SEMESTER II EXAMINATIONS - 2012-2013

EEEN 20060

Communication Systems

Solutions

Question 1

Answer any **four** parts of this question. All parts carry equal marks.

a) A device transmits text, encoded using ASCII, using an asynchronous serial protocol. The line idles at logic 1, which is represented by a negative voltage. Each character group has 1 start bit; 7 data bits, sent least-significant bit first; 1 parity bit, giving odd parity; and 1 stop bit. Figure 1 shows an example of the received signal. Identify the characters and any parity errors.

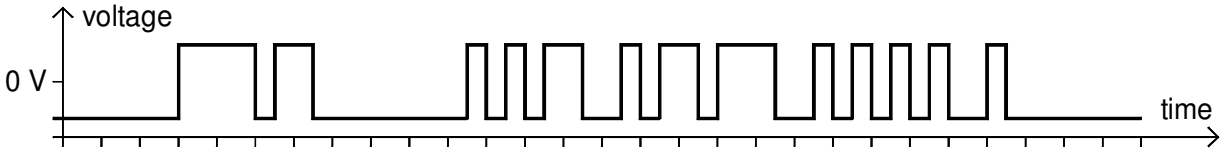


Figure 1: Received Signal

First find the groups of 10 bits, beginning with 0 (start bit) and ending with 1 (stop bit). The first group is between two idle periods: 0000100111. Removing the start and stop bits and reversing the order to put the LSB on the right: 11001000. The count of 1s in this is 3, which is odd as expected. The leftmost bit here is the parity bit, removing this gives the 7-bit data 1001000, which translates to decimal 72, and character H.

The next groups are consecutive, with the line returning to idle at the end. The bits are shown here in groups of ten: 0101001101 0010001101 0101011011.

The first of these reduces and reverses to give the 8-bit group is 01100101. This has an even number of 1s, so contains an error. It appears to be decimal 101, or character e, but it is most likely to be something else.

The next group gives 01100010. This has an odd number of 1s, so is probably valid. The value is decimal 98, character b.

The last group gives 10110101. This also has an odd number of 1s. The value (without the parity bit) is decimal 53, character 5.

b) Figure 2 shows a small packet-switched network that uses simple fixed routing tables. The number beside each link represents the cost of using that link. Draw up the routing table at node E, showing, for each destination, the preferred next node and an alternative (for use if the preferred link is broken). Consider what happens if node F fails, so that all links leading to F appear to be broken. Is there any risk of packets circulating in a loop?

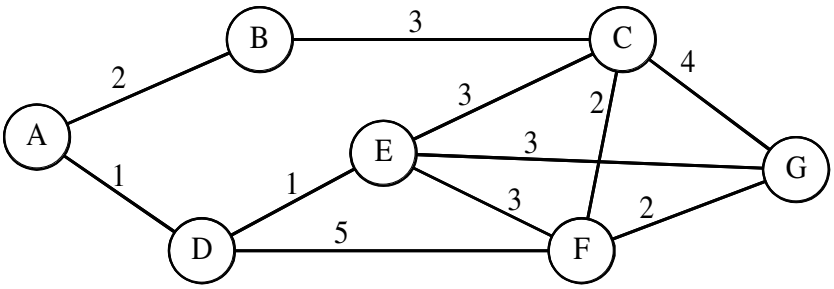


Figure 2: Network Diagram

Routing table should have the minimum cost route as first choice, and the route with the next higher cost as alternative. For example, consider the route to node D. The lowest-cost route from E to D is clearly the direct route, so next node D. If that is unavailable, the route via F has cost 8, and should be the alternative. The only other route that comes close is E-C-B-A-D, but this has cost 9. Repeating this analysis for all nodes except E itself gives the table:

Destination	Preferred next node	Alternative next node
A	D	C
B	D	C
C	C	F
D	D	F
F	F	C (G would also be correct)
G	G	F

Note that there are two equal-cost alternative routes from E to F. It would be correct to put either of these in the routing table, but one of them must be put in the table – a decision is needed.

If node F fails, E will see that the direct link is broken, but it will not know why. It will use its alternative route, sending packets destined for F to node C, for example.

