



# PowerAnalyzer for Pocket Computers

Todd Austin and Trevor Mudge  
University of Michigan

Dirk Grunwald  
University of Colorado



# Project Overview



## ■ Project Goal

- Develop the first high-performance validated architectural-level power model of a power-sensitive embedded target

## ■ PowerAnalyzer

- Cycle-level architectural simulator for early power/performance studies
- Capable of system-wide simulation accounting for OS and I/O behavior
- Calibrated against detailed physical models and a real system (iPAQ)

## ■ Deliverables

- SimpleScalar ARM simulator and iPAQ device performance models
- MiBench embedded benchmark suite
- PowerAnalyzer technology integrated into SimpleScalar/ARM models



# Team Members



## ■ University of Michigan

- Trevor Mudge
- Todd Austin
- Students:
  - Dan Ernst
  - Nam Kim
  - Jeff Ringenberg

## ■ University of Colorado

- Dirk Grunwald
- Students:
  - Soraya Ghiasi
  - Jason Casmira

## ■ Industrial Partners

- Intel
  - Support for two students
  - Mentors - George Cai (Texas), Doug Carmean and Rich Uhlig (Portland)



# Accomplishments to Date



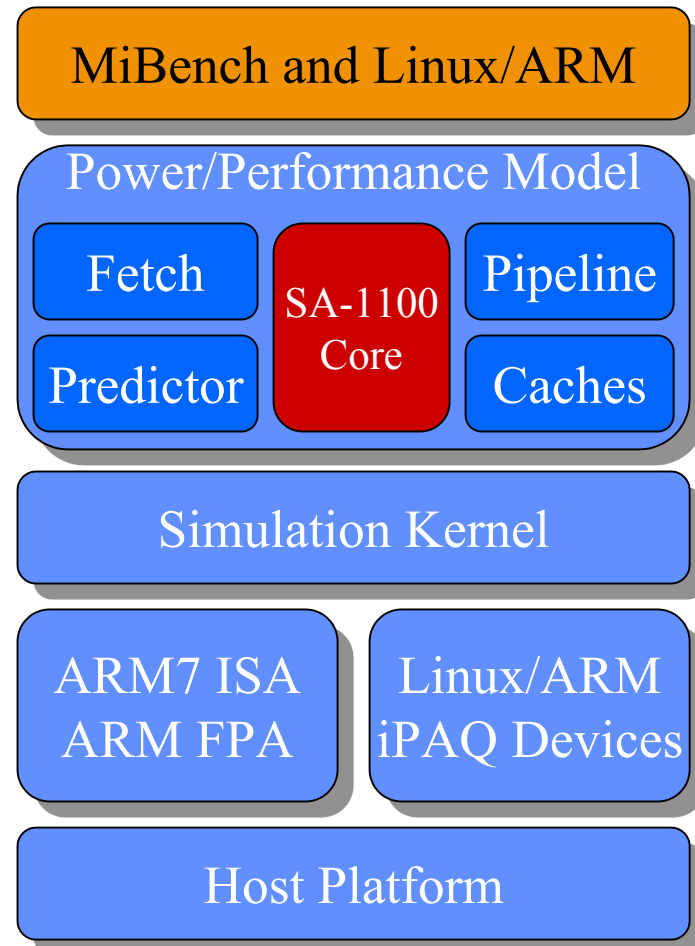
- SimpleScalar/ARM
  - Retargeted SimpleScalar instruction set simulator to ARM7 ISA
  - Available to PAC/C researchers
- MiBench embedded benchmark suite
  - Assembled 22 benchmarks from 6 embedded program domains
  - Available to PAC/C researchers pre-compiled for SimpleScalar/ARM target
- PowerAnalyzer technology
  - Detailed studies of previous efforts completed
  - Component-level power model development and calibration underway



# SimpleScalar/ARM



- Based on SimpleScalar Tool Set
  - Developed by SimpleScalar LLC
  - Freely available with source to academic sites
- Target applications run on emulator
  - SPEC'95, SPEC'2k, MiBench
  - Linux/ARM (*ongoing*)
- Power/performance models tracks time
  - Sim-outorder high performance core model
  - SA-1100 performance core (*ongoing*)
- ARM instruction set support
  - ARM 7 ISA and Floating Point Accelerator
- Multiple target I/O support
  - Easy to use Linux/ARM syscall target
  - Detailed iPAQ device model (*ongoing*)
- Flexible host support
  - Full function on Linux, SPARC, NT, and OSF





