3rd Annual
Austin Conference on Integrated Systems & Circuits
2008 Publication

KEYNOTE ADDRESS **Dr. Necip Sayiner,** ***President, CEO, and Board Member*** ***Silicon Laboratories***	
Track 1: DIGITAL DESIGN-1 **Session Chair:** ***Mike Seningen***	Track 2: ANALOG/BIO **Session Chair:** ***Ka Leung***
1-1 Design Automation and Verification Methodology Challenges of the Intel Atom Processor ***Rajesh Gupta***	***2-1*** ***Invited Paper:*** Biosensor Microarrays in CMOS ***Arjang Hassibi***
1-2 Reducing Flip-Flop Power for DSP Design ***Bassam Mohd,*** *Martin Saint-Laurent****, Paul Bassett, and Shahid Imam***	2-2 Rail to Rail Fully Differential Sample and Hold Based on Clocked Diffrential Difference Amplifier using Resistive Local Common Mode Feedback ***Jaime Ramirez-Angulo, Clara Lujan-Martinez, Carlos Rubia-Marcos, Ramon G. Carvajal, and Antonio Lopez-Martin***
1-3 Adaptive Voltage Tuning for Dual-Vdd ASICs ***Stephen Bijansky and Adnan Aziz***	2-3 Buck-Boost Converter Based Power Conditioning Circuit for Low Excitation Vibrational Energy Harvesting ***Arvindh Rajasekaran, Abhiman Hande, and Dinesh Bhatia***
Track 3: CAD-1 **Session Chair:** ***Michael Solka***	Track 4: COMPUTER ARITHMETIC **Session Chair:** ***Earl Swartzlander***
3-1 Modeling of NBTI-Induced PMOS Degradation under Arbitrary Dynamic Temperature Variation ***Bin Zhang and Michael Orshansky***	***4-1*** ***Invited Paper:*** Automated Multiplier Design ***K'Andrea C. Bickerstaff and Earl Swartzlander***

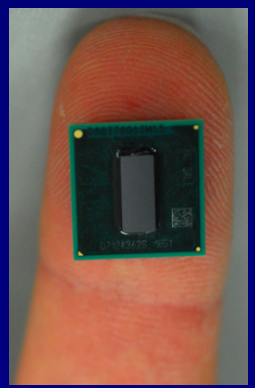
3-2 ELIAD: Efficient Lithography Aware Detailed Router with Compact Post-OPC Printability Prediction ***Minsik Cho, Kun Yuan, Yongchan Ban, and David Z. Pan***	4-2 Full Adder Evaluation and Selection for a Parallel Multiplier ***Mike Spear, Gaurav Tuteja, and Earl Swartzlander***
3-3 COOLER - A Fast Multiobjective Fixed-Outline Thermal Floorplanner ***Debarshi Chatterjee, Theodore Manikas, and Igor Markov***	4-3 Dual Vdd Design Optimization of Fast Multipliers ***Eun Jung Jang, Earl Swartzlander, and Jebediah Keefe***
3-4 Dynamic Compaction with Recursive Learning for Delay Test ***Zheng Wang and Duncan M. H. Walker***	4-4 A Hybrid Approach to Last Stage Addition for Wallace and Dadda Multipliers ***James Haydn Nelson, Eiman Ebrahimi, Mahnaz Sadoughi, and Earl Swartzlander***
TUTORIAL Fractional-N PLL ***Dr. Axel Thomsen,*** ***Silicon Labs***	
KEYNOTE ADDRESS **Dr. Jason Rhode,** ***President and CEO*** ***Cirrus Logic***	
Track 5: DIGITAL DESIGN-2 **Session Chair:** ***Adnan Aziz***	Track 6: ANALOG AND RF **Session Chair:** ***TR Viswanathan***
5-1 ***Invited Paper:*** A Sub 1W to 2W Low Power IA Processor for Mobile Internet Devices in 45nm Hi-K Metal Gate CMOS ***Gianfranco Gerosa, Steve Curtis, Mike D'Addeo, Bo Jiang, Belliappa Kuttanna, Feroze Merchant, Binta Patel, Mohammed Taufique, Haytham Samarchi, and Christopher Weaver***	***6-1*** ***Invited Paper:*** Receivers Design: Case Studies ***Hesam Amir-Aslanzadeh and Edgar Sanchez-Sinencio***
5-2 A High Speed 128-Point Fast Fourier Transform Circuit for OFDM Systems ***Tung-Yeh Wu and Jacob Abraham***	6-2 Feed-Forward Interference Suppression for Broadband Systems ***Xin Wang and Ranjit Gharpurey***

5-3 Hardware Trojan Modeling and Detection Techniques ***Daniel G. Saab, Fatih Kocan, and Jacob Abraham***	6-3 A Linear Transconductor using Series-Connected CMOS Quad ***Venkatesh Acharya, Bhaskar Banerjee, and TR*** *Viswanathan*
5-4 ***Invited Paper:*** Migration of Cell Broadband Engine from 65nm SOI to 45nm SOI ***Osamu Takahashi*** ***(Scott Cottier presenting)***	6-4 Indirect Compensation Techniques for Three-Stage CMOS Op-Amps ***Vishal Saxena, Jacob Baker, and TR Viswanathan***
Track 7: CAD-2 **Session Chair:** ***Michael Orshansky***	Track 8: SYSTEMS & ARCHITECTURES **Session Chair:** ***Mattan Erez***
7-1 3D Resistance Extraction with Lithographic and Scattering Effect ***Ying Zhou, Zhuo Li, and Weiping Shi***	8-1 Power Analysis of a Path-Based Perception Branch Predictor ***Justin Friesenhahn, Lizy Kurian John, and Mark McDermott***
7-2 Accelerating Statistical Static Timing Analysis using Graphics Processing Units ***Kanupriya Gulati and Sunil Khatri***	8-2 Hardware / Software Tradeoffs in Multicore Architectures ***Steven Guccione***
7-3 Autonomous Optical Proximity Correction: The New Frontier of Design for Manufacturing? ***Shanhu Shen, Peng Yu, and David Z. Pan***	8-3 Workload Slicing for Detailed Pre-Silicon Power Estimation ***Hassan Al-Sukhni, James Holt, David Lindberg, and Michele Reese***
TUTORIAL Holistic Coupling of Manufacturing and Design ***Dr. Sani Nassif,*** **IBM**	
TUTORIAL Out of Order Superscalar Architecture ***Dr. Derek Chiou,*** **University of Texas at Austin**	
TUTORIAL Constraint Solving for Functional Verification ***Dr. Andreas Kuhlmann,*** **Cadence Berkeley Labs**	

Design Automation & Verification Methodology Challenges of the Intel Atom Processor

Rajesh Gupta, Albert Holguin, Yonglin Zhang,
Craig Horton, Olga Zaporojets, Venkat Chiluvuri, Laurent Chaouat,
Dinos Moundanos, Srinivas Gummadi, Brian Klaas

Intel Ultra Mobility Group
Austin, TX
May 7, 2008



Outline

- Intel Atom processor design overview
- Intel 45nm process technology features
- Challenges and approaches
- Design & verification methodology overview
- Conclusion

Intel Atom Design Overview

- Intel Architecture (IA) processor, fully x86 64-bit compatible, capable of running desktop OS
- Targeted at Ultramobile PC (UMPC) and Mobile Internet Device (MID) applications
- Thermal Design Power (TDP) of only 2W at 2GHz frequency
- Silverthorne is the first member of the Atom family
- Design contains 47M transistors in <25mm^2 area (13.8M transistors in the CPU core)

- *Details: [Gerosa et al., this conf. and ISSCC 2008]; [Halfhill, "Intel's Tiny Atom", Microprocessor Report 4/7/08]*

Intel 45nm Process

- Uses new Intel® 45nm high-k metal gate silicon technology and hafnium-based low leakage circuits
- Dramatic increase in energy efficiency
- Same process technology supports low power and high speed
- 9 routing metal layers

- *Details: [Mistry et al., Tech. Dig. IEDM, Dec. 2007]*

The Design Challenge

- Power, power, power!
  Reduce by 10x

The Design Challenge

- Power, power, power!
  Reduce by 10x
- Don’t forget frequency

The Design Challenge

- Power, power, power! Reduce by 10x
- Don’t forget frequency
- Don’t forget x86 compatibility

Can Low Power be Fast?

Can Low Power be Fast? Maybe

Sharp-Willcom D4, equipped with Intel's Atom processor, on sale in June 2008

Thinking Low Power

- Generations of x86 processor designs have been pushed for frequency and performance
- Intel architecture and process technology were traditionally tuned for performance
- Intel 45nm process enables low power and high frequency on the same die
- Across the board rethinking of architecture, logic and circuit techniques *[see Gerosa paper]*

Low Power Strategy

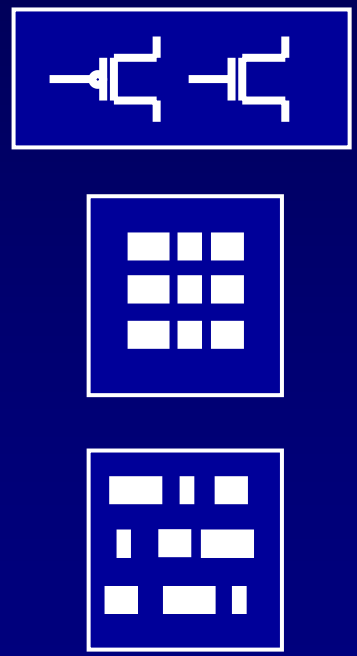
- RTL and logic design
- Low leakage circuits
- Power-hungry timing optimization
- Power grid design
- Low power clock tree synthesis
- Power-friendly layout

Key Success Factors

- Design partitioning and hierarchy
- Mixed cell/transistor level design & analysis
- Timing-power optimization
- Unified data model
- Productive custom design
- Clock tree synthesis
- RTL power savings

Design Partitioning

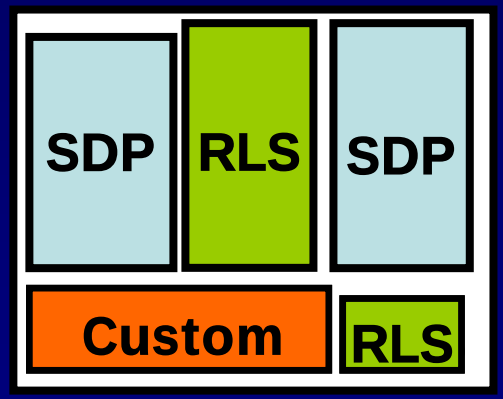
- Most of the low level functional unit blocks (FUB's) are adapted from legacy designs
- Three ways to implement the FUB's
  - Custom: highly critical or analog logic → manual schematics & layout
  - SDP (structured data path): critical array/data path logic → user-guided automation
  - RLS (RTL to layout synthesis): control logic and less critical data paths → fully automated



Physical Design Partitioning

- For physical design, the FUB's can be merged or kept separate in the hierarchy
  - Separate: sea of FUB's (hierchical floorplanning and optimization)
  - Smashed: sea of cells (flat optimization)
- Atom design uses sea of FUB's

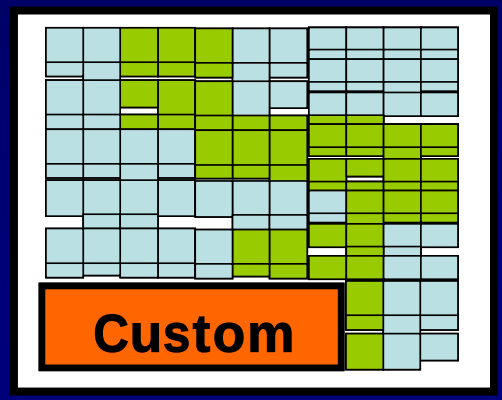
"Sea of FUB's"



Hierarchical SDP

Hierarchical RLS

"Sea of Cells"

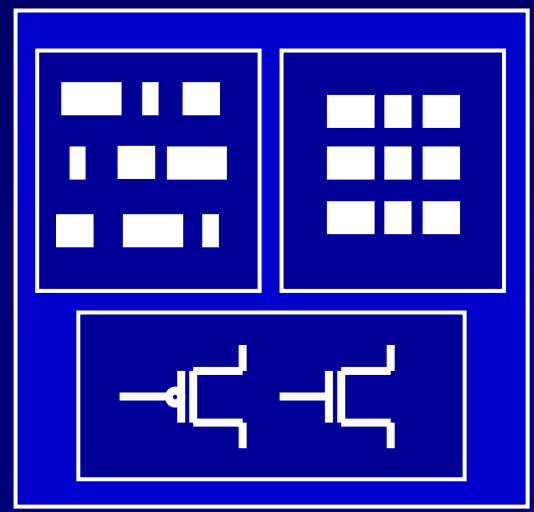


Smashed SDP

Smashed RLS

Intel Atom Design Hierarchy

- Two level physical design hierarchy:
  chip → FUB → standard cell
- Chip floorplan contains 291 FUB's of 3 types
  1. RTL to layout synthesis (RLS): 46%
     - Synthesis, placement & auto routing flow
  2. Structured data path (SDP): 45%
     - Designer-guided schematic, placement & trunk routing; auto routing
  3. Custom: 9%
     - Full custom schematic & layout



Design Style Comparison

Most manual effort

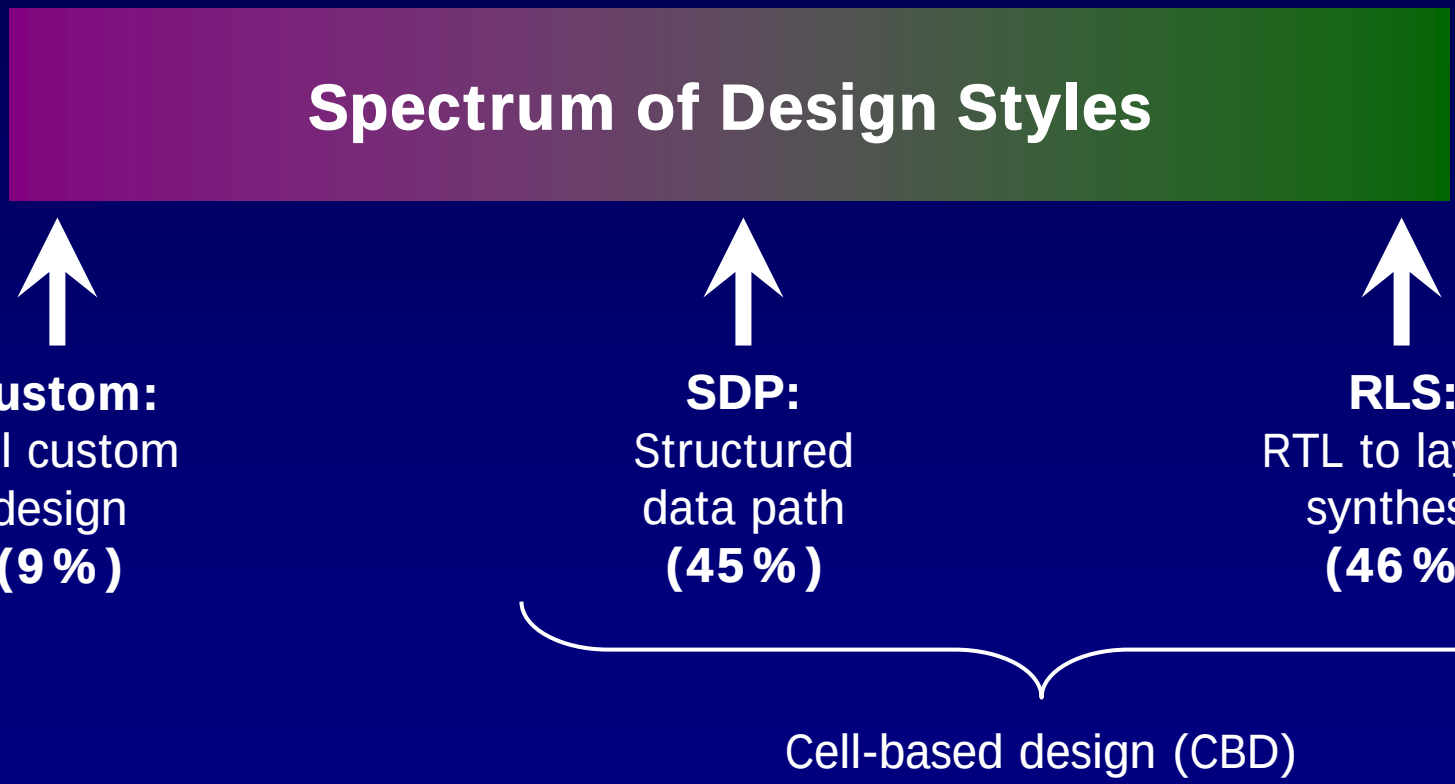
Best design control

Best design optimization

Most automation

Least predictability

Fastest turnaround



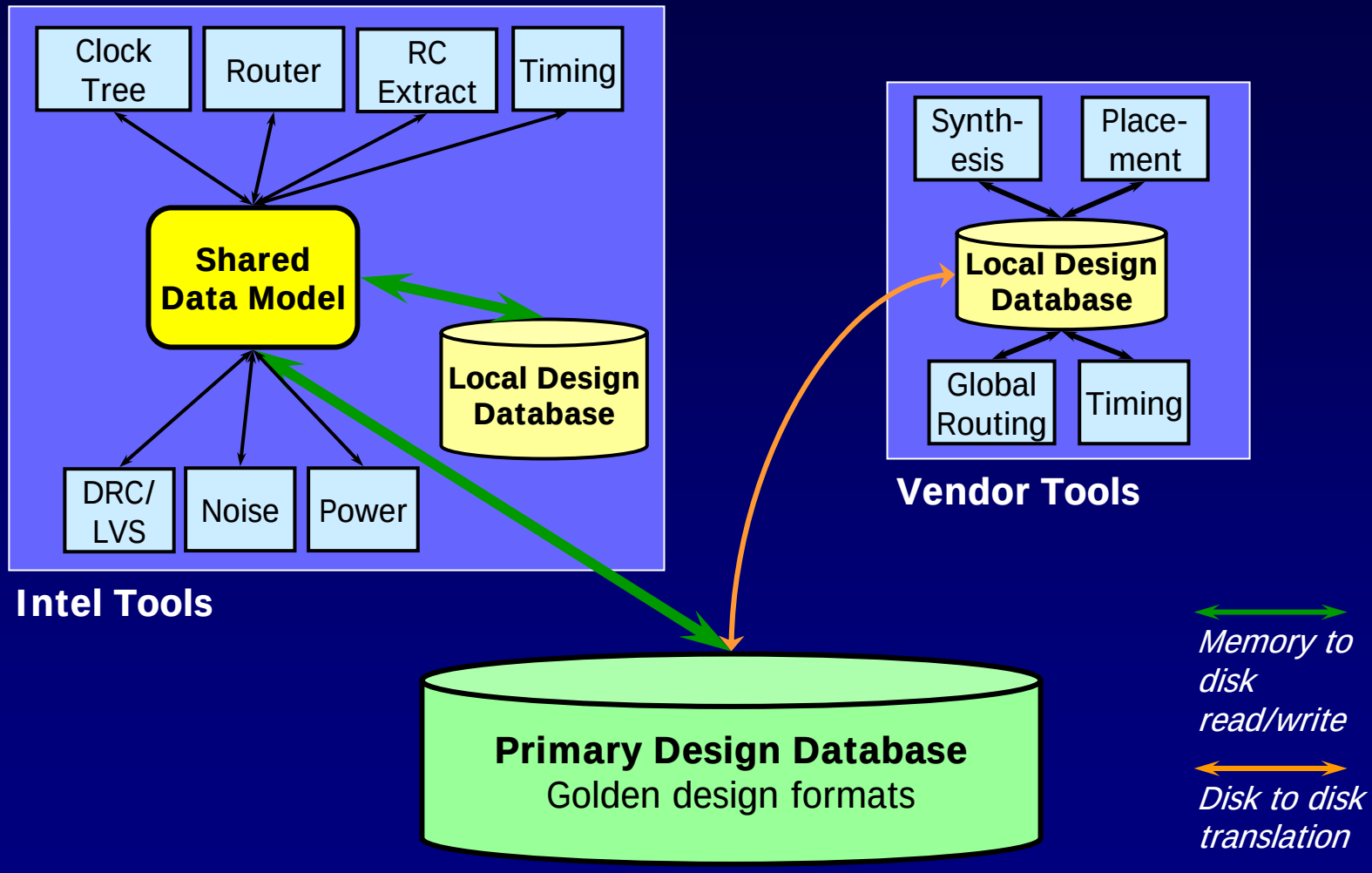
Mixed Level Timing Analysis

- RLS FUB's are built with characterized standard cells
- SDP FUB designers can create custom cells, which
  - are not characterized and cannot be used for synthesis
  - have schematic and layout customized for specific design scenario
  - allow more critical logic to be implemented as SDP rather than Custom FUB's
- Custom FUB's have high content of uncharacterized transistor level schematics
- Static timing analysis flow efficiently handles the mix
  - Standard cells (characterized) are timed using I→O delay equations
  - Custom cells (uncharacterized) are timed at the transistor level
- Enables higher productivity and better design optimization

Timing/Power Optimization

- Low power strategy makes the timing optimization harder
- Timing closure must be achieved with the signoff timer
- Custom and SDP FUB's use the signoff timer throughout the design cycle
  - RC extractor and static timer are tightly integrated with the user-guided placement and routing tools
  - Unified data model enables fast iterations between logic, physical and timing domains
- RLS FUB's timing optimized in two phases
  - Global design phase accomplished via synthesis & placement tools
  - Detailed refinement and optimization coupled with signoff timer; utilizes algorithms and custom tricks

Unified Data Models



Optimization & Convergence on Unified Data Model

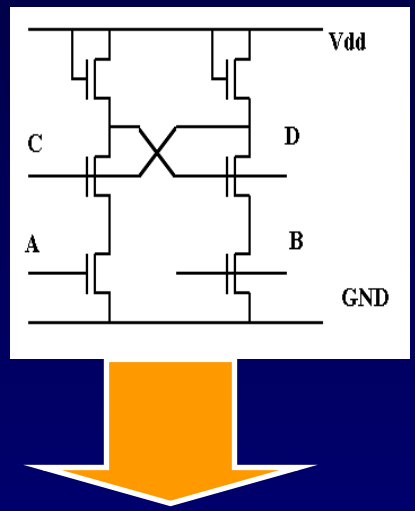
- Data Model Features
  - Hierarchical netlist/layout data model based on cell-pin-net object-oriented schema
  - Interconnect parasitic & virtual repeater data model
  - Persistent (on-disk) database
  - Potential future linkage with Si2's OpenAccess
- Improved Convergence
  - Detailed routing and RC extraction tools see and interpret the detailed layout constructs in a consistent way
  - Faster convergence between layout and timing

Custom Design Flow Overview

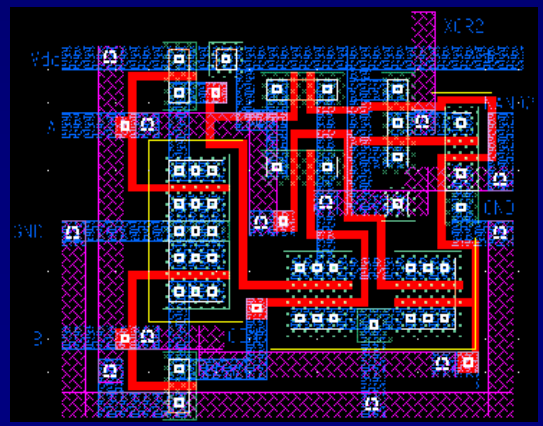
Design Spec (RTL)

```
module chip (i, o, ckgridm1n00, controlinput);
  input  i;
  output o;
  input  ckgridm1n00;
  input  controlinput;
  wire   i;
  reg    o;
  wire   ckgridm1n00;
  wire   controlinput;

  wire   DeclarationForSakeOfInterfaceSeparation;

  // Internal signals
  reg    ff1, ff2;
  wire   myclk, lceout, lcbout;

  // Clock network

  v00rbe00ln0f0 myrbe (.atpgovrd(1'b0),
                       .clk(ckgridm1n00),
                       .dften(1'b1),
                       .fd(1'b1),
                       .glben(1'b1),
                       .o(myclk),
                       .rd(1'b1),
                       .rpen(1'b0),
                       .rpovrd(1'b0));

  // Input drives inverter 1 and flop 1
  always @ (posedge myclk)
    begin
      ff1 <= ~i;
    end

...
```

Schematic Design



Physical Design

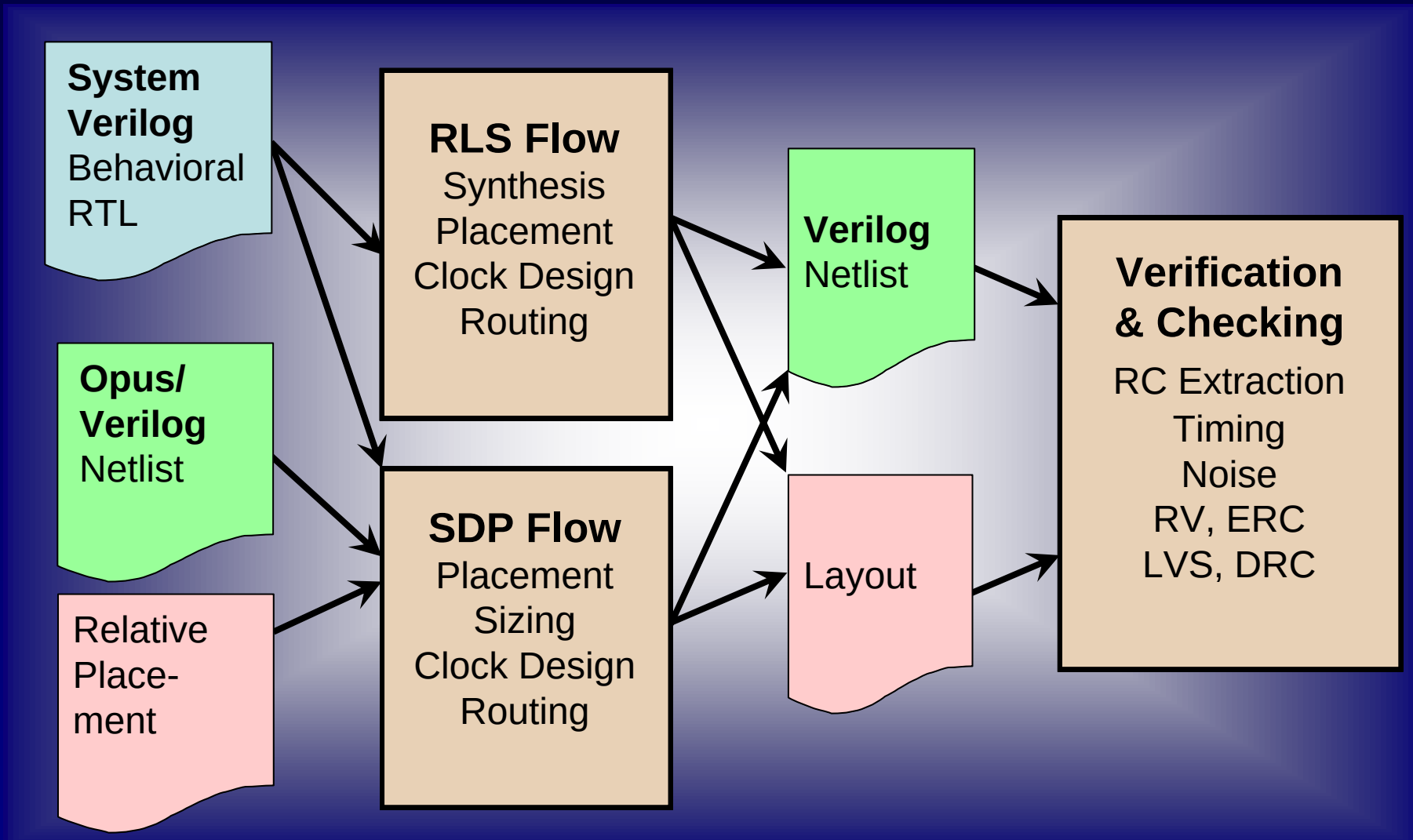


Verify

Design Objectives:

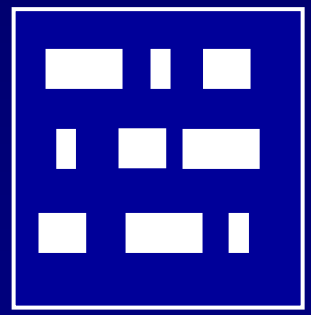
- Area—die size constraint
- Timing—clock cycle constraint
- Power—thermal and battery life constraints
- Noise—signal integrity constraints
- Reliability—long life
- Manufacturability—can be fabricated!

SDP / RLS Cell Based Design Overview



RTL to Layout Synthesis (RLS) Blocks

- Standard cell based synthesis & placement
  - Industry standard CAD flows
  - Converge with associated timing tool
- Post-placement convergence on tightly integrated PD-timing tools
  - Physical design and timing tools integrated via unified data model
    - Clock tree synthesis
    - Detailed routing
    - Performance verification
    - Timing/power optimization

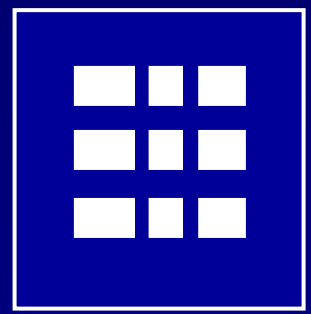


Low Power Clock Tree Design

- Clock signals are the most active and have high capacitance
  - 16x the active power of data signals on average
  - Wire contributes about half of the clock capacitance
- Partial irregular grid for global clock distribution
  - Irregular—clock spine locations are adjusted based on receiver distribution
  - Partial—ends of clock spines are trimmed to reduce load
- Constrained clock tree synthesis for local clock distribution
  - Multiple levels of clock buffers inside the FUB's, rooted at the clock spines
  - Each FUB's clock tree rooted at multiple clock spines
  - Clock tree synthesis tool does automatic clock buffer insertion and clock preroute/shield creation
  - Constraints on fanout and buffer range are tuned to meet high frequency, low skew requirements

Structured Data Path (SDP) Blocks

- The SDP design tool allows the designer to effeciently guide the placement and routing to meet critical area and timing goals
- Placement of array/datapath elements is controlled in a highly productive way, using text-based placement directives
- Pre-routes (individual or vectored busses) are generated as user-guided trunk routes with adaptive alignment controls
- Auto routing onwards is similar to the flow for RLS (synthesized) blocks



Front End Design Methodology Challenges

- Scan for testability was limited due to area/power considerations
  - Chip had only ~33% of sequential elements scanned
  - As a result ATPG was hard, required full initial state calculation, sequential algorithms and provided targeted coverage
  - Rest of coverage to achieve DPM requirements achieved through functional test
- RTL design for low power
  - Addressed early when RTL changes can have high impact
  - Used static and dynamic power aware checkers, aggressive identification of clock gating opportunities and early power estimation
- Formal equivalence verification (FEV) challenges mainly for custom FUB's (digital/analog interfaces)
  - Advanced CPU design is actually less challenging for FEV compared to large SOC designs
  - Logic between sequential elements is shallow, in contrast to low frequency SOC blocks, where there are more challenges

CAD Framework Challenges

- Processing of large volumes of data to facilitate quick convergence with rollup/rolldown methodology
- Handling of different types of design blocks (RLS, SDP, full custom) as seamlessly as possible to the users
- Enabling multi-site support to add design resources and meet schedules

Future Research

- Better placement optimization of mixed structured/unstructured logic for timing and power
- CAD frameworks for mixed design styles
- Power grid efficiency and integrity
- Low power clock distribution

Conclusion

- Major power reduction in x86 design achieved thanks to a combination of 45nm process, architecture, logic and circuit design changes
- Major design automation and verification methodology enhancements to meet the aggressive power/performance goals

Acknowledgements

- Intel Design Technology Solutions (DTS) for 45nm CAD tools and significant support for the design effort
- Silverthorne design experts were key partners in defining and validating the design automation and verification flows
- Nehalem (desktop/server processor) design team was a partner in the methodology and automation development

Reducing Flip-Flop Power for DSP Design

Bassam Mohd Martin Saint-Laurent Paul Bassett Shahid Imam

***Abstract*—Flip-flop (FF) power consumption is one of the major power dissipation sources in DSP cores. This paper examines the total power, clock and FF power dissipation in a 65-nm DSP design. It analyzes the FF power dissipation across FF types and FF sizes. From a power perspective, FFs can be classified as data FFs and control FFs. Several power reduction techniques are examined, some of which are targeted for specific FF type. A new low-power clock-gating cell is proposed. Finally, the paper presents a novel approach to model the impact of physical design on enabled clock power.**

***Index Terms*—Digital Signal Processors, Microprocessors, VLSI**

I. Introduction

Flip-flops (FF) are very important circuit elements in synchronous VLSI designs. Because leakage power is typically less than 5% of the total power (for active processors in low-leakage processes in 65 nm), the focus of this paper is to reduce dynamic power in FFs. FF dynamic power can be attributed to two sources. The first source is the power dissipated to deliver a *clock* to each FF. This includes both global clock distribution and final buffering of the clock local to the FF. The second source of FF power dissipation is the *internal FF activity* associated with the data flow through the FF, including buffering input, output and feedback circuitry.

A variety of FF designs and implementations have been proposed to enhance speed, reduce power and improve race tolerance [1-[13]. The master-slave FF, which consists of a pair of latches, has superior race tolerance but tends to suffer from poor latency. Examples of master-slave FFs are PowerPC603 FF [6] and the pseudo-static flip-flop of [7]. Pulse-triggered FFs consist of a pulse generator and a latch. They tend to have a lower latency but suffer from poor race tolerance. Based on where the pulse is generated, pulse-triggered FFs can be classified into implicit pulse-triggered FF (ip-FF) and explicit pulse-triggered FF. Examples of ip-FF include the hybrid latch FF [8], semi-dynamic FF [9] and implicit pulse data-close-to-output FF [10]. An example of ep-FF is the explicit-pulse data-close-to-output (ep-DCO) FF [2]. ep-DCO dissipates power when input is constant, which is addressed in explicit-pulse static FF [11] and conditional discharge FF [12]. Dual edge-trigger FF (DETFF) consists of two latches connected in parallel and activated in different clock phases [13]. DETFF can help reduce clock frequency to half while maintaining the same data throughput.

Most of the published work focuses on designing and optimizing FFs outside their usage model. As a result, opportunities for further power optimization have not been examined. In this paper, we focus on the usage model by analyzing FF spectrum graphs, which plots the FF power against FF bit-width. Careful examination of the FF spectrum graph indicates that FFs can be divided into two groups: data FFs and control FFs. Data FF store data between major pipeline stages and blocks. They exist in arrays, with array size equal to multiple the word sizes. For example, if data and address size is 32-bit, data FFs exists mostly in the following array size: 8, 16, 32, 64 and 256. Because it is very common to drop the least bits of the address bus, FFs arrays of 30 and 31 bits are common as well. The other type of FFs is control FFs which are mostly used in finite state machines. The control FFs exist in smaller number, usually 8 or less per state machine. For enhanced timing, state machines are commonly coded using one-hot state assignments. However, we will show the one-hot design style does not necessarily yield the most energy efficient implementation.

Several FFs power saving techniques will be discussed. The paper examines reducing clock power using a new low power clock-gating cell (CGC). Because all FFs are not created equal, different power optimization techniques are examined per type. For data FFs, we will examine the FF bank implementation. FF bank is an array of pulsed FFs with a shared pulse generator. Several optimization techniques are utilized to reduce overall FF bank power. To reduce the control FFs power, it is very important to reduce number of FFs. The paper compares the power consumption of an example state machine implemented in one-hot encoding with other encodings.

The rest of the paper is organized as follows. Section 2 discusses the power simulation results from the DSP design. Section 3 explains methods to reduce FF power dissipation.

II. FF Power Analysis

This section analyzes the major power components for the DSP design. Power estimates are calculated using vector-based power flow, illustrated in Figure 1. Simulation generated waveforms are fed into power tools to calculate design power. Two tests are used for power bench mark: COMP test (computation intensive test) and BUS test (bus transaction intensive test.) Further power numbers have been normalized with COMP test.

