zilog
Embedded in Life
An IXYS Company

# Z8FS021

# ZMOTION® Intrusion Detection

## Product Specification

PS028807-0917

ZMOTION™

www.zilog.com

⚠ **Warning:** DO NOT USE THIS PRODUCT IN LIFE SUPPORT SYSTEMS.

## LIFE SUPPORT POLICY

ZILOG'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS PRIOR WRITTEN APPROVAL OF THE PRESIDENT AND GENERAL COUNSEL OF ZILOG CORPORATION.

### As used herein

Life support devices or systems are devices which (a) are intended for surgical implant into the body, or (b) support or sustain life and whose failure to perform when properly used in accordance with instructions for use provided in the labeling can be reasonably expected to result in a significant injury to the user. A critical component is any component in a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system or to affect its safety or effectiveness.

### Document Disclaimer

Information in this publication concerning the devices, applications, or technology described is intended to suggest possible uses and may be superseded. ZILOG, INC. DOES NOT ASSUME LIABILITY FOR OR PROVIDE A REPRESENTATION OF ACCURACY OF THE INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED IN THIS DOCUMENT. ZILOG ALSO DOES NOT ASSUME LIABILITY FOR INTELLECTUAL PROPERTY INFRINGEMENT RELATED IN ANY MANNER TO USE OF INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED HEREIN OR OTHERWISE. The information contained within this document has been verified according to the general principles of electrical and mechanical engineering.

ZMOTION and Z8 Encore! XP are trademarks or registered trademarks of Zilog, Inc. (An IXYS Company). All other product or service names are the property of their respective owners.

# Revision History

Each instance in the following revision history table reflects a change to this document from its previous version. For more details, refer to the corresponding pages or appropriate links provided in the table.

| Date | Revision Level | Description | Page Number |
|---|---|---|---|
| Sep 2017 | 07 | Corrected the ZMOTION Engine revision. | 34 |
| Mar 2015 | 06 | Added note indicating that the device is not preprogrammed and user is required to download generic intrusion detection code RD0001-SC01. | 2, 4, 10, 47 |
| Sep 2014 | 05 | Individual lenses and pyros are no longer bundled and can now be ordered separately. Therefore, the Ordering Information, PIR Engine Revisions, and ZDS II Project Settings sections have been updated, and the Lens Specifications appendix has been deleted. | 47, 59 |
| Oct 2011 | 04 | Corrected error in ePIR_SC1 description, Table 9; modified Packaging section. | 29, 46 |
| Jul 2011 | 03 | Corrections to ePIR_SC1, ePIR_Process_Rate and ePIR_Signal_DC registers | 29, 40, 44 |
| Apr 2011 | 02 | Corrections to PIR Noise Sensitivity Level Register and PIR Transient Sensitivity Level Register. | 42 |
| Feb 2011 | 01 | Original issue. | All |

# Table of Contents

# List of Figures

# List of Tables

# Overview

The ZMOTION Intrusion Detection device is an integrated and flexible solution for Passive Infrared (PIR)-based motion detection applications. The ZMOTION family includes a series of high-performance microcontrollers with integrated motion-detection algorithms and a selection of lenses and PIR sensors to fit a wide range of intrusion detection and security applications. Zilog's ZMOTION Intrusion Detection Solution provides a dramatic improvement in both sensitivity and stability over traditional security-related motion detection designs with integrated functions such as White Light detection and Pet Immunity. As a result, it is the ideal solution for security applications in which intrusion detection capability is vital.

The ZMOTION Intrusion Detection Solution, based on Zilog's Z8FS021 MCU, combines the programmability and rich peripheral set of our Z8 Encore! XP family of In-Circuit Programmable Flash MCUs with built-in motion detection software algorithms to provide the functions necessary for PIR motion detection applications. These algorithms comprise the PIR Engine and run in the background while control and status of the Engine is accessed through a software Application Programmer Interface (API). These APIs allow designers to create their own application-specific software while taking advantage of Zilog's ZMOTION Motion Detection Technology. Additional API settings are provided to match PIR Engine operation to each lens and pyroelectric sensor combination.

The Flash in-circuit programming capability of the Z8FS021 MCU allows for faster development time, more flexible manufacturing and firmware changes in the field.

As with all of Zilog's ZMOTION products, the ZMOTION Intrusion Detection MCU provides optimized configuration parameters for each lens/sensor combination to ensure the best possible performance while significantly reducing development risk and minimizing time to market.

## ZMOTION Intrusion Detection Features

- Software-based Motion Detection (PIR) Engine controlled and monitored via software API registers
- Select from an assortment of lenses and pyroelectric sensors to best fit your application
- API settings provided for each lens and pyroelectric sensor combination
- Sensitivity control, range control and directionality detection
- Accurate frequency discrimination and programmable pet immunity
- No temperature compensation required
- White light detection using status LED reduces system cost (eliminates CDS photocell)
- White Light Anti-Jam feature and programmable sensitivity to support a wide range of LED and light pipe configurations

- Programmable transient and noise detection

## Z8FS040 MCU Features

- High-performance eZ8® CPU core
- 2 KB in-circuit programmable Flash available for application code

**Note:** The Z8FS021 MCU is not preprogammed with application code. A generic version of the ZMOTION Intrusion Detection source code (RD0001-SC01) can be downloaded from zilog.com and programmed into the MCU.

- Single-pin debug with unlimited breakpoints
- Flexible clocking scheme
- Internal precision oscillator running at 5.53 MHz
- External oscillator operating up to 20 MHz
- Sigma Delta ADC
- Up to 6 single-ended channels or 3 differential channels available
- On-chip analog comparator with independent programmable reference voltage
- Full-duplex UART with dedicated BRG
- Two 16-bit timers with input capture, output compare and PWM capability (11 modes total)
- Watchdog timer (WDT) with dedicated internal oscillator
- Up to 20 vectored interrupts
- 6 to 25 I/O pins depending upon package
- 2.7 V to 3.6 V operating voltage with extended operating temperature range –40°C to +105°C
- Low power modes

# Z8FS040 MCU Block Diagram

Figure 1 displays a block diagram of the Z8FS040 MCU.

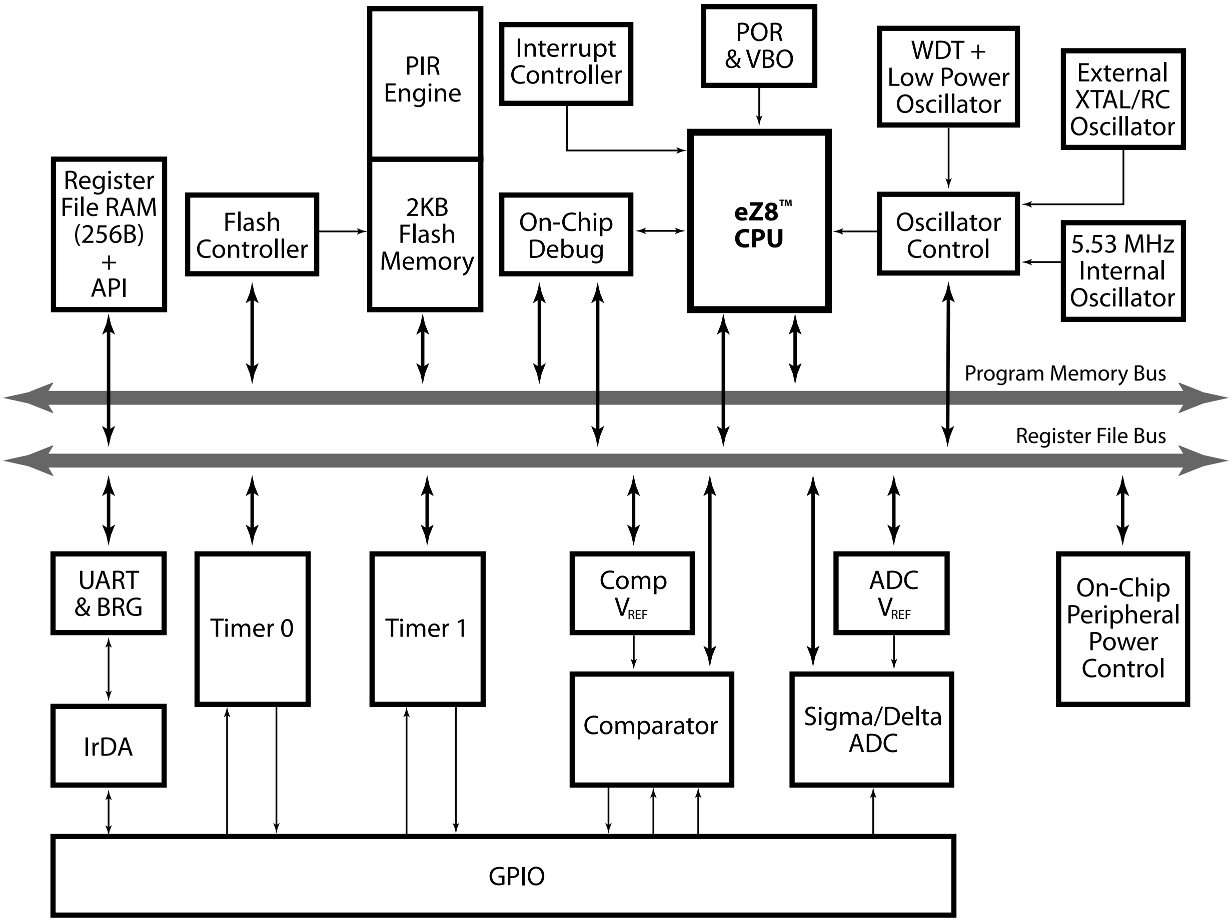

**Figure 1. Z8FS040 MCU Block Diagram**

# MCU Part Selection Guide

Table 1 displays the basic features and package styles available for each device within the Z8FS021 ZMOTION Intrusion Detection MCU devices. Select the package type that is most suitable to your application based on required General Purpose I/Os and ADC channels. The table suggests references to the pin configuration diagrams for the peripheral functions available on each I/O pin.

See the Ordering Information section on page 47 for a list of all ZMOTION Intrusion Detection part numbers.

**Note:** The Z8FS021 MCU is not preprogammed with application code. A generic version of the ZMOTION Intrusion Detection source code (RD0001-SC01) can be downloaded from zilog.com and programmed into the MCU.

**Table 1. Z8FS021 ZMOTION Intrusion Detection MCU Part Selection Guide**

| ZMOTION MCU Part Number | Z8 Encore! XP Base Part Number | Flash Memory | GPIO | ADC Channels | Package | Pin Configuration Diagram |
|---|---|---|---|---|---|---|
| Z8FS021xSB20EG | Z8F082ASB020EG | 2KB | 5 | 3 | 8-pin SOIC | Figure 2 |
| Z8FS021xHH20EG | Z8F082AHH020EG | 2KB | 16 | 4 | 20-pin SSOP | Figure 3 |
| Z8FS021xHJ20EG | Z8F082AHJ020EG | 2KB | 22 | 6 | 28-pin SSOP | Figure 4 |
| Note: x = PIR Engine Revision Identifier (see Table 4 on page 23.) | | | | | | |

# Pin Configuration

The Z8FS021 MCU is available in a variety of package styles and pin configurations. This chapter describes the signals and available pin configurations for each of the package styles. For information about the physical package specifications, see the Packaging section on page 46.

Figures 2 through 4 display the pin configurations of all the packages available for the ZMOTION MCU Series. For a description of the signals, see Tables 6 through 8 on pages 24 through 26.

At reset, all port pins are set to GPIO input state except /RESET/DE0/T1OUT (8-pin) which is configured to /RESET, PA0/T0IN/T0OUT/XIN/DBG (8-pin), which is configured to DBG and RESET/PD0 (20- and 28-pin) which are configured to /RESET.

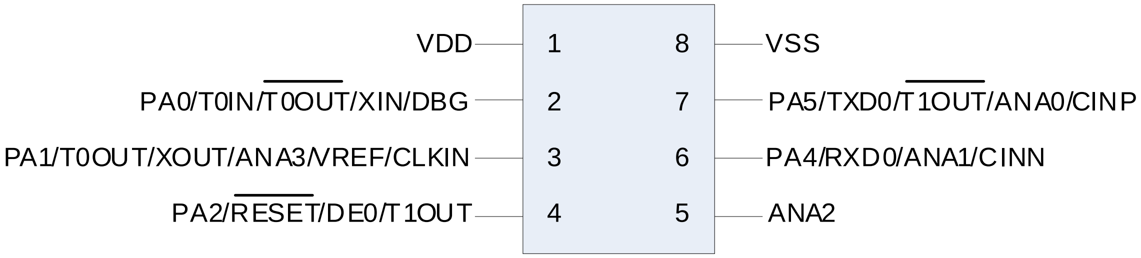

Figure 2. 8-Pin SOIC Package Diagram – Z8FS021xSB20EG

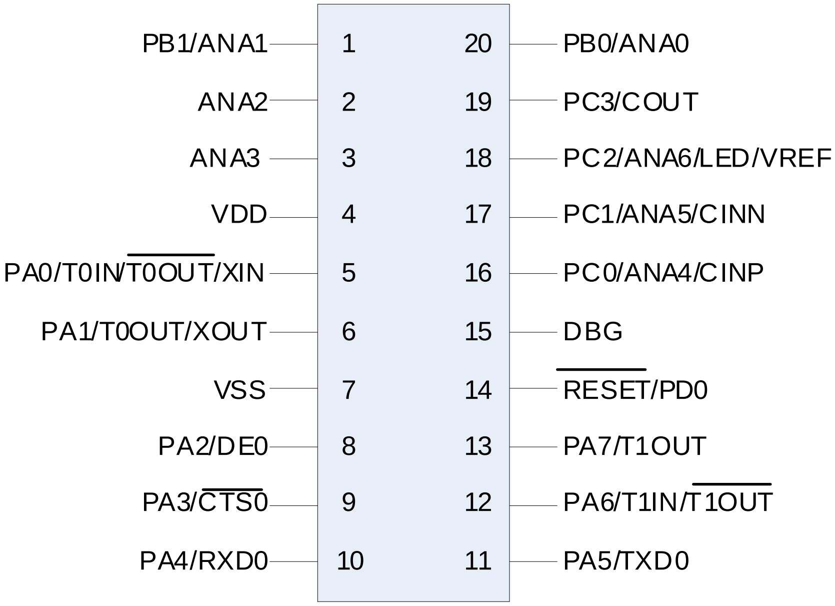

**Figure 3. 20-Pin SSOP Package Diagram – Z8FS021xHH20EG**

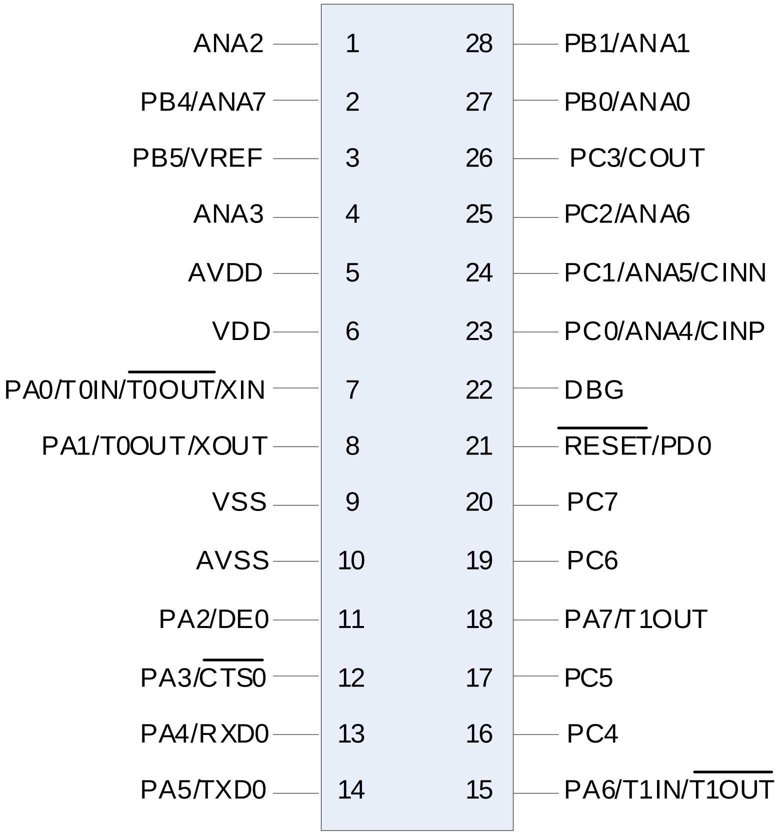

**Figure 4. 28-Pin SSOP Package Pin-Out, Z8FS021xHJ20EG**

# Signal Descriptions

Table 2 describes the Z8FS021 MCU signals. Signal availability is package dependent. See the Pin Configuration section on page 5 for signal availability multiplexing.

