# ارزیابی کارایی ۲

... معماری کامپیوتر

۱۳۹۱-۱۱-۱۳

جلسه‌ی سوم


دانشگاه شهید بهشتی

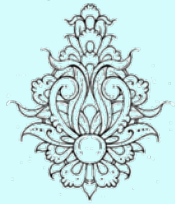
دانشکده‌ی مهندسی برق و کامپیوتر

زمستان ۱۳۹۱

احمد محمودی ازناوه

# فهرست مطالب

- مروری بر جلسه‌ی پیش
- کارایی
  - دستورالعمل‌ها با سیکل‌های متفاوت



# دستورات با سیکل‌های متفاوت

- در صورتی که یک cpu دارای دستورات با سیکل‌های متفاوت باشد، تعداد پالس ساعت برای اجرای یک برنامه به صورت زیر محاسبه می‌شود.

$$\text{Clock Cycles} = \sum_{i=1}^{n} (\text{CPI}_i \times \text{Instruction Count}_i)$$

- و CPI کلی با میانگین وزن‌دهی شده به دست می‌آید:

$$\text{CPI} = \frac{\text{Clock Cycles}}{\text{Instruction Count}} = \sum_{i=1}^{n} \left( \text{CPI}_i \times \frac{\text{Instruction Count}_i}{\text{Instruction Count}} \right)$$

Relative frequency

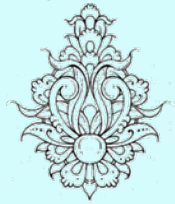
# مثال

- برای اجرای یک تکه کد، دو شیوه قابل به کارگیری است، کدامیک سریع‌تر خواهد بود؟

| Class | A | B | C |
|---|---|---|---|
| CPI for class | 1 | 2 | 3 |
| IC in sequence 1 | 2 | 1 | 2 |
| IC in sequence 2 | 4 | 1 | 1 |

- Sequence 1: IC = 5
  - Clock Cycles
    = 2×1 + 1×2 + 2×3
    = 10
  - Avg. CPI = 10/5 = 2.0

- Sequence 2: IC = 6
  - Clock Cycles
    = 4×1 + 1×2 + 1×3
    = 9
  - Avg. CPI = 9/6 = 1.5

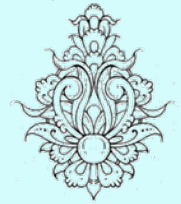


# جمع‌بندی در مورد کارایی

دورنی

$$\text{CPU Time} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Clock cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Clock cycle}}$$

- کارایی یک برنامه به عوامل زیر بستگی دارد:
  - الگوریتم IC CPI
  - زبان برنامه‌نویسی CPI IC
  - کامپایلر CPI IC
  - معماری مجموعه‌ی دستورالعمل‌ها (ISA) IC CPI T



# تمرین کلاسی

| Op | Freq | $CPI_i$ | Freq x $CPI_i$ | | | |
|---|---|---|---|---|---|---|
| ALU | 50% | 1 | .5 | .5 | .5 | .25 |
| Load | 20% | 5 | 1.0 | .4 | 1.0 | 1.0 |
| Store | 10% | 3 | .3 | .3 | .3 | .3 |
| Branch | 20% | 2 | .4 | .4 | .2 | .4 |
| | | | $\Sigma =$ 2.2 | 1.6 | 2.0 | 1.95 |

- در صورتی که با به‌کارگیری حافظه‌ی نهان بهتر، دستورات نوشتن به میانگین دو سیکل برسند، سرعت تا چه میزان بهبود می‌یابد؟