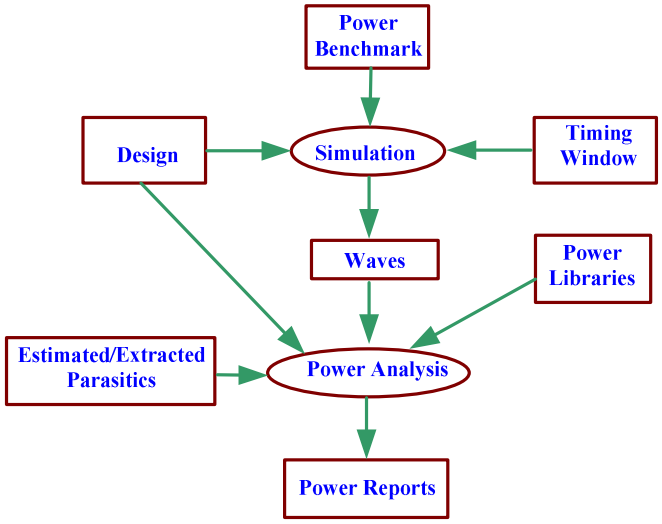


Figure 1. Power Estimation Flow

Figure 2 show how total power is divided across memory arrays (register files and RAMs), FF, clock, logic and others (buffers, latches, etc.) We can clearly see that FF dissipates from 25% to 30% of the total power. The design uses mostly four types of FFs:

- Enable-FF
- Reset FF
- Enable-Reset FF
- No-Enable No-Reset FF

Figure 3 shows that FFs with enable capability dissipates around 67% of the total FF power. Hence, Enabled FFs are dominant.

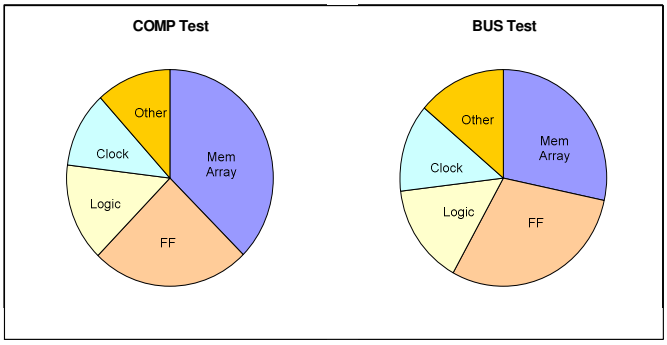


Figure 2. Power Components for COMP and BUS Tests

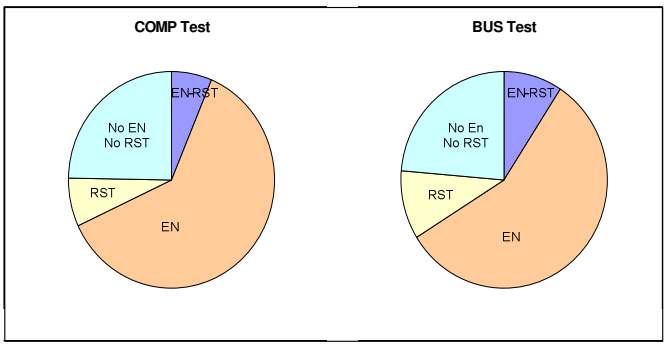


Figure 3. FF Power across FF types

Figure 4 and Figure 5 plots the FF power (normalized to average) against FF bit-width. These graphs are referred to as FF spectrum graph. Figure 6 shows the average FF spectrum graph for both testsFrom Figure 4, Figure 5 and Figure 6, we notice the following:

- The first observation is that 32% of FF power is dissipated by FF size 16-bit, 32-bit and 64-bit. The 32-bit FF is the most power consuming FF. These are data FFs, which store data and addresses. Some address do not included the least bits, which explains power seen FF size 30- and 31-bits. Some data bus includes few status bits which also explain the power shown in FF size 33-, 34- and 35-bits. FFs carrying data and addresses are referred to as *data FFs*. They have the distinctive feature having size at or close to half-, one- or double word. FF banks, explained below, targets data FFs.
- The second observation is that 43% of FF power is consumed by FF sized 8-bit or less. 1-bit FF is the second most consuming FFs. These are mostly *control FFs* in state machines and small counters. Control FFs implemented in 1-hot encoding is a prime target for further power reduction, as explained in the following section.

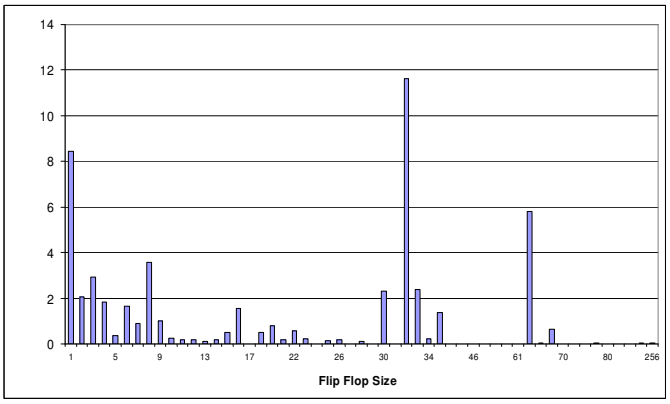


Figure 4. FF spectrum graph for COMP Test

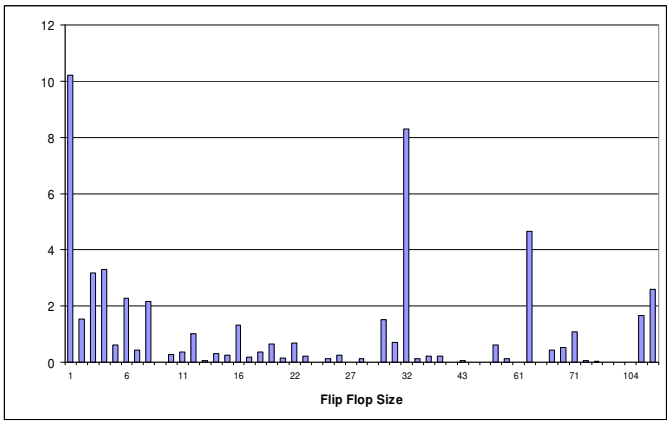


Figure 5. FF spectrum graph for BUS Test

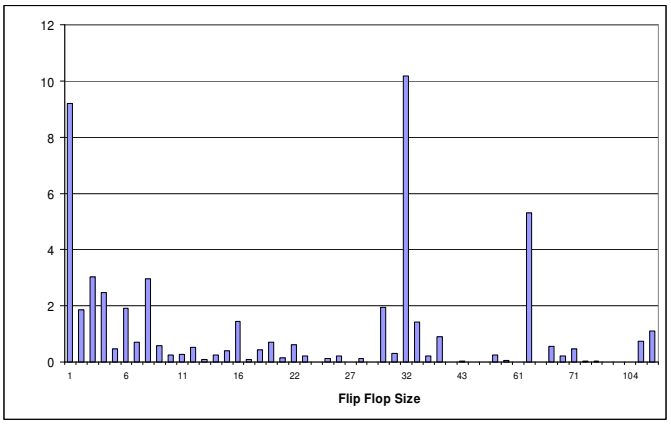


Figure 6. FF spectrum graph (averaging COMP and BUS Tests)

III. Reducing FF Power

In this section we will discuss FF power reduction techniques, as well as modeling gated clocks.

A. Clock Gating

Clock gating is commonly used to reduce FF power. Typically, this method disables the clock going to one or several FFs by introducing an *AND* gate on the clock path. The *and* gate can be used to block the clock or let it reach the flip-flops based on the value of a control signal called the enable. This enable should not change while the clock is high to avoid undesirable clock glitches. Consequently, the enable is typically held stable by a latch during that time.

A conventional clock gating cell (CGC) combining the *AND* gate with the latch holding the enable is shown in Figure 7. When the clock toggles, 9 transistors always toggle. A lower-power alternative where only 4 transistors always toggle is shown in Figure 8. When enabled, the low-power CGC consumes 7% less power than the conventional one. But when disabled, the low-power CGC is about 3 times better. The low-power CGC uses fewer devices. Its area is estimated to be about 10% smaller. The input capacitance of the low-power CGC is 1.7 fF. For the conventional implementation, the input capacitance is 2.1 fF. The setup time of the enable is about two inverter delays (fanout of four) worst for the low-power CGC. Having both types of CGCs available will allow design flow to optimize clock gating paths based on area/speed/power tradeoffs.

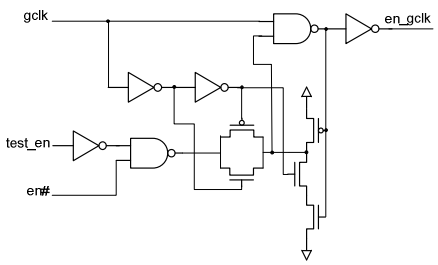


Figure 7 Conventional CGC

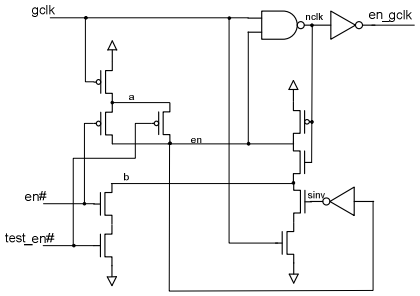


Figure 8 Low-Power CGC

B. Impact of Physical Design on Gated Clock Power

More clock gating is not always better. Because clock gating introduces additional clock domains, it imposes physical design constraints. The FFs of a given clock domain are not always placed in close proximity. This increases the interconnect length that the CGC must drive, which in turn increases the capacitance of the local clock. Clock gating does not always sufficiently reduce the toggle rate of the local clock to compensate this increase in capacitance. This section examines the impact of number of enables on clock power.

To a first order, the FF and clock power can be estimated as:

$$\text{Power}_{(FF+CLK)} = P_{Interconnect} + P_{Flip\text{-}Flop} + P_{Enable\text{-}Logic} \quad (1)$$

Error! Reference source not found. defines the terms of the above equation.

Table 2 Power terms and definitions

Term	Definition
$P_{Interconnect}$	$L^* C_{wire}$
$P_{Flip\text{-}Flip}$	$\alpha_{ave} * N * C_{FF}$
$P_{Enable\text{-}Logic}$	$E * C_{en\_logic}$
L	Length of clock interconnect
C_{wire}	Wire capacitance per unit length
α_{avg}	Average activity factor for the clock
N	Number of clock loads in the design.
C_{eff}	Total capacitance derated by activity factor
C_{FF}	Effective clock capacitance per FF
A	Design area. Areas are assumed to be squared.
E	Number of enables
C_{en_logic}	Capacitance of enable logic

The interconnect power can be estimated using the model proposed by Bern in [17]. According to the model, the length L of a minimal rectilinear Steiner tree (MRST) connecting N loads uniformly distributed on a square of area A is:

$$L = \beta\sqrt{NA} \quad (5)$$

where $\beta \approx 0.73$ [19].

Most designs have a large number of clock domains that are controlled individually by an enable signal. The effective interconnect length switching per cycle in a particular domain i is:

$$L_i = \alpha_i \beta \sqrt{A_i N_i} \quad (6)$$

where α_i is switching activity of its clock, A_i is its area, and N_i is the number of clock loads it has. For a design with E enables, the total effective length L_{eff} can be expressed as:

$$L_{eff} = \sum_{i=1}^{E} L_i = \beta \sum_{i=1}^{E} \alpha_i \sqrt{A_i N_i} \quad (7)$$

Usually, the micro-architecture and logic design determine E, α_{avg}, and N_i. On the other hand, the physical design determines A_i. If all the E clock domains have the same area A_x and the same number of loads, then $N_i = N / E$ and:

$$L_{eff} = \alpha_{avg} \beta \sqrt{A_x N E} \quad (8)$$

where α_{avg} is the average clock switching activity:

$$\alpha_{avg} = \frac{1}{E} \sum_{i=1}^{E} \alpha_i$$

It is interesting to note that the interconnect length that is effectively switching increases fairly slowly with the number of enables. Doing more clock gating tends to reduce α_{avg}, but it does increases the number of domains. It also tends to increase the number of loads because additional CGCs are required. Conceptually, (8) can be used to determine if clock gating sufficiently reduces α_{avg} to compensate the other increases.

It is convenient to introduce a parameter r describing the size of each clock domain A_x relative to the chip area A:

$$A_x = r A \quad (9)$$

where $0 \le r \le 1$. Because they may overlap, it is possible for every region to occupy the entire chip area as shown in Figure 9 (a). The overlap can also be partial as shown in Figure 9 (a) or non existent as shown in Figure 9 (c, d.)

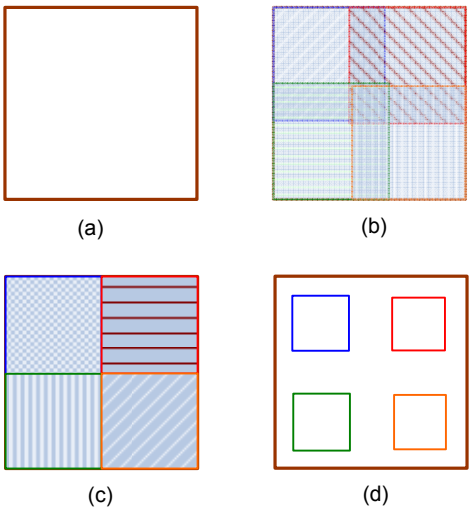


Figure 9 Different sizes of Ax

C. *Reducing Data FF Power: FF Bank*

A strategy to help ensure efficient clock gating and reduce clock power is to bundle several flip-flops driven by the same CGC in a single cell, as shown in Figure 10. The length of the clock wire to the flip-flops can then be minimized and the circuitry overhead of the clock gating and buffering can be amortized more effectively. Extra spacing can also be added to reduce the clock capacitance even more. Moreover, circuit techniques typically used in custom designs, like the use of pulsed clocks on latches having a small number of clocked transistors, can also be applied to further reduce the clock power.

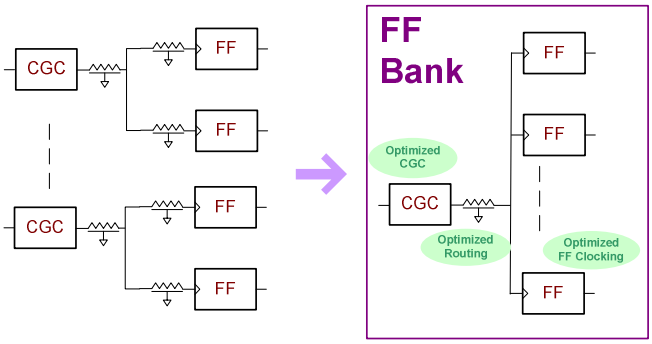


Figure 10 FF Bank Optimizations

For a bank of 32 pulsed latches using the improved write-port topology of Figure 11, the clock power can be reduced by a factor of 4 compared to a bank of conventional master-slave flip-flops where the clock is optimally routed. The power savings using 32-bit FFBank is estimated to be ~5%

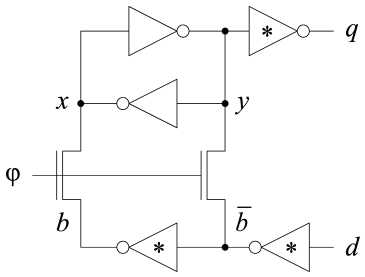


Figure 11 Improved Write-Port Latch

D. *Reducing Control FF Power: FSM Encoding*

One hot-encoding style improves the design speed and it is very commonly used in state machine implementations and control blocks. In this section we will demonstrate the impact of state machine encoding on FSM power dissipation. The power evaluation will use a "101"-sequence detector FSM shown in left side of Figure 12. The state machine was coded and tested using 100,000 random input bits. The right side of Figure 12 shows the average state transitions during the test. Table 1 summarizes the power dissipation for various cases. The first case implements the four states of the FSM in 1-hot encoding style. The second and third cases implement FSM states using 2-FFs, but with different encodings. It is clear that the 1-hot encoding doubles power dissipation. Also, the third encoding saves about 11% compared to the second.

The question then is how to decide on the most efficient 2-state encoding for the state machine. The main factor is the power dissipated by the combinational logic for each encoding. Each encoding results in different combinational logic realization.

Table 2 shows the activity factors for FF inputs for case-2

and case-3. Those activity factors can be derived from the state activity factors shown in right side of Figure 12. It is clear that activity factors for case-2 is lower. But despite this, it has the higher power.

In summary, 1-hot encoding does not necessarily yield the lowest power implementation. N-state FSM should be encoded using $\log_2 N$ FFs. Several encoding experiments should be done to find the most power efficient encoding.

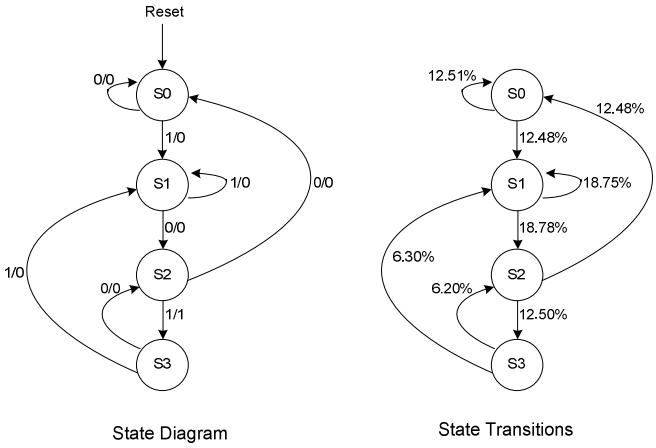


Figure 12 Sequence Detector State machine

Table 1. FSM Power dissipation for various encodings

Case	State Encoding				Power (uW)
	S0	S1	S2	S3	
Case 1 (one -hot)	0001	0010	0100	1000	79.2
Case 2	00	11	01	10	34.3
Case 3	00	01	11	10	30.7

Table 2. Activity Factors for FF Inputs

Case	Activity Factor		
	FF[1]	FF[0]	Average (of FF[1] and FF[0])
Case 2	56.6	48.6	52.6
Case 3	52.3	59.3	55.8

IV. CONCLUSION

This paper focused on the power consumption of FFs in a 65-nm DSP core. The paper presented several power reduction techniques. The first technique is clock gating which disables the clock feeding the FF when it is functionally not required to be active. Another technique is the use of low power CGCs which have less switching capacitance than regular CGCs. The FF bank technique reduces clock power for grouped FFs. Finally, the state encoding analysis concludes that one-hot encoding does not necessarily yield the lowest power implementation. Other state encoding should be examined with further synthesis trials.

Future work should focus on optimum multi-level clock gating. Each addition clock gating reduces the activity of some FFs further more increases the overhead of generating and routing enabled clocks. Another area of improvement is determining the most energy-efficient state machine encoding. This would require a combination of algorithms and trial synthesis.

REFERENCES

[1] Stojanovic, V.; Oklobdzija, V.G.; "Comparative analysis of master-slave latches and flip-flops for high-performance and low-power systems", *IEEE Journal of Solid-State Circuits,* April 1999 Page(s):536 – 548.

[2] Tschanz, J.; Narendra, S.; Zhanping Chen; Borkar, S.; Sachdev, M.; Vivek De; "Comparative delay and energy of single edge-triggered and dual edge-triggered pulsed flip-flops for high-performance microprocessors", *International Symposium on Low Power Electronics and Design,* Aug. 2001 Page(s):147 – 152.

[3] Markovic, D.; Nikolic, B.; Brodersen, R.W.; "Analysis and design of low-energy flip-flops", *International Symposium on Low Power Electronics and Design,* 2001 Page(s):52 – 55.

[4] Phyu, M.W.; Goh, W.L.; Yeo, K.S., "A low-power static dual edge-triggered flip-flop using an output-controlled discharge configuration", *IEEE International Symposium on Circuits and Systems*, May 2005 Page(s):2429 - 2432

[5] Peiyi Zhao; Darwish, T.K.; Bayoumi, M.A.;"High-performance and low-power conditional discharge flip-flop" *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, May 2004 Page(s):477 – 484.

[6] G. Gerosa *et al.,* "A 2.2W, *80* MHz superscalar RISC microprocessor," *IEEE J. Solid-state Circuits,* vol. 29, pp. 1440-1454, Dec. 1994.

[7] Y. Suzuki, K. Odagawa, and T. Abe, "Clocked CMOS calculator circuitry," *IEEE J. Solid-state Circuits,* vol. SC-8, pp. 462-469, Dec. 1973.

[8] H. Partovi, R. Burd, U. Salim, F. Weber, L. DiGregorio, and D. Draper, "Flow-through latch and edge-triggered flip-flop hybrid elements," in *Proc. Dig. ISSCC*, Feb. 1996, pp. 138–139.

[9] F. Klass, C. Amir, A. Das, K. Aingaran, C. Truong, R.Wang, A. Mehta, R. Heald, and G. Yee, "Semi-dynamic and dynamic flip-flops with embedded logic," in *Proc. Symp. VLSI Circuits, Dig. Tech. Papers*, June 1998, pp. 108–109.

[10] J. Tschanz, S. Narendra, Z. P. Chen, S. Borkar, M. Sachdev, and V. De, "Comparative delay and energy of single edge-triggered & dual edge triggered pulsed flip-flops for high-performance microprocessors," in *Proc. ISPLED'01*, Huntington Beach, CA, Aug. 2001, pp. 207–212.

[11] P. Zhao, T. Darwish, and M. Bayoumi, "Low power and high speed explicit-pulsed flip-flops," in *Proc. 45th Int. Midwest Symp. On Circuits and Systems Conf.*, Tulsa Oklahoma, Aug 2002, vol. 2, pp. II-477–II-480.

[12] P. Zhao, T. Darwish, and M. Bayoumi, "High-performance and low-power conditional discharge flip-flop," *IEEE Transc. on Very Large Scale Integration (VLSI) Systems*, vol. 12, pp. 477–484, May 2004.

[13] W. Chung, T. Lo, and M. Sachdev, "A comparative analysis of low-power low-voltage dual-edge-triggered flip-flops," *IEEE Transc. on Very Large Scale Integration (VLSI) Systems*, vol. 10, pp. 913–918, Dec 2002.

[14] K. S. Yeo, and K. Roy, *Low-Voltage, Low-Power VLSI Subsystems*, McGraw-Hill, USA, 2005.

[15] K. Roy and S. Prasad, *Low-Power CMOS VLSI Circuit Design*, Wiley-Interscience, USA, 2000

[16] Veendrick, H.J.M., "Short-circuit dissipation of static CMOS circuitry and its impact on the design of buffer circuits" *IEEE Journal of Solid-State Circuits, Aug 1984*, Volume 19, Issue 4, Page(s):468 – 473

[17] M. W. Bern, "Two Probabilistic Results on Rectilinear Steiner Trees", *ACM Symposium on Theory of Computing,*, 1986, pp. 433-441.

[18] Samuel Fung, *et al.*, "65nm CMOS High Speed, General Purpose and Low Power Transistor Technology for High Volume Foundry Application", *2004 Symposium on VLSI Technology Digest of Technical Papers*, June, 2004, pp. 92 – 93.

[19] Martin Saint-Laurent, "Modeling and Analysis of High-Frequency Microprocessor Clocking Networks", Ph.D. Dissertation, Georgia Institute of Technology, 2003.

Reducing Flip-Flop Power for DSP Design

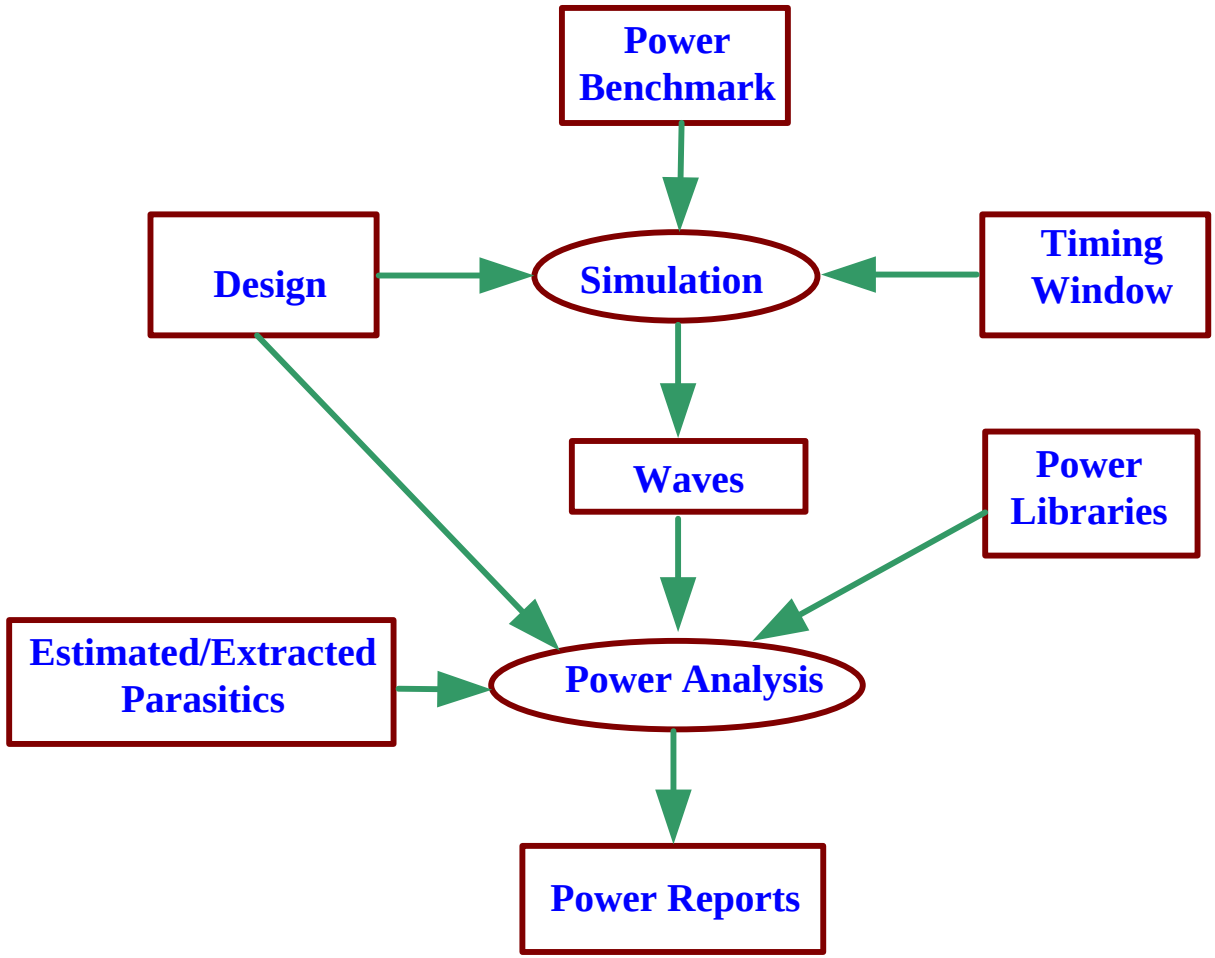
Bassam Mohd Martin Saint-Laurent Paul Bassett Shahid Imam

DSP Design Center
Qualcomm, Inc.

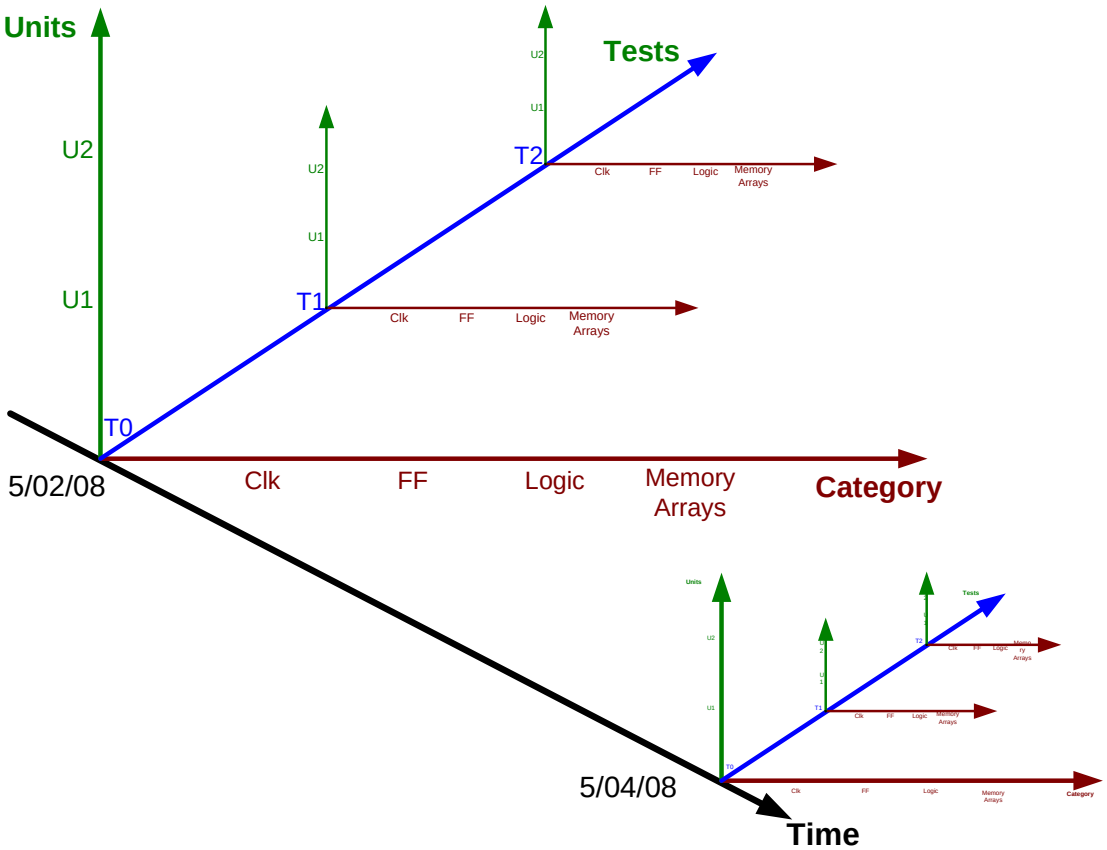
Outline

- Power estimation & analysis
  - Examine total power and FF power
- FF Power Reduction
  - Discuss couple of techniques
- Modeling the impact of enables on clock power

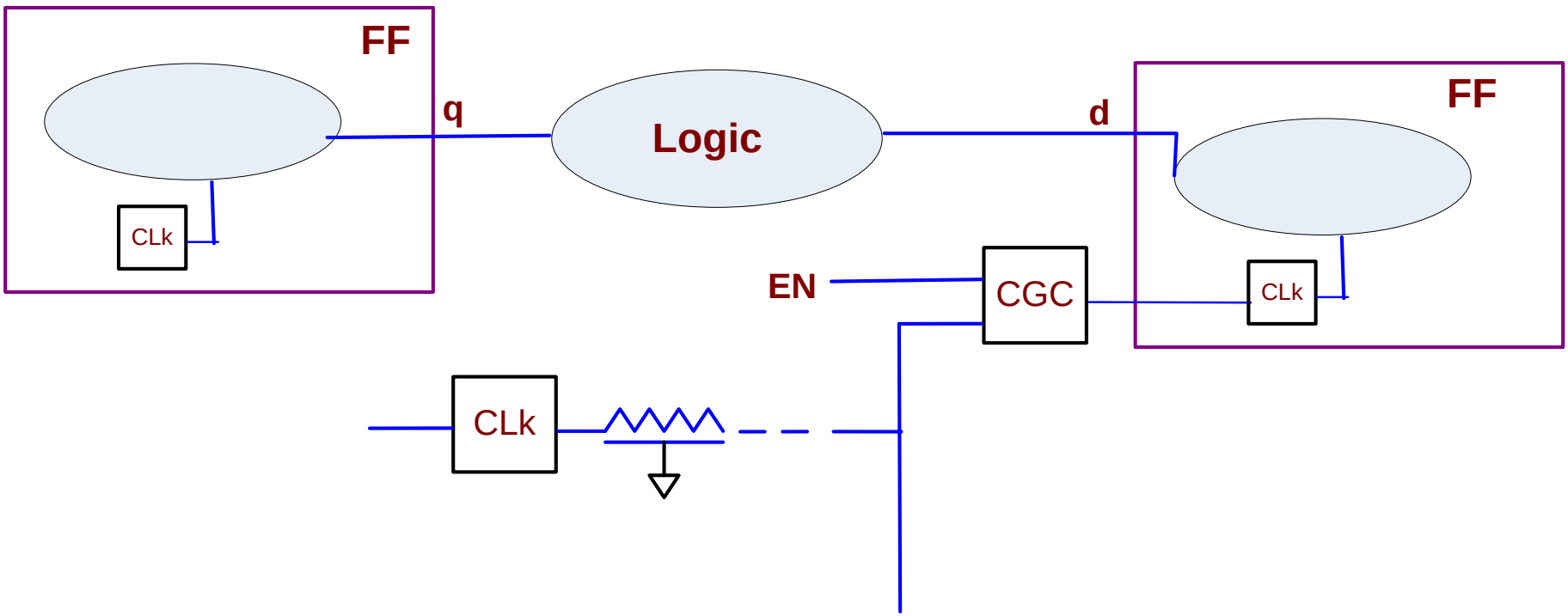
Power Estimation Flow



Power Analysis



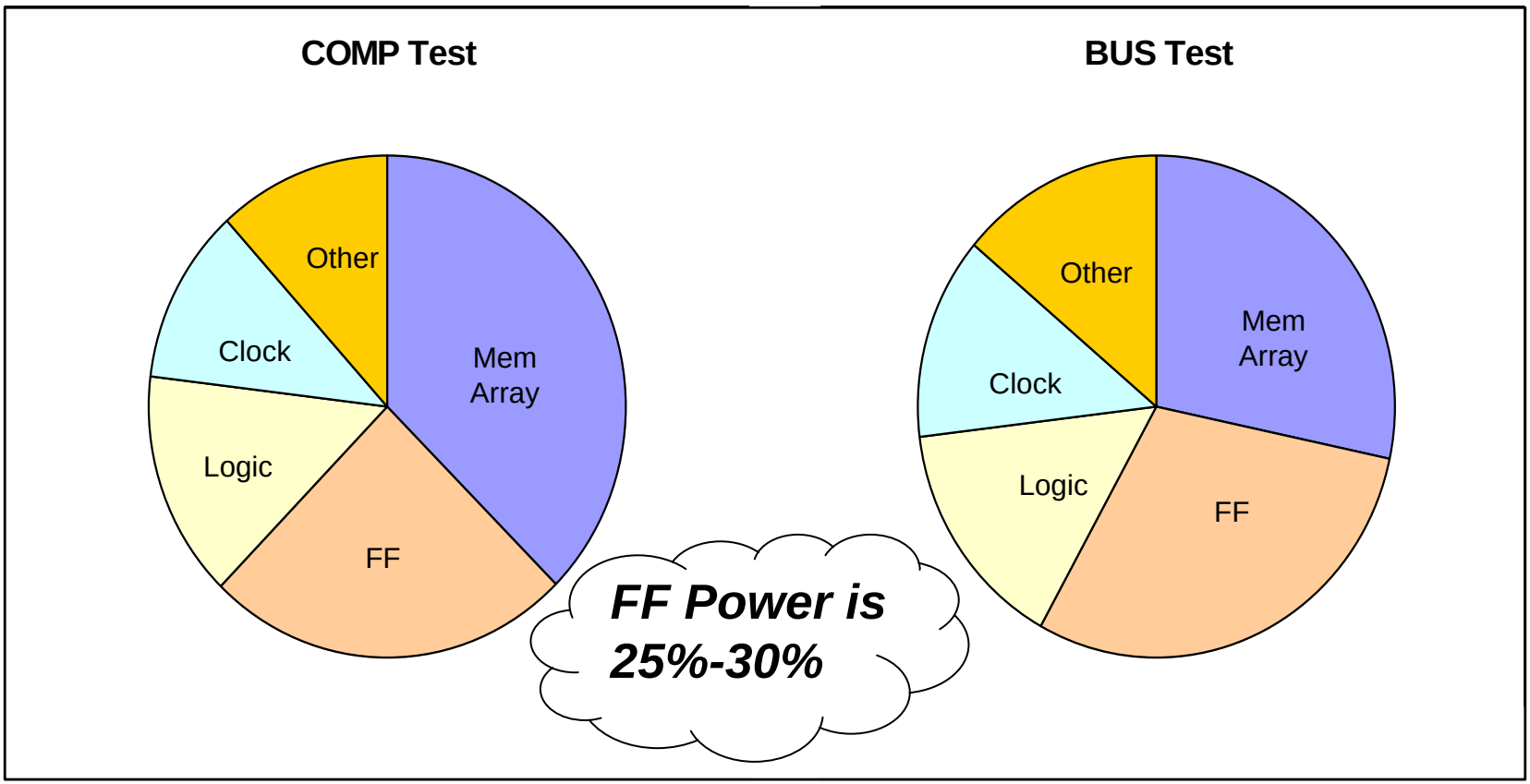
FF Power Category



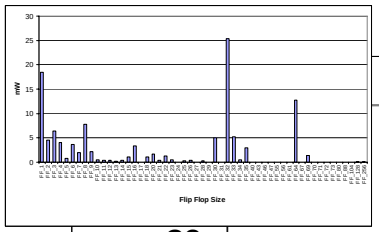
Power Analysis

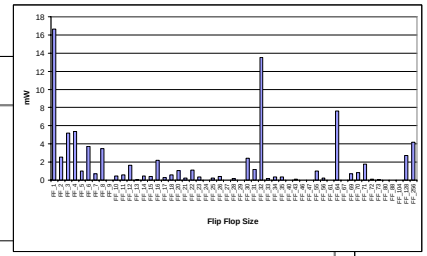
- Two tests are used for power analysis in this presentation.
  - COMP Test
    - Math intensive
  - BUS Test
    - Lots of Load/Stores

