FPGAs at 28nm: Meeting the Challenge of Modern Systems-on-a-Chip

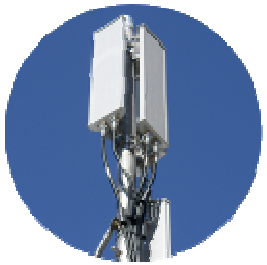
Vaughn Betz
Senior Director, Software Engineering
Altera

Overview

- Process scaling & FPGAs
  - End user demand
  - Technological challenges
- FPGAs becoming SoCs
  - Stratix V: more hard IP
  - FPGA families targeted at more specific markets
- Stratix V & 28 nm
  - Challenges & features
  - Partial reconfiguration
- Designer productivity
  - Challenges
  - Possible software stack solutions

Demand and Scaling Trends

Broad End Market Demand



Communications

- Mobile internet and video driving bandwidth at 50% annualized growth rate
- Fixed footprints

Broadcast

- Proliferation of HD/1080p
- Move to digital cinema and 4k2k

Military

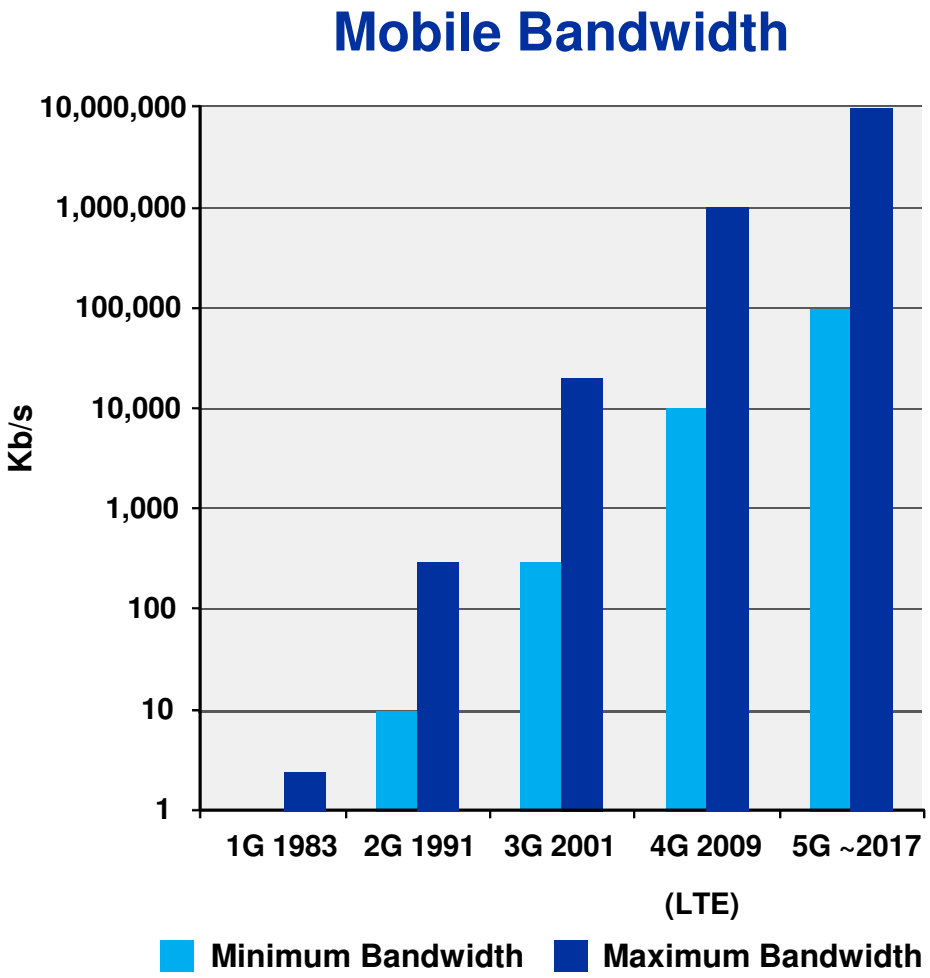
- Software defined radio
- More sensors, higher precision
- Advanced radar

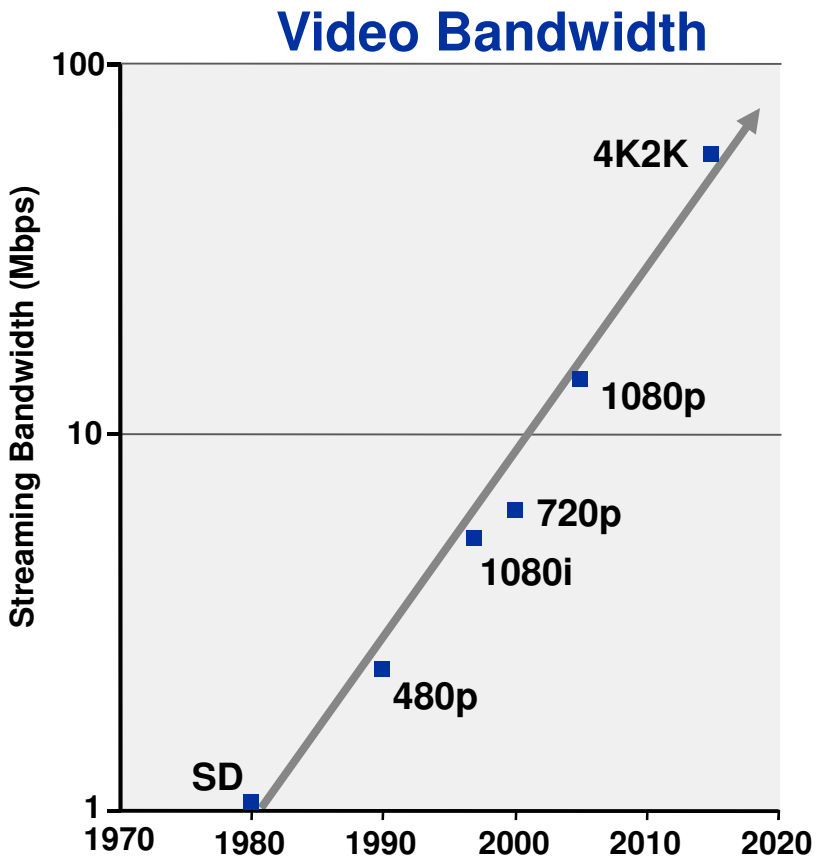
Consumer/industrial

- Smart cars and appliances
- Smart Grid

Need more processing in same footprint, power and cost

Driving Factors—Mobility and Video





Evolution of Video-Conferencing

Today

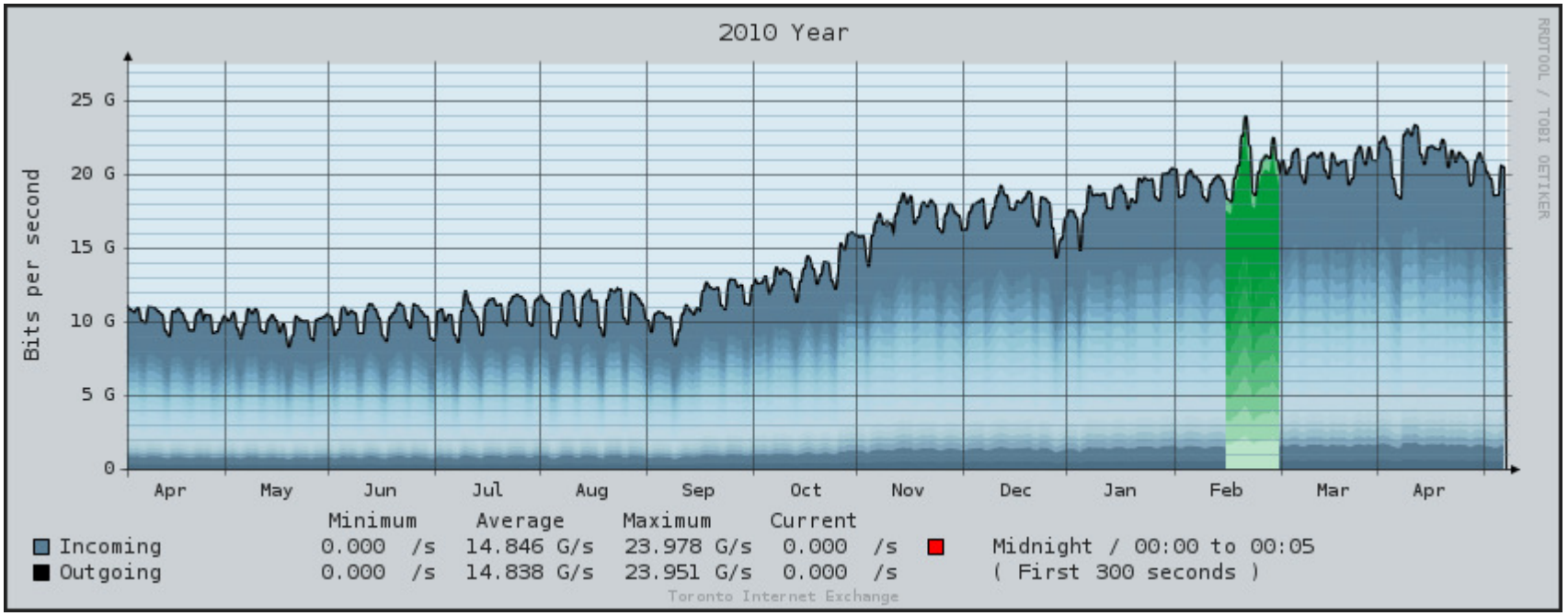
Tomorrow

- High end in 2000: 384 kbps
- Cisco telepresence: 15 Mbps

Communication Processing Needs

- More bandwidth: CAGR of 25% to 131% / year [By domain, Cisco]
- More data through fixed channel → more processing per symbol
- Security and quality of service needs → deep packet inspection

Toronto Internet Exchange (TorIX), 2009-2010 [Courtesy W. Gross, McGill]

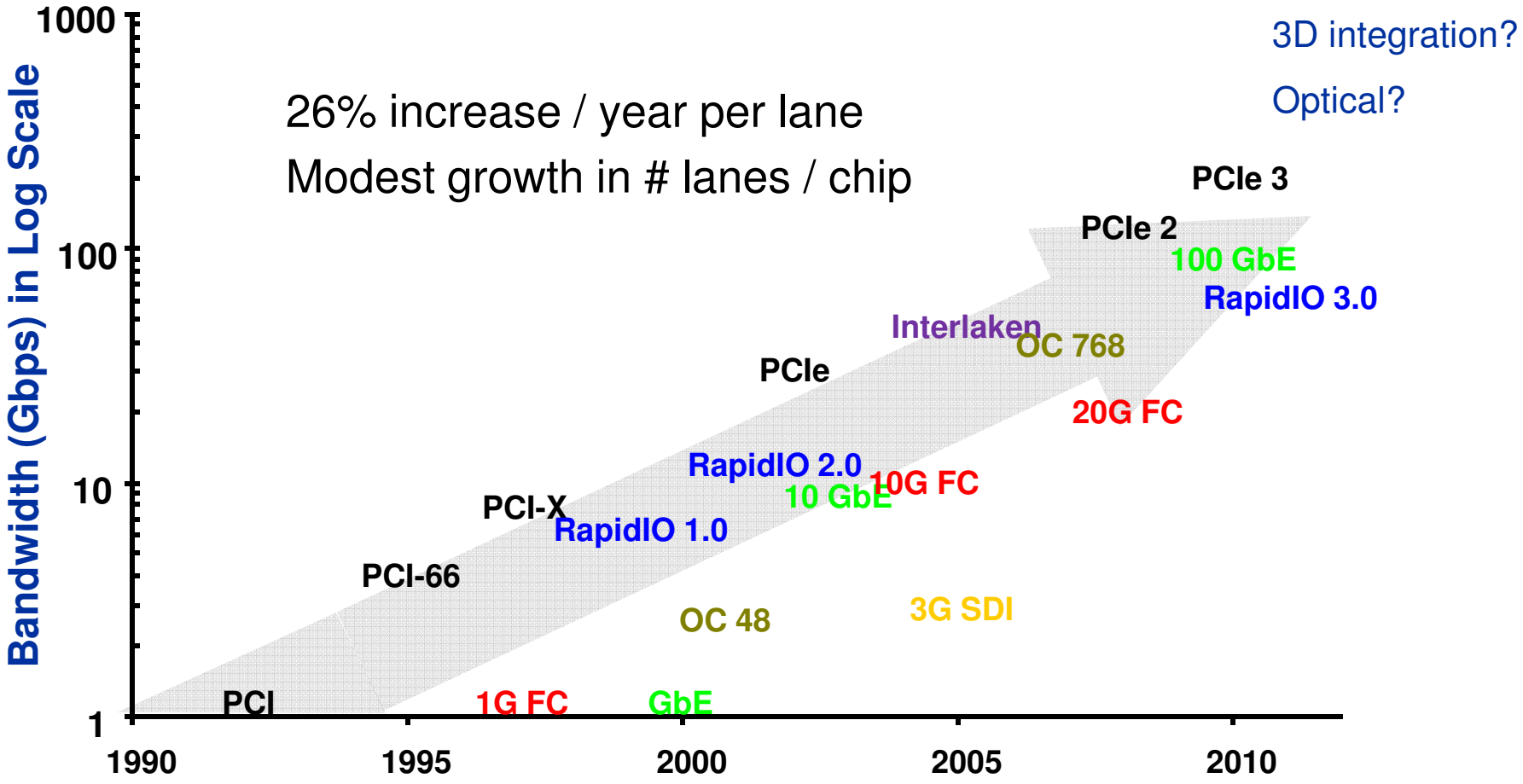


Moore's Law: On-Chip Bandwidth

- Datapath width * datapath speed
- 40% / year increase in transistor density
- 20% / year transistor speed until ~90 nm
  - Total ~60% gain / year

