بنام خداوند جان و خرد

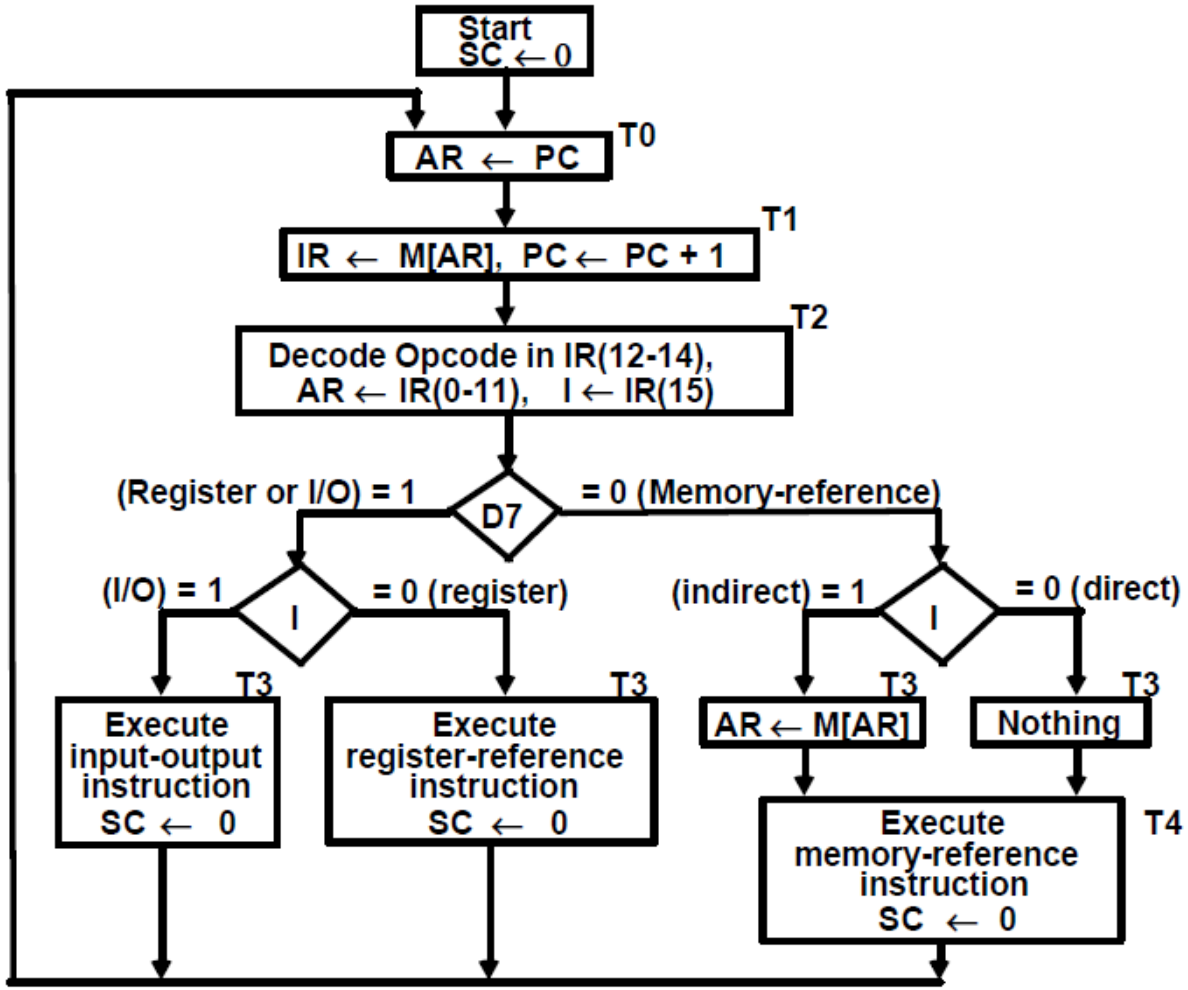
معماری کامپیوتر

جلسه بیست‌ودوم: پیاده‌سازی واحد کنترل (سیم‌بندی شده)

