Informasjon

Oppgave	Maks poeng	Oppgavetype
i		Dokument

2'er komplement

Oppgave	Maks poeng	Oppgavetype
1	2	Sammensatt

Digital representasjon

Oppgave	Maks poeng	Oppgavetype
2	1	Sammensatt
3	4	Flervalg (flere svar)
4	2	Paring

Kombinatorisk og sekvensiell logikk

Oppgave	Maks poeng	Oppgavetype
5	12	Plasser i bilde
6	2	Flervalg
7	4	Flervalg (flere svar)
8	2	Flervalg

Hardwarespråk- HDL

Oppgave	Maks poeng	Oppgavetype
9	5	Nedtrekk

Minne

Oppgave	Maks poeng	Oppgavetype
10	2	Fyll inn tall

Programmerbar logikk

Oppgave	Maks poeng	Oppgavetype
11	2	Nedtrekk

Digitale byggeblokker

Oppgave	Maks poeng	Oppgavetype
12	12	Nedtrekk

Datamaskinarkitektur

Oppgave	Maks poeng	Oppgavetype
13	6	Nedtrekk
14	4	Flervalg (flere svar)
15	4	Flervalg (flere svar)

Maskinkode

Oppgave	Maks poeng	Oppgavetype
16	4	Nedtrekk

Mikroarkitektur

Oppgave	Maks poeng	Oppgavetype
17	4	Flervalg (flere svar)
18	6	Nedtrekk
19	4	Flervalg (flere svar)

Oppbygning av cache

Oppgave	Maks poeng	Oppgavetype
20	8	Fyll inn tall

Bruk av cache

Oppgave	Maks poeng	Oppgavetype
21	10	Fyll inn tall

Skriftlig eksamen i IN2060
2020 HØST
Varighet: 3 timer
Tidspunkt: 24. november kl. 09:00 til 12:00

Det er viktig at du leser denne forsiden nøye før du starter.

Generelt:

- Husk at besvarelsen skal være anonym, du skal ikke oppgi navnet ditt.
- Alle hjelpemidler er tillatt ved skriftlig hjemmeeksamen. Det er opp til deg å innhente informasjon fra tilgjengelige kilder, vurdere kvaliteten og sette det hele sammen til en besvarelse basert på egen bearbeiding av stoffet. Besvarelsen som leveres skal reflektere eget kunnskapsnivå.

Samarbeid:

- Det er ikke tillatt å samarbeide eller kommunisere med andre under eksamen om oppgavene.
  Man kan trekkes ut til samtale for å kontrollere eierskap til sin besvarelse https://www.mn.uio.no/om/hms/koronavirus/kontrollsamtale/. Samtalen har ikke innvirkning på sensuren/karakteren, men kan lede til at instituttet oppretter fuskesak. Les mer om hva som regnes som fusk på UiOs nettsider:
  https://www.uio.no/om/regelverk/studier/studier-eksamener/fuskesaker/

Digital trøsterunde:

- Brukes kun hvis du har spørsmål til oppgavene, for alle andre spørsmål, se Kontaktinfo.
- Zoom med venterom benyttes.
- Tidspunkt er 10:00 til 11:00
- Én og én får snakke med foreleser av gangen.
- Den som ikke responderer ved sin tur blir tatt ut av køen.
- https://uio.zoom.us/j/61384641671?pwd=NGFObFVXYXhrZFhwTDBqZk9OeFdZQT09
  Meeting ID: 613 8464 1671
  Passcode: 640004

Kontaktinfo:

- Brukerstøtte eksamen
  (www.mn.uio.no/om/hms/koronavirus/brukerstotte/brukerstotte-eksamen-h20.html)

Lykke til!

1 Hva er 2'er komplementet av det binære tallet 00101101? Svaret skal skrives på binærform med 8 bit lengde.

(11010011)

Maks poeng: 2

2 Hvis du lagrer et bilde med 8-bit fargeinformasjon. Hvor mange forskjellige farger er da tilgjengelige?

(256)

Maks poeng: 1

3 Hva er korrekte utsagn om digital representasjon under

Velg de to alternativene som er korrekte

- [ ] Datamaskiner jobber mest effektivt med det hexadesimale tallsystemet.
- [x] Vi kan få overflow når vi legger sammen tall på en datamaskin.
- [ ] Det er 16 bit i en byte
- [ ] Det minst signifikante bittet bestemmer vanligvis fortegnet til tallet
- [x] Vi kan øke nøyaktigheten til et flyttall ved å øke antall bit vi representerer tallet med
- [ ] 2'er komplement er den eneste metoden vi har for å representere negative tall.

Maks poeng: 4

4 Under ser dere tall i forskjellige tallformater. De øverste tallene (i kolonne-raden) er oppgitt i desimaltall. Finn de verdiene som hører sammen.

