بنام خداوند جان و خرد

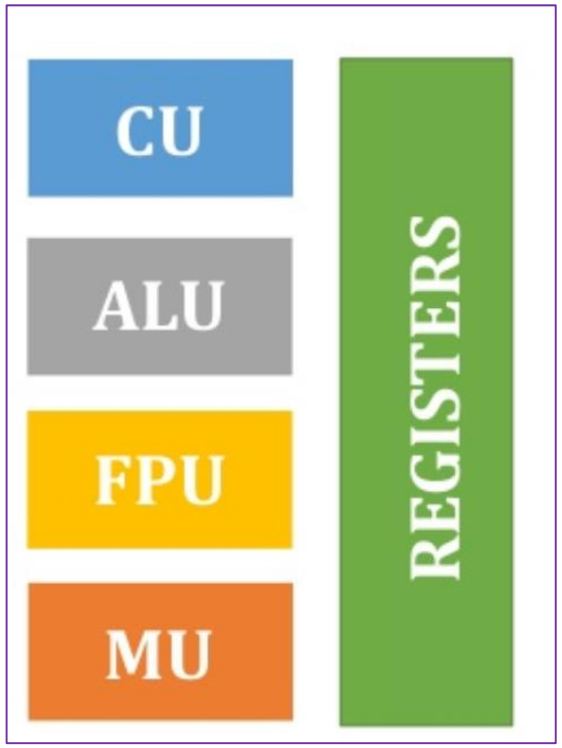
معماری کامپیوتر

جلسه هجدهم: واحد کنترل (Control Unit)

واحد کنترل پردازنده

- در ادامه بحث آشنایی با اجزا و معماری سخت‌افزاری سیستم‌های کامپیوتری در حوزه پردازنده
- چگونگی طراحی واحد کنترل پردازشگر
  - مقدماتی بر طراحی پردازنده‌ها
  - آشنایی با وظایف و اهداف واحد کنترل
    - استفاده از الگوریتم ترتیبی Von Neumann
  - انواع طراحی واحدهای کنترل
    - کنترل برنامه‌پذیر
    - کنترل سیم‌بندی شده
  - شیوه ارتباط واحد کنترل با تجهیزات ورودی و خروجی

اجزای واحد پردازشگر مرکزی

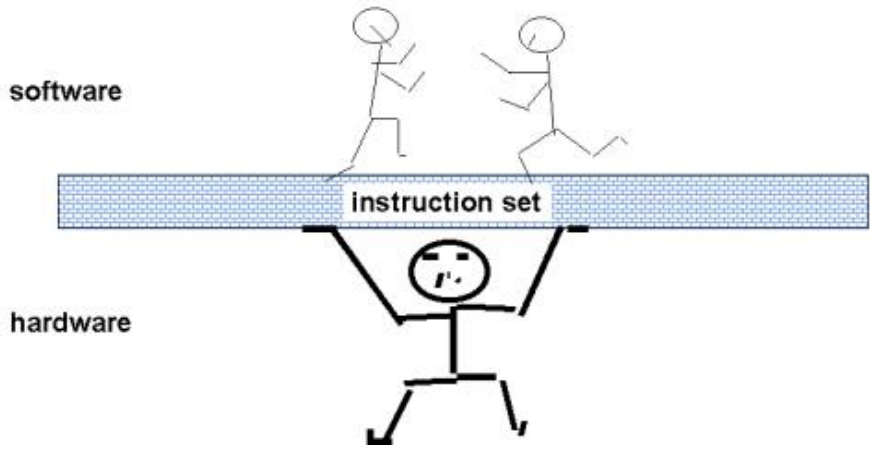


CPU

- واحدهای ALU و FPU
  - انجام عملیات محاسباتی و منطقی برروی اعداد صحیح و اعشاری
- واحد MU
  - نگهداری دستورالعمل‌ها و داده‌های موردنیاز در اجرا
- واحد CU
  - نظارت بر ترتیب اجرای عملیات و دستورالعمل‌ها

