PowerV*i*P:
SoC Power Estimation Framework at Transaction Level

Jan. 26, 2006

Ikhwan Lee et. al.
Corporate Computer-Aided Engineering
Semiconductor Business
Samsung Electronics Co., Ltd.

Outline

- Introduction
- Component Power Modeling
  - ARM926EJS processor
  - AMBA AXI bus fabrics
  - Custom IP blocks
- PowerV*i*P
- Concluding Remarks

Outline

- Introduction
- Component Power Modeling
  - ARM926EJS processor
  - AMBA AXI bus fabrics
  - Custom IP blocks
- PowerV*i*P
- Concluding Remarks

Low Power Design Solutions

Software	System/ Architecture	Logic	Circuit/ Physical	Device
Power Management	HW / SW Partitioning	Path Balancing	Multi-Vth	Dual Gate
Dynamic Volt. Scaling	Memory Optimization	Gate Sizing	VTCMOS	FinFET / 3D
Instruction-level Opt.	Adaptive Volt. Scaling	Clock Gating	Voltage Island	SOI
Control-Data Transform.	Parallelism	Technology Mapping	Power gating	Multi-Tox

“You can’t manage it until you can estimate it!”

Power Estimation	Power Estimation	Power Estimation	Power Estimation	Power Estimation

Why System Level Power Estimation?

- **Advantages**
  - Larger opportunities for power reduction
    - x10~x20 as compared to logic level
  - Faster estimation
    - Enables thorough design space exploration
  - Power profile given in the system context
    - Prevents from falling into a local optimum

The Requirements and Problems

- **Requirements**
  - System level simulation platform
    - V*i*P (Virtual Platform)
  - Power models of system components
- **Problems**
  - Diversity of components (power characteristics)
  - Trade-off among the below three factors
    - Simulation speed → to maximize
    - Estimation accuracy → to maximize
    - Modeling effort → to minimize

Observation in a mobile SoC family

Frequently reused IPs

- Processor: ARM, StarCore
- Bus: AXI, AHB
- Off-chip memory: DDR, Flash
- Memory controller
- Peripheral: GPIO, USB
- Image filters

Platform-based design enables maximum reusability

Continuously evolving IPs

- Video IP (MPEG4, H.264)

3rd party IPs

- 3D graphics

Model once, reuse many times

Our Contributions

- Identification of IP classes
- Power models for major SoC components
  - Speed
  - Accuracy
  - Modeling effort
- Provide cycle-accurate power profile in the system context

Outline

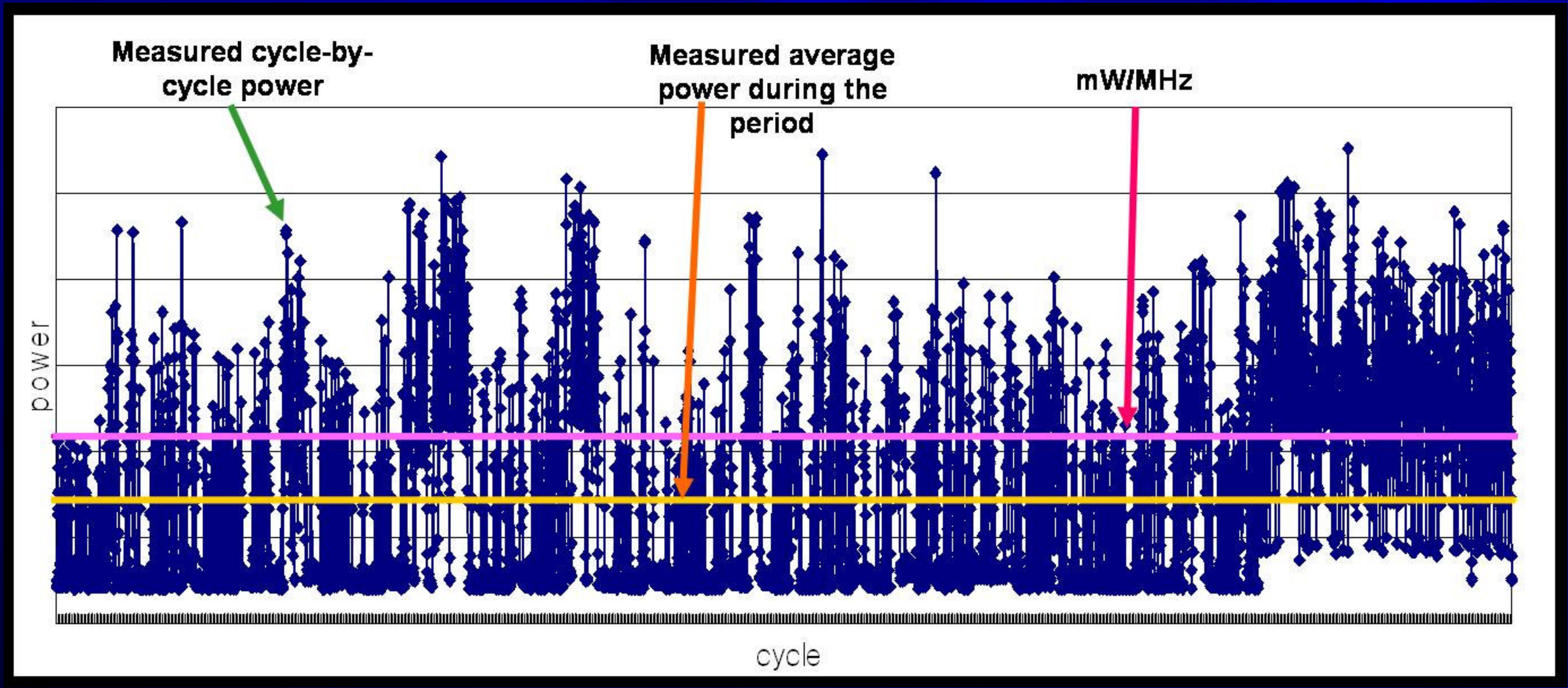
- Introduction
- **Component Power Modeling**
  - ARM926EJS processor
  - AMBA AXI bus fabrics
  - Custom IP blocks
- PowerV*i*P
- Concluding Remarks

