## **State Switch Module**

The purpose of the state Switch Module is very crucial in the design of the box. Its purpose is to detect any changes made in the state of any switch, and when a change is made, it updates its output to display the state of the switches BEFORE the change.

Essentially, this circuit always displays the previous state of the switches at any point in time.

We need this circuit because it will allow us to know which switch was turned off last, and in turn, allow us to have a deciding factor that drives the circuit to the appropriate sequence state.

We specifically use the output of this circuit when we are in the sequence detector state. The only way to reach the sequence detector state is to have a switch state of 0000 and after the 4 second timer runs out. This means that there are four different ways to enter this state, from 1000 to 0000, from 0100 to 0000, from 0010 to 0000, and from 0001 to 0000.

This circuit will allow us to know which of these different paths were taken to reach the sequence detector.

A diagram of a state switch module

AI-generated content may be incorrect.

Figure 1 - State Switch Module

The above figure illustrates the black box schematic of the module. As shown, it has five inputs, and the clock.

There are four outputs which represent the previous state of the inputs,

Note that the time dimensions in the operations of this circuit are NOT tied by the clock, they are tied to the change in state of the switches, (t – 1) denotes the time before the change in state of any switch.

### **Principle Of Operation**

This module relies on the ability to detect when a change in state occurs and the ability to store the previous value before the change settles in. This is only possible using a flip-flop, to understand further how this is achieved we draw the following illustration.

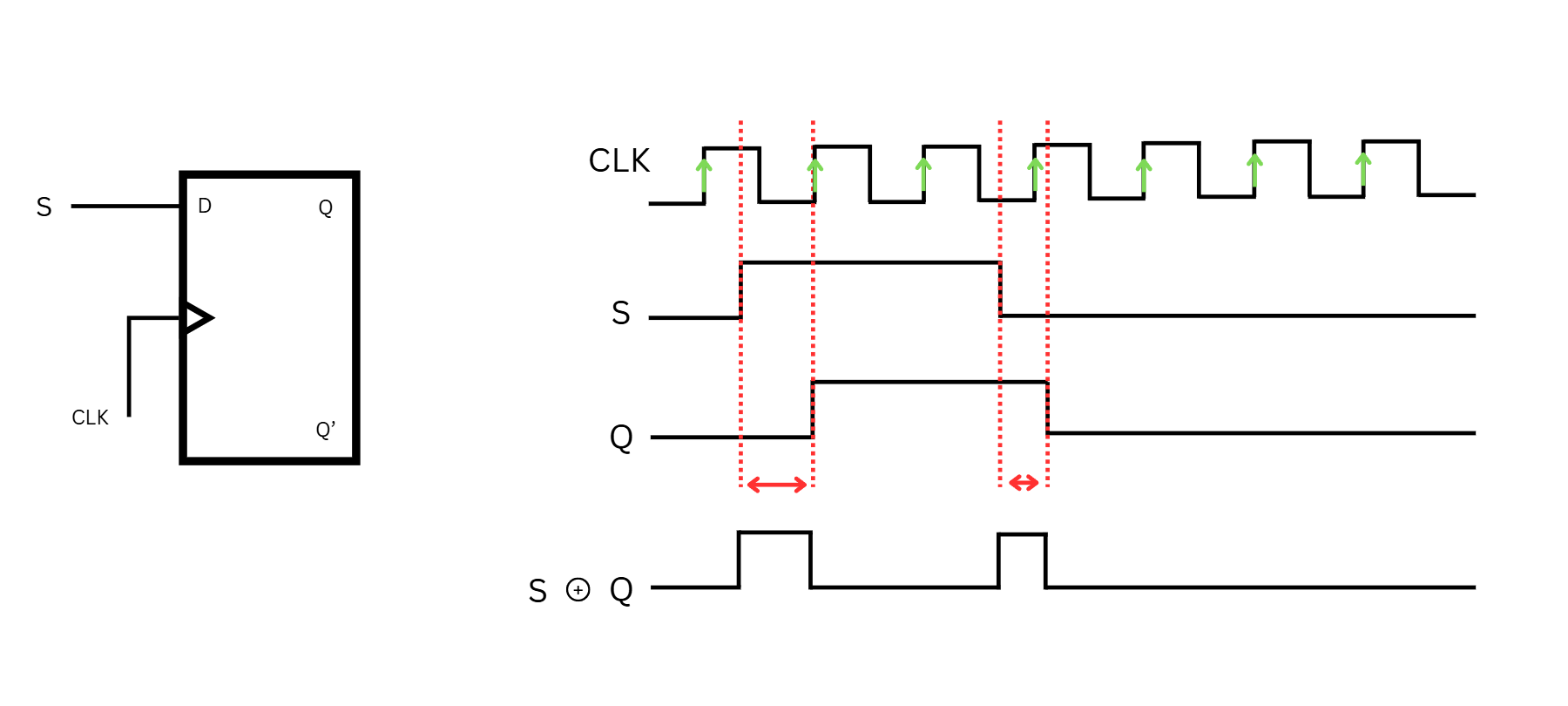


Figure 2 - Switch State Flip Flop Concept

We know that a D flip flop with a clock connected to it only saves the input values at the rising edge of the clock. What this essentially achieves is a brief interval of time (Red Arrows) in which the input did change but the output has not yet registered that change.

