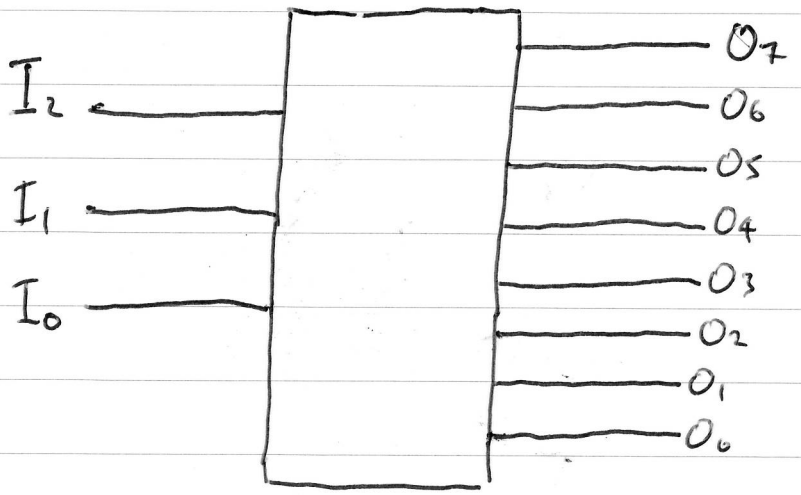
3 TO 8 DECODER (TRUTH TABLE)

I_2	I_1	I_0	O_7	O_6	O_5	O_4	O_3	O_2	O_1	O_0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

$O_0 = \bar{I}_2 \cdot \bar{I}_1 \cdot$
$O_1 = \bar{I}_2 \cdot \bar{I}_1 \cdot$
$O_2 = \bar{I}_2 \cdot I_1 \cdot$
$O_3 = \bar{I}_2 \cdot I_1 \cdot$
$O_4 = I_2 \cdot \bar{I}_1 \cdot$
$O_5 = I_2 \cdot \bar{I}_1 \cdot$
$O_6 = I_2 \cdot I_1 \cdot$
$O_7 = I_2 \cdot I_1 \cdot$

