

# *ECE390*

## *Computer Engineering II*

### *Lecture 1*

---

Dr. Zbigniew Kalbarczyk  
University of Illinois at Urbana- Champaign

### *Your Instructor*

---

Dr. Zbigniew Kalbarczyk

[kalbar@crhc.uiuc.edu](mailto:kalbar@crhc.uiuc.edu)

(217) 244-7110

Center for Reliable and High-Performance Computing

Coordinate Science Laboratory (CLS)

1308 W. Main, Urbana 61801

**Office:** 267

**Office Hours:** 1:00 – 3.00 pm Thursday and by appointment

## *Your TAs*



Ryan Rivera:        rrivera@uiuc.edu

Marc Enevold:      enevold@uiuc.edu

Brandon Swamy:    bswamy2@uiuc.edu

.

## *ECE390 Web Site*



[www.ece.uiuc.edu/ece390](http://www.ece.uiuc.edu/ece390)

- Everything you need to know about ECE390
- Homework is distributed, completed, and graded online.
- MP Handouts are online.
- Your textbook is online.
- Access all of these from the 390 web site.

## *ECE291 Lab*

---

- Location: 238 Everitt Lab
- Hours: 24 Hour Access
  
- Network Accounts
- Go to <http://accounts.ad.uiuc.edu> and follow the instructions.

## *Evaluation*

---

Your grade will be based on:

- |                         |               |
|-------------------------|---------------|
| • 7 Homeworks           | 140 (max 100) |
| • 4(5) Machine Problems | 290           |
| • 1 Final Project       | 110           |
| • 2 Exams               | 280           |
| • 1 Final Exam          | 220           |

## Miscellaneous

- The lab notes may be purchased in the IEEE bookstore in the basement for approximately \$18.
- They have been extensively revised and should be very useful.
- You may obtain PDF or HTML versions from the 390 Web Site.

Z. Kalbarczyk

ECE390

## Course Goals I

<b>ECE290</b>	<b>ECE390 (ECE291)</b> (Everything in between)	<b>ECE411 (ECE312)</b>
Binary numbers digital logic, state machines	Machine-level operations, computer organization, data movement	Computer Organization and Design

ECE390 bridges the gap between your logic classes and programming courses through assembly-level programming of a real (80x86) computer

Z. Kalbarczyk

ECE390

## *Course Goals II*



- You will become proficient in assembly-level programming
- You will learn to organize large programs.
- You will learn how to interface to hardware.

Z. Kalbarczyk

ECE390

## *Brief Historical Perspective on Computers*



## *Pioneers - Blaise Pascal (1623-1662)*

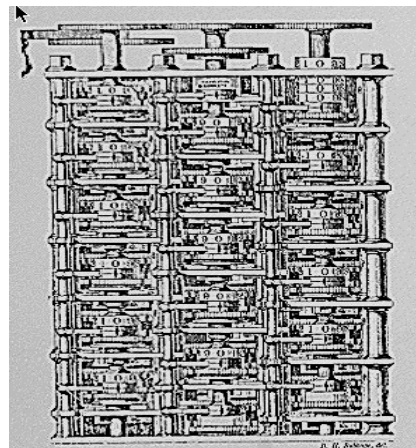
### 1642

- French mathematician who invented the first operational calculating machine
- Arithmetic Machine - introduced 1642
  - addition and subtraction
  - subtraction done using complementary techniques (similar techniques are used in modern computers)
  - multiplication and division implemented by performing a series of additions and subtractions
- The basic principle of this calculator is still used today in water meters and modern-day odometers

## *Pioneers - Charles Babbage (1791-1871)*

### 1822

- British mathematician who invented the first device that might be considered as a computer in the modern sense of the word
  - Difference Engine (1822) - partially build
  - Analytical Engine (1830) - never build
- Difference Engine was eventually constructed from original drawings by a team at London's Science Museum
  - 4000 components,
  - weight 3 tons, 10 feet wide, and 6.5 feet long
  - the device performed its first sequence of calculations in the early 1990's and return results to 31 digits of accuracy



## Claude Shannon

---

### 1938

- C. Shannon Master Thesis (*possibly the most important master's thesis of the twentieth century*) (1938)
  - demonstrated how Boolean concepts of TRUE and FALSE can be used to represent the function of switches in electronic circuits

