Comparison of Reed-Solomon Codec Implementations

Heather Bowers & Hui Zhang

CS252 Project, Fall 1996

http://infopad.eecs.berkeley.edu/~hui/cs252/rs.html

Why Reed-Solomn Codec?

- The Reed-Solomn coding is a very efficient and popular Forward Error Correction technique.
- It has very much variability in the parameters of implementation, but has regular operations — Great for reconfigurable logic?
- Many literatures available.

Introduction to RS Codec

- Concept of Forward Error Correction
  - » Add redundancy to a message in order to eliminate the need of retransmission by error correction.
- Theory
  - » Reed Solomon coding operates on a finite field called Galios Field.
  - » Generator polynomial: $G(x) = (x-a^0)(x-a^1)\cdot\cdot\cdot(x-a^{2t-1})$, where *a* is a root of the binary primitive polynomial $x^8+x^4+x^3+x^2+1$.
  - » Message polynomial: $M(x)=(M_{k-1}x^{k-1}+ M_{k-2}x^{k-2} + \ldots + M_1x^1 + M_0)\bullet x^{2t-1}$
  - » Check polynomial: $C(x)=M(x) / G(x)$

    $= C_{2t-1}x^{2t-1}+ C_{2t-2}x^{2t-2} + \ldots + C_1x^1 + C_0$
  - » Encoded codes: $(M_{k-1} , M_{k-2} , \ldots, M_0 , C_{2t-1} , C_{2t-2} , \ldots, C_0)$

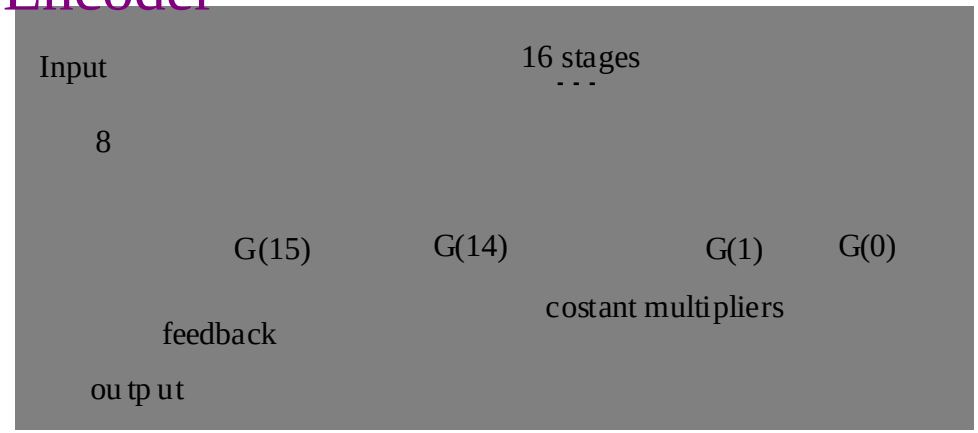
Define RS Codec as a Component

- Specs and Parameters
  - m = number of bits per symbol
  - n = code length in symbols (up to 2^m-1)
  - K = original message length in symbols
  - r = n-k = number of check symbols
  - t = 1/2 (n-k) = error correction capability
  - Total gate counts ≈ 1.6 ($182mt + 292t + 215m + 5m2^m + 208$)
- Interface

Decoder data input	m input bits	Control input bus	m input bits	Decoder input message start	1 input bit
Decoder data output	m output bits	Decoder system clk	1 input bit	Decoder output message start	1 output bit
Encoder data input	m input bits	Encoder system clk	1 input bit	Encoder input message start	1 input bit
Encoder data output	m output bits	reset	1 input bit	Encoder output message start	1 output bit

