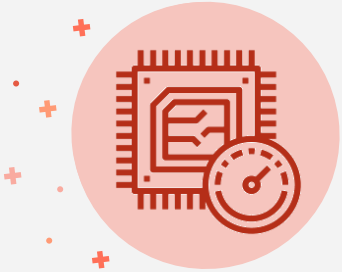


سیستم‌های دیجیتال ۲

اسلاید ۲

سازمان و طراحی یک کامپیوتر پایه

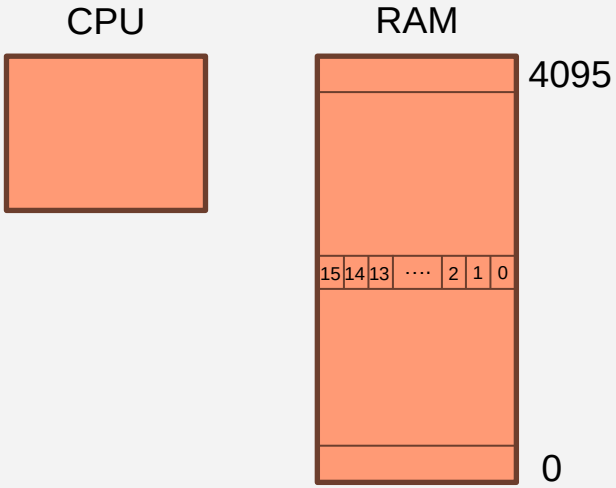
محمدرضا رازقی‌زاده

رئوس مطالب

Instruction Codes	کدهای دستورالعمل
Computer Registers	ثبات‌های کامپیوتر
Computer Instructions	دستورالعمل‌های کامپیوتر
Timing and Control	زمان‌بندی و کنترل
Instruction Cycle	سیکل دستورالعمل
Memory Reference Instructions	دستورالعمل‌های ارجاع به حافظه
Input-Output and Interrupt	ورودی – خروجی و وقفه
Design of Basic Computer	طراحی کامپیوتر پایه
Design of Accumulator Logic	طراحی مدار منطقی انباره

- هر پردازنده طراحي خاص خود (ثبات ها، گذرگاه ها، ريزعمل ها، دستورالعمل‌هاي ماشين و ...) را دارد.
- كامپيوترهاي مدرن ساختار پيچيده‌اي دارند كه شامل موارد زير هستند:
  - ✓ ثبات هاي فراوان
  - ✓ چندين واحدهاي محاسباتي هم براي اعداد صحيح هم براي اعداد ممیزدار
  - ✓ استفاده از چندين واحد خط لوله تا به اين ترتيب سرعت اجرا افزايش يابد.
  - ✓ و موارد ديگر
- در ادامه براي فهم اينكه كامپيوتر چگونه كار مي كند از يك مدل ساده شده استفاده شده است. اين مدل را آقاي مانو) (Manoمعرفي كرده و نام آن را كامپيوتر پايه گزارده است. اين مدل شبيه كامپيوترهايي است كه ۳۰سال پيش كار مي كرده اند.
- ما از اين مدل براي معرفي سازمان پردازنده و ارتباط RTL با سطح بالاتر پردازنده استفاده مي كنيم.

- حافظه این کامپیوتر ۴۰۹۶ کلمه دارد.
- ۲^{12} = ۴۰۹۶ ، یعني به ۱۲ خط آدرس نیاز داریم.
- هر کلمه ۱۶ بیت طول دارد.



دستورالعمل ها

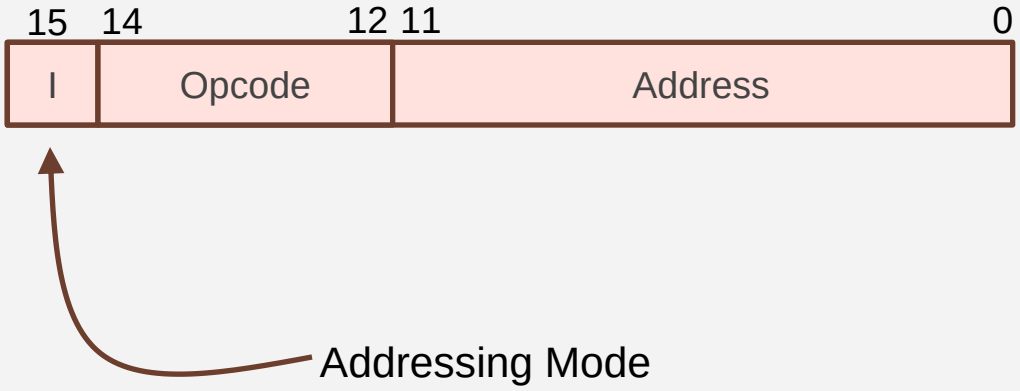
- برنامه
  - ✓ يك دنباله از دستورالعمل ها
- دستورالعمل
  - ✓ يك گروه از بيت ها كه به كامپيوتر اعلام مي كنند كه يك عمل خاص را انجام دهند (يك دنباله از ريزعمل ها).
- دستورالعمل هاي يك كامپيوتر به همراه همه داده هاي لازم در حافظه ذخيره شده اند.
- CPU دستور بعدي را از حافظه مي خواند.
- اين دستور در يك ثبات به نام ثبات دستورالعمل (IR) ذخيره شده است.
- دستورالعمل به دنباله اي از ريزعمل ها تبديل مي شود تا با انجام ريزعمل ها دستورالعمل مورد نظر اجرا شود.

فرمت دستوالعمل ها

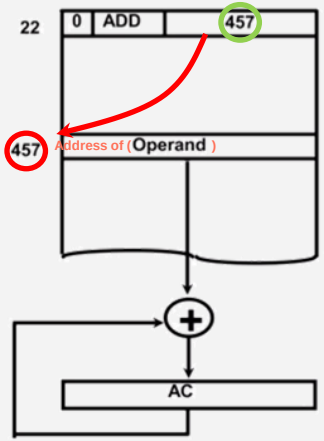
- يك دستورالعمل اغلب از دو بخش تشكيل شده است:
- كد عمليات (opcode): عملي را كه دستورالعمل بايد انجام دهد، مشخص مي كند.
- آدرس (address): ثبات يا جايي از حافظه را كه دستورالعمل بايد عمل كند مشخص مي كند.
- همانطور كه گفتيم در كامپيوتر پايه ۱۲بيت براي آدرس دهي حافظه داريم.
- در كامپيوتر پايه بيت ۱۵ دستورالعمل، مد آدرس دهي (Addressing Mode) را مشخص مي كند.
- صفر: آدرس دهي مستقيم (Direct Addressing)
- يك: آدرس دهي غير مستقيم (Indirect Addressing)
- چون كلمه هاي حافظه و بنابراين دستورالعمل ها ۱۶ بيتي هستند، ۳ بيت باقي‌مانده براي كد دستورالعمل مورد استفاده قرار مي گيرد.

فرمت دستوالعمل ها

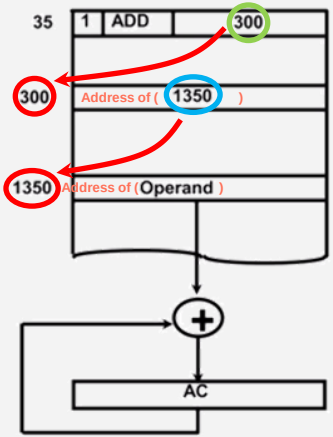
- چون كلمه هاي حافظه و بنابراين دستورالعمل ها ۱۶ بيتي هستند، ۳ بيت باقي‌مانده براي كد دستورالعمل مورد استفاده قرار مي گيرد.