If we connect the input and the output to a XOR gate, we are able to detect when a change occurs, because a XOR will output a HIGH signal when the inputs are different, and a LOW when they are the same.

Now we are able to detect a change right before it occurs, all we need to do now is to store the current value before it is altered. The timing for this is crucial, so what we need is another D flip flop that is triggered to save as soon as a change is detected.

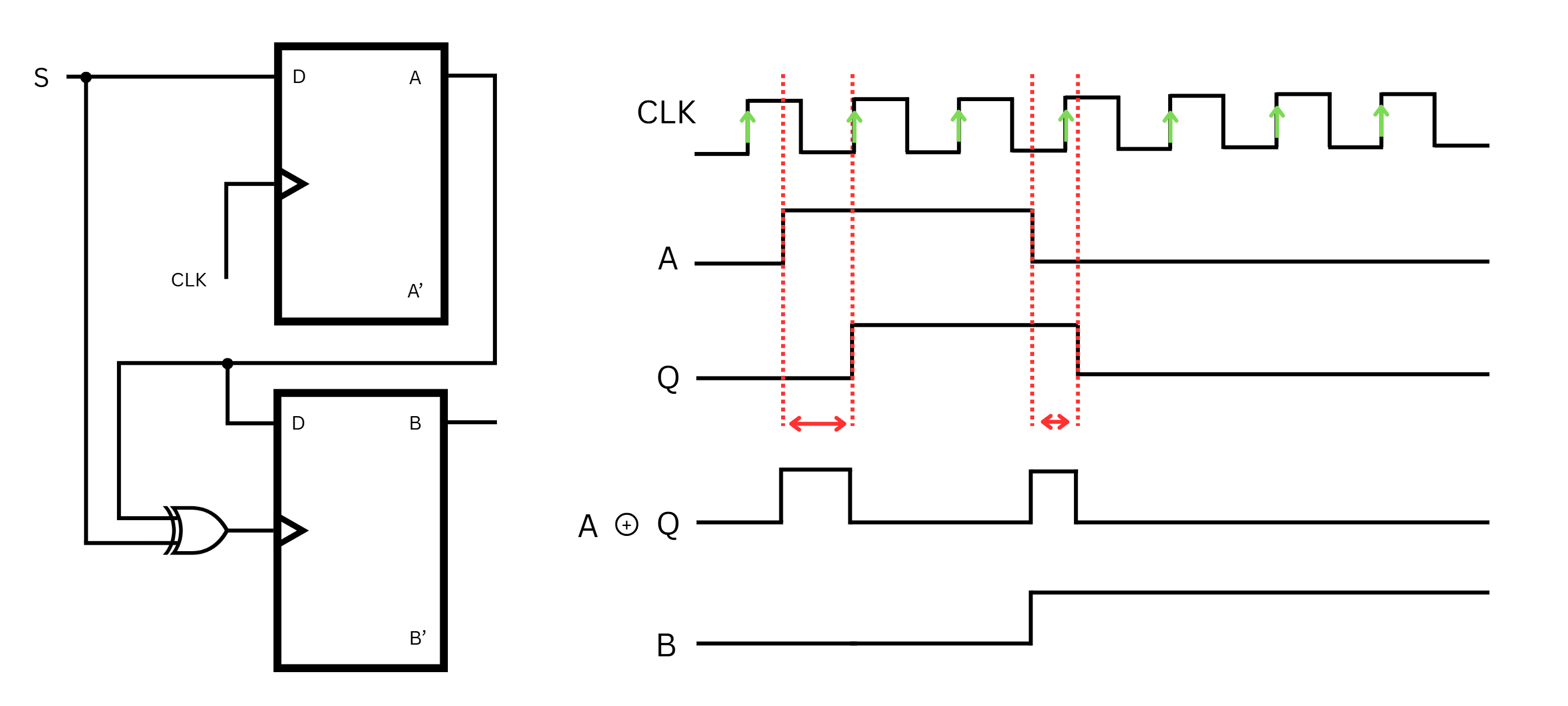


Figure 3 - Single Switch Flip Flop Module

Denote A as the first D flip flop and B as the second D flip flop. If we connect the output of A to the input of B, B will store the value of A only on a rising edge of its clock. Since we set the edge detector as the clock for B, it will save the value of A exactly before it changes, thus we have made a circuit that saves the previous state of the switch.

### **Final Assembly**

In order to accommodate for four switches, we need to add these flip flop pairs for every switch with one difference. The clock input for flip flop B must save must serve to allow all the B flip flops to save when the state of ANY switch changes. Thus, we take the XOR circuits from each switch and OR them, then, we connect said output to the clock input of each secondary flip fop.

A diagram of a block diagram

AI-generated content may be incorrect.

Figure 4 Switch State Module Final Circuit

With the final circuit visualized we construct the module on the breadboard using four 74LS74 ICs, one 74LS86 IC and one 74LS32 IC.

A diagram of a circuit board

AI-generated content may be incorrect.