Electrical & Computer ENGINEERING

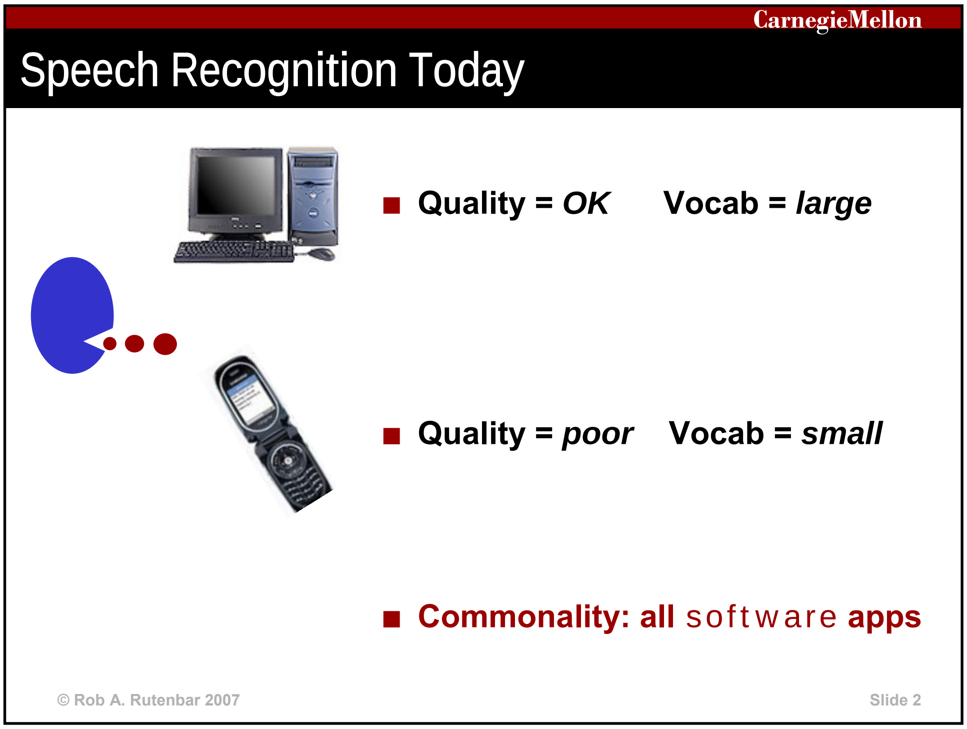
2007 STARC Forum

Towards Speech Recognition in Silicon: The Carnegie Mellon *In Silico Vox* Project

Rob A. Rutenbar
Professor, Electrical & Computer Engineering
rutenbar@ece.cmu.edu

© R.A. Rutenbar 2007

Carnegie Mellon

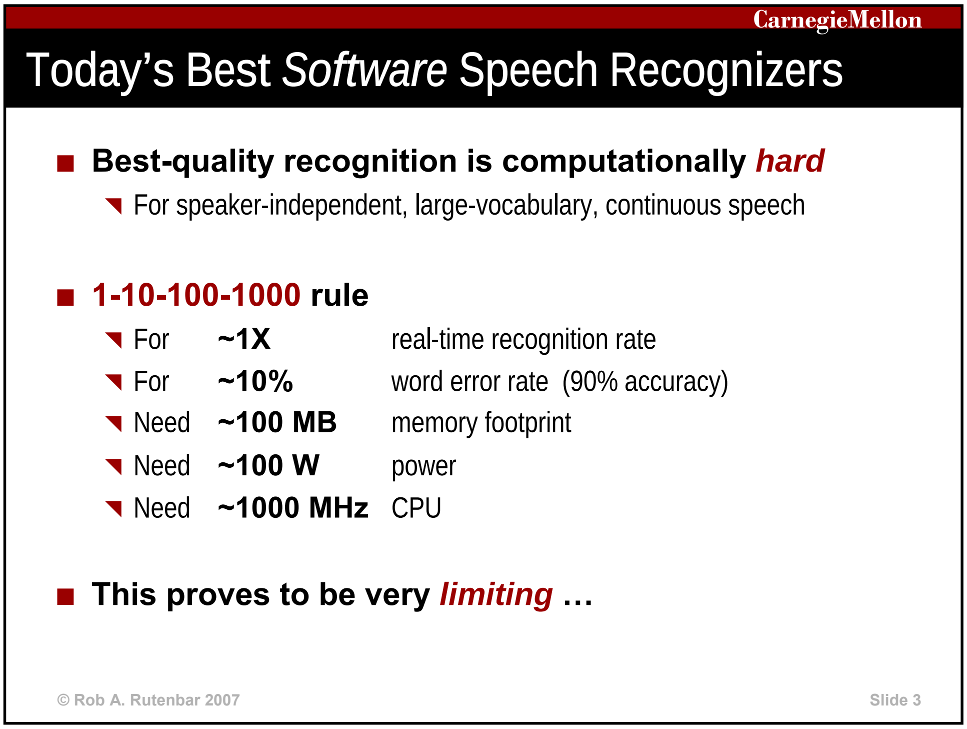


CarnegieMellon

Speech Recognition Today

- Quality = *OK*   Vocab = *large*
- Quality = *poor*   Vocab = *small*
- **Commonality: all** software **apps**

© Rob A. Rutenbar 2007   Slide 2



CarnegieMellon

Today's Best *Software* Speech Recognizers

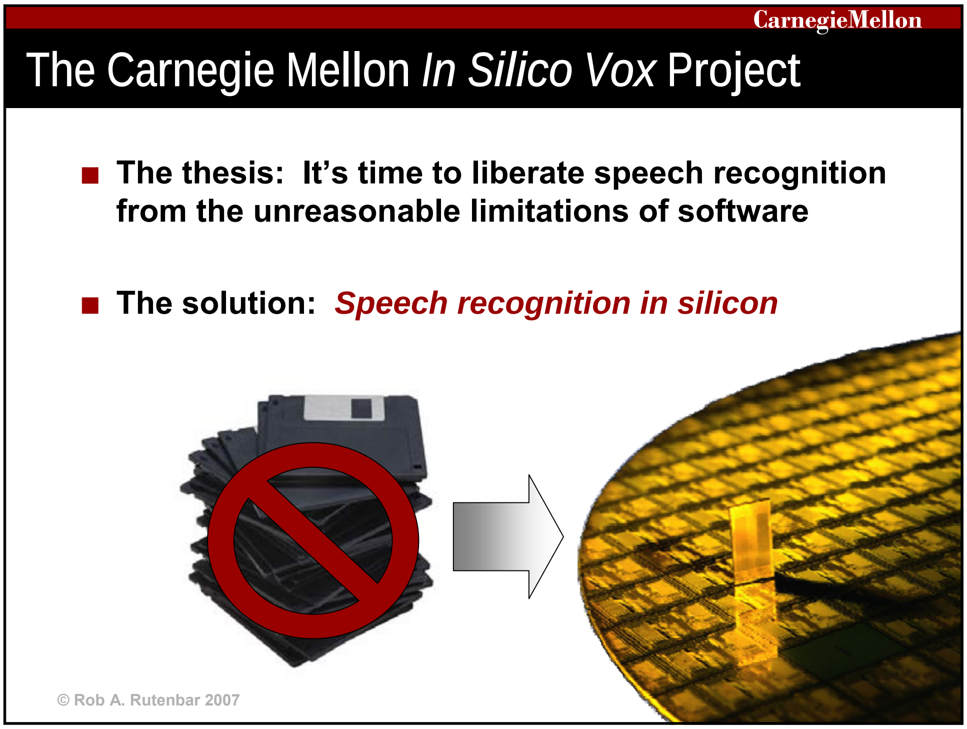
- **Best-quality recognition is computationally *hard***
  - For speaker-independent, large-vocabulary, continuous speech
- **1-10-100-1000 rule**

For	**~1X**	real-time recognition rate
For	**~10%**	word error rate (90% accuracy)
Need	**~100 MB**	memory footprint
Need	**~100 W**	power
Need	**~1000 MHz**	CPU

- **This proves to be very *limiting* …**

© Rob A. Rutenbar 2007

Slide 3

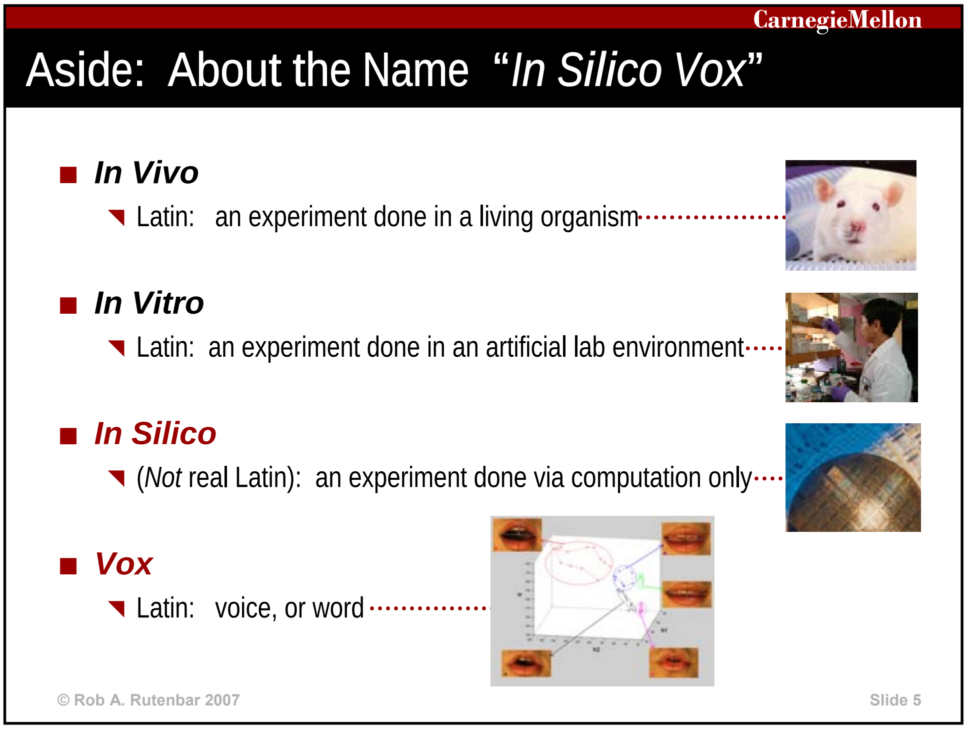


CarnegieMellon

The Carnegie Mellon *In Silico Vox* Project

- **The thesis: It's time to liberate speech recognition from the unreasonable limitations of software**
- **The solution: *Speech recognition in silicon***

