Engineering Tripos Part IIA THIRD YEAR

Module 3F5: Computer and Network Systems

Solutions to 2014 Tripos Paper

Authors: Andrew Gee and Tim Wilkinson

1. **MIPS datapath, load-store architectures**

(a) A load-store instruction set architecture is an example of a general purpose register (GPR) architecture where the operands of arithmetic and logic instructions must be located in registers, not memory. This has various consequences for the design of the CPU hardware. In particular, a load-store architecture:

- facilitates fixed length, fixed format instructions which are easy to decode;
- demands a large set of general purpose registers to store intermediate results;
- does not admit complex instructions, so the datapath can be controlled without excessive use of microcode;
- results in all instructions having similar complexity, which facilitates effective pipelining. [20%]

(b) The instruction rate is limited by the longest path through the datapath, which is for the `lw` instruction. This requires $2 + 1 + 2 + 2 + 1 = 8$ ns. So the datapath could process $10^9/8 = 125 \times 10^6$ instructions per second. [10%]

(c) (i) add3 could be mapped to an R-format instruction

op	rs	rt	rd	shamt	funct
6 bits	5 bits	5 bits	5 bits	5 bits	6 bits

using the `shamt` field to specify the third source register. This would work since the `shamt` field is otherwise redundant in add-type instructions: it is used only for shift instructions. [10%]

(ii) The required changes are:

- Register file to output three registers, with third source register identified by bits 10–6 of the instruction.
- The third register output from the register file feeds one input of a new adder.
- The other input of the new adder comes from the ALU output.
- The ALU output no longer goes directly into MUX3. Instead it goes via a new multiplexor, MUX5, which selects between the ALU output and the output of the new adder.
- A new control signal Add3 to switch MUX5. The control signal to be high for add3, low for all other instructions.

- All other control signals the same for add3 and add. [25%]

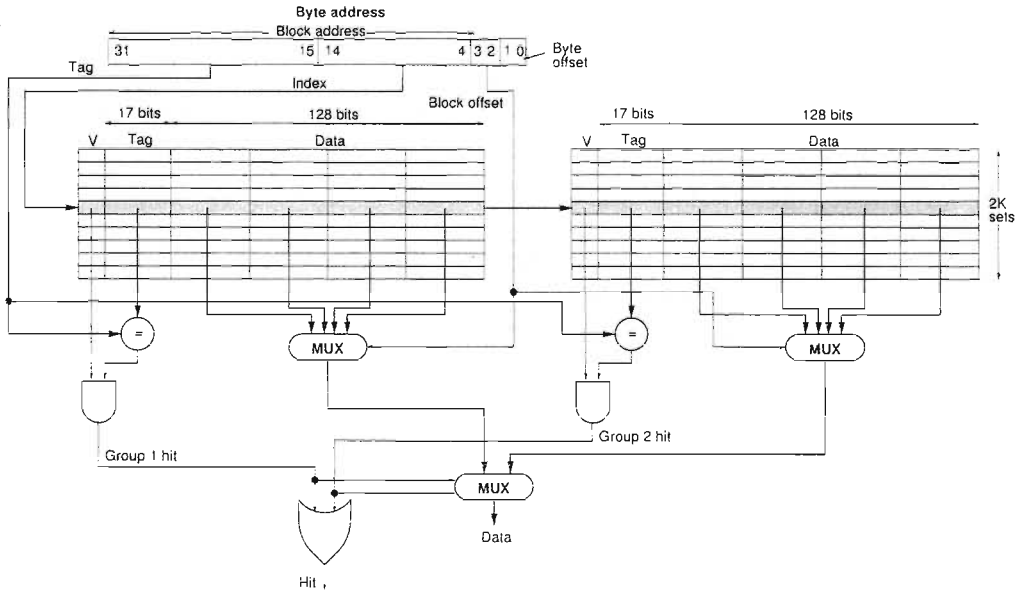
(iii) The add3 path through the datapath requires $2+1+2+2+1=8$ ns, the same as for `lw`. So the datapath can still process 125×10^6 instructions per second. [10%]

