Solutions to Assignment 3

Question 1

Each answer will be different, but all must start with a flag sequence (0111 1110) followed by: five 'one' bits (1111 1), one stuffed zero bit (0), the remaining three 'one' bits (111), the binary digits of your student number with a stuffed bit after every five 'one' bits (110 11011 in this example), and a terminating flag sequence (0111 1110).

Question 2

(a) The remainder calculation can be done by long division by the generator polynomial of the data after appending three bits (one less than the number of bits in the generator polynomial):

```
        11001011
      -----------
1001 |11010010000
       1001
       ----
        1000
        1001
        ----
         0010
         ----
          0101
          ----
           1010
           1001
           ----
            0110
            ----
             1100
             1001
             ----
              1010
              1001
              ----
               011
```

The CRC is the remainder: 011.

(b) If an incorrect message is received (e.g. with the third bit inverted) the CRC calculation would be:

```
        11101111
      -----------
1001 |11110010011
       1001
       ----
        1100
        1001
        ----
         1010
         1001
         ----
          0111
          ----
           1110
           1001
           ----
            1110
            1001
            ----
             1111
             1001
             ----
              1101
              1001
              ----
               100
```

Since the CRC computation does not give a zero result there must have been an error.

Question 3

64-ary modulation transmits $\log_2(64) = 6$ bits per symbol. To transmit 64 bytes or $64 \times 8 = 512$ bits would require $\lceil \frac{512}{6} \rceil = 86$[1] symbols. 86 symbols will carry $86 \times 6 = 516$ bits so there will be $516 - 512 = 4$ padding bits.

Question 4

(a) Section 4.4.2 says of RFC 1662 says "Flag Sequence may serve as both a frame end and a frame begin."

[1] The $\lceil \cdot \rceil$ denotes "ceiling" or "round up".

It also recommends "always sending an opening Flag Sequence if the new frame is not back-to-back with the last."

If more than one Flag Sequence is sent between frames then there will be empty (zero-length) frames. Section 3.1 says "Two consecutive Flag Sequences constitute an empty frame, which is silently discarded, and not counted as a FCS error."

Therefore at least one flag character is required between frames but there is no maximum.

(b) Section 4.5.2 says of RFC 1662 says "All octets are transmitted least significant bit first, with one start bit, eight bits of data, and one stop bit. There is no provision for seven bit asynchronous links."

Thus only one start bit, one stop bit and eight data bits are allowed. Parity is not mentioned ~~so presumably it is permitted~~ in this section, however the introduction states that the specification provides for "...asynchronous links with 8 bits of data and no parity.". Therefore 9600,E,8,1 (9600 bps, even parity, eight data bits and one stop bit) would presumably not be allowed.

Question 5

A 10 Gb/s link will transmit $10 \times 10^9 \times 24 \times 60 \times 60 = 864 \times 10^{12}$ bits over 24 hours.

(a) A ML-PRBS sequence of length n can be generated using a shift-register of $K = \log_2(n+1)$ bits. For $n = 864 \times 10^{12}$, the number of bits of state required is $K = \log_{10} 864 \times 10^{12} / \log_2 10 = 49.6$ so a 50-bit shift register would be required to generate a sufficiently-long sequence.

(b) A ML sequence has l runs of ones of length $K - l$ except for $l = 1$ where there is one run of length K. The latter will be the longest run of ones. Thus the longest run of one's will consist of 50 consecutive one bits.

(c) Using 4-level signaling two bits are transmitted per symbol so the 50 bits will require 25 symbols to transmit. Transmitting 50 bits at 10 Gb/s will take $\frac{50}{10 \times 10^9} = 5$ ns.

Question 6

(a) 802.11 uses a frame-synchronous scrambler because the scrambling sequence is set on a frame-by-frame basis.

(b) The specification says the data "shall be scrambled with a length-127 frame-synchronous scrambler." Thus the scrambling sequence length is 127 bits.

(c) Although the same scrambling sequence is used for each frame, the starting point within the sequence changes from frame to frame because the initial state of the scrambler is set to a "pseudo-random non-zero state."

(d) The receiver uses the least-significant 7 bits of the SERVICE field in the preamble to initialize the descrambler. The transmitter sets these bits to zero so that the transmitted bits will be the same as the bits going into the scrambler's shift register (since $1 \oplus 0 = 1$ and $0 \oplus 0 = 0$). At the receiver these seven bits are simply shifted into the receiver descrambler's shift register to initialize it.