

ECE 5990

Note 10

Security and Privacy for RFID Systems

Edwin C. Kan

School of Electrical and Computer Engineering

Cornell University

Fall 2014

Outline

- Detectable and correctable digital codes
- Error-correction code (ECC) – an Example
- Hamming distance and Shannon theory
- Hash function and CRC codes
- Security and privacy in RFID systems
- Circuits implementation

Quotable Quotes

“Anyone who attempts to generate random numbers by deterministic means is, of course, living in a state of sin.”

John von Neumann



*Hardware-oriented
security!!!*

E. C. K.

Errors in the RFID Codes

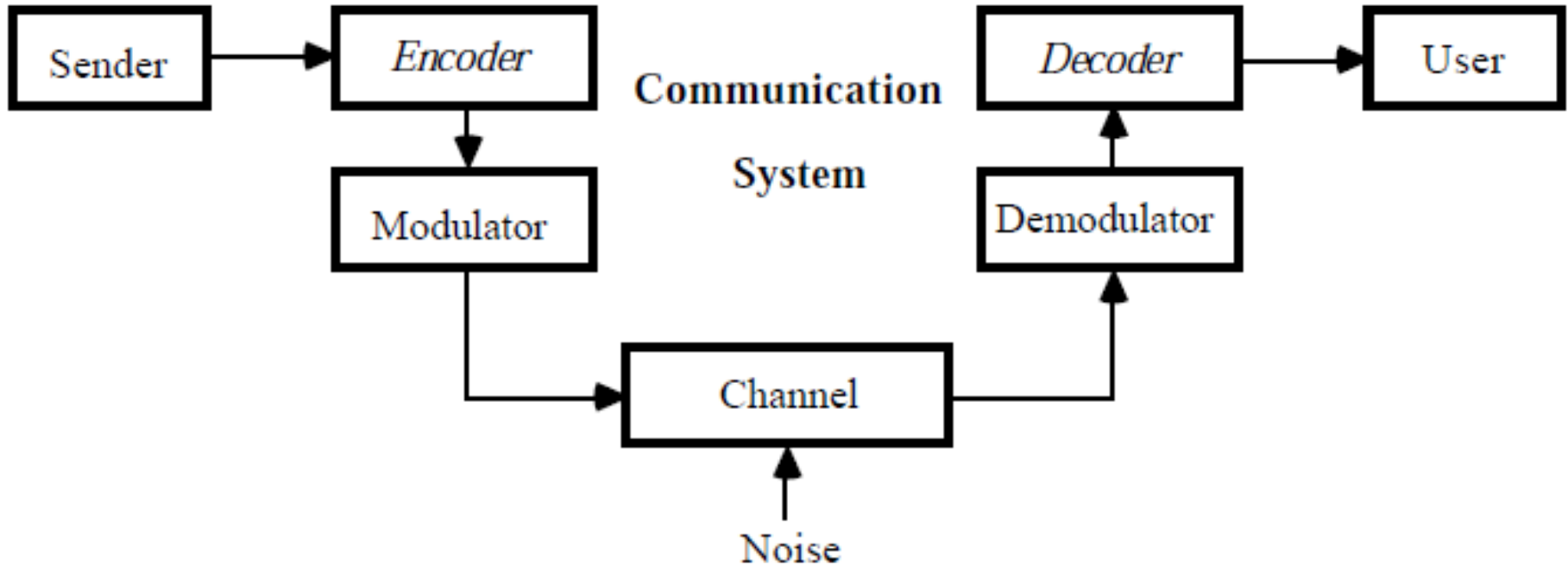
- **Random errors:** the bit error probabilities are independent or nearly independent of each other. Example: thermal noise; interferences.
- **Burst errors:** the bit error occurs sequentially in time or as groups of “stuck-at”. Example: weak signal when the antenna is detuned or scratch in DVD
- **Impulse errors:** large blocks of the data are full of random errors. Example: transmission collision; reader interference.

Error correction \Rightarrow Data Redundancy!!!

Levels of Error Correction Code (ECC)

- Repetition code: based on majority vote
 - Very inefficient in code rate
 - Fast recovery
 - No assumption on the position of bits in each packet
- Forward correction code
 - Bit-level: Hamming code: popular in memory
 - Block code: Reed-Solomon code: popular in serial storage and communication channels
- Automatic repeat request (ARQ)
 - Stop and wait
 - Continuous duplex: popular in multi-level circuits (MLC)

Error Correction System



- Probability of error:
 - probability of uncorrectable errors: P_{UE}
 - probability that the channel will change a symbol during the processing or transmission: P_{SE} .
 - $P_{UE} = P_{SE}$ if there is no error correction system.

Bit-Level Hamming Code Example

	Data Bits				Check Bits		
	A	B	C	D	E	F	G
0	0	0	0	0	0	0	0
1	0	0	0	1	0	1	1
2	0	0	1	0	1	0	1
3	0	0	1	1	1	1	0
4	0	1	0	0	1	1	0
5	0	1	0	1	1	0	1
6	0	1	1	0	0	1	1
7	0	1	1	1	0	0	0
8	1	0	0	0	1	1	1
9	1	0	0	1	1	0	0
10	1	0	1	0	0	1	0
11	1	0	1	1	0	0	1
12	1	1	0	0	0	0	1
13	1	1	0	1	0	1	0
14	1	1	1	0	1	0	0
15	1	1	1	1	1	1	1

b_7 b_6 b_5 b_4 b_3 b_2 b_1

Check-Bit Generation Before and After

- Bit A, B, C, and D: original bits: $2^4 = 16$ possible combinations.
- Bit E, F and G: parity bits: total $2^7 = 128$ possible combinations.

$$E = A \oplus B \oplus C \quad Eq.(1)$$

$$F = A \oplus B \oplus D \quad Eq.(2)$$

$$G = A \oplus C \oplus D \quad Eq.(3)$$

- (7, 4) Hamming code:
- $k = 4$: the uncoded bits in a word
- $n = 7$: total number of bits in a codeword
- $(n - k)$: the parity check bits
- t : the number of bits correctable $\sim (n - k)/2$

Finding the Error-Bit Address

Bit in Error	Eq. 1 $E' = A \oplus B \oplus C$	Eq. 2 $F' = A \oplus B \oplus D$	Eq. 3 $G' = A \oplus C \oplus D$
None	True	True	True
A	False	False	False
B	False	False	True
C	False	True	False
D	True	False	False
E	False	True	True
F	True	False	True
G	True	True	False

The position of error always corresponds to a unique combination in the truth table!!

