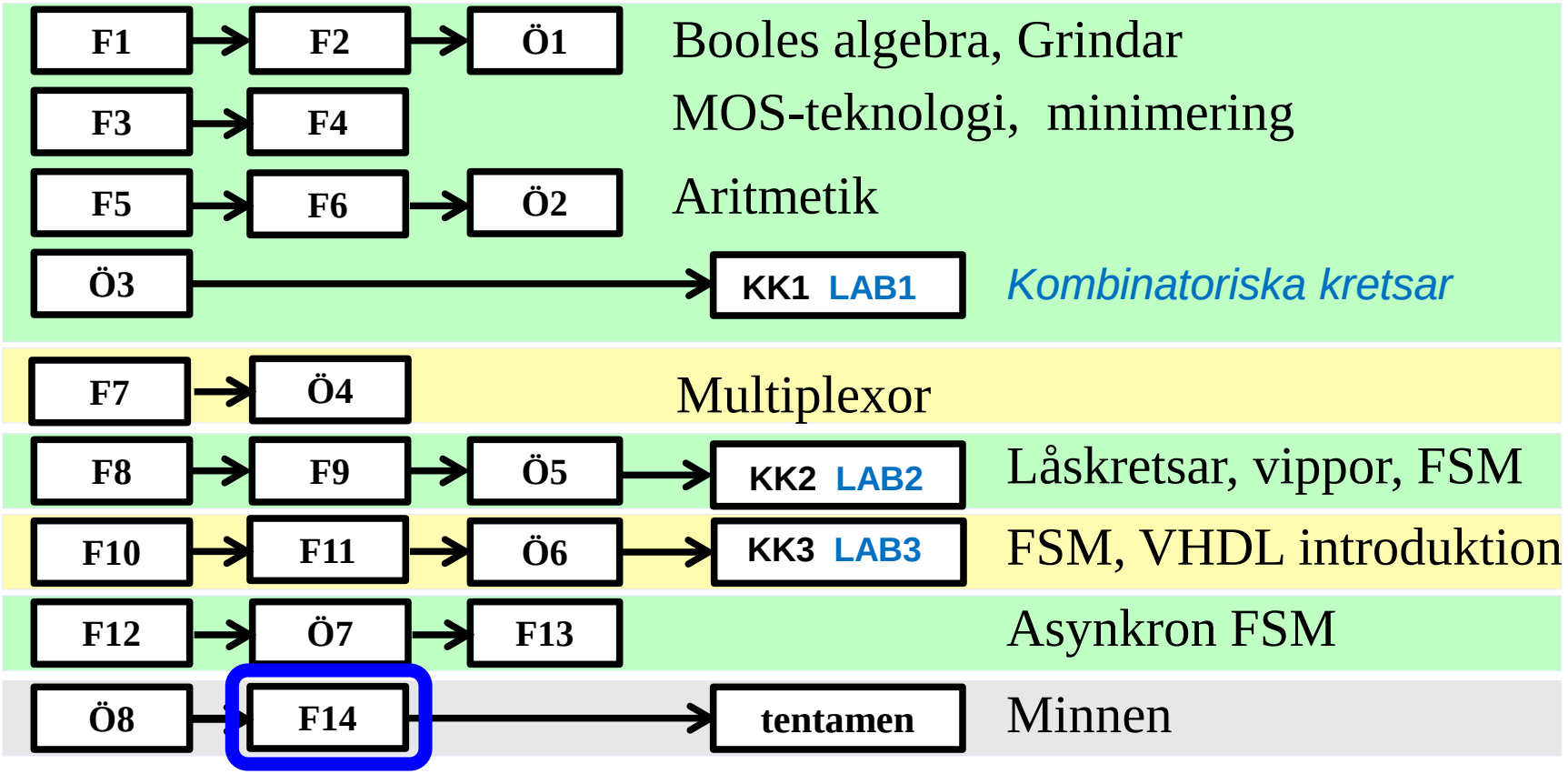
Digital Design IE1204

F14 Halvledarminnen, mikrodatorn

william@kth.se

IE1204 Digital Design



Föreläsningar och övningar bygger på varandra! Ta alltid igen det Du missat!
Läs på i förväg – delta i undervisningen – arbeta igenom materialet efteråt!

William Sandqvist william@kth.se

Detta har hänt i kursen …

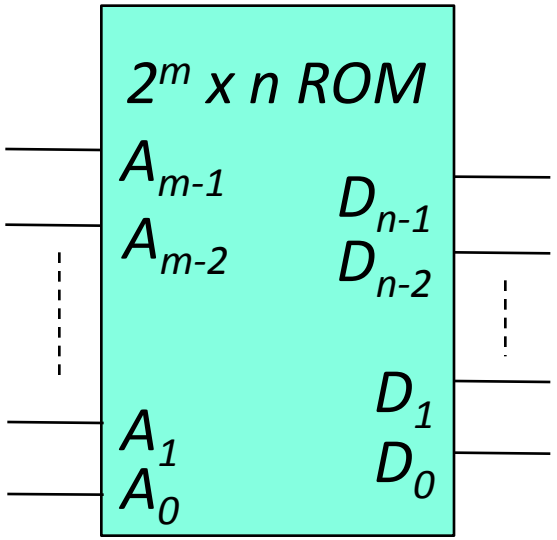
Decimala, hexadecimala, oktala och binära talsystemen
AND OR NOT EXOR EXNOR Sanningstabell, mintermer Maxtermer PS-form Booles algebra SP-form deMorgans lag Bubbelgrindar Fullständig logik NAND NOR CMOS grindar, standardkretsar Minimering med Karnaugh-diagram 2, 3, 4, 5, 6 variabler
Registeraritmetik tvåkomplementrepresentation av binära tal
Additionskretsar Multiplikationskrets Divisionskrets
Multiplexorer och Shannon dekomposition Dekoder/Demultiplexor Enkoder Prioritetsenkoder Kodomvandlare
VHDL introduktion
Vippor och Låskretsar SR-latch D-latch D-vippa JK-vippa T-vippa Räknare Skiftregister Vippor i VHDL Moore-automat Mealy-automat Tillståndskod Oanvända tillstånd Analys av sekvensnät Tillståndsminimering Tillståndsmaskiner i VHDL Asynkrona sekvensnät flödestabell exitationstabell tillståndskodning

Hasard Metastabilitet

William Sandqvist william@kth.se

Läsminne Read Only Memory ROM

- Ett läsminne har addressingångar och datautgångar
- Med *m* addresslinjer kan man accessa 2^m olika minnesadresser
- På varje address finns det ett dataord på *n* bitar
- Oftast har ROM minnet också en Output Enable (OE) ingång

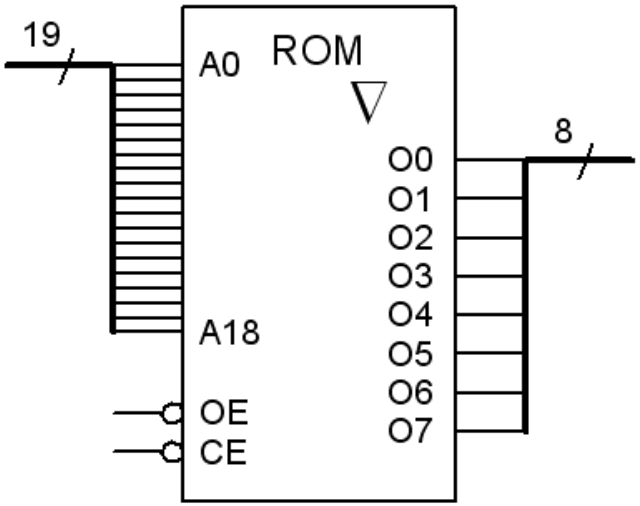


Läsminne Read Only Memory ROM

Exempel på ett ROM

Läsminne:

ROM 4M 512k × 8 bit

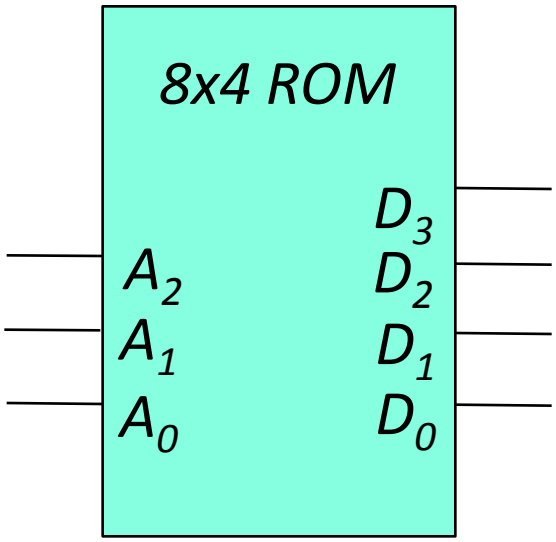


$\overline{CE}$ **Chip Enable** aktiverar chippet

$\overline{OE}$ **Output Enable** kopplar in datautgångarna
(annars är dom i Three-state läge)

