Lecture 13
Review / Exam Study Guide
IAT267 Introduction to Technological Systems

Fall 2011

Final exam – December 16, 2011, 8:30 – 11:30

- Closed book exam

- Questions from the entire course content
  - Lectures
  - Workshops
  - Assignments

- You will not be given to write code from scratch, or draw circuits – however you will be asked to explain or modify the functionality of a given circuit or code

- No multiple choice questions
- The exam has three parts, corresponding to three parts of the course
  - 1. Technological systems – computer systems
  - 2. Sensors, Arduino, Processing
  - 3. Networking
- Help available in office hours in the week before the exam - TBA

Exam questions

- Will ask you to explain /describe various topics from the course
  - For example:
    - explain the piezoelectric effect and how it is used in sensors
    - Describe the von Neumann computer architecture (each component)
    - Describe the characteristics of the UDP protocol

- Explain and/or modify given code – Processing, Arduino

- For a given circuit, explain the functionality, calculate some values, and maybe modify the circuit
  - For example:
    - circuit containing a voltage divider – calculate voltage
    - Series/parallel circuits – calculate equivalent resistance
    - Circuits containing sensors – modify the circuit

Grades on WebCT

- Check your participation grades
  - Workshop, quizzes

- If you see errors, inaccuracies please email Helmine asap

Course Project

- Presentations this Friday, during workshops
- Review Milestone 4 on webct
- All team members should participate actively in the project presentation

Unit by Unit Review

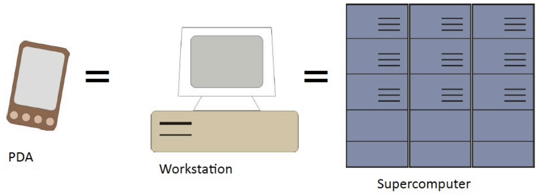
- In today's class we will list the topics required for the exam
- We will have a brief review for each topic
- To prepare for the exam, study the topics from the lecture slides and readings associated with lectures

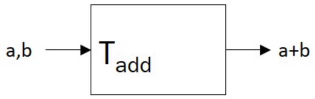
Unit 1: Technological Systems

- What are technological systems
  - – Classification
  - – Discussion on
    - – Computer systems
    - – Embedded systems
  - – Why study technological systems
    - – Abstraction
    - – Hardware / software
  - – Computer systems

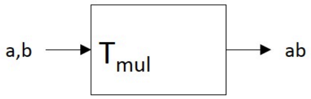
Unit 1: Technological Systems (cont)

- Two general principles of computer systems:
  - Universal computing device (Turing)
    - Time, memory
    - Turing machine
    - Universal Turing machine

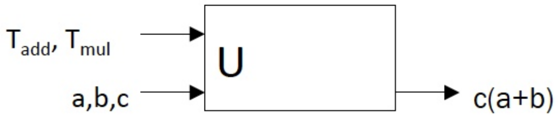




Turing machine that adds

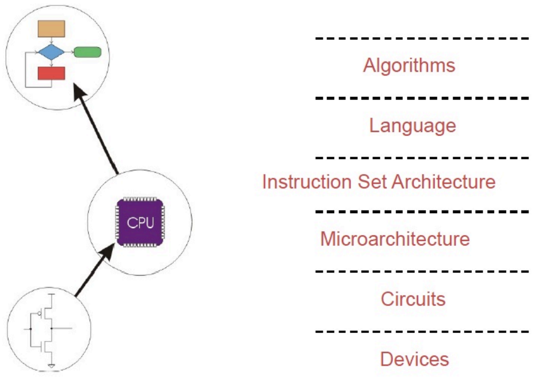


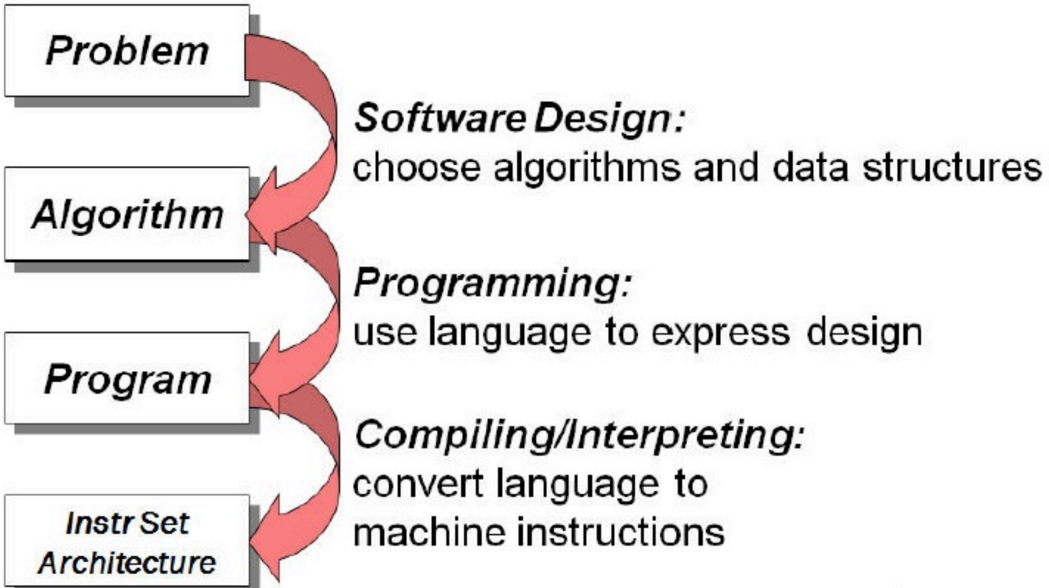
Turing machine that multiplies



Universal Turing Machine

  - Transformation between layers





Problem

Algorithm

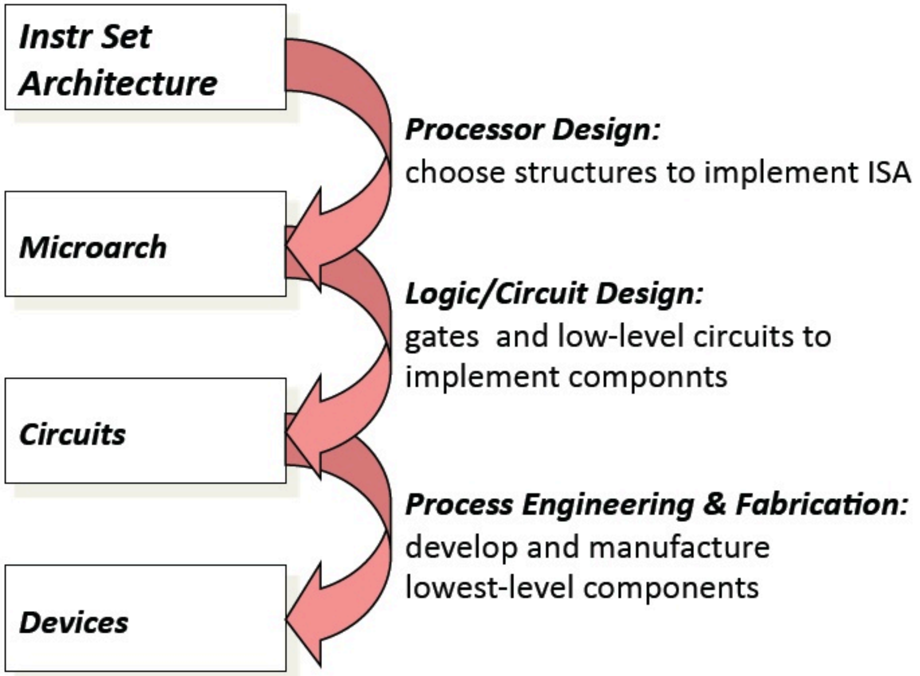
Program

Instr Set Architecture

Software Design:
choose algorithms and data structures

Programming:
use language to express design

Compiling/Interpreting:
convert language to
machine instructions



Instr Set Architecture

Microarch

Circuits

Devices

Processor Design:
choose structures to implement ISA

Logic/Circuit Design:
gates and low-level circuits to
implement componnts

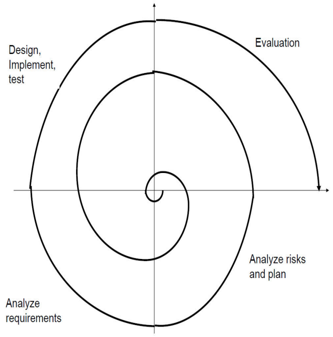
Process Engineering & Fabrication:
develop and manufacture
lowest-level components

