README.md

MIC-1 Fibonacci

The approach I took to solving the Fibonacci MIC-1 program was to first write the program in C, so I could better understand the task and have something to translate into MIC-1 assembly. I wrote the C version (which can be found on GitHub) to use the following formula: Fn = Fn-1 + Fn-2. I made a Fibonacci function which accepts an integer and returns the Fibonacci number for that number. After writing the C version I wrote several versions of the MIC-1 assembly. It took a while to figure out MIC-1 and I used the class website as well as Prof. Moloney's examples on his site. In particular Prof. Moloney's adder.asm program was helpful for testing out the MIC-1 executables (masm / mic1) and for modifying it to see how MIC-1 assembly works. I built the masm / mic1 binaries by git cloning the mic1 repository.

At the moment I'm not aware of any problems with the MIC-1 program itself. It pushes the first 25 Fibonacci numbers to the stack, and from my testing I can enter the debugger at location 999 and see the first Fibonacci number, 0, and I can then print out the next 24 Fibonacci numbers (1 1 2 3 5 etc). Technically I calculate the Fibonacci numbers in versus order since I start at 25 and work my way down to zero. I don't believe that is a problem though since when you enter the debugger it starts at the first Fibonacci number, since that one was the last number pushed to the stack. It is also the easiest way I could think of doing in MIC-1 since you can only jump if zero, negative, positive or not equal to zero. I did have some issues with the Makefile and trying to get the test.out to match the program's actual output. It seems that when redirecting input using the "<" operator whatever was inputed is not outputted to the screen, and as a result it doesn't get redirected to the file actual.out (using the ">" operator). I had to modify what I thought the program would output to match what it actually outputs when running the following command:

```
./mic1 prom.dat fib.obj 0 1024 < test.in > actual.out || true
```

I learned a fair amount of MIC-1 assembly. I feel pretty confident with assembly again, after having taken the assembly class roughly a year ago. I had some experience with MIPS as well so it was pretty nice to pick up on the MIC-1 instructions quickly.

I feel since the program outputs the first 25 Fibonacci numbers to the stack, and that I use a recursive function to calculate out each Fibonacci number that I deserve a .9011 for this work.