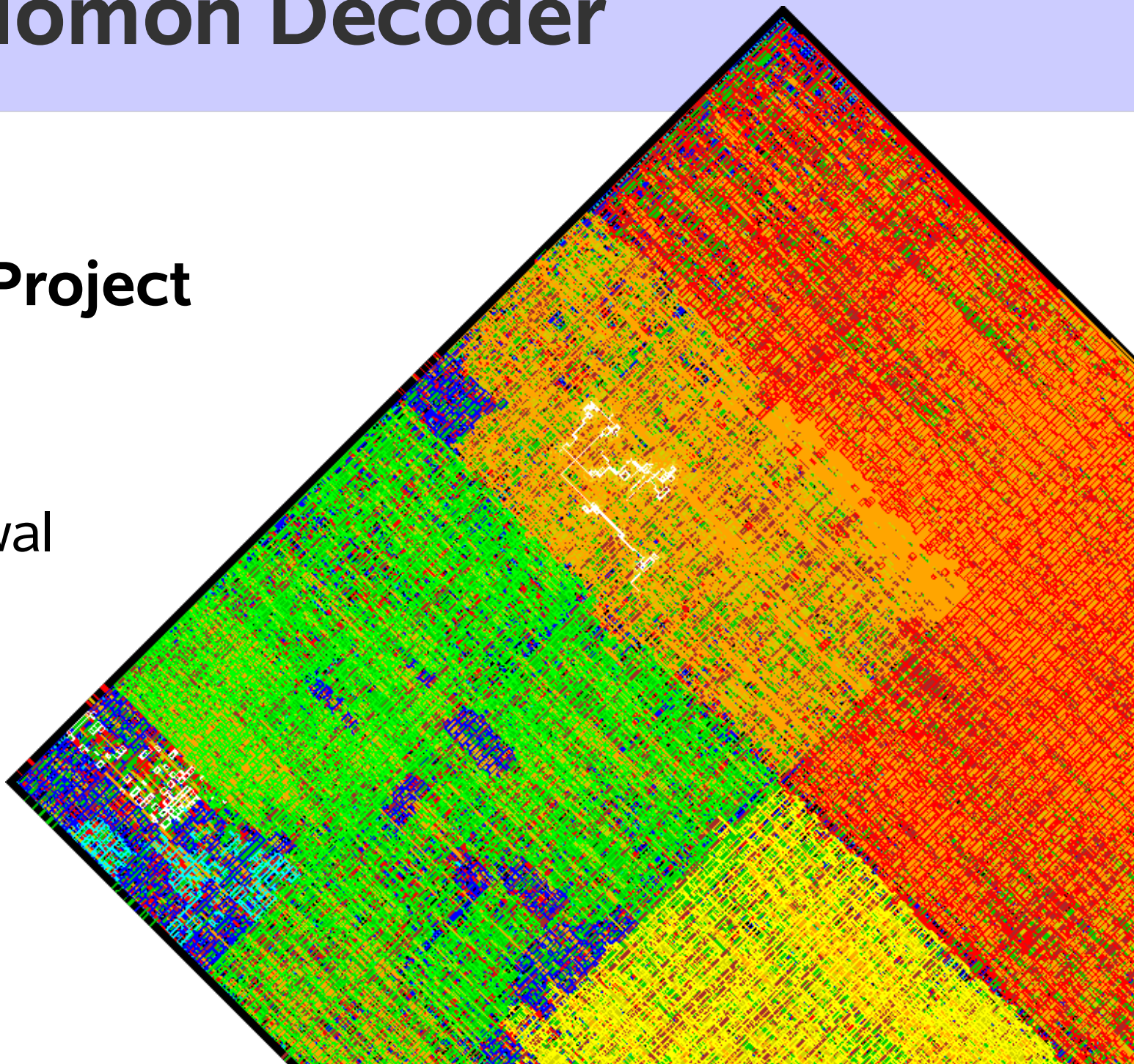
Reed Solomon Decoder

6.375 Final Project

Group 3
Abhinav Agarwal
S Branavan
Grant Elliott

14th May 2007



Outline

- Error Correcting Codes
- Mathematical Foundation of Reed Solomon Codes
- Decoder Architecture
- Decoder Implementation and Exploration
- Performance
- Synthesis and Layout
- Conclusions

Reed Solomon Applications

- Space Probes
- DSL
- WiMax
- CD and DVD
  - Cross-Interleaved Decoder – Two layers of Reed Solomon, the first of which tags uncorrectable blocks as erasures
- 802.16
  - Dynamically adjustable k (6-255) and t (0-16)
  - Specifies Galois Field primitive polynomial
  - Maximum throughput of 29.1Mbps

Error Correction Code

- Add parity so that corrupted messages are closest to the original signal with a given distance metric.
- Corrupt messages fail to meet some criterion, labeling them corrupt
- Search the space around corrupt messages for the nearest (most likely) original message
- Search space is huge. Decoding is intractable in general
- Error Correcting Codes designed to make decoding search feasible

A Brief Word About Math

- Based on Galois Field 2^8 (256) arithmetic
- Symbols encode polynomial coefficients

$$173 = 8\text{b}10101101 = x^7 + x^5 + x^3 + x^2 + x^0$$

- Arithmetic is modulo the primitive polynomial
  - Addition is bitwise modulo 2 (XOR)
  - General Multiplication is polynomial multiplication modulo the primitive polynomial
  - Multiplication by roots of primitive polynomial is particularly easy because x is a root
  - General case inversion is difficult

Galois Field Addition

- Addition in GF(256) reduces to XOR
- Independent of primitive polynomial choice

$$\sum a_i x^i + \sum b_i x^i = \sum (a_i + b_i) x^i$$

$$\begin{array}{ccccc} 123+72 & = & 8b01111011+8b01001000 & = & x^6+x^5+x^4+x^3+x^1+x^0+x^6+x^3 \\ 5 & = & 8b00110011 & = & x^5+x^4+x^1+x^0 \end{array}$$

