28-ppf-notes

Architecture Past, Present, and Future

Agenda

- 0. Re-Orienting
- 1. Love your motherboard!
- 2. The RISC Revolution
- 3. Some Real-World CPUs
- 4. HW Support for Virtualization
- 5. Multi-Core
- 6. Tour of Topics
- 7. What's Next

Reading: RISC sidebar on pp342-344 and on 448-449; multicore sidebar on p586

On Friday, we talked a bit about the NX (no execute) protection bit. See p 266 in the textbook for more discussion.

---

the slides (https://ssl.cs.dartmouth.edu/~sws/cs51-s15/28-ppf/slides.pdf)

---

0. Re-Orienting

---

1. Love Your Motherboard

(Leftover from Friday)

Ch 6: some layers between the CPU and main memory

(and this doesn't even show the cache!)

Chapter 1 shows even more stuff below the I/O bridge.

With this more complex architecture, we can start doing other sorts of data transfers.

- Example: DMA

Sneaky trend:

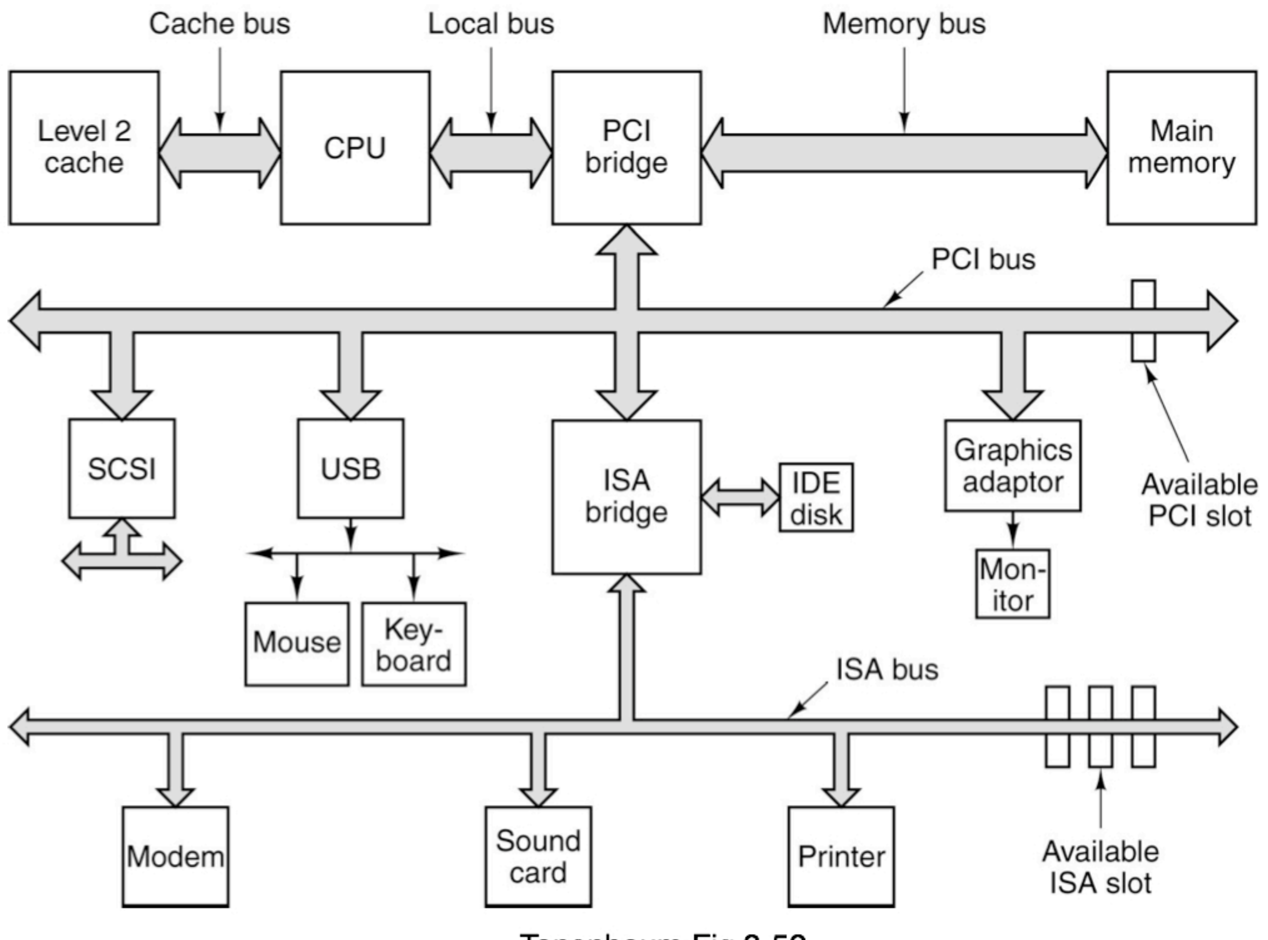
Let's stop thinking of the bus as a way to reach out and get an instruction or operand during instruction execution... and instead just think about it as a way of sending data around

