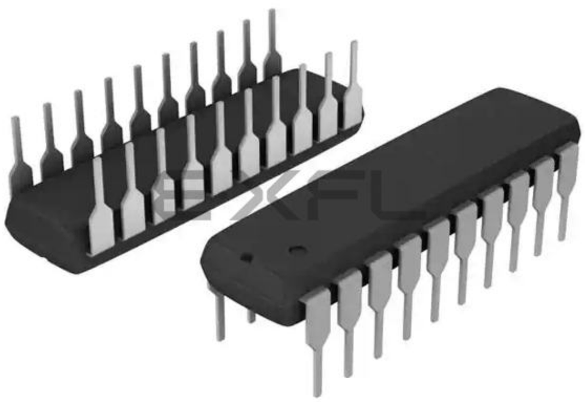

Details	
Product Status	Obsolete
Core Processor	eZ8
Core Size	8-Bit
Speed	20MHz
Connectivity	IrDA, UART/USART
Peripherals	Brown-out Detect/Reset, LED, LVD, POR, PWM, Temp Sensor, WDT
Number of I/O	17
Program Memory Size	4KB (4K x 8)
Program Memory Type	FLASH
EEPROM Size	128 x 8
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 3.6V
Data Converters	A/D 7x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°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/z8f042aph020eg

Pin Description

The Z8 Encore! XP F082A Series products are available in a variety of packages styles and pin configurations. This chapter describes the signals and available pin configurations for each of the package styles. For information about physical package specifications, see the Packaging chapter on page 245.

Available Packages

The following package styles are available for each device in the Z8 Encore! XP F082A Series product line:

- SOIC: 8-, 20- and 28-pin
- PDIP: 8-, 20- and 28-pin
- SSOP: 20- and 28- pin
- QFN 8-pin (MLF-S, a QFN-style package with an 8-pin SOIC footprint)

In addition, the Z8 Encore! XP F082A Series devices are available both with and without advanced analog capability (ADC, temperature sensor and op amp). Devices Z8F082A, Z8F042A, Z8F022A and Z8F012A contain the advanced analog, while devices Z8F081A, Z8F041A, Z8F021A and Z8F011A do not have the advanced analog capability.

Pin Configurations

Figure 2 through Figure 4 display the pin configurations for all the packages available in the Z8 Encore! XP F082A Series. See Table 2 on page 10 for a description of the signals. The analog input alternate functions (ANA*x*) are not available on the Z8F081A, Z8F041A, Z8F021A and Z8F011A devices. The analog supply pins (AV_{DD} and AV_{SS}) are also not available on these parts and are replaced by PB6 and PB7.

At reset, all Port A, B and C pins default to an input state. In addition, any alternate functionality is not enabled, so the pins function as general purpose input ports until programmed otherwise. At powerup, the PD0 pin defaults to the $\overline{\text{RESET}}$ alternate function.

The pin configurations listed are preliminary and subject to change based on manufacturing limitations.

Low-Power Modes

The Z8 Encore! XP F082A Series products contain power-saving features. The highest level of power reduction is provided by the STOP Mode, in which nearly all device functions are powered down. The next lower level of power reduction is provided by the HALT Mode, in which the CPU is powered down.

Further power savings can be implemented by disabling individual peripheral blocks while in Active mode (defined as being in neither STOP nor HALT Mode).

STOP Mode

Executing the eZ8 CPU's STOP instruction places the device into STOP Mode, powering down all peripherals except the Voltage Brown-Out detector, the Low-power Operational Amplifier and the Watchdog Timer. These three blocks may also be disabled for additional power savings. Specifically, the operating characteristics are:

- Primary crystal oscillator and internal precision oscillator are stopped; X_{IN} and X_{OUT} (if previously enabled) are disabled and PA0/PA1 revert to the states programmed by the GPIO registers
- System clock is stopped
- eZ8 CPU is stopped
- Program counter (PC) stops incrementing
- Watchdog Timer's internal RC oscillator continues to operate if enabled by the Oscillator Control Register
- If enabled, the Watchdog Timer logic continues to operate
- If enabled for operation in STOP Mode by the associated Flash option bit, the Voltage Brown-Out protection circuit continues to operate
- Low-power operational amplifier continues to operate if enabled by the Power Control Register
- All other on-chip peripherals are idle

To minimize current in STOP Mode, all GPIO pins that are configured as digital inputs must be driven to one of the supply rails (V_{CC} or GND). Additionally, any GPIOs configured as outputs must also be driven to one of the supply rails. The device can be brought out of STOP Mode using Stop Mode Recovery. For more information about Stop Mode Recovery, see the Reset, Stop Mode Recovery and Low Voltage Detection chapter on page 22.

