COM1006 Devices and Networks (Autumn)
COM1090 Computer Architectures

Lecture #6

▶ Implementing Logic

Dr Dirk Sudholt
Department of Computer Science
University of Sheffield

d.sudholt@sheffield.ac.uk
http://staffwww.dcs.shef.ac.uk/~dirk/campus_only/com1006/

Based on parts of Chapter 2 in Clements, Principles of Computer Hardware

Aims of this lecture

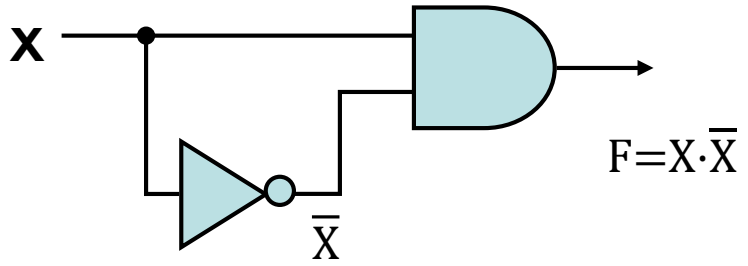
- To discuss various aspects of implementing logic in hardware.
- To introduce the effect of propagation delays on digital circuits
- To illustrate two special-purpose logic elements (the multiplexer and demultiplexer)
- To introduce “don’t care” conditions, and how we can deal with them
- To show how logic elements can be connected to a bus using tri-state buffers

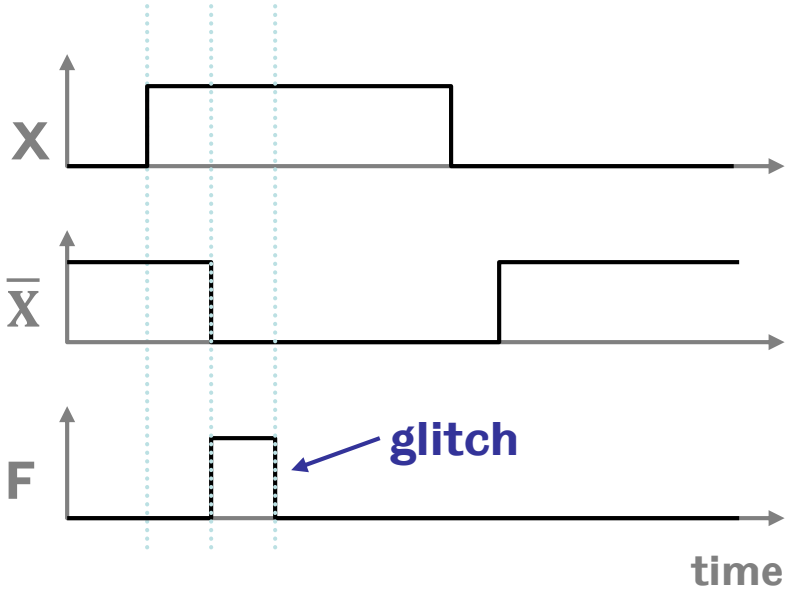
Effect of propagation delays

- We have assumed that signals applied instantaneously to the input terminals of a circuit produce an instantaneous output.
- Real gates suffer a **propagation delay**.
- The propagation delay is the time difference between the arrival of the inputs and the appearance of the required function of the inputs at the outputs of a logical unit.
- These delays are short (approx 1 nsec, or 10^{-9} sec) but can be problematic because they cause **glitches**.
- Removing glitches requires special Boolean logic or using a clock signal to access the output only after the glitch has died away.

