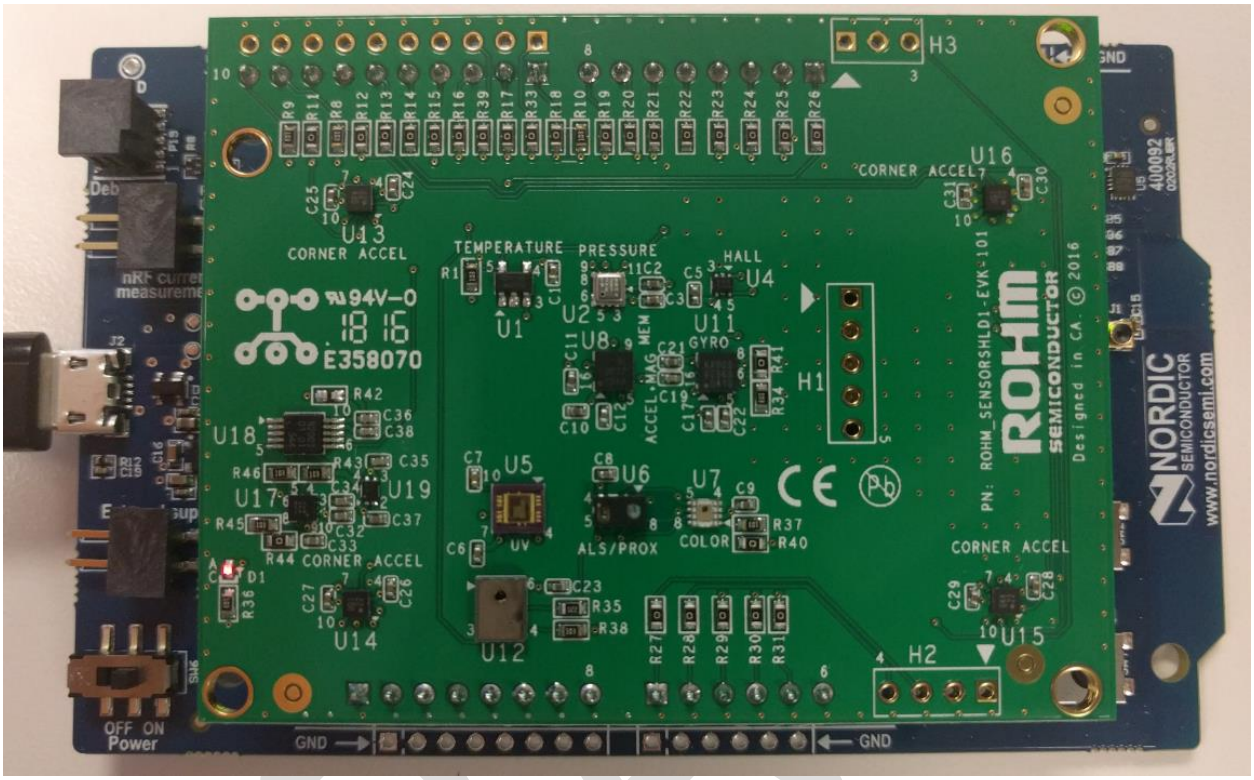
ROHM SEMICONDUCTOR
U.S. DESIGN CENTER

Connecting the ROHM Sensor Shield to the Nordic nRF51-DK



Above: ROHM SENSORSHLD1-EVK-101 directly connected to the Nordic nRF51-DK Platform

Table of Contents

Copyright and License

[Need to add Nordic References Here]

Please note that all references to ROHM's Multi-Sensor Shield are also shared under open source guidelines under the GNU General Public License, Version 3. Details can be found at the following link: https://github.com/ROHMUSDC/ROHM_SensorPlatform_Multi-Sensor-Shield.

Revision History

Date	Description	Revision ID
11 September 2015	First Draft	A
25 November 2015	Added Additional Sensor Information	A

Introduction

The following document was written to provide a brief connection guide and starting point for using ROHM's Multi-sensor shield with the Nordic nRF51-DK. This guide assumes that the user has basic functional knowledge of both the ROHM Multi-Sensor Shield and the Nordic Platform itself. If this is not correct, please see the following links for other guides and information on these products.

ROHM's Multi-Sensor Shield GitHub Repository Page:
https://github.com/ROHMUSDC/ROHM_SensorPlatform_Multi-Sensor-Shield

Nordic NRF51-DK MBED Platform Page: https://developer.mbed.org/platforms/Nordic-nRF51-DK/

Please note that the Nordic NRF51-DK Platform is a microcontroller platform; thus, this document will explain and show examples of how to connect to and convert values for the ROHM Sensor Kit Sensors. This includes explanations on the following sensors:

- ROHM BDE0600G – Analog Temperature Sensor
- LAPIS ML8511 – Analog UV Sensor
- ROHM BU52011HFV – Hall Switch Sensor
- KIONIX KMX62 – Digital Accelerometer and Magnetometer
- ROHM BM1383AGLV – Digital Barometric Pressure Sensor
- ROHM RPR-0521 – Digital Ambient Light Sensor and Proximity Sensor
- ROHM BH1745 – Digital Color Sensor
- KIONIX KX122 – Digital Accelerometer
- KIONIX KXG03 – Digital Gyroscopic Sensor
- ROHM BM1422 – MI Magnetometer Sensor

Code Differences between ROHM SHLD0 and SHLD1

- Pressure Sensor is different on this board and uses an updated register map. See the Pressure sensor datasheet for more details (Temperature and pressure output are switched in I2C register map)
- Added section for newly added BM1422 Magnetometer Sensor IC
- Gyro Device I2C address changed due to conflict with other sensors. 7bit address was 0x4F and is not 0x4E (ADDR tied to GND on EVK board)

Getting Started

1. Initial Setup
    a. ROHM Multi-Sensor Shield Board
        i. For this guide we will connect ROHM's Multi-Sensor Shield board directly to the Nordic Platform using the standard Arduino Shield Headers
    b. The following are recommended for using the Nordic Platform for this guide
        i. PC
        ii. MBED account for online compiler access
        iii. USB Cable to power and program
        iv. Import the example code from the following repository on the MBED site into your online MBED complier
            1. **https://developer.mbed.org/teams/ROHMUSDC/code/Nordic_UART_TEMPLATE_ROHM_SHLD1Update/**

Demo Operation Quick Start Guide

This section of this document is intended for using this code to demo the sensor shield without deep diving into code provided for this shield. This section will explain where to find relevant files in GitHub, how to program the nRF51-DK using the pre-compiled HEX files, and how to get view data output from the nRF51-DK on a BTLE enabled phone (iPhone or Android environment).