William Sandqvist william@kth.se

Litet ROM



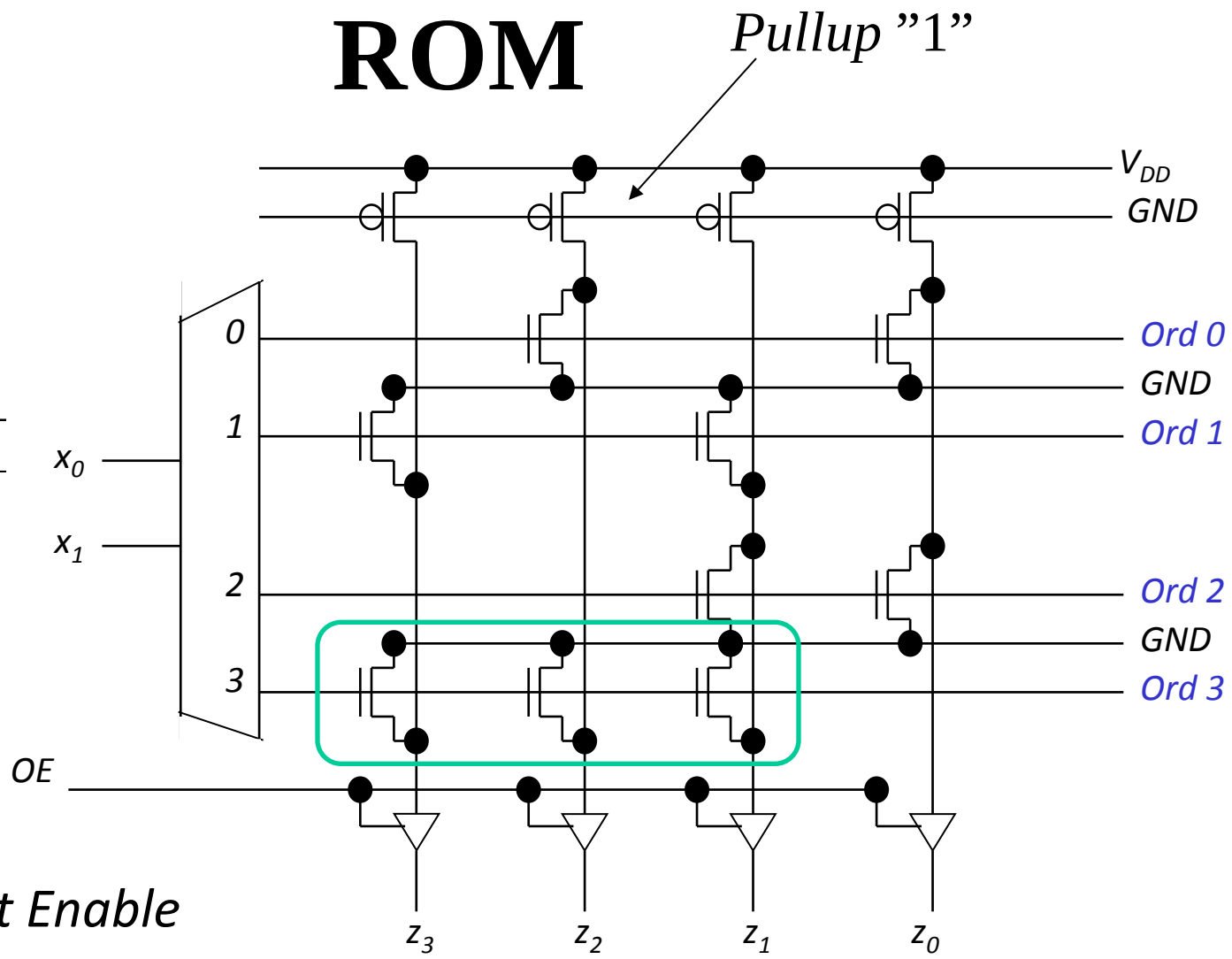
Möjlig minnesinnehåll

A_2	A_1	A_0	D_3	D_2	D_1	D_0
0	0	0	0	0	1	0
0	0	1	0	1	1	0
0	1	0	1	1	1	1
0	1	1	1	1	0	1
1	0	0	0	0	1	1
1	0	1	0	0	0	0
1	1	0	1	0	0	1
1	1	1	0	0	1	1

William Sandqvist william@kth.se

ROM

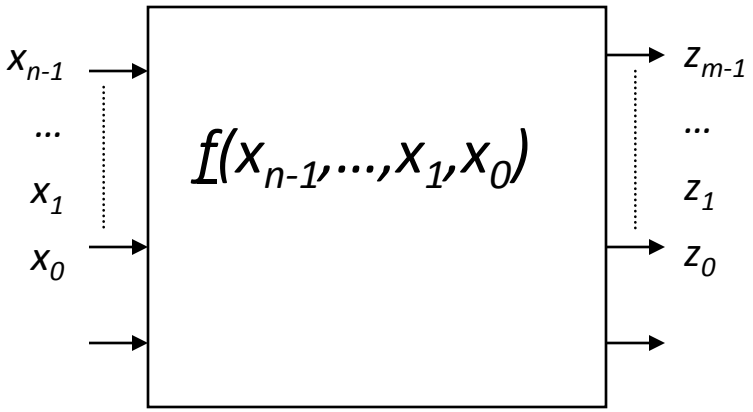
Adress	Innehåll
x_1x_0	$z_3z_2z_1z_0$
00	1010
01	0101
10	1100
11	0001



OE=Output Enable

William Sandqvist william@kth.se

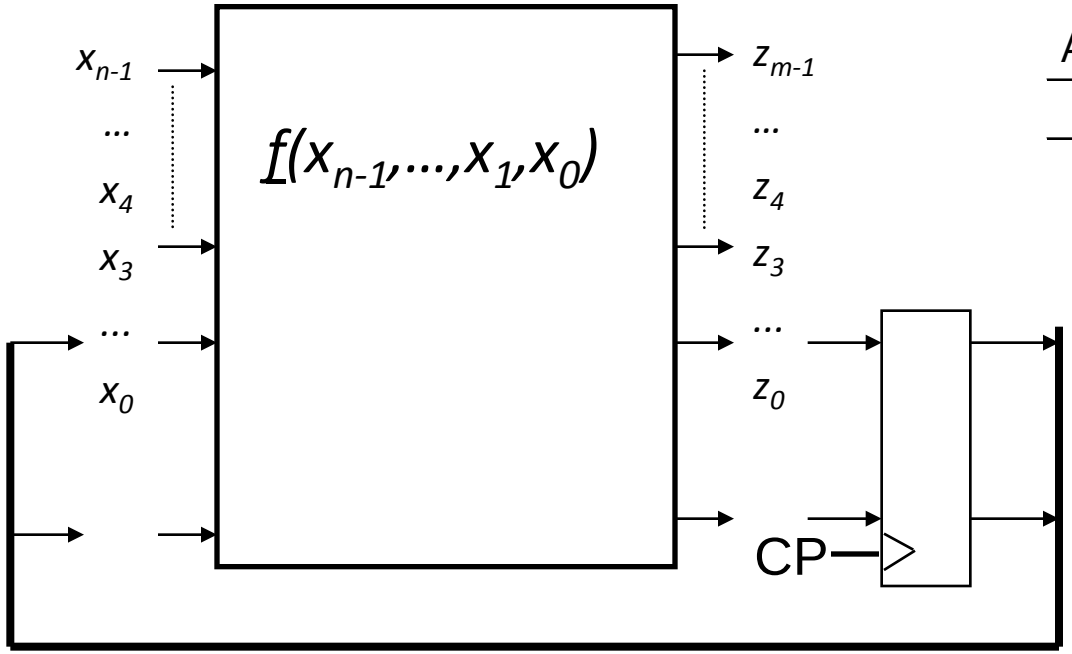
ROM implementering av kombinatoriska funktioner



Adress	Innehåll
$x_{n-1}\ldots x_1x_0$	$z_{m-1}\ldots z_1z_0$
0…00	1…10
0…01	0…01
…	…
1…11	0…01

Ett ROM med n ingångar med m utgångar kan användas för att implementera en kombinatorisk funktion med m utgångar och
2^n min-termer

ROM implementering av sekvenskrets



Adress	Innehåll
$x_{n-1}\ldots x_3x_2x_1x_0$	$z_{m-1}\ldots z_3z_2z_1z_0$
0…0000	1…0001
0…0001	0…0010
…	…
1…1111	0…0000

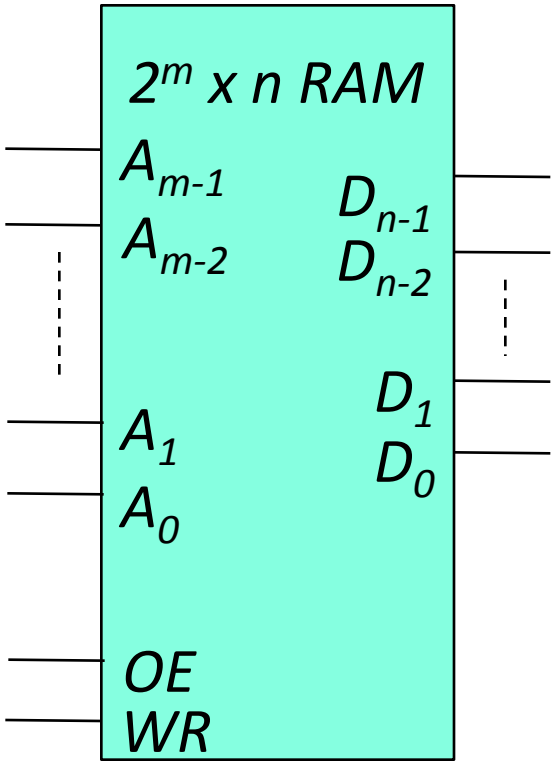
En Moore-automat = ett ROM och ett register med D-vippor

Mha återkopplingar kan ROMet användas till att generera sekvenser och implementera tillståndsmaskiner

William Sandqvist william@kth.se

Läs-Skriv-Minne
Random Access Memory RAM

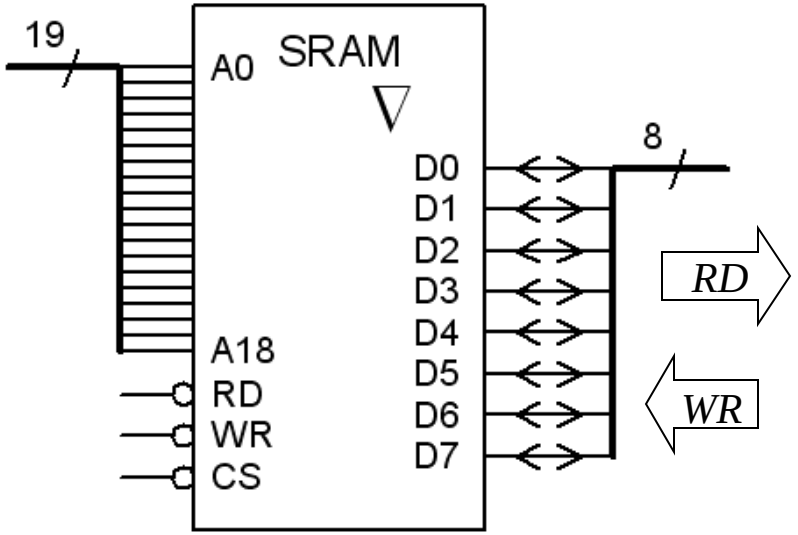
- RAM-minnet har även en Write (*WR*) ingång som möjliggör att skriva in ett dataord på en given address
- $D_{n-1} \ldots D_0$ är alltså *både* in- och utgångar



Läs-Skriv-Minne Random Access Memory RAM

Läs/Skrivminne:

SRAM 4M 512k ×8 bit



$\overline{CS}$ **Chip Select** aktiverar chippet

$\overline{RD}$ **RD** läsning från minnet, datautgångarna är aktiva

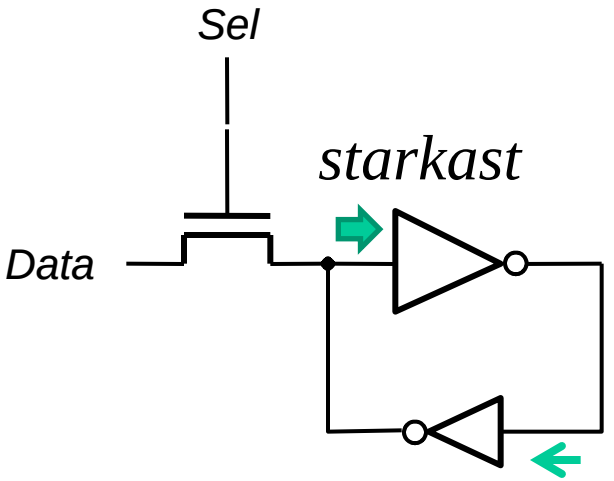
$\overline{WR}$ **WR** skrivning i minnet

(vid write är datautgångarna i three-state läge)

