Computer Systems Assignment 1

Daniel Coady (102084174) – 12:30 Wednesday

25/08/2019

Stage 1

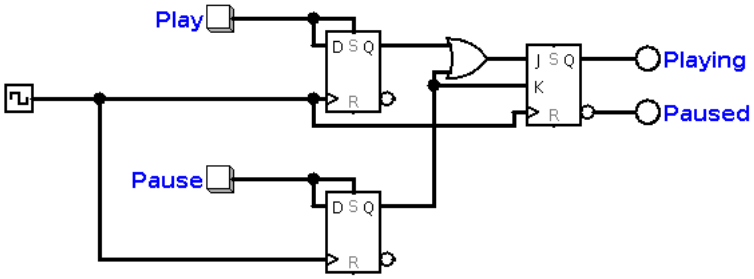

Figure 1: Circuit used for stage one – Play/pause functionality

For this stage I have two inputs in the form of buttons for play and pause. The way this is structured is by having a sort of latch for each button so that on press the state is stored for the next clock cycle to act upon. The latch is very simple, consisting of a D flip-flop that has both the set and D pins connected to the button. This is so that as soon as the button is pressed it's state is set and stored until the next clock cycle. It's important to note as well that the D pin must be connected to the button along with the set pin. This is because without doing so the latch would not be able to ever reset back to a 0 state, making it not very useful. Once we pass the latch, the play button goes into the J input of a JK flip-flop which turns on the Q output for us. The play button is similar in that it then feeds into the K input of the JK flip-flop, however we also need to give the pause button the ability to start playing again from a paused state if pressed. To do this we can connect the pause button to both J and K inputs, transforming it's functionality into a T flip-flop when the pause button is pressed. There is one issue however: the play button is already connected to the J input to trigger the playing state. To get around this so that we can allow both the play and pause buttons use the J input we can use an or gate which both buttons will connect to and have the output going into the J input. With this full circuit we can now press play to go into the playing state and press pause to both go into the paused and playing state.

Stage 2

For this stage I assumed that the pins/buttons could continuously make the volume go up or down. With this in mind, I found it to be most appropriate to use a bi-directional shift register to control volume values. This is for two reasons:

- It allows for easy raising or lowering of the required bar graph display
- It inherently does not allow for overflowing or underflowing values

Of course, this isn't just a bi-directional shift register. I've made some modifications to the inputs of circuit so that it may fit our application better. There are two inputs to

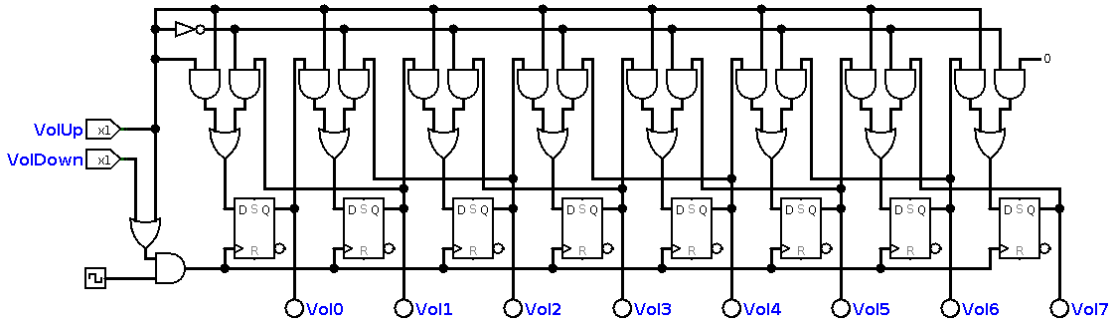

Figure 2: Circuit used for stage two – Volume control

this ciruit: the volume up and volume down pins. Both of these pins are connected to an or gate which feeds into an and gate that also has the clock as the input. This allows us to control when we allow the clock pulse to feed into the circuit so that it can hold state and not cause any unexpected behaviour.

This design is not without flaws however, since if you put both inputs on then it will simply continue increasing the volume. I did attempt to fix this by replacing the or gate with a xor gate, but this had it's own issue. When the xor gate was in use, occasionally when you go from having both inputs on to just having the volume down input on it would go all the way to zero from any state, which is not behaviour that we want. This may be solvable with a latch to time the inputs to the clock, but I haven't been able to apply this yet. As such, I chose the solution that gives the most consistent and useful output to the user.

