Parallel Processing
Interconnect Architectures

“Topologies”

Joseph Wunderlich PhD

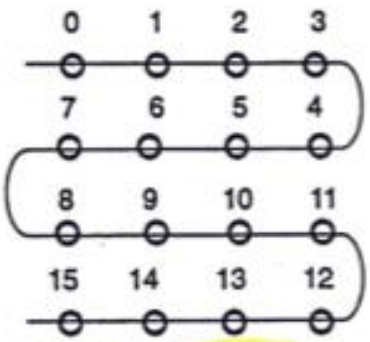
For Processors and Cores

- As well as Computer Networks

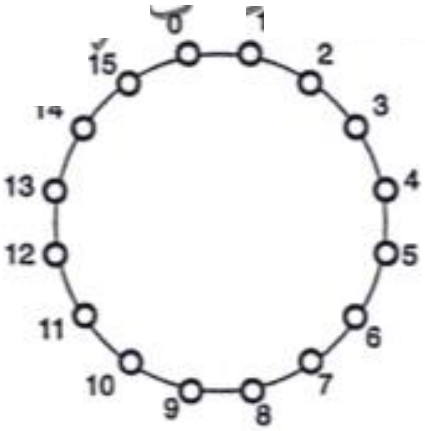
Parallel Processing
Interconnect Architectures

- Static
  - Architectures
  - Network Characteristics
  - Scalability
- Dynamic
  - Architectures
  - Scalability

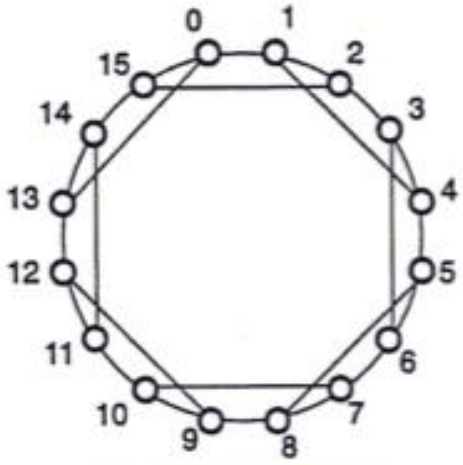
STATIC INTERCONNECT ARCHITECTURES



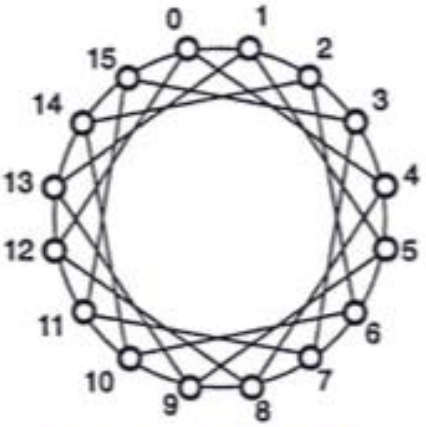
(a) Linear array



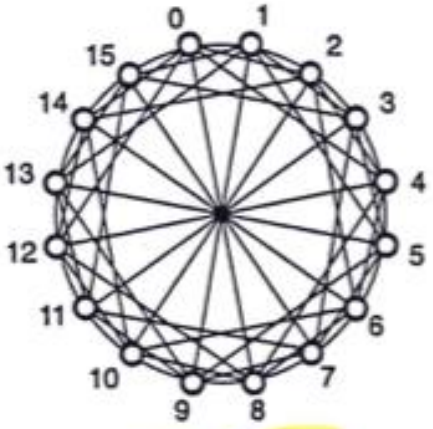
(b) Ring



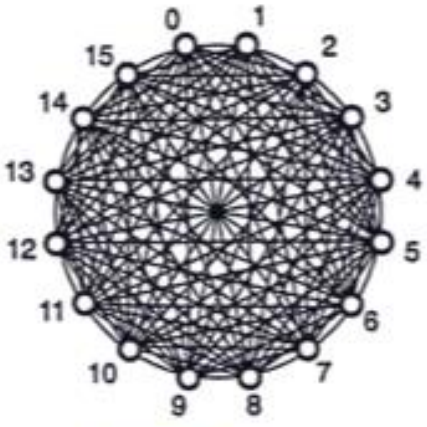
(c) Chordal ring of degree 3



(d) Chordal ring of degree 4 (same as Illiac mesh)



(e) Barrel shifter



(f) Completely connected

Figure 2.16 Linear array, ring, chordal rings of degrees 3 and 4, barrel shifter, and completely connected network.

K. Hwang, " Advanced Computer Architecture: Parallelism, Scalability, Programmability," Boston, MA: McGraw-Hill, 1992.

STATIC INTERCONNECT ARCHITECTURES

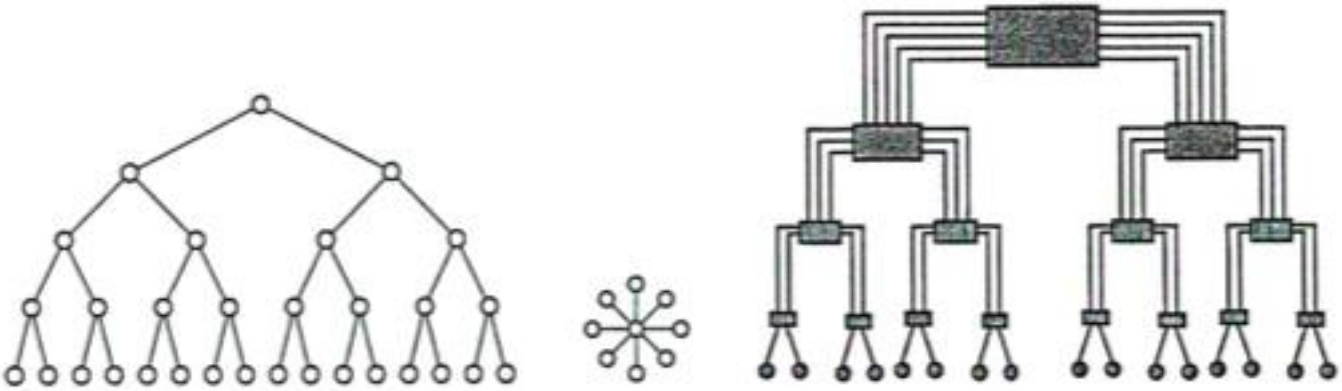
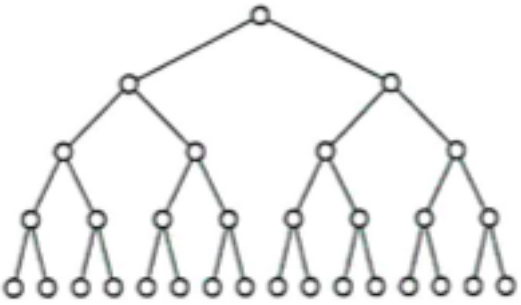


Figure 2.17 Tree, star, and fat tree.

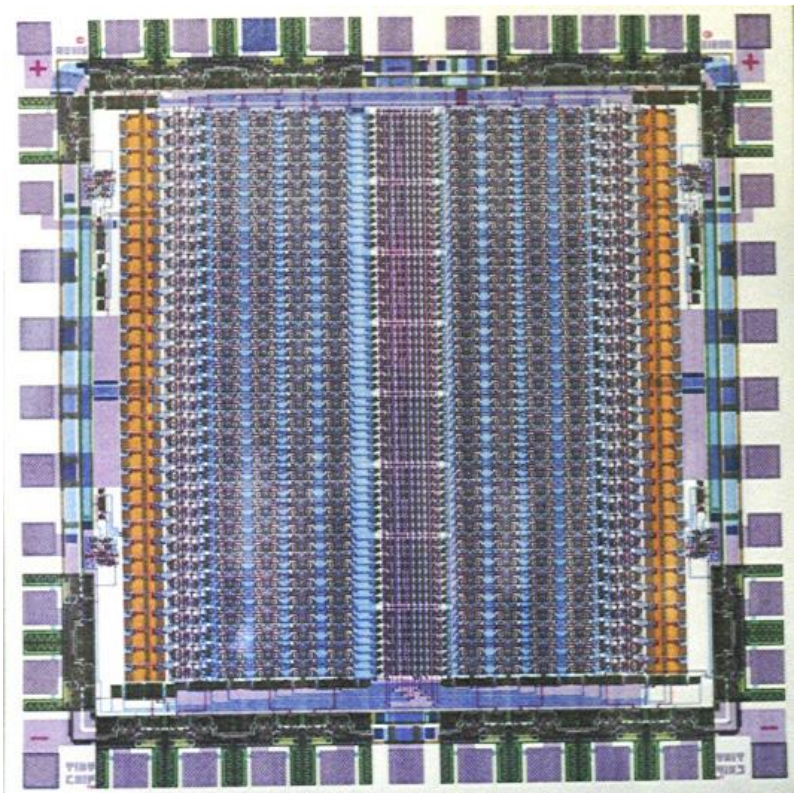
K. Hwang, " Advanced Computer Architecture: Parallelism, Scalability, Programmability," Boston, MA: McGraw-Hill, 1992.

STATIC INTERCONNECT ARCHITECTURES

J. Wunderlich Neural Network Processor 1992



(a) Binary tree



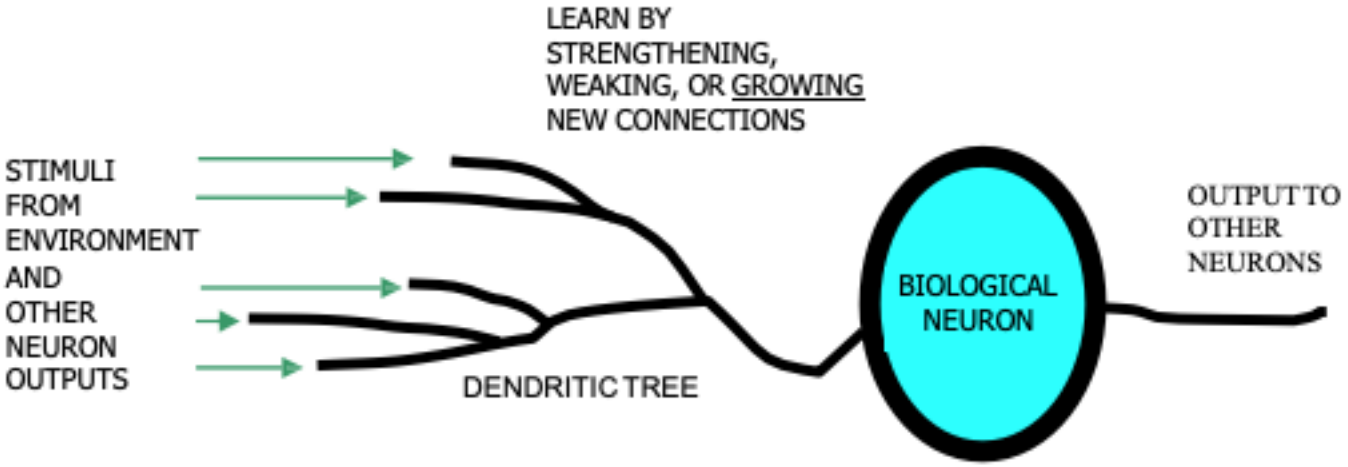
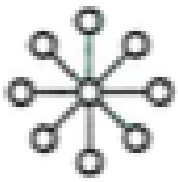


Figure 1. Biological neuron for bottom-up neurocomputer design.

Hardware for Machine Learning
Two Single-Chip Neurocomputer Designs; One Bottom-Up, One Top-Down
DRAFT BOOK CHAPTER, *Joseph T Wunderlich PhD*

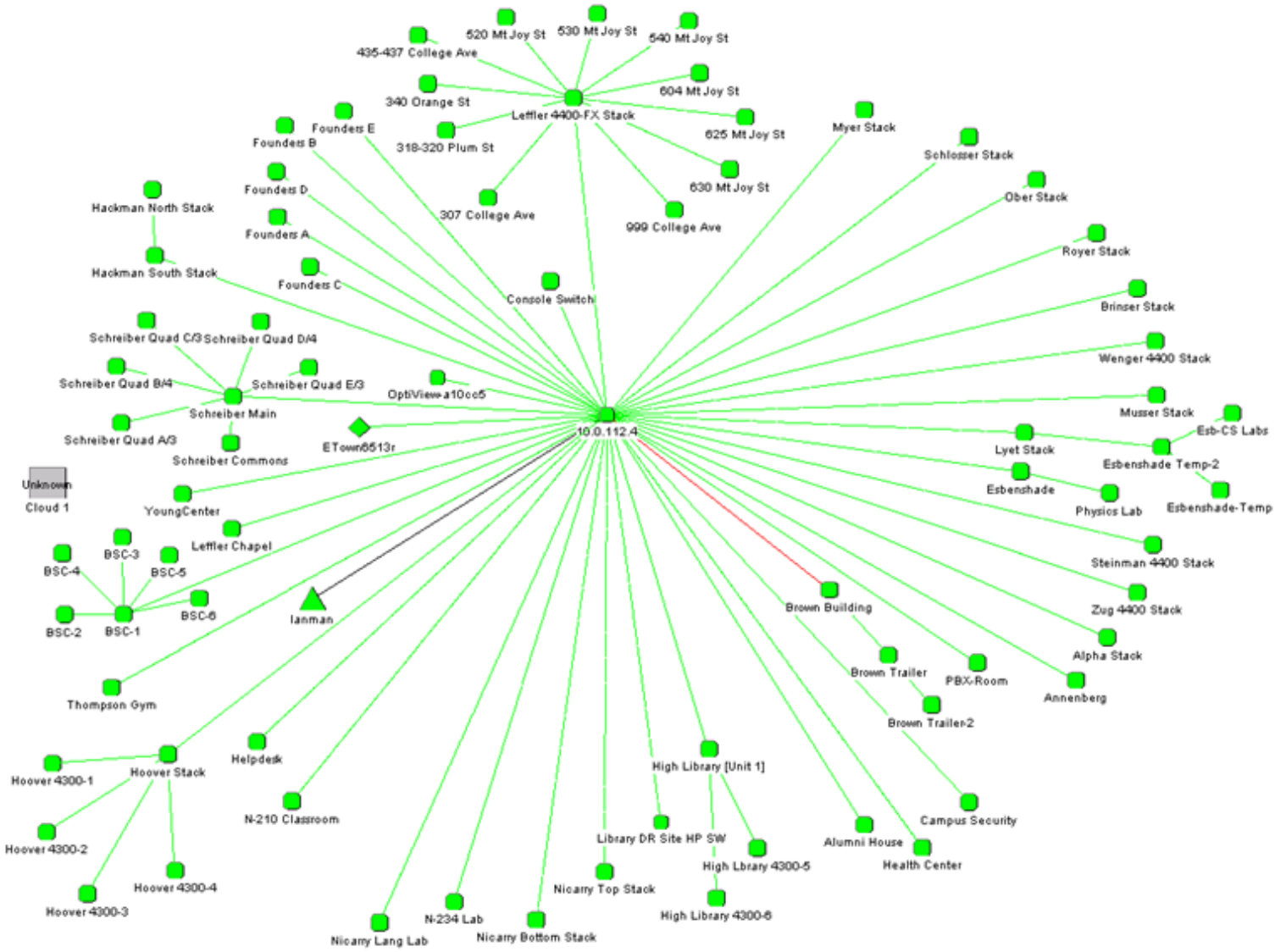
PPTX-w/audio PDF MP4 YouTube

STATIC INTERCONNECT ARCHITECTURES



(b) Star

Elizabethtown College Network
from a couple years ago



STATIC INTERCONNECT ARCHITECTURES

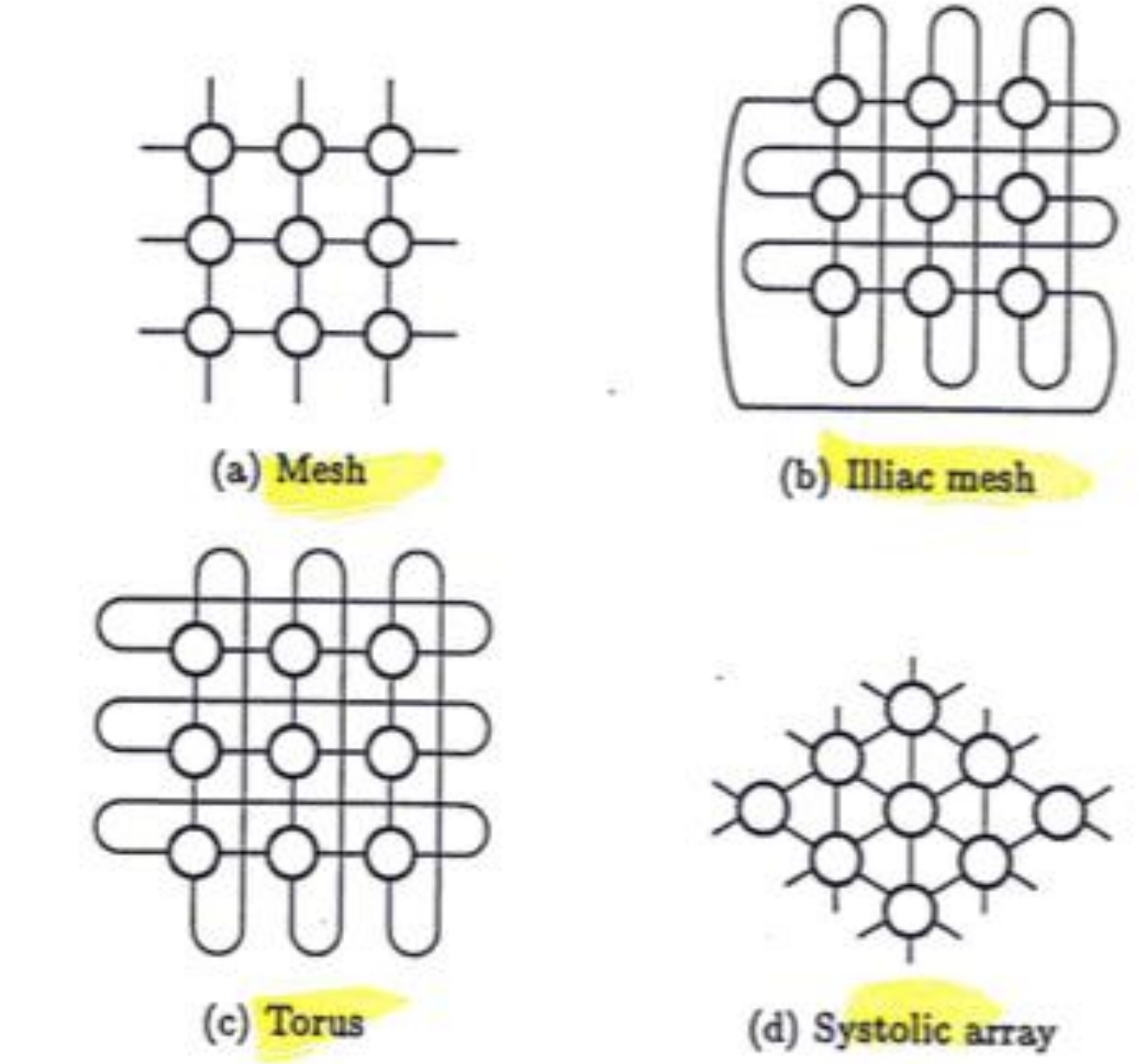
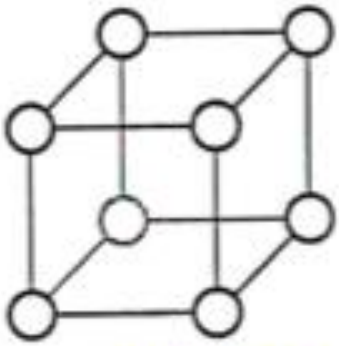


