Uppgift 3.2.1

De första raderna, i blocket som kallas för ”Start of program” i kommentarerna, ser ut så här:

```
.CSEG
.ORG RESET
RJMP init
.ORG PM_START
```

Följande frågor ska besvaras:

- ***Vad har .CSEG för innebörd?***
- ***Vad händer om man skriver .ORG? Vad menas det som skrivs till höger?***
- ***Vad sker när den tredje raden (RJMP init) exekveras?***

Svar: CSEG står för code segment vilket refererar till start av kod segmentet
.ORG sätter räknaren till det absolut högsta värdet i minnet. Reset till höger betyder att den återställs till 0 vilket tar bort allt minne. RJMP init betyder relative jump vilket kan hoppa igenom all kod beroende på hur stor koden är till då i detta fall till subrutinen init.

Uppgift 3.2.2

Nästa kodblock inleds med att konfigurera stackpekaren, men detta behandlas i en kommande laboration. Efter detta anropas en subrutin (init_pins) som hanterar konfigurering (även kallad initialisering) av in- och utgångar. Koden för subrutinen ser ut så här:

```
init_pins:
          LDI R16, 0b10000000
          OUT DDRC, R16
          RET
```

Följande frågor ska besvaras:

- **Vad har instruktionen** LDI **för syfte?**

LDI betyder load immediate vilket är att man laddar ett värde i Register 16 ex. LDI R16,0xFF

- **Vad har instruktionen** OUT **för syfte?**

OUT skriver ut värdet exempelvis till en port eller pin

- **Vad har instruktionen** RET **för syfte?**

RET fungerar som return i java vilket returnerar till där koden var sist innan den kom in till en subrutin.

- **Vad är** R16**?**

R16 är ett register i minnet vilket man kan lagra ett värde på.

- **Vad är** DDRC**?**

DDRC står för data direction c. DDR kan läsa och skriva ifrån portar.

Uppgift 3.2.3

Beskriv (steg för steg) vad som händer när koden körs:

```
LDI R16, 0b10000000

OUT DDRC, R16
```

Svar: Vi laddare register 16 (R16) med det binära talet 1000 0000, sedan så skriver vi ut R16 till
DDRC.

Uppgift 3.2.5

Huvuddelen av programmet exekveras som en oändlig slinga (infinite loop). Det enda som utförs, förutom hoppet tillbaka till ”main”, är exekveringen av:

```
SBI PORTC, 7
```

Vad sker när ovanstående exekveras?

Svar: SBI betyder (setbit) att vi gör portc på utgång 7 till set vilket betyder att den blir en 1.

Uppgift 3.2.6

I den utkommenterade koden finns en liknande instruktion:

```
CBI PORTC, 7
```

Vad sker om detta exekveras?

Svar: Som förgående svar så är detta tvärtom, här så gör vi portc på utgång 7 att den blir 0. (clear bit)

Uppgift 3.2.7

I den utkommenterade koden finns även ett antal förekomster av denna instruktion:

