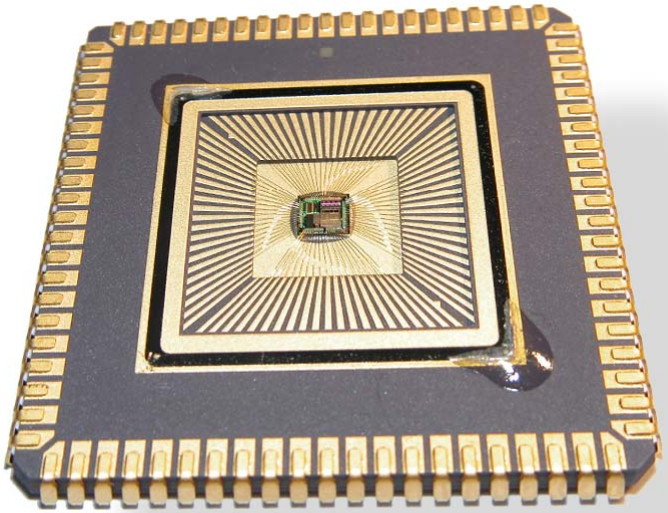
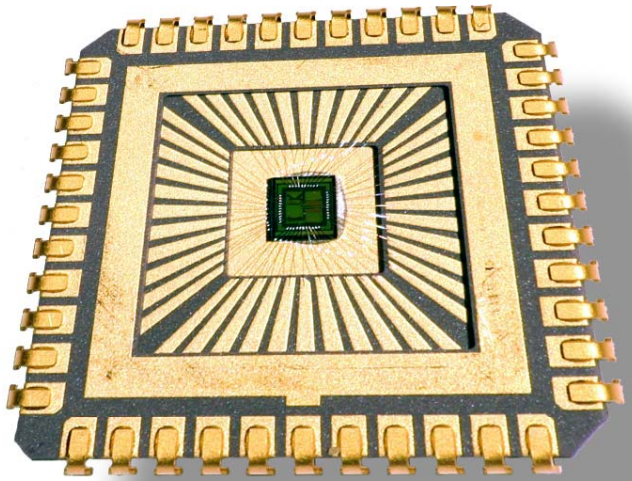
Digitala elektroniksystem





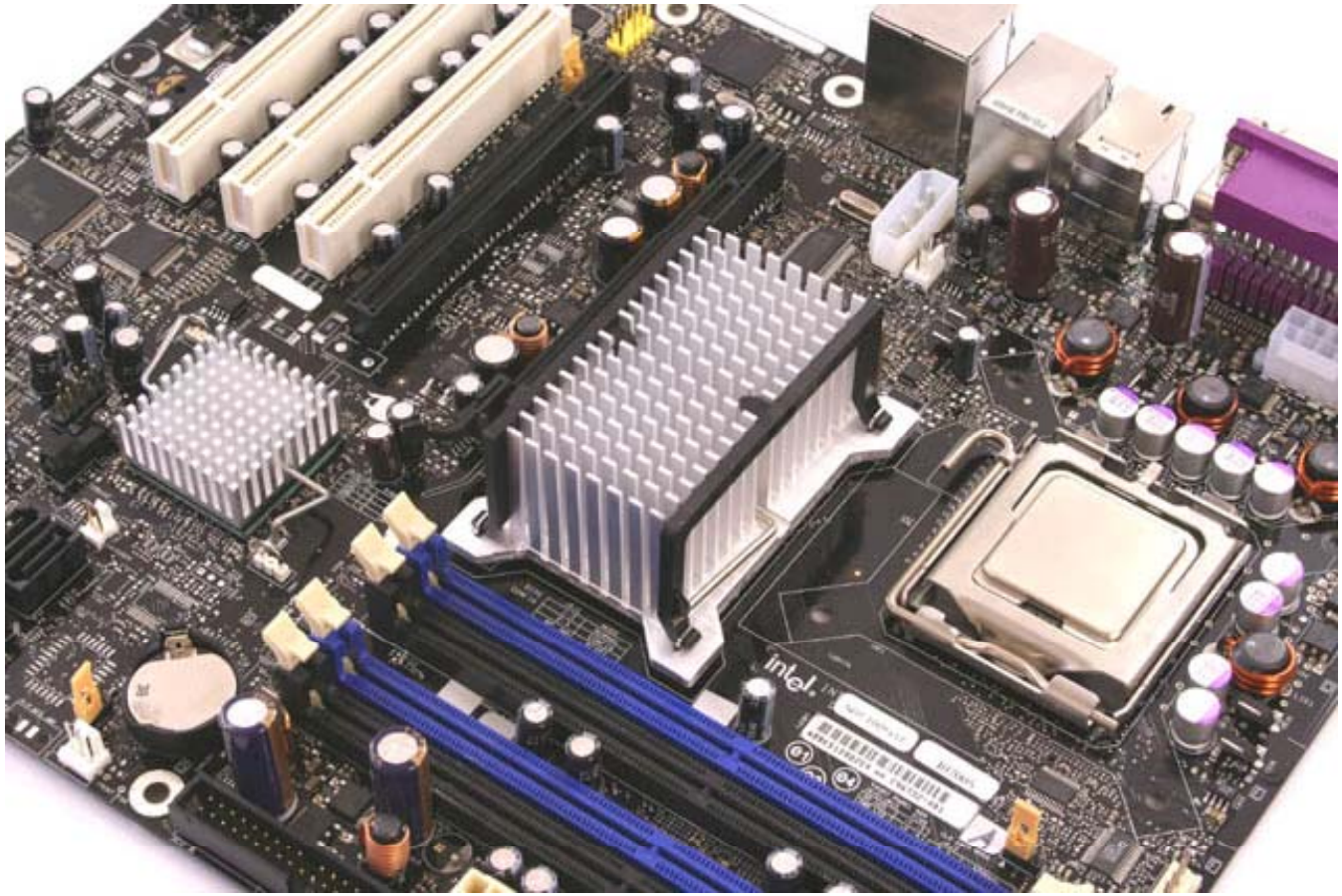
Professor Per Larsson-Edefors

perla@chalmers.se

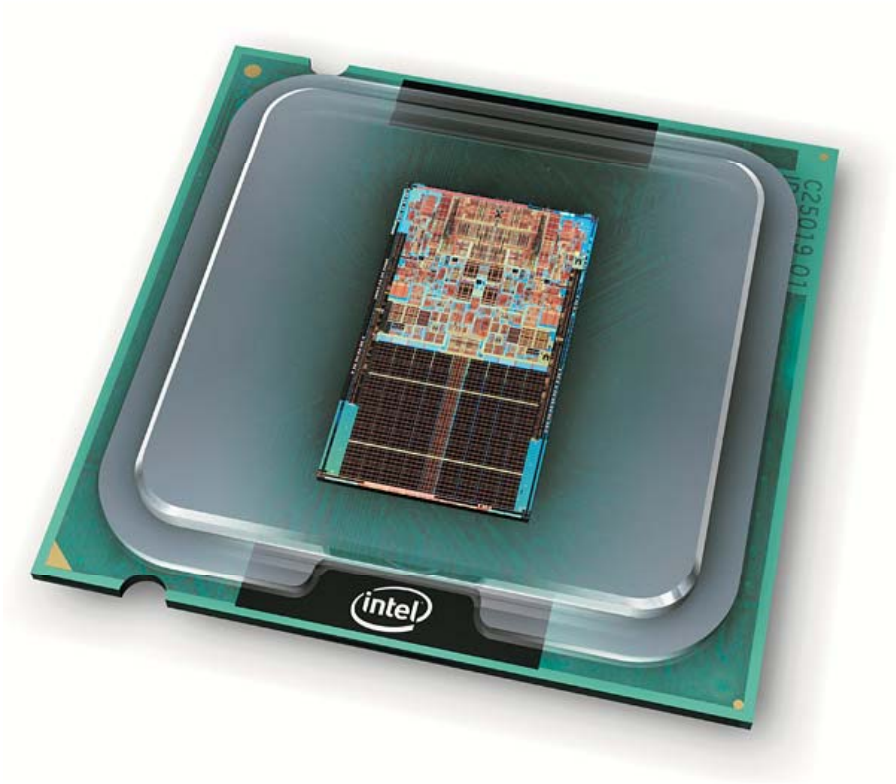
Konstruktionsalternativ

- Kretskort med diskreta standardkomponenter.
  - Utvecklingskostnad och prestanda låga.
- Rekonfigurerbara kretsar.
  - Utvecklingskostnad och prestanda medelhöga.
- Applikationsspecifika kretsar.
  - Utvecklingskostnad och prestanda höga -> stora serier.
  - Hög integration (System på kisel), digitalt och analogt.

