KUNGLIGA TEKNINSKA HÖGSKOLAN

LEGO Mindstorm-robot

Programmering av LEGO Mindstorm-robot i NXC

Steven Ly
2012-09-03
sly@kth.se
Introduktionskurs i Datateknik och II1310

Sammanfattning

Ingenjörer bedriver sina arbeten i projekt och en ICT-studen måste även kunna programmera, laborationen som utförst har då varit en lätt introduktion till att bedriva projekt och att programmera. Uppgiften var att göra om ett färdigt program, som en robot kommer använda sig av, så att det fungerar korrekt, roboten ska alltså kunna utföra sin uppgift med hjälp av den korrigerade koden. Vi fick fram en lösning som funkade, men det fanns många andra liknande lösningar som också hade funkat.

Sättet lösningen togs fram med var trial and error, vilket funkade bra på grund av tidigare erfarenheter av programmering. Det som mer är viktigt, är att noggrant läsa igenom Labb-PM så man inte missa enkla fel som kan förekomma.

Innehållsförteckning

1. Inledning

Rapporten kommer handla om programmering och flashning av en LEGO Mindstorm-robot, där man fått en färdig kod som ska ändras så att den funkar.

Självaste laborationen är en introduktion till programmering och arbetssättet kring ett projekt som ingenjörer oftast bedriver.
Syftet är att eleven ska introduceras till programmering och hur man arbetar kring IT, genom att uppnå målet och få roboten att göra som tänkt.

1.1 Bakgrund

Ingenjörer jobbar oftast i projekt och därför måste en student på ICT lära sig bedriva sådana, eftersom det är ICT så måste studenten även lära sig hur man bedriver IT-system i detta fall programmera.

Eftersom det är många som tidigare inte bedrivit någon form av projekt så är detta en väldig bra introduktion till detta, man får biten av at bedriva ett litet projekt och lära sig lite om programmering.

1.2 Syfte och målsättning

Syftet med uppgiften är att på ett roligt och relativt enkelt sätt introducera programmering i form av parprogrammering.

Syftet är också att introducera arbetsgången vid ingenjörsarbete samt ge träning och arbetsvana i de IT-system för utbildning som finns vid ICT-skolan.

Uppgiften syftar även till att ge träning i felsökning och testning och därmed ge en grund till effektivt programmeringsarbete i framtiden. Förhoppningen är även att studenterna ska få en förståelse för hur små enkla fel i koden kan ge oväntade resultat vid körning, och på så sätt bidra till ett noggrannhetstänk.

Målet med uppgiften är att få ett färdigt program skrivet i NXC (Not eXactly C) att fungera och därmed få en LEGO-robot att utföra sina uppgifter på önskat sätt.

För mig så var målet att lära mig mer om andra språk, i detta fall NXC, än språken som jag redan kan, även att programera en robot

2. Genomförande

Först lusläste lab-pm lite snabbt . Efter läsningen nerladdades och installerades de nödvändiga programvarorna och drivrutinerna, när drivrutinerna för LEGO-roboten väl val installerad, kopplades roboten till datorn genom usb. Den färdigakoden, som fanns på bilda, laddades ner som man senare modifierade i BricxCC.

Det andra som gjordes var att kompilera in koden till roboten för att sedan testköras, robotens beteende noterades. Koden jämfördes med beteendet och ändrades till det tänktes vara korrekt. Senare tiden bestod det praktiska av att ändra i koden och och testkörningar av roboten, try and error, detta gav en klarare bild och Lab-PM lästes senare igenom nogrannare. Även robotens konstruktion kontrollerades.

Efter ett tag byttes personen som kodade med personen som kollade på för att tillämpa parprogrammeringen och detta gjordes ett antal gånger under laborationen tills allt blev klart.

Uppgiften kunde sedan lösas genom ändringar i koden som var fel, ändringarna skrevs ner på papper i tabellform , vissa rader togs bort och roboten testkördes en sista gång. Lösningen visades sedan upp för handledaren.

