DEPARTMENT OF COMPUTER SCIENCE

COMPSCI 314 S1 C 2006

Assignment 2, due Wednesday, 5 April 2006 (Answers)

Marks may be lost for poor presentation, or for answers without adequate explanation (a reader should not have to guess where some number or formula comes from).

Question 1.

This question assumes the Cyclic Redundancy Check generator polynomial x^4+x+1.

NOTE – It is essential that your answer have a "standard" layout, with columns properly aligned etc. Use a fixed-width font for the division details, or put them in a table or spreadsheet, etc. Marks may be lost for poor presentation

i. Show that the codeword from checksumming the information word 0010 0011 0110 is 0010001101101000. [5 marks]

ii. Check the validity of the received codeword 1110011100010101001. [3 marks]

iii. Check the validity of the received codeword 1110011010010101001. [2 marks]

iv. Check the validity of the received codeword 1110011100010111010. [2 marks]

v. You expect that each of the previous 3 codewords came from the same original information word. Comment on the three results, stating if possible the original information word. [bonus 3 marks]

part (i)

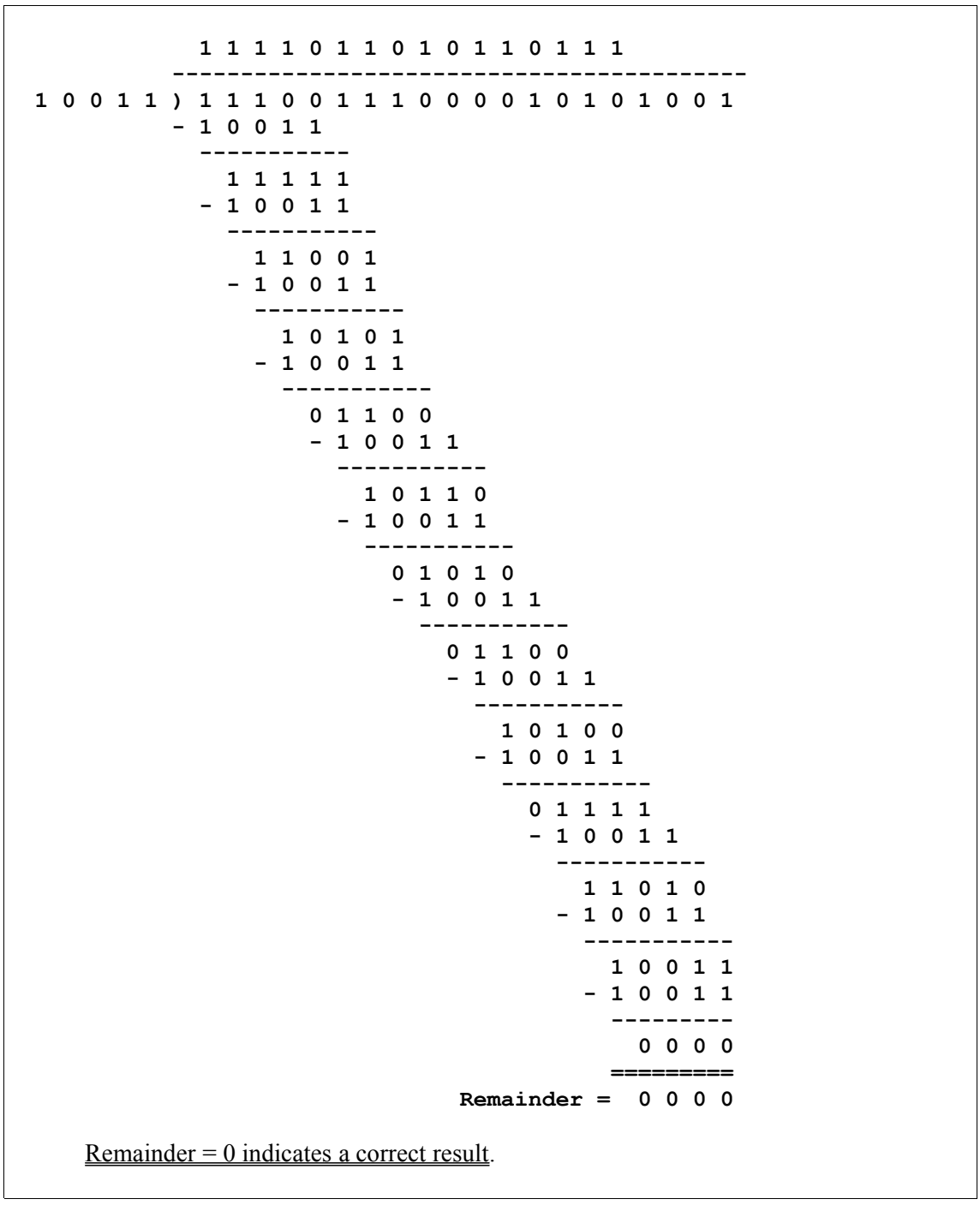
```
              0 0 1 0 0 1 0 1 1 0 0 0
            ---------------------------------
  1 0 0 1 1 ) 0 0 1 0 0 0 1 1 0 1 1 0 0 0 0 0
                - 1 0 0 1 1
                -----------
                  0 0 1 0 1
                    - 1 0 0 1 1
                      -----------
                        0 1 1 0 1
                        - 1 0 0 1 1
                          -----------
                            1 0 0 1 0
                          - 1 0 0 1 1
                            -----------
                              0 0 0 1 0
                                  =========
                         Remainder =  1 0 0 0

Codeword =   0 0 1 0 0 0 1 1 0 1 1 0   1 0 0 0
```