- فيلد آدرس يك دستورالعمل مي تواند به يكي از دو فرمت زير بيان شود:
  - ✓ آدرس دهي مستقيم: آدرس داده مورد نظر در حافظه (آدرس عملوند)
  - ✓ آدرس دهي غير مستقيم: آدرس آدرس داده مورد نظر در حافظه (آدرس آدرس عملوند)



Address of (Operand) = 457



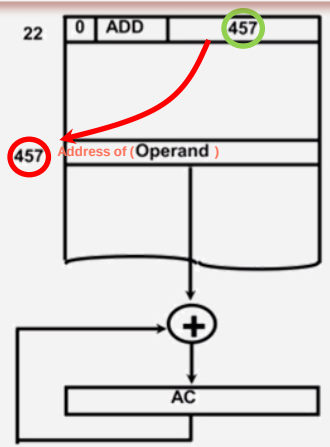
Address of (Operand) = 1350

Address of (1350) = 300

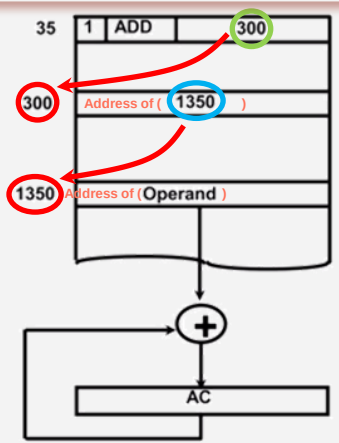
Address of (Address of (Operand)) = 300

آدرس نهايي براي يافتن عملوند، آدرس موثر (EA) نامیده می‌شود. به طور مثال، در آدرس دهی غیرمستقیم شکل سمت راست آدرس موثر ۱۳۵۰ و در آدرس دهی مستقیم شکل سمت چپ آدرس موثر ۴۵۷ است.

ملوند)



Address of (Operand) = 457



Address of (Operand) = 1350

Address of (1350) = 300

Address of (Address of (Operand)) = 300

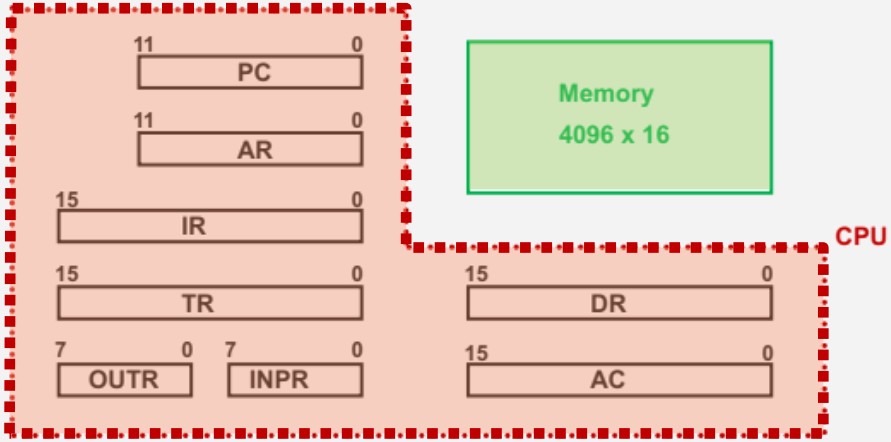
ثبات هاي پردازنده

- هر پردازنده تعداد زيادي ثبات براي نگهداري دستورالعمل‌ها، آدرس‌ها و داده‌ها و ... دارد.
- پردازنده يك ثبات به نام شمارنده برنامه (Program Counter (PC)) دارد كه آدرس دستوري را كه بايد اجرا شود، نگهداری می‌کند.
- ازآنجا که حافظه در کامپیوتر پایه شامل ۴۰۹۶ کلمه می‌شود، پس PC ثباتی ۱۲ بيتي است.
- در آدرس دهي مستقيم يا غير مستقيم پردازنده براي آنكه آدرس عملوند را نگه دارد، از يك ثبات به نام ثبات آدرس (Address register (AR)) استفاده مي كند.
- چون حافظه در کامپیوتر پایه ۴۰۹۶ کلمه دارد پس AR ثباتی ۱۲ بيتي است.
- پس از آنكه عملوند در حافظه پيدا شد، در آدرس دهي مستقيم يا غير مستقيم، عملوند به يك ثبات به نام ثبات داده (Data Register (DR)) منتقل مي شود.

ثبات هاي پردازنده

- كامپيوتر پايه يك ثبات همه منظوره به نام ثبات انباره (Accumulator (AC)) دارد.
- در كامپيوتر پايه از يك ثبات براي نگهداري داده هاي مياني يا موقتي استفاده شده است كه اين ثبات، ثبات موقت (Temporary Register (TR)) ناميده می‌شود.
- كامپيوتر پايه يك مدل بسيار ساده ورودي / خروجي دارد.
  - ✓ دستگاه‌هاي ورودي كاراكترهاي ۸ بيتي را به پردازنده مي‌فرستند.
  - ✓ پردازنده كاراكترهاي ۸ بيتي را به دستگاه‌هاي خروجي مي‌فرستد.
- ثبات ورودي (Input Register (INPR))، داده ۸ بيتي را كه از دستگاه ورودي رسيده
- نگهداری می‌کند.
- ثبات خروجي (Output Register (OUTR))، داده ۸ بيتي را كه به دستگاه خروجي می‌فرستد، نگهداری مي‌كند.

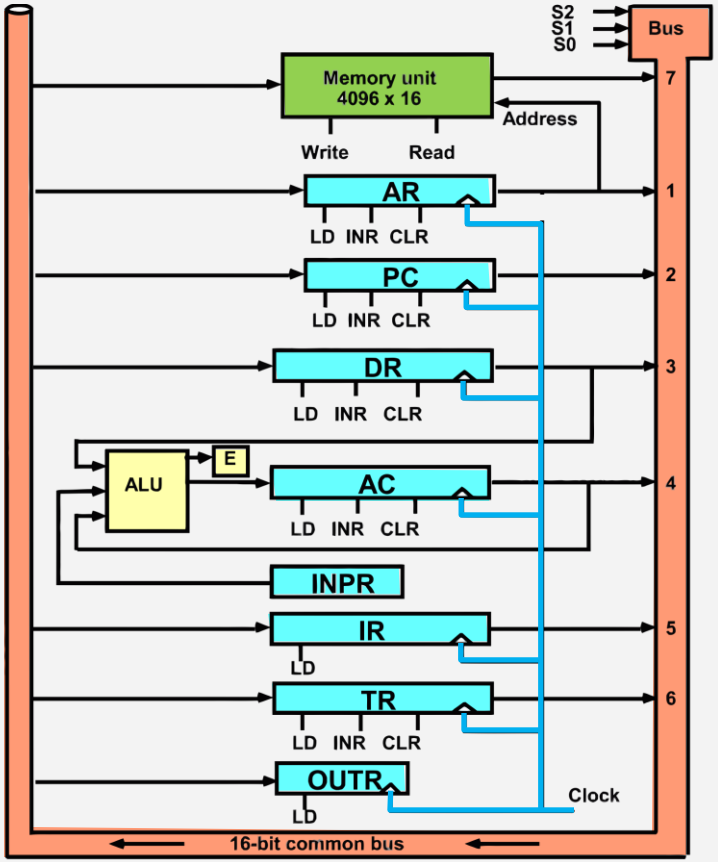
ثبات هاي كامپيوتر پايه



DR	16	Data Register	نگهداری مقدار عملوند
AR	12	Address Register	نگهداری آدرس عملوند
AC	16	Accumulator	ثبات همه منظوره
IR	16	Instruction Register	نگهداری کد عملیات
PC	12	Program Counter	نگهداری آدرس دستورالعمل
TR	16	Temporary Register	نگهداری داده‌های موقت
INPR	8	Input Register	نگهداری کاراکتر ورودی
OUTR	8	Output Register	نگهداری کاراکتر خروجی

