DA215A-Datateknik MAH-Malmö

högskola

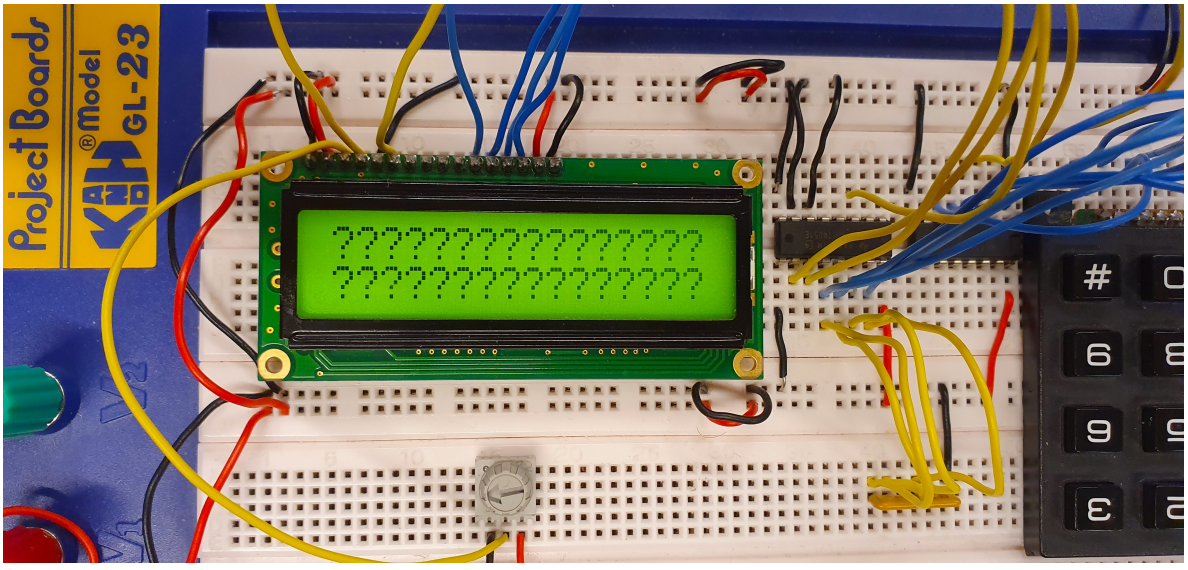
Labbrapport 4: Gissa talet

Laboranter:

Namn1: Datorid: Henrik Goth

Namn2: Datorid: Abdirahman Omar Ali

Datum då laborationen genomfördes: 3/12 - 19



Genom att skicka in labbrapporten intygar du/ni att följande regler har följts:

1. Laborationsuppgifter skall lösas självständigt av varje laborationsgrupp. Det är tillåtet att diskutera lösningar, men INTE att kopiera lösningar! Det är alltså INTE tillåtet att ge laborationsresultat eller färdiga lösningar till en annan grupp.
2. Bägge gruppmedlemmarna förväntas ta aktiv del i genomförandet av laborationen och skrivandet av rapporten. Detta inkluderar att bygga, programmera, dokumentera, testa och felsöka. Bägge gruppmedlemmarna skall kunna svara på frågor om hur laborationen genomförts och vilka resultat som erhållits.

3. Examination baseras alltid på individuella resultat

Uppgift 3.2.4

R24 används som "in-parameter". Men vilka övriga register kan användas för samma syfte? Varför kan man i detta fall inte använda något annat register istället för R24?

Svar:

R18-R27, R30 -R31 kan användas av assembly kod som anropas av C-kod. Funktionen använder bara en parameter som kommer hamna automatiskt på R2. Argument tar upp till två register där LSB hamnar i R24.

Uppgift 5.2.2 (redovisas i rapport)

*Det finns även en annan funktion som påverkar markören på displayen. Med funktionen **lcd_set_cursor_pos()** anger man var markören ska placeras. Med en kombination av bitvisa operationer skapar man instruktionen som skickas till displayen. Förklara hur de bitvisa operationerna i funktionen **lcd_set_cursor_pos()** fungerar. Ge gärna exempel på resultatet av bitoperationerna för en valfri rad och kolumn, exempelvis rad 1 (andra raden) och kolumn 2 (tredje kolumnen).*

Svar:

Med hjälp avbitvisa operationer kan man t.ex.OR tala om att flytta markoren till önskad plats. 0x80 talar om att markören ska flytttas. Raden ska shiftas 6 steg åt vänster med hjälp av left shift. Exempel på bitvisa operationer med OR:

0x80 i binärt: 0b10000000.

0b10000000 | 0b01000000 | 0b00000010 = 0b11000010

Uppgift 5.2.4 (redovisas i rapport)

*Beskriv hur funktionen **lcd_write_str()** fungerar, genom att referera till hur man använder pekaren för att stega igenom teckensträngen.*

Svar:

Funktionen får en cahr som argument. Strängar i C sparas som Char array så kommer nästa index i arrayen ligga på nästa minnesplats. Om man ökar pekaren varje gång, en char skrivs ut till pekaren får NULL. Då kommer hela strängen att skrivas ut.

Uppgift 7.2.3 (redovisas i rapport)

I början av funktionen deklareras en tecken-array (numbers), som används för att skapa en sträng av siffertecken. Funktionen tillåter endast att maximalt tre siffror kan matas in. Som ni ser så dimensioneras arrayen till att rymma fyra tecken. Vad används den sista positionen till?

Svar: Arrayen är en sträng eftersom alla tangenter är mappade till chars och extra element för Null behövs.

Uppgift 8.2.5 (redovisas i rapport)

Redogör för era erfarenheter och kunskaper från denna laboration (minst en halv A4-sida):

Vad har ni lärt er?

Om ni får välja en sak, upplevde ni något som var intressant/givande?

Fanns det något som upplevdes som svårt?

Gick allting bra eller stötte ni på problem? Om allting gick bra, vad var i så fall anledningen
detta? Om ni stötte på problem, hur löste ni i så fall dem?

Abdirahman:

Första C-laborationen har varit en intressant upplevelse då man fick äntligen lämna assembler vilket var rätt så utmannade. I laboration 4 fick vi en förståelse på hur man programmer inbyggda system i C som att skapa makrofunktioner, användning av pekare och adressreferenser till variabler. Vi fick dessutom hantera bitvisa operationer i C miljö. Vi hade inte riktig stora problem men var mest som en push till att kunna komma igång tanke på att detta var annorlunda jämfört med assembler men hade sina likheter. Sån tur med hjälp från klasskamrater och Magnus fick vi tydligare bild på hur vi skulle till väga.

Henrik:

Det var väldigt intressant att kunna skapa något som gick att använda. När allting sitter ihop så blir det lättare för mig att förstå varför vissa saker måste se ut som de gör i koden. Min största utmaning var att jag inte tyckte det är så intuitivt och hade en hel del utmaningar i att omvandla texten i uppgifterna till faktisk kod. De problem vi hade grundade sig också i detta, att förstå var vi skulle komplettera med kod, men ibland också varför. Vi fick såklart hjälp och handledning under vägens gång för att lyckas.