Real Time System

Third Level

Lecture Two

Input and Output Interfacing

Dr. Hussein AbdulAmeer

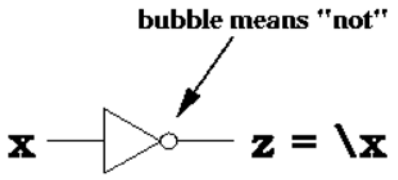
Hussein.Alkhamees@uomus.edu.iq

1. Input interface:-

The sources of binary information (0, 1) are usually the output of logic gates. It could also be provided by some other circuits such as switches. One of the input interface devices is the three states gate which is usually used to input data from devices to controlling system.

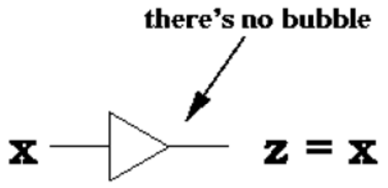
Tri-state buffer

Before we talk about tri-state buffers, let's talk about an inverter. An inverter is called a NOT gate, and it looks like:



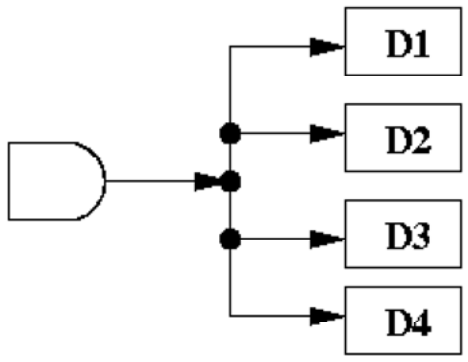
The inverter is a triangle, followed by a circle/bubble. That circle means negation.

What if we remove the circle? What kind of gate would we have? We'd have a buffer.



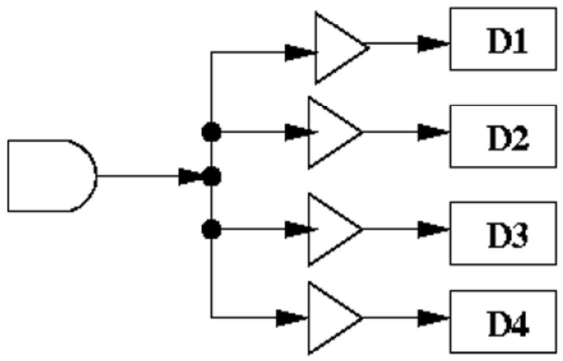
You might think that a buffer is useless. After all, the output is exactly the same as the input. What's the point of such a gate? The answer is a practical issue from real circuits. As you may know, logic gates process 0's and 1's. 0's and 1's are really electric current at certain voltages. If there isn't enough current, it's hard to measure the voltage.

The current can decrease if the fan out is large. Here's an example:



The "fan out" is the number of devices that an output is attached to. Thus, the AND gate above is attached to the inputs of four other devices. It has a fan out of 4. If the current coming out of the AND gate is I, then assuming each of the four devices gets equal current, then each device gets I / 4 of the current.

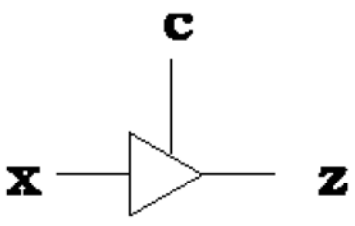
However, if we put in a buffer:



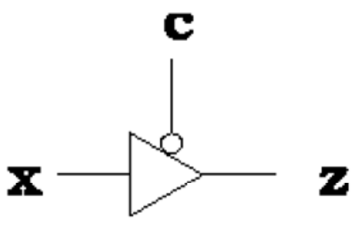
Then the current can be "boosted" back to the original strength. Thus, a buffer (like all logic gates) is an active device. It requires additional inputs to power the gate, and provide it voltage and current.

Tri-state buffer: It's a Valve

A buffer's output is defined as z = x. Thus, if the input, x is 0, the output, z is 0. If the input, x is 1, the output, z is 1. A tri-state buffer is a useful device that allows us to control when current passes through the device, and when it doesn't. Here's two diagrams of the tri-state buffer.



tri–state buffer with
active high control



tri–state buffer with
active low control

A tri-state buffer has two inputs: a data input x and a control input c. The control input acts like a valve. When the control input is active, the output is the input. That is, it behaves just like a normal buffer. The "valve" is open.

