Reconfigurable Interconnect for Next Generation Systems

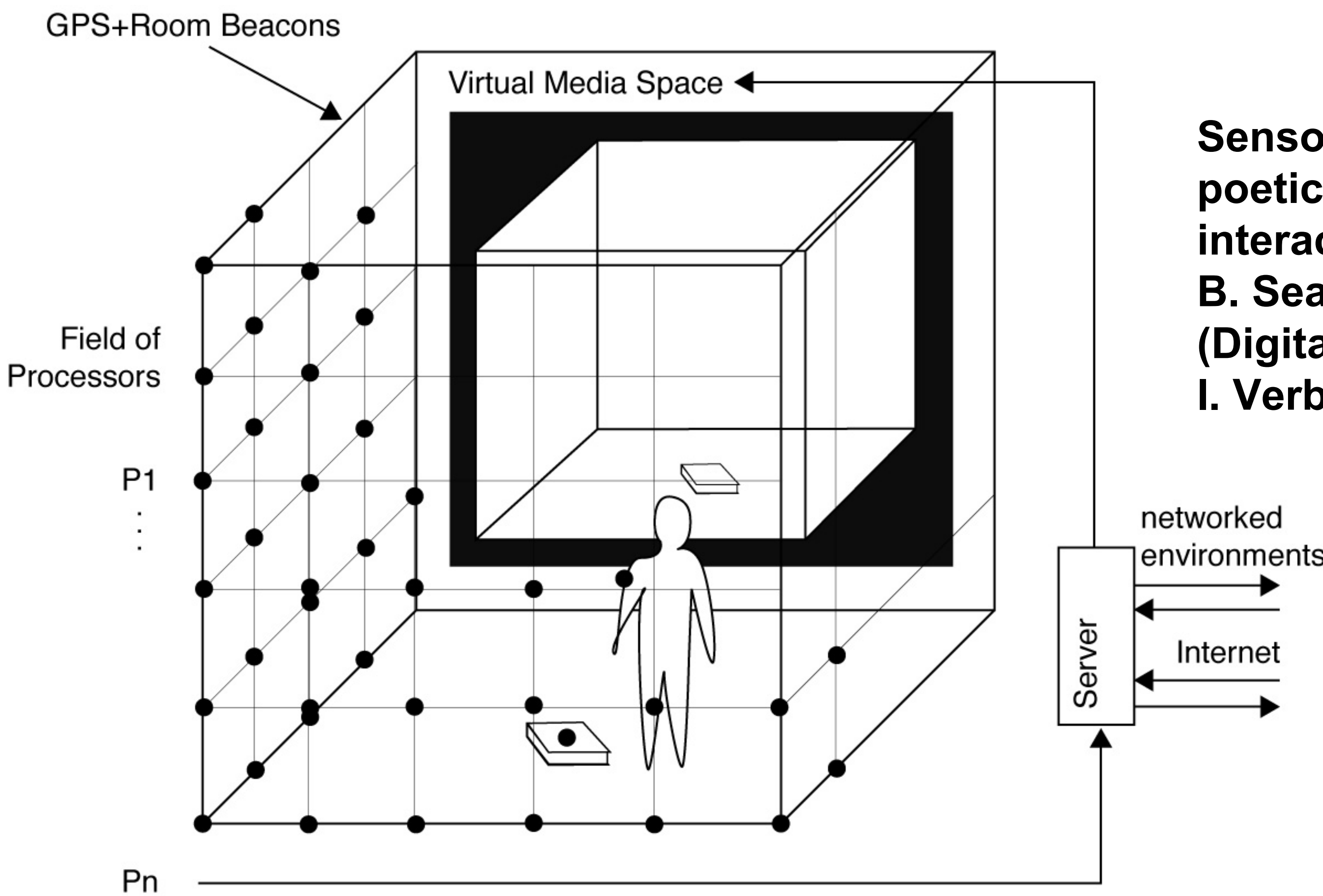
Ingrid Verbauwhede, M.F. Chang
Electrical Engineering Department, UCLA
7440B Boelter Hall
ingrid@ee.ucla.edu

with thanks to: Jongsun Kim, Patrick Schaumont, UCLA

Outline

- Post PC Era – Heterogeneous distributed embedded systems
- Challenge 1: Energy – Flexibility conflict
- Approach: Architecture design
  - Domain specialization
  - Reconfiguration hierarchy
- Challenge 2: Interconnect & Memory bandwidth
- Approach:
  - Reconfigurable RF – Interconnect
  - On chip
  - Off chip
- Impact to SLIP
- Conclusions

Next generation applications

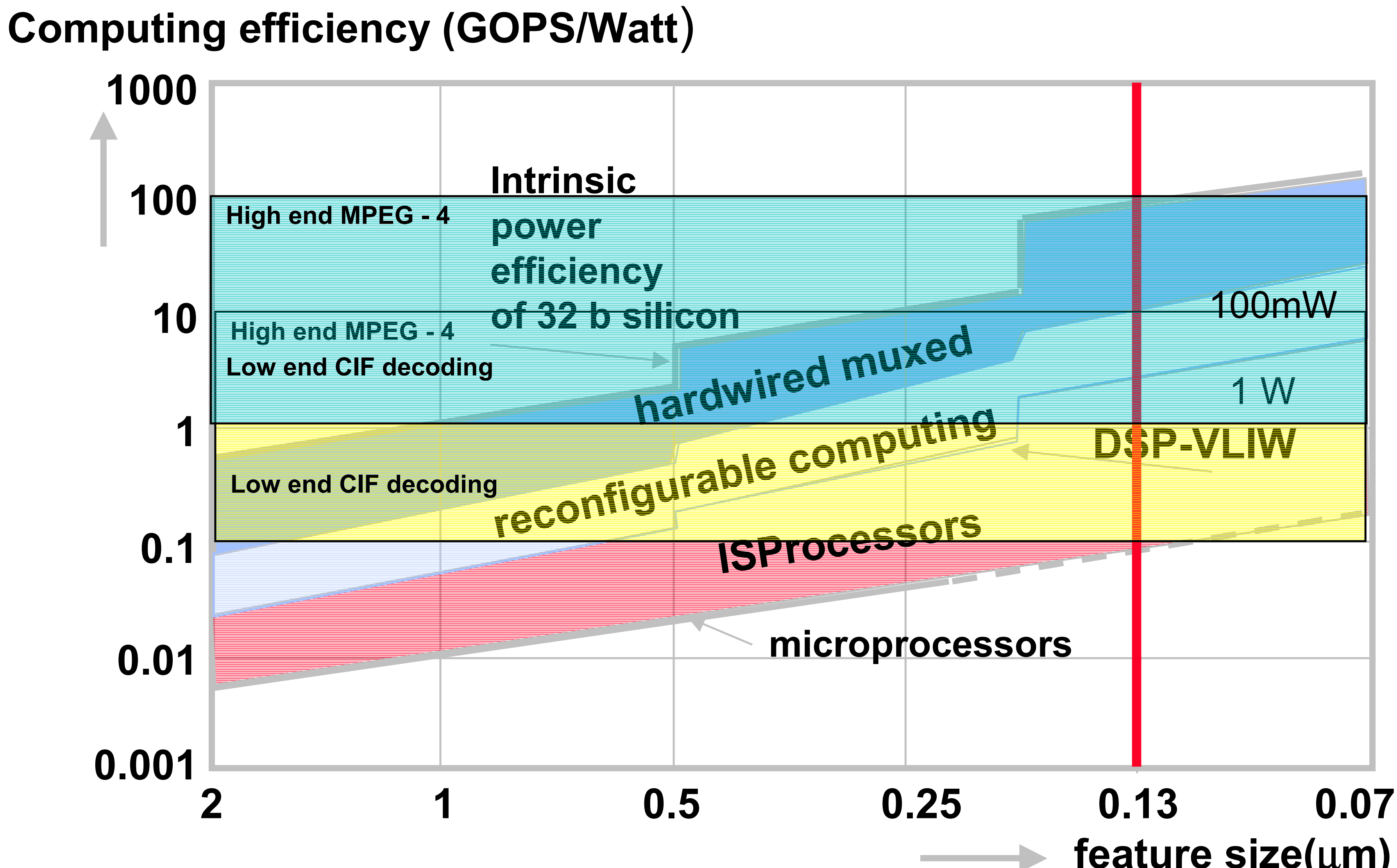


Sensor based poetic/informational interactive system
B. Seaman,
(Digital Media Arts)
I. Verbauwhede

Macromedia Shockwave: http://students.dma.ucla.edu/~fwinkler/Poly-Sensing

Challenge 1: Energy – Flexibility trade-off

The Energy-Flexibility Conflict



[T.Claasen et al. ISSCC99]

Example: AES co-processor

AES 128bit key 128bit data	Throughput	Power	Figure of Merit (Gb/s/W)
Domain Specific Processor (0.18 μm CMOS) [1]	1.28 Gbits/sec	56 mW	22.8 (100%)
FPGA [2],[3]	640 Mbits/sec	1.63 W	0.39 (1.7%)
Pentium III [4],[5]	607 Mbits/sec	41.4 W	0.015 (0.06%)

[1] H. Kuo, I. Verbauwhede, P. Schaumont, CICC2002

[2] Altera Technical Brief TB57: Xilinx VII 700 mW @ 100 MHz, 660 CLB

[3] 3th AES Candidate Conference, Elbirt et al: Xilinx VII 14.1 MHz, 5302 CLB (w/o key schedule = overhead 33%)

[4] Helger Lipmaa, http://www.cs.tut.fi/~helger/aes/rijndael.html: PIII assembly handcoded

[5] Intel Pentium III Datasheet 1.13 GHz (Vcc=1.8V, Icc=23A)

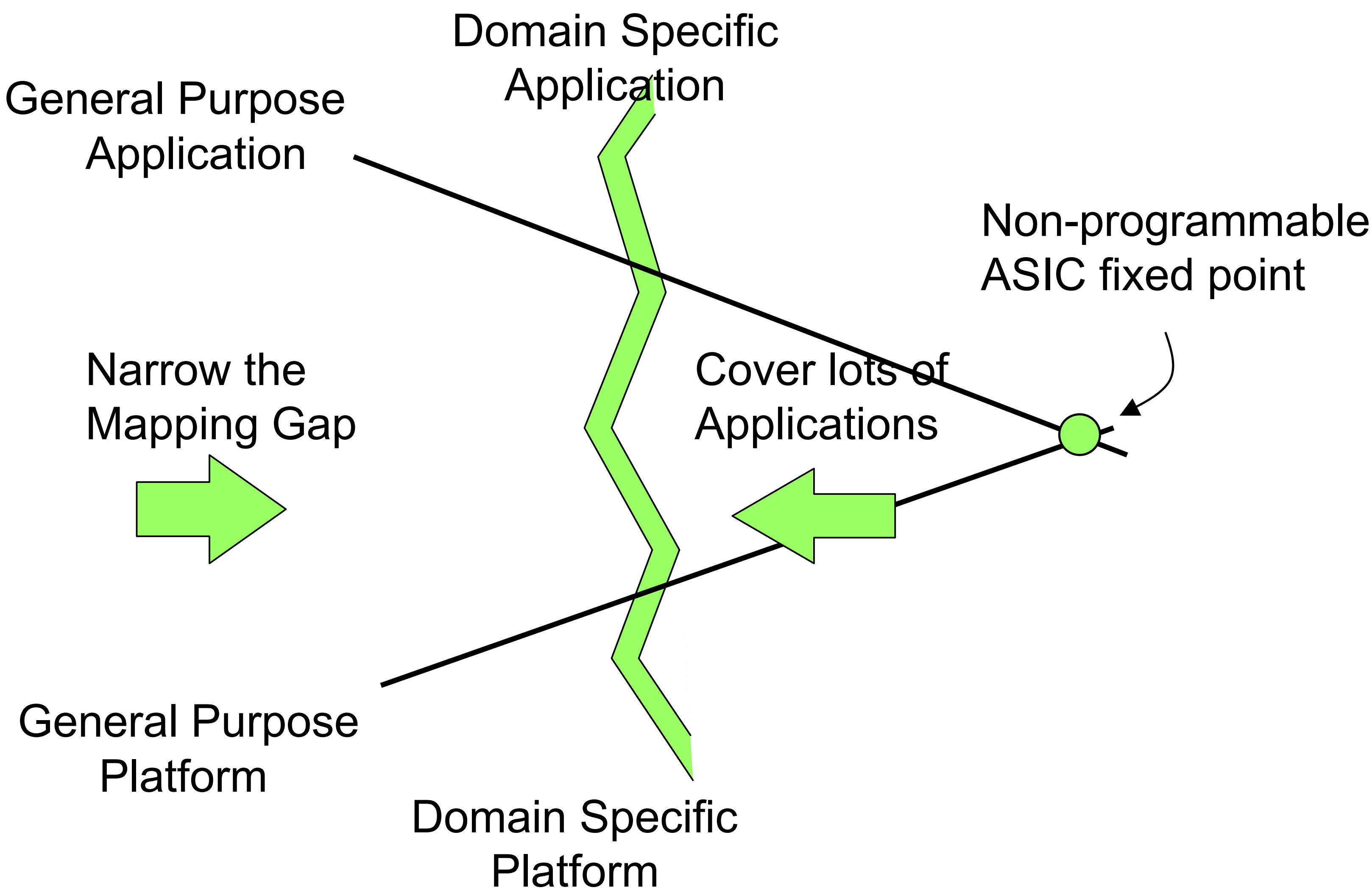
Approach: domain specialization

- General purpose processor is not the solution
  (programmable at instruction set)
- FPGA is not the solution
  (“programmable” = reconfigurable at CLB level)
- Fixed ASIC is not the solution

