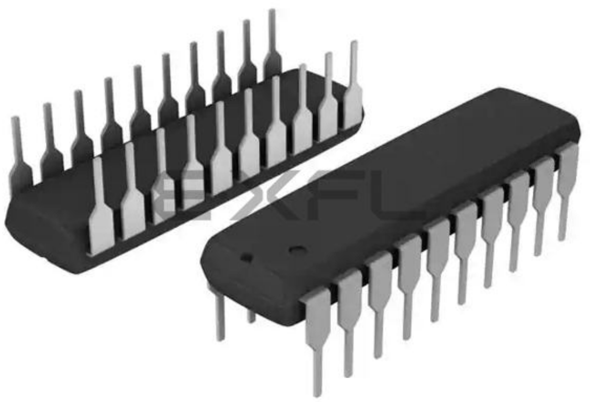

Welcome to E-XFL.COM

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	Through Hole
Package / Case	20-DIP (0.300", 7.62mm)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f0813ph005sg

returns FFH. Writing to these unimplemented Program Memory addresses produces no effect. Table 6 describes the Program Memory maps for the Z8 Encore! XP F0823 Series products.

Table 6. Z8 Encore! XP F0823 Series Program Memory Maps

Program Memory Address (Hex)	Function
Z8F0823 and Z8F0813 Products	
0000–0001	Flash Option Bits
0002–0003	Reset Vector
0004–0005	WDT Interrupt Vector
0006–0007	Illegal Instruction Trap
0008–0037	Interrupt Vectors*
0038–003D	Oscillator Fail Traps*
003E–0FFF	Program Memory
Z8F0423 and Z8F0413 Products	
0000–0001	Flash Option Bits
0002–0003	Reset Vector
0004–0005	WDT Interrupt Vector
0006–0007	Illegal Instruction Trap
0008–0037	Interrupt Vectors*
0038–003D	Oscillator Fail Traps*
003E–0FFF	Program Memory
Z8F0223 and Z8F0213 Products	
0000–0001	Flash Option Bits
0002–0003	Reset Vector
0004–0005	WDT Interrupt Vector
0006–0007	Illegal Instruction Trap
0008–0037	Interrupt Vectors*
0038–003D	Oscillator Fail Traps*
003E–07FF	Program Memory

Note: *See the Trap and Interrupt Vectors in Order of Priority section on page 55 for a list of the interrupt vectors and traps.

Table 6. Z8 Encore! XP F0823 Series Program Memory Maps (Continued)

Program Memory Address (Hex)	Function
Z8F0123 and Z8F0113 Products	
0000–0001	Flash Option Bits
0002–0003	Reset Vector
0004–0005	WDT Interrupt Vector
0006–0007	Illegal Instruction Trap
0008–0037	Interrupt Vectors*
0038–003D	Oscillator Fail Traps*
003E–03FF	Program Memory

Note: *See the Trap and Interrupt Vectors in Order of Priority section on page 55 for a list of the interrupt vectors and traps.

Data Memory

Z8 Encore! XP F0823 Series does not use the eZ8 CPU's 64KB Data Memory address space.

Flash Information Area

Table 7 lists the F0823 Series Flash Information Area. This 128B Information Area is accessed by setting bit 7 of the Flash Page Select Register to 1. When access is enabled, the Flash Information Area is mapped into the Program Memory and overlays the 128 bytes at addresses `FE00H` to `FF7FH`. 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.

Table 7. F0823 Series Flash Memory Information Area Map

Program Memory Address (Hex)	Function
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	Zilog Calibration Data.
FE80–FFFF	Reserved.

PA0 and PA6 contain two different timer functions, a timer input and a complementary timer output. Both of these functions require the same GPIO configuration, the selection between the two is based on the timer mode. For more details, see the Timers chapter on page 69.

! **Caution:** For pins with multiple alternate functions, Zilog recommends writing to the AFS1 and AFS2 subregisters before enabling the alternate function via the AF Subregister. This prevents spurious transitions through unwanted alternate function modes.

