Computer Systems Coursework 1
Anyi Guo
Msc. Data Science, student ID: 13154855
02/2019

The function F is defined as$F(1)=F(2)=F(3)=1$ and for $n \geq 3$,
$F(n+1) = F(n) \cdot (F(n-1) + F(n-2))$

i.e., the $(n + 1)$th value is given by the product of the nth value and the sum of the $(n - 1)$th and $(n - 2)$th values.

(a) Write an assembly program for computing the kth value F(k), where k is an integer bigger than 3 read from a memory location M, and storing F(k) at memory location

Answer:

```
1. LOAD r3, M <- M stores number k
2. LOAD r0, #1
3. LOAD r1, #1
4. LOAD r2, #1
5. SUB r3, #1 <- this is the counter
6. ADD r4, r0, r1
7. MUL, r4, r4, r2
8. LOAD r0, r1
9. LOAD r1, r2
10. LOAD r2, r4
11. BNE I5, r3, #3 <- go back to Instruction 5 if r3 != 3
12. STOR M, r4
```

(b) Consider a pipelined processor, where the pipeline stages are F (fetch), D (decode), R (register read), E (execute) and W (write back). Describe what happens in the pipeline stages for the various types (data movement, data processing, control) of instructions

Answer

- All instructions go through IF (instruction is loaded onto the CPU) and ID (control unit decodes the instruction).
- Using memory addressing doesn't need reading registers, so I1 skips RR.
- Using immediate addressing doesn't need reading registers or data transfer, so I2, I3 and I4 skips R and EX.
- I8, I9 and I10 needs to read register but requires no data transfer, so they skip EX.
- We assume that data transfer between main memory and the MBR of the CPU happens during EX, and that the I11 (BNE condition) is checked during EX too.
- Finally, the data is copied from the MBR into the target memory during WB.
- Arithmetic instructions need RR, EX and WB.

(c) Show the execution of your program on the above pipelined processor for k = 5 by drawing a diagram. Assume that the fetched and decoded instructions are stored in an instruction window IW with a capacity of 12 instructions, and that there is no resource conflict between fetching instructions and executing data transfer instructions. Explain where and why delay slots appear.

For k = 5

Assumptions:

1. Out-of-order jumps allowed - instructions which jump out of order are **bolded**
2. No conflict between FI and executing data transfer instructions -> FI and LOAD data instructions can happen at the same time
3. I11 (BNE) is checked in Ex window

	F	**D**	**IW**	**R**	**X**	**W**	**Notes**
1	I1						
2	I2	I1					
3	I3	I2			I1		I1 is in X because it is loaded from Main memory. Data bus busy
4	I4	I3	I2			I1	r3 =5. I2 is in IW because W is busy writing I1
5	I5	I4	I3			I2	I2 skips R, E
6	I6	I5	I4			I3	I3 skips R,E
7	I7	I6		I5		I4	I4 skips R,E
8	I8	I7		I6	I5		
9	I9	I8	I7		I6	I5	r3 = 4
10	I10	I9	I7	**I8**		I6	r4 not ready -> I7 stays in IW. I8 jumps out of order
11	I11	I10	I9	I7		I8	
12	I12	I11	I10	I9	I7		
13		I12	I10		**I11**	I7	I8 skips X (disappears in this row but reappears in the next row). I11 jumps out-of order, and the condition is not true, so we go back to instruction 5

	F	D	IW	RR	X	W	
14	I5			I10		I9	I12 discarded. I9 skips X
15	I6	I5				I10	I10 skips X
16	I7	I6		I5			
17	I8	I7		I6	I5		
18	I9	I8	I7		I6	I5	r3 = 3
19	I10	I9	I7	**I8**		I6	r4 not ready -> I7 stays in IW
20	I11	I10	I9	I7		I8	
21	I12	I11	I10	I9	I7		
22		I12	I10	**I11**		I7	I9 skips X (disappears in this row but reappears in the next row). r4 not ready -> I10 stays in IW
23			I12	I10	I11	I9	Condition met, exit the loop
24				I12		I10	
25					I12		

Delay slots:

- In cycle 10 & 19, I7 has to wait in IW because r4 is not ready yet.
- In cycle 13 & 22, I10 has to wait because r4 is not ready (as I7 is updating r4)

2. A computer has a cache, main memory and a hard disk. If a referenced word is in the cache, it takes 15 ns to access it. If it is in main memory but not in the cache, it takes 85 ns to load (the block containing) it into the cache (this includes the time to originally check the cache), and then the reference lookup is started again. If the word is not in main memory, it takes 10 ms to load (the block containing) it from the disk into main memory, and then the reference lookup is started again. The cache hit ratio is 0.4. In the case of a cache miss, the probability that the word is in the main memory is 0.7. Compute the average load time.

Answer

Assume 100 searches in total:

40 searches are in cache. 60 not in cache - out of which 42 are found in main memory, 18 found in hard disk.

Avg load time = [40 * 15 + 42 * (85+15) + 18* (10^6+85+15)] / 100 *ns*

= (600 + 4200 + 18 * 10^ 6) / 100 *ns* = 180066 *ns*