High-Throughput Iterative Decoders

Borivoje Nikolić
Electrical Engineering and Computer Sciences
University of California, Berkeley

bora@eecs.berkeley.edu

19 June 2003

Outline

- **Motivation**
- **Power-performance tradeoffs**
  - Viterbi decoder examples
- **Iterative decoder architectures**
  - Turbo decoders
  - Parallel LDPC decoders
  - Serial LDPC decoders
- **Conclusion**

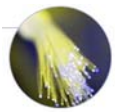
What Does 3db Buy?

3dB coding gain can:

- Double DSL subscriber radius
- Double battery life of cell phone
- Double number of channels on satellite TV
- Twice as many bytes on a hard drive
- Reduce antenna diameter by 30%

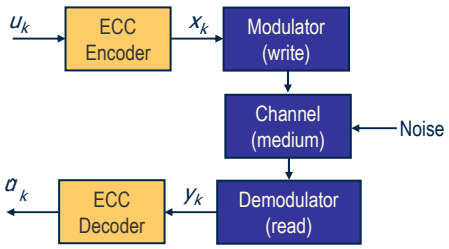






How To Achieve 3dB Gain?

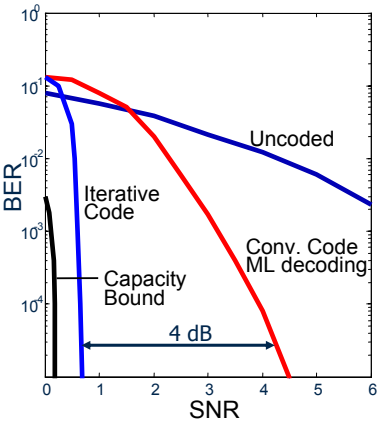
- Error correction codes (Forward error correction)
  - Convolutional codes with Viterbi decoding
  - Reed-Solomon codes
  - Turbo, low-density parity-check, and other iterative codes
- Modulation techniques
  - 64 QAM, CDMA. OFDM
  - Multiple antenna, MIMO systems
- Channel detection techniques
  - Partial response signaling
  - Adaptive equalization
  - Decision feedback equalization (DFE)



50 Years of Information Theory

Year	Rate ½ Code	SNR Required (BER = 10^{-5})
1948	SHANNON	0dB
1967	(255,123) BCH	5.4dB
1977	Convolutional Code	4.5dB
1990	Convolutional Code + RS	3.9dB
1993	*Iterative Turbo Code	0.7dB
2001	**Iterative Low Density Parity Check (LDPC) Code	0.0045dB

SNR vs. BER for rate 1/2 codes



Issues:

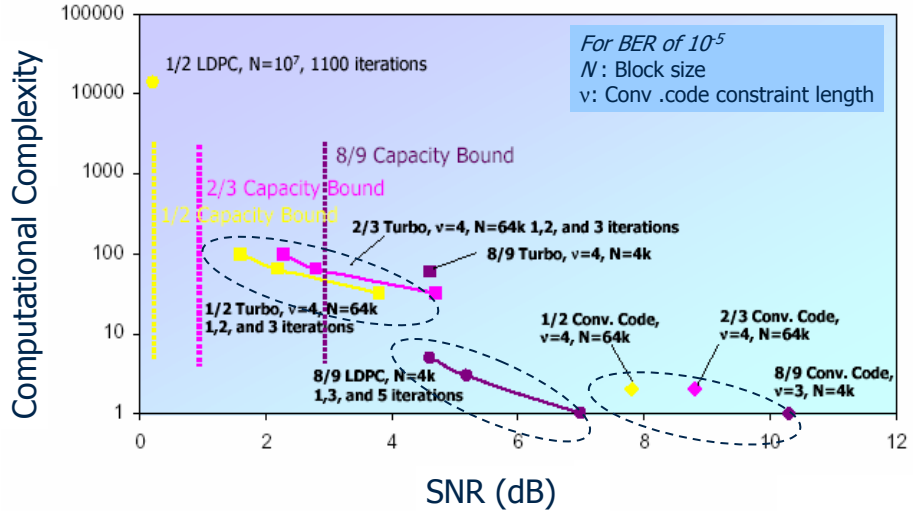
- Decoder implementation, complexity
- Design and emulation
- Timing recovery, etc

*[C. Berrou et. al., ICC 1993]
** [Chung, Forney, Richardson, Urbanke, Trans. Inform. Theory 2001]

Complexity Issues

- **Most systems are cost (area) and power constrained**
  - And have certain throughput/latency requirements
- **Iterative decoders are by more than an order of magnitude more complex than Viterbi decoders**
  - The code that is 0.0245dB away from Shannon bound has 10^6 block length, 1000 iterations, irregular construction
- **To achieve practical systems:**
  - Lower complexity codes
  - Scaling of CMOS

Relative Complexities



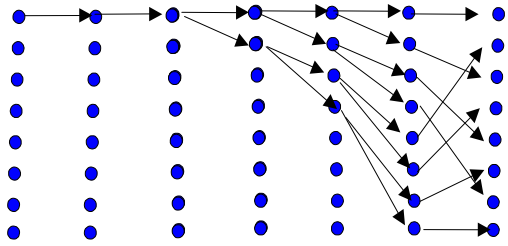
Difference in complexity ~ 5 orders of magnitude

