Ειδικά Θέματα Αρχιτεκτονικής και Προγραμματισμού Μικροεπεξεργαστών

Ενότητα 1: Εισαγωγή

Διδάσκων: Βαρτζιώτης Φώτιος
Τμήμα Πληροφορικής και Τηλεπικοινωνιών

Διάλεξη - Σχεδιασμός

Διάλεξη:

- Πέμπτη, 11:40 – 13:40

Εργαστήριο:

- Τρίτη, 12:40 – 13:40

ΑΠ:

- Τρίτη, 11:40 – 12:40

Εισαγωγή

Βιβλιογραφία:

1. Computer Organization and Design: The Hardware / Software interface, 5th Edition, John Hennessy and David Patterson, Morgan Kaufmann Publishers
2. Computer Architecture: A Quantitative Approach, 5th Edition, John Hennessy and David Patterson, Morgan Kaufmann Publishers
3. Διαφάνειες Θεωρίας και Εργαστηρίου (MS Teams)

Προαπαιτούμενα

Γνώσεις από το μάθημα «Αρχιτεκτονική Υπολογιστών»:

- Μνήμη:
  - Κύρια μνήμη, Cache, Αποθηκευτικά μέσα
- Κεντρική Μονάδα Επεξεργασίας
  - Δομή, Σετ εντολών, Διευθυνσιοδότηση
- Assembly
  - Γλώσσα μηχανής, βασικές αρχές

Περιεχόμενο μαθήματος - Θεωρία

Εβδομάδα 1	Εισαγωγή – Βιβλιογραφία, περιεχόμενο μαθήματος, προαπαιτούμενα, εξελίξεις στους επεξεργαστές
Εβδομάδα 2	Απόδοση των μικροεπεξεργαστών, παραδείγματα εφαρμογής
Εβδομάδα 3	Αρχιτεκτονική RISC, το παράδειγμα του MIPS, σετ εντολών
Εβδομάδα 4	Υλοποίηση καίριων μηχανισμών στην Αριθμητική και Λογική Μονάδα ενός μικροεπεξεργαστή, παραδείγματα εφαρμογής
Εβδομάδα 5	Μικροεπεξεργαστής MIPS: Σχεδίαση απλού κύκλου
Εβδομάδα 6	Μικροεπεξεργαστής MIPS: Σχεδίαση πολλαπλών κύκλων

Περιεχόμενο μαθήματος - Θεωρία

Εβδομάδα 7	Μικροεπεξεργαστής MIPS: Pipelining
Εβδομάδα 8	Μικροεπεξεργαστής MIPS: Forwarding, Stalls, Branch Prediction
Εβδομάδα 9	Δυναμική δρομολόγηση εντολών
Εβδομάδα 10	Στατική Δρομολόγηση εντολών
Εβδομάδα 11	Δίκτυα Διασύνδεσης
Εβδομάδα 12	Παράλληλη Επεξεργασία και Υπερβαθμωτή (superscalar) Οργάνωση
Εβδομάδα 13	Πολυπύρηνα Συστήματα.

Περιεχόμενο μαθήματος - Εργαστήριο

Εβδομάδα 1	Εγγραφές
Εβδομάδα 2	Εισαγωγή – περιβάλλον εργαλείου προγραμματισμού και προσομοίωσης
Εβδομάδα 3	Προγραμματισμός σε Assembly - παραδείγματα εφαρμογής
Εβδομάδα 4	Προγραμματισμός σε Assembly - παραδείγματα εφαρμογής, εργασία στο σπίτι και 1η αναφορά υλοποίησης
Εβδομάδα 5	Προγραμματισμός σε Assembly - παραδείγματα εφαρμογής, εργασία στο σπίτι και 2η αναφορά υλοποίησης
Εβδομάδα 6	Ανάθεση 1ης εργασίας αξιολόγησης: περιγραφή, διευκρινήσεις

Περιεχόμενο μαθήματος - Εργαστήριο

Εβδομάδα 7	Ερωτήσεις / διευκρινήσεις για την 1η εργασία
Εβδομάδα 8	Παράδοση 1ης εργασίας αξιολόγησης και ανάθεση 2ης εργασίας αξιολόγησης: περιγραφή, διευκρινήσεις. Εισαγωγή στο περιβάλλον πρόσθετου εργαλείου προγραμματισμού και προσομοίωσης (εάν απαιτείται από την 2η εργασία)
Εβδομάδα 9	Ερωτήσεις / διευκρινήσεις για την 2η εργασία. Εξέταση της 1ης εργασίας
Εβδομάδα 10	Ερωτήσεις / διευκρινήσεις για την 2η εργασία. Εξέταση της 1ης εργασίας
Εβδομάδα 11	Ερωτήσεις / διευκρινήσεις για την 2η εργασία
Εβδομάδα 12	Παράδοση 2ης εργασίας αξιολόγησης
Εβδομάδα 13	Εξέταση της 2ης εργασίας

