

1. Explain the difference between asynchronous and synchronous transmission. Assuming asynchronous transmission, one start bit, two stop bits, one parity bit, and two bits per signaling element, derive the useful information transfer rate in bps for each of the following signaling (baud) rates:

The main difference between asynchronous and synchronous transmission is the clock system that is used when receiving data. Synchronous systems, good for large frames, require a synchronization step prior to data being transferred. This synchronization step allows both sending and receiving parties to be operating on the same clock and allows for more advanced methods of error correction and compression. Asynchronous systems on the other hand, do not require the beginning step before data can be transferred. Asynchronous systems, good for randomly sent frames, require a start bit and one or more stop bits with the data sent between. The receiver must decode the data stream and look for the individual bits without the help of a synchronized clock.

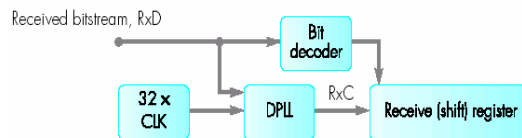
$$Baud = CharacterSize * BitRate$$

$$CharacterSize = 8DataBits + 1StartBit + 2StopBits + 1ParityBit = 12Bits$$

- | | |
|-----------------|--|
| (i) 400 baud | $BitRate = Baud / CharacterSize = 400 / 12 = 33.3bps$ |
| (ii) 800 baud | $BitRate = Baud / CharacterSize = 800 / 12 = 66.6bps$ |
| (iii) 1000 baud | $BitRate = Baud / CharacterSize = 1000 / 12 = 83.3bps$ |
| (iv) 2400 baud | $BitRate = Baud / CharacterSize = 2400 / 12 = 200bps$ |

2.
 - (a) Explain under what circumstances data encoding and a DPLL circuit may be used to achieve clock synchronization. Also, with the aid of a diagram, explain the operation of the DPLL circuit.

When a predefined set of symbols is transmitted prior to data in a system, a Digital Phase Locked Loop may be implemented to more easily find the center to each data bit and lock-in on the signal in order to synchronize the receiver clock with the data being transmitted.



The received bitstream signal is sampled at the local (receiver) clock speed. Each sample, a one or zero bit, is placed in the receive (shift) register. The need to adjust the phase is detected by reviewing the set of samples of the received signal. At each regenerated bit period, the receive (shift) register is consulted. If the center of the received bit lies at the center of the receive (shift) register, the sending and receiving clocks are in sync. If the received clock signal lags behind the received bitstream signal, then the phase needs to be adjusted to compensate for the difference. If the received clock signal leads the received bitstream signal, the phase needs to be adjusted in the opposite direction.

(ii) The receiver frame is checked for transmission errors. Use the generator polynomial. $x^4 + x^2 + 1$

The following example takes the error detection bits from the above example and the message from the above example. I have modified the message to 1110 0010 making it a one bit difference (last bit) to show an example of an error. The remainder of this example should not be equal to 0000 (which indicates no error).

$$\begin{array}{r}
 \\
 1\ 0\ 1\ 0\ 1\ \Big| \\
 + \\
 \hline
 0\ 1\ 0\ 0\ 1\ 0 \\
 + \\
 \hline
 0\ 0\ 1\ 1\ 1\ 1 \\
 + \\
 \hline
 0\ 1\ 1\ 1\ 1\ 0 \\
 + \\
 \hline
 0\ 1\ 0\ 1\ 1\ 1 \\
 + \\
 \hline
 0\ 0\ 0\ 1\ 0\ 0 \\
 + \\
 \hline
 0\ 0\ 1\ 0\ 0\ 0 \\
 + \\
 \hline
 0\ 1\ 0\ 0\ 0\ 0 \\
 + \\
 \hline
 \text{Remainder:} \quad \underline{\underline{0\ 1\ 0\ 1}}
 \end{array}$$

Here our remainder is 0101 which is not equal to 0000. This indicates an error and the frame must be retransmitted.

(b) Use an example to show how an error pattern equal to the generator polynomial used in (a) will not be detected.

Again, we take the message 1110 0011 and the appended error pattern 1000. This time we simulate a signal error that matches the divisor and we will not be able to detect the error, our remainder will be 0000.

$$\begin{array}{r}
 \\
 1\ 0\ 1\ 0\ 1\ \Big| \\
 + \\
 \hline
 0\ 1\ 0\ 0\ 1\ 0 \\
 + \\
 \hline
 0\ 0\ 1\ 1\ 1\ 1 \\
 + \\
 \hline
 0\ 1\ 1\ 1\ 1\ 0 \\
 + \\
 \hline
 0\ 1\ 0\ 1\ 1\ 1 \\
 + \\
 \hline
 0\ 0\ 0\ 1\ 0\ 1 \\
 + \\
 \hline
 0\ 0\ 1\ 0\ 1\ 0 \\
 + \\
 \hline
 0\ 1\ 0\ 1\ 0\ 1 \\
 + \\
 \hline
 \text{Remainder:} \quad \underline{\underline{0\ 0\ 0\ 0}}
 \end{array}$$