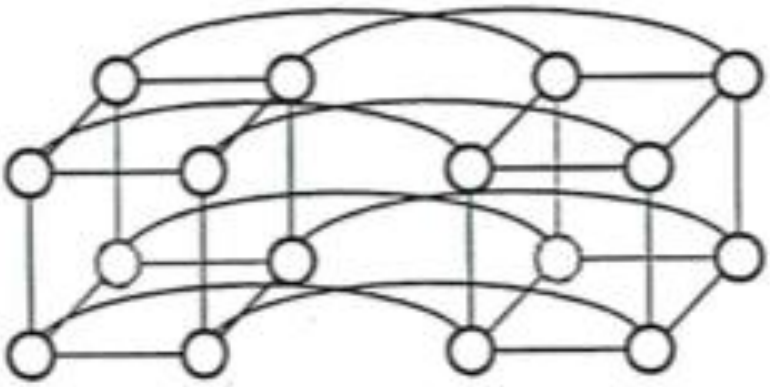
Figure 2.18 Mesh, Illiac mesh, torus, and systolic array.

K. Hwang, " Advanced Computer Architecture: Parallelism, Scalability, Programmability," Boston, MA: McGraw-Hill, 1992.

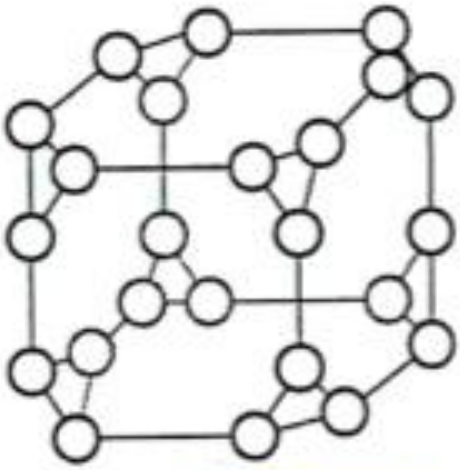
STATIC INTERCONNECT ARCHITECTURES



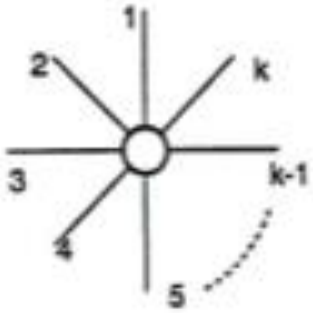
(a) 3-cube

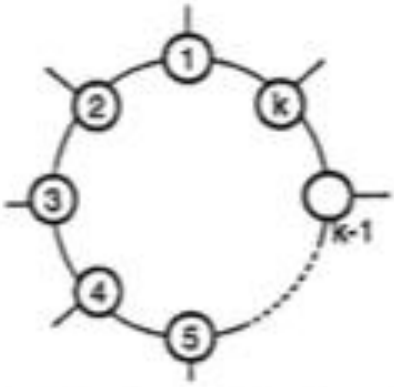


(b) A 4-cube formed by interconnecting two 3-cubes



(c) 3-cube-connected cycles





(d) Replacing each node of a k-cube by a ring (cycle) of k nodes to form the k-cube-connected cycles

Figure 2.19 Hypercubes and cube-connected cycles.

K. Hwang, " Advanced Computer Architecture: Parallelism, Scalability, Programmability," Boston, MA: McGraw-Hill, 1992.

STATIC INTERCONNECT ARCHITECTURES

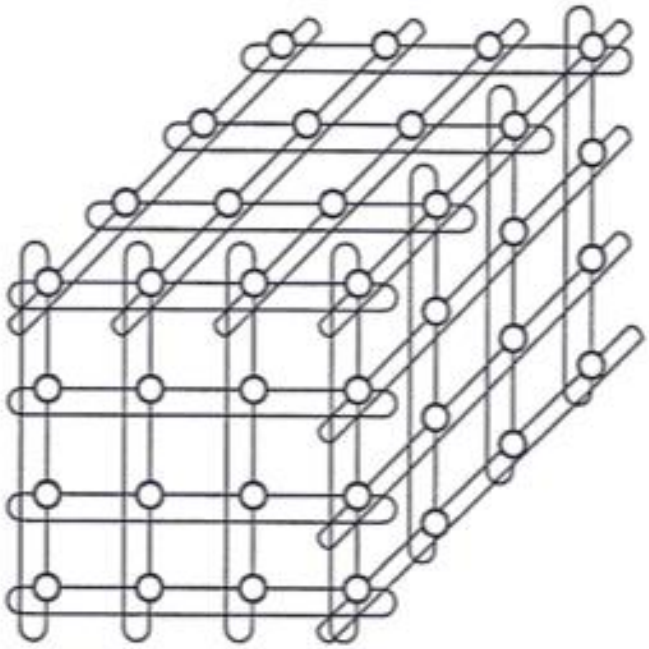
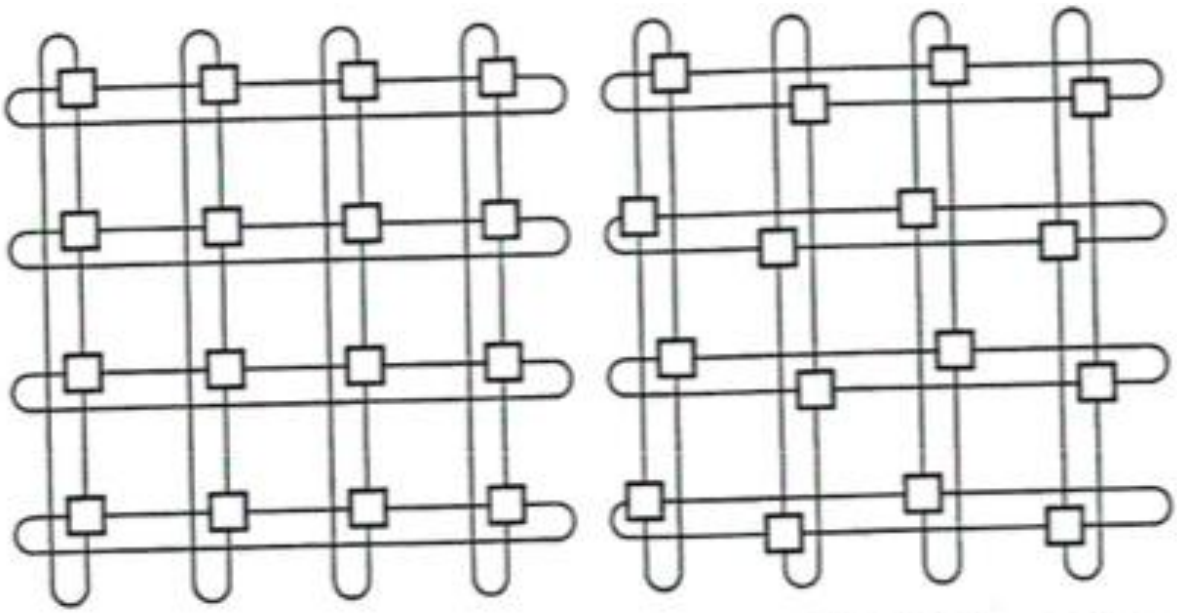


Figure 2.20 The k-ary n-cube network shown with k = 4 and n = 3; hidden nodes or connections are not shown.



(a) Traditional torus (a 4-ary 2-cube) (b) A torus with folded connections

Figure 2.21 Folded connections to equalize the wire length in a torus network. (Courtesy of W. Dally; reprinted with permission from *IEEE Trans. Computers*, June 1990)

K. Hwang, " Advanced Computer Architecture: Parallelism, Scalability, Programmability," Boston, MA: McGraw-Hill, 1992.

STATIC INTERCONNECT ARCHITECTURES

CHARACTERISTICS

N = # OF NODES
☆ OPTIMIZE FOR APPLICATION AND TYPICAL PROBLEM GRAIN SIZE

WIRE LENGTH = "CHANNEL" LENGTH = "LINK" LENGTH = "GRAPH EDGE" LENGTH
☆ TRY TO MINIMIZE

D = NETWORK DIAMETER
(SHORTEST DISTANCE TO FURTHEST NODE)

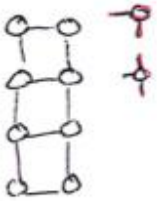
D = 4

D = 3

D = 4

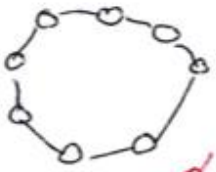
☆ MINIMIZE D
TO REDUCE COMMUNICATION DELAYS

d = DEGREE
(AVE # OF CONNECTION-CHANNELS INTO NODES)



d = 2.5

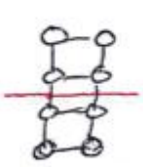
d = 3



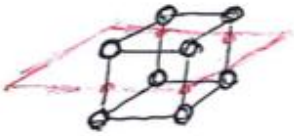
d = 2

☆ A CONSTANT d IS GOOD FOR MODULARITY AND SCALABILITY

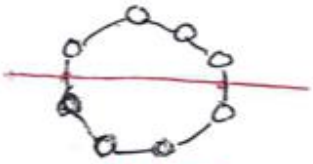
B = BISECTION WIDTH
(NUMBER OF LINKS THROUGH SMALLEST CROSS-SECTION)



B = 2



B = 4



B = 2

☆ MINIMIZE B TO MINIMIZE REQUIRED WIRING
☆ MAXIMIZE B TO MAX NETWORK COMM BANDWIDTH
☆ AS B↑, D↓ BUT d↑

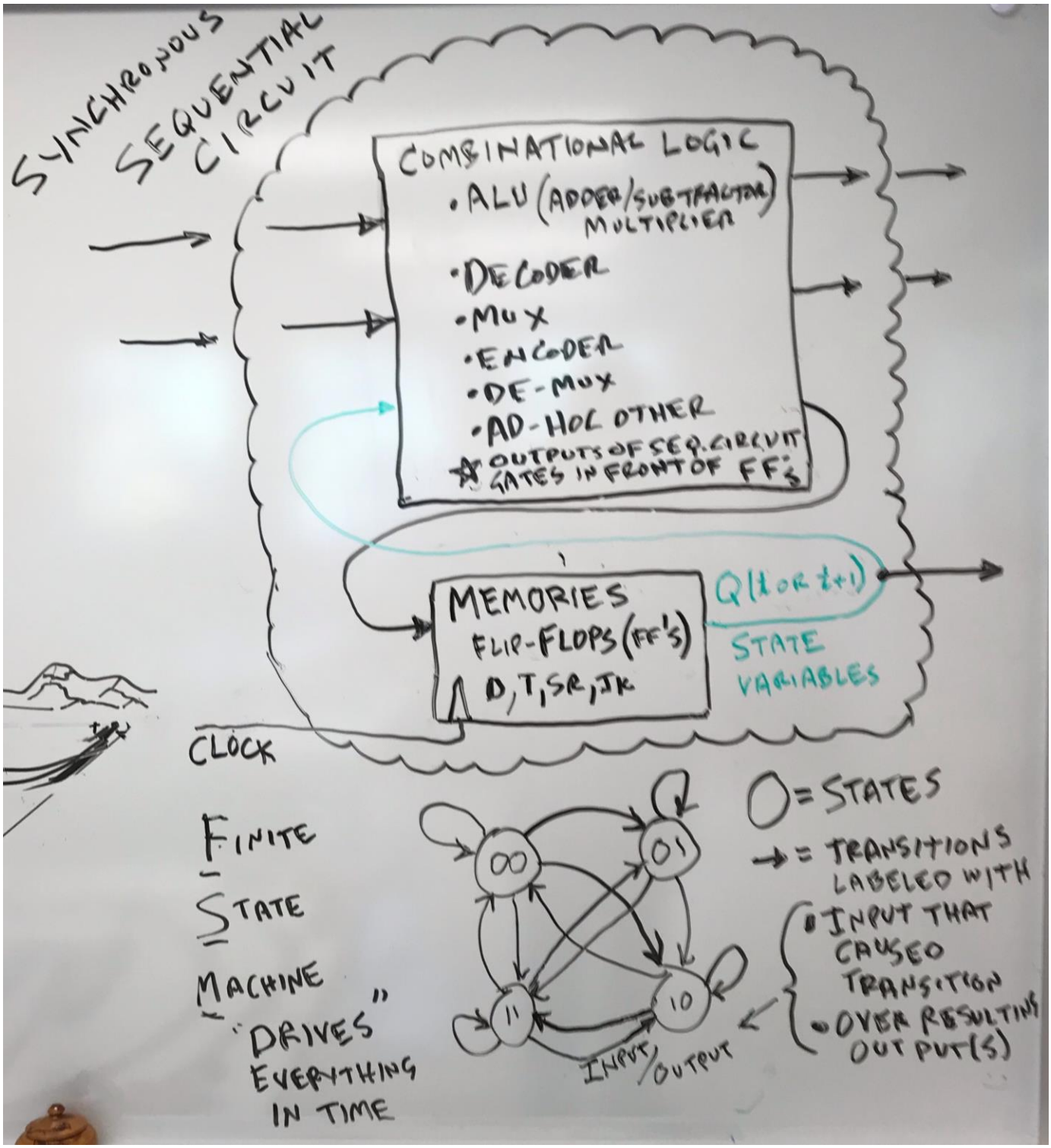
W = # OF WIRES IN ONE "CHANNEL", "LINK", "EDGE"

STATIC INTERCONNECT ARCHITECTURES

CHARACTERISTICS

W = # OF WIRES IN ONE "CHANNEL", "LINK", "EDGE"