Unit 1: Technological Systems (cont)

- Computer systems: key concepts
  - Purpose of a computer
  - Hardware and software
  - Basic operations
    - Input
    - Processing
    - Storage
    - Output
    - communications

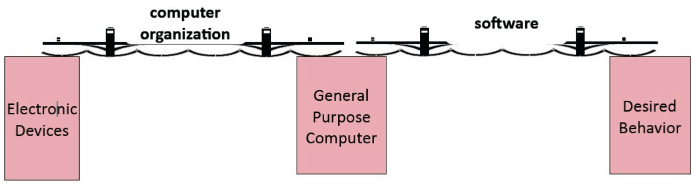
Unit 1: Technological Systems (cont)

- Technological systems: key concepts
  - Iterative design
  - Trade-offs
  - Real-world constraints
  - Feedback
  - Complexity management techniques

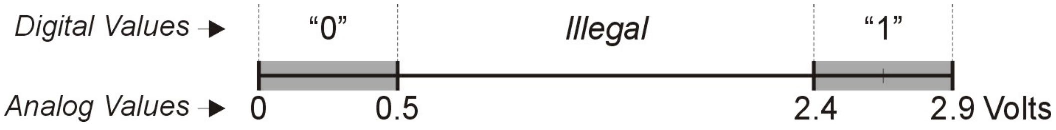


Unit 2: Technological Systems – Key Concepts. Computer Hardware

- Computer organization

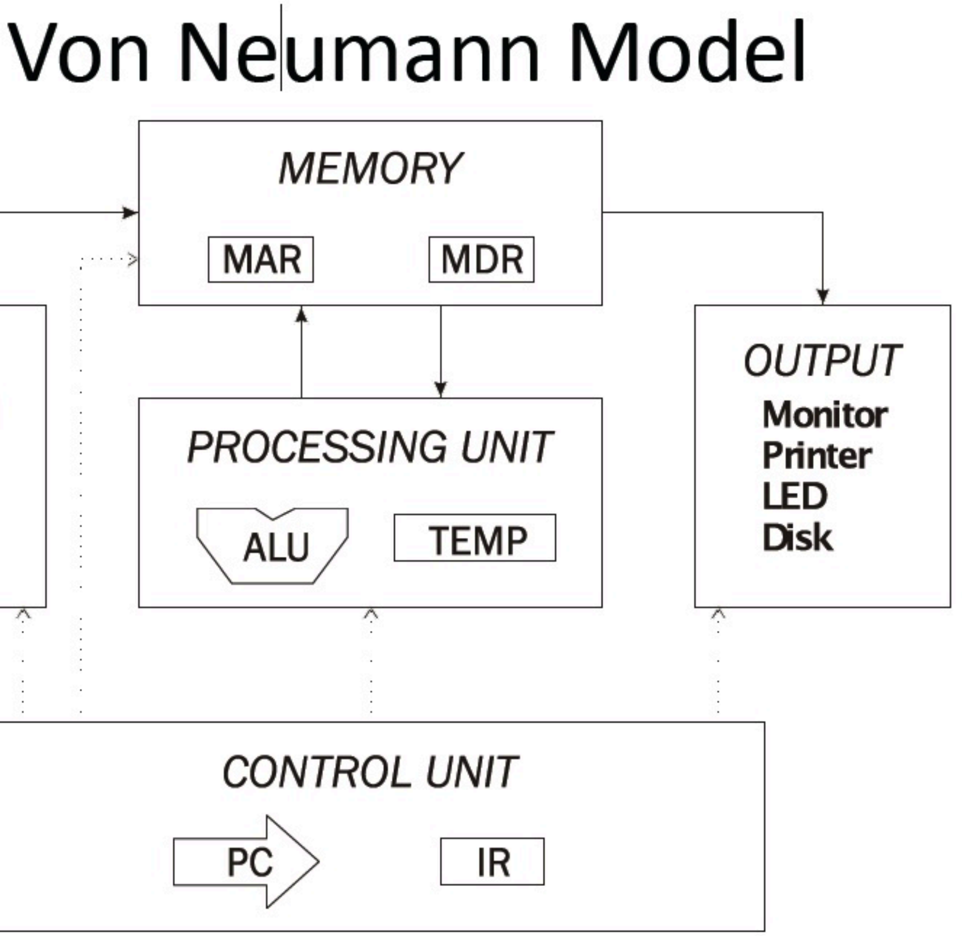


- Representation of data
  - Analog and digital data
  - Binary representation



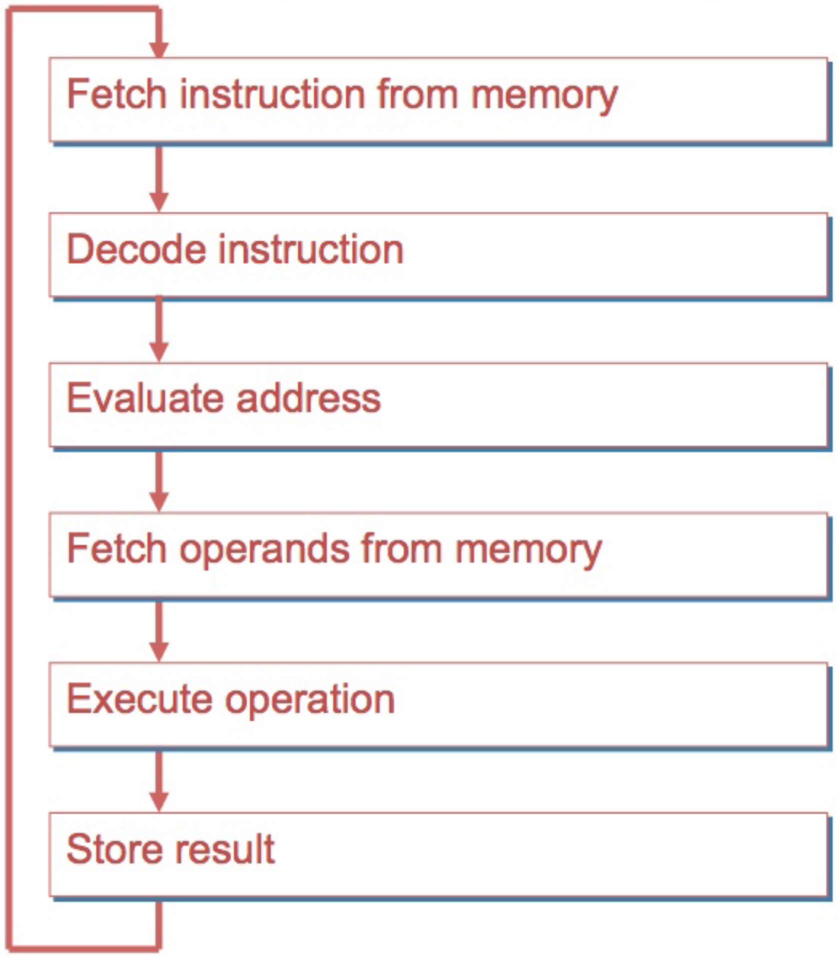
Unit 2: Technological Systems – Key Concepts. Computer Hardware (cont)