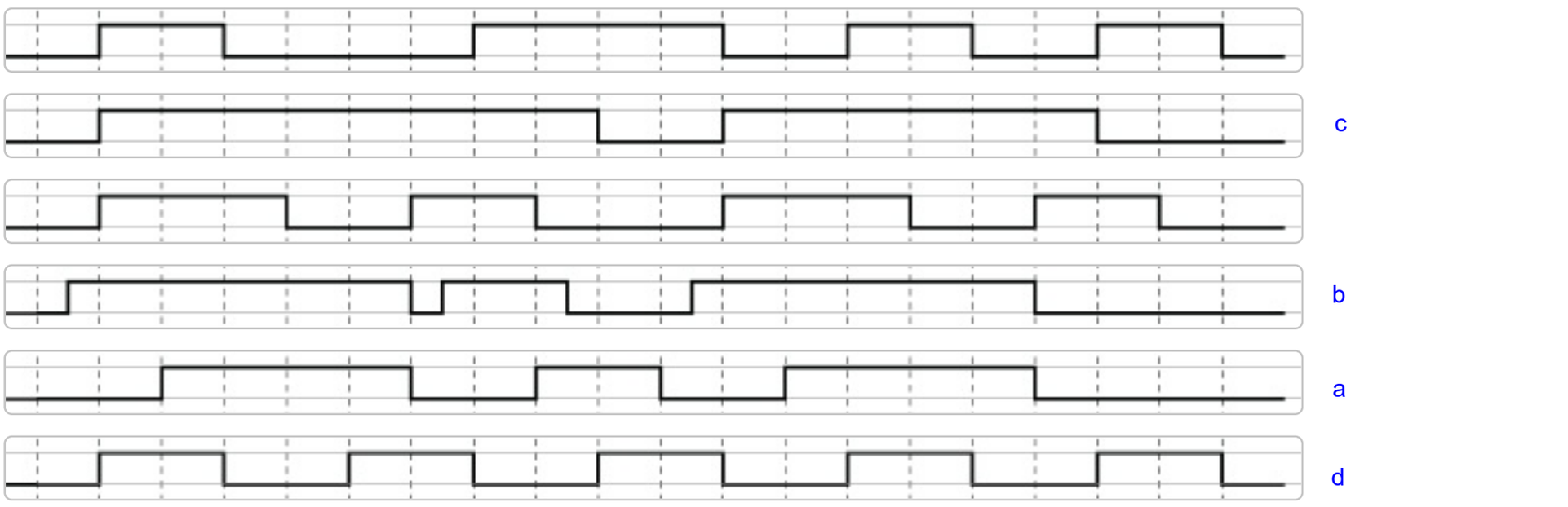
Finn de som passer sammen:

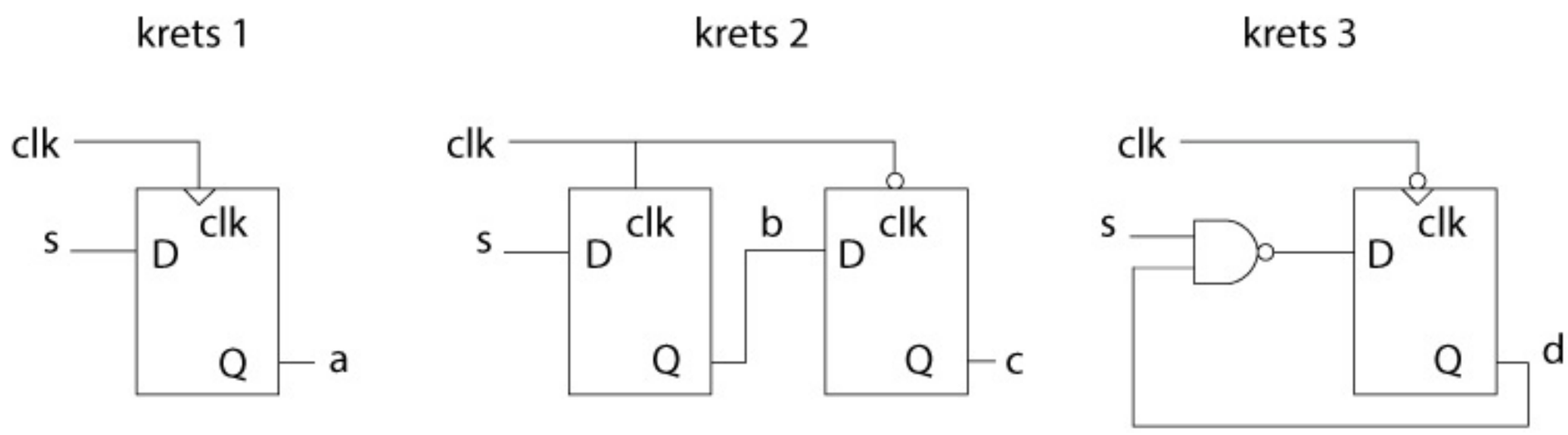
	-47	37	-15	58
00111010	○	○	○	○ ✔
0xFFD1	○ ✔	○	○	○
0x0025	○	○ ✔	○	○
11110001	○	○	○ ✔	○

