Lab B-05 : Programming Exercise 3

Student's name :

1) JASMYNE YAP (2424C244PT)

2) NUR ALEEZ DANIA BT MOHD SHAHRUL AZMAN (2424C244QB)

1. MOV R2, #0x6000

MOV R3, #0x7000

MOV R4, #6

MOV R1, #1

LOOP1

STR R1, [R2], #4

ADD R1, R1, #1

SUB R4, R4, #1

CMP R4, #0

BNE LOOP1

```
        ;       Initialize memory addresses and counter loop

        MOV     R2, #0X6000     ; R2 = source memory address
        MOV     R3, #0X7000     ; R3 = destination memory address
        MOV     R4, #6      ; R4 = loop counter (6 iteration)

        ;       LOOP1 : upload data to memory

        MOV     R1, #1      ; R1 = initial value to store

LOOP1

        STR     R1, [R2], #4
        ADD     R1, R1, #1
        SUB     R4, R4, #1
        CMP     R4, #0
        BNE     LOOP1

        ;       reinitialize registers for second loop

        MOV     R2, #0X6000     ; reset R@ to source address
        MOV     R4, #6

        ;       LOOP2 : transfer data to new memory area

LOOP2

        LDR     R5, [R2], #4
        STR     R5, [R3], #4
        SUB     R4, R4, #1
        CMP     R4, #0
        BNE     LOOP2

        ;       END OF PROGRAM
        ;       INFINITE LOOP TO END PROGRAM

        END
```

R0	0x0
R1	0x7
R2	0x6018
R3	0x7018
R4	0x0
R5	0x6
R6	0x0
R7	0x0
R8	0x0
R9	0x0
R...	0x0
R...	0x0
R...	0x0
R...	0xFF000000
LR	0x0
PC	0x48

MOV R2, #0x6000

MOV R4, #6

LOOP2

LDR R5, [R2], #4

STR R5, [R3], #4

SUB R4, R4, #1

CMP R4, #0

BNE LOOP2

END

View Memory Contents

Start address: 0x100 End address: 0x10000

Word Address	Byte 3	Byte 2	Byte 1	Byte 0	Word Value
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
0x7018	0x0	0x0	0x0	0x0	0x0

Word Value Form... Dec Hex Memory Map K... Instructi... Data

2. Modify Q1 to transfer the data to the locations 0x7000 to 0x7014 in the reverse order

```
; Initialise memory and counters
        MOV     R2, #0x6000
        MOV     R3, #0x7000
        MOV     R4, #6          ; R4 = loop counter (6 values to store)
        MOV     R1, #0x01       ; R1 = data value to store, starting from 1
LOOP1
        STR     R1, [R2], #0x04 ; store R1 at address in R2, then R2 += 4
        ADD     R1, R1, #0x01   ; Increment R1 for next value
        SUB     R4, R4, #0x01   ; Decrement loop counter
        CMP     R4, #0x00       ; Check if counter is zero
        BNE     LOOP1           ; If not zero, repeat loop

        MOV     R2, #0x6000     ; Reset R2 to start of source memory
        ADD     R2, R2, #0x14   ; Move R2 to last data address (0x6014)
        MOV     R4, #6          ; Reset loop counter to 6
LOOP2
        LDR     R5, [R2], #-4   ; Load value from R2, then R2 -= 4 ; 6014-4 = 6010
        STR     R5, [R3], #0x04 ; Store value to R3, then R3 += 4 ; 7004
        SUB     R4, R4, #0x01   ; Decrement loop counter
        CMP     R4, #0x00       ; Check if done
        BNE     LOOP2           ; If not done, repeat loop

        END
```

```
1           MOV         R2, #0x6000     ;
2           MOV         R3, #0x7000     ;
3           MOV         R4, #6          ;
4           MOV         R1, #0x01
5
6 LOOP1
7
8           STR         R1, [R2], #0x04 ;
9           ADD         R1, R1, #0x01   ;
10          SUB         R4, R4, #0x01   ;
11          CMP         R4, #0x00
12          BNE         LOOP1           ;
13
14          MOV         R2, #0x6000     ;
15          ADD         R2, R2, #0x14   ;
16          MOV         R4, #6          ;
17
18 LOOP2
19          LDR         R5, [R2], #-4   ;
20          STR         R5, [R3], #0x04 ;
21          SUB         R4, R4, #0x01   ;
22          CMP         R4, #0x00
23          BNE         LOOP2           ;
24
25          END
26
```

R0	0	Dec	Bin	Hex
R1	7	Dec	Bin	Hex
R2	24572	Dec	Bin	Hex
R3	28696	Dec	Bin	Hex
R4	0	Dec	Bin	Hex
R5	1	Dec	Bin	Hex
R6	0	Dec	Bin	Hex
R7	0	Dec	Bin	Hex
R8	0	Dec	Bin	Hex
R9	0	Dec	Bin	Hex
R10	0	Dec	Bin	Hex
R11	0	Dec	Bin	Hex
R12	0	Dec	Bin	Hex
R13	-16777216	Dec	Bin	Hex
LR	0	Dec	Bin	Hex
PC	84	Dec	Bin	Hex

View Memory Contents

Start address: 0x100 End address: 0x10000

Memory ...