Galois Field Multiplication

- Multiplication is polynomial multiplication, modulo the primitive polynomial.

$$c = \left(\sum_i x^i \left(\sum_{j>i} a_j b_{i-j}\right)\right) mod\ p$$

- Closed form derives from modulus by successive subtraction

$$\begin{array}{lcll}
123*72 = 8b01111011*8b01001000 & = & (x^6+x^5+x^4+x^3+x^1+x^0)*(x^6+x^3) & mod\,(x^8+x^4+x^3+x^2+x^0) \\
 & = & x^{12}+x^{11}+x^{10}+2x^9+x^8+2x^7+2x^6+x^4+x^3 & mod\,(x^8+x^4+x^3+x^2+x^0) \\
 & = & x^{12}+x^{11}+x^{10}+x^8+x^4+x^3 & mod\,(x^8+x^4+x^3+x^2+x^0) \\
 & = & x^{11}+x^{10}+x^7+x^6+x^3 & mod\,(x^8+x^4+x^3+x^2+x^0) \\
 & = & x^{10}+x^5 & mod\,(x^8+x^4+x^3+x^2+x^0) \\
84 = 8b01010100 & = & x^6+x^4+x^2 &
\end{array}$$

Galois Field Multiplication by Polynomial Roots

- Multiplication by the root $\alpha = x$ is much simpler
- Requires just 8 XORs and 8 ANDs in general. Most optimize if the primitive polynomial p is a constant
- Allows for iterative or combinational calculation of multiplication by powers of α

$$
\begin{aligned}
ax &= \left(\sum a_i x^{i+1}\right) mod\ p \\
&= \left(\sum a_i x^{i+1}\right) + a_8 x\, p \\
&= \sum \left(a_{i-1} + a_8 p_i\right) x^i
\end{aligned}
$$

$$
\begin{aligned}
187 * 2 &= 8b10111011 * 8b00000010 = (x^7+x^5+x^4+x^3+x^1+x^0) * x \quad mod\,(x^8+x^4+x^3+x^2+x^0) \\
&= x^8+x^6+x^5+x^4+x^2+x^1 \quad mod\,(x^8+x^4+x^3+x^2+x^0) \\
107 &= 8b01101011 = x^6+x^5+x^3+x^1+x^0
\end{aligned}
$$

Galois Field Inversion

- The inverse of a is defined as the number b such that $ab = 1 \bmod p$
- Unique inverses are guaranteed if the modulus is primitive
- Characterization for an arbitrary p is difficult. Involves solving simultaneous equations, so closed form is determinants of binary matrices

$$
\begin{array}{lcccl}
123*187 & = & 8b01111011*8b10111011 & = & (x^6+x^5+x^4+x^3+x^1+x^0)*(x^7+x^5+x^4+x^3+x^1+x^0) \quad mod\,(x^8+x^4+x^3+x^2+x^0) \\
 & & & = & x^{13}+x^{12}+x^{10}+x^9+x^2+x^0 \quad mod\,(x^8+x^4+x^3+x^2+x^0) \\
 & & & = & x^{12}+x^{10}+x^8+x^7+x^5+x^2+x^0 \quad mod\,(x^8+x^4+x^3+x^2+x^0) \\
 & & & = & x^{10}+x^6+x^5+x^4+x^2+x^0 \quad mod\,(x^8+x^4+x^3+x^2+x^0) \\
1 & = & 8b00000001 & = & x^0
\end{array}
$$

Reed Solomon Encoder Concept

- Encoder treats data stream of k symbols as coefficients to a k order polynomial

$$M(x) = \sum_{0 \le i < k} m_i x^i$$

- Data polynomial is oversampled at powers of the root of the primitive polynomial, yielding 2t additional parity symbols