When the control input is not active, the output is "Z". The "valve" is open, and no electrical current flows through. Thus, even if x is 0 or 1, that value does not flow through. Here's a truth table describing the behavior of a active-high tri-state buffer.

c	x	z
0	0	Z
0	1	Z
1	0	0
1	1	1

In this case, when the output is Z, that means its high impedance, neither 0, nor 1, i.e., no current.

c	z
0	Z
1	x

As you can see, when c = 1 the valve is open, and z = x. When c = 0 the valve is closed, and z = Z (e.g., high impedance/no current).

Active-low tri-state buffers:

Some tri-state buffers are active low. In an active-low tri-state buffer, c = 0 turns open the valve, while c = 1 turns it off.

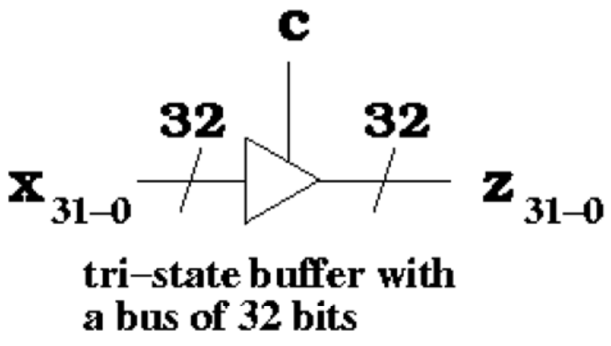
Here's the condensed truth table for an active-low tri-state buffer.

c	z
0	x
1	Z

As you can see, when c = 0 the valve is open, and z = x. When c = 1 the valve is closed, and z = Z (e.g., high impedance/no current). Thus, it has the opposite behavior of a tri-state buffer.

Multi-bit Tri-State Buffers

So far, we've talked about a tri-state buffer controlling the output of a single wire or bit. However, it's more common to deal with many wires. Here's an example:

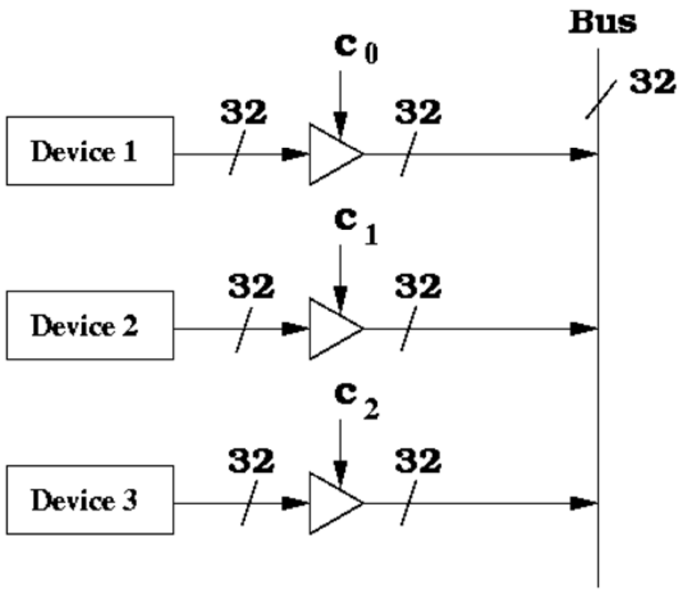


In this case, we have 32 wires coming into the tri-state buffer. We have 32 wires coming out of the tri-state buffer. There's still only 1 control bit.

This can easily be implemented using 32 tri-state buffers taking one bit as input and one bit as output.

Why Tri-State Buffers?