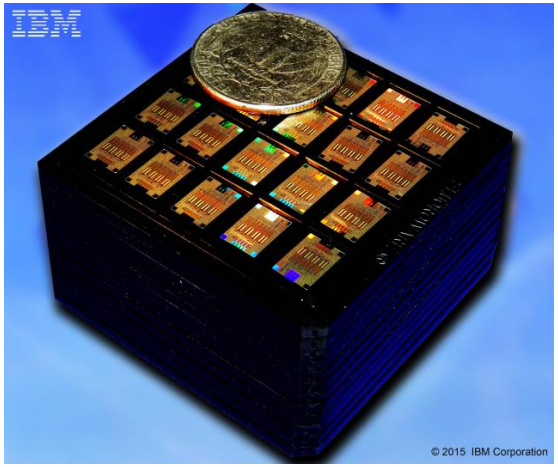
Προγραμματισμός και Προσομοίωση

- Περιβάλλον προγραμματισμού και προσομοίωσης μικροεπεξεργαστή
  - QtSpim
  - Pin – Intel
- Ασκήσεις σχετικές με:
  - Αριθμητικές - λογικές πράξεις
  - Διακλαδώσεις / Υπερχείλιση
  - Κλήση απλών διεργασιών και διεργασιών συστήματος
  - Εφαρμογές μικροεπεξεργαστών

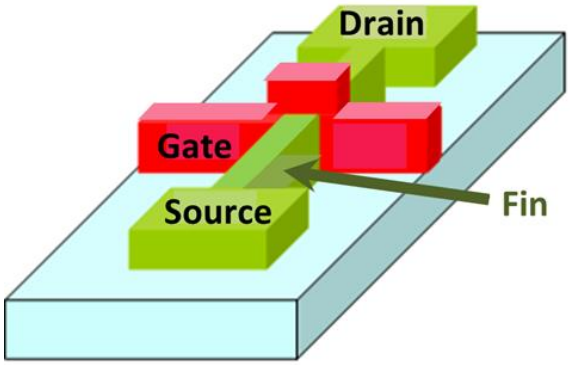
Αξιολόγηση - Επιτυχία

- Διάλεξη και Εργαστήριο
  - Η τελική βαθμολογία για το μάθημα θα προκύπτει 40% από το βαθμό της 1ης εργασίας, 60% από τον βαθμό της 2ης εργασίας.
  - Όλοι οι φοιτητές υποχρεούνται να παραδώσουν 2 αναφορές υλοποίησης, οι οποίες αξιολογούνται με το κριτήριο «επιτυχής / μη επιτυχής» για να κατοχυρώσουν το εργαστήριο και να έχουν δικαίωμα να πάρουν τις εργασίες.
- Επιτυχία
  - Βαθμός > 50/100

