/\*

 First Wheels Fluid Motion Device V 2.2

 Jesus said, “Let the little children come to me, and do not hinder them, for the kingdom of heaven belongs to such as these.” Matthew 19:14

Reads digital input pins to determine button states. Based on current motor speeds and directions, the program will smoothly

transition to the requested state (ex: transition from accelerating forward to turning/spinning left).

Reads an analog input pin (A0) connected to a potentiometer, maps the result to a range from 0 to 255

and uses the result to set the max speed.

Second analog pin (A2) connected to second potentiometer to provide steering trim control to attempt to compensate for variations

in speed between the motors.

Originally created March 2015

 by Blake Palmer of

 First Wheels Special Needs Mobility Ministry,

 A project of Thru the Roof Special Needs Ministry at Houston's First Baptist Church.

 http://www.firstwheelshouston.org

 http://www.houstonsfirst.org

 \*/

 /\* v2.2 6/3/2015 - adds logic for steering left/right trim \*/

 // The circuit:

 // \* analog pin 0 - throttle/governor potentiometer signal (middle/white wire).

 // Center pin of the potentiometer goes to the analog pin.

 // side pins of the potentiometer go to +5V (red) and ground (black).

 // \* analog pin 2 - steering trim potentiometer signal (middle/white wire).

 // Center pin of the potentiometer goes to the analog pin.

 // side pins of the potentiometer go to +5V (red) and ground (black).

 // \* Digital pin 2 - Emergency Stop button

 // \* Digital pin 3 - Forward button

 // \* Digital pin 4 - Reverse button

 // \* Digital pin 5 - Left turn button

 // \* Digital pin 6 - Right turn Button

 // \* Digital pin 11 - Serial output to Sabertooth

 #include <Servo.h>

 #include <SoftwareSerial.h>

 Servo Sabertooth; // We'll name the Sabertooth object Sabertooth.

 // For how to configure the Sabertooth, see the DIP Switch Wizard for

 // http://www.dimensionengineering.com/datasheets/SabertoothDIPWizard/start.htm

 // Be sure to select Simplified Serial Mode for use with this library.

 // This sample uses a baud rate of 9600.

 //

 // Connections to make:

 // Arduino D11 -> Sabertooth S1

 // Note: Serial output to Sabertooth is moved from TX01 (default) to D11 for easier debugging.

 // Arduino GND -> Sabertooth 0V

 // Arduino VIN -> Sabertooth 5V (OPTIONAL, if you want the Sabertooth to power the Arduino)

 // D2 -> Emergency Stop Signal |

 // D3 -> Forward Button Signal | The other wire from each of the digital input

 // D4 -> Reverse Button Signal | buttons will be connected to a ground bus, which

 // D5 -> Left Button Signal | will in turn be connected to the gnd pin of the

 // D6 -> Right Button Signal | Arduino.

 //

 // 0V -> Governor/potentiometer lead 1 - black (ground)

 // A0 -> Governor/potentiometer lead 2 - white (signal)

 // 5v -> Governor/potentiometer lead 3 - red (+)

 //

 // 0V -> Trim/potentiometer lead 1 - black (ground)

 // A2 -> Trim/potentiometer lead 2 - white (signal)

 // 5v -> Trim/potentiometer lead 3 - red (+)

 SoftwareSerial mySerial(10, 11); // RX, TX

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//\*\*\* Adjustable parameters: \*\*

const byte ndebug = 0; //\*\*\* 0=run (debug messages off) 1=debug messages on \*\*

const byte accelFactor = 2; //\*\*\* accelerate increment \*\*

const byte decelFactor = 8; //\*\*\* decelerate increment \*\*

const byte trimSens = 6;//\*\*\* trim sensitivity: 128 = highest sensitivity \*\*

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// These constants shouldn't change.