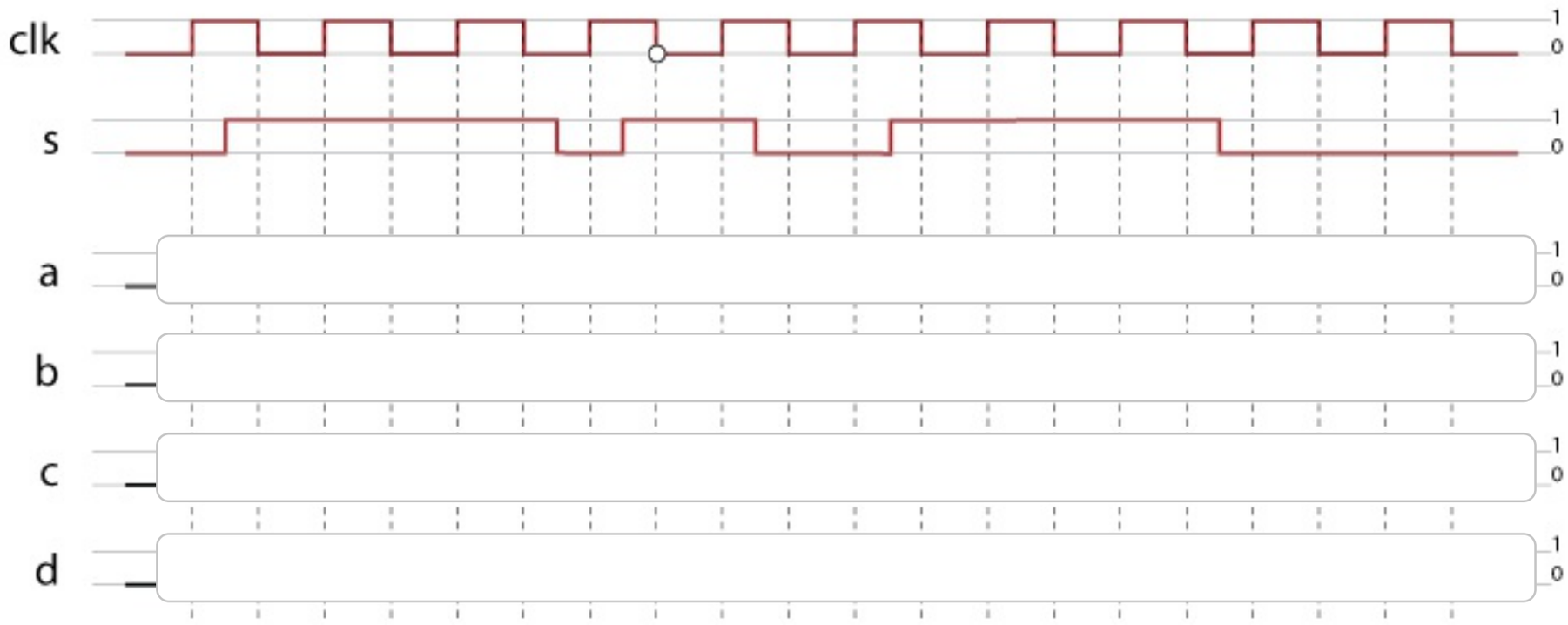
Maks poeng: 2

5 Hver av de tre kretsene under tar inn klokkesignalet **clk** og inngangsignalet **s**. Anta at utgangene **a**, **b**, **c** og **d** har startverdien **0**. Hvilke tidsforløp under hører til de forskjellige utgangene? Legg dem på riktig plass. Legg merke til at det er gitt to tidsforløp for mye. Du skal ikke ta hensyn til portforsinkelse. **Studer kretsene nøye og merk forskjellen på latcher og flippflopper i illustrasjonen.**

Hjelp

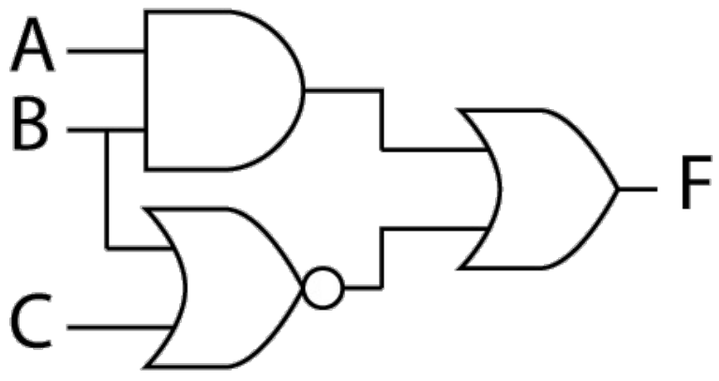






Maks poeng: 12

6 Hvilket funksjonsuttrykk gjenspeiler portimplementasjonen under?



Velg ett alternativ:

- F = AB+(B'+C')
- Ingen av alternativene er korrekte.
- F = AB+(B+C)' ✔
- F = AB(B+C)'
- F = AB(B'+C)

