Lecture I: Introduction and Overview

Xuan Guo

CSC 3210 Computer Organization and Programming
Georgia State University

January 13, 2015
Plan for the lecture:

- Textbook
- Syllabus
- What is about CSC 3210

Chapter 1:

- Hand-programmable calculator
• Required Prerequisites
  – CSc 2010 Intro to CSc
  – CSc 2310 Java Programming
  – CSc 2510 Discrete Math
• 2 absences allowed – otherwise could be dropped
• No make ups for quizzes and rarely for exams

• Required Textbook

![Image of textbook cover]
• About 6 programming assignments
• Penalty for late submissions is 20%
• Must be your own work

Class Policies

• No cell phones out during class
• One warning and then point deductions afterwards – no warning for exams/quizzes
Grading Policies and Exams

<table>
<thead>
<tr>
<th>Component</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Midterm exam</td>
<td>20%</td>
</tr>
<tr>
<td>Final exam</td>
<td>30%</td>
</tr>
<tr>
<td>Quizzes</td>
<td>20%</td>
</tr>
<tr>
<td>Assignments</td>
<td>20%</td>
</tr>
<tr>
<td>Attendance</td>
<td>10%</td>
</tr>
</tbody>
</table>

• All re-grading requests must be made within 2 classes from returned work
Why learn Assembly Language?

- One-to-one correspondence between the language and the architecture's machine code instructions.
- Specific to a particular computer architecture
- 10110000 01100001
- B0 61
- 10110 (x86/IA-32 processor to move)
- The identifier for the AL register is 000
- **MOV AL, 61h**
Why learn Assembly Language?

• Knowledge of assembly language is essential to understanding how computers are designed
• Provides the ability to optimize the code
• First word – speed
  – Gaming
  – Simulations
  – Medical equipment
• Second word – security
  – Knowing how to hack code
Chapter 1  Introduction to Computer Architecture
Chapter 2  SPARC Architecture
Chapter 3  Digital Logic and Binary Numbers
Chapter 4  Binary Arithmetic (excluding 4.10-4.11)
Chapter 5  The Stack
Chapter 7  Subroutines
Chapter 8  Machine Instructions
Chapter 10  Introduction to Input/Output
Architecture of Programmable Calculator

Memory

Control Unit

Arithmetic Logic Unit

Accumulator

Input

Output
Programmable Calculators

- Numeric keyboard and function keys
- Single register – accumulator
- Arithmetic logic unit – for computations
- Stack provides memory
  - LIFO data structure
  - Pushing/popping operations
  - No addresses for the memory cells
Emulator available at www.hp15c.com
Postfix vs. Infix

- Postfix notation
  - Operators follow operands
  - $3 \ 4 \ +$
  - Uses the stack to save memory
  - No need for parenthesis

- Infix notation
  - Operators are between operands
  - $3 + 4$
  - Need to specify order of operations -- parenthesis
An example for postfix

\[
\frac{(A + B) \times (C - D)}{E + \frac{F}{G}}
\]

\[
A \ B + C \ D - \times E \ F \ G \div + \div
\]
Another example

\[ y = \frac{(x-1)(x-7)}{(x-11)} \]

\[
\begin{align*}
(10 - 1) &= 9 \\
(10 - 7) &= 3 \\
(9 \times 3) &= 27 \\
(10 - 11) &= -1 \\
\frac{27}{(-1)} &= -27
\end{align*}
\]

10 enter 1 – 10 enter 7 – * 10 enter 11 – /

Figure 1.2: The Evaluation of \( y = \frac{(10 - 1)(10 - 7)}{(10 - 11)} \)
• Why would we want to use registers?
• Registers are provided to hold constants
• 10 registers – named r0 thru r9
• 3.14159 sto 0 – stores value in r0 and leaves it on top of stack
• rcl 0 -- copy contents of r0 to top of stack
• Must specify register name
In program mode, keystrokes not executed, code for each key is stored in memory

Memory has an address and holds data

Principal key designation

Function keys

Machine language – codes for keystrokes

Central processing unit

Program counter – holds address of next instruction to be executed
Figure 1.1: The HP-15C Programmable Calculator
Xuan Guo

Lecture I: Introduction and Overview

\[ y = \frac{(x - 1)(x - 7)}{(x - 11)} \]

3.14159 sto 0
Place the constant on the stack
and store value in register r0

1 –
Push 1, subtract, now TOP=2.14159

rcl 0
Place value of r0 on stack,
TOP=3.14159

7 –
Push 7, subtract, TOP= -3.8584

* 
Multiply, TOP = -8.2631

rcl 0
Place value of r0 on stack,
TOP=3.14159

11 –
Push 11, subtract, TOP = -7.8584

/ 
Divide, TOP = 1.0515
• Memory used to store program
• Memory is addressed
• May compute memory addresses – unlike registers
• Registers may be selected – not indexed
Figure 1.1: The HP-15C Programmable Calculator
• Program stored using machine language – key codes of the calculator

• Central processing unit (CPU) executes the codes

• Program counter (PC) holds address of next instruction to be executed
<table>
<thead>
<tr>
<th>Address</th>
<th>M/C code</th>
<th>Keystrokes</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>000 – 001</td>
<td>44 0</td>
<td>sto 0</td>
<td>Store in register 0</td>
</tr>
<tr>
<td>002</td>
<td>1</td>
<td>1</td>
<td>Enter 1</td>
</tr>
<tr>
<td>003</td>
<td>30</td>
<td>–</td>
<td>Subtract</td>
</tr>
<tr>
<td>004 - 005</td>
<td>45 0</td>
<td>rcl 0</td>
<td>Register 0 to stack</td>
</tr>
<tr>
<td>006</td>
<td>7</td>
<td>7</td>
<td>Enter 7</td>
</tr>
<tr>
<td>007</td>
<td>30</td>
<td>–</td>
<td>Subtract</td>
</tr>
<tr>
<td>008</td>
<td>20</td>
<td>*</td>
<td>Multiply</td>
</tr>
<tr>
<td>009 - 010</td>
<td>45 0</td>
<td>rcl 0</td>
<td>Register 0 to stack</td>
</tr>
<tr>
<td>011</td>
<td>1</td>
<td>1</td>
<td>Enter 1</td>
</tr>
<tr>
<td>012</td>
<td>1</td>
<td>1</td>
<td>Make it 11</td>
</tr>
<tr>
<td>013</td>
<td>30</td>
<td>–</td>
<td>Subtract</td>
</tr>
<tr>
<td>014</td>
<td>10</td>
<td>/</td>
<td>Divide</td>
</tr>
<tr>
<td>015 - 016</td>
<td>43 32</td>
<td>g Rtn</td>
<td>Return to calculator mode</td>
</tr>
</tbody>
</table>