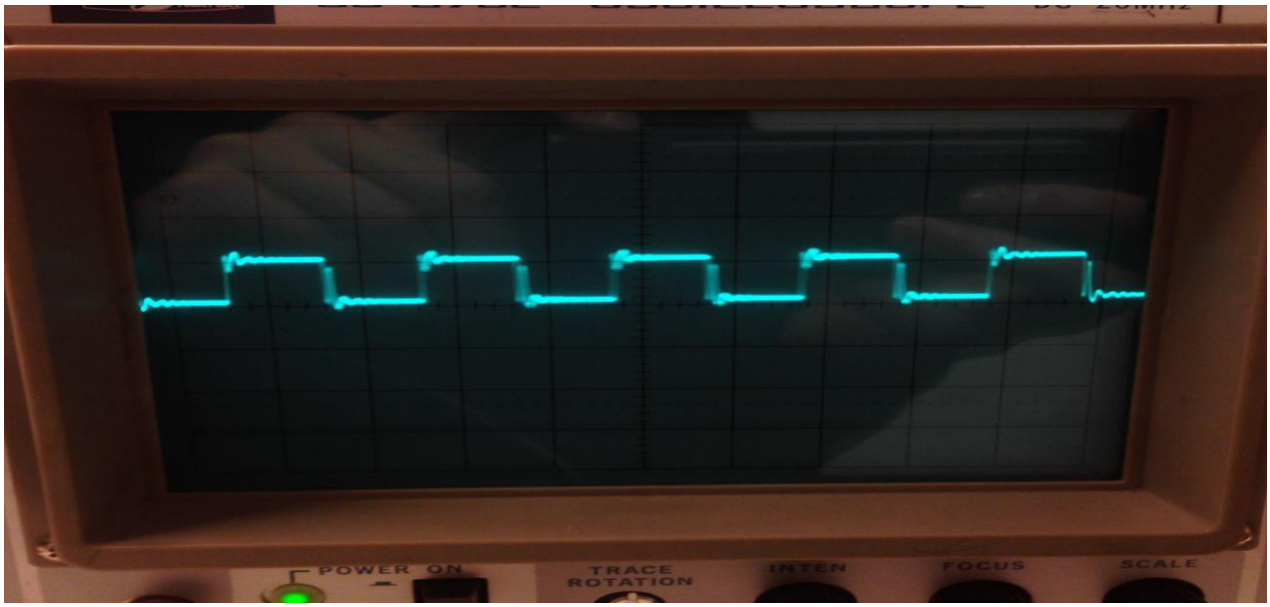
```
NOP
```

Vad sker om denna instruktion exekveras?

Svar: Nop har ingen funktion utan den tar bara tid att utföra. Den fungerar som en delay.

Uppgift 3.3.1.2

Fotografera oscilloskopsbilden och bifoga den i rapporten.



Uppgift 6.4.1

Beskriv med egna ord hur multiplexern fungerar. Redogör inte bara för hur den fungerar i denna tillämpning, utan i ett generellt sammanhang.

Svar: En multiplexer är en komponent som väljer en eller flera analoga eller digitala input signaler och skickar sedan dem valda input till en enda linje. En multiplexer har ett fåtal ingångar men flera n antal utgångar som multiplexer bestämmer beroende på ingången hur den ska skicka ut. I detta fallet så när vi trycker på telefontangentbordet så skickas en signal vilket sedan bestämmer positionen på leddisplayen.

Uppgift 6.4.2

Varför behöver man ansluta multiplexerns pinne 9 (ingång C) till jord?

Svar: Eftersom c ingången representerar 00001111 så behövs dem inte. I detta fallet använder vi dem 4 första bitarna. Därför kan vi utesluta c och sätta den till jord.

Uppgift 6.4.3

Hur lång tid tar det för en signal att färdas från ingång till utgång (Signal Input to Output)?

Ange maximalt värde! Kolla upp detta i databladet!

Svar: Det tar max 60ns att signalen ska färdas.

Uppgift 6.4.4

Hur lång tid krävs för omställning av adress till att motsvarande ingång kan avläsas på den gemensamma utgången (Address-to-Signal OUT)? Ange maximalt värde! Kolla upp detta i databladet!

Svar: Det tar 720ns för att signalen ska färdas.

Uppgift 7.2.3

Assemblera och för över programmet. Förmodligen kommer inte en enda av lysdioderna att lysa. En förklaring till problemet ges i följande illustration:

För stunden får PORTF ett värde enligt tabellen ovan. Men eftersom lysdioderna är anslutna till PF4- PF7 så krävs det att man skiftar bitarna i registret R24 (RVAL) till vänster, innan värdet överförs till PORTF. Vilken instruktion ska användas för detta?

Svar: För att siffrorna ska skifta måste vi skriva LSL RVAL vilket skiftar åt vänster ett steg, då måste vi skriva detta 4 gånger för att bli 11110000.

Uppgift 7.2.6

Som ni säkert lägger märke till så fungerar inte hanteringen av vissa tangenter. Detta beror på att koden i read keyboard inte tar hänsyn till uppgifterna som ni fick fram i uppgift 6.4.3 och 6.4.4. För att lösa detta måste ni lägga till några fler NOP-instruktioner. Hur många NOP-instruktioner behöver man inkludera totalt (inkl. befintliga)? Redovisa även beräkningar och hänvisa till uppgifterna ni fick från databladet!

Svar: För att beräkna antalet nop instruktioner så adderar vi både svaret från 6.4.3 och 6.4.4 vilket ger 60ns+720ns= 780ns

Microprocessorn oscillerar på 16MHz vilket ger tiden 1/16*10^6= 6.25e-8
Klockcyklar = (780*10^-9)/ 6.25e-8 = 12.48 cyklar

Uppgift 7.2.9

Redogör för era erfarenheter från denna laboration. Vad har ni lärt er? 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?

Jag har lärt mig med denna laboration hur man med programkod vi kan reglera och styra in-och utgångar på kortet. Jag har även lärt mig nya begrepp som kan styra koden. Allt gick bra och kopplingen gick lite slött i början då jag fick kolla upp pinmappningen men gick ganska snabbt därefter att koppla upp då jag kontrollerade flera gånger att allt var korrekt kopplat.