von Neumann Computer

Key idea: Memory holds not only data, but coded instructions that make up a program

INSTRUCTIONS coded as binary data

PROGRAM COUNTER or PC:
Address of next instruction to be executed

logic to translate instructions into control signal for path

Fetch/Execute loop:
Fetch Mem[PC]
PC = PC+4
Execute fetched instruction
Repeat

Binary layout of the two Beta instruction formats:

Types of Beta assembly instructions:
**Register Arithmetic:** ADD(R1,R2,R3) R3 = R1 + R2
LD(R1, c, R2) Mem[R2+c] = R1
CMPLT(R1,R2,R3) R3 = 1 if R1 < R2; 0 o.w.
BEQ(R1, 0x20, R2) R3 = PC+4
PC = PC+4+4*0x20 if R2;
PC = PC+4 o.w.

More on memory:
LD(Ra, literal, Rc): “load value at address Ra+literal into register Rc”
ST(Rc, literal, Ra): “store value in register Rc into address literal+Ra”

```
ans = 0x100
R0 = 0 (the value at address 0x100)
```
Programmable Machines

Problem 1

F Build a controller that will cause the circuit above to execute the following algorithm:

while (a != b)
    if (a > b) a = a - b;
    else b = b - a;

### Machine Language:

<table>
<thead>
<tr>
<th>Reg A &lt;- A</th>
<th>DRA</th>
<th>DRB</th>
<th>DRALU</th>
<th>LDA</th>
<th>LDB</th>
<th>FN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reg B &lt;- B</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>X</td>
</tr>
<tr>
<td>A==B?</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>X</td>
</tr>
<tr>
<td>(if A&gt;B) Reg A &lt;- Reg A – Reg B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(else) Reg B &lt;- Reg B – Reg A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(loop up to A=B?)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(if A==B) do nothing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Enjoy Spring Break!