Detection of Errors in Digital Systems

An Online Continuing Education Course for Engineers

Course Number: IC-2008

Credit: 2 Hours / 2 PDH / 2 CPD

© 2023 Decatur Professional Development, LLC. All rights reserved.

Detection of Errors in Digital Systems

Mark A. Strain, P.E.

Introduction

We are exposed to a lot of information every day, from viewing content on a website on the Internet to listening to a song on our smart phone. This information (or data) is constantly stored, transmitted and processed. It is important that the data is correct or relatively error-free. Some amount of error is acceptable, depending on the application. For example, a few bit errors in a music data file in an MP3 format are acceptable, but a few bit errors in the data being transferred to the flight controls of a rocket could be catastrophic.

Information theory is a branch of applied mathematics and electrical engineering involving the quantification of information. Claude Elwood Shannon ushered in the modern electronic communications age by developing information theory in 1948. He laid the groundwork for both the computer industry and telecommunications. A key feature of Shannon's theory was the introduction of the concept of entropy to information, which he demonstrated to be equivalent to a shortage in the information content in a message. The entropy or shortage of information in a message introduces errors in the data.

Error Detection Techniques

Most communications systems, whether wired or wireless, are unreliable to some degree. The application of error detection and correction techniques ensures the reliable delivery of data transmitted over a communication channel. Error detection also allows a user to reliably recover data stored in a memory device.

Errors are often times introduced in the channel during the transmission of data. Communication channels are subject to channel noise. Employing an error detection technique allows a user to detect an error at the receiving end. Employing an error correction technique allows a user to reconstruct (or request a retransmission of) data that was lost or corrupted. Data does not have to be transmitted great distances to benefit from error detection/correction methods. The data could be from one memory device to another. The channel could be as long as a ground station to satellite link or as short as a microprocessor to a RAM device on the same circuit board.

The basic approach of error detection and correction is to add some overhead (extra bits) to the data that will be utilized at the data's destination to determine if any errors had occurred in the channel during the transfer. The extra data (or overhead) may be inserted at regular intervals in the data stream (like a parity bit) or at the end of a packet of data or at the end of the entire transfer (like a checksum, cyclic redundancy check or hash function).

Parity

In mathematics, parity refers to the property of an integer with respect to being odd or even. In digital communications, parity refers to the number of bits within a word (e.g. a byte) being either odd or even. The property of a group of bits being odd or even is easily computed using an

Portions ©2015 Decatur Professional Development, LLC All rights reserved

XOR (^) function. Therefore parity is a common method and one of the simplest forms of error detection code.

Table 1 - Computation of Odd and Even Parity

Hex Number	Binary Equivalent	Parity Computation	Parity
0xCE	1100 1110	1 ^ 1 ^ 0 ^ 0 ^ 1 ^ 1 ^ 1 ^ 0 = 1	odd parity
0xCF	1100 1111	1 ^ 1 ^ 0 ^ 0 ^ 1 ^ 1 ^ 1 ^ 1 = 0	even parity
0x58	0101 1000	0 ^ 1 ^ 0 ^ 1 ^ 1 ^ 0 ^ 0 ^ 0 = 1	odd parity
0x59	0101 1001	0 ^ 1 ^ 0 ^ 1 ^ 1 ^ 0 ^ 0 ^ 1 = 0	even parity

A parity bit is a bit that is added to a set of bits to ensure that the number of bits in the set is odd or even. There are two types of parity checks: even parity and odd parity.

Even Parity
When even parity is used, the parity bit is set to make the number of bits in a set even. If the number of bits in the set was odd the parity bit would be set to 1 to make the number even again. If the number of bits in the set was even, the parity bit would be set to 0 to maintain the even number of bits.

Odd Parity
When odd parity is used, the parity bit is set to make the number of bits in a set odd. If the number of bits in the set was even the parity bit would be set to 1 to make the number odd again. If the number of bits in the set was odd, the parity bit would be set to 0 to maintain the odd number of bits.

Table 2 - Even Parity and Odd Parity

Hex Number	Binary Equivalent	Even Parity	Odd Parity
0x00	0000000	[0]0000000	[1]0000000
0x08	0001000	[1]0001000	[0]0001000
0x28	0101000	[0]0101000	[1]0101000
0x2A	0101010	[1]0101010	[0]0101010
0x6D	1101101	[1]1101101	[0]1101101
0x7F	1111111	[1]1111111	[0]1111111

Parity may be computed in software, but often times the operation is performed by hardware because the hardware computation is faster. The parity bit may be computed and injected into the data stream by the hardware itself, like a UART (universal asynchronous receiver/transmitter). In a digital communication system the transmitter hardware (e.g. a UART) will load a set of bits into a register and compute the parity bit and insert the parity bit either as the most significant bit or least significant bit in the set. The set of bits including the parity bit will then be shifted out and transmitted.

