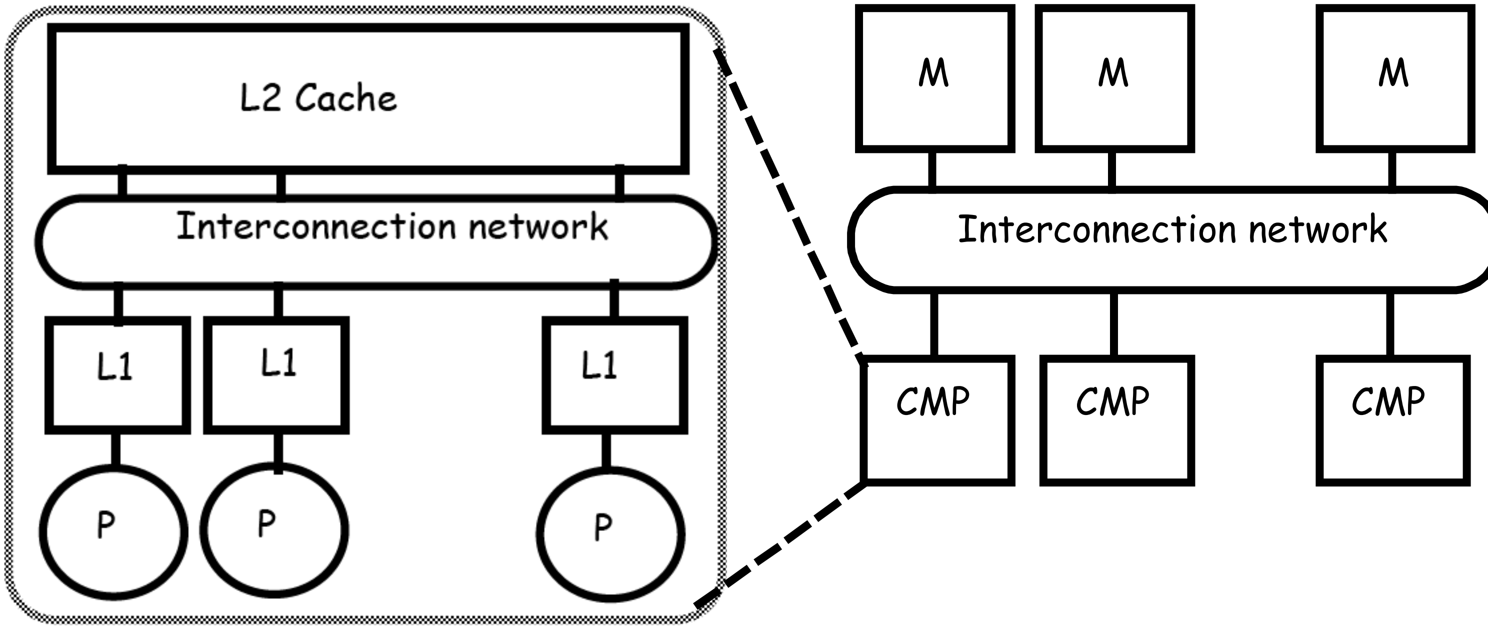
CHAPTER 6

INTERCONNECTION NETWORKS

- DESIGN SPACE
- SWITCHING STRATEGIES
- FLOW CONTROL
- TOPOLOGIES
- ROUTING TECHNIQUES

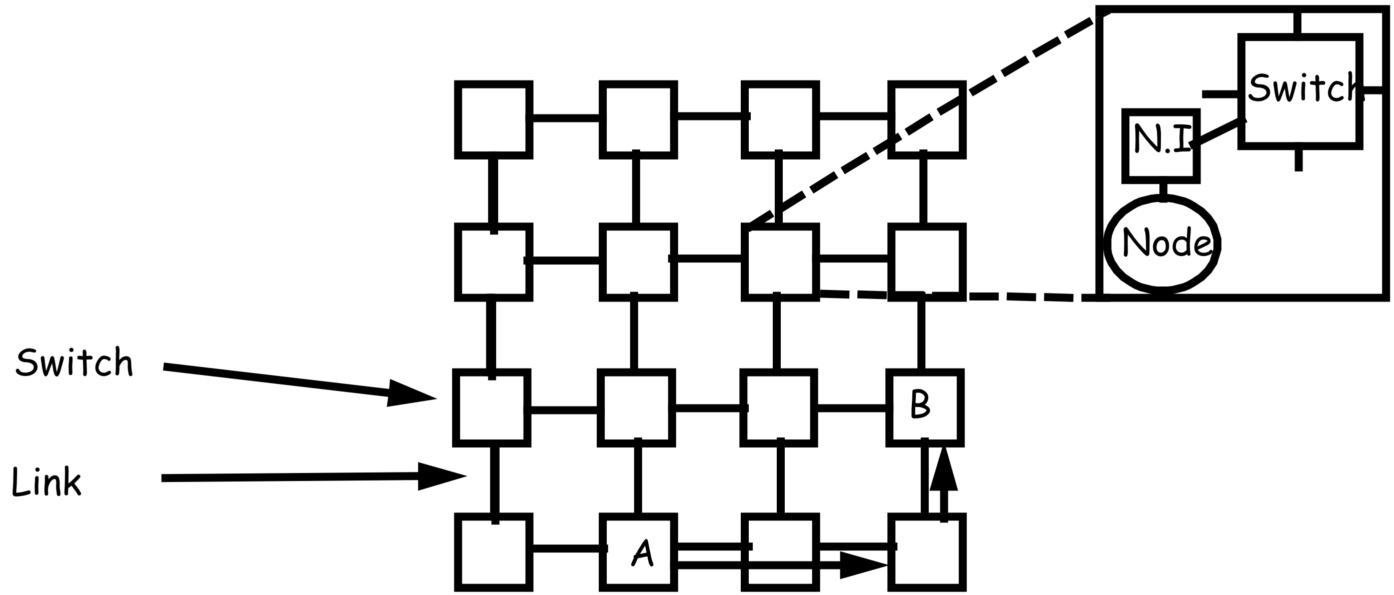
PARALLEL COMPUTER SYSTEMS



- INTERCONNECT BETWEEN PROCESSOR CHIPS (SYSTEM AREA NETWORK--SAN)
- INTERCONNECT BETWEEN CORES ON EACH CHIP (ON-CHIP-NETWORK--OCN or NETWORK ON A CHIP--NOC)

- OTHERS (NOT COVERED):
  - WAN (WIDE-AREA NETWORK)
  - LAN (LOCAL AREA NETWORK)

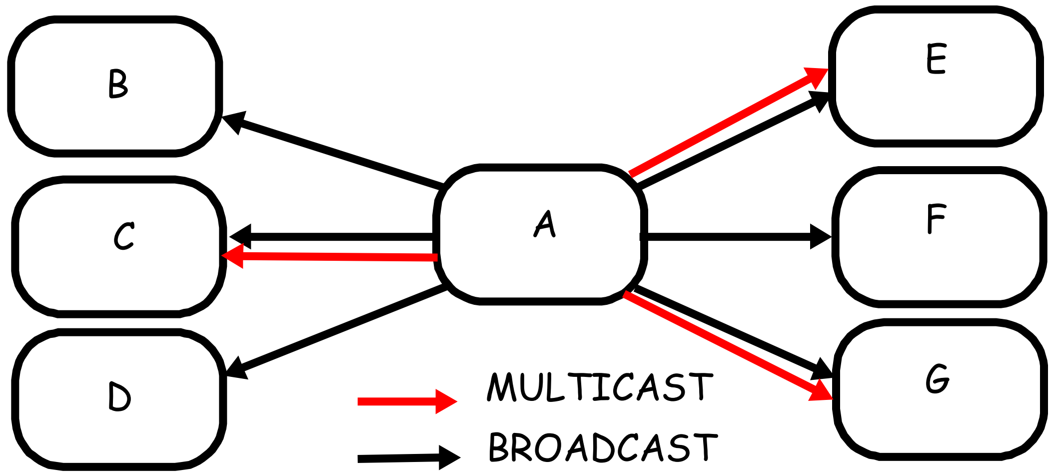
EXAMPLE: MESH



- **CONNECTS NODES: CACHE MODULES, MEMORY MODULES, CMPS…**
  - NODES ARE CONNECTED TO SWITCHES THROUGH A NETWORK INTERFACE (NI)
  - SWITCH: CONNECTS INPUT PORTS TO OUTPUT PORTS
  - LINK: WIRES TRANSFERING SIGNALS BETWEEN SWITCHES
- **LINKS**
  - WIDTH, CLOCK
  - TRANSFER CAN BE SYNCHRONOUS OR ASYNCHRONOUS
- **FROM A TO B: HOP FROM SWITCH TO SWITCH**