const byte deadStop1 = 64; // command value at center/stopped for motor 1

const byte deadStop2 = 192; // command value at center/stopped for motor 2

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Initialize work variables:

int motorControl1 = deadStop1;

int motorControl2 = deadStop2;

byte motorByte1 = 0;

byte motorByte2 = 0;

byte rawSpeed1 = 0;

byte rawSpeed2 = 0;

int speedDifference = 0;

byte direction1 = 1; // '1' or '0' (Forward or Reverse) are valid values;

byte direction2 = 1;

byte buttonState = 0; // value read from the button

byte buttonPinNumber; // input pin associated with this button

byte buttonPressedIndicator = 0; // Flag to indicate if any button pressed or not

float potValue = 0; // value read from the potentiometer

int trimAdjustment = 0;

// These values are the max command values as determined by the throttle (potentiomemter) setting

byte maxFwdSpeed1 = 0;

byte maxFwdSpeed2 = 0;

byte maxRevSpeed1 = 0;

byte maxRevSpeed2 = 0;

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void setup()

{

 delay(2000); // Give the Sabertooth time to intialize.

 // initialize serial communications at 9600 bps:

 Serial.begin(9600);

 mySerial.begin(9600);

 //Note: Pin 1 is the Arduino serial output which connects to the Sabertooth S1 terminal.

 // Define the input pins (which correspond to button jacks)

 pinMode(2, INPUT\_PULLUP); // Emergency Stop

 digitalWrite(2, HIGH);

 pinMode(3, INPUT\_PULLUP); // Forward

 digitalWrite(3, HIGH);

 pinMode(4, INPUT\_PULLUP); // Reverse

 digitalWrite(4, HIGH);

 pinMode(5, INPUT\_PULLUP); // Left

 digitalWrite(5, HIGH);

 pinMode(6, INPUT\_PULLUP); // Right

 digitalWrite(6, HIGH);

 Sabertooth.attach(1);

};

void loop()

{

 // check throttle potentiometer setting:

 potValue = analogRead(A0);

 // set max command values based on throttle (potentiometer) setting

 maxRevSpeed1 = map(potValue, 0,1023,63,1);

 maxFwdSpeed1 = map(potValue, 0,1023,65,127);

 maxRevSpeed2 = map(potValue, 0,1023,191,128);

 maxFwdSpeed2 = map(potValue, 0,1023,193,255);

 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 // check steering trim potentiometer setting:

 potValue = analogRead(A2);

 // set steering correction based on trim (potentiometer) setting

 trimAdjustment = map(potValue, 0,1023,(-1 \* trimSens),trimSens);

 if (ndebug == 1){

 Serial.print("\t potValue = " );

 Serial.println(potValue);

 Serial.print("\t trimAdjustment = " );

 Serial.println(trimAdjustment);

 };

 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 if (ndebug == 1){

 // print the results to the serial monitor:

 Serial.print("\t MRS1 = " );

 Serial.print(maxRevSpeed1);

 Serial.print("\t MFS1 = " );

 Serial.println(maxFwdSpeed1);

 Serial.print("\t MRS2 = " );

 Serial.print(maxRevSpeed2);

 Serial.print("\t MFS2 = " );

 Serial.println(maxFwdSpeed2);

 };

 buttonPressedIndicator = 0; // reset button pressed indicator

 // poll the motion buttons (pins 2-6):

 // Note that having E-stop assigned to pin #2 with break after processing ensures that it will

 // always have first priority.

 for (buttonPinNumber = 2; buttonPinNumber <= 6; ++buttonPinNumber)

 {

 buttonState = digitalRead(buttonPinNumber);

 if (buttonState == LOW)

 {

 buttonPressedIndicator = ++buttonPressedIndicator;

 buttonPressedFunction(buttonPinNumber);

 if (ndebug == 1)

 {

 Serial.println("Button pressed - Breaking loop \*\*\*" );

 };

 break;

 };

 };

 // Compute equivalent motor speeds so that left vs right can be compared.

 if ((motorControl1 == 0 or motorControl1 == deadStop1) and (motorControl2 == 0 or motorControl2 == deadStop2))

 { //1

 rawSpeed1 = 0;

 rawSpeed2 = 0;

 if (ndebug == 1)

