TALLINNA TEHNIKAÜLIKOOL

Ahto Moorast

Teeme Ise 2011

Aruanne

Tallinn 2011

Kursusest

Olen Elektroonika ja bioonika erialal esimese kursuse tudeng ja minu jaoks oli selle kursuse valik ilmselge. Mind huvitas, kuidas ikkagi on võimalik koodiga riistvara käitumist mõjutada.

Kõige alguses võtsime läbi põhilise e. C keele. Programmeerimine on olnud üks mu huvidest ja üritasin alguses oma seniseid teadmisi lihtsalt kinnistada. Uuemad arusaamad bit-operatsioonidest sain sellelt kursuselt, kuna nendega on kokkupuuted seni olnud kasinad.

Esimene ülesanne, LEDi põlema saamine, oli alguses mulle päris pead murdev. Tuli anda väärtusi mingisugustesse registritesse ja 'sisse lülitada' õigeid bite jne. Paralleelselt käinud Arvutid I kursust sai vähemalt praktiliselt kinnistatud. Kõige keerulisem oligi aru saada just registritest ja nende kasutamise võimalustest.

Edasi sai kätt harjutatud mitmete teiste vajalike osadega nagu loendurid, PWM, andmelehelt lugemine. Viimast oli vaja lugeda üpris tihti. Registrid olid ka omavahel seotud ja osutus vajalikuks ajada sõrmega jälge, mis mida teeb. Tarvis oli ka vaadata roboti skeemi ja seda õigesti lugeda, et paika panna õiged sisendid ja väljundid, mida koodis kasutada. Vastavalt sellele sai panna tööle mootorid, lugeda valgustundlikke sensoreid jms.

Kõige huvitavamaks läks kursuse lõpupoole, kus roboti sai liikuma pandud musta joonega veetud labürindis. Selleks tuletasime PID meetodi ja implementeerisime selle ka robotisse. Edasi sai juba eelneva kogemuse ja õpitu põhjal hakata oma robotit modifitseerima, et see leiaks tee labürindi lõppu. Selleks tuli kasutada kõike kursuselt õpitut. Seda ise pusida ja näha robotit vastavalt käitumas, oli väga vinge.

Lõppkokkuvõttes andis see kursus mulle enesekindlust hakata tegelema teiste seadmetega, mida juba eelnevast suvest olen tahtnud ise ehitada. Mõistes, et see riistvara programmeerimine polegi nii keeruline ja andmelehelt lugemise kogemus ka olemas, annab see motivatsiooni teha rohkem midagi ise.

Algoritm

Kuna järgnevad kursuslased hakkavad kindlasti sarnaseid, kui mitte samu asju tegema, siis antud aruandes hoian täieliku allikkoodi lahtiseletamise ja väljatoomise kõrvale. Muidu pole lõbus ise teha. Toon välja ja seletan ainult koodijuppe, mis on antud algoritmi osade täitmise juures olulised.

Ülesanne oli Pololu 3pi robot panna liikuma mööda mustade joontega veetud labürindi. Põhilisteks tööriistadeks selle realiseerimisel said valgustundlikud sensorid roboti all ja PID(mis on koodina). Ülejäänud oli ainult tingimuste kirjeldamine koodis ja õigete käitumiste andmine eri olukordades – ristmikul, tupikus, lõpp.

Esmalt proovisimegi saada robotit järgima joont. Ristmiku esinedes panime roboti seisma. PID võimaldas robotil sõita edasi, korrigeerides enda asendit joone suhtes. Ristmiku kontroll oli määratud sensoritega. Kui kaks kõige äärmist sensorit olid piisavalt mustad, seiskasime roboti mootorid.

```
//Täielik ristmik:
if (linesensor[0]>900 && linesensor[4]>900) //Kõige äärmised
                                              näevad tumedat
{
_delay_ms(100);//Väike viivitus et sõidaks natukene edasi ja
                jääks joone keskele ilusti
drive(0,0);//Seiskame roboti
}
```

Linesensors[ ] muutujad tulenesid funktsioonist linesensor_read(), mis luges kõikide sensorite väärtused, et neid saaks sellistes if tingimustes võrrelda mingi konstandiga. See constant sõltub suuresti keskkonnast. Katsetused labürindil osutusid need väärtused üpris edukateks.

Kui olime veendunud selle töökindluses, kirjutasime olukordi juurde. Võttes arvesse erinevaid võimalikke ristmike, kus suund ainult vasakule või paremale ja otse.

```
if (linesensor[0]<400 && linesensor[4]>900) //Vasakpoolne ristmik
{
_delay_ms(100);
drive(0,0);
left_motor_reverse(60);
right_motor_forward(60);
_delay_ms(270);
}
```

Näiteks selles jupis kontrollitakse ristmikku, kus harutee läheb vasakule. Sel juhul on kõige vasakpoolsem sensor tume ja kõige parempoolsem mitte. Robot keeratakse eraldi

mootoreid kontrollivate funktsioonidega vasakule. Selleks pöörame ühte ratast vastusuunas ja teist pärisuunas. Selle lõpus saab robot jälle PID juhendusel mööda joont edasi sõita kuni uue olukorra tekkimiseni.

```
void left_motor_reverse(uint8_t speed)
{
      OCR2A = 255-speed;
      OCR2B = 255;
}

void right_motor_forward(uint8_t speed)
{
      OCR0A = 255;
      OCR0B = 255-speed;
}
```

OCRxx on AVRi enda teekides ära määratud registri nimed, kust loetakse väärtus, millal AVRi loendur katkestab voolu mootoris. Selle kaudu reguleeritakse roboti liikumiskiirust.

Üldiselt oli algoritm mõeldud järgima **Vasaku käe reeglit**. See tähendab, et robot eelistas kõigi võimalike suundade hulgast vasakule pööret. Võib ka kasutada parema suuna eelistust. Selline lähenemisviis viib alati labürindi lahendumiseni aga ei pruugi olla kõige efektiivsem.

Lisaülesandena oli tarvis muidu juurde kirjutada ka kõige lühema tee meelde jätmine, mille kood jäi ajapuuduse ja katsetusvõimaluste vähesuse tõttu implementeerimata. Üldine idee oli roboti poolt tehtud ‘käigud’ massiivi salvestada ja robotit nende järgi pöördeid sooritama panna. Kui sensoritega tuvastatakse esimene ristmik, loetakse ‘kaart_massiiv’-ist esimene element, mis ütleb, kuhu tuli sealt edasi minna ja vastavalt sellele ka robot, kas ära pöörata või hoopis edasi sõidutada.