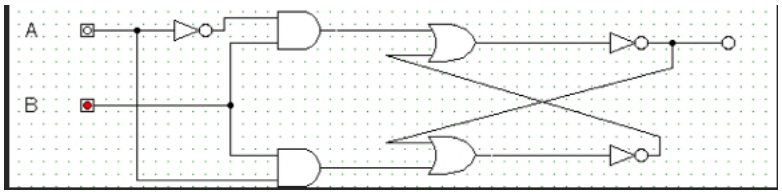
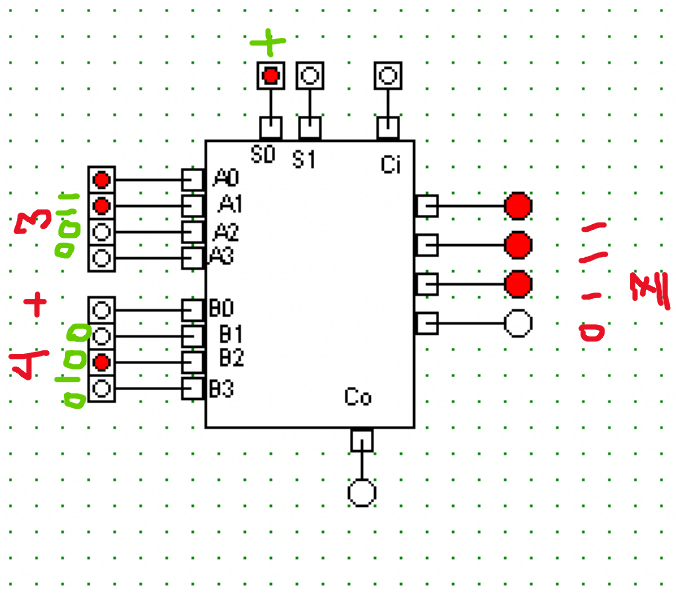
Ukesoppgaver 5 – Datamaskinsimulering, rettigheter, omdirigering

1. (Oblig) Den påviste figuren på bildet, er en D-vippe (D-latch), som vi bruker til å lagre en bit (data):

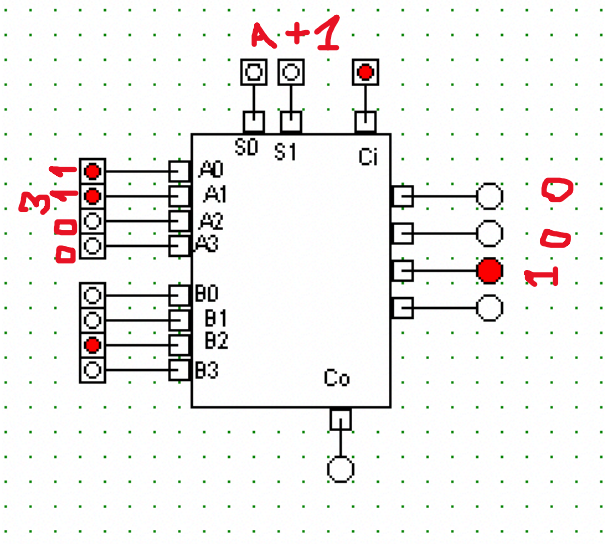


A, inputfelt, er der det sendes inn en verdi. Kontrollenheten, input B, er det som lar oss velge om vi vil lagre enheten. Dersom B er 1 vil inputen fra A lagres, men dersom B er 0 vil den lagrede verdien holdes av kretsen helt til B blir 1 igjen.

2. (Oblig)

