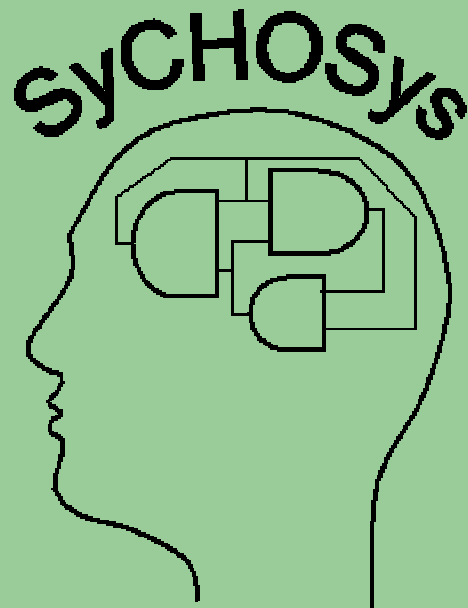


SyCHOSys

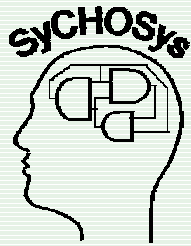
Synchronous Circuit Hardware Orchestration System



Ronny Krashinsky
Seongmoo Heo
Michael Zhang
Krste Asanovic

MIT Laboratory for Computer Science

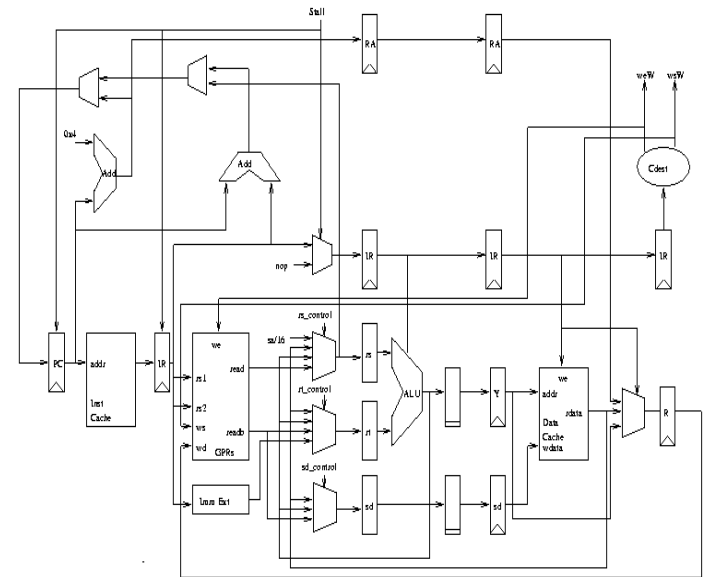
www.cag.lcs.mit.edu/scale
ronny@mit.edu



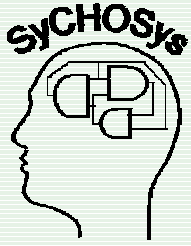
Motivation

Given a proposed processor architecture, we want to:

- Simulate performance (cycle count)
- Determine energy usage (Joules)
- Investigate SW, compiler, and architecture changes



Existing simulators – Prohibitively slow or inaccurate



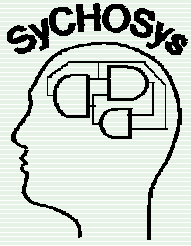
SyCHOSys

SyCHOSys generates compiled cycle simulators

Can optionally track energy usage:

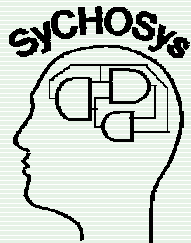
- Exploits low power microprocessor design domain to obtain accurate transition-sensitive energy models
- Factors out common transition counts
- Uses fast bit-parallel transition counting
- 7 orders of magnitude faster than SPICE (7% error)
- 5 orders of magnitude faster than PowerMill

SyCHOSys can accurately simulate the energy usage of a CPU circuit at speeds on the order of a billion cycles per day

