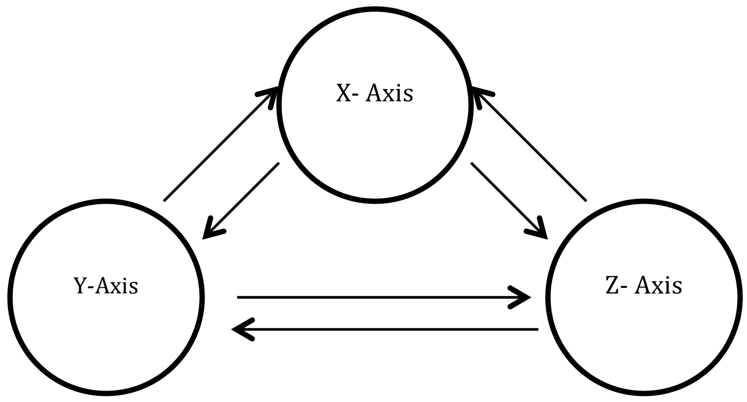
P542 Hardware System Design II

Lab Report 3

Enrique Areyan

Description of solution

Using the ideas developed in lab 1 and lab 2, for lab 3 the easiest solution was to represent the program as a state machine. Three states were needed, one for each axis. The following diagram represents the program:



In this case we can go from one state to either one of the other two states. Two different events trigger the movement between states: either the user presses the user button or we receive a character via keyboard to move to a state. If the user presses the user button we move to the next state in the sequence X, Y, Z. If the user inputs a character, then we move directly to the state receive as an input either 'X', 'Y' or 'Z'. Any other characters are ignored.

Like lab 2, note that all of the state machine is wrapped around a `while(1)` and implemented as a `switch` statement. We poll data from the gyroscope at each iteration right after the while statement and regardless of the state of the machine. In this case we avoid any potential buffer issues like those faced in lab2.

Description of issues

The main issue was to get the data from the gyroscope. In turn, this issue could be divided into two: (1) set up the gyro; (2) get the data in the correct format.

(1) Setting up the gyro entailed writing a driver ds_gyro.c with various functions. Like labs before, we had to use the pattern: initialize clock, initialize structure, setup pins and, additionally, initialize the SPI.

(2) Getting the data in the appropriate format entailed writing appropriate read and write functions. The most complex part was that to send bytes one needed to also receive, a key difference between UART and SPI. Finally, to receive data correctly formatted from the gyro we had to use an int16_t data type.