William Sandqvist william@kth.se

SRAM
Static Random Access Memory

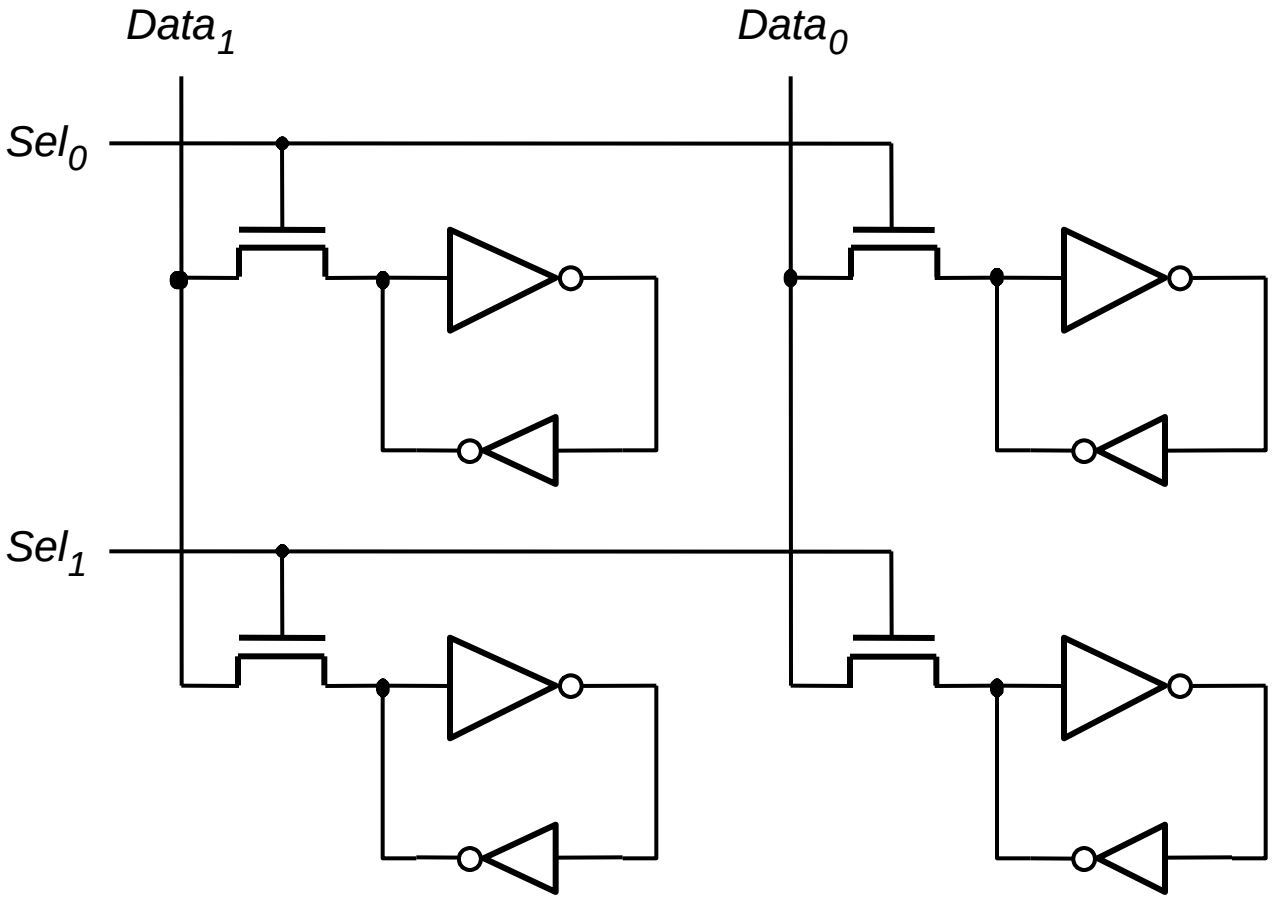
- Ett SRAM-minne innehåller en matris av SRAM-celler
- För att skriva används ‘Data’ som ingång!
  - ‘Sel’ sätts till 1 och det värdet som läggs ut på ‘Data’ sparas i cellen
- För att läsa används ‘Data’ som utgång!
  - ‘Sel’ sätts till 1, och värdet i cellen hamnar på utgången



William Sandqvist william@kth.se

SRAM

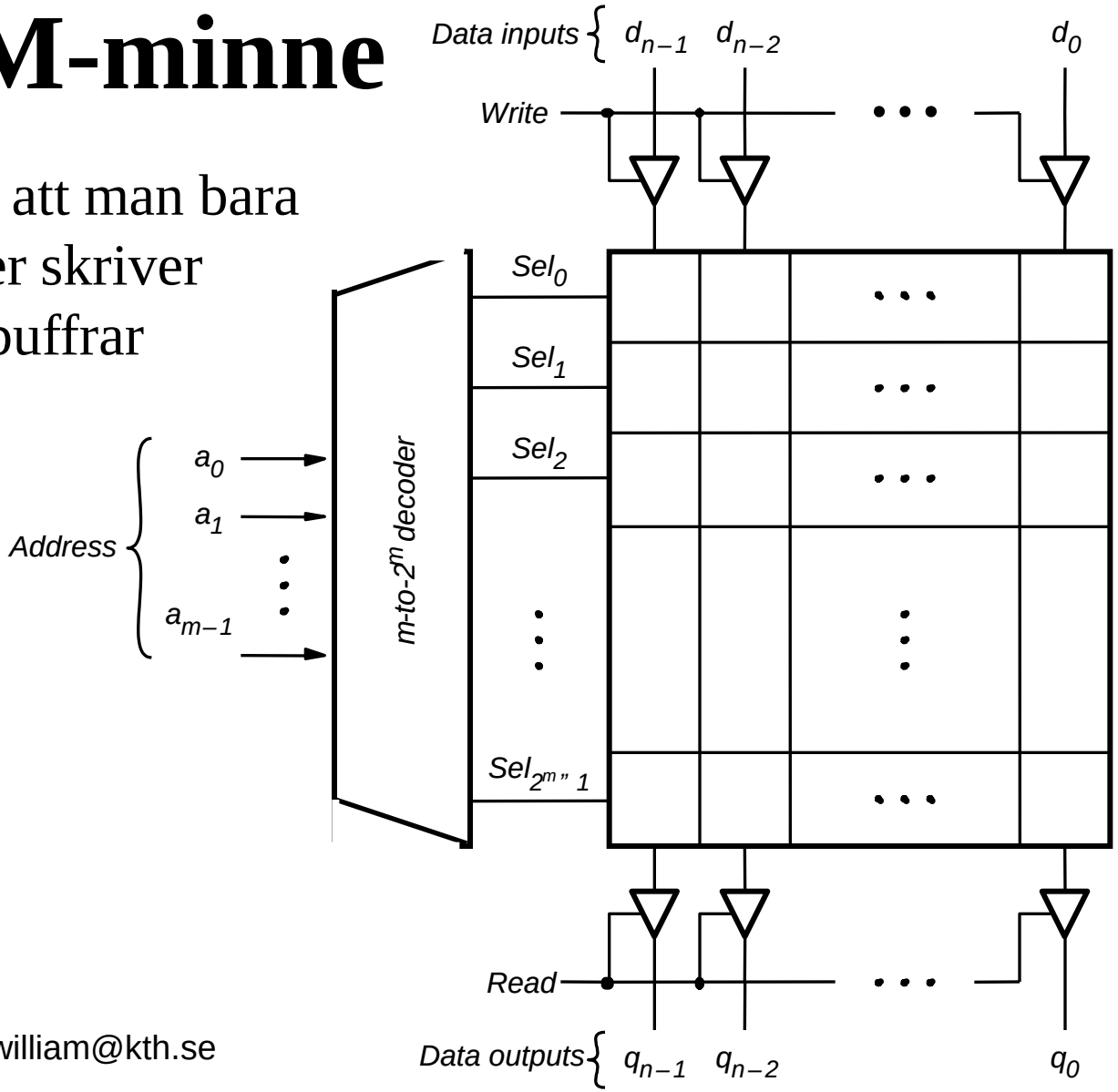
Man bildar nu en matris av 2^m x n SRAM-celler



William Sandqvist william@kth.se

SRAM-minne

För att säkerställa att man bara antingen läser eller skriver används Tristate-buffrar



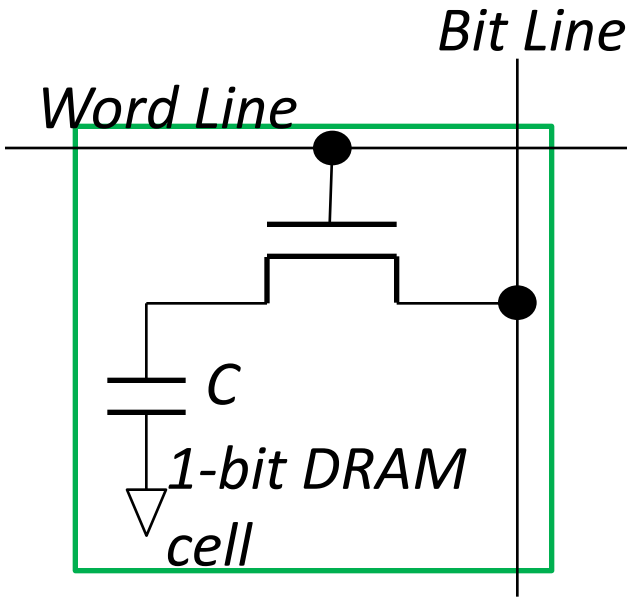
William Sandqvist william@kth.se

DRAM Dynamiskt RAM

- **SRAM** minnescellen behöver 4 transistorer och det blir för kostsamt att implementera ett stort minne
- **DRAM** minnescellerna använder bara en transistor och en kondensator

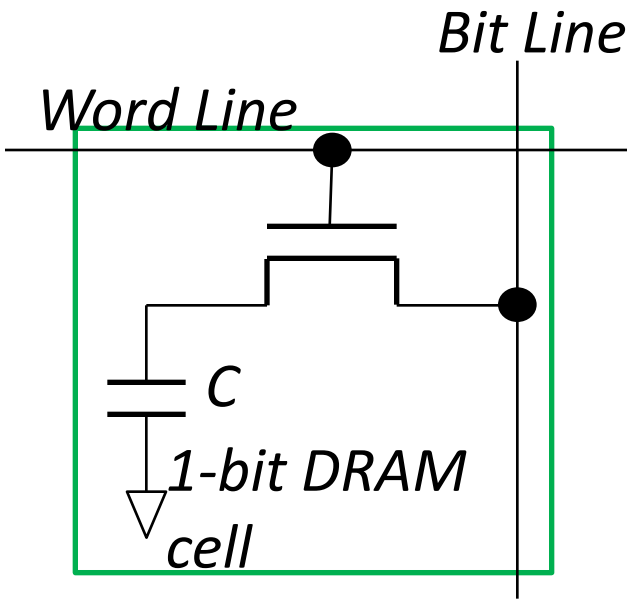
DRAM Minnescell

- DRAM-cellen består bara av en transistor och en kondensator
- Skrivning
  - För att ladda cellen ska ordlinjen sättas till ‘1’
    - Cellen får nu bitlinjens värde



DRAM Minnescell

- Läsning är mer komplex
  - Man vill inte tappa informationen vid läsning!
  - Bit-linjen sätts på en spänning mellan High and Low
  - För att läsa cellen ska ordlinjen sättas till ‘1’
    - Bitlinjens justerar nu sin spänning antingen uppåt eller nedåt
    - En extra krets (sense amplifier) detekterar ändringens rikting och skapa en riktig 0:a eller 1:a
    - Även laddningen i kondensatorn C måste återställas!



