What is PUF?

Examples of PUFs

PUF for identification and Authentication

PUF Notation, An approach to make a PUF experiment

Helper Data Algorithm or ECC based noise removal

BCH encoding and decoding for error correction

Security, Threats and Environmental Parameters
PUF as a Function

- Let \( x = 2^n \), \( y = 2^m \) where \( x \) denotes a set of \( n \)-bit challenges and \( y \) denotes a set of \( m \)-bit responses.
- Mapping between challenges and responses
- \((x_i, y_i)\) pairs are determined by manufacturing variations of the device.
Design philosophy

- $H(x_i, y_i)$ is distributed normally from 1 to $n$.
- $H(y_i, y_j)$ is approx. equal to $\frac{n}{2}$ when $H(x_i, x_j) = 1$ is given

"Apply crypt. hash function to response"
Ideal PUF

- Design philosophy
  - $H(x_i, y_i)$ is distributed normally from 1 to $n$.
  - $H(y_i, y_j)$ is approx. equal to $\frac{n}{2}$ when $H(x_i, x_j) = 1$ is given
    - “Apply crypt. hash function to response”
  - Observation requires evaluation of PUF
    - “Ring oscillator type PUF satisfies this”
Ideal PUF

- Design philosophy
  - $H(x_i, y_i)$ is distributed normally from 1 to $n$.
  - $H(y_i, y_j)$ is approx. equal to $\frac{n}{2}$ when $H(x_i, x_j) = 1$ is given
    - “Apply crypt. hash function to response”
  - Observation requires evaluation of PUF
    - “Ring oscillator type PUF satisfies this”

- Summary:
  - $PUF_i$ should be unique to device $i$

<table>
<thead>
<tr>
<th>Challenge</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>100010101</td>
<td>110110</td>
</tr>
<tr>
<td>100010101</td>
<td>011010</td>
</tr>
<tr>
<td>101010101</td>
<td>010011</td>
</tr>
<tr>
<td>011010010</td>
<td>0101100</td>
</tr>
</tbody>
</table>

- Challenge
- Response

- Challenge
- Response

- Challenge
- Response

- Challenge
- Response

- Challenge
- Response

- Challenge
- Response

- Challenge
- Response

- Challenge
- Response

- Challenge
- Response

- Challenge
- Response
Ideal PUF

- **Design philosophy**
  - $H(x_i, y_i)$ is distributed normally from 1 to $n$.
  - $H(y_i, y_j)$ is approx. equal to $\frac{n}{2}$ when $H(x_i, x_j) = 1$ is given
    
    “Apply crypt. hash function to response”
  - Observation requires evaluation of PUF
    
    “Ring oscillator type PUF satisfies this”

- **Summary:**
  - $PUF_i$ should be unique to device $i$
  - Each challenge and response pair should not be linkable


**Ideal PUF**

- **Design philosophy**
  - \( H(x_i, y_i) \) is distributed normally from 1 to \( n \).
  - \( H(y_i, y_j) \) is approx. equal to \( \frac{n}{2} \) when \( H(x_i, x_j) = 1 \) is given
  
    “Apply crypt. hash function to response”
  - Observation requires evaluation of PUF
    
    “Ring oscillator type PUF satisfies this”

- **Summary:**
  - \( PUF_i \) should be unique to device \( i \)
  - Each challenge and response pair should not be linkable
  - Challenge and response pairs should be independent

<table>
<thead>
<tr>
<th>Challenge</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>100010101</td>
<td>110110</td>
</tr>
<tr>
<td>100010101</td>
<td>010110</td>
</tr>
<tr>
<td>101010101</td>
<td>100011</td>
</tr>
<tr>
<td>011010010</td>
<td>011100</td>
</tr>
</tbody>
</table>

![Diagram](chart.png)
Principle of PUF

R1 \neq R2 \neq R3 \neq \cdots \neq Rm
### Process variations in MOSFET

<table>
<thead>
<tr>
<th>Components (MOSFET)</th>
<th>Interconnect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Geometry</td>
<td></td>
</tr>
<tr>
<td>Effective channel length</td>
<td>Line width and space</td>
</tr>
<tr>
<td>Gate length</td>
<td>Metal thickness</td>
</tr>
<tr>
<td>Component width</td>
<td>Contact and via size</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Material Parameters</td>
<td></td>
</tr>
<tr>
<td>Doping variations</td>
<td>Contact and via resistance</td>
</tr>
<tr>
<td>Deposition and anneal</td>
<td>Metal resistivity</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Electrical Parameters</td>
<td></td>
</tr>
<tr>
<td>Threshold voltage</td>
<td>Line resistivity</td>
</tr>
<tr>
<td>Parasitic capacities</td>
<td>Line capacity</td>
</tr>
<tr>
<td>Gate and source resistivity</td>
<td></td>
</tr>
<tr>
<td>Leakage currents</td>
<td></td>
</tr>
</tbody>
</table>

Some important parameters that are affected by process variations, and the electrical parameters on which this is an impact.

Each challenge selects a unique pair of delay paths

Digital race condition on two paths with an identical delay in design
  - Random, uncontrollable process variations determines who will win
  - Arbiter output creates 1-bit output response
  - Multiple bits are obtained by duplicating the circuit or use of different challenge
Each challenge selects a unique pair of delay paths

Digital race condition on two paths with an identical delay in design
  - Random, uncontrollable process variations determines who will win
  - Arbiter output creates 1-bit output response
  - Multiple bits are obtained by duplicating the circuit or use of different challenge
Delay difference and so the arbiter output will be **device specific.** of an Arbiter PUF.

*** A *metastable state* occurs if both delays are nearly identical. After a short time, arbiter leaves the metastable state and outputs a random binary value.

In that case, the arbiter output is not device-specific. This phenomenon is the cause of unreliability (**noise**) of the responses.
Delay difference and so the arbiter output will be **device specific**. of an Arbiter PUF.

*** A *metastable state* occurs if both delays are nearly identical. After a short time, arbiter leaves the metastable state and outputs a random binary value.

In that case, the arbiter output is not device-specific. This phenomenon is the cause of unreliability (**noise**) of the responses.
Delay difference and so the arbiter output will be device specific. of an Arbiter PUF.

*** A metastable state occurs if both delays are nearly identical. After a short time, arbiter leaves the metastable state and outputs a random binary value.

In that case, the arbiter output is not device-specific. This phenomenon is the cause of unreliability (noise) of the responses.
Delay difference and so the arbiter output will be **device specific**. of an Arbiter PUF.

*** A *metastable state* occurs if both delays are nearly identical. After a short time, arbiter leaves the metastable state and outputs a random binary value.

In that case, the arbiter output is not device-specific. This phenomenon is the cause of unreliability (**noise**) of the responses.
Delay difference and so the arbiter output will be **device specific**. of an Arbiter PUF.

*** **A metastable state** occurs if both delays are nearly identical. After a short time, arbiter leaves the metastable state and outputs a random binary value.

In that case, the arbiter output is not device-specific. This phenomenon is the cause of unreliability (**noise**) of the responses.
Delay-based PUF: Arbiter PUF

- Delay difference and so the arbiter output will be device specific. of an Arbiter PUF.

*** A metastable state occurs if both delays are nearly identical. After a short time, arbiter leaves the metastable state and outputs a random binary value.

- In that case, the arbiter output is not device-specific. This phenomenon is the cause of unreliability (noise) of the responses.
Delay difference and so the arbiter output will be **device specific**. of an Arbiter PUF.

*** A *metastable state* occurs if both delays are nearly identical. After a short time, arbiter leaves the metastable state and outputs a random binary value.

