

FPGA Based DSP System Design

Wonyong Sung

School of Electrical Engineering
Seoul National University

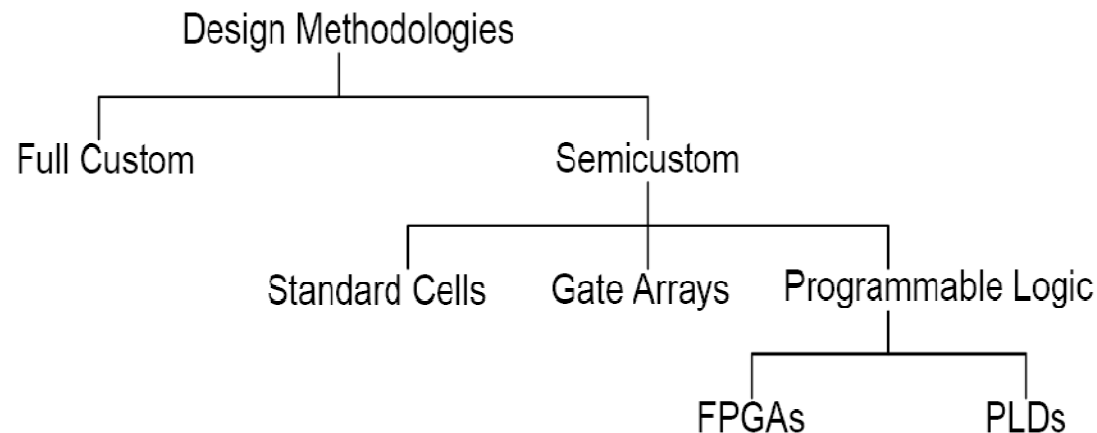
Contents

- 1. FPGA Overview**
- 2. Platform based FPGA**
- 3. Comparison of FPGA and programmable digital signal processors**
- 4. FPGA Top-down Design Flow for Communication System**

1. FPGA Overview

❖ FPGA (Field Programmable Gate Array)

- Field programmable: users can program the hardware/software at their workplace.
 - For programming, it uses CMOS switch, thin contact, or EPROM tr.
- Gate array: a kind of ASIC, contains a lot of gates (can be configured to NAND, NOR, FF) on a chip, while metal lines and contacts are built according to the customers' request
 - Requires a fairly small mask and process charge because a customer pay only the cost of a few masks (not 20) and the base chips are made in quantity.
 - The chip cost is still higher than that of the full or standard cell based custom IC's.



FPGA advantages

- ❖ **The NRE for fabrication and the delivery time is almost zero, although the chip price is much higher.**
 - Advantageous for prototyping and small quantity production.
 - Note that the NRE for modern CMOS process exceeds \$1M, and is increasing as the process technology becomes complex.

FPGA disadvantages

- ❖ **Chip price: easily goes more than \$100.**
 - But many are around \$20~.
- ❖ **Chip area (density): the area is about 30 times of the full custom design**
 - $\text{FPGA density} < \text{Gate array} < \text{Standard cell} < \text{Full custom}$
- ❖ **Speed or delay: the delay is a few times.**
- ❖ **Power consumption: still much higher**

FPGA technology

❖ Conventional FPGA

- Many configurable logic blocks (CLB in Xilinx or PLD type for Altera)
 - Each configurable logic block corresponds to 10~30 gates (Xilinx)
- Programmable interconnects

❖ Platform FPGA

- Logic blocks
- Programmable interconnects
- Microprocessors
- RAM blocks
- MAC (multiply-accumulator) for DSP

Programming (configuration) methods

❖ Static RAM technology based

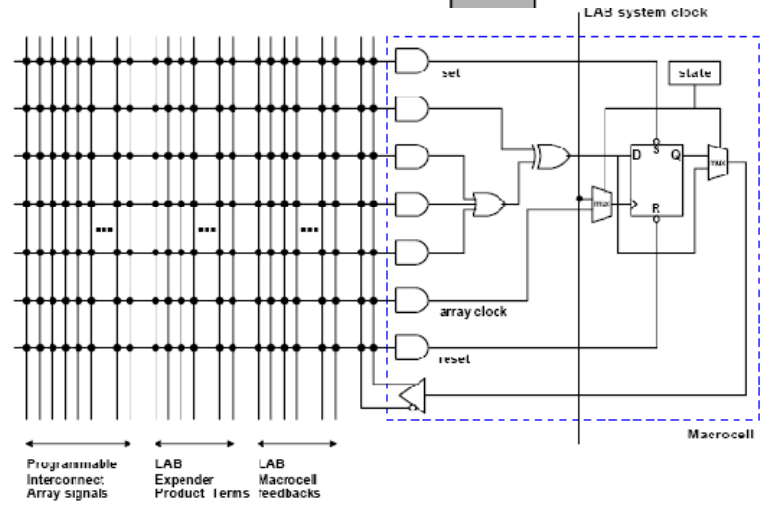
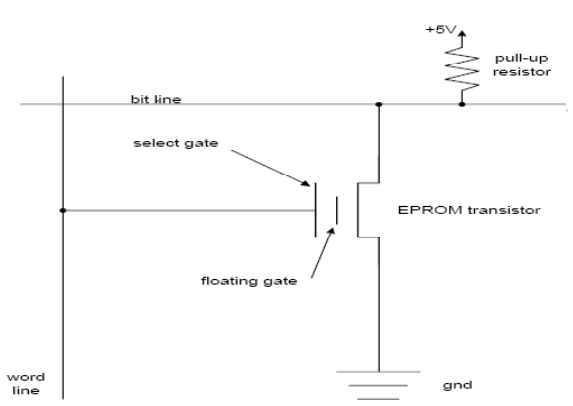
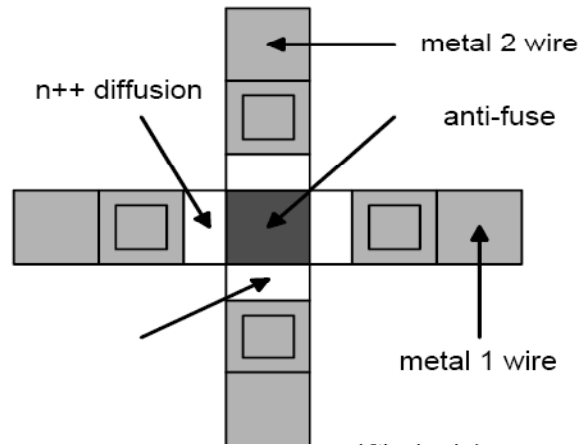
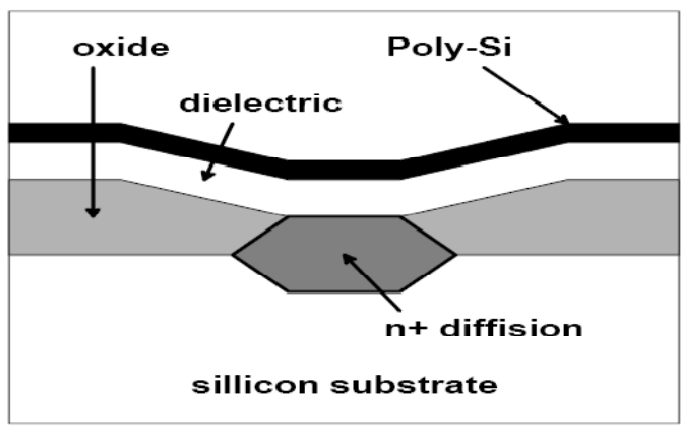
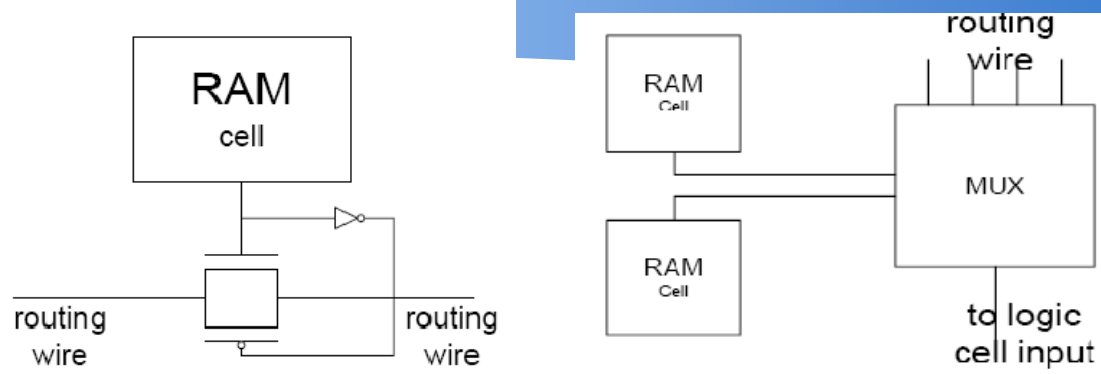
- Use CMOS mux and the switch information is stored at SRAM cells
- Volatile (so it needs external ROM) and reprogrammable
- Ordinary CMOS process compatible
- Xilinx

❖ Antifuse

- Uses antifuse (becoming conductive by programming)
- Non-volatile, very secure, but one-time no reprogrammable
- Special process needed (disadvantageous!).
- Actel

❖ EPROM (EEPROM)

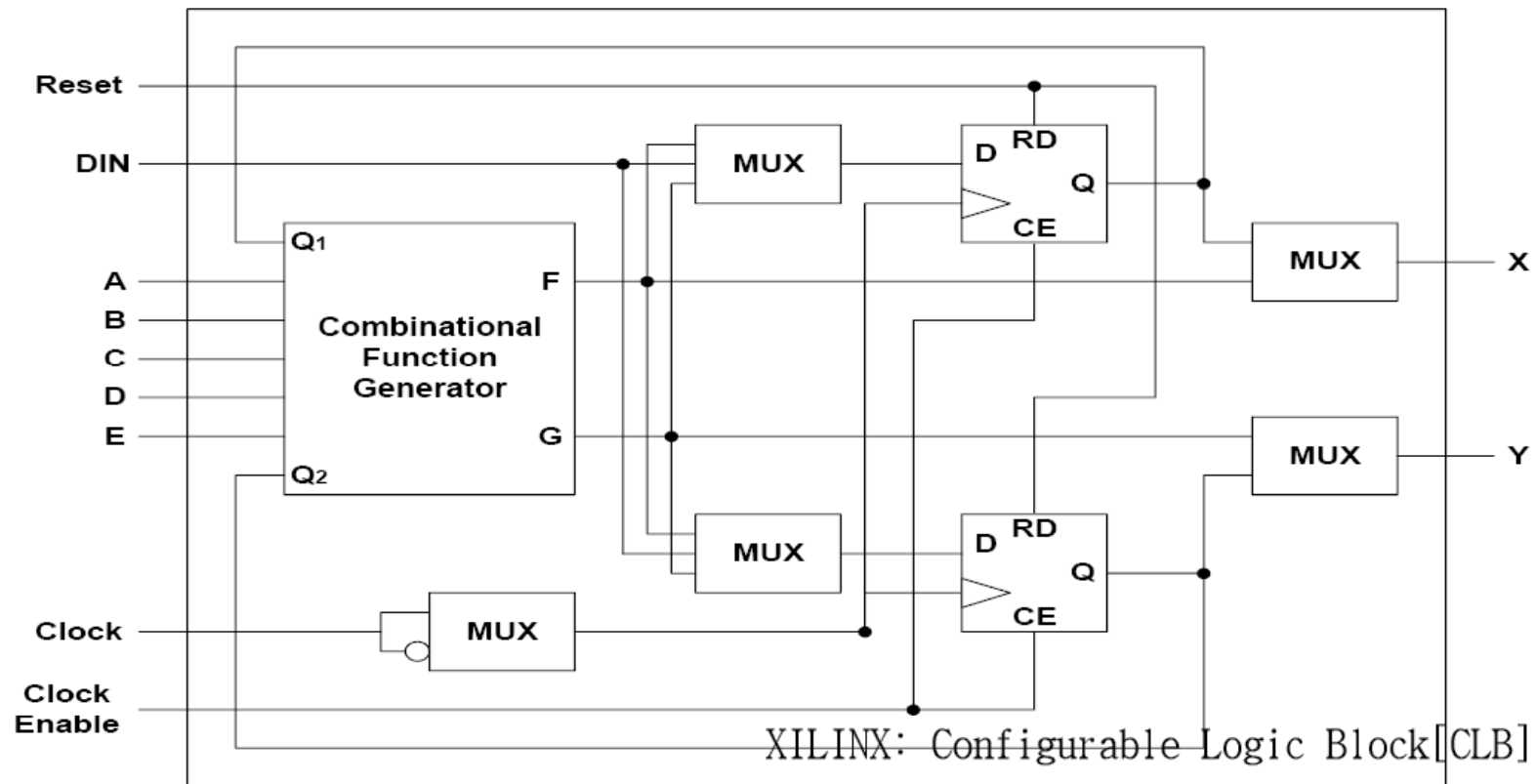
- Programming information stored at EPROM cell
- Non-volatile and reprogrammable
- Altera



| Technology | Volatile | Re-prog | Chip area | Contact R (ohm) | Contact C (ff) |
|------------------------|----------|--------------------------|---------------------------------|-----------------|----------------|
| Static RAM | yes | In-ckt, even at run time | large | 1k~2k | 10~20 |
| Actel antifuse (PLICE) | no | no | Small anti-fuse, large prog tr. | 300~500 | 3~5 |
| Vialink antifuse | no | no | Small anti-fuse, large prog tr. | 50~80 | 1.3 |
| EPROM | no | Out of ckt | small | 2~4K | 10~20 |
| EEPROM | no | In ckt | 2xEPROM | 2~4K | 10~20 |

