SYSC3601
Microprocessor Systems

Prof. James Green

Unit 1: Introduction & History

Course Outline

- Goal: To familiarize students with microprocessor-based circuit design.
- The course deals with the applications, organization, architecture, and design of microprocessor systems.
- Topics covered include: addressing, bus structures, memory and I/O interfacing, interrupt mechanisms, and related techniques at the hardware and assembly language levels.

History of the μProc

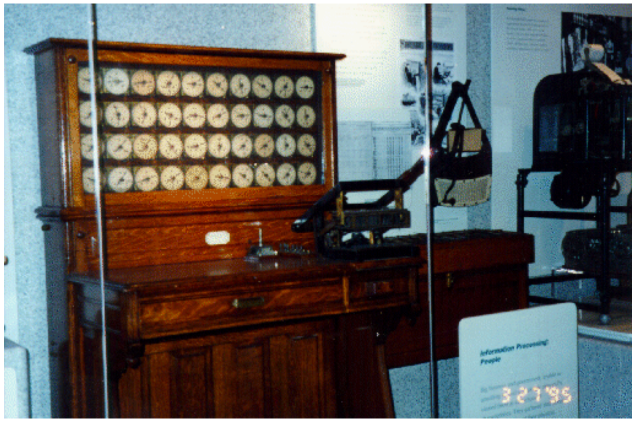
- 400BC: Abacus invented in Babylonia (now Iraq)
- 100BC: Antikythera mechanism, used for registering and predicting the motion of the stars and planets
- 700AD: Arabic numbers introduced (zero, 10's, 100's, etc)
- 1641: John Napier invents logs
- 1623: Wilhelm Schickard builds first mechanical calculator. 6 digits. Prototype only
- 1642: Blaise Pascal builds mechanical calculator. 8 digits, trouble with carries, jams

History of the μProc

- 1820: Charles Babbage conceives of "Difference Engine" to print astronomical tables. Cancelled after 40 years. Analytical machine next (steam powered), but can't build due to manufacturing of 1000s of small cogs, etc.
- 1833: Augusta Ada Byron meets Babbage. Analyses programming potential and outlines fundamentals of computer programming.

History of the μProc

- 1800’s to ~1939: Mechanical machines



Census tabulator used ~1890

History of the μProc

- WWII:
  - Konrad Zuse builds first general purpose programmable calculator (1941).
    - Pioneers use of binary math and Boolean logic in electronic calculation.
    - Relay logic, 5.33 Hz
  - Mechanical encrypting machine ENIGMA by Siemens
  - Turing Colossus code-breaking machine – vacuum tubes. Non-programmable. TOP SECRET. (1943)
  - Electronic Numerical Integrator Analyzor and Computer (ENIAC). Ballistics tables. 1st programmable. (1946)
    - 17,000 vacuum tubes, 1500 relays, 30 tonnes, 140 KW power
    - Programmed via 6000 switches, 100’s jumpers
    - Too late for war effort, but spawned many research programs

History of the μProc

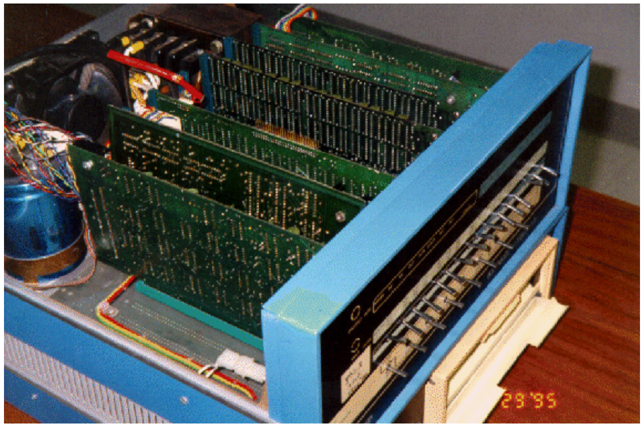
- 1947: Bell Labs invents the transistor
- 1952: Von Neumann creates IAS
  - most modern computers use this design.
  - Program in memory.
- 1959: Texas Instruments and Fairchild announce integrated circuit
- 1960: DEC PDP-1.
  - First minicomputer (50 sold, $120K, first video game!, 512x512 display)
- 1964: IBM 360 – popular business machine
- 1965:
  - An IC that cost $1000 in 1959 now costs $10.
  - Gordon Moore predicts the number of components in an IC will double every 18 months (still true)