مجموعه دستورالعمل‌ها

- ورودی واحد پردازشگر مرکزی (CPU):
  - مجموعه دستورالعمل‌ها (ISA: Instruction Set Architecture)
  - مجموعه دستورالعمل‌ها دید سطح بالا از سیستم کامپیوتری فراهم می‌کند
    - نشانگر ساختار مفهومی و رفتار کارکردی سیستم کامپیوتری
    - قابلیت‌های سیستم کامپیوتری را نشان می‌دهد
    - سطح بالاست و جزئیات پیاده‌سازی را نشان نمی‌دهد
    - واسط بین سخت‌افزار و پایین‌ترین سطح نرم افزار است



مجموعه دستورالعمل‌ها

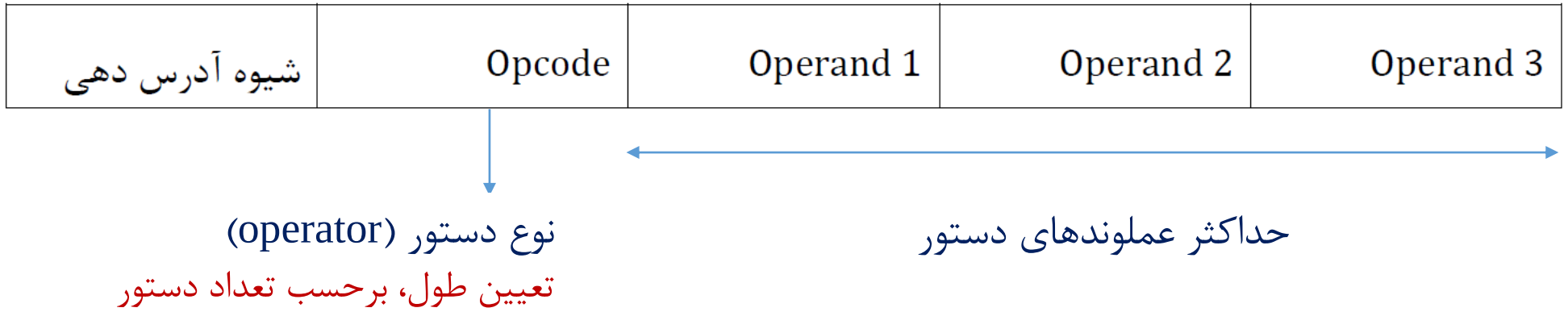
- هر دستور از دو بخش تشکیل شده است
  - عملگر (operator) و عملوند (operand)
    - عملگرها نوع عملیاتی که باید انجام شود را نشان می‌دهند
      - مانند: add، sub، mov و ...
      - عملگرها در دستور توسط کد باینری نشان داده می‌شوند (opcode)
    - عملوندها داده‌هایی هستند که عملگرها روی آن‌ها اعمال می‌شوند
      - مانند mov a,b یا add a
      - برحسب نوع عملیات، مبدا مقصد یا بعدی هستند

طراحی مجموعه دستورالعمل‌ها

- با داشتن مجموعه دستورالعمل‌ها، اطلاعات زیر درباره پردازنده قابل استخراج است (یا برعکس):
  - کامپیوتر از نوع RISC است یا CISC
    - با دانستن ماهیت دستورالعمل‌ها (I/O، ALU-based، memory-based)
  - تعداد دستورالعمل‌ها
  - تنوع دستورالعمل‌ها
  - تعداد عملوندهای (operand) دستورالعمل
    - هرچه بیشتر و متنوع باشد به ساختار CISC نزدیک‌تر می‌شویم
  - شیوه‌های آدرس‌دهی (شیوه‌های تعریف عملوندها در فضای کم دستورالعمل)

قالب دستورات

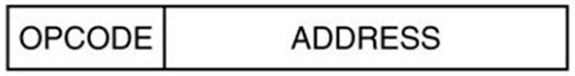
- در طراحی پردازشگر قالب دستورات می‌بایست مشخص شده باشد
  - قالب دستورات ISA



انواع ماشین از حیث تعداد عملوندهای دستورالعمل‌ها

- دستورالعمل‌ها از نظر تعداد عملوندها:
  - صفر آدرسی (پشته‌ای): دستورالعمل‌ها فاقد عملوند هستند و ماشین با پشته (stack) کار می‌کند
    - داده‌ها داخل stack هستند و به‌ترتیب از آن خوانده می‌شوند

  - تک‌آدرسی: دستورالعمل‌ها یک عملوند دارند
    - عملوند دوم در دستوراتی مانند add در رجیستر از پیش تعیین شده قرار دارد



  - دو آدرسی: دستورالعمل‌ها دو عملوند دارند

  - و...