DRAM Minne

Minnesmodul med 8st kapslar

Kapsel
256Mbit (32M×8)

William Sandqvist william@kth.se

SRAM vs DRAM

- SRAM tar mer plats en DRAM men kräver en enklare accesslogik och är därför snabbare (men också dyrare)
- DRAM används för RAM-minnen i våra vanliga datorer
- När man ta bort strömmen försvinner innehållet av SRAM eller DRAM-minnet!

Minnestyper

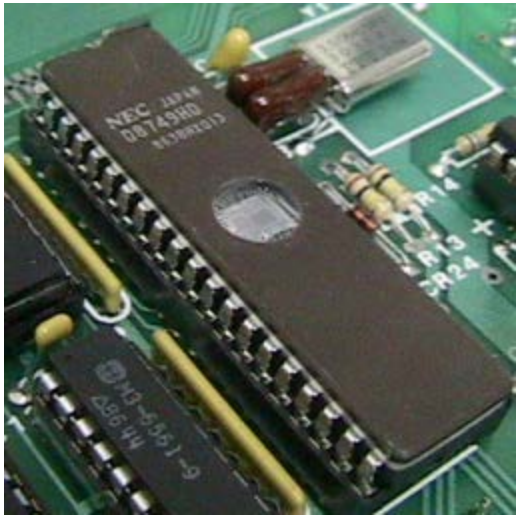
- Flyktiga minnen
  - Minnen tappar sin information om man kopplar bort strömförsörjningen
    - static RAM (SRAM)
    - dynamic RAM (DRAM)
- Icke-flyktiga minnen
  - Minnen behåller sin information om man kopplar bort strömförsörjningen
    - **Flash (blockvis skrivning)**
    - EPROM, **EEPROM (bytevis skrivning)**

Det behövs en kombination av olika minnen i en elektroteknisk konstruktion!

Flash-minne

- icke-flyktigt minne
- låg kostnad och låg effektförbrukning
- kan suddas och uppdaters, men det tar mycket mer tid än i ett RAM-minne

EPROM Erasable Programmable ROM



Programmerbart ROM-minne (kan programmeras med en kretsprogrammerare)
Erasable – kan raderas med hjälp av UV-ljus för att därefter programmeras om.
Därav ”fönstret” på chippets ovansida.

I moderna elektronikutrustningar slipper Du träffa på EPROM.

William Sandqvist william@kth.se

Minnesteknologier

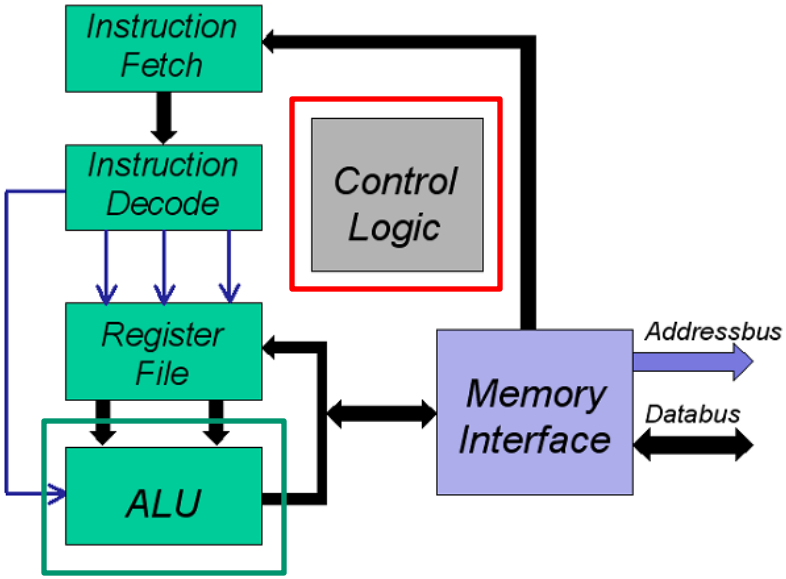
Teknologi	Accesstid	Kostnad $/GB
SRAM	1 ns	1000
DRAM	50 ns	100
HDD	10 ms	1

Snabba minnen är dyra och billiga minnen är slöa!

Principiella ”mellan tummen och pekfingret” siffror.

William Sandqvist william@kth.se

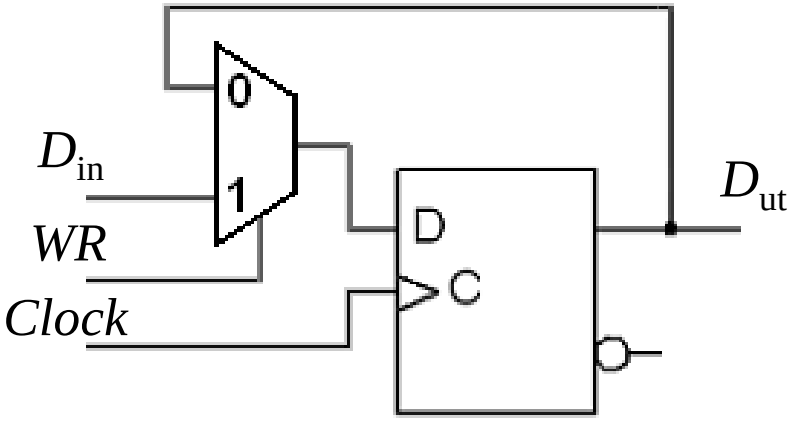
Logik i mikroprocessorn



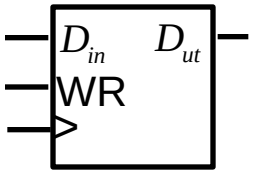
Det finns både kombinatorisk och sekventiell logik i en processor. ***Kontrollogiken*** *är en tillståndsmaskin medan* ***ALU****:n är mest kombinatorik.*

