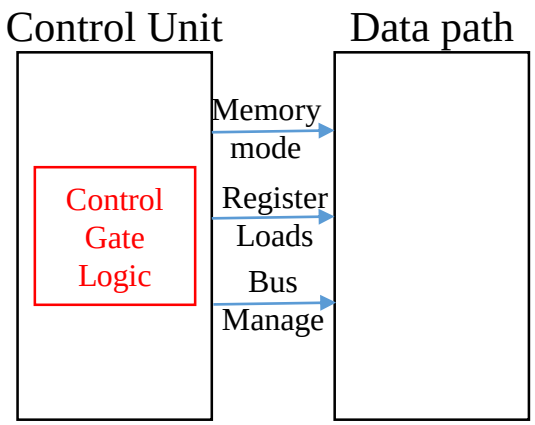
بنام خداوند جان و خرد

معماری کامپیوتر

جلسه بیست و سوم: پیاده‌سازی واحد کنترل (ریزبرنامه‌ریزی شده)

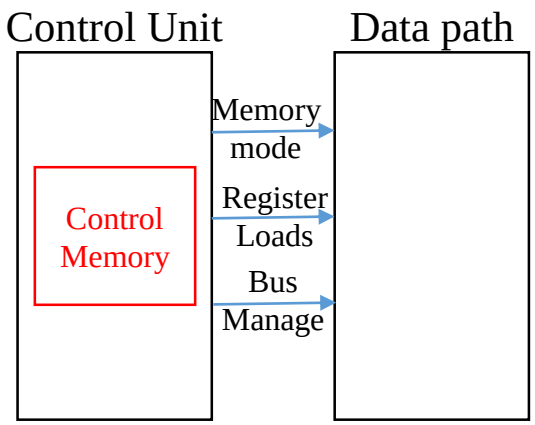
شیوه طراحی سیم‌بندی شده (Hard-wired)

- تبدیل مجموعه دستورالعمل‌ها به ریزدستورالعمل
  - RTL: Register Transfer Language
  - برحسب چرخه fetch-decode-operands-execute در چه مرحله هستیم
  - در هر کلاک چه عملیاتی انجام می‌شود
  - هر ریزدستورالعمل: دو بخش شرط و عملیات
    - برحسب شرایط و ثبات‌های مقصد(سمت چپ)، طراحی پایه‌های load و INC
    - برحسب شرایط و ثبات‌های درگیر در عملیات (سمت راست)، طراحی مدیر باس



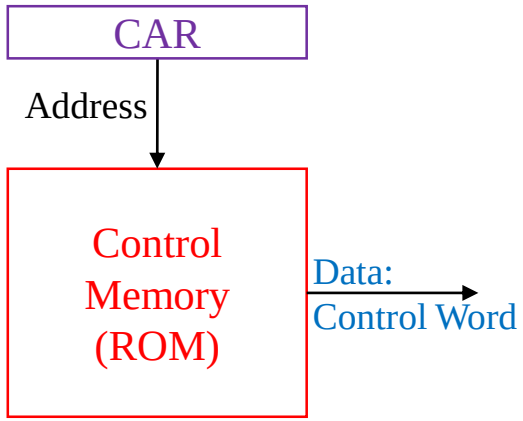
شیوه طراحی ریزبرنامه‌پذیر (Micro-programmed)

- واحد کنترل قابل برنامه‌ریزی است
  - در صورت نیاز به تغییر، برنامه عوض می‌شود و نیاز به جایگزینی سخت‌افزار نیست
  - ساختار واحد کنترل یک حافظه برنامه‌پذیر است (به‌جای مدار ترکیبی)
    - عملیات کنترلی توسط ریزدستورالعمل‌ها انجام می‌شوند
    - روال اجرای دستورات توسط برنامه موجود در حافظه کنترلی (ROM)
    - کنترلها (control words) توسط حافظه کنترلی تولید می‌شود