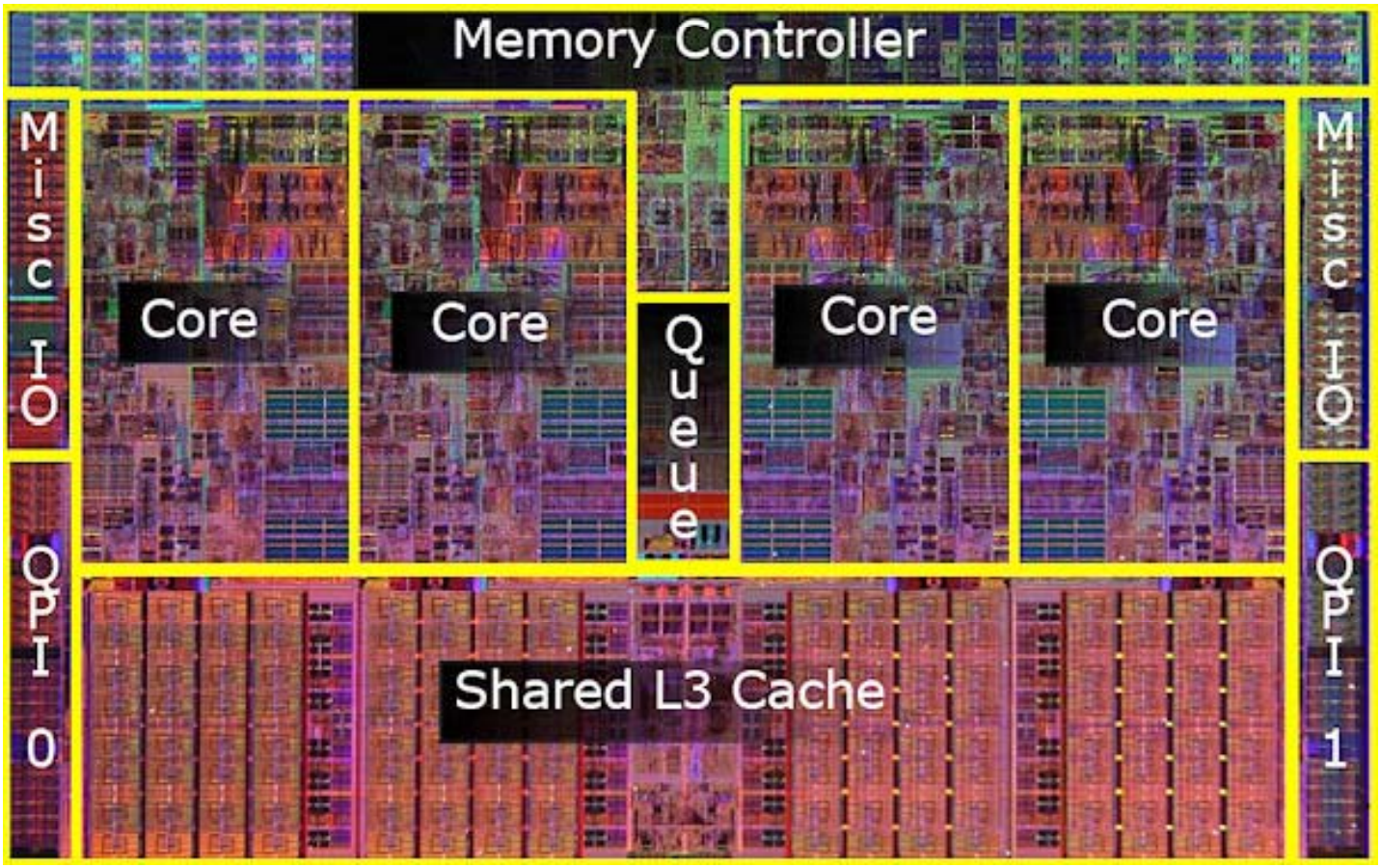
Moderkort med processor



Processorn – hjärnan i systemen

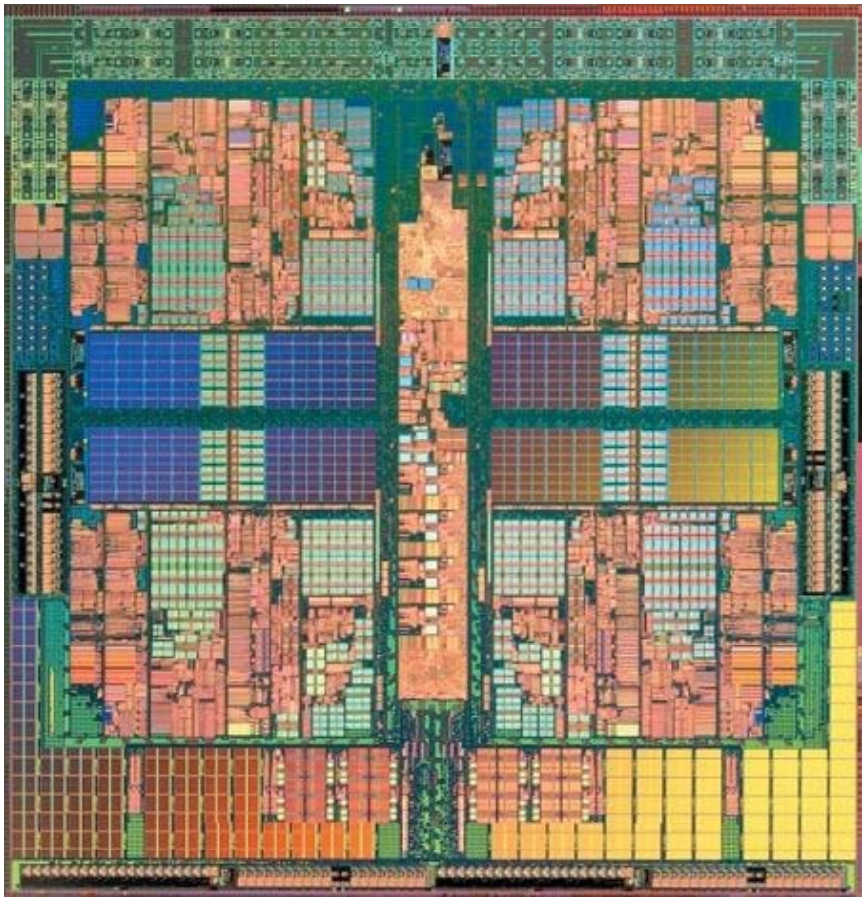


45-nm X4 i7 - 2008



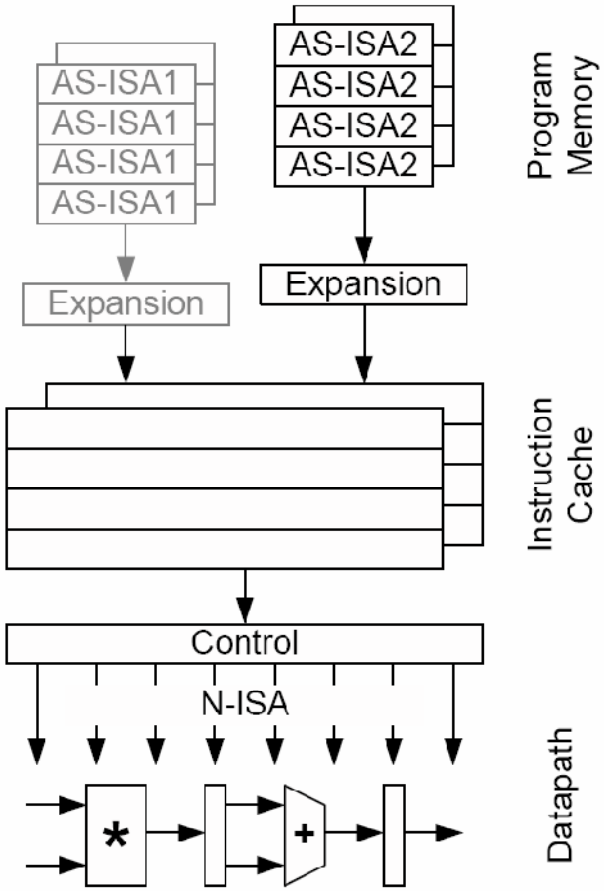
Source: Intel

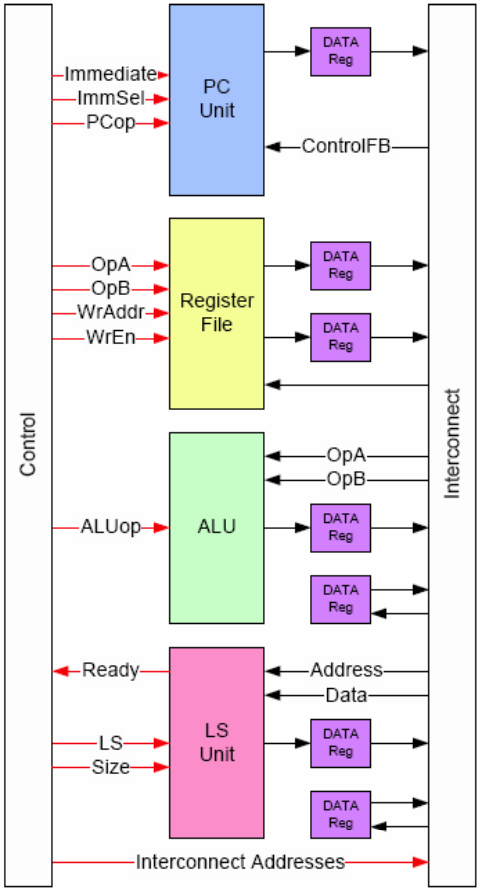
65-nm SOI X4 Phenom - 2008



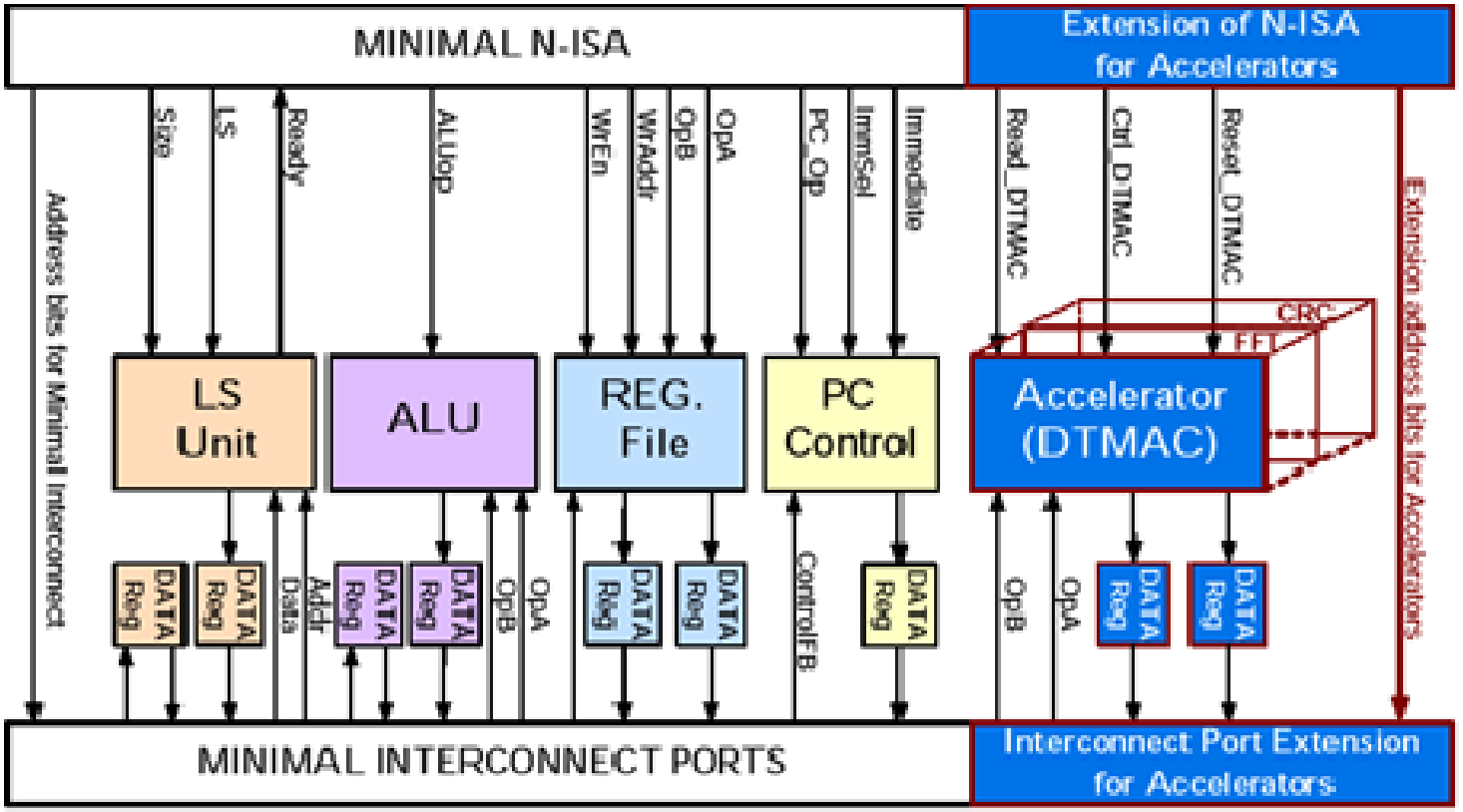
Source: AMD

FlexCore – en Chalmersprocessor

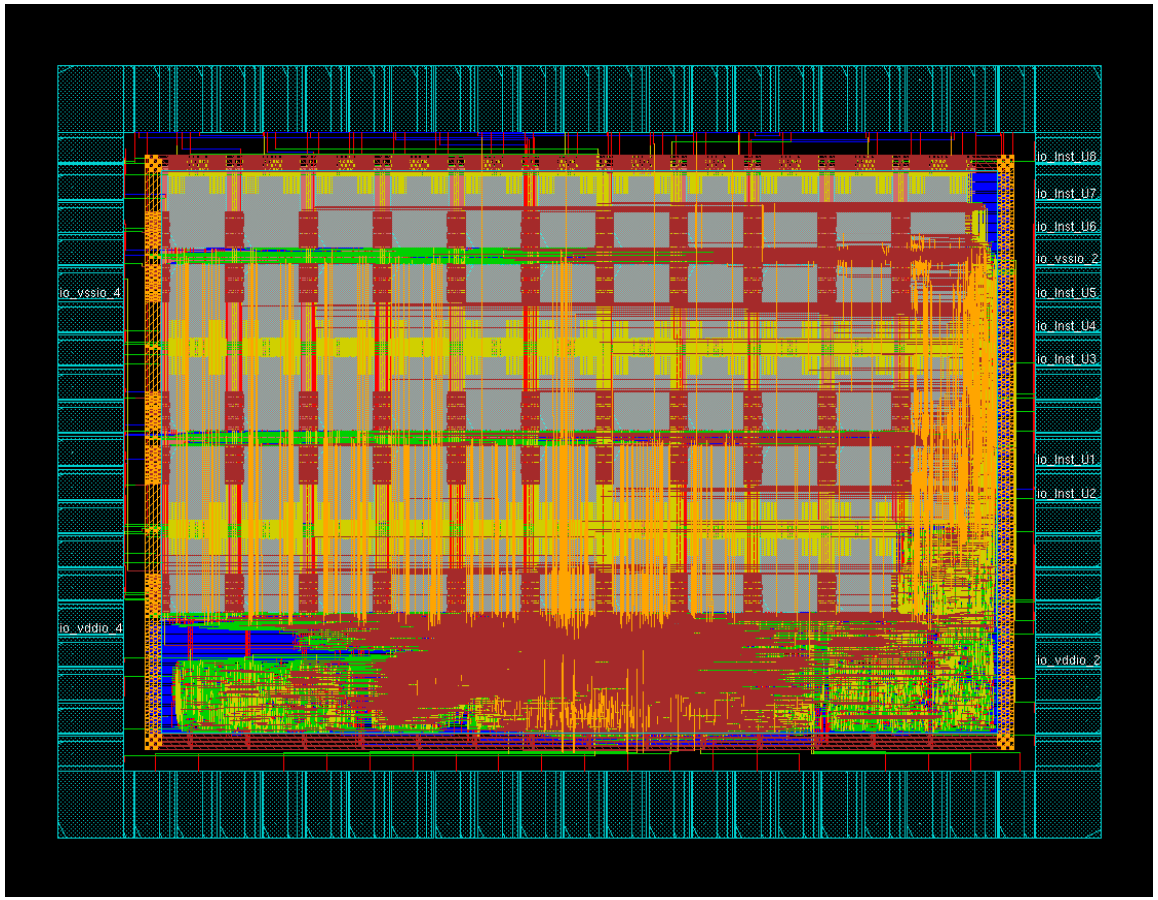




Utbyggbar processor



FlexCore-chip

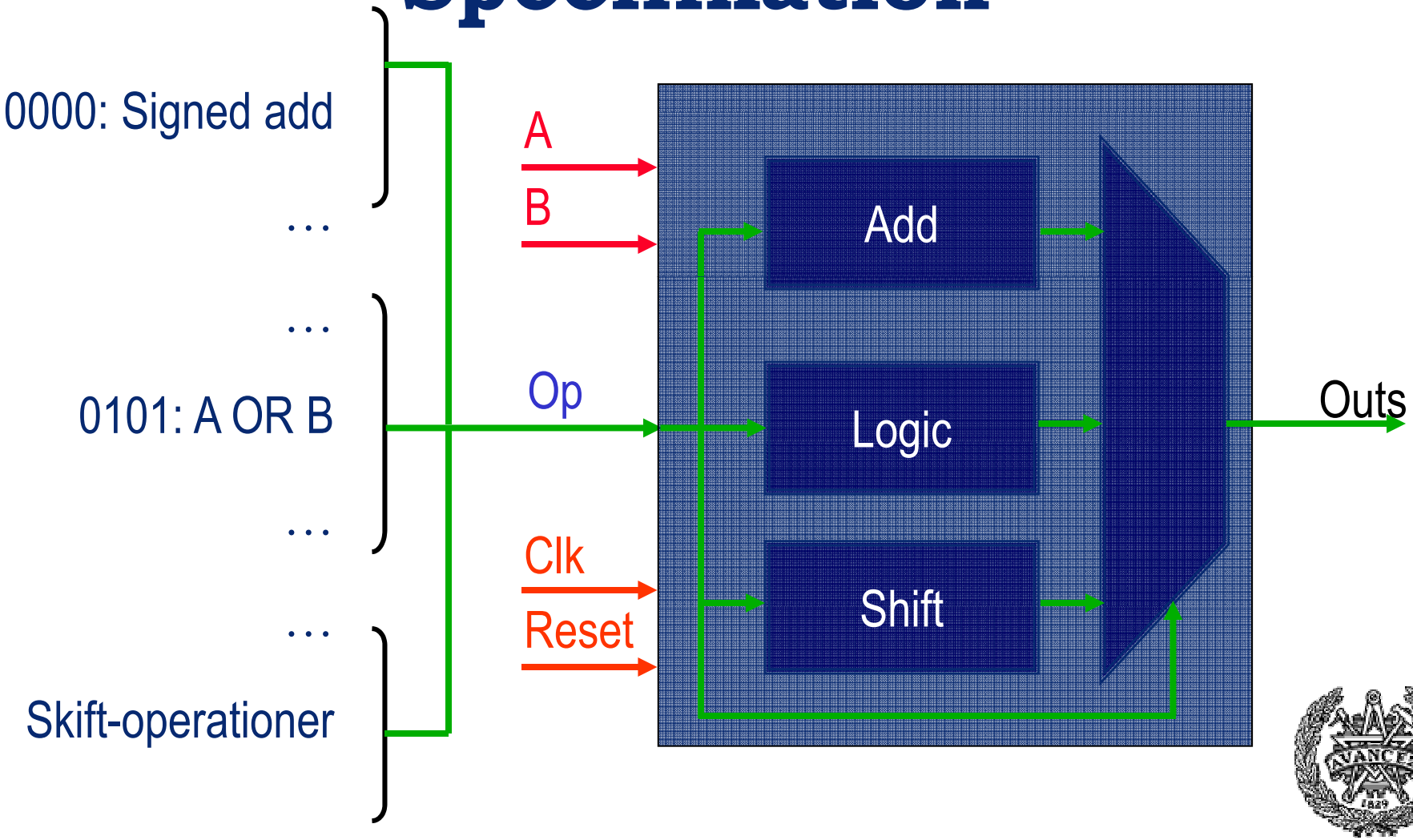


Konstruktion av digital hårdvara

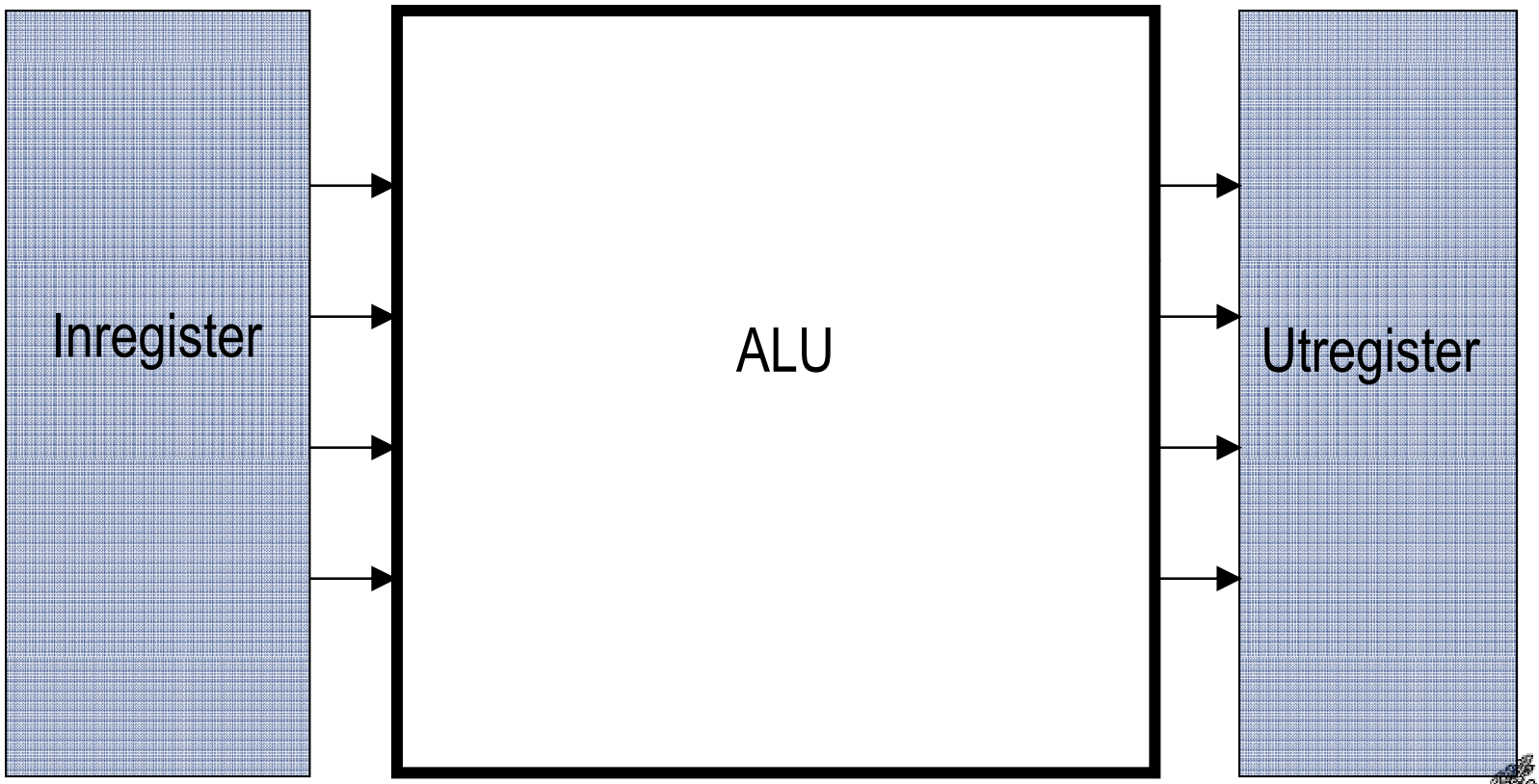
- Hårdvarubeskrivande språk:
  Ett mjukvaruliknande språk som beskriver hårdvaran funktionellt/logiskt.
  - VHDL, Verilog, etc.

- Låt oss bygga en ALU…

Specifikation



In- och utregister



VHDL – öppna ALUns portar

```
entity ALU is
  port(Clk    : in  std_logic;
    Reset     : in  std_logic;
    A         : in  std_logic_vector(31 downto 0);
    B         : in  std_logic_vector(31 downto 0);
    Op        : in  std_logic_vector(3 downto 0);
    Outs      : out std_logic_vector(31 downto 0));
