معماری تمرین شماره ۹ - مهدی نیک‌نژاد - ۹۶۳۰۰۳۹

پردازنده ۱۶ بیتی ← ۱۰ بیت آدرس حافظه

MM ← 2^{10} ردیف ۲ بایتی

مجموعه ثبات‌هایی که داریم ← AX انباشتگر، BX، CX، DX ← ۴ تا

$\log_2 4 = \boxed{2}$ بیت

۲ بیت برای کد کردن ثباتها کافی است.

MOV <immediate - 8bit> — انتقال داده‌ی بلافصل ۸ بیتی به ثبات انباشتگر

ADD <immediate - 8bit> — جمع کردن داده‌ی ۸ بیتی با ثبات انباشتگر

ADD <memory address> — جمع کردن داده‌ی موجود در آدرس مورد نظر از حافظه با ثبات انباشتگر

SUB <register1>, <register2> — تفریق ۲ ثبات از یکدیگر

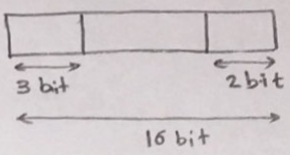
PUSH <register> — انتقال محتوای ثبات به داخل پشته Stack

POP <register> — انتقال محتوای پشته به داخل ثبات مورد نظر

۱ - قالب دستورالعمل ←

۶ دستور داریم ← پس $\boxed{3}$ بیت برای op-code نیاز است.

$2^3 = 8 > 6$



$IR[12] = r$

* این اختلاف را در بیت‌ها گذاشتم تا این ۲ تا مدل ADD از هم تفکیک بشوند!

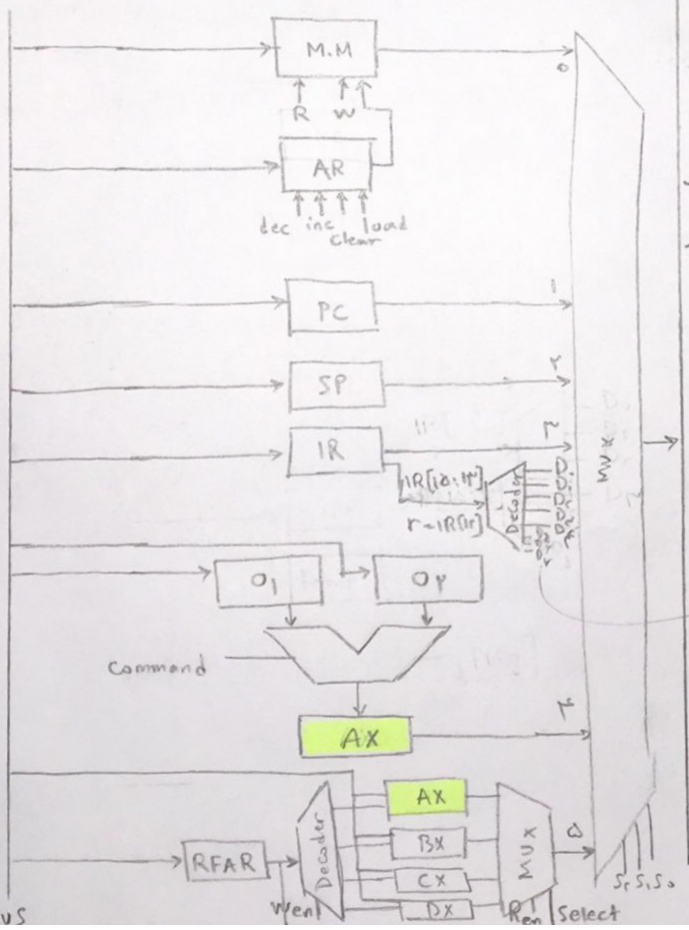
قالب آدرس

ISA	۱۵	۱۴	۱۳	۱۲	۱۱	۱۰	۹	۸	۷	۶	۵	۴	۳	۲	۱	۰	
MOV	0	0	0	X	X	X	X	X	i	i	i	i	i	i	i	i	→ D_0
ADD_i	0	0	1	0	X	X	X	X	i	i	i	i	i	i	i	i	→ D_1
ADD_M	0	0	1	1	X	X	a	a	a	a	a	a	a	a	a	a	→ ۱۰ تا برای آدرس حافظه
SUB	0	1	0	X	X	X	X	X	X	X	X	X	R_1	R_1	R_2	R_2	
PUSH	0	1	1	X	X	X	X	X	X	X	X	X	X	X	R	R	
POP	1	0	0	X	X	X	X	X	X	X	X	X	X	X	R	R	

ADD با مملوئه آدرس حافظه ←

3bit op-code + 10bit آدرس = 13 bit

۱۳ بیت نیاز دارد، پس ما همان یک word از حافظه را می‌خوانیم یعنی ۱۶ بیت.



۲ -

* توجه: پایه‌های load، Clear، inc (increment) و dec (decrement) را برای همه رسم نمی‌کنیم ولی همه دارند.

