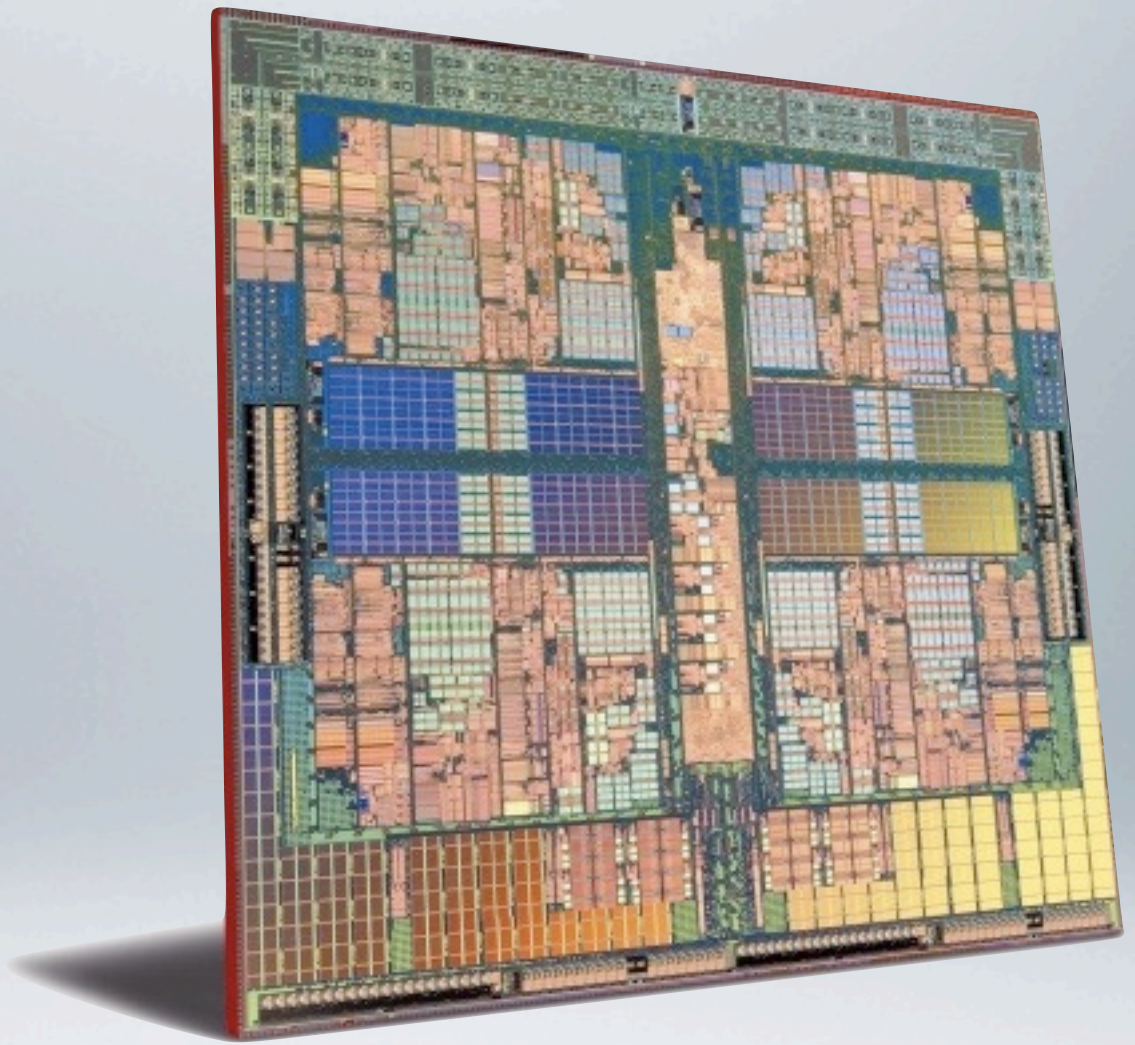


GUC

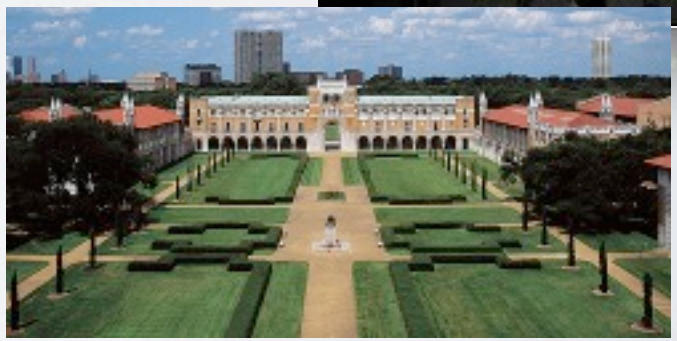
CSEN 601: COMPUTER SYSTEM ARCHITECTURE

Lecture 1

Dr. Cherif Salama

THE LECTURER

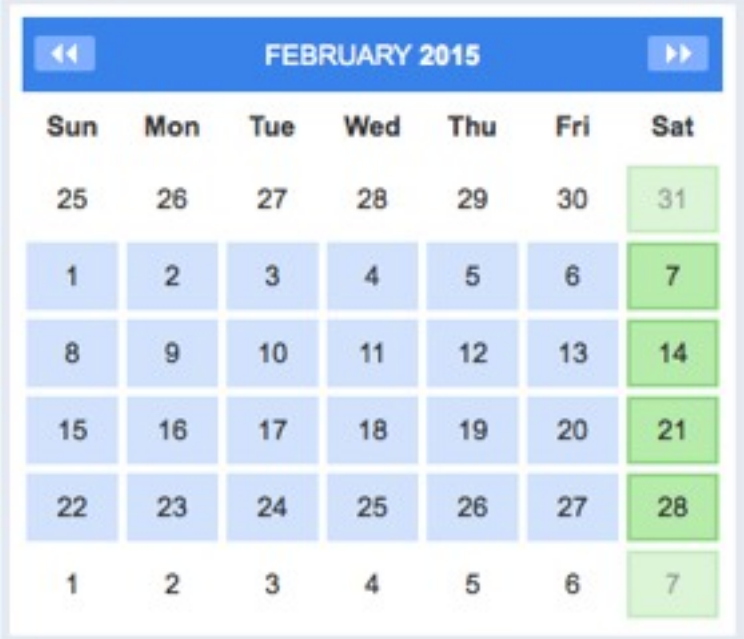
- B.Sc. and M.Sc. from **Ain Shams University**, Egypt
- Ph.D. from **Rice University**, TX, USA
- Collaborated with **Intel** Strategic CAD Labs, OR, USA
- Worked in **IBM** Austin Research Labs, TX, USA
- Lectured at the **EELU** and the **MIU**
- Currently lecturer at the **GUC** and **ASU**
- Other **Research Interests**
  - CAD Tools
  - GPU Computing
  - Programming Languages
  - Multistage Programming
  - Artificial Intelligence (game playing)



CONTACT INFO

- My Contacts:
  - cherif.salama@guc.edu.eg, C7-310
- Office hours:
  - After lectures or by email appointment
- TAs:
  - Amal Rizkallah (amal.rizkalla@guc.edu.eg, C6-206)
  - Lydia Mamdouh (lydia.mamdouh@guc.edu.eg, C7.301)
  - Jailan Salah (jailan.salah@guc.edu.eg, C7.301)

COURSE INFO



- **Lectures**: Saturdays
  - For CSEN: 2nd slot in H18
  - For NETW+DMET: 3rd slot in H8
  - 12 lectures in total
  - Attendance is **critically important**
  - **Plagiarism = Cheating = ZERO**
- **Tutorials**: Starting Saturday, Feb 14th
- **Labs**: Starting Saturday, Feb 14th

COURSE INFO

- Textbook and reference:
  - David A. Patterson and John L. Hennessy, Computer Organization and Design: The Hardware/Software Interface, 5th Edition, 2013
- Course Prerequisites:
  - CSEN 402: Computer Organization and System Programming
- Assessment:
  - Quizzes: 10%
  - Project+Report: 15+10=25%
  - Midterm: 25%
  - Final: 40%

WHY STUDY COMPUTER SYSTEM ARCHITECTURE ?

- Computers are everywhere
- Get a deeper understanding of
  - Computers inner workings
  - Factors affecting computer performance & performance metrics
- Become a better programmer
- Microprocessors architecture is simply beautiful!
- You are an engineering student

MOORE'S LAW

50 years ago (in 1965) Gordon Moore Intel co-founder observed that the number of transistors on a chip was roughly doubling every 2 years and predicted that this trend was going to continue for the 10 following years. His prediction now known as Moore's law is still valid nowadays and is expected to continue till 2025.

On the road to a billion transistors per chip, Intel has developed transistors so small that about 200 million of them could fit on the head of each of these pins.