Shared Debug Pin

On the 8-pin version of this device only, the Debug pin shares function with the PA0 GPIO pin. This pin performs as a general purpose input pin on power-up, but the debug logic monitors this pin during the reset sequence to determine if the unlock sequence occurs. If the unlock sequence is present, the debug function is unlocked and the pin no longer functions as a GPIO pin. If it is not present, the debug feature is disabled until/unless another reset event occurs. For more details, see the On-Chip Debugger chapter on page 180.

Crystal Oscillator Override

For systems using a crystal oscillator, PA0 and PA1 are used to connect the crystal. When the crystal oscillator is enabled, the GPIO settings are overridden and PA0 and PA1 are disabled. See the Oscillator Control Register Definitions section on page 196 for details.

5V Tolerance

All six I/O pins on the 8-pin devices are 5V-tolerant, unless the programmable pull-ups are enabled. If the pull-ups are enabled and inputs higher than V_{DD} are applied to these parts, excessive current flows through those pull-up devices and can damage the chip.

Note: In the 20- and 28-pin versions of this device, any pin which shares functionality with an ADC, crystal or comparator port is not 5 V-tolerant, including PA[1:0], PB[5:0] and PC[2:0]. All other signal pins are 5 V-tolerant and can safely handle inputs higher than V_{DD} except when the programmable pull-ups are enabled.

External Clock Setup

For systems using an external TTL drive, PB3 is the clock source for 20- and 28-pin devices. In this case, configure PB3 for alternate function CLKIN. Write the Oscillator Control (OSCCTL) Register such that the external oscillator is selected as the system clock. See the Oscillator Control Register Definitions section on page 196 for details. For 8-pin devices, use PA1 instead of PB3.

it is appropriate to have the Timer Output make a state change at a One-Shot time-out (rather than a single cycle pulse), first set the TPOL bit in the Timer Control Register to the start value before enabling ONE-SHOT Mode. After starting the timer, set `TPOL` to the opposite bit value.

Observe the following steps for configuring a timer for ONE-SHOT Mode and initiating the count:

1. Write to the Timer Control Register to:
   - Disable the timer
   - Configure the timer for ONE-SHOT Mode.
   - Set the prescale value.
   - Set the initial output level (High or Low) if using the Timer Output alternate function.
2. Write to the Timer High and Low Byte registers to set the starting count value.
3. Write to the Timer Reload High and Low Byte registers to set the reload value.
4. If appropriate, enable the timer interrupt and set the timer interrupt priority by writing to the relevant interrupt registers.
5. If using the Timer Output function, configure the associated GPIO port pin for the Timer Output alternate function.
6. Write to the Timer Control Register to enable the timer and initiate counting.

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

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

CONTINUOUS Mode

In CONTINUOUS Mode, the timer counts up to the 16-bit reload value stored in the Timer Reload High and Low Byte registers. The timer input is the system clock. Upon reaching the reload value, the timer generates an interrupt, 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 enabled, the Timer Output pin changes state (from Low to High or from High to Low) at timer Reload.

Observe the following steps for configuring a timer for CONTINUOUS Mode and initiating the count:

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. Further, the high or low state of the input signal pulse must be no less than twice the system clock period. A shorter pulse may not be captured.

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 enabled, the Timer Output pin changes state (from Low to High or from High to Low) at timer Reload.

Observe the following steps for configuring a timer for COUNTER Mode and initiating the count:

1. Write to the Timer Control Register to:
   - Disable the timer.
   - Configure the timer for COUNTER Mode.
   - Select either the rising edge or falling edge of the Timer Input signal for the count. This selection also sets the initial logic level (High or Low) for the Timer Output alternate function. However, the Timer Output function is not required to be enabled.
2. Write to the Timer High and Low Byte registers to set the starting count value. This only affects the first pass in COUNTER Mode. After the first timer Reload in COUNTER Mode, counting always begins at the reset value of `0001H`. In COUNTER Mode the Timer High and Low Byte registers must be written with the value `0001H`.
3. Write to the Timer Reload High and Low Byte registers to set the reload value.
4. If appropriate, enable the timer interrupt and set the timer interrupt priority by writing to the relevant interrupt registers.
5. Configure the associated GPIO port pin for the Timer Input alternate function.
6. If using the Timer Output function, configure the associated GPIO port pin for the Timer Output alternate function.
7. Write to the Timer Control Register to enable the timer.

