Sean Wilson
11 - 8 -15
CS 552

Stage 1 Report

Folder Hierarchy:
Project

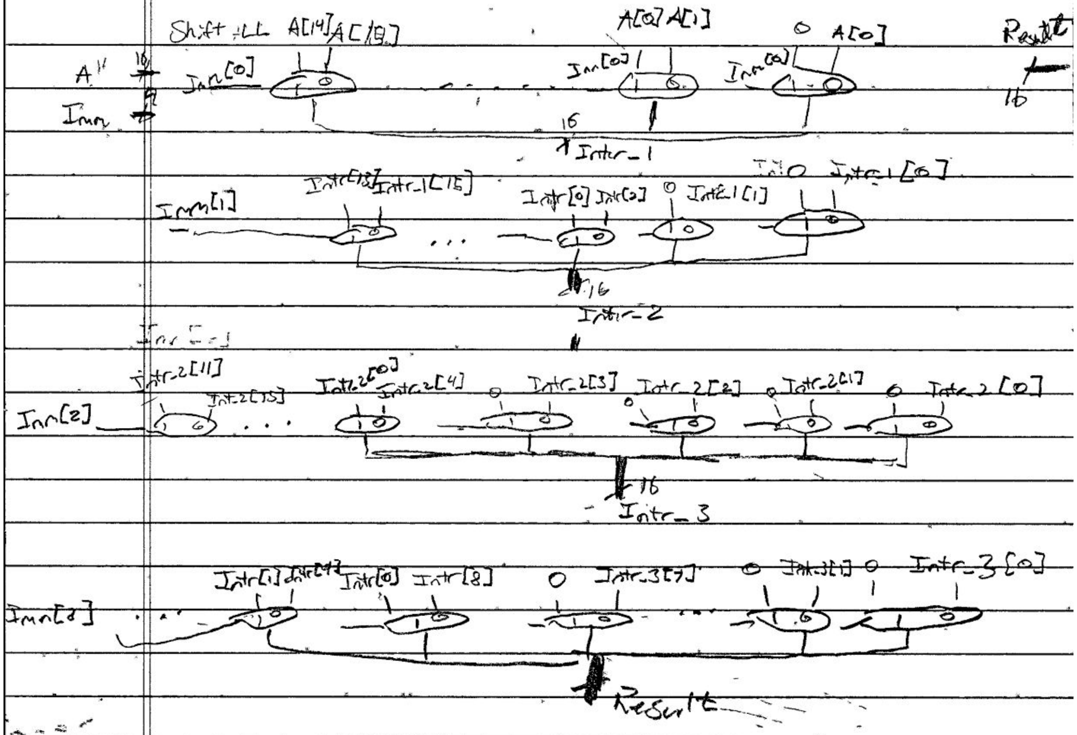
- **compiled_tests: Contains the compiled testbench for all test programs**
  - **test_progs_hex: Contains the hex format instruction mem progs**
- **Report: Contains this and screenshots tests.**
- **wisc_15_processor:**
  - makefile - type make to create a.out of cpu_tb
  - control.v - processes incoming instruction to produce control signals
  - cpu_tb.v - Used to build and test cpu.v
  - instr_logic.v - used to handle branch usage
  - rf_pipelined.v - register file
  - instr_mem.v - instruction memory
  - data_mem.v -data memory
  - hazard_detect.v - The hazard detection unit (VERY BASIC)
  - **alu: Contains all the makings of the alu**
    - alu.v - The top level of the alu, muxes out shift, add, sub, paddsb, lb
    - au.v - The arithmetic unit: Handles add, sub, paddsb
    - full_adder.v - A full adder used in au.v
    - shifter.v - Handles all versions of shifting
  - **tbs: Contains various testbenches used throughout production**

SCHEMATICS ORDERING:
Shift LL, Shift RL/RA , AU, Overflow Detection, and sum bit masking, ALU Top Level, CPU Top Level, Control Signals

Test result screencaps are labeled on the last line of the command line in the printout.

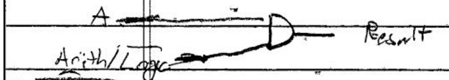
Design comments:
Branches are taken in the EX stage. HLT also occurs in EX stage which thus allows the PC to go two instructions past halt. Finally, a “HLT” Instruction causes a “rolling shutdown”. The pipeline continues, but once the HLT instruction passes through a phase of the pipeline the register proceeding it can no longer be written.

To run tests:
Tests are binaries compiled on an x86 linux system using iverilog. They should be runnable on CSL linux computers by typing ./<TESTNAME>.bin