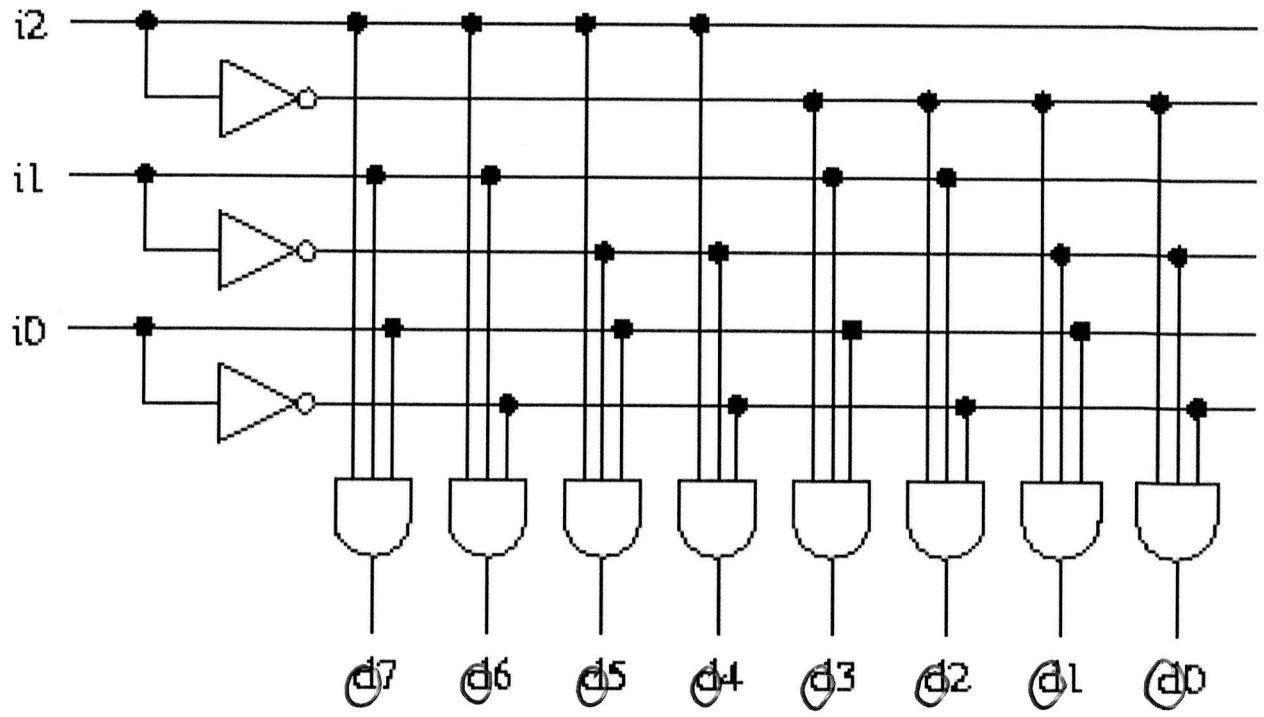
BLOCK DIAGRAM FOR 3-8 DECODER



DEPENDING ON THE INPUTS, THE DECODER SELECTS, PUTS A HIGH VOLTAGE OUT ON ONLY ONE OF THE OUTPUT LINES

E.G. FIRST LINE OF TRUTH TABLE ONLY LINE O_0 HAS A HIGH VOLTAGE. ALL OTHER OUTPUT LINES ARE LOW.

3 TO 8 DECODER (GATE LEVEL DIAGRAM)



TRI-STATE BUFFER

FIRST WE WILL LOOK AT A **BUFFER**

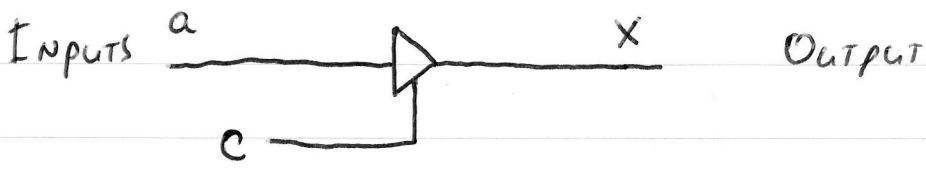
SYMBOL FOR BUFFER

BUFFER CAN BE USED TO AMPLIFY THE SIGNAL SO THAT IT CAN BE DISTRIBUTED TO MORE GATES THAN THE ORIGINAL SIGNAL COULD SUPPLY.

TRI-STATE BUFFER

IS A CIRCUIT, SIMILAR TO THE BUFFER EXCEPT THAT IT HAS AN EXTRA INPUT, CALLED THE CONTROL INPUT, C IN THE NEXT DIAGRAM.

SYMBOL FOR TRI-STATE BUFFER



(2)

WHEN C=1 THE GATE ACTS LIKE A BUFFER

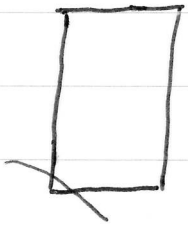
WHEN C=0 THE GATE OUTPUT GOES TO A HIGH IMPEDENCE STATE (HIGH Z IN TABLE BELOW).

TRUTH TABLE FOR TRI-STATE BUFFER

c	a	x	
0	0	HIGH Z	NOTHING COMING OUT
0	1	HIGH Z	NOTHING COMING OUT
1	0	0	LOW VOLTAGE
1	1	1	HIGH VOLTAGE

HIGH Z MEANS HIGH RESISTANCE

NOTHING COMING OUT ON THE OUTPUT LINE X.

(1) 

FLOODS

DOOR SHUT

SOME WATER WILL GET THROUGH

(2) 

DOOR OPEN

MORE WATER WILL POUR THROUGH.