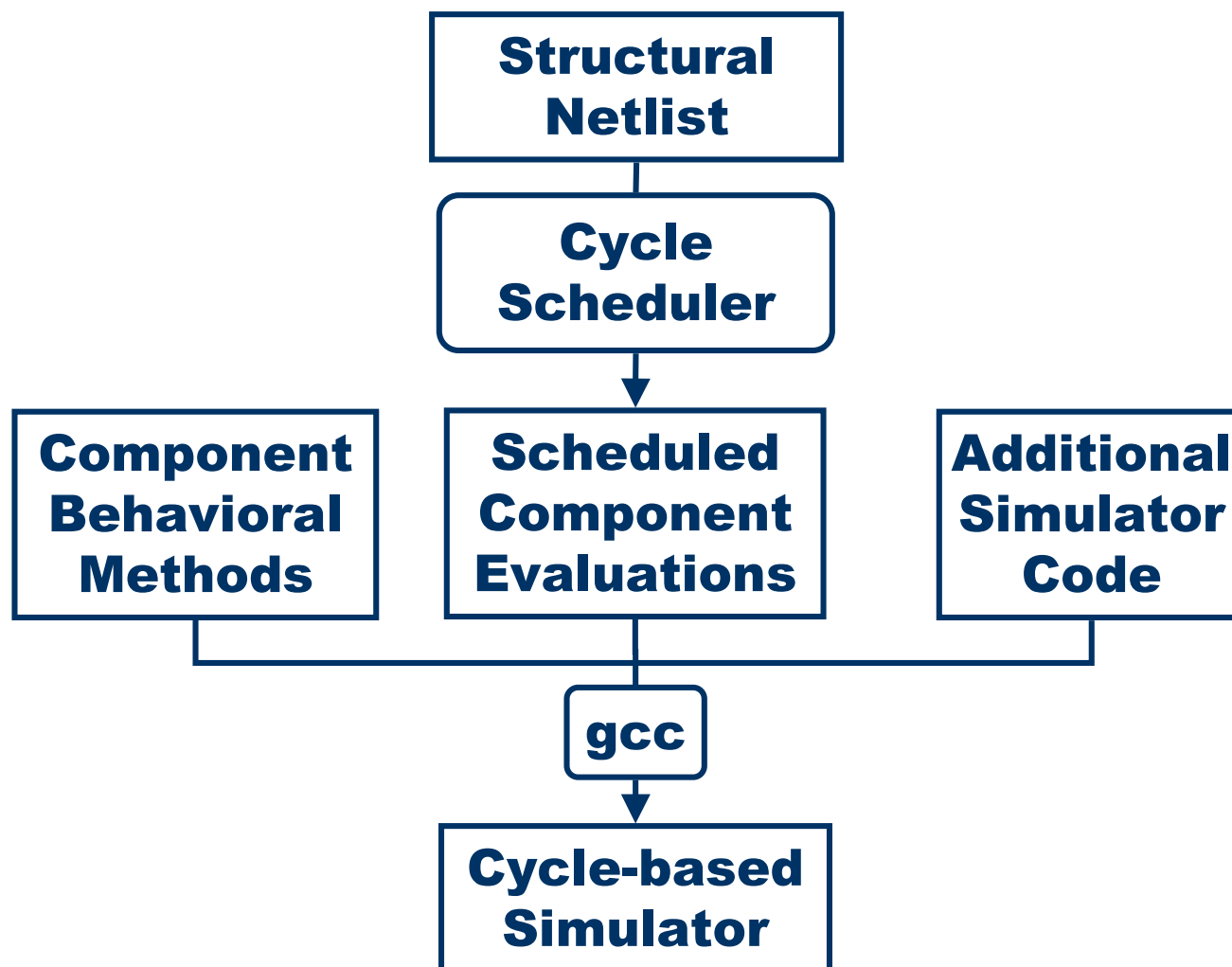


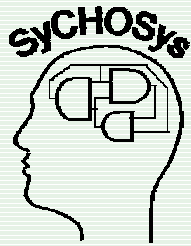
Overview of talk

- SyCHOSys Framework
- Microprocessor Energy Modeling
- Energy Simulation in SyCHOSys
- Results
- Status & Future Work

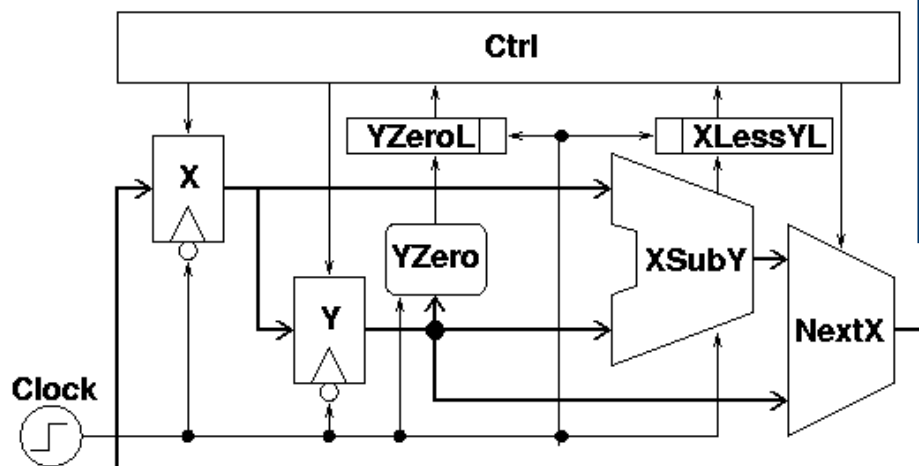
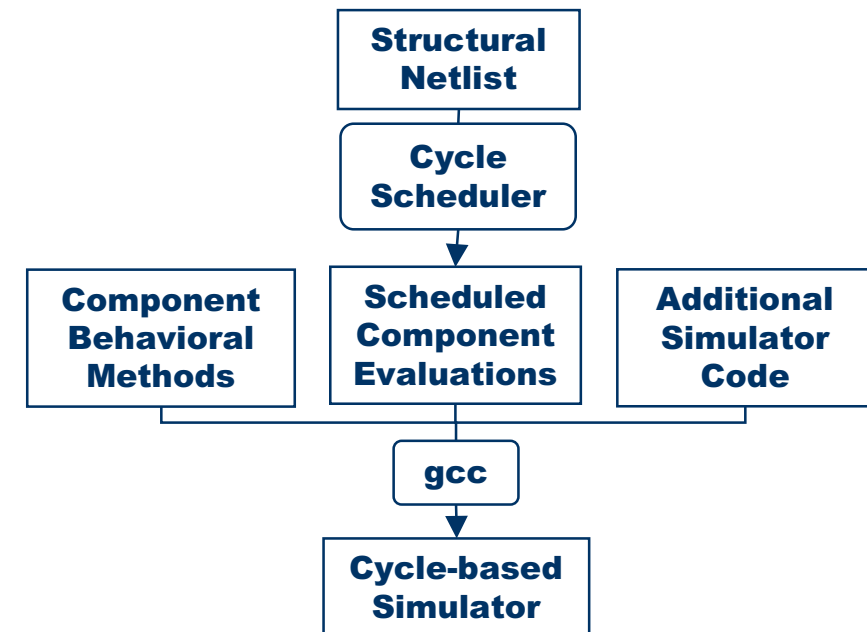


SyCHOSys Framework

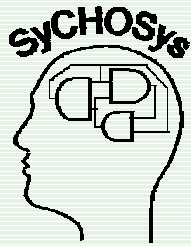




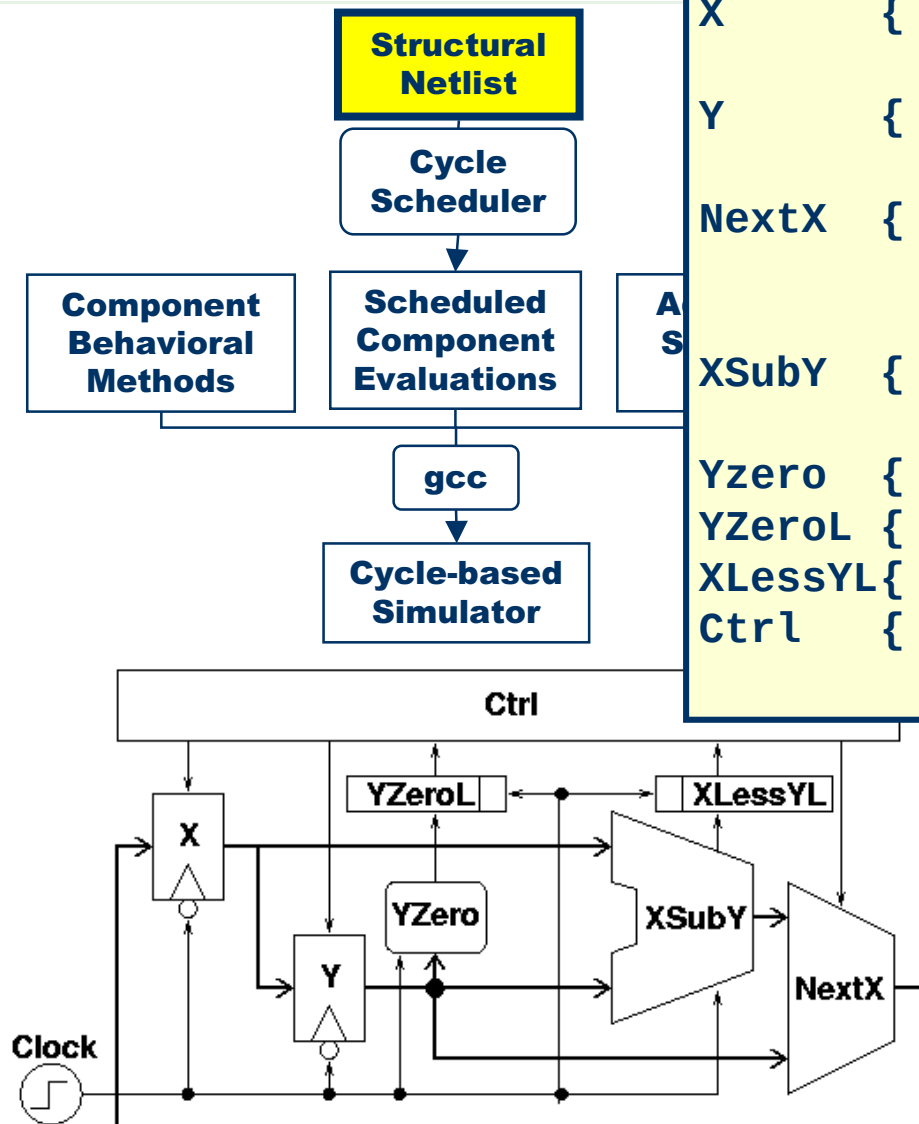
SyCHOSys Framework



```
GCD(x, y) {  
    if (x < y)      return GCD(y, x);  
    else if (y!=0) return GCD(x-y, y);  
    else           return x;  
}
```

