1232, Z8F0830, Z8F0430, Z8F0230, Z8F0130	20-pin	Yes	[7:0]	[3:0]	[3:0]	[0]	17
Z8F1233, Z8F0831 Z8F0431, Z8F0231 Z8F0131	20-pin	No	[7:0]	[3:0]	[3:0]	[0]	17
Z8F1232, Z8F0830, Z8F0430, Z8F0230, Z8F0130	28-pin	Yes	[7:0]	[5:0]	[7:0]	[0]	23
Z8F1233, Z8F0831 Z8F0431, Z8F0231 Z8F0131	28-pin	No	[7:0]	[7:0]	[7:0]	[0]	25

Note: 20-pin and 28-pin and 10-bit ADC Enabled or Disabled can be selected via the option bits.

Table 16. Port Alternate Function Mapping (Continued)

Port	Pin	Mnemonic	Alternate Function Description	Alternate Function Set Register AFS1
Port B <sup>2</sup>	PB0	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 input clock	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 <sub>REF</sub>	ADC reference voltage	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 and Port D (PD0) pin, the Alternate Function Set registers are not implemented for Port A and Port D (PD0). Enabling alternate function selections (as described in the [Port A–D Alternate Function Subregisters](#) section on page 42) automatically enables the associated alternate function.
2. Because there are at most two choices of alternate functions for any Port B pin, the AFS2 Alternate Function Set Register is implemented but is not used to select the function. Additionally, alternate function selection (as described in the [Port A–D Alternate Function Subregisters](#) section on page 42) must also be enabled.
3. Because there are at most two choices of alternate functions for any Port C pin, the AFS2 Alternate Function Set Register is implemented but is not used to select the function. Additionally, alternate function selection (as described in the [Port A–D Alternate Function Subregisters](#) section on page 42) must also be enabled.

## GPIO Interrupts

Many of the GPIO port pins can be used as interrupt sources. Some port pins can be configured to generate an interrupt request on either the rising edge or falling edge of the input pin signal. Other port pin interrupt sources, generate an interrupt when any edge occurs (both rising and falling). See the [Interrupt Controller](#) chapter on page 53 for more information about interrupts using the GPIO pins.

## GPIO Control Register Definitions

Four registers for each port provide access to GPIO control, input data and output data; Table 17 lists these port registers. Use the Port A–D Address and Control registers together to provide access to subregisters for port configuration and control.

**Table 17. GPIO Port Registers and Subregisters**

Port Register Mnemonic	Port Register Name
PxADDR	Port A–D Address Register (selects subregisters)
PxCTL	Port A–D Control Register (provides access to subregisters)
PxIN	Port A–D Input Data Register
PxOUT	Port A–D Output Data Register
Port Subregister Mnemonic	Port Register Name
PxDD	Data Direction
PxAF	Alternate Function
PxOC	Output Control (open-drain)
PxHDE	High Drive Enable
PxSMRE	Stop Mode Recovery Source Enable
PxPUE	Pull-Up Enable
PxAFS1	Alternate Function Set 1
PxAFS2	Alternate Function Set 2

### Port A–D Pull-up Enable Subregisters

The Port A–D Pull-Up Enable Subregister is accessed through the Port A–D Control Register by writing 06H to the Port A–D Address Register. See Table 26. Setting the bits in the Port A–D Pull-Up Enable subregisters enables a weak internal resistive pull-up on the specified port pins.

**Table 26. Port A–D Pull-Up Enable Subregisters (PxPUE)**

Bit	7	6	5	4	3	2	1	0
Field	PPUE7	PPUE6	PPUE5	PPUE4	PPUE3	PPUE2	PPUE1	PPUE0
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 06H in Port A–D Address Register, accessible through the Port A–D Control Register							

Bit	Description
[7:0]	<b>Port Pull-Up Enable</b>
PxPUE	0 = The weak pull-up on the port pin is disabled. 1 = The weak pull-up on the port pin is enabled.

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

## Port A–D Alternate Function Set 2 Subregisters

The Port A–D Alternate Function Set 2 Subregister, shown in Table 28, is accessed through the Port A–D Control Register by writing 08H to the Port A–D Address Register. The Alternate Function Set 2 subregisters select the alternate function available at a port pin. Alternate functions selected by setting or clearing bits in this register are defined in Table 16 in the [GPIO Alternate Functions](#) section on page 34.