In COUNTER Mode, the number of Timer Input transitions since the timer start is computed via the following equation:

$$\text{COUNTER Mode Timer Input Transitions} = \text{Current Count Value} - \text{Start Value}$$

COMPARATOR COUNTER Mode

In COMPARATOR COUNTER Mode, the timer counts input transitions from the analog comparator output. The `TPOL` bit in the Timer Control Register selects whether the count occurs on the rising edge or the falling edge of the comparator output signal. In COMPARATOR COUNTER Mode, the prescaler is disabled.

– 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 TxCTL0 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.
7. Counting begins on the first appropriate transition of the Timer Input signal. No interrupt is generated by this first edge.

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

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

Reading the Timer Count Values

The current count value in the timers can be read while counting (enabled). 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 while 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

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

Bit	Description (Continued)
[5:3] PRES	**Prescale value** The timer input clock is divided by 2^{PRES}, where PRES can be set from 0 to 7. The prescaler is reset each time the Timer is disabled. This reset ensures proper clock division each time the Timer is restarted. 000 = Divide by 1. 001 = Divide by 2. 010 = Divide by 4. 011 = Divide by 8. 100 = Divide by 16. 101 = Divide by 32. 110 = Divide by 64. 111 = Divide by 128.
[2:0] TMODE	**Timer Mode** This field, along with the TMODEHI bit in the TxCTL0 Register, determines the operating mode of the timer. TMODEHI is the most significant bit of the Timer mode selection value. The entire operating mode bits are expressed as {TMODEHI, TMODE[2:0]}. The TMODEHI is bit 7 of the TxCTL0 Register while TMODE[2:0] is the lower 3 bits of the TxCTL1 Register. 0000 = ONE-SHOT Mode. 0001 = CONTINUOUS Mode. 0010 = COUNTER Mode. 0011 = PWM SINGLE OUTPUT Mode. 0100 = CAPTURE Mode. 0101 = COMPARE Mode. 0110 = GATED Mode. 0111 = CAPTURE/COMPARE Mode. 1000 = PWM DUAL OUTPUT Mode. 1001 = CAPTURE RESTART Mode. 1010 = COMPARATOR COUNTER Mode.

Timer 0–1 High and Low Byte Registers

The Timer 0–1 High and Low Byte (TxH and TxL) registers, shown in Tables 52 and 53, 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 when the timers are enabled. When the timer is disabled, reads from TxL read the register 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, so simultaneous 16-bit writes are not possible. If either the Timer High or Low Byte registers are written 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.

Timer Reload High and Low Byte Registers

The Timer 0–1 Reload High and Low Byte (TxRH and TxRL) registers, shown in Tables 54 and 55, 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 reload value.

In COMPARE Mode, the Timer Reload High and Low Byte registers store the 16-bit Compare value.

Table 54. Timer 0–1 Reload High Byte Register (TxRH)

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	F02H, F0AH							

Table 55. Timer 0–1 Reload Low Byte Register (TxRL)

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	F03H, F0BH							

Bit	**Description**
[7:0] TRH, TRL	**Timer Reload Register High and Low** These two bytes form the 16-bit reload value, {TRH[7:0], TRL[7:0]}. This value sets the maximum count value which initiates a timer reload to `0001H`. In COMPARE Mode, these two bytes form the 16-bit Compare value.