end ALU;
```

VHDL – inkludera adderare

```
architecture RTL of ALU is
    component Adder
        port ( OpA      : in  std_logic_vector(31 downto 0);
               OpB      : in  std_logic_vector(31 downto 0);
               Cin      : in  std_logic;
               Cout     : out std_logic;
               Result   : out std_logic_vector(31 downto 0));
    end component;
```

VHDL – definiera register

```
process
begin
    if Reset='1' then
        ALU_inA <= (others => '0');
        ALU_inB <= (others => '0');
    elsif rising_edge(Clk) then
        ALU_inA <= A;
        ALU_inB <= B;
    end if;
end process;
```

VHDL – definiera logik

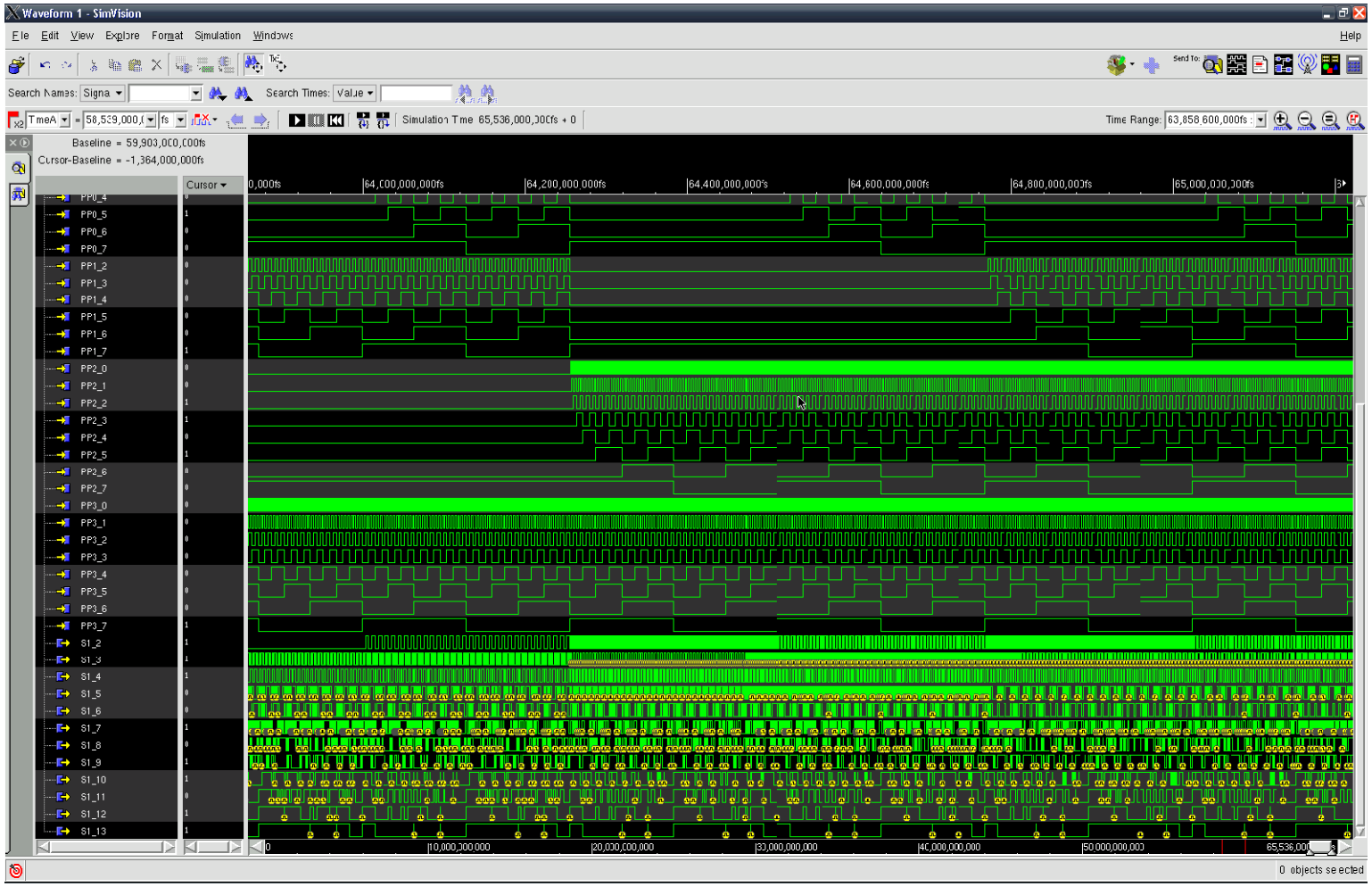
```
process
begin
    case ALU_op is
        when "0000" =>
                ALU_Out <= Add_Out;
        ...
        when "0101" =>
                ALU_Out <= ALU_InA or ALU_InB;
        …
```

Verifiering

- Simulering av hårdvarubeskrivningen avslöjar om vi konstruerat det vi avsett,

  om specifikationen och VHDL-implementationen överensstämmer.