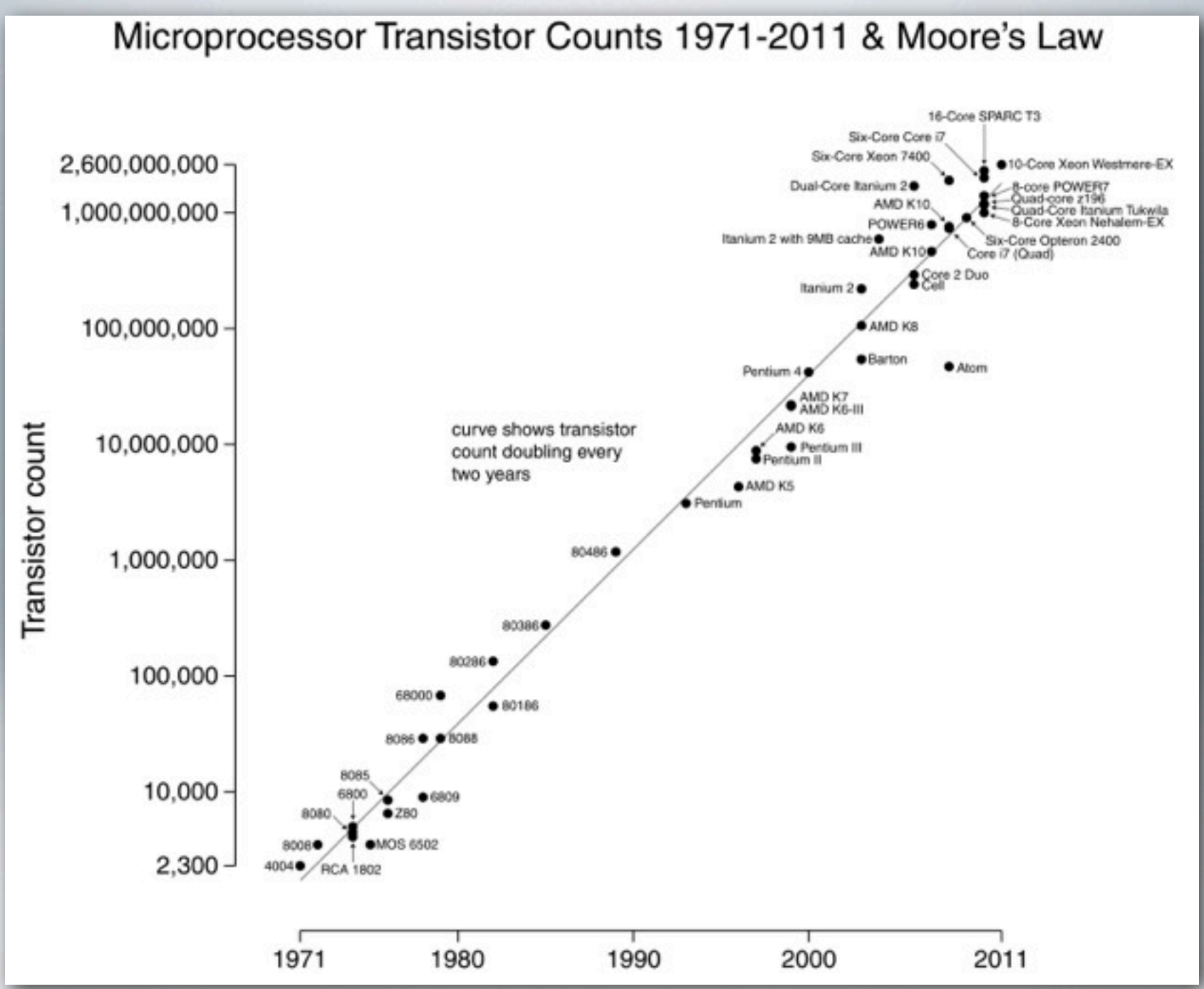
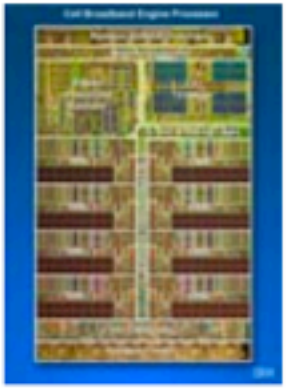


Figure from http://en.wikipedia.org/wiki/Moore's_law

Gordon Moore estimated in 2003 that the number of transistors shipped in a year had reached about 10,000,000,000,000,000,000 (10^{18}). That's about 100 times the number of ants estimated to be in the world.