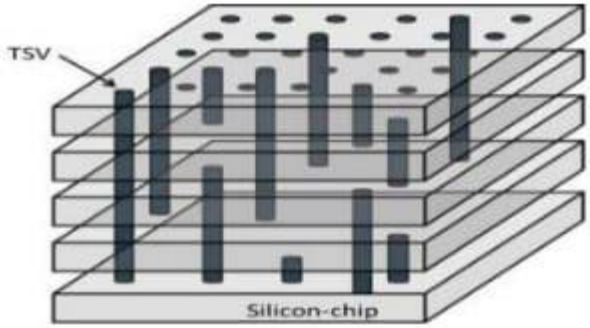
Technology constantly on the move



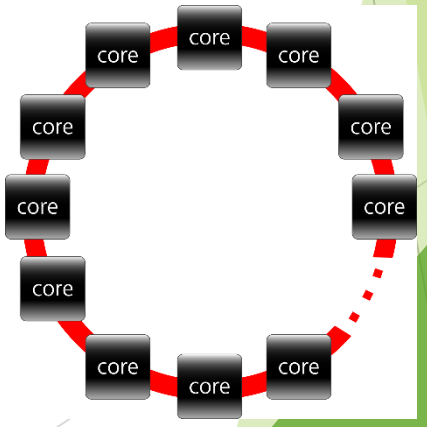
Sandwiches of silicon



FinFET

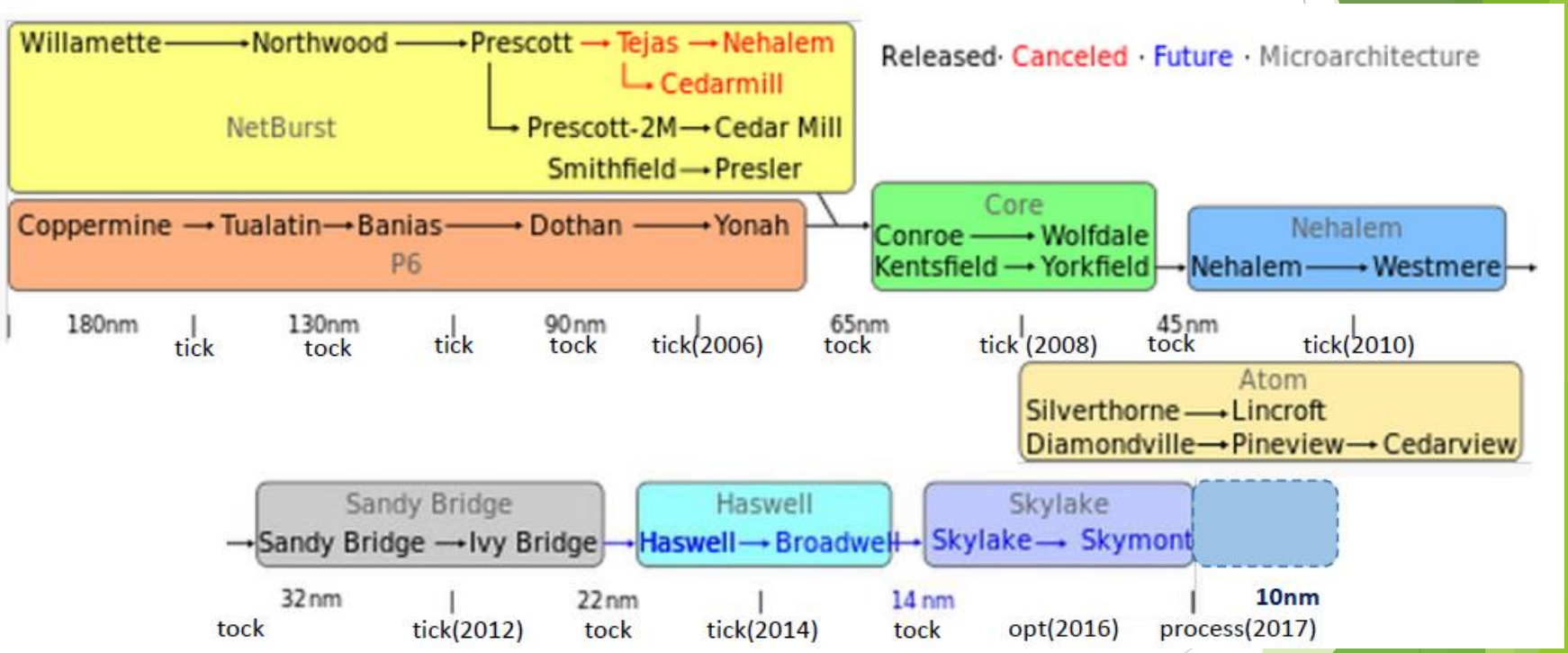


Through silicon Vias



On chip optical connections

Technology vs microarchitecture: Intel CPU evolution



source: CS425 - Vassilis Papaefstathiou

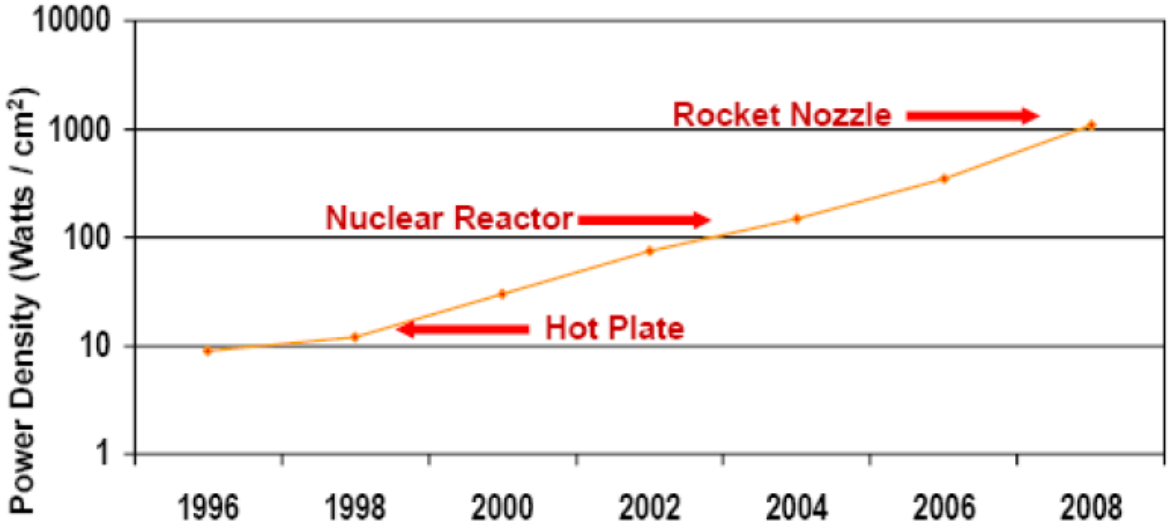
CPU die comparison

CPU	Technology Process	Cores	GPU	Transistor Count	Die Size mm^2
AMD Epyc Rome (64-bit, SIMD, caches)	7 & 12nm	32?		39.5B	1088(!)
Apple A13 (iphone11+)	7nm	4	GT2	8.5B	~99
Haswell GT3 4C	22nm	4	GT3	?	~264
Haswell GT2 4C	22nm	4	GT2	1.4B	177
Haswell ULT GT3 2C	22nm	2	GT3	1.3B	181
Intel Ivy Bridge 4C	22nm	4	GT2	1.2B	160
Intel Sandy Bridge E 6C	32nm	6	N/A	2.27B	435
Intel Sandy Bridge 4C	32nm	4	GT2	995M	216

Πηγές: cpu-world.com, anandtech.com, Wikipedia, κλπ.