Table 16. Port Alternate Function Mapping (8-Pin Parts)

Port	Pin	Mnemonic	Alternate Function Description	Alternate Function Select Register AFS1	Alternate Function Select Register AFS2
Port A	PA0	T0IN	Timer 0 Input	AFS1[0]: 0	AFS2[0]: 0
		Reserved		AFS1[0]: 0	AFS2[0]: 1
		Reserved		AFS1[0]: 1	AFS2[0]: 0
		T0OUT	Timer 0 Output Complement	AFS1[0]: 1	AFS2[0]: 1
	PA1	T0OUT	Timer 0 Output	AFS1[1]: 0	AFS2[1]: 0
		Reserved		AFS1[1]: 0	AFS2[1]: 1
		CLKIN	External Clock Input	AFS1[1]: 1	AFS2[1]: 0
		Analog Functions*	ADC Analog Input/V_{REF}	AFS1[1]: 1	AFS2[1]: 1
	PA2	$\overline{DE0}$	UART 0 Driver Enable	AFS1[2]: 0	AFS2[2]: 0
		$\overline{RESET}$	External Reset	AFS1[2]: 0	AFS2[2]: 1
		T1OUT	Timer 1 Output	AFS1[2]: 1	AFS2[2]: 0
		Reserved		AFS1[2]: 1	AFS2[2]: 1
	PA3	$\overline{CTS0}$	UART 0 Clear to Send	AFS1[3]: 0	AFS2[3]: 0
		COUT	Comparator Output	AFS1[3]: 0	AFS2[3]: 1
		T1IN	Timer 1 Input	AFS1[3]: 1	AFS2[3]: 0
		Analog Functions*	ADC Analog Input	AFS1[3]: 1	AFS2[3]: 1
	PA4	RXD0	UART 0 Receive Data	AFS1[4]: 0	AFS2[4]: 0
		Reserved		AFS1[4]: 0	AFS2[4]: 1
		Reserved		AFS1[4]: 1	AFS2[4]: 0
		Analog Functions*	ADC/Comparator Input (N)	AFS1[4]: 1	AFS2[4]: 1
	PA5	TXD0	UART 0 Transmit Data	AFS1[5]: 0	AFS2[5]: 0
		T1OUT	Timer 1 Output Complement	AFS1[5]: 0	AFS2[5]: 1
		Reserved		AFS1[5]: 1	AFS2[5]: 0
		Analog Functions*	ADC/Comparator Input (P)	AFS1[5]: 1	AFS2[5]: 1

Note: *Analog Functions include ADC inputs, ADC reference and comparator inputs. Also, alternate function selection as described in the Port A–C Alternate Function Subregisters section on page 43 must be enabled.

Table 43. IRQ1 Enable High Bit Register (IRQ1ENH)

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	PA7VENH	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] PA7VENH	**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** For selection of Port A as the interrupt source, see the Shared Interrupt Select Register section on page 67.
Note: x indicates the specific GPIO Port A pin number (5–0).	

Table 44. IRQ1 Enable Low Bit Register (IRQ1ENL)

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	PA7VENL	PA6CENL	PA5ENL	PA4ENL	PA3ENL	PA2ENL	PA1ENL	PA0ENL
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	FC5H							

Bit	**Description**
[7] PA7VENL	**Port A Bit[7] Interrupt Request Enable Low Bit**
[6] PA6CENL	**Port A Bit[7] or Comparator Interrupt Request Enable Low Bit**
[5:0] PAxENL	**Port A Bit[*x*] Interrupt Request Enable Low Bit**
Note: x indicates the specific GPIO Port A pin number (5–0).	

1. Write to the Timer Control Register to:
   - Disable the timer
   - Configure the timer for CONTINUOUS Mode
   - Set the prescale value
   - If using the Timer Output alternate function, set the initial output level (High or Low)
2. Write to the Timer High and Low Byte registers to set the starting count value (usually `0001H`). This action only affects the first pass in CONTINUOUS Mode. After the first timer reload in CONTINUOUS Mode, counting always begins at the reset value of `0001H`.
3. Write to the Timer Reload High and Low Byte registers to set the reload value.
4. Enable the timer interrupt (if appropriate) and set the timer interrupt priority by writing to the relevant interrupt registers.
5. Configure the associated GPIO port pin (if using the Timer Output function) for the Timer Output alternate function.
6. Write to the Timer Control Register to enable the timer and initiate counting.

