App. Note Code: 2RA-H
Revision: 1

APPLICATION NOTE

Eppley PIR
Precision Infrared Radiometer

CAMPBELL SCIENTIFIC, INC.®
WHEN MEASUREMENTS MATTER

Eppley PIR
Precision Infrared Radiometer

The Eppley Laboratory, Inc. Precision Infrared Radiometer (PIR) consists of a thermopile and two thermistors and provides several measurement options. This application note describes interfacing Campbell Scientific dataloggers with the PIR, explains how to measure the thermopile and thermistors, and apply the temperature compensation to the thermopile output. This note assumes previous experience with Campbell Scientific dataloggers.

Wiring

The thermopile is connected, using wires A and C, to a differential analog channel. Connect C to the high (H) side and A to the low (L) side of the differential channel.

The two thermistors are each wired to a single-ended analog channel and analog ground; thermistors are not polarity sensitive. Wires D and E are the case thermistor. Wires F and G are for the dome thermistor. The dome thermistor allows you to correct for the dome temperature. This is not necessary for many applications and is not discussed in this application note. Information about correcting for dome temperature is provided in the Eppley PIR manual.

To complete the circuit, a bridge completion resistor must be wired between an excitation channel and the single-ended measurement channel. The bridge completion resistor should be a 1 kOhm, precision- and temperature-stable resistor. The power (watt) rating of the resistor is not important. Other resistors can be used, but the datalogger must be programmed accordingly.

Programming the Datalogger

The datalogger program performs three basic tasks, typically in the following order:

- Measures the sensor (voltage and resistance measurements).
- Converts the measurements to appropriate units.
- Writes the data to final storage.

815 W. 1800 N., Logan, UT 84321-1784 (435) 753-2342

Measuring the Sensors

Thermopile

Program Instruction 2 (P02) is used to make the differential voltage measurement. The PIR sensor has a very small full-scale output range, less than 1.5 mV. Using the smallest voltage range (±2.5 mV) of the CR10X will return the best measurement. Use a differential channel from 1 through 6, ensuring it's the same channel in which the sensor is wired. The output is in millivolts.

```
1:  Volt (Diff) (P2)
  1:        1          Reps
  2:        1          2.5 mV Slow Range
  3:        1          DIFF Channel
  4:        1          Loc [ PIR_mV    ]
  5:        1.0        Mult
  6:        0.0        Offset
```

Thermistor

Each thermistor's half-bridge circuit is measured with Program Instruction 5 (P5). Select the ±2500 mV Fast Range and a single-ended channel from 1 through 12, ensuring the single-ended channel is not used by another input. Choose an excitation channel and use ±2500 mV for the excitation voltage. The same excitation channel can be used by both thermistors, but don't share bridge completion resistors.

```
2:  AC Half Bridge (P5)
  1:        1          Reps
  2:       15          2500 mV Fast Range
  3:        3          SE Channel
  4:        1          Excite all reps w/Exchan 1
  5:    2500           mV Excitation
  6:        2          Loc [ Case_Res  ]
  7:        1.0        Mult
  8:        0.0        Offset
```

Converting to Appropriate Units

The thermistor measurements must be converted to temperature in °K. The mV output from the thermopiles is converted to W m^{-2}, then corrected for the temperature effects on the PIR's case.

815 W. 1800 N., Logan, UT 84321 (435) 753-2342

Converting Thermistor Measurements to Resistance

The P5 instruction used to measure the thermistor returns a ratiometric output. To convert to resistance, use Instruction 59 (P59). Note the multiplier of 1000; this is the value of the bridge completion resistor.

```
3:  BR Transform Rf[X/(1-X)] (P59)
  1:        1          Reps
  2:        2          Loc [ Case_Res  ]
  3:    1000           Multiplier (Rf)
```

With the resistance of the thermistor in an input location, convert the resistance to temperature.