William Sandqvist william@kth.se

Registerelement



Symbol



$WR / \overline{hold}$

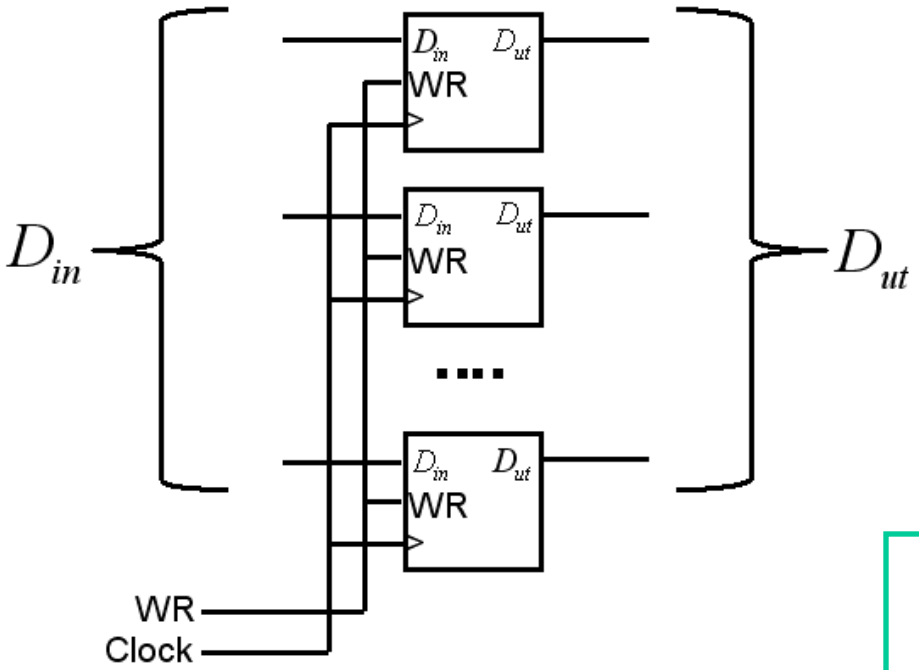
WR = 1 synkron skrivning

WR = 0 hold

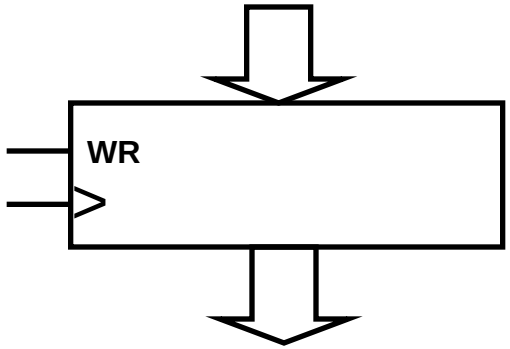
1 *logikelement i en FPGA*

William Sandqvist william@kth.se

Register



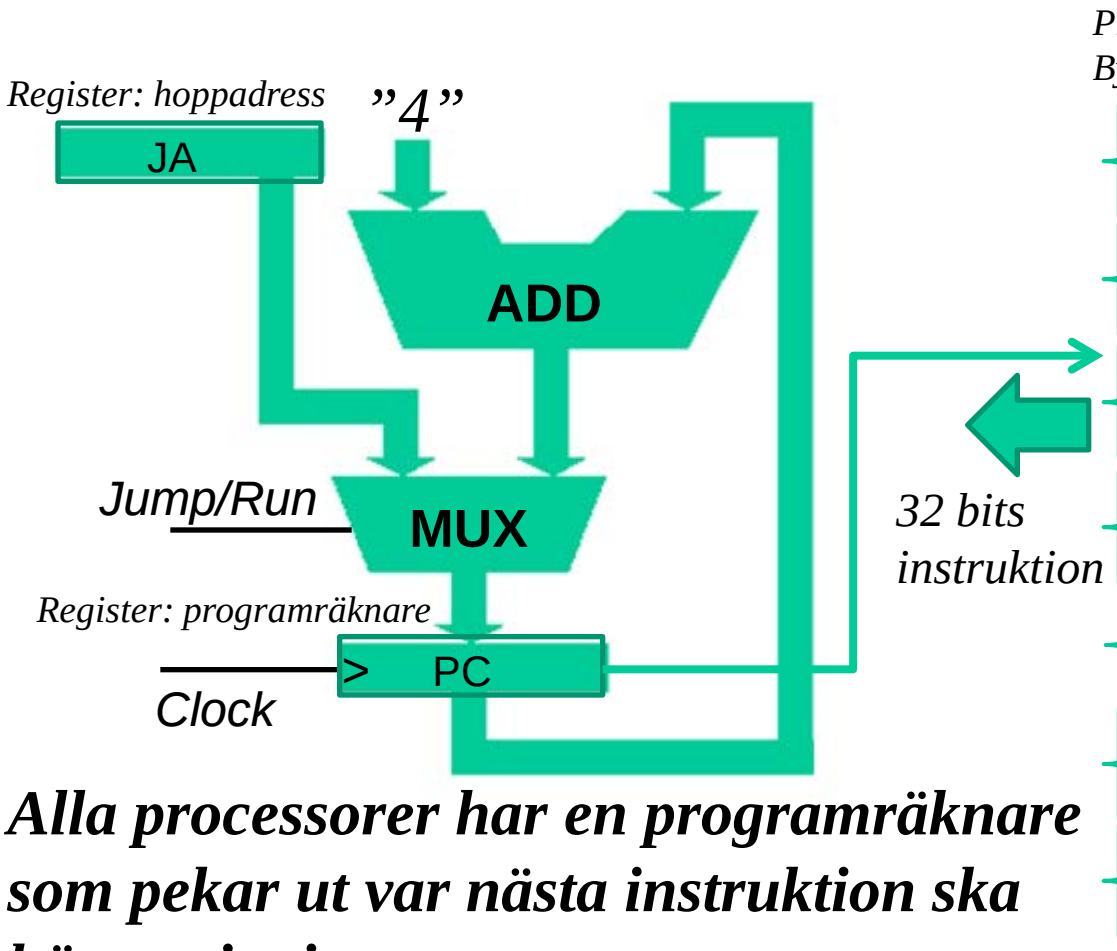
Symbol



*32 bitars register är **32** logikelement i en FPGA*

William Sandqvist william@kth.se

Programräknar-register



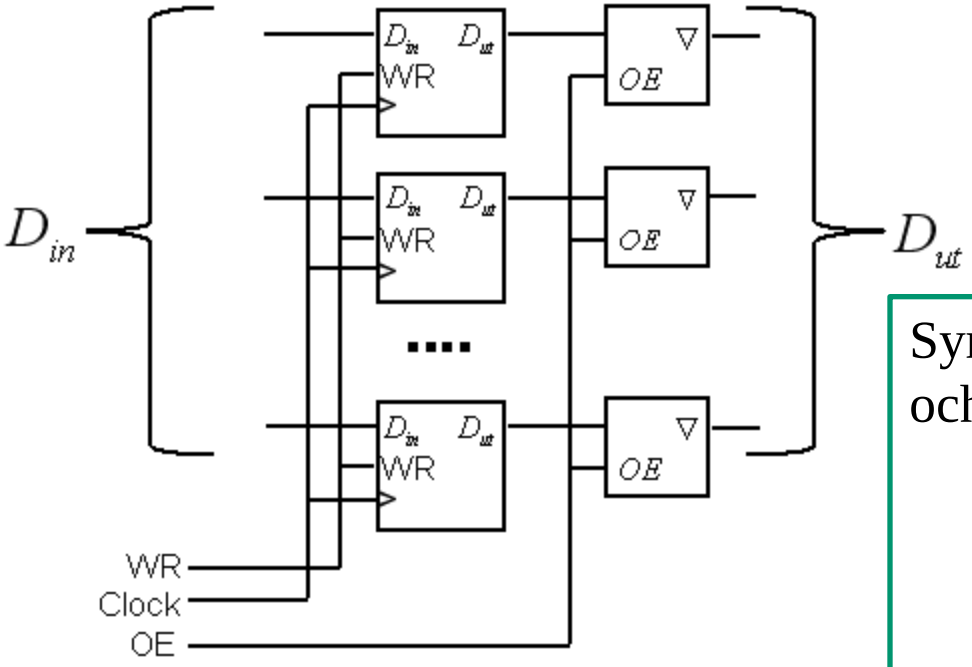
En instruktion är 4 Byte
PC, Program Counter räknas upp med ”4” efter varje instruktion.

Vid programhopp laddas PC med hoppadressen **JA** (Jump Adress) och fortsätter sedan därifrån.

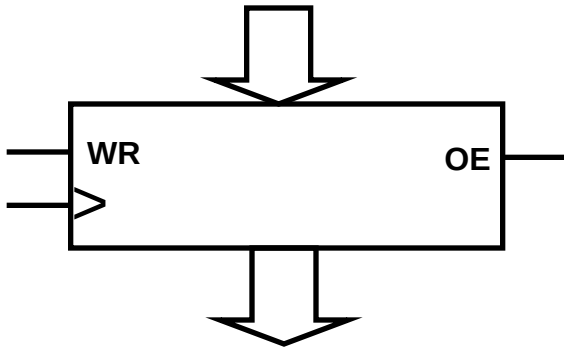
Alla processorer har en programräknare som pekar ut var nästa instruktion ska hämtas i minnet.

William Sandqvist william@kth.se

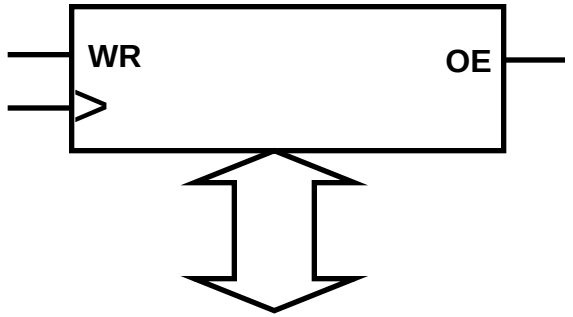
Register med threestateutgång



Symbol

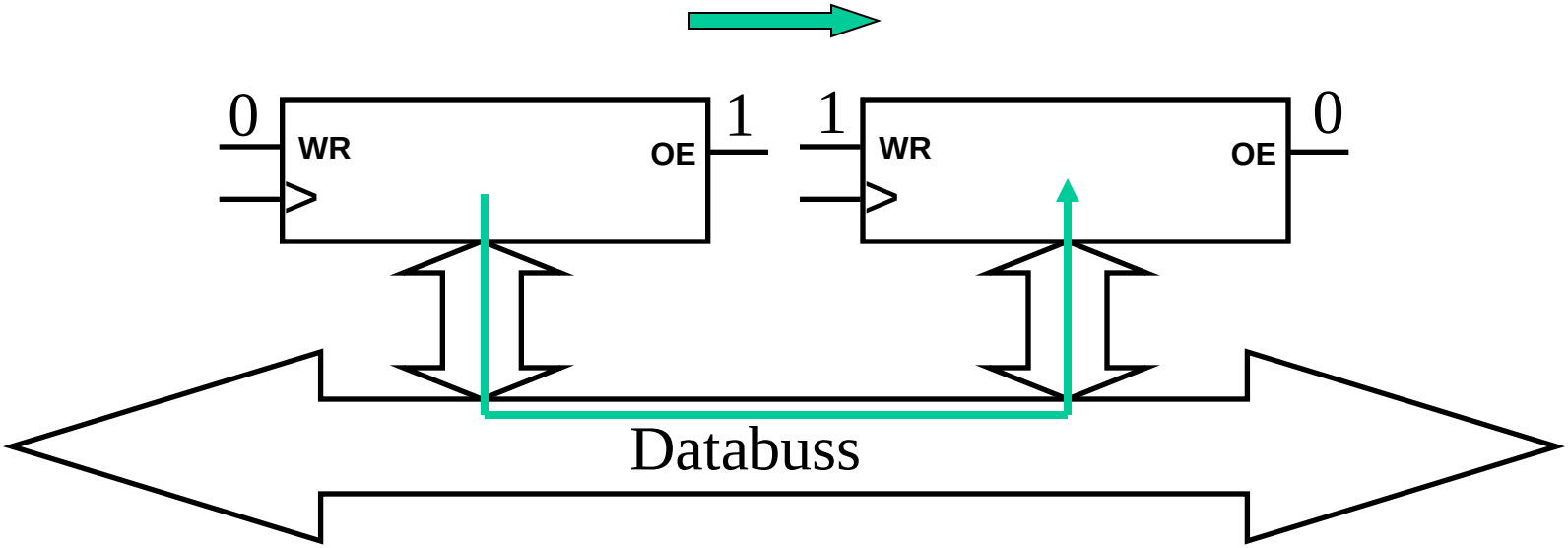


Symbol **dubbelriktat** (ingångar och utgångar är nu ihopkopplade)



