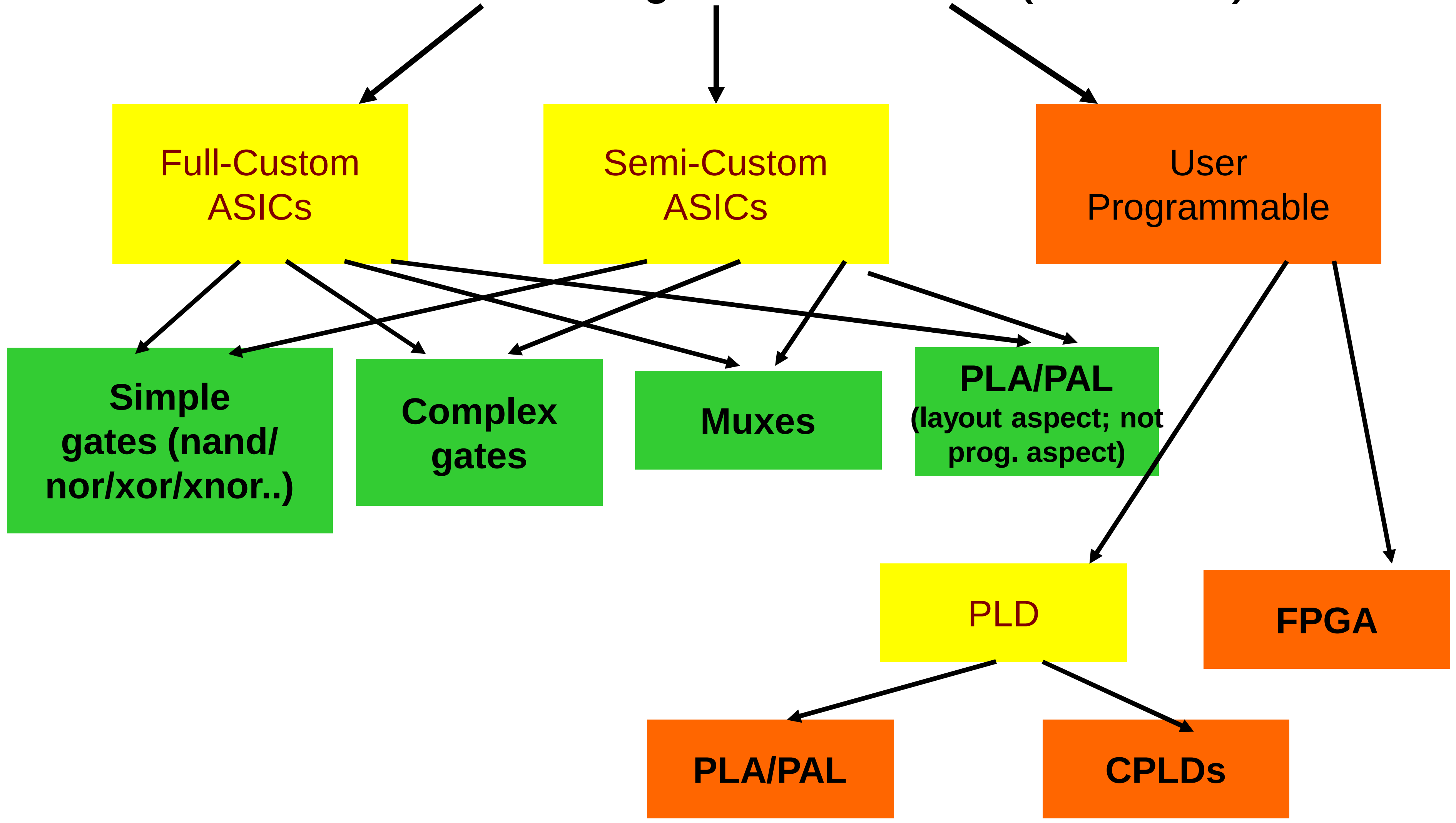
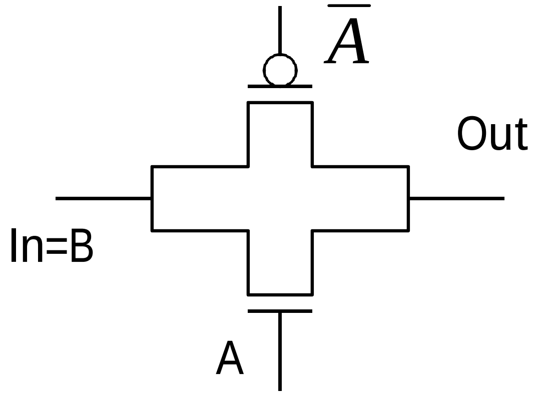
Steering Logic