*CPU time new = 1.6 x IC x CC so 2.2/1.6 means 37.5% faster*

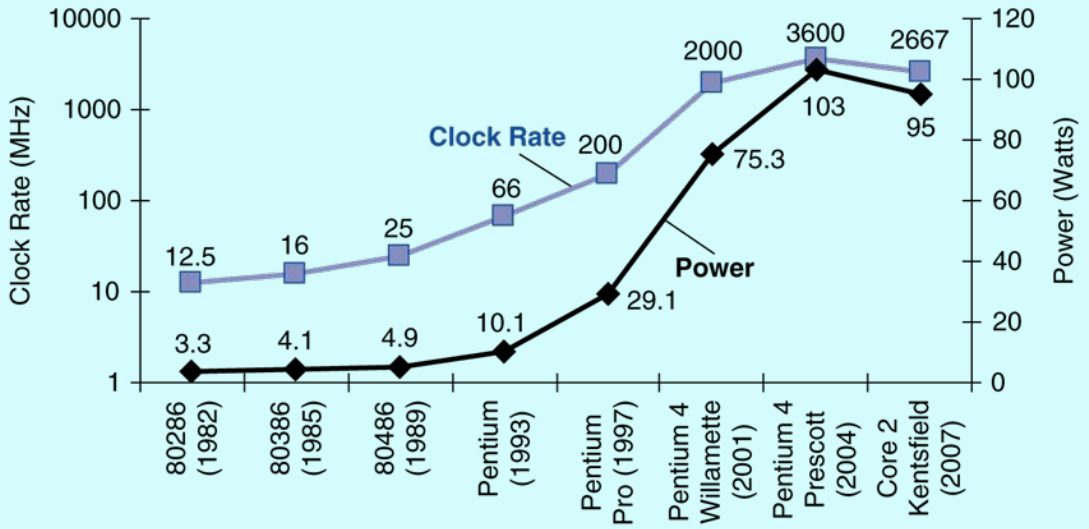
- در صورتی که با پیش‌بینی دستورات پرش شرطی سرعت میانگین زمان اجرای آن‌ها نصف شود، سرعت چه تغییری خواهد کرد؟

*CPU time new = 2.0 x IC x CC so 2.2/2.0 means 10% faster*

درصورتی که دو دستور همزمان در ALU اجرا شوند، سرعت چه تغییری خواهد کرد؟

*CPU time new = 1.95 x IC x CC so 2.2/1.95 means 12.8% faster*

# رویه تغییرات توان مصرفی




- در تکنولوژی CMOS

$$\text{Power} = \text{Capacitive load} \times \text{Voltage}^2 \times \text{Frequency}$$