3. Resultat

Radnr:	Gammalkod:	Nya koden:	Kommentarer:
2	#define SpeedSlow 80	#define SpeedSlow 35	Denna raden ändrades flera gånger innan ett lagom ”lägre” hastighet hittades
3	#define SpeedFast 100	#define SpeedFast 55	Raden ändrades tills en bra ”högre” hastighet hittades
35-36	”person1”	”ahklad”,”steven”	Bytte så att våra namn skulle visas
46	TextOut(0, (LCD_LINE2 - (8*i-16)), names[i]);	TextOut(0, (LCD_LINE2 - (8*i)), names[i]);	-16 togs bort då det gjorde så att namnen förflyttades ut “ovanför” skärmen
87	OnFwd(OUT_A, SpeedSlow);	OnFwd(OUT_A, SpeedFast);	Gasar på motor A om roboten håller på att åka ut
91	OnFwd(OUT_B, SpeedFast);	OnFwd(OUT_B, SpeedSlow);	Får motor B att köra på långsamt då den var på strecket
115	dance();	//dance();	Kommenterades bort då den störde roboten i start
116	OnFwd(OUT_AB, SpeedSlow);	//OnFwd(OUT_AB, SpeedSlow);	Onödig kod, roboten kör ändå på grund av ”followLine” funktionen

Resultatet blev att roboten följde den svarta tejpen, om den håll på att åka ut så svängde den in i igen. Men om den skulle köra in i något med trycksensorerna så stannade den och skrev ut våra namn i skärmen.

4. Analys

Det tog också en kort tid innan vi förstog att ”dance” funktionen var och spökade, den fick roboten att åka snabbt med ena hjulet och långsamt med den andra, i detta fall så att den snurra runt.

Koden som fanns på rad 87 ändrades eftersom det tidigare gjorde så att robotens hastighet i den högra motorn var lika långsam oavsett tejpen sågs av sensorn eller ej, och detta i kombination med rad 91 som gjorde så att andra motorn skulle åka snabbare oavsett ljus, fick roboten att alltid svänga åt höger. Detta korrigerades så att robotens motorer kör långsamt med båda motorerna så längde den följer längst tejpen, men gasar till med respektive motor om den skulle tappa bort tejpen.

Det som var mycket bra var att roboten åkte i långsamt längst tejpen men svängde snabbt, det gjorde så att roboten justerar sin position snabbt då den åker utan för, detta gjorde så att roboten håller sig kvar bättre på tejpen och förhindrar den från att åka för snabbt så att den tappar spåret.

5. Diskussion

Syftet med labben var bra då det är många som inte programmerat förut, eller endast för bruk som mjukvara.

Vi stötte på problemet att sensorn inte funkade som den skulle, detta på grund av att vi inte kollat igenom labb-pm ordentligt och på så sätt kontrollerat hur lego-roboten var ihop kopplat så programmet läste av fel sensor, trycksensor istället för ljussensorn, vilket inte var bra. Jag har programmerat förut fast då endast som mjukvara inte i hårdvara.

Jag har lärt mig att kolla igenom information man fått tidigare, nogrannare än vad jag brukar göra, det hade lett till att jag blivit klar med uppgiften på kortare tid. Detta är då bra att göra så att jag inte slarvar och gör fel senare i framtiden.

NXC var ett ganska lätt språk och programmet var lätt hanterligt, eftersom jag programmerat i C++ innan så kände man igen lite, med programmet man programmera i skiljde sig ganska mycket från microsoft visual studios.

Trial and Error är ett bra sätt att felsöka som kan ta lång tid eller kort tid beroende på ifall man har tur, det kan även bero på förkunskaper om ämnet.

Referenser

Labb-PM (https://bilda.kth.se/courseId/8498/content.do?id=19150198)

Bilagor

Personal note | 30 August at 17:22

Idag har jag haft laboration i introduktionkursen datateknik.

Det vi skulle göra var att ladda ner färdig kod som fanns, sen fick vi ändra koden så att när vi sen kompilera in det i en LEGO-robot så skulle den med hjälp ev en ljus sensor följa en svart tejpbit på golvet utan att åka ifrån den.

Det som var bra var att det var en väldigt rolig laboration trots att den var enkel och man fick sig en bild hur det kunde gå till i projekt. Att den var så enkel gjorde laborationen till en bra introduktion.

Det som var dåligt var att den större delen av koden var redan klart så det blev mest att ändra enstaka saker, det hade varit roligare om man fick göra en större del av koden.