Marks for –

- Setting up divisor & dividend 1 mark
- appending 4 low-order zeros before division 1 mark
- correct division 2 marks
- stating the result 1 mark
- general presentation 1 or more marks may be deducted

part (ii)



Remainder = 0 indicates a correct result.

Marks for –

- including 4 low-order digits within division (don't add zeros) 1 mark
- correct division 1 mark
- stating the result (that zero remainder is good) 1 mark

part (iii)

```
            1 1 1 1 0 1 1 1 0 0 0 1 1 0 0 0
          ----------------------------------------
1 0 0 1 1 ) 1 1 1 0 0 1 1 0 1 0 0 0 1 0 1 0 1 0 0 1
          - 1 0 0 1 1
            -----------
              1 1 1 1 1
            - 1 0 0 1 1
              -----------
                1 1 0 0 1
              - 1 0 0 1 1
                -----------
                  1 0 1 0 0
                - 1 0 0 1 1
                  -----------
                    0 1 1 1 1
                  - 1 0 0 1 1
                    -----------
                      1 1 0 1 0
                    - 1 0 0 1 1
                      -----------
                        1 0 0 1 0
                      - 1 0 0 1 1
                        -----------
                          0 0 0 1 1
                              - 1 0 0 1 1
                                -----------
                                  1 0 0 1 1
                                - 1 0 0 1 1
                                  -----------
                                    0 0 0 0 0
                                        =========
                             Remainder =  0 0 0 1
```

Non-zero Remainder indicates an error

Marks for –

- correct division 1 mark
- stating the result (that a non-zero remainder is bad) 1 mark

part (iv)

```
              1 1 1 1 0 1 1 0 1 0 1 1 0 1 1 0
            ---------------------------------------
1 0 0 1 1 ) 1 1 1 0 0 1 1 1 0 0 0 0 1 0 1 1 1 0 1 0
          - 1 0 0 1 1
            -----------
              1 1 1 1 1
            - 1 0 0 1 1
              -----------
                1 1 0 0 1
              - 1 0 0 1 1
                -----------
                  1 0 1 0 1
                - 1 0 0 1 1
                  -----------
                    0 1 1 0 0
                  - 1 0 0 1 1
                    -----------
                      1 0 1 1 0
                    - 1 0 0 1 1
                      -----------
                        0 1 0 1 0
                      - 1 0 0 1 1
                        -----------
                          0 1 1 0 0
                        - 1 0 0 1 1
                          -----------
                            1 0 1 0 1
                          - 1 0 0 1 1
                            -----------
                              0 1 1 0 1
                            - 1 0 0 1 1
                              -----------
                                1 0 0 1 1
                              - 1 0 0 1 1
                                -----------
                                  0 0 0 0 0
                                  =========
                         Remainder =  0 0 0 0
```

Zero Remainder indicates no error, OR an error a multiple of to the generator polynomial

Marks for –

- correct division 1 mark
- stating the result (zero remainder is good) 1 mark
- the error may be a multiple of generator 1 mark (an extra bonus!)

part (iv)

A CRC check does NOT detect all errors. In particular, it will not detect an error if the error pattern is a multiple of the generator polynomial.. Here, results (ii) and (iv) differ by the error polynomial added into the last place and both pass the CRC check. But which one is correct? The only plausible method (and it is no more than that) is to count the number of differing bits (the Hamming distance). Thus (ii) and (iii) differ in 2 places, (ii) and (iv) in 3 places and (iii) and (iv) in 5 places. It therefore seems that (ii) is the correct word and that (iv) has suffered major corruption (which is the case).

- Recognise the situation 2 marks
- Making any further justified conclusion 1 mark

Question 2.

An SDLC/HDLC protocol is being used to send information via a geosynchronous satellite, with the parameters –

- satellite altitude 35,786 km (assume 36,000km)
- data rate 50 kbit/s
- information part of the frame 128 octets
- Frame Check Sequence 16 bits
- velocity of radio waves 300,000 km/s (3×10^8 m/s)

Assume no overheads apart from the message encapsulation and end-to-end transmission delay.