**Table 2. Z8FS021 MCU Signal Descriptions**

| Signal<br>Mnemonic | I/O | Description |
|---|---|---|
| **General-Purpose I/O Ports A–D** | | |
| PA[7:0] | I/O | Port A. These pins are used for general-purpose I/O. |
| PB[5:0] | I/O | Port B. These pins are used for general-purpose I/O. |
| PC[7:0] | I/O | Port C. These pins are used for general-purpose I/O. |
| PD[0] | O | Port D. This pin is used for general-purpose output only. |
| **UART Controllers** | | |
| TXD0 | O | Transmit Data. This signal is the transmit output from the UART and IrDA. |
| RXD0 | I | Receive Data. This signal is the receive input for the UART and IrDA. |
| CTS0 | I | Clear To Send. This signal is the flow control input for the UART. |
| DE | O | Driver Enable. This signal allows automatic control of external RS-485 drivers. It is approximately the inverse of the Transmit Empty (TXE) bit in the UART Status 0 Register. The DE signal can be used to ensure that the external RS-485 driver is enabled when data is transmitted by the UART. |
| **Timers** | | |
| T0OUT/T1OUT | O | Timer Output 0–1. These signals are outputs from the timers. |
| T0OUT/T1OUT | O | Timer Complement Output 0–1. These signals are output from the timers in PWM Dual Output mode. |
| T0IN/T1IN | I | Timer Input 0–1. These signals are used as the capture, gating and counter inputs. |
| **Comparator** | | |
| CINP/CINN | I | Comparator Inputs. These signals are the positive and negative inputs to the comparator. |
| COUT | O | Comparator Output. |
| **Analog** | | |
| ANA[7:0] | I | Analog Port. These signals are used as inputs to the analog-to-digital converter (ADC). |
| $V_{REF}$ | I/O | Analog-to-digital converter reference voltage input, or buffered output for internal reference. |

**Table 2. Z8FS021 MCU Signal Descriptions (Continued)**

| Signal Mnemonic | I/O | Description |
|---|---|---|
| **Oscillators** | | |
| XIN | I | External Crystal Input. This pin is the input pin to the crystal oscillator. A crystal can be connected between it and the XOUT pin to form the oscillator. In addition, this pin is used with external RC networks or external clock drivers to provide the system clock. |
| XOUT | O | External Crystal Output. This pin is the output of the crystal oscillator. A crystal can be connected between it and the XIN pin to form the oscillator. |
| **Clock Input** | | |
| CLKIN | I | Clock Input Signal. This pin can be used to input a TTL-level signal to be used as the system clock. |
| **LED Drivers** | | |
| LED | O | Direct LED drive capability. All port C pins have the capability to drive an LED without any other external components. These pins have programmable drive strengths set by the GPIO block. |
| **On-Chip Debugger** | | |
| DBG | I/O | Debug. This signal is the control and data input and output to and from the On-Chip Debugger. |
| ⚠ **Caution:** | | The DBG pin is open-drain and requires a pull-up resistor to ensure proper operation. |
| **Reset** | | |
| RESET | I/O | RESET. Generates a Reset when asserted (driven Low). Also serves as a reset indicator; the Z8 Encore! XP forces this pin low when in reset. This pin is open-drain and features an enabled internal pull-up resistor. |
| **Power Supply** | | |
| $V_{DD}$ | I | Digital Power Supply. |
| $AV_{DD}$ | I | Analog Power Supply. |
| $V_{SS}$ | I | Digital Ground. |
| $AV_{SS}$ | I | Analog Ground. |

# Flash Memory Map

The Z8FS021 MCU is based on Zilog's Z8F082A device, which contains a total of 8KB Flash memory. Zilog's PIR technology is located in the 6KB address range `0800h–1FFFh`, a code space that is locked and cannot be erased by the user or by the Zilog Debug Interface (ZDI) mass erase or page erase commands. The remaining 2 KB of this Flash memory space, in the address range `0000h–07FFh`, is available for user application code.

**Note:** The Z8FS021 MCU is not preprogammed with application code. A generic version of the ZMOTION Intrusion Detection source code (RD0001-SC01) can be downloaded from zilog.com and programmed into the MCU.

A memory map of the Z8FS021 MCU's Flash code space is illustrated in Figure 5.

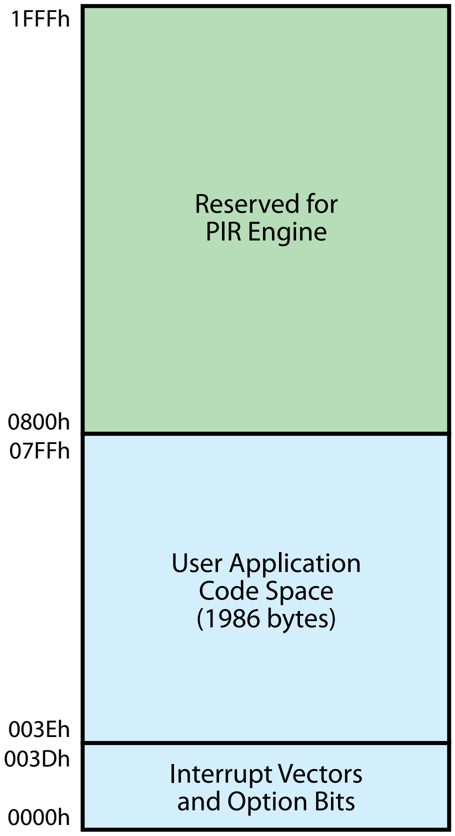

**Figure 5. Z8FS040 MCU Program Memory Map**

# RAM Memory Map

There is a total of 1KB of RAM available on the base Z8F082A device. Some of this RAM (from `070h` to `0EFh` and from `190h` to `3FFh`) is used by the ZMOTION PIR Engine. The remainder of the RAM, from `000h` to `06Fh` and from `110h` to `18Fh` (240 bytes), is available to the application. The MCU Control Registers are located at the top of memory, from `F00h` to `FFFh`, and are also available to the application. The area from `400h` to `EFFh` contains no device memory. See Figure 6.

The ZMOTION Motion Detection API is a series of registers located in the RAM memory space in the address range `0F0h–10Fh`. It is through these memory locations that configuration and status are passed between the PIR Engine and the user application. Advanced API registers are located in the range `0F0h–0FFh`. See the PIR Engine and API section on page 21 for details about these API registers and to set up the project memory environment.

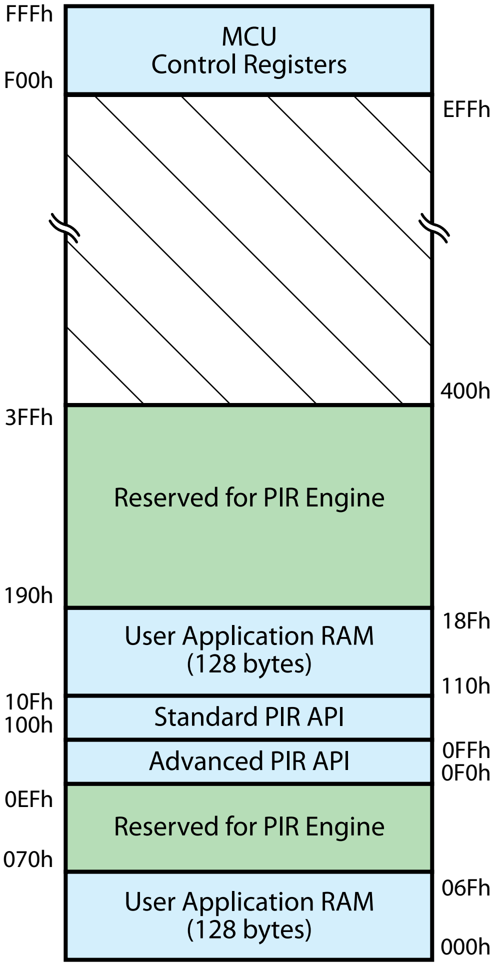

**Figure 6. Z8FS040 MCU RAM Memory Map**

# Peripherals

The following sections describe the differences, changes or limitations placed on any of the Z8FS021 peripherals or other functions from the base Z8F082A device. For more information about the operation of each peripheral, please refer to the appropriate section of the Z8 Encore! XP F082A Series Product Specification (PS0228).

## Peripheral Availability

Table 3 shows how the Z8FS040 MCU peripherals are used by Zilog's PIR technology and how these peripherals differ from their counterparts on the base Z8F082A device. The peripherals used by the PIR Engine should not be used by the application unless the Engine is disabled through the PIR Engine Enable Register.

**Table 3. Peripheral Availability**

| | **Device** | | |
|---|---|---|---|
| | **Z8FS021xSB20EG** | **Z8FS021xHH20EG** | **Z8FS021xHJ20EG** |
| Base MCU Device | Z8F082ASB020EG | Z8F082AHH020EG | Z8F082AHJ020EG |
| Pins/Package | 8-pin SOIC | 20-pin SSOP | 28-pin SSOP |
| ADC | ANA1 is used for White Light Detection when enabled.<br>ANA2 is used for PIR sensor input.<br>ANA3 is used for a second sensor input in Dual Pyro Mode. | ANA1 is used for White Light Detection when enabled.<br>ANA2 is used for PIR sensor input.<br>ANA3 is connected to ANA6/$V_{REF}$.<br>ANA3 is used for a second sensor input and ANA6 becomes available in Dual Pyro Mode. | ANA1 is used for White Light Detection when enabled.<br>ANA2 is used for PIR sensor input.<br>ANA3 is connected to $V_{REF}$.<br>ANA3 is used for a second sensor in Dual Pyro Mode. |
| $V_{REF}$ | Internal $V_{REF}$ used by the PIR Engine and set to 1V. | Internal $V_{REF}$ used by the PIR Engine and set to 1V. | Internal $V_{REF}$ used by the PIR Engine and set to 1V. |
| Timer 0 | Available to application. | Available to application. | Available to application. |
| Timer 1 | Available to application. | Available to application. | Available to application. |
| GP I/O | PA3/PA1 are multiplexed with ANA2/ANA3 and used for PIR sensor input (ANA2 for single pyro mode and ANA2/ANA3 for dual pyro mode). | PB2, PB3 & PC2 are used for PIR functions.<br>In dual pyro mode, PC2 becomes available. | PB2, PB3 & PB5 are used for PIR functions.<br>In dual pyro mode, PB5 becomes available. |

**Table 3. Peripheral Availability (Continued)**

| | | | |
|---|---|---|---|
| Low Power Op Amp | Not Available | Not Available | Not Available |
| Comparator | Available to application. | Available to application. | Available to application. |
| UART | Available to application – No CTS. | Available to application. | Available to application. |
| Temperature Sensor | Not available. | Not available. | Not available. |
| LED Drive | — | Available to application. | Available to application. |
| WDT | Available to application. | Available to application. | Available to application. |

The remainder of this section further describes the differences in application availability between the 8-pin, 20-pin and 28-pin peripheral sets.

## Analog to Digital Signal Conversion

Zilog's PIR technology requires exclusive access to the ADC peripheral to detect motion. However, ADC conversions can be requested by the application via the API (PIR Status/ Control Register 3). If it is necessary for the user application to utilize the ADC peripheral directly, the PIR Engine must first be disabled via the PIR Engine Enable Register in the API. Motion detection is not possible while the PIR Engine is disabled. When the user application is finished with the ADC peripheral, it must reenable the PIR Engine.