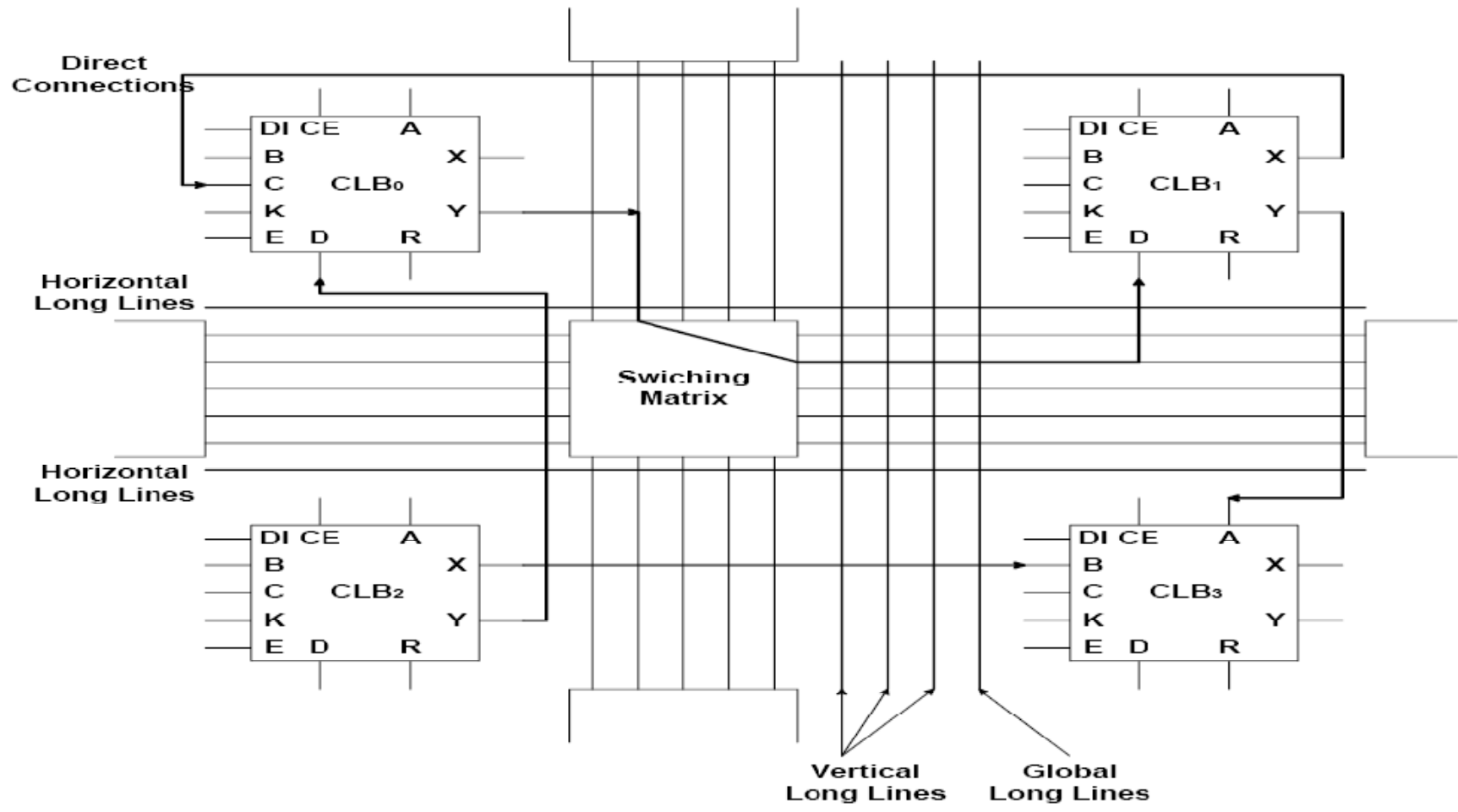
Xilinx CLB

- ❖ **Combinational function generator receives 5 (4) input signal and generates 2 latched output signal**

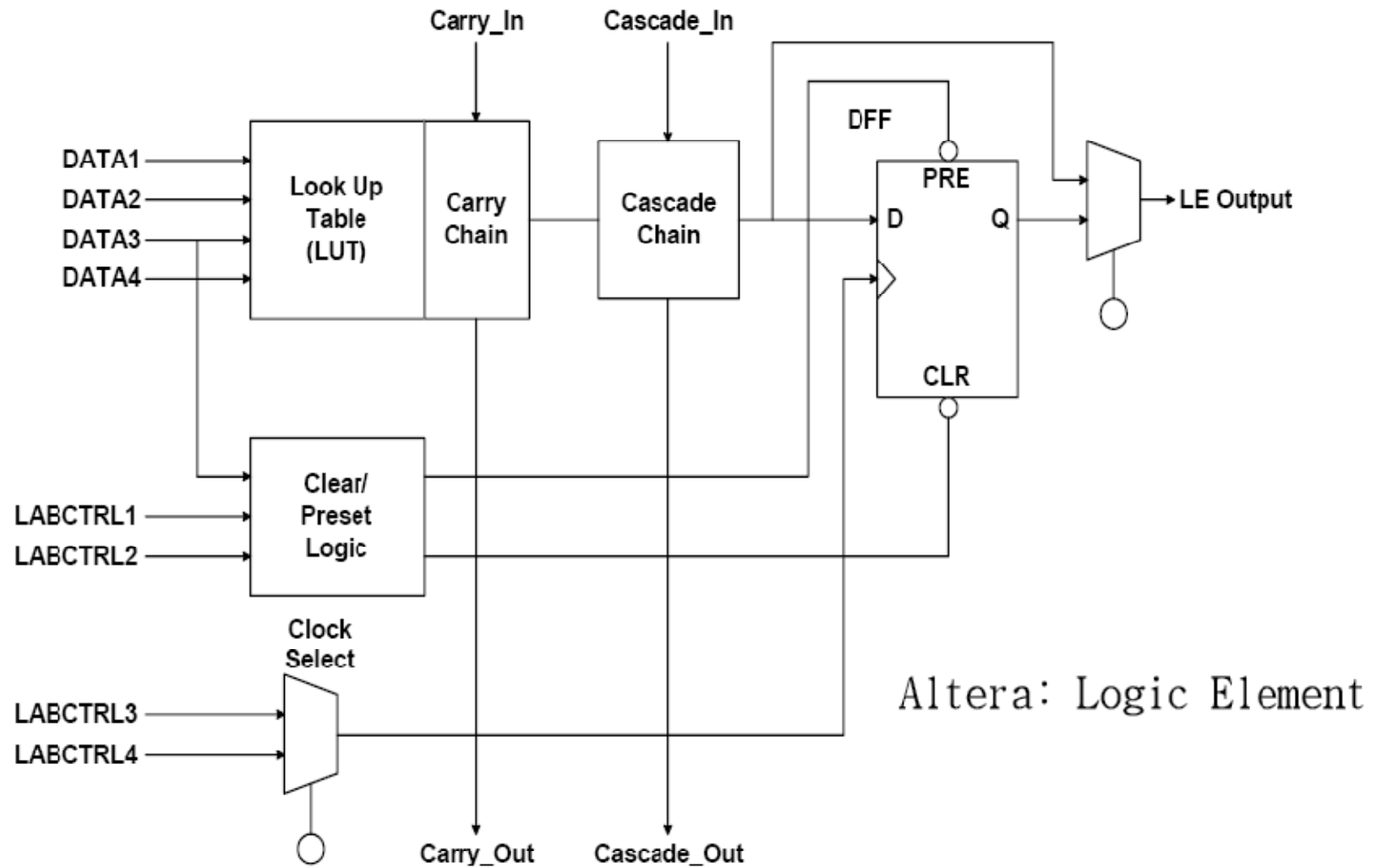


Xilinx interconnection

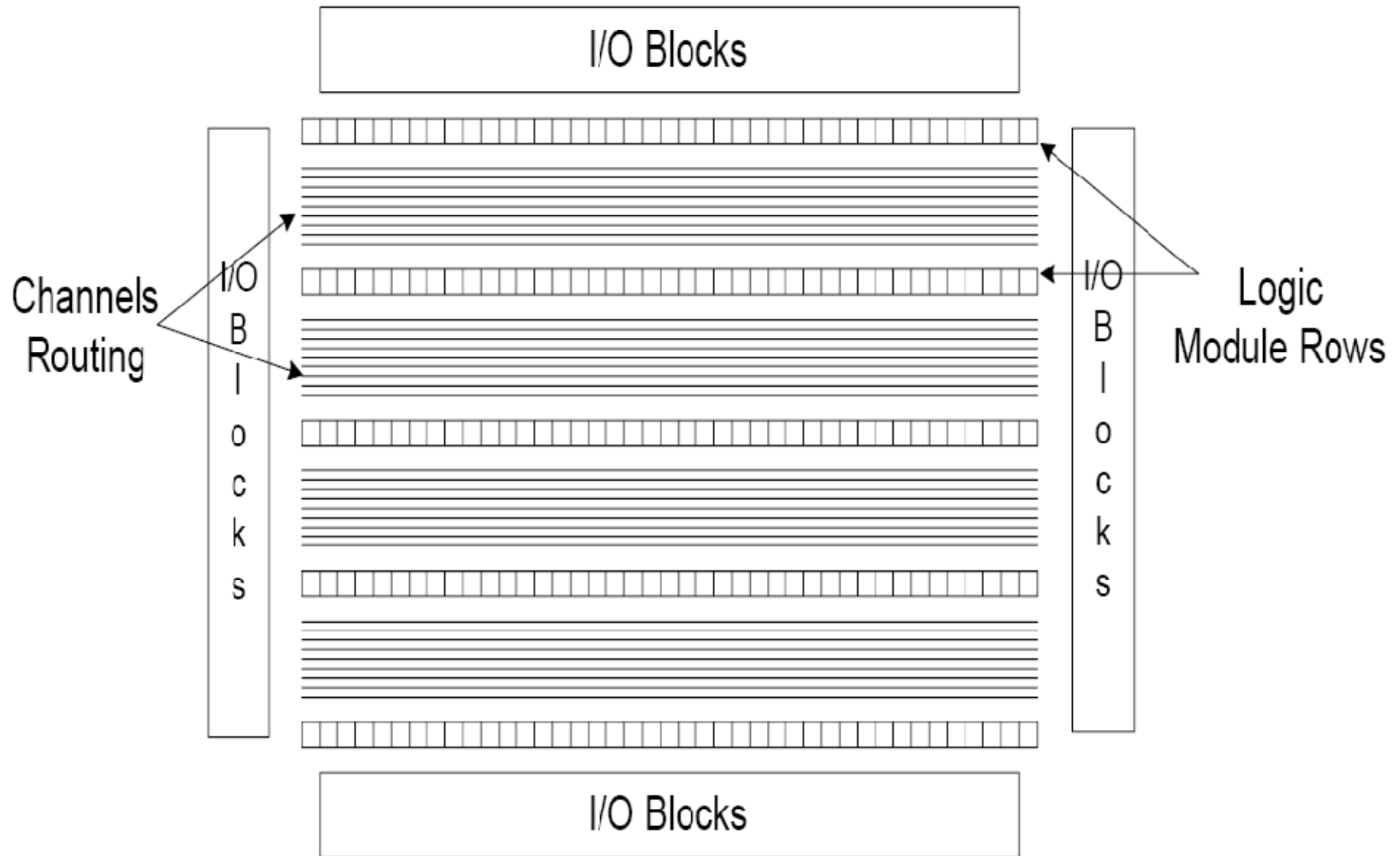
- ❖ **CLB placements and 2-D (horizontal and vertical) interconnections**
 - Direct interconnection between neighboring CLB's.
 - Horizontal and vertical long lines
 - Global lines
- ❖ **Switching matrix is used for interconnect control.**