Power-Performance-Area Tradeoffs

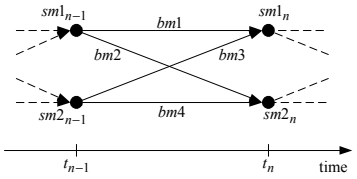
- **Microarchitectural transformations**
- **Non-recursive systems**
  - Parallelism, pipelining, retiming, transposition, …
- **Recursive systems**
  - Loop unrolling, retiming, …
- **Circuit level optimizations**
  - Gate sizing
  - Supply, threshold optimizations

Example: Viterbi Decoder

- Most likely path through trellis is traced



- 2-state example:



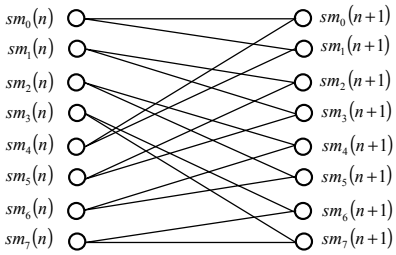
$$sm1_n = \min\left(sm1_{n-1} + bm1,\ sm2_{n-1} + bm3\right)$$

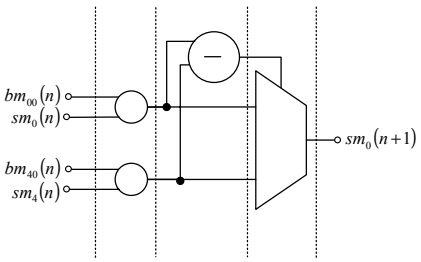
$$sm2_n = \min\left(sm1_{n-1} + bm2,\ sm2_{n-1} + bm4\right)$$

Add Add

Select Compare

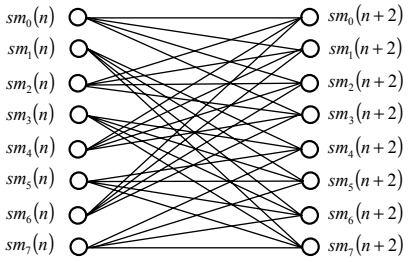
Example: Viterbi Decoder

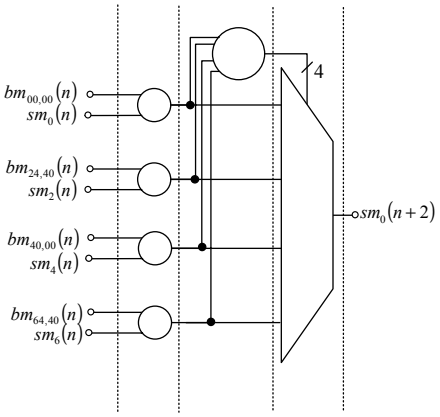




- Critical path: Add-compare-select (ACS) recursion

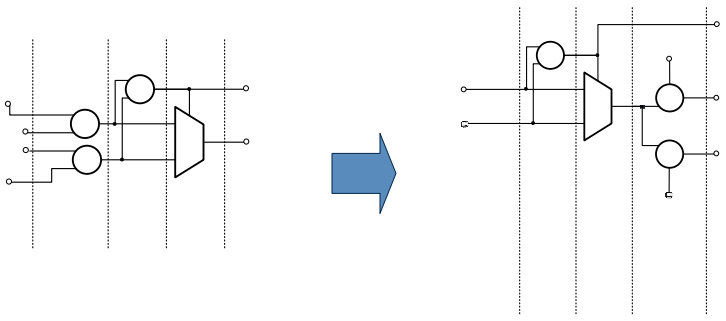
Viterbi Decoder: Loop Unrolling





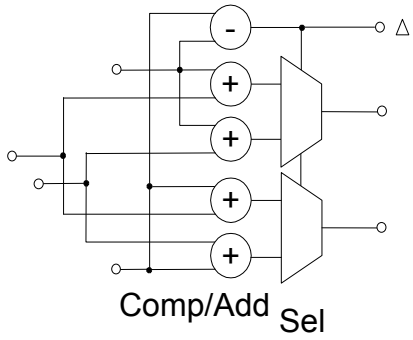
- Transforms radix-2 trellis into radix-4 trellis
- Min operation is implemented through six parallel comparisons
- Add-compare-select in 2 symbol times (with larger fanouts)

Viterbi Decoder: Retiming



- Retiming usually doesn't help in recursive algorithms

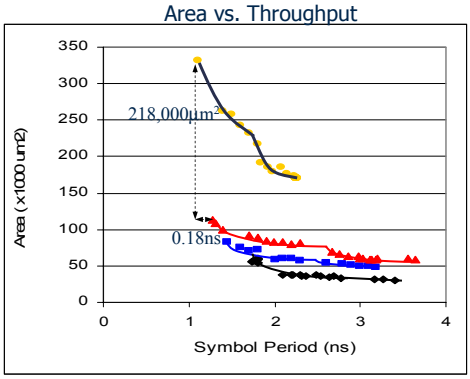
Viterbi Decoder: Compare/Add-Select



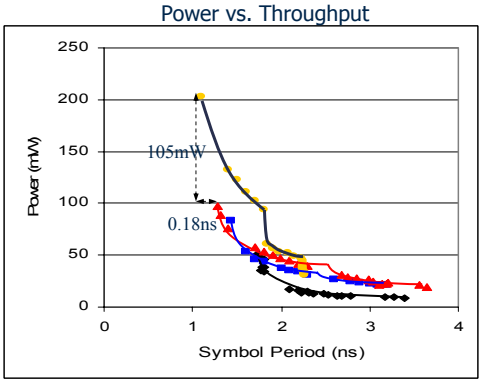
- Another retiming pushes adds back through the select
- Add and compare performed in paralell

