Lab B3

TT0L - GROUP 0

Person 1	1111111111
Person 2	1111111111

---

1. Write ARM instructions to store the following block of data (32 bit words) in memory locations starting from 0x6000 to 0x6014. Subsequently, copy the data to the locations 0x7000 to 0x7014.
   Data (H): 0x01, 0x02, 0x03, 0x04, 0x05, 0x06

```
1           MOV     R2, #0x6000 ;source address
2           MOV     R1, #1
3           MOV     R4, #0x0006
4 NEXT      STR     R1, [R2], #0x0004 ;store value of R1 into memory address specified by R2, then increment R2 by 4
5           ADD     R1, R1, #0x0001 ;add value of R1 and #0x1
6           SUBS    R4, R4, #0x0001 ;sub with conditional flag (carry)
7           CMP     R4, #0x0000 ;compare
8           BNE     NEXT ;jump to next if both R4 and #0x0000 are not equal
9
10          MOV     R2, #0x6000 ;source address
11          MOV     R3, #0x7000 ;destination address
12          MOV     R4, #0x0006
13 loop
14          LDR     R5, [R2], #4 ;get value from memory and load into register, then increment memory address by 4
15          STR     R5, [R3], #4 ;store value from register into memory, then increment memory address by 4
16          SUBS    R4, R4, #0x01 ;sub with conditional flag (carry)
17          CMP     R4, #0x00 ;compare
18          BNE     loop ;jump to loop if both R4 and #0x00 are not equal
19          END
```

Start address: 0x6000 End address: 0x7014

Word Address	Byte 3	Byte 2	Byte 1	Byte 0	Word Value
0x6000	0x0	0x0	0x0	0x1	0x1
0x6004	0x0	0x0	0x0	0x2	0x2
0x6008	0x0	0x0	0x0	0x3	0x3
0x600C	0x0	0x0	0x0	0x4	0x4
0x6010	0x0	0x0	0x0	0x5	0x5
0x6014	0x0	0x0	0x0	0x6	0x6
0x6018	0x0	0x0	0x0	0x0	0x0
0x7000	0x0	0x0	0x0	0x1	0x1
0x7004	0x0	0x0	0x0	0x2	0x2
0x7008	0x0	0x0	0x0	0x3	0x3
0x700C	0x0	0x0	0x0	0x4	0x4
0x7010	0x0	0x0	0x0	0x5	0x5
0x7014	0x0	0x0	0x0	0x6	0x6

2. Modify Question 1 to transfer the data to the locations 0x7000 to 0x7014 in the reverse order.
(E.g. the data byte 06H at location 0x6014 should be stored at location 0x7000.)

```
1           MOV       R2, #0x6000 ;source address
2           MOV       R1, #1
3           MOV       R4, #0x0006
4 NEXT      STR       R1, [R2], #0x0004 ;store value of R1 into memory address specified by R2, then increment R2 by 4
5           ADD       R1, R1, #0x0001 ;add value of R1 and #0x1
6           SUBS      R4, R4, #0x0001 ;sub with conditional flag (carry)
7           CMP       R4, #0x0000 ;compare
8           BNE       NEXT ;jump to next if both R4 and #0x0000 are not equal
9
10          MOV       R2, #0x6000 ;source address
11          MOV       R3, #0x7000 ;destination address
12          MOV       R4, #0x0006
13          ADDS      R3, R3, #0x14 ;add #0x14 into #0x7000 (R3) with carry
14 loop
15          LDR       R5, [R2], #4 ;get value from memory and load into register, then increment memory address by 4
16          STR       R5, [R3], #-4 ;store value from register into memory, then decrement memory address by 4
17          SUBS      R4, R4, #0x01 ;sub with conditional flag (carry)
18          CMP       R4, #0x00 ;compare
19          BNE       loop ;jump to loop if both R4 and #0x00 are not equal
20          END
```

