

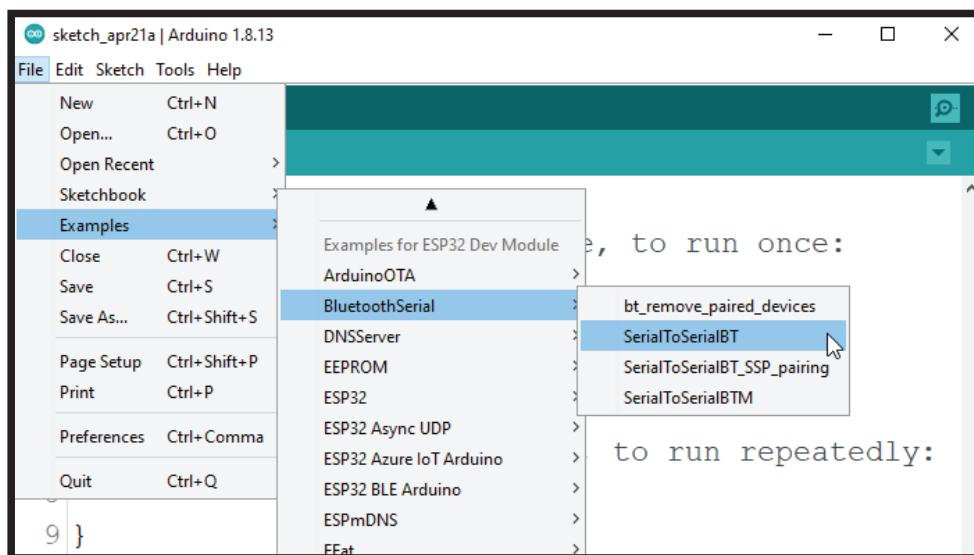
৮/ ব্লুটুথ কন্ট্রোল এবং অটোমেশন সিস্টেম:

প্রয়োজনীয় উপকরণ:

- ১/ এলইডি লাইট
- ২/ ব্রেডবোর্ড
- ৩/ জাম্পার ওয়্যার
- ৪/ JRC বোর্ড
- ৫/ ডিসি মটর ও প্রপেলার
- ৬/ ২২০ওহম রেজিস্ট্র

বর্ণনা: এই প্রজেক্টে আমরা JRC বোর্ডের একটি ইউনিক ফিচার নিয়ে কাজ করবো, সেটি হলো অন বোর্ড ব্লুটুথ সিস্টেম। এটি ব্লুটুথ ৪.০ টেকনোলজি এবং একই সাথে ব্লুটুথ লো এনার্জি (BLE) সমর্থন করে। এর মাধ্যমে যেমন তোমরা স্মার্টফোন এর সাথে যোগাযোগ স্থাপন করতে পারো, আবার অন্যান্য লো এনার্জি ডিভাইসের সাথেও এডভান্সড যোগাযোগ এর কাজও করতে পারবে। এই প্রজেক্টে আমরা শুধু স্মার্টফোন দিয়ে ব্লুটুথ কমিউনিকেশন করে দেখাবো যা কিনা খুব সহজেই কোডের মাধ্যমেই সম্পাদনা করা যায়। অতিরিক্ত কোন মডিউলের প্রয়োজনই নেই!

JRC বোর্ডে সাধারণ অবস্থায় ব্লুটুথ সিস্টেম বন্ধ অবস্থায় থাকে এবং আমরা পরবর্তীতে কোডের মাধ্যমে এটি অন করার ব্যবস্থা করে থাকি। প্রথমেই চলো দেখে নিই কিভাবে এক্সাম্পল কোড ওপেন করে সেখানে ব্লুটুথ কমিউনিকেশন সিস্টেম টি যাচাই করে দেখতে পারিঃ



এখানে Arduino IDE এর file এ ক্লিক করে এরপর examples এ ক্লিক করে এরপর SerialToSerialBT ফাইলটি সিলেক্ট করলে একটি নতুন এক্সাম্পল কোড ওপেন হবে। নিচে কোডের মূল অংশগুলো তুলে ধরা হচ্ছেঃ

```
#include "BluetoothSerial.h"
BluetoothSerial SerialBT;
void setup(){
    Serial.begin(115200);
    SerialBT.begin("ESP32test");
    Serial.print("BT Started!");
}
void loop(){
    if(Serial.available()){
        SerialBT.write(Serial.read());
    }
    if(SerialBT.available()){
        Serial.write(SerialBT.read());
    }
    delay(20);
}
```

এবার কোডের ব্যাখ্যার পালা। প্রথমেই তুমি যদি JRC বোর্ডে ব্লুটুথ ফিচার নিয়ে কাজ করতে চাও, তোমাকে অবশ্যই "BluetoothSerial" নামক লাইব্রেরি ব্যবহার করতে হবে যেখানে সকল দরকারী কমান্ড এবং ফাংশন সেভ করা থাকে। এই লাইব্রেরি কোডে প্রয়োগ করতে তোমাকে #include "BluetoothSerial.h" লাইন টি ব্যবহার করতে হয়েছে। এরপরের হলো তোমাকে একটা ফাংশন নাম তৈরী করা যে নাম দিয়েই তুমি বাদবাকি কাজ গুল সম্পাদন করে যাবে। এর জন্য আমরা BluetoothSerial SerialBT; লাইনটি ব্যবহার করেছি। এখানে ফাংশনের নাম দেয়া হয়েছে SerialBT। এই নামের দ্বারা তিনটা ফাংশনের প্রয়োগ আমরা কোডে দেখতে পাই। একটি হলো SerialBT.available() যার কাজ হলো ব্লুটুথে কোন ডাটা এসেছে কিনা সেটা পরীক্ষা করে দেখা। আরেকটি হলো SerialBT.write() যার কাজ হলো ব্লুটুথের মাধ্যমে অন্য ডিভাইসে ডাটা পাঠানো। সর্বশেষ হলো SerialBT.read() যার কাজ হলো অন্য ডিভাইস থেকে ব্লুটুথে যে ডাটা এসেছে সেটা রিসিভ করে রিডিং নেয়া।