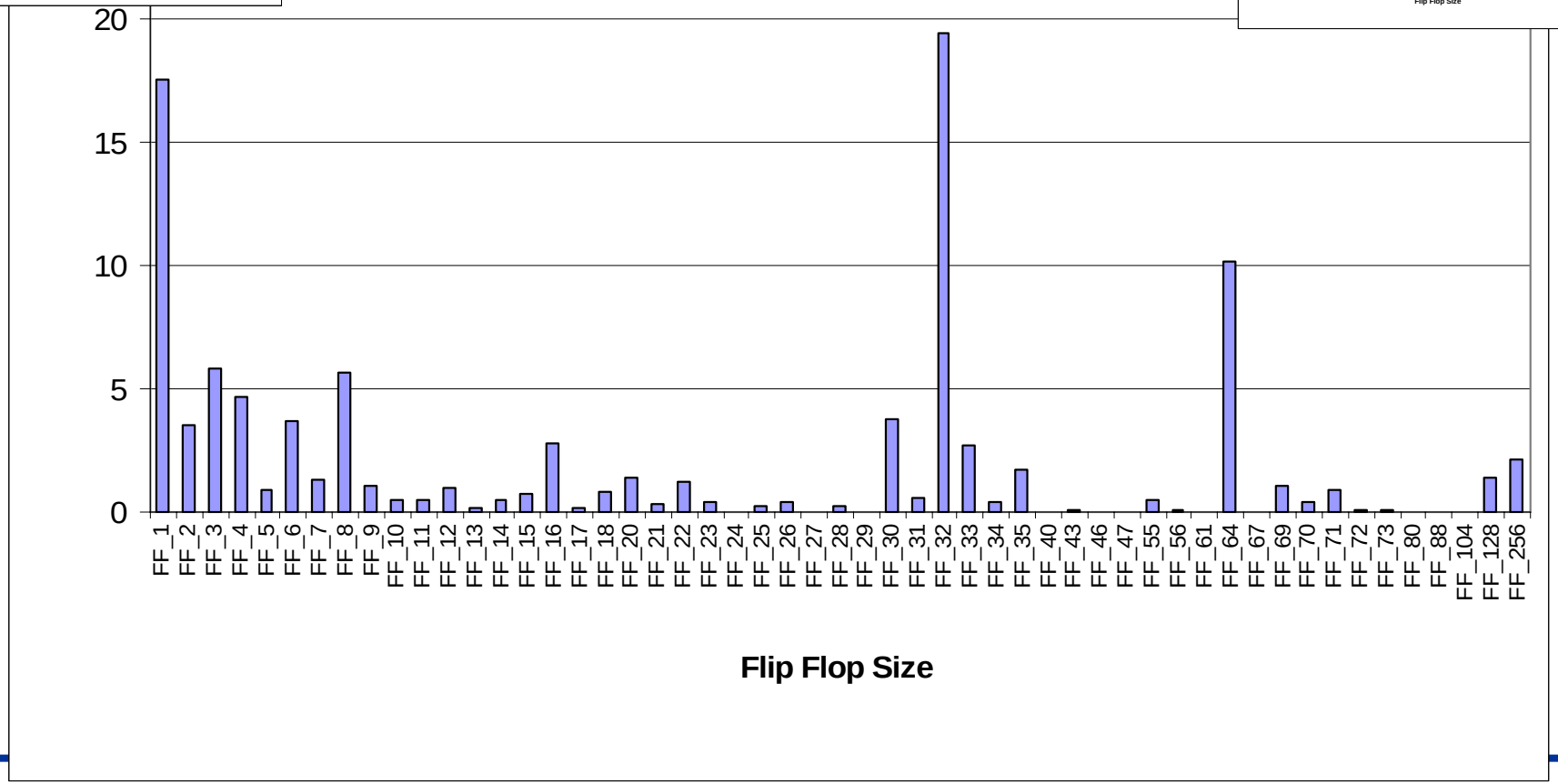
Total Power across Logic Types



FF Spectrum


mW
Flip Flop Size


mW
Flip Flop Size


20
15
10
5
0
FF_1
FF_2
FF_3
FF_4
FF_5
FF_6
FF_7
FF_8
FF_9
FF_10
FF_11
FF_12
FF_13
FF_14
FF_15
FF_16
FF_17
FF_18
FF_20
FF_21
FF_22
FF_23
FF_24
FF_25
FF_26
FF_27
FF_28
FF_29
FF_30
FF_31
FF_32
FF_33
FF_34
FF_35
FF_40
FF_43
FF_46
FF_47
FF_55
FF_56
FF_61
FF_64
FF_67
FF_69
FF_70
FF_71
FF_72
FF_73
FF_80
FF_88
FF_104
FF_128
FF_256
Flip Flop Size

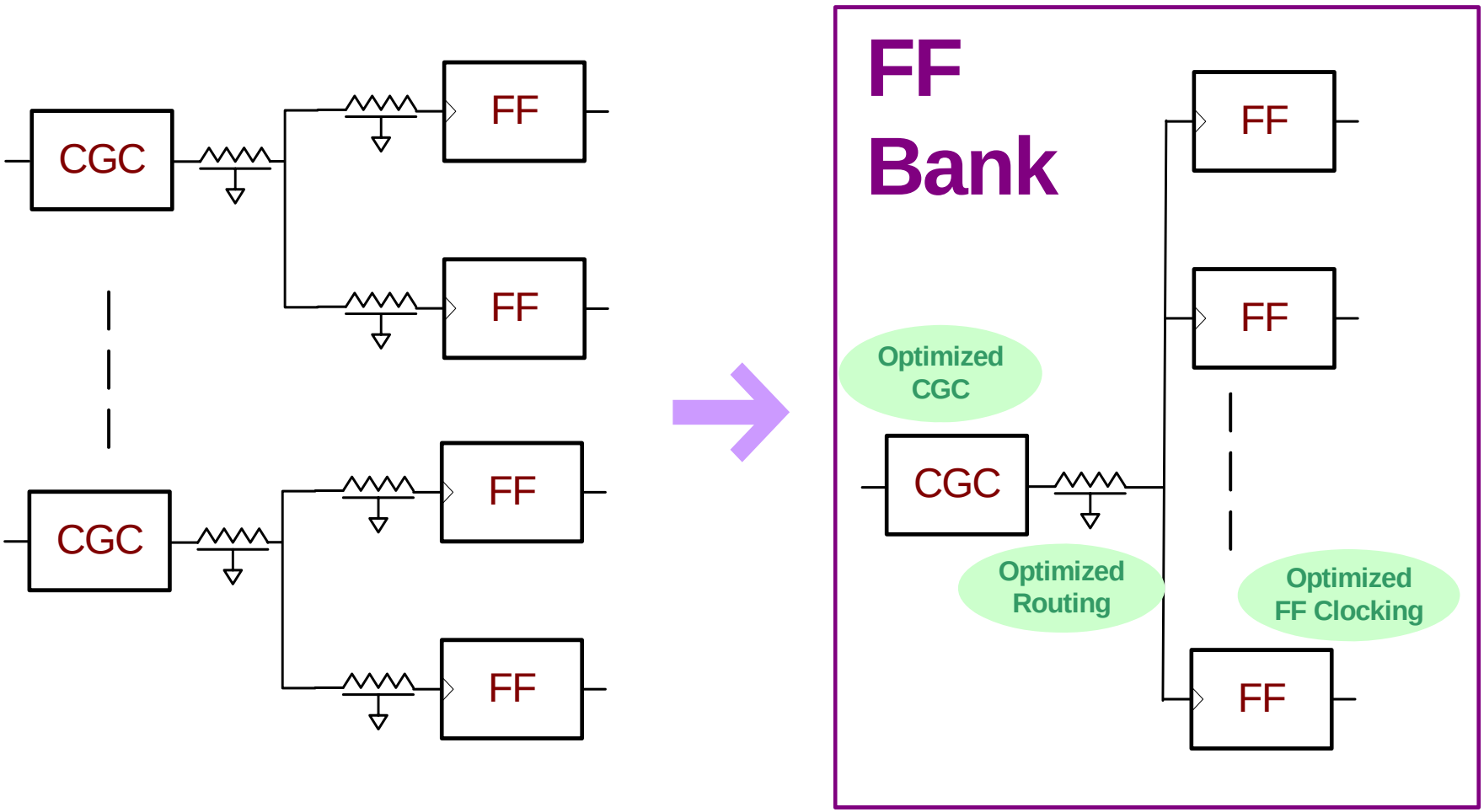
Analysis and Results

- Data FF
  - 32% of FF power is dissipated by FF size 16-bit, 32-bit and 64-bit
  - The 32-bit FF is the most power consuming FF

Analysis and Results

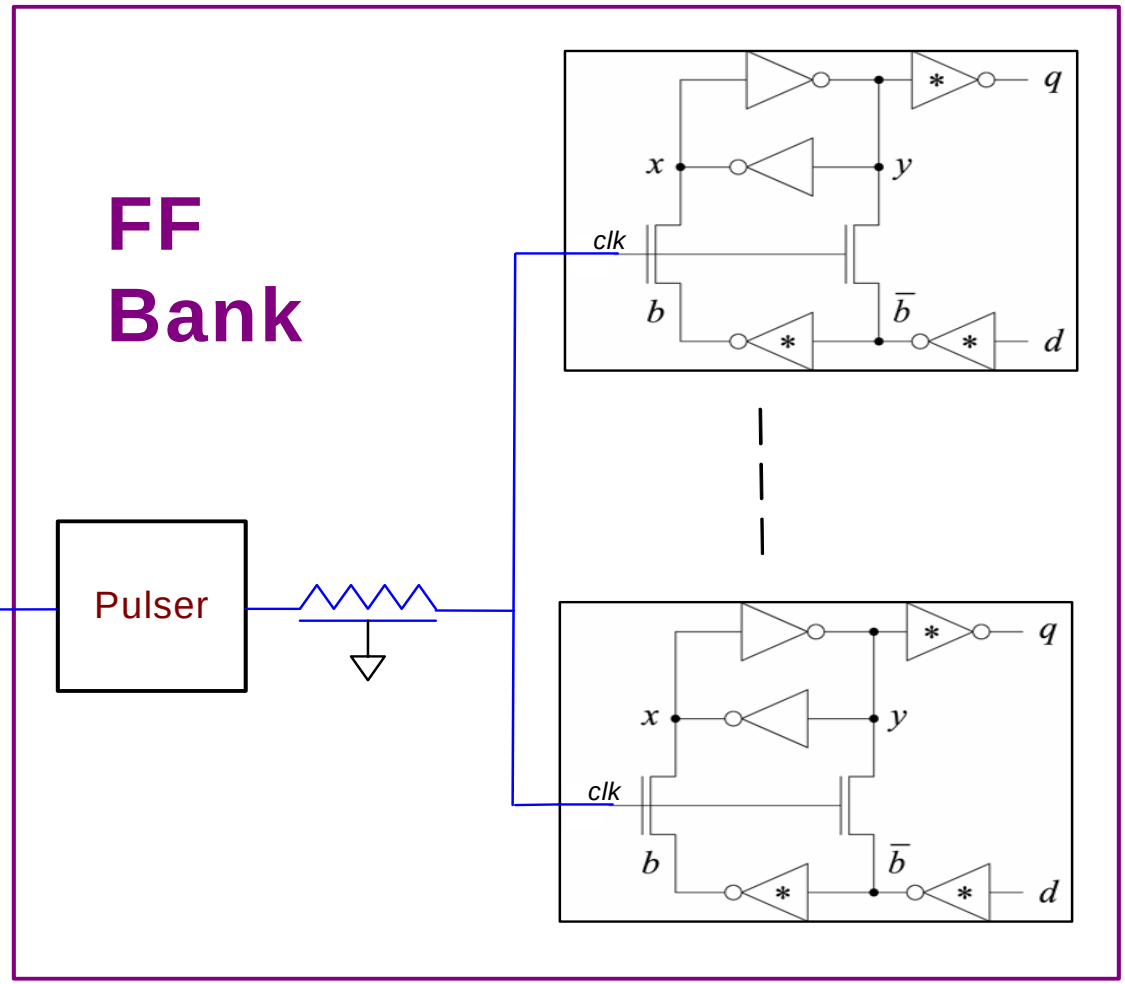
- Control FFs
  - 43% of FF power is consumed by FF sized 8-bit or less.
  - 1-bit FF is the second most consuming FFs.
  - These are mostly *control FFs* in statemachines and small counters.

FF Bank


CGC
FF
FF
CGC
FF
FF
FF
Bank
FF
FF
Optimized
CGC
CGC
Optimized
Routing
Optimized
FF Clocking
FF

FF Bank

- FFBank impalements pulser which uses improved write port topology.
- Clock power can be reduced by a factor of 4.



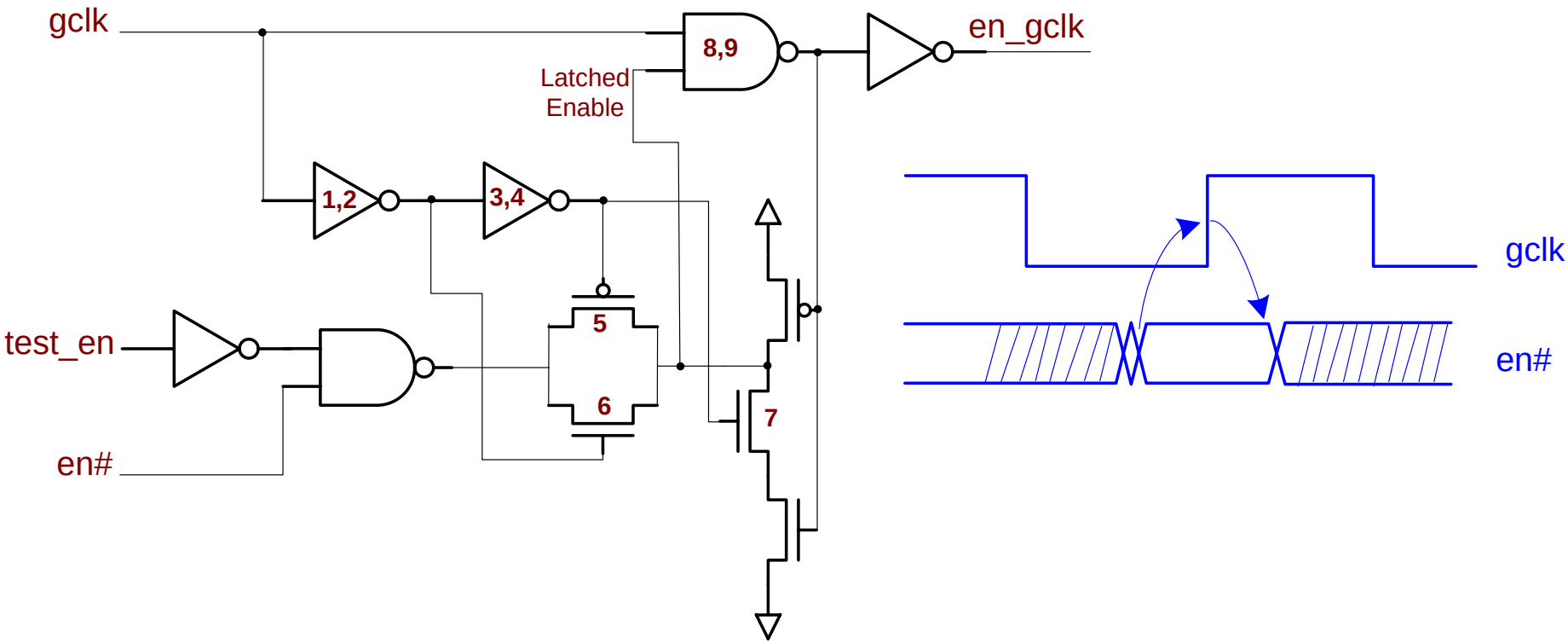
FF Bank Results

	Without FF Bank	**With FF Bank**
COMP	X	$0.94 * X$
BUS	Y	$0.96 * Y$

FFBank Size

- What is the size of designed FFbank
  - For data: study the FF spectrums
  - For control:
    - Use the most smallest FFbank.
    - The challenge is to merge the control bits in one FFbank.

Conventional Clock Gating Cell (CGC)

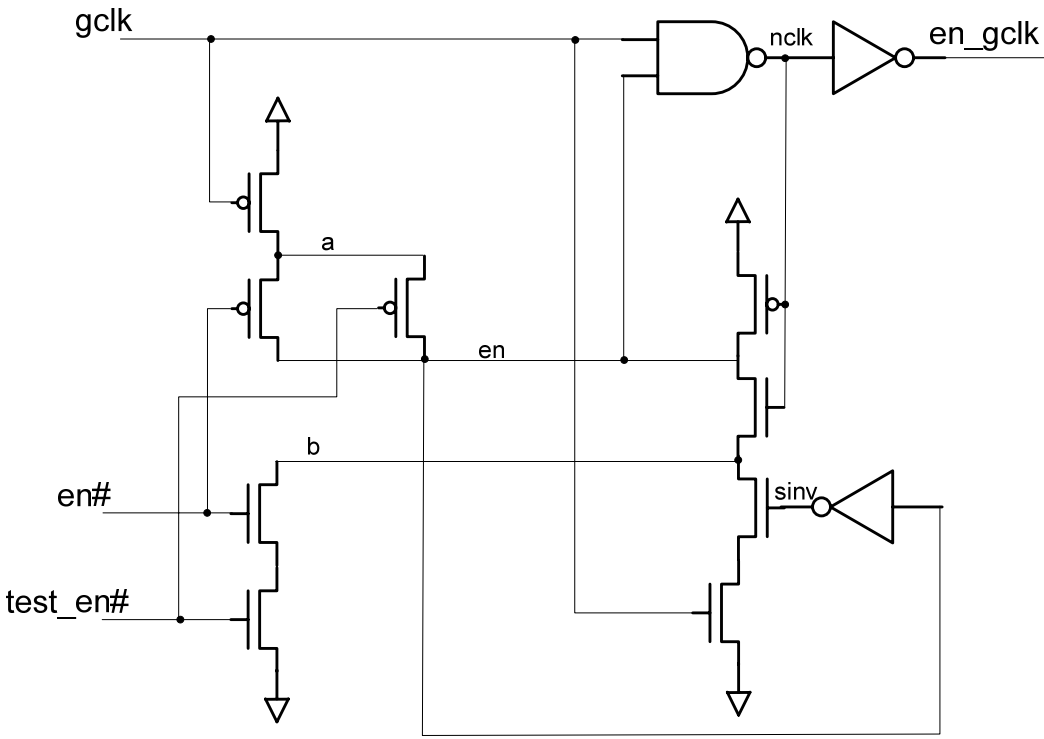


Conventional CGC

When the clock toggles, 9 transistors always toggle.

Low Power CGC

Compared with Conventional CGC, Low Power CGC:	
Transistors Always Toggle	4 (9)
When enabled, less power by	7%
When disabled, less power by	66%
Area is less by	10%
Set up time (en#) is worse by	2 INV Delay



Clock Gating: when it does not help

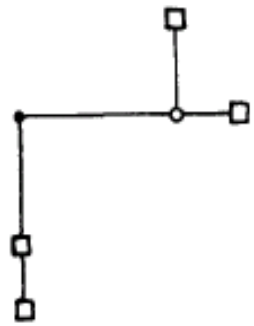
- An attempt to model a real design issue where gated clocks ended up increasing the power.
- What is the relationship between number of enables, enable activity, ….etc.

Clock Gating, cont.

- The length *L* of a minimal rectilinear Steiner tree (MRST) connecting *N* loads uniformly distributed on a square of area *A* is

$$L = \beta\sqrt{NA}$$

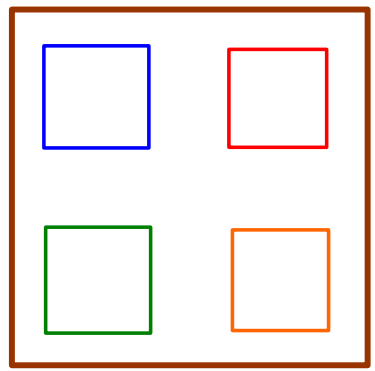
$$\beta = 0.73$$



Clock Gating, cont.

- The effective interconnect length switching per cycle in a particular domain *i*

$$L_i = \alpha_i \beta \sqrt{A_i N_i}$$



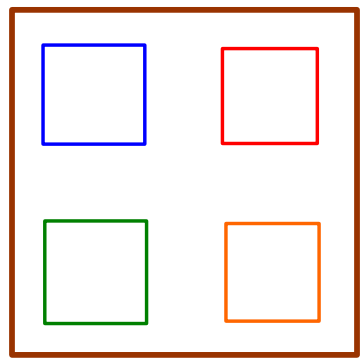
$$L_{eff} = \sum_{i=1}^{E} L_i = \beta \sum_{i=1}^{E} \alpha_i \sqrt{A_i N_i}$$

Clock Gating, cont.

- If all the *E* clock domains have the same area A_x and the same number of loads, then $N_i = N / E$

where $$L_{eff} = \alpha_{avg} \beta \sqrt{A_x NE}$$

$$\alpha_{avg} = \frac{1}{E} \sum_{i=1}^{E} \alpha_i$$



Clock Gating, cont.

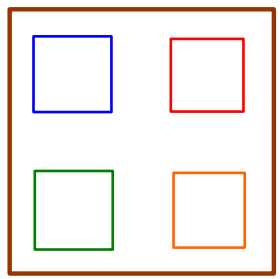
- It is interesting to note that the interconnect length that is effectively switching increases fairly slowly with the number of enables (*E*).

$$L_{eff} = \alpha_{avg} \beta \sqrt{A_x NE}$$

Clock Gating, cont.

- Doing more clock gating tends to reduce a_{avg}, but:
  - It does increases the number of domains (*E*).
  - It also tends to increase the number of loads because additional CGCs are required.

$$L_{eff} = \alpha_{avg} \beta \sqrt{A_x NE}$$



Clock Gating, cont.

- The optimum solution depends on balancing all factors: *E*, a_{ave} and A_x
  - When an enable is introduced, does it offset its cost?

$$L_{eff} = \alpha_{avg} \beta \sqrt{A_x NE}$$

Conclusion

- Presented power analysis flow and results
- Techniques to reduce FF power
- Modeling the impact of enables

Adaptive Voltage Tuning for Dual-Vdd ASICs

Stephen Bijansky Adnan Aziz

Electrical and Computer Engineering
The University of Texas at Austin

ABSTRACT

Modern CMOS manufacturing processes have significant variability, which necessitates guard banding to achieve reasonable yield. We study an ASIC architecture with a dual voltage supply wherein the supply voltage for individual blocks can be assigned after fabrication; this yields a mechanism for fixing chips that fail because of manufactured transistors being slower than designed. The fundamental advance our work makes is that we assign voltages based on manufactured data rather than designed values.

Keywords

Process Variation, Tuning, Configurability, Yield, Delay

1. INTRODUCTION

1.1 Process Variation is Real

Process variations originate from many sources, including lithography, substrate doping, and chemical-mechanical polishing [1, Ch. 14]. Variations can be characterized as either systematic or random. Systematic variations are then broken down into wafer-to-wafer variations, die-to-die variations, and within-die variations. There have been many studies on the effects of variations. One representative study is Borkar *et al.* [2], which showed that for a batch of microprocessors all on the same wafer, variations in transistor channel length and variations in threshold voltage contributed to a 30% difference in chip frequencies. Borkar also reports that the standby leakage current varied by as much as $20\times$. The variation in leakage current is particularly significant since leakage continues to increase in importance in recent designs—in a dual core 65 nm Xeon processor, leakage accounted for 30% of the total power [3].

An increase in the number of variation sources has led to even more corner cases that need to be simulated for each design. Nassif [4] and the ITRS [5] have estimated that the $3\sigma/\mu$ variation for 65 nm process technology is as high as 30% for transistor channel length and transistor threshold voltage. Future processes could have even larger amounts of variation.

1.2 Contributions

The key contributions of our work are a CAD methodology and a quantitative study using realistic data on the latest process technologies of the impact of post-manufacturing tuning on yield and power for dual-Vdd ASICs. We find that, for a representative modern process, post-manufacturing tuning increases the yield and decreases power compared with a conventional dual-Vdd design that selects the voltage supply pre-manufacturing.

Broadly speaking, our approach differs from previous work in that we tune individual chips after fabrication. Therefore, this tunable chip is able to respond to actual chip variations instead of estimated values. After the chip is fabricated, a series of BIST tests are performed to determine which blocks will use a high voltage supply and which blocks will use a low voltage supply. Since this tuning and configuration takes place after manufacturing, designers will still be able to use all of the current tools and techniques to which they are accustomed.

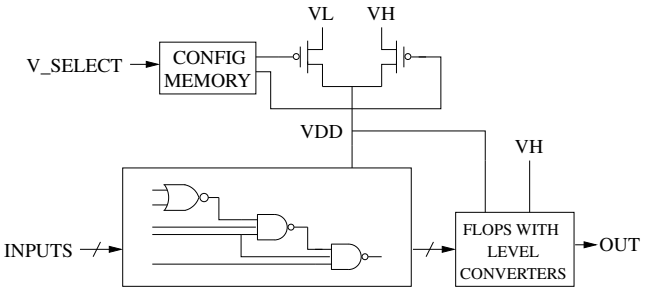


Figure 1: Tunable dual-Vdd block with configurable voltage supply PMOS transistors. The standard cell gates are connected to the local Vdd. VH is connected to the high supply voltage grid, and VL is connect to the low supply voltage grid. The flops includes a level converter. Therefore, the block output is always high voltage.

2. DUAL VOLTAGE DESIGN

We focus on tuning using a dual voltage supply because changes in the supply voltage have such a large impact on both the speed and power consumption of the chip. Recall that by selecting a higher supply voltage, the chip will operate faster but will use more power; a lower supply voltage leads to a slower chip that uses less power. The designer can use the lower voltage supply for logic on paths that easily meet their delay constraints. These non-critical paths can therefore run slower and use less power.

In a dual voltage design style, there are two voltage grids across the entire chip. Every gate is partitioned into a block of spatially related gates. Each block of gates is connected to the appropriate supply grid through a pmos pass transistor. By having only one pmos pass transistor turned on at a time, each block can have one of two supply voltage choices.

Since there will be multiple voltage supplies on the chip, level converters are required at all junctions between low voltage outputs and high voltage inputs. In its simplest form, a level converter is a buffer with additional transistors to prevent short circuit current. This work incorporates the level converters into the flip-flops needed for each pipeline stage.

2.1 Tunable Block of Gates

Figure 1 shows the tunable block that we designed. This block uses normal standard cell gates and flops with level converters. All of the gates are connected to a common Vdd that can be switched between high voltage and low voltage depending on on the voltage select configuration memory. The flops with level converters always have a high voltage output, and they accept either low voltage or high voltage inputs. The common Vdd supply voltage configuration is stored in a register with complementary outputs so that only one pmos pass transistor can be turned on at a time. In this design,

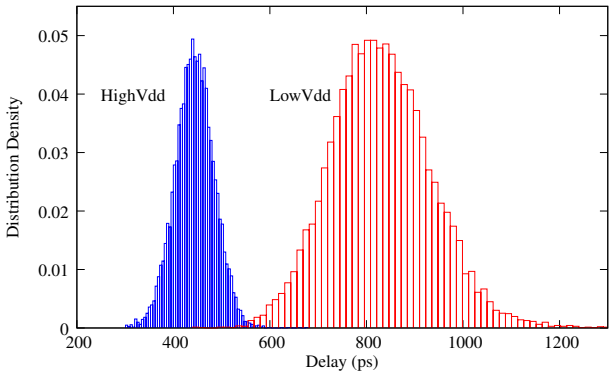


Figure 2: Dual-Vdd delay distribution for a single block

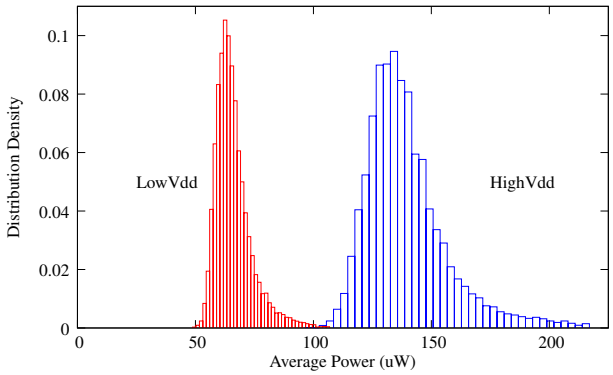


Figure 3: Dual-Vdd total power distribution for a single block

the supply voltage configuration is programmed before normal operation of the block begins.

2.2 Area Overhead

In order to size the pmos pass transistor, a SPICE simulation sweep was performed. The width was varied from $10\times$ minimum size to $300\times$ minimum size. The chosen pass transistor size of $133\times$ minimum size resulted in less than a 5% increase in block delay compared to having no pass transistor. After layout, the area of one pass transistor was comparable to a standard cell gate. In situations in which total area and power are more important than delay, a smaller pass transistor can be used.

The STR6 level converter [6] is composed of a buffer with 6 additional small transistors. These 6 additional transistor have about the same transistor width as a buffer. Since flops internally already have a buffer structure, the STR6 level converter can be incorporated with minimal area overhead. Therefore, the overall area cost of the level converter is less than scan logic already built into many current flops.

3. BLOCK DELAY STATISTICS

We performed Monte Carlo simulations on the dual-Vdd block shown in Figure 1. The 65 nm Berkeley PTM model [7] was used for the SPICE transistor models. For each transistor in the block, both the transistor length and threshold voltage were modeled as uncorrelated Gaussian distributions with the $3\sigma/\mu$ variation chosen to be 20% [8, 9]. The high voltage supply was set to 1.2 V and the low voltage supply was set to 0.8 V. The simulation temperature was set to 85 °C.

The block was design to be a full adder cell that will be used for the experiments in Section 5. We measured the performance of our block in 100,000 Monte Carlo simulations using HSPICE. The block was simulated for the worse-case delay. Delay was measured from the 50% crossing of the primary input signal to the 50% crossing of the input to the flop.

In Figure 2, the worse-case output transition delay distribution is shown for all 100,000 trials. In this figure, the bars on the left represent the delay when all the block gates are connected to the high voltage supply through the pmos pass transistor. Then, the block of gates is disconnected from the high voltage supply and connected to the low voltage supply through the other pmos pass transistor. The bars on the right represent the delay when all of the block gates are connected to the low voltage supply. In order to present an equal comparison, the number of bars on the left is equal to the number of bars on the right. This graph shows that there is a large spread of delay values for this block when intra-chip variation is taken into account. Moreover, the low-Vdd distribution has a larger amount of variation.

The average power was measured from when the primary input changes by 1% until the input of the flop reaches 99% of its final value. The power includes both switching and leakage power for the block gates and the power transistors. Figure 3 show the total power values; the power spreads are similar to the delay spreads. There are also similar graphs for only leakage power.

Taken together, these graphs show that there is potential to reduce both dynamic power and leakage power. The following summarizes the potential power savings for a single block when using a dual-Vdd approach:

1. The delay increases from a mean of 444 ps using high-Vdd to 831 ps using low-Vdd.

2. The power decreases from a mean of 140 μw using high-Vdd to 70 μw using low-Vdd.

3. The leakage power decreases from a mean of 25.7 μw using high-Vdd to 15.7 μw using low-Vdd.

If the designer is able to decrease the speed on non-critical path blocks by switching those blocks to a low voltage supply, power can be reduced. Even with blocks on the critical path, if there is enough slack, some of those blocks could use a low supply voltage while the other blocks use a high supply voltage. Overall, there is potential to reduce total power by almost 50%.

4. VOLTAGE ASSIGNMENT

After the die is manufactured, voltage assignment is performed using an on-chip BIST based algorithm. First, all of the blocks are assigned to use a high-Vdd supply. Then, a pass-fail BIST is performed to determine if the chip met the delay target. If the chip fails the BIST test, then the chip is marked as failing. Otherwise, an iterative assignment algorithm powers down blocks while checking that delay target is still being met.

Associated with each design is an ordered block criticality list. Using nominal pre-manufacturing transistor parameters, each block is given a criticality rating based on the amount of timing slack for that block. A block with a large amount of slack is a good candidate for being powered down. Therefore, the tuning begins by selecting the block with the largest amount of timing slack. The selected block is assigned to use a low-Vdd supply. Next, the pass-fail BIST is performed again to determine if the chip still meets the delay target. If the delay target is met, the selected block is permanently assigned to use the low-Vdd supply. Otherwise, if the delay target is not met, the selected block is permanently assigned to use the low-Vdd supply. This process is repeated until all of the blocks have been assigned to use either a high-Vdd or a low-Vdd voltage supply.

Table 1: 32-bit multiplier using 4 different voltage assignments. The multiplier was instantiated on 10,000 chips, each of which have independent variations. Tunable: post-manufacturing voltage assignment. All High: every block is connected to the high-Vdd supply. All Low: every block is connected to the low-Vdd supply. Guard Band: pre-manufacturing dual-Vdd voltage assignment with a 10% guard band.

Target Delay (ns)	Yield (%)				Average Power (mW)				Average High-Vdd Blocks (% of Total)			
	Tunable	All High	All Low	Guard Band	Tunable	All High	All Low	Guard Band	Tunable	All High	All Low	Guard Band
600	50	50	0	50	143.5	143.5	n/a	143.5	0	0	n/a	0
723	100	100	0	100	143.5	143.5	n/a	143.5	0	0	n/a	0
846	100	100	0	100	143.5	143.5	n/a	143.5	0	0	n/a	0
969	100	100	0	0	128.3	143.5	n/a	n/a	21	0	n/a	n/a
1092	100	100	0	0	84.8	143.5	n/a	n/a	81	0	n/a	n/a
1215	100	100	9	9	74.0	143.5	71.4	71.4	96	0	100	100
1339	100	100	50	50	72.2	143.5	71.4	71.4	98	0	100	100

5. EXPERIMENTS

We used a 32-bit by 32-bit multiplier to study the advantages of post-silicon adaptive blocks. The multiplier was constructed using carry save full adders followed by a ripple carry vector merging step [10]. This experiment includes only datapath logic. This multiplier was designed at the schematic level using Cadence. The schematic was converter to verilog for functionality testing. The verilog was placed and then partitioned into blocks.

The multiplier was instantiated on 10,000 chips using the block data from Section 3. Each block has independent variations. The BIST algorithm of Section 4 was simulated for each chip for a range of delay targets. The delay, power, and percentage of high voltage blocks was recorded for each chip.

5.1 Setup for Experiments

Results are summarized in Table 1. Key details about the experimental setup are as follows:

- Our implementation of pre-manufacturing voltage assignment has a 10% guard band. This guard band is discussed in more detail in Section 5.3.
- Yield, power, and the percentage of high-Vdd supply blocks are calculated for each target delay.
- The range of target delays was chosen such that the all high-Vdd has a yield of 50% at the fastest target delay and the all low-Vdd has a yield of 50% at the slowest target delay.

5.2 Experimental Results

The key take-aways from the experimental results in Table 1 are as follows:

- Compared to High, Tunable had exactly the same yield; this is to be expected since Tunable can set all blocks to high-voltage if needed. Tunable improved the power for the slower target delays by 50%.
- Compared to guard banding, Tunable had dramatically better yield and used comparable power.

Further observations from analyzing the results are as follows:

- Guard band pre-manufacturing assignment yields are not always monotonically increasing because each target delay has a custom high-Vdd map. Therefore, some target delays will have a better yield for a given assignment map than other target delays.
- Determing the best guard band for the pre-manufacturing assignment is non-trivial. For some target delays, 10% was a good value, but more guard banding increased the power.