- Computer architecture


Von Neumann Model

Unit 2: Technological Systems – Key Concepts. Computer Hardware (cont)

Instruction Processing



Fetch instruction from memory

Decode instruction

Evaluate address

Fetch operands from memory

Execute operation

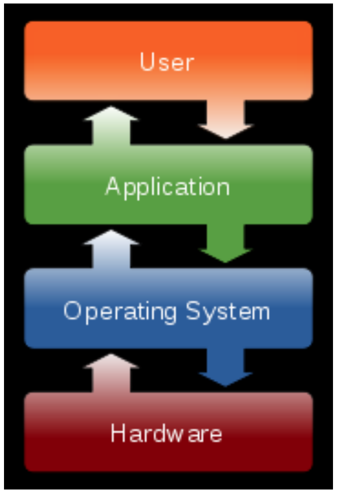
Store result

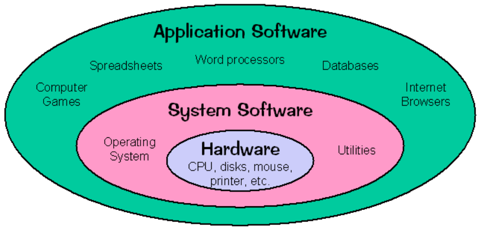
Unit 2: Technological Systems – Key Concepts. Computer Hardware (cont)

- Software
  - Application software
  - System software
    - Operating system
    - Device driver and utilities

Unit 3: Computer Hardware. Software. Intro to sensors.

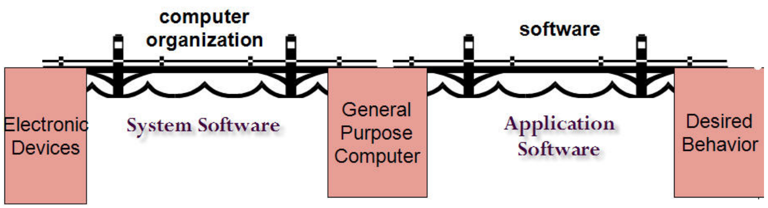
- Classification of software
- Operating system
  - Role
  - Benefits
  - API
  - Core tasks of an OS





1. Processor management
2. Memory management
3. Device management
4. Storage management
5. Application Interface
6. User Interface

Unit 3: Computer Hardware. Software. Intro to sensors (cont)



computer organization

software

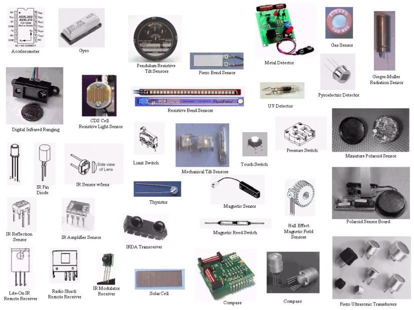
Electronic Devices

System Software

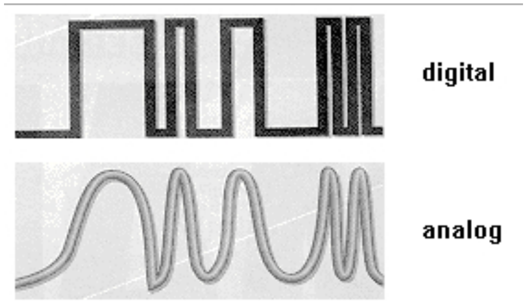
General Purpose Computer

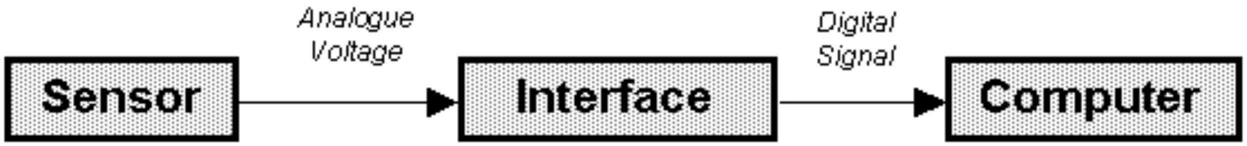
Application Software

Desired Behavior



- Extending the computer system
  - analog data
  - sensors





Sensor

Analogue Voltage

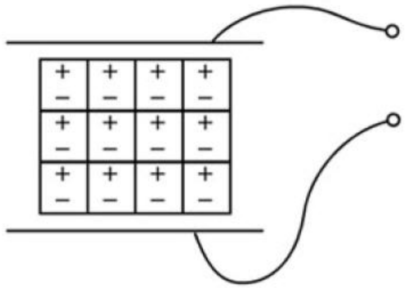
Interface

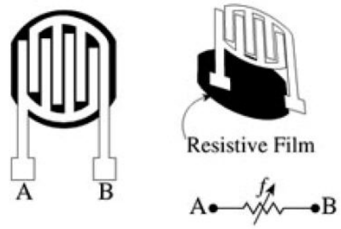
Digital Signal

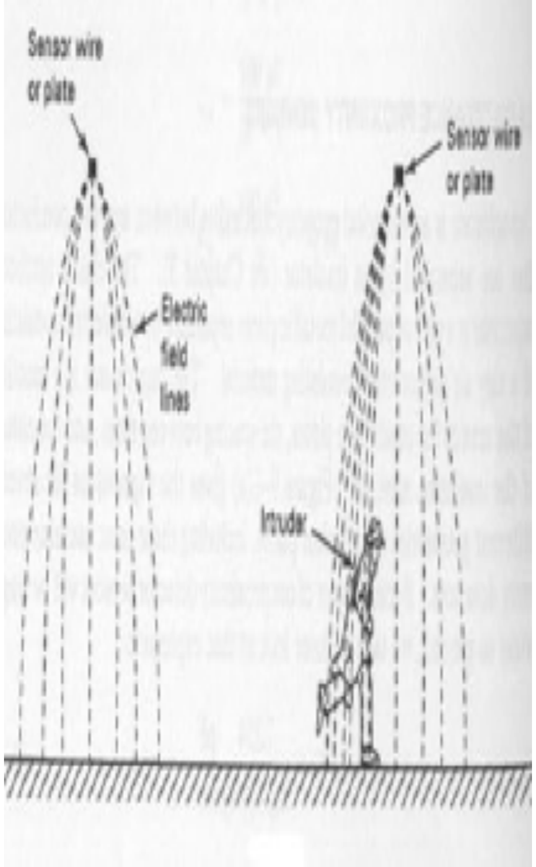
Computer

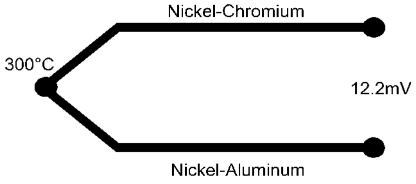
Unit 3: Computer Hardware. Software. Intro to sensors.(cont)