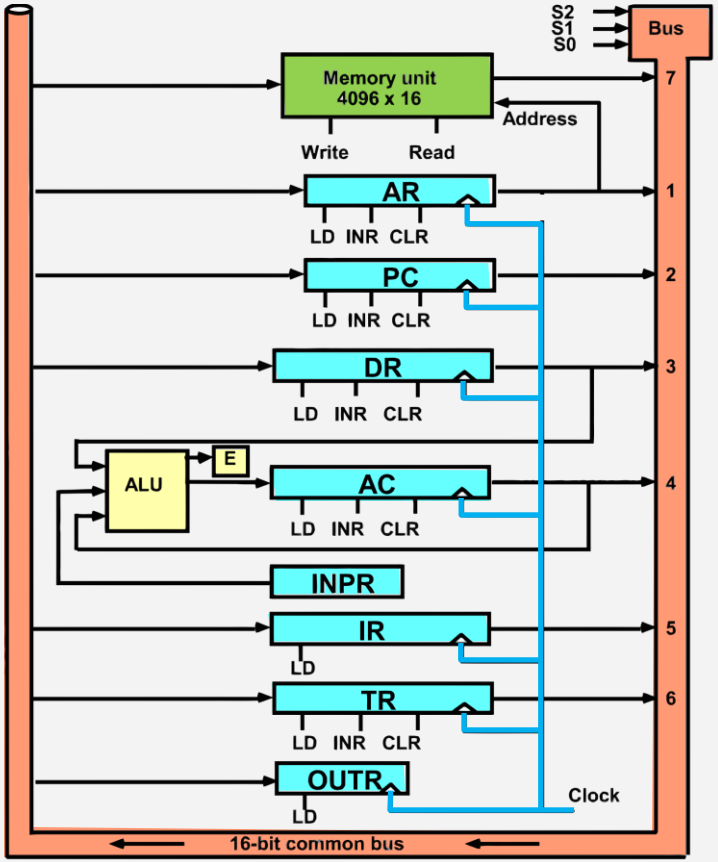
سیستم گذرگاه عمومی

- ثبات‌ها در کامپیوتر پایه با استفاده از گذرگاه به هم متصـل شده‌اند.



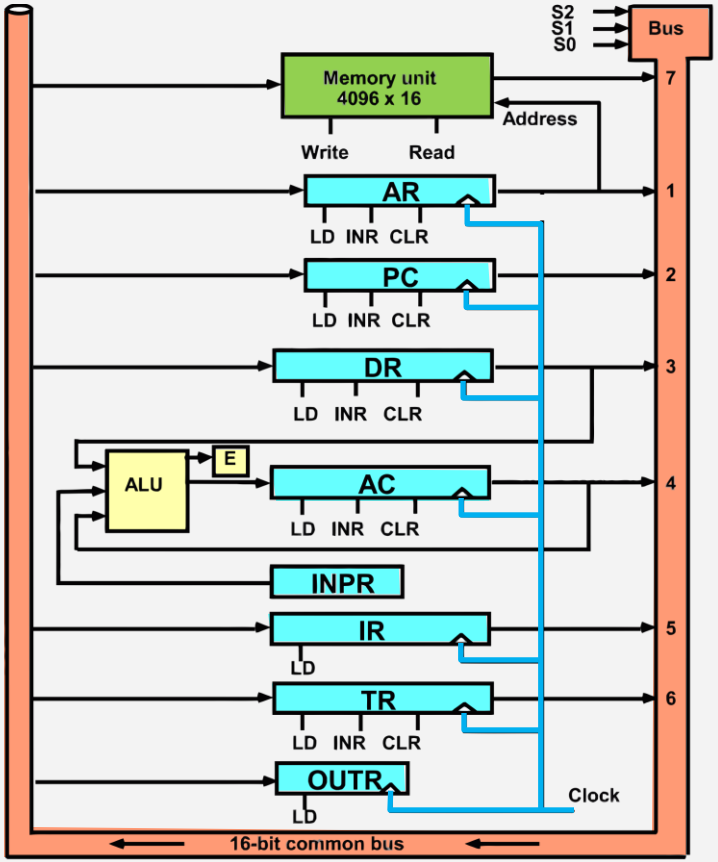
سیستم گذرگاه عمومی

- ✓ توسط سه خط کنترل s۰، s۱ و s۲ انتخاب ثبات ورودي کنترل مي‌شود.



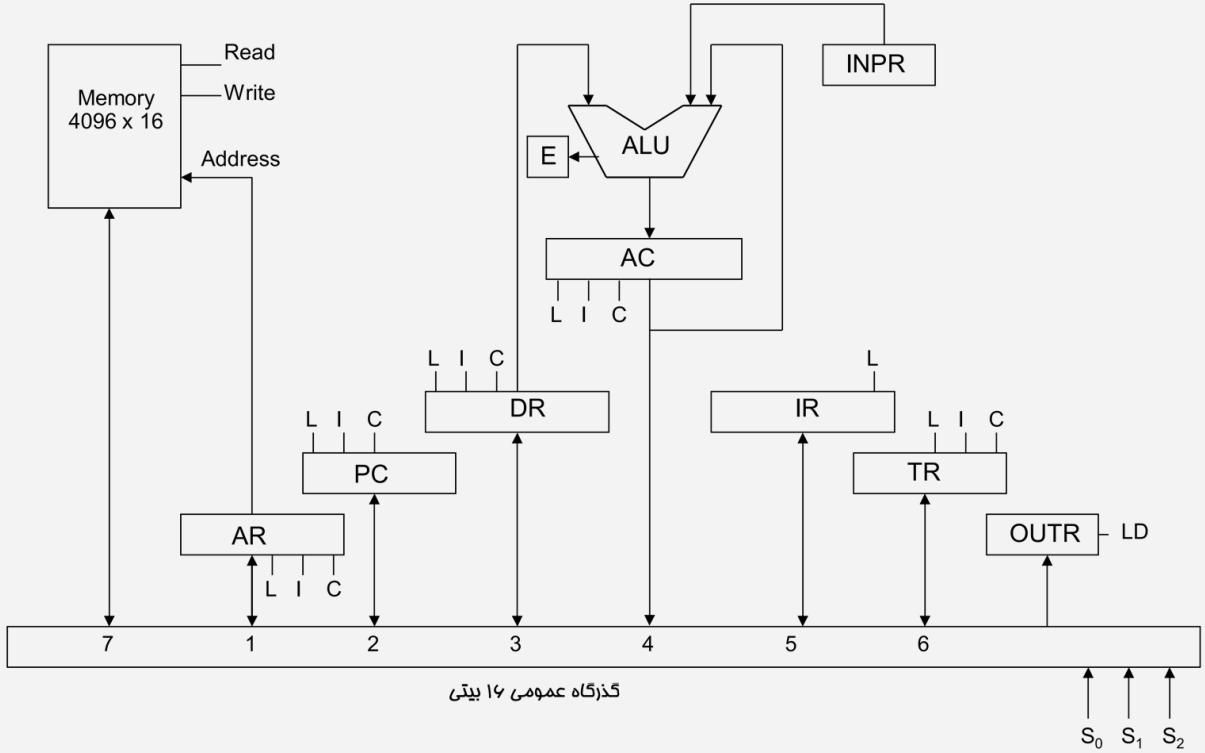
S_2	S_1	S_0	Register
0	0	0	x
0	0	1	AR
0	1	0	PC
0	1	1	DR
1	0	0	AC
1	0	1	IR
1	1	0	TR
1	1	1	Memory

سیستم گذرگاه عمومی



- ✓ براي خروجي داده از گذرگاه مشترک: پايه Read حافظه يا پايه Load ثبات‌ها فعال مي‌شود.
- ✓ وقتي ثبات هاي ۱۲ بيتي (AR و PC) روي گذرگاه اطلاعات قرار مي‌گيرند، ۴ بيت بـا ارزش تـر گـذرگاه مقدار صفر مي گيرد.