(iv) At first sight, add3 would appear to do the job of two add instructions, providing an apparent $2\times$ speed-up. But Amdahl's Law tells us that the actual speed-up will depend on how much the add3 instruction is used. In the most extreme case, where the program does nothing but add a large number N of numbers, the instruction mix will be approximately $N\times$ `lw` (since the numbers need to be fetched before adding) and $N/2\times$ add3 (since each add3 can add two further numbers to the running total). This assumes complete loop-unrolling, in practice there will be a fair number of branches in the mix too. So the asymptotic limit on the proportion of add3 instructions is 33.3%, and Amdahl's Law then puts the overall speedup at $1.2\times$. But even this extreme-case speedup is doubtful, since we are assuming no change in the CPU clock speed: recall, the extra adder for add3 took the same time as the data cache access in `lw`. If the extra adder were to take longer than a data cache access, or just generally add some parasitic capacitance, we would need to reduce the clock speed to accommodate it. [25%]

2. **Caches and locality of reference**

(a) Direct mapped caches have the lower hit time, since the index points to a unique block and no searching of the cache is required. However, since there is no choice of which block to replace on a miss, we might replace blocks that are going to be referenced again soon: this will lead to a high miss rate. In a set-associative cache, the index points to a (typically small) set of blocks that must be searched, increasing the hit time. But there is some flexibility as to which block to replace on a miss. We could, for example, consider temporal locality of reference and replace the least recently used (LRU) block, thereby reducing the miss rate. [20%]

(b)



[40%]

(c) Matrices are stored in memory row by row, so the elements of `a` are accessed sequentially inside the inner loop, with excellent spatial locality of reference and subsequent cache hits. The same cannot be said of `a_t`, where the access pattern (in bytes) is:

0 R 2R 3R ... 1 R+1 2R+1 3R+1 ... 2 R+2 2R+2 3R+2 ...

So at each iteration of `i` there are `C` accesses at a stride of `R` bytes, before wrapping back to a similar set of addresses (just shifted along by one byte) at the next iteration of `i`.

Now consider the four different scenarios. Note that in each case there are 200×10^7 matrix elements to process, so any difference in execution time is not due to the total operation count, but most likely cache effects.

Starting from the left, with a stride of 100 bytes we do not seem to be getting cache hits as we access each element of `a_t` (the block size appears to be less than 100 bytes). Furthermore, we do not wrap back to the beginning of `a_t` until 2×10^7 accesses later, by which time the first block of `a_t` would most likely have been replaced in the cache. So we get mostly cache misses and a high execution time.

Next, with a stride of 2×10^7 bytes we are not going to get cache hits as we access the first 100 elements of `a_t`. But access 100 will probably be to the same block as access 0, which is most likely still in the cache. Likewise for access 101, which will be adjacent to access 1, and so on. So we get cache hits and a considerable reduction in execution time.

Next, with a stride of 2 bytes we are going to be getting lots of cache hits from the start (the cache block size of modern processors is typically 64 bytes), without having to wait for the wrap-around after 100×10^7 accesses. So another low execution time.

Finally, even though the stride is 100×10^7, the wrap-around comes after just two accesses, so we will start getting cache hits almost immediately, and another low execution time. [40%]

Question 3

a) Voice services: Voice services are intolerant to any form of delay (fixed or variable). If there is any significant delay (more than 20-30msec) then reflections can lead to echoes being generated. Voice services are tolerant to transmission errors. We can put up with considerable distortion to a voice signal, hence the tolerable bandwidth limits of voice are set at 3.4kHz.

Voice data was originally sent over circuit switched networks which is an example of a connection oriented transport service (COTS). In a connection oriented technique a circuit, virtual circuit, connection or virtual connection (VC) is established between sender and receiver before information is transported. This is designed to minimise delays.

