AN3641

Low-Power, Cost-Efficient PIR Motion Detection Using the tinyAVR® 2 Family

Features

- 12-bit ADC, Differential Measurement
- ADC Sample Accumulation Up To 1024 Samples
- Programmable Gain Amplifier (PGA) Up To 16x Gain
- Signal Amplifying, Oversampling And Analyzing Using The Built-in ADC With PGA
- Hardware Averaging Using Burst Mode
- ADC Triggered Through The Event System Using The Periodic Interrupt Timer (PIT)
- Reduced Bill Of Material (BoM) Cost Compared To Typical Passive InfraRed (PIR) Solutions
- Visualize Data Using The Microchip Data Visualizer And Data Streamer Protocol
- Visualize Power Consumption Using The Microchip Data Visualizer And a Power Debugger

Introduction

Author: Kjetil Kirkholt, Microchip Technology Inc.

This application note describes how to use the 12-bit differential ADC with PGA in the tinyAVR® 2 Family of microcontrollers and how to collect measurements using a Passive InfraRed (PIR) sensor while keeping the current consumption at a minimum.

In this application example, the signal from the PIR sensor is amplified, oversampled, digitally filtered, and analyzed to determine if movement has occurred. LED0 on the ATtiny1627 Curiosity Nano (DM080104) board is used to indicate when movement is detected. The measured data can be sent to the serial terminal and illustrated in the *Data Visualizer* application or plugin for Atmel Studio.

The output signal levels from a PIR sensor are typically very low and less than 1 mV. To detect the movement and avoid false detections, the signal needs to be amplified before being sampled by the ADC. In typical PIR solutions, this is achieved by using several Operational Amplifier (op amp) stages with high gain. In this application example, no external op amps are used, and the amplification of the signal is purely done inside the microcontroller (MCU) by using the internal PGA. The ADC supports sampling in bursts, where a configurable number of conversion results are accumulated automatically into a single ADC result (sample accumulation). The application example accumulates 16 samples, which increases the resolution of the 12-bit ADC by two bits. It is possible to oversample the signal up to 1024 times and gain five additional bits, resulting in a 17-bit ADC resolution, if needed.

More information on oversampling can be found in the ADC Oversampling with tinyAVR 0- and 1-series, and megaAVR 0-series product brief.

For demonstration purposes, the ATtiny1627 Curiosity Nano board, Curiosity Nano Base for Click boards™, and a modified MIKROE-3339 PIR Click boards™ are used.

The example code for replicating the results described in this application note is available from Atmel START:

- PIR sensor with tinyAVR 2-series ADC
  - start.atmel.com/#example/Atmel%3AApplication_AVR_Examples%3A1.0.0%3A%3AApplication%3APIR_Motion_Detection%3A

The bare metal code example (without using Atmel START drivers) is available on GitHub:

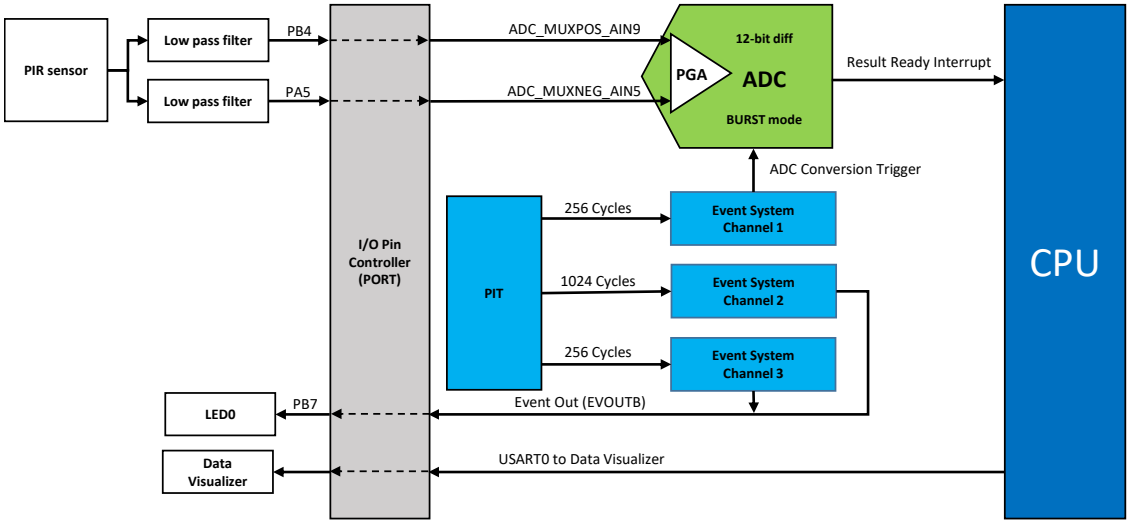
More information about the application is described in Sections Demo Operation and Code Implementation.

Additional details on the ADC performance and general configuration are available in the device data sheet.

Table of Contents

1. Block Diagram

Figure 1-1. Block Diagram of the System

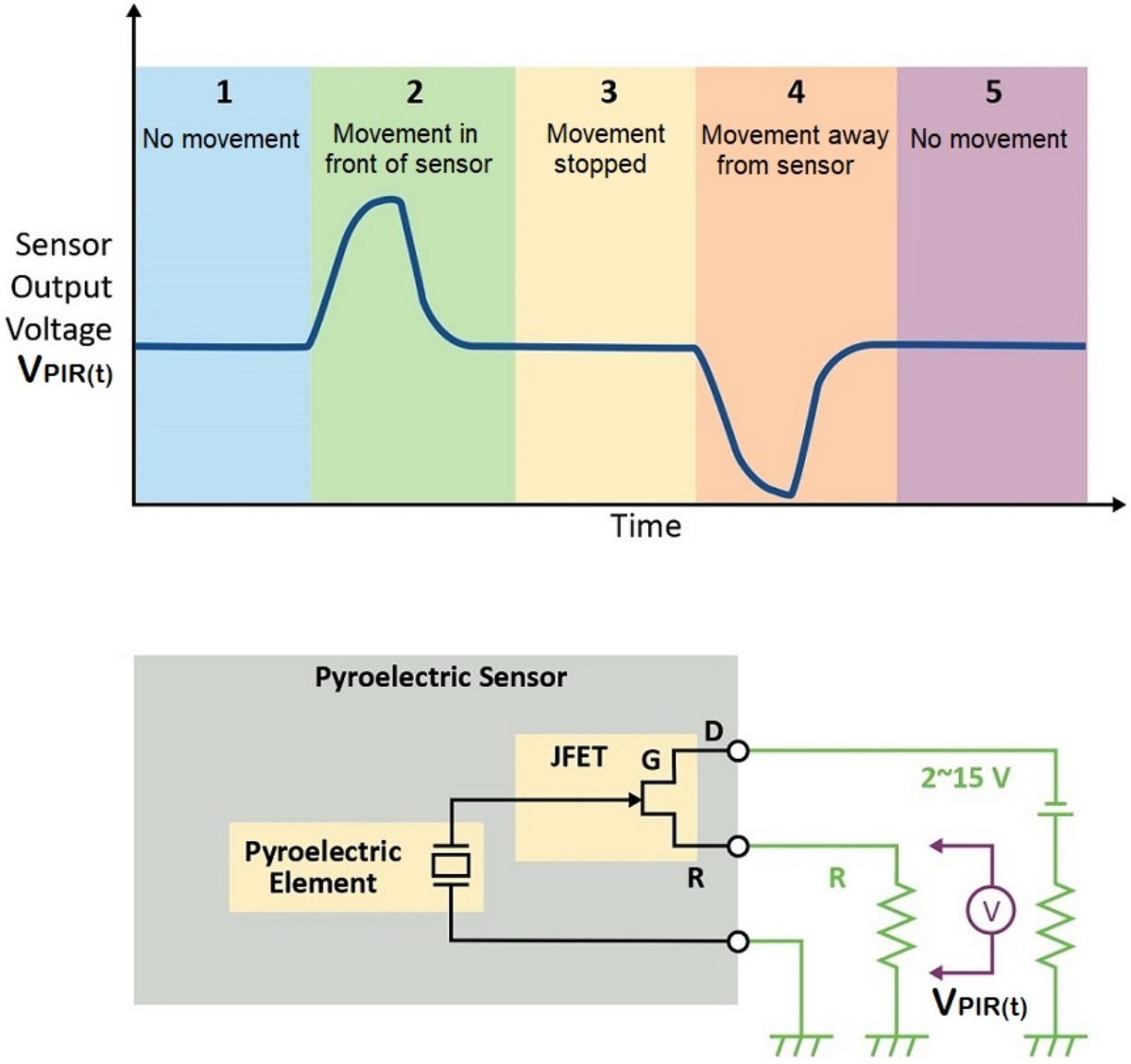


2. PIR Sensor Theory

A PIR sensor detects changes in the amount of infrared radiation “seen” by the sensor elements, which varies depending on the temperature and surface characteristics of the object in front of the sensor.