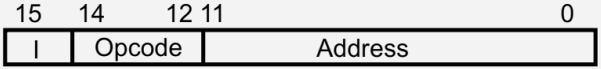
سیستم گذرگاه عمومی



گذرگاه عمومی ۱۶ بیتی

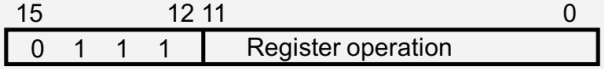
دستورالعمل هاي كامپيوتر پايه

- ✓ فرمت دستوالعمل هاي كامپيوتر پايه



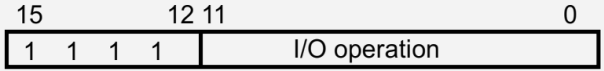
- ✓ دستورالعمل هاي مراجعه به حافظه

(OP-code = 000 ~ 110)



- ✓ دستورالعمل هاي مراجعه به ثبات

(OP-code = ۱۱۱, I = ۰)



- ✓ دستورالعمل هاي ورودي خروجي

(OP-code =111, I = 1)

دستورالعمل هاي كامپيوتر پايه

سمبل	Hex Code I = 0	Hex Code I = 1	توضيح
AND	0xxx	8xxx	AND memory word to AC
ADD	1xxx	9xxx	Add memory word to AC
LDA	2xxx	Axxx	Load AC from memory
STA	3xxx	Bxxx	Store content of AC into memory
BUN	4xxx	Cxxx	Branch unconditionally
BSA	5xxx	Dxxx	Branch and save return address
ISZ	6xxx	Exxx	Increment and skip if zero
CLA	7800		Clear AC
CLE	7400		Clear E
CMA	7200		Complement AC
CME	7100		Complement E
CIR	7080		Circulate right AC and E
CIL	7040		Circulate left AC and E
INC	7020		Increment AC
SPA	7010		Skip next instr. if AC is positive
SNA	7008		Skip next instr. if AC is negative
SZA	7004		Skip next instr. if AC is zero
SZE	7002		Skip next instr. if E is zero
HLT	7001		Halt computer
INP	F800		Input character to AC
OUT	F400		Output character from AC
SKI	F200		Skip on input flag
SKO	F100		Skip on output flag
ION	F080		Interrupt on
IOF	F040		Interrupt off

- هر کامپيوتر بايد مجموعه اي از دستورالعمل‌ها را داشته باشد که کاربر بتواند براي محاسبه هر تابعي که محاسبه پذير بودن آن محرز است، برنامه‌اي به زبان ماشين بنويسد. لذا کامل بودن مجموعه دستورالعمل‌های کامپیوتر بسیار حائز اهمیت می‌باشد.

- انواع دستورالعمل‌ها

✓ دستورالعمل هاي عملياتي

حسابي، منطقي و جابجايي مانند: ADD, CMA, INC, CIR, CIL, AND, CLA

✓ تبادل اطلاعات

تبادل اطلاعا ت با حافظه و ثبات‌هاي کامپيوتر مانند: LDA, STA

✓ دستورالعمل هاي کنترلي

دستورالعملهاي کنترل برنامه و وارسي شرايط مانند: BUN, BSA, ISZ

✓ دستورالعمل هاي ورودي / خروجي

ورودي / خروجي مانند: INP, OUT

واحد کنترل

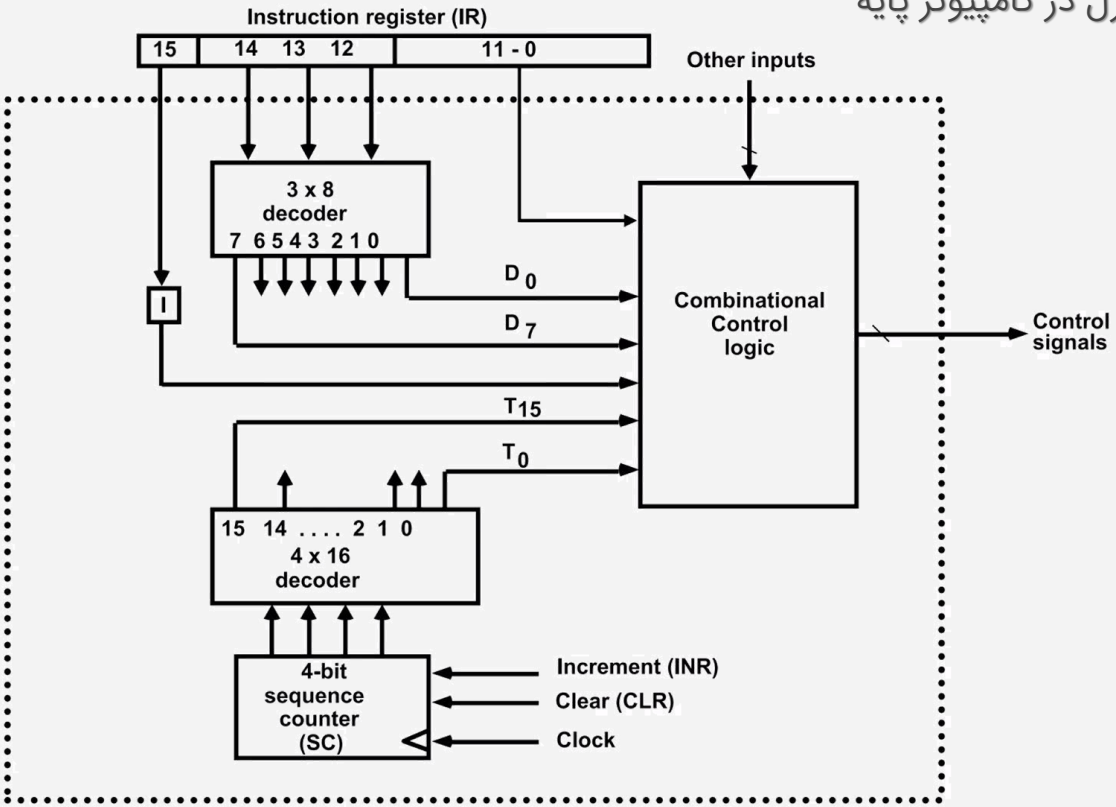
- واحد کنترل همة دستورالعمل هاي ماشين را به سيگنال هاي كنترلي تبديل مي كند. اين سيگنالهاي كنترلي براي كنترل ريزعمل‌ها بكار مي روند.
- واحد کنترل به دو طريق قابل ساخت مي باشد:
  - ✓ کنترل سخت افزاري (Hardwired):

واحد کنترل از مدارهاي تركيبي و ترتيبي ساخته شده است كه كار آنها توليد سيگنال‌هاي كنترلي است.

  - ✓ کنترل ريزبرنامه نويسي شده (Microprogrammed):

يك حافظه كنترلي درپروسسور وجود دارد كه شامل ريزعمل‌هايي است كه سيگنالهای كنترلي لازم را توليد مي كند.