Example – glitching

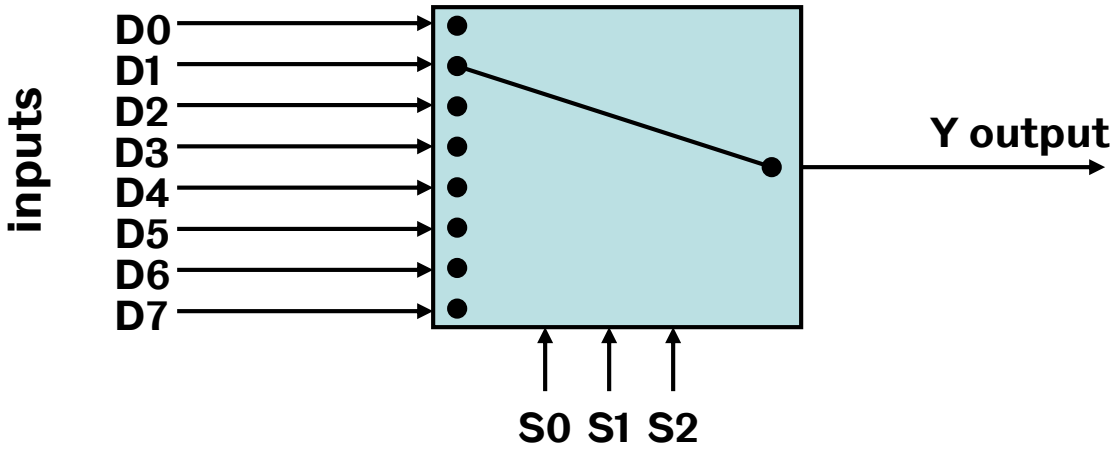
- These are **timing diagrams**.
- The output $F=X\cdot\overline{X}$ should always be 0.
- Due to the delay in the inverter, the inputs to the AND gate are briefly both high.
- After its own delay, the AND gate produces a short glitch in which the output is erroneously 1.





Multiplexer

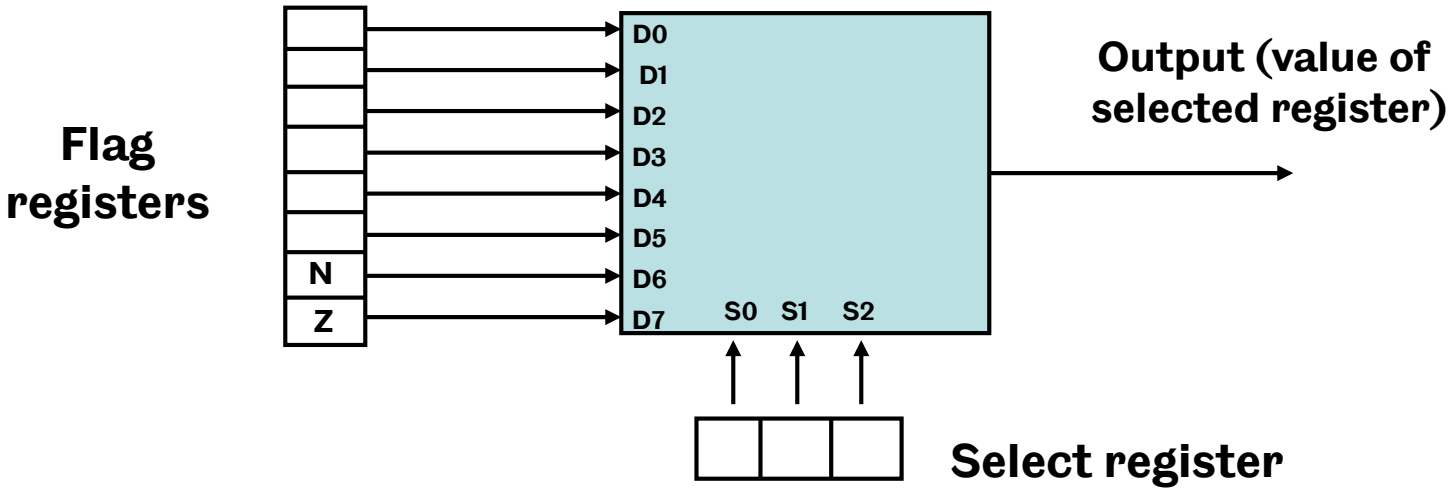
- Recall that a multiplexer is a switch.
- A 1-of-8 multiplexer is shown here, which has 8 data inputs D0-D7, three select inputs S0-S2 and an output Y.