Calculate the maximum data throughput (as seen by a user) for –

i. 8 bit address & control [5 marks]

ii. 16 bit address & control [5 marks]

Answers

- The total distance end-to-end and return to receive an ACK is 4*36000 = 144,000km, giving a round-trip propagation delay of 144,000/300,000 = **0.48 seconds**.
- In "8-bit mode" the overhead per frame is 6 octets (48 bits and in "16-bit mode" is 8 octets (64 bits), so that to send 128 octets or 1024 bits, we must send (1024+48=1072) bits in 8-bit mode or (1024+64=1088) bits in 16-bit mode.
- The time to send a frame is (1072/50000 = 0.02144)s (8-bit) or 1088/500000= 0.02176)s (16-bit).
- With a propagation time of 0.480 s, we have frames in transit (0.480/0.2144=22.39 for 8-bit) and (0.480/0.02176 = 22.06 for 16-bit).
- Both modes therefore have about 22 frames in transit (and 22 really is accurate enough), but 8-bit mode allows only 7 unacknowledged frames. Therefore in any 0.480s round-trip time it can send only 7 frames before the first is acknowledged, or sending only 7*1024 bits of user data. The user sees data transfer at 7*1024/.48 = 14933 bit/s. NOTE that this rate is independent of the physical data rate on the link, as long as that is greater than about 15kb/s. The performance is limited by window size and round-trip delay.
- The 16-bit mode can have as many as 127 unacknowledged frames, allowing the full 22 frames per second to be sent. The user sees the raw data rate, scaled by the fraction of each frame that contains user data, or 50000*1024/1088 = 47059 bps.

Marks for –

- round-trip delay (each case) 1 mark
- total frame size (each case) 1 mark
- round-trip frames-in-transit (each case) 1 mark
- recognise different situation (each case) 1 mark
- final calculation user data rate (each case) 1 mark

High accuracy is not required and anything within about 5% is quite acceptable. The ideas are more important.

Question 3.

The diagram shows part of the State Transition Diagram for HDLC, operating in Asynchronous Balanced Mode (ABM) . The diagram is copied as part of Fig 1.37 (b) on page 82; if it differs from the original the textbook is definitive.

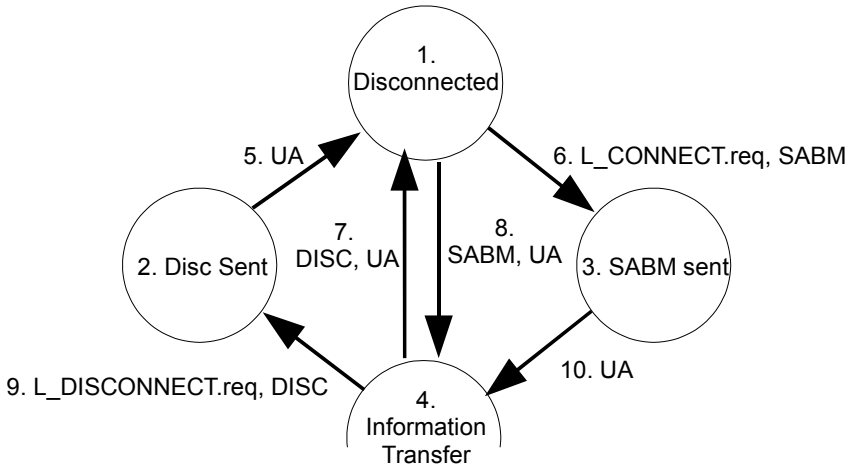
(*You should not need to worry about the fine details of the HDLC protocol; the textbook description of state machines should be adequate, together with your general knowledge of data communications protocols*.)

i. What is the general purpose of this part of the diagram? [2 marks]

The diagram covers the setting up of a connection at the start of a session, and the "breaking" of that connection at the end of the session. It excludes any data transfer.

ii. Briefly describe each of the 10 numbered states, actions, etc, giving for each as appropriate –

- the purpose of the state (or action etc),
- whether it applies to primary (initiating station), secondary (responding station) or both
- the reason for a transition etc,
- the action taken by a transition,
- the state resulting from a transition
- anything else that seems relevant. [5 marks]



Some important abbreviations are –

- UA Unnumbered acknowledgement Acknowledge a command
- SABM Set Asynchronous Balanced Mode Connection request
- DISC Disconnect Disconnection request

Num	Item text		Description
1	Disconnected state	Prim & Sec	Waiting for connection request, either local (L_CONNECT.req) or as SABM message
2	DISC command has been sent	Prim	Wait for UA to signal disconnect
3	SABM has been sent	Prim	Wait for connection acknowledgement
4	Information Transfer	Prim & Sec	Entered from State 3 after UA received (Prim), OR from State 1 after receiving SABM and sending UA (Sec)
5	UA received	Prim	After DISC sent, wait for this UA to enter Disconnect state (corresponds to UA of item 7)
6	L_CONNECT.req	Prim	After L_CONNECT.req command in DISC mode, send SABM message and enter state 3.
7	DISC, UA	Sec	Secondary receives DISConnect message, takes its own action, sends UA response and moves to “Disconnected” state
8	SABM, UA	Sec	Station receives SABM message while disconnected, takes action to set up connection, responds with UA and moves to “Information Transfer”
9	L_DISCONNECT.req, DISC	Prim	Primary has received software disconnect request; sends DISC message and moves to DISC sent state to wait for response
10	UA	Prim	Having sent SABM and moved to SABM state, primary waits for UA from secondary and then moves to “Information Transfer” state.