# Rev A2

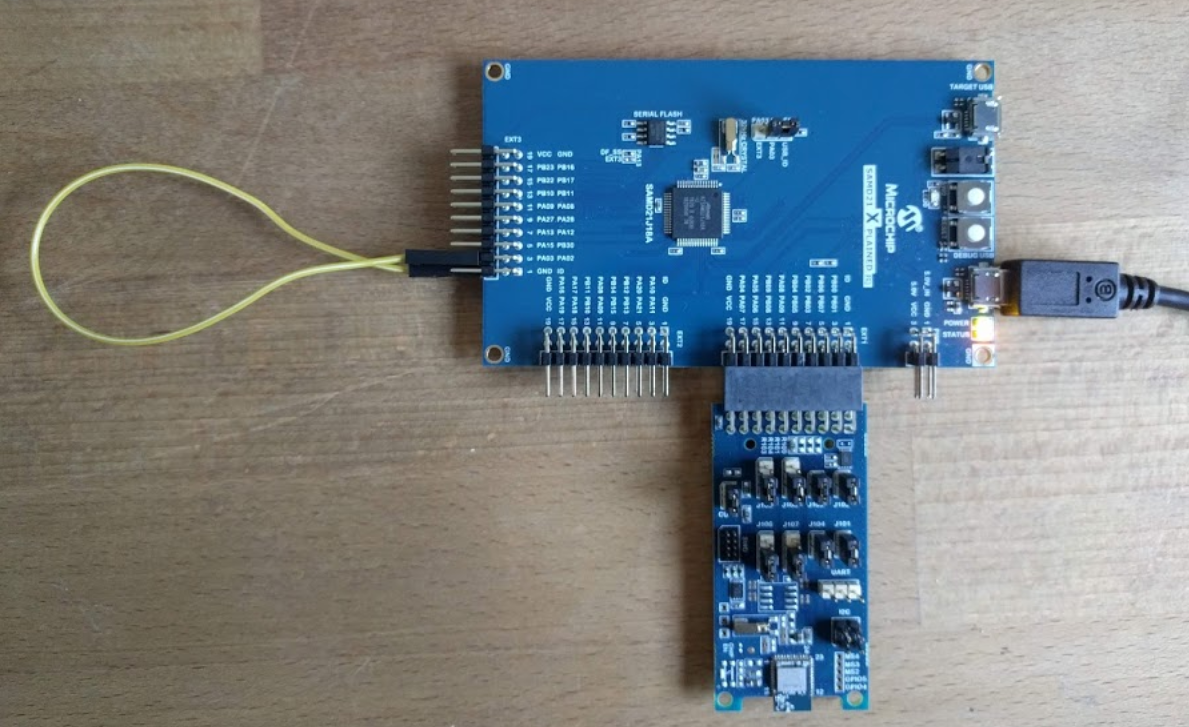
Defined two new characteristics to be able to stream data for demo app.

* 9 - “Streaming”
  + Write to any value to begin streaming when not streaming
  + Write to any value to end streaming when already streaming
  + UUID
    - F05ABAF7-3936-11E5-87A6-0002A5D5C51B
* 10 - “Stream Data”
  + When streaming is “on”, is updated continuously
  + Sends a single sensor value between 0-255 every 0.1 seconds
  + Notification is also sent when data is updated
  + UUID
    - F05ABAF8-3936-11E5-87A6-0002A5D5C51B

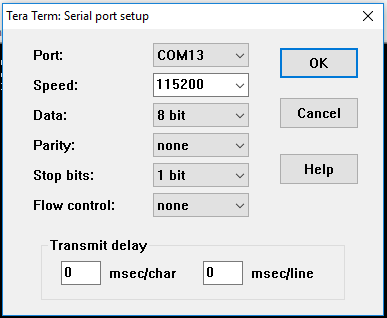
# Rev A1

## Set-up

Connect the bluetooth breakout board to the EXT1 header. Connect a microUSB cable to the DEBUG USB port on the device. Using a jumper, connect the pin PA02 to GND or VCC.



I use the program Tera Term VT to receive Debug log outputs from the firmware. Settings shown below with COM Port set to where the EDBG shows on your computer:



The device’s code has been written in AtmelStudio 7.0 in the file “startup\_template.c”

## Start-Up Procedure

On powering up the device, the BruxA Service is advertised over BLE. I have been using the app “Microchip Bluetooth Data” on Android, then click “Bluetooth Smart”, then Start Scan, and choose “BruxA” from the list. Note that if the device is reflashed, you must forget BruxA from the list of remembered devices on the phone to be able to reconnect. Passcode is 123456.