 {

 Serial.println("rawSpeeds = 0." );

 };

 } //1

 else

 { //1

 if (ndebug == 1)

 {

 Serial.println("Calculating rawSpeeds." );

 Serial.print("\t");

 Serial.print("MC1 = ");

 Serial.print(motorControl1);

 Serial.print("\t");

 Serial.print("MC2 = ");

 Serial.println(motorControl2);

 };

 rawSpeed1 = abs(deadStop1 - motorControl1);

 rawSpeed2 = abs(deadStop2 - motorControl2);

 if (motorControl1 >= deadStop1)

 { //2

 direction1 = '1';

 } //2

 else

 { //2

 direction1 = '0';

 }; //2

 if (motorControl2 >= deadStop2)

 { //2

 direction2 = '1';

 } //2

 else

 { //2

 direction2 = 0;

 }; //2

 if (ndebug == 1)

 {

 Serial.print("\t");

 Serial.print("rawSpeed1: ");

 Serial.print(rawSpeed1);

 Serial.print("\t");

 Serial.print("dir 1: ");

 Serial.print(direction1);

 Serial.print("\t");

 Serial.print("rawSpeed2: ");

 Serial.print(rawSpeed2);

 Serial.print("\t");

 Serial.print("dir 2: ");

 Serial.println(direction2);

 };

 }; //1

 if (ndebug == 1)

 {

 Serial.print("MFS1 - MC1: ");

 Serial.print("\t");

 Serial.println(maxFwdSpeed1 - motorControl1);

 Serial.print("maxFwdSpeed2 - motorControl2: ");

 Serial.print("\t");

 Serial.println(maxFwdSpeed2 - motorControl2);

 };

 if (buttonPressedIndicator == 0)

 { // 0 If no buttons have been pressed decelerate.

 if ((motorControl1 == 0 or motorControl1 == deadStop1) and (motorControl2 == 0 or motorControl2 == deadStop2))

 { //1

 motorControl1 = deadStop1;

 motorControl2 = deadStop2;

 if (ndebug == 1)

 {

 Serial.println("Stopped." );

 };

 } //1

 else

 { //1

 if (ndebug == 1)

 {

 Serial.println("No button pressed - decelerate." );

 };

 decelerateFunction();

 }; //1

 sendCommandFunction();

 }; // 0

 // This just blinks the LED once to indicate the completion of each iteration of the loop.

 digitalWrite(13, HIGH); // turn the LED on (HIGH is the voltage level)

 delay(150); // wait for 150 millseconds

 digitalWrite(13, LOW); // turn the LED off by making the voltage LOW

 // wait 20 milliseconds before the next loop

 // for the analog-to-digital converter to settle

 // after the last reading:

 //delay(20);

 if (ndebug == 1)

 {

 Serial.println("End Void Loop \*\*\*\*\*" );

 // delay(100);

 };

}; // End of Void Loop \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Functions: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Send Command:

 void sendCommandFunction(){

 if (ndebug == 1)

 {

 Serial.println("Send Command function \*\*\*\*\*" );

 };

 // Ensure that command values are within valid range:

 if (motorControl1 < 1)

 {

 motorControl1 = 1;

 };

 if (motorControl1 > 127)

 {

 motorControl1 = 127;

 };

 if (motorControl2 < 128)

 {

 motorControl2 = 128;

 };

 if (motorControl2 > 255)

 {

 motorControl2 = 255;

 };

 if (ndebug == 1)

 {

 Serial.print("\t");

 Serial.print("MC1: " );

 Serial.print(motorControl1);

 Serial.print("\t");

 Serial.print("MC2: " );

 Serial.println(motorControl2);

 };

 // convert command values to byte format for serial communication

 motorByte1 = motorControl1;

 motorByte2 = motorControl2 ;

 // Trim correction to track straight in forward or reverse

 if (rawSpeed1 > abs(trimAdjustment)) // this ensures that trim correction won't cause motor to overshoot zero.