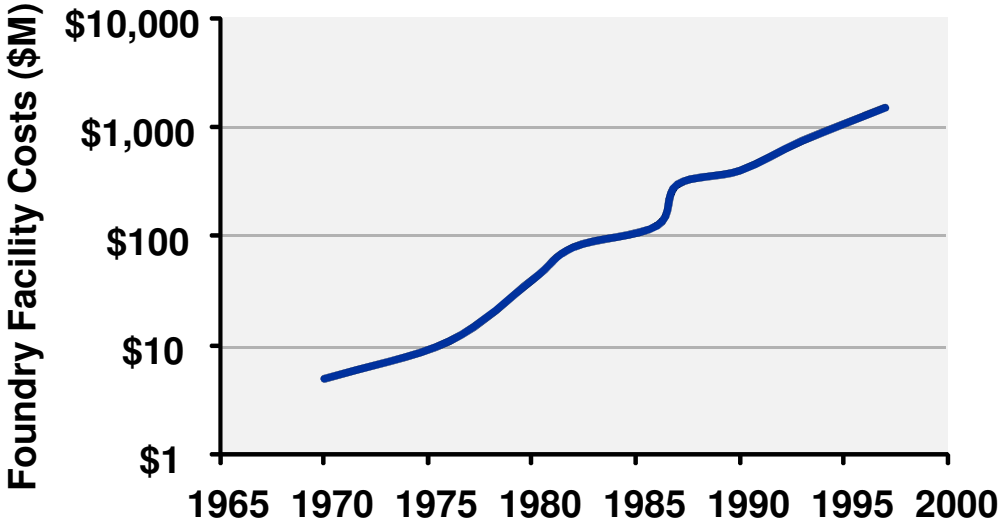
- 40 nm and beyond:
  - Little intrinsic transistor speed gain once power controlled
  - ~40% gain / year from pure scaling
  - Need to innovate to keep up with demand

Increasing I/O Bandwidth

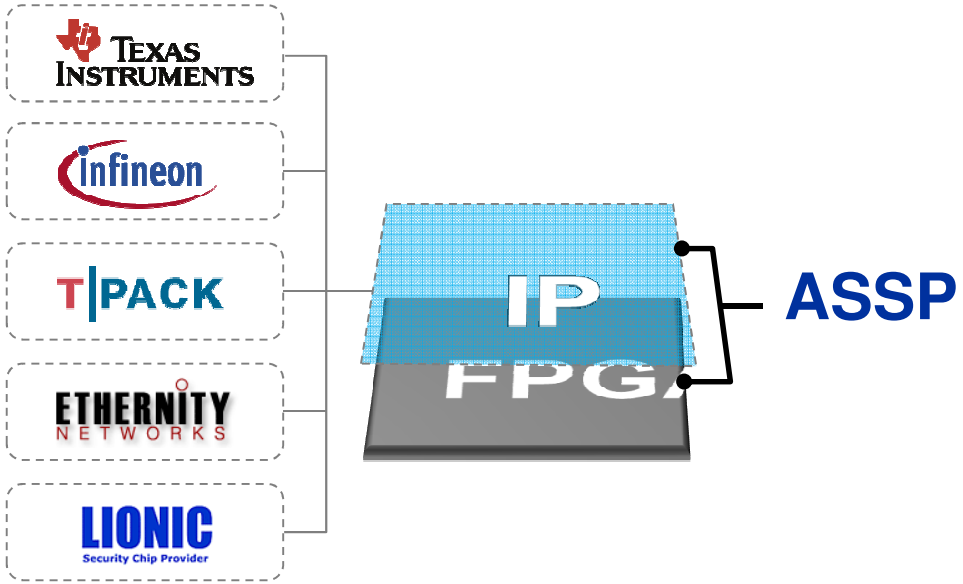


Scaling Economics

- TSMC Fab 15: $9B
  - 40 & 28 nm
- 90's fab cost → fabless industry



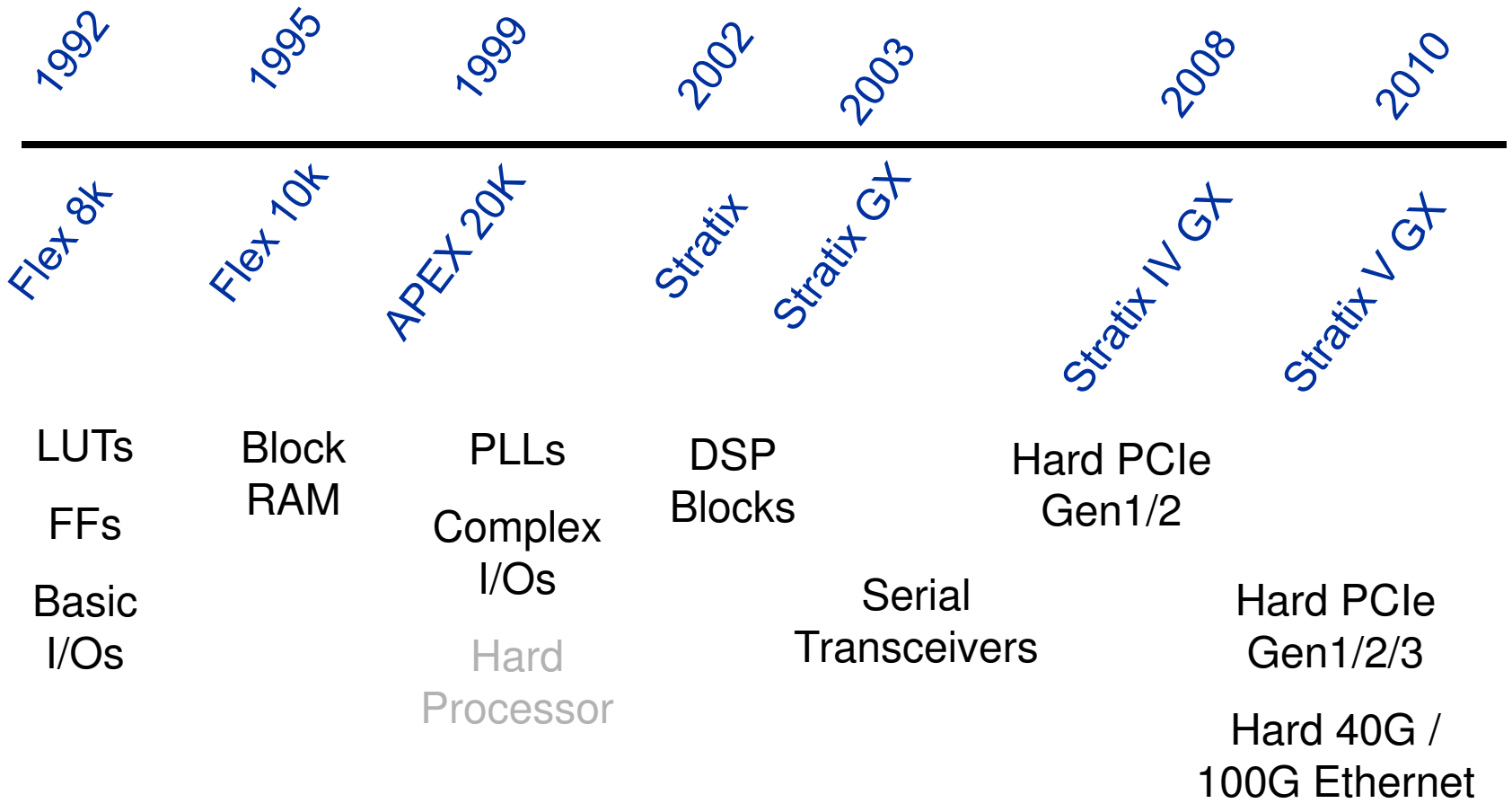
- Chip cost @ 28 nm ~$100M
- Need big market → go programmable
- "Chipless" industry emerging



