Welcome to E-XFL.COM

## Embedded - Microcontrollers - Application Specific: Tailored Solutions for Precision and Performance

Embedded - Microcontrollers - Application Specific represents a category of microcontrollers designed with unique features and capabilities tailored to specific application needs. Unlike general-purpose microcontrollers, application-specific microcontrollers are optimized for particular tasks, offering enhanced performance, efficiency, and functionality to meet the demands of specialized applications.

## What Are Embedded - Microcontrollers - Application Specific?

| Details | |
|---|---|
| Product Status | Active |
| Applications | Infrared Motion Detection |
| Core Processor | eZ8® |
| Program Memory Type | FLASH (4kB) |
| Controller Series | Z8 Encore! XP® |
| RAM Size | 256 x 8 |
| Interface | UART/USART |
| Number of I/O | 16 |
| Voltage - Supply | 2.7V ~ 3.6V |
| Operating Temperature | -40°C ~ 105°C |
| Mounting Type | Surface Mount |
| Package / Case | 20-SSOP (0.209", 5.30mm Width) |
| Supplier Device Package | 20-SSOP |
| Purchase URL | https://www.e-xfl.com/product-detail/zilog/z8fs040bhh20eg |

# Table of Contents

# List of Tables

# Overview

Zilog's ZMOTION Detection and Control Family of products provides an integrated and flexible solution for motion detection applications based on Passive Infrared (PIR) technology, including a high-performance ZMOTION microcontroller with integrated motion detection algorithms and a selection of lenses and PIR sensors to fit a wide range of application requirements. Optimized configuration parameters for the MCU are provided for each lens/sensor combination to ensure the best possible performance while significantly reducing development risk and minimizing time to market.

Depending upon your application requirements, the ZMOTION Detection and Control Family offers a broad range of solutions, from a simple combination of the Z8FS040 MCU and an API to a full set of pyros and lenses that are bundled with the Z8FS040 MCU. The ZMOTION MCU is also packaged as a complete motion detection solution, the ZMOTION Module.

Zilog's Z8FS040 MCU combines the programmability and rich peripheral set of Zilog's Z8 Encore! XP® Flash MCUs with built-in motion detection software algorithms to provide the functions necessary for PIR motion detection applications. These motion detection algorithms comprise Zilog's PIR technology and run in the background while control and status of the PIR Engine is accessed through a software API. As a result, the designer can create application-specific software while taking advantage of Zilog's ZMOTION Motion Detection Technology.

API settings are provided to match the Engine operation to each of the lens and pyroelectric sensor combinations provided.

The Flash in-circuit programming capability of the Z8FS040 allows for faster development time, more flexible manufacturing and firmware changes in the field.

Zilog's PIR motion detection technology provides a dramatic improvement in both sensitivity and stability over traditional designs and is scalable to many market segments including Lighting Control, HVAC, Access Control, Vending, Display, Proximity, Power Management, Occupancy Sensing and many others.

## Features

Key features of the Z8FS040 MCU include:

- High performance eZ8® CPU core
- 4KB in-circuit programmable Flash available for application code
- Single-pin debug with unlimited breakpoints
- Flexible clocking scheme