Start address: 0x6000 End address: 0x7014

Word Address	Byte 3	Byte 2	Byte 1	Byte 0	Word Value
0x6000	0x0	0x0	0x0	0x1	0x1
0x6004	0x0	0x0	0x0	0x2	0x2
0x6008	0x0	0x0	0x0	0x3	0x3
0x600C	0x0	0x0	0x0	0x4	0x4
0x6010	0x0	0x0	0x0	0x5	0x5
0x6014	0x0	0x0	0x0	0x6	0x6
0x6018	0x0	0x0	0x0	0x0	0x0
0x6FFC	0x0	0x0	0x0	0x0	0x0
0x7000	0x0	0x0	0x0	0x6	0x6
0x7004	0x0	0x0	0x0	0x5	0x5
0x7008	0x0	0x0	0x0	0x4	0x4
0x700C	0x0	0x0	0x0	0x3	0x3
0x7010	0x0	0x0	0x0	0x2	0x2
0x7014	0x0	0x0	0x0	0x1	0x1

Word Value Format Dec Hex Memory Map Key Instructions Data

3. A bus that connects major computer components (processor, memory, I/O) is called a system bus.

   I. Define the function of the system bus.

   - A bus that connects major computer components

   ii. List and describe the THREE major modules of the system bus.

   Data Bus

   - Data lines that provide a path for moving data among system modules

   Address Bus

   - Used to designate the source or destination of the data on the data bus

   Control Bus

   - Used to control the access and the use of the data and address lines

4. Assume a three-stage pipeline (fetch, execute and write). Draw a timing diagram to show how many units are needed for three instructions

   .

	1	2	3	4	5
Instruction 1	FETCH	EXECUTE	WRITE		
Instruction 2		FETCH	EXECUTE	WRITE	
Instruction 3			FETCH	EXECUTE	WRITE

   5 units are needed for three instructions

5. Assume that a processor employs a memory address register (MAR), a memory buffer register (MBR), a program counter (PC), and an instruction register (IR). List the sequence of events of the instruction cycle (fetch cycle.)

   The PC contains the address of the next instruction to be fetched. This address is moved to the MAR and placed on the address bus. The control unit requests a memory read, and the result is placed on the data bus and copied into the MBR and then moved to the IR. Meanwhile, the PC is incremented by 1, preparatory for the next fetch.

6. List and describe the SIX status flags of an Intel 8086 microprocessor.

- Carry flag (CF)
  - Carry is generated when performing n bit operations and the result is more than n bits, then this flag is set (1), otherwise, it's cleared (0).
- Parity flag (PF)
  - If after any arithmetic or logical operation the result has even parity, an even number of 1 bit, the parity register is set (1), otherwise it's cleared (0)
- Auxiliary carry flag (AF)
  - AF is set (1) if there is a carry-out from the low nibble into the high nibble or a borrow-in from the high nibble into the low nibble of the lower byte in a 16-bit word; otherwise, AF is cleared (0)
- Zero flag (ZF)

    - ZF is set (1) if the result of an instruction is zero, otherwise, ZF is cleared (0)
- Sign flag (SF)
    - The MSB of the result is copied into SF. Thus, SF is set (1) is the result is a negative number or cleared (0) if it is positive.
- Overflow flag (OF)
    - When OF is set, it indicates that the signed result is out of range. If the result is not out of range, OF remains reset

7. Assume there is a four-stage instruction pipeline - Fetch (F), Decode (D), Execute (E) and Write (W) running in a microprocessor. Assume that each stage requires one time unit and no branch instruction is involved.
    I. Based on the answer in (i), how many time units are needed to complete these Six instructions with pipelining?

	1	2	3	4	5	6	7	8	9
INS 1	F	D	E	W					
INS 2		F	D	E	W				
INS 3			F	D	E	W			
INS 4				F	D	E	W		
INS 5					F	D	E	W	
INS 6						F	D	E	W

T with pipeline = [4+(6-1)] time unit
= 9 time unit

II. By using formula, calculate the total time required to execute SIX instructions without pipelining.

$$T_{1,n} = nk\tau$$

T without pipeline = 6*4 time unit
= 24 time unit

III. Calculate the speedup factor for the same number of instructions.

nk/k+(n−1) = 24/9 time unit
= 2.67