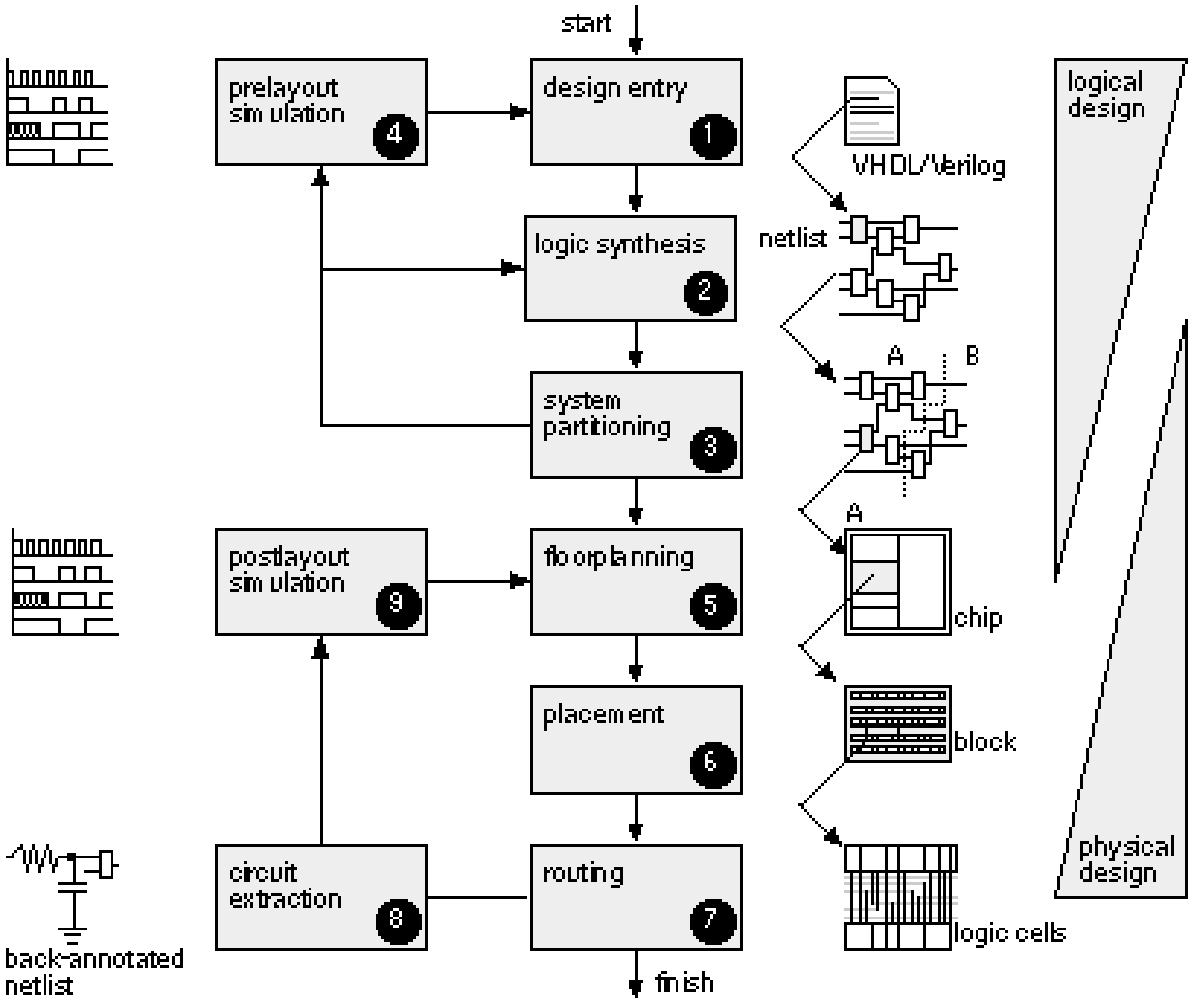
Simulering


Waveform 1 - SimVision
File Edit View Explore Format Simulation Windows Help
Search Names: Signa
Search Times: Value
Simulation Time 65,536,000,300fs + 0
Time Range: 63,858,600,000fs
Baseline = 59,903,000,000fs
Cursor-Baseline = -1,364,000,000fs
Cursor
64,000,000,000fs
64,200,000,000fs
64,400,000,000fs
64,600,000,000fs
64,800,000,000fs
65,000,000,300fs
PP0_5
PP0_6
PP0_7
PP1_2
PP1_3
PP1_4
PP1_5
PP1_6
PP1_7
PP2_0
PP2_1
PP2_2
PP2_3
PP2_4
PP2_5
PP2_6
PP2_7
PP3_0
PP3_1
PP3_2
PP3_3
PP3_4
PP3_5
PP3_6
PP3_7
SI_2
SI_3
SI_4
SI_5
SI_6
SI_7
SI_8
SI_9
SI_10
SI_11
SI_12
SI_13
10,000,300,000
20,000,000,000
30,000,000,000
40,000,000,000
50,000,000,000
0 objects selected

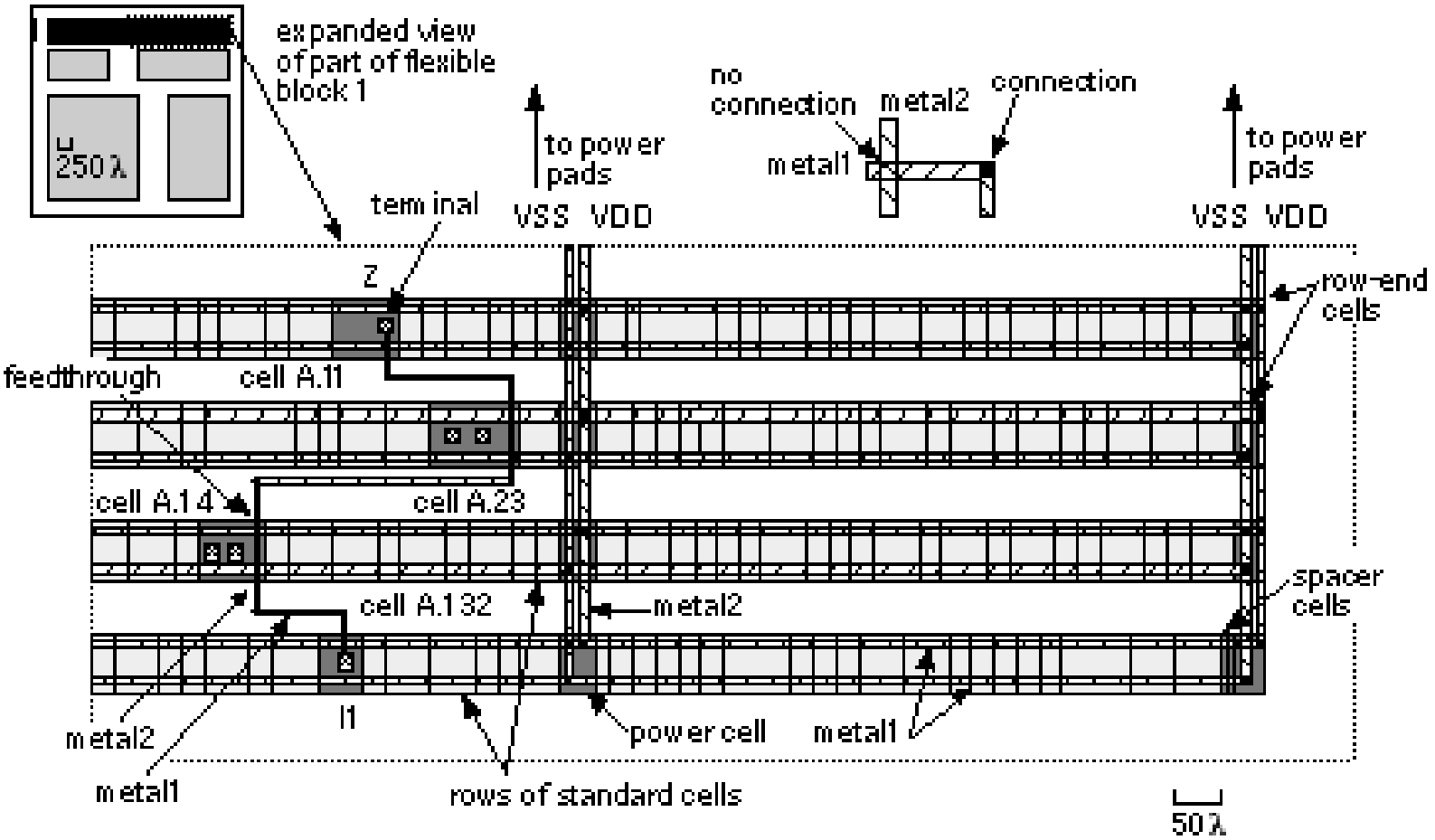
Fler datorhjälpmedel …

- När vår beskrivning är korrekt,
  omvandlar CAD-verktyg denna beskrivning
  till fysiska kretsar,
  under villkor på fördröjning mellan register.

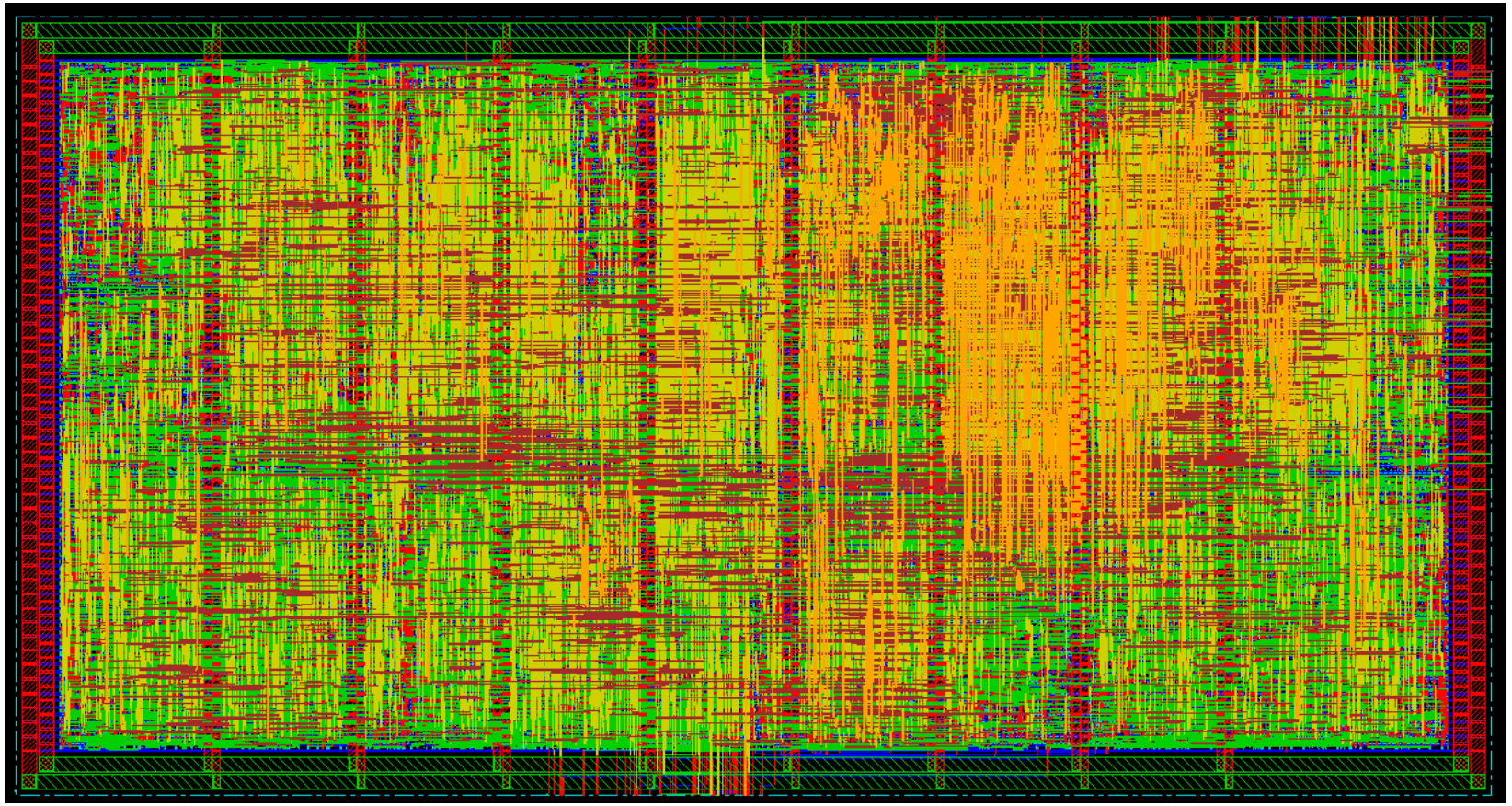
”ASIC Design Flow”