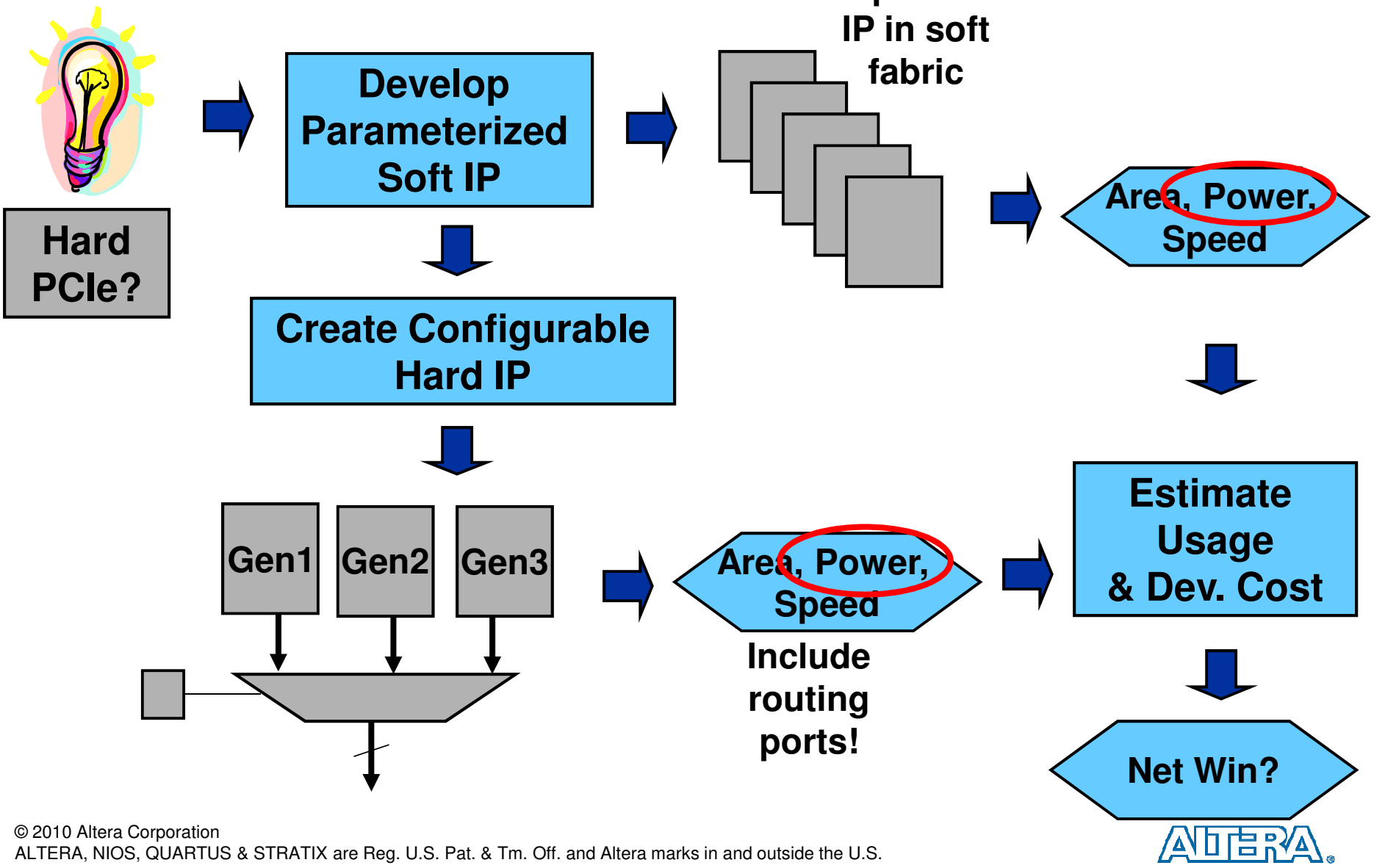
FPGAs Becoming SoCs

Example: Stratix V

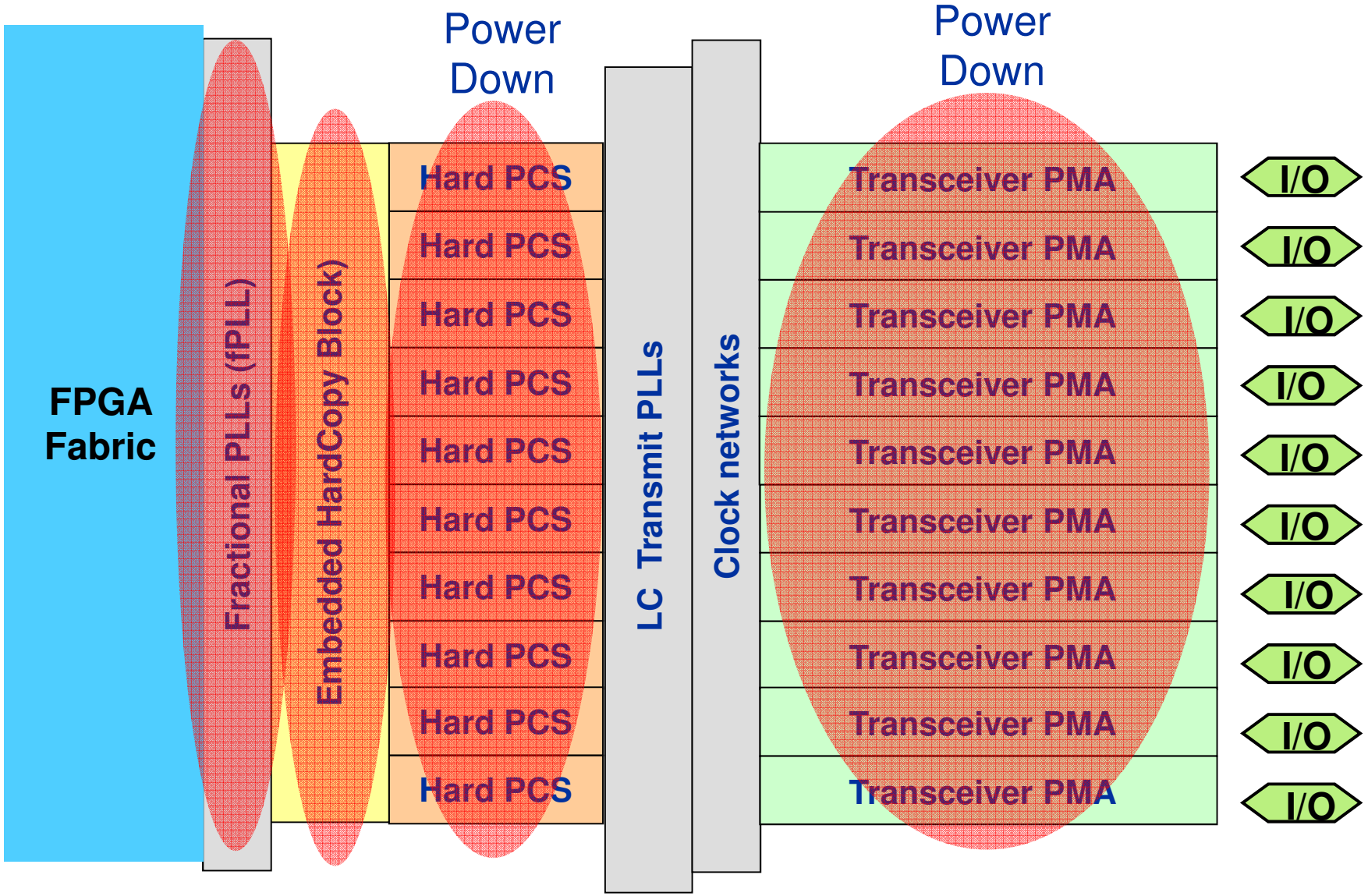
From Glue Logic to SoC



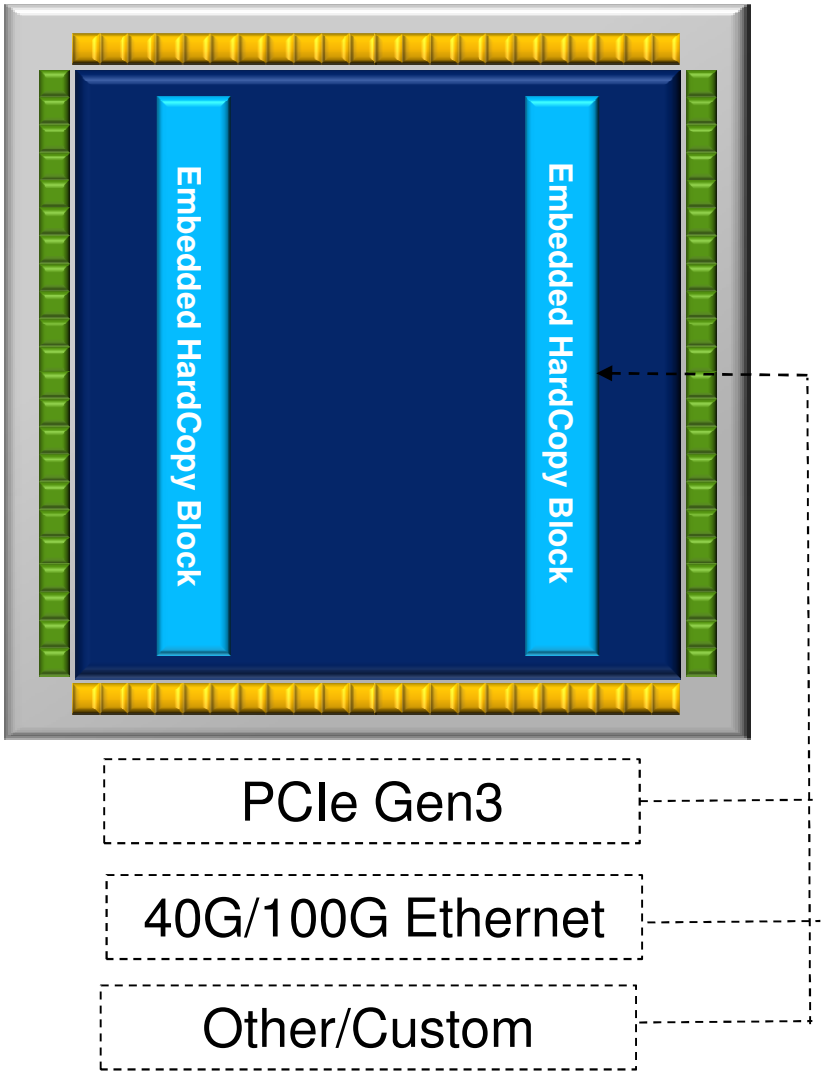
Hard Block Evaluation



Stratix V Transceivers



Embedded HardCopy Blocks



- Metal programmed: reduces cost of adding device variants with new hard IP
- 700K equivalent LEs
- 14M ASIC gates
- 5X area reduction vs. soft logic
- 65% reduction in operating power
- Very low leakage when unused

Hard IP Example: PCIe & Interlaken

Interlaken – PCI Express Switch/Bridge

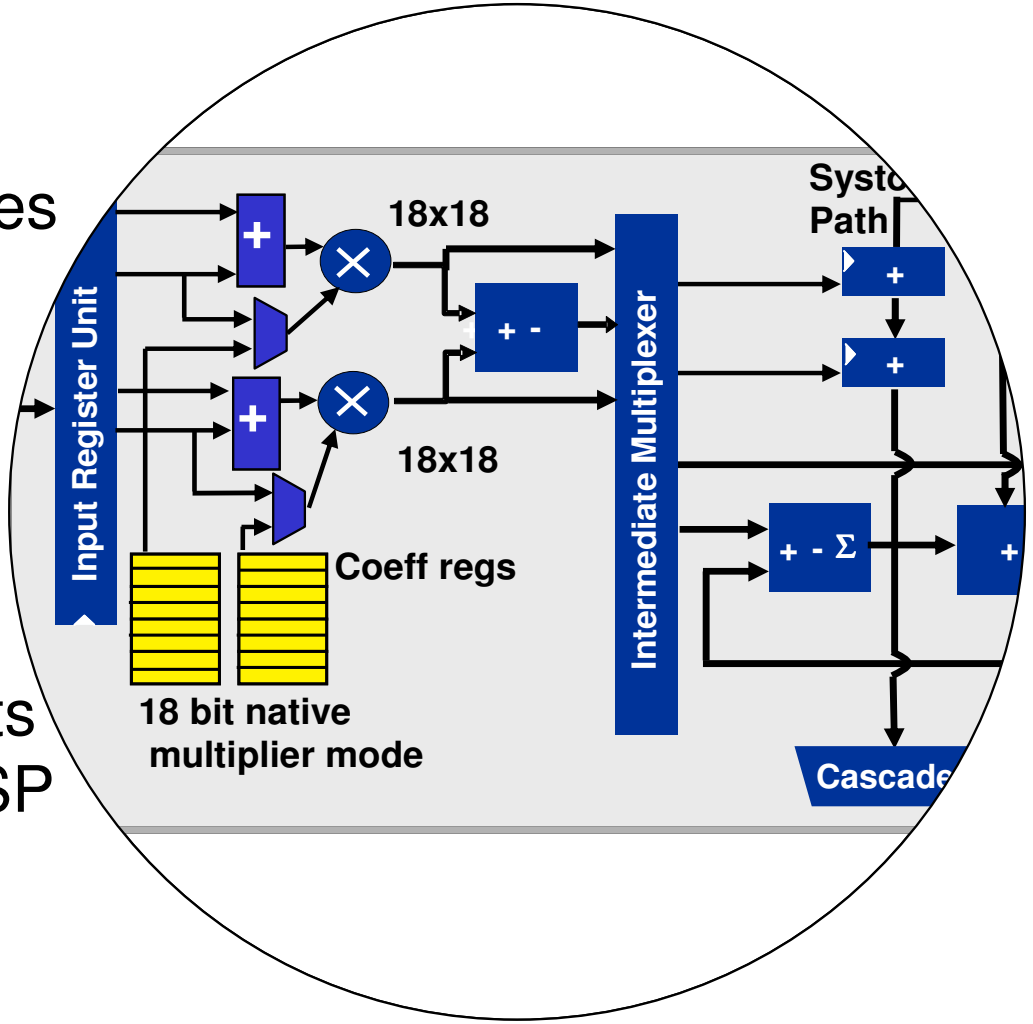
Hard IP	LE Savings
Interlaken (24 Ch @ 5K LEs)	120K LEs
PCIe Gen3 x8 (2 x 160K LEs)	320K LEs
Total LE savings	440K LEs

630K LEs + 440K LEs = 1,070K LEs

Lower power
Higher effective density
Guaranteed timing closure → ease of use

Variable-Precision DSP Block

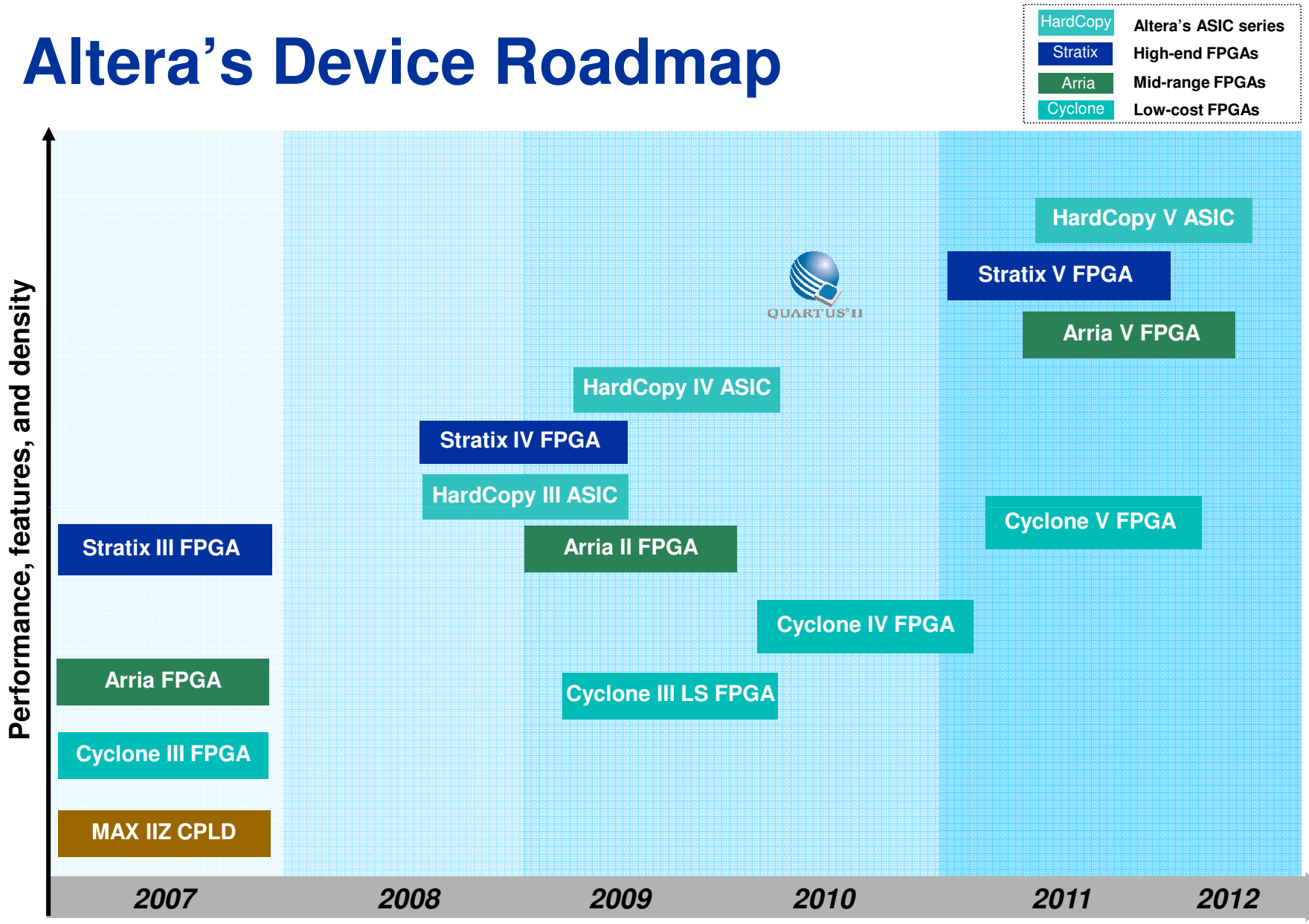
- Efficiently supports 9x9, 18x18 and 27x27 multiplies
  - 27x27 well suited to floating point
- Cascade blocks for larger multiplies
- Can store filter coefficients in register bank inside DSP