انواع ماشین از حیث تعداد عملوندهای دستورالعمل‌ها

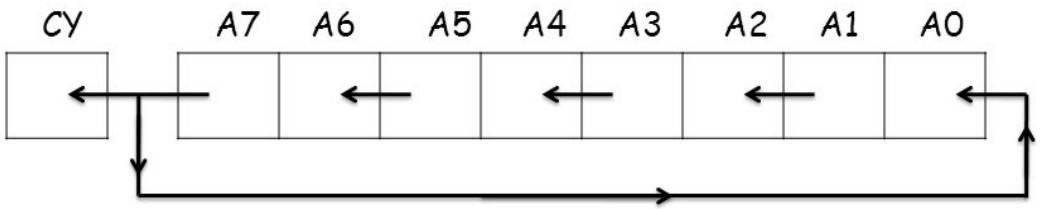
a=((b+c) * d)-e

3-Address	2-Address	1-Address	0-Address
add a, b, c	load a, b	lda b	push b
mult a, a, d	add a, c	add c	push c
sub a, a, e	mult a, d	mult d	add
	sub a, e	sub e	push d
		sta a	mult
			push e
			sub
			pop a

شیوه‌های آدرس‌دهی

- مشخص می‌کنند که عملوندهای موجود به آدرس داده اشاره دارند یا خود داده

1. آدرس‌دهی ضمنی (Implicit): دستوراتی که نیاز به عملوند ندارند
   - کامپیوترهای اولیه با این شیوه کار می‌کردند اما برنامه‌نویسی در آن‌ها بسیار محدود بود
   - به عملوند نیازی نیست یا مکان عملوند مبدا و مقصد از قبل مشخص و ثابت است
   - CLC: Clear Carry, STD: Set Direction, CMA: Complement A



RLC

شیوه‌های آدرس‌دهی

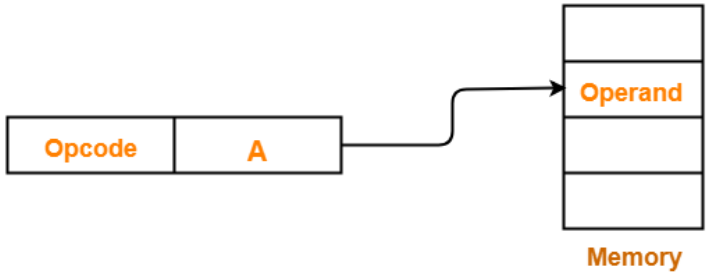
2. آدرس‌دهی بلافصل (Immediate): عدد موردنظر مستقیم به‌عنوان عملوند داده می‌شود

- این روش بازهم محدودکننده است و در موارد ایستا قابل اعمال است
- مناسب برای مقداردهی اولیه
- Add 5- Mul 10- MVI A,15h

Move Immediate

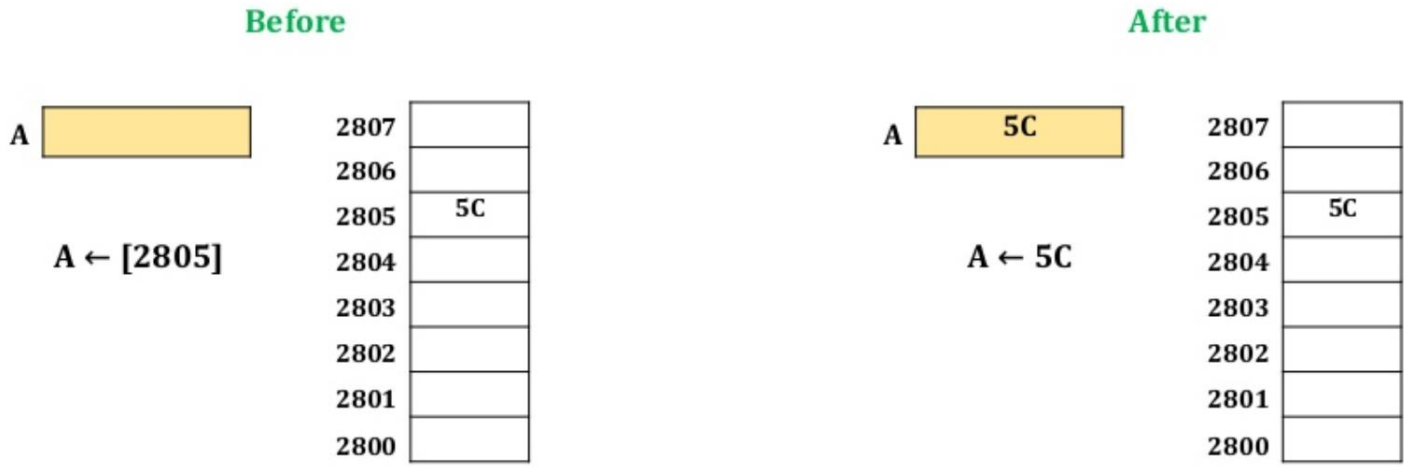
MVI A , 15h $A \leftarrow 15h$ Here 15h is the immediate operand