5.3 Pre-Manufacturing Voltage Assignment

As a comparison to Tunable voltage assignments, we also experimented with a guard banding approach using pre-manufacturing dual-Vdd voltage assignment based on on the work by Li *et al.* [11]. Li's work performed voltage assignment based on power sensitivity. Our implementation of Li's voltage assignment uses the same methodology as the work in this paper except that the block delays use nominal transistor parameter values. The list of blocks that use a high-Vdd supply is then saved for individual chip configuration.

In order to improve the yield of the pre-manufacturing voltage assignment, the target delay was given a 10% guard band. For example, if the target delay after manufacturing was 5 ns, then the pre-manufacturing voltage assignment would switch enough blocks so that the design would have at most 4.5 ns of delay when using nominal transistor values. Since the cell delays are discrete values, the actual target delay would most likely be less than 4.5 ns.

6. CONCLUSION

Current designs are heavily pipelined and have many short paths. New process technologies increase the number of transistors available for a given design, which in turn leads to an increase in the number of critical paths in a chip. When short paths are combined with an increase in the total number of paths, there will be an even greater impact from process variations. Tuning a chip post-manufacturing significantly reduces the effects of that process variation. Additionally, this tuning is dependent only on post-silicon BIST testing; in particular, it is independent of the delay distribution and correlations between block delays.

We have shown experiments in which post-manufacturing tuning had a 50% power improvement compared to an all high-Vdd voltage assignment. Moreover, post-manufacturing tuning significantly improved both the yield and power compared with selecting voltages pre-manufacturing based on nominal delays, even with a 10% guard band.

7. REFERENCES

[1] D. Chinnery, *Closing the Gap Between ASIC & Custom: Tools and Techniques for High-Performance ASIC Design*. Springer, 2002.

[2] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, and V. De, "Parameter Variations and Impact on Circuits and Microarchitecture," *Design Automation Conference*, 2003.

[3] S. Rusu, S. Tam, H. Muljono, D. Ayers, J. Chang, B. Cherkauer *et al.*, "A 65-nm Dual-Core Multithreaded Xeon Processor with 16-MB L3 Cache," *IEEE J. Solid-State Circuits*, 2007.

[4] S. Nassif, "Modeling and Analysis of Manufacturing Variations," *IEEE Custom Integrated Circuits Conference*, 2001.

[5] SIA, International Technology Roadmap for Semiconductors, 2005.

[6] S. Kulkarni and D. Sylvester, "High performance level conversion for Dual Vdd esign," *IEEE Trans. VLSI Syst.*, 2004.

[7] Y. Cao, T. Sato, M. Orshansky, D. Sylvester, and C. Hu, "New Paradigm of Predictive MOSFET and Interconnect Modeling for Early Circuit Simulation," *IEEE Custom Integrated Circuits Conference*, 2000.

[8] H. Chang and S. S. Sapatnekar, "Prediction of leakage power under process uncertainties," *ACM Trans. Des. Autom. Electron. Syst.*, 2007.

[9] A. Srivastava, S. Shah, K. Agarwal, D. Sylvester, D. Blaauw, and S. Director, "Accurate and Efficient Gate-Level Parametric Yield Estimation Considering Correlated Variations in Leakage Power and Performance," *Design Automation Conference*, 2005.

[10] K. Parhi, *VLSI Digital Signal Processing Systems*. Wiley New York, 1999.

[11] Y. Lin, F. Li, and L. He, "Circuits and Architectures for Field Programmable Gate Array with Configurable Supply Voltage," *IEEE Trans. VLSI Syst.*, 2005.

Adaptive Voltage Tuning

Stephen Bijansky

Scott Lee

Adnan Aziz

UT ECE

Outline

Adaptive Voltage Tuning Overview

- Increasing process variation → parametric yield loss

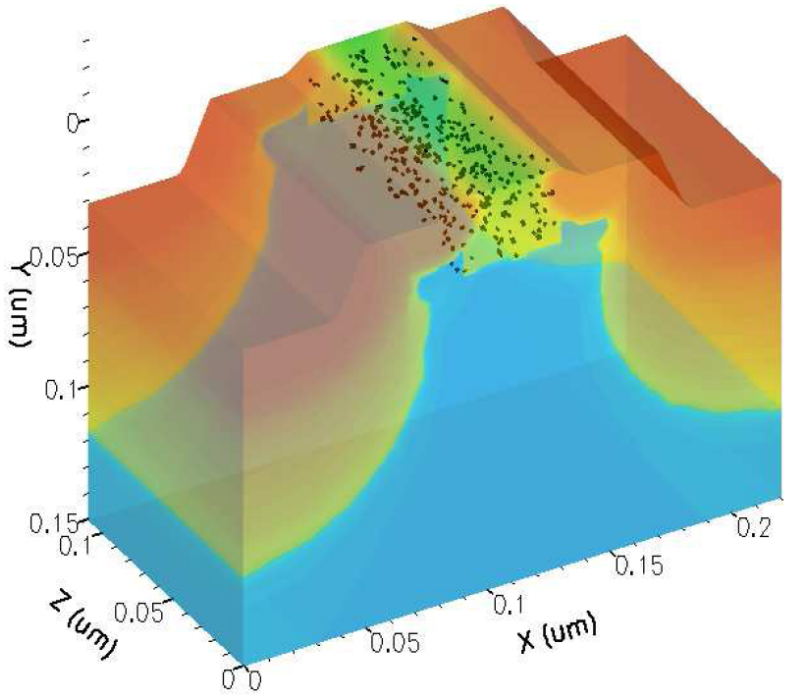


Figure: Dopant Atoms – Intel IEMD 2007

Adaptive Voltage Tuning Overview

- Increasing process variation $\longrightarrow$ parametric yield loss
- Logic design with a dual voltage supply

Adaptive Voltage Tuning Overview

- Increasing process variation $\longrightarrow$ parametric yield loss
- Logic design with a dual voltage supply
- Assign block voltage post-manufacturing

Figure: Shimano Rear Derailleur

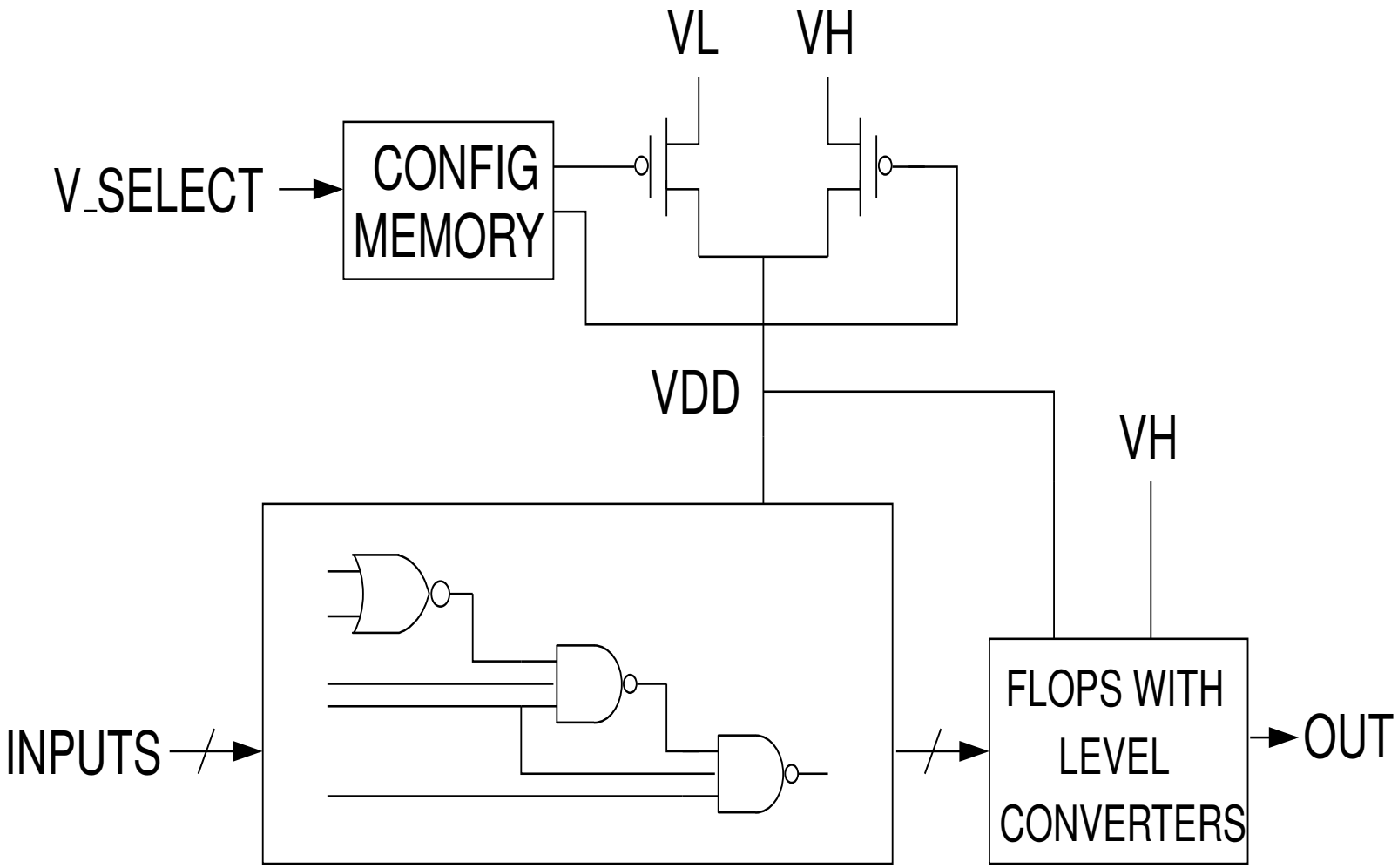
Adaptive Voltage Tuning Overview

- Increasing process variation $\longrightarrow$ parametric yield loss
- Logic design with a dual voltage supply
- Assign block voltage post-manufacturing

Assign voltages based on manufactured data

Fix chips that otherwise fail

Tunable Dual-Vdd Block



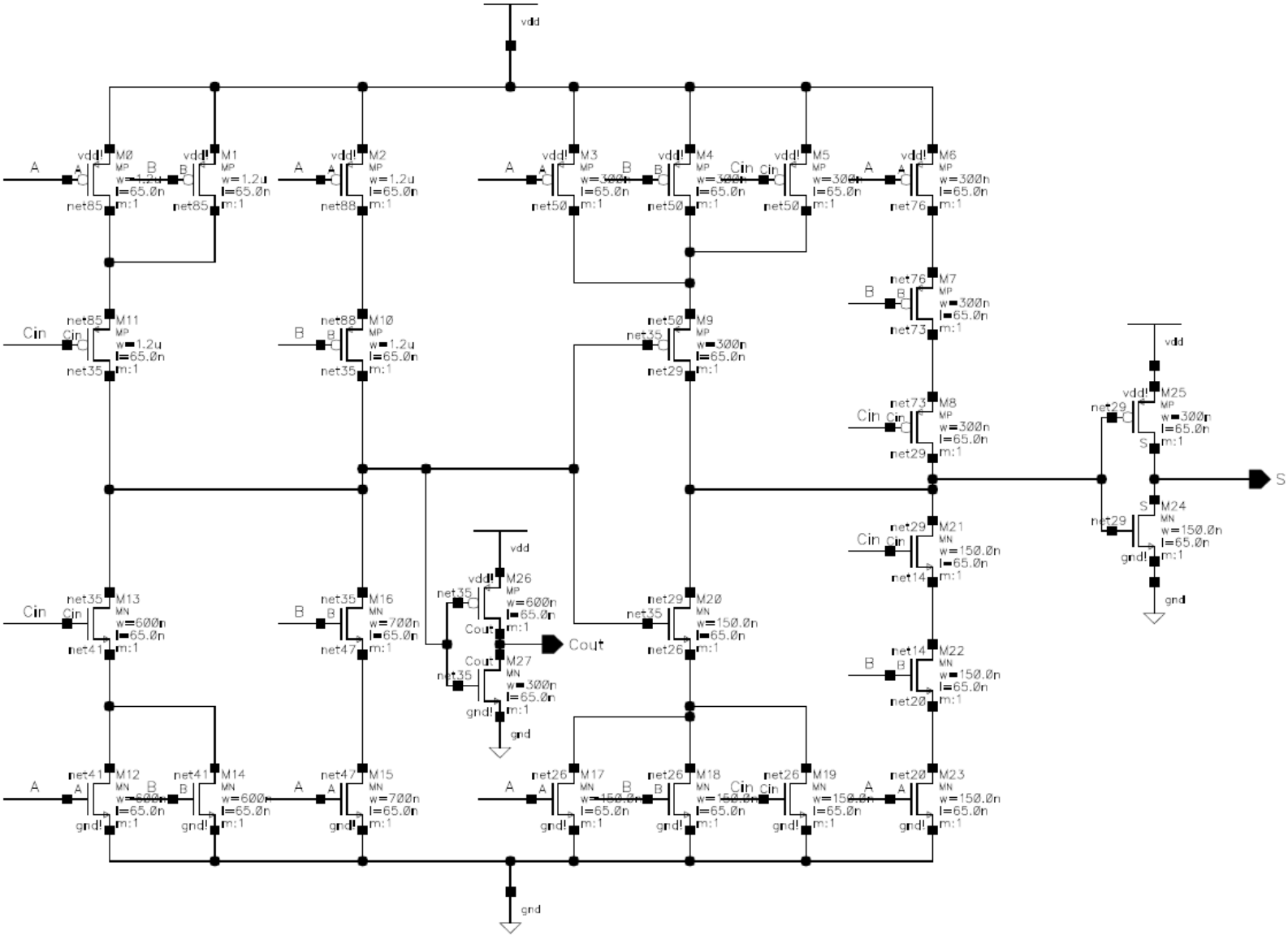
Outline

Multiplier

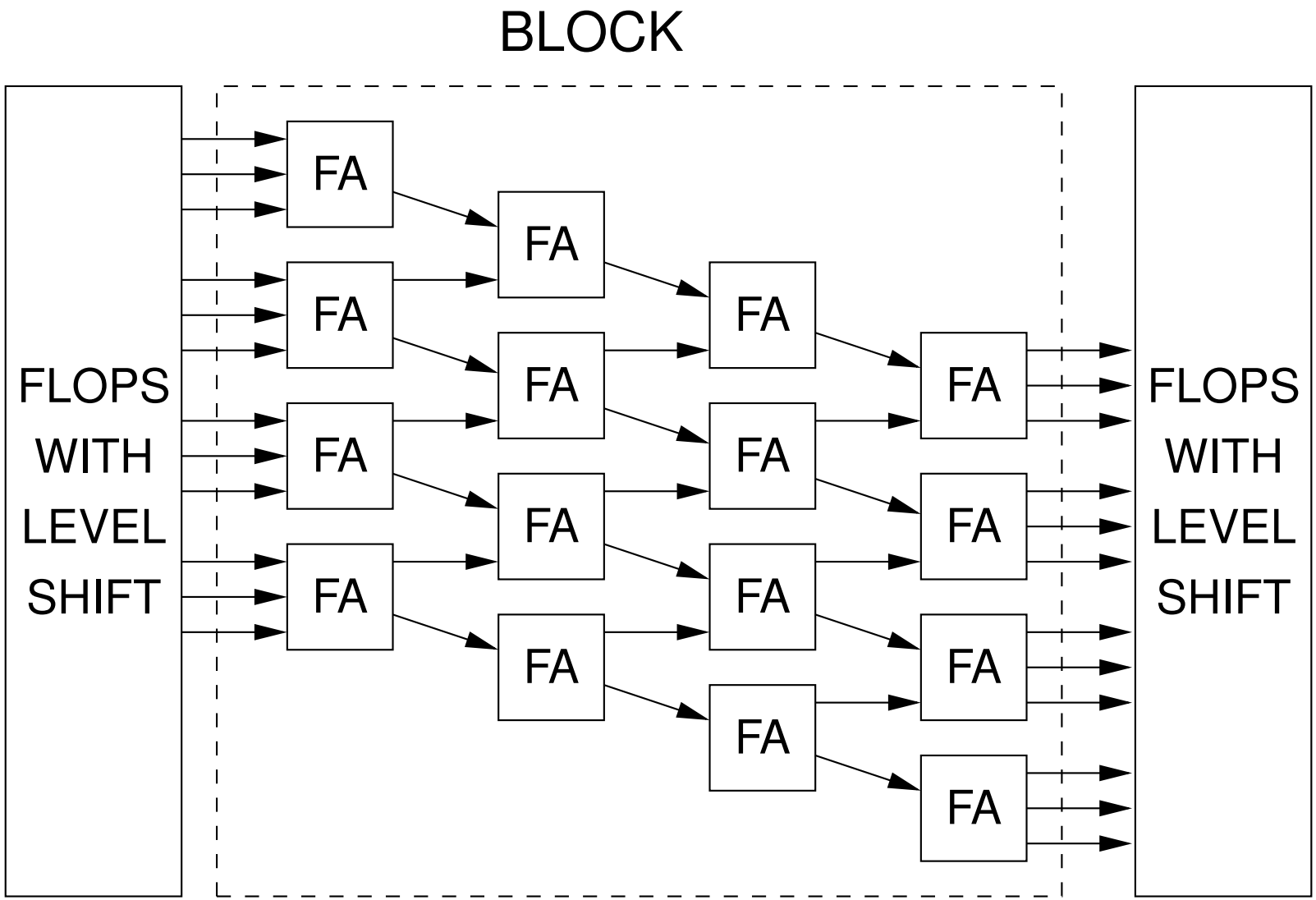
- Graphics processors have more than 320 multipliers
- Digital signal processors
- Important building block for many applications

Multiplier is a good case study

Full Adder

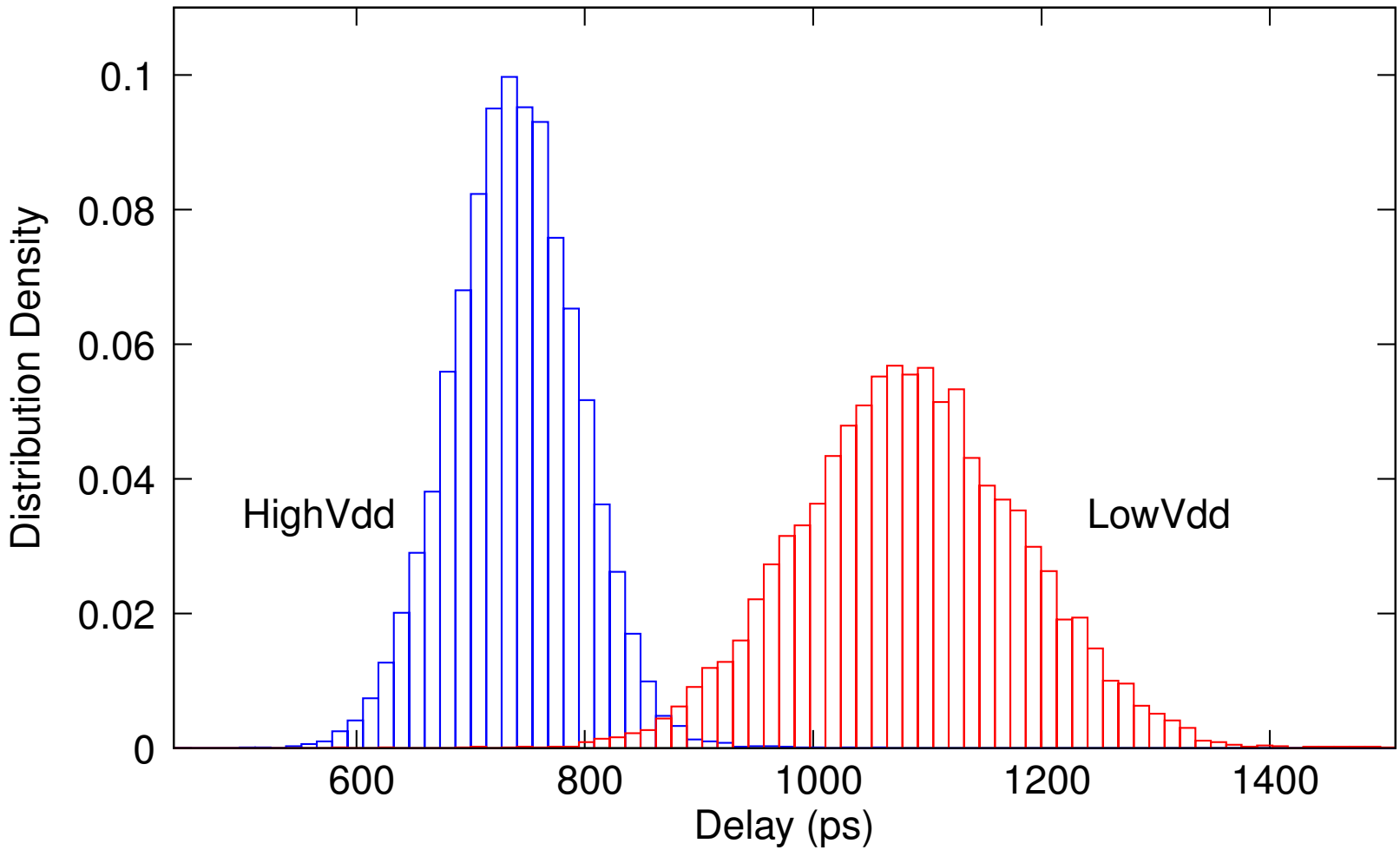


Single Block for 32x32-bit Multiplier

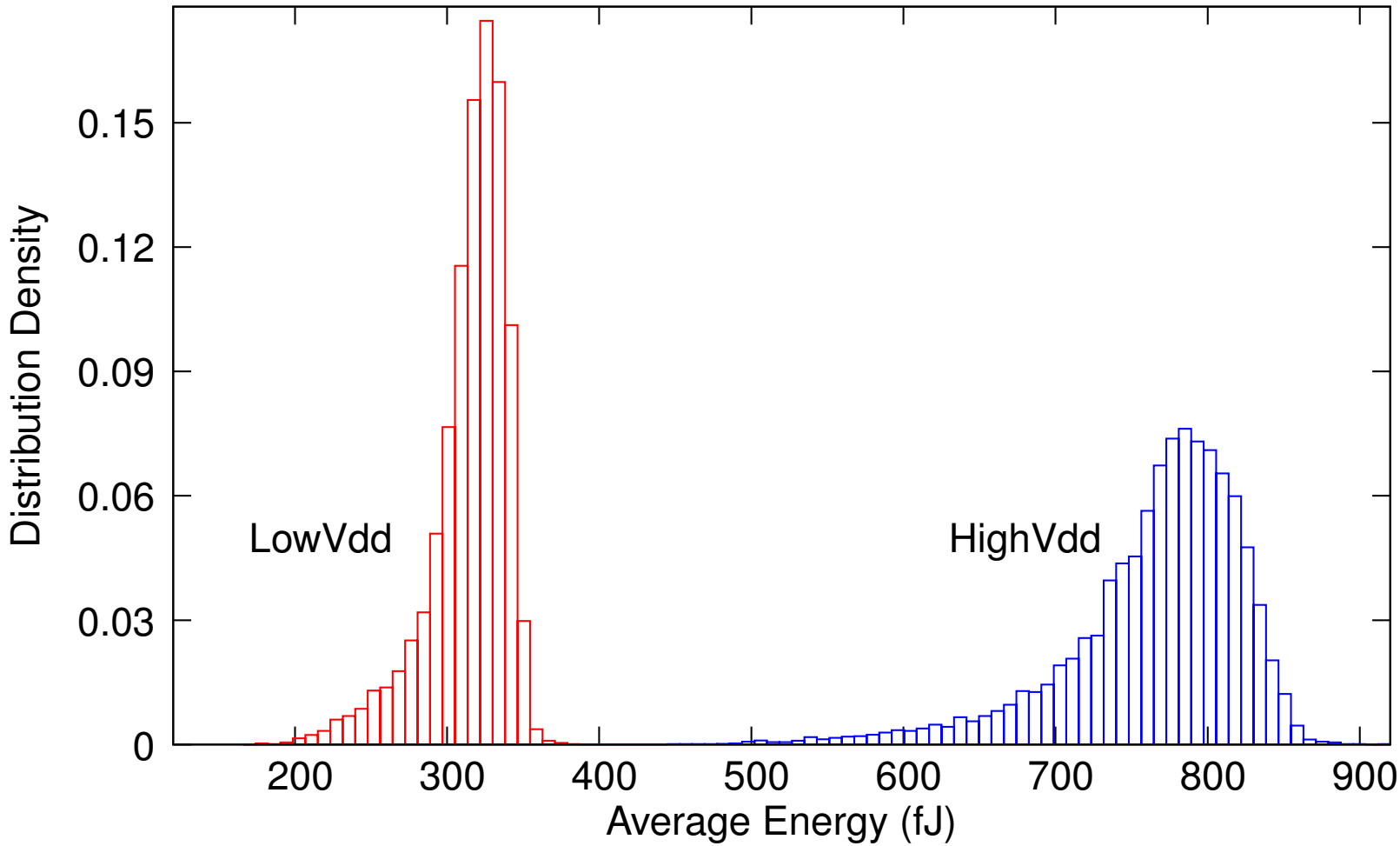


Outline

Single Block Dual-Vdd Delay Distribution



Single Block Dual-Vdd Power Distribution



Outline

Adaptive Voltage Assignment Algorithm

First determine if chip works

- All tunable blocks assigned to use high-Vdd supply
- At-speed test for delay
- Check for failing chips

Iterative adaptive voltage assignment algorithm (AVA)

- General problem is NP-hard
- Devised a greedy algorithm
- Uses at-speed testing

Adaptive Voltage Assignment Algorithm

- Each tunable block is given a criticality rating
  - Based on timing slack for that tunable block
  - Large slack $\Rightarrow$ good candidate for being powered down
  - Use nominal pre-manufacturing transistor parameters
- Select block with largest slack
  - Assign that block to low-Vdd supply
  - Perform at-speed test
  - Passing test $\Rightarrow$ keep low-Vdd
  - Failing test $\Rightarrow$ revert to high-Vdd
- Repeat for all tunable blocks

Outline

Experiments

- Simulated 10,000 multipliers
- 64 tunable blocks in each multiplier
- Independent variations for each block
- AVA assigned voltage for each block
- Compared to a fixed voltage design
- Range of target operating frequencies

Yield (%)

Target (ps)	Adapt. Tune	Fixed Voltage (No Power Transistor)				
		1.2 V	1.1 V	1.0 V	0.9 V	0.8 V
925.0	90	96	86	10	0	0
970.0	96	99	95	65	0	0
1016.0	99	100	99	89	3	0
1062.0	100	100	100	95	45	0
1108.0	100	100	100	99	80	0
1154.0	100	100	100	100	89	0.6
1200.0	100	100	100	100	95	16

Energy (pJ)

Target (ps)	Adapt. Tune	Fixed Voltage (No Power Transistor)				
		1.2 V	1.1 V	1.0 V	0.9 V	0.8 V
925.0	47.6	52.6	43.0	34.6	n/a	n/a
970.0	45.7	52.6	43.0	34.6	n/a	n/a
1016.0	42.4	52.6	43.0	34.6	27.3	n/a
1062.0	37.4	52.6	43.0	34.6	27.3	n/a
1108.0	31.9	52.6	43.0	34.6	27.3	n/a
1154.0	27.2	52.6	43.0	34.6	27.3	20.9
1200.0	23.8	52.6	43.0	34.6	27.3	21.0

Adaptive Voltage Tuning Low-Vdd Blocks

Target (ps)	Yield (%)	Average Energy (pJ)	Average Low-Vdd Blocks (%)
925.0	90	47.6	5
970.0	96	45.7	12
1016.0	99	42.4	23
1062.0	100	37.4	40
1108.0	100	31.9	59
1154.0	100	27.2	75
1200.0	100	23.8	87

Outline

Coarse Grained Voltage Assignment

- Multiplier split into 6 stages [Li-ISCA-08]
- Brute force approach that measures power
- Iteratively test all 2^6 possible configurations
- Scaling considerations using brute force
- Very large power transistor
- Why 6 blocks?

Adaptive Voltage Scaling (AVS)

- Single supply voltage varies based on performance [Ch-TVLSI-03]
- Adaptive voltage tuning is orthogonal to AVS
- Use AVS to set the high-Vdd supply voltage
- Set low-Vdd to 200 mV less than high-Vdd

Adaptive voltage tuning would use less power with AVS

Outline

Adaptive Voltage Tuning Key Takeaways

- More tightly meet the design goals
- Achieve additional yield increases and power decreases
- Independent of the delay distribution and correlations between blocks

Adaptive Voltage Tuning Key Takeaways

- More tightly meet the design goals
- Achieve additional yield increases and power decreases
- Independent of the delay distribution and correlations between blocks

Google Bijansky

A Sub 1W to 2W Low Power IA Processor for Mobile Internet Devices in 45nm Hi-K Metal Gate CMOS

Gianfranco Gerosa, Steve Curtis, Mike D'Addeo,
Bo Jiang, Belliappa Kuttanna, Feroze Merchant,
Binta Patel, Mohammed Taufique,
Haytham Samarchi and Chris Weaver

Intel Corporation
Austin, TX

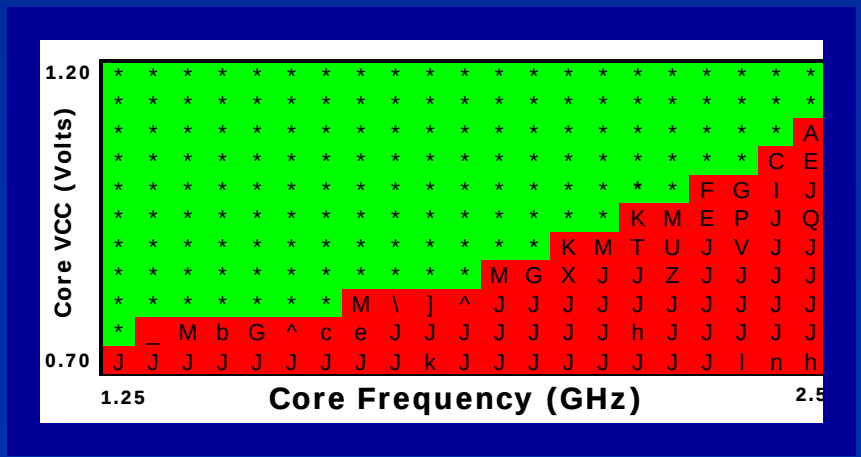
OUTLINE

- Block diagram and processor pipeline
- Power states
- 45 nm CMOS technology features
- Sea-of-FUBs floor-plan
- Power optimization
  - Grid-less clock distribution
  - Register files/L2 cache/ROM
  - CMOS mode in FSB
  - Split FSB power supply
- Conclusions

16 STAGE PROCESSOR PIPELINE

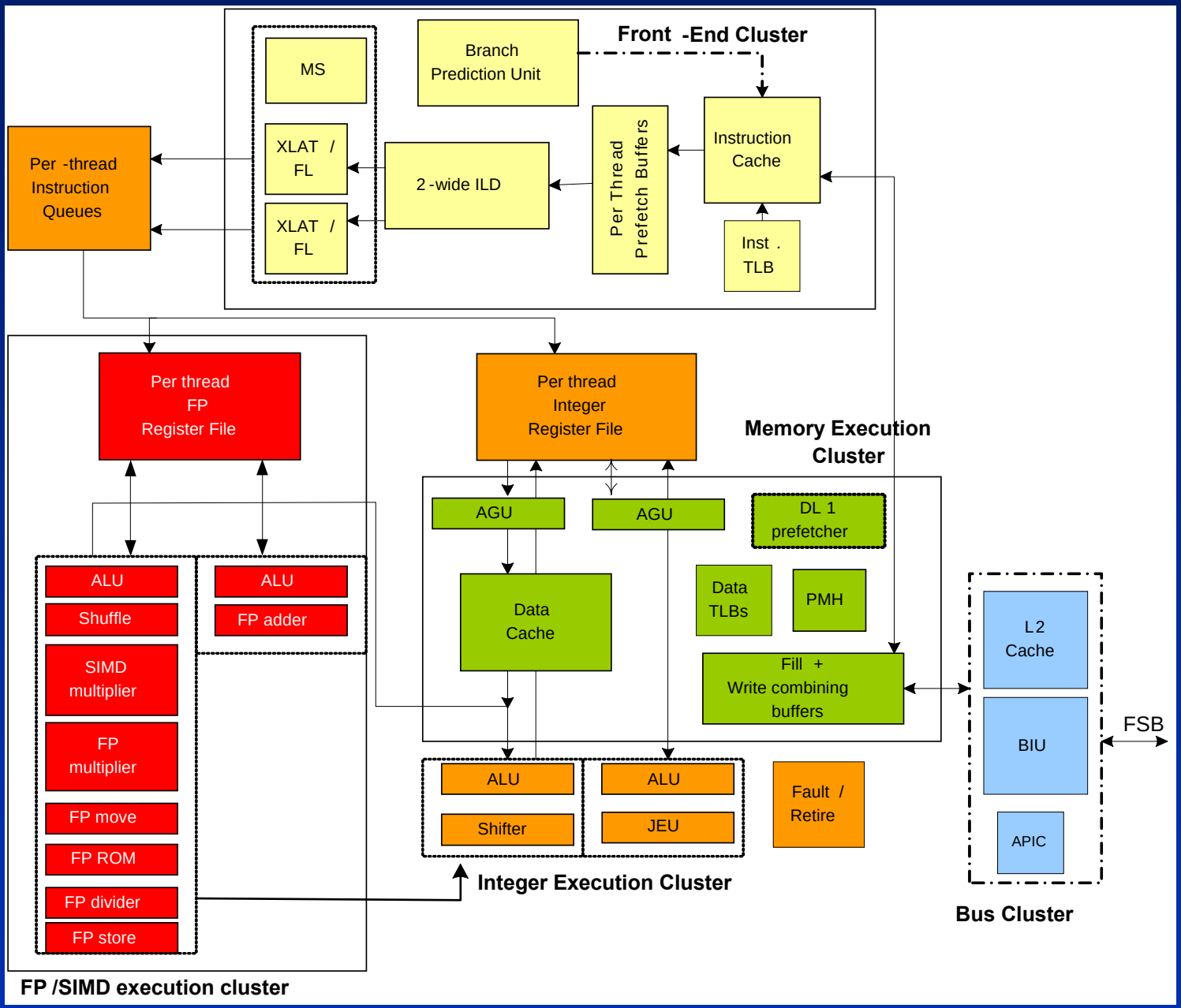
IF1	IF2	IF3	ID1	ID2	ID3	SC	IS	IRF
Instruction Fetch			Instruction Decode			Instruction Dispatch		Source Operand Read

AG	DC1	DC2	EX1	FT1	FT2	IWB / DC1
Data Cache Access			Execute	Exceptions and MT handling		Commit



Typical silicon results show that this pipeline is capable of high core frequencies.