► **Note:** Alternate function selection on the port pins must also be enabled, as described in the [Port A–D Alternate Function Subregisters](#) section on page 42.

**Table 28. Port A–D Alternate Function Set 2 Subregisters (PxAFS2)**

Bit	7	6	5	4	3	2	1	0
Field	PAFS27	PAFS26	PAFS25	PAFS24	PAFS23	PAFS22	PAFS21	PAFS20
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 08H in Port A–D Address Register, accessible through the Port A–D Control Register							

Bit	Description
[7:0]	<b>Port Alternate Function Set 2</b>
PAFS2x	0 = The Port Alternate function is selected, as defined in Table 16 in the <a href="#">GPIO Alternate Functions</a> section on page 34. 1 = The Port Alternate function is selected, as defined in Table 16 in the <a href="#">GPIO Alternate Functions</a> section on page 34.

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

## IRQ1 Enable High and Low Bit Registers

Table 41 describes the priority control for IRQ1. The IRQ1 Enable High and Low Bit registers, shown in Tables 42 and 43, form a priority-encoded enabling service for interrupts in the Interrupt Request 1 Register. Priority is generated by setting the bits in each register.

**Table 41. IRQ1 Enable and Priority Encoding**

IRQ1ENH[x]	IRQ1ENL[x]	Priority	Description
0	0	Disabled	Disabled
0	1	Level 1	Low
1	0	Level 2	Nominal
1	1	Level 3	High

Note: x indicates register bits in the address range 7–0.

**Table 42. IRQ1 Enable High Bit Register (IRQ1ENH)**

Bit	7	6	5	4	3	2	1	0
Field	PA7ENH	PA6CENH	PA5ENH	PA4ENH	PA3ENH	PA2ENH	PA1ENH	PA0ENH
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	FC4H							

Bit	Description
[7] PA7ENH	Port A Bit[7] Interrupt Request Enable High Bit
[6] PA6CENH	Port A Bit[7] or Comparator Interrupt Request Enable High Bit
[5:0] PAxENH	Port A Bit[x] Interrupt Request Enable High Bit See the interrupt port select register for selection of either Port A or Port D as the interrupt source.

Note: x indicates register bits in the address range 5–0.

# Comparator

The Z8 Encore! F0830 Series devices feature a general purpose comparator that compares two analog input signals. A GPIO (CINP) pin provides the positive comparator input. The negative input (CINN) can be taken from either an external GPIO pin or from an internal reference. The output is available as an interrupt source or can be routed to an external pin using the GPIO multiplex. The comparator includes the following features:

- Positive input is connected to a GPIO pin
- Negative input can be connected to either a GPIO pin or a programmable internal reference
- Output can be either an interrupt source or an output to an external pin

## Operation

One of the comparator inputs can be connected to an internal reference that is a user-selectable reference and is user-programmable with 200mV resolution.

The comparator can be powered down to save supply current. For details, see the [Power Control Register 0](#) section on page 31.

---

**! Caution:** As a result of the propagation delay of the comparator, Zilog does not recommend enabling the comparator without first disabling interrupts and waiting for the comparator output to settle. This delay prevents spurious interrupts after comparator enabling.

---

The following example shows how to safely enable the comparator:

```
di
ld cmp0,r0; load some new configuration
nop
nop      ; wait for output to settle
clr irq0 ; clear any spurious interrupts pending
ei
```

Flash information area is mapped into program memory and overlays the 128 bytes in the address range FE00H to FE7FH. When the information area access is enabled, all reads from these program memory addresses return the information area data rather than the program memory data. Access to the Flash information area is read-only.

The trim bits are handled differently than the other Zilog Flash option bits. The trim bits are the hybrid of the user option bits and the standard Zilog option bits. These trim bits must be user-accessible for reading at all times using external registers regardless of the state of bit 7 in the Flash Page Select Register. Writes to the trim space change the value of the Option Bit Holding Register but do not affect the Flash bits, which remain as read-only.

**Table 70. Z8F083 Flash Memory Area Map**

<b>Program Memory Address (Hex)</b>	<b>Function</b>
FE00–FE3F	Zilog option bits
FE40–FE53	Part number 20-character ASCII alphanumeric code Left justified and filled with FH
FE54–FE5F	Reserved
FE60–FE7F	Reserved

## Operation

The Flash Controller programs and erases Flash memory. The Flash Controller provides the proper Flash controls and timing for byte programming, page erase and mass erase of Flash memory.

