//Modified from examples by Pat McMahon 8/11/2020

//A043

// Use the MD\_MAX72XX library to scroll text on the display

//Copy the code from patsrobots.com go to A043 copy and paste it into the Arduino IDE, alter for the number of Devices (8x8) you have and type your own message to display.

// Demonstrates the use of the call back function to control what

// is scrolled on the display text.

//

// User can enter text on the serial monitor and this will display as a

// scrolling message on the display.

// Speed for the display is controlled by a pot on SPEED\_IN analog in.

//

#include <MD\_MAX72xx.h>

#include <SPI.h>

#define USE\_POT\_CONTROL 1

#define PRINT\_CALLBACK 0

#define PRINT(s, v) { Serial.print(F(s)); Serial.print(v); }

// Define the number of devices we have in the chain and the hardware interface

// NOTE: These pin numbers will probably not work with your hardware and may

// need to be adapted

#define HARDWARE\_TYPE MD\_MAX72XX::DR1CR0RR0\_HW

#define MAX\_DEVICES 20

#define CLK\_PIN 13 // or CLK

#define DATA\_PIN 11 // or DIN

#define CS\_PIN 10 // or CS

// SPI hardware interface

MD\_MAX72XX mx = MD\_MAX72XX(HARDWARE\_TYPE, CS\_PIN, MAX\_DEVICES);

// Arbitrary pins

//MD\_MAX72XX mx = MD\_MAX72XX(HARDWARE\_TYPE, DATA\_PIN, CLK\_PIN, CS\_PIN, MAX\_DEVICES);

// Scrolling parameters

#if USE\_POT\_CONTROL

#define SPEED\_IN A0

#else

#define SCROLL\_DELAY 75 // in milliseconds

#endif // USE\_POT\_CONTROL

#define CHAR\_SPACING 1 // pixels between characters

// Global message buffers shared by Serial and Scrolling functions

#define BUF\_SIZE 300

char curMessage[BUF\_SIZE];

char newMessage[BUF\_SIZE];

bool newMessageAvailable = false;

uint16\_t scrollDelay; // in milliseconds

void readSerial(void)

{

 static uint8\_t putIndex = 0;

 while (Serial.available())

 {

 newMessage[putIndex] = (char)Serial.read();

 if ((newMessage[putIndex] == '\n') || (putIndex >= BUF\_SIZE-3)) // end of message character or full buffer

 {

 // put in a message separator and end the string

 newMessage[putIndex++] = ' ';

 newMessage[putIndex] = '\0';

 // restart the index for next filling spree and flag we have a message waiting

 putIndex = 0;

 newMessageAvailable = true;

 }

 else if (newMessage[putIndex] != '\r')

 // Just save the next char in next location

 putIndex++;

 }

}

void scrollDataSink(uint8\_t dev, MD\_MAX72XX::transformType\_t t, uint8\_t col)

// Callback function for data that is being scrolled off the display

{

#if PRINT\_CALLBACK

 Serial.print("\n cb ");

 Serial.print(dev);

 Serial.print(' ');

 Serial.print(t);

 Serial.print(' ');

 Serial.println(col);

#endif

}

uint8\_t scrollDataSource(uint8\_t dev, MD\_MAX72XX::transformType\_t t)

// Callback function for data that is required for scrolling into the display

{

 static char \*p = curMessage;

 static uint8\_t state = 0;

 static uint8\_t curLen, showLen;

 static uint8\_t cBuf[8];

 uint8\_t colData;

 // finite state machine to control what we do on the callback

 switch(state)

 {

 case 0: // Load the next character from the font table

 showLen = mx.getChar(\*p++, sizeof(cBuf)/sizeof(cBuf[0]), cBuf);

 curLen = 0;

 state++;

 // if we reached end of message, reset the message pointer

 if (\*p == '\0')

 {

 p = curMessage; // reset the pointer to start of message

 if (newMessageAvailable) // there is a new message waiting

 {

 strcpy(curMessage, newMessage); // copy it in

 newMessageAvailable = false;

 }

 }

 // !! deliberately fall through to next state to start displaying

 case 1: // display the next part of the character

 colData = cBuf[curLen++];

 if (curLen == showLen)

 {

 showLen = CHAR\_SPACING;

 curLen = 0;

 state = 2;

 }

 break;

 case 2: // display inter-character spacing (blank column)

 colData = 0;

 curLen++;

 if (curLen == showLen)

 state = 0;

 break;

 default:

 state = 0;

 }

 return(colData);

}

 void scrollText(void)

{

 static uint32\_t prevTime = 0;

 // Is it time to scroll the text?

 if (millis()-prevTime >= scrollDelay)

 {

 mx.transform(MD\_MAX72XX::TSL); // scroll along - the callback will load all the data

 prevTime = millis(); // starting point for next time

 }

}

uint16\_t getScrollDelay(void)

{

#if USE\_POT\_CONTROL

 uint16\_t t;

 t = analogRead(SPEED\_IN);

 t = map(t, 0, 1023, 25, 250);

 return(t);

#else

 return(SCROLL\_DELAY);

#endif

}

void setup()

{

 mx.begin();

 mx.setShiftDataInCallback(scrollDataSource);

 mx.setShiftDataOutCallback(scrollDataSink);

#if USE\_POT\_CONTROL

 pinMode(SPEED\_IN, INPUT);

#else

 scrollDelay = SCROLL\_DELAY;

#endif

 strcpy(curMessage, "Hi DATTA Vic participants, Welcome to the Whittlesea Tech School. My name is Pat McMahon and I am going to share with you, Getting Started with ARDUINO Microcontrollers, the easy way. ");

 newMessage[0] = '\0';

 Serial.begin(57600);

 Serial.print("\n[MD\_MAX72XX Message Display]\nType a message for the scrolling display\nEnd message line with a newline");

}

void loop()

{

 scrollDelay = getScrollDelay();

 readSerial();

 scrollText();

}