DECENTRALIZED (DIRECT)

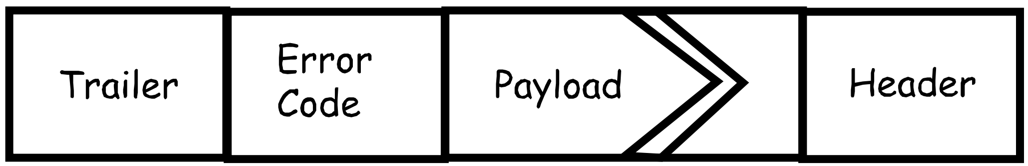
SIMPLE COMMUNICATION MODEL



- POINT-TO-POINT MESSAGE TRANSFER
- REQUEST/REPLY: REQUEST CARRIES ID OF SENDER
- MULTICAST: ONE TO MANY
- BROADCAST: ONE TO ALL

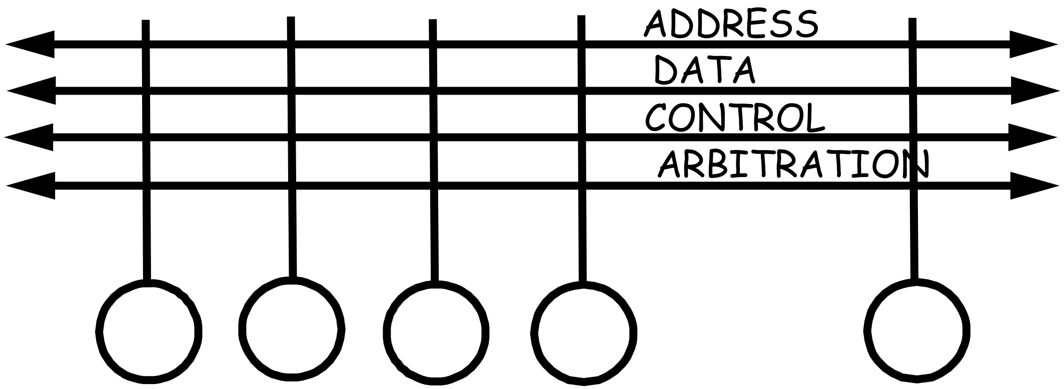
MESSAGES AND PACKETS

- MESSAGES CONTAIN THE INFORMATION TRANSFERED
- MESSAGES ARE BROKEN DOWN INTO PACKETS
- PACKETS ARE SENT ONE BY ONE



- PAYLOAD: MESSAGE--NOT RELEVANT TO INTERCONNECTION
- HEADER/TRAILER: CONTAINS INFORMATION TO ROUTE PACKET
- ERROR CODE: ECC TO DETECT AND CORRECT TRANSMISSION ERRORS
- HEADER+ECC+TRAILER = PACKET ENVELOPE

EXAMPLE: BUS



- BUS=WIRES
- BROADCAST/BROADCALL COMMUNICATION.
- NEEDS ARBITRATION.
- CENTRALIZED vs DISTRIBUTED ARBITRATION
- LINE MULTIPLEXING (ADDRESS/DATA FOR EXAMPLE)
- PIPELINING
- FOR EXAMPLE: ARBITRATION => ADDRESS => DATA
- SPLIT-TRANSACTION BUS vs CIRCUIT-SWITCHED BUS

CENTRALIZED (INDIRECT)
LOW COST
SHARED
LOW BANDWIDTH

SWITCHING STRATEGY

DEFINES HOW CONNECTIONS ARE ESTABLISHED IN THE NETWORK

CIRCUIT SWITCHING

- **ESTABLISH A CONNECTION FOR THE DURATION OF THE NETWORK SERVICE**
  - EXAMPLE.: REMOTE MEMORY READ ON A BUS:
    - CONNECT WITH REMOTE NODE
    - HOLD THE BUS WHILE THE REMOTE MEMORY IS ACCESSED
    - RELEASE THE BUS WHEN THE DATA HAS BEEN RETURNED
  - EXAMPLE: CIRCUIT SWITCHING IN MESH
    - ESTABLISH PATH IN NETWORK
    - TRANSMIT PACKET
    - RELEASE PATH
    - LOW LATENCY; HIGH BANDWIDTH
    - GOOD WHEN PACKETS ARE TRANSMITTED CONTINUOUSLY BETWEEN TWO NODES