**8-Pin Device.** PA3 (ANA2) is reserved as the analog ADC input from the pyroelectric sensor. Therefore, ANA2 is not available for user applications. Additionally, ANA3 is used for a second sensor input in Dual Pyro Mode. All other channels are available to the user application.

| ADC Channel | Available to Application |
|---|---|
| 0 | Yes |
| 1 | Yes – used for White Light detection |
| 2 | No |
| 3 | Only in Single Pyro Mode |

**20-Pin Device.** PB2 (ANA2) is reserved as the analog ADC input from the pyroelectric sensor. Therefore, ANA2 is not available for user applications. Additionally, ANA3 and ANA6 are not available because PB3 (ANA3) must be tied directly to PC2 (ANA6/ $V_{REF}$). PC2 is configured as $V_{REF}$ output by the PIR Engine. In Dual Pyro Mode, ANA3 is used for a second sensor input rather than being tied to $V_{REF}$; ANA6/$V_{REF}$ therefore becomes available. All other channels are available to the user application.

| ADC Channel | Available to Application |
|---|---|
| 0 | Yes |
| 1 | Yes – used for White Light detection |
| 2 | No |
| 3 | No |
| 4 | Yes |
| 5 | Yes |
| 6 | Only in Dual Pyro Mode |

**28-Pin Device.** PB2 (ANA2) is reserved as the analog ADC input from the pyroelectric sensor. Therefore, ANA2 is not available for user applications. Also, ANA3 is not available because it is tied directly to PB5/$V_{REF}$. PB5 will be configured as $V_{REF}$ output by the PIR Engine. In Dual Pyro Mode, ANA3 is used for a second sensor input rather than being tied to $V_{REF}$; PB5 therefore becomes available. All other channels are available to the user application.

| ADC Channel | Available to Application |
|---|---|
| 0 | Yes |
| 1 | Yes – used for White Light detection |
| 2 | No |
| 3 | No |
| 4 | Yes |
| 5 | Yes |
| 6 | Yes |
| 7 | Yes |

## Timers

The Z8FS021 MCU offers two independent and identical 16-bit multi-function timers, Timer 0 and Timer 1; both are available to the user application.

| Timer 0 | |
|---|---|
| 8-Pin Device | T0OUT is not available in Dual Pyro Mode; it is configured as ANA3 to support a second sensor input. All other external Timer 0 functions are available for the user application. |

| | |
|---|---|
| 20-Pin Device | All external Timer 0 functions are available for the user application. |
| 28-Pin Device | All external Timer 0 functions are available for the user application. |

| Timer 1 | |
|---|---|
| 8-Pin Device | T1IN is configured as ANA2 to support the signal input from the pyroelectric sensor and is not available to the user application. All other Timer 1 functions are available. |
| 20-Pin Device | All external Timer 1 functions are available for the user application. |
| 28-Pin Device | All external Timer 1 functions are available for the user application. |

## Watchdog Timer

No changes or limitations are placed on WDT functions by the PIR Engine; the WDT is available to the user application.

## Comparator

| | |
|---|---|
| 8-Pin Device | The external pin that carries COUT is configured as ANA2 to support the signal input from the Pyroelectric sensor. However, the Comparator is still able to generate an interrupt internally without COUT. |
| 20-Pin Device | All external Comparator functions are available for the user application. |
| 28-Pin Device | All external Comparator functions are available for the user application. |

## UART

| | |
|---|---|
| 8-Pin Device | /CTS0 is configured as ANA2 to support the signal input from the Pyroelectric sensor. It is therefore not available to the user application. The UART is still able to function correctly without /CTS when CTSE in the U0CTL0 Register is set to 0. |
| 20-Pin Device | All external UART functions are available for the user application. |
| 28-Pin Device | All external UART functions are available for the user application. |

## Oscillator Control

All devices can be operated with the internal 5.54MHz IPO. For applications that require more processing power or a more accurate time base, an external crystal oscillator or ceramic resonator can be used.

When using the 8-pin device, external oscillator support is limited to Single Pyro Mode only, because ANA3 (the ADC input for a second pyro sensor) is multiplexed with $X_{OUT}$. The 20- and 28-pin devices can be operated with an external oscillator in both Single and Dual Pyro modes.

⚠ **Caution:** Do not operate at frequencies lower than the IPO frequency while the PIR Engine is enabled or motion detection performance will be degraded.

No other changes or limitations are placed on oscillator control functions by the PIR Engine.

## Flash Memory

The control registers associated with Flash memory are all available to the application. The PIR Engine uses the value programmed into the Flash Frequency registers (FFREQ) to determine the required sample rate of the ADC and other functions. The Flash Frequency High (FFREQH) and Flash Frequency Low Byte (FFREQL) registers must be programmed prior to initializing the PIR Engine. These two registers combine to form a 16-bit value, FFREQ. This value is the System Clock Frequency in KHz and is calculated using the following equation.

```
FFREQ[15:0] = {FFREQH[7:0],FFREQL[7:0]} = (System Clock
Frequency)/1000
```

## Interrupt Controller

No changes or limitations are placed on the interrupt controller functions by the PIR Engine.

## Temperature Sensor

The temperature sensor is not tested or calibrated (trim bits are not available). Therefore, this peripheral is not available on any of the Z8FS040 devices.

## Low-Power Operational Amplifier

The AMPINP signal is multiplexed with ANA2 which is used for the pyro sensor input. Therefore, this peripheral is not available on any of the Z8FS040 devices.

## Non-Volatile Data Storage

There is no dedicated non-volatile data storage on the Z8FS040 devices.

# Pin Availability

Although most pins on the ZMOTION MCU Series are available to the application, some pins are dedicated to supporting the functions of the PIR Engine. This section describes which pins are reserved and which are available to the application. The pins used by the PIR Engine are automatically configured when the Engine is initialized.

## General-Purpose Input/Output

All of the General Purpose I/Os are available except for those used for the PIR circuit. See Appendix A. Example Application Schematics on page 49 for more information.

| | |
|---|---|
| 8-Pin Device | Pin 5 (ANA2) is reserved as the analog ADC input from the pyroelectric sensor. Any other functions multiplexed with Pin 5 (PA3//CTS0, COUT and T1IN) are not available for user applications.<br>In Dual Pyro Mode, in which the application uses 2 pyroelectric sensors, Pin 3 (ANA3) is used as an analog ADC input for a second sensor and is therefore not available for other functions (T0OUT/$V_{REF}$/CLKIN). |
| 20-Pin Device | Pin 2 (ANA2) is reserved as the analog ADC input from the pyroelectric sensor. In Single Pyro Mode, Pin 3 (ANA3) must be externally tied to $V_{REF}$ on Pin 18 (PC2/ANA6/LED/$V_{REF}$). PC2 will be configured as the $V_{REF}$ output by the PIR Engine when it is enabled.<br>In Dual Pyro Mode (supporting 2 pyroelectric sensors), Pin 3 (ANA3) is used for the second sensor. In this mode the Pin 18 $V_{REF}$ signal is not connected externally to any other ADC inputs and is therefore available to the application (PC2/ANA6/LED/$V_{REF}$). |

| | |
|---|---|
| 28-Pin Device | Pin 1 (ANA2) is reserved as the analog ADC input from the pyroelectric sensor. In Single Pyro Mode, Pin 4 (ANA3) must be externally tied to $V_{REF}$ on Pin 3 (PB5/ $V_{REF}$). PB5 will be configured as $V_{REF}$ output by the PIR Engine when it is enabled.<br>In Dual Pyro Mode, which supports 2 pyroelectric sensors, Pin 4 (ANA3) is used for the second sensor. In this mode the Pin 3 $V_{REF}$ signal is not connected externally to any other ADC inputs and is therefore available to the application (PB5/$V_{REF}$). |

# Hardware Connection Requirements

This section describes the required external hardware connection for the ZMOTION MCU Series. See Appendix A. Example Application Schematics on page 49 for example schematic diagrams showing these connections.

Pins are automatically configured to their required function when the PIR Engine is initialized via the EPIR_INIT macro. The device can be operated in Single Pyro Mode when supporting one pyroelectric sensor or in Dual Pyro Mode when supporting two pyroelectric sensors. Both of these modes can be operated in Normal or Low Scan Rate modes.

Depending on the application, there can be up to four connection requirements supporting these modes: one for each of two PIR sensors, one for the ADC and one for the White Light Detection and Status LED. A description of each of these elements follows in this section. For a broader description of the White Light Detection component, see the White Light Detection section on page 19.

## Pyroelectric PIR Sensor #1

The signal from the PIR sensor is connected directly to the ANA2 input of the ADC. The ADC is configured for differential buffered mode by Zilog's PIR technology. The sensor signal should be connected directly to the ADC input with no additional signal conditioning circuitry unless specified by the pyroelectric sensor manufacturer.

## ADC $V_{REF}$

The on-chip $V_{REF}$ is configured for 1V of nominal voltage. The PIR Sensor signal is connected to the positive (+) differential input of the ADC (ANA2), and the $V_{REF}$ signal is connected to the negative (–) differential input (ANA3). The 8-pin device provides an internal connection from $V_{REF}$ to ANA3 to support this configuration; therefore, no external hardware connection is required. The 20- and 28-pin devices require an external connection from the $V_{REF}$ output signal to the ADC negative (–) (ANA3) input.

## Pyroelectric PIR Sensor #2

In Dual Pyro Mode, the ADC is still used in differential buffered mode (i.e., as is true for Single Pyro Mode). The signal from the second PIR sensor is connected to ANA3. The

$V_{REF}$ signal is no longer connected to ANA3 negative (–) ADC input. The fist PIR sensor is connected to the positive (+) ADC input (ANA2) as it is in Single Pyro Mode. The $V_{REF}$ signal is still used internally for the ADC, but the external pin is unused in Dual Pyro Mode.

## White Light Detection and Status LED

When enabled via the White Light Threshold field in the PIR Status/Control Register 2, the PIR Engine automatically performs ADC conversions on the ANA1 input. The signal on this input should be proportional to the amount of white light to which the PIR sensor is being subjected. Typically, this pin would be connected to a high-efficiency (status) LED which generates a small voltage in the presence of white light. The PIR Engine configures the pin to a (single-ended) analog input, perform the necessary ADC conversion and reconfigure it to a digital output. See the following White Light Detection section for more information about the operation of this mode.

# White Light Detection

Due to the nature of the pyroelectric sensor, sudden large changes in white light will cause a DC shift in the signal output, which could potentially generate a false motion event. Sources such as car headlights have been known to cause this issue. The PIR Engine can monitor this occurrence and be configured to ignore the event.

When light hits the pyroelectric sensor, its energy is converted into a signal output. If the pattern and intensity of the light is optimal, then the signal can appear exactly like a motion event. However, the PIR Engine can determine when the sensor is being exposed to light and use this information to filter out an invalid motion event.

When enabled via the White Light Threshold (WLT) bits in the ePIR_SC2 Register (see Table 10 on page 31), the PIR Engine automatically performs ADC conversions on ANA1, which should be connected to a high-efficiency LED (typically a status LED). When light shines on an LED, it generates a small voltage which the PIR Engine can measure. The PIR Engine reconfigures the pin to an analog input and performs the necessary ADC conversion. The system should be configured such that if light is shining on the LED, it is also shining on the PIR sensor. As a result, it is advisable to avoid placing the LED behind a lens with white light filtering.

When the Engine detects a sudden change in voltage on ANA1 greater than the value set in the White Light Threshold field of ePIR_SC2, the motion detection algorithms automatically compensate for the associated signal level shift from the pyroelectric sensor (caused by the light simultaneously shining on the LED and the pyroelectric sensor). This compensation allows the PIR Engine to suppress false motion events caused by White Light while still detecting real motion events. However, sensitivity is reduced during this period. The White Light Detected bit of the ePIR_ASC0 Register is set to indicate the detection of White Light and stays set until cleared by the application. No other action is required by the application.

Additionally, an Anti-Jam feature is available via bit 5 of the ePIR_ASC0 Register to prevent intentional jamming of the detector. When Anti-Jam is on, the Engine looks for continuous white light events occurring within a short time span (for example, a malicious intruder continuously flashing a bright light onto the detector). After 12 white light events within a short time period, the Engine automatically begins ignoring the white light events and returns to normal motion detection. The Engine begins responding to white light events after a period without white light events.

When White Light Detection Mode is enabled by programming the White Light Threshold bits in PIR Status/Control Register 2 (ePIR_SC2), the application is not required to perform any other functions. The PIR Engine performs all required tasks related to White Light immunity. The White Light Detected bit in PIR Status/Control Register 1 simply provides the application with an indication that a white light event has occurred.

## LED Requirements

The PIR Engine uses the voltage generated by the LED connected to ANA1 to determine the level of white light to which the system is being subjected. While most high-efficiency LEDs will perform correctly for this function, there are certain requirements placed on the specifications of the LED that is used in the system.

These requirements are:

- Do not place the LED behind any white light filtering material. If it is behind a lens or a light pipe, these materials should be transparent to white light.
- Ensure that the light source for the LED is originating from the same general direction as the PIR sensor. It is important that the PIR sensor and the LED receive the light at the same time.
- LEDs are available with a large range of electrical specifications. The White Light Threshold Detection bits in the ePIR_SC02 Register allow the PIR Engine the flexibility to work with many LED types, but generally LEDs that are more efficient at generating a voltage from a light source perform better as white light detectors.
- Most high-efficiency LEDs in red, yellow or green with a forward voltage drop less than 2V @2mA are well suited for white light detection.

# PIR Engine and API

The ZMOTION MCU Series is developed upon the Z8 Encore! XP-based Z8F082A MCU with the added functionality of a motion detection (PIR) Engine. The PIR Engine is located in the upper 4KB area of the 8KB device, leaving 4KB of code space to the user application. The PIR Engine operates in the background and is controlled and monitored via an Application Programmer Interface (API). The API is a series of reserved registers in memory.

There are two sections to the API: Standard API Registers and Advanced API Registers, as described below.

**Standard API Registers.** These registers include all of the status and control functions required by most applications. These include sensitivity control, motion detection/direction status and operational modes.

**Advanced API Registers.** These registers provide additional control over the PIR Engine operation and allows it to be configured to support the pyroelectric sensor and lens being used in the application.

## PIR Engine Timer Tick

Bit 7 of PIR Status/Control Register 1 provides a one-second time base for the PIR Engine to perform housekeeping operations. This bit must be set to 1, once per second by the user application. The bit is checked and cleared during the EPIR_ADC_ISR routine.

## PIR Engine Entry Points

There are two entry points to the PIR Engine that are accessed via two predefined macros – one is an initialization macro that is used to start the Engine and the other is executed upon every ADC interrupt. Both macros save and initialize the Register Pointer, perform a call to the PIR Engine entry point and then restore the Register Pointer before returning control to the application. It is the responsibility of the application software to execute these macros at the appropriate time.

**EPIR_INIT Macro.** This macro is executed to initialize the PIR Engine after reset. It is normally only executed once and is used in conjunction with the PIR Engine Enable Register in the Standard API section. The application should initialize all API registers, write the PIR Enable Pattern to the PIR Engine Enable Register, then execute this macro. ADC conversions are started by this macro.

```
EPIR_INIT Macro:
PUSHX                     RP
LDX                       RP, #%E0
CALL                      %1FFD
POPX                      RP
```

```
CPU Cycles: 261
```

Peripherals initialized: ADC and GPIO, depending on API-selected options. The ADC IRQ is set for medium priority.

**EPIR_ADC_ISR Macro.** This macro is executed for each ADC conversion. The application handles the ADC interrupt and executes this macro. All motion detection processing is performed by this macro.

```
EPIR_ADC_ISR Macro:
PUSHX                      RP
LDX                        RP, #%E0
CALL                       %1000
POPX                       RP
```

The CPU cycles used by the **EPIR_ADC_ISR** macro vary depending on Engine state and configuration.

