Practical 11

Error Detection: LRC and Checksum

a. Split the message stream into sub-frames and find checksum by performing XOR operations on frame bits.

b. Perform LRC detection method while sending / receiving message at sender & receiver side.

* Objective

The objective is to implement error detection techniques using Checksum and LRC (Longitudinal Redundancy Check). This involves calculating the checksum by performing XOR on frame bits and applying LRC for error detection during transmission between sender and receiver.

* Code :-

```
#include <stdio.h>

unsigned char calculateLRC(unsigned char *data, int length) {
    unsigned char lrc = 0;
    for (int i = 0; i < length; i++) {
        lrc += data[i];
    }

    lrc = (~lrc) + 1;
    return lrc;
}

void printBinary(unsigned char byte) {
    for (int i = 7; i >= 0; i--) {
        printf("%d", (byte >> i) & 1);
    }
}

int main() {
    unsigned char dataToSend[] = {0x41, 0x42, 0x43,
                                  0x44};
    int dataLength = sizeof(dataToSend) /
                     sizeof(dataToSend[0]);

    unsigned char lrc = calculateLRC(dataToSend,
                                     dataLength);

    dataToSend[dataLength] = lrc;
```

```
printf("Data with appended LRC (in binary):\n");
for (int i = 0; i < dataLength + 1; i++) {
    printBinary(dataToSend[i]);
    printf(" ");
}
printf("\n");

return 0;
}
```

*Output

Data with append LRC (in binary):

01000001 01000010 01000011 01000100 11110110

*Learning Outcome

- Learn LRC and checksum-based error detection
- Understand XOR operations for checksum calculation.
- Gain hands-on experience with bitwise operations and binary data representation.