Alexis Savoie

3389902

Lab 5

June 18th, 2017

Part 1:

1. The code from Lab5.c ran as expected.
2. The loop is running at 8388608 loops per second which can be represented as 2^23 loops per second.
3. A) The registers corresponding to the C-language variables are:

HEXDISP: r4

HEXCTRL: r3

count: r2

B) The code generated for the while loop produced the following:

stw r2, 0(r4)

add r2, r2, r3

br -0xc

For the most part this would have been fairly similar to the code that we wrote, however we would have used a label and branched back to the name of the label rather than branching to that specific place in memory.

C) There are 3 assembly language instructions in the while loop. This means that the number of instructions executed per second is 8 388 603/3 = 2 796 202.

Part 2:

When scanning the bar code we observed that the black bars varied in size while the white bands remained a constant size. We expected that the last black bar, which was wide, would be twice the width as the last white bar. This was then repeated at different speeds and the same results were produced relative to the speed that the scan was performed at. Due to the fact that the size of the white bars remained constant we were able to determine if the next black bar was wide or narrow by comparing it’s processing time to that of the white bar that came before it.

The width of the first black bar was difficult to determine as there is no white bar that comes before it. We accounted for this by outputting a ‘S’ char to indicate the start of the scan as seen in figure 1 so that the results of scanning the first bar would be more consistent.

To enhance the program, it is possible to add another do while loop that would allow us to read in the other direction. In order to access this do while loop we could have a conditional statement inside of our infinite while loop that would determine which direction the bar code is being read from and the access the appropriate loop from there.