Preparing the Environment

For operating the shield demo, you will need the following:

- PC environment
- nRF51-DK Board
- Android or iPhone
- ROHM Sensor Shield (SENSORSHLD1-EVK-101)

Download the HEX Output File to Load into the nRF51-DK

First, we want to download the compiled HEX output to program the nRF51-DK Board. This can be found at the GitHub repository at the following link:

https://github.com/ROHMUSDC/ROHM_SensorPlatform_Multi-Sensor-Shield

Location: "…ROHM_SensorPlatform_Multi-Sensor-Shield\Platform Code\Nordic_nRF51-DK_FirmwareExample\SHLD1\HEX Output Files\Nordic_nRF51-DK_ROHMSensorShield_SHLD1Update_NRF51_DK.hex"

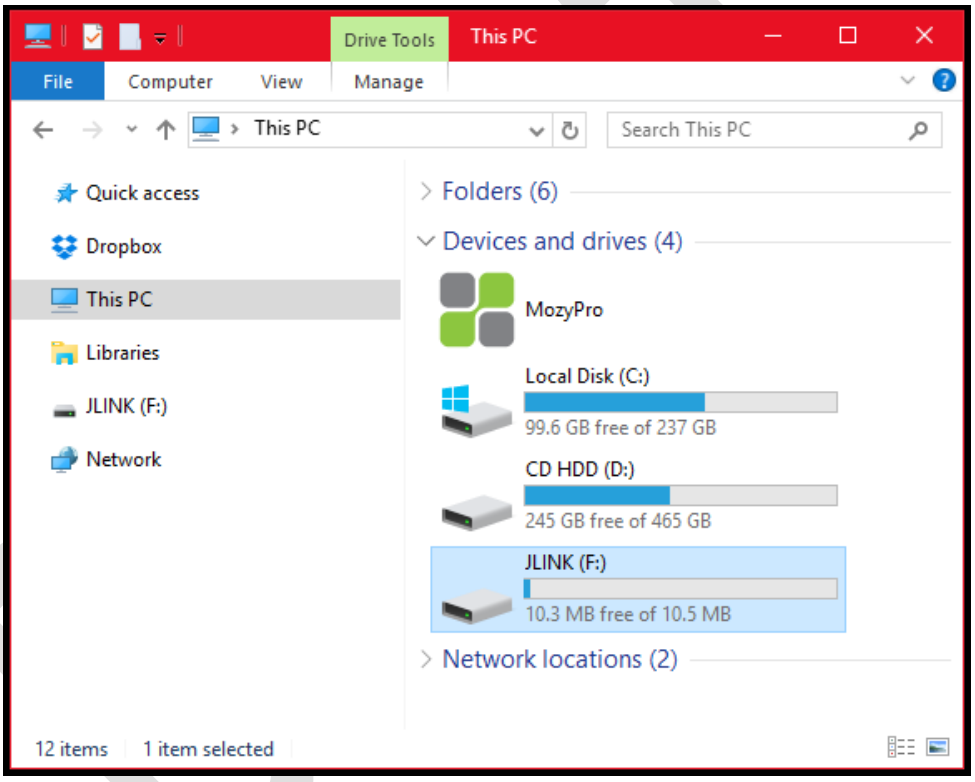
For this, please be sure to check your ROHM Sensor Shield board (SENSORSHLD1-EVK-101) for a sticker on board. If there is a yellow sticker on board, please note that you should use the other file named "Nordic_nRF51-DK_ROHMSensorShield_SHLD1Update_NRF51_DK_YELLOWSTICKER.hex".

The yellow sticker on-board designates that the board was built using the same SCH/BOM/LAYOUT as the SENSORSHLD1-EVK-101 design files show, however implements BM1383GLV instead of BM1383AGLV. Please refer to the sensor difference explanation between the BM1383GLV vs. BM1383AGLV at the following repository link:

> Location: ...ROHM_SensorPlatform_Multi-Sensor-Shield\Documentation\Sensor Datasheets\ROHM_PRESSURE _BM1383xGLV\BM1383AGLV_changing_specification160127.pdf

Load the HEX file onto the nRF51-DK Board

Next, connect the nRF51-DK to the PC. If all the correct drivers have been installed, this device will connect to the PC as a removable device named "JLINK".



Once you see this file, copy and paste the appropriate *.hex file downloaded from the previous section into the "JLINK" device.

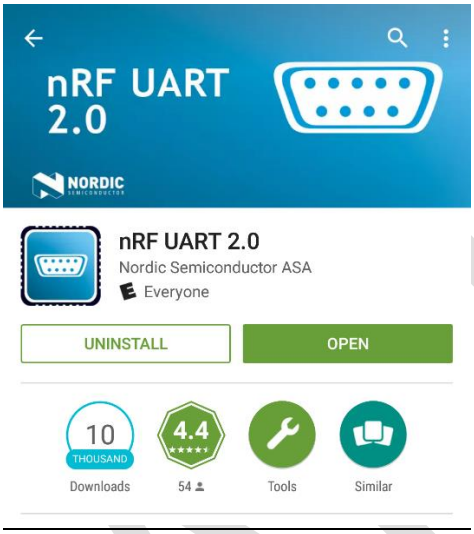
After completing the copy, the "JLINK" device will disconnect from the PC and immediately reconnect to the PC. Upon reconnecting to the PC, note that the *.hex file will not be there; however, please note that if the above sequence completed successfully, then the nRF51-DK will be programmed correctly.