Altera FPGA macrocell

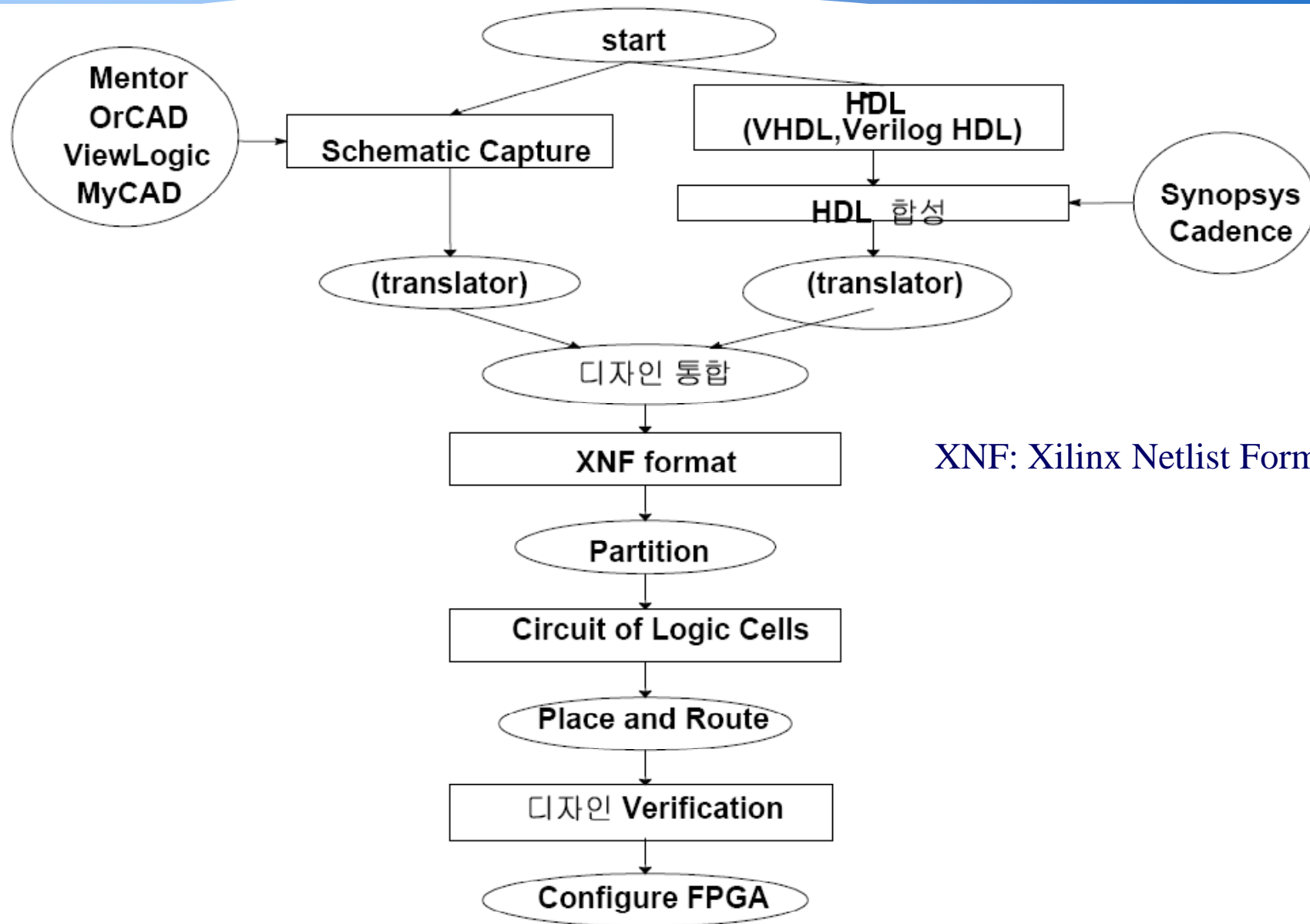


Altera FPGA interconnects



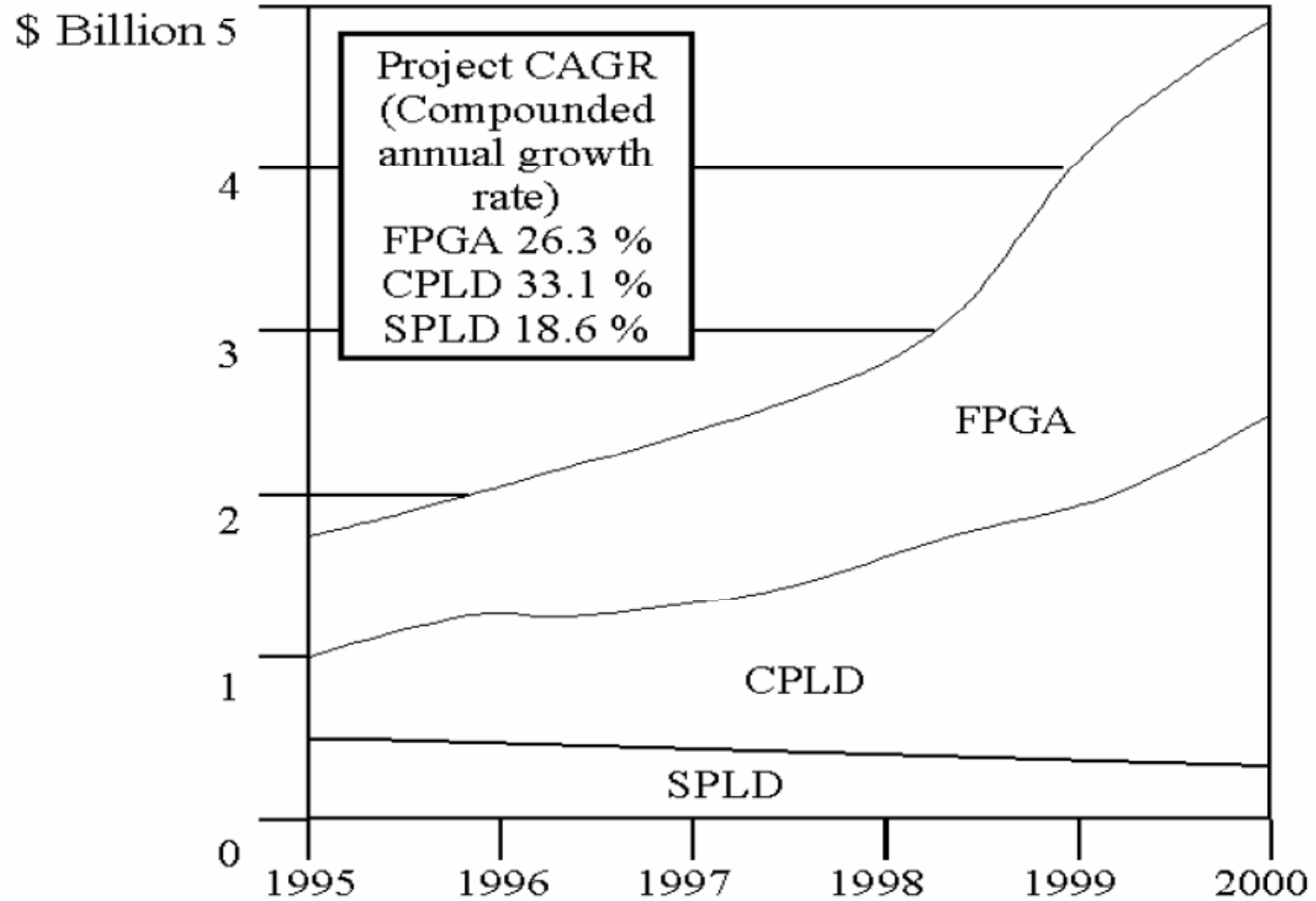
FPGA design flow

- ❖ **Start with HDL or schematic entry (same with conventional digital design)**
 - Synthesize and generates RTL design
 - Simulate
 - Synthesize after target to FPGA, and add I/O pads
 - Generates XNF file for transportation to Xilinx tools
 - XNF is the Xilinx Netlist Format
 - In ordinary ASIC design, EDIF is used.
- ❖ **Placement and route:**
 - Chip makers supply for their own tools
 - Mapping to FPGA resource
 - Placement (allocate logic blocks)
 - Routing (interconnection of logic blocks)
 - Generates “bitstream” file
 - Conducts simulation of SDF (Standard Delay Format) file that contains the information of cell delays, setup, hold times.

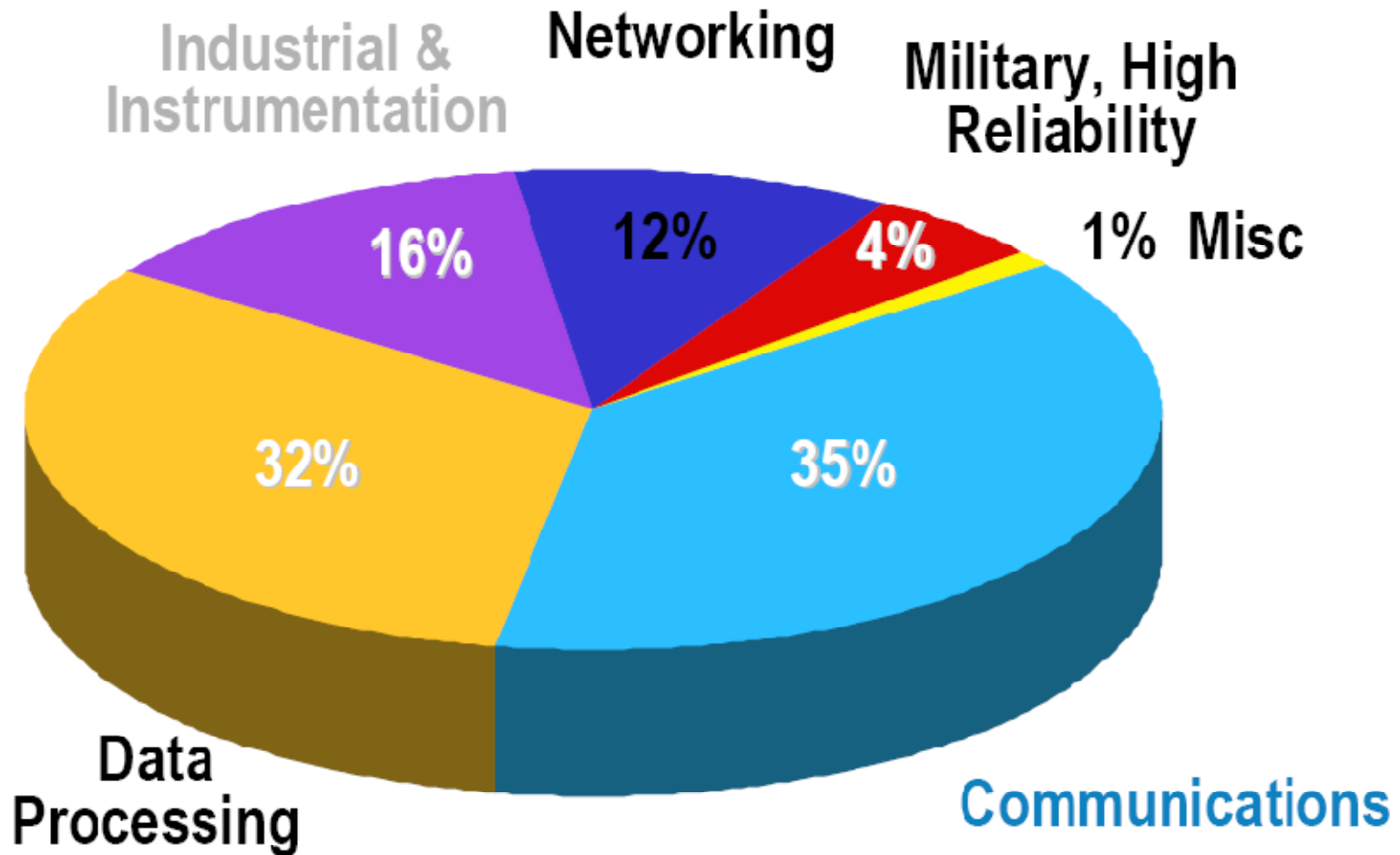


XNF: Xilinx Netlist Format

FPGA markets



FPGA market by applications



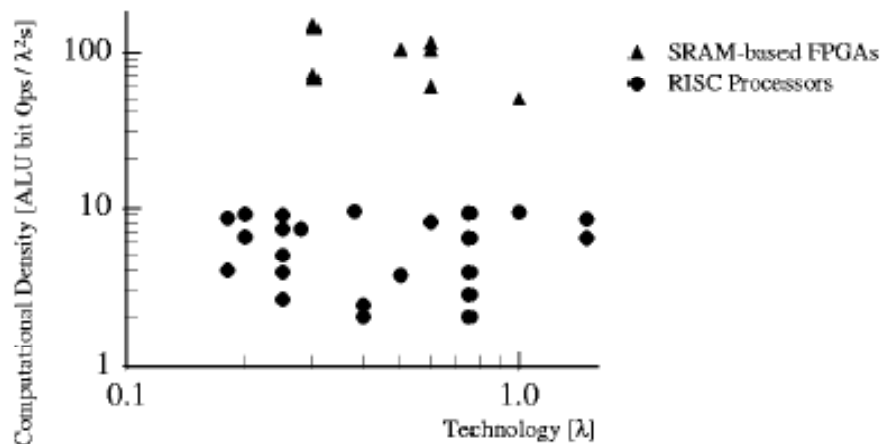
FPGA and ASIC

- ❖ Area ratio: 17:1
- ❖ Critical path delay ratio: 3 ~ 4:1
- ❖ Dynamic power ratio: 7.1 ~ 1

FPGA and CPU

- ❖ **FPGA exploits spatial parallel computation**
 - X10 advantages in the computational density
 - Exploits locality and parallelism in the applications
 - Drawbacks: each resource is dedicated to single function, which may mean inefficient use of resources for some implementations


Density Comparison



FPGA inefficiency

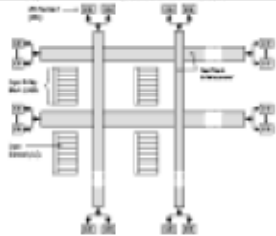
❖ Interconnects dominate

- Area
- Delay
- Power

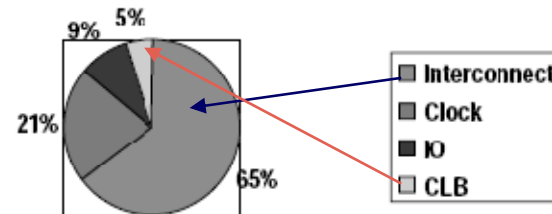
$$A_{bit_dim} = A_{fixed} + \frac{N_{SIP}(N_p, w_p)}{interconnect} + \frac{\binom{c}{m} \cdot n_{bits} \cdot A_{mem_cell}}{instruction\ memory} + \frac{d \cdot A_{mem_cell}}{retiming\ memory}$$


| Function | Area (λ^2) |
|--------------------|-------------------------------|
| LUT MUX + ff | 20K (generous, closer to 10K) |
| Programming Memory | 80K (240K typical unencoded) |
| Interconnect | 700K (for $N_p = 2048$) |

| Design | Path | Total Delay | LUT Delay | Inter. % |
|------------------|-----------------------------------|-------------|-----------|----------|
| Altera 10K130V-2 | LUT-local-LUT | 2.5 ns | 2.1 ns | 16% |
| | LUT-row-local-LUT | 6.6 ns | 2.1 ns | 68% |
| | LUT-column-local-LUT | 11.1 ns | 2.1 ns | 81% |
| | LUT-row-column-local-LUT | 15.6 ns | 2.1 ns | 87% |
| | LUT-row-fanout-local-LUT (fanout) | 28 ns | 2.1 ns | 90% |



Dominant in Power



XC4003A data from Eric Kusse (UCB MS 1997)

Nevertheless why FPGA is the future choice?

