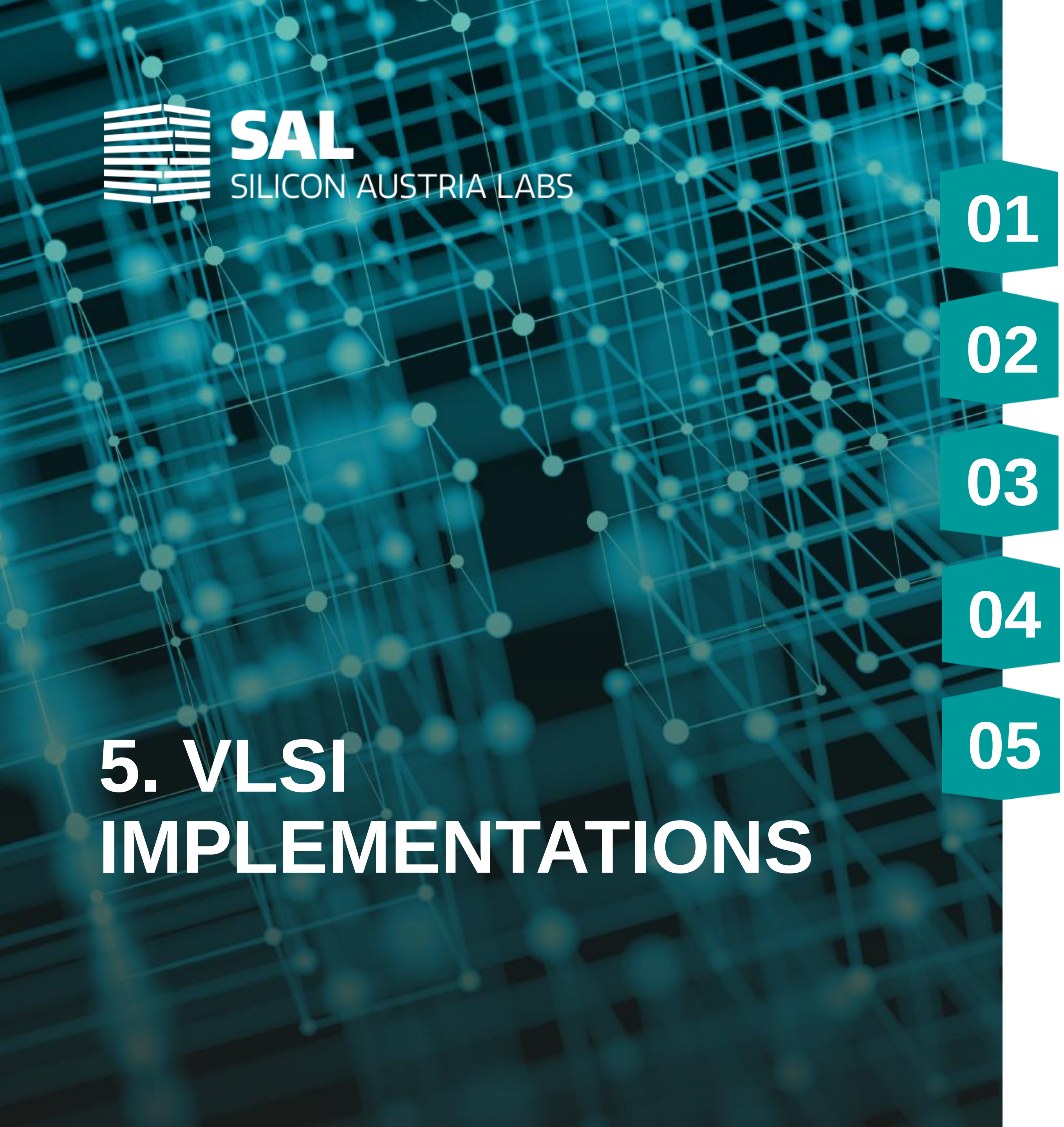
SAL
SILICON AUSTRIA LABS

VLSI FOR MACHINE LEARNING

PEDRO JULIAN, DIEGO GIGENA, NICOLÁS RODRÍGUEZ

CAE 2023, Córdoba



SAL
SILICON AUSTRIA LABS

5. VLSI IMPLEMENTATIONS

01 Basic Elements

02 Area and Power

03 Quantization

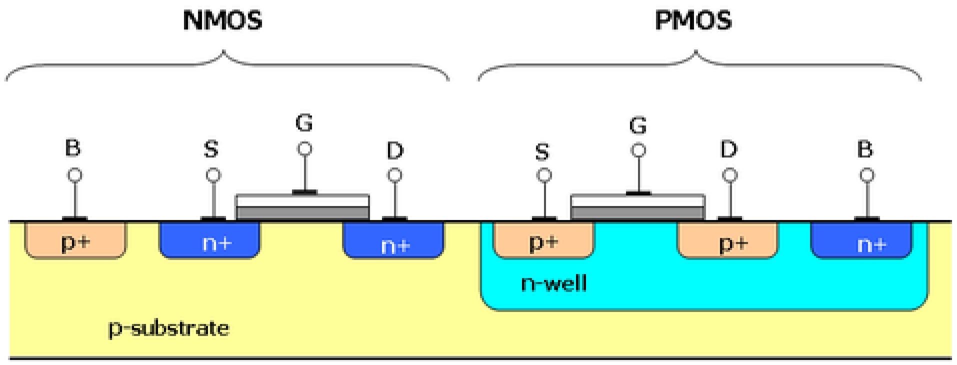
04 Advanced Elements

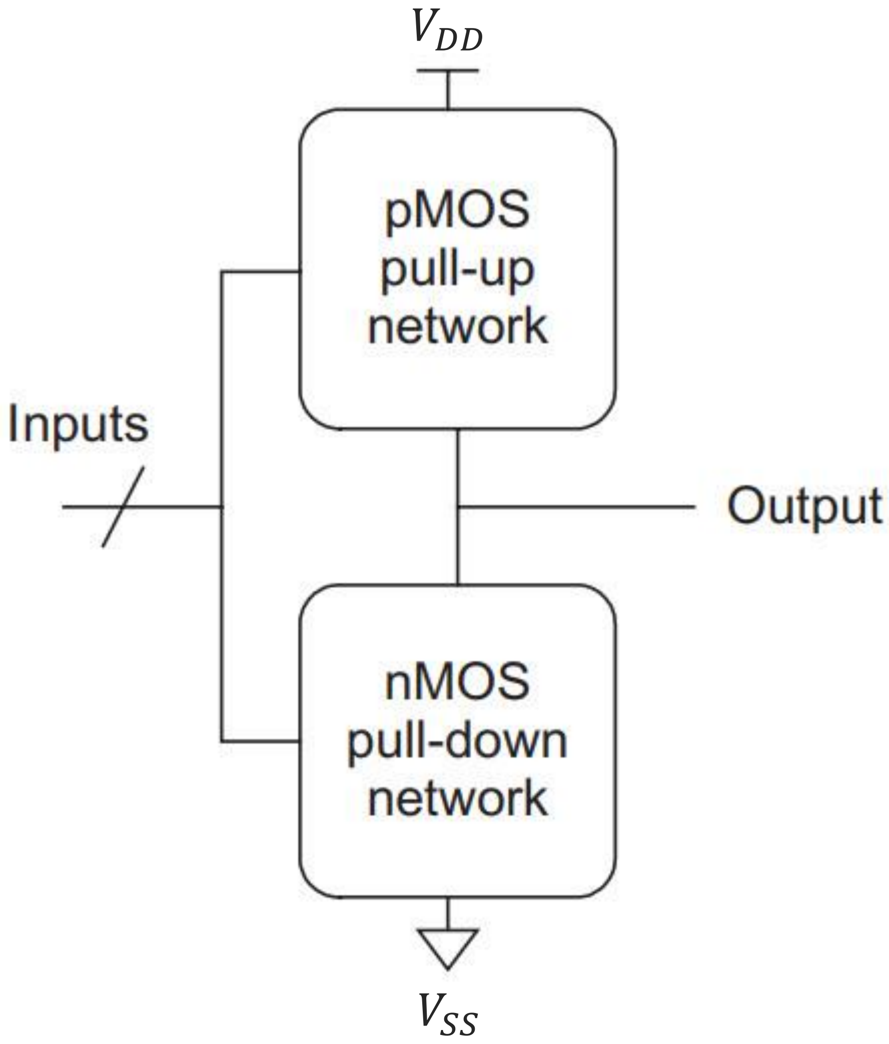
05 Architectures

BASIC ELEMENTS

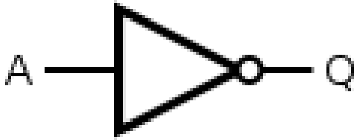
CMOS LOGIC

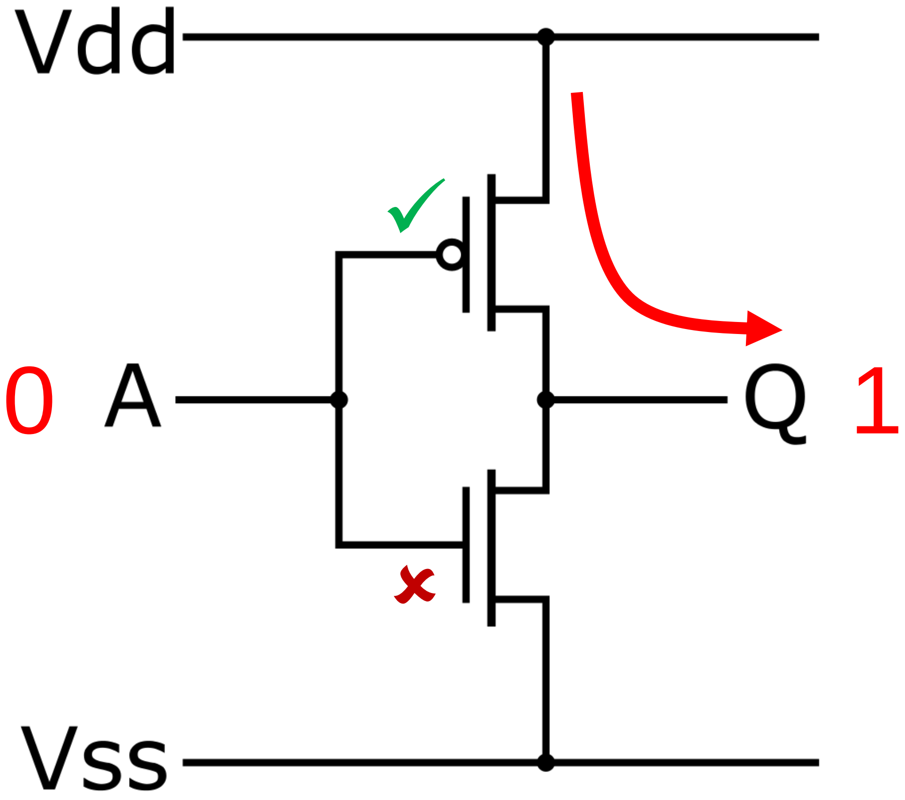
- Complementary Metal-Oxide Semiconductor
- MOS transistors used as switches
  - pMOS activated with LOW
  - nMOS activated with HIGH
- Logic gates composed from:
  - pMOS pull-up
  - nMOS pull-down



