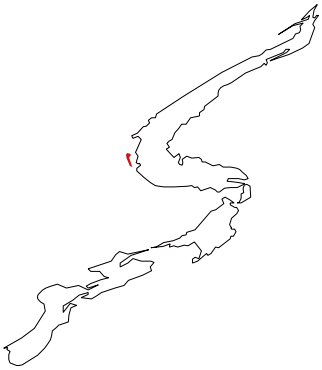
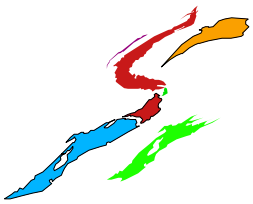


DLC Continued

Dr. G. A. Marin

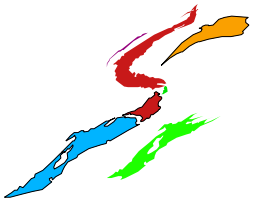
Problem Set 2: Chapter 3

- Problems 1 through 10
- Problems: 11,12,14,19,24,25,27,30,31
- Due Date: 2/7/2001



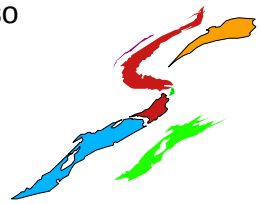
ACK and Bandwidth Efficiency

- Clearly acking each frame may be inefficient
- Example: send 1000-bit frames over satellite link on a 50-kbps channel with 500 ms roundtrip delay.
  - t=0 start sending first frame
  - t=20 msec (1000bits/50kbps) finished sending first frame
  - t=270 ms last bit of first frame arrives at destination
  - t=520 ms first bit of ack arrives back at sender.
  - Result: sending max of 1000 bits/520ms or 1923 bits/sec on 50,000 bps channel!
  - Channel is 3.8% utilized
- Note satellite channel is among worst cases



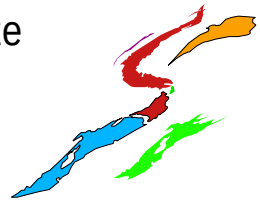
Windowing/Pipelining

- Last time we considered windows larger than 1 to allow sender to send more than 1 frame at a time (called pipelining)
- In general:
  - channel capacity b bps, frame size n bits, round trip prop time R secs
  - Time to xmit 1 frame is n/b secs and total roundtrip is R.
  - utilization is n/b / (n/b+R) = n/(n+bR)
  - window size should be equal to (bandwidth)R bits
  - In satellite example 50,000bpsx0.5sec = 25,000 bits so window size should be 25 or 26 frames (say 26).



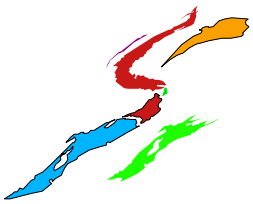
When frame in middle of window is lost...

- Go Back n
  - Receiver (Rx) discards all frames after lost or damaged one (Rx window of 1)
  - Sender fills his window then waits until timer pops
  - Sender then retransmits all unacked frames in order
  - Can waste bandwidth if high error rate



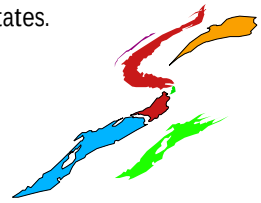
Lost/damaged frame (continued)

- Selective repeat
  - Rx DLC stores all correct frames following bad one
  - Only acks good frames at lower edge of Rx window
  - Sender eventually times out on single un-acked frame and retransmits it alone.
  - Receiver quickly acks all good frames.
  - Note receiver must have DLC buffer space available.