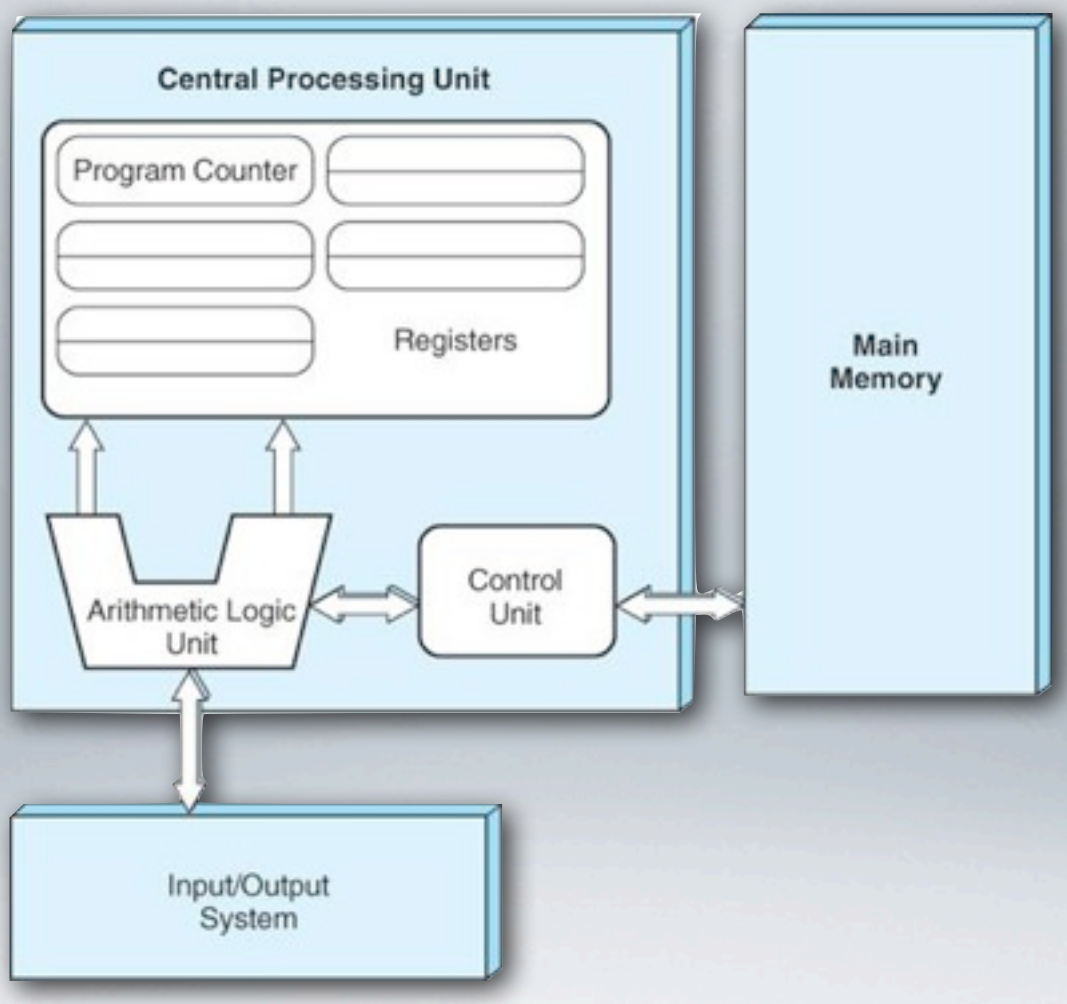
1965	1977	1998	2009
			
IBM System 360/50 0.15 MIPS 64 KB $1M $6.6M per MIPS	DEC VAX11/780 1 MIPS 1 MB $200K $200K per MIPS	Dell Dimension XPS-300 725 MIPS 64 MB $2412 (1/4/98) $3.33 per MIPS	PS3 & GPUs $500 $0.01 per MIPS

SO WHAT?