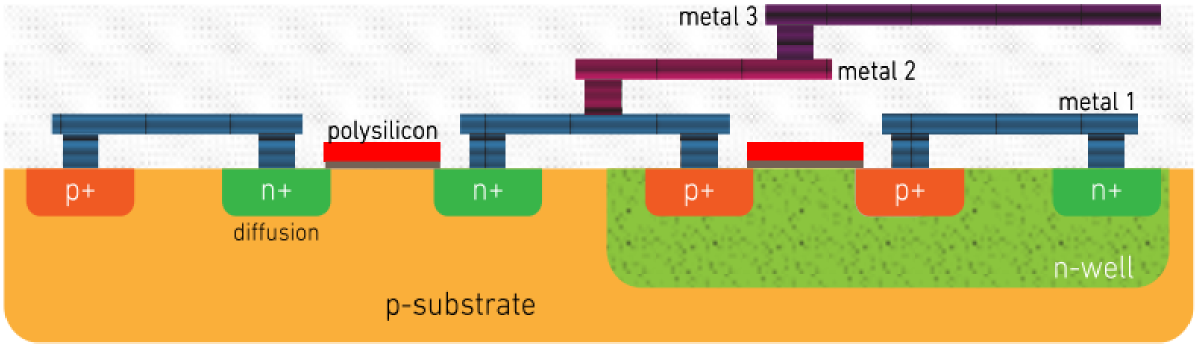


CMOS INVERTER

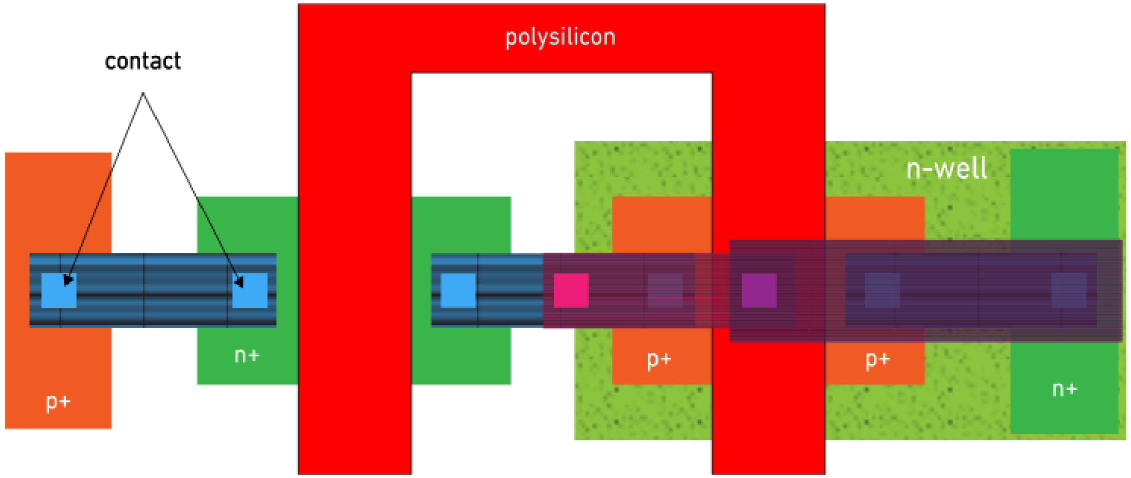




Cross section

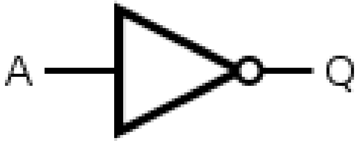


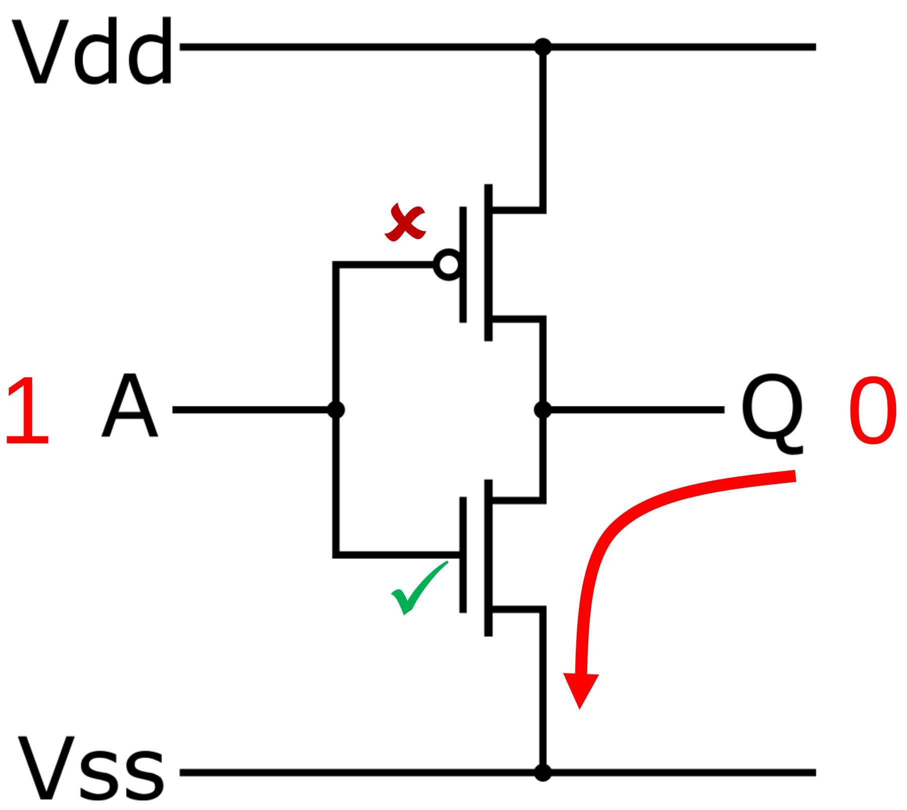
Top view



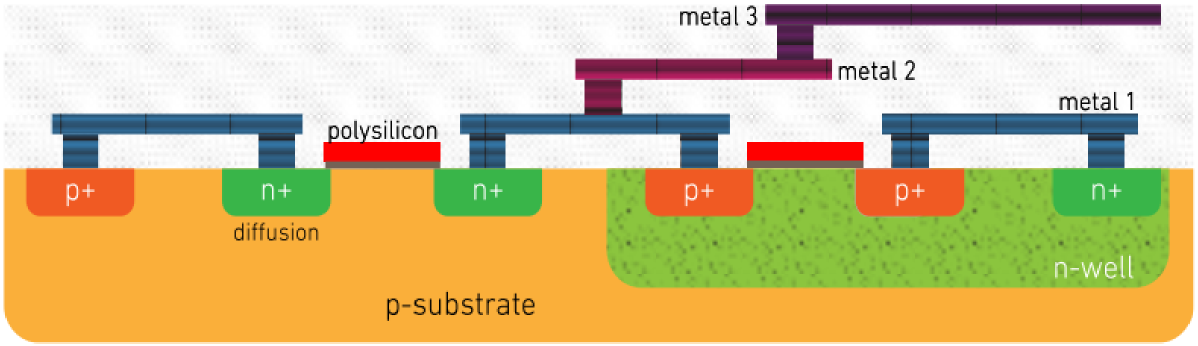
BASIC ELEMENTS

CMOS INVERTER

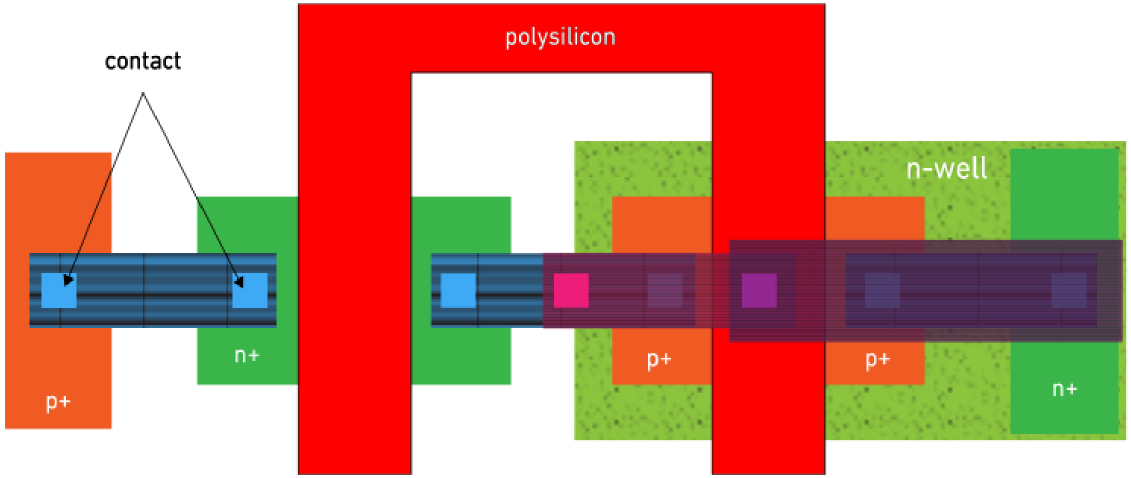




Cross section

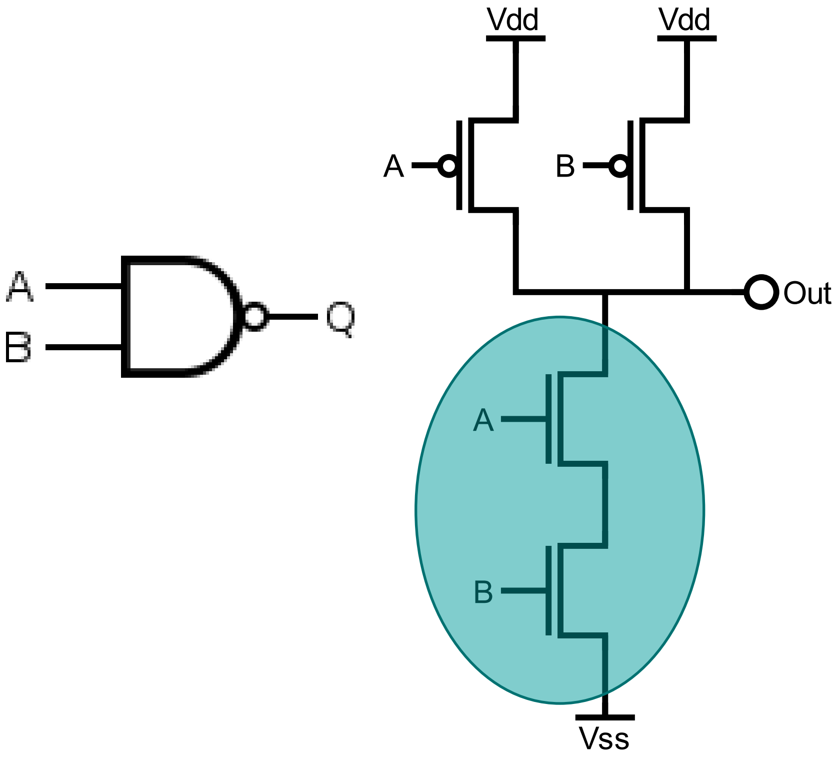


Top view

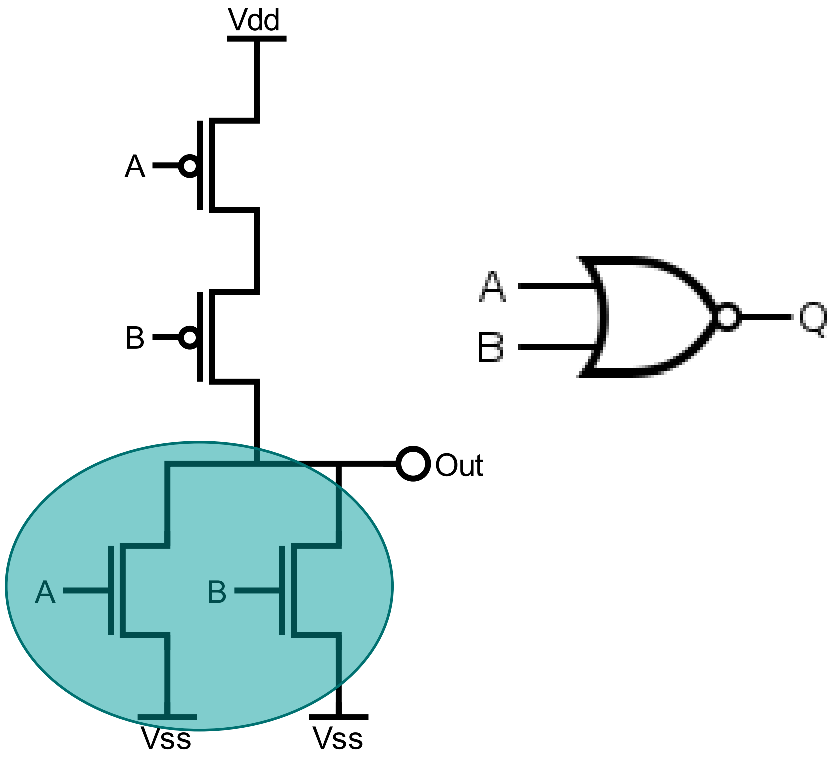


CMOS NAND/NOR

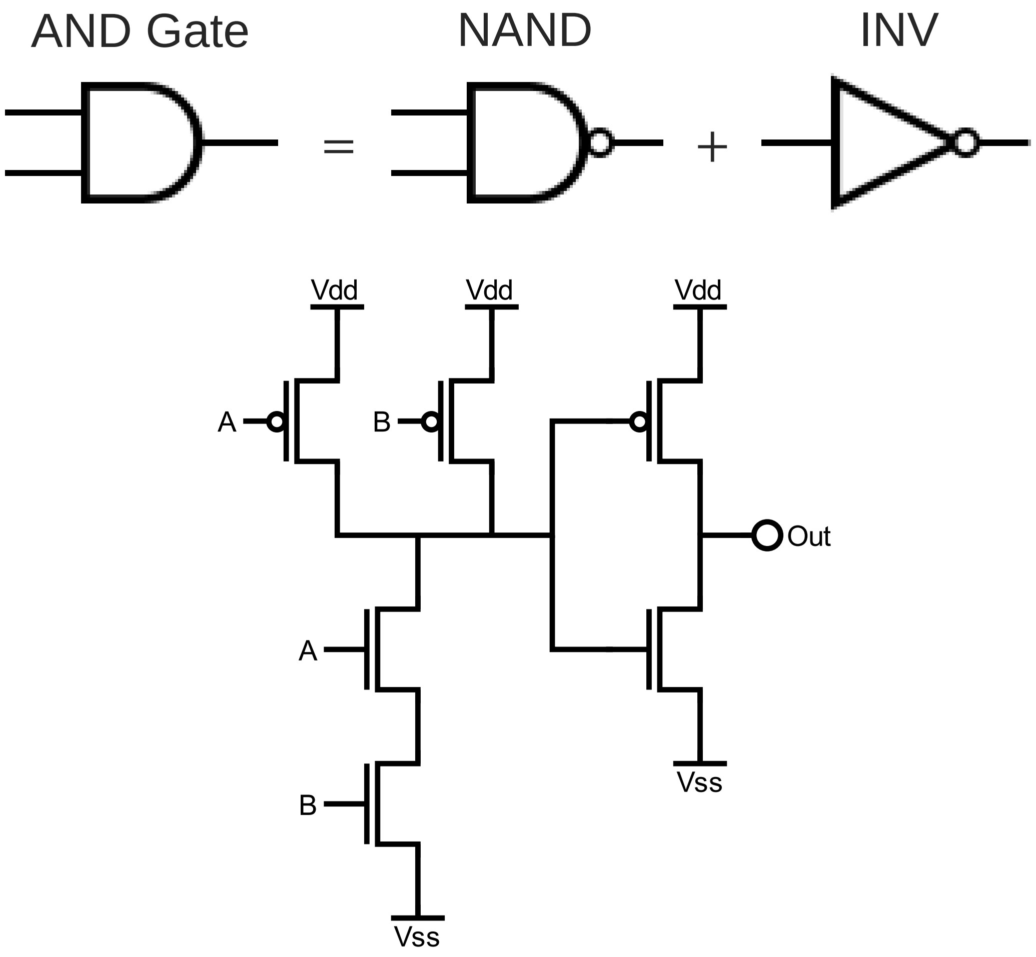
NAND Gate

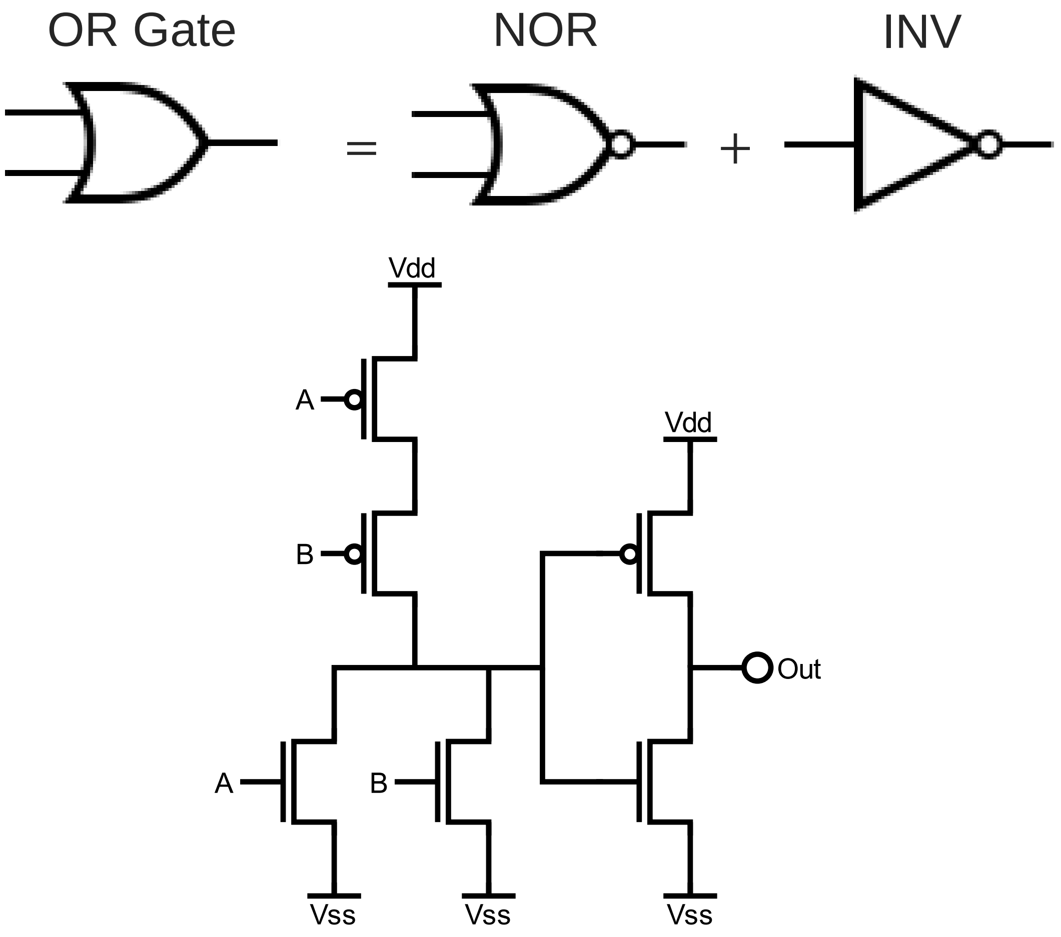


NOR Gate