# PIR Engine CPU Stack Usage

The PIR Engine shares the processor stack with the user application. There are no special requirements on the placement of the stack in memory, but it is essential that the user provide enough stack space for both the user application and the PIR Engine.

The PIR Engine requires a maximum 6 bytes of stack.

# Standard API Register Set

The Standard API Register set, shown in Table 4, is a series of registers implemented in the Z8FS040 MCU RAM that allows the user code to configure and communicate with the PIR Engine. The default values are loaded only when the PIR Engine is enabled via the PIR Enable Register.

**Table 4. PIR Engine Standard API Registers**

| API Register Name | Address | Mnemonic | Description |
|---|---|---|---|
| PIR Engine Enable Register (ePIR_Enable) | 100h | ePIR_Enable | Enable PIR Engine. |
| PIR Sensitivity Register (ePIR_Sensitivity) | 101h | ePIR_Sensitivity | Motion Sensitivity. |
| PIR Status/Control Register 0 (ePIR_SC0) | 102h | ePIR_SC0 | Motion Status and Engine Mode Control. |
| PIR Status/Control Register 1 (ePIR_SC1) | 103h | ePIR_SC1 | Engine Status and Control. |
| PIR Status/Control Register 2 (ePIR_SC2) | 104h | ePIR_SC2 | Range Control. |
| PIR Status/Control Register 3 (ePIR_SC3), 28-pin SSOP | 105h | ePIR_SC3 | ADC Scan Request. |
| PIR ADC Result Value (ePIR_ADC_Result) | 10Ah/10Bh | ePIR_ADC_Result | ADC Scan Result. |
| PIR Version (ePIR_Version) | 10Ch | ePIR_Version | PIR Engine Software Version. |

Tables 5 through 15 describe each of the standard API registers listed in Table 4.

**Table 5. PIR Engine Enable Register (ePIR_Enable)**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | PIR Enable/Disable Pattern | | | | | | | |
| Control | Read/Write | | | | | | | |
| Address | 100H | | | | | | | |

**PIR Enable/Disable Pattern (Bits 7–0)**

PIR Enable/Disable Register; controlled by the application.

- The PIR Enable Register controls the overall operation of the PIR Engine. As an added level of protection, there are specific 8-bit enable and disable values; all other values are reserved. Reading this register returns the last value written. Once enabled, the PIR Engine reads the application-controlled Status/Control Register values and sets the Engine-controlled values to their default states.
- To enable the PIR Engine, first write the ePIR_ENABLE_PATTERN to the PIR Enable Register, then execute the EPIR_INIT macro. See Table 6.

**Table 6. PIR Software Enable Patterns**

| Pattern | Name | Description |
|---|---|---|
| 00h | ePIR_DISABLE_PATTERN | Disables all PIR Engine functions, including motion detection. Used to temporarily or permanently shut down the Engine. |
| 11h | ePIR_ENABLE_PATTERN | Enables the PIR Engine. All primary Engine functions, as configured in Engine Status/Control Registers, are enabled. Confirmation of enabled status is provided through the Engine Disabled bit in Status/Control Register 0. |

### Table 7. PIR Sensitivity Register (ePIR_Sensitivity)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| **Field** | Sensitivity | | | | | | | |
| **Default** | U | U | U | U | U | U | U | U |
| **Control** | Read/Write | | | | | | | |
| **Address** | 101H | | | | | | | |

**Sensitivity (Bits 7)**

PIR Sensitivity Setting; controlled by the application. The PIR Sensitivity Register is used to adjust the sensitivity of the PIR Engine to target motion. Lower values produce higher sensitivity to motion, with 00h being the most sensitive and FFh being the least sensitive. The user application should load this register with the appropriate value to provide the appropriate sensitivity. This parameter acts as a motion signal duration check. A smaller sensitivity value means that the duration of a valid motion signal can be shorter. A larger sensitivity value means a valid motion signal's duration must be longer.

**Notes:**

1. The setting of this register also affects the range of detection. Lower values increase range and higher values decrease range.
2. Depending on the lens and pyroelectric sensor used, values above 3Fh can result in very limited detection.

**Table 8. PIR Status/Control Register 0 (ePIR_SC0)**

| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
|---|---|---|---|---|---|---|---|---|
| **Field** | Extended Detection | | Engine Disabled | MD Suspend | Motion Direction Control | Motion Direction | Motion Detected | PIR Stable |
| **Control** | R/W | | R | R/W | R/W | R | R/W | R |
| **Address** | 102H | | | | | | | |

### Extended Detection Level (Bits 7–6)

Sets the sensitivity level of the extended detector; controlled by the application.

- These two bits enhance the motion detection algorithms to detect slower, faster and/or more subtle motion. The Extended Detection level is selected to provide a balance between additional sensitivity while maintaining stability (no false detections). In certain applications such as lighting control, the Extended Detection level can be increased when normal motion has been detected. Extended detection is dependent on the lens pattern used. Smaller lens beams tend to provide more subtle motion detection.
- The Extended Detection level affects user control over the range provided in ePIR_SC2. As the Extended Detection level is increased, the Range setting becomes less effective.

00 = Extended Detection Level 0 – Minimum (least sensitive).

01 = Extended Detection Level 1.

10 = Extended Detection Level 2.

11 = Extended Detection Level 3 – High (most sensitive).

### Engine Disabled (Bit 5)

PIR Engine Disable/Suspend Acknowledged; controlled by the PIR Engine.

- This bit indicates the operational status of and is controlled by the PIR Engine. When the Engine is initialized and enabled by loading the PIR Enable Register with the ePIR_ENABLE_PATTERN value, this bit is cleared to indicate that the Engine is ready. When the Engine is disabled by loading the PIR Enable Register with the ePIR_DISABLE_PATTERN, it will respond by setting this bit to 1 and perform no further operations until reenabled. For the Engine to detect that it has been disabled, the user must allow the Engine ADC interrupt to run at least once after loading the PIR Enable Register with the ePIR_DISABLE_PATTERN.

0 = Engine is enabled and operational.

1 = Engine is disabled and not operational.

### MD Suspend (Bit 4)

Motion Detection Suspend; controlled by the application.

- Temporarily suspends the PIR Engine from running and places it in a very low processing overhead state and can be used when the application requires significant CPU processing power. While suspended, motion detection is disabled; however, to ensure fast recovery from this mode, ADC interrupts still occur and samples continue to be buffered. When the application clears this bit, suspend mode is exited upon the next ADC interrupt.

0 = Normal Motion Detection.

1 = Suspended Motion Detection.

### Motion Direction Control (Bit 3)

Motion Direction Control Enable; controlled by the application.

- This bit enables directional motion detection. The relative direction of the detected motion is indicated in bit 2 (Motion Direction) of this same register. When configured as a directional detector (bit 3 set to 1), direction is indicated in bit 2 as a positive or negative relative to the PIR sensor.

  0 = Standard Motion Detection Mode. Motion is detected in any direction; the Motion Direction status bit (Bit 2) is not valid.

  1 = Directional Motion Detection Mode. Motion is detected in any direction; relative direction is indicated via the Motion Direction status bit (Bit 2).
- The directional polarity of PIR sensors is arbitrary at the time of manufacturing. Therefore, it is necessary for the user application to calibrate to each individual PIR sensor using a controlled target (i.e. moving in a known direction) and internally recording the polarity to identify which polarity represents that direction.

### Motion Direction (Bit 2)

Relative Direction of Last Motion Detected; controlled by the PIR Engine.

When directional motion detection is enabled, this bit indicates the relative direction of the last motion detected. When the PIR Engine sets the Motion Detected bit in PIR Status Register 0, this bit is set or cleared to indicate the direction of the motion. The status is latched until the user application clears the Motion Detected bit.

0 = Last detected motion was negative.

1 = Last detected motion was positive.

This status bit is undefined when Motion Direction Control is disabled.

### Motion Detected (Bit 1)

Motion Detected on PIR Sensor

Set by the PIR Engine; cleared by the application.

This bit indicates that the Engine has detected a motion event. The user application should routinely check this bit to determine if motion has been detected. This bit is set by the Engine and must be cleared by the user application.

0 = No motion detected by the Engine.

1 = Motion has been detected by the Engine.

| PIR Stable (Bit 0) |
|---|
| Passive Infrared (PIR) sensor signal stabilized bit; controlled by the PIR Engine.<br>After periods of non-use, the PIR sensor will take some time to stabilize before it can be used reliably. The amount of time is dependent on the PIR Sensor being used as well as environmental conditions; it can range from a few seconds to as much as one minute. To relieve the application software from having to assume a worst-case stabilization time, the PIR Engine automatically monitors the DC offset of the PIR sensor and sets this bit when it determines that it has become stable. This bit indicates that the PIR sensor has stabilized after one of the following conditions:<br>• After initial power on (cold start).<br>• After re-enabling the Engine via PIR Enable Register.<br>• After returning from sleep mode.<br>0 = PIR sensor signal is not stable; motion detected events are not valid.<br>1 = PIR sensor signal is stable; motion detected events are valid. |

### Table 9. PIR Status/Control Register 1 (ePIR_SC1)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Engine Timer Tick | Frequency Response | | | | PIR Scan Rate | 2-Pulse Mode | Dual Pyro Enable |
| Control | R/W | Read/Write | | | | R/W | R/W | R/W |
| Address | 103H | | | | | | | |

### Engine Timer Tick (Bit 7)

PIR One-Second Timer Tick

Set by the application; cleared by the PIR Engine.

- This bit must be set to 1 one time per second by the user application to provide the Engine with a one-second tick to perform housekeeping operations relating to motion detection. The Engine will routinely poll this bit to obtain a one-second tick. This bit is cleared by the Engine.

  0 = Cleared by the PIR Engine.

  1 = A one-second interval has occurred.

### Frequency Response (Bits 6–3)

Frequency Response of PIR Engine; controlled by application

- This value determines the frequency response of the motion detection system. Higher values allow lower frequencies to be accepted by the PIR Engine. Lower values cause the Engine to ignore targets that generate lower frequencies. These targets typically include horizontally oriented objects such as pets.
- The frequency of the signal that is presented to the PIR Engine is largely dependent on the structure of the PIR lens being used (number and dispersion of beams). A lens with several evenly distributed beams provides better frequency response performance than a lens with an uneven beam distribution.

**Note:** Lower programmed values also have the effect of reducing the relative range of detection. The valid range is 0 (disabled) to a maximum value determined by the Window Size selected in the PIR Advanced Status/Control Register 2.

The maximum Frequency Response values for each window size can be summarized as:

| Window Size | Max Frequency Response Value |
|---|---|
| Small | Fh |
| Medium | Fh |
| Large | Ch |

### PIR Scan Rate (Bit 2)

PIR ADC conversion rate for the Pyroelectric Sensor; controlled by the application.

- The PIR Engine performs the necessary ADC conversions on the PIR sensor input. Each conversion generates an interrupt that is processed by the PIR Engine from the EPIR_ADC_ISR macro. The PIR Scan Rate bit determines the rate at which the ADC conversions are generated.
- In Normal Scan Rate Mode (PIR Scan Rate set to 0), the Z8FS021 ADC peripheral is set to continuous conversion mode which causes a conversion to be carried out automatically every 256 system clocks. In this mode, the application is only required to execute the EPIR_ADC_ISR macro for each ADC interrupt. The ADC continually runs and continuously generates interrupts.
- When Low Scan Rate Mode is selected by setting this bit to a 1, continuous conversion mode is disabled and the ADC is operated in single-shot mode such that each conversion takes 5129 system clocks to complete. In this mode, the application software must initiate the ADC conversion request (set bit 7 of ADCCTL0) and execute the EPIR_ADC_ISR macro once every 5ms.
- In Low Scan Rate Mode, the ADC is disabled between conversions to reduce power consumption. Power consumption can be reduced further if the application software uses this mode in conjunction with the CPU’s Halt or Stop modes. Alternately, this mode can be used to provide the application software with additional CPU processing time.
- Although the Low Scan Rate Mode provides the application with more processing power and the opportunity for the system to reduce power consumption, the normal scan rate will provide better sensitivity and range. While operating in Low Scan Rate Mode, sensitivity is reduced by approximately 20%. The performance of Direction Detection can also be reduced in this mode. EMC immunity is disabled while in Low Scan Rate Mode.
- EMC immunity features, such as transient sensitivity and noise sensitivity, are disabled while in Low Scan Rate Mode.
- If the PIR Scan Rate bit is changed during Engine operation, the Engine will stop detecting motion for up to 200ms to avoid potential false motion detection. When changing the PIR Scan Rate mode, the Advanced API registers must first be updated with the appropriate values.

0 = Normal Scan Rate Mode

1 = Low Scan Rate Mode

### 2-Pulse Enable (Bit 1)

2-Pulse Detection Mode Enable; controlled by application
This bit determines if motion detection requires one or two motion pulses as the target passes across the beams created by the Fresnel lens. With a Pulse Count of 1, a single motion pulse from the target is required to qualify as valid motion. A Pulse Count of 2 requires two motion pulses from the target – one positive and one negative – to qualify as valid motion. A Pulse Count of 2 is typically used in harsher environmental conditions to decrease the chances of false triggers from sources such as fast heating and cooling.

0 = One pulse of motion required for detection.

1 = Two pulses of motion required for detection.

### Dual Pyro Mode (Bit 0)

Dual Pyroelectric Sensor Signaling Mode; controlled by the application.

- This bit determines if the PIR Engine should accept signals from one or two pyroelectric sensors.
- When configured for single pyro operation, only one sensor is used (connected to ANA2). When configured for dual pyro operation, the Engine will scan two sensors simultaneously. Dual pyro mode is typically used to provide a larger area of coverage. The second pyroelectric sensor is connected to input ANA3. In Dual Pyro Mode, motion on either sensor will generate a motion detected event.

  0 = Single pyroelectric sensor mode.

  1 = Dual pyroelectric sensor mode.

**Table 10. PIR Status/Control Register 2 (ePIR_SC2)**

| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
|---|---|---|---|---|---|---|---|---|
| **Field** | White Light Threshold | | | | | Range Control | | |
| **Control** | Read/Write | | | | | Read/Write | | |
| **Address** | 104H | | | | | | | |

**Bits 7–3 reserved**
White Light Detection Threshold*
Controlled by application

- These bits determine how sensitive the White light detector is to changes in detected white light. Larger values cause the white light detector to be less sensitive. A value of 00000 disables White Light Detection. The light is sensed via the LED connected to the ANA1 pin. Light shining on the LED creates a small current that can generate enough voltage on the pin to measure changes accurately. The PIR Engine automatically configures the ADC to single-ended buffered mode and performs the necessary conversions and processing. See the White Light Detection section on page 19 for a description of White Light operation.

Note: *This register is ignored in Low Power Mode; White Light Detection is disabled by Lower Power Mode.