Implications:

- System behavior can stop being (easily) deterministic
- It can be harder to infer what one part of the system (e.g., the CPU) is doing simply by looking at current bus activity
- Tuning or even simply measuring performance can be tricky.

A few example architectures

Older Pentium-based architecture:



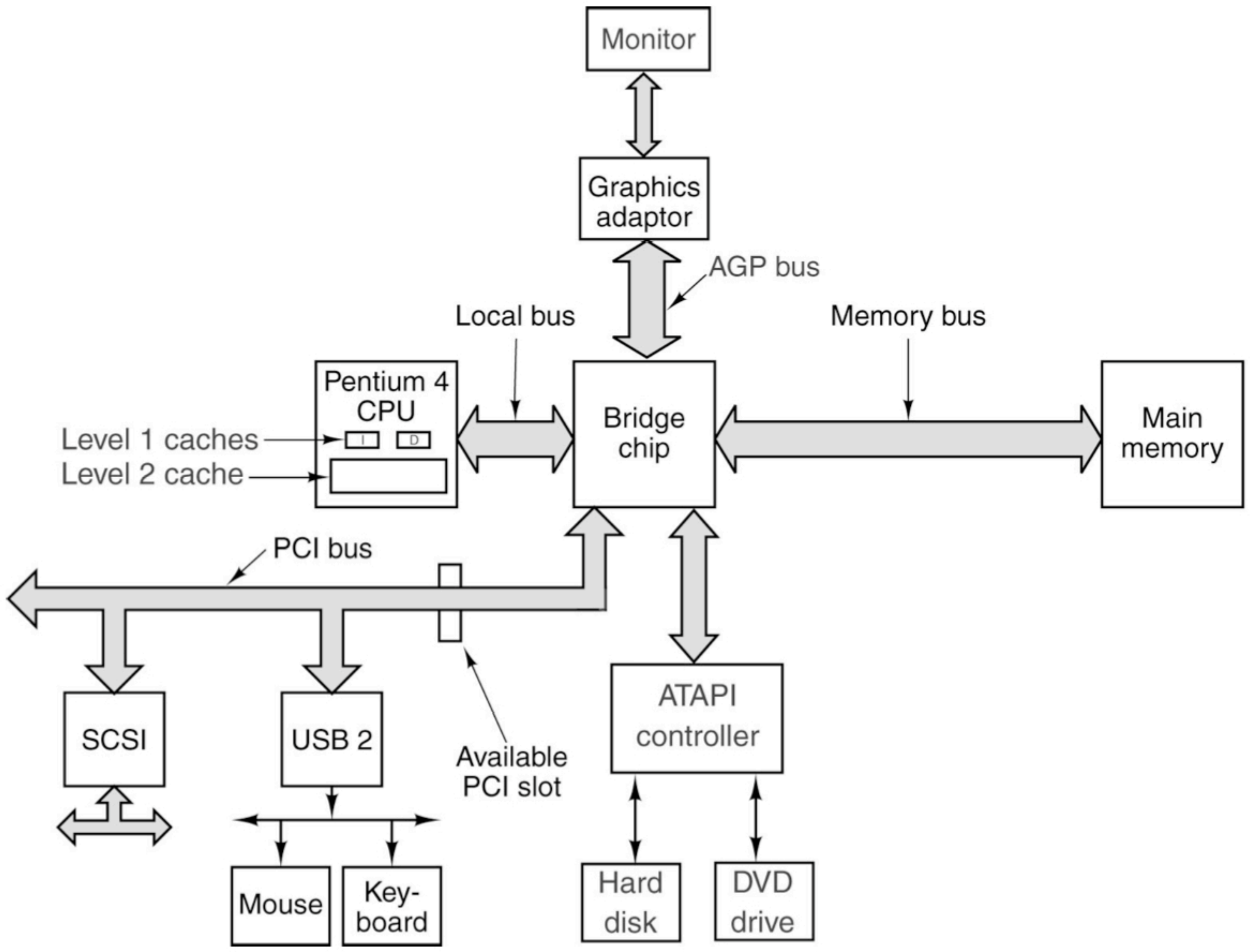
Tanenbaum Fig 3-52

- What new things do you see here?

some specialized terms, not always in textbooks (but in this one!):

- northbridge
- southbridge
- frontside bus

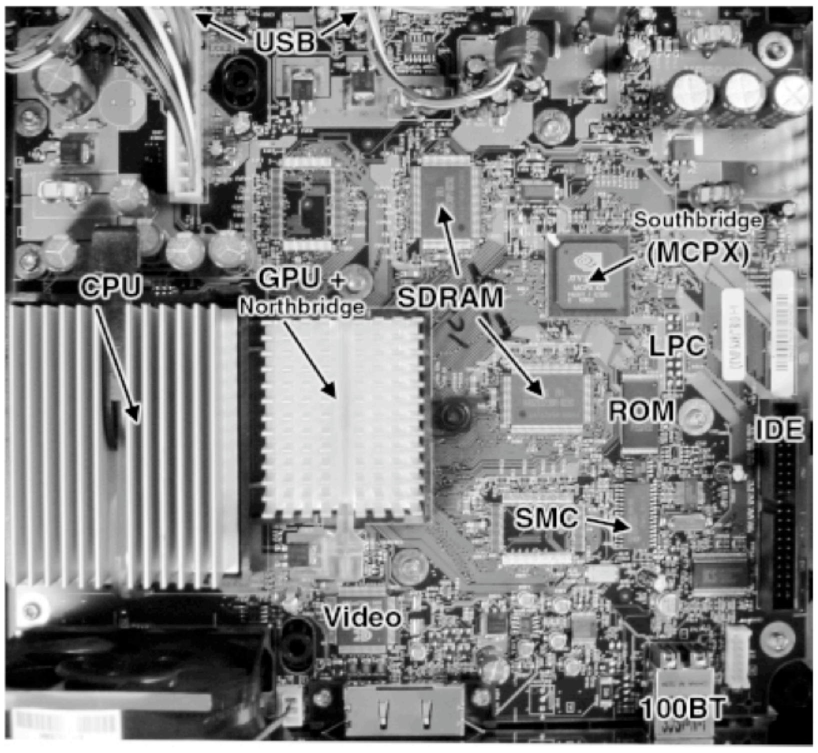
Newer Pentium-based architecture:



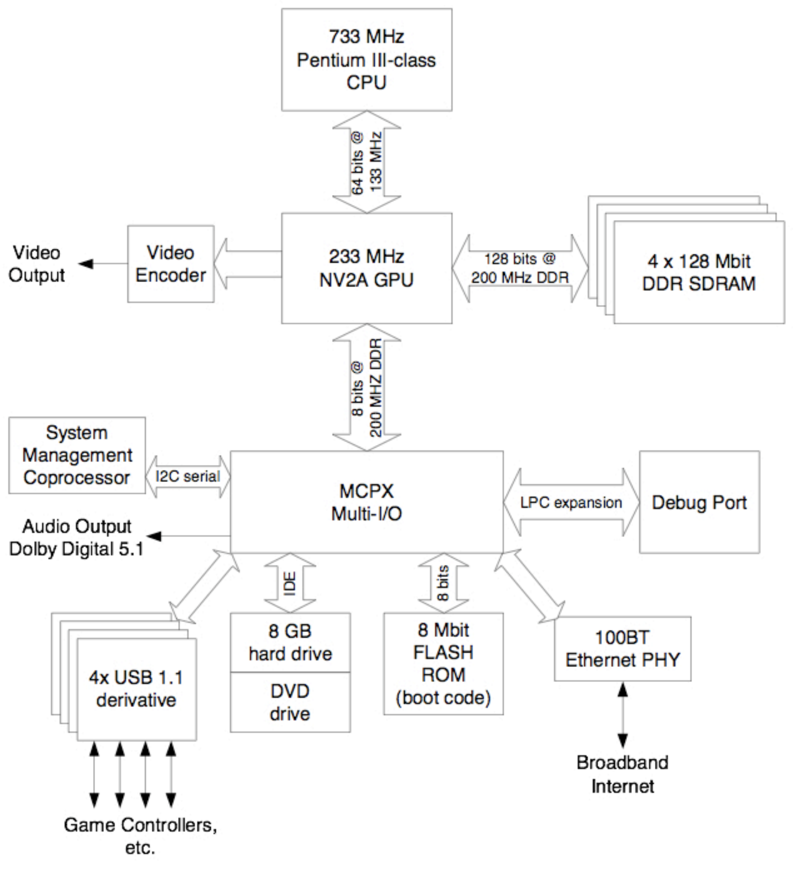
Tanenbaum Fig 3-53

- What new things do you see here?

The X-Box (From [***Hacking the X-Box*** (http://hackingthexbox.com)](http://hackingthexbox.com)):



from *Hacking the X-Box*



from *Hacking the X-Box*

Since Matt asked...

http://electronics.howstuffworks.com/xbox-three-sixty.htm [(http://electronics.howstuffworks.com/xbox-three-sixty.htm)](http://electronics.howstuffworks.com/xbox-three-sixty.htm) [(http://electronics.howstuffworks.com/xbox-three-sixty.htm)](http://electronics.howstuffworks.com/xbox-three-sixty.htm)

What this means when you are playing video games is that the Xbox 360 can dedicate one core entirely to producing sound, while another may run the game's collision and physics engine. The system may allocate an entire processor just to rendering hi-def graphics

...

The original Xbox took a lot of criticism for how large it was. The cooling system that kept the Xbox's rather hefty processor cool was the single greatest factor that contributed to the console's robust size. Microsoft changed all this for the Xbox 360.

...

In order to fit all that hardware into the stylish and slim form factor of the Xbox 360, Microsoft devised a cooling system that combines a small, vacuum-sealed, liquid-cooled system with fans to maintain a comfortable temperature inside the 360. The system regulates the temperature of the cores and adjusts the flow of liquid and fan speed accordingly. Additionally, the cooling system monitors the core's workload: If one or more cores are not needed for the job at hand (for instance, if you are using the Xbox 360 to watch a DVD), then the unused cores are automatically turned off.

---

1 1/2: Real-World Buses

The general idea: a "common" channel that allows numerous devices to talk to each other.

