

**INTERNET OF THINGS USING
TEXAS INSTRUMENTS
MSP430F5529 LAUNCHPAD AND
CC3100 WIFI NETWORK PROCESSOR**

Pre-requisites

To follow this lab manual, the following pre-requisites are required:

Hardware: MSP430F5529 LaunchPad + CC3100 BoosterPack

Software: Energia 1.8.7E21 or higher(latest)

Operating Systems: Windows 8,10(Recommended)

List of Experiments

1. Setting up of Energia Software
2. A simple GPIO configuration example- LED and Switch interface
3. Using the 12-bit ADC- configuration of LM35 temperature sensor
4. Configuration of IoT Node as an Access Point
5. Establish connection between IoT Node and Access point and Display of local IP and Gateway IP
6. Network Communication- Establish one way communication between client and server using 2 IoT Nodes
7. Network Communication- Establish 2-way communication between client and server using 2 IoT Nodes
8. Design and Implementation of HTTP based IoT Web Server to control the status of LED
9. Design and Implementation of HTTP based IoT Web Server to display sensor value

1. Setting up of Energia Software

Energia is an open-source IDE, very similar to Arduino software and is used to design applications using the TI family of microcontrollers. Energia can be downloaded from the link below:

<https://energia.nu/downloads/downloadv4.php?file=energia-1.8.7E21-windows.zip>

Energia is also supported on operating systems such as Linux and MacOS, however, in this manual we are going to use Windows 10.

If you are interested in learning more about Energia, please consider going through the links below:

1. Guide: <https://energia.nu/guide/>
2. Pin-maps: <https://energia.nu/pinmaps/>
3. Reference: <https://energia.nu/reference/>

2. Getting Started with Energia

- a. Once you have downloaded Energia, you will have a zip folder(compressed). This has to be extracted to a working folder. For example, your working folder is-

C:\EnergiaLabs

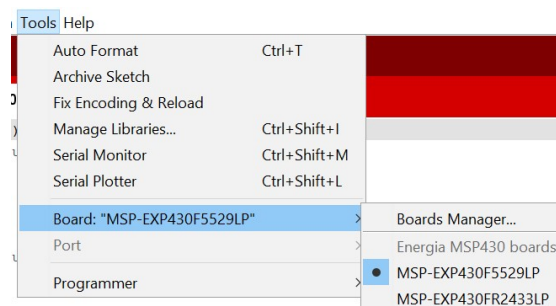
- b. Now, extract the downloaded zip folder to the working folder, after this you should have a directory as below:

C:\EnergiaLabs\energia-1.8.7E21

If you go to the directory above, and open the energia-1.8.7E21 folder, you will get the executable file, there is no installation required, you can double click on the executable to launch Energia.

| | | | | |
|---|-----------------|------------------|------------------------|----------|
|  | arduino-builder | 11-10-2018 15:26 | Application | 7,926 KB |
|  | energia | 11-10-2018 15:44 | Application | 143 KB |
|  | energia.l4j | 11-10-2018 15:27 | Configuration settings | 1 KB |
|  | energia_debug | 11-10-2018 15:44 | Application | 141 KB |

- c. Launch Energia, if you see a pop-up alert from Windows Security Alert, click on Allow access.
- d. In Energia, go to Tools → Board → MSP-EXP430F5529LP



- e. In this step, we shall install the required drivers for the MSP430 family of microcontrollers from the link below:

http://s3.amazonaws.com/energiaUS/files/energia_drivers.zip

In the download folder, you should get something like this



Extract the zip folder and install the drivers as per your operating system 32 or 64 bit, windows 10 users can select 64-bit option.



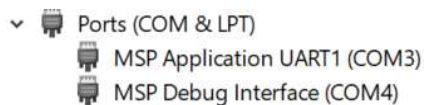
Install Energia Drivers for Windows 32 bit

Install Energia Drivers for Windows 64 bit

- f. For Windows 8 and 10 users, you may need to disable your signed driver feature, follow this guide:

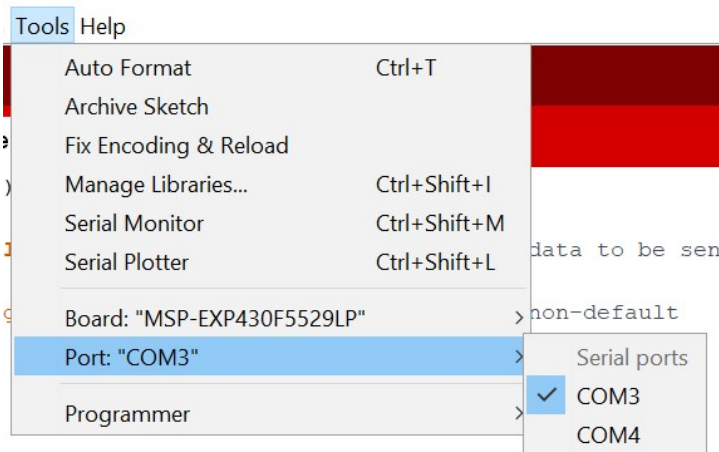
<https://learn.sparkfun.com/tutorials/disabling-driver-signature-on-windows-8>

- g. Having installed the drivers, you are now ready to go ahead and connect your MSP430 LaunchPad to your PC. To check if the MSP430 board is correctly identified by your PC, go to Device Manager on your PC and check the following:



Note: The COM number may be different in your case!

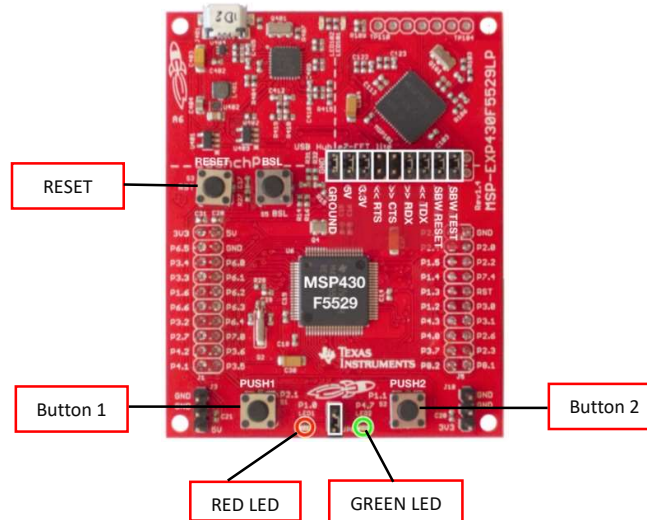
- h. In Energia, go to Tools → Port → Select the COM port as per your Device Manager info.