## John Von Neumann

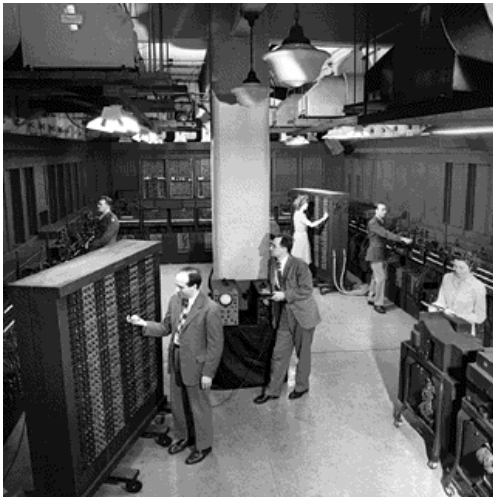
---

### 1945

- Mathematician who introduced basic elements of the stored-program computer
  - **A memory** containing both data and instructions
  - **A calculating unit** capable of performing both arithmetic and logical operations on the data
  - **A control unit**, which could interpret an instruction retrieved from the memory and select alternative courses of action based on the results of previous operations

# *First Electronic Computer*

## 1940's – 1950's



- Vacuum tubes & mechanical relays: UNIVAC, ENIAC
- 30 tons
- 150 Kwatts
- 80 bytes of memory
- Key problem was reliability
  - About 50 tubes had to be replaced per day

Z. Kalbarczyk

ECE390

# *First Electronic Computer (cont.)*



## 1940's – 1950's

- ILLIAC
  - built at the University of Illinois was the first computer owned by an academic department
  - The String Quartet #4 "*the Illiac Suite*" - first music composed by a computer (1957)

Z. Kalbarczyk

ECE390

## *The First Transistor*

### 1948

- Properties of semiconductors were not well understood until the 1950s
- Bell Laboratories began research into semiconductors in 1945
- William Shockley, Walter Brattain, and Jhon Bardeen succeeded in creating the *first point-contact germanium transistor* on the 23rd December 1947
  - they took a break for Christmas before publishing the achievement that is why the reference books state that *the first transistor was created 1948*
- Bipolar junction transistor (Shockley) - 1950
- Field effect transistor (MOS FET) - 1962

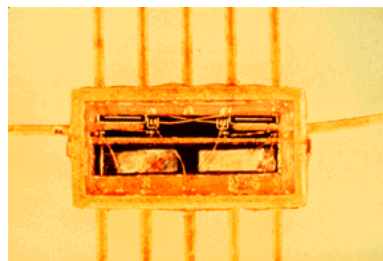
Z. Kalbarczyk

ECE390

## *The First Integrated Circuit*

### 1958

- Jack Kilby (Texas Instruments) in 1958 succeeded in fabricating multiple components on a single piece of semiconductors
  - a phase shift oscillator
- 1961 Fairchild and Texas Instruments fabricate first commercial integrated circuit comprising simple logic functions
  - two logic gates (four bipolar transistors and four resistors)



Z. Kalbarczyk

ECE390

# *IBM Mainframes*

---

## **Late 1960**

- Powerful, centralized CPUs with terminals
- Age of “big iron”

# *Towards First Personal Computer*

---

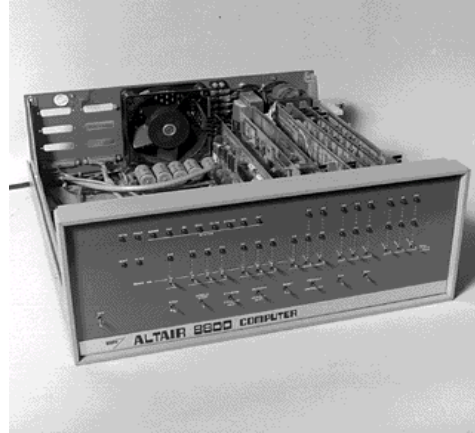
## **1974**

- In 1970 Japanese calculator company Busicom requested from Intel a set of twelve IC for use in a new calculator
  - T. Hoff, a designer at Intel, inspired by the request created the first microprocessor, the 4004
  - 2300 transistors; 60 000 operation per second
- First general-purpose microprocessor, the 8080, introduced by Intel in 1974
  - 8-bit device, 4500 transistors, 200000 operations per second

# Personal Computer

## 1974

- Altair 8800 (1974)
  - based on 8080 microprocessor,
  - affordable price of \$375
  - no keyboard, no screen, no storage,
  - 4k memory, programmable by means of a switch panel
- Bill Gates and Paul Allen founded Microsoft (1975)
  - BASIC 2.0 on the Altair 8800
  - first high-level language available on a home computer



Z. Kalbarczyk

ECE390

## Personal Computer (cont)

- S. Wozniak and S. Jobs create Apple
  - 16k ROM, 4k of RAM, a keyboard, and color display
- TRS-80 (Z80-based system) from Radio Shack - 1977
  - 4k ROM, 4k RAM, keyboard, and cassette type drive
  - price \$600
- Personal Computer from IBM - 1981
  - 16-bit microprocessor 8088, ROM BASIC, cassette interface 360k floppy, DOS 1.0
  - price \$1365

Z. Kalbarczyk

ECE390

## *Personal Computer (cont)*

---

- 1983 IBM XT gets hard disk (10Mb hard disk costs \$3000)
- 1985 Intel Introduces 80386
  - first 32-bit 80x86 family
- 1986 Compaq introduces first 80386-based system
- 1989 Intel introduces 80486, includes math co-processor
- 1992 Intel Pentium (64-bit) memory bus,
  - AMD, Cyrix 486 compatible processors
- 1999 AMD Athlon (650 MHz)
  - Cyrix M II - 300MHz to MII -433MHz
  - .....