We've had a long discussion about what a tri-state buffer is, but not about what such a device is good for. That a common way for many devices to communicate with one another is on a bus, and that a bus should only have one device writing to it, although it can have many devices reading from it. Since many devices always produce output (such as registers) and these devices are hooked to a bus, we need a way to control what gets on the bus, and what doesn't. A tri state buffer is good for that. Here's an example:

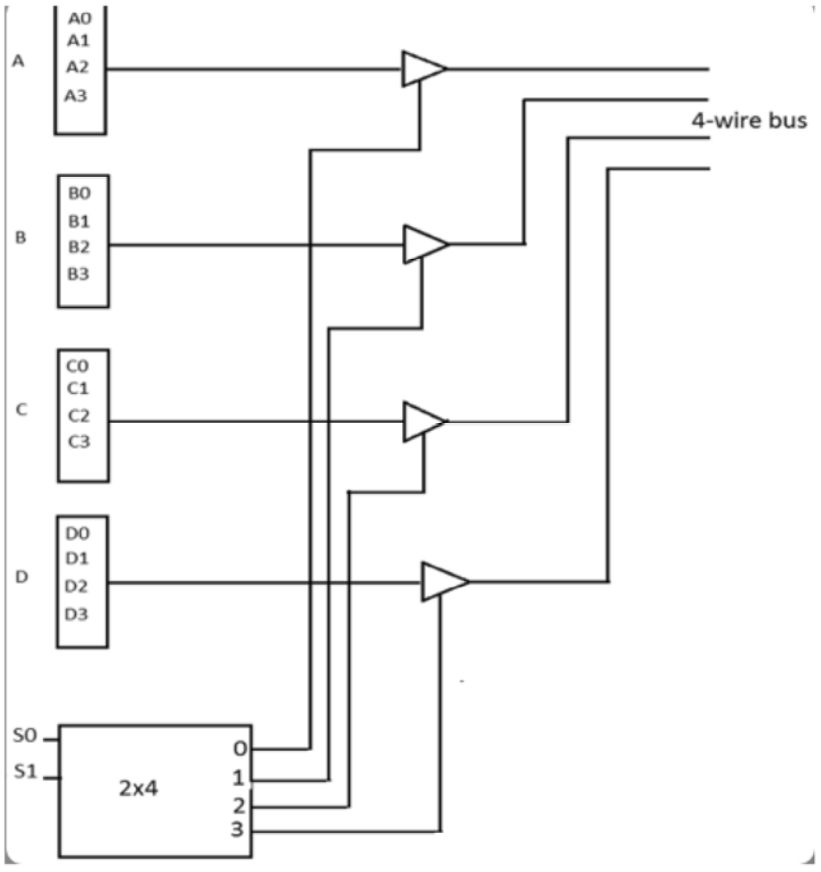


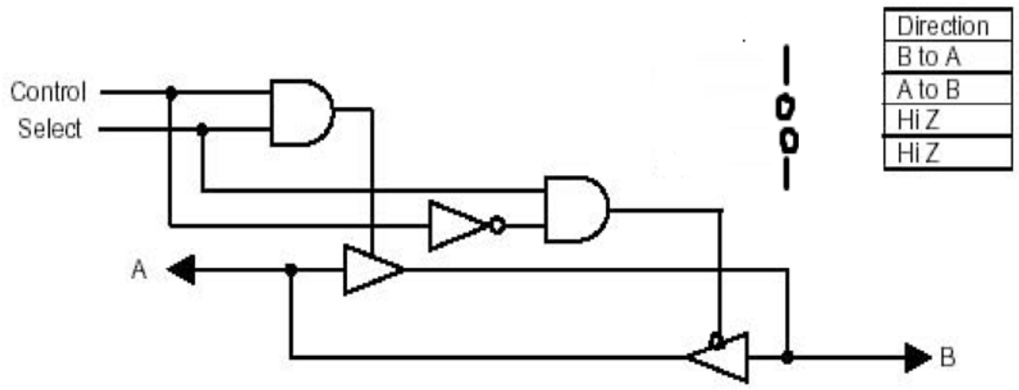
There are three devices, each of which output 32 bits. These devices have their outputs hooked to a 32 bit bus.

We want to prevent more than one device from writing to the bus. Ordinarily, these devices always generate output, so we're in trouble merely by attaching more than one device's output to the bus.

As long as at most one of the following control bits, c0, c1, c2, is 1, the bus is fine. That is, the bus will not have two devices attempting to write to it at the same time.