Once the above is complete, connect the ROHM Sensor Shield to the nRF51-DK to the board and power on the shield board.

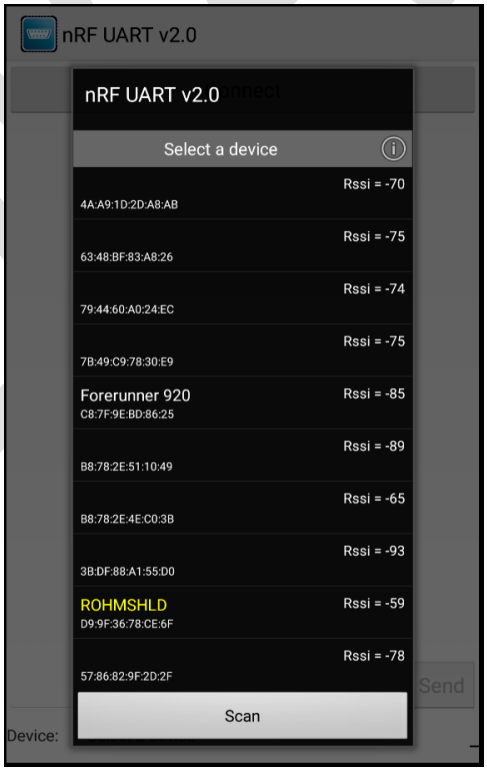
Viewing Sensor Output Messages

After completing the above, we can now connect the nRF51-DK to the Phone.

To do this, we recommend downloading the nRF UART App (generated and built by Nordic Semiconductor) from the respective Apple App store or Google Play store.



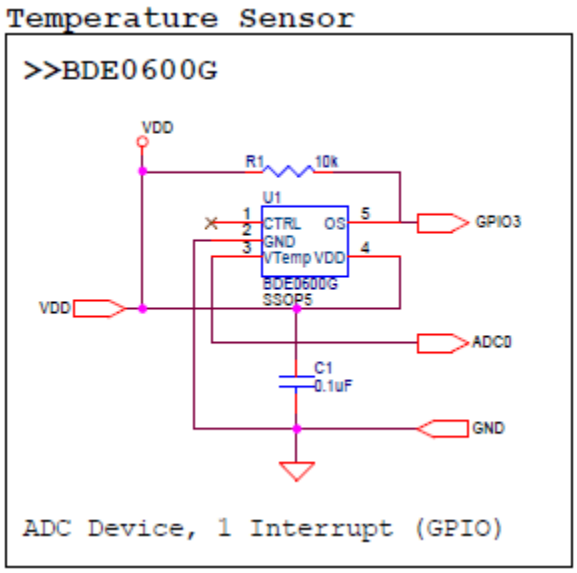
Once this is downloaded, open the app and click the connect button. The nRF51-DK should be shown in the app list as "ROHMSHLD". Once you've found this device, click the device to connect.



After a couple seconds, the App should connect with the nRF51-DK board. Then the nRF51-DK board will start streaming the sensor data directly to the phone application.

Connecting ROHM Multi-Sensor Shield to the Nordic NRF51-DK Platform

Hardware Connection for ROHM BDE0600G Temp Sensor to the Nordic Platform

- ROHM BDE0600G – Analog Temperature Sensor
  - 

- As seen in the schematic above, this device connects 4 pins.
  - VDD – Connect to Nordic 3.3V output pin on power connector
  - GND – Connect to Nordic GND pin on the power connector
  - ADC0 – ADC Output for Temperature Readings, Connected to pin P0.03 on Nordic Board
  - GPIO3 – This pin is used to trigger the thermostat output function. This pin is not used in the example application.

Software Explanation for ROHM BDE0600G Temp Sensor to the Nordic Platform

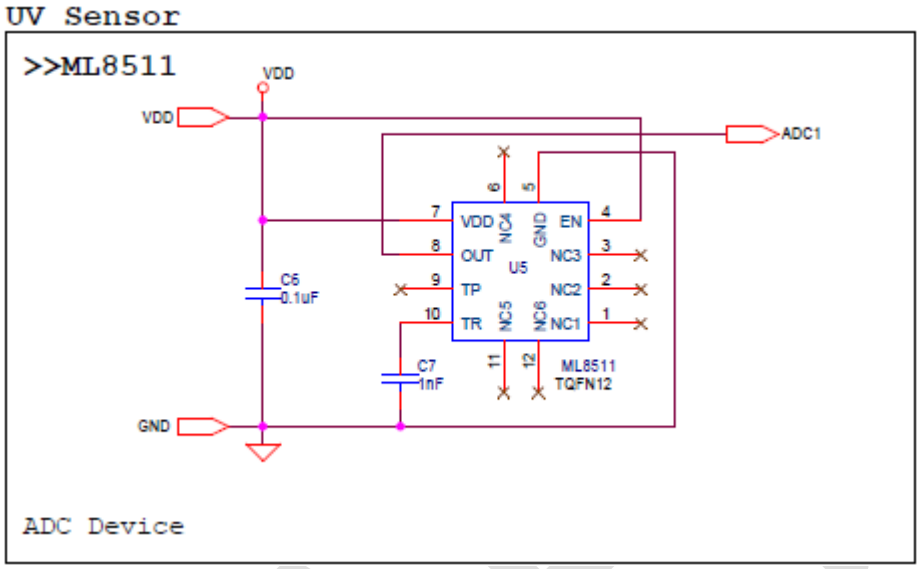
- Code Segments pertaining to this sensor can be found by defining and seeing code within the "***AnalogTemp***" #ifDef statements
- Pseudo-Code Explanation
  - Define Relevant Variables
  - Begin Loop()
    1. Read back the Analog Sensor Output from Pin P0.03
       - Note: Default Nordic Reference voltage is 5V; however we are supplying 3.3V to the sensor. Take these values into account when performing your conversions!
    2. Convert value to V, then to Temperature reading
       - Known Values
         - Temperature Sensitivity = -10.68mV/degC

