```
/*

  Launchpad DSTR Control Code

  Target Device: CC3200-LAUNCHXL (CC3200 LaunchPad)

  Needed BoosterPacks: None

  Additional Hardware Required:
    * L298N Dual H-Bridge Motor Driver or compatible Motor control board from PWMs
    * Motors compatible with your power driver



  @Brief -
    Start WiFi in in AP mode creating a network that you can connect to
    from a SmartPhone. (NOTE: There is a limitation that only ONE device
    can be connected. Trying to connect to the network with a second client
    will fail. To connect with a different device, first disconnect the device
    currently connected.)

    The "DSTR Controller" App available for Android and iPhone will send packets
    to Port 3553. The value for localPort declared in this code needs to match
that.
    Packets received will contain four bytes, two that represent motor speeds and
    two that represent a direction

    This control code must differentiate between the direction values received in
the
    packet in order to decide which



  Originally written by Vince Rodriguez in July of 2016
  Editted by Colby Ryan in March of 2017 for educational purposes at Texas A&M
University
*/

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

char wifi_name[] = "pickleperry"; //Change this value to the desired broadcast
name of your Wi-Fi
char wifi_password[] = "dstrdstr"; //Changing this is not required

/* NOTE: This port is already defined in the "DSTR" app, changing this will make
```

```
the default app non-functional */
unsigned int localPort = 3553;      // local port to listen on
WiFiUDP Udp;

char packetBuffer[255]; //buffer to hold incoming packet
char  ReplyBuffer[] = "acknowledged";       // a string to send back

void setup()
{

  // Start WiFi and create a network with wifi_name as the network name
  // with wifi_password as the password.
  pinMode(GREEN_LED, OUTPUT);  //On the CC3200 Launchpad, "GREEN_LED" is
boosterpack pin 10 - marked P02 on the silk screen
  pinMode(YELLOW_LED, OUTPUT); //On the CC3200 Launchpad, "YELLOW_LED" is
boosterpack pin 9 - marked P01 on the slik screen
  pinMode(RED_LED, OUTPUT);  //On the CC3200 Launchpad, "RED_LED" is boosterpack
pin 29 - marked P64 on the silk screen
  pinMode(31, OUTPUT);     //On the CC3200 Launchpad, boosterpack pin 31 is
marked P17 on the silk screen
  analogWrite(GREEN_LED, 0);
  analogWrite(YELLOW_LED, 0);
  analogWrite(RED_LED, 0);
  analogWrite(31, 0);
  Serial.begin(115200);
  Serial.print("Starting network...");
  WiFi.beginNetwork(wifi_name, wifi_password);
  Serial.println("done.");
  Udp.begin(localPort);
  Serial.println("Please input PWM in LMOTORS|RMOTORS format!");




}

void loop()
{

  int packetSize = Udp.parsePacket(); //Waits for receipt of a packet and stores
the data into a rx buffer
  if (packetSize)
  {
    Serial.print("Received packet of size ");
    Serial.println(packetSize);
```

```
    Serial.print("From ");
    IPAddress remoteIp = Udp.remoteIP(); //Returns the IP of the remote device
connected to the launchpad
    Serial.print(remoteIp);
    Serial.print(", port ");
    Serial.println(Udp.remotePort()); //Returns the remote port which the packet
was received on
    // read the packet into packetBufffer
    int len = Udp.read(packetBuffer, 255); //Copies up to 255 bytes of data from
the rx buffer into the packetBuffer array
    if (len > 0) packetBuffer[len] = 0;
    Serial.println("Contents:");
   // Serial.println(packetBuffer);
    Serial.println(packetBuffer[0], DEC);
    Serial.println(packetBuffer[1], DEC);
    Serial.println(packetBuffer[2], DEC);
    Serial.println(packetBuffer[3], DEC);

    if(len == 4)
      {

      if(packetBuffer[0] == 0xaa){ // direction 1
        analogWrite(GREEN_LED,packetBuffer[3]);
        analogWrite(YELLOW_LED,255);
      }
      if(packetBuffer[0] == 0xbb){ // direction 1
        analogWrite(GREEN_LED,255);
        analogWrite(YELLOW_LED,packetBuffer[3]);
      }
      if(packetBuffer[2] == 0xaa){ // direction 2
        analogWrite(RED_LED,packetBuffer[1]);
        analogWrite(31,255);
      }
      if(packetBuffer[2] == 0xbb){ // direction 2
        analogWrite(RED_LED,255);
        analogWrite(31,packetBuffer[1]);
      }

      Serial.println("PWM SET");
     }
  }
}
```