© Rob A. Rutenbar 2007



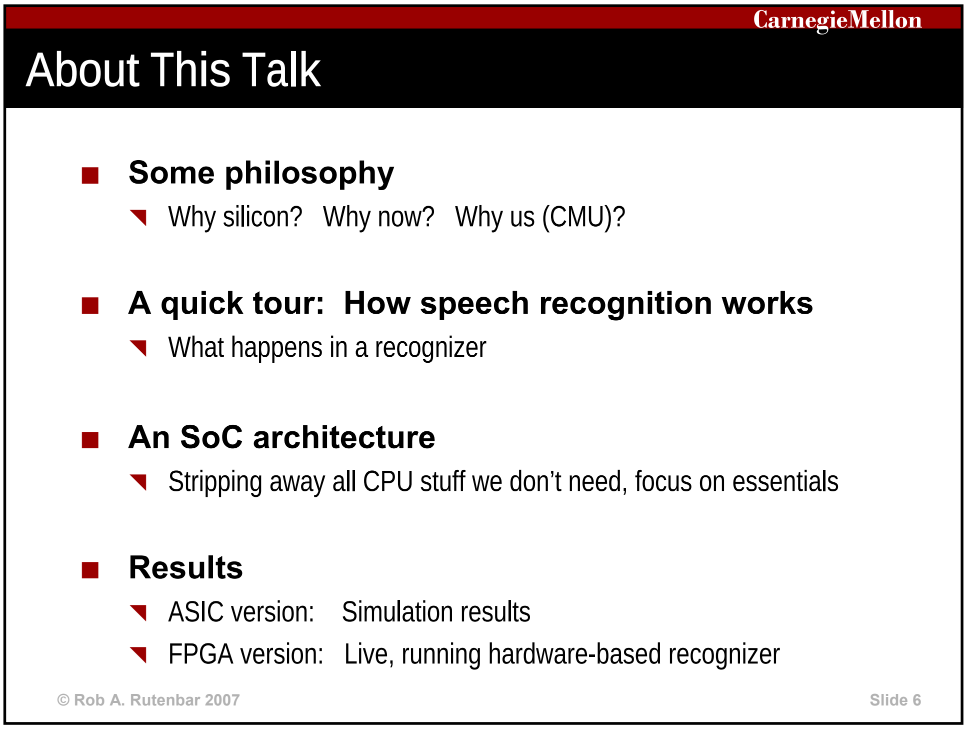
CarnegieMellon

Aside: About the Name "*In Silico Vox*"

- ***In Vivo***
  - Latin: an experiment done in a living organism
- ***In Vitro***
  - Latin: an experiment done in an artificial lab environment
- ***In Silico***
  - (*Not* real Latin): an experiment done via computation only
- ***Vox***
  - Latin: voice, or word

© Rob A. Rutenbar 2007

Slide 5



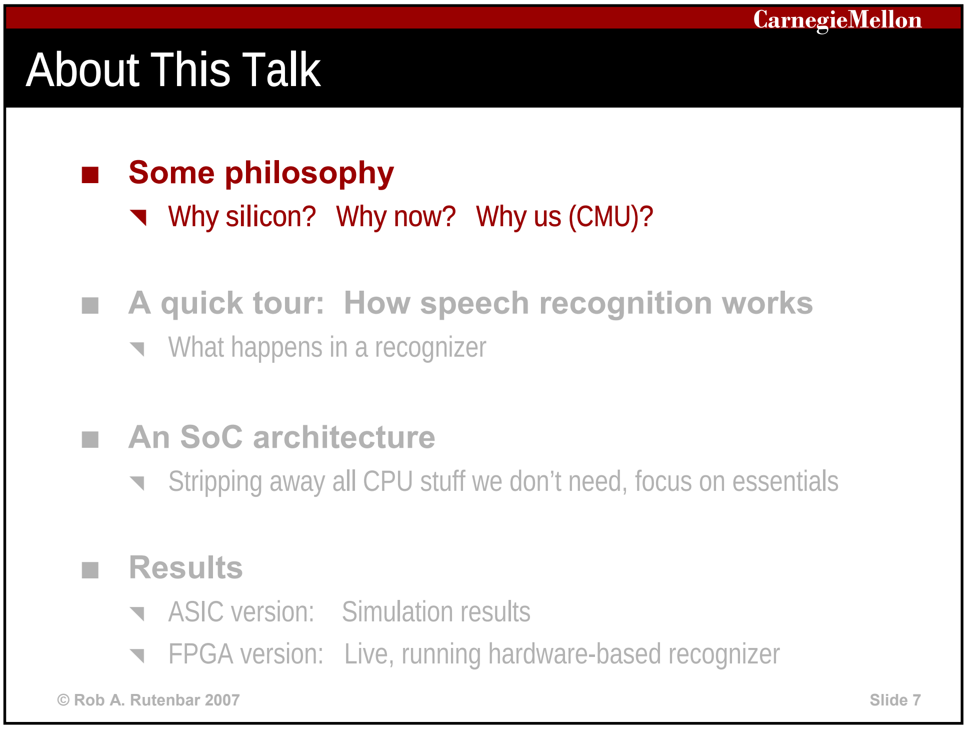
CarnegieMellon

About This Talk

- **Some philosophy**
  - Why silicon? Why now? Why us (CMU)?
- **A quick tour: How speech recognition works**
  - What happens in a recognizer
- **An SoC architecture**
  - Stripping away all CPU stuff we don't need, focus on essentials
- **Results**
  - ASIC version: Simulation results
  - FPGA version: Live, running hardware-based recognizer

© Rob A. Rutenbar 2007

Slide 6

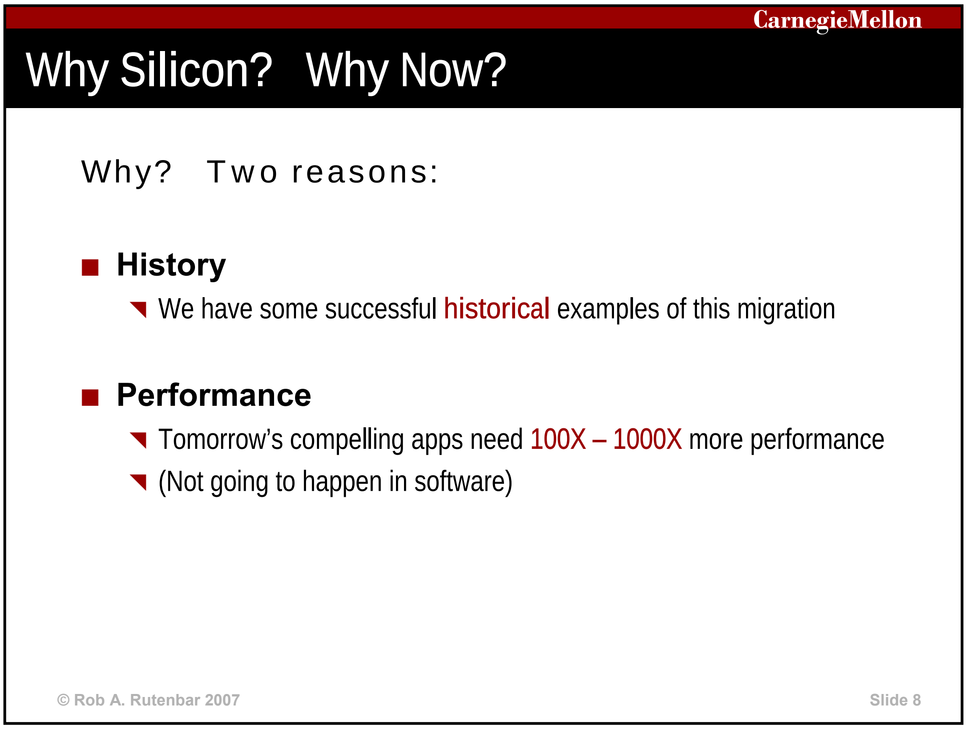


CarnegieMellon

About This Talk

- **Some philosophy**
  - Why silicon? Why now? Why us (CMU)?
- **A quick tour: How speech recognition works**
  - What happens in a recognizer
- **An SoC architecture**
  - Stripping away all CPU stuff we don't need, focus on essentials
- **Results**
  - ASIC version: Simulation results
  - FPGA version: Live, running hardware-based recognizer

© Rob A. Rutenbar 2007 Slide 7



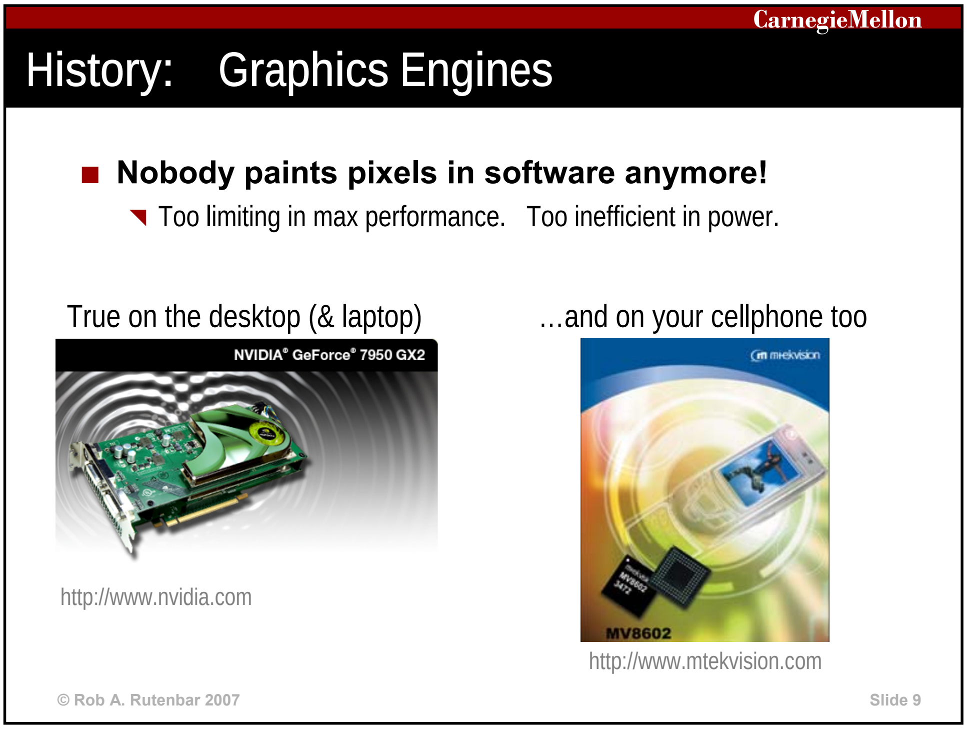
CarnegieMellon

Why Silicon? Why Now?

Why? Two reasons:

- **History**
  - We have some successful historical examples of this migration
- **Performance**
  - Tomorrow's compelling apps need 100X – 1000X more performance
  - (Not going to happen in software)

© Rob A. Rutenbar 2007 Slide 8



CarnegieMellon

History: Graphics Engines

- Nobody paints pixels in software anymore!
  - Too limiting in max performance. Too inefficient in power.