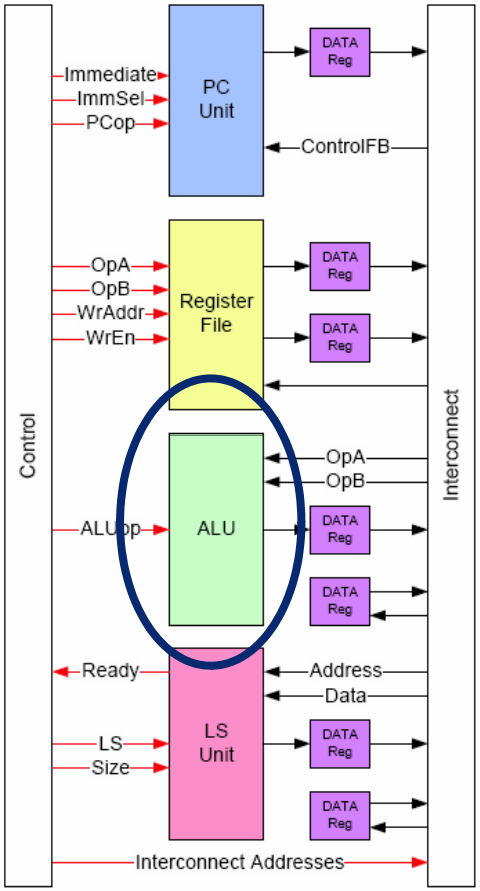
Standardcells-konstruktion



Slutresultatet



Adderare i processorer

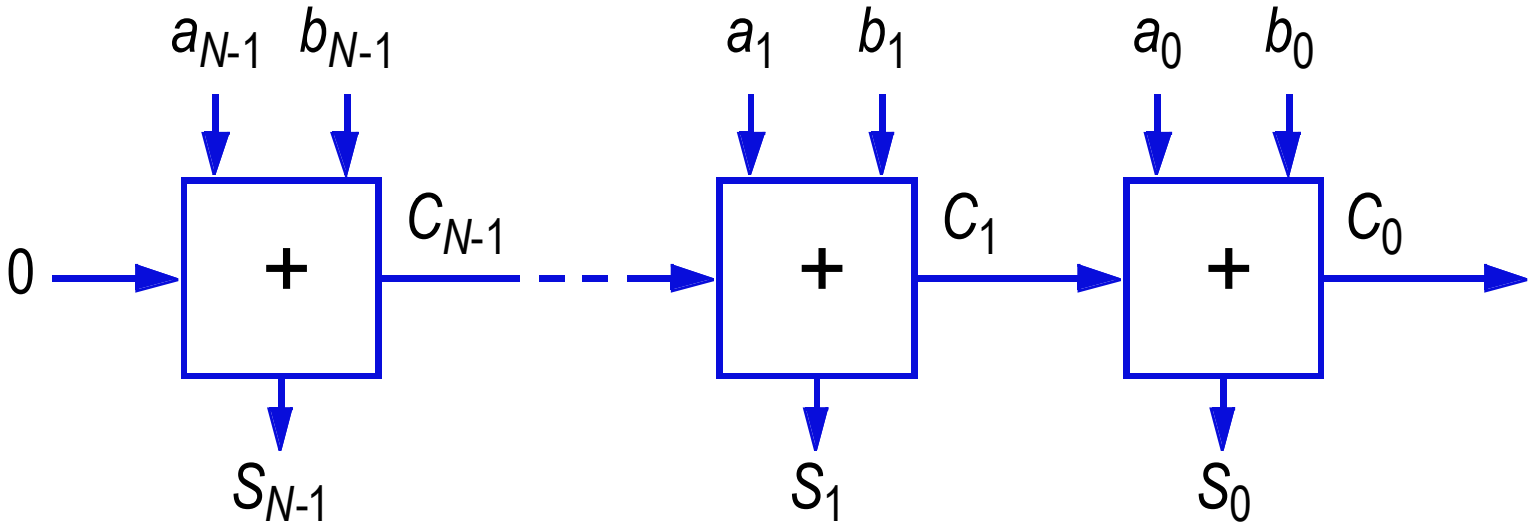


Addition

```
  11111111111111111111111111111
+ 00000000000000000000000000001
-------------------------------
 100000000000000000000000000000
```

- Vi måste alltid konstruera för de värsta fallen som kan hända!
- Längst fördröjning = en carrybit i LSB ripplar ända till MSB.

Ripple-Carry Adder

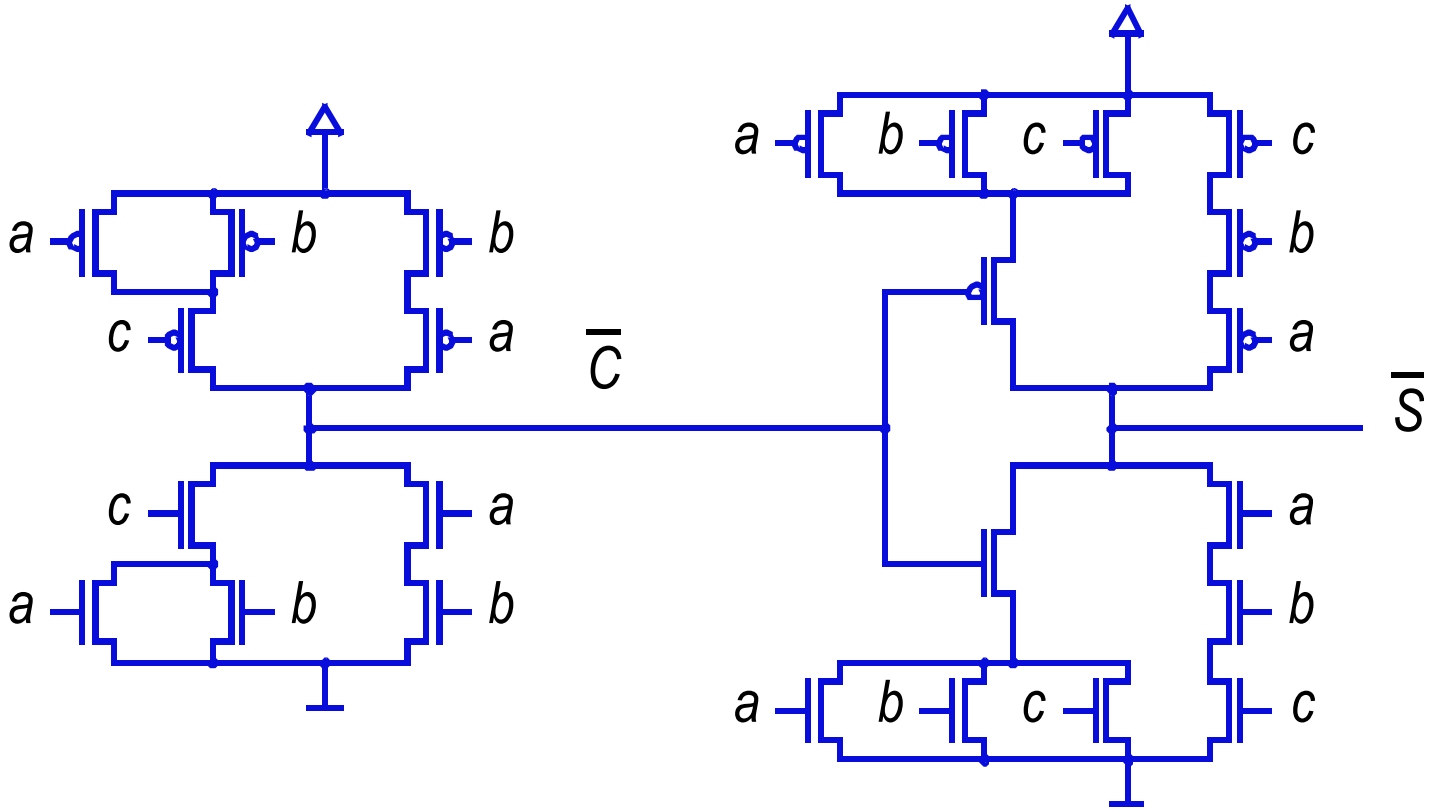


- Ripple-carry-adderare (RCA):
  längsta fördröjningen beror linjärt av ordlängden.

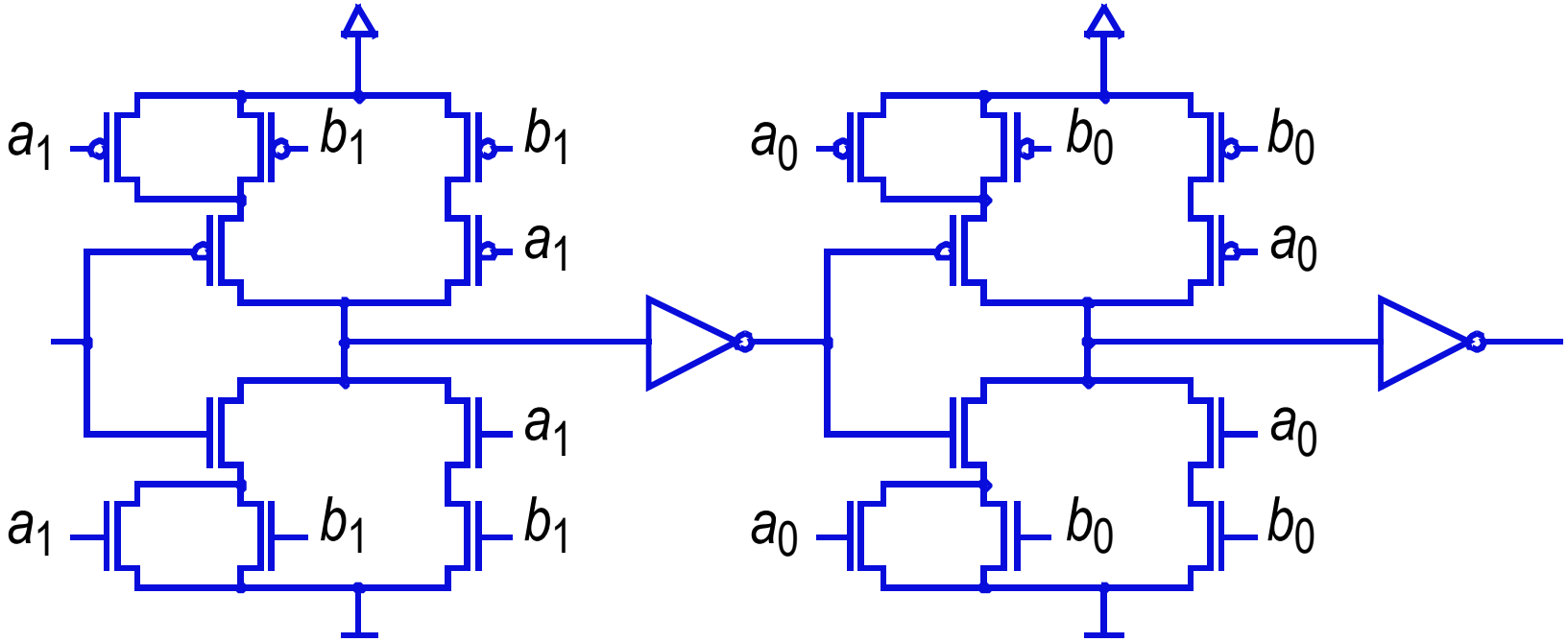
Heladderaren

A	B	C_{in}	C_{ut}	S_{ut}
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Transistorkrets för heladderare