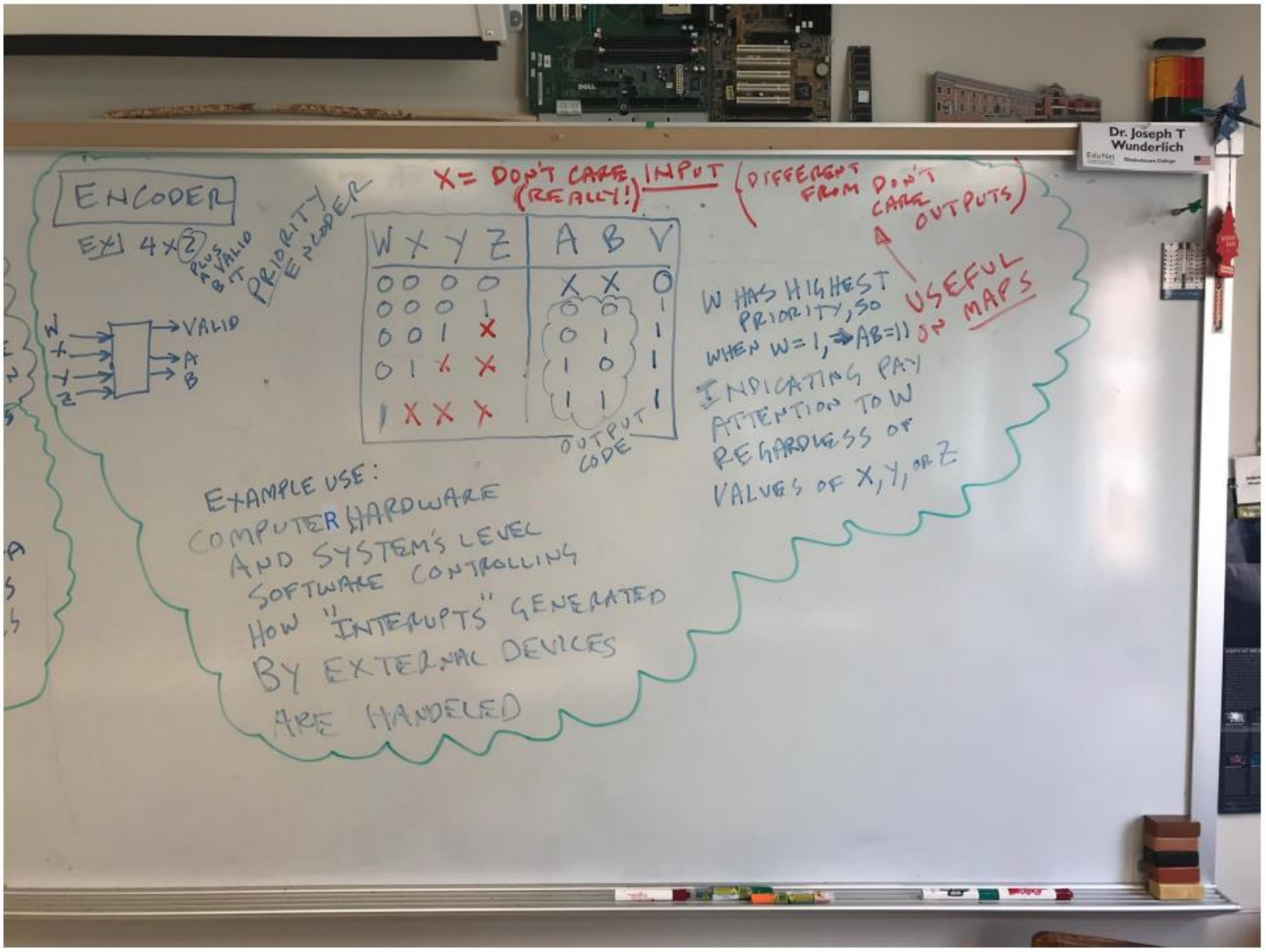
Recall PARALLEL (Combinational) and SERIAL (Sequential) Digital Design



From Prerequisite EGR/CS332 Digitqal Design I *(before Spring 2022)* *or* EGR330 Digital & Embedded Systems *(after Spring 2022)*

STATIC INTERCONNECT ARCHITECTURES

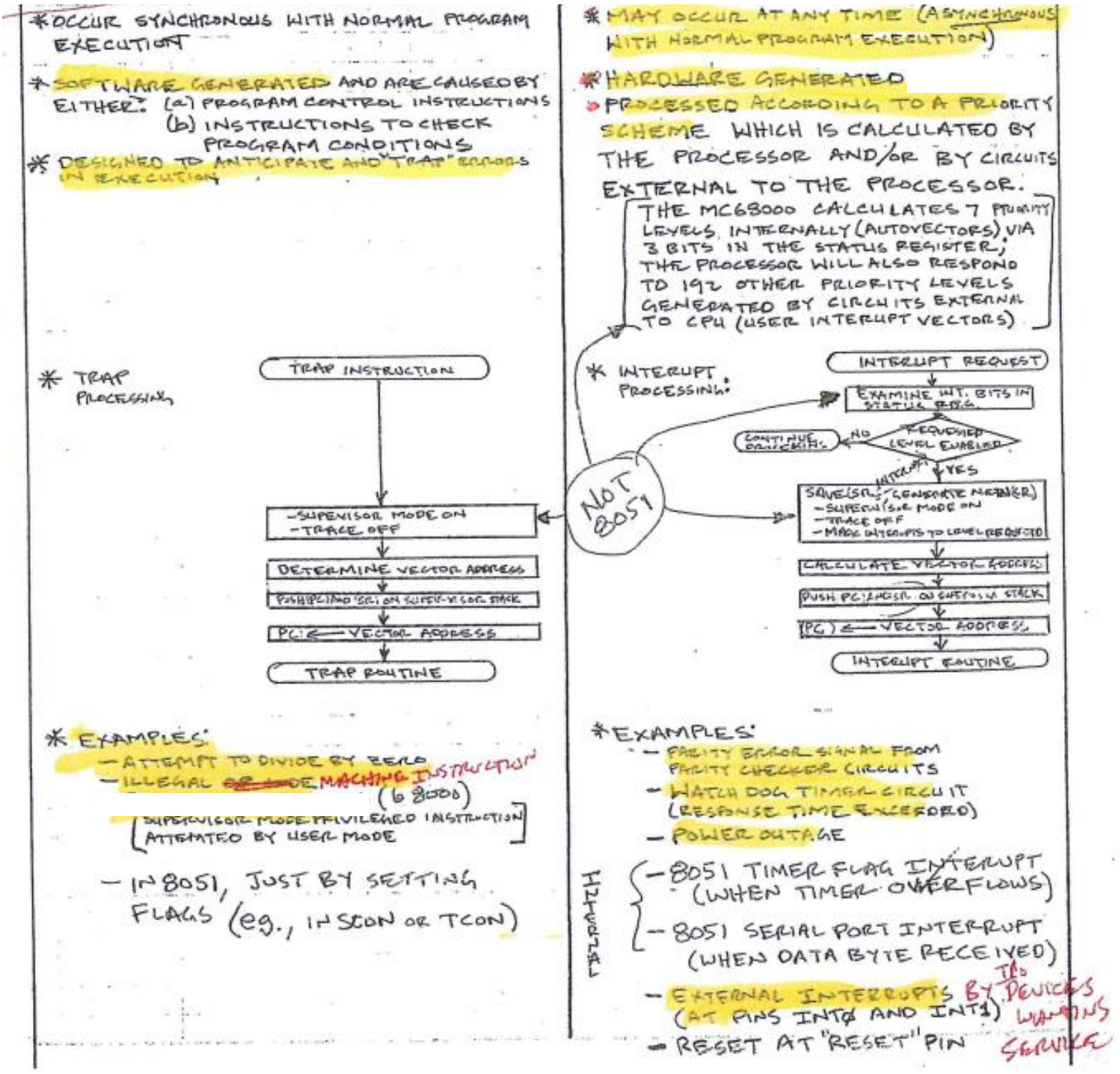
PARALLEL (Combinational) Digital Design
For COMMUNICATION with external device via INTERUPTS



From Prerequisite EGR/CS332 Digitqal Design I *(before Spring 2022)*
or EGR330 Digital & Embedded Systems *(after Spring 2022)*

COMMUNICATION with external devices via INTERUPTS

Software “TRAP” vs. Hardware “INTERUPT”



From [EGR/CS333 Digital Design II, Assembly Language, & Interfacing](#) *(before 2022)* or [EGR433/430 Advanced-Computer-Engineering/Parallel-Procewsing](#) *(after 2022)*

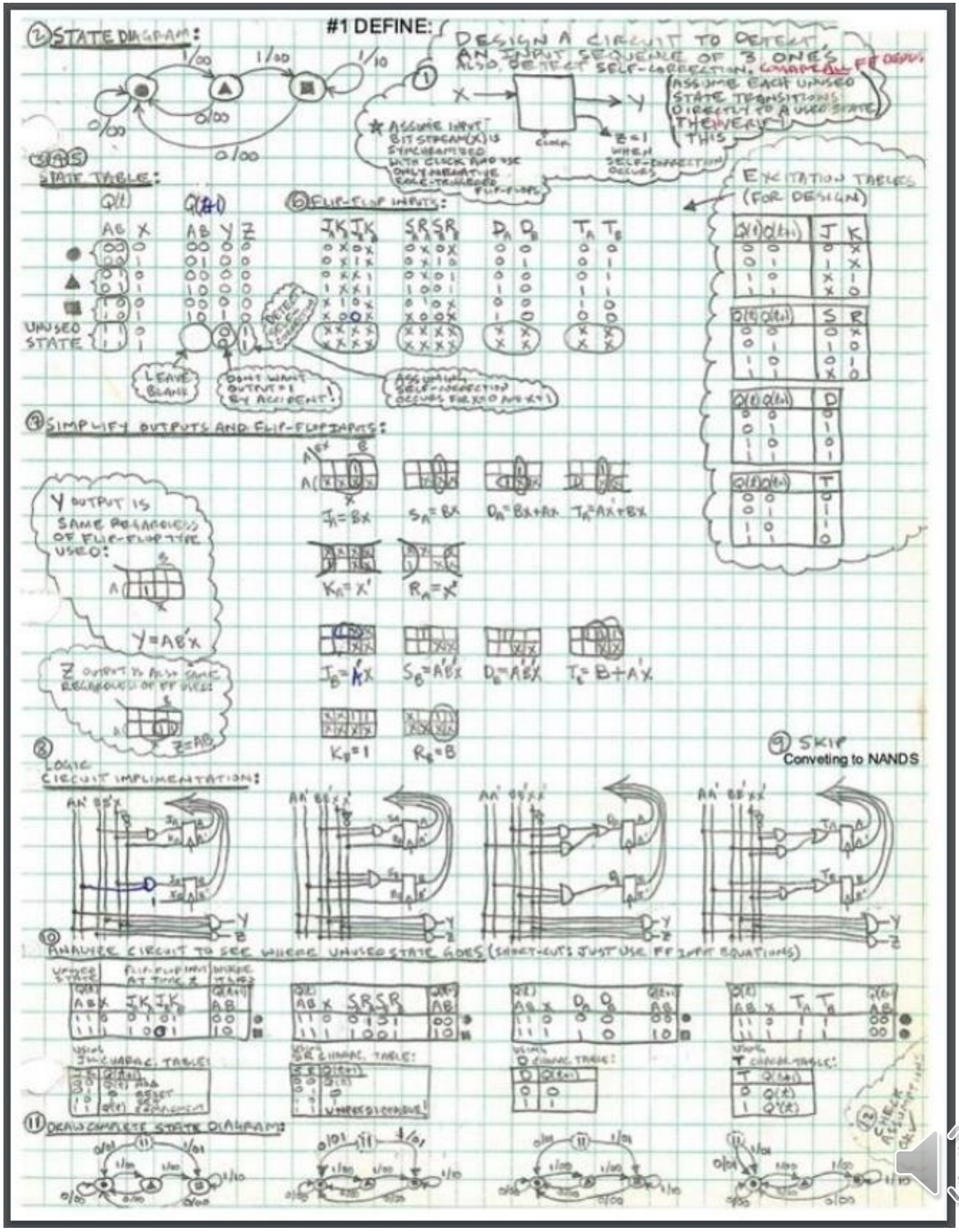
W= # OF WIRES IN ONE "CHANNEL", "LINK", "EDGE"

STATIC INTERCONNECT ARCHITECTURES

CHARACTERISTICS

SERIAL (Sequential) Digital Design

for **IPC** (Inter-Processor Communication) **BETWEEN PROCESSORS/CORES**



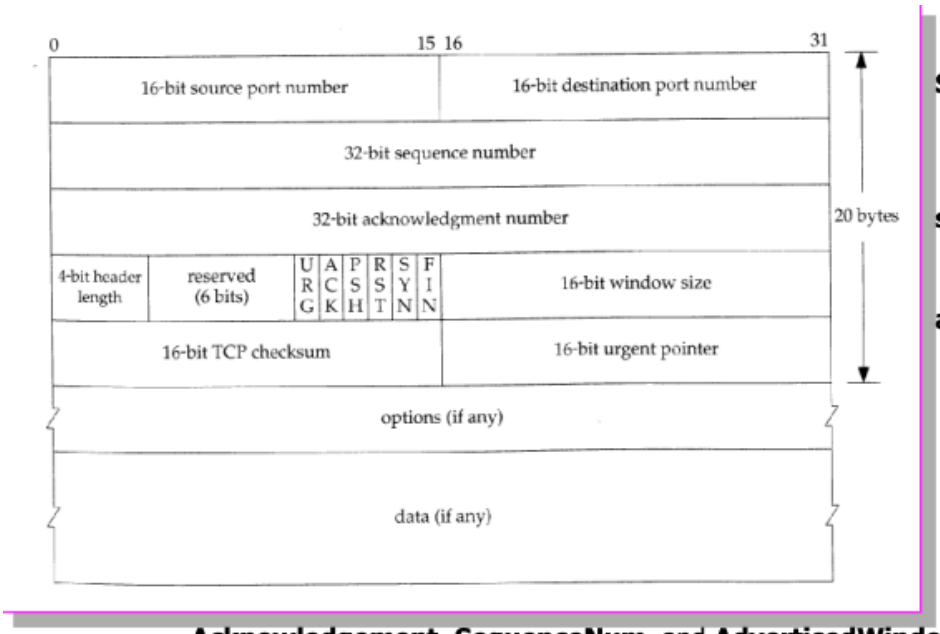
From Prerequisite EGR/CS332 Digitqal Design I *(before Spring 2022)*
or EGR330 Digital & Embedded Systems0 *(after Spring 2022)*

W= # OF WIRES IN ONE "CHANNEL", "LINK", "EDGE"

STATIC INTERCONNECT ARCHITECTURES CHARACTERISTICS

SERIAL **IPC** (Inter-Processor Communication) **BETWEEN PROCESSORS/CORES**

Sometimes PACKETIZED



SrcPort and **DstPort** fields identify source and destination ports. These plus source and destination IP addresses combine to identify each TCP connection.

sequence number identifies byte in data stream from sending TCP to receiving TCP that the first byte of data in this segment represents.

acknowledgement number is next sequence number that sender of acknowledgement expects to receive. i.e., sequence number plus 1 of last successfully received byte of data. This field is valid only if ACK flag is on. Once a connection is established Ack flag is always on.

Acknowledgement, **SequenceNum**, and **AdvertisedWindow** involved in TCP's sliding window algorithm.The Acknowledgement and AdvertisedWindow field carry info about flow of data going in other direction. In TCP's sliding window algorithm receiver advertises a window size to sender using the AdvertisedWindow field. The sender is then limited to having no more than a value of AdvertisedWindow bytes of unacknowledged data at any given time. The receiver sets a suitable value for the AdvertisedWindow based on the amount of memory allocated to the connection for the purpose of buffering data.

header length (in 32-bit words) Required because length of options field is variable.

6-bit Flags field used to relay control info between TCP peers. SYN and Fin flags for establishing and terminating a TCP connection, ACK flag is set any time Acknowledgement field is valid, implying that the receiver should pay attention to it. URG flag signifies this segment contains urgent data. When set, UrgPtr indicates where non-urgent data in this segment begins. PUSH flag signifies sender invoked push operation, which indicates to receiving side of TCP that it should notify the receiving process of this. RESET flag signifies receiver has become confused and so wants to abort connection.

Checksum (FOR ERROR DETECTION) is a mandatory field calculated by sender, then verified by receiver.

Option field is maximum segment size option, called MSS. Each end of connection normally specifies this option on first segment exchanged. It specifies maximum sized segment sender wants to receive.

Data portion of TCP segment (optional, but it's the actual data you are most likely trying to send!) i.e., everything else is communication overhead !!