- If the binary value at the data select input is *i*, then the output is given by Y = D*i*

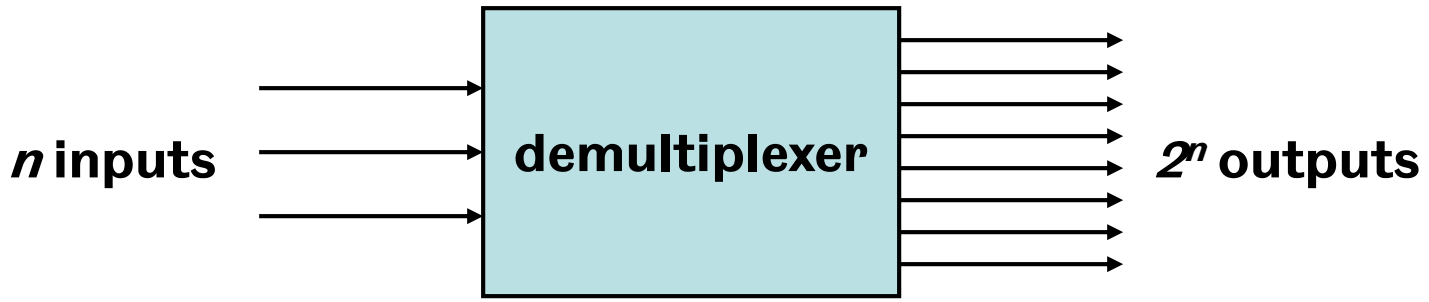
Application of the multiplexer

- Flag registers are used to indicate the state of a computer, e.g. certain bits of the flag register may be set if the result was zero or negative after an arithmetic operation.
- A multiplexer can be used to select the value of one of 8 flag bits according to the value in a 3-bit select register.



Demultiplexer

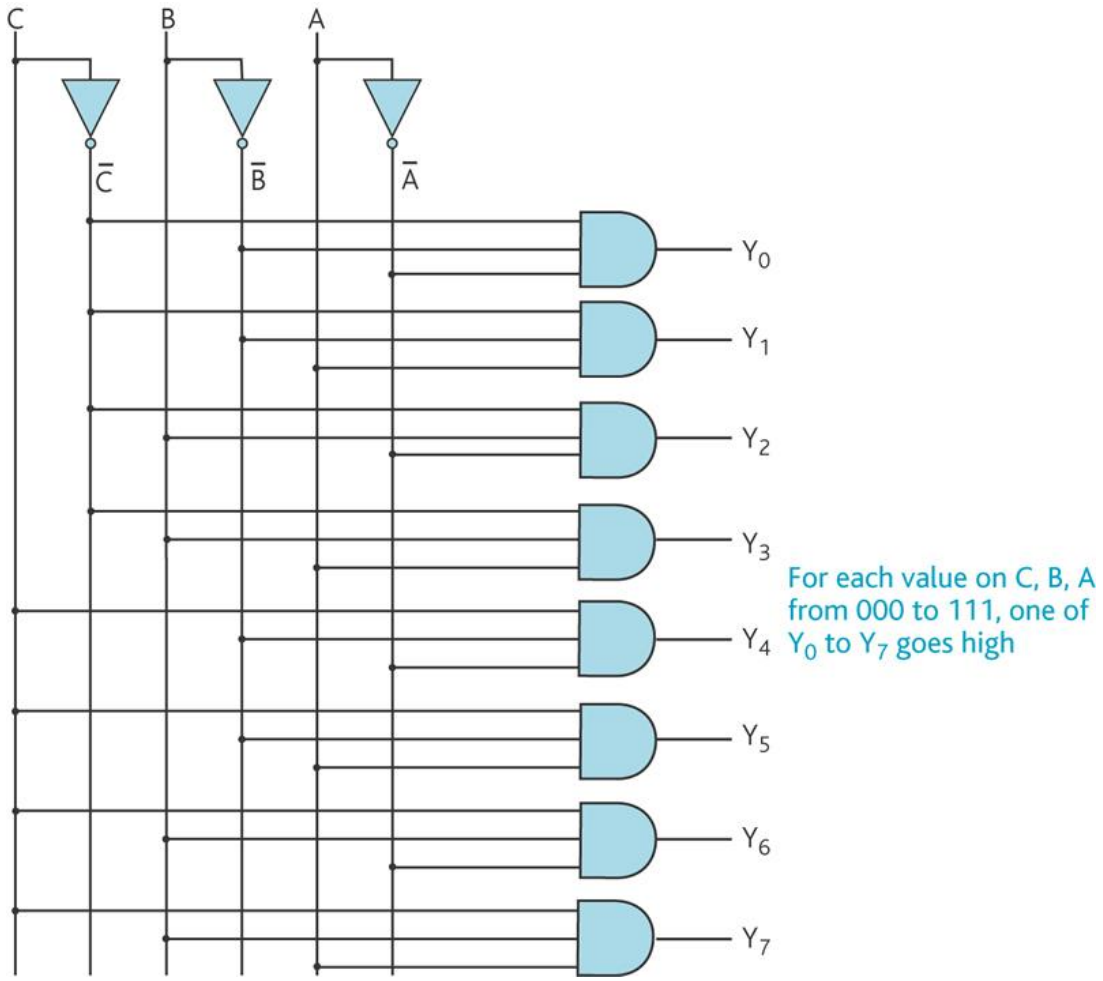
- A demultiplexer performs the inverse operation of a multiplexer.
- Converts a binary code on n inputs into 2^n outputs.
- Only one output is true – which one depends on the inputs.



