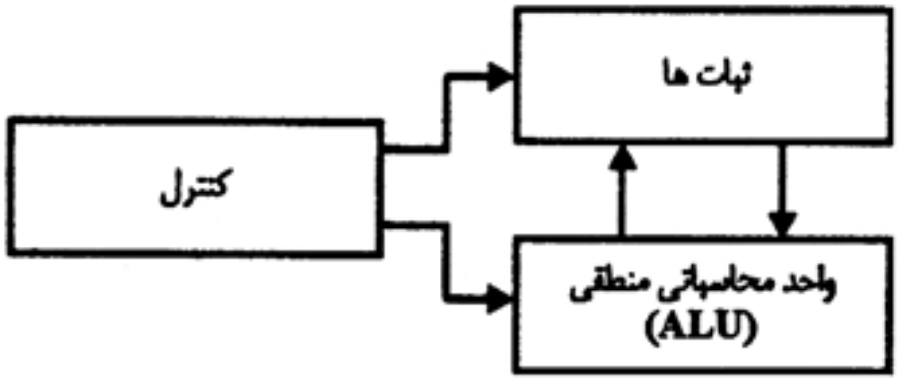
Computer System Architecture

واحد مرکزی پردازش

CPU

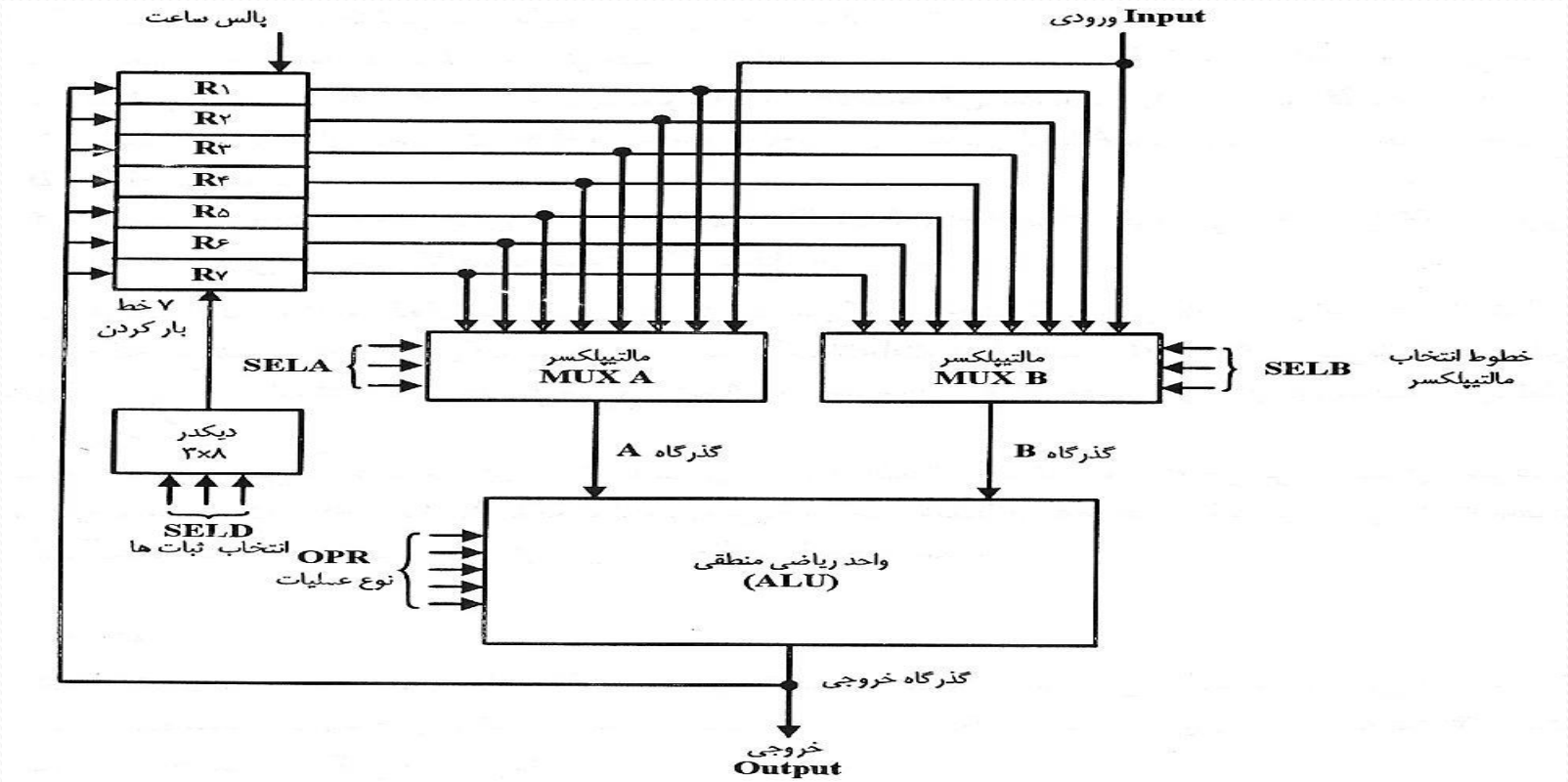
CPU از ۳ قسمت اصلی تشکیل شده است:

