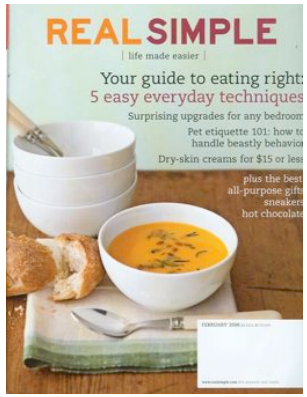
Lecture 01
Introduction to CSE 30321

Why I will not be here for a few days

- At the very beginning of this semester (i.e. the first 2-3 weeks or so), I will not be found in office as frequently as I will be for the rest of the semester – as my wife and I are expecting a baby girl on August 29th.
- However, I do intend to keep office hours as discussed in the syllabus a week or so after she arrives.
- I can always communicate via email too at any time – but it may take me a few days to respond around August 29th ☺.
- In my absence, Peter Kogge, Sharon Hu, or Aaron Dingler will teach class
  - All have taught this class before so you will be in good hands

A motivating example

All of the following are magazines that are regularly delivered to the Niemier household.



You can learn about good routes to run if you're visiting Chicago...

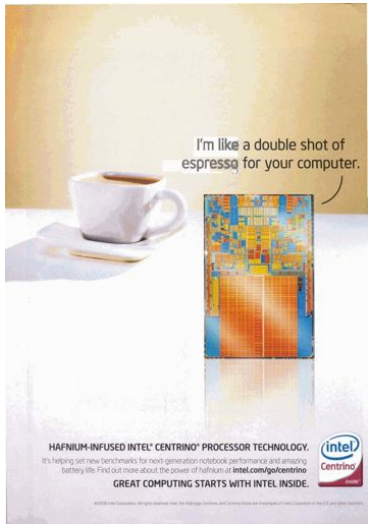
You can learn about the best ice cream...

Actually, I learned this summer that this place in Ephraim, WI has the best ice cream…

You can read about Pat Robertson and Al Sharpton advocating ways to reduce the effects of global climate change!

And if you're in the market for a wide-body jet, Boeing has just the thing…

And if you're looking to buy a computer, Intel suggests a dual core Centrino processor...



As you might guess, this brings us closer to CSE 30321

This is essentially a picture of one of Intel's ***multi-core*** computer architectures...

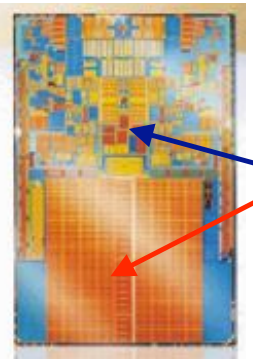
We'll learn about how different parts of a processor are organized and work together

Let's discuss some course goals (i.e. what you're going to learn)

Goal #1

- At the end of the semester, you should be able to...
  - **...describe the fundamental components required in a single core of a modern microprocessor**
    - (Also, explain how they interact with each other, with main memory, and with external storage media...)

Example



How do on-chip memory, processor logic, main memory, disk interact?

Goal #2 Motivation: which plane is best?

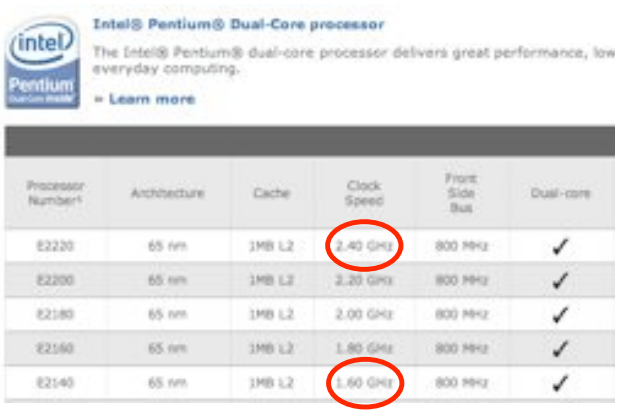
Which is best?

Plane	People	Range (miles)	Speed (mph)	Avg. Cost (millions)
737-800	162	3,060	530	63.5
747-8I	467	8000	633	257.5
777-300	368	5995	622	222
787-8	230	8000	630	153

Goal #2

- **At the end of the semester, you should be able to...**
  - **...compare and contrast different computer architectures to determine which one performs better...**

Example



If you want to do X, which processor is best?

Goal #3 Motivation…

- Which plane would you rather fly to South Bend?