- Classification of sensors
  - by the underlying physics of their operation
  - by the particular phenomenon they measure
  - by a particular application
- Sensor technology: underlying physical principl



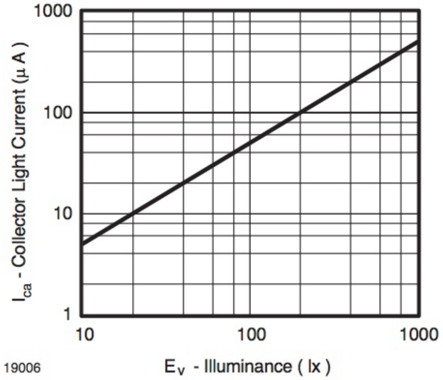


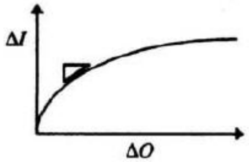




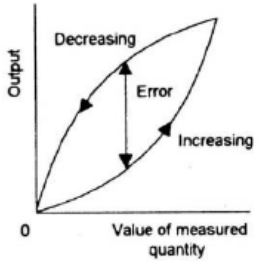
Unit 4: Quality Parameters of a Sensor System. Analog sensors. Voltage divider principle.

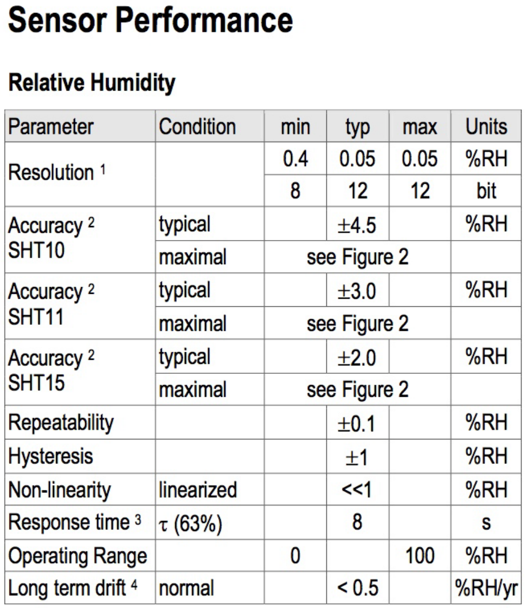
- Quality parameters





Sensitivity $S = \frac{\Delta O}{\Delta I}$





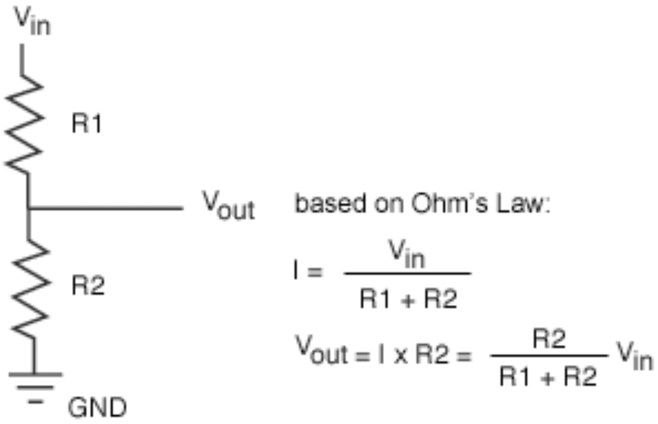
Sensor Performance

Relative Humidity

Parameter	Condition	min	typ	max	Units
Resolution [1]		0.4	0.05	0.05	%RH
		8	12	12	bit
Accuracy [2] SHT10	typical		±4.5		%RH
	maximal	see Figure 2			
Accuracy [2] SHT11	typical		±3.0		%RH
	maximal	see Figure 2			
Accuracy [2] SHT15	typical		±2.0		%RH
	maximal	see Figure 2			
Repeatability			±0.1		%RH
Hysteresis			±1		%RH
Non-linearity	linearized		<<1		%RH
Response time [3]	τ (63%)		8		s
Operating Range		0		100	%RH
Long term drift [4]	normal		< 0.5		%RH/yr

Unit 4: Quality Parameters of a Sensor System. Analog sensors. Voltage divider principle (cont)

- Sources of sensor error
  - Insertion errors
  - Application errors
  - Characteristic errors
  - Dynamic errors
  - Environmental errors
- Sensor output
  - Digital
  - Analog
- Voltage divider principle



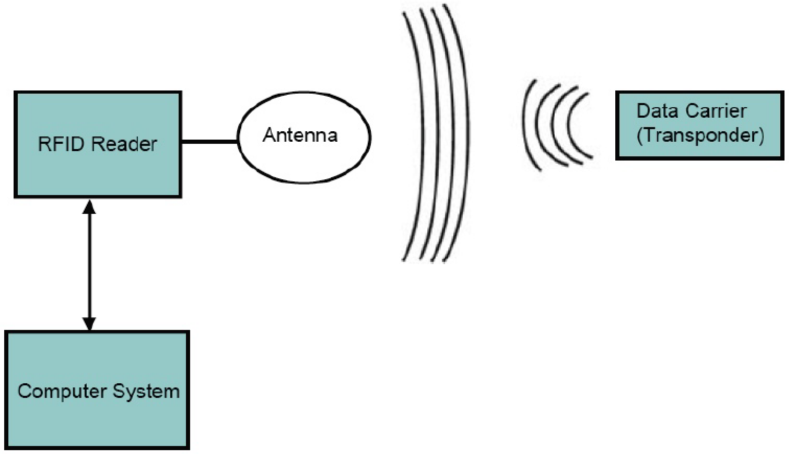
based on Ohm's Law:

$$I = \frac{V_{in}}{R1 + R2}$$

$$V_{out} = I \times R2 = \frac{R2}{R1 + R2} V_{in}$$

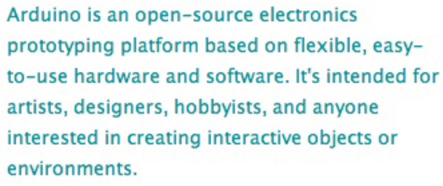
Unit 4: Quality Parameters of a Sensor System. Analog sensors. Voltage divider principle (cont)

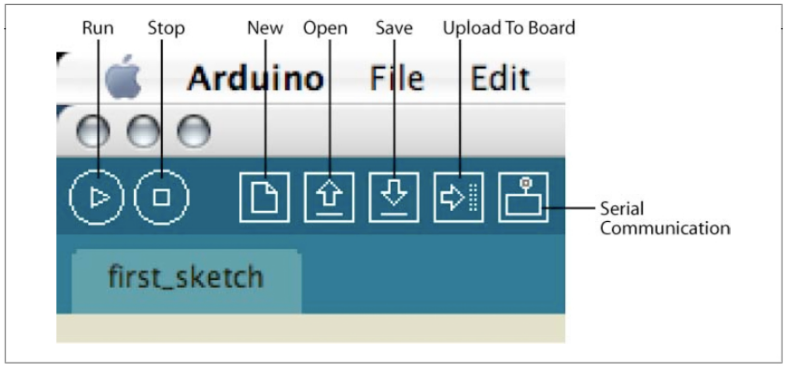
- RFID
  - Components of an RFID system
  - Principle of operation
  - Advantages

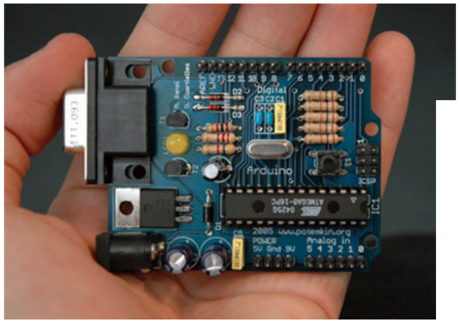


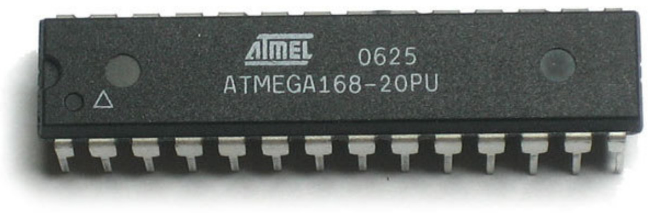
Unit 5: Arduino Basics.