Node C will have the direct link to F as its preferred route, but will use its alternative route. This could be back to node E, which would then send the packet to node C, etc. Even if C uses the route through node G as its alternative route to F, G will also have to choose an alternative, which will be either through node C (giving another loop) or node E (giving a triangular loop).

c) When a packet arrives at a node in the network in Figure 2, a routing decision is made and the packet is placed in a first-in-first-out queue for the appropriate link. Packets arriving at node D can be modelled as a Poisson random process, with parameter $\lambda = 2000$ s^{-1}. Of these, 25% need to travel on link D-E, which operates at 1 Mbit/s. The packet size has exponential probability density function, with average 1500 bits. The link layer protocol adds an overhead of 64 bits. Find the average waiting time in the queue for link D-E at node D. Also find the probability of finding the queue empty.

Packets are entering the queue for link D-E with average rate 500 packets/s. This is the arrival rate for the queue, λ.

The average service time is the time to transmit 1564 bits. At 1 Mbit/s, this is 1.564 ms. This gives a service rate $\mu = 639.4$ packets/s. The link utilisation $\rho = 0.782$.

From the lecture notes, with exponential distribution of service times (packet lengths), and arrivals a Poisson random process, the average waiting time is $E[t_W] = E[t_Q] - \frac{1}{\mu} = \frac{\rho}{\mu(1-\rho)}$. Substituting values give 5.61 ms.

The probability of finding the queue empty is P_0. Again, from the notes, $P_n = \rho^n(1-\rho)$, so $P_0 = 1 - \rho = 0.218$.

d) A byte-oriented link-layer protocol uses a separate checksum to detect errors in the header, which contains 10 bytes, including one header checksum byte. The checksum is such that when all 10 header bytes are added at the receiver, modulo 256, the result should be 0. Calculate the checksum byte that should be appended to the 9 header bytes below. Explain what is meant by failure of an error detection system, and give examples of error combinations that could cause this error detection system to fail.

1 198 18 157 23 165 49 0 82 ???

Adding the 9 bytes gives 693. The remainder after division by 256 is 181. Subtracting this from 256 gives 75. So the checksum byte should be 75. Then adding all 10 bytes gives 768, which is exactly 3 times 256.

The error detection system fails if the checksum appears to be correct, even though errors have occurred. That means that bits have been altered in the physical layer, but when the receiver adds the ten header bytes, modulo 256, the answer is 0.

No single bit error could cause this. The simplest cause would be two bit errors, changing the same bit in two different bytes, in opposite directions. In the case of the leftmost bit, two changes in the same direction would also yield a valid checksum.

A failure could also be caused by other means, such as two bits of equal value being changed from 0 to 1, and one bit of twice that value being changed from 1 to 0. However, all of these would need three or more bit errors in the block, so would occur with much lower probability.

e) Explain briefly why a hierarchical addressing scheme is needed in a network of networks, such as the worldwide telephone network or the Internet. A PC is configured with IP Address: 137.43.68.29, Subnet Mask: 255.255.224.0 , Gateway: 137.43.66.1. What range of IP addresses is in its subnet? What will it do with packets destined for IP addresses outside this range?

Hierarchical addressing is needed for two reasons:

Each network needs to be able to assign and manage its own addresses, without having to consult every other network. This is easy to arrange if part of the address identifies the network and the other part, locally managed, identifies a node on the network.

Each node needs to be able to find a route to any other node, on any network. Treating this as a single routing problem would be very difficult. A hierarchical address means that any node can recognise that an address is not in its own network, so the route to the destination is via a gateway node. The routing problem is then reduced to finding routes between gateway nodes. Once the destination network gateway is reached, it can solve the local routing problem in its own network.

In the example, the third byte of the subnet mask is 1110 0000. This means that the first 19 bits in an IP address identify the network, and the remaining 13 identify a node on that network.

The third byte of the example IP address is 0100 0100, or 64 + 4 = 68. Only the three leftmost bits of this belong to the network part of the address: 64. So masking the IP address with the subnet mask gives 137.43.64.0.

The remaining 5 bits of the third byte belong to the subnet part of the address, and can add up to a maximum of 31. The entire fourth byte also belongs to the subnet part of the address. Thus the range of addresses in the subnet is from 137.43.64.0 to 137.43.95.255.

If the PC wants to send a packet to an address outside this range, it will send it to the gateway node, which will then route it to the destination subnet.

Question 2

You are to design a link-layer protocol to provide reliable transmission of arbitrary binary data over a 60 km radio link. The physical layer delivers 1.2 Mbit/s in either direction, with each bit having probability of error 2×10^{-5}. The hardware at each end of the link can be switched to transmit mode or to receive mode, but cannot do both at the same time. Each change of mode takes 20 µs, and it is not possible to transmit or to receive during this time. You may assume that the speed of light is 3×10^{8} m/s.