But this is the one that you (uncomfortably) do fly…

Photo Credits:
http://www.airliners.net/aircraft-data/stats.main?id=196
http://images.businessweek.com/ss/07/06/0615_boeing787/image/787interior.jpg

Goal #3

- At the end of the semester, you should be able to...
  - …apply knowledge about a processor's datapath, different memory hierarchies, performance metrics, etc. to design a microprocessor that (a) meets a target set of performance goals and (b) is realistically implementable

Example

Climate Agency Awards $350 Million For Supercomputers

The National Oceanic and Atmospheric Administration will pay CSC and Cray to plan, build, and operate high-performance computers for climate prediction research.

By J. Nicholas Hoover
InformationWeek

<6 MHz MP3 Decode

Tensilica optimized the MP3 decoder for its HiFi DSPs. This MP3 decoder now runs at the lowest power and is the most efficient in the industry, requiring just 5.7 MHz when running at 128 Kbps, 44.1 KHz and dissipating 0.45 mW in TSMC's 65nm LP process (including memories). This makes Tensilica's Xtensa HiFi 2 Audio Engine ideal for adding MP3 playback to cellular phones, where current carrier requirements are for 100 hours of playback time on a battery charge, and increasing to 200 hours in the near future.

http://www.tensilica.com/pro[illegible]

(click for larger image and for full photo gallery)

In terms of a research and development computer, NOAA found it requires one the power of which will be ultimately measured in petaflops, which would make the future machine one of the world's most powerful supercomputers.

http://www.informationweek.com/news/government/enterprise-apps/showArticle.jhtml?articleID=225000152

Goal #4

- At the end of the semester, you should be able to...
  - ...understand how code written in a high-level language (e.g. C) is eventually executed on-chip...

Example

In C:

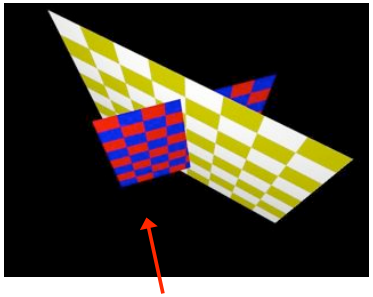
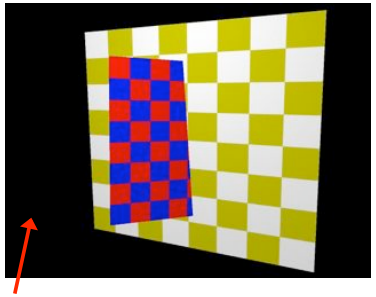
```
void insertionSort(int numbers[], int array_size)
{
  int i, j, index;

  for (i=1; i < array_size; i++)
  {
    index = numbers[i];
    j = i;
    while ((j > 0) && (numbers[j-1] > index))
    {
      numbers[j] = numbers[j-1];
      j = j - 1;
    }
    numbers[j] = index;
  }
}
```

In Java:

```
public static void insertionSort(int[] list, int length) {
   int firstOutOfOrder, location, temp;

   for(firstOutOfOrder = 1; firstOutOfOrder < length; firstOutOfOrder++) {
      if(list[firstOutOfOrder] < list[firstOutOfOrder - 1]) {
         temp = list[firstOutOfOrder];
         location = firstOutOfOrder;

         do {
            list[location] = list[location-1];
            location--;
         }
         while (location > 0 && list[location-1] > temp);

         list[location] = temp;
      }
   }
}
```

Both programs could be run on the same processor...
How does this happen?

Goal #5 Motivation (part 1)

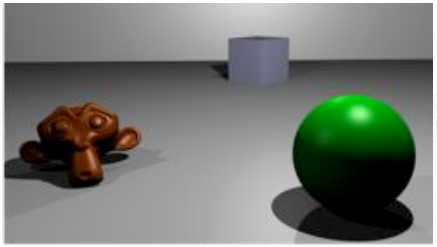
For this image, and this perspective, which pixels should be rendered?