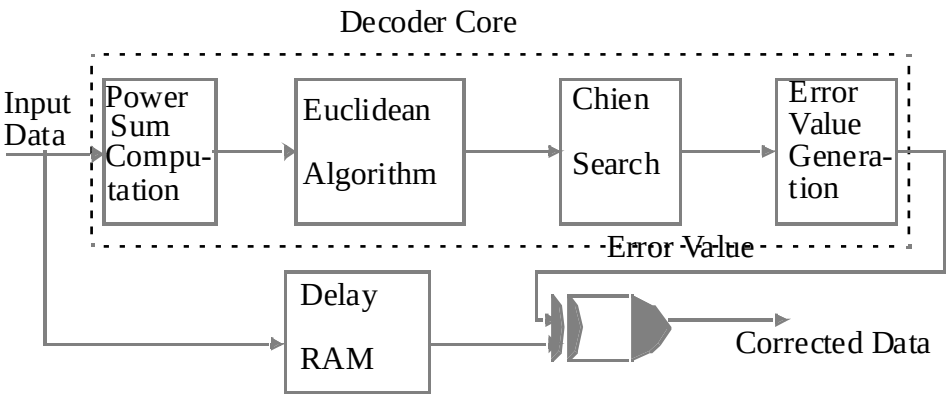
Architecture of RS Codec

Encoder



- Bit-parallel architecture
- 16-stage 8-bit register
- 16 GF(2^8) adders
- 16 GF(2^8) constant multipliers

Decoder



- 4 separate stages
- 7 general GF(2^8) multipliers
- 2 GF(2^8) inverters
- 1 delay RAM

Outline

- Performance Metrics
- Implementations
- Quantitative Evaluation
- Results/Conclusions
- Future Work
  - » Power evaluation
  - » Design cost (financial modeling)
  - » Other components

Metrics

- Silicon real estate — Chip area (λ^2)
- Performance — Throughput (symbol / second)
- Combined metric
  - » Throughput / Area (symbol / second•λ^2)
- Others
  - » Power (mW)
  - » Design cost (engineer month)
- Ultimate single metric

Throughput / mW / $

Custom ASIC Implementation

- LSI Logic's Reed-Solomon Codec

Error Correction	Code Length	Technology	Die Size	Throughput	Logic Gate Count		Memory	Design Time
					Encoder	Decoder		
8 symbols	programmable	1 μ compacted array	9.5mm x 9.5mm	40 MB/s	2K	18K	3Kb RAM 4Kb ROM	6 months

Ref: Po Tong, "A 40-MHz Encoder-Decoder Chip Generated by a Reed-Solomon Code Compiler"

- RS decoder for the Hubble Space Telescope

	Error Correction (t)	Code Length (N)	Technology	Die Size	Throughput	Gate Counts
Standard cell	10 symbols	programmable	1.6 μ 2-metal			82 K
Full custom				8.2 mm x 8.4 mm	10 MB/s	30 K

Microprocessor Implementation

- Algorithm — lookup table based algorithm
  - The C program was downloaded from the internet.
  - The program was modified to conform to the specs of the LSI Logic chip.
  - Codes were added to measure the run time of the encoding and decoding functions.
- Results

SUN HyperSparc 100 MHz processor

Error Correction (t)	Code Length (N)	Technology	Die Size	Throughput (B/s)		Design Time
				Encoder	Decoder	
8 symbols	255	0.5 µ	327 mm^2	478.4 K	46.2 K	4 days