Limiting Force: Power Density

Moore's Law Extrapolation:
Power Density for Leading Edge Microprocessors

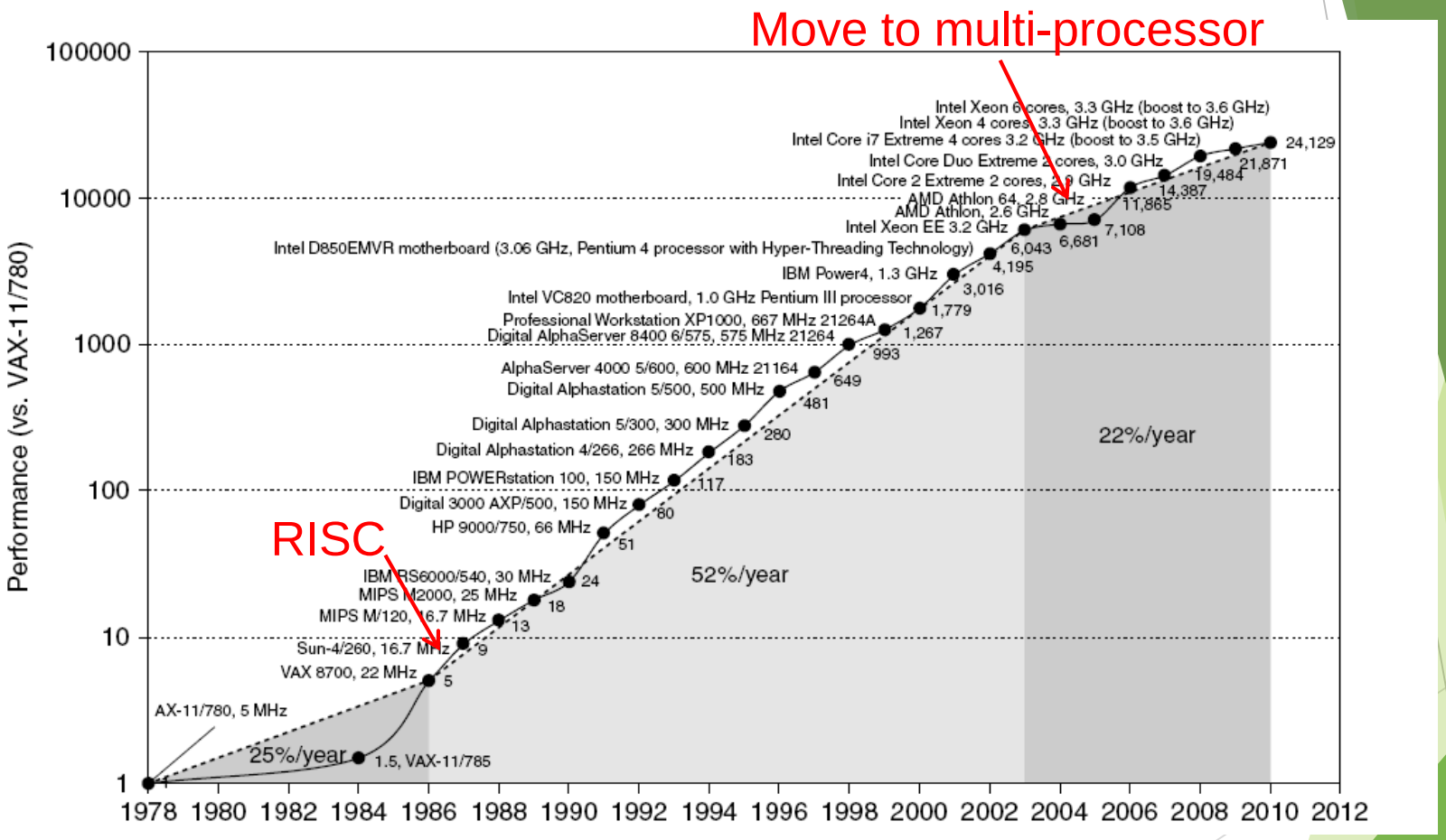


Microprocessor Size ≈ 2 cm²

Power Density Becomes Too High to Cool Chips Inexpensively

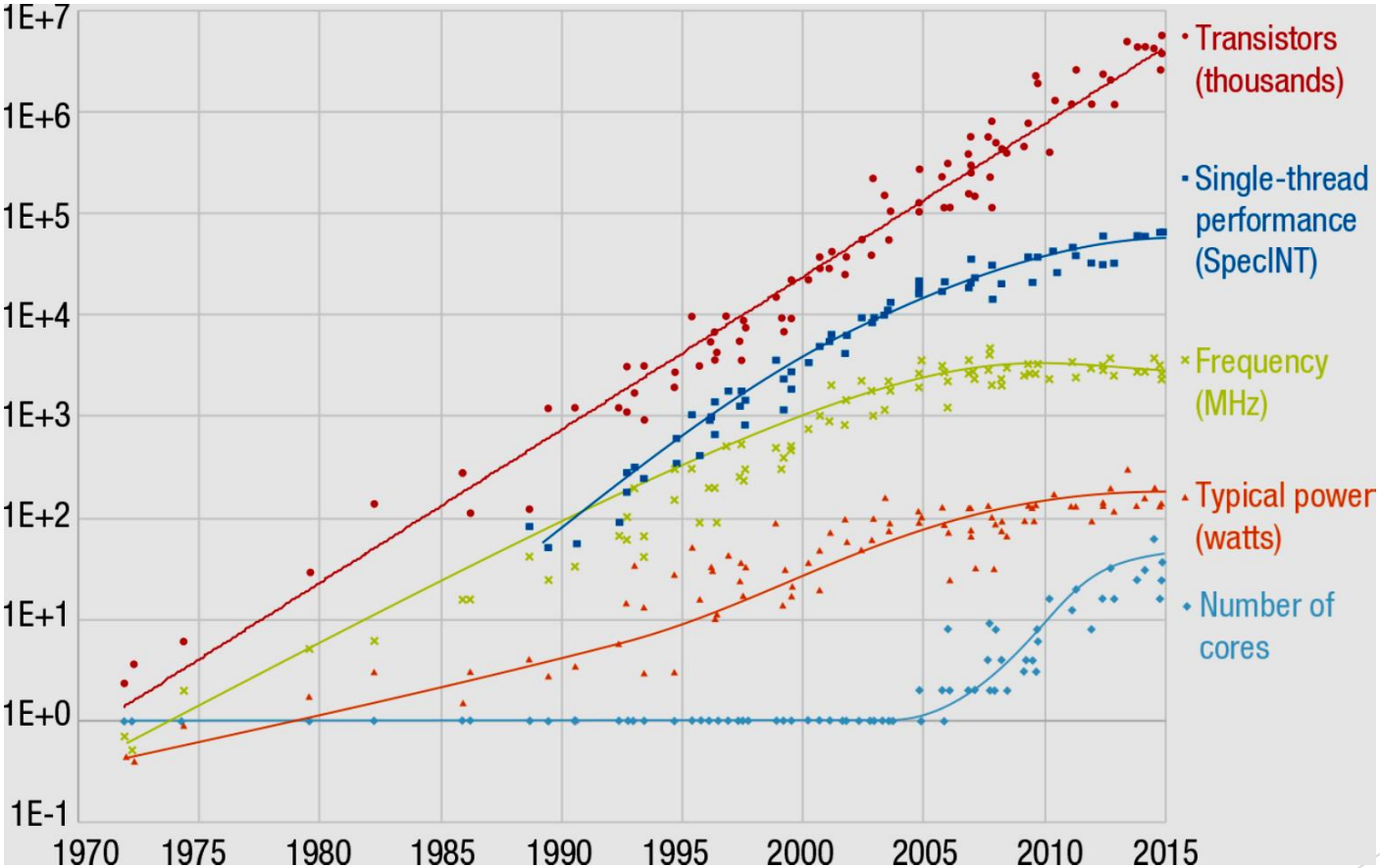
Source: Shekhar Borkar, Intel Corp

Crossroads: Uniprocessor Performance



Constrained by power, instruction level parallelism, memory latency

Microprocessor Trends



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2015 by K. Rupp

The End of the Uniprocessor Era

- **Power wall**: power expensive, transistors free
  - can put more on chip than can afford to turn on
- **ILP wall**: law of diminishing returns on more HW for ILP
- **Memory wall**: Memory slow, multiplies fast
  - 200 clock cycles to DRAM memory vs. 4 clocks for multiply
- **Power Wall + ILP Wall + Memory Wall = Brick Wall**