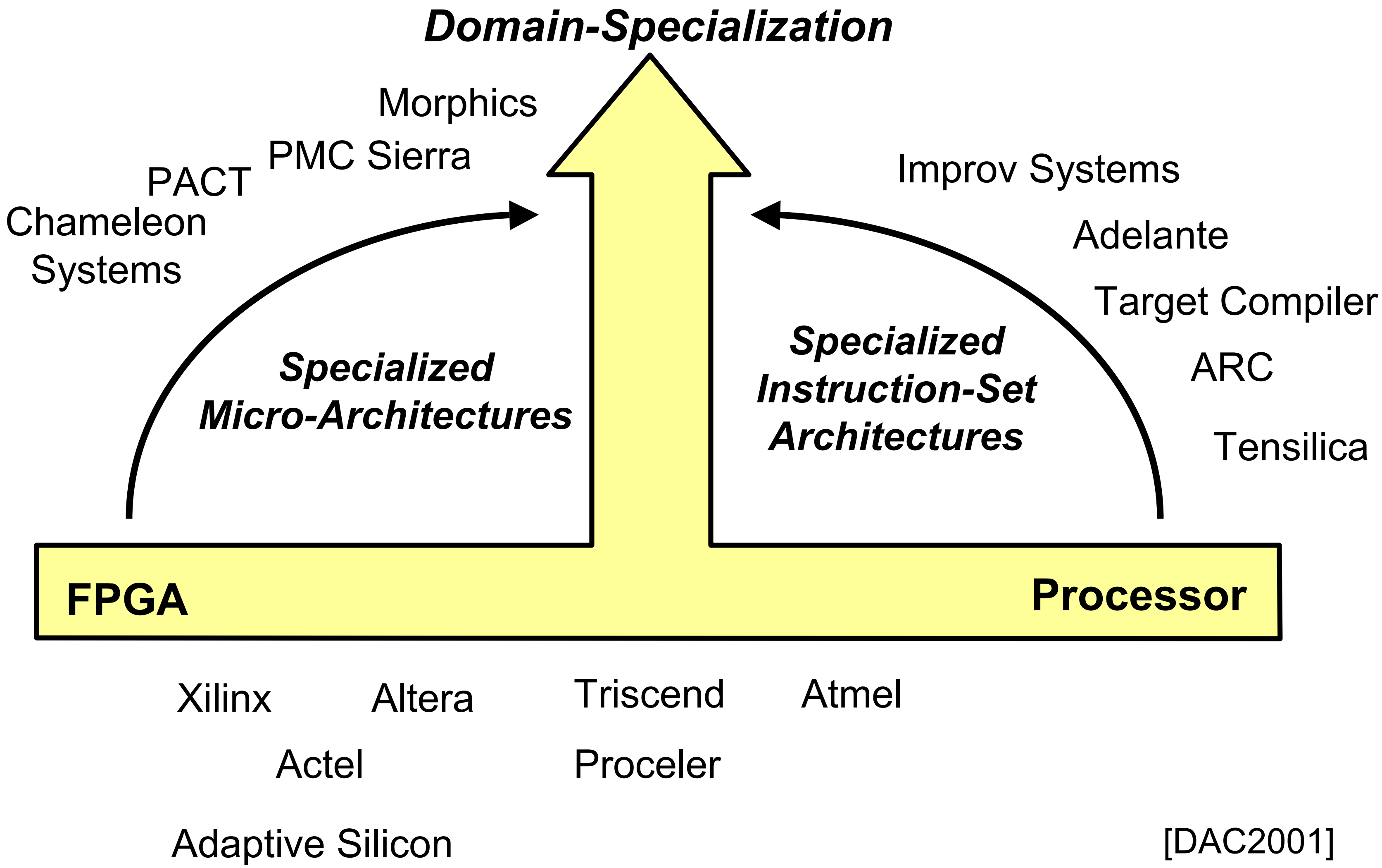
Reconfiguration hierarchy (architecture view)
& domain specialization (system view)

- Domains: wireless communications, security, network processing
  (high throughput, low energy, unusual arithmetic)

Domain specific processing

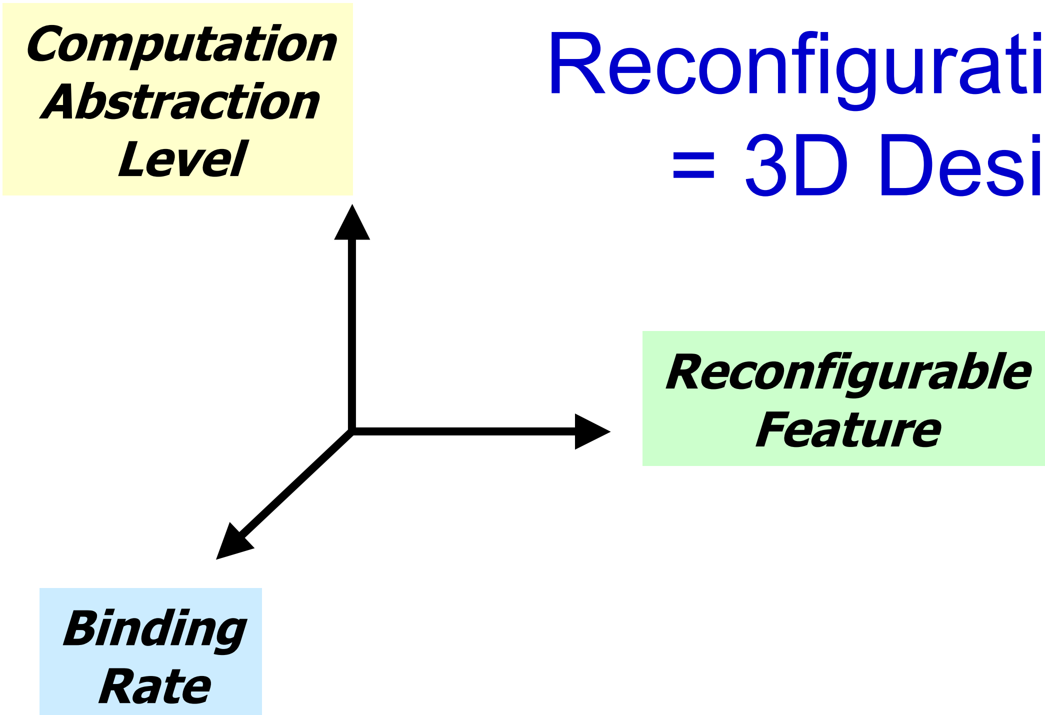


Trading on the Energy-Flexibility Boundary is hard



[DAC2001]

Reconfiguration Hierarchy = 3D Design Space



	Communication	Storage	Processing
Implementation	Switches Muxes	RAM Organization	CLB Parametrizable IP-block
Micro-Architecture	Crossbar Busses	Register File Size Cache Architecture	Execution Unit Type Interpreter Levels
Instruction Set Architecture	Size of address/data bus	Register Set Memory Architecture	Custom Instructions Interrupt Architecture
Process Architecture/ Systems Architecture	Interconnection network	Buffer Size	Number and type of asynchronou processes and tasks

Challenge 2: reconfigurable interconnect

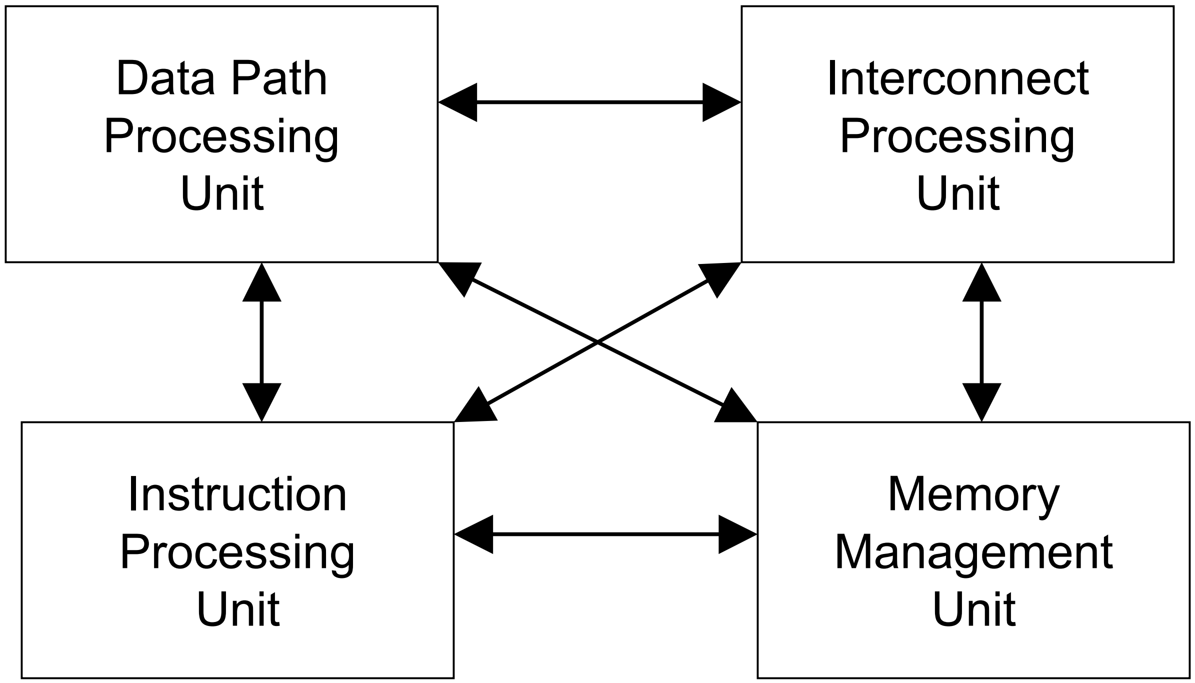
Datapaths are NOT the problem

but

Memory bandwidth and interconnect are!

DSP Processor Fundamentals

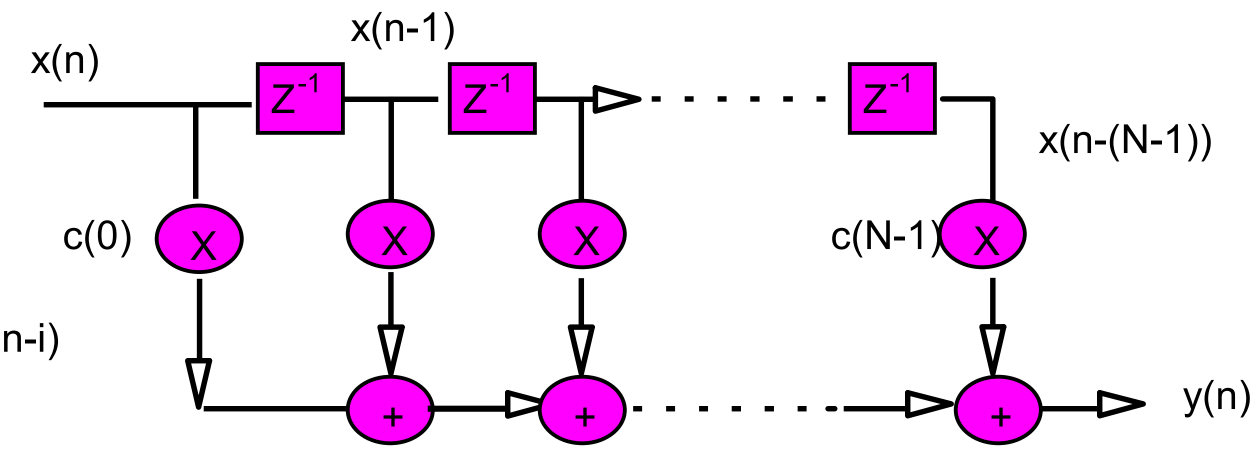
Processor Components [Skillikorn-88]



Adapt **ALL** components to the application domain!

