# Struktureret tekst

## Opbygning af programmer

Samlet set har vi valgt, at lave fire forskellige programmer. Dette har vi valgt at gøre, for at holde det overskueligt, så vi nemt kan finde et bestemt stykke kode. Vi har lavet de fire programmer, som vil blive beskrevet senere i dette afsnit.

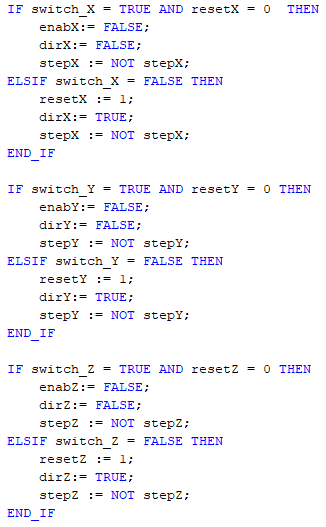
* Draw
* Reset
* EmergencyStop
* TCP

### Reset-program

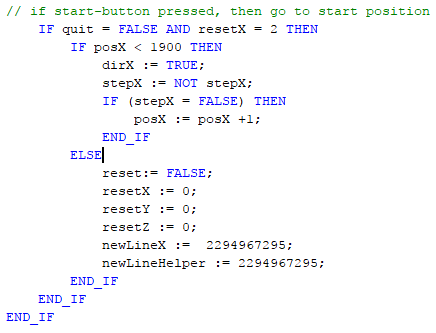
Dette program er lavet til at sætte robotten i den samme startposition, hver gang vi enten starter robotten eller af en eller anden grund ønsker at ‘genstarte’ programmet. Ud over at sætte robotten i startposition findes der også en række ‘testfunktioner’, som bruges til at køre med robotten på de forskellige akser. Reset-programmet benytter de knapper (beskrevet i elektronikdelen) der er blevet monteret på de tre akser, som vi kalder x, y og z.

I dette program benytter vi en lille række variable, som kan ses i nedenstående tabel.

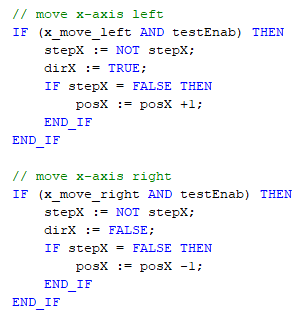
|  |  |
| --- | --- |
| **Navn** | **Type** |
| resetX | USINT |
| resetY | USINT |
| resetZ | USINT |
| switch\_X | BOOL |
| switch\_Y | BOOL |
| switch\_Z | BOOL |
| reset | BOOL |
| testEnab | BOOL |
| dirX, dirY, dirZ | BOOL |
| enabX, enabY, enabZ | BOOL |
| stepX, stepY, stepZ | BOOL |
| quit | BOOL |

Størstedelen af programmet er indkapslet i et if-statement, som spørger, om reset = true og om testEnab = false. Dette sørger for, at programmet kører, når knappen reset er trykket ind, og dermed true. Herefter køres der igennem koden fra top til bund. Hvis disse betingelser er opfyldte begynder den på reset-sekvensen, som kører alle akserne tilbage til udgangspunktet. Dette gøres igen ved et if-statement, som spørger om en knap er trykket ind eller ej. Hvis knappen ikke er trykket ind sættes retningen af motoren mod den ønskede position hvorefter den kører med motoren indtil knappen bliver trykket ind. Umiddelbart efter spørges der om knappen er trykket ind og kører herefter ud igen, indtil der ikke længere trykkes på knappen. Dette kan ses på Billede 1. Dette gentages for hver af de tre akser, så vi ender ud med et udgangspunkt for resten af programmerne, så vi altid ved, at vi starter i samme position, hver gang vi kører programmet. Herefter sættes posX, posY og posY til nul, så de kan benyttes senere i koden. Dette gøres ved endnu et if-statement, som spørger om alle akserne er blevet resat, efterfulgt af en lille pause på 100ms, hvorefter værdierne sættes.

Billede 1: Udklip af reset-program

Derudover er der et stykke kode, som kører robotten hen i den bestemte startposition, som sættes i gang, når der bliver trykket på knappen ‘Start’. Som det kan ses på Billede 2 er der opsat et if-statement, som spørger om quit = false(startknap) og om resetX = 2. Dette sørger for, at der ikke kan trykkes start, før akserne er blevet resat. Herefter køres robotten i position ved hjælp af et if-statement på x-aksen. Der køres 1900 ‘steps’ ud af x-aksen.