Z. Kalbarczyk

ECE390

*Material Review*  
*(from previous classes)*

---

## Number Systems

- Base 10 representation (decimal) (0..9):

$$d_n * 10^n + d_{n-1} * 10^{n-1} + \dots + d_2 * 10^2 + d_1 * 10^1 + d_0 * 10^0$$

$$\text{Eg: } 3045 = 3 * 10^3 + 4 * 10^1 + 5 * 10^0$$

$$= 3000 + 40 + 5 = 3045$$

- 

- Base 2 representation (binary) (0..1):

$$d_n * 2^n + d_{n-1} * 2^{n-1} + \dots + d_2 * 2^2 + d_1 * 2^1 + d_0 * 2^0$$

$$\text{Eg: } 101101 = 1 * 2^5 + 1 * 2^3 + 1 * 2^2 + 1 * 2^0$$

$$= 32 + 8 + 4 + 1 = 45$$

## Number Systems

- Base 16 representation (hex) (0..9,A..F):

$$d_n * 16^n + d_{n-1} * 16^{n-1} + \dots + d_2 * 16^2 + d_1 * 16^1 + d_0 * 16^0$$

$$\text{Eg., } 3AF = 3 * 16^2 + 10 * 16^1 + 15 * 16^0 = 3 * 256 + 10 * 16 + 15 = 943$$

## Base Conversion

Example: Convert 45 to binary

45 divides by 32 ( $2^5$ ) once, leaves 13

13 divides by 8 ( $2^3$ ) once, leaves 5

5 divides by 4 ( $2^2$ ) once, leaves 1

1 divides by 1 ( $2^0$ ) once, leaves nothing [done]

Thus: 45 (base 10) == 101101 (base 2)

## Signed & Unsigned Numbers

- We need a way to represent negative numbers
- Simple idea: Use the first bit as a sign bit!
  - $s = 0$ : positive (+)
  - $s = 1$ : negative (-)
- *Problem*: There are TWO zeros 1...0 and 0...0
  - Difficult to process negative numbers (special case)
  - There is a better way to handle negative numbers!
- Solution: Use **Two's complement**
- Numeric Formula:

$$-d_n * 2^n + d_{n-1} * 2^{n-1} + \dots + d_2 * 2^2 + d_1 * 2^1 + d_0 * 2^0$$

- Notice the negative sign in front of  $d_n$

## *Signed & Unsigned Numbers (cont.)*

- To subtract  $A-B$ , perform  $A+(-B)$ .
  - the addition operator works for negative numbers
  - first bit still *represents* the sign of the number.
    - $s=0$ : positive (+)
    - $s=1$ : negative (-)
- Three methods to compute a negative number ( $n$ ) (choose one method)
  - Inverting bits then add 1
  - Take largest number (all ones), subtract  $n$ , add 1
  - Scan  $n$  from right to left, copy zeros, copy 1st one, invert rest

## *Review From Previous Classes* *Signed & Unsigned Numbers - examples*

- Examples (8-bit):
  - $-1 = 1111,1111$
  - $-2 = 1111,1110$
  - $-128 = 1000,0000$
  - $+1 = 0000,0001$
  - $+127 = 0111,1111$
  - $+128 = \text{Invalid}$
- Examples: (16-bit)
  - $-1 = 1111,1111,1111,1111$
  - $-2 = 1111,1111,1111,1110$
  - $-32,768 = 1000,0000,0000,0000$

## Sign and Zero Extension

- Consider the value 64
  - the 8-bit representation is 40h
  - the 16-bit equivalent is 0040h
- Consider the value -64
  - the 8-bit two's complement is C0h
  - the 16-bit equivalent is FFC0h
- The rule: to sign extend a value from some number of bits to a greater number of bits - **copy the sign bit into all the additional bits in the new format**
- Remember - **you must not sign extend unsigned values**

## Sign Contraction

- Given an n-bit number, you cannot always convert it into an m-bit number if  $m < n$
- Consider the value -448
  - the 16-bit representation is FE40h
  - the magnitude of this number is too great to fit into 8-bit value
    - you cannot convert it to eight bits
    - this is an example of an overflow condition that occurs upon conversion
- To properly sign contract one value to another the bits that you wish to remove must all contain either zero or FFh
  - e.g., FF80h *can be signed contracted to* 80h
  - 0100h cannot be sign contracted to 8-bit representation