×30 | 5V → 1V | ×1000

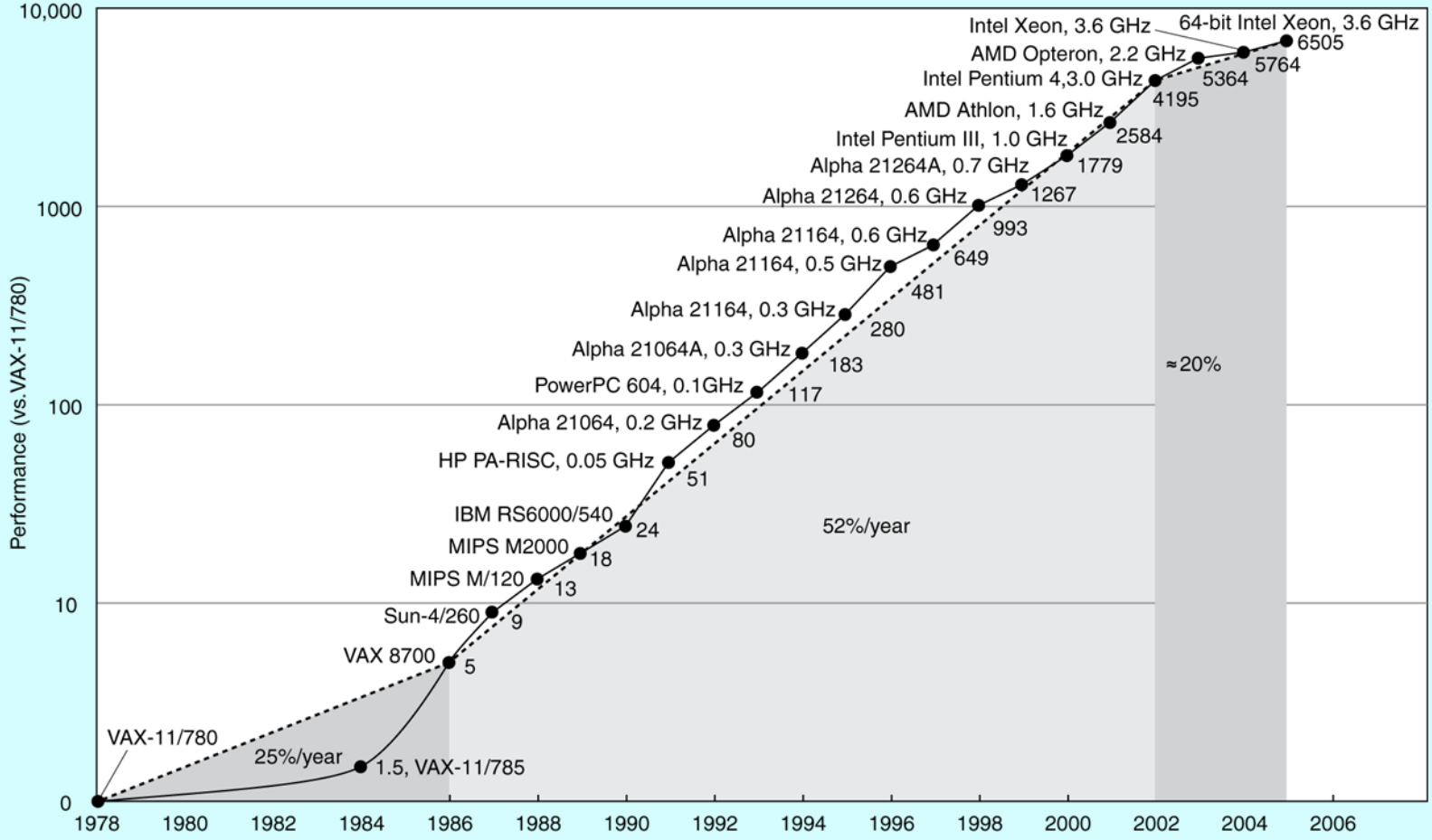
# مثال

- در صورتی که در یک پردازنده طراحی به گونه‌ای تغییر کند که ولتاژ، فرکانس پالس ساعت و بار خازنی هر یک پانزده‌درصد کاسته شوند، توان مصرفی چه تغییری می‌کند؟

$$\frac{P_{new}}{P_{old}} = \frac{C_{old} \times 0.85 \times (V_{old} \times 0.85)^2 \times F_{old} \times 0.85}{C_{old} \times V_{old}^{\ 2} \times F_{old}} = 0.85^4 = 0.52$$

- بیش از این نمی‌توان ولتاژ را کاهش داد.
- روش‌های متفاوتی برای خنک کردن cpu به کار گرفته شده است.