ACS Comparison

Area vs. Throughput



Power vs. Throughput



Synthesis results using 0.18μm general-purpose standard cell library

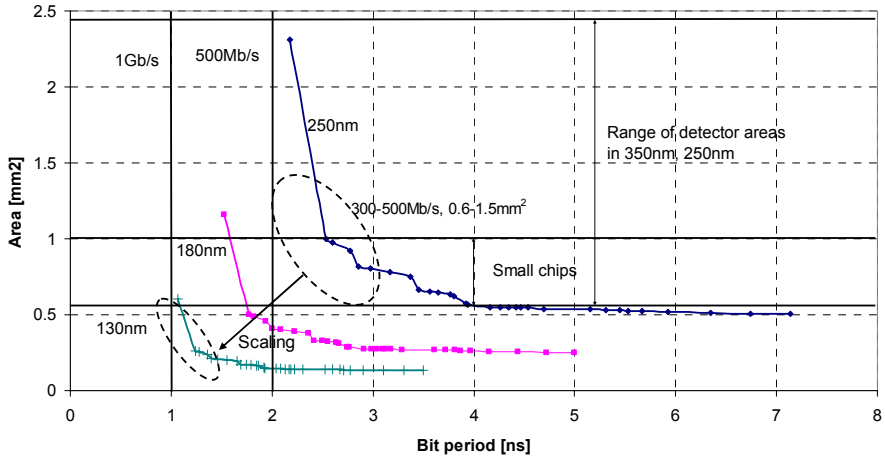
◆ACS: Traditional add-compare-select

■ Concurrent ACS: Concurrent add and comparison *[S. Sridharan and Carley, JSSC 2000]*

▲CSA: Retimed Compare-Select-Add *[Fettweis, et. al., Globecom 1995] [Lee and Sonntag, Globecom 2000]*

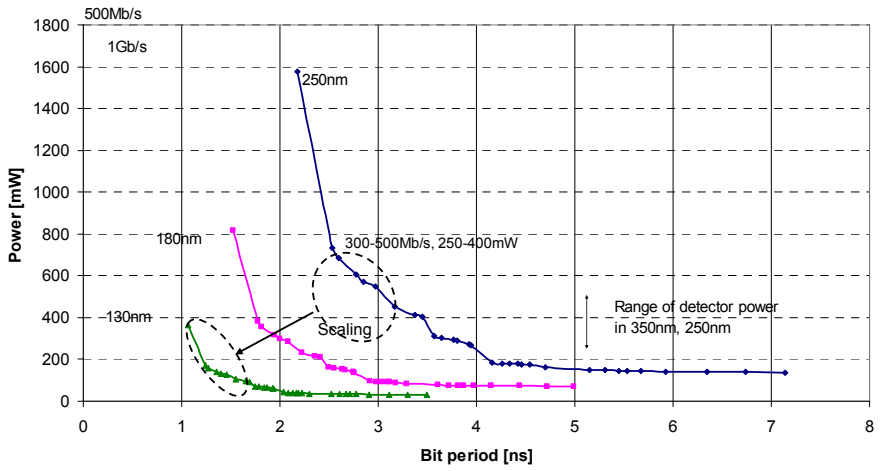
●R4 ACS: Radix-4 ACS *[Yeung and Rabaey, ISSCC 1995] [Black and Meng, JSSC 1992]*

Case Study: Disk Drive Read Channels



- 8-16-state Viterbi decoders were barely reaching 500Mb/s in 0.25µm
- In 0.13µm they would be 2x faster and 4x smaller

Case Study: Disk Drive Read Channels



- Power is reduced by 4x, too
- Allows integration of larger (e.g. 32-state detectors)

Iterative Decoders

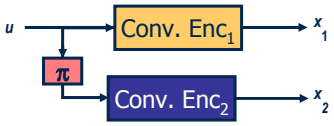
- **Can they be made practical to fit:**
  - Optical receivers
  - Disk drive read channels
  - Satellite (DVB) receivers
  - Digital subscriber line (xDSL) modems ?
- **Would a practical high-throughput iterative decoder be feasible in 130nm? 90nm?**

Iterative Decoders

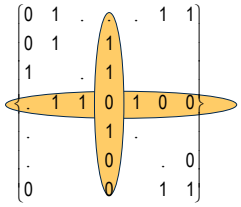
- **Two classes: turbo and message passing decoders**
- **Turbo decoders**
  - Soft-input-soft-output building blocks
  - Maximum-a-posteriori (MAP) decoder implementing BCJR algorithm
  - Soft-output Viterbi decoder (SOVA)
  - Interleavers
- **Message-passing decoders**
  - Low-density-parity check (LDPC) codes
  - Turbo-product codes are decoded similarly + interleaver