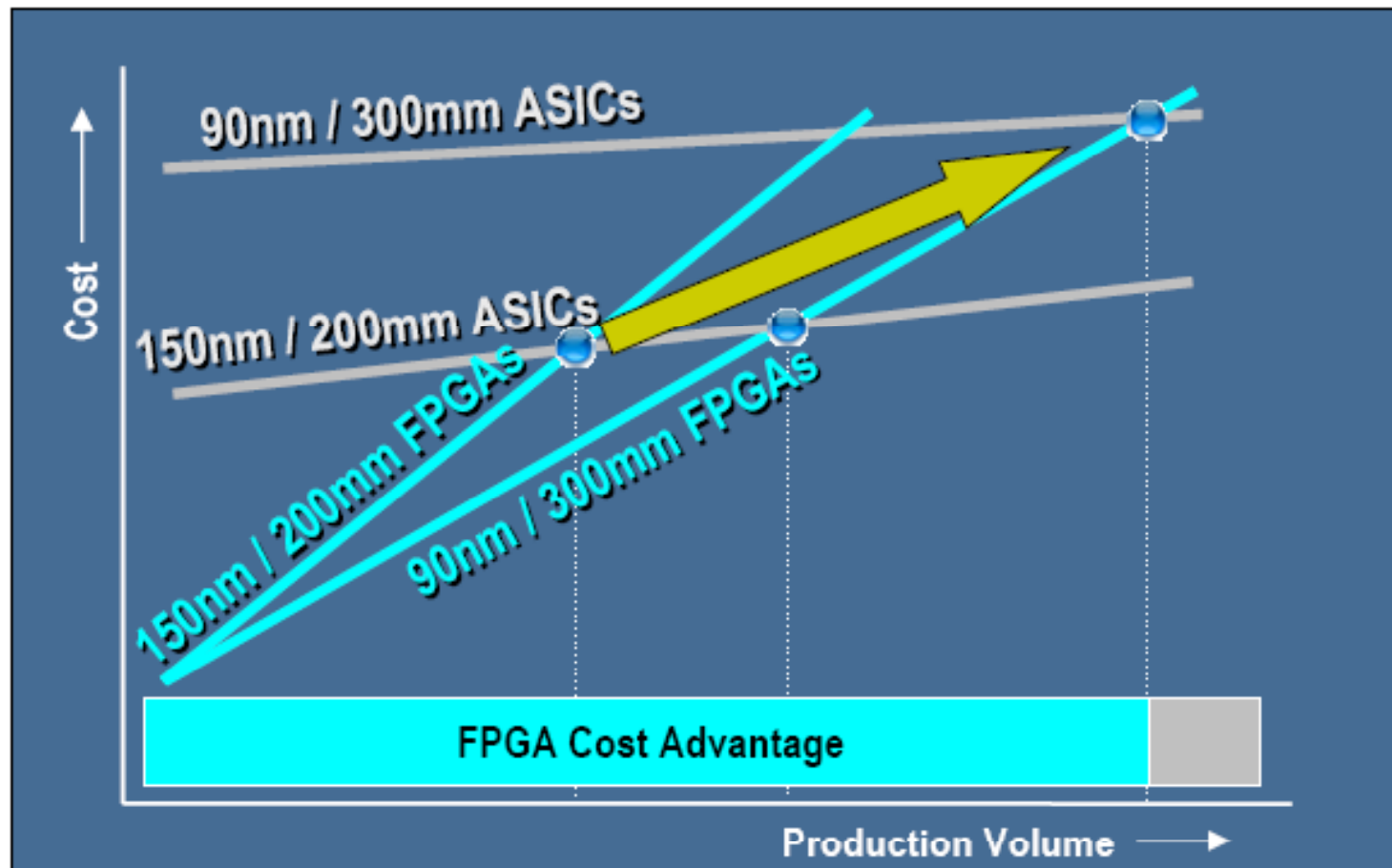
❖ NRE

The ASIC/ASSP Platform

| Cost increases over previous generation: | | Building a next generation SoC ASIC (90nm-) ? | |
|--|------|---|----------|
| Design | 60% | Development Cost | \$30-50M |
| Manufacturing | 40% | Total Cost of bringing product to market | \$80M |
| Mask Costs | 100% | Revenues needed to breakeven in 2 yrs | \$150M |

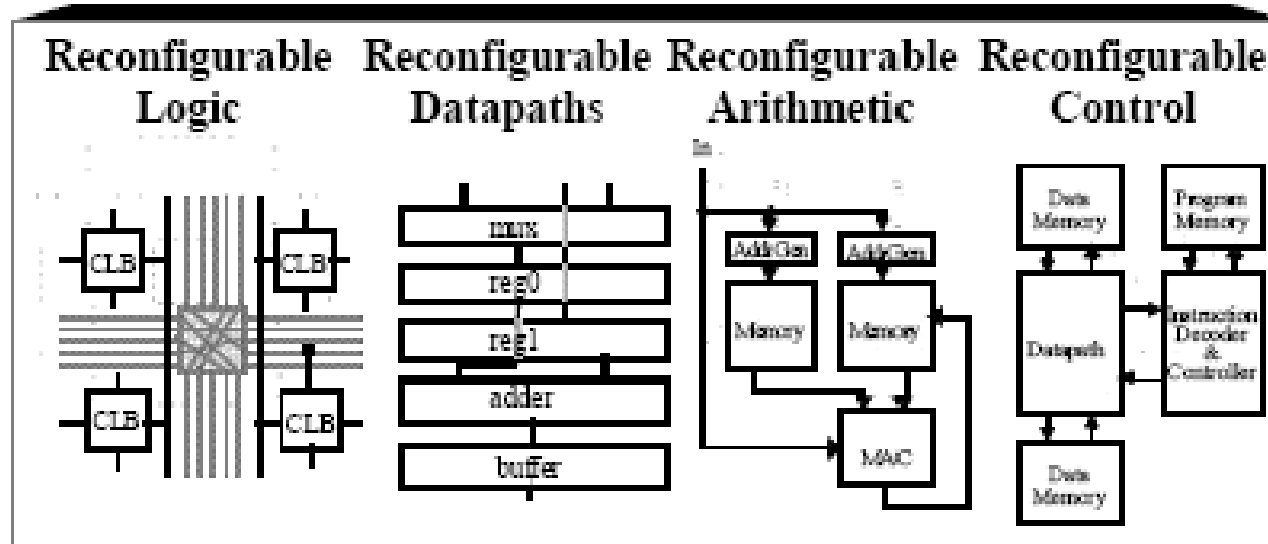
- 2007 ASIC/ASSP forecast = ~\$80B
- ~500 **successful** ASIC/ASSP design starts
 - NRE \$30M, R&D 20% of revenue, \$150M revenue

FPGA- ASIC/ASSP Crossover



Choice of reconfiguration

- ❖ Fine grain block is more general, however is less efficient and requires more interconnection



Bit-Level Operations
e.g. encoding

Dedicated data paths
e.g. Filters, AGU

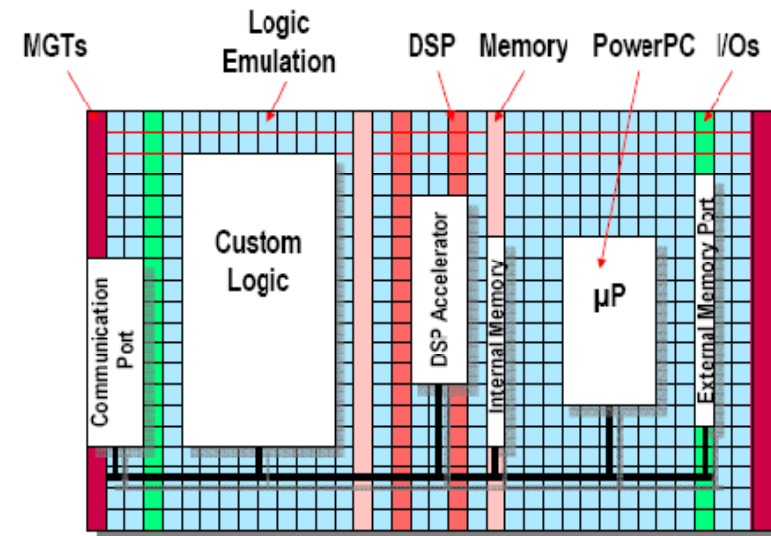
Arithmetic kernels
e.g. Convolution

RTOS
Process management

JR.800 21

2. Platform FPGA

- ❖ Mix of coarse and fine grain blocks
- ❖ The choice of coarse grain blocks is application dependent but much more efficient (in area, speed)
 - Hard-core CPU (Power PC) for data-processing and embedded applications
 - MAC (multiply-accumulate) units for DSP application
 - Fast interconnects
 - Large amount of block RAMs

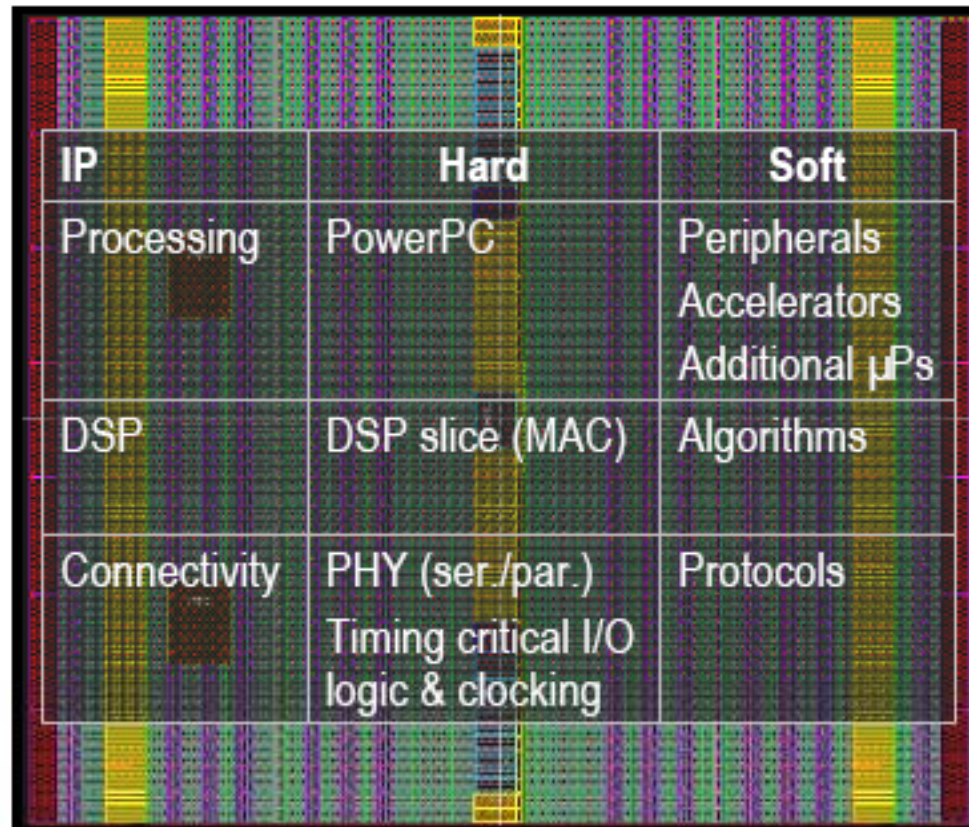


Programmable hard IP

- Up to 10x less area
- Up to 10x lower power
- Up to 2x performance

Customizable soft IP

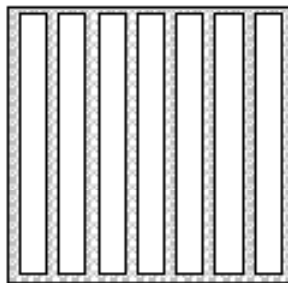
- Most flexible
- Widest selection



| IP | Hard | Soft |
|--------------|--|--|
| Processing | PowerPC | Peripherals Accelerators Additional μ Ps |
| DSP | DSP slice (MAC) | Algorithms |
| Connectivity | PHY (ser./par.) Timing critical I/O logic & clocking | Protocols |