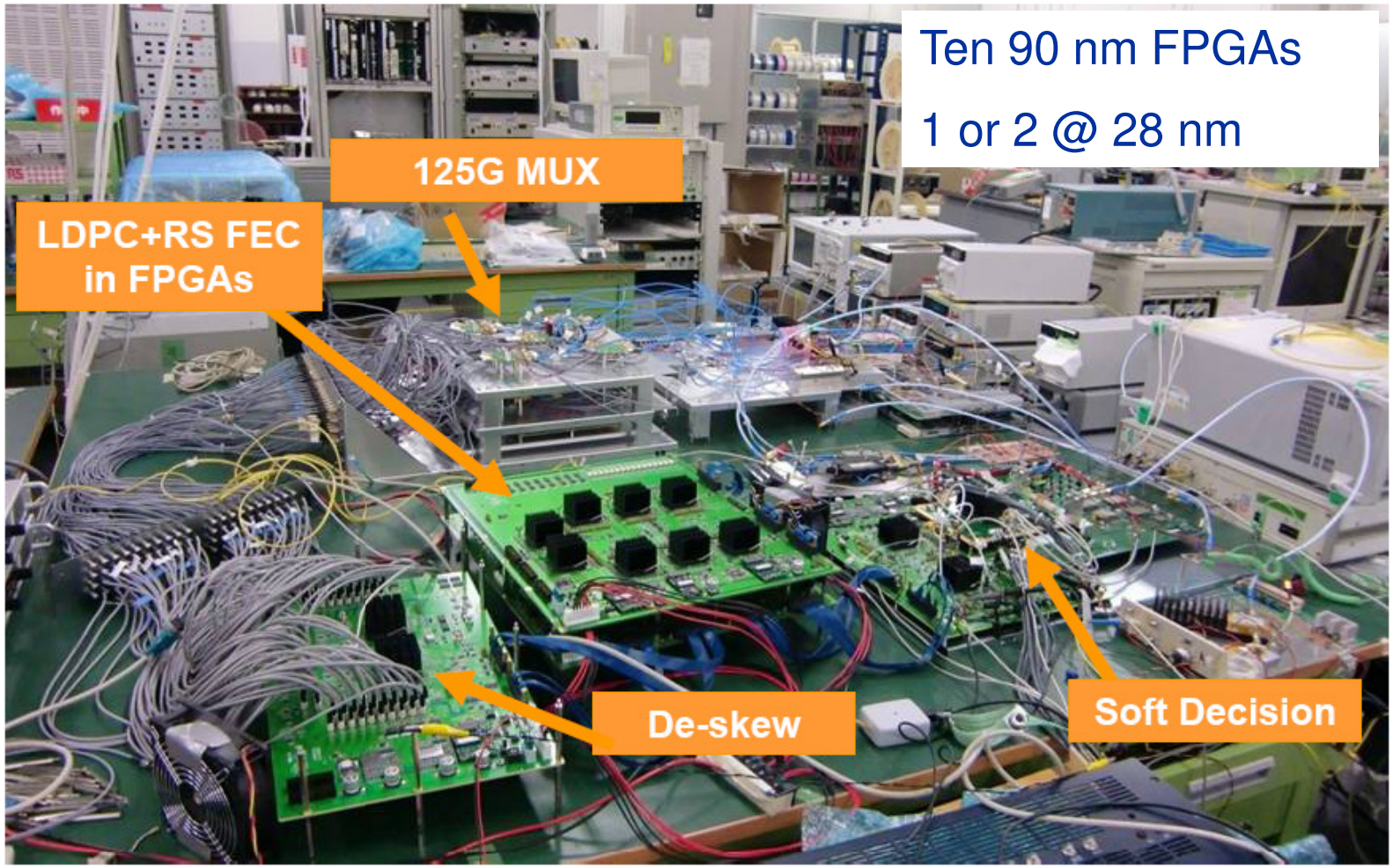
Stratix V Maximum Capacities

Feature	Stratix V
Logic Elements	1.1 M
RAM bits	52 Mb + 7.3 Mb
18x18 multipliers	3680
High-speed serial links	GX: 66 full-duplex @ 12.5 Gb/s GT: 4 @ 28 Gb/s + 32 @ 12.5 Gb/s
Hard PCIe blocks	4
Hard 40G / 100G PCS	Yes
Memory interfaces	7 x 72-bit DDR3 DIMM @ 800 MHz
On-chip memory bandwidth	~20,000 GB/s
I/O Bandwidth	~300 GB/s
18x18 MACs	1,840 GMAC/s

Altera's Device Roadmap



100G Optical System (Stratix II GX)



T. Mizuochi, et al, “Experimental demonstration of concatenated LDPC and RS codes by FPGAs emulation,” IEEE Photon Technol. Lett., 2009

Other Challenges & Enhancements @ 28 nm

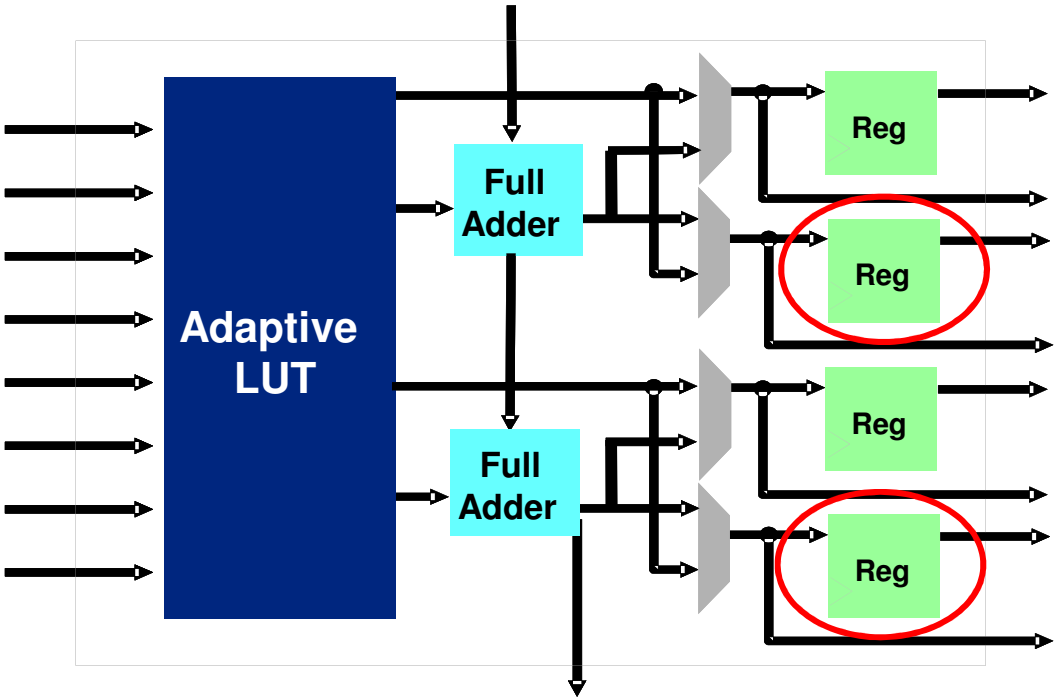
Controlling Power

Stratix V FPGA Power Reduction (New techniques highlighted in yellow)	Lower Static Power	Lower Dynamic Power
28-nm process (high-k, more strain, small C)	✓	✓
Programmable Power Technology	✓	
Lower core voltage (0.85 V)	✓	✓
Extensive hardening of IP, Embedded HardCopy Blocks	✓	✓
Hard power-down of more functional blocks	✓	
More granular clock gating		✓
Selective use of high-speed transistors	✓	
Dynamic on-chip termination	✓	✓
Quartus II software PowerPlay power optimization	✓	✓

Fabric Performance

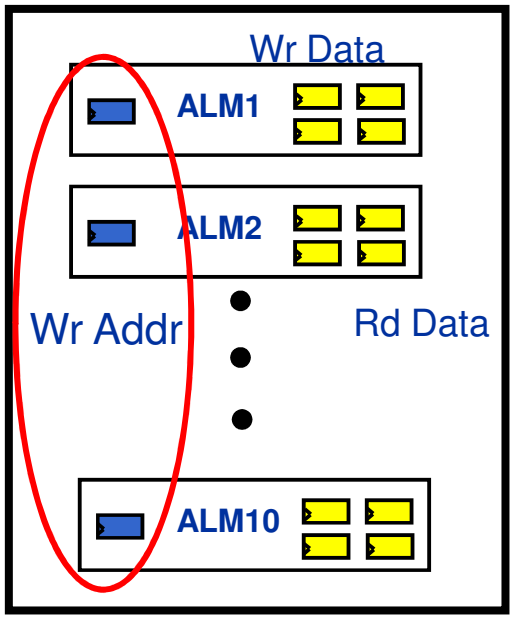
- Low operating voltage key to reasonable power
  - But costs speed
  - Logic still speeding up, routing more challenging
  - → Optimize process for FPGA circuitry (e.g. pass gates)
  - → Trend to bigger blocks / more hard IP
- Wire resistance rapidly increasing
  - → Co-optimize metal stack & FPGA routing architecture
  - Greater mix of wire types and metal layers (H3, H6, H20, V4, V12)
- Delay to cross chip not scaling
  - Above ~300 MHz, designers pipelining interconnect

Fabric: More Registers



- Double the logic registers (4 per ALM)
- Faster registers
- Aids deep pipelining & interconnect pipelining