NOTE: PACKETIZED SERIAL COMMUNICATION **between Systems also in Robotics**

SOURCE: : Crouse, J. (2008). **The Joint Architecture for Unmanned Systems (JAUS): a subsystem of the wunderbot 4**. *Elizabethtown College research report.*

Wunderbot 4 Wireless **JAUS** Communication is like TCP/IP , but different. Used for military comm

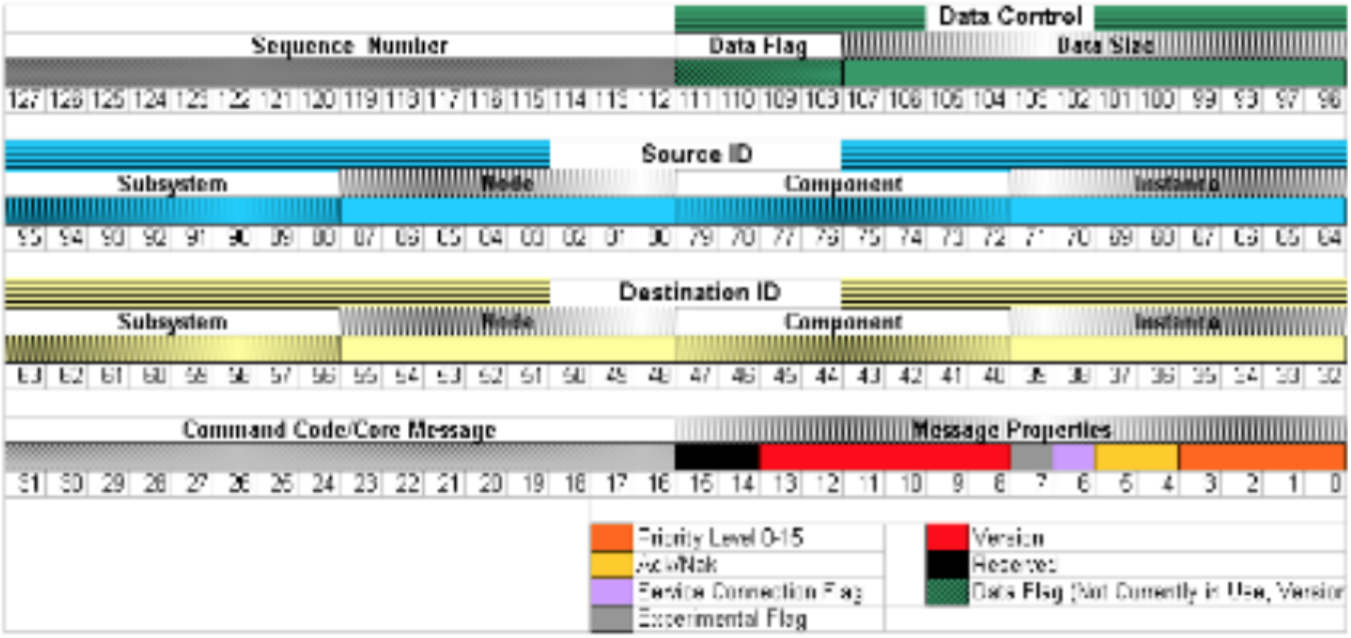
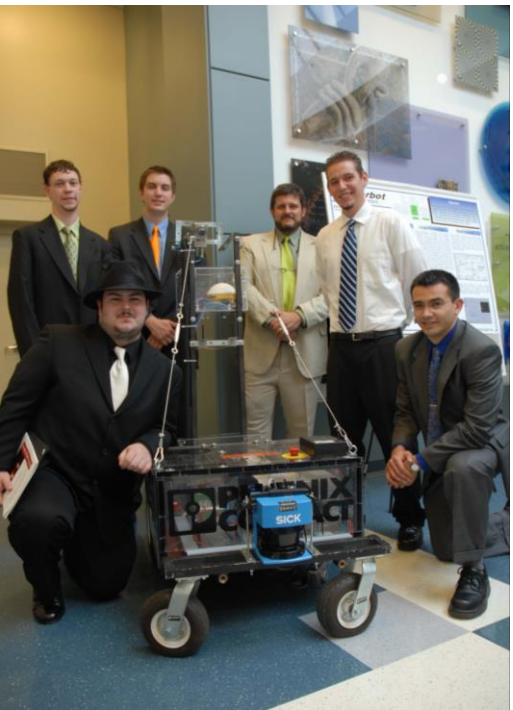


Figure 7: JAUS message header detailed structure [6]



For communication with other autonomous ground vehicles, like military tanks on Battlefield

From EGR/CS434 Robotics & Machine Intelligence

NOTE: PACKETIZED SERIAL COMMUNICATION **between Systems also in Robotics**

Navigation

Implementation and integration of the most recent Wunderbot systems:


Wunderbot 4

Wunderbot - Main VI Labview Tutorial
Wunderbot - GPS Subsystem Labview Tutorial
Wunderbot - LADAR Subsystem Labview Tutorial
Wunderbot - JAUS Subsystem Labview Tutorial
Wunderbot - Vision Subsystem Labview Tutorial
Wunderbot - Motor Control Subsystem Labview Tutorial
Wunderbot - Digital Compass Subsystem Labview Tutorial
Wunderbot - MCglobal08 Subsystem Labview Tutorial
nanoLC Robot Simulation

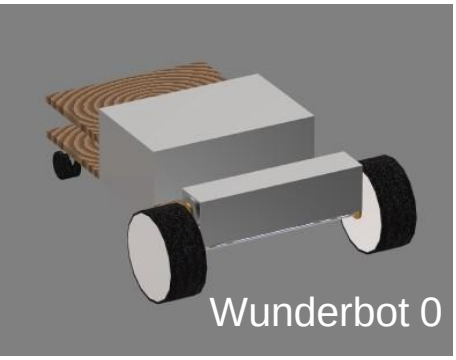
And theory and design decisions here:

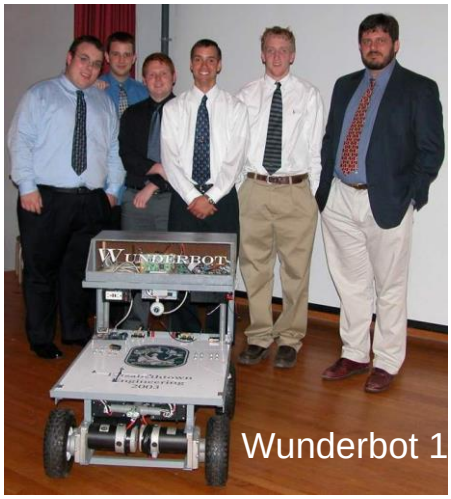
[1] Painter, J. and Wunderlich, J.T. (2008). **Wunderbot IV: autonomous robot for international competition.** In *Proceedings of the 12th World Multi-Conference on Systemics, Cybernetics and Informatics: WMSCI 2008, Orlando, FL*: (pp. 62-67). And HERE

[2] Coleman, D. and Wunderlich, J.T. (2008). **O^3: an optimal and opportunistic path planner (with obstacle avoidance) using voronoi polygons.** In *Proceedings of IEEE the 10th international Workshop on Advanced Motion Control, Trento, Italy.* vol. 1, (pp. 371-376). IEEE Press.

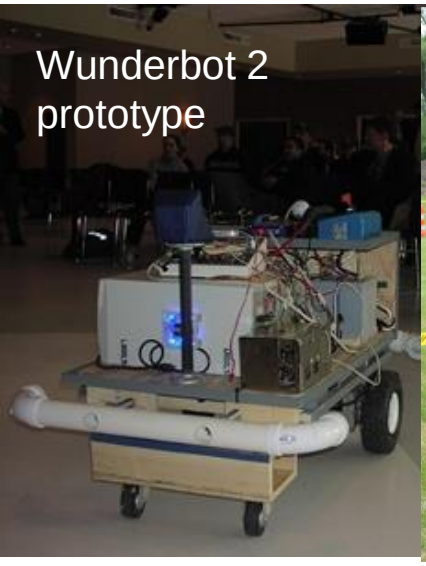
[3] JAUS wireless packetized communication by Jeremy Crouse

NOTE: PACKETIZED SERIAL COMMUNICATION **between Systems also in Robotics**





All recent Wunderbots share website: http://www2.etown.edu/wunderbot/



See evolution 1999:
http://users.etown.edu/w/wunderjt/home_researchMINE.html
http://users.etown.edu/w/wunderjt/Weblab_archive.htm

From EGR/CS434 Robotics & Machine Intelligence

STATIC INTERCONNECT ARCHITECTURES

SCALABILITY

Table 2.2 Summary of Static Network Characteristics

Network type	Node degree, d	Network diameter, D	No. of links, l	Bisection width, B	Symmetry	Remarks on network size
Linear Array	2	$N-1$	$N-1$	1	No	N nodes
Ring	2	$\lfloor N/2 \rfloor$	N	2	Yes	N nodes
Completely Connected	$N-1$	1	$N(N-1)/2$	$(N/2)^2$	Yes	N nodes
Binary Tree	3	$2(h-1)$	$N-1$	1	No	Tree height $h = \lceil \log_2 N \rceil$
Star	$N-1$	2	$N-1$	$\lfloor N/2 \rfloor$	No	N nodes
2D-Mesh	4	$2(r-1)$	$2N-2r$	r	No	$r \times r$ mesh where $r = \sqrt{N}$
Illiac Mesh	4	$r-1$	$2N$	$2r$	No	Equivalent to a chordal ring of $r = \sqrt{N}$
2D-Torus	4	$2\lfloor r/2 \rfloor$	$2N$	$2r$	Yes	$r \times r$ torus where $r = \sqrt{N}$
Hypercube	n	n	$nN/2$	$N/2$	Yes	N nodes, $n = \log_2 N$ (dimension)
CCC	3	$2k-1+\lfloor k/2 \rfloor$	$3N/2$	$N/(2k)$	Yes	$N = k \times 2^k$ nodes with a cycle length $k \geq$
k-ary n-cube	$2n$	$n\lfloor k/2 \rfloor$	nN	$2k^{n-1}$	Yes	$N = k^n$ nodes

K. Hwang, " Advanced Computer Architecture: Parallelism, Scalability, Programmability," Boston, MA: McGraw-Hill, 1992.

DYNAMIC INTERCONNECT ARCHITECTURES

BUS

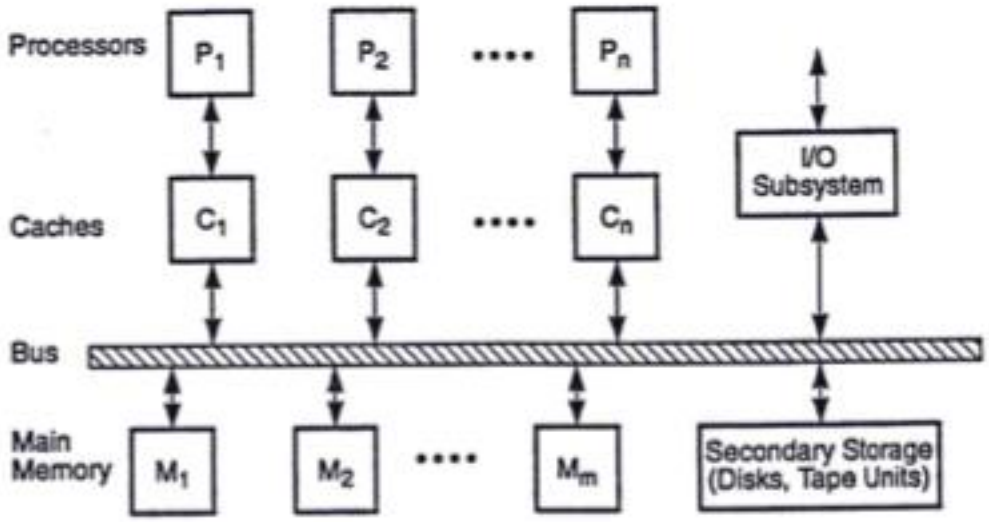


Figure 2.22 A bus-connected multiprocessor system, such as the Sequent Symmetry S1.

K. Hwang, " Advanced Computer Architecture: Parallelism, Scalability, Programmability," Boston, MA: McGraw-Hill, 1992.

DYNAMIC INTERCONNECT ARCHITECTURES

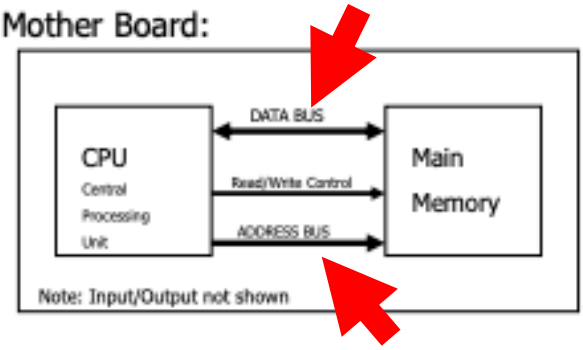
PC (Personal Computer) BUS's

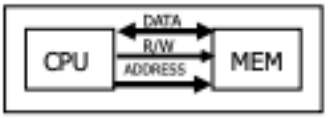
Conceptual Computer Architecture

Dr. J. T. Wunderlich
Elizabethtown College

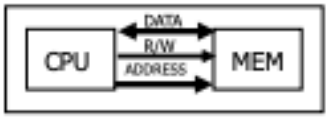
Conceptual Computer Architecture for typical PC

- Mother Board:

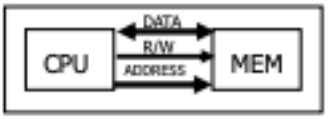




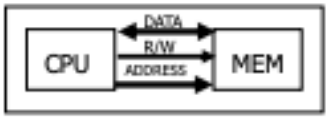
- CPU (Central Processing Unit):
  - Processes all Machine Instructions
  - Contains some "fast-access" memory (usually static-RAM or "Latches" made from "D type Flip-Flops")



- MEMORY ("Main Memory"):
  - Stores Machine instructions
  - Stores Data (i.e., "operands" for instructions)
  - Typically made from Dynamic RAM (Random Access Memory)



- DATA BUS:
  - 8 to 128 bit wide electrical pathway
  - Exchange data between CPU and "Memory"
  - Exchange machine instructions from CPU to "Memory"



- ADDRESS BUS:
  - 16 to 40 bit wide electrical pathway
  - Takes Memory addresses generated by CPU to address memory locations

From EGR/CS230 Computer Architecture

DYNAMIC INTERCONNECT ARCHITECTURES

BUS's

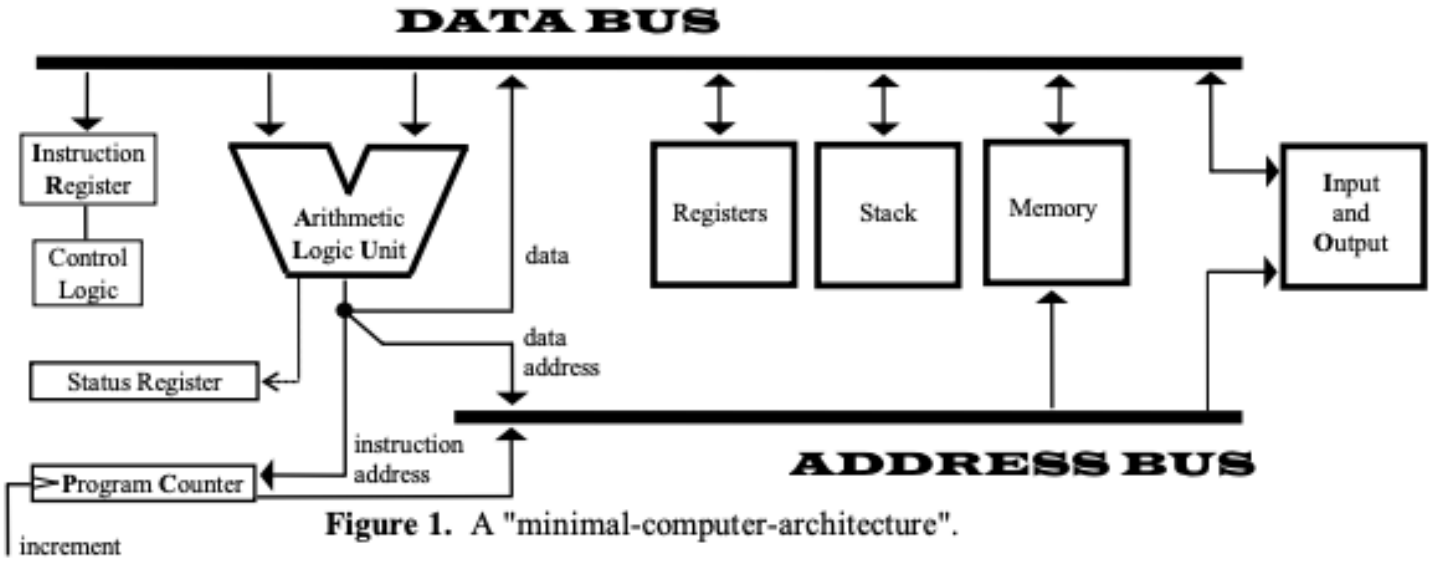


Figure 1. A "minimal-computer-architecture".

As shown in Fig. 1, each device contains:

- A program counter to address instructions to be fetched from memory.
- An instruction register to put the fetched instruction in.
- Control logic to create all routing signals after decoding the fetched instruction.
- An ALU for arithmetic and logical manipulation of data and addresses.
- Registers for storing intermediate results of calculations.
- A status register for status flags and condition codes.
- Memory for storing data and instructions.
- A stack for storing addresses (or processor status) for returning from program-calls (or interrupts).
- I/O which is addressed as memory (i.e., memory-mapped I/O).

Wunderlich, J.T. (1999). **Focusing on the blurry distinction between microprocessors and microcontrollers.** In *Proceedings of 1999 ASEE Annual Conference & Exposition, Charlotte, NC*: (session 3547), [CD-ROM]. ASEE Publications.