Outline

- Introduction
- **Component Power Modeling**
  - **ARM926EJS processor**
  - AMBA AXI bus fabrics
  - Custom IP blocks
- PowerV*i*P
- Concluding Remarks

ARM926EJS Power Profile

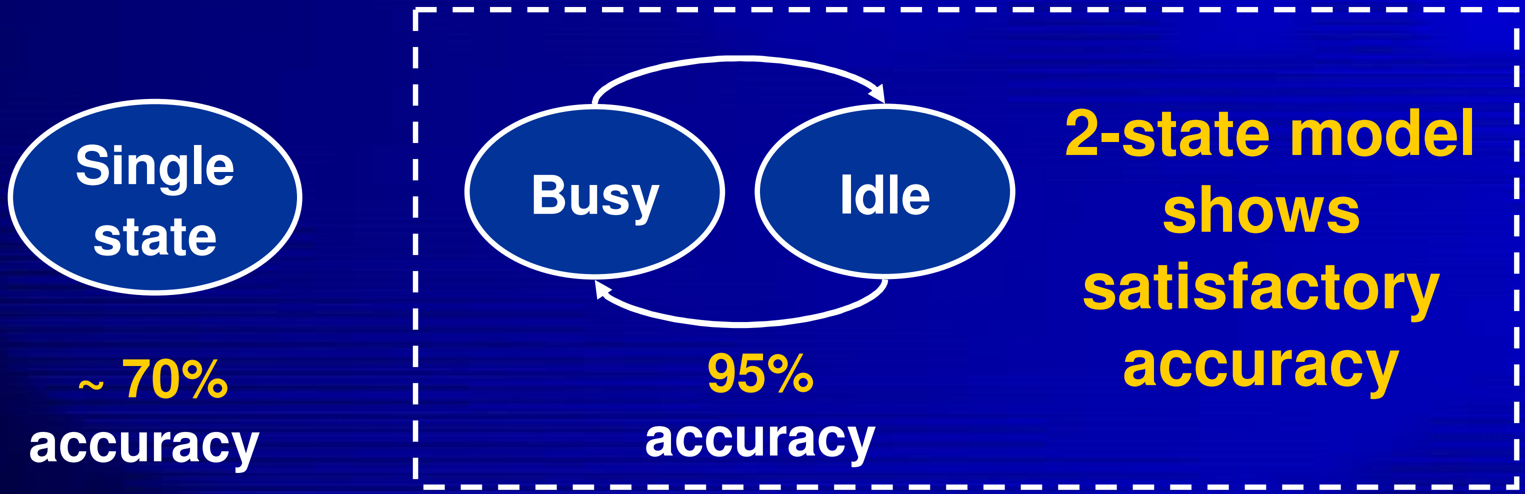
- **Simple mW/MHz model does not reflect power phase transitions during the course of a program execution**



Defining Power States (1)

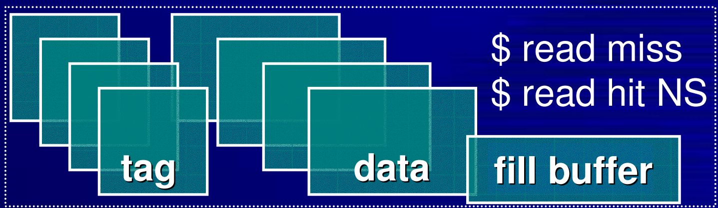
- Separate core and cache power states
  - Cache size needs to be configurable
  - Cache power shows large variation (3~60% of total power)

- Core power states

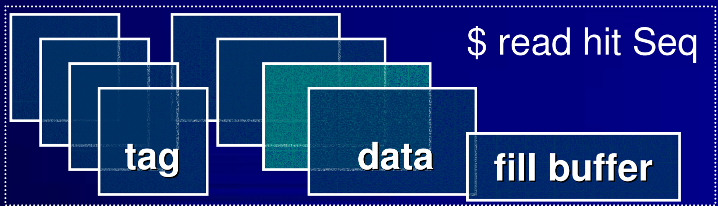


Defining Power States (2)

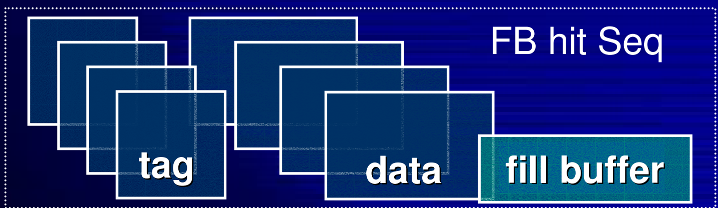
- **Cache power states**
  - **Activity based coarse-grained power model**
  - **Differentiates non-sequential, sequential, and fill buffer accesses**