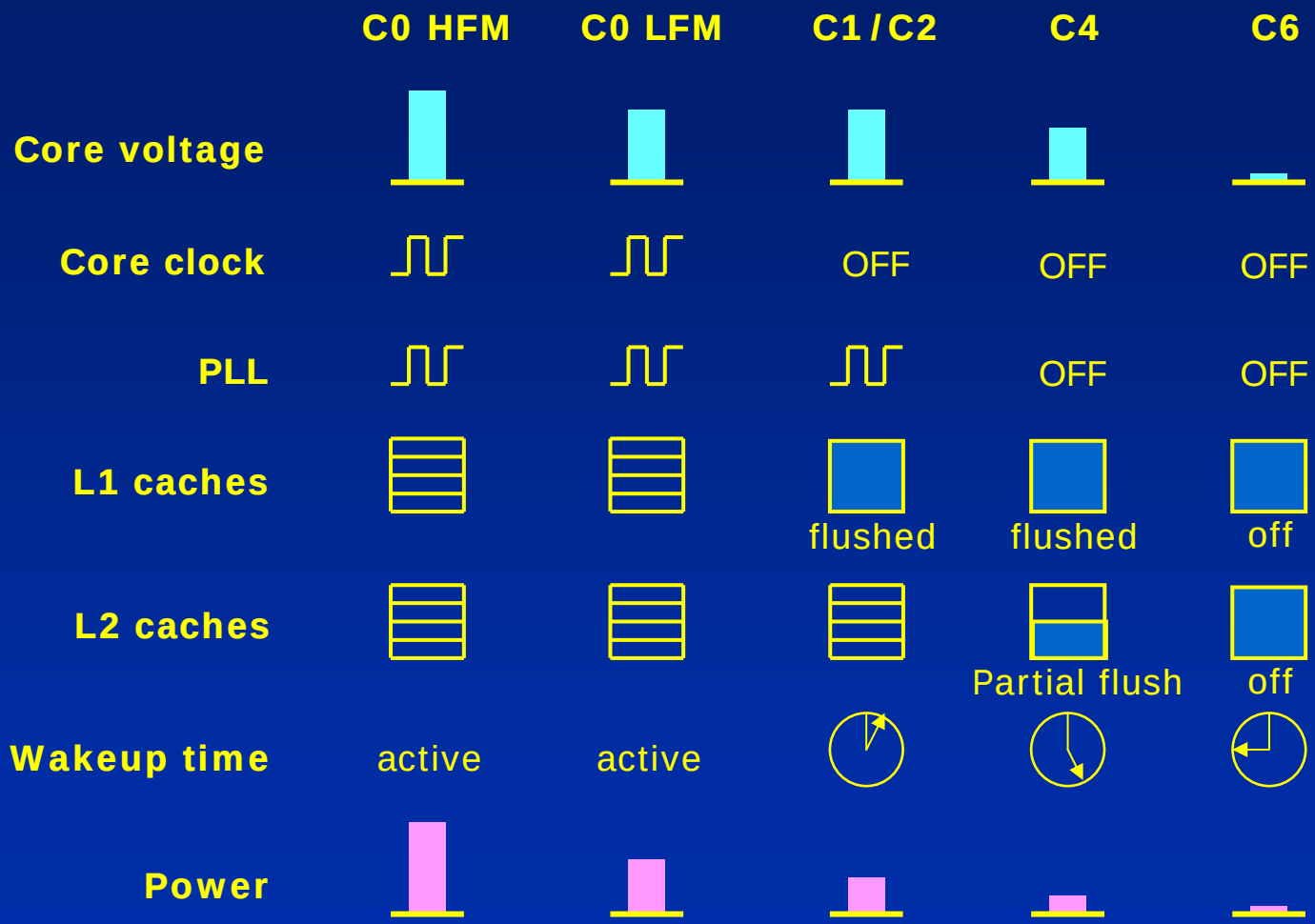
BLOCK DIAGRAM



MICRO-ARCHITECTURE HIGH-LIGHTS

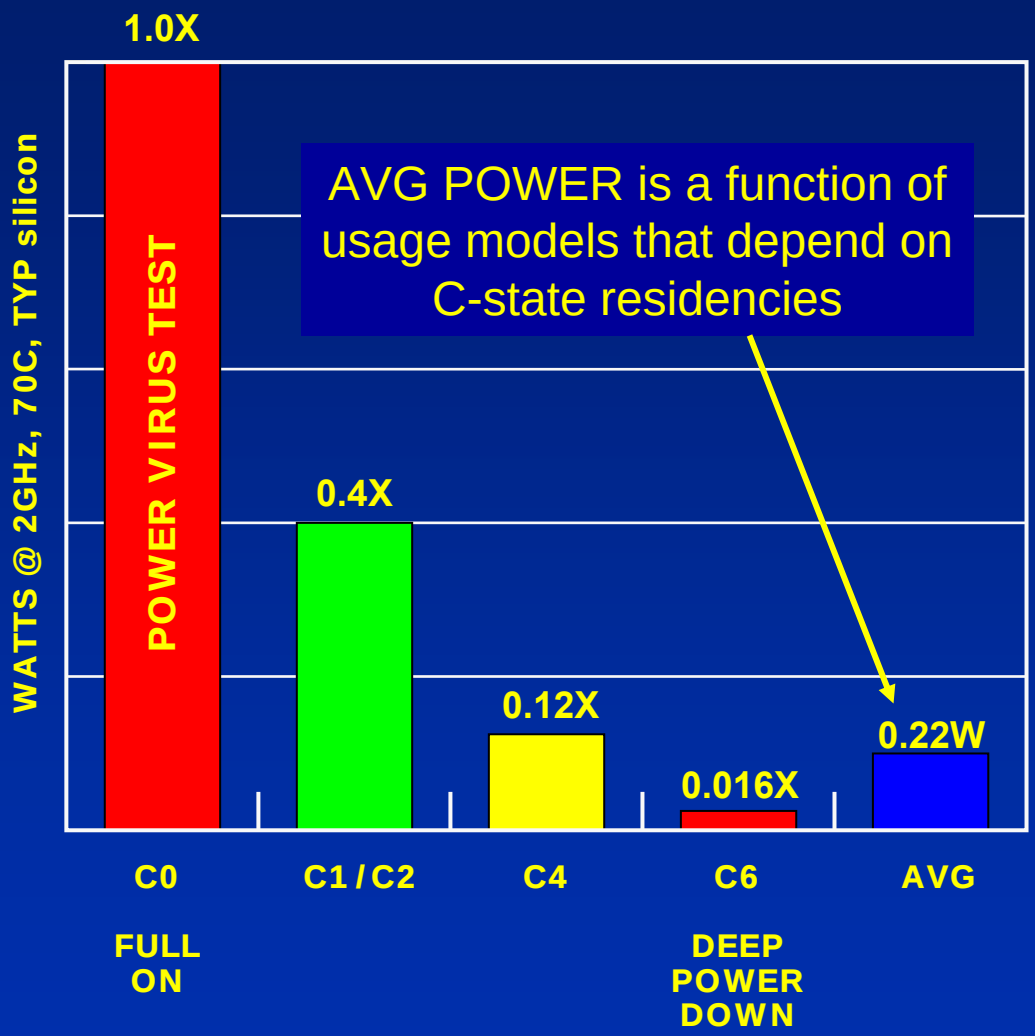
- Dual-decode, dual-issue, in-order execution
- Support for two threads
- Core 2-duo instruction set compatible
- 128 bit SIMD integer ALUs, 64 bit FP multiplier/divider execution units
- 32KB instruction L1 and 24KB data L1, unified L2 cache w/ in-line ECC
- External Front-Side-Bus interface supports 400 and 533 MT/s in both CMOS and GTL

POWER C- STATES



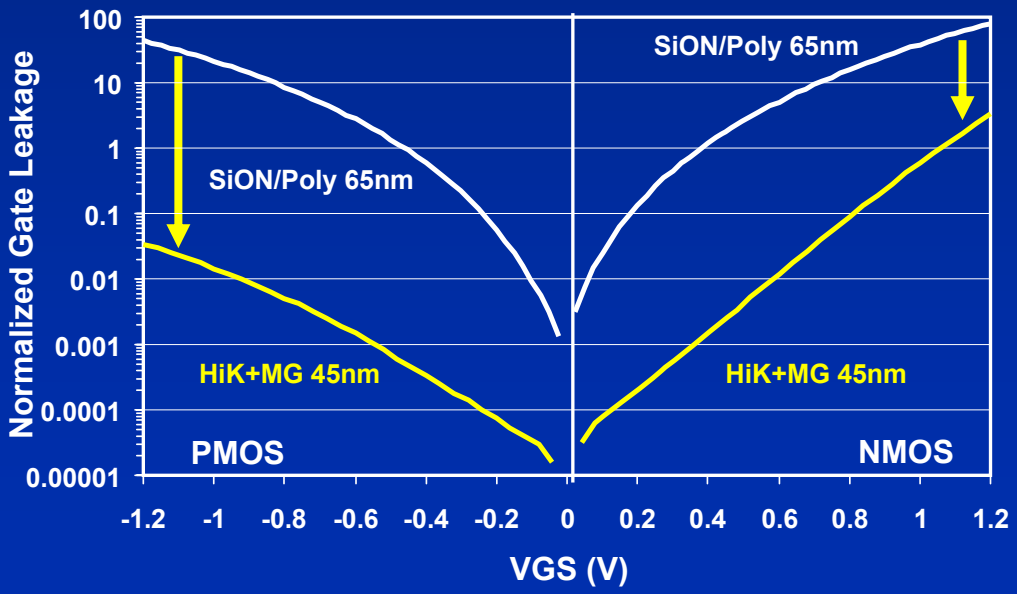
C6 residency can be as high as 80-90 % depending on usage and workload.

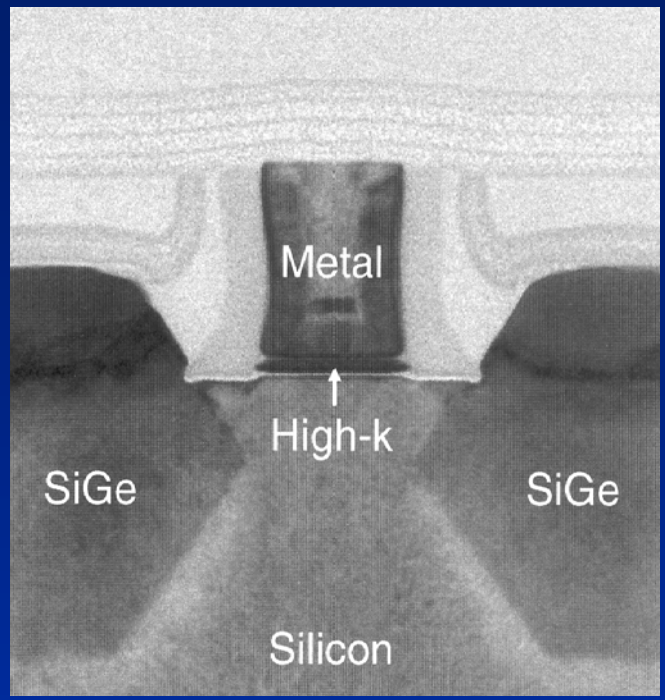
POWER RESULTS for VARIOUS C- STATES

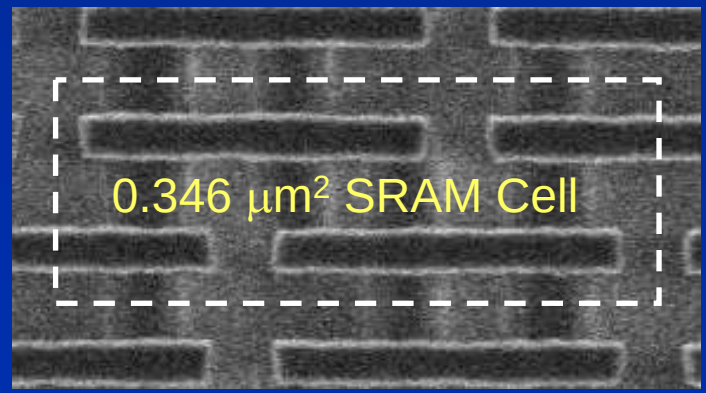


45 nm CMOS TECHNOLOGY

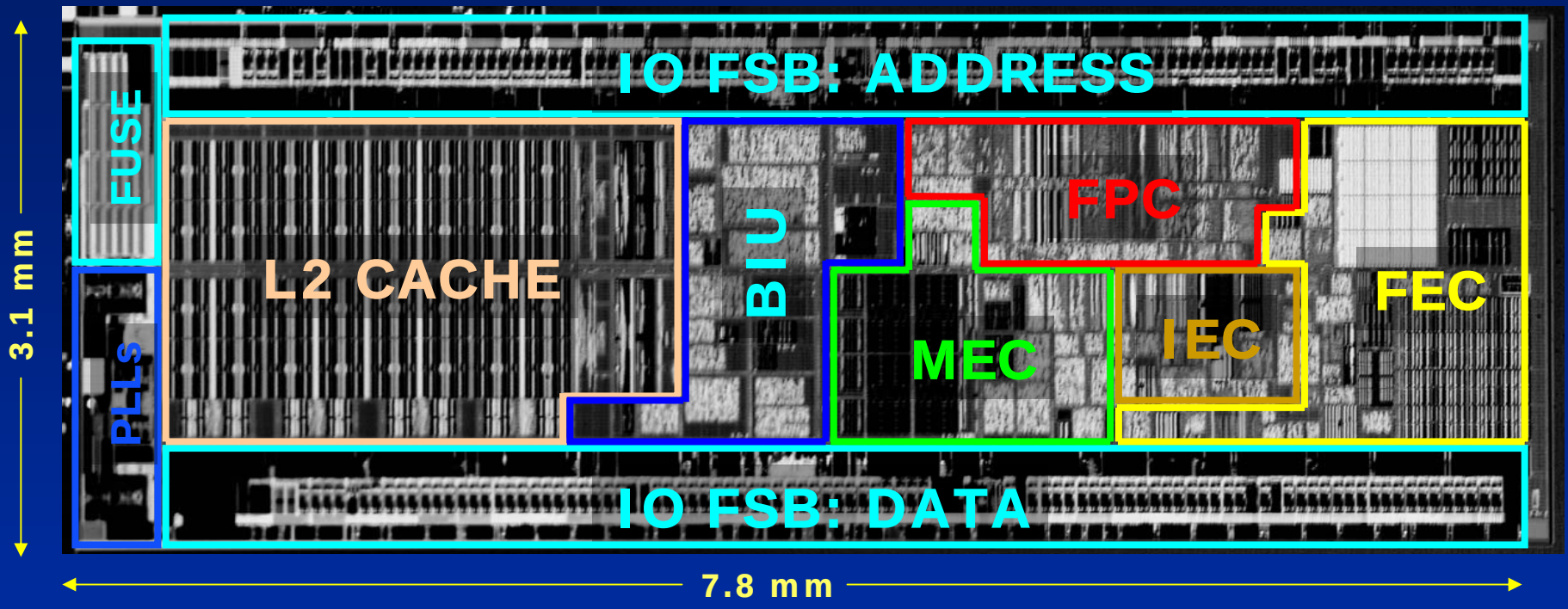
- High-k + Metal Gate transistors
- 25X-1000X gate leakage reduction
- 0.346 μm² and 0.382 μm² SRAM cells
- 100% use of longer channel lengths at a modest drop in transistor performance
- 9 layers of metal







SEA-OF-FUBs CHIP LAYOUT

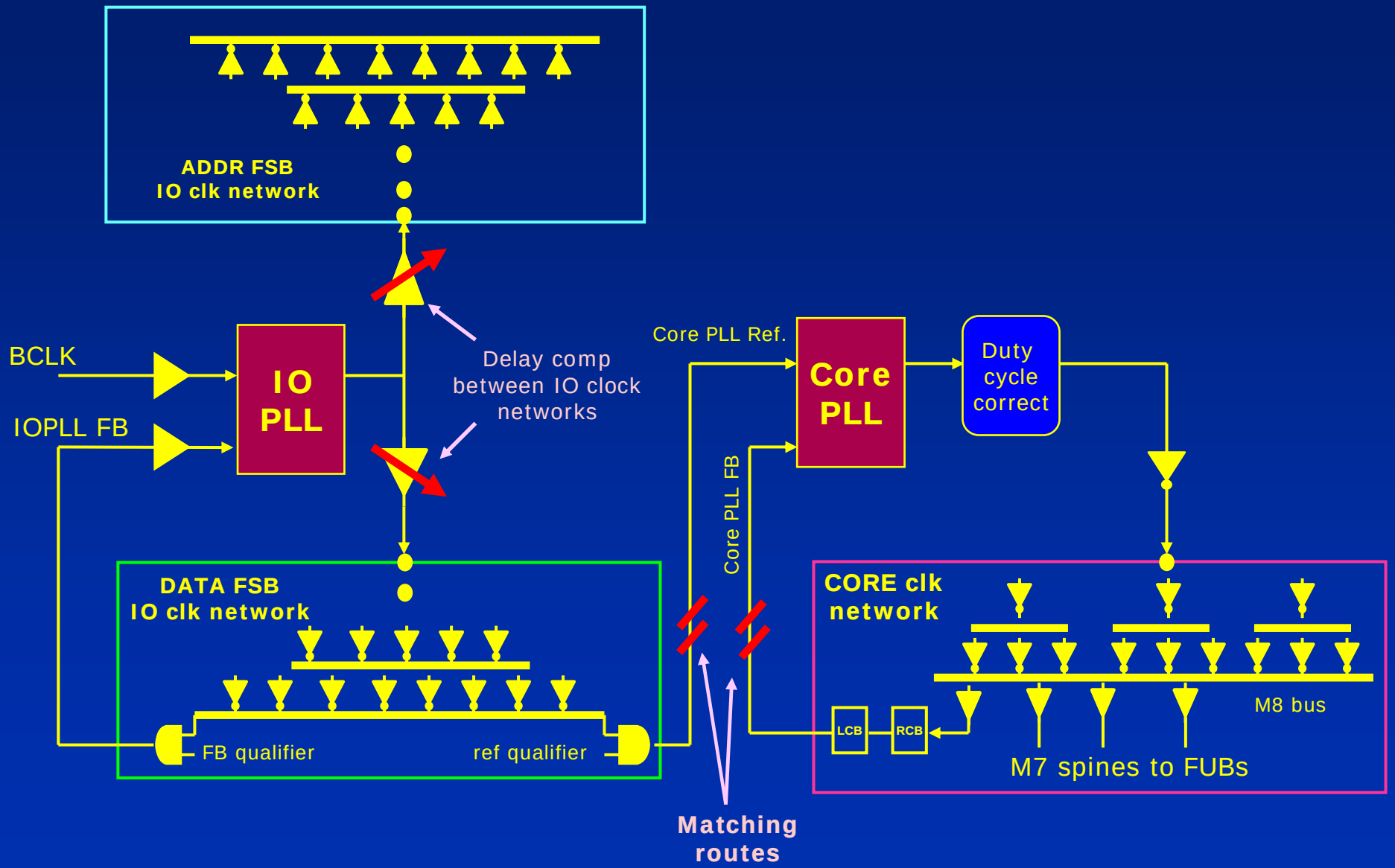


SCHEMATIC TRANSISTORS:

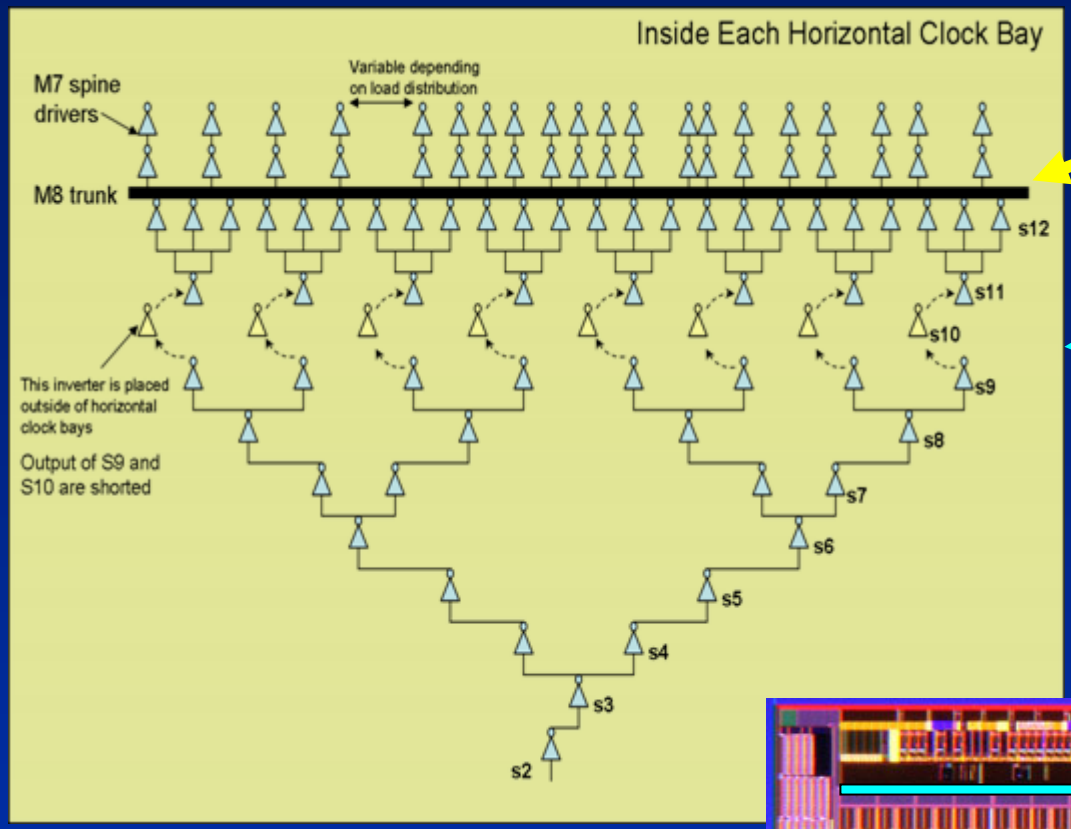
Core:	13,828,574
Uncore:	2,738,951
L2 & L2 tag:	30,644,682
TOTAL:	**47,212,207**

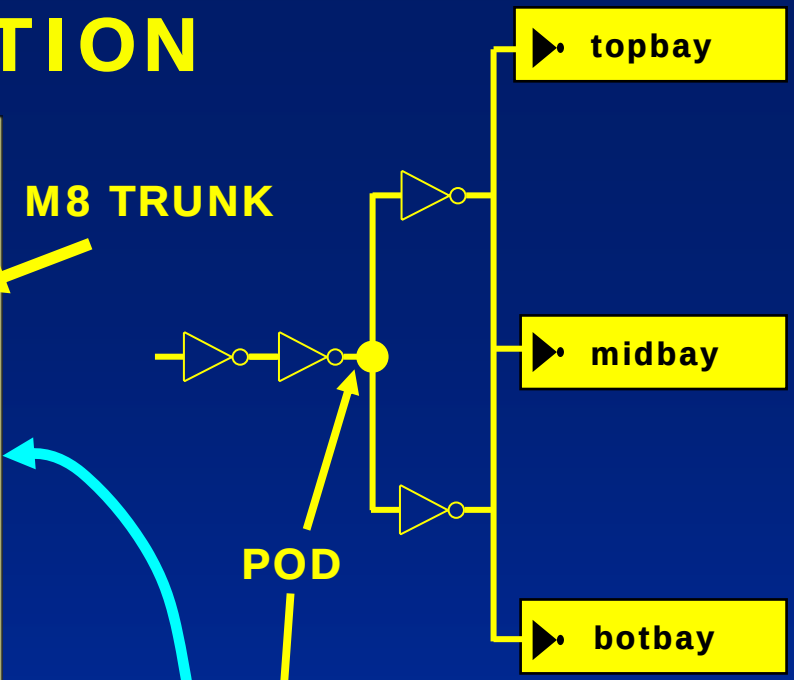
type	unique	instances
Random Logic Synthesized	92	92
Structured Data Paths	88	140
L2 sub-arrays	2	40
Custom	18	19
Repeater Stations	-	317
TOTAL	200	608

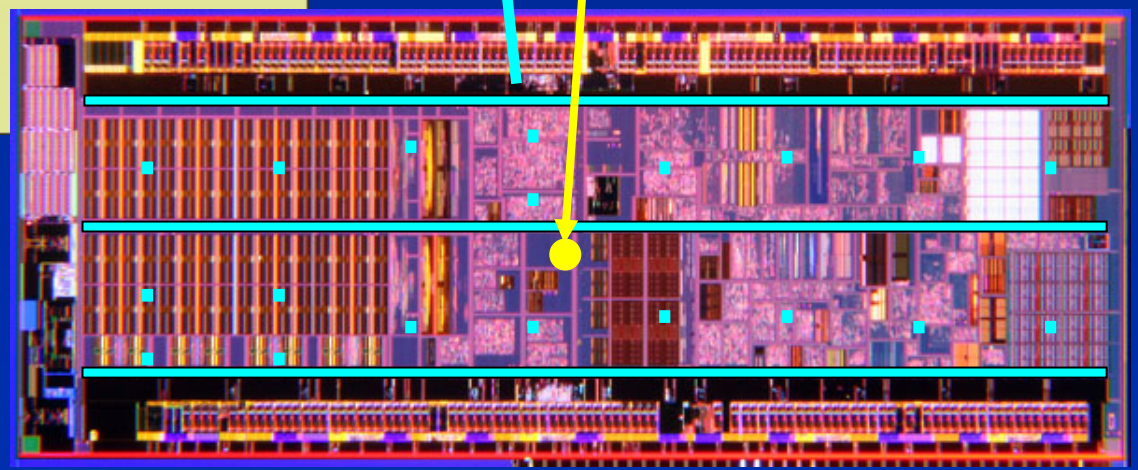
TWO PLL CLOCK DISTRIBUTION



CORE CLOCK DISTRIBUTION







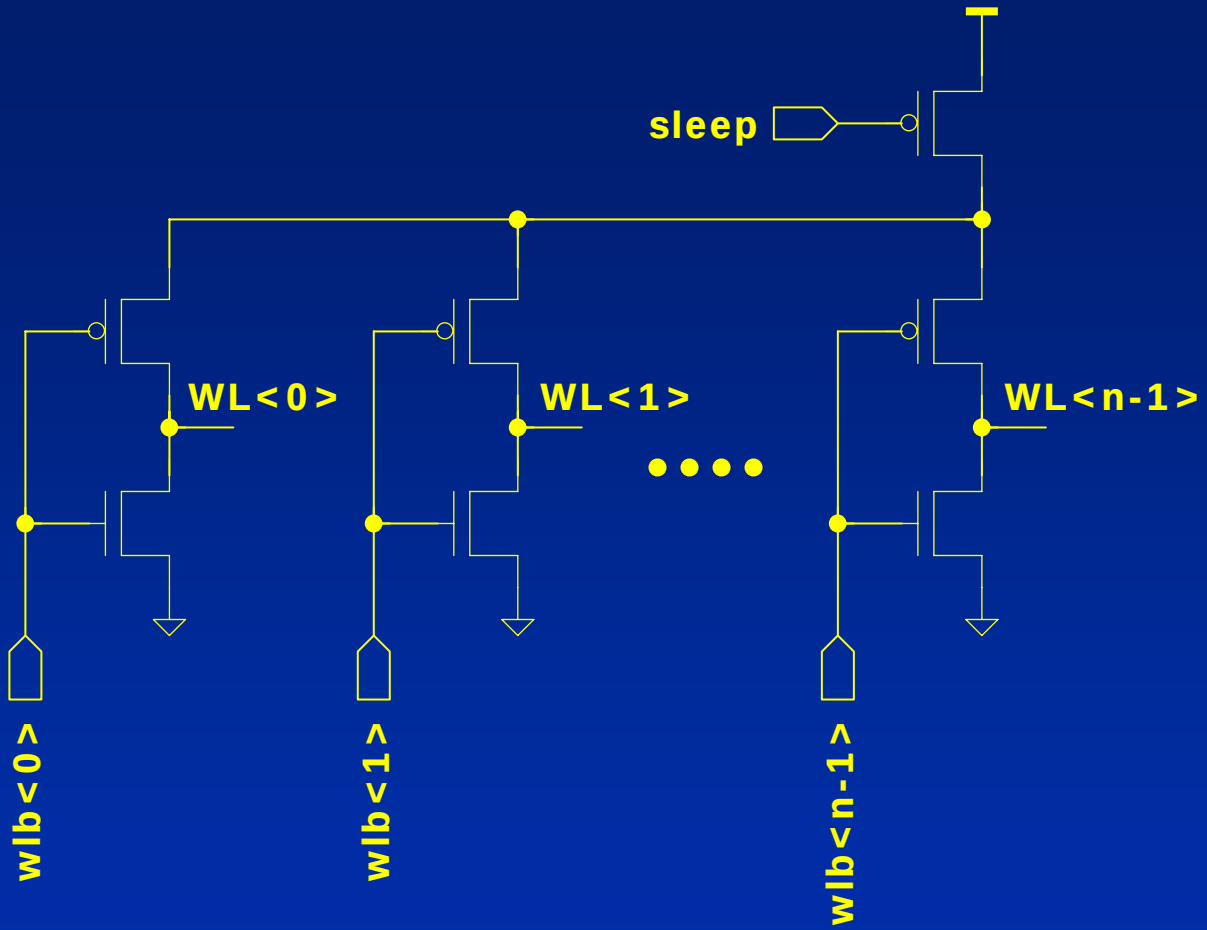
Clock distribution power is < 7% of the total processor power with < 10ps skew

REGISTER FILES / L2 CACHE / ROM

Power Reduction Techniques

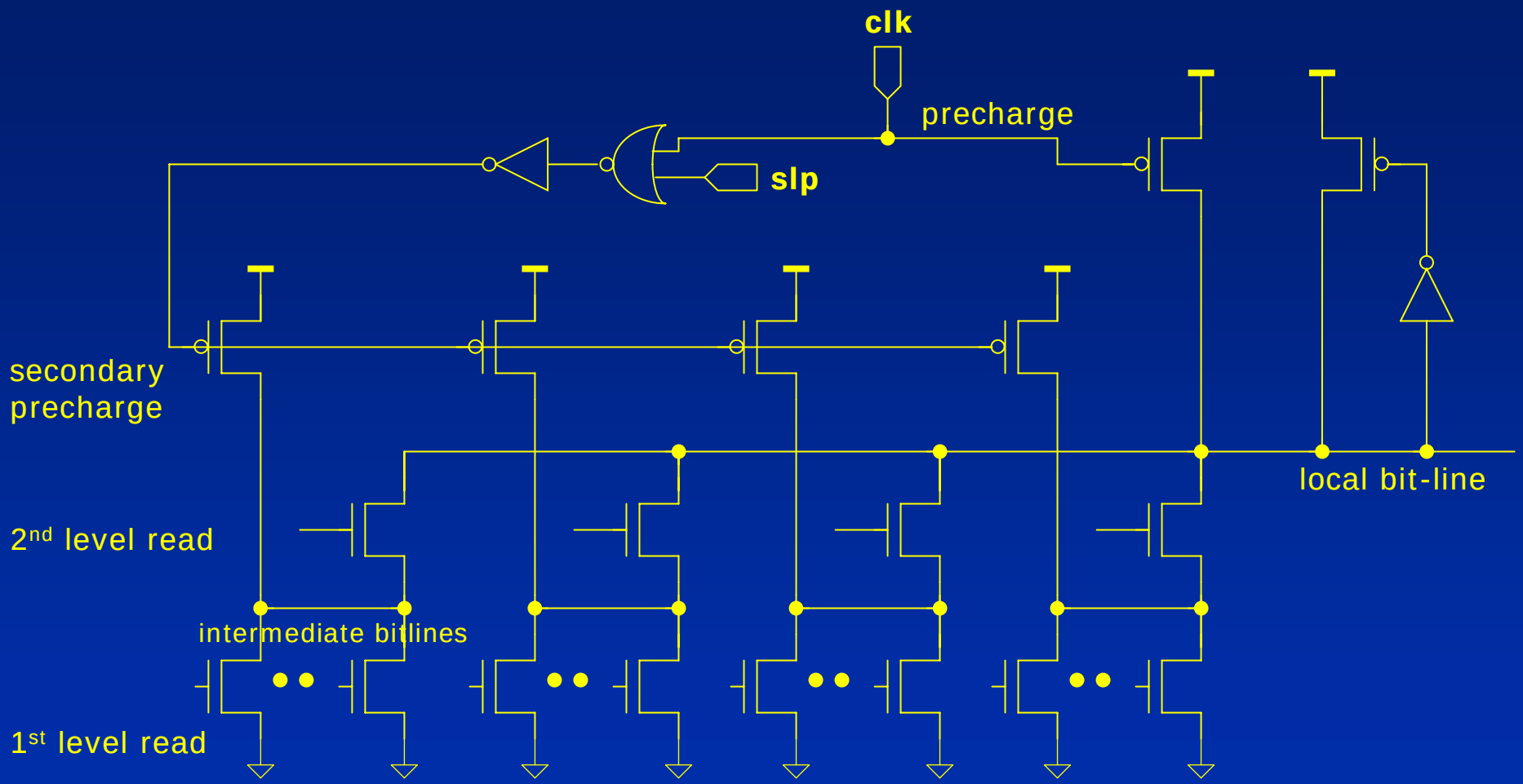
- Clock gating everywhere
- “floating” bit-lines in quiescent arrays
- Fine granularity “sleep” on word-line drivers
- Adjustable L2 cache retention power supply via fuses
- Register file and L2 cache bit-cells optimized to operate at lowest power supply

FINE GRANULARITY SLEEP WL DRIVER



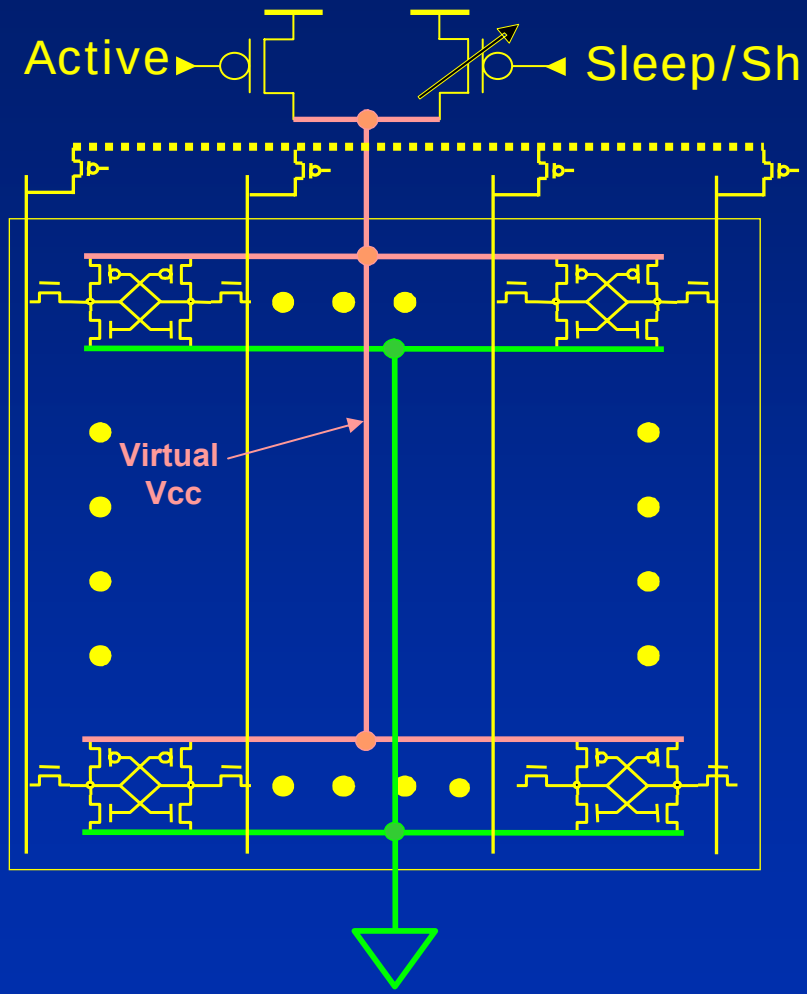
During IDLE state, the SLEEP PFET is off, thus reducing the WORDLINE driver sub-threshold leakage due to stacking.

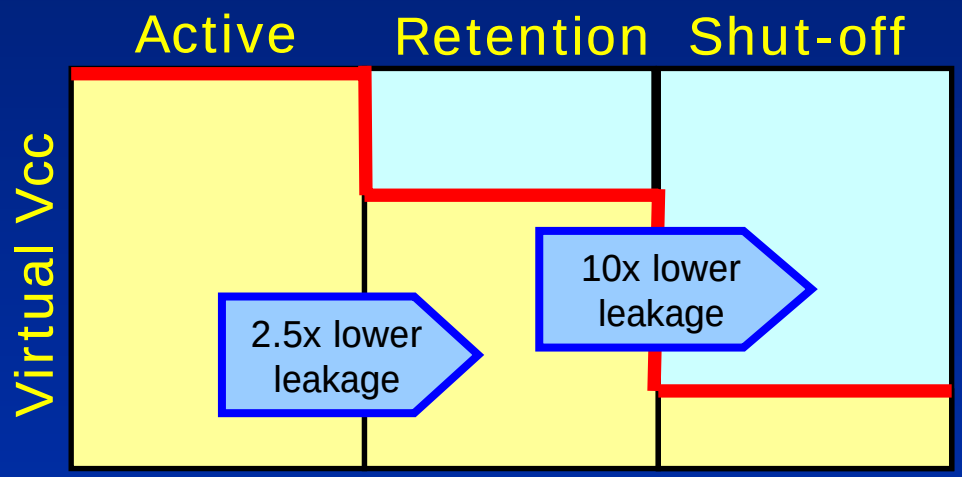
ROM BITLINE FLOATING SCHEME



Sub-threshold leakage is reduced on the intermediate bitlines.