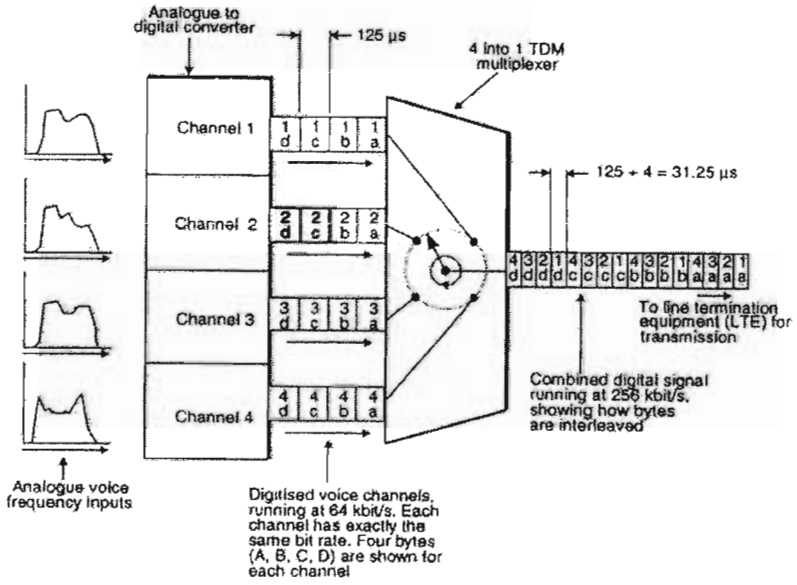
Voice services are connection oriented and often use path oriented routing. A fixed path is chosen for a given logical channel (virtual circuit) at the time of call set-up. The path itself is chosen based on the current loading of the network and the available topology. Should any link in the path become unavailable during the course of the call (say, because of transmission failure), then an alternative path is sought, without breaking the connection. Packets are stored until a new path is found but delays are minimised. The advantage of path oriented routing is that the packets pertaining to a given local connection all take the same path, all suffering about the same amount of queuing delay in the buffers and arrive pretty much in the same order as they were sent (allowing for lost packets along the way). The disadvantage of path oriented routing is inflexibility at high traffic rates.

Data services: Data services are highly sensitive to bit errors. Data is nearly always sent as binary bits and a transmission error of a 0 instead of a 1, could be catastrophic (especially in my bank account). Data networks are much more tolerant to delays in transmission. Often there is no penalty for delay as long as the message gets there.

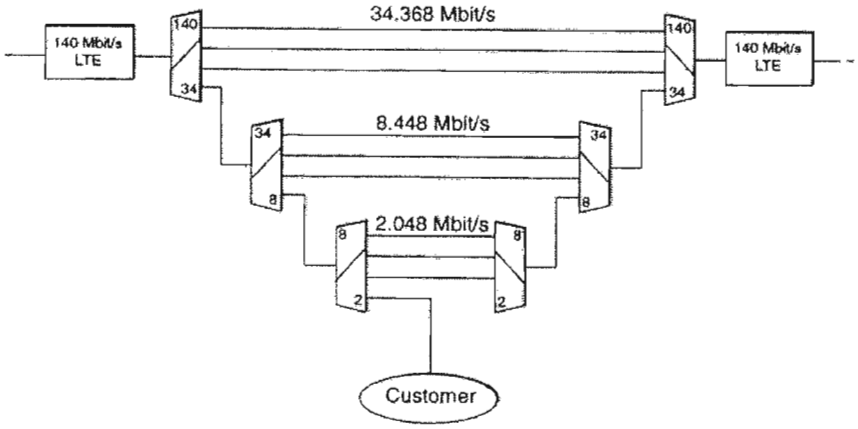
Data services often use connectionless switching techniques or a connectionless network service (CONS). This allows messages to be dispatched, maybe without checking the validity of the address. The advantage of a connectionless service is that the sender does not need to wait for the receiver to be ready to accept the message and the network does not need to make a complete connection. The disadvantage of this system is that the sender never knows if their message has been received, hence there is no control of delay. Data is sent without knowing if the receiver is connected or what route will be taken. Data is often switched using datagram based routing systems. This allows for more dynamic routing of individual packets and thus has potential for better overall network efficiency, however delay is not normally guaranteed as there is a lot of buffering required and often queuing. But the technique requires more sophisticated equipment and powerful switch processors (routers) capable of determining routes for individual packets.

b) Circuit switched networks work well and are ideally suited to voice communications, however they are limited in scale. As the number of subscribers and trunks increases so does their physical size and space requirements. Wire density is limited by space and Strowger exchanges became huge in order to house the switches. Hence it became necessary to multiplex data streams to allow multiple subscribers on each physical wire. This was further enhanced by invention of pulse code modulation (digital circuits) and the optical fibre (huge bandwidth). PCM converts 4kHz BW to 64kbit/sec digital stream, which avoids noise and other analogue problems. More importantly it allowed many PCM channels to be multiplexed together using Time Division Multiplexing (TDM) to increase the BW used in the transmission medium. Many 64kbit/sec channels could be byte interleaved onto a high capacity link. The 64kbit/sex data rate gives a byte period of 125usec which

is the fundamental time unit in the TDM process. Each Mux must repeat every 1325usec to maintain a continuous voice service.

