Laboration 4 – Tangentbord och LCD_ASM

Laboranter:

Namn1 : Alexander Johansson
Datorid : AF2015

Namn2 : Ludwig Ninn
Datorid : AF9292

Datum då laborationen genomfördes: 2016-12-08

Genom att skicka in labrapporten 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

Resultat

Uppgift 3.2.1

Följande frågor ska besvaras:

• Vad har .EQU för innebörd?

.EGU sätter ett värde till ett namn.

Example:

```
.EQU    COUNT = 0x25
LDI     R21,        COUNT       ;R21 = 0x25
```

• Vad har .CSEG för innebörd?

CSEG betyder Code Segment, och innebär att man allokerar plats i dataminnet.

• Vad händer om man skriver .ORG? Vad menas det som skrivs till höger?

.ORG betyder program origin och innebär att man allokerar plats till programmet för kod och deklarationer i programminnet. Värdet till höger är start adressen i programminnet.

• Vad sker när raden (RJMP init) exekveras?

Hoppar till initialisation som initierar vissa delar av programme men som också i sin tur kallar på andra initialisering-metoder och till slut main.

Uppgift 3.2.2

Vad har instruktionen SBI för syfte?

SBI betyder Set Bit in I/O Register och sätter en bit på en position helt enkelt.

• Vad är DDRB?

DDRB sätter en port till output eller input.

• Vad har instruktionen RET för syfte?

RET är en return till där den blev tillkallad.

Uppgift 3.2.3

Huvuddelen av programmet exekveras som en oändlig slinga (infinite loop). Det enda som utförs,förutom hoppet tillbaka till ”main”, är exekveringen av koden nedan. Vad sker när den exekveras?

Sätter biten i en position 0-7 alltså bit 6 i detta fallet till hög(1).

Uppgift 3.2.4

I den utkommenterade koden finns en liknande instruktion. Vad sker när denna exekveras?

Clearar biten i en position 0-7 alltså bit 7 i detta fallet till låg(0).

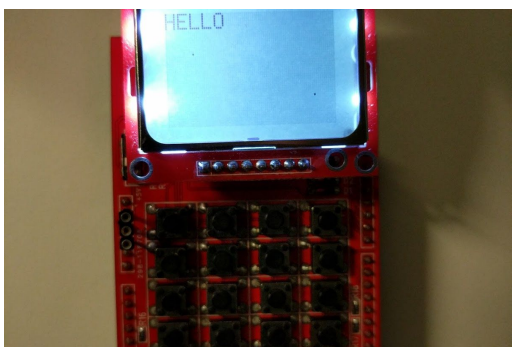
Uppgift 3.2.5 (redovisas i rapport)

I den kommenterade koden finns även ett antal förekomster av instruktionen nedan. Vad sker när denna exekveras?

NOP gör inga operationer utan bara 1 cykel (cycle).

Uppgift 5.2.3 (redovisas i rapport)

Fotografera displayen, som visar texten ”HELLO” enligt specifikationen ovan.



Uppgift 6.1.2 (redovisas i rapport)

Beskriv hur funktionen lcd_write_str fungerar, genom att referera till hur man använder pekaren för att stega igenom teckensträngen. Varför behöver Z-pekaren sparas undan innan man skriver ut varje tecken?

lcd_write_str spara temporärt det i R3:R2 för att lcd_write_string hoppar till makrot LCD_WRITE_CHR som sedan hoppar till lcd_write_char som använder sig av Z registret. För då inte förlora resterande chars sparas det temporärt.

Uppgift 7.3.5

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

• Vad har ni lärt er?

Vi har införskaffat en mer grundlig förståelse för Assembly. Lärt oss hur man initierar programkod. Skillnaden mellan programminnet och dataminnet. Skillnader mellan C och Assembly.

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

Det var intressant att programmera hur det faktiskt fungerar nära kärnan med jumps och register. Det är har varit en ögonöppnare för oss hur komplext ett program fungerar bakom kompilatorer även om det är kompilatorer på Assembly också så är det en abstraktionslager borttaget som ger en en djupare förståelse för hur saker och ting fungerar.

• Fanns det något som upplevdes som svårt?

Syntaxen var svår att lära sig och uppgifterna var som vanligt diffusa och svårtolkade.

• 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?

Labben flöt på relativt bra, fram till den sista delen. Till en början var det oklart varför vi behövde använda macros för att sätta PORTH, men vi fick det förklarat för oss. Det största problemet vi hade vara att göra samma följd som vi gjorde i C - fast i Assembly. Main-loopen skulle innehålla möjligheten att kunna hålla ned tangenter fast inte skriva ut tecken, precis som i C-labben. Problemet var egentligen inte att skriva själva koden - men att använda rätt register;

vi använde R16 för att hålla i föregående tecken, men upptäckte att det var just där felet satt. Efter vi sedan bytt till R25 fungerade vår kod.