شیوه‌های آدرس‌دهی



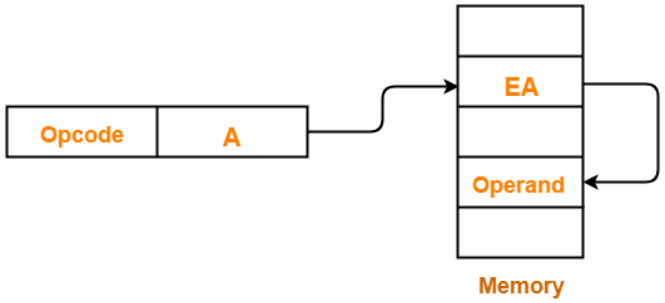
3. آدرس‌دهی حافظه‌ای مستقیم (memory direct):

- داده در داخل خانه‌ای از حافظه و در دستورالعمل نهفته است
- داده مستقیما از حافظه اصلی خوانده می‌شود
- Add[6] در حافظه، عدد موجود در آدرس ۶ را با مقدار مشخصی جمع بزن

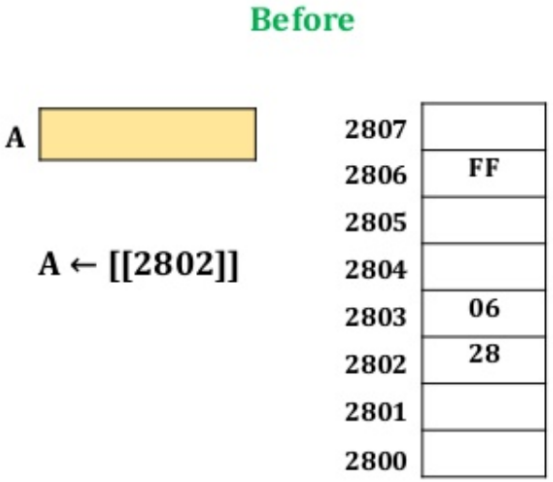


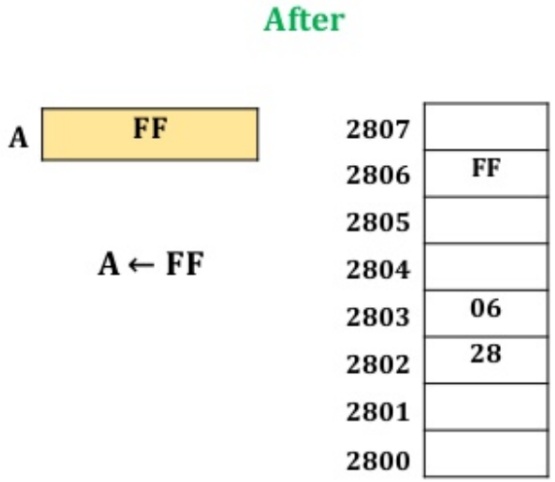
شیوه‌های آدرس‌دهی

4. آدرس‌دهی حافظه‌ای غیرمستقیم (memory indirect):