- Making use of the large number of available transistors and the impressive technological advances is crucial
- This course will show you how to do that in the context of a commercial computer
  - MIPS ISA and programming (Chapter 2)
  - Simple MIPS implementation (Chapter 4)
  - Pipelined MIPS implementation (Chapter 4)
  - Parallel Processors (Chapter 6)

A QUICK REVIEW

- Von-Neumann model
- Key Concepts of abstraction
- The Basic Computer



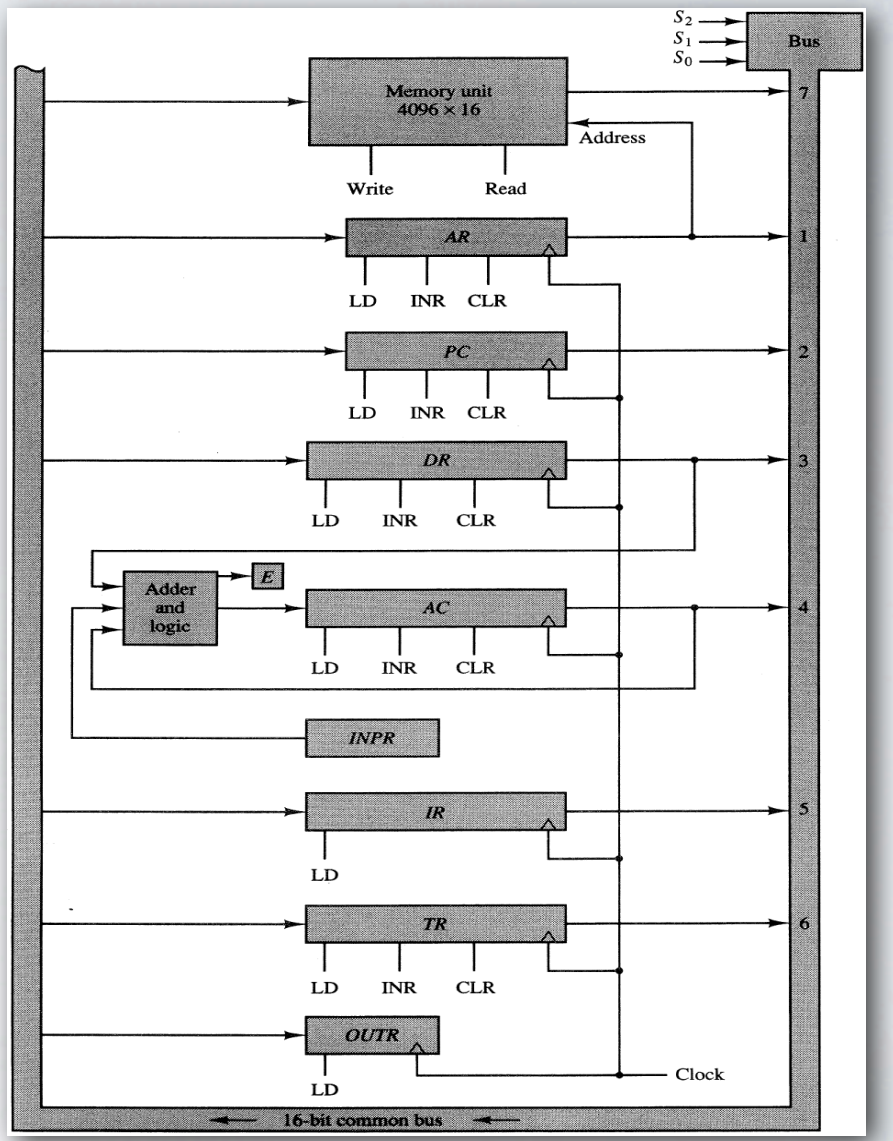
VON-NEUMANN MODEL OF A COMPUTER

KEY CONCEPTS OF ABSTRACTION

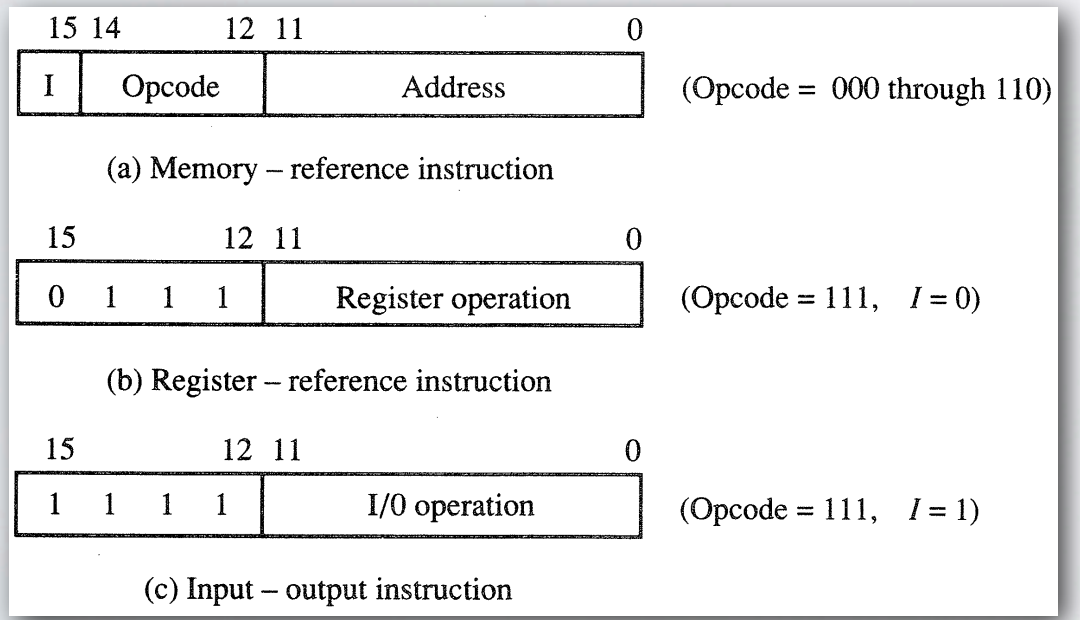
- Instruction Set Architecture (ISA)
  - Functional interface to the assembly level programmer
  - Ex: Intel (x86)
- Implementation (Machine Organization)
  - Mechanism that interprets and executes the instructions
  - Ex: Intel Core i7 Processor
- Realization
  - Physical fabrication (depends on technology)
  - Ex: Intel Core i7 Processor using 32 nm