### Range Control (Bits 2–0)

Motion Detection Range Control; controlled by the application.

- These bits determine the relative range of motion detection; larger values decrease the range of detection. This parameter acts as a motion signal amplitude check. A smaller range value means that a valid motion signal’s amplitude can be lower. A larger range value means that a valid motion signal’s amplitude must be higher.
- Typical values used for range control are dependent on the lens and pyroelectric sensor being used. Range is also dependent on target size, speed and relative temperature. For example, a range control setting that rejects one target of a particular size at a given distance does not guarantee that a larger target will be rejected at the same distance.

**Table 11. PIR Status/Control Register 3, 28-Pin SSOP (ePIR_SC3)**

| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
|---|---|---|---|---|---|---|---|---|
| **Field** | ANA7 Scan Request | ANA6 Scan Request | ANA5 Scan Request | ANA4 Scan Request | Reserved | Buffered Mode | ANA1 Scan Request* | ANA0 Scan Request |
| **Control** | R/W | R/W | R/W | R/W | 0 | R/W | R/W | R/W |
| **Address** | 105H | | | | | | | |
| Note: *ANA1 is reserved for White Light detection when enabled via PIR Status/Control Register 2. | | | | | | | | |

**Table 12. PIR Status/Control Register 3, 20-Pin SSOP (ePIR_SC3)**

| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
|---|---|---|---|---|---|---|---|---|
| **Field** | Reserved | ANA6 Scan Request | ANA5 Scan Request | ANA4 Scan Request | Reserved | Buffered Mode | ANA1 Scan Request* | ANA0 Scan Request |
| **Control** | 0 | R/W Reserved in Single Pyro Mode | R/W | R/W | 0 | 0 | R/W | R/W |
| **Address** | 105H | | | | | | | |
| Note: *ANA1 is reserved for White Light detection when enabled via PIR Status/Control Register 2. | | | | | | | | |

**Table 13. PIR Status/Control Register 3, 8-Pin SOIC (ePIR_SC3)**

| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
|---|---|---|---|---|---|---|---|---|
| **Field** | Reserved | Reserved | Reserved | Reserved | ANA3 Scan Request | Reserved | ANA1 Scan Request* | ANA0 Scan Request |
| **Control** | 0 | 0 | 0 | 0 | R/W Reserved in Dual Pyro Mode | 0 | R/W | R/W |
| **Address** | 105H | | | | | | | |
| Note: *ANA1 is reserved for White Light detection when enabled via PIR Status/Control Register 2. | | | | | | | | |

**ANAx Scan Request**

Analog Channel 0, 1, 3-7 Scan Requested Bits

Set by the application; cleared by the PIR Engine.

These bits allow the user application to request the Engine to perform an A/D conversion on the non-reserved analog inputs. When requested, the Engine will reconfigure the appropriate I/O pin to a single-ended, unbuffered input using a 2 Volt reference. It will then take the next sample and store it in the PIR ADC Result Value Registers and clear all ANAx Scan Request bits. The I/O configuration for the ANAx pin is not returned to its previous configuration by the Engine. If required, the user application must perform this task.

If multiple request bits are set simultaneously, the Engine will only scan the lowest-numbered ADC channel requested and ignore any other requests. The user application should set one request bit, then poll it to determine when the conversion is complete and the data is ready.

When ADC Scan requests are being serviced by the PIR Engine, ADC conversions on the PIR sensor are suspended. Therefore, the user application should be careful not to continuously request ADC Scans. The PIR Process Rate Register in the Advanced API Register Set section can be monitored to ensure the Engine is receiving enough time to perform its required PIR Sensor ADC scans.

0 = no conversion requested/last conversion completed.

1 = perform a conversion on this channel.

**Table 14. PIR ADC Result Value (ePIR_ADC_Result)**

| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Field** | ADC Result Value | | | | | | | | | | | | | | | |
| **Control** | Read | | | | | | | | | | | | | | | |
| **Address** | 10AH | | | | | | | | 10BH | | | | | | | |

### PIR ADC Result Value (Bits 0–15)

ADC Scan Request Result Value

Controlled by PIR Engine

The PIR ADC Result Value contains the result of the last application requested ADC conversion.

The data format is identical to that discussed in the Z8 Encore! XP Product Specification (PS0228) for registers ADCD_H and ADCD_L.

Example for requesting an ANA0 Conversion:

- Set bit 0 (ANA0 Scan Request) in PIR Status/Control Register 3 (ePIR_SC3).
- Wait until the ANA0 Scan Request bit is cleared by the Engine.
- Read the ADC conversion result from the PIR ADC Result Value register.

Note: Even though the ADC Result Value is a 16 bit register, atomic operations are not required since the value is only updated at the request of the application.

**Table 15. PIR Version (ePIR_Version)**

| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
|---|---|---|---|---|---|---|---|---|
| **Field** | Version | | | | | | | |
| **Control** | Read | | | | | | | |
| **Address** | 10CH | | | | | | | |

### Version (Bits 0–7)

PIR Engine software version; controlled by the PIR Engine.

- The value stored in this register indicates the software version of the PIR Engine.

| Value | PIR Engine Software Version |
|---|---|
| 43h | 2.00 |

# Advanced API Register Set

The registers are for advanced configuration of the PIR Engine. They include customization for lens and pyroelectric sensor configurations. These registers are not initialized by the PIR Engine.

**Table 16. PIR Engine API Advanced Registers**

| API Advanced Register Name | Address | Mnemonic | Description |
|---|---|---|---|
| PIR Advanced Status/Control Register 0 | F0h | ePIR_ASC0 | Engine control and status. |
| PIR Advanced Status/Control Register 1 | F1h | ePIR_ASC1 | White Light Debounce and Scan Rate. |
| PIR Advanced Status/Control Register 2 | F2h | ePIR_ASC2 | Window Size, Lock Level, and Window Update Rate. |
| PIR Engine Process Rate | F3h/F4h | ePIR_Process_Rate | Relative Processing available to the PIR Engine. |
| PIR Sample Size Register | F5h | ePIR_Sample_Size | Controls amount of sensor signal averaging. |
| PIR Debounce Time Register | F6h | ePIR_Debounce_Time | Controls time to Debounce motion signal. |
| PIR Debounce Batch Size Register | F7h | ePIR_Debounce_Batch | Controls out of window samples required for Debounce. |
| PIR Transient Sensitivity Level | F8h | ePIR_Transient_Sense | Sets PIR Engine sensitivity to transient detection. |
| PIR Noise Sensitivity Level | F9h | ePIR_Noise_Sense | Sets PIR Engine sensitivity to noise detection. |
| PIR Pyro Signal | FAh/FBh | ePIR_Signal | Current Pyro Sensor signal sample. |
| PIR Pyro DC Signal Level | FCh/FDh | ePIR_Signal_DC | Current calculated Pyro Sensor DC offset. |
| PIR Extended Detection Sensitivity Level | FEh | ePIR_Extended_Sense | Controls sensitivity of extended detector. |
| PIR Extended Detection Debounce Timeout | FFh | ePIR_Extended_Timeout | Controls. |

Table 17. PIR Advanced Status/Control Register 0 (ePIR_ASC0)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| **Field** | External $V_{REF}$ | White Light Detected | White Light Anti-Jam | Buffer Refresh | New Sample | MD Origin | EM Noise Detected | EM Transient Detected |
| **Control** | R/W | R/W | R/W | R/W | R/W | R | R/W | R/W |
| **Address** | F0H | | | | | | | |

### External $V_{REF}$ (Bit 7)

Select between internal or external $V_{REF}$; controlled by the application.

- This bit selects between internal and external $V_{REF}$ for the ADC when scanning the PIR sensor. When set to 0, internal $V_{REF}$ is used and it is automatically set to 1V when the PIR Engine is initialized via the EPIR_INIT macro. When set to a 1, external $V_{REF}$ is selected and a regulated 1V signal must be supplied to this pin by the user. The advantage of using an external $V_{REF}$ is that it is affected less by internal operations of the MCU like high drive on an output pin.

  0 = Internal $V_{REF}$ selected.

  1 = External $V_{REF}$ selected.

Note: This register affects the $V_{REF}$ selection for ADC scans of the PIR sensor only. ADC channel scan requests made through ePIR SC3 always use an internal 2V reference.

### White Light Detected (Bit 6)

White Light Detected on LED; cleared by the application.

- This bit is set by the PIR Engine when a White Light event has been detected. This status is provided to the user application to indicate that a White Light event has occurred and associated false motion event(s) may have been suppressed by the Engine. This bit is set by the Engine and must be cleared by the user application. The threshold or amount of light required to trigger a White Light event is controlled by the White Light Threshold bits in PIR Status/Control Register 2.
- If White Light detection is disabled (White Light Threshold=00000), this bit is undefined and should be ignored by the user application.

### White Light Anti-Jam (Bit 5)

Stop malicious jamming of the detector via continuous white light events; controlled by the application.

- White Light Anti-Jam is used to prevent a repetitive white light signal from jamming motion detection. Without Anti-Jam a constant source of white light events can indefinitely inhibit motion detection. With Anti-Jam enabled, after 12 White Light events within a short duration, the Engine will begin to ignore them and return to regular motion detection. After a period of no White Light events, the Engine will begin looking for White Light events again. The feature is intended to prevent malicious jamming of the detector.

  0 = White Light Anti-Jam disabled.

  1 = White Light Anti-Jam enabled.

### Buffer Refresh (Bit 4)

Uses a fast fill algorithm to quickly refill the motion detection buffers; controlled by the application.

- This bit is used to restart motion detection by quickly reinitializing and refilling the motion detection constructed sample buffers. This can be used as a method to restore motion detection after waking up from sleep mode or it can be used to help ignore external events that can cause false detections.
- **Waking up from Sleep Mode**
  If this bit is set when the EPIR_INIT macro is executed the Engine re-fills the constructed sample buffers with a fast fill algorithm that allows it to quickly restore motion detection. Typically, a simple external wake-up circuit would be implemented that provides an unqualified motion detection signal to wake up the MCU from Sleep mode (SMR). Upon SMR, the application would set the Buffer Refresh bit, execute EPIR_INIT, and then continue with normal motion detection functions for some period of time before returning to Sleep mode. By setting this bit prior to EPIR_INIT, the Engine buffers are filled much faster enabling it to analyze the original signal seen by the external wake up circuit and determine if it is 'real' motion.
- **Ignoring False Detection Events**
  If the MCU is used to control external components (LED's, relays, Lights, Triac's, etc) a fluctuation on the power supply can be created as the external device is turned on or off. The Buffer Refresh bit can be used to ignore any false detection that could be created by these fluctuations. When the external device is turned on or off, the application can set the Buffer Refresh bit to effectively reset the motion detection history and therefore ignore any effect from the external device.

### New Sample (Bit 3)

New sample available from PIR Signal High/Low register; set by the PIR Engine, cleared by the application.

- This bit indicates that the PIR Engine has a new sensor signal input sample available that can be read by the application. This status is available as an advanced feature as the application is not normally required to read the sampled PIR sensor signal. The application must clear this bit when the sample has been read.

### MD Origin (Bit 2)

Origin of last motion detection event; controlled by the PIR Engine.

- This bit indicates how the PIR Engine detected the last Motion Detected Event. When the Engine sets the Motion Detected bit in ePIR_SC0, it also sets this bit according to which detection Engine registered the event.

  0 = Normal Motion Detector

  1 = Extended Motion Detector

### EM Noise Detected (Bit 1)

New sample available from PIR Signal High/Low Register; set by the PIR Engine, cleared by the application.

- This bit indicates that the PIR Engine has a new sensor signal input sample available that can be read by the application. This status is available as an advanced feature as the application is not normally required to read the sampled PIR sensor signal. The application must clear this bit when the sample has been read.

### EM Transient Detected (Bit 0)

EM Transient Detected on PIR Signal; set by the PIR Engine; cleared by the application.

- This bit indicates if the Engine has detected a transient on the PIR signal. This event is provided to the user application to indicate that an EM transient event has occurred and associated motion event(s) may have been suppressed by the Engine. This bit is does not have to be read for normal operation and is provided as status only. The application must clear this bit after it has been read.

**Table 18. PIR Advanced Status/Control Register 1 (ePIR_ASC1)**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | White Light Debounce | | | | White Light Scan Rate | | | |
| Control | R/W | | | | R/W | | | |
| Address | F1H | | | | | | | |

### White Light Debounce (Bits 7–4)

White light debounce time; controlled by the application.

- This value determines the amount of debounce applied to the White Light detection. White Light Debounce is the number of sequential WL samples above the threshold value set in the PIR Status/Control Register 2 required to consider it as a WL event. Larger numbers result in more stable White Light detection at the cost of slower White Light response.

  0 = Internal $V_{REF}$ selected.

  1 = External $V_{REF}$ selected.

### White Light Scan Rate (Bits 3–0)

White light (LED) scan rate; controlled by the application – see the equation that follows.

- This value determines the rate (in seconds) at which the White Light signal is sampled. WL scan is performed at a maximum rate of once every 5 PIR constructed samples when this value is set to 0. The PIR constructed sample time is determined by the PIR Sample Size Register. Each increment of 1 to the White Light Scan Rate value adds 5 constructed samples to the time between White Light scans.

Use the following equation to determine the White Light scan rate, in seconds:

$$\frac{(1280 * ePIR\_Sample\_Size + 4873) * (White\_Light\_Scan\_Rate + 1)}{System\_Clock\_Frequency}$$

Larger values create a slower sample rate, which results in slower White Light response. Smaller numbers will increase White Light event response at the cost of potentially degrading motion detection.

**Table 19. PIR Advanced Status/Control Register 2 (ePIR_ASC2)**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | Lock Level | | | Window Size | | Window Update Rate | | |
| Control | R/W | | | R/W | | R/W | | |
| Address | F2H | | | | | | | |

## Lock Level (Bits 7–5)

Controlled by the application.

- This parameter sets the minimum slope change in the signal that can be considered valid motion. This prevents small signal changes caused by environmental or $V_{CC}$ shifts from causing a false detection. Use this value in combination with PIR Sensitivity and Range Control settings to balance sensitivity and stability to the particular lens and pyroelectric sensor being used.
  - Smaller values allow subtle signals with lower slopes to be considered motion events at the expense of potential false motion events.
  - Larger values allow the system to ignore smaller signal slope changes at the expense of potentially missing smaller motion events.

## Window Size (Bits 4–3)

Controlled by the application.

- This register determines the size of the control limit window. A larger window size produces more stable control limits at the cost of additional CPU usage. If a smaller window size is used, the more frequently the window can be calculated which allows it to track the signal better.

  00 = Reserved.

  01 = Small window.

  02 = Medium window.

  03 = Large window.

## Window Update Rate (Bits 2–0)

Controlled by the application.