এখানে একটা ব্যাপার লক্ষণীয় যে এই ব্লুটুথ সিস্টেম কিন্তু সিরিয়াল কমিউনিকেশন করছে তাইনা? একই ভাবে আমরা কিন্তু কম্পিউটার এর সাথে JRC বোর্ডের তথ্য আদান প্রদানের জন্যও কিন্তু এই সিরিয়াল কমিউনিকেশন করে থাকি। এক্ষেত্রে SerialBT এর বদলে সরাসরি **Serial** নাম ব্যবহার করে থাকি এবং এখানে একই রকম তি কমান্ড ফাংশন রয়েছে। এক্ষেত্রে আমরা কোডটি আপলোড দিয়ে JRC বোর্ডের সাথে কম্পিউটারের সংযোগ থাকা অবস্থায় সিরিয়াল মনিটর ওপেন করে সেখানে বোর্ডের সাথে কথা বলার কাজ তি করে থাকি যে ব্যাপারে তোমরা পূর্বেই দেখে এসেছো। তখন আমরা সেন্সর রিডিং নিয়ে সেটার ভ্যালু দেখার জন্য সিরিয়াল মনিটর কিন্তু ওপেন করে দেখতাম তাইনা? কোডের **loop()** ফাংশনের ভেতরে যে কাজটি করে দেখা হয়েছে তা তোমাদের এখন সহজে বুঝে ফেলার কথা। এখানে প্রথমে সিরিয়াল মনিটর চেক করে দেখছে যে তাকে আমরা কোন কমান্ড দিয়েছি কিনা। যদি কোন আমরা কোন লেখা সিরিয়াল মনিটরে ইনপুট করে থাকি সেক্ষেত্রে সিরিয়াল মনিটর সেই লেখাটি ইনপুট নিয়ে তা ব্লুটুথ এর মাধ্যমে অন্য ডিভাইসে পাঠিয়ে দেয়। এবং কোডের পরের অংশে উলটো কাজ হয়ে থাকে। যে ডিভাইসের সাথে ব্লুটুথ সংযোগ রয়েছে সেখান থেকে যদি কোন ডাটা আসে সেটি সিরিয়াল মনিটরে দেখানোর ব্যবস্থা করে থাকে। এভাবেই আমরা ব্লুটুথের মাধ্যমে ডাটা আদান প্রদান এর ব্যাপারটি পরীক্ষা করে দেখতে পারি।

The screenshot shows two windows from the Arduino IDE. The top window is titled "SerialToSerialBT | Arduino 1.8.13" and contains the following code:

```

SerialToSerialBT §
1 #include "BluetoothSerial.h"
2 BluetoothSerial SerialBT;
3 void setup() {
4     Serial.begin(115200);
5     SerialBT.begin("ESP32test");
6     Serial.print("BT Started!");
7 }
8 void loop() {
9     if(Serial.available()) {
10         SerialBT.write(Serial.read());
}