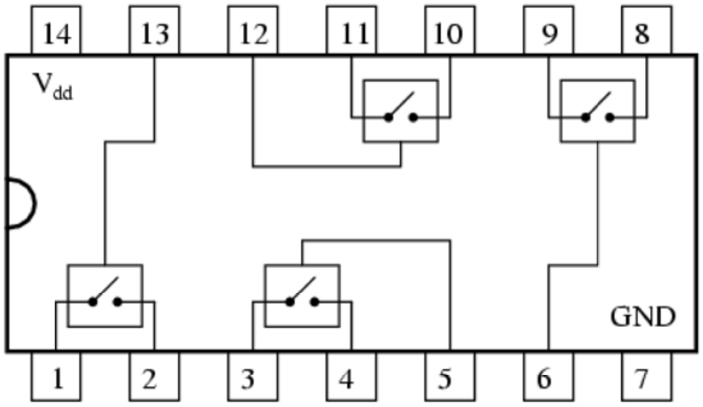
<u>**Examples**</u>





Direction
B to A
A to B
Hi Z
Hi Z

This type of circuit is used inside memory circuits to control the input/output lines.



Summary

A tri-state buffer is a device that allows you to control when an output signal makes it to the bus. When the tri-state buffer's control bit is active, the input of the device makes it to the output. This is when the "valve" is open.

When it's not active, the output of the device is Z, which is high-impedance or, equivalently, nothing. This is when the "valve" is closed, and no electrical signal is allowed to pass to the output.

2. <u>Output interfacing:-</u>

This type of interfacing is used to output the data from data bus to the external devices connected to the controlling system. One of the interfacing components is the D-flip flop.

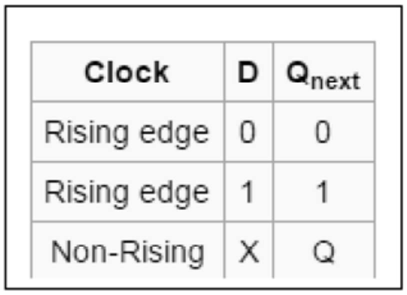
<u>D-flip flop</u>

In electronics, a flip-flop is a circuit that has two stable states and can be used to store state information. The circuit can be made to change state by signals applied to one or more control inputs and will have one or two outputs. Flip-flops are a fundamental building block of digital electronics systems used in computers, communications, and many other types of systems.

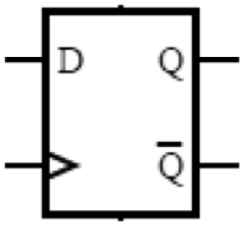
Flip-flops and latches are used as data storage elements. A flip-flop stores a single bit (binary digit) of data; one of its two states represents a "one" and the other represents a "zero". When used some systems, the output and next state depend not only on its current input, but also on its current state (and hence, previous inputs).

Flip-flops can be divided into common types: the SR ("set-reset"), D ("data" or "delay"), T ("toggle"), and JK types are the common ones.

The D flip-flop is widely used. It is also known as a "data" or "delay" flip-flop. The D flip-flop captures the value of the D-input at a definite portion of the clock cycle (such as the rising edge of the clock). That captured value becomes the Q output. At other times, the output Q does not change. The D flip-flop can be viewed as a memory cell.



Clock	D	Q_{next}
Rising edge	0	0
Rising edge	1	1
Non-Rising	X	Q



The data bus lines connected the D-input and the Q-output is connected to the destination. The controller generates a control signal applied at the CLK of the flip flop to latch data from D to Q.

3. <u>Multiple Source and Destination:-</u>

Buffers and latches are used to construct input and output interfacing devices. The latches such as 74LS374 are used to latch and hold data for input data to the controller and output to the external devices. The other job of the latch is to protect the controller in case of any short might happen to the external devices which might destroy the controller circuitry.

Two types of buffers can be used such as the unidirectional 74LS244 and the bidirectional 74LS245 can be used in interfacing circuitry to input and output data from CPU and external devices.

What is the difference between buffer and latch?

What is the difference latch and flip flop?

A buffer is a device whose output will always follow its input. Buffers pass an input through to output after some time, possibly increasing drive strength.

A latch's output will follow its input only while the latch enable is active; when the enable goes inactive the latch will hold the last value it had.

Real Time System

Third Level

Lecture Eight

I/O Interfacing and Programmable Devices