Homework III **Benjamin Streit**
Professor Gary J. Minden EECS 388
Date Assigned: 09/25/18 Date Due: 10/02/18

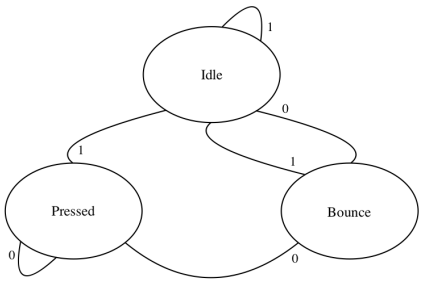
1. What are the three states (as discussed in lecture) of pressing a button?

   The three states of pressing a button are as follows: (I) Idle/Off, (II) Bounce, and (III) Pressed/On.

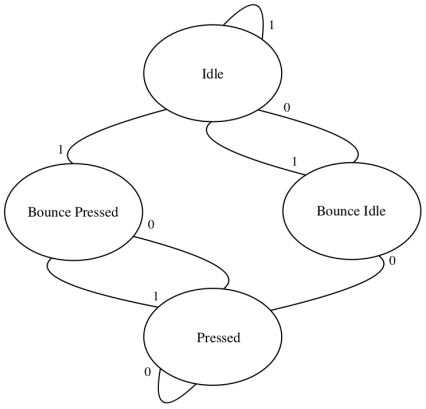
2. Suppose a button bounces on an initial press and on release. Describe enhancing the button state approach to sensing the button.

   To begin, the state diagram discussed in lecture will be analyzed, where no bounce upon state change from pressed to idle is assumed.

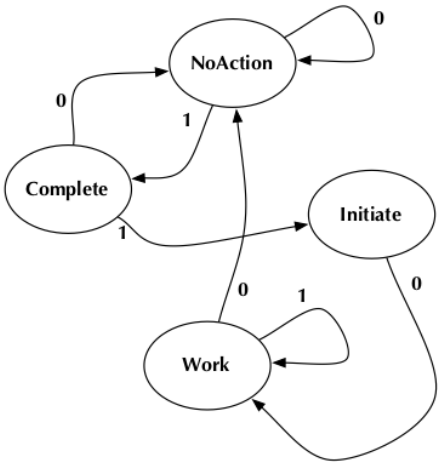
   At Idle, if Idle, stay at Idle, but if Pressed, move to Bounce. At Bounce (after waiting), if Pressed, move to Pressed, but if Idle, move back to Idle. At Pressed, if Pressed, stay at Pressed, but if Idle, move back to Idle. See below diagram for a visual representation of this description.

   This state approach will be modified slightly to account for the bounce upon release of the button. Rather than an `Enum` consisting of Idle, Bounce, and Pressed being used to define the states, an `Enum` consisting of Idle, Bounce Idle, Pressed, and Bounce Pressed will be used. With the use of this `Enum`, Bounce Idle takes the role of Bounce from the state diagram above. In addition, at Pressed, if Idle, move to Bounce Pressed, but if Pressed, stay at Pressed. At Bounce Pressed (after waiting), if Pressed, move back to Pressed, but if Idle, move to Idle. All else behaves the same as the above state diagram. See below state diagram for a visual representation of this description.

For questions 3 - 5, consider the state diagram below.



3. Write a C statement that defines the states of the state diagram.

The states of the above state diagram are described using an `Enum`.

```
Enum ButtonState { Initiate, Work, NoAction, Complete };
```

4. Which C control structure might you use to implement the state diagram?

The above state diagram would be well implemented using a `switch` control structure. See below pseudo-code.

```
switch ( currentButton ) {
    case Initiate: {
        < sample currentButton >
        if ( Button == 0 ) {
            currentButton = Work;
        }
        break;
    }
    case Work: {
        < sample currentButton >
        if ( Button == 0 ) {
            currentButton = NoAction;
        }
        break;
    }
    case NoAction: {
        < sample currentButton >
        if ( Button == 1 ) {
            currentButton = Complete;
        }
        break;
    }
    case Complete: {
        < sample currentButton >
        if ( Button == 0 ) {
            currentButton = NoAction;
        } else {
            currentButton = Initiate;
        }
        break;
    }
}
```

5. Is the state diagram completely defined? Why or why not?

The state diagram is not completely defined. For the diagram to be completely defined, there must be definitions for each possible state, as well as definitions for each possible value for said states. The state diagram in question has definitions for `1`'s and `0`'s for every state except for `Initiate`, which only has a definition for a value of `0`. In order for the state diagram to be completely defined, in must include a definition for `Initiate` with a value of `1`.

6. For a frequency of 510 Hz, what is the period?

The period is determined as follows:

$$\text{Period} = \frac{1}{\text{Frequency}}$$

Thus, for a frequency of `510 Hz`, the period is:

$$\text{Period} = \frac{1}{510\ \texttt{Hz}}$$

$$\text{Period} = 1.960784314\ \texttt{ms}$$

7. Given our SysTick frequency is 10,000 Hz or 0.1 ms resolution and our desired frequency is 510 Hz, what is the actual frequency?

With a resolution of `0.1 ms`, it is clear the closest achievable period to the period for a frequency of `510 Hz` is `2.0 ms` (see question six). Thus, we must determine the frequency given a period of `2.0 ms`. This frequency is determined as follows:

$$\text{Period} = \frac{1}{\text{Frequency}}$$

$$0.002\ \texttt{s} = \frac{1}{\text{Frequency}}$$

$$\text{Frequency} = \frac{1}{0.002\ \texttt{s}}$$

$$\text{Frequency} = 500\ \texttt{Hz}$$

From this calculation, it is seen that given a `0.1 ms` resolution and a desired frequency of `510 Hz`, the actual frequency achieved is `500 Hz`.