- The outputs implement all 2^n **minterms**.
- Prefabricated demultiplexers can be useful components for implementing multiple (non-simplified) sums-of-products.

Implementation of a demultiplexer

- Each output is a **minterm**, i.e. an AND of all variables in true or negated form.
- ORing the outputs $Y_0 \ldots Y_7$ allows us to implement sums-of-products.



Application of the demultiplexer

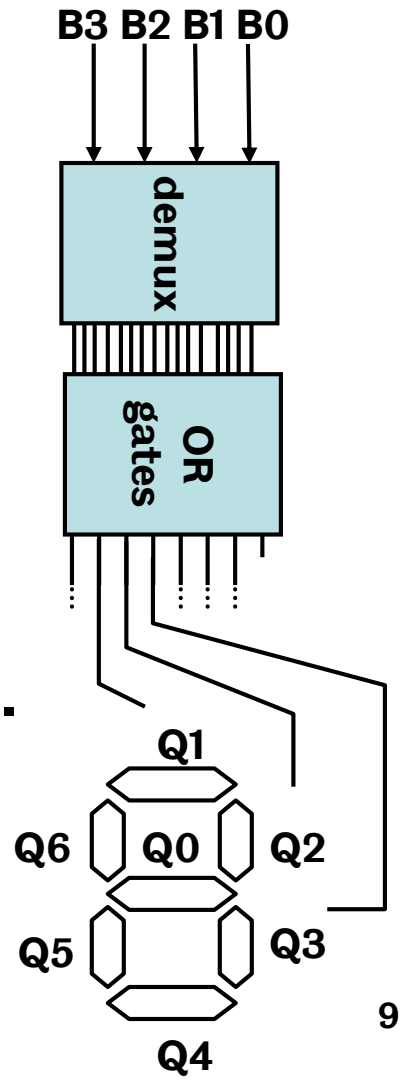
- Can use a demultiplexer to light a 7-segment display.

B3	B2	B1	B0	Q0	Q1	Q2	Q3	Q4	Q5	Q6
0	0	0	0	0	1	1	1	1	1	1
0	0	0	1	0	0	1	1	0	0	0
0	0	1	0	1	1	1	0	1	1	0

- Other important applications in device selection and instruction decoding.

❓ **Complete the remainder of the truth table.**

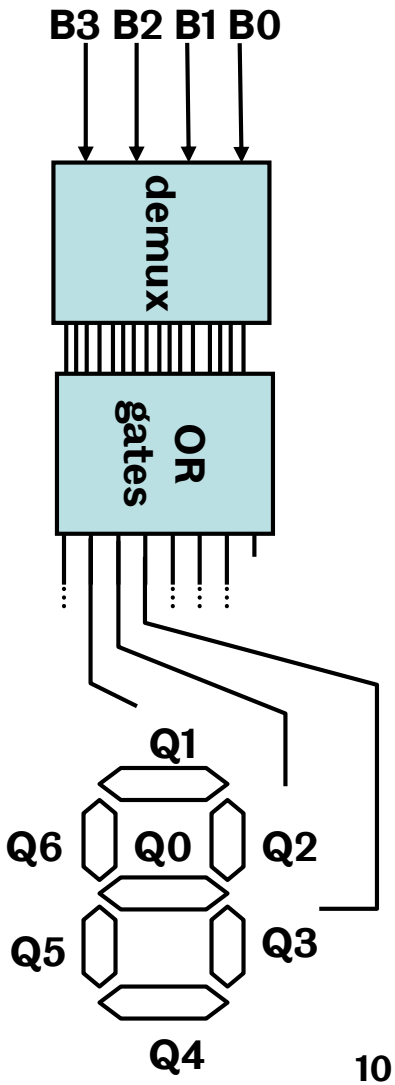
❓ **Advanced: simplify these formulas (Lecture 5) and build your own 7-segment display in logisim out of AND, OR, and NOT gates.**



"Don't care" conditions

For certain inputs the output can be undefined. We **don't care** what the output is.