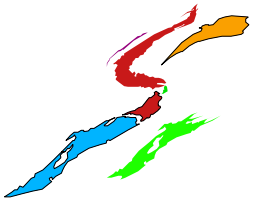
Protocol Specification/Verification

- At all layers specifying and verifying protocols is a major challenge
- Two major techniques used are Finite State Machines and Petri Nets.
- Finite State Machine
  - Protocol machine (Tx or Rx) represented as being in one of many states.
  - Generally take "waiting" states and in addition describe state by values of all (key) variables - n variables means 2^n states.
  - Model receiver states and sender states plus channel states to get model of entire system.
  - Initial state is some convenient starting point
  - From each state there are zero or more transitions to other states.
  - Thus, FSM is a quadruple (S,M,I,T) where:
    - S is set of states for receiver, sender, and channel
    - M is set of frames that can be sent over channel
    - I is set of initial states
    - T is the set of transitions possible among states.



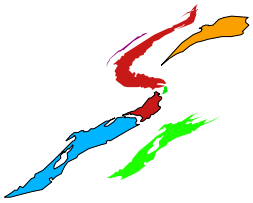
Common Errors that FSMs Find

- Incompleteness - A certain frame occurs in a certain state and the FSM does not say what to do.
- Deadlock - There is a state (or more than one) from which there is no exit and from which no progress can be made.
- Extraneous transition - The protocol specifies how to handle an event in a state where the event cannot occur.
- Unreachability - There is a defined state or states that can never occur.

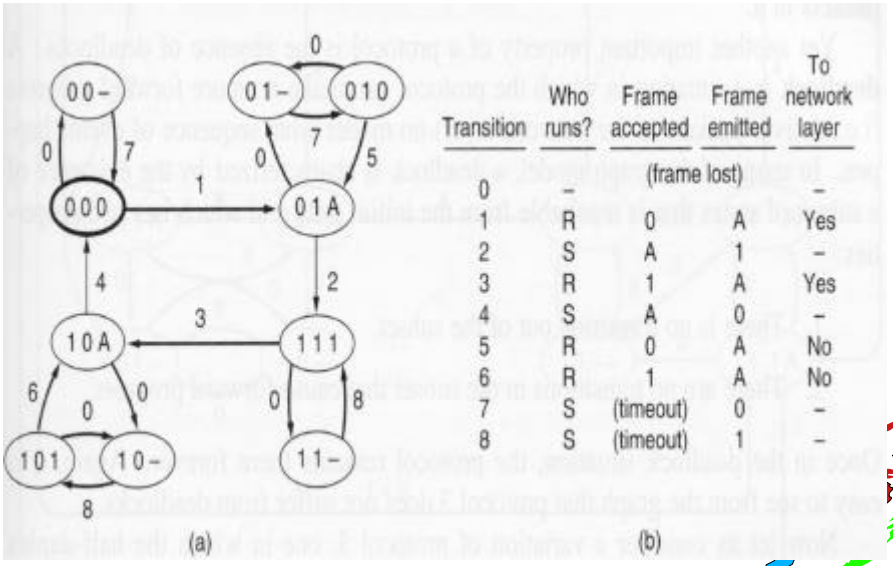


Ex: One-way data flow over unreliable channel

- Tx uses seq number 0 and 1 only
- Tx window size is 1
- Rx window size is 1
- Rx acks good data (acks not numbered)
- Data or acks may be lost on the channel
- Two losses may NOT occur consecutively
- Tx always has data to send
- Rx NL is infinitely fast



FSM for This Protocol: (sender,receiver,channel)=(0/1,0/1,0/1/A/-)

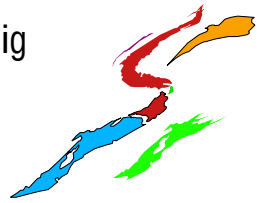


Transition	Who runs?	Frame accepted	Frame emitted	To network layer
0	–	(frame lost)		–
1	R	0	A	Yes
2	S	A	1	–
3	R	1	A	Yes
4	S	A	0	–
5	R	0	A	No
6	R	1	A	No
7	S	(timeout)	0	–
8	S	(timeout)	1	–

(b)

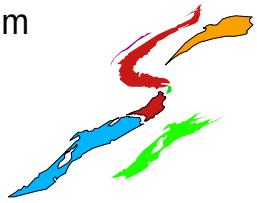
Checking...