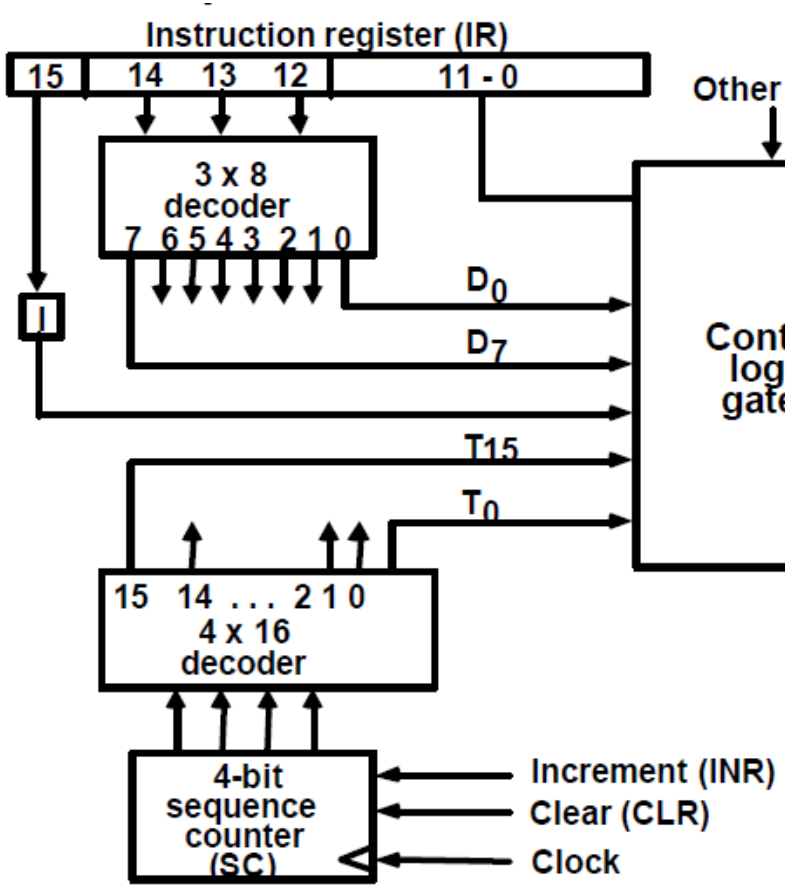
طراحی واحد کنترل

- معماری مجموعه دستورالعمل‌های کامپیوتر پایه (ISA)
- ساختار مسیر داده و واحد کنترل
- طراحی واحد کنترل
  - مشخص کردن توالی اجرای هر دستورالعمل
  - مشخص کردن عملیات هر مرحله از چرخه دستورالعمل (برحسب کلاک): microinstruction
  - با هدف مدیریت باس و تنظیم ورودی‌های ثبات‌ها و حافظه در کلاک مشترک با مسیر داده

فلوچارت مراحل چرخه دستورالعمل



واحد کنترل کامپیوتر پایه



- تنظیم ورودی‌های LD,CLR,IN
- مدیریت باس (MUX)

دستورات قابل اجرا در کامپیوتر پایه

	r:	SC ← 0
CLA	rB_{11}:	AC ← 0
CLE	rB_{10}:	E ← 0
CMA	rB_9:	AC ← AC’
CME	rB_8:	E ← E’
CIR	rB_7:	AC ← shr AC, AC(15) ← E, E ← AC(0)
CIL	rB_6:	AC ← shl AC, AC(0) ← E, E ← AC(15)
INC	rB_5:	AC ← AC + 1
SPA	rB_4:	if (AC(15) = 0) then (PC ← PC+1)
SNA	rB_3:	if (AC(15) = 1) then (PC ← PC+1)
SZA	rB_2:	if (AC = 0) then (PC ← PC+1)
SZE	rB_1:	if (E = 0) then (PC ← PC+1)
HLT	rB_0:	S ← 0 (S is a start-stop flip-flop)