(a) Non-sequential access



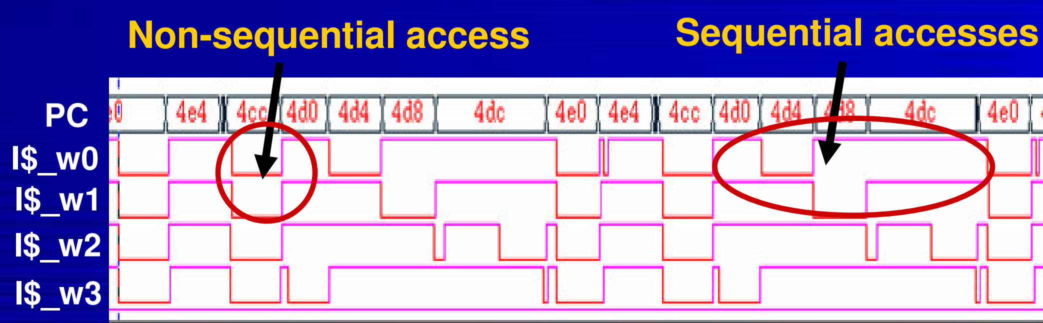
(b) Sequential access



(c) Fill buffer access

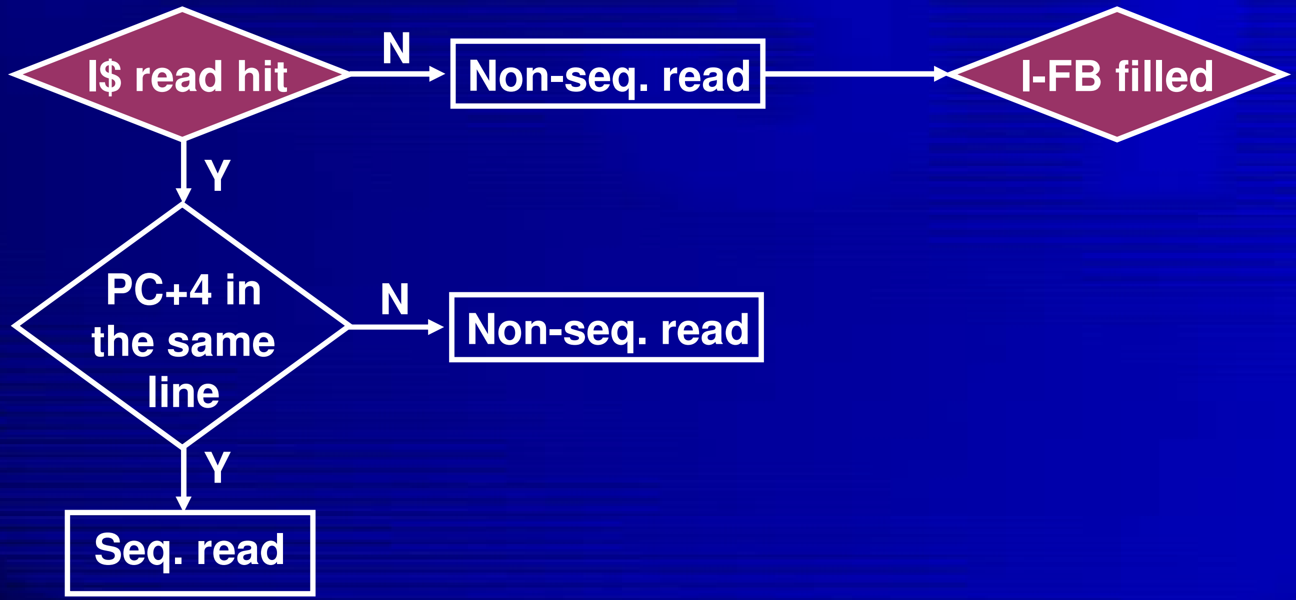
```
for (i=0;i<10;i++)
    *(word_wr+i)=i+1;
```

```
4CC:  ADD    r0,r4,#1
4D0:  STR    r0,[r7,r4,LSL #2]
4D4:  ADD    r4,r4,#1
4D8:  CMP    r4,#0xa
4DC:  BLT    0x4cc
4E0:  MOV    r4,#0
4E4:  B      0x500
```



Power Annotation

- Core states are visible in the ARM926EJS ISS (instruction set simulator)
- Cache states need to be inferred from transaction level activities