Example: FIR implementation

$$y(n) = \sum_{i=0}^{N-1} c(i)\ x(n-i)$$




```
y(0) = c(0)x(0) + c(1)x(-1) + c(2)x(-2) + . . .  + c(N-1)x(1-N);
y(1) = c(0)x(1) + c(1)x(0) + c(2)x(-1) + . . .  + c(N-1)x(2-N);
y(2) = c(0)x(2) + c(1)x(1) + c(2)x(0) + . . .  + c(N-1)x(3-N);
. . .
y(n) = c(0)x(n) + c(1)x(n-1) + c(2)x(n-2)+ . . + c(N-1)x(n-(N-1));
```

Software (Von Neumann): Execute row by row = **2 reads for one MAC**

FIR on DSP Lode™ (1996)

- FIR filter: two outputs in parallel with delay register

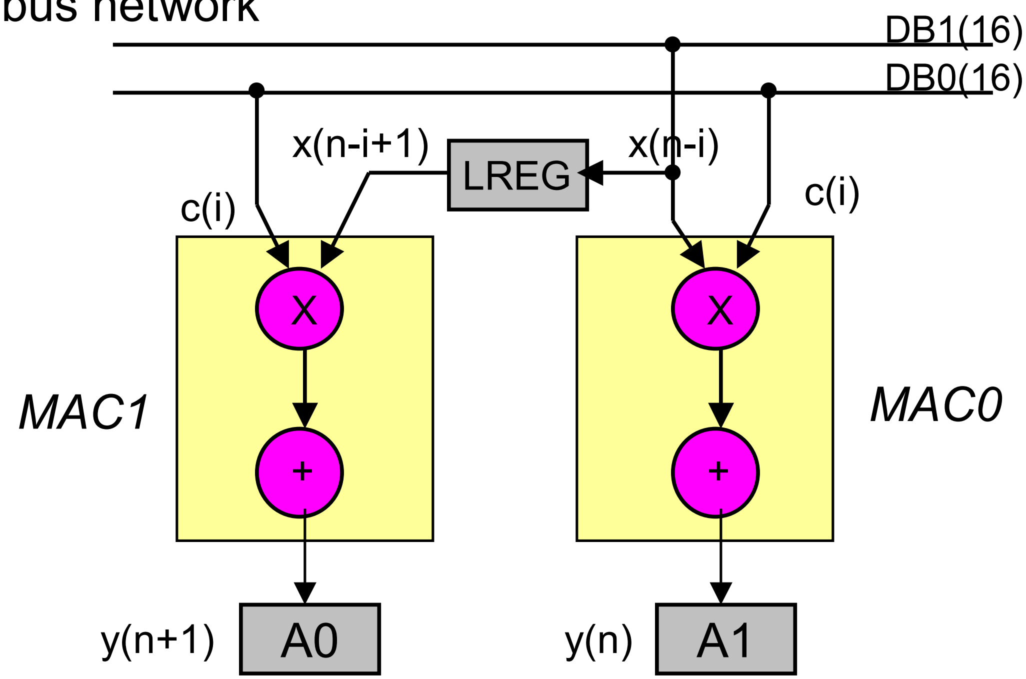
```
y(0) = c(0)x(0) + c(1)x(-1) + c(2)x(-2) + . . .  + c(N-1)x(1-N);
y(1) = c(0)x(1) + c(1)x(0) + c(2)x(-1) + . . .  + c(N-1)x(2-N);
y(2) = c(0)x(2) + c(1)x(1) + c(2)x(0) + . . .  + c(N-1)x(3-N);
y(3) = c(0)x(3) + c(1)x(2) + c(2)x(1) + . . .  + c(N-1)x(4-N);
```

- Two MAC units with dedicated bus network

2 reads for 2 MAC's

(or 2 reads for N MAC's)

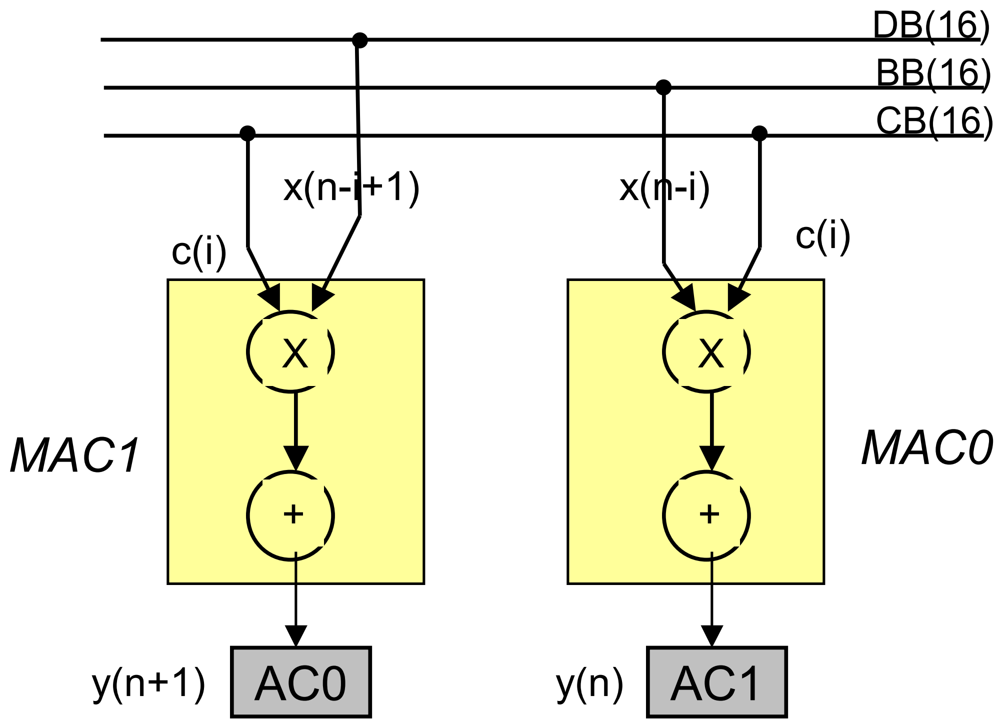
ONLY 2 busses!



FIR on TI C55x (2000)

- FIR filter: two outputs in parallel with 3 busses

```
y(0) = c(0)x(0) + c(1)x(-1) + c(2)x(-2) + . . .  + c(N-1)x(1-N);
y(1) = c(0)x(1) + c(1)x(0) + c(2)x(-1) + . . .  + c(N-1)x(2-N);
y(2) = c(0)x(2) + c(1)x(1) + c(2)x(0) + . . .  + c(N-1)x(3-N);
y(3) = c(0)x(3) + c(1)x(2) + c(2)x(1) + . . .  + c(N-1)x(4-N);
```



Energy comparison

- Total energy for one output sample:

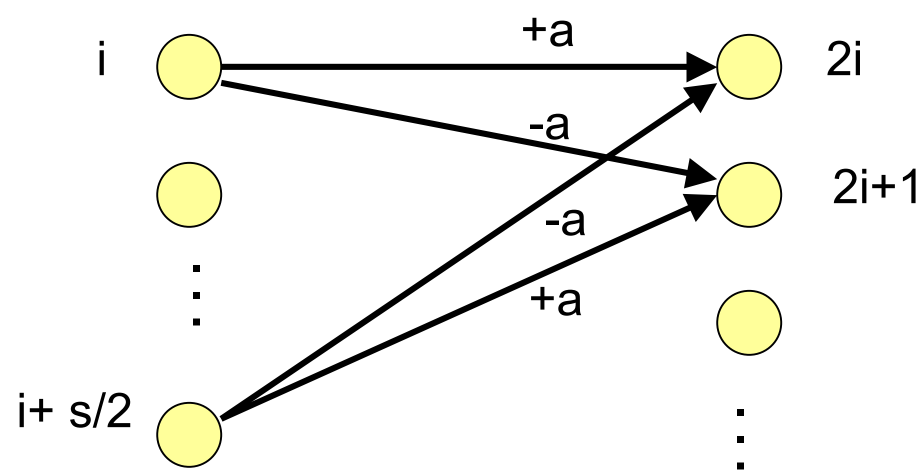
Energy	Single MAC	Dual MAC	Dual MAC 3 busses	Dual MAC with REG
No. of MAC operations	N	N	N	N
No of Memory reads	2N	2N	1.5N	N
No of Instruction Cycles	N	N/2	N/2	N/2

Adaptation of the datapath: MAC, DMAC
Adaptation of the **memory architecture and bus network**
Adaptation of the instruction set

Example 2: Viterbi

- Viterbi butterfly

  i = state index
  s = # of states = 2^{k-1}
  w = decoding window



- Basic equations:

$$d(2i) = \min\{\, d(i) + a,\ d(i + s/2) - a \,\}$$
$$d(2i + 1) = \min\{\, d(i) - a,\ d(i + s/2) + a \,\}$$

- Key operation: *Add-Compare-Select (ACS)*

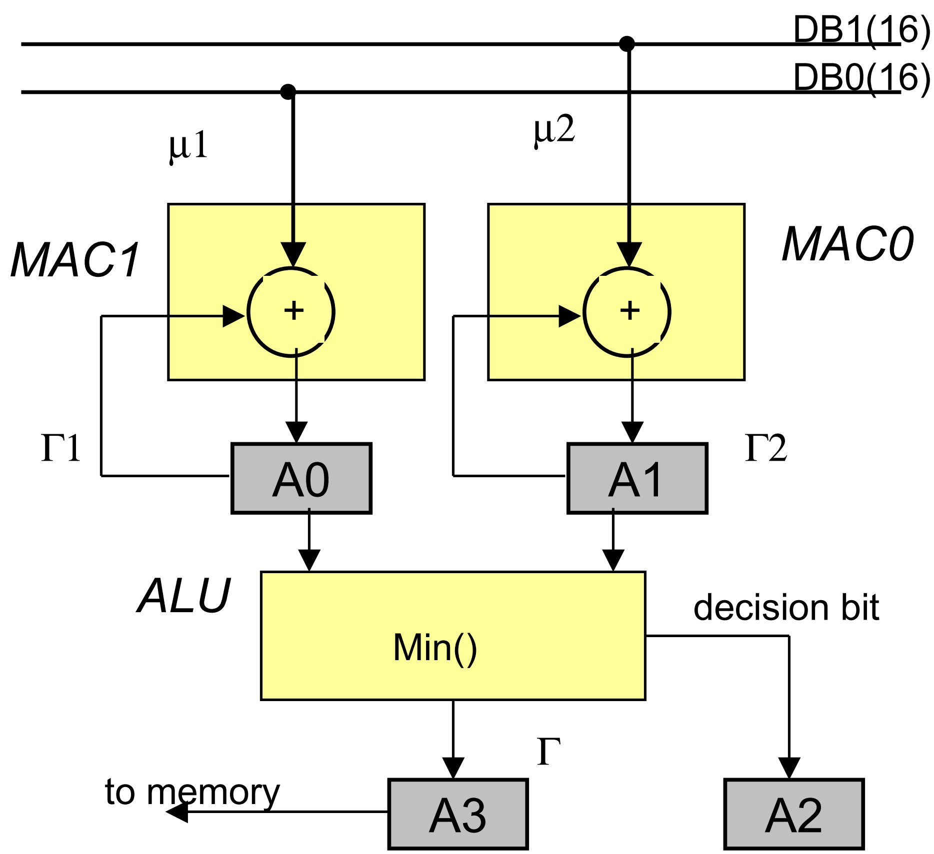
- IS-95: $k = 9$, 256 states, $w = 192$, means 2^8 x 192 x (cycles for one ACS)

Viterbi on Lode

- Two MAC units & ALU: Add-Compare-Select

$$\Gamma = \min [(\Gamma 1 + \mu 1), (\Gamma 2 + \mu 2)]$$

- DMAC operates as dual add/subtract unit
- ALU finds minimum
- Shortest distance saved
- Path indicator saved
- 4 cycles / butterfly