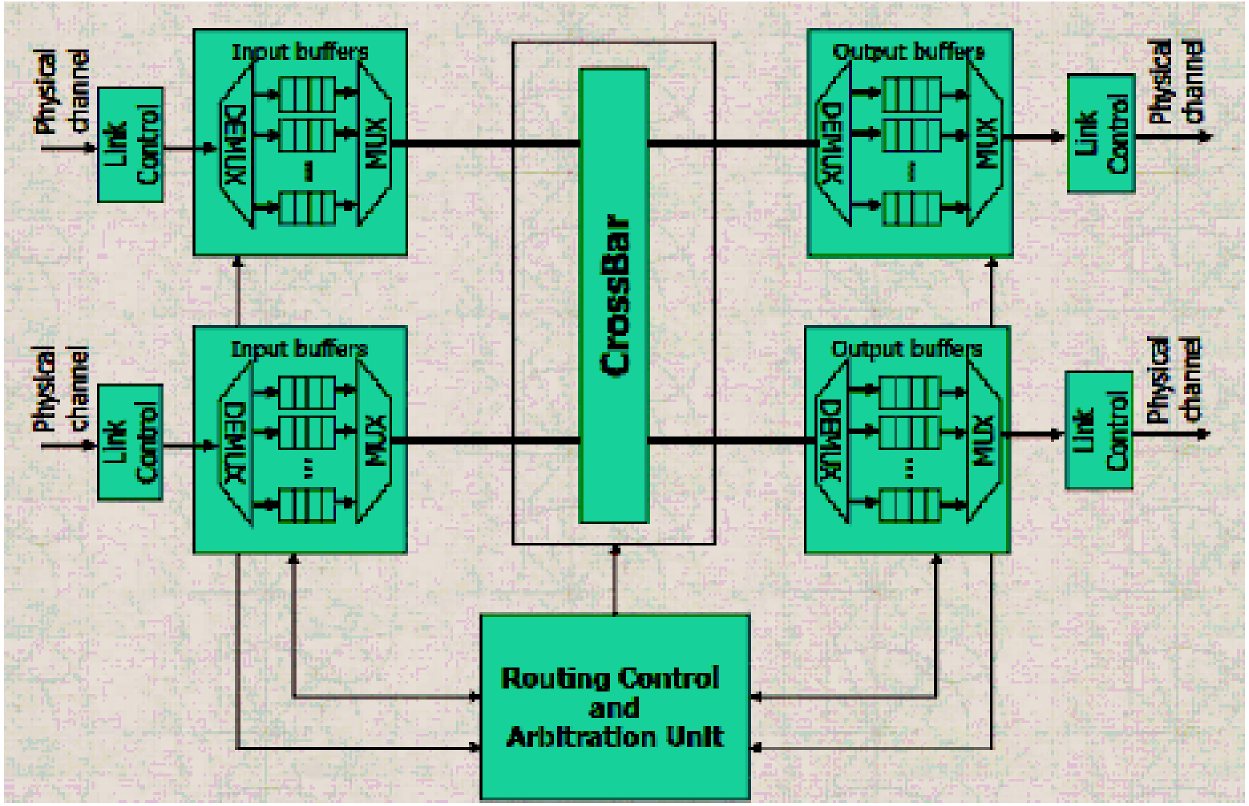
PACKET SWITCHING

- **MULTIPLEX SEVERAL SERVICES BY SENDING PACKETS WITH ADDRESSES**
  - EXAMPLE: REMOTE MEMORY ACCESS ON A BUS
    - SEND A REQUEST PACKET TO REMOTE NODE
    - RELEASE BUS WHILE MEMORY ACCESS TAKES PLACE
    - REMOTE NODE SENDS REPLY PACKET TO REQUESTER
    - IN BETWEEN SEND AND REPLY, OTHER TRANSFERS ARE SUPPORTED
  - EXAMPLE: REMOTE MEMORY ACCESS ON A MESH

SWITCHING STRATEGY

- **PACKET SWITCHING STRATEGIES**
    - TWO STRATEGIES: STORE-AND-FORWARD AND CUT-THROUGH PACKET SWITCHING
    - IN STORE-AND-FORWARD PACKET SWITCHING, PACKETS MOVE FROM NODE TO NODE AND ARE STORED IN BUFFERS IN EACH NODE
    - IN CUT-THROUGH PACKET SWITCHING, PACKETS CAN MOVE THROUGH NODES IN PIPELINE FASHION, SO THAT THE ENTIRE PACKET MOVES THROUGH SEVERAL NODES AT ONE TIME
- **TWO IMPLEMENTATIONS OF CUT-THROUGH PACKET SWITCHING**
    - IN PRACTICE WE MUST DEAL WITH CONFLICTS AND STALL PACKETS
    - VIRTUAL CUT-THROUGH SWITCHING:
        - EACH NODE HAS ENOUGH BUFFERING FOR THE ENTIRE PACKET
        - THE ENTIRE PACKET IS BUFFERED IN THE NODE WHEN THERE IS A TRANSMISSION CONFLICT
        - WHEN TRAFFIC IS CONGESTED AND CONFLICTS ARE HIGH, VIRTUAL CUT THROUGH BEHAVES LIKE STORE-AND-FORWARD
    - WORMHOLE SWITCHING:
        - EACH NODE HAS ENOUGH BUFFERING FOR A FLIT (FLOW CONTROL UNIT)
        - A FLIT IS MADE OF CONSECUTIVE PHITs (PHYSICAL TRANSFER UNIT), WHICH BASICALLY IS THE WIDTH OF A LINK (NUMBER OF BITS TRANSFERRED PER CLOCK)
        - THE FLIT IS THE BASIC UNIT OF TRANSFER SUBJECT TO FLOW CONTROL AND IT MUST AT LEAST CONTAIN THE ROUTING INFORMATION AT THE HEAD OF THE PACKET
        - IN VIRTUAL CUT-THROUGH THE FLIT IS THE WHOLE PACKET
        - IN WORMHOLE SWITCHING, IS A FRACTION OF THE PACKET, SO THE PACKET MUST BE STORED IN SEVERAL NODES (ONE FLIT PER NODE) ON ITS PATH AS IT MOVES THROUGH THE NETWORK.

SWITCH MICROARCHITECTURE



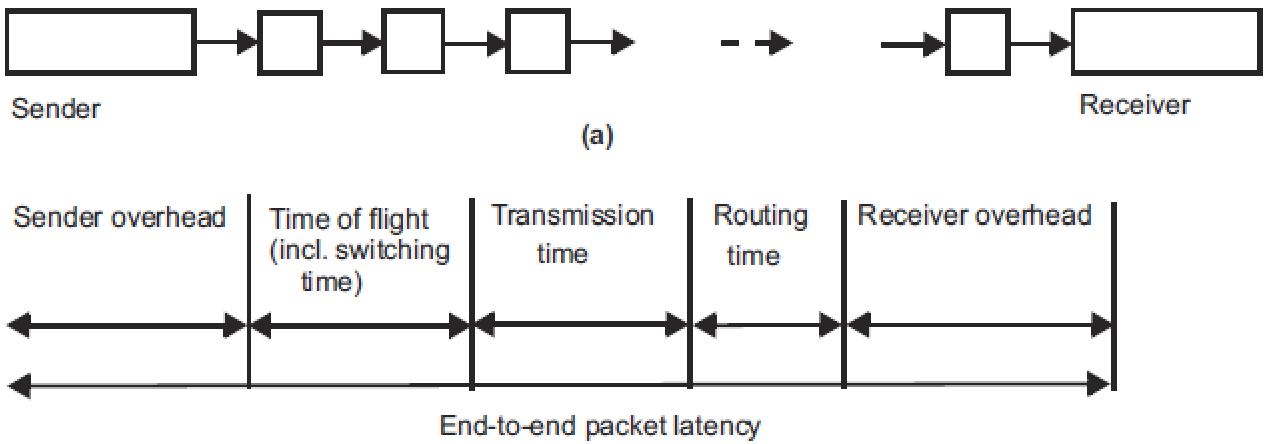
– From Duato and Pinkston in Hennessy and Patterson, 4th edition

PHYSICAL CHANNEL = LINK

VIRTUAL CHANNEL = BUFFERS + LINK

LINK IS TIME-MULTIPLEXED AMONG FLITS

LATENCY MODELS



End-to-end packet latency = Sender OV + Time of flight + Transmission time + Routing time + Receiver OV

- SENDER OVERHEAD: CREATING THE ENVELOPE AND MOVING PACKET TO NI
- TIME OF FLIGHT: TIME TO SEND A BIT FROM SOURCE TO DESTINATION WHEN THE ROUTE IS ESTABLISHED AND WITHOUT CONFLICTS. (INCLUDES SWITCHING TIME.)
- TRANSMISSION TIME: TIME TO TRANSFER A PACKET FROM SOURCE TO DESTINATION, ONCE THE FIRST BIT HAS ARRIVED AT DESTINATION
  - PHIT: NUMBER OF BITS TRANSFERED ON A LINK PER CYCLE
  - BASICALLY: PACKET SIZE/PHIT SIZE
  - FLIT: FLOW CONTROL UNIT

LATENCY MODELS

- **ROUTING TIME: TIME TO SET UP SWITCHES**
- **SWITCHING TIME: DEPENDS ON SWITCHING STRATEGY (STORE-AND-FORWARD vs CUT-THROUGH vs CIRCUIT-SWITCHED). AFFECTS TIME OF FLIGHT AND INCLUDED IN THAT.**
- **RECEIVER OVERHEAD: TIME TO STRIP OUT ENVELOPE AND MOVE PACKET IN**