- Packets delivered in sequence order: 0,1,0,1,0,1...
  - transition 1 -> trans 3 -> trans 1...
- Sender must not change state twice 0 to 1 to 0... without receiver changing
  - would result in two frames lost forever
- No deadlocks
  - Don't get stuck in some subset of states with no useful work
- ASSIGNMENT: practice walking through Fig 3-21.

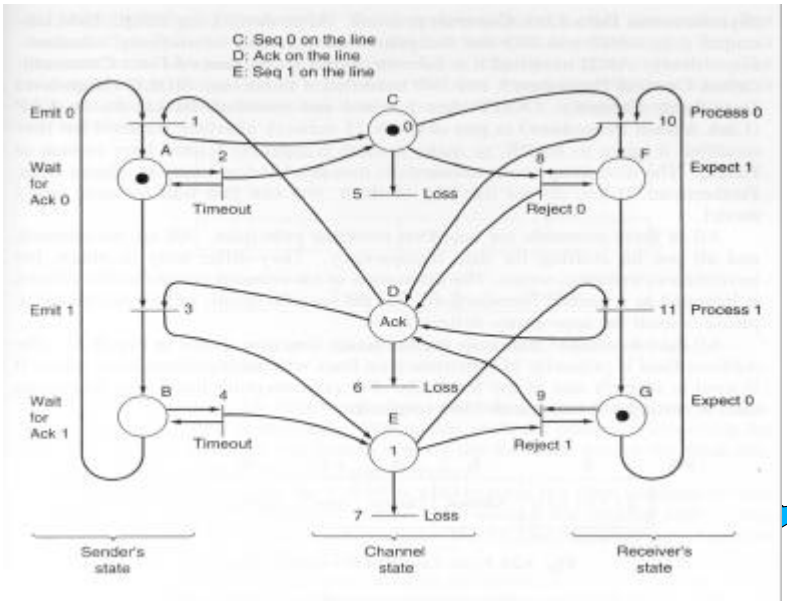


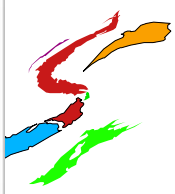
Petri Net

- Four elements: places, transitions, arcs, tokens
  - place: state that system is in
  - token: indicates which state is active
  - transition: indicated by horizontal bar
  - each transition has input arcs and output arcs coming from input places and going to output places
  - transition enabled if token in each input place
  - transition fires (at will) and moves tokens from each input place to each output place