Uniprocessor performance now 2X every ..many.. years

The future: Many Core Chips

AND

New models for performance:

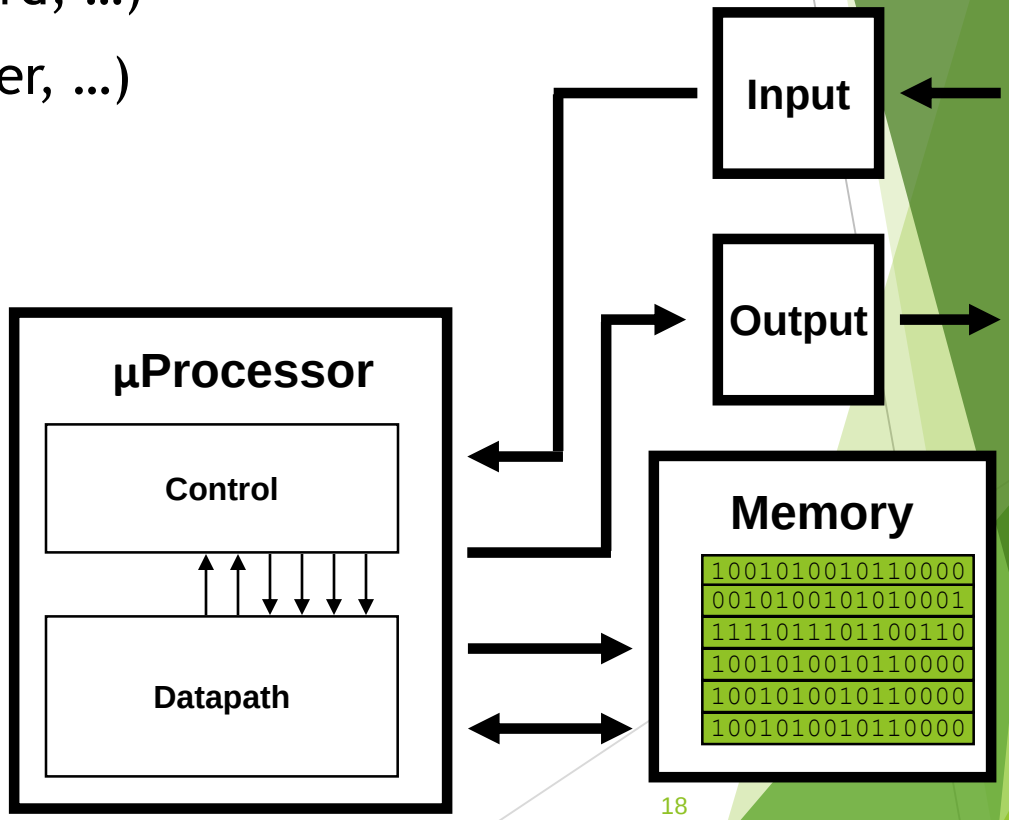
- Data-level parallelism (DLP)
- Thread-level parallelism (TLP)
- Request-level parallelism (RLP)

These require explicit restructuring of the application

But enough is enough … ☺

The Five Classic Components of a µComputer

- Input (mouse, keyboard, …)
- Output (display, printer, …)
- Memory
- Datapath
- Control



Our Primary Focus

- The processor (CPU)…
  - ✓ datapath
  - ✓ control
- …implemented using millions of transistors
- …impossible to understand by looking at individual transistors
- we need...

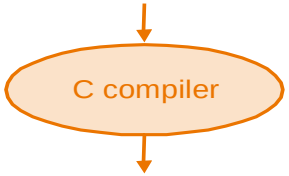
Abstraction

- Delving into the depths reveals more information, but…
- An abstraction omits "unneeded" detail, helps us cope with complexity

From the figure on the right, how does abstraction help the programmer and how does she avoid too much detail?