B3	B2	B1	B0	Q0	Q1	Q2	Q3	Q4	Q5	Q6
0	0	0	0	0	1	1	1	1	1	1
0	0	0	1	0	0	1	1	0	0	0
…	…	…	…	…	…	…	…	…	…	…
1	0	1	0	X	X	X	X	X	X	X
1	0	1	1	X	X	X	X	X	X	X
1	1	0	0	X	X	X	X	X	X	X
1	1	0	1	X	X	X	X	X	X	X
1	1	1	0	X	X	X	X	X	X	X
1	1	1	1	X	X	X	X	X	X	X

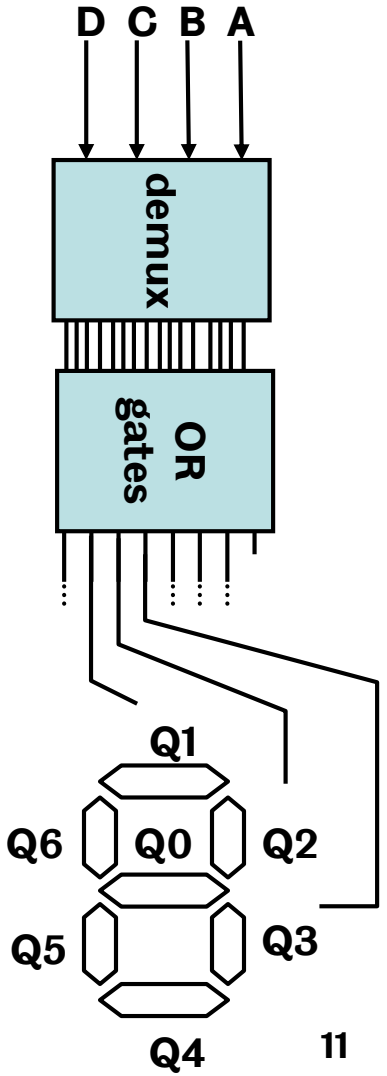


How to deal with don't care conditions

- How do we deal with don't care conditions in circuit design?
- Answer: we can specify "X"s to our advantage!
- Karnaugh maps: "X" cells can be covered or not. Example: Q2

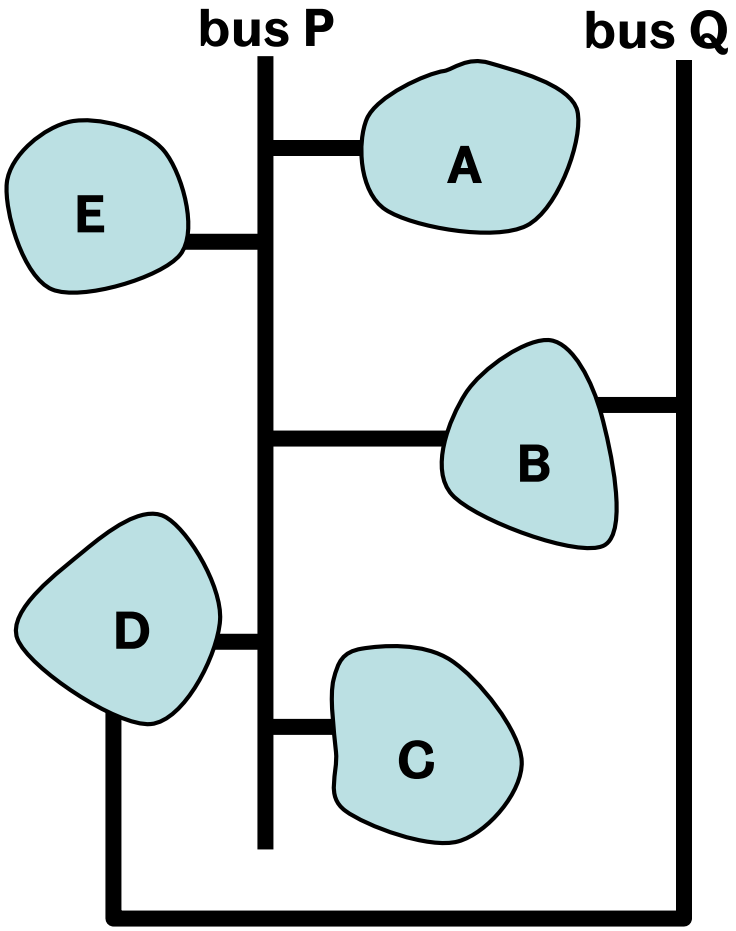
BA \ DC	00	01	11	10
00	1	1	X	1
01	1		X	1
11	1	1	X	X
10	1		X	X

