Error Detection and Correction

1) see lecture slides

2) Parity Check

a) 01101010 11010100 10101111 11001100 10001000 01110010 11100111

b)
01101010
11010100
10101111
11001100
10001000
01110010
11100111
11000000

c) Parity checks are simple to compute but can only detect single-bit errors.

3) Cyclic Redundancy Check (CRC)

a) Sender Side

Divisor in polynomial form:
x^8+x^2+x+1

Divisor in binary form:
100000111

Data:
1011001001001011

Sender calculates:
10110010010010110000000 (Data extended by m zeros where m is the power of the polynomial form) divided by 100000111

CRC:
10010011

Sender sends:
10110010010010111010011

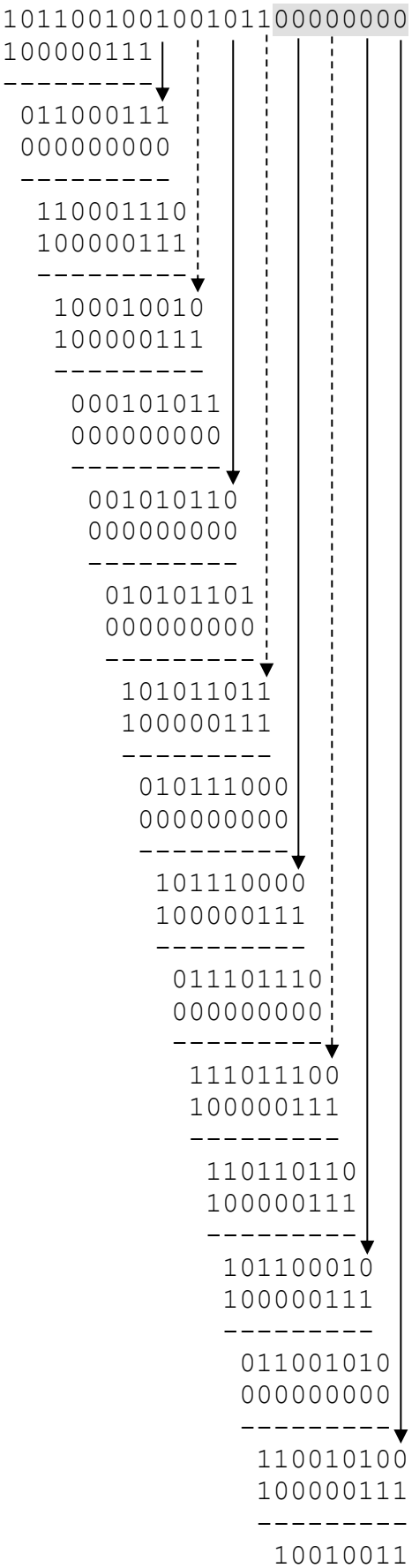
Receiver calculates:
10110010010010111010011 divided by 100000111

Remainder of this calculation after error-free transmission should be 0.