```

The bottom window is titled "COM3" and shows the serial monitor output. The text in the monitor is:

```

ets Jun 8 2016 00:22:57

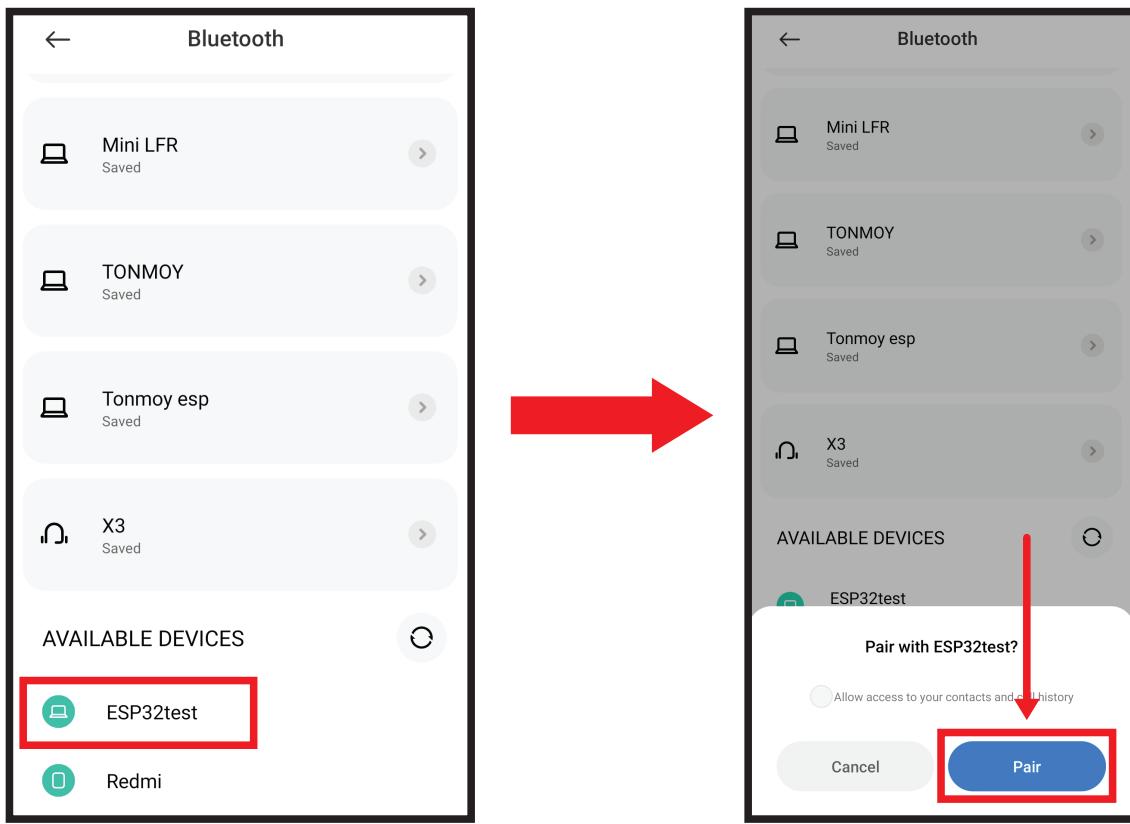
rst:0x1 (POWERON_RESET),boot:0x13 (SP)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs'
mode:DIO, clock div:1
load:0x3fff0018,len:4
load:0x3fff001c,len:1216
ho 0 tail 12 room 4
load:0x40078000,len:10944
load:0x40080400,len:6388
entry 0x400806b4
BT Started!

< >
 Autoscroll  Show timestamp
 No line ending  115200 baud  Clear output

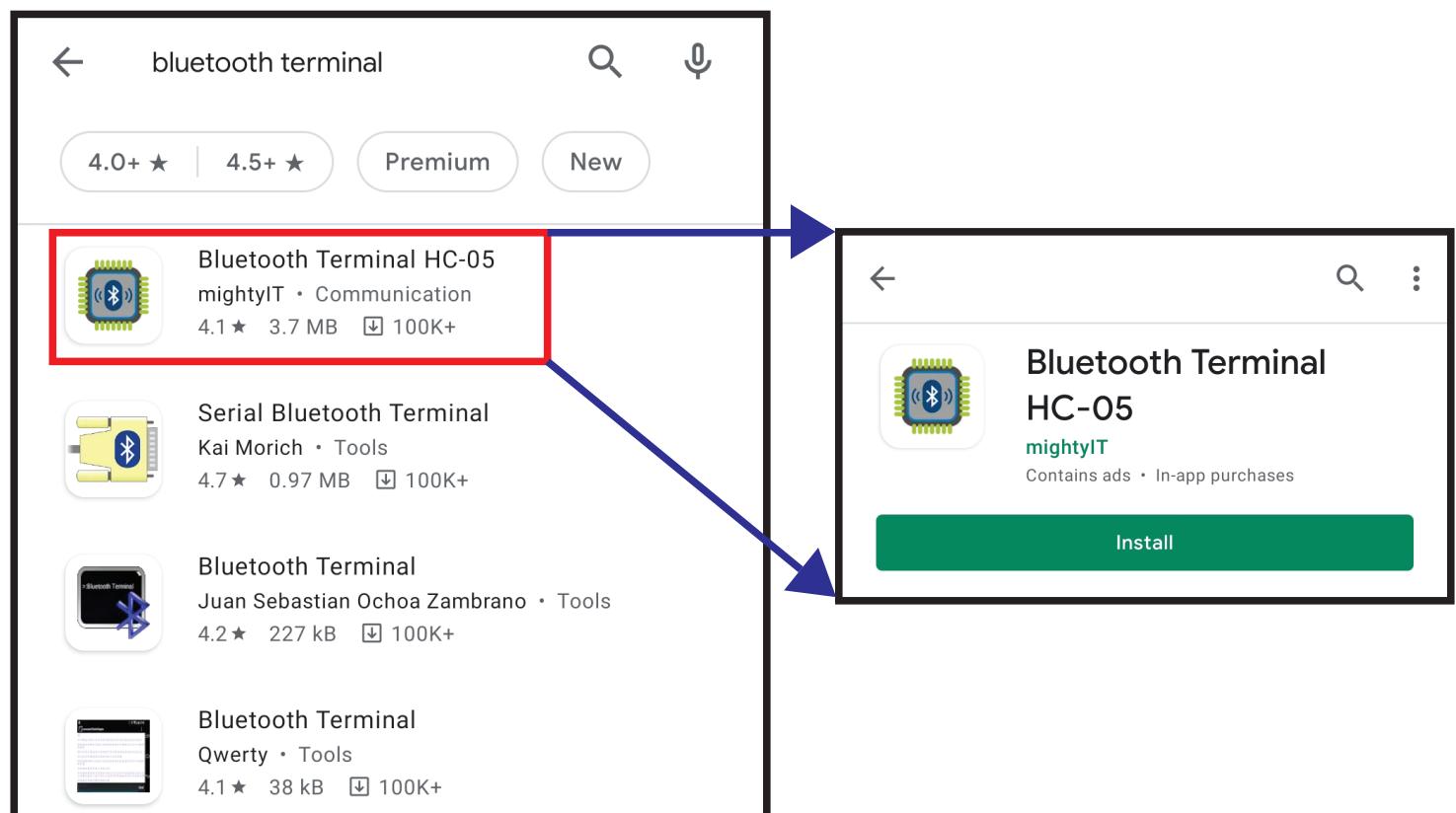
```

A red arrow points to the blue "Send" button in the top right corner of the Serial monitor window.