DYNAMIC INTERCONNECT ARCHITECTURES

BUS's

Microprocessors vs. Microcontrollers

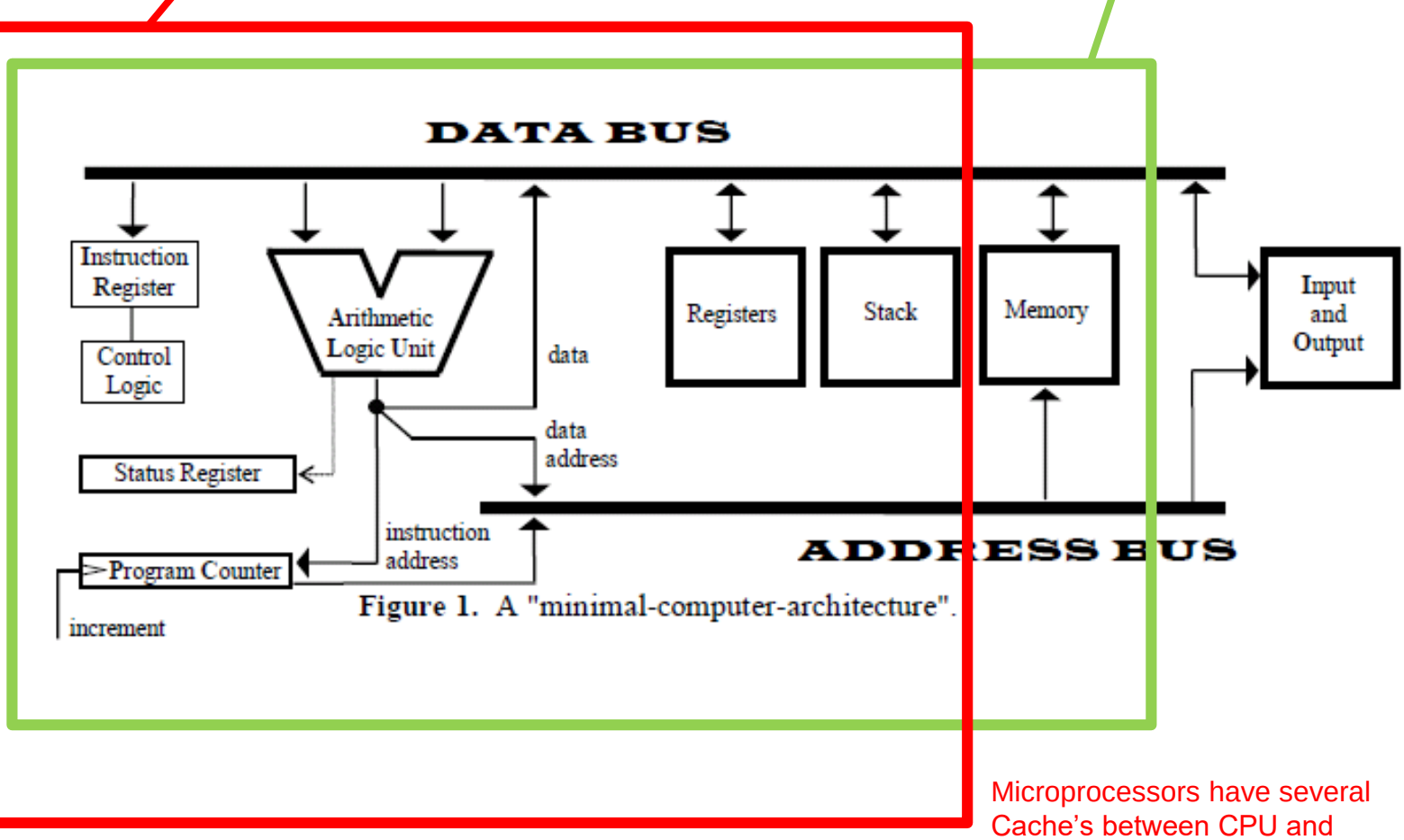


Figure 1. A "minimal-computer-architecture".

Microprocessors have several Cache's between CPU and "Memory"

Wunderlich, J.T. (1999). **Focusing on the blurry distinction between microprocessors and microcontrollers.** In *Proceedings of 1999 ASEE Annual Conference & Exposition, Charlotte, NC*: (session 3547), [CD-ROM]. ASEE Publications.

DYNAMIC INTERCONNECT ARCHITECTURES

BUS's

In PC's: **Back-Side Bus, Front-Side-Bus (FSB)**

Microprocessors

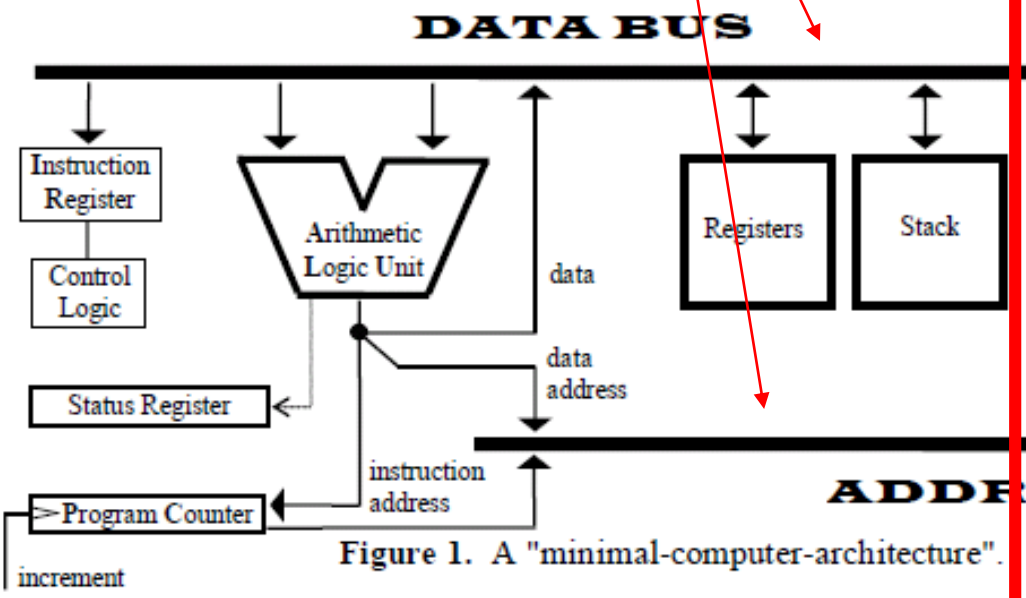


Figure 1. A "minimal-computer-architecture".

Microprocessors have several Cache's between CPU and "Memory"

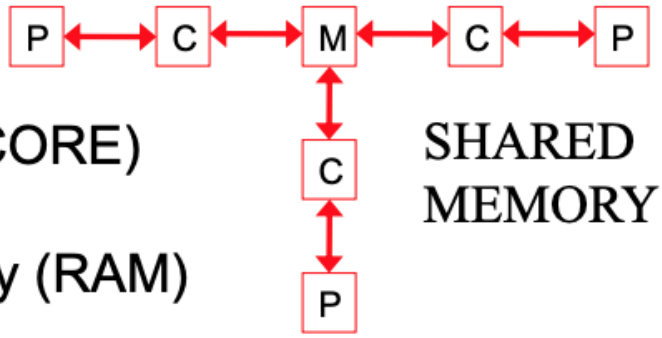
Wunderlich, J.T. (1999). **Focusing on the blurry distinction between microprocessors and microcontrollers.** In *Proceedings of 1999 ASEE Annual Conference & Exposition, Charlotte, NC*: (session 3547), [CD-ROM]. ASEE Publications.

DYNAMIC INTERCONNECT ARCHITECTURES

PC (Personal Computer) BUS's

PC Design Steps (Pick CPU)

- Multi-core, (multiple CPU's in chip package)
  - Don't assume speed-up proportional, yet! – there is recent research on treating Cores like network nodes where performance of overall system actually increases with more nodes, unlike the decrease in performance when many more cores share one memory
  - SMP (Symmetric Multi-Processing)



P = Processor (CORE)
C = Cache
M = Main Memory (RAM)

SHARED MEMORY !

- Cache(s) common "on-chip" (usually now in same chip package, and on same **silicon chip** -- i.e., same piece of silicon chipped off of wafer that is sliced off of big cylindrical ingot)
  - Typically L1 cache now integrated into CPU''s ("CORE's")
  - Typically L2 cache connected to CPU's via "Back-Side-Bus" (in same chip-package!) connecting everything together on the chip (unlike the **FSB** Front Side Bus which connects the CPU chip package to the main memory RAM on the motherboard

- Cooling requirements (e.g., heat-sink, dedicated-fan, heat-pipe, liquid)
  - Especially if you plan on overclocking CPU

From Lecture on "Design a PC"
PDF PPTX-w/audio MP4 YouTube

in EGR/CS230 Computer Architecture

DYNAMIC INTERCONNECT ARCHITECTURES

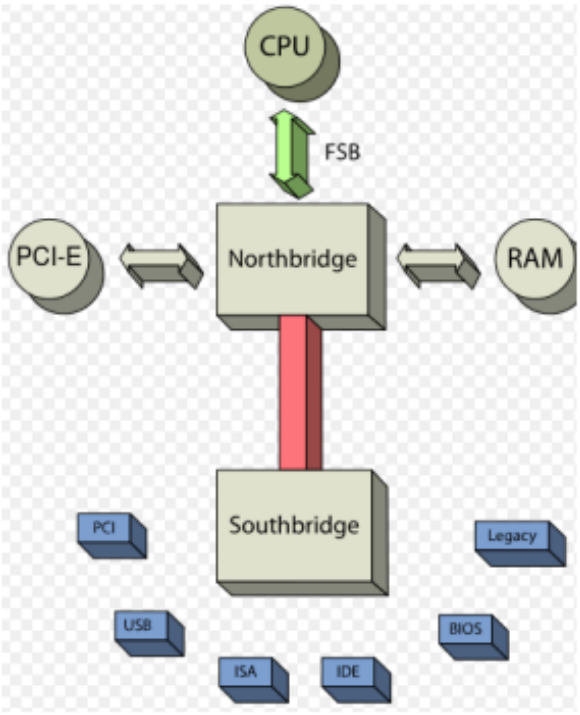
PC (Personal Computer) BUS's

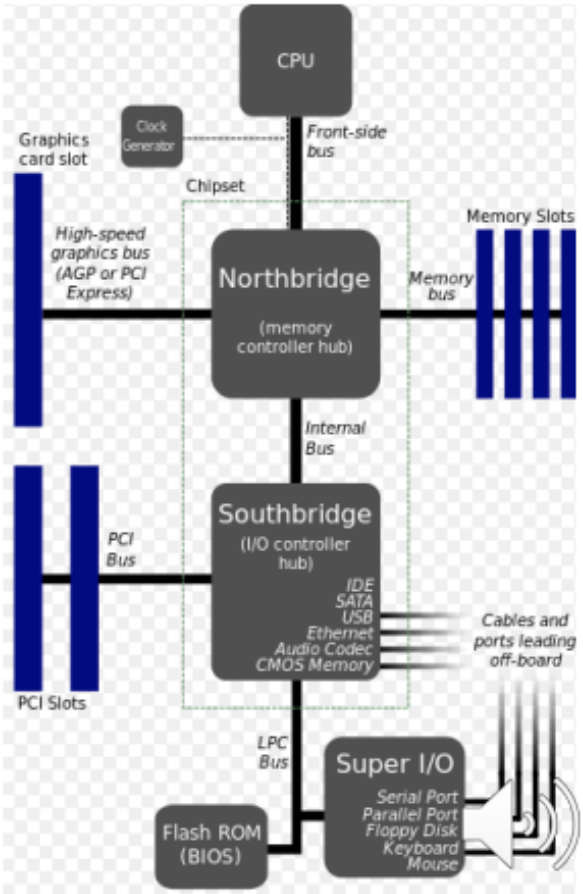
PC Design Steps (Pick Motherboard)

- Match speed of CPU, RAM, and Motherboard **F**ront-**S**ide-**B**us (**FSB**) which connects CPU and RAM
- Make sure it has socket to to plug in your CPU (i.e. Intel or AMD)
- Make sure it has correct **chip-set** to handle your CPU, RAM, Graphics Card, and other I/O needs

Northbridge for RAM and video card control,
and restricts overclocking

Southbridge for power, clock, and other I/O control





Images from:
https://en.wikipedia.org/wiki/Northbridge_(computing) v

From Lecture on "Design a PC"
PDF PPTX-w/audio MP4 YouTube

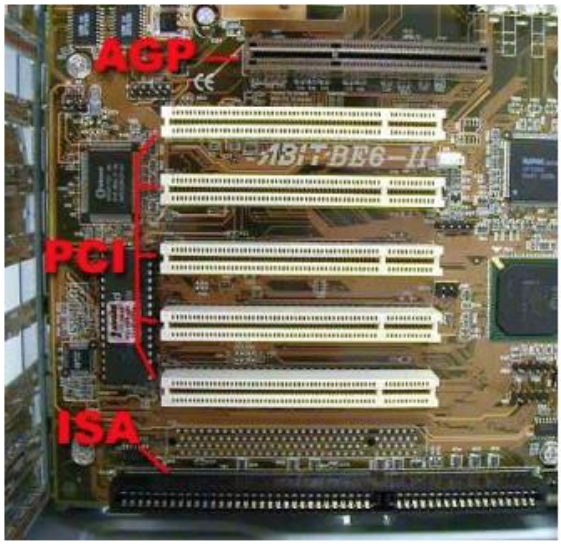
in EGR/CS230 Computer Architecture

DYNAMIC INTERCONNECT ARCHITECTURES

PC (Personal Computer) BUS's

PC Design Steps (Pick Motherboard)

- May want **Dual-Channel** capability (can handle two banks of RAM concurrently)
- Make sure Motherboard can **handle your Graphics Card** (NVIDIA, etc)



OLD:

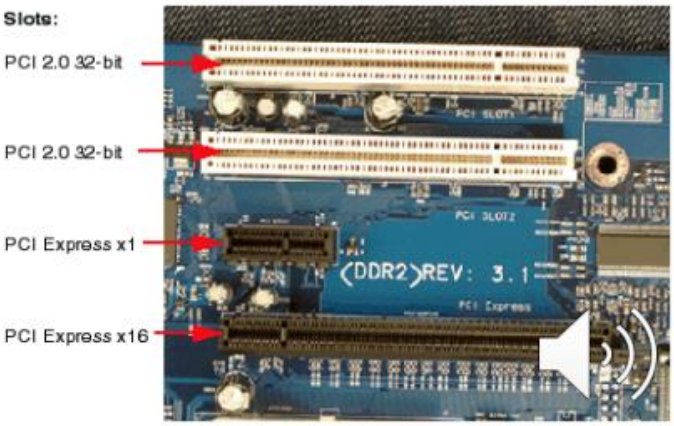
- **AGP** ("Accelerated Graphics Port")
- **PCI**
- **ISA**

RECENT:

PCIexpess (not a bus protocol)

ISA, PCI, and AGP use PARALLEL communication of data

*PCIexpress uses a packetizing **SERIAL** protocol like that used for Ethernet TCP/IP, and then many serial lines are implemented in parallel*



From Lecture on "Design a PC"

PDF PPTX-w/audio MP4 YouTube

in EGR/CS230 Computer Architecture

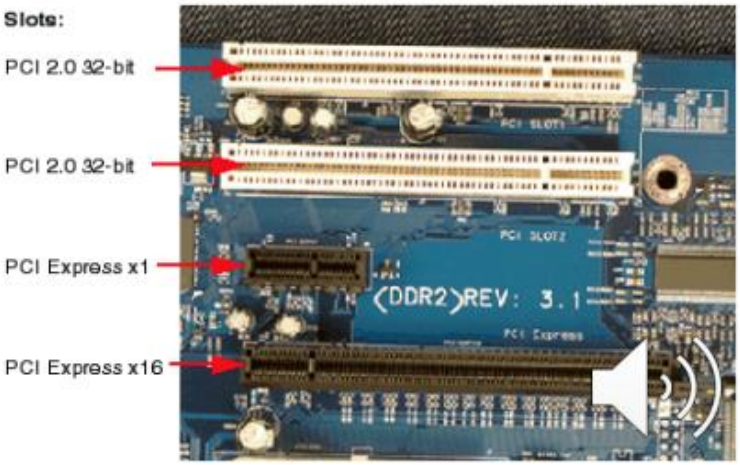
DYNAMIC INTERCONNECT ARCHITECTURES

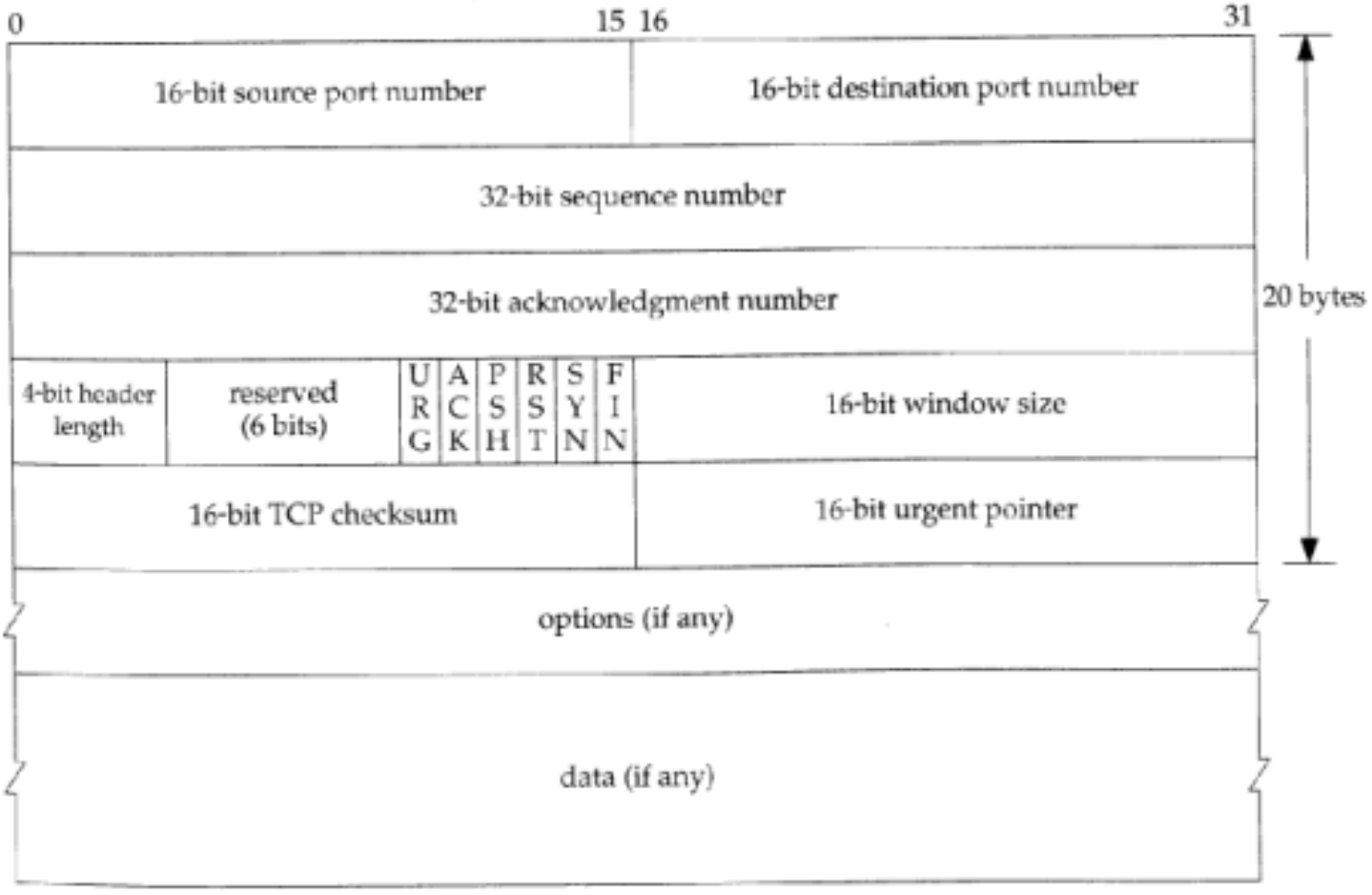
PC (Personal Computer) BUS's

PCIexpess (not a bus protocol)

ISA, PCI, and AGP use PARALLEL communication of data

*PCIexpress uses a packetizing **SERIAL** protocol like that used for Ethernet TCP/IP, and then many serial lines are implemented in parallel*





From Lecture on "Design a PC"
PDF PPTX-w/audio MP4 YouTube
in EGR/CS230 Computer Architecture

DYNAMIC INTERCONNECT ARCHITECTURES

BUS

Design-Lab of a **SERIAL BUS** in EGR/CS333 Digital Design II, Assembly Language, & Interfacing

Create a Serial BUS and your own communication protocol (like packetized TCP/IP, but simpler) to connect previous lab projects.