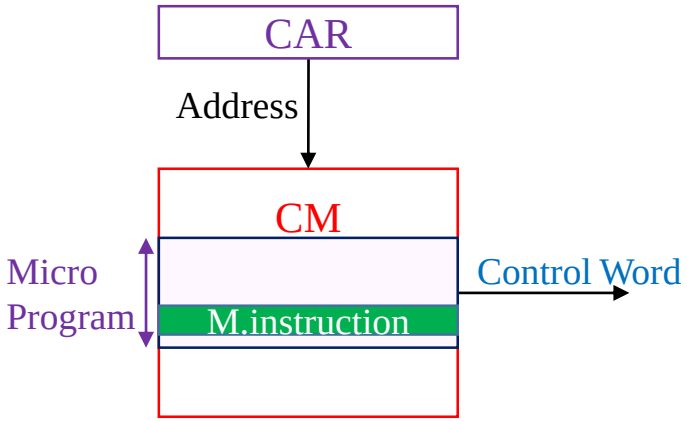
شیوه طراحی ریزبرنامه‌پذیر (Micro-programmed)

- ساختار و روال عملکرد حافظه کنترلی
  - از جنس ROM است و یک بخش آدرس و یک بخش داده دارد
    - آدرس از واحد CAR(Control Address Register) تامین می‌شود
    - داده، همان کنترلهای خروجی واحد کنترل است (code words)
      - هر خط حافظه کنترلی یک ریزدستورالعمل از ریزبرنامه است
      - هر ریزدستورالعمل متشکل از تعدادی ریزعملگر است که موازی اجرا می‌شوند



شیوه طراحی ریزبرنامه‌پذیر (Micro-programmed)

- ریز برنامه (micro-program)
  - مجموعه ریزدستورالعمل‌های یک دستور که در حافظه برنامه‌پذیر ذخیره می‌شوند و سیگنال‌های کنترلی براساس آن‌ها تولید می‌شود
- ریزدستورالعمل (micro-instruction)
  - بخشی از دستورالعمل که قابل اجرا در یک پالس کلاک است
  - شامل انتقال مقدار بین ثبات‌ها و حافظه با هدف اجرای دستور است