On the other end the receiver will receive each set of bits, strip off the parity bit and compute the parity for the set of bits it received. It will compare the newly computed parity bit to the one

Portions ©2015 Decatur Professional Development, LLC All rights reserved

stripped off from the receiver. If they match then the data received is deemed good. If they do not match, then an error has occurred during the transmission.

…[1] 0 0 0 1 0 0 0 [0] 0 1 0 1 0 0 0 [1] 0 1 0 1 0 1 0 [1] 1 1 0 1 1 0 1…

Figure 1 - Example Data Stream Incorporating Even Parity

…[0] 0 0 0 1 0 0 0 [1] 0 1 0 1 0 0 0 [0] 0 1 0 1 0 1 0 [0] 1 1 0 1 1 0 1…

Figure 2 - Example Data Stream Incorporating Odd Parity

The parity check is suitable only for detecting errors, not for correcting them. If a discrepancy in the parity check occurs (meaning if an error is detected), that portion of data must be discarded and retransmitted. If during a transmission the parity matches and the receiver deems the data is good, it may still be in error if two bits flipped instead of one, but there will be no way to tell from the parity check.

Transmitted data
…[1] 0 0 0 1 0 0 0 [0] 0 1 0 1 0 0 0 [1] 0 1 0 1 0 1 0 [1] 1 1 0 1 1 0 1…

Received data
…[1] 0 0 0 1 0 0 0 [0] 0 1 0 1 0 0 0 [1] 0 1 1 1 0 1 0 [1] 1 1 0 1 1 0 1…
↑
error detected (parity computes to 0, but parity received is 1)

Figure 3 - Example of Data Transmission and Error Detection

Advantages and Disadvantages of the Parity Method

Parity checking is simple, fast and takes little overhead, but is not the most robust form of error detection.

Checksum

Another form of error detection in digital systems incorporates a checksum within the data. A checksum is a fixed-sized (usually 8-bit, 16-bit or 32-bit) numerical data value that is computed over a message or block of data using some form of checksum function. Computing a checksum for a block of data and inserting the checksum at the end of the block of data before the data is transmitted is a simple error detection technique.

The integrity of the transferred data can be checked by computing the checksum over the received message and comparing it to the received checksum. The checksum function (or checksum algorithm) computes the checksum value over a block of data.

Portions ©2015 Decatur Professional Development, LLC All rights reserved

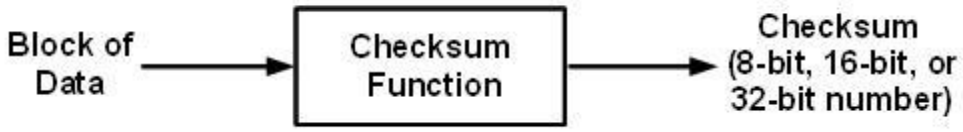

Figure 4 - Checksum Function

One method of computing a checksum is to XOR all of the words of the message block together. A word will typically be 8-bits, 16-bits or 32-bits in length. So, for a 32-bit checksum, a running XOR of every 4 bytes will be computed for the entire message. This XOR method is called a longitudinal parity check or horizontal parity check.

0xAE ^ 0xCA ^ 0x85 ^ 0x4D = 0xAC

Figure 5 - Longitudinal Parity Check Checksum

A simple form of checksum algorithm is to sum all of the words of a message in a counter. Any overflow is discarded. The counter is the same size as the words being counted.

0xAE + 0xCA + 0x85 + 0x4D = 0x4A

Figure 6 - Running Summation Checksum

During a data transmission a block of data is run through the checksum function to produce a checksum value. This value is appended to the message and the data is transmitted. The receiver receives and stores the message and strips off the checksum. The received data minus the checksum is run through the checksum function to produce a new checksum. This newly-computed checksum is compared with the received checksum. If the two match, the received data is deemed good. If the two checksums are different, an error has occurred and the data will need to be retransmitted.

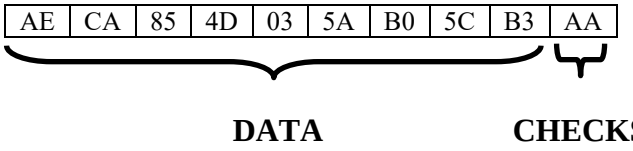

Figure 7 - Packet of Data Containing 8-bit Checksum

Advantages and Disadvantages of the Checksum Method

The use of checksums as an error detection technique is simple, fast and takes little overhead in resources and time. However, the technique is weak regarding error detection. If the bytes were transmitted in a different order the checksum result would be the same. If the bytes were transposed from big endian to little endian, for example, the checksum result would be the same.