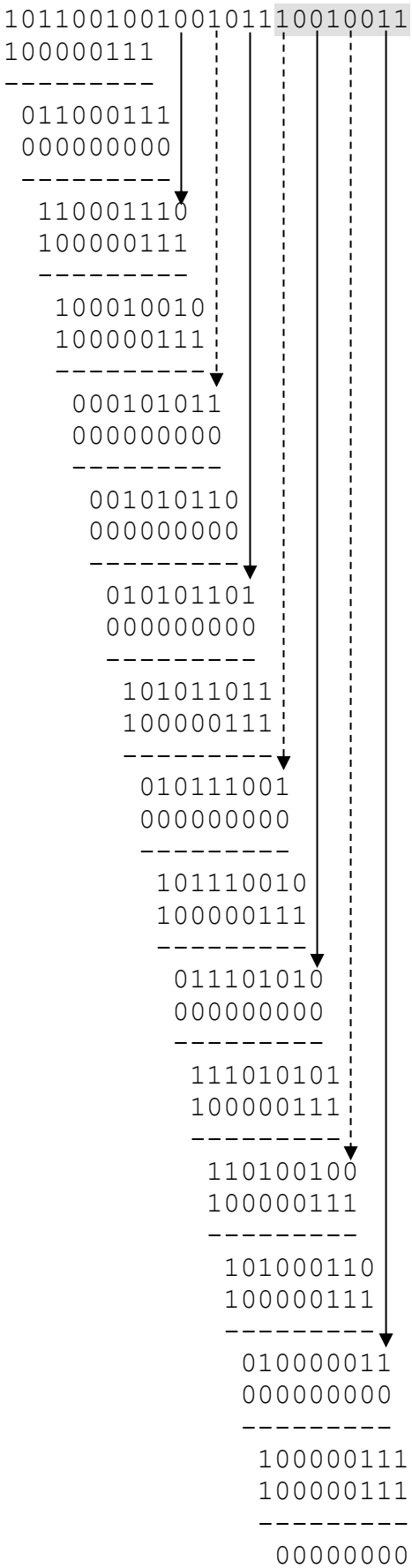
b) Receiver Side

If the left-most bit of the data is inverted the calculation will return a result other than 0. This will indicate to the receiver that the data have been corrupted.

Sender Calculation:



Receiver Calculation



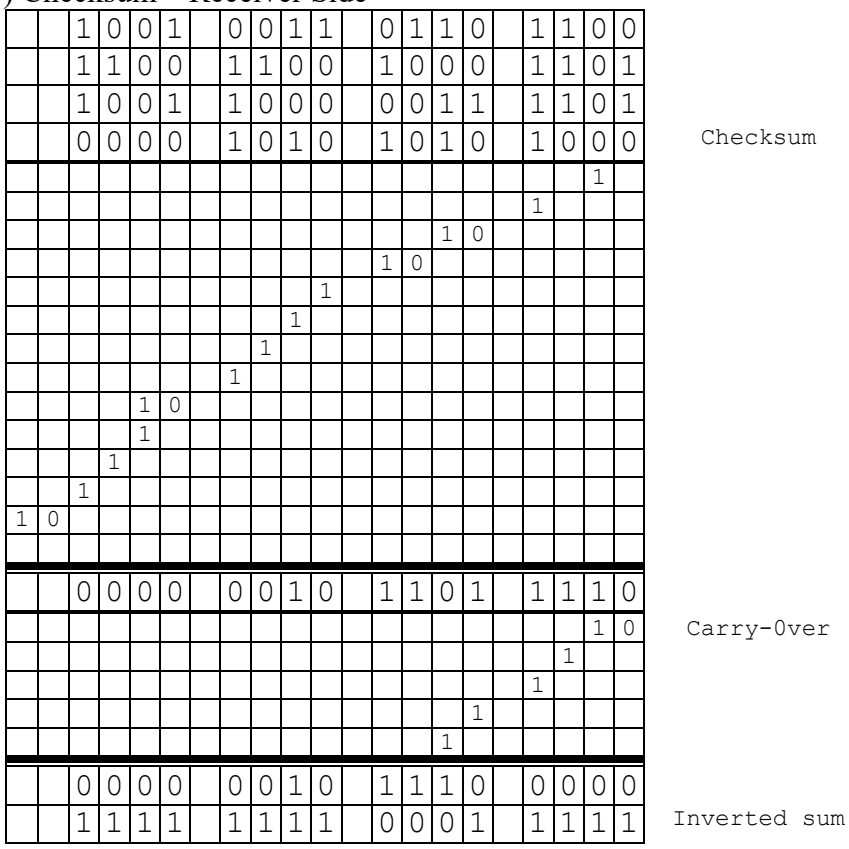
4) Checksum – Sender Side

```
     1001 0011 0110 1100
     1100 1001 1010 1101
     1001 1000 0011 1101
                       1
                     1
                   1 0
                 1 0
               1
             1
           1
         1
     1
     -------------------
     1111 0101 0101 0110
                       1
     -------------------
Sum  1111 0101 0101 0111
     0000 1010 1010 1000
```

