# 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.

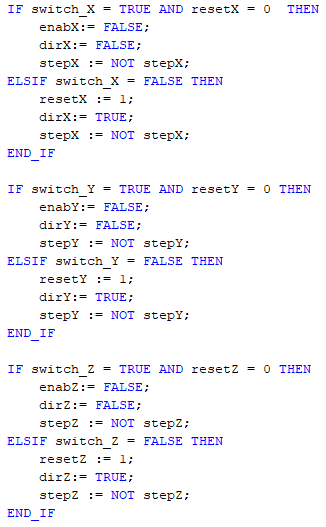
* Draw
* Reset
* EmergencyStop
* TCP

### Reset-program

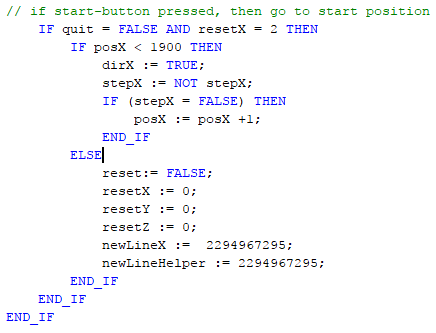
Dette program er lavet til at sætte robotten i den samme startposition, hver gang vi enten starter robotten eller af en eller anden grund ønsker at ”genstarte” programmet. Ud over at sætte robotten i startposition findes der også en række ”testfunktioner”, som bruges til at køre med robotten på de forskellige 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.

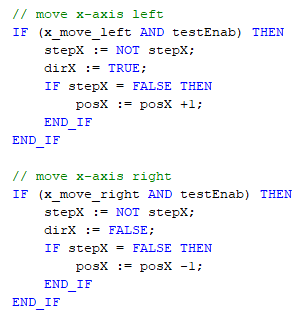
|  |  |
| --- | --- |
| **Navn** | **Type** |
| resetX | USINT |
| resetY | USINT |
| resetZ | USINT |
| switch\_X | BOOL |
| switch\_Y | BOOL |
| switch\_Z | BOOL |
| reset | BOOL |
| testEnab | BOOL |
| dirX, dirY, dirZ | BOOL |
| enabX, enabY, enabZ | BOOL |
| stepX, stepY, stepZ | BOOL |
| quit |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

### Draw-program

### Emergency-program

### TCP-program

### 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.

### Fejl og mangler