Subject: 1

Year. Month. Date. ( )

سعید حاجی خواه 9231066 تمرین 2

1- به جای MOV AX, 0 می توان از AND AX, 0 و به جای CMP AX, 0 می توان از TEST AX, 0 استفاده کرد.

2-

Code	AX	BX	CX	DX	SP	BP	SI	DI	CF	ZF	SF	OF	PF	AF
	0710	0000	014D	0000	0100	0000	0000	0000	0	0	0	0	0	0
MOV BL, 02FH	0710	002F	014D	0000	0100	0000	0000	0000	0	0	0	0	0	0
MOV CL, 2	0710	002F	0102	0000	0100	0000	0000	0000	0	0	0	0	0	0
SAL BL, CL	0710	00BC	0102	0000	0100	0000	0000	0000	0	0	1	1	0	0
MOV AL, 2AH	072A	00BC	0102	0000	0100	0000	0000	0000	0	0	1	1	0	0
MOV CH, 056H	072A	00BC	5602	0000	0100	0000	0000	0000	0	0	1	1	0	0
ADD AL, CH	0780	00BC	5602	0000	0100	0000	0000	0000	0	0	1	1	0	1
DEC AL	077F	00BC	5602	0000	0100	0000	0000	0000	0	0	0	1	0	1
NEG AL	0781	00BC	5602	0000	0100	0000	0000	0000	1	0	1	0	1	1
SBB AL, 3EH	0742	00BC	5602	0000	0100	0000	0000	0000	0	0	0	1	1	1
XOR BL, BL	0742	0000	5602	0000	0100	0000	0000	0000	0	1	0	0	1	1
MOV [SI], BL	0742	0000	5602	0000	0100	0000	0000	0000	0	1	0	0	1	1

3- اشکال اول دستور SCASB است در این دستور محتوای ES:[DI] با AL مقایسه می شود ولی حالت ما مقایسه با رشته دیگر است پس باید از CMPSB استفاده شود. همچنین مقایسه باید با شرط برابر انجام گیرد پس باید REPNE به REPE تغییر یابد.

2000

Subject: 2

Year. Month. Date. ( )

4-1- شرط که در اینجا با برسی (N.E) است برقرار نباشد.

2- مقدار موجود در cx برابر صفر شود.

5- LEA آدرس NUMBDS را درون BX قرار می دهد ولی MOV مقدار آن را.

6- اگر فرض کنیم MEMWDS در آدرس 1234H تعریف شده باشد که بصورت زیر خواهد بود:

81 1E	34 12	00 20
Opcode	آدرس MEMWDS	2000H

MEMWDS ← MEMWDS - (2000H + CF)

این دستور مقدار 2000H را از MEMWDS کم خواهد کرد، چنین اگر CF برابر یک باشد، یک واحد دیگر از مقدار قبل کم خواهد شد و در MEMWDS ذخیره خواهد شد.

7- MR: خواندن MOV AL,[4040H]

MR: خواندن [4040H] و قرار دادن در AL

MR: خواندن TMP [BX]

MR: خواندن [BX] و قرار دادن در IP

MR: خواندن SHR WORD PTR[SI], 1

MR: خواندن WORD PTR[SI]

IDLE: محاسبه مقدار شیفت یافته

MW: نوشتن محاسبه در WORD PTR[SI]

2000

MR_4_7 : خواندن STOSW

MW : نوشتن AX در ES:[DI]

IDLE : افزایش یا کاهش DI (با توجه به مقدار پرچم DF

البته این دو به طراحی بستگی دارد. اگر ثبات ها

پایه ی INC و DEC داشته باشند این عمل می تواند

در مرحله قبل انجام گیرد و نیازی به ALU نباشد

MR_5 : خواندن IN AX, FFH

IOR : خواندن مقدار از IO و قرار دادن آن در AX

MR_6 : خواندن CALL [DI]

MW : نوشتن اطلاعات لازم در پشته

MR : خواندن [DI] و قرار دادن آن در IP

MR_7 : خواندن SUB BYTEPTR[BX], CH

MR : خواندن BYTEPTR[BX]

IDLE : محاسبه حاصل تفریق

MW : نوشتن حاصل در BYTEPTR[BX]

MR_8 : خواندن RET 8

MR : خواندن اطلاعات لازم از پشته و نوشتن آدرس بازگشت در IP

IDLE : جمع کردن عدد 2+8 با SP و نوشتن حاصل در SP

MR_9 : خواندن PUSH DX

MR : خواندن اطلاعات از پشته و نوشتن آن ها در DX

IDLE : جمع کردن SP با عدد 2 ، در صورتی که چیزی پاپ نشود

برای SP در نظر گرفته شده باشد می توان این کار را

همزمان با عمل قبل انجام داد و نیازی به این چرخه

نخواهد بود.

MOV CX, WORD PTR[BP+20H] -8

PUSHER_PROC PROC NEAR -9

MOV DX, 0B000H

IN AL, DX

TEST AL, 10H

JZ PUSHER_END

POP BX

;POP CX

PUSH AX

;PUSH CX

PUSH BX

PUSHER_END:

RET

PUSH CX, POP CX در خط
صورتی که PROC از نوع FAR
باشد باید اجرا شوند زیرا اگر
PROC از نوع NEAR باشد
فقط IP هنگام اجرای CALL به
پشته اضافه می شود ولی در صورتی
که FAR باشد علاوه بر IP،
CS هم به پشته اضافه می شود

MOV AL, 10 ; n is stored in AL -10

MOV BL, AL

MOV CL, 5

MOV DX 0

ADD_START:

CMP CL, BL

JG ADD_END

MOV AL, CL

MUL CL

ADD DX, AX

ADD CL, 5

JMP ADD_START

ADD_END:

MOV AX, DX

2000

11- اعداد در DS در متغیری به نام ARR تعریف شده اند و تعداد آن ها برابر متغیری به نام LEN است.

```
MAIN_LOOP:
    MOV CX, 1
    LEA SI, ARR
    FOR_LOOP_START:
        ; for loop condition check
        CMP CL, LEN
        JGE FOR_LOOP_END
        INC CL
        ; for loop body
        MOV DL, BYTE PTR[SI]
        INC SI
        CMP DL, BYTE PTR[SI]
        JG FOR_LOOP_START
        INC CH
        MOV DH, BYTE PTR[SI]
        MOV BYTE PTR[SI-1], DH
        MOV BYTE PTR[SI], DL
        JMP FOR_LOOP_START
    FOR_LOOP_END:

    CMP CH, 0
    JNZ MAIN_LOOP
```