Word Address	Byte 3	Byte 2	Byte 1	Byte 0	Word Value
0x5FFC	0x0	0x0	0x0	0x0	0x0
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	0x6	0x6

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

The system bus acts as the central communication pathway, enabling data transfer and control signals between the CPU, memory and I/O devices

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

① Data bus : It consists of multiple data lines that serve at pathways for transferring data between system modules such as the processor, memory and I/O devices. The width of the data bus commonly 32, 64, 128 bits or more refers to the number of lines it contains, which directly determines how many bits can be transferred simultaneously. A wider data bus allows for more data to be moved at once, making it a crucial factor on determining the overall performance of the system.

② Address Bus : It is used to specify the source or destination of data on the data bus. When the processor wants to read data from memory, it places the address of the desired word onto the address lines. The width of the address bus determines the maximum memory capacity the system can support. The address bus is used to address I/O parts with the higher order bits selecting the specific module (like memory or I/O) and the lower-order bits identifying the exact location or port within that module.

③ Control bus : It is responsible for managing access to and usage of the shared data and address lines in a computer system. Since these lines are used by multiple components, the control bus ensures orderly communication by transmitting control signals. These signals include both command information (such as read or write operations) and timing information, which indicates when data and addresses are valid and should be acted upon. The control bus is essential for coordinating the actions of system modules and maintaining proper data flow.

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

	T_1	T_2	T_3	T_4	T_5
Instruction 1	F	E	W		
Instruction 2		F	E	W	
Instruction 3			F	E	W

5 units ※

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

① Carry Flag (CF) - Indicates an overflow condition for unsigned arithmetic operations. It is set if there is a carry out of the most significant bit in an addition or a borrow into the most significant bit in a subtraction.

② Parity Flag (PF) - Indicates the parity of the result of an arithmatic or logical operation. It is set if the number of set bits in the least significant byte of the result is even.

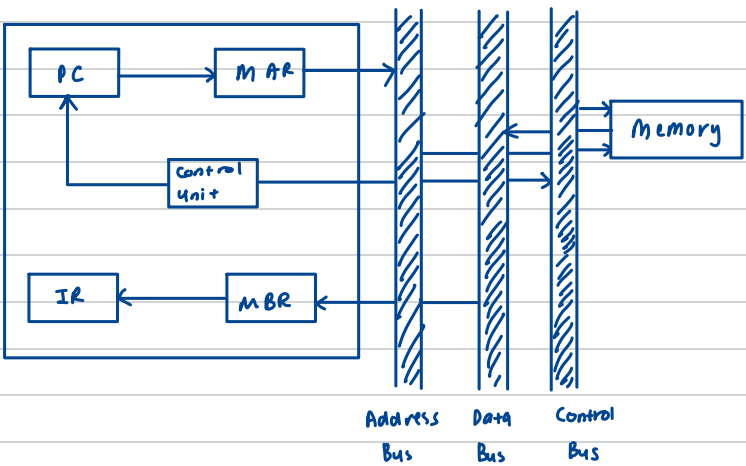
③ Auxiliary Carry Flag (AF) - Set if there is a carry out of a bit 3 into bit 4 in an arithmetric operation, which indicates an overflow in the nibble (4 bits) of the byte.

④ Zero Flag (ZF) - Set if the result of an arithmetric or logical operation is zero.

⑤ Sign Flag (SF) - Set if the most significant bit of the result is 1.

⑥ Overflow Flag (OF) - Set if there is a signed overflow in arithmetric operation, which means the result is too large to be represented in the destination opperand.

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.

7. Assume there is a four-stage instruction pipeline - Fetch (F), Decode (D), Execute (E) and write (W) running in 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?

$T = [k + (n-1)]$

T with pipelining = $[4 + (6-1)] \times 1$ time unit = 9 time unit.

ii) By using formula, calculate the total time required to execute six instructions without pipeline

$T = nkT$

T without pipelining = $4 * 6 * 1$ time unit
= 24 time unit

iii) Calculate the speedup factor for the same number of instructions.

$$\text{speedup factor} = \frac{nk}{k + (n-1)}$$

$$= \frac{24}{9}$$

$$= 2.67$$