Examples of Iterative Codes

- **Turbo codes**
  - Concatenated convolutional codes
    - Parallel *[Berrrou93]*
    - Serial *[Benedetto96]*



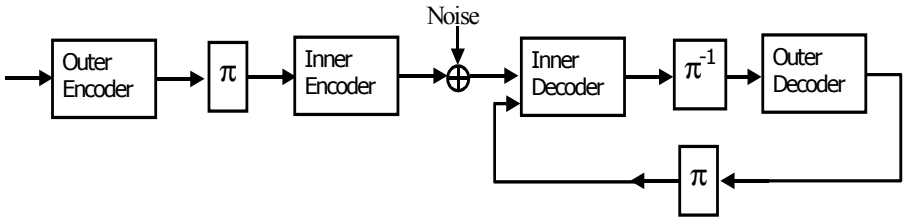
- **Turbo product codes**
  - Transmitted bits fill 2D matrix
  - Rows/columns constrained by
    - Hamming
    - BCH *[Pyndiah99]*



- **Low density parity check codes** ***[Gallager63]***
  - Sparse binary parity check matrix, ***H***
  - Nullspace of H forms set of codewords, $\tilde{x}$
  - Simple even parity checksums and modulo-2 arithmetic
    - Density evolution *[Richardson01]*
    - Finite field constructions *[Lin00]*
    - Rammanujan graphs *[Rosenthal00]*

$$H\tilde{x} = 0$$

Iterative Decoders: Turbo serial

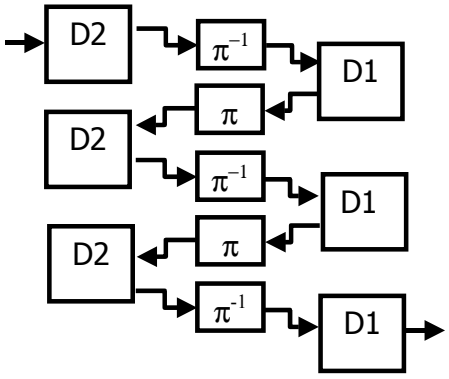


π: Pseudo Random Interleaver

Turbo decoding for partial response channels:

- Inner Code: Trellis (Convolutional) Code based on Partial Response Channel
- Outer Code: Convolutional or LDPC Outer Code
  - *T. Souvignier, et. al., "Turbo Decoding for PR4; parallel vs. serial concatenation," ICC 1999*

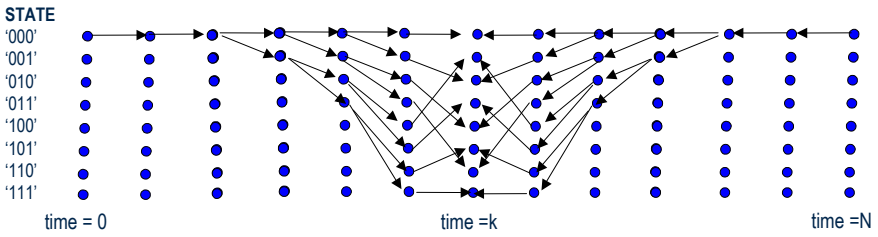
Iterative Decoders: Unrolling



Unrolled decoder employs multiple pipelined SISO stages to achieve desired throughput rates (> 1Gbps)

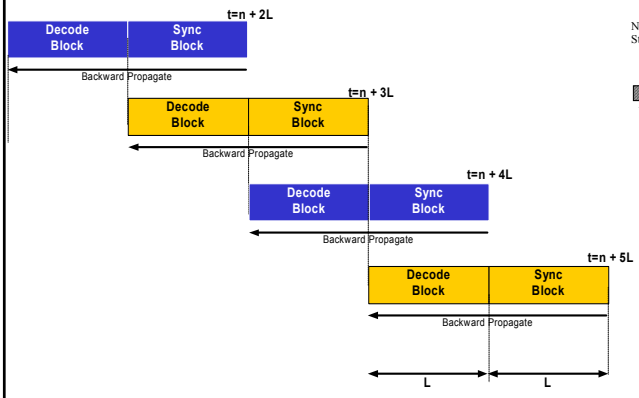
- *G. Masera, et. al., "VLSI Architectures for Turbo Codes", IEEE Transactions On VLSI Systems, Vol. 7, No. 3, Sep. 1999.*

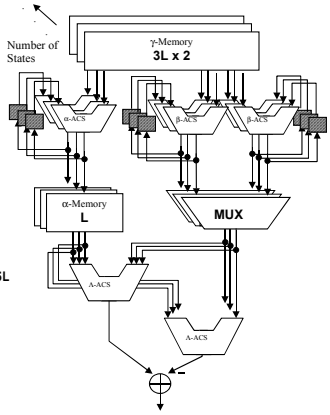
SISO Blocks: BCJR Algorithm



- Implements Maximum A-Posteriori Decoding
- Bi-directional trellis path propagation
- Path metric is a probability measure of the particular ending state
- Long latencies for full path propagation
- Large memory requirements
- **Solution: Windowed Methods**