The World of Integrated Circuits (LSI/VLSI)



<u>Logic Design Using Multiplexers</u>

A Transmission Gate (T-Gate)

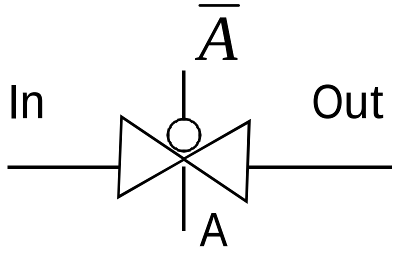


Steering gate.

When A=1, 'In' is "Steered" to 'Out'.
[I.e., the T-gate conducts]
Thus Out = B when A=1

$\Longrightarrow$ Out = AB

Symbolic for T-gate:



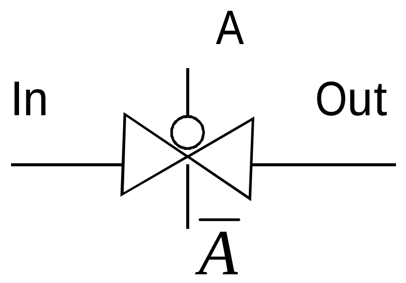
A is the control input (CI)
Normal CI connected to A
Bubbled CI connected to $\overline{A}$ $\Big\}$ T-gate conducts when A=1.

Steering Logic are the circuits that route data inputs to outputs based on the setting of control signals.

- For any combination of the control inputs there must be at least one conducting path from an input to the output. i,.e the output node should always be driven from some input.
- There should never be more than one conducting path between the inputs and an output. i.e in that case one path might attempt to drive the output node to logic 1 while another drives to logic 0.

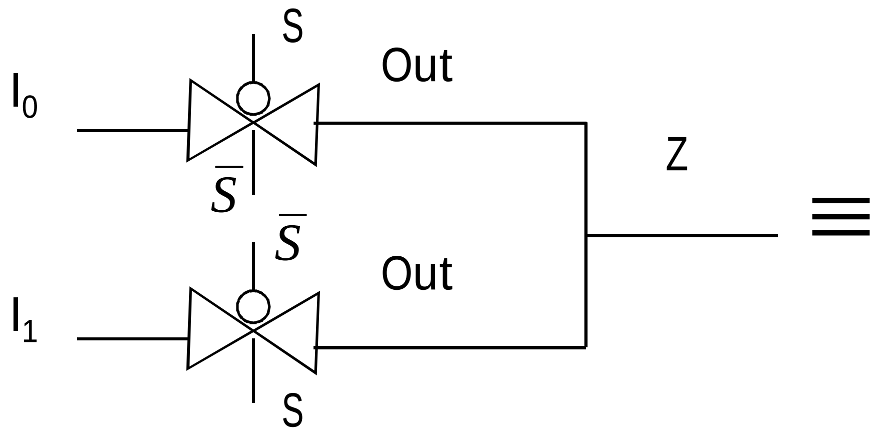
Reversing the connection of A: A $\longrightarrow$ bubble CI

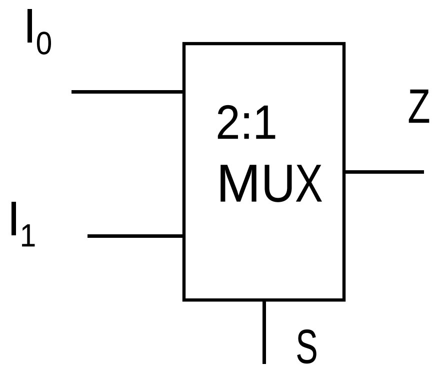
$\bar{A}$ $\longrightarrow$ normal CI



T-gate conducts when A=0.