The error bit address can be uniquely determined and then toggle to correct!!

Group Activity

Split to four groups of roughly equal sizes with tree protocol:
Write down 6 bits of random numbers.

Group 1: For one bit correctable, how many bits need to be added to the 4-bit word in the majority vote scheme?

Group 2: Why is (7,4) code much more efficient than the majority vote?

Group 3: Hard drive (read by a magnetic head) has very high bit error rate (~ 0.05 to 0.1), should we use majority vote or Hamming code

Group 4: DNA may contain error... What does nature do?

Outline

- Detectable and correctable digital codes
- Error-correction code (ECC) – an Example
- **Hamming distance and Shannon theory**
- Hash function and CRC codes
- Security and privacy in RFID systems
- Circuit implementation

Criteria to Generate the Check Bits

“Distance” between legal codes can be increased by check bits!!

	Data Bits				Parity Bit
	A	B	C	D	$A \oplus B \oplus C \oplus D$
					E
0	0	0	0	0	0
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	0
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	1
9	1	0	0	1	0
10	1	0	1	0	0
11	1	0	1	1	1
12	1	1	0	0	0
13	1	1	0	1	1
14	1	1	1	0	1
15	1	1	1	1	0

- Original Word ABCD: each word can be differed by only one bit: if one bit is wrong, it will still be legal.
- If we add just a parity bit, then each word is differed by at least 2 bits! If only one bit is wrong, then it will become an illegal word.
- However, two words with one parity bit can go to the same illegal word, so when we have the wrong word, we cannot distinguish where it is from.

Original Word 1: 00101

Original Word 2: 00110

Received Word: 00100

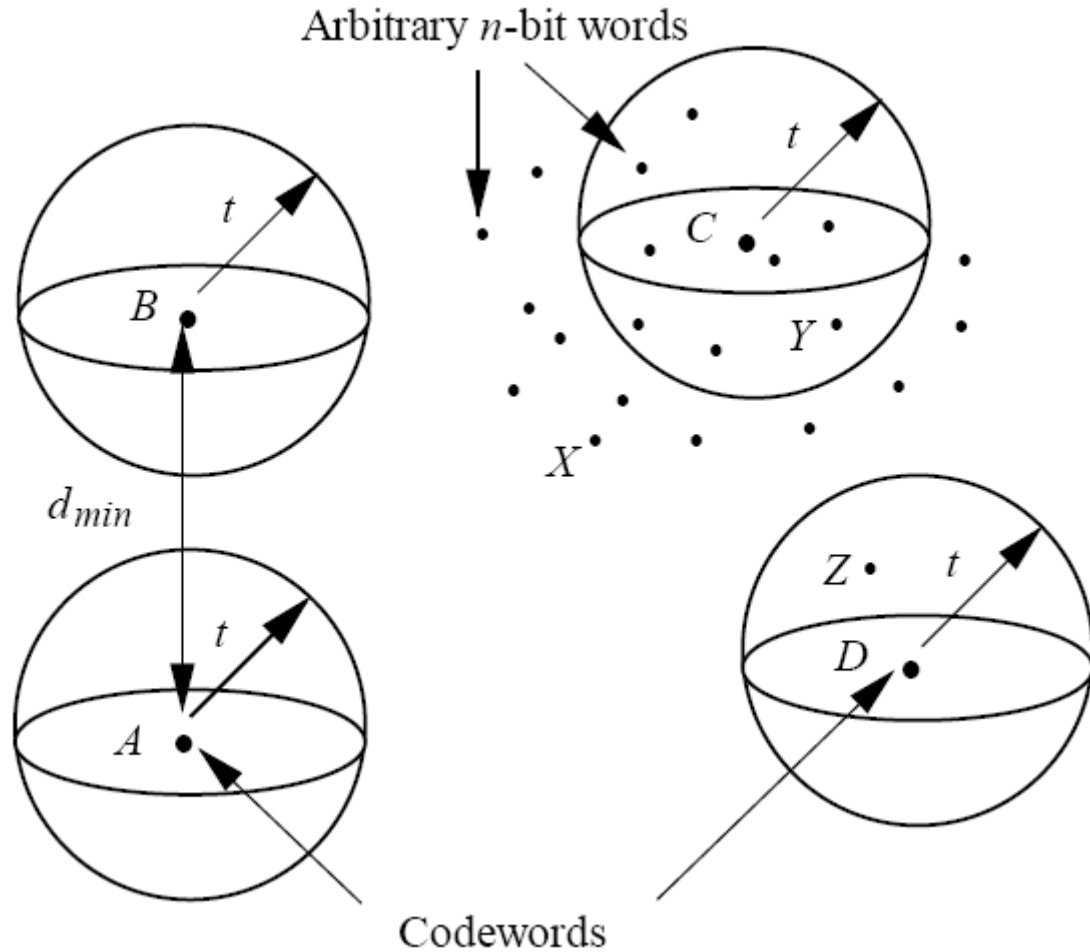
Bit “Distance” in Legal Words

Intuitively, if we are expecting m bits can be in error, then the original legal words (plus whatever check bits added):

- Need to have at least distance $m+1$ for the error to be detectable!
- Need to have at least distance $2m+1$ for the error to be correctable!
- This does not prescribe how efficient is the detection or correction, but just whether there is no ambiguity when we receive a word containing at most m -bit errors!

Hamming Distance of Legal Words

- The minimum “Hamming” distance is defined as the smallest number of places that any two codewords (block words) in the codebook differ.
- Error correction code is to add check bits to enlarge that distance!



