Lecture 6: Addressing Modes

Overview

- Addressing Modes
- AVR Instruction Examples

Operands

- Instructions need to specify where to get operands from
- Some possibilities
  - Value is in instruction
  - Value is in a register
    - Register number is in the instruction
  - Value is in memory
    - Address is in instruction
    - Address is in a register
      - Register number is in the instruction
      - Address is register value plus some offset
        - Register number is in the instruction
        - Offset is in the instruction (or in a register)
- These are called addressing modes

Immediate Addressing

- Not really an addressing mode – since there is no address
- Instruction doesn’t have address of operand – it has the value itself
  - i.e. the operand is immediately available
- Limits to the size of the operand you can fit in an instruction (especially in RISC machines which have instruction word size = data word size)
Immediate Addressing Examples

- **AVR**
  - **SUBI**
    - 0101  KKKK  Rd  KKKK
  - **ADWI**
    - 1001 0110  KK  Rd  KKKK

- **Pentium**
  - **mov ebx, KKKK**
    - 1011 1  011
    - KKKK  KKKK  KKKK  KKKK
    - KKKK  KKKK  KKKK  KKKK

Direct Addressing

- Address of the memory operand is in the instruction
- Useful for global variables (accessible from all subroutines)
- AVR Datasheet calls this “Data Direct Addressing” and I/O Direct Addressing.

Direct Addressing Examples

- **AVR**
  - **STS**
    - 1001001  Rd  0000
    - KKKK  KKKK  KKKK  KKKK

- **Pentium**
  - **mov eax,[kk]**
    - 1010 0001
    - KKKK  KKKK  KKKK  KKKK
    - KKKK  KKKK  KKKK  KKKK

Register Direct Addressing

- Register numbers are contained in the instruction
- Data in the registers.
- Fastest mode and most common mode in RISC
- Example: **ADD**

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AVR</strong></td>
<td></td>
<td>0000 11</td>
<td>r</td>
<td>ddddd</td>
<td>r r r</td>
</tr>
<tr>
<td><strong>Pentium</strong></td>
<td>0000 0000</td>
<td>11</td>
<td>r r r</td>
<td>ddd</td>
<td></td>
</tr>
<tr>
<td><strong>Sparc</strong></td>
<td>10</td>
<td>rd</td>
<td>00 0000</td>
<td>rs1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td></td>
<td></td>
<td>rs2</td>
<td></td>
</tr>
</tbody>
</table>
Register Indirect Addressing

• Register number in instruction, as with register addressing
• However, contents of the register is used to address memory
  □ the register is used as a pointer
• AVR datasheet calls this “Data Indirect” addressing

Indexed Addressing

• Reference memory at a known offset from a register
• Two main uses:
  □ Register holds address of object; fixed offset indexes into the object (structure, array, etc)
  □ Address of object is constant; register has index into the object
• AVR datasheet calls this “Data Indirect with Displacement” addressing (fixed offset, not in register)

Register Indirect Addressing Example

• AVR
  LDD Rd, Y

| 1000000 | ddddd | 1000 |

Operation: Rd ← (Y)
Y: r29: r28

Indexed Addressing Examples

• AVR (data indirect with displacement)
• Only 6 bit displacement (q bits)
• Only Y or Z index registers
  □ determined by this bit

LDD Rd, Y+q

| 10 | q | 0 | qq | 0 | ddddd | 1 | qqq |

• Operation: Rd ← (Y + q)
**Auto Increment**

- Some architectures allow modification of the index register as a side effect of the addressing mode.
- Usually add or subtract a small constant, often equal to the operand size.
- Could happen before or after the index register is used in the address calculation.
- Most common are post increment and pre decrement.
  - AVR supports these:
    - “Data Indirect with Pre-decrement”
    - “Data Indirect with Post-increment”

**Auto Increment Examples**

- AVR
  
  LDD Rd, -Y

  | 1001000 | ddddd | 1010 |

  Operation: Y ← Y−1 Rd ← (Y)

  LDD Rd, Y+

  | 1001000 | ddddd | 1001 |

  Operation: Rd ← (Y) Y ← Y+1

**Code Memory Constant Addressing**

- AVR has separate data and instruction memories.
- Sometimes need to get data constants out of instruction memory (flash memory).
- Special instruction provided to do this (LPM)

  LPM Rd, Z

  | 1001000 | ddddd | 0100 |

  Operation: Rd ← (Z)

  Load a byte at the address contained in register Z (r30: r29)

**Branch Instructions**

- Specify where in the program to go next (i.e. change the program counter rather than just increment)
  - Program flow is no longer linear.
- Types of Branch Instructions
  - Unconditional – always do it
  - Conditional – do it if some condition is satisfied (e.g. check status register)
  - Jumps – no return
  - Subroutines (function calls) – can return
Branch Instruction Addressing Modes

• Direct addressing
  - Address to jump to included in the instruction
  - Not every AVR device support this (JMP and CALL instructions)

• Indirect addressing
  - Address to jump to is in a register
  - AVR examples: IJMP, ICALL

• Program Counter Relative addressing
  - AVR calls this “Relative Program Memory” addressing
  - Add a constant value to the Program Counter
  - AVR examples: RJMP, RCALL, conditional branch instructions…

Branch Instruction Addressing Modes: AVR Examples

• JMP k
  Operation: PC ← k (0 ≤ k ≤ 4M)
  
  | 1001 010 | kkkkk | 110 | k |
  | kkkk | kkkk | kkkk | kkkk |

• IJMP
  Operation: PC ← Z
  
  | 1001 0100 0000 1001 |

Branch Instruction Addressing Modes: AVR Examples

• BRGE k (signed)
  Operation: If Rd ≥ Rr then PC ← PC+k+1 else PC ← PC+1
  Operand: -64 ≤ k ≤ +63

  | 111101 | kkkkkkk | 100 |