THE BASIC COMPUTER

Common bus



Instruction Format

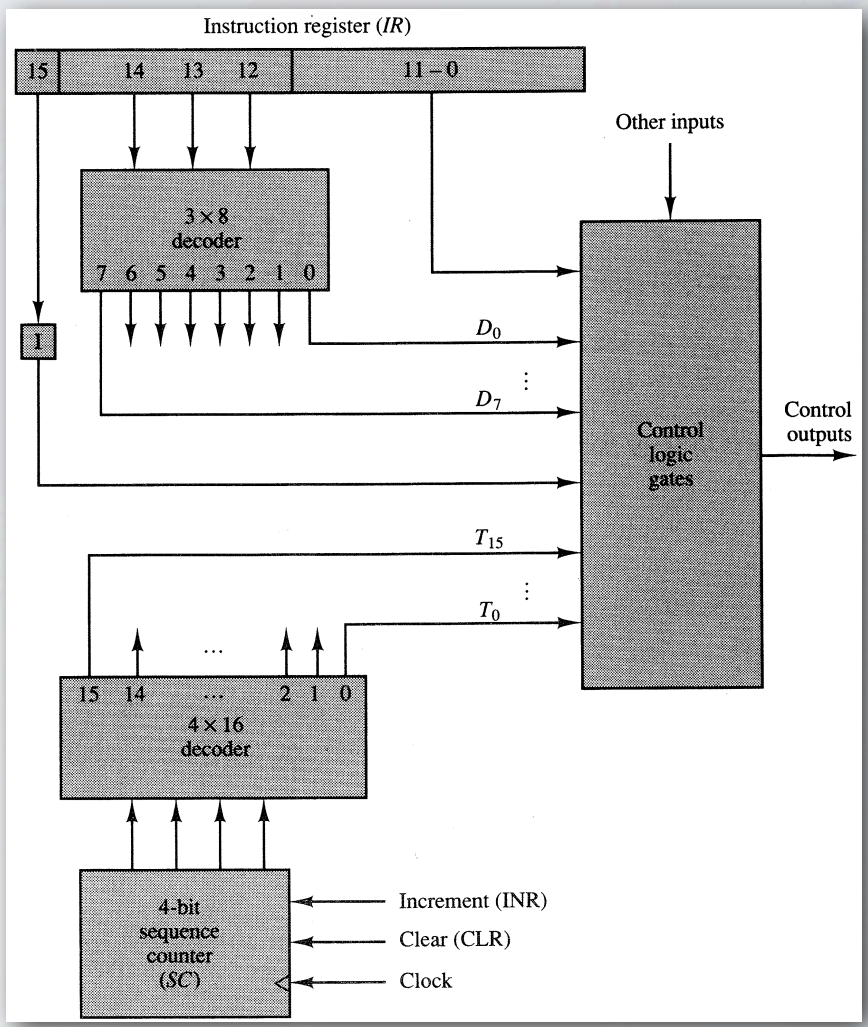


THE BASIC COMPUTER

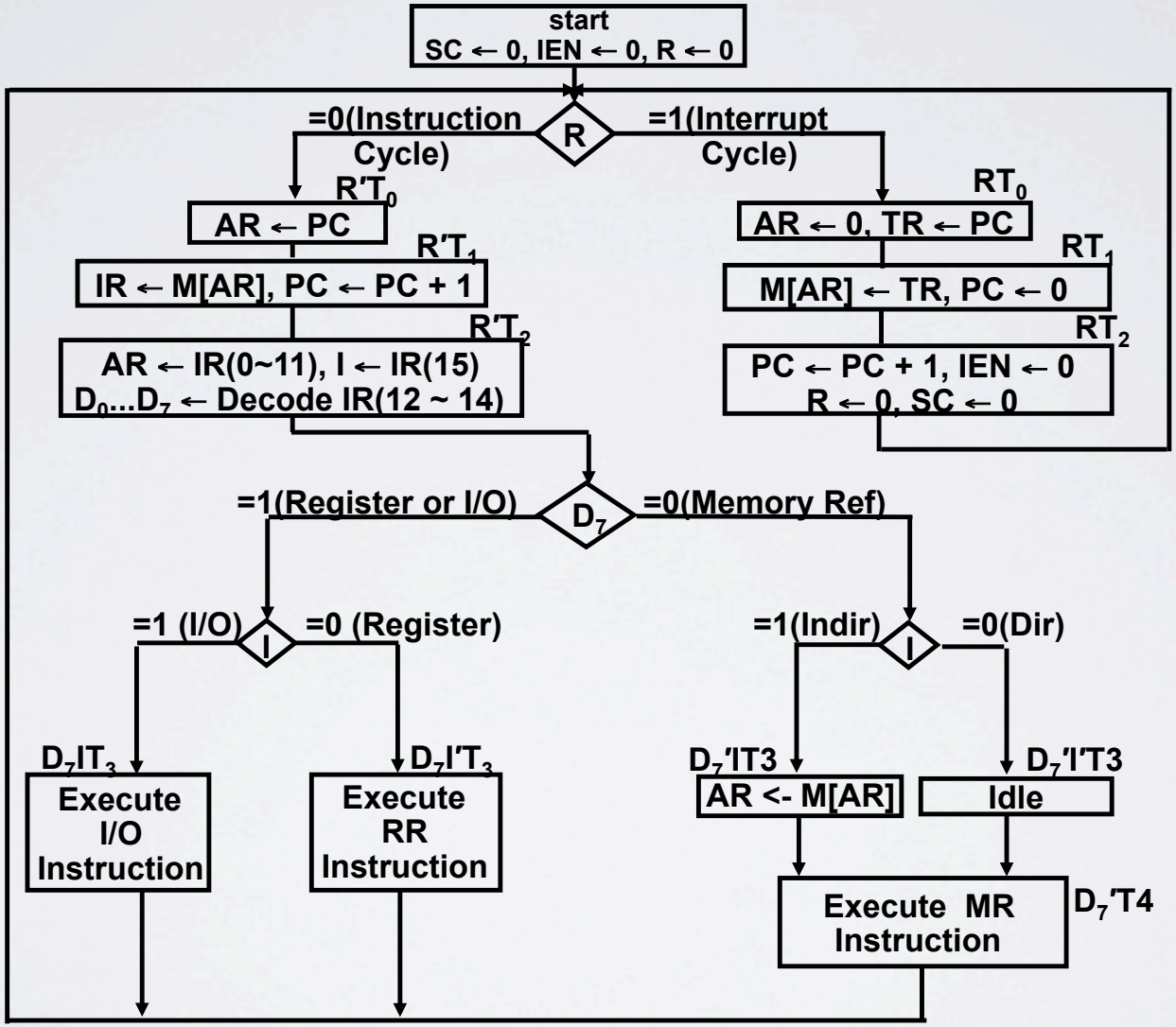
Instructions

Symbol	Hexadecimal code $I = 0$	Hexadecimal code $I = 1$	Description
AND	0xxx	8xxx	AND memory word to *AC*
ADD	1xxx	9xxx	Add memory word to *AC*
LDA	2xxx	Axxx	Load memory word to *AC*
STA	3xxx	Bxxx	Store content of *AC* in memory
BUN	4xxx	Cxxx	Branch unconditionally
BSA	5xxx	Dxxx	Branch and save return address
ISZ	6xxx	Exxx	Increment and skip if zero
CLA	7800		Clear *AC*
CLE	7400		Clear *E*
CMA	7200		Complement *AC*
CME	7100		Complement *E*
CIR	7080		Circulate right *AC* and *E*
CIL	7040		Circulate left *AC* and *E*
INC	7020		Increment *AC*
SPA	7010		Skip next instruction if *AC* positive
SNA	7008		Skip next instruction if *AC* negative
SZA	7004		Skip next instruction if *AC* zero
SZE	7002		Skip next instruction if *E* is 0
HLT	7001		Halt computer
INP	F800		Input character to *AC*
OUT	F400		Output character from *AC*
SKI	F200		Skip on input flag
SKO	F100		Skip on output flag
ION	F080		Interrupt on
IOF	F040		Interrupt off

Control Unit



THE BASIC COMPUTER



COMPLETE BC MICROOPS 1/2

Fetch	$R'T_0$:	AR ← PC
	$R'T_1$:	IR ← M[AR], PC ← PC + 1
Decode	$R'T_2$:	D0, ..., D7 ← Decode IR(12 ~ 14), AR ← IR(0 ~ 11), I ← IR(15)
Indirect	$D_7'IT_3$:	AR ← M[AR]
Interrupt		
	$T_0'T_1'T_2'(IEN)(FGI + FGO)$:	R ← 1
	RT_0:	AR ← 0, TR ← PC
	RT_1:	M[AR] ← TR, PC ← 0
	RT_2:	PC ← PC + 1, IEN ← 0, R ← 0, SC ← 0
Memory-Reference		
AND	D_0T_4:	DR ← M[AR]
	D_0T_5:	AC ← AC ∧ DR, SC ← 0
ADD	D_1T_4:	DR ← M[AR]
	D_1T_5:	AC ← AC + DR, E ← C_{out}, SC ← 0
LDA	D_2T_4:	DR ← M[AR]
	D_2T_5:	AC ← DR, SC ← 0
STA	D_3T_4:	M[AR] ← AC, SC ← 0
BUN	D_4T_4:	PC ← AR, SC ← 0
BSA	D_5T_4:	M[AR] ← PC, AR ← AR + 1
	D_5T_5:	PC ← AR, SC ← 0
ISZ	D_6T_4:	DR ← M[AR]
	D_6T_5:	DR ← DR + 1
	D_6T_6:	M[AR] ← DR, if(DR=0) then (PC ← PC + 1), SC ← 0

COMPLETE BC MICROOPS 2/2

Register-Reference		
	$D_7I'T_3 = r$	(Common to all register-reference instr)
	$IR(i) = B_i$	(i = 0,1,2, ..., 11)
	r:	SC ← 0
CLA	rB_{11}:	AC ← 0
CLE	rB_{10}:	E ← 0
CMA	rB_9:	AC ← AC′
CME	rB_8:	E ← E′
CIR	rB_7:	AC ← shr AC, AC(15) ← E, E ← AC(0)
CIL	rB_6:	AC ← shl AC, AC(0) ← E, E ← AC(15)
INC	rB_5:	AC ← AC + 1
SPA	rB_4:	If(AC(15) =0) then (PC ← PC + 1)
SNA	rB_3:	If(AC(15) =1) then (PC ← PC + 1)
SZA	rB_2:	If(AC = 0) then (PC ← PC + 1)
SZE	rB_1:	If(E=0) then (PC ← PC + 1)
HLT	rB_0:	S ← 0
Input-Output	$D_7IT_3 = p$	(Common to all input-output instructions)
	$IR(i) = B_i$	(i = 6,7,8,9,10,11)
	p:	SC ← 0
INP	pB_{11}:	AC(0-7) ← INPR, FGI ← 0
OUT	pB_{10}:	OUTR ← AC(0-7), FGO ← 0
SKI	pB_9:	If(FGI=1) then (PC ← PC + 1)
SKO	pB_8:	If(FGO=1) then (PC ← PC + 1)
ION	pB_7:	IEN ← 1
IOF	pB_6:	IEN ← 0

REFERENCES

- Mano, Computer System Architecture, Third Edition, 1992
- Microprocessor Architecture: From simple pipelines to chip multiprocessors, Jean-Loup Baer, 2010
- Computer Organization and Design: The Hardware/Software Interface, Fifth Edition, Patterson and Hennessy, 2013