DSP Implementations

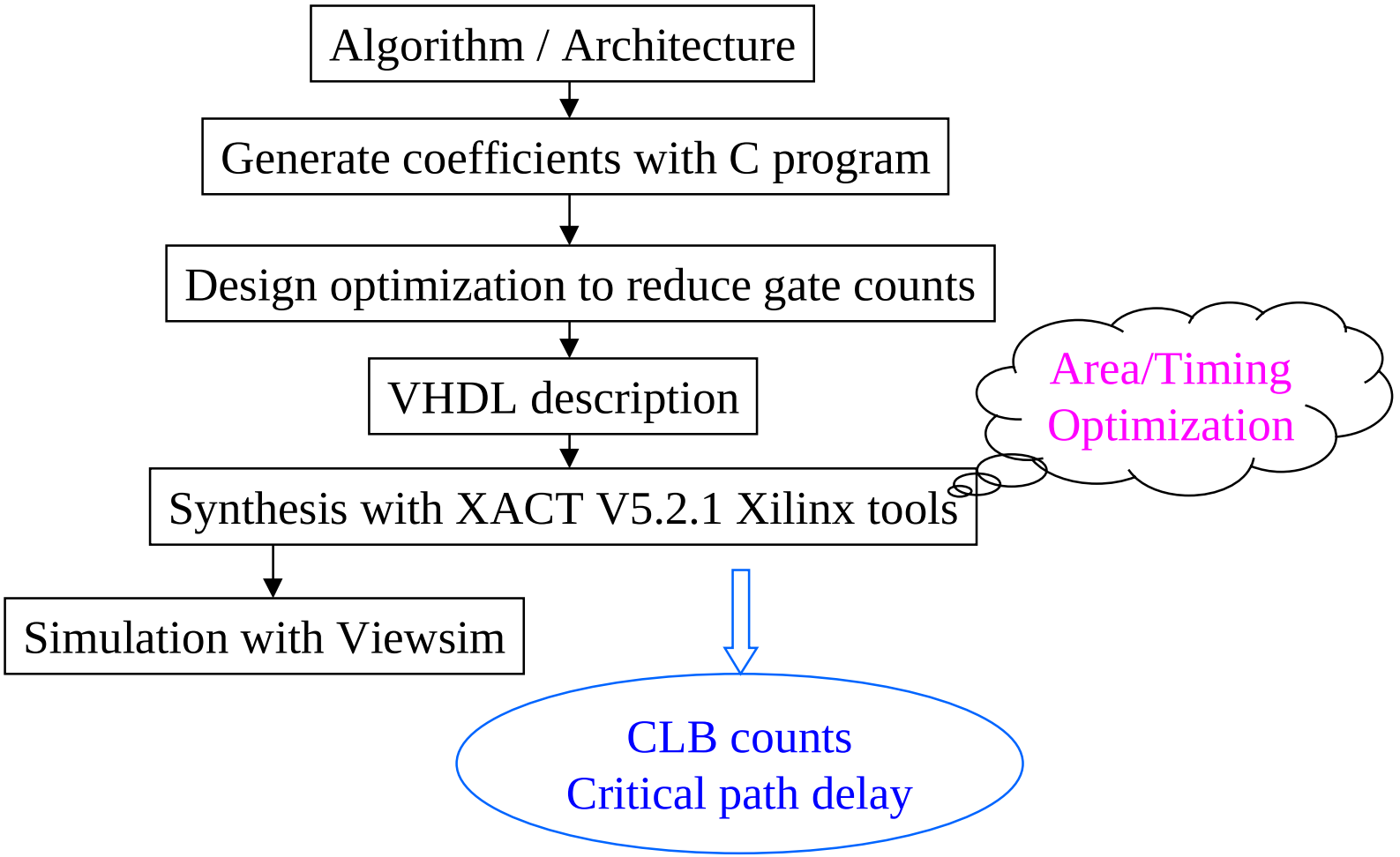
- Implementation Process
  - » The C code was modified to be compatible with TI C compiler for TMS320C3X DSPs
  - » Code was compiled with optimization at TI's online DSP lab
  - » Benchmarked on TI's EVM module with a 33 MHz TMS320C30 DSP
- Results

  TMS320C30: 32-bit floating point, 33 MHz, 16 K words of SRAM

Error Correction (t)	Code Length (N)	Technology	Die Size	Throughput		Design Time
				Encoder	Decoder	
8 symbols	255	0.5 μ	51 mm^2	40.6 KB/s	18.8 KB/s	1 week

