Course: Summer Industrial Training 2015

**Lab Report:** Lab No: 2

Lab Name: Home automation using Psoc BLE

**Report By:** Virag Gada

Forum Suthar

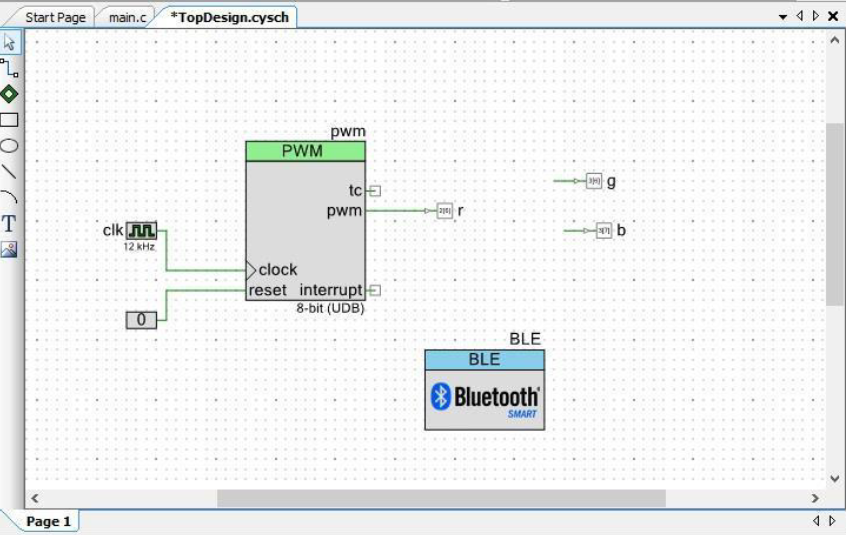
**Executive Summary:**

The objective for the project was to create home automation system which can control various devices through mobile phone (Bluetooth).

**Project Description:**

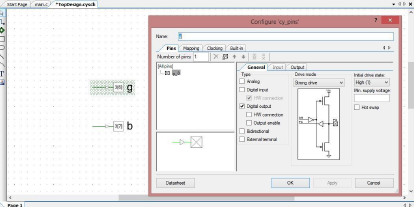
The input is given from the mobile to control various devices like on/off control intensity control etc. as we give input to psoc BLE board it controls various devices according to the input given. we create our own custom profile to contol various thing.