2. A simple GPIO configuration example- LED and Switch interface (Push Buttons)

Experiment Overview:

The MSP430F5529 microcontroller is a 16-bit MCU with its CPU running at a maximum clock frequency of 16 MHz. The MSP430 interacts with the external devices using the input and output pins. These pins are known as General Purpose Input and Output (GPIO). The goal of this lab is to understand how the GPIO pins can be configured to perform simple tasks such as toggle the on-board LEDs and read the status of push buttons.



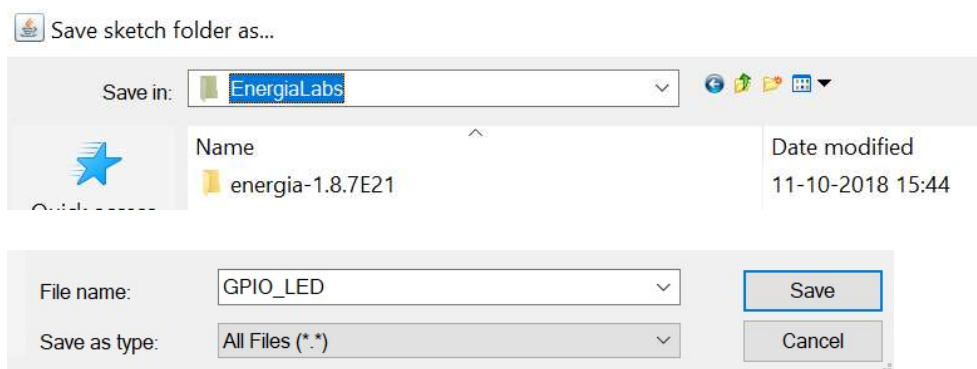
Procedure: Part 1- Toggle the Led

1. Launch Energia
2. Each project is known as a Sketch in Energia. It looks as below:

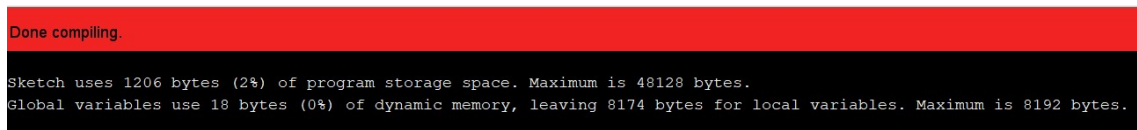
```
File Edit Sketch Tools Help
[Icons]
sketch_may09a
void setup() {
  // put your setup code here, to run once:
}

void loop() {
  // put your main code here, to run repeatedly:
}
```

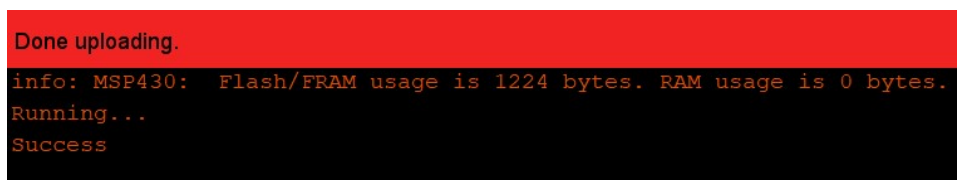
3. First, let us save this sketch as GPIO_LED. Go to File → Save As → Give File Name as GPIO_LED → Save in your working directory (E.g., C:/EnergiaLabs). Follow the screenshot below:



4. Clear the content of this sketch. CTRL+A and Delete
Copy and Paste the code for Experiment No.2(Blink_LED) from the Lab Manual Folder.
5. The next step is to compile or verify the sketch. Click on the verify icon. The code should compile without any errors.



6. Next, click on the upload icon to program your MSP430 LaunchPad with the compiled binary file. Once uploaded to the Flash memory, you should see the Red Led blinking at a rate of 1 sec. If not, press the Reset button on the LaunchPad.



7. Assignment- Try to modify the rate from 1 sec to 0.5 sec. Also, modify the code to toggle the Green Led. In the next part of this lab, we will use Push Button 1 to change the state of the Red Led.

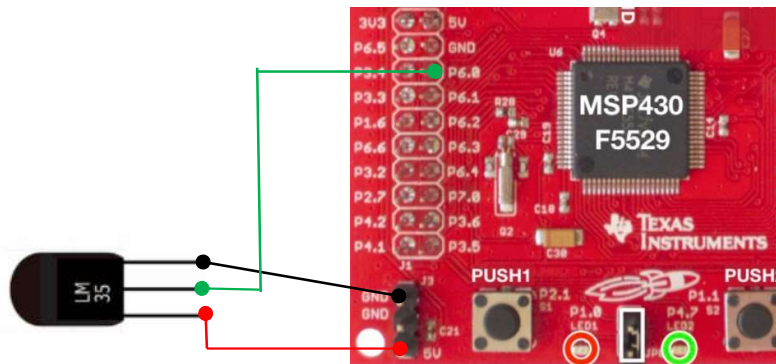
Part 2- Push Button

1. Open a new sketch on Energia. You can close the GPIO_LED sketch. Go to File→ New. This should launch a new sketch. Save this new sketch as Push_Button in your working directory(workspace).
2. Clear the content of this sketch. Copy and Paste the code from Experiment No.2 (Push_Button) in the Lab Manual Folder.
3. Verify and Upload the Sketch. Press the switch/button located at the bottom left of the MSP430 LaunchPad. You can also refer to the experiment overview diagram as described previously. Pressing the button should toggle the state of the Red Led.
4. Assignment- Try to modify the code to change the state of Green Led using Push Button 2.