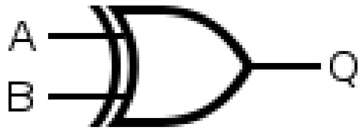


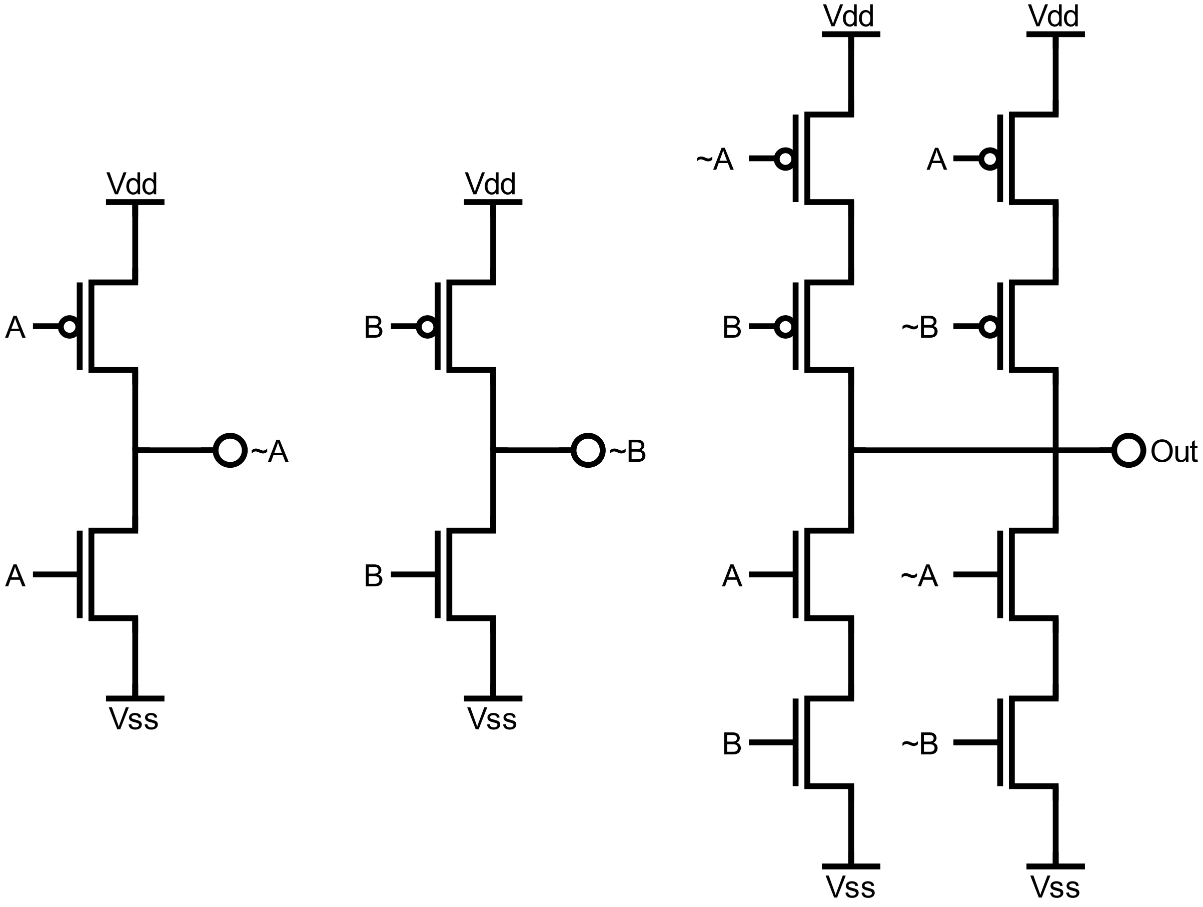
CMOS AND/OR


AND Gate
=
NAND
+
INV
Vdd
Vdd
Vdd
A
B
Out
A
B
Vss
Vss


OR Gate
=
NOR
+
INV
Vdd
A
B
Vdd
Out
A
B
Vss
Vss
Vss

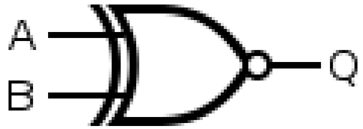
CMOS XOR/XNOR

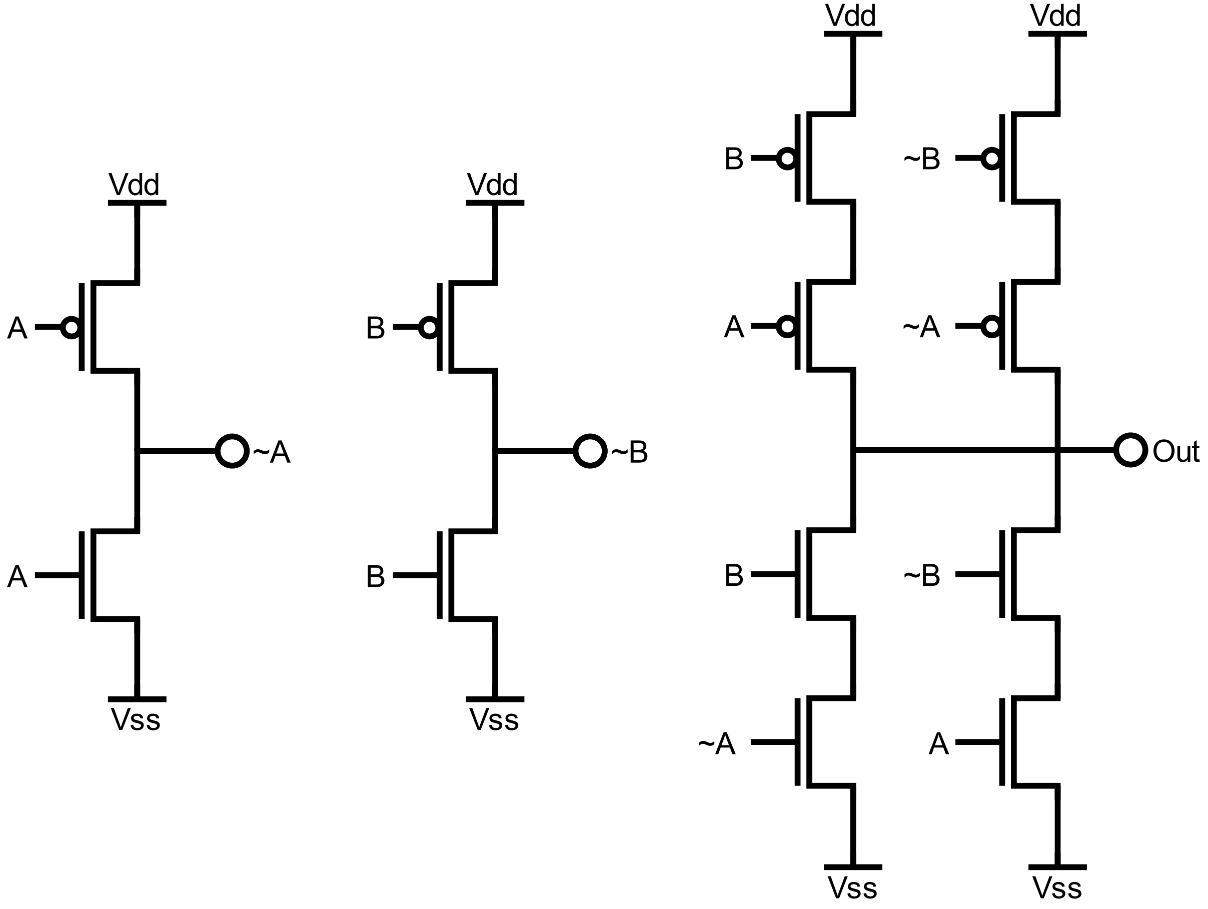




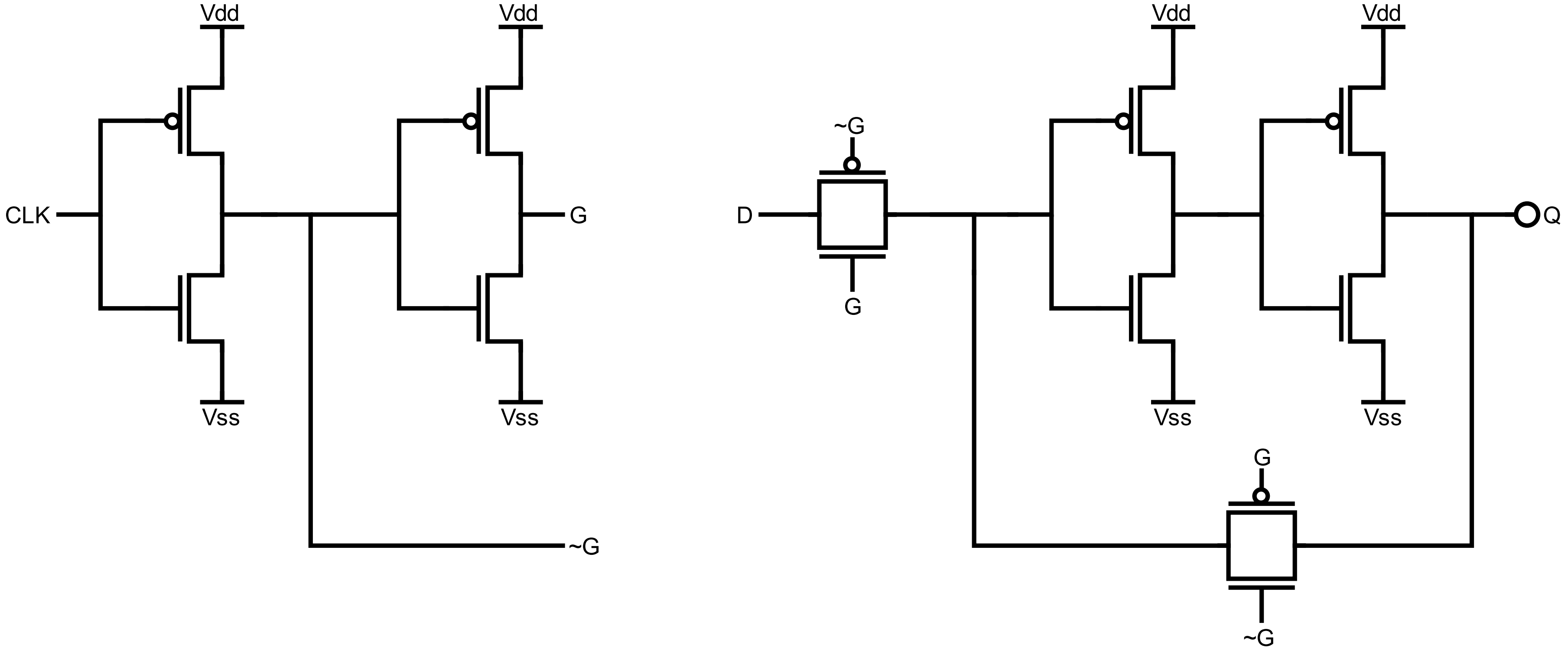
CMOS XOR/XNOR

- XNOR has same number of transistors

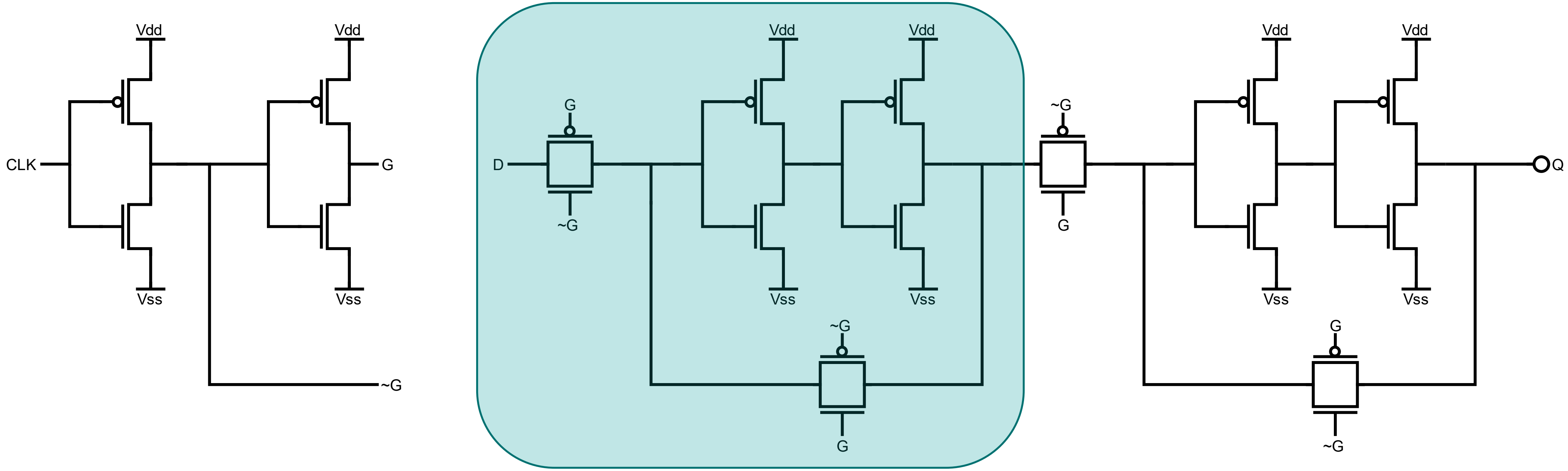




CMOS D-LATCH



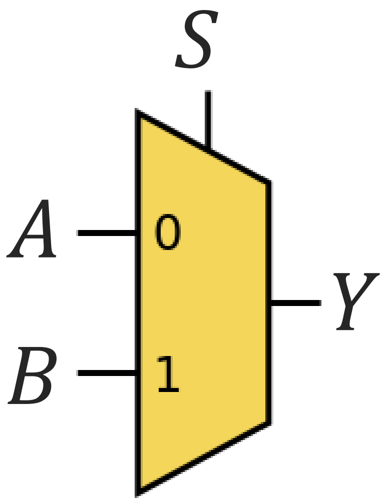
CMOS D-FLIP-FLOP

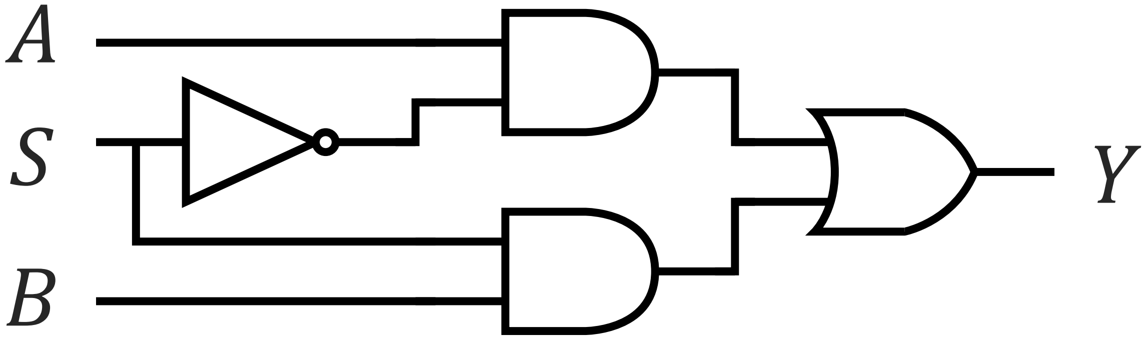


