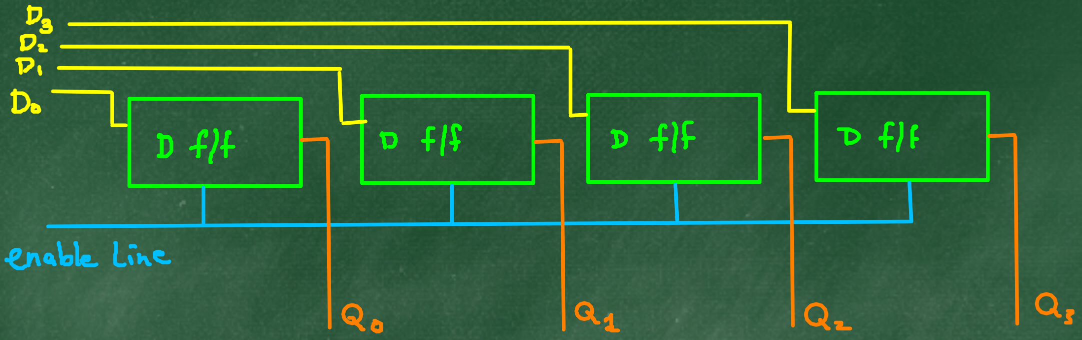
Creating a 4-bit Register from clocked D-Flip Flops.

We have seen in lecture 18 that a clocked D-flipflop acts like a single bit memory.

The clock input (also called the enable input) is used to capture the value of the data input and to store it in the flip flop. The stored value is represented by the value of the Q output.

We will now combine 4 clocked D-flipflops to create a storage component capable of storing 4-bits of information. This component is called a 4-bit register.

In general we can combine n clocked D-flip flops to create an n-bit register.



The enable line is connected to the enable inputs of all the flip flops. We will call it the "write enable" because it will allow us to write to the register.

When the write enable is turned on (i.e, when it is asserted), the values on the Data Lines: $D_0 D_1 D_2 D_3$ are reflected onto the lines labelled $Q_0 Q_1 Q_2 Q_3$, respectively.

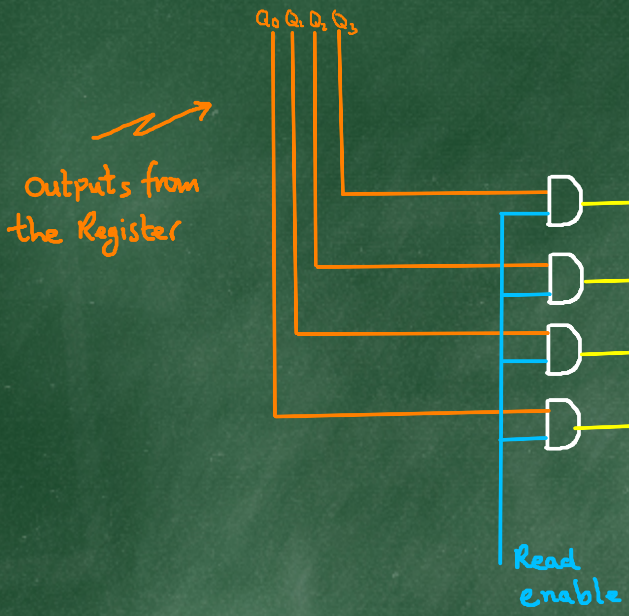
Collectively, the Data lines are known as the Data Bus.

When the write enable is turned off, the values on the Data bus can change without affecting the values of $Q_0 Q_1 Q_2 Q_3$.

In effect, those values on the data bus are stored in (written to) the register when the write enable transitions from on → off.

What about getting these values out of the register?

Consider the following:

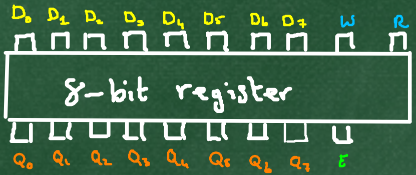


When the value of the Line called 'Read enable' is 0, the outputs of the register are isolated from the data bus.

When the Read enable =1, the register contents appears on the data bus.

We could create an 8-bit (1-byte) register by using 8 -D-f/fs instead of 4.

Doing this, and abstracting it into a black-box package we could imagine creating the following:



D_0 D_1 D_2 D_3 D_4 D_5 D_6 D_7 W R
8-bit register
Q_0 Q_1 Q_2 Q_3 Q_4 Q_5 Q_6 Q_7 E

In this Component, $D_0 ... D_7$ are the Inputs from the data bus, $Q_0 ... Q_7$ (the outputs of the register) Can be Connected back onto the data bus

The w input Corresponds to the write enable line
The r Input Corresponds to the read enable line

I have also introduced a new input at this level of abstraction: the E Input, which I will Call the register enable input, will be used to turn the whole register on and off. When it has the input 0, the register is effectively Isolated from its Surrounding and When it is 1, it is Connected to its Surroundings.

The ability to Connect and disconnect a Component from its Surroundings in this way is very important for, and is fundamental to, Creating different Pathways through a Circuit at different times.

In the next steps to Creating a memory module, we will use register enable Inputs to Implement memory addressing.