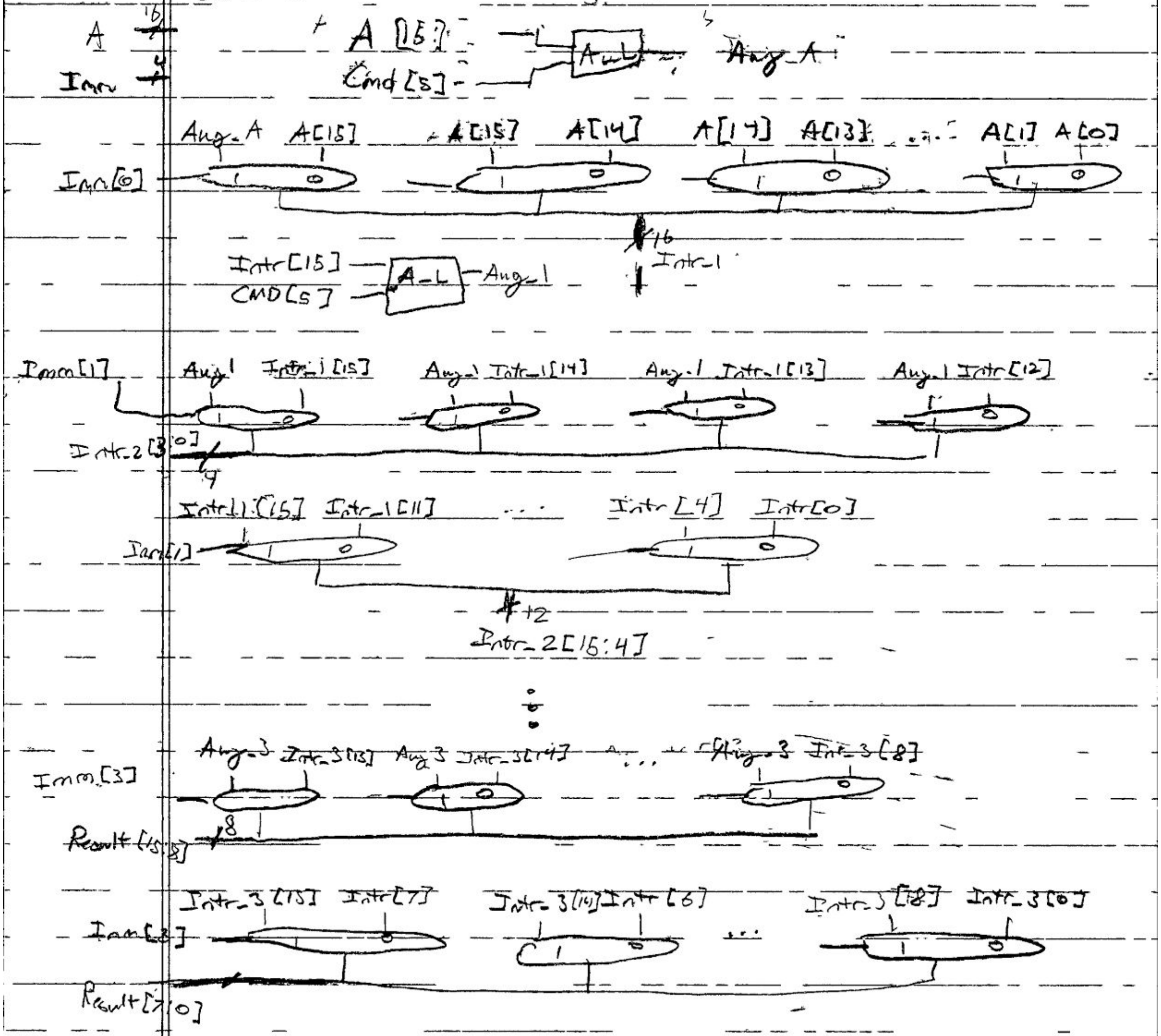


Arithmetic/Logical Right "Other" Bit logic (A_L)

A_L = Arithmetic Mode • Bit



Shift R


A 16
Imm 4
A [15]
Cmd [5]
A_L
~~Arg_A~~
Arg_A A[15]
A[15] A[14]
A[14] A[13]
A[1] A[0]
Imm[0]
1
0
16
Intr_1
Intr[15]
CMD[5]
A_L
Arg_1
Imm[1]
Arg 1 Intr_1[15]
Arg_1 Intr_1[14]
Arg_1 Intr_1[13]
Arg_1 Intr[12]
Intr_2[3:0]
4
Intr_1[15] Intr_1[11]
...
Intr[4] Intr[0]
Imm[1]
12
Intr_2[15:4]
Arg_3 Intr_3[13] Arg_3 Intr_3[14]
~~Arg_3 Intr_3[8]~~
Imm[3]
Result[15:8]
8
Intr_3[15] Intr[7]
Intr_3[14] Intr[6]
Intr_3[8] Intr_3[0]
Imm[3]
Result[7:0]

SW 4

ALU

Arithmetic Units (Alls)

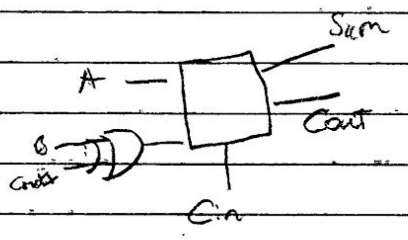
[15:0]

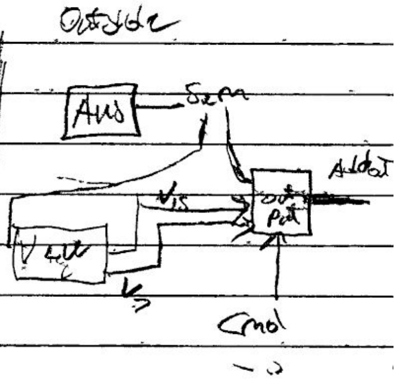
A —

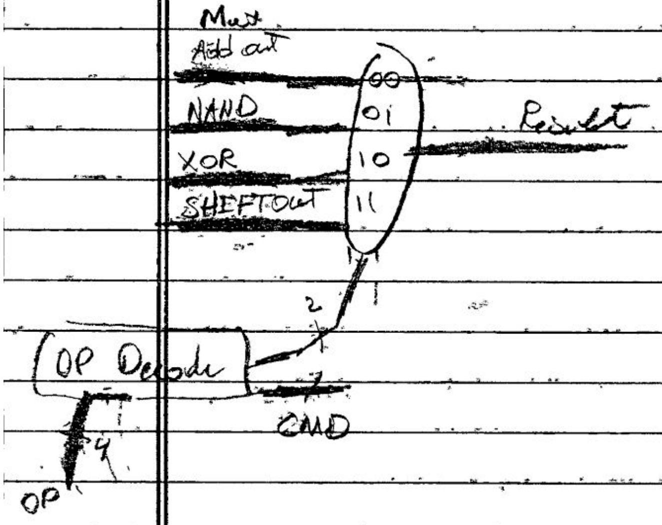
B —

CIN —

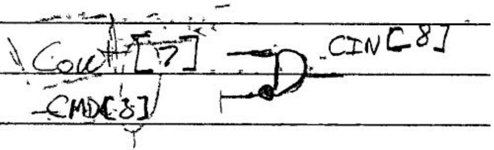
Cmd 1 —







CIn for All [8]



Signals PADDSB

CIN for All[0]

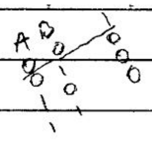
C1 → CIN[0]

Logic:

$V = \overline{a_{n-1}} b_{n-1} s_{n-1} + a_{n-1} b_{n-1} \overline{S_{n-1}}$

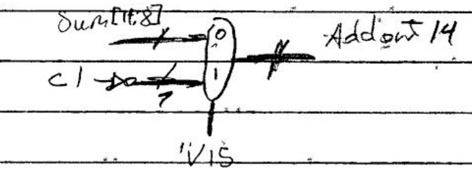
$N = \overline{V} \cdot S_{n-1} + V \cdot C_{n-1}$

Z = ORC Bits [0:16]

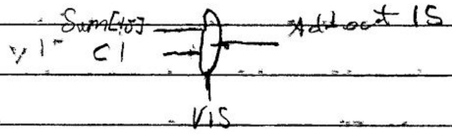


Add Out Unit (Upper half)

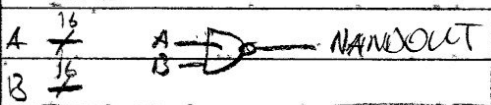
// Bits [14:8]



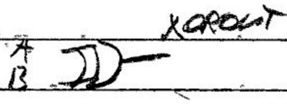
// 15th Bit



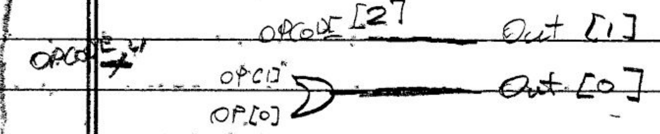
NAND



XOR

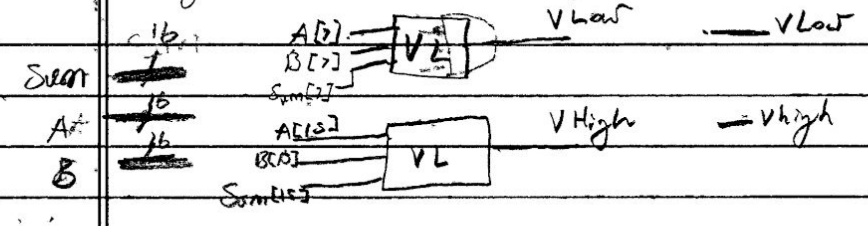


OP "Decode"