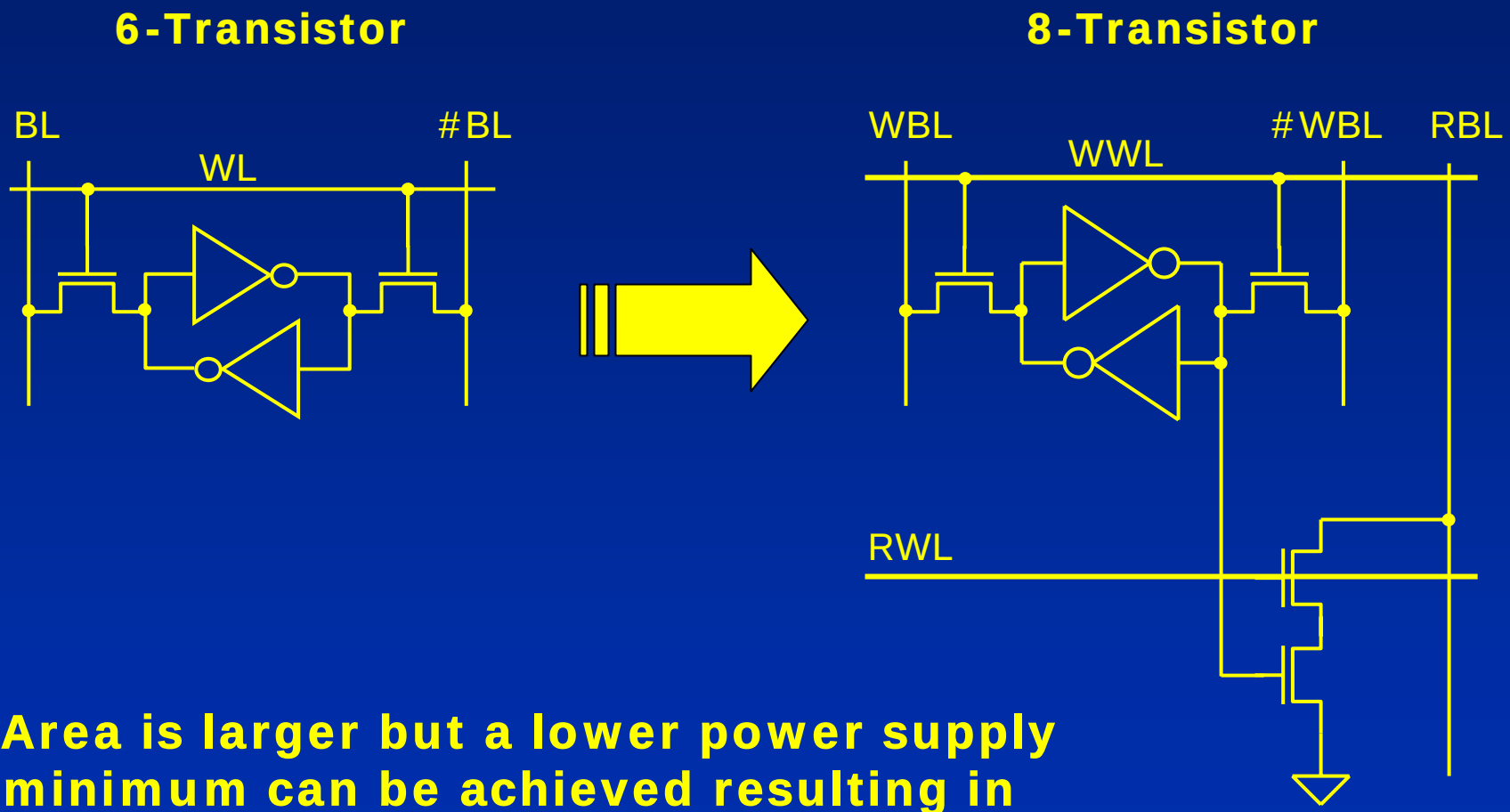
L2 CACHE ACTIVE/SLEEP/SHUT-OFF





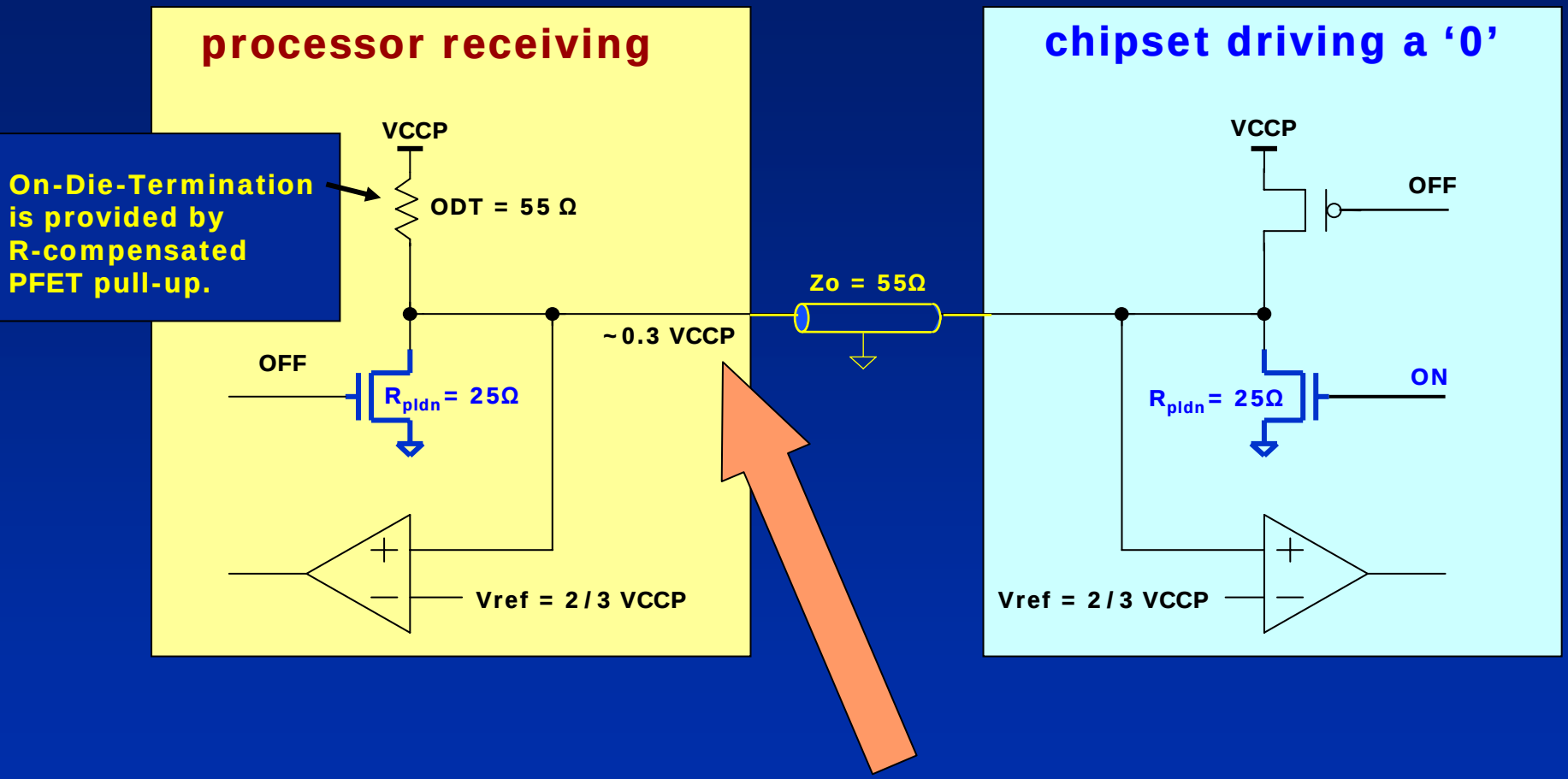
Lower array leakage is attained by lower retention Virtual VCC and by shutting off un-used ways.

L1 CACHES USE 1R/1W 8-T BITCELLS



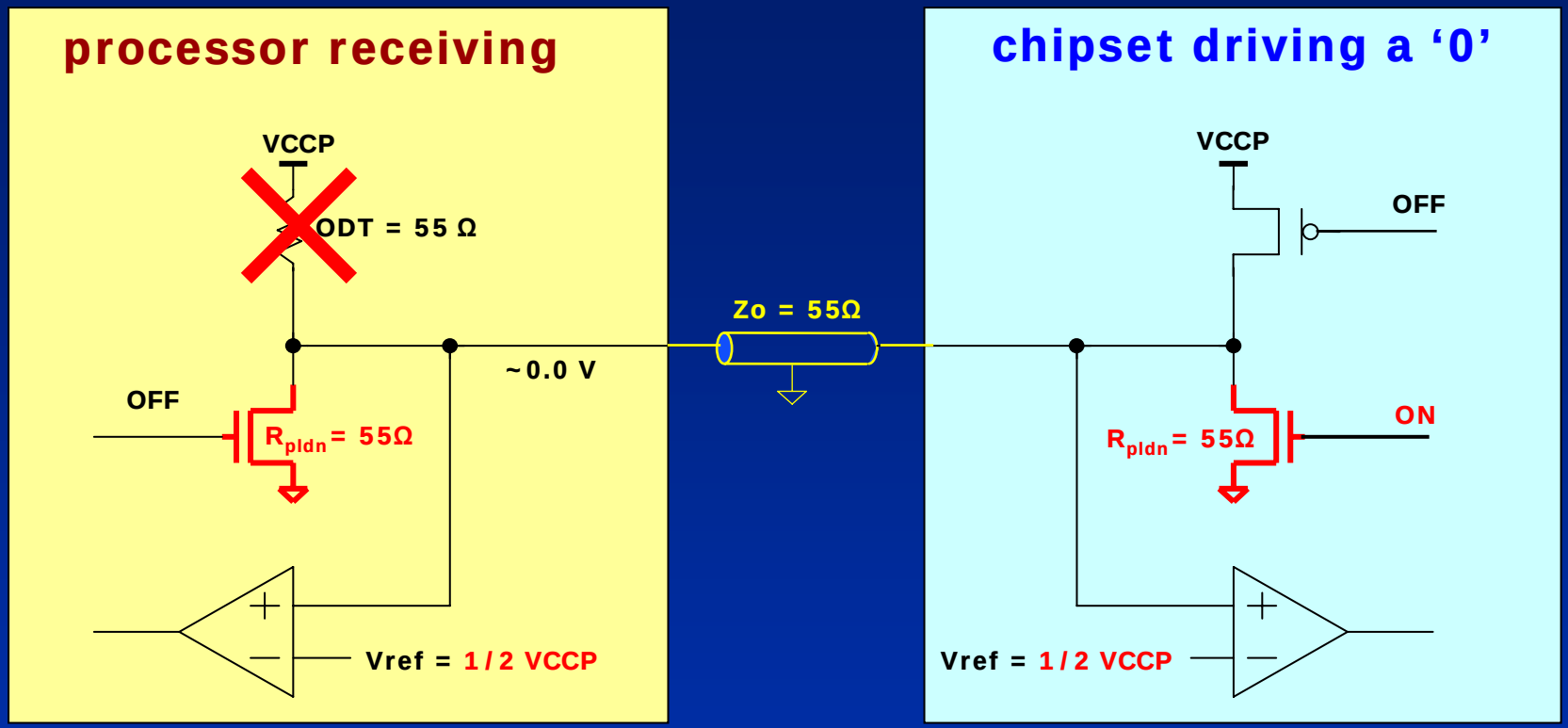
Area is larger but a lower power supply minimum can be achieved resulting in overall lower chip power.

TRADITIONAL GTL FSB DRIVER/RECEIVER



~12 mA of DC current contributes to FSB power

CMOS MODE FSB DRIVER/RECEIVER

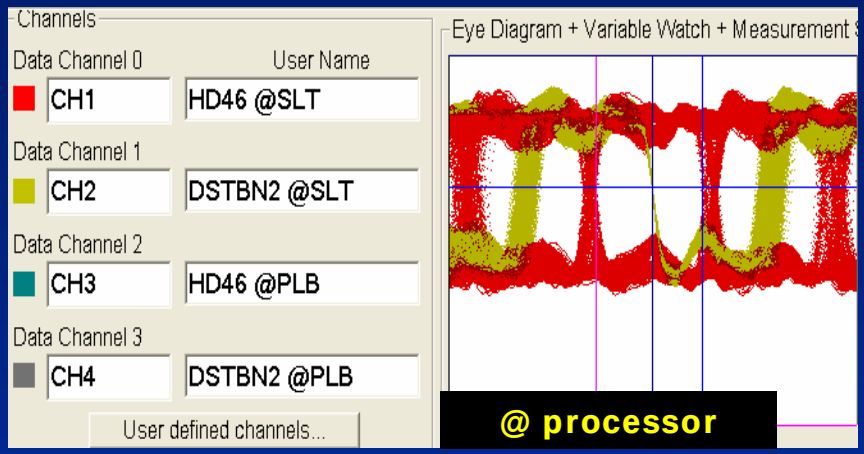


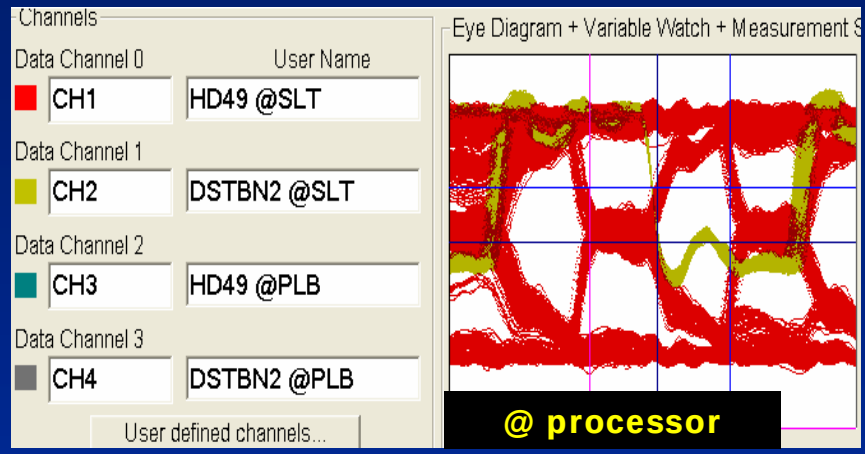
- Both NFET pull-down impedances are changed to 55 ohms
- 55 ohm On-Die-Termination is turned-off
- VREF on the input receivers is changed to ½ VCCP
- **200-500 mWatts FSB platform power reduction @ 400 MT/s**

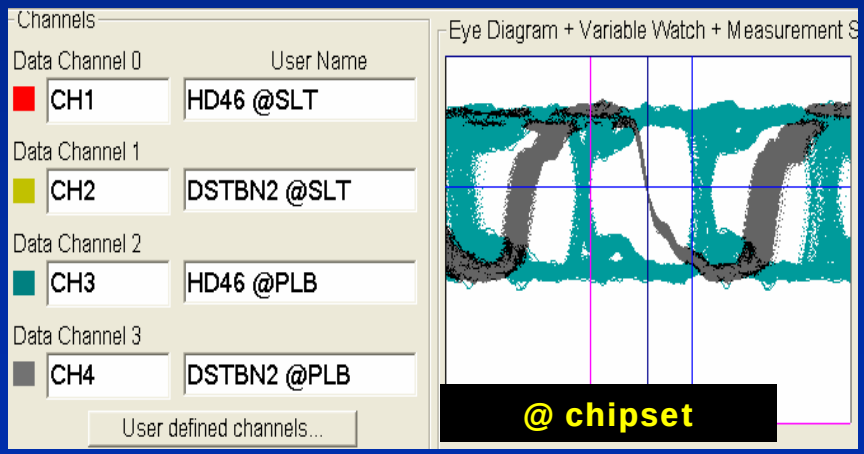
DATA / STROBE WAVEFORMS

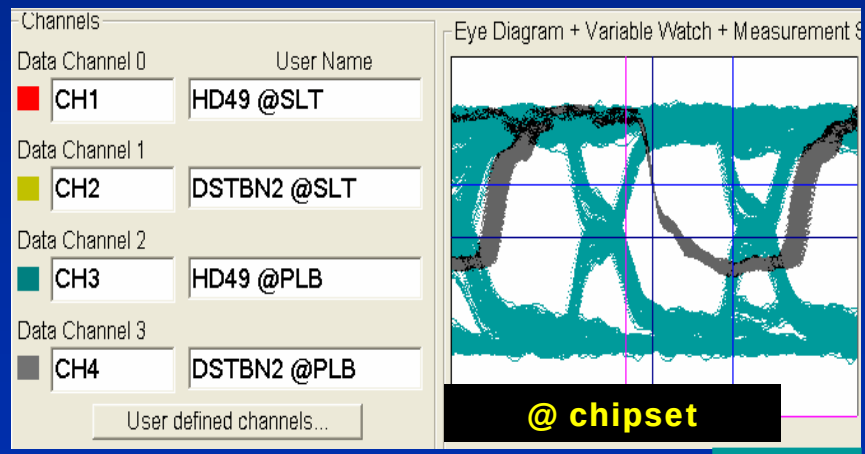
strobe

data









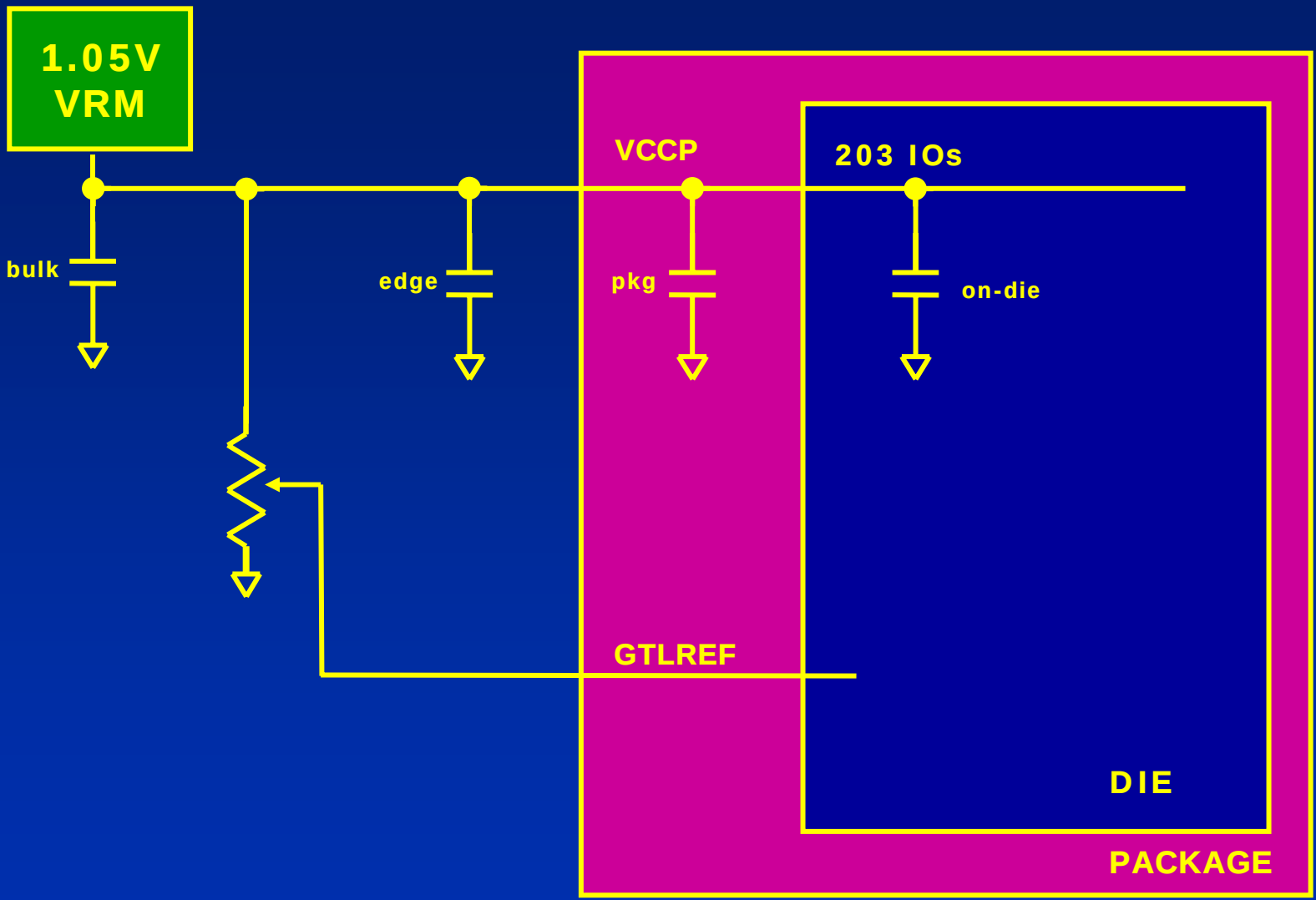
data

strobe

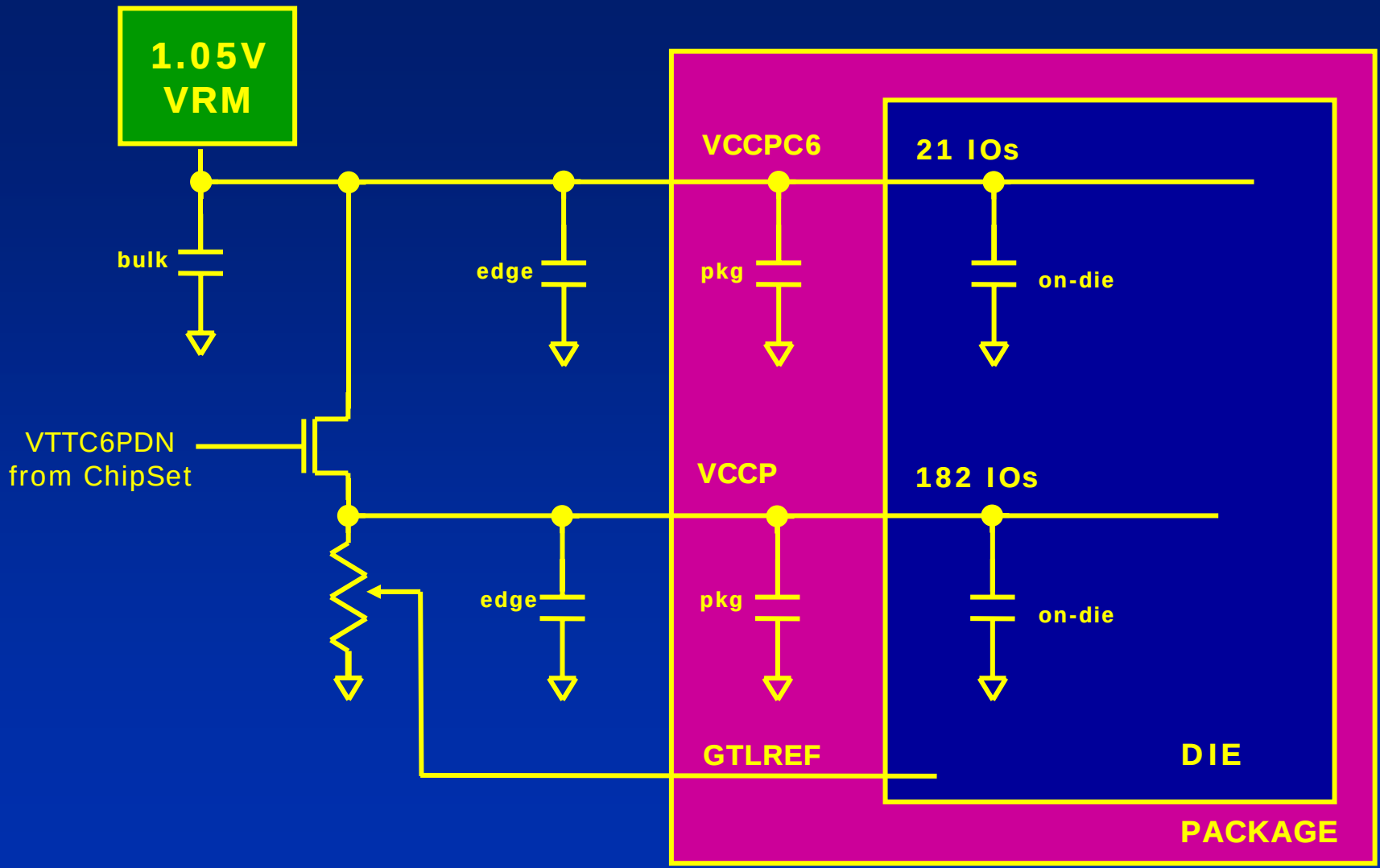
GTL MODE

CMOS MODE

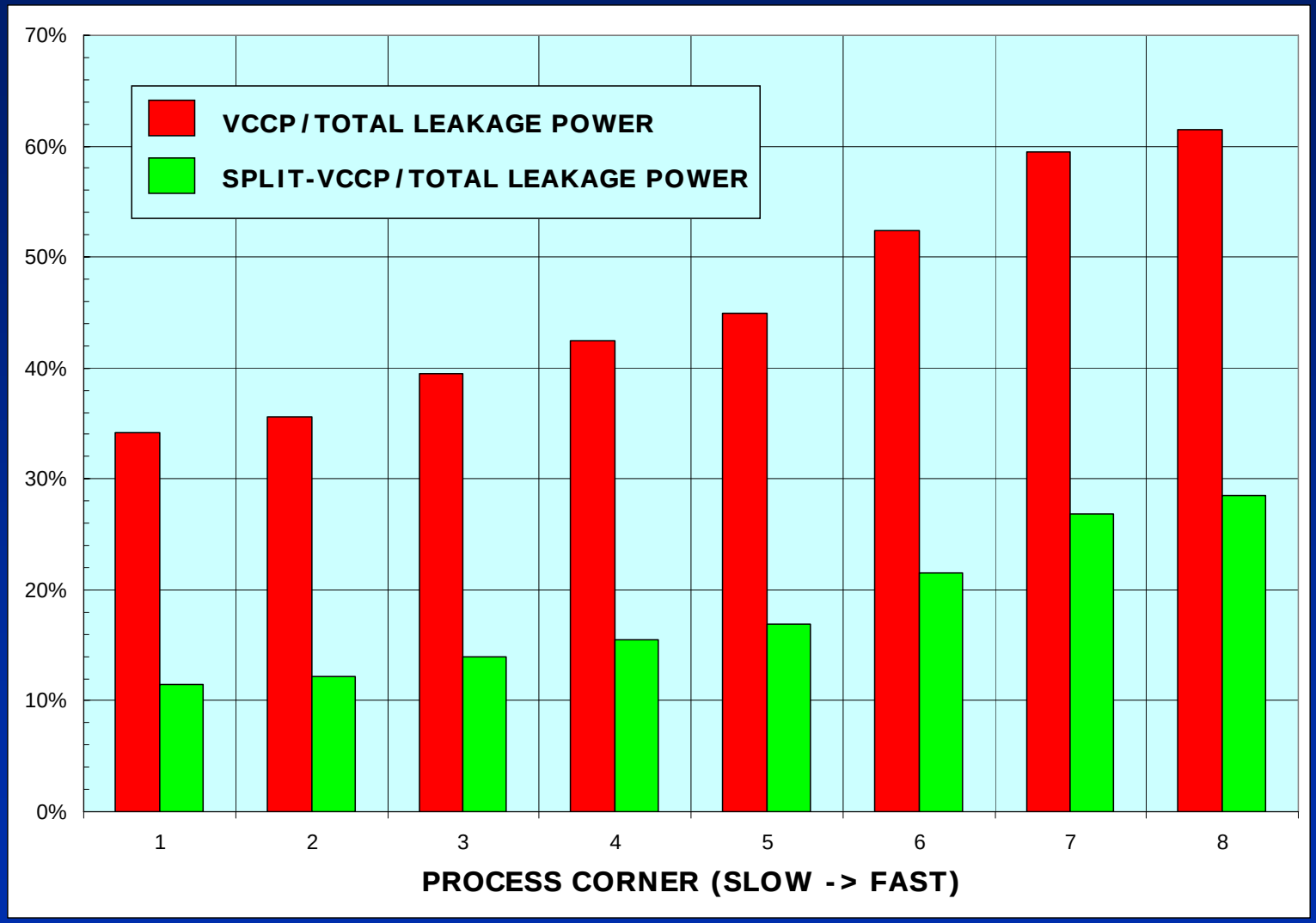
SINGLE IO POWER SUPPLY (VCCP)



SPLIT-VCCP IO POWER SUPPLY (VCCP, VCCPC6)

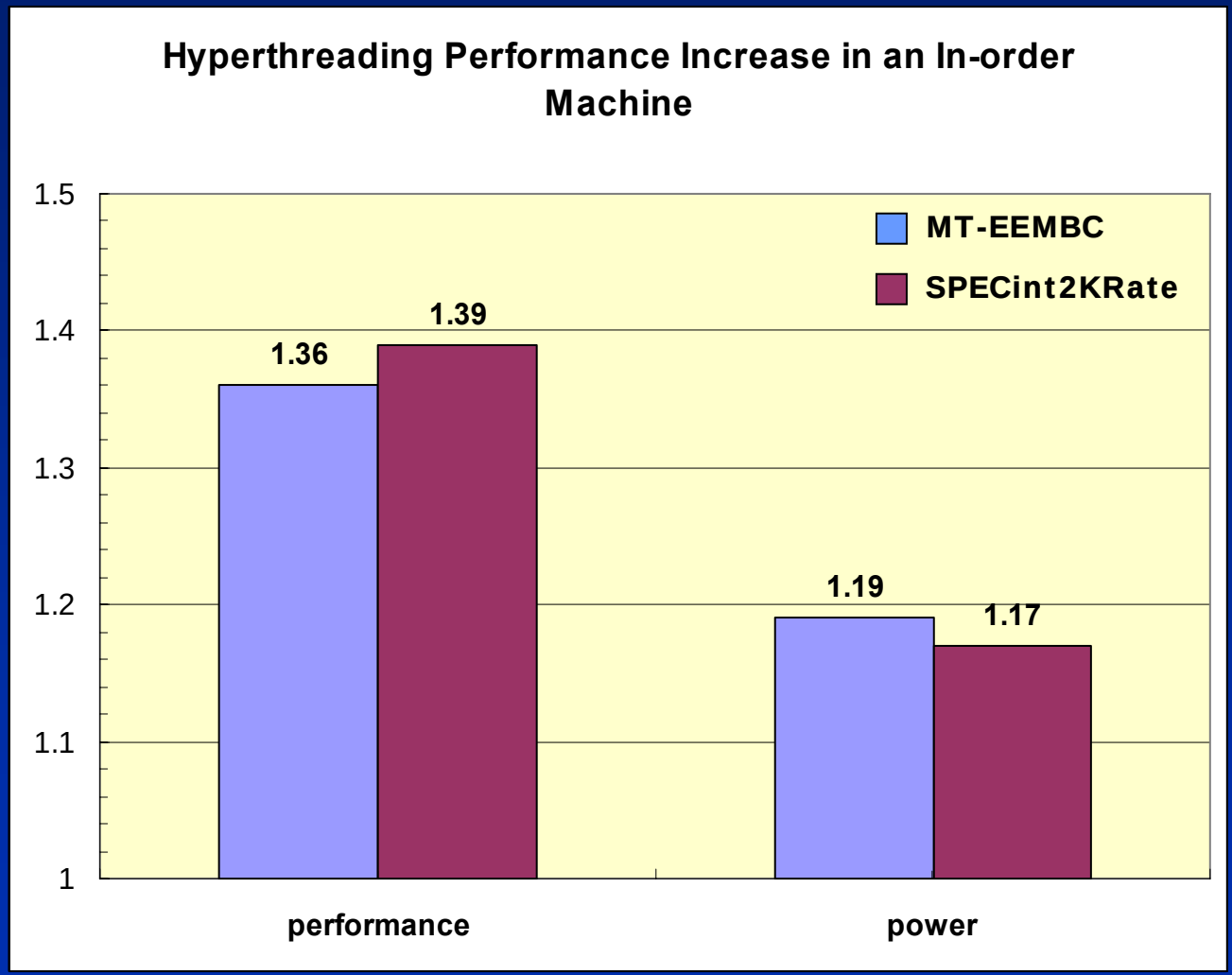


SPLIT-VCCP LEAKAGE IMPROVEMENT

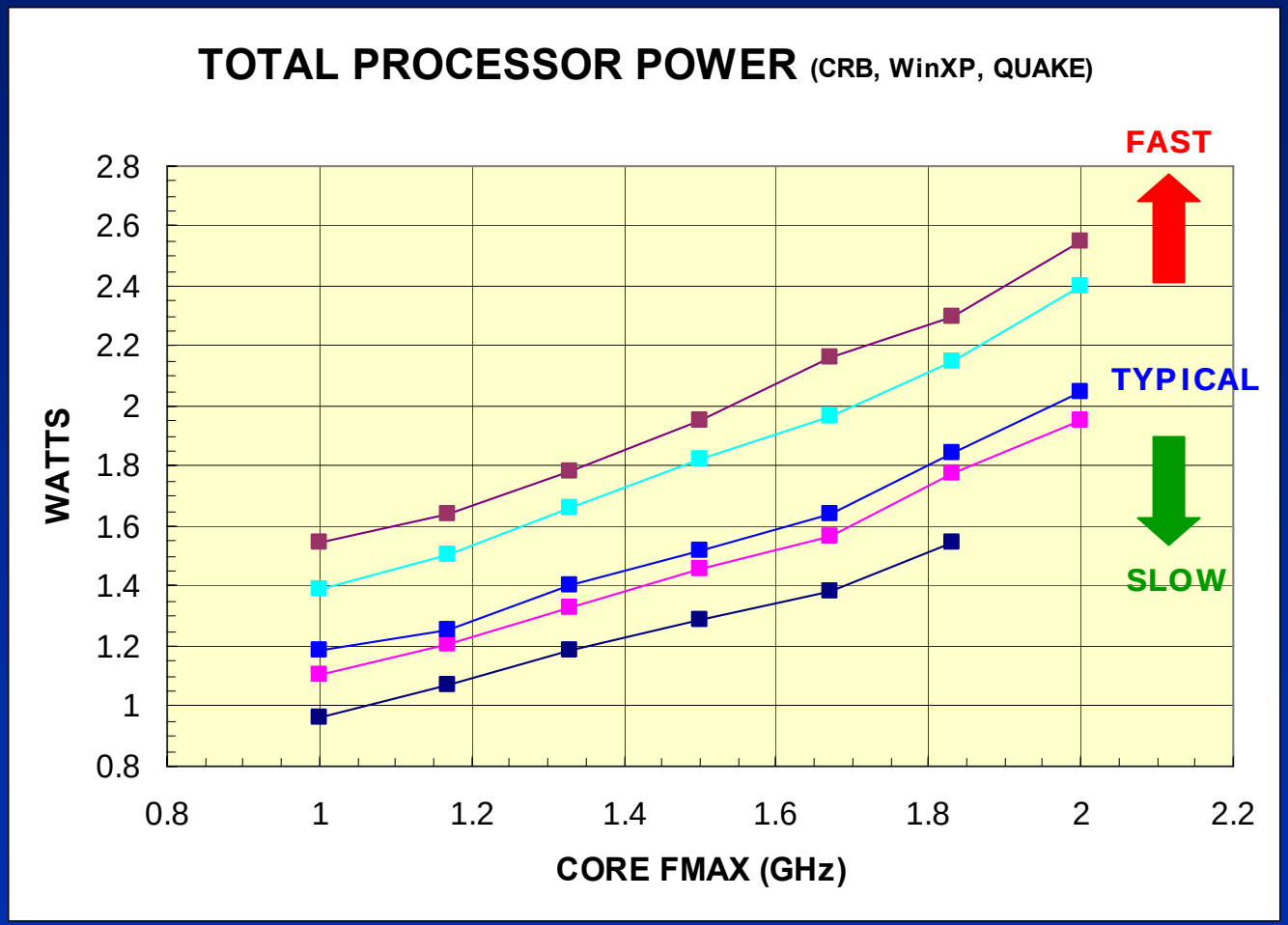


SPLIT VCCP reduces IDLE Power IO leakage contribution over 2X

TWO THREADS PERFORMANCE / POWER



TOTAL PROCESSOR POWER



Power measured in a reference platform running QUAKE for 30 minutes with FSB=667 MT/s.