D-Latch Block ($\overline{CLK}$)

1BIT MULTIPLEXER

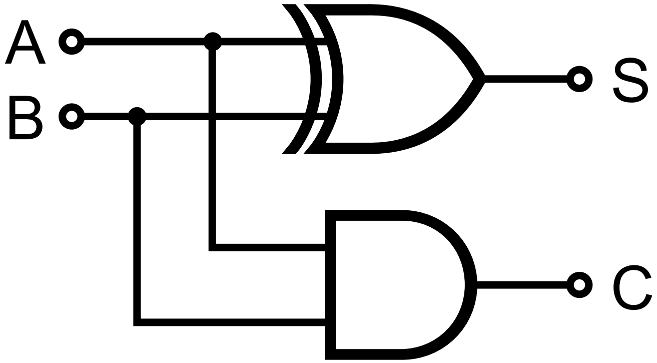
$$Y = A\bar{S} + BS$$



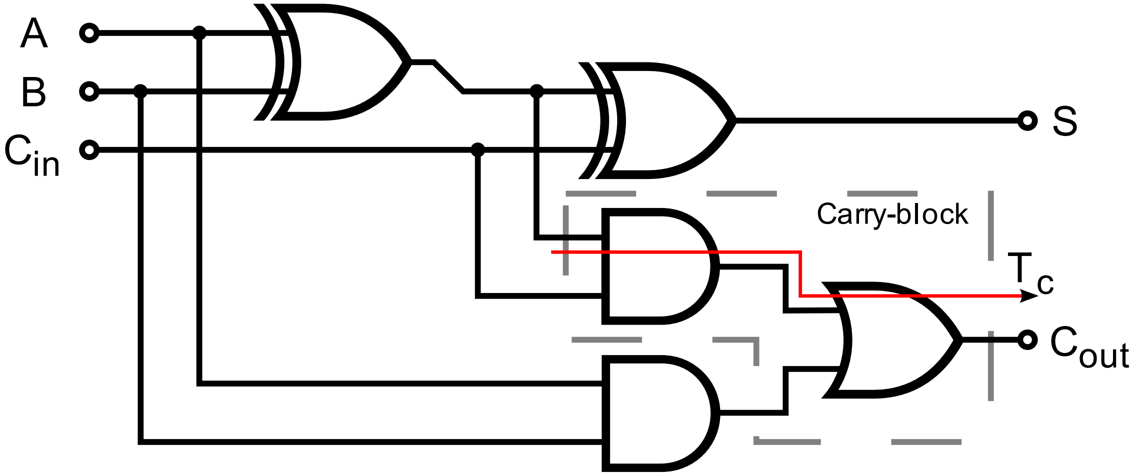


1-BIT ADDERS

Half Adder



Full Adder



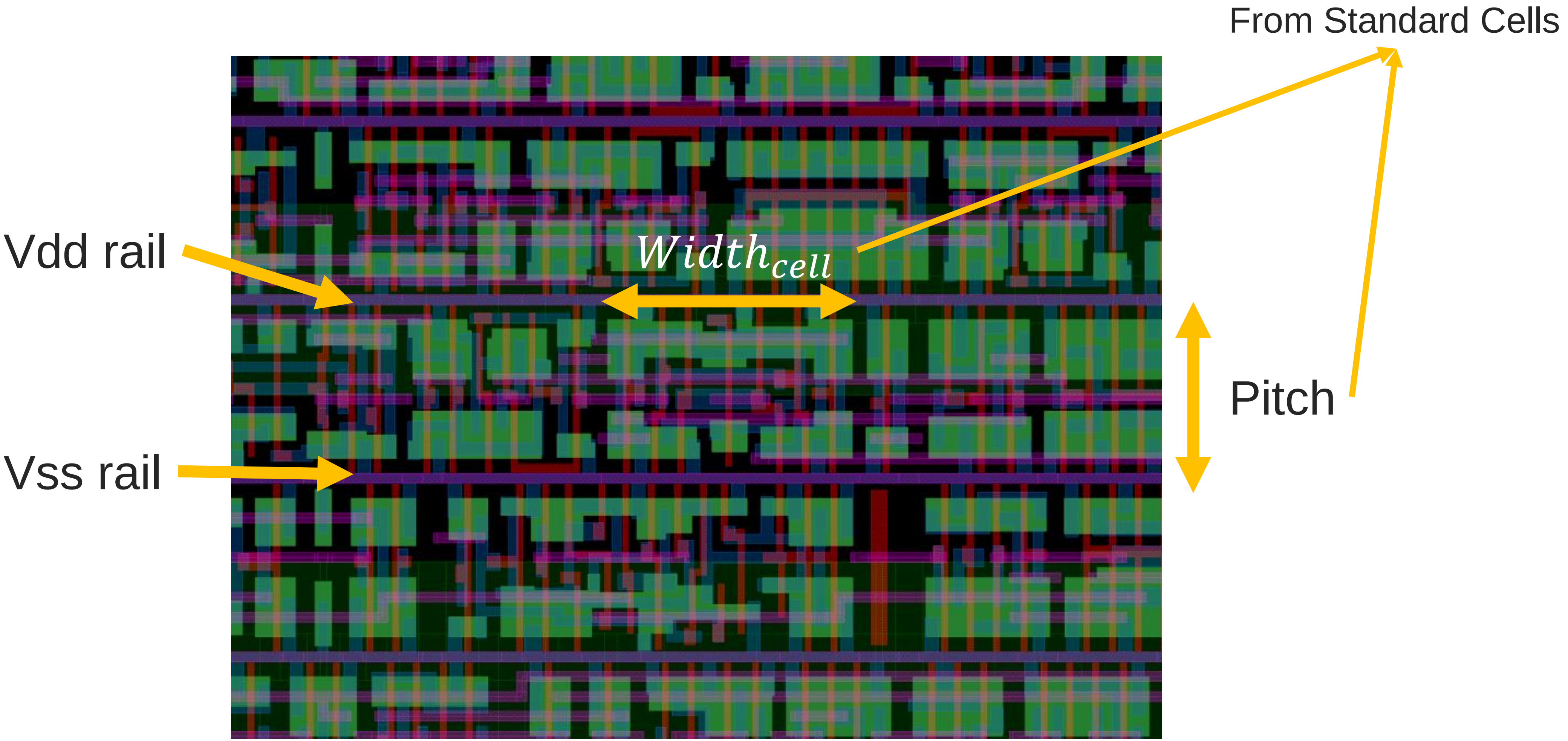
SUMMARY BASIC ELEMENTS

- Number of transistors:
  - INV: 2
  - NAND/NOR: 4
  - AND/OR: 6
  - XOR/XNOR: 12
  - D-Latch: 12
  - D-Flip-Flop: 20+
  - MUX: 20
  - HA: 18
  - FA: 42

AREA AND POWER

AREA

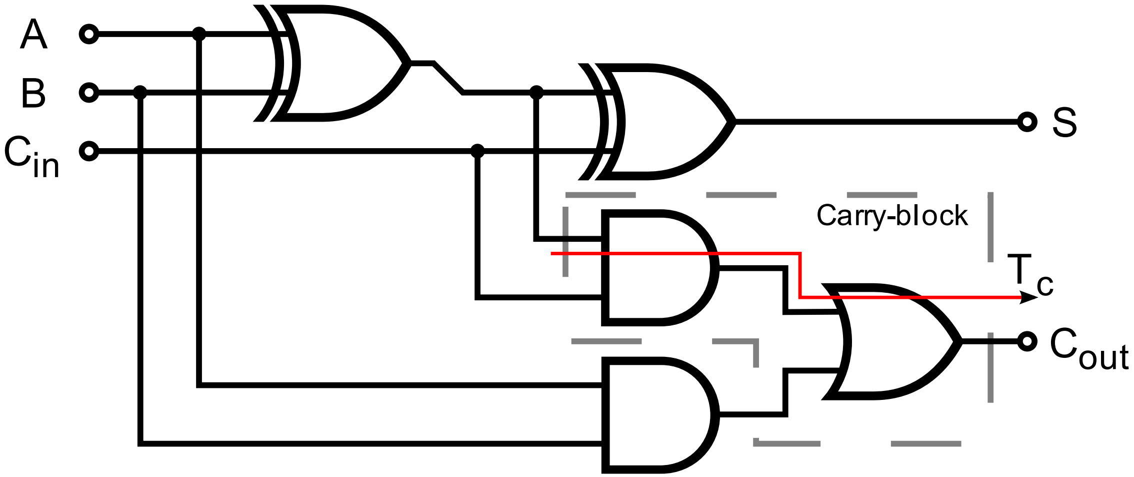
$$Area = \sum Area_{cell} = \sum Pitch \ * Width_{cell}$$