Review Course Lecture: http://users.etown.edu/w/wunderjt/Communications.pdf
And our custom Lab Manuals:

- **2018 IC's, Circuit Trainer, and Power Supply**
- **2019 Relays**
- **2019 NanoLC PLC*(Programmable Logic Controller)***
- **2019 Rasberry Pi and ARM Microcontroller**
- **2019 Intel 8051 Microcontroller** (2015, 2014, Pre-2013)
- **2019 AXIOLINE PLC**
- **2021 PLC-NEXT** *(in development during this course)*

REAL-TIME

1. Make a NanolineLC **(or PLC NEXT or Axioline PLC,)** implement shifting in a nibble; And, shifting out a nibble; shift it in, store it, and then shift it back back out of PLC. Deal with the parity of bits coming into, or leaving PLC (i.e., do error-detection/parity-check)
2. Using tri-state buffers and appropriate relays between devices (Phoenix Contact, Arduino, or others located in lab), and needed control lines communicating within and between devices, create a BIDIRECTIONAL SINGLE WIRE SERIAL communication DATA BUS shared by one circuit trainer (with circuits from a previous lab) and the PLC. You can have as many control lines as you wish, but only one shared bidirectional serial single-wire data bus.
3. Demonstrate bidirectional serial communication with parity checking
4. Use your circuit trainer clock generator to see how fast you can communicate to the PLC (i.e., if you can communicate as fast as the fastest clock generator available)

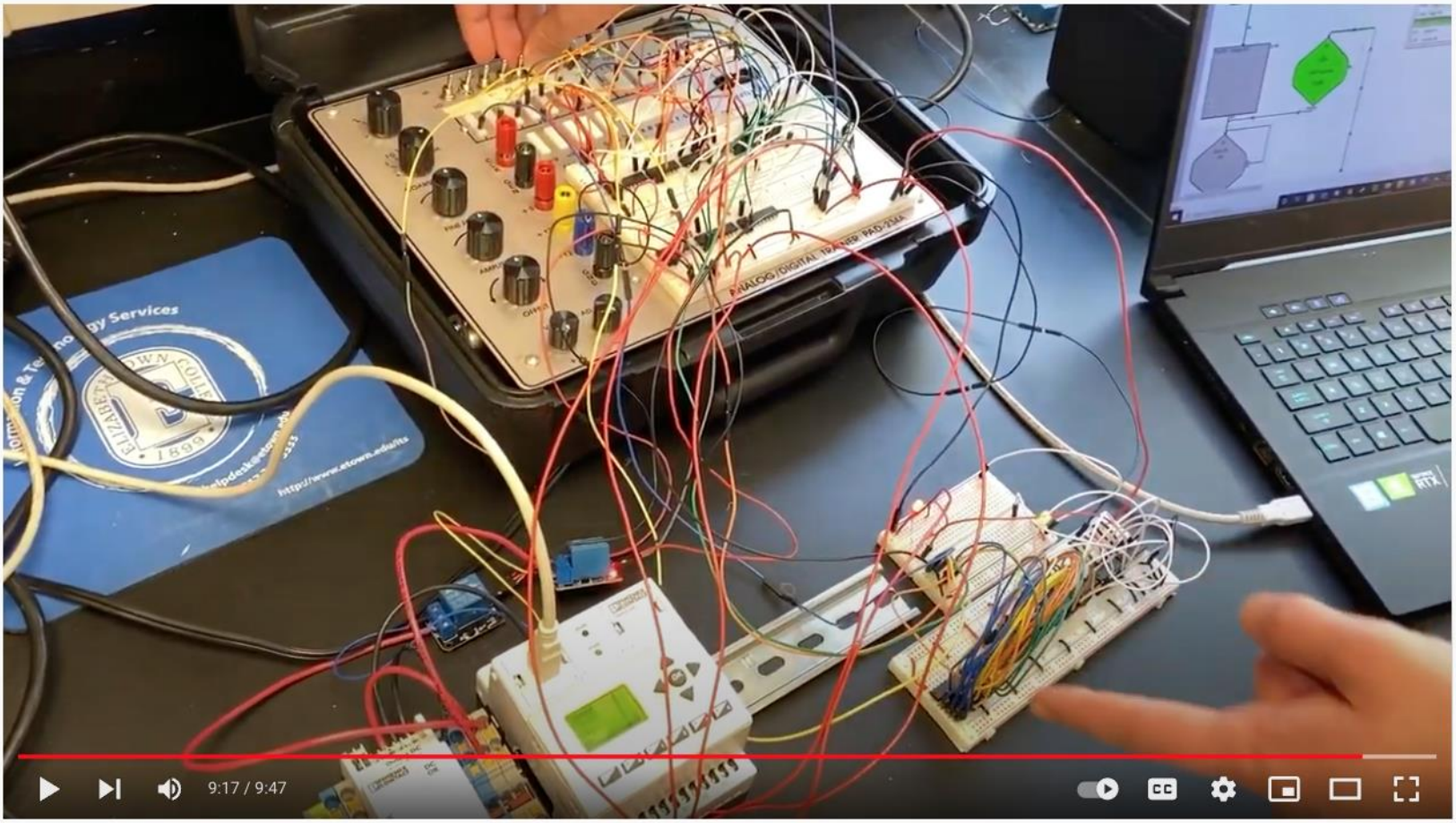
SIMULATION

1. Create a complete simulation in Logisim of everything above including forced error in data transmission, clock generators, and a logic facsimile of what your PLC is actually doing (i.e., translate your PLC logical decisions and actions into equivalent logic gates in Logisim)
2. Demonstrate simulated versions of control schemes of other devices when available in development system (e.g., NanoLC)
3. Simulate all above communications in 8051 Assembly Language (just some approximation of functionality)

GRADE: (75%) 8 to 10 minute (total) video of all Simulation and Real-Time Circuits; (25%) Lab Report **PER SYLLABUS SPECIFICATIONS!) of entire Design process** Just one person submit both of these, and the same grade will be given to each team member.

DYNAMIC INTERCONNECT ARCHITECTURES
BUS

Design-Lab of a **SERIAL BUS**
in [EGR/CS333 Digital Design II, Assembly Language, & Interfacing]



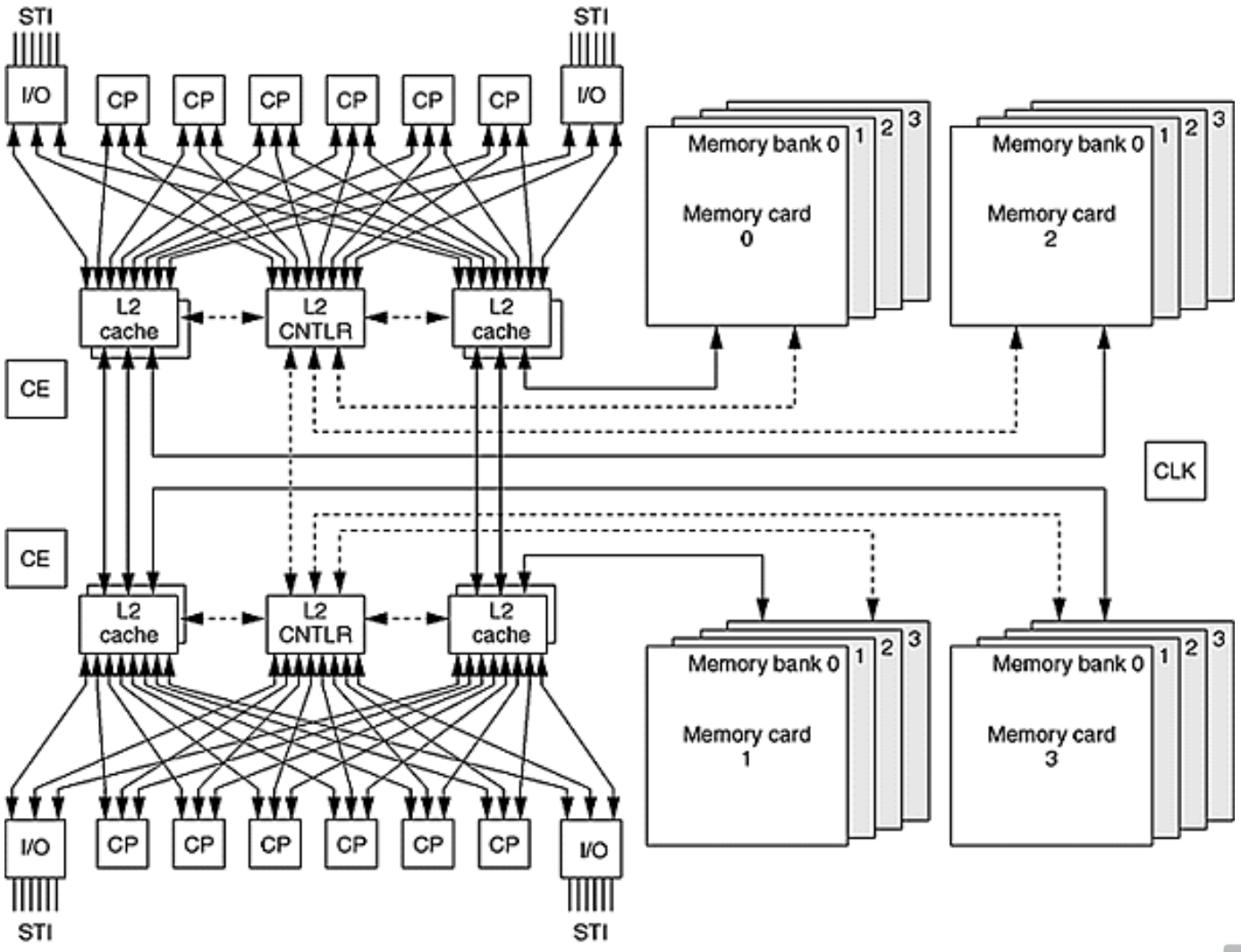
Digital Design II Lab 4 Simulations and Real Time Circuit Demonstration

https://www.youtube.com/watch?v=2csorAkpjyg

DYNAMIC INTERCONNECT ARCHITECTURES
BUS

IBM S/390 of 1990's
(IBM Z-Series 2000 to 2020's)

J Wunderlich PhD.
Advisory-Level Engineer, & Researcher



https://www.computer.org/csdl/magazine/mi/1999/02/m2012/13rRUNvyaq0

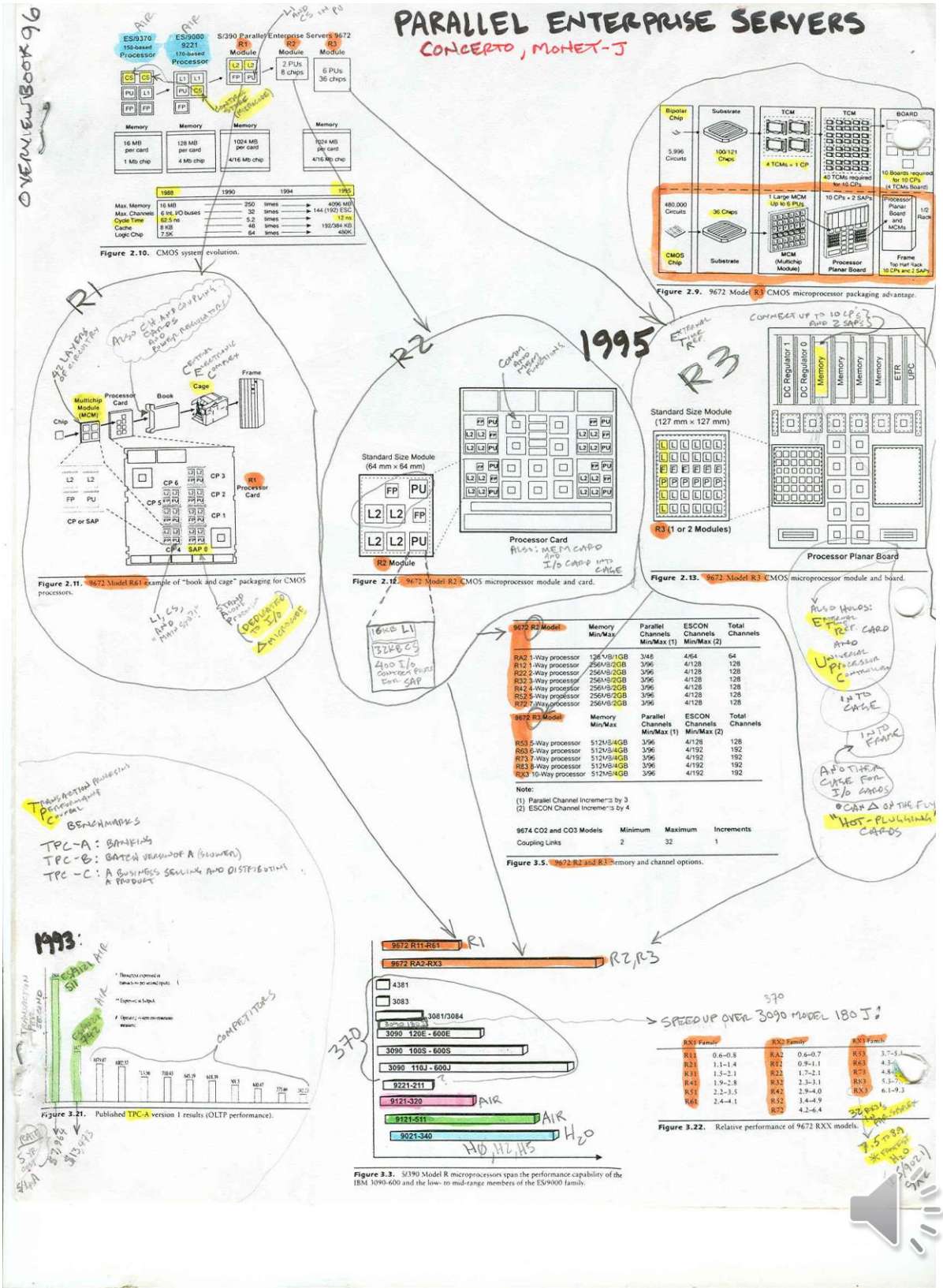
DYNAMIC INTERCONNECT ARCHITECTURES

BUS

IBM S/390 of 1990's (IBM Z-Series 2000 to 2020's)

J Wunderlich PhD.
Advisory-Level Engineer, & Researcher

http://users.etown.edu/w/wunderjt/home_IBM.html



DYNAMIC INTERCONNECT ARCHITECTURES

BUS

IBM S/390 of 1990's (IBM Z-Series 2000 to 2020's)