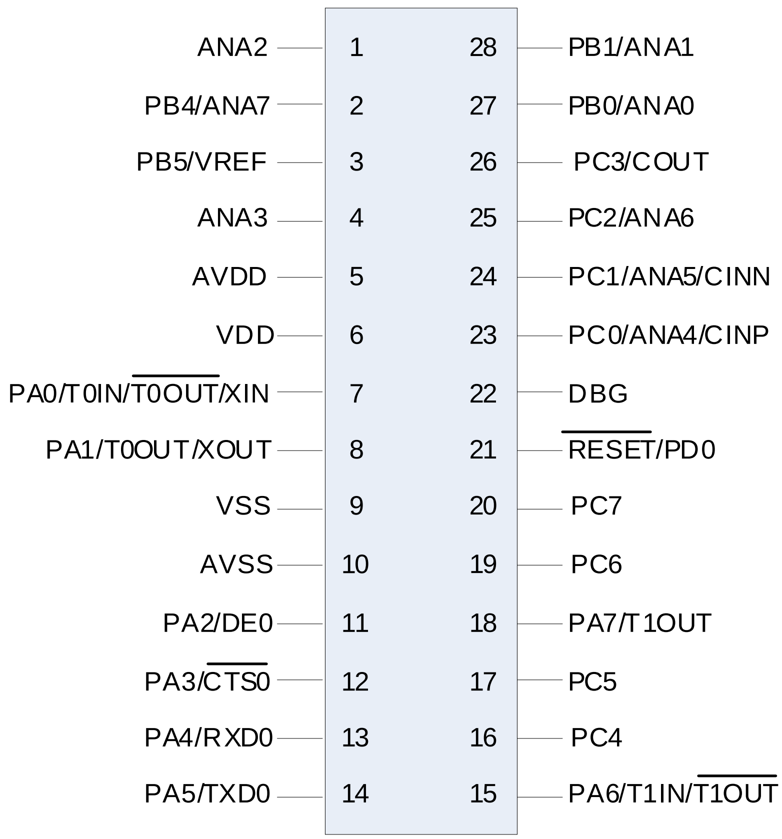



**Figure 4. 28-Pin SSOP Package Diagram – Z8FS040xHJ20EG**

# Signal Descriptions

At reset, all port pins are set to the GPIO input state on the 8-pin SOIC package except for $\overline{\text{RESET}}$/DE0/$T1_{OUT}$, which is configured to $\overline{\text{RESET}}$, and PA0/$T0_{IN}$/$T0_{OUT}$/$X_{IN}$/DBG, which is configured to DBG. On the 20- and 28-pin SSOP packages, $\overline{\text{RESET}}$/PD0 is configured to $\overline{\text{RESET}}$.

Table 3 describes the Z8FS040 Series signals.

**Table 3. Z8FS040 MCU Signal Descriptions**

| Signal 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** | | |
| $T0_{OUT}$/$T1_{OUT}$ | O | Timer Output 0–1. These signals are outputs from the timers. |
| $T0_{OUT}$/$T1_{OUT}$ | O | Timer Complement Output 0–1. These signals are output from the timers in PWM DUAL OUTPUT Mode. |
| $T0_{IN}$/T1IN | I | Timer Input 0–1. These signals are used as the capture, gating and counter inputs. |
| **Comparator** | | |
| $C_{IN}$P/$C_{IN}$N | I | Comparator Inputs. These signals are the positive and negative inputs to the comparator. |
| $C_{OUT}$ | O | Comparator Output. |

not available since 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 second sensor input rather than being tied to $V_{REF}$, and therefore ANA6/$V_{REF}$ becomes available. All other channels are available to the user application.

| ADC Channel | Available to Application |
|---|---|
| 0 | Yes |
| 1 | Yes |
| 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 since 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 $V_{REF}$, and PB5 therefore becomes available. All other channels are available to the user application.

| ADC Channel | Available to Application |
|---|---|
| 0 | Yes |
| 1 | Yes |
| 2 | No |
| 3 | No |
| 4 | Yes |
| 5 | Yes |
| 6 | Yes |
| 7 | Yes |

**Table 5. PIR Engine Standard API Registers (Continued)**

| | | | |
|---|---|---|---|
| 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 |

**Table 6. 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 7.

**Table 7. PIR Software Enable Patterns**

| Pattern | Name | Description |
|---|---|---|
| 00h | ePIR_DISABLE_PATTERN | Disables all 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 Engine Disabled bit in Status/Control Register 0. |

**Reserved (Bit 1)**

**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 11. PIR Status/Control Register 2 (ePIR_SC2)**

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

**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.
- Typical values used for Range are dependant 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 12. PIR Status/Control Register 3 (ePIR_SC3), 28-Pin SSOP**

| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
|---|---|---|---|---|---|---|---|---|
| **Field** | ANA7 Scan Request | ANA6 Scan Request | ANA5 Scan Request | ANA4 Scan Request | Reserved | Reserved | ANA1 Scan Request | ANA0 Scan Request |
| **Control** | R/W | R/W | R/W | R/W | 0 | 0 | R/W | R/W |
| **Address** | 105H | | | | | | | |

**Table 15. 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 15–0)**

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 16. PIR Version (ePIR_Version)**

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

**Version (Bits 7–0)**

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 |
|---|---|
| 03h | 2.00 |

# Advanced API Register Set

The registers listed in Table 17 are available for advanced configuration of the PIR engine. They include customizations for lens and pyroelectric sensor configurations. These registers, each described in this section, are not initialized by the PIR engine.