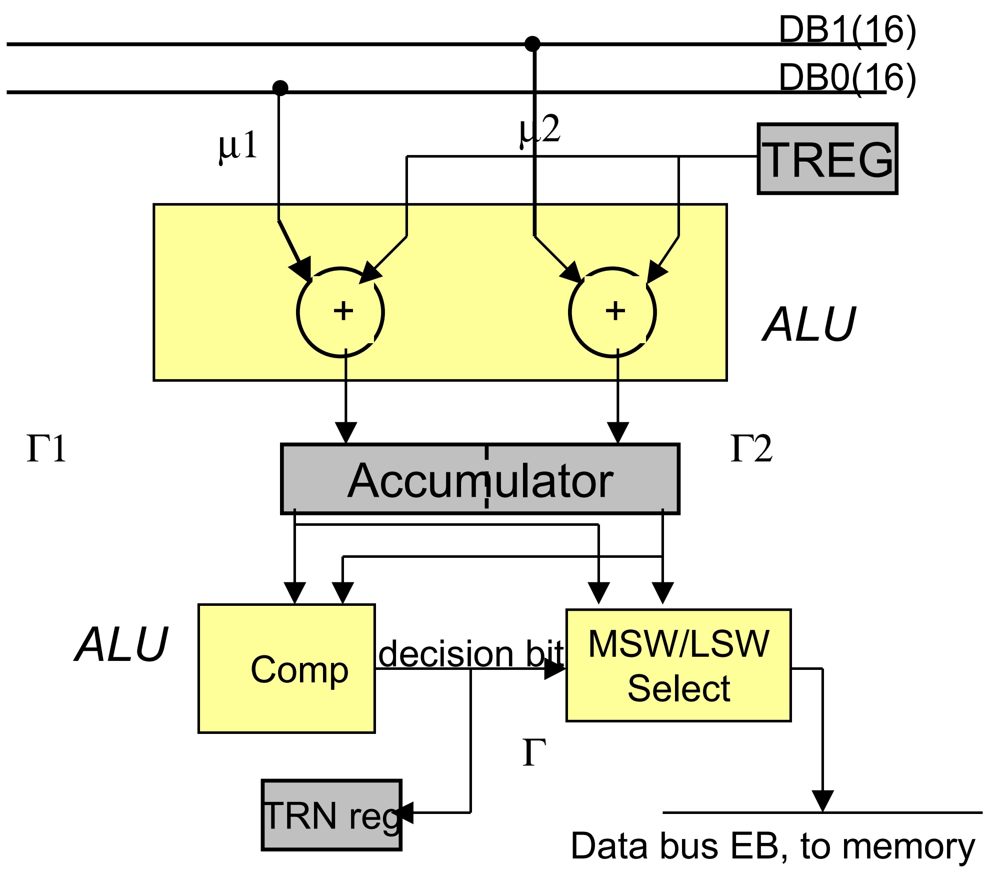


Viterbi on TIC54x

- ALU and CSSU: CMPS instruction

$$\Gamma = \min [(\Gamma 1 + \mu 1), (\Gamma 2 + \mu 2)]$$

- ALU splits in 16 bit halves
- ACC splits in half
- Shortest distance saved
- CSSU compares halves
- Path indicator saved
- 4 cycles / butterfly



Source: TI Application Report, Viterbi Decoding in the TMS320C54x family, document SPRA071

Viterbi on TI ‘C6x

3-cycle 2-ACS Inner-Loop

```
LOOP:
  [b1]       b      .s1    LOOP
||[b1]       sub    .s2    b1,1,b1
||[!a2]      sth    .d1    b12,*+a6[8]
||[!a2]      add    .d2    b0,b14,b14
||           cmpgt  .l1    a11,a10,a1
||           cmpgt  .l2    b11,b10,b0
||           mpy    .m1x   1,b5,a4

  [a2]       sub    .s1    a2,1,a2
||[!a2]      sth    .d1    a12,*a6++
||[a1]       add    .s2    2,b0,b0
||[b0]       mpy    .m2    1,b11,b12
||           mpy    .m1    1,a10,a12
||           sub    .l2x   a7,b5,b10
||           ldh    .d2    *++b9,b5

             shl    .s2    b14,2,b14
||[a1]       mpy    .m1    1,a11,a12
||           add    .s1    a7,a4,a10
||           sub    .l1x   b13,a4,a11
||           add    .l2    b13,b5,b11
||           mpy    .m2    1,b10,b12
||           ldh    .d2    *b4++[2],a7
||           ldh    .d1    *a5++[2],b13
; end of LOOP
```

x 8

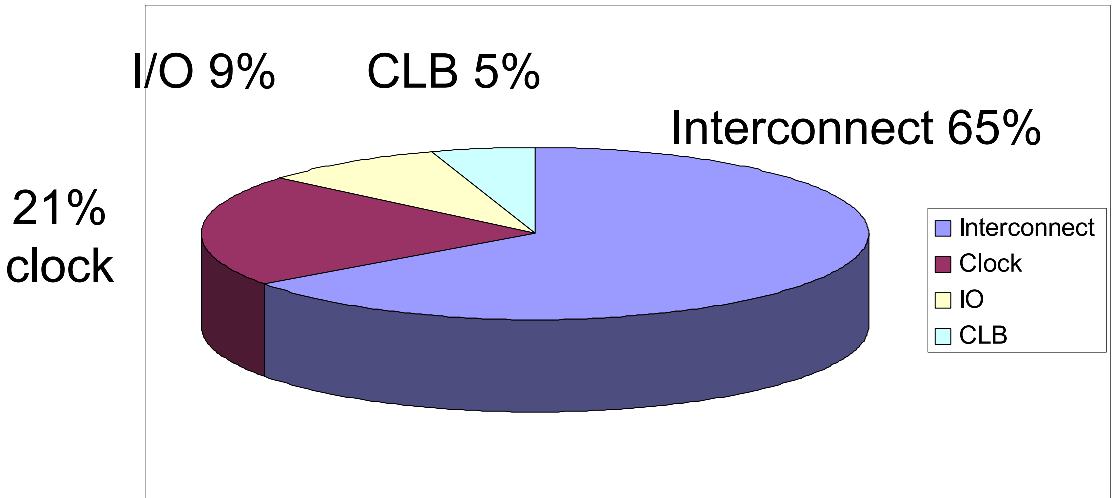
- 16-state Viterbi decoder for GSM
  from TI site: ftp://ftp.ti.com/pub/tms320bbs/c62xfiles/vitgsm.asm
  - 3 cycles per butterfly
  - 32 cycles per GSM timeslot (8 butterflies)
  - **MPY instructions used to move data!**

```
||[b0]      mpy     .m2     1,b11,b12
||          mpy     .m1     1,a10,a12
```


[ISSCC 2000 Tutorial]

Reconfigurable Interconnect

- Is **expensive!**
- FPGA power break –down :



[E. Kusse, ISLPED 1998]

Reconfigurable interconnect

“We don’t know what the customer will run”

World’s Fastest Logic & Routing

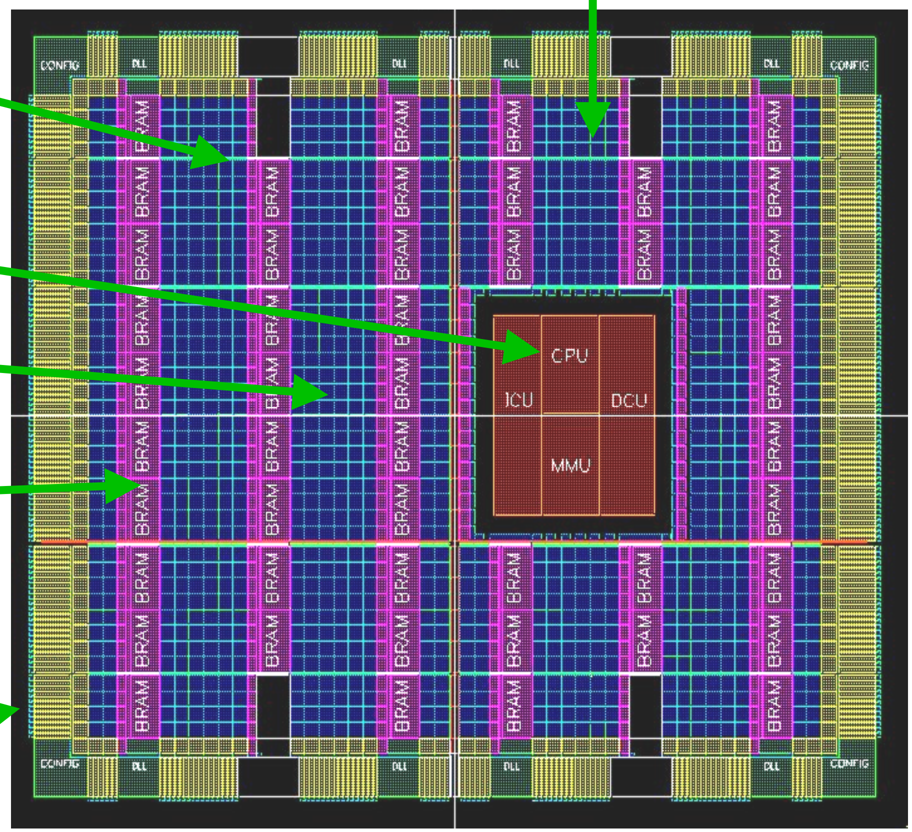
Conexant 3.125Gb Serial

IBM PowerPC® RISC CPU

XtremeDSP™

Synchronous Dual-Port RAM

SelectIO-Itra™ SystemIO™ & XCITE ™



[courtesy: Xilinx]

FPGA 2002: dynamic power: 60% routing, 16% logic, 14% clock!

Xilinx – Off Chip

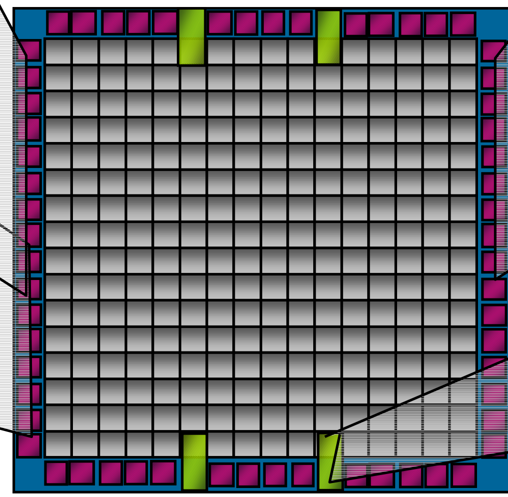
“We don’t know what the customer will run”

System-Synchronous Parallel

- PCI-66
- PCI-X133

Source-Synchronous Parallel

- RapidIO
- HyperTransport
- SPI-4
- POS-PHY3
- POS-PHY4
- FlexBus3
- FlexBus4
- XGMII



SystemIO

Advanced Memory Interfaces

- ZBT SSRAM
- DDR SDRAM
- QDR SSRAM
- CAM

High-Speed Serial

- Multi-Gigabit Backplanes
- Infiniband™
- Gb Ethernet
- 10 Gb Ethernet (XAUI)
- Fibre Channel

Approach: RF-Interconnect

Conventional interconnect:

- space division
- time division

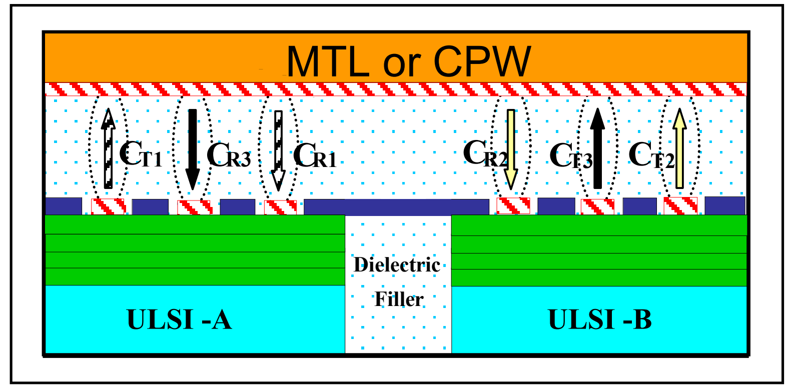
New interconnect:

- code division (CDMA)
- frequency division (FDMA)
- any combination of the above

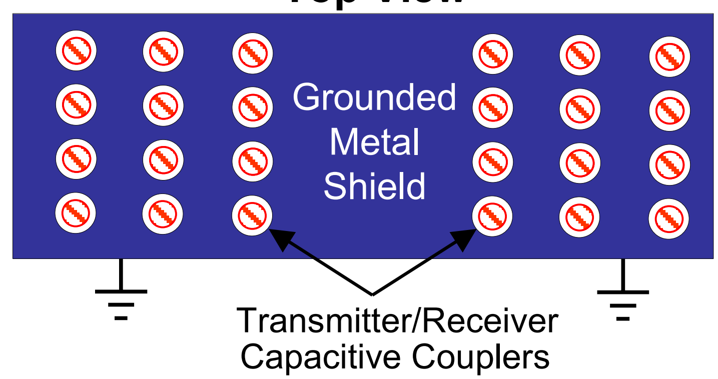
Reconfigurable!