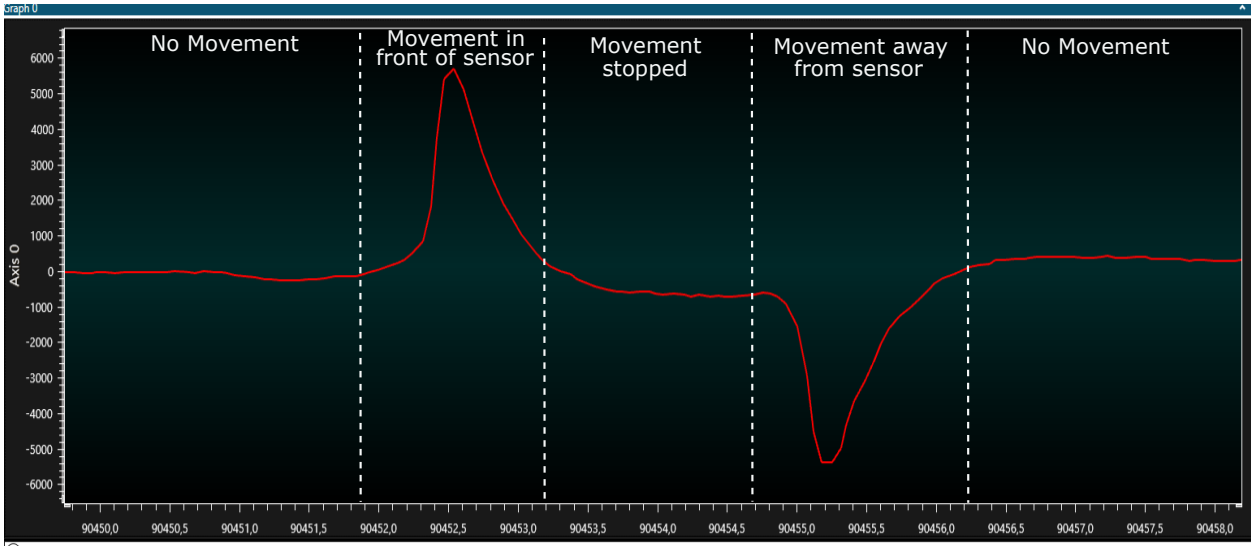
When a person passes between the sensor and the background, the sensor detects the change from ambient temperature to body temperature, and back again. The sensor converts the resulting change in the incoming infrared radiation into a change in the output voltage (VPIR(t)), as shown in the PIR sensor motion detection principle. Other objects with the same temperature as the background, but with different surface characteristics, will also cause the sensor to detect a different emission pattern.

Figure 2-1. PIR Sensor Motion Detection Principle



The figure below shows a plot of the raw data captured from the sensor using Data Visualizer. A hand is placed over the sensor and held there for a short time and then removed.

Figure 2-2. PIR Sensor Raw Data Motion Detection Using Data Visualizer



3. Demo Operation

When using a PIR sensor in an application, it is necessary to allow the sensor some time to warm up. During warm-up the PIR sensor will reach equilibrium to the IR radiation in its current field of view and thus “learn” its environment to stabilize the output signal. During this time, it is important to keep the environment in the sensor's field of view undisturbed. The warm-up time can be adjusted in the application firmware by changing the define `PIR_WARMUP_TIME_MS`. While in warm-up, LED0 will flash at 1 Hz. When LED0 stops flashing, the warm-up is complete, and the system is now operating normally.

Waving a hand or walking in front of the sensor will now be detected and cause LED0 to flash at 4 Hz and will continue flashing as long as there is movement in the senor's field of view. When the movement stops, LED0 turns off.

3.1 Configuring, Customizing, Tuning the PIR Application Sampling, Filtering and Detection Parameters

The demo application has been created in a way that makes it easy to modify the sampling acquisition parameters, the filtering parameters, and detection threshold parameters. These parameters are all conveniently located in the main.c file of the project. Tuning these parameters will allow you to customize your design to optimize the signal integrity, range, and power consumption for your design.

3.1.1 Sample Acquisition Parameters

- PIR_OVERSAMPLE_RATE
  - This parameter sets the numbers of times the signal is oversampled. Increasing this parameter will result in an increased resolution, thus increasing the detection range, but it will also increase power consumption. Decreasing this parameter will have the opposite effect on the detection range and power consumption.
- PIR_SAMPLE_RATE_PER_SECOND
  - This parameter sets the number of samples done each second. Increasing this parameter will make the application respond faster to movement, increase the detection range, but it will also increase power consumption. Decreasing this parameter will have the opposite effect on responsiveness, range, and power consumption.
- PIR_PGA_GAIN
  - This parameter sets the PGA gain. Increasing or decreasing this parameter will affect the PGA gain resulting in a higher or lower signal to the ADC, which will affect the effective detection range that the system. Changing the gain will not affect power consumption.

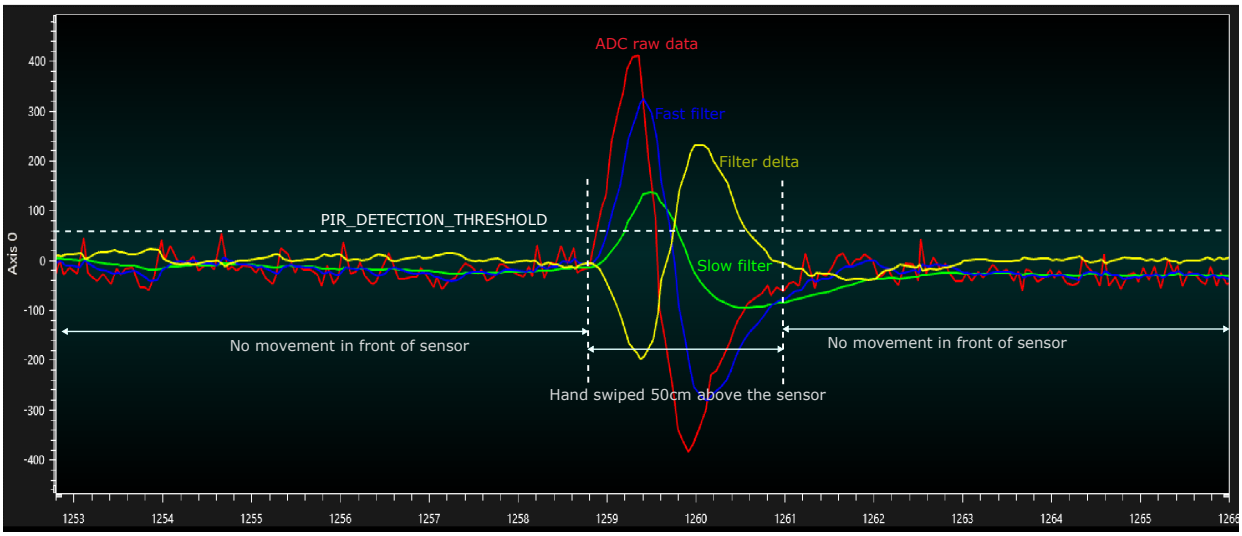
3.1.2 Filtering and Detection Algorithm Parameters

- PIR_WARMUP_TIME
  - This parameter sets the PIR warm-up time. This parameter should be adjusted to match the warm-up time needed for the selected PIR sensor in the application.
- PIR_DETECTION_THRESHOLD
  - This parameter sets the detection threshold of the application. Decreasing this parameter can increase the effective range that the system can detect movement on, but at the same time increase the chance of false detections due to sudden changes in ambient temperatures or other disturbances. Increasing this parameter decreases the range, but the likelihood of false detections will be lower. Setting this parameter too high can result in no detections.
- PIR_LONG_TERM_FILTER_RANGE
  - This parameter sets the length of the long term filter. Changes to this parameter need to be done carefully while also considering the PIR_SHORT_TERM_FILTER_RANGE and PIR_DETECTION_THRESHOLD parameters. Changes can affect the effective range of the system and increase power consumption.
- PIR_SHORT_TERM_FILTER_RANGE
  - This parameter sets the length of the short term filter. Changes to this parameter need to be done carefully while also considering the PIR_LONG_TERM_FILTER_RANGE and PIR_DETECTION_THRESHOLD parameters. Changes can affect the effective range of the system and increase power consumption.

3.1.3 Debug Interface Parameters

- PIR_DEBUG_MESSAGES
  - This parameter, if defined, will unlock the possibility to send debug data over UART to Data Visualizer. Adding this parameter will increase the power consumption of the system and should be removed if low power consumption is important.

Figure 3-1. Data Visualizer with Movement



3.2 Hardware Prerequisite

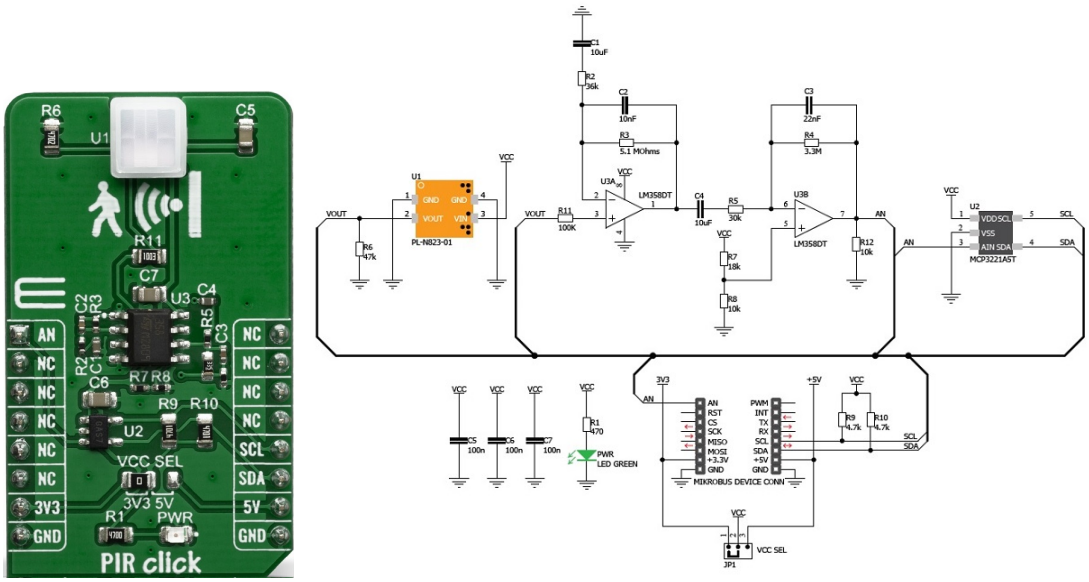
- ATtiny1627 Curiosity Nano (DM080104): www.microchip.com/developmenttools/ProductDetails/DM080104
- Curiosity Nano Base for Click boards™ (AC164162): www.microchip.com/DevelopmentTools/ProductDetails/AC164162
- MIKROE-3339 Click board: www.mikroe.com/pir-click

3.2.1 Click board™ Modifications

Before the Click board can be used in this demo, it needs to be modified by removing the op amp gain stage and some other components.

