Using the `include “filename.v” syntax, all the child module files are already included in the 2bitcomputer.v file. So to run program only call the 2bitcomputer.v file.

1. Download all .v files
2. Compile

Ex) Using Icarus Verilog:

cd\iverilog\bin

iverilog –o 2bc.vvp 2bitcomputer.v

vvp 2bc. vvp

gtkwave 2bc.vcd

**Note:** In the test bench printout it says register R1 starts at 01 when it actually is initialized at 00. Set is initialized at 0 and Reset at 1 (all flip flops and counters go to 0). At time #0 seconds Reset = 0, and R1 is incremented, hence the test bench prints out 01 as the first value.

To load different data into memory addresses, use the X and Y values in the test bench to change the inputs on the muxs. Thus if one wanted to load data 01 into address 01 they would initiate a Y1=0 and X1=1. See comments in code.

The address pins (PC1\_final,PC0\_final) are set through the muxs to store bits as follows:

|  |  |
| --- | --- |
| Address (PC) | Data (D1,D0) |
| 00 | 00 |
| 01 | 01 |
| 10 | 00 |
| 11 | 10 |

The data pins (D1 and D0) are hard wired to trigger each function on certain values.

(INC on 00, JNO on 01, HLT on 10)

So the code will execute whichever instruction is pointed to by the data in the address PC chooses.

Ex) When PC = 00, Data = 00, therefore INC is now high

When PC = 01, Data = 01 therefore JNO is now high

When PC tries to go to 10 it jumps back to address 00 contained in data

Once RS is high, PC goes directly to 11 from 01, skipping 10. Now data = 10, therefore HLT is now high and everything freezes