- Temperature Sensitivity = -0.01068V/degC
- Temp Known Point = 1.753V @ 30 degC

- Calculation
  - ADC_Voltage = (sensorValue / 1024) * 2.9V
  - ADC_Voltage = sensorValue * (2.9V/1024)
  - ADC_Voltage = sensorValue * 0.00283
  - Temperature (in deg C) = (ADC_Voltage - 1.753)/(-0.01068) + 30

3. Format Serial Output and Return Information

Hardware Connection for LAPIS ML8511 UV Sensor to the Nordic Platform

- LAPIS ML8511 – Analog UV Sensor
  - 

- As seen in the schematic above, this device connects 3 pins.
  - VDD – Connect to Nordic 3.3V output pin on power connector
  - GND – Connect to Nordic GND pin on the power connector
  - ADC1 – ADC Output for UV Sensor Output, Connected to pin P0.05 on Nordic Board

Software Explanation for LAPIS ML8511 UV Sensor to the Arduino Uno

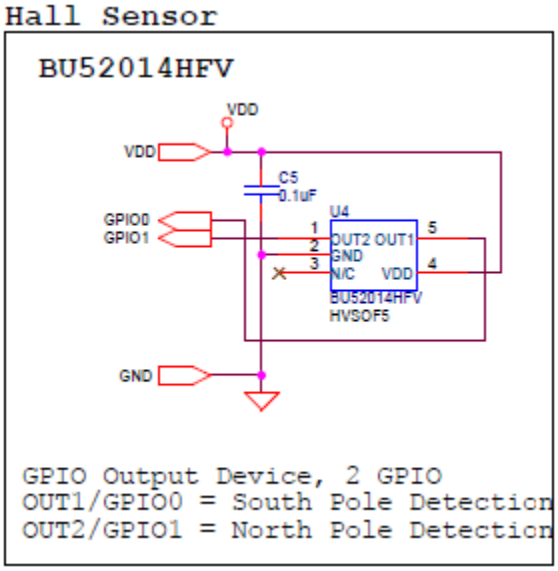
- Code Segments pertaining to this sensor can be found by defining and seeing code within the "***AnalogUV***" #ifDef statements
- Pseudo-Code Explanation
  - Define Relevant Variables
  - Begin Loop()
    1. Read back the Analog Sensor Output from Pin P0.05
       - Note: Default Nordic Reference voltage is 5V; however we are supplying 3.3V to the sensor. Take these values into account when performing your conversions!
    2. Convert value to V, then to UV Intensity
       - Known Values
         - UV Sensitivity = 0.129 V/(mW/cm^2)
         - Temp Known Point = 2.2V @ 10mW/cm^2
       - Calculation
         - ADC_Voltage = (sensorValue / 1024) * 2.9V
         - ADC_Voltage = sensorValue * (2.9V/1024)

- ADC_Voltage = sensorValue * 0.0029
- UV Intensity (in mW/cm^2)= (ADC_Voltage - 2.2)/(0.129) + 10

3. Format Serial Output and Return Information

Hardware Connection for ROHM BU52014 Hall Sensor to the Nordic Platform

- ROHM BU52011HFV – Hall Switch Sensor

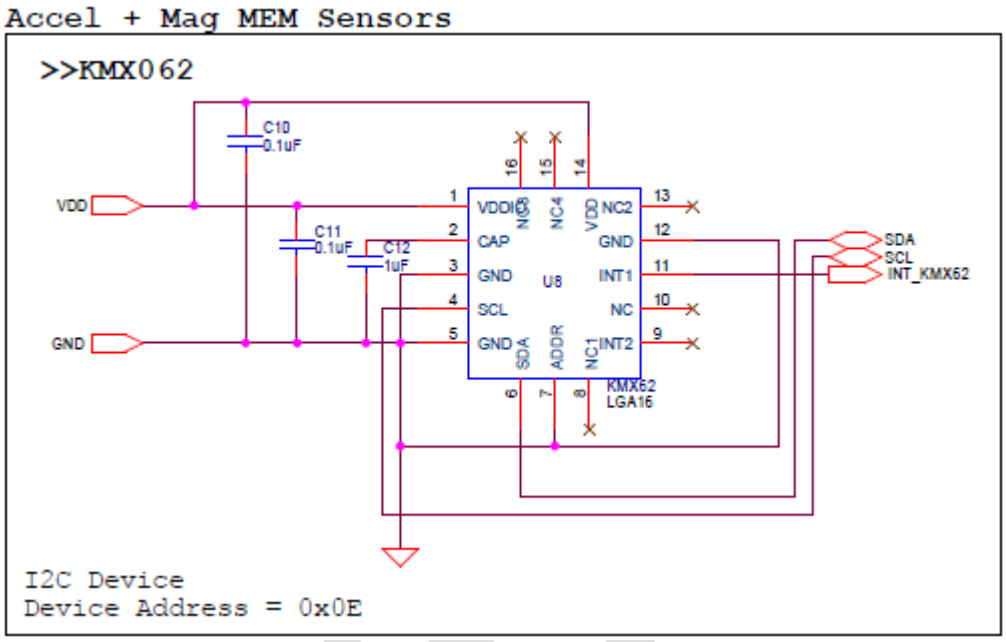


- As seen in the schematic above, this device connects 4 pins.
  - VDD – Connect to Nordic 3.3V output pin on power connector
  - GND – Connect to Nordic GND pin on the power connector
  - GPIO0, GPIO1 – Indicates Hall Switch Output
    - GPIO0: Indicates South Pole Detection
    - GPIO1: Indicates North Pole Detection

Software Explanation for ROHM BU52014 Hall Sensor to the Nordic Platform

- Code Segments pertaining to this sensor can be found by defining and seeing code within the "***HallSensor***" #ifDef statements
- Pseudo-Code Explanation
  - Define Relevant Variables
  - Begin setup()
    1. Setup Nordic Pins P0.14 and P0.15 and Input pins
  - Begin loop()
    1. Perform digital reads on pins P0.14 and P0.15
       - Output will be either 1 or 0 and will indicate presence of north or south magnetic fields
    2. Format Serial Output and Return Information

Hardware Connection for Kionix KMX62 Accel+Mag Sensor to the Nordic Platform

- Kionix KMX62 – Digital Accelerometer and Magnetometer
  - 

- As seen in the schematic above, this device connects 6 pins.
  - VDD – Connect to Nordic 3.3V output pin on power connector
  - GND – Connect to Nordic GND pin on the power connector
  - SDA/SCL – I2C Connection, connected to pin P0.30 and P0.07
  - INT_KMX62 - This pin is used monitor the interrupt pins on the KMX61. This pin is not used in the example application.

