## INF2270 Oblig2

Tormod Brændshøi/tormobr April 4, 2017 I denne oppgaven skulle vi lage en mini cpu. Dette løste jeg ved hjelp av en counter, ett sett med programminnet, dataminnet, et register og logikk.

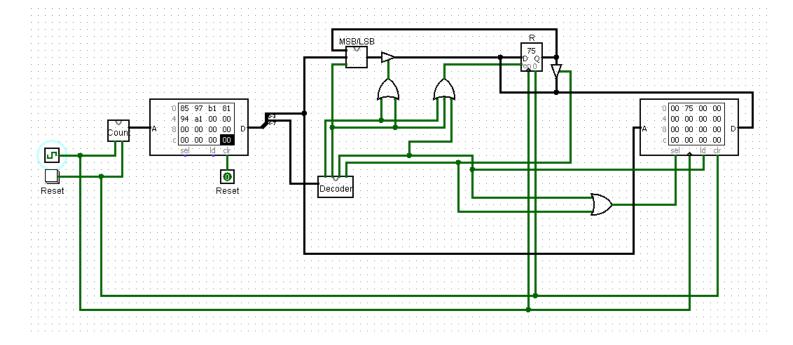
## Hovedkomponentene i løsningen min er:

- Counter(En teller som fungerer som en adressepeker til programminnet)
- Programminnet(RAM hvor instrukser lagres)
- Dekoder(Instruksene dekodes)
- LSB/MSB(Velger LSB eller MSB til å erstatte innholdet i registeret)
- Register
- Dataminnet(RAM som vi skriver til, og loader fra)

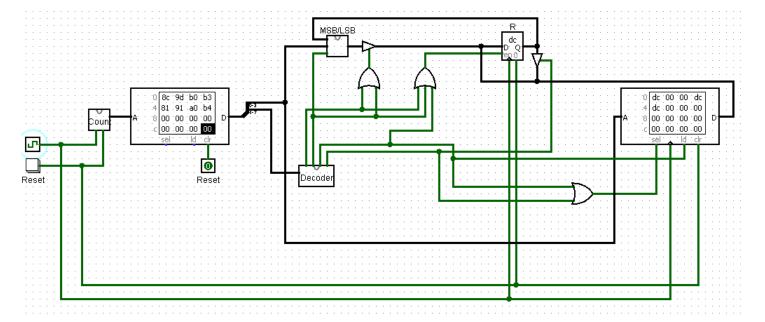
## Steg for steg hva som skjer:

- 1. En counter økes for hvert klokkesignal. Når telleren øker beveger vi oss gjennom instruksene som ligger i RAM.
- 2. Når vi når en ny instruks i RAM sendes den videre. Vi bruker så en splitter hvor det ene signalet går inn i en dekoder, og det andre går til LSB/MSB og inn som adresse-peker i dataminnet.
- 3. I dekoderen henter vi ut 4 signaler, hvor de ulike signalene er ulike instrukser, som er gitt i oppgaven. På instruks 8 og 9(1000 og 1001) skal vi legge til data i registeret, men på instruks A og B(1010 og 1011) skal vi skrive eller lese fra dataminnet. I LSB/MSB henter vi det som er i registeret, og legger til det nye signalet som LSB eller MSB avhengig av instruksene som ble hentet fra dekoderen. På instruks 8 skal vi erstatte LSB og på 9 skal vi erstatte MSB.
- 4. Avhengig av instruksene blir det nye signalet/bits sendt inn i registeret, leses fra dataminnet til registeret eller skriver fra registeret til dataminnet.
- 5. Når vi har gått gjennom alle instruksene stopper telleren, og det skjer ikke noe mer.

På bildet under kan vi se resultatet av å kjøre program 1 fra oppgaven. Vi ser at resultatet blir: 0 75.



Her ser vi resultatet av å kjøre program 2 fra oppgaven. Her bli resultatet: dc 0 0 dc dc.



Under kan vi se logging av hva som skjer når vi utfører instruksene våre. klokken går av og på, og beveger seg gjennom instruksene. Dataminnet i loggen viser xxxx xxxx med mindere instruksjonen sier at vi skal loade noe fra, eller skrive noe til RAM. Det som logges fra dataminnet er verdien på den plassen som de fire bitsene til høyre i instruksjonen angir.