- “Low Loss, dispersion-free, ultra-high data rate (100Gbps/channel & 20Tbps/chip)”

Miniature LAN in MCM

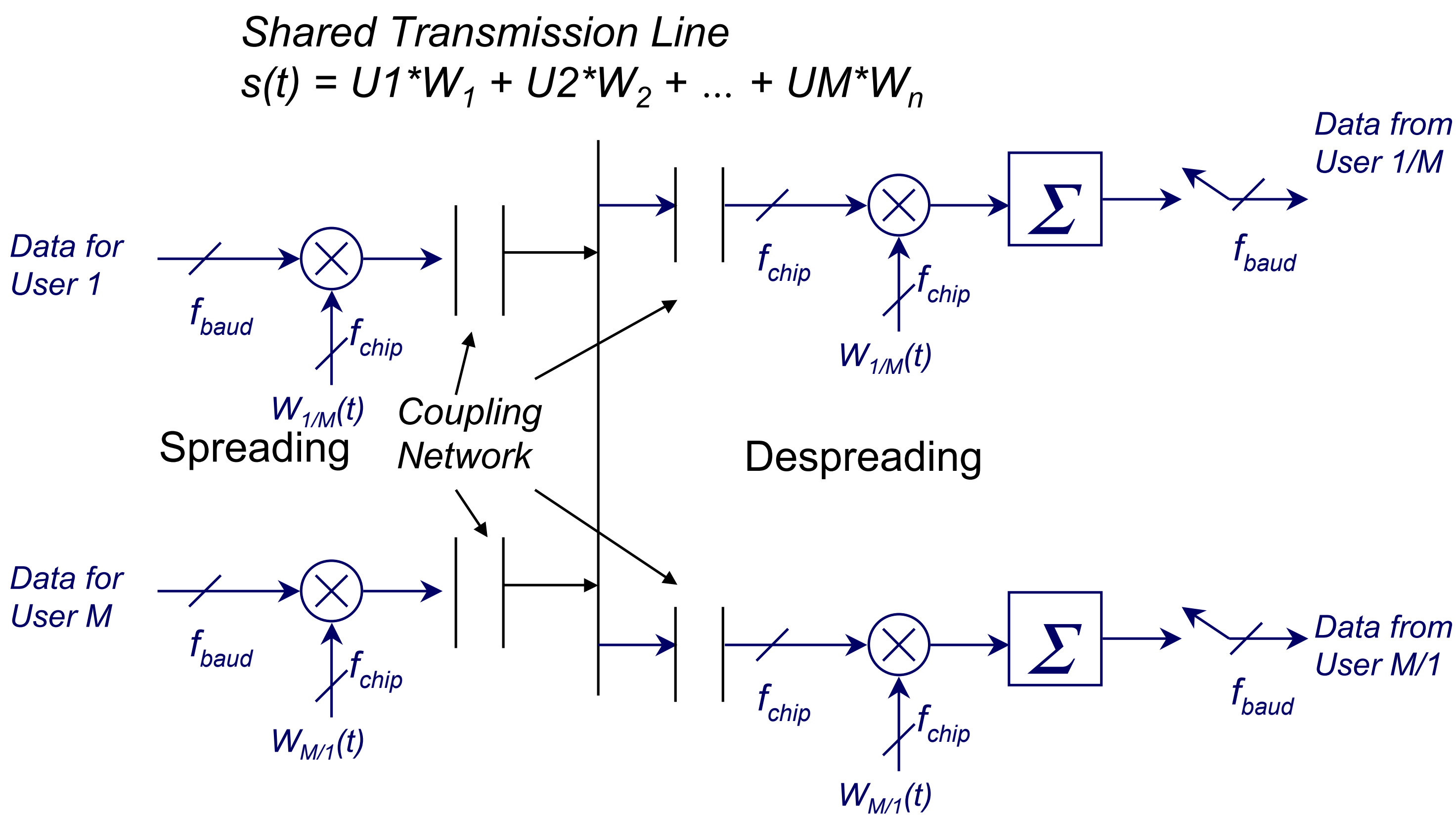


Top View



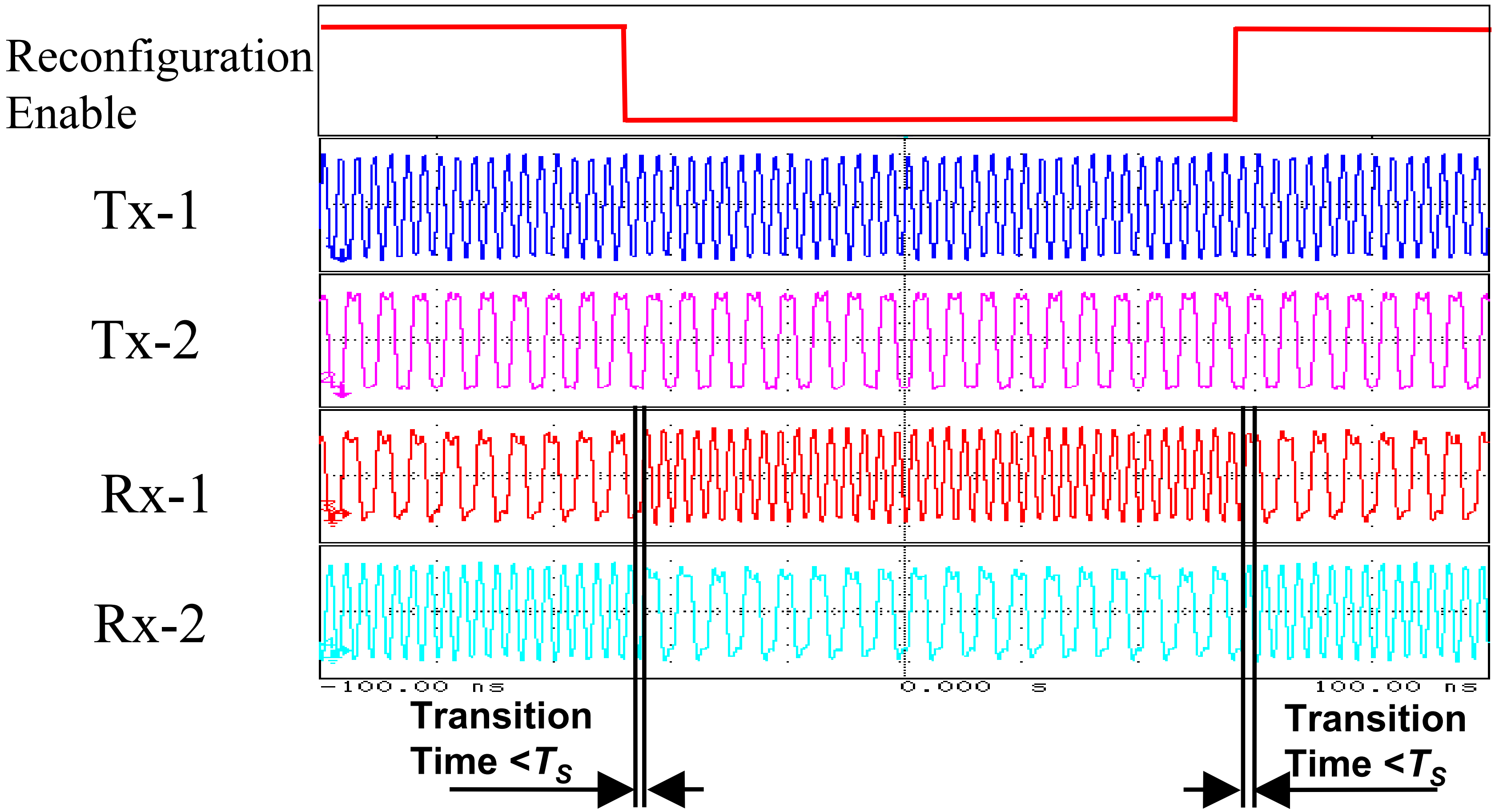
[M.F. Chang, Proc. IEEE 2001]

CDMA-Interconnect

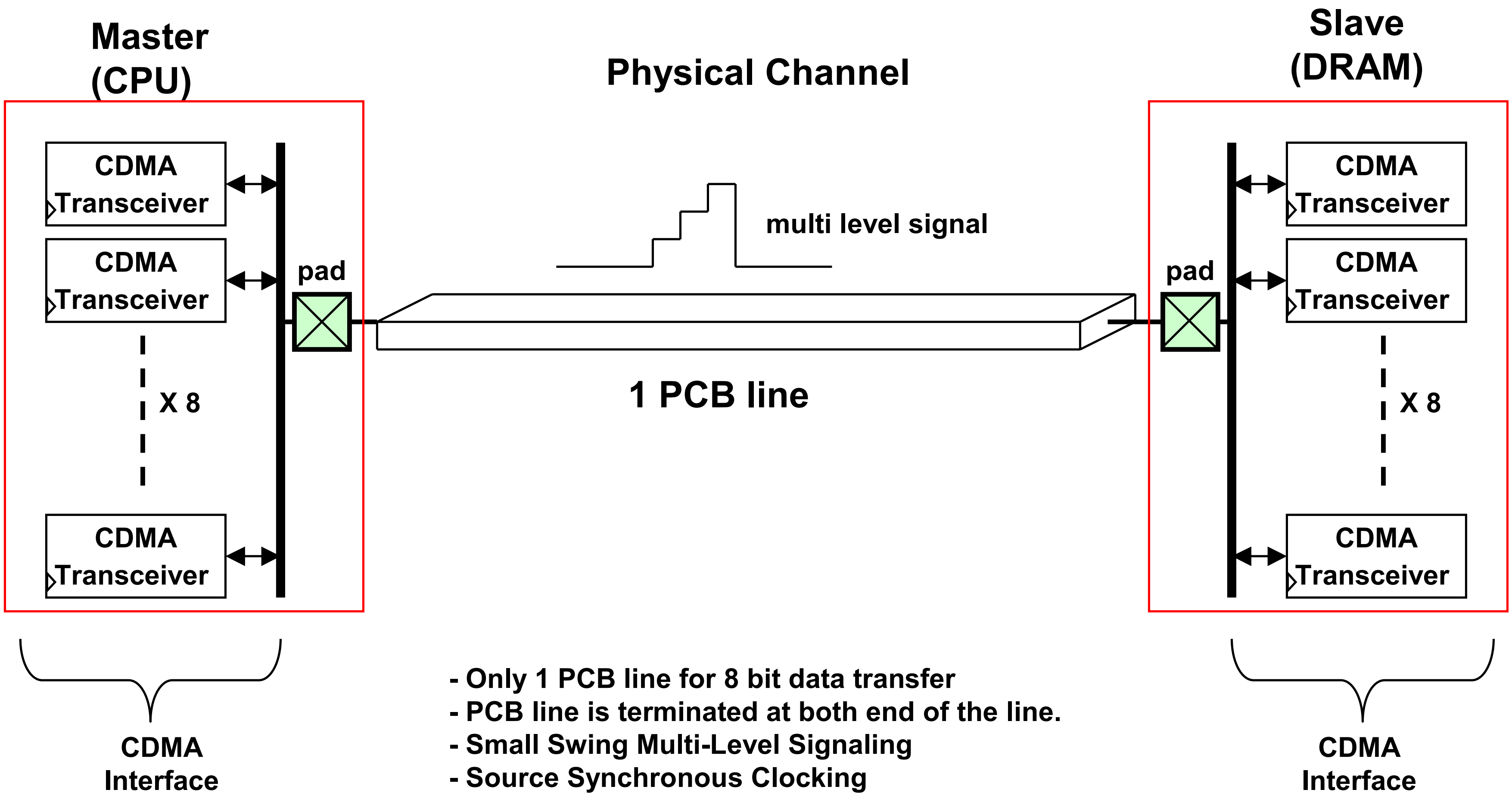


[M.F. Chang, Proc. IEEE 2001]