واحد کنترل در کامپیوتر پایه

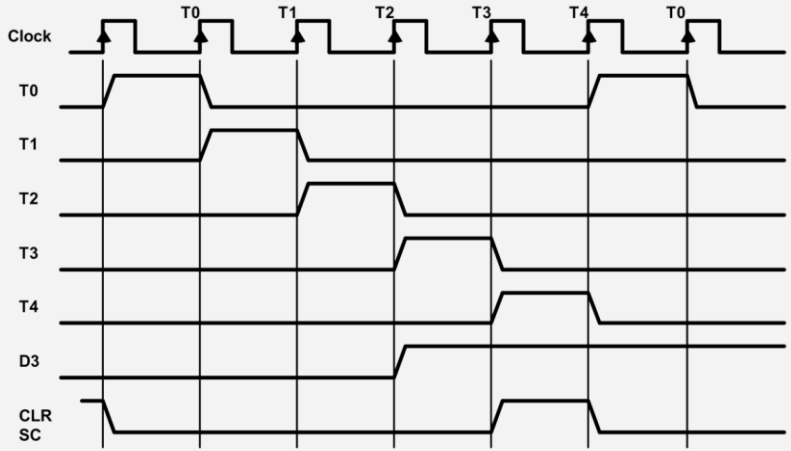


سیگنالهای زمان بندی

- سیگنال هاي زمان بندي، به وسيله دنباله شمار ۴ بيتي و ديكدر ۴*۱۶ توليد مي‌شود.
- SC مي تواند افزايش يافته يا پاك شود.

به عنوان مثال : . . . ,T۰, T۱, T۲, T۳, T۴, T۰, T۱

- فرض: در زمان SC، T۴پاك مي شود اگر خروجي ديكدر D۳ فعال باشد.



D_3T_4: SC ← 0

سیکل دستورالعمل

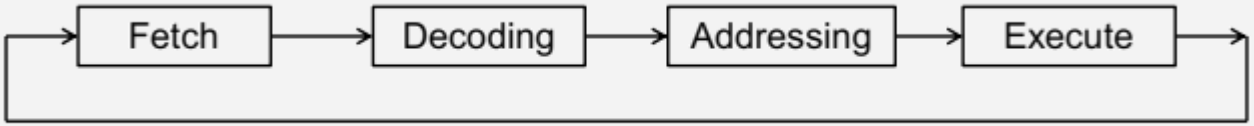
- يك برنامه مجموعه اي از دستورات است كه به طـور متـوالي از حافظـه خوانده و اجرا مي شوند.
- اجراي هر دستور در كامپيوتر پايه شامل مراحل زير است:

۱ ـ واكشي دستورالعمل از حافظه (Fetch)

۲ ـ كدگشايي دستورالعمل (Decoding)

۳ ـ آدرس دهي يا بدست آوردن آدرس موثر (Addressing)

۴- اجرای دستورالعمل (Execute)



- ✓ هر پردازنده‌ای سیکل دستورالعمل خاص خود را دارد.

واكشي

در ابتدا PC با آدرس اولين دستور برنامه پر شده و دنباله شمار (SC) را صفر مي‌شود تا T۰ فعال شود. سپس با هر CP يك واحد به SC اضافه مي شود تا سيگنال هاي T۰تا T۱۵به ترتيب فعال شوند.

- ✓ ريزعمل هايي كه در مرحله fetch انجام مي شوند:

$$T_0 : AR \leftarrow PC$$

$$T_1 : IR \leftarrow M[AR] , PC \leftarrow PC + 1$$

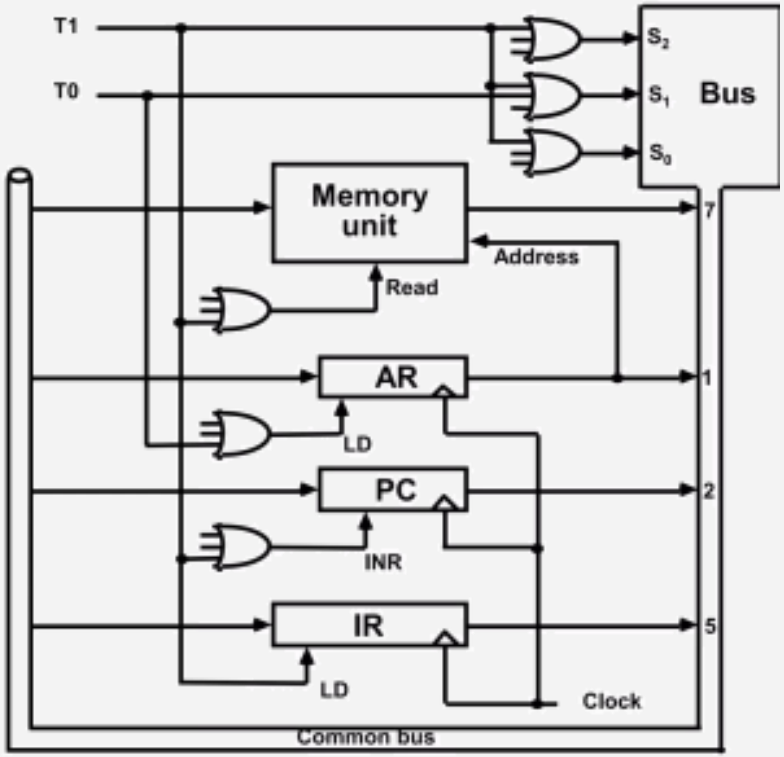
کدگشايي

در مرحله کدگشايي (Decoding) در واقع نوع دستور و بيت هاي آدرس از يکديگر متمايز مي شوند.

- ✓ ريزعمل هايي که در مرحله کدگشايي انجام مي شوند:

$$T_2 : AR \leftarrow IR(0 - 11) , D_7 - D_0 \leftarrow IR(12 - 14) , I \leftarrow IR(15)$$

نحوه پیاده‌سازی سخت افزاری کنترل واکشی و کدگشایی



T_0 : AR ← PC

T_1 : IR ← M[AR] , PC ← PC +1

T_2 : AR ← IR(0 – 11) ,
D_7 –D_0 ← IR(12 – 14) ,
I ← IR(15)

آدرس دهي

در اين مرحله در صورتي نياز به محاسبه آدرس مؤثر است كه:

- ✓ نخست آنكه دستور از نوع مراجعه به حافظه باشد (D۷=۰)
- ✓ دوم آنكه آدرس دهي از نوع غير مستقيم باشد (I=۱)

بنابراين، ريزعمل هايي كه در مرحله Addressing انجام مي شوند:

$D_7'\ I\ T_3 : AR \leftarrow M[AR]$ → دستور مراجعه به حافظه (غيرمستقيم)

$D_7'\ I'\ T_3 : Nothing$ → دستور مراجعه به حافظه (مستقيم)

$D_7\ I'\ T_3 : Nothing$ → دستور مراجعه به ثبات

$D_7\ I\ T_3 : Nothing$ → دستور مراجعه به I/O

اجرا

دستورهاي مراجعه به ثبات و مراجعه بـه I/O در همـان بازه زماني T۳ به بعد قابل اجرا هستند.

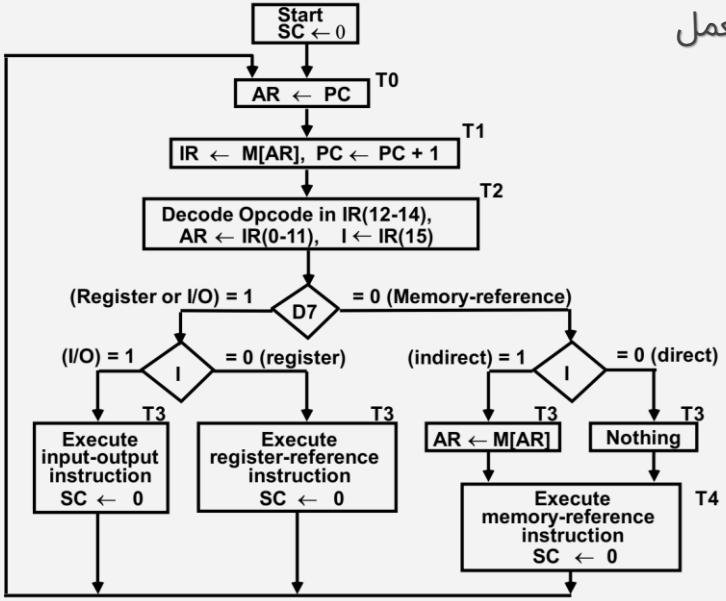
مـثلاً، ريزعمـل هـايي كـه در مرحلـه اجـراي دسـتور AND بـا Ac انجـام مي شوند:

$$D_0\ T_4 : DR \leftarrow M[AR]$$

$$D_0\ T_5 : AC \leftarrow AC \wedge DR\ ,\ SC \leftarrow 0$$

- ✓ در پايان هر سيكل دستور، حتماً بايستي SC صفر شـود تـا بـراي سـيكل دستور بعدي آماده گردد. به عبارت ديگر، پس از اجرای هر دستورالعمل، سیکل مجددا از مرحله ۱ (واکشی) برای دستورالعمل بعدی آماده می‌شود.