Converting Resistance to Temperature

Equation 1 converts the resistance of the thermistor to temperature in °K. This conversion requires some basic math functions that can be entered in a subroutine or in the main body of the program table.

$$T = 1/(A + B * Ln(R) + C * (Ln)^3) \qquad \text{(Eq. 1)}$$

Where:
T = temperature in degrees Kelvin
A = 0.0010295 or 1.0295E -3
B = 0.0002391 or 2.391E -4
C = 0.0000001568 or 1.568E -7
R = the measured resistance of the thermistor
Ln(R) = the natural log of the thermistor resistance

```
; Load constants A, B and C to input locations
4:  Z=F (P30)
  1:        1.0295    F
  2:       -3         Exponent of 10
  3:        6         Z Loc [ ConstA    ] ; *** A term

5:  Z=F (P30)
  1:        2.391     F
  2:       -4         Exponent of 10
  3:        7         Z Loc [ ConstB    ]

6:  Z=F (P30)
  1:        1.568     F
  2:       -7         Exponent of 10
  3:        8         Z Loc [ ConstC    ]
```


815 W. 1800 N., Logan, UT 84321-1784 (435) 753-2342

```
; Convert resistance of thermistor to natural log of resistance
7:  Z=LN(X) (P40)
  1:      2        X Loc [ Case_Res  ]
  2:      9        Z Loc [ Ln_ResS1  ]  ; *** Natural log of resistance

; Multiply constant B with Natural log of resistance
; Store result in "B term", bLn_res
8:  Z=X*Y (P36)
  1:      7        X Loc [ ConstB    ]
  2:      9        Y Loc [ Ln_Res    ]
  3:     10        Z Loc [ bLn_res   ]    ; *** B term

; Square natural log of resistance
9:  Z=X*Y (P36)
  1:      9        X Loc [ Ln_Res    ]
  2:      9        Y Loc [ Ln_Res    ]
  3:     11        Z Loc [ Ln_res2   ]    ;Natural log of resistance squared

; Cube natural log of resistance
10:  Z=X*Y (P36)
  1:      9        X Loc [ Ln_Res    ]
  2:     11        Y Loc [ Ln_res2   ]
  3:     12        Z Loc [ Ln_res3   ]    ;Natural log of resistance cubed

; Multiply constant C with natural log of resistance raised to the third power
11:  Z=X*Y (P36)
  1:      8        X Loc [ ConstC    ]
  2:     12        Y Loc [ Ln_res3   ]
  3:     12        Z Loc [ Ln_res3   ]    ; *** C term

; Add A term to B term, store in location case_temp
12:  Z=X+Y (P33)
  1:      6        X Loc [ ConstA    ]
  2:     10        Y Loc [ bLn_resS1 ]
  3:      2        Z Loc [ case_temp ]

; Add C term to A and B term, store in location Case_temp
13:  Z=X+Y (P33)
  1:      2        X Loc [ case_temp ]
  2:     12        Y Loc [ Ln_res3]
  3:      2        Z Loc [ case_temp ]
```

```
; Take the reciprocal of the "Case_temp", result is temperature in degrees Kelvin
14:  Z=1/X (P42)
  1:        2          X Loc [ case_temp ]
  2:        2          Z Loc [ case_temp ] ; *** Temp in degrees K
```

Converting Thermopile Measurements to W m^{-2}

The thermopile measurement output is in mV. To convert mV to W m^{-2}, the PIR sensitivity is divided into the mV output. Sensitivity is given in units of $\mu V\ W^{-1}\ m^2$. The proper multiplier is: 1/PIR sensitivity.

Example:
Thermopile sensitivity = 3.41 $\mu V\ W^{-1}\ m^2$ or .00341 mV / W/m^2

The thermopile sensitivity is obtained from the PIR calibration sheet.

Multiplier = (1)/(.00341 mV / W/m^2) = 293.255

Now that the multiplier has been calculated, use Instruction 37 to convert the thermopile mV output to W m^{-2}.

```
3:  Z=X*F (P37)
  1:        1          X Loc [ PIR_mV    ]
  2:     293.255       F                        ; *** Multiplier
  3:       14          Z Loc [ PIR_Aterm ]
```

Correcting for PIR Case Temperature

You must account for the effects of the PIR case temperature. The thermopile output is adjusted using Equation 2:

Corrected Thermopile Output = A + (C * T^4)   (Eq. 2)

Where:
A = Thermopile output in W m^{-2}
C = Stefan-Boltzmann constant = 5.6697E -8 W m^{-2} K^{-4}
T^4 = the case temperature in degrees K raised to the fourth power

Datalogger program instructions can be placed in a subroutine or the main body of the program.