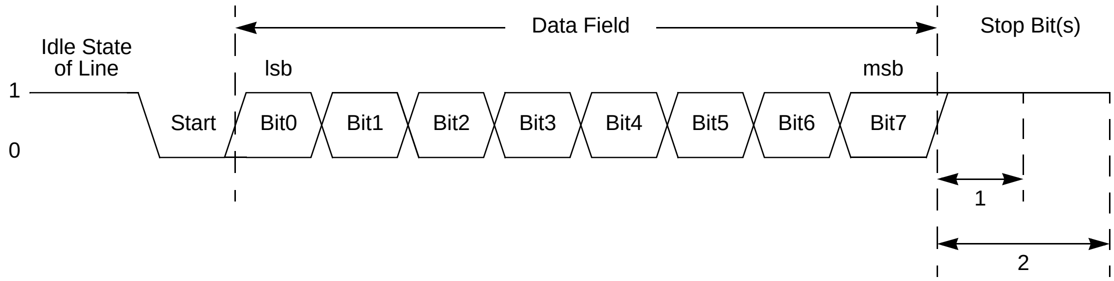

Figure 11. UART Asynchronous Data Format without Parity

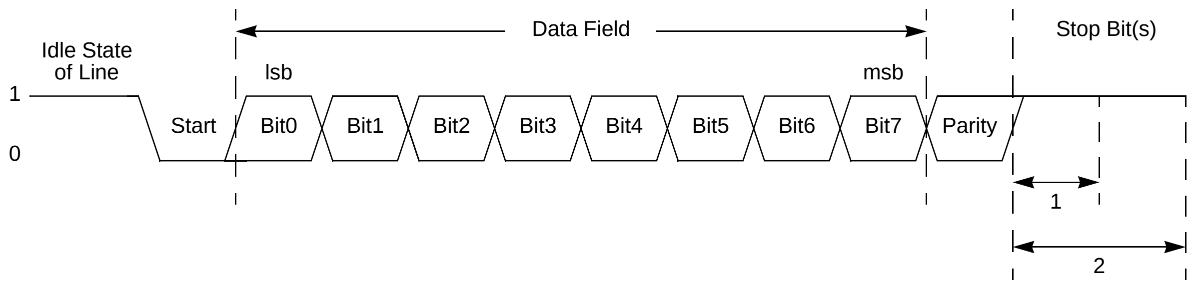

Figure 12. UART Asynchronous Data Format with Parity

Transmitting Data using the Polled Method

Observe the following steps to transmit data using the polled method of operation:

1. Write to the UART Baud Rate High and Low Byte registers to set the required baud rate.
2. Enable the UART pin functions by configuring the associated GPIO port pins for alternate function operation.
3. Write to the UART Control 1 Register, if MULTIPROCESSOR Mode is appropriate, to enable MULTIPROCESSOR (9-bit) Mode functions.
4. Set the Multiprocessor Mode Select (MPEN) bit to enable MULTIPROCESSOR Mode.
5. Write to the UART Control 0 Register to:
   – Set the transmit enable bit (TEN) to enable the UART for data transmission
   – Set the parity enable bit (PEN), if parity is appropriate and MULTIPROCESSOR Mode is not enabled and select either even or odd parity (PSEL)

The third scheme is enabled by setting MPMD[1:0] to `11b` and by writing the UART's address into the UART Address Compare Register. This mode is identical to the second scheme, except that there are no interrupts on address bytes. The first data byte of each frame remains accompanied by a `NEWFRM` assertion.

External Driver Enable

The UART provides a Driver Enable (DE) signal for off-chip bus transceivers. This feature reduces the software overhead associated with using a GPIO pin to control the transceiver when communicating on a multi-transceiver bus, such as RS-485.

Driver Enable is an active High signal that envelopes the entire transmitted data frame including parity and Stop bits as displayed in Figure 14. The Driver Enable signal asserts when a byte is written to the UART Transmit Data Register. The Driver Enable signal asserts at least one UART bit period and no greater than two UART bit periods before the Start bit is transmitted. This allows a setup time to enable the transceiver. The Driver Enable signal deasserts one system clock period after the final Stop bit is transmitted. This one system clock delay allows both time for data to clear the transceiver before disabling it, plus the ability to determine if another character follows the current character. In the event of back to back characters (new data must be written to the Transmit Data Register before the previous character is completely transmitted) the DE signal is not deasserted between characters. The DEPOL bit in the UART Control Register 1 sets the polarity of the Driver Enable signal.