We've seen examples already:

- address bus in simple computer model
- data bus in simple computer model

Getting more formal....

someone is the "master"

other parties are "slaves"

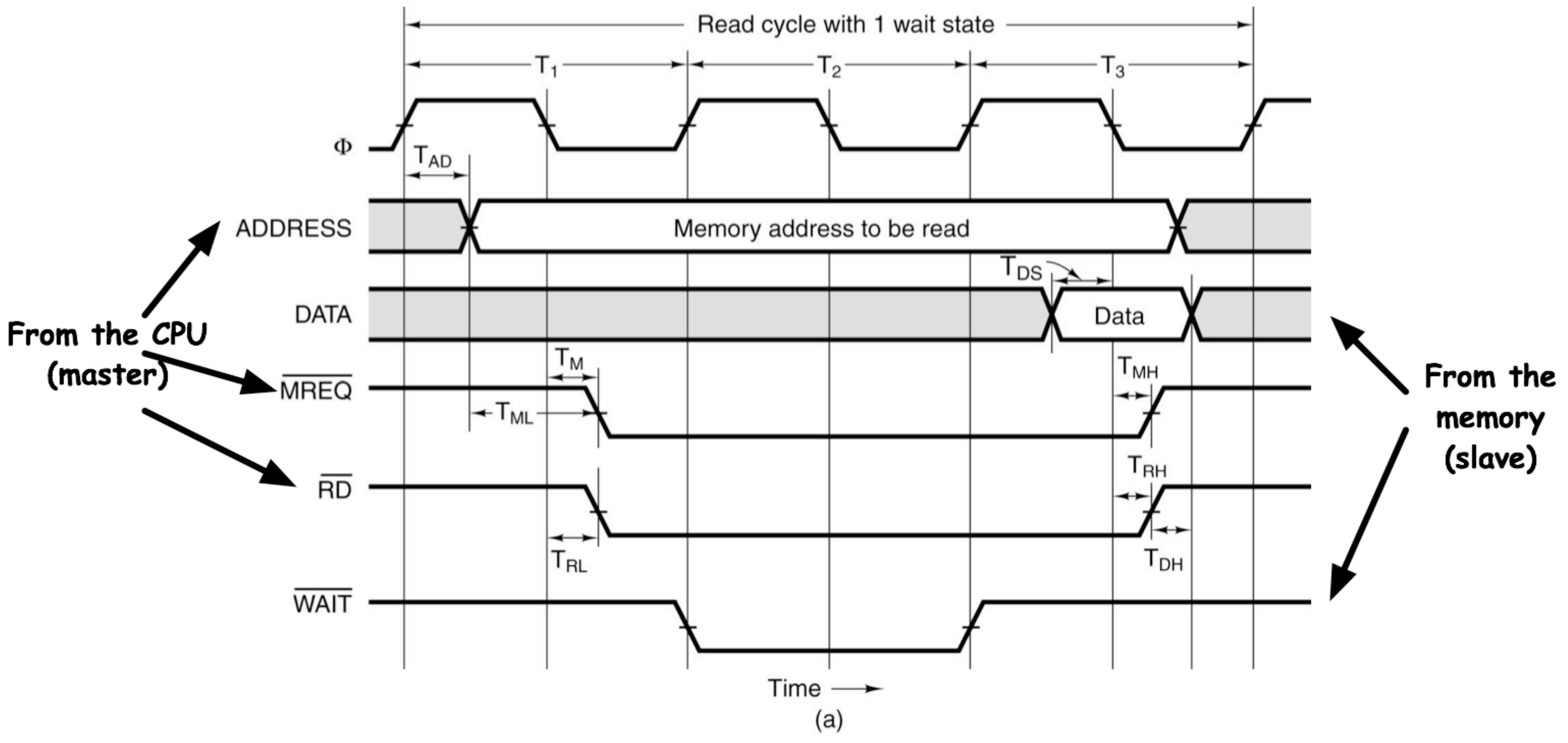
we need to worry about "width", although sometimes we can "multiplex" the bus

- what IS a "multiplexed" bus, anyway?

how to organize communication on the bus?