دستورات ثباتی با شرط

$D_7I'T_3B[i]$

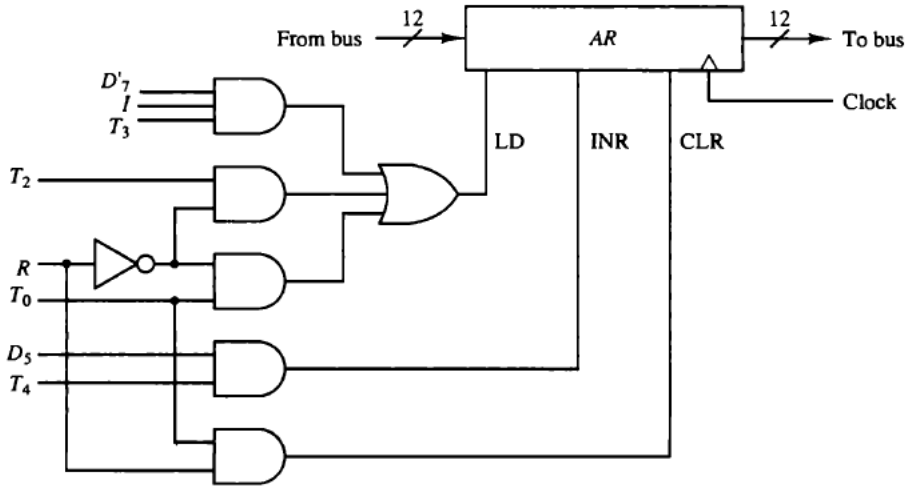
Symbol	Operation Decoder	Symbolic Description
AND	D_0	AC ← AC ∧ M[AR]
ADD	D_1	AC ← AC + M[AR], E ← C_{out}
LDA	D_2	AC ← M[AR]
STA	D_3	M[AR] ← AC
BUN	D_4	PC ← AR
BSA	D_5	M[AR] ← PC, PC ← AR + 1
ISZ	D_6	M[AR] ← M[AR] + 1, if M[AR] + 1 = 0 then PC ← PC+1

دستورات حافظه‌ای با شرط

$D_{0\le i\le 6}T_4$

واحد کنترل کامپیوتر پایه

- در انتها براساس ریزدستورات، پایه‌های متصل به باس را برنامه‌ریزی می‌کنیم
  - مثلا همه زمان‌هایی که AR مقداردهی می‌شود را باهم OR می‌کنیم
  - نتیجه این OR را به پایه load ثبات AR روی باس می‌دهیم



- $LD[AR] \leftarrow R'T_0+R'T_2+D_7'IT_3$

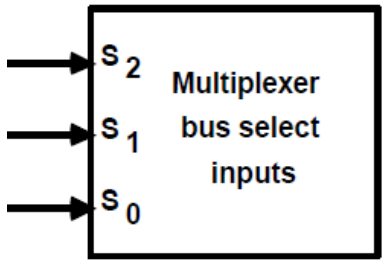
$AR \leftarrow PC,\ AR \leftarrow IR(0\text{-}11),AR \leftarrow M[AR]$

- $CLR[AR] \leftarrow RT_0$
- $INR[AR] \leftarrow D_5T_4$

واحد کنترل کامپیوتر پایه

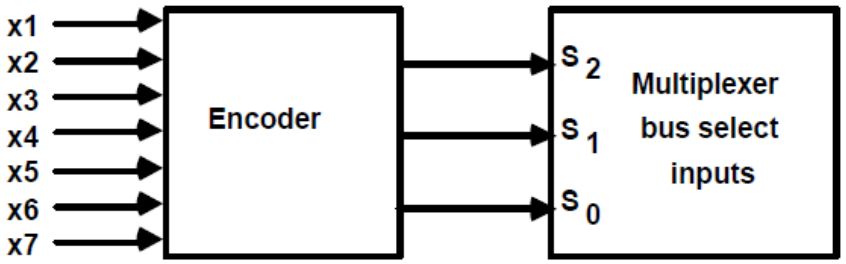
- مدیریت باس مشترک با استفاده از MUX با سه خط انتخاب
  - فعال شدن هر عنصر به‌معنای دسترسی به باس است
    - دسترسی به باس با هدف قرار دادن داده
  - در هر لحظه فقط یک عنصر به باس دسترسی داشته باشد
    - افزودن چه ماژول سخت‌افزاری؟

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



واحد کنترل کامپیوتر پایه

- مدیریت باس مشترک با استفاده از MUX با سه خط انتخاب
  - به هر عنصر روی باس عددی می‌دهیم (X_i)
  - فعال شدن هر عدد به‌معنای در اختیار گرفتن باس است
  - فعال شدن X_iها برچه اساسی؟