# SimpleScalar/ARM



```

InitFlats.....
InitSprites
InitColormaps
R_InitData
R_InitPointToAngle
R_InitTables
R_InitPlanes
R_InitLightTables
R_InitSkyMap
R_InitTranslationsTables
P_Init: Init Playloop state.
I_Init: Setting up machine state.
I_InitSound: configured audio device
I_InitSound: pre-cached all sound data
I_InitSound: sound module ready
D_CheckNetGame: Checking network game status.
startskill 2 deathmatch: 0 startmap: 1 startepisode: 1
player 1 of 1 (1 nodes)
S_Init: Setting up sound.
S_Init: default sfx volume 8
HU_Init: Setting up heads up display.
ST_Init: Init status bar.
warning: partially supported sigaction() call...

```

<b>7</b>	<b>85%</b>	2 3 4 5 6 7		<b>100%</b>	BULL 42 / 200 SHEL 700 / 50 ROCK / 50 CELL 0 / 300
AMMO	HEALTH	ARMS		ARMOR	



# SimpleScalar/ARM Frontend GUI



File Settings Help

**Simulation Settings**

**Simulator :** /nfs/binomial/k/simplesim-3.0/sim-outorder **Permissions :** File Exists,r,w,x  
**Run Script :** /nfs/binomial/k/scripts/newspecrun **Permissions :** File Exists,r,w,x  
**Where to run the program at :** /z/ **Permissions :** Dir Exists,d,r,w,x  
**Where to store results :** /nfs/binomial/k/results **Permissions :** Doesn't Exist!!  
**Simple Scalar Config file :** config/relax.cfg **Permissions :** File Exists,r,w,x  Use SS Config  
**Benchmark :**   
**Where to find benchmark :** /nfs/binomial/k/spec2000\_newcomp/ **Permissions :** Dir Exists,d,r,w,x  
**Benchmark Extension :** peak.ev6 **Test Size :** test  
**Simulation Tag :** myresults  
**Where to find bench inputs :** /nfs/binomial/k/spec2000\_newcomp/benchsj **Permissions :** Dir Exists,d,r,w,x

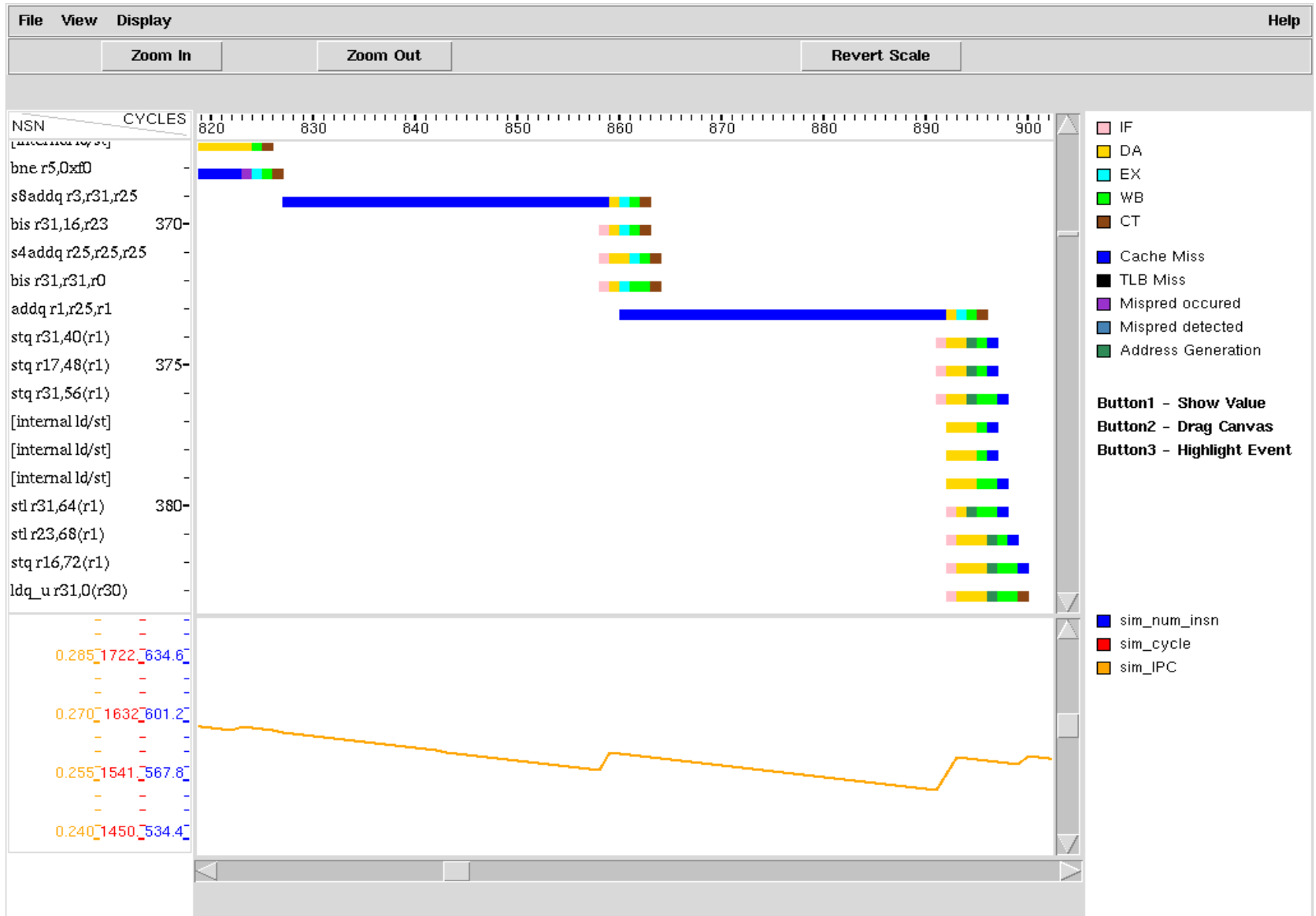
Simulator Option	Valid Values	Value	Simulator Option	Valid Values	Value
config	string	<null>	dumpconfig	string	<null>
h	true false	false	v	true false	false
d	true false	false	i	true false	false
seed	int	1	q	true false	false
chkpt	string	<null>	redir:sim	string	<null>
redir:prog	string	<null>	nice	int	0
max:inst	uint	0	fastfwd	int	0
ptrace	string list...	<null>	fetch:ifqsize	int	4
fetch:mplat	int	3	fetch:speed	int	1
bpred	string	bimod	bpred:bimod	int	2048