Solution provided by *z-buffering algorithm*

- Depth of each object in 3D scene used to paint 2D image
- Algorithm steps through list of polygons
  - # of polygons tends to be >> (for more detailed scene)
  - # of pixels/polygon tends to be small

Image source: www.cs.unc.edu/~pmerrell/comp575/Z-Buffering.ppt

Goal #5 Motivation (part 2)



A simple three-dimensional scene

Z-buffer representation

Image source: http://en.wikipedia.org/wiki/Z-buffering

Often a dynamic data structure (e.g. linked list)

Given: A list of polygons {P1,P2,....Pn}
Output: A COLOR array, which display the intensity of the visible polygon surfaces.
Initialize:

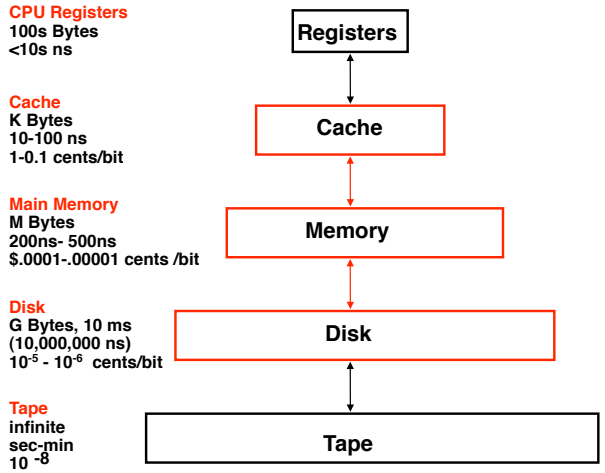
```
note : z-depth and z-buffer(x,y) is positive.....
z-buffer(x,y)=max depth; and
COLOR(x,y)=background color.
```

Begin:

```
for(each polygon P in the polygon list) do{
  for(each pixel(x,y) that intersects P) do{
    Calculate z-depth of P at (x,y)
    If (z-depth < z-buffer[x,y]) then{
      z-buffer(x,y)=z-depth;
      COLOR(x,y)=Intensity of P at(x,y);
    }
  }
}
display COLOR array.
```


Goal #5 Motivation (part 3)

Data structure may not fit in fast memory



Goal #5

- **At the end of the semester, you should be able to...**
  - **...use knowledge about a microprocessors underlying hardware (or "architecture") to write more efficient software…**

Example

Given: A list of polygons {P1,P2,....Pn}
Output: A COLOR array, which display the intensity of the visible polygon surfaces.
Initialize:

```
note : z-depth and z-buffer(x,y) is positive.....
z-buffer(x,y)=max depth; and
COLOR(x,y)=background color.
```

Begin:

```
for(each polygon P in the polygon list) do{
  for(each pixel(x,y) that intersects P) do{
    Calculate z-depth of P at (x,y)
    If (z-depth < z-buffer[x,y]) then{
      z-buffer(x,y)=z-depth;
      COLOR(x,y)=Intensity of P at(x,y);
    }
  }
}
display COLOR array.
```

Write this code…

…so it always fits in fast, on-chip memory

Let's digress...

- **Transition discussion before last goal**
  - **Some more interested in HW**
  - **Some more interested in SW**
- **But...changes in technology are having a profound impact on conventional/established computer architectures**
  - **We're presently at the very beginning of this storm...**
- **We'll need significant engagement from programmers to continue the processor performance scaling trends of the last 40 years...**
  - **... this will impact your career...**

Way we process information hasn't changed much since 1930s and 1940s

A little history... programs

- Stored program model has been around for a long time...

First Draft of a Report
on the EDVAC

by

John von Neumann

Contract No. W–670–ORD–4926

Between the

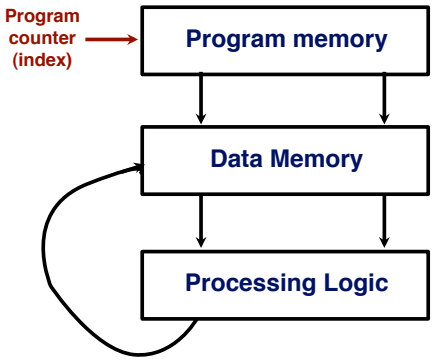
United States Army Ordnance Department

and the

University of Pennsylvania

Moore School of Electrical Engineering
University of Pennsylvania

June 30, 1945



Stored Program (continued)

A hypothetical translation:

```
for i=0; i<5; i++ {
      a = (a*b) + c;
}
```

```
MULT temp,a,b  # temp ← a*b
MULT r1,r2,r3  # r1 ← r2*r3
```

```
ADD a,temp,c   # a ← temp+c
ADD r2,r1,r4   # r2 ← r1+r4
```

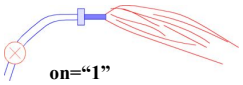
Can define codes for MULT and ADD
Assume MULT = 110011 & ADD = 001110