Channel Reconfigration at f_{clk}=2.8GHz



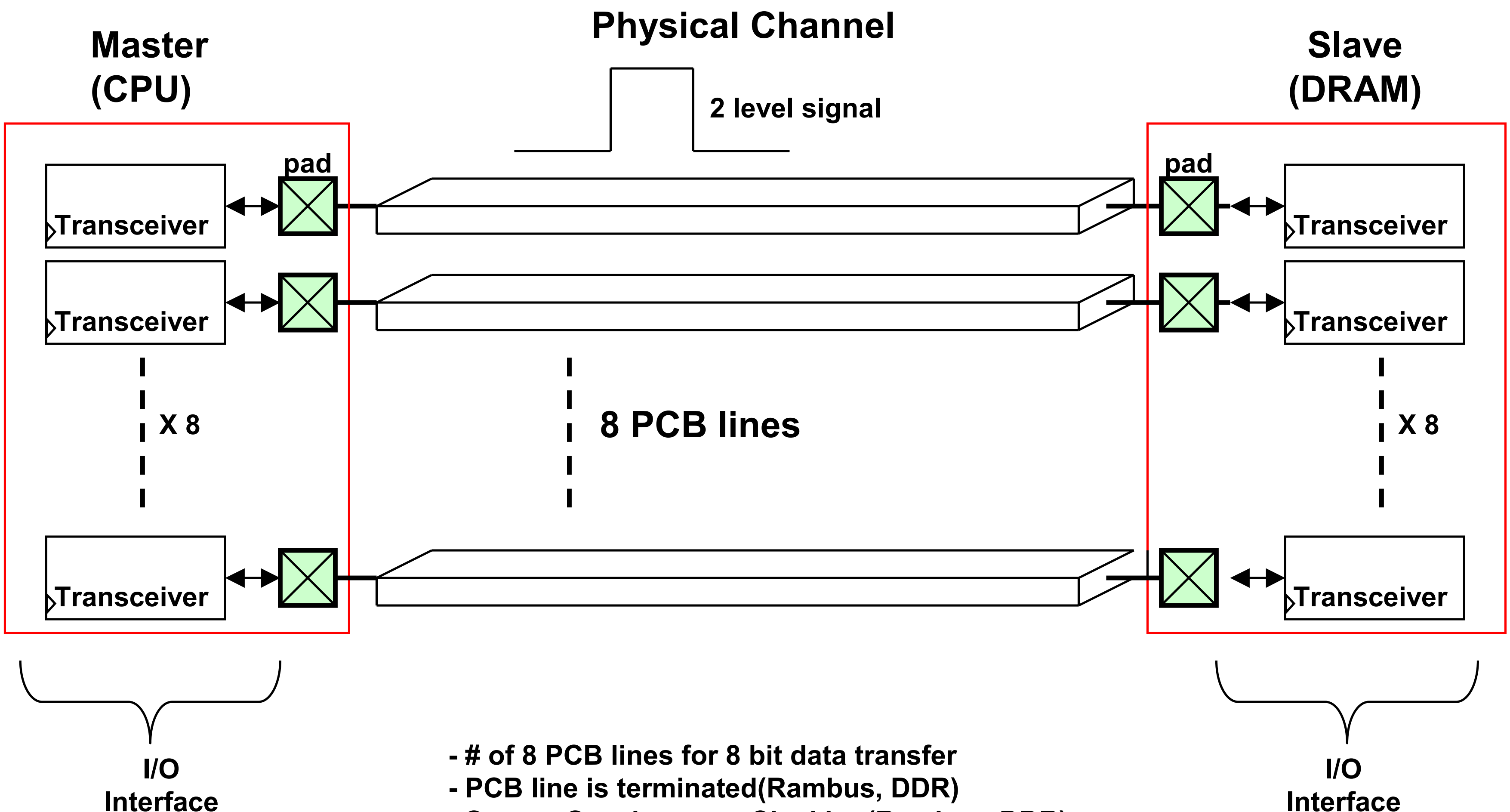
RF – Off chip: CDMA based SDRAM bus



- Only 1 PCB line for 8 bit data transfer
- PCB line is terminated at both end of the line.
- Small Swing Multi-Level Signaling
- Source Synchronous Clocking
  (Clock and Data Transfer together)

[courtesy: Jongsun Kim, UCLA]

SDRAM interface: conventional bus



- # of 8 PCB lines for 8 bit data transfer
- PCB line is terminated(Rambus, DDR)
- Source Synchronous Clocking(Rambus, DDR)

[courtesy: Jongsun Kim, UCLA]

Comparison

	# of Parallel high-speed Data Channels	# of Parallel High-speed Address Channels	# of Parallel High-speed Clock Channels	Total # of Parallel High-speed Channels	Min. # of Shielding Channels for Data & Add.	Data Rate (Mbps)	Total Data Bandwidth
SDRAM (PC-133)	*64*	12	1	*77*	-	133	*1.1 Gbyte/s*
Rambus DRAM	*16*	*8*	2 (differential)	*28*	*24*	400 * 2	*1.6 Gbyte/s*
DDR(SSTL-2)	*64*	12	8 (data strobe)	*84*	-	133 * 2	*2.1 Gbyte/s*
CDMA DRAM	*2*	*1*	2 (differential)	*7*	*3*	1600 * 4	*1.6 Gbyte/s*

- RDRAM use 2 differential (CTM/CTMB,CFM/CFMB) clock lines for source synchronous clocking
- DDR use 8 data strobe (DQS) lines for source synchronous clocking
- CDMA DRAM use 2 differential clock lines for source synchronous clocking
- Both SDRAM and DDR DRAM has some additional command lines which are not listed above.

[courtesy: Jongsun Kim, UCLA]

CDMA DRAM Bandwidth options UCLA

	Rambus DRAM	CDMA DRAM case 1	CDMA DRAM case 2	CDMA DRAM case 3	CDMA DRAM case 4
DRAM Core Freq.	100MHz	100MHz	100MHz	100MHz	100MHz
DRAM Core Internal Bus	8bit*C=128b	8bit*N*C=128b	8bit*N*C=1024b	8bit*N*C=128b	8*N*C=512b
DRAM Interface Freq.	400 MHz * 2	400 MHz * 2	400MHz * 2	300MHz * 2	300MHz * 2
DLL Freq.	400 MHz	1.6 GHz	1.6 GHz	1.2 GHz	1.2 GHz
I/O Interface Freq.	400 MHz * 2	1.6 GHz * 2	1.6 GHz * 2	1.2 GHz * 2	1.2 GHz * 2
# of Data PCB Channels	*C=16*	*C=2*	*C=16*	*C=4*	*C=16*
# of CDMA Users	-	N=8	N=8	N=4	N=4
Total Bandwidth	*1.6 Gbyte/s*	*1.6 Gbyte/s*	*12.8 Gbyte/s*	*1.6 Gbyte/s*	*6.4 Gbyte/s*

- Rambus DRAM use dual edge of 400MHz DLL clock for I/O Interface
- CDMA DRAM use dual edge of 1.6GHz DLL clock for I/O CDMA Interface

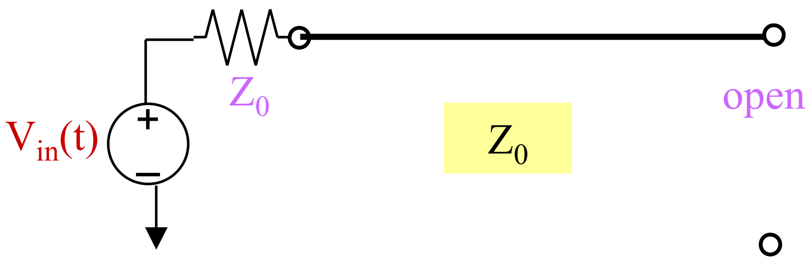
[courtesy: Jongsun Kim, UCLA]

“System Level Interconnect Prediction”

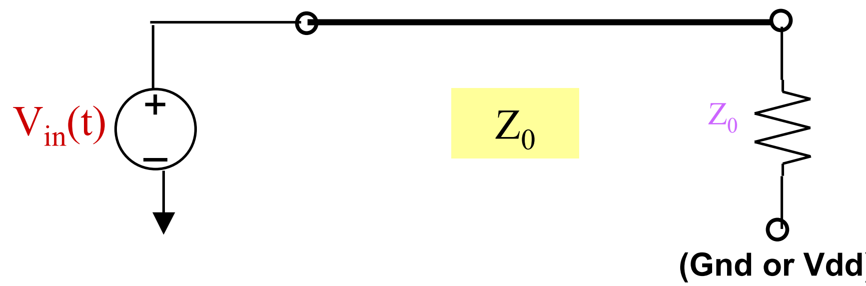
- Andrew’s question: what is the message for SLIP?
  - Max capacity for min energy
- Example: EE215B (Advanced Digital Integrated Circuits Class)
  - 10” lossy transmission line, 1 Gbs
  - Study termination and coupling noise
  - Part II: Inter symbol interference,
    - max throughput for given length (10”)
    - Max length for given throughput (1Gbs)
  - Part III: low swing
    - Based on C.Svensson, JSSC July 2001, “Optimum Voltage Swing for On-Chip and Off-Chip Interconnect