Maks poeng: 2

7 Hva er riktig om kombinatorisk og sekvensiell logikk

Velg de to alternativene som er korrekte

- Vi kan implementere de fleste digitale kretser med kombinatorisk logikk alene, men vi kan få raskere kretser ved å inkludere synkron sekvensiell logikk
- Vi er ikke avhengige av kombinatorisk logikk for å kunne implementere tilstandsmaskiner.
- Kombinatorisk logikk er avhengig av et klokkesignal.
- Vi kan spesifisere funksjonaliteten til en synkron sekvensiell krets med tabeller. ✔
- Kombinatorisk logikk kan huske tidligere verdier.
- En kombinatorisk krets kan spesifiseres med et uendelig antall funksjonsuttrykk. ✔

Maks poeng: 4

8 Finn det maksimalt forenklede funksjonsuttrykket til Y basert på sannhetstabellen under.

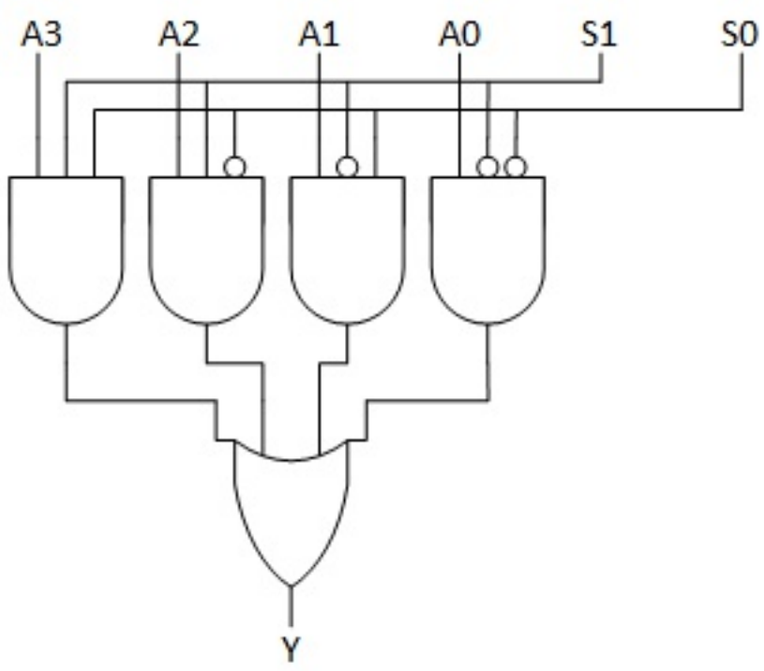
A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Velg ett alternativ

- Y = AB+AC'
- Y = A'B' + AC'
- Y = A'B'C + AB ✔
- Y = A'B+AB
- Y = ABC'+AB

Maks poeng: 2

9



Figur 1: Krets

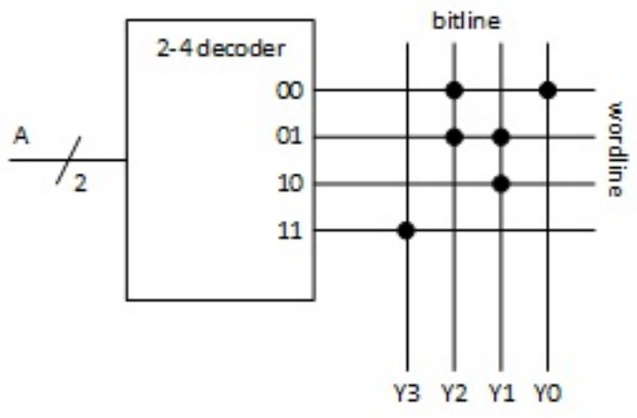
Fullfør VHDL-koden under, slik at den implementerer kretsen over.
Merk: ***Benytt valget <blank> dersom det ikke skal stå noe i feltet.***

```
library IEEE;
  use IEEE.std_logic_1164.all;

[ ](module, architecture, component, entity)min_krets is
port(
  A : [ ](out, inout, output, buffer, input, in) [ ](signal, std_logic, vector,
std_logic_vector, buffer) [ ]((4 downto 0);, (3 downto 0);, <blank>, (2 downto 0);, (1 downto 0);)
  S : [ ](inout, input, out, in, buffer, output) [ ](std_logic_vector, vector, std_logic,
signal, buffer) [ ](<blank>, (3 downto 0);, (4 downto 0);, (1 downto 0);, (2 downto 0);)
  Y : [ ](input, out, buffer, inout, output, in) [ ](vector, signal, std_logic, buffer,
std_logic_vector) [ ](<blank>, (4 downto 0);, (3 downto 0);, (2 downto 0);, (1 downto 0);)
);

[ ](architecture, module, component, entity) dataflyt of min_krets is
begin
  Y [ ](is:, assign to :, <=, =>)
        (A3 and S1 [ ](and, and not, or not, nand, nor, or) S0) [ ](and not, or, or not, xor, and,
xnor)
        (A2 [ ](or, or not, nand, nor, and, and not) S1 [ ](and, or, nand, or not, nor, and not) S0)
or
        (A1 [ ](nor, and not, or not, and, nand, or) S1 [ ](or not, nor, or, nand, and not, and) S0)
[ ](xnor, xor, and, or, or not, and not)
        (A0 and not S1 [ ](nor, or not, and not, nand, or, and) S0);
end;
```

Maks poeng: 5

I adressebitene er A0 er minst signifikant, og A1 mest signifikant

10



Vi har et ROM minne lik den som er gitt på figuren over. Hver dott representerer logisk '1'.