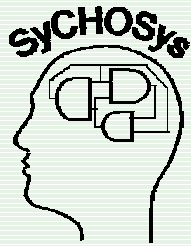


SyCHOSys Framework

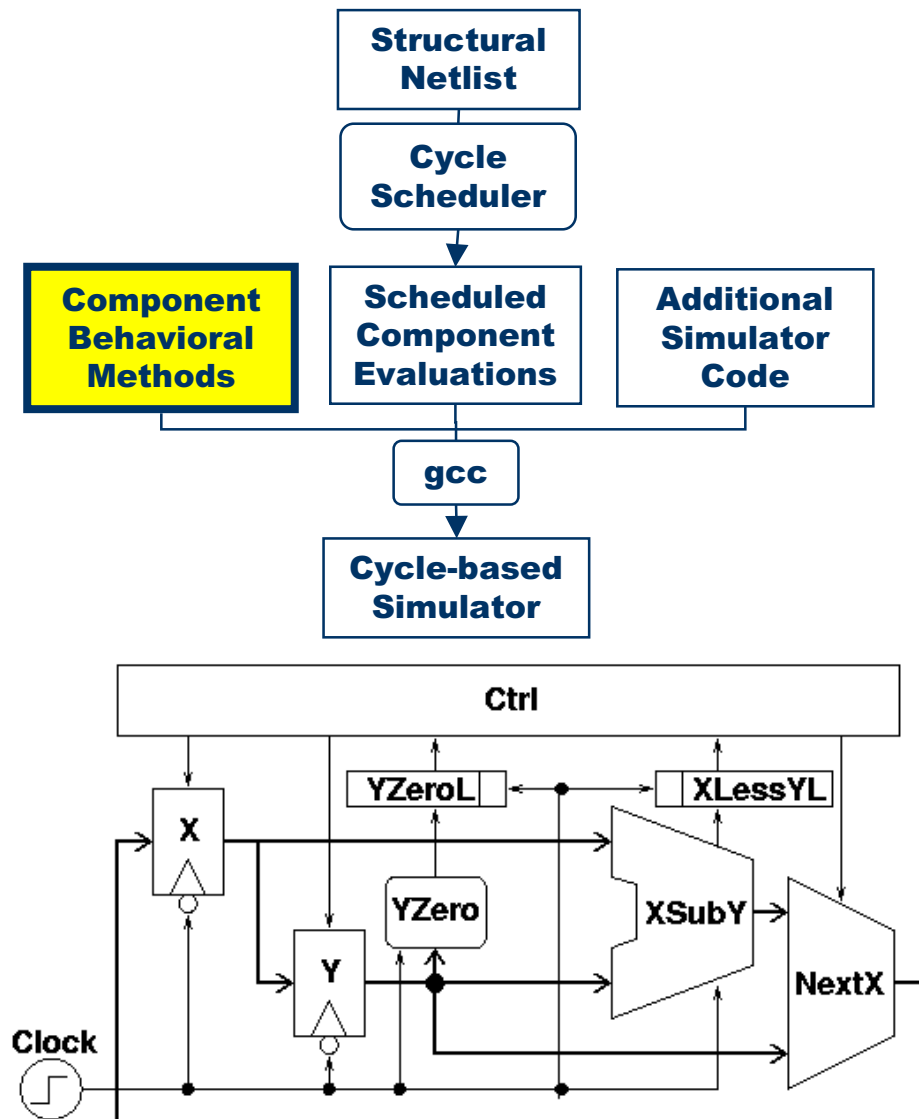


```

X      { N-CLK   FF_En<32> } (NextX.out,
                               Ctrl.Xen);
Y      { N-CLK   FF_En<32> } (X.out,
                               Ctrl.Yen);
NextX  {           Mux2<32> } (Y.out,
                               XSubY.out,
                               Ctrl.XMuxSel);
XSubY   { H-DYNAM Sub<32> } (X.out,
                              Y.out);
Yzero   { H-DYNAM Zero<32> } (Y.out);
YZeroL  { H-LATCH Latch<1> } (YZero.out);
XLessYL { H-LATCH Latch<1> } (XSubY.signbit);
Ctrl    {           GCDCtrl } (XLessYL.out,
                              YZeroL.out);
  