<u>Multiplexer (MUX) Design</u>: A 2:1 MUX.



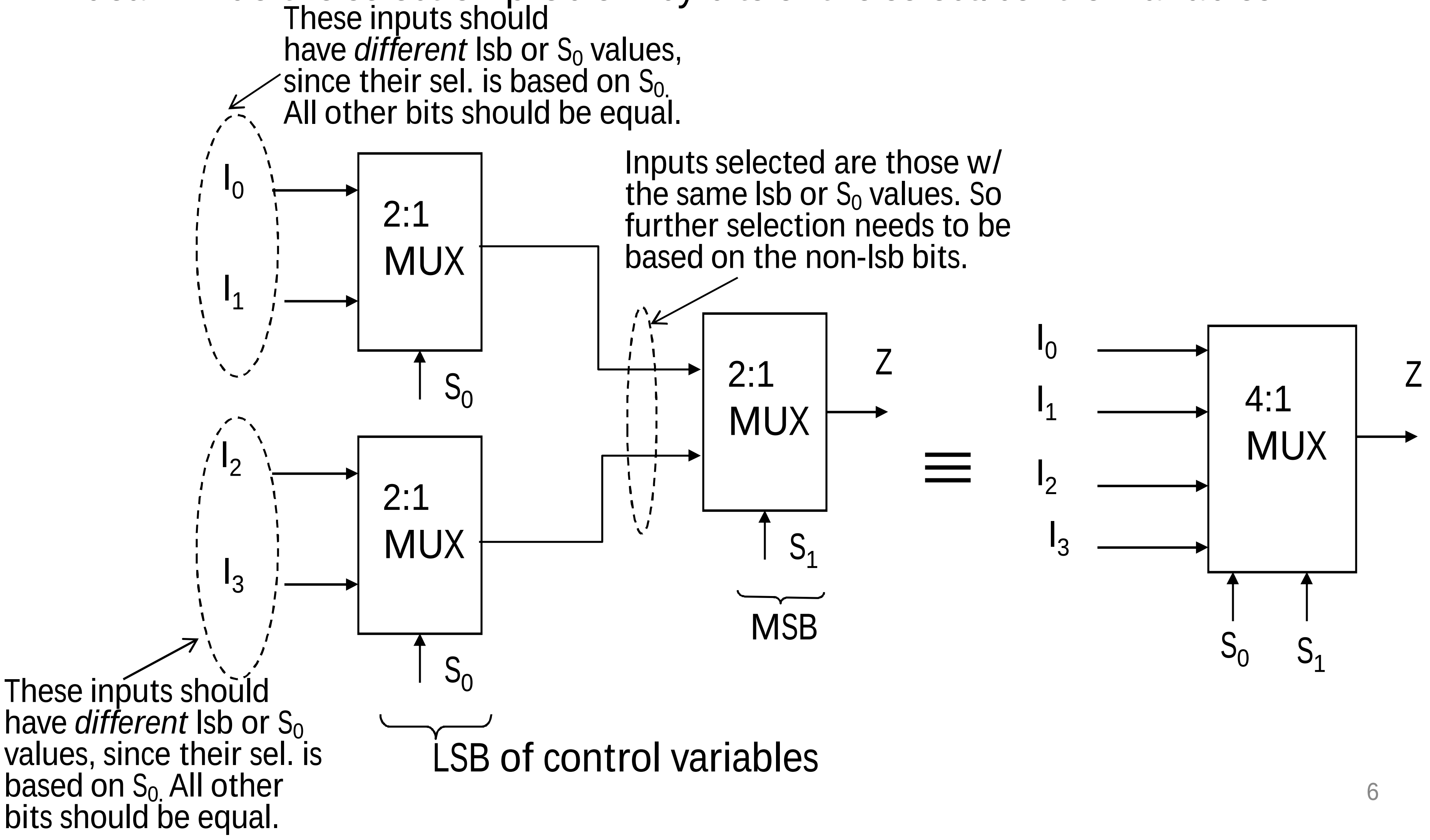


Z= I_0 when S=0
Z= I_1 when S=1

$$\Longrightarrow \quad Z = \bar{S}I_0 + SI_1$$

Design of MUXes using Divide-&-Conquer

- Saw the design of a 2:1 MUX using T-gates, as well as logic gates
- Messy and expensive to design larger MUXes using a flat TT based approach
- A 4:1 MUX can be hierarchically constructed using 2:1 MUXes
- Idea: Divide the selection problem by bits of the select/control variables

