I am developing, piloting, and sharing research-based active learning materials for Computer Architecture and Organization.

Activities

<table>
<thead>
<tr>
<th>Name</th>
<th>Learning objectives</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bits and numbers</td>
<td>• Translate integers and fixed point between number bases</td>
</tr>
<tr>
<td></td>
<td>• Express positive and negative integers in two’s complement</td>
</tr>
<tr>
<td></td>
<td>• Identify the largest and smallest integers representable using N bits</td>
</tr>
<tr>
<td>Stored Programs</td>
<td>• Discuss the correspondence between assembly language instructions and binary machine code</td>
</tr>
<tr>
<td></td>
<td>• Read the assembly language/machine code documentation</td>
</tr>
<tr>
<td></td>
<td>• Translate arithmetic and load/store instructions between assembly language and machine code</td>
</tr>
<tr>
<td></td>
<td>• Translate labels to addresses for branch and jump instructions</td>
</tr>
<tr>
<td>Dynamic memory and strings</td>
<td>• Describe the purpose of data segment, text segment, stack, and heap memory</td>
</tr>
<tr>
<td></td>
<td>• Draw a diagram of memory contents for an executing program</td>
</tr>
<tr>
<td></td>
<td>• Translate high level code that uses dynamically allocated objects</td>
</tr>
<tr>
<td>Adders, shifters, and multipliers</td>
<td>• Compare the delay of various implementations of arithmetic circuits</td>
</tr>
<tr>
<td></td>
<td>• Build variable bit shifters using various approaches</td>
</tr>
<tr>
<td></td>
<td>• Build a multiplier from shifters and adders</td>
</tr>
<tr>
<td>Memories and the Add Instruction</td>
<td>• Build an addressable RAM from registers or smaller memories with fewer ports</td>
</tr>
<tr>
<td></td>
<td>• Build a simple datapath that can execute a single instruction, and program it</td>
</tr>
<tr>
<td></td>
<td>• Modify the datapath to support a second instruction, and program it</td>
</tr>
<tr>
<td>Engineering Digital Systems</td>
<td>• Calculate the delay of the critical path in a synchronous circuit, and use it to determine minimum clock period and throughput</td>
</tr>
<tr>
<td></td>
<td>• Plot and interpret a Pareto optimal curve of delay vs area</td>
</tr>
<tr>
<td></td>
<td>• Describe the advantages and limitations of pipelining</td>
</tr>
</tbody>
</table>

Additional activities under development for a full course on Computer Organization and Architecture:
- Bitwise operations
- Introduction to assembly programming
- Procedure calls
- Compile, assemble, link, load
- Introduction to combinational logic
- Introduction to sequential logic
- MIPS datapath and control
- Pipelining and hazards
- Cache analysis with a simulator

Example activity

```
Input:
Register A: 2
Register B: 1

CLK: 0 1 2 3 4 5 6 7

Q outputs: 0 1 2 3 4 5 6 7

Draw the value of the Q outputs of the registers given the following Input signal.
```

The **width** (W) is the number of bits per element and the **length** (L) is the number of elements... A **write port** consists of two inputs, the WriteData and the WriteAddress...

```
WriteData: 0 1 0 1 0 1 0 1
WriteAddress: 0 1 0 1 0 1 0 1
```

Complete the drawing for a W=8, L=4, 1 write port memory. Label each entry with its bit width. (ignore right side of registers).

Motivation

- Evidence supports that active learning in CS increases student performance and engagement
- There are insufficient materials to support a Computer Organization and Architecture course
- Access to materials is a common barrier to using research-based instruction strategies in Computer Engineering and Computer Science

Approach

- Created POGIL activities for Computer Architecture and Organization (AR in ACM/IEEE curricula 2013)
- Piloted the activities with 2 semesters of students
- Activities to be peer reviewed and shared on CSPOGIL.org

POGIL

- Process-oriented guided inquiry learning (POGIL) is based on a theory of instruction: learning cycle of exploration, concept introduction, and application
- Objectives of a POGIL activity are that students acquire conceptual knowledge and develop process skills, such as interpretation and application of knowledge
- Researchers found using POGIL or POGIL-like instruction improved student performance in CS1 compared to using traditional instruction
- CS-POGIL project has created and published over 200 POGIL activities for CS

Pilots

- Computer Organization, required intermediate level course, pre-requisite for OS, networks, and security. First and only required course dedicated to assembly language or digital logic.
- Mostly Junior-level Computer Science majors, small number of computing-related majors
- 6 activities with 36 students in Summer 2017 (1 instructor and 1 assistant) and 70 students in Fall 2017 (1 instructor and 2 assistants)
- Facilitated in student-centered TILE (Spaces to Transform, Interact, Learn, Engage) classrooms

Observations

Authorship
- Piloting reveals opportunities to elaborate or decompose exploration questions
- mark divergent questions to help teams manage time

Facilitation
- there are tradeoffs for synchronous and asynchronous groups
- groups that are together a few weeks are more efficient and have the opportunity to improve process skills
- it is feasible to give frequent formative assessment and feedback with 5-minute group exit slips
- incentivizing individuals is a challenge but can be done with different deliverables per role
- while preparation is possibly less important than in a flipped class, it helps groups make timely progress

Acknowledgements

- Clif Kusmarau – guidance on CS-POGIL and authorship
- Renée Cole - POGIL facilitation materials and observations

Funded by the SIGCSE Special Projects Grant program