AREA: EXAMPLE

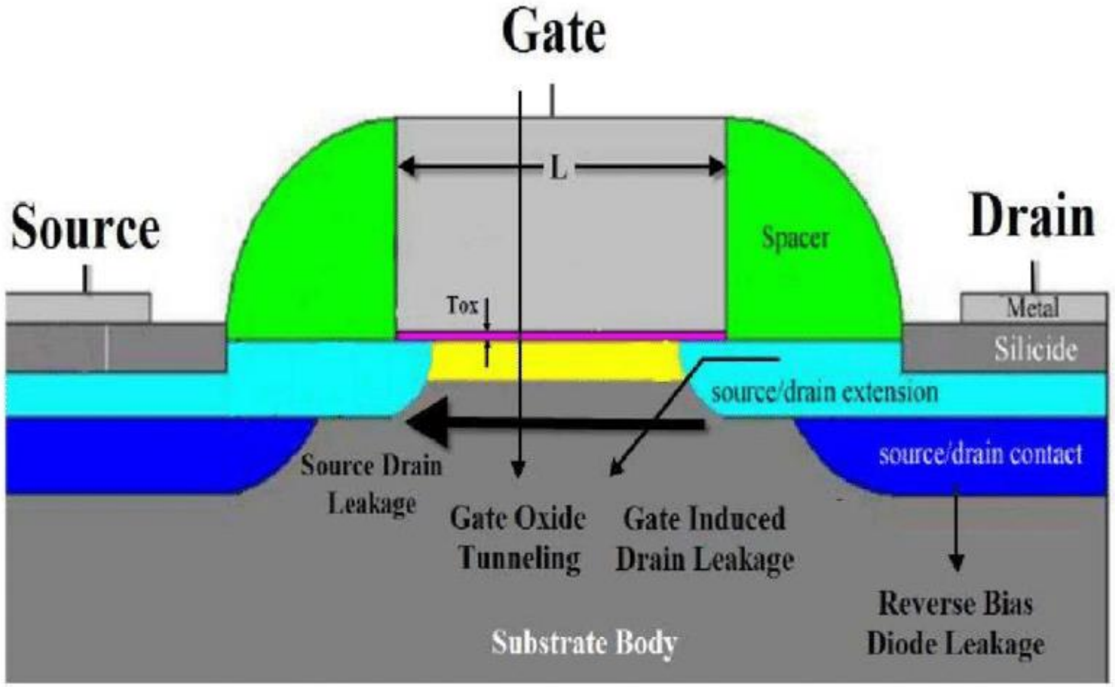
- Compute area of Full Adder:
  - XOR width: $2\ \mu m$
  - AND width: $1\ \mu m$
  - OR width: $1\ \mu m$
  - Pitch: $2\ \mu m$

$$Area_{FA} = 2 * Area_{XOR} + 2 * Area_{AND} + Area_{OR}$$
$$Area_{FA} = 2\mu m\ *\ (2 * 2\mu m + 2 * 1\mu m + 1\mu m)$$
$$Area_{FA} = 14\ \mu m^2$$



LEAKAGE POWER

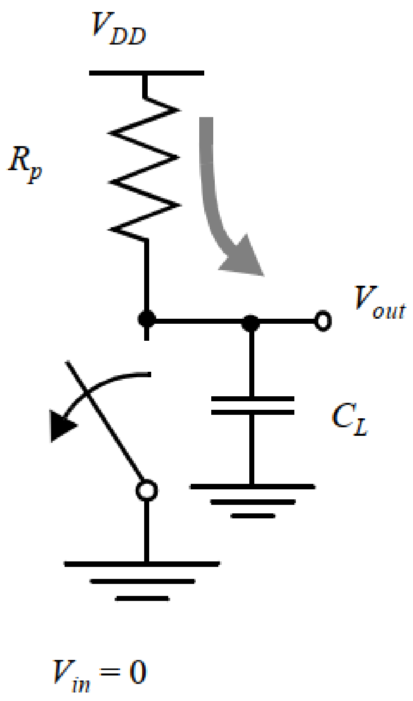
- Continuous power consumption
- Caused by:
  - Reverse bias current in parasitic diodes
  - Sub-threshold current
  - Gate oxide tunneling (thin gate oxide)
  - Gate-induced drain current



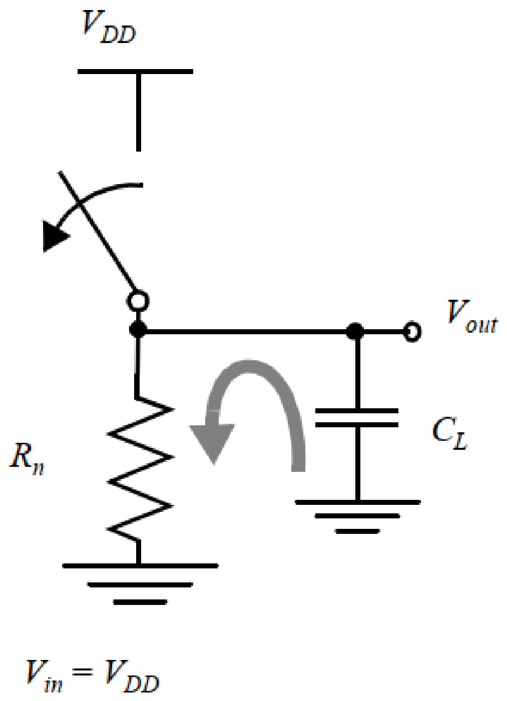
DYNAMIC POWER

- Power consumption during voltage transitions
- Depends on:
  - Transistors’ sizes (capacitance)
  - Transition (rising or fall) times
  - Device threshold
  - Temperature
  - Switching activity $(\alpha_{0\rightarrow 1}\ or\ \alpha_{1\rightarrow 0})$
    - Node transitions per clock

$$Power_{dyn} = \frac{1}{2} \times \alpha \times C_L \times V_{DD}^2 \times f$$



Load charge



Load discharge

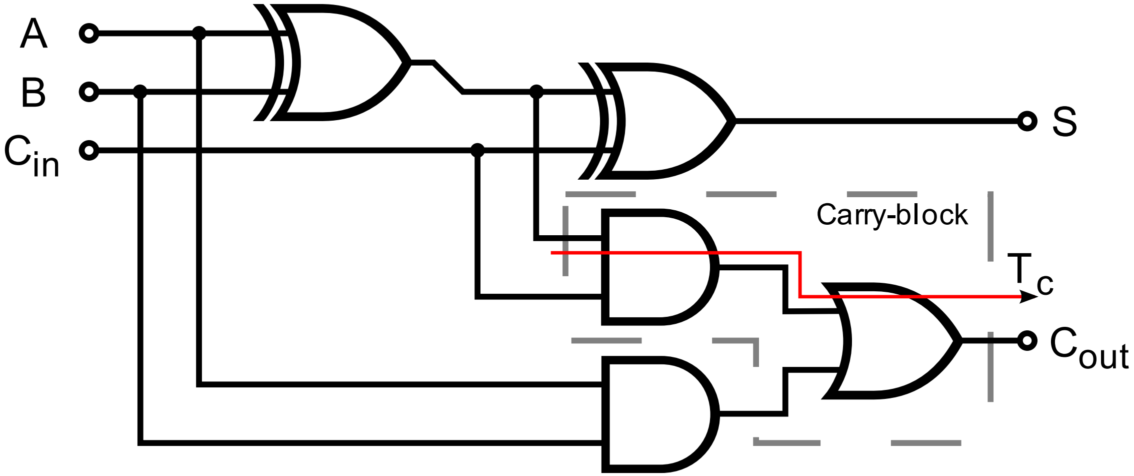
POWER CONSUMPTION

- Simplest way of estimating dynamic power consumption:

$$Power = \sum Power_{cell} = \sum leak_{cell} + \left(f_{CLK} \times \sum avg\_dyn\_power_{cell} \times \alpha_{cell} \right)$$

POWER: EXAMPLE

- Compute Full Adder's power consumption:
  - Typical Dynamic Power $[\mu W/MHz]$:
    - XOR: 0.0040
    - AND: 0.0022
    - OR: 0.0025
  - Leakage Power $[\mu W]$:
    - XOR: 0.000016
    - AND: 0.000014
    - OR: 0.000013
  - Clock frequency: $100\ MHz$
  - Switching Activity:
    - All gates have one transition every CLK
    - $\alpha_{cell} = 1$



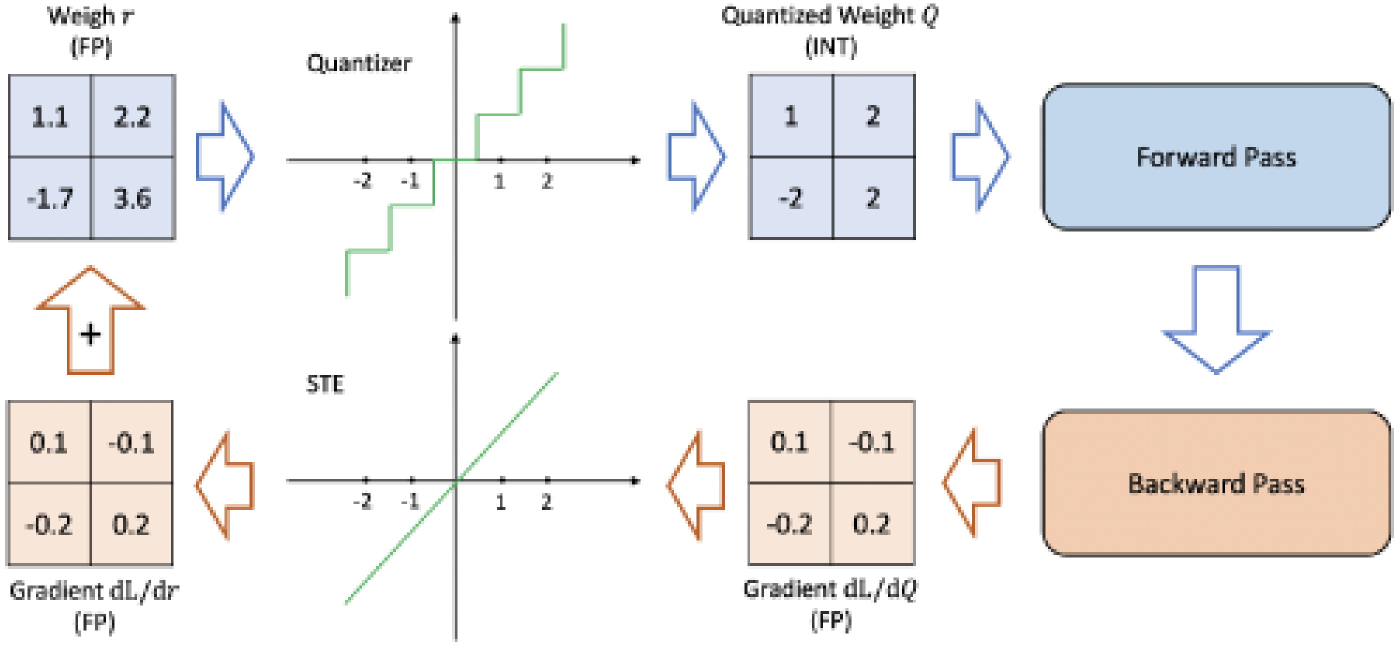
$$
\begin{aligned}
Power_{FA} &= 2*(100*0.004*1+0.000016)\\
&+2*(100*0.0022*1+0.000014)\\
&+(100*0.0025*1+0.000013)
\end{aligned}
$$

$$Power_{FA} = 1.490073\ \mu W$$

QUANTIZATION

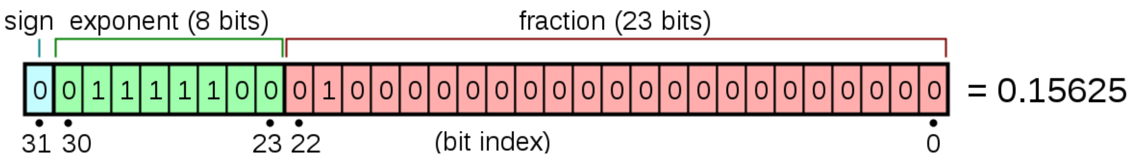
WHY QUANTIZATION

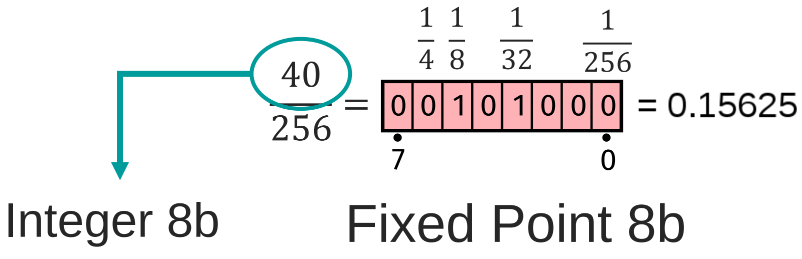
- Neural Networks run in CPU or GPU in general with FP operations
- VLSI/FPGA Accelerators mostly operate with INT values
- Quantization: mapping FP values into INT domain



FLOATING VS FIXED POINT

Floating Point 32b





$$Range_{fp32} = \begin{cases} [-3.40282347 \times 10^{38}, -1.17549435 \times 10^{-38}] \\ [0] \\ [1.17549435 \times 10^{-38}, 3.40282347 \times 10^{38}] \end{cases}$$

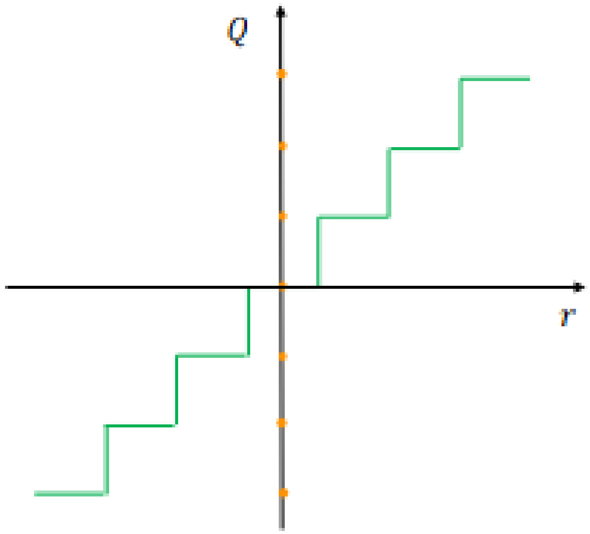
$$Range_{int8} = [-128, 127]$$

$$\downarrow$$

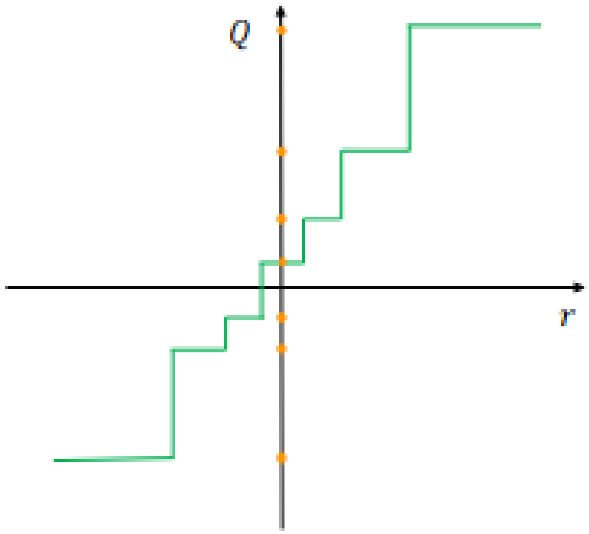
$$\left[\frac{-128}{Scale}, \frac{127}{Scale}\right]$$