Further Reading on Hamming Distance and Quad Logic

- *R. W. Hamming, Coding and information theory; 2nd ed.*
Richard W. Hamming , Prentice Hall, 1986
- *Z. Kohavi, Switching and finite automata theory*, McGraw-Hill,
1970, 1987 (Coding and quad logic)



Richard Hamming
(1915 – 1998)

Detectability and Correctability

For a (n,k) coding scheme (2^n codewords to represent 2^k data)

- Assume $t = \left\lfloor \frac{n-k-1}{2} \right\rfloor$ is the minimal distance between codewords
- $n - k - 1$ of error bits will be detectable (or at least $2t$ number of error bits will be detectable)
- t number of error bits will be correctable

Exercise: How many bits are less than Hamming distance 2 from $(7,4)$ coding scheme?

Exercise: How many bits are less than Hamming distance 4 from (n,k) coding scheme?

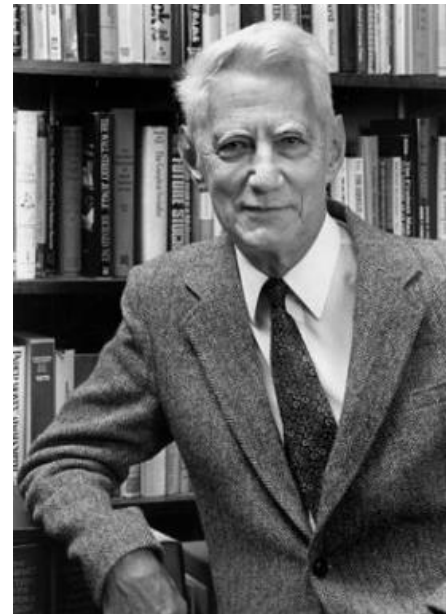
Shannon Coding Limits

The Shannon limit was posted in 1948, but only until Reed-Solomon coding is published in 1960, practical, efficient coding is available.

- C : Upper limit to the number of bits per second that can be reliably transmitted across a channel
- W : channel bandwidth in Hz; R : transmitted bit rate (bits/s)
- S : received signal power
- N : additive noise power
- E_b : signal energy per bit
- N_0 : noise power level in W/Hz

$$C = W \log_2 \left(1 + \frac{S}{N} \right)$$

$$C = W \log_2 \left(1 + \frac{E_b}{N_0} \frac{R}{W} \right)$$



Claude Shannon
(1916 -2001):

“I just wondered
how things were
put together.”

Transmission Rate

- For the accomplished transmission rate R (bits/sec), if
 - $R < C$: Arbitrarily small error rate can be achieved
 - $R > C$: Not possible to achieve reliable error rate no matter what code is used.

$$C = W \log_2 \left(1 + \frac{E_b}{N_0} \frac{R}{W} \right)$$

Probability of Random Error

- Probability of uncorrectable errors: P_{UE}
- Probability that the channel will change a symbol during the processing or transmission: P_{SE}
- Assume random errors are uncorrelated (no burst)

$$P_{UE} = 1 - \sum_{i=0}^t C_i^n P_{SE}^i (1 - P_{SE})^{n-i}$$

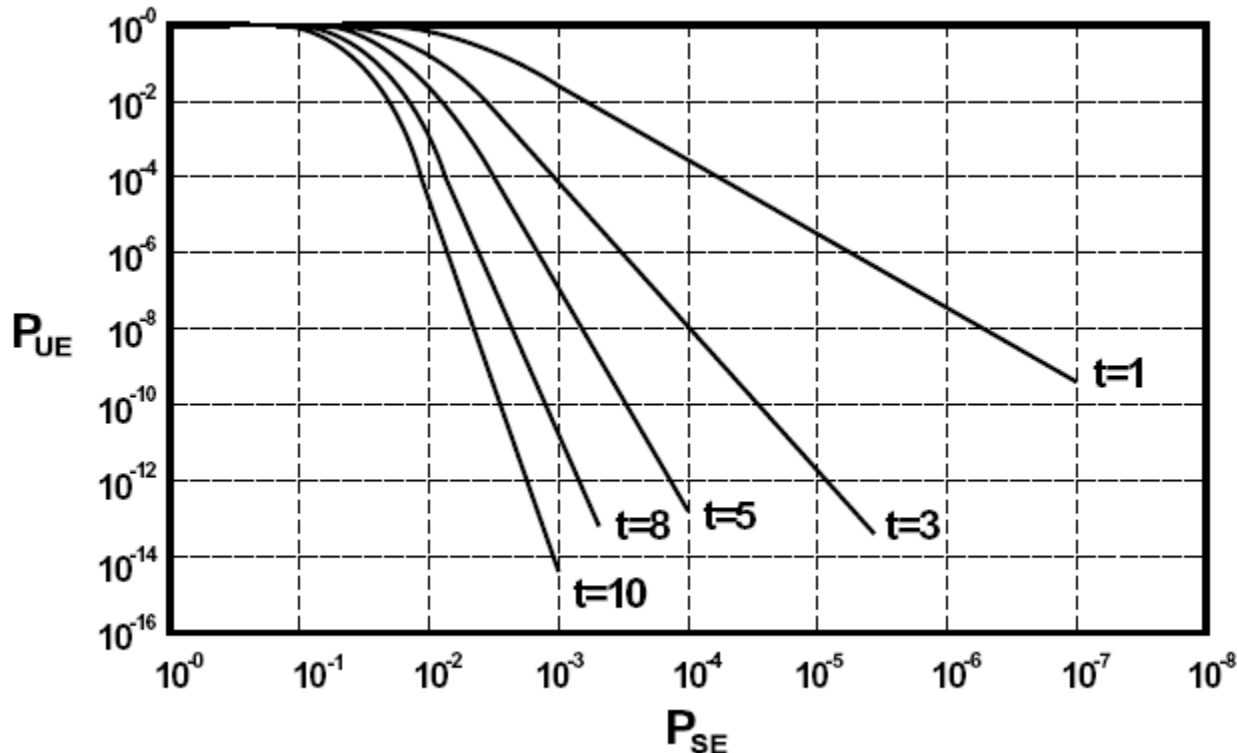
- Another way to measure error probability: corrected bit error rate
 - CBER = the reciprocal of the expected number of correct bits between uncorrectable errors