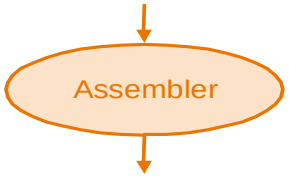
High-level language program (in C)

```
swap(int v[], int k)
{int temp;
   temp = v[k];
   v[k] = v[k+1];
   v[k+1] = temp;
}
```



Assembly language program (for MIPS)

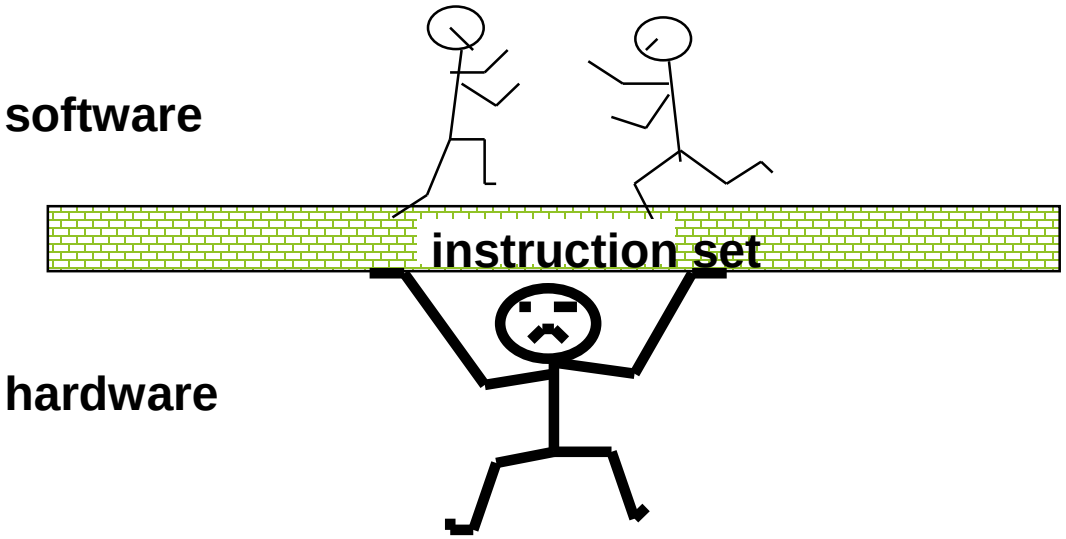
```
swap:
      muli $2, $5,4
      add  $2, $4,$2
      lw   $15, 0($2)
      lw   $16, 4($2)
      sw   $16, 0($2)
      sw   $15, 4($2)
      jr   $31
```



Binary machine language program (for MIPS)

```
00000000101000010000000000011000
00000000100011100001100000100001
10001100011000100000000000000000
10001100111100100000000000000100
10101100111100100000000000000000
10101100011000100000000000000100
00000011111000000000000000001000
```

The Instruction Set: a Critical Interface



Properties of a good ISA

- Lasts through many generations (portability)
- Used in many different ways (generality)
- Provides convenient functionality to higher levels
- Permits an efficient implementation at lower levels

Instruction Set Architecture

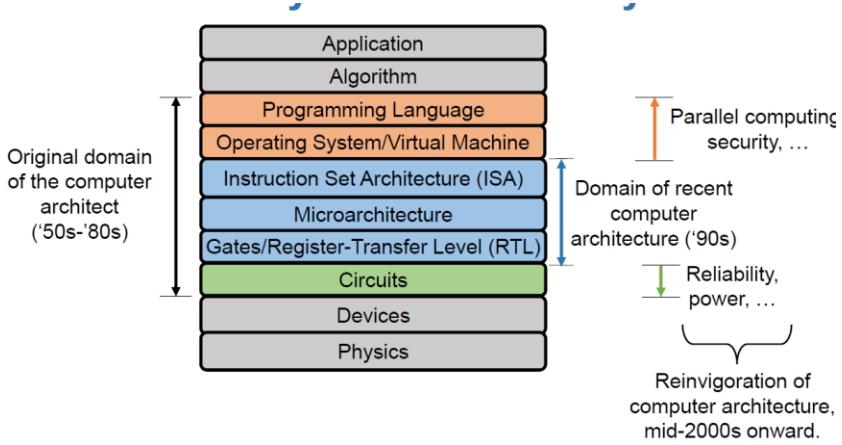
- A very important abstraction:
  - interface between hardware and low-level software
  - standardizes instructions, machine language bit patterns, etc.
  - advantage: allows different implementations of the same architecture
  - disadvantage: sometimes prevents adding new innovations

