Course Introduction

CSC 236

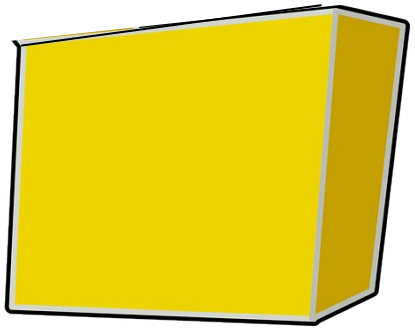
Computer Organization and Assembly Language Programming

First course goal

What happens beneath high-level languages

Machine code

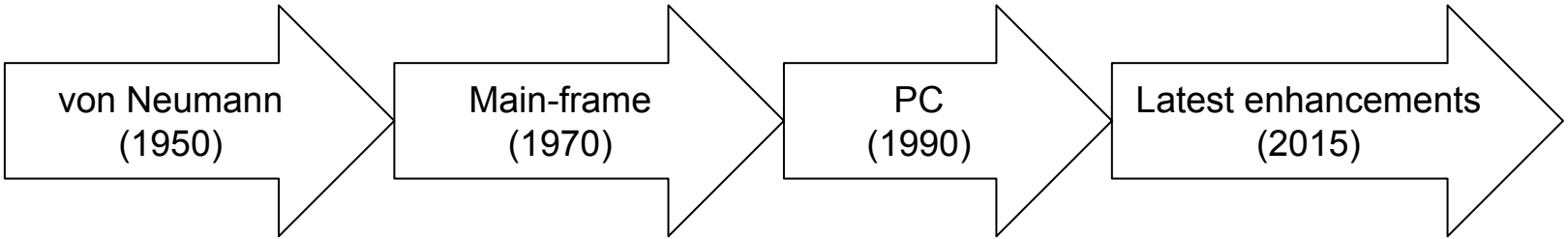
How it is created and executed



Second course goal

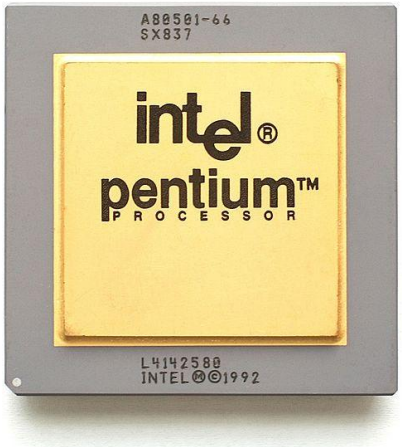
How computers are designed and organized

Developed over time

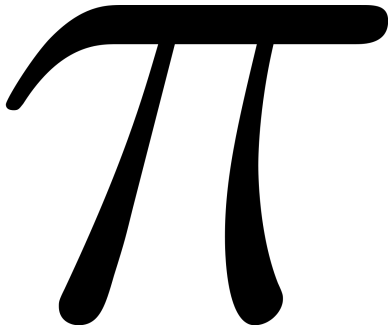


Second course goal

Examine several architectures



x86



floating point

Java VM

ARM

Third course goal

Performance and efficiency

It still matters even though

- clock is greater than 1GHz and
- memory is greater than 32GB.

Why

Isn’t is enough to know Java?

No.

Your degree will not say

- Application programmer or
- Web designer

It will say “computer science”

Information sources

- Class Notes
  - online & coursepak
  - text & 8086 reference
- Slides
  - Links on Moodle page
- Moodle site
  - Homeworks
  - Previous tests
  - Grades
- GitHub
  - Programs

Grading policies

Exams are online and open book

Tradeoff:

- Closed book tests memorization
- Open book requires more difficult questions

Allowed

- All content on the course web page
- All context created by student (self -- not others) **BEFORE** exam begins
- Textbooks

Homework

Nine homeworks

- HW0 -- answer some questions
- HW1-4
  - Fixed-point representation
  - Addition and subtraction in binary
  - Self/auto-graded
- HW5-8
  - Multiplication and division
  - Self/auto-graded

Self-grading assignments

- In general, you will know your score before you submit
- You can work on an assignment until you have the grade you are satisfied with

Self-grading assignment

- In general, you will know your score before you submit
- You can work on an assignment until you have the grade you are satisfied with

Won't everyone get a perfect score?