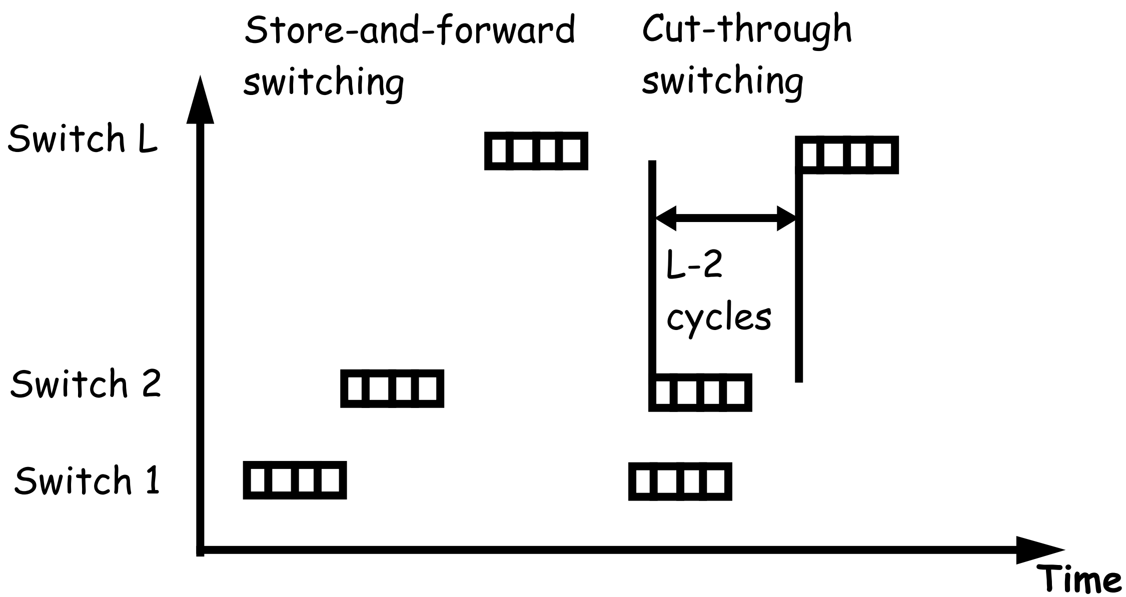
- **MEASURES OF LATENCY:**
  - ROUTING DISTANCE: NB OF LINKS TRAVERSED BY A PACKET
  - AVERAGE ROUTING DISTANCE: AVERAGE OVER ALL PAIRS OF NODES
  - NETWORK DIAMETER: LONGEST ROUTING DISTANCE OVER ALL PAIRS OF NODES

- **PACKETS OF A MESSAGE CAN BE PIPELINED**
  - TRANSFER PIPELINE HAS 3 STAGES
    - SENDER OVERHEAD-->TRANSMISSION -->RECEIVER OVERHEAD
  - TOTAL MESSAGE TIME = TIME FOR THE FIRST PACKET + (N-1)/PIPELINE THROUGHPUT

End-to-end message latency = Sender OV + Time of Flight + Transmission time + Routing time + (N-1) x MAX (Sender OV, Transmission time, Receiver OV)

SWITCHING STRATEGIES

- **CIRCUIT SWITCHING:**
  - ROUTE IS SET UP FIRST
  - Routing time = L x R + Time of flight
    - R to set each switch, L number of switches, and ToF to inform the node back
- **PACKET SWITCHING**
  - ROUTE IS SET UP AS THE PACKET MOVES FROM SWITCH TO SWITCH
  - STORE-AND-FORWARD, CUT-THROUGH



SWITCHING STRATEGIES

Packet latency = Sender ov + ToF (incl.Switching time) + Transmission time + Routing time + Receiver ov

R: routing time per switch; N: Nb of phits; L: Nb of switches; ToF: Time of flight

- **CIRCUIT SWITCHING**
  - PACKET LATENCY = SENDER OV + 2xToF + N + LxR + RECEIVER OV
  - ToF = L BECAUSE THERE ARE L SWITCHES AND NB OF PHITS TO SWITCH IS ONE
- **STORE-AND-FORWARD**
  - PACKET LATENCY = SENDER OV + ToF + N + LxR + RECEIVER OV
  - ToF = LxN BECAUSE SWITCHING INVOLVES A WHOLE PACKET (N PHITS)
- **CUT-THROUGH SWITCHING**
  - PACKET LATENCY = SENDER OV + ToF + N + LxR + RECEIVER OV
  - ToF = L AS IN CIRCUIT SWITCHING
- **VIRTUAL CUT-THROUGH SWITCHING**
  - SIMILAR TO CIRCUIT SWITCHING BUT BETTER BW
  - NOTE THAT WHEN TRAFFIC IS CONGESTED, CUT-THROUGH = STORE-AND-FORWARD
- **WORMHOLE SWITCHING**
  - HANDLES CONFLICTS DIFFERENTLY
  - SWITCH PORT HAS AT LEAST ENOUGH BUFFERING FOR A FLIT
  - BLOCKED PACKETS SIMPLY STAY IN THE FLIT BUFFERS PROVIDED IN THEIR PATH
  - PACKET FLITS HOLD CIRCUITS IN MULTIPLE SWITCHES

BANDWIDTH MODELS

- **BOTTLENECKS INCREASE LATENCY**
  - TRANSFERS ARE PIPELINED
  - EFFECTIVE BANDWIDTH = PACKET_SIZE/MAX(SENDER OV, RECEIVER OV, TRANSMISSION TIME)

- **NETWORK CONTENTION AFFECTS LATENCY AND EFFECTIVE BANDWIDTH (NOT COUNTED IN ABOVE FORMULA)**

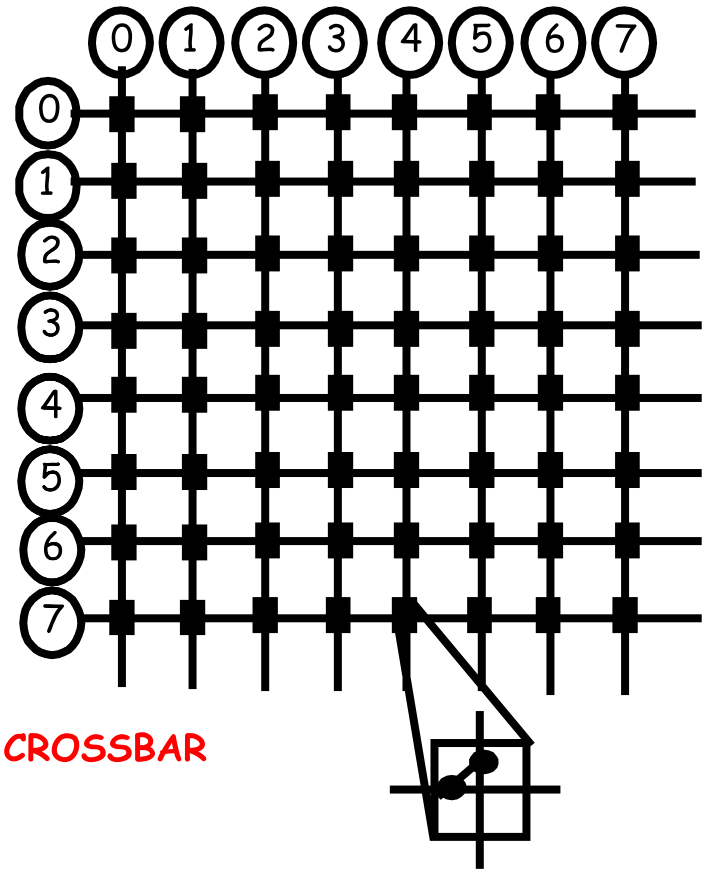
- **BISECTION WIDTH**
  - NETWORK IS SEEN AS A GRAPH
  - VERTICES ARE SWITCHES AND EDGES ARE LINKS
  - BISECTION IS A CUT THROUGH A MINIMUM SET OF EDGES SUCH THAT THE CUT DIVIDES THE NETWORK GRAPH INTO TWO ISOMORPHIC --I.E., SAME-- SUBGRAPHS
    - EXAMPLE: MESH
  - MEASURES BANDWIDTH WHEN ALL NODES IN ONE SUBGRAPH COMMUNICATE ONLY WITH NODES IN THE OTHER SUBGRAPH