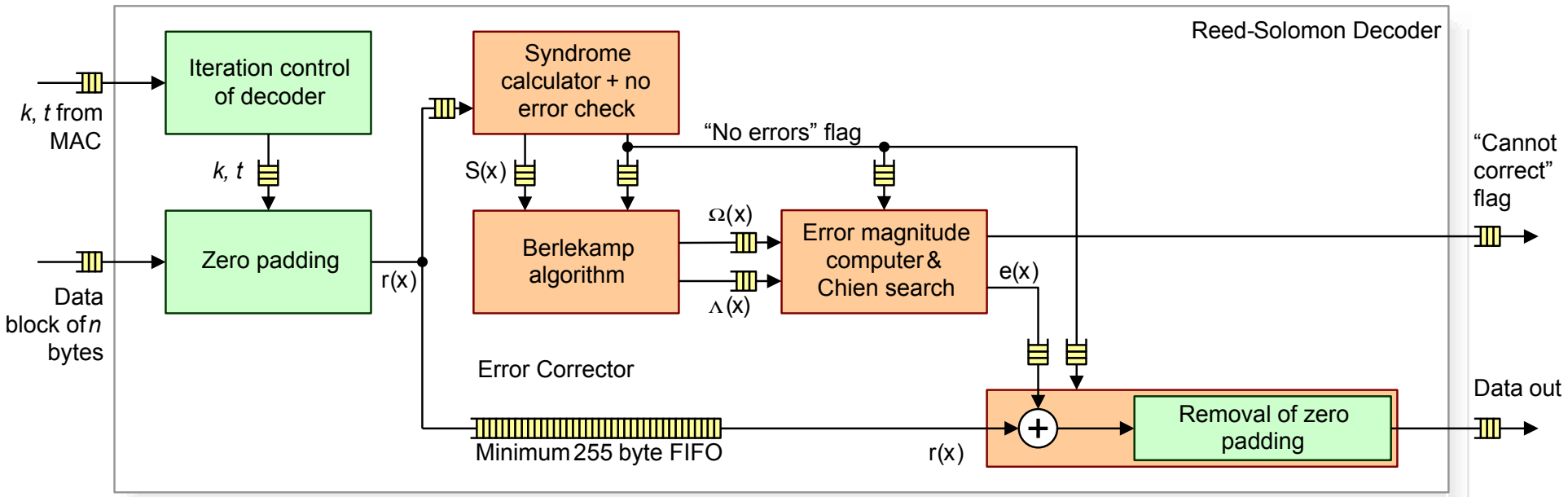
$$c_i = M(\alpha^{i-1})$$

- n=k+2t symbols are transmitted

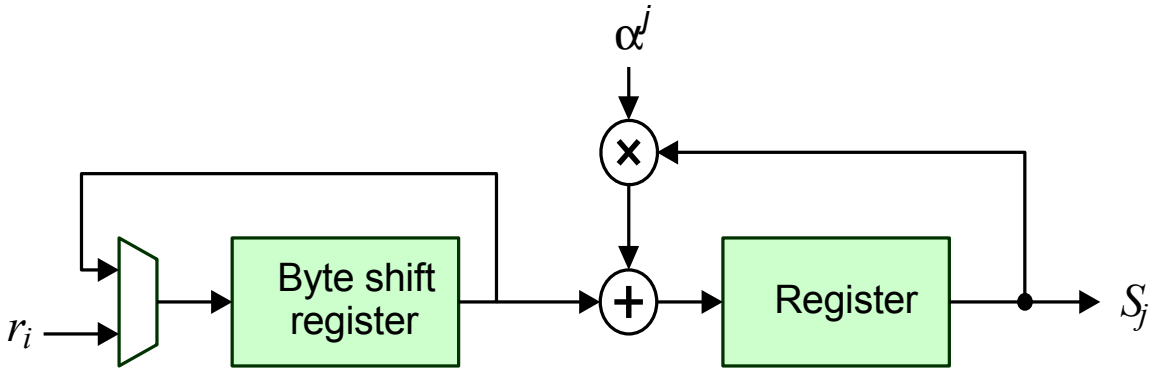
Reed Solomon Decoder Concept

- The first $2t$ powers of α are roots of any valid codeword – a test for corruption.
- For corrupt messages, search for the closest polynomial that meets this criteria
- Can correct up to t symbol errors or $2t$ symbol erasures
- Intractable search made feasible by Berklekamp's algorithm for generating error locator and error evaluator polynomials

Reed Solomon Decoder Architecture



Syndrome Calculator



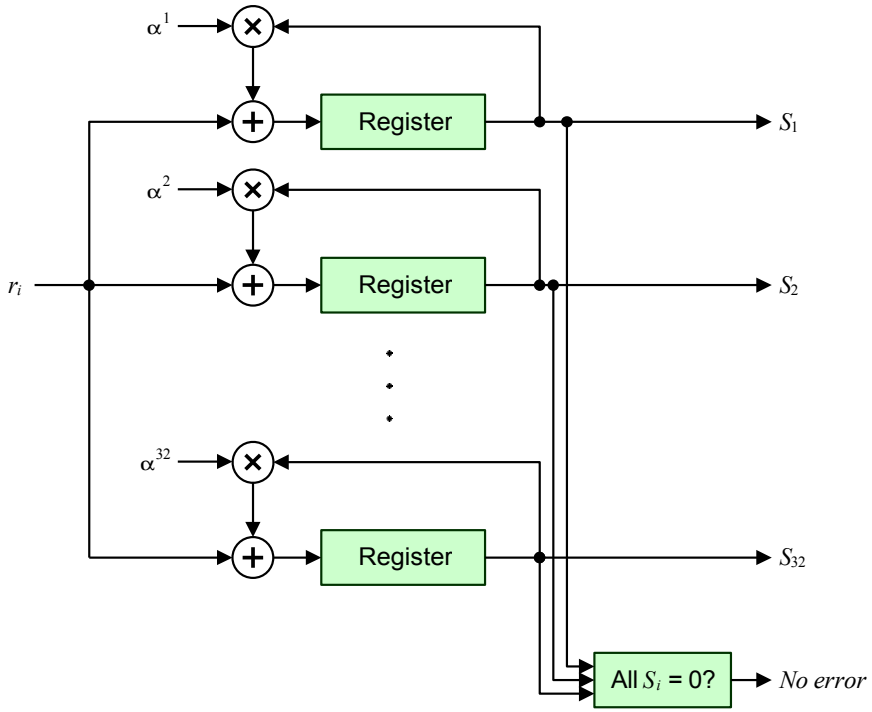
- $2t$ syndromes are calculated by evaluating received polynomial for powers of α

$$S_j = R(\alpha^j)$$

- The $2t$ powers of α are roots of any valid codeword
- If no errors present, later stages are bypassed