RS Code Performance Curves

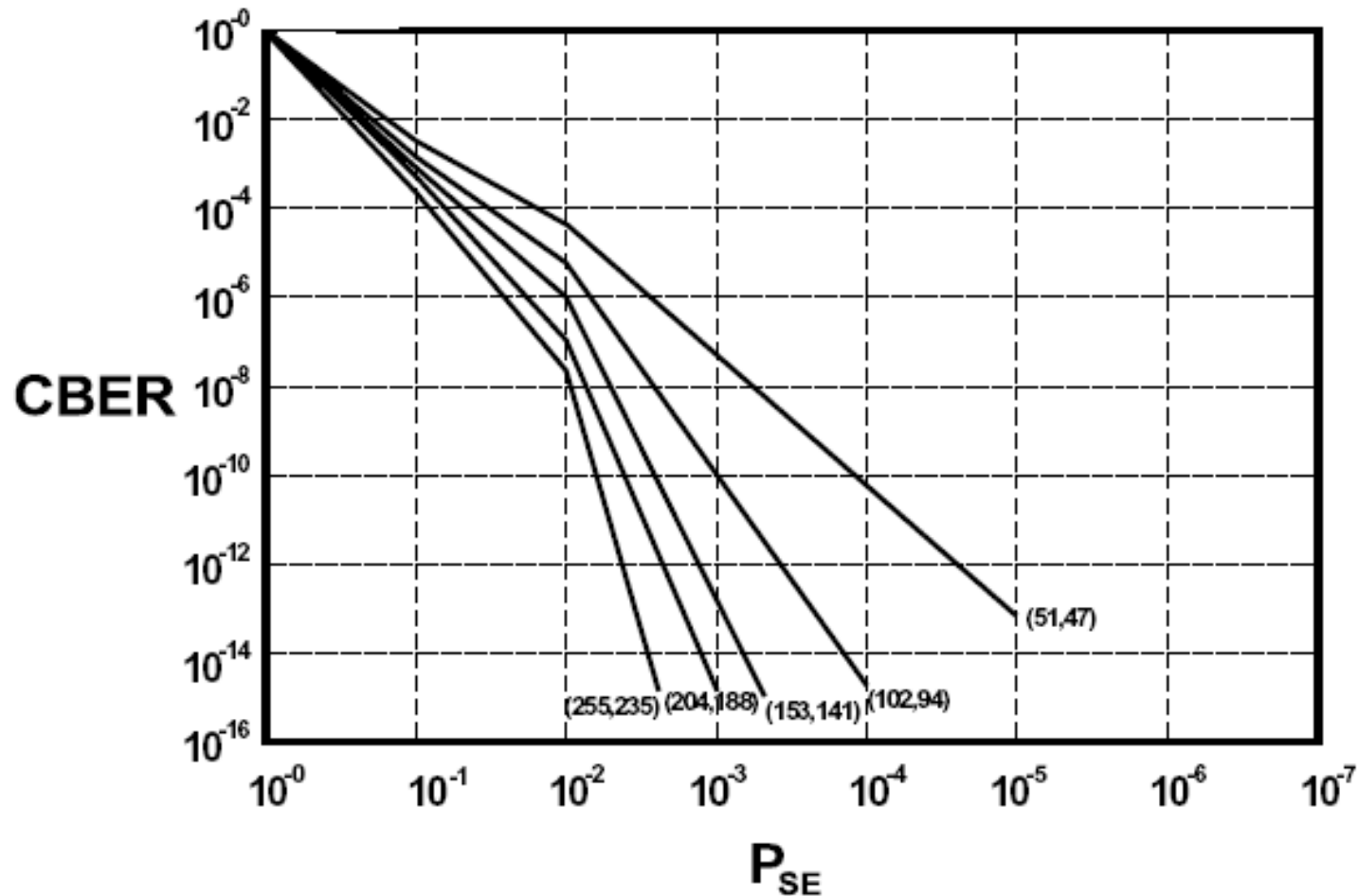
System performance of Reed-Solomon (RS) Block error correction code

- RS(255, k) code
 - k = 244; t = 10
 - k = 253; t = 1

Notice that within reasonably small number of check bits (< 10 check bits for 244 data bits), if P_{SE} is larger than 0.1, ECC does not perform well at all!!!



Bit-Error-Rate Curves



Example: For RS(255, 235) code with $P_{SE} = 10^{-3}$, $CBER = 10^{-17}$. That is, on average 1 error will happen after 10^{17} bits read. For a 1Gbit/s channel, this takes about 3 years!

Hamming Codes

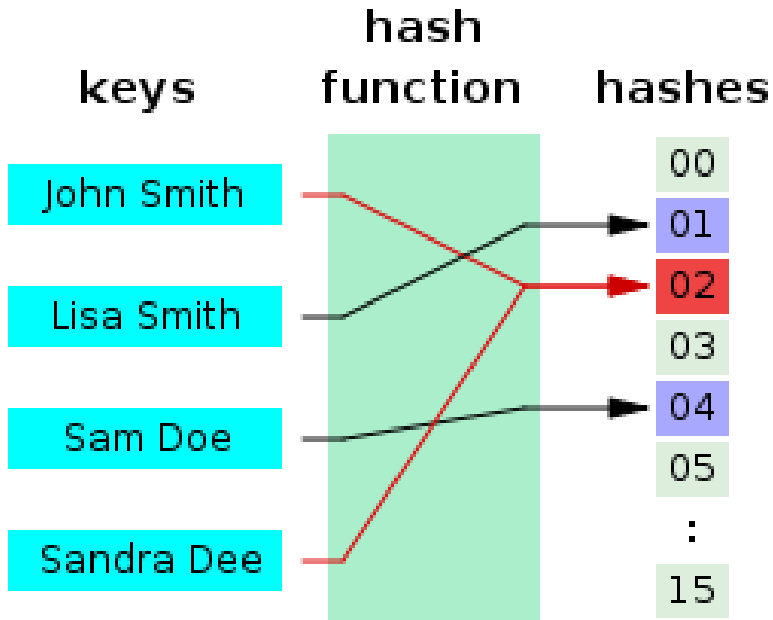
- To detect 2-bit errors and correct 1-bit error by parity bit locations, the block length $n = 2^r - 1$ and message length $k = 2^r - r - 1$ forms (n, k) Hamming code.
- Coding efficiency $\xi = k/n$