Basic Termination Methods

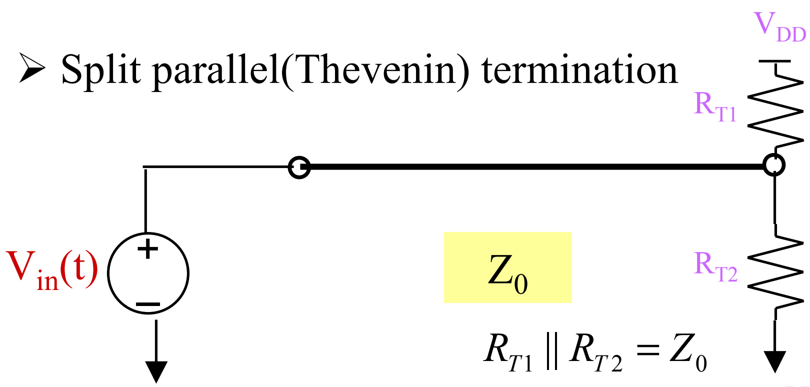
➢ Series termination



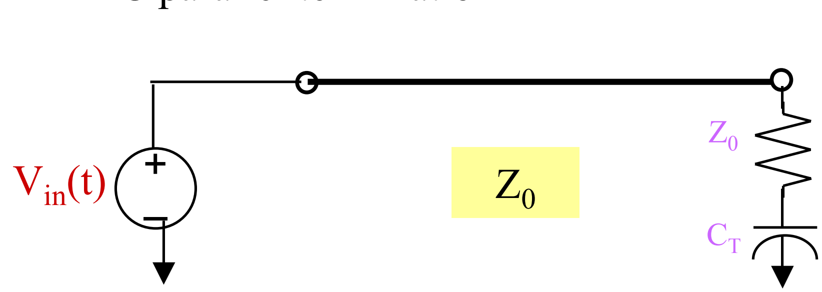
➢ Parallel termination



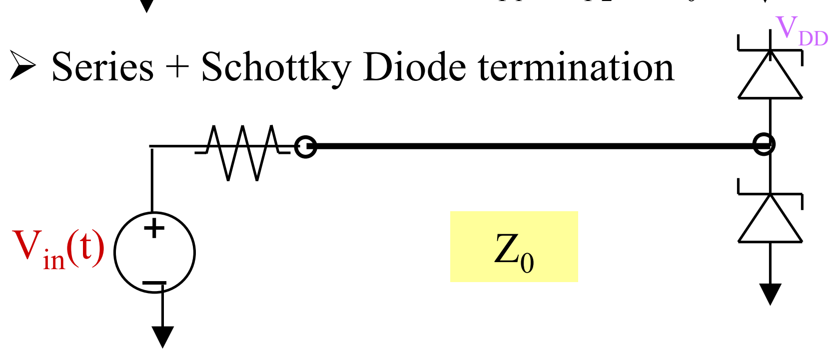
➢ Split parallel(Thevenin) termination



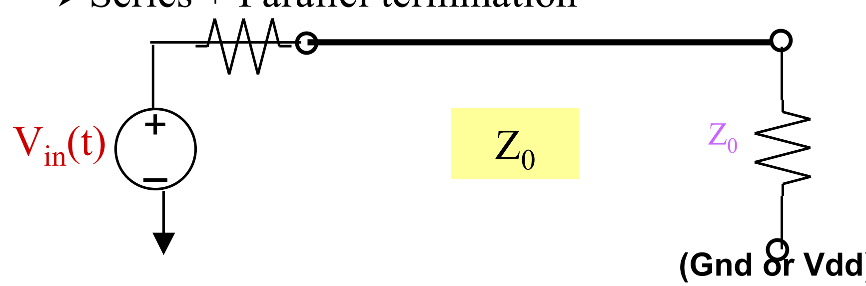
➢ AC parallel termination



➢ Series + Schottky Diode termination

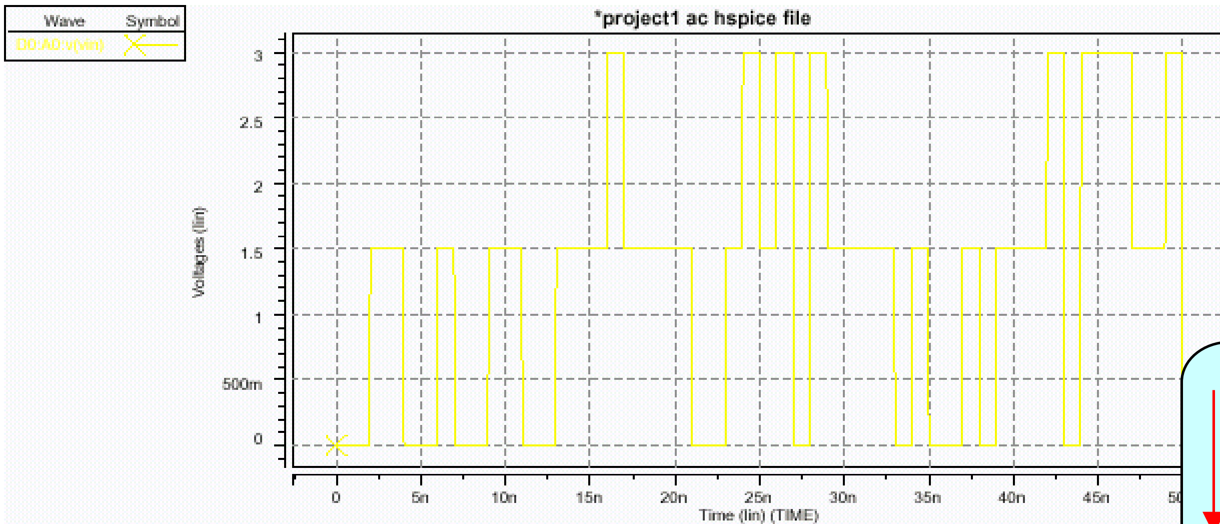


➢ Series + Parallel termination

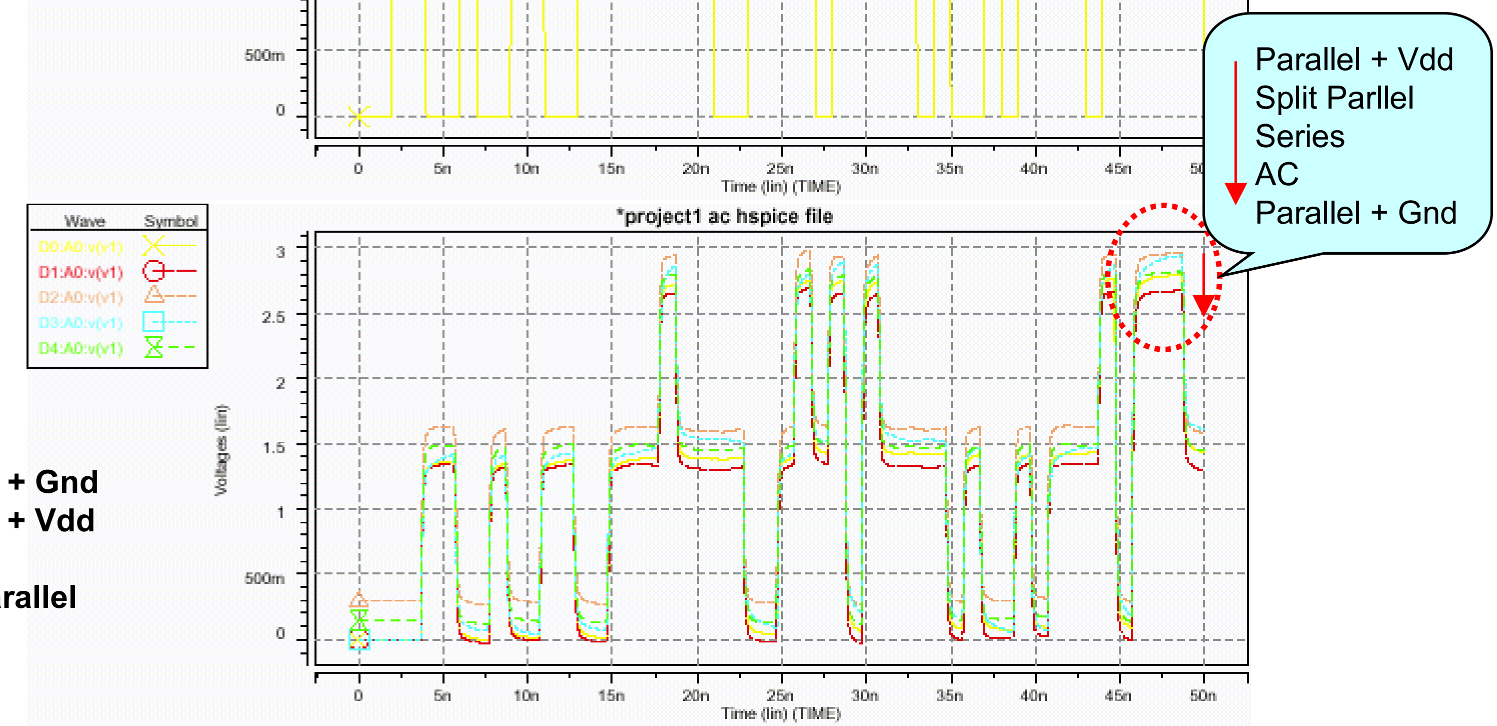


Received node voltage probing I

Input

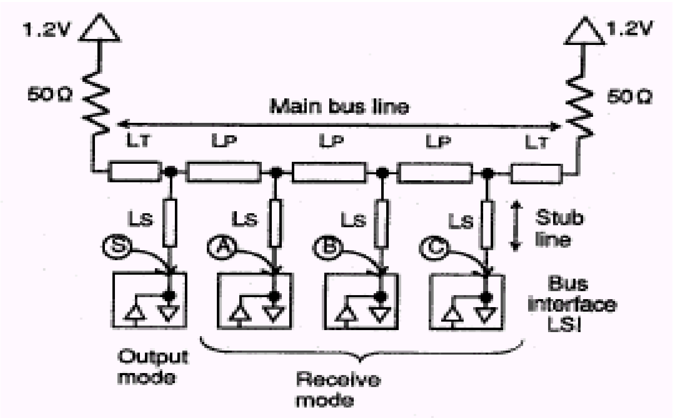


D0 : AC
D1 : Parallel + Gnd
D2 : Parallel + Vdd
D3 : Series
D4 : Split Parallel

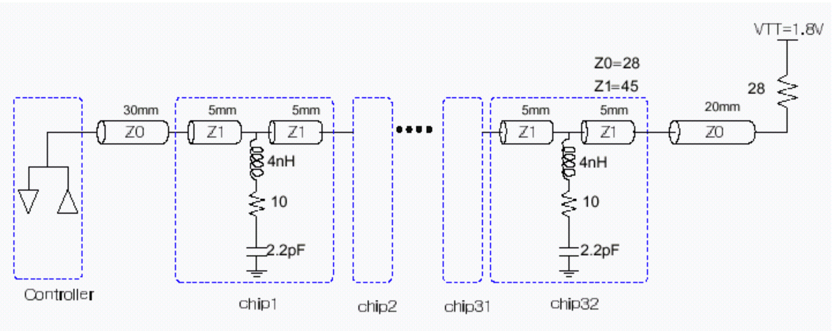


Advanced Termination Methods

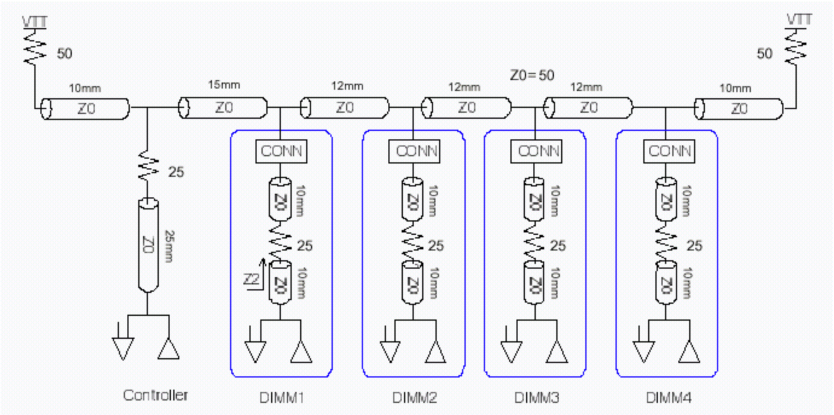
- GTL



- RSL



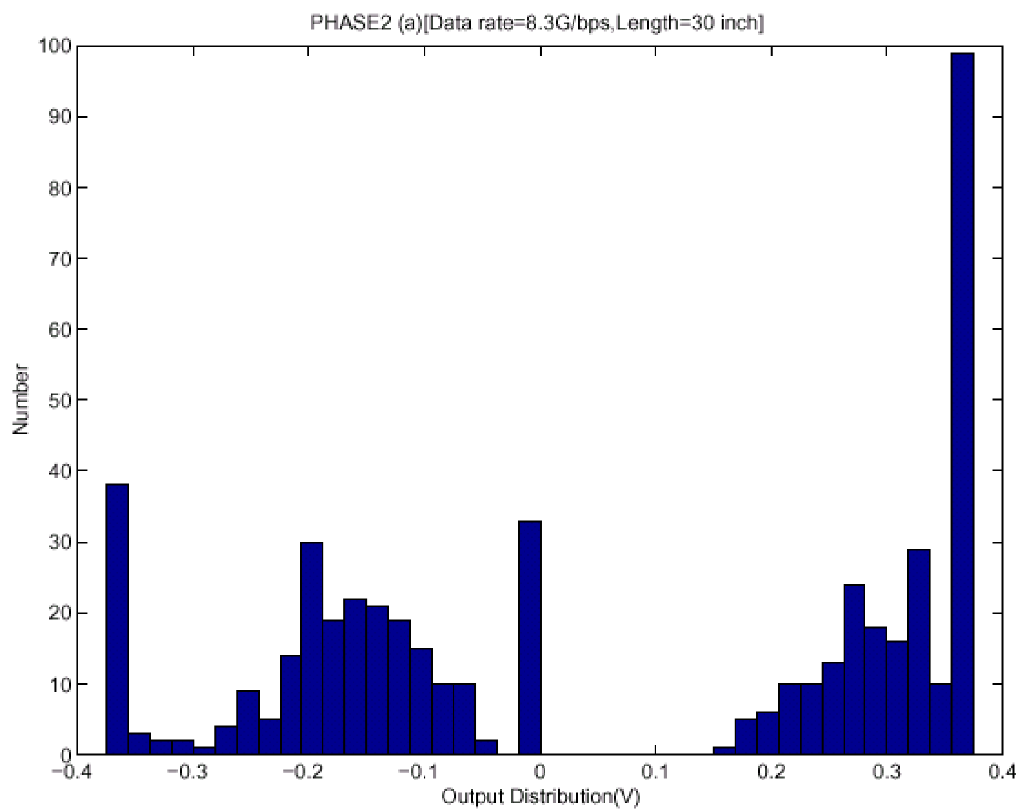
- SSTL-2

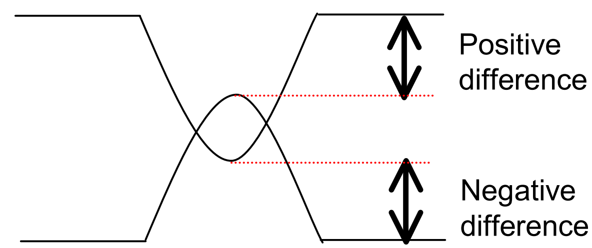


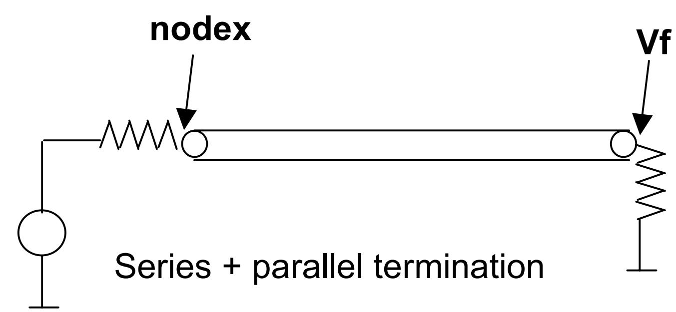
Inter Symbol Interference

(1) Length=30 inches and increase data-rate until 8.3 Gbps

(consider ideal signal level (no ISI) : High=750mV, Low=0mV)