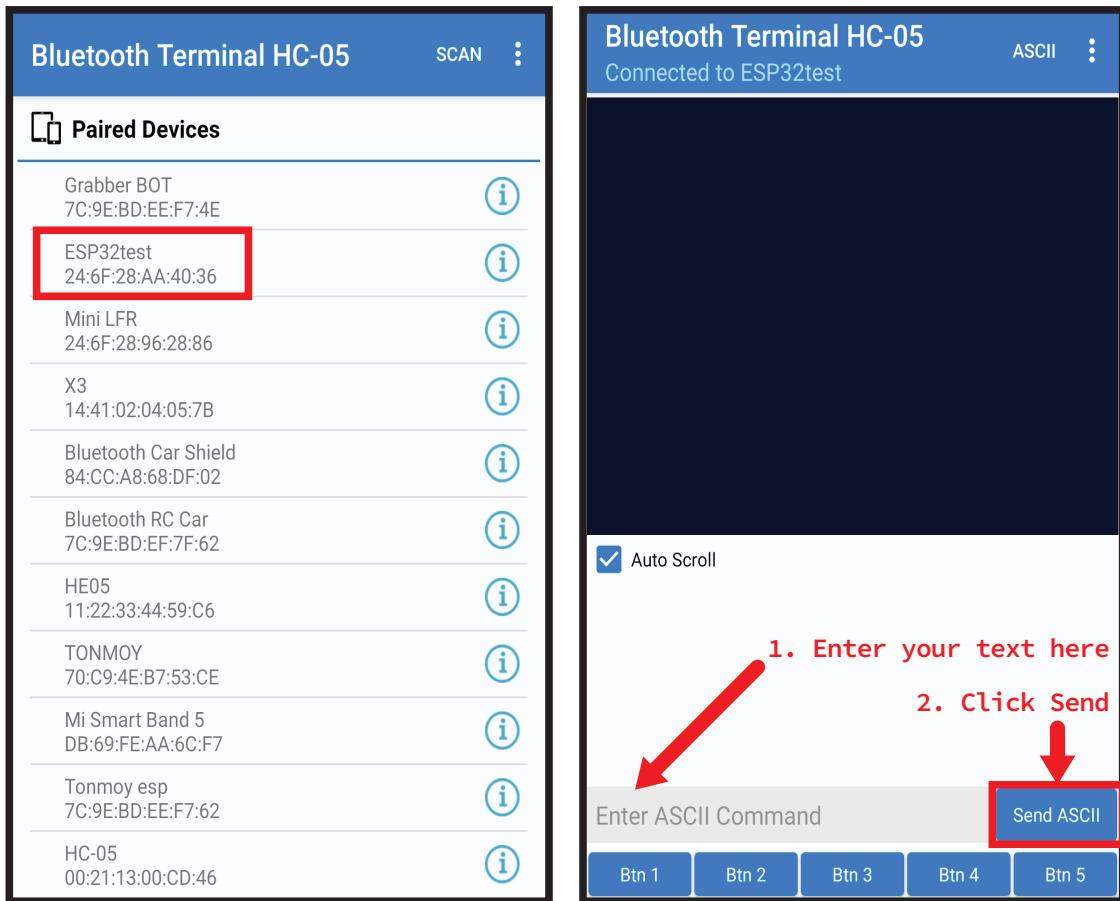
এখানে JRC বোর্ডের সাথে সংযোগ থাকা অবস্থায় সিরিয়াল মনিটর ওপেন করলে যদি দেখতে পাও যে এখানে "BT Started" লেখা তি প্রদর্শন করছে, এর মানে তুমি বুঝে ফেলবে যে তোমার JRC বোর্ডের ব্লুটুথ এস্টিভেট হয়ে গেছে। এবার কাজ হলো মোবাইল থেকে এটি পেয়ার করা। তুমি যদি তোমার স্মার্টফোনটির ব্লুটুথ অপশন তি ওপেন করো, সেখানে এভেইলেবল ব্লুটুথ ডিভাইসের তালিকার মধ্যে "ESP32test" নামে একটি ব্লুটুথ ডিভাইস দেখতে পাবে। সেটির সাথে পেয়ার করে ফেলো।



মোবাইলের ব্লুটুথে পেয়ার হয়ে গেলে এখন আমাদের যোগাযোগের জন্য একটি এপ নামাতে হবে। এর জন্য আমরা ব্লুটুথ পেয়ে স্টোরে গিয়ে "Bluetooth Terminal" এপ টি ডাউনলোড করে ফেলি।



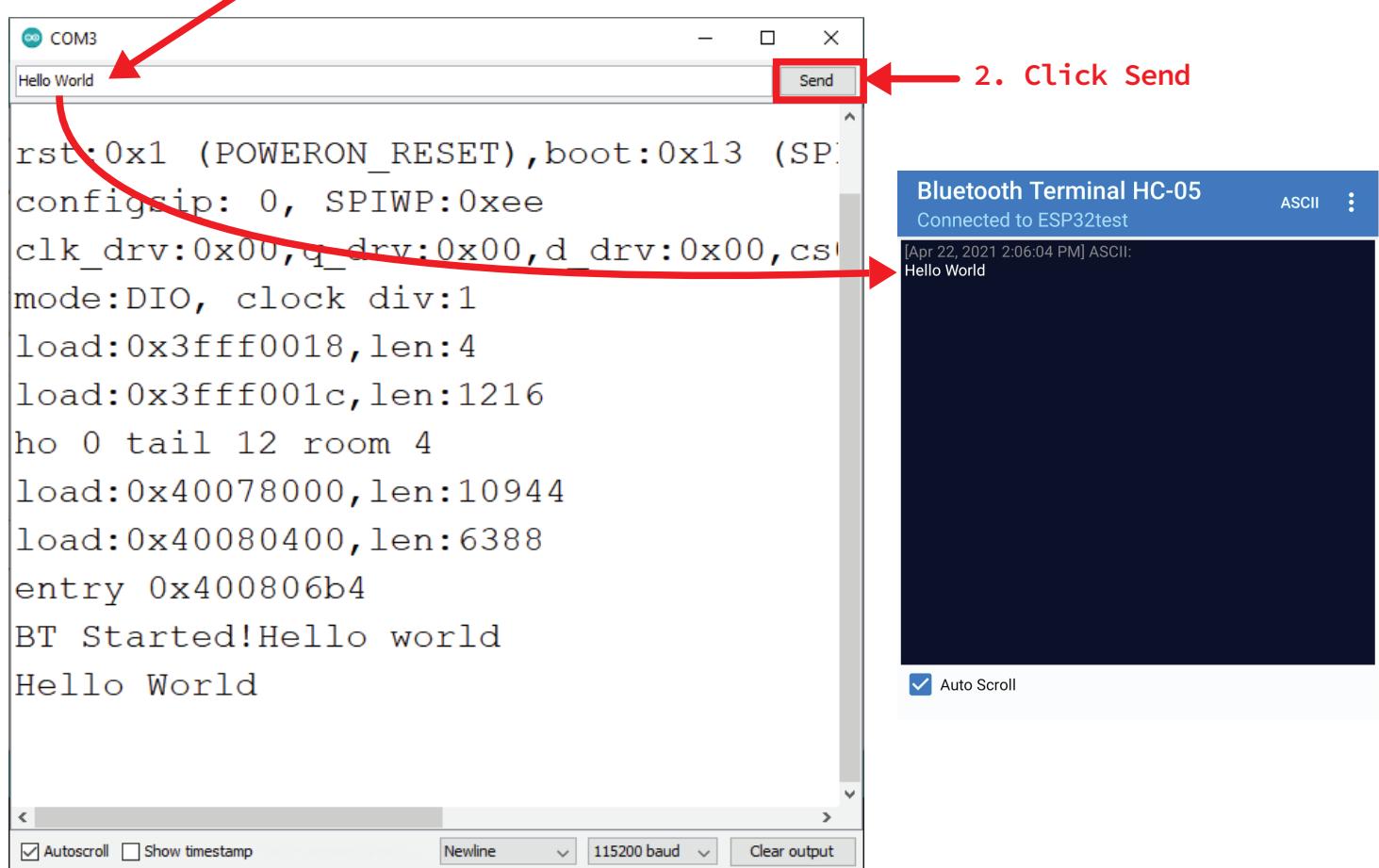
এপটি ইন্স্টল হয়ে গেলে ওপেন করে নিই। এখানে আমরা শুরুতেই যেসকল ব্লুটুথ ডিভাইস পেয়ার ক্রে রেখেছিলাম তার একটি তালিকা প্রদর্শন করবে। সেখান থেকে আমাদের কাঞ্চিত যে ব্লুটুথ ডিভাইস "ESP32test" সেটি সিলেক্ট করে নিই। লক্ষ্য রাখবে JRC বোর্ড টি যাতে সচল অবস্থায় থাকে। কানেকশন সম্পূর্ণ হলে একটি নতুন উইন্ডোতে নিয়ে যাবে যেখান থেকে আমরা ফোন এ JRC বোর্ডের মাঝে যোগাযোগ করতে পারি।