CONCLUSION

- An IA Micro-architecture optimized for Mobile Internet Device performance has been presented. The design is instruction-set compatible with the 65nm CORE 2 DUO processor.
- Various power management techniques are used to reduce dynamic and leakage power
  - Deep power down (C-6) architecture
  - Non-grid clock distribution
  - Optimized register-file and cache 6T bit-cells for lowest power-supply operation
  - Pervasive clock gating, word-line driver gating, floating bitlines and programmable L2 array retention supply
  - CMOS mode on quad-pumped FSB IO
  - Split IO power supply to further reduce IDLE leakage power

A High Speed 128-Point Fast Fourier Transform Circuit for OFDM Systems

Tung-Yeh Wu
Computer Engineering Research Center
The University of Texas at Austin
Austin, TX 78712
tywu@cerc.utexas.edu

Jacob A. Abraham
Computer Engineering Research Center
The University of Texas at Austin
Austin, TX 78712
jaa@cerc.utexas.edu

***Abstract*—The Fast Fourier Transform (FFT) is an efficient way to calculate the Discrete Fourier Transform (DFT), which is widely used in digital signal processing and Orthogonal Frequency Division Multiplexing (OFDM) communication systems. The data throughput rate of the FFT computation is crucial, since it can be the bottleneck of the whole communication system. In this paper, we propose a high-speed 128-point FFT circuit which targets next generation wireless communication systems with a 1.5 GHz sampling rate. It utilizes a high speed arithmetic unit with deep pipelining to achieve the desired data throughput rate. The data width is 6 bits wide on the input and 12 bits wide internally for fixed point computations. Our design adopts decimation in frequency and a radix-2 structure. It is designed and implemented in UMC 0.13 μm technology.**

I. Introduction

Orthogonal Frequency Division Multiplexing (OFDM) is now the most promising solution for modern high speed wireless communication systems. OFDM plays a key role in several current wireless communication standards such as 802.11 Wireless Local Area Network (WLAN) and 802.15 Wireless Personal Area Network (WPAN) [1]. It has the benefit of achieving a higher data transmission rate with the best bandwidth usage by modulating signals onto several sub-carriers which are orthogonal to each other within one channel. It also has better Inter-Symbol Interference (ISI) immunity against the delay spread because of the multi-path transmission.

Figure 1 shows a simplified OFDM based transceiver frontend. The Fast Fourier Transform and the Inverse Fast Fourier Transform (FFT/IFFT) are two of the most important computation units. FFT computation has the natural characteristic of transferring the original signal onto an orthogonal basis set without using multiple mixers. Furthermore, the computation time of the FFT/IFFT is crucial, since it can become the bottleneck of the whole communication system. The FFT computation requirement of the current communication standards is shown in Table I.

In this paper, we propose a high speed 128-point deeply-pipelined FFT computation unit which supports a next generation wireless communication system with a 1.5 GHz data sampling rate. The paper is organized as follows. In Section II, we explain the radix-2 decimation-in-frequency FFT algorithm. Section III describes the system architecture and design issues. Section IV shows the implementation result, and the final section is the conclusion of this paper.

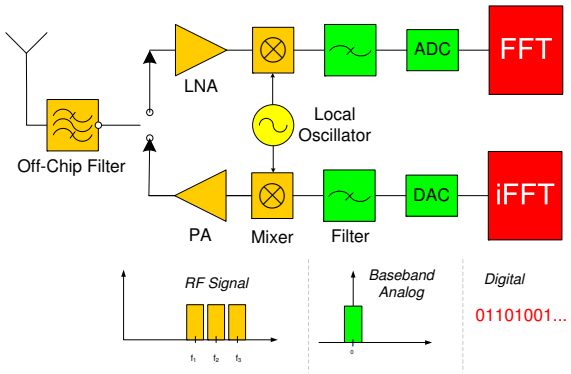


Fig. 1. OFDM wireless transceiver frontend

TABLE I
FFT REQUIREMENT FOR CURRENT COMMUNICATION STANDARDS

Application	WLAN (802.11a/g)	WPAN(802.15.3a)
FFT Size	64	128
T_{FFT}	3.2 μs	312.5ns

II. The FFT Algorithm

The FFT is a efficient way to calculate the Discrete Fourier Transform (DFT) [2]. The original N-point DFT computation can be performed as shown in Equation 1.

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn} \tag{1}$$

W_N^{kn} is called the twiddle factor, which is a complex number in the format of

$$W_N^{kn} = e^{-j(\frac{2\pi}{N})kn}. \tag{2}$$

The computation complexity of the DFT calculation in Equation 1 is $O(N^2)$. One DFT calculation requires N^2 complex additions and multiplications, which becomes infeasible as N increases. However, the FFT can reduce the computation

complexity dramatically by utilizing the symmetry and the periodicity of the twiddle factor W_N. The most common and well-known FFT algorithm is the Cooley-Tukey FFT algorithm published in 1965 [3] which can reduce the computation complexity to $Nlog_2N$.

To convert the DFT calculation into the radix-2 decimation in frequency FFT, we can split the equation into an even number indexed sequence and an odd number indexed sequence in the frequency domain.

$$X[2r] = \sum_{n=0}^{N-1} x[n]W_N^{n(2r)}, r = 0, 1, ..., \frac{N}{2} - 1 \quad (3)$$

$$X[2r+1] = \sum_{n=0}^{N-1} x[n]W_N^{n(2r+1)}, r = 0, 1, ..., \frac{N}{2} - 1 \quad (4)$$

Equation 3 can be separated again in the time domain by sequence.

$$X[2r] = \sum_{n=0}^{(N/2)-1} x[n]W_N^{2nr} + \sum_{n=N/2}^{N-1} x[n]W_N^{2nr} \quad (5)$$

Substituting n to $n + \frac{N}{2}$ for the second term in Equation 5 yields

$$X[2r] = \sum_{n=0}^{(N/2)-1} x[n]W_N^{2nr} + \sum_{n=0}^{(N/2)-1} x[n+\frac{N}{2}]W_N^{2r[n+(N/2)]}. \quad (6)$$

Owing to the periodicity of the twiddle factor, $W_N^{2r[n+(N/2)]} = W_N^{2rn}W_N^{rN} = W_N^{2rn}$, and $W_N^2 = W_{N/2}$, we can derive the equivalent equation from the original DFT computation for the even number indexed sequence after the first radix-2 decomposition is applied.

$$X[2r] = \sum_{n=2}^{(N/2)-1} (x[n] + x[n+\frac{N}{2}])W_{N/2}^{rn} \quad (7)$$

A similar method can be applied to obtain the odd number indexed sequence of the original DFT equation.

$$X[2r+1] = \sum_{n=0}^{(N/2)-1} (x[n] - x[n+\frac{N}{2}])W_N^n W_{N/2}^{nr} \quad (8)$$

Recursively applying Equations 7 and 8, we can derive the radix-2 decimation in frequency FFT as shown in Figure 2. Each cross in Figure 2 is called a butterfly and includes one complex addition, one complex subtraction, and one complex multiplication.

III. System Architecture

Based on Equations 7 and 8, the conceptual architecture of the system is shown in Figure 3. There are several components in this system.

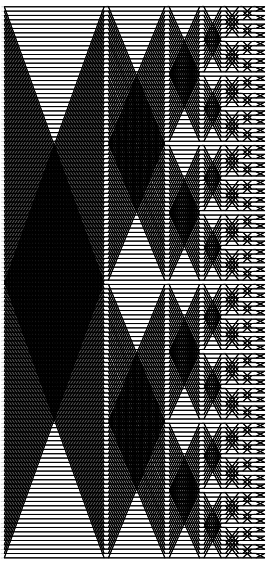


Fig. 2. Radix-2 decimation in frequency 128-point FFT calculation

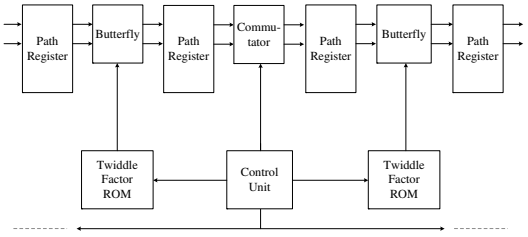


Fig. 3. Conceptual Architecture of the FFT Circuit

a. Control Unit: The control unit is simply a counter which serves as a finite state machine. It supplies control signals to the twiddle factor ROMs and the switches.

b. Butterfly: The butterfly is the key component in the FFT computation and also the most critical part related to circuit performance. It includes one complex adder, one complex subtractor, and one complex multiplier as shown in Figure 4. For the multiplier design, we select a Dadda multiplier [4], since it has the advantage of high speed, low latency, and smaller area. It is also proven that the Dadda multiplier has better performance in complexity and delay than the Wallace multiplier does [5][6].

The pipeline structure of the butterfly is a major concern, since the FFT circuit should receive the input data at a 1.5 GHz sampling rate. Therefore, the period of one FFT calculation should be no more than 86ns. In order to reach this speed, the clock frequency is required to be at least 750MHz, if we do not apply any parallel butterfly. Every butterfly unit contains five pipeline stages to meet the speed requirement.

Though there are a total of seven butterfly stages within the FFT circuit, the butterflies in the last two stages can be replaced by partial butterflies which contain only complex adders and complex subtractors, since only trivial multiplication is required in the last two stages.

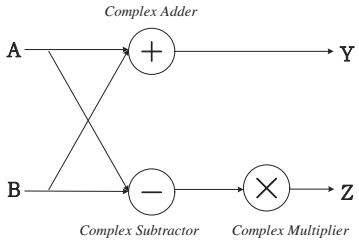


Fig. 4. Structure of a Butterfly

TABLE II
ROM SIZE FOR EACH BUTTERFLY

Butterfly	1^{st}	2^{nd}	3^{rd}	4^{th}	5^{th}
ROM Size(bits)	1536	768	384	192	96

c. Twiddle Factor ROM: The twiddle factor is stored in five hard wired ROMs. The twiddle factor is a complex number containing 12 bits in both the real and the imaginary parts. The ROM size for each butterfly is shown in Table II. Since only addition and subtraction are performed in the last two stages, the butterfly units in these two stages do not need the twiddle factor ROMs.

d. Path Register: There are plenty of path registers to store the intermediate data between every two butterflies. We do not utilize RAM as the storage of those intermediate data, which is also the most common method in other FFT circuit/processor designs [7][8], because the required clock speed, 750MHz, is so fast that RAM can not follow in 0.13 μm technology. Utilizing a shift register as an alternative can solve the problem of the speed, but we have to pay an area and power penalty.

e. Commutator: The commutator simply works as a switch to deliver the correct data sequence to the butterfly. It receives control signals directly from the control unit.

IV. Implementation and Results

This circuit is designed using Verilog and simulated with VCS. According to the simulation results, we choose the width of the input/output data to be 6 bits and the width of the intermediate data to be 12 bits for both the real and the imaginary parts. The design is synthesized with Synopsys Design Compiler and is auto placed and routed using Cadence SOC Encounter in UMC 0.13 μm 1p8m CMOS technology. The performance of the implementation results reaches the specification of 86ns. Table III shows the details of the implementation results. The chip picture is in Figure 5.

V. Conclusions

A high speed 128-point FFT/IFFT circuit which can support the next generation OFDM wireless communication with a 1.5GHz sampling rate is descirbed in this paper. The design is implemented in UMC 0.13 μm 1p8m CMOS technology. After post layout simulation, the FFT computation period meets our specification of 86 ns, and the clock frequency is 750 MHz in the worst corner case. Finally, the chip also passes post silicon scan-based test with correct functionality.

TABLE III
IMPLEMENTATION RESULTS FOR THE FFT CIRCUIT

Technology	UMC 0.13 μm 1p8m CMOS
Supply Voltage	1.2 V
Max Frequency	750 MHz
Core Power	238 mW
Gate Count	28362
I/O Wordlength	6 bits
Internal Wordlengths	12 bits
Core Size	860 μm $\times$ 860 μm
Die Size	1500 μm $\times$ 1500 μm

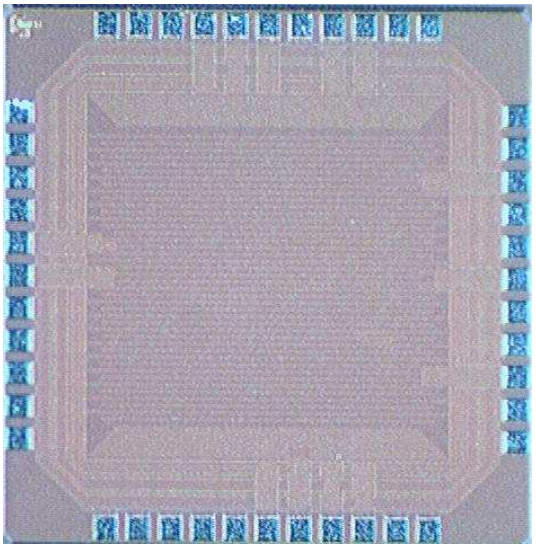


Fig. 5. Photo of the fabricated chip

References

[1] S. Fechtel, "OFDM: from the idea to implementation," *Advances in Radio Science*, vol. 3, pp. 27–37, 2005.
[2] A. Oppenheim and R. Schafer, *Discret-time signal processing*. Prentice Hall Inc., 1999.
[3] J. W. Cooley and J. W. Tuckey, "An algorithm for the machine calculation of complex Fourier series," *Math. Comput.*, vol. 19, pp. 297–301, 1965.
[4] L. Dadda, "Some schemes for parallel multipliers," *Alta Frequenza*, vol. 34, pp. 349–356, 1965.
[5] C. S. Wallace, "A suggestion for a fast multiplier," *IEEE Trans. on Computers*, vol. 13, pp. 14–17, 1964.
[6] W. J. Townsend, E. E. Swartzlander, Jr., and J. A. Abraham, "A comparison of Dadda and Wallace multiplier delays," in *SPIE Advanced Signal Processing Algorithms, Architectures, and Implementations XIII*, August 6-8, 2003, pp. 552–560.
[7] J. Kuo, C. Wen, and A. Wu, "Implementation of a programmable 64/spl sim/2048-point FFT/IFFT processor for ofdm-based communication systems," in *IEEE International Symposium on Circuits and Systems*, vol. 2, May 2003, pp. 121 – 124.
[8] K. Maharatna, E. Grass, and U. Jagdhold, "A 64-point Fourier transform chip for high-speed wireless LAN application using OFDM," *IEEE Journal of Solid-State Circuits*, vol. 39, pp. 484 – 493, March 2004.

A High Speed 128-Point Fast Fourier Transform Circuit for OFDM Systems

Tung-Yeh Wu and Jacob A. Abraham

CERC, UT Austin

Purpose of This Work

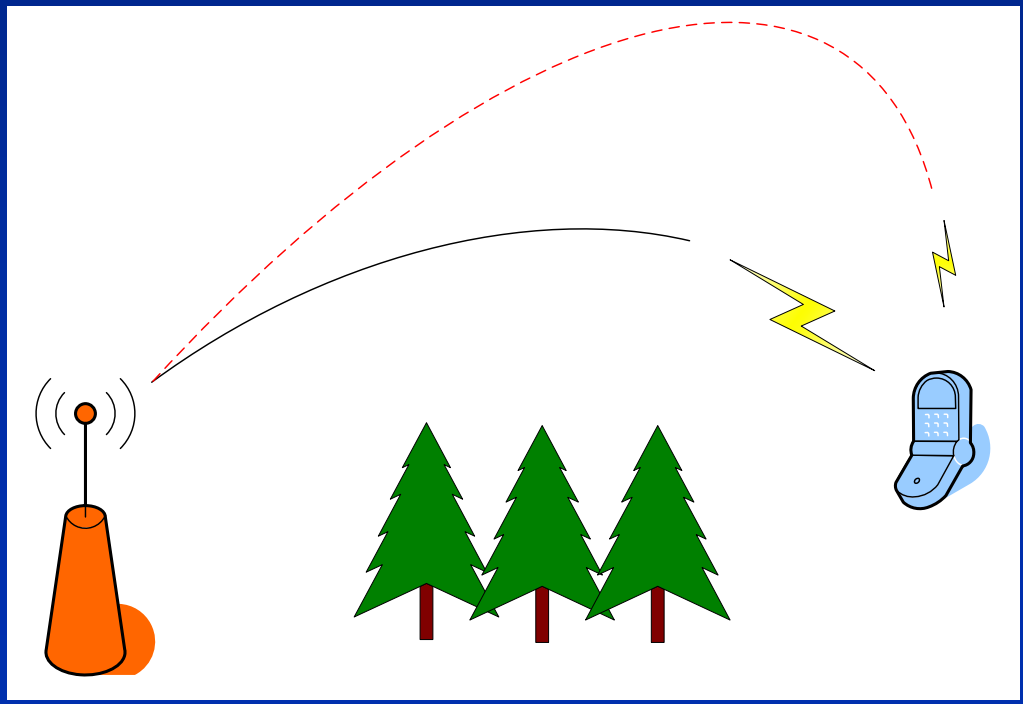
- Design A High Speed 128 Point FFT Circuit
- Handle Data with Sample Rate 1.5 GS/s
- 86 ns for One FFT Calculation
- UMC 0.13 μm 1p8m CMOS Technology

Outline

- Introduction
  - OFDM
  - FFT
- Design Issues
- Implementation Results

Increase Data Rate for Wireless Communication

- Increase Symbol Rate
  - Limited by Inter Symbol Interference (ISI)
- Put More Information into One Symbol

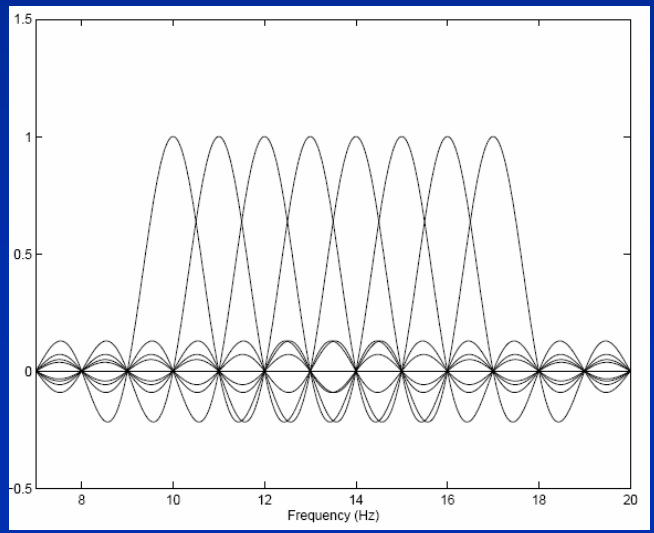


Orthogonal Frequency Division Multiplexing

- Commonly Known as OFDM
- Multicarrier Modulation Technique
  - Orthogonal subcarriers
- Adopted in Current Communication Standard
  - 802.11 WLAN
  - 802.15 UWB WPAN

Benefits of OFDM

- High Data Rate with Long Symbol Period
  - Good Inter Symbol Interference (ISI) Immunity
- Efficient Bandwidth Usage



[Jeffery G. Andrew]

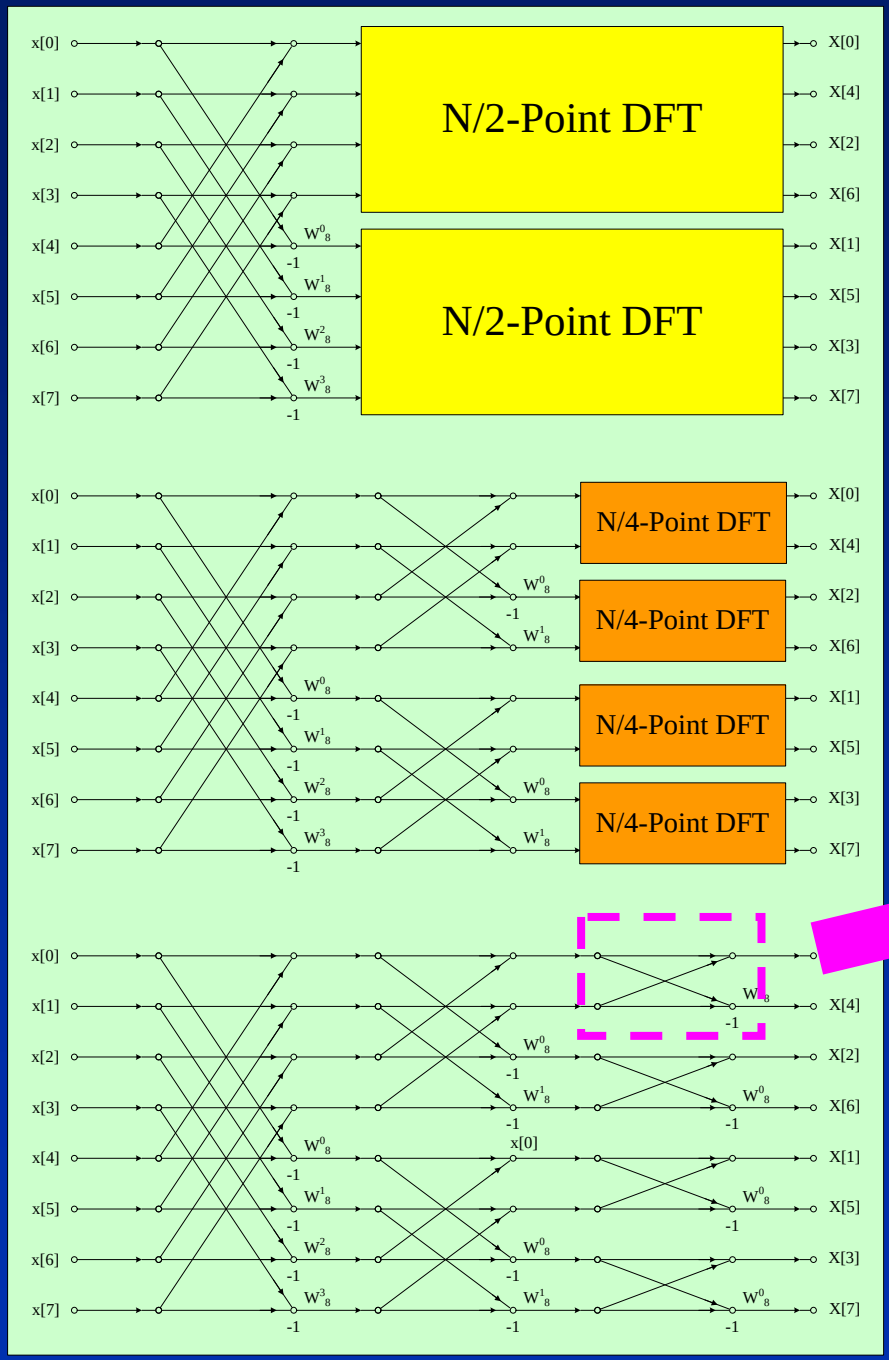
Fast Fourier Transform

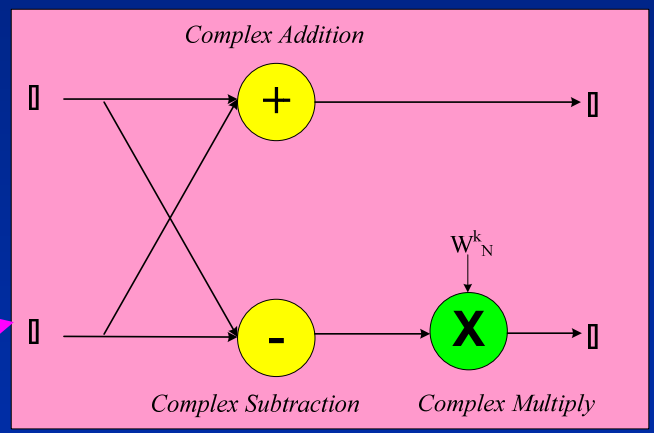
- An Efficient Way to Calculate Discrete Fourier Transform

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, k = 0,1,..., N-1$$

$$W_N^{kn} = e^{-j(\frac{2\pi}{N})kn}$$

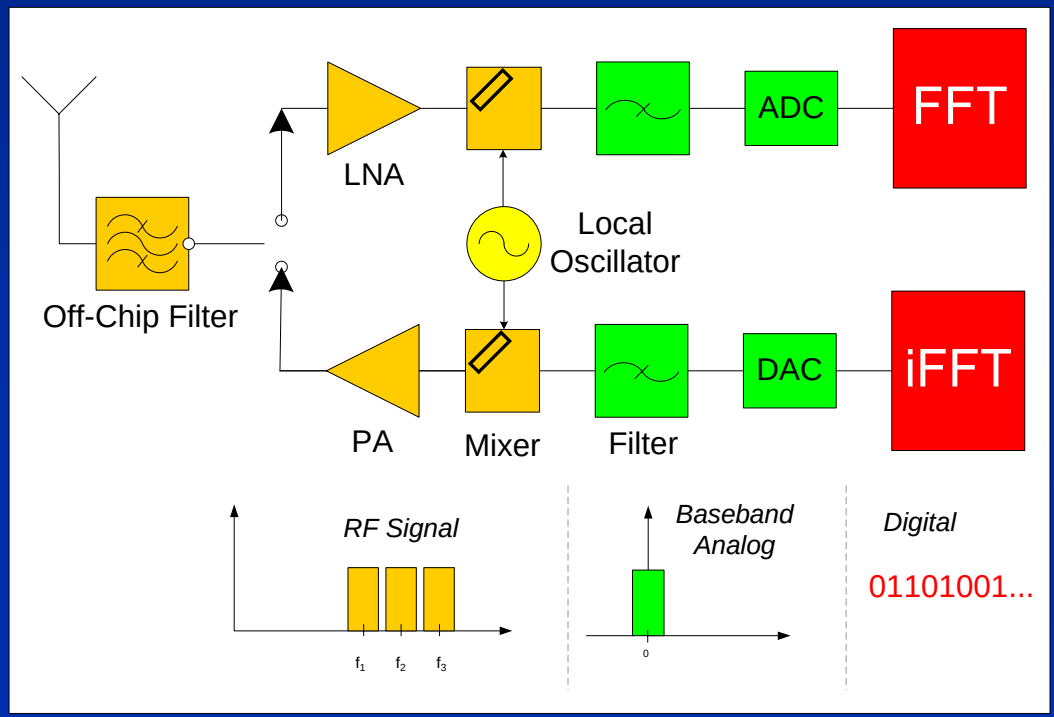
- Split the DFT to Smaller DFT Recursively
- Options
  - Radix number: Radix-2, Radix-4, …
  - Decimation-in-frequency or decimation-in-time


x[0]
x[1]
x[2]
x[3]
x[4]
x[5]
x[6]
x[7]
N/2-Point DFT
N/2-Point DFT
X[0]
X[4]
X[2]
X[6]
X[1]
X[5]
X[3]
X[7]
W^0_8
W^1_8
W^2_8
W^3_8
-1
N/4-Point DFT
N/4-Point DFT
N/4-Point DFT
N/4-Point DFT
x[0]


Complex Addition
+
-
X
W^k_N
Complex Subtraction
Complex Multiply

FFT in OFDM

- FFT/IFFT Modulates Signal to Sub-Carriers
  - No Mixer Required

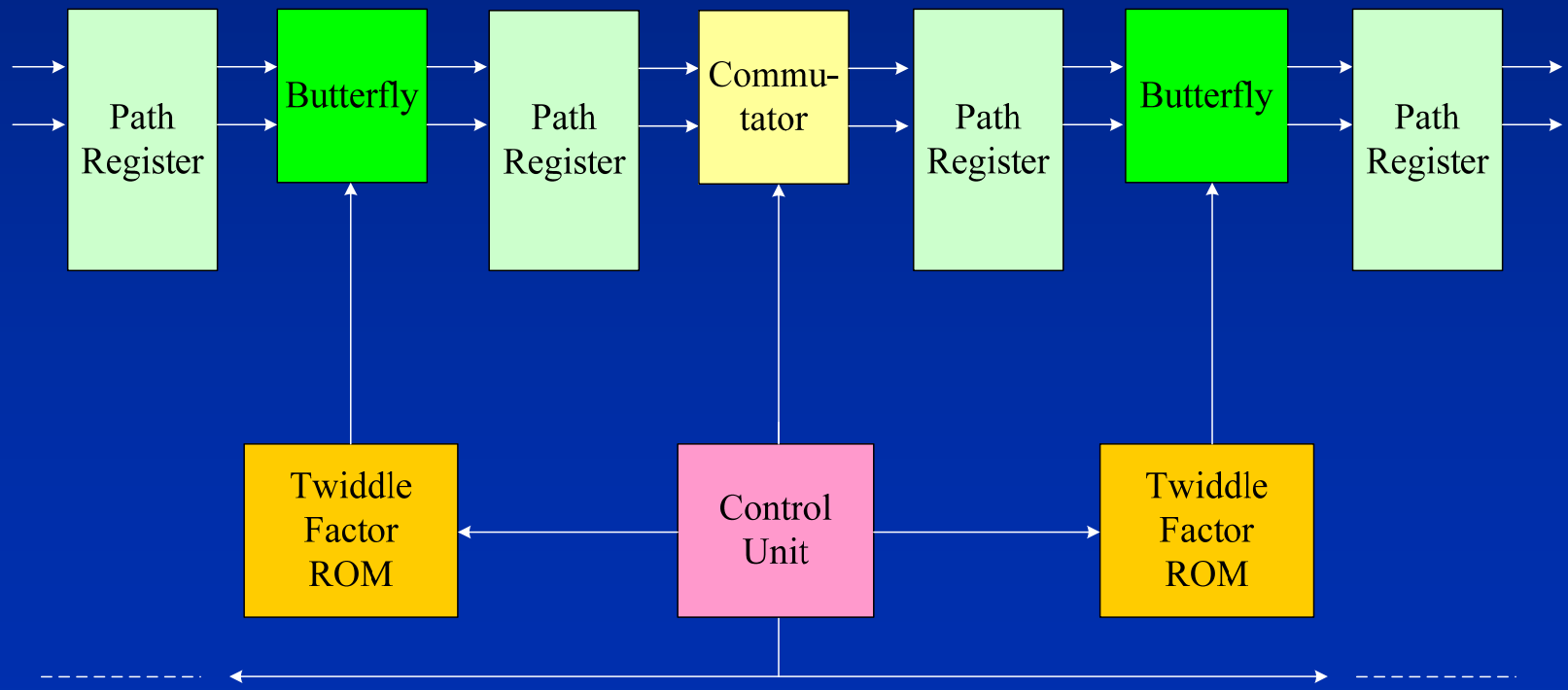


FFT Requirement

Application	802.11	802.15	Our Work
FFT Size	64	128	128
T_{FFT}	3.2 μs	312.5 ns	86 ns

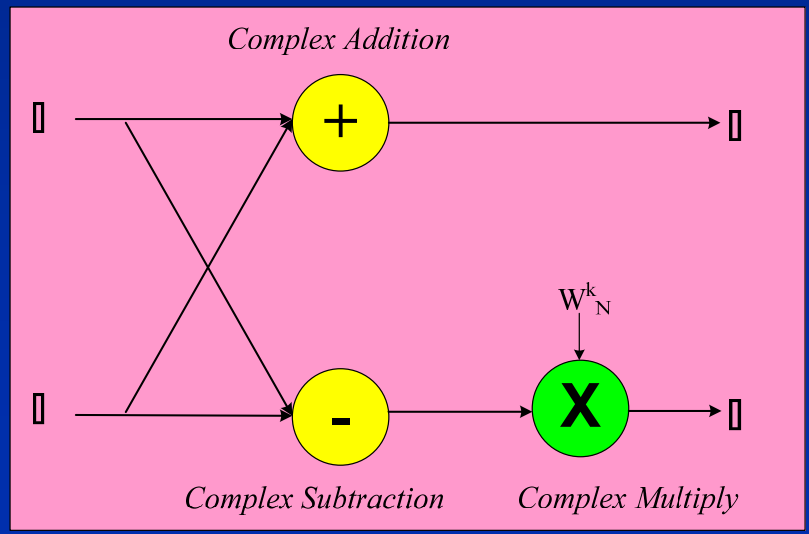
FFT Design

- Pipelined-Based Structure
- Radix-2 Decimation-in-Frequency



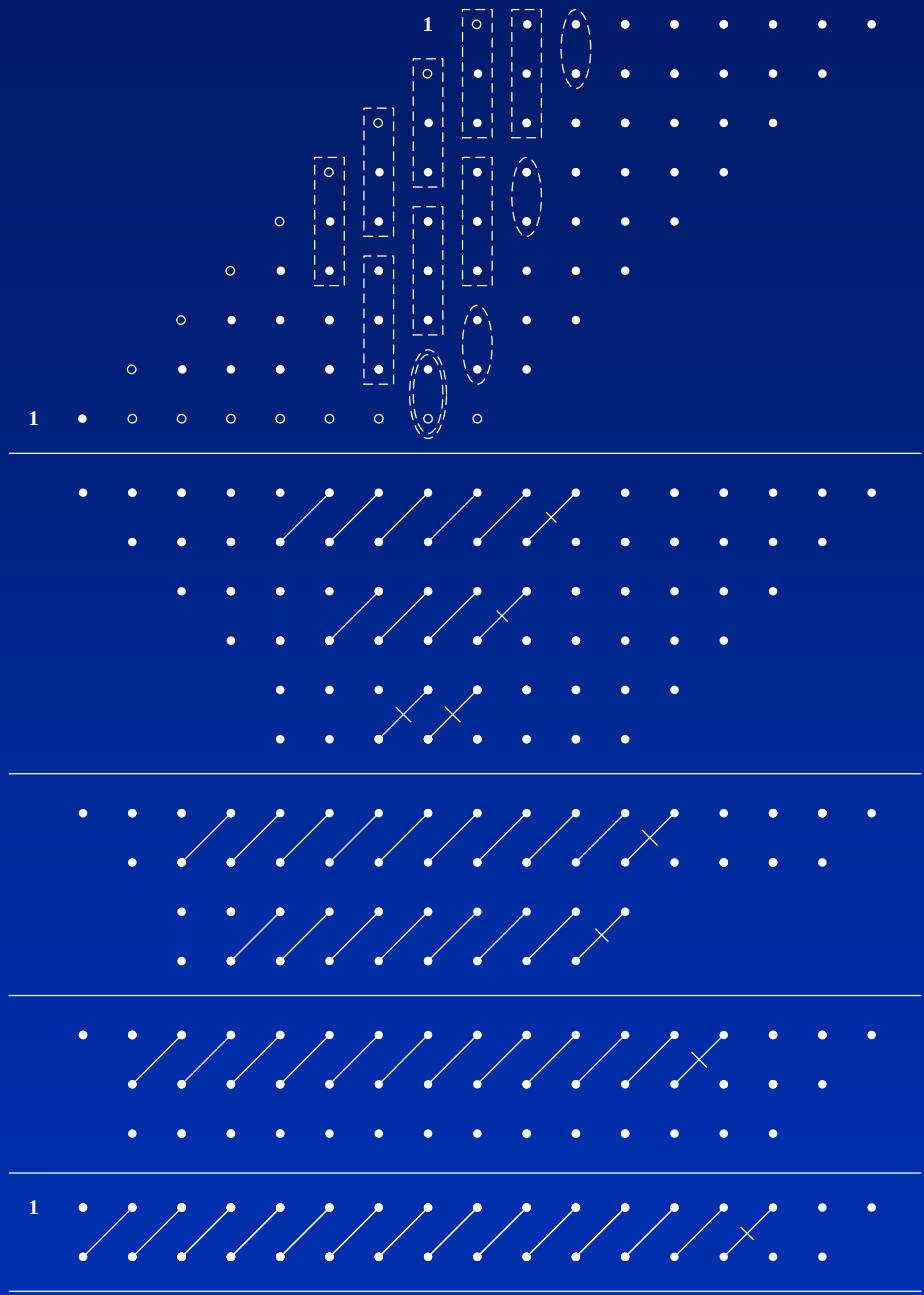
Butterfly Design

- Most Critical Component
- No Multiplier in the Last Two Stages
- Performance Directly Relates to the Multiplier
  - Dadda Multiplier



Dadda Multiplier

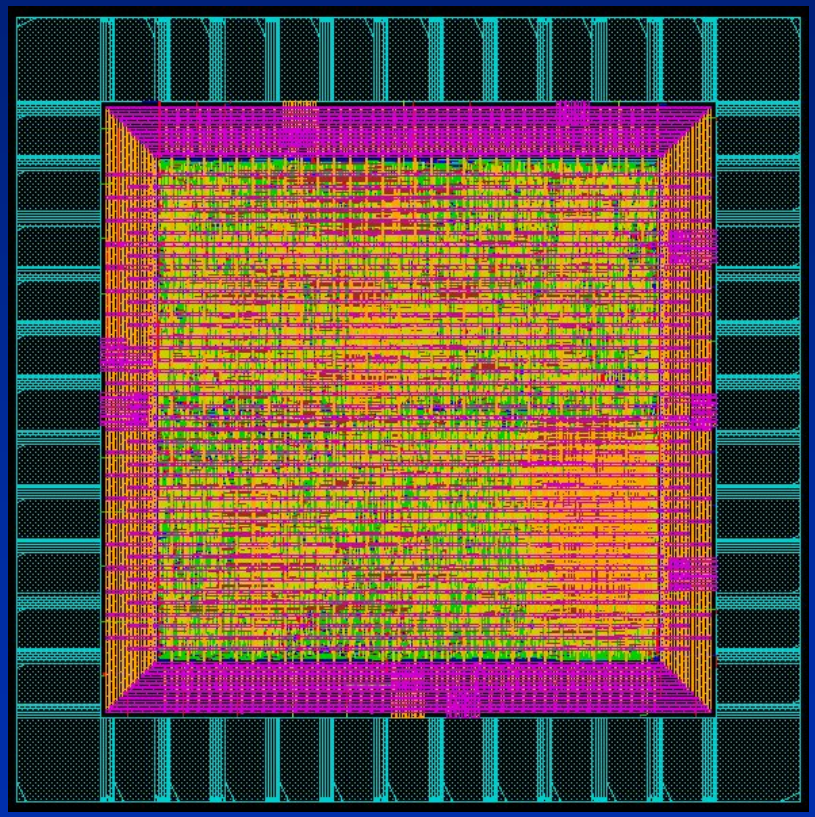
- Generate the Partial Product Array
- Reduce the Size of Array by Carry Save Addition
  - As late as possible
- Only One Complete Addition Is Required

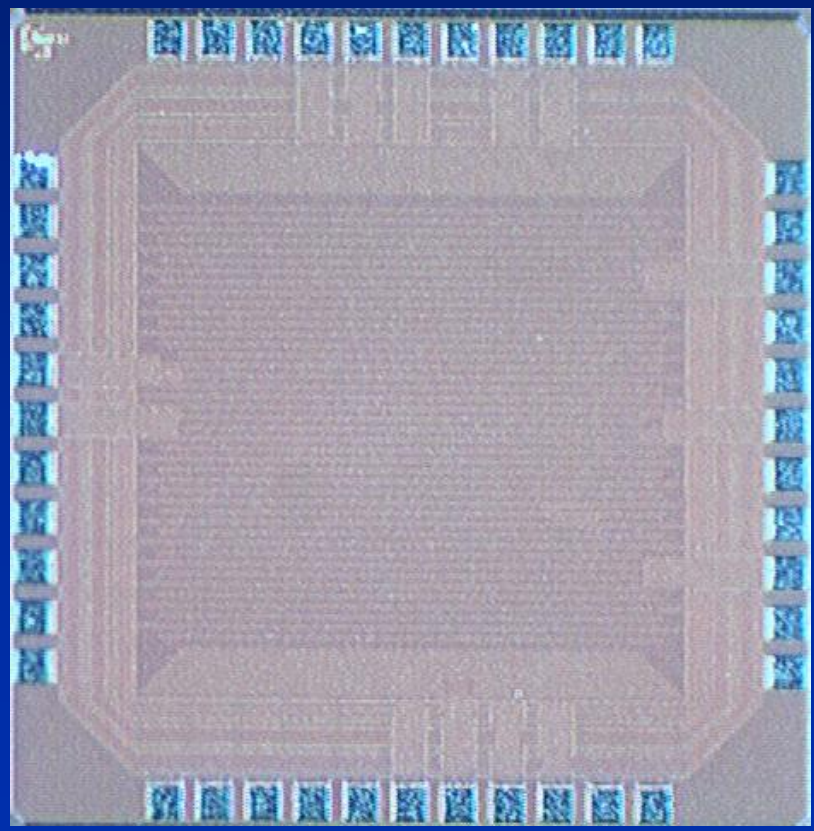

1
1
1

Implementation Result

Technology	UMC 0.13 μm 1p8m CMOS
Supply Voltage	1.2V
Max Frequency	750 MHz
Core Power	238 mW
Gate Count	28362
I/O Wordlength	6 bits
Internal Wordlength	12 bits
Core Size	860 μm × 860 μm
Die Size	1500 μm × 1500 μm

Layout and Photo of Design





Conclusion

- Propose A High Speed FFT Circuit
  - 128 points
  - 1.5 GS/s
  - 86 ns per FFT calculation
  - Pipelined structure without parallel scheme

Thank You for Your Listening

Questions?