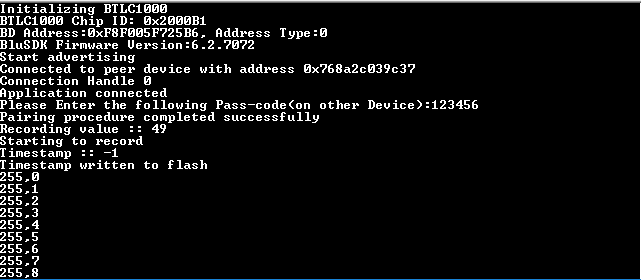
## To record sensor data

The pressure sensor is read on pin PA02 in 8-bit resolution at 10 hertz. The data is saved to a buffer in the microcontroller for 182 samples, then this buffer is written to the on board flash chip, which has 8Mbit capacity. We are still developing the sensor so for now have provided a jumper cable in order to connect this pin to either VCC or GND to ensure the value changes.

To begin recording of the sensor, write the “Record Characteristic” (#3) to hex value 0x31, ASCII of 1, or decimal 49. A callback recognizes the change and begins recording. My output is shown below where the first number (255) is the sensor reading and the second is the sample number (0). Note on the second screen the “Page Saved!” note indicates that the buffer has been written to flash memory.

### *Timestamp and Flash memory handling*

The device has no way to know what time is has begun recording sensor data so the first four bits in the flash memory are reserved to be used as a timestamp. When the start recording signal is sent, the first action is the device reads the four timestamp bits from the memory. The flash memory is then completely erased. The timestamp previously read is then rewritten to the first four bits of the flash memory, followed by the recorded sensor data. The timestamp has no way of being re-written currently and is saved as a Unix timestamp for Nov 24th, 2018 8:00:00am. The timestamp can be read from the timestamp characteristic (#2) but not written currently.





To stop recording, change this same record characteristic (#3) to any value except 0x31. The debug log will indicate “Stopping Recording”. Note that the values recorded after the last page save are not recorded in the flash memory.

## To sync data to the application

To send the recorded sensor data from the flash memory to phone application, write the Sync Characteristic (#4) to Hex 0x31, Dec 49, or ASCII 1. Upon doing so, the device will first determine the number of “pages” (182 8-bit values) that are written in the flash memory. Then the Pressure Characteristic (#1) wil1 be populated with the first page of data, which can be read.

To receive subsequent pages, write the Pressure Read Characteristic (#6) to any value. This will trigger a callback upon which the device will read the next page of data and populate that data in the Pressure Characteristic (#1). After reading this page of data, again write the Pressure Read Characteristic (#6) to any value. This process is repeated for number of pages recorded in the flash memory.

Once the last page has been written to the Pressure Characteristic (#1), the Stop Sync Characteristic is changed to Hex 0x00 to indicate the end of the recorded data has been reached.

## To read the device’s battery level

To get the current battery level of the device, set the Battery Read Characteristic (#8) to any value. The current percentage of the battery will be updated on Battery Characteristic (#7). The battery level is read on pin PA\_03.

The input voltage on this pin should vary between 2.1V (representing a fully charged (100%) 4.2V Li-Ion battery) to 1.65V (representing a fully discharged (0%) battery at 3.7V).

## Service Characteristic UUID’s & Pin Definitions

* BruxA Service UUID:
  + F05ABAC0-3936-11E5-87A6-0002A5D5C51B

1. Pressure Characteristic UUID:
   1. F05ABAD8-3936-11E5-87A6-0002A5D5C51B
2. Timestamp Characteristic UUID:
   1. F05ABAE1-3936-11E5-87A6-0002A5D5C51B
3. Record Characteristic UUID:
   1. F05ABAF1-3936-11E5-87A6-0002A5D5C51B
4. Sync Characteristic UUID:
   1. F05ABAF2-3936-11E5-87A6-0002A5D5C51B
5. Stop Sync Characteristic UUID:
   1. F05ABAF3-3936-11E5-87A6-0002A5D5C51B
6. Pressure Read Characteristic UUID:
   1. F05ABAF4-3936-11E5-87A6-0002A5D5C51B
7. Battery Characteristic UUID:
   1. F05ABAF5-3936-11E5-87A6-0002A5D5C51B
8. Battery Read Characteristic UUID:
   1. F05ABAF6-3936-11E5-87A6-0002A5D5C51B

* Pressure Sensor Pin: PA02
* Battery Level Pin: PA03

## Changes Needed

* Recording should occur following the removal of the device from the charger. Right now, data recording starts and stops once the user signals it from their phone.
  + Some thought needs to be put into what if the charger connection is intermittent and the device believes it is removed before it actually is.
* Timestamp at the beginning of recording should be set automatically based off the mobile device’s current time.
* The bluetooth module should be put to sleep to conserve battery power once data recording begins. The module should be woken back up once the device is placed on the charger.
* Sensor data should be automatically synced to the smartphone once connection is re-established after being placed on the charger.
* After paired once with the device, bluetooth pairing should occur automatically when the module begins advertising.
* Battery level of the device is currently only given when requested by the app. This data should be synced automatically when the devices are paired.