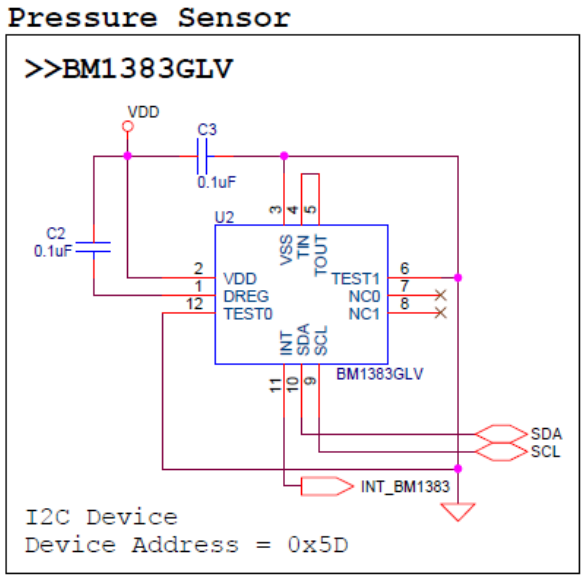
Software Explanation for Kionix KMX62 Accel+Mag Sensor to the Nordic Platform

- Code Segments pertaining to this sensor can be found by defining and seeing code within the "***KMX62***" #ifDef statements
- Pseudo-Code Explanation
  - Define Relevant Variables
  - Begin Main()
    1. Set Baud Rate for Serial Output via Nordic BT Protocols
    2. Initialize Board Level LEDs
    3. Setup Interrupt Callback for "Periodic Callback"
    4. Setup PB for SW4
    5. Configure the Accel/Mag Sensor once by performing the following writes
       - 1. CNTL2 Register (0x3A), write 0x5F (4g Sensor Resolution, Max RES, Enable Temp/Mag/Accel)

- Note: Please see the KMX62 Datasheet for additional information on these registers

- Begin loop()
  1. Read back the Accelerometer output by reading 6 Bytes starting from address 0x0A. [0][1]...[5]
  2. Format Each of the X, Y, and Z axis acceleration
     - *Note:* to save some time with conversions, please note that these outputs are meant to be unsigned 14bit values. Thus, to make it easier to convert we recommend the following…
       - Xout = (float)(([1]<<8) | ([0]))/4
       - Yout = (float)(([3]<<8) | ([2]))/4
       - Zout = (float)(([5]<<8) | ([4]))/4
  3. Convert Returned Value to g
     - Axis_ValueInG = MEMS_Accel_axis / 2048
  4. Read back the Magnetometer output by reading 6 Bytes starting from address 0x10. [0][1]...[5]
  5. Format Each of the X, Y, and Z axis mag data
     - *Note:* to save some time with conversions, please note that these outputs are meant to be unsigned 14bit values. Thus, to make it easier to convert we recommend the following…
       - Xout = (float)(([1]<<8) | ([0]))/4
       - Yout = (float)(([3]<<8) | ([2]))/4
       - Zout = (float)(([5]<<8) | ([4]))/4
  6. Convert Returned Value to uT
     - Axis_ValueInuT = MEMS_Mag_axis * 0.146
  7. Format Serial Output and Return Information

Hardware Connection for ROHM BM1383GLV Barometric Pressure Sensor to the Nordic Platform

- ROHM BM1383GLV Barometric Pressure Sensor
  - 

- As seen in the schematic above, this device connects 6 pins.
  - VDD – Connect to Nordic 3.3V output pin on power connector
  - GND – Connect to Nordic GND pin on the power connector
  - SDA/SCL – I2C Connection, connected to pin P0.30 and P0.07
  - INT_BM1383 - This pin is used monitor the interrupt pins on the BM1383. This pin is not used in the example application.

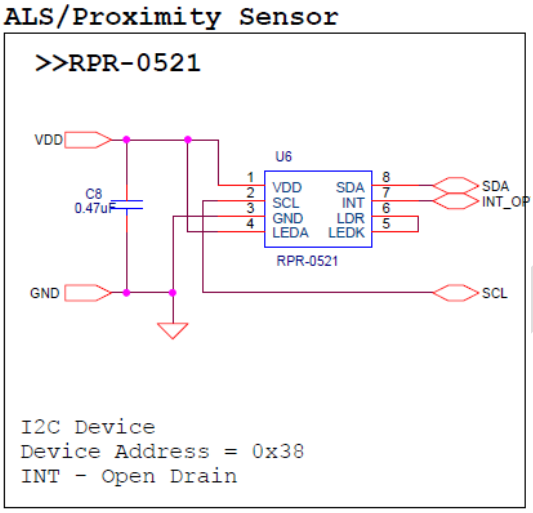
Software Explanation for ROHM BM1383GLV Barometric Pressure Sensor to the Nordic Platform

- Code Segments pertaining to this sensor can be found by defining and seeing code within the "***Pressure***" #ifDef statements
- Pseudo-Code Explanation
  - Define Relevant Variables
  - Begin Main()
    1. Set Baud Rate for Serial Output via Nordic BT Protocols
    2. Initialize Board Level LEDs
    3. Setup Interrupt Callback for "Periodic Callback"
    4. Configure the Pressure Sensor once by performing the following writes:
       - 1. PWR_DOWN Register (0x12), write (0x01)
       - 2. SLEEP Register (0x13), write (0x01)
       - 3. Mode Control Register (0x14), write (0xC4)

- Note: Please see the BM1383 datasheet for additional information on these registers