In CONTINUOUS Mode, the system clock always provides the timer input. The timer period is computed via the following equation:

$$\text{CONTINUOUS Mode Time-Out Period (s)} = \frac{\text{Reload Value} \times \text{Prescale}}{\text{System Clock Frequency (Hz)}}$$

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

COUNTER Mode

In COUNTER Mode, the timer counts input transitions from a GPIO port pin. The timer input is taken from the GPIO port pin Timer Input alternate function. The `TPOL` bit in the Timer Control Register selects whether the count occurs on the rising edge or the falling edge of the timer input signal. In COUNTER Mode, the prescaler is disabled.

! Caution: The input frequency of the timer input signal must not exceed one-fourth the system clock frequency.

Upon reaching the reload value stored in the Timer Reload High and Low Byte registers, the timer generates an interrupt, the count value in the Timer High and Low Byte registers is reset to `0001H` and counting resumes. Also, if the Timer Output alternate function is input capture and reload events. If appropriate, configure the timer interrupt to be generated only at the input capture event or the reload event by setting TICONFIG field of the TxCTL1 Register.

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

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

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

CAPTURE RESTART Mode

In CAPTURE RESTART Mode, the current timer count value is recorded when the acceptable external Timer Input transition occurs. The capture count value is written to the Timer PWM High and Low Byte registers. The timer input is the system clock. The `TPOL` bit in the Timer Control Register determines if the capture occurs on a rising edge or a falling edge of the Timer Input signal. When the capture event occurs, an interrupt is generated and the count value in the Timer High and Low Byte registers is reset to `0001H` and counting resumes. The `INPCAP` bit in TxCTL1 Register is set to indicate the timer interrupt is because of an input capture event.

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

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

1. Write to the Timer Control Register to:
   – Disable the timer
   – Configure the timer for CAPTURE RESTART Mode; setting the mode also involves writing to `TMODEHI` bit in TxCTL1 Register
   – Set the prescale value
   – Set the capture edge (rising or falling) for the Timer Input
2. Write to the Timer High and Low Byte registers to set the starting count value (typically `0001H`).
3. Write to the Timer Reload High and Low Byte registers to set the reload value.

two bits to configure timer interrupt definition, and a status bit to identify if the most recent timer interrupt is caused by an input capture event.

Table 57. Timer 0–1 Control Register 0 (TxCTL0)

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, F0EH							

Bit	Description
[7] TMODEHI	**Timer Mode High Bit** This bit along with the TMODE field in TxCTL1 Register determines the operating mode of the timer. This is the most-significant bit of the Timer mode selection value.
[6:5] TICONFIG	**Timer Interrupt Configuration** This field configures timer interrupt definition. 0x = Timer Interrupt occurs on all defined reload, compare and input events. 10 = Timer Interrupt only on defined input capture/deassertion events. 11 = Timer Interrupt only on defined reload/compare events.
[4]	**Reserved** This bit is reserved and must be programmed to 0.
[3:1] PWMD	**PWMD—PWM Delay value** This field is a programmable delay to control the number of system clock cycles delay before the Timer Output and the Timer Output Complement are forced to their active state. 000 = No delay. 001 = 2 cycles delay. 010 = 4 cycles delay. 011 = 8 cycles delay. 100 = 16 cycles delay. 101 = 32 cycles delay. 110 = 64 cycles delay. 111 = 128 cycles delay.
[0] INPCAP	**Input Capture Event** This bit indicates if the most recent timer interrupt is caused by a Timer Input capture event. 0 = Previous timer interrupt is not a result of Timer Input capture event. 1 = Previous timer interrupt is a result of Timer Input capture event.