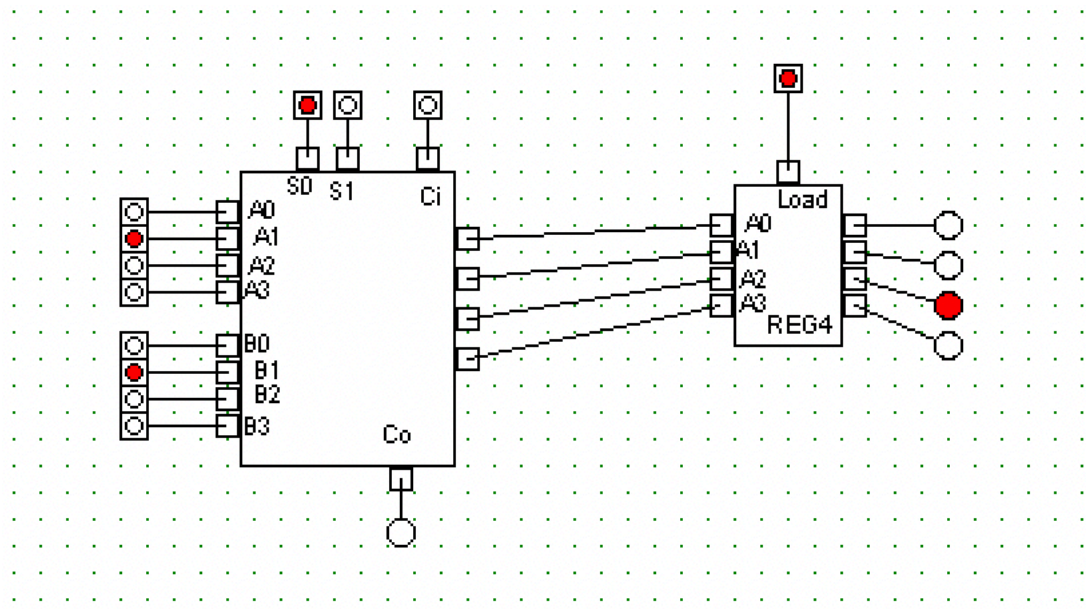


Figur 1: 3 + 4 = 7



Figur 2: Øke verdien av A med 1 → 3++ = 4

3.

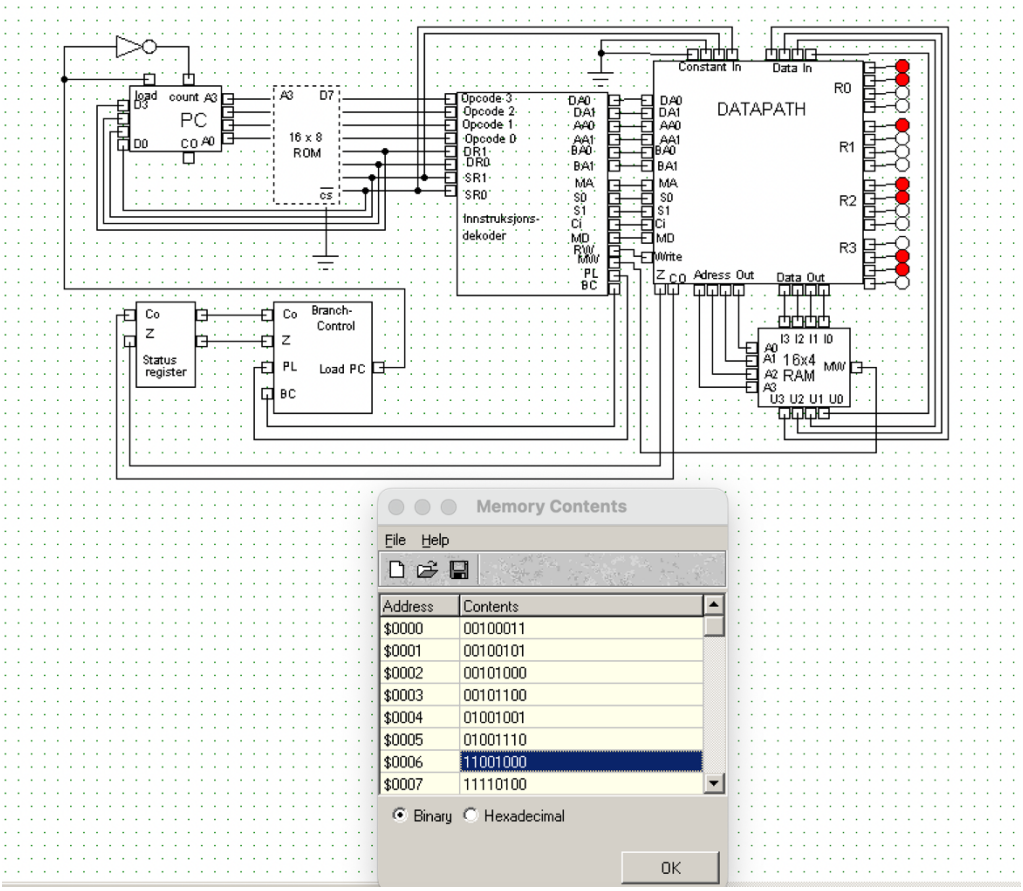


Inputbitene er vist på skjermen og det er også valgt S0 som adderer inputen. Load må ha en knapp på for at det skal lagres i registeret. Funksjonen klokka har når denne operasjonen utføres er å tikke 1 fordi da vil slavene lese verdien fra master og lagre verdien.