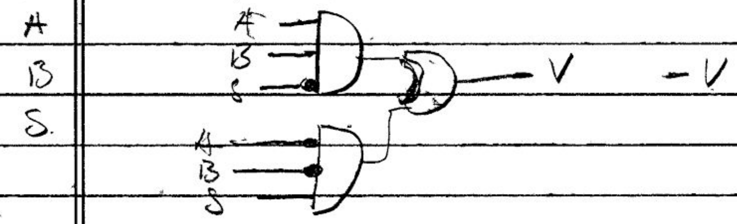


n_low = n & ~Cmd[3] | Cmd

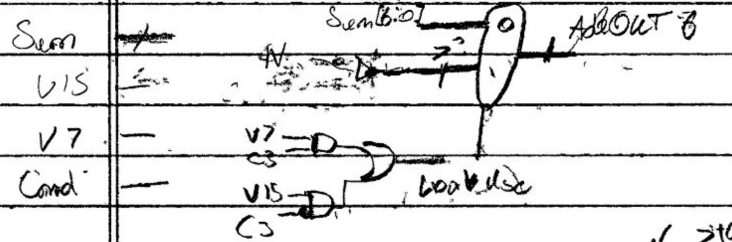
V logic Unit



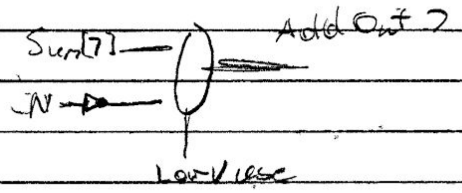
VL



Add Out Unit (Lower Half) //First 6 Bits

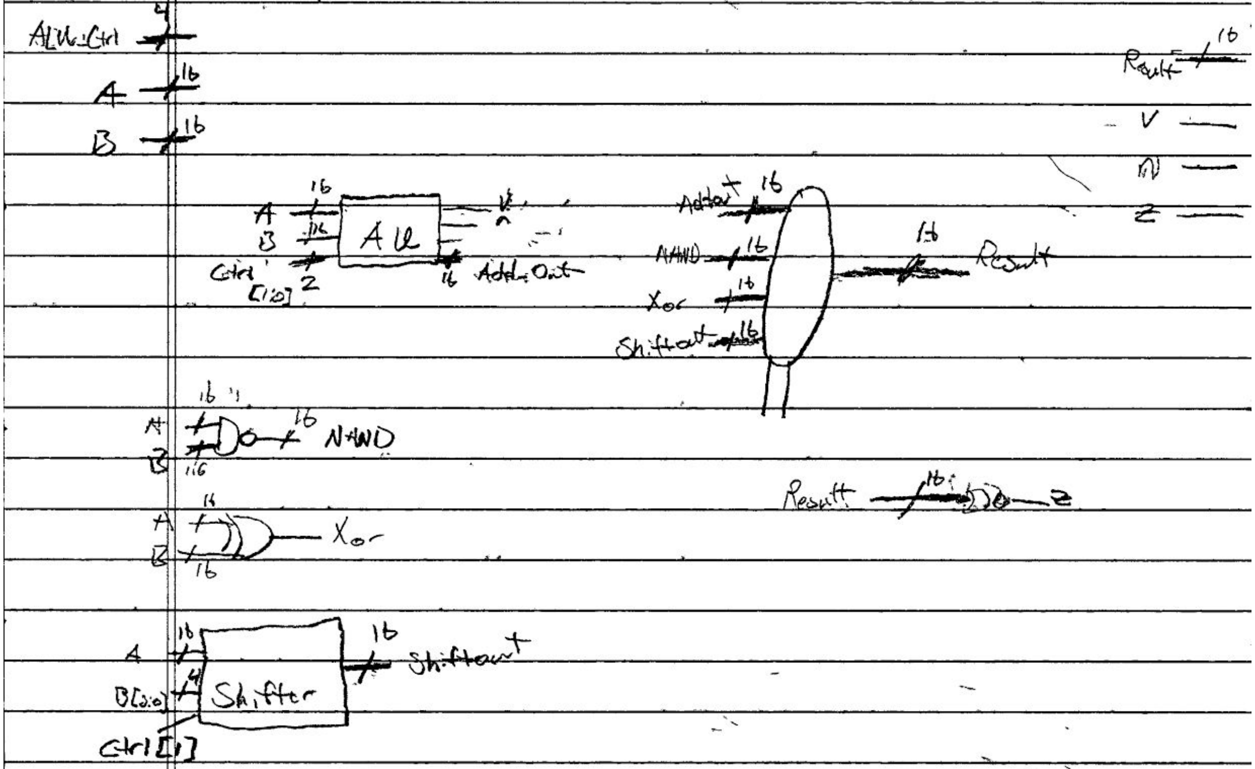


// 7th Bit



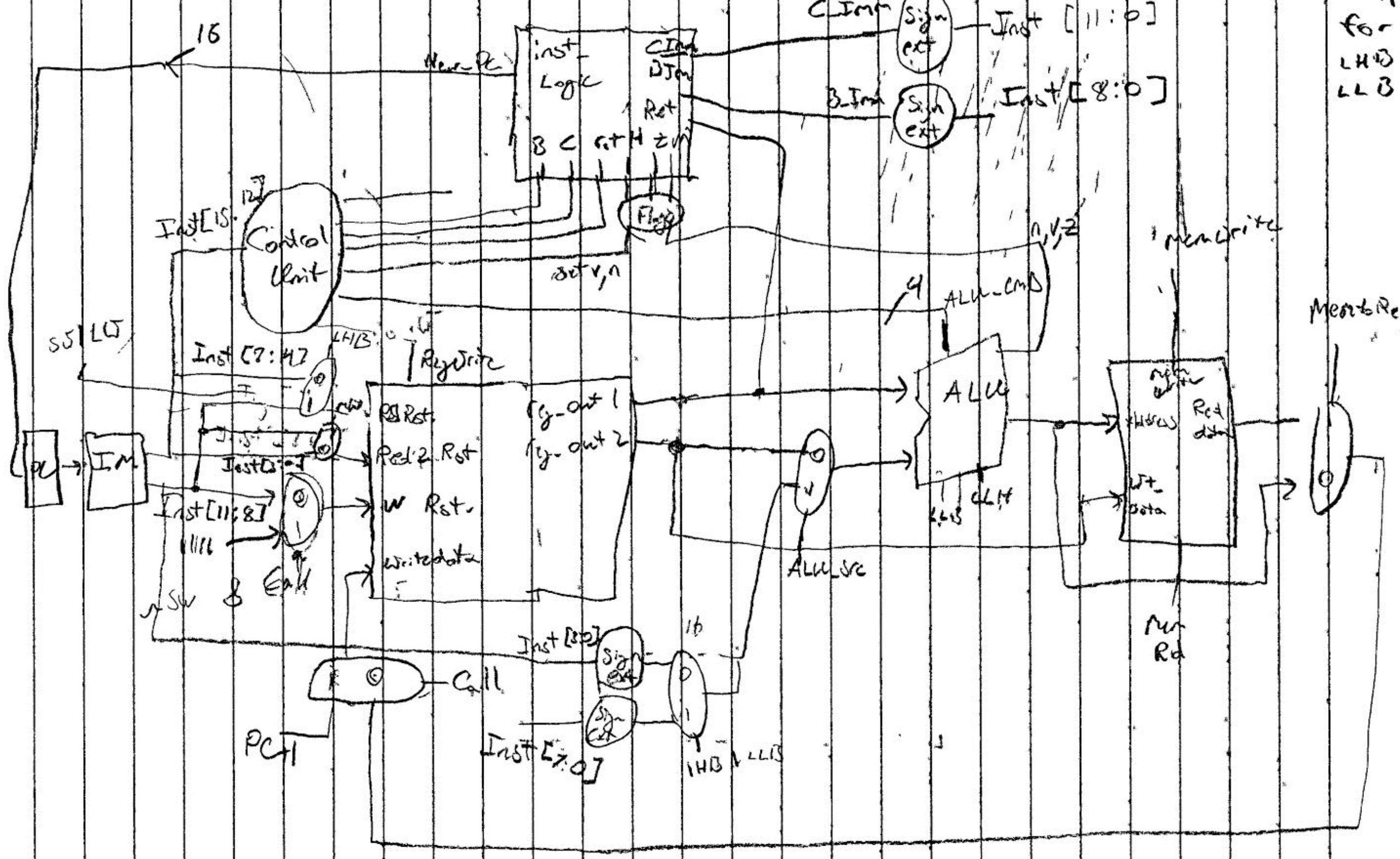
ToDo: Check C1 might need to be replaced with N flag