FPGA Implementation

Implementation Process



FPGA RS Encoder Results

CLB usage results of Xilinx XC4005 design of RS encoder

Implementations	Usage of I/O (%)	Usage of F-generators (%)	Usage of FFs (%)	Usage of CLBs (%)	Critical path delay
Programmable length	46	77	80	157 / 196	33.8 ns
Fixed length	31	72	78	153 / 196	29.6 ns

Performance/Area Metric Results

Implementations	Area = (# of CLB) x 1.25 M λ^2	Technology	Throughput	Design Time
Programmable length	196	~ 0.8 μ	29.6 MB/s	4 weeks
Fixed length	191		33.7 MB/s	

GF(2^8) Multiplier Design

- Galois Field multiplier is the most critical component of RS decoder design.
- Results

Parts	# of CLBs	Area = (# of CLB) x 1.25 M λ^2	Technology	Delay	Throughput*	Design Time
XC4003	32	40	~ 0.8 μ	49.4 ns	10 MB/s	2 weeks
XC4013E(1)	33	41.2	~ 0.5 μ	16.8 ns	30 MB/s	
XC4013E(2)	30	37.5	~ 0.5 μ	28.2 ns	18 MB/s	

*The critical path delay of decoder is estimated to be twice of that of multiplier

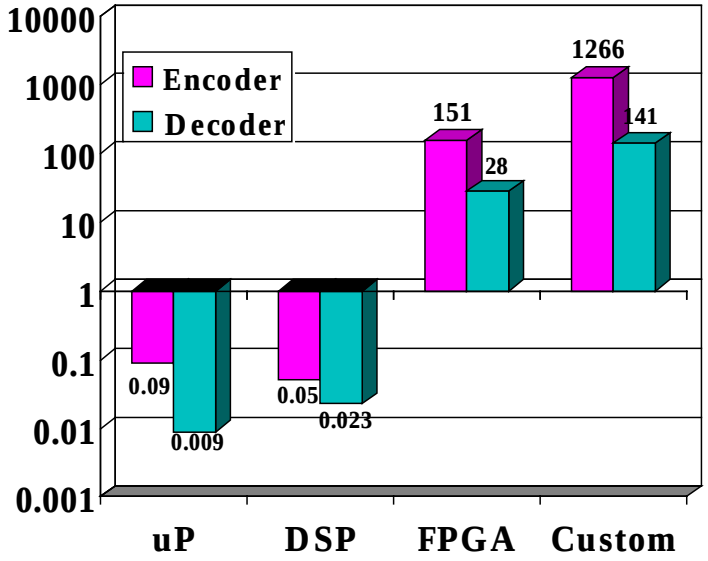
XC4013E implementations were referenced from a paper.

(1) is with timing optimization, and (2) is with area optimization

Comparison

Implementations	Throughput	Performace/Area (KB/s / M λ^2)	Technology	Area (mm^2)	Scaled performance (to 0.25µm)
Microprocessor	478 KB/s	0.09	0.5 µ	327	1 MB/s
DSP	41 KB/s	0.05	~ 0.5 µ	51	82 KB/s
FPGA	30 MB/s	151	~ 0.8 µ	~ 20	96 MB/s
Custom	40 MB/s	1266	1 µ	90	160 MB/s

Implementations	Our Design Time	Initial Design Time	Design Reuse Time
Microprocessor	4 days	1 week	4 hrs
DSP	1 week	1.5 week	2 days
FPGA	4 weeks	3 months	1 week
Custom	6 months	6 months	6 months



Power Estimation

- Microprocessor
  - » C Instruction level power modeling
- DSP
  - » XDS Emulator — a realtime, in-circuit emulator
  - » Assembly Instruction level power modeling
- FPGA
  - » Xilinx data sheet
  - » CLB and interconnect power modeling
- Custom chip
  - » Buy one and measure it