Institut for Matematik og Datalogi
Syddansk Universitet, Odense

22. september 2023
RF

# Øvelsestimer DM573 Uge 39/40

Husk principperne for timerne i opgaveregning i DM573: Opgaverne i gruppe I løser man i timerne med opgaveregning, sammen med de andre i sin studiegruppe. Disse opgaver skal altså *ikke* løses på forhånd, og man skal blot have læst på stoffet fra forelæsningen inden timen i opgaveregning. Opgaverne i gruppe II løse man hjemme, sammen med sin studiegruppe, inden de næste øvelsestimer i ugen efter (her uge 40).

Opgaverne denne gang handler om programmering i maskinsprog. Vi skal bruge simulatoren linket til på kursets webside, som simulerer og visualiserer en simpel CPU. CPU'ens struktur og kommandosæt er beskrevet i slides om CPUer. Vi gennemgik kort simulatoren til forelæsningen, men de vigtigste elementer gennemgås igen her:

I simulatoren vises et vindue med en CPU med registre og program counter til venstre og en række hukommelsesceller (RAM) til højre. Numre på celler og registre angives i det hexadecimale talsystem (se slides om repræsentation af tal). En hukommelsescelle og et register indeholder hver ét ord (8 bits), og dette indhold angives med to hexadecimale cifre. Hukommelsescellerne i RAM er opstillet parvis, fordi en kommando fylder to ord (16 bits). En kommando angives med fire hexadecimale cifre.

Simulatorens virkemåden er rimelig ligetil - fyld blot hukommelse med program (og data) ved direkte indtastning. Værdien af program counter kan også ændres. Tryk på Step gentagne gange (eller Play) for at udføre det indtastede program.

Man kan også skrive programmer i en tekstboks, hvilket ofte er hurtigere. Vælg Import/Export for at få en tekstboks, hvor man kan få udskrevet (export) og indlæst (import) indhold i hukommelsescellerne. Formatet er simpelt: hver linie i tekstboksen svarer til en (dobbelt) hukommelsescelle, så skriv fire hexadecimale cifre pr. linie. Tomme linier giver tomme celler. Man kan flytte programmer mellem denne tekstboks og `txt`-filer åbnet i en editor ved hjælp af copy-and-paste, og man kan dermed gemme programmer og bruge dem igen. Bemærk at websiden gemmer dit aktuelle program, så du kan lukke siden og åbne den igen uden tab.

Brug Help til f.eks. af få liste over alle kommandoer i maskinsproget. For at køre et program igen skal man bruge Reset CPU (som nulstiller registre og program counter). For at slette et helt program skal man bruge Reset RAM.

RAM cellerne indeholder både programmer og data. Læg gerne programmerne først (startende i celle 00) og data senere (gerne adskilt fra programmet af et par tomme celler).

## I: Løses i løbet af øvelsestimerne i uge 39

1. Hvad er den hexadecimale notation for kommandoerne til at gøre følgende (husk at numre på registre og RAM celler angives hexadecimalt):

   (a) Kopiere indholdet af register C til RAM celle 0A.

   (b) Lægge bitmønstret 10110011 ind i register 2.

   (c) Addere register 3 og 4, og lægge resultatet i register 5.

   (d) Lave bit-wise XOR af register B og C.

   (e) Hoppe til instruktionen i RAM celle 14 hvis indholdet i register C er større (>) end indholdet i register 0.

2. [Gentagelse fra forelæsningen] På kursets webside er der to `txt`-filer med eksempelprogrammer. Download disse, indlæs dem i simulatoren ved hjælp af copy-paste metoden ovenfor, og kør dem.

   Forstå deres virkemåde ved at læse kommentarerne til dem på slides om CPU.

   Følg også med i listen over instruktionssættet (enten i slides eller under `Help` i simulatoren).

3. Forklar hvad følgende program gør:

```
1110
1212
5112
1214
5112
3118
C000
```

4. Lav et program som læser to heltal fra RAM cellerne 10 og 12, finder deres sum, og skriver resultatet i RAM celle 14. [Hint: det er en let forandring af det første eksempelprogram.]

