The ATmega328
Instruction Set Architecture

- **ISA**: All of the *programmer-visible* components and operations of the computer
  - memory organization
    - address space -- how may locations can be addressed?
    - addressibility -- how many bits per location?
  - register set
    - how many? what size? how are they used?
  - instruction set
    - opcodes
    - data types
    - addressing modes

- ISA provides all information needed for someone who wants to write a program in *machine language* (or translate from a high-level language to machine language).
Memory and Registers

• Program Memory
  – address space: 32K (K=1024) locations (15-bit addresses)
  – addressability: 16 bits

• Data Memory
  – address space: 2K (K=1024) locations (11-bit addresses)
  – addressability: 8 bits

• Registers
  – temporary storage, accessed in a single machine cycle
    • accessing memory generally takes longer than a single cycle
  – 32 general-purpose registers: R0 – R31
    • each 8 bits wide
    • how many bits to uniquely identify a register?
  – other registers
    • not directly addressable, but used by (and affected by) instructions
    • PC, IR, etc.
Instruction Set

- **30 Opcodes**
  - **ALU** instructions: ADC, ADD, AND, NOP, CP, EOR, OR, MOV
  - **Immediate** instructions: CPI, ORI, ANDI, LDI
  - **Unary Logical** instructions: COM, NEG, ASR, LSR
  - **Load/Store** instructions: LDS, STS
  - **Branch** instructions: BRBS, BRBC (aka BRZS, BRZC, BRCS, BRCC)
    - Conditional branch based upon SREG bits ("Condition Codes")
  - **Input/Output** instructions: IN, OUT
  - **Call/Jump** instructions: CALL, JMP
  - **Return** instructions: RET, RETI
  - **Stack** instructions: PUSH, POP
  - **Relative Jump** instructions: RCALL, RJMP

- **Addressing Modes**
  - Specify how the location of an operand is resolved
  - non-memory addresses: *immediate, register direct, etc.*
  - memory addresses: *data direct, data indirect, data indirect with displacement, etc*
Control Instructions

- Used to alter the sequence of instructions (by changing the PC)

- **Conditional Branch**
  - branch is *taken* if a specified condition is true (BRBC, BRBS)
    - BRCC/BRC is alias of BRBC/BRBS with field “sss” being the SREG bit index of the condition code
    - signed offset is added to PC to yield new PC
  - else, the branch is *not taken*
    - PC is not changed, points to the next sequential instruction

- **Unconditional Branch (or Jump or Call)**
  - JMP, RJMP, CALL, RCALL
    - Relative forms add signed offset to PC to yield new PC
  - always change the PC (there is no condition)
Condition Codes

• Used by Conditional Branch instructions to determine whether a branch should be taken
• Located in the Status Register (SREG)
  – Indexed by bit position
    • C = 0, Z = 1, ... I = 7
    • The sss field of BRBC/BRBS specifies which bit is tested
• Are changed by certain instructions
  – One instruction can change many CC bits
  – Change depends upon
    • logic of the instruction
    • operands to the instruction
    • state of the datapath
Review the AVR Instruction-Set Reference