* توجه: اینجا برای دستور MOV نیاز نیست که یک ثبات temp هم بگیریم چون باید داده‌ی موجود در ثبات را در ثبات انباشتگر AX بریزیم

* توجه: اینجا می‌توانیم برای دستور SUB، از یک ثبات IR استفاده کنیم، نه ۲ تا!

* مطمئن نیستم که می‌شود آیا حاصل عملیات با ALU را مستقیم به ثبات آکومولاتور ریخت یا نه؟

به تعداد دستورالعمل‌های متمایز

در اینجا ADD را یکبار گذاشتیم، یک بیت اضافه برای مشخص کردن نوع ADD گذاشتیم!

* مطمئن نیستم ۲ تا AX در این شکل درسته یا نه؟ شاید باید جایی RFAR را صفر کنم ولی نمی‌دانم؟؟

Scanned with CamScanner

۳- دیزعملیات ها ←

* دستور ADD IMMEDIATE، نیازی نبود از RFAR استفاده کنیم!

Instruction Fetch (IF):

T_0 : AR ← PC

T_1 : IR ← M[AR], PC ← PC+1

Instruction Decode :

T_2 : Decode IR[15:12] into D_0, D_1, D_2, D_3, D_4 , r=IR[11]

MOV :

$T_3 D_0$: AX ← IR[7:0] , SC ← 0

ADD IMMEDIATE :

$T_3 D_1 r'$: O_1 ← AX

$T_4 D_1 r'$: O_2 ← IR[7:0]

$T_5 D_1 r'$: AX ← $O_1 + O_2$, SC ← 0

ADD MEMORY ADDRESS :

$T_3 D_1 r$: AR ← IR[9:0]

$T_4 D_1 r$: O_1 ← M[AR]

$T_5 D_1 r$: O_2 ← AX

$T_6 D_1 r$: AX ← $O_1 + O_2$, SC ← 0

SUB :

$T_3 D_2$: RFAR ← IR[3:2]

$T_4 D_2$: O_1 ← RF[RFAR]

$T_5 D_2$: RFAR ← IR[1:0]

$T_6 D_2$: O_2 ← RF[RFAR]

$T_7 D_2$: AX ← $O_2 - O_1$, SC ← 0

PUSH :

$T_3 D_3$: RFAR ← IR[1:0]

$T_4 D_3$: AR ← SP

$T_5 D_3$: M[AR] ← RF[RFAR], SP ← SP-1, SC ← 0

POP :

$T_3 D_4$: RFAR ← IR[1:0] , SP ← SP+1

$T_4 D_4$: AR ← SP

$T_5 D_4$: RF[RFAR] ← M[AR] , SC ← 0

$T_3 D_1 r'$: RFAR ← AX

$T_4 D_1 r'$: O_1 ← RF[RFAR]

$T_5 D_1 r'$: RFAR ← IR[7:0]

$T_6 D_1 r'$: O_2 ← RF[RFAR]

$D_7 D_1 r'$: AX ← $O_1 + O_2$, SC ← 0

ظاهراً این غلط!

تفاوت

$T_3 D_1 r$: AR ← IR[9:0]

$T_4 D_1 r$: O_1 ← M[AR]

$T_5 D_1 r$: RFAR ← AX

$T_6 D_1 r$: O_2 ← RF[RFAR]