SISO Blocks: Windowed BCJR





- One forward sliding window.
- Two alternating backward propagating windows

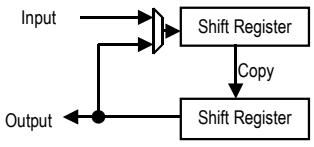
[A. Viterbi, J. on Sel. Areas in Comm., 1998]

[E. Yeo, etc. , Trans. Magnetics, 2001]

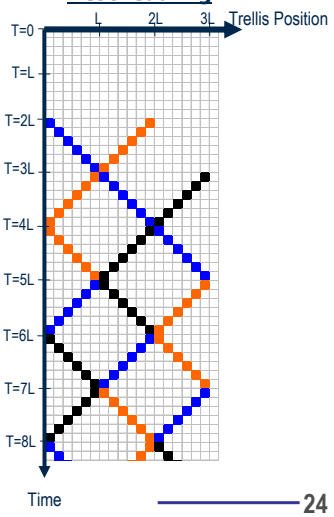
Memory Access in MAP Decoder

- Direct windowed approach: complex memory access.
  - Rescheduling
  - Partition memory into 3 sections
- Memories to be implemented with fast shift registers.

[E. Yeo, et. al., TMRC 2000]
[A. Worm, et. al., VLSI Design 2001]
[Bougard, et. al., ISSCC 2003]

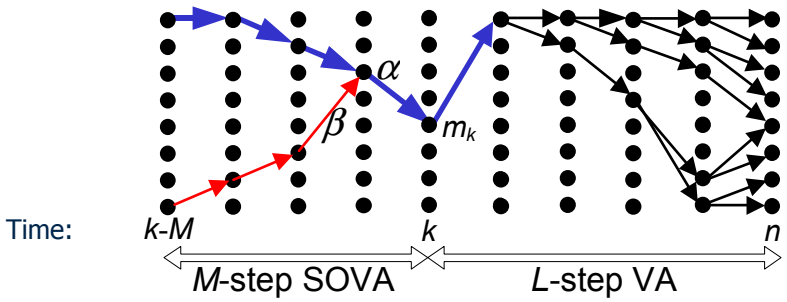


Memory Partitioning & Rescheduling



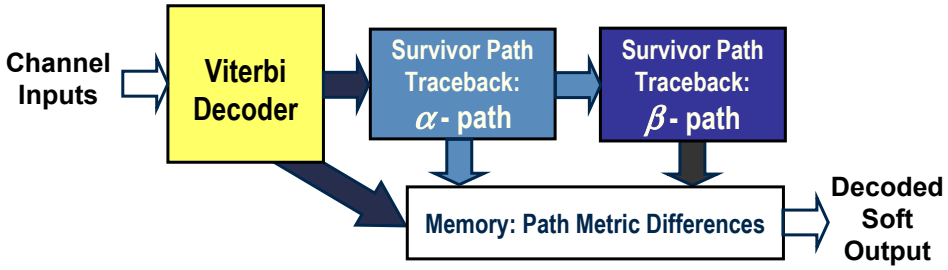
Forward Window
Backward Window 1
Backward Window 2

Soft Output Viterbi Algorithm (SOVA)



- Determines the two most likely path (α and β) …
- …that provide complementary bit decisions
- Soft output = Difference in path metric of α and β

SOVA Decoder Architecture



- Register exchange implementation of survivor path traceback
- Throughput limited by Viterbi decoder
- Cache realized with synthesized flip-flops

Die Micrograph

1.8V, 0.18μm CMOS with 6 metal layers

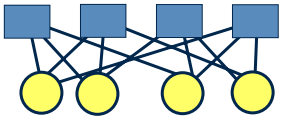
Decoder	Speed	Power	Trans. Count
8-state Octal(11,13)	500Mb/s	400mW	174k
8-state EPR4 channel	500Mb/s	395mW	164k

Standard cell design with customized clock tree for 500MHz CLK

[E. Yeo, et.al. ESSCIRC 2002.]

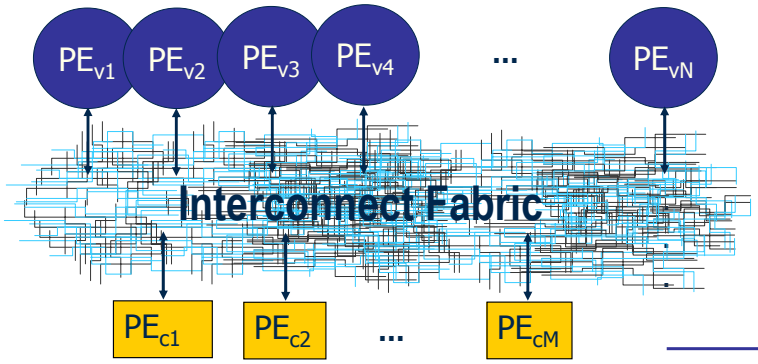
…more details to appear in IEEE Journal of Solid-State Circuits, July 2003

Parallel LDPC Decoder Architecture



- **Fully parallel structure**
  - e.g. satellite receiver:
    - 13,000 variable node processing elements, $\mathbf{PE_v}$
    - 6,000 check node processing elements, $\mathbf{PE_c}$
- **High throughput; low power**
- Routing complexity

[A. Blanksby and C. J. Howland, JSSC 2002]