History of the μProc

- 1968: Moore and Noyce found Intel
- 1970:
  - Fairchild Semiconductor introduces 256 bit RAM chip
  - Intel introduces 1K RAM & 4004 μP
  - DEC PDP-11. Dominates minicomputers in 1970s.
- 1971
  - Bill Gates and Paul Allen form Traf-O-Data.
  - Steve Jobs & Steve Wozniak building “blue boxes”
- 1972: Intel 8008 – 8 bit version of 4004.
  - First general purpose computer on a chip.

History of the μProc

- 1975:MITS Altair 8800.
  - Hailed as first personal computer.
  - Sold as a kit



History of the μProc

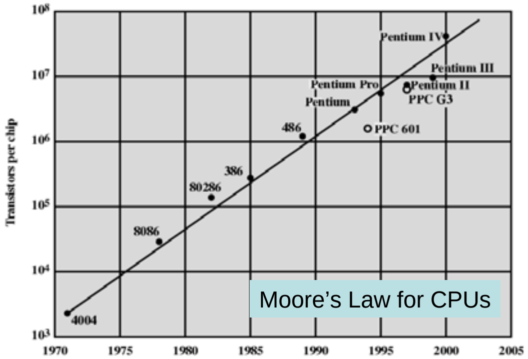
- 1975: Paul Allen & Bill Gates develop BASIC.
  - Microsoft is born.
- 1977: Apple II. 16K RAM. US$1195 (no monitor)
- 1978: DEC VAX – First 32-bit superminicomputer
- 1981: IBM PC. 8088 & COTS. No patent. Sold cct diag books for $49 →clones! Packaged with MS DOS.
- 1984: Apple Macintosh
- 1985: MS Windows 1.0
- 1985: MIPS – First commercial RISC machine
- 1987: Sun SPARC – First SPARC-based RISC workstation
- 1989: MS Sales reach $1 billion per year

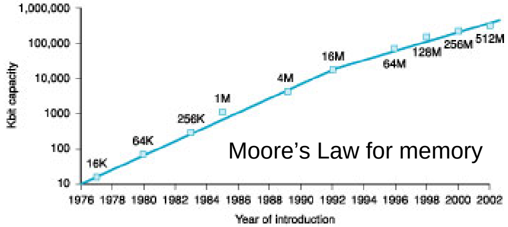
History of x86 μP

- 4004 the first microprocessor (4-bit) 16K RAM
- 8008 (8-bit)
- 8080 (8-bit) 64K RAM, 2Mhz clock
- 8088 (8 bit)
- 8086 (16-bit) 1M RAM, 5MHz clock
- 80286 (16-bit) 16M RAM, 16MHz clock
- 80386, 4G RAM, 33 MHz clock
- 80486, 4G RAM, 66 MHz clock
- Pentium, 4G RAM, 66 MHz clock
- Pentium Pro, 64G RAM, 133 MHz clock
- Pentium II, 64G RAM, 233 MHz clock
- Pentium III, 64G RAM, 500 MHz clock
- Pentium 4, 64G RAM, 1.5 GHz clock

Trends

Moore's Law: an empirical observation that the number of transistors is doubling every 18 months (possibly holds true until 2020).

