// Firmware for Tachistoschcope FW\_Tscope\_v1.4

// See text at the end

const byte numChars = 32;

char receivedChars[numChars]; // an array to store the received data

char messageFromPC[32] = { 0 };

char Inputstring[32] = { 0 };

int integerFromPC0 = 0;

int integerFromPC1 = 0;

int integerFromPC2 = 0;

int keuze = 0;

char recvChar;

char endMarker = '>';

boolean newData = false;

int k = 0;

int ledPin1 = 13; // the number of the LED1 pin

int ledPin2 = 12; // the number of the LED2 pin

void setup() {

Serial.begin(9600);

pinMode(ledPin1, OUTPUT); // set LED1 as output

digitalWrite(ledPin1, LOW); //turn off LED1

pinMode(ledPin2, OUTPUT); // set LED2 as output

digitalWrite(ledPin2, LOW); //turn off LED2

}

void loop() {

recvWithEndMarker();

parsNewdata();

execute\_command();

}

void recvWithEndMarker() {

static byte ndx = 0;

char endMarker = '>';

char rc;

while (Serial.available() > 0 && newData == false) {

rc = Serial.read();

if (rc != endMarker) {

receivedChars[ndx] = rc;

ndx++;

if (ndx >= numChars) {

ndx = numChars - 1;

}

}

else {

receivedChars[ndx] = '\0'; // terminate the string

ndx = 0;

newData = true;

}

}

}

void parsNewdata() {

char rc;

if (newData == true) {

Serial.print("This in parsed ");

Serial.println(receivedChars);

// split the data into its parts

char \* strtokIndx; // this is used by strtok() as an index

strtokIndx = strtok(receivedChars, ","); // get the first part - the string

strcpy(messageFromPC, strtokIndx); // copy it to messageFromPC

strtokIndx = strtok(0, ","); // this continues where the previous call left off

integerFromPC0 = atoi(strtokIndx); // convert this part to an integer

strtokIndx = strtok(0, ","); // this continues where the previous call left off

integerFromPC1 = atoi(strtokIndx); // convert this part to an integer

strtokIndx = strtok(0, ","); // this continues where the previous call left off

integerFromPC2 = atoi(strtokIndx); // convert this part to an integer

for (k = 1; k < 33; k++) {

if ((messageFromPC[k]) != (' ')) Inputstring[1] = messageFromPC[k];

}

Serial.println("This after parsed ");

Serial.println(messageFromPC);

Serial.println(String(integerFromPC0));

Serial.println(String(integerFromPC1));

Serial.println(String(integerFromPC2));

Serial.println(Inputstring[1]);

}

}

void execute\_command() {

char rc;

if (newData == true) {

if (Inputstring[1] == 'T') {

Serial.println(Inputstring[1]);

Serial.println("Input: T");

newData = false;

if (integerFromPC0 == 1) {

int k = 1;

do {

rc = Serial.read();

if (rc == 'E') k = 2;

digitalWrite(ledPin1, HIGH); //turn on LED1

delay(integerFromPC1);

digitalWrite(ledPin1, LOW); //turn off LED1

delay(integerFromPC2);

} while (k == 1);

Serial.println("Testseries 1 has stopped");

}

if (integerFromPC0 == 2) {

int k = 1;

do {

rc = Serial.read();

if (rc == 'E') k = 2;

digitalWrite(ledPin2, HIGH); //turn on LED1

delay(integerFromPC1);

digitalWrite(ledPin2, LOW); //turn off LED1

delay(integerFromPC2);

} while (k == 1);

Serial.println("Testseries 2 has stopped");

}

}

if (newData == true) {

if (Inputstring[1] == 'A') {

Serial.println("Input: A");

newData = false;

k = 1;

if (integerFromPC0 == 1) {

do {

rc = Serial.read();

if (rc == 'S') {

k = 2;

digitalWrite(ledPin1, HIGH); //turn on LED1

delay(integerFromPC1);

digitalWrite(ledPin1, LOW); //turn off LED1

}

} while (k == 1);

Serial.println("OutArd: " + String(integerFromPC0) + "," + String(integerFromPC1) + "," + String(integerFromPC2));

}

if (integerFromPC0 == 2) {

do {

rc = Serial.read();

if (rc == 'S') {

k = 2;

digitalWrite(ledPin2, HIGH); //turn on LED2

delay(integerFromPC2);

digitalWrite(ledPin2, LOW); //turn off LED2

}

} while (k == 1);

Serial.println("OutArd: " + String(integerFromPC0) + "," + String(integerFromPC1) + "," + String(integerFromPC2));

}

}

}

if (newData == true) {

if (Inputstring[1] == 'B') {

Serial.println("Input: B");

newData = false;

k = 1;

do {

rc = Serial.read();

if (rc == 'S') {

k = 2;

digitalWrite(ledPin1, HIGH); //turn on LED1

delay(integerFromPC1);

digitalWrite(ledPin1, LOW); //turn off LED1

if (integerFromPC0 > 0) delay(integerFromPC0); // if can be left out

digitalWrite(ledPin2, HIGH); //turn on LED2

delay(integerFromPC2);

digitalWrite(ledPin2, LOW); //turn off LED2

Inputstring[1] = ' ';

}

} while (k == 1);

Serial.println("OutArd: " + String(integerFromPC0) + "," + String(integerFromPC1) + "," + String(integerFromPC2));

}

}

}

if (newData == true) {

if (Inputstring[1] == 'C') {

Serial.println("Input: C");

newData = false;

Serial.println("Connected with Arduino");

}

}

}

// Firmware for Tachistoschcope FW\_Tscope\_v1.4

// Arduino/genuine uno is used . Ouput 13 and 12

// Command code for serial port of Arduino has the following structure:

// Capital, integerNumber1, integerNumber2, integerNumber3>

// “>” = the endmarker of a commandstring

// Capitals are:

// A for one channel approach

// B for two channel approach

// C for connect signal

// E Stop-signal for T (without >)

// S Start-signal for A and B commands (without >)

// T for Testprogram timing

// integerNumber1 (values: 0,1,2) in combination with Capital

// A with value 1 = one channel device number 1 (integerNumber2 is the on-time duration)

// A with value 2 = one channel device number 2 (integerNumber3 is the 0n-time duration)

// B with value 0 = two channel devices number 1 and 2 (integerNumber2 is the on-time

duration of device 1, integernumber3 is the on-time duration of device number 2)

// T with value 1 = timecontrol of device number 1 (integerNumber 2 is on-time duration,

integerNumber3 is the off-time duration)

// T with value 2 = timecontrol of device number 2 (integerNumber 2 is on-time duration,

integerNumber3 is the off-time duration)

// B with integerNumber 1 as value(>0) can serve as delay time between on-time duration of device number 1 and on-time duration of device number 2.

// After execution of the command string with “S”, the Arduino gives back:

// OutArd: integerNumber1, integerNumber2, integerNumber3

// this is done to give the VB program the possibility to check the executed time-values.

// For questions and improvements: website.vcr@gmail.com