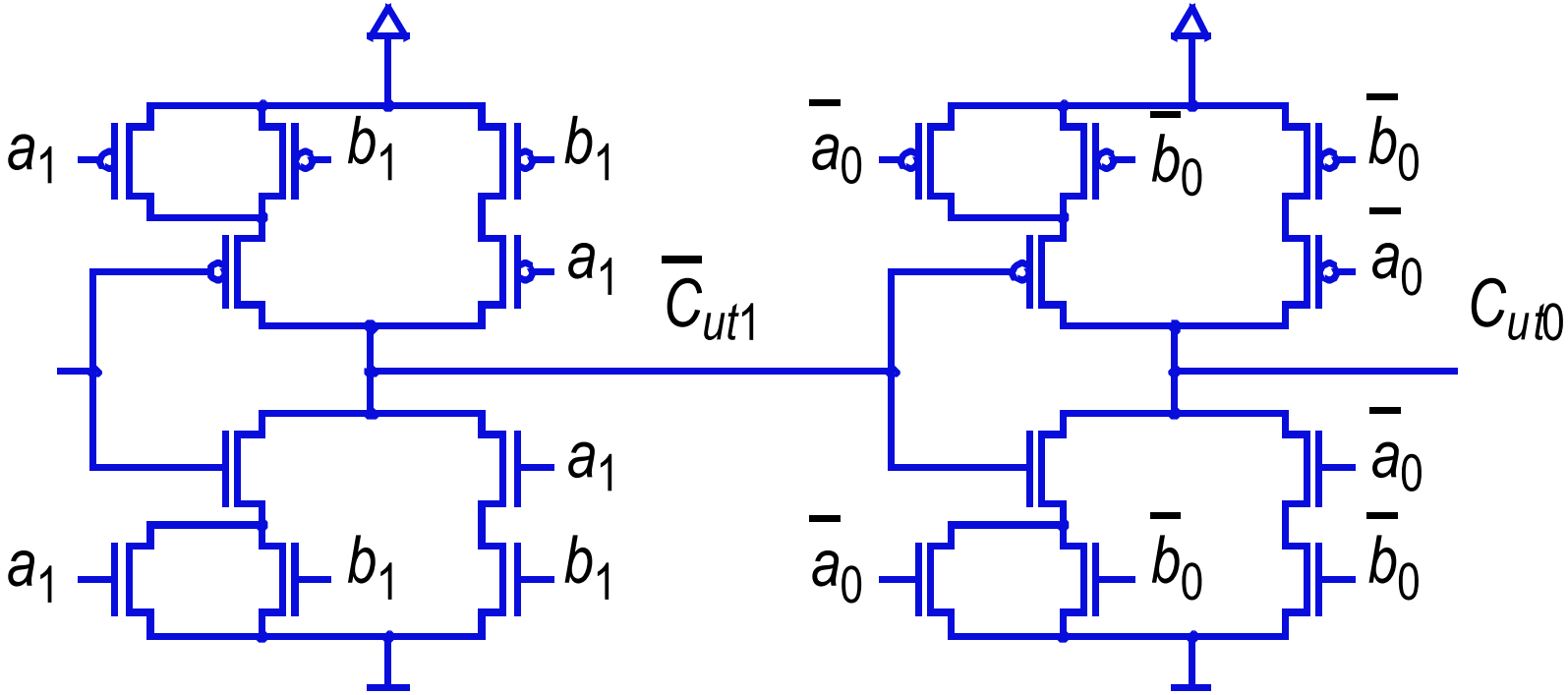


Titta speciellt på carryvägen



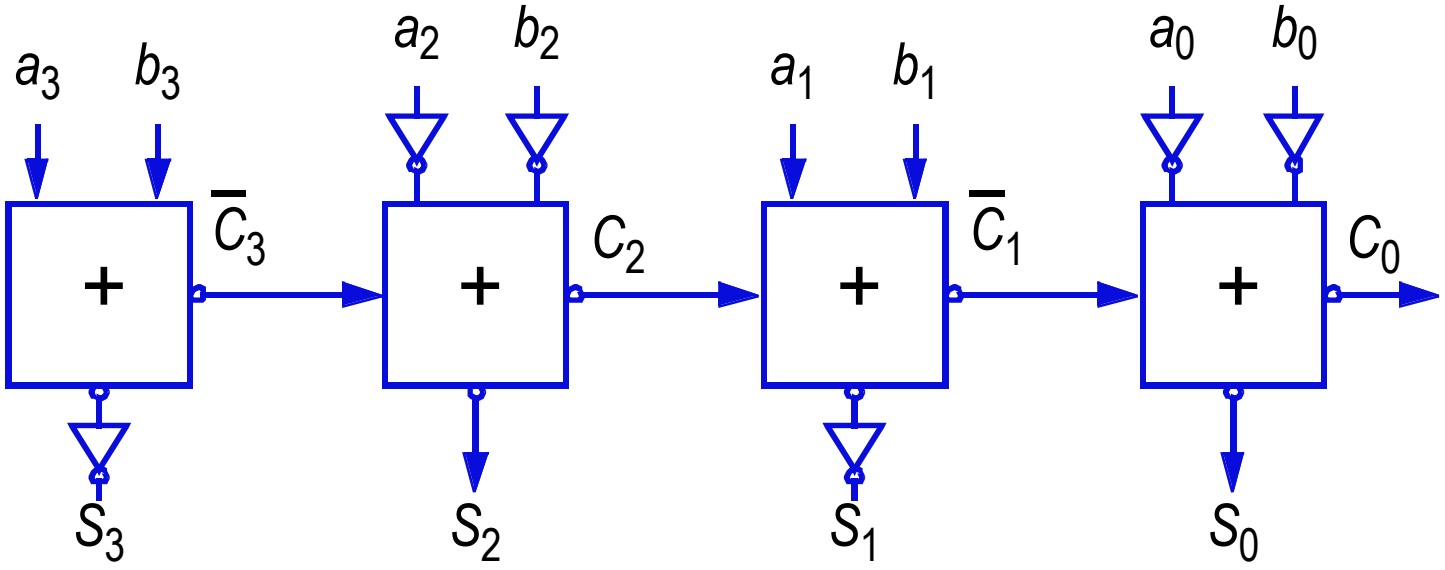
- Det finns nu onödiga inverterare i kritiska vägen.

En snabbare carryväg

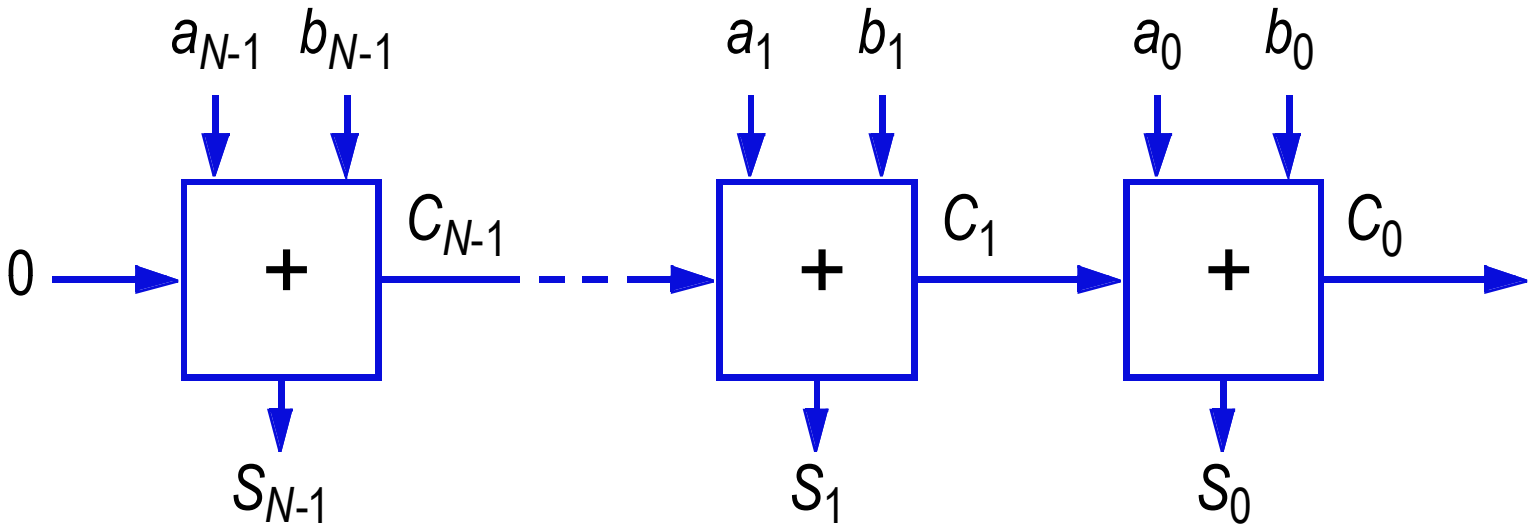


- Utnyttjar att $C'(a, b, c) = C(a', b', c')$

Bättre, men fortfarande $\propto N$

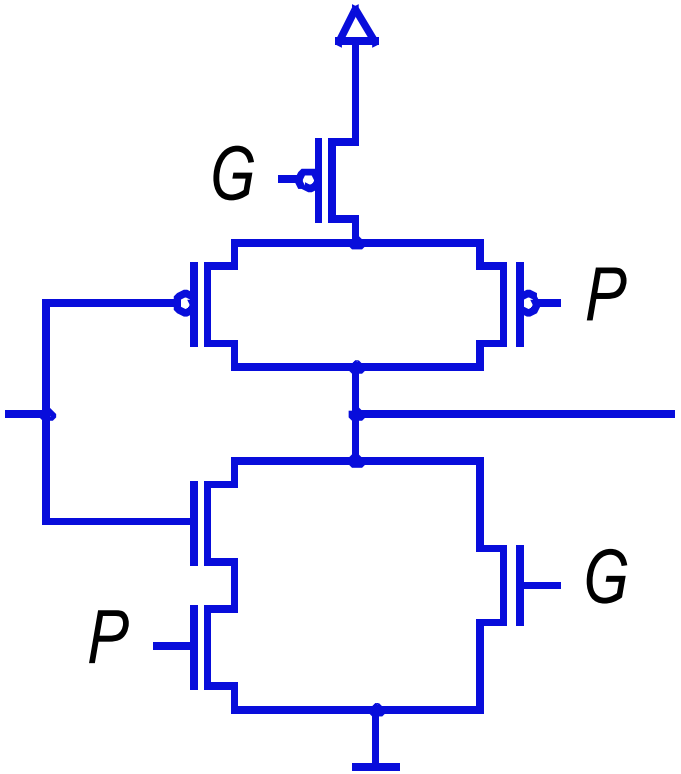


Studera ingångarna!



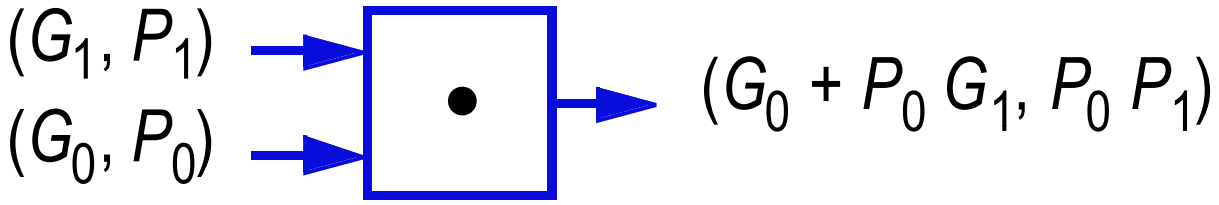
- *a* och *b* ansluter till heladderarna ”utifrån”.
- *c* kommer från annan heladderare.
- Gör förbearbetning, utanför kritiska vägen: Generate (*a b*) och Propagate (*a* + *b*)