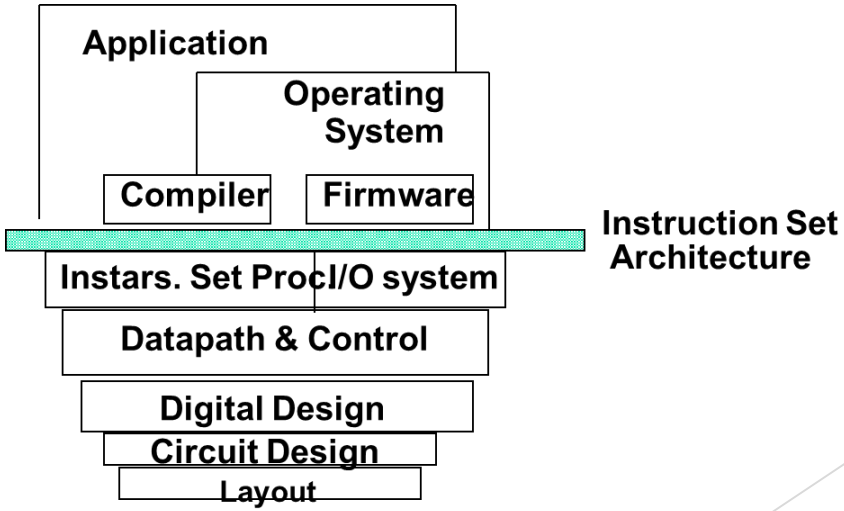
What is μComputer Architecture? Easy Answer

Computer Architecture =
Instruction Set Architecture +
Machine Organization

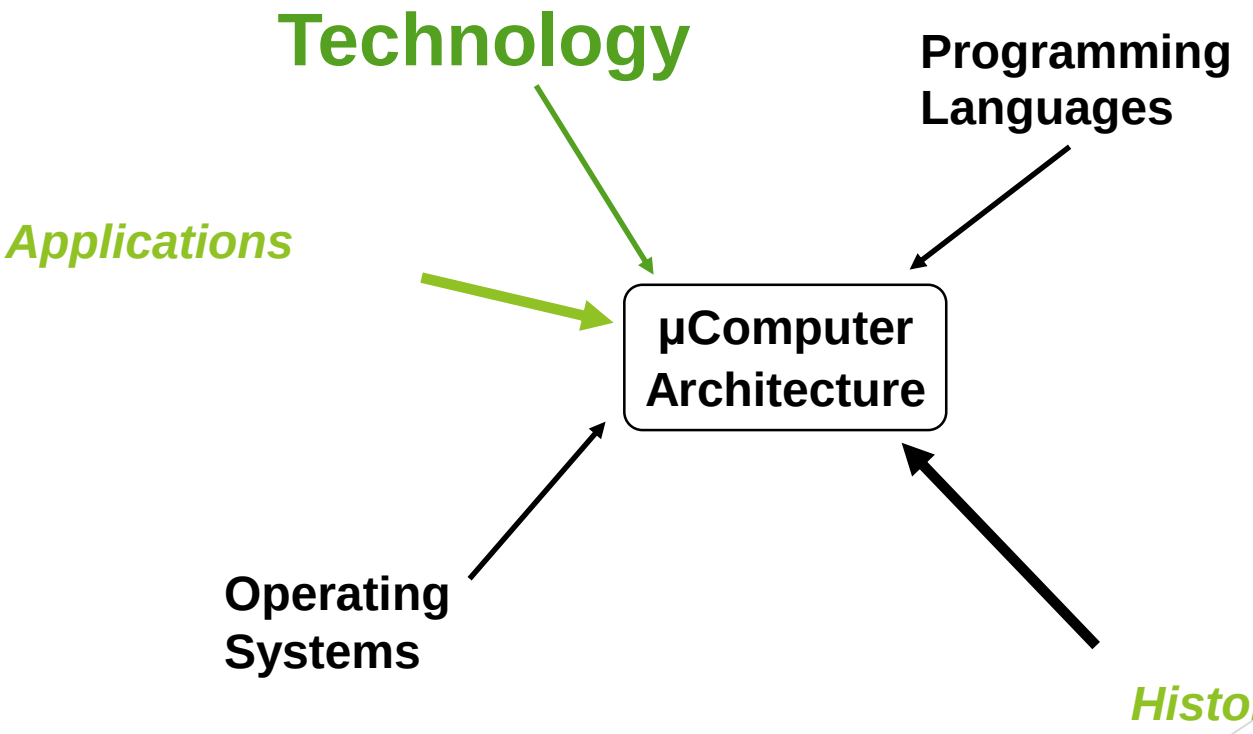
What is µComputer Architecture? Better (More Detailed) Answer

- Coordination of many levels of abstraction
- Under a rapidly changing set of forces
- Design, Measurement, and Evaluation





Forces on µComputer Architecture



µComputer Architecture is an Integrated Approach

- What really matters is the functioning of the complete system
  - hardware, runtime system, compiler, operating system, and application
- Computer architecture is not just about transistors, individual instructions, or particular implementations
  - E.g., Original RISC projects replaced complex instructions with a compiler + simple instructions
- It is very important to think across all hardware/software boundaries
  - New technology -> New Capabilities -> New Architectures -> New Tradeoffs
  - Delicate balance between backward compatibility and efficiency

Architecture = ISA (+prog . lang.) + Organization + Hardware

Τέλος Ενότητας