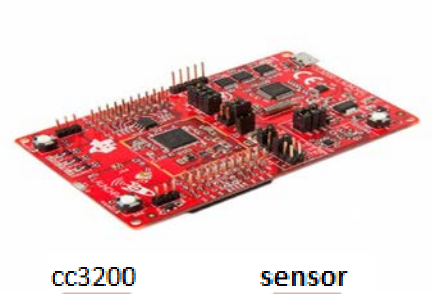
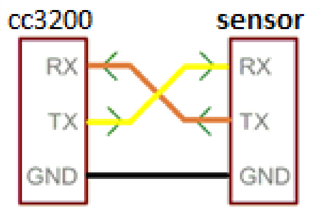
Application Note AN167:
TI CC3200 to SenseAir S8 CO2 Sensor via UART

Introduction



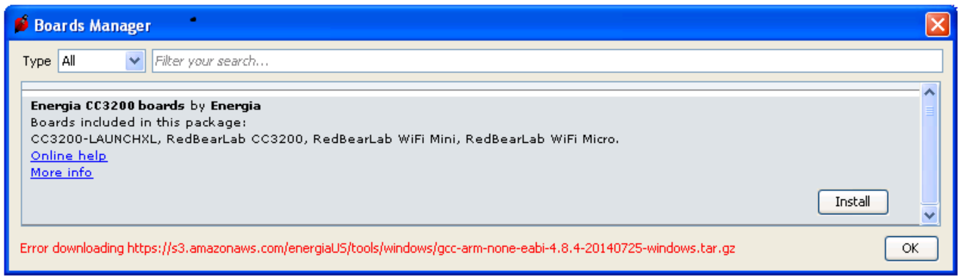
The TI CC3200 LAUNCHPADXL is a WIFI chip with an integrated MCU. This allows the device to operate independently of a CPU. This application note is used to demonstrate the capabilities of the CC3200LAUNCHXL as a powerful alternative to other IOT (Internet of Things) devices. Follow the instructions below to connect the CC3200 to the S8 sensor, the first step in displaying the sensor's CO2 readings over a WIFI network or online.



Serial communications

Setting up your CC3200 LAUNCHPADXL

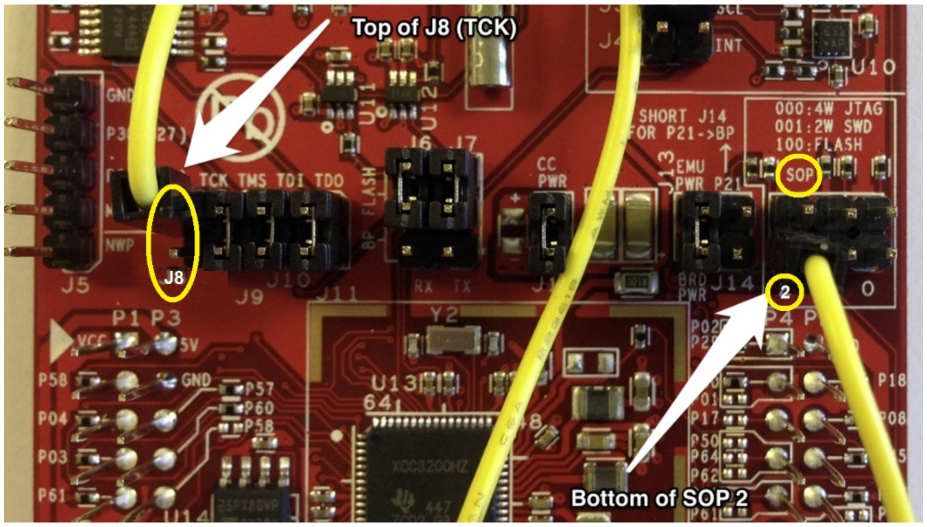
1. Create a new directory on your computer, such as C:\CC3200.
2. Download the free Energia IDE. We will be using the "Energia" programming software to flash data to and from the TI CC3200 LAUNCHPADXL. Energia is a free and open source software that can be found at http://energia.nu/download/
    a. Select the latest Windows binary release and download this to your computer.
    b. Move this file to your newly created directory (Example: C:\CC3200).
    c. **UNZIP/EXTRACT** the **ENTIRE** folder into this directory.
    d. To start the Energia application, click on **Energia.exe**.
3. **Do not connect your CC3200 LaunchPad to your PC. If you have already plugged it into your PC then unplug it before proceeding.**
4. Download the CC3200 Drivers for Windows 32 or 64 bit at http://energia.nu/files/cc3200_drivers_win.zip.
    a. Move this file to your newly created directory (Example: C:\CC3200).
    b. **UNZIP/EXTRACT** the **ENTIRE** folder into this directory.
    c. double click DPinst.exe for Windows 32bit or DPinst64.exe for Windows 64 bit.
    d. Follow the installer instructions.
5. Launch Energia.
6. Click on Tools>Board, then scroll down and select CC3200. Click install.
7. Connect your CC3200 LaunchPad to your PC. The CC3200 will be automatically recognized.