Grading policies that help you

If being late is bad -- being early should be ... good?

Extra credit (+5%) for early submissions

Early = 72 hours before the deadline.

Late work is accepted (up to 1 week) with a penalty (-15%)

Grading policies that help you

The majority of the course grade is based on the programs & homeworks.

- Two midterms and a final
  - 12% Exam 1
  - 12% Exam 2
  - 20% Final Exam

Grading policies that help you

A few optional team assignments.

Some assignments can be done in teams of two students.

See the syllabus.

Program development

Using technology from the dawn of PC architecture

Program development

Windows programming

Program development

~~Windows programming~~

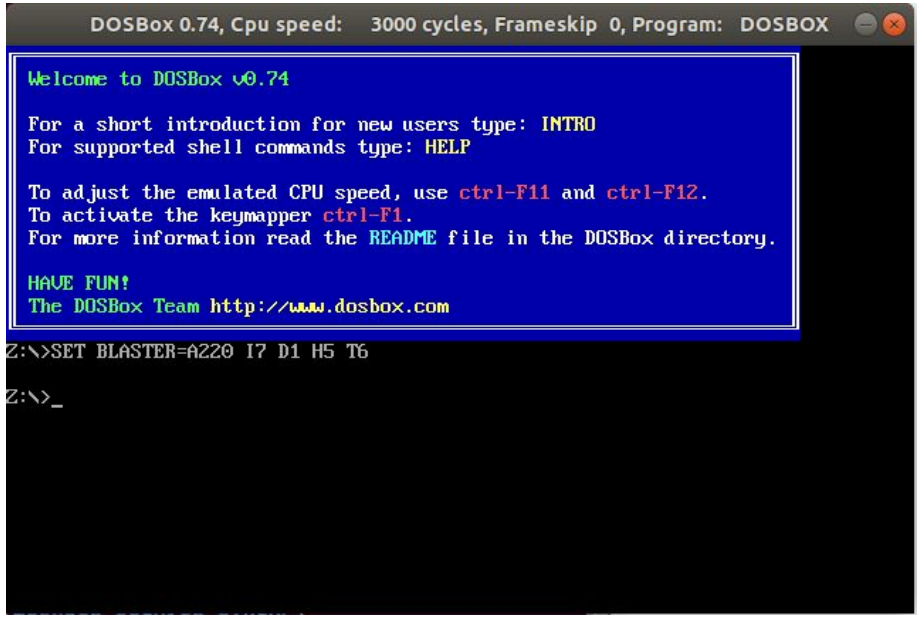
Nope: DOS programming

Program development

- IBM compatible 16-bit DOS box
- DOSBox
  - Emulator
  - Runtime environment
- Download DOSBox to your computer
- Launch it

Program development

- IBM compatible 16 bit DOS box
- DOSBox
  - Emulator
  - Runtime environment
- Download DOSBox to your computer
- Launch it



Program development

- IBM compatible 16 bit DOS box
- DOSBox
  - Emulator
  - Runtime environment
- Download DOSBox to your computer
- Launch it
  - Get a DOS prompt
  - Run commands
  - Uses your native filesystem

What is DOS

- DOS -- disk operating system
  - Disk is a *floppy disk*

What is DOS

- DOS -- disk operating system
  - Disk is a *floppy disk*

360KB
double sided

What is DOS

- DOS -- disk operating system
  - Disk is a *floppy disk*
- Command line driven
  - dir -- list directory contents
  - cd -- change directory
  - copy -- copy a file

What is DOSBox

- An emulator
  - It is a substitute for a “DOS box” -- i.e., a PC running DOS
  - Definition: Computers To imitate the function of (another system)
- Not a simulator
  - Definition: To create a representation or model of (a physical system or particular situation)

What is 16-bit

- More later in the course
- Original IBM PC
  - Ran MS-DOS on
  - Intel 8086 processor
- 8086 is 16-bit architecture
  - The basic word size is 16 binary digits
- Intel chips have evolved
  - To 32-bit and
  - Now 64-bit
- The 16-bit architecture is much simpler

HW0

- Shows the essence of CSC236
- Run a short program
  - Java or C

Java Program

```
import java.io.*;
public class hw0
{
  public static void main(String args[])
  {
  short a;                              //declare 'a'
  a = 20000;                            //set to 20000
  a = (short)(a + a);                   //double  'a'
  System.out.println(a);                //display 'a'
  }
}
```