BA \ DC	00	01	11	10
00	1	1	X	1
01	1		X	1
11	1	1	X	X
10	1		X	X

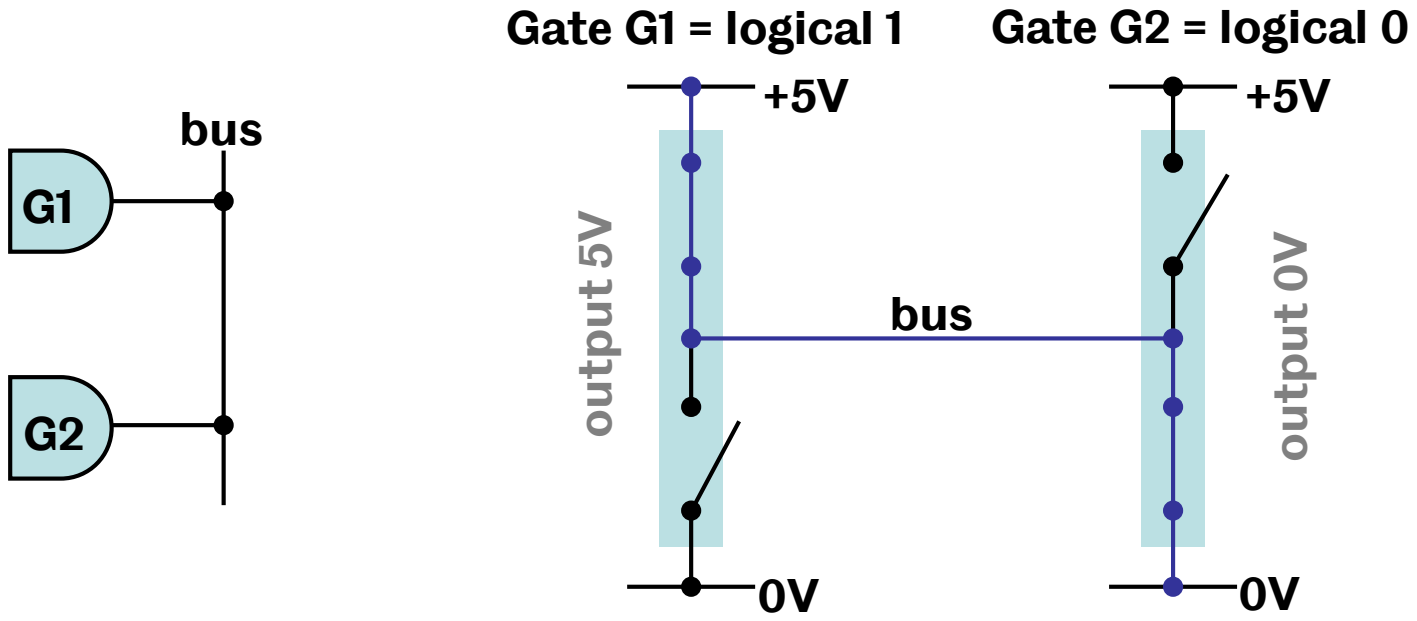


How to link logic elements with buses

- A computer is like a city.
- Just as roads link buildings, **buses** link processing units, storage devices and interfaces.
- Data flows onto a bus from a device connected to it, and off the bus to another device.
- Buses can be unidirectional (data flows one way) or bidirectional (data can flow in two directions, but not simultaneously).