**Table 17. PIR Engine Advanced Registers**

| API Advanced Register Name | Address | Mnemonic | Description |
|---|---|---|---|
| PIR Advanced Status/Control Register 0 (ePIR_ASC0) | F0h | ePIR_ASC0 | EM noise and MD origin status |
| PIR Advanced Status/Control Register 2 (ePIR_ASC2) | F2h | ePIR_ASC2 | Window Size, Lock Level, and Window Update Rate |
| PIR Process Rate (ePIR_Process_Rate) | F3h/F4h | ePIR_Process_Rate | Relative Processing available to PIR engine |
| PIR Sample Size Register (ePIR_Sample_Size) | F5h | ePIR_Sample_Size | Controls amount of sensor signal averaging |
| PIR Debounce Time Register (ePIR_Debounce) | F6h | ePIR_Debounce_Time | Controls time to Debounce motion signal |
| PIR Debounce Batch Size Register (ePIR_Debounce_Batch) | 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 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 |

#### 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 PIRStatus0, 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)

EM Noise Detected on PIR Signal

Set by the PIR engine; cleared by the application.

- This bit indicates if the engine has detected noise on the PIR signal. This event is provided to the user application to indicate that an EM noise event has occurred and associated motion event(s) may have been suppressed by the engine. This bit 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.

#### 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 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 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.

**Table 25. PIR Noise Sensitivity Level (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)**

**Noise Sensitivity (Bits 6–0)**

Controlled by the application.

This register determines how sensitive the noise detection part of the engine is to random 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 ambient and target temperature or 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. See Table 26.

**Table 26. Noise Sensitivity as determined by Window Size**

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

# Packaging

Zilog's ZMOTION Detection and Control Family takes advantage of the Z8FS040 MCU, which is available in the following 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 Detection and Control Series comprises a number of product combinations that include the ZMOTION MCU plus a number of selectable lens and pyroelectric sensor options. Construct your part number based on the specific combination of MCU, lenses and PIR sensors you wish to order.

Each character in the Zilog part numbering schema corresponds to a designated part attribute. To aid in determining the appropriate part(s) to order, Table 29 breaks down a typical ZMOTION product number (as differentiated from an MCU part number) by character position to include the specific ZMOTION product, its package, and any lens and pyro options you choose. Each of these character positions is further described in Tables 30 through 34.

**Table 29. Part Number Designations**

| Position | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Field | Z | M | O | T | MCU | | MCU Package | | Lens | | PIR | | G |
| | | | | | Selectable Options | | | | | | | | |

**Table 30. Positions 1–4**

| | |
|---|---|
| ZMOT | The ZMOTION Product Family. |

**Table 31. Positions 5–8: MCU and MCU Package Selector***

| MCU Part Number | Description | PIR Software Revision | MCU Field** (Pos 5, 6) | MCU Package Field (Pos 7, 8) |
|---|---|---|---|---|
| Z8FS040xSB20EG | Occupancy, 8 pin SOIC | 2.00 | 0B | SB |
| Z8FS040xHH20EG | Occupancy, 20 pin SSOP | 2.00 | 0B | HH |
| Z8FS040xHJ20EG | Occupancy, 28 pin SSOP | 2.00 | 0B | HJ |

Note: *To purchase the ZMOTION MCU alone, select the appropriate ZMOTION MCU part number from the first column of this table.
**The second character in the MCU field refers to the PIR software engine revision.