ALU Top Level

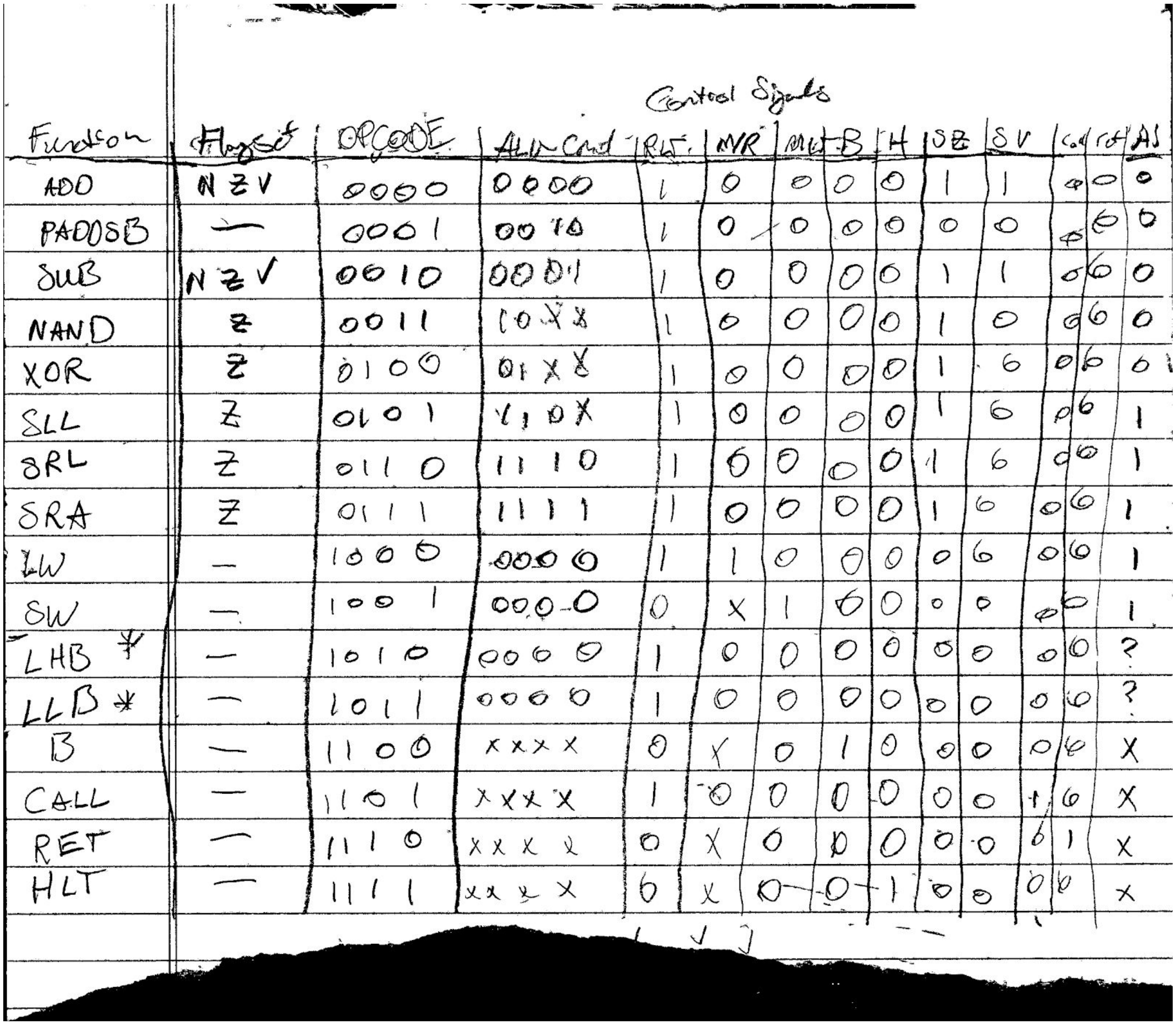


Processor Datapath

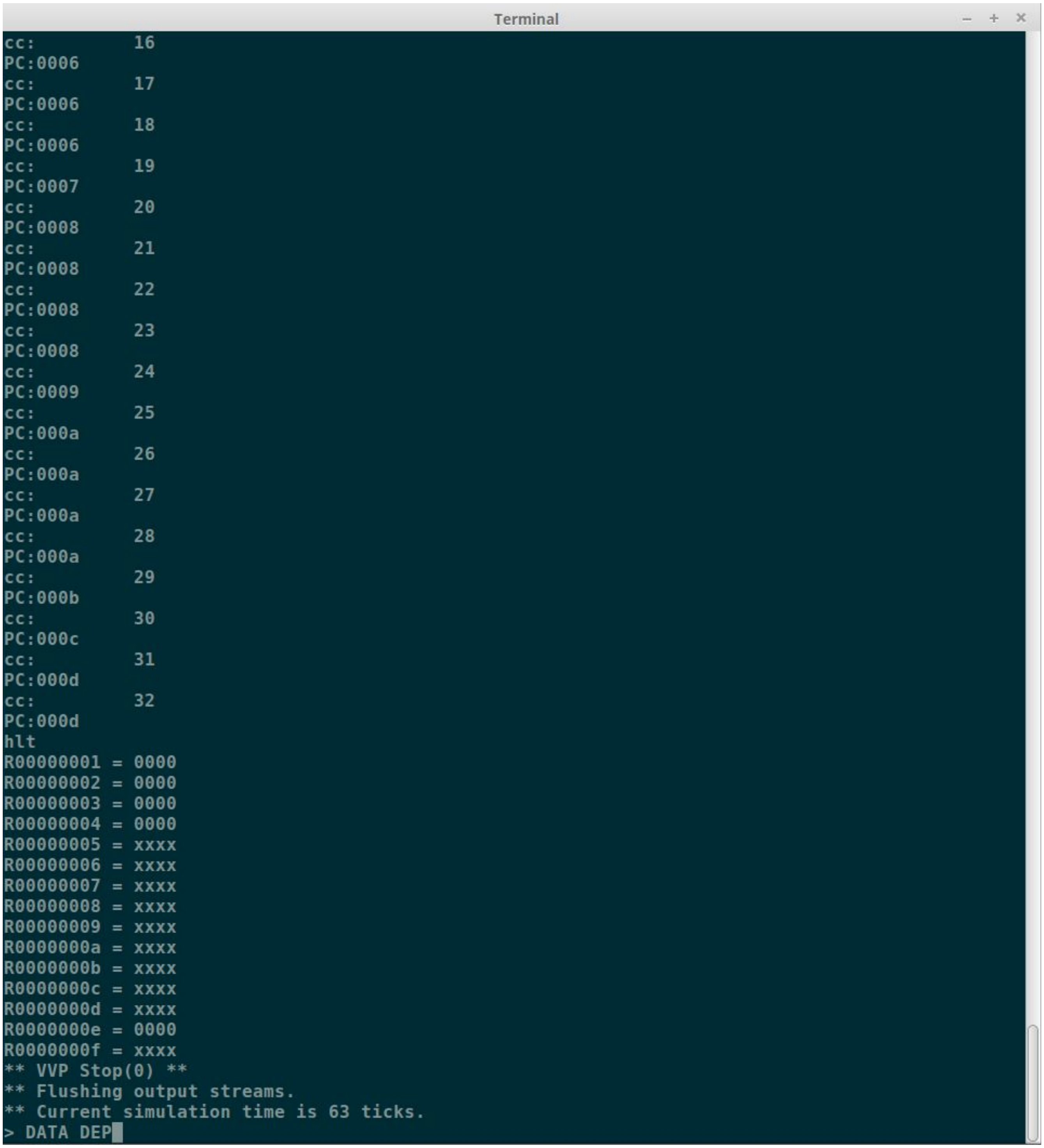
TODO:
~~Add~~
Support
for
LHB
LLB



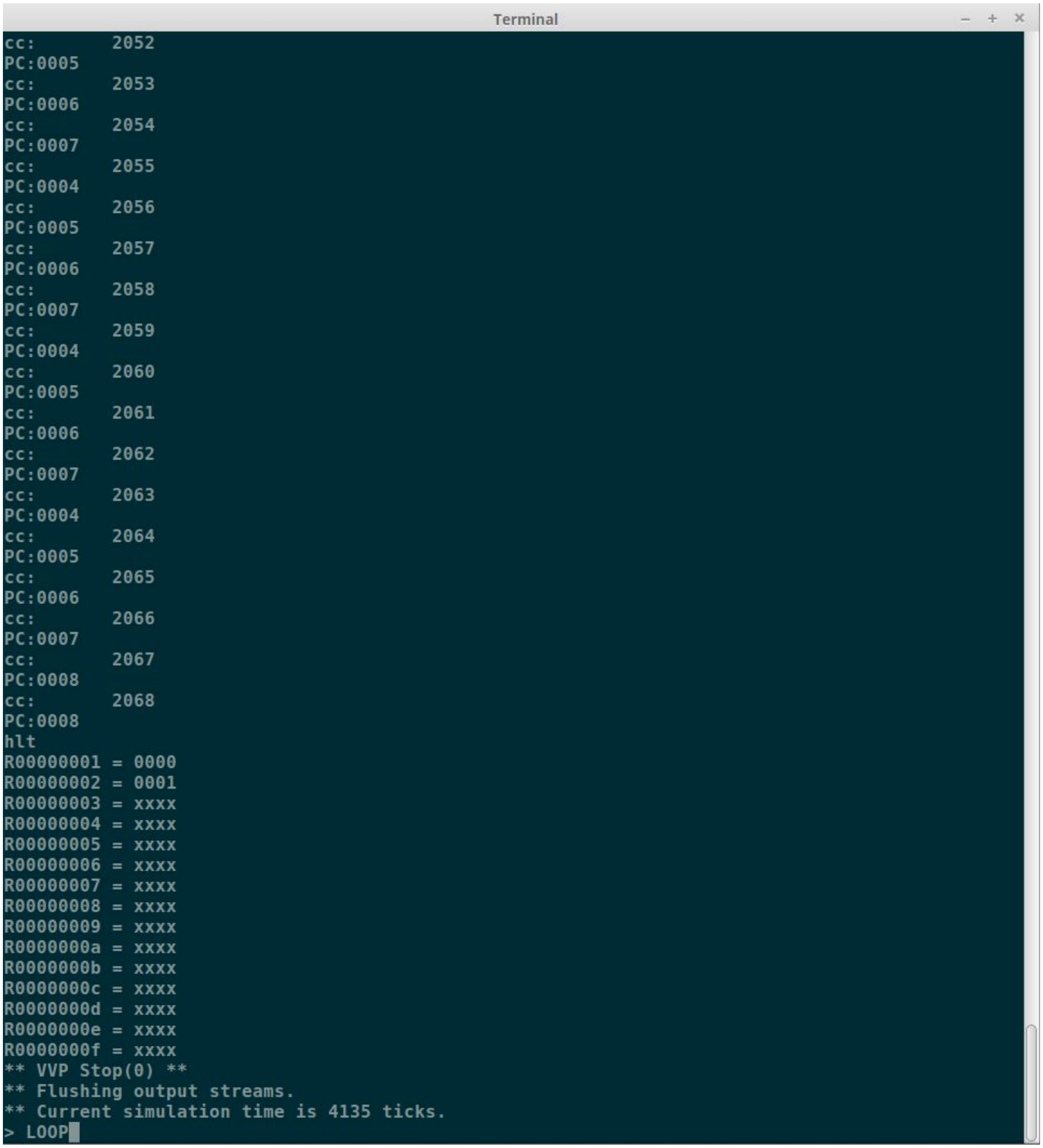
SW 9



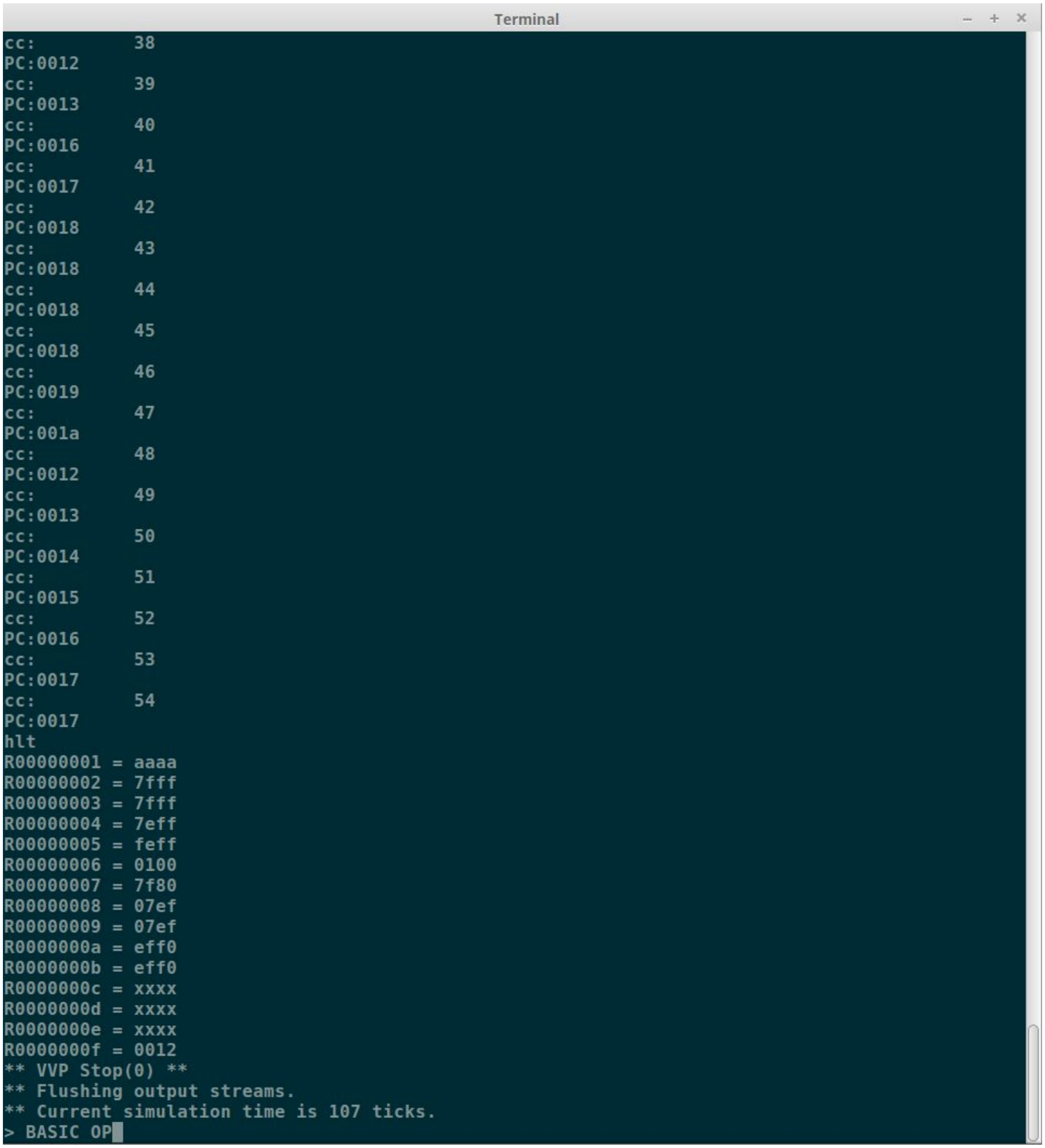
				Control Signals									
Function	Flag set	OPCODE	ALU Cmd	RW.	MVR	MW	B	H	SB	SV	Call	ret	AS
ADD	N Z V	0000	0000	1	0	0	0	0	1	1	0	0	0
PADDSB	—	0001	0010	1	0	0	0	0	0	0	0	0	0
SUB	N Z V	0010	0001	1	0	0	0	0	1	1	0	0	0
NAND	Z	0011	10XX	1	0	0	0	0	1	0	0	0	0
XOR	Z	0100	01XX	1	0	0	0	0	1	0	0	0	0
SLL	Z	0101	110X	1	0	0	0	0	1	0	0	0	1