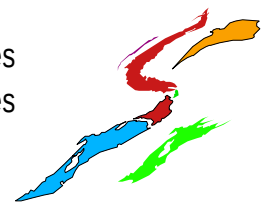
Petri Net Example



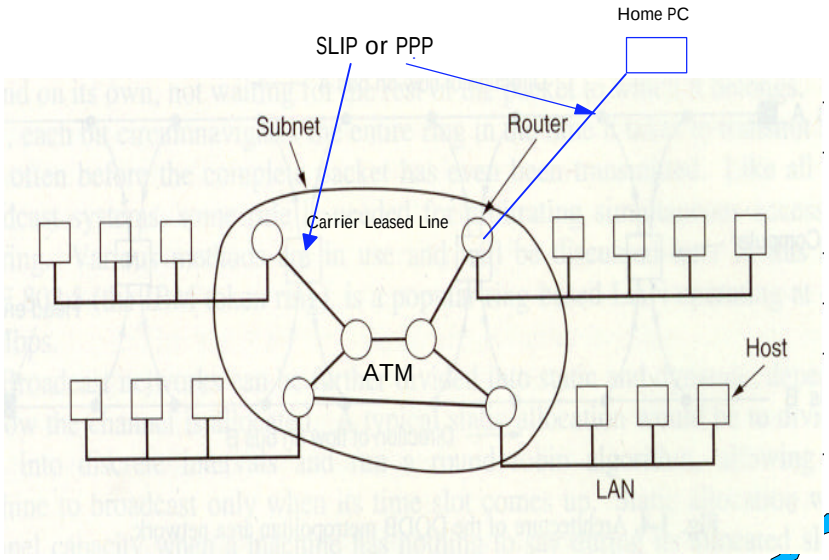


Transitions in Petri Net Example

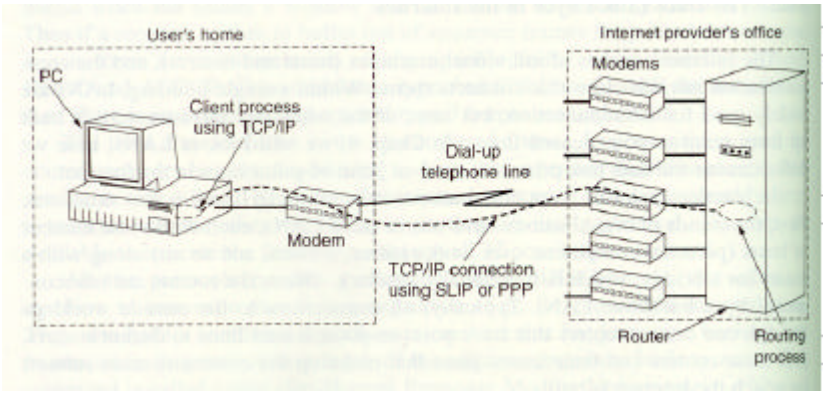
- Trans 1: normal send of frame 0
- Trans 2: send timeout and resend 0
- Trans 3: normal send of frame 1
- Trans 4: send timeout and resend 1
- Trans 5: lose frame 0
- Trans 6: lose ack
- Trans 7: lose frame 1
- Trans 8: reject 0 because expecting 1
- Trans 9: reject 1 because expecting 0
- Trans 10: accept 0 because expecting 0 and arrives
- Trans 11: accept 1 because expecting 1 and arrives



Typical Business Network

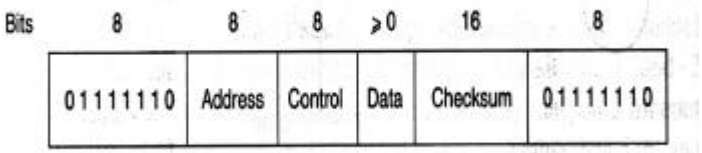


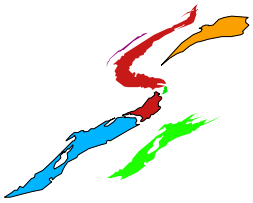
Typical Home Connection



Example Data Link Protocols (Business)

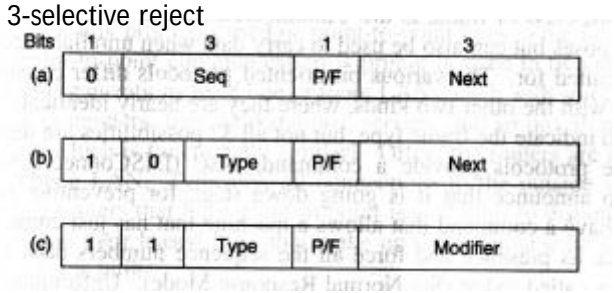
- HDLC(ISO) (similar to SDLC(IBM), ADCCP(ANSI), LAPB(CCITT))
- Bit oriented
- Bit stuffing for "data transparency"
- Checksum uses CCITT generator polynomial.
- Format below for point-to-point or polling.
  - control field used for seq no, acks, etc.