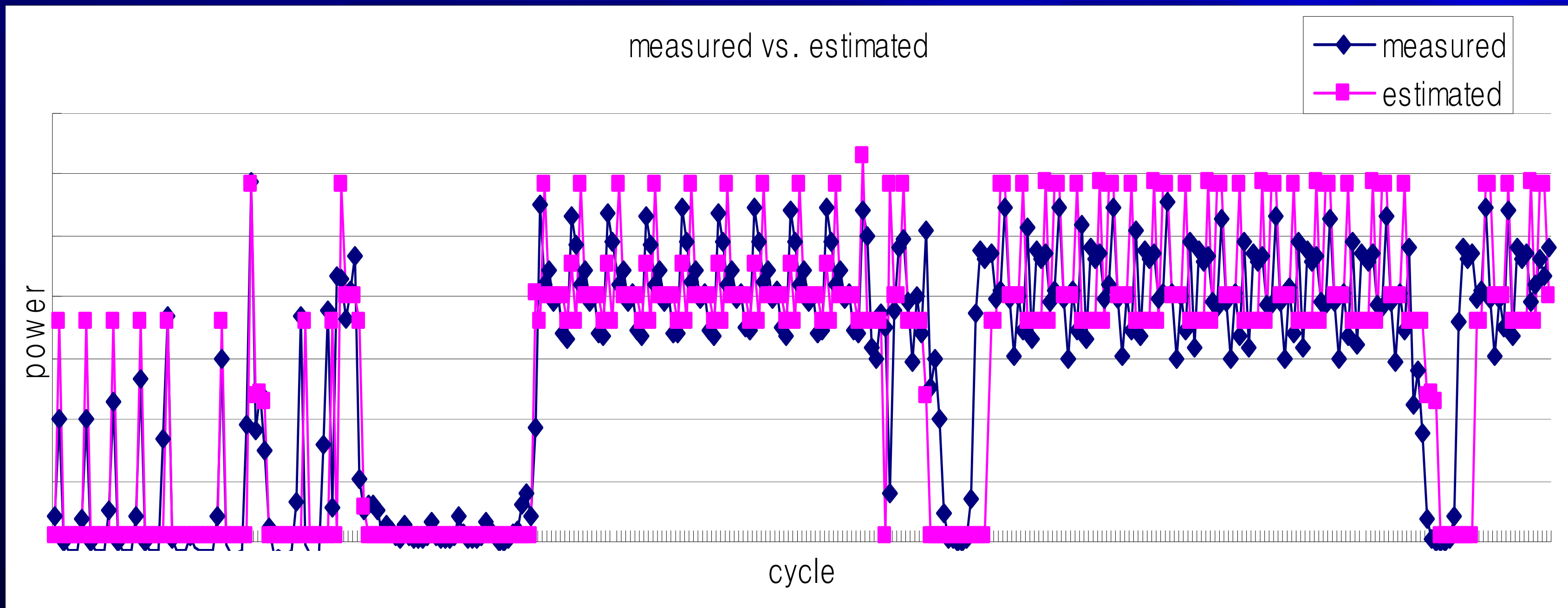


Estimation Accuracy vs. Gate-level

- **Average estimation accuracy (< 93%)**

dhrystone	cav_detect	adpcm	FFT	h264 enc
97.1%	97.3%	98.2%	96.6%	93.1%

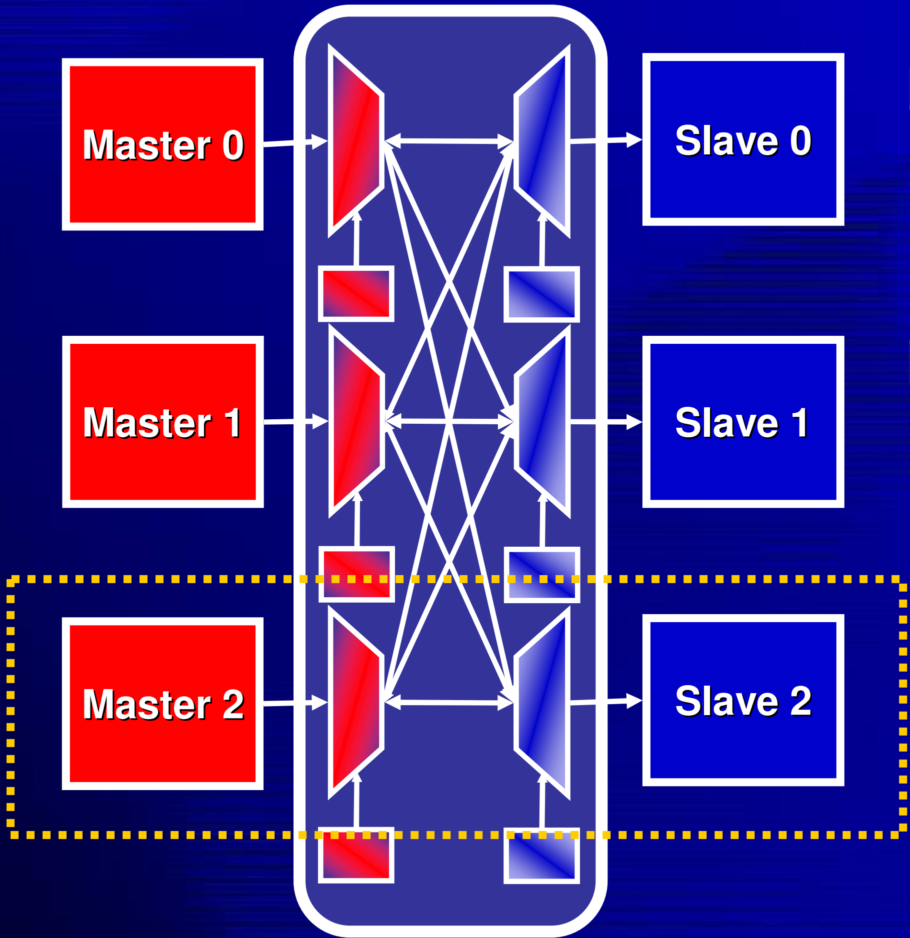
- **Cycle-by-cycle power profile**



Outline

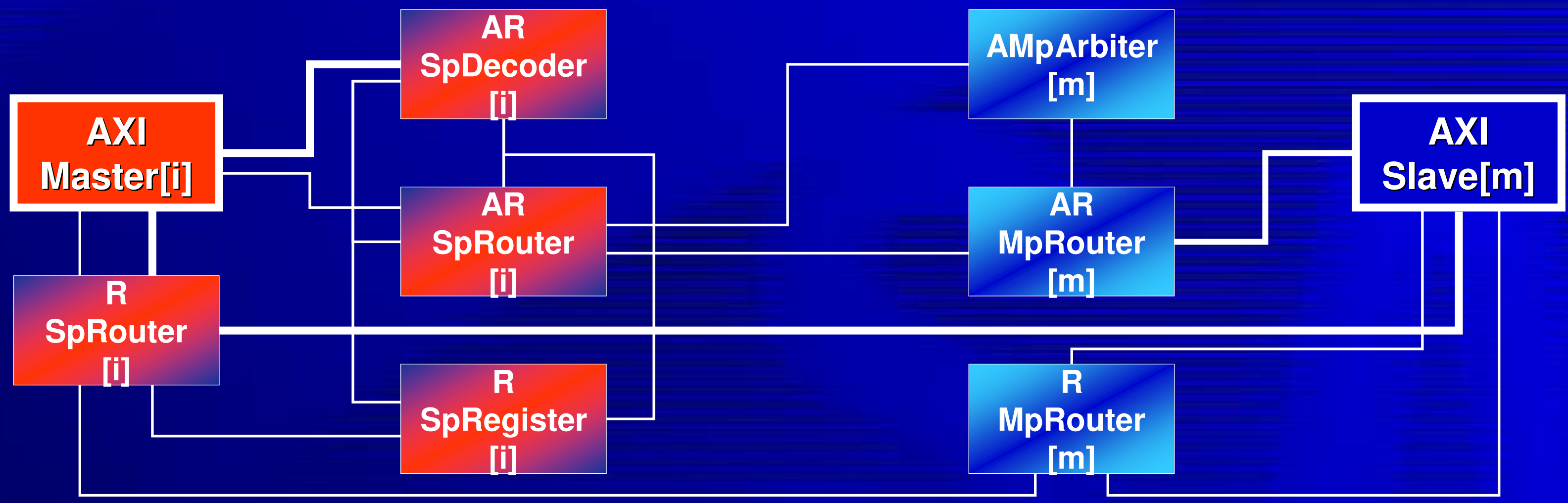
- Introduction
- **Component Power Modeling**
  - ARM926EJS processor
  - **AMBA AXI bus fabrics**
  - Custom IP blocks
- PowerV*i*P
- Concluding Remarks

PL300 AXI Interconnect



- Full crossbar architecture
- Power characterization when only one master and one slave are active

Component-based Power Model: PL300



- Characterize each component
  - For each basic state, find out which component is active and how much power it consumes
- Compose the basic model
  - For each cycle, add the power consumption of all active components
- Linear regression model