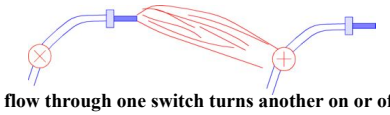
stored program becomes

PC	110011	000001	000010	000011
PC+1	001110	000010	000001	000100

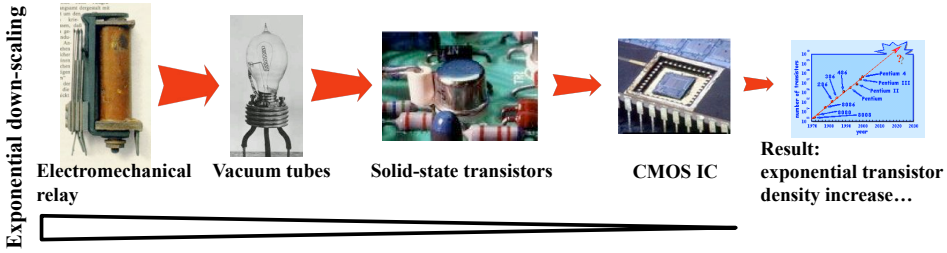
A little history... Zuse's paradigm

- Konrad Zuse (1938) Z3 machine
  - Use binary numbers to encode information
  - Represent binary digits as on/off state of a current switch



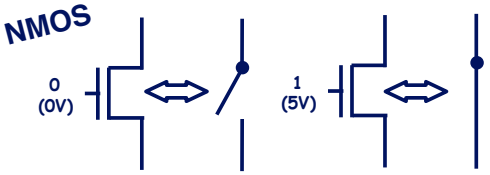


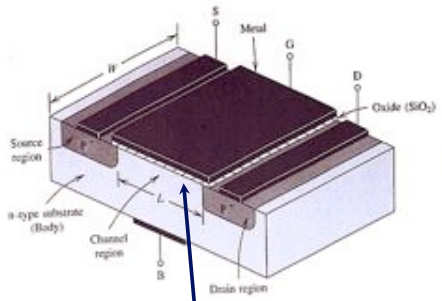
The flow through one switch turns another on or off.



Transistors used to manipulate/store 1s & 0s

Switch-level representation **Cross-sectional view**





(can act as a capacitor storing charge)

Using above diagrams as context, note that if we (i) apply a suitable voltage to the gate & (ii) then apply a suitable voltage between source and drain, current will flow.

Moore's Law

- "Cramming more components onto integrated circuits."

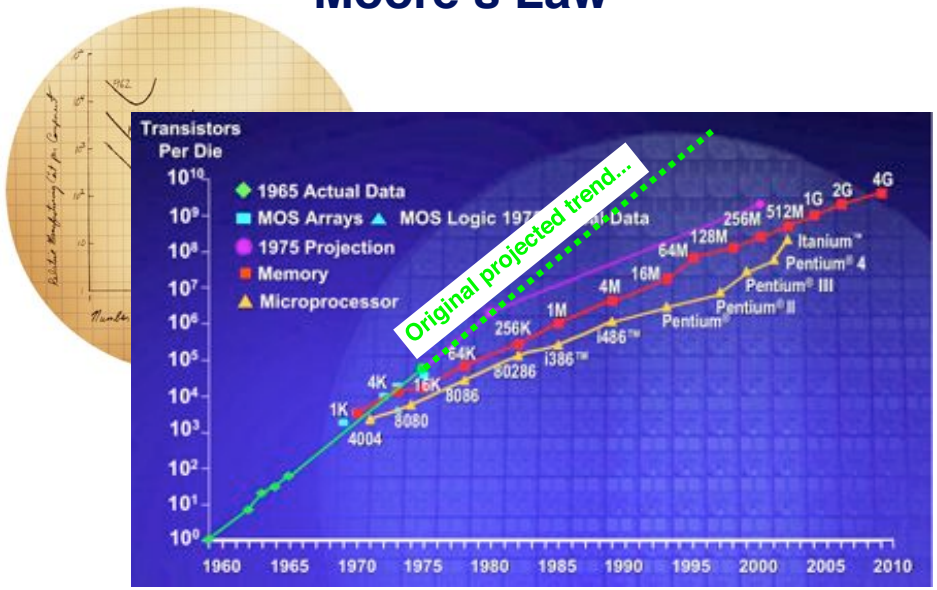
  - G.E. Moore, Electronics 1965

– **Observation: DRAM transistor density doubles annually**
  - Became known as "Moore's Law"
  - Actually, a bit off:
    – Density doubles every 18 months (now more like 24)
    – (in 1965 they only had 4 data points!)