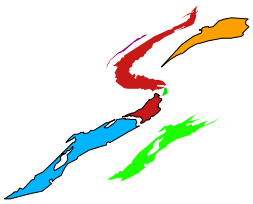




Control Fields: (a) Info Frame, (b) Supervisory Frame, (c) Unnumbered Frame

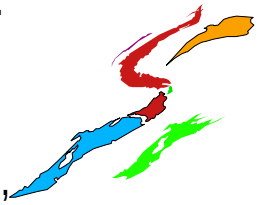
- Sliding window, 3-bit seq no, 7 acks max outstanding
- Seq in (a) is frame seq number
- Next is for piggybacked ack
- P/F used in polling
  - P invites terminal to send data
  - Terminal responds with all P's except Final
- P may be used to force supervisory frame containing ack.
- Type indicates kinds of supervisory frames (0-ack,1-nak,ack but NR, 3-selective reject





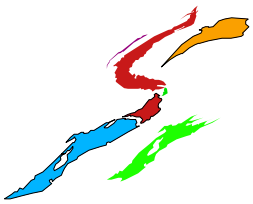
Serial Line IP (RFC 1055)

- Link layer sends IP packets with 0xCD at end for framing
- 0xDB,0xDC sent if 0xCD occurs in data (char stuffing)
  - 0xDB also "stuffed" if inside data
- Sometimes compress headers by leaving out fields if same as previous packet.
- No error detection or correction.
  - Left up to higher layers
- Supports IP only (not Appletalk, IPX, APPN,...)



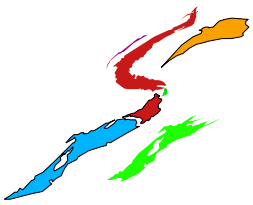
Point-to-Point Protocol: PPP (RFC 1661,1662,1663)

- Supports error detection, multiple protocols, and authentication
- Includes flags to start and end frames
- Includes link control protocol (LCP)
  - negotiate: max payload size, authenitication use, protocol used, header compression options, line quality testing, CRC length...
- Includes network control protocol (NCP) appropriate to supported network layers
  - For TCP/IP supports dynamic addressing

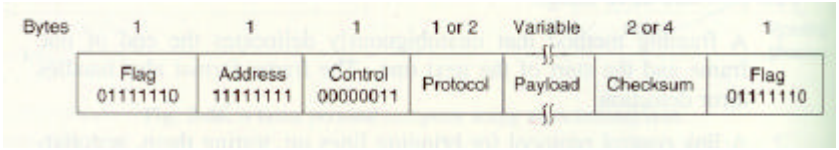


Example: PPP Setup and Takedown (modems)

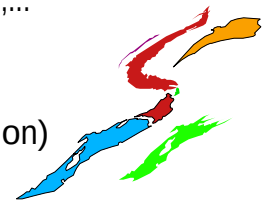
- Modems establish physical connection
- PC (host) sends PPP frames to internet router
- LCP used to establish initial link parameters and authenticate if desired.
- NCP (IP) configures network layer
  - PC gets IP host address from router
- Data transmission phase
- NCP releases network layer connection
  - Frees buffers
  - Frees IP address for use elsewhere
- LCP shuts down DLC



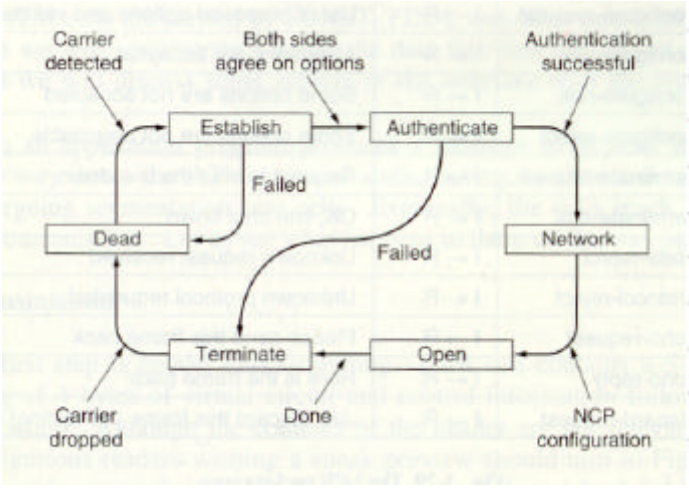
PPP Frame Sturcture

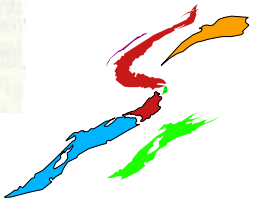


- Start/End of Frame Flag: 01111110
- Address: 11111111 "all stations"
- Control Field: Unnumbered Frame
  - note no seq numbers and no acks
- Protocol ID: LCP,NCP,IP,IPX,Appletalk,APPN,...
- Payload field defaults to 1500 bytes
  - Can be negotiated using LCP
- Checksum (for CRC) is usually 16 bits (32 option)