- آدرس داده موردنظر داخل خانه‌ای از حافظه است
- نیاز به دومرتبه دسترسی به حافظه داریم در نتیجه روش وقت‌گیری می‌باشد
- Add [[6]] به آدرس ۶ حافظه برو، محتوای این خانه را بخوان و به آن آدرس برو و داده را بردار و با مقدار مشخصی جمع بزن

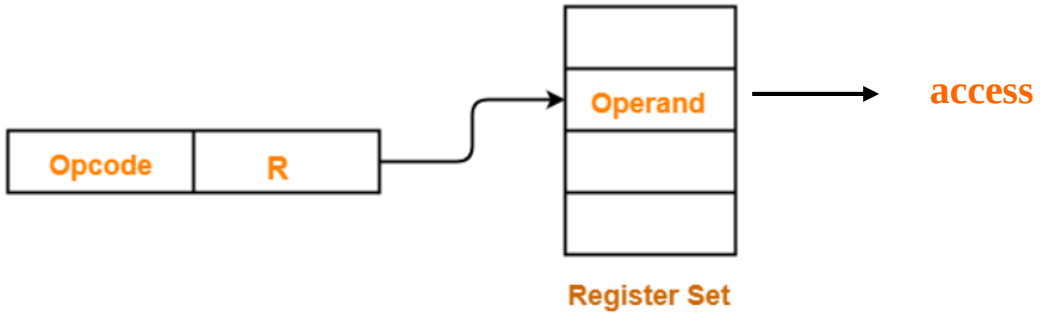




شیوه‌های آدرس‌دهی

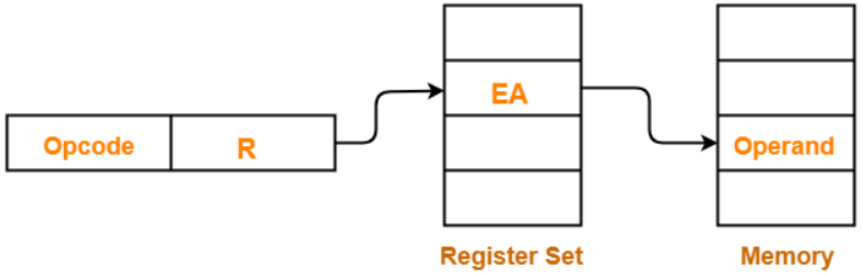
5. آدرس‌دهی ثباتی مستقیم (register direct):

- داده در داخل ثباتی است که اطلاعات ثبات جلوی دستور قرار داده شده
- Add R مقدار عدد موجود در ثبات R را مقدار مشخصی جمع بزن

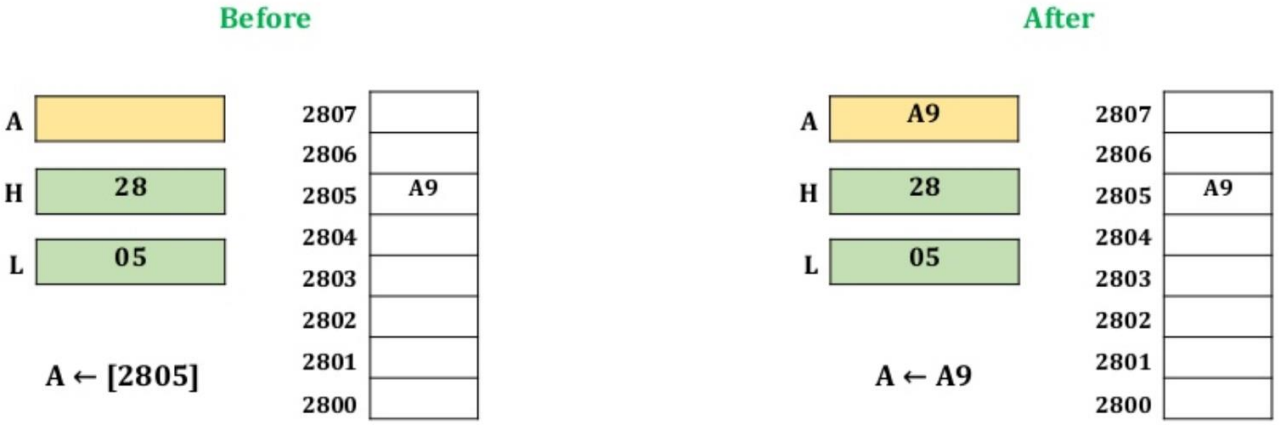


شیوه‌های آدرس‌دهی

6. آدرس‌دهی ثباتی غیرمستقیم (register indirect):