4. (Oblig)

ALU-en er inni DATAPATH og er hjernen til CPU-en. Prosessoren igjen er hjernen til datamaskinen så det er der alt skjer. Alt vi programmerer til syvende og sist vil også skjer her.

5. (Oblig)

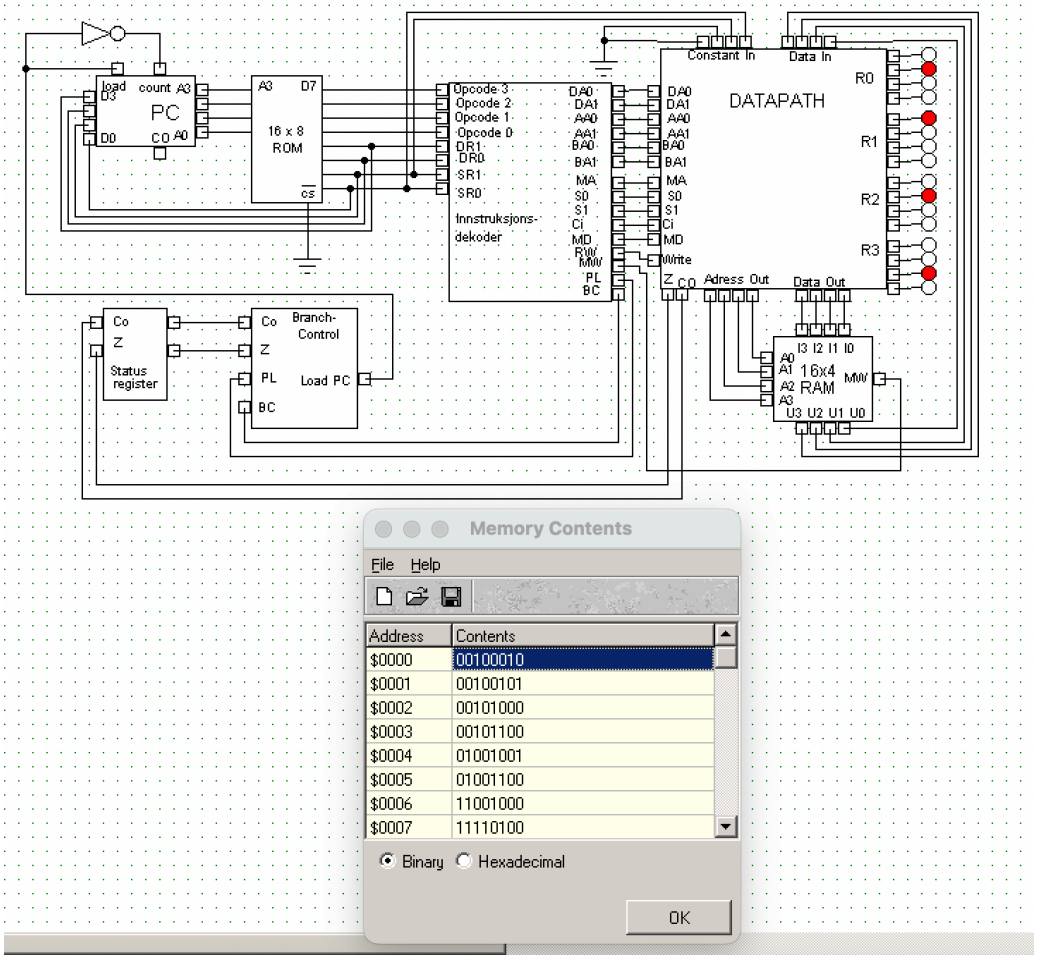


Figur 3: Resultatet registreres i R2. Resultatet blir lagt til i RAM når beregningen er ferdig, og er 6 (110)

6. (Oblig)

Vi vet at i Von Neumann arkitektur sendes data og instruksjoner gjennom samme buss. Hvis tegningen over skulle hatt en Von Neumann arkitektur hadde programmet som ligger i ROM vært i RAM. Maskinen skulle da ha lest instruksjonene, utføre de instruksjonene, og samtidig når den skulle skrive data, skulle den gå ut igjen (data output). Denne strukturen likner mer Harvard arkitektur fordi veien er annerledes her.