**Table 32. Positions 9–12: Lens and PIR Sensor Selector**

| Manufacturer | Part Number | Description | Lens Field (Pos 9, 10) | PIR Sensor* | PIR Field (Pos 11, 12) |
|---|---|---|---|---|---|
| Fresnel Technologies | AA 0.9 GI T1 | Animal Alley Array (88°) | 0A | RE200B-P | 0A |
| | | | | SDA02-54-P | 0B |
| Fresnel Technologies | CM 0.77 GI V3 | Ceiling Mount Array (360°) | 0B | RE200B-P | 0A |
| | | | | SBDI46-504AA | 0C |
| Fresnel Technologies | CM 0.77 GI V5 | Ceiling Mount Array (360°) | 0C | RE200B-P | 0A |
| | | | | SBDI46-504AA | 0C |
| Fresnel Technologies | CWM 0.5 GI V1 | Ceiling/Wall Mount Array (360°) | 0D | RE200B-P | 0A |
| | | | | SBDI46-504AA | 0C |
| Nicera | NCL-9(26) | Clip-on 15mm Array (360°) | 1A | RE200B-P | 0A |
| | | | | SBDI46-504AA | 0C |
| | NCL-10IL | 10mm wall mount array (70°) | 1B | RE200B-P | 0A |
| | NCL-3B | 10mm wall mount array (40°) | 1C | RE200B-P | 0A |
| | NCL-3R | 10mm ceiling/wall array (360°) | 1D | RE200B-P | 0A |
| | | | | SBDI46-504AA | 0C |
| | NCL-10S | 10mm ceiling/wall array (18°) | 1E | RE200B-P | 0A |

Note: See Table 34 for an additional description of these PIR sensors.

**Table 33. Position 13**

| | |
|---|---|
| G | RoHS-compliant. |

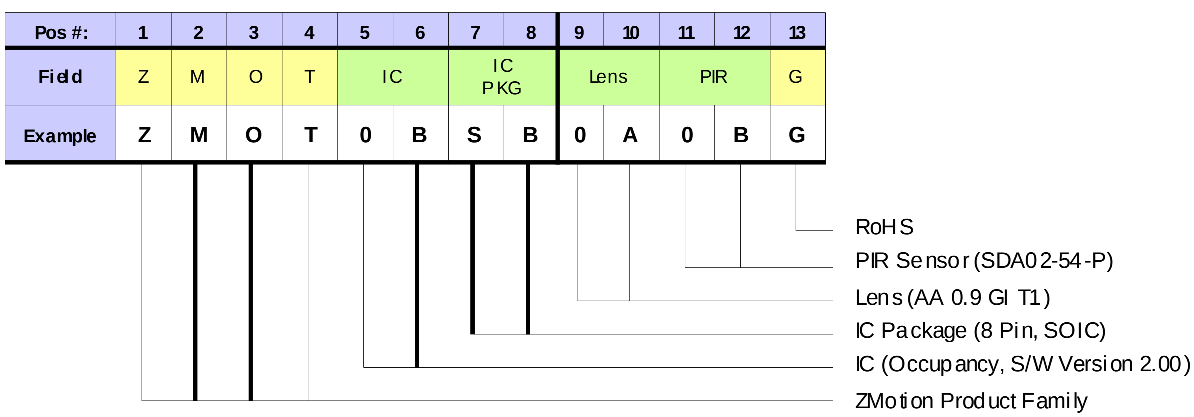
**Table 34. PIR Sensor Information**

| Manufacturer | Part Number | Description |
|---|---|---|
| Nicera | RE200B-P | Basic Dual Element |
| Nicera | SDA02-54-P | Premium Dual Element |
| Nicera | SBDI46-504AA | Quad Element |

Refer to the ZMOTION Lens and Pyroelectric Sensor Product Specification (PS0286) for detailed descriptions about the lens and pyroelectric sensors used in the above ZMOTION products.

## Ordering Example

The figure helps determine the part for an example 8-pin SOIC ZMOTION product bundled with an 88° Fresnel Technologies Animal Alley Array Lens and a Nicera Premium Dual Element PIR Sensor; the resulting ZMOTION product number is ZMOT0BSB0A0BG.




To learn more about ordering the ZMOTION that's right for your application requirements, please consult your local Zilog Sales office. The Zilog Worldwide Sales Locations page on zilog.com lists all regional offices and can connect you to additional product information.