True on the desktop (& laptop)

NVIDIA® GeForce® 7950 GX2

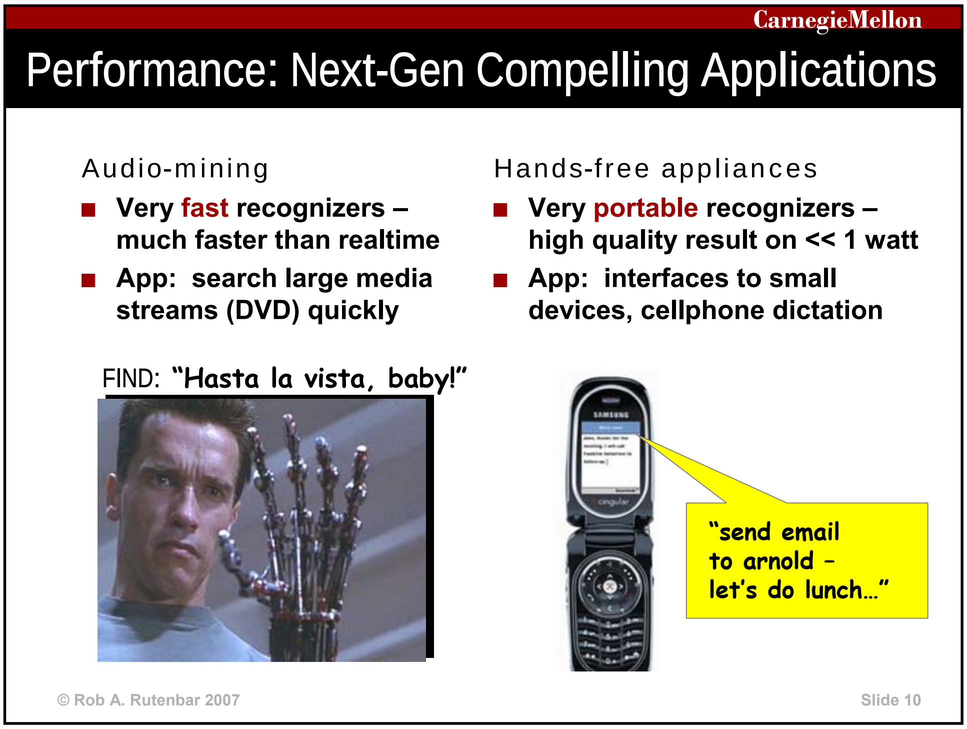
http://www.nvidia.com

…and on your cellphone too

mtekvision
MV8602

http://www.mtekvision.com

© Rob A. Rutenbar 2007 Slide 9



CarnegieMellon

Performance: Next-Gen Compelling Applications

Audio-mining

- **Very fast recognizers – much faster than realtime**
- **App: search large media streams (DVD) quickly**

Hands-free appliances

- **Very portable recognizers – high quality result on << 1 watt**
- **App: interfaces to small devices, cellphone dictation**

FIND: "Hasta la vista, baby!"

SAMSUNG
cingular

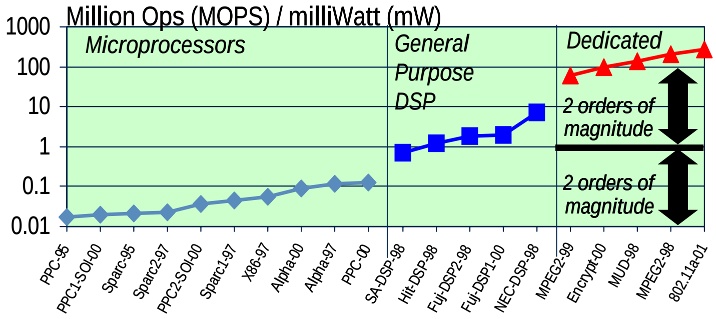
"send email to arnold – let's do lunch…"

© Rob A. Rutenbar 2007 Slide 10

CarnegieMellon

Silicon Solution: Speed *and* Power Wins

- **A famous graph from Prof. Bob Brodersen of Berkeley**
  - Study looked at 20 designs published at ISSCC, from 1997-2002
  - In slightly older technologies, relative to today: 180nm – 250nm
  - Dedicated designs up to 10,000X better energy efficiency (MOPS/mW)

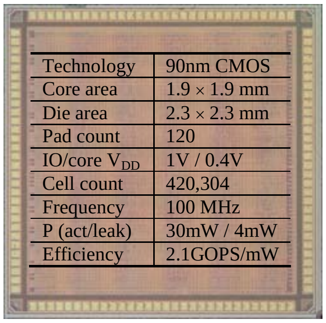


© Rob A. Rutenbar 2007 Slide 11

CarnegieMellon

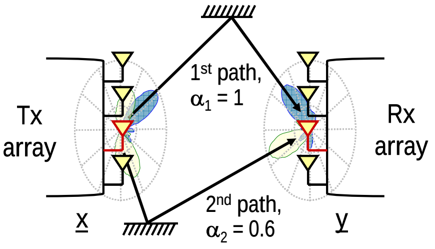
Recent Example: Parallel Radio Baseband DSP

- **90nm CMOS: adaptive DSP for multipath MIMO channel**
  - Power efficiency = 2.1GOPS/mW
  - Area efficiency = 20GOPS/mm^2



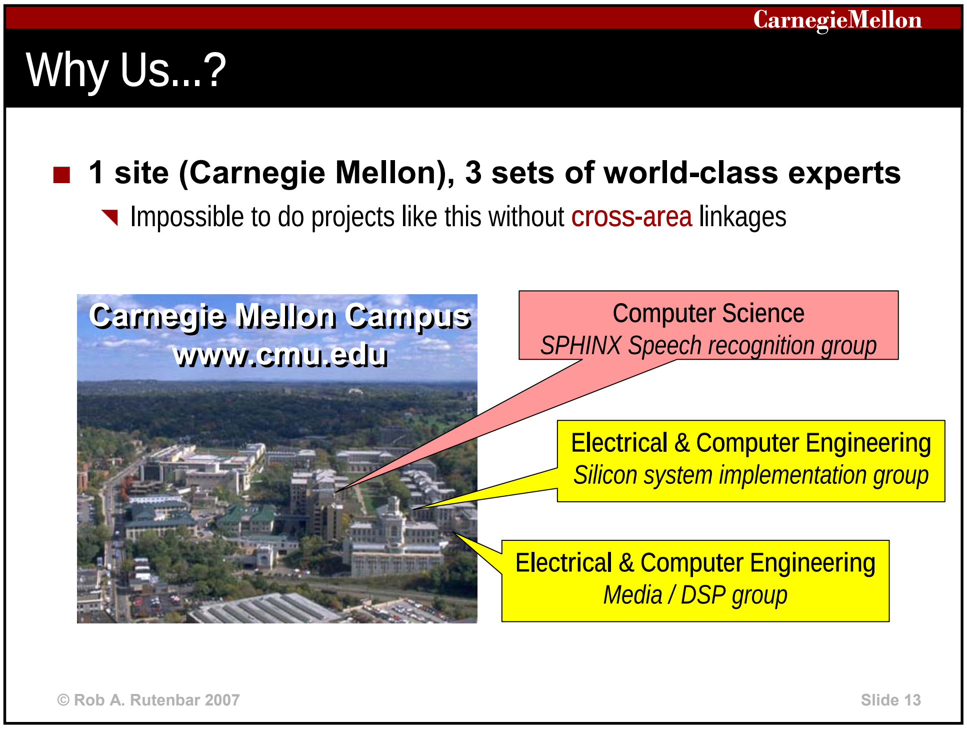
Technology	90nm CMOS
Core area	1.9 × 1.9 mm
Die area	2.3 × 2.3 mm
Pad count	120
IO/core V_{DD}	1V / 0.4V
Cell count	420,304
Frequency	100 MHz
P (act/leak)	30mW / 4mW
Efficiency	2.1GOPS/mW

(Source: Prof. Dejan Markovitz, UCLA)



Data rate up to 250Mbps over 16 sub-carriers
Measured 34mW @ VDD=385mV

© Rob A. Rutenbar 2007 Slide 12



CarnegieMellon

Why Us...?

- **1 site (Carnegie Mellon), 3 sets of world-class experts**
  - Impossible to do projects like this without cross-area linkages

Carnegie Mellon Campus
www.cmu.edu

Computer Science
SPHINX Speech recognition group

Electrical & Computer Engineering
Silicon system implementation group

Electrical & Computer Engineering
Media / DSP group

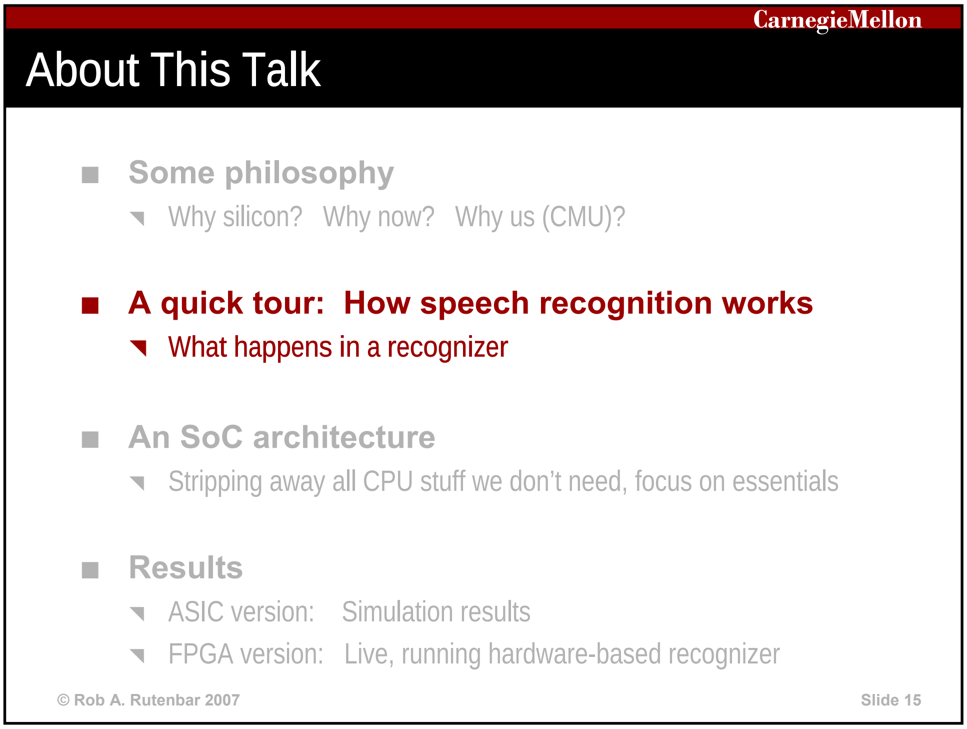
© Rob A. Rutenbar 2007

Slide 13

CarnegieMellon

Us: the CMU *In Silico Vox* Team

From left: Kai Yu, Rob Rutenbar, Edward Lin, Richard Stern, Tsuhan Chen, Patrick Bourke (not shown: Jeff Johnston)