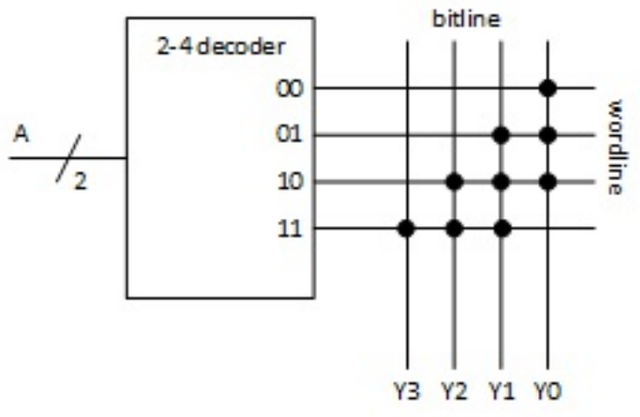
Hva blir verdien på Y om adressen (A) er 01?

Skriv tallet på vanlig form **(heltall, base 10)**: (6) .

Maks poeng: 2

Inngang A1 er mest signifikant, A0 er minst signifikant

11



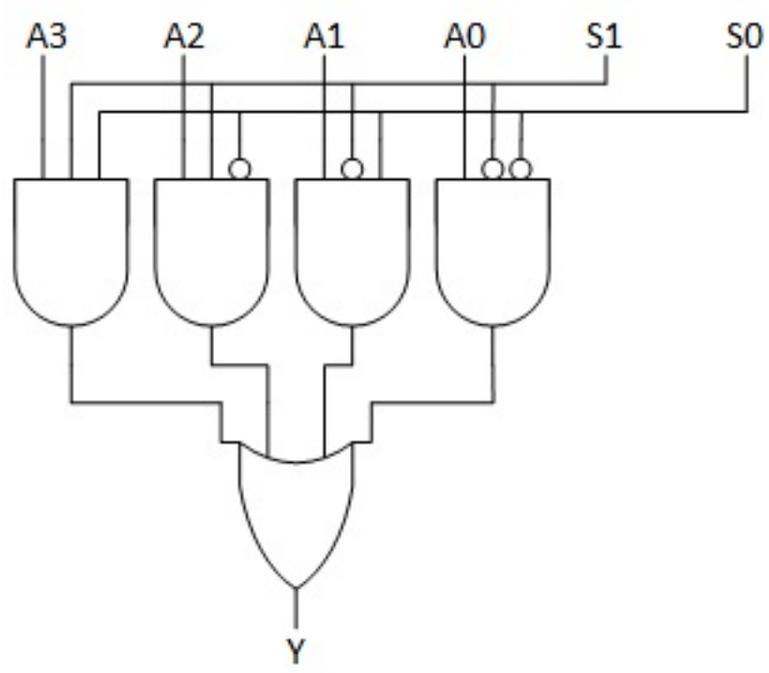
En ROM eller oppslagstabell (LUT- look up table) kan brukes som både minne og til å lage logiske funksjoner.
Bruk nedtrekksmenyene til å sette sammen den logiske funksjonen til Y2:

MERK: Alle nedtrekksmenyene må benyttes for å oppnå poeng.
Hvis et felt skal stå tomt, benytt alternativet <blank>

(<blank>, A1, not A1) (<blank>, and, nand, or, nor, xor, xnor) (not A0, A0, <blank>)

Maks poeng: 2

12

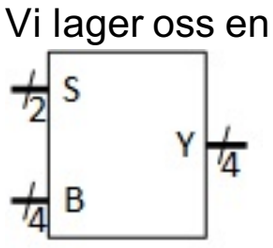


Figur 1: Krets A

Velg påstander som er sanne

a) Kretsen på figur 1 implementerer en [ ] (multiplekser, andorian, enkoder, dekoder, teller)

Vi lager oss en krets med grensesnitt som vist på figuren under:



Figur 2: Krets B.

Krets B bygges med 4 stk krets A. Vi kaller A-kretsene for krets3, krets2, krets1 og krets0.
Dette gjøres slik at:

S0 og S1 kobles til S0 og S1 i alle kretsene.

Y0, Y1, Y2 og Y3 i B kretsen kobles henholdsvis til Y i hver sin A-krets

B0 kobles til A0 i krets0, A1 i krets1, A2 i krets2 og A3 i krets3.

B1 kobles til A0 i krets1, A1 i krets2 og A2 i krets3.