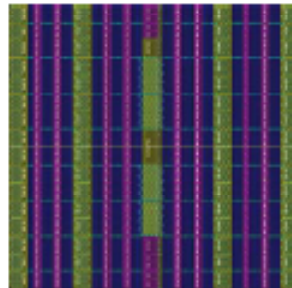
Xilinx domain optimized platforms

Column based features



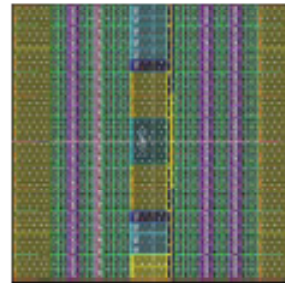
-  Logic
-  Memory
-  DSP
-  Processing
-  High-speed I/O

Virtex-4 LX



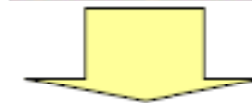
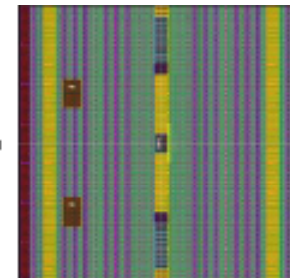
Logic Domain
Highest logic density

Virtex-4 SX



DSP Domain
Highest DSP performance

Virtex-4 FX



Connectivity Domain
Embedded Processors
High-speed Serial I/O

Enables "Dial-In" hard IP Mix

Logic, DSP, BRAM, I/O, MGT, DCM, PowerPC

Enabled by Flip-Chip Packaging

I/O Columns Distributed Throughout the Device

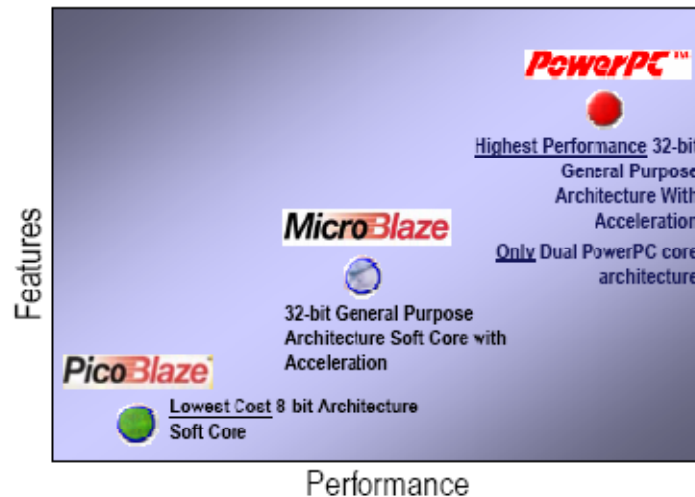


Platform for embedded processing

- ❖ CPU centric system
- ❖ Needs to increase the CPU performance utilizing the reconfigurability
 - Add instructions, add co-processor
 - Fast communication is needed

• Range of processing solutions

Soft MicroBlaze
150DMIPS
45 cents
200 in single FPGA



Plus: A broad range of common peripherals and IP

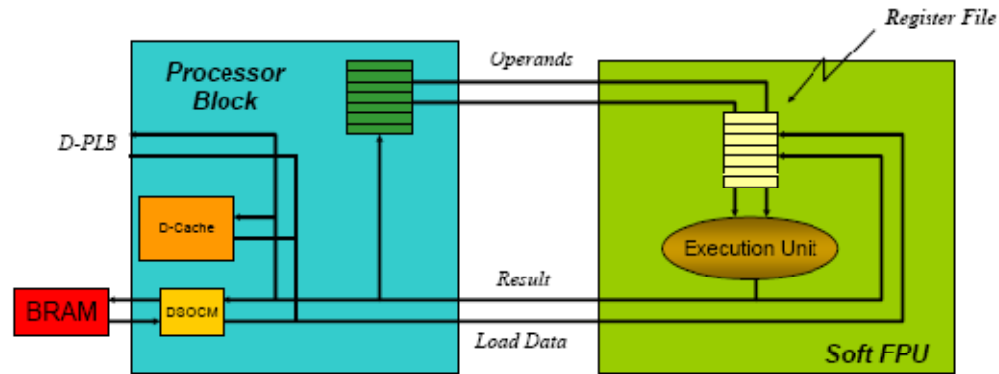
Hard CPU core

❖ PowerPC 405 RISC Core

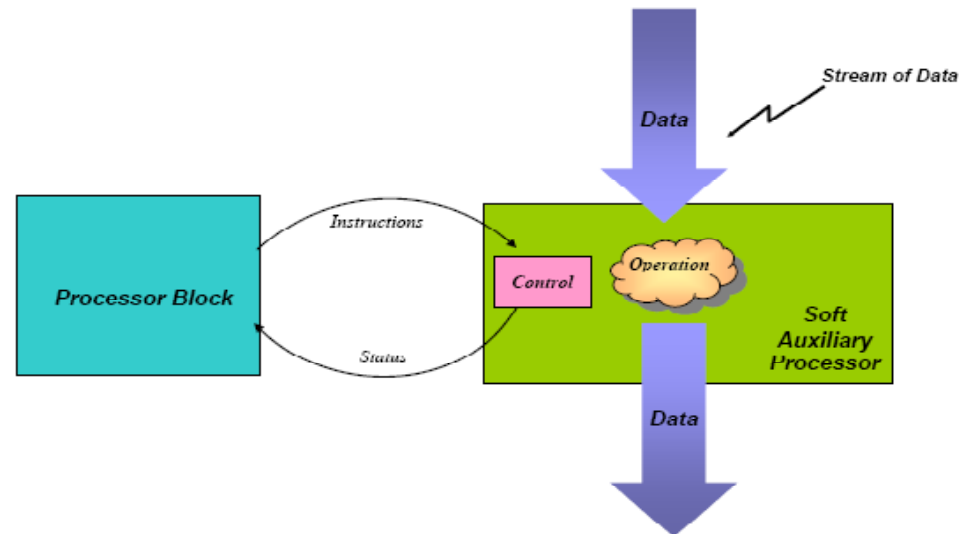
- Embedded PowerPC 405 (PPC405) core
 - Up to 450 MHz operation
 - Five-stage data path pipeline
 - 16 KB instruction cache
 - 16 KB data cache
 - Enhanced instruction and data on-chip memory (OCM) controllers
 - Additional frequency ratio options between PPC405 and Processor Local Bus
- Auxiliary Processor Unit (APU) Interface for direct connection from PPC405 to coprocessors in fabric
 - APU can run at different clock rates
 - Supports autonomous instructions: no pipeline stalls
 - 32-bit instruction and 64-bit data
 - 4-cycle cache line transfer