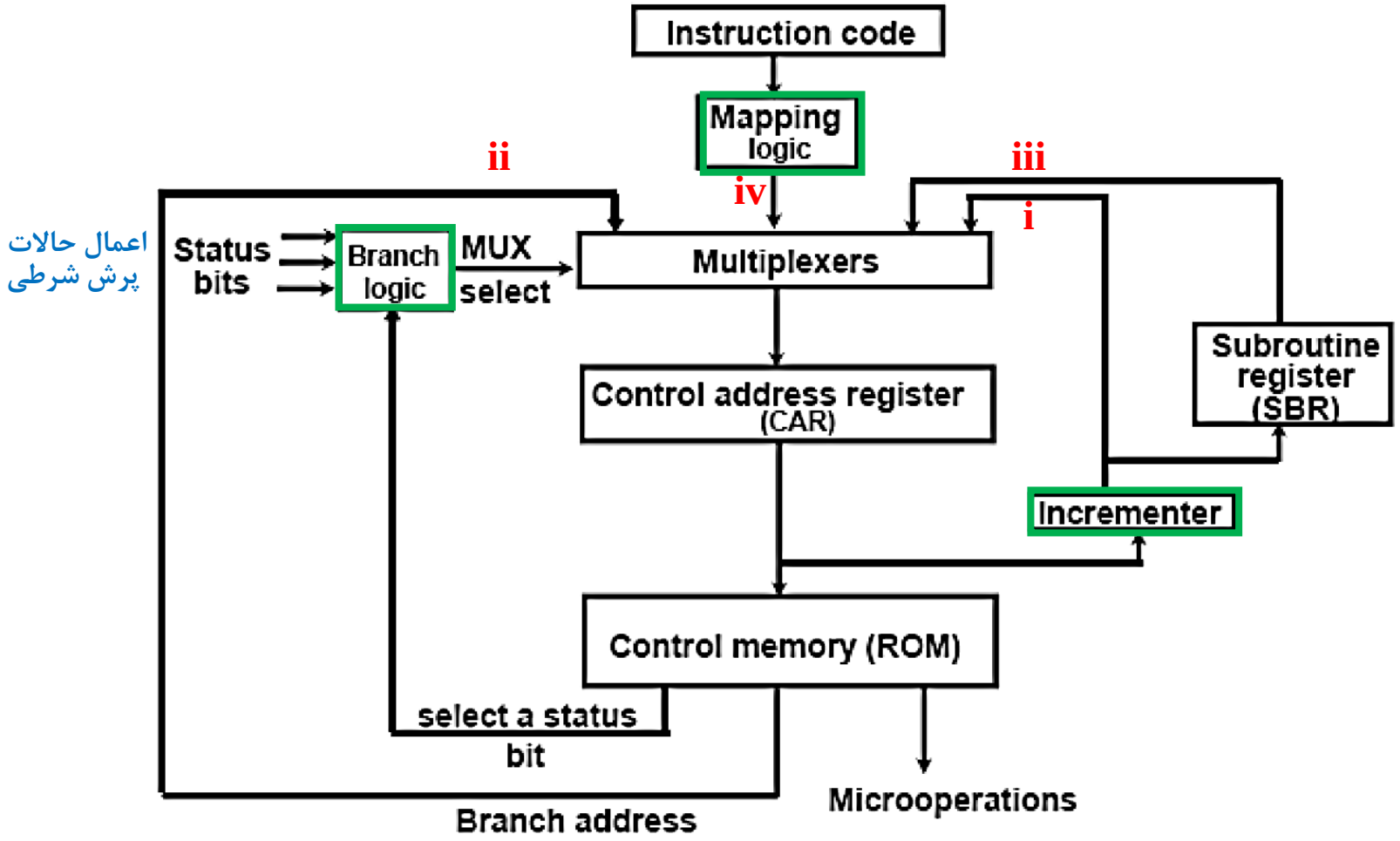
شیوه طراحی ریزبرنامه‌پذیر (Micro-programmed)

- ساختار و روال عملکرد حافظه کنترلی
  - روال عملکرد
    - ابتدا آدرس، تامین شده و به حافظه کنترلی داده می‌شود (CAR)
    - به مکانی که آدرس مشخص شده رفته و داده آن را استخراج می‌شود (شیوه ذخیره داده مهم است)
    - قسمتی از داده که مربوط به اطلاعات کنترلی است در اختیار data path قرار می‌گیرد
    - آدرس بعدی تولید می‌شود (address sequencer)
      - از روی اطلاعات مکان فعلی حافظه و سایر اطلاعات

شیوه طراحی کنترلر ریزبرنامه‌پذیر (Micro-programmed)

- **واحد تولید آدرس بعدی (Next Address Sequencer Logic/Address Sequencer)**
  - هدف: تعیین ترتیب آدرس‌ها در حافظه کنترل (CM)
  - حالت‌های مختلف ایجاد می‌شود پس با یک MUX طراحی می‌کنیم
    i. **اجرای مرتب:** آدرس بعدی آدرس فعلی (یک واحد increment)
    ii. **اجرای Branch**: آدرس بعدی با پرش (branch) که آدرس پرش از کلمه کنترلی استخراج می‌شود
    iii. **اجرای Subroutine**: آدرس بعدی مربوط به یک روتین، فراخوانی با دستور call (ذخیره آدرس بازگشت در SBR)
    iv. **نگاشت:** هر دستور یک روتین ذخیره شده در حافظه کنترلی دارد که براساس کدعملیاتی به شروع آن می‌رویم
      - تبدیل opcode به آدرس حافظه کنترلی