B2 kobles til A0 i krets2 og A1 i krets3.

B3 kobles til A0 i krets3

A3, A2 og A1 i krets0 kobles til jord (logisk '0').

A3 og A2 i krets1 kobles til jord (logisk '0').

A3 i krets 2kobles til jord (logisk'0').

b) Hva slags krets er B?

Krets B er en 4-inputs [ ] (multiplekser, dekoder, shifter, ALU, komparator, enkoder)

Tips: Tegn en hjelpefigur på et ark.

c) Vi tar i bruk krets B, og setter B-inngangene til 0xD og S til "01" Hva blir Y?

Y utgangen blir [ ] (0xB, 0x2, 0xC, 0xA, 0x8, 0x5)

Maks poeng: 12

13 Vi ønsker å oversette følgende program til ARM assembler. Du kan anta at '*f*' ligger i '*R0*' og '*i*' ligger i '*R1*'.
Velg riktige instruksjoner under.

```
if (f == i)
   f = f + i;
else
   f = f – i;
```

[ ] (ADDS R0, R0, R1, BEQ L1, CMP R0, R1, ADD R0, R0, R1)

[ ] (ADDEQ R0, R0, R1, SUB R0, R1, R0, LSR R0, R1, R0, ADDS R0, R1, R0)

[ ] (SUB RO, RO, #1, SUBPL R0, R0, R1, SUBNE R0, R0, R1, SUBAL R0, R0, #1)

Maks poeng: 6

14 Hva er korrekt om datamaskinarkitekturer?
Velg det to alternativene som er korrekte

- [ ] Arkitekturen setter typisk grenser for hvilken klokkehastighet som prosessoren kan kjøre på.
- [ ] ARM arkitekturen støtter svært mange kompliserte instruksjoner.
- [ ] Arkitekturen definerer hvordan vi skal implementere prosessoren.
- [ ] Hovedmålet med en RISC arkitektur er å få en raskest mulig prosessor.
- [ ] Et program som er skrevet for en arkitektur er kompatibel med alle implementasjoner av denne arkitekturen. ✔
- [ ] En CISC arkitektur vil typisk ta større plass på en chip enn en RISC arkitektur. ✔

Maks poeng: 4

15 Før vi kaller en funksjon under ARM funksjonskallkonvensjonen kan det hende at vi trenger å lagre unna noen verdier, hvor kan disse lagres?
Velg de to alternativene som er korrekte

- [ ] CPSR
- [ ] R12
- [ ] R0-R3
- [ ] Stack-en ✔
- [ ] PC
- [ ] R4-R11 ✔

Maks poeng: 4

16 Dekod følgende ARM instruksjon slik det er beskrevet i læreboken, og velg de alternativene som danner tilsvarende assembler-instruksjon.

$$0xE0010002$$

(ADD, ADDS, ANDEQ, AND) (R2, R1, R0, R3) (R2, R0, R1, R3) (R2, R1, #2, R0)

Maks poeng: 4

17 Hva er korrekte utsagn om *data hazards* og *control hazards*.

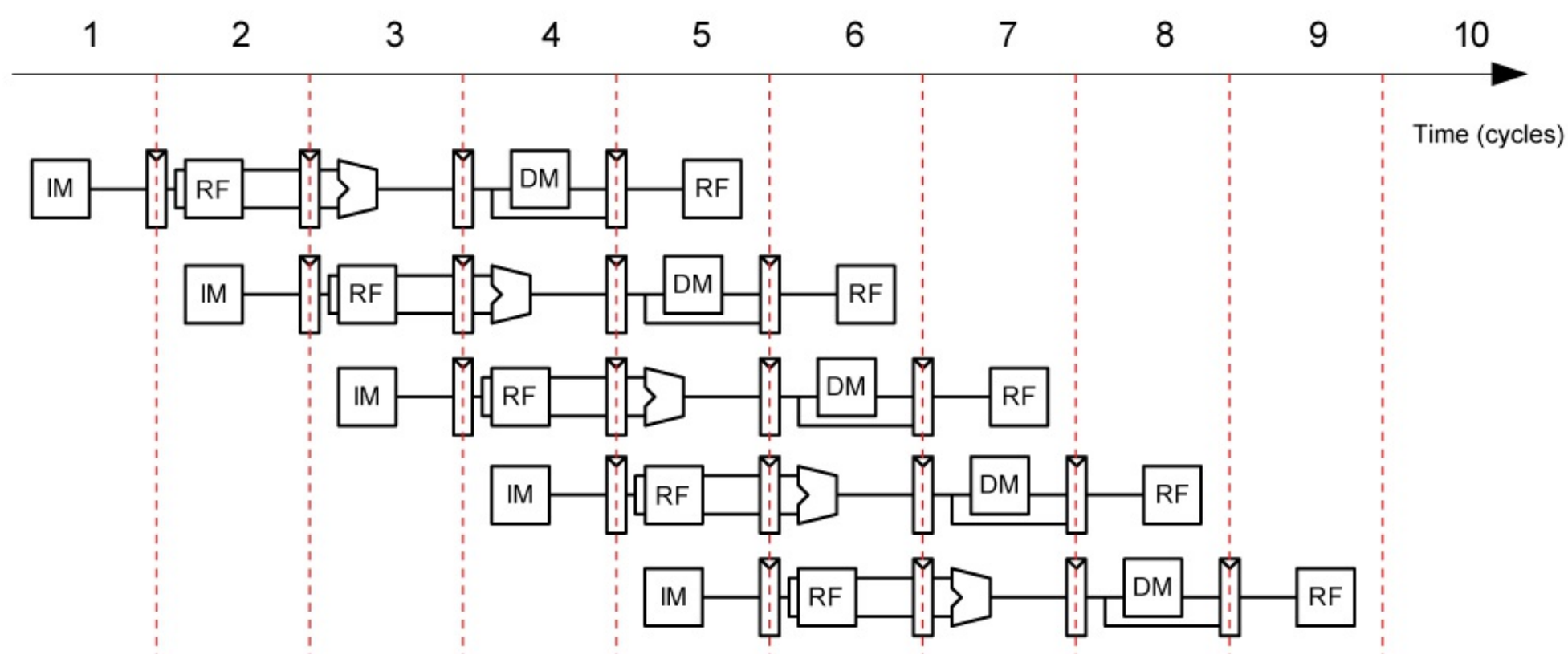
Velg de to alternativene som er korrekte

