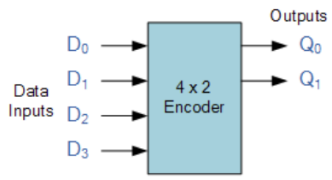


Inputs				Outputs	
D_3	D_2	D_1	D_0	Q_1	Q_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1
0	0	0	0	x	x

Priority Encoder

Priority Encoders take all of their data inputs one at a time and converts them into an equivalent binary code at its output

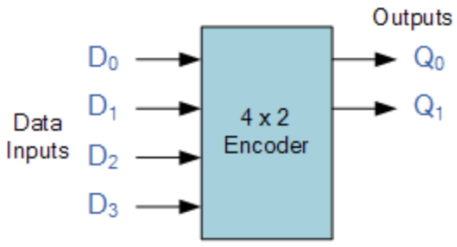
Unlike a multiplexer that selects one individual data input line and then sends that data to a single output line or switch. The job of a priority encoder is to produce a binary output address for the input with the highest priority.

The **Digital Encoder** more commonly called a **Binary Encoder** takes ALL its data inputs one at a time and then converts them into a single encoded output. So we can say that a binary encoder, is a multi-input combinational logic circuit that converts the logic level "1" data at its inputs into an equivalent binary code at its output.

Generally, digital encoders produce outputs of 2-bit, 3-bit or 4-bit codes depending upon the number of data input lines. An "n-bit" binary encoder has 2^n input lines and n-bit output lines with common types that include 4-to-2, 8-to-3 and 16-to-4 line configurations.

The output lines of a digital encoder generate the binary equivalent of the input line whose value is equal to "1" and are available to encode either a decimal or hexadecimal input pattern to typically a binary or "B.C.D" (binary coded decimal) output code.

4-to-2 Bit Binary Encoder



Inputs				Outputs	
D_3	D_2	D_1	D_0	Q_1	Q_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1
0	0	0	0	x	x

One of the main disadvantages of standard digital encoders is that they can generate the wrong output code when there is more than one input present at logic level "1". For example, if we make inputs D_1 and D_2 HIGH at logic "1" both at the same time, the resulting output is neither at "01" or at "10" but will be at "11" which is an output binary number that is different to the actual input present. Also, an output code of all logic "0"s can be generated when all of its inputs are at "0" OR when input D_0 is equal to one.

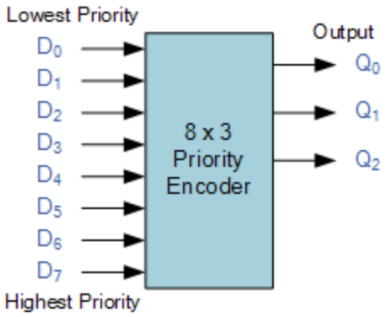
One simple way to overcome this problem is to "Prioritise" the level of each input pin. So if there is more than one input at logic level "1" at the same time, the actual output code would only correspond to the input with the highest designated priority. Then this type of digital encoder is known commonly as a **Priority Encoder** or **P-encoder** for short.

Priority Encoder

The **Priority Encoder** solves the problems mentioned above by allocating a priority level to each input. The *priority encoders* output corresponds to the currently active input which has the highest priority. So when an input with a higher priority is present, all other inputs with a lower priority will be ignored.

The priority encoder comes in many different forms with an example of an 8-input priority encoder along with its truth table shown below.

8-to-3 Bit Priority Encoder



Inputs								Outputs		
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Q_2	Q_1	Q_0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	x	0	0	1
0	0	0	0	0	1	x	x	0	1	0
0	0	0	0	1	x	x	x	0	1	1
0	0	0	1	x	x	x	x	1	0	0
0	0	1	x	x	x	x	x	1	0	1
0	1	x	x	x	x	x	x	1	1	0
1	x	x	x	x	x	x	x	1	1	1

X = dont care

Priority encoders are available in standard IC form and the TTL 74LS148 is an 8-to-3 bit priority encoder which has eight active LOW (logic "0") inputs and provides a 3-bit code of the highest ranked input at its output.

Priority encoders output the highest order input first for example, if input lines "D2", "D3" and "D5" are applied simultaneously the output code would be for input "D5" ("101") as this has the highest order out of the 3 inputs. Once input "D5" had been removed the next highest output code would be for input "D3" ("011"), and so on.