3. Using the 12-bit ADC- configuration of LM35 temperature sensor

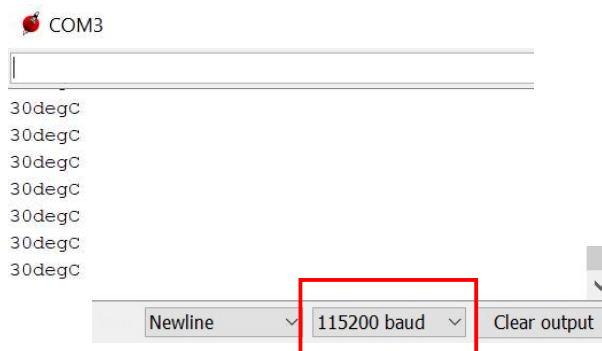
Experiment Overview:

The goal of this experiment is to understand the configuration of 12-bit Analog to Digital Converter on the MSP430 LaunchPad. ADC is widely used to interface analog sensing elements. There are 10 Analog channels that can be used as input pins to sensors on this LaunchPad. Based on the application requirements, you can use single or multiple channels at the same time. ADC supports simultaneous sampling of multiple channels. In this lab, we shall use LM35 temperature sensor connected to analog input channel number 0 (A0). The temperature sensor values will be displayed on the Serial monitor.



Procedure:

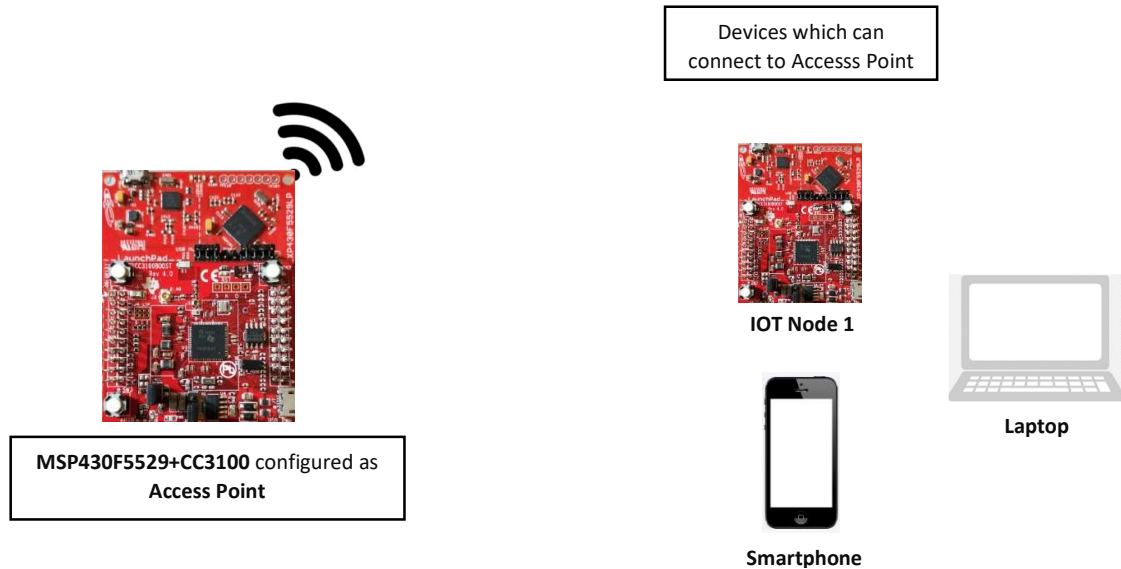
1. Make the connections as per the figure shown above.
2. Create a new sketch in Energia. Save the new sketch in your working directory as Analog_Input.
3. Clear the content of the sketch. Copy and Paste the code from Experiment No. 3 (TemperatureSensor) to your current sketch.
4. Verify and Upload your Sketch.
5. In Energia, Go to Tools → Port → Make sure the correct COM port is selected as per the info on Device Manager.
6. Go to Tools → Serial Monitor → Set the Baud rate at 115200. Reset the LaunchPad. The temperature sensor values should be displayed on the Serial Monitor.



4. Configuration of IoT Node as an Access Point

Experiment Overview:

An IoT Node is typically designed using a microcontroller connected to a wide variety of sensors and actuators. IoT Nodes can communicate with other devices using Internet as a medium. In this lab, we shall use MSP430 LaunchPad with CC3100 BoosterPack. The BoosterPack features an ARM based WIFI Network Processor providing MSP430 the power to connect to the internet securely. In Wireless Local Area Networks (WLANs), the Access Point acts as a portal to establish links with end devices using a WIFI. Multiple end devices can communicate among each other via an Access Point. The goal of this lab is to understand the configuration of MSP430 as an IoT Node and an overview of the Energia WIFI Library.



Procedure:

1. First let us understand the correct setup of CC3100 BoosterPack. The BoosterPack is just a network processor dedicated towards handling the TCP/IP and UDP communications over the network. The BoosterPack is used with a Host Microcontroller, MSP430 in our case. The BoosterPack derives its power either from MSP430 or externally from USB via a cable. Its very important to correctly set up the CC3100 BoosterPack before using it. The following figure will help.

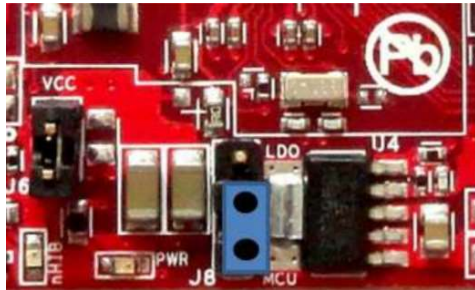
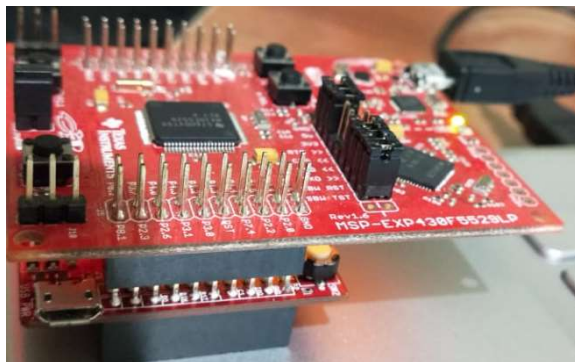


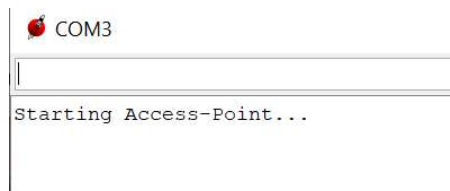
Fig: Power from MCU (to be used in this mode for all the labs)

2. The BoosterPack can be connected to the MSP430 in two ways- either CC3100 on top or at the bottom. For all the labs, we shall connect CC3100 at the bottom, because this allows you to easily use the push buttons on the MSP430 and also identify the pins correctly on MSP430. Please refer the figure below.