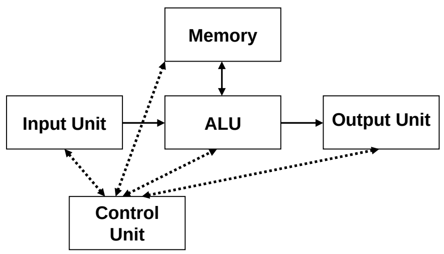




Type	Price ($)	Example application
Disposable computer	1	Greeting cards
Embedded computer	10	Watches, cars, appliances
Game computer	100	Home video games
Personal computer	1K	Desktop or portable computer
Server	10K	Network server
Collection of Workstations	100K	Departmental minisupercomputer
Mainframe	1M	Batch data processing in a bank
Supercomputer	10M	Long range weather prediction

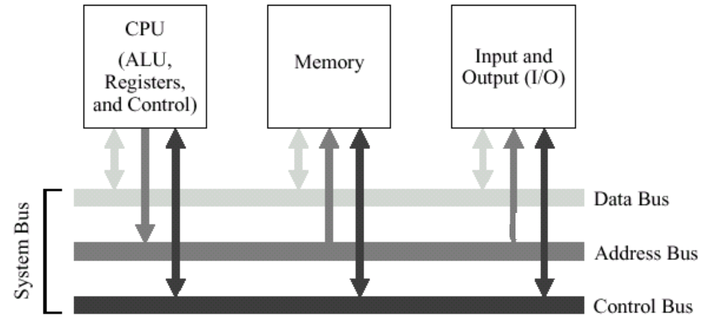
Von Neumann Model

- Consists of **5 major components**:
  - **Arithmetic and Logic Unit (ALU)**: Performs mathematical and logical operations on its operands
  - **Control Unit**: Produces control signals to orchestrate functioning of all other units (the boss!)
  - **Memory Unit**: Holds both data and program (in a stored program computer)
  - **Input Unit**: Obtains data from external sources
  - **Output Unit**: Provides data to external sources

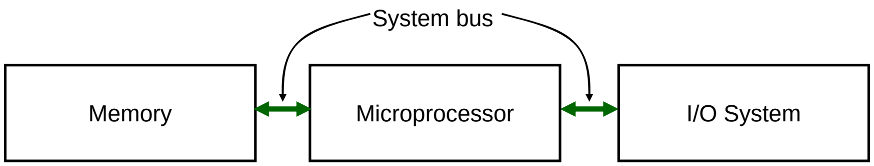


System Bus Model1

- Refinement of the von Neumann Model
  - Same 5 components, but CPU (Central Processing Unit) or microprocessor now contains both ALU and Control Unit.
- All components are attached to a shared communication pathway called the **system bus**.

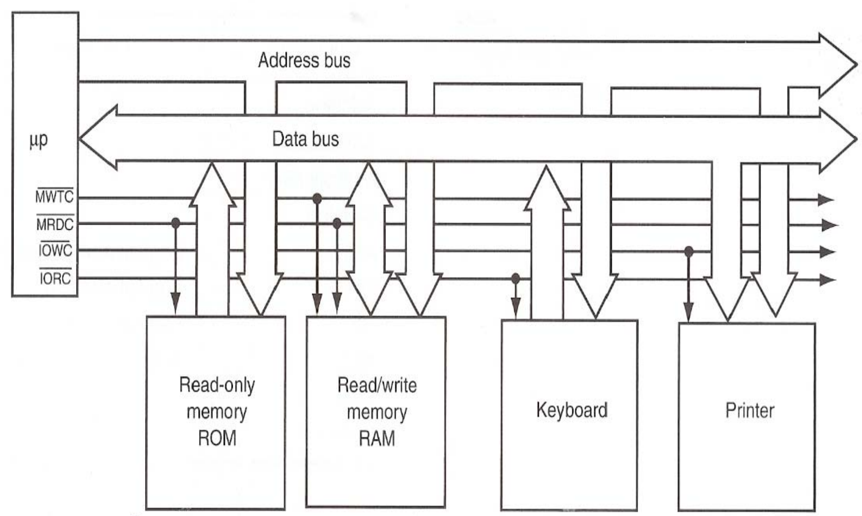


System Bus Model2



Memory	Microprocessor	I/O System
DRAM (Dynamic RAM)	8088	Hard disk
SRAM (Static RAM)	8086	Monitor
Cache	68020	Printer
ROM	6811	Serial communications
Flash Memory	Pentium 4	Mouse
EEPROM	AMD Athlon64	DVD-RW
SDRAM		Keyboard
RAMBUS		Tape backup
DDR RAM		