```

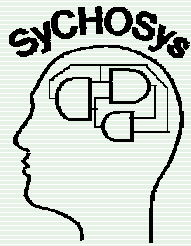


SyCHOSys Framework

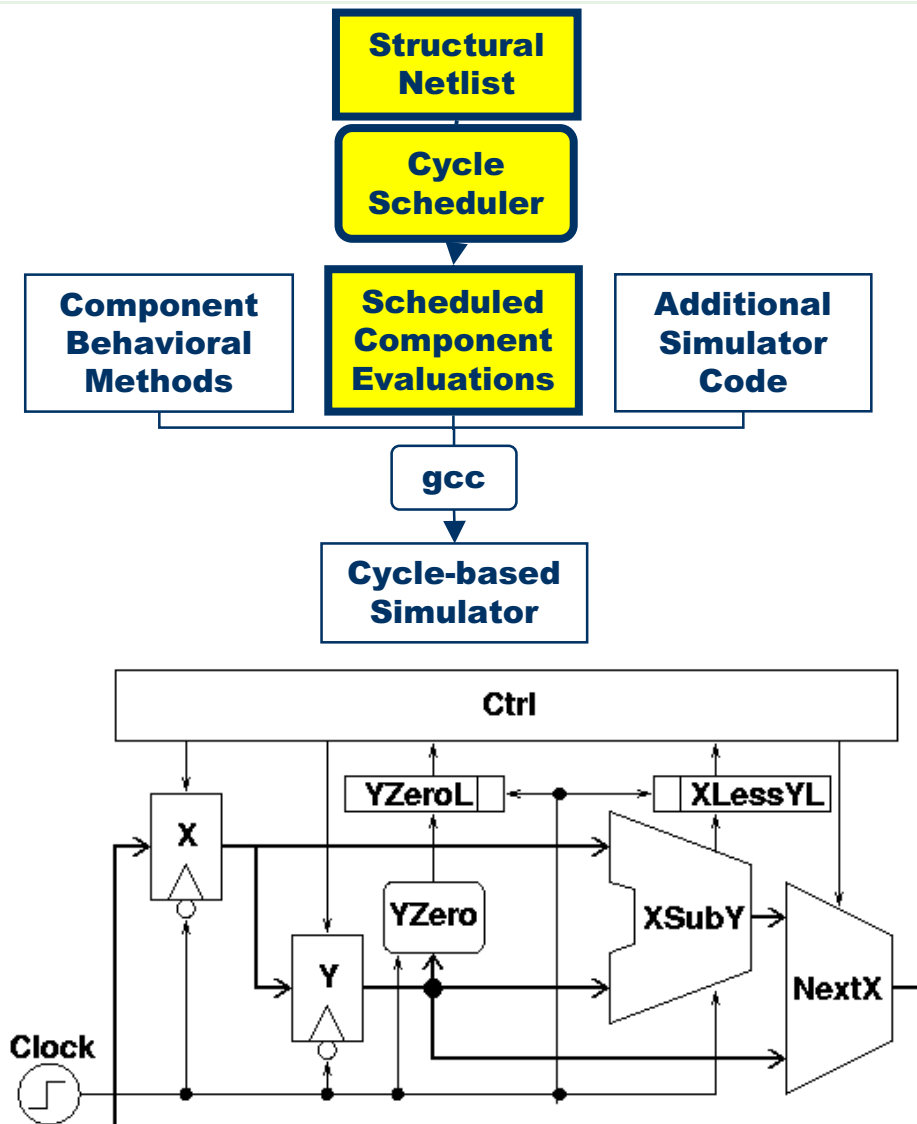


```

template<int bits>
class Mux2 {
    inline void Evaluate(
        BitVec<bits> input0,
        BitVec<bits> input1,
        BitVec<1> select) {
        if (select) out = input1;
        else      out = input0;
    }
    BitVec<bits> out;
}
    
```

SyCHOSys Framework

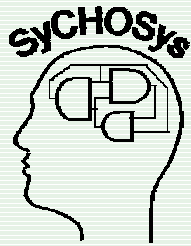


```
GCD::clock_rising() {}
```

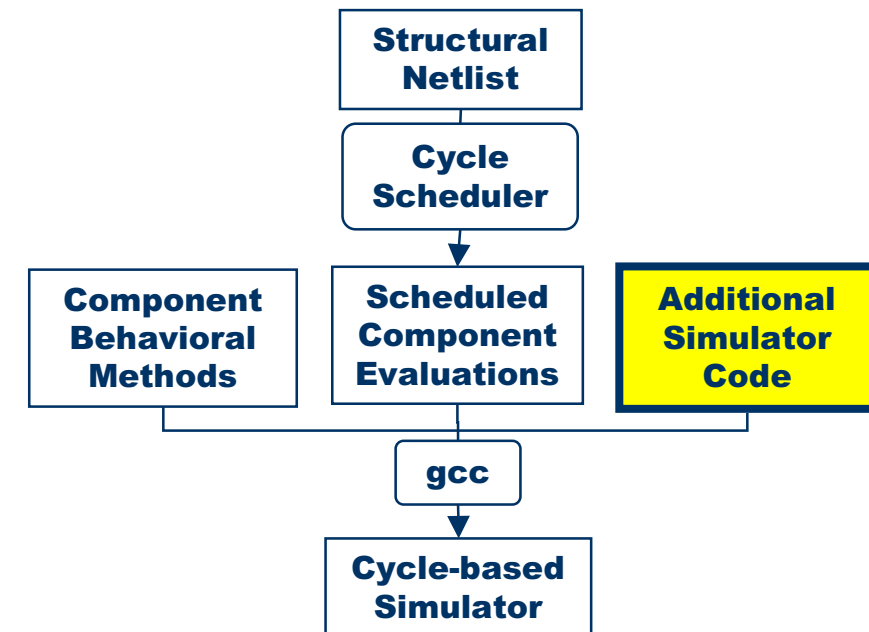
```
GCD::clock_high() {
    YZero.Evaluate(Y.out);
    YZeroL.Evaluate(YZero.out);
    XSubY.Evaluate(X.out, Y.out);
    XLessYL.Evaluate(XSubY.signbit);
    Ctrl.Evaluate(XLessYL.out,
                  YZeroL.out);
    NextX.Evaluate(Y.out, XSubY.out,
                  Ctrl.XMuxSel);
}
```

```
GCD::clock_falling() {
    Y.Evaluate(X.out, Ctrl.Yen);
    X.Evaluate(NextX.out, Ctrl.Xen);
}
```

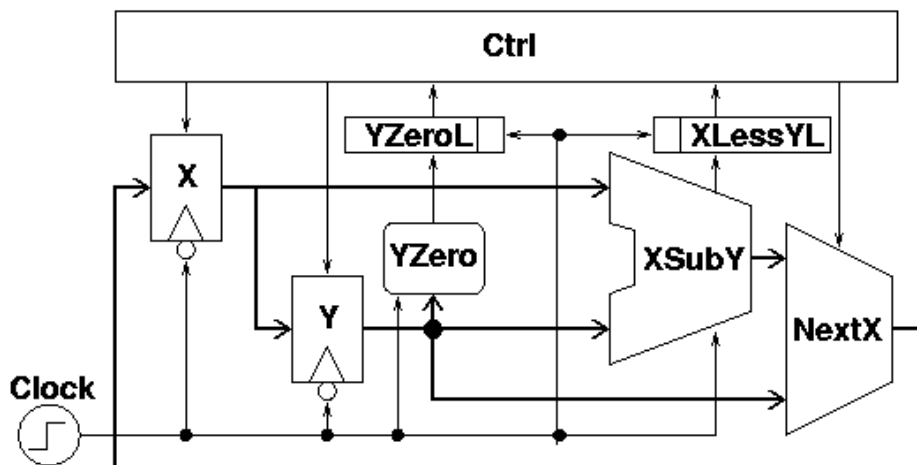
```
GCD::clock_low() {
    YZero.Precharge();
    XSubY.Precharge();
    NextX.Evaluate(Y.out, XSubY.out,
                  Ctrl.XMuxSel);
}
```