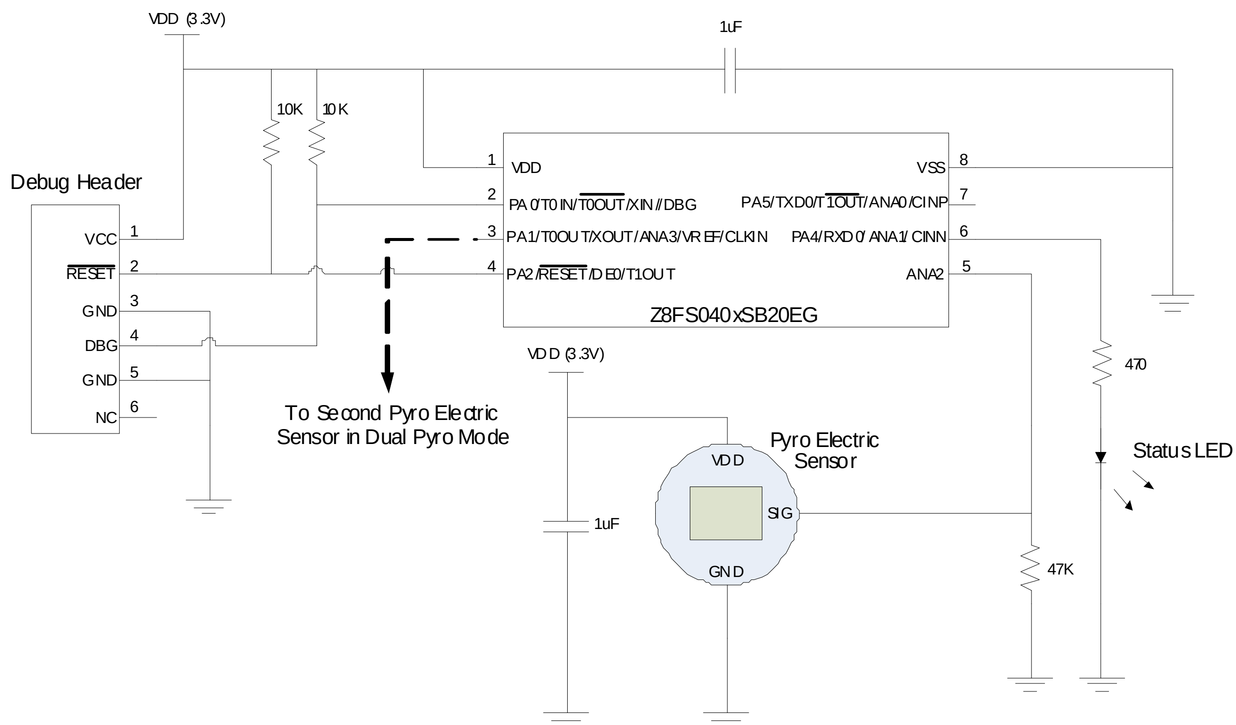



**Figure 7. Required Circuit Connections for the Z8FS040xSB20EG(8-Pin) Motion Detection MCU**

# 20-Pin Z8FS040xHH20EG

The 20-pin Z8FS040xHH20EG part offers both dual and quad pyroelectric sensors; each of these modes is described in this appendix.

## Single Pyroelectric Sensors

Figure 11 shows an example circuit for the 20 pin device of the ZMOTION Detection and Control MCU Family with a single Pyro Electric 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). The status LED is driven by pin 19 (PC3/$C_{OUT}$) which is normally configured as a GPIO by the application to control the state of the LED. This pin provides a programmable constant current sink specifically for LED drive without using an external resistor. 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 may 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 may be used as required. The power supply design is left to the application requirements.