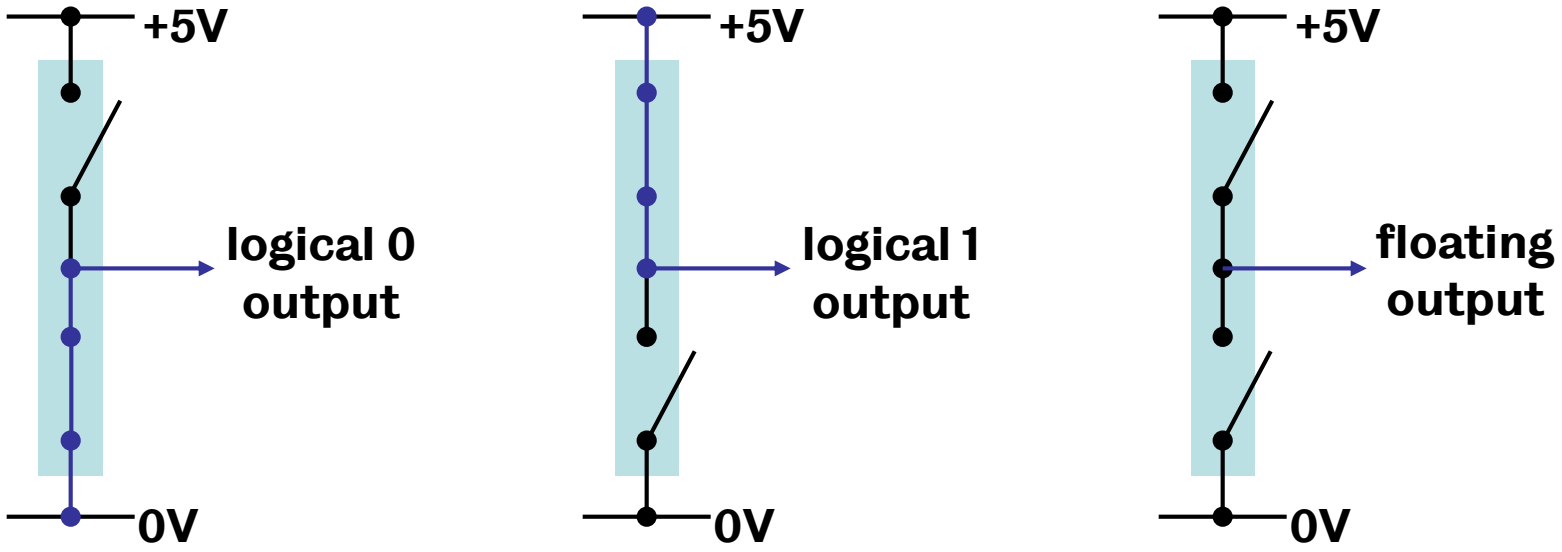


A problem with connecting outputs



- The outputs of two gates cannot simply be connected to a bus.
- If G1=1 and G2=0 there is a philosophical problem (the bus cannot have different logical values at each end) and a practical problem (a short-circuit; current will flow from 5V to ground (0V), possibly destroying the gates).

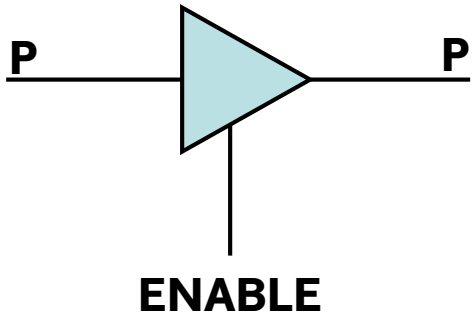
A solution – Tri-state logic



- Outputs can be connected to a bus using tri-state gates.
- Tri-state logic disconnects from the bus all those gates that are not actively engaged in transmitting data.
- When disconnected a tri-state output is “floating” (its value fluctuates with the value on the bus).