– **Corollaries:**
  - Cost per transistor halves annually (18 months)
  - Power per transistor decreases with scaling
  - Speed increases with scaling
    – Of course, it depends on how small you try to make things
      » (I.e. no exponential lasts forever)

Remember these!

Moore's Law



Feature sizes...

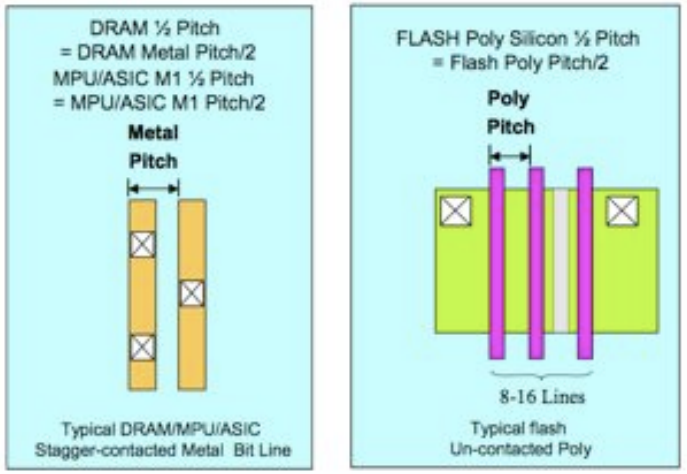


Figure 2 2005 Definition of Pitches

Moore's Law

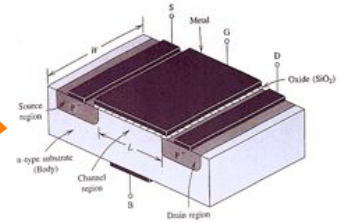
- Moore's Curve is a self-fulfilling prophecy
  - 2X every 2 years means ~3% per month
    - I.e. ((1 X 1.03) * 1.03)*1.03… 24 times = ~2
  - Can use 3% per month to judge performance features
  - If feature adds 9 months to schedule…it should add at least 30% to performance
    - ($1.03^9 = 1.30$ ⇨ 30%)

A bit on device performance...

- One way to think about switching time:
  - Charge is carried by electrons
  - Time for charge to cross channel = length/speed
    - $= L^2/(mV_{ds})$

Thus, to make a device faster, we want to either increase V_{ds} or decrease feature sizes (i.e. L)

- What about power (i.e. heat)?
  - Dynamic power is: $P_{dyn} = C_L V_{dd}^2 f_{0\text{-}1}$
    - $C_L = (e_{ox}WL)/d$
      - e_{ox} = dielectric, WL = parallel plate area, d = distance between gate and substrate



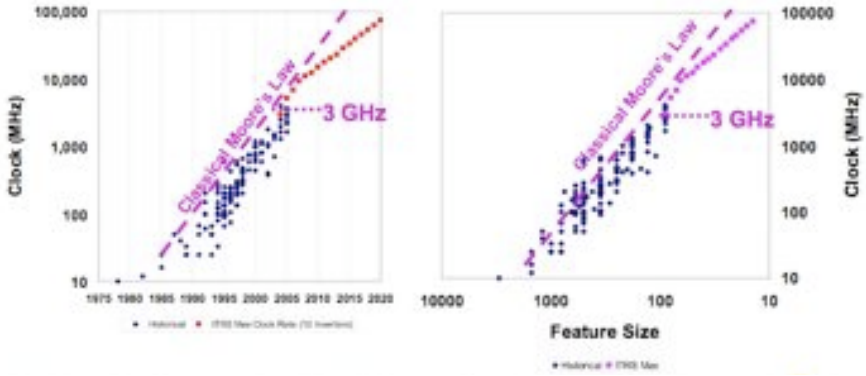
Summary of relationships

- (+) If V increases, speed (performance) increases
- (-) If V increases, power (heat) increases
- (+) If L decreases, speed (performance) increases
- (?) If L decreases, power (heat) does what?
  - P could improve because of lower C
  - P could increase because >> # of devices switch
  - P could increase because >> # of devices switch faster!

