**Technisch Ontwerp**

**Vending Machine**

datum: 12-dec-2018

Versie: V0

door: Maarten Jakobs & Max van den Boom

klas: IC16AO.b

|  |  |  |
| --- | --- | --- |
| Naam: DOCENTNAAM | Naam: KLANT | Naam: AUTEUR (Student) |
| Functie: | Functie: | Functie: |
| Datum: | Datum: | Datum: |
| Goedgekeurd (handtekening) | Goedgekeurd (handtekening)  De klant kan dezelfde persoon zijn als de docent. | Goedgekeurd (handtekening) |

**Goedkeuring**

Inhoud

[Inleiding 3](#_Toc441942874)

[Eisen 3](#_Toc441942875)

[Beslissingen 3](#_Toc441942876)

[Afspraken 3](#_Toc441942877)

[Grenzen 3](#_Toc441942878)

[Plan van Aanpak 3](#_Toc441942879)

[Ontwerponderdelen GUI 4](#_Toc441942880)

[Kiezen kleur op Form 4](#_Toc441942881)

[Ontwerponderdelen datahandling 4](#_Toc441942882)

[Basisopbouw dataverzending 4](#_Toc441942883)

[Stap 1: Send ASCII vanuit C# 4](#_Toc441942884)

[Stap 2: Receive serial data vanuit USB 5](#_Toc441942885)

[Stap 3: IO pinnen Arduino 6](#_Toc441942886)

[Stap 4: Splits byte-array in RGB 7](#_Toc441942887)

[Over dit document 9](#_Toc441942888)

[Afkortingen 9](#_Toc441942889)

[Referenties 9](#_Toc441942890)

[Definities 9](#_Toc441942891)

[Gebruikte materialen 9](#_Toc441942892)

[Bijlagen 10](#_Toc441942893)

[Ascii tabel 10](#_Toc441942894)

[Arduino MEGA 2560 pinout 11](#_Toc441942895)

# Inleiding

Dit technisch ontwerp is gemaakt voor de opdracht ‘Vending machine’. Door het lezen van dit bestand moet het duidelik worden wat het product is en hoe het werk.

# Eisen

de vending machine gaat werken met een website. Je kan op de website een lijstje maken van producten die je uit de vending machine wilt halen. Deze producten kun je dan meteen afrekenen op dezelfde website. De vending machine heeft een qr code scanner waar de klant heen kan gaan wanneer hij de bestelling heeft afgerond en een speciale qr code genereerd is.   
de vending machine bestaat uit 2 lossen applicatie’s 1 voor de webkant en 1 voor de arduino kant. De programeer talen die we voor de webkant gebruiken zijn html, sql, css, javascript, php en phpmyadmin 4.8.3. de progameer taal voor de arduino kant is C. We gebruiken voor de webkant Sublime Text 3 en voor de arduino kant gebruiken we de arduino IDE.

# Beslissingen

We moesten meerdere beslissingen maken voor de werking van ons product

* Kinect of webcam. We hebben voor de kinect gekozen omdat deze een vele hogere resolutie heeft en een snellere manier van datahanding. Het voordeel ook van de kinect is dat er al heel veel programmas voor gemaakt zijn die wij dus makkelijk kunnen implemteren.
* Website voor telefoon of voor pc. We hebben de keus gemaakt om de website voor de telefoon te maken zodat mensen makkelijk de bestellingen on the go kunnen maken. En dan ook gemakkelijk de qr code gebruiken.
* Traditionele barcode of qr code. We hebben voor een qr code gekozen omdat in een qr code veel meer data verwerkt kan worden. Daarnaast ook dat een qr code genereren redelijk makkelijk is.

# Afspraken

Beschrijf hier kort:

* Aanleveren materialen of code, door wie en wanneer
* Wie wat doet, en wie iets NIET doet

We hebben het project in 2 delen verdeeld het Arduino gedeelte en de web kant.

