Using the Parallax PIR

PN: 555-28027

Document Version: 0.2 • 04 OCT 2007
Copyright © 2007 EFX-TEK

DESCRIPTION

The Parallax PIR (passive infrared) sensor is designed to detect the presence of a human being in its field of view; this makes it a good, non-contact trigger mechanism for the Prop-1, Prop-2, or Prop-SX controllers.

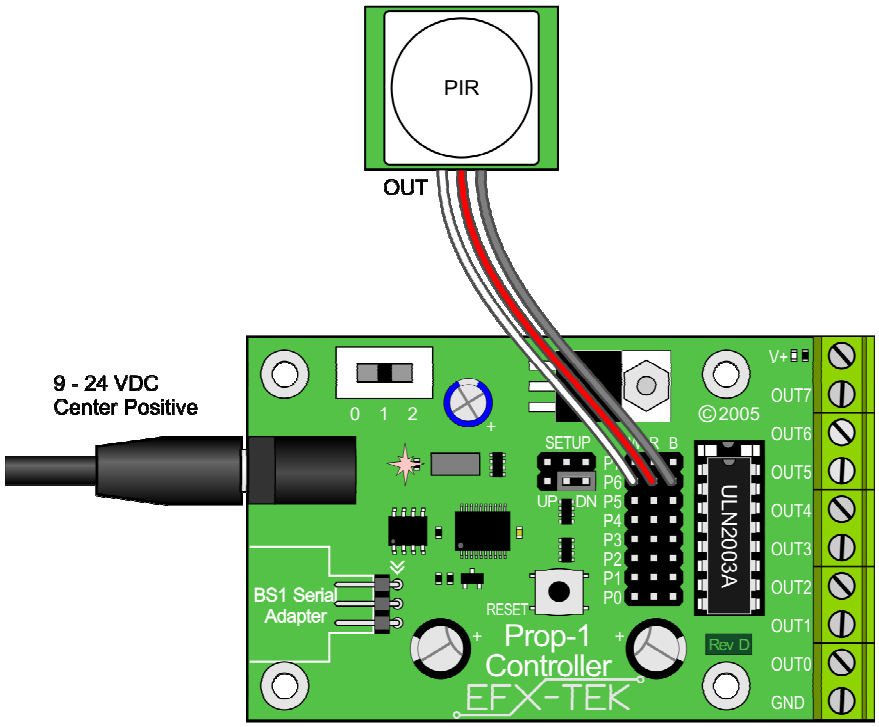
PRODUCT APPLICATIONS

- Props and holiday displays
- Home-brew alarms systems

Connections

The design of the PIR sensor allows it to be connected to any prop controller I/O pin, though for standardization with our demo code EFX-TEK recommends using P6 on the Prop-1, and P14 on the Prop-2 and Prop-SX. Connecting to an EFX-TEK prop controller is straightforward using the 12" extension cable (#805-00002). When connecting this cable to the PIR, the white wire will align with the PIR's **OUT** header pin, the red wire to **+**, the black wire to **–**.

The illustration below show a typical connection to the Prop-1 controller:



Programming

The output of the PIR is a high-going pulse, so the controller will look for a “1” on the associated input pin. In order to prevent possible false triggering, we suggest that you “debounce” the PIR by filtering the input so that pulses less than 0.25 seconds in duration are eliminated; the programs below show how to do this.

Prop-1 Example:

```
' ================================================================================
'
'   File...... PIR.BS1
'
'   {$STAMP BS1}
'   {$PBASIC 1.0}
'
' ================================================================================


' -----[ I/O Definitions ]------------------------------------------------

SYMBOL  PIR             = PIN6                  ' SETUP = DN (or out)


' -----[ Constants ]------------------------------------------------------

SYMBOL  IsOn            = 1                     ' for active-high in/out
SYMBOL  IsOff           = 0

SYMBOL  ScanDelay       = 10                    ' 10 ms scan delay


' -----[ Variables ]------------------------------------------------------

SYMBOL  pirTimer        = B2                    ' for debouncing


' -----[ Initialization ]-------------------------------------------------

Reset:
  PAUSE 20000                                   ' warm-up/inter-show delay
  pirTimer = 0                                  ' clear timer for scan


' -----[ Program Code ]---------------------------------------------------

Main:
  PAUSE ScanDelay
  pirTimer = pirTimer + ScanDelay * PIR         ' advance/clear timer
  IF pirTimer < 250 THEN Main

  ' control code here

  GOTO Reset
```

Prop-2 Example:

```
' =================================================================================
'
'   File...... PIR.BS2
'
'   {$STAMP BS2}
'   {$PBASIC 2.5}
'
' =================================================================================


' -----[ I/O Definitions ]-------------------------------------------------

PIR             PIN     14                      ' SETUP = DN (or out)


' -----[ Constants ]-------------------------------------------------------

IsOn            CON     1                       ' for active-high in/out
IsOff           CON     0

ScanDelay       CON     10                      ' 10 ms scan delay


' -----[ Variables ]-------------------------------------------------------

pirTimer        VAR     Byte                    ' for debouncing


' -----[ Initialization ]--------------------------------------------------

Reset:
  PAUSE 20000                                   ' warm-up/inter-show delay
  pirTimer = 0                                  ' clear timer for scan


' -----[ Program Code ]----------------------------------------------------

Main:
  PAUSE ScanDelay
  pirTimer = (pirTimer + ScanDelay) * PIR       ' advance/clear timer
  IF (pirTimer < 250) THEN Main

  ' control code here

  GOTO Reset
```

The key in both programs is the following line:

```
  pirTimer = pirTimer + ScanDelay * PIR         ' advance/clear timer
```

This line will advance the value of *pirTimer* by *ScanDelay* if the PIR is active (1), otherwise it will reset (clear) *pirTimer* to zero, as any value multiplied by zero is zero (the BASIC Stamp uses left-to-right evaluation of expressions). Once the PIR output stabilizes and stays active for at least 250 milliseconds, the PIR output is considered valid and the program will drop through to the main control section.

Note: For best performance the value of *ScanDelay* should be set to 5 to 10 milliseconds.

BASIC Stamp is a registered trademark of Parallax, Inc.