In that case, the arbiter output is not device-specific. This phenomenon is the cause of unreliability (**noise**) of the responses.
Memory-based PUF: SRAM PUFs

(1) Logic circuit of an SRAM PUF (a cell)

(2) Transistor-level circuit of an SRAM PUF in a standard CMOS tech.
Memory-based PUF: Butterfly PUFs

(1) Logic circuit of an SRAM PUF (a cell)

(2) Schematic-level circuit of a butterfly PUF cell
PUF for Identification

- PUF-based identification method uses a unique and untamperable device identifier exploiting the physics of the device.

- Identification is similar to authentication but in this context it has weaker concept.

- PUF should provide the identity of the device without any convincing proof.

- In certain situations, identification is enough to achieve entity authentication.
PUF for Authentication

.toggle

- PUF gives a measure of the device specific physical feature.

- Therefore, device authentication is provided by PUF (not message authentication).

- Besides plain identification, also corroborative evidence of the device identity is required.

- Corroborative evidence means that it could only have been created by that particular device.

- For the device authentication:
  - the identity of a second party involved
  - the second party is active at the time of the evidence is created.
Device Authentication

Untrusted Supply chain

Is this device authentic to the manufacturer?

Device A

PUF A

(c_i, r_i)

Challenge

Response

10010101, 010110
10010101, 010110
10010101, 010110

PUF A

Challenge

Response

10010101, 010110
10010101, 010110
10010101, 010110

==
Creation of a PUF Instance:

\[ \mathcal{P} \equiv \{ \text{puf}_i \leftarrow \mathcal{P}.\text{Create}(r^C_i) : \forall i, r^C_i \leftarrow \{0, 1\}^* \} \]

Random evaluation of PUF instance \( \text{puf}_i \) for challenge \( x \):

\[
y^{(j)}_i(x) \leftarrow \text{puf}_i(x).\text{Eval}(r^E_j \leftarrow \{0, 1\}^*)
\]

\[
Y_i(x) \leftarrow \text{puf}_i(x).\text{Eval}
\]

where \( r^C \) and \( r^E \) are randomization variables representing an undetermined fair coin tosses.
a PUF class generates responses for the challenges provided to it

\[ Y_i(x) \leftarrow \text{puf}_i(x).\text{Eval} \]

where \( Y_i \)'s are responses for challenge \( x \).

Statistically, the distribution of PUF responses should be estimated.

This require experiments to understand the distribution statistics from observed PUF responses:

1. Experiments estimating the distribution for same challenge \( x \) on distinct evaluations.
2. These experiments should be repeated on responses for different challenges.
3. Experiments on responses from different PUF instances
PUF Experiment

- A PUF class generates responses for the challenges provided to it

\[
\text{Experiment}_P(N_{puf}, N_{chal}, N_{meas}) \rightarrow Y_{Exp(P)}
\]

\[
Y_{Exp(P)} = [y_i^{(j)}(x_k) \leftarrow \text{puf}_i(x_k).\text{Eval}(r_j^E)]
\]

with

\[
\forall 1 \leq j \leq N_{meas} : r_j^E \leftarrow \{0, 1\}^*,
\]

\[
\forall 1 \leq k \leq N_{chal} : x_k \leftarrow \mathcal{X}_P,
\]

\[
\forall 1 \leq i \leq N_{puf} : \text{puf}_i \leftarrow \mathcal{P},
\]

- If the conditions are important while doing experiments, then they should be specified with a parameter.
PUF Response Distances: \textit{intra} distance

- A PUF response \textit{intra} distance is a random variable refers to the distance between two responses from the same PUF instance and using the same challenge.

\[
D_{\text{intra}}^{\text{puf}_i}(x) \triangleq \text{dist}[Y_i(x); Y'_i(x)],
\]

with $Y_i(x)$ and $Y'_i(x)$ two distinct random evaluations of PUF instance $\text{puf}_i$ on the same challenge $x$. $\text{dist}[;]$ denotes hamming distance between two random variables.

- For the \textit{reproducibility} of a PUF class, distribution of $D_{\text{intra}}^{\text{puf}_i}$ has some statistical properties.

- Estimation of mean ($\mu_{\text{intra}}^{\text{puf}_i}$), standard deviation ($\sum[D_{\text{intra}}^{\text{puf}_i}]$), histogram and order statistics should be found.
A PUF response **inter** distance is a random variable refers to the distance between two responses from different PUF instances and using the same challenge.

\[ D_{\text{inter}}^P(x) \triangleq \text{dist}[Y(x); Y'(x)], \]

with \( Y(x) \) and \( Y'(x) \) two distinct random evaluations of two distinct PUF instances on the same challenge \( x \).

For the **reproducibility** of a PUF class, distribution of \( D_{\text{inter}}^\text{puf}_i \) has some statistical properties.

Estimation of mean \( (\mu_{\text{inter}}) \), standard deviation \( (\sum[D_{\text{inter}}^P]) \), histogram and order statistics should be found.
Quantitative tests yield to model a PUF by Hori et al., *Quantitative and Statistical Performance Evaluation of Arbiter Physical Unclonable Functions on FPGAs*. ReConFig2010, pp.298-303, 2010.
## Experimental uniqueness results

<table>
<thead>
<tr>
<th>PUF</th>
<th>No</th>
<th>(N_{bits})</th>
<th>(\mu_{inter})</th>
<th>(\sigma_{inter})</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRAM PUF</td>
<td>0</td>
<td>65536</td>
<td>49.59%</td>
<td>0.33%</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>65536</td>
<td>49.61%</td>
<td>0.33%</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>65536</td>
<td>49.68%</td>
<td>0.31%</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>65536</td>
<td>49.72%</td>
<td>0.30%</td>
</tr>
<tr>
<td>Arbiter PUF (basic)</td>
<td>0</td>
<td>65536</td>
<td>47.13%</td>
<td>0.44%</td>
</tr>
<tr>
<td>Arbiter PUF (2-XOR)</td>
<td>0</td>
<td>32768</td>
<td>49.74%</td>
<td>0.29%</td>
</tr>
<tr>
<td>Ring Oscillator PUF (L.G.)</td>
<td>0</td>
<td>12544</td>
<td>46.86%</td>
<td>0.48%</td>
</tr>
<tr>
<td>Latch PUF</td>
<td>0</td>
<td>8192</td>
<td>34.84%</td>
<td>1.20%</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>8192</td>
<td>37.01%</td>
<td>1.23%</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>8192</td>
<td>33.17%</td>
<td>1.62%</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>8192</td>
<td>16.37%</td>
<td>2.02%</td>
</tr>
</tbody>
</table>

**Inter-distances statistics (at nominal condition)**

Two problems are solved with HDA or FE:

1. PUF responses are noisy. Some processing has to be employed to remove the noise.

2. PUF responses are not generally uniformly random. Hence, a processing is required to transform the possibly unknown distribution of the physical measurements into a uniform distributed response value.
## FE in Secure Key Generation

### Enrollment phase
Generating activation code

- $K \leftarrow \text{PUF}(x)$
- $C \leftarrow \text{ECC-Encoding}(r)$, $r$ is random
- $h \leftarrow K + C$

### Reconstruction phase

- $K' \leftarrow \text{PUF}(x)$
- $C' \leftarrow K' + h$
- $C \leftarrow \text{ECC-Decoding}(C')$
- $K \leftarrow C + h$
The action of eliminating noise found in the measurements is called Information Reconciliation.

Mapping the response $X$ onto an element $W$ of a set of Helper Data.