 {

 if (ndebug == 1)

 {

 Serial.print("Trim Adjust - Pin: ");

 Serial.println(buttonPinNumber);

 };

 switch (buttonPinNumber) {

 case 3: //Forward motion

 if (trimAdjustment < 0) // left trim - slow down motor 1

 {

 if (ndebug == 1)

 {

 Serial.println("Fwd motion - left trim");

 };

 motorByte1 = motorControl1 + trimAdjustment;

 motorByte2 = motorControl2 ;

 }

 else // right trim - slow down motor 2

 {

 if (ndebug == 1)

 {

 Serial.println("Fwd motion - right trim");

 };

 motorByte1 = motorControl1 ;

 motorByte2 = motorControl2 - trimAdjustment;

 };

 break;

 case 4: // Reverse motion

 if (trimAdjustment < 0) // left trim - slow motor 1 (increase value)

 {

 if (ndebug == 1)

 {

 Serial.println("Rev motion - left trim");

 }; motorByte1 = motorControl1 - trimAdjustment;

 motorByte2 = motorControl2 ;

 }

 else // right trim - slow motor 2 (increase value)

 {

 if (ndebug == 1)

 {

 Serial.println("Rev motion - right trim");

 };

 motorByte1 = motorControl1 ;

 motorByte2 = motorControl2 + trimAdjustment;

 };

 break;

 }; //End switch

 };

 if (ndebug == 1)

 {

 Serial.print("\t");

 Serial.print("motorByte1: " );

 Serial.print(motorByte1);

 Serial.print("\t");

 Serial.print("motorByte2: " );

 Serial.println(motorByte2);

 };

 //Send motor commands to Sabertooth

 mySerial.write(motorByte1);

 mySerial.write(motorByte2);

 if (ndebug == 1)

 {

 Serial.println(" ");

 Serial.println("End Send Command Function \*\*\*\*\*\*\*");

 };

 };

// End of Send Command Function \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Button pressed:

 void buttonPressedFunction(int buttonPinNumber){ // 0

 buttonState = digitalRead(buttonPinNumber);

 if (ndebug == 1){

 Serial.println("buttonPressedFunction: \*\*\*\*\*" );

 Serial.print("pin: " );

 Serial.print(buttonPinNumber);

 Serial.print("\t");

 Serial.print("buttonState = " );

 Serial.println(buttonState);

 };

 if (motorControl1 == 0)

 {

 motorControl1 = deadStop1;

 };

 if (motorControl2 == 0)

 {

 motorControl2 = deadStop2;

 };

 switch (buttonPinNumber) { //1

 // 1. Determine which button was pressed

 // 2. Compare current motor speeds and directions.

 // 3. Take appropriate action.

 case 2: // Emergency Stop \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 motorControl1 = deadStop1;

 motorControl2 = deadStop2;

 if (ndebug == 1)

 {

 Serial.println("E-stop sent.");

 };

 break;

 case 3: // Forward button pressed\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

 if (ndebug == 1)

 {

 Serial.println("Accel forward.");

 };

 accelerateForward();

 break;

 case 4: // Reverse \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 if (ndebug == 1)

 {

 Serial.println("Accel in reverse.");

 };

 accelerateReverse();

 break;

 case 5: // Left \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 if (ndebug == 1)

 {

 Serial.println("Left button pressed");

 };

 leftButtonPressed();

 break;

 case 6: // Right button pressed. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 if (ndebug == 1)

 {

 Serial.println("Right button pressed.");

 };

 rightButtonPressed();

 break;

 }; //1 End Switch \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 sendCommandFunction();

 if (ndebug == 1)

 {

 Serial.println("End of buttonPressFunction \*\*\*\*\*\*");

 };

 digitalWrite(buttonPinNumber, HIGH);

 }; // 0

 // End of buttonPressedFunction \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 // Accelerate Forward \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 void accelerateForward(){ //0

 if (ndebug == 1)

 {

 Serial.println("accelerateForward function");