  - Consider the coupling effect

Linear Regression Model

- **Each AXI sub-component has its own linear regression model.**

$$E_{total} = E_{est} + n\_AR * E_{br\_RD} + n\_AW * E_{br\_WT} + \frac{n\_RD * E_{cyc\_RD} + n\_WT * E_{cyc\_WT}}{n\_RD + n\_WT}$$

$$E_{est} = a_1 E_{comp} + a_0$$

$$a_1 = \frac{n\_RD * a_{1\_RD} + n\_WT * a_{1\_WT}}{n\_RD + n\_WT}$$

$$a_0 = \frac{n\_RD * a_{0\_RD} + n\_WT * a_{0\_WT}}{n\_RD + n\_WT}$$

Coupling effects

Perspective of our bus power TLM

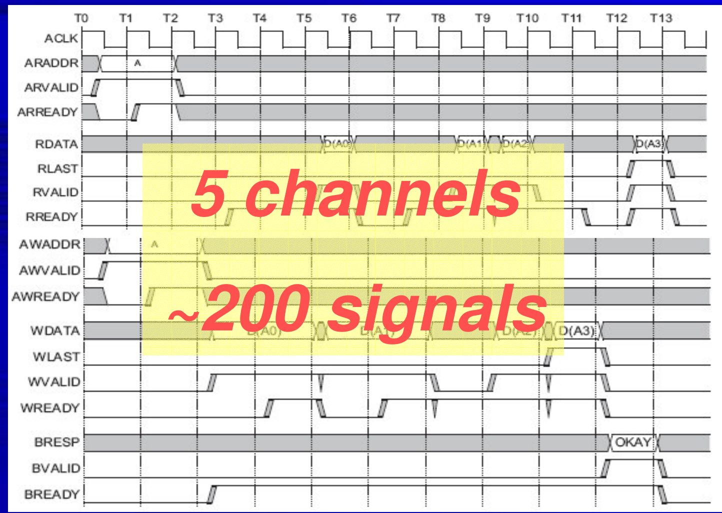
AXI Master → **AXI Slave**

Write Address/Control (52~56)

- AWID[3:0] write addr ID (0~4 bits)
- AWADDR[31:0] write addr ID
- AWLEN[3:0] burst length
- AWSIZE[2:0] burst size
- AWBURST[1:0] burst type
- AWLOCK[1:0] lock info
- AWCACHE[3:0] cache type
- AWPROT[2:0] protection type
- AWVALID write address valid
- AWREADY write address ready