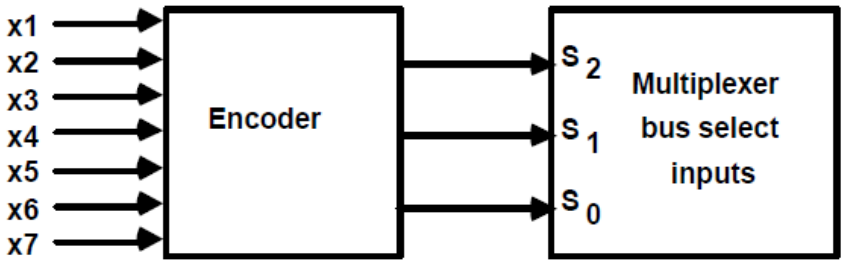


واحد کنترل کامپیوتر پایه

- مدیریت باس مشترک با استفاده از MUX با سه خط انتخاب
  - ریزدستوراتی که به مقدار x_i نیاز دارند را or می‌کنیم

$$X1 = D_4T_4 + D_5T_5$$

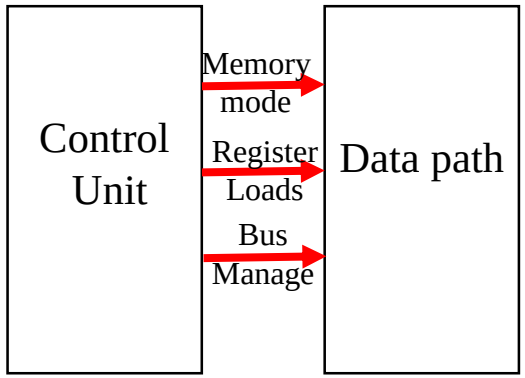
- D4T4:PC←AR
- D5T5:PC←AR



x1	x2	x3	x4	x5	x6	x7	S2	S1	S0	selected register
0	0	0	0	0	0	0	0	0	0	none
1	0	0	0	0	0	0	0	0	1	AR
0	1	0	0	0	0	0	0	1	0	PC
0	0	1	0	0	0	0	0	1	1	DR
0	0	0	1	0	0	0	1	0	0	AC
0	0	0	0	1	0	0	1	0	1	IR
0	0	0	0	0	1	0	1	1	0	TR
0	0	0	0	0	0	1	1	1	1	Memory

طراحی واحد کنترل

- تا اینجا با وظایف، اجزا و سازمان واحد کنترل آشنا شدیم
  - هر دستور را در قالب ریزدستورالعمل نوشتیم
    - در هر کلاک دقیقا چه عملیاتی باید انجام شود
- درنتیجه با دانستن اینکه در هر کلاک اطلاعات را از کجا می‌خوانیم و به کجا می‌دهیم
  - طراحی اجزای واحد کنترل را در قالب مدارات ترکیبی انجام دادیم
  - باهدف تولید سیگنال‌های کنترلی مسیر داده در حین اجرای دستورات