- What is Arduino
- Arduino family
  - Hardware board
  - Programming language
  - Developing environment


Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments.

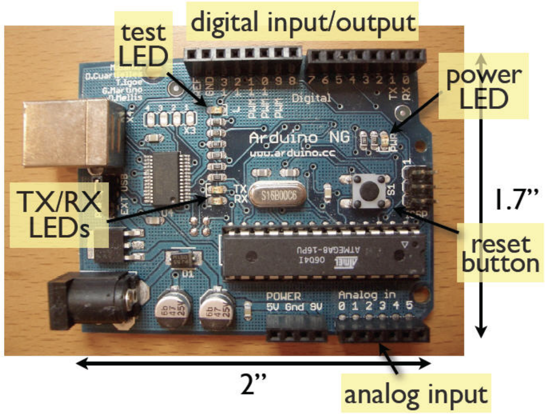






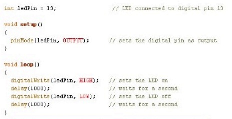
Unit 5: Arduino Basics (cont)

- Arduino board
  - Components / role



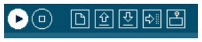
- Arduino development cycle
- Arduino language - Wiring

- Edit code

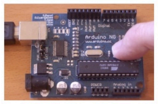


```
int ledPin = 13;                // LED connected to digital pin 13

void setup()
{
  pinMode(ledPin, OUTPUT);      // sets the digital pin as output
}

void loop()
{
  digitalWrite(ledPin, HIGH);   // sets the LED on
  delay(1000);                  // waits for a second
  digitalWrite(ledPin, LOW);    // sets the LED off
  delay(1000);                  // waits for a second
}
```

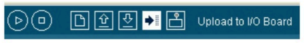
- Compile



- Reset board

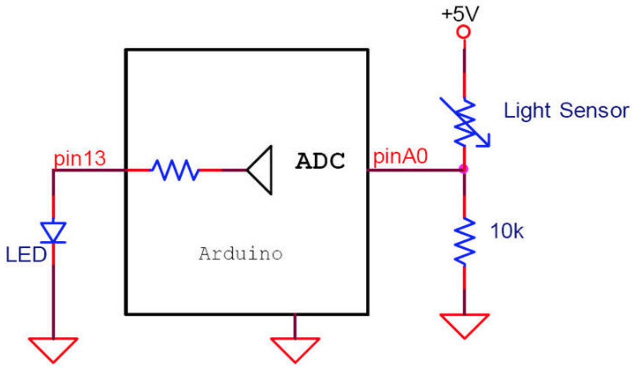


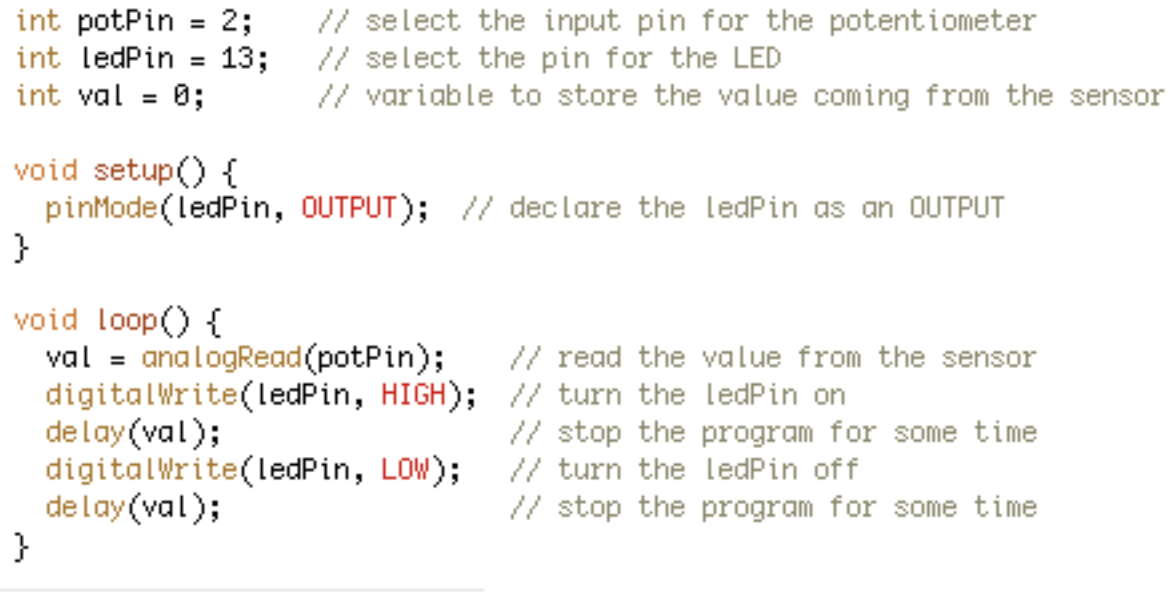
- Upload



Unit 6: Arduino – analog input.

- Arduino code – general structure
  - Be able to explain instructions in a sample code

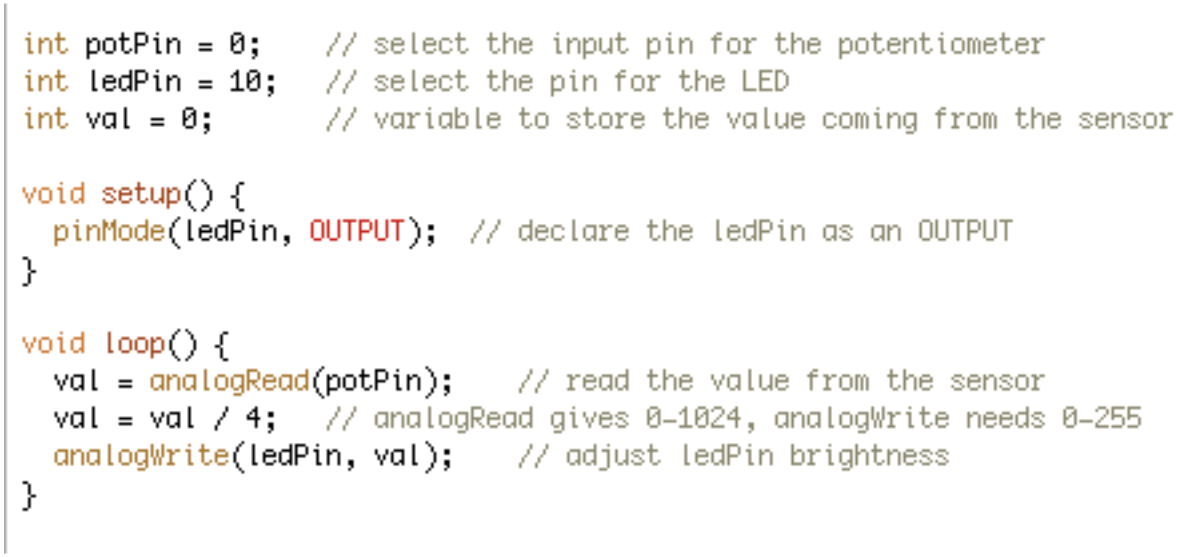




```
int potPin = 2;    // select the input pin for the potentiometer
int ledPin = 13;   // select the pin for the LED
int val = 0;       // variable to store the value coming from the sensor

void setup() {
  pinMode(ledPin, OUTPUT);  // declare the ledPin as an OUTPUT
}

void loop() {
  val = analogRead(potPin);    // read the value from the sensor
  digitalWrite(ledPin, HIGH);  // turn the ledPin on
  delay(val);                  // stop the program for some time
  digitalWrite(ledPin, LOW);   // turn the ledPin off
  delay(val);                  // stop the program for some time
}
```

Unit 7: Arduino – analog input and output. Serial communication.