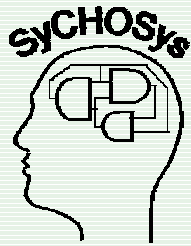


SyCHOSys Framework

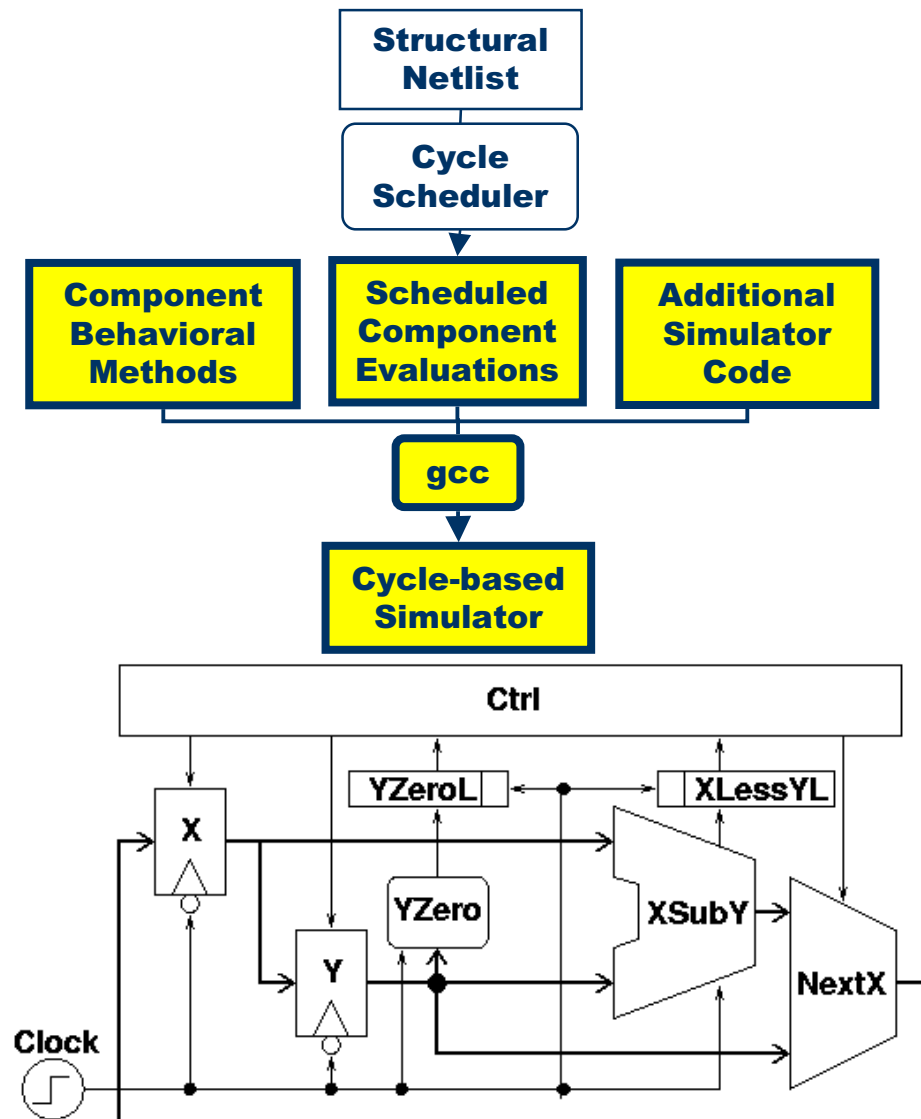


```
void gcd_clock_tick() {  
    gcd->clock_rising();  
    gcd->clock_high();  
    gcd->clock_falling();  
    gcd->clock_low();  
}
```

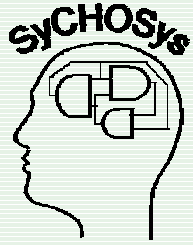




SyCHOSys Framework



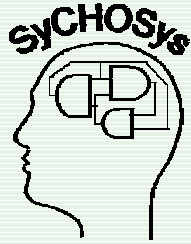
- Optimizing compiler
- Component evaluation calls are inlined



Energy Modeling

Power consumption in digital CMOS:

- Dynamic Switching
- Short Circuit Current
- Leakage Current



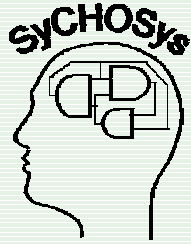
Energy Modeling

Power consumption in digital CMOS:

- Dynamic Switching: around 90%

$$a \cdot C_{load} \cdot V_{SWING} \cdot V_{DD} \cdot f$$

a	switching activity	data dependent
C_{load}	load capacitance	varies dynamically
V_{SWING}	voltage swing	fixed
V_{DD}	supply voltage	fixed
f	clock frequency	fixed



Energy Modeling

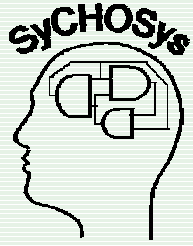
Power consumption in digital CMOS:

- Dynamic Switching: around 90%

$$a \cdot C_{load} \cdot V_{SWING} \cdot V_{DD} \cdot f$$

a	switching activity	data dependent
C_{load}	load capacitance	varies dynamically
V_{SWING}	voltage swing	fixed
V_{DD}	supply voltage	fixed
f	clock frequency	fixed

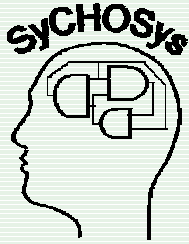
We simplify our task by taking advantage of our restricted domain of well designed low power microprocessors



Microprocessor Energy

Energy usage in a microprocessor:

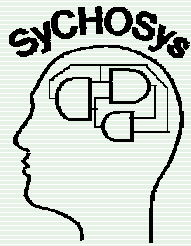
- Memory arrays
- Datapaths
- Control



Microprocessor Energy

Energy usage in a microprocessor:

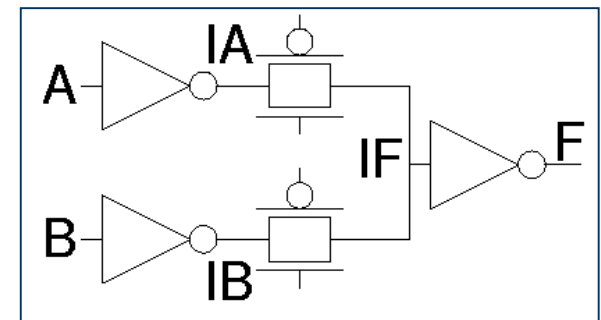
- **Memory arrays**
 - Datapaths
 - Control
-
- Extremely regular
 - Calibrate models with several test cases:
 - Accounts for partial voltage swings, effective capacitance values, etc.
 - Estimate energy based on cycle by cycle address and data trace (3% error)



Microprocessor Energy

Energy usage in a microprocessor:

- Memory arrays
- **Datapaths**
- Control

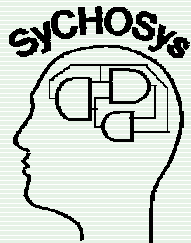


Determine a and C_{load} for every node

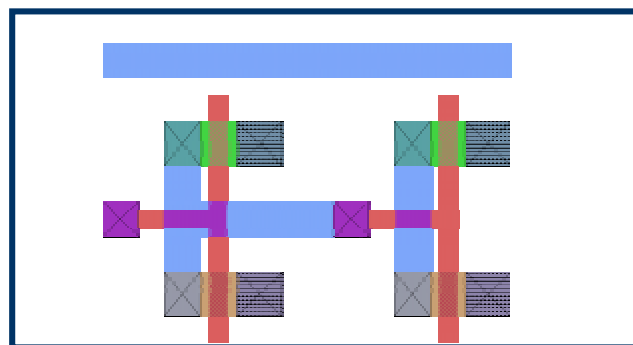
- Effective C_{load} is calculated statically
- a is determined based on simulation statistics

Optimizations for determining switching activity:

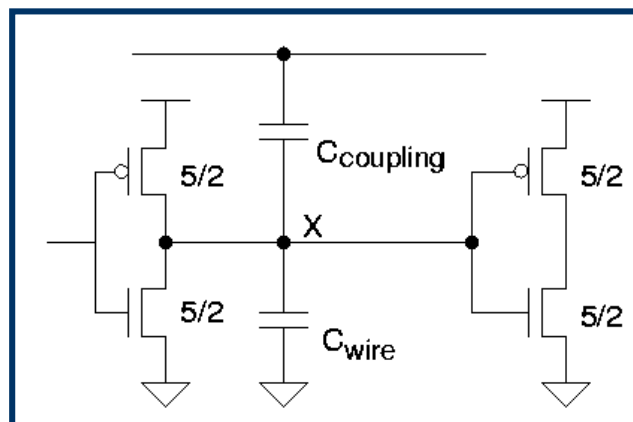
- Factor out common transition counts
- Fast bit-parallel transition counting



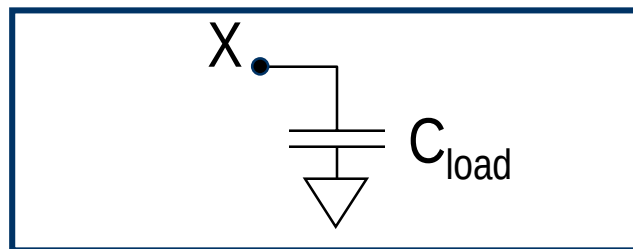
Effective Load Capacitance



SPACE 2D
extractor

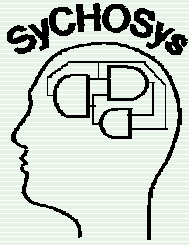


MergeCap



Gate and Drain Capacitance
Models

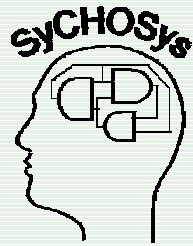
- Characterized using FO4 delays and rise/fall times



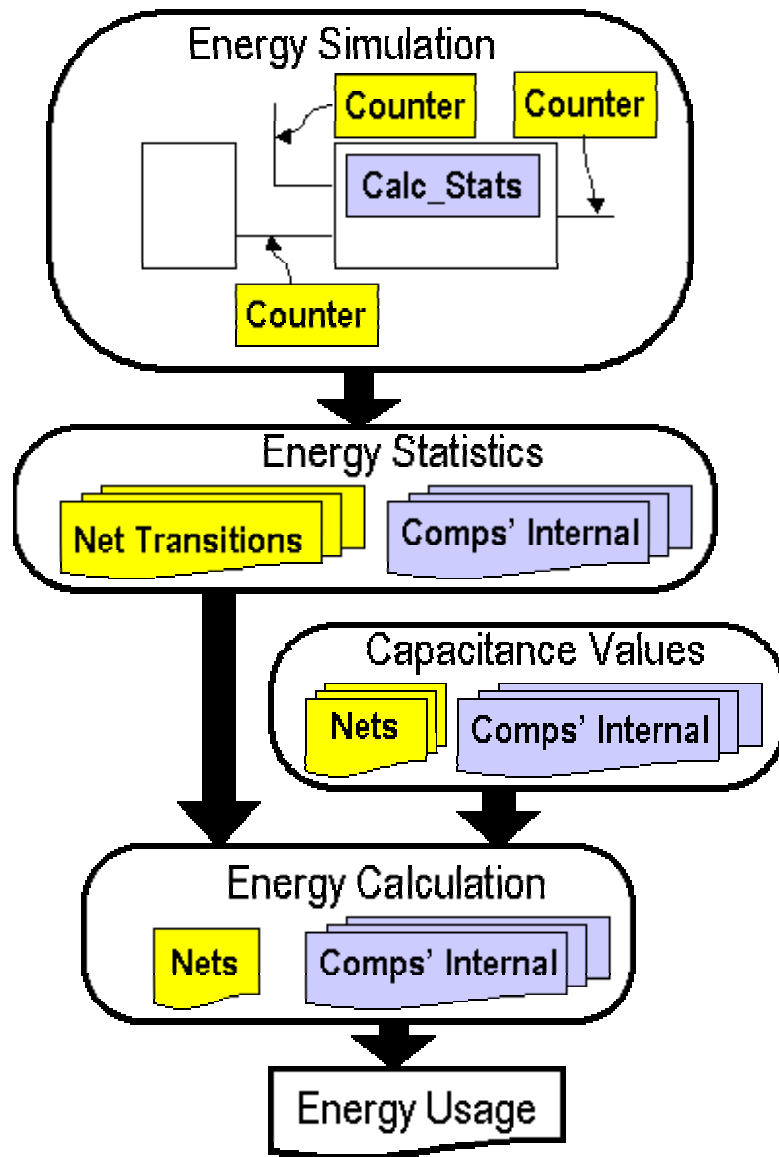
Microprocessor Energy

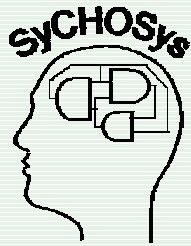
Energy usage in a microprocessor:

- Memory arrays
 - Datapaths
 - **Control**
-
- Synthesized using automated tools — Irregular, hard to model
 - Less than 10% of energy in simple RISC designs
 - Will become more important in low power designs
 - Can be modeled at the level of standard cell gates
 - Work in progress