TYPES OF QUANTIZATION

- Uniformity: Uniform vs Non-Uniform



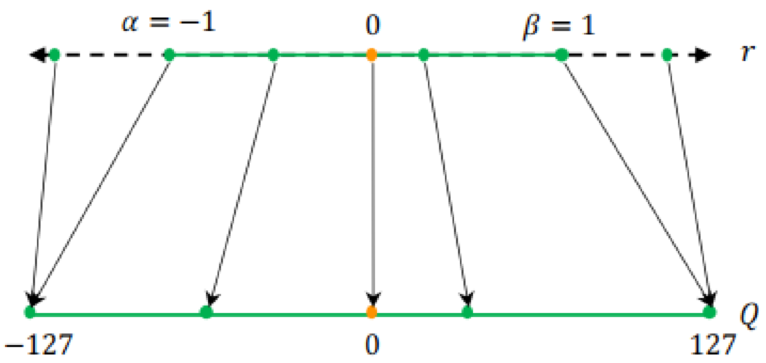
Uniform



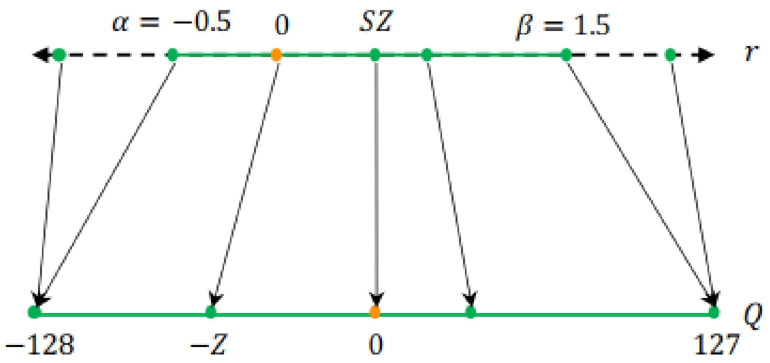
Non-Uniform

TYPES OF QUANTIZATION

- Uniformity: Uniform vs Non-Uniform
- Symmetry: Symmetric vs Asymmetric (both Uniform)



Symmetric



Asymmetric

$$Range = [\alpha, \beta]$$

$$Int(\,) = \begin{Bmatrix} Round \\ Trunc \\ Floor \end{Bmatrix} + clip$$

$$x_{int} = Int\left(S \times x_{fp}\right)$$

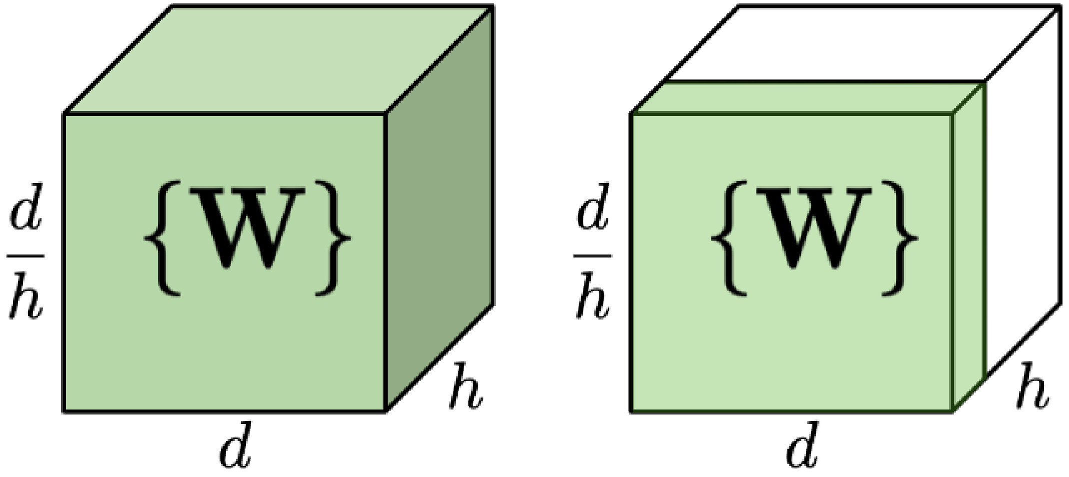
$$S = \frac{2^{b-1}}{max(|\alpha|, |\beta|)}$$

$$x_{int} = Int\left(S \times x_{fp}\right) - Z$$

$$S = \frac{2^{b-1}}{\beta - \alpha}$$

TYPES OF QUANTIZATION

- Uniformity: Uniform vs Non-Uniform
- Symmetry: Symmetric vs Asymmetric (both Uniform)
- Granularity: Tensorwise vs Channelwise



Tensorwise (Layerwise)

Channelwise

TYPES OF QUANTIZATION

- Uniformity: Uniform vs Non-Uniform
- Symmetry: Symmetric vs Asymmetric (both Uniform)
- Granularity: Tensorwise vs Channelwise
- Range Computation:
  - Dynamic (during runtime)
  - Static (after training)

QUANTIZED NEURON

- Example:
  - Uniform
  - Symmetric
  - Tensorwise

$$y_{fp} = b_{fp} + \sum x_{fp} \times w_{fp}$$

$$\frac{y_{int}}{S_y} = \frac{b_{int}}{S_b} + \sum \frac{x_{int}}{S_x} \times \frac{w_{int}}{S_w}$$

$$\Big\downarrow \quad S_y = S_b = S_x \times S_w$$

$$\boxed{y_{int} = b_{int} + \sum x_{int} \times w_{int}}$$

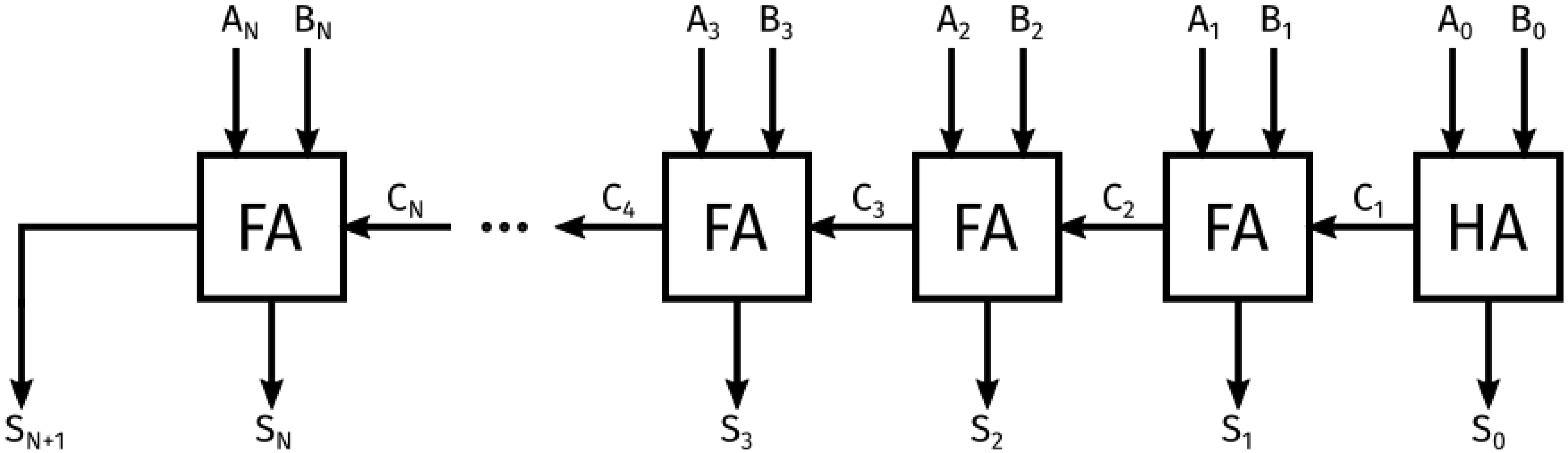
$$y_{fp} = \frac{y_{int}}{S_y}$$

$$\boxed{x_{int}^{l+1} = y_{int}^{l} \frac{S_x^{l+1}}{S_y^{l}}}$$

ADVANCED ELEMENTS

N-BIT ADDER

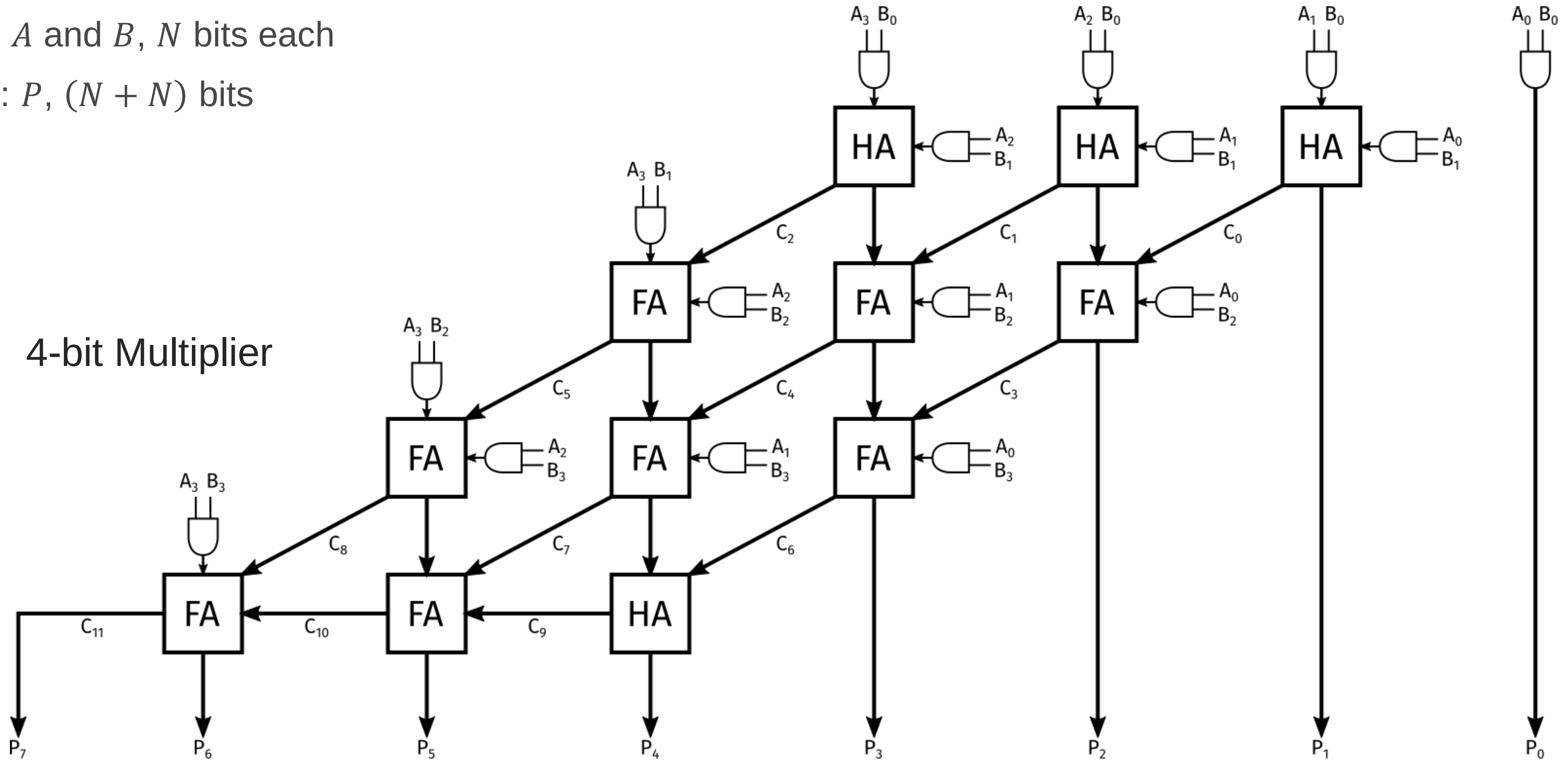
- Inputs: A and B, N bits each
- Output: S, $(N+1)$ bits