(3) 

DOOR SHUT

SEALED WITH SILICONE : HIGH RESETANCE TO WATER OUTSIDE

SANDBAGS NOTHING GETS THROUGH

http://users.ece.utexas.edu/~valvano/Volume1/E-Book/C4_DigitalLogic.htm

Clearly the logic gate circuitry for a **4 × 1** and an **8 × 1** multiplexer get progressively more complicated.

(DL.28) Uses of the multiplexer

Multiplexers are mainly used in PCs to allow different components use the same "wires" to communicate with the CPU (Central Processing Unit). For example the "wires" used to pass data from the CPU to the modem, the network card and the sound card are called the **data bus**. In order for different components to use the same **data bus** a multiplexer must be used.

Multiplexers are also used in networking to allow, for example, more than one PC to access the same telephone line for use of the internet, etc.

(DL.29) An example of multiplexers to connect two 4-bit registers to a single bus

A 4-bit register is simply an area of computer memory which can store 4 binary digits (i.e. 4 bits). If two of these registers are to be connected to a 4 line bus which reads all four digits from one of the registers only, then it could be constructed using four **2 × 1 multiplexers** as shown below:

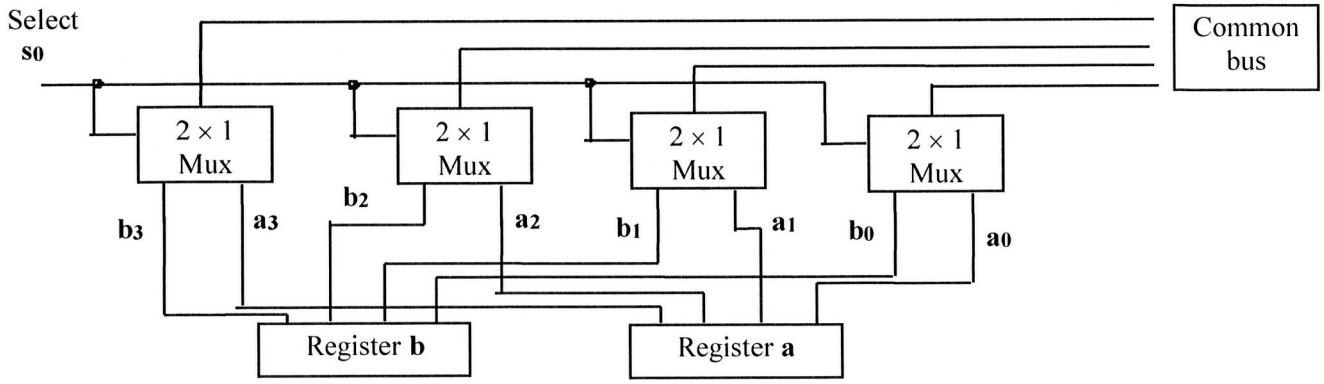


Diagram (DL.30)

In this diagram, if the select input $s_0 = 0$, then the bus is connected to the four digits in register **a** only i.e. values a_3 a_2 a_1 and a_0 are passed to the bus. Similarly if $s_0 = 1$, then the bus is connected to register **b** only.

This set up has the very simple function table :

s_0	bus
0	**a**
1	**b**

Diagram (DL.30) above is often simplified to the following diagram:

