Introductory Concepts and Tools
Computer Science 240

Laboratory 1

• Administrivia
• Lab Environment
• Basic Electronics (Ohm’s law, transistors, logic gates)
• Linux (open source UNIX operating system)
• C (language) and Emacs (editor for creating programs)
• Bitbucket and Mercurial (source control applications to manage and share your work)
Lab Environment

• All lab exercises and reports will be Google Docs, and should be shared with lab partner and the instructor
• Bring a laptop to lab if you have it (helpful to use a second computer for the lab report)
• From lab machine booted to Linux, you can enter Linux commands using a terminal/shell
• You can also use a terminal from either Mac (Terminal) or PC (PuTTY) to open a remote connection to a Linux machine for command-line entry

NOTE: for some exercises and assignments, you will be required to use the lab machines to compile and run your programs
Basic Concepts of Electricity

Electricity = **the movement of electrons** in a material

Materials tend to have a net negative or positive charge

Difference of charge between two points = **potential difference** (V)

Rate at which electrons flow through = **current** (A).

Ease of conduction, or current flow = **resistance** (Ω)
Ohm's Law, \( V = IR \).

Open circuit = no current
Short circuit = infinite current, since $V/0 = \text{infinite current}$:

Infinite current swiftly results in the destruction of the circuit!
### Basic Gate Symbols

- **NOT**
- **NAND**
- **NOR**
- **AND**
- **OR**

### Notation and Truth Tables for Basic Logic Gates

<table>
<thead>
<tr>
<th></th>
<th>NOT</th>
<th>NAND</th>
<th>NOR</th>
<th>AND</th>
<th>OR</th>
</tr>
</thead>
<tbody>
<tr>
<td>F =</td>
<td>(A')</td>
<td>((AB)')</td>
<td>((A+B)')</td>
<td>(AB)</td>
<td>(A + B)</td>
</tr>
<tr>
<td>A F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 1</td>
<td>0 0 1</td>
<td>0 0 1</td>
<td>0 0 0</td>
<td>0 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td>1 0</td>
<td>0 1 1</td>
<td>0 1 0</td>
<td>0 1 0</td>
<td>0 1 0</td>
<td>0 1 1</td>
</tr>
<tr>
<td></td>
<td>1 0 1</td>
<td>1 0 0</td>
<td>1 0 0</td>
<td>1 0 0</td>
<td>1 0 1</td>
</tr>
<tr>
<td></td>
<td>1 1 0</td>
<td>1 1 0</td>
<td>1 1 1</td>
<td>1 1 1</td>
<td>1 1 1</td>
</tr>
</tbody>
</table>
Basic Gates are built using Transistors

You have seen the circuits for NOT and NAND in lecture:

**NOT** – 1 transistor

**NAND** - 2 transistors

**AND** – uses 3 transistors (send the output of a NAND through another transistor acting as a NOT gate to complement the result):
Similarly, these are the transistor circuits for a NOR and OR gate:

**NOR** – 2 transistors

**OR** – 3 transistors
Linux

- Filesystem

- `/` (root directory)
- `./` (working directory)
- `../` (up one levels from working directory)
- `pwd` (print working directory)
- `cd` (change working directory)
- `ls` (list files and folders)
- `ls -l` (list files and folders with more information)
- `ls -a` (list all files and folders)
- `mv` (rename)
- `cp` (copy)
- `rm` (remove)
- `mkdir` (create directory)
- `rmdir` (remove directory)
- `diff` (difference between two files)
- `grep` (search)
- `echo` (echo to terminal)
- `cat` (concatenate)
- `>` (redirect)
- `gcc -Wall -g --std=c99 -o flnm flnm.c` (compile C program)
- `make` (compile using Makefile in working directory)
- `./flnm` (run executable program named flnm)
C programming language

- Basic syntax quite similar to Java and Python
- Some key differences:
  - No objects
  - Everything is a function
  - Begin execution at `main()`
  - Does not have a `boolean` data type

```c
/* CS 240: A simple first C program. */

/* Import definitions of standard library functions. */
#include <stdlib.h>

/* Import definitions of standard library input and output functions. */
#include <stdio.h>

/* The main function is called when the program is executed. Its return value is the exit status of the program. (0 = success, anything else = error) */

int main(int argc, char** argv) {
  // Print "Hello, Jean!" to standard output.
  printf("Hello, Jean!\n");
  // Exit with success.
  return 0;
}

- printf and scanf (formatted I/O)

  // Prompt for and read in an integer variable
  int x;
  printf("Enter a positive integer: ");
  scanf("%d", &x);
  printf("The value is = %d", x);
```
Bitbucket/Mercurial

Manage course materials and student individual/team work. Focus on individual workflow for today.

• Bitbucket (cloud/server):
  o fork repositoryname
  o change permissions/share using menu interface

• Mercurial (local/client) command-line interface:
  o \texttt{hg help}
  o \texttt{hg clone bitbucketrepository}
  o \texttt{hg add}
  o \texttt{hg commit}
  o \texttt{hg status}
  o \texttt{hg push}
  o \texttt{hg log}
  o \texttt{hg serve}
  o \texttt{hg revert}
  o \texttt{hg rename}
  o \texttt{hg mv}
  o \texttt{hg pull}
  o \texttt{hg push}
  o \texttt{hg update}
  o \texttt{hg incoming}
  o \texttt{hg merge}
  o \texttt{hg resolve}