- **AGGREGATE BANDWIDTH**
  - BANDWIDTH ACROSS ALL LINKS DIVIDED BY THE NUMBER OF NODES

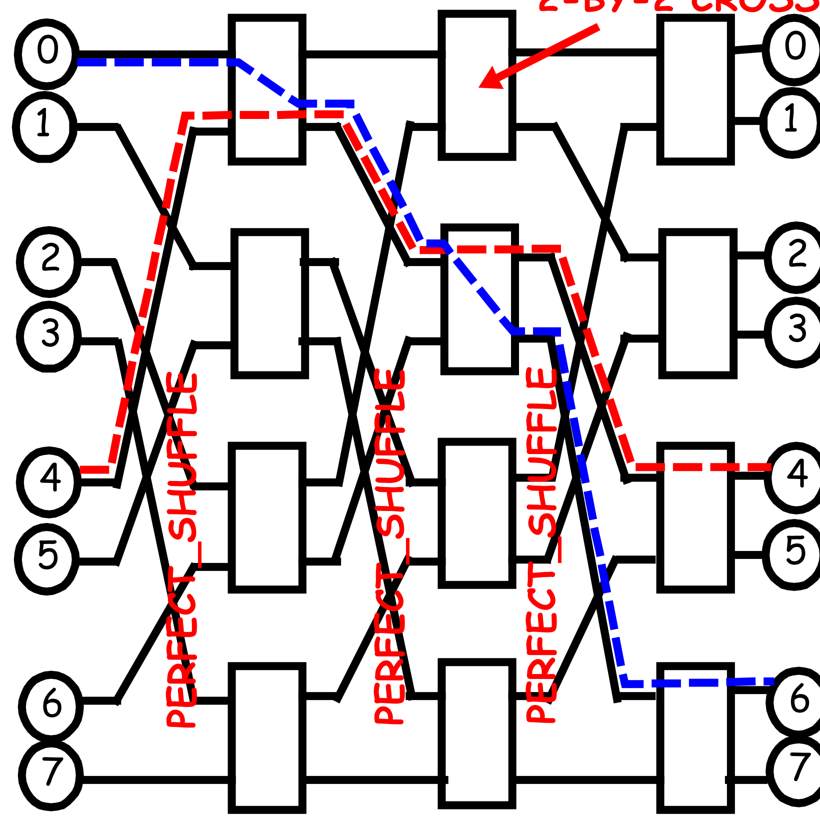
TOPOLOGIES

INDIRECT NETWORKS: IN IS CENTRALIZED

- BUS
- CROSSBAR SWITCH
- MULTISTAGE INTERCONNECTION NETWORK (MIN)



CROSSBAR

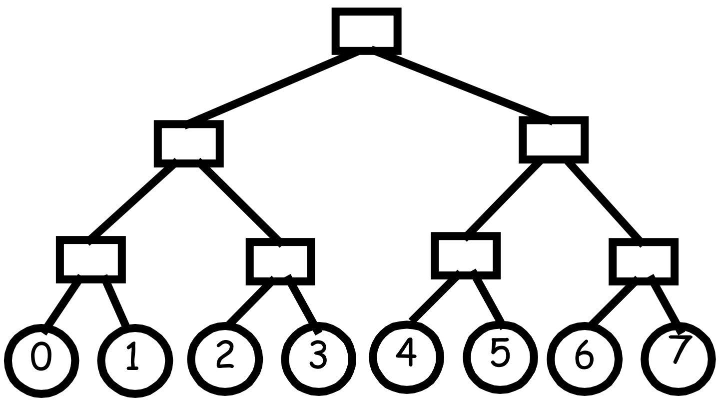


OMEGA NETWORK

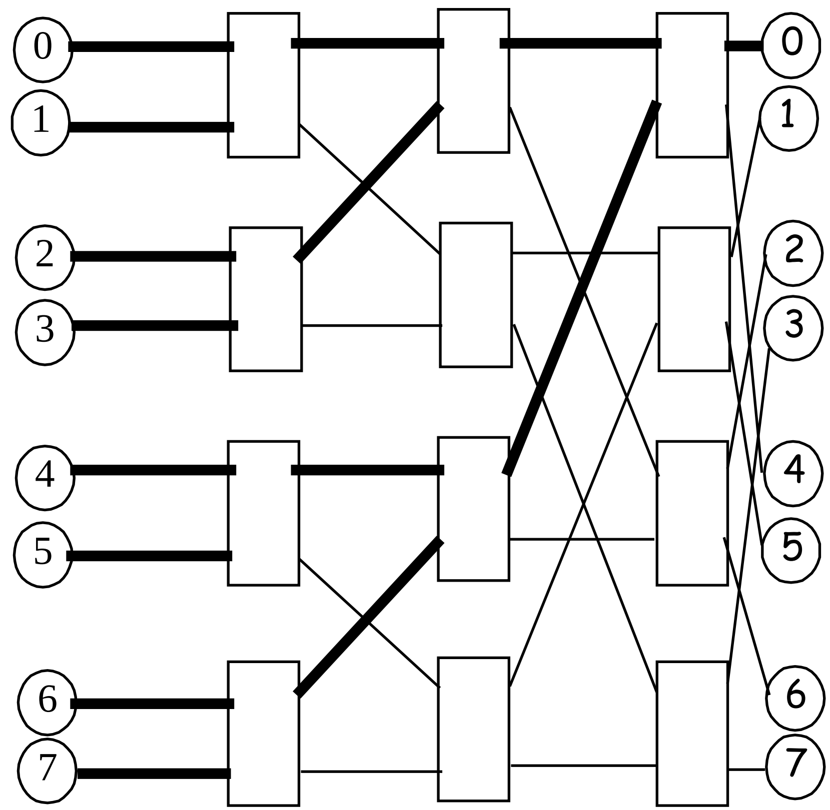
TOPOLOGIES

INDIRECT NETWORKS: IN IS CENTRALIZED

- TREE
- BUTTERFLY



BINARY TREE



BUTTERFLY: EMBEDDED TREES

BEST TO CONNECT DIFFERENT TYPES OF NODES

TOPOLOGIES

DIRECT NETWORKS: NODES ARE DIRECTLY CONNECTED TO ONE ANOTHER
DECENTRALIZED

- LINEAR ARRAY AND RING

DIAMETER: N-1

ARRAY

DIAMETER:N/2

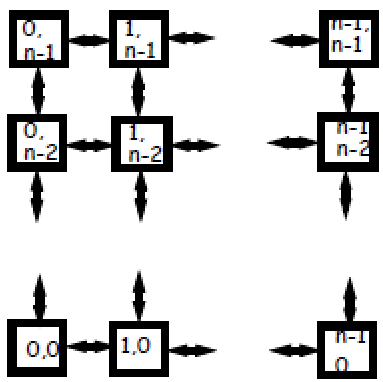
RING
UNDIRECTIONAL
BIDIRECTIONAL

- MESH AND TORI

MESH

DIAMETER:2(n-1)