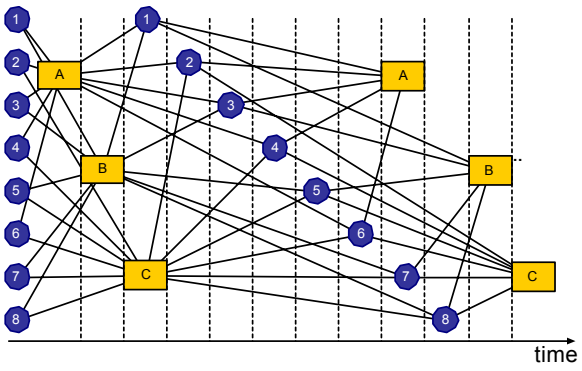
LDPC Code Performance

- **BER performance of LDPC depends on**
  - Graph girth
  - Code expansion + block size
  - Hamming distance
- **"Random" parity-check matrices usually achieve good performance**
  - The interconnect network is random, too
  - A 1024-bit LDPC decoder in 0.15µm occupies 7mm x 7mm with 50% density [Blanksby, Howland, JSSC'02]
- **Structured codes achieve good performance with structured interconnect**
  - Ramanujan and Cayley graphs

Serial LDPC Decoder Architecture

- **Latency dependent on total number of nodes**
- **Messages are stored in SRAM**
  - Large memory requirement
- **Natural structure for microprocessors, DSPs, etc.**
- **Parallelizing computation with limited PEs**

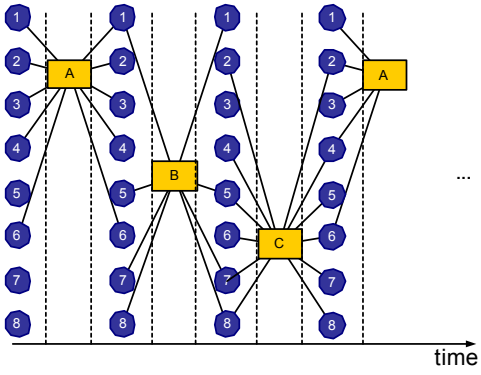
[G. Al-Rawi, J. Cioffi, and M. Horowitz, ITCC 2001]



Staggered Serial LDPC Decoder

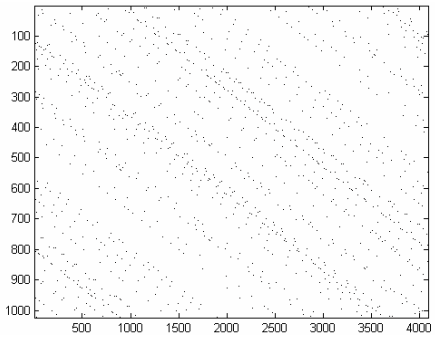
- **Increase number of variable node PEs**
- **Staggered message updating**
  - reduced complexity of PE_v
- **Messages stored in variable node PEs**
  - reduced memory requirement
- **Improved BER @ reduced iteration counts**

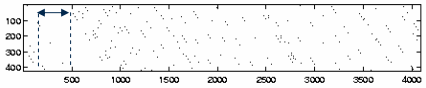
[E. Yeo, et. al. Globecom2001]



LDPC codes based on Galois Fields

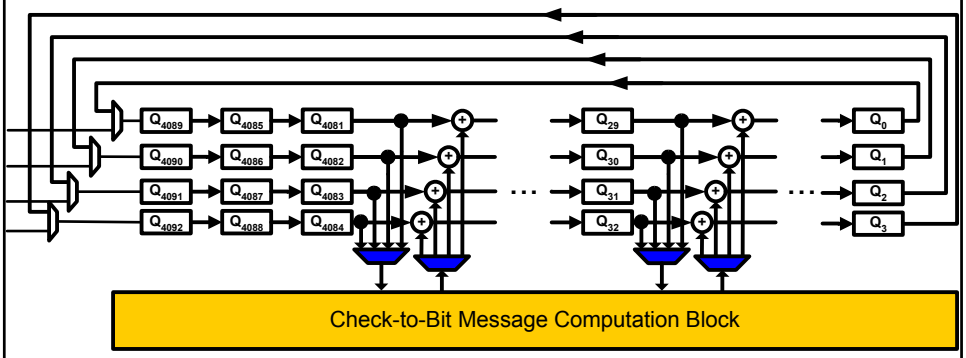
- **Codes based on GF projections are low rate**.
  - No cycles of length 4 (short loop)
  - Cyclic rows
  - e.g. (1023 x 1023) code has rate of 0.68
- ***Column splitting***
  - Each column in original matrix is split into four
  - Non-zero entries in original column are cycled through the 4 new columns
  - eg. (1023 x 4092) code has rate of 0.75
  - Partial loss of regularity (cyclic structure)
  - Complex O(N^2) encoding
- ***Puncturing***
  - Truncate height of PC matrix
  - Columns in the maximum zero runlength region correspond to parity bit locations
  - Cyclic encoding using direct application of PC matrix now possible





[Y. Kou, et. al. ISIT 2000]

Shift register-based implementation

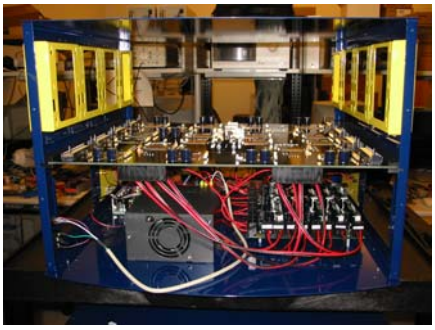


- Staggered decoding.
- Regularity of codes based on Finite Field geometries.