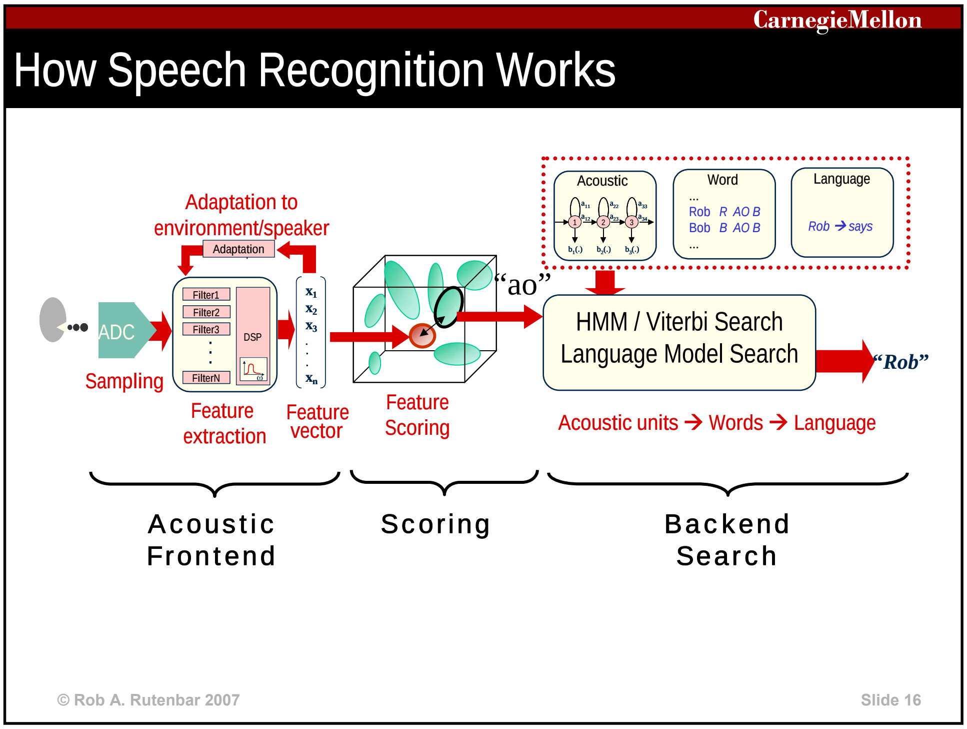
CarnegieMellon

About This Talk

- **Some philosophy**
  - Why silicon? Why now? Why us (CMU)?
- **A quick tour: How speech recognition works**
  - What happens in a recognizer
- **An SoC architecture**
  - Stripping away all CPU stuff we don't need, focus on essentials
- **Results**
  - ASIC version: Simulation results
  - FPGA version: Live, running hardware-based recognizer

© Rob A. Rutenbar 2007

Slide 15



CarnegieMellon

How Speech Recognition Works

HMM / Viterbi Search
Language Model Search

Sampling

Feature extraction

Feature vector

Feature Scoring

Acoustic units → Words → Language

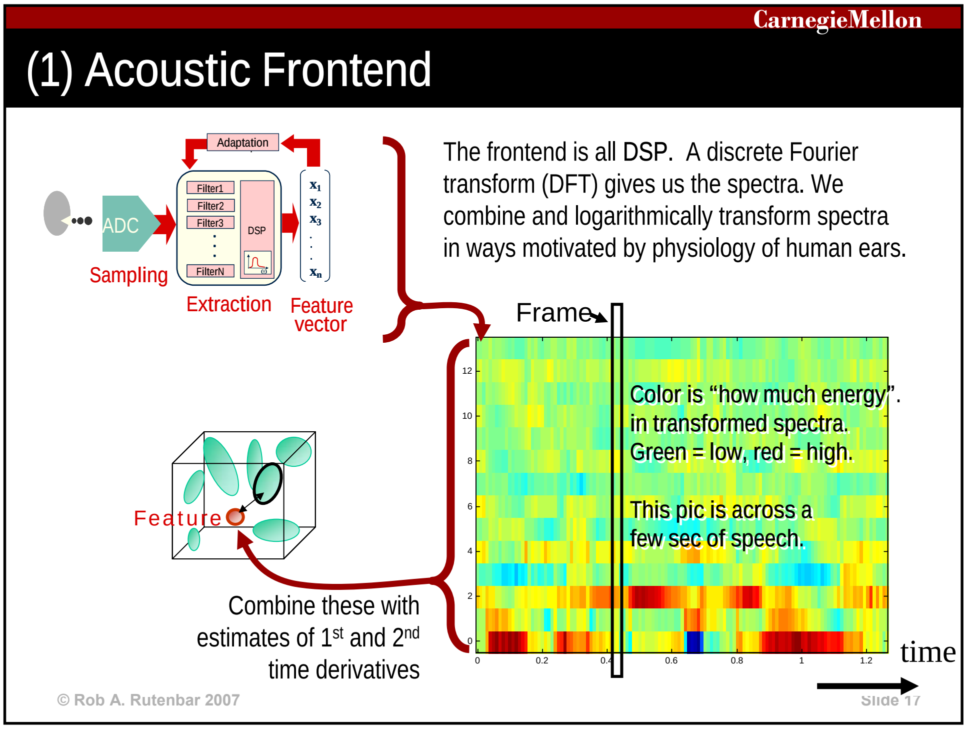
Acoustic Frontend

Scoring

Backend Search

© Rob A. Rutenbar 2007

Slide 16



CarnegieMellon

(1) Acoustic Frontend

The frontend is all DSP. A discrete Fourier transform (DFT) gives us the spectra. We combine and logarithmically transform spectra in ways motivated by physiology of human ears.

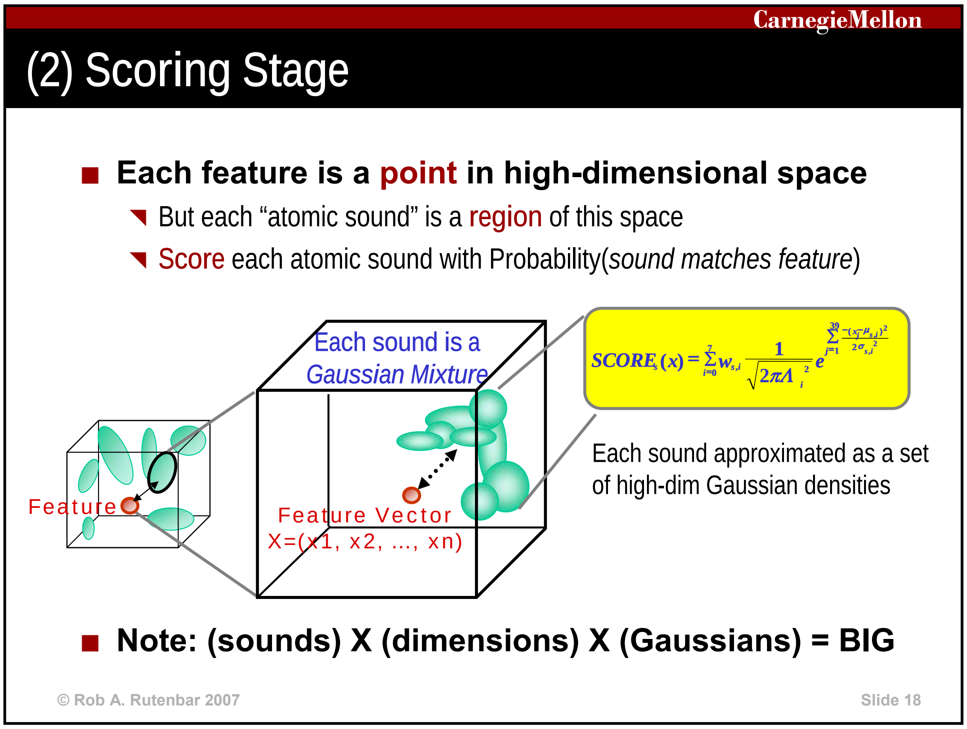
Color is "how much energy". in transformed spectra. Green = low, red = high.

This pic is across a few sec of speech.

Combine these with estimates of 1st and 2nd time derivatives

© Rob A. Rutenbar 2007

Slide 17



CarnegieMellon

(2) Scoring Stage

- **Each feature is a point in high-dimensional space**
  - But each "atomic sound" is a region of this space
  - Score each atomic sound with Probability(*sound matches feature*)

$$SCORE_s(x) = \sum_{i=0}^{7} w_{s,i} \frac{1}{\sqrt{2\pi \Lambda_i^2}} e^{\sum_{j=1}^{39} \frac{-(x_j-\mu_{s,i})^2}{2\sigma_{s,i}^2}}$$

Each sound approximated as a set of high-dim Gaussian densities

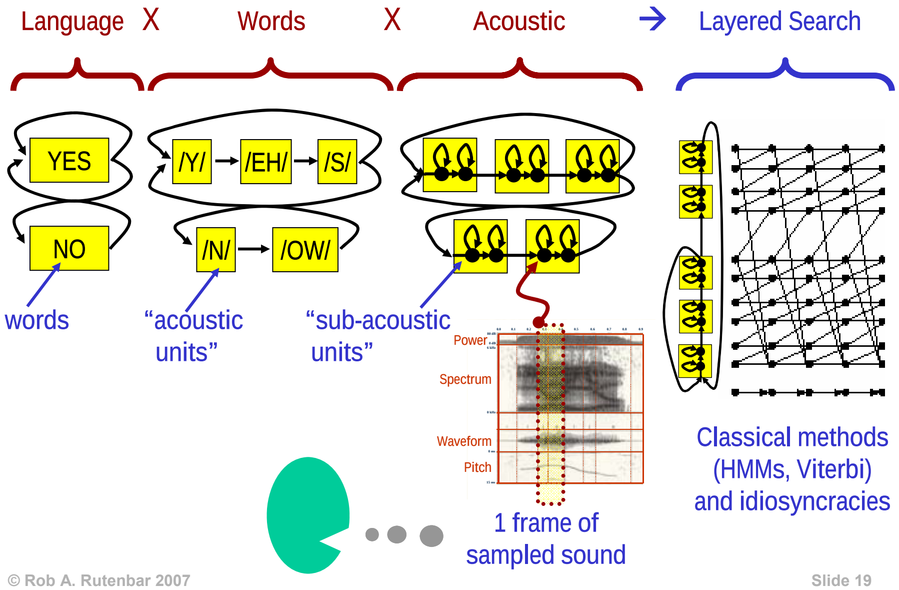
- **Note: (sounds) X (dimensions) X (Gaussians) = BIG**

© Rob A. Rutenbar 2007

Slide 18

CarnegieMellon

(3) Search: Speech Models are *Layered* Models



Language X Words X Acoustic → Layered Search

words

"acoustic units"

"sub-acoustic units"

1 frame of sampled sound

Classical methods (HMMs, Viterbi) and idiosyncracies

© Rob A. Rutenbar 2007

Slide 19

CarnegieMellon

Context Matters: At Bottom -- *Triphones*

- **English has ~50 atomic sounds (phones) but we recognize ~50x50x50 context-dependent triphones**
  - Because "I" sound in "five" is different than the "I" in "nine"