J Wunderlich PhD.
Advisory-Level Engineer, & Researcher

http://users.etown.edu/w/wunderjt/home_IBM.html

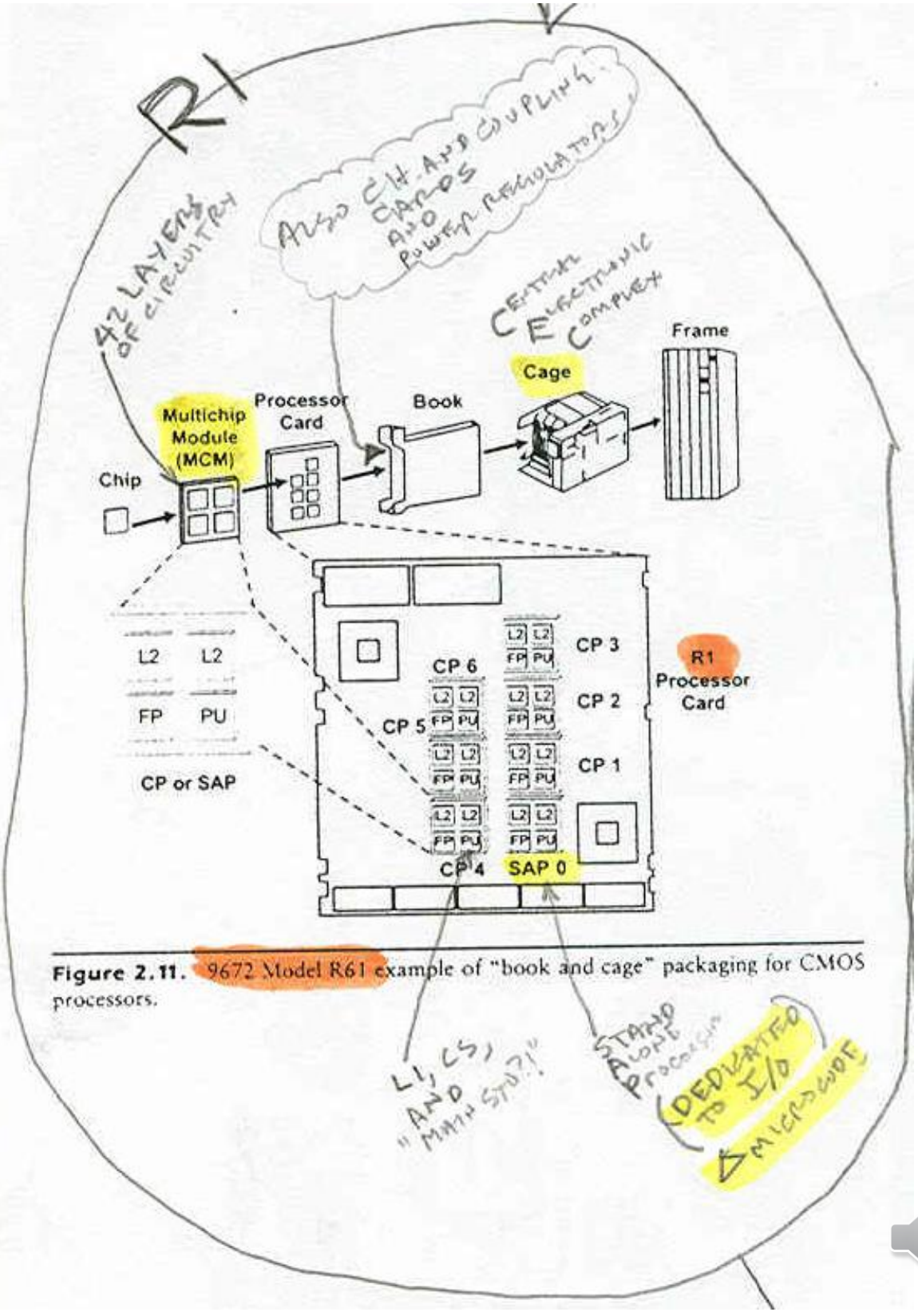


Figure 2.11. 9672 Model R61 example of "book and cage" packaging for CMOS processors.

DYNAMIC INTERCONNECT ARCHITECTURES

BUS

IBM S/390 of 1990's (IBM Z-Series 2000 to 2020's)

J Wunderlich PhD.
Advisory-Level Engineer, & Researcher

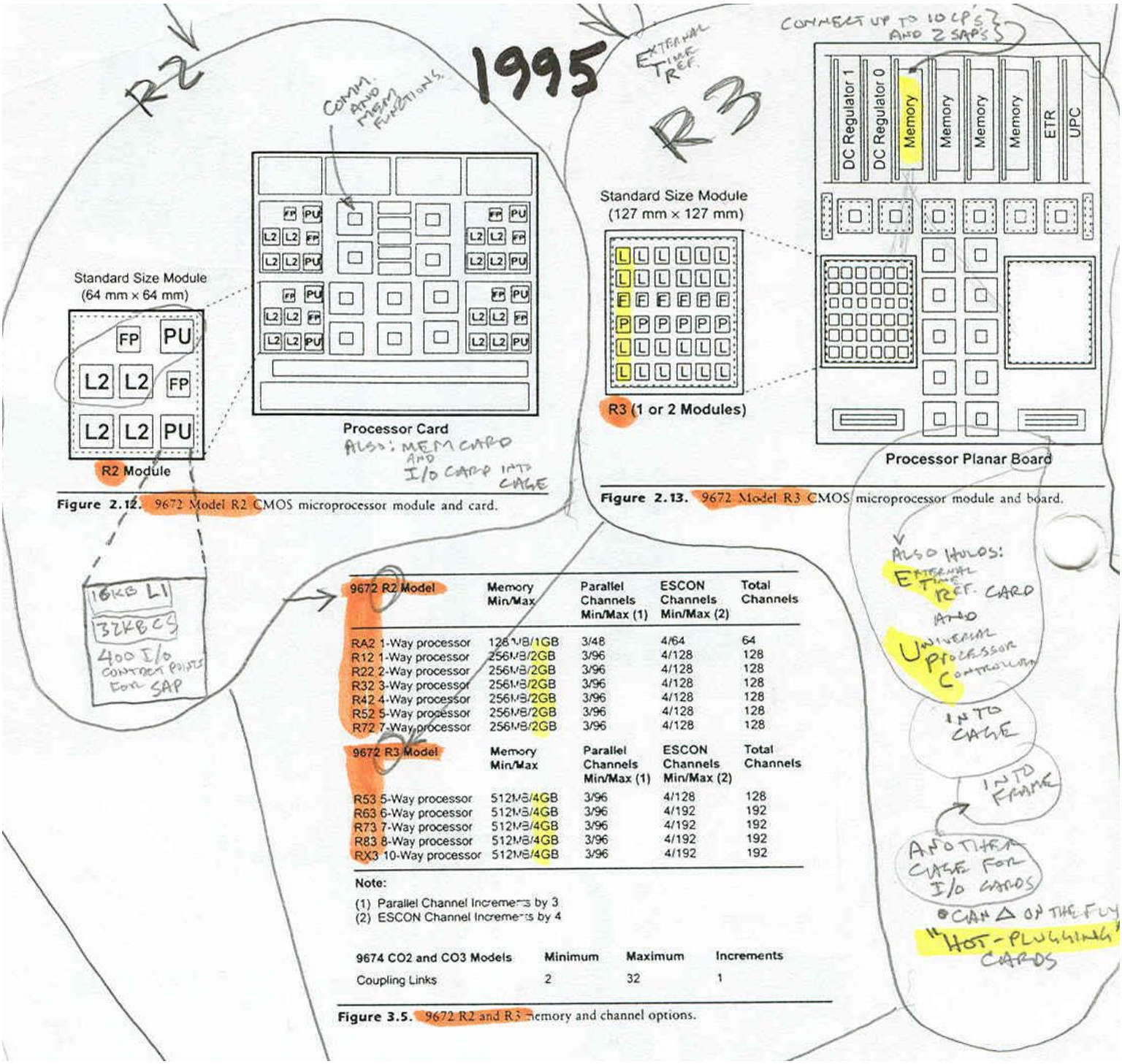


Figure 2.12. 9672 Model R2 CMOS microprocessor module and card.

Figure 2.13. 9672 Model R3 CMOS microprocessor module and board.

9672 R2 Model	Memory Min/Max	Parallel Channels Min/Max (1)	ESCON Channels Min/Max (2)	Total Channels
RA2 1-Way processor	128MB/1GB	3/48	4/64	64
R12 1-Way processor	256MB/2GB	3/96	4/128	128
R22 2-Way processor	256MB/2GB	3/96	4/128	128
R32 3-Way processor	256MB/2GB	3/96	4/128	128
R42 4-Way processor	256MB/2GB	3/96	4/128	128
R52 5-Way processor	256MB/2GB	3/96	4/128	128
R72 7-Way processor	256MB/2GB	3/96	4/128	128
9672 R3 Model	**Memory Min/Max**	**Parallel Channels Min/Max (1)**	**ESCON Channels Min/Max (2)**	**Total Channels**
R53 5-Way processor	512MB/4GB	3/96	4/128	128
R63 6-Way processor	512MB/4GB	3/96	4/192	192
R73 7-Way processor	512MB/4GB	3/96	4/192	192
R83 8-Way processor	512MB/4GB	3/96	4/192	192
RX3 10-Way processor	512MB/4GB	3/96	4/192	192

Note:
(1) Parallel Channel Increments by 3
(2) ESCON Channel Increments by 4

9674 C02 and C03 Models	Minimum	Maximum	Increments
Coupling Links	2	32	1

Figure 3.5. 9672 R2 and R3 Memory and channel options.

http://users.etown.edu/w/wunderjt/home_IBM.html

DYNAMIC INTERCONNECT ARCHITECTURES
MULTISTAGE INTERCONNECTION NETWORK

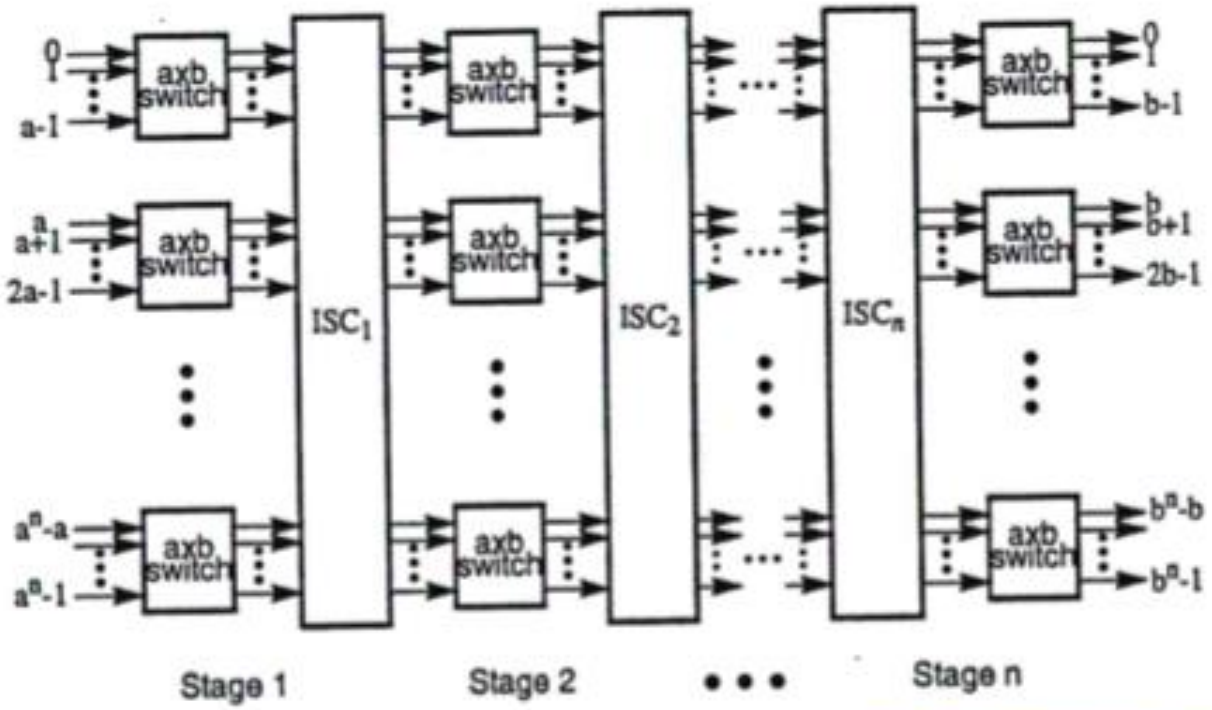


Figure 2.23 A generalized structure of a multistage interconnection network (MIN) built with a × b switch modules and interstage connection patterns $ISC_1, ISC_2, ..., ISC_n$.

K. Hwang, " Advanced Computer Architecture: Parallelism, Scalability, Programmability," Boston, MA: McGraw-Hill, 1992.

DYNAMIC INTERCONNECT ARCHITECTURES
OMEGA NETWORK

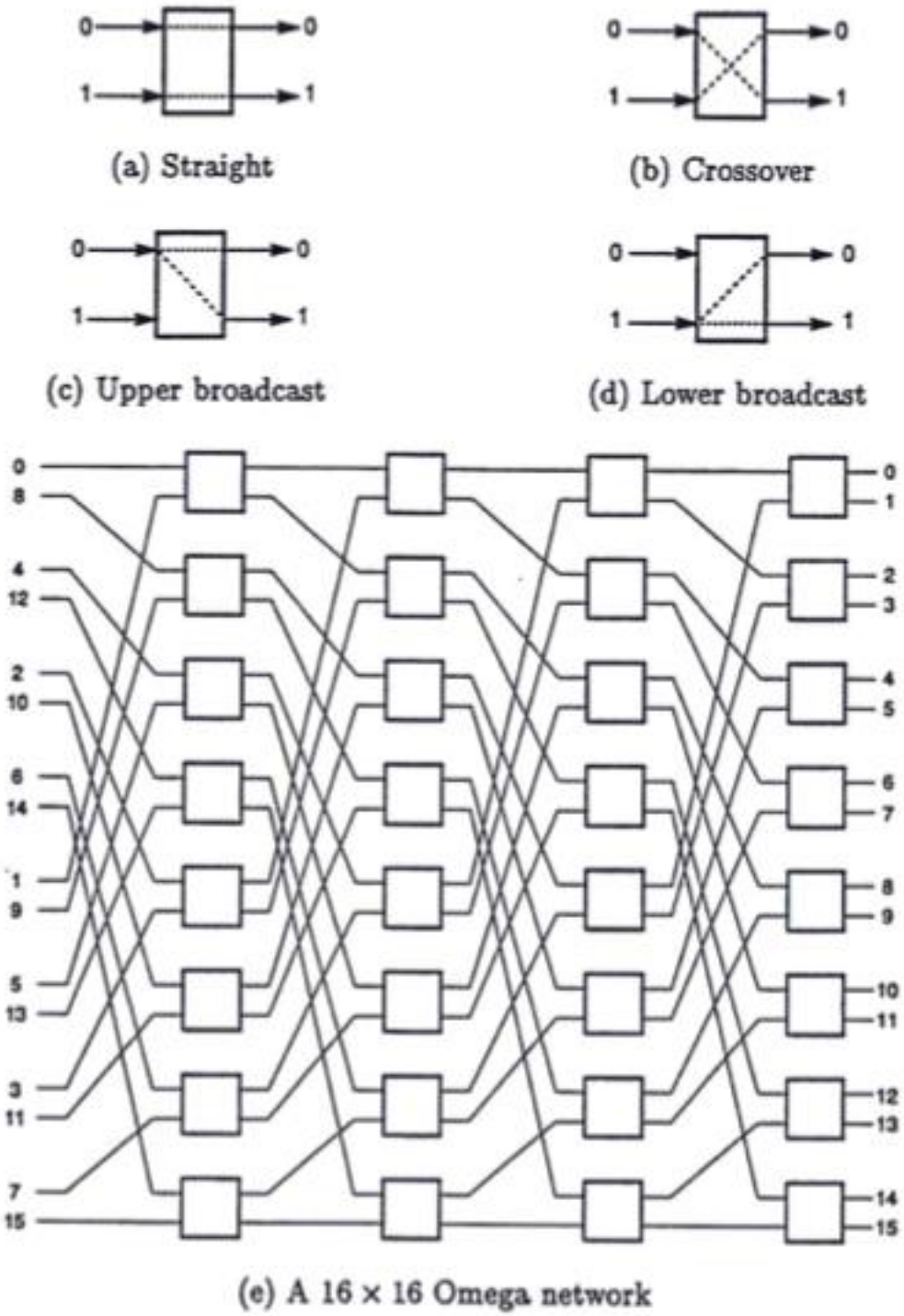
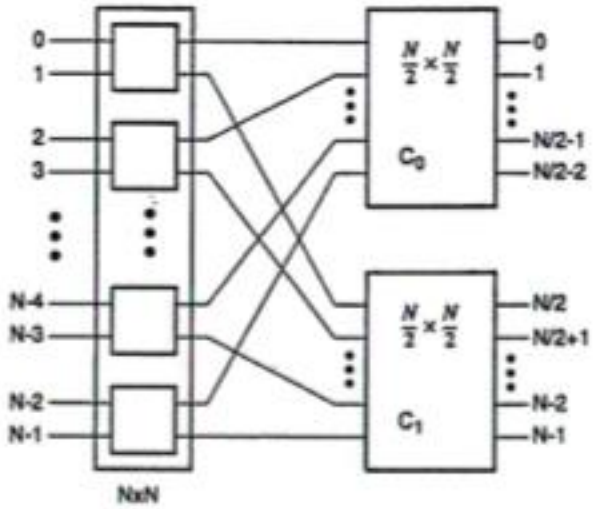