Billede 2: Udklip af start-knap

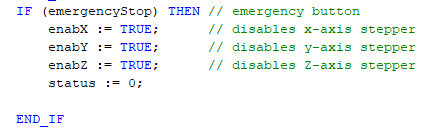
Til sidst i dette program har vi seks if-statements, som hver gør det muligt at vi kan styre de forskellige akser. Disse har vi brugt i testfasen, hvor vi evt. skulle finde ud af, hvor robotten skulle starte, hvor blyantspidseren er placeret osv. Vi benytter et boolesk udtryk

Billede 3: Testknapper til styring af robot

### Draw-program

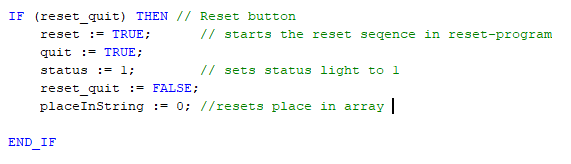
### Emergency-program

Vi har valgt at lave et dedikeret program til robottens ‘Emergency Stop’. Vi har gjort dette, da vi vil være sikker på, at motorerne stopper, når man trykker på knappen. Ved denne løsning køres programmet hvert 100ms og dermed får vi en hurtig reaktion når der trykkes på knappen. Som det kan ses på billede 4, er programmet ret simpelt. Det eneste der sker er, at et if-statement spørger efter variablen ‘emergencyStop’, og hvis denne er true, så sættes enabX, enabY og enabZ til true, hvilket stopper stepper-motorerne.



Billede : Nødstop

Derudover findes der et stykke kode der styrer knappen ‘reset’. Hvis denne bliver trykket sættes variablen ‘reset’ til true. Dette gør, at programmet ‘Reset’ starter med at køre, og sætter robotten tilbage til startpositionen. Derudover sættes en række andre variable, som er nødvendige for at kunne starte reset-programmet.



### TCP-program

### HMI - Human Machine Interface

Vi har valgt at lave et grafisk interface til at styre tegnerobotten. Vi har lavet dette fordi det giver et rigtigt godt overblik over de forskellige variable og på den måde kan vi følge med i, hvad robotten gør. Vi har valgt at lave tre forskellige sider, med hver deres funktion. På hver af disse sider findes der et ’Common layer’. Dette lægges som et lag ovenpå de enkelte sider. Her har vi placeret de vigtigste ting, såsom ’nødstop’, ’reset’ og ’start’. Derudover er der også en inputboks, hvori man indtaster den længe der stikker ud af blyantsholderen. Dette benyttes i koden i programmet ’Draw’. Til sidst findes der tre knapper, som fører hen til de sider, der findes i interfacet (Kan ses i Bilag 1).

Vi har lavet siderne ’Information’ og ’Konfiguration’ til fremtidige tilføjelser. Disse er tiltænkt, som de beskriver, forskellige informationer om position, variables værdier, evt. billedet den er ved at tegne osv. Derudover var ideen, at man kunne sætte nogle indstillinger i konfigurationen, om evt. blyanten man benytter, tiden der skal spidses osv. Disse sider er endnu blanke, men kan ses i Bilag 2 og 3.

Til sidst er der en side dedikeret til testning. Her kan der ændres på forskellige variable, som det kan ses i bilag 4. Derudover er der knapper, som hver kan kører robotten i forskellige retninger. Til sidst er der indsat variable for positionerne på de forskellige akser. Denne side har været rigtig brugbar gennem størstedelen af forløbet.

### Forbedringer

Da dette programmeringssprog er nyt for os alle har vi nogle ting, vi ikke har haft mulighed og tid til at få lavet. Disse ting er alle løsninger der hver løser mindre problemer eller gør tegningen af billedet mere effektiv.

Den ting, som vi ser som den største forbedring vi kan lave er, at få robotten til at tegne hurtigere. Som det ser ud nu, så tegner robotten en linje, hvorefter den kører tilbage til start og tegner endnu en streg. Derfor ville det være en fordel at få robotten til at tegne billedet både på vej frem og tilbage. Dette kan gøres på flere forskellige måder. Enten ved at ændre java-programmet, og måden det er sat op på. Derudover kan det også gøres i struktureret tekst, dog tænker vi, at det bliver meget sværere.