The Flash Controller contains several protection mechanisms to prevent accidental programming or erasure. These mechanism operate on the page, sector and full-memory levels.

The flowchart in Figure 19 display basic Flash Controller operation. The following subsections provide details about the various operations (Lock, Unlock, Byte Programming, Page Protect, Page Unprotect, Page Select Page Erase and Mass Erase) displayed in Figure 19.

**Table 71. Flash Code Protection using the Flash Option Bits**

FHSWP	FWP	Flash Code Protection Description
0	0	Programming and erasing disabled for all Flash program memory. In user code programming, page erase and mass erase are all disabled. Mass erase is available through the On-Chip Debugger.
0 or 1	1	Programming, page erase and mass erase are enabled for all of the Flash program memory.

At reset, the Flash Controller is locked to prevent accidental program or erasure of Flash memory. To program or erase Flash memory, first write the target page to the page select register. Unlock the Flash Controller by making two consecutive writes to the Flash Control Register with the values 73H and 8CH, sequentially. The page select register must be rewritten with the same page previously stored there. If the two page select writes do not match, the controller reverts to a Locked state. If the two writes match, the selected page becomes active. See Figure 19 for details.

After unlocking a specific page, you can enable either page program or erase. Writing the value 95H causes a page erase only if the active page resides in a sector that is not protected. Any other value written to the Flash Control Register locks the Flash Controller. Mass erase is not allowed in the user code, but is allowed through the debug port.

After unlocking a specific page, the user can also write to any byte on that page. After a byte is written, the page remains unlocked, allowing for subsequent writes to other bytes on the same page. Further writes to the Flash Control Register causes the active page to revert to a Locked state.

### Sector Based Flash Protection

The final protection mechanism is implemented on a per-sector basis. The Flash memories of Z8 Encore! devices are divided into maximum number of eight sectors. A sector is one-eighth of the total size of Flash memory, unless this value is smaller than the page size, in which case the sector and page sizes are equal. On Z8 Encore! F0830 Series devices, the sector size is varied according to the Z8 Encore! F0830 Series Flash Memory Configuration shown in [Table 69](#) on page 108 and in Figures 14 through 18, which follow the table

The Flash Sector Protect Register can be configured to prevent sectors from being programmed or erased. After a sector is protected, it cannot be unprotected by user code. The Flash Sector Protect Register is cleared after reset and any previously written protection values is lost. User code must write this register in their initialization routine if they want to enable sector protection.

The Flash Sector Protect Register shares its Register File address with the Page Select Register. The Flash Sector Protect Register is accessed by writing the Flash Control Register with 5EH. After the Flash Sector Protect Register is selected, it can be accessed at the Page Select Register address. When user code writes the Flash Sector Protect Register,

# Nonvolatile Data Storage

Z8 Encore! F0830 Series devices contain a Nonvolatile Data Storage (NVDS) element of up to 64 bytes (except when in Flash 12KB mode). This type of memory can perform over 100,000 write cycles.

## Operation

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

---

► **Note:** The products in the Z8 Encore! F0830 Series feature multiple NVDS array sizes. See the Z8 Encore! F0830 Series Family Part Selection Guide section on page 2 for details.

---

## 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 predefined address outside of program memory that is accessible to the user. Both the NVDS address and data are single-byte values. In order to not disturb the user code, these routines save the working register set before using it so that 16 bytes of stack space are required to preserve the site. After finishing the call to these routines, the working register set of the user code is recovered.

During both read and write accesses to the NVDS, interrupt service is not disabled. Any interrupts that occur during NVDS execution must not disturb the working register and existing stack contents; otherwise, the array can become corrupted. Zilog recommends the user disable interrupts before executing NVDS operations.

Use of the NVDS requires 16 bytes of available stack space. The contents of the working register set are saved before calling NVDS read or write routines.

For correct NVDS operation, the Flash Frequency registers must be programmed based on the system clock frequency. See *the Flash Operation Timing Using the Flash Frequency Registers section on page 114.*

Because the minimum read time is much less than the write time, however, actual speed benefits are not always realized.