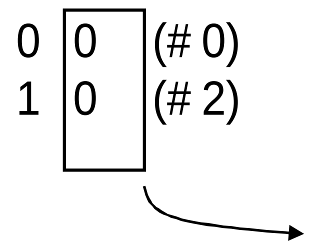

These inputs should have *different* lsb or S_0 values, since their sel. is based on S_0. All other bits should be equal.
I_0
I_1
I_2
I_3
2:1 MUX
2:1 MUX
S_0
S_0
Inputs selected are those w/ the same lsb or S_0 values. So further selection needs to be based on the non-lsb bits.
2:1 MUX
Z
S_1
MSB
≡
I_0
I_1
I_2
I_3
4:1 MUX
Z
S_0
S_1
These inputs should have *different* lsb or S_0 values, since their sel. is based on S_0. All other bits should be equal.
LSB of control variables

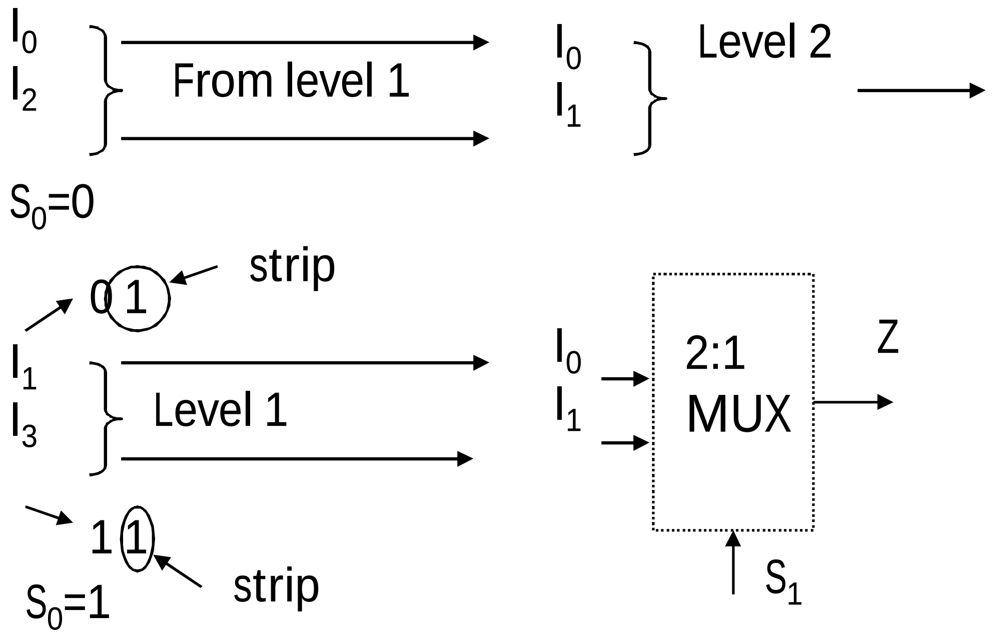
When S_0=0, I_0, I_2 get selected at the 1st level, i.e., Input w/ 0 in LSB.
When S_0=1, I_1, I_3 (LSB=1) get selected at the 1st level.

If S_0 = 0, I_0, I_2, become the 0th & 1st inputs to the next level.

At the next level, the I/P order # is determined by the rest of the bits of their index after stripping off the LSB.

$$\text{Thus}\quad \begin{array}{l} I_0 \\ I_2 \end{array} \begin{array}{c} \longrightarrow \\ \longrightarrow \end{array} \begin{array}{l} I_0 \\ I_1 \end{array} \Bigg\}\ \text{At level 2}$$