Använda P och G i en krets



- Färre transistorer ger lägre grindfördröjning.
- $G = a\,b$
- $P = a + b$

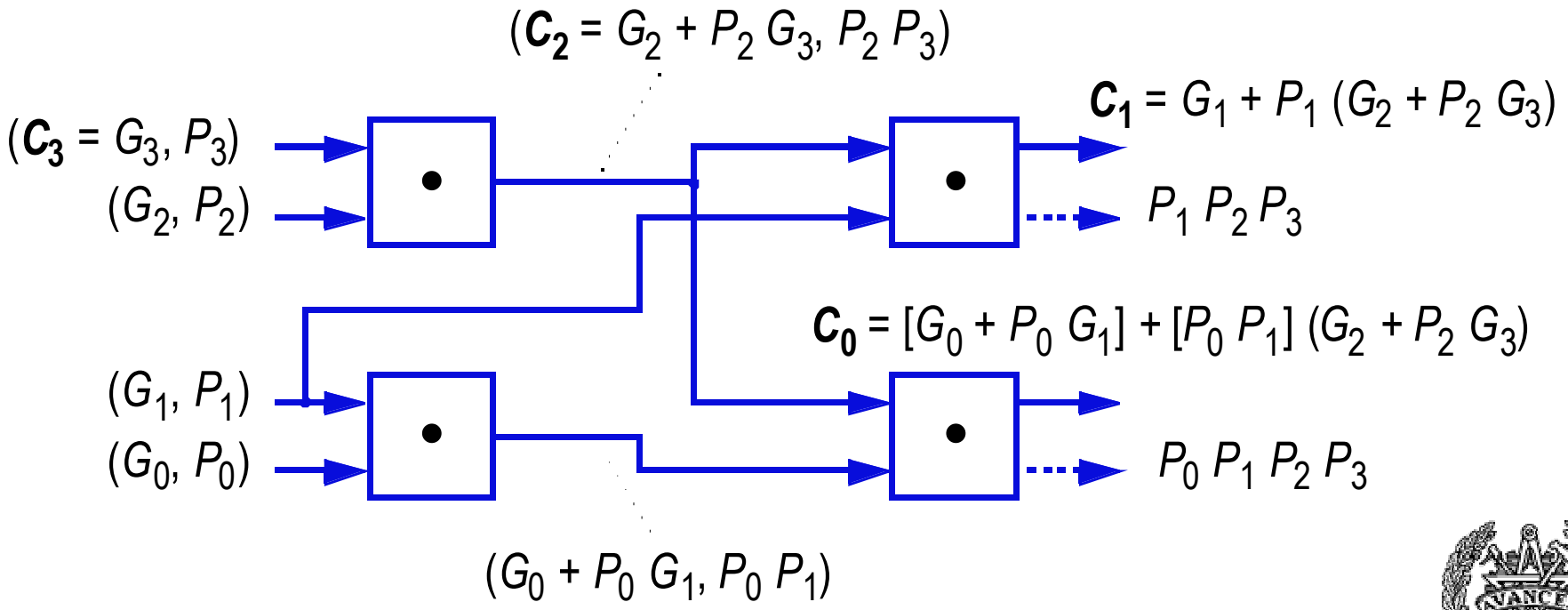
En ny operator, istället för +



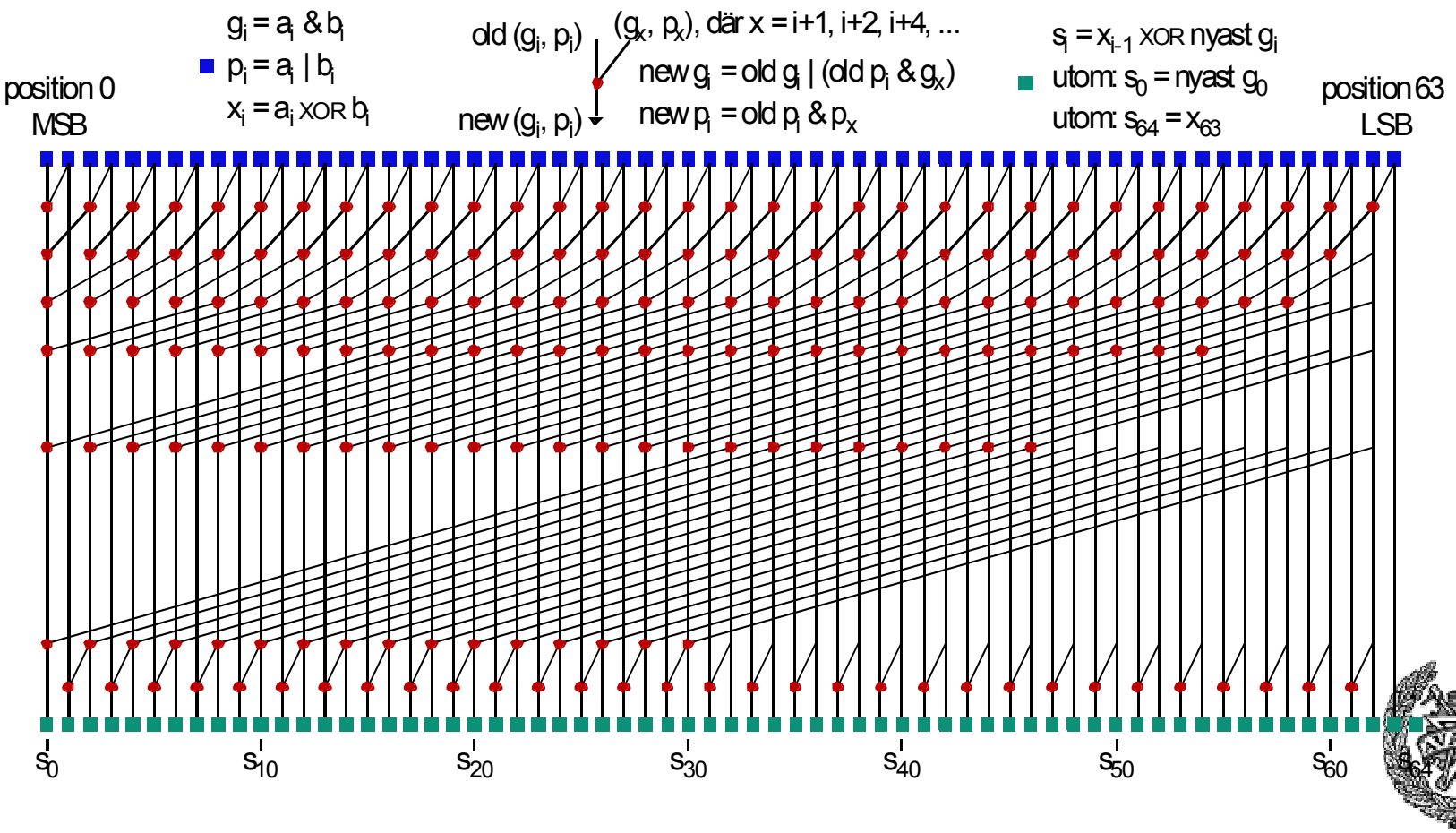
- Den s.k. dot-operatorn ger oss möjlighet att bygga associativa funktioner.

Exempel på addition

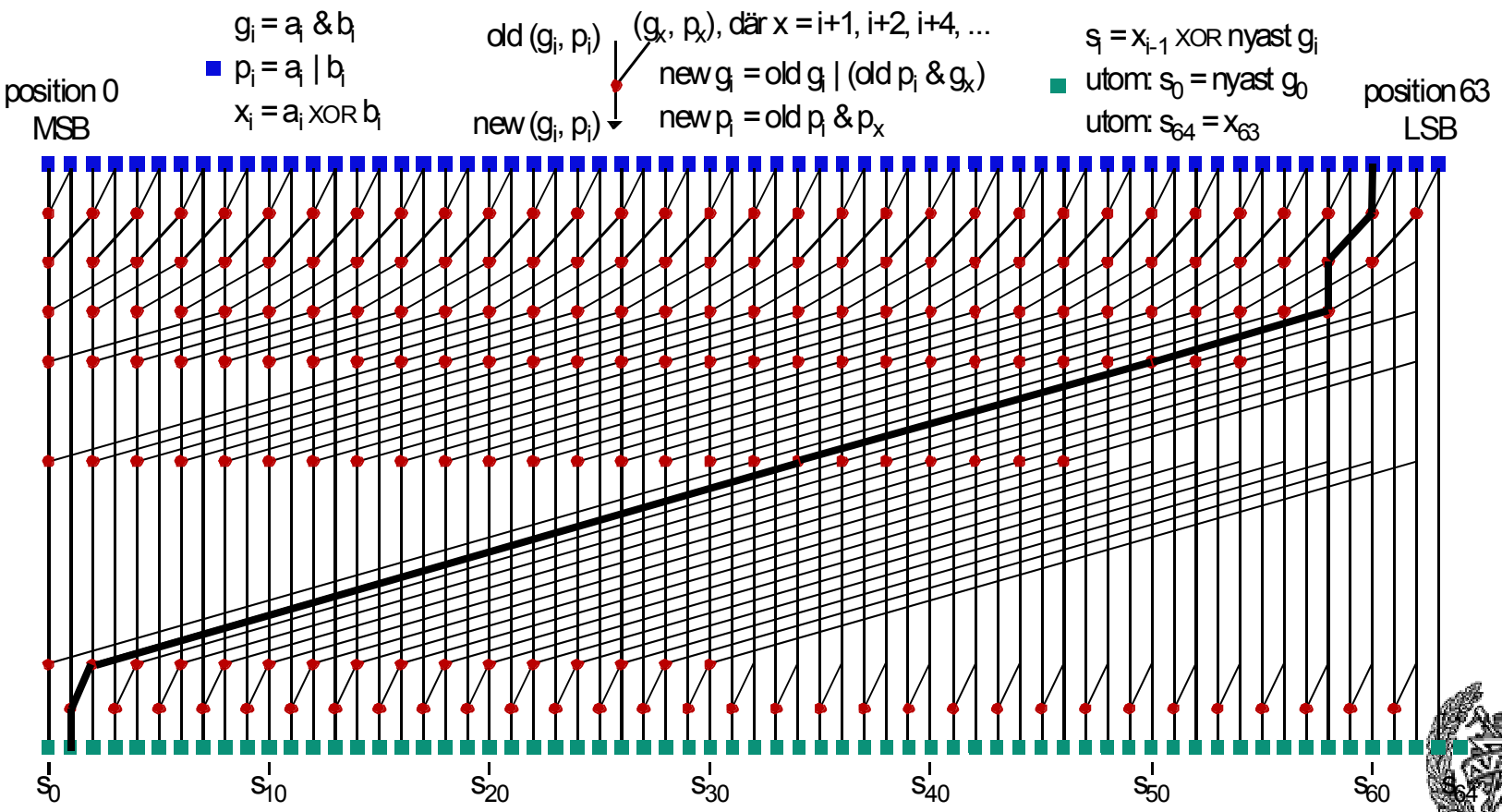
En 4-bitars addition:



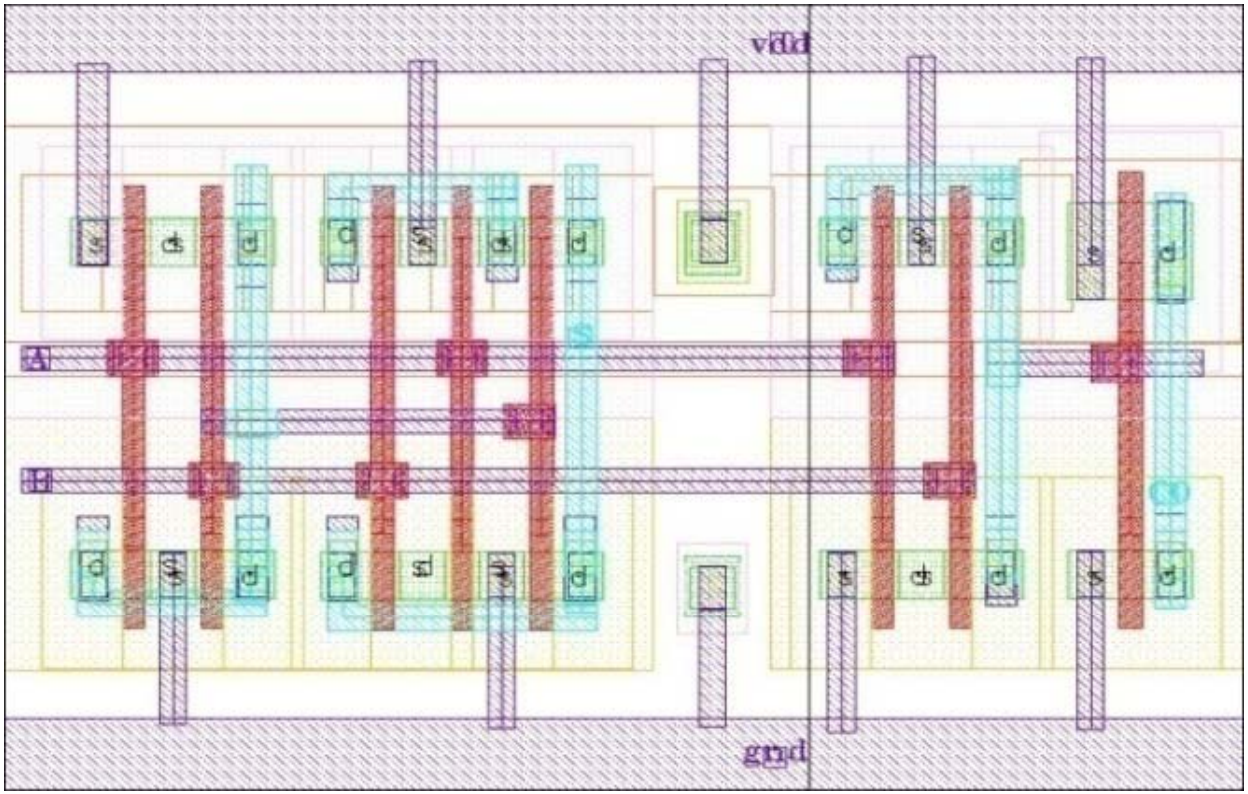
64-bitars processoradderare



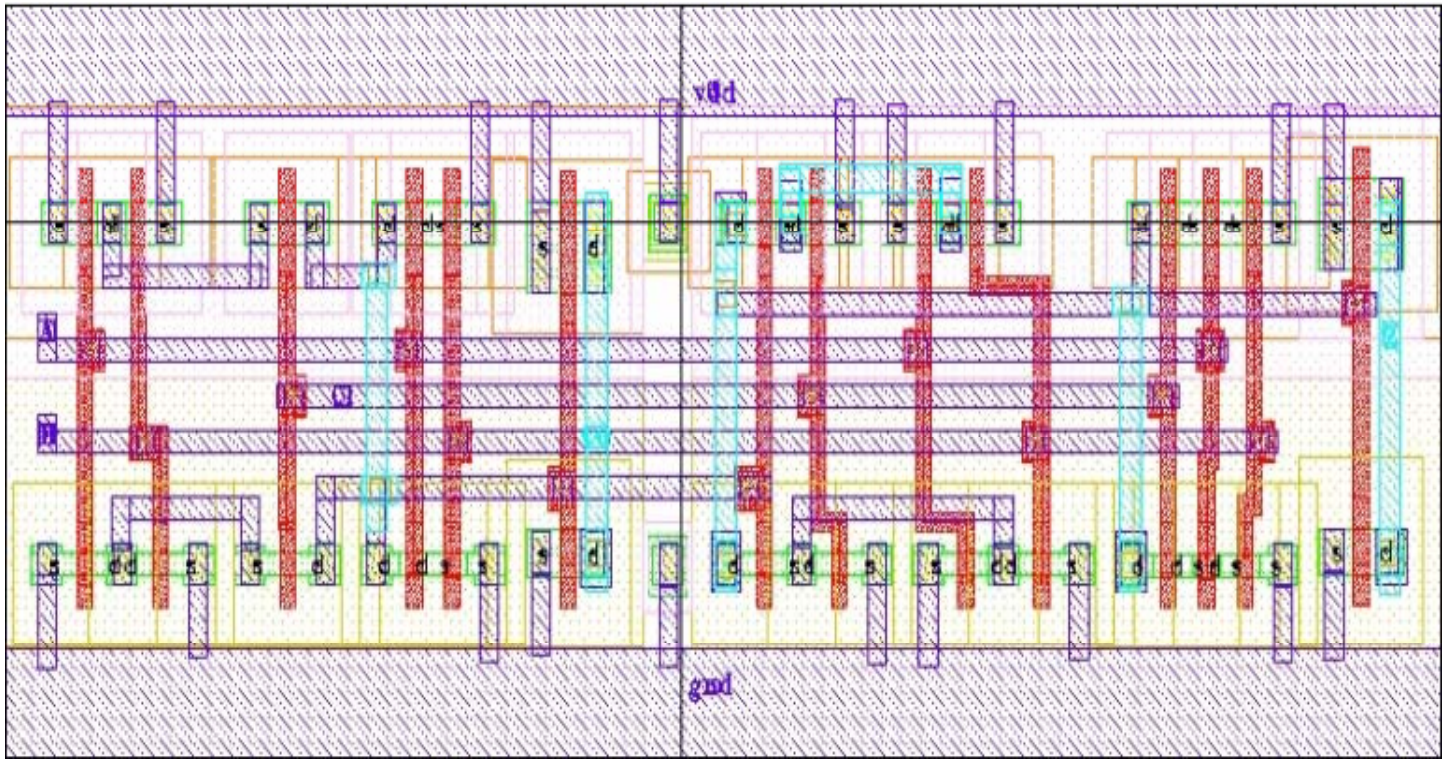
Logikdjup = 8 (jfr. 64 för RCA)



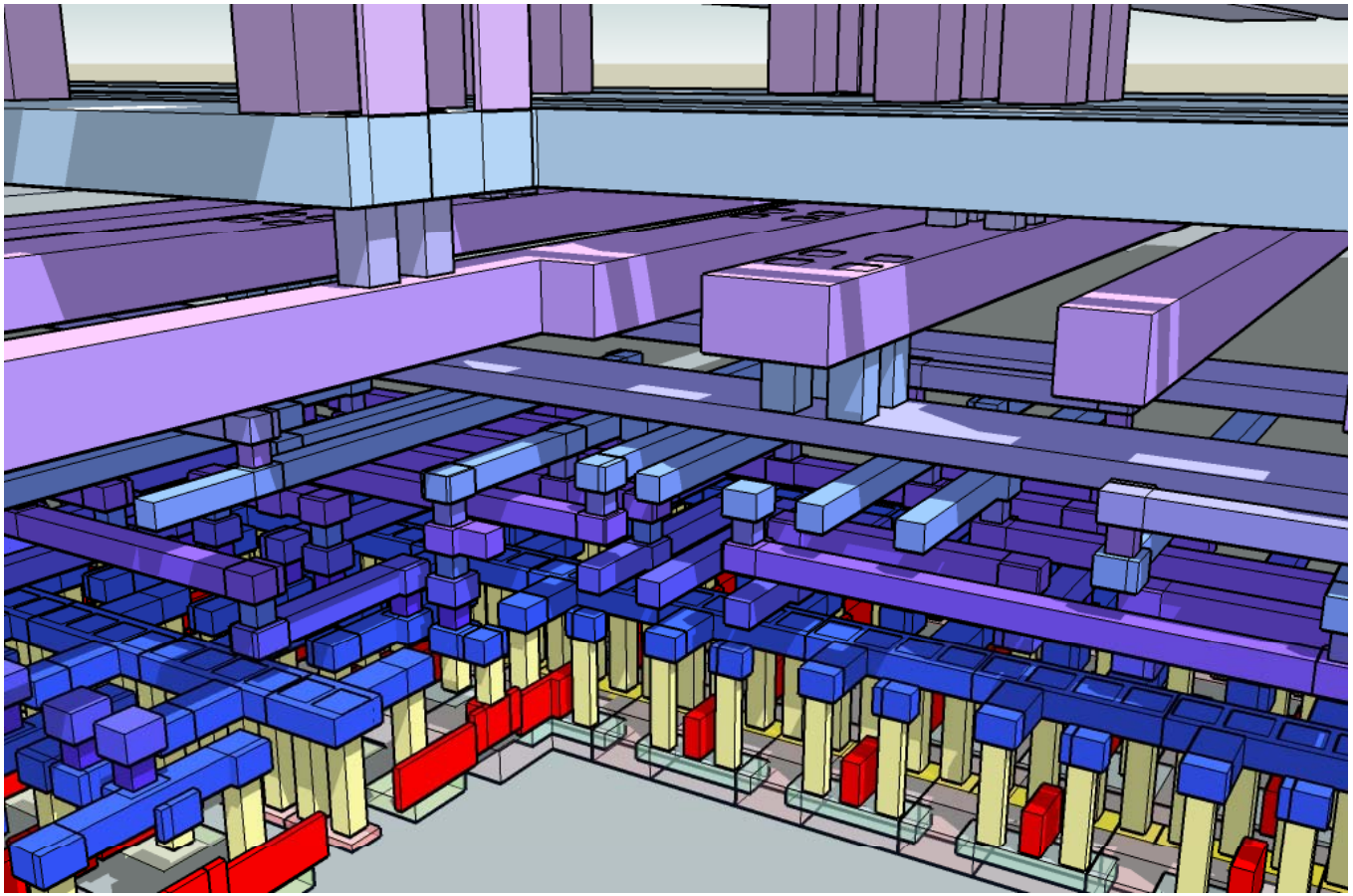
Layout för halvadderare



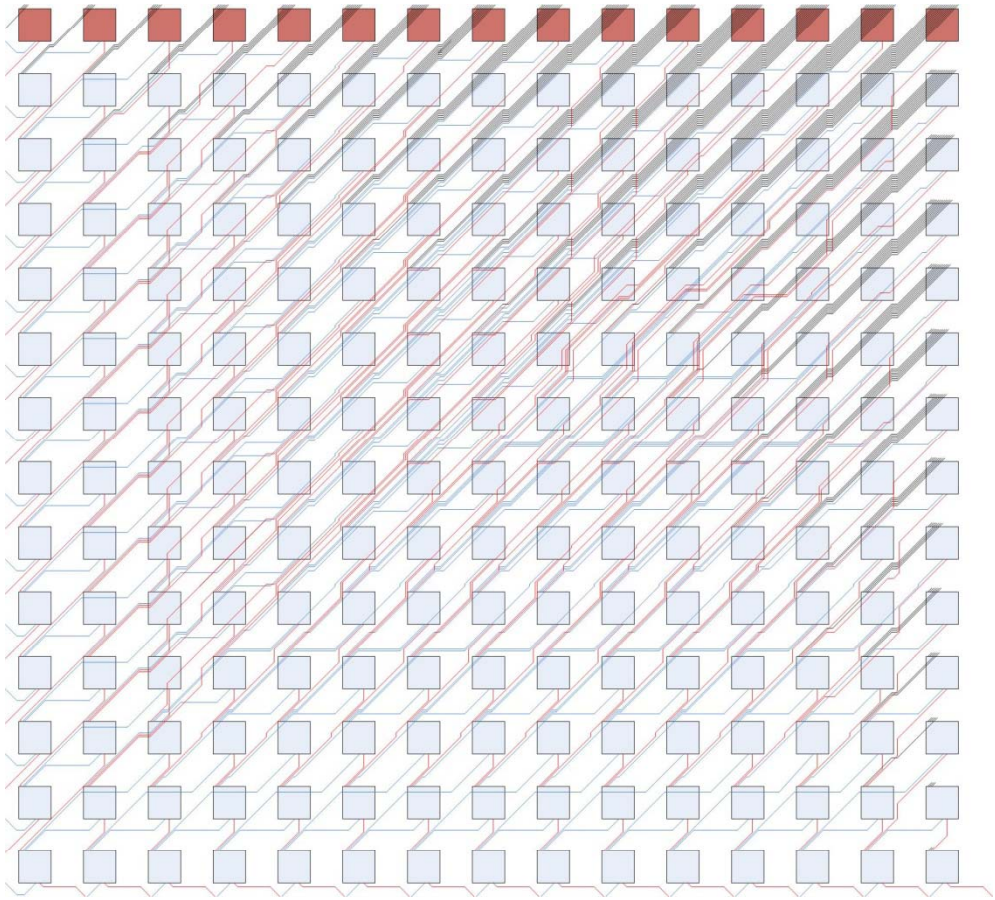
Layout för heladderare



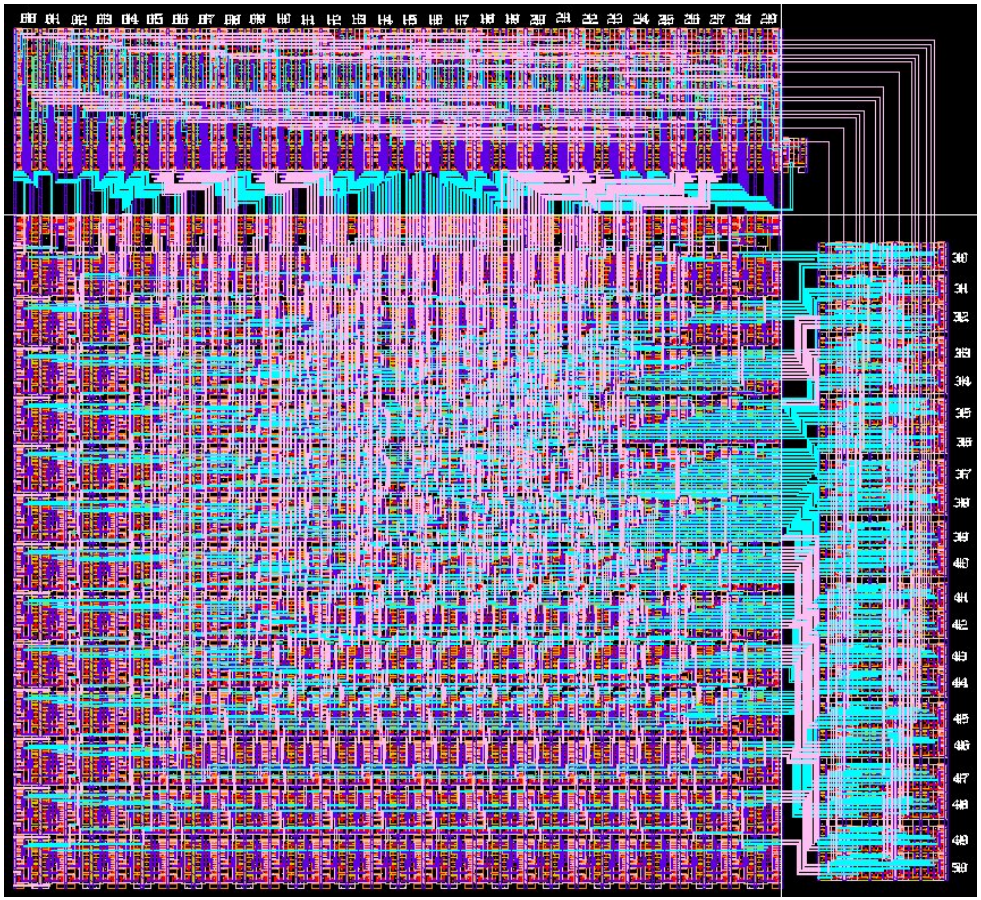
Den integrerade kretsen



Skiss på 16-bitars multiplikator



Layout av 16-bitars multiplikator



Avslutning

- Många konstruktionsalternativ!
- Olika teknologiplattformar.
  - ASIC, FPGA, standardkomponenter.
- Olika konstruktionsmetoder.
  - Manuell design blandat med CAD-stödd design.
- Utmaningen:
  - Komplexa system krävs för hög funktionalitet och hög prestanda.
  - Hur konstruerar man dessa?