Note: Be very careful while connecting both the boards, make sure the pins are not bent, also the connection has to be made only after disconnecting the LaunchPad from the PC.

3. Create a new sketch in Energia and clear its content. Save this new sketch in your working directory as IoT_AP. Now, browse to the Lab Manual folder. Copy and Paste the code from Experiment No. 4 (Access_Point) to this new sketch. Verify and Upload the Sketch.
4. Open the Serial Monitor. Make sure that the correct COM port is selected and the baud rate is set to 115200. Reset the LaunchPad. After a short while, you can see the following message on the Serial Monitor.



5. The IoT Node has been successfully configured as an Access Point. This means that a Wireless Local Area Network is now available with the following credentials.

Network Name(SSID)- IoT

Password- Energia1234

If you check the code, these are the credentials that you have set in your code.

```
#include <SPI.h>
#endif
#include <WiFi.h>

char ssid[]="IoT";
char password[]="Energia1234";

void setup()
{
    Serial.begin(115200);

    Serial.print("Starting Access-Point...");
    WiFi.beginNetwork(ssid,password);

}
```

6. Now to test the Access Point, you can either use your laptop or a smartphone. Scan the available wifi networks. There should be one by the name "IoT". Select this network and connect to it. When asked for password, provide the password as- Energia1234.
7. Assignment- Create a WLAN Access Point with the following credentials:
Network Name(SSID)- StarTrek
Password- Jupiterabc123

5. Establish connection between IoT Node and Access point and Display of local IP and Gateway IP

Experiment Overview:

The goal of this experiment is to understand the configuration of IoT Node to connect to an Access Point. Any device when connected to a network is assigned an IP address, which is a unique number for each device on a network. We shall also see how to identify the IP network of our IoT Node. The Access Point also has an IP address, known as the Gateway IP. These concepts are fundamentals to understand the architecture of an IoT Network.

Hardware Required:

MSP430F5529 LaunchPad + CC3100 BoosterPack



Smartphone generates the WIFI network,
acts like an **Access Point** and is assigned a
Gateway IP



MSP430F5529+CC3100 acts as **IOT Node**,
connects to a WIFI network and is assigned a
Local IP Address

Procedure:

1. Repeat steps 1 and 2 from the previous experiment. After this, your hardware setup is configured for the experiment.
2. Create a new sketch in Energia and clear its content. Save this new sketch in your working directory as IoT_LocalIP. Now, browse to the Lab Manual folder. Copy and Paste the code from Experiment No. 5 (Local_IP) to this new sketch.
3. Next, we shall create a WIFI network for our IoT Node to connect to. Using your smartphone, create a hotspot, give a meaningful name to your hotspot. For example, using my phone I have generated a hotspot with the following credentials:

Hotspot Name: IOTLAB

Password: abcdef123

Now your smartphone will act as an Access Point to which the IoT Nodes can get connected to.

4. In the code, make the following changes:

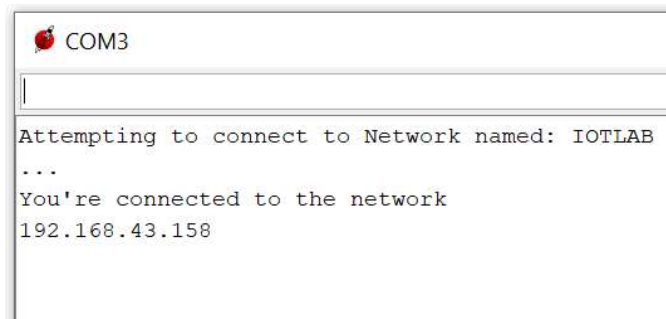
```
#include <SPI.h>
#endif
#include <WiFi.h>
```

```
char ssid[]="IOTLAB";
char password[]="abcdef123";
```

← The SSID is the name of the Hotspot generated by your phone

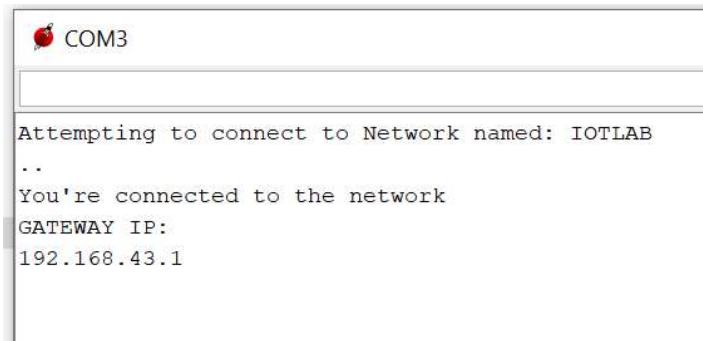
```
IPAddress ip;
```

5. Verify and Upload your code. Open the Serial Monitor with the correct COM port and Baud Rate set to 115200.
6. Reset the LaunchPad and you will see some information displayed on the Serial Monitor as follows:



7. The IoT Node successfully connects to the Access Point created by your smartphone and is also assigned an IP address. Here in my case, it is 192.168.43.158. You can also verify the same on your smartphone, which will show that 1 device is connected.

8. Troubleshoot: Check if the MSP430 and CC3100 are connected properly. Also check the ssid and password in your code are set as per your smartphone. Check if the COM port is correct and Baud rate is set to 115200, and try resetting the LaunchPad.
9. Gateway IP: For this part of the experiment, you can create a new Sketch and save it as IOT_GatewayIP in your working folder. To find the code go to IOT Manual folder → Experiment No.5(Gateway_IP). All the other steps are same as that of the Local_IP experiment.
10. Verify and Upload the Sketch. Open the Serial Monitor. Now you should see message on the Serial Monitor similar to below:



The screenshot shows the Serial Monitor window for COM3. The text displayed is as follows:

```
Attempting to connect to Network named: IOTLAB
..
You're connected to the network
GATEWAY IP:
192.168.43.1
```

11. The Gateway IP is displayed as 192.168.43.1

6. Network Communication- Establish one way communication between client and server using 2 IoT Nodes

Experiment Overview:

Devices communicate with each other over a network using a Client Server paradigm. The Server facilitates a service whereas the Client requests the Server for a service. A communication is initiated when the Client pings the Server using the Server IP address. The Server is always in Listen mode and is ready to respond to any Client trying to connect with the Server. On successful connection, the Server sends data to the Client. The goal of this experiment is to understand the Client Server communication model in an IoT Network using Real Time IoT hardware.