Other devices use the radio channel, so every transmission must include the address of the intended receiver, which is an 8-bit number. The blocks of data to be transmitted can vary in size from 64 bits to 3200 bits, always a multiple of 16 bits, with an average size of 2000 bits.

a) Decide how your protocol will operate. Explain how it will recognise the start and end of each frame, and how it will detect errors in the frame. Specify the size of the sending and receiving windows (if any). Specify what responses the receiver should send, and what the sender should do if it receives no response. Give reasons for your design decisions. (30%)

This is a design question, so there are many possible solutions. The blocks of data are always a multiple of 8 bits long, and the address is also 8 bits, so a byte-oriented protocol could be used. However, a bit-oriented protocol would also work, and might be a simpler choice.

For byte-oriented designs, start and end marker bytes need to be defined. Some mechanism is needed to avoid confusion with data bytes – either some form of byte stuffing or a byte count (well protected) in the header. Error detection could use parity check bits or a checksum – probably 16 bits to give reliable protection of this number of bytes.

For bit-oriented designs, a unique flag sequence can mark both start and end of a frame. Bit stuffing is the obvious way to avoid this flag from occurring in the frame contents. Error detection could use a CRC – again, probably 16 bits long.

As transmission in both directions at the same time is not possible, sliding window protocols will not bring any benefits. A stop-and-wait protocol should be used, so sending and receiving widows will have size 1.

Responses must be at least an acknowledgement of data blocks received without errors, so the sender knows it can move on to the next block. Sending negative acknowledgements is optional.

The sender will have a maximum time to wait for a response. If this time limit is reached, it could re-transmit the previous data block. Sending an enquiry frame is an option, but brings little benefit.

If there are no negative acknowledgements, the time limit should be only slightly longer than the expected time waiting for a response. Enquiries do not make sense in this case, as the usual reason for lack of response is an error in the received frame, so re-transmission will save time.

b) Design the frame structure. Specify what information is in the header of a data frame, and in the trailer, and the number of bits allocated to each item. Similarly, specify the structure of any non-data frames that you use. Give reasons for your design decisions. (30%)

Again, there are many possible solutions. Minimum information in the header is a start byte or flag, address as specified, and sequence number. If the frame identification relies on a byte count, this must also be included, and the header should possibly be protected by its own checksum. Another option would be a frame type indicator, to distinguish between data and control frames – the byte count could also do this, if present.

The trailer just contains the check bits for error detection, and an end marker byte or flag.

The number of bits allocated should be sufficient to hold the largest value needed. For example, the byte count could be up to 400, so 9 bits would be needed. In a byte-oriented protocol, this might be assigned 2 bytes, or it could share two bytes with some other data.

Control frames could be just like data frames, but with no data. They could be identified by a type indicator in the header, or by the byte count being 0, or by the byte count being 500 or some other value that is not possible in a data frame.

c) Consider a cycle starting as the sender switches to transmit mode after receiving an acknowledgement, and ending when the next acknowledgement is received. Hence calculate the throughput that your protocol can deliver, using the average size of data block and assuming that there is always data waiting to be transmitted. Show clearly how you arrive at your answers, and explain any additional assumptions that you make. (40%)

This will depend on the design decisions above. For example, use a bit-oriented protocol with a flag defined as 6 consecutive 1 bits, protected using bit stuffing. An extra bit will be added after any sequence of 5 consecutive 1 bits – with random data, this might occur with probability 1/32.

In this example, header contains 8 bits to provide the flag, 8 bits for the address, and 4 bits each for sequence number and frame type. Trailer contains a 16-bit CRC and another 8-bit flag. Thus header and trailer add overhead of 48 bits. With the average data block, bit stuffing is applied to 2032 bits, so might add another 64 bits to the frame. Total frame size is 2112 bits.

Assume positive and negative acknowledgements are sent, using the same header and trailer, so expected length is 49 bits (including bit stuffing).

Cycle time is:
time for sender to switch to transmit mode: 0.02 ms.
time to send frame: 1.76 ms;
propagation delay for frame to reach receiver: 0.2 ms – sender can switch to receive mode during this time;
time for receiver to switch to transmit mode: 0.02 ms;
time to send acknowledgement: 0.041 ms;
propagation delay for acknowledgement to reach sender: 0.2 ms – receiver can switch back to receive mode during this time.