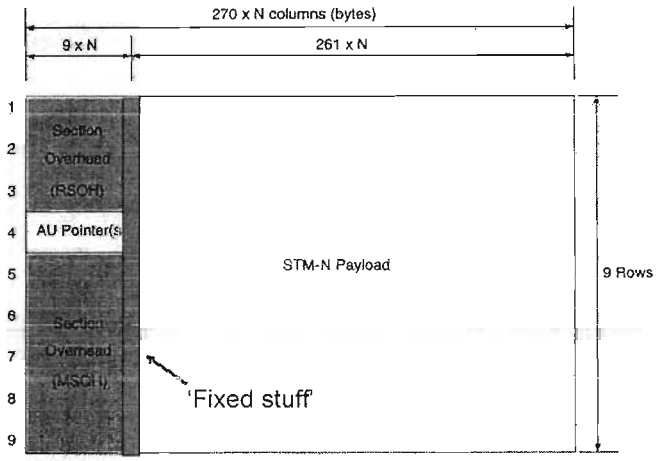


c) The problem occurs when a single 2Mbit channel needs to be dropped or added from a higher order Mux. (Drop and Insert). All of the channels must be demuxed and then muxed again as the position of a channel is unknown in the higher orders. ie 140 to 34, 34 to 8 and then 8 to 2Mbit/sec. This is very expensive as it requires a lot of equipment and invariably leads to delay in any voice services.



There is also inadequate provision of network management, which is important when combining voice and computer data. The aim of SDH was to correct the defects of PDH.

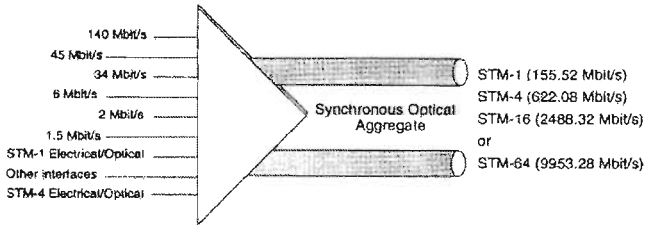
- ∝ Synchronisation. All elements of the system are synchronised to the same master clock. All tributaries are at a common rate before Mux. Each voice channel has 1 byte per STM-1 frame
- ∝ Pointers. Information about the muxed signals is transmitted at fixed intervals, which indicate the positions of units within the mux process.
- ∝ Control and management. Time slots are put aside for a variety of tasks in ensuring the synchronicity of voice and data services. Guaranteed bandwidth



1st basic SDH mux unit is the STM-1 (synchronous transport module) frame which is at a data rate of 155Mbits/sec. STM-1 signal is a repeated series of 125usec frames of 270x9 (2430) bytes. 155Mbit/sec = 243 x 64kbit/sec

Positions in the frame are given to data bytes from different sources and pointers are set at the start of each frame to allow the data to be found at Demux. Each frame should contain one byte from each voice mux to maintain delay free service. The 2D structure of the frame is created to synchronise the clock. The Fixed Stuff section is a segmented synchronisation code which is repeated 9 times per STM-1 frame. The STM-1 frame must be capable of muxing a variety of different sources, including all of the old PDH rates. Hence there are standards to map 1.5 – 140Mbits/sec onto STM-1. The SDH standard defines a series of containers, each of which corresponds to a PDH rate. These are basically time slots.

d)