Helper Data has the following properties:

1. With a deterministic recovery function $\text{Rec}$, $X$ can be recovered by means of $W$ and a noisy response of the response $X'$.

2. The amount of uncertainty about an unknown response $X$ only decreases with a limited amount when $W$ for that response is known. This loss of useful uncertainty is unavoidable.

For the Helper Data $W$, the use of error correcting codes (ECC) is one of the solutions.
Privacy Amplification

- Besides noise removal, to guarantee a uniform distribution of the derived keys is another important issue and called as Privacy Amplification.

- Approximation of the distribution is obtained by collecting a lot of samples from different PUFs.

- There is a concept Strong Extractor:

  $$\text{Ext} : \{0, 1\}^n \rightarrow \{0, 1\}^l$$

  It can transform a random variable $X$, with an unknown distribution, into a new random variable $K$, with a distribution close to uniform.

  - Strong extractors can be implemented as pairwise independent universal hash functions.
  - A non-uniform selection of a hash function or keep using the same hash function for all the time gives only a minor deviation from uniformity in the distribution $K$. 
A class of cyclic error correcting code
BCH Code: Bose-Chaudhuri-Hoequenghem

- A class of cyclic error correcting code
- It is constructed in Finite Fields
A class of cyclic error correcting code

It is constructed in Finite Fields

It is a generalized parity code.

BCH \((N,K,t)\) code can correct \(t\) bit errors

\[ t = \left(\frac{N}{K}\right)^m - 1 \]

where \(m\) is an integer such that \(N = 2^m - 1\).
BCH Code: Bose-Chaudhuri-Hocequenghem

- A class of cyclic error correcting code
- It is constructed in Finite Fields
- It is a generalized parity code.
- Binary BCH codes are more popular
BCH Code: Bose-Chaudhuri-Hoquenghem

- A class of cyclic error correcting code
- It is constructed in Finite Fields
- It is a generalized parity code.
- Binary BCH codes are more popular
- Reed Solomon codes (RS) are a class of non-binary BCH codes
BCH Code: Bose-Chaudhuri-Hocquenghem

- A class of cyclic error correcting code
- It is constructed in Finite Fields
- It is a generalized parity code.
- Binary BCH codes are more popular
- Reed Solomon codes (RS) are a class of non-binary BCH codes
- Some special types of BCH codes are called as Golay code: $(23, 12), (31, 16)$, etc.
BCH Code: Bose-Chaudhuri-Hoequenghem

- A class of cyclic error correcting code
- It is constructed in Finite Fields
- It is a generalized parity code.
- Binary BCH codes are more popular
- Reed Solomon codes (RS) are a class of non-binary BCH codes
- Some special types of BCH codes are called as Golay code: (23, 12), (31, 16), etc.
- BCH \((N, K, t)\) code can correct \(t\) bit errors

\[
t = \frac{(N - K)}{m}
\]  

(1)

where \(m\) is an integer such that \(N = 2^m - 1\).
Construction of \((N, K, t)\) BCH Code

- \(\alpha\) is the primitive element in GF\((p^m)\)
Construction of \((N, K, t)\) BCH Code

- \(\alpha\) is the primitive element in \(\text{GF}(p^m)\)
- \(N\) is the block length: \(N = 2^m - 1\)
- \(K\) is the message length
BCH Code

Construction of \((N, K, t)\) BCH Code
- \(\alpha\) is the primitive element in \(\text{GF}(p^m)\)
- \(N\) is the block length: \(N = 2^m - 1\)
- \(K\) is the message length
- \(u(X)\) is the message polynomial where
  \[u(X) = u_{K-1}X^{K-1} + \cdots + u_1X + u_0\]
BCH Code

Construction of \((N, K, t)\) BCH Code
- \(\alpha\) is the primitive element in \(GF(p^m)\)
- \(N\) is the block length: \(N = 2^m - 1\)
- \(K\) is the message length
- \(u(X)\) is the message polynomial where \(u(X) = u_{K-1}X^{K-1} + \cdots + u_1X + u_0\)
- it is capable of correcting \(t\) errors
Construction of \((N, K, t)\) BCH Code

- \(\alpha\) is the primitive element in \(\text{GF}(p^m)\)
- \(N\) is the block length: \(N = 2^m - 1\)
- \(K\) is the message length
- \(u(X)\) is the message polynomial where
  \[u(X) = u_{K-1}X^{K-1} + \cdots + u_1X + u_0\]
- it is capable of correcting \(t\) errors
- bit length of the redundancy is \(N - K\)
BCH Code

Construction of \((N, K, t)\) BCH Code

- \(\alpha\) is the primitive element in \(\text{GF}(p^m)\)
- \(N\) is the block length: \(N = 2^m - 1\)
- \(K\) is the message length
- \(u(X)\) is the message polynomial where
  \[ u(X) = u_{K-1}X^{K-1} + \cdots + u_1X + u_0 \]
- it is capable of correcting \(t\) errors
- bit length of the redundancy is \(N - K\)
- \(N - K \leq mt\)
- \(d_{min} \geq 2t + 1\)
BCH Code

Construction of $(N, K, t)$ BCH Code

- $\alpha$ is the primitive element in $\text{GF}(p^m)$
- $N$ is the block length: $N = 2^m - 1$
- $K$ is the message length
- $u(X)$ is the message polynomial where $u(X) = u_{K-1}X^{K-1} + \cdots + u_1X + u_0$
- it is capable of correcting $t$ errors
- bit length of the redundancy is $N - K$
- $N - K \leq mt$
- $d_{min} \geq 2t + 1$
- $g(X)$ is the generator polynomial and it is computed by LCM of minimal polynomials

$$ g(X) = \text{LCM} \{ \Lambda_1(X), \cdots, \Lambda_{2t}(X) \} $$

$$ g(X) = g_{N-K}X^{N-K} + \cdots + g_1X + g_0 $$

where $\Lambda_i(X)$ is the minimal polynomials of $\alpha^i$, and $\Lambda_i(X) = \Lambda_{i \times 2}(X)$
Galois Field

- \( \text{GF}(p) \)
  - Elements: \( \{0, 1, 2, \ldots, p - 1\} \)
  - \( p \) is prime and operation is “modulo-\( p \)” addition and multiplication
Galois Field

- \( \mathbb{GF}(p) \)
  - Elements: \( \{0, 1, 2, \cdots, p - 1\} \)
  - \( p \) is prime and operation is “modulo-\( p \)” addition and multiplication

- \( \mathbb{GF}(p^m) \)
  - Primitive polynomial of degree \( m \) over \( \mathbb{GF}(p) \) with a root \( \alpha \)

\[
p(X) = X^m + p_{m-1}X^{m-1} + \cdots + p_1X + p_0
\]
Galois Field

- **GF\((p)\)**
  - Elements: \(\{0, 1, 2, \cdots, p - 1\}\)
  - \(p\) is prime and operation is “modulo-\(p\)” addition and multiplication

- **GF\((p^m)\)**
  - Primitive polynomial of degree \(m\) over \(GF(p)\) with a root \(\alpha\)
    
    \[
p(X) = X^m + p_{m-1}X^{m-1} + \cdots + p_1X + p_0\]
  
  - Elements: \(\{0, 1, \alpha, \alpha^2 \cdots, \alpha^{p^m-2}\}\)
Galois Field

- **GF(p)**
  - Elements: \( \{0, 1, 2, \cdots, p - 1\} \)
  - \( p \) is prime and operation is “modulo-\( p \)” addition and multiplication