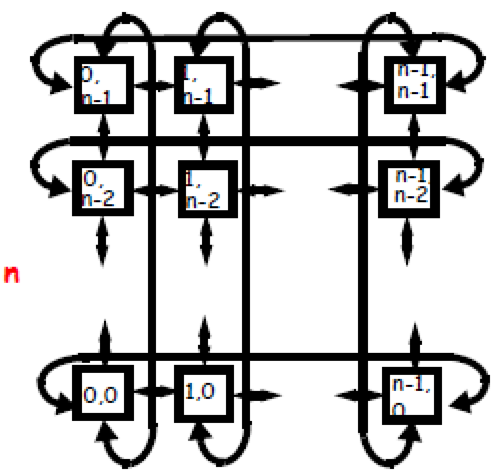
$N=n^2$



TORI

DIAMETER: n

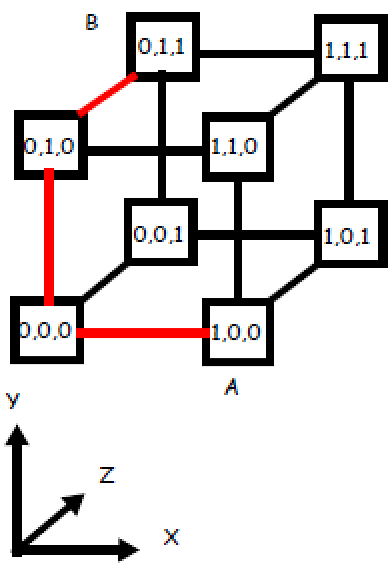
$N=n^2$



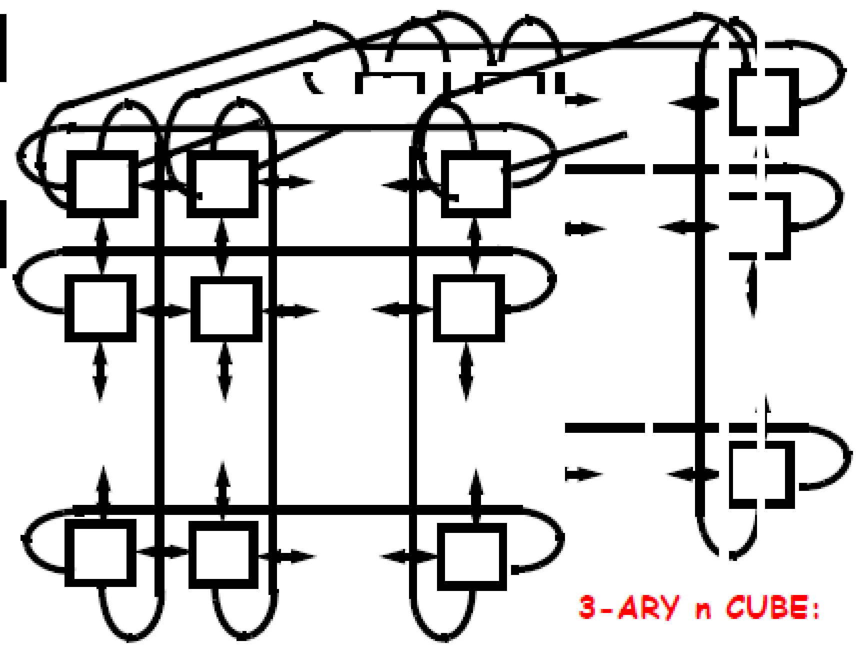
TOPOLOGIES

DIRECT NETWORKS: NODES ARE DIRECTLY CONNECTED TO ONE ANOTHER

HYPERCUBE AND k-ARY n-CUBE



3-D HYPERCUBE
3-ARY 2 CUBE



3-ARY n CUBE:

k (-ary): DIMENSION
n (-cube): NB OF NODES IN EACH DIMENSION

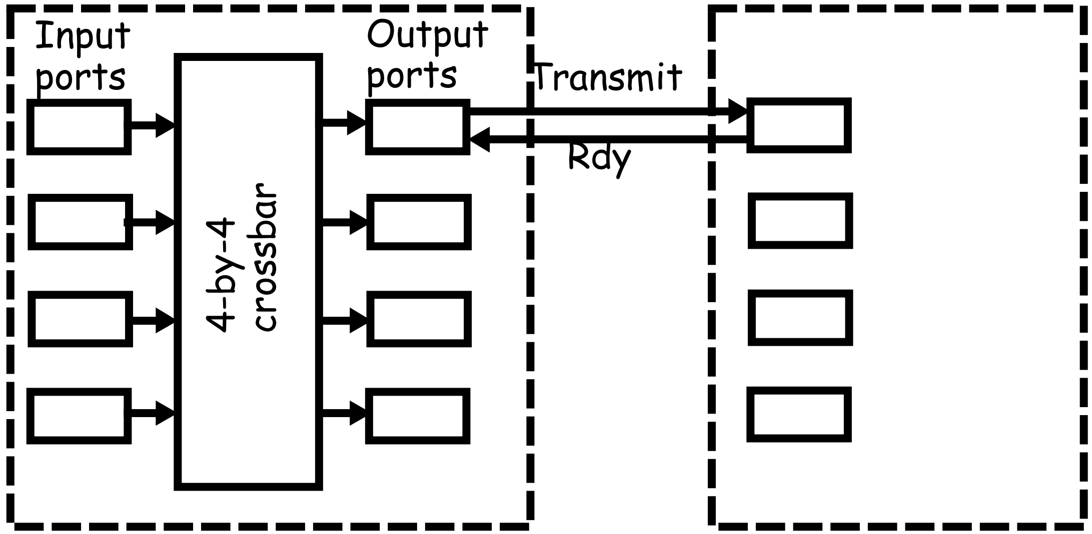
COMPARISON BETWEEN TOPOLOGIES

Interconnection network	Switch degree	Network diameter	Bisection width	Network size
Crossbar switch	N	1	N	N
Butterfly (built from k-by-k switches)	k	$\log_k N$	N/2	N
k-ary tree	k+1	$2\log_k N$	1	N
Linear array	2	N-1	1	N
Ring	2	N/2	2	N
n-by-n mesh	4	2(n-1)	n	$N=n^2$
n-by-n torus	4	n	2n	$N=n^2$
k-dimensional hypercube	k	k	2^{k-1}	$N=2^k$
k-ary n-cube	2k	nk/2	$2k^{n-1}$	$N=n^k$

- SWITCH DEGREE: NUMBER OF PORTS FOR EACH SWITCH (SWITCH COMPLEXITY)
- NETWORK DIAMETER: WORST-CASE ROUTING DISTANCE BETWEEN ANY TWO NODES
- BISECTION WIDTH: NB OF LINKS IN BISECTION (WORST-CASE BW)
- NETWORK SIZE: NB OF NODES