For verification, the calculation on the receiver side:

```
     1001 0011 0110 1100
     1100 1001 1010 1101
     1001 1000 0011 1101
     0000 1010 1010 1000
                       1
                     1
                   1 0
                 1 0
               1
             1
           1
         1
     1
     -------------------
     1111 1111 1111 1110
                       1
     -------------------
Sum  1111 1111 1111 1111
     0000 0000 0000 0000
```

5) Checksum – Receiver Side



6) Hamming Code – Sender Side

```
1001011 1101100 1001001

100?101?1?? 110?110?0?? 100?100?1??

10011010110 11001100000 10011001111
```

7) Hamming Code – Receiver side

```
Bit 1, parity bit for 1,3,5,7,9,11
11101010000  0 (even number of 1s)
Bit 2, parity bit for 2,3,6,7,10,11
11101010000  1 (odd number of 1s)
Bit 4, parity bit for 4,5,6,7
11101010000  0 (even number of 1s)
Bit 8, parity bit for 8,9,10,11
11101010000  1 (odd number of 1s)

1010 = 10 => Single-bit error in bit 10
```

8)

a) The general layout of a HDLC frame includes a flag byte, an address, a control byte, the payload of the frame, a CRC and a flag byte. The flag byte marks the beginning and end of a frame. The address is either indicates the destination if the frame is being send

by a master node or the sender if it being send by a slave node. The control byte defines the type of the frame and may include a sequence of the frame and/or a sequence number for an acknowledgement. The CRC provides a mechanism for error detection.

b)
Flow control mechanisms are implemented through the I-frames and S-frames in HDLC. The control byte of the I-frame contains a sequence number for the sender and a sequence number for acknowledgements that indicates which frame is expected next. The control byte of the S-frame have a field for a code that indicates the type of S-frame and a field for a sequence number that can be used to indicate the sequence number of the next expected frame e.g. for ready-receive (RR) frames that represent acknowledgements or the sequence number of a frame that was not received.