1. ثبات ها داده های موقت را که برای اجرای دستورات نیاز هستند در خود ذخیره می کند.
2. ALU، ریزاعمال لازم برای اجرای دستورات را انجام می دهد.
3. واحد کنترل، مدریت انتقال داده ها را در بین ثباتها دارد(ALU، را هدایت می کند).



شکل (۸-۱) قسمت‌های اساسی CPU

سازمان ثبات عمومی



الف: ساختار کلی

۳	۳	۳	۵
SELA	SELB	SELD	OPR

ب: کلمه کنترل

شکل ۸-۲ واحد ریاضی / منطقی ALU و ثبات‌های پروسسور

کد کردن انتخاب ثبات ها

جدول(۸-۱)، کُد کردن انتخاب ثبات‌ها

کُدباینری	SELA خروجی MUXA	SELB خروجی MUXB	SELD ورودی دیکُدر
۰۰۰	Input	Input	هیچ‌ثبات
۰۰۱	R۱	R۱	R۱
۰۱۰	R۲	R۲	R۲
۰۱۱	R۳	R۳	R۳
۱۰۰	R۴	R۴	R۴
۱۰۱	R۵	R۵	R۵
۱۱۰	R۶	R۶	R۶
۱۱۱	R۷	R۷	R۷

تخصیص کد به اعمال ALU

جدول (۸-۲) کُد کردن عملیات ALU

فرم نماد	عملیات	مقدار OPR
TSFA	انتقال A	۰۰۰۰۰
INCA	یکی به A اضافه کن	۰۰۰۰۱
ADD	جمع A + B	۰۰۰۱۰
SUB	تفریق A - B	۰۰۱۰۱
DECA	یکی از A کم‌کردن	۰۰۱۱۰
AND	AND	۰۱۰۰۰
OR	OR	۰۱۰۱۰
XOR	XOR	۰۱۱۰۰
COMA	مکمل A	۰۱۱۱۰
SHRA	شیفت به راست A	۱۰۰۰۰
SHLA	شیفت به چپ A	۱۱۰۰۰

مثال

جدول (۸-۳) مثال‌هایی از ریز عملیات برای CPU

ریز عملیات	تخصیص سمبولیک			دستور	کلمه کنترل
	SELA	SELB	SELD	OPR	
R۱ ← R۲ - R۳	R۲	R۳	R۱	SUB	۰۱۰ ۰۱۱ ۰۰۱ ۰۰۱۰۱
R۴ ← R۴ ∨ R۵	R۴	R۵	R۴	OR	۱۰۰ ۱۰۱ ۱۰۰ ۰۱۰۱۰
R۶ ← R۶ + ۱	R۶	_	R۶	INCA	۱۱۰ ۰۰۰ ۱۱۰ ۰۰۰۰۱
R۷ ← R۱	R۱	_	R۷	TSFA	۰۰۱ ۰۰۰ ۱۱۱ ۰۰۰۰۰
Output ← R۲	R۲	_	هیچ	TSFA	۰۱۰ ۰۰۰ ۰۰۰ ۰۰۰۰۰
Output ← Input	Input	_	هیچ	TSFA	۰۰۰ ۰۰۰ ۰۰۰ ۰۰۰۰۰
R۴ ← shl R۴	R۴	_	R۴	SHAL	۱۰۰ ۰۰۰ ۱۰۰ ۱۱۰۰۰
R۵ ← ۰	R۵	R۵	R۵	XOR	۱۰۱ ۱۰۱ ۱۰۱ ۰۱۱۰۰