With what model have we operated so far?

Tanenbaum talks first about the "synchronous" model:



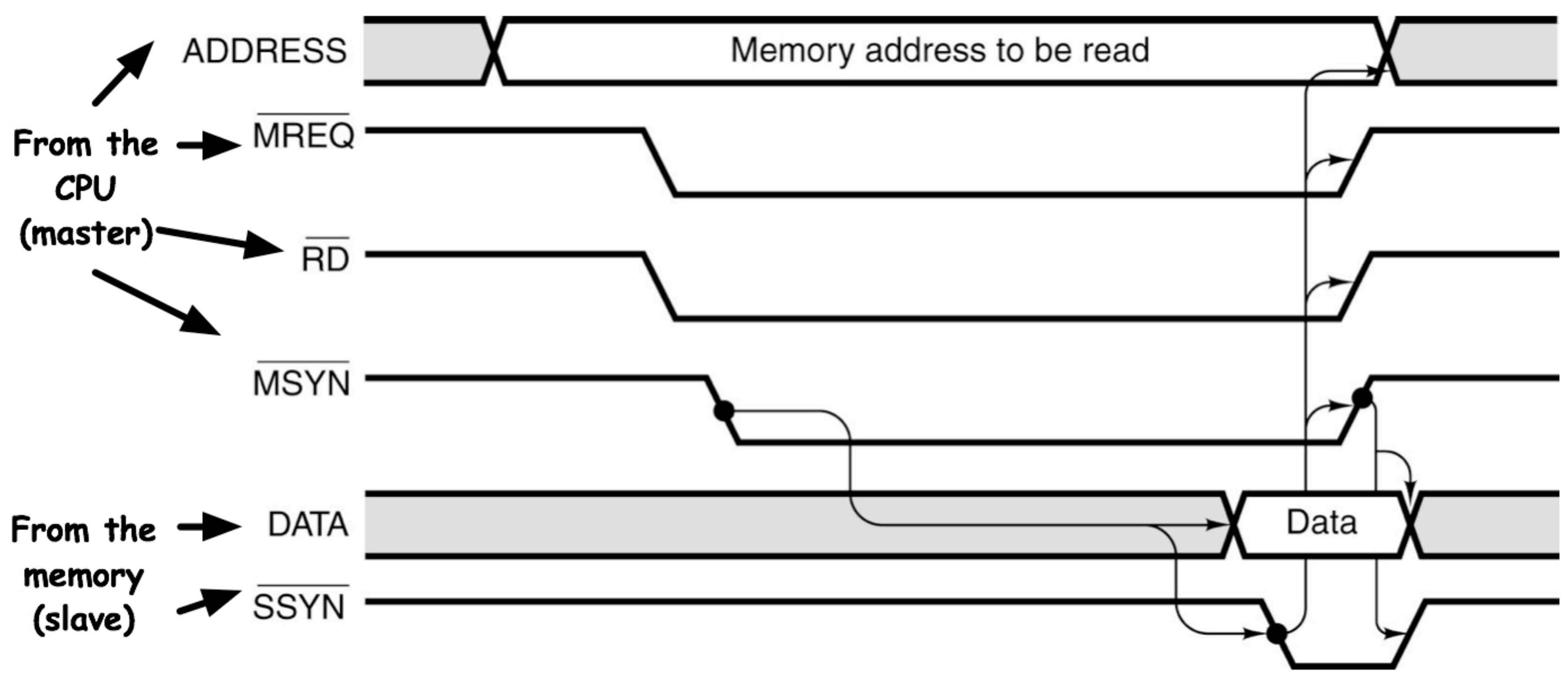
Tanenbaum Fig 3-38 a

There's a lot to take a way from this sort of diagram, such as:

- what's going on

- why none of the vertical lines are truly vertical
- why the creator of the diagram felt compelled to identify all sorts of special time intervals

Then the "asynchronous" model:



Tanenbaum Fig 3-39

if we have multiple masters, then we need to worry about "bus arbitration"

A few example buses

Warning: this is a land of alphabet soup, and half of the buses you learn about will be obsolete in 10 years. (Who remembers PCMCIA?)