En anden væsentlig ting ved vores program er, at alle de billeder vi tegner bliver spejlvendte. Dette kan dog ændres ret nemt ved at sætte robotten til at starte i det modsatte hjørne. Vi har valgt ikke at gøre det i programmet, da vi mener det ikke betyder så meget. Dette problem kommer først rigtigt frem, hvis man tegner billeder med tekst på. Her vil teksten altså blive spejlvendt i både den ene og den anden retning.

Derudover overvejede vi metoder der kunne laves forskellige nuancer af grå, som ville gøre de billeder vi tegnede mere detaljerede. Dette kunne gøres ved forskellige grader af spidsning af blyanten. Derudover kan det også gøres ved at tegne ovenpå billedet flere gange, så man får mørkere grå de ønskede steder. Vi valgte ikke at lave dette, da det ville blive meget komplekst og desuden ikke et krav.

Vi har også tænkt en del på, hvordan vi kunne udregne længden af blyanten. Problemet opstår efter der er blevet tegnet et stykke tid. Det er tid til at spidse blyanten, hvorefter blyanten bliver kortere. Dette vil påvirke længden blyanten skal sænkes ned mod papiret, men også længden der skal køres ned for at spidse. Vi har forsøgt at udregne forholdet mellem længden af blyanten og antallet af ’steps’ der skal køres ned til papiret.

Til sidst er der nogle småting, som kunne optimeres i programmet. Vi ville gerne have nogle flere elementer i interfacet. Her skulle man gerne kunne ændre forskellige indstiller og kunne se forskellige variables værdier. Derudover resetter vi de tre akser på samme tid. Da vi gerne vil undgå, at blyanten kolliderer med blyantspidseren eller tegne streger på ved op i startpositionen ville vi gerne starte med at køre z-aksen helt i nul-position.

* Læse flere pakker fra TCP(Gustav)