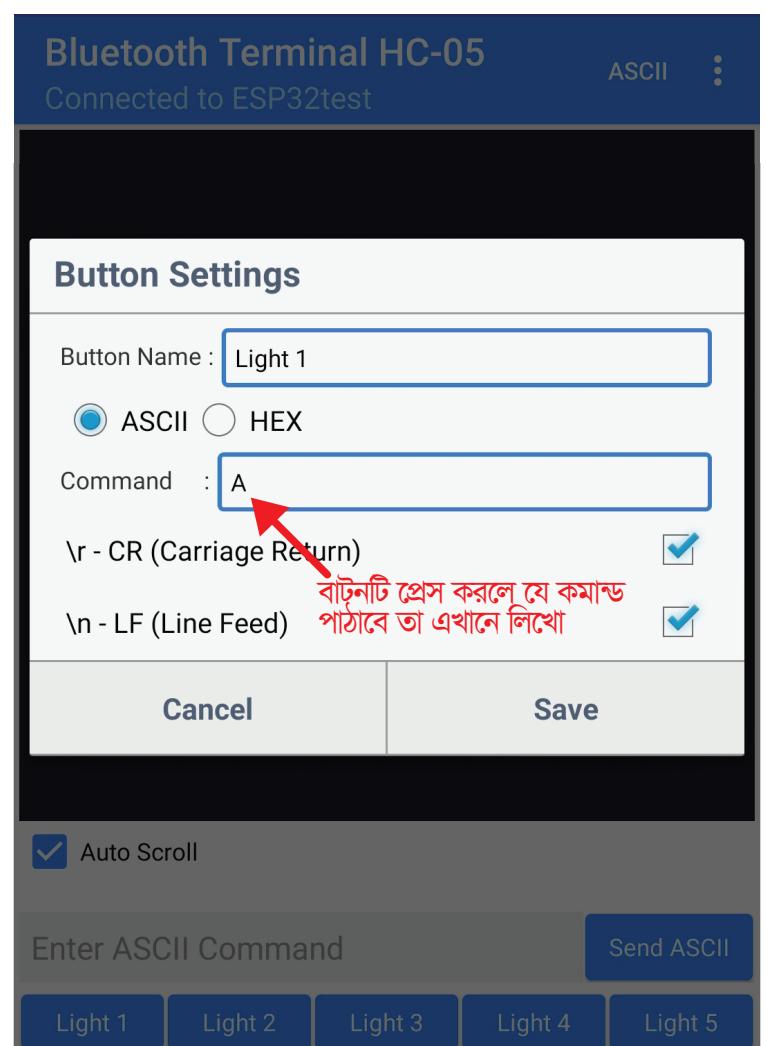
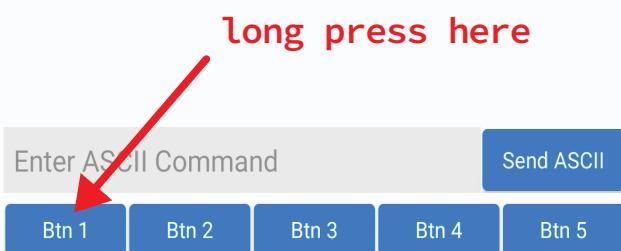
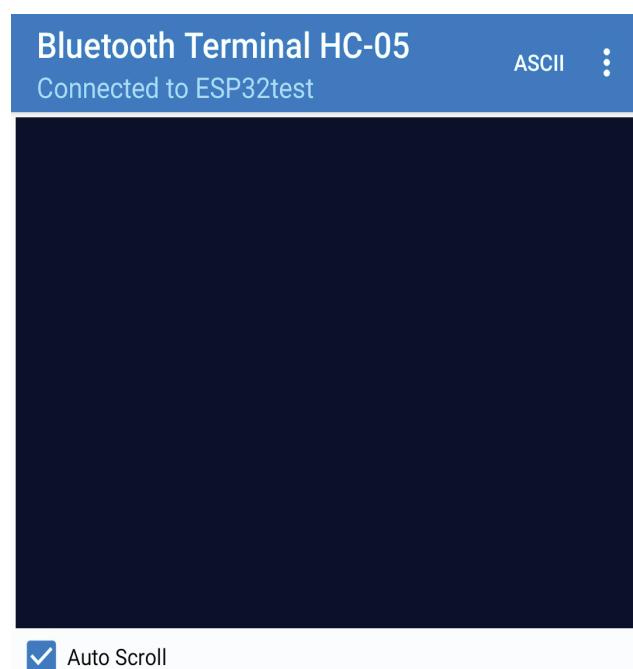
এখন আমরা চিহ্নিত ফাঁকা জায়গায় নিজের মনমতো মেসেজ লিখে সেন্ড করতে পারি। আমরা যে লেখা সেন্ড করবো সেটিই JRC বোর্ড রিসিভ করে কম্পিউটারের সিরিয়াল মনিটরে প্রদর্শন করবে।

