Practicum

Week 3



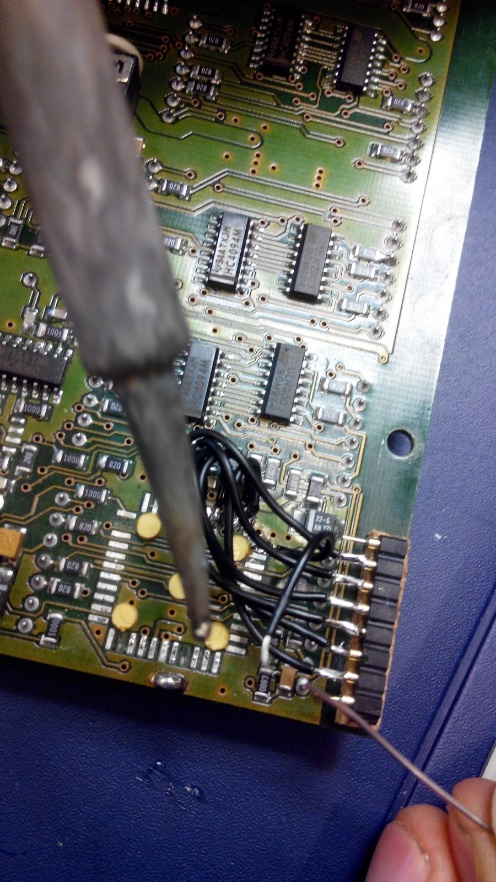
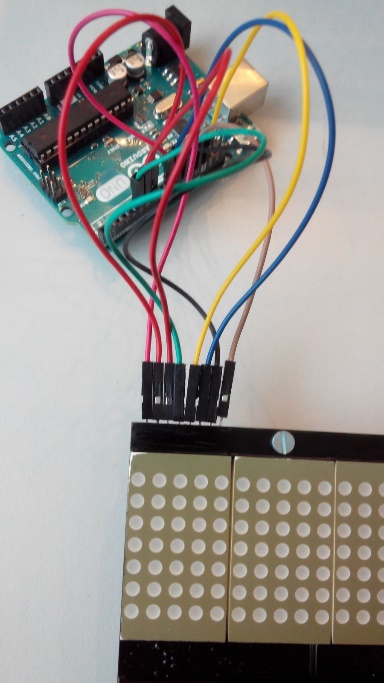
**Cas Koopmans**

**22/02/2017**

# Opdracht (zelf)

Ik heb samen met Jacob aan het grote scherm gewerkt en daar de volgende dingen uit gekregen.

* We kunnen tekst schrijven op het scherm
* We kunnen individuele pixels aansturen
* Een looplicht heen en weer sturen.

Hier onder wat foto’s en de source code in bijlages.

# Bijlage 1: looplicht

#define numOfRegisterPins 34 \* 8

int testkolom = 0;

int dataPin = 11;

int strobePin = 10;

int clockPin = 13;

const int resredPin = 9;

boolean registers[numOfRegisterPins];

const int row\_a = 5;

const int row\_b = 6;

const int row\_c = 7;

int speed=1;

void setup(){

pinMode(dataPin, OUTPUT);

pinMode(strobePin, OUTPUT);

pinMode(clockPin, OUTPUT);

pinMode (resredPin, OUTPUT);

digitalWrite (resredPin, HIGH);

pinMode (row\_c, OUTPUT);

pinMode (row\_b, OUTPUT);

pinMode (row\_a, OUTPUT);

clearRegisters();

writeRegisters();

}

void clearRegisters(){

for(int i = numOfRegisterPins - 1; i >= 0; i--){

registers[i] = LOW;

}

}

void writeRegisters(){

digitalWrite(strobePin, LOW);

for(int i = numOfRegisterPins - 1; i >= 0; i--){

digitalWrite(clockPin, LOW);

int val = registers[i];

digitalWrite(dataPin, val);

digitalWrite(clockPin, HIGH);

}

digitalWrite(strobePin, HIGH);

}

void setRegisterPin(int index, int value){

registers[index] = value;

}

void loop(){

for(int col = 0; col < 270; col++)

{

for (int row=0; row<7; row++) {

setRow(row);

setRegisterPin(col, HIGH);

writeRegisters();

delay(speed);

}

clearRegisters();

delay(speed);

}

}

void setRow (int row)

{

digitalWrite (row\_a, row & 1);

digitalWrite (row\_b, row & 2);

digitalWrite (row\_c, row & 4);

}

# Bijlage 2: tekst weergeven

#include <SPI.h> // Om snelheid te winnen maken we gebruik van SPI.