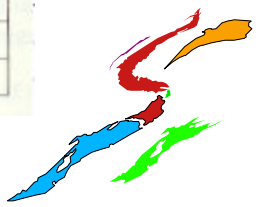
PPP Setup and Takedown Sequence





LCP Frame Types

Name	Direction	Description
Configure-request	I → R	List of proposed options and values
Configure-ack	I ← R	All options are accepted
Configure-nak	I ← R	Some options are not accepted
Configure-reject	I ← R	Some options are not negotiable
Terminate-request	I → R	Request to shut the line down
Terminate-ack	I ← R	OK, line shut down
Code-reject	I ← R	Unknown request received
Protocol-reject	I ← R	Unknown protocol requested
Echo-request	I → R	Please send this frame back
Echo-reply	I ← R	Here is the frame back
Discard-request	I → R	Just discard this frame (for testing)



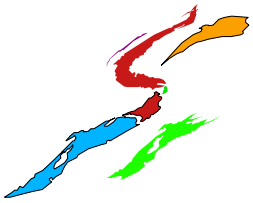
BISDN Architecture

- envisioned for *universal networking...*

- integrated networking
  * *voice, video, data, and image in the same network*

- scaleable in distance
  * *LAN, MAN, WAN*

- scaleable in bandwidth
  * *1.5 Mbps to several Gbps*

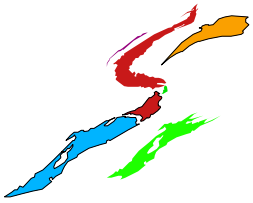


Motivations

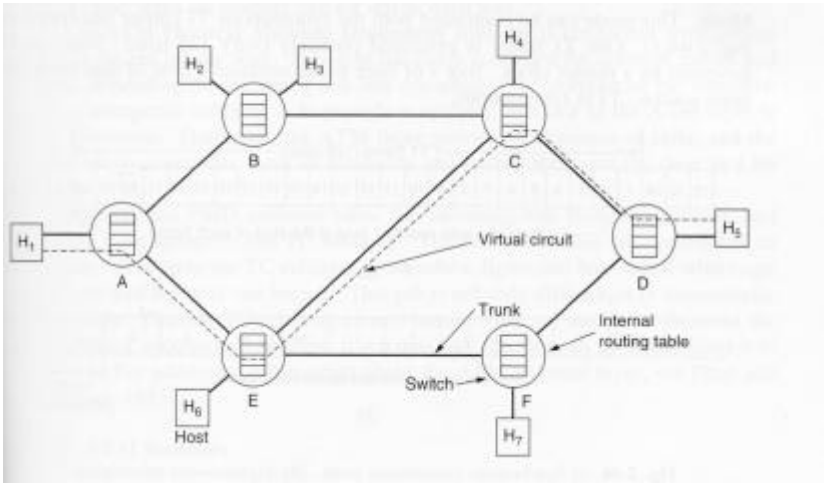
Why connection oriented?
- providing service guarantees require resources to be reserved in the network
- simpler network management

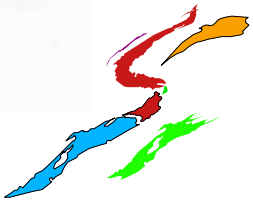
Why fixed size cells?
- efficient switching
- hardware switching