System Bus Model3



The Microprocessor

- Controls memory & I/O through bus
- 3 main tasks:
  1. Data transfer: MOV, Push*, Pop*, IN, OUT.
  2. Arithmetic and logic: ADD, SUB, MUL*, DIV*, AND, OR, XOR, NOT, NEG, shift, rotate.
  3. Program flow: Branch, Jump, Trap, Loop.
     – Possibly based on flags (e.g. ZERO, SIGN, CARRY, OVERFLOW, PARITY)

* Depends on architecture

The Microprocessor

- Program stored in memory as binary data
  – i.e. stored program computer
- Data widths:
  – Byte (8-bits) (octet in network world)
  – Word (16-bits) *
  – Double word / Long word (32-bits) *

* Depends on architecture

The Microprocessor

- Types of microprocessors
  – General purpose processors
    - Desktop processors (e.g. Intel P4, PowerPC, AMD Athlon XP)
  – GPU – Graphics Processing Unit
    - Specialized for high-throughput graphics processing. Emphasis on data throughput and floating point operations.
  – DSPs – Digital Signal Processors
    - Like GPU, emphasis on throughput (multiple streams) and floating point operations (e.g. multiply&add in single operation)
  – Microcontrollers
    - Entire computer on a single chip. Mainly used for embedded applications. Dominates market.

Memory

- Each addressable location is typically 1 byte of binary data
  – Each memory element (byte) has an address, usually specified in hexadecimal notation.
- Memory size chart:

1KB	2^{10} bytes	1,024 bytes
1MB	2^{20} bytes	1,048,576 bytes
1GB	2^{30} bytes	1,073,741,824 bytes

- Ex: 64KB = 64 x 2^{10} bytes = 65536 bytes
  64K = 2^{16} : need 16 address lines.
- Side note: difference between ‘kilo’ and ‘kibi’
  1 kilo=1K=10^3; 1 kibi=1Ki=2^{10}; 1K = 0.976Ki; Will ignore this…

System Bus

- A bus is a collection of wires or traces that interconnect components.
- Grouped by function.
  - **Data bus**: Moves data among the system components. Can be one- or two-way.
  - **Address bus**: Carries address of element that is being read from or written to.
  - **Control bus**: Carries control signals that coordinate access to the data and address busses.
- Busses can be multiplexed to save pins/wires

Address Bus

- Selects a location in memory or I/O space for reading or writing
- N address lines can access 2^N locations
  - 8086/8088: N=20, 2^{20}, 1M byte
  - 80286/386/68000: N=24, 2^{24}, 16MB
  - 80386DX/486/68020: N=32, 2^{32}, 4GB
  - Pentium II: N=36, 2^{36}, 64GB
  - AMD Athlon64: N=40, 2^{40}, 1TB

Data Bus

- Transfers information between μP and memory or I/O
- Data transfers vary in size:
  - 8088/68008/6811: 8 bits
  - 8086/80286/some 386/68000/68010: 16 bits
  - 386DX/486/68020: 32 bits
  - Pentium… : 64 bits*
  - Some AMD Athlon64: 128 bits*

* Fetches are to cache

Control Bus

- In most ix86 systems, these 4 control signals are found:
  - $\overline{MRDC}$: Memory ReaD Control
  - $\overline{MWTC}$: Memory WriTe Control
  - $\overline{IORC}$: I/O Read Control
  - $\overline{IOWC}$: I/O Write Control

  Note: all signals are active low

Control Bus

- Example read cycle:
  1. μP puts address on address bus
  2. μP drops $\overline{MRDC}$ to cause memory to place data on data bus
  3. μP reads data from data bus