Use model : Coprocessor

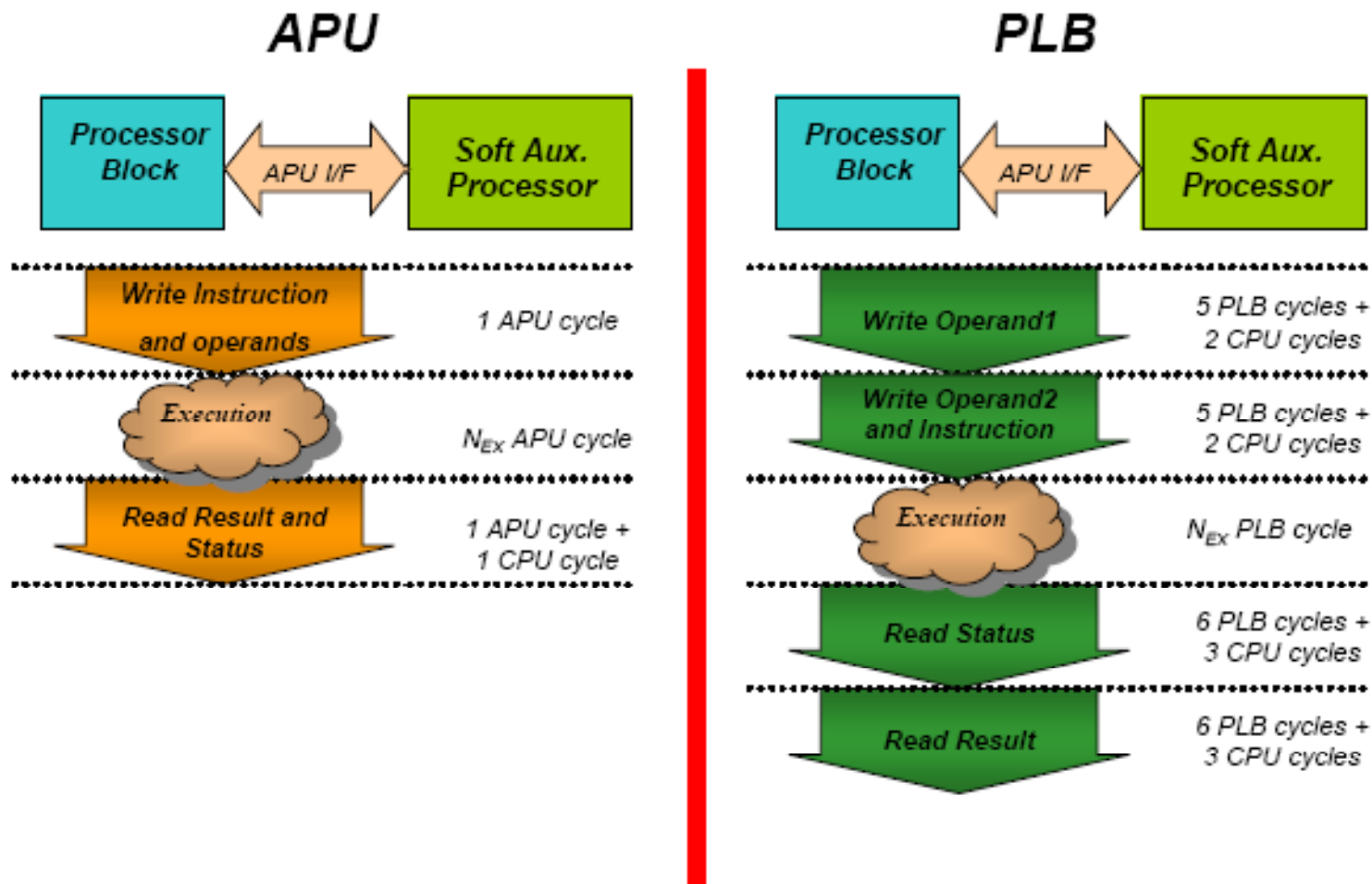
Floating Point Unit



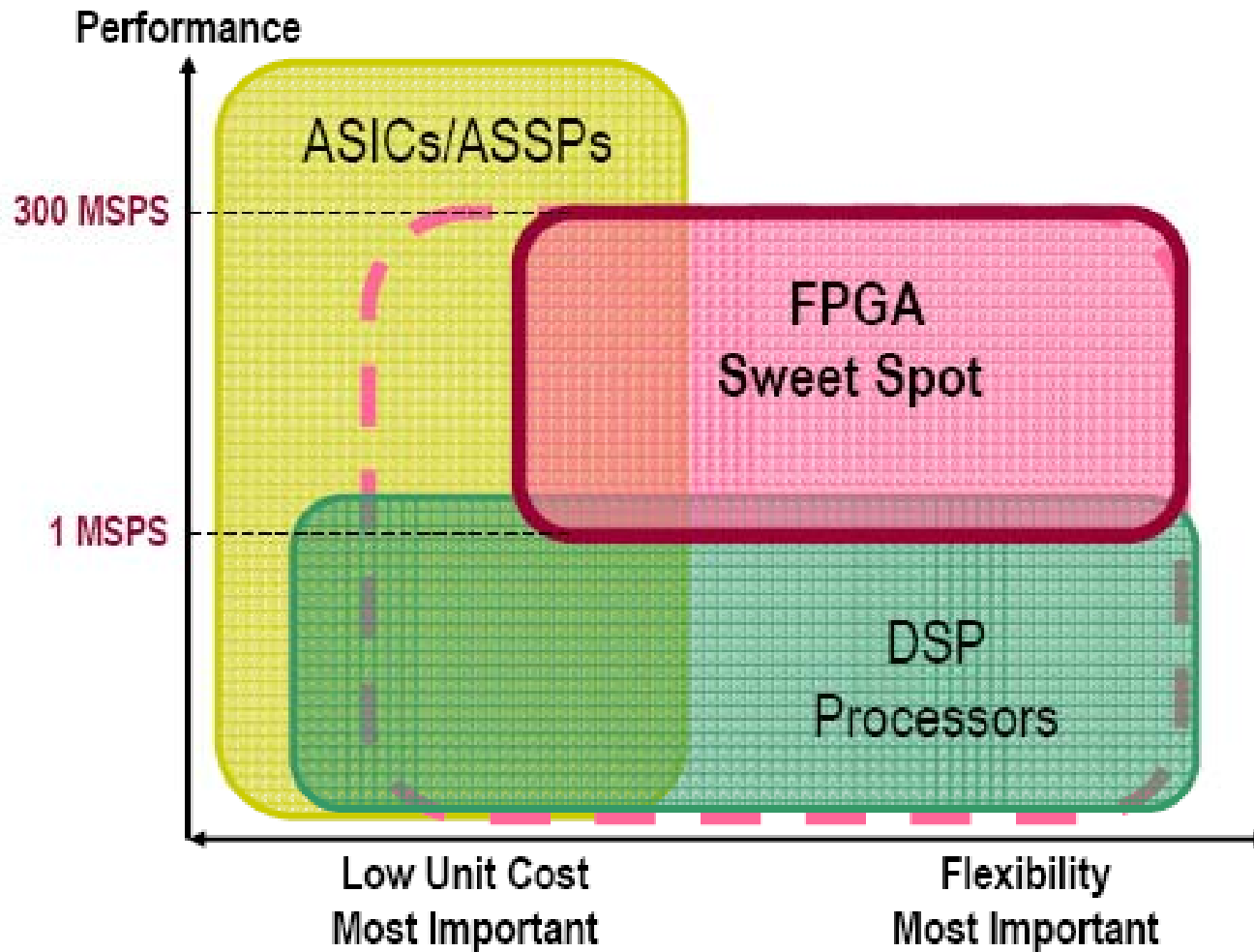
Use Model : Streaming



Comparison with Traditional Bus-based



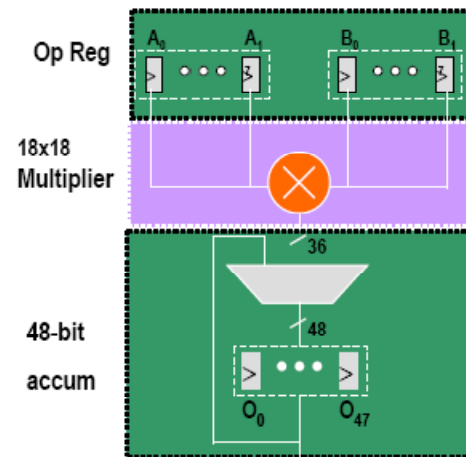
DSP domain specific platform



❖ 500 MHz XtremeDSP Slices

- Dedicated 18-bit x 18-bit multiplier, multiply-accumulator, or multiply-adder blocks
- Optional pipeline stages for enhanced performance
- Optional 48-bit accumulator for multiply accumulate (MACC) operation
- Integrated adder for complex-multiply or multiply-add operation
- Cascadeable Multiply or MACC

The DSP 'LUT'



500MHz

Available on all Virtex-4 Platforms



Other Virtex 4 features

❖ 500 MHz Integrated Block Memory

- Up to 10Mb of integrated block memory
- Optional pipeline stages for higher performance
- Multi-rate FIFO support logic
 - Full and Empty Flag support
 - Fully programmable AF and AE Flags
 - Synchronous/ Asynchronous Operation
- Dual-port architecture
- Independent read and write port width selection (RAM only)
- 18 Kbit blocks (memory and parity/sideband memory support)
- Configurations from 16K x 1 to 512 x 36 (4K x 4 to 512 x 36 for FIFO operation)
- Byte-write capability (connection to PPC405, etc.)
- Dedicated cascade routing to form 32K x 1 memory without using FPGA routing

Other Virtex 4 features (2)

❖ **SelectIO Technology**

- Up to 960 user I/Os
- Wide selections of I/O standards from 1.5V to 3.3V
- Extremely high-performance
 - 600 Mb/s HSTL & SSTL (on all single-ended I/O)
 - 1 Gb/s LVDS (on all differential I/O pairs)
- True differential termination
- Selected low-capacitance I/Os for improved signal integrity
- Same edge capture at input and output I/Os
- Memory interface support for DDR and DDR-2 SDRAM, QDR-II, RLDRAM-II, and FCRAM-II

System Blocks Specific to the FX Family (3)

❖ Tri-mode Ethernet Media Access Controller

- IEEE 802.3 compliant
- Operates at 10, 100, and 1,000 Mb/s
- Supports tri-mode auto-detect
- Receive address filter
- Fully monolithic 1000Base-X solution with RocketIO MGT
- Implements SGMII through RocketIO MGT to external PHY device
- Supports multiple PHY (MII, GMII, etc.) interfaces through an I/O resource
- Receive and transmit statistics available through separate interfaces
- Separate host and client interfaces
- Support for jumbo frames

Virtex-4 Family Members

| Device | Configurable Logic Blocks (CLBs) ⁽¹⁾ | | | | XtremeDSP Slices ⁽²⁾ | Block RAM | | DCMs | PMCDs | PowerPC Processor Blocks | Ethernet MACs | RocketIO Transceiver Blocks | Total I/O Banks | Max User I/O |
|-----------|---|-------------|--------|--------------------------|---------------------------------|--------------|--------------------|------|-------|--------------------------|---------------|-----------------------------|-----------------|--------------|
| | Array Row x Col | Logic Cells | Slices | Max Distributed RAM (Kb) | | 18 Kb Blocks | Max Block RAM (Kb) | | | | | | | |
| XC4VLX15 | 64 x 24 | 13,824 | 6,144 | 96 | 32 | 48 | 964 | 4 | 0 | N/A | N/A | N/A | 9 | 320 |
| XC4VLX25 | 96 x 28 | 24,192 | 10,752 | 168 | 48 | 72 | 1,296 | 8 | 4 | N/A | N/A | N/A | 11 | 448 |
| XC4VLX40 | 128 x 36 | 41,472 | 18,432 | 288 | 64 | 96 | 1,728 | 8 | 4 | N/A | N/A | N/A | 13 | 640 |
| XC4VLX60 | 128 x 52 | 59,904 | 26,624 | 416 | 64 | 160 | 2,880 | 8 | 4 | N/A | N/A | N/A | 13 | 640 |
| XC4VLX80 | 160 x 56 | 80,640 | 35,840 | 560 | 80 | 200 | 3,600 | 12 | 8 | N/A | N/A | N/A | 15 | 768 |
| XC4VLX100 | 192 x 64 | 110,592 | 49,152 | 768 | 96 | 240 | 4,320 | 12 | 8 | N/A | N/A | N/A | 17 | 960 |
| XC4VLX160 | 192 x 88 | 152,064 | 67,584 | 1056 | 96 | 288 | 5,184 | 12 | 8 | N/A | N/A | N/A | 17 | 960 |
| XC4VLX200 | 192 x 116 | 200,448 | 89,088 | 1392 | 96 | 336 | 6,048 | 12 | 8 | N/A | N/A | N/A | 17 | 960 |
| XC4VSX25 | 64 x 40 | 23,040 | 10,240 | 160 | 128 | 128 | 2,304 | 4 | 0 | N/A | N/A | N/A | 9 | 320 |
| XC4VSX35 | 96 x 40 | 34,560 | 15,360 | 240 | 192 | 192 | 3,456 | 8 | 4 | N/A | N/A | N/A | 11 | 448 |
| XC4VSX55 | 128 x 48 | 55,296 | 24,576 | 384 | 512 | 320 | 5,760 | 8 | 4 | N/A | N/A | N/A | 13 | 640 |
| XC4VFX12 | 64 x 24 | 12,312 | 5,472 | 86 | 32 | 36 | 648 | 4 | 0 | 1 | 2 | N/A | 9 | 320 |
| XC4VFX20 | 64 x 36 | 19,224 | 8,544 | 134 | 32 | 60 | 1,224 | 4 | 0 | 1 | 2 | 8 | 9 | 320 |
| XC4VFX40 | 96 x 44 | 41,904 | 15,552 | 243 | 48 | 144 | 2,592 | 8 | 4 | 2 | 4 | 12 | 11 | 448 |
| XC4VFX60 | 128 x 52 | 56,880 | 25,280 | 395 | 128 | 232 | 4,176 | 12 | 8 | 2 | 4 | 15 | 13 | 576 |
| XC4VFX100 | 160 x 68 | 94,896 | 42,176 | 659 | 160 | 376 | 6,768 | 12 | 8 | 2 | 4 | 20 | 15 | 768 |
| XC4VFX140 | 192 x 84 | 142,128 | 63,168 | 987 | 192 | 552 | 9,936 | 20 | 8 | 2 | 4 | 24 | 17 | 896 |

Design Time



No additional engineering resources, tools, or conversion processes are required.

Cost

| | Virtex-4 EasyPath | Structured ASICs | Cell-Based ASICs |
|--|--|--|--|
| Product offering Density Technology Node | High-Density (XC4VFX140) 90nm | High-Density FPGAs 130nm | High-Density FPGAs+ 130nm |
| Total Cost Vendor NRE % Cost Reduction Tools Hidden Costs Minimum Orders | \$75K 50-80% None** None Low to moderate | \$100K-200K* 50-80% ~\$120K-\$250K* Engineering, Tools, Respin, Delay Moderate to High | \$1M-\$3M* 90% >\$300K* Engineering, Tools, Respin, Delay High |
| Risk Feature Parity Performance Functionality Architecture | Guaranteed Match Guaranteed Match Guaranteed Match Guaranteed Match | Not Guaranteed Need to re-optimize Converted Converted | Not Guaranteed Need to re-optimize Converted Converted |
| Incremental Investment Conversion Verification Qualification Board Re-Design | None None None None | Vendor Resources Customer Customer Customer | Customer & Vendor Customer Customer Customer |
| Lead Times+ To Prototypes To Production Re-order | Not required 8 weeks Days | 8-10 weeks 16-20 weeks Weeks-Months | Weeks-Months Months Weeks |

3. Comparison of FPGA and Programmable DSP

❖ Programmable digital signal processors

- Texas Instruments TMS 320C5x, TMS 64x (VLIW)
- Assembly and C program based programming
- Faster than ordinary embedded processors for DSP
- The number of multipliers in C64x (16*16): 4
- In short: easier design, but limited throughput
- Applications: voice coder, video decoder, ordinary control

❖ FPGAs

- Large number of arithmetic elements: hundreds of hardware multipliers
- Large amount of block RAMs (large bandwidth)
- Needs a hardware level design (VHDL)
- In short: design is more difficult, but throughput is unlimited
- Application: high bandwidth communication system design

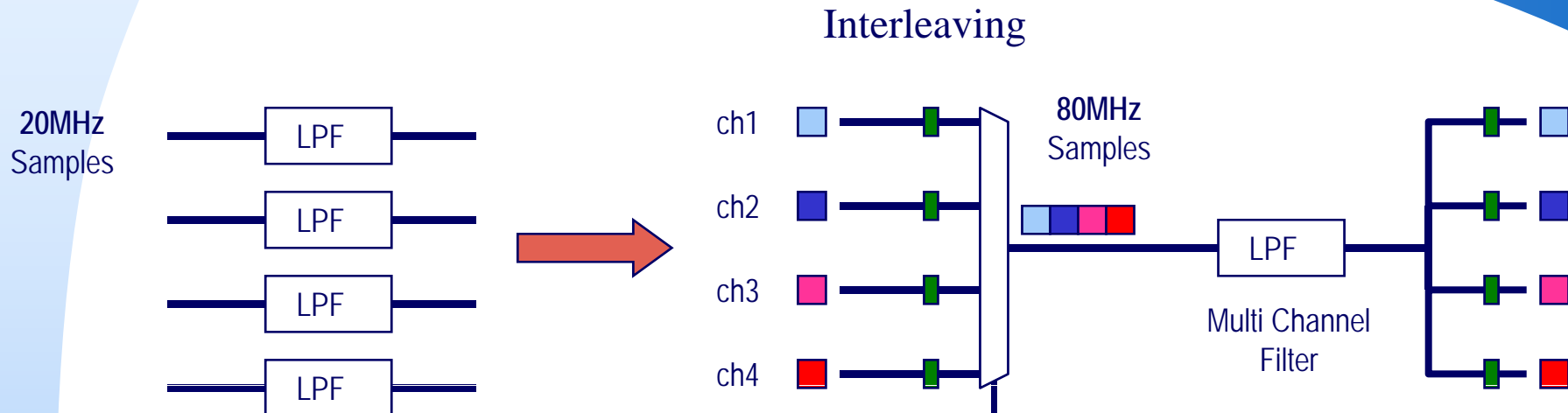
Why DSP in FPGAs

- ❖ **Availability of fast analog-to-digital converters (ADCs)**
 - Enables digital methods for functions traditionally done in RF components (supports high sampling rates)
- ❖ **Massive parallel processing**
 - FPGAs may have several hundred embedded multipliers on-chip
 - One FPGA can replace many DSP Processors

Architectural Considerations

- ❖ **FPGA architectures are vendor specific**
 - Unlike ASICs, no two are alike
- ❖ **FPGA vendors develop distinct competencies**
 - In device architecture design
 - In intellectual property (dsp functions, bus controllers, etc)
 - In design tool flows
- ❖ **Vendor independent HDL can be written but this usually achieves mediocre results in clock speed and design size instantiation**

FPGAs Are Massive Parallel Computing Machines

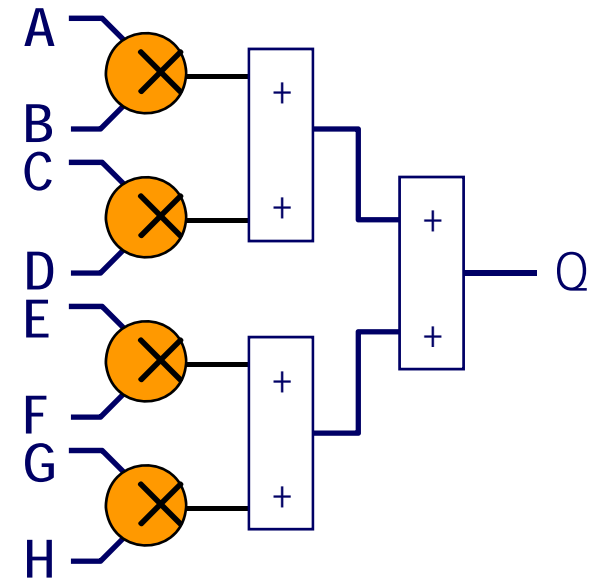


- ❖ **FPGAs are ideally suited for multi-channel DSP designs**
 - Many low sample rate channels can be multiplexed (e.g. TDM) and processed in the FPGA, at a high rate
 - Interpolation (using zeros) can also drive sample rates higher

FPGAs Allow Space/Speed Trade-offs

$$Q = (A \times B) + (C \times D) + (E \times F) + (G \times H)$$

can be implemented in parallel



But is this the only way in the FPGA?