SRL	Z	0110	1110	1	0	0	0	0	1	0	0	0	1
SRA	Z	0111	1111	1	0	0	0	0	1	0	0	0	1
LW	—	1000	0000	1	1	0	0	0	0	0	0	0	1
SW	—	1001	0000	0	X	1	0	0	0	0	0	0	1
LHB *	—	1010	0000	1	0	0	0	0	0	0	0	0	?
LLB *	—	1011	0000	1	0	0	0	0	0	0	0	0	?
B	—	1100	XXXX	0	X	0	1	0	0	0	0	0	X
CALL	—	1101	XXXX	1	0	0	0	0	0	0	1	0	X
RET	—	1110	XXXX	0	X	0	0	0	0	0	0	1	X
HLT	—	1111	XXXX	0	X	0	0	1	0	0	0	0	X



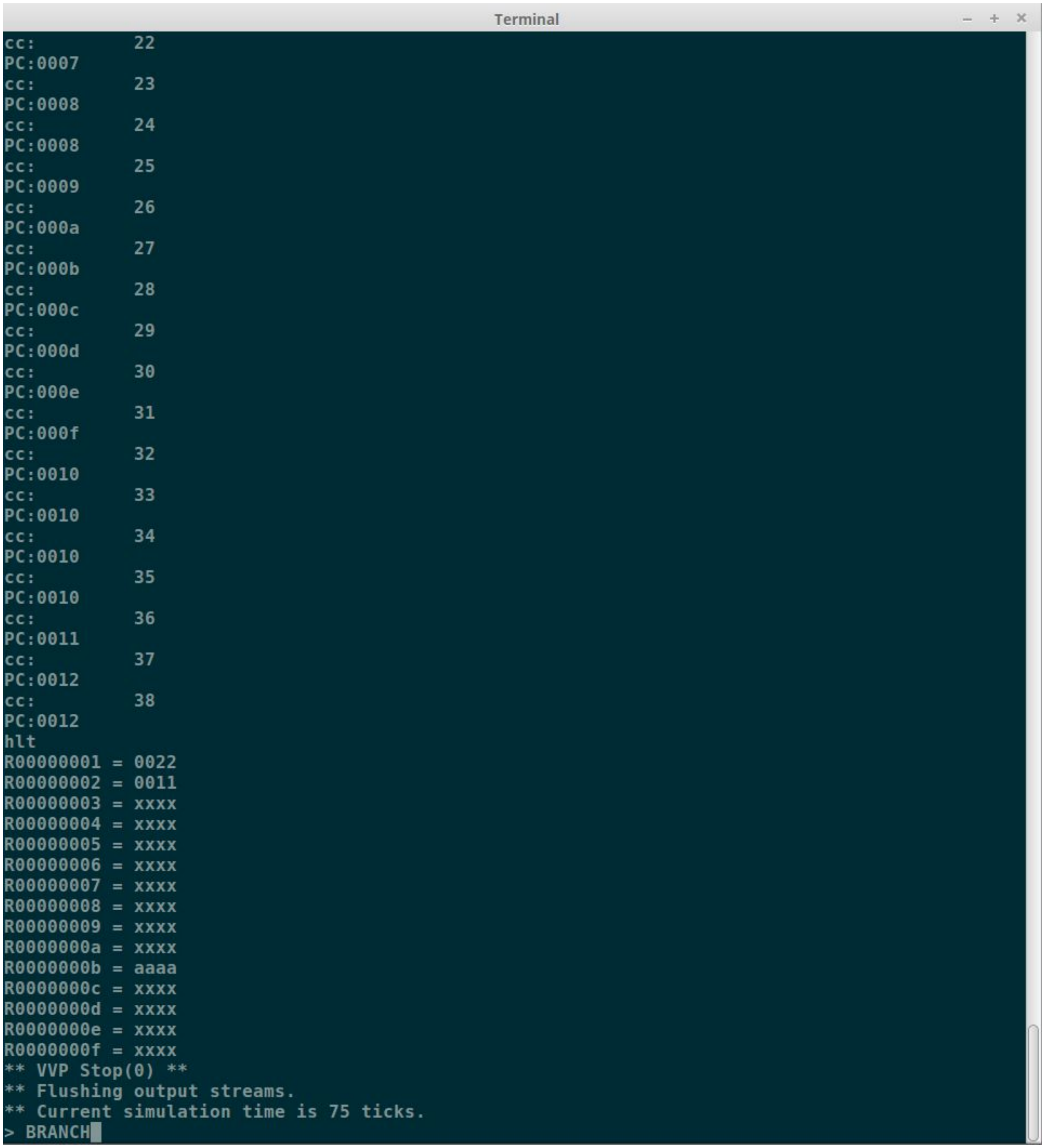
```
cc:          16
PC:0006
cc:          17
PC:0006
cc:          18
PC:0006
cc:          19
PC:0007
cc:          20
PC:0008
cc:          21
PC:0008
cc:          22
PC:0008
cc:          23
PC:0008
cc:          24
PC:0009
cc:          25
PC:000a
cc:          26
PC:000a
cc:          27
PC:000a
cc:          28
PC:000a
cc:          29
PC:000b
cc:          30
PC:000c
cc:          31
PC:000d
cc:          32
PC:000d
hlt
R00000001 = 0000
R00000002 = 0000
R00000003 = 0000
R00000004 = 0000
R00000005 = xxxx
R00000006 = xxxx
R00000007 = xxxx
R00000008 = xxxx
R00000009 = xxxx
R0000000a = xxxx
R0000000b = xxxx
R0000000c = xxxx
R0000000d = xxxx
R0000000e = 0000
R0000000f = xxxx
** VVP Stop(0) **
** Flushing output streams.
** Current simulation time is 63 ticks.
> DATA DEP
```