- **GF(p^m)**
  - Primitive polynomial of degree \( m \) over \( GF(p) \) with a root \( \alpha \)
    \[
p(X) = X^m + p_{m-1}X^{m-1} + \cdots + p_1X + p_0
    \]
  - Elements: \( \{0, 1, \alpha, \alpha^2 \cdots, \alpha^{p^m-2}\} \)
  - \( p \) is prime and operation is “modulo-\( p \)” addition and multiplication
**Galois Field**

- **$\mathbb{GF}(p)$**
  - Elements: $\{0, 1, 2, \cdots, p - 1\}$
  - $p$ is prime and operation is "modulo-$p$" addition and multiplication

- **$\mathbb{GF}(p^m)$**
  - Primitive polynomial of degree $m$ over $\mathbb{GF}(p)$ with a root $\alpha$
    \[p(X) = X^m + p_{m-1}X^{m-1} + \cdots + p_1X + p_0\]
  - Elements: $\{0, 1, \alpha, \alpha^2, \cdots, \alpha^{p^m-2}\}$
  - $p$ is prime and operation is "modulo-$p$" addition and multiplication
  - Construction
    1. $f(X) = q(X)p(X) + r(X)$, where $\text{deg } r < \text{deg } p$.
    2. $\alpha^i = \left(r_{i,m-1}, \cdots, r_{i,1}, r_{i,0}\right)$
### Enumeration of the elements of $\text{GF}(2^4)$

- **$\text{GF}(2^4)$ and $p(X) = X^4 + X + 1$**

<table>
<thead>
<tr>
<th>Power representation</th>
<th>Polynomial representation</th>
<th>Binary representation</th>
<th>Decimal representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0000</td>
<td>0</td>
</tr>
<tr>
<td>$\alpha^0$</td>
<td>1</td>
<td>0001</td>
<td>1</td>
</tr>
<tr>
<td>$\alpha^1$</td>
<td>$\alpha$</td>
<td>0010</td>
<td>2</td>
</tr>
<tr>
<td>$\alpha^2$</td>
<td>$\alpha^2$</td>
<td>0100</td>
<td>4</td>
</tr>
<tr>
<td>$\alpha^3$</td>
<td>$\alpha^3$</td>
<td>1000</td>
<td>8</td>
</tr>
<tr>
<td>$\alpha^4$</td>
<td>$\alpha + 1$</td>
<td>0011</td>
<td>3</td>
</tr>
<tr>
<td>$\alpha^5$</td>
<td>$\alpha^2 + \alpha$</td>
<td>0110</td>
<td>6</td>
</tr>
<tr>
<td>$\alpha^6$</td>
<td>$\alpha^3 + \alpha^2$</td>
<td>1100</td>
<td>12</td>
</tr>
<tr>
<td>$\alpha^7$</td>
<td>$\alpha^3 + \alpha + 1$</td>
<td>1011</td>
<td>11</td>
</tr>
<tr>
<td>$\alpha^8$</td>
<td>$\alpha^2 + 1$</td>
<td>0101</td>
<td>5</td>
</tr>
<tr>
<td>$\alpha^9$</td>
<td>$\alpha^3 + \alpha$</td>
<td>1010</td>
<td>10</td>
</tr>
<tr>
<td>$\alpha^{10}$</td>
<td>$\alpha^2 + \alpha + 1$</td>
<td>0111</td>
<td>7</td>
</tr>
<tr>
<td>$\alpha^{11}$</td>
<td>$\alpha^3 + \alpha^2 + \alpha$</td>
<td>1110</td>
<td>14</td>
</tr>
<tr>
<td>$\alpha^{12}$</td>
<td>$\alpha^3 + \alpha^2 + \alpha + 1$</td>
<td>1111</td>
<td>15</td>
</tr>
<tr>
<td>$\alpha^{13}$</td>
<td>$\alpha^3 + \alpha^2 + 1$</td>
<td>1101</td>
<td>13</td>
</tr>
<tr>
<td>$\alpha^{14}$</td>
<td>$\alpha^3 + 1$</td>
<td>1001</td>
<td>9</td>
</tr>
<tr>
<td>$\alpha^{15}$</td>
<td>1</td>
<td>0001</td>
<td>1</td>
</tr>
</tbody>
</table>
Galois Field

- Minimal polynomial of $\alpha$

$$\Lambda(X) = \lambda_d X^d + \cdots + \lambda_1 X + \lambda_0$$

where $\lambda_i \in \text{GF}(p)$ and $d$ is the least integer such that $\Lambda(\alpha)|_{\alpha \neq 0} = 0$
Galois Field

- Minimal polynomial of $\alpha$

\[ \Lambda(X) = \lambda_d X^d + \cdots + \lambda_1 X + \lambda_0 \]

where $\lambda_i \in \text{GF}(p)$ and $d$ is the least integer such that $\Lambda(\alpha)|_{\alpha \neq 0} = 0$

- Conjugates of $\alpha$ in $\text{GF}(p^m)$

\[ \alpha, \alpha^p, \alpha^{p^2}, \cdots, \alpha^{p^m} \]

It is due to: $\Lambda(\alpha) = \Lambda(\alpha^p) = \Lambda(\alpha^{p^2}) = \cdots = \Lambda(\alpha^{p^m}) = 0$
Galois Field

- Minimal polynomial of $\alpha$

  \[ \Lambda(X) = \lambda_d X^d + \cdots + \lambda_1 X + \lambda_0 \]

  where $\lambda_i \in \mathbb{GF}(p)$ and $d$ is the least integer such that $\Lambda(\alpha)|_{\alpha \neq 0} = 0$

- Conjugates of $\alpha$ in $\mathbb{GF}(p^m)$

  $\alpha, \alpha^p, \alpha^{p^2}, \cdots, \alpha^{p^m}$

  It is due to: $\Lambda(\alpha) = \Lambda(\alpha^p) = \Lambda(\alpha^{p^2}) = \cdots = \Lambda(\alpha^{p^m}) = 0$

- Example $\alpha$ in $\mathbb{GF}(2^4)$ and $p(X) = X^4 + X + 1$
  1. $\{\alpha, \alpha^2, \alpha^4, \alpha^8\}$, where min poly is $X^4 + X + 1$
  2. $\{\alpha^3, \alpha^6, \alpha^9, \alpha^{12}\}$, $\rightarrow X^4 + X^3 + X^2 + X + 1$
  3. $\{\alpha^5, \alpha^{10}\}$, $\rightarrow X^2 + X + 1$
  4. $\{\alpha^7, \alpha^{11}, \alpha^{13}, \alpha^{14}\}$, $\rightarrow X^4 + X^3 + 1$
BCH Code: $g(X)$

- Minimal polynomial sets: $\alpha$ in $\text{GF}(2^4)$ and $p(X) = X^4 + X + 1$

1. $\{\alpha, \alpha^2, \alpha^4, \alpha^8\}$, where min poly is $X^4 + X + 1$
2. $\{\alpha^3, \alpha^6, \alpha^9, \alpha^{12}\}$, $\rightarrow X^4 + X^3 + X^2 + X + 1$
3. $\{\alpha^5, \alpha^{10}\}$, $\rightarrow X^2 + X + 1$
4. $\{\alpha^7, \alpha^{11}, \alpha^{13}, \alpha^{14}\}$, $\rightarrow X^4 + X^3 + 1$
BCH Code: \( g(X) \)

