```
//////AK_FINAL_CPT///////
//v3.1.0
// Enjoy! - Ali Khan
/*
Note to self:
DO NOT MAKE A LOCAL VAR, YOUR MOTORS WILL NOT WORK
× int motorspeed = map
✔ motorspeed = map

TB6612FNG mappings:
vm = motor voltage; vcc = logic voltage; gnd = ground; a01, a02 = outputs
for motor A
b02, b03 = outputs for motor B; PWMA = A channel pwm; PWMB = B channel
PWM;
AI2, AI1 = input control for A; BI1, BI2 = input control for B;

CPT checklist:
STAGE 1: BASIC MOTOR CONTROL ✔ | STAGE 2: ASSEMBLE CHASSIS ✔ | STAGE 3:
DUAL MOTOR ✔
STAGE 4: JOYSTICK ✔ | STAGE 5: SCHEMATIC ✔ | STAGE 6: WIRING ✔ | STAGE
7: CODING ✔
STAGE 8: AWD (4 Wheels), JOYSTICK, TWO RED LEDS (SERIES), MATRIX LED
CONFIG ✔ | STAGE 9: FINAL ASSEMBLY ✔
*/

#pragma region ARDUINO LED MATRIX

/*
Make an instance of the LED Matrix and using the matrix load
sequence function, load these files and play them onto the matrix.
Finally set the boolean in matrix.show() function to true to allow
displaying the
animations.
*/

//HEADER FILES//
#include "Arduino_LED_Matrix.h"
#include "anim_back.h"
#include "anim_forward.h"
#include "anim_right.h"
#include "anim_left.h"

ArduinoLEDMatrix matrix;

#pragma endregion

#pragma region PIN DEFINITION

//POWER//
#define pwmM 5

//SET 1//
// Motor A -> FRONT LEFT
#define in1A 3
```

```
#define in2A 4
// Motor B -> FRONT RIGHT
#define in1B 7
#define in2B 8

//SET 2//
//Motor A -> BACK RIGHT
#define in1A_2 10
#define in2A_2 11
//Motor B -> BACK LEFT
#define in1B_2 12
#define in2B_2 13

//JOYSTICK//
#define btn_joy 2
#define xAxis_In A0
#define yAxis_In A1

//LED SERIES//
#define LED_IN 6

//INIT//
int MotorSpeed = 0;
int xAxis;
int yAxis;

#pragma endregion

#pragma region SETUP(); FOR PINS
void setup()
{
  // Set all the motor control pins to outputs
  Serial.begin(9600);

  //PIN SET 1//
  // Motor A
  pinMode(in1A, OUTPUT);
  pinMode(in2A, OUTPUT);
  // Motor B
  pinMode(in1B, OUTPUT);
  pinMode(in2B, OUTPUT);

  //PIN SET 2//
  // Motor A_2
  pinMode(in1A_2, OUTPUT);
  pinMode(in2A_2, OUTPUT);
  // Motor B_2
  pinMode(in1B_2, OUTPUT);
  pinMode(in2B_2, OUTPUT);

  //JOYSTICK_IN//
  pinMode(btn_joy, INPUT);

  //LED//
```

```
  pinMode(LED_IN, OUTPUT);

  //MATRIX DISPLAY//
  matrix.begin();
}
#pragma endregion

#pragma region ALL SUBROUTINES: FORWARD(); -> BACKWARD(); -> LEFT(); ->
RIGHT(); -> STOP(); -> PRINTDATA();

void forward() { //Set all motors to go forward.

  digitalWrite(in1A, HIGH);
  digitalWrite(in2A, LOW);

  digitalWrite(in1B, LOW);
  digitalWrite(in2B, HIGH);

  digitalWrite(in1A_2, LOW);
  digitalWrite(in2A_2, HIGH);

  digitalWrite(in1B_2, HIGH);
  digitalWrite(in2B_2, LOW);

}

void backward() { //Set all motors to go backwards.