Hardware Required:

2 MSP430F5529 LaunchPads + 2 CC3100 BoosterPack



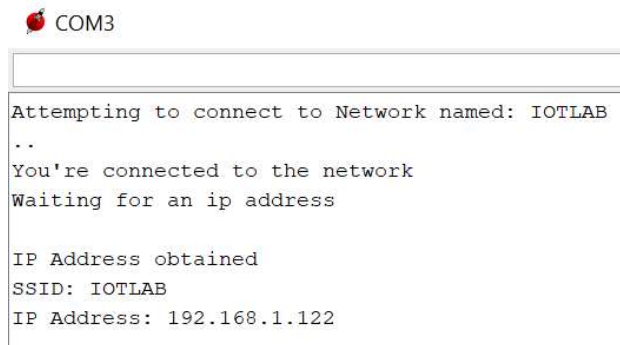
Procedure:

1. Prepare your hardware as per the steps from the previous experiments. In this lab we would require 2 IoT Nodes. One each for the Server and Client side.
2. Let's first configure the Server side. Create a new Sketch, clear its contents. Save it as IOT_Server in your working directory. From the IOT Manual folder, find the code for Experiment No. 6 (Server_Static). Copy and Paste this code into your Sketch and save it.
Note: There are two codes for configuring your IoT Node as Server (Server_Dynamic & Server_Static). Server_Dynamic uses the dynamic IP allocation to the IoT Node i.e., every time you power on the Server, a random IP address will be assigned. However, in the Server_Static, the user can specify the IP address in the code itself and the same is assigned to the IoT Node. Since the Client should know the IP address of the Server, it is better to use the static IP address because it does not change.
3. Create a hotspot or WIFI network (using smartphone) to which your Server and Client would get connected to. Modify your Server code with the correct credentials:

```
// your network name also called SSID
char ssid[] = "your WIFI network name";
// your network password
char password[] ="WIFI password";
```

4. The Server always uses a Port to communicate with the incoming Clients. Define the port number as shown:

```
WiFiServer server(9999);
```
5. Verify and Upload the Sketch. Make sure the correct COM Port is selected and Baud Rate is set to 115200. Reset the LaunchPad. Find the following message on the Serial Monitor.



```
COM3
Attempting to connect to Network named: IOTLAB
..
You're connected to the network
Waiting for an ip address

IP Address obtained
SSID: IOTLAB
IP Address: 192.168.1.122
```

6. The Server is configured and is running. Notice that the same IP address is assigned to the Server as written in the code.
7. Next, we need to configure another IoT Node as Client. On another PC, use another board to be configured as Client. Create a new Sketch, Save it as IOT_Client in your working directory. Copy and Paste the code from Experiment No. 6 (Client) to this Sketch. Modify the network credentials (SSID & Password) in your code and save it. Modify additional parameters as shown below:

```
uint16_t port =9999;      // port number of the server
IPAddress server(192,168,1,122);  // IP Address of the server
WiFiClient client;
```

The port number should be the same as in the Server code and IP address of Server should also be the same as assigned to the Server.

8. Verify and Upload the Sketch. Open the Serial Monitor, set the baud rate to 115200. On the other side, connect the Server to another PC and check if the Server is powered on. Reset the Client LaunchPad, the client will connect to the Server and display a “Hello Client” message coming from the Server.

```
WiFi Connected
Attempting to connect to Server
Waiting for Wi-Fi Connection
.Waiting for Wi-Fi Connection
.WiFi Connected
Attempting to connect to Server
Connected to Server
HELLO CLIENT
HELLO CLIENT
HELLO CLIENT
```

9. Assignment: Try to change the “port” number and “IPAddress server” in Server and Client code. Also, you can try sending some other message from the Server to the Client.

Note: Both the Server and Client devices should be connected to the same WIFI network.

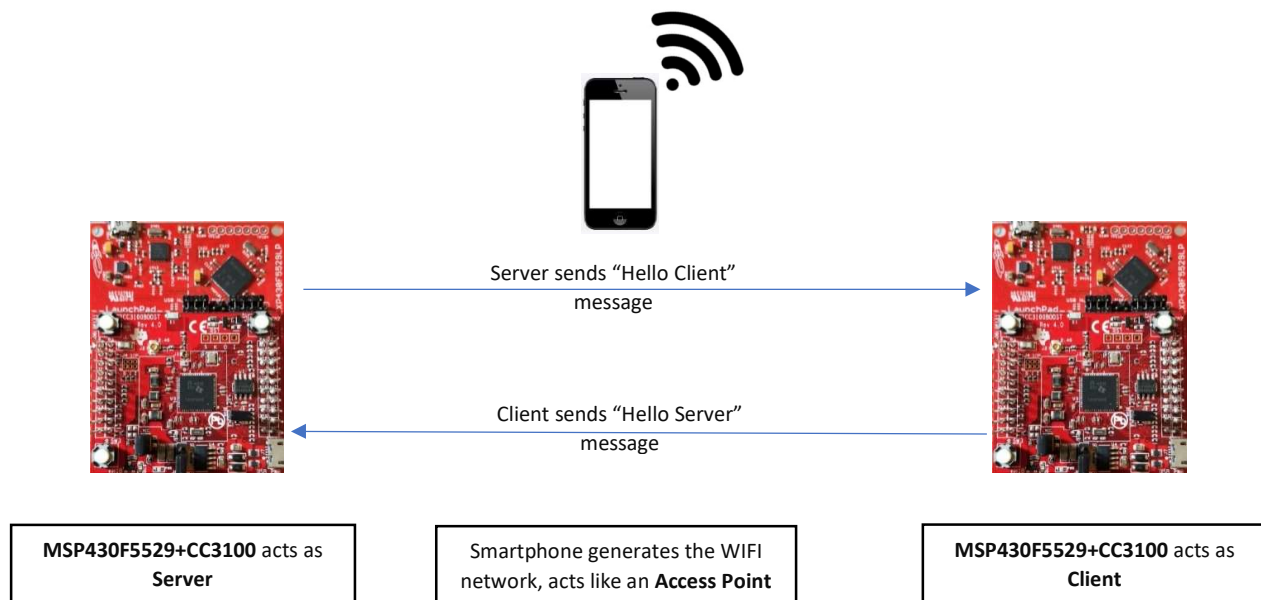
7. Network Communication: Establish 2-way communication between client and server using 2 IoT Nodes

Experiment Overview:

In the previous experiment, only the Server transmitted the message and Client received the message. The goal of this experiment is to configure both the Server and Client to transmit and receive messages. The push buttons will be used to send the messages on each side. The messages received either by the Client or the Server will be displayed on the Serial Monitor.