- Minimal polynomial sets: \( \alpha \) in \( GF(2^4) \) and \( p(X) = X^4 + X + 1 \)
  1. \( \{ \alpha, \alpha^2, \alpha^4, \alpha^8 \} \), where min poly is \( X^4 + X + 1 \)
  2. \( \{ \alpha^3, \alpha^6, \alpha^9, \alpha^{12} \} \), \( \rightarrow X^4 + X^3 + X^2 + X + 1 \)
  3. \( \{ \alpha^5, \alpha^{10} \} \), \( \rightarrow X^2 + X + 1 \)
  4. \( \{ \alpha^7, \alpha^{11}, \alpha^{13}, \alpha^{14} \} \), \( \rightarrow X^4 + X^3 + 1 \)

- When \( t = 2 \),

\[
g(X) = \text{LCM} \{ \Lambda_1(X), \Lambda_3(X) \}
\]
BCH Code: $g(X)$

- **Minimal polynomial sets:** $\alpha$ in $\text{GF}(2^4)$ and $p(X) = X^4 + X + 1$
  1. $\{\alpha, \alpha^2, \alpha^4, \alpha^8\}$, where min poly is $X^4 + X + 1$
  2. $\{\alpha^3, \alpha^6, \alpha^9, \alpha^{12}\}$, $\rightarrow X^4 + X^3 + X^2 + X + 1$
  3. $\{\alpha^5, \alpha^{10}\}$, $\rightarrow X^2 + X + 1$
  4. $\{\alpha^7, \alpha^{11}, \alpha^{13}, \alpha^{14}\}$, $\rightarrow X^4 + X^3 + 1$

- **When $t = 2$,**

  $$g(X) = \text{LCM} \{\Lambda_1(X), \Lambda_3(X)\}$$

- **When $t = 3$,**

  $$g(X) = \text{LCM} \{\Lambda_1(X), \Lambda_3(X), \Lambda_5(X)\}$$
(N, K, t) BCH Code

- \( c(X) = u(X) \cdot g(X) \)
- \( X^{N-K}u(X) = q(X) \cdot g(X) + r(X) \)
- \( c(X) = X^{N-K}u(X) + r(X) \)
(\(N, K, t\)) \(\text{BCH Code}\)

\[
\begin{align*}
  c(X) &= u(X) \cdot g(X) \\
  X^{N-K}u(X) &= q(X) \cdot g(X) + r(X) \\
  c(X) &= X^{N-K}u(X) + r(X)
\end{align*}
\]

Encoding as simple as computing polynomial modulo

\[
r(X) = X^{N-K}u(X) \mod g(X)
\]
(15, 5, 3) BCH Code in GF(2^4) and \( p(X) = X^4 + X + 1 \)
BCH Code Encoding: An Example

- (15, 5, 3) BCH Code in GF($2^4$) and $p(X) = X^4 + X + 1$
- Message is (0, 1, 1, 0, 1) where $u(X) = X^4 + X^2 + X$
BCH Code Encoding: An Example

- \((15, 5, 3)\) BCH Code in \(\text{GF}(2^4)\) and \(p(X) = X^4 + X + 1\)
  - Message is \((0, 1, 1, 0, 1)\) where \(u(X) = X^4 + X^2 + X\)
  - \(N = 15, K = 5, t = 3\)
(15, 5, 3) BCH Code in $\mathbb{GF}(2^4)$ and $p(X) = X^4 + X + 1$

- Message is $(0, 1, 1, 0, 1)$ where $u(X) = X^4 + X^2 + X$
- $N = 15$, $K = 5$, $t = 3$
- $15 - 5 \leq 4 \times 3$
BCH Code Encoding: An Example

- **(15, 5, 3) BCH Code in GF(2^4) and p(X) = X^4 + X + 1**
  - Message is (0, 1, 1, 0, 1) where \( u(X) = X^4 + X^2 + X \)
  - \( N = 15, \ K = 5, \ t = 3 \)
  - \( 15 - 5 \leq 4 \times 3 \)
  - \( d_{min} \geq 2 \times 3 + 1 \)
(15, 5, 3) BCH Code in GF($2^4$) and $p(X) = X^4 + X + 1$
- Message is (0, 1, 1, 0, 1) where $u(X) = X^4 + X^2 + X$
- $N = 15$, $K = 5$, $t = 3$
- $15 - 5 \leq 4 \times 3$
- $d_{min} \geq 2 \times 3 + 1$

$g(X) = \text{LCM} \{\Lambda_1(X), \Lambda_3(X), \Lambda_5(X)\}$
BCH Code Encoding: An Example

(15, 5, 3) BCH Code in GF(2^4) and \( p(X) = X^4 + X + 1 \)
- Message is (0, 1, 1, 0, 1) where \( u(X) = X^4 + X^2 + X \)
- \( N = 15, \ K = 5, \ t = 3 \)
- \( 15 - 5 \leq 4 \times 3 \)
- \( d_{min} \geq 2 \times 3 + 1 \)

\[
g(X) = \text{LCM}\{\Lambda_1(X), \Lambda_3(X), \Lambda_5(X)\}
= (X^4 + X + 1) \times (X^4 + X^3 + X^2 + X + 1) \times (X^2 + X + 1)
\]
BCH Code Encoding: An Example

- \((15, 5, 3)\) BCH Code in \(\text{GF}(2^4)\) and \(p(X) = X^4 + X + 1\)
  - Message is \((0, 1, 1, 0, 1)\) where \(u(X) = X^4 + X^2 + X\)
  - \(N = 15, K = 5, t = 3\)
  - \(15 - 5 \leq 4 \times 3\)
  - \(d_{\text{min}} \geq 2 \times 3 + 1\)

\[
g(X) = \text{LCM} \{\Lambda_1(X), \Lambda_3(X), \Lambda_5(X)\}
\]
\[
= (X^4 + X + 1) \times (X^4 + X^3 + X^2 + X + 1) \times (X^2 + X + 1)
\]
\[
= (X^{10} + X^8 + X^5 + X^4 + X^2 + X + 1)
\]
BCH Code Encoding: An Example

- \((15, 5, 3)\) BCH Code in \(GF(2^4)\) and \(p(X) = X^4 + X + 1\)
  - Message is \((0, 1, 1, 0, 1)\) where \(u(X) = X^4 + X^2 + X\)
  - \(N = 15, K = 5, t = 3\)
  - \(15 - 5 \leq 4 \times 3\)
  - \(d_{\text{min}} \geq 2 \times 3 + 1\)

\[
g(X) = \text{LCM} \{\Lambda_1(X), \Lambda_3(X), \Lambda_5(X)\}
\]
\[
= (X^4 + X + 1) \times (X^4 + X^3 + X^2 + X + 1) \times (X^2 + X + 1)
\]
\[
= (X^{10} + X^8 + X^5 + X^4 + X^2 + X + 1)
\]

\[
P(X) = X^{15-5} u(X) = (X^{14} + X^{12} + X^{11})
\]
BCH Code Encoding: An Example

