CSC2510 - Computer Organization
Lecture 13: Revision
Processor Unit
Philip Leong

Execution of Add (R3), R1
- Fetch the instruction
- Fetch the first operand (R3)
- Perform the addition
- Load result to R1

- What should MDRin be?

Execution
- Step 1: PC loaded into MAR, read request to memory, MUX gives 4, added to B (PC) and stored in Z
- Step 2: Z moved to PC while waiting for memory
- Step 3: Word fetched from memory and loaded into IR
- Step 4: figure out what the instruction should do and set control circuitry for steps 4-7. R3 transferred to MAR and memory read operation initiated
- Step 5: contents of R1 moved to Y
- Step 6: read operation completed and is in MDR as well as B input of ALU. Select Y as second input of ALU and add performed
- Step 7: result is transferred to R1, End causes a goto step 1

Which steps are the instruction fetch?

Branch Instructions
- Steps 1-3, instruction fetch
- Step 4: add the offset to the PC
- Step 5: update the PC

Step Action
1. PCout, MAR in, Read, Select4, Add, Zin
2. Zout, PCin, Yin, WMFC
3. MDRout, IRin
4. Offset-field-of-IRout, Add, Zin
5. Zout, PCin, End

Now do you understand why the branch offset is calculated from the next address to be executed?

For conditional e.g. branch < 0, step 4 is replaced with

Offset-field-of-IRout, Add, Zin, if N=0 then End

Multiple Buses
- One disadvantage of our single bus scheme is that only one data item can be transferred over the bus per cycle
- A solution is multiple internal buses
- All registers combined into a register file with 3 ports
  - Why are there 2 outputs?
  - What is the input for?
  - What does 3 port mean?
- Buses A and B allow simultaneous transfer of the two operands for the ALU
  - ALU is able to just pass one of its operands to R e.g. R=A
- Incrementer unit computes PC+4, means we don’t need the ALU for this
  - ALU still has a 4 input for other instructions such as postincrement
Three bus datapath

• What does this do?
• (WMFC means wait for memory function completed)
• What are the advantages and disadvantages over a single bus?

Step Action
1. PC_{out}, R=B, MAR_{in}, Read, IncPC
2. WMFC
3. MDR_{out}, R=B, IR_{in}
4. R4outA, R5outB, SelectA, Add, R6, End

Hardwired Control

• How do we generate the control signals?
  – Hardwired control
  – Microprogrammed control
• A hardwired control is called a finite state machine
  – Sequences using a counter and produces control signals at the right time
  – Control signals are functions of the IR, external inputs and condition codes
  – Can you give an example for each?

Microprogrammed Control

• The control signals are stored in a memory as sequences of control words which are the individual bits of the control signals
• Microinstructions are executed in a manner similar to machine code

Microprogrammed Control Unit

<table>
<thead>
<tr>
<th>Microinstruction</th>
<th>R_{0}</th>
<th>R_{1}</th>
<th>R_{2}</th>
<th>R_{3}</th>
<th>R_{4}</th>
<th>R_{5}</th>
<th>Select</th>
<th>Add</th>
<th>MDR_{out}</th>
<th>ALU_{out}</th>
<th>WMFC</th>
<th>End</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Scheme to allow Conditional Branching

• “Starting and branch address generator”
  – Loads new address into uPC when instructed
  – Has condition codes and external inputs which can affect uPC
• uPC incremented every cycle except
  – When new instruction loaded into IR, uPC loaded with starting address of the microroutine
  – For taken branches, uPC updated to branch address
  – For End microinstruction, uPC set to 0

Microprogrammed Control Unit

Address Microinstruction

<table>
<thead>
<tr>
<th>Address</th>
<th>PC_{out}, MAR_{in}, Read, IncPC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>PC_{out}, MAR_{in}, Read, IncPC</td>
</tr>
<tr>
<td>1</td>
<td>Z_{out}, PC_{out}, Y_{out}, WMFC</td>
</tr>
<tr>
<td>2</td>
<td>MDR_{ad} - RI_{in}</td>
</tr>
<tr>
<td>3</td>
<td>Branch to starting address of appropriate microroutine</td>
</tr>
<tr>
<td></td>
<td>25 IF N=0, then branch to microinstruction 0</td>
</tr>
<tr>
<td></td>
<td>26 Offset-field-of-IR_{out}, SelectY, Add, Z_{in}</td>
</tr>
<tr>
<td></td>
<td>27 Z_{out}, PC_{out}, End</td>
</tr>
</tbody>
</table>

Figure 7.16: Basic organization of a microprogrammed control unit.

Figure 7.17: Microroutine for the instruction Branch<0.