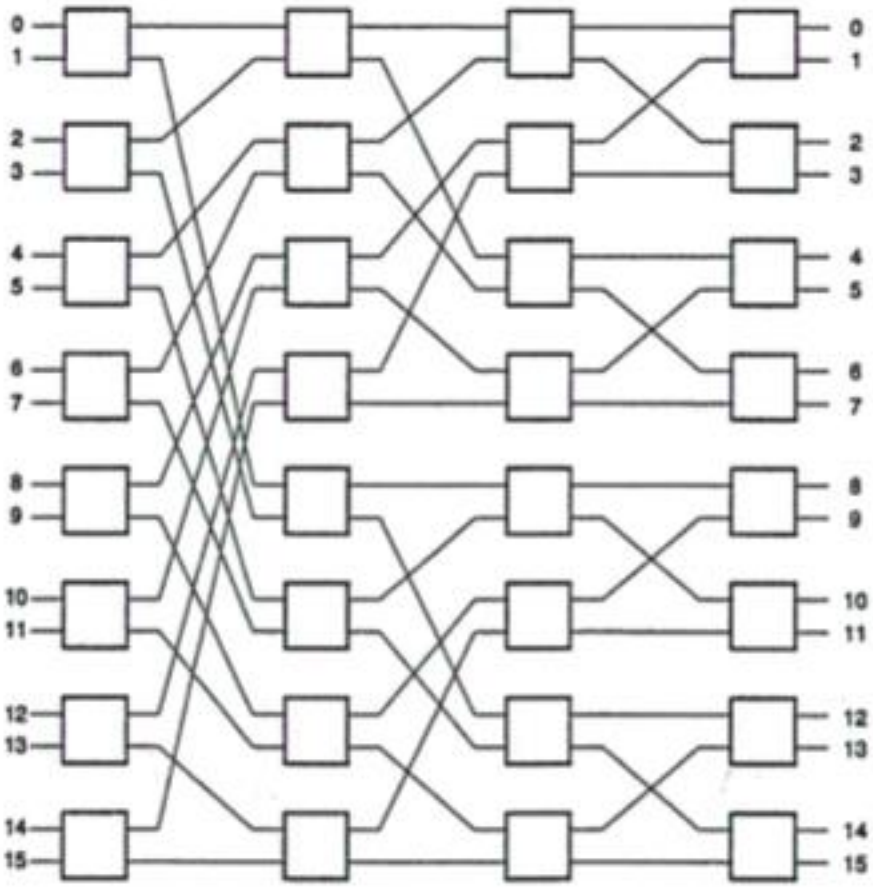
Figure 2.24 The use of 2×2 switches and perfect shuffle as an interstage connection pattern to construct a 16 × 16 Omega network. (Courtesy of Duncan Lawrie; reprinted with permission from *IEEE Trans. Computers*, Dec. 1975)

K. Hwang, " Advanced Computer Architecture: Parallelism, Scalability, Programmability," Boston, MA: McGraw-Hill, 1992.

DYNAMIC INTERCONNECT ARCHITECTURES
BASELINE NETWORK



(a) Recursive construction



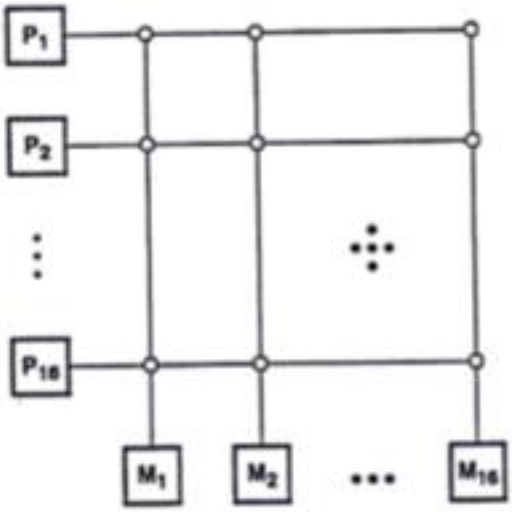
(b) A 16 × 16 Baseline network

Figure 2.25 Recursive construction of a Baseline network. (Courtesy of Wu and Feng; reprinted with permission from *IEEE Trans. Computers, August 1980*)

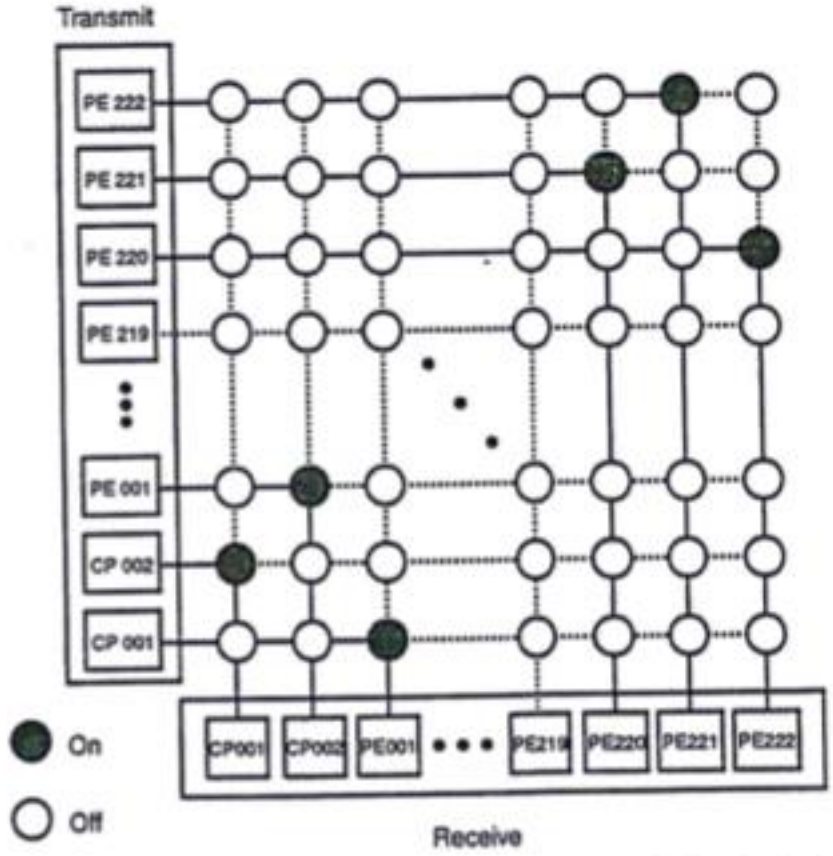
K. Hwang, " Advanced Computer Architecture: Parallelism, Scalability, Programmability," Boston, MA: McGraw-Hill, 1992.

DYNAMIC INTERCONNECT ARCHITECTURES

CROSSBAR SWITCH



(a) Interprocessor-memory crossbar network built in the C.mmp multiprocessor at Carnegie-Mellon University (1972)



(b) The interprocessor crossbar network built in the Fujitsu VPP500 vector parallel processor (1992)

Figure 2.26 Two crossbar switch network configurations.

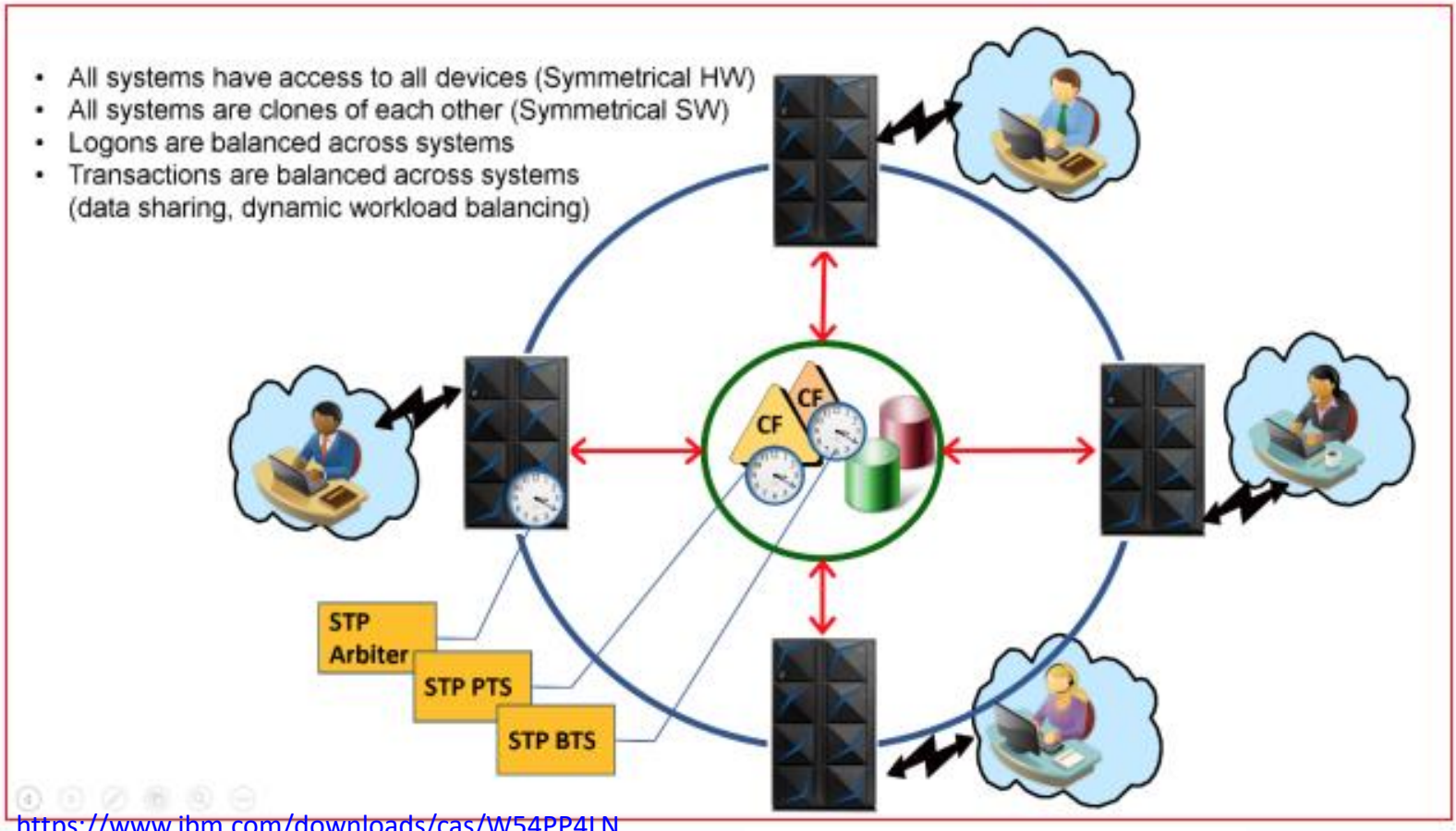
K. Hwang, " Advanced Computer Architecture: Parallelism, Scalability, Programmability," Boston, MA: McGraw-Hill, 1992.

DYNAMIC INTERCONNECT ARCHITECTURES

CROSSBAR SWITCH

IBM PARALLEL SYSPLEX 1990's to 2020's

Example of a four-way Parallel Sysplex:



https://www.ibm.com/downloads/cas/W54PP4LN

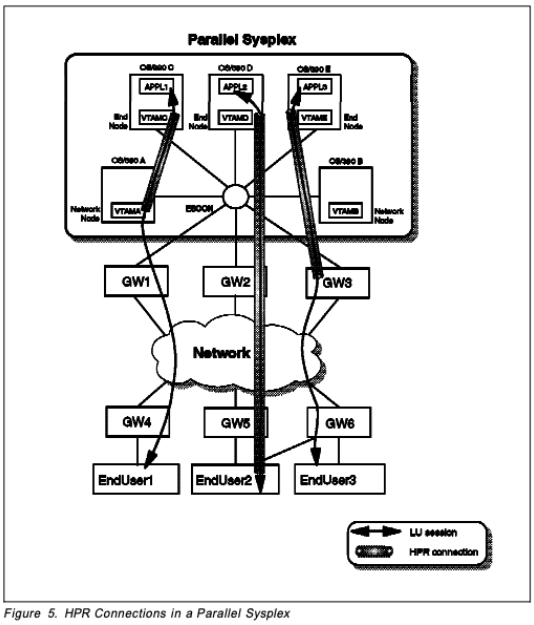


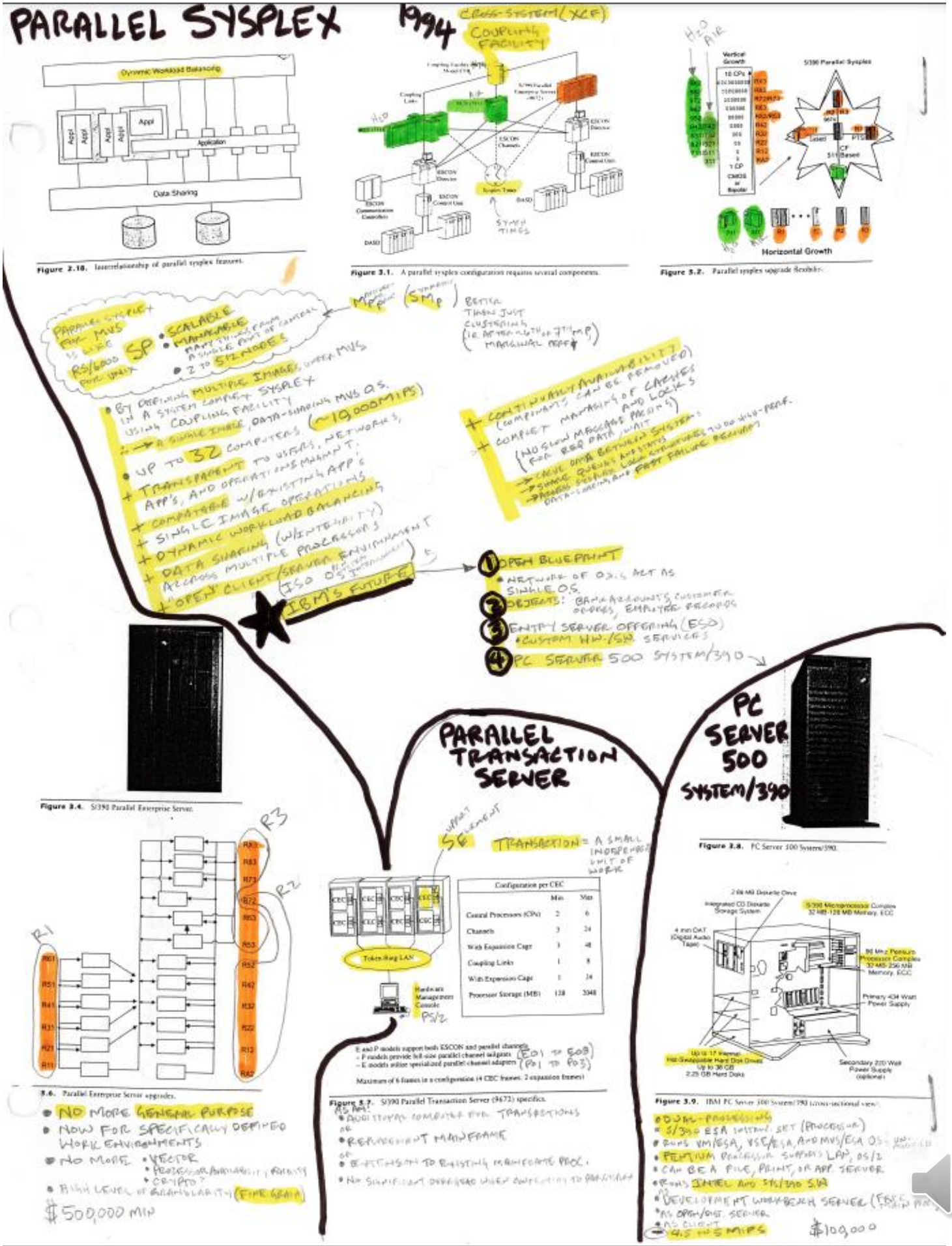
Figure 5. HPR Connections in a Parallel Sysplex

https://www.redbooks.ibm.com/redbooks/pdfs/sg245639.pdf

DYNAMIC INTERCONNECT ARCHITECTURES

CROSSBAR SWITCH

IBM PARALLELL SYSPLEX 1990's to 2020's



DYNAMIC INTERCONNECT ARCHITECTURES ("TOPOLOGIES")

SCALABILITY

Table 2.4 Summary of Dynamic Network Characteristics

Network Characteristics	Bus System	Multistage Network	Crossbar Switch
Minimum latency for unit data transfer	Constant	$O(\log_k n)$	Constant
Bandwidth per processor	$O(w/n)$ to $O(w)$	$O(w)$ to $O(nw)$	$O(w)$ to $O(nw)$
Wiring Complexity	$O(w)$	$O(nw \log_k n)$	$O(n^2 w)$
Switching Complexity	$O(n)$	$O(n \log_k n)$	$O(n^2)$
Connectivity and routing capability	Only one to one at a time.	Some permutations and broadcast, if network unblocked	All permutations, one at a time.
Representative computers	Symmetry S-1, Encore Multimax	BBN TC-2000, IBM RP3	Cray Y-MP/816, Fujitsu VPP500
Remarks	Assume n processors on the bus; bus width is w bits.	$n \times n$ MIN using $k \times k$ switches with line width of w bits.	Assume $n \times n$ crossbar with line width of w bits.

IBM S/390, Z--Series

IBM Parallel Sysplex

K. Hwang, " Advanced Computer Architecture: Parallelism, Scalability, Programmability," Boston, MA: McGraw-Hill, 1992.