واحد تولید آدرس بعدی



واحد تولید آدرس بعدی

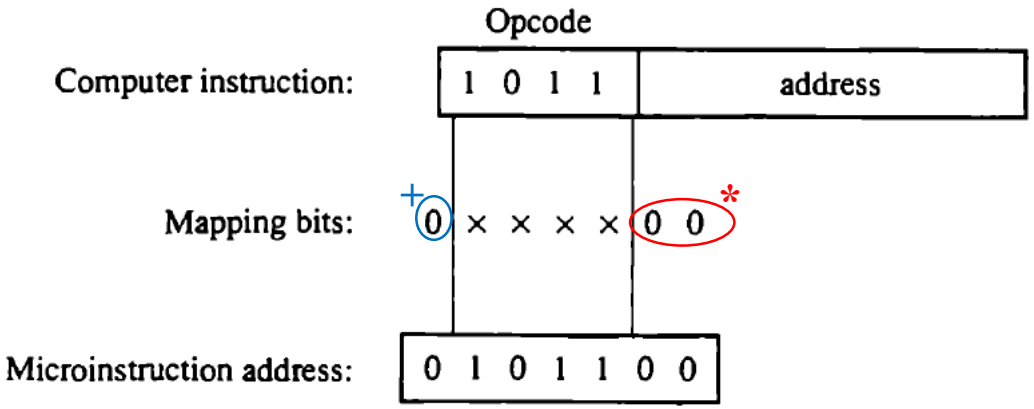
- در این طراحی، یک SBR داشتیم پس یک فراخوانی مدیریت می‌شود
  - برای اجرای چندین فراخوانی تو در تو می‌توان این ثبات را مشابه پشته طراحی کرد
- واحد نگاشت با هدف استخراج آدرس اجرای هر دستور از روی کدعملیاتی آن
  - مشابه برنامه‌نویسی event-driven و ورودی آن از IR است
  - تبدیل کدعملیاتی به آدرس شروع روتین با تنظیم تعداد بیت‌ها برحسب آدرس‌ها
    - طراحی به‌صورت یک مدار ترکیبی یا یک حافظه برنامه‌پذیر دیگر

واحد تولید آدرس بعدی-نگاشت

- **نگاشت بیتی:**
  - اگر فرض کنیم هر روتین در چهار خط حافظه قرار دارد
  - کدعملیاتی (opcode) چهاربیتی باشد
  - چند بیت افزونه به opcode نیاز داریم؟

واحد تولید آدرس بعدی-نگاشت

- اگر فرض کنیم هر روتین در چهار خط حافظه قرار دارد (*)
  - نیاز به دو بیت در بخش کم‌ارزش داریم
- اگر ۴ خط برای ذخیره‌سازی کم بود، یک فضای رزرو درنظر می‌گیریم (+)
- در مجموع با ۷ بیت آدرس‌سازی می‌کنیم
  - فضای حافظه دارای ۱۲۸ خط است

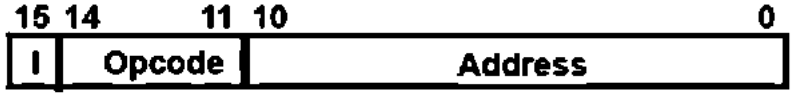


شیوه طراحی کنترلر ریزبرنامه‌پذیر (Micro-programmed)

- **ذخیره داده (microinstruction) در حافظه کنترلی**
  - تاکنون دیدیم که در هر خط حافظه کنترلی یک ریزدستورالعمل داریم
    - اطلاعات ریزعملگرها
    - اطلاعات استخراج آدرس بعدی
    - اطلاعاتی برای ساخت کلمه کنترلی و ارسال به مسیر داده
  - برای طراحی این بخش، یک مثال دیگر می‌بینیم.
    - طراحی کنترلر یک کامپیوتر ساده