- This register determines how frequently the control limits are calculated. It is measured in PIR samples. A smaller number produces more frequent calculations which allow the control limits to track the signal better, at the cost of increased CPU usage. The valid range is 0 to 7.
- The window is updated every 4 + (Window Update Rate * 2) PIR samples.

Table 20. PIR Process Rate
(ePIR_Process_Rate)

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | PIR Process Rate | | | | | | | | | | | | | | | |
| Control | Read | | | | | | | | | | | | | | | |
| Address | F3H | | | | | | | | F4H | | | | | | | |

**ePIR Process Rate (Bits 15–0)**

Controlled by the PIR Engine.

- The PIR Process Rate Indicator is provided by the Engine to determine if the user application process and interrupts overhead is impacting the performance of the Engine. If the Engine process rate drops significantly, its ability to detect motion can be significantly reduced. This value is typically used at the application development stage. This number gives an indication of how much CPU time the Engine is receiving. Higher numbers are better. Generally, if the process rate drops below 0080h, the ability to detect motion could be compromised.

**Note:** The 16-bit value provided by these two 8-bit registers must be read as an atomic operation by the application. This read can be ensured by using either the CPU's ATM instruction or by disabling interrupts while reading the two 8-bit registers.

Table 21. PIR Sample Size Register
(ePIR_Sample_Size)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | PIR Sample Size | | | | | | | |
| Control | Read/Write | | | | | | | |
| Address | F5H | | | | | | | |

**PIR Sample Size (Bits 7–0)**

Controlled by the application.

- This register controls the amount of averaging that the Engine performs on the incoming PIR signal ADC samples. More averaging improves signal noise immunity at the cost of a slower sample rate.

### Table 22. PIR Debounce Timeout Register (ePIR_Debounce)

| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
|---|---|---|---|---|---|---|---|---|
| **Field** | PIR Debounce Time | | | | | | | |
| **Control** | Read/Write | | | | | | | |
| **Address** | F6H | | | | | | | |

**PIR Debounce Time (Bits 7–0)**

Controlled by the application.

- This register controls the amount of time that the Engine will wait to fully debounce a motion signal. Longer times result in detection of subtle motion at the cost of more potential false motion detections. Valid range is from 01h to FFh.
- Using a value less than the value in the PIR Sensitivity Register will result in no motion detection.

### Table 23. PIR Debounce Batch Size Register (ePIR_Debounce_Batch)

| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
|---|---|---|---|---|---|---|---|---|
| **Field** | PIR Debounce Batch Size | | | | | | | |
| **Control** | Read/Write | | | | | | | |
| **Address** | F7H | | | | | | | |

**PIR Debounce Batch Size (Bits 7–0)**

Controlled by the application.

- This register determines the number of consecutive out-of-window samples required in order to consider the sequence a valid debounce count. The field works as a mask. Increasing the mask size (i.e. more bits set to 1) will increase the noise immunity of the Engine but result in lower sensitivity to subtle motion signals.
- Valid values are 01h, 03h, 07h, 0Fh, 1Fh, 3Fh, 7Fh and FFh.

Table 24. PIR Transient Sensitivity Level Register (ePIR_Transient_Sense)

| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
|---|---|---|---|---|---|---|---|---|
| **Field** | Reserved | PIR Transient Sensitivity | | | | | | |
| **Control** | 0 | Read/Write | | | | | | |
| **Address** | F8H | | | | | | | |

### Reserved (Bit 7)

This bit is reserved and must be 0.

### Transient Sensitivity (Bits 6–0)

Controlled by the application.

- This register determines how sensitive the transient detection part of the engine is to sudden changes in the PIR signal. A lower number makes the engine more sensitive at the cost of potential rejection of large signal motion (for example, a warm target very close to the detector). The valid range is 0 (disabled) to 64h.

Table 25. PIR Noise Sensitivity Level Register (ePIR_Noise_Sense)

| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
|---|---|---|---|---|---|---|---|---|
| **Field** | Reserved | PIR Noise Sensitivity | | | | | | |
| **Control** | 0 | Read/Write | | | | | | |
| **Address** | F9H | | | | | | | |

### Reserved (Bit 7)

This bit is reserved and must be 0.

### Noise Sensitivity (Bits 6–0)

Controlled by the application – see Table 26 for values.

- This register determines how sensitive the noise detection part of the Engine is to random broadband noise in the PIR signal. A lower number makes the noise detector more sensitive at the cost of potential rejection of small-signal motion (for example, a small delta between the ambient and target temperatures, or a distant target). The valid range is 0 (disabled) to a maximum value determined by the Window Size selected in the PIR Advanced Status/Control Register 2.

Table 26 summarizes the maximum Noise Sensitivity values for each window size.

**Table 26. Maximum Noise Sensitivity Values**

| Window Size | Max PIR Noise Sensitivity Value | Typical Value |
|---|---|---|
| Small | 0Ch | 08h |
| Medium | 1Dh | 12h |
| Large | 46h | 2D |

**Table 27. PIR Signal (ePIR_Signal)**

| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Field** | PIR Signal | | | | | | | | | | | | | | | |
| **Control** | Read | | | | | | | | | | | | | | | |
| **Address** | FAH | | | | | | | | FBH | | | | | | | |

**PIR Signal (Bits 15–0)**

Controlled by the PIR Engine.

- These registers contain the last PIR signal obtained by the Engine. Each time the Engine generates a new PIR signal sample it will place it in these registers and set the New Sample bit in the PIR Advanced Status/Control 0 Register. This gives the application direct visibility to the PIR generated signal for debugging purposes.
- **Note:** The 16 bit value provided by these two 8 bit registers must be read as an atomic operation by the application. This can be ensured by either using the CPU's ATM instruction or by disabling interrupts while reading the two 8 bit registers.

Table 28. PIR DC Signal Level
(ePIR_Signal_DC)

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | PIR Signal DC | | | | | | | | | | | | | | | |
| Control | Read | | | | | | | | | | | | | | | |
| Address | FCH | | | | | | | | FDH | | | | | | | |

### PIR Signal DC (Bits 15–0)

Controlled by the PIR Engine.

- These registers contain the last PIR signal DC Level calculated by the Engine. Each time the Engine generates new control limits it will place the DC component level in these registers.
- **Note:** The 16 bit value provided by these two 8 bit registers must be read as an atomic operation by the application. This can be ensured by either using the CPU's ATM instruction or by disabling interrupts while reading the two 8 bit registers.

Table 29. PIR Extended Detection Sensitivity Level
(ePIR_Extended_Sense)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | PIR Extended Detection Sensitivity | | | | | | | |
| Control | Read/Write | | | | | | | |
| Address | FEH | | | | | | | |

### Extended Detection Sensitivity (Bits 7–0)

Controlled by the application.

- This register determines how sensitive the Extended Detection part of the Engine is to fast or slow speed targets in the PIR field of view. A lower number makes the extended detector more sensitive, at the cost of potential false motion events. The valid range is 1 (most sensitive) to 255 (least sensitive) with typical values between 6 and 25. Related control features include the PIR Extended Detection Level (PIR_SC0) and the PIR Extended Detection Debounce Timeout.

Table 30. PIR Extended Detection Debounce Timeout (ePIR_Extended_Timeout)

| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
|---|---|---|---|---|---|---|---|---|
| **Field** | PIR Extended Detection Debounce Timeout | | | | | | | |
| **Control** | Read/Write | | | | | | | |
| **Address** | FEH | | | | | | | |

**Extended Detection Debounce Timeout (Bits 7–0)**

Controlled by the application.

- This register determines how long the Extended Detection part of the Engine waits for a motion event to be confirmed. A higher number makes the extended detector wait longer, at the cost of potential false motion events. The valid range is 1 (short debounce) to 255 (long debounce) with typical values between 3 and 30. Related control features include the PIR Extended Detection Level (PIR_SC0) and the PIR Extended Detection Sensitivity Level.

# Packaging

Zilog's Intrusion Detection Solution takes advantage of the Z8FS021 MCU, which is available in the following three packages:

- 8-Pin Small Outline Integrated Circuit Package (SOIC)
- 20-Pin Small Shrink Outline Package (SSOP)
- 28-Pin Small Shrink Outline Package (SSOP)

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.

# Ordering Information

The ZMOTION Intrusion Family PIR Motion Detection Solution can be ordered as a stand-alone MCU that includes the PIR Motion Detection Engine.

## ZMOTION MCU

Table 31 displays the basic features and package styles available for each device within Zilog's line of Z8FS021 ZMOTION Intrusion Detection MCU devices.

For more information about ordering your ZMOTION Intrusion Detection device, contact your local Zilog Representative, Distributor or Sales Office. The Zilog website (www.zilog.com) lists all regional offices and provides additional product information.

**Note:** The Z8FS021 MCU is not preprogammed with application code. A generic version of the ZMOTION Intrusion Detection source code (RD0001-SC01) can be downloaded from zilog.com and programmed into the MCU.

**Table 31. Z8FS021 ZMOTION Intrusion Detection Series Part Selection Guide**

| ZMOTION MCU Part Number* | Z8 Encore! XP Base Part Number | Flash Memory | GPIO | ADC Channels | Package |
|---|---|---|---|---|---|
| Z8FS021*x*SB20EG | Z8F082ASB020EG | 2KB | 5 | 3 | 8-pin SOIC |
| Z8FS021*x*HH20EG | Z8F082AHH020EG | 2KB | 16 | 4 | 20-pin SSOP |
| Z8FS021*x*HJ20EG | Z8F082AHJ020EG | 2KB | 22 | 6 | 28-pin SSOP |
| ZMOTIONS200ZCOG | ZMOTION Intrusion Detection 20-Pin Development Kit | | | | |
| *Where x = PIR Engine Revision Identifier (see Table 32). | | | | | |

## PIR Engine Revisions

Table 32 displays the current PIR Engine versions and Part Number Engine Revision Identifier used to order the ZMOTION MCU offerings listed in Table 31.

**Table 32. PIR Engine Revision Identifiers**

| Version | Part Number Engine Revision Identifier | Description |
|---|---|---|
| 1.00 | — | Internal release. 20-pin version only. |
| 2.00 | A | Initial production release. 20-pin version only. |

# Related Documents

Additional information about the ZMOTION Family of Motion Detection MCUs can be found in the following documents, which are available from the Zilog website at www.zilog.com.

| Document Number | Description |
|---|---|
| PB0230 | ZMOTION Intrusion Product Brief |
| PS0286 | ZMOTION Lens and Pyroelectric Sensor Product Specification |
| PS0228 | Z8 Encore! XP® F082A Series Product Specification |
| WP0017 | A New PIR Motion Detection Architecture White Paper |
| WP0018 | ZMOTION Detection Lens and Pyro Sensor Configuration Guide |
| **Other ZMOTION Family Products** | |
| PS0285 | ZMOTION Detection and Control Product Specification |
| PB0225 | ZMOTION Detection and Control Product Brief |
| PS0284 | ZMOTION Detection Module Product Specification |
| PB0223 | ZMOTION Detection Module Product Brief |

Please refer to the base part number in the Z8 Encore! XP F082A Series Product Specification (PS0228) for all MCU functions, features and specifications not covered in this document.

# Appendix A. Example Application Schematics

Whether you choose to employ single or dual pyro modes using 8-pin, 20-pin or 28-pin ZMOTION Intrusion Detection packages, this appendix offers assistance toward creating your circuit design.

## Z8FS021xSB20EG (8-Pin Device)

Figure 7 shows an example circuit for the 8-pin version of the ZMOTION Intrusion Detection MCU. The interface to the pyroelectric sensor is via the dedicated input ANA2 (Pin 5).

The status LED is driven by Pin 6, which is normally configured as a GPIO by the application to control the state of the LED. When White Light Detection is enabled, the PIR Engine automatically reconfigures this pin to an ADC input to perform the necessary ADC conversion and returns it to a GPIO when conversion is complete.

Pin 2 is used as the debug input to the chip, but can be used for other functions as required. Pin 4 is set up for the Reset function, but can also be used for other functions as the application requires. Pull-up resistors (10KΩ) are provided on the Debug and Reset signals as required for the Debug interface.

The signals on pins 3 and 7 can be used as required by the application. The power supply design remains available to application requirements.

In Dual Pyro mode, the second Pyroelectric sensor is connected to Pin 3 (ANA3). All other connections remain the same.

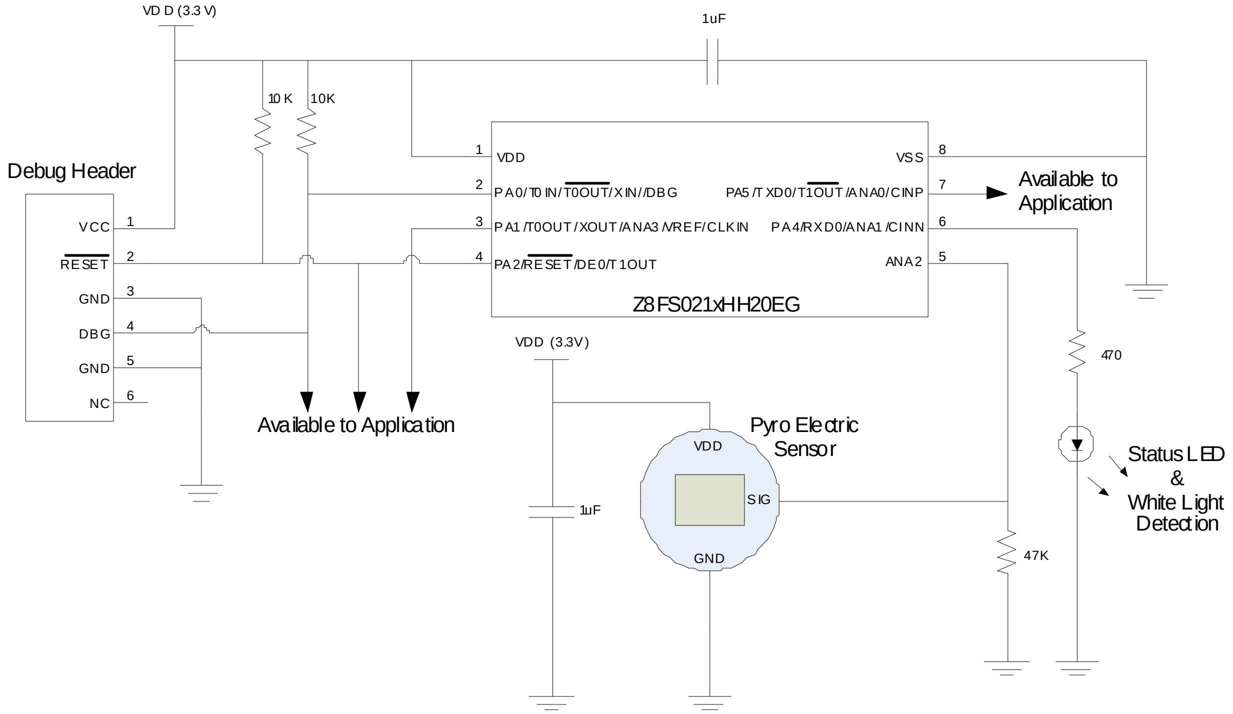

**Figure 7. Example Circuit with the Z8FS021xSB20EG (8-Pin) ZMOTION Intrusion Detection MCU**

# Z8FS021xHH20EG (20-Pin)

Examples of single and dual pyro circuit designs employing the 20-pin ZMOTION Intrusion Detection MCU are described in this section.