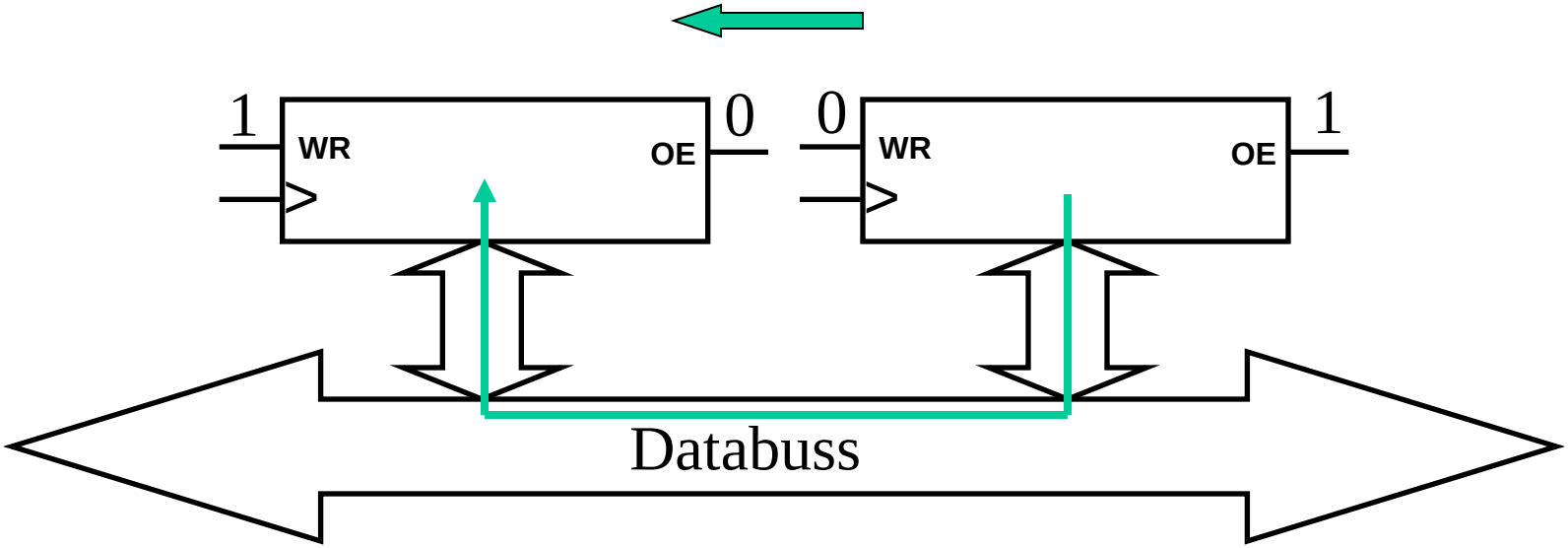
William Sandqvist william@kth.se

Register och Databuss



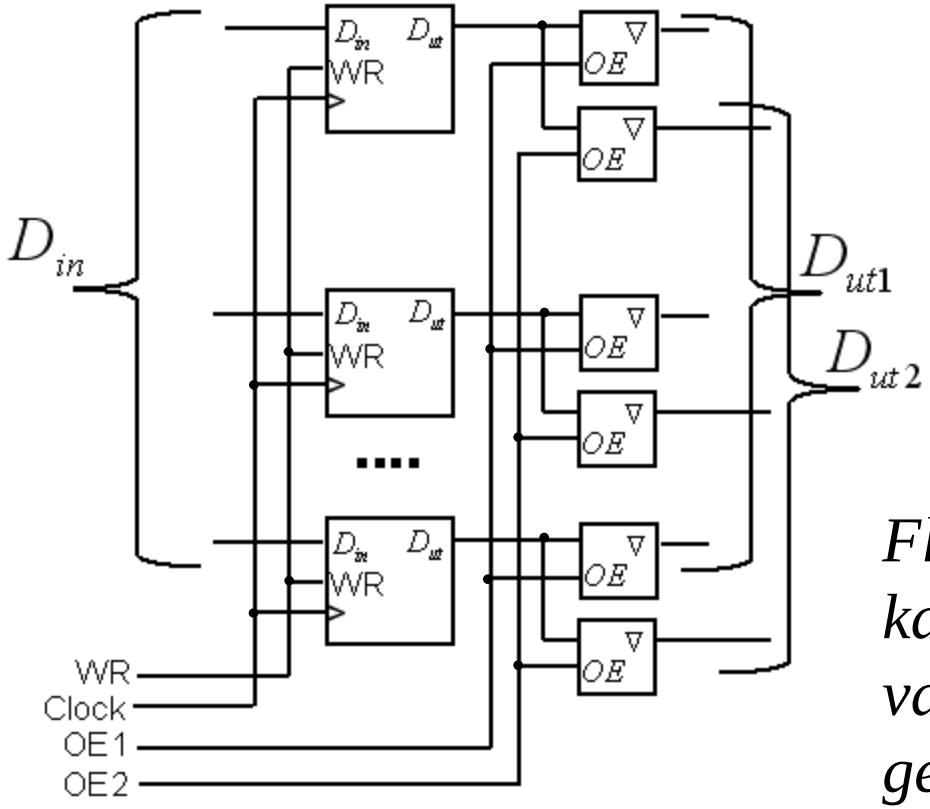
Flera dubbelriktade register med threestateutgångar kan kopplas ihop med varandra för att bilda en gemensam databuss.

Register och Databuss

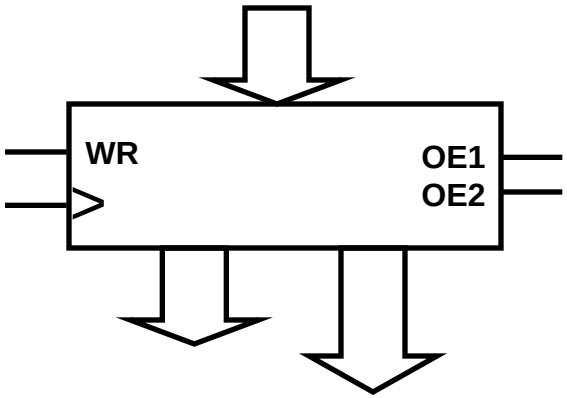


Data kan nu styras att kopieras mellan ***alla*** register på databussen.

Dubbelport register



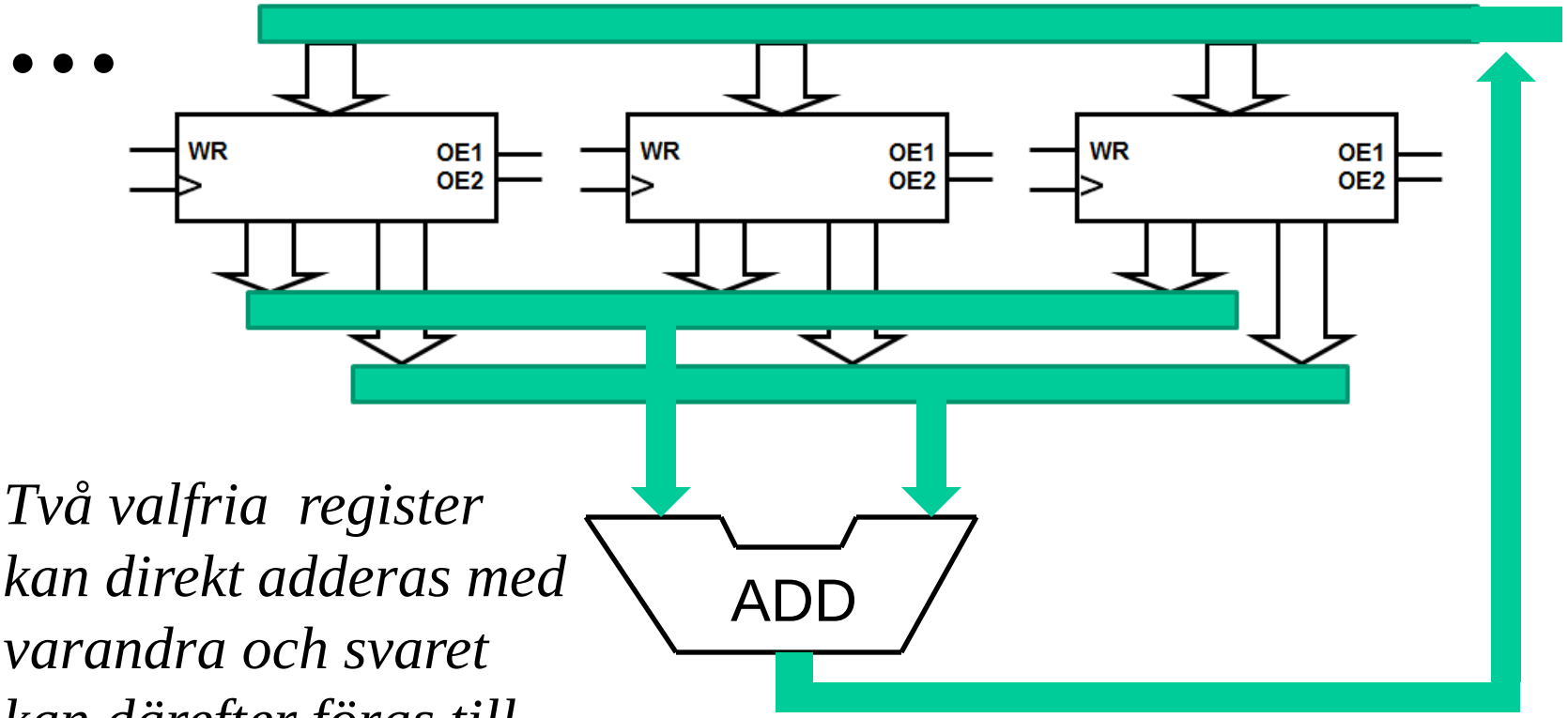
Symbol



*Flera **Dubbelportregister** kan kopplas ihop med varandra till **två** gemensamma utgångsbussar*

William Sandqvist william@kth.se

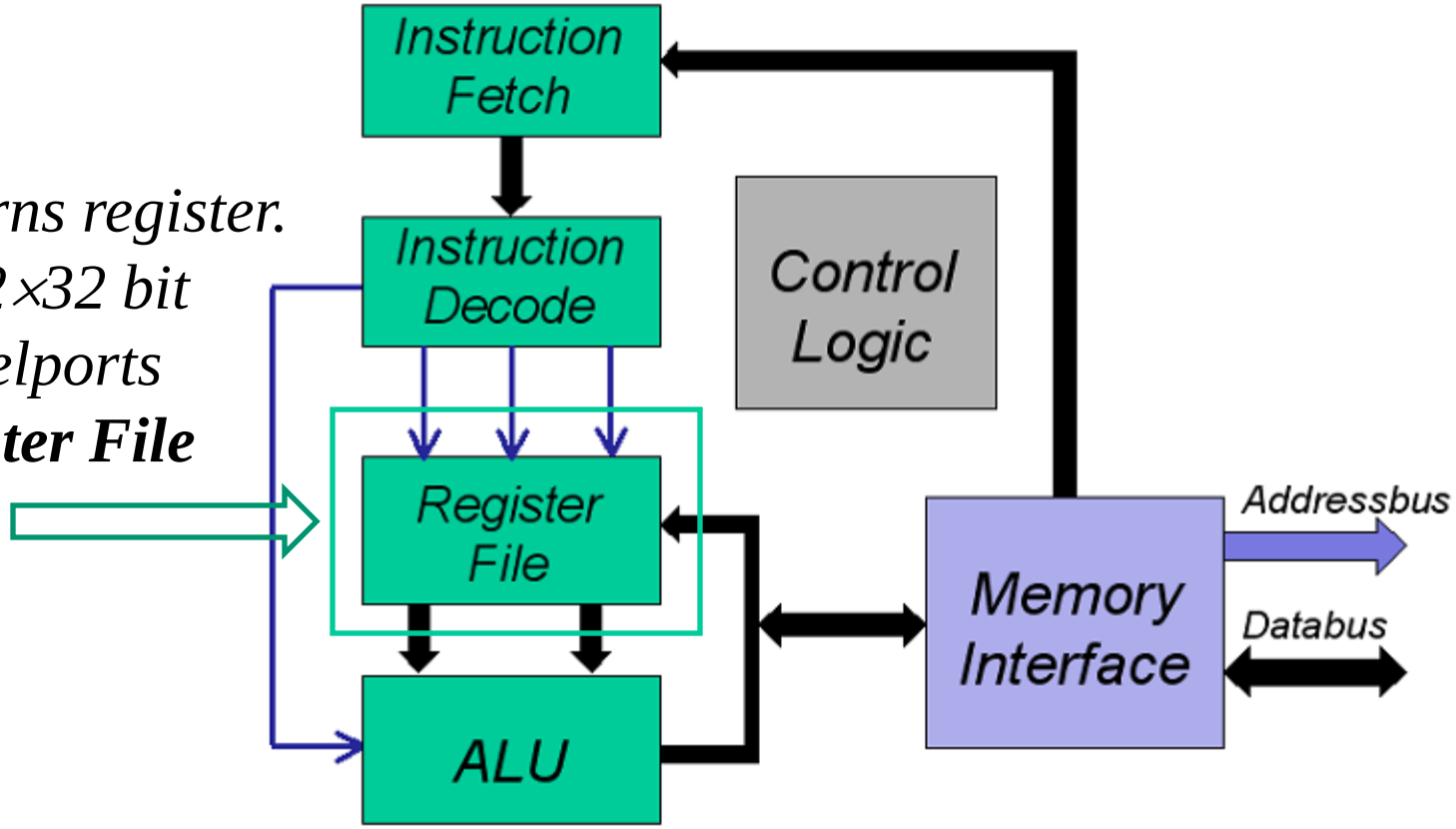
Register file



Två valfria register kan direkt adderas med varandra och svaret kan därefter föras till valfritt register