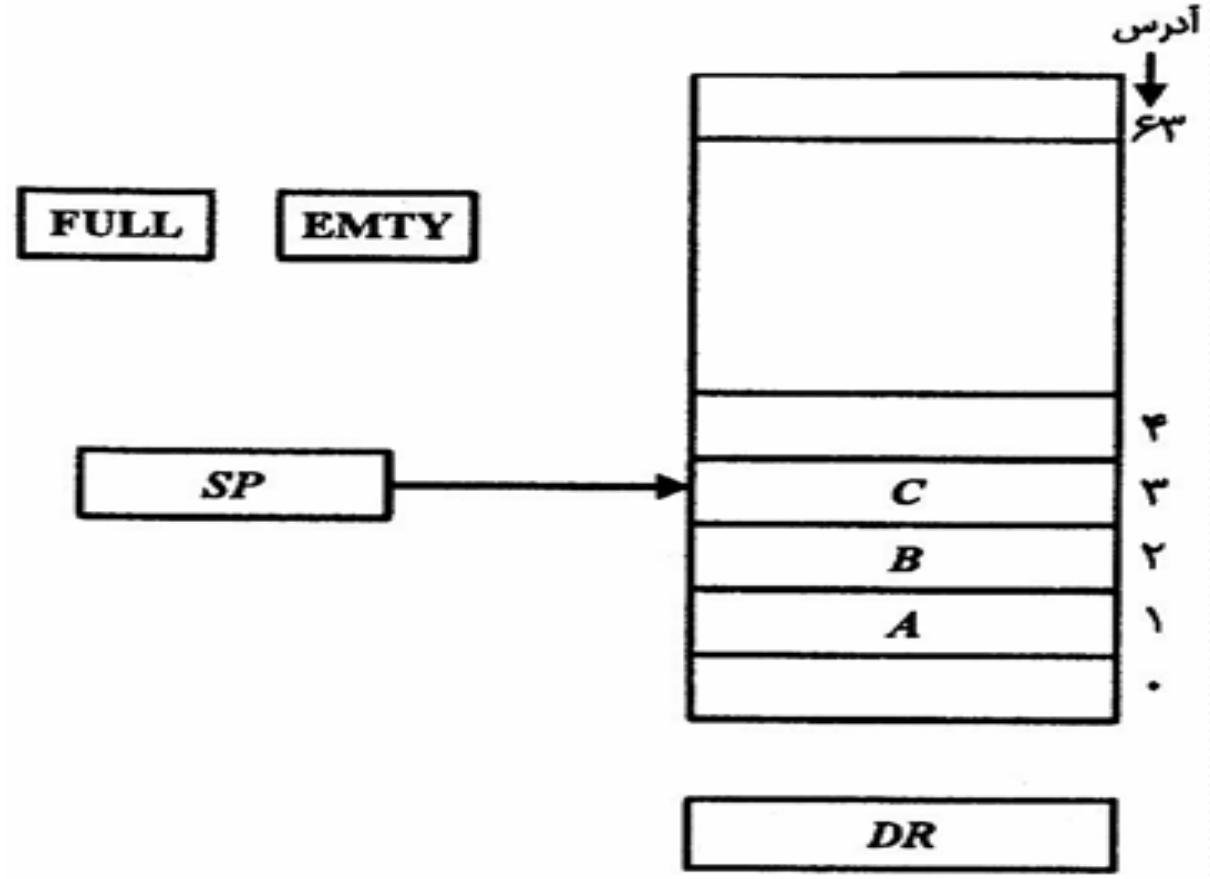
سازمان پشته

- حافظه ایست که بر اساس **LIFO** کار می کند.
- پشته می تواند، خود حافظه ای جداگانه باشد و یا بخشی از حافظه کامپیوتر.