- (15, 5, 3) BCH Code in GF($2^4$) and $p(X) = X^4 + X + 1$
  - Message is (0, 1, 1, 0, 1) where $u(X) = X^4 + X^2 + X$
  - $N = 15$, $K = 5$, $t = 3$
  - $15 - 5 \leq 4 \times 3$
  - $d_{min} \geq 2 \times 3 + 1$

  $$g(X) = \text{LCM} \{\Lambda_1(X), \Lambda_3(X), \Lambda_5(X)\}$$
  $$= (X^4 + X + 1) \times (X^4 + X^3 + X^2 + X + 1) \times (X^2 + X + 1)$$
  $$= (X^{10} + X^8 + X^5 + X^4 + X^2 + X + 1)$$

  $$P(X) = X^{15-5}u(X) = (X^{14} + X^{12} + X^{11})$$

  $$r(X) = P(X) \mod g(X)$$
BCH Code Encoding: An Example

- \((15, 5, 3)\) BCH Code in \(\text{GF}(2^4)\) and \(p(X) = X^4 + X + 1\)
  - Message is \((0, 1, 1, 0, 1)\) where \(u(X) = X^4 + X^2 + X\)
  - \(N = 15\), \(K = 5\), \(t = 3\)
  - \(15 - 5 \leq 4 \times 3\)
  - \(d_{\text{min}} \geq 2 \times 3 + 1\)

\[
g(X) = \text{LCM}\{\Lambda_1(X), \Lambda_3(X), \Lambda_5(X)\}
\]
\[
= (X^4 + X + 1) \times (X^4 + X^3 + X^2 + X + 1) \times (X^2 + X + 1)
\]
\[
= (X^{10} + X^8 + X^5 + X^4 + X^2 + X + 1)
\]

\[
P(X) = X^{15-5}u(X) = (X^{14} + X^{12} + X^{11})
\]

\[
r(X) = P(X) \mod g(X)
\]
\[
= X^8 + X^4 + X^3 + X^2 + X
\]
BCH Code Encoding: An Example

1. **(15, 5, 3)** BCH Code in $\mathbb{GF}(2^4)$ and $p(X) = X^4 + X + 1$
   - Message is $(0, 1, 1, 0, 1)$ where $u(X) = X^4 + X^2 + X$
   - $N = 15$, $K = 5$, $t = 3$
   - $15 - 5 \leq 4 \times 3$
   - $d_{min} \geq 2 \times 3 + 1$

   \[ g(X) = \text{LCM} \{\Lambda_1(X), \Lambda_3(X), \Lambda_5(X)\} \]
   \[ = (X^4 + X + 1) \times (X^4 + X^3 + X^2 + X + 1) \times (X^2 + X + 1) \]
   \[ = (X^{10} + X^8 + X^5 + X^4 + X^2 + X + 1) \]

   \[ P(X) = X^{15-5}u(X) = (X^{14} + X^{12} + X^{11}) \]

   \[ r(X) = P(X) \mod g(X) \]
   \[ = X^8 + X^4 + X^3 + X^2 + X \]

   \[ c(X) = P(X) + r(X) \]
BCH Code Encoding: An Example

- \((15, 5, 3)\) BCH Code in \(\text{GF}(2^4)\) and \(p(X) = X^4 + X + 1\)
  - Message is \((0, 1, 1, 0, 1)\) where \(u(X) = X^4 + X^2 + X\)
  - \(N = 15, \; K = 5, \; t = 3\)
  - \(15 - 5 \leq 4 \times 3\)
  - \(d_{\text{min}} \geq 2 \times 3 + 1\)

\[
g(X) = \text{LCM} \{\Lambda_1(X), \Lambda_3(X), \Lambda_5(X)\} \\
= (X^4 + X + 1) \times (X^4 + X^3 + X^2 + X + 1) \times (X^2 + X + 1) \\
= (X^{10} + X^8 + X^5 + X^4 + X^2 + X + 1)
\]

\[
P(X) = X^{15-5}u(X) = (X^{14} + X^{12} + X^{11})
\]

\[
r(X) = P(X) \mod g(X) \\
= X^8 + X^4 + X^3 + X^2 + X
\]

\[
c(X) = P(X) + r(X) \\
= X^{14} + X^{12} + X^{11} + X^8 + X^4 + X^3 + X^2 + X
\]
BCH Code Encoding: An Example

- **(15, 5, 3)** BCH Code in \(GF(2^4)\) and \(p(X) = X^4 + X + 1\)
  - Message is \((0, 1, 1, 0, 1)\) where \(u(X) = X^4 + X^2 + X\)
  - \(N = 15, K = 5, t = 3\)
  - \(15 - 5 \leq 4 \times 3\)
  - \(d_{min} \geq 2 \times 3 + 1\)

\[
g(X) = \text{LCM} \{\Lambda_1(X), \Lambda_3(X), \Lambda_5(X)\}
\]

\[
= (X^4 + X + 1) \times (X^4 + X^3 + X^2 + X + 1) \times (X^2 + X + 1)
\]

\[
= (X^{10} + X^8 + X^5 + X^4 + X^2 + X + 1)
\]

\[
P(X) = X^{15-5}u(X) = (X^{14} + X^{12} + X^{11})
\]

\[
r(X) = P(X) \mod g(X)
\]

\[
= X^8 + X^4 + X^3 + X^2 + X
\]

\[
c(X) = P(X) + r(X)
\]

\[
= X^{14} + X^{12} + X^{11} + X^8 + X^4 + X^3 + X^2 + X
\]

- **Codeword:**
  - \(c = (0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1)\)
\[ c(\alpha^i) = c_{N-1}\alpha^{N-1}i + \cdots + c_1\alpha^i + c_0 = 0 \text{ for } 1 \leq i \leq 2t \]

It is because

- \( g(X) \mid c(X) \)
- \( g(\alpha) = g(\alpha^2) = \cdots = g(\alpha^{2t}) = 0 \)

Then, parity check matrix: \( cH^T = 0 \)

\[
H = \begin{bmatrix}
1 & \alpha & \alpha^2 & \cdots & \alpha^{N-1} \\
1 & \alpha^2 & (\alpha^2)^2 & \cdots & (\alpha^2)^{N-1} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
1 & \alpha^{2t} & (\alpha^{2t})^2 & \cdots & (\alpha^{2t})^{N-1}
\end{bmatrix}
\]
Syndrome and Error Location

- Received data: \( r = c + e \)
- The noise: \( e = (e_0, e_1, \cdots, e_{N-1}) \) where \( e_j \in \{0, 1\} \)
- Computation of syndrome:

\[
S = rH^T = (c + e)H^T \\
= eH^T = (S_1, S_2, \cdots, S_{2t}) \\
S_i = \sum_{j=0}^{N-1} e_j \alpha^{ij}, \text{ for } i = 1, 2, \cdots, 2t \\
= \boxed{e(\alpha^i)} \text{ because } \\
e(X) = e_{N-1}X^{N-1} + \cdots + e_1X + e_0
\]

- Where \( v \leq t \), \( v \) errors exists at \( j_1, j_2, \cdots, j_v \)
(\( 0 \leq j_1 < j_2 < \cdots < j_v < N \))

\[
e(X) = X^{j_1} + X^{j_2} + \cdots + X^{j_v}
\]
\[ S_i = e(\alpha^i) = (\alpha^{j_1})^i + (\alpha^{j_2})^i + \cdots + (\alpha^{j_v})^i, \quad \text{for } i = 1, 2, \cdots, 2t \]

- Solving \(2t\) equations will give the set \((\alpha^{j_1}, \alpha^{j_2}, \cdots, \alpha^{j_v})\)
- Error locations are \((j_1, j_2, \cdots, j_v)\)

- Location numbers of error is denoted as \(\beta_l = \alpha^{j_l}\)
- Syndrome is also computed with respect to \(\beta_l\)s

\[
S_1 = \beta_1 + \beta_2 + \cdots + \beta_v \\
S_2 = \beta_1^2 + \beta_2^2 + \cdots + \beta_v^2 \\
\vdots \\
S_{2t} = \beta_1^{2t} + \beta_2^{2t} + \cdots + \beta_v^{2t}
\]
Error Locator Polynomial

- Error locator polynomial is denoted with $\sigma(X)$ and

$$\sigma(X) = (1 + \beta_1 X)(1 + \beta_2 X) \cdots (1 + \beta_v X) = \sigma_v X^v + \cdots + \sigma_1 X + \sigma_0$$