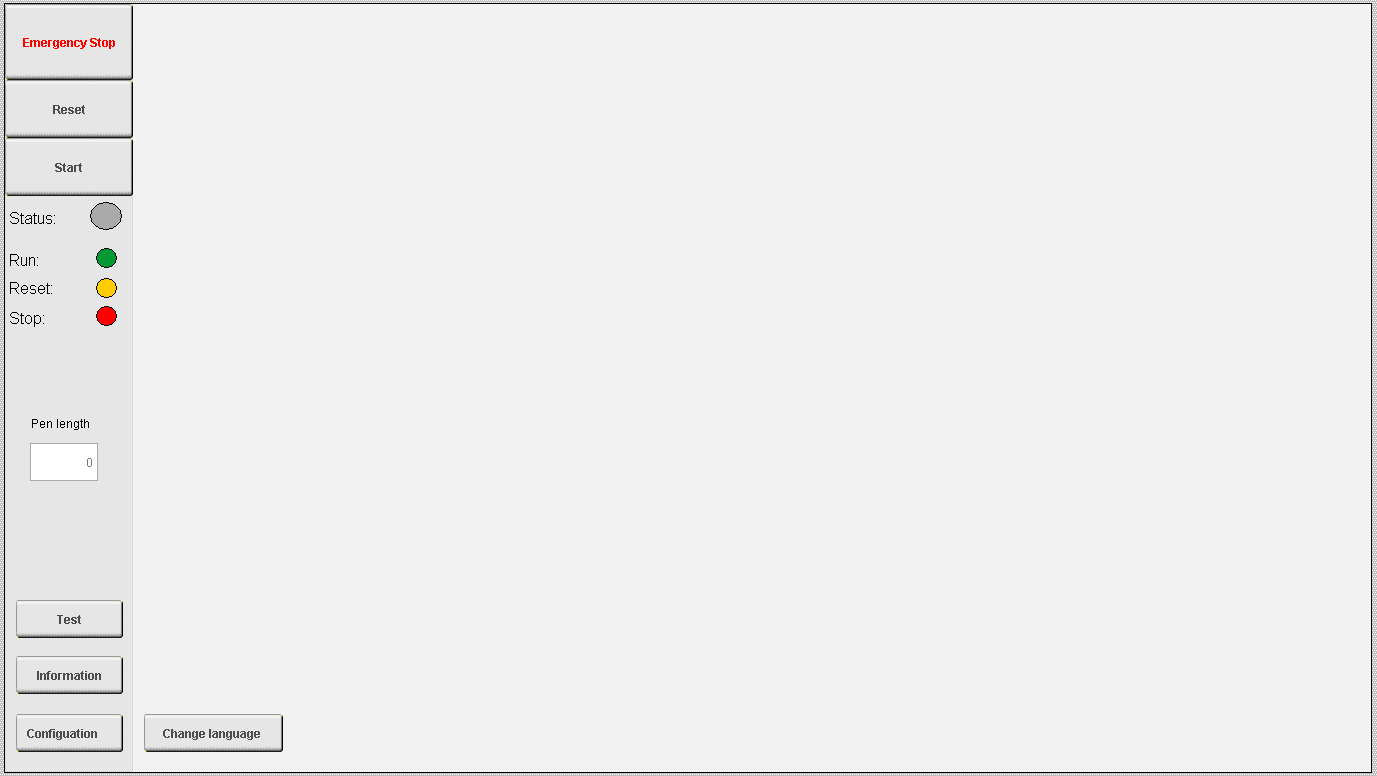
### Fejl og mangler

## Bilag

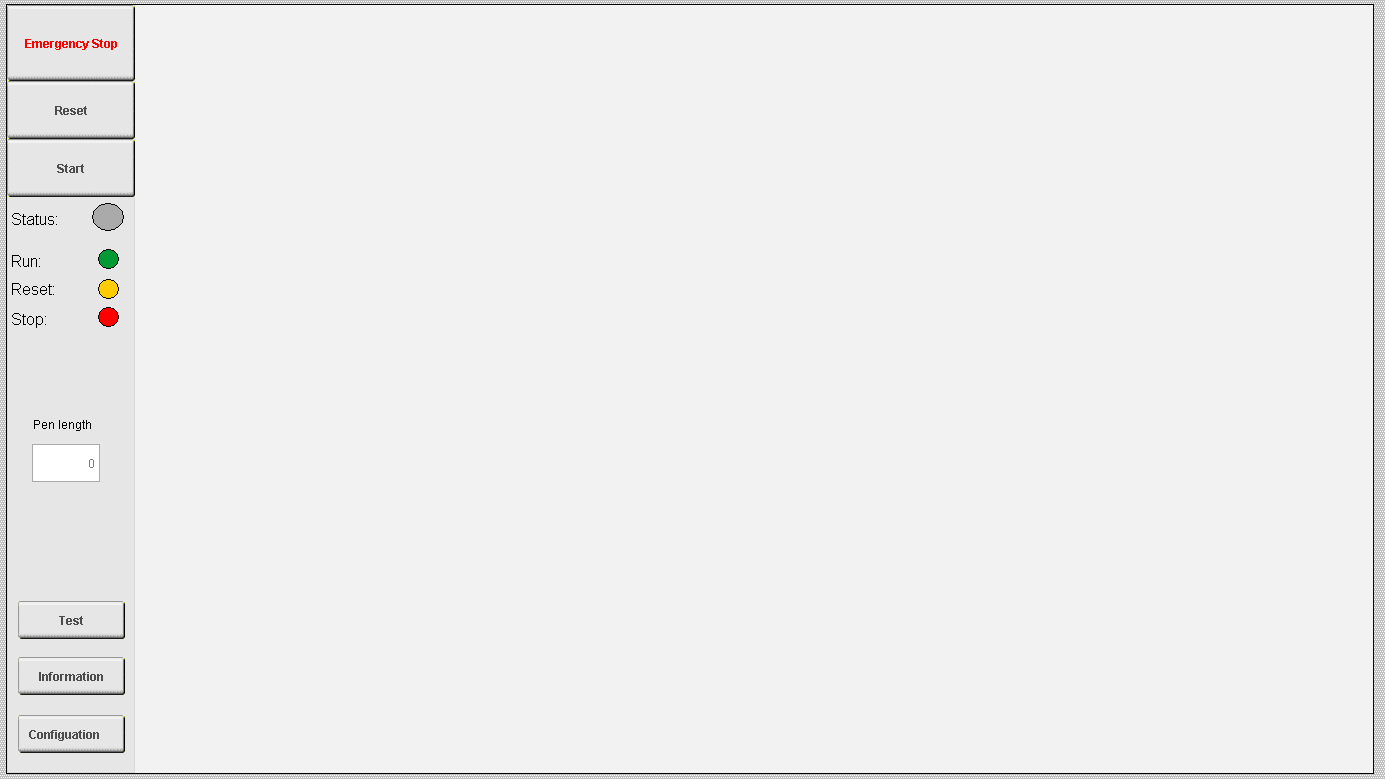
Bilag 1: HMI - Common Layer



Bilag 2: HMI - Config



Bilag 3: HMI - Information



Bilag 4: HMI - Testing