Stage 2

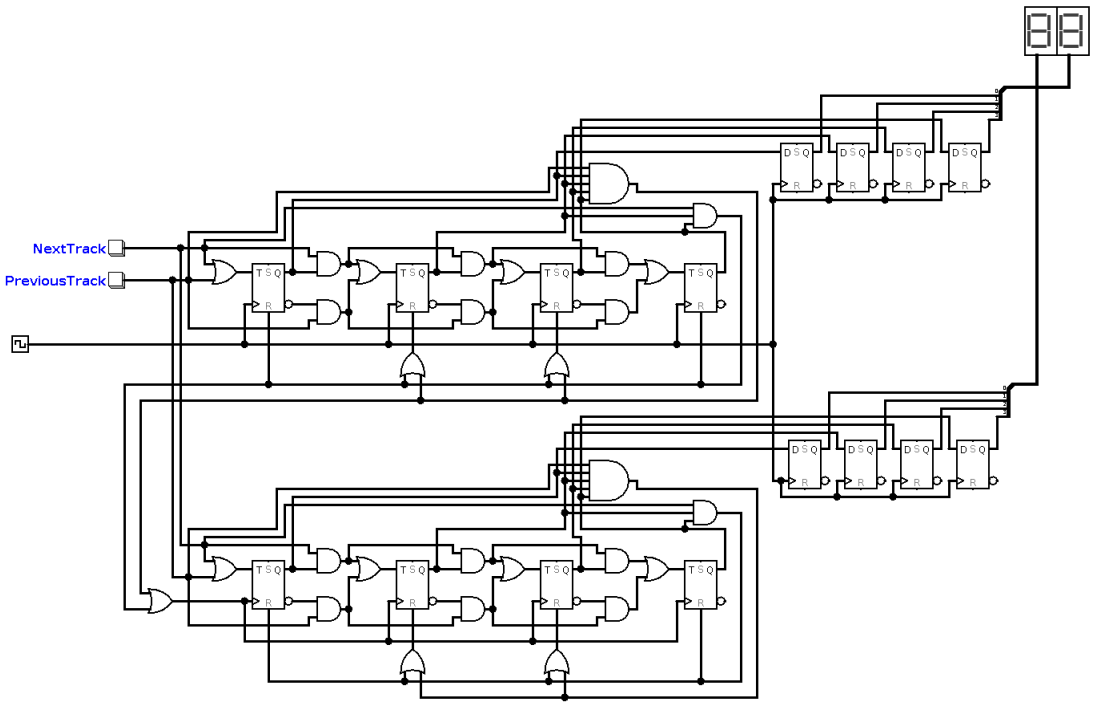

Figure 3: Circuit used for stage three – Track control

Here we have the track control circuit that can be interacted with by using two buttons. For this I assumed that for inputs we can have buttons that continuously increment or decrement the track number. This circuit, unlike the previous two, are what I like to refer to as "semi synchronous" since while half of it is synched to a common clock, the other half is actually aysynchronously clocked to the output of the first half.

The basic overview is that this circuit works by having two 4 bit mod 10 bi-directional counters, one for the units column and one for the tens column of our decimal number that is being displayed on two hex displays. For both counters there is logic to allow it to count forwards or backwards in the form of two and gates and an or gate for each T flip flop. As well as this, there are two and gates that are used to check for a state of 10 (little endian 0101) and 15 (little endian 1111) so that it can wrap around to the correct numbers. For each and gate there is also a control bit which is controlled by the next and previous button respectively to ensure that they are only "active" as long as they need to be for checking states. This makes sure that we're not getting false positive states when going either direction, allowing for smooth operation.

For each circuit however, there are very slight differences. The main thing of note is that it the second counter is not connected to the common clock for the circuit. This is

because the first counter will instead feed the signal from overflow or underflow states (when it hits 10 or 15 respectively) into the clock input of the next counter. This is why I mentioned previously that this circuit is "semi synchronous". However, this does not end up mattering all too much since there is a fully clocked register that is used as a buffer to ensure that all output signals are clean and no illegal states get out.

There is one issue with this circuit, that being that the registers delay the output signals by a tick. This is a worthwhile trade off in my opinion however, since this ensures data integrity which is important for preventing unexpected behaviour caused by illegal states.