- Roots of $\sigma(X)$ is $(\beta_1^{-1}, \beta_2^{-1}, \cdots, \beta_v^{-1})$

Finding the roots of $\sigma(X)$ will reveal the error locations

- Error locations are simply the inverse of the roots of $\sigma(X)$
1 Syndrome computation
2 Computing the error locator polynomial $\sigma(X)$
   - Peterson-Gorenstein-Zierler algorithm
   - Berlekamp-Massey algorithm
   - Euclid’s algorithm
3 Finding error location: Chien search for $u$ roots
4 Correcting the errors
   - Decoding result is simply $(r - e)$
Peterson-Gorenstein-Zierler algorithm

1. Initialize $v = t$
Peterson-Gorenstein-Zierler algorithm

1. Initialize $v = t$
2. Compute the determinant of $M$

$$\text{det}(M) = \begin{bmatrix} S_1 & S_2 & \cdots & S_v \\ S_2 & S_3 & \cdots & S_{v+1} \\ \vdots & \vdots & \ddots & \vdots \\ S_v & S_{v+1} & \cdots & S_{2v-1} \end{bmatrix}$$
Peterson-Gorenstein-Zierler algorithm

1. Initialize $v = t$
2. Compute the determinant of $M$

$$\det(M) = \begin{bmatrix} S_1 & S_2 & \cdots & S_v \\ S_2 & S_3 & \cdots & S_{v+1} \\ \vdots & \vdots & \ddots & \vdots \\ S_v & S_{v+1} & \cdots & S_{2v-1} \end{bmatrix}$$

3. Find the correct value of $v$: it may less than $t$

$$\begin{cases} 
\text{go to step 4} & \det(M) \neq 0 \\
 v \leftarrow v - 1, \text{ and then go to step 2} & \det(M) = 0
\end{cases}$$
**Peterson-Gorenstein-Zierler algorithm**

1. **Initialize** \( v = t \)
2. **Compute the determinant of** \( M \)

\[
\begin{vmatrix}
S_1 & S_2 & \cdots & S_v \\ 
S_2 & S_3 & \cdots & S_{v+1} \\ 
\vdots & \vdots & \ddots & \vdots \\ 
S_v & S_{v+1} & \cdots & S_{2v-1}
\end{vmatrix}
\]

3. **Find the correct value of** \( v \): it may less than \( t \)

\[
\begin{cases}
\text{go to step 4} & \text{det}(M) \neq 0 \\
\quad v \leftarrow v - 1, \text{ and then go to step 2} & \text{det}(M) = 0
\end{cases}
\]

4. **Invert** \( M \)
**Peterson-Gorenstein-Zierler algorithm**

1. Initialize \( v = t \)
2. Compute the determinant of \( \mathbf{M} \)

\[
\det(\mathbf{M}) = \begin{bmatrix}
S_1 & S_2 & \cdots & S_v \\
S_2 & S_3 & \cdots & S_{v+1} \\
\vdots & \vdots & \ddots & \vdots \\
S_v & S_{v+1} & \cdots & S_{2v-1}
\end{bmatrix}
\]

3. Find the correct value of \( v \): it may less than \( t \)

\[
\begin{cases}
\text{go to step 4} & \text{if } \det(\mathbf{M}) \neq 0 \\
v \leftarrow v - 1, \text{ and then go to step 2} & \text{if } \det(\mathbf{M}) = 0
\end{cases}
\]

4. Invert \( \mathbf{M} \)
5. Compute \( \sigma(X) \) by simply multiplying \( \mathbf{M}^{-1} \) by a syndrome vector

\[
\begin{bmatrix}
\sigma_v \\
\sigma_{v-1} \\
\vdots \\
\sigma_1
\end{bmatrix} = \mathbf{M}^{-1} \begin{bmatrix}
-S_{v+1} \\
-S_{v+2} \\
\vdots \\
-S_{2v}
\end{bmatrix}
\]
\( (15, 5, 3) \) BCH Code in \( \text{GF}(2^4) \) and \( p(X) = X^4 + X + 1 \)

- \( g(X) = \text{LCM} \{ \Lambda_1(X), \Lambda_3(X), \Lambda_5(X) \} \)
  \[ = X^{10} + X^8 + X^5 + X^4 + X^2 + X + 1 \]
- \( ^*c(X) = X^{14} + X^{12} + X^{11} + X^8 + X^4 + X^3 + X^2 + X \)
- \( e(X) = X^{14} + X^3 \)

Then, received codeword is \( r(X) = X^{12} + X^{11} + X^8 + X^4 + X^2 + X \)

*Same codeword as in Encoding example*
(15, 5, 3) BCH Code in GF($2^4$) and $p(X) = X^4 + X + 1$

- $g(X) = \text{LCM} \{\Lambda_1(X), \Lambda_3(X), \Lambda_5(X)\}$
  $$= X^{10} + X^8 + X^5 + X^4 + X^2 + X + 1$$
- $^*c(X) = X^{14} + X^{12} + X^{11} + X^8 + X^4 + X^3 + X^2 + X$
- $e(X) = X^{14} + X^3$
- Then, received codeword is $r(X) = X^{12} + X^{11} + X^8 + X^4 + X^2 + X$

First, computing the syndrome

$$S_1 = r(\alpha) = 1 \quad S_2 = r(\alpha^2) = 1 \quad S_3 = r(\alpha^3) = \alpha^8$$
$$S_4 = r(\alpha^4) = 1 \quad S_5 = r(\alpha^5) = \alpha^5 \quad S_6 = r(\alpha^6) = \alpha$$

*Same codeword as in Encoding example*
1. $v = t = 3$
1. $v = t = 3$

2. Computing determinant of $M$

$$\det(M) = \begin{pmatrix} S_1 & S_2 & S_3 \\ S_2 & S_3 & S_4 \\ S_3 & S_4 & S_5 \end{pmatrix} = \begin{pmatrix} 1 & 1 & \alpha^8 \\ 1 & \alpha^8 & 1 \\ \alpha^8 & 1 & \alpha^5 \end{pmatrix} = 0$$
① \( v = t = 3 \)

② Computing determinant of \( M \)

\[
\text{det}(M) = \begin{bmatrix} S_1 & S_2 & S_3 \\ S_2 & S_3 & S_4 \\ S_3 & S_4 & S_5 \end{bmatrix} = \begin{bmatrix} 1 & 1 & \alpha^8 \\ 1 & \alpha^8 & 1 \\ \alpha^8 & 1 & \alpha^5 \end{bmatrix} = 0
\]

③ \( \text{det}(M) = 0 \), then go to step 2 with \( v = v - 1 = 2 \)
BCH Code Decoding: An Example

1. \( v = t = 3 \)

2. Computing determinant of \( M \)

\[
\det(M) = \begin{bmatrix} S_1 & S_2 & S_3 \\ S_2 & S_3 & S_4 \\ S_3 & S_4 & S_5 \end{bmatrix} = \begin{bmatrix} 1 & 1 & \alpha^8 \\ 1 & \alpha^8 & 1 \\ \alpha^8 & 1 & \alpha^5 \end{bmatrix} = 0
\]

3. \( \det(M) = 0 \), then go to step 2 with \( v = v - 1 = 2 \)

2. Computing determinant of new \( M \)

\[
\det(M) = \begin{bmatrix} S_1 & S_2 \\ S_2 & S_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & \alpha^8 \end{bmatrix} = \alpha^2
\]

Look at the Table
BCH Code Decoding: An Example

1. \( v = t = 3 \)