## Single Pyroelectric Sensor

Figure 8 shows an example circuit for the 20-pin ZMOTION Intrusion Detection MCU with a single pyroelectric sensor. The interface to the pyroelectric sensor is via the dedicated input ANA2 (pin 2).

$V_{REF}$ (pin 18) must be externally tied to ANA3 (pin 3). This is considered Internal $V_{REF}$ and bit 7 of PIR_ASC0 is set to '0' since it is being generated by the internal voltage reference. If an externally generated voltage reference is to be used, tie it directly to ANA3 (pin 3) and $V_{REF}$ (pin 18) and set bit 7 of PIR_ASC0 to '1'.

The status LED is driven by pin 1 (PB1/ANA1) which is normally configured as a GPIO by the application to control the state of the LED. When White Light Detection is enabled, the PIR Engine automatically reconfigures this pin to an ADC input to perform the necessary ADC conversion and returns it to a GPIO when done.

Pin 15 is dedicated as the Debug pin and is connected to pin 4 of the Debug Header. Pin 14 is set up for the Reset function, but can also be used as PD0 (general purpose I/O) as the application requires. Pull-up resistors (10K) are provided on the Debug and Reset signals as required for the Debug interface. All other signals can be used as required.

The power supply design remains available to application requirements.

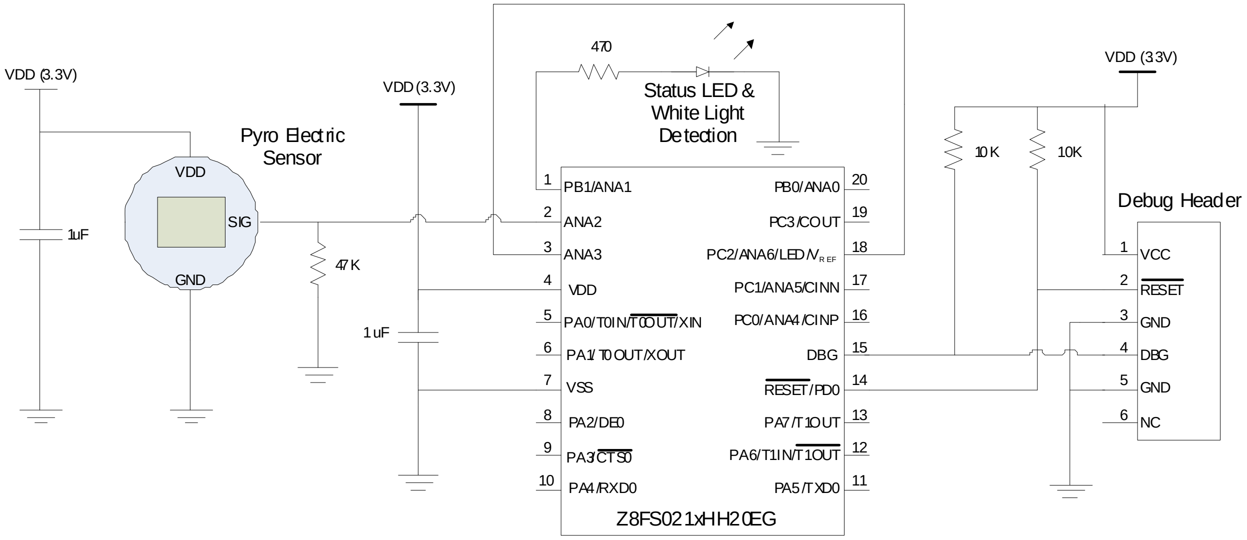

**Figure 8. Example Circuit with the Z8FS021xHH20EG (20-Pin) ZMOTION Intrusion Detection MCU, Single Pyro Mode**

## Dual Pyroelectric Sensors

In Dual Pyro mode, the second pyroelectric sensor is connected to ANA3. The signal from $V_{REF}$ to ANA3 is not required. In this design example, the internal $V_{REF}$ is used with PIR_ASC0 bit 7 set to '0'. The user could choose to use an externally generated $V_{REF}$ source in which case it would be connected to pin 18 and PIR_ASC0 bit 7 would be set to '1'. All other connections remain the same as Single Pyro Mode. See Figure 9.

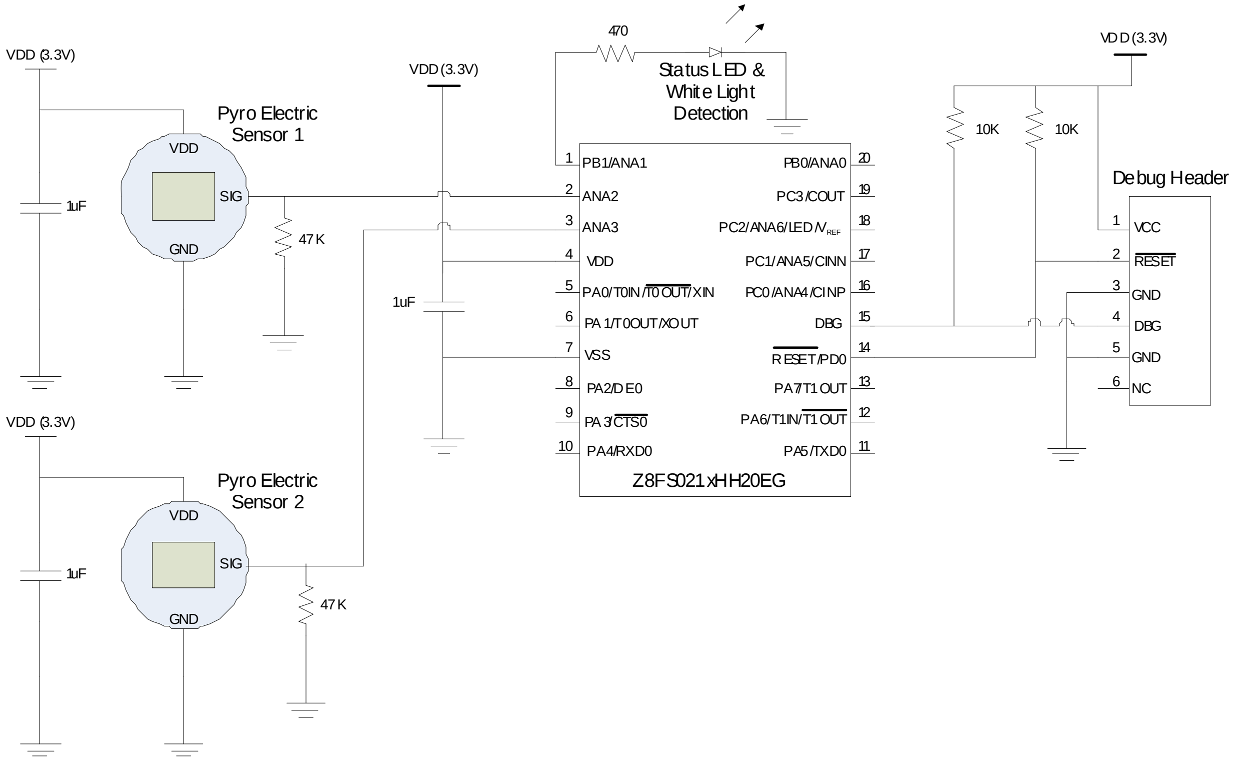

**Figure 9. Example Circuit with the Z8FS021xHH20EG (20-Pin) ZMOTION Intrusion Detection MCU, Dual Pyro Mode**

# Z8FS021xHJ20EG (28-Pin)

Examples of single and dual pyro circuit designs employing the 28-pin ZMOTION Intrusion Detection MCU are described in this section.

## Single Pyroelectric Sensor

Figure 10 shows an example circuit for the 28 pin device of the ZMOTION Intrusion Detection MCU with a single pyroelectric sensor. The interface to the pyroelectric sensor is via the dedicated input ANA2 (pin 1). $V_{REF}$ (pin 3) must be externally tied to ANA3 (pin 4). This is considered Internal $V_{REF}$ and bit 7 of PIR_ASC0 is set to '0' since it is being generated by the internal voltage reference. If an externally generated voltage reference is to be used, tie it directly to ANA3 (pin 4) and $V_{REF}$, (pin 3) and set bit 7 of PIR_ASC0 to '1'.

The status LED is driven by pin 28 (PB1/ANA1) which is normally configured as a GPIO by the application to control the state of the LED. When White Light Detection is enabled, the PIR Engine automatically reconfigures this pin to an ADC input to perform the necessary ADC conversion and returns it to a GPIO when done.

Pin 22 is dedicated as the Debug pin and is connected to pin 4 of the Debug Header. Pin 21 is set up for the Reset function, but can also be used as PD0 (general purpose I/O) as the application requires. Pull-up resistors (10K) are provided on the Debug and Reset signals as required for the Debug interface. All other signals can be used as required.

The power supply design remains available to application requirements.

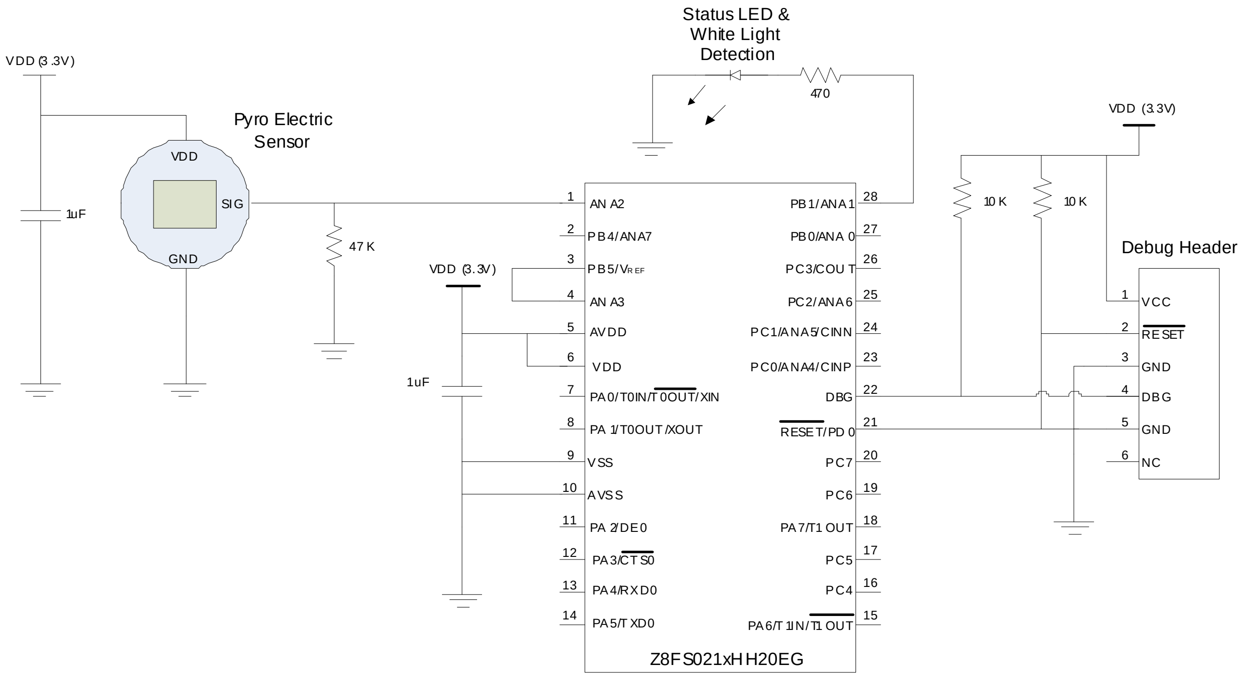

**Figure 10. Example Circuit with the Z8FS021xHJ20EG (28-Pin) ZMOTION Intrusion Detection MCU, Single Pyro Mode**

## Dual Pyroelectric Sensors

In Dual Pyro mode, the second pyroelectric sensor is connected to ANA3. The signal from $V_{REF}$ to ANA3 is not required. In this design example, the internal $V_{REF}$ is used with PIR_ASC0 bit 7 set to 0. The user could choose to use an externally generated $V_{REF}$ source in which case it would be connected to pin 3 and PIR_ASC0 bit 7 would be set to 1. All other connections remain the same as Single Pyro Mode. See Figure 11.

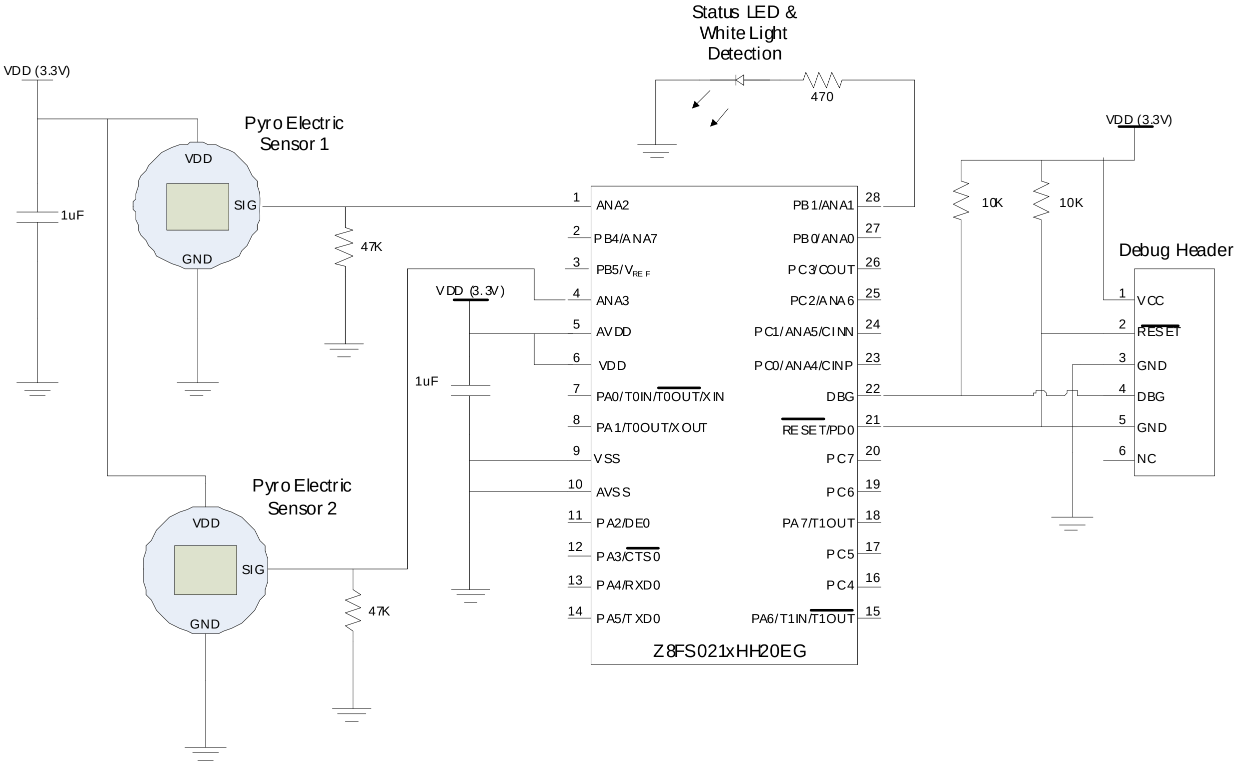

**Figure 11. Required Circuit Connections for the Z8FS021xHJ20EG (28-Pin) ZMOTION Intrusion Detection MCU, Dual Pyro Mode**

# Appendix B. PIR Engine Initialization and Control

The application software must execute an initialization procedure to enable the PIR Engine. Once the PIR Engine is enabled, it runs in the background from the ADC interrupt. Every ADC conversion generates an interrupt and the PIR Engine performs its functions during this time. The user application code runs in the foreground and monitors the status through the API and performs any other functions required for the application.