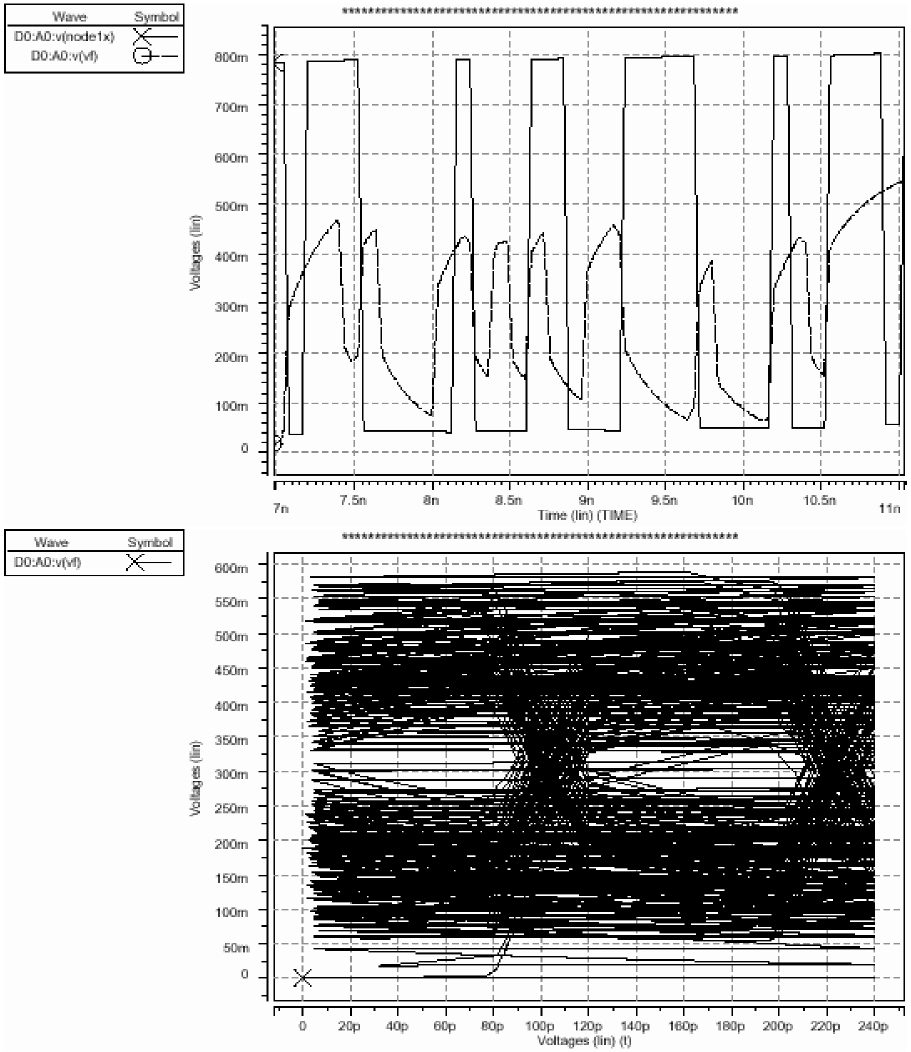




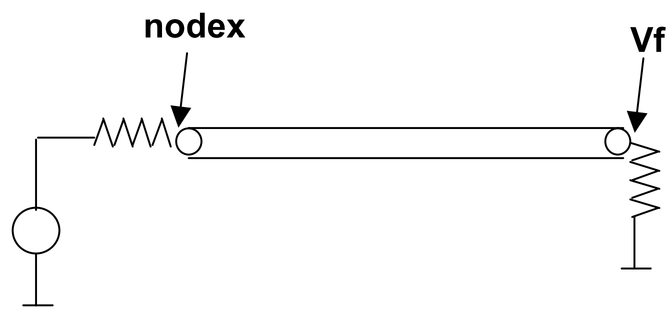


Series + parallel termination

ISI: Eye Diagram

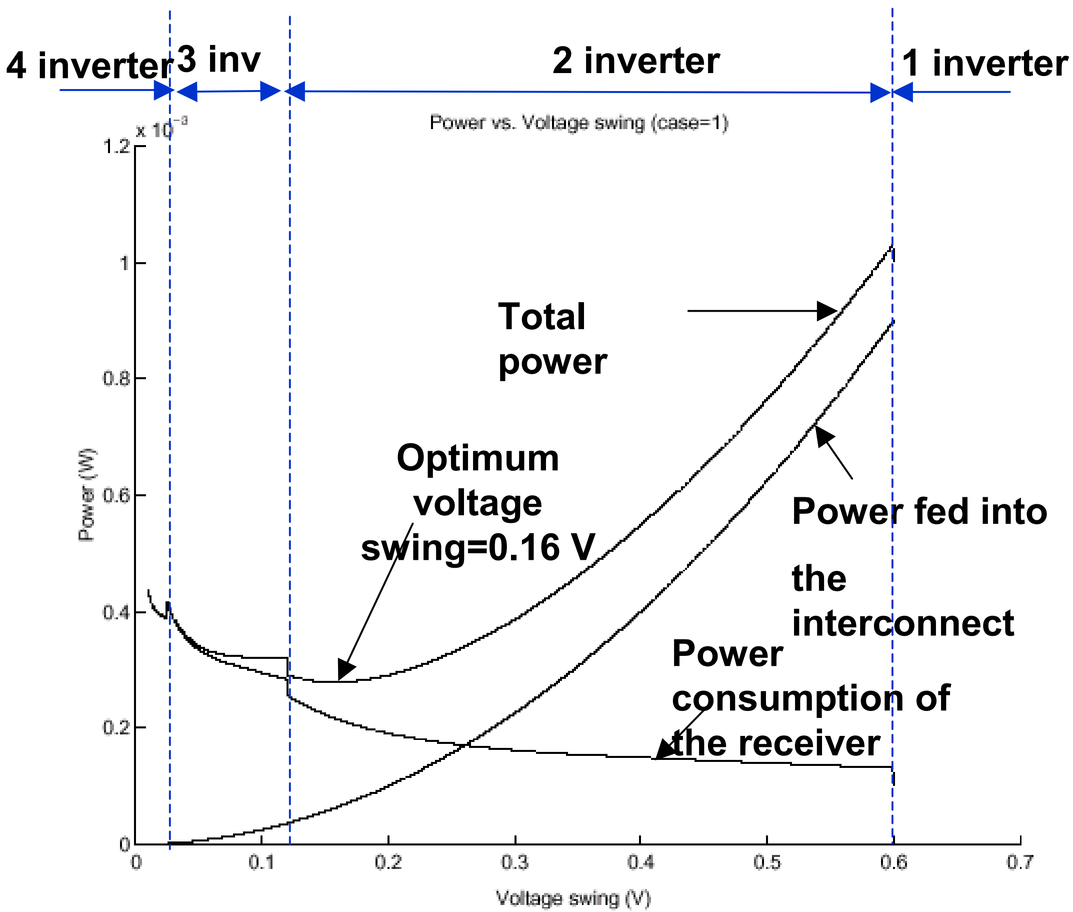


Length=30 inches and increase data-rate until 8.3 Gbps



Next step:
multilevel signals

Optimal voltage swing

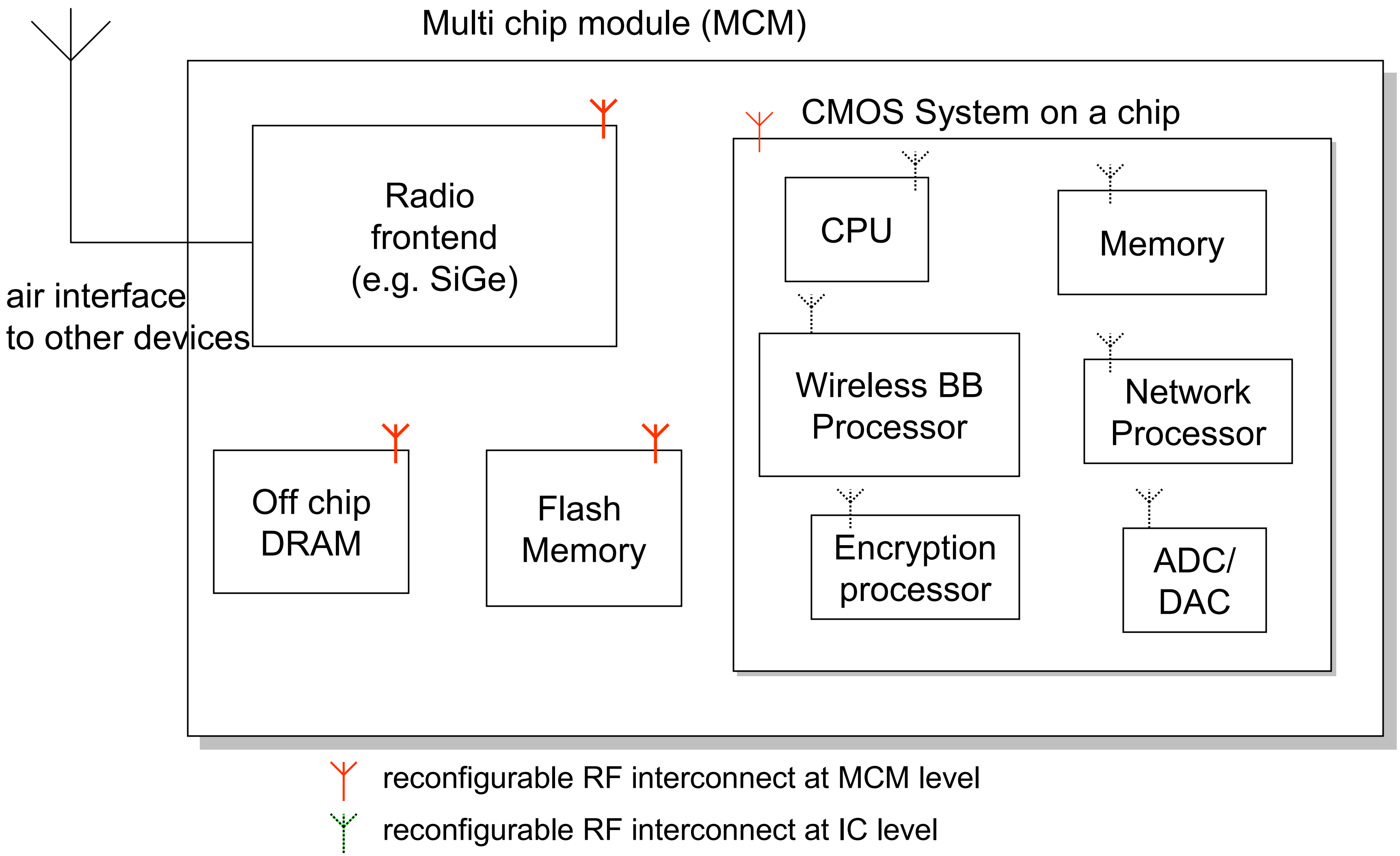


Trade-off
- power of transmitter and receiver
- power into line

10Gbs: min swing 0.16V

Ingrid Verbauwhede

Next generation system on a chip



Ingrid Verbauwhede

Conclusion

- Challenge 1: energy – flexibility tradeoff
- Goal: max capacity for min energy
- Approach: domain specific processing - reconfiguration hierarchy
- Challenge 2: reconfigurable interconnect
- Approach at the system architecture level
- Approach at “circuit” level: RF interconnect
- Many problems to be addressed