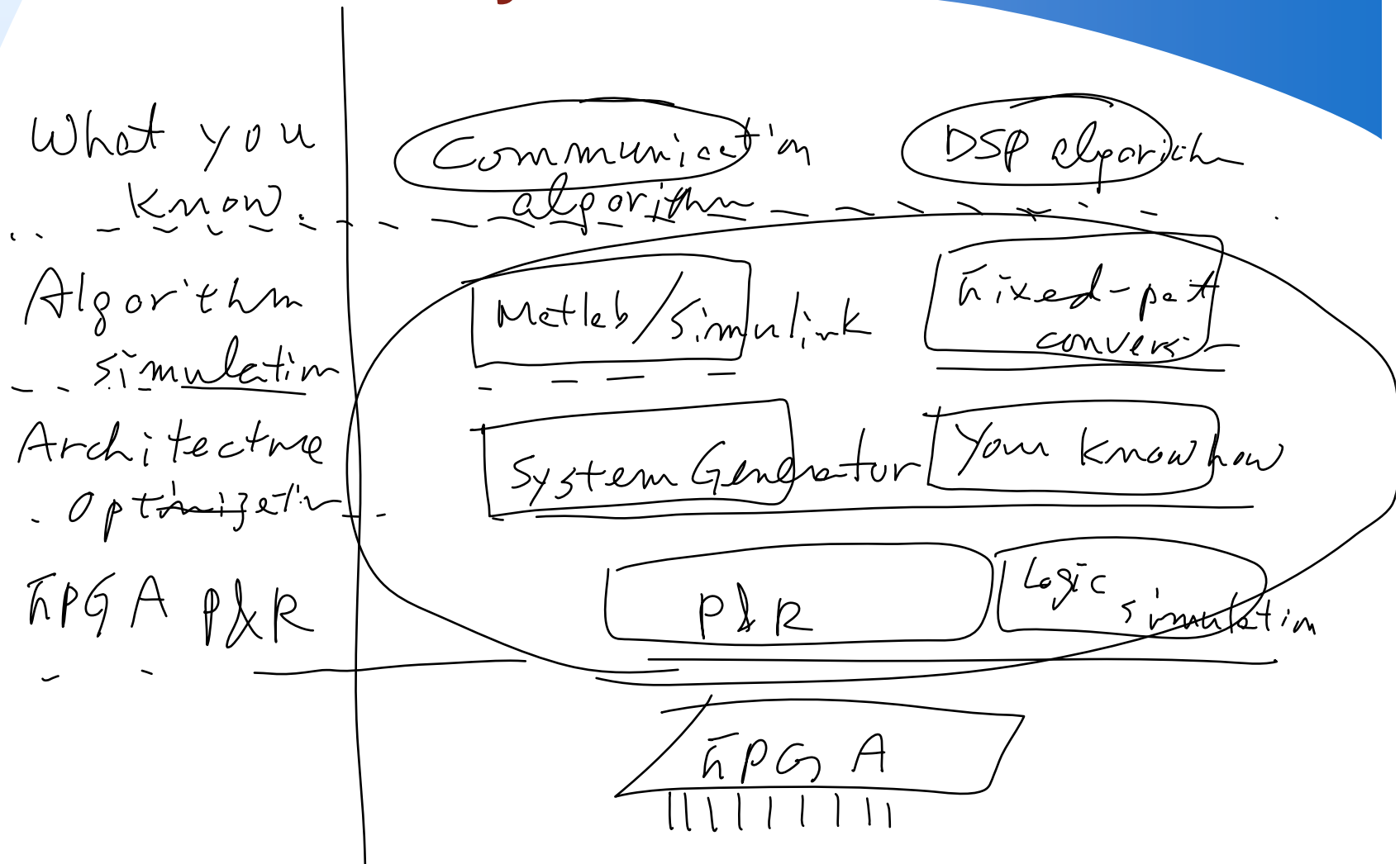
Exploiting The Xilinx Architecture For DSP Functions

- ❖ Memory Blocks that can be configured as ROMs, dual port RAMs, FIFOs
- ❖ Embedded 18x18 multipliers that can be ganged to form a 35x35 bit multiply
- ❖ SRL16 shift registers
 - A patented technique for turning the 4 input lookup table (2 per slice) into an addressable shift register

Conventional FPGA design flow (Digital System Design course)

- ❖ **RTL description**
 - (Synthesizable) VHDL or schematic (conduct simulation for design verification)
- ❖ **Logic synthesis -> netlist (logic netlist containing gates, memory, ..)**
- ❖ **Translate to FPGA configurable blocks**
 - (logic simulation with logic delay)
- ❖ **Place and route**
- ❖ **Timing verification (backannotation)**
- ❖ **Download programming data and Run**

4. FPGA top-down design flow for comm. systems



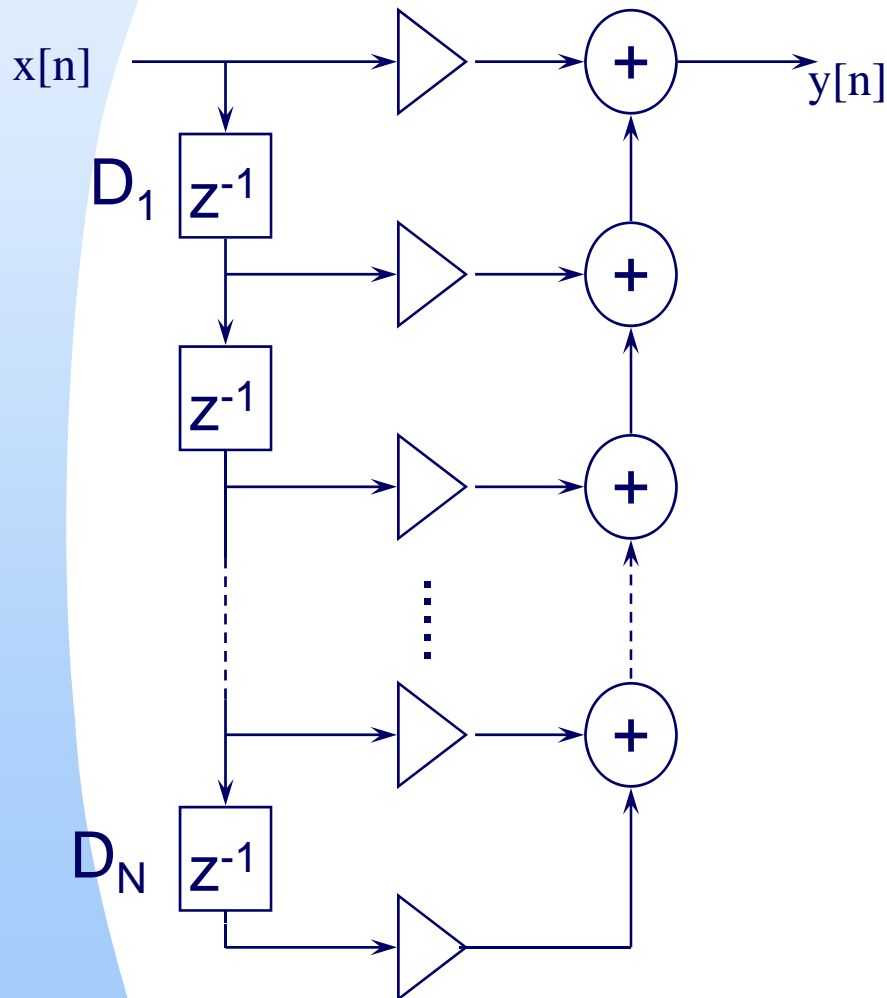
FPGA based design example

❖ An FIR Filter

- Filter coefficients, order determined by a filter design program <- Covered by Digital Signal Processing course
- We learned logic design

❖ A straightforward approach (for 100^{tap} filter):

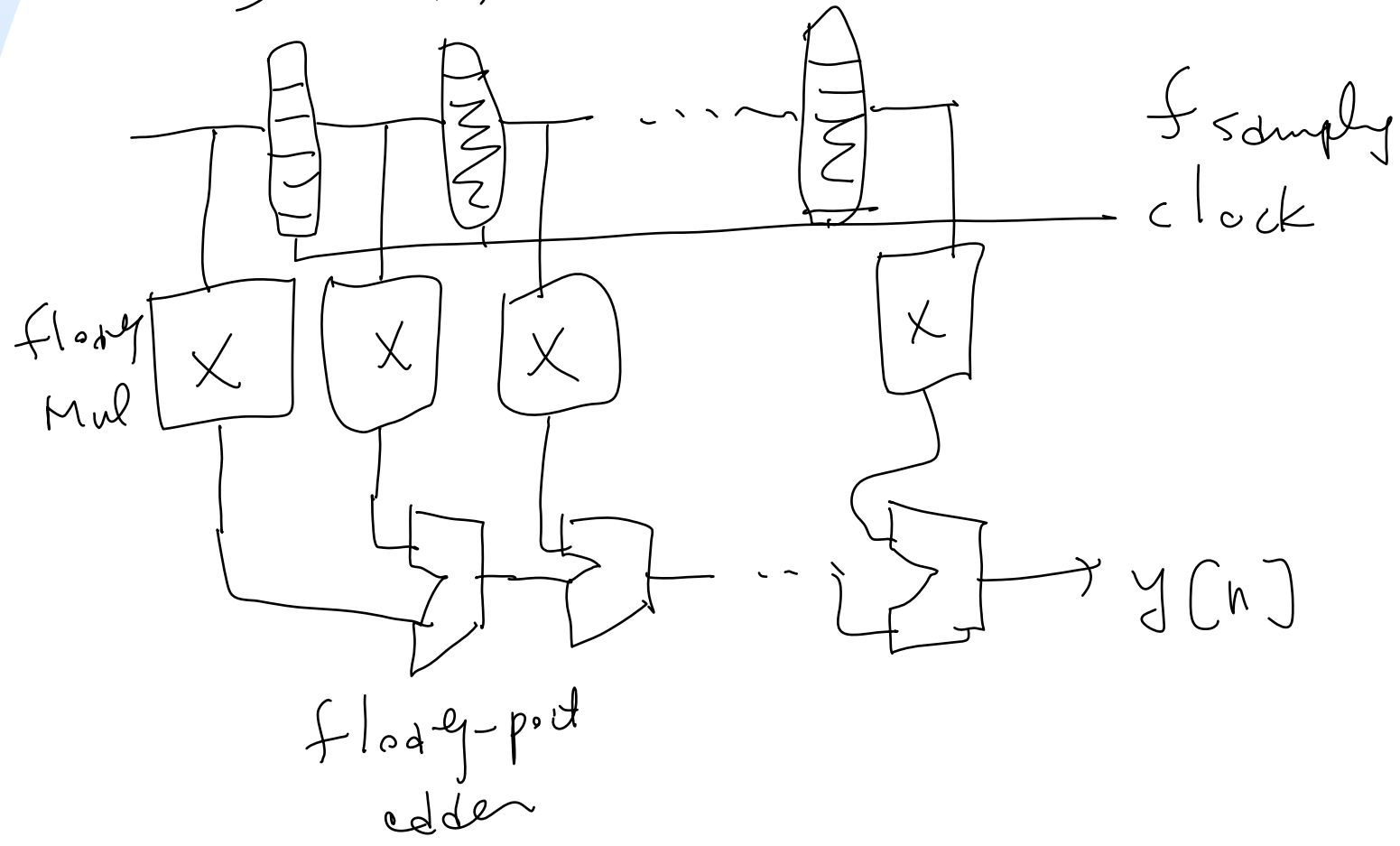
- Multiplication/ addition in the Signal Flow Graph -> A floating-point hardware multiplier/adder
- Delay (z^{-1}) -> 32bit flip-flop array (the IEEE floating-point standard use 32bit)
- Clock -> sampling clock frequency



Design results, are you satisfied?