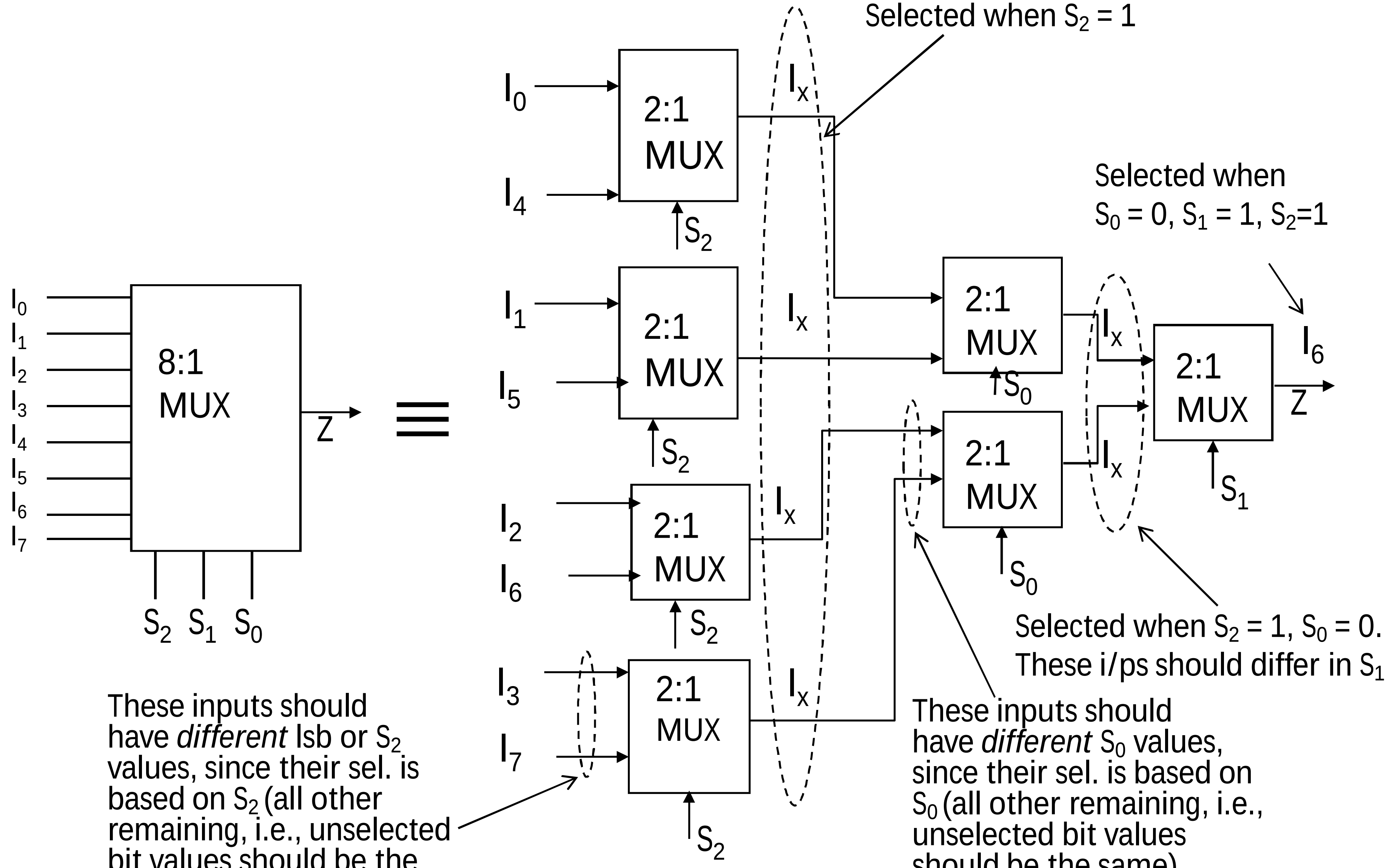
Z= I_0 of level 2
(I$_0$ of level 1), S_1=0.
= I_1 of level 2
(I_2 of level1) if S_1=1.

Z= I_0 of Level 2
(I_1 of Level1) if S_1=0

= I_1 of level 2
(I_3 of level 1) if S_1=1

Thus the design works as a 4:1 MUX.

8:1 MUX's: Input groupings for a different control variable order



These inputs should have *different* lsb or S_2 values, since their sel. is based on S_2 (all other remaining, i.e., unselected bit values should be the same). Similarly for other i/p pairs at 2:1 Muxes at this level.