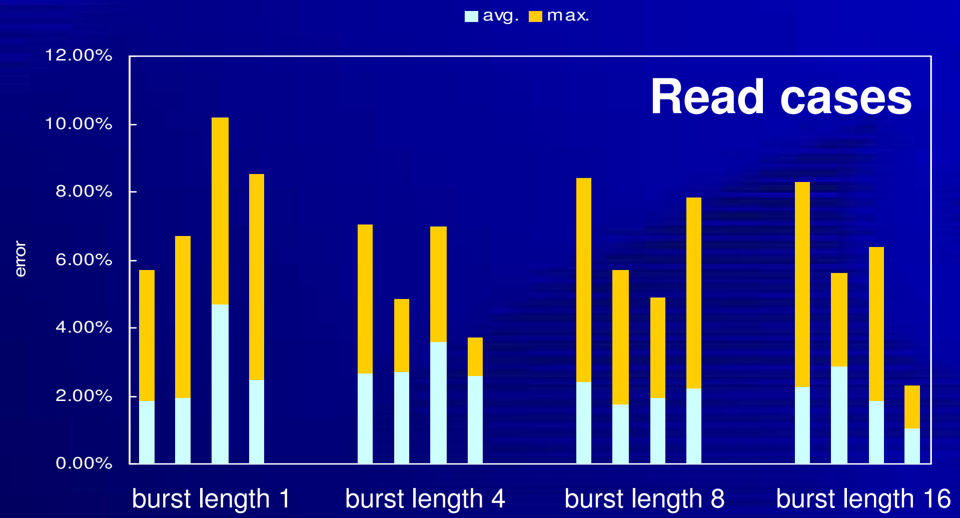
Write data (39~43)

- WID[3:0] write ID tag, AWID = WID (0~4 bits)
- WDATA[31:0] write data
- WSTRB[3:0] write strobes
- WLAST write last
- WVALID write valid
- WREADY write ready



READ (AR,R)	AR	ID	R	R	ID	ID	R
WRITE (AW,W,B)	ID	AW	W	ID	W	ID	B

Estimation Accuracy vs. Gate-level



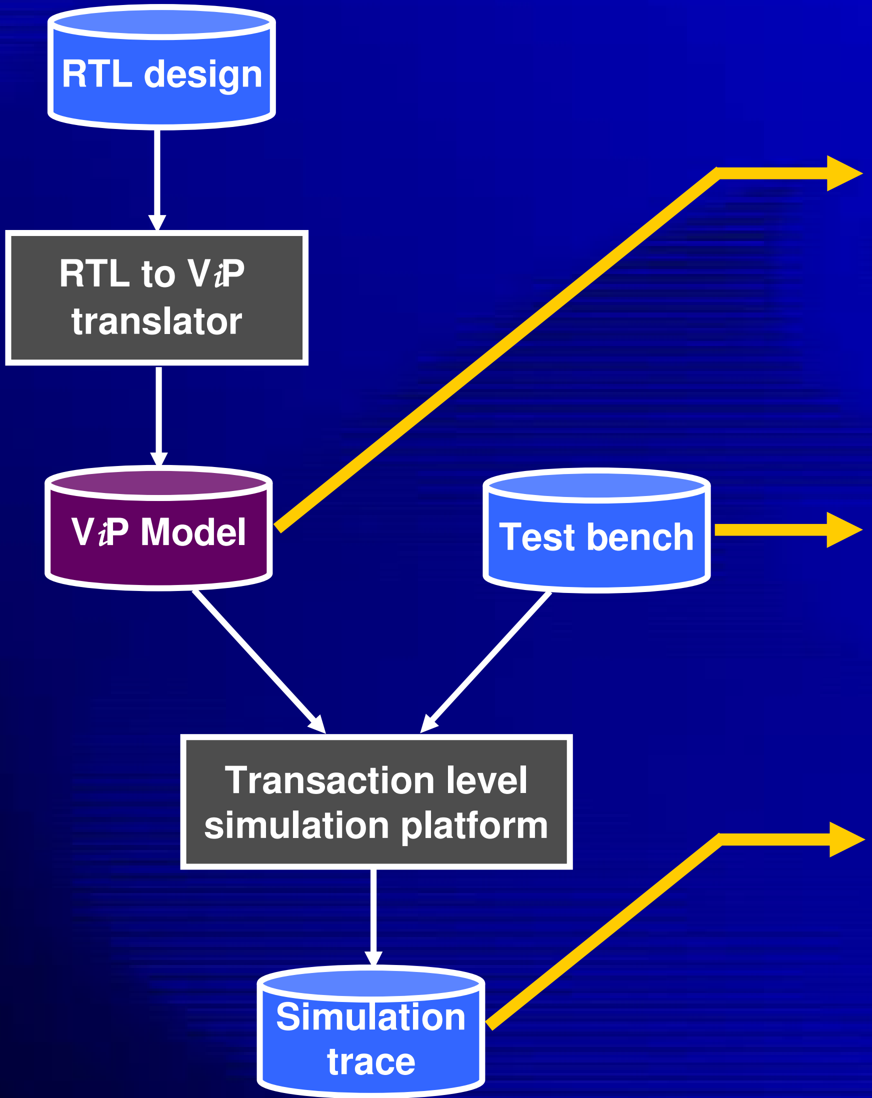
	# master	# slave	ID width	data width
conf. 1	4	4	0	32
conf. 2	4	4	4	32
conf. 3	6	2	4	32
conf. 4	7	3	5	64

Max < ~10% estimation error

Outline

- Introduction
- Component Power Modeling
  - ARM926EJS processor
  - AMBA AXI bus fabrics
  - Custom IP blocks
- PowerV*i*P
- Concluding Remarks