Hardware Required:

2 MSP430F5529 LaunchPads + 2 CC3100 BoosterPack

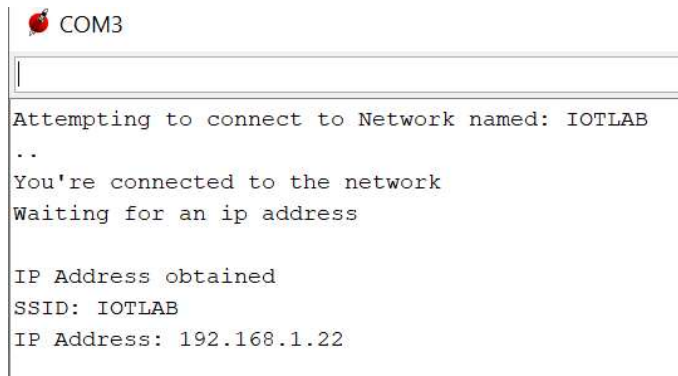


Procedure:

1. The overall procedure is same as the previous experiment. Only the code changes. Make sure that you have performed the previous experiment properly.
2. Let's configure the Server. Create a new Sketch, clear its contents and Save it as Server_2way. Copy and Paste the code from Experiment No.7 (Server).
3. Observe the IP address of the Server and the Port number defined in the Server code.
4. Create a hotspot or WIFI network (using smartphone) to which your Server and Client would get connected to. Modify your Server code with the correct credentials:

```
// your network name also called SSID
char ssid[] = "your WIFI network name";
// your network password
char password[] ="WIFI password";
```

5. Verify and Upload the Sketch. Select the correct COM port and baud rate set to 115200. Reset the Server LaunchPad.



The screenshot shows a serial monitor window with a red apple icon and the label 'COM3'. The text displayed is as follows:

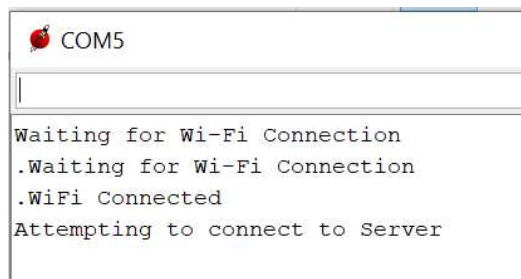
```
Attempting to connect to Network named: IOTLAB
..
You're connected to the network
Waiting for an ip address

IP Address obtained
SSID: IOTLAB
IP Address: 192.168.1.22
```

6. On another PC (with Energia Software), connect the other LaunchPad and Configure this as a Client. Create a new Sketch, clear its contents and Save it as Client_2way. Copy and Paste the code from Experiment No.7 (Client).
7. In the Client code, make sure that the IP address of the Server and the Port number are the same as defined in the Server code.
8. Modify your Client code with the correct credentials:

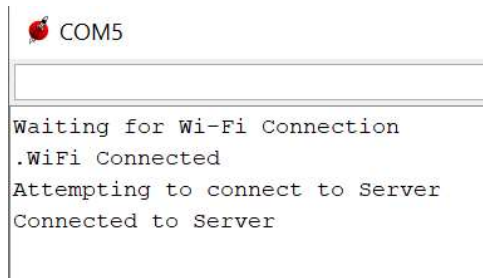
```
// your network name also called SSID
char ssid[] = "your WIFI network name";
// your network password
char password[] ="WIFI password";
```

9. Verify and Upload the Sketch. Select the correct COM port and baud rate set to 115200. Reset the Client LaunchPad. Also, Reset the Server LaunchPad. You will get a message, saying that "Attempting to connect to Server".



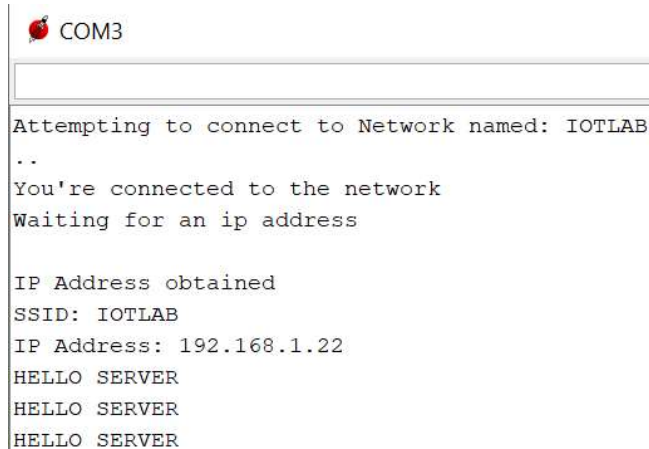
```
COM5
Waiting for Wi-Fi Connection
.Waiting for Wi-Fi Connection
.WiFi Connected
Attempting to connect to Server
```

After successful connection with the Server LaunchPad, the Client will display the message, saying "Connected to Server". Now, the system is ready for 2 way communication.



```
COM5
Waiting for Wi-Fi Connection
.WiFi Connected
Attempting to connect to Server
Connected to Server
```

10. Press the **PUSH1** button on the **Client Side** and you should see the following message on the Serial Monitor at the Server side.



```
COM3
Attempting to connect to Network named: IOTLAB
..
You're connected to the network
Waiting for an ip address

IP Address obtained
SSID: IOTLAB
IP Address: 192.168.1.22
HELLO SERVER
HELLO SERVER
HELLO SERVER
```

11. Press the **PUSH1** button on the **Server side** and you should see the following message on the Serial Monitor at the Client side.

 COM5

```
Waiting for Wi-Fi Connection
.WiFi Connected
Attempting to connect to Server
Connected to Server
HELLO Client
HELLO Client
HELLO Client
```

Note: Both the Server and Client devices should be connected to the same WIFI network.