[E. Yeo, et. al. Globecom2001]

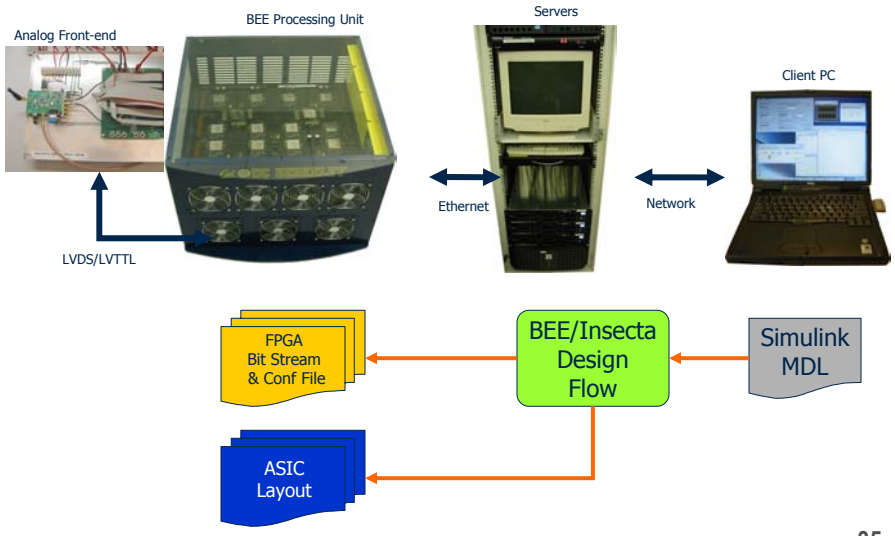
Berkeley Emulation Engine

- **A real-time FPGA-based hardware emulator, with speed** *up to 60 MHz*
- **Emulation capacity of** *10 Million* **ASIC gate-equivalents per module, corresponding to** *600 Gops* **(16-bit adds).**
- *2400* **external parallel I/O providing** *192 Gbps* **raw bandwidth.**
- **Automated design flow from Simulink to FPGA emulation, integrated with INSECTA ASIC design flow.**