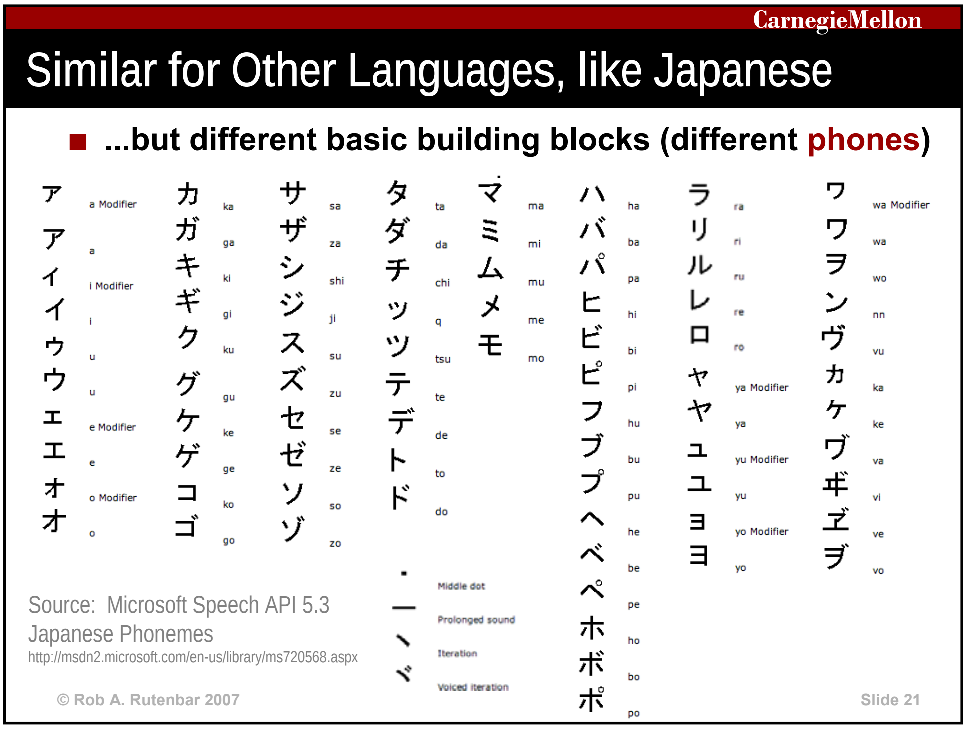
Five $F(-, I)_{\text{cross-word}}$ $I(F, V)_{\text{word-internal}}$ $V(I, -)_{\text{cross-word}}$

Nine $N(-, I)_{\text{cross-word}}$ $I(N, N)_{\text{word-internal}}$ $N(I, -)_{\text{cross-word}}$

"I" in "five" ≠ "I" in "nine"

© Rob A. Rutenbar 2007

Slide 20



CarnegieMellon

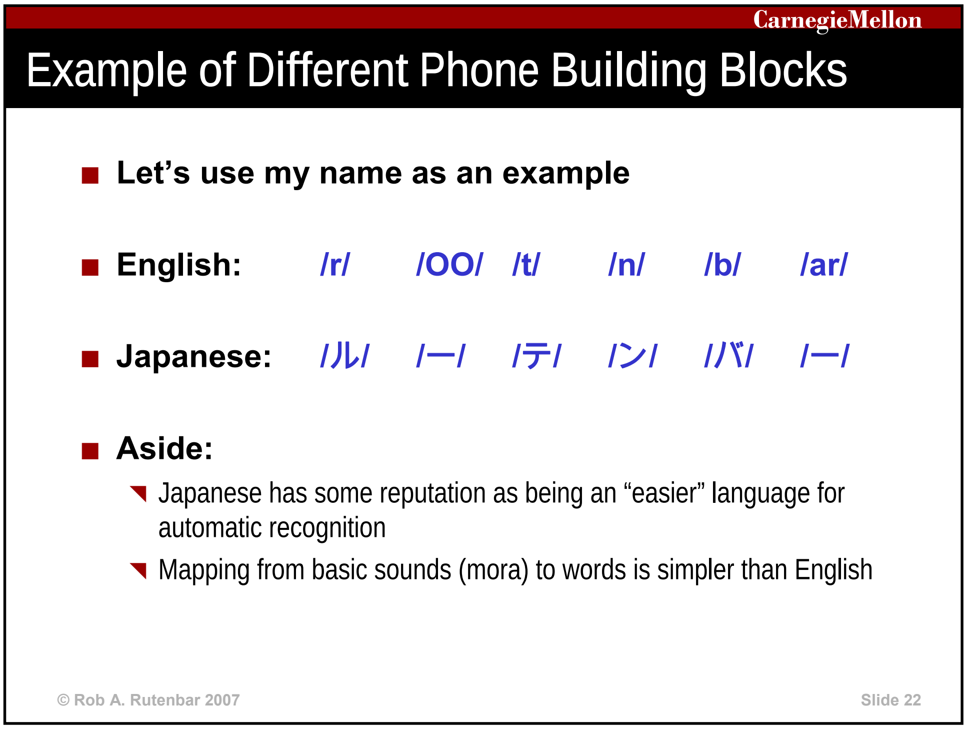
Similar for Other Languages, like Japanese

- ...but different basic building blocks (different **phones**)

ァ	a Modifier
ア	a
ィ	i Modifier
イ	i
ゥ	u
ウ	u
ェ	e Modifier
エ	e
ォ	o Modifier
オ	o
カ	ka
ガ	ga
キ	ki
ギ	gi
ク	ku
グ	gu
ケ	ke
ゲ	ge
コ	ko
ゴ	go
サ	sa
ザ	za
シ	shi
ジ	ji
ス	su
ズ	zu
セ	se
ゼ	ze
ソ	so
ゾ	zo
タ	ta
ダ	da
チ	chi
ッ	q
ツ	tsu
テ	te
デ	de
ト	to
ド	do
・	Middle dot
ー	Prolonged sound
ヽ	Iteration
ヾ	Voiced iteration
マ	ma
ミ	mi
ム	mu
メ	me
モ	mo
ハ	ha
バ	ba
パ	pa
ヒ	hi
ビ	bi
ピ	pi
フ	hu
ブ	bu
プ	pu
ヘ	he
ベ	be
ペ	pe
ホ	ho
ボ	bo
ポ	po
ラ	ra
リ	ri
ル	ru
レ	re
ロ	ro
ャ	ya Modifier
ヤ	ya
ュ	yu Modifier
ユ	yu
ョ	yo Modifier
ヨ	yo
ヮ	wa Modifier
ワ	wa
ヲ	wo
ン	nn
ヴ	vu
ヵ	ka
ヶ	ke
ヷ	va
ヸ	vi
ヹ	ve
ヺ	vo

Source: Microsoft Speech API 5.3 Japanese Phonemes
http://msdn2.microsoft.com/en-us/library/ms720568.aspx

© Rob A. Rutenbar 2007 Slide 21

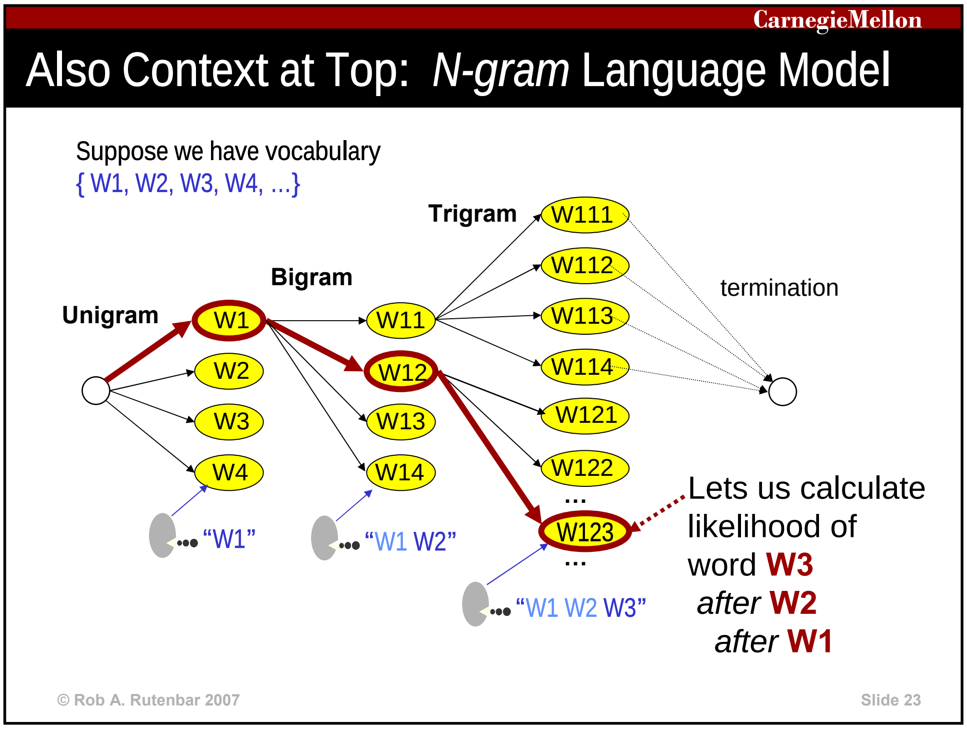


CarnegieMellon

Example of Different Phone Building Blocks

- **Let's use my name as an example**
- **English:** /r/ /OO/ /t/ /n/ /b/ /ar/
- **Japanese:** /ル/ /ー/ /テ/ /ン/ /バ/ /ー/
- **Aside:**
  - Japanese has some reputation as being an "easier" language for automatic recognition
  - Mapping from basic sounds (mora) to words is simpler than English

© Rob A. Rutenbar 2007 Slide 22



CarnegieMellon

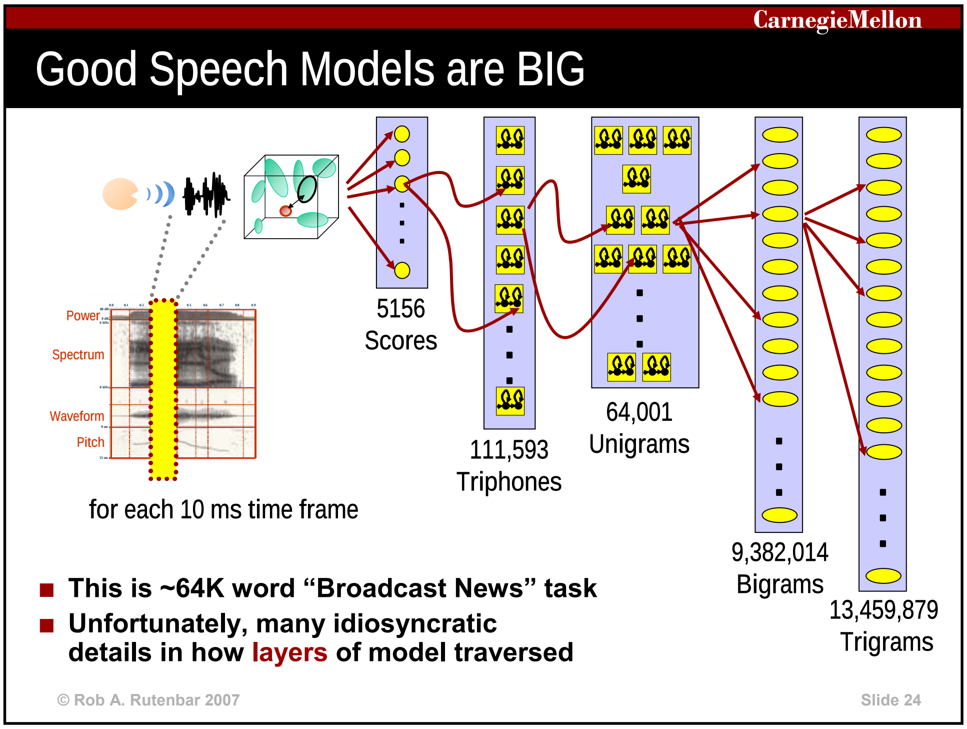
Also Context at Top: *N-gram* Language Model