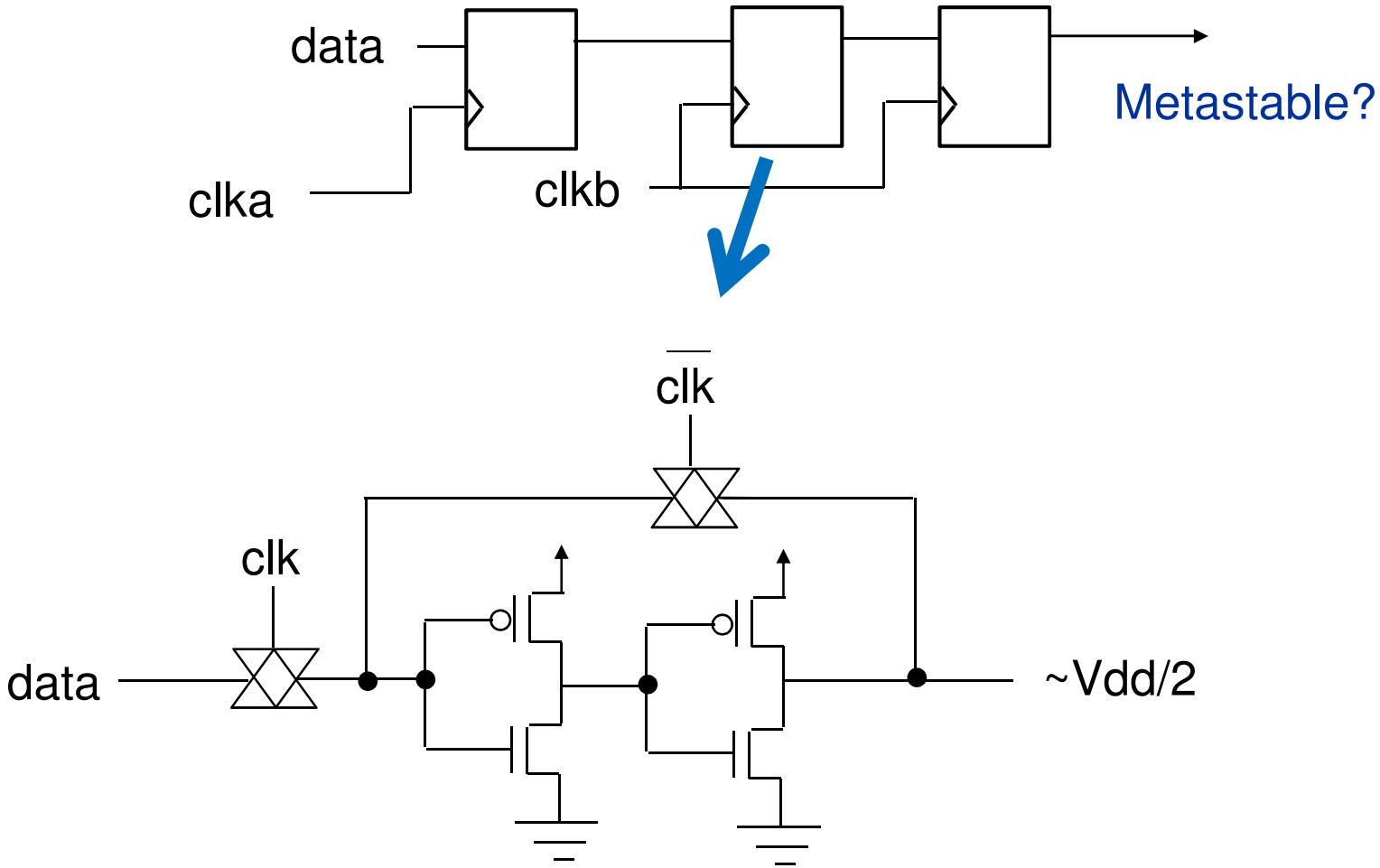
MLAB



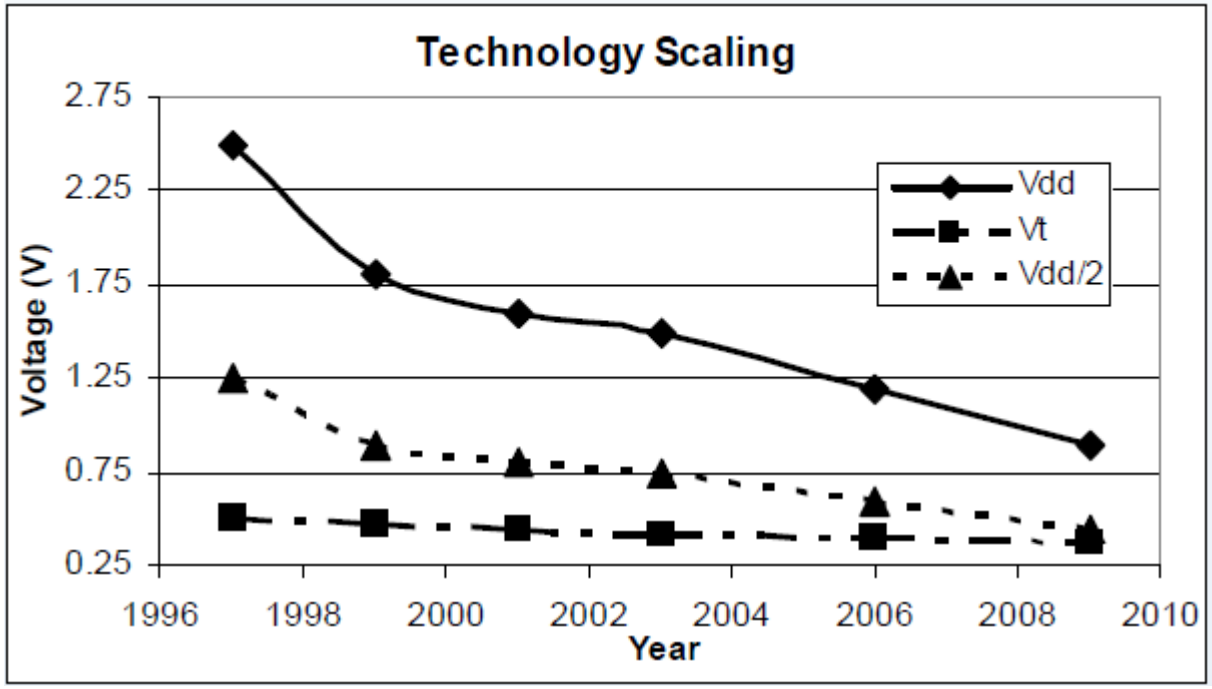
Memory mode: 5 registers

- Re-uses 4 ALM registers
- Adds extra register for write address
- Easier timing

Metastability Robustness



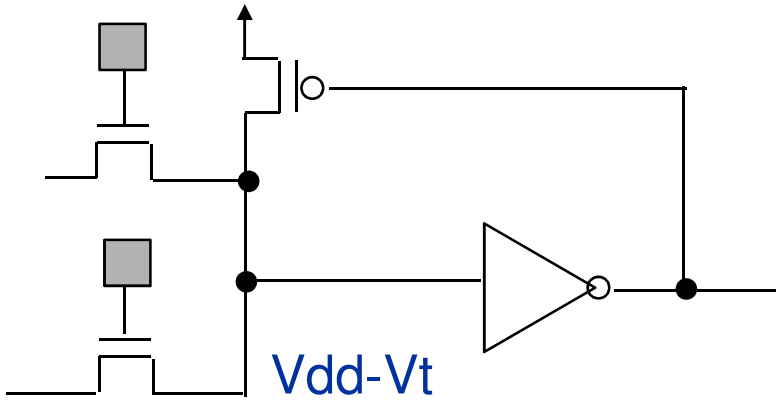
Metastability Robustness



Source: Chen, FPGA 2010

- Loop gain at Vdd/2 dropping → τ_{met} increasing
- Solution: register design (e.g. use lower Vt)
- Solution: CAD system analyzes & optimizes
  - 20,000 to 200,000 increase in MTBF

Pass Transistors



- Most area-efficient routing mux
- But Vdd – Vt dropping

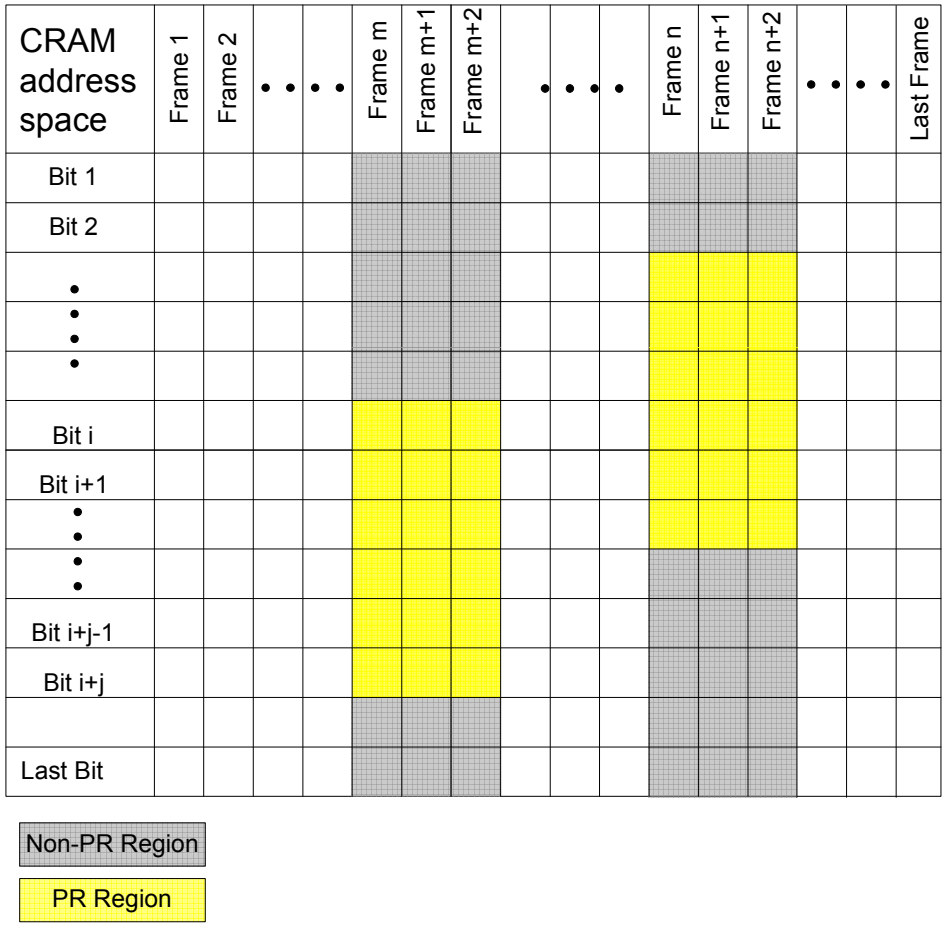
- Bias Temperature Instability (BTI) makes worse
  - Increase / hysteresis in Vt due to Vgs state over time
  - All circuits affected, but pass transistors more sensitive to Vt shift
- Careful process and circuit design needed
- Future scaling:
  - Full CMOS?
  - Opening for a new programmable switch?

Soft Errors

- Block RAM: new M20K block has hard ECC
  - MLAB: can implement ECC in soft logic
- Configuration RAM: background ECC
  - But could take up to 33 ms to detect
- Config. RAM circuit design to minimize SEU
- Trends with SRAM scaling:
  - Smaller target → lower FIT rate / Mb (constant per die)
  - Less charge → higher FIT for alpha, stable for neutron
  - Will this stabilize at an acceptable rate?
  - Known techniques to greatly reduce (at area cost) → does not threaten scaling

Stratix V Partial Reconfiguration

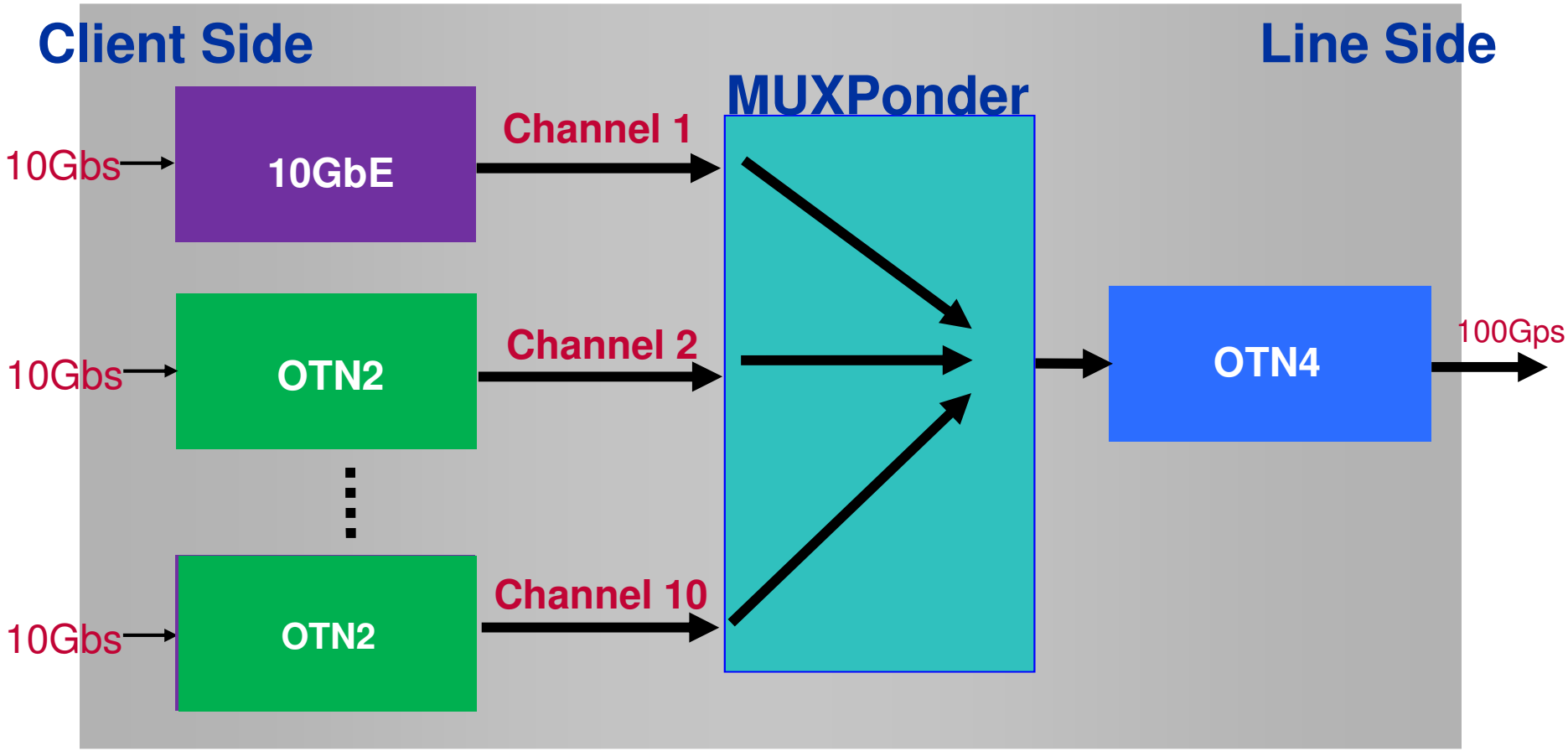
- Very flexible HW
- Reconfigure individual LABs, block RAMs, *routing muxes*, …
- Without disrupting operation elsewhere