// Daarom halen we de SPI library erbij.

// Pinnen voor het aansturen van het FDS132 bord.

const int strobePin = 10; // pin voor strobe signaal, ook wel latch genaamd.

const int clockPin = 13; // pin met clock signaal.

const int dataPin = 11; // pin voor het serieel sturen van data.

const int resredPin = 9; // resred, ook wel OutputEnable of OE genoemd.

const int row\_a = 5; // ieder ledmatrix heeft 7 rijen. Hardwarematig

const int row\_b = 6; // gebruiken we een 3-naar-8 decoder

const int row\_c = 7; // type 74HC238 (U4 op het schema).

const int zero = 0; // nul, zero, nada.

int row; // int voor bijhouden welke rij we aansturen.

unsigned char character\_map[40][7] = // map met de letters die we gaan gebruiken. 34 karakters, zeven rijen elks

{

0x00,0x00,0x00,0x00,0x00,0x00,0x00,//0 spatie

0x0E,0x11,0x11,0x1F,0x11,0x11,0x11,//1 A

0x0F,0x11,0x11,0x0F,0x11,0x11,0x0F,//2 B

0x1C,0x02,0x01,0x01,0x01,0x02,0x1C,//3 C

0x07,0x09,0x11,0x11,0x11,0x11,0x0F,//4 D

0x1F,0x01,0x01,0x0F,0x01,0x01,0x1F,//5 E

0x1F,0x01,0x01,0x0F,0x01,0x01,0x01,//6 F

0x0E,0x11,0x01,0x1D,0x11,0x11,0x0E,//7 G

0x11,0x11,0x11,0x1F,0x11,0x11,0x11,//8 H

0x1F,0x04,0x04,0x04,0x04,0x04,0x1F,//9 I

0x1F,0x10,0x10,0x10,0x10,0x11,0x0E,//10 J

0x11,0x09,0x05,0x03,0x05,0x09,0x11,//11 K

0x01,0x01,0x01,0x01,0x01,0x01,0x1F,//12 L

0x11,0x1B,0x15,0x15,0x11,0x11,0x11,//13 M

0x11,0x13,0x15,0x19,0x11,0x11,0x11,//14 N

0x0E,0x11,0x11,0x11,0x11,0x11,0x0E,//15 O

0x0F,0x11,0x11,0x0F,0x01,0x01,0x01,//16 P

0x0E,0x11,0x11,0x11,0x15,0x09,0x16,//17 Q

0x0F,0x11,0x11,0x0F,0x05,0x09,0x11,//18 R

0x1E,0x01,0x01,0x0E,0x10,0x10,0x0F,//19 S

0x1F,0x04,0x04,0x04,0x04,0x04,0x04,//20 T

0x11,0x11,0x11,0x11,0x11,0x11,0x0E,//21 U

0x11,0x11,0x11,0x11,0x11,0x0A,0x04,//22 V

0x11,0x11,0x11,0x11,0x15,0x15,0x0A,//23 W

0x11,0x11,0x0A,0x04,0x0A,0x11,0x11,//24 X

0x11,0x11,0x0A,0x04,0x04,0x04,0x04,//25 y

0x1F,0x10,0x08,0x04,0x02,0x01,0x1F,//26 Z

0x00,0x00,0x00,0x00,0x00,0x00,0x00,//27 spatie

0x10,0x10,0x08,0x04,0x02,0x01,0x01,//28 /

0xAA,0x55,0xAA,0x55,0xAA,0x55,0xAA,//29 XXX

0x0E,0x11,0x19,0x15,0x13,0x11,0x0E,//30 0

0x04,0x06,0x04,0x04,0x04,0x04,0x0E,//31 1

0x0E,0x11,0x10,0x08,0x04,0x02,0x1F,//32 2

0x0E,0x11,0x10,0x0C,0x10,0x11,0x0E,//33 3

0x11,0x11,0x11,0x1F,0x10,0x10,0x10,//34 4

0x1F,0x01,0x01,0x0E,0x10,0x11,0x0E,//35 5

0x0E,0x01,0x01,0x0F,0x11,0x11,0x0E,//36 6

0x1F,0x10,0x10,0x08,0x04,0x02,0x01,//37 7

0x0E,0x11,0x11,0x0E,0x11,0x11,0x0E,//38 8

0x0E,0x11,0x11,0x1E,0x10,0x11,0x0E,//39 9

};

void setup() {

pinMode (strobePin, OUTPUT); // zet alle pinnen als output om de shift registers aan te sturen.