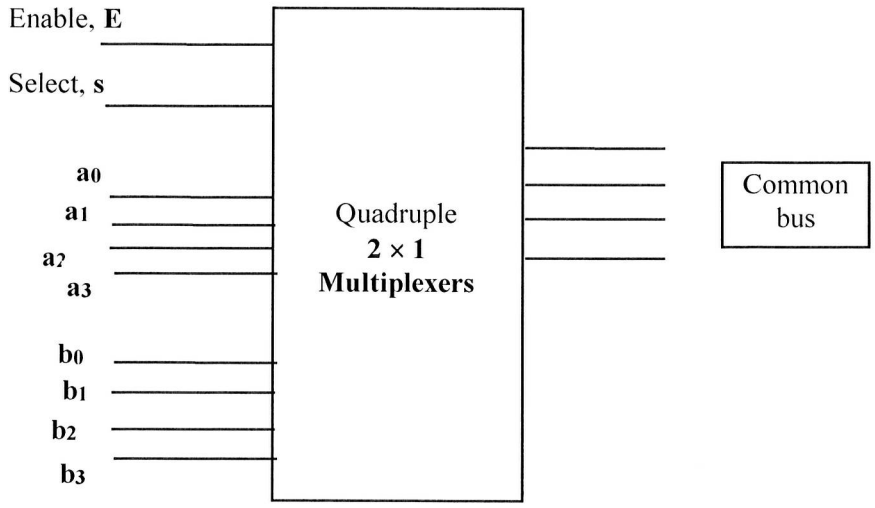


Diagram (DL.31)

Notice that this diagram has an extra input called **enable,E**. This will simply turn on and off the whole array of multiplexers. If **E = 0** then all outputs are disabled i.e. they are all zero. When the enable switch is active i.e. **E = 1** then the component operates normally as a group of four multiplexers.

In practice, multiplexers will have an enable input. The function table for **diagram (DL.31)** is :

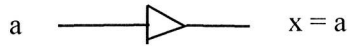
E	**s0**	bus
0	0	All 0
0	1	All 0
1	0	**a**
1	1	**b**

The buses used on the Pentium II are 32-bit (address bus) and 64-bit (external data bus), so even to allow two external components to comunicate with the CPU would require 130 input lines and 64 output bus lines. The multiplexer could be designed by use of a group of 64 of the **2 × 1** multiplexers described above.

(DL.32) Tri-state buffer or three-state buffer

The simplest logic gate used is the **buffer** or **amplifier** or **delay**

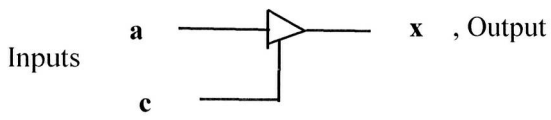
Symbol for **buffer**



The buffer is sometimes used to delay a signal, this is possible since there is always a time difference between the moment when the input is applied to the gate and the moment when the output appears (about 20 nanoseconds, ns). It can be also used to amplify the signal so that it can be distributed to more components that the original signal could supply.

The tri-state buffer is a digital circuit, similar to the buffer above, except that it has an extra input, called the **control input, c** in the diagram below. When **c = 1** the gate acts like a normal buffer. However when **c = 0** the gate goes to a high impedance state (high z , in table below), which essentially disconnects the output.

Symbol for **tri-state buffer**

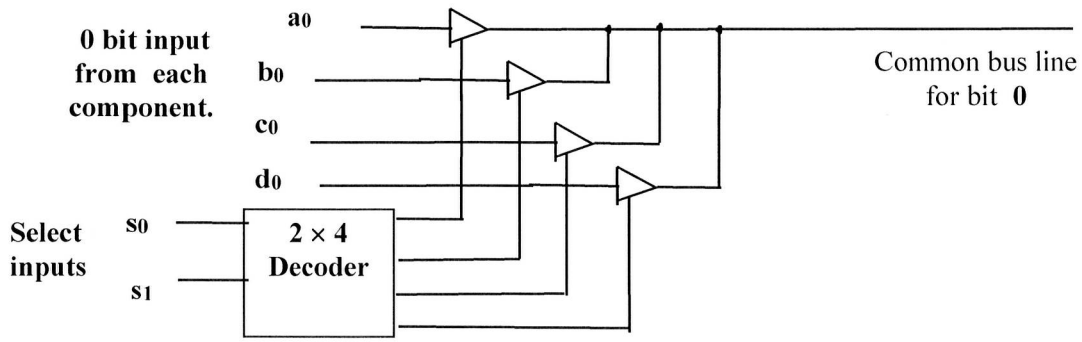


Truth Table

c	a	x
0	0	High z
0	1	High z
1	0	**0**
1	1	**1**

(DL.33) Tri-state buffers used with decoders to create a bus system

Tri-state buffers can be used in conjuction with a decoder to connect different components to a single bus line. For example, the **4 x 1 mux** in **DL.25** above could be designed as follows to connect four components to one bus :