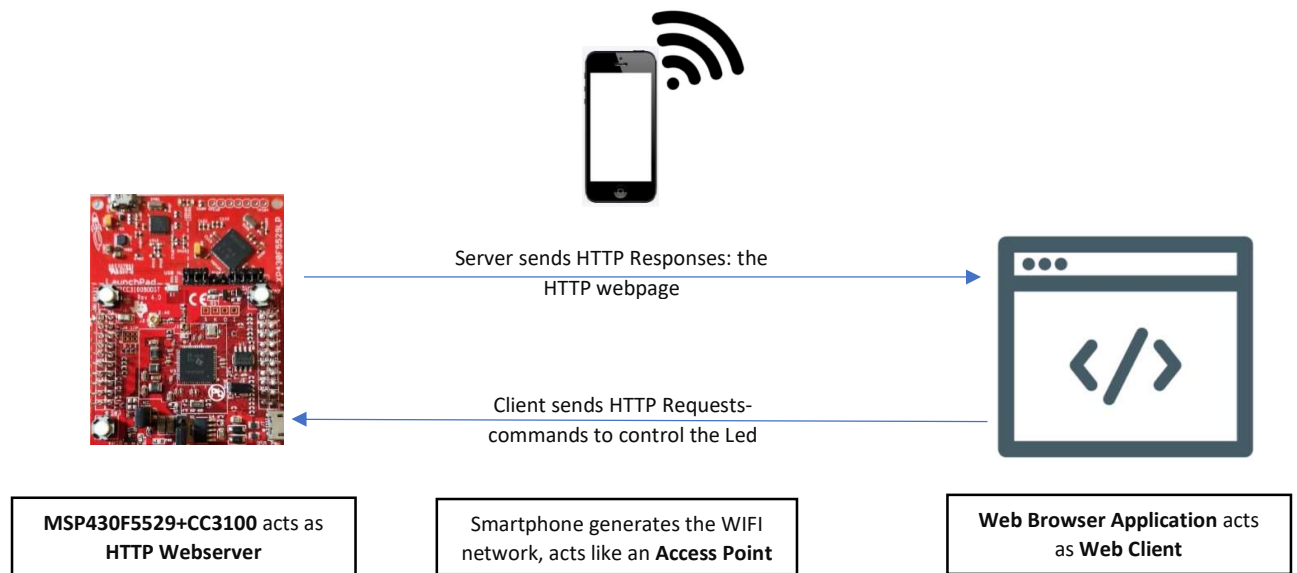
8. Design and Implementation of HTTP based IoT Web Server to control the status of LED

Experiment Overview:

An HTTP based webserver provides access to data to an HTTP client such as web browser. In IoT applications, the IoT Nodes are connected to sensors and actuators. The sensor data can be accessed using a web browser and also the actuators can be controlled from the web browser. This facilitates a wide variety of applications in IoT domain. The underlying protocol used for this communication between a Client and a Server is HTTP. The goal of this lab is to understand the configuration and working principle of an IoT web server. Using a browser, such as google chrome, we will send some HTTP based commands to the web server. Based on the type of command, the IoT Node web server will toggle the LEDs.

Hardware Required:

1 MSP430F5529 LaunchPad + 1 CC3100 BoosterPack



Procedure:


1. First, let's configure the IoT Node as a web server. The concept is very similar to the client server lab that you have performed earlier. The only major difference is that this lab uses HTTP protocol.
2. Create a new Energia Sketch and clear its contents. Save this Sketch as IOT_webserver in your working directory. Now, copy the code from the IOT Manual folder, Experiment No. 8- HTTP_webserver and paste it to your Sketch. Save it.
3. Create a WIFI network/ Hotspot using your smartphone. Modify the following part of code with the correct credentials.

```
// your network name also called SSID
char ssid[] = "your Wifi Network name";
// your network password
char password[] = "Wifi Password";
```

4. In the code, observe that the port number is 80. HTTP based servers always use port 80 for communications with the clients. This example also uses dynamic IP allocation.

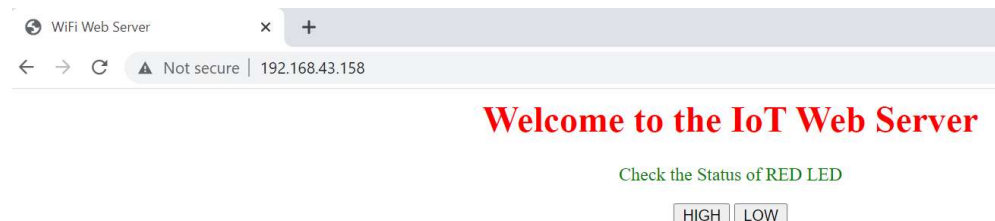
```
WiFiServer server(80);
```

5. Connect the IoT Node (LaunchPad) to your PC. Verify and Upload the Sketch. Open the Serial Monitor, select the correct COM Port and baud rate set to 115200. Reset the LaunchPad

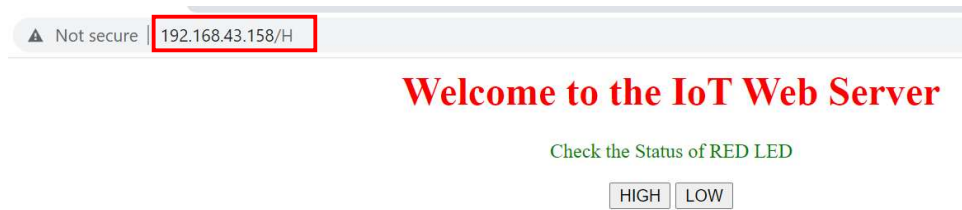
 COM3

```
Attempting to connect to Network named: IOTLAB
..
You're connected to the network
Waiting for an ip address
..
IP Address obtained
SSID: IOTLAB
IP Address: 192.168.43.158
signal strength (RSSI):-53 dBm
To see this page in action, open a browser to http://192.168.43.158
Starting webserver on port 80
Webserver started!
```

6. The IoT Node has been configured as a web server. This IoT webserver can be communicated using its IP address (in our case its- 192.168.43.158)
7. Now let's use a PC/ smartphone/ laptop connected to the same WIFI network used by the IoT webserver. This example works on local networks only. Open google chrome web-browser and type the IP address of the webserver. Hit Enter.



8. If the connection does not happen, try resetting the LaunchPad and also refreshing the chrome browser page. As you see, there are two buttons on the page to modify the status of the on-board Red Led. Click on **HIGH button** to see the Red Led turn **ON** on the LaunchPad. Repeat the same on **LOW** button and notice the changes.
9. When you click on the HIGH or LOW button, the client (web browser) sends “GET/H” or “GET/L” message to the IoT Node. Notice it on the browser.