2. Computing determinant of \( M \)

   \[
   \text{det}(M) = \begin{bmatrix}
   S_1 & S_2 & S_3 \\
   S_2 & S_3 & S_4 \\
   S_3 & S_4 & S_5 \\
   \end{bmatrix} = \begin{bmatrix}
   1 & 1 & \alpha^8 \\
   1 & \alpha^8 & 1 \\
   \alpha^8 & 1 & \alpha^5 \\
   \end{bmatrix} = 0
   \]

3. \( \text{det}(M) = 0 \), then go to step 2 with \( v = v - 1 = 2 \)

2. Computing determinant of new \( M \)

   \[
   \text{det}(M) = \begin{bmatrix}
   S_1 & S_2 \\
   S_2 & S_3 \\
   \end{bmatrix} = \begin{bmatrix}
   1 & 1 \\
   1 & \alpha^8 \\
   \end{bmatrix} = \alpha^2
   \]

3. \( \text{det}(M) = \alpha^2 \), then go to step 4

Look at the Table
Inverting $\mathbf{M}$

$$\mathbf{M}^{-1} = \begin{bmatrix} \alpha^6 & \alpha^{13} \\ \alpha^{13} & \alpha^{13} \end{bmatrix}$$
BCH Code Decoding: An Example

4 Inverting $M$

$$M^{-1} = \begin{bmatrix} \alpha^6 & \alpha^{13} \\ \alpha^{13} & \alpha^{13} \end{bmatrix}$$

5 Computing $\sigma(X)$

$$\begin{bmatrix} \sigma_2 \\ \sigma_1 \end{bmatrix} = \begin{bmatrix} \alpha^6 & \alpha^{13} \\ \alpha^{13} & \alpha^{13} \end{bmatrix} \begin{bmatrix} \alpha^8 \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha^2 \\ 1 \end{bmatrix}$$

$$\sigma(X) = \alpha^2 X^2 + X + 1$$
BCH Code Decoding: An Example

4 Inverting $M$

$$M^{-1} = \begin{bmatrix} 6 & 13 \\ 13 & 13 \end{bmatrix}$$

5 Computing $\sigma(X)$

$$\begin{bmatrix} \sigma_2 \\ \sigma_1 \end{bmatrix} = \begin{bmatrix} 6 & 13 \\ 13 & 13 \end{bmatrix} \begin{bmatrix} 8 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

$$\sigma(X) = \alpha^2 X^2 + X + 1$$

$\sigma(X)$ will tell you the location of the errors
4 Inverting $M$

$$M^{-1} = \begin{bmatrix} \alpha^6 & \alpha^{13} \\ \alpha^{13} & \alpha^{13} \end{bmatrix}$$

5 Computing $\sigma(X)$

$$\begin{bmatrix} \sigma_2 \\ \sigma_1 \end{bmatrix} = \begin{bmatrix} \alpha^6 & \alpha^{13} \\ \alpha^{13} & \alpha^{13} \end{bmatrix} \begin{bmatrix} \alpha^8 \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha^2 \\ 1 \end{bmatrix}$$

$$\sigma(X) = \alpha^2 X^2 + X + 1$$

- $\sigma(X)$ will tell you the location of the errors
  $$\sigma(\alpha) = 0, \quad \sigma(\alpha^2) = \alpha^{14}, \quad \cdots \quad \sigma(\alpha^{12}) = 0$$
BCH Code Decoding: An Example

4. Inverting $M$

$$M^{-1} = \begin{bmatrix} \alpha^6 & \alpha^{13} \\ \alpha^{13} & \alpha^{13} \end{bmatrix}$$

5. Computing $\sigma(X)$

$$\begin{bmatrix} \sigma_2 \\ \sigma_1 \end{bmatrix} = \begin{bmatrix} \alpha^6 & \alpha^{13} \\ \alpha^{13} & \alpha^{13} \end{bmatrix} \begin{bmatrix} \alpha^8 \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha^2 \\ 1 \end{bmatrix}$$

$$\sigma(X) = \alpha^2 X^2 + X + 1$$

- $\sigma(X)$ will tell you the location of the errors
  $$\sigma(\alpha) = 0, \quad \sigma(\alpha^2) = \alpha^{14}, \quad \cdots \quad \sigma(\alpha^{12}) = 0$$

- After finding the roots, we now compute the inverse of the roots $\alpha$ and $\alpha^{12}$

$$\alpha^{-1} = \alpha^{14}$$

$$\alpha^{-12} = \alpha^{3}$$
Finally, $e(X) = X^{14} + X^3$

Corrected codeword:

$$r(X) + e(X) = X^{14} + X^{12} + X^{11} + X^8 + X^4 + X^3 + X^2 + X$$

$$\bar{c} = (0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1)$$
Security of PUF

- **Entropy**
  - PUF should have sufficient amount of randomness

<table>
<thead>
<tr>
<th>PUF type</th>
<th>Entropy per 1000 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRAM PUF</td>
<td>950</td>
</tr>
<tr>
<td>Delay PUF</td>
<td>130</td>
</tr>
<tr>
<td>Butterfly PUF</td>
<td>600</td>
</tr>
</tbody>
</table>
Security of PUF

- Entropy
  - PUF should have sufficient amount of randomness

<table>
<thead>
<tr>
<th>PUF type</th>
<th>Entropy per 1000 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRAM PUF</td>
<td>950</td>
</tr>
<tr>
<td>Delay PUF</td>
<td>130</td>
</tr>
<tr>
<td>Butterfly PUF</td>
<td>600</td>
</tr>
</tbody>
</table>

- Tamper evidence
  - Read-proof hardware against *invasive* and *non-invasive*
  - Invasive attacks make the chip functionally destroyed
  - Currently, there is no non-invasive attack that gives information about the PUF
Security of PUF

- Entropy
  - PUF should have sufficient amount of randomness

<table>
<thead>
<tr>
<th>PUF type</th>
<th>Entropy per 1000 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRAM PUF</td>
<td>950</td>
</tr>
<tr>
<td>Delay PUF</td>
<td>130</td>
</tr>
<tr>
<td>Butterfly PUF</td>
<td>600</td>
</tr>
</tbody>
</table>

- Tamper evidence
  - Read-proof hardware against *invasive* and *non-invasive*
  - Invasive attacks make the chip functionally destroyed
  - Currently, there is no non-invasive attack that gives information about the PUF

- Unclonability
  - Unique physical property of PUF is translating into a bitstream
  - This physical property prevents constructing a same PUF
  - Keep the challenge response pairs secret
Effects of Environmental Parameters on PUF

- Die temperature and supply voltage
  - Heavily affect the delay in ring oscillator PUFs
  - Less affect the arbiter PU due to the differential measurement
Threats to PUFs

  - delay-based PUF constructions can be modeled using ML algorithms.
  - CRPs should not be made public in protocols.
  - Required a Trust to Manufacturers
Threats to PUFs

  - delay-based PUF constructions can be modeled using ML algorithms.
  - CRPs should not be made public in protocols.
  - Required a Trust to Manufacturers

- Helfmeier *et al.* cloned an SRAM PUF with a failure analysis equipment in at HOST 2013. *Cloning Physically Unclonable Functions*.

- Nedospasov *et al.* described successful invasive attempts on SRAM PUFs at FDTC 2013. *Invasive PUF Analysis*.

- EM analyses on RO PUFs have been carried out by Merli *et al.* at WESS 2011. *Semi-invasive EM attack on FPGA RO PUFs and countermeasures*. 
I would like to acknowledge:

Questions?