HW0

- Shows the essence of CSC236
- Run a short program
  - Java or C
- Answer 11 questions about the program's execution
- Create and submit a **Plain ASCII text** file with your answers

HW1-8

Generated and graded on your system

Install DOSBox

Retrieve and unpack the program file

- unpack.exe -- located in under “Homeworks” on our Moodle page
- Run `unpack.exe` in DOSBox
  it creates the files you need
- Run `hw14`
  - Generates unique questions
  - (Print and answer the questions)
  - Enter answers and get a grade
  - Re-do if not happy with grade

When satisfied with grade;
submit the file

hwsubmit.txt

to locker on web site.

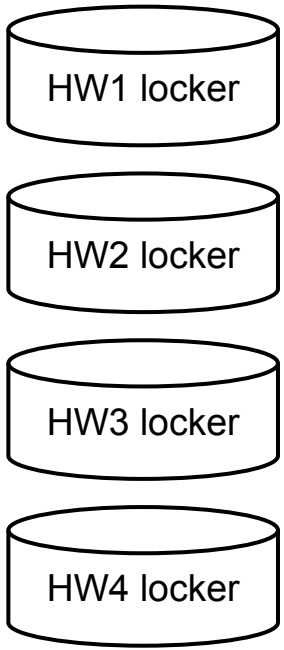
HW14

HW14 program works for 4 assignments.

```
CSC23x: Homework Grades Student: Sauron
Created: 01/08/18

HW# Started       Status   Grade
HW1 01/08/20      D 01/09  096
HW2
HW3
HW4

End of file
```



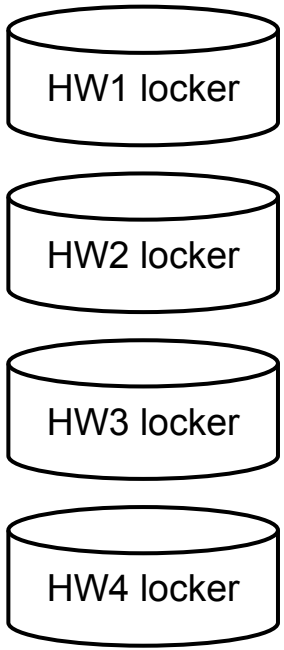
HW14

HW14 program works for 4 assignments.

```
CSC23x: Homework Grades Student: Sauron
Created: 01/08/18

HW# Started       Status   Grade
HW1 01/08/20      D 01/09  096
HW2 01/10/20      D 01/11  100
HW3
HW4

End of file
```



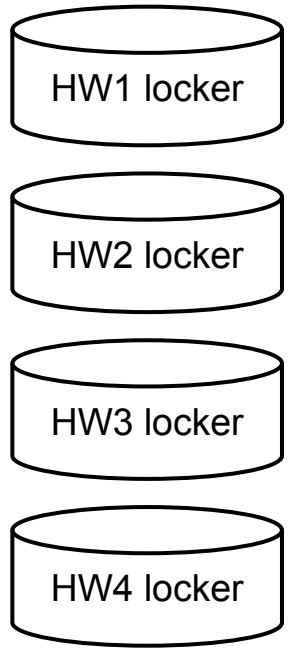
HW14

HW14 program works for 4 assignments.

```
CSC23x: Homework Grades Student: Sauron
Created: 01/08/18

HW# Started       Status   Grade
HW1 01/08/20      D 01/09  096
HW2 01/10/20      D 01/11  100
HW3 01/16/20      D 01/17  100
HW4

End of file
```



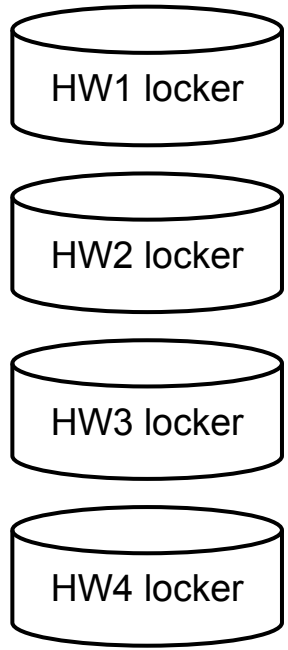
HW14

HW14 program works for 4 assignments.

```
CSC23x: Homework Grades Student: Sauron
Created: 01/08/18

HW# Started       Status   Grade
HW1 01/08/20      D 01/09  096
HW2 01/10/20      D 01/11  100
HW3 01/16/20      D 01/17  100
HW4 02/04/20      D 02/05  094

End of file
```