Timer 0–1 Control Register 1

The Timer 0–1 Control (TxCTL1) registers enable/disable the timers, set the prescaler value, and determine the timer operating mode.

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

Transmitter Interrupts

The transmitter generates a single interrupt when the Transmit Data Register Empty bit (`TDRE`) is set to 1. This indicates that the transmitter is ready to accept new data for transmission. The TDRE interrupt occurs after the Transmit shift register has shifted the first bit of data out. The Transmit Data Register can now be written with the next character to send. This action provides 7 bit periods of latency to load the Transmit Data Register before the Transmit shift register completes shifting the current character. Writing to the UART Transmit Data Register clears the `TDRE` bit to 0.

Receiver Interrupts

The receiver generates an interrupt when any of the following occurs:

- A data byte is received and is available in the UART Receive Data Register. This interrupt can be disabled independently of the other receiver interrupt sources. The received data interrupt occurs after the receive character has been received and placed in the Receive Data Register. To avoid an overrun error, software must respond to this received data available condition before the next character is completely received.

Note: In MULTIPROCESSOR Mode (`MPEN` = 1), the receive data interrupts are dependent on the multiprocessor configuration and the most recent address byte.

- A break is received
- An overrun is detected
- A data framing error is detected

UART Overrun Errors

When an overrun error condition occurs the UART prevents overwriting of the valid data currently in the Receive Data Register. The Break Detect and Overrun status bits are not displayed until after the valid data has been read.

After the valid data has been read, the UART Status 0 Register is updated to indicate the overrun condition (and Break Detect, if applicable). The `RDA` bit is set to 1 to indicate that the Receive Data Register contains a data byte. However, because the overrun error

Analog-to-Digital Converter

The Analog-to-Digital Converter (ADC) converts an analog input signal to its digital representation. The features of this sigma-delta ADC include:

- 10-bit resolution
- Eight single-ended analog input sources are multiplexed with general-purpose I/O ports
- Interrupt upon conversion complete
- Bandgap generated internal voltage reference generator with two selectable levels
- Factory offset and gain calibration

Architecture

Figure 19 displays the major functional blocks of the ADC. An analog multiplexer network selects the ADC input from the available analog pins, ANA0 through ANA7.

Figure 20. Flash Memory Arrangement

Flash Information Area

The Flash information area is separate from program memory and is mapped to the address range `FE00H` to `FFFFH`. Not all these addresses are accessible. Factory trim values for the analog peripherals are stored here. Factory calibration data for the ADC is also stored here.

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.

Figure 21 displays a basic Flash Controller flow. 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 21.

Flash Status Register

The Flash Status Register indicates the current state of the Flash Controller. This register can be read at any time. The read-only Flash Status Register shares its Register File address with the write-only Flash Control Register.

Table 82. Flash Status Register (FSTAT)

Bit	7	6	5	4	3	2	1	0
Field	Reserved		FSTAT					
RESET	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R
Address	FF8H							

Bit	Description
[7:6]	**Reserved** These bits are reserved and must be programmed to 0 when read.
[5:0] FSTAT	**Flash Controller Status** 000000 = Flash Controller locked. 000001 = First unlock command received (73H written). 000010 = Second unlock command received (8CH written). 000011 = Flash Controller unlocked. 000100 = Sector protect register selected. 001xxx = Program operation in progress. 010xxx = Page erase operation in progress. 100xxx = Mass erase operation in progress.

Flash Page Select Register

The Flash Page Select (FPS) register shares address space with the Flash Sector Protect Register. Unless the Flash controller is unlocked and written with 5EH, writes to this address target the Flash Page Select Register.

The register is used to select one of the eight available Flash memory pages to be programmed or erased. Each Flash Page contains 512 bytes of Flash memory. During a Page Erase operation, all Flash memory having addresses with the most significant 7-bits given by FPS[6:0] are chosen for program/erase operation.

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

Table 85. 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** High byte of the 16-bit Flash Frequency value.

Table 86. 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** Low byte of the 16-bit Flash Frequency value.