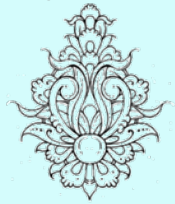
# حرکت به سوی پردازش موازی



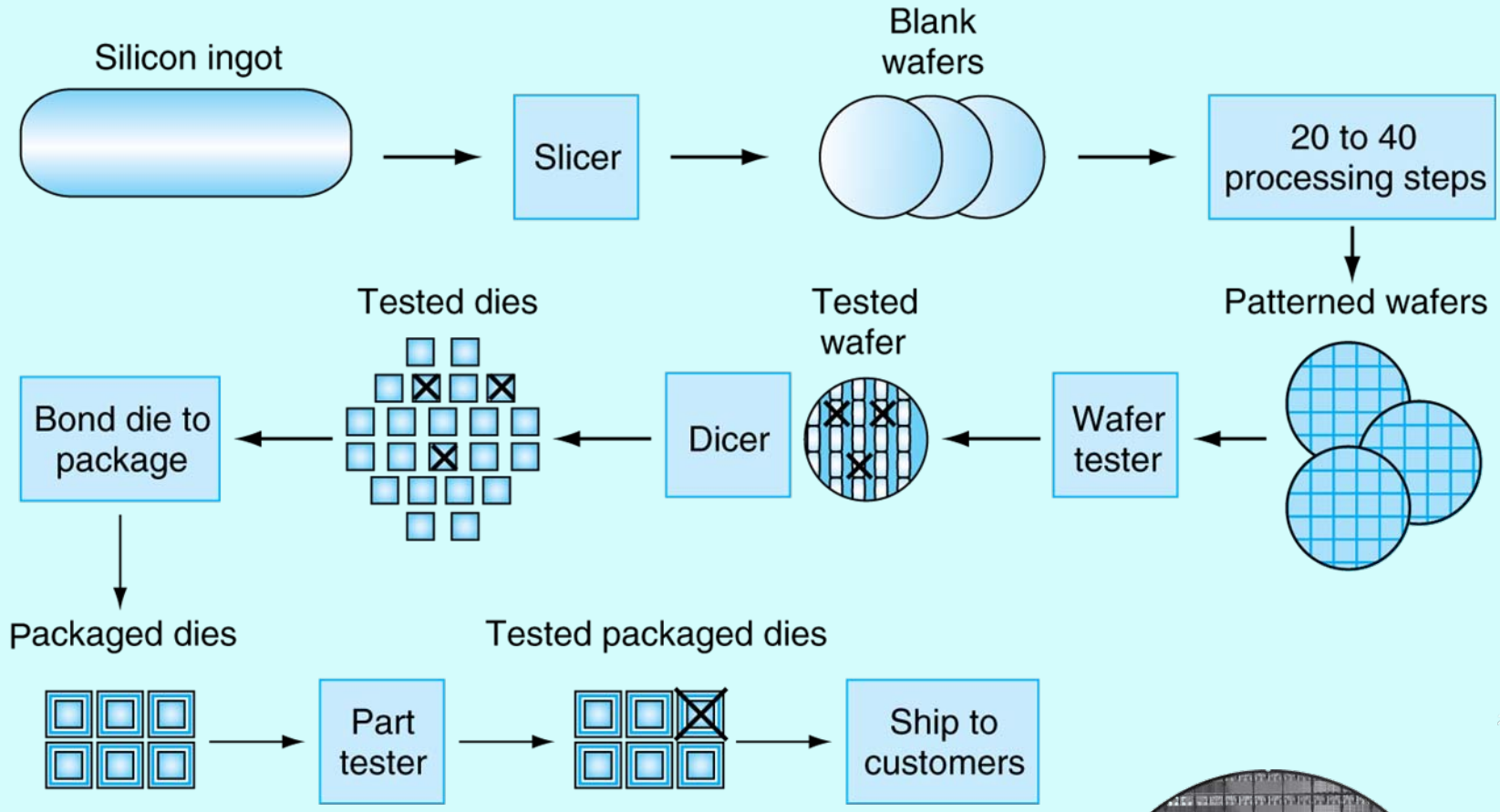
# چندپردازشی

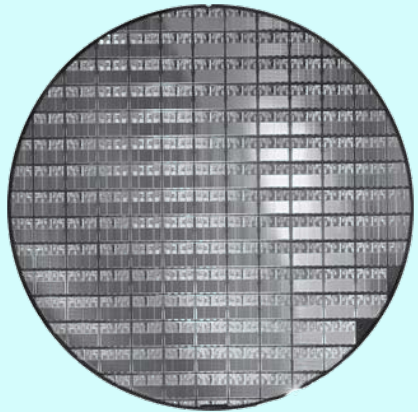
## multiprocessing

- پردازنده‌های چند هسته‌ای
  - بیش از یک پردازنده در یک تراشه
- نیاز به نوشتن برنامه‌های که قابلیت اجرا شدن به صورت موازی را دارند.
- دشواری‌های برنامه‌نویسی به صورت موازی
  - باید کارا باشد، در این حالت تنها پاسخ درست مد نظر نیست
  - توزیع بار (Load balancing)
  - کاهش ارتباط و نیاز به هماهنگ کردن



# ساخت تراشه



**300mm wafer, 117 chips, 90nm technology**

# محک آزمون برای بررسی کارایی

- workload
  - مجموعه برنامه‌ای که بر روی یک کامپیوتر اجرا می‌شود.
- محک آزمون :
  - مجموعه برنامه‌هایی که برای ارزیابی یک سیستم کامپیوتری مورد استفاده قرار می‌گیرد.
- SPEC برای ارزیابی کارایی نمونه‌ی محک عرضه می‌کند.
  - SPEC CPU2006، عملیات ورودی خروجی را در نظر نمی‌گیرد، و در نتیجه بر روی کارایی پردازنده تمرکز دارد.