$T_7 D_1 r$: AX ← $O_1 + O_2$, SC ← 0

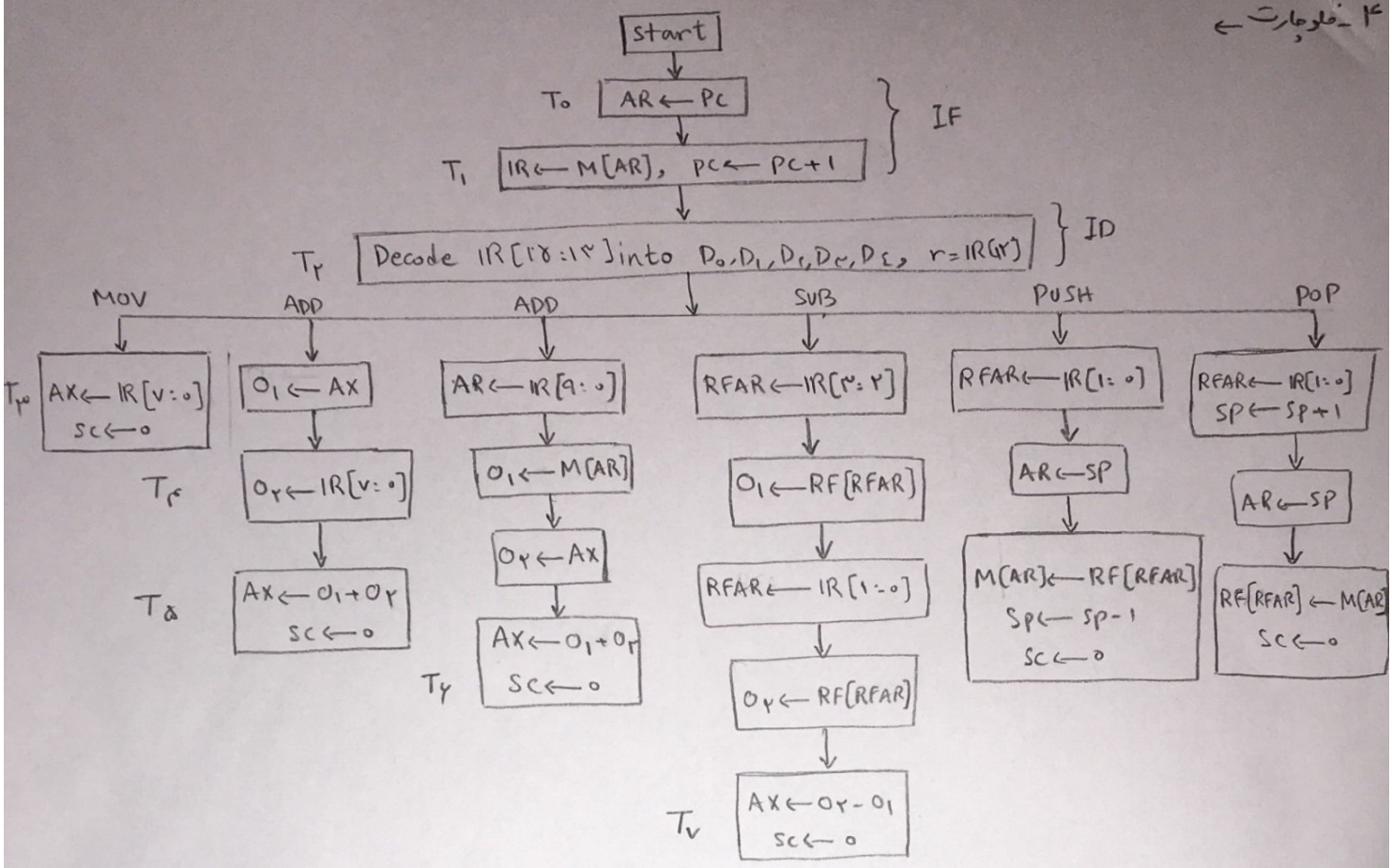
ظاهراً این غلط!

تفاوت

اینها را اصلی دادم درستش یا نه ولی باز فرق درستش چی هست چی دل کردم!

Scanned with CamScanner

۴- فلوچارت ←



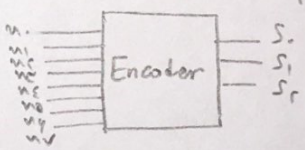
۵-

طولانی ترین دستورالعمل مربوط به دستور SUB است با ۷ تا clock ,

کوتاه ترین هم مربوط به MOV است با ۳ تا clock .

۶- کنترل سیم بندی ←

AR را اینجا نمی کشیم؟



register	x_7	x_6	x_5	x_4	x_3	x_2	x_1	x_0	S_2	S_1	S_0
MM	0	0	0	0	0	0	0	1	0	0	0
PC	0	0	0	0	0	0	1	0	0	0	1
SP	0	0	0	0	0	1	0	0	0	1	0
IR	0	0	0	0	1	0	0	0	0	1	1
AX	0	0	0	1	0	0	0	0	1	0	0
RF	0	0	1	0	0	0	0	0	1	0	1

طرف راست
ریزعملیات

$$x_0 = T_1 + T_4 D_1 r + T_5 D_4$$

$$x_1 = T_0$$

$$x_2 = T_4 D_3 + T_4 D_4$$

$$x_3 = T_3 D_0 + T_4 D_1 r' + T_3 D_1 r + T_3 D_2 + T_5 D_2 + T_3 D_3 + T_3 D_4$$

$$x_4 = T_3 D_1 r' + T_5 D_1 r$$

$$x_5 = T_4 D_2 + T_6 D_2 + T_5 D_3$$

Scanned with CamScanner

ALU Command : $T_5 D_1 r' + T_4 D_1 r + T_7 D_2$

SP decrement : $T_5 D_3$

SP increment : $T_3 D_4$

sp load : $T_4 \longrightarrow$ ؟!

sc clear : $T_3 D_0 + T_5 D_1 r' + T_4 D_1 r + T_7 D_2 + T_5 D_3 + T_5 D_4$

MM read : $T_1 + T_4 D_1 r + T_5 D_4$

MM write : $T_5 D_3$

AR Load : $T_0 + T_3 D_1 r + T_4 D_3 + T_4 D_4$

PC increment : T_1

IR load : T_1

O_1 load : $T_3 D_1 r' + T_4 D_1 r + T_4 D_2$

O_2 load : $T_4 D_1 r' + T_5 D_1 r + T_4 D_2$

AX Load : $T_3 D_0 + T_5 D_1 r' + T_4 D_1 r + T_7 D_2$

RFAR Load : $T_3 D_2 + T_5 D_2 + T_3 D_3 + T_3 D_4$

Decoder enable (write en) : $T_5 D_4$

* Mux enable (read en) : $T_3 D_4 + T_4 D_4 + T_5 D_4$

Scanned with CamScanner