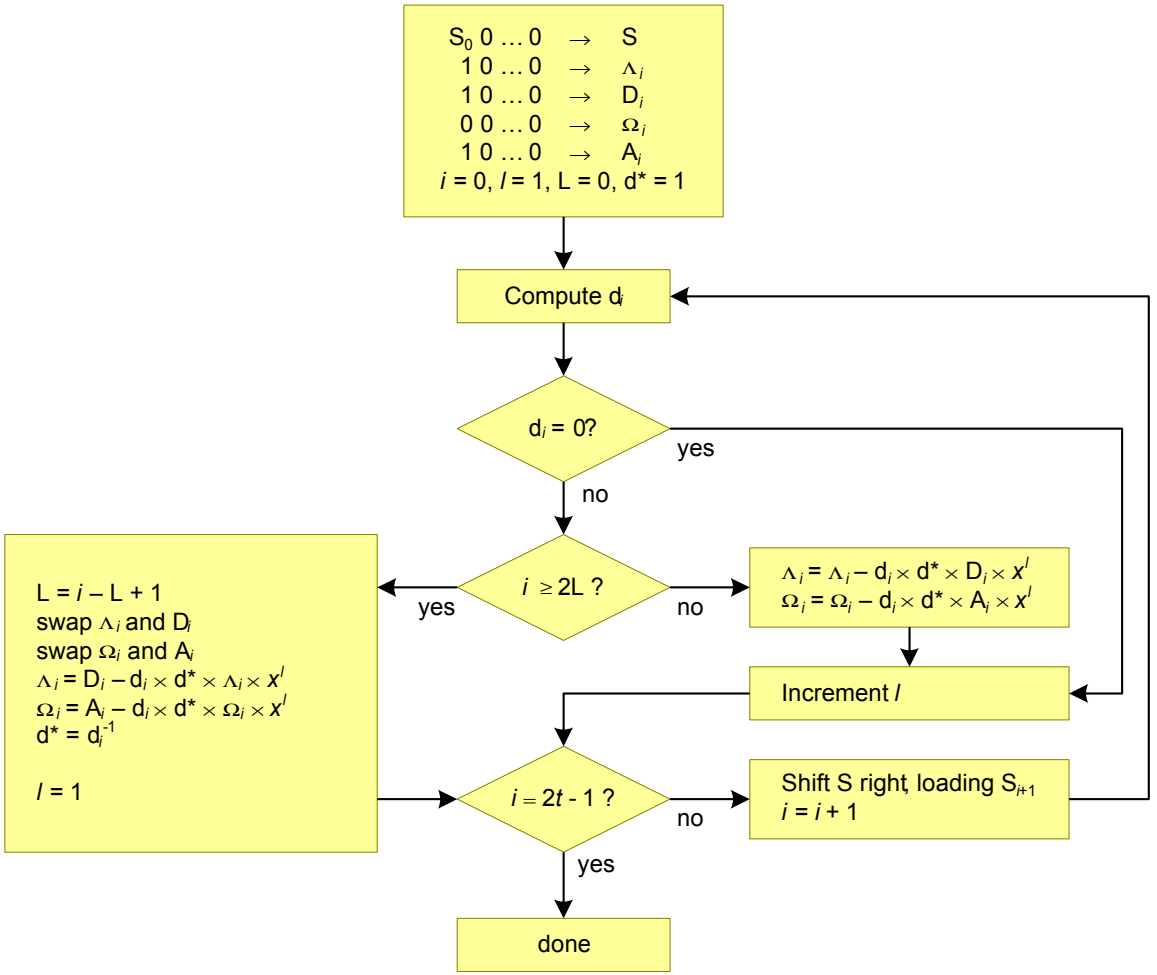
An Efficient Version

- Serial calculation requires 255 byte shift register
- Implementation in parallel is both smaller and faster
- Powers of alpha are functions only of primitive polynomial and enumerate to constants



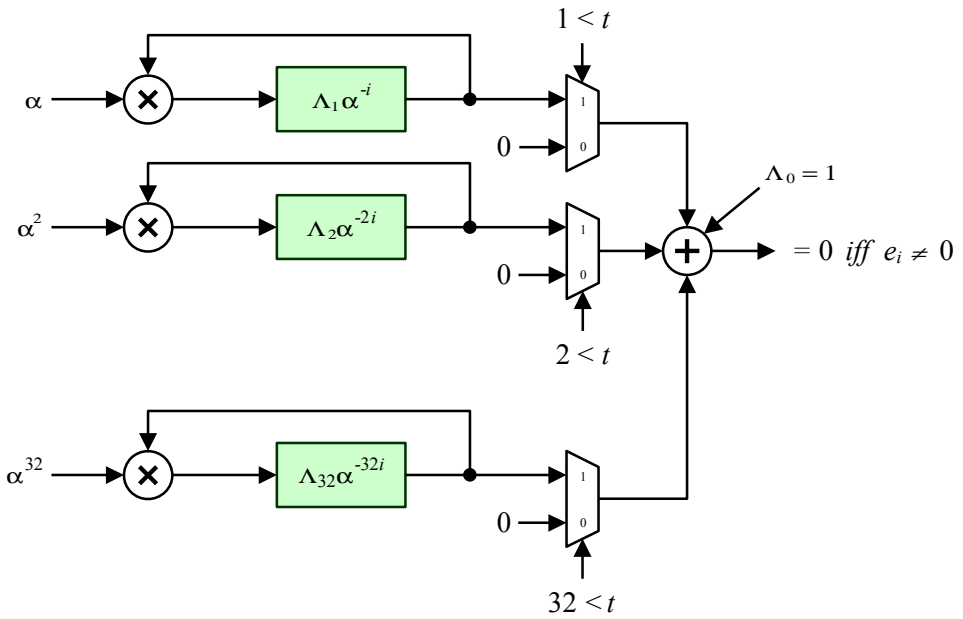
Berlekamp Algorithm

- Calculates the error locator and evaluator polynomials from the syndromes
- Makes Reed Solomon Decoding tractable
- Effectively solves t simultaneous equations



Chien Search and Error Magnitude Calculator

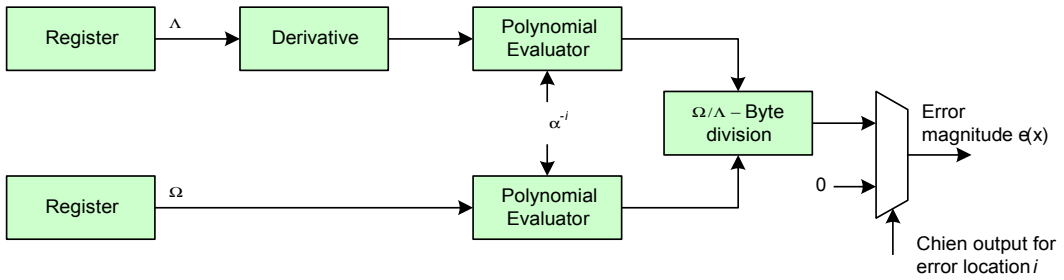
- Finds inverse roots of error locator polynomial to calculate error locations