دو عمل روی پشته انجام می شود: **۱. PUSH** **۲. POP**

- ثباتی که آدرس پشته را در خود ذخیره می کند، **ثبات پشته (SP)** نام دارد.
- SP می تواند به خانه پر بالای پشته اشاره کند یا به خانه خالی.

پشته ثباتی



شکل (۸–۳) یک حافظه پُشته ۶۴ کلمه‌ای

عملیات PUSH و POP

:PUSH

SP ← SP + 1	افزایش اشاره‌گر پشته
M [SP] ← DR	نوشتن داده در مکان بالای داده
If(SP= 0) then (FULL ←1)	آیا پشته پر است یا نه
EMTY ← 0	علامت خالی نبودن پشته

:POP

DR ← M[SP]	خواندن داده از مکان بالای پشته
SP ← SP−1	کاهش اشاره‌گر پشته
If (SP=0) then (EMTY ← 1)	آیا پشته خالی است
FULL ← 0	علامت پرنبودن پشته

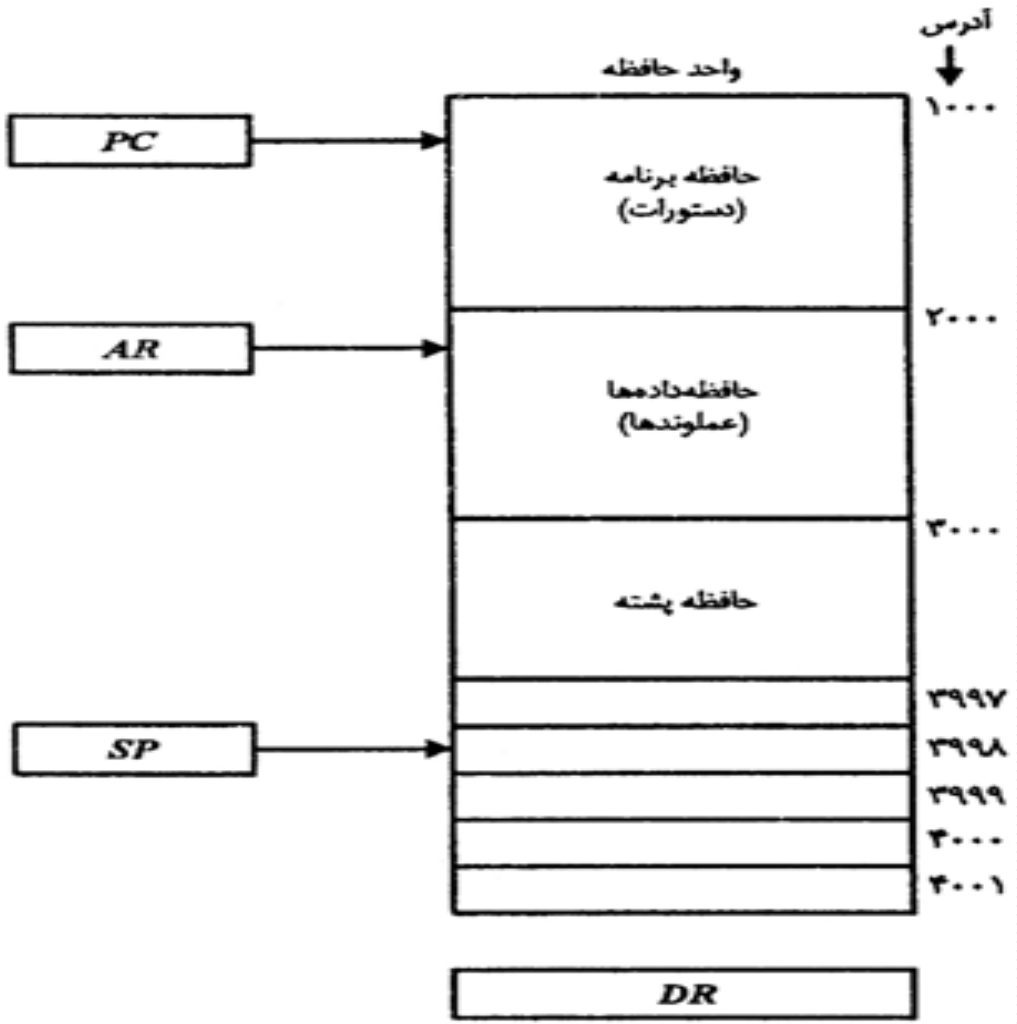
پشته حافظه ای

$SP \leftarrow SP - 1$

$M[SP] \leftarrow DR$

$DR \leftarrow M[SP]$

$SP \leftarrow SP + 1$



شکل (۸-۴) حافظه کامپیوتر با قسمت‌های حافظه برنامه، حافظه داده‌ها و حافظه پشته.