Partial Reconfiguration (PR) Overview

- Software flow is key
  - Build on existing incremental design & floorplanning tools
  - Enter *design intent*, automate low-level details
  - Simulation flow for operation, *including reconfiguration*
- Partial reconfiguration can be controlled by soft logic, or an external device
  - Load partial programming files while device operating
- Target: multi-modal applications

Example System: 10*10Gbps→OTN4 Muxponder



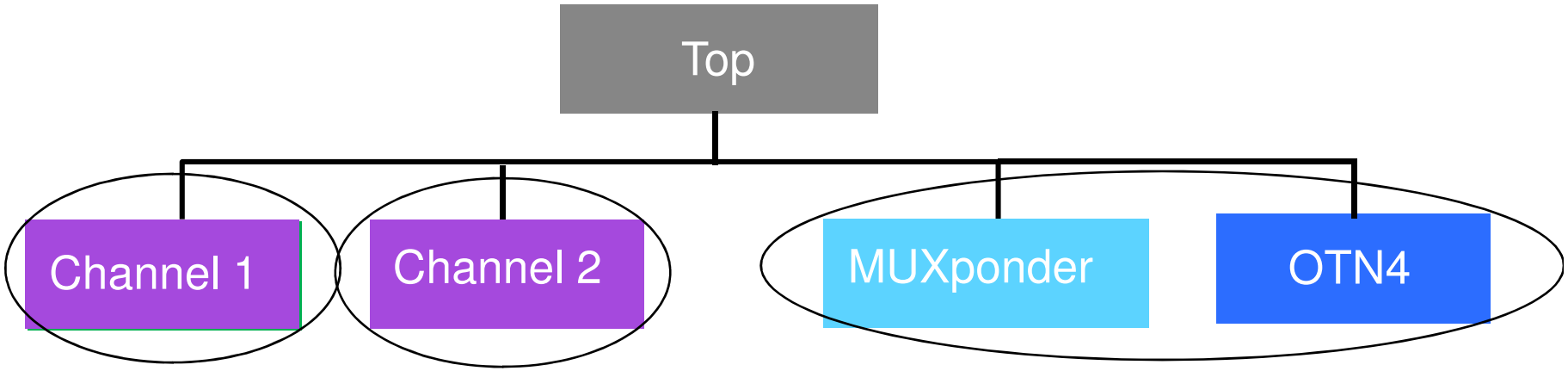
Design Entry & Simulation

- One set of HDL
- Tools to simulate during reconfig

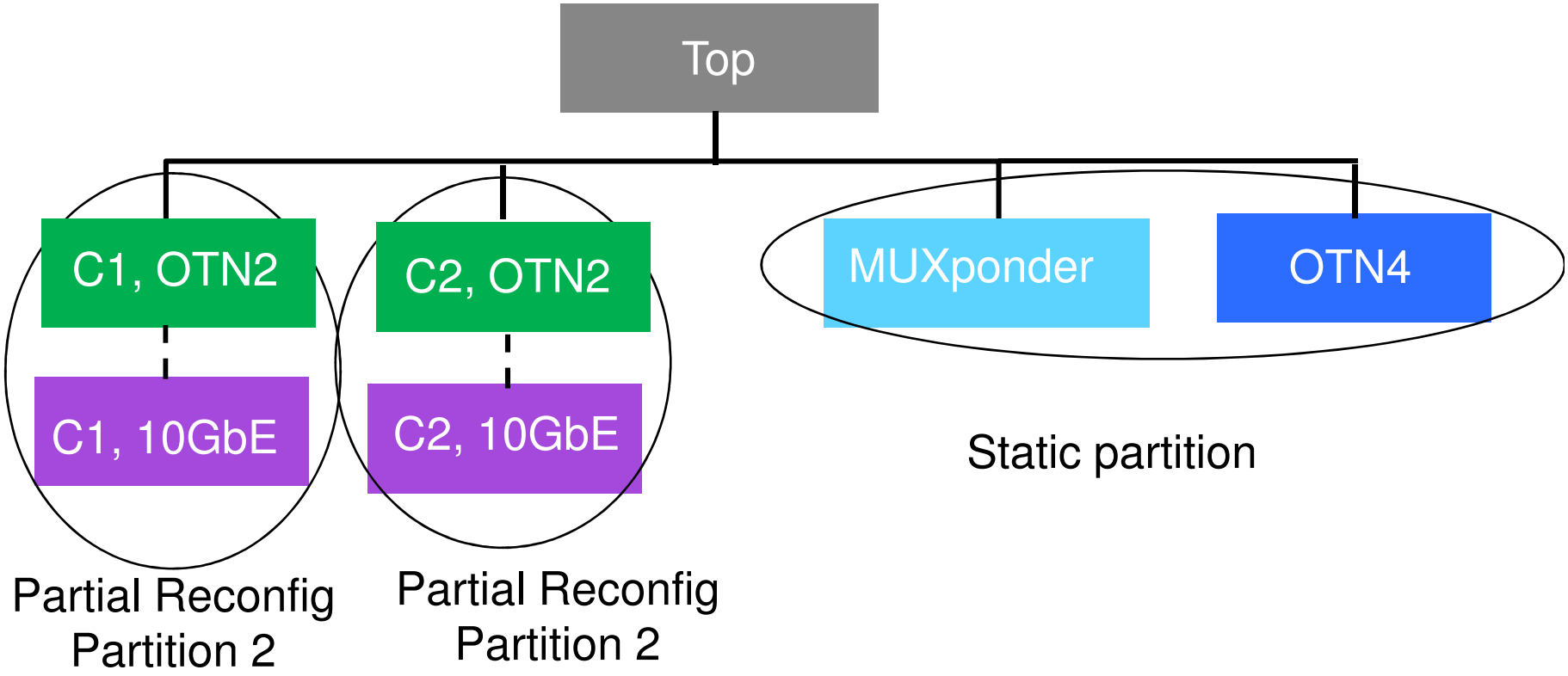
```
module reconfig_channel (clk, in, out);
input clk, in;
output [7:0] out;

parameter VER = 2;  // 1 to select 10GbE, 2 to select OTN2

generate
        case (VER)
        1: gige m_gige (.clk(clk), .in(in), .out(out));
        2: otn2 m_otn2 (.clk(clk), .in(in), .out(out));
        default: gige m_gige(.clk(clk), .in(in), .out(out));
        endcase
endgenerate

endmodule
```

Incremental Design Flow Background

- Specify *partitions* in your design hierarchy
- Can independently recompile any partition
  - CAD optimizations across partitions prevented
  - Can preserve synthesis, placement and routing of unchanged partitions

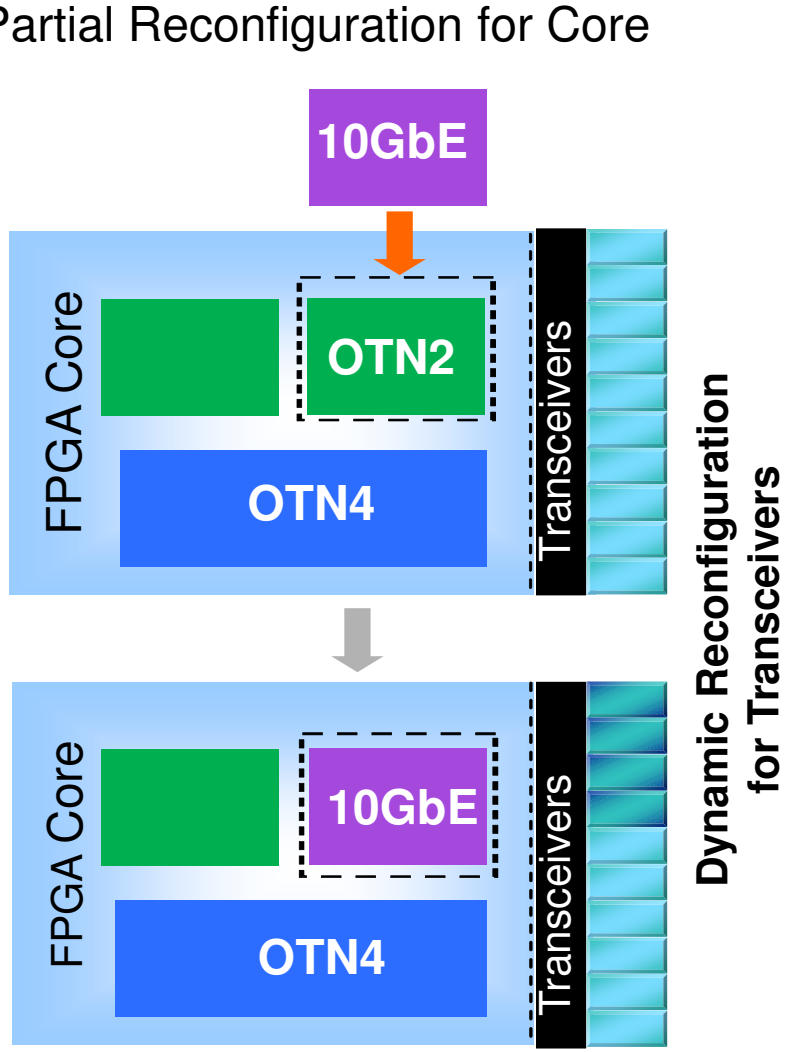


Partial Reconfig Instances



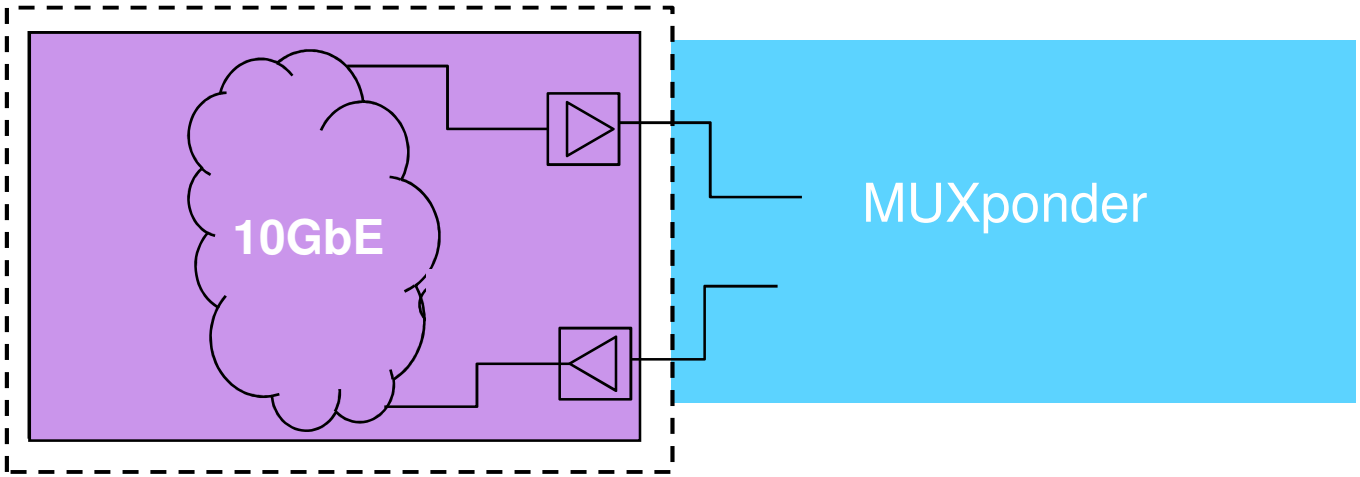
Partial Reconfiguration: Floorplanning

- Define partial reconfiguration regions
  - Non-rectangular OK
  - Any number OK
- Works in conjunction with transceiver dynamic reconfiguration for dynamic protocol support
  - "Double-buffered" partial reconfig



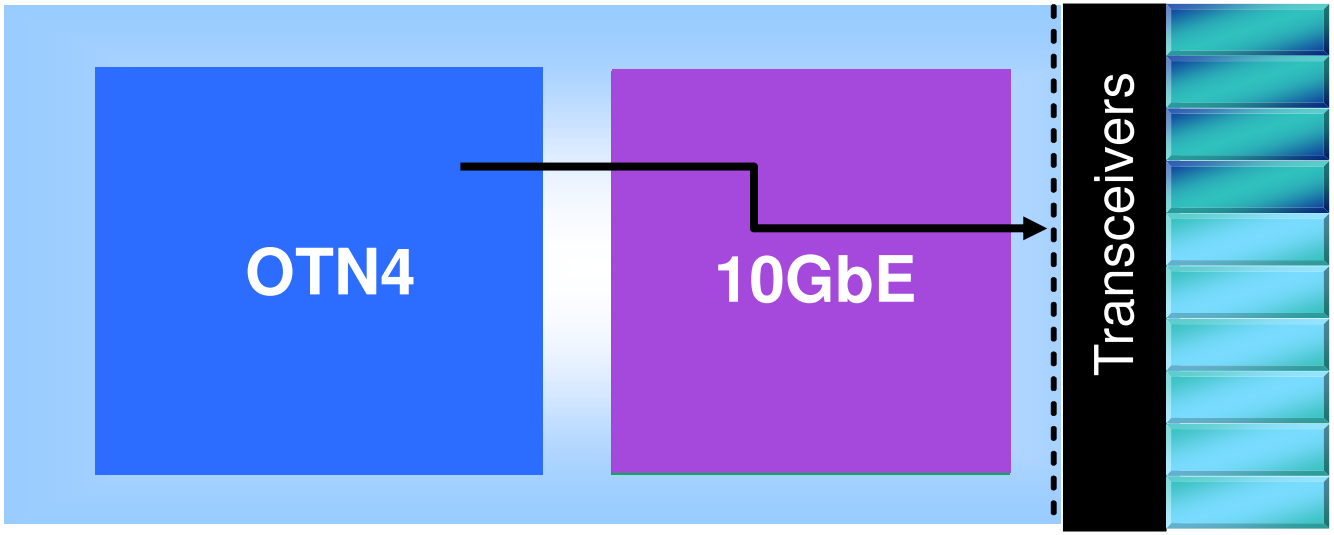
Physical: I/Os

- PR region I/Os must stay in same spot
  - So rest of design can communicate with any instance
- Same wire?
  - FPGAs not designed to route to/from specific wires
- Solution: *automatically* insert "wire LUT"
  - *Automatically* lock down in same spot for all instances