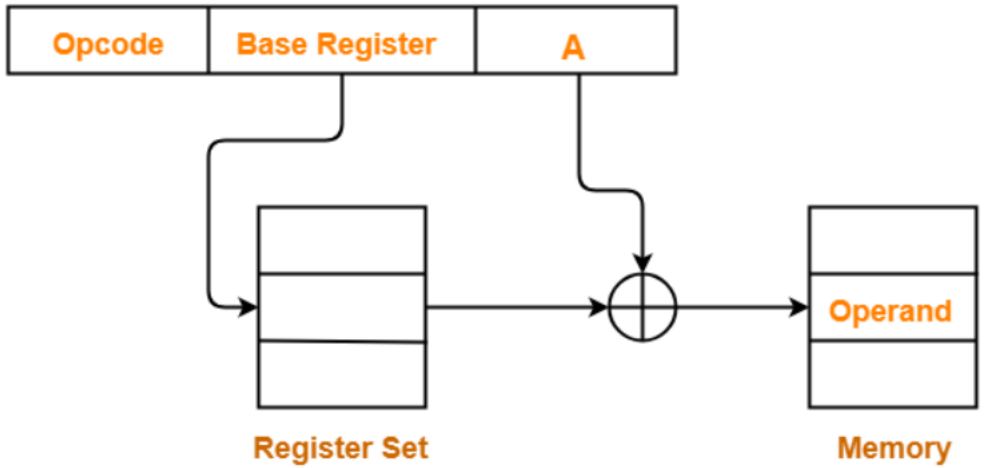
- آدرس داده موردنظر درون یک ثبات است
- در واقع داده به‌صورت pointer در دسترس است
- عملیات آرایه‌ای را ساده‌تر می‌کند
- Add [R] داخل ثبات R آدرسی موجود است که به خانه شامل داده موردنظر در حافظه اصلی اشاره دارد



شیوه‌های آدرس‌دهی

7. آدرس‌دهی نسبی با ثبات پایه (base addressing):

- کامپیوترها برای این نوع آدرس‌دهی یک ثبات پایه دارند (base register)
- انتقال نقطه شروع داده ساختار به مکان دیگر (آدرس شروع آرایه قابل تغییر و آفست آن ثابت است)
  - آدرس شروع در ثبات پایه و آفست در بخش آدرس
- مناسب برای انتقال یک برنامه در حافظه