- ❖ **When $f_{\text{sampling}} = 10 \text{ KHz}, 1\text{MHz}, 10\text{MHz}, 100\text{MHz}, 500\text{MHz}$**
- ❖ **The hardware resources: 100 floating-point multipliers and 99 adders + latches**

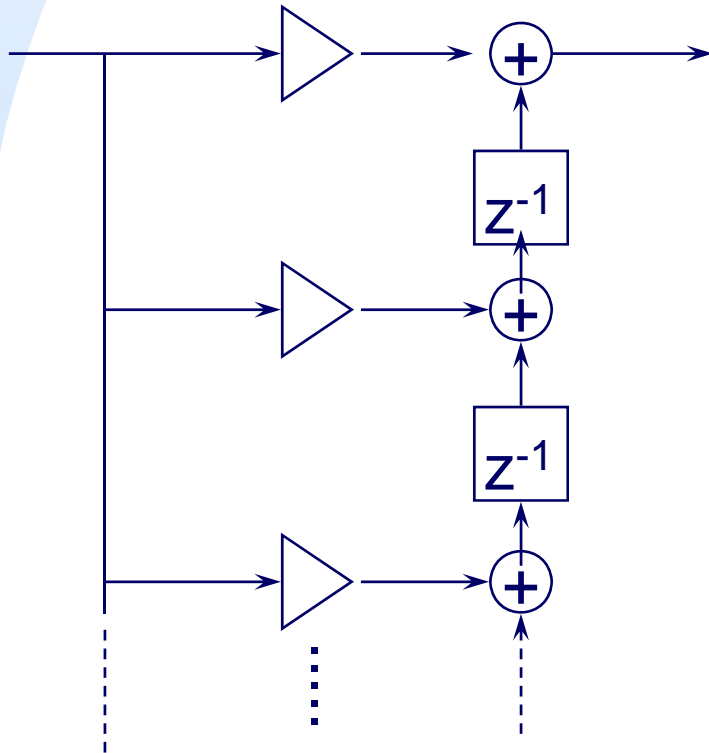
32 bit f/f



Speed consideration

- ❖ Assuming 10ns for an adder/multiplier, what would be the maximum operating frequency of this hardware?
- ❖ Ans: just 1MHz (not 100MHz)
- ❖ How we can improve the speed?
- ❖ Using a high speed circuit?
 - Maybe you can get twice the speed, but not 10 times or over.
 - You need signal flow graph transformation

Signal flow graph transformation



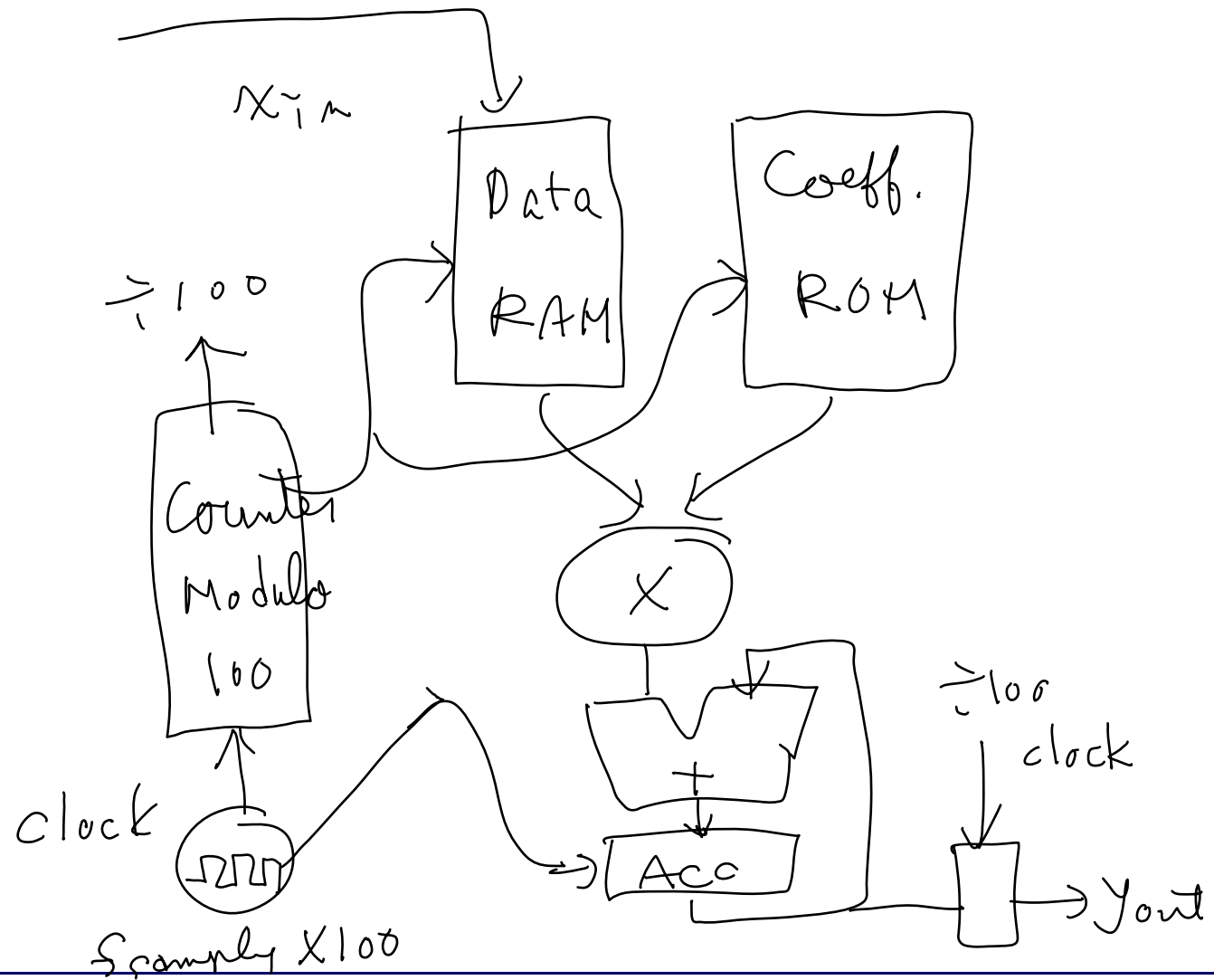
- ❖ The critical path delay is reduced to $T_{mul} + T_{adder}$ (or 20ns in this example)
- ❖ 50 times reduction (still there is a problem of broadcasting network).

Floating-point to fixed-point conversion

- ❖ **If the needed SQNR and the dynamic range is not extremely high (say 100 dB), it is reasonable to use fixed-point hardware**
 - We need to determine the appropriate word-length (mostly less than 16 bits unless for high performance audio applications)
 - If you can reduce the word-length further while satisfying the system performance, this would be good for efficient hardware implementation
 - We need to scale the signals (implemented using shift operations) to prevent overflows.
- ❖ **Benefits of fixed-point conversion (say floating-point -> 16 bit conversion)**
 - About two~four times of speed improvement
 - Two to four times of hardware saving

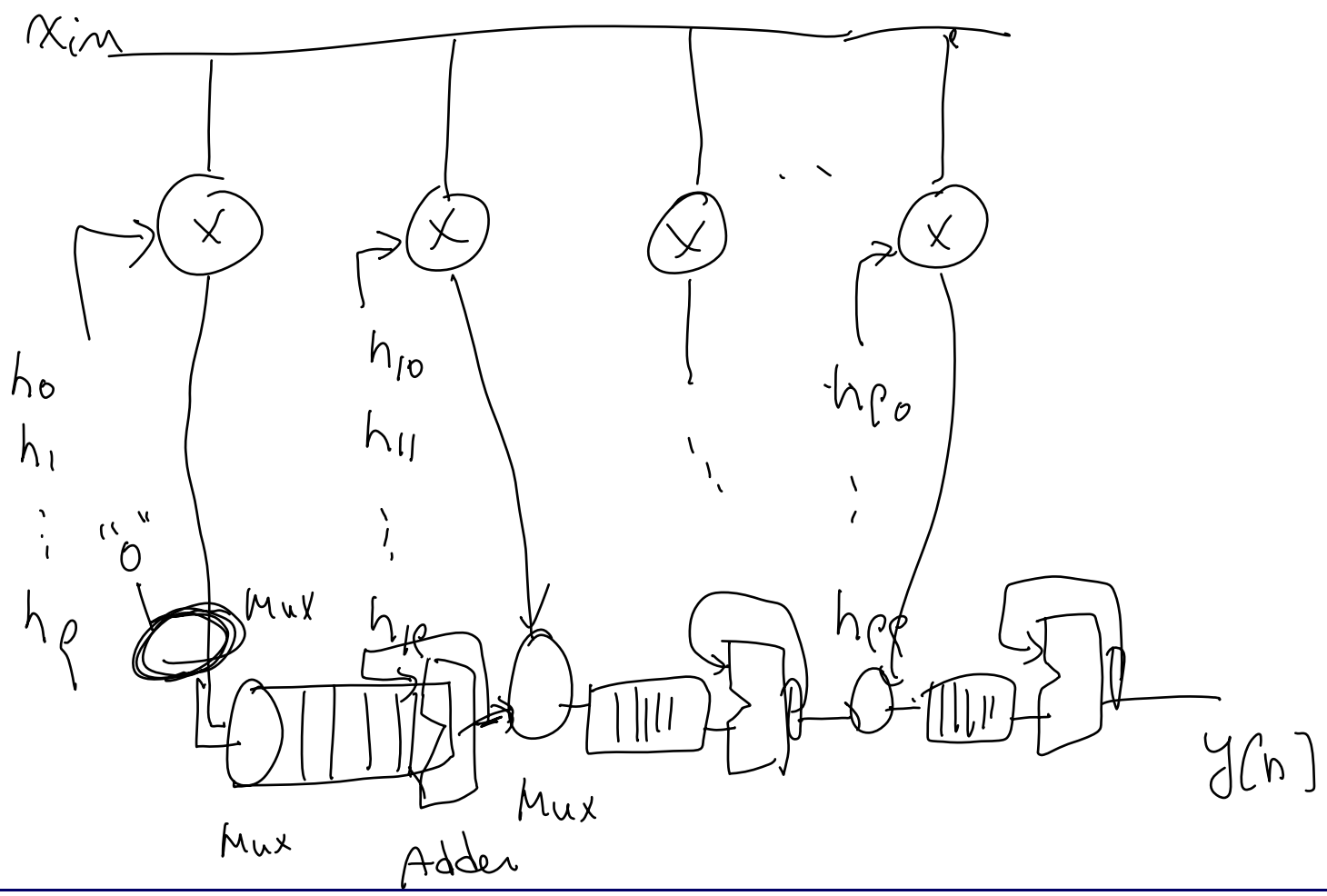
Architecture optimization #1: for a system with a low sampling clock frequency

- ❖ Assume $f_{\text{sampling}} = 1$ MHz or lower
- ❖ Most of the time, the hardware is in idle (max hardware speed is, e.g. 100MHz)
- ❖ Flip flops are not area efficient
- ❖ Serialized implementation
 - 1 multiplier, 1 adder
 - 1 block RAM (instead of 100 flip flop arrays)
- ❖ Speed: OK upto $f_{\text{sampling}} =$ about 0.5MHz (or 1MHz by further tuning)



Architecture #2: Time-multiplexed Implementation

- ❖ What if f_{sampling} higher than 1 MHz, but much lower than 100MHz
- ❖ We need to employ multiple multipliers and adders, shift registers: bandwidth matching.
 - Say $f_{\text{sampling}} = 10\text{MHz}$, 10 hardware blocks
 - Shift registers (instead of FF's or block RAM)



High throughput system

❖ **What is $f_{\text{sampling}} = 500\text{MHz}$?**

❖ **Answers:**

- Learn circuit design? Needs to pay high NRE for VLSI fabrication
- Parallel and pipelined architecture
 - Pipelining: reduce the critical path delay by inserting pipelining registers
 - Parallel: duplicate the circuit so that the throughput requirement for each circuit is reduced.
- Parallel and pipelining transforms are not always possible, especially when there is feedback, it is difficult to apply.

Summary

- ❖ **There is a large gap between the DSP algorithm and actual system**
 - Signal flow graph transformation: we get x50 critical path delay reduction in this example
 - Floating-point to fixed-point conversion: we obtained about 10 times improvement in area x delay (area times delay product) performance figure (a few times for area, a few times for delay)
 - Architecture optimization: up to 100 times improvement in hardware cost saving.
 - Bandwidth matching: the number of arithmetic elements needed is $f(\text{filter order and } f_{\text{sampling}})$
 - Implementation of delay blocks: flip flops (for high sampling clock), shift registers, and RAM block (for low speed)

This course covers

- ❖ **FPGA structure and general design tool**
- ❖ **Top-down communication system design flow for FPGA (higher design productivity)**
 - Matlab, Simulink, System Generator
- ❖ **FPGA and VLSI architecture for digital signal processing**
 - Signal processing algorithm to VLSI architecture transformation (time multiplexing, retiming, distributed arithmetic)
 - Floating to fixed-point conversion
- ❖ **Optimization of DSP algorithms for real-time implementation**
 - Digital down converter, CORDIC, adaptive filtering

Conclusions

- Programmable FPGA platforms are at the forefront of electronic systems design
- Simple entry point is to treat system on chip as shrunken version of system on board
- Opportunities for innovative use of (programmable) functional IP units, memories and (configurable) networks on chip
- Alternatives require not just new tools and methodologies, but also *new thinking*