2. Use as few unique addresses as possible to optimize the impact of refreshing.

**Table 98. Oscillator Configuration and Selection**

Clock Source	Characteristics	Required Setup
Internal precision RC oscillator	<ul style="list-style-type: none"> <li>32.8 kHz or 5.53MHz</li> <li>± 4% accuracy when trimmed</li> <li>No external components required</li> </ul>	<ul style="list-style-type: none"> <li>Unlock and write to the Oscillator Control Register (OSCCTL) to enable and select oscillator at either 5.53MHz or 32.8 kHz</li> </ul>
External crystal/resonator	<ul style="list-style-type: none"> <li>32 kHz to 20MHz</li> <li>Very high accuracy (dependent on crystal or resonator used)</li> <li>Requires external components</li> </ul>	<ul style="list-style-type: none"> <li>Configure Flash option bits for correct external OSCILLATOR Mode</li> <li>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)</li> </ul>
External RC oscillator	<ul style="list-style-type: none"> <li>32 kHz to 4MHz</li> <li>Accuracy dependent on external components</li> </ul>	<ul style="list-style-type: none"> <li>Configure Flash option bits for correct external OSCILLATOR Mode</li> <li>Unlock and write OSCCTL to enable crystal oscillator and select as system clock</li> </ul>
External clock drive	<ul style="list-style-type: none"> <li>0 to 20MHz</li> <li>Accuracy dependent on external clock source</li> </ul>	<ul style="list-style-type: none"> <li>Write GPIO registers to configure PB3 pin for external clock function</li> <li>Unlock and write OSCCTL to select external system clock</li> <li>Apply external clock signal to GPIO</li> </ul>
Internal Watchdog Timer Oscillator	<ul style="list-style-type: none"> <li>10 kHz nominal</li> <li>± 40% accuracy; no external components required</li> <li>Low power consumption</li> </ul>	<ul style="list-style-type: none"> <li>Enable WDT if not enabled and wait until WDT oscillator is operating.</li> <li>Unlock and write to the Oscillator Control Register (OSCCTL) to enable and select oscillator</li> </ul>

**! Caution:** Unintentional accesses to the Oscillator Control Register can actually stop the chip by switching to a nonfunctioning 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 have 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.

When selecting a new clock source, the primary oscillator failure detection circuitry and the Watchdog Timer Oscillator failure circuitry must be disabled. If POFEN and WOFEN are not disabled prior to a clock switch-over, it is possible to generate an interrupt for a failure of either oscillator. The failure detection circuitry can be enabled anytime after a successful write of OSCSEL in the Oscillator Control Register.

The Internal Precision Oscillator is enabled by default. If the user code changes to a different oscillator, it may be appropriate to disable the IPO for power savings. Disabling the IPO does not occur automatically.

## Clock Failure Detection and Recovery

### Primary Oscillator Failure

The Z8F04xA family devices can generate nonmaskable interrupt-like events when the primary oscillator fails. To maintain system function in this situation, the clock failure recovery circuitry automatically forces the Watchdog Timer Oscillator to drive the system clock. The Watchdog Timer Oscillator must be enabled to allow the recovery. Although this oscillator runs at a much slower speed than the original system clock, the CPU continues to operate, allowing execution of a clock failure vector and software routines that either remedy the oscillator failure or issue a failure alert. This automatic switch-over is not available if the Watchdog Timer is the primary oscillator. It is also unavailable if the Watchdog Timer Oscillator is disabled, though it is not necessary to enable the Watchdog Timer reset function outlined in the Watchdog Timer chapter of this document.

The primary oscillator failure detection circuitry asserts if the system clock frequency drops below 1 KHz  $\pm 50\%$ . If an external signal is selected as the system oscillator, it is possible that a very slow but nonfailing clock can generate a failure condition. Under these conditions, do not enable the clock failure circuitry (POFEN must be deasserted in the OSCCTL Register).

### Watchdog Timer Failure

In the event of failure of a Watchdog Timer Oscillator, a similar nonmaskable interrupt-like event is issued. This event does not trigger an attendant clock switch-over, but alerts the CPU of the failure. After a Watchdog Timer failure, it is no longer possible to detect a primary oscillator failure. The failure detection circuitry does not function if the Watchdog Timer is used as the primary oscillator or if the Watchdog Timer Oscillator has been disabled. For either of these cases, it is necessary to disable the detection circuitry by deasserting the WDFEN bit of the OSCCTL Register.