 };

 speedDifference = abs((maxFwdSpeed1 - motorControl1) - (maxFwdSpeed2 - motorControl2)); // compare motor speeds

 if (speedDifference != 0) // speeds mismatched

 { //1

 if (ndebug == 1)

 {

 Serial.println("Motor spd mismatch");

 };

 if (speedDifference < accelFactor) // speeds mismatched by < accelFactor

 { //2

 if (ndebug == 1)

 {

 Serial.print("Spd mismatch = ");

 Serial.println(speedDifference);

 };

 if ((maxFwdSpeed1 - motorControl1) > (maxFwdSpeed2 - motorControl2)) // left motor slow

 { //3

 motorControl1 = motorControl1 + speedDifference; //speed up motor 1 to match motor 2

 if (ndebug == 1)

 {

 Serial.println("MC1 adjusted");

 };

 } //3

 else

 { //3

 motorControl2 = motorControl2 + speedDifference;

 if (ndebug == 1)

 {

 Serial.println("MC2 adjusted");

 };

 }; //3

 }; //2

 }; //1

 // Motor speeds equal \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 if (speedDifference == 0)

 { // 1

 if (ndebug == 1)

 {

 Serial.println("rawSpeed1 == rawSpeed2");

 };

 if (motorControl1 < deadStop1)

 {

 motorControl1 = (motorControl1 + decelFactor);

 }

 else

 {

 motorControl1 = (motorControl1 + accelFactor);

 };

 if (motorControl1 == deadStop1) // Skip dead spot.

 { //3

 motorControl1 = (motorControl1 + accelFactor);

 if (ndebug == 1)

 {

 Serial.println("Motor1 - skip dead spot");

 };

 }; //3

 if (motorControl1 > maxFwdSpeed1) // Check for overspeed

 { //3

 motorControl1 = maxFwdSpeed1;

 if (ndebug == 1)

 {

 Serial.println("Motor1 - Governed to MFS1");

 Serial.print("motorControl2 = ");

 Serial.println(motorControl2);

 };

 }; //3

 if (motorControl2 < deadStop2)

 {

 motorControl2 = (motorControl2 + decelFactor);

 }

 else

 {

 motorControl2 = (motorControl2 + accelFactor);

 };

 if (motorControl2 == deadStop2) // Skip dead spot.

 { //3

 motorControl2 = (motorControl2 + accelFactor);

 if (ndebug == 1)

 {

 Serial.println("Motor2 - skip dead spot");

 };

 }; //3

 if (motorControl2 > maxFwdSpeed2) // Check for overspeed

 { //3

 motorControl2 = maxFwdSpeed2;

 if (ndebug == 1)

 {

 Serial.println("Motor2 - Governed to MFS2");

 };

 }; //3

 return;

 }; //1

/\* else

 { //2

 if (motorControl1 > maxFwdSpeed1) // Check for overspeed

 {

 motorControl1 = maxFwdSpeed1;

 motorControl2 = maxFwdSpeed2;

 };

 return;

 }; //2

 }; //1 \*/

 // Left motor slow \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 if ((maxFwdSpeed1 - motorControl1) > (maxFwdSpeed2 - motorControl2))

 { //1

 if (ndebug == 1)

 {

 Serial.println("motor 1 slow");

 };

 motorControl1 = (motorControl1 + accelFactor);

 if (motorControl1 == deadStop1) // Skip dead spot.

 { //2

 motorControl1 = (motorControl1 + accelFactor);

 }; //2

 if (motorControl1 > maxFwdSpeed1) // Check for overspeed.

 { //2

 motorControl1 = maxFwdSpeed1;

 }; //2

 return;

 } //1

 else

 { //1 - right motor slow

 if (ndebug == 1)

 {

 Serial.println("motor 2 slow");

 };

 motorControl2 = (motorControl2 + accelFactor);

 if (motorControl2 == deadStop2) // Skip dead spot.

 { //2

 motorControl2 = (deadStop2 + accelFactor);

 }; //2

 if (motorControl2 > maxFwdSpeed2) // Check for overspeed.