**SPEC: (System Performance Evaluation Cooperative )**

# AMD Opteron X4

$$\sqrt[n]{\prod_{i=1}^{n} \text{Execution time ratio}_i}$$

| Name | Description | IC×10$^9$ | CPI | Tc (ns) | Exec time | Ref time | SPECratio |
|---|---|---|---|---|---|---|---|
| perl | Interpreted string processing | 2,118 | 0.75 | 0.40 | 637 | 9,777 | 15.3 |
| bzip2 | Block-sorting compression | 2,389 | 0.85 | 0.40 | 817 | 9,650 | 11.8 |
| gcc | GNU C Compiler | 1,050 | 1.72 | 0.47 | 24 | 8,050 | 11.1 |
| mcf | Combinatorial optimization | 336 | 10.00 | 0.40 | 1,345 | 9,120 | 6.8 |
| go | Go game (AI) | 1,658 | 1.09 | 0.40 | 721 | 10,490 | 14.6 |
| hmmer | Search gene sequence | 2,783 | 0.80 | 0.40 | 890 | 9,330 | 10.5 |
| sjeng | Chess game (AI) | 2,176 | 0.96 | 0.48 | 37 | 12,100 | 14.5 |
| libquantum | Quantum computer simulation | 1,623 | 1.61 | 0.40 | 1,047 | 20,720 | 19.8 |
| h264avc | Video compression | 3,102 | 0.80 | 0.40 | 993 | 22,130 | 22.3 |
| omnetpp | Discrete event simulation | 587 | 2.94 | 0.40 | 690 | 6,250 | 9.1 |
| astar | Games/path finding | 1,082 | 1.79 | 0.40 | 773 | 7,020 | 9.1 |
| xalancbmk | XML parsing | 1,058 | 2.70 | 0.40 | 1,143 | 6,900 | 6.0 |
| Geometric mean | | | | | | | 11.7 |

**High cache miss rates**

# قانون Amdahl

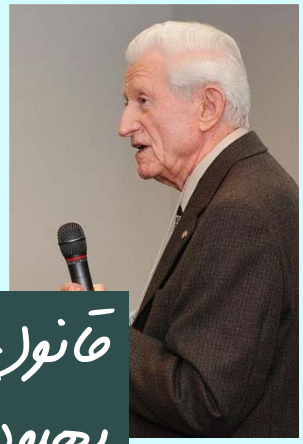
- برنامه‌ای را در نظر بگیرید که زمان پاسخ آن ۱۰۰ ثانیه است، ۸۰ ثانیه‌ی مربوط به عملیات ضرب می‌باشد، سرعت عملیات ضرب چند برابر شود تا سرعت برنامه پنج بر ابر شود؟

$$T_{improved} = \frac{T_{affected}}{improvement\ factor} + T_{unaffected}$$

$$20 = \frac{80}{n} + 20$$

**Amdahl's law**

دام: نباید انتظار داشت متناسب با بهبود یک بخش عملکرد کلی بهبود یابد شدنی نیست!



**قانون Amdhal**

بهبود کارایی سیستم ، هنگامی که بخشی از آن بهبود یابد

نتیجه: بهتر است قسمت‌های پرکاربرد بهبود یابند

- توان مصرفی X4
  - در صورت به‌کارگیری ۱۰۰ درصدی 295W
  - در صورت به‌کارگیری ۵۰ درصدی 246W
  - در صورت به‌کارگیری ۱۰ درصدی 180W
- سرورهای گوگل اغلب با ده تا پنجاه درصد ظرفیت کار می‌کنند و تنها یک درصد اوقات بار آن به صددرصد می‌رسد.

**Energy-proportional computing**

# معیاری دیگر برای ارزیابی کارایی

**MIPS: million instructions per second**

$$\text{MIPS} = \frac{\text{Instruction count}}{\text{Execution time} \times 10^6}$$

- در ISAهای متفاوت، توانایی‌ها فرق می‌کند.
- MIPS در یک کامپیوتر خاص برای برنامه‌های متفاوت، مقادیر متفاوتی خواهد داشت.
- برای یک برنامه با تعداد دستورات بیشتر اما سریع‌تر معیار خوبی نخواهد بود.

$$\text{MIPS} = \frac{\text{Instruction count}}{\frac{\text{Instruction count} \times \text{CPI}}{\text{Clock rate}} \times 10^6} = \frac{\text{Clock rate}}{\text{CPI} \times 10^6}$$