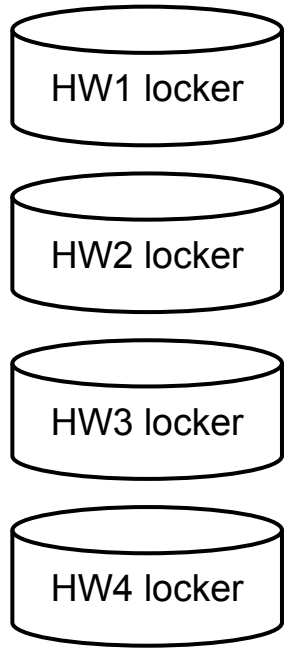
HW14

HW14 program works for 4 assignments.

```
CSC23x: Homework Grades Student: Sauron
Created: 01/08/18

HW# Started       Status   Grade
HW1 01/08/20      D 01/09  096
HW2 01/10/20      D 01/11  100
HW3 01/16/20      D 01/17  100
HW4 02/04/20      D 02/06  098

End of file
```



How to succeed

- Read the handouts!
- Do the programs!
- Start early!

Introduction to Computer Architect and Assembly Language

CSC 236

Answer four questions

- What is computer architecture?
- What is assembly language?
- How are they related?
- What is the purpose of this class?

Architecture

- Definition: Art or science of building

Computer architecture

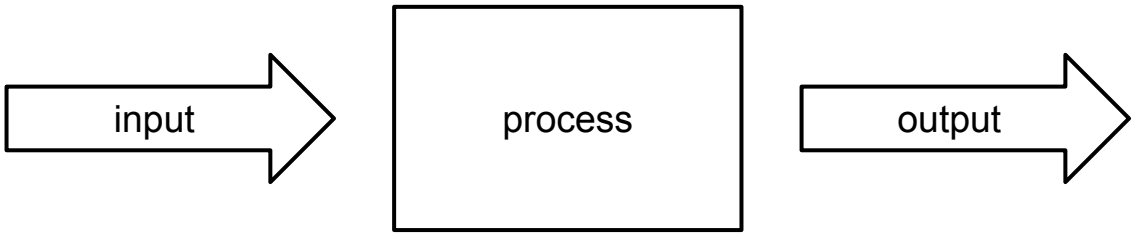
- Describes how computers are built
- Definition: The specification of the logical relationships among the parts of a computer system (*)

(*) Prentice Hall Illustrate dictionary of computing.

The specification of the **logical** relationships among the parts of a computer **system**

System

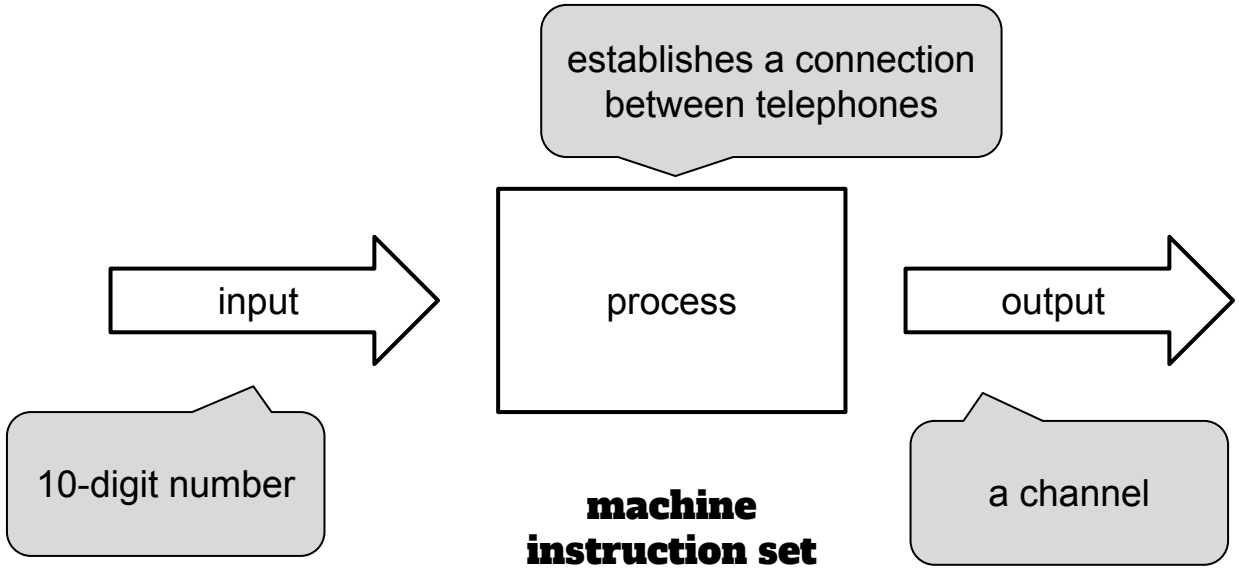
A set of components organized to perform a task