The example in **Diagram (DL.30)** could be designed, as in the diagram below, using a single decoder and tri-state buffers to connect two registers to a single bus line :

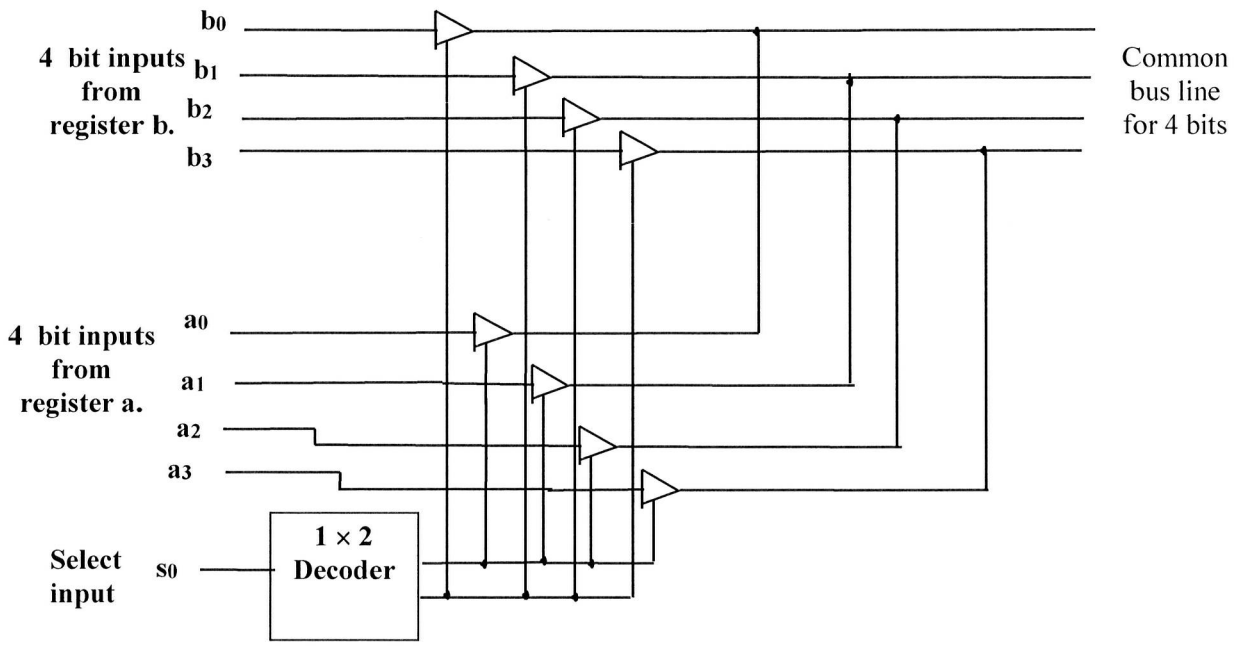


Diagram (DL.34)

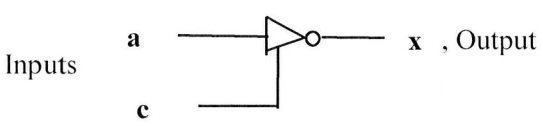
Diagram (DL.34) is a cheaper method of creating a bus system, since only one decoder is needed here, whereas in **diagram (DL.30)** there were **four** multiplexers used.

If the common bus system had 32 lines, then the multiplexer system would need 32 multiplexers, but the decoder and tri-state buffer system still only uses one decoder.

(DL.35) Inverted tri-state buffers

For completeness we include the symbol and truth table for the **inverted tri-state buffer**, which is the same as the standard tri-state buffer above with the output inverted.

Symbol for **inverted tri-state**



Truth Table

c	a	x
0	0	High z
0	1	High z
1	0	**1**
1	1	**0**