The truth table for a 8-to-3 bit priority encoder is given as:

Digital Inputs								Binary Output		
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Q_2	Q_1	Q_0
0	0	0	0	0	0	0	**1**	0	0	0
0	0	0	0	0	0	**1**	X	0	0	1
0	0	0	0	0	**1**	X	X	0	1	0
0	0	0	0	**1**	X	X	X	0	1	1
0	0	0	**1**	X	X	X	X	1	0	0
0	0	**1**	X	X	X	X	X	1	0	1
0	**1**	X	X	X	X	X	X	1	1	0
1	X	X	X	X	X	X	X	1	1	1

Where X equals "dont care", that is it can be at a logic "0" level or at a logic "1" level.

From this truth table, the Boolean expression for the encoder above with data inputs D_0 to D_7 and outputs Q_0, Q_1, Q_2 is given as:

Output Q_0

$$Q_0 = \sum(1, 3, 5, 7)$$
$$Q_0 = \sum\left(\bar{D}_7\bar{D}_6\bar{D}_5\bar{D}_4\bar{D}_3\bar{D}_2D_1 + \bar{D}_7\bar{D}_6\bar{D}_5\bar{D}_4D_3 + \bar{D}_7\bar{D}_6D_5 + D_7\right)$$
$$Q_0 = \sum\left(\bar{D}_6\bar{D}_4\bar{D}_2D_1 + \bar{D}_6\bar{D}_4D_3 + \bar{D}_6D_5 + D_7\right)$$
$$Q_0 = \sum\left(\bar{D}_6\left(\bar{D}_4\bar{D}_2D_1 + \bar{D}_4D_3 + D_5\right) + D_7\right)$$

Output Q_1

$$Q_1 = \sum(2, 3, 6, 7)$$
$$Q_1 = \sum\left(\bar{D}_7\bar{D}_6\bar{D}_5\bar{D}_4\bar{D}_3D_2 + \bar{D}_7\bar{D}_6\bar{D}_5\bar{D}_4D_3 + \bar{D}_7D_6 + D_7\right)$$
$$Q_1 = \sum\left(\bar{D}_5\bar{D}_4D_2 + \bar{D}_5\bar{D}_4D_3 + D_6 + D_7\right)$$
$$Q_1 = \sum\left(\bar{D}_5\bar{D}_4(D_2 + D_3) + D_6 + D_7\right)$$

Output Q_2

$$Q_2 = \sum(4, 5, 6, 7)$$
$$Q_2 = \sum\left(\bar{D}_7\bar{D}_6\bar{D}_5D_4 + \bar{D}_7\bar{D}_6D_5 + \bar{D}_7D_6 + D_7\right)$$
$$Q_2 = \sum(D_4 + D_5 + D_6 + D_7)$$

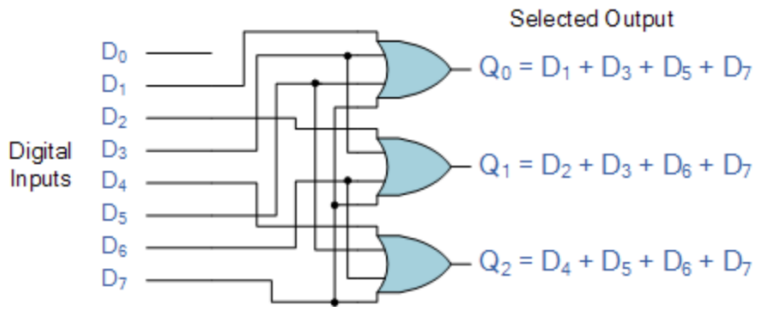
Then the final Boolean expression for the priority encoder including the zero inputs is defined as:

Priority Encoder Output Expression

$$Q_0 = \sum\left(\bar{D}_6\left(\bar{D}_4\bar{D}_2D_1 + \bar{D}_4D_3 + D_5\right) + D_7\right)$$
$$Q_1 = \sum\left(\bar{D}_5\bar{D}_4(D_2 + D_3) + D_6 + D_7\right)$$
$$Q_2 = \sum(D_4 + D_5 + D_6 + D_7)$$

In practice these zero inputs would be ignored allowing the implementation of the final Boolean expression for the outputs of the 8-to-3 **priority encoder**. We can constructed a simple encoder from the expression above using individual OR gates as follows.

Digital Encoder using Logic Gates



Digital Encoder Applications

Keyboard Encoder

Priority encoders can be used to reduce the number of wires needed in a particular circuits or application that have multiple inputs. For example, assume that a microcomputer needs to read the 104 keys of a standard QWERTY keyboard where only one key would be pressed either “HIGH” or “LOW” at any one time.