Energia CC3200 boards by **Energia**
Boards included in this package:
CC3200-LAUNCHXL, RedBearLab CC3200, RedBearLab WiFi Mini, RedBearLab WiFi Micro.
Online help
More info

```
Error downloading https://s3.amazonaws.com/energiaUS/tools/windows/gcc-arm-none-eabi-4.8.4-20140725-windows.tar.gz
```

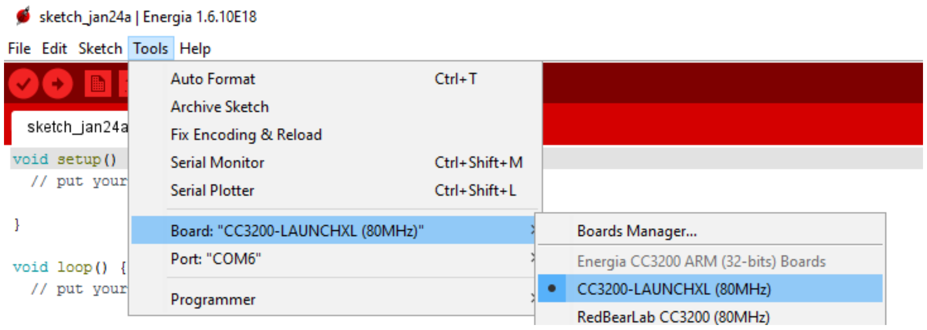
Setting up Energia to Recognize the CC3200 LAUNCHPADXL



Before you can run a program on the CC3200 LAUNCHPADXL, you must set up Energia and the CC3200 LAUNCHPADXL to communicate. This is done by removing the jumpers on **J8** and **SOP 2** and **placing a jumper wire from the top pin of J8 to the bottom pin of SOP2.** This will put the CC3200 LAUNCHPADXL into debug mode. Debug mode allows for the CC3200 LAUNCHPADXL to be programmed and have the device configuration changed.

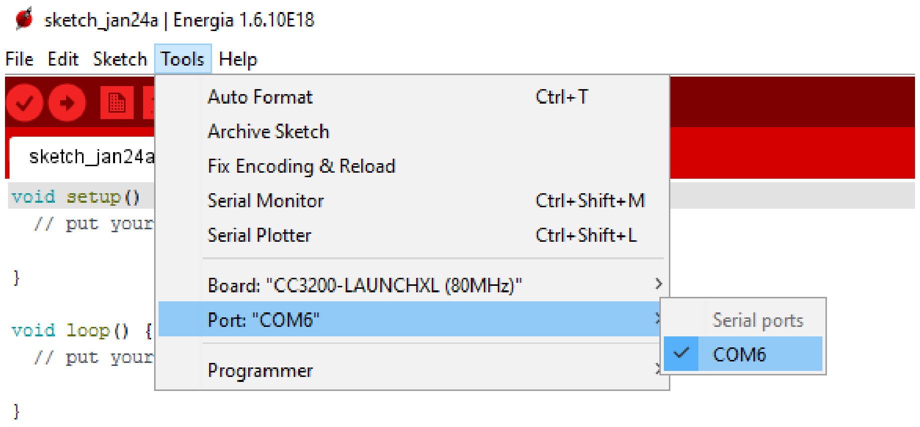
Next, we need Energia to recognize, and know which device we are trying to program. This set up process is only needed one time. After the device has been set up, it will automatically be configured.