Total bit n	Data bit k	Redundant bit $n - k$	Hamming code	Efficiency ξ
3	1	2	(3, 1) triple repetition code	0.333
7	4	3	(7, 4)	0.571
15	11	4	(15, 11)	0.733
31	26	5	(31, 26)	0.839
255	248	8	(255, 248)	0.972

Outline

- Detectable and correctable digital codes
- Error-correction code (ECC) – an Example
- Hamming distance and Shannon theory
- **Hash function and CRC codes**
- Security and privacy in RFID systems
- Circuit implementation

Hash Function: Lossy Compression



Ways to see hash function

- Compiler: efficient use and search of variable addresses
- Error correction code
- Cryptography

- For $k > n$, if $x \in 2^k$ and $y = h(x) \in 2^n$, the hash function (or lossy compression) is the mapping that guarantee: if $h(x_1) \neq h(x_2)$, then $x_1 \neq x_2$

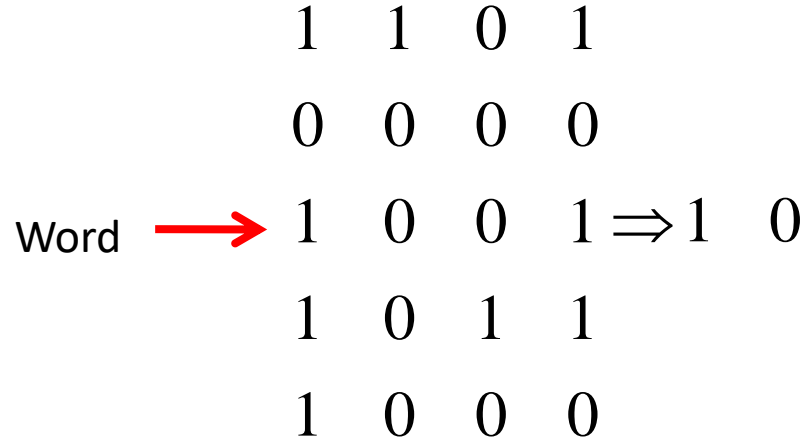
$$f : \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n$$

Collision or Degeneracy in Hash Function

- When $x_1 \neq x_2$, but $h(x_1) = h(x_2)$, this is called “collision” or “degeneracy”
- For ECC, all errors in x (defined by distance m from the word x) should map into the same valid $y = h(x)$, or they are all “collided”.

Hash Function in ECC

- All x can be compressed into a unique $y = h(x)$
- For correctability, all errors in x (defined by distance m from the word x), and ONLY those words, map into the same valid $y = h(x)$



- Say for 5 bits mapping into 4 bits, this is impossible from the pigeon-hole theory, as least $2^4 \times 5$ is needed for $x = 2^5$.
- But $2^7 > 2^4 \times 7$, so (7,4) can have one-bit correctability

Hash Function in Cryptography

- For cryptography, the hash function must be:
 - Preimage resistant (One Way): for all y , it is computationally infeasible to find any preimage x such that $h(x) = y$.
 - Collision resistant: it is computationally infeasible to find any x_1 and x_2 such that when $x_1 \neq x_2$, $h(x_1) = h(x_2)$.
- Any hash mapping can be thought as a matrix operation of $(k, n) \times (n, 1) = (k, 1)$

$$\begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Class Activity: Observation of the Taboo Game

- Hash function? What is collision?
- Private channels: for authentic listeners or for eavesdroppers?
- Use of “association” in addition to hash function: when “if $h(x_1) \neq h(x_2)$, then $x_1 \neq x_2$ “ is violated, it can be treated as “noises”.

One Time Pad (OTP)

- Modular addition of “message” (plaintext) and “pad” (key) of equal length
- Say if A = 0, B= 1, ..., Z = 25, OTP is
ciphertext = (plaintext + key) mod 26
or $c = (p + k) \text{ mode } 26$

H	E	L	L	O	Plaintext
7	4	11	11	14	p
X	M	C	K	L	key
23	12	2	10	11	k
4	16	13	21	25	(p + k) mod 26
E	Q	N	V	Z	ciphertext

Restriction of Pad in OTP

- Pad has the SAME length as the plaintext
- Perfectly random pad codes (say, if the pad is “AAAAA”, all secrets will be revealed)
- Pad is entirely secure (the attacker has NO information on pad)
- Pad can only be used ONCE (to avoid data remanence)

If all four restrictions are satisfied, OTP is information-theoretically secure (perfectly secure)!!!

Properties of One Time Pad

- Impossible to crack (the ciphertext contains NO information of the plaintext).
- If you only have a ciphertext but do not know the key, the ciphertext can be translated into ANY messages that are equally likely.
- Immune to brute-force attack: if the attacker tries EVERY possible key, he will get ALL possible plain texts with equal probability.
- If the attacker knows some part of the plaintext, it will help to decipher the other part.
- However, any practical consideration introduce potential vulnerability (if pad is so secure, why not use that as message?)