Need to carefully consider tradeoffs between speed and heat

A funny thing happened on the way to 45 nm

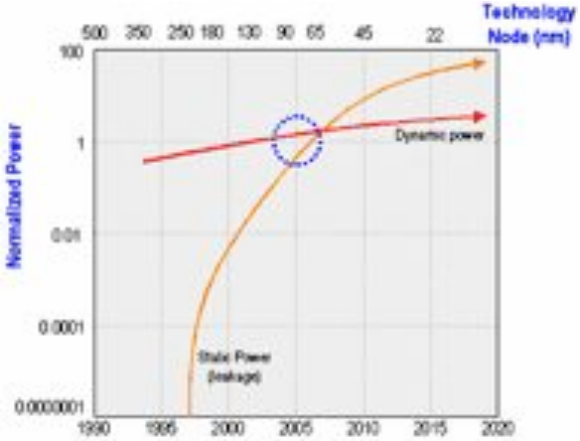
- Speed increases with scaling...



2005 projection was for 5.2 GHz - and we didn't make it in production. Further, we're still stuck at 3+ GHz in production.

A funny thing happened on the way to 45 nm

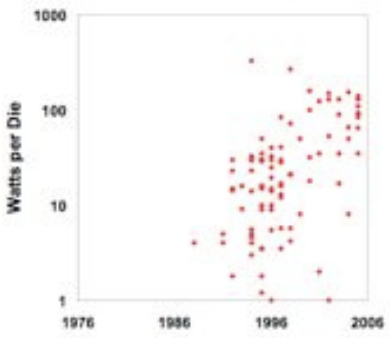
- Power decreases with scaling...



A funny thing happened on the way to 45 nm

- Speed increases with scaling...
- Power decreases with scaling...

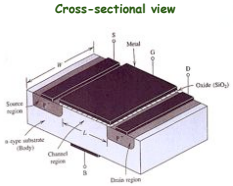
Why the clock flattening? POWER!!!!

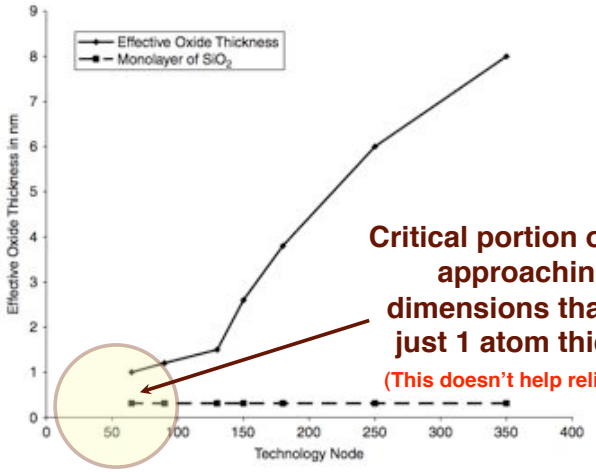


A funny thing happened on the way to 45 nm

- What about scaling...

One quick example:





Critical portion of gate approaching dimensions that are just 1 atom thick!!!

(This doesn't help reliability)

Materials innovations were – and still are – needed

Oh … transistors used for memory too

Saw earlier, that transistors used for *on chip* memory too...

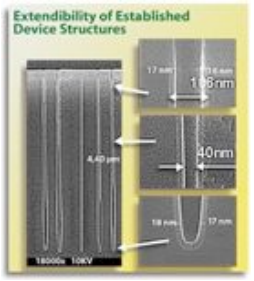
Problem: Program, data = bigger + power problem isn't going away...

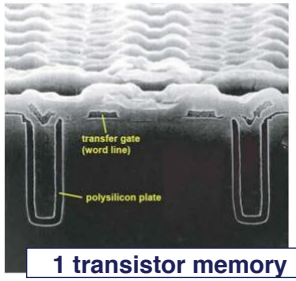
- Why? Put faster memory closer to processing logic...
  - SRAM (logic): density +25%, speed +20%
  - DRAM (memory): density +60%, speed +4%
  - Disk (magnetic): density +25%, speed +4%

Also, "memory wall"

DRAM vs. SRAM

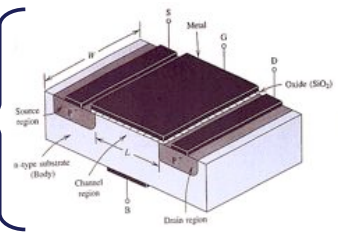
DRAM transistors are "deep trenches"

