#include <Arduino.h>

#include <stdint.h>

#include "SCMD.h"

#include "SCMD\_config.h" //Contains #defines for common SCMD register names and values

#include "Wire.h"

const int buttonPin = 32;

const int buttonPinI = 14;

int buttonPushCounter = 0; // counter for the number of button presses

int buttonState = 0; // current state of the button

int lastButtonState = 0; // previous state of the button

int buttonPushCounterI = 0; // counter for the number of button presses

int buttonStateI = 0; // current state of the button

int lastButtonStateI = 0; // previous state of the button

SCMD myMotorDriver; //This creates the main object of one motor driver and connected peripherals.

int rk=1;

void setup()

{

pinMode(8, INPUT\_PULLUP); //Use to halt motor movement (ground)

pinMode(buttonPin, INPUT\_PULLUP);

pinMode(buttonPinI, INPUT\_PULLUP);

Serial.begin(9600);

Serial.println("Starting sketch.");

//\* Configure the Motor Driver's Settings \*//

// .commInter face is I2C\_MODE

myMotorDriver.settings.commInterface = I2C\_MODE;

// set address if I2C configuration selected with the config jumpers

myMotorDriver.settings.I2CAddress = 0x5D; //config pattern is "1000" (default) on board for address 0x5D

// set chip select if SPI selected with the config jumpers

myMotorDriver.settings.chipSelectPin = 10;

//\*initialize the driver get wait for idle\*//

while ( myMotorDriver.begin() != 0xA9 ) //Wait until a valid ID word is returned

{

Serial.println( "ID mismatch, trying again" );

delay(500);

}

Serial.println( "ID matches 0xA9" );

// Check to make sure the driver is done looking for peripherals before beginning

Serial.print("Waiting for enumeration...");

while ( myMotorDriver.ready() == false );

Serial.println("Done.");

Serial.println();

//\*Set application settings and enable driver\*//

//Uncomment code for motor 0 inversion

//while( myMotorDriver.busy() );

//myMotorDriver.inversionMode(0, 1); //invert motor 0

//Uncomment code for motor 1 inversion

while ( myMotorDriver.busy() ); //Waits until the SCMD is available.

myMotorDriver.inversionMode(1, 1); //invert motor 1

while ( myMotorDriver.busy() );

myMotorDriver.enable(); //Enables the output driver hardware

}

#define left\_MOTOR 0

#define right\_MOTOR 1

void loop()

{

while(rk==1)

{

// read the pushbutton input pin:

buttonState = digitalRead(buttonPin);

// compare the buttonState to its previous state

if (buttonState != lastButtonState)

{

// if the state has changed, increment the counter

if (buttonState == HIGH)

{

// if the current state is HIGH then the button went from off to on:

buttonPushCounter++;

Serial.println("on");

Serial.print("number of button pushes: ");

Serial.println(buttonPushCounter);

}

else

{

// if the current state is LOW then the button went from on to off:

Serial.println("off");

}

// Delay a little bit to avoid bouncing

}

// save the current state as the last state, for next time through the loop

lastButtonState = buttonState;

// turns on the LED every four button pushes by checking the modulo of the

// button push counter. the modulo function gives you the remainder of the

// division of two numbers:

if (buttonPushCounter % 2 == 0)

{

rk=1;

myMotorDriver.setDrive( left\_MOTOR, 1, 200); // 0 reverse

myMotorDriver.setDrive( right\_MOTOR, 0, 200); // upper // my side

Serial.println("X motor");

}

else

{

myMotorDriver.setDrive( left\_MOTOR, 0, 200); // 0 reverse

myMotorDriver.setDrive( right\_MOTOR, 1, 200); // upper // my side

Serial.println("Stop");

delay(2200);

myMotorDriver.setDrive( left\_MOTOR, 0, 0); // black side upper

myMotorDriver.setDrive( right\_MOTOR, 0, 0); // black side upper

rk++;

}

}

while(rk==2)

{

// read the pushbutton input pin:

buttonStateI = digitalRead(buttonPinI);

// compare the buttonState to its previous state

if (buttonStateI != lastButtonStateI)