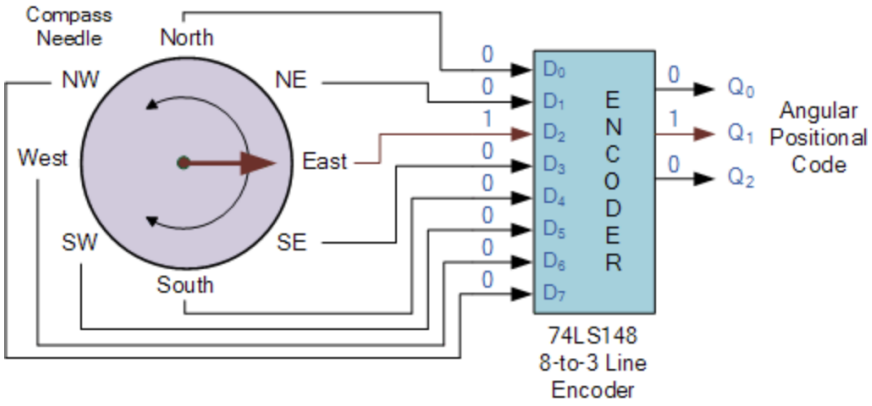
One way would be to connect all 104 wires from the individual keys on the keyboard directly to the computers input but this would be impractical for a small home PC. Another alternative and better way would be to interface the keyboard to the PC using a priority encoder.

The 104 individual buttons or keys could be encoded into a standard ASCII code of only 7-bits (0 to 127 decimal) to represent each key or character of the keyboard and then input as a much smaller 7-bit B.C.D code directly to the computer. Keypad encoders such as the 74C923 20-key encoder are available to do just that.

Positional Encoders

Another more common application is in magnetic positional control as used on ships navigation or for robotic arm positioning etc. Here for example, the angular or rotary position of a compass is converted into a digital code by a 74LS148 8-to-3 line priority encoder and input to the systems computer to provide navigational data and an example of a simple 8 position to 3-bit output compass encoder is shown below. Magnets and reed switches could be used at each compass point to indicate the needles angular position.

Priority Encoder Navigation



Compass Direction	Binary Output		
	Q_0	Q_1	Q_2
North	0	0	0
North-East	0	0	1
East	0	1	0
South-East	0	1	1
South	1	0	0
South-West	1	0	1
West	1	1	0
North-West	1	1	1

Interrupt Requests

Other applications especially for **Priority Encoders** may include detecting interrupts in microprocessor applications. Here the microprocessor uses interrupts to allow peripheral devices such as the disk drive, scanner, mouse, or printer etc, to communicate with it, but the microprocessor can only "talk" to one peripheral device at a time so needs some way of knowing when a particular peripheral device wants to communicate with it.

The processor does this by using "Interrupt Requests" or "IRQ" signals to assign priority to all the peripheral devices to ensure that the most important peripheral device is serviced first. The order of importance of the devices will depend upon their connection to the priority encoder.

IRQ Number	Typical Use	Description
IRQ 0	System timer	Internal System Timer.
IRQ 1	Keyboard	Keyboard Controller.
IRQ 3	COM2 & COM4	Second and Fourth Serial Port.
IRQ 4	COM1 & COM3	First and Third Serial Port.
IRQ 5	Sound	Sound Card.
IRQ 6	Floppy disk	Floppy Disk Controller.
IRQ 7	Parallel port	Parallel Printer.
IRQ 12	Mouse	PS/2 Mouse.
IRQ 14	Primary IDE	Primary Hard Disk Controller.
IRQ 15	Secondary IDE	Secondary Hard Disk Controller.

Because implementing such a system using priority encoders such as the standard 74LS148 priority encoder IC involves additional logic circuits, purpose built integrated circuits such as the 8259 Programmable Priority Interrupt Controller is available.

Digital Encoder Summary

Then to summarise, the **Digital Encoder** is a combinational circuit that generates a specific code at its outputs such as binary or BCD in response to one or more active inputs. There are two main types of digital encoder. The **Binary Encoder** and the **Priority Encoder**.

We have seen that the **Binary Encoder** converts one of 2^n inputs into an n-bit output. Then a binary encoder has fewer output bits than the input code. Binary encoders are useful for compressing data and can be constructed from simple AND or OR gates.

One of the main disadvantages of a standard binary encoder is that it would produce an error at its outputs if more than one input were active at the same time. To overcome this problem priority encoders were developed.

The **Priority Encoder** is another type of combinational circuit similar to a binary encoder, except that it generates an output code based on the highest prioritised input. Priority encoders are used extensively in digital and computer systems as microprocessor interrupt controllers where they detect the highest priority input.