فلوچارت تعیین نوع دستورالعمل



$D_7'\ I\ T_3 : AR \leftarrow M[AR]$ ← دستور مراجعه به حافظه (غیرمستقیم)

$D_7'\ I'\ T_3 : Nothing$ ← دستور مراجعه به حافظه (مستقیم)

$D_7\ I'\ T_3 :$ Execute Reg Instruction ← دستور مراجعه به ثبات

$D_7\ I\ T_3 :$ Execute I/O Instruction ← دستور مراجعه به I/O

دستورات مراجعه به حافظه

Symbol	Operation Decoder	Symbolic Description
AND	D_0	$AC \leftarrow AC \wedge M[AR]$
ADD	D_1	$AC \leftarrow AC + M[AR], E \leftarrow C_{out}$
LDA	D_2	$AC \leftarrow M[AR]$
STA	D_3	$M[AR] \leftarrow AC$
BUN	D_4	$PC \leftarrow AR$
BSA	D_5	$M[AR] \leftarrow PC, PC \leftarrow AR + 1$
ISZ	D_6	$M[AR] \leftarrow M[AR] + 1$, if $M[AR] + 1 = 0$ then $PC \leftarrow PC+1$

LDA: Load to Accumulator
STA: Store Accumulator
BUN: Branch Unconditionally
BSA: Branch and Save Return Address
ISZ: Increment and Skip if Zero

دستورات مراجعه به حافظه

AND to AC D_0T_4: DR ← M[AR] D_0T_5: AC ← AC ∧ DR, SC ← 0	Read operand AND with AC
ADD to AC D_1T_4: DR ← M[AR] D_1T_5: AC ← AC + DR, E ← C_{out}, SC ← 0	Read operand Add to AC and store carry in E
LDA: Load to AC D_2T_4: DR ← M[AR] D_2T_5: AC ← DR, SC ← 0	Read operand Load DR content in AC
STA: Store AC D_3T_4: M[AR] ← AC, SC ← 0	Store AC content in Memory

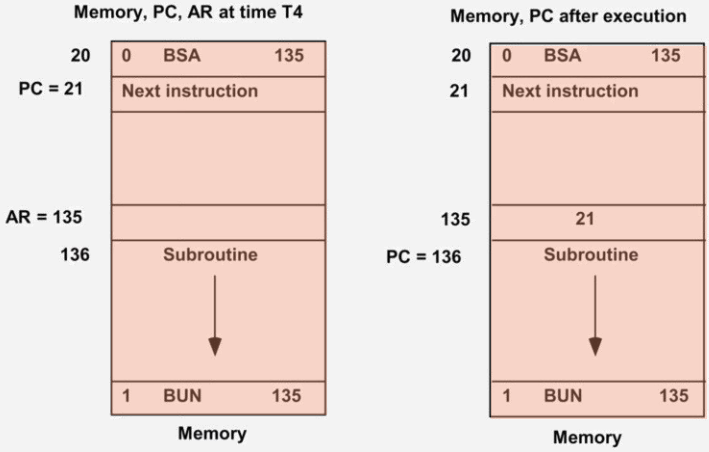
BUN: Branch Unconditionally

D_4T_4: PC ← AR, SC ← 0

BSA:

D_5T_4: M[AR] ← PC, AR ← AR + 1

D_5T_5: PC ← AR, SC ← 0



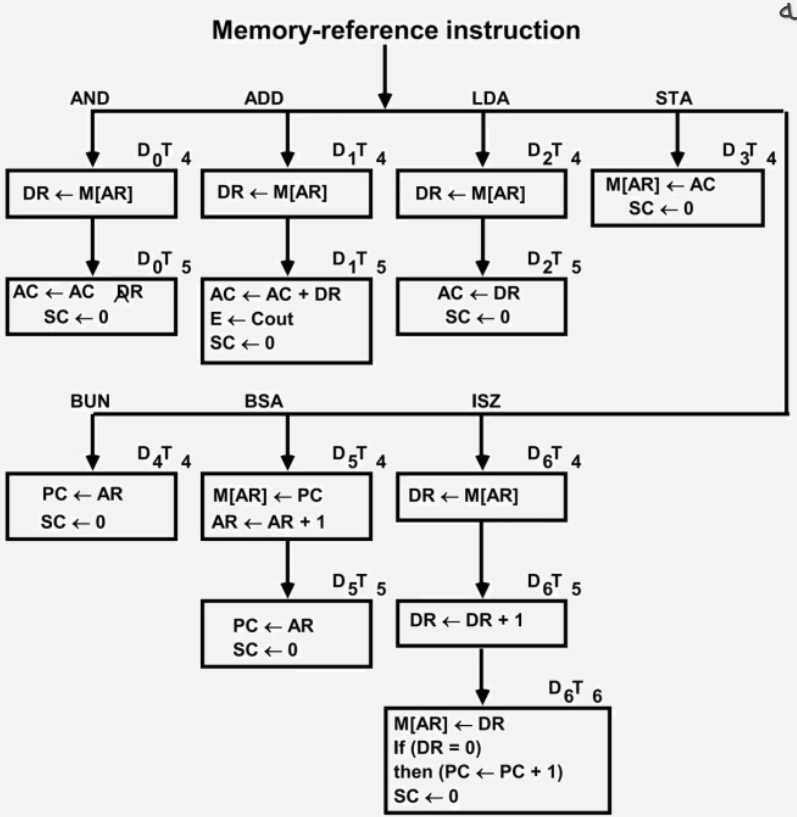
ISZ: Increment and Skip-if-Zero

D_6T_4: DR ← M[AR]