N-BIT MULTIPLIER

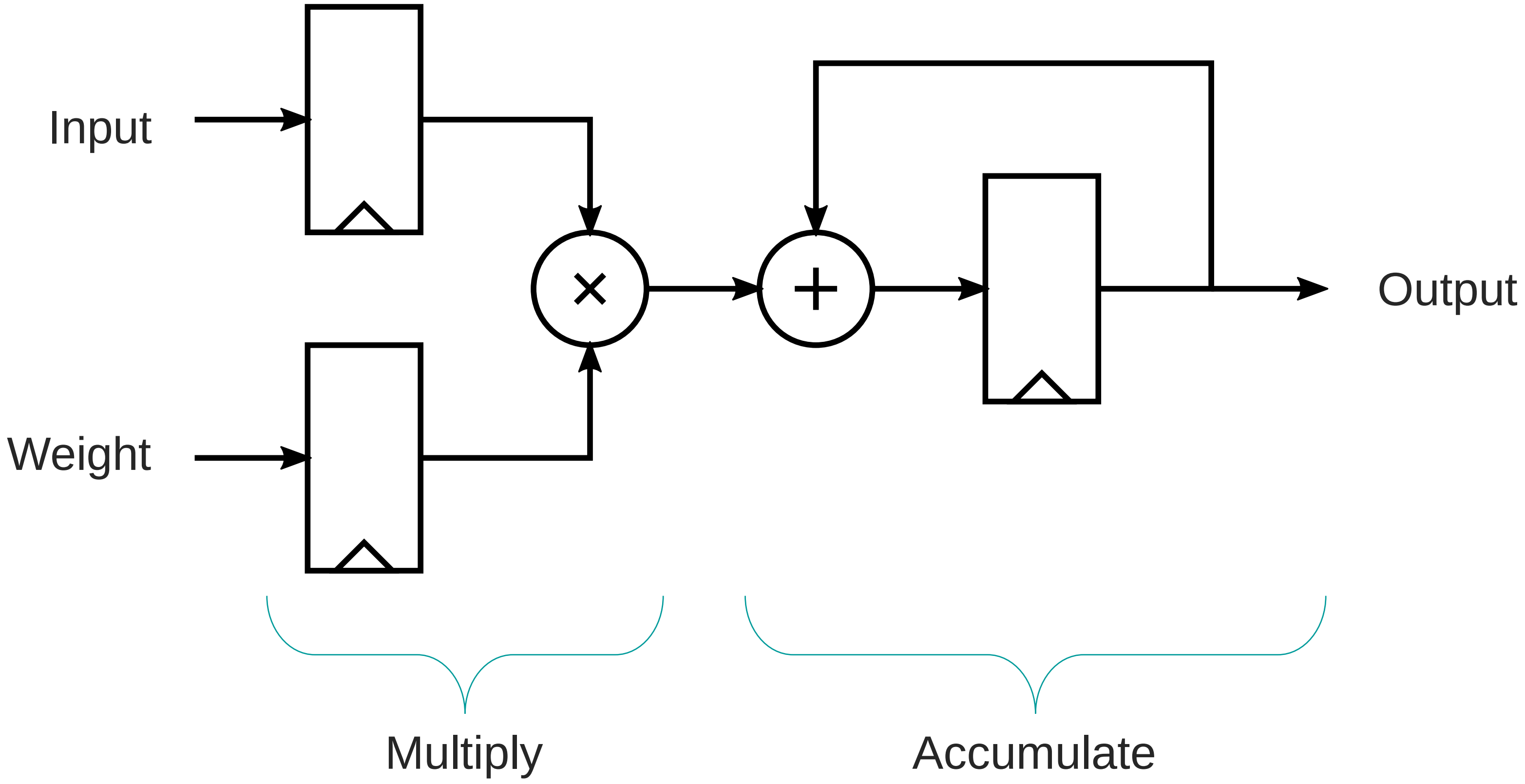
- Inputs: A and B, N bits each
- Output: P, $(N + N)$ bits

4-bit Multiplier



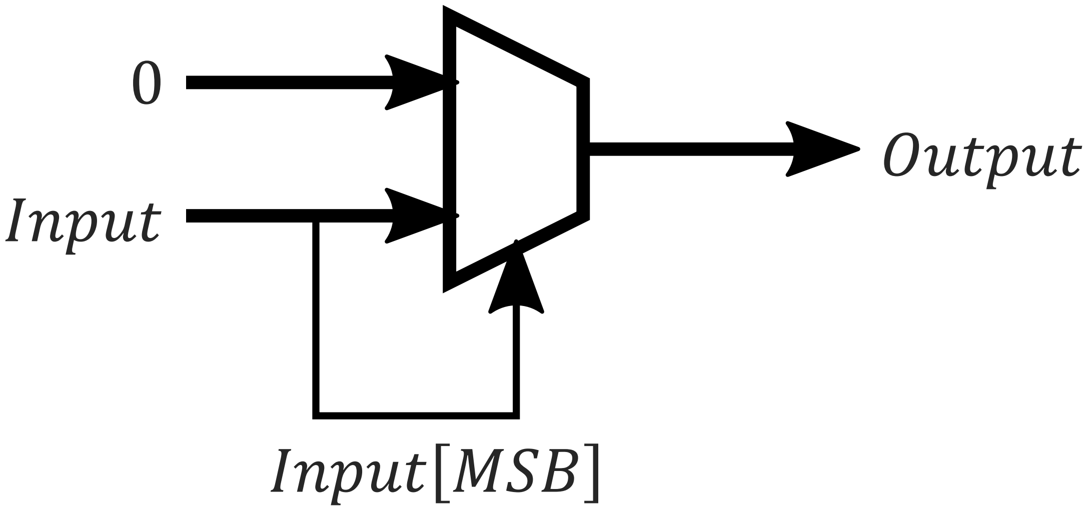
MAC

- Multiply and Accumulate operation



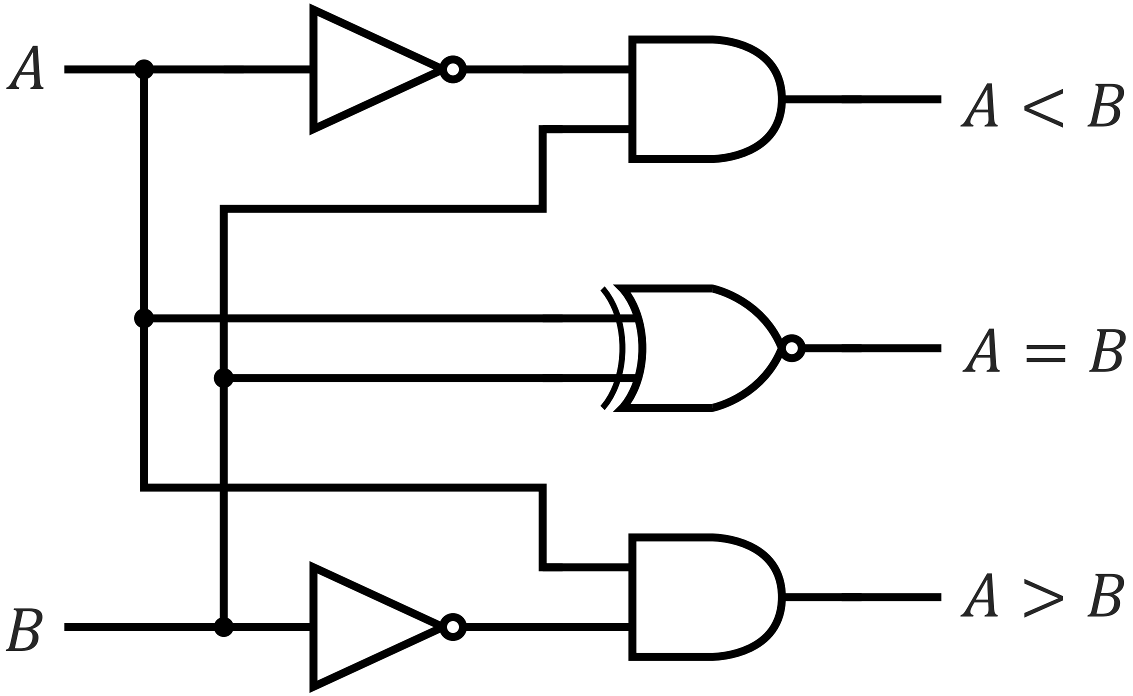
RELU OPERATOR

- Most Significant Bit (MSB) tells sign
  - Sign-magnitude
  - Ones’ complement
  - Two’s complement



COMPARATORS

- Equality $(A = B)$
- Inequality
  - $A > B$
  - $A < B$



1-bit Comparator

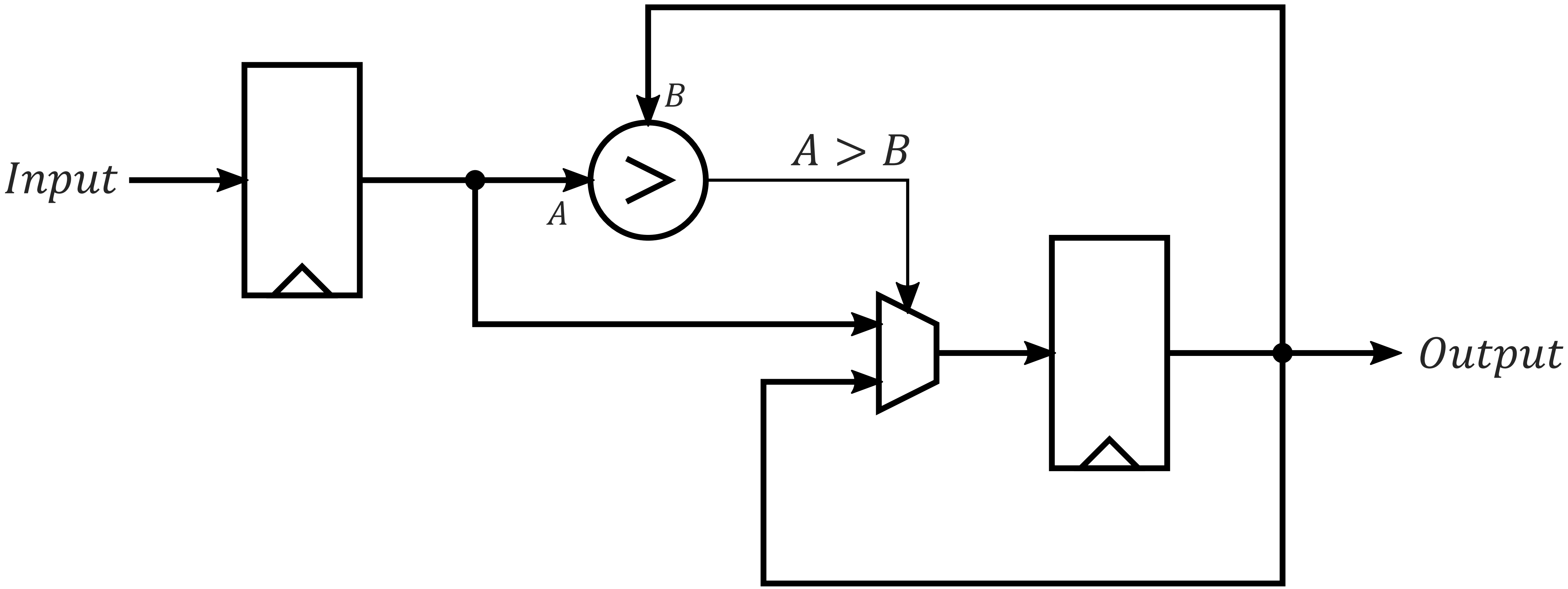
COMPARATORS

- Equality $(A = B)$
- Inequality
  - $A > B$
  - $A < B$

$$(A = B) = (A_N \oplus B_N) \cdots (A_1 \oplus B_1)(A_0 \oplus B_0)$$

$$x_i = (A_i = B_i) \left\{ \begin{array}{l} (A > B) = A_N\overline{B_N} + \cdots + (x_N \cdots x_2 A_1 \overline{B_1}) + (x_N \cdots x_2 x_1 A_0 \overline{B_0}) \\ \\ (A < B) = \overline{A_N} B_N + \cdots + (x_N \cdots x_2 \overline{A_1} B_1) + (x_N \cdots x_2 x_1 \overline{A_0} B_0) \end{array} \right.$$