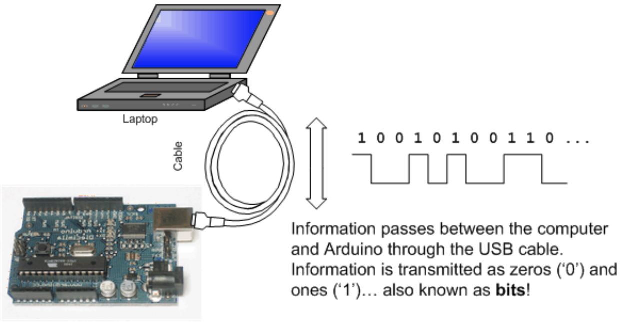
- Adjust the brightness of LED
- PWM pins – analogWrite()



```
int potPin = 0;    // select the input pin for the potentiometer
int ledPin = 10;   // select the pin for the LED
int val = 0;       // variable to store the value coming from the sensor

void setup() {
  pinMode(ledPin, OUTPUT);  // declare the ledPin as an OUTPUT
}

void loop() {
  val = analogRead(potPin);    // read the value from the sensor
  val = val / 4;   // analogRead gives 0-1024, analogWrite needs 0-255
  analogWrite(ledPin, val);    // adjust ledPin brightness
}
```

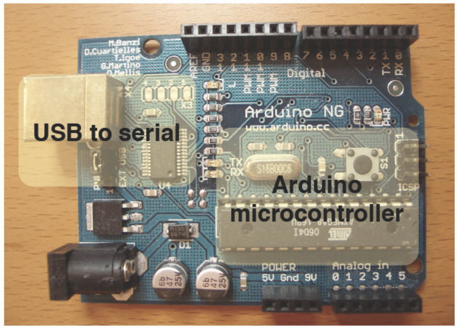
Unit 7: Arduino – analog input and output. Serial communication (cont)

- Serial communication
  - Definition
  - Arduino and serial comm
  - How serial comm happened within Arduino (usb/ serial)

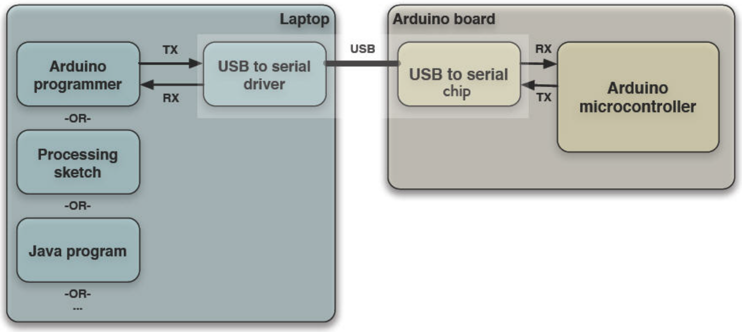


Information passes between the computer and Arduino through the USB cable. Information is transmitted as zeros ('0') and ones ('1')... also known as **bits**!

Arduino board is really two circuits



Arduino to Computer

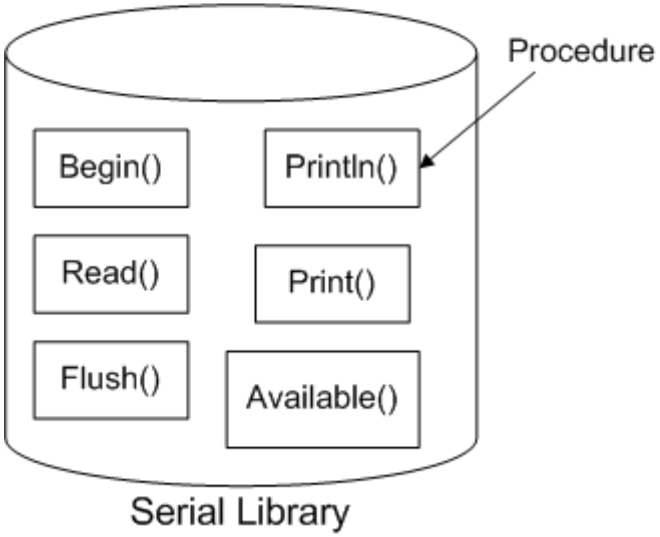


USB is totally optional for Arduino
But it makes things easier

Unit 7: Arduino – analog input and output. Serial communication (cont)

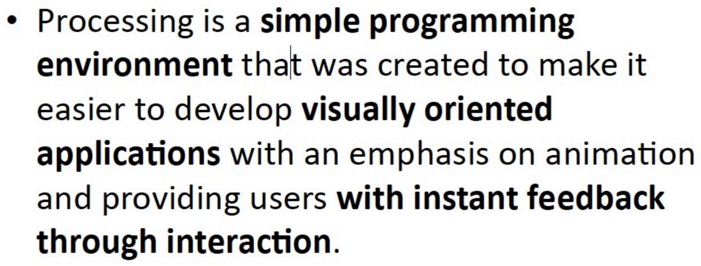
- Serial comm protocol
- Arduino Serial Library

- Physical Connection – serial port
- Timing – speed (bps)
- Electrical Connection
- Package size



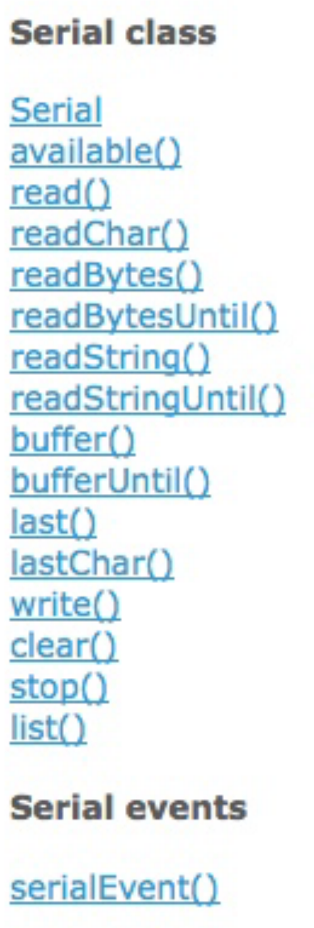
Unit 8: Processing (and Arduino)

- What is Processing



- Processing is a **simple programming environment** that was created to make it easier to develop **visually oriented applications** with an emphasis on animation and providing users **with instant feedback through interaction**.