10. Assignment: Make changes in the code to control the Green Led. Also, based on your understanding from previous labs, modify this example for Static IP Address. The IoT Node should be assigned the same IP address as mentioned in the code by the user.

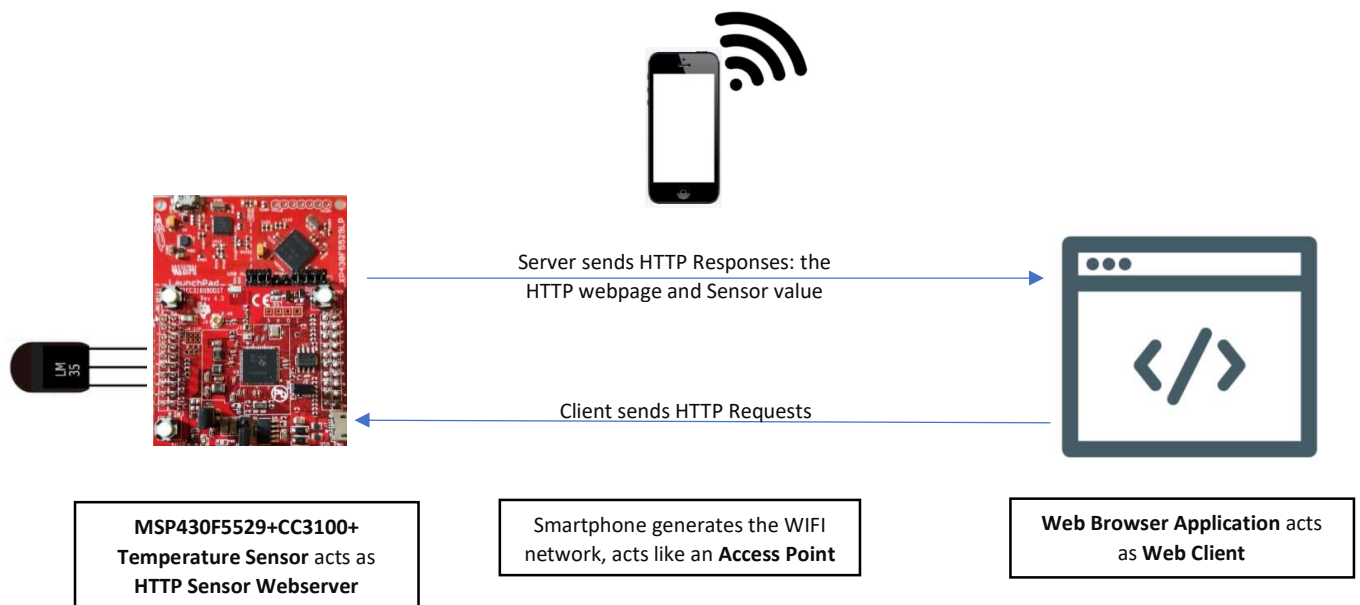
9. Design and Implementation of HTTP based IoT Web Server to display sensor value

Experiment Overview:

This experiment is an extension of the previous one. The goal of this lab is to integrate the HTTP webserver with Sensor. As we know, most IoT Nodes are equipped with sensors and the IoT systems should provide these sensor data to users. In this lab, we shall use LM35 temperature sensor connected to an IoT Node. The IoT Node is configured as a Webserver and the temperature sensor data can be accessed via a web browser. The understanding of this lab will also enable users to collect data from multiple sensors and monitor them via a web browser.

Hardware Required:

1 MSP430F5529 LaunchPad + 1 CC3100 BoosterPack + 1 LM35 Temperature Sensor

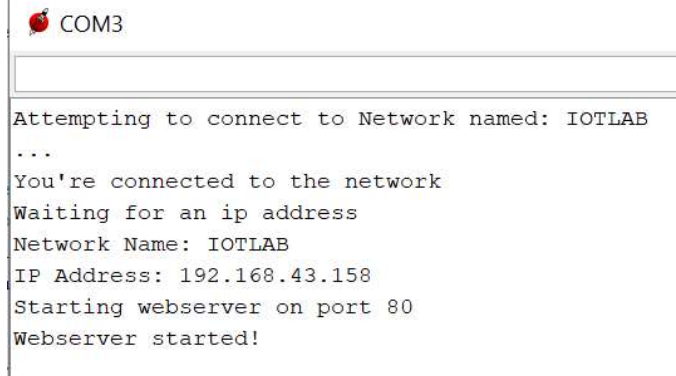


Procedure:

1. Connect the LM35 Temperature sensor to LaunchPad as in Experiment No.3.
2. Now, configure a webserver. Create a new Energia Sketch and clear its contents. Save this Sketch as IOT_sensor in your working directory. Now, copy the code from the IOT Manual folder, Experiment No. 9- HTTP_SensorServer and paste it to your Sketch. Save it.
3. Create a WIFI network/ Hotspot using your smartphone. Modify the following part of code with the correct credentials

```
// your network name also called SSID
char ssid[] = "your Wifi Network name";
// your network password
char password[] = "Wifi Password";
```

4. Connect the IoT Node (LaunchPad) to your PC. Verify and Upload the Sketch. Open the Serial Monitor, select the correct COM Port and baud rate set to 115200. Reset the LaunchPad



```
COM3
Attempting to connect to Network named: IOTLAB
...
You're connected to the network
Waiting for an ip address
Network Name: IOTLAB
IP Address: 192.168.43.158
Starting webserver on port 80
Webserver started!
```

5. The IoT Node has been configured as a Sensor Webserver. This IoT webserver can be communicated using its IP address (in our case its- 192.168.43.158)
6. Now let's use a PC/ smartphone/ laptop connected to the same WIFI network used by the IoT webserver. This example works on local networks only. Open google chrome web-browser and type the IP address of the webserver. Hit Enter.



Welcome to Sensor Server

Temperature Sensor Value is:

31

7. Observe that the HTTP code implemented on the server refreshes the web page every 2 seconds.

```
client.println("Refresh: 2"); // refresh the page automatically every 2 secs
client.println();
```

8. Assignment: You can make changes in the HTTP code to make the web page look more attractive and user friendly. Also, you can try interfacing different types of sensors and display their corresponding values on the web page.