Portions ©2015 Decatur Professional Development, LLC All rights reserved

Cyclic Redundancy Check
A cyclic redundancy check (or CRC) is a technique for detecting errors in digital data. This method is sometimes called a checksum, but is more robust than the simple method of adding or XORing the words. The value computed by the CRC function is called a check value, checksum or CRC.

The technique of error detection is the same as that of the checksum method. A given block of data is run through an algorithm to compute a numerical value. The numerical value (or CRC) is appended to the block of data and the data is transmitted. The receiving end computes the CRC and compares it to the one that was received. If they match the received data is deemed good. If they differ then an error has occurred and the data will have to be retransmitted.

CRC Algorithm
The CRC algorithm is based on cyclic codes which involve polynomial arithmetic in something called a Galois field (GF). The arithmetic is done using modulo-2, which means the arithmetic is accomplished using an XOR operator.

To compute a CRC a binary message is divided by a predetermined polynomial called a generator polynomial. The remainder of this operation is the CRC.

The CRC is computed using a generator polynomial, for example, $x^8 + x^2 + x + 1$ is a generator polynomial that may be used to generate a CRC-8 (8 bits in length). The generator polynomial is selected to maximize error detecting capabilities. The length (or exponential degree) of the polynomial determines the length of the CRC. They are typically 8, 16, 32 and 64 bits in length. A CRC may be computed in hardware or software.

Hardware Implementation
A hardware realization of a CRC circuit includes a shift register and some XOR gates. A circuit to compute the CRC for a generator polynomial of $G = x^3 + x^2 + 1$ is shown in **Figure 8**.

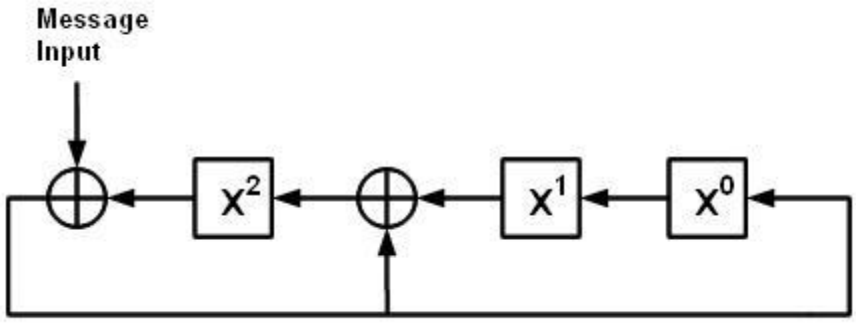

Figure 8 - Circuit of CRC Generation Using $G = x^3 + x^2 + 1$

Portions ©2015 Decatur Professional Development, LLC All rights reserved

For example, the message (M) 11010111 processed through the circuit produces a CRC of 001:

Table 3 - CRC Computed Using G = $x^3 + x^2 + 1$ and M = 11010111

Message	CRC Register
	0 0 0
1	1 0 1
1	0 1 0
0	1 0 0
1	0 0 0
0	0 0 0
1	1 0 1
1	0 1 0
1	0 0 1

So, the CRC-3 for the mes olynomial G = $x^3 + x^2 + 1$ is what is contained ocessed: CRC-3 = 001.

A circuit used to c = $x^8 + x^2 + x + 1$ is shown in **Figure 9**.

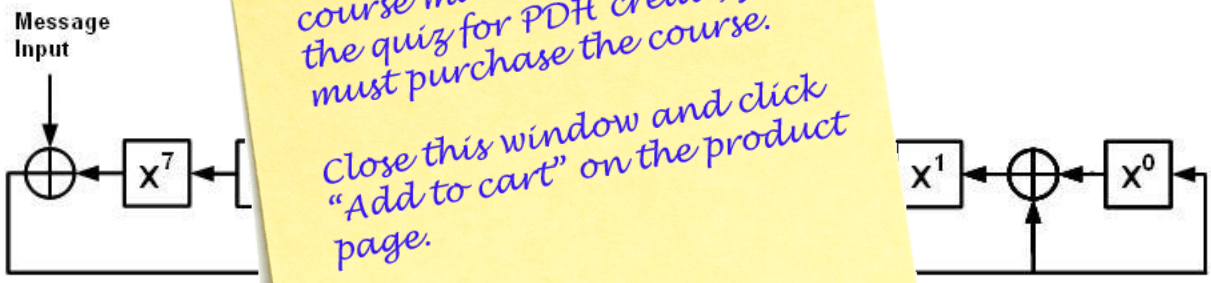

Figure + x + 1

For example, the messa through the circuit produces a CRC of 110

To view the remainder of the course material and to take the quiz for PDH credit, you must purchase the course.

Close this window and click "Add to cart" on the product page.

Portions ©2015 Decatur Professional Development, LLC All rights reserved