  digitalWrite(in1A, LOW);
  digitalWrite(in2A, HIGH);

  digitalWrite(in1B, HIGH);
  digitalWrite(in2B, LOW);

  digitalWrite(in1A_2, HIGH);
  digitalWrite(in2A_2, LOW);

  digitalWrite(in1B_2, LOW);
  digitalWrite(in2B_2, HIGH);
}

void left() { //Set a row of motors to go left while the motor moves
backwards, ensuring a smooth left turn.

  digitalWrite(in1B, LOW);
  digitalWrite(in2B, HIGH);

  digitalWrite(in1A_2, LOW);
  digitalWrite(in2A_2, HIGH);

  digitalWrite(in1A, LOW);
  digitalWrite(in2A, HIGH);

  digitalWrite(in1B_2, LOW);
```

```
  digitalWrite(in2B_2, HIGH);

}

void right() { //Set a row of motors to go right while the motor moves
backwards, ensuring a smooth right turn.

  digitalWrite(in1A, HIGH);
  digitalWrite(in2A, LOW);

  digitalWrite(in1B_2, HIGH);
  digitalWrite(in2B_2, LOW);

  digitalWrite(in1B, HIGH);
  digitalWrite(in2B, LOW);

  digitalWrite(in1A_2, HIGH);
  digitalWrite(in2A_2, LOW);

}


void stop() { //Set to all low + zero power == no movement (full stop).

  digitalWrite(in1A, LOW);
  digitalWrite(in2A, LOW);

  digitalWrite(in1B, LOW);
  digitalWrite(in2B, LOW);

  digitalWrite(in1A_2, LOW);
  digitalWrite(in2A_2, LOW);

  digitalWrite(in1B_2, LOW);
  digitalWrite(in2B_2, LOW);

  analogWrite(pwmM, 0);
}

void printData(int datType) { //Moved print code here to offer a cleaner
viewing experience

/*Note these names are all messed up, the rotations are not as they seem,
when editing the pixels in the Arduino matrix editor.  (weird naming
conventions). */

  switch(datType) {
    case 1:
    Serial.print("Y:");
    Serial.println(yAxis);
    Serial.print(" moving backwards");
    Serial.println("\n");
    matrix.loadSequence(anim_left);
    matrix.play(true);
```

```
    break;
    case 2:
    Serial.print("Y:");
    Serial.println(yAxis);
    Serial.print(" moving forwards");
    Serial.println("\n");
    matrix.loadSequence(anim_right);
    matrix.play(true);

    break;
    case 3:
    Serial.print("X:");
    Serial.println(xAxis);
    Serial.print("moving left");
    Serial.println("\n");
    matrix.loadSequence(anim_forward);
    matrix.play(true);

    break;
    case 4:
    Serial.print("X:");
    Serial.println(xAxis);
    Serial.print("moving right");
    Serial.println("\n");
    matrix.loadSequence(anim_back);
    matrix.play(true);
    break;
    case 5:
    Serial.println("\n");
    Serial.println("Force Stopped Initiated");
    matrix.play(true);
  }

}

#pragma endregion

#pragma region MAIN LOOP();
void loop() {

  xAxis = analogRead(xAxis_In);  //Read x-axis
  yAxis = analogRead(yAxis_In);  //Read y-axis

  if (digitalRead(btn_joy) == 1) //Force stop
  {
    stop();
    printData(5);
  }

  if (yAxis < 470) //Move backwards
  {
    MotorSpeed = map(yAxis, 470, 0, 0, 255);
    analogWrite(LED_IN, MotorSpeed);
```

```
    backward();
    printData(1);
  }


  else if (yAxis > 550) //Move forward
  {
    MotorSpeed = map(yAxis, 550, 1023, 0, 255);
    forward();
    printData(2);
   }


  else if (xAxis < 470) //Move left
  {
    MotorSpeed = map(xAxis, 470, 0, 0, 255);
    left();
    printData(3);
  }

  else if (xAxis > 550) //Move right
  {
    MotorSpeed = map(xAxis, 550, 1023, 0, 255);
    right();
    printData(4);
  }
    else {
    analogWrite(LED_IN, 0);
    MotorSpeed = 0;
    stop();
  }
  analogWrite(pwmM, MotorSpeed);
}
#pragma endregion
```