D_6T_5: DR ← DR + 1

D_6T_4: M[AR] ← DR, if (DR = 0) then (PC ← PC + 1), SC ← 0

دستورات مراجعه به حافظه



دستورات مراجعه به حافظه

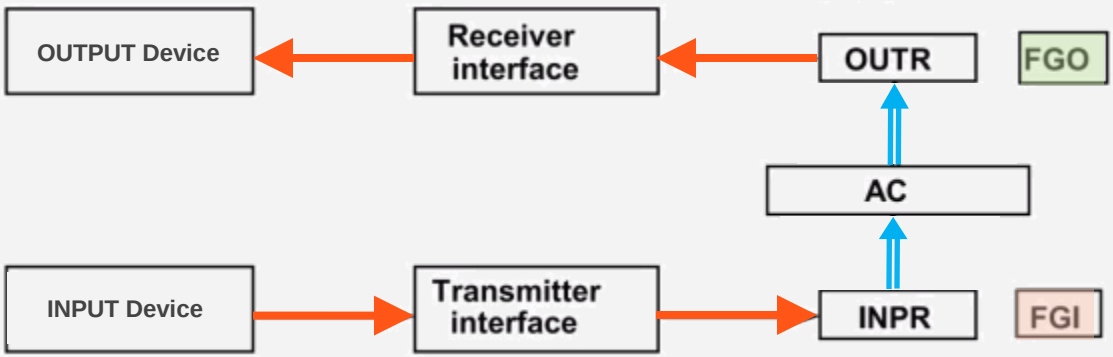
Fetch	$R'T_0$:	$AR \leftarrow PC$
	$R'T_1$:	$IR \leftarrow M[AR], PC \leftarrow PC + 1$
Decode	$R'T_2$:	$D0, ..., D7 \leftarrow$ Decode $IR(12 \sim 14)$, $AR \leftarrow IR(0 \sim 11), I \leftarrow IR(15)$
Indirect	$D_7'IT_3$:	$AR \leftarrow M[AR]$
Interrupt		
	$T_0'T_1'T_2'(IEN)(FGI + FGO)$:	$R \leftarrow 1$
	RT_0:	$AR \leftarrow 0, TR \leftarrow PC$
	RT_1:	$M[AR] \leftarrow TR, PC \leftarrow 0$
	RT_2:	$PC \leftarrow PC + 1, IEN \leftarrow 0, R \leftarrow 0, SC \leftarrow 0$
Memory-Reference		
AND	D_0T_4:	$DR \leftarrow M[AR]$
	D_0T_5:	$AC \leftarrow AC \wedge DR, SC \leftarrow 0$
ADD	D_1T_4:	$DR \leftarrow M[AR]$
	D_1T_5:	$AC \leftarrow AC + DR, E \leftarrow C_{out}, SC \leftarrow 0$
LDA	D_2T_4:	$DR \leftarrow M[AR]$
	D_2T_5:	$AC \leftarrow DR, SC \leftarrow 0$
STA	D_3T_4:	$M[AR] \leftarrow AC, SC \leftarrow 0$
BUN	D_4T_4:	$PC \leftarrow AR, SC \leftarrow 0$
BSA	D_5T_4:	$M[AR] \leftarrow PC, AR \leftarrow AR + 1$
	D_5T_5:	$PC \leftarrow AR, SC \leftarrow 0$
ISZ	D_6T_4:	$DR \leftarrow M[AR]$
	D_6T_5:	$DR \leftarrow DR + 1$
	D_6T_6:	$M[AR] \leftarrow DR$, if(DR=0) then $(PC \leftarrow PC + 1)$, $SC \leftarrow 0$

	$D_7I'T_3 = r$	(Common to all register-reference instr)
	$IR(i) = B_i$	(i = 0,1,2, ..., 11)
	r:	$SC \leftarrow 0$
CLA	rB_{11}:	$AC \leftarrow 0$
CLE	rB_{10}:	$E \leftarrow 0$
CMA	rB_9:	$AC \leftarrow AC'$
CME	rB_8:	$E \leftarrow E'$
CIR	rB_7:	$AC \leftarrow shr\ AC, AC(15) \leftarrow E, E \leftarrow AC(0)$
CIL	rB_6:	$AC \leftarrow shl\ AC, AC(0) \leftarrow E, E \leftarrow AC(15)$
INC	rB_5:	$AC \leftarrow AC + 1$
SPA	rB_4:	$If(AC(15) = 0)\ then\ (PC \leftarrow PC + 1)$
SNA	rB_3:	$If(AC(15) = 1)\ then\ (PC \leftarrow PC + 1)$
SZA	rB_2:	$If(AC = 0)\ then\ (PC \leftarrow PC + 1)$
SZE	rB_1:	$If(E = 0)\ then\ (PC \leftarrow PC + 1)$
HLT	rB_0:	$S \leftarrow 0$

دستورات I/O

	$D_7IT_3 = p$	(Common to all input-output instructions)
	$IR(i) = B_i$	(i = 6,7,8,9,10,11)
	p:	$SC \leftarrow 0$
INP	pB_{11}:	$AC(0\text{-}7) \leftarrow INPR, FGI \leftarrow 0$
OUT	pB_{10}:	$OUTR \leftarrow AC(0\text{-}7), FGO \leftarrow 0$
SKI	pB_9:	If(FGI=1) then ($PC \leftarrow PC + 1$)
SKO	pB_8:	If(FGO=1) then ($PC \leftarrow PC + 1$)
ION	pB_7:	$IEN \leftarrow 1$
IOF	pB_6:	$IEN \leftarrow 0$

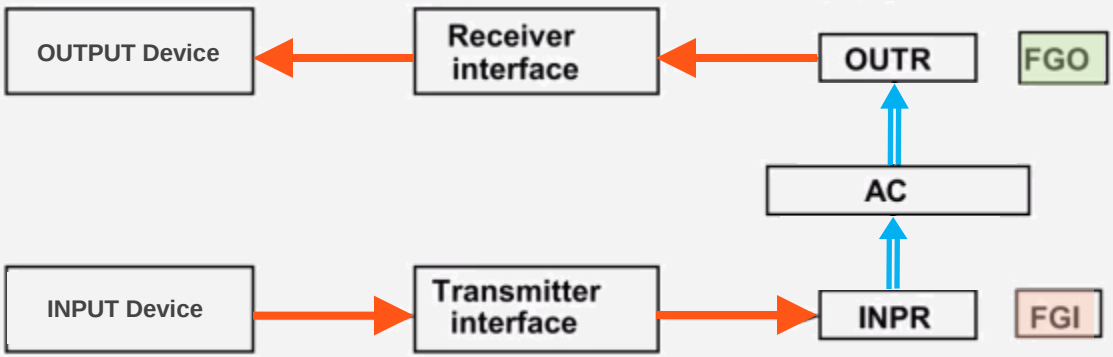
دستورات I/O



: مسیر ارتباط سریال

: مسیر ارتباط موازی

دستورات I/O



- ✓ دستگاه‌های ورودی/خروجی داده‌ها را به صورت سریال منتقل می‌کنند.
- ✓ داده‌های سریال دستگاه ورودی به کمک واسط به صورت سریال به ثبات INPR منتقل می شوند.
- ✓ داده‌های ذخیره شده در ثبات OUTR به صورت سریال به واسط انتقال داده می‌شوند.
- ✓ داده‌های انتقال داده شده به واسط به صورت سریال به دستگاه خروجی انتقال داده می‌شوند.
- ✓ ثبات‌های INPR و OUTR به صورت موازی با AC تبادل داده دارند در حالی که با دستگاه‌های ورودی/خروجی به صورت سریال تبادل داده می کنند.
- ✓ به جهت سنکرون کردن اختلاف زمانی بین دستگاه‌های ورودی/خروجی و کامپیوتر از پرچم (flag) های FGI و FGO استفاده می‌شود.

دستورات I/O

- **فرایند ورود داده‌ها**
  - ابتدا FGI=۰است.
  - فرضاً اگر کليدي فشرده شود، کد ۸بيتي آن به طـور سـريال وارد INPRشـده و هنگامي که ۸بيت INPRکامل شد، FGI=۱مي شود تا کـامپيوتر بفهمـد کـه INPRپر شده و دستگاه جانبي نيز فعلاً داده‌های ارسال نمي کند.
  - وقتي که FGI=۱شد کامپيوتر محتواي INPRرا به طور مـوازي بـه انبـاره (AC) منتقل مي کند و FGI=۰مي شود تا داده‌های بعدي بتواند وارد شود.