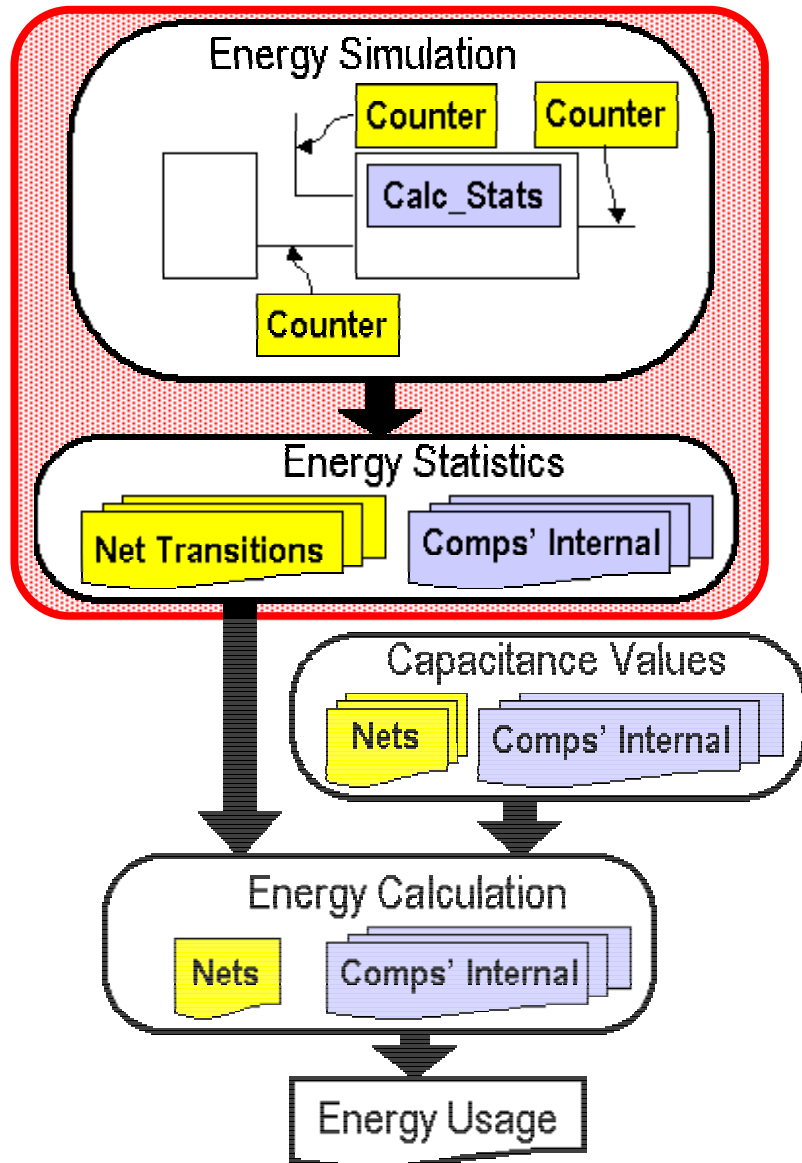


SyCHO Energy Analysis





SyCHO Energy Analysis



Energy Statistics Gathering

Minimal statistics gathering during simulation
Simple to add to SyCHOSys

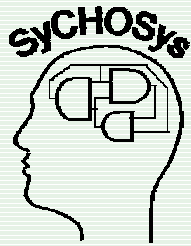
- Structure of design is explicit
- Values on all nets are cycle-accurate
- Can incorporate arbitrary C++ code

Nets:

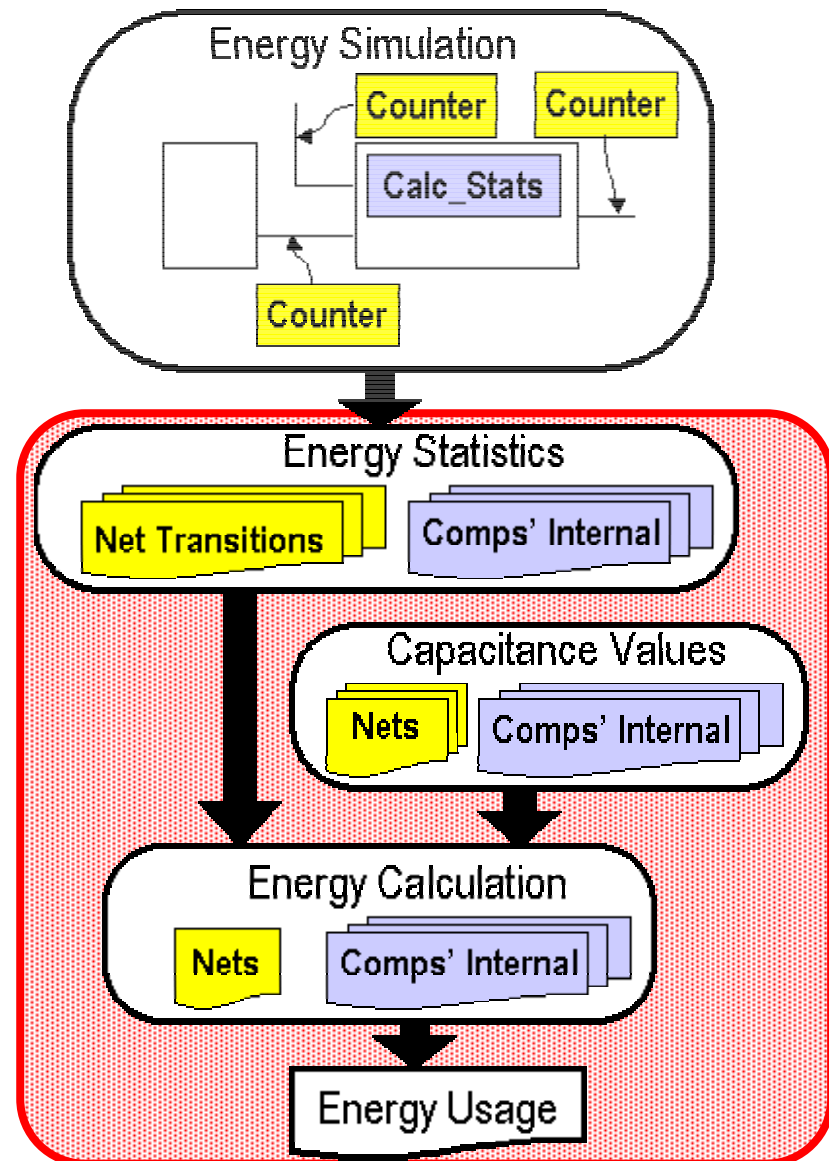
- Count transitions during simulation
- Counters generated automatically

Components:

- Each component tracks arbitrary per-cycle internal statistics



SyCHO Energy Analysis



Energy Calculation

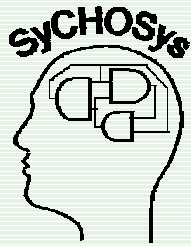
Use simulation statistics and calculated capacitance values to compute energy

Nets:

- Multiply switching frequency by capacitance

Components:

- Each component defines internal energy calculation routine
- Based on internal statistics, input and output switching frequencies, and internal capacitance values