The Watchdog Timer Oscillator failure detection circuit counts system clocks while looking for a Watchdog Timer clock. The logic counts 8004 system clock cycles before determining that a failure has occurred. The system clock rate determines the speed at which the Watchdog Timer failure is detected. A very slow system clock results in very slow detection times.

### Example 2

In general, when an instruction format requires an 8-bit register address, the 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 102. Assembly Language Syntax Example 2**

<b>Assembly Language Code</b>	ADD	43H,	R8	(ADD dst, src)
<b>Object Code</b>	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 the notational shorthand listed in Table 103.

**Table 103. Notational Shorthand**

Notation	Description	Operand	Range
b	Bit	b	b represents a value from 0 to 7 (000B to 111B).
cc	Condition Code	—	See condition codes overview in the eZ8 CPU User Manual.
DA	Direct Address	Addr	Addr. represents a number in the range of 0000H to FFFFH
ER	Extended Addressing Register	Reg	Reg. represents a number in the range of 000H to FFFH
IM	Immediate Data	#Data	Data is a number between 00H to FFH
Ir	Indirect Working Register	@Rn	n = 0 –15
IR	Indirect Register	@Reg	Reg. represents a number in the range of 00H to FFH
Irr	Indirect Working Register Pair	@RRp	p = 0, 2, 4, 6, 8, 10, 12 or 14
IRR	Indirect Register Pair	@Reg	Reg. represents an even number in the range 00H to FEH
p	Polarity	p	Polarity is a single bit binary value of either 0B or 1B.
r	Working Register	Rn	n = 0 – 15

Table 113. eZ8 CPU Instruction Summary (Continued)

Assembly Mnemonic	Symbolic Operation	Address Mode		Op Code(s) (Hex)	Flags						Fetch Cycles	Instr. Cycles
		dst	src		C	Z	S	V	D	H		
AND dst, src	$\text{dst} \leftarrow \text{dst AND src}$	r	r	52	–	*	*	0	–	–	2	3
		r	lr	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	$\text{dst} \leftarrow \text{dst AND src}$	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	$\text{dst}[\text{bit}] \leftarrow 0$	r		E2	–	*	*	0	–	–	2	2
BIT p, bit, dst	$\text{dst}[\text{bit}] \leftarrow \text{p}$	r		E2	–	*	*	0	–	–	2	2
BRK	Debugger Break			00	–	–	–	–	–	–	1	1
BSET bit, dst	$\text{dst}[\text{bit}] \leftarrow 1$	r		E2	–	*	*	0	–	–	2	2
BSWAP dst	$\text{dst}[7:0] \leftarrow \text{dst}[0:7]$	R		D5	X	*	*	0	–	–	2	2
BTJ p, bit, src, dst	if $\text{src}[\text{bit}] = \text{p}$ $\text{PC} \leftarrow \text{PC} + \text{X}$		r	F6	–	–	–	–	–	–	3	3
			lr	F7							3	4
BTJNZ bit, src, dst	if $\text{src}[\text{bit}] = 1$ $\text{PC} \leftarrow \text{PC} + \text{X}$		r	F6	–	–	–	–	–	–	3	3
			lr	F7							3	4
BTJZ bit, src, dst	if $\text{src}[\text{bit}] = 0$ $\text{PC} \leftarrow \text{PC} + \text{X}$		r	F6	–	–	–	–	–	–	3	3
			lr	F7							3	4
CALL dst	$\text{SP} \leftarrow \text{SP} - 2$	IRR		D4	–	–	–	–	–	–	2	6
	$@\text{SP} \leftarrow \text{PC}$ $\text{PC} \leftarrow \text{dst}$	DA		D6							3	3
CCF	$\text{C} \leftarrow \sim \text{C}$			EF	*	–	–	–	–	–	1	2

Note: Flags Notation:

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

– = Unaffected.

X = Undefined.

0 = Reset to 0.

1 = Set to 1.