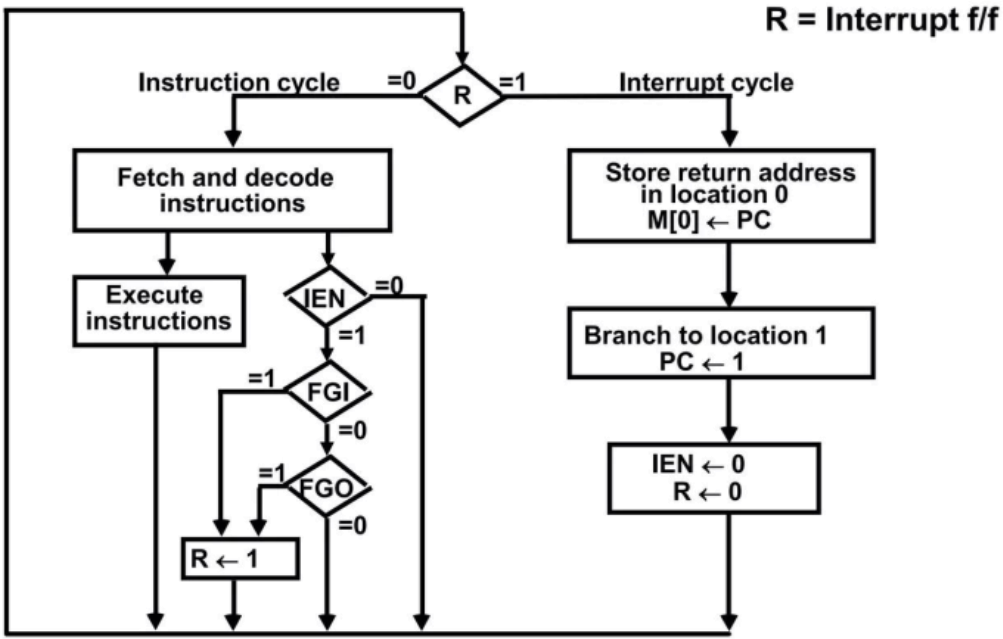
- **فرایند خروج داده‌ها**
  - ابتدا FGO=۱است و به معناي آن است که OUTRخالي است.
  - بنابراين داده‌های ACبه طور موازي به OUTRمنتقل شده و FGO=۰مي شود.
  - در آخر با صفر شدن FGO داده‌ها به صورت سریال به دستگاهر خروجی منتقل می‌شوند.


FGR
= 0
FGI
= 1

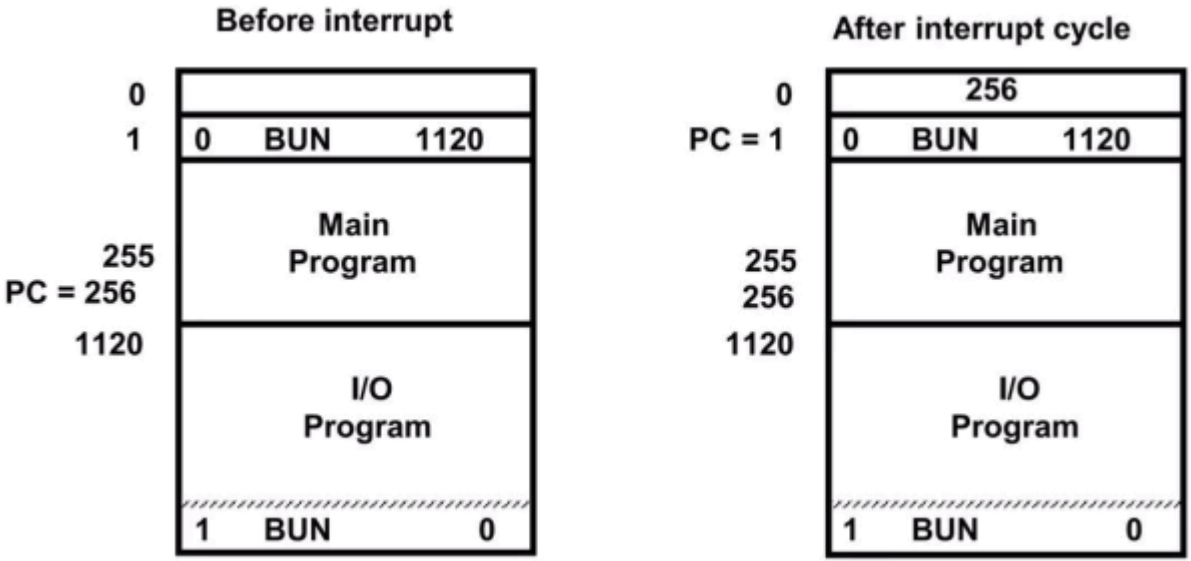

FGR
= 1
FGI
= 0

وقفه

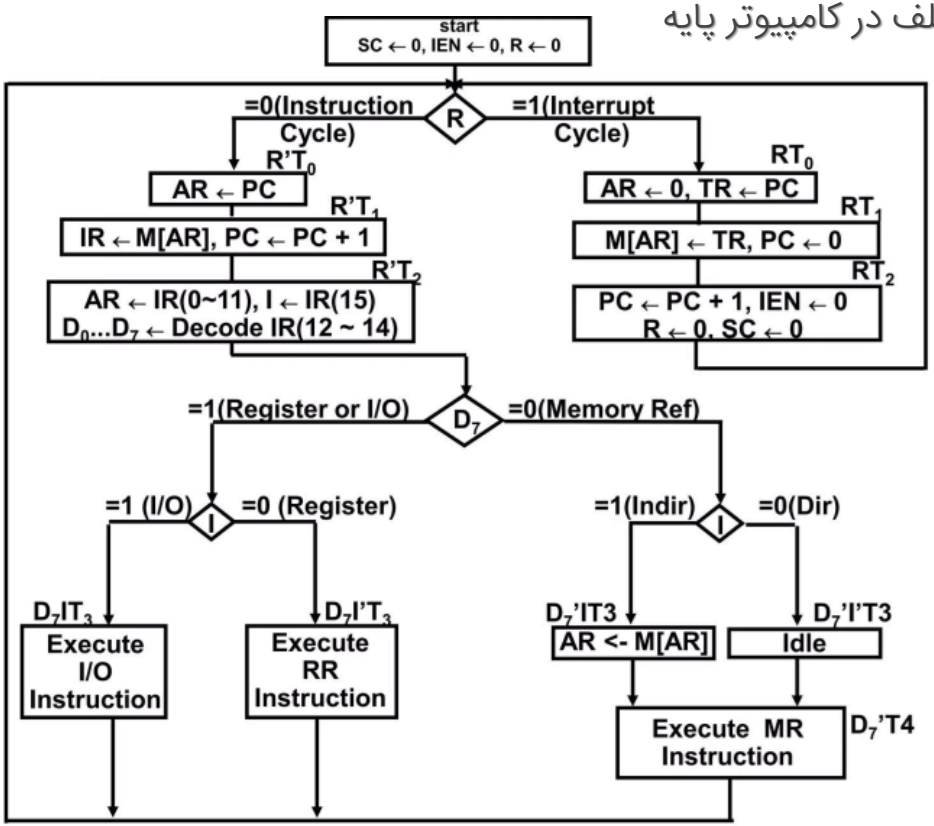
وقفه


R = Interrupt f/f
Instruction cycle
=0
R
=1
Interrupt cycle
Fetch and decode instructions
Execute instructions
IEN
=0
=1
FGI
=1
=0
FGO
=1
=0
R ← 1
Store return address in location 0
M[0] ← PC
Branch to location 1
PC ← 1
IEN ← 0
R ← 0

وقفه


Before interrupt
0
1
255
PC = 256
1120
0 BUN 1120
Main Program
I/O Program
1 BUN 0
After interrupt cycle
0
PC = 1
255
256
1120
256
0 BUN 1120
Main Program
I/O Program
1 BUN 0

فلوچارت اعمال مختلف در کامپیوتر پایه



razeghizade@gmail.com

Razeghizade.pudica.ir

CREADITS: This presentation was created by M.Razeghizadeh
Please keep this slide for attribution