AVR Addressing Modes

AVR specific program and data addressing modes

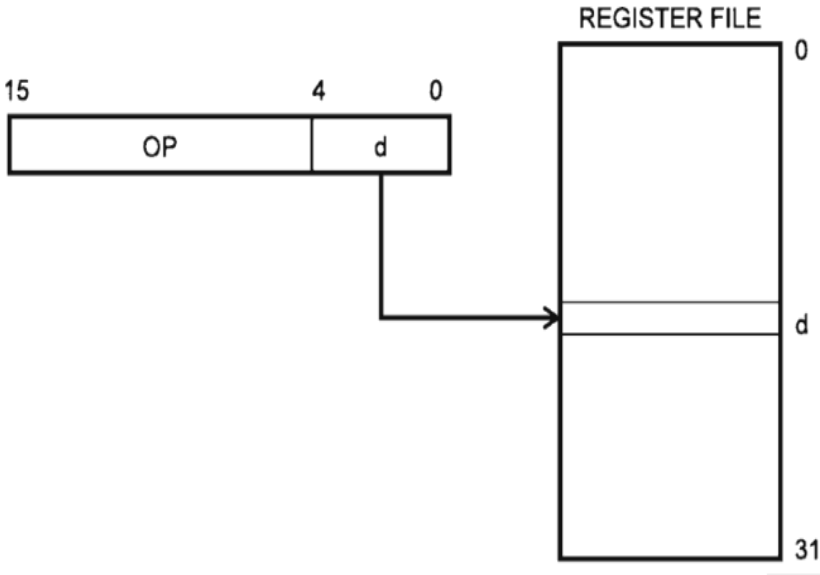
Credit to Dr. Robucci for slide information

Instructions and Addressing

- Instruction set
  - Decides what operations the processor can perform
  - Each instruction controls some part of the processor
- Addressing Modes
  - Instructions can be categorized based on how they *access* data and operate on it
  - AVR instructions fall in about 10 categories
- Each instruction has 2 parts
  - Opcode: Indicates to ALU what to do
  - Operands: Numbers on which the ALU operates

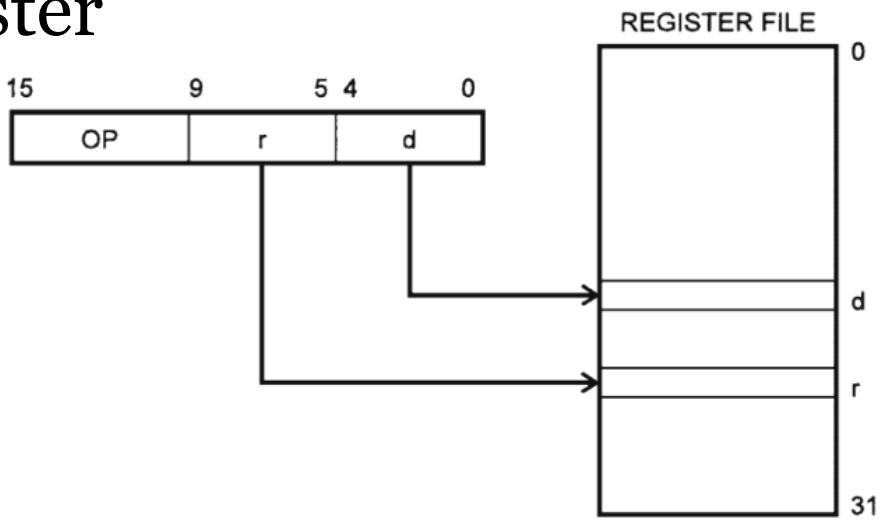
1: Register Direct (Single Reg)

- Can operate on any of the 32 registers
- Operation:
  - Read contents of register
  - Operate on contents
  - Store back in same register
- Example Commands:
  - INC R0, DEC R5, LSL R9



2: Register Direct (Two Reg)

- Two Registers
  - Rs: Source Register
  - Rd: Destination Register
- Reads two registers, operates on contents, stores result in destination register



- Examples:
  - Add R1,R3
  - Sub R5,R7

3: Immediate Mode

- Constant value is in the instruction
  - Stored with program code in memory
- Operates on register and immediate, stores value in register

- Examples:
  - SUBI R4, 8 // x = x-8
  - ADIW R26, 5 //R27:26 = R27:26 + 5

4: I/O Direct

- Instructions are used to access I/O space
  - Not extended I/O registers
- I/O Registers can be accessed using
  - In Rd, PORTADDRESS
  - Out PORTADDRESS, Rs
  - Rd, Rs: Any register from register file
  - PORTADDRESS: Any register from entire range of 0x00 to 0x3F

4: I/O Direct

- Unsigned char I = PINB;
- Unsigned char k = 54
- PORTB = k;

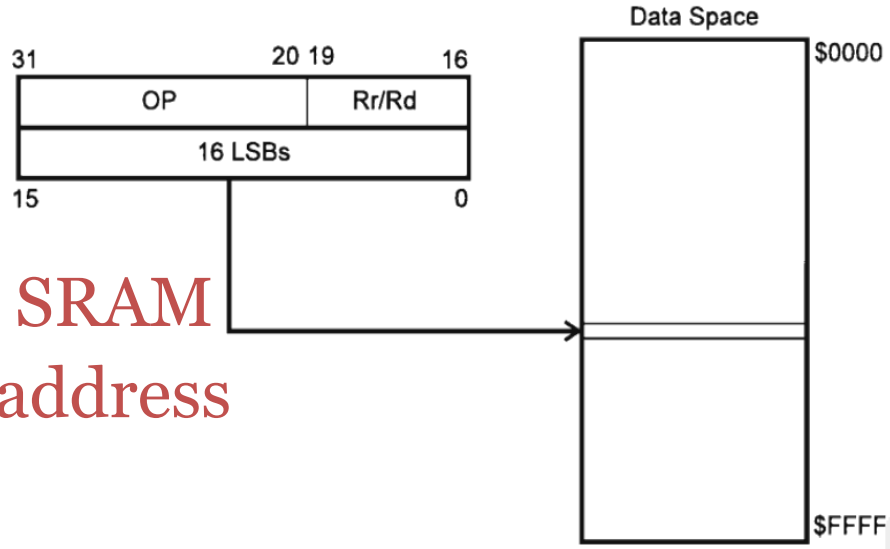
- IN R10, PINB
- OUT PORTB, R1

0x09 (0x29)	PIND
0x08 (0x28)	PORTC
0x07 (0x27)	DDRC
0x06 (0x26)	PINC
0x05 (0x25)	PORTB
0x04 (0x24)	DDRB
0x03 (0x23)	PINB
0x02 (0x22)	PORTA
0x01 (0x21)	DDRA
0x00 (0x20)	PINA

*page 376 of data sheet