These inputs should have *different* S_0 values, since their sel. is based on S_0 (all other remaining, i.e., unselected bit values should be the same). Similarly for other i/p pairs at 2:1 Muxes at this level.

Non-Gate Logic

Introduction

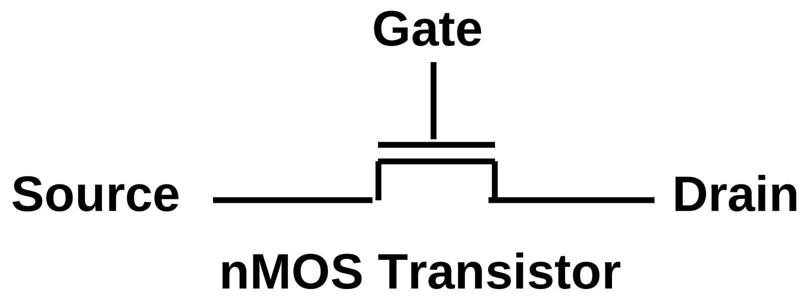
AND-OR-Invert	Generalized Building Blocks
PAL/PLA	Beyond Simple Gates

Kinds of "Non-gate logic":

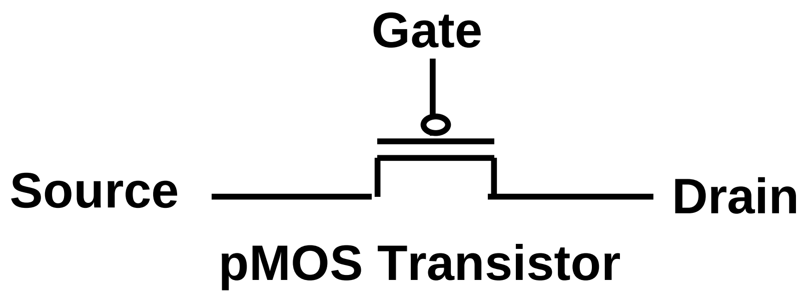
- switching circuits built from CMOS transmission gates, ROMs
- multiplexer/selecter functions
- decoders
- tri-state and open collector gates
- read-only memories

Steering Logic

Voltage Controlled Switches



Logic 1 on gate,
Source and Drain connected
Normally open switch



Logic 0 on gate,
Source and Drain connected
Normally closed switch

Steering Logic

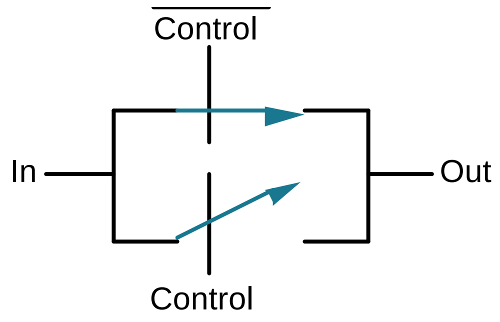
CMOS Transmission Gate

nMOS transistors good at passing 0's but bad at passing 1's

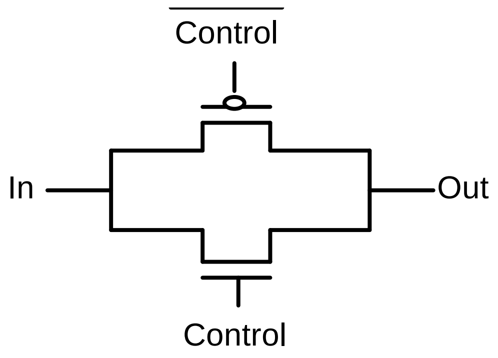
pMOS transistors good at passing 1's but bad at passing 0's

perfect "transmission" gate places these in parallel:

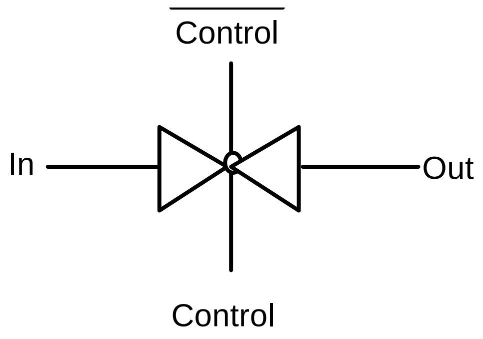
Steering logic circuit – route data inputs to outputs based on the settings of control signals. Ex) selector fun or mux