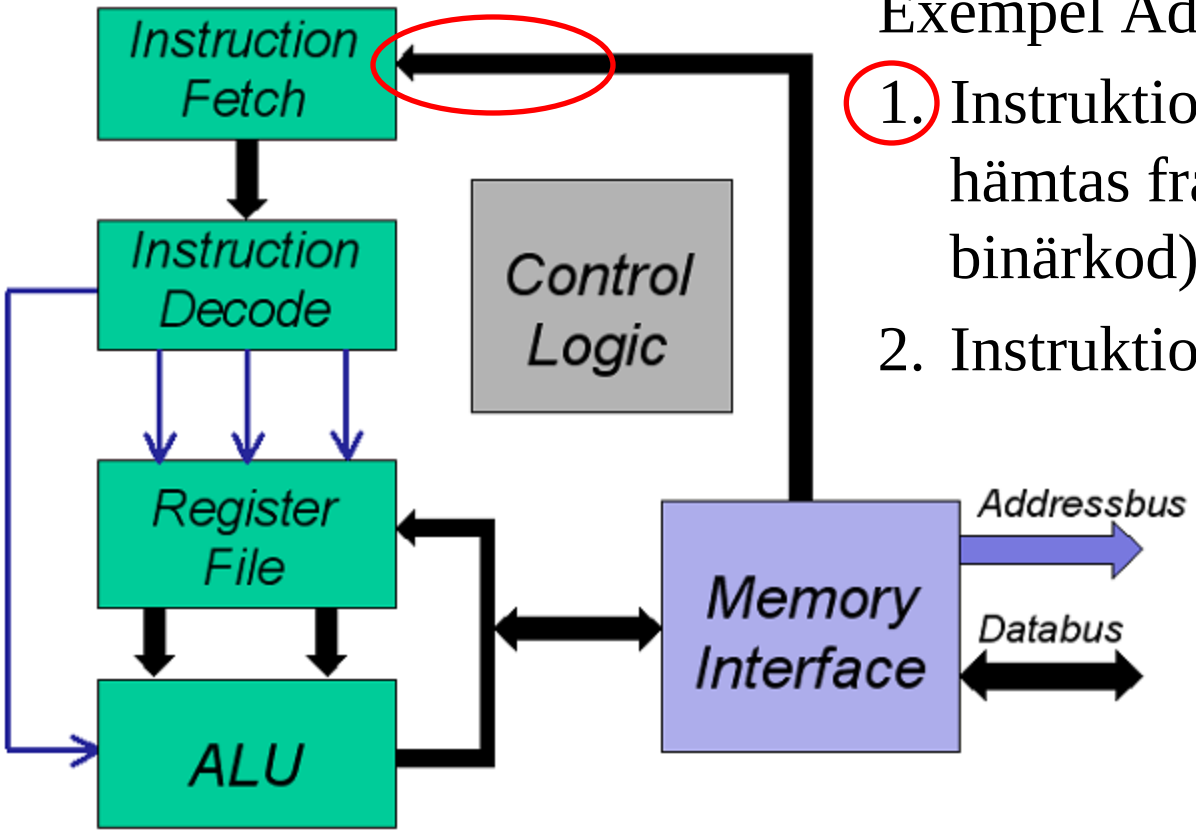
Mikrodatorn - arkitektur

Datorns register.
En 32×32 bit dubbelports
Register File



William Sandqvist william@kth.se

Mikrodatorn - **add**

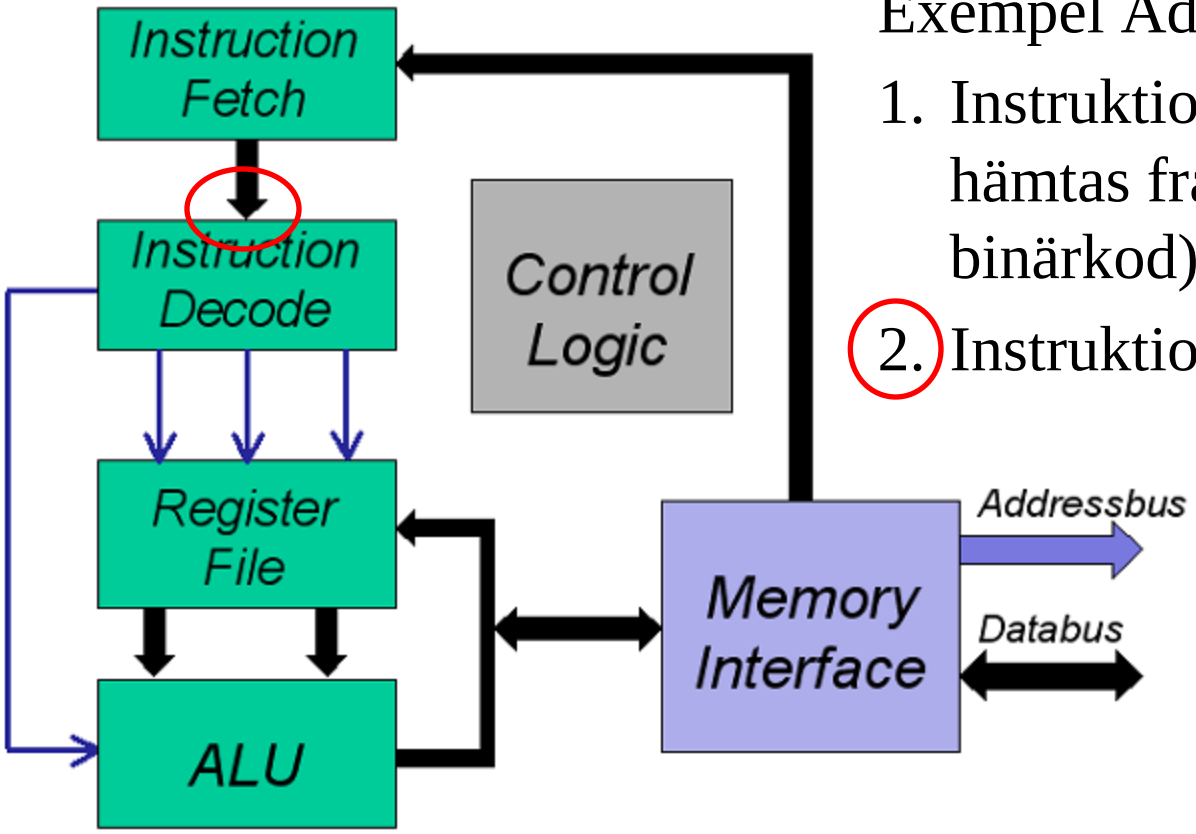


Exempel Add instruktion

1. Instruktion **`add R1,R2,R3`** hämtas från minnet (som binärkod)
2. Instruktionen avkodas

William Sandqvist william@kth.se

Mikrodatorn - **add**

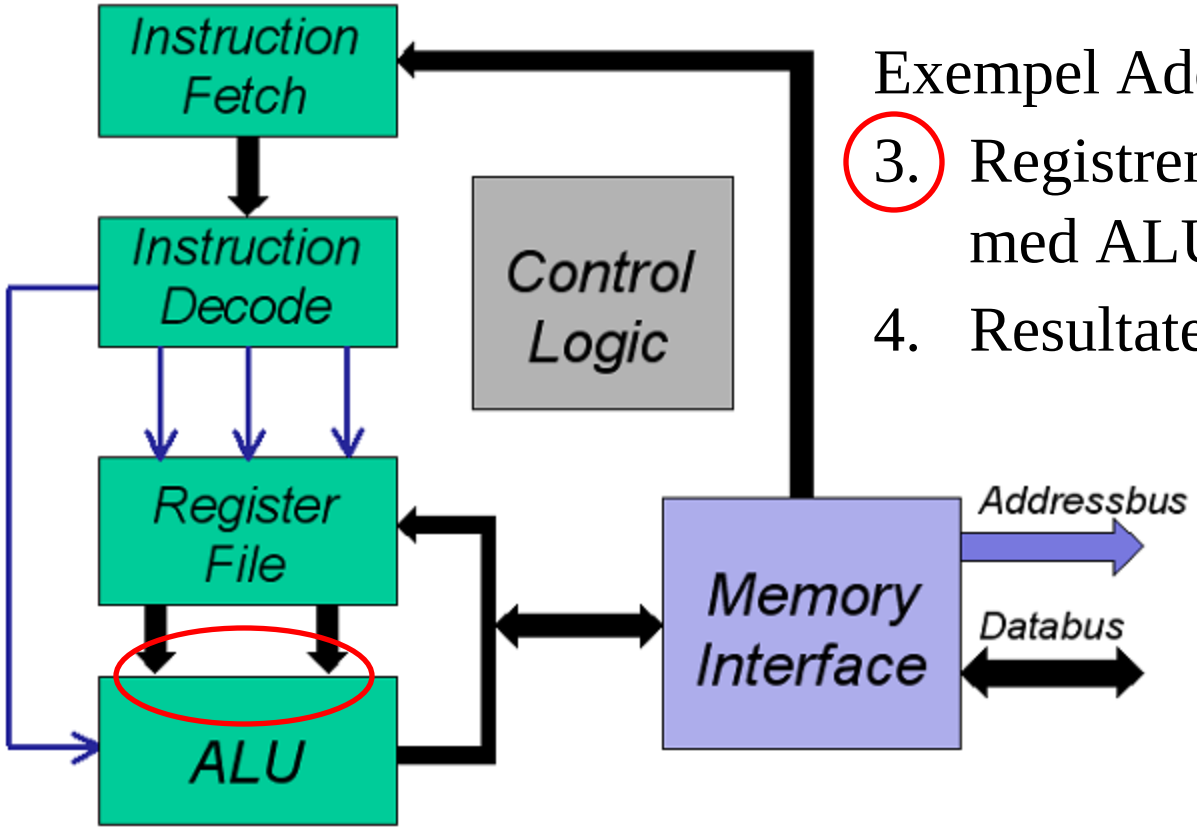


Exempel Add instruktion

1. Instruktion **`add R1,R2,R3`** hämtas från minnet (som binärkod)
2. Instruktionen avkodas

William Sandqvist william@kth.se

Mikrodatorn - **add**

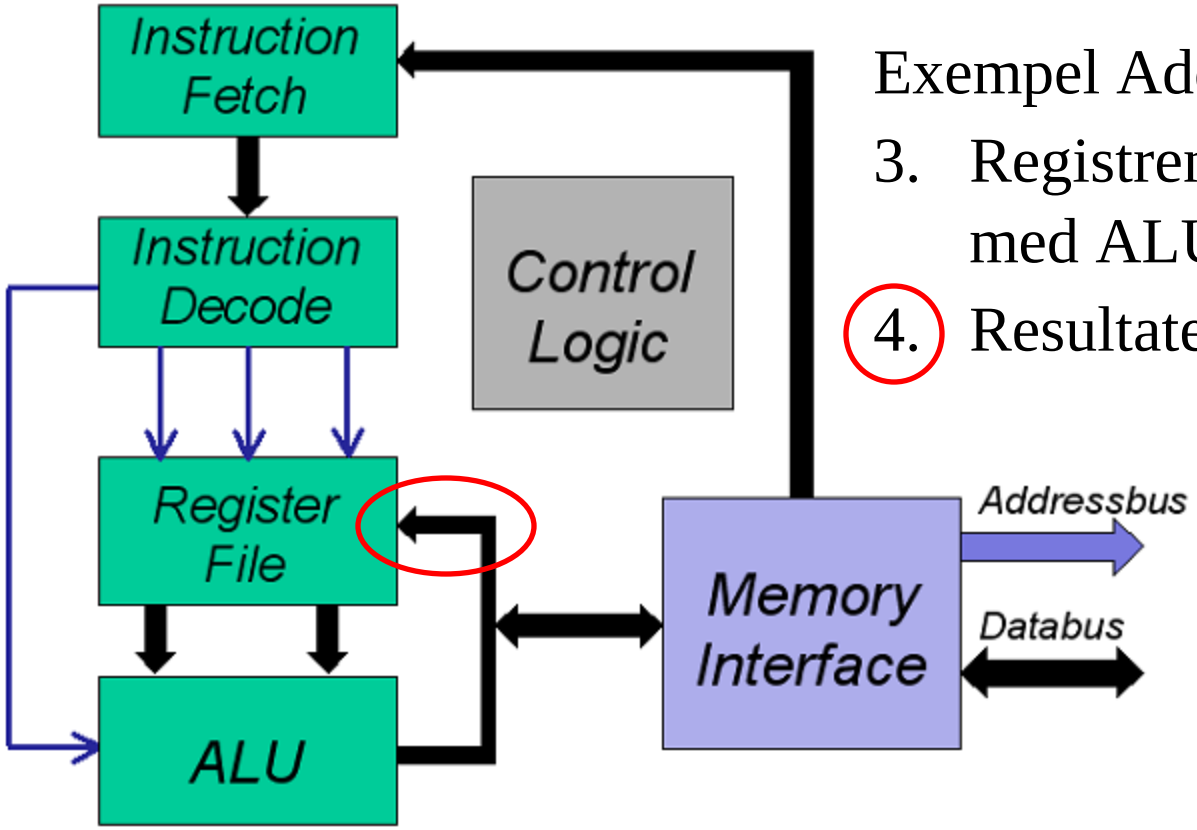


Exempel Add instruktion

3. Registren **R2** och **R3** adderas med ALU:n
4. Resultatet skrivs till register **R1**

William Sandqvist william@kth.se

Mikrodatorn - **add**



Exempel Add instruktion

3. Registren **R2** och **R3** adderas med ALU:n
4. Resultatet skrivs till register **R1**

Det krävs således flera klockpuls-perioder (här 4) för att genomföra en instruktion.

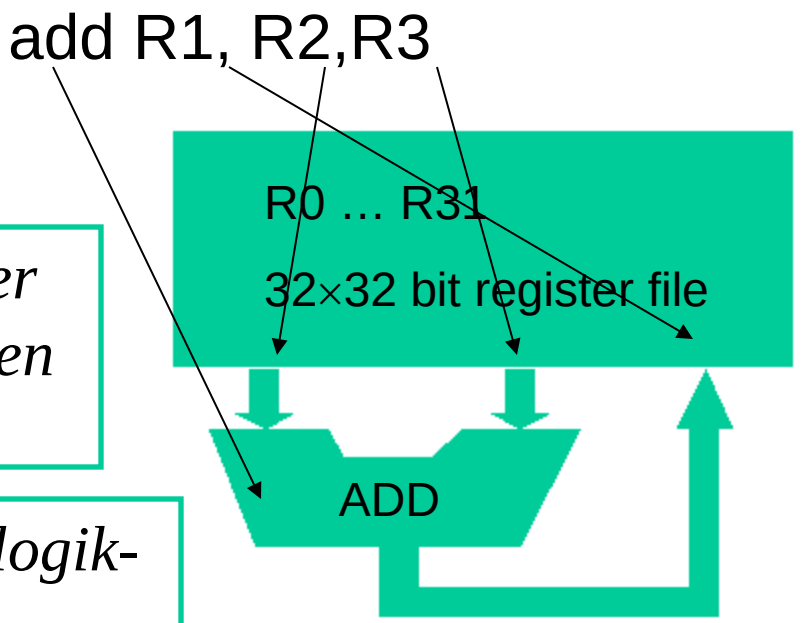
(man kan kanske ordna med en Pipeline?)

William Sandqvist william@kth.se

Register file

Processorn har en 32×32 bitars register file (med dubbelports-register). Man kan därför samtidigt *läsa* från *två* valfria register eller skriva till *ett* valfritt register per klockpuls.

Datorinstruktionen add, innebär att summan R2+R3 läggs i R1



*En register file med 32 register är 32^2 = **1024** logikelement i en FPGA*

*En 32 bitars adderare är **32** logik-element i en FPGA*

Möjligt instruktionsformat

32 bitars instruktion

```
add r1, r2, r3
```

5 bitar
1 av 32 reg

5 bitar
1 av 32 reg

5 bitar
1 av 32 reg

Operationskod – att vi vill addera

William Sandqvist william@kth.se

Tentamensläsning kombinatorik

- Karnaugh
- SP/PS
- Implementering av två-nivå-logik
- Logiska grindar
- Multiplexor
- Boolesk algebra
- Talsystem
- Aritmetik

William Sandqvist william@kth.se

Tentamensläsning sekvenskretsar