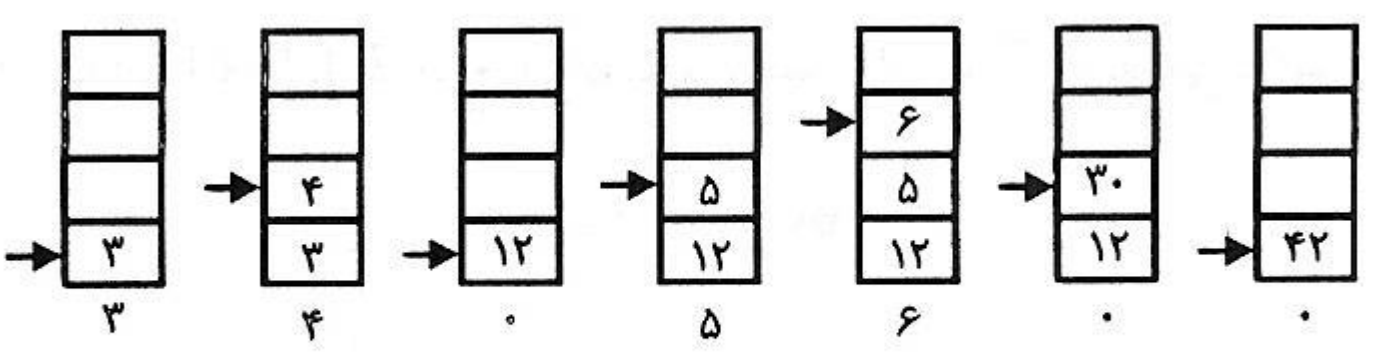
کاربرد پشته در محاسبه عبارات ریاضی

به منظور استفاده از پشته برای محاسبات ریاضی، باید عبارات را به فرم پسوندی (معکوس لهستانی) نوشت.

از سمت چپ عبارت را بررسی می کنیم.

با رسیدن به هر عملوند آنرا به پشته PUSH می کنیم.

با رسیدن به هر عملگر، دو عملوند بالای پشته را برداشته و با هم عمل می کنیم و حاصل را به پشته بر می گردانیم. **مثال: +*56*34**



شکل ۸-۵ عملیات حافظه پشته برای محاسبه ۶ × ۵ + ۴ × ۳

فرمت دستورالعمل

اغلب کامپیوترها دارای سه نوع ساختار کلی هستند:

1. ساختار مبتنی بر یک آکومولاتور `AC ← AC + M [X]`
2. ساختار چند ثباتی `ADD R1, R2, R3` `ADD R1, R2`
3. ساختار پشته ای از دستورات PUSH و POP استفاده می کنند. (از یک فیلد آدرس استفاده می کنند.) `PUSH X`

مثال

می خواهیم دستور X = (A+B)*(C+D) را با دستور صفر، یک، دو و سه آدرسه محاسبه کنیم:

۱. دستورات ۳ آدرسی:

ADD	R_1, A, B	$R_1 \leftarrow M[A] + M[B]$
ADD	R_2, C, D	$R_2 \leftarrow M[C] + M[D]$
MUL	X, R_1,R_2	$M[X] \leftarrow R_1 * R_2$

۲. دستورات دو آدرسی:

MOV	R_1, A	$R_1 \leftarrow M[A]$
ADD	R_1, B	$R_1 \leftarrow R_1 + M[B]$
MOV	R_2, C	$R_2 \leftarrow M[C]$
ADD	R_2, D	$R_2 \leftarrow R_2 + M[D]$
MUL	R_1, R_2	$R_1 \leftarrow R_1 * R_2$
MOV	X, R_1	$M[X] \leftarrow R_1$

۲. دستورات تک آدرسی:

LOAD	A	AC ← M[A]
ADD	B	AC ← AC + M[B]
STORE	T	M[T] ← AC
LOAD	C	AC ← M[C]
ADD	D	AC ← AC + M[D]
MUL	T	AC ← AC * M[T]
STORE	X	M[X] ← AC

۲. دستورات بدون فیلد آدرس:

PUSH	A	TOS ← A
PUSH	B	TOS ←B
ADD		TOS ← (A + B)
PUSH	C	TOS ← C
PUSH	D	TOS ← D
ADD		TOS (C + D)
MUL		TOS (C+D)*(A+B)
Pop	X	M[X] ← TOS

دستورات RISC

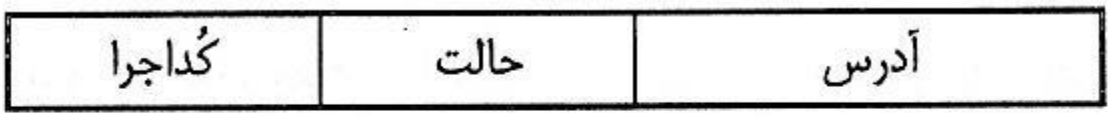
معماری کم دستور

مجموعه دستورات محدود به Loadو Store.

ثایر دستورات فقط روی ثبات ها عمل می کند.

LOAD	R_1 , A	$R_1 \leftarrow M[A]$
LOAD	R_2 , B	$R_2 \leftarrow M[B]$
LOAD	R_3 , C	$R_3 \leftarrow M[C]$
LOAD	R_4 , D	$R_4 \leftarrow M[D]$
ADD	R_1 , R_1,R_2	$R_1 \leftarrow R_1 + R_2$
ADD	R_3 , R_3, R_4	$R_3 \leftarrow R_3 + R_4$
MUL	R_1 , R_1 ,R_3	$R_1 \leftarrow R_1 * R_3$
STORE	X , R_1	$M[X] \leftarrow R_1$

مدهای آدرس دهی



شکل (۸-۶) فرمت دستور با حالت آدرس دهی

مدهای آدرس دهی

مد ضمنی: عملوند به طور ضمنی در خود دستور نهفته است. مثل دستور CMA (Complement AC) عملوند در ثبات AC می باشد.

مد بلافصل: یک دستور در این مد دارای فیلد عملوند است نه فیلد آدرس.

مد ثباتی: عملوند در حافظه نیست بلکه در یکی از ثباتهای پردازنده است.

مد ثباتی غیر مستقیم: دستور، ثباتی از CPU را مشخص می کند که آدرس عملوند در آن است.

مد افزاینده یا کاهنده خودکار: مثل روش قبل است با این تفاوت که ثبات، بعد یا قبل استفاده، یک واحد افزایش یا کاهش میابد.

مدهای آدرس دهی

مد آدرس دهی مستقیم: عملوند در حافظه و آدرس آن در فیلد آدرس دستورالعمل.