First the board must be selected. To do this, click on the "**Tools**" menu bar, then click on "**Board**". From this submenu, select "**CC3200-LAUNCHXL (80MHz)**". This will tell Energia how to send the program over USB to the CC3200-LAUNCHXL.


sketch_jan24a | Energia 1.6.10E18
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
Board: "CC3200-LAUNCHXL (80MHz)"
Port: "COM6"
Programmer
Boards Manager...
Energia CC3200 ARM (32-bits) Boards
● CC3200-LAUNCHXL (80MHz)
RedBearLab CC3200 (80MHz)
sketch_jan24a

```
void setup()
  // put your

}

void loop() {
  // put your
```

Next, we will tell Energia which COM port to send the data through. This can be done be selecting "Tools" from the menu bar, then clicking "Port:". In the submenu, you should see a COM port that is occupied. Because a driver was installed for the CC3200 LAUNCHPADXL, this COM port will be the CC3200 LAUNCHPADXL.


sketch_jan24a | Energia 1.6.10E18
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
Board: "CC3200-LAUNCHXL (80MHz)"
Port: "COM6"
Programmer
Serial ports
✓ COM6
sketch_jan24a

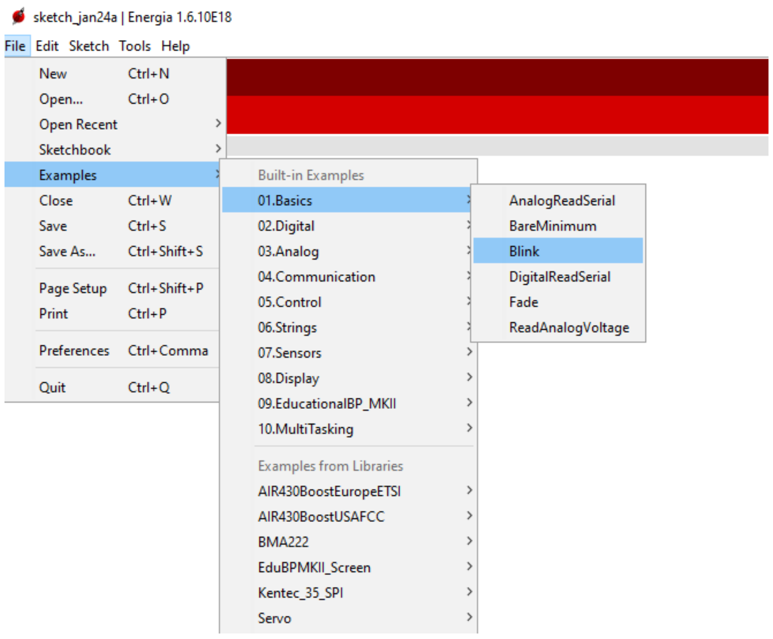
```
void setup()
  // put your

}

void loop() {
  // put your

}
```

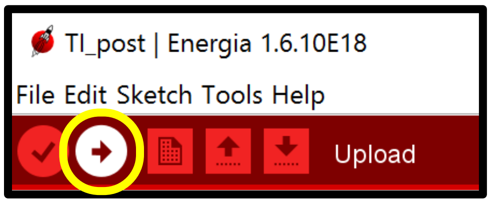
CO2Meter.com
CO2 Measurement Specialists

