CYCLE	INST RETIRED	REASON
1	HALT (invalid)	RST signal Os latched
2	HALT (invalid)	RST signal Os latched
3	HALT (invalid)	1st invalid halt
4	HALT (invalid)	2nd invaid halt
5	HALT (invalid)	3rd invalid halt
6	HALT (invalid)	4th invalid halt
7	LBI r0, 0	1st instruction
8	LBI r5, 43	2nd instruction
9	LBI r6, 43	3rd instruction
10	LBI r7, 43	4th instruction
11	LD r1, r0, 0	5th instruction
12	NOP	RAW on r1 from LD/ST so we wait until LD is retired
13	NOP	
14	NOP	
15	ST r5, r1, 0	hazardous LD retired
16	LD r1, r0, 2	7th instruction
17	NOP	RAW on r1 from LD/ST so we wait until LD is retired
18	NOP	
19	NOP	
20	ST r6, r1, 1	hazardous LD retired
21	LD r1, r0, 4	9th instruction
22	NOP	RAW on r1 from LD/ST so we wait until LD is retired
23	NOP	
24	NOP	
25	ST r7, r1, 1	hazardous LD retired
26	HALT	end of prgrm
27		

Full pipeline analysis is on pages

↓ 2-3 ↓

	F	D	E	M	WB
1	LB1	~~Invalid~~	~~Invalid~~	~~Invalid~~	~~Invalid~~
2	LB1	LB1	~~Invalid~~	~~Invalid~~	~~Invalid~~
3	LB1	LB1	LB1	~~Invalid~~	~~Invalid~~
4	LB1	LB1	LB1	LB1	~~Invalid~~
5	LD	LB1	LB1	LB1	LB1
6	ST	LD	LB1	LB1	LB1
7	LD	ST	LD	LB1	LB1
8	LD	ST	NOP	LD	LB1
9	LD	ST	NOP	NOP	LD
10	LD	ST	NOP	NOP	NOP
11	ST	LD	ST	NOP	NOP
12	LD	ST	LD	ST	NOP
13	LD	ST	NOP	LD	ST
14	LD	ST	NOP	NOP	LD

15	LD	ST	NOP	NOP	NOP
16	ST	LD	ST	NOP	NOP
17	LD	ST	LD	ST	NOP
18	LD	ST	NOP	LD	ST
19	LD	ST	NOP	NOP	LD
20	HALT	ST	NOP	NOP	NOP
21	HALT	HALT	ST	NOP	NOP
22	HALT	HALT	HALT	ST	NOP
23	HALT	HALT	HALT	HALT	ST
24	HALT	HALT	HALT	HALT	HALT
25					
26					

some explanations:

1) The 4 LBI instructions execute normally as there are no data hazards
2) First LD executes normally
3) First ST arrives in decode stage, prompting our hazard detection unit to catch the data hazards on r1
4) ST is stalled in decode with fetch stage frozen
5) hazardous LD propogates through and retires in WB, writing data to hazardous register
6) steps 3-5 repeat twice more due to RAW data hazards on r1
7) HALT retires in WB halting the processor

- Total cycle count: 27 with resets
- Total instructions: 11
- CPI = 2.5
- NOTE: we use a valid bit to invalidate HALTS generated on reset so our program can begin