5. Lav et program som læser et heltal $k$ fra RAM celle 18 og som skriver summen $1 + 2 + 3 + \ldots + (k - 1)$ i RAM celle E1. [Hint: det er en let forandring af det andet eksempelprogram.] Da man med 8 bits heltal i two's complement kun kan repræsentere heltal op til 127, skal vi have $k \leq 16$ for at kunne repræsentere resultatet.

6. Lav et program som læser to heltal fra RAM celle 16 og 18, og som skriver det største af dem i celle 14.

7. Lav et program som læser et heltal $k$ fra RAM celle 20 og som skriver bitmønsteret 11111111 (hexadecimalt: FF) i RAM celle 22 hvis $k$ er forskellig fra 0, og skriver bitmønsteret 01010101 (hexadecimalt: 55) i RAM celle 22 hvis $k$ er lig 0.

8. Lav et program som læser et bitmønster fra RAM celle 10, laver de første fire bits om til 0'er, og skriver svaret i RAM celle 12. (Med "første bits" mener vi dem mest til venstre.) Hint: det kan gøres med bit-wise AND med et bestemt bitmønster (hvilket?).

9. Lav et program som læser to bitmønster $x$ og $y$ fra RAM cellerne 20 og 22, laver et nyt bitmønster, som består af de første fire bits fra $x$ efterfulgt at de sidste fire bits fra $y$, og skriver svaret i RAM celle 22. Hint: brug ideen fra sidste opgave to gange, samt bit-wise OR.

10. Lav et program som læser to bitmønster $x$ og $y$ fra RAM cellerne 20 og 22, laver et nyt bitmønster, som består af de sidste fire bits fra $y$ efterfulgt at de første fire bits fra $x$, og skriver svaret i RAM celle 22. Hint: brug ideen fra sidste opgave, samt cyklisk rotation af bits.

## II: Løses hjemme inden øvelsestimerne i uge 40

1. [Gentagelse fra forelæsningen] Løs opgaven fra sidste side i slides om CPUer og maskinkode, dvs. lav et program som tæller ned i stedet for op. Mere præcist, lav et program som efter tur skriver tallene 6, 5, 4, 3,. . . , 0 (dvs. indhold 06, 05, 04, 03,. . . , 00) i RAM celle 1C, hvis RAM celle 1A indeholder 07 til at starte med.

2. I opgave II.12 fra uge 37/38 blev beskrevet følgende alternative metode til at skifte fortegn på heltal repræsenteret i two's complement:

   > Invertér alle bits i tallet og læg derefter 1 til tallet.

   Implementer denne metode i et program.

   Mere præcist, lav et program som læser et heltal $x$ (i two's complement) fra RAM celle 20 og skriver tallet $-x$ (i two's complement) i celle 22.

   [Hint: bits i $x$ kan inverteres ved bitwise XOR af $x$ med et bestemt bitmønster (hvilket?).]

3. [Lidt svær] CPU-simulatoren har kommandoer til addition, men ikke multiplikation. Find på en metode til at lave multiplikation ud fra de eksisterende kommandoer.

   Mere præcist, lav et program som læser to heltal $x$ og $y$ (i two's complement) fra RAM celle 20 og 22, og derefter beregner $x \cdot y$ og skriver resultatet i RAM celle 24.

   Dit program behøver kun fungere for ikke-negative tal, dvs. $x, y \geq 0$. Resultatet kan også kun forventes at være korrekt når $x \cdot y \leq 127$, eftersom CPU-simulatoren bruger 8-bits heltal i two's complement.

   [Hint: Én metode kan baseres på at multiplikation pr. definition er en masse additioner (dvs. at $7 \cdot 15$ er det samme som $15 + 15 + \ldots + 15$, hvor 15 optræder i alt 7 gange). En anden metode kan baseres på opgave II.6 fra uge 38/39 (udfordrende at implementere, men giver et hurtigere program).]