 { //2

 motorControl2 = maxFwdSpeed2;

 }; //2

 return;

 }; //1

 }; //0

 // End of accelerateForward \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 // Accelerate Reverse \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 void accelerateReverse(){ //0

 if (ndebug == 1)

 {

 Serial.println("accelerateReverse function");

 };

 if ((maxFwdSpeed1 - motorControl1) == (maxFwdSpeed2 - motorControl2))

 { // 1

 if (ndebug == 1)

 {

 Serial.println("rawSpeed1 == rawSpeed2");

 };

 if (motorControl1 > deadStop2)

 {

 motorControl1 = (motorControl1 - decelFactor);

 }

 else

 {

 motorControl1 = (motorControl1 - accelFactor);

 };

 if (motorControl1 == deadStop1) // Skip dead spot.

 {

 motorControl1 = (deadStop1 - accelFactor);

 };

 if (motorControl1 < maxRevSpeed1)

 {

 motorControl1 = maxRevSpeed1;

 };

 if (motorControl2 > deadStop2)

 {

 motorControl2 = (motorControl2 - decelFactor);

 }

 else

 {

 motorControl2 = (motorControl2 - accelFactor);

 };

 if (motorControl2 == deadStop2) // Skip dead spot.

 {

 motorControl2 = (deadStop2 - accelFactor);

 };

 if (motorControl2 < maxRevSpeed2)

 {

 motorControl2 = maxRevSpeed2;

 };

 return;

 };

 // Left motor fast \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 if ((maxFwdSpeed1 - motorControl1) < (maxFwdSpeed2 - motorControl2))

 { //1

 if (ndebug == 1)

 {

 Serial.println("left motor fast");

 };

 motorControl1 = (motorControl1 - accelFactor);

 if (motorControl1 == deadStop1) // Skip dead spot.

 { //4

 motorControl1 = (deadStop1 - accelFactor);

 }; //4

 if (motorControl1 < maxRevSpeed1) //Check for overspeed

 {

 motorControl1 = maxRevSpeed1;

 };

 return;

 } //1

 else

 { //1 Left motor slow

 motorControl2 = (motorControl2 - accelFactor);

 if (motorControl2 == deadStop2) // Skip dead spot.

 { //3

 motorControl2 = (deadStop2 - accelFactor);

 }; //3

 if (motorControl2 < maxRevSpeed2) //Check for overspeed

 {

 motorControl2 = maxRevSpeed2;

 };

 }; //1

 return;

 }; //0

 //End of accelerateReverse Function \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 // leftButtonPressed Function \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 void leftButtonPressed(){

 if (ndebug == 1)

 {

 Serial.println("leftButtonPressed function");

 };

 if (rawSpeed1 == rawSpeed2)

 { //1

 if (ndebug == 1)

 {

 Serial.println("rawSpeed1 == rawSpeed2");

 Serial.print(rawSpeed1);

 Serial.print("\t");

 Serial.println(rawSpeed1);

 };

 if ((direction1 != direction2) or (rawSpeed1 == 0))

 { //2

 if (ndebug == 1)

 {

 if (direction1 != direction2)

 {

 Serial.println("dir 1 != dir 2");

 };

 if (rawSpeed1 == 0)

 {

 Serial.println("rawSpeed1 == 0");

 };

 };

 motorControl1 = (motorControl1 - accelFactor);

 motorControl2 = (motorControl2 + accelFactor);

 if (motorControl1 == deadStop1) // Skip dead spot

 { //3

 motorControl1 = (deadStop1 - accelFactor);

 }; //3

 if (motorControl1 < maxRevSpeed1) //Check for overspeed

 {

 motorControl1 = maxRevSpeed1;

 };

 if (motorControl2 == deadStop2) // Skip dead spot.