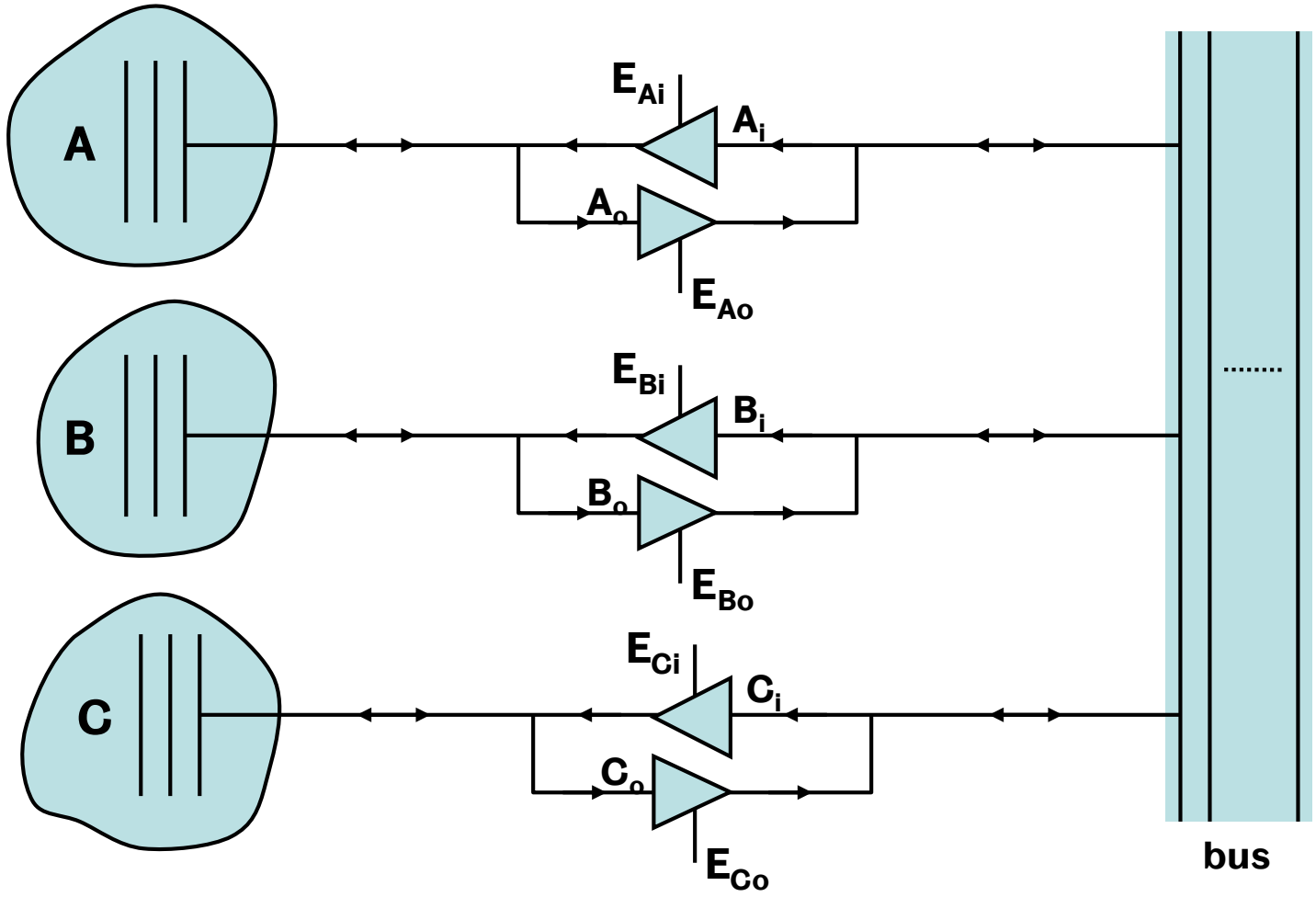
Tri-state buffers

- Any kind of gate can have a tri-state output.
- **Tri-state buffers** are used that simply copy the input to the output but have a special ENABLE input.
- When ENABLE = 1, the gate behaves normally and copies the output to the input.
- When ENABLE = 0, the gate is disconnected.



ENABLE	Input	Output
0	0	floating
0	1	floating
1	0	0
1	1	1

Connecting logic elements to a bus (1)



Connecting logic elements to a bus (2)

- If element A wishes to send data to element C via the bus, the output buffer E_{Ao} is enabled (set to logical 1) and the input buffer E_{Ci} is enabled.
- All other inputs and outputs are disabled (logical 0).
- It is essential that:
  - no more than one of the output buffers E_{Ao}, E_{Bo}, E_{Co} are enabled at the same time
  - no more than one of the input buffers E_{Ai}, E_{Bi}, E_{Ci} are enabled at the same time
- Note that the figure only shows connections to one line of the bus for clarity.

Summary

- Implementation of digital circuits is complicated by the issue of propagation delays
- Special-purpose logic elements such as the multiplexer and demultiplexer are frequently used
- We can use don't care conditions to our advantage, covering or not covering X cells in Karnaugh maps to yield the simplest expression.
- Tri-state logic is necessary in order to connect logic elements to a bus