pinMode (clockPin, OUTPUT);

pinMode (dataPin, OUTPUT);

pinMode (row\_c, OUTPUT);

pinMode (row\_b, OUTPUT);

pinMode (row\_a, OUTPUT);

pinMode (resredPin, OUTPUT);

digitalWrite (resredPin, HIGH); // resred werkt als een schakelaar. Hoog betekent display aan, laag is display uit.

digitalWrite (strobePin, LOW); // strobe pas hoog brengen als data naar het shiftregister is gestuurd.

SPI.begin(); // initialiseer de SPI library.

SPI.setBitOrder(MSBFIRST); // Most Significant Bit eerst (LSB geeft ieder karakter in spiegelbeeld weer).

}

void loop()

{

for (row=0; row<7; row++) // rij teller.

{

digitalWrite(strobePin, LOW); // strobePin laag opdat de LEDs niet wijzigen als we de bits doorsturen.

digitalWrite (resredPin, LOW); // en we doven de display om ghosting te voorkomen.

setRow(row); // we sturen alle zeven rijen aan.

zeroDisplay(); // alle uitgangen van de shiftregisters op nul.

setDisplay(row); // haal de te tonen data op.

digitalWrite(strobePin, HIGH); // update de shiftregisters.

digitalWrite (resredPin, HIGH); // en zet display terug aan.

delayMicroseconds(1000); // pauseren want de update gaat te vlug.

}

}

void setRow (int row) // deze functie kan gebruikt worden voor het multiplexen van de zeven rijen.

{

digitalWrite (row\_a, row & 1); // zet decimale waarden 0 tot 7 om naar binaire waarden (000 - 111)

digitalWrite (row\_b, row & 2); // om de corresponderende rowpinnen die de matrixlijnen aansturen

digitalWrite (row\_c, row & 4); // te activeren.

}

void zeroDisplay() // deze functie zet alle uitgangen v/d shiftregisters op nul,

{ // waardoor alle LED's doven

int i;

for(i=0; i<34; i++)

{

SPI.transfer(0); // door 34 keer een byte met waarde nul te sturen.

}

}

void setDisplay(int row) // haal de tekstdata uit de eerder gedefinieerde map met karakters.

{

SPI.transfer(character\_map[19][row]);

SPI.transfer(character\_map[1][row]);

SPI.transfer(character\_map[3][row]);

SPI.transfer(character\_map[0][row]);

SPI.transfer(character\_map[0][row]);

SPI.transfer(character\_map[0][row]);

SPI.transfer(character\_map[0][row]);

SPI.transfer(character\_map[0][row]);

SPI.transfer(character\_map[0][row]);

SPI.transfer(character\_map[0][row]);

SPI.transfer(character\_map[0][row]);

SPI.transfer(character\_map[0][row]);

SPI.transfer(character\_map[14][row]);

SPI.transfer(character\_map[5][row]);

SPI.transfer(character\_map[0][row]);

SPI.transfer(character\_map[0][row]);

SPI.transfer(character\_map[0][row]);

SPI.transfer(character\_map[0][row]);

SPI.transfer(character\_map[0][row]);

SPI.transfer(character\_map[0][row]);

SPI.transfer(character\_map[0][row]);

SPI.transfer(character\_map[0][row]);

SPI.transfer(character\_map[2][row]);

SPI.transfer(character\_map[15][row]);

SPI.transfer(character\_map[3][row]);

SPI.transfer(character\_map[1][row]);

SPI.transfer(character\_map[10][row]);

SPI.transfer(character\_map[0][row]);

SPI.transfer(character\_map[0][row]);

SPI.transfer(character\_map[0][row]);

}