Why small size cell?
- mainly for voice

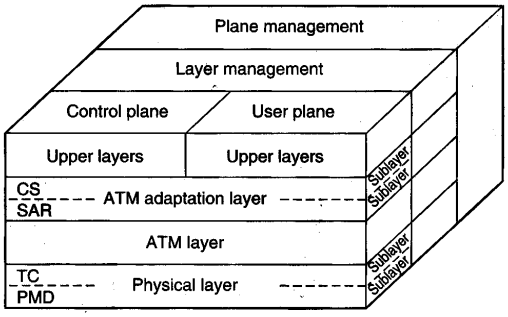


PVC and SVC (ATM connections)

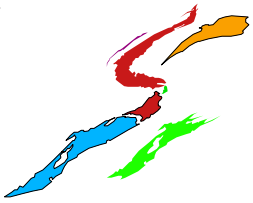




ATM Reference Model

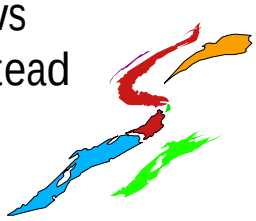


CS: Convergence sublayer
SAR: Segmentation and reassembly sublayer
TC: Transmission convergence sublayer
PMD: Physical medium dependent sublayer



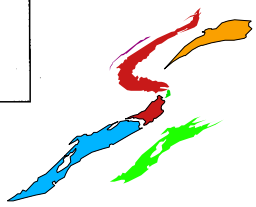
ATM Layer Functions

- Physical: voltages, bit timings, carrier synch, carrier structure (T1,T3,SONET...)
- ATM: Layout of cells (53 bytes) definition of header fields, establishment and release of virtual circuits
- AAL (ATM Adaptation Layer): allows users to submit packets to ATM instead of cells, does segmentation and reassembly (SAR) if needed

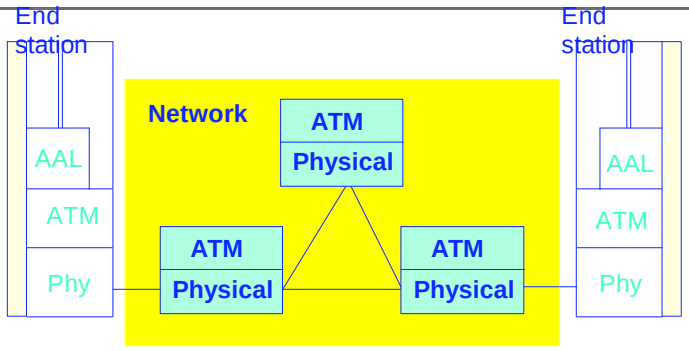


ATM Layer Functions

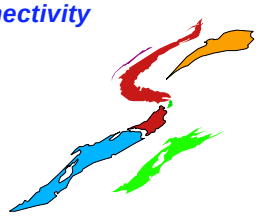
OSI layer	ATM layer	ATM sublayer	Functionality
3/4	AAL	CS	Providing the standard interface (convergence)
		SAR	Segmentation and reassembly
2/3	ATM		Flow control Cell header generation/extraction Virtual circuit/path management Cell multiplexing/demultiplexing
2	Physical	TC	Cell rate decoupling Header checksum generation and verification Cell generation Packing/unpacking cells from the enclosing envelope Frame generation
1		PMD	Bit timing Physical network access



ATM Network

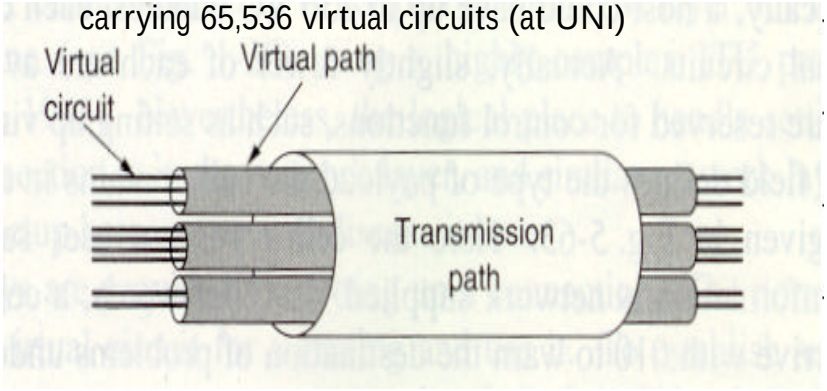


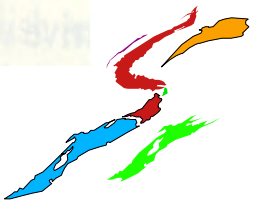
An ATM network provides an ATM layer connectivity among ATM end stations