Suppose we have vocabulary
{ W1, W2, W3, W4, …}

Lets us calculate likelihood of word **W3** *after* **W2** *after* **W1**

© Rob A. Rutenbar 2007

Slide 23



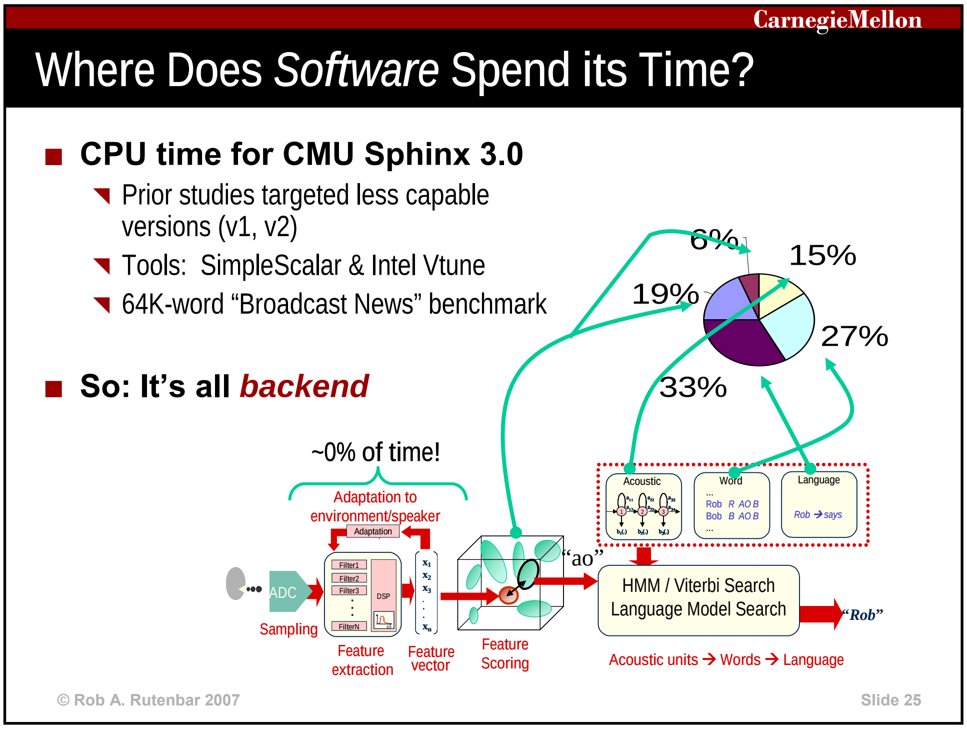
CarnegieMellon

Good Speech Models are BIG

- This is ~64K word "Broadcast News" task
- Unfortunately, many idiosyncratic details in how layers of model traversed

© Rob A. Rutenbar 2007

Slide 24



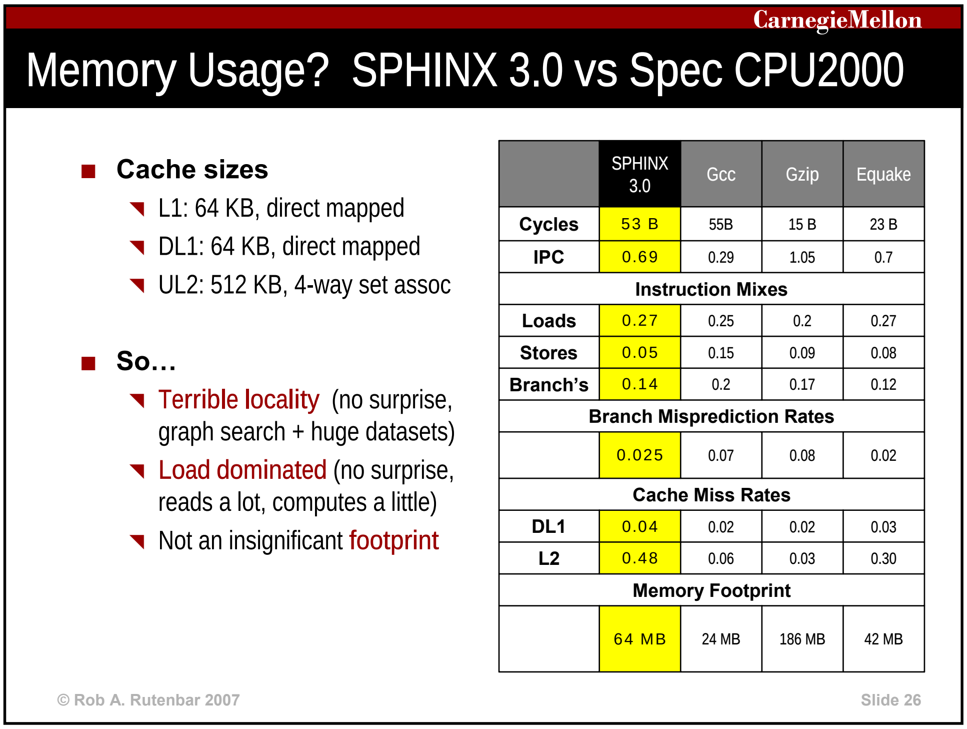
CarnegieMellon

Where Does *Software* Spend its Time?

- **CPU time for CMU Sphinx 3.0**
  - Prior studies targeted less capable versions (v1, v2)
  - Tools: SimpleScalar & Intel Vtune
  - 64K-word "Broadcast News" benchmark
- **So: It's all *backend***

© Rob A. Rutenbar 2007

Slide 25



CarnegieMellon

Memory Usage? SPHINX 3.0 vs Spec CPU2000

- **Cache sizes**
  - L1: 64 KB, direct mapped
  - DL1: 64 KB, direct mapped
  - UL2: 512 KB, 4-way set assoc
- **So…**
  - Terrible locality (no surprise, graph search + huge datasets)
  - Load dominated (no surprise, reads a lot, computes a little)
  - Not an insignificant footprint

	SPHINX 3.0	Gcc	Gzip	Equake
Cycles	53 B	55B	15 B	23 B
IPC	0.69	0.29	1.05	0.7
Instruction Mixes				
Loads	0.27	0.25	0.2	0.27
Stores	0.05	0.15	0.09	0.08
Branch's	0.14	0.2	0.17	0.12
Branch Misprediction Rates				
	0.025	0.07	0.08	0.02
Cache Miss Rates				
DL1	0.04	0.02	0.02	0.03
L2	0.48	0.06	0.03	0.30
Memory Footprint				
	64 MB	24 MB	186 MB	42 MB

© Rob A. Rutenbar 2007

Slide 26

Rob A. Rutenbar
Carnegie Mellon University

CarnegieMellon

About This Talk

- Some philosophy
  - Why silicon? Why now? Why us (CMU)?
- A quick tour: How speech recognition works
  - What happens in a recognizer
- **An SoC architecture**
  - Stripping away all CPU stuff we don't need, focus on essentials
- Results
  - ASIC version: Simulation results
  - FPGA version: Live, running hardware-based recognizer

© Rob A. Rutenbar 2007

Slide 27

CarnegieMellon

This Talk: How to Get to *Fast*...

Audio-mining

- **Very fast recognizers – much faster than realtime**
- **App: search large media streams (DVD) quickly**

Hands-free appliances

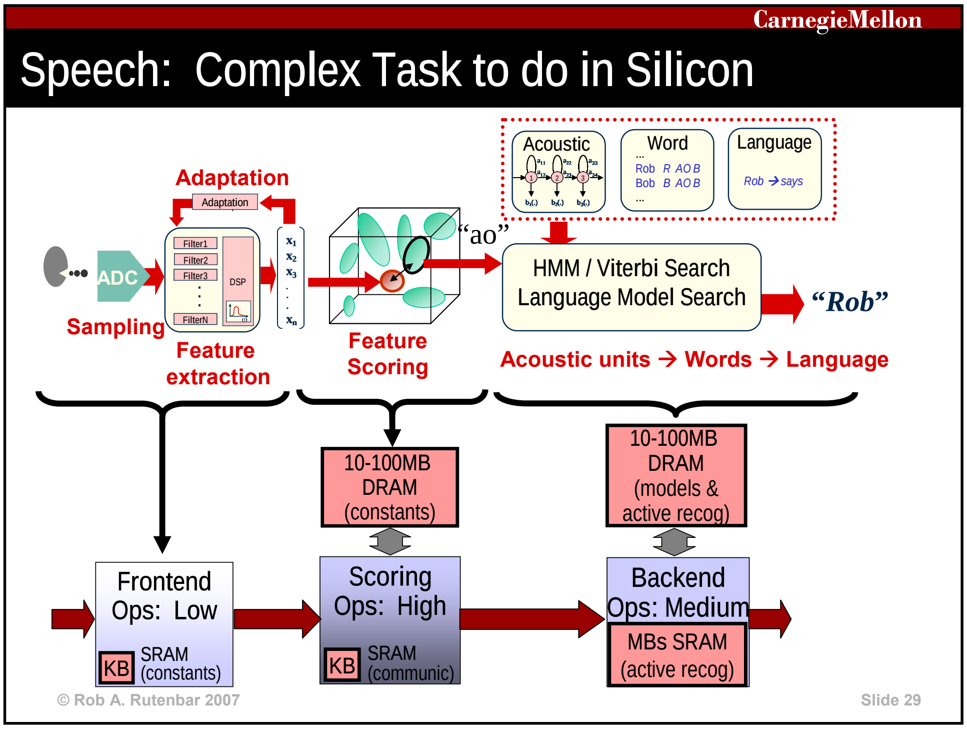
- Very portable recognizers – high quality result on << 1 watt
- App: interfaces to small devices, cellphone dictation

FIND: "Hasta la vista, baby!"

"send email to arnold – let's do lunch..."