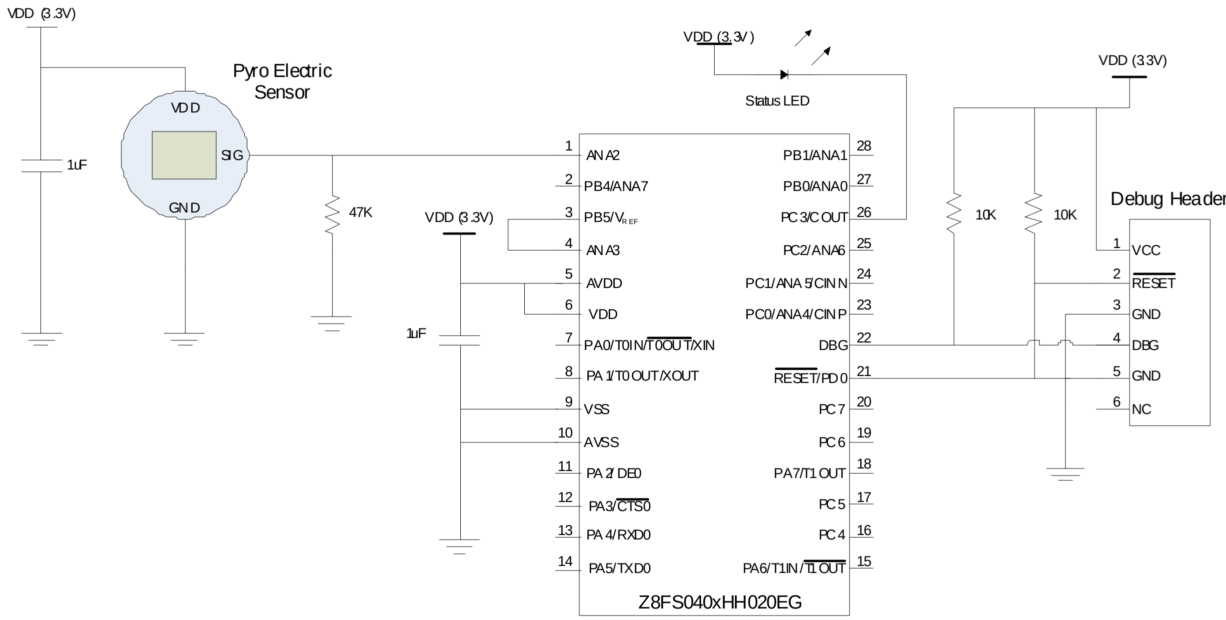



**Figure 10. Required Circuit Connections for the Z8FS040xHJ20EG (28-Pin) Motion Detection MCU in SINGLE PYRO Mode**

# 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 housekeeping operations and to keep track of its sampling rate. This tick must be provided by the user application through the Status/Control Register 1 (Engine Timer Tick). Once per second, this bit should be set to a 1 by the application software to provide the engine with a 1-second time base. The accuracy of this time 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 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 frequency (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
```

Observe the following procedure to initialize the PIR engine – a process that is common to both the Normal Scan Rate and Low Scan Rate modes:

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 PIR Init.

# Appendix C. Software Support Files and Project Configuration

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

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

**ePIR_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 this file are available from the Zilog website with tested configurations supporting the available lenses and pyroelectric sensors. Refer to Appendix D. Lens Selection Guide on page 55 to select the appropriate `API_INIT_xx` file for the selected lens.

**startupePIR.asm.** This is the C startup file that replaces startups.asm or startupl.asm in ZDS II. It contains the environment initialization, stack and register pointer configurations required specifically for a PIR project.

## ZDS II Project Settings

Zilog Developer Studio (ZDS II) is used for software development. Since the compiled application code has no vision into the operation of the PIR engine, it is important to ensure that the application working RAM area is not effected by engine operations. To facilitate this, the PIR engine uses working register group E (addresses `E0h` to `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.

To support the defined memory map, ZDS II project settings must be configured as follows (sample projects are available that have these settings already configured).

### Application Project Settings (Small Model)

- RData: `20h–6Fh`, `F0h–FFh`
  - Defined in ZDS II Project Settings under Linker Address Spaces
  - This allows for 16 bytes of stack space starting at `7Fh`. If additional space is required, reduce the `6Fh` value.
  - The compiler uses address `00h` to `0Fh` for working registers

- Address range 10h to 1Fh is the working register group reserved for first level interrupt
- If more than 1 level of interrupt nesting is required by the application, the 20h must be increased by 10h for every additional nesting level.
- Address range F0h to FFh contains the Advanced API Registers

- EData: 100h–10Fh, 110h–18Fh
  - Defined in ZDS II Project Settings under Linker Address Spaces
  - Address range 100h to 10Fh contains the Standard API Registers

- SP = 80h
  - Defined in startupePIR.asm
  - First stack location is 7Fh and it grows down

- RP = 00h
  - Defined in startupePIR.asm
  - The application code uses working register group 0

- __intrp = 10h
  - Defined in startupePIR.asm
  - First level interrupt uses working register group 1

- Engine RP = E0h
  - This is the working register group used by the PIR engine
  - Defined by the Engine Entry macro's EPIR_INIT and EPIR_ADC_ISR