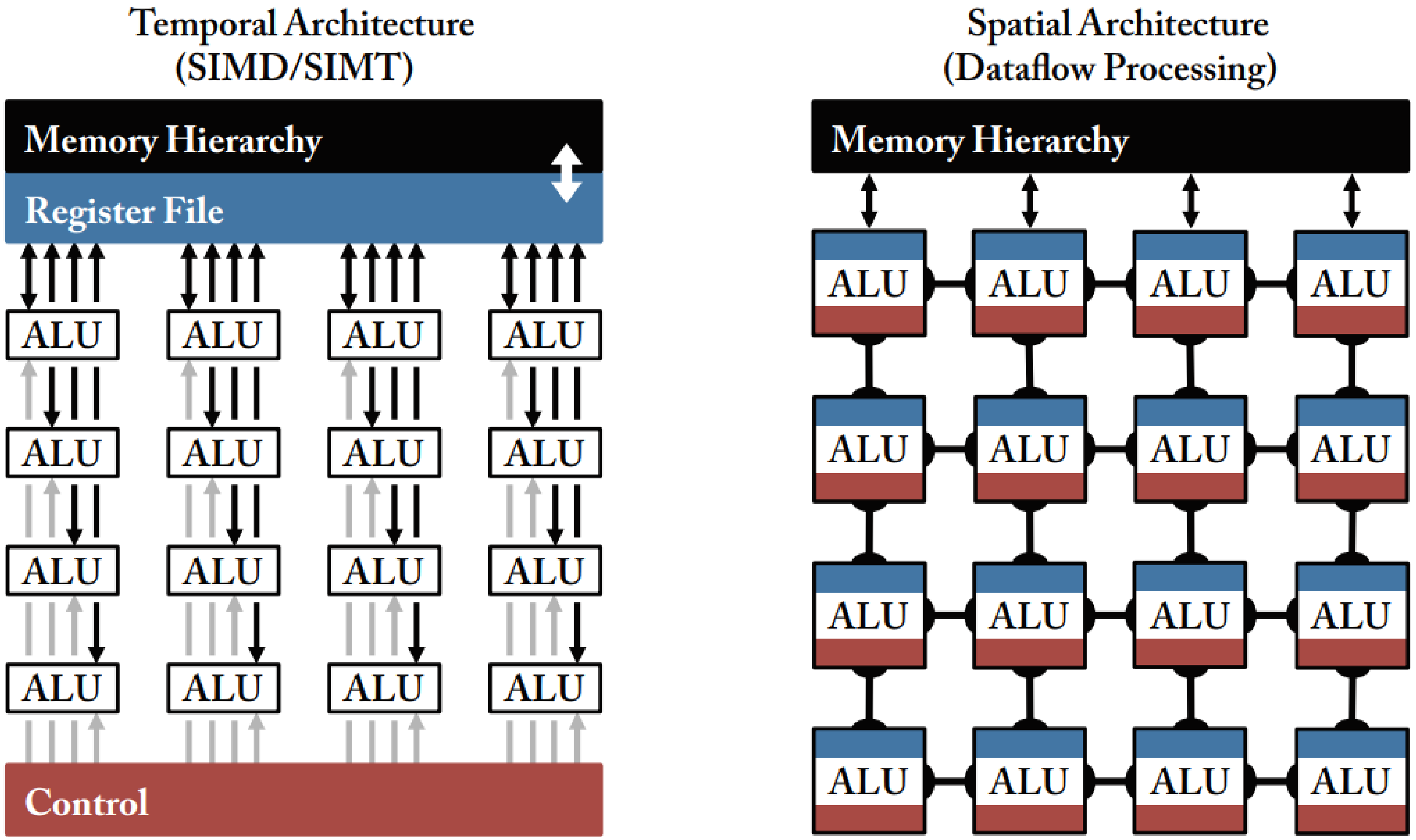
MAXPOOL OPERATOR



ARCHITECTURES

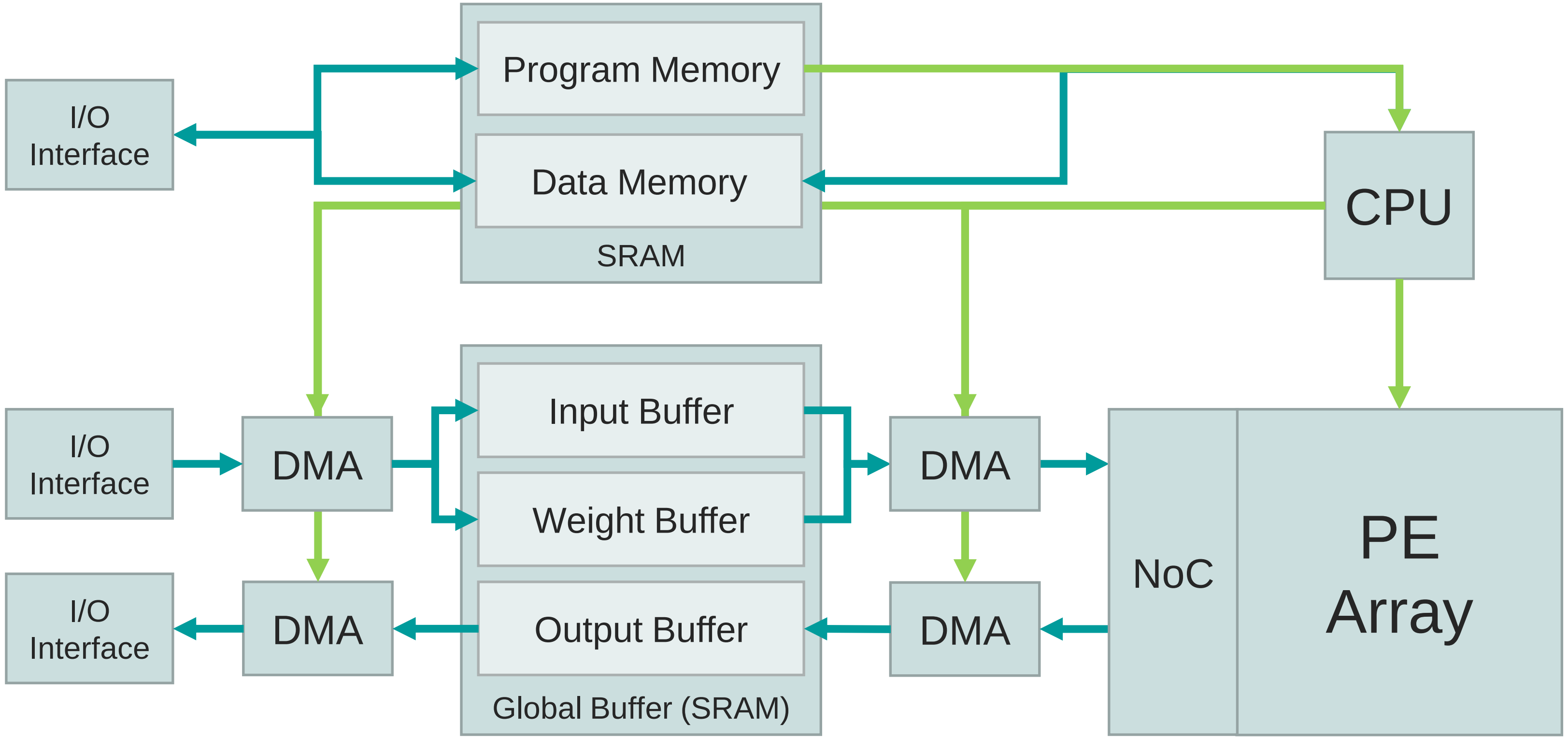
PARALLEL COMPUTING

- Two main paradigms
  - Temporal
    - Centralized Control
    - Centralized Memory
    - Mostly CPUs/GPUs
  - Spatial
    - Local Control
    - Local Memory
    - Inter-ALU Communication
    - Mostly ASIC/FPGA
    - ALUs + MEM + CTRL called PEs



ACCELERATOR BLOCKS

- Accelerator Main Blocks:
  - PE Array
  - CPU (Global Control)
  - SRAM
    - Program
    - Data
    - Accelerator Buffers
  - DMAs
  - NoC
    - Buffer – PE
    - PE - PE
  - I/O Interfaces

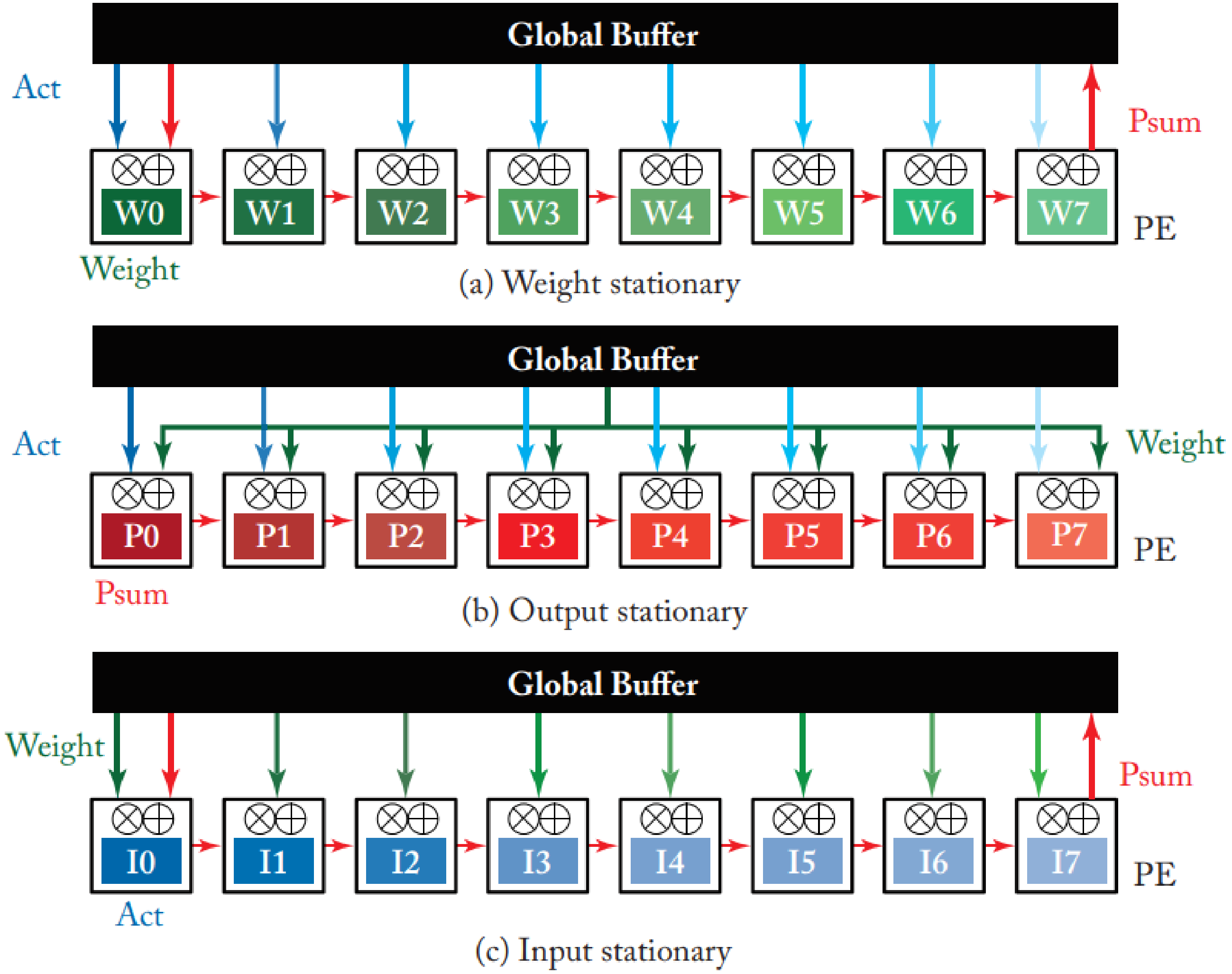


DATAFLOWS

- Exploit data reuse
- Dataflow styles:
    - Weight stationary
    - Input stationary
    - Output stationary

Conv Loop

```
for $nn$ in $Out_H$:
    for $mm$ in $Out_W$:
        for $ll$ in $Out_{Ch}$:
            for $kk$ in $In_{Ch}$:
                for $jj$ in $Ker_W$:
                    for $ii$ in $Ker_W$:
                        $sum(W \times In)$
```

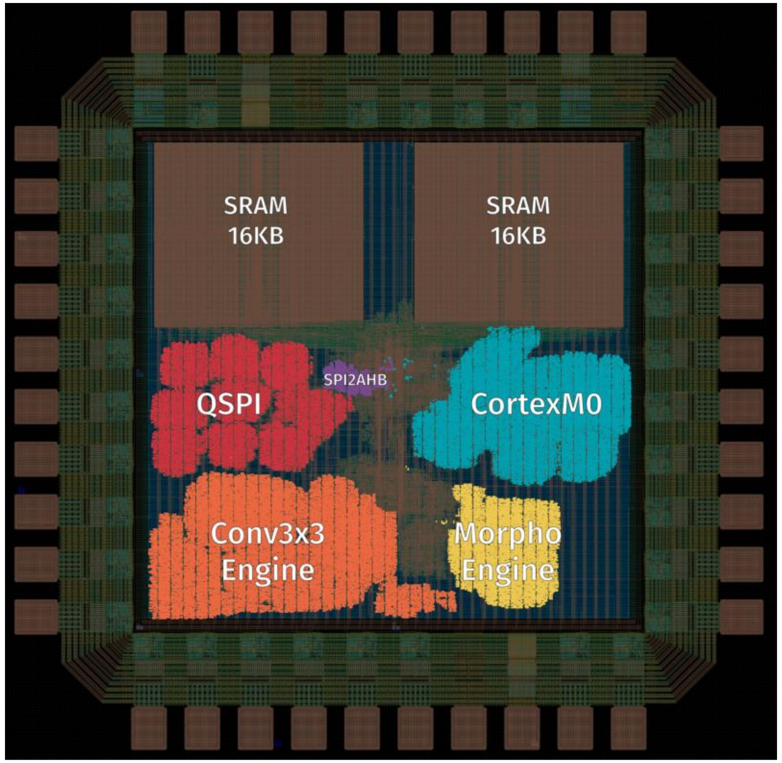


(a) Weight stationary

(b) Output stationary

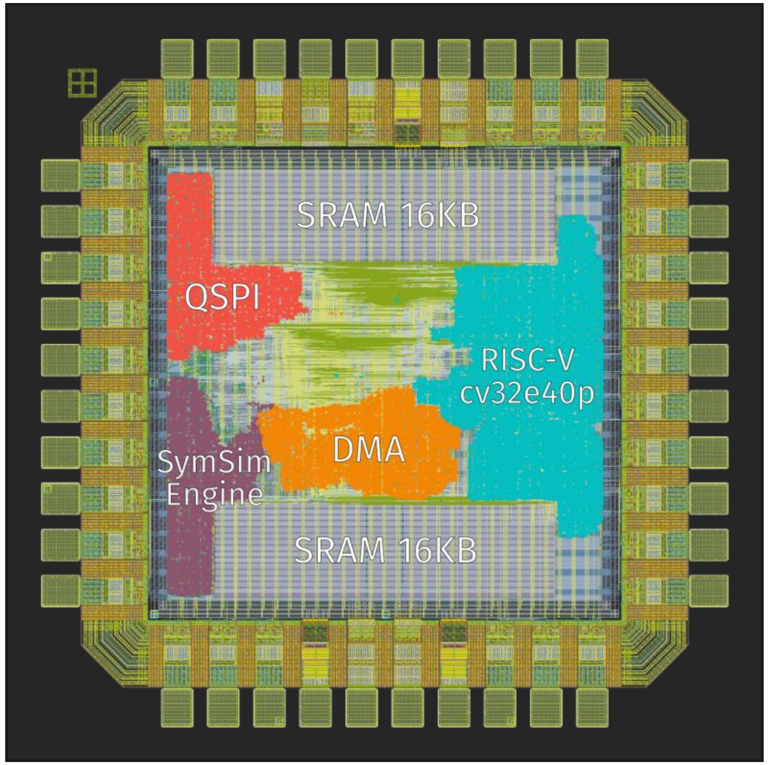
(c) Input stationary

ACCELERATOR CHIPS



DigineuronV1

	DigineuronV1 (measured)	DigineuronV2 (simulated)
Technology	65 nm	65 nm
Power Supply	1.2 V	1.2 V
Clock freq.	100 MHz	100 MHz
Power active	2.37 mW	2.29 mW
TOPS/W	0.58	1.17



DigineuronV2

QUESTIONS ?

SAL
SILICON AUSTRIA LABS

UNFOLD THE FUTURE