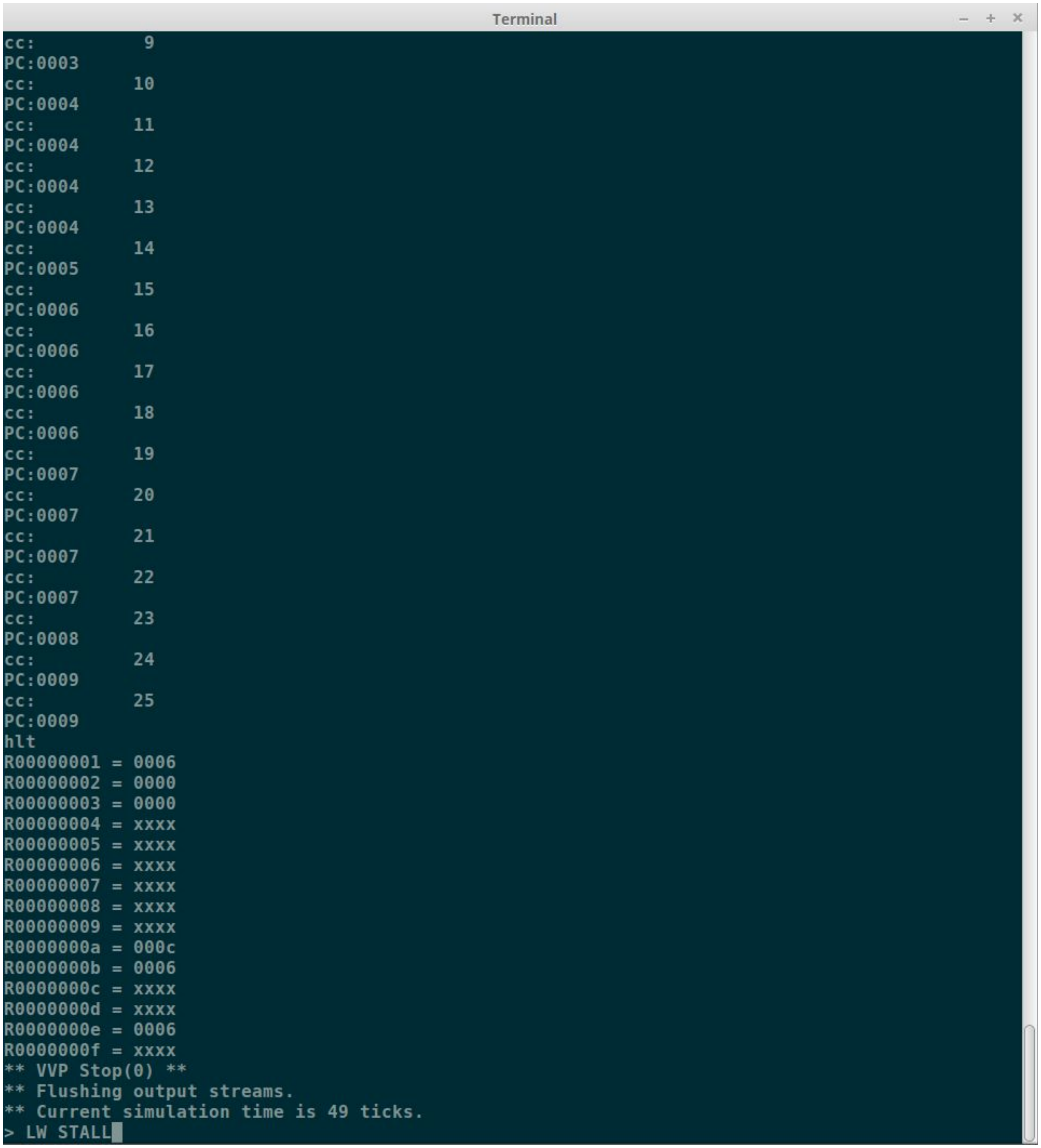
```
cc:        2052
PC:0005
cc:        2053
PC:0006
cc:        2054
PC:0007
cc:        2055
PC:0004
cc:        2056
PC:0005
cc:        2057
PC:0006
cc:        2058
PC:0007
cc:        2059
PC:0004
cc:        2060
PC:0005
cc:        2061
PC:0006
cc:        2062
PC:0007
cc:        2063
PC:0004
cc:        2064
PC:0005
cc:        2065
PC:0006
cc:        2066
PC:0007
cc:        2067
PC:0008
cc:        2068
PC:0008
hlt
R00000001 = 0000
R00000002 = 0001
R00000003 = xxxx
R00000004 = xxxx
R00000005 = xxxx
R00000006 = xxxx
R00000007 = xxxx
R00000008 = xxxx
R00000009 = xxxx
R0000000a = xxxx
R0000000b = xxxx
R0000000c = xxxx
R0000000d = xxxx
R0000000e = xxxx
R0000000f = xxxx
** VVP Stop(0) **
** Flushing output streams.
** Current simulation time is 4135 ticks.
> LOOP
```