ISA: "Industry standard architecture". Arose from the early IBM PC.

PCI: "Peripheral Component Interconnect"

USB: "Universal Serial Bus"

- SERIAL? Does that mean things can only send one bit to each other? What good is that?

2. The RISC Revolution

Interpretation vs. "direct hardware execution"?

Early history: complex instructions, with much interpretation going on

- quantitative argument: fast ROM
- implication: updatable ROM

"Closing the semantic gap"

- between ISA and high-level languages
- (hence more complex ISAs, and dependence on interpreted rather than direct hw execution)

The RISC Revolution

Patterson at Berkeley; Hennessy at Stanford

keep it simple but fast

- quantitative argument: RAM speed "caught up" with ROM. "Interpretation penalty"

Insight: the importance of ISSUING instructions quickly

Life lesson: think about going against the stream

What won in the marketplace

Intel "CISC".

- economics of "code museum"
- CISC/RISC hybrid since 486

Life lesson: victory through assimilation

RISC "Design Principles" from Tanenbaum 2.1.4

- "All instructions are directly executdd by hw"
- "Maximize rate instructions are issued"
- "Easy to decode"
- "Only loads and stores should reference memory"
- "Plenty of Registers"

---

3. Some Real-World CPUs

Pentium 4.

backwards compatible with older Intel 32-bit ISAs, but has some 64-bit hw

36-bit address bus (bytes)

478 pins, 63 watts of power

register width. ("Code museum")

bigger PSR

support for BCD

http://en.wikipedia.org/wiki/X86 (http://en.wikipedia.org/wiki/X86)

The NetBurst arch

(Do you know what these elements are, why they're there, and what the design issues might be?)

Trace Cache stores micro-ops.

"precise interrupts"

"in-order retirement"

UltraSPARC III

1368-pin chip.

64-bit words, but data bus is double-wide

43-bit address space (but probably bytes, not words)

register windows

RISC

"core instructions are already micro-ops"

"prefetch cache"

MMU, TLB

compare the pipelines

From P. Song, "UltraSparc-3 Aims at MP Servers," *Microprocessor Report,* Oct. 27, 1997, pp. 29-34. "UltraSparc-3 (US-3) avoids rename registers and out-of-order-issue queues,resulting in a simple and fast CPU core without many handcrafted transistors in the control logic"

8051

---

microcontroller

40-pin chip

16-bit address bus, 8-bit words

---

4. HW Support for Virtualization

Quick Overview: SW Structure of Real-World Systems

User-level code

OS-level code

Tasks of an OS

- handling traps
- handling interrupts
- handling exceptions
- managing the machine
- managing user-level processes

OS tasks... and the hw/sw interface

SW Structure and Architecture

The OS and memory management

The OS and privilege levels

The OS and traps/interrupts/exceptions

Intro to Virtualization

What it is

Type 1

Type 2

Why??

Hardware and Virtualization

The guest OS expects a certain level of hw/sw interface. How do we map its expectations to the real hw/sw interface?

Some trouble spots...

- priv level of OS (and behavior differences)
- memory management (who, how much)
- traps

Quick and dirty solutions

Longer-term, cleaner solution

- Intel: Vanderpol, VT **(paper) (https://ssl.cs.dartmouth.edu/~sws/cs51-s15/28-ppf/intel-vt.pf)**
- AMD: Pacifica

5. Multi-Core

The Official Reasons

How to squeeze more performance out of the chips?

If building a faster CPU is not feasible or simply too expensive, maybe we can get the same bang from a larger number of slower ones?