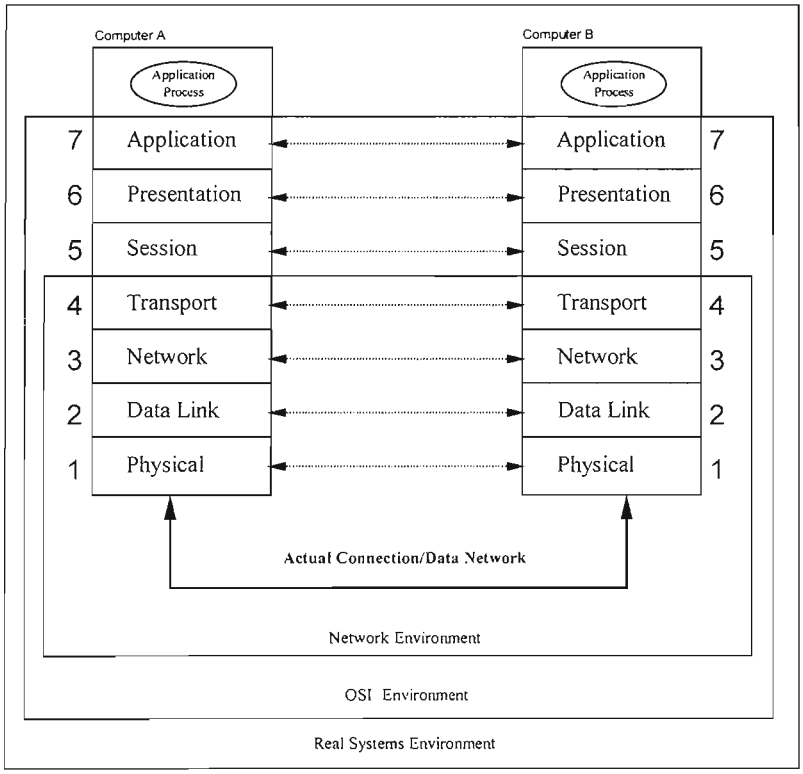


How can we accommodate both voice and data on SDH? SDH has been designed for low latency and minimal delays, which is ideal for voice. The synchronous Mux has been designed with flexibility in mind which allows different data type to ne mapped into STM-N frames with careful management. Data (especially Internet data) tends to be bursty and has been suited to packet switched networks. This more difficult to map onto SDH, especially compressed video streams which are bursty and delay sensitive. Video can be mapped effectively as it has a continuous requirement of bandwidth, but the more bursty internet traffic is more difficult as it requires more management overhead to map onto SDH and is not so efficient to manage within the network.

Asynchronous transfer mode (ATM) was designed to allow computer data to be sent within SDH frames. A 48byte cell + 5byte header. The advantage of the STM-1 frame is that it offers a huge wealth of virtual bandwidth as well as management services which will make ATM or IP over SDH easier than just pure ATM or IP systems.

Question 4

a) For successful data communication across a network, appropriate operating procedures must be established. They must be specified in detail and strictly adhered to by the sending data terminal (or computer) and any intervening switching centres. These procedures are called protocols. Many local area networks (LANs) interconnect data terminals (or computers) from the same manufacturer and operate using proprietary protocols. The need arose for communication between computers and terminals from different manufacturers. Open systems interconnect (OSI), to enable networks to be machine independent.
Computers may use different languages, data formats and operating systems; hence, the interface between user (application) programs, normally referred to as applications processes, and underlying communications services may be different. Development of the necessary specifications and protocols or OSI was undertaken by the international standards organisation (ISO). The ISO standards are based on a seven layer protocol known as the ISO reference model for OSI. Both the network dependent and application oriented (network independent) components of the OSI model are in turn implemented in the form of a number of protocol layers. The boundaries and processes for each layer have been selected based on experience gained from other standards in the past.



The OSI model can be broken up within this system into 7 separate layers, each with a clearly defined purpose and protocol.

- Each layer is a service user to the layer below and a service provider for the layer above.
- Each layer is specified independently of the other layers;
- Each layer has a defined interface with the layer above and below.

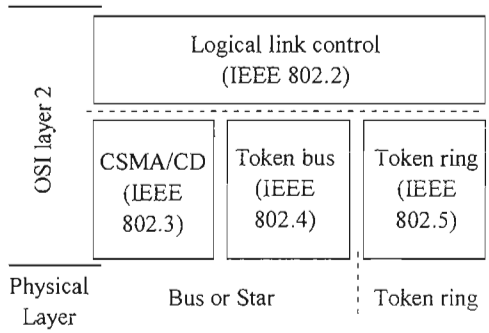
As far as users are concerned, communication appears to take place across each layer. Each data exchange passes down to the bottom layer (the physical layer) at the sending terminal, crosses the network to the receiving terminal and then passes up again.