Running the Blink Example

If you are new to Energia and the CC3200 LAUNCHPADXL, it is highly advisable to run the Blink example. Running the example will confirm your set up of the CC3200 LAUNCHPADXL. Blink can be opened by clicking File>Examples>01.Basics>Blink. To compile and run Blink:

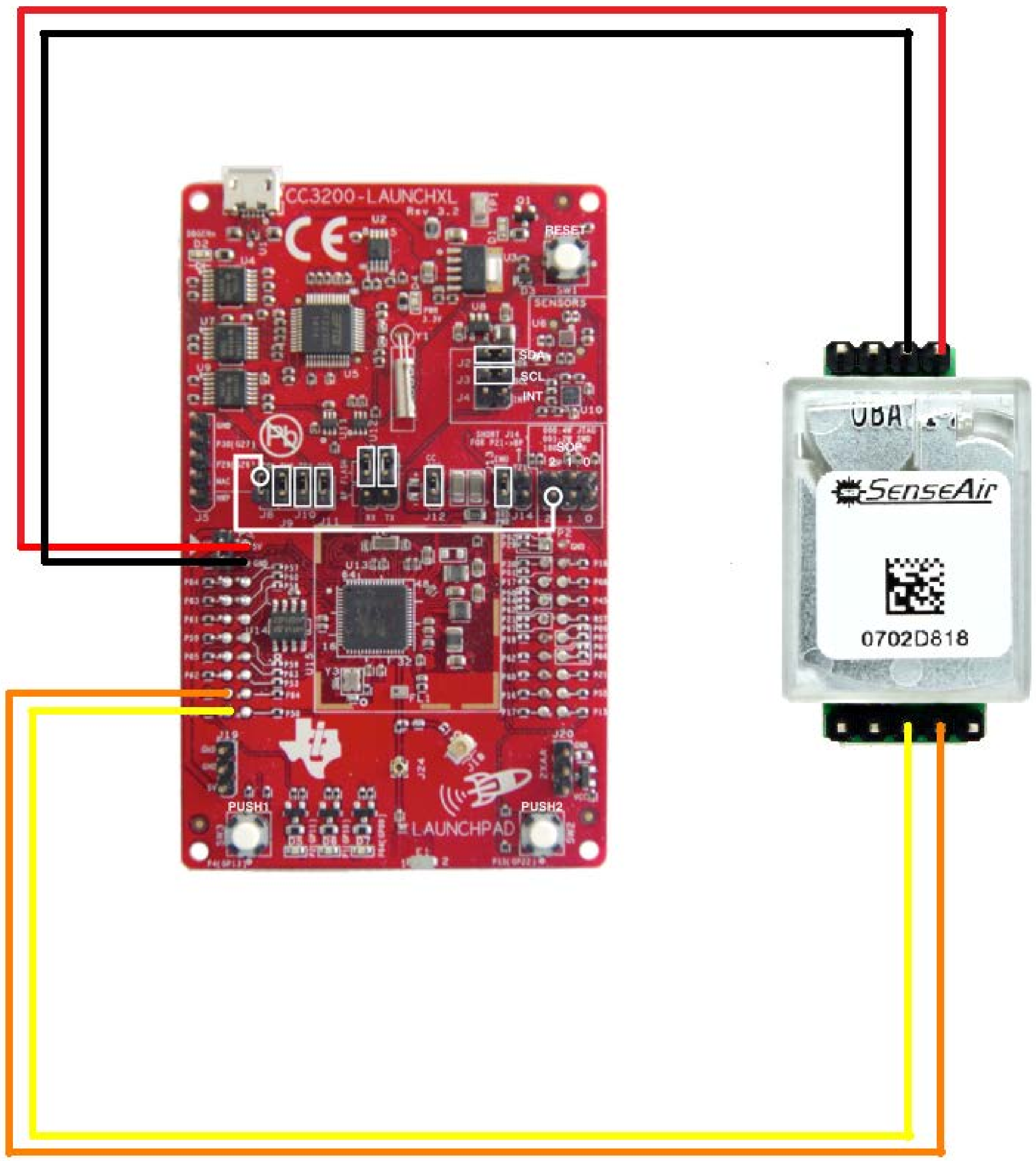
1. Select the 'Upload' arrow (Outlined in yellow). This will send the 'Blink' example to the CC3200 LAUNCHPADXL.
2. Observe the red blinking LED on the CC3200 LAUNCHPADXL.
3. If you do not see the LED blink, go back and follow the steps above.