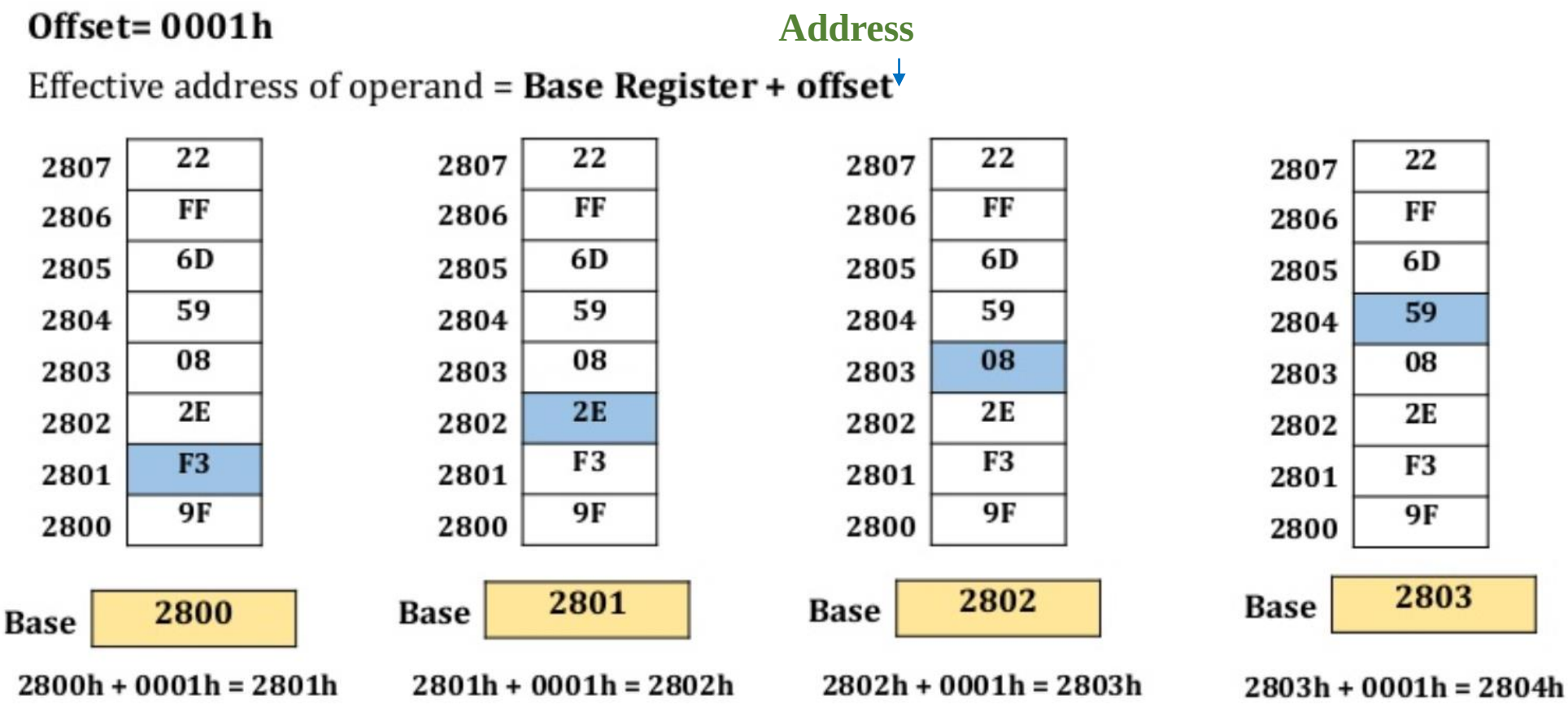


Effective Address

= Content of Base Register + Address part of the instruction

شیوه‌های آدرس‌دهی

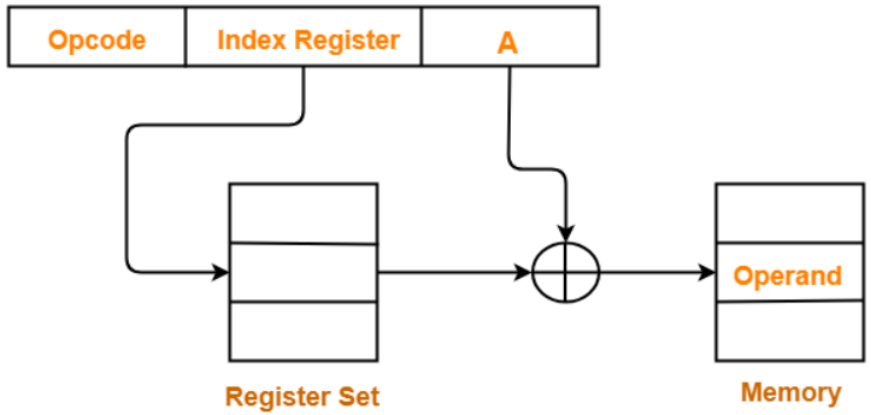
7. آدرس‌دهی نسبی با ثبات پایه (base addressing):



شیوه‌های آدرس‌دهی

8. آدرس‌دهی شاخص‌دار (indexed addressing):

- مشابه حالت قبل است فقط در اینجا با ثبات ایندکس کار می‌کند و آدرس همان شروع داده است
- زمانی استفاده می‌شود که ایندکس متغیر است (مانند حلقه‌ها) و بیشتر برای پیمایش آرایه استفاده می‌شود
- از طریق ایندکس‌های مختلف به خانه‌های بعدی آدرس می‌رسیم (آدرس در حافظه = محتوای ثبات ایندکس + آدرس)
- Add Value [SI]: آدرس شروع ثابت و ایندکس متغیر



Effective Address

= Content of Index Register + Address part of the instruction

شیوه‌های آدرس‌دهی

8. آدرس‌دهی شاخص‌دار (indexed addressing):