# Grenzen

Beschrijf hier kort:

* Wat er NIET in dit ontwerp zit
* Waarom iets er NIET in zit

# Plan van Aanpak

De LED cube wordt bestuurd vanuit een C# applicatie maar de werkelijke hardware (LED’s) wordt bestuurd door Arduino’s. Omdat er meer outputs nodig zijn dat de Arduino MEAG2560 heeft is er gekozen voor meerdere Arduino’s die elk een deel van de cube (128 LED’s per Arduino) voor zijn rekening neemt. De Arduino’s communiceren met de PC via USB, en onderling via het SPI protocol. Er bestaan dus verschillende punten in de totale applicatie die als Showstopper geïndentificeerd kunnen worden. In het volgende hoofdstuk staan de stappen die ondernomen moeten worden om er zeker van te zijn dat de totale applicatie een kans van slagen heeft.

De stappen worden in het hoofdstuk Ontwerponderdelen verwerkt.

# Ontwerponderdelen GUI

## Kiezen kleur op Form

Gebruik usercontrol  
maak een array hiervan zodat het eenvoudig te adresseren is  
voorbeeld code  
voorbeeld aanpak:

* kies panel, zet kleur, ok
* kies meerdere panels in een range, segment en x, y, geef aan in panel met letter dat deze is geselecteerd

# Ontwerponderdelen datahandling

## Basisopbouw dataverzending

GUI > panel met adres(nr) >

## Stap 1: Send ASCII vanuit C#

Kleuren worden door een waarheidstabel aangegeven met de eerste 3 bits uit een byte: bit 0 is Rood, bit 1 is Groen, bit 2 is Blauw. Indien een bit 0 is, is de LED uit; indien een bit 1 is, is de LED aan. Voorbeeld: 000 0101; Rood aan, Groen uit, Blauw aan.

De colorData wordt verstuurd met ASCII:

* 0000 0000 is ASCII character 0
* ASCII value is d48.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Kleur | B7 | B6 | B5 | B4 | B3 | B2 Blue | B1 Green | B0 Red | ASCII char | ASCII value |
| Niets | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | d48 |
| rood | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | d49 |
| groen | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 2 | d50 |
| geel | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 3 | d51 |
| blauw | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 4 | d52 |
| magenta | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 5 | d53 |
| cyaan | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 6 | d54 |
| wit | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 7 | d55 |

Data in: Array  
  
opbouw: type(Master, Slave), nr(1..4), segment(1 of2), colorData (64 bytes)  
variable type in C#: char[]  
totale commandolengte: 67 bytes (char)  
  
colorData: alle bytes achter elkaar verstuurd, eerst de kolommen, dan de regels  
  
voorbeeld inhoud: Slave 1, segment 1, kolom 1 alleen rood, kolom 2 alleen groen, kolom 3 alleen blauw, dan alles  
  
voorbeeld command: S-1-1-1-2-4-7 (- toegevoegd voor de duidelijkheid, normaal niet aanwezig)  
  
voorbeeld in ASCII : 83-49-49-49-50-52-55  
voorbeeld array index : 0 1 2 3 4 5 6

## Stap 2: Receive serial data vanuit USB

Voorbeeld code?   
Dit werkt al!

## Stap 3: IO pinnen Arduino

De IO pins kunnen worden gewijzigd door gebruik te maken van digitalWrite. Het is echter zo dat bepalen welke pin aan of uit moet dit veel code vereist. Voor elke LED moet een voorwaarde (if) worden gemaakt en voor elke pin een aparte variable die de pin aanstuurt. Het is beter om een volledige PORT van de Arduino te besturen, waarbij er dan 8 bits tegelijk worden gezet. Het voordeel is dat dit veel sneller werkt en dat er eenvoudig met loops, integers en masking kan worden gewerkt. Zie ook het volgende hoofdstuk.

*Let op*  
Voor de communicatie tussen de Arduino’s wordt gebruik gemaakt van SPI. Deze bus is ‘hard-wired’ op de Arduino MEGA2560, op de SlaveSelect (SS) na. De volgende pinnen zijn in gebruik:

MISO pin 50, MOSI pin 51, SCK pin 52, SS pin 53



Hiernaast een voorbeeld voor een Arduino UNO om de PORT# te besturen. De waarden in de code worden nu gezet door een binaire waarde maar kunnen ook worden gezet door een decimale of hexadecimale waarde naar de PORT te schrijven. Met de decimale waarden is beter te werken omdat deze in een ‘loop’ kunnen worden gebruikt.