Total cycle time: 2.241 ms. Each cycle allows one data block of 2000 bits to be delivered, but only some fraction of these are successful. For a constant probability of bit error, the probability of success is the probability of successful transmission of every bit in the frame and acknowledgement: $0.99998^{2161} = 0.9577$.

Throughput = $2000 \times 0.9577 / 0.002241 = 854.7$ kbit/s

Question 3

In a robot soccer competition, each team has 5 robots, playing in an arena which is 10 m wide and 20 m long. The robots communicate using infra-red light, transmitted using a special prism at the top of each robot, which directs light in all directions in the horizontal plane and also allows light to be received from all directions. Transmission is at 50 kbit/s, using Manchester coding, with the light either on or off.

The aim is to have each robot broadcast a message every 2 or 3 seconds, giving information such as its position, velocity, intentions, etc. Messages will be about 200 bits long, and will be encrypted so that they can only be interpreted by robots on the same team. It is accepted that transmissions will sometimes be blocked when there is another robot in the path, but as the robots are moving most of the time, it is expected that enough messages will get through to make the system work.

Each team has a leader, and this robot could take on additional communication responsibilities if necessary. The rules also allow a communication node to be installed at one corner of the arena, if required. You may make reasonable assumptions about the other details of this system, but you must state these assumptions clearly in your answers.

a) Suggest some options that could be considered in the design of the medium-access control protocol for this system, and outline the advantages and disadvantages of each, in the context of this system. Make it clear which of your suggestions would require a fixed communication node, and which would give extra work to the team leader. (60%)

This is a design question, so there are many possible answers. This is an example:

First estimate the traffic on the shared channel: Messages of 200 bits, sent at 50 kbit/s, will have transmission time 4 ms. Ten robots, each sending a message every 2 seconds, will send 5 messages per second, occupying the channel for 20 ms in every second.

Propagation delay is short: maximum distance is less than 22.4 m, so delay less than 75 ns – far less than the time it takes to send one bit.

General point: If a comms node could be installed at a corner of the arena, slightly higher than the robots, it should have a clear path to all of them all (or most) of the time. So for any protocol that needs a master device, this would be far more reliable than using the team leader as master. It would involve some extra cost, but it simplifies the protocol design (especially as there are two team leaders…).

1) Fixed TDMA – not unreasonable for a system with reasonably stable traffic, and all robots transmitting similar volumes of data. Some external timing would be needed, to mark at least the start of each TDMA frame – this could be from a comms node. Pro: simple, does not rely on all robots receiving all transmissions. Con: Needs extra node, not very flexible, possibly long delay in accessing channel.

2) Demand assigned TDMA. Simple version would use a master, but the extra complexity seems pointless, as the demand is fairly static. Sending messages requesting permission to transmit, and granting that permission, increases the load on the channel, and increases the delay in getting access to the channel, for a small improvement in flexibility. Alternatives with no master would be unreliable here, as these protocols rely on all users receiving all transmissions, and we know that transmissions will sometimes be blocked.

3) Polling. Using a master would be preferred - blocked transmissions would be a big problem for token passing. As all robots will transmit messages fairly regularly, polling at an appropriate rate should get a useful response almost every time – you could view the polling

message as an instruction to transmit rather than an invitation to transmit. This would be similar to the fixed TDMA above, but with a prompt for every transmission, not just a marker for the start of the TDMA frame. In that respect it might be preferred to option 1, as it makes the task of the robots simpler.

4) Random access. A simple "transmit when you wish" protocol could work quite effectively here, as the channel utilisation is only 2%. As we can accept some lost transmissions anyway, there is no need for robots to know that a collision has occurred. Pro: Simple system, instant access to the channel, reasonably high probability of success, no master device needed. Con: Cannot guarantee anything. More complex protocols such as CSMA could be considered, but the risk of blocked transmissions would make them less reliable, and the potential gain is small.

b) Choose one of your suggestions as the medium-access control protocol to be used, and give reasons for your choice. Specify the details of your proposed protocol, and estimate the throughput that it could provide, in terms of messages per second. (40%)

Again, this is an example:

Choose option 4. Reasons are all the pros already mentioned. The nearest alternative is probably polling, as discussed in option 3 – this could offer a better probability of success, but at the cost of more complexity and the need for a master.