- Control hazard fører til at vi får feil under kjøring av en instruksjon.
- Data hazard oppstår når vi prøver å lese ut en verdi fra minnet.
- Å omorganisere koden under kompilering kan være en mer optimal metode for å unngå hazards, enn å legge inn NOP's. ✔
- Med en godt designet hazard-enhet kan vi unngå control hazards.
- Vi kan redusere misprediction penalty ved å bruke en tilsvarende metode som data forwarding. ✔
- Stalling er en effektiv løsning for control hazard problemet.

Maks poeng: 4

18 Gitt følgende ARM-assemblerprogram:

```
SUB R0, R1, R2
MOV R3, #24
SUB R7, R1, R5
AND R6, R6, R0
ORR R4, R6, R5
```



Hva slags pipelineforløp vil koden over gi? Anta en 5-stegs pipelinet prosessor som illustrert over (tilsvarende som i boka), men uten noen form for hazard-håndtering. Her skriver vi til registerfilen i første halvdel av klokkeperioden, og leser av i andre halvdel av klokkeperioden. Velg de alternativene under som er korrekte for pipelineforløpet.

Hva slags register aktivitet har vi i følgende klokkesykler?

- I sykel 3 (leser fra registeret, skriver til registeret, leser fra og skriver til registeret, ingen register aktivitet)
- I sykel 4 (leser fra registeret, skriver til registeret, leser fra og skriver til registeret, ingen register aktivitet,)
- I sykel 5 (ingen register aktivitet, leser fra og skriver til registeret, leser fra registeret, skriver til registeret)

Hvilken hazard har vi i følgende klokkesykler.

- I sykel 4 (ingen hazard, data hazard, control hazard)
- I sykel 5 (ingen hazard, control hazard, data hazard)
- I sykel 6 (control hazard, data hazard, ingen hazard)