The screenshots show the process of sending an ASCII message from a mobile application to a serial terminal window. In the mobile app, the message 'Hello World' is typed into the command field and sent via the 'Send ASCII' button. The terminal window then displays the received message 'Hello World'.

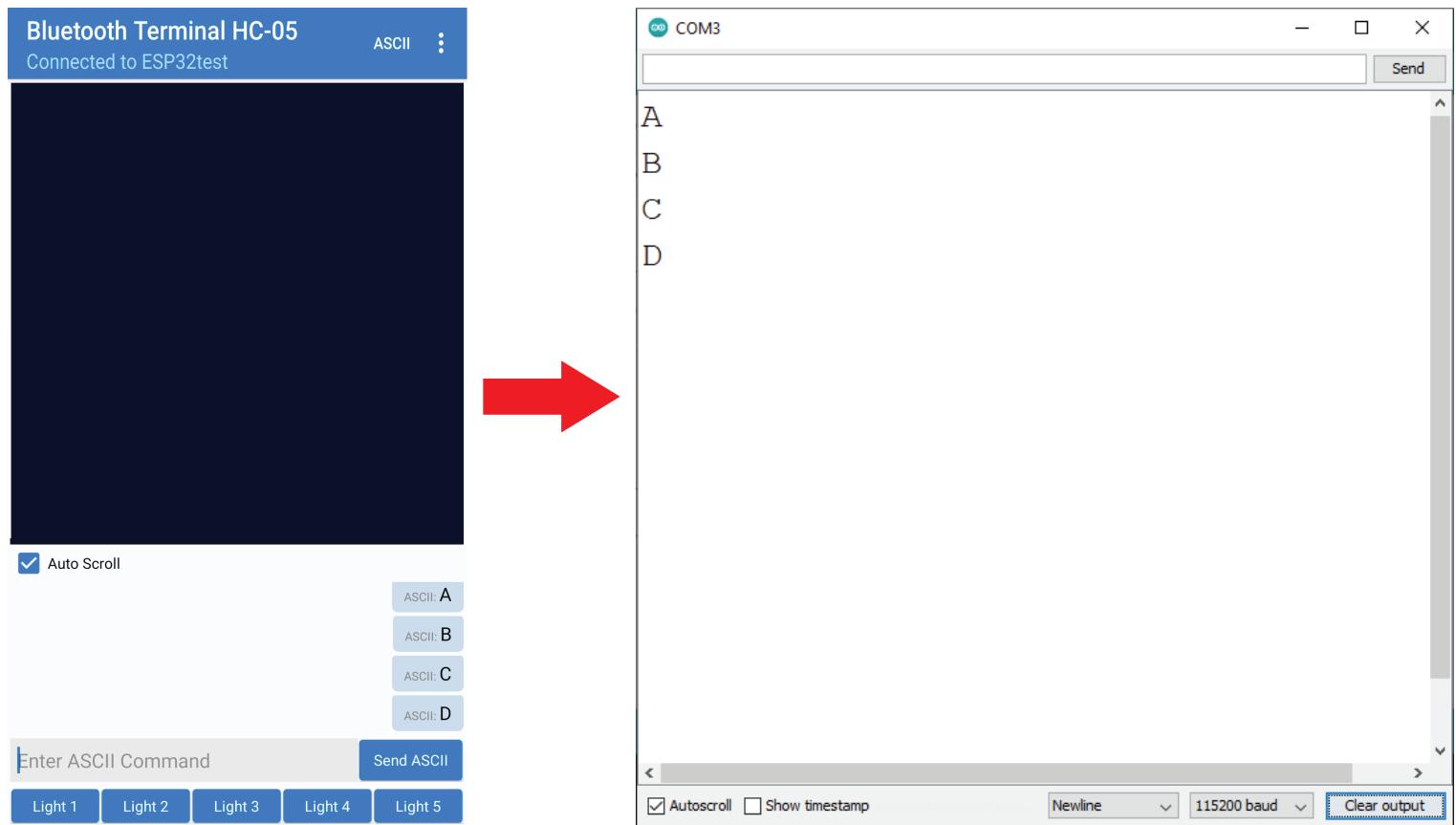
শুধু থেকে স্মার্টফোন থেকে বোর্ডে মেসেজ পাঠাতে পারবে তাইই নয়, একই সাথে তুমি চাইলে JRC বোর্ড থেকে স্মার্টফোনেও মেসেজ পাঠাতে পারবে। এক্ষেত্রে তোমাকে সিরিয়াল মনিটরে উপরে ফাঁকা বক্সে মনমতো মেসেজ লিখে সেন্ড বাটনে ক্লিক করলেই সেই মেসেজ টি স্মার্টফোনের স্ক্রিনে প্রদর্শন করবে।