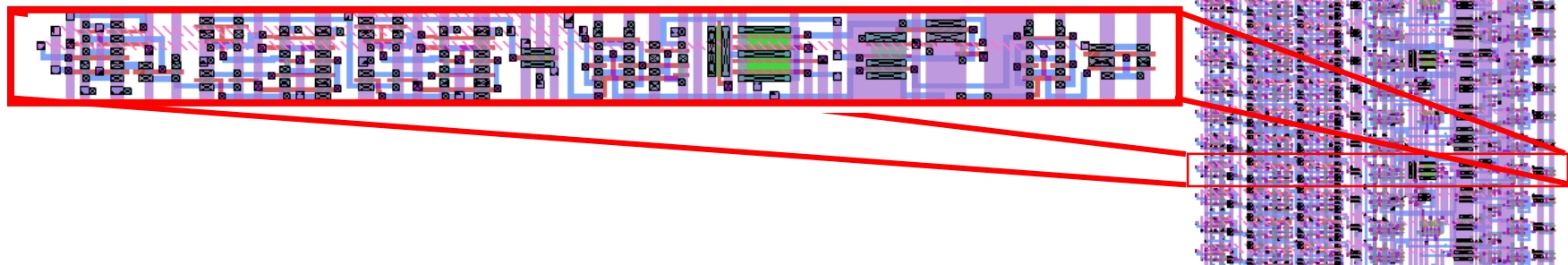


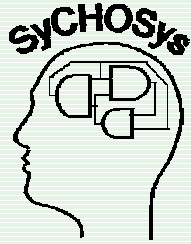
Energy-Performance Model Evaluation

Used GCD circuit as an example datapath

- Various component types (Flip-Flops, Latches, Dynamic)
- Small enough for SPICE simulation

Hand-designed layout (0.25 μm TSMC)

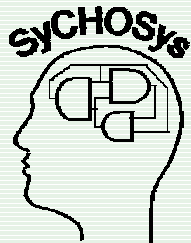




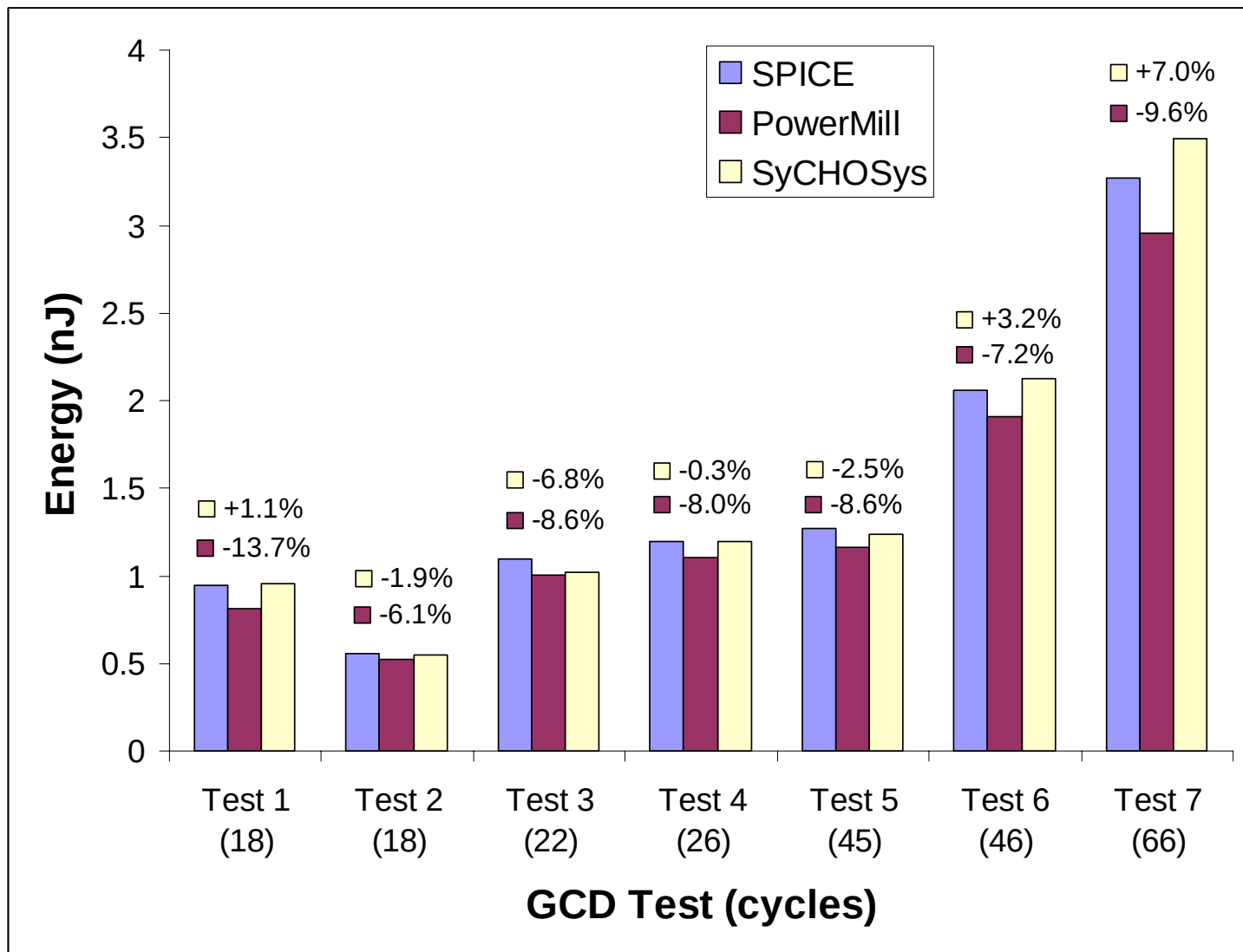
Simulation Speed

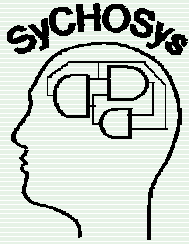
Simulation model	Compiler / Simulation Engine	Simulation Speed (Hz)
C-Behavioral	gcc -O3	109,000,000.00
Verilog-Behavioral	vcs -O3 +2+state	544,000.00
Verilog-Structural	vcs -O3 +2+state	341,000.00
SyCHOSys-Structural	gcc -O3	8,000,000.00
SyCHOSys-Energy	gcc -O3	195,000.00
Extracted Layout	PowerMill	0.73
Extracted Layout	Star-Hspice	0.01

- All tests run on 333 MHz Sun Ultra-5 (Solaris 2.7)



Energy Simulation Results





Status & Future Work

Microprocessor simulation:

- Five stage MIPS RISC including caches and exception handling
- Runs SPECint programs
- Most of energy modeling is complete
- Energy simulation of over 2000 nodes at 16 kHz

Future Work:

- Short circuit & leakage current modeling
- Control logic modeling
- Take energy statistics per static program instruction
- Incorporate SyCHOSys into VLSI tool flow