Number Systems

- Review of binary, decimal, and hexadecimal numbers (see text chap 1)
- Binary Coded Hexadecimal (nibbles)
- Examples:

  $7FE_{16} = 0111\ 1111\ 1110_2$

  $3A9h = 0011\ 1010\ 1001_2$

  $6B4Ch = 0110\ 1011\ 0100\ 1100_2$

- Polynomial method (convert to decimal)

  $ABC_{16} = 10(16^2)+11(16^1)+12(16^0) = 2748_{10}$

Hex	Dec	BCH
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

Remainder Method

- Used to convert from decimal to binary
- Example: $123_{10} = 1111011_2 = 7B_{16}$

	Integer	Remainder	
123/2=	61	1	←LSB
61/2=	30	1	
30/2=	15	0	
15/2=	7	1	
7/2=	3	1	
3/2=	1	1	
1/2=	0	1	←MSB

Stop @ 0

Hex Addition

- Examples:

  $C_{16} + 5_{16} = 12_{10} + 5_{10} = 17_{10} = 16_{10} + 1_{10} = 11_{16}$

(carry)	1	1	1	
7EBDh	7_{10}	14_{10}	11_{10}	13_{10}
+ 4ACh		4_{10}	10_{10}	12_{10}
8369h	8_{10}	19_{10}	22_{10}	25_{10}
	8_{10}	$16_{10} + 3_{10}$	$16_{10} + 6_{10}$	$16_{10} + 9_{10}$
	8_{16}	3_{16}	6_{16}	9_{16}

Representation of negative numbers

- Definition: Radix means 'number base'
  - Defines the range of valid digits
  - E.g. decimal->Radix=10, binary->Radix=2, hexadecimal->Radix=16, octal->Radix=8
- Four ways to represent signed numbers:
  - Sign&magnitude
  - Radix-1 complement (e.g. 1's comp)
  - Radix complement (e.g. 2's comp)
  - Excess or biased
- Essentially use half the bit patterns to represent positive numbers, and half to represent negative numbers.

Sign & Magnitude

- Use left-most bit to represent sign.
- Remaining bits represent magnitude
- Examples:

  $-12_{10} = 10001100_2$

  $+12_{10} = 00001100_2$
- For an 8-bit number:
  - Range = [-127,+127]
  - Smallest = -127 = 11111111_2
  - Largest = +127 = 01111111_2
- Pro: simple
- Con: two representations for zero! (+0/-0)

Radix-1 complement (1's comp)

- Subtract each digit from radix-1
- For binary, complement each digit
- Examples:

  $+12_{10} = 00001100_2$

  $-12_{10} = 11110011_2$

  15's comp of $4AB_{16} = B54_{16}$
- For an 8-bit number:
  - Range = [-127,+127]
  - Smallest = -127 = 10000000_2
  - Largest = +127 = 01111111_2
- Pro: simple
- Con: two representations for zero! (+0/-0)

Radix complement (2's comp)

- Add 1 to Radix-1 complement
- Examples:

  $+12_{10} = 00001100_2$

  $-12_{10} = 11110100_2$

  16's comp of $4AB_{16} = B55_{16}$
- For an 8-bit number:
  - Range = [-128,+127]
  - Smallest = -128 = 10000000_2
  - Largest = +127 = 01111111_2
- Pro: Good for arithmetic, single zero bit pattern
- Con: Discontinuity in bit pattern @ 0

  (i.e. 1=00000001, 0=00000000, -1=11111111)
- **Widely** used today

Excess or biased

- Add bias to Radix complement
- Examples (8-bit, excess-127):
  $+12_{10} = 10001011_2$
  $-12_{10} = 01110011_2$
- For an 8-bit excess-127 number:
  - Range = [-127,+128]
  - Smallest = $-127 = 00000000_2$
  - Largest = $+128 = 11111111_2$
- Pro: Smaller numbers have smaller bit patterns; single zero; continuous bit pattern @ 0
  (i.e. 1=10000000, 0=01111111, -1=01111110)
- Con: Difficult to compute by hand
- Used for IEEE-754 floating point

Data Types

- Integers (whole or fixed point)
  - Can be represented using byte (char), word (int), double word (long int)
  - Compiler-dependent
- Binary coded decimal
  - Each nibble stores a single digit
  - Funky instructions
  - No longer used much
- Floating point (deferred)
- ASCII (EBBCDIC)
  - Characters and strings
- All data represented using bits in microprocessor, regardless of type!