The PIR Engine also requires a one-second tick to perform several house-keeping operations and to keep track of its sampling rate. This one-second time base must be provided by the user application through Status/Control Register 1 (Engine Timer Tick). This bit should be set to 1 at the rate of once per second by the application software to provide the Engine with a one-second time base. The accuracy of this timing is not critical, but should be within +/– 10%.

There are two basic modes in which the PIR Engine operates: Normal Scan Rate mode and Low Scan Rate mode. See the description of the PIR Scan Rate bit in the PIR Status/Control Register 1 on page 29 for more details.

The PIR Engine runs in the background from the ADC interrupt (initiated by the application). Engine processing is done during the ADC interrupt. Therefore, CPU loading is based on the sample rate of the ADC. To ensure a consistent sample rate, the Engine must know the MCU operating (system clock) frequency. It uses the Flash Frequency Control Registers to determine the operating frequency, which must be initialized prior to starting the Engine.

The Flash Frequency High (FFREQH) and Flash Frequency Low Byte (FFREQL) registers combine to form a 16-bit value FFREQ primarily to control timing for Flash program and erase functions. This value is also used by the PIR Software Engine to calculate the required sample rate of the ADC and other functions. The 16-bit value for FFREQ is the System Clock Frequency in KHz and is calculated using the following equation:

FFREQ[15:0] = {FFREQH[7:0],FFREQL[7:0]} = (System Clock Frequency)/1000

The process for initializing the PIR Engine is common to both Normal Scan Rate and Low Scan Rate modes. The sequence goes as follows:

1. Set up the API control registers (standard and advanced).
2. Initialize the FFREQH and FFREQL registers with the MCU clock frequency.
3. Write the PIR Enable Pattern to the PIR Enable Register.
4. Call the PIR Init.
5. Initialize any application-specific I/Os and peripherals.
6. Enable interrupts.
7. Ensure that the PIR Sensor Stable bit (PIR_SC0:0) is set.
8. Continue with the application.

The flow diagram shown in Figure 12 displays the general software operation for Normal Scan Rate mode.

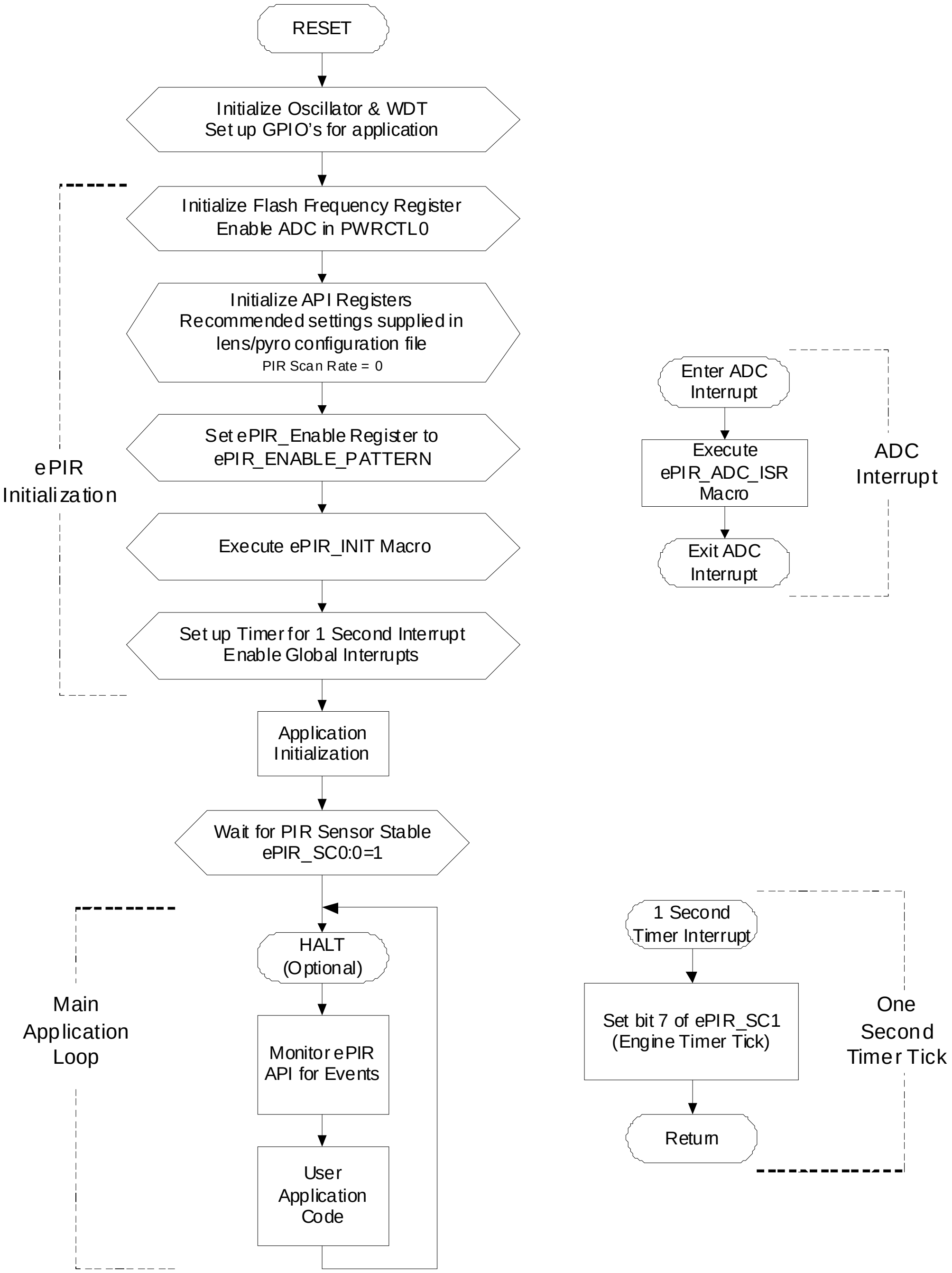

**Figure 12. Application Flow Diagram, Normal Scan Rate**

The flow diagram shown in Figure 13 displays the general software operation for Low Scan Rate mode.

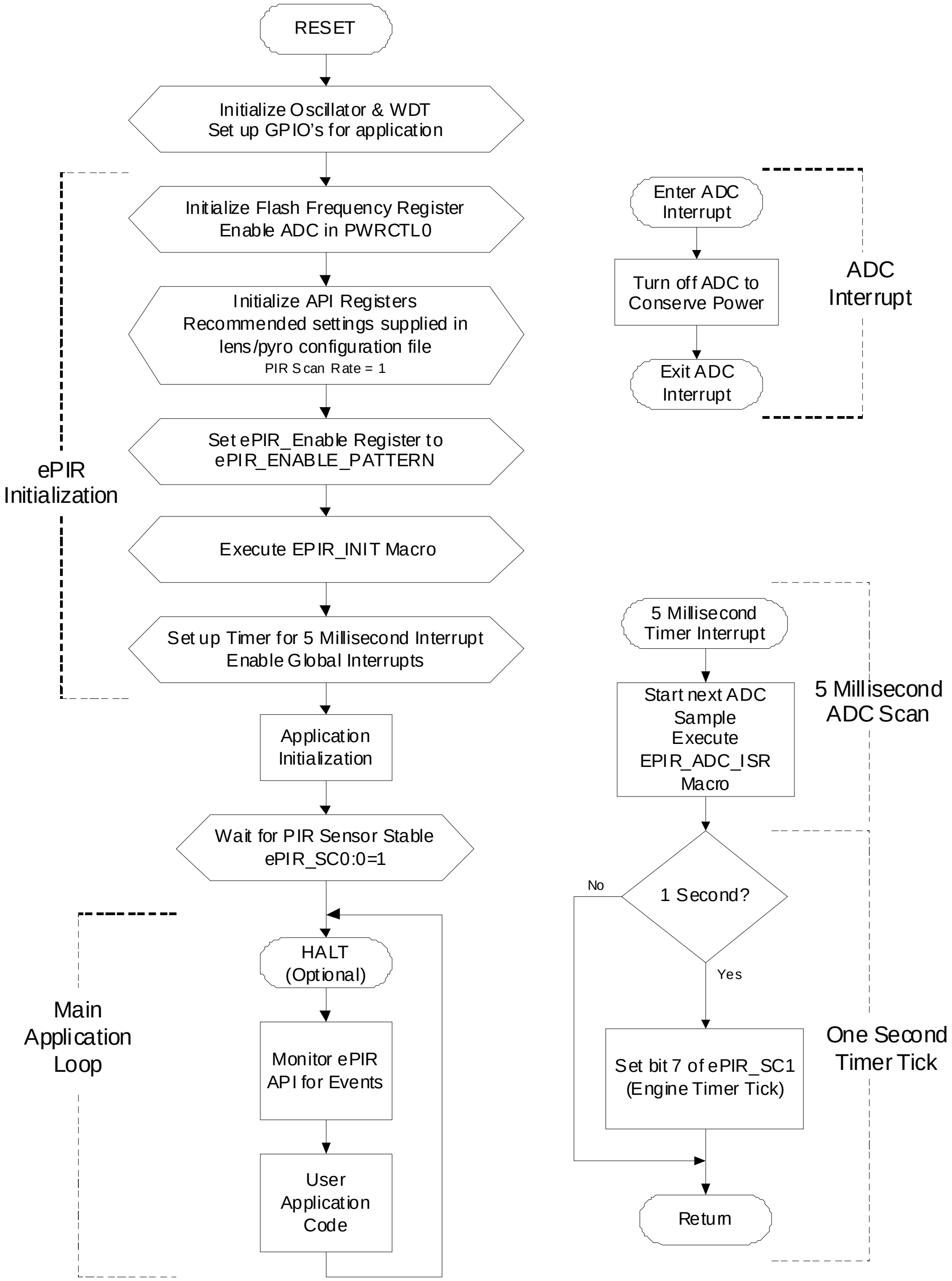
RESET
Initialize Oscillator & WDT
Set up GPIO's for application
Initialize Flash Frequency Register
Enable ADC in PWRCTL0
Initialize API Registers
Recommended settings supplied in lens/pyro configuration file
PIR Scan Rate = 1
Set ePIR_Enable Register to ePIR_ENABLE_PATTERN
Execute EPIR_INIT Macro
Set up Timer for 5 Millisecond Interrupt
Enable Global Interrupts
ePIR Initialization
Application Initialization
Wait for PIR Sensor Stable
ePIR_SC0:0=1
HALT (Optional)
Monitor ePIR API for Events
User Application Code
Main Application Loop
Enter ADC Interrupt
Turn off ADC to Conserve Power
Exit ADC Interrupt
ADC Interrupt
5 Millisecond Timer Interrupt
Start next ADC Sample
Execute EPIR_ADC_ISR Macro
5 Millisecond ADC Scan
1 Second?
No
Yes
Set bit 7 of ePIR_SC1 (Engine Timer Tick)
One Second Timer Tick
Return

**Figure 13. Application Flow Diagram, Normal Scan Rate**

# Appendix C. Software Support Files and Project Configuration

The following four files are provided to support the PIR Engine:

**PIR_API.c.** Contains the API register definitions and locates them at there appropriate places in memory.

**PIR_API.h.** Provides the bit definitions for the API registers and also contains the macro definitions for EPIR_INIT and EPIR_ADC_ISR.

**API_INIT_xx.h.** This header file contains the default API settings specific to the lens and pyroelectric sensor being used. The application code loads the API registers with these values prior to executing the EPIR_INIT macro. Several versions of these files are available from the Zilog website with tested configurations supporting the available lenses and pyroelectric sensors. These header files are included as the source code associated with the ZMOTION Detection Lens and Pyro Sensor Configuration Guide (WP0018). Refer to WP0018 and to the ZMOTION Lens and Pyroelectric Sensor Product Specification (PS0286) to select the appropriate API_INIT_xx file for the selected lens.

**startupePIR.asm.** This C start-up file replaces `startups.asm` or `startupl.asm` in ZDSII. It contains the environment initialization, stack, and register pointer configurations specifically required for a ZMOTION project.

## ZDSII Project Settings

The Zilog Developer Studio Integrated Development Environment (ZDSII IDE) is used for software development. Because the compiled application code has no vision into the operation of the PIR Engine, it is important to ensure that the application's working RAM area is not affected by Engine operations. To facilitate this, the PIR Engine uses working register group E (in the address range `E0h–EFh`) as its working RAM area, and the application code uses working register group 0 (as defined in `startupePIR.asm`). These operations are automatically handled by the compiler, and examples are provided with the available sample projects. The *Small Memory Model* must be used for the application software.

# Customer Support

To share comments, get your technical questions answered, or report issues you may be experiencing with our products, please visit Zilog's Technical Support page at http://support.zilog.com.

To learn more about this product, find additional documentation, or to discover other facets about Zilog product offerings, please visit the Zilog Knowledge Base at http://zilog.com/kb or consider participating in the Zilog Forum at http://zilog.com/forum.

This publication is subject to replacement by a later edition. To determine whether a later edition exists, please visit the Zilog website at http://www.zilog.com.