V*i*P Model Generation and Simulation



Virtually no effort is needed for building cycle-accurate V*i*P models

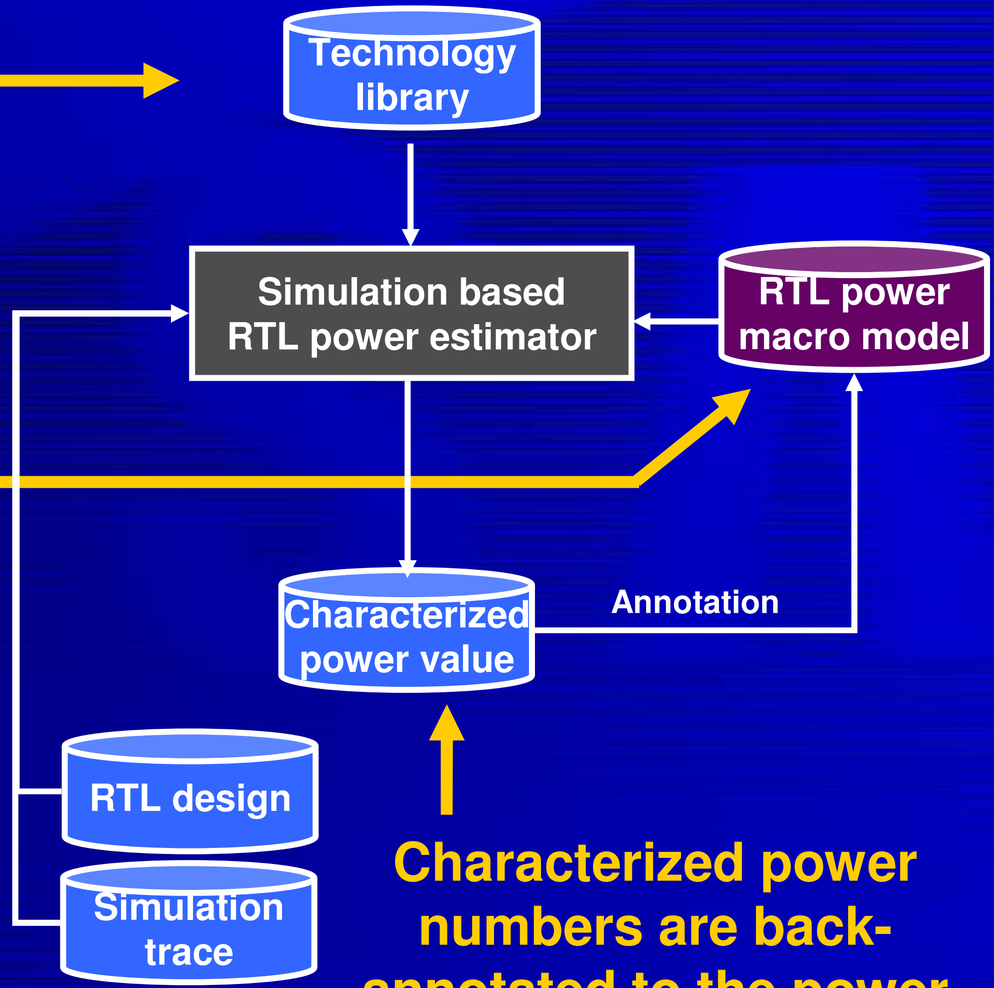
Easy-to-prepare system level simulation test benches

Register transfer level simulation traces can be acquired (VCD format)

RTL Power Estimation & Characterization

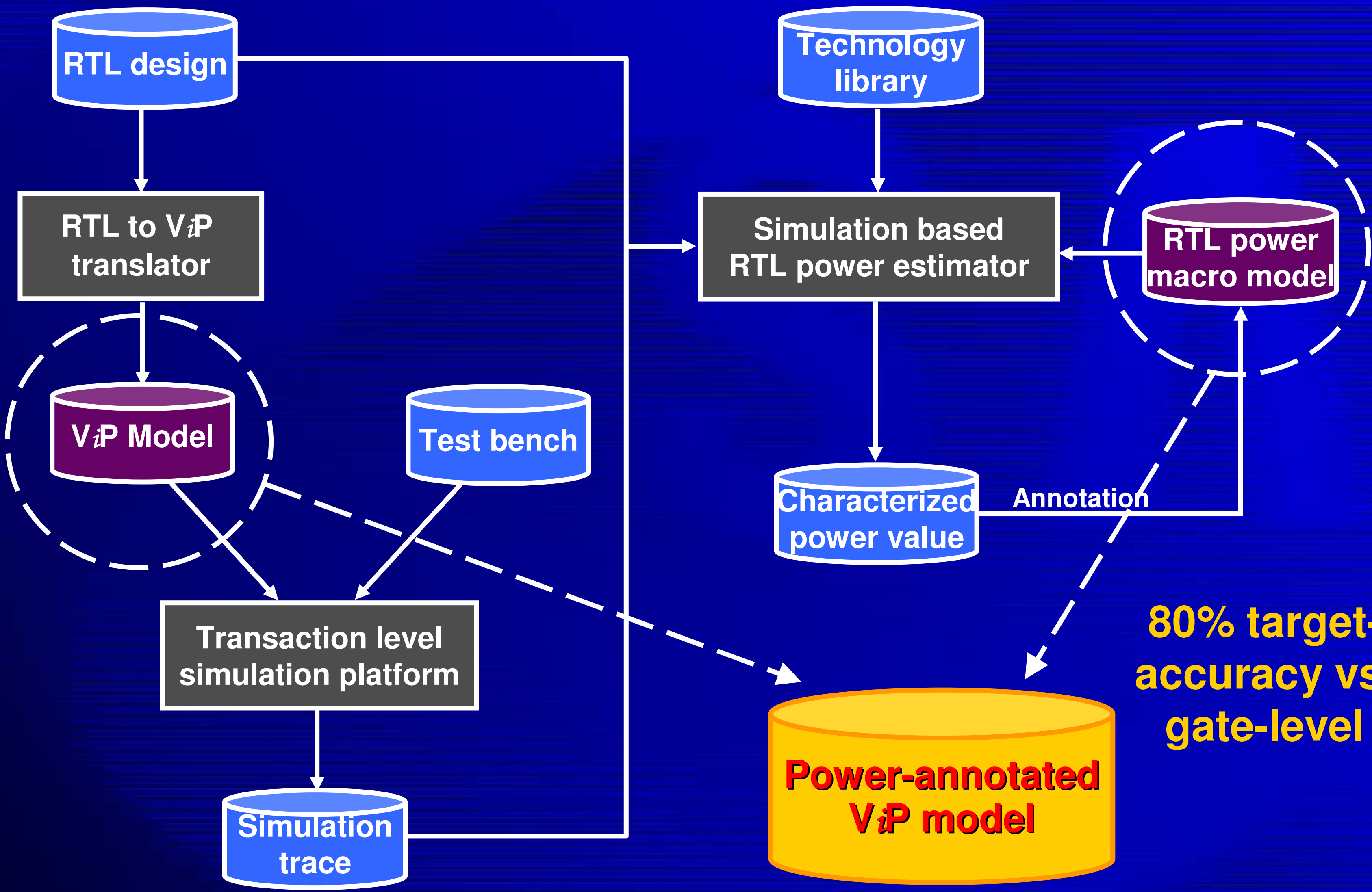
Process-specific libraries (130G, 90LP, etc.)