{

// if the state has changed, increment the counter

if (buttonStateI == HIGH)

{

// if the current state is HIGH then the button went from off to on:

buttonPushCounterI++;

Serial.println("on");

Serial.print("number of button pushes: ");

Serial.println(buttonPushCounterI);

}

else

{

// if the current state is LOW then the button went from on to off:

Serial.println("off");

}

// Delay a little bit to avoid bouncing

}

// save the current state as the last state, for next time through the loop

lastButtonStateI = buttonStateI;

// turns on the LED every four button pushes by checking the modulo of the

// button push counter. the modulo function gives you the remainder of the

// division of two numbers:

if (buttonPushCounterI % 2 == 0)

{

rk=2;

myMotorDriver.setDrive( left\_MOTOR, 1, 200); // outside

myMotorDriver.setDrive( right\_MOTOR, 1, 200); // upper

Serial.println("X motor");

}

else

{

myMotorDriver.setDrive( left\_MOTOR, 0, 200); // outside

myMotorDriver.setDrive( right\_MOTOR, 0, 200); // upper

Serial.println("Stop");

delay(800);

myMotorDriver.setDrive( left\_MOTOR, 0, 0); // black side upper

myMotorDriver.setDrive( right\_MOTOR, 0, 0); // black side upper

rk++;

}

}

delay(10000);

while(rk==3)

{

myMotorDriver.setDrive( left\_MOTOR, 1, 200); // inside 1

myMotorDriver.setDrive( right\_MOTOR, 0, 200); //

delay(800);

myMotorDriver.setDrive( left\_MOTOR, 0, 0); // stop

myMotorDriver.setDrive( right\_MOTOR, 0, 0); //

delay(1000);

myMotorDriver.setDrive( left\_MOTOR, 0, 200); // 2

myMotorDriver.setDrive( right\_MOTOR, 0, 200); //

delay(800);

myMotorDriver.setDrive( left\_MOTOR, 0, 0); // stop

myMotorDriver.setDrive( right\_MOTOR, 0, 0); //

delay(1000);

myMotorDriver.setDrive( left\_MOTOR, 0, 200); // inside 3

myMotorDriver.setDrive( right\_MOTOR, 1, 200); //

delay(800);

myMotorDriver.setDrive( left\_MOTOR, 0, 0); // stop

myMotorDriver.setDrive( right\_MOTOR, 0, 0); //

delay(1000);

myMotorDriver.setDrive( left\_MOTOR, 1, 200); // 4

myMotorDriver.setDrive( right\_MOTOR, 1, 200); //

delay(800);

myMotorDriver.setDrive( left\_MOTOR, 0, 0); // stop

myMotorDriver.setDrive( right\_MOTOR, 0, 0); //

delay(1000);

myMotorDriver.setDrive( left\_MOTOR, 0, 0); // 5

myMotorDriver.setDrive( right\_MOTOR, 0, 185); //

delay(1700);

myMotorDriver.setDrive( left\_MOTOR, 0, 0); // stop

myMotorDriver.setDrive( right\_MOTOR, 0, 0); //

delay(1000);

myMotorDriver.setDrive( left\_MOTOR, 0, 187); // 6

myMotorDriver.setDrive( right\_MOTOR, 0, 0); //

delay(885);

myMotorDriver.setDrive( left\_MOTOR, 0, 0); // stop

myMotorDriver.setDrive( right\_MOTOR, 0, 0); //

delay(1000);

myMotorDriver.setDrive( left\_MOTOR, 0, 0); // 7

myMotorDriver.setDrive( right\_MOTOR, 1, 187); //

delay(885);

myMotorDriver.setDrive( left\_MOTOR, 0, 0); // stop

myMotorDriver.setDrive( right\_MOTOR, 0, 0); //

delay(1000);

myMotorDriver.setDrive( left\_MOTOR, 1, 188); // 8

myMotorDriver.setDrive( right\_MOTOR, 0, 0); //

delay(1700);

myMotorDriver.setDrive( left\_MOTOR, 0, 0); // stop

myMotorDriver.setDrive( right\_MOTOR, 0, 0); //

delay(1000);

rk++;

}

}