- How to communicate between Arduino and Processing
  - Processing serial library
- Explain code
  - Drawing, color, shapes
  - Keyboard / mouse interaction



Serial class

Serial
available()
read()
readChar()
readBytes()
readBytesUntil()
readString()
readStringUntil()
buffer()
bufferUntil()
last()
lastChar()
write()
clear()
stop()
list()

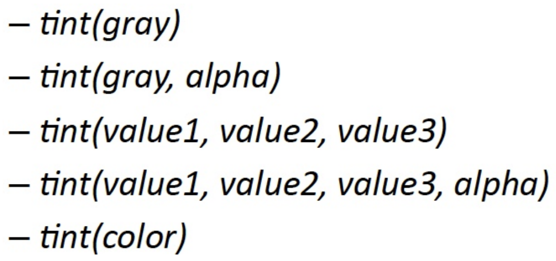
Serial events

serialEvent()

Unit 9: Processing and Arduino topics.

- Mouse, keyboard interactivity, events

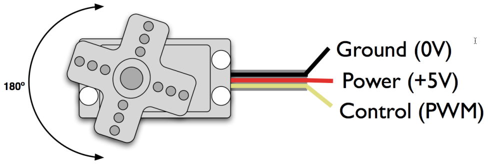
- Images in Processing
  - Color depth
  - Color, transparency – parameters – different ways of using parameter (rgb, grayscale, create color)

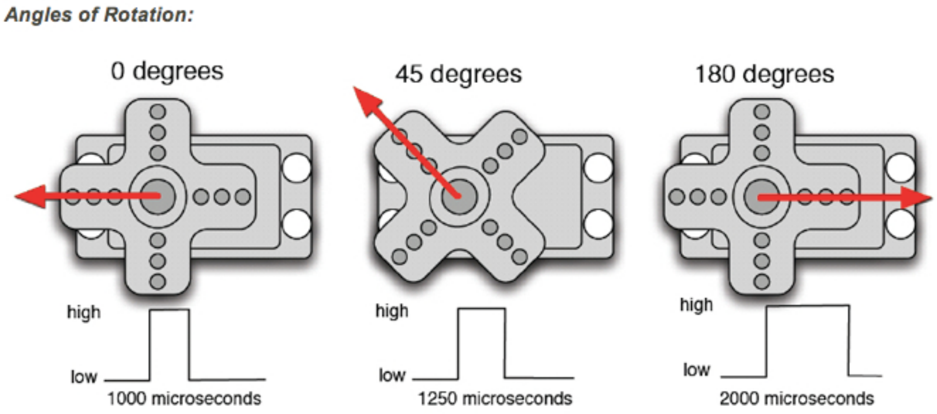


– *tint(gray)*
– *tint(gray, alpha)*
– *tint(value1, value2, value3)*
– *tint(value1, value2, value3, alpha)*
– *tint(color)*

Unit 10: Processing and Arduino topics.

- Servomotors
  - What are servomotors
  - How can we control them: pulse, angle
  - Explain codes for both modalities of control
  - Servo library
  - Advantages of using servos





Unit 10: Processing and Arduino topics (cont)

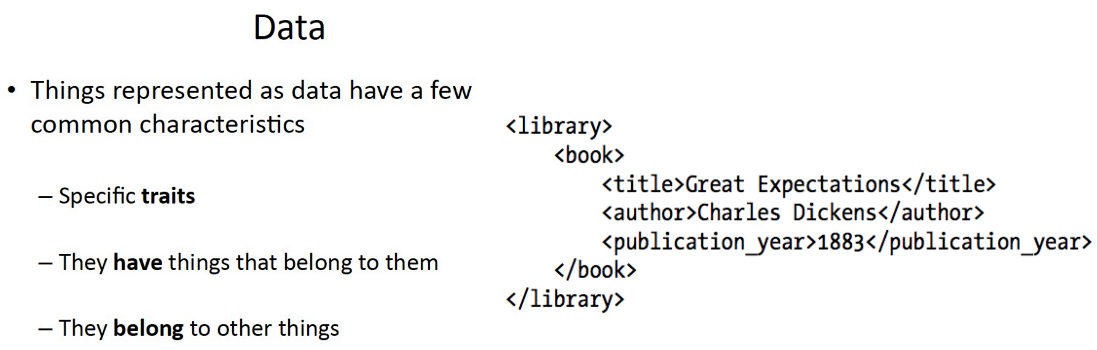
- Networking and Processing
  - Data in an application: sources of data
    - User: gets data from an application:
      - To understands what the application is doing
      - How the user actions have been interpreted
      - How to get the application to do what the user wants

      →FEEDBACK
    - Data sent by the user to the application

      →INPUT
  - Network protocols
    - What is being communicated
    - How that communication is going to be encoded
    - What channels the communication is going to take place on
    - Speed of transmission
  - Communicating over networks
    - examples
      - Remote data
      - Remote access
      - Remote control

Unit 10: Processing and Arduino topics (cont)

- XML
  - What is XML
  - Code structure



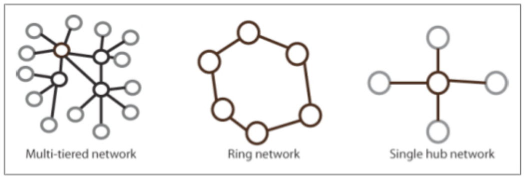
Data

- Things represented as data have a few common characteristics
  - Specific **traits**
  - They **have** things that belong to them
  - They **belong** to other things

```
<library>
    <book>
        <title>Great Expectations</title>
        <author>Charles Dickens</author>
        <publication_year>1883</publication_year>
    </book>
</library>
```

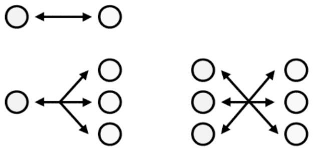
- XML library in Processing
  - Look at the examples
    - Google Geocode API

- Network organization



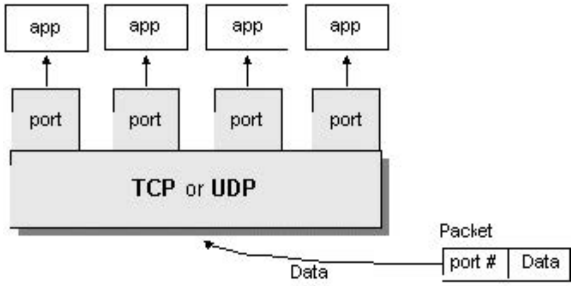
- Client-server
- Peer-to-peer

Communication Patterns (1)



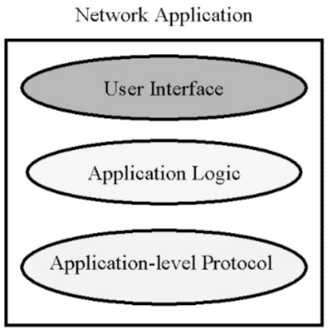
Unit 11: Networking

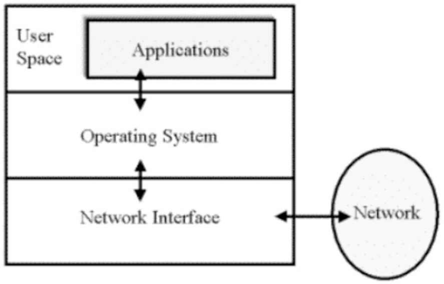
- Network organization
  - Machines on the the network need to be **identifiable** to each other and themselves
  - Machines need to know **how to connect** with one another
  - Machines need to know **what protocol** the other machines are using
- Network identification
  - The name or address of the host machine
  - The identity of the receiving process on the destination host
    - Receiving process = application
- Protocols
  - The types of messages exchanged: e.g., request and response messages
  - The syntax of various message types, that is, the fields in the message and how the fields are delineated
  - The semantics of the fields – the meaning of the information in the fields
  - Rules for determining when and how a process sends messages and responds to messages