شیوه طراحی کنترلر ریزبرنامه‌پذیر (Micro-programmed)

- یک کامپیوتر نمونه با چهار دستورالعمل درنظر می‌گیریم:
  - مجموعه دستورالعمل‌ها (ISA): Add, Branch, Store, Exchange (همگی نوع حافظه‌ای)
  - فرمت دستورالعمل‌ها: ۱۱ بیت آدرس و ۴ بیت کدعملیاتی و ۱ بیت نوع آدرس‌دهی



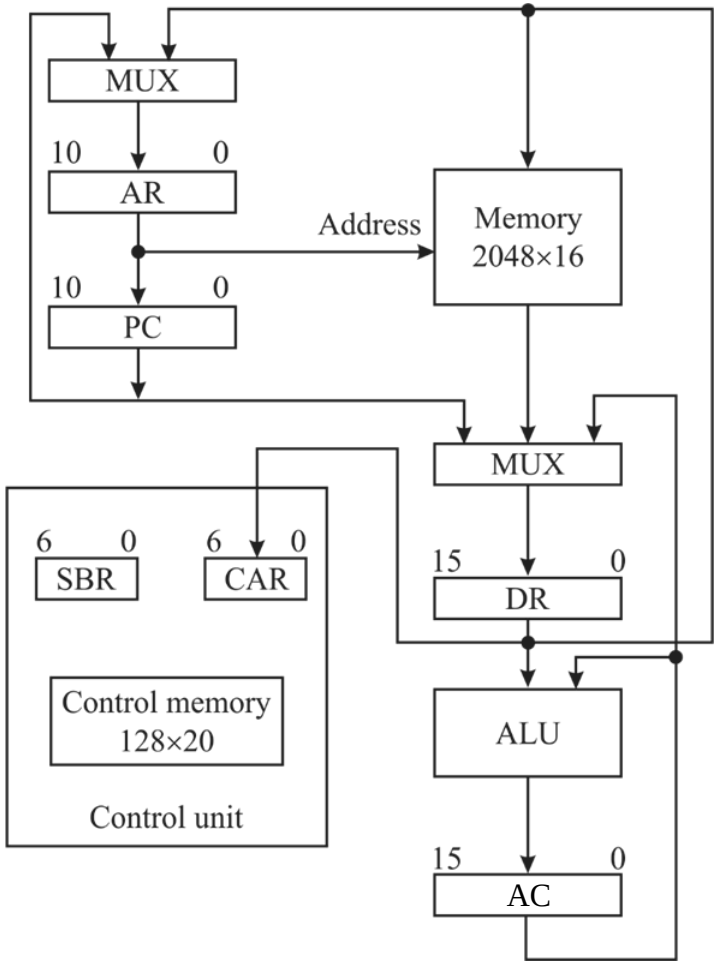
  - کدعملیاتی دستورالعمل‌ها:
  - EA: Effective Address

Symbol	OP-code	Description
ADD	0000	AC ← AC + M[EA]
BRANCH	0001	if (AC < 0) then (PC ← EA)
STORE	0010	M[EA] ← AC
EXCHANGE	0011	AC ← M[EA], M[EA] ← AC

شیوه طراحی کنترلر ریزبرنامه‌پذیر (Micro-programmed)

- مسیر داده کامپیوتر نمونه با چهار دستورالعمل متشکل است از:
  - حافظه با 2^{11} خانه و هر خانه ۱۶ بیتی
  - ثبات AR یازده بیتی داریم برای آدرس‌دهی
  - داده به ثبات DR می‌رود که ۱۶ بیتی است
  - ثبات PC داریم که با AR در تماس است
  - ALU و ثبات AC داریم

شیوه طراحی کنترلر ریزبرنامه‌پذیر (Micro-programmed)



- ساختار پردازنده کامپیوتر نمونه چهاردستورالعملی:
  - ثبات دستورالعمل نداریم چگونه مدیریت کنیم؟