Use the following instructions:

```
; load number 4 into input location Power4
16:  Z=F (P30)
  1:        4           F
  2:        0           Exponent of 10
  3:       11           Z Loc [ Power4    ]

; Raise temperature to the fourth power
17:  Z=X^Y (P47)
  1:       10           X Loc [ case_temp ]
  2:       11           Y Loc [ Power4    ]
  3:       10           Z Loc [ case_temp ]

; Load 5.6697E -8 into PIR_Bterm
18:  Z=F (P30)
  1:        5.669       F
  2:       -8           Exponent of 10
  3:       13           Z Loc [ PIR_Bterm ]

; Multiply 5.6697E -8 by Temp K raised to the fourth
; This is the PIR B term
19:  Z=X*Y (P36)
  1:       13           X Loc [ PIR_Bterm ]
  2:       10           Y Loc [ case_temp ]
  3:       13           Z Loc [ PIR_Bterm ]

; Add PIR A term to PIR B term. Output is in Watts per meter squared.
20:  Z=X+Y (P33)
  1:       14           X Loc [ PIR_Aterm ]
  2:       13           Y Loc [ PIR_Bterm ]
  3:       15           Z Loc [ PIR_Watt  ]
```

Writing Data to Final Storage

First determine when to write data to final storage then what data to write. The program should set the output flag before any output processing instructions are executed. Below is the portion of a datalogger program that outputs the data to final storage.

```
; Every 60 minutes at the top of the hour, set the Output Flag
6:  If time is (P92)
  1:        0           Minutes (Seconds --) into a
  2:       60           Interval (same units as above)
  3:       10           Set Output Flag High (Flag 0)
```

```
;Output the day, hour, and minute
7:  Real Time (P77)
  1:    0220         Day, Hour/Minute (midnight = 2400)

; Store the PIR hourly average
8:  Average (P71)
  1:      1          Reps
  2:     15          Loc [ PIR_Watt  ]

; Store the PIR minimum value for the past hour
9:  Minimum (P74)
  1:      1          Reps
  2:     00          Time Option
  3:     15          Loc [ PIR_Watt  ]

; Store the PIR maximum value for the past hour
10:  Maximum (P73)
  1:      1          Reps
  2:     00          Time Option
  3:     15          Loc [ PIR_Watt  ]
```

Sample CR10X Program

```
;{CR10X}
;
*Table 1 Program
  01:    10          Execution Interval (seconds)

; Measure thermopile
1:  Volt (Diff) (P2)
  1:      1          Reps
  2:      1          2.5 mV Slow Range
  3:      1          DIFF Channel
  4:      1          Loc [ PIR_mV    ]
  5:      1.0        Mult
  6:      0.0        Offset
```

```
; Measure case thermistor
2:  AC Half Bridge (P5)
  1:        1          Reps
  2:       15          2500 mV Fast Range
  3:        3          SE Channel
  4:        1          Excite all reps w/Exchan 1
  5:     2500          mV Excitation
  6:        2          Loc [ Case_Res  ]
  7:        1.0        Mult
  8:        0.0        Offset

; Apply eppley calibration to thermopile output
3:  Z=X*F (P37)
  1:        1          X Loc [ PIR_mV    ]
  2:      256.891      F
  3:       14          Z Loc [ PIR_Aterm ]

; Call Subroutine 1, used to calculate case temperature
4:  Do (P86)
  1:        1          Call Subroutine 1

; Call Subroutine 2, Used to correct thermopile output
5:  Do (P86)
  1:        2          Call Subroutine 2

; Store the data hourly
6:  If time is (P92)
  1:        0          Minutes (Seconds --) into a
  2:       60          Interval (same units as above)
  3:       10          Set Output Flag High (Flag 0)

7:  Real Time (P77)
  1:     0220          Day, Hour/Minute (midnight = 2400)

8:  Average (P71)
  1:        1          Reps
  2:       15          Loc [ PIR_Watt  ]

9:  Minimum (P74)
  1:        1          Reps
  2:       00          Time Option
  3:       15          Loc [ PIR_Watt  ]
```

```
10:  Maximum (P73)
  1:      1          Reps
  2:     00          Time Option
  3:     15          Loc [ PIR_Watt  ]

*Table 2 Program
  02:       0.0000   Execution Interval (seconds)

*Table 3 Subroutines

1:  Beginning of Subroutine (P85)
  1:      1          Subroutine 1
```

; Equation to convert YSI 10 K thermistor resistance to temperature
; in degrees K
; 1/T = a + b(LnR) + c(LnR)^3

; Coefficients used in equation above
; a = 0.0010295 = ConstA
; b = 0.0002391 = ConstB
; c = 1.568E-7 = ConstC