- Error values

$$e_i = \frac{\Omega(\alpha^{-i})}{\Lambda'(\alpha^{-i})}$$

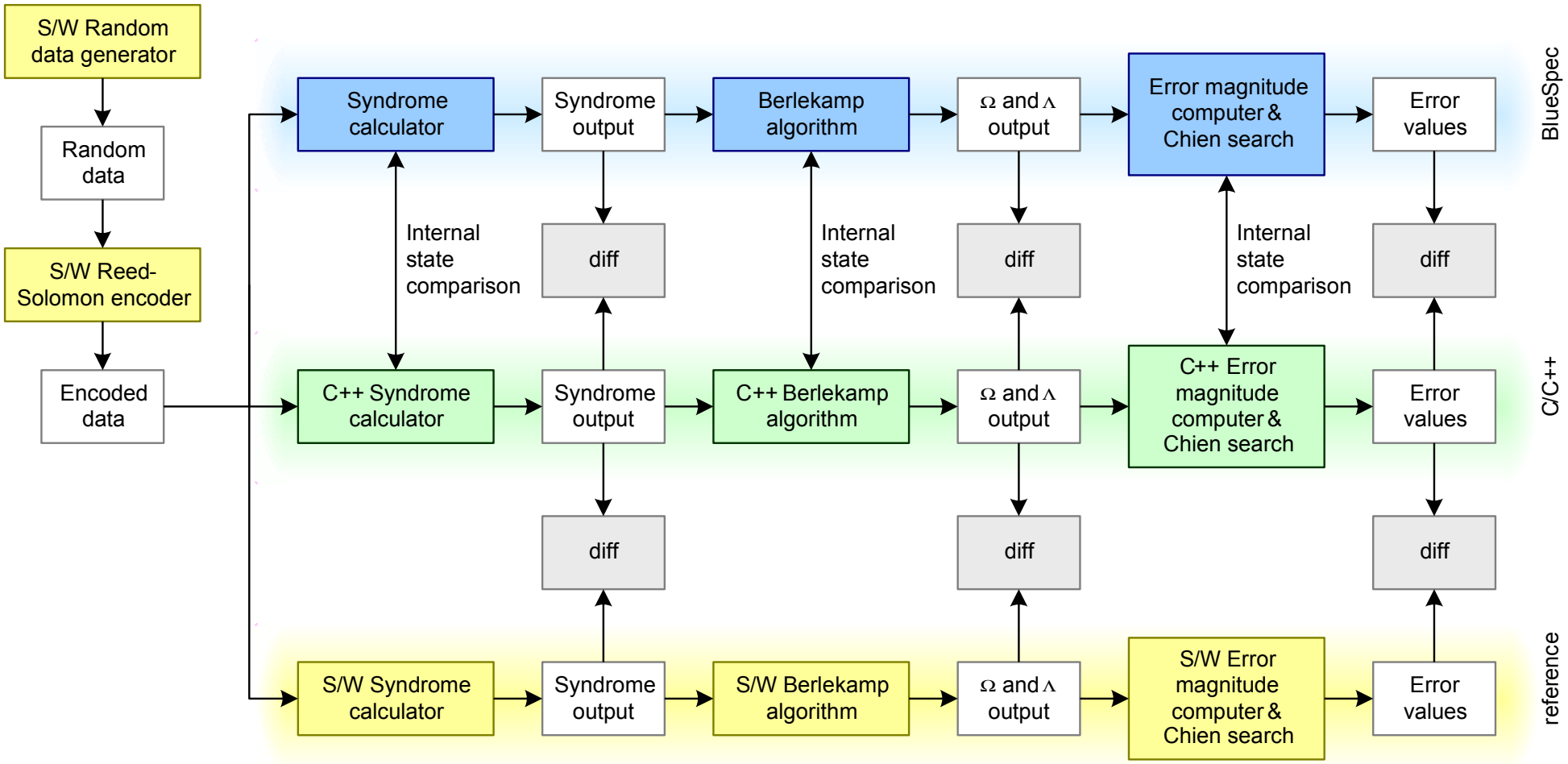
are computed for each error location



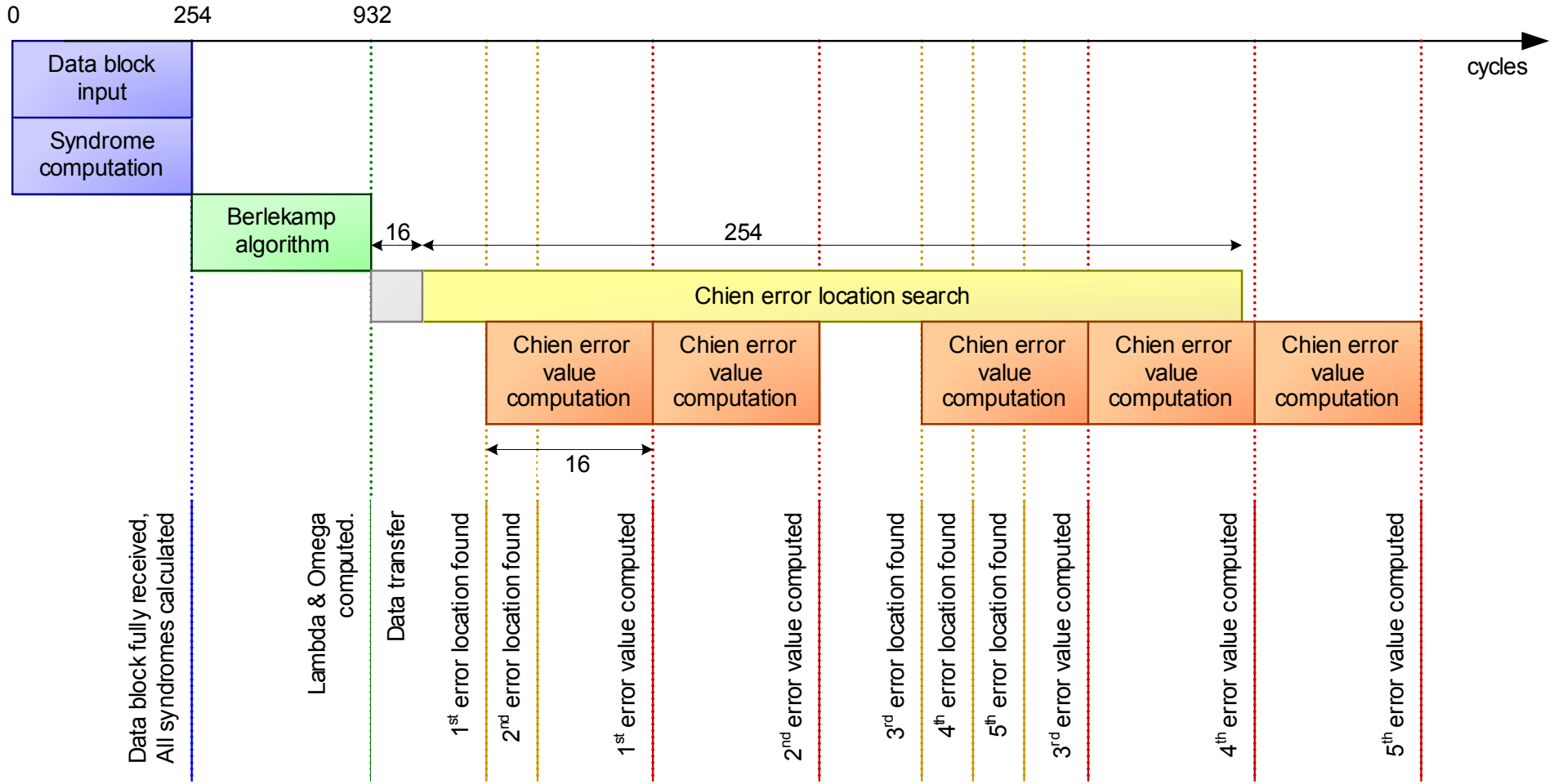
Verification & Testing

- Individual modules verified against
  - C++ implementations of our design
  - Publicly available C decoder
- Full decoder verified against
  - MATLAB Communications Toolbox implementation
- Automatic testing
  - Large number (10,000) of randomly generated messages with random lengths, parities & errors.

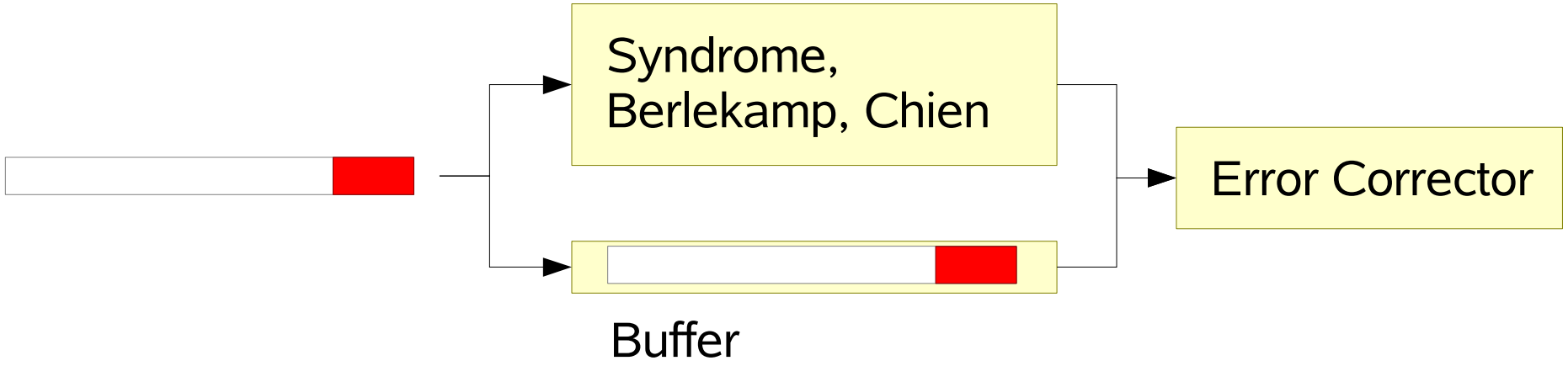
Testing


S/W Random data generator
Random data
S/W Reed-Solomon encoder
Encoded data
Syndrome calculator
Syndrome output
Berlekamp algorithm
Ω and Λ output
Error magnitude computer & Chien search
Error values
BlueSpec
Internal state comparison
diff
Internal state comparison
diff
Internal state comparison
diff
C++ Syndrome calculator
Syndrome output
C++ Berlekamp algorithm
Ω and Λ output
C++ Error magnitude computer & Chien search
Error values
C/C++
diff
diff
diff
S/W Syndrome calculator
Syndrome output
S/W Berlekamp algorithm
Ω and Λ output
S/W Error magnitude computer & Chien search
Error values
reference