Det første bildet er loggingen fra program 1, altså: v2.0 raw 85 97 B1 81 94 A1 00 00 00 00 00 00 00 00 00 00

Bilde nummer 2 er logging fra program 2, altså: v2.0 raw 8C 9D B0 B3 81 91 A0 B4 00 00 00 00 00 00 00 00

Clock(170,360)	Instruction	R	Dataminne
1	1000 0101	0000 0000	XXXX XXXX
0	1001 0111	0000 0101	XXXX XXXX
1	1001 0111	0000 0101	XXXX XXXX
0	1011 0001	0111 0101	0111 0101
1	1011 0001	0111 0101	0111 0101
0	1000 0001	0111 0101	XXXX XXXX
1	1000 0001	0111 0101	XXXX XXXX
0	1001 0100	0111 0001	XXXX XXXX
1	1001 0100	0111 0001	XXXX XXXX
0	1010 0001	0100 0001	0111 0101
1	1010 0001	0100 0001	0111 0101
0	0000 0000	0111 0101	XXXX XXXX
1	0000 0000	0111 0101	XXXX XXXX
0	0000 0000	0111 0101	XXXX XXXX
1	0000 0000	0111 0101	XXXX XXXX
0	0000 0000	0111 0101	XXXX XXXX
1	0000 0000	0111 0101	XXXX XXXX
0	0000 0000	0111 0101	XXXX XXXX
1	0000 0000	0111 0101	XXXX XXXX
0	0000 0000	0111 0101	XXXX XXXX
1	0000 0000	0111 0101	XXXX XXXX
0	0000 0000	0111 0101	XXXX XXXX
1	0000 0000	0111 0101	XXXX XXXX
0	0000 0000	0111 0101	XXXX XXXX
1	0000 0000	0111 0101	XXXX XXXX
0	0000 0000	0111 0101	XXXX XXXX
1	0000 0000	0111 0101	XXXX XXXX
0	0000 0000	0111 0101	XXXX XXXX
1	0000 0000	0111 0101	XXXX XXXX
0	0000 0000	0111 0101	XXXX XXXX
1	0000 0000	0111 0101	XXXX XXXX
0	0000 0000	0111 0101	XXXX XXXX

Clock(170,360)	Instruction	R	Dataminne
1	1000 1100	0000 0000	XXXX XXXX
0	1001 1101	0000 1100	XXXX XXXX
1	1001 1101	0000 1100	XXXX XXXX
0	1011 0000	1101 1100	1101 1100
1	1011 0000	1101 1100	1101 1100
0	1011 0011	1101 1100	1101 1100
1	1011 0011	1101 1100	1101 1100
0	1000 0001	1101 1100	XXXX XXXX
1	1000 0001	1101 1100	XXXX XXXX
0	1001 0001	1101 0001	XXXX XXXX
1	1001 0001	1101 0001	XXXX XXXX
0	1010 0000	0001 0001	1101 1100
1	1010 0000	0001 0001	1101 1100
0	1011 0100	1101 1100	1101 1100
1	1011 0100	1101 1100	1101 1100
0	0000 0000	1101 1100	XXXX XXXX
1	0000 0000	1101 1100	XXXX XXXX
0	0000 0000	1101 1100	XXXX XXXX
1	0000 0000	1101 1100	XXXX XXXX
0	0000 0000	1101 1100	XXXX XXXX
1	0000 0000	1101 1100	XXXX XXXX
0	0000 0000	1101 1100	XXXX XXXX
1	0000 0000	1101 1100	XXXX XXXX
0	0000 0000	1101 1100	XXXX XXXX
1	0000 0000	1101 1100	XXXX XXXX
0	0000 0000	1101 1100	XXXX XXXX
1	0000 0000	1101 1100	XXXX XXXX
0	0000 0000	1101 1100	XXXX XXXX
1	0000 0000	1101 1100	XXXX XXXX
0	0000 0000	1101 1100	XXXX XXXX
1	0000 0000	1101 1100	XXXX XXXX
0	0000 0000	1101 1100	XXXX XXXX