**Options Coloring Key:**

**Help Messages**



# SimpleScalar/ARM Visualization GUI





# MiBench Benchmark Suite



- Freely available embedded benchmark suite
  - Includes source code and benchmark inputs
  - With binaries compiled for SimpleScalar/ARM simulator
  - Preliminary report details benchmarks and performance characteristics
  
- Six embedded programming domains (22 benchmarks)
  - Automotive/industrial
    - Process control kernels from engine control, sensor monitoring
  - Networking
    - Shortest path router, Patricia tree, packet processor, CRC32
  - Security
    - Private and Public key ciphers, digest routines
    - 3DES, Blowfish, SHA, AES finalists
  - Consumer
    - Multimedia, image processing, entertainment
    - JPEG, Dither, RGBA, MediaBench, DOOM
  - Office
    - Spell, Grep, Ghostscript Postscript Interpreter
  - Telecommunications
    - FFT, GSM, ADPCM



## ■ Initial study [PACS'00]

- Compared Wattch and Cai/Lim models
- Both built on SimpleScalar
- Examined scheduling optimizations

## ■ Results confirm further work needed

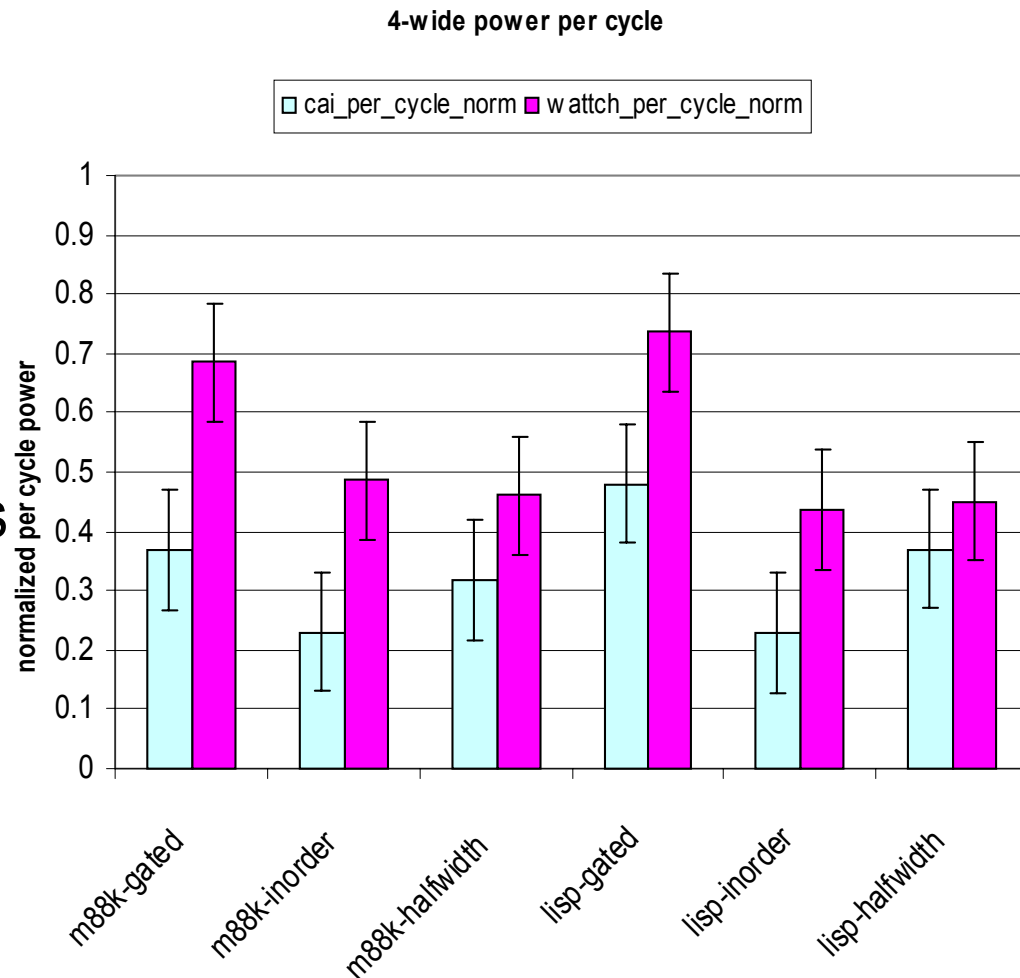
- Statistically significant differences
- Different assessments of benefits

## ■ Primary reasons for differences

- Different measured activities
- Different component power models
- Different effects due to reconfiguration
- Different technology scaling factors
- Different handling of clock power

## ■ Other deficiencies

- Cai/Lim component models are proprietary
- Wattch not sufficiently detailed
- Neither approach considers data dependencies
- Neither correlated real power at component level





- Based on physical analysis of standard microarchitectural components
  - Functional Unit: ALU, shifter, rotator, multiplier
  - Memories: caches, branch predictors, BTBs, ROBs
  - CAMs: TLBs, reservation stations, load/store scheduler
  - leveraging physical designs created in previous projects (MARM)