 { //3

 motorControl2 = (deadStop2 + accelFactor);

 }; //3

 if (motorControl2 > maxFwdSpeed2) //Check for overspeed

 { //3

 motorControl2 = maxFwdSpeed2;

 }; //3

 return;

 }; //2

 if (direction1 == direction2)

 { //2

 if (ndebug == 1)

 {

 Serial.print("dir 1 == dir 2: ");

 Serial.println(direction2);

 };

 if (direction1 == 1)

 { //3

 motorControl1 = (motorControl1 - decelFactor);

 if (motorControl1 == deadStop1) // Skip dead spot

 { //4

 motorControl1 = (deadStop1 - decelFactor);

 }; //4

 if (motorControl1 < maxRevSpeed1) //Check for overspeed

 { //4

 motorControl1 = maxRevSpeed1;

 }; //4

 return;

 } //3

 else // direction = 0

 { //3

 motorControl2 = (motorControl2 + decelFactor);

 if (motorControl2 == deadStop2) // Skip dead spot

 { //4

 motorControl2 = (deadStop2 + decelFactor);

 }; //5

 if (motorControl2 > maxFwdSpeed2) //Check for overspeed

 { //5

 motorControl2 = maxFwdSpeed2;

 }; //4

 return;

 }; //3

 }; //2

 if (direction1 != direction2)

 { //2

 if (motorControl2 < maxFwdSpeed2)

 { //3

 motorControl2 = (motorControl2 + accelFactor);

 if (motorControl2 == deadStop2) // Skip dead spot.

 { //4

 motorControl2 = (deadStop2 + accelFactor);

 }; //4

 if (motorControl2 > maxFwdSpeed2) //Check for overspeed

 { //4

 motorControl2 = maxFwdSpeed2;

 }; //4

 return;

 }; //3

 }; //2

 } // 1

 else

 { // 1 rawSpeed1 != rawSpeed2

 if (rawSpeed1 < rawSpeed2)

 { //2

 if (ndebug == 1)

 {

 Serial.println("rawSpeed1 < rawSpeed2");

 };

 if (motorControl1 > maxRevSpeed1)

 { //3

 motorControl1 = (motorControl1 - accelFactor);

 if (motorControl1 == deadStop1) // Skip dead spot.

 { //4

 motorControl1 = (deadStop1 - accelFactor);

 }; //4

 if (motorControl1 < maxRevSpeed1) //Check for overspeed

 { //4

 motorControl1 = maxRevSpeed1;

 }; //4

 }; //3

 return;

 }; //2

 if (rawSpeed1 > rawSpeed2)

 { //2

 if (ndebug == 1)

 {

 Serial.println("rawSpeed1 > rawSpeed2");

 };

 if (motorControl2 < maxFwdSpeed2)

 {//3

 motorControl2 = (motorControl2 + accelFactor);

 if (motorControl2 == deadStop2) // Skip dead spot.

 { //4

 motorControl2 = (deadStop2 + accelFactor);

 }; //4

 if (motorControl2 > maxFwdSpeed2) //Check for overspeed

 { //4

 motorControl2 = maxRevSpeed2;

 }; //4

 }; //3

 return;

 }; //2

 }; //1

 if (ndebug == 1)

 {

 Serial.println("No action taken.");

 };

 }; //0 End of leftButtonPressed Functiion \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 // rightButtonPressed Functiion \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 void rightButtonPressed(){ //0

 if (rawSpeed1 == rawSpeed2)

 { //1

 if (ndebug == 1)

 {

 Serial.println("rawSpeed1 == rawSpeed2");

 Serial.print(rawSpeed1);

 Serial.print("\t");

 Serial.println(rawSpeed1);

 };

 if ((direction1 != direction2) or (rawSpeed1 == 0))

 { //2

 if (ndebug == 1)

 {Serial.println("dir 1 != dir 2");};

 if (motorControl1 < maxFwdSpeed1)

 { //3

 motorControl1 = (motorControl1 + accelFactor);

 motorControl2 = (motorControl2 - accelFactor);

 if (motorControl1 == 0 or motorControl1 == deadStop1) // Skip dead spot.

 { //4

 motorControl1 = (deadStop1 + accelFactor);

 }; //4

 if (motorControl2 == 0 or motorControl2 == deadStop2) // Skip dead spot.