**Components used:**

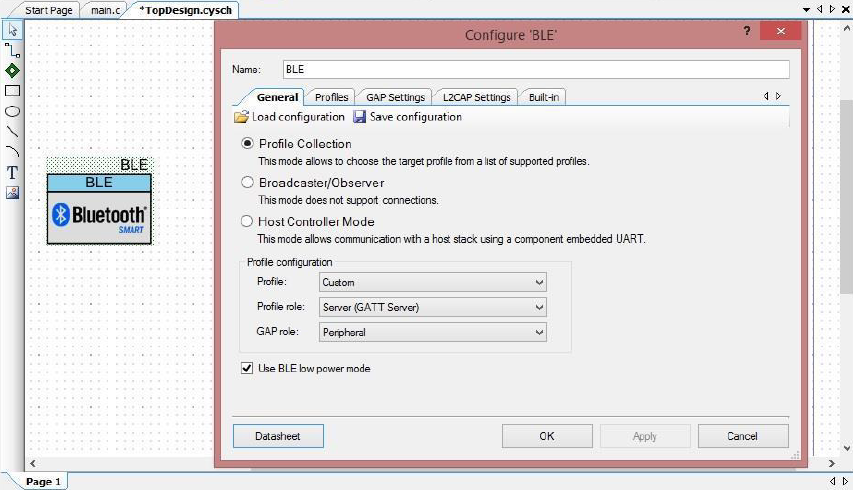


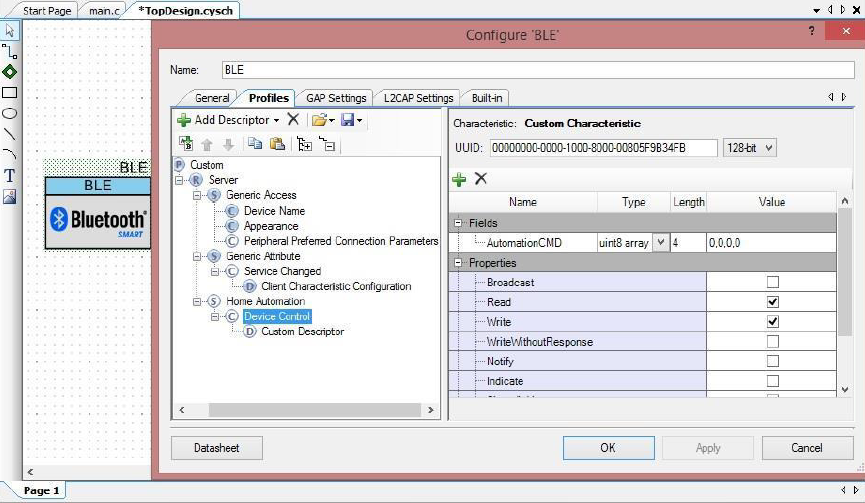
**Component Configuration and Pin map:**