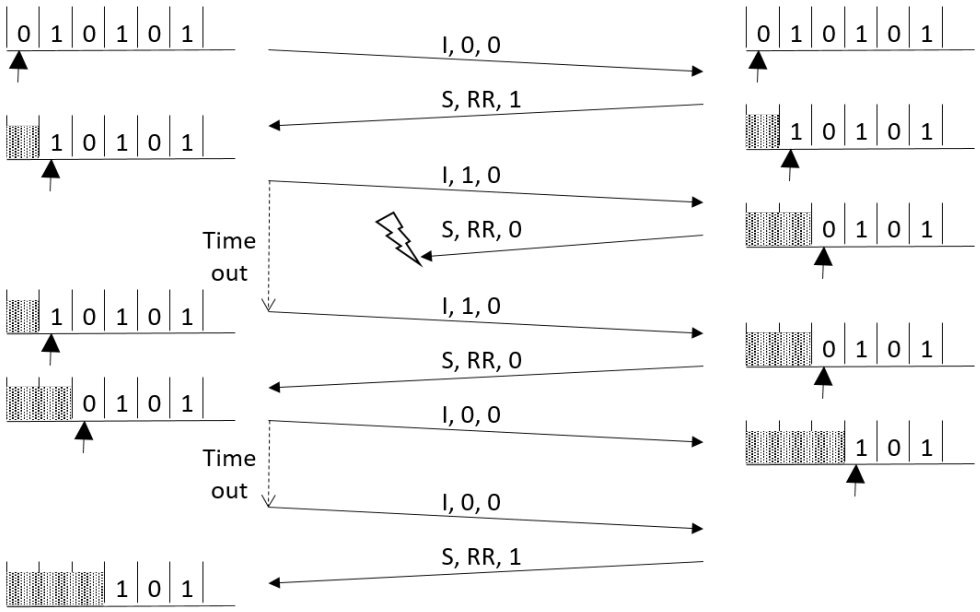


Figure 1: Stop & Wait - The diagram shows the progression of the sender – the arrow indicates the active frame - whenever it receives an acknowledgement and the progression of the receiver when it receives a frame with an expected sequence number. Whenever a timeout occurs, the sender will retransmit the current frame. **Important:** The receiver is passive and only acts when a frame has been received.

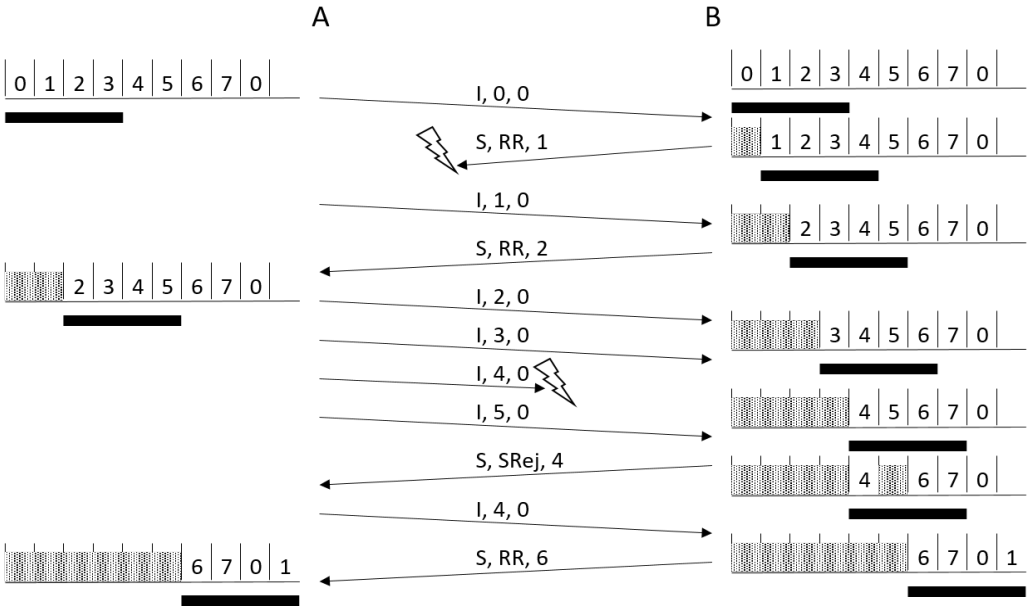


Figure 2: Selective-Repeat: The diagram shows the progression of the windows at the sender and the receiver – indicated by the bar underneath the list of frames. The window at the sender progresses whenever an acknowledgement is being received. An acknowledgement carries the sequence number that the receiver expects next; this indicates that the receiver has received all frames with sequence numbers up to the one indicated in the acknowledgement. The window at the receiver progresses when the first frame in the window has been received. When station B receives a frame without having received a predecessor, it will transmit a selective-repeat request for the frame that is missing.

(The above is also a good example for how to answer exam questions: Give an introduction, state any assumptions that you making, then – if the question asks for diagrams – draw diagrams and then include a description of the diagrams. A number of books and publications only provide very short captions e.g. "Stop & Wait", leaving it to the reader to interpret the figure – which is error-prone. If you provide a figure, provide an explanation to assure that the reader understands the figure.)

9)
The frame should look like the following – starting with a flag byte, then the address, followed by a control byte with assuming that the sequence number for both sender and receiver are 1, followed by the payload, a checksum and finally another flag byte:

```
01111110 00001100 00010001  <200 bytes> 011001110 01111110
Flag     Address  Control byte  Payload Checksum  Flag
```