7. (Oblig)



Her kan man se at resultatet blir 100 binært (4) og resultatet lagres i R3. Mot slutten vil verdiene i registrene være R0 = 0010, R1 = 0001 og R2 = 0010.

8.

9.

10.

11.

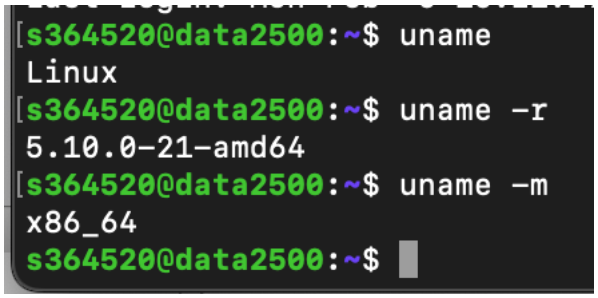
12.

8-12 må gjøres

13. (Oblig) Linux-kommando:

```
[s364520@data2500:~$ hostname
data2500
```

14. Operativsystemet datamaskinen kjører og hvilken versjon:



15. bruke grep på data2500 til å finne alle grupper brukeren «haugerud» er med i:

Søker gjennom filen «/etc/group» etter linjer som inneholder teksten «haugerud» og returnerer resultatene. Hver linje viser en gruppe, og de medlemmene som tilhører gruppen er angitt etter den første kolon. -w gir grep beskjed om å søke etter akkurat «haugerud».

16. (Oblig) Når man kopierer en fil eid av en annen bruker i linux, vil eierskapet til den nye filen bli satt til den brukeren som utfører kopieringen. For å kopiere filen "/home/haugerud/haugerud.txt" fra en annen bruker på en linux-server, kan man først logge seg inn som sin egen bruker og deretter bruke følgende kommando:

    $ cp /home/haugerud/haugerud.txt ~/

    dette vil kopiere filen til brukerens hjemmekatalog (i dette tilfelelt /home/brukernavn), og den nye filen vil bli eid av den innloggede brukeren. Husk at du kanskje ikke har tilgang til å kopiere filer eid av andre brukere, avhengig av tillatelser på filen og katalogene den er plassert i.

17. Skriverettigheter gir deg rett til å endre eller slette en fil, men de gir ikke automatisk rett til å lese filen. Du må ha separate leserrettigheter for å kunne åpne og lese en fil. For å sette leserrettigheter for en fil eller katalog, kan man bruke følgende:

    $ chmod +r filnavn.

For å sette både skriverettigheter og leserettigheter for meg selv og gruppen jeg tilhører, kan jeg bruke følgende kommando:

$chmod u+rw,g+rw filnavn

Dette gir deg og din gruppe både leserettigheter (r) og skriverettigheter (w) til filen.

18. nei, man kan ikke slette en fil man selv eier, men ikke har skriverettigheter til. Kommandoen sudo gir deg root-tilgang og da kan man få tillatelse til å endre eierskapet til filen.

19. (Oblig) ettersom globale variabler blir sendt videre til programmer vil DINVAR kunne vise innhold da jeg åpner et nytt bash.

```
[s364520@data2500:~$ ls
as       fa.txt   fil.txt        nyttScript.sh  script.sh   sumMain.c
as.s     fb.txt   hemmelig.txt   oppgave13      script.sh~  testShell
change   fil      newfile        oppgave15      sum
[s364520@data2500:~$ minvar=hei
[s364520@data2500:~$ export DINVAR=HALLO
[s364520@data2500:~$ bash
[s364520@data2500:~$ ps
    PID TTY          TIME CMD
1124078 pts/31   00:00:00 bash
1124288 pts/31   00:00:00 bash
1124290 pts/31   00:00:00 ps
[s364520@data2500:~$ echo $minvar

[s364520@data2500:~$ echo $DINVAR
HALLO
s364520@data2500:~$
```