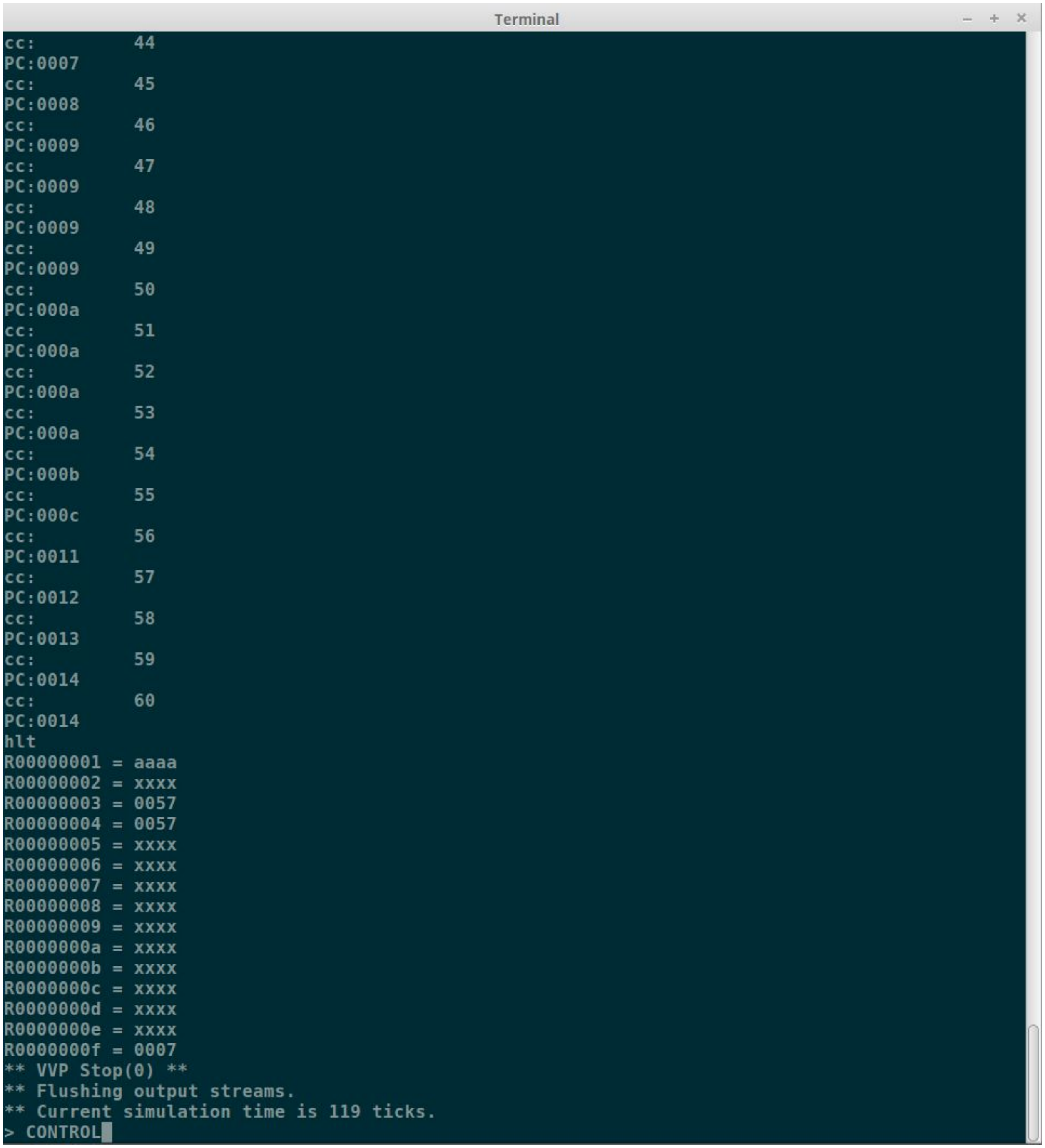
```
cc:         38
PC:0012
cc:         39
PC:0013
cc:         40
PC:0016
cc:         41
PC:0017
cc:         42
PC:0018
cc:         43
PC:0018
cc:         44
PC:0018
cc:         45
PC:0018
cc:         46
PC:0019
cc:         47
PC:001a
cc:         48
PC:0012
cc:         49
PC:0013
cc:         50
PC:0014
cc:         51
PC:0015
cc:         52
PC:0016
cc:         53
PC:0017
cc:         54
PC:0017
hlt
R00000001 = aaaa
R00000002 = 7fff
R00000003 = 7fff
R00000004 = 7eff
R00000005 = feff
R00000006 = 0100
R00000007 = 7f80
R00000008 = 07ef
R00000009 = 07ef
R0000000a = eff0
R0000000b = eff0
R0000000c = xxxx
R0000000d = xxxx
R0000000e = xxxx
R0000000f = 0012
** VVP Stop(0) **
** Flushing output streams.
** Current simulation time is 107 ticks.
> BASIC OP
```



```
cc:          22
PC:0007
cc:          23
PC:0008
cc:          24
PC:0008
cc:          25
PC:0009
cc:          26
PC:000a
cc:          27
PC:000b
cc:          28
PC:000c
cc:          29
PC:000d
cc:          30
PC:000e
cc:          31
PC:000f
cc:          32
PC:0010
cc:          33
PC:0010
cc:          34
PC:0010
cc:          35
PC:0010
cc:          36
PC:0011
cc:          37
PC:0012
cc:          38
PC:0012
hlt
R00000001 = 0022
R00000002 = 0011
R00000003 = xxxx
R00000004 = xxxx
R00000005 = xxxx
R00000006 = xxxx
R00000007 = xxxx
R00000008 = xxxx
R00000009 = xxxx
R0000000a = xxxx
R0000000b = aaaa
R0000000c = xxxx
R0000000d = xxxx
R0000000e = xxxx
R0000000f = xxxx
** VVP Stop(0) **
** Flushing output streams.
** Current simulation time is 75 ticks.
> BRANCH
```



```
cc:          9
PC:0003
cc:         10
PC:0004
cc:         11
PC:0004
cc:         12
PC:0004
cc:         13
PC:0004
cc:         14
PC:0005
cc:         15
PC:0006
cc:         16
PC:0006
cc:         17
PC:0006
cc:         18
PC:0006
cc:         19
PC:0007
cc:         20
PC:0007
cc:         21
PC:0007
cc:         22
PC:0007
cc:         23
PC:0008
cc:         24
PC:0009
cc:         25
PC:0009
hlt
R00000001 = 0006
R00000002 = 0000
R00000003 = 0000
R00000004 = xxxx
R00000005 = xxxx
R00000006 = xxxx
R00000007 = xxxx
R00000008 = xxxx
R00000009 = xxxx
R0000000a = 000c
R0000000b = 0006
R0000000c = xxxx
R0000000d = xxxx
R0000000e = 0006
R0000000f = xxxx
** VVP Stop(0) **
** Flushing output streams.
** Current simulation time is 49 ticks.
> LW STALL
```



```
cc:          44
PC:0007
cc:          45
PC:0008
cc:          46
PC:0009
cc:          47
PC:0009
cc:          48
PC:0009
cc:          49
PC:0009
cc:          50
PC:000a
cc:          51
PC:000a
cc:          52
PC:000a
cc:          53
PC:000a
cc:          54
PC:000b
cc:          55
PC:000c
cc:          56
PC:0011
cc:          57
PC:0012
cc:          58
PC:0013
cc:          59
PC:0014
cc:          60
PC:0014
hlt
R00000001 = aaaa
R00000002 = xxxx
R00000003 = 0057
R00000004 = 0057
R00000005 = xxxx
R00000006 = xxxx
R00000007 = xxxx
R00000008 = xxxx
R00000009 = xxxx
R0000000a = xxxx
R0000000b = xxxx
R0000000c = xxxx
R0000000d = xxxx
R0000000e = xxxx
R0000000f = 0007
** VVP Stop(0) **
** Flushing output streams.
** Current simulation time is 119 ticks.
> CONTROL
```