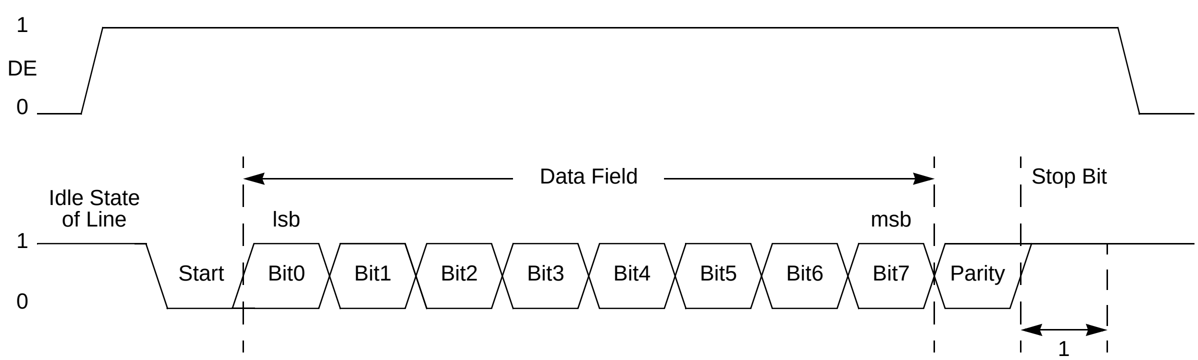

Figure 14. UART Driver Enable Signal Timing (shown with 1 Stop Bit and Parity)

The Driver Enable-to-Start bit setup time is calculated as follows:

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

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 section on page 99.

! **Caution:** To prevent spurious signals during IrDA data transmission, set the IREN bit in the UART Control 1 Register to 1 to enable the Infrared Encoder/Decoder before enabling the GPIO Port alternate function for the corresponding pin.

Output Data

The output format of the corrected ADC value is shown below.

MSB	LSB
s v b a 9 8 7 6	5 4 3 2 1 0 – –

The overflow bit in the corrected output indicates that the computed value was greater than the maximum logical value (+1023) or less than the minimum logical value (–1024). Unlike the hardware overflow bit, this is not a simple binary flag. For a normal (nonoverflow) sample, the sign and the overflow bit match. If the sign bit and overflow bit do not match, a computational overflow has occurred.

Input Buffer Stage

Many applications require the measurement of an input voltage source with a high output impedance. This ADC provides a buffered input for such situations. The drawback of the buffered input is a limitation of the input range. When using unity gain buffered mode, the input signal must be prevented from coming too close to either V_{SS} or V_{DD}. See Table 139 on page 236 for details.

This condition applies only to the input voltage level (with respect to ground) of each differential input signal. The actual differential input voltage magnitude may be less than 300mV.

The input range of the unbuffered ADC swings from V_{SS} to V_{DD}. Input signals smaller than 300mV must use the unbuffered input mode. If these signals do not contain low output impedances, they might require off-chip buffering.

Signals outside the allowable input range can be used without instability or device damage. Any ADC readings made outside the input range are subject to greater inaccuracy than specified.

ADC Control Register Definitions

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

ADC Control Register 0 (ADCCTL0): see page 134

ADC Control/Status Register 1 (ADCCTL1): see page 136

ADC Data High Byte Register (ADCD_H): see page 137

ADC Data Low Byte Register (ADCD_L): see page 137

Table 74. ADC Control/Status Register 1 (ADCCTL1)

Bit	7	6	5	4	3	2	1	0
Field	REFSELH	Reserved				BUFMODE[2:0]		
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	F71H							

Bit	Description
[7] REFSELH	**Voltage Reference Level Select High Bit** In conjunction with the Low bit (REFSELL) in ADC Control Register 0, this determines the level of the internal voltage reference; the following details the effects of {REFSELH, REFSELL}; this reference is independent of the Comparator reference. 00= Internal Reference Disabled, reference comes from external pin. 01= Internal Reference set to 1.0V. 10= Internal Reference set to 2.0V (default). 11= Reserved.
[6:3]	**Reserved** These bits are reserved and must be programmed to 0000.
[2:0] BUFMODE[2:0]	**Input Buffer Mode Select** 000 = Single-ended, unbuffered input. 001 = Single-ended, buffered input with unity gain. 010 = Reserved. 011 = Reserved. 100 = Differential, unbuffered input. 101 = Differential, buffered input with unity gain. 110 = Reserved. 111 = Reserved.

ADC Data High Byte Register

The ADC Data High Byte (ADCD_H) Register contains the upper eight bits of the ADC output. The output is an 13-bit two's complement value. During a single-shot conversion, this value is invalid. Access to the ADC Data High Byte Register is read-only. Reading the ADC Data High Byte Register latches data in the ADC Low Bits Register.