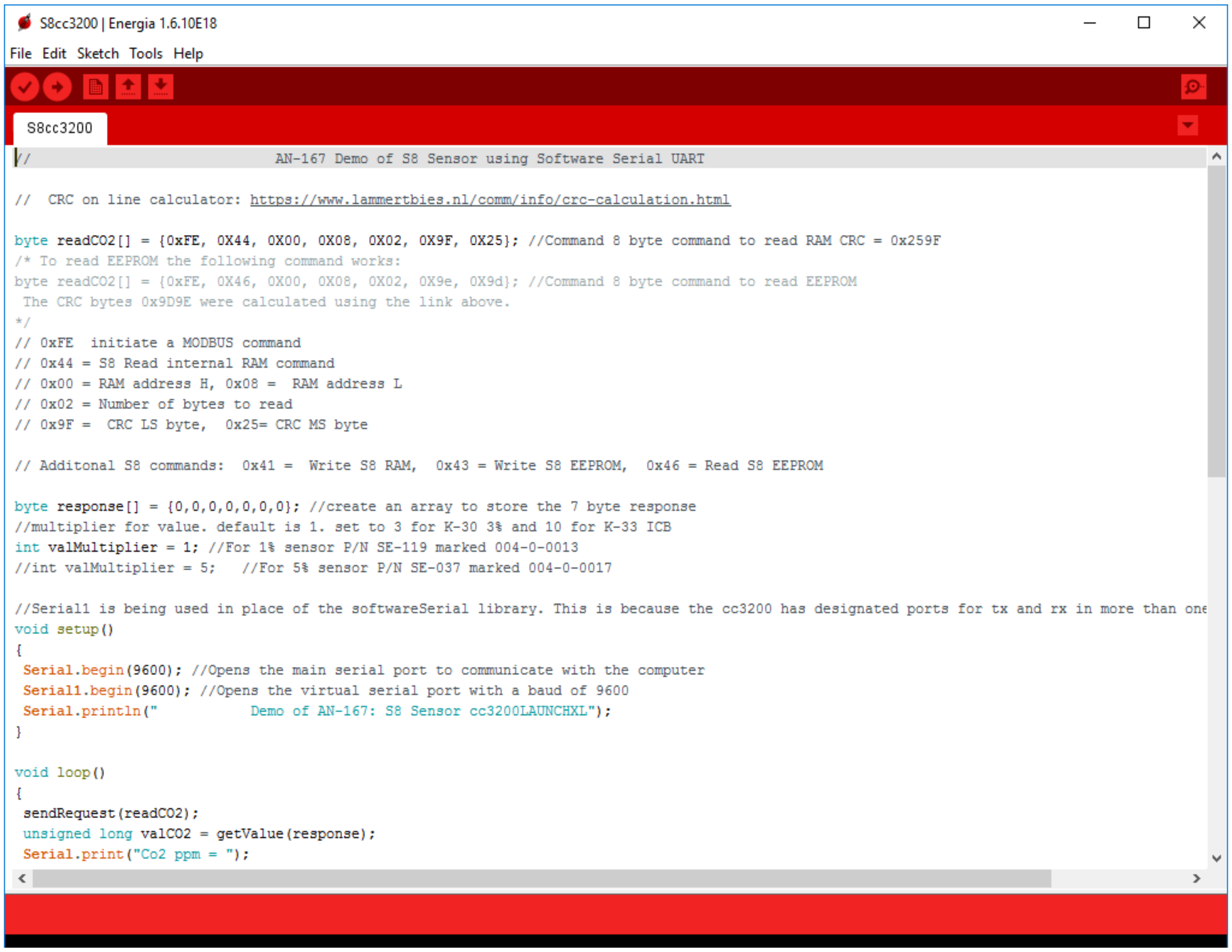
Connecting CC3200 LAUNCHPADXL to the S8 Sensor