20. (Oblig) Med script:

Ingenting kommer ut selvom DIN er en global variabel:

```
[s364520@data2500:~$ ls
as.s     fb.txt   hemmelig.txt   oppgave13  script.sh~  testShell
change   fil      newfile        oppgave15  sum         vari.sh
fa.txt   fil.txt  nyttScript.sh  script.sh  sumMain.c
[s364520@data2500:~$ chmod 700 vari.sh
[s364520@data2500:~$ ./vari.sh
[s364520@data2500:~$ echo $min

[s364520@data2500:~$ echo $DIN

s364520@data2500:~$
```

21. (Oblig)

```
-rwx------ 1 s364520 20364520    37 Feb  6 20:01 vari.sh
[s364520@data2500:~$ ps aux | jed vari.sh
[s364520@data2500:~$ ps aux | jed lagg.sh
[s364520@data2500:~$ ps aux | grep lagg.sh
s364520  1126223  0.0  0.0  24048  7632 pts/31   S+   20:12   0:00 jed lagg.sh
s364520  1126491  0.0  0.0   9232   712 pts/14   S+   20:17   0:00 grep lagg.sh
[s364520@data2500:~$ kill 1126223
[s364520@data2500:~$ ps aux | grep jed
```

22. (Oblig) Variabel med kommando inni

```
[s364520@data2500:~$ OS=$(uname)
[s364520@data2500:~$ echo $OS
Linux
s364520@data2500:~$
```

23. må prøve å gjøre denne!

24. bilder under

her lager jeg scriptet:

```
[s364520@data2500:~$ cat garbage.sh
#!/bin/bash

rm -f *.[bB][aA][kK]
rm -f *%

rm -f "#"*
```

Her lager jeg et par filer og kjører scriptet:

```
[s364520@data2500:~$ touch 1.BAK 2.bak 3.bAk 4.BaK 5% 6test
[s364520@data2500:~$ ~/garbage.sh
```

Her er resultatet etter å ha kjørt scriptet og man kan se at det fungerte:

```
[s364520@data2500:~$ ls -l
total 76
-rw-r--r-- 1 s364520 20364520     0 Feb  7 13:08 6test
-rw-r--r-- 1 s364520 20364520   741 Feb  2 17:10 as.s
-rwx------ 1 s364520 20364520    34 Feb  2 14:32 change
-rw-r--r-- 1 s364520 20364520    15 Jan 25 22:56 fa.txt
-rw-r--r-- 1 s364520 20364520    15 Jan 25 22:56 fb.txt
-rw-r--r-- 1 s364520 20364520     0 Feb  1 17:27 fil
-rwxr-xr-- 1 s364520 20364520     0 Jan 26 18:07 fil.txt
-rwx------ 1 s364520 20364520    55 Feb  7 13:03 garbage.sh
-r-------- 1 s364520 20364520     0 Jan 26 18:00 hemmelig.txt
-rw-r--r-- 1 s364520 20364520    12 Jan 25 23:49 newfile
-rwxr--r-- 1 s364520 20364520    69 Feb  1 17:14 nyttScript.sh
drwxr-xr-x 3 s364520 20364520  4096 Feb  1 17:29 oppgave13
drwxr-xr-x 2 s364520 20364520  4096 Jan 16 22:04 oppgave15
-rwxr--r-- 1 s364520 20364520    77 Feb  1 17:16 script.sh
-rwxr--r-- 1 s364520 20364520    69 Feb  1 16:49 script.sh~
-rwxr-xr-x 1 s364520 20364520 16704 Feb  2 17:11 sum
-rw-r--r-- 1 s364520 20364520   139 Feb  2 17:09 sumMain.c
drwxr-xr-x 2 s364520 20364520  4096 Feb  1 16:39 testShell
-rwx------ 1 s364520 20364520    37 Feb  6 20:01 vari.sh
s364520@data2500:~$
```

25. **Basename** er en linux kommando som tar en filbane som input og returnerer navnet på filen uten dens katalogdel.