FLOW CONTROL

- **REFERS TO MECHANISMS TO HANDLE CONFLICTS IN SWITCH-BASED NETWORKS**

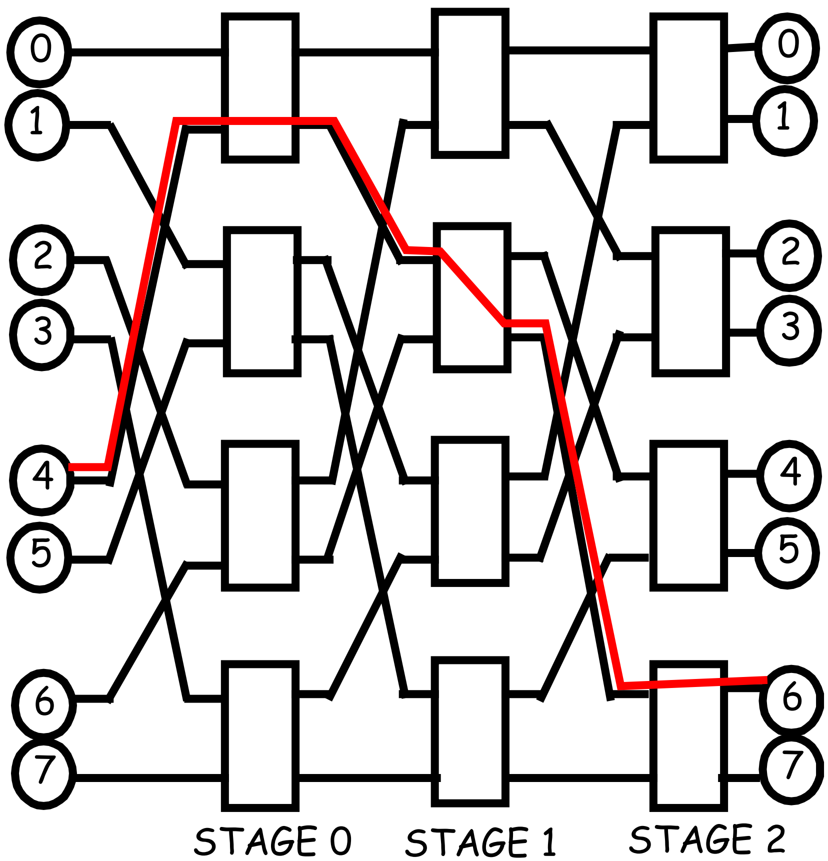


  - BUFFERS AT INPUT AND OUTPUT PORTS
    - IN VIRTUAL CUT-THROUGH: BUFFER FOR ENTIRE PACKET
    - IN WORMHOLE: BUFFER FOR INTEGRAL NUMBER OF FLITS
- **LINK-LEVEL FLOW CONTROL**
  - HANDSHAKE SIGNAL
    - Rdy INDICATES WHETHER FLITS CAN BE TRANSMITTED TO THE DESTINATION
    - BUFFERING FOR CUT-THROUGH (whole packet) vs WORMHOLE (a few flits)
- **HOT SPOT CONTENTION AND TREE SATURATION**

ROUTING ALGORITHMS

- USE SOURCE AND/OR DESTINATION ADDRESSES

OMEGA NETWORK



USE THE DESTINATION ADDRESS IN THIS CASE, 3 BITS <d2,d1,d0>

USE ith BIT OF THE DESTINATION (di) TO SELECT UPPER OR LOWER OUTPUT PORT FOR STAGE n-1-i

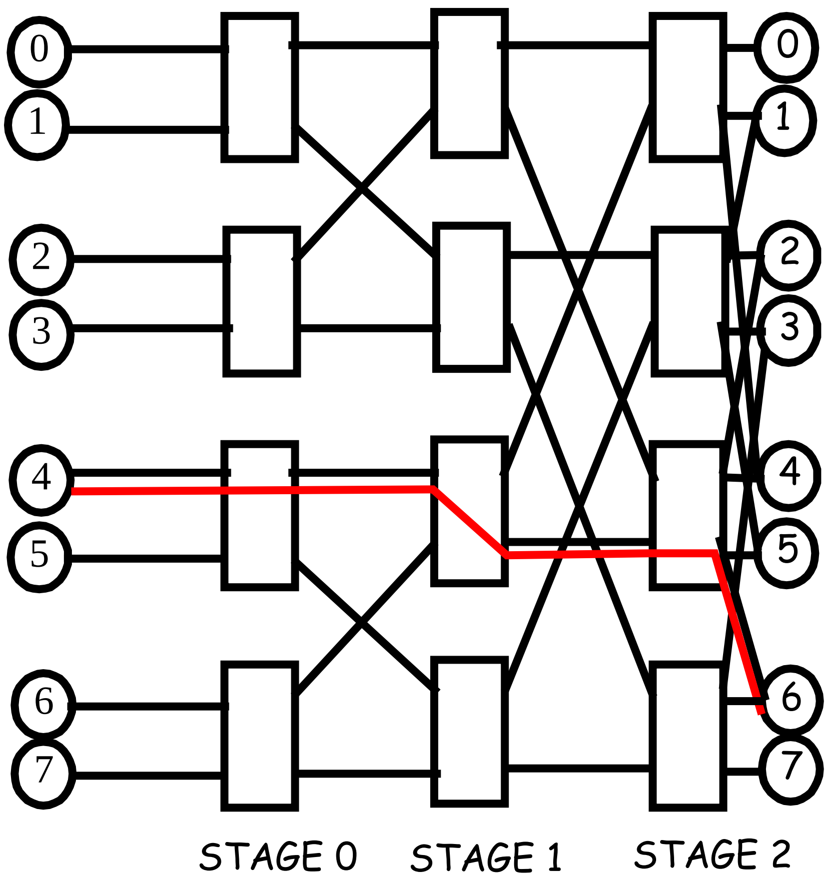
- 0 => UP
- 1 => DOWN

EXAMPLE: ROUTE FROM 4 TO 6

- DESTINATION ADDRESS IS 110
  - DOWN IN STAGE 0
  - DOWN IN STAGE 1
  - UP IN STAGE 2

ROUTING ALGORITHMS

BUTTERFLY NETWORK



USE RELATIVE ADDRESS

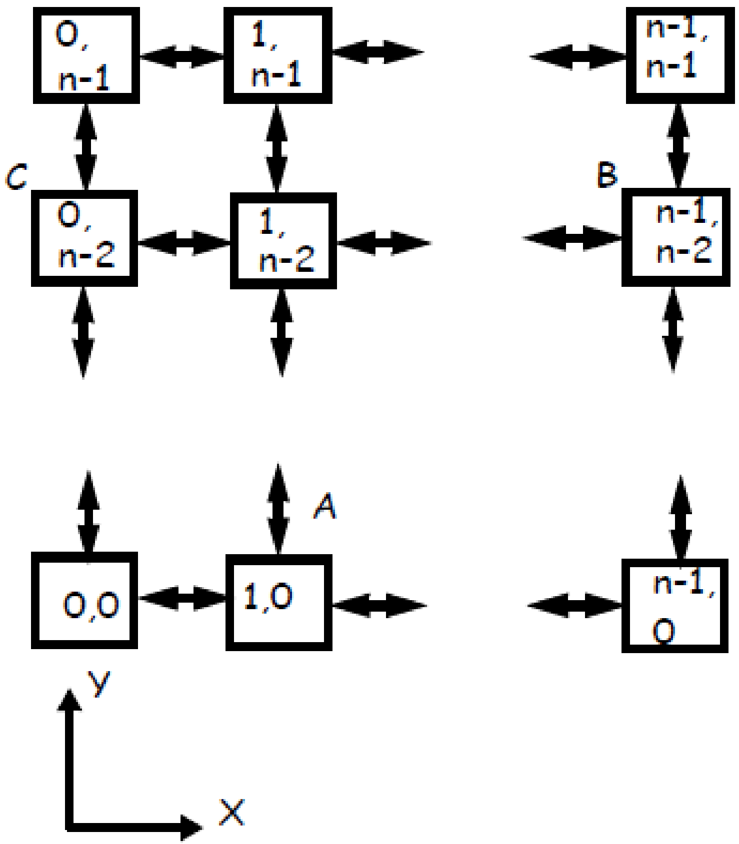
- BITWISE EXCLUSIVE OR OF SOURCE AND DESTINATION ADDRESSES TO FORM THE ROUTING ADDRESS
- IF BIT i IS ZERO, ROUTE STRAIGHT
- IF BIT i IS ONE, ROUTE ACROSS

EXAMPLE: ROUTE FROM 4 TO 6

- SOURCE: 100
- DESTINATION: 110
- EX-OR: 010

ROUTING ALGORITHMS

DIMENSION-ORDER ROUTING (DETERMINISTIC)



NUMBER NODES SO THAT LOWER LEFT CORNER IS (0,0) AND UPPER RIGHT CORNER IS (n-1,n-1)

USE RELATIVE ADDRESS:

- $(X,Y)=(X_B-X_A,Y_B-Y_A)$
- ROUTE FIRST HORIZONTALLY
  - DECREMENT X
- WHEN X=0, ROUTE VERTICALLY
  - DECREMENT Y
  - UNTIL Y=0

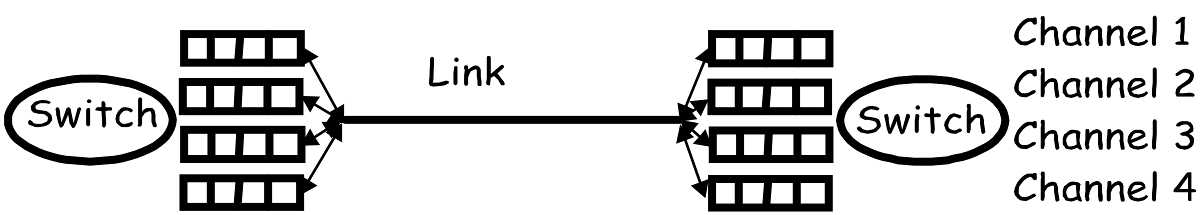
EXAMPLE: MOVE PACKET FROM A TO B

- RELATIVE ADDRESS IS (X,Y)=(n-2,n-2)
- FIRST MOVE PACKET HORIZONTALLY
  - DECREMENT X BY 1 (RIGHT MOVE)
- WHEN X=0, MOVE PACKET VERTICALLY
  - DECREMENT Y BY 1 (UP MOVE)

TRY B TO C

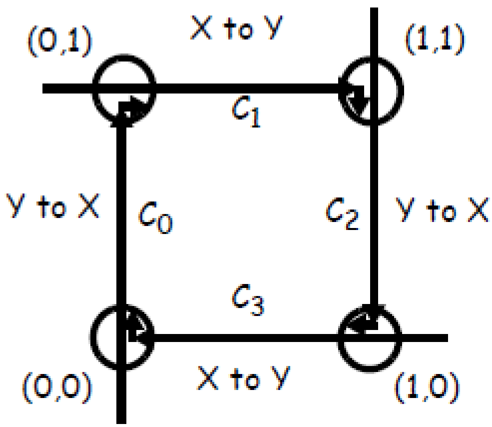
DEADLOCK AVOIDANCE

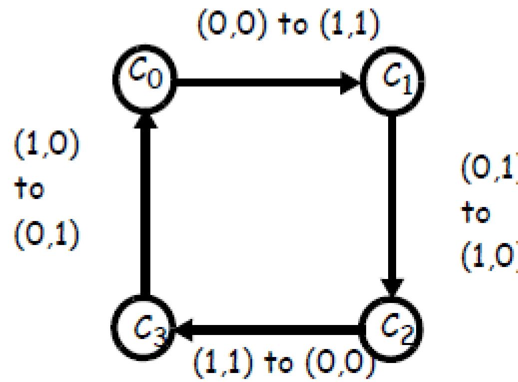
- **IN GENERAL THERE ARE FOUR NECESSARY CONDITIONS FOR DEADLOCK, GIVEN A SET OF AGENTS ACCESSING A SET OF SHARED RESOURCES:**
  - MUTUAL EXCLUSION
    - ONLY ONE AGENT CAN ACCESS THE RESOURCE AT A TIME
  - NO PREEMPTION
    - ONCE AN AGENT HAS ACQUIRED A SHARED RESOURCE, NO MECHANISM CAN FORCE IT TO RELINQUISH THE RESOURCE
  - HOLD AND WAIT
    - AGENT HOLDS ON ITS ACQUIRED RESOURCES WHILE WAITING FOR OTHERS
  - CIRCULAR WAIT
    - A SET OF AGENTS WAIT ON EACH OTHER TO ACQUIRE EACH OTHERS' RESOURCES SO THAT NO ONE CAN MAKE ANY PROGRESS
- **IN GENERAL, THE SHARED RESOURCES CAN BE SOFTWARE OR HARDWARE**
  - CRITICAL SECTIONS, DISK, PRINTER, ETC...
- **IN THE CASE OF NETWORKS**
  - AGENTS = PACKETS; RESOURCES = PHYSICAL OR LOGICAL CHANNELS



DEADLOCK AVOIDANCE

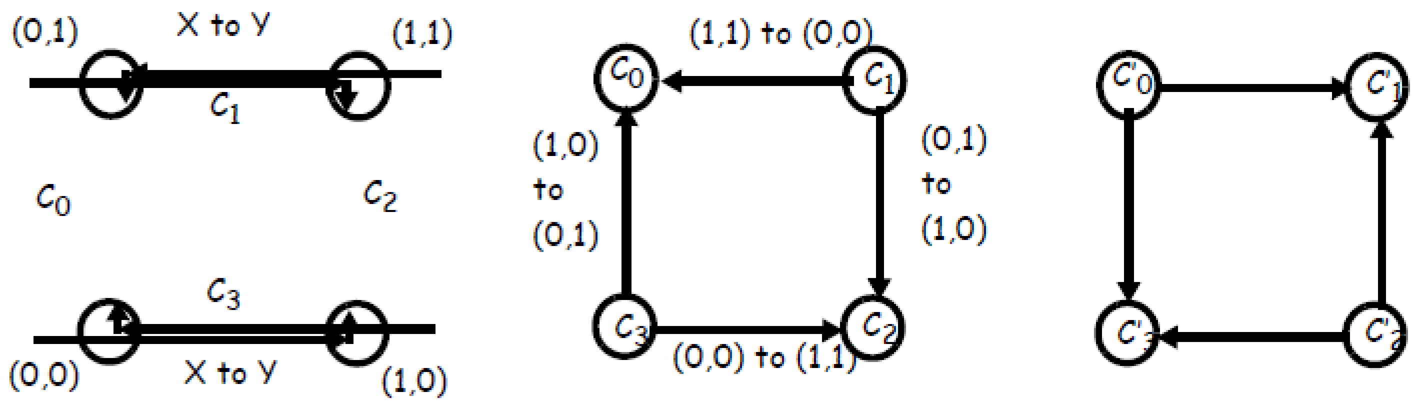
- ASSUME MESH OR TORI
- ASSUME THAT PACKETS ARE FREE TO FOLLOW ANY ROUTE





- IN THIS EXAMPLE EACH NODE IS TRYING TO SEND A PACKET TO THE DIAGONALLY OPPOSITE NODE AT THE SAME TIME
  - E.G., (0,0) TO (1,1)
- TO AVOID LINK CONFLICTS, (1,0) USES C_3 THEN C_0 AND (0,0) USES C_0 THEN C_1, ETC...
- THE RESOURCE ACQUISITION GRAPH (or *CHANNEL-DEPENDENCY GRAPH*) ON THE RIGHT SHOWS CIRCULAR WAIT
  - MEANS: DEADLOCK IS POSSIBLE

DEADLOCK AVOIDANCE



- **ENFORCE DIMENSION-ORDER ROUTING (XY ROUTING)**
  - PACKET MOVES FIRST HORIZONTALLY
  - THEN VERTICALLY
  - NO CYCLE!!!
- **PROBLEM: CONTENTION FOR CHANNELS**
  - IF (0,0) WANTS TO SEND A PACKET TO (1,1), IT MUST FIRST USE C_3
  - IF C_3 IS OCCUPIED, COULD TAKE ALTERNATE ROUTE C_0 => C_1
- **TO AVOID DEADLOCKS, USE VIRTUAL CHANNELS**
  - ALTERNATE SET OF CHANNELS IN WHICH YX ROUTING IS ENFORCED E.G., C'_1
  - IF C_3 IS OCCUPIED, THE PACKET CAN SAFELY ROUTE THROUGH C'_0 AND C'_1.