Dynamic Compaction with Recursive Learning for Delay Test

Zheng Wang D. M. H. Walker

Dept. of Computer Science
Texas A&M University
College Station TX 77843-3112
Tel: (979) 862-4387
Fax: (979) 847-8578
Email: {wangz, walker}@cs.tamu.edu

The exponential number of paths in the circuit size limits the usage of path delay model. One alternative strategy is to test a subset of paths which contains at least one of the longest testable paths through each line or gate. In [1], an efficient automatic test pattern generation (ATPG) algorithm was developed to test the K Longest Paths Per Gate (KLPG). In [2], a dynamic compaction approach was proposed to compact paths together based on their necessary assignments during test generation. As each path is generated, its necessary assignments are checked against a pool of test patterns. If the necessary assignments of the path are compatible with those in a pattern, an attempt is made to justify a vector pair to apply the test, using a PODEM-like algorithm. This algorithm was incorporated into the KLPG algorithm and significantly reduced pattern count (up to 3x compared to static compaction) without coverage loss. The KLPG pattern counts on ISCAS89 and two industrial circuits are competitive with those of transition fault tests generated by a commercial ATPG. This indicates that it is possible to detect small delay defects without a blow-up in pattern count.

We have found that in some circuits, the final justification step during dynamic compaction fails at a high rate, leading to high CPU times. In other words, there is still a significant chance that a path with necessary assignments compatible with a pattern may still fail the justification check. To make our dynamic compaction approach practical for industrial use, we must drastically reduce CPU time.

Recursive learning [3] is widely used for test generation, design verification, as well as redundancy identification. It is a general method in the sense that it is not restricted to any logic alphabet and can be recalled recursively to find all necessary assignments. This paper proposes an improved dynamic compaction algorithm with recursive learning, which tries to identify more necessary assignments in the circuit for each path, so that the path to test pattern matching using necessary assignments is more accurate. These necessary assignments also reduce the search effort required in final justification. Recursive learning is called whenever a new path is generated or a pattern in POOL is updated. Large depth of recursion is in a sense not practical because CPU time is exponential in depth. However, since it's most likely that unknown necessary assignments must lie in the "logic neighborhood" of the know logic necessary assignments from which they are caused, it can be expected that the maximum depth of recursion to determine all necessary assignments is relatively low in general. So only a few depths of recursion need to be considered to derive the indirect necessary assignments of level 0. In our implementation we consider maximum depth 3. Our experiments show that by using recursive learning, the failure rate of final justification during dynamic compaction can be significantly reduced. Our results also show that our PODEM-like final justification algorithm struggles with the large number of necessary assignments in highly-compacted test patterns. Future research will explore more advanced search algorithms suitable for this problem.

Keywords: *delay test, path delay fault, dynamic compaction, recursive learning, test generation, ATPG*

[1] W. Qiu, J. Wang, D. M. H. Walker, D. Reddy, X. Lu, Z. Li, W. Shi and H. Balachandran, "K Longest Paths Per Gate (KLPG) Test Generation for Scan-Based Sequential Circuits," *IEEE Int'l Test Conf.*, Oct. 2004, pp. 223-231.

[2] Z. Wang and D.M.H. Walker, "Dynamic Compaction for High Quality Delay Test", *IEEE VLSI Test Symposium (VTS)*, 2008

[3] W. Kunz and D.K. Pradhan, "Recursive Learning: A New Implication Technique for Efficient solution to CAD Problems – Test, Verification, and Optimization", *IEEE trans. on Computer-Aided Design*, Sep. 1994, Vol. 13, No. 9, pp.1143-1158

Migration of Cell Broadband Engine from 65nm SOI to 45nm SOI

Presenter: Scott Cottier

IBM System & Technology Group, Austin, TX

Authors

- Osamu Takahashi[1]
- Chad Adams[2]
- David Ault[1]
- Erwin Behnen[1]
- Owen Chiang[1]
- Scott R. Cottier[1]
- Paula Coulman[1]
- James Culp[3]
- Gilles Gervais[1]
- Michael S. Gray[4]
- Yasuhito Itaka[5]
- Christopher J. Johnson[2]
- Fumihiro Kono[5]
- Lisa Maurice[1]
- Kevin W. McCullen[4]
- Lam Nguyen[1]
- Yoichi Nishino[6]
- Hiromi Noro[5]
- Juergen Pille[7]
- Mack Riley[1]
- Mike Shen[1]
- Chiaki Takano[6]
- Shunsaku Tokito[6]
- Tina Wagner[3]
- Hiroshi Yoshihara[6]

[1] IBM Systems and Technology Group, Austin TX
[2] IBM Systems and Technology Group, Rochester, MN
[3] IBM Systems and Technology Group, Hopewell Junction, NY
[4] IBM Systems and Technology Group, Essex Junction, VT
[5] Toshiba America Electronic Components, Austin, TX
[6] Sony Computer Entertainment of America, Austin, TX
[7] IBM Systems and Technology Group, Boeblingen, Germany

OUTLINE

- Migration Objectives
- Technologies Overview
- Migration Flow
- DfM Enhancement
- Power Optimization
- Challenges of SRAM Design Migration
- Area Scaling
- Die Photo
- Conclusions

Migration Objectives & Constraints

- Design migration to an advanced technology
  - Technologies with fairly scalable design rules
- Effective design migration
  - Automated process wherever applicable
- 30% power reduction
- 30% area reduction
- Design for Manufacturability (DfM) improvement
- Must preserve cycle-by-cycle machine behavior
  - Logic
  - Frequency

Two Technologies Overview

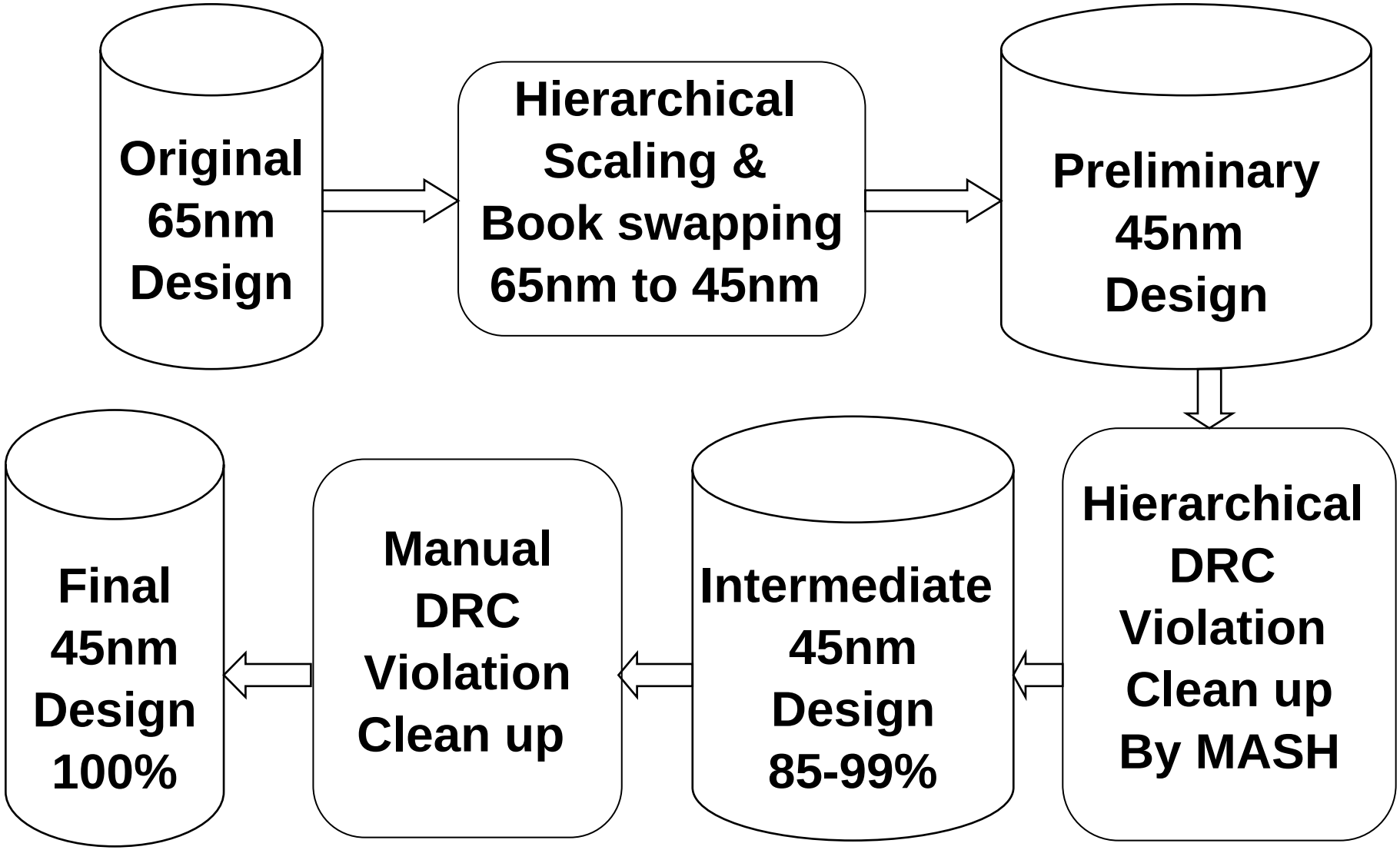
Technology		65nm SOI	45nm SOI	Scaling
		CMOS Technology on p- SOI Substrate	CMOS Technology on p- SOI Substrate	
Used Device Type	Thin Ox Hvt	yes	yes	
	Thin Ox Rvt	yes	yes	
	Thick Ox Rvt	yes	yes	
Thin Ox Tox		1.12 nm	1.16 nm	
Thick Ox Tox		2.35 nm	2.50 nm	
Nominal Supply (Thin Ox)		1.0 V	0.9-1.0 V	
Nominal Supply (Thick Ox)		1.5 V	1.5-1.8 V	
M1 Minimum Width		0.1* um	0.076* um	0.76
M1 Minimum Spacing		0.1* um	0.076* um	0.76
Metal Layers		10	10	
SRAM Cell Area		0.700 um^2	0.404 um^2	$(0.76)^2$

* Cell/B.E. design specific. Not technology specific

Design Migration Overview

- Automated schematic migration
  - Preserve the circuit topology / Scale applicable device sizes
- Physical design migration
  - Parameterized cells by programming change
  - Common cells (flip-flop & local clock buffer)
    - Preserve pin locations & metal blockages
    - Migration Assistant Shape Handler (MASH)
    - Scale & design rule violation correction
    - Manual correction
  - Custom cells
    - Automated process of scale / replacement & MASH
    - Manual correction
- Synthesized design migration
  - Scale the original placement information
  - Re-route with an opportunity of improving the routing

Custom Physical Design Migration Flow



DfM Enhancement

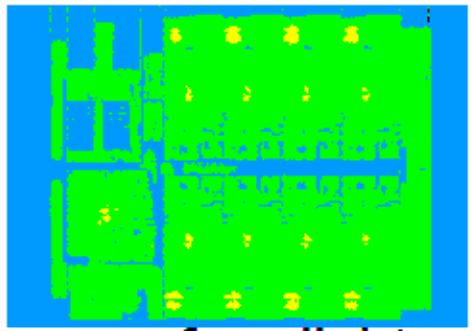
- Mostly automated process
  - MASH with DfM Focus
    - Via redundancy, via coverage, short edge fix
    - Quick turn-around-time to reflect design rule update
  - Supplemented by manual correction
- Yield-checking software
  - Design rule checker with DfM focus
    - Manufacturing systematics
    - Longer term defect reduction
    - Performance optimization
  - Calculate yield score for each checking item
    - The number of violations against the total occurrences
  - Each score must surpass the pre-defined threshold value
- Review of critical IP blocks for DfM optimization

Power Optimization

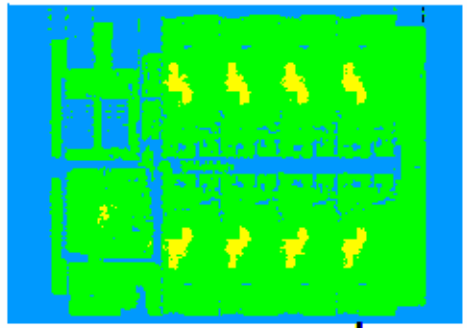
- Power supply grid optimization
  - Known design
  - Known workloads
  - Additional power C4 bumps
- Opportunity of lowering power supply
  - Must preserve the cycle-by-cycle behavior including frequency
  - Taking advantages of devices in faster technology
- Replace some dynamic circuits by static implementation
  - Some dynamic programmable logic arrays (PLA) and adder
- Minimizing DC power
  - High Vth devices as default
  - Regular Vth devices are used in critical paths only

Simulated IR Drop on Power Supply Grid for Various Workloads

daxpy



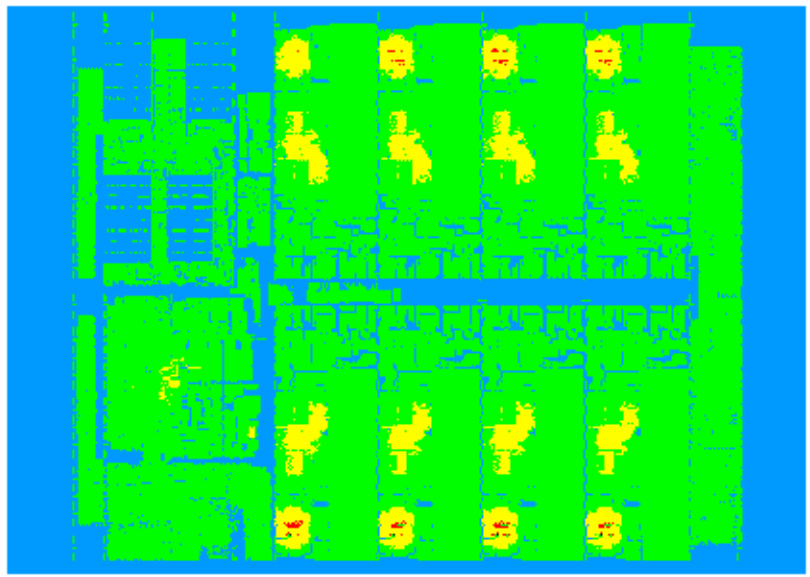
xformlight



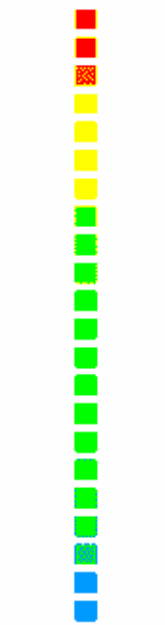
encrypt

max

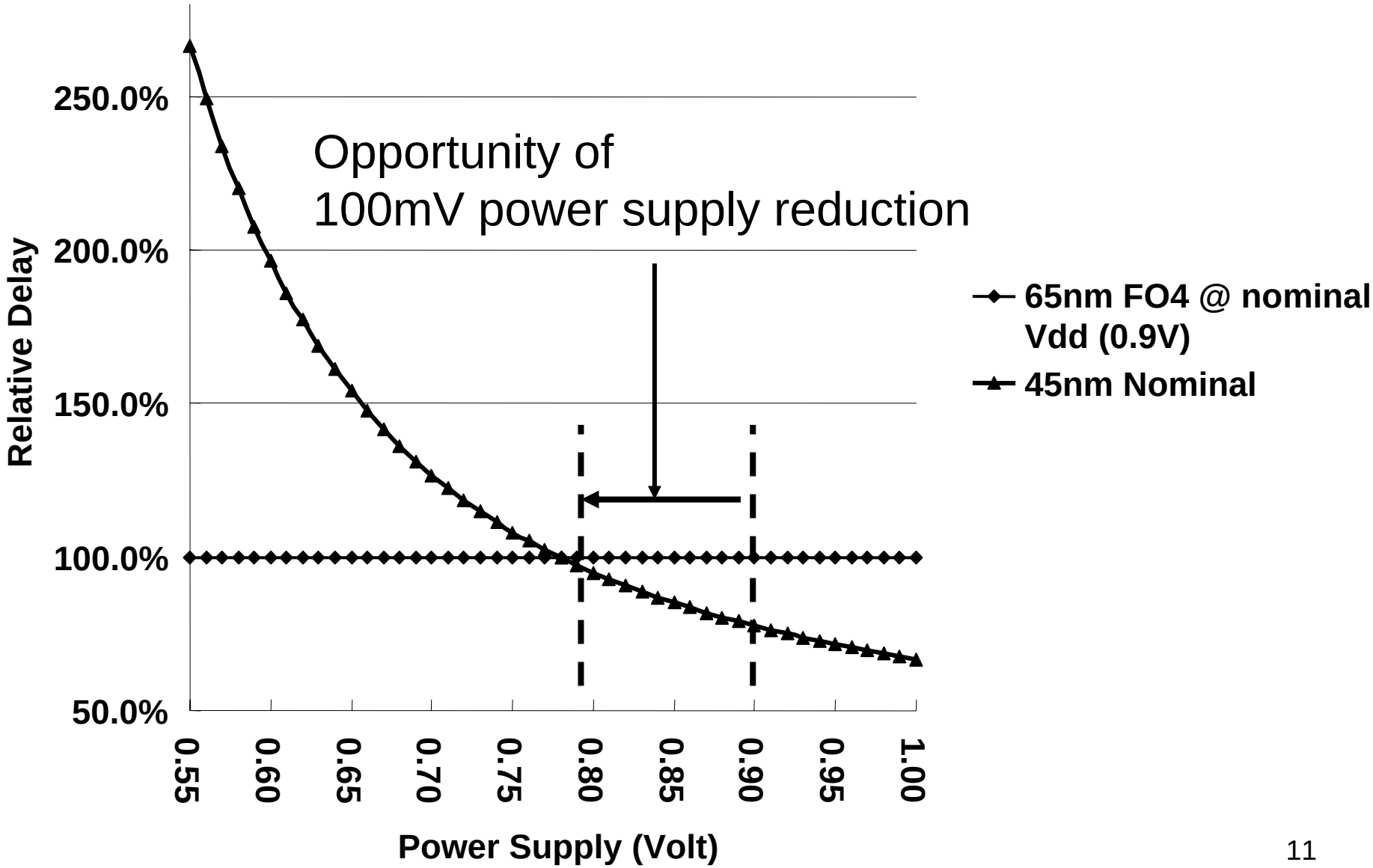
saxpy



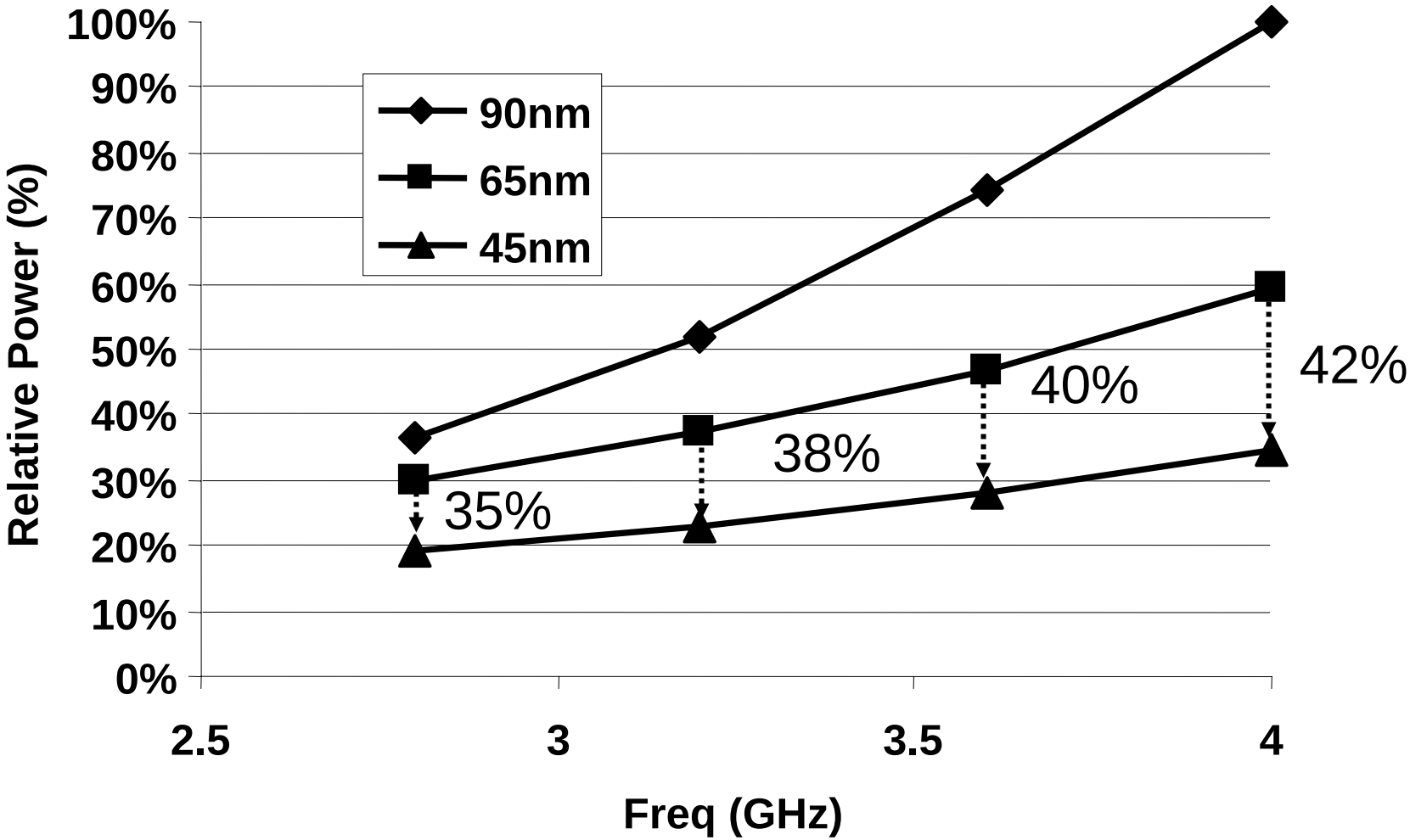
Super composition of all 5 workloads.



Relative FO4 Delay of 45nm and 65nm



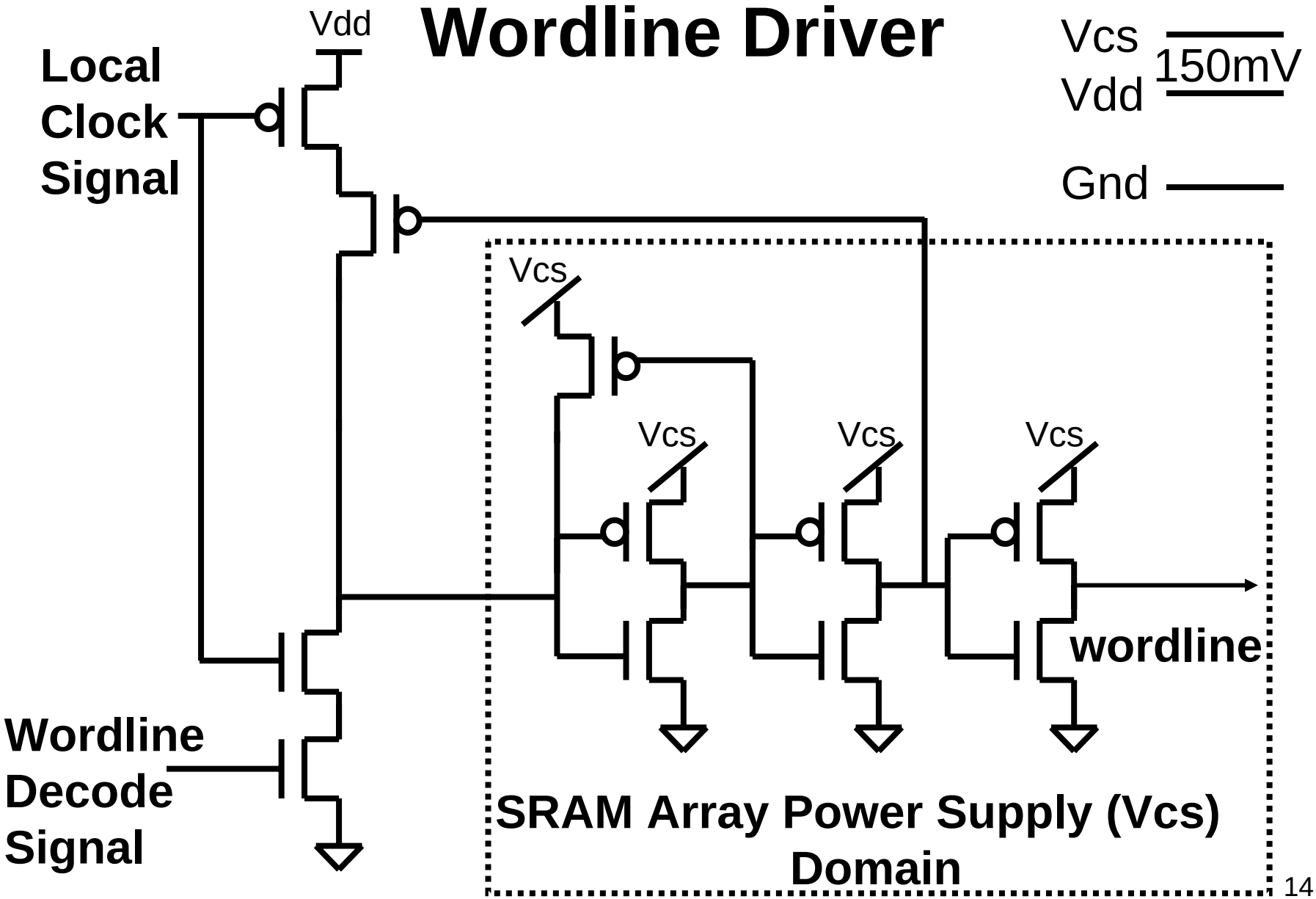
Simulated Relative Power of Cell/B.E. with Three Technologies



Challenges of SRAM Design Migration

- Lowered power supply
  - Slower operation
  - Less stable
- Smaller cell
  - More pronounced process variability
  - Less stable
- Challenges addressed
  - SRAM cell optimization
    - Normal thin oxide devices with SRAM specific implants
  - Re-optimizing the READ / WRITE circuits
    - 2007 ISSCC J. Pille et al.
  - Separate power supply (Vcs)
    - Array of SRAM cells
    - Wordline driver
  - Level shifter embedded in wordline driver

Embedded Level-Shifter in SRAM Wordline Driver



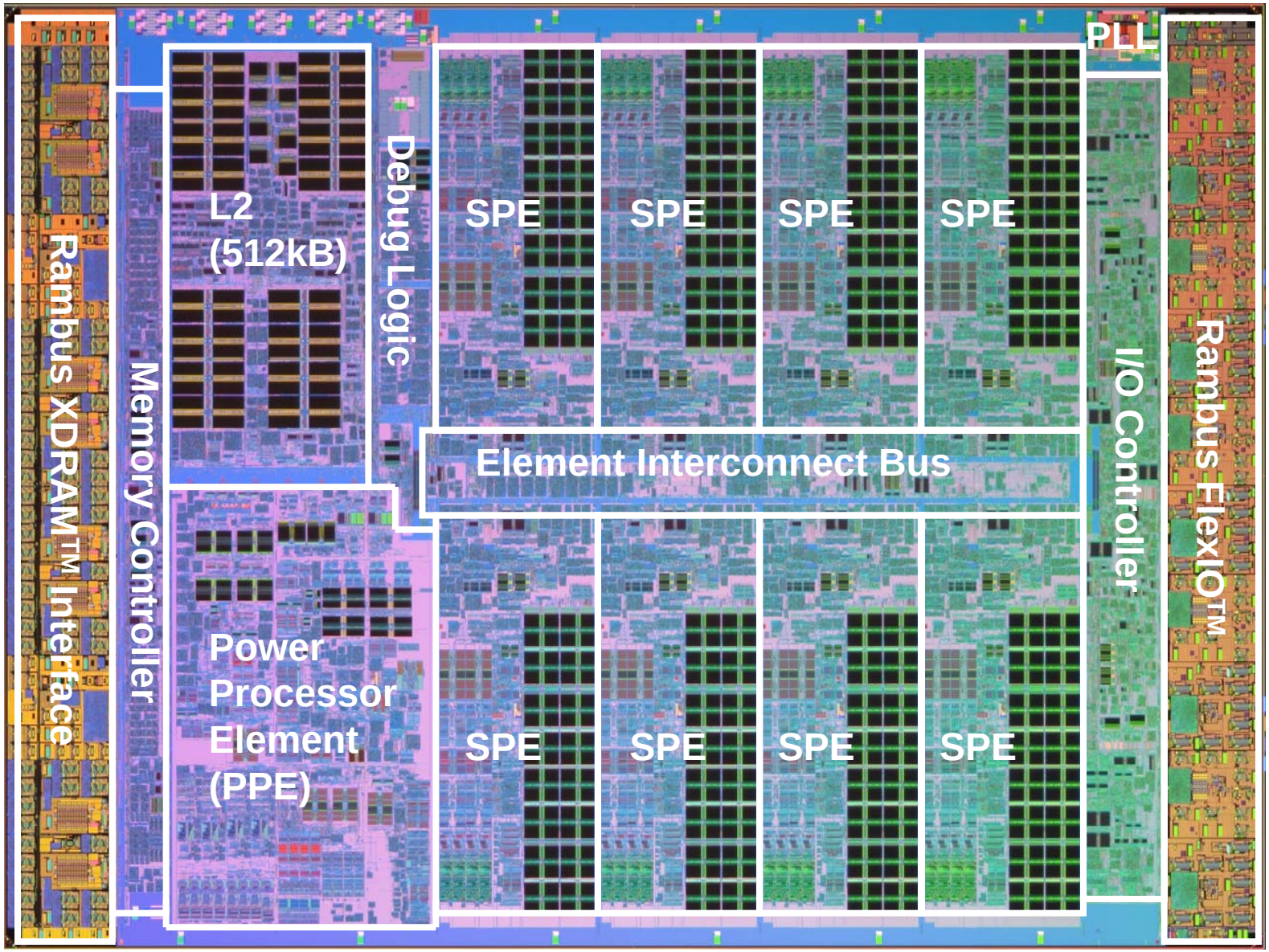
Area Scaling Observations

- Digital blocks scaled very well
- Memory arrays scaled fairly well
  - Extensive work required beyond simple scaling
  - Different technology characteristics required new circuitry and design modifications
  - Additional area for new circuitry and design changes
    - Level shifter / optimizing READ & WRITE circuits
- I/O & Analog circuits did not shrink
  - Larger area required for decoupling capacitors
- C4 bumps
  - The same pitch as the previous technology
  - This dictated the chip dimensions

Area Comparison of Cell/B.E. & its Components by Technology

Cell/B.E.					
Generation	W (mm)	H (mm)	Area (mm^2)	Scaling from 90nm	Scaling from 65nm
90nm	19.17	12.29	235.48	100.0%	
65nm	15.59	11.20	174.61	74.2%	100.0%
45nm	12.75	9.06	115.46	49.0%	66.1%
Synergistic Processor Element (SPE)					
Generation	W (mm)	H (mm)	Area (mm^2)	Scaling from 90nm	Scaling from 65nm
90nm	2.54	5.81	14.76	100.0%	
65nm	2.09	5.30	11.08	75.0%	100.0%
45nm	1.59	4.09	6.47	43.9%	58.5%
Power Processor Element (PPE)					
Generation	W (mm)	H (mm)	Area (mm^2)	Scaling from 90nm	Scaling from 65nm
90nm	4.44	6.05	26.86	100.0%	
65nm	3.50	5.60	19.60	73.0%	100.0%
45nm	2.66	4.26	11.32	42.1%	57.7%

45nm Cell/B.E. Die Photo



Conclusions

- Migration constraints & objectives
  - Effective migration from 65nm SOI to 45nm SOI
  - Must preserve cycle-by-cycle machine behavior
  - 30% power & area reduction
- Migration flow
  - Mostly automated process including DfM enhancement
- Power optimization
  - Power supply grid optimization
  - Lowering power supply
  - Exceeded the objective
- SRAM design challenges
  - Separate power supply
  - Embedded level shifter in wordline driver
- Area Scaling
  - C4 bump dictated the chip dimensions
  - Exceeded the objective