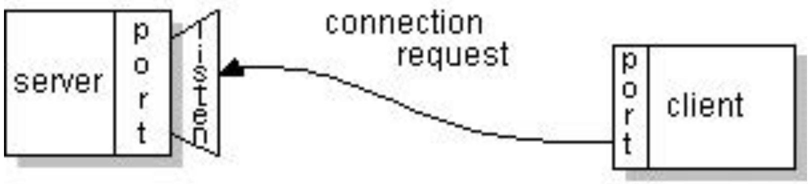
Unit 11: Networking (cont)

- Network application
  - Context / definition
  - Structure
  - Examples

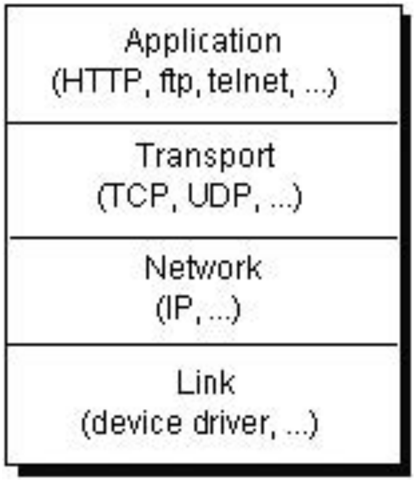




- Socket
- Packet



- Characteristics of server
- Characteristics of client
- Network layering


Application
(HTTP, ftp, telnet, ...)

Transport
(TCP, UDP, ...)

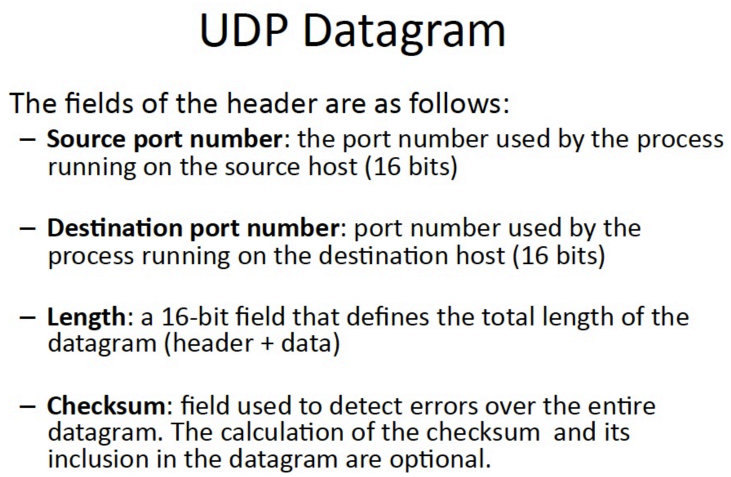
Network
(IP, ...)

Link
(device driver, ...)

Unit 12: Network Programming

- Network protocols: TCP and UDP
  - Characteristics
  - Advantages, drawbacks
  - Services offered
  - UDP datagram: header, data
  - TCP: virtual channel

- UDP and TCP Programming
  - Explain sample code



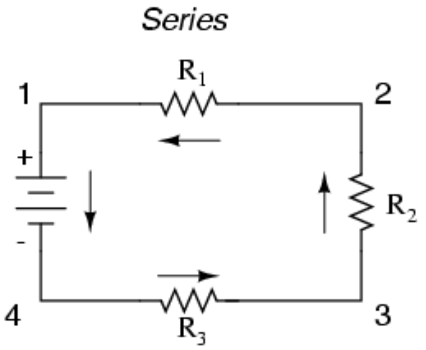
UDP Datagram

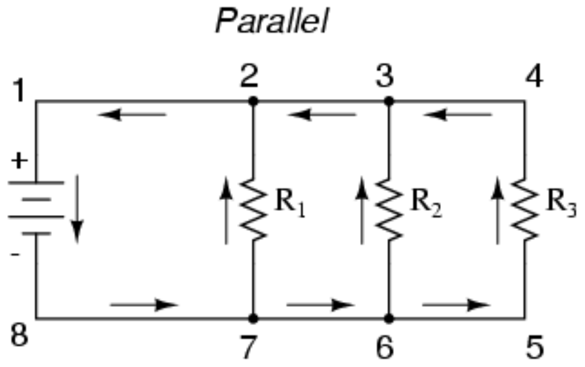
The fields of the header are as follows:

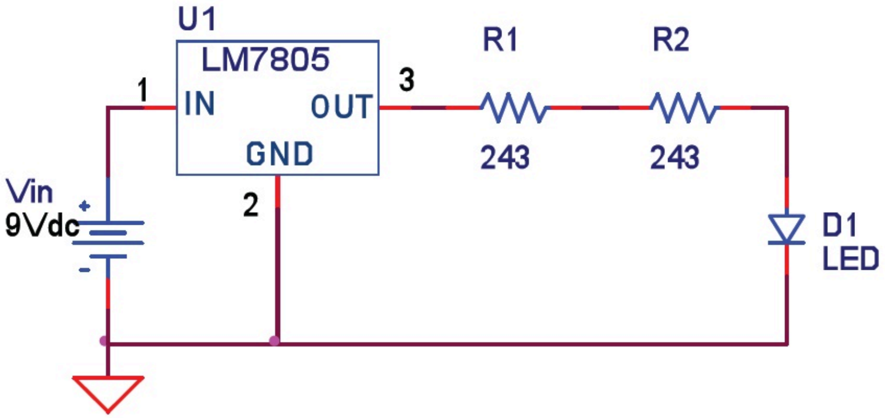
- **Source port number**: the port number used by the process running on the source host (16 bits)
- **Destination port number**: port number used by the process running on the destination host (16 bits)
- **Length**: a 16-bit field that defines the total length of the datagram (header + data)
- **Checksum**: field used to detect errors over the entire datagram. The calculation of the checksum and its inclusion in the datagram are optional.

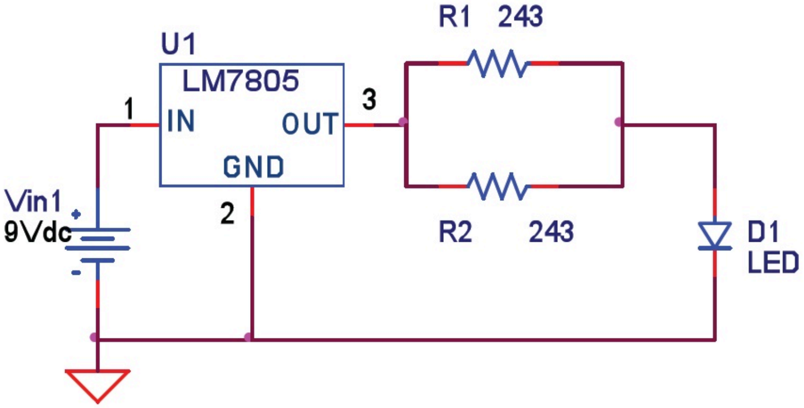
Workshops

- Review all circuits and codes from the workshops
- Know relationships for series and parallel circuits – what happened to the current / brightness of LED?






U1
LM7805
1
IN
OUT
3
GND
2
R1
R2
243
243
Vin
9Vdc
+
-
D1
LED


U1
LM7805
1
IN
OUT
3
GND
2
R1 243
R2 243
Vin1
9Vdc
+
-
D1
LED

Thank you

Questions?