Cyclic Hash Function Generation

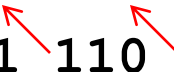
- Repeated use of hash mapping (say 1,024-bit key is used 1,024 times generate a 1M-bit key) can be compromised easily.
- For $k = 512$ and $n = 160$, a message m is padded and decomposed to blocks of m_i of bit length k
- h_i is defined as: $h_i = f(m_i, h_{i-1})$
- h_0 is the initiation vector

Repeated hash function

```
101 110 011 Plaintext
101 101 101 Repeated key
000 011 110 Ciphertext
```

Cyclic hash function

```
101 110 011 Plaintext
110 110 101 Repeated key
011 110 110 Ciphertext
```



Most Efficient Binary Operations

- XOR (one-to-one mapping with given key)
- Shift (rotation, or times/divides by 2)
- modular addition: $(p + k) \bmod 2^m$ (XOR is bitwise mod 2)

Cyclic Redundancy Check (CRC) Codes

```
11010011101100 000 <--- input right padded by 3 bits
1011              <--- divisor
01100011101100 000 <--- result
 1011            <--- divisor ...
00111011101100 000
 1011
00010111101100 000
 1011
00000001101100 000
      1011
00000000110100 000
      1011
00000000011000 000
      1011
00000000001110 000
      1011
00000000000101 000
      101 1 -----
00000000000000 100 <--- remainder (3 bits)
```

**Cyclic Redundancy Check (CRC):
Very good in detecting errors;
but less efficient in error
correction.**

CRC Error Detection Properties

- Often expressed in “generator polynomial over integer modulo 2”
- A divisor key of 1100001101 is $x^9 + x^8 + x^3 + x^2 + 1$.
- A key of length r (or polynomial of rank r) will increase the Hamming distance between any two words, in the best case, $r - 1$.
- In the best case, the probability of undetectable error is $1/(2^{r-1})$
- When $r > 4$, CRC can detect all single and double bit faults, and most of the burst errors (continuous stuck-at “0” or “1”)

Outline

- Detectable and correctable digital codes
- Error-correction code (ECC) – an Example
- Hamming distance and Shannon theory
- Hash function and CRC codes
- **Security and privacy in RFID systems**
- Circuit implementation

Desirable Security and Privacy

- Identification
- Authentication
- Privacy/indistinguishability
- Forward security from tampered tags to break other tags
- Delegation/restriction in tag reuse
- Proof of existence
- Distance bounding
- Synchronization of key updates

Classes of Security and Privacy

**Class
2 - 4**

Distance
Bounding

Proof of
Existence

Forward
Security

Delegation
and
Restriction

Synchro-
nization

Class 1

Authentication

Privacy

Class 0

Identification

Threats Against RFID Systems (1)

Method of Attack	Tag	Wireless Channel	Reader/Backend
Passive reading	-	Eavesdropping <i>(encryption before transmission)</i>	-
Active reading	Tag reading by manipulation of tag communication from a fake reader (reader authentication)	Replay <i>(challenge and response auth.)</i> Relay <i>(distance bounding or shielding)</i> Modification <i>(integrity checking)</i>	Reading reader's command to manipulate communication from a fake tag (tag authentication)
Rewriting	Tag rewriting (authentication or memory lock) Virus/malware to infect other tags	-	Reader rewriting by fake tag (tag authentication) Virus/malware to rewrite reader registers

Threats Against RFID Systems (2)

Method of Attack	Tag	Wireless Channel	Reader/Backend
Cloning	Tag cloning (<i>authentication, tamper-proof, physical unclonable functions</i>)	-	Reader cloning
Destruction Denial of Service	Wireless destruction (<i>kill protection</i>) Denial of reading	Jamming (no real solution)	Wireless destruction Denial of operation
Scanning and tracking	Tag scanning and tracking (<i>physical shielding or encryption</i>)	-	-
Side channel	Wireless side channel analysis (<i>side channel proofing</i>)	-	Wireless side channel analysis

Lightweight Cryptography for RFID

- Tags have few GEs, so no general public-key computation possible.
- CRC-16 for authentication and 32-bit access password for XOR encryption

ElGamal Tag Re-Encryption

- An initial $E(\text{ID}, r)$ is stored on the tag, where r is a random number agreed and known by the reader, and is used to encrypt the ID.
- The encrypted scheme will satisfy the hash function lossy compression and error correction properties simultaneously.
- After reader retrieving the tag $E(\text{ID}, r)$, a new $E(\text{ID}, r')$, where $r' \neq r$ and is randomly chosen, is *rewritten* to the tag (the re-encryption step)
- r can be linked to “password”, “date” or “group ID”

Parity Learning with Noise

- An attacker tries to guess a function $f(x)$ from given trials of $(x, f(x))$
- A simple example: $f(x)$ computes the parity of bits at selected fixed location of x (f is the binary map of bit selection, and then perform XOR).
- If sufficient $(x, f(x))$ is given, Gaussian elimination can be used to determine $f(x)$
- To prevent attacker learning, a noise y can be injected. Instead of presenting $(x, f(x))$, (x, y) is presented where $y = f(x)$ most of the time, but $y = 1 - f(x)$ with some small probability.

Class Activity: Parity Learning

- I have a secret key of 3 digits with non-repeating digits (entropy = 720, very small)
- You guess a number, and I will reply:
 - “a”: One of your guessed digits is correct and at the right place
 - “b”: One of your guessed digits appears in my key but not at the right place

Challenge and Response Authentication (HB Protocol)