- If the PA2/$\overline{\text{RESET}}$ pin is held Low while a 32-bit key sequence is issued to the PA0/ DBG pin, the DBG feature is unlocked. After releasing PA2/$\overline{\text{RESET}}$, it is pulled high. At this point, the PA0/DBG pin can be used to autobaud and cause the device to enter DEBUG Mode. For more details, see the OCD Unlock Sequence (8-Pin Devices Only) section on page 161.

Exiting DEBUG Mode

The device exits DEBUG Mode following any of these operations:

- Clearing the DBGMODE bit in the OCD Control Register to 0
- Power-On Reset
- Voltage Brown-Out reset
- Watchdog Timer reset
- Asserting the $\overline{\text{RESET}}$ pin Low to initiate a Reset
- Driving the DBG pin Low while the device is in STOP Mode initiates a system reset

OCD Data Format

The OCD interface uses the asynchronous data format defined for RS-232. Each character is transmitted as 1 Start bit, 8 data bits (least-significant bit first), and 1 Stop bit as displayed in Figure 25.

START	D0	D1	D2	D3	D4	D5	D6	D7	STOP

Figure 25. OCD Data Format

> **Note:** When responding to a request for data, the OCD may commence transmitting immediately after receiving the stop bit of an incoming frame. Therefore, when sending the stop bit, the host must not actively drive the DBG pin High for more than 0.5 bit times. Zilog recommends that, if possible, the host drives the DBG pin using an open-drain output.

OCD Autobaud Detector/Generator

To run over a range of baud rates (data bits per second) with various system clock frequencies, the OCD contains an auto-baud detector/generator. After a reset, the OCD is idle until it receives data. The OCD requires that the first character sent from the host is the character `80H`. The character `80H` has eight continuous bits Low (one Start bit plus 7 data

```
DBG ← 0BH
DBG ← Program Memory Address[15:8]
DBG ← Program Memory Address[7:0]
DBG ← Size[15:8]
DBG ← Size[7:0]
DBG → 1–65536 data bytes
```

Write Data Memory (0CH). The Write Data Memory command writes data to Data Memory. This command is equivalent to the `LDE` and `LDEI` instructions. Data can be written 1–65536 bytes at a time (65536 bytes can be written by setting size to 0). If the device is not in DEBUG Mode or if the Flash Read Protect Option Bit is enabled, the data is discarded.

```
DBG ← 0CH
DBG ← Data Memory Address[15:8]
DBG ← Data Memory Address[7:0]
DBG ← Size[15:8]
DBG ← Size[7:0]
DBG ← 1–65536 data bytes
```

Read Data Memory (0DH). The Read Data Memory command reads from Data Memory. This command is equivalent to the `LDE` and `LDEI` instructions. Data can be read 1 to 65536 bytes at a time (65536 bytes can be read by setting size to 0). If the device is not in DEBUG Mode, this command returns `FFH` for the data.

```
DBG ← 0DH
DBG ← Data Memory Address[15:8]
DBG ← Data Memory Address[7:0]
DBG ← Size[15:8]
DBG ← Size[7:0]
DBG → 1–65536 data bytes
```

Read Program Memory CRC (0EH). The Read Program Memory Cyclic Redundancy Check (CRC) command computes and returns the CRC of Program Memory using the 16-bit CRC-CCITT polynomial. If the device is not in DEBUG Mode, this command returns `FFFFH` for the CRC value. Unlike most other OCD Read commands, there is a delay from issuing of the command until the OCD returns the data. The OCD reads the Program Memory, calculates the CRC value, and returns the result. The delay is a function of the Program Memory size and is approximately equal to the system clock period multiplied by the number of bytes in the Program Memory.

```
DBG ← 0EH
DBG → CRC[15:8]
DBG → CRC[7:0]
```

Step Instruction (10H). The Step Instruction steps one assembly instruction at the current Program Counter (PC) location. If the device is not in DEBUG Mode or the Flash Read Protect Option bit is enabled, the OCD ignores this command.

```
DBG ← 10H
```