; Calculate resistance from P5 instruction (Table 1, line 2)

```
2:  BR Transform Rf[X/(1-X)] (P59)
  1:      1          Reps
  2:      2          Loc [ Case_Res  ]
  3:   1000          Multiplier (Rf)
```

; Load constants a, b and c to input locations

```
3:  Z=F (P30)
  1:      1.0295     F
  2:     -3          Exponent of 10
  3:      3           Z Loc [ ConstA    ]    ; *** A term

4:  Z=F (P30)
  1:      2.391      F
  2:     -4          Exponent of 10
  3:      4          Z Loc [ ConstB    ]

5:  Z=F (P30)
  1:      1.568      F
  2:     -7          Exponent of 10
  3:      5          Z Loc [ ConstC    ]
```

```
; Convert resistance of thermistor to natural log of resistance
6:  Z=LN(X) (P40)
  1:        2           X Loc [ Case_Res  ]
  2:        6           Z Loc [ Ln_Res    ]    ; *** Natural log of resistance

; Multiply constant B with Natural log of resistance
; Store result in "B term, bLn_res”
7:  Z=X*Y (P36)
  1:        4           X Loc [ ConstB    ]
  2:        6           Y Loc [ Ln_Res    ]
  3:        7           Z Loc [ bLn_res   ]    ;  *** B term

; Square natural log of resistance
8:  Z=X*Y (P36)
  1:        6           X Loc [ Ln_Res    ]
  2:        6           Y Loc [ Ln_Res    ]
  3:        8           Z Loc [ Ln_res2   ]    ;Natural log of resistance squared

; Cube natural log of resistance
9:  Z=X*Y (P36)
  1:        6           X Loc [ Ln_Res    ]
  2:        8           Y Loc [ Ln_res2   ]
  3:        9           Z Loc [ Ln_res3   ]    ;Natural log of resistance cubed

; Multiply constant C with natural log of resistance raised to the third power
10:  Z=X*Y (P36)
  1:        5           X Loc [ ConstC    ]
  2:        9           Y Loc [ Ln_res3   ]
  3:        9           Z Loc [ Ln_res3   ]    ; *** C term

; Add A term to B term natural log of resistance, store in location
; case_temp
11:  Z=X+Y (P33)
  1:        3           X Loc [ ConstA    ]
  2:        7           Y Loc [ bLn_res   ]
  3:       10           Z Loc [ case_temp ]

; Add C term to A and B term, store in location case_temp
12:  Z=X+Y (P33)
  1:       10           X Loc [ case_temp ]
  2:        9           Y Loc [ Ln_res3   ]
  3:       10           Z Loc [ case_temp ]
```

```
; Invert A plus B plus C term.  Result is temperature in degrees
; Kelvin
13:  Z=1/X (P42)
  1:      10          X Loc [ case_temp ]
  2:      10          Z Loc [ case_temp ]   ; *** Temp in degrees K

14:  End (P95)

15:  Beginning of Subroutine (P85)
  1:       2          Subroutine 2

; Apply case thermistor correction to PIR thermopile output

; Correction is PIR output in watts per meter squared plus
; 5.669E -8 multiplied by the case temperature raised to the fourth power.

; Load number 4 into input location Power4
16:  Z=F (P30)
  1:       4          F
  2:       0          Exponent of 10
  3:      11          Z Loc [ Power4    ]

; Raise temperature to the fourth power
17:  Z=X^Y (P47)
  1:      10          X Loc [ case_temp ]
  2:      11          Y Loc [ Power4    ]
  3:      10          Z Loc [ case_temp ]

; Load 5.669E -8 into PIR_Bterm
18:  Z=F (P30)
  1:       5.669      F
  2:      -8          Exponent of 10
  3:      13          Z Loc [ PIR_Bterm ]

; Multiply 5.669E -8 by Temp K raised to the fourth
; This is the PIR B term
19:  Z=X*Y (P36)
  1:      13          X Loc [ PIR_Bterm ]
  2:      10          Y Loc [ case_temp ]
  3:      13          Z Loc [ PIR_Bterm ]
```

```
; Add PIR A term to PIR B term. Output is in Watts per meter squared.
20:  Z=X+Y (P33)
  1:      14          X Loc [ PIR_Aterm ]
  2:      13          Y Loc [ PIR_Bterm ]
  3:      15          Z Loc [ PIR_Watt  ]

21:  End (P95)

End Program
```


815 W. 1800 N., Logan, UT 84321 (435) 753-2342