Connect the CC3200 LAUNCHPADXL to the S8 sensor as shown. Powering the S8 can be done through the CC3200 LAUNCHPADXL as the device has a dedicated 5v output.



Creating your CC3200 LAUNCHPADXL-S8 Project

1. Launch the Energia IDE, create a new project and cut and paste the source code at the end of this document into the project window. You should see the screen below



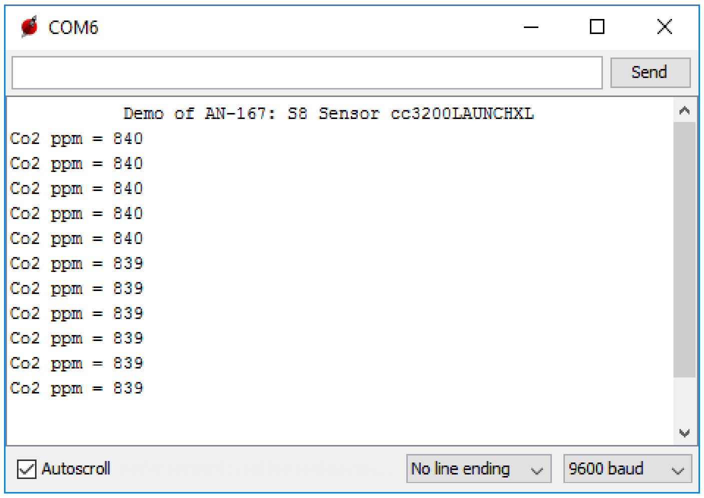
```
//                              AN-167 Demo of S8 Sensor using Software Serial UART

//  CRC on line calculator: https://www.lammertbies.nl/comm/info/crc-calculation.html

byte readCO2[] = {0xFE, 0X44, 0X00, 0X08, 0X02, 0X9F, 0X25}; //Command 8 byte command to read RAM CRC = 0x259F
/* To read EEPROM the following command works:
byte readCO2[] = {0xFE, 0X46, 0X00, 0X08, 0X02, 0X9e, 0X9d}; //Command 8 byte command to read EEPROM
 The CRC bytes 0x9D9E were calculated using the link above.
*/
// 0xFE  initiate a MODBUS command
// 0x44 = S8 Read internal RAM command
// 0x00 = RAM address H, 0x08 =  RAM address L
// 0x02 = Number of bytes to read
// 0x9F =  CRC LS byte,  0x25= CRC MS byte

// Additonal S8 commands:  0x41 =  Write S8 RAM,  0x43 = Write S8 EEPROM,  0x46 = Read S8 EEPROM

byte response[] = {0,0,0,0,0,0,0}; //create an array to store the 7 byte response
//multiplier for value. default is 1. set to 3 for K-30 3% and 10 for K-33 ICB
int valMultiplier = 1; //For 1% sensor P/N SE-119 marked 004-0-0013
//int valMultiplier = 5;   //For 5% sensor P/N SE-037 marked 004-0-0017

//Serial1 is being used in place of the softwareSerial library. This is because the cc3200 has designated ports for tx and rx in more than one
void setup()
{
 Serial.begin(9600); //Opens the main serial port to communicate with the computer
 Serial1.begin(9600); //Opens the virtual serial port with a baud of 9600
 Serial.println("           Demo of AN-167: S8 Sensor cc3200LAUNCHXL");
}

void loop()
{
 sendRequest(readCO2);
 unsigned long valCO2 = getValue(response);
 Serial.print("Co2 ppm = ");
```

To show the serial output click on Tools>Serial Monitor and have 9600 baud selected.