Unofficial Reasons

Moore's Law, again

- what it is
- why it matters

---

economics of chip production

"Throwing a `Hail Mary pass' and then running down the field and catching it"

Terms

But a warning: these are not used consistently out there in the world.

Synonyms or not?

- "parallel"
- "concurrent"

Software elements:

- process
- thread

Software design issues:

- how to specify multiple processes/threads
- how to implement them (particularly on a uniprocessor)
- how to coordinate them?
- how on earth to design sw to exploit "parallelism"
- ...and can we do this automagically somehow?

CPU

core

- and "softcore"

die

package

---

uniprocessor

hardware threads

- and "hyperthreading"

multiprocessor

multicomputer

grid computing

- or "cloud computing" ?

---

multicore

- "chip-level multiprocessing" (CMP)
- debate: how many dies/packages
- many-core, "network on a chip"
- kentsfield **Wikipedia article** (http://en.wikipedia.org/wiki/Kentsfield_(microprocessor))

The Next Big Thing?

(the **wikipedia article on multi-core** (http://en.wikipedia.org/wiki/Multi-core_(computing)) is very good!)

---

some more motivations for multicore:

- memory wall
- ILP wall
- power wall

Brooks' observation

---

6. Tour of Topics

Foundations

Electricity

Transistors

Gates

Binary

- powers of 2
- 2's complement
- shifting bits

Combinatorial logic

Latches

Flip-flops

Sequential logic

- the idea of an FSM

RAM

ROM

byte addressing

address decoding

Processor architecture

- data path
- registers
- fsm

---

outside-facing architecture

- address bus
- data bus
- control bus
- ...and generalizations

---

real-world motherboards

---

the ISA

- registers
- types of instructions
- state flags
- traps, interrupts, exceptions
- I/O

---

assembly language

I/O programming

stacks, stack frames

subroutines

data structures

---

high-level languages

- globals
- locals

---

bonus CPU features

---

dirty tricks

- ..and CISC vis RISC
- pre-fetching
- pipelining
- hazards, stalls

- superscalar!
- caching
- branch prediction

---

the future

- multiprocessing, multicore
- virtualization

---

the hardware-software interface!

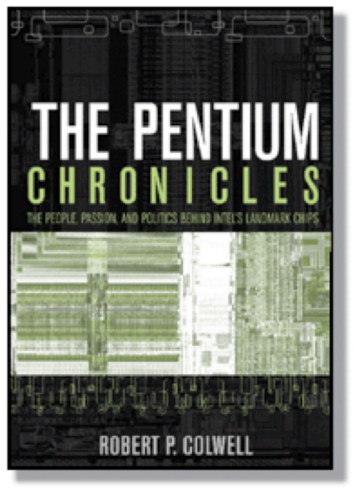
---

7. What's Next

The final..

Friday, 11:30, right here.

Top two scores will receive **The Pentium Chronicles**


THE PENTIUM
CHRONICLES
THE PEOPLE, PASSION, AND POLITICS BEHIND INTEL'S LANDMARK CHIPS
ROBERT P. COLWELL

- CS50: C, Unix tools
- CS58: How the OS manages the hardware to provide each userland process with the convenient illusion that it has the whole machine to itself
- CS57 or 59: How programming languages provide convenient illusions for the programmer
- CS56: Digitial Design (Thayer)
- Potential research opportunities with Profs Bratus, Kotz, Campbell, Balkcom, Jayanti, (or me)

I hope you learned something... and had the satisfaction of building

your own CPU, installing it inside a computer, and running C code on it!

FYI

Interesting books:

- John Markoff. *What the Dormouse Said: How the 60s Counterculture Shaped the Personal Computer.* Viking. 2005.
- Robert P. Colwell. *The Pentium Chronicles: The People, Passion, and Politics Behind Intel's Landmark Chips.* Wiley/IEEE. 2005.