- Measurements
  - Parameterized dynamic power (DP)
    - parameterized by geometry and technology of component (size, width, etc...)
    - function of weighted Hamming distance of consecutive inputs (hd)
    - parameterized by clock/voltage gating model
  - Parameterized static power (SP)
    - parameterized by geometry and technology of component (size, width, etc...)
  - Total active/inactive cycles by component
  - Input Hamming distances by component (possibly sampled)
- $\text{Power} = \text{DP}(\text{hd}, \text{geom}) * \text{freq}_{\text{active}} + \text{SP}(\text{geom})$ 
  - Extended to peak power and di/dt noise
- Validated using whitebox (MARM) and blackbox (iPAQ) calibration



# Plans



<b>2000</b>	<b>H2</b>	<b>SA-1 processor core simulator Component-level power design</b>
<b>2001</b>	<b>H1</b>	<b>SA-1110 processor simulator Power model calibration &amp; integration</b>
	<b>H2</b>	<b>System-level simulator (iPAQ running Linux) Whitebox calibration of existing design (MARM)</b>
<b>2002</b>	<b>H1</b>	<b>Demonstration experiments with Power Analyzer Blackbox calibration of pocket computer (iPAQ)</b>



- Alpha release available today, includes
  - SimpleScalar/ARM instruction set simulation
  - SimpleScalar/ARM performance modeling infrastructure
  - First cut at StrongARM SA-1110, undergoing validation
  - SPEC'95, SPEC'2k pre-compiled benchmarks (no inputs due to copyright)
  - MiBench source, inputs, and pre-compiled benchmarks
  
- Requirements
  - CVS installed in your development environment
  - SSH installed in your development (release via SSH-based CVS checkout)
  - Rebel NetWinder workstation (only necessary to build ARM binaries)
    - Available from rebel.com
    - Cross-compilation environment available shortly
  
- E-mail Todd Austin ([austin@umich.edu](mailto:austin@umich.edu)) for CVS login and password