Switches



Transistors



Transmission or "Butterfly" Gate

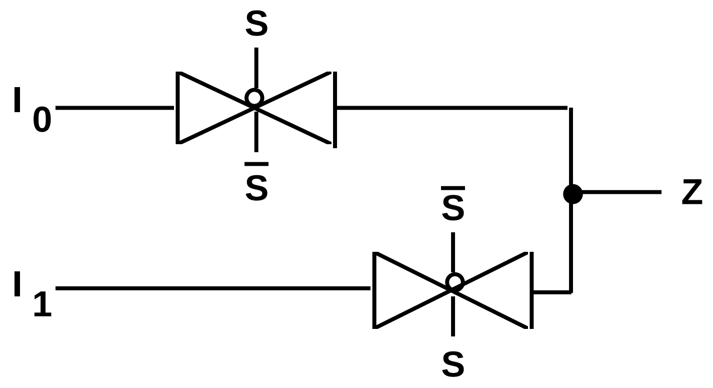
Steering Logic

Selection Function/Demultiplexer Function with Transmission Gates

Selector:

Choose I0 if S = 0

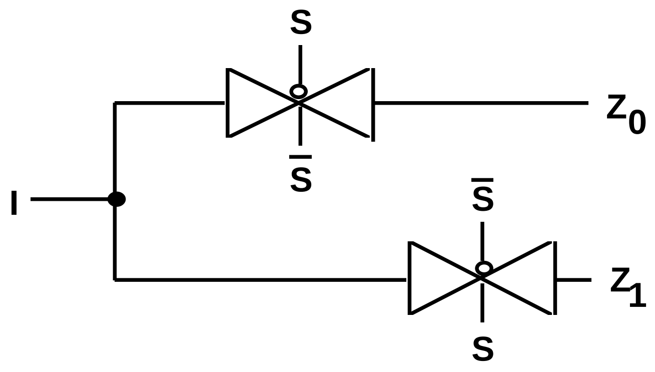
Choose I1 if S = 1



Demultiplexer:

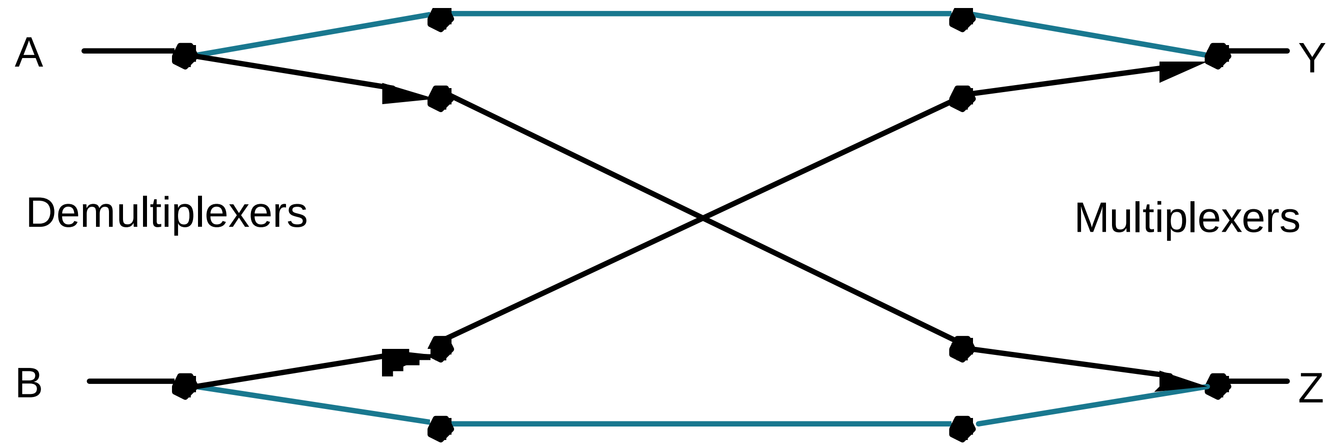
I to Z0 if S = 0

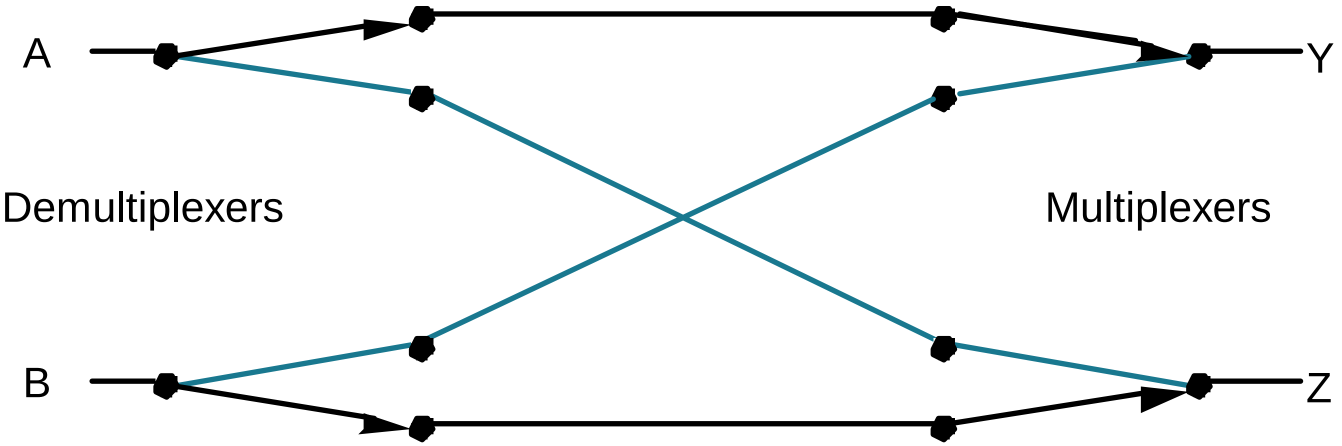
I to Z1 if S = 1



Steering Logic

Use of Multiplexer/Demultiplexer in Digital Systems





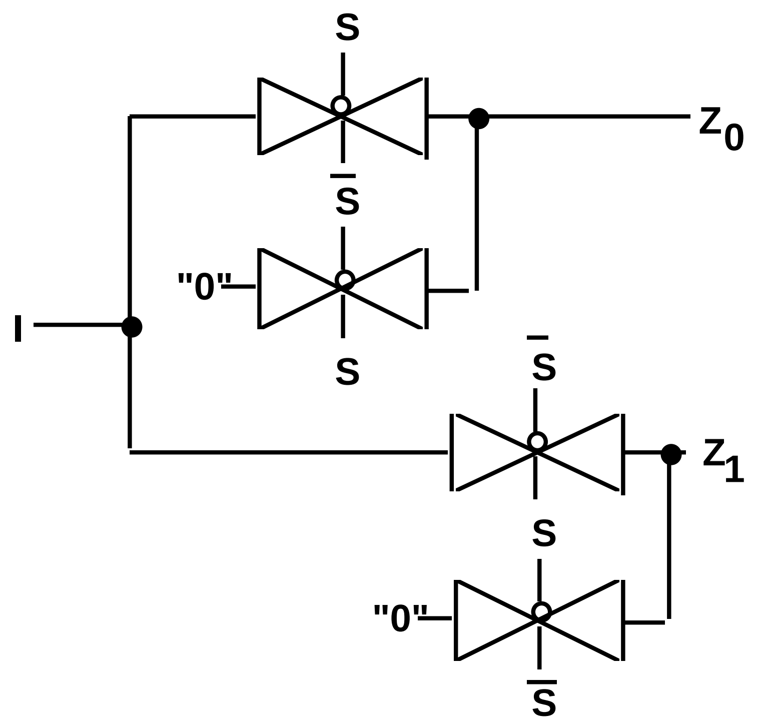
So far, we've only seen point-to-point connections among gates

Mux/Demux used to implement multiple source/multiple destination interconnect

Steering Logic

Well-formed Switching Networks

Problem with the Demux implementation:
multiple outputs, but only one connected to the input!



The fix: additional logic to drive every output to a known value
(steer ‘0’ to Z0 or Z1)
Never allow outputs to "float"

Assignment No. 19

- State steering logic and discuss design of any combinational circuit using steering logic.