Table 113. CPU Control Instructions

Mnemonic	Operands	Instruction
ATM	—	Atomic Execution
CCF	—	Complement Carry Flag
DI	—	Disable Interrupts
EI	—	Enable Interrupts
HALT	—	HALT Mode
NOP	—	No Operation
RCF	—	Reset Carry Flag
SCF	—	Set Carry Flag
SRP	src	Set Register Pointer
STOP	—	STOP Mode
WDT	—	Watchdog Timer Refresh

Table 114. Load Instructions

Mnemonic	Operands	Instruction
CLR	dst	Clear
LD	dst, src	Load
LDC	dst, src	Load Constant to/from Program Memory
LDCI	dst, src	Load Constant to/from Program Memory and Auto-Increment Addresses
LDE	dst, src	Load External Data to/from Data Memory
LDEI	dst, src	Load External Data to/from Data Memory and Auto-Increment Addresses
LDWX	dst, src	Load Word using Extended Addressing
LDX	dst, src	Load using Extended Addressing
LEA	dst, X(src)	Load Effective Address
POP	dst	Pop
POPX	dst	Pop using Extended Addressing
PUSH	src	Push
PUSHX	src	Push using Extended Addressing

On-Chip Debugger Timing

Figure 31 and Table 132 provide timing information for the DBG pin. The DBG pin timing specifications assume a 4 ns maximum rise and fall time.

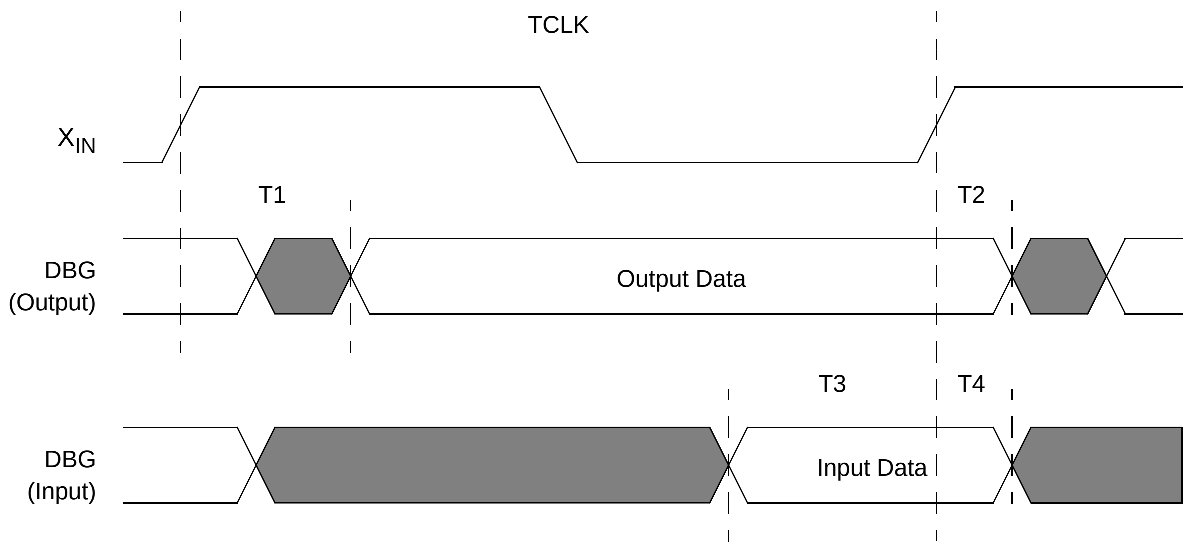

Figure 31. On-Chip Debugger Timing

Table 132. On-Chip Debugger Timing

		Delay (ns)	
Parameter	**Abbreviation**	**Minimum**	**Maximum**
DBG			
T_1	X_{IN} Rise to DBG Valid Delay	–	15
T_2	X_{IN} Rise to DBG Output Hold Time	2	–
T_3	DBG to X_{IN} Rise Input Setup Time	5	–
T_4	DBG to X_{IN} Rise Input Hold Time	5	–

J

L