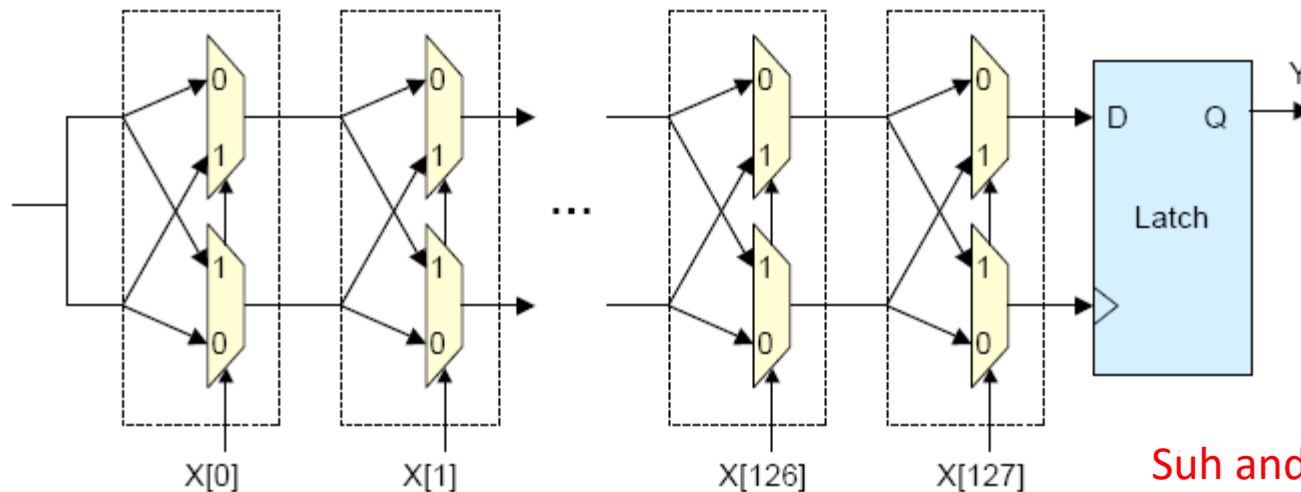
- Reader R knows secret key (x, y) . Tag T knows secret key (x, y, η) and has a “biased random bit” v (noise) with $\text{Prob}(v=1) = \eta$ and $0 < \eta \ll 1/2$.
- x and y are secret keys $\in \{0, 1\}^k$
- Challenge and response authentication:
 1. Tag T selects random $b \in \{0, 1\}^k$ (b is called the blinding factor) and sends b to reader
 2. Reader R receives b and selects a (a is called the challenge) and sends a to the tag
 3. Tag T receives a and computes $z = (a \cdot x) \oplus (b \cdot y) \oplus v$. Tag T sends z to reader
 4. Reader R receives z and accepts it if $z = (a \cdot x) \oplus (b \cdot y)$. Note: may need majority vote here

Physical Unclonable Function (PUF)

- A secret key or signature not assigned by user or stored in any data base, but is generated from “uncontrollable” but “repeatable” part of the hardware, such as timing and threshold voltage variations.
- This secret key is called physical unclonable function (PUF), and is always transmitted under encryption
- PUF can prevent “fire sale” and malware that attacks central data base.

PUF Implementations

- Timing channel
- SRAM in low VDD
- Flash memory
 - Dopant fluctuation
 - Oxide variation
 - Trap generation (stuck-at by bias-temperature instability (BTI) and random by random telegraph noise (RTN))



Suh and Devadas, 2007

Side Channel Attack

- Side channels: when computation and communication are performed, there will be traces left in temperature OR spectrum
- Used in battles and tactics in history
- In World War II, side channels of radar and radios are extensively investigated
- Reduce the search space of the attackers

Outline

- Detectable and correctable digital codes
- Error-correction code (ECC) – an Example
- Hamming distance and Shannon theory
- Hash function and CRC codes
- Security and privacy in RFID systems
- **Circuits implementation**


Constructing Error Correction Circuits

- Enabled by the digital system: only some values are allowed. Error correction means tremendous S/N gain in the analog sense, which is NOT likely in analog circuits.
- Broadly used in:
 - Wired and wireless communication channels
 - A/D data converters
 - Memory circuits
- Cannot be applied yet in generic logic circuits without large penalty: we have assumed correct bit location in space and time!!

Mathematical Forms of Coding

- We will use Hamming Code as an example, but applicable to RS and BCH codes with multi-bit error correction.
- Assume in the Hamming C(7, 4) code in the previous example
 - The data vector of length $k = 4$: m
 - The codeword of length $n = 7$: c
 - A generator matrix G exists with (7×4) entries such that $c = m \cdot G$

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Ex: $m = (0110)$  $c = m \cdot G = (0110) \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} = (0110011)$

Error, Syndrome and Parity Matrix

- Let e be the error vector, the read vector r of the memory becomes:

$$r = c + e$$

- In decoding with correction, the syndrome is obtained as:

$$s = r \cdot H^T$$

- The parity matrix H is constructed to satisfy


$$GH^T = 0$$

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

Notice: it is 1 – 7 in the column vector

- There is an infinite number of equivalent ones exists. Choice here is to let the syndrome to become the address bit where the error occurs.

Ex: $c = (0110011)$




$$s = c \cdot H^T = (0110011) \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix} = (000)$$

Uniqueness of Syndrome by Error

$$s = r \cdot H^T = (c + e) \cdot H^T = mGH^T + eH^T = eH^T$$

- The syndrome is univocally determined by the error vector e .
- The i -th column of H represents the syndrome for an error in the i -th position of vector r .

Ex: $c = (0100011)$  $s = c \cdot H^T = (0100011) \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix} = (011) = 3$

Majority and Cyclic Error Correction Circuits

- Matrix form is general, but the algorithm takes the multiplication of a large matrix, which may not be fast or small enough for memory blocks.
- Other forms of error correction circuits: majority (Reed-Muller, or RM); cyclic (Reed-Solomon, or RS); cyclic (Bose-ray-Chauduri-Hocquenghem, or BCH)

Reed-Muller R(8,4) code:

$$G = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

- $m = (a_0, a_1, a_2, a_3)$
- $c = m \cdot G = (c_0, c_1, c_2, c_3, c_4, c_5, c_6, c_7)$
- $r = c + e$
- d is then obtained from selective majority operations.