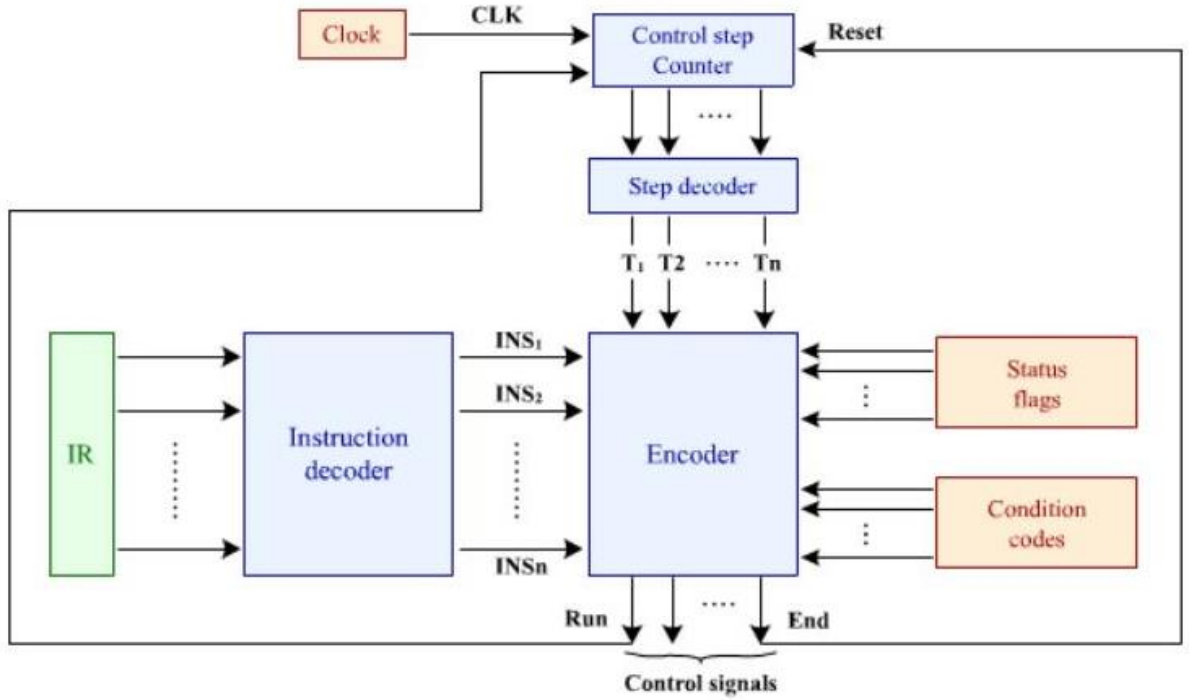


طراحی واحد کنترل

- شیوه‌های طراحی واحد کنترل:
  - Hardwired (سیم‌بندی شده): طراحی سخت‌افزاری واحد کنترل و لحیم کردن سیم‌ها
    - در این طراحی با گیت کار می‌کنیم و طراحی استاتیک است و تحمل تغییرات آتی را ندارد (تا الان انجام دادیم)
  - Micro-programmed (ریزبرنامه‌پذیر): قراردادن همه اجزا واحد کنترل داخل یک بلوک برنامه‌پذیر
    - یک کنترلر داخل کنترلر دیگر می‌گذاریم و با تغییرات کافی‌است برنامه‌ریزی را عوض کنیم و طراحی پویاست
    - یک واحد حافظه کنترلی (control memory) داخل واحد کنترل قرار می‌دهیم

شیوه طراحی سیم‌بندی شده (Hardwired)

- واحد کنترل به‌صورت یک مدار ترکیبی ساخته می‌شود
- ورودی‌هایی را می‌گیرد (clock, IR, flag) و آن‌ها را به سیگنال‌های کنترلی تبدیل می‌کند

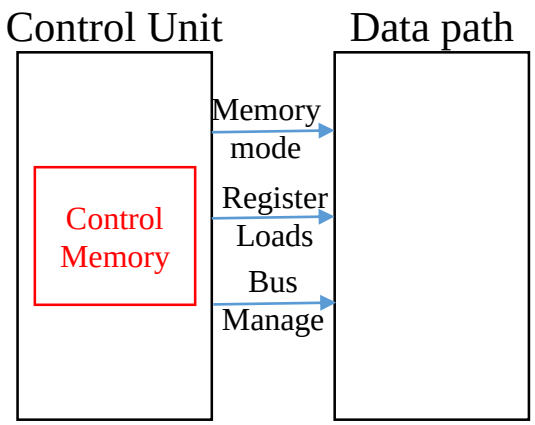


شیوه طراحی سیم‌بندی شده (Hardwired)

- ویژگی‌ها
  - سرعت بیشتر به‌دلیل پیاده‌سازی سخت‌افزاری
  - ممکن است قابلیت پیاده‌سازی دستورات پیچیده را نداشته باشد (کامپیوترهای CISC)
  - اتصالات، شیوه عملکرد و دستورات غیرقابل تغییر هستند
    - هر تغییر مستلزم طراحی دوباره و ساخت واحد کنترل از ابتدا می‌باشد

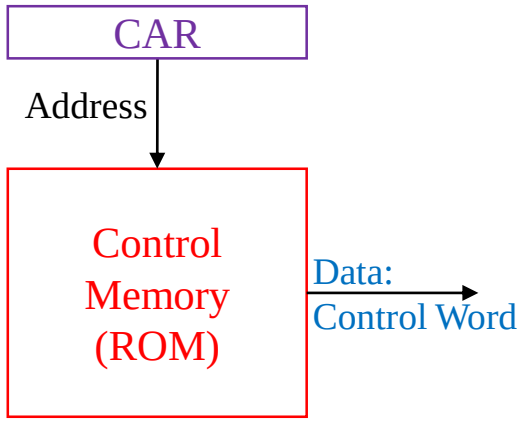
شیوه طراحی ریزبرنامه‌پذیر (Micro-programmed)

- 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
    - هر کد عملیاتی، به یک قسمت از حافظه برنامه‌پذیر نگاشت 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)

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