- Begin loop()
  1. Read back the pressure and temperature output by reading 5 Bytes starting from address 0x1A. [0][1]...[4]
  2. Format raw temperature and pressure
     - Raw ADC Pressure (Integer Portion) = ([0]<<3) | ([1]>>5)
     - Raw ADC Pressure (Decimal Portion) = (([1] & 0x1f) << 6 | (([2] >> 2)));
     - Raw ADC Temp = ([3]<<8) | ([4])
  3. Convert Returned Raw Value to Real Values
     - Temp in C = Raw ADC Temp / 32
     - Pressure in hPa =
       - (Raw ADC Pressure(int)) + (Raw ADC Pressure(dec)*2^{-11})
  4. Format Serial Output and Return Information

Hardware Connection for ROHM RPR-0521 3-in-1 Ambient Light Sensor, Proximity Sensor, and IR LED Combo Package for the Nordic Platform

- ROHM RPR-0521 ALS/PROX Sensor
  - 

- As seen in the schematic above, this device connects 6 pins.
  - VDD – Connect to Nordic 3.3V output pin on power connector
  - GND – Connect to Nordic GND pin on the power connector
  - SDA/SCL – I2C Connection, connected to pin P0.30 and P0.07
  - INT_Optical - This pin is used monitor the interrupt pins on the RPR-0521. This pin is not used in the example application.

Software Explanation for ROHM RPR-0521 3-in-1 Ambient Light Sensor, Proximity Sensor, and IR LED Combo Package to the Nordic Platform

- Code Segments pertaining to this sensor can be found by defining and seeing code within the "***RPR0521***" #ifDef statements
- Pseudo-Code Explanation
  - Define Relevant Variables
  - Begin Main()
    1. Set Baud Rate for Serial Output via Nordic BT Protocols
    2. Initialize Board Level LEDs
    3. Setup Interrupt Callback for "Periodic Callback"
    4. Configure the Ambient Light Sensor and Proximity Sensor once by performing the following writes:
       - 1. Mode Control Register (0x41), write (0xC6)
       - 2. ALS_PS_Control Register (0x42), write (0x03)
       - 3. Persist Register (0x43), write (0x20)
       - Note: Please see the RPR-0521 datasheet for additional information on these registers

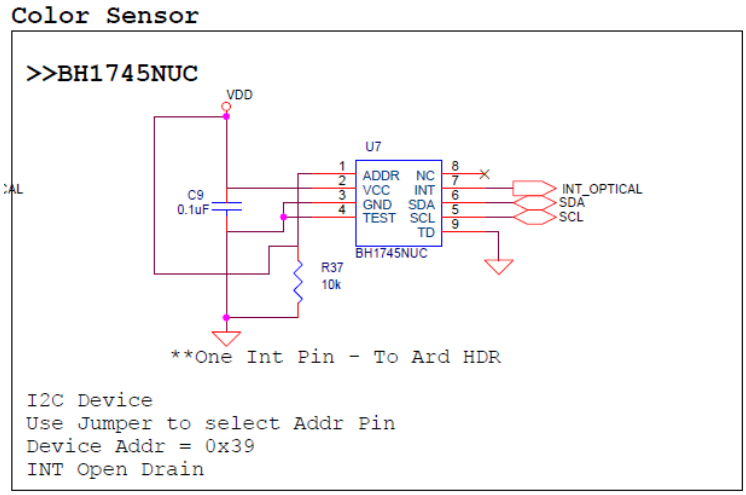
- Begin loop()
    1. Read back the ALS output by reading 6 Bytes starting from address 0x44. [0][1]...[5]
    2. Format Raw ambient light and proximity
        - RAW PROX ADC = ([1]<<8) | ([0])
        - RAW ALS DATA0 = ([3]<<8) | ([2])
        - RAW ALS DATA1 = ([5]<<8) | ([4])
    3. Convert Raw Returned Value to Real Values
        - ALS can be returned by using the following code block to return ALS value in lx (RPR0521_ALS_OUT)

```
RPR0521_ALS_DataRatio = (float) RPR0521_ALS_D1_RAWOUT /
(float)RPR0521_ALS_D0_RAWOUT;

if(RPR0521_ALS_DataRatio < 0.595){
      RPR0521_ALS_OUT = (1.682*(float) RPR0521_ALS_D0_RAWOUT - 1.877*(float)
      RPR0521_ALS_D1_RAWOUT);
}
else if(RPR0521_ALS_DataRatio < 1.015){
      RPR0521_ALS_OUT = (0.644*(float) RPR0521_ALS_D0_RAWOUT - 0.132*(float)
      RPR0521_ALS_D1_RAWOUT);
}
else if(RPR0521_ALS_DataRatio < 1.352){
      RPR0521_ALS_OUT = (0.756*(float) RPR0521_ALS_D0_RAWOUT - 0.243*(float)
      RPR0521_ALS_D1_RAWOUT);
}
else if(RPR0521_ALS_DataRatio < 3.053){
      RPR0521_ALS_OUT = (0.766*(float) RPR0521_ALS_D0_RAWOUT - 0.25*(float)
      RPR0521_ALS_D1_RAWOUT);
}
else{
      RPR0521_ALS_OUT = 0;
}
```

    4. Format Serial Output and Return Information

Hardware Connection for ROHM BH1745NUC Color Sensor for the Nordic Platform

- ROHM BH1745NUC Color Sensor
  - 

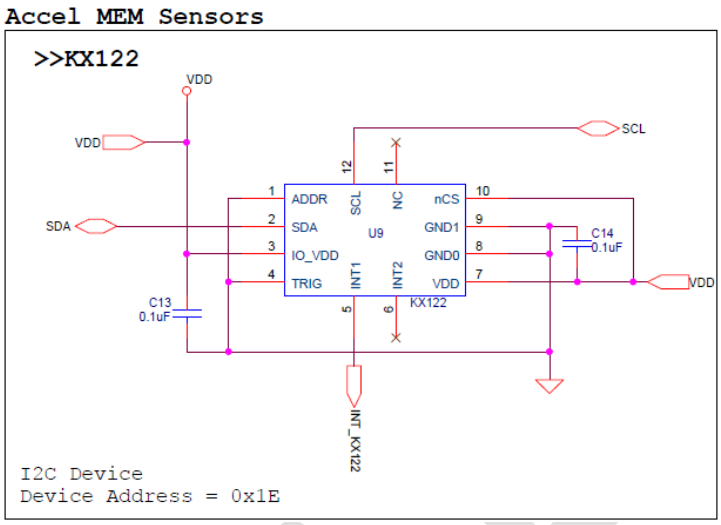
- As seen in the schematic above, this device connects 6 pins.
  - VDD – Connect to Nordic 3.3V output pin on power connector
  - GND – Connect to Nordic GND pin on the power connector
  - SDA/SCL – I2C Connection, connected to pin P0.30 and P0.07
  - INT_Optical - This pin is used monitor the interrupt pins on the RPR-0521. This pin is not used in the example application.