© Rob A. Rutenbar 2007

Slide 28

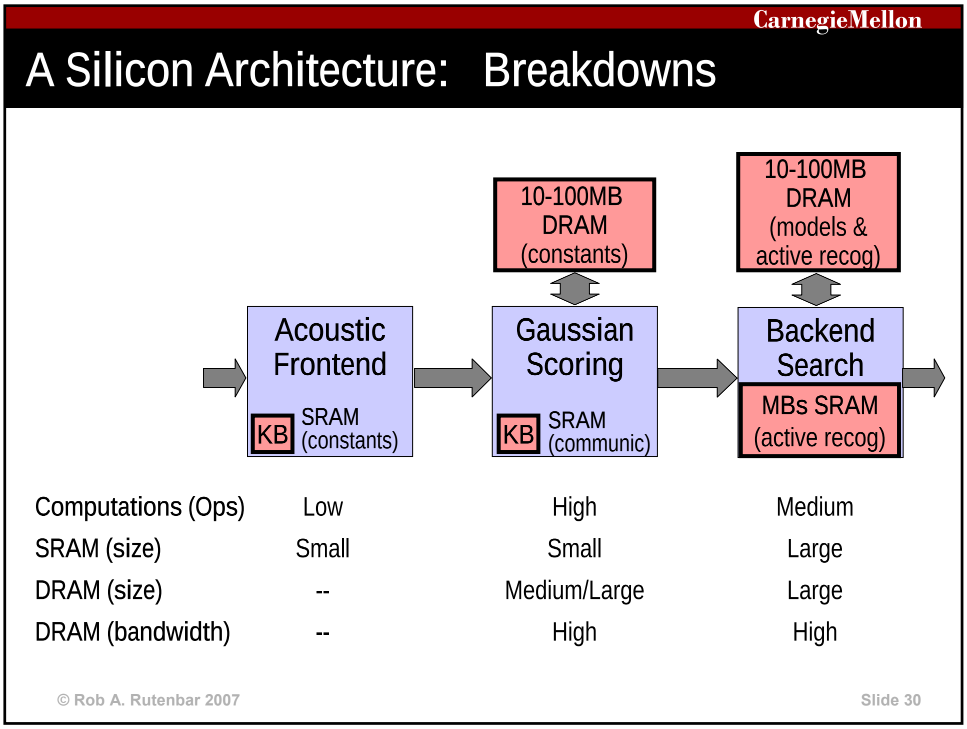


CarnegieMellon

Speech: Complex Task to do in Silicon

© Rob A. Rutenbar 2007

Slide 29



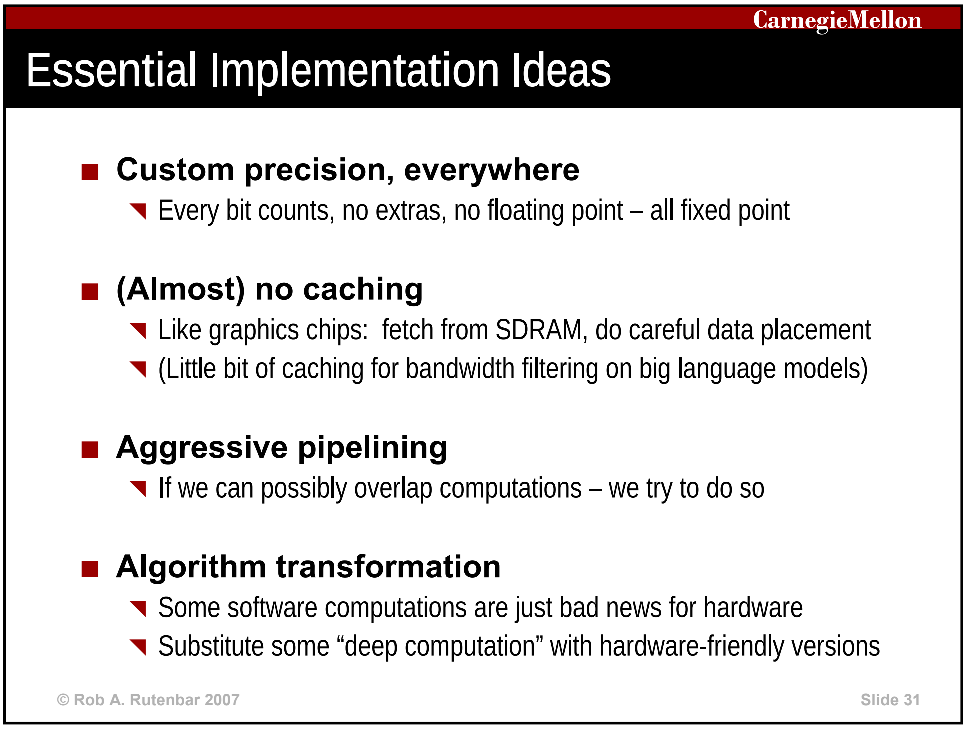
CarnegieMellon

A Silicon Architecture: Breakdowns

Computations (Ops)	Low	High	Medium
SRAM (size)	Small	Small	Large
DRAM (size)	--	Medium/Large	Large
DRAM (bandwidth)	--	High	High

© Rob A. Rutenbar 2007

Slide 30



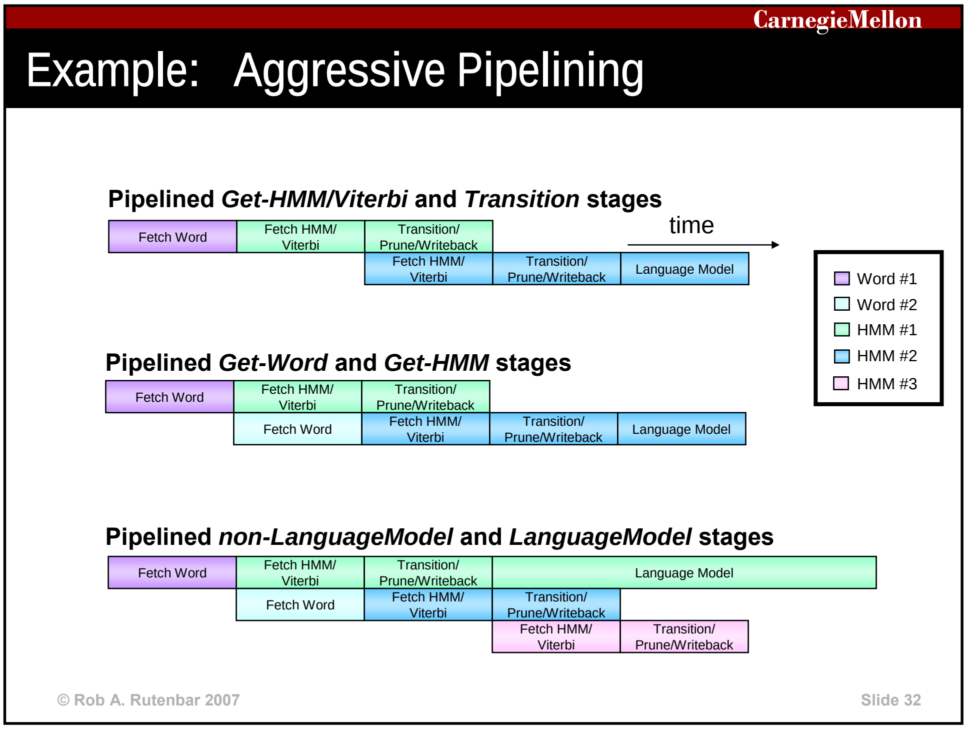
CarnegieMellon

Essential Implementation Ideas

- **Custom precision, everywhere**
  - Every bit counts, no extras, no floating point – all fixed point
- **(Almost) no caching**
  - Like graphics chips: fetch from SDRAM, do careful data placement
  - (Little bit of caching for bandwidth filtering on big language models)
- **Aggressive pipelining**
  - If we can possibly overlap computations – we try to do so
- **Algorithm transformation**
  - Some software computations are just bad news for hardware
  - Substitute some “deep computation” with hardware-friendly versions

© Rob A. Rutenbar 2007

Slide 31



CarnegieMellon

Example: Aggressive Pipelining

Pipelined *Get-HMM/Viterbi* and *Transition* stages

Pipelined *Get-Word* and *Get-HMM* stages

Pipelined *non-LanguageModel* and *LanguageModel* stages

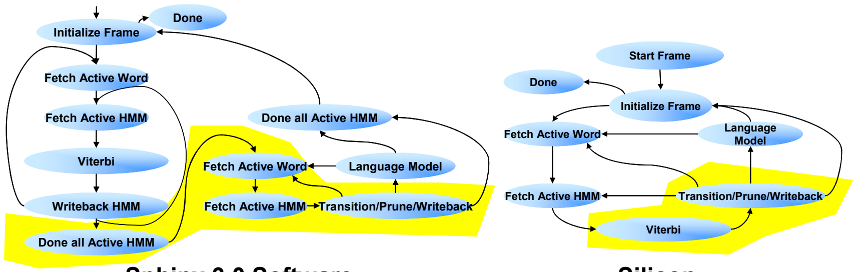
© Rob A. Rutenbar 2007

Slide 32

CarnegieMellon

Example: Algorithmic Changes

- **Acoustic-level pruning threshold**
  - Software: Use best score of *current* frame (after Viterbi on Active HMMs)
  - Silicon: Use best score of *previous* frame (nixes big temporal bottleneck)
- **Tradeoffs**
  - Less memory bandwidth, can pipeline, little pessimistic on scores



Sphinx 3.0 Software   Silicon

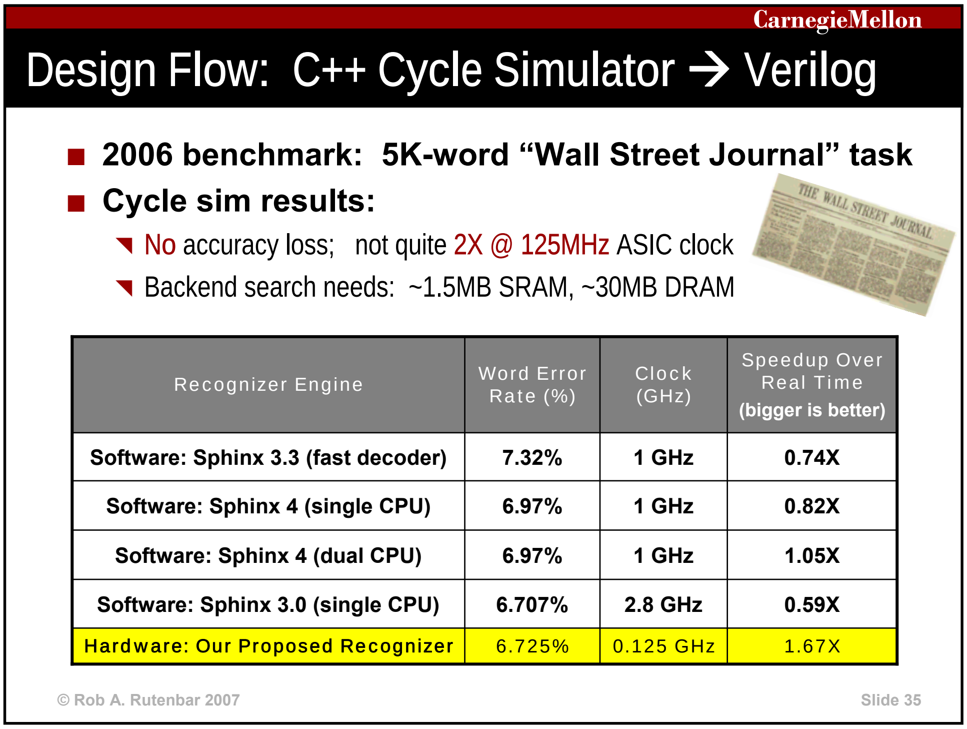
© Rob A. Rutenbar 2007 Slide 33

CarnegieMellon

About This Talk

- **Some philosophy**
  - Why silicon? Why now? Why us (CMU)?
- **A quick tour: How speech recognition works**
  - What happens in a recognizer
- **An SoC architecture**
  - Stripping away all CPU stuff we don't need, focus on essentials
- **Results**
  - ASIC version: Simulation results
  - FPGA version: Live, running hardware-based recognizer