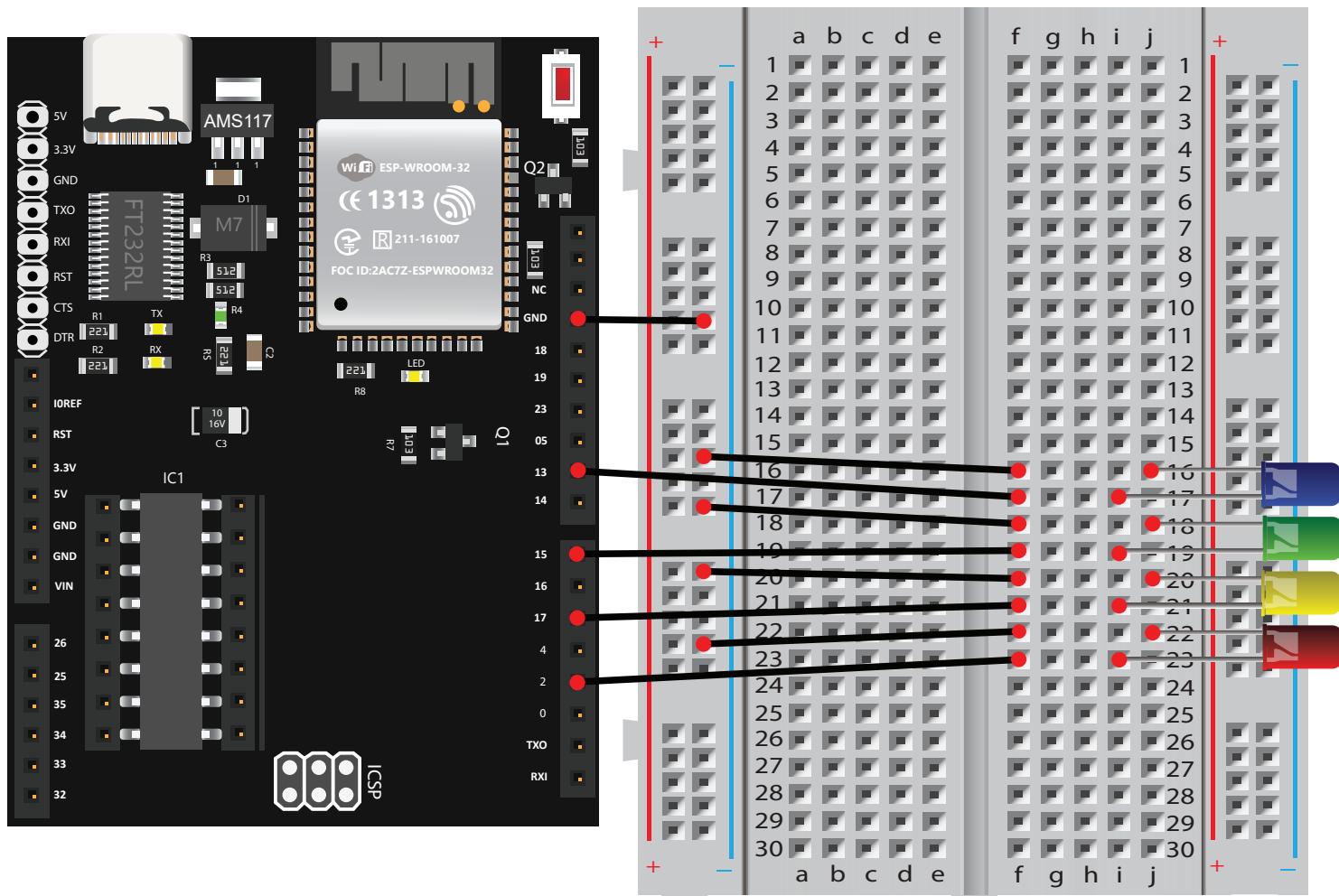
এভাবে চাইলে তুমি JRC বোর্ড এবং স্মার্টফোনের মাঝে ব্লুটুথের মাধ্যমে যোগাযোগ করতে পারবে। কিভাবে মেসেজ পাঠাতে হয় সেটা আমরা কিন্তু দেখে ফেলেছি। এখন সময় হচ্ছে সে মেসেজ কে কন্ডিশন স্টেটমেন্ট এর ভেতরে ফেলে সে অনুযায়ী বিভিন্ন ডীভাইস কন্ট্রোল করার। তার আগে লক্ষ্য করে দেখো যে এই এপে সবার নিচে কিছু বাটন রয়েছে যেগুলো চাইলেই মডিফাই করে নেয়া যায়। বাটনগুলো চাপ দিলে তোমার নির্দেশনা অনুযায়ী ক্যারেন্টার পাঠাতে পারে। নিচের ছবিতে একটি বাটন মডিফাই করে দেখানো হচ্ছে যা দেখে তোমরা বাকিগুলোও মডিফাই করে নিতে পারো।



মডিফাই করা হয়ে গেলে এই বাটনগুলোকে তুমি ব্লুটুথ রিমোট হিসেবে ব্যবহার করতে পারবে। এক্ষেত্রে তুমি একেকটি বাটন প্রেস করলে JRC বোর্ডে যে ক্যারেষ্টার পাঠায়, সেটির উপর কন্ডিশন প্রয়োগ করে দিয়েই কিন্তু লাইট জ্বালানো-নিভানো যাবে। যেমন এখানে ৫ টি বাটন এর প্রতিটি প্রেস করলে যথাক্রমে A, B, C, D, E এই ক্যারেষ্টারগুলো পাঠাবে। এর মানে আমরা এখানে কোন ক্যারেষ্টার রিসিভ করছি সে অনুযায়ী কন্ডিশন সাজিয়ে নিতে পারি।