**Digital Pin**

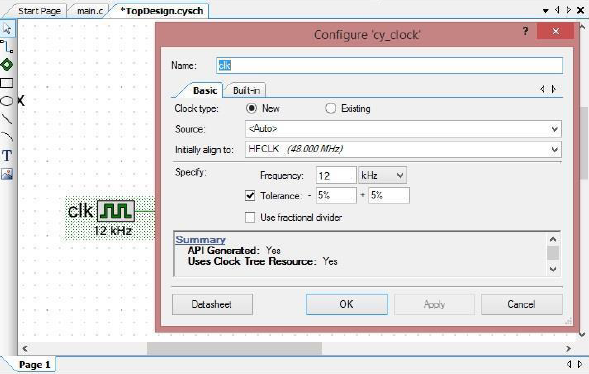


**Bluetooth Low Energy Module**

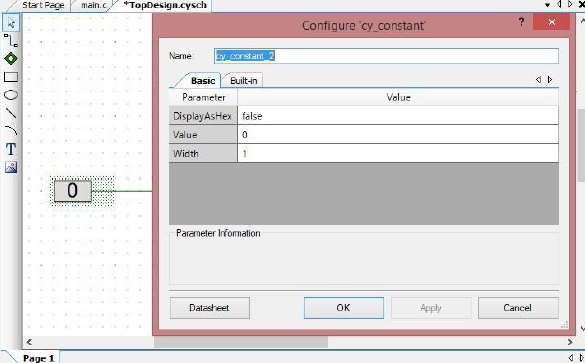




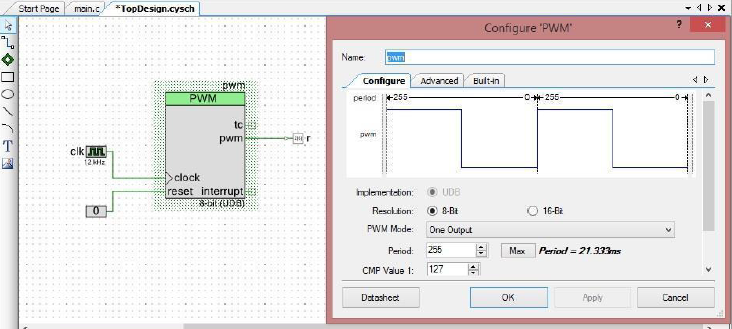
**Clock**



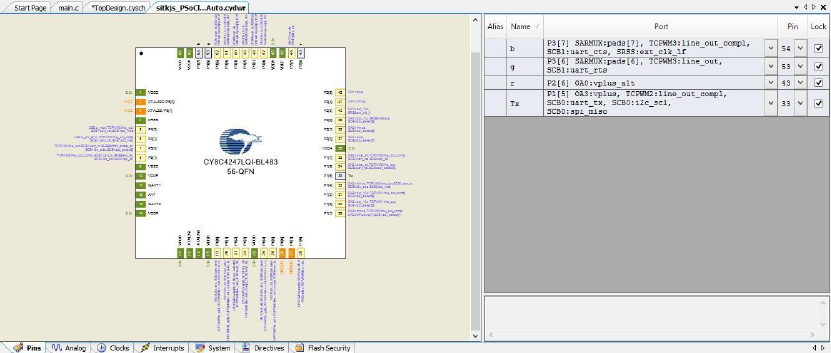
**Logic 0**



**PWM block**



**Pin Map**



**API Description**:

**BLE**

* **CYBLE\_EVT\_STACK\_ON:** This event is received when BLE stack is initialized and turnedON.
* **CyBle\_GappStartAdvertisement:** This function is used to start the advertisementusing the advertisement data set in the Component customizer's GUI
* **CYBLE\_EVT\_GAP\_DEVICE\_DISCONNECTED:** Disconnected from remotedevice. Parameter returned with the event contains pointer to the reason for disconnection
* **CYBLE\_EVT\_GATT\_CONNECT\_IND:** After completion of authentication events,peer and local GATT profiles are connected.
* **CYBLE\_EVT\_GATTS\_WRITE\_REQ:** This function has to be invoked in response to avalid Write Request event from the GATT Client to acknowledge that the attribute has been successfully written.
* **CyBle\_Start** : This function initializes the BLE Stack. It takes care of initializing the Profilelayer, schedulers, Timer and other platform related resources required for the BLE Component.
* **CyBle\_ProcessEvents** :This function checks the internal task queue in the BLE Stack, andpending operation of the BLE Stack.
* **CyBle\_GattsWriteRsp:** This function sends a Write Response from a GATT Server to theGATT Client. This function has to be invoked in response to a valid Write Request event from the GATT Client.

**PWM**

* **pwm\_Start() :** Initializes the PWM with default values
* **pwm\_WriteCompare ()**: Writes the compare values to configure the pwm componentaccording to the user.

**Test and Debug:**

* CySmart application could not detect the Bluetooth device. We re-installed the application and the problem was solved.
* The BLE device was not able to receive the data so no output was displayed.
* Once the LED was OFF still the intensity was changed. There were minor changes to be made in the program.

**Output Observed:**

* There where four bytes of data to be sent from CySmart Application to control the devices in our case LED.
* The first byte of data was for type of device whether ON/OFF or Intensity control.
* The second byte of data was for which device to control.
* The third byte of data was for either to turn ON or OFF or change the intensity of the device.
* The fourth byte of data was reserved.
* If the data sent was 0x01 0x02 0x01 0x00 then Green LED was turned on.
* If the data sent was 0x02 0x02 0x02 0x00 then we can control the intensity of the Green LED.

**Learning Outcomes:**

* How to configure the Bluetooth Low Energy Module.
* The basics of BLE was learnt.

**Code:**

#include <project.h>

#include <stdio.h>

CYBLE\_CONN\_HANDLE\_T connectionHandle; //Stores Connection Parameter CYBLE\_GATTS\_WRITE\_REQ\_PARAM\_T \*wrReqParam;

uint8 DevType,DevCode,DevParam,Reserve;

void HandleDevice()

{

if(DevType==0x01)

{

switch(DevCode)

{

{

break;

case 0x02: g\_Write(DevParam);

break;

case 0x03: b\_Write(DevParam);

break;

default: break;

}

}

}

if(DevType==0x02)

{

if(DevCode==0x01)

{

pwm\_Start(); pwm\_WriteCompare(DevParam);

}

if(DevCode==0x02)

{

pwm\_Stop();

}

}

}

void CustomEventHandler( uint32 event,void \*eventParam)

{

switch(event)

{

case CYBLE\_EVT\_STACK\_ON: //single and ready to mingle CyBle\_GappStartAdvertisement(CYBLE\_ADVERTISING\_FAST); break;

case CYBLE\_EVT\_GAP\_DEVICE\_DISCONNECTED: //break up and ready to mingle CyBle\_GappStartAdvertisement(CYBLE\_ADVERTISING\_FAST);

break;

case CYBLE\_EVT\_GATT\_CONNECT\_IND:

connectionHandle = \*(CYBLE\_CONN\_HANDLE\_T \*)eventParam

break;

case CYBLE\_EVT\_GATTS\_WRITE\_REQ:

wrReqParam = (CYBLE\_GATTS\_WRITE\_REQ\_PARAM\_T \*)eventParam; if(CYBLE\_HOME\_AUTOMATION\_DEVICE\_CONTROL\_CHAR\_HANDLE == wrReqParam ->

handleValPair.attrHandle)

{

DevType = (uint8) wrReqParam -> handleValPair.value.val[0]; DevCode = (uint8) wrReqParam -> handleValPair.value.val[1]; DevParam = (uint8) wrReqParam -> handleValPair.value.val[2]; Reserve = (uint8) wrReqParam -> handleValPair.value.val[3]; HandleDevice();

CyBle\_GattsWriteRsp(connectionHandle);

}

break;

default: break;

}

}

int main()

{

CyGlobalIntEnable;

CyBle\_Start(CustomEventHandler); uart\_Start();

uart\_PutString("Home Automation\n");

for(;;)

{

CyBle\_ProcessEvents ();

}

}