Physical: Route-Throughs

- Can partially reconfigure individual routing muxes
- Enables routing through partial reconfig regions
  - Simplifies / removes many floorplanning restrictions
  - Quartus II records routing reserved for top-level use
  - Prevents PR instances from using it



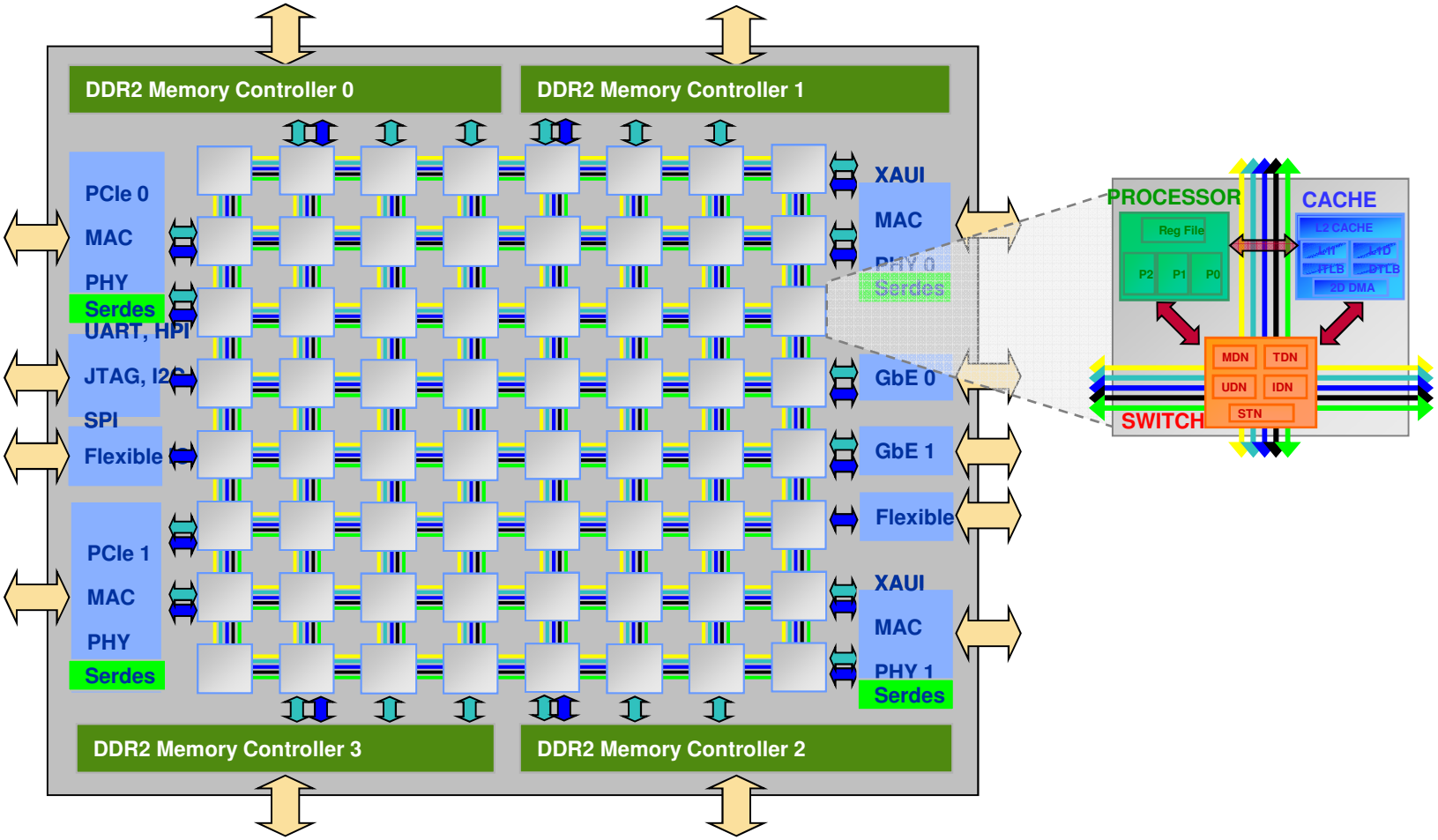
Extending & Improving the Software Stack

Design Flow Challenges

- HDL: low-level parallel programming language
  - RTL ~300 kLOCs, behavioural ~40 kLOCs [NEC, ASPDAC04]
- Timing closure
  - Fabric speed flattening, but processing needs growing
  - Datapaths widening, device sizes growing exponentially
  - 4x28 Gbps → 336 bit datapath @ 333 MHz → need good P & R
  - Need more latency? → may cause major HDL changes
- Compile, test, debug cycle slower than SW
  - And tools to observe HW state less mature
  - Any timing closure issues exacerbate
- Firmware development needs working HW

The Competition: Many Core

Tilera TILE64



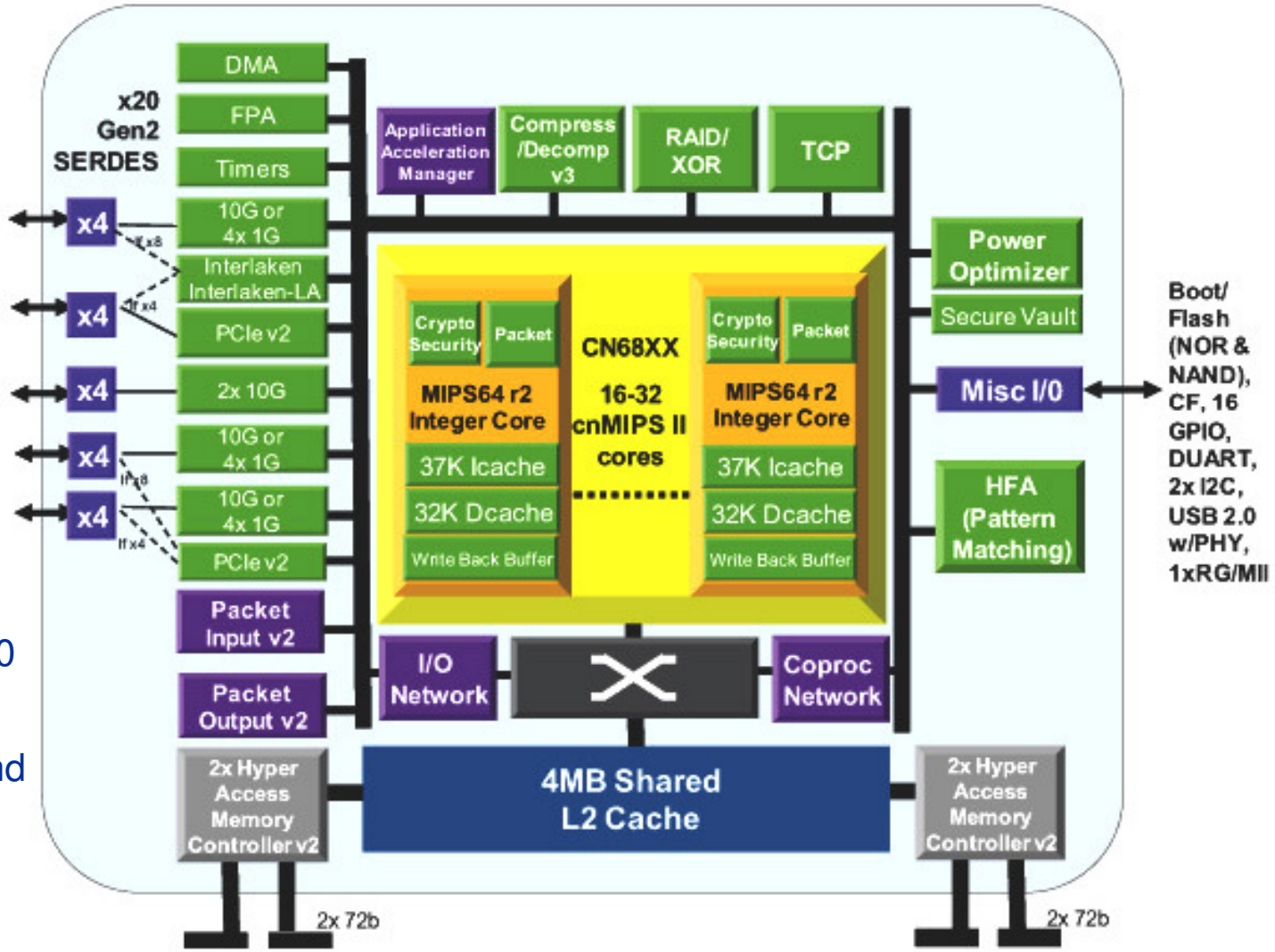
Competition: ASSP w/HW Accelerators

Ex. Cavium – Octeon CN68XX

85 application accelerators

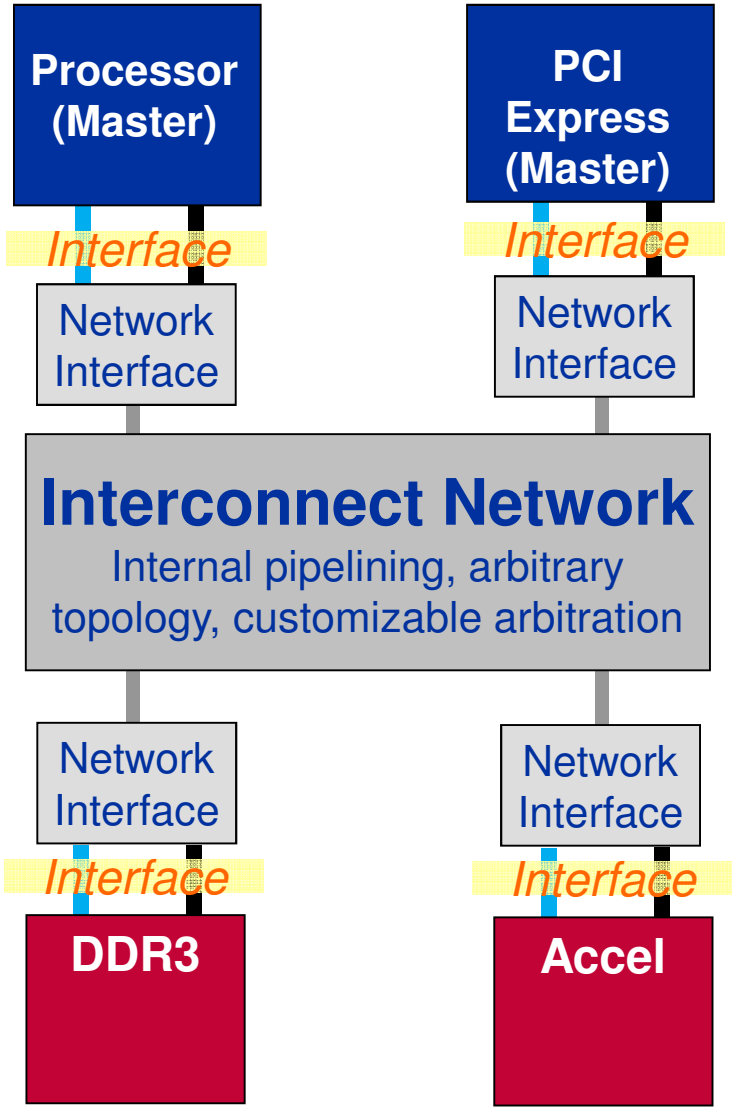
65 nm in Q4 2010

2 process generations behind FPGAs



“Bespoke” ASSPs in FPGAs

- Connect IP with SoPC Builder
  - Integrates system & builds software headers
- Next generation: general **Network-on-a-Chip**
  - Topology, latency: selectable
  - Scalable enough to form heart-of-the-system



High-Level Synthesis

- Good results in some problem domains (e.g. DSP kernels)
- Often difficult to scale to large programs
- Debugging and timing closure difficult
  - **Unclear how the code relates to the synthesized solution**
  - How to change the 'C' code to make hardware run faster?
  - Few tools to drive profiling data back to the high-level code
  - Few tools to debug HW in a software-centric environment