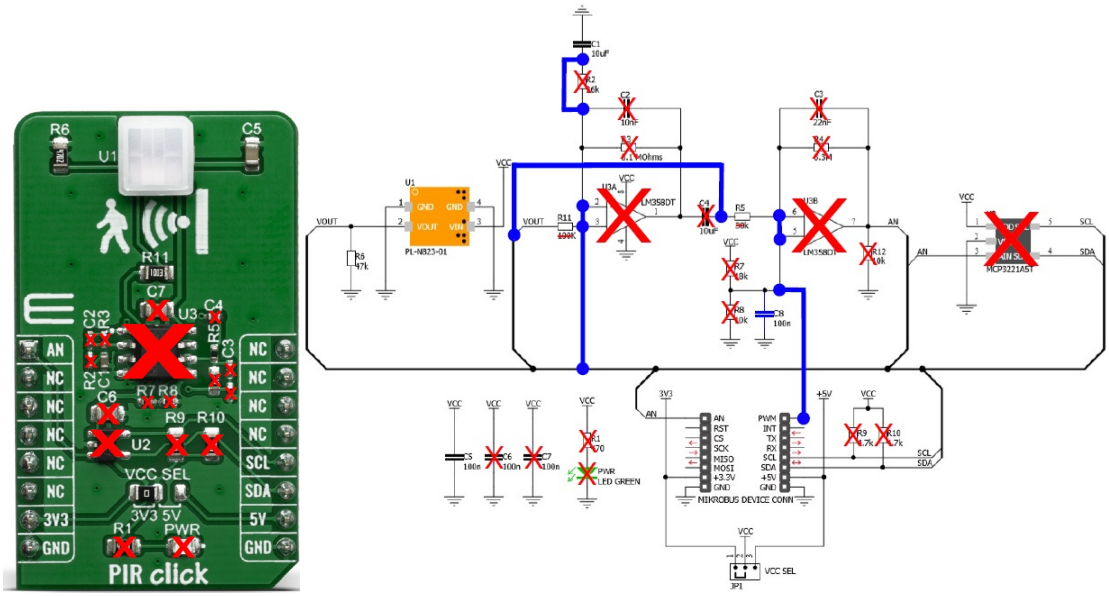
The PIR Click board is used for demonstration purposes. The original Click board has an op amp solution for amplifying the signal, which is shown in the figure below. This needs to be removed to demonstrate that the tinyAVR® 2 family microcontroller can amplify the raw signal from the PIR sensor using the integrated PGA and differential ADC.

Figure 3-2. Original Click board™ and Schematic



Unnecessary components are removed from the Click board showed by the red Xs in the figure below. New connections and components are in blue. U2, R9, R10, R1, and PWR LED are not part of the original signal chain but are removed to minimize the power consumption.

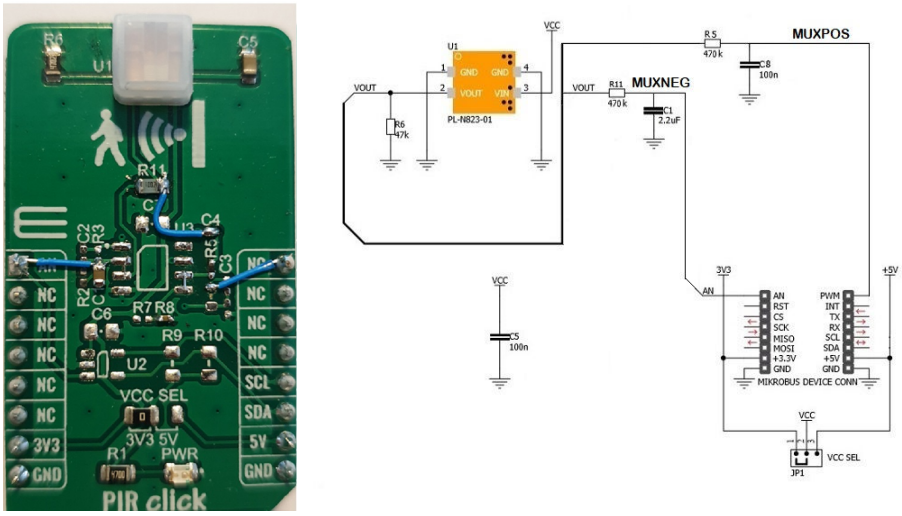
Figure 3-3. Planned Modifications to the Click board™ and Schematics



Resistor R5 is changed from 30 kΩ to 470 kΩ. R5 and C8 (100 nF) form a low-pass filter with a cutoff frequency of 3.38 Hz. This filter will transfer the full AC signal from the PIR sensor in addition to the DC bias and filters out high-frequency noise.

Resistor R11 is changed from 100 kΩ to 470 kΩ, and C1 is changed from 10 µF to 2.2 µF. When R11 is combined with capacitor C1, it forms a low-pass filter with a cutoff frequency of 0.154 Hz. This filter will block most of the AC signal from the PIR and only transfer the sensors DC bias to the negative input of the ADC.

Figure 3-4. Modified Click board™ and Schematics



The cut-off frequency for both filters should be selected based on the application needs and what type of movement the system is designed to detect.

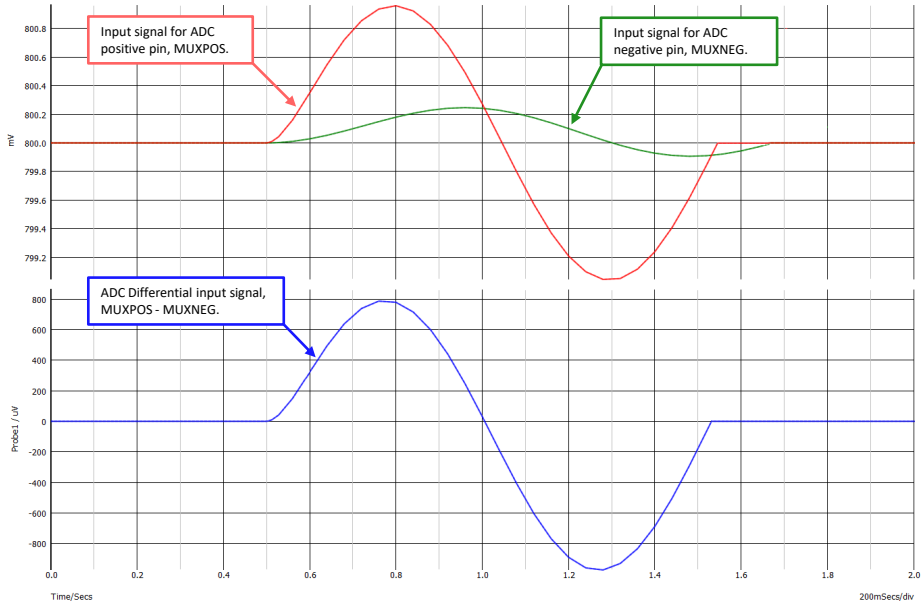
3.2.1.1 Filter Simulation Using MPLAB® Mindi™ Analog Simulator

When connecting the PIR sensor, as shown in the figure above, the DC bias will be applied to both the positive and the negative input of the ADC. Since the ADC is configured in differential mode, the DC bias is eliminated, and the full ADC range will be available for the AC signal.

By using the MPLAB® Mindi™ Analog Simulator, it is possible to simulate the behavior of the two low-pass filters when a simulated PIR signal is applied. The simulation also shows how the differential signal to the ADC will look like after the DC bias voltage is removed by the ADC differential inputs.

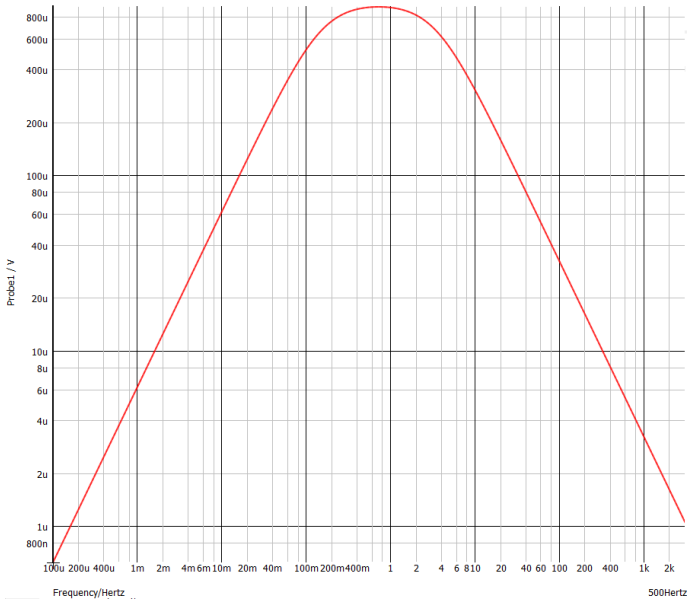
The PIR signal is simulated using a 1 Hz sine wave with 1 mV amplitude placed at an 800 mV bias voltage.

Figure 3-5. MPLAB® Mindi™ Filter Simulation



The two low-pass filters, in combination with the ADC differential input, effectively form a bandpass filter. The figure below shows an MPLAB® Mindi™ simulation of the frequency response of the two filters in combination with the differential ADC.

Figure 3-6. MPLAB® Mindi™ Filter Response



More information about MPLAB® Mindi™ Analog Simulator, how to download and to use it can be found here: www.microchip.com/mplab/mplab-mindi.