Analysis is based on the ALOHA example in the lecture notes, but with no knowledge of collisions, there will be no re-transmission attempts, and no back-off algorithm. Assuming transmission attempts can be modelled as a Poisson random process, with average rate λ, probability of successful transmission is $P_S = e^{-\lambda 2T_F}$. Setting a target probability of success of 0.9 allows 13 messages per second – comfortably more than required.

Question 4

A physical-layer protocol uses a set of 4 signals to send binary data along a cable of length 30 km. Each of the four signals is represented by a different voltage, which is transmitted for 400 ns. The next signal follows immediately. The cable has attenuation 2.2 dB/km at frequencies up to 4 MHz. From the viewpoint of the transmitter, the cable looks like a resistance of 200 Ω. The average transmitter output power is limited to 100 mW.

a) What voltage would you use to represent each of the signals? Give reasons for your proposal. You may assume that all bit sequences are equally likely to occur. (30%)

The four voltages should be equally spaced, to minimise probability of error. They should be symmetrical about 0 V, to give an average voltage of 0V (a DC component conveys no information, but uses power). An example would be -3 V, -1 V, +1 V, +3 V. If all signals are equally likely to be transmitted, the mean squared voltage would be 5 V^2. With a 200 Ω load, the average power would be 25 mW. This is only 25% of the permitted power, so all the voltages could be doubled.

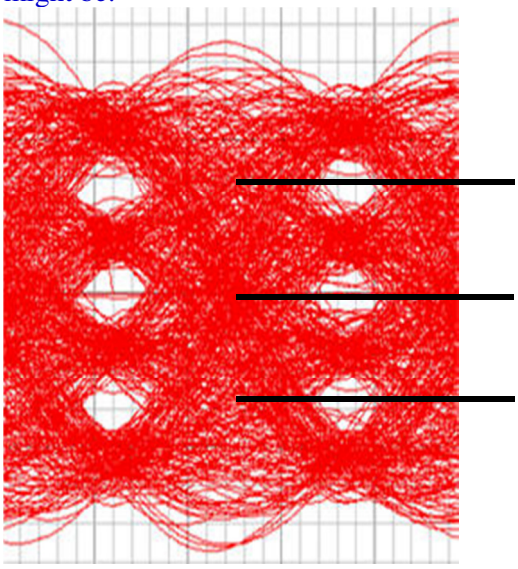
Hence propose a set of -6 V, -2 V, +2 V, +6 V. Average power will be 100 mW.

b) Explain briefly how the receiver would make decisions on the received signals, which will have been distorted by the channel, and will have some random noise added. Sketch an eye diagram, and mark the position of the thresholds that the receiver should use. (30%)

The receiver would sample the received signal in each symbol interval, and compare the sample with three thresholds. Any sample above the highest threshold would be taken as the highest

signal value. Samples below that threshold, but above the next, would be taken as the next signal value, etc.

Eye diagram should show 4 signals, consistent with voltages chosen in part a. An example might be:



Thresholds should be placed half-way between the most likely signal sample values, roughly at the positions marked by black lines in the diagram. The centre threshold would be at 0 V, as the proposed signals are symmetrical about 0 V.

c) The signal-to-noise power ratio (S/N) at the receiver is 45 dB. Figure 3 shows part of the relationship between the probability of bit error and S/N, assuming a sensible choice in part a. Explain why this protocol would not work with a cable length of 100 km. Suggest a method of achieving a reasonable probability of error at 100 km, by adding extra equipment at a point or points along the cable. (40%)

S/N at the receiver is 45 dB with a cable length of 30 km. Adding another 70 km of cable would add another 154 dB of attenuation, reducing S/N to -109 dB. Although this is off the graph, it is obvious that the probability of error would be far too high to be useful.

To operate at 100 km, the signal must be either amplified or regenerated at points along the cable. Regeneration is preferred for digital signals, as noise does not accumulate. The regenerator must be placed at a point where S/N is high, so probability of error is very low.

For example, S/N = 20 dB would give a low probability of error of about 10^{-10}. That means an additional 25 dB of attenuation could be tolerated, corresponding to an extra 11 km of cable, or a total of 41 km. So two regenerators would be needed, and these could be placed at 34 km and 68 km, to give even lower probability of error.

(One regenerator at 50 km would have S/N = 1 dB, which is too low to be useful.)