Data communication between layers is done through the addition and reading of headers on the data

b) Layer 3: **The network layer**. This is concerned with the operation of the network between the terminals. It is responsible for establishing the correct connections between the appropriate network nodes, including network routing (addressing) and in some instances, flow control across the computer to network interface. Routing and network management at the packet level are done in this layer.

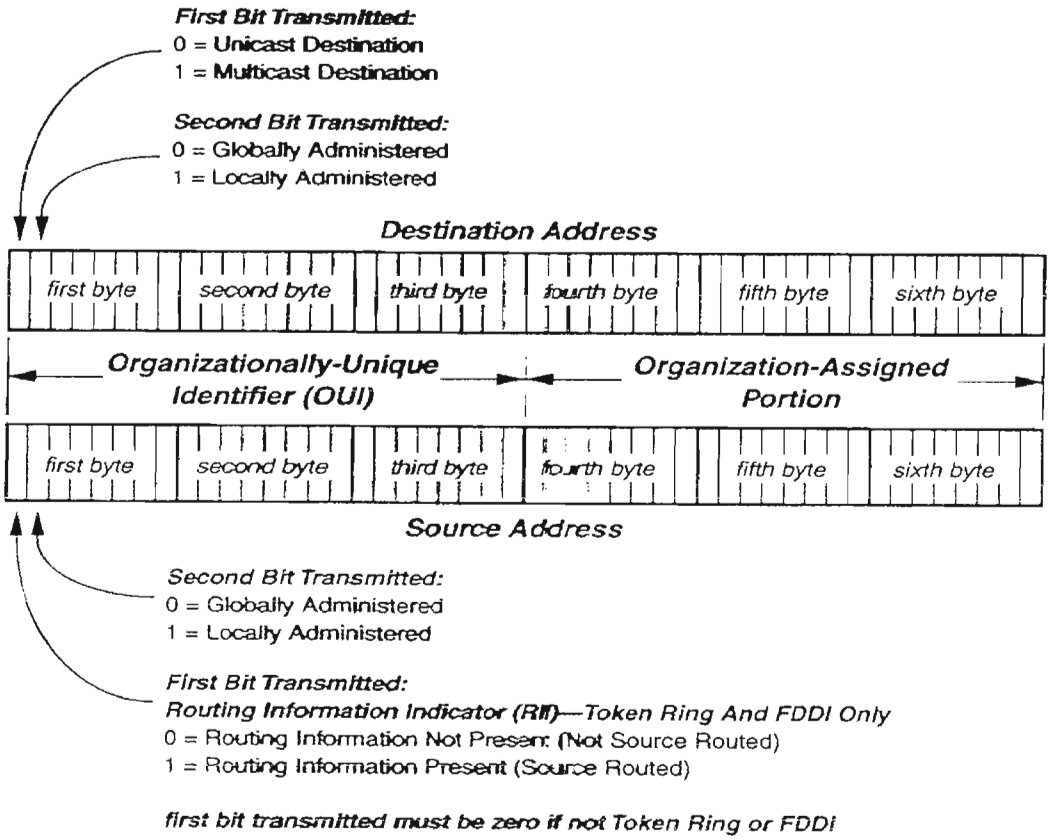
Layer 2: **The data link layer**. This provides error detection and correction for a link to ensure that the exchange of data is reliable. Provides error free data to the network layer. Adds control information to the data blocks and a frame check sequence (FCS) for error-checking or bits for

synchronisation. Retransmits data if errors have occurred. The data-link layer is also responsible for determining the length of data segments to be sent to the physical layer. If the data provided by the network layer is too long, then it is broken into suitable sized packets. In the case of LANs, the data-link layer provides local area address management on top of that provided by the network layer through medium access protocols (MACs).



Layer 2 is split into the LLC and the MAC. This was to split the control of the LAN from the media dependant functions. The different types of LAN are characterised by their distinctive topologies. They all comprise a single transmission path interconnecting all the data terminal devices, together with the appropriate protocols (called the logical link control (LLC) and the medium access control (MAC)) to enable data transfer. The LLC and MAC split the data-link layer (Layer 2)

c) The key to MAC layer protocols is a uniform addressing structure so that LAN hardware can easily identify stations on the network and transmit packets between them.