- Vippor och latchar
- Tillståndsmaskiner (Moore/Mealy)
- Tillståndsdiagram
- Tillståndsminimering
- Analys av sekvensnät
- Syntes av sekvensnät
- Tidsbeteende (Setup/Hold)

William Sandqvist william@kth.se

Tentamensläsning CMOS

- Vad är funktionen i en CMOS-krets?
- Tristate

William Sandqvist william@kth.se

Tentamensläsning VHDL

- Tolka en beskrivning

William Sandqvist william@kth.se

Tentamensläsning Asynkrona sekvensnät

- Analys
- Syntes
- Hasard

William Sandqvist william@kth.se

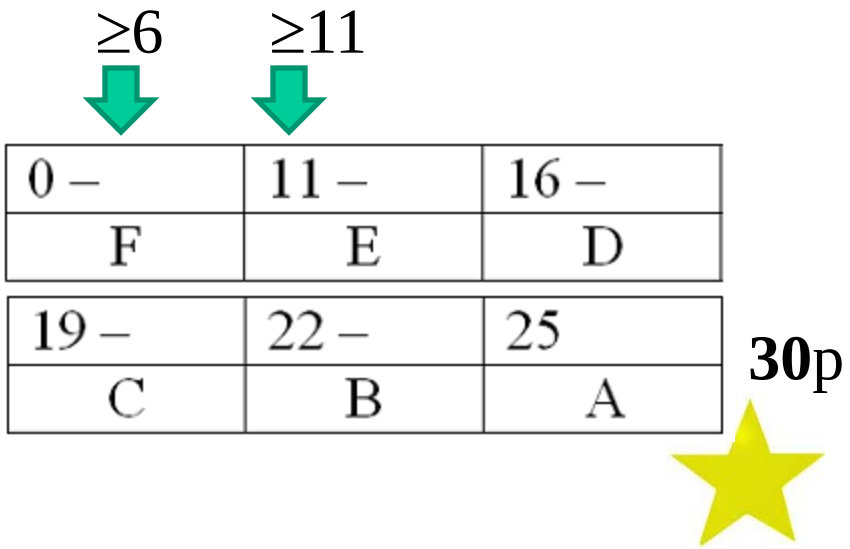
Tentamensläsning Halvledarminnen

- RAM-minnen
- ROM-minnen
- Funktion, Principbild

William Sandqvist william@kth.se

Tentamen struktur

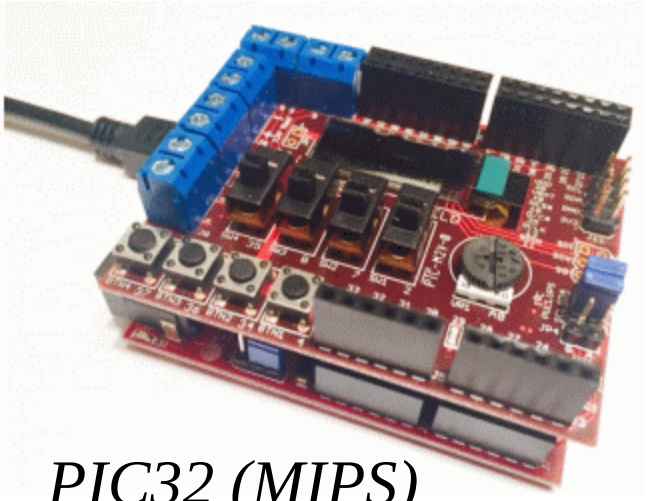
- **Del A1** (**10** poäng)
  - Fokus på ***analys***
  - Korta uppgifter (1 eller 0 poäng) ***Observera!***
- **Del A2** (**10** poäng)
  - Fokus på ***metodik***
- **Del B** (**10** poäng)
  - Fokus på ***design***
  - Problem



Nästa kurs - Datorteknik

- Hur fungerar en dator inuti?
- *Processor?*
- *Pipeline?*
- *Cacheminne?*
- *Threads?*
- *Interrupts?*
- Efter Datorteknik
  har **du** förklaringarna

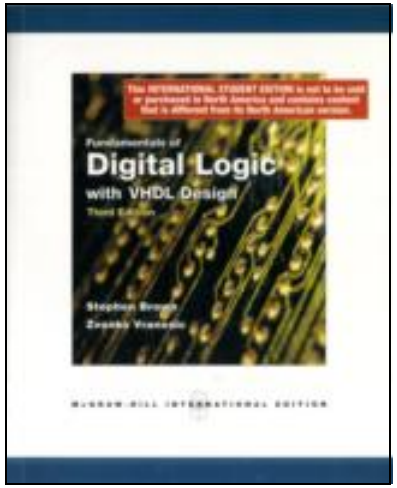
IS1200



PIC32 (MIPS)

William Sandqvist william@kth.se

Mer VHDL?

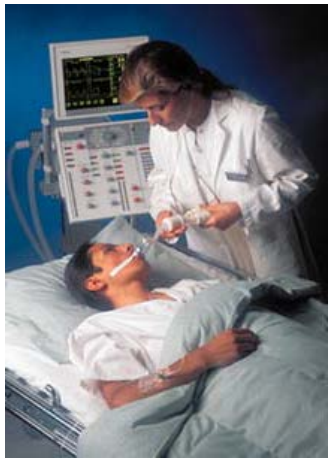


Valbar kurs IL1331 VHDL-design 7,5hp
valbar för CINTE och TCOMK obligatorisk
för TIEDB kursen går årligen i P1.
(har ni läst IL1331 och sedan väljer Embeddedprogrammet så får ni där utrymme för en extra kurs!)

- *Läroboken blir då användbar en gång till, men denna gång med **alla** VHDL-avsnitten inkluderade!*

William Sandqvist william@kth.se

Inbyggda system – överallt!



William Sandqvist william@kth.se

Kursutvärdering

- Det är viktigt att vi får feedback!
- Ni kommer snart att få en e-mail med instruktionerna för kursutvärderingen som kommer att göras på webben
- Hjälp oss att förbättra kursen med konstruktiv kritik (gärna kommentarer)

Tack för uppmärksamheten!

Lycka till med tentan!

William Sandqvist william@kth.se