© Rob A. Rutenbar 2007 Slide 34



CarnegieMellon

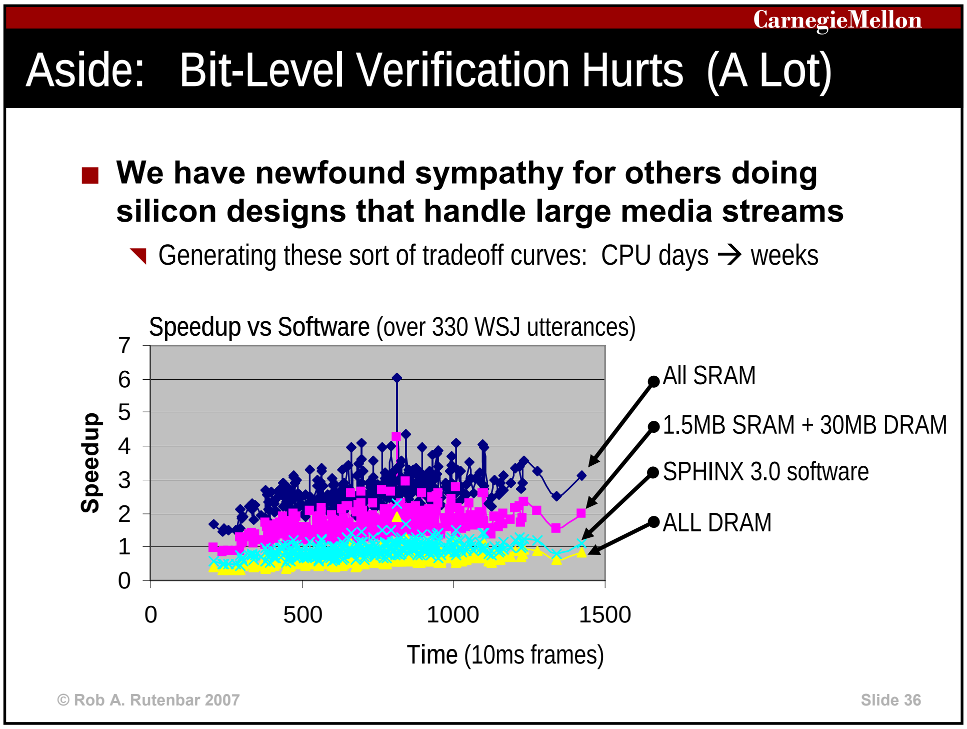
Design Flow: C++ Cycle Simulator → Verilog

- **2006 benchmark: 5K-word “Wall Street Journal” task**
- **Cycle sim results:**
  - No accuracy loss; not quite 2X @ 125MHz ASIC clock
  - Backend search needs: ~1.5MB SRAM, ~30MB DRAM

THE WALL STREET JOURNAL

Recognizer Engine	Word Error Rate (%)	Clock (GHz)	Speedup Over Real Time **(bigger is better)**
Software: Sphinx 3.3 (fast decoder)	**7.32%**	**1 GHz**	**0.74X**
Software: Sphinx 4 (single CPU)	**6.97%**	**1 GHz**	**0.82X**
Software: Sphinx 4 (dual CPU)	**6.97%**	**1 GHz**	**1.05X**
Software: Sphinx 3.0 (single CPU)	**6.707%**	**2.8 GHz**	**0.59X**
Hardware: Our Proposed Recognizer	6.725%	0.125 GHz	1.67X

© Rob A. Rutenbar 2007 Slide 35



CarnegieMellon

Aside: Bit-Level Verification Hurts (A Lot)

- **We have newfound sympathy for others doing silicon designs that handle large media streams**
  - Generating these sort of tradeoff curves: CPU days → weeks

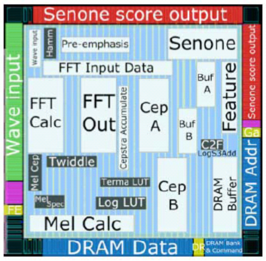
Speedup vs Software (over 330 WSJ utterances)

© Rob A. Rutenbar 2007 Slide 36

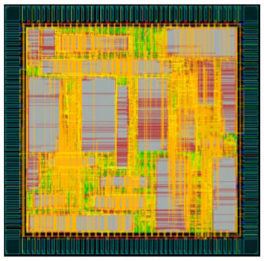
CarnegieMellon

Aside: Pieces of Design = Great Class Projects

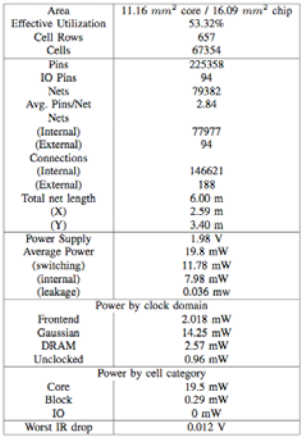
- CMU student team: Patrick Chiu, David Fu, Mark McCartney, Ajay Panagariya, Chris Thomas



Floorplan



Final Layout



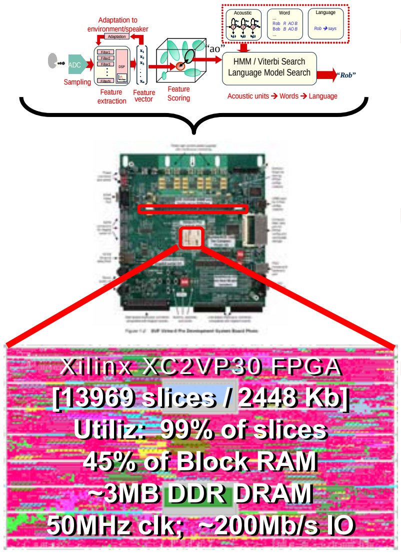
Area	11.16 mm^2 core / 16.09 mm^2 chip
Effective Utilization	53.32%
Cell Rows	657
Cells	67354
Pins	225358
IO Pins	94
Nets	79382
Avg. Pins/Net	2.84
Nets	
(Internal)	77977
(External)	94
Connections	
(Internal)	146621
(External)	188
Total net length	6.00 m
(X)	2.59 m
(Y)	3.40 m
Power Supply	1.98 V
Average Power	19.8 mW
(switching)	11.78 mW
(internal)	7.98 mW
(leakage)	0.036 mw
Power by clock domain	
Frontend	2.018 mW
Gaussian	14.25 mW
DRAM	2.57 mW
Unclocked	0.96 mW
Power by cell category	
Core	19.5 mW
Block	0.29 mW
IO	0 mW
Worst IR drop	0.012 V

Final Stats

© Rob A. Rutenbar 2007

CarnegieMellon

A Complete Live Recognizer: FPGA Demo



Xilinx XC2VP30 FPGA
[13969 slices / 2448 Kb]
Utiliz: 99% of slices
45% of Block RAM
~3MB DDR DRAM
50MHz clk; ~200Mb/s IO

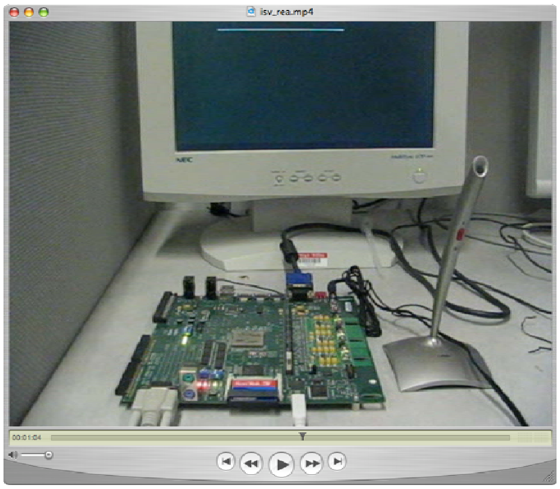
- In any "system design" research, you reach a point where you just want to see it work – *for real*
- Goal: *Full recognizer 1 FPGA + 1 DRAM*
- A benchmark that fits on chip
  - 1000-word "Resource Mgt" task
  - Slightly simplified: no tri-grams
  - Slower: not real time, ~2.3X slower
  - Resource limited: slices, mem bandwidth

© Rob A. Rutenbar 2007

CarnegieMellon

FPGA Experimental Results

- **Aside: as far as we know, this is the *most complex* recognizer architecture ever fully mapped into a running, hardware-only form**

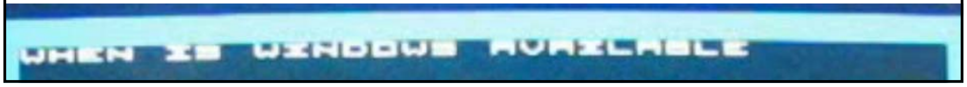


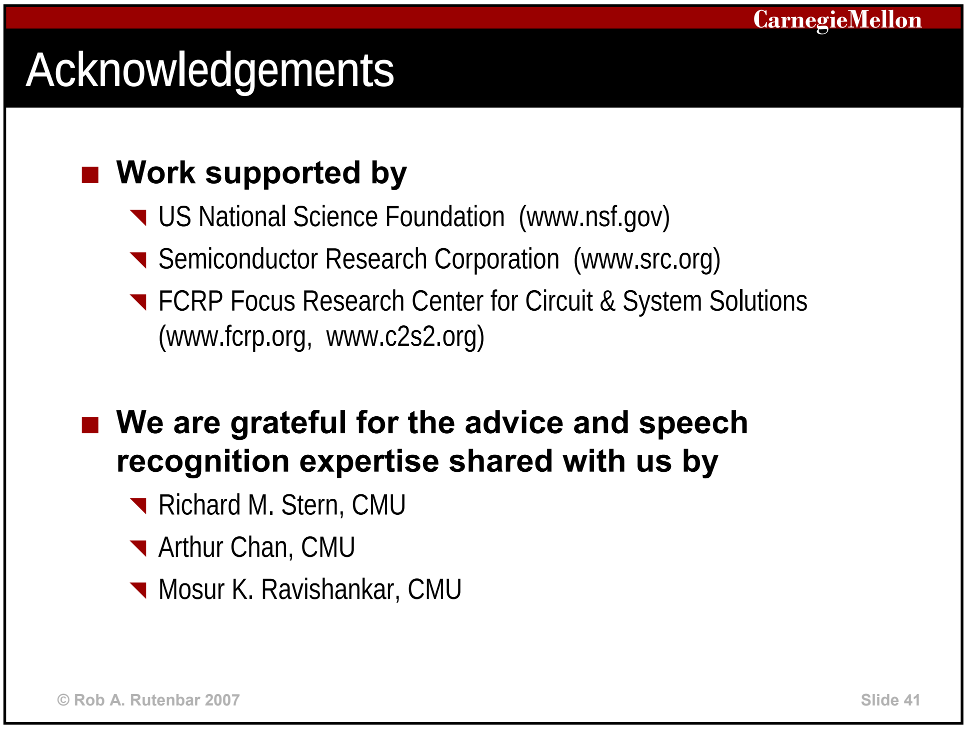
© Rob A. Rutenbar 2007 Slide 39

CarnegieMellon

Summary

- **Software is too constraining for speech recognition**
  - Evolution of graphics chips suggests alternative: Do it in silicon
  - Compelling performance and power reasons for silicon speech recog
- **Several "*in silico vox*" architectures in design**
  - SoC and FPGA versions
  - ~10X realtime speedup architecture in progress at CMU
- **Reflections**
  - Some of the most interesting experiences happen when you get people from very different backgrounds – silicon + speech – on same team


WHEN IS WINDOWS AVAILABLE



CarnegieMellon

Acknowledgements

- **Work supported by**
  - US National Science Foundation (www.nsf.gov)
  - Semiconductor Research Corporation (www.src.org)
  - FCRP Focus Research Center for Circuit & System Solutions (www.fcrp.org, www.c2s2.org)
- **We are grateful for the advice and speech recognition expertise shared with us by**
  - Richard M. Stern, CMU
  - Arthur Chan, CMU
  - Mosur K. Ravishankar, CMU

© Rob A. Rutenbar 2007