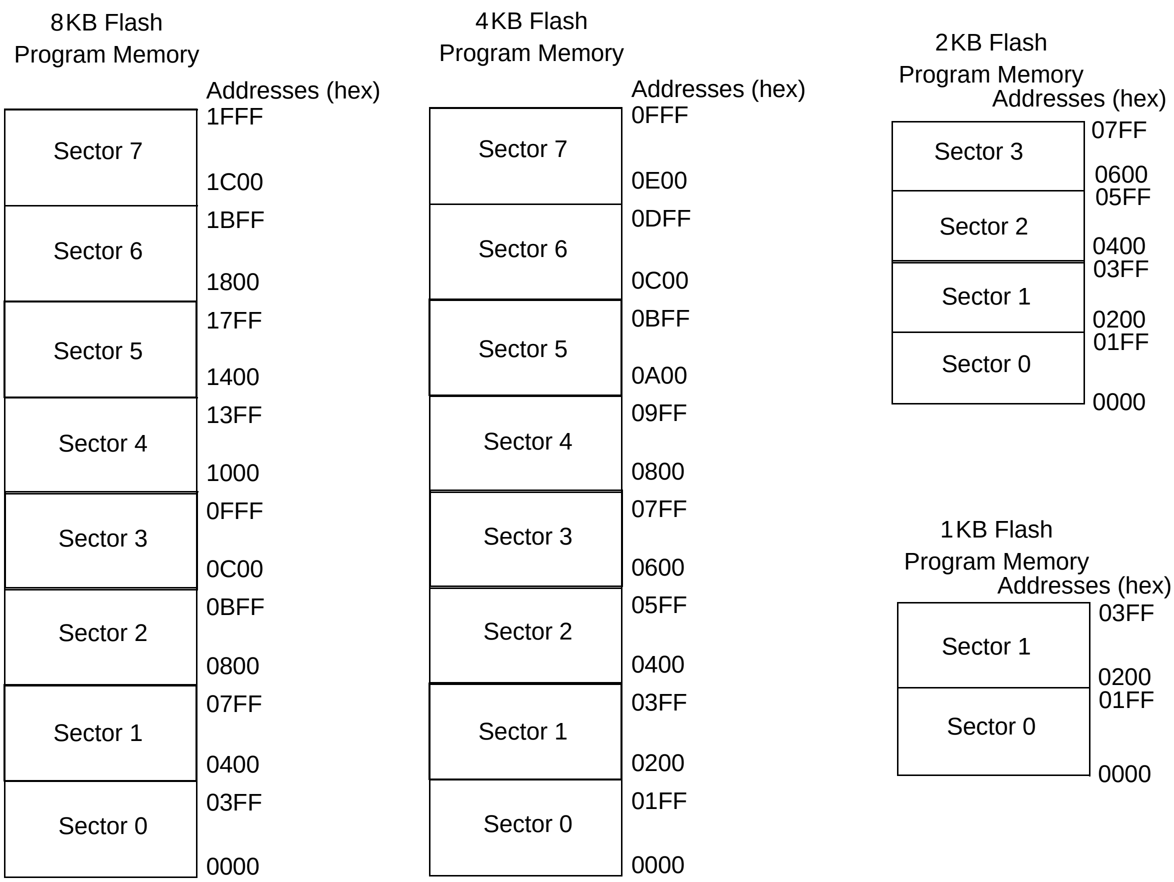

Figure 21. Flash Memory Arrangement

Flash Information Area

The Flash information area is separate from Program Memory and is mapped to the address range `FE00H` to `FFFFH`. This area is readable but cannot be erased or overwritten. 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.

Flash Operation Timing Using the Flash Frequency Registers

Before performing either a program or erase operation on Flash memory, you must first configure the Flash Frequency High and Low Byte registers. The Flash Frequency registers allow programming and erasing of the Flash with system clock frequencies ranging from 32 kHz (32768 Hz) through 20 MHz.

The Flash Frequency High and Low Byte registers 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). This value is calculated using the following equation:

$$\text{FFREQ[15:0]} = \frac{\text{System Clock Frequency (Hz)}}{1000}$$

! Caution: Flash programming and erasure are not supported for system clock frequencies below 32 kHz (32768 Hz) or above 20 MHz. The Flash Frequency High and Low Byte registers must be loaded with the correct value to ensure operation of the Z8 Encore! XP F082A Series devices.