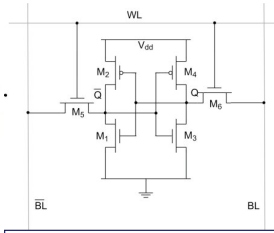




1 transistor memory

SRAM transistors made with logic process

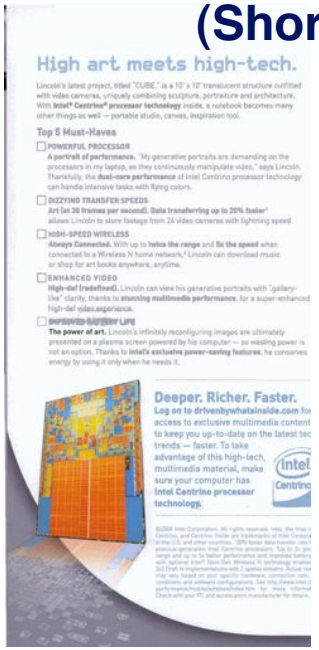




6 transistor memory

http://web.sfc.keio.ac.jp/~rdv/keio/sfc/teaching/architecture/architecture-2008/6t-SRAM-cell.png
http://www.micromagazine.com/archive/02/06/0206MI22b.jpg
http://www.ieee.org/portal/site/sscs/menuitem.f07ee9e3b2a01d06bb9305765bac26c8/index.jsp?&pName=sscs_level1_article&TheCat=2171&path=sscs/08Winter&file=Sunami.xml

(Short term?) Solution



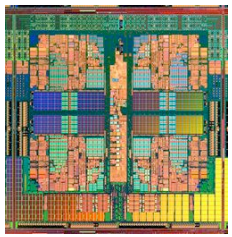
- Processor complexity is good enough
- Transistor sizes can still scale
- Slow processors down to manage power
- Get performance from...

Parallelism

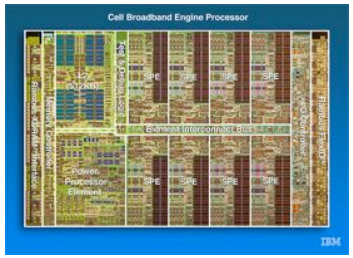
(i.e. 1 processor, 1 ns clock cycle vs. 2 processors, 2 ns clock cycle)

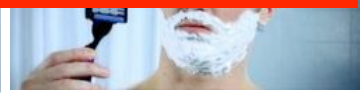
This idea has been extended...

Quad core chips...



7, 8, and 9 core chips...





Practical problems must be addressed!

Advances in parallel programming are necessary! (We'll get into later in the semester)

Multi-core only as good as algorithms that use it

- **Defined by Amdahl's Law:**

$$\text{Speedup} = \frac{1}{\left[1 - \text{Fraction}_{\text{parallelizable}}\right] + \frac{\text{Fraction}_{\text{parallelizable}}}{N}}$$

- **An example:**

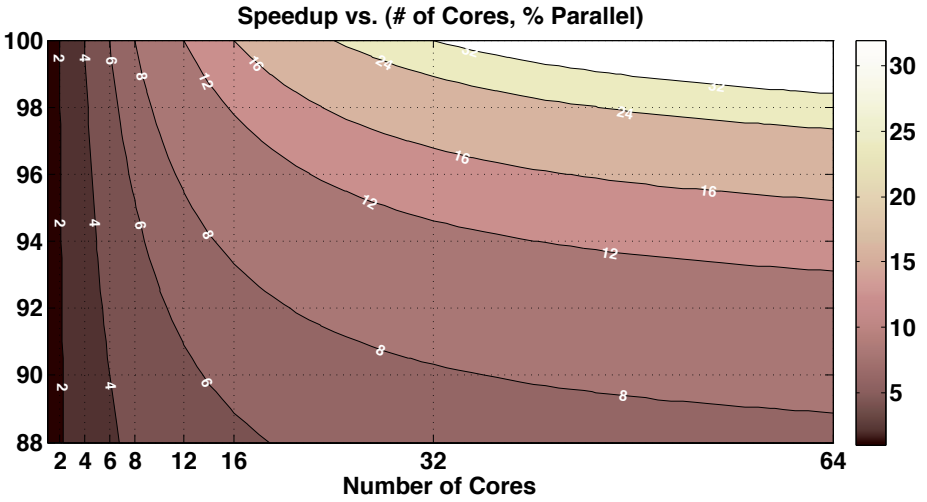
Software Defined Radio – A High Performance Embedded Challenge

Hyunseok Lee, Yuan Lin, Yoav Harel, Mark Woh, Scott Mahlke, Trevor Mudge[1], and Krisztian Flautner[2]

Table 3. Peak workload profile of the W-CDMA physical layer and its variation according to the operation state