Table 113. eZ8 CPU Instruction Summary (Continued)

Assembly Mnemonic	Symbolic Operation	Address Mode		Op Code(s) (Hex)	Flags						Fetch Cycles	Instr. Cycles
		dst	src		C	Z	S	V	D	H		
CLR dst	dst ← 00H	R		B0	–	–	–	–	–	–	2	2
		IR		B1							2	3
COM dst	dst ← ~dst	R		60	–	*	*	0	–	–	2	2
		IR		61							2	3
CP dst, src	dst - src	r	r	A2	*	*	*	*	–	–	2	3
		r	lr	A3							2	4
		R	R	A4							3	3
		R	IR	A5							3	4
		R	IM	A6							3	3
		IR	IM	A7							3	4
CPC dst, src	dst - src - C	r	r	1F A2	*	*	*	*	–	–	3	3
		r	lr	1F A3							3	4
		R	R	1F A4							4	3
		R	IR	1F A5							4	4
		R	IM	1F A6							4	3
		IR	IM	1F A7							4	4
CPCX dst, src	dst - src - C	ER	ER	1F A8	*	*	*	*	–	–	5	3
		ER	IM	1F A9							5	3
CPX dst, src	dst - src	ER	ER	A8	*	*	*	*	–	–	4	3
		ER	IM	A9							4	3
DA dst	dst ← DA(dst)	R		40	*	*	*	X	–	–	2	2
		IR		41							2	3
DEC dst	dst ← dst - 1	R		30	–	*	*	*	–	–	2	2
		IR		31							2	3
DECW dst	dst ← dst - 1	RR		80	–	*	*	*	–	–	2	5
		IRR		81							2	6
DI	IRQCTL[7] ← 0			8F	–	–	–	–	–	–	1	2

Note: Flags Notation:

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

– = Unaffected.

X = Undefined.

0 = Reset to 0.

1 = Set to 1.

General Purpose I/O Port Output Timing

Figure 34 and Table 125 provide timing information for the GPIO port pins.

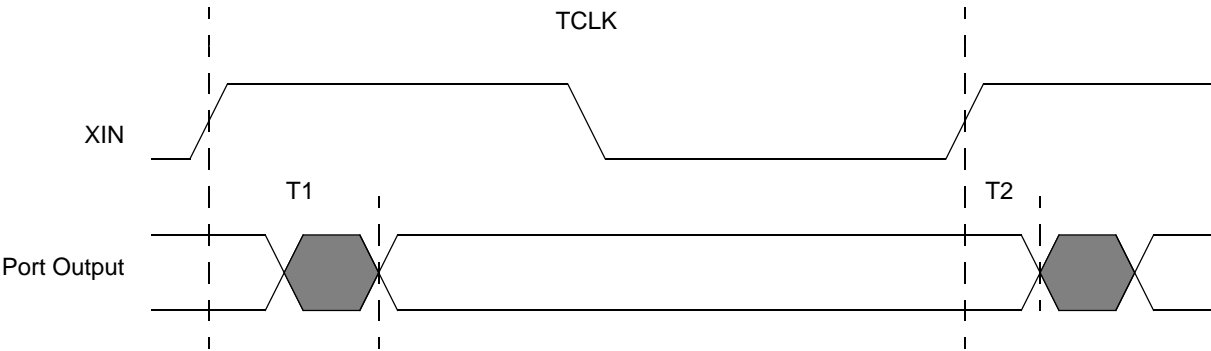


Figure 34. GPIO Port Output Timing

Table 125. GPIO Port Output Timing

Parameter	Abbreviation	Delay (ns)	
		Minimum	Maximum
GPIO Port Pins			
T <sub>1</sub>	XIN Rise to Port Output Valid Delay	–	15
T <sub>2</sub>	XIN Rise to Port Output Hold Time	2	–

**Hex Address: F05**

**Table 135. Timer 0 PWM Low Byte Register (T0PWML)**

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							

**Hex Address: F06**

**Table 136. Timer 0 Control Register 0 (T0CTL0)**

Bit	7	6	5	4	3	2	1	0
Field	TMODEHI	TICONFIG		Reserved	PWMD			INPCAP
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	F06H							

**Hex Address: F07**

**Table 137. Timer 0 Control Register 1 (T0CTL1)**

Bit	7	6	5	4	3	2	1	0
Field	TEN	TPOL	PRES			TMODE		
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	F07H							

**Hex Address: F08**

**Table 138. Timer 1 High Byte Register (T1H)**

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	F08H							