Software Explanation for ROHM BH1745NUC Color Sensor to the Nordic Platform

- Code Segments pertaining to this sensor can be found by defining and seeing code within the "***Color***" #ifDef statements
- Pseudo-Code Explanation
  - Define Relevant Variables
  - Begin Main()
    1. Set Baud Rate for Serial Output via Nordic BT Protocols
    2. Initialize Board Level LEDs
    3. Setup Interrupt Callback for "Periodic Callback"
    4. Configure the Color Sensor once by performing the following writes:
       - 1. Persistence Register (0x61), write (0x03)
       - 2. Mode Control 1 Register (0x41), write (0x00)
       - 3. Mode Control 2 Register (0x42), write (0x92)
       - 4. Mode Control 3 Register (0x43), write (0x02)
       - Note: Please see the color sensor datasheet for additional information on these registers

- Begin loop()
    1. Read back the color output by reading 6 Bytes starting from address 0x50. [0][1]...[5]
    2. Format Raw color
        - RED ADC = ([1]<<8) | ([0])
        - GREEN ADC = ([3]<<8) | ([2])
        - BLUE ADC = ([5]<<8) | ([4])
    3. Convert Raw Returned Value to Real Values
        - Coming Soon!
    4. Format Serial Output and Return Information

Hardware Connection for Kionix KX122 Accelerometer Sensor for the Nordic Platform

- Kionix KX122 Accelerometer Sensor
  - 

- As seen in the schematic above, this device connects 6 pins.
  - VDD – Connect to Nordic 3.3V output pin on power connector
  - GND – Connect to Nordic GND pin on the power connector
  - SDA/SCL – I2C Connection, connected to pin P0.30 and P0.07
  - INT_KX122 - This pin is used monitor the interrupt pins on the KX122. This pin is not used in the example application.

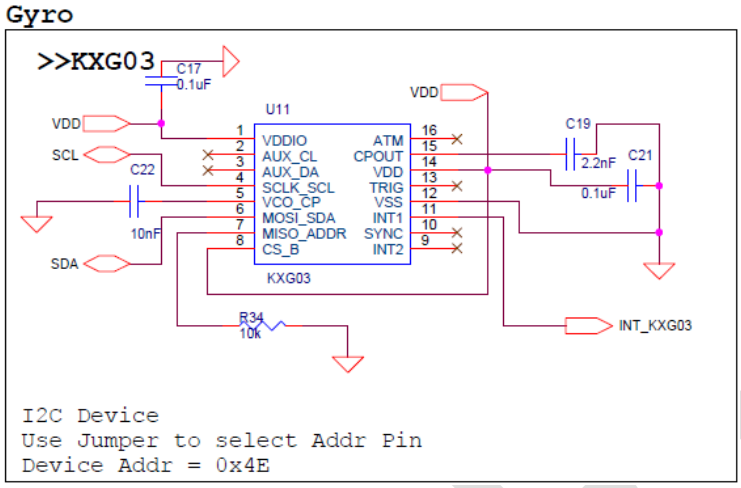
Software Explanation for Kionix KX122 Accelerometer Sensor to the Nordic Platform

- Code Segments pertaining to this sensor can be found by defining and seeing code within the "***KX122***" #ifDef statements
- Pseudo-Code Explanation
  - Define Relevant Variables
  - Begin Main()
    1. Set Baud Rate for Serial Output via Nordic BT Protocols
    2. Initialize Board Level LEDs
    3. Setup Interrupt Callback for "Periodic Callback"
    4. Configure the Accelerometer Sensor once by performing the following writes:
       - 1. CNTL1 Register (0x18), write (0x41)
       - 2. ODCNTL Register (0x1B), write (0x02)
       - 3. CNTL3 Register (0x1A), write (0xD8)
       - 4. TILT_TIMER Register (0x22), write (0x01)
       - 5. CNTL1 Register (0x18), write (0xC1) (Enable bit to ON)

- Note: Please see the KX122 datasheet for additional information on these registers

- Begin loop()
  1. Read back the Accelerometer output by reading 6 Bytes starting from address 0x06. [0][1]...[5]
  2. Format Raw accelerometer
     - ACCEL X Axis Raw = ([1]<<8) | ([0])
     - ACCEL Y Axis Raw = ([3]<<8) | ([2])
     - ACCEL Z Axis Raw = ([5]<<8) | ([4])
  3. Convert Raw Returned Value to g
     - ACCEL X in g = ACCEL X Axis Raw / 16384
     - ACCEL X in g = ACCEL Y Axis Raw / 16384
     - ACCEL X in g = ACCEL Z Axis Raw / 16384
  4. Format Serial Output and Return Information

Hardware Connection for Kionix KXG03 Gyroscopic Sensor for the Nordic Platform

- Kionix KXG03 Gyroscopic Sensor
  - 

- As seen in the schematic above, this device connects 6 pins.
  - VDD – Connect to Nordic 3.3V output pin on power connector
  - GND – Connect to Nordic GND pin on the power connector
  - SDA/SCL – I2C Connection, connected to pin P0.30 and P0.07
  - INT_KXG03 - This pin is used monitor the interrupt pins on the KXG03. This pin is not used in the example application.

