



FEBRUARY 26, 2024


CS963- CN1, MOHAMMED JAWED

HOME WORK - 3

MOHAMMED, JAWED

233560019

CyberSecurity



1. (20 points) Write a program (in any language) that generates an n -bit frame for transmission from a k -bit data block D and a $(n-k+1)$ bit CRC divisor P . Compile and run the program with at least two set of inputs to confirm that this program is generating CRC patterns correctly.

Now, modify the program that performs the following steps:

(a) Generates a message of $k=10$ bits.

(b) Uses the previous code with $P=110101$ to generate the corresponding 15-bit frame T for transmission.

(c) Generates transmission errors at any bit positions of T .

(d) Applies CRC to the received frame (i.e. frame T after introducing errors) to determine if the frame should be accepted or discarded.

To accomplish this task, I have utilized a Python script as per the instructions provided in the preceding question. The Python script **randomly generates K-10 bits** of data and utilizes the **polynomial $P=110101$** to generate a **15-bit frame for transmission**.

Additionally, the script introduces transmission **errors at random positions** within the 15-bit frame.

Subsequently, these error bits are fed into the **Modulo-2 Arithmetic CRC** method to validate whether the transmitted frame should be **accepted or discarded**.

Scripts:

This report includes **Python scripts** such as **HW3_1.py** (which contains all input data and interactions with the user) and **crc_encoder.py** (which contains all the logic such as XOR, Modulo-2 division, etc.).

For convenience in execution, I have included a **batch file named "RunHW3.bat"**(for windows) as well as **"RunHW3.sh"** (for Linux) to allow users to conveniently run this script as per the instructions for **Question 1**.

To execute the script(windows):

1. Open a command prompt or PowerShell window.
2. Navigate to the folder path containing the report/scripts.
3. Run "RunHW3.bat" in the command prompt or PowerShell window (The script will also check for Python; if it's not available, it will install Python from the target location.).
4. The output window will display all the results related to Question 1(see screenshot below).

To execute the script(Linux):

1. Open a shell terminal.
2. Navigate to the folder path containing the report/scripts.
3. Run "RunHW3.sh" in the shell terminal (The script will also check for Python; if it's not available, it will install Python from the target location.).
4. The output terminal will display all the results related to Question 1(see screenshot below).

Note: Each run will produce a different result as random frames are generated and errors are introduced within the frame.

Result shown as below,

===== Actual RUN: START =====

Using CRC Divisor P as: **110101**

(a) Generated a message **1100110111** of $k=10$ bits

(b) Generated the corresponding **15-bit frame T** as **110011011101000** for transmission

(c) Frame with error: **111011011101000**

(d) **Error detected!!, Frame should be discarded.**

Screenshot of Result Prompt Q1,

```

C:\CN1\HW-3>RunHW3.bat
-----
Name: Mohammed,Jawed
RollNumber: 233560019
Module: CN 1
Home Work Assignment 3
Date: 26 Feb 2024
-----
Running HW3 Q1 Python script...
+++++++ CRC Using MODULO 2 Arithmetic ++++++
===== TEST RUN =====
Validating for 1 input data
1st Input data: 11010111
1st Input data length: 8
1st Frame : 1101011101010
1st Frame length: 13
crc_divior used: 110101
crc_divior length:6
Genrated CRC pattern is correct for test input 11010111
-----
Validating for 2 input data
2st Input data: 111010
2st Input data length: 6
2st Frame : 11101010111
2st Frame length: 11
crc_divior used: 110101
crc_divior length:6
Genrated CRC pattern is correct for test input 111010
-----
===== TEST RUN END =====
===== Actual RUN: START =====
Using CRC Divisor P as: 110101
(a) Generated a message 1101011001 of k= 10 bits
(b) Genrated the corresponding 15-bit frame T as 110101100100011 for transmission
(c) Frame with error: 110101100000011
(d) Error detected!!, Frame should be discarded.
===== Actual RUN: END =====
Press any key to continue . . .

```

2. (20 points) (a). In a CRC error-detecting scheme, choose $P(x) = X^4 + X + 1$. Encode the bits 10010011011.

A complete instruction using polynomial CRC method as been provided under “HW-3-Q2.pdf” with steps-by-steps instruction.

Summary:

Encoded bit: 10010011011100

(b). Suppose the channel introduces an error pattern 100010000000000 (i.e., a flip from 1 to 0 or from 0 to 1 in position 1 and 5). What is received? Can the error be detected?

Please refer to **HW-3-Q2.pdf** (Q2.b)) for comprehensive instructions and details.

Summary:

What is received: **000110110111100**

Can the error be detected: **Yes, the error can be detected as the remainder after division is non-zero (1110).**

The result has been validated using the Python script developed for Q1.

(c). Repeat part (b) with error pattern 100110000000000

Please refer to **HW-3-Q2.pdf** (Q2.C)) for comprehensive instructions and details.

Summary:

What is received: **000010110111100**

Can the error be detected: **NO, the remainder after division by P is Zero, which is Zero hence, Error not detected.**

The result has been validated using the Python script developed for Q1.