- basename /etc/passwd → vil vise «passwd». I dette tilfellet er det en filbane til en fil med navnet passwd i katalogen etc.
- basename /etc/hosts.allows → vil vise hosts.allows
- basename /etc/hosts.allows allows → vil bare vise allows fordi basename kun returnerer den siste delen av filbanen.

Dirname er en linux-kommando som tar en filbane som input og returnerer katalogdelen til filen uten filnavnet.

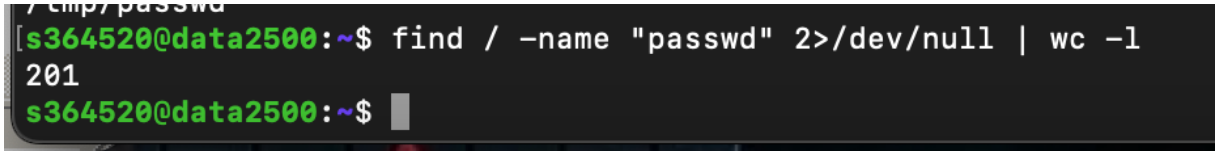
- dirname /usr/bin/find → vil vise «/usr/bin»
- dirname /usr/bin → vil bare vise usr
- dirname /hva/som/helst → vil bare vise /hva/som

26. (Oblig)

For å unngå «permission denied»-feilmeldinger mens du leter etter filer med navn «passwd», kan man skrive

Find / -name «passwd» 2>/dev/null

Som vil sende feilmeldinger til /dec/null, som er en spesiell fil i linux som ikke lagrer noe data. Dette vil resultere i at kun resultater hvor filen «passwd» blir funnet, vil li vist på skjermen. Denne har jeg grepet med wc for å se antall



27. UKENS UTFORDRING

Her er innholdet i scriptet:

Kjøring av scriptet:

For å omdirigere «Error!» til err.txt må jeg gjøre den siste kommandoen:

```
[s364520@data2500:~$ cat err.sh
#!/bin/bash
echo "Hallo"
echo "Error!">&2
[s364520@data2500:~$ ./err.sh
Hallo
Error!
[s364520@data2500:~$ ./err.sh 2>err.txt
Hallo
s364520@data2500:~$
```

28. (Oblig)

Grep bruker til å søke etter ordet «systemd» i filen /ets/systemd» og returnerer hver forekomst på en ny linje. «wc -w» teller antallet linjer og resultatet blir skrevet til filen «systemd.txt» med operatøren >.

```
[s364520@data2500:~$ grep -o 'systemd' /etc/systemd/* | wc -l > systemd.txt
grep: /etc/systemd/network: Is a directory
grep: /etc/systemd/system: Is a directory
grep: /etc/systemd/user: Is a directory
[s364520@data2500:~$ cat systemd.txt
22
s364520@data2500:~$
```

29. Forklar hva hvert ledd gjør:

```
ps aux | awk '{print $1}' | sort | uniq | wc -l
```

- Ps aux → viser liste over kjørende prosesser.
- 'awk '{print $1}'' → bruker awk programmet til å skrive ut første felt i hver linje av dataene. Print $1 er en awk-script som bestemmer hva som skal gjøres med dataene, her er det å printe innholdet i første felt i hver linje.
- Sort → sorterer linjene i en tekstfil med output fra en annen kommando alfabetisk eller numerisk.
- Uniq → er en kommando i linux som brukes til fjerne dupliserte linjer i en tekstfil eller output fra en annen kommando.
- Wc -l → brukes til å telle antall linjer i en tekstfil eller output fra en annen kommando

30. UKENS UTFORDRING 2:

Grep søker etter den angitte tekststrengen i filene som angis etter, her passwd og group

```
s364520@data2500:~$ grep -c 'haugerud' /etc/passwd /etc/group
/etc/passwd:0
/etc/group:1
s364520@data2500:~$
```

31. (Oblig)

```
s364520@data2500:~$ cat usrbin.bash
#!/bin/bash

cd /usr/bin
echo "er i $(pwd)"
s364520@data2500:~$ ./usrbin.bash
er i /usr/bin
s364520@data2500:~$
```

Etter at scriptet har kjørt er jeg i /usr/bin.

32. /proc/meminfo gir en bruksrapport om minnet på systemet.

33. UKENS UTFORDRING 3:

34. UKENS UTFORDRING 4: