

Programmering i NXC

En felsökningslabb med LEGO NXC-robotar

Arvid Viderberg

28-08-2014

arvidvi@kth.se

Introduktionskurs i datateknik I1310

Sammanfattning

Introduktionskursen I1310 är en kurs som introducerar programmering för nya studenter på ICT-skolan. Syftet är att lära ut kunskapen om vilka problem man kan stöta på som programmerare och att vara noggrann i sitt programmeringsarbete. Målet med uppgiften var att kunna förbereda drivrutiner och program innan laboration, samt att programmera en LEGO-robot att köra längs med en svart linje. Om roboten kom utanför den svarta linjen skulle den kunna hitta tillbaka och om den körde emot ett objekt skulle den spela upp en melodi och skriva ut deltagarnas namn på en inbyggd display.

Innan laborationen började hämtades nödvändiga drivrutiner och programmet Bricx Control Center laddades ner för att kunna kompilera och exekvera kod.

Vid labbtillfället hämtades en varsin LEGO robot, samt en kodmall som innehöll fel. Successivt arbetade sig gruppen igenom koden genom att noggrant granska de funktioner som kördes från main-funktionen, och hitta felen i dem. Ett par problem uppstod när två sensorer på roboten var trasiga, men efter att de bytts ut kunde arbetet fortsätta. När koden väl var korrekt testades roboten genom att köra längs med en svart linje och sedan kör in i en vägg. Roboten klarade av målet. Testet lyckades och därmed var uppgiften avklarad.

Det största problemet bestod av hårdvarufel, samt att delar av gruppen inte var bekant med programmering. Det var en personlig fördel att tidigare ha programmerat i C#/C++ eftersom det har stora likheter med NXC.

Innehållsförteckning

1. Inledning.....	4
1.1 Bakgrund	4
1.2 Syfte och målsättning	4
2. Genomförande	4
3. Resultat	5
4. Analys	6
5. Diskussion	6
Referenser	7
Bilagor	7

1. Inledning

I dagens samhälle utvecklas och skapas nya IT-produkter varje dag. Därför är det viktigt att en ingenjör bemästrar programmering för att kunna se möjligheter och problem i ett projekt som involverar programmering. Kursen är framtagen för att introducera studenterna för programmering genom att låta de, två och två, felsöka en kod som styr en LEGO-robot. Roboten ska med hjälp av en ljussensor köra längs en svart linje. Om den kör in i något ska den spela en melodi, skriva ut deltagarnas namn och sedan avsluta programmet.

1.1 Bakgrund

Som blivande KTH student är det viktigt att få bekanta sig med programmering och komma in i det problemlösningstänk som en ingenjör behöver. Produkter och IT-tjänster utvecklas hela tiden och där ska en ingenjör inom informationsteknik ligga i framkant.

1.2 Syfte och målsättning

Syftet med laborationen är att genom parprogrammering introducera studenter för programmering och vilka problem man kan stöta på som programmerare. På samma sätt introduceras studenter för arbetsgången vid ingenjörsarbete och de IT-system som används på ICT-skolan.

Uppgifter syftar också på att främja studenters noggrannhet vid programmering, genom att visa hur små fel ger oväntade resultat vid kompilering och exekvering.

Målet är att skapa en fungerande kod som kan få en LEGO NXC-robot med två motorer, två trycksensorer och en ljussensor att följa en svart linje och hitta tillbaka till linjen om den skulle hamna utanför. Den ska även känna av om den har kört emot något och i så fall spela en melodi och skriva ut deltagarnas namn på den skärm som finns monterad på roboten.



2. Genomförande

Innan laborationstillfället lästes det LabbPM som tilldelats för att introducera terminologi och programmeringsspråket NXC (NoteXactlyC), samt att förklara syftet med uppgiften. Den förklarar även hur man programmerar i Bricx Command Center, kompilerar koden, och för över den till LEGO-roboten via en USB-kabel.

Arbetet fortsatte innan laborationen med att ladda ner de drivrutiner som krävdes för att Windows skulle kunna upptäcka att roboten var ansluten till datorn, och på så sätt kunna föra över kompilerad kod till roboten. Drivrutinerna var uppladdade på KTHs utbildningswebb. På samma sida hämtades även programmet Bricx Control Center, vilket är den programmeringsmiljö som används för att skriva kod i NXC.

Vid laborationstillfället tilldelades varje grupp en robot och sedan hämtades en programmeringsfil från KTHs utbildningswebb. Koden innehöll flera medvetna fel som skulle hittas och korrigeras för att få en fungerande programkod.

Koden började bearbetas i den ordning som funktionerna kördes i main-funktionen. Genom att ändra och kompilera koden mellan varje ändringsmoment utformades en enkel problemlösningsteknik, vilket ledde till att resultatet av korrigeringarna syntes tydligt när

roboten testkördes mellan varje ändring. Halvvägs in i laborationen byttes ljussensorn och den ena trycksensorn ut, eftersom de var defekta. Problem uppstod även med att få text på skärmen att hamna på rätt ställe, men det löstes genom att noggrant gå igenom de villkor som var skapade för utskrivning av text. Roboten fick följa en svart linje, och för att bevisa att den kunde hitta tillbaka till linjen om den hamnade utanför sattes roboten en dryg decimeter ifrån linjen. Programmet exekverades på roboten och den utförde uppgiften genom att hitta linjen och sedan följa den.

3. Resultat

Eftersom alla fel blev korrigerade uppförde sig roboten på önskat vis. Den körde längs med en svart linje och när den körde in i väggen spelade den upp en melodi och skrev ut namnen på skärmen. Den kunde starta utanför linjen, hitta linjen och sedan följa den.

Följande fel korrigerades:

Radnummer	Ny kod	Kommentar
2	#define SpeedSlow 30	Sänkte hastigheten för att öka precisionen.
3	#define SpeedFast 60	Sänkte hastigheten för att öka precisionen.
33	string groupMembers[] = {	Bytte från int till string, eftersom int står för integer och därmed behandlar siffror, inte text.
35,36,37	"Akash", "Gustaf", "Arvid"	Tog bort siffrorna framför namnen, eftersom en string-variabel inte definieras med siffror.
46	TextOut(0, (LCD_LINE2 - (8*i)), groupMembers[i]);	Tog bort -16 eftersom LCD_LINE endast minskar med 8 per rad.
53	OnFwd(OUT_C, 20);	Korrigerade variablerna i funktionen för att starta motorerna, eftersom de inte stämde med den fysiska kopplingen.
62	if(Sensor(IN_1) Sensor(IN_2))	Korrigerade variablerna för trycksensorerna för att motsvara den fysiska kopplingen.
76	lightIntensity = SensorRaw(IN_4);	Korrigerade variablerna för ljussensorn för att motsvara den fysiska kopplingen.
86	Off(OUT_AC);	Korrigerade variablerna för motorerna för att motsvara den fysiska kopplingen.
90	if(lightIntensity > TopThreshold)	Bytte tecken från < till > eftersom lightIntensity ska vara större än TopThreshold
96	if(lightIntensity < BotThreshold)	if(Bytte tecken från > till < eftersom lightIntensity ska vara mindre än BotThreshold

98, 99	OnFwd(OUT_C, SpeedSlow);	Korrigerade variablerna i funktionen för att starta motorerna, eftersom de inte stämde med den fysiska kopplingen.
111	SetSensorType(IN_4, IN_TYPE_LIGHT_ACTIVE);	Korrigerade variablerna för ljussensorn för att motsvara den fysiska kopplingen.
112	SetSensorMode(IN_4, IN_MODE_RAW);	Korrigerade variablerna för ljussensorn för att motsvara den fysiska kopplingen.
113	SetSensorType(IN_2, IN_TYPE_SWITCH);	Korrigerade variablerna för trycksensorerna för att motsvara den fysiska kopplingen.
114	SetSensorMode(IN_2, IN_MODE_BOOLEAN);	Korrigerade variablerna för trycksensorerna för att motsvara den fysiska kopplingen.
116	OnFwd(OUT_AC, SpeedSlow);	Korrigerade variablerna i funktionen för att starta motorerna, eftersom de inte stämde med den fysiska kopplingen.);

4. Analys

Laborationen gick relativt bra. Det kändes bra att ha kunskap om programmering sedan tidigare, eftersom det hjälpte gruppen. Funktionen Dance var den funktion som gjorde att projektet gick sämre, men när den väl var bort-kommenterad flöt arbetet på bra. Gruppen hade ett bra samarbete och kunde effektivt resonera kring ändringarna som gjordes i koden, därav gick arbetet snabbt framåt.

5. Diskussion

Det största problemet uppstod när det gjordes korrigeringar som inte gjorde någon skillnad på roboten, vilket visade sig vara fel på hårdvaran. Funktionen "Dance" var till bekymmer i början, innan den kommenterades bort, då den kördes hela tiden, och därmed blockerade alla andra funktioner från att köras. Jag har bekantat mig med programmering innan, men har inte programmerat med program som tar in information och modifierar resultatet av körningen baserat på vad det får för input från sensorerna.

Det kommer vara till stor nytta att ha bekantat sig med NXC, eftersom vi kommer ha flera projekt med LEGO-robotar i framtiden. Det var dessutom nyttigt att få felsöka och resonera kring koden i grupp. Personligen tyckte jag att det var lätt att använda programvaran och att koda i NXC, eftersom jag tidigare programmerat i C#/C++.

Vi använde oss av ett enkelt sätt att bearbeta koden och det visade sig vara effektivt då vi efter att ha bytt ut defekt hårdvara löste uppgiften snabbt.

Referenser

LabbPM - <https://bilda.kth.se/course/11430/content.do?id=22224147> (2014-09-03)

Mjukvara info - <https://bilda.kth.se/course/11430/content.do?id=22224145> (2014-09-03)

BricxCC, drivrutiner, kodfiler - <https://bilda.kth.se/course/11430/content.do?id=22224136> (2014-09-03)

Bilagor

Inlägg Egen anteckning

Idag har vi fått laborera med LEGO NXT robotar. Vi fick i uppdrag att få en robot till att följa en svart linje, och om den kom utanför skulle den hitta tillbaka och följa linjen igen. Det var en uppgift som var utmanande och rolig. Vi hade lite problem med att vår ljussensor och den ena trycksensorn men det löste vi genom att byta robot. Arbetet fortsatte genom att börja felsöka den kod som vi fått vid början av labbtillfället. De största problemen som vi stötte på var att få en text på skärmen att hamna på rätt ställe, och att felsöka koden för att roboten skulle parera när den började hamna utanför den svarta linjen.

Jag tyckte uppgiften var relativt lätt, men jag kan tänka mig att den kan ha varit ganska svår för de som inte har programmerat överhuvudtaget innan.

Arvid Viderberg skrev inlägget | 28 augusti 10:38