OpenCL: *Explicitly Parallel C*

- The OpenCL programming model allows us to:
  - **Define Kernels**
    - Data-parallel computational units → can hardware accelerate
    - Including communication mechanism to kernels
  - **Describe parallelism within & between kernels**
  - **Manage Entire Systems**
    - Framework for mix of HW-accelerated and software tasks
- Still C
- Multi-target

OpenCL Structure

```
__kernel void sum { … }

__kernel void transpose {…}

float cross_product { … }
```

Program: kernels and functions

Task-level parallelism, overall framework

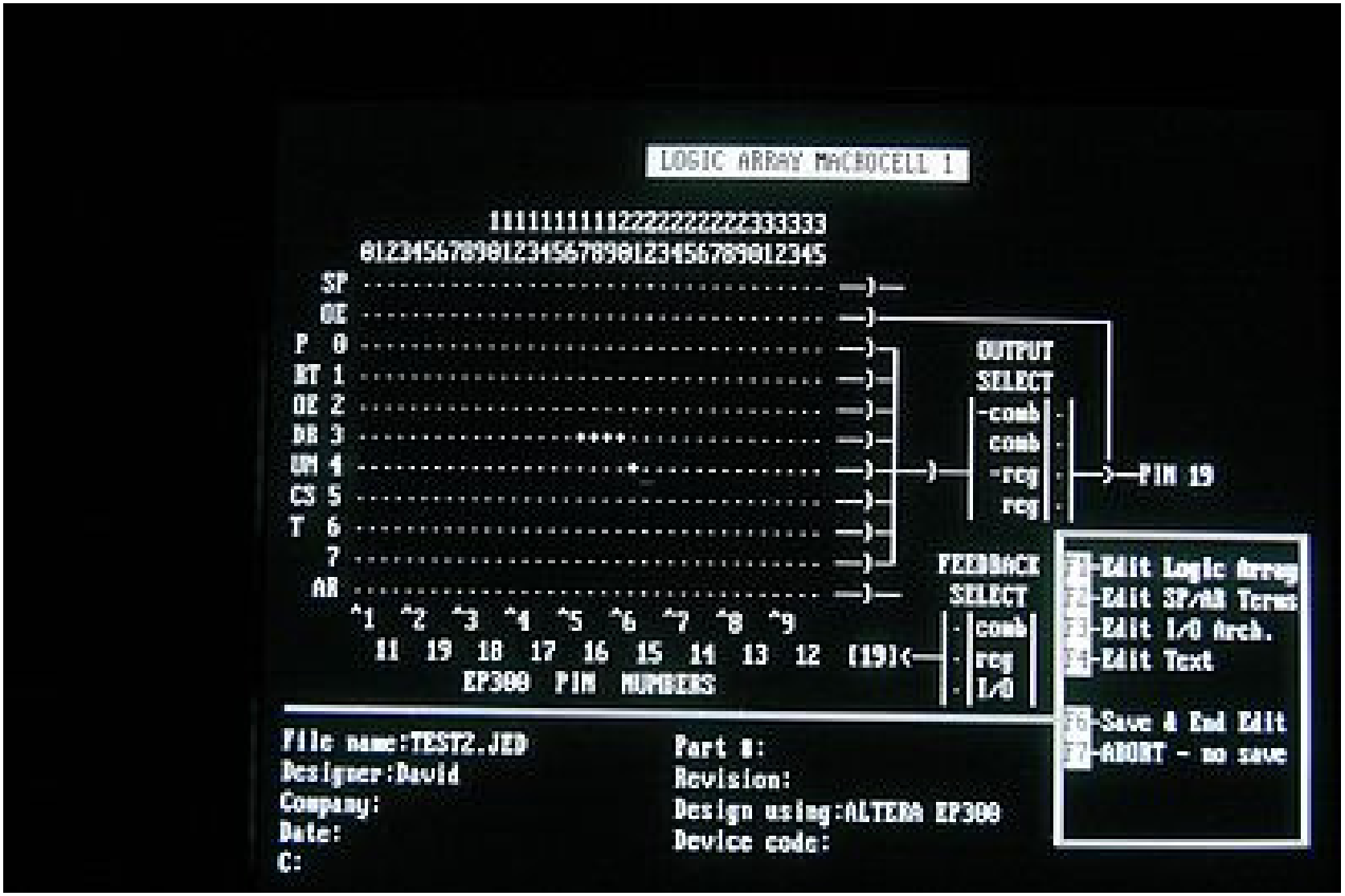
```
__kernel void sum
 (__global const float *a,
  __global const float *b,
  __global float *answer)
{
   int xid = get_global_id(0);
   answer[xid] = a[xid] + b[xid];
}
```

Kernels: data-level parallelism

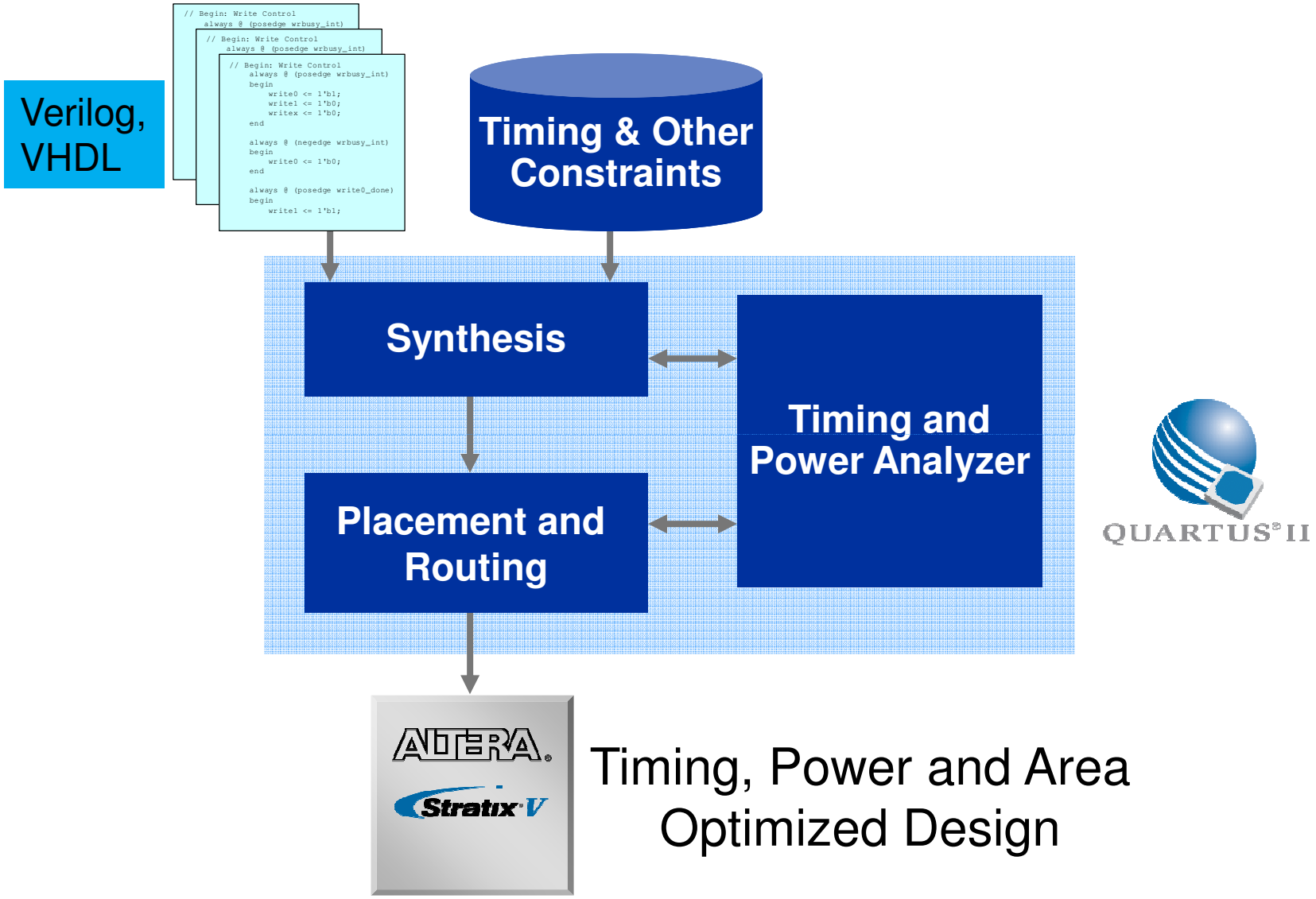
Suitable for HW or parallel SW implementation

Specify memory hierarchy

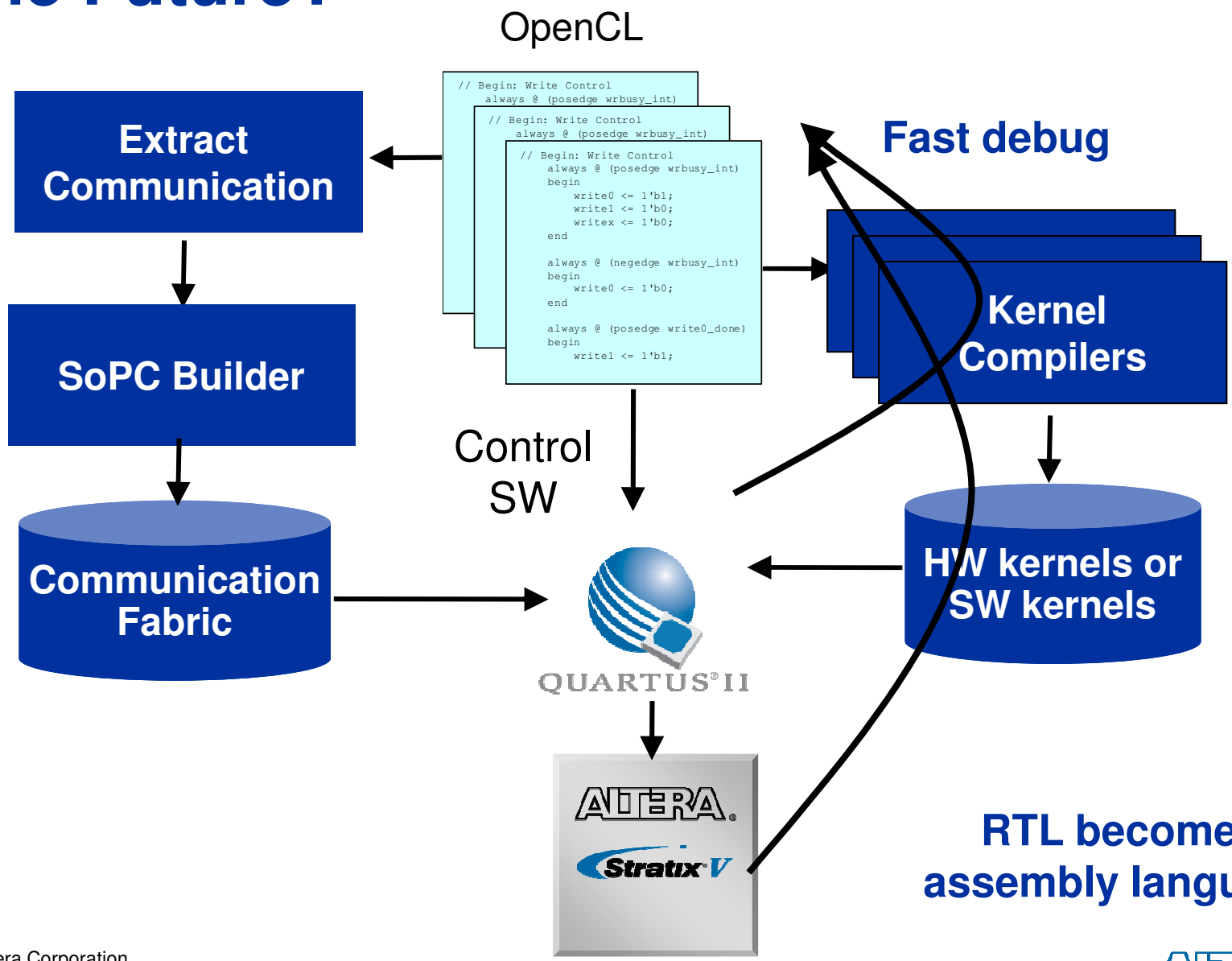
The Past (1984): Editing Switches



The Present: HDL Design Flow



The Future?



RTL becomes assembly language

Summary

Summary

- Huge demand for more processing
  - Possibly outstripping Moore's law & off-chip bandwidth
- FPGAs becoming SoCs
  - More heterogeous/hard function units
  - FPGAs specializing to markets
- 28 nm & Stratix V
  - -30 to -50% power, 1.5x I/O bandwidth, 1.5x – 2x more processing
  - Partial reconfiguration
- FPGA robustness with scaling
  - Innovation overcoming issues → scaling continues
- Tool innovation needed
  - Higher-level, fast debug cycles, push-button timing closure