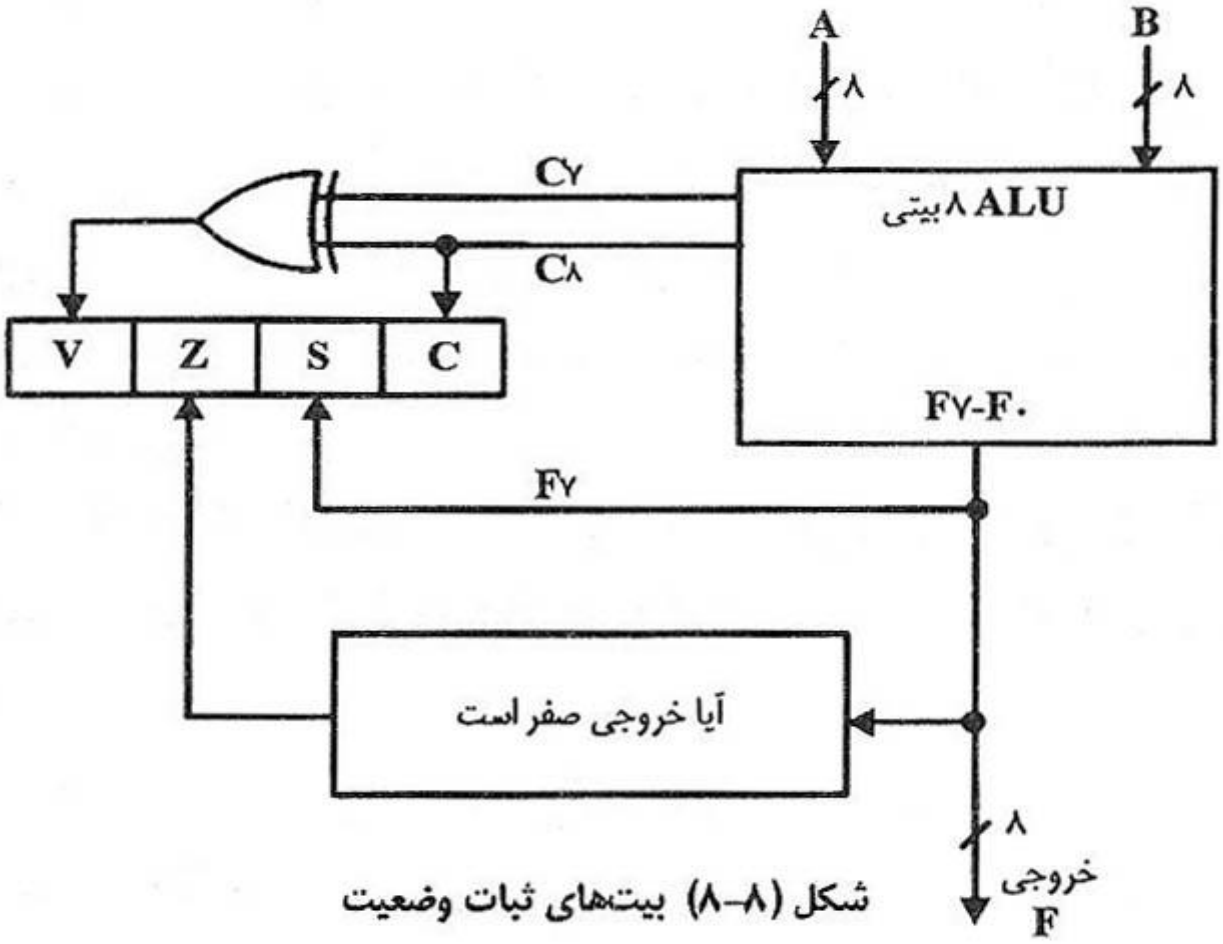
مد آدرس دهی غیرمستقیم: فیلد آدرس دستور آدرس آدرس موثر است.

مد آدرس دهی نسبی: محتویات PC به فیلد آدرس افزوده می شود تا آدرس موثر محاسبه شود.

مد آدرس دهی ایندکس: محتویات یک ثبات ایندکس به قسمت آدرس دستور اضافه می شود.

مد آدرس دهی ثبات پایه: محتویات یک ثبات پایه به قسمت آدرس دستور اضافه می شود.

بیتهای وضعیت



شکل (۸-۸) بیت‌های ثبات وضعیت