Cycle	Instruction Retired	Reason
1	N/A	1st Instruction in F
2	N/A	2nd Instruction in F
3	N/A	3rd Instruction in F
4	N/A	4th Instruction in F
5	LBI R0, 0	There are no RAW data dependencies between the first 5 instructions so they can all complete uninterrupted with no stalls
6	LBI R5, 43	
7	LBI R6, 43	
8	LBI R7, 43	
9	LD R1, R0, 0	
10	NOP	There is a RAW conflict between the LD and ST instructions (Read from R1 after writing to it) so we stall two cycles until the LD instruction is in Writeback before ST can start its decode. This is possible because register bypassing is implemented.
11	NOP	
12	ST R5, R1, 0	ST completes 2 cycles later as it had to be stalled.
13	LD R1, R0, 2	LD is only reading from R0 so it can be completed immediately after the previous instruction.
14	NOP	There is a RAW conflict between the LD and ST instructions (Read from R1 after writing to it) so we stall two cycles until the LD instruction is in Writeback before ST can start its decode. This is possible because register bypassing is implemented.
15	NOP	
16	ST R6, R1, 1	ST completes 2 cycles later as it had to be stalled.
17	LD R1, R0, 4	LD is only reading from R0 so it can be completed immediately after the previous instruction.
18	NOP	There is a RAW conflict between the LD and ST instructions (Read from R1 after writing to it) so we stall two cycles until the LD instruction is in Writeback before ST can start its decode. This is possible because register bypassing is implemented.
19	NOP	
20	ST R7, R1, 1	ST completes 2 cycles later as it had to be stalled.
21	HALT	Halt has no data interactions so it proceeds normally right after the previous instruction.