machine instruction set

All systems have these 4 characteristics

Example: Telephone system



Two views of a system

Logical view

- Function
- What are its parts
- What does it do
- User's manual

Physical view

- How it is built
- What is it made of
- Speed, size, weight
- Cost

Orthogonal

Example: Shovel

Logical view

- Moves things
- Handle
- Connecting rod
- Blade

Physical view

Example: Shovel

Logical view

- Moves things
- Handle
- Connecting rod
- Blade

Physical view

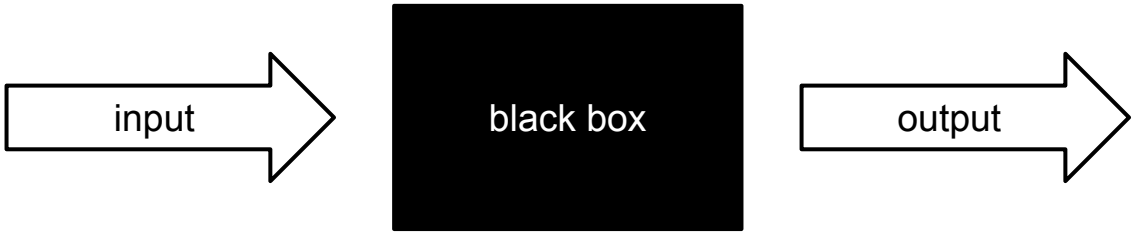
Example: Shovel

Logical view

- Moves things
- Handle
- Connecting rod
- Blade

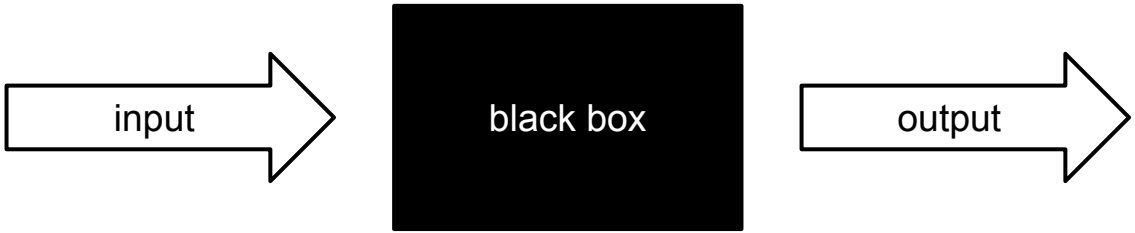
Physical view

Can one study a system only logically



machine instruction set

Can one study a system only logically



machine instruction set

Instructions: **add, subtract, multiply, divide**
Data: **integer, floating point, character**

Assembly language

- Symbolic programming language whose
- Statements match 1:1 with instructions
- A specific machine instruction set

Assembly language

- Symbolic programming language whose
- Statements match 1:1 with instructions
- A specific machine instruction set

What does this mean?

Assembly language

- Symbolic programming language whose
- Statements match 1:1 with instructions
- A specific machine instruction set

This is significantly better than programming with machine code, ie 0s & 1s

- Every machine has certain capabilities
- No high-level language features
- Code at the architectural level -- but symbolically

Why is this important?

- To a user, a computer is
  - A tool to get a job done
- To an analyst, a computer is
  - The end product
- As developers, we want to make our programs
  - Faster
  - Cheaper
  - More capable
  - More reliable
  - More secure
  - ...