Flash Code Protection Against External Access

The user code contained within the Flash memory can be protected against external access by the on-chip debugger. Programming the FRP Flash option bit prevents reading of the user code with the On-Chip Debugger. See the Flash Option Bits chapter on page 159 and the On-Chip Debugger chapter on page 180 for more information.

Flash Code Protection Against Accidental Program and Erasure

The Z8 Encore! XP F082A Series provides several levels of protection against accidental program and erasure of the Flash memory contents. This protection is provided by a combination of the Flash option bits, the register locking mechanism, the page select redundancy and the sector level protection control of the Flash Controller.

Flash Code Protection Using the Flash Option Bits

The FRP and FWP Flash option bits combine to provide three levels of Flash Program Memory protection, as shown in Table 79. See the Flash Option Bits chapter on page 159 for more information.

Table 79. Flash Code Protection Using the Flash Option Bits

FWP	Flash Code Protection Description
0	Programming and erasing disabled for all of Flash Program Memory. In user code programming, Page Erase and Mass Erase are all disabled. Mass Erase is available through the On-Chip Debugger.
1	Programming, Page Erase and Mass Erase are enabled for all of Flash Program Memory.

Flash Code Protection Using the Flash Controller

At Reset, the Flash Controller locks to prevent accidental program or erasure of the Flash memory. To program or erase the Flash memory, first write the Page Select Register with the target page. 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 target page. 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 22 on page 148 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 only in through the Debug Port.

After unlocking a specific page, you 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 cause 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! XP devices are divided into maximum number of 8 sectors. A sector is 1/8 of the total Flash memory size unless this value is smaller than the page size – in which case, the sector and page sizes are equal. On Z8 Encore! F082A Series devices, the sector size is varied according to the Flash memory configuration shown in Table 78 on page 146.

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 are lost. User code must write this register in their initialization routine if they prefer 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 Regisenabled, 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.

Runtime Counter

The On-Chip Debugger contains a 16-bit Runtime Counter. It counts system clock cycles between Breakpoints. The counter starts counting when the On-Chip Debugger leaves DEBUG Mode and stops counting when it enters DEBUG Mode again or when it reaches the maximum count of FFFFH.

On-Chip Debugger Commands

The host communicates to the on-chip debugger by sending OCD commands using the DBG interface. During normal operation, only a subset of the OCD commands are available. In DEBUG Mode, all OCD commands become available unless the user code and control registers are protected by programming the Flash Read Protect Option bit (FRP). The Flash Read Protect Option bit prevents the code in memory from being read out of the Z8 Encore! XP F082A Series device. When this option is enabled, several of the OCD commands are disabled. See Table 109.

Table 110 on page 191 is a summary of the on-chip debugger commands. Each OCD command is described in further detail in the bulleted list following this table. Table 110 also indicates those commands that operate when the device is not in DEBUG Mode (normal operation) and those commands that are disabled by programming the Flash Read Protect Option bit.

Table 109. Debug Command Enable/Disable

Debug Command	Command Byte	Enabled when Not in DEBUG Mode?	Disabled by Flash Read Protect Option Bit
Read OCD Revision	00H	Yes	–
Reserved	01H	–	–
Read OCD Status Register	02H	Yes	–
Read Runtime Counter	03H	–	–
Write OCD Control Register	04H	Yes	Cannot clear DBGMODE bit.
Read OCD Control Register	05H	Yes	–

Packaging

Zilog's Product Line of MCUs includes the Z8F011A, Z8F012A, Z8F021A, Z8F022A, Z8F041A, Z8F042A, Z8F081A and Z8F082A devices, which are available in the following packages:

- 8-pin Plastic Dual-Inline Package (PDIP)
- 8-Pin Quad Flat No-Lead Package (QFN)/MLF-S[1]
- 8-pin Small Outline Integrated Circuit Package (SOIC)
- 20-pin Small Outline Integrated Circuit Package (SOIC)
- 20-pin Small Shrink Outline Package (SSOP)
- 20-pin Plastic Dual-Inline Package (PDIP)
- 28-pin Small Outline Integrated Circuit Package (SOIC)
- 28-pin Small Shrink Outline Package (SSOP)
- 28-pin Plastic Dual-Inline Package (PDIP)

Current diagrams for each of these packages are published in Zilog's Packaging Product Specification (PS0072), which is available free for download from the Zilog website.

1. The footprint of the QFN)/MLF-S package is identical to that of the 8-pin SOIC package, but with a lower profile.