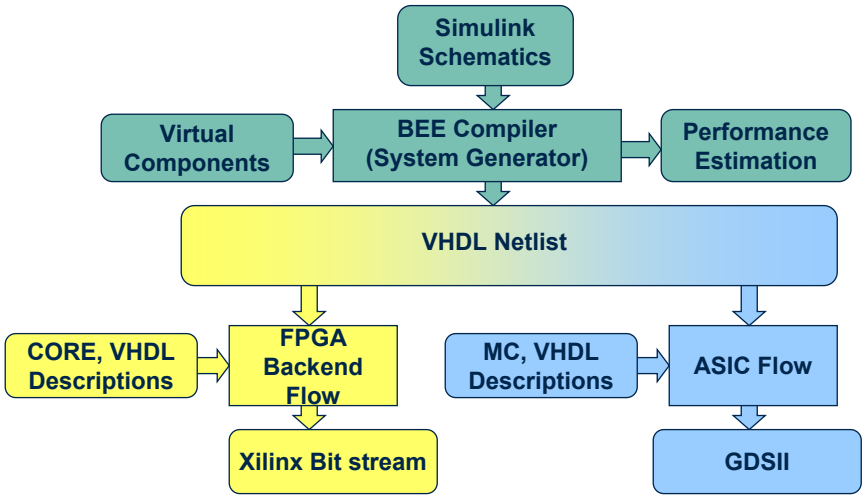


C. Chang, et al, FPGA'03

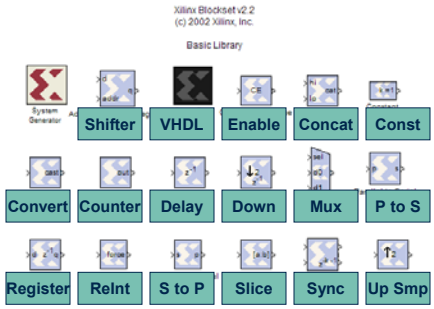
BEE Design Environment

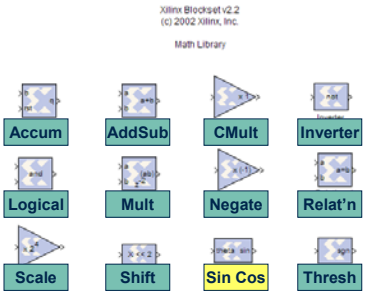


Design Flow: Users' Perspective



Basic Blocks

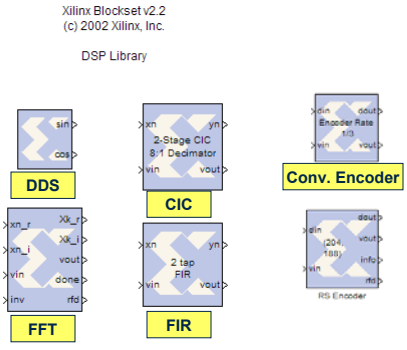


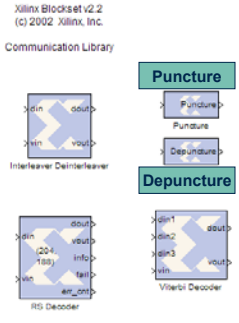


FPGA+ASIC Support

FPGA Support Only

Communication & DSP Blocks

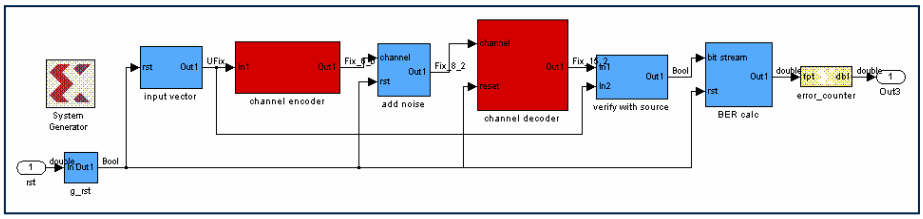




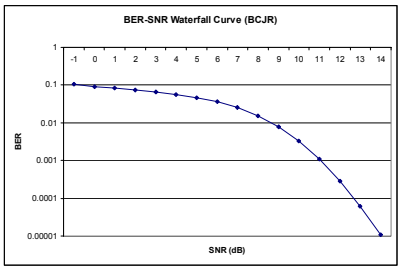
FPGA+ASIC Support

FPGA Support Only

MAP Simulation

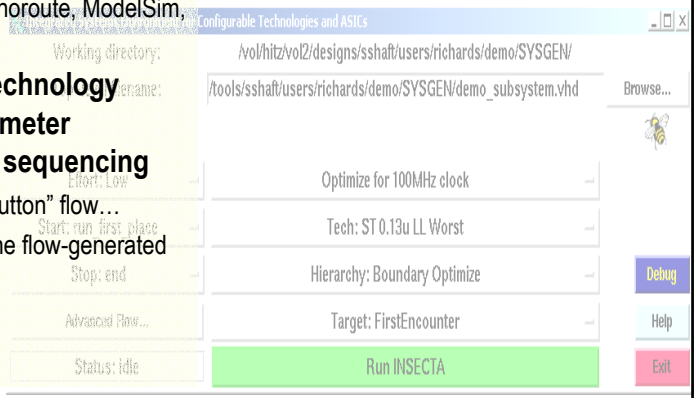


- 10 MHz system clock
- SNR 14db → -1db
- 10^9 Samples
- <30minutes

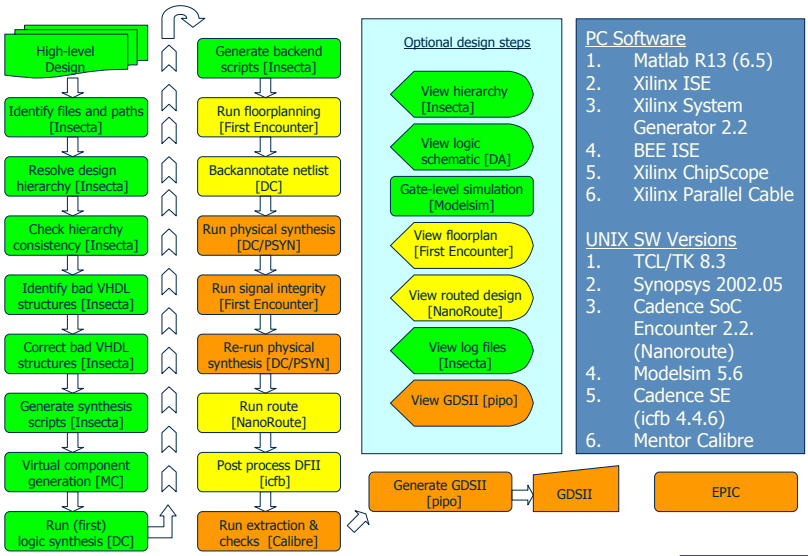


ASIC Flow: INSECTA

- Tcl/Tk code drives the flow
  - Same scripting language used by several EDA tools: First Encounter, Nanoroute, ModelSim, Synopsys…
- GUI controls technology selection, parameter selection, flow sequencing
  - A real "Push Button" flow…
  - Users can refine flow-generated scripts



ASIC Flow: Details



Conclusions

- Communications and storage systems are power and cost constrained, but performance is what sells
- Iterative decoders are several times larger than conventional Viterbi decoder
- Scaling of technology allows integration of iterative decoders
- Emulation technology is necessary for evaluating BER performance
- 90nm technology will likely bring us simple iterative decoders at lower throughput

Acknowledgments

- Most of iterative decoder work was done by Engling Yeo, UC Berkeley
- BEE and INSECTA were developed in Berkeley Wireless Research Center, UC Berkeley by C. Chang, K. Kuusilinna, R. Davis and many others (Prof. Brodersen's group)
- ST Microelectronics, Marvell, Texas Instruments funded through UC Micro program
- ST Microelectronics fabricated chips and provided support