A MAC address is 48 bit long (6 bytes) and the address space is split into two halves:

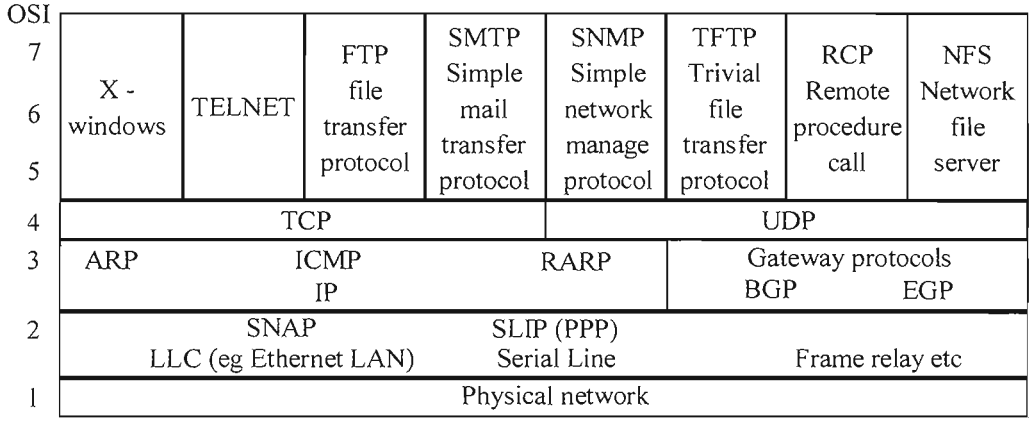
- A *unicast address* identifies a single device or network interface. The source address is always unicast. Often referred to as *individual, physical* or *hardware addresses.*
- A *multicast address* identifies a group of logically related devices. Often referred to as *group* or *logical addresses.*

The first bit of the destination address defines if it is a unicast (0) or a multicast (1) address. Source addresses are always unicast so the first bit is zero except in token ring or FDDI, where is describes how the packet is routed. The second bit of the address defines whether the address is globally unique (0) or locally unique (1) to the LAN. Globally unique addresses are assigned by the

manufacturer, whereas locally unique addresses are assigned by the LAN administrator, carefully! The 48-bit MAC address is divided into two parts. The first 24 bits constitute the organisationally unique identifier (OUI), which indicates which organisation (typically the manufacturer) is responsible for assigning the remaining 24 bits of the address. A MAC address is usually expressed in canonical byte form *aa-bb-cc-dd-ee-ff* where the first 3 pairs of bytes are the OUI and the last 3 pairs of bytes are set by the manufacturer. An address is read *aa* byte first.

The address evolved in layer 2 as LANs were originally only designed for local connection so a simple hardware address was used to identify all end users. Connection between LANs was originally managed by bridges which switch purely on the basis of the MAC address structure. As LANs got more complex and started to connect globally, then the MAC address became globally unique, but also restricted by the layer 2 functionality. This was alleviated by the inclusion of layer 3 functions which allow more network oriented processes and can be used to manage large networks. This also included the layer 3 address which is where addressing should be handled. Hence we have been left with the MAC address as a method of locating users locally. This is not a true OSI definition, however it allows us to uniquely identify hardware as well as manage network addressing in a separate manner.

d)



Mapping the destination to a local data-link address (ARP mapping) – The structure of the station identifier section of the IP address does not provide a simple mapping onto 48 bit data link addresses. It is not possible to determine the 48 bit data link address solely from the station identifier. Hence for packets destined for a locally connect network (such as the last hop port) must undergo a second look up process to find the destination station. In some instances this could be a separate look up operation in the ARP cache or continuation of the router look up process. Either way it will comprise one of three classes:

1. The packet is destined for the router itself. ie the destination IP address corresponds to one of the IP addresses of the router. This packet will be passed the higher layer entities in the router.
2. The ARP mapping for the indicated station is unknown. In this case, the router must initiate a discovery procedure (ARP request). This may take some time (it is a form of layer 3 flooding) and could result in the packet being dropped. This is not part of the fast path. ARP requests are not normally part of the steady state operation of a router.
3. The station is destined for a known station on the directly attached network. In this, very common case, the router successfully determines the mapping from the ARP cache and continues the routing process