	Active		Control Hold		Idle	
	(MOPS)	%	(MOPS)	%	(MOPS)	%
Searcher	**26538.0**	**42.1**	**26358.0**	**58.4**	**3317.3**	**37.7**
Interleaver	2.2	0.0	2.2	0.0	-	-
Deinterleaver	0.2	0.0	0.2	0.0	-	-
Conv. encoder	0.0	0.0	0.0	0.0	-	-
Viterbi Decoder	200.0	0.3	200.0	0.4	-	-
Turbo encoder	0.0	0.0	0.0	0.0	-	-
Turbo decoder	**17500.0**	**27.8**	**0.0**	**0.0**	-	-
Scrambler	245.3	0.4	245.3	0.5	-	-
Descrambler	**2621.4**	**4.2**	**2621.4**	**5.8**	**889.2**	**10.1**
Spreader	297.5	0.5	297.5	0.7	-	0.0
Despreader	**3642.5**	**5.8**	**3642.5**	**8.0**	**607.1**	**6.9**
LPF-Rx	**3993.6**	**6.3**	**3993.6**	**8.8**	**3993.6**	**45.3**
LPF-Tx	**7897.2**	**12.6**	**7897.2**	**17.4**	-	-
Power control	0.0	0.0	0.0	0.0	-	-
Total	62937.0	-	45272.9	-	8807.2	-

Multi-core only as good as algorithms that use it

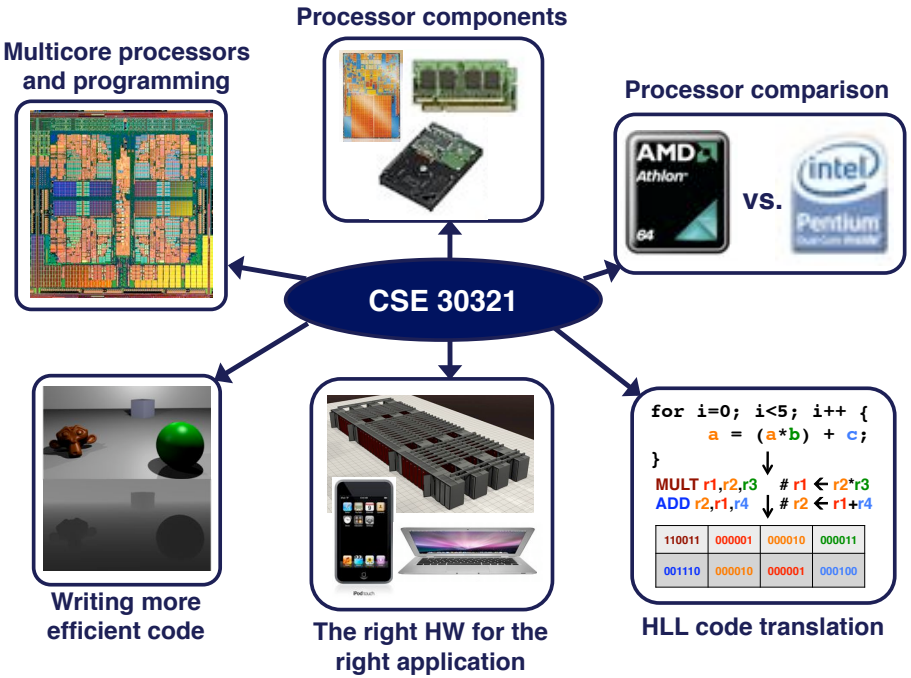


Goal #6

- At the end of the semester, you should be able to...
  - ...explain and articulate why modern microprocessors now have more than 1 core and how SW must adapt to accommodate the now prevalent multi-core approach to computing
- Why?
  - For 8, 16 core chips to be practical, we have to be able to *use them*
    - Students in this class should go on to play a role in making such chips useful...

Course goal summary

1. Describe the fundamental components required in a single core of a modern microprocessor as well as how they interact with each other, with main memory, and with external storage media.
2. Suggest, compare, and contrast potential architectural enhancements by applying appropriate performance metrics.
3. Apply fundamental knowledge about a processor's datapath, different memory hierarchies, performance metrics, etc. to design a microprocessor such that it (a) meets a target set of performance goals and (b) is realistically implementable.
4. Explain how code written in (different) high-level languages (like C, Java, C++, Fortran, etc.) can be executed on different microprocessors (i.e. Intel, AMD, etc.) to produce the result intended by the programmer.
5. Use knowledge about a microprocessor's underlying hardware (or "architecture") to write more efficient software.
6. Explain and articulate why modern microprocessors now have more than one core and how software must adapt to accommodate the now prevalent multi-core approach to computing.



Now, let's look at the syllabus