Power-representative FSM must be manually extracted from the RTL design



Characterized power numbers are back-annotated to the power macro model

Power Modeling of Custom IP Blocks

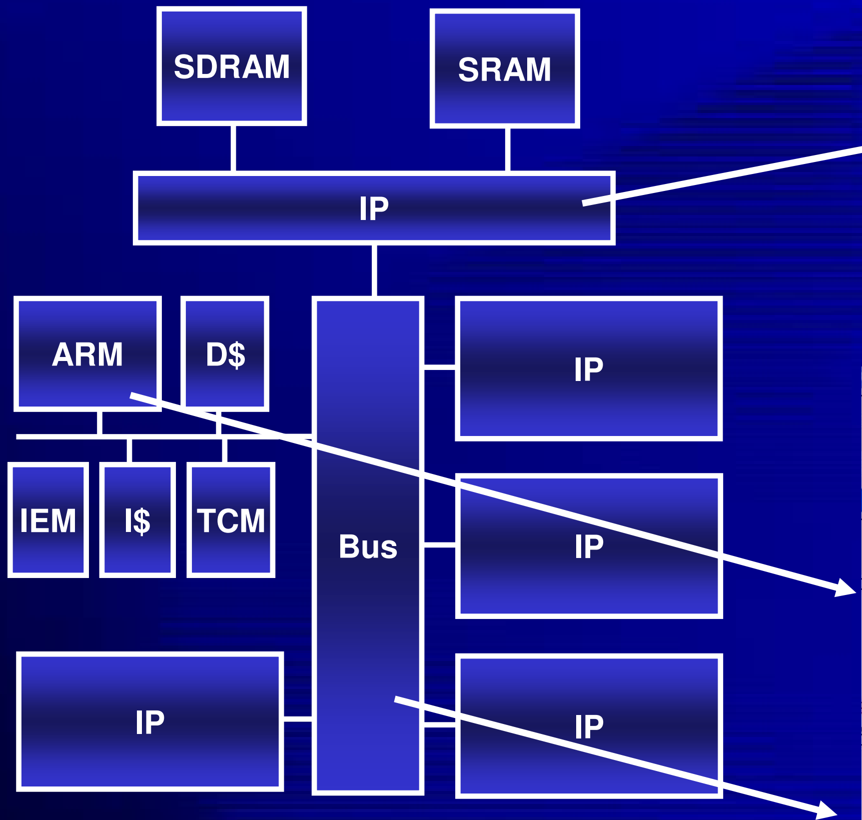


80% target-accuracy vs. gate-level

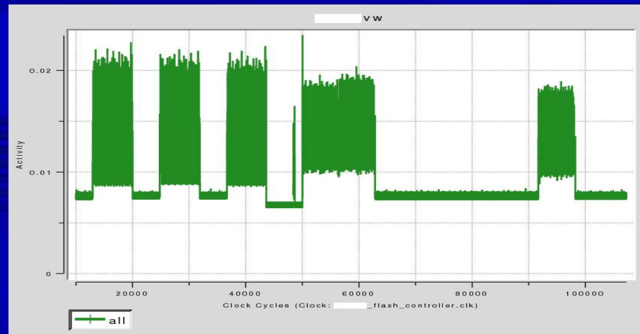
Outline

- Introduction
- Component Power Modeling
  - ARM926EJS processor
  - AMBA AXI bus fabrics
  - Custom IP blocks
- PowerV*i*P
- Concluding Remarks

Integration of Component Power Models



Custom IP power will be integrated into V*i*P



Processor and bus logic power: Integrated into V*i*P

Application of PowerV*i*P

- Peak power analysis
  - We can find realistic test patterns to avoid "over-design" of power grid
- Low power bus architecture exploration
- Early development of power management software
- Software code optimization for low power

Outline

- Introduction
- Component Power Modeling
  - ARM926EJS processor
  - AMBA AXI bus fabrics
  - Custom IP blocks
- PowerV*i*P
- Concluding Remarks

Concluding Remarks

- **Development of component power models**
  - **93% accuracy for ARM926EJS**
  - **95% accuracy for AXI bus**
  - **80% target-accuracy for custom IP blocks**
- **Integration into single simulation platform**
- **Cycle-accurate power profile of each component is shown**
- **PowerV*i*P can be used in variety of application**