Virtual Circuits in Virtual Paths

256 virtual paths possible on each link with each path carrying 65,536 virtual circuits (at UNI)



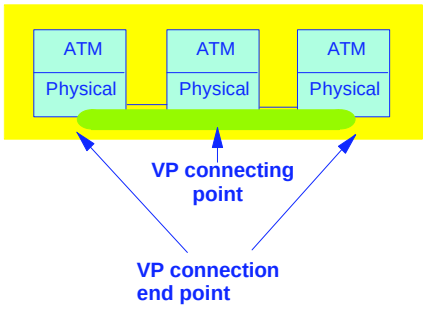


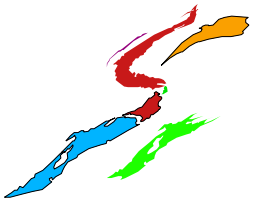
VPIs and VCIs

- pre-established, semi-permanent connections
- allocated fixed bandwidth

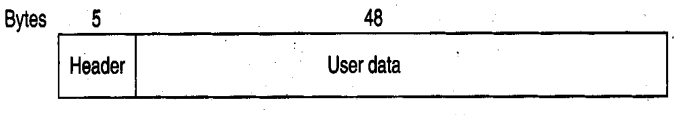
- VCI values are unique only in a given VPI
- VPI values are unique only in a given physical link

Incoming link id + VPI/VCI	Outgoing link id + VPI/VCI

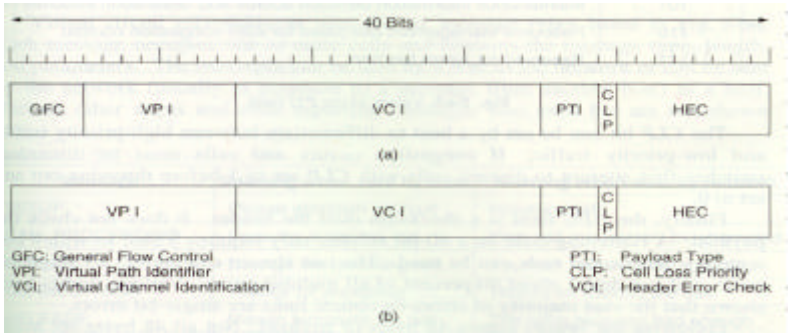


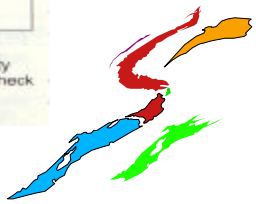


ATM Cell Structure



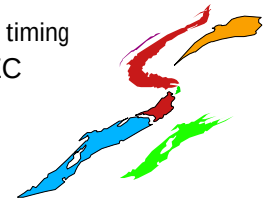
Header Structure





Function of Transmission Convergence Sublayer

- Take a sequence of cells (from ATM layer), add a HEC Byte to each header, produce output bitstream, and match physical layer
- HEC (Header Error Control) Byte
  - Checks header only
  - Corresponds to $x^8 + x^2 + x + 1$
  - Catches all single-bit errors
  - Prob of undetected multibit error = 10^{-12}*P(1-bit error)
- Match transmission media
  - If asynchronous, no timing restrictions
  - If synch, match timing pattern
  - May send idle cells if no data
  - Operation and Maintenance (OAM) cells used for control and timing
- Receiving TC layer locates frame boundaries and checks HEC
  - Usual case is SONET where frame structure points to cell
  - Otherwise use HEC with bit-shift
  - But probability of random pattern match is 1/256 so...



Synch States and Transitions