The C++ program has been verified to work properly. Click on Sketch>Verify/Compile.

To run the program, Sketch>Upload. The output is shown here.

Refer to the CC3200 documentation for instructions to display the data over a WIFI network.



```
           Demo of AN-167: S8 Sensor cc3200LAUNCHXL
Co2 ppm = 840
Co2 ppm = 840
Co2 ppm = 840
Co2 ppm = 840
Co2 ppm = 840
Co2 ppm = 839
Co2 ppm = 839
Co2 ppm = 839
Co2 ppm = 839
Co2 ppm = 839
Co2 ppm = 839
```

```
//  AN-167 Demo of S8 Sensor using Software Serial UART
//  CRC on line calculator: https://www.lammertbies.nl/comm/info/crc-calculation.html

byte readCO2[] = {0xFE, 0X44, 0X00, 0X08, 0X02, 0X9F, 0X25}; //Command 8 byte command to read RAM CRC = 0x259F
/* To read EEPROM the following command works:
byte readCO2[] = {0xFE, 0X46, 0X00, 0X08, 0X02, 0X9e, 0X9d}; //Command 8 byte command to read EEPROM
 The CRC bytes 0x9D9E were calculated using the link above.
*/
// 0xFE  initiate a MODBUS command
// 0x44 = S8 Read internal RAM command
// 0x00 = RAM address H, 0x08 =  RAM address L
// 0x02 = Number of bytes to read
// 0x9F =  CRC LS byte,  0x25= CRC MS byte
// Additional S8 commands:  0x41 = Write S8 RAM, 0x43 = Write S8 EEPROM, 0x46 = Read S8 EEPROM

byte response[] = {0,0,0,0,0,0,0}; //create an array to store the 7 byte response
//multiplier for value. default is 1. set to 3 for K-30 3% and 10 for K-33 ICB
int valMultiplier = 1; //For 1% sensor P/N SE-0119 marked 004-0-0013
//int valMultiplier = 5;   //For 5% sensor P/N SE-0037 marked 004-0-0017

//Serial1 is being used in place of the softwareSerial library. This is because the CC3200 has designated ports for tx
and rx in more than one pin configuration
void setup()
{
 Serial.begin(9600); //Opens the main serial port to communicate with the computer
 Serial1.begin(9600); //Opens the virtual serial port with a baud of 9600
 Serial.println("           Demo of AN-167: S8 Sensor CC3200LAUNCHXL");
}

void loop()
{
 sendRequest(readCO2);
 unsigned long valCO2 = getValue(response);
 Serial.print("Co2 ppm = ");
 Serial.println(valCO2);
 delay(2000);

}

void sendRequest(byte packet[])
{
 while(!Serial1.available()) //keep sending request until we start to get a response
 {
 Serial1.write(readCO2,7);
 delay(50);
 }

 int timeout=0; //set a timeout counter
 while(Serial1.available() < 7 ) //Wait to get a 7 byte response
 {
 timeout++;
 if(timeout > 10) //if it takes to long there was probably an error
 {
 while(Serial1.available()) //flush whatever we have
 Serial1.read();

 break; //exit and try again
 }
 delay(50);
 }

 for (int i=0; i < 7; i++)
 {
 response[i] = Serial1.read();
 }
}
unsigned long getValue(byte packet[])
{
 int high = packet[3]; //high byte for value is 4th byte in packet in the packet
 int low = packet[4]; //low byte for value is 5th byte in the packet


 unsigned long val = high*256 + low; //Combine high byte and low byte with this formula to get value
 return val* valMultiplier;
}
```