Maks poeng: 6

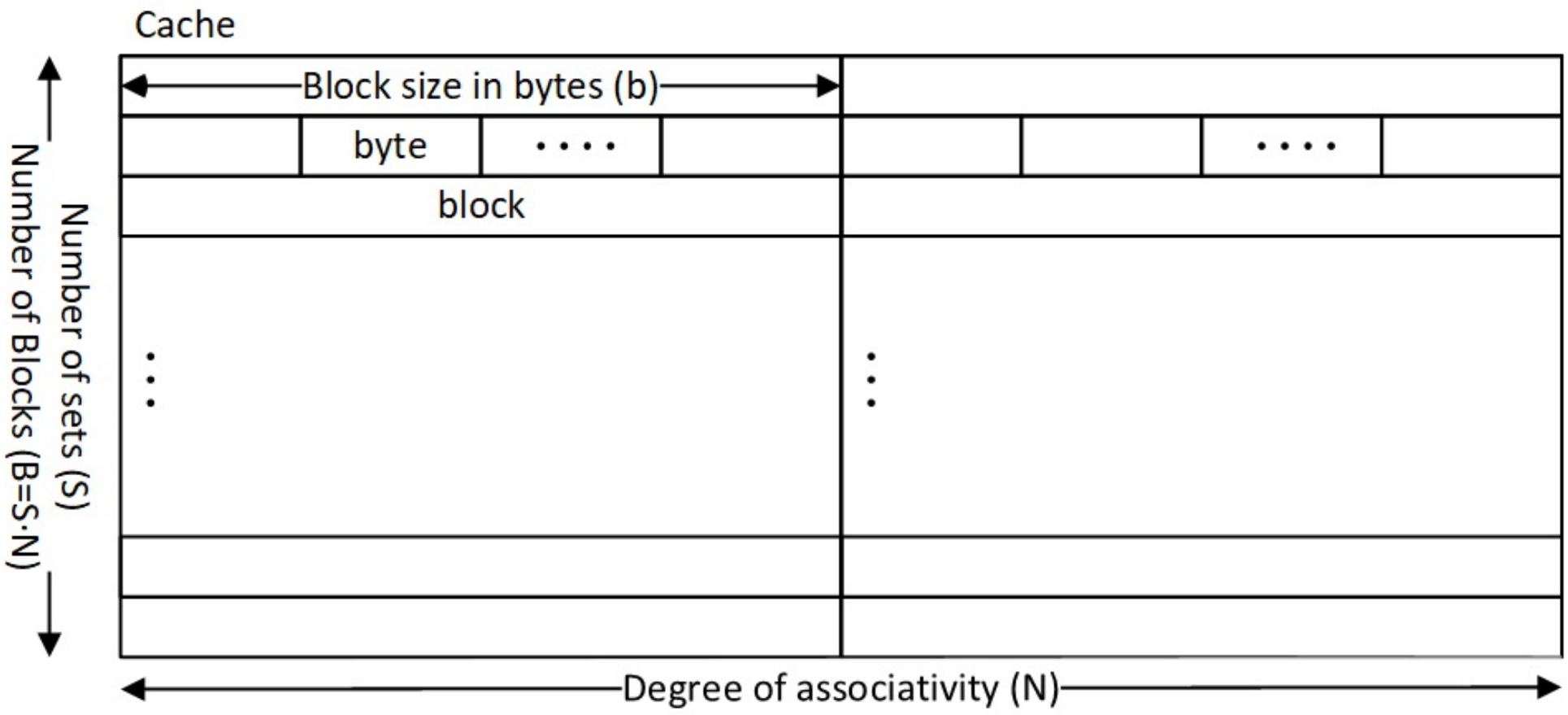
19 Hva er korrekt om prosessorytelse

Velg de to alternativene som er korrekte

- MIPS (Million Instructions Per Second) er en god målestokk for å sammenligne ytelsen til prosessorer som er basert på forskjellige arkitekturer.
- En instruksjon vil ta kortere tid å utføre på single-cycle prosessor enn på en tilsvarende pipeline prosessor. ✔
- Vi kan til en viss grad øke ytelsen til en prosessor ved å øke antall pipeline steg ✔
- Vi må bruke statistikk for å beregne CPI til single-cycle prosessorer.
- CPI'en til en pipeline prosessor øker proporsjonalt med antall pipeline steg.
- Antall steg i en multi-cycle prosessor definerer CPI'en.

Maks poeng: 4

20



Figuren over viser en generisk cache.
Til et fysisk minne på 256MB bruker vi en direkte-mappet cache (N=1) på 64kB og med blokkstørrelse på 1 ord. Prosessorarkitekturen bruker en ordstørrelse på 4 byte per ord, og hver byte har en egen adresse.

I hver av boksene skal det et heltall. I denne oppgaven må svarene være eksakte.

a) Hvor mange bit må vi ha i adressene til det fysiske minnet?

(28)

b) Hvor mange bit blir det i blokk-offset for cachen?

(0)

c) Hvor mange set vil cachen ha? fyll inn toerpotensen (e.g. *10* for 1024, siden 1024 = 2^*10*)

2^ (14)

d) Hvor mange bit vil adresse-tagene til cachen ha?
(I denne oppgaven forutsetter vi at vi kun benytter fysiske addresser, ikke virtuelle)

(12)

Maks poeng: 8

21 *I denne oppgaven skal du oppgi tallsvar. Tallsvarene kontrolleres med en presisjon på 2 desimaler. Det er greit å benytte flere desimaler, men ikke nødvendig ved korrekt avrunding.*

a) Vi har en direkte-mappet cache med kapasitet på 16 ord, og en blokkstørrelse på 4 ord. Ordstørrelsen er på 4 byte.

Vi utfører en sekvens av minneoperasjoner **to ganger.**

Sekvensen av adresser som blir benyttet er som følger

0x10, 0x14, 0x38, 0x54, 0x30, 0x34

Hva blir **hit-raten** til utføringen? [ ] (0,58 - 0,59)

b) Hvis vi utfører den samme operasjonen (sekvensen kjøres to ganger) med en arkitektur med en to-veis, set-assosiativ cache, med en kapasitet på 16 ord, blokkstørrelse på 2 ord og ordstørrelse på 4 byte. Vi forutsetter at blokken det er lengst siden ble benyttet byttes ved konflikt (LRU/ least recently used replacement).

Hva blir **miss-raten** denne gangen ? [ ] (0,58 - 0,59)

Hva konvergerer **miss-raten** til dersom sekvensen gjentas uendelig mange ganger med denne arkitekturen?

[ ] (0,495 - 0,505)

Maks poeng: 10