3.3 Hardware Setup

This section provides information on the hardware setup and pin configuration, as shown below

Figure 3-7. Visual Representation of HW Setup

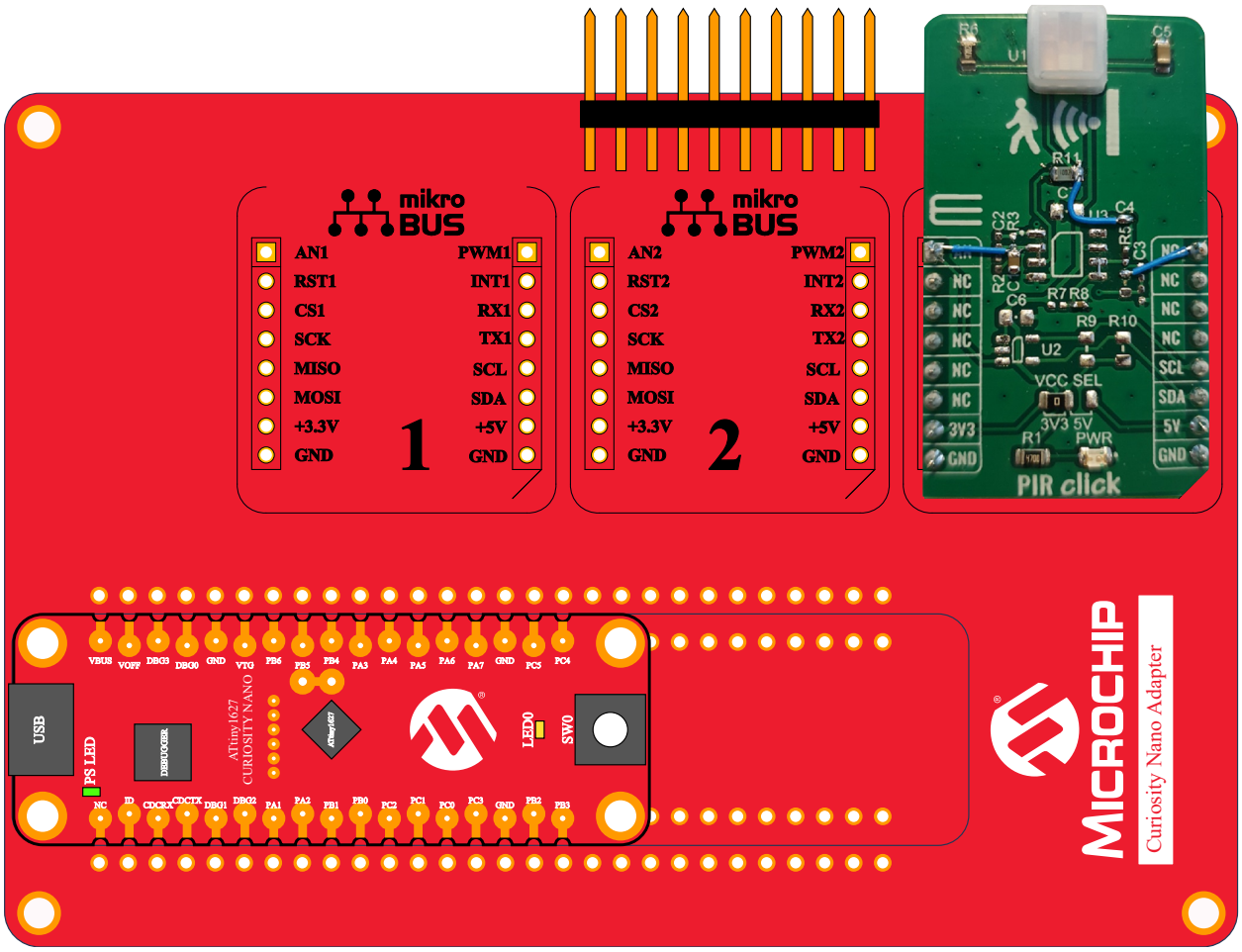


Table 3-1. Pin Configuration and Click board™

Curiosity Nano Adapter Slot	Click board™	Curiosity Nano Adapter Pin Name	MCU Pin Name
Slot 3	Modified PIR Click	AN3	PA5
		PWM3	PB4

3.4 Software Prerequisite

- Atmel Studio 7 (Version 7.0.1931 or later)
- Atmel Studio ATtiny_DFP version 1.4.310 or later
- Data Visualizer, standalone or Extension in Atmel Studio

3.5 Running the Example

1. Connect the modified Click board to slot 3 of the Curiosity Nano Base for Click boards™.
2. Connect the ATtiny1627 Curiosity Nano to the Curiosity Nano Base for Click boards™.
3. Connect the ATtiny1627 Curiosity Nano to a computer using a USB cable.

4. Download the application, as explained in Section 8. Get Code Examples from Atmel START or Section 9. Get Code Examples from GitHub.
5. Choose one of the two code projects and program the ATtiny1627.
6. Wave a hand or walk in front of the sensor and observe the red LED flashing.

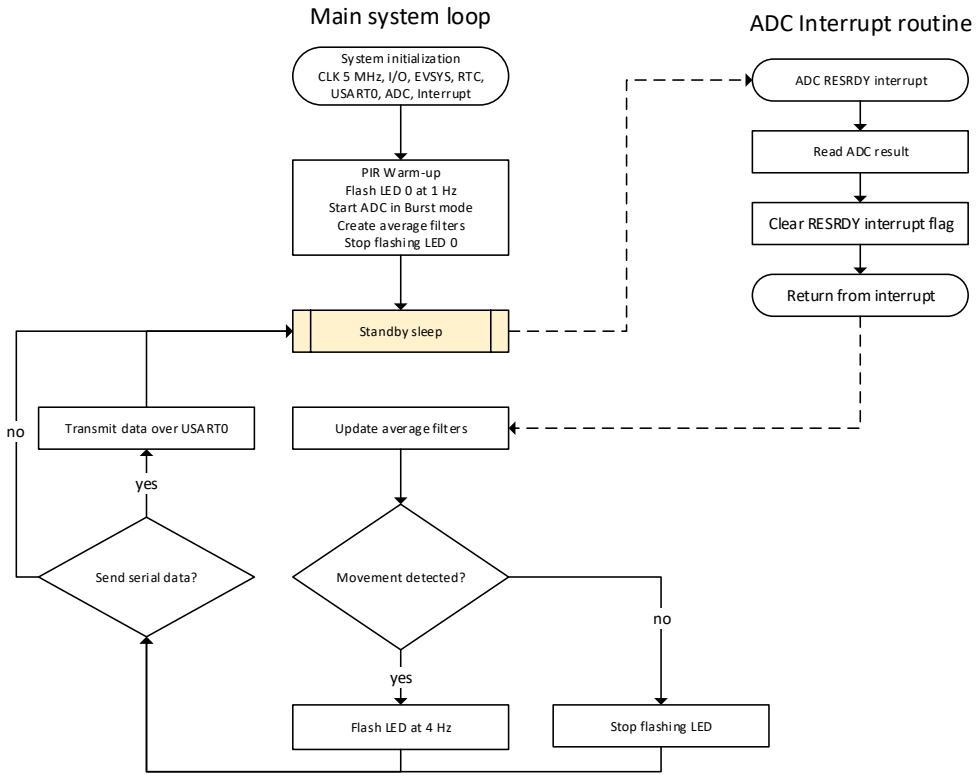
4. Source Code Overview

4.1 MCU and Peripheral Configuration

- CPU clock: 10 MHz in Active mode, 5 MHz in Standby sleep mode
- Peripherals used:
  - ADC in differential mode
    - PGA set to 16x gain
    - ADC MUXPOS input channel is AIN9: pin PB4
    - ADC MUXNEG input channel is AIN5: pin PA5
    - ADC reference voltage: 1.024V
    - ADC clock: 2.5 MHz (F_{CPU}/2)
    - ADC triggered trough the Event System
  - RTC/PIT:
    - RTC clock running at 1024 Hz
    - PIT/256 connected to ADC trough the event system. The conversion trigger rate is 4 Hz.
    - PIT/256 connected to the event out pin PB7. LED0 toggle rate is 4 Hz when movement is detected.
    - PIT/1024 connected to the event out pin PB7. LED0 toggle rate is 1 Hz during warm-up.
  - Event System
    - Connects PIT to ADC
    - Connects PIT to LED0
  - USART0:
    - TXD: PB2
    - RXD: PB3
    - Baud rate: 115200, ADC result is sent to the serial terminal

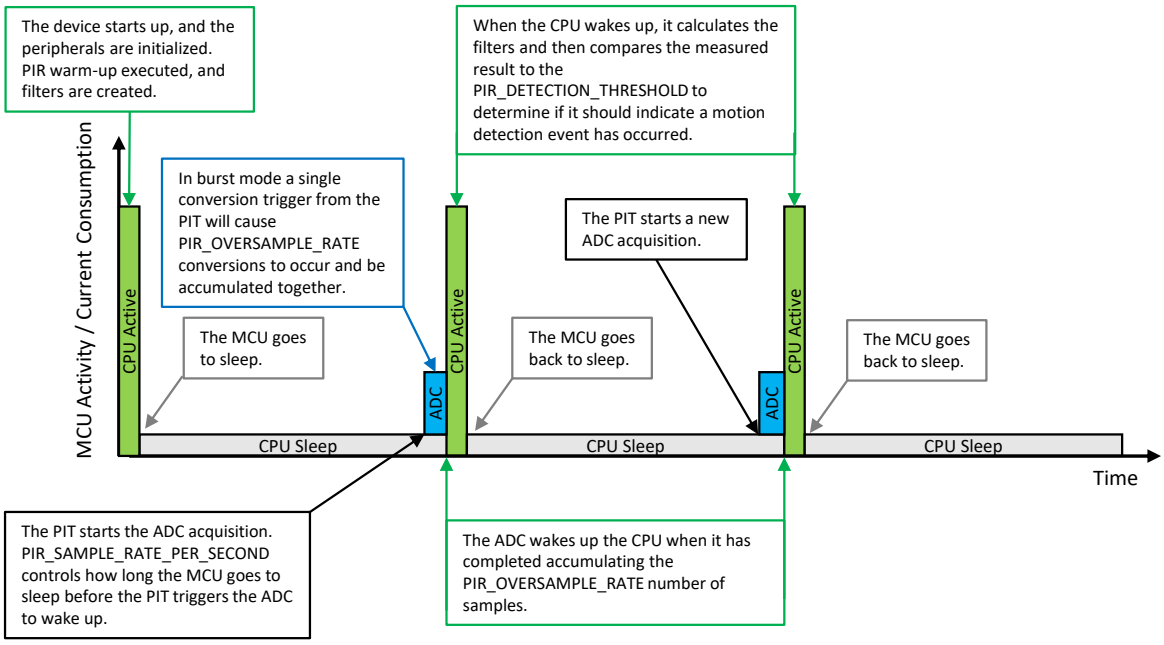
4.2 Flow Chart

Figure 4-1. Firmware Flowchart



4.3 Timing Diagram

Figure 4-2. Firmware Timing Diagram



4.4 Code Implementation

The code consists of these main parts:

- Peripheral initialization
- Main loop
- Warm-up and filter creation
- Updating average filters
- Checking for movement
- Handling the ADC interrupt routine

4.4.1 Peripheral Initialization

The initialization of the peripherals in ATtiny1627 can be done using START drivers. As this application note is focusing on the ADC, only the bare metal ADC initialization is detailed here.

```
void ADC_0_init()
{
    ADC0.CTRLB = ADC_PRESC_DIV2_gc;              /* System clock divided by 2 */
    ADC0.CTRLF = PIR_OVERSAMPLE_RATE;            /* Samples accumulation */
    ADC0.CTRLC = ADC_REFSEL_1024MV_gc            /* Internal 1.024V Reference */
            | (TIMEBASE_VALUE << ADC_TIMEBASE_gp);  /* timebase value */

    ADC0.CTRLE = 0x12;                       /* Sample Duration */

    ADC0.DBGCTRL = ADC_DBGRUN_bm;               /* Debug run: enabled */
    ADC0.COMMAND = ADC_DIFF_bm                  /* Differential ADC Conversion enabled */
              | ADC_MODE_BURST_gc;              /* Burst Accumulation */

    ADC0.INTCTRL = ADC_RESRDY_bm;               /* Result Ready Interrupt enabled */
```

```
    ADC0.MUXNEG = ADC_VIA_PGA_gc                /* Via PGA */
                | ADC_MUXNEG_AIN5_gc;           /* ADC input pin 5, PIR sensor */

    ADC0.MUXPOS = ADC_VIA_PGA_gc                /* Via PGA */
                | ADC_MUXPOS_AIN9_gc;           /* ADC input pin 9, PIR sensor */

    ADC0.PGACTRL = ADC_PGAEN_bm                 /* PGA Enabled */
                 | PIR_PGA_GAIN                /* Gain */
                 | ADC_ADCPGASAMPDUR_15CLK_gc   /* PGA Sample Duration */
                 | ADC_PGABIASSEL_1_2X_gc;      /* PGA bias set to 1/2 */

    ADC0.CTRLA = ADC_ENABLE_bm                  /* Enable ADC */
               | ADC_RUNSTDBY_bm;               /* Enable ADC to run in Standby Sleep mode */

    ADC0.INTFLAGS = ADC_RESRDY_bm;              /* Clear ADC Result Ready Interrupt Flag */
}
```

In the example application, the ADC is initialized in `ADC_0_init()`.

The PGA has been enabled to amplify the signal coming from the PIR sensor, and the gain can be adjusted by changing the define `PIR_PGA_GAIN` found in `main.c`. The ADC is running in *Differential mode* and *Burst mode* to accumulate samples and amplify the signal further. The number of accumulated samples can be adjusted in `PIR_OVERSAMPLE_RATE`.

In *Differential mode*, the voltage difference between the two inputs is measured by the ADC.

In *Burst mode*, a burst of *n* conversions is accumulated as fast as possible after a single trigger, and the conversion results are accumulated into a single ADC result.

4.4.2 Main Code

In the main function, after initialization, the MCU will enter a while(1) loop where it will go into Standby sleep mode. The MCU is only woken when the ADC conversion is complete. The filters are then updated, and movement determination is made. To reduce the average power consumption, the peripheral clock speed is reduced before entering Standby sleep mode and increased again after waking up, to speed up updating the filters, and determine if any movement has occurred. This change in clock speed is only effective if the number of accumulated ADC samples configured in `PIR_OVERSAMPLE_RATE` is greater than eight. Otherwise, the time to change the clock speed is longer than the time spent doing the ADC conversions, thus increasing the power consumption.

```
int main(void)
{
    /* Initial clock set to
5MHz.
   */
    /* If number of samples accumulated by the ADC is less than 16, CLKCTRL_PDIV_2X_gc in
MCLKCTRLB and ADC0.CTRLB = ADC_PRESC_DIV4_gc;    */
    /*  should be used to lower average power
consumption
*/
    ccp_write_io((void *)&(CLKCTRL.MCLKCTRLB), CLKCTRL_PDIV_4X_gc | 1 << CLKCTRL_PEN_bp ); /*
Divide main clock by 4x*/

    IO_init();                  /* Initialize IO pins */
    EVENT_SYSTEM_0_init();          /* Initialize Event System */
    RTC_init();                 /* Initialize PIT */
    #ifdef PIR_DEBUG_MESSAGES
    USART_init();               /* Initialize USART if SEND_SERIAL_DATA is defined */
    #endif

    ADC_0_init();               /* Initialize the ADC */

    SLPCTRL.CTRLA = SLPCTRL_SMODE_STDBY_gc; /* Select standby sleep mode */

    sei();                      /* Enable global interrupt */

    warm_up_and_filter_creation();      /* Run warm-up and collect ADC data to create
filters */

    while (1)
    {
```

```
        /* Reduce main clock speed to lower peripheral clock tree power consumption when ADC
is running in sleep mode */
        /* Only effective if ADC is configured to accumulate more that 8
samples.                                                */
        /* If number of samples accumulated is less than 16, the line below should be
removed                                 */
        ccp_write_io((void *)&(CLKCTRL.MCLKCTRLB), CLKCTRL_PDIV_4X_gc | 1 <<
CLKCTRL_PEN_bp ); /* Divide main clock by 4x */

        sleep_mode();                /* Enter sleep mode */

        /* Increase main clock to run the code faster in active mode, this will reduce the
average power consumption */
        /* If number of samples accumulated is less than 16, the line below should be
removed                               */
        ccp_write_io((void *)&(CLKCTRL.MCLKCTRLB), CLKCTRL_PDIV_2X_gc | 1 <<
CLKCTRL_PEN_bp ); /* Divide main clock by 2x */

        update_average_filters();    /* Update filters with new measurement */
        check_for_movement();        /* Check if movement has happened by examining delta
between filters */

        #ifdef PIR_DEBUG_MESSAGES    /* Send data to data visualizer if SEND_SERIAL_DATA is
defined */
        usart_tx(adc_result, long_term_average, short_term_average, filter_delta);
        #endif
    }
}
```

4.4.3 Warm-up and Filter Creation

This section of the code is run before the main loop as part of the initialization of the system. The warm-up time is just a delay loop to give the PIR sensor time to adjust to the IR radiation and can be adjusted in the define `PIR_WARMUP_TIME_MS`. The PIT is connected through the Event System to LED 0 on the ATtiny1627 Nano and will flash at 1 Hz through the entire warm-up and filter creation part of the code to indicate to the user that the system is warming up.

When the warm-up time is over, the ADC is started using the following command:

```
ADC0.COMMAND |= ADC_START_EVENT_TRIGGER_gc;
```

and the MCU is put in Standby sleep mode.

The PIT is configured to start the ADC conversion through the Event System. When the PIT sends the event to start an ADC conversion, the ADC is started and performs the configured conversion and issues a result ready (REDRDY) interrupt on completion that wakes the MCU. How often the PIT sends an event can be configured in the define `PIR_SAMPLE_RATE_PER_SECOND`.

```
void warm_up_and_filter_creation()
{
    uint8_t i = 0;

    EVSYS.USEREVSYSEVOUTB = EVSYS_USER_CHANNEL2_gc; /* Flash LED0 at 1 Hz to indicate warm-up
*/

    /* Put warm-up delay/code here */
    _delay_ms(PIR_WARMUP_TIME_MS);
    /* End of warm-up delay/code */

    ADC0.COMMAND |= ADC_START_EVENT_TRIGGER_gc;        /* Enable ADC to be triggered by
Event*/

    while(i < PIR_LONG_TERM_FILTER_RANGE)
    {
        sleep_mode();                                  /* Enter Standby sleep mode and wait for
ADC to complete*/
        accumulated_long_term_average = accumulated_long_term_average + adc_result;

        if (i < PIR_SHORT_TERM_FILTER_RANGE)
        {
            accumulated_short_term_average = accumulated_short_term_average + adc_result;
```

```
        }
        i++;
    }

    EVSYS.USEREVSYSEVOUTB = 0; /* Warm up and average filter creation complete, stop flashing
LED0 */
}
```

4.4.4 Update Average Filters

These filters are used to average out the noise in the PIR sensor measurement. This will reduce the chance of false detections when the noise is high. The `short_term_average` filter is the average of the four most recent ADC measurements and can be adjusted in the define `PIR_SHORT_TERM_FILTER_RANGE`. For each new ADC measurement, 1/(size of filter) is subtracted, and the new ADC measurement is added. This way, the filter is constantly updated and will be able to track rapid changes in the ADC measurements, such as movement. The `long_term_average` filter swap out 1/(size of filter) for each new ADC measurement. The `long_term_average` filter is slower but will track and average the noise in the measurement in addition to the slower drift in the system. When there is no movement, the filters will converge around the same value, and the `filter_delta` will stay low. When movement happens, the signal from the PIR will change, and the `filter_delta` will increase since the `short_term_average` will follow the changing signal faster than the `long_term_average`.

```
void update_average_filters()
{
    accumulated_long_term_average
    -= accumulated_long_term_average / PIR_LONG_TERM_FILTER_RANGE; /* Subtract 1/x from
accumulated filter value */
    accumulated_short_term_average
    -= accumulated_short_term_average / PIR_SHORT_TERM_FILTER_RANGE; /* Subtract 1/y from
accumulated filter value */

    accumulated_long_term_average += adc_result;  /* Add new ADC measurement (1/x) to
accumulated filter value */
    accumulated_short_term_average += adc_result; /* Add new ADC measurement (1/y) to
accumulated filter value */

    long_term_average = accumulated_long_term_average / PIR_LONG_TERM_FILTER_RANGE; /* Divide
the accumulated_long_term_average on X to create long_term_average */
    short_term_average = accumulated_short_term_average    / PIR_SHORT_TERM_FILTER_RANGE; /*
Divide the accumulated_long_term_average on Y to create long_term_average */

    filter_delta = long_term_average - short_term_average; /* Find delta between filters */
}
```

4.4.5 Check for Movement

This part of the code will compare the absolute value of `filter_delta` to the `PIR_DETECTION_THRESHOLD` to determine if any movement has occurred or not. If movement has been detected, the PIT will be connected to LED0 and flash at 4 Hz as long as the movement is present.

```
void check_for_movement()
{
        if (abs(filter_delta) > PIR_DETECTION_THRESHOLD )
        {
            EVSYS.USEREVSYSEVOUTB = EVSYS_USER_CHANNEL3_gc; /* Movement detected, flash LED0
*/
        }
        else
        {
            EVSYS.USEREVSYSEVOUTB = 0; /* No movement, stop flashing LED0 */
        }
}
```

4.4.6 ADC Result Ready Interrupt Routine

The ADC Result Ready interrupt routine will be called when the ADC conversion is complete, and the configured number of samples have been accumulated. Once a conversion is complete, the result will be read and stored in `adc_result`.

```
ISR(ADC0_RESRDY_vect)
{
    adc_result = ADC0.RESULT;             /* Read ADC result register */
    ADC0.INTFLAGS = ADC_RESRDY_bm;         /* Clear ADC result ready interrupt flag */
}
```

5. Summary

The example described in this application note shows how the built-in 12-bit differential ADC and Programmable Gain Amplifier in the ATtiny1627 and the tinyAVR® 2 Family can be used to design a low-power, cost-efficient, and small footprint PIR motion detection solution.

Compared to the original Click board configuration, this solution will offer a significant reduction in BOM cost and board space, saving two op amps, six resistors, and four capacitors in the signal chain. Other significant benefits compared to the original solutions is that the sensitivity, detection threshold, filtering, and signal gain can be adjusted in firmware and not in hardware, giving the developer the possibility to create the code in a way that it can adapt and change based on the needs of the application. Having access to the PIR signal gives the developer the possibility to implement smart algorithms in firmware to enable product differentiation.

The total power consumption of the system in the selected configuration is about 13.7 µA at ambient temperature. The Click board contributes about 11 µA to the total current consumption with the original bias configuration. Increasing the bias resistor R6 will reduce the Click board power consumption, at the expense of reduced output signal from the sensor.

The power consumption is application dependent and will vary based on the configuration of the PIR sensor, sample acquisition, and filtering parameters, which will also affect the detection range and/or sensitivity. Consider adjusting these parameters to further reduce the power consumption in times were the demands of the application are lower.

In the table below, there are some examples of settings and the respective MCU power consumption.

Table 5-1. MCU Average Power Consumption

Setting	MCU Power Consumption
PIR_SAMPLE_RATE_PER_SECOND = 4 Hz PIR_OVERSAMPLE_RATE = 2	1.5 µA
PIR_SAMPLE_RATE_PER_SECOND = 8 Hz PIR_OVERSAMPLE_RATE = 2	2.1 µA
PIR_SAMPLE_RATE_PER_SECOND = 16 Hz PIR_OVERSAMPLE_RATE = 2	3.5 µA
PIR_SAMPLE_RATE_PER_SECOND = 4 Hz PIR_OVERSAMPLE_RATE = 16	2.7 µA
PIR_SAMPLE_RATE_PER_SECOND = 8 Hz PIR_OVERSAMPLE_RATE = 16	4.6 µA
PIR_SAMPLE_RATE_PER_SECOND = 16 Hz PIR_OVERSAMPLE_RATE = 16	8.6 µA
PIR_SAMPLE_RATE_PER_SECOND = 4 Hz PIR_OVERSAMPLE_RATE = 64	6.1 µA
PIR_SAMPLE_RATE_PER_SECOND = 8 Hz PIR_OVERSAMPLE_RATE = 64	11.4 µA
PIR_SAMPLE_RATE_PER_SECOND = 16 Hz PIR_OVERSAMPLE_RATE = 64	21.7 µA

6. Plotting Graph in *Data Visualizer*

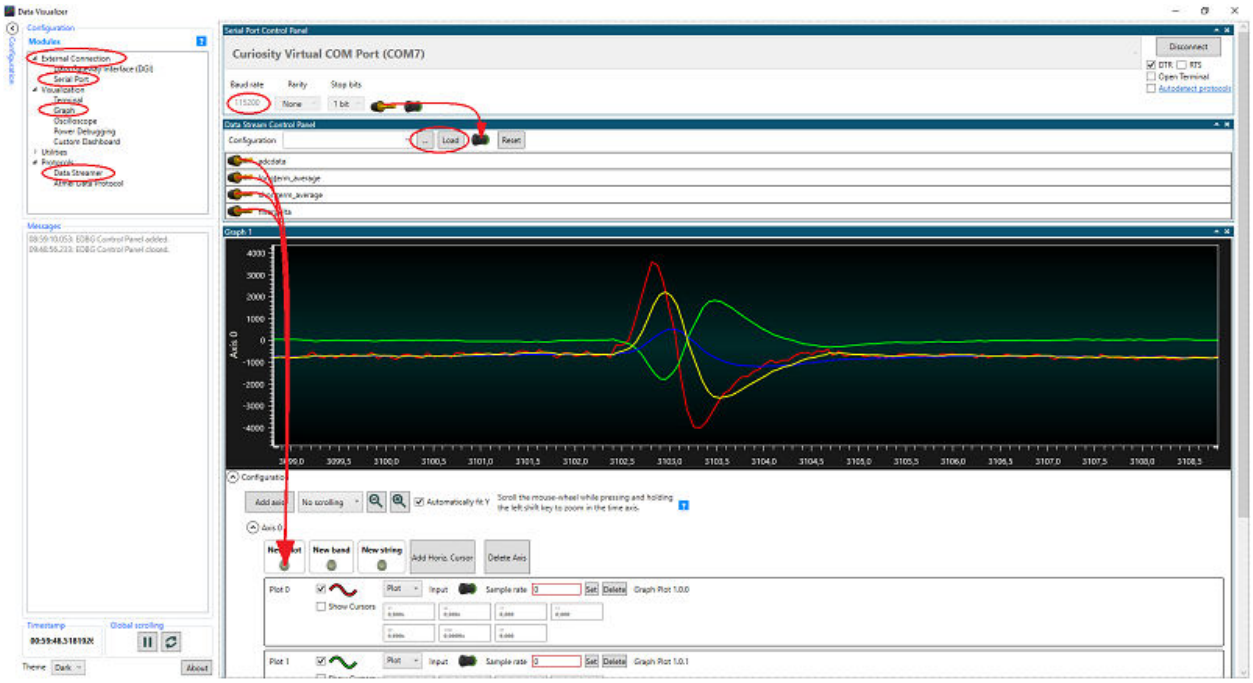
The following instructions show how to plot USART data in Data Visualizer by using the Data Stream protocol.
Note: For detailed information on **Data Visualizer,** refer to the Data Visualizer User's Guide.

1. Open **Data Visualizer**.
2. Open *Configuration > External Connection > Serial Port* in **Data Visualizer**.
3. Select the Curiosity Virtual COM port, Baud rate: 115200, and then select **Connect**.
4. Open *Configuration > Protocols > Data Streamer*.
5. In the **Data Stream Control Panel**, under **Configuration**, browse to the configuration file and then select **Load**.
   Note: In this case, the configuration file is `tiny2_PIR_datastreamer.cfg` and can be found in the example source code project folder.

   Note: For more details on the **Data Stream Protocol**, refer to Data Visualizer User's Guide, Data Stream Protocol section.

1. Open *Configuration > Visualization > Graph*.
2. Drag the connections, as shown with red arrows in the figure below, to plot the graph.

Figure 6-1. Data Stream Graph in Data Visualizer



To adjust the Y-axis in the graph, follow the steps below:

1. Under **Configuration** in **Graph**, deselect **Automatically Fit Y**.
2. Click somewhere inside the plot area.
3. Scroll the mouse-wheel while pressing or holding the **Ctrl** key.

To adjust the X-axis in the graph, follow the steps below:

1. Click somewhere inside the plot area.
2. Scroll the mouse-wheel while pressing or holding the **Shift** key.

Note: For more details on *Data Visualizer > Graph*, refer to the Data Visualizer User's Guide, Graph section.

7. Plotting Power Consumption

The following instructions show how to analyze power consumption using the Power Debugger and Data Visualizer.

Notes:

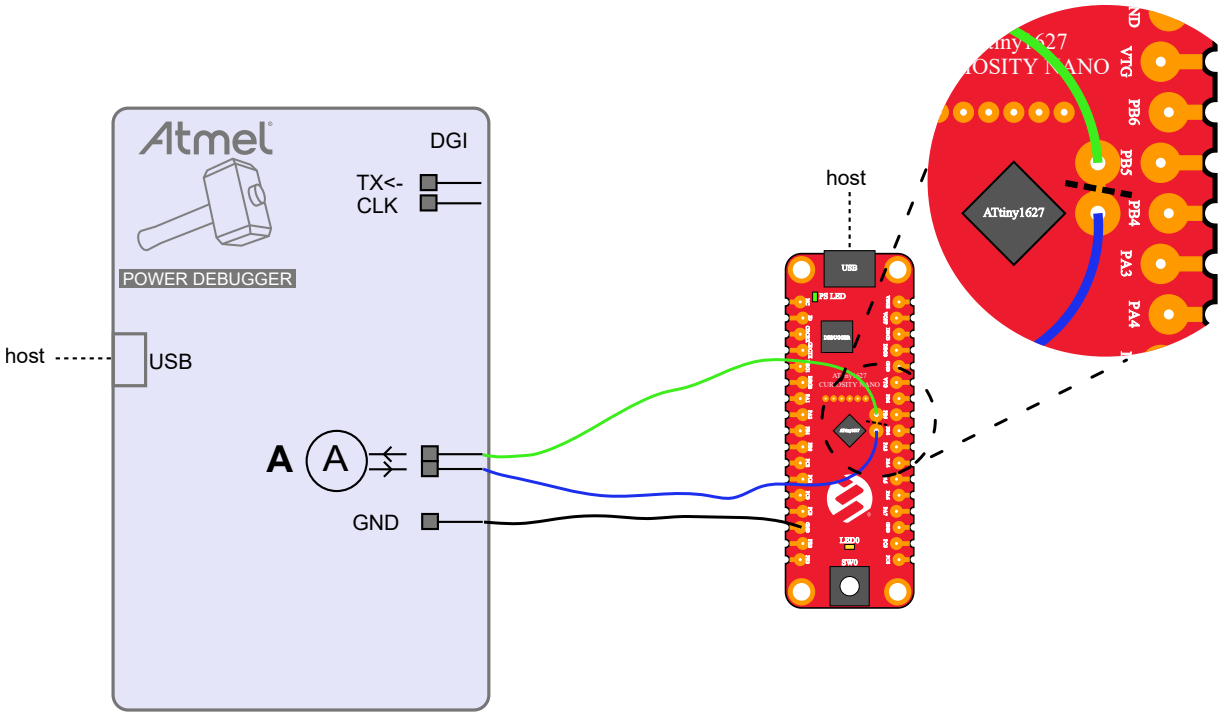
- For detailed information on the Power Debugger, refer to the Power Debugger User's Guide
- For detailed information on the Data Visualizer, refer to the Data Visualizer User's Guide

Power Debugger connection:

1. Connect one of the ground reference pins of the Power Debugger to a ground pin on the Curiosity Nano board.
2. Connect the VOFF pin on the Curiosity Nano board to a ground pin to turn off the connection between the USB power supply and the tinyAVR device.
3. Connect the VBUS pin on the Curiosity Nano board to the input current pin of measurement circuit A on the Power Debugger.
4. Connect the output current pin of measurement circuit A on the Power Debugger to the VTG pin on the Curiosity Nano board.

Measurement circuit A of the Power Debugger is now connected in series between the USB power source and the tinyAVR microcontroller and can measure the total current used.

Figure 7-1. Power Debugger Connection



Data Visualizer setup:

1. Open **Data Visualizer**.
2. Open *Configuration* > *Visualization* > *Power Debugging* in Data Visualizer.
3. On the **DGI Control Panel** pane, select *Power Debugger Data Gateway* > *Connect*.
4. Check the **Power** check box.
5. Expand the **Channel A** dropdown in the **Control Panel** in the **Power Analysis** pane.
6. Drag the plug from **A Current** in the **DGI Control Panel** pane to the **Current** socket in the **Power Analysis** pane.
7. Press **Start** in the **DGI Control Panel** pane.

8. Optional: Enable cursors by checking the **Enabled** check box in the **Cursors** dropdown.

Figure 7-2. Power Debugging Graph in Data Visualizer

7.1 Power Consumption

Using the original settings from the code, the average power consumption is very low, as shown in the figure below.

Figure 7-3. Power Analysis, the Average Current Consumption of MCU

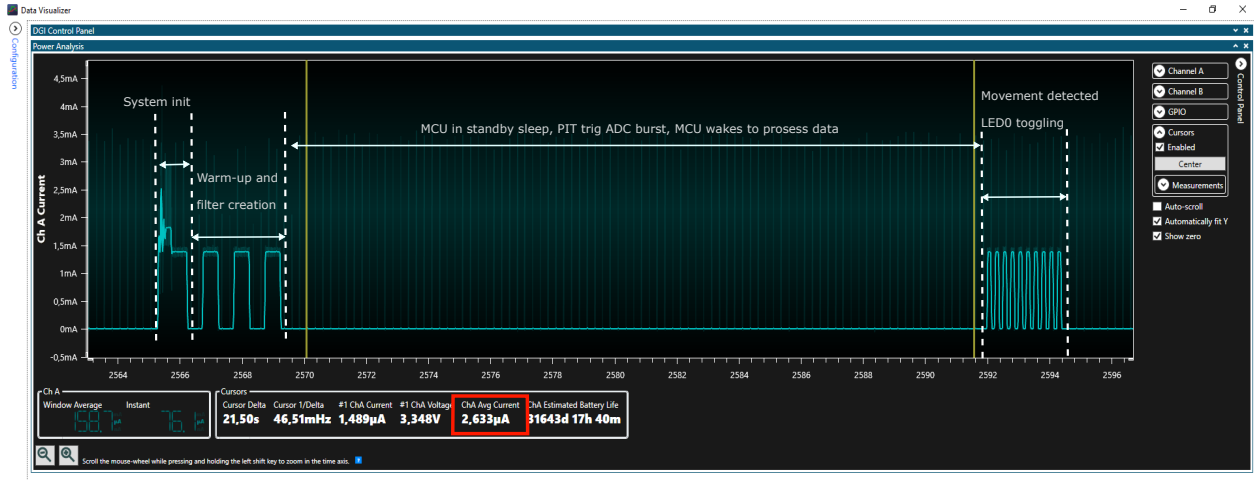


Figure 7-4. Power Analysis, Four Bursts

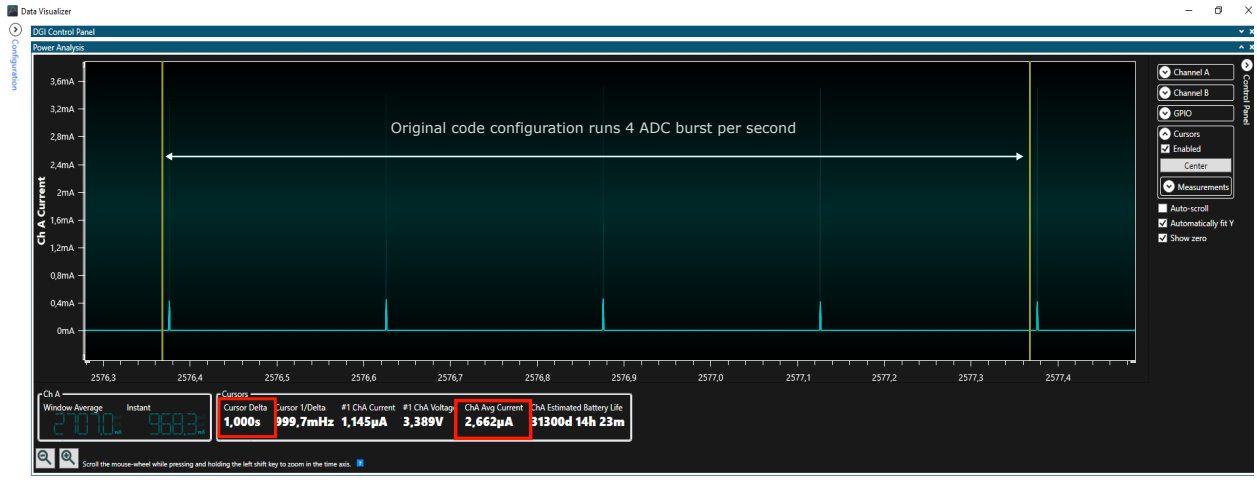
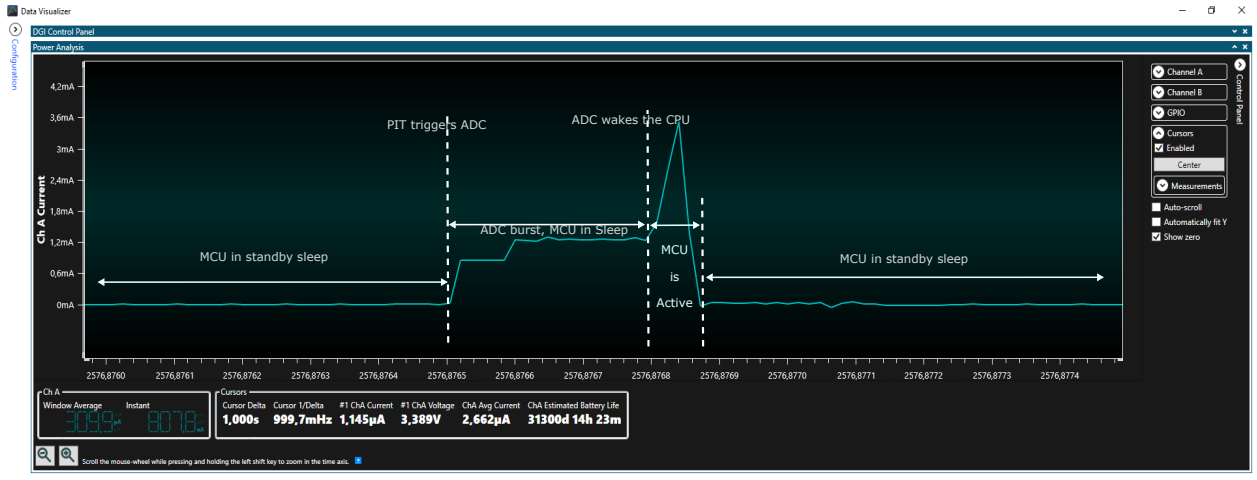


Figure 7-5. Power Analysis, one Burst



8. Get Code Examples from Atmel START

The code examples are available through Atmel START, which is a web-based tool that enables the configuration of the application code through a Graphical User Interface (GUI). The code can be downloaded for Atmel Studio

MPLAB® X and IAR Embedded Workbench® via the direct example code link below or the **Browse Examples** button on the Atmel START front page.

The Atmel START webpage: http://start.atmel.com/.

Code Examples

- PIR sensor with tinyAVR 2-series ADC
  - start.atmel.com/#example/Atmel%3AApplication_AVR_Examples%3A1.0.0%3A%3AApplication%3APIR_Motion_Detection%3A

Click **User Guide** in Atmel START for details and information about example projects. The **User Guide** button can be found in the example browser, and by clicking the project name in the dashboard view within the Atmel START project configurator.

Atmel Studio

Download the code as a `.atzip` file for Atmel Studio from the example browser in Atmel START by clicking **Download Selected example**. To download the file from within Atmel START, click **Export project** followed by **Download pack**.

Double click the downloaded `.atzip` file, and the project will be imported to Atmel Studio 7.0.

MPLAB® X

Download the code as a `.atzip` file for MPLAB X IDE from within Atmel START by clicking **Export project** followed by **Download pack**.

To open the Atmel START example in MPLAB X, select from the menu in MPLAB X, *File > Import > START MPLAB Project* and navigate to the `.atzip` file.

IAR Embedded Workbench®

For information on how to import the project in IAR Embedded Workbench, open the Atmel START User Guide, select **Using Atmel Start Output in External Tools**, and **IAR Embedded Workbench**. A link to the Atmel START User Guide can be found by clicking *Help* from the Atmel START front page or **Help And Support** within the project configurator, both located in the upper right corner of the page.

9. Get Code Examples from GitHub

The code examples are available through GitHub, which is a web-based server that provides the application codes through a Graphical User Interface (GUI). The code examples can be opened in both Atmel Studio and MPLAB X. To open the Atmel Studio project in MPLAB X, select from the menu in MPLAB X, *File > Import > Atmel Studio Project* and navigate to `.cproj` file.

The GitHub webpage: GitHub.

Code Examples

Finding example code for devices in the tinyAVR 2 family can be done by searching for the device name, e.g. ATtiny1627, in the GitHub example browser.

Download the code as a `.zip` file from the example page on GitHub by clicking the **Clone** or **download** button.

10. Revision History

Doc. Rev.	Date	Comments
A	11/2020	Initial document release

The Microchip Website

Microchip provides online support via our website at www.microchip.com/. This website is used to make files and information easily available to customers. Some of the content available includes:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQs), technical support requests, online discussion groups, Microchip design partner program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Product Change Notification Service

Microchip's product change notification service helps keep customers current on Microchip products. Subscribers will receive email notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, go to www.microchip.com/pcn and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Embedded Solutions Engineer (ESE)
- Technical Support

Customers should contact their distributor, representative or ESE for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in this document.

Technical support is available through the website at: www.microchip.com/support

Product Identification System

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods being used in attempts to breach the code protection features of the Microchip devices. We believe that these methods require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Attempts to breach these code protection features, most likely, cannot be accomplished without violating Microchip's intellectual property rights.
- Microchip is willing to work with any customer who is concerned about the integrity of its code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is "unbreakable." Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Legal Notice

Information contained in this publication is provided for the sole purpose of designing with and using Microchip products. Information regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications.

THIS INFORMATION IS PROVIDED BY MICROCHIP "AS IS". MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE OR WARRANTIES RELATED TO ITS CONDITION, QUALITY, OR PERFORMANCE.

IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL OR CONSEQUENTIAL LOSS, DAMAGE, COST OR EXPENSE OF ANY KIND WHATSOEVER RELATED TO THE INFORMATION OR ITS USE, HOWEVER CAUSED, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP'S TOTAL LIABILITY ON ALL CLAIMS IN ANY WAY RELATED TO THE INFORMATION OR ITS USE WILL NOT EXCEED THE AMOUNT OF FEES, IF ANY, THAT YOU HAVE PAID DIRECTLY TO MICROCHIP FOR THE INFORMATION. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Trademarks

The Microchip name and logo, the Microchip logo, Adaptec, AnyRate, AVR, AVR logo, AVR Freaks, BesTime, BitCloud, chipKIT, chipKIT logo, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, HELDO, IGLOO, JukeBlox, KeeLoq, Kleer, LANCheck, LinkMD, maXStylus, maXTouch, MediaLB, megaAVR, Microsemi, Microsemi logo, MOST, MOST logo, MPLAB, OptoLyzer, PackeTime, PIC, picoPower, PICSTART, PIC32 logo, PolarFire, Prochip Designer, QTouch, SAM-BA, SenGenuity, SpyNIC, SST, SST Logo, SuperFlash, Symmetricom, SyncServer, Tachyon, TimeSource, tinyAVR, UNI/O, Vectron, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

ISBN: 978-1-5224-6689-5

Quality Management System

For information regarding Microchip's Quality Management Systems, please visit www.microchip.com/quality.

Worldwide Sales and Service

AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC	EUROPE
Corporate Office 2355 West Chandler Blvd. Chandler, AZ 85224-6199 Tel: 480-792-7200 Fax: 480-792-7277 Technical Support: www.microchip.com/support Web Address: www.microchip.com **Atlanta** Duluth, GA Tel: 678-957-9614 Fax: 678-957-1455 **Austin, TX** Tel: 512-257-3370 **Boston** Westborough, MA Tel: 774-760-0087 Fax: 774-760-0088 **Chicago** Itasca, IL Tel: 630-285-0071 Fax: 630-285-0075 **Dallas** Addison, TX Tel: 972-818-7423 Fax: 972-818-2924 **Detroit** Novi, MI Tel: 248-848-4000 **Houston, TX** Tel: 281-894-5983 **Indianapolis** Noblesville, IN Tel: 317-773-8323 Fax: 317-773-5453 Tel: 317-536-2380 **Los Angeles** Mission Viejo, CA Tel: 949-462-9523 Fax: 949-462-9608 Tel: 951-273-7800 **Raleigh, NC** Tel: 919-844-7510 **New York, NY** Tel: 631-435-6000 **San Jose, CA** Tel: 408-735-9110 Tel: 408-436-4270 **Canada - Toronto** Tel: 905-695-1980 Fax: 905-695-2078	**Australia - Sydney** Tel: 61-2-9868-6733 **China - Beijing** Tel: 86-10-8569-7000 **China - Chengdu** Tel: 86-28-8665-5511 **China - Chongqing** Tel: 86-23-8980-9588 **China - Dongguan** Tel: 86-769-8702-9880 **China - Guangzhou** Tel: 86-20-8755-8029 **China - Hangzhou** Tel: 86-571-8792-8115 **China - Hong Kong SAR** Tel: 852-2943-5100 **China - Nanjing** Tel: 86-25-8473-2460 **China - Qingdao** Tel: 86-532-8502-7355 **China - Shanghai** Tel: 86-21-3326-8000 **China - Shenyang** Tel: 86-24-2334-2829 **China - Shenzhen** Tel: 86-755-8864-2200 **China - Suzhou** Tel: 86-186-6233-1526 **China - Wuhan** Tel: 86-27-5980-5300 **China - Xian** Tel: 86-29-8833-7252 **China - Xiamen** Tel: 86-592-2388138 **China - Zhuhai** Tel: 86-756-3210040	**India - Bangalore** Tel: 91-80-3090-4444 **India - New Delhi** Tel: 91-11-4160-8631 **India - Pune** Tel: 91-20-4121-0141 **Japan - Osaka** Tel: 81-6-6152-7160 **Japan - Tokyo** Tel: 81-3-6880- 3770 **Korea - Daegu** Tel: 82-53-744-4301 **Korea - Seoul** Tel: 82-2-554-7200 **Malaysia - Kuala Lumpur** Tel: 60-3-7651-7906 **Malaysia - Penang** Tel: 60-4-227-8870 **Philippines - Manila** Tel: 63-2-634-9065 **Singapore** Tel: 65-6334-8870 **Taiwan - Hsin Chu** Tel: 886-3-577-8366 **Taiwan - Kaohsiung** Tel: 886-7-213-7830 **Taiwan - Taipei** Tel: 886-2-2508-8600 **Thailand - Bangkok** Tel: 66-2-694-1351 **Vietnam - Ho Chi Minh** Tel: 84-28-5448-2100	**Austria - Wels** Tel: 43-7242-2244-39 Fax: 43-7242-2244-393 **Denmark - Copenhagen** Tel: 45-4485-5910 Fax: 45-4485-2829 **Finland - Espoo** Tel: 358-9-4520-820 **France - Paris** Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79 **Germany - Garching** Tel: 49-8931-9700 **Germany - Haan** Tel: 49-2129-3766400 **Germany - Heilbronn** Tel: 49-7131-72400 **Germany - Karlsruhe** Tel: 49-721-625370 **Germany - Munich** Tel: 49-89-627-144-0 Fax: 49-89-627-144-44 **Germany - Rosenheim** Tel: 49-8031-354-560 **Israel - Ra'anana** Tel: 972-9-744-7705 **Italy - Milan** Tel: 39-0331-742611 Fax: 39-0331-466781 **Italy - Padova** Tel: 39-049-7625286 **Netherlands - Drunen** Tel: 31-416-690399 Fax: 31-416-690340 **Norway - Trondheim** Tel: 47-72884388 **Poland - Warsaw** Tel: 48-22-3325737 **Romania - Bucharest** Tel: 40-21-407-87-50 **Spain - Madrid** Tel: 34-91-708-08-90 Fax: 34-91-708-08-91 **Sweden - Gothenberg** Tel: 46-31-704-60-40 **Sweden - Stockholm** Tel: 46-8-5090-4654 **UK - Wokingham** Tel: 44-118-921-5800 Fax: 44-118-921-5820