 { //4

 motorControl2 = (deadStop2 - accelFactor);

 }; //4

 }; //3

 if (ndebug == 1)

 {

 Serial.println("Break");

 };

 return;

 }; //2

 if (direction1 == direction2)

 { //2

 if (ndebug == 1)

 {

 Serial.print("dir 1 == dir 2 : ");

 Serial.println(direction2);

 };

 if (direction1 == 0)

 { //3

 if (motorControl1 < maxFwdSpeed1)

 {//4

 motorControl1 = (motorControl1 + accelFactor);

 if (motorControl1 == 0 or motorControl1 == deadStop1) // Skip dead spot.

 { //5

 motorControl1 = (deadStop1 + accelFactor);

 }; //5

 };//4

 return;

 } //3

 else

 { //3

 if (motorControl2 > maxRevSpeed2)

 { //4

 motorControl2 = (motorControl2 - accelFactor);

 if (motorControl2 == 0 or motorControl2 == deadStop2) // Skip dead spot.

 { //5

 motorControl2 = (deadStop2 - accelFactor);

 }; //5

 if (motorControl2 < maxRevSpeed2) // Govern

 { //5

 motorControl2 = maxRevSpeed2;

 }; //5

 return;

 }; //4

 }; //3

 }; //2

 } // 1

 else

 { // 1 rawSpeed1 != rawSpeed2

 if (rawSpeed1 < rawSpeed2)

 { //2

 if (ndebug == 1)

 {Serial.println("rawSpeed1 < rawSpeed2");};

 if (motorControl1 < maxFwdSpeed1)

 { //3

 motorControl1 = (motorControl1 + accelFactor);

 if (motorControl1 == 0 or motorControl1 == deadStop1) // Skip dead spot.

 { //4

 motorControl1 = (deadStop1 + accelFactor);

 }; //4

 return;

 }; //3

 }; //2

 if (rawSpeed1 > rawSpeed2)

 { //2

 if (ndebug == 1)

 {Serial.println("rawSpeed1 > rawSpeed2");};

 if (motorControl2 > maxRevSpeed2)

 {//3

 motorControl2 = (motorControl2 - accelFactor);

 if (motorControl2 == deadStop2) // Skip dead spot.

 { //4

 motorControl2 = (deadStop2 - accelFactor);

 }; //4

 if (motorControl2 < maxRevSpeed2) // Govern

 { //4

 motorControl2 = maxRevSpeed2;

 }; //4

 return;

 }; //3

 }; //2

 }; //1

 if (ndebug == 1)

 {Serial.println("No action taken.");};

 }; //0 End of rightButtonPressed Function \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 // Decelerate:

 void decelerateFunction(){

 if (ndebug == 1)

 {

 Serial.println("Decel function" );

 };

 if (motorControl1 > deadStop1)

 { //3

 motorControl1 = (motorControl1 - decelFactor);

 if (motorControl1 < deadStop1) // Makes sure that command value doesn't overshoot dead stop value.

 { //4

 motorControl1 = deadStop1;

 }; //4

 }; //3

 if (motorControl1 < deadStop1)

 { //3

 motorControl1 = (motorControl1 + decelFactor);

 if (motorControl1 > deadStop1) // Makes sure that command value doesn't overshoot dead stop value.

 { //4

 motorControl1 = deadStop1;

 }; //4

 }; //3

 if (motorControl2 > deadStop2)

 { //3

 motorControl2 = (motorControl2 - decelFactor);

 if (motorControl2 < deadStop2) // Makes sure that command value doesn't overshoot dead stop value.

 { //4

 motorControl2 = deadStop2;

 }; //4

 }; //3

 if (motorControl2 < deadStop2)

 { //3

 motorControl2 = (motorControl2 + decelFactor);

 if (motorControl2 > deadStop2) // Makes sure that command value doesn't overshoot dead stop value.

 { //4

 motorControl2 = deadStop2;

 }; //4

 }; //3

 }; //2

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*