## Stap 4: Splits byte-array in RGB

*Werking*:  
Splits byteArray met RGB colorData in 3 aparte arrays (voor elke kleur 1), status van de kleur in het byte  
  
opbouw byte: xxxx x###, x heeft geen waarde of betekenis, # is de kleurstatus

Voorbeeld

* voor segment 1, led 0, kleur rood
* waarde ASCII is 49

*Declaratie*:  
in Arduino voor de kolommen, bij een Common Anode LED type wordt de kathode ‘ge-sync-ed’, ofwel de kathode wordt aan de GND gelegd.

int doSyncLedRowRed[8];  
int doSyncLedRowGreen[8];  
int doSyncLedRowBlue[8];  
int ledValueInAscii;

*Handling*:  
onderstaande zijn vast geprogrammeerde waarden, in werkelijkheid moet er een for loop worden gebruikt.

Voorbeeld voor colorData 3

ledValueInAscii = (int)m\_data[3] – 48;   
//result for colorData 0 is: 1

doSyncLedRowRed[0] = ledValueInAscii AND 0x01;  
doSyncLedRowGreen[0] = ledValueInAscii AND 0x02;  
doSyncLedRowBlue[0] = ledValueInAscii AND 0x04;  
//result for LED 0 is: red is on, green is off, blue is off

Voorbeeld voor colorData 6

ledValueInAscii = (string)m\_data[6] – 48;   
//result for colorData 6 is: 7

doSyncLedRowRed[4] = ledValueInAscii AND 0x01;  
doSyncLedRowGreen[4] = ledValueInAscii AND 0x02;  
doSyncLedRowBlue[4] = ledValueInAscii AND 0x04;  
//result for LED 0 is: red is on, green is on, blue is on

## Stap 5: patronen

Rode tekst: kernwoorden en ideeën

Multi dimensionaal array

Element 1: RGB waarden

Element 2: patroonstap

Voorbeeld: doSyncLedRowRed[regelWaardenIndex][patroonNummerIndex]

Implementatie:

* Arduino [][]
* C# [,]

Afhandeling in Arduino

* In de loop loopt een patroonTicker, bij elke keer dat de loop doorlopen wordt gaat er 1 af, tot bij de ingestelde waarde, dan weer op de default
* Met delays zou het hele systeem vertragen

Afhandeling in C#

* Het tweede gestuurde karakter is het patroon nummer

# Over dit document

## Afkortingen

|  |  |
| --- | --- |
| Afkorting | Omschrijving |
| WBS | WorkbreakDownStucture |
|  |  |

## Referenties

|  |  |
| --- | --- |
| Referentie | Omschrijving |
|  | Voorbeelden: ‘bovenliggende’ documenten (ProjectPlan, Functioneel Ontwerp, etc.) |
|  |  |

## Definities

|  |  |
| --- | --- |
| Definitie | Omschrijving |
| Showstopper | Technisch onderdeel van een ontwerp dat er voor kan zorgen dat de totale haalbaar heid van het project in gevaar komt. Het is belangrijk dat de eerst de showstoppers geïdentificeerd worden en getest worden.  Voorbeeld: als een apparaat bestuurd moet worden vanuit een PC maar de communicatie werkt niet tussen PC en apparaat, zal het dus nooit vanuit een PC bestuurd kunnen worden. Oplossing: eerst dit testen, dan pas naar de volgende stap. |
|  |  |

## Gebruikte materialen

|  |  |
| --- | --- |
| Onderdeel | Omschrijving |
|  | Voorbeelden: hardware (elektronica, arduino, PC-type), software (IDE, DLL’s) |
|  |  |

# Bijlagen

## Ascii tabel



## Arduino MEGA 2560 pinout