যেহেতু এটি দিয়ে আমরা JRC বোর্ডে কিছু এলাইডি কন্ট্রোল করতে চাচ্ছি, সেক্ষেত্রে নিচের সার্কিট টি তৈরী করে নিতে পারি।



উপরের সার্কিটে আমরা 4 টি এলইডি লাইটের সাথে JRC বোর্ডের সংযোগ স্থাপন করেছি যেখানে সবগুলো লাইটের খণ্ডত্বক প্রাপ্ত একত্রিত করে গ্রাউন্ড পিনের সাথে এবং অন্যান্য পিনগুলো যথাক্রমে 13, 15, 17 এবং 2 নাম্বার ডিজিটাল পিনের সাথে সংযোগ প্রদান করা হয়েছে যেখানে যেকোন একটি ডিজিটাল পিনে হাই আউটপুট দিলেই লাইট টি জ্বলে উঠবে।

এবার কোডিং এর পালা। আমরা জানি যে JRC বোর্ডের ব্লুটুথ কিভাবে এস্টিভেট করতে হয় যেখানে কিনা আমরা **BluetoothSerial SerialBT;** লাইনের মাধ্যমে **SerialBT** নামে একটি ফাংশন তৈরী করা হয়েছে। এবং এক্ষেত্রে **SerialBT.available()** এর মাধ্যমে সিগ্নাল আসছে কিনা চেক করছি এবং **SerialBT.read()** এর মাধ্যমে আমরা ডাটা রিডিং নিচ্ছি। এখানে স্মার্টফোনে এপের ওই বাটনগুলো প্রেস করলে A, B, C, D, E এই পাঁচটির মধ্যে কোন ক্যারেন্টার আসছে সেটা চেক করে দেখি এবং সে অনুযায়ী আমরা লাইট জ্বলাতে-নিভাতে পারি। নিচের কোডটি লক্ষ্য করো।

```
#include "BluetoothSerial.h"
BluetoothSerial SerialBT;

int state1=0,state2=0,state3=0,state4=0;

void setup(){
    Serial.begin(115200);
    SerialBT.begin("ESP32test");
    Serial.print("BT Started!");
    pinMode(13, OUTPUT);
    pinMode(15, OUTPUT);
    pinMode(17, OUTPUT);
    pinMode(02, OUTPUT);
}

void loop(){
    if(SerialBT.available()) {
        char x = SerialBT.read();
        Serial.println(x);
        if(x=='A'){
            state1 = !state1; digitalWrite(13,state1);
        }
        else if(x=='B'){
            state2 = !state2; digitalWrite(15,state2);
        }
        else if(x=='C'){
            state3 = !state3; digitalWrite(17,state3);
        }
        else if(x=='D'){
            state4 = !state4; digitalWrite(02,state4);
        }
    }
    delay(20);
}
```

এখানে চারটি লাইট জ্বলে আছে নাকি নিতে আছে সেই অবস্থা মনে রাখার জন্য আমরা চারটি ভ্যারিয়েবল নিচ্ছি যেগুলো নাম দিয়েছি যথাক্রমে state1, state2, state3, state4। এগুলোর মান শুরুতে 0 ধরে রেখেছি কারণ শুরুতে সবগুলো লাইট অফ হয়ে আছে। এরপর আমরা ব্লুটুথ ইনিশিয়েট করেছি এবং লাইটগুলো যে পিনে লাগানো সেই পিনগুলোতে আউটপুট হিসেবে ডিক্রেয়ার করেছি। এরপর ভেতরে আমরা বার বার চেক করে দেখছি যে কোন ক্যারেন্টার আসছে কিনা। যদি কোন ক্যারেন্টার রিসিভ হয়, সেক্ষেত্রে আমরা চেক করে দেখছি যে ক্যারেন্টার কোনটা। ধরে নিচ্ছি আমরা A রিসিভ করেছি। এক্ষেত্রে আমরা এর দ্বারা প্রথম লাইট টি কন্ট্রোল করবো। এখন আমরা এখানে চেক করে দেখবো যে প্রথম লাইটের কন্ডিশন কি। "state1 = !state1" এই লাইনটিতে '!state1'-এখানে একটি আশ্চর্যবোধক চিহ্ন রয়েছে। এর কাজ হলো state1 এর মধ্যে জমা রাখা ভ্যালু টি উলটে দেয়া। এখানে যদি 0 সেভ করা থাকে, সেক্ষেত্রে আমরা 1 পাবো, যদি 1 সেভ করা থাকে, সেক্ষেত্রে 0 পাবো। যে ভ্যালু টা পাচ্ছি সেটা আবার আমরা state1 ভ্যারিয়েবল এর মধ্যেই সেভ করে রাখছি। এর মানে যেটা দাঁড়ায় সেটা হলো state1 নিজেই নিজের ভ্যালু উলটে সেভ করে রাখছে। এরপর আমরা এই ভ্যালু ঠিক পরেই digitalWrite() ফাংশনের ভেতরে ব্যবহার করছি। এখানে আমরা যদি digitalWrite(13, 1) লিখতাম, সেক্ষেত্রে লাইট টি জ্বলে উঠতো (কারণ HIGH মানে 1 ধরে নেয়া হয়)। আবার যদি digitalWrite(13, 0) লিখতাম, সেক্ষেত্রে লাইট টি নিতে যেতো (কারণ LOW মানে 0 ধরে নেয়া হয়)। তাহলে ব্যাপার টা কি দাঢ়ালো? একবার A ক্যারেন্টার টি পেলে লাইট জ্বলে উঠবে, আরেকবার A ক্যারেন্টার পেলে লাইট টি নিতে যাবে। এভাবে বাকি লাইটগুলোতেও একই সিস্টেম করে দেয়া হয়েছে। সুতরাং তোমাদের ব্লুটুথ কন্ট্রোলড এলইডি লাইট প্রজেক্ট টি রেডি!