RM Majority Decoding

$$c_0 = a_0$$

$$c_1 = a_0 + a_3$$

$$c_2 = a_0 + a_2$$

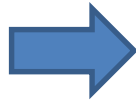
$$c_3 = a_0 + a_2 + a_3$$

$$c_4 = a_0 + a_1$$

$$c_5 = a_0 + a_1 + a_3$$

$$c_6 = a_0 + a_1 + a_2$$

$$c_7 = a_0 + a_1 + a_2 + a_3$$



$$a_1 = c_0 + c_4$$

$$a_1 = c_2 + c_6$$

$$a_1 = c_3 + c_7$$

$$a_1 = c_1 + c_5$$

$$a_2 = c_0 + c_2$$

$$a_2 = c_1 + c_3$$

$$a_2 = c_4 + c_6$$

$$a_2 = c_5 + c_7$$

$$a_3 = c_0 + c_1$$

$$a_3 = c_2 + c_3$$

$$a_3 = c_4 + c_5$$

$$a_3 = c_6 + c_7$$

$$a_0 = c_0$$

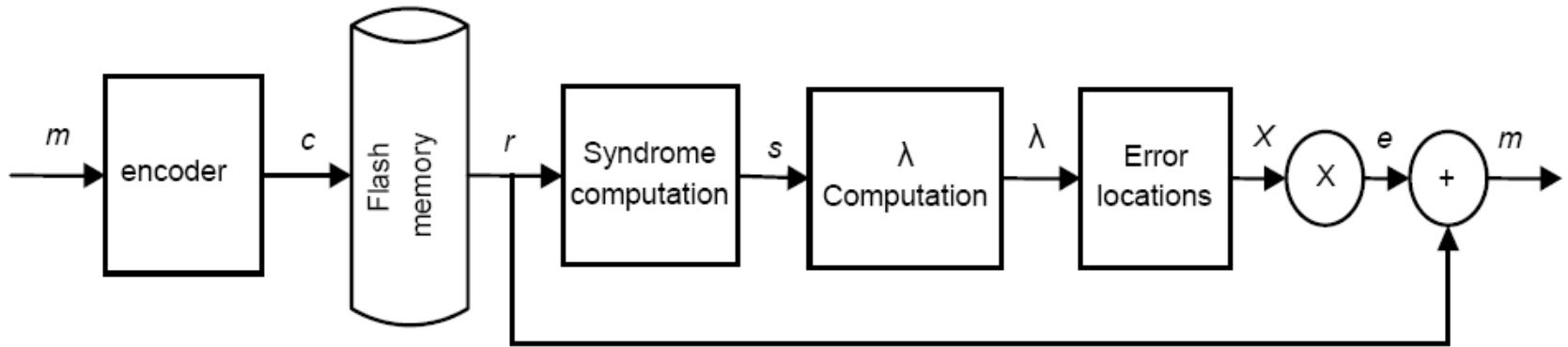
$$a_0 = c_1 + c_6 + c_7$$

$$a_0 = c_2 + c_5 + c_7$$

$$a_0 = c_3 + c_4 + c_7$$

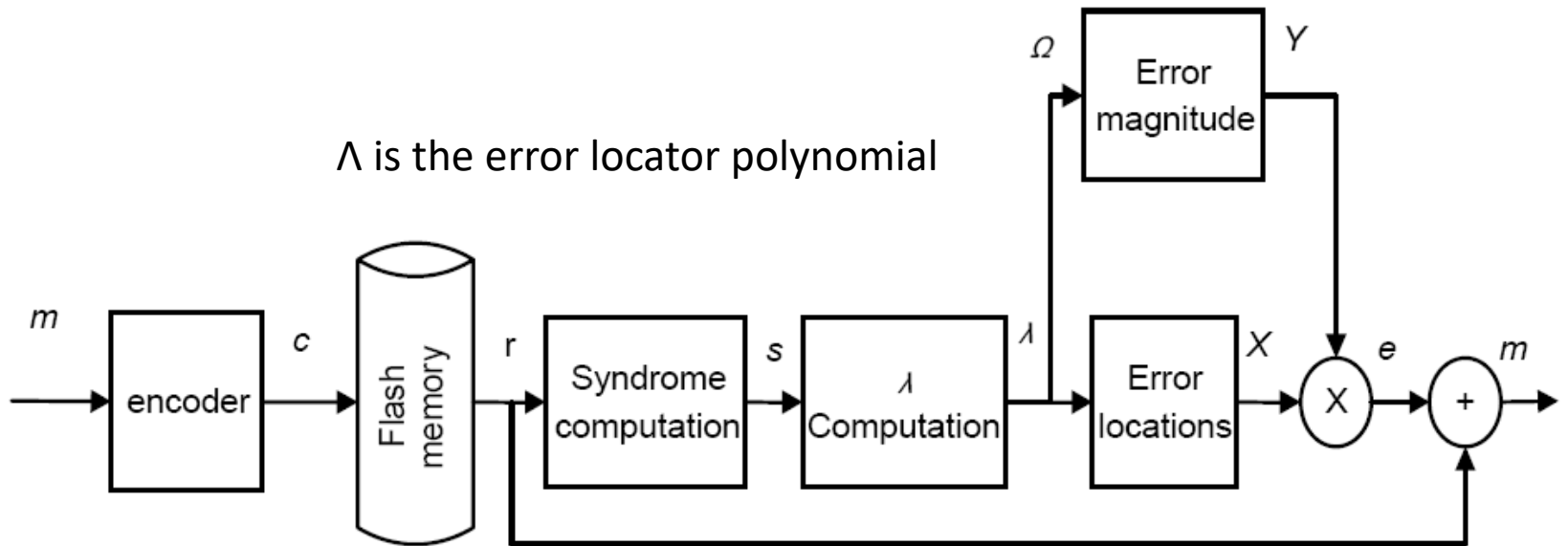
- If there is only one 1 in e , then each set of four equations for a_1 to a_4 can be wrong for one time. Correct answer can be obtained from a majority gate.
- $d_1 = \text{maj}(c_0 + c_4, c_2 + c_6, c_3 + c_7, c_1 + c_5)$, etc.

Cyclic Decoding



BCH Cyclic code

λ is the error locator polynomial



RS Cyclic code

What Do You Learn

- Adding check bits can increase the distance between legal codeword.
- Error correction coding and Shannon theory
- It is important to have small random error probability ($< 10^{-3}$) to efficiently drive down the uncorrectable probability to below 10^{-17}
- Once error correction code is added, often encrypting, authentication and watermarking can be considered together.

For Serious Readers

- R. Micheloni, A. Marelli and R. Ravasio, *Error Correction Codes for Nonvolatile Memories*, Springer 2008.
- C. Heegard and S. B. Wicker, *Turbo Coding*, Kluwer, 1999
- D. MacKay, *Information Theory, Inference and Learning Algorithms*, 2005.
- T. Richardson and R. Urbanke, *Modern Coding Theory*, 2007.

Questions?