Performance



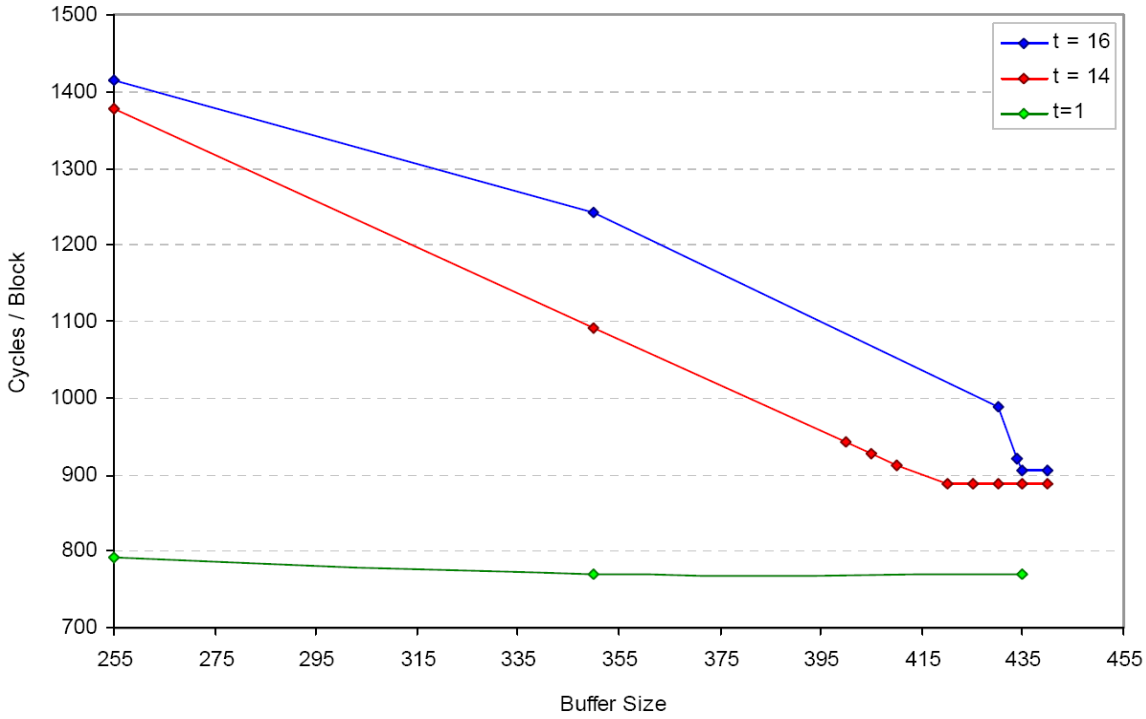
Worst-case data stream



- The Buffer needs to maintain data until corrected.
- The worst-case scenario is when a burst error occurs at the start of a block.

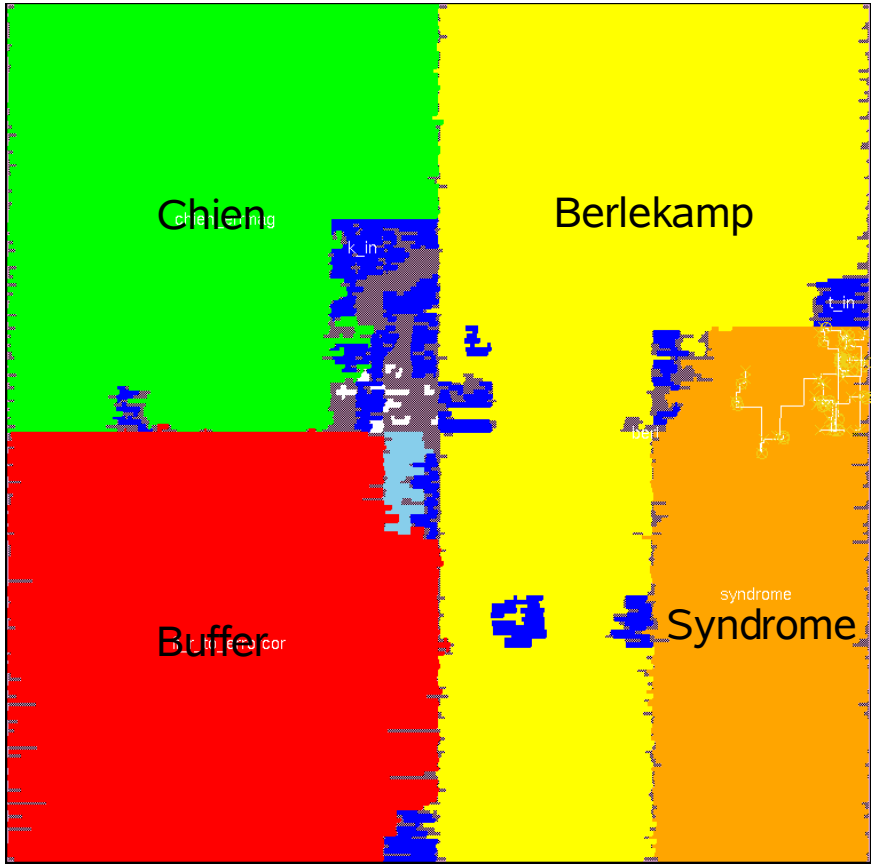
Buffer Design Exploration

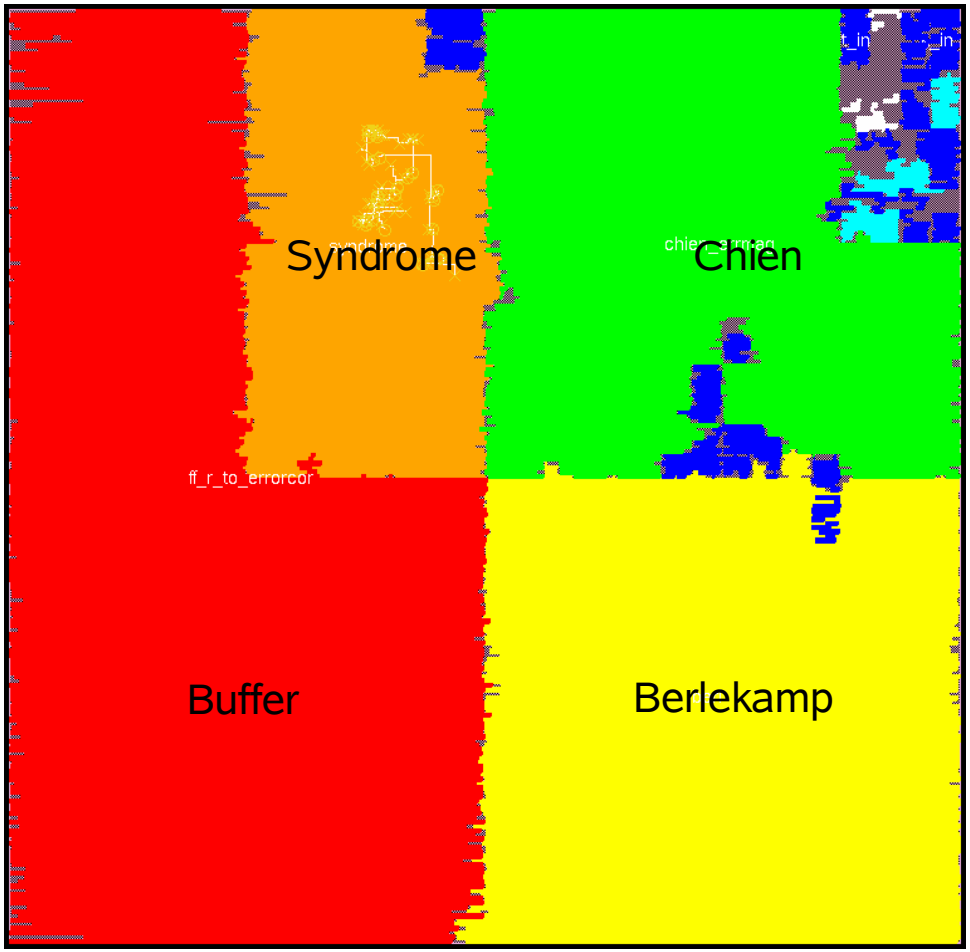
- Buffer must hold at least one full block to prevent stalling and no benefit is derived from holding more than three
- Optimal choice empirically confirmed