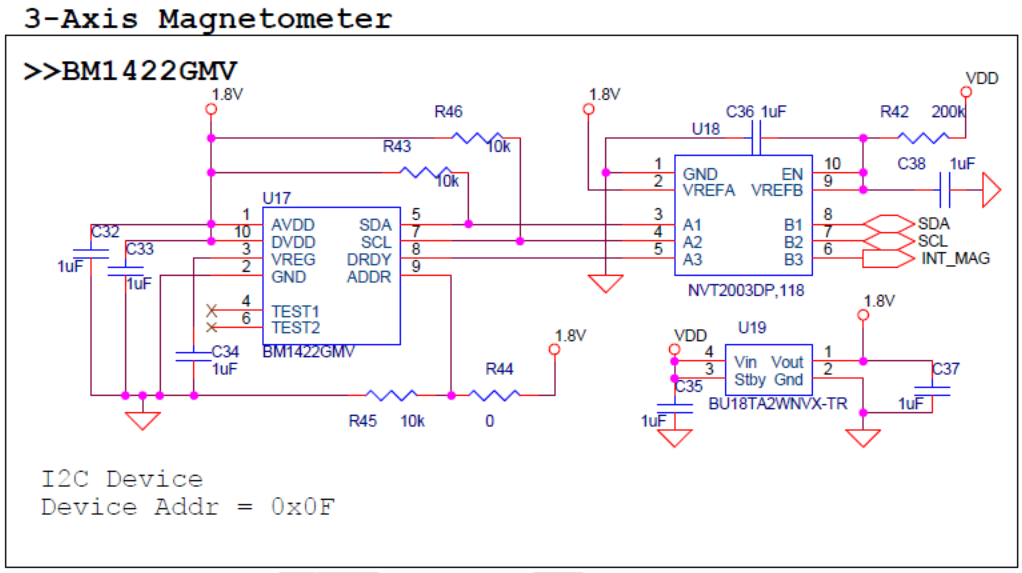
Software Explanation for Kionix KXG03 Gyroscopic Sensor to the Nordic Platform

- Code Segments pertaining to this sensor can be found by defining and seeing code within the "***KXG03***" #ifDef statements
- Pseudo-Code Explanation
  - Define Relevant Variables
  - Begin Main()
    1. Set Baud Rate for Serial Output via Nordic BT Protocols
    2. Initialize Board Level LEDs
    3. Setup Interrupt Callback for "Periodic Callback"
    4. Configure the Gyroscopic Sensor once by performing the following writes:
       - STDBY Register (0x43), write 0x00.
       - Note: Please see the KXG03 datasheet for additional information on these registers
  - Begin loop()
    1. Read back the Gyroscopic output by reading 6 Bytes starting from address [0x02]. [0][1]...[5]
    2. Format Raw Gyro data

- Gyro_X_RawOUT = ([1]<<8) | ([0])
- Gyro_Y_RawOUT = ([3]<<8) | ([2])
- Gyro_Z_RawOUT = ([5]<<8) | ([4])

3. Cancel for offset value
    - Measure RawOUT data 10 times in order to proofread offset value.
    - Calculate average offset value. This data is defined as proofread data.
    - Subtract proofread data from each RawOUT data.
4. Convert Raw Returned Value to deg/sec
    - Gyro_ X = (float)KXG03_Gyro_X_RawOUT2 * 0.007813 + 0.000004
    - Gyro_ Y = (float)KXG03_Gyro_Y_RawOUT2 * 0.007813 + 0.000004
    - Gyro_Z = (float)KXG03_Gyro_Z_RawOUT2 * 0.007813 + 0.000004
5. Format Serial Output and Return Information

Hardware Connection for ROHM BM1422 Magnetometer Sensor for the Arduino Uno

- ROHM BM1422Magnetometer Sensor
  - 

- As seen in the schematic above, this device is slightly more complicated than the other devices on this shield.
  - Part Explanation:
    - U17 = BM1422GMV Magnetometer
      - 1.8V max input voltage… therefore, we can't use the 3.3V rail native to the system. U18 and U19 are added to protect the magnetometer from the 3.3V system voltage.
    - U18 = NVT2003DP Level Shifter for I2C and Interrupt lines
      - Used to reduce I2C interface and interrupt pin connection's 3.3V signals from the Arduino board to 1.8V to the sensor.
    - U19 = BU18TA2WNVX LDO Regulator
      - Used to source power to the magnetometer and the 1.8V logic level side of the level shifter U18
- When looking at this system together, we can see than ultimately, there are 5 connections
  - VDD – Connect to Arduino 3.3V output pin on power connector
  - GND – Connect to Arduino GND pin on the power connector
  - SDA/SCL – I2C Connection, connected to A4 and A5
  - INT_MAG - This pin is used monitor the interrupt pins on the BM1422GMV. This pin is not used in the example application.

Software Explanation for ROHM BM1422 Magnetometer Sensor to the Arduino Uno

- Code Segments pertaining to this sensor can be found by defining and seeing code within the "***Magnetometer***" #ifDef statements
- Pseudo-Code Explanation
    - Define Relevant Variables
    - Begin Main()
        1. Set Baud Rate for Serial Output via Nordic BT Protocols
        2. Initialize Board Level LEDs
        3. Setup Interrupt Callback for "Periodic Callback"
        4. Configure the Magnetometer Sensor once by performing the following writes:
            - 1. CNTL1 Register (0x1B), write 0xC0
            - 2. CNTL4 Register (0x5C), write 0x00
            - 3. CNTL4 Register (0x5D), write 0x00
            - 4. CNTL3 Register (0x1D), write 0x40
            - Note: Please see the BM1422GMV datasheet for additional information on these registers
            -
    - Begin loop()
        1. Read back the Magnetometer Sensor output by reading 6 Bytes starting from address 0x10. [0][1]...[5]
        2. Format Raw Magnetometer data
            - Mag_X_RawOUT = ([1]<<8) | ([0])
            - Mag_Y_RawOUT = ([3]<<8) | ([2])
            - Mag_Z_RawOUT = ([5]<<8) | ([4])
        3. Format Serial Output and Return Information