Data sizes

- Bytes (signed or unsigned)
  - Bit ordering: 7 6 5 4 3 2 1 0
  - For signed byte, high order bit is sign and carries weight of -128_{10}
- Words (signed or unsigned)
  - Formed from 2 bytes
  - Little endian (Intel) vs. Big endian (Motorola)
- Double (long) words
  - Formed from 4 bytes
  - Little endian vs Big endian

Little Endian (Intel)

- Least significant byte stored in lower-numbered memory location
- Consistent with BIT ordering
- Require byte flipping in network code
- Ex: store AC43h to memory location 2000h:

2002H		
2001H	AC	High order byte
2000H	43	Low order byte
1FFFH		

1FFFH	2000H	2001H	2002H
	43	AC	

Big Endian (Motorola, Network)

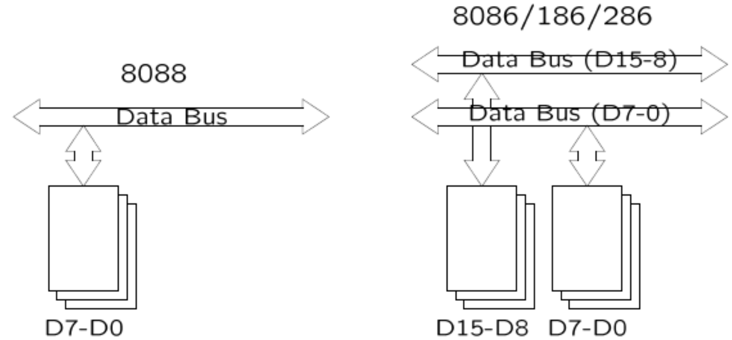
- Most significant byte stored in lower-numbered memory location
- When incrementing through memory, highest (most significant) byte is first (like writing a number by hand)
- Ex: store AC43h to memory location 2000h:

2002h		
2001h	43	Low order byte
2000h	AC	High order byte
1FFFh		

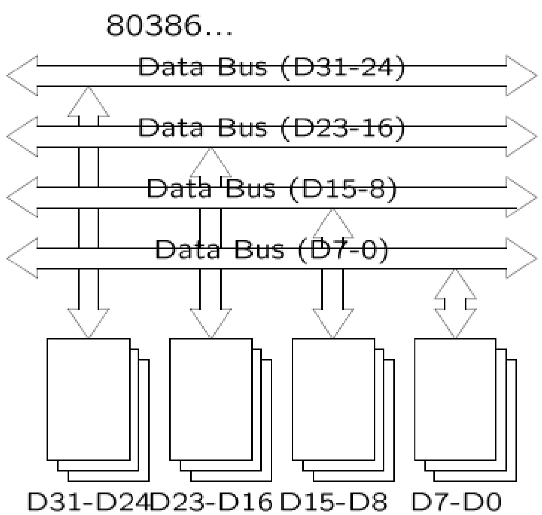
1FFFh	2000h	2001h	2002h
	AC	43	

Memory Organization

- Memory devices are arranged in bytes of 8-bits (modulo parity/ECC)
- μP may have 8, 16, 32, or 64 data lines
- Each memory chip returns a single byte
  - Therefore, multiple banks of memory chips are used.
- Each bank requires a ‘bank enable’ signal



Memory Organization – 32 bit data bus



- What do we do for a 64 bit data bus? 128?
- See webpage for problem set 1