Buffer Size	Area (μm^2)	Power (mW)	Clock (ns)	Cycle Count
255	846089.7	214.76	4.849	1414
435	1030593.1	272.31	4.728	905

Die comparison with Buffer Size





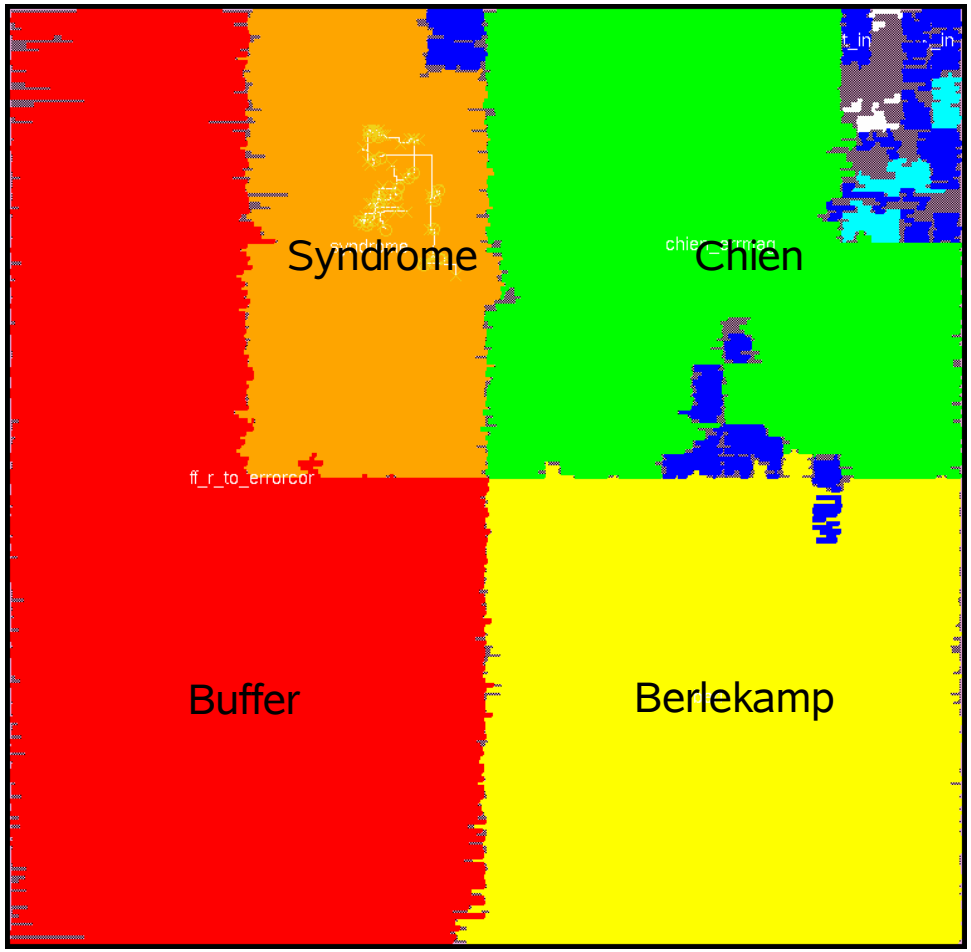
Synthesis and Layout

Module	Area (μm^2)	Percentage
Syndrome	140474	13.6
Berlekamp	242773	23.6
Chien	175672	17.0
Error Corrector	5246	0.5
Buffer	417586	40.5
TOTAL	1030593	100

Clock period : 4.728 ns
Critical path : 4.720 ns
[No error check] in Syndrome

Synthesis for 802.16

Clock : 64 ns (15.6 Mhz)
Power : 13.94 mW



Throughput

- Highest number in Ng et al : 29.1 Mbps
- Highest number in 802.16 spec : 134 Mbps
- Maximum empirical achieved by optimal design : **393 Mbps**

Conclusions

- Reed Solomon decoder parameterized by the primitive polynomial.
- Meets specification for 802.16, but easily adaptable to other applications, such as CD.
- Exceeds 802.16 throughput requirement by a factor of 10.
- Designed for integration with the OFDM framework.
- Developed generalized GF arithmetic functions for Bluespec.

References

- George C. Clark & Bibb Cain, Error-Correction Coding for Digital Communications.
- Todd K. Moon, Error Correction Coding - Mathematical Methods and Algorithms.
- Ng et al, From WiFi to WiMAX: Techniques for IP Reuse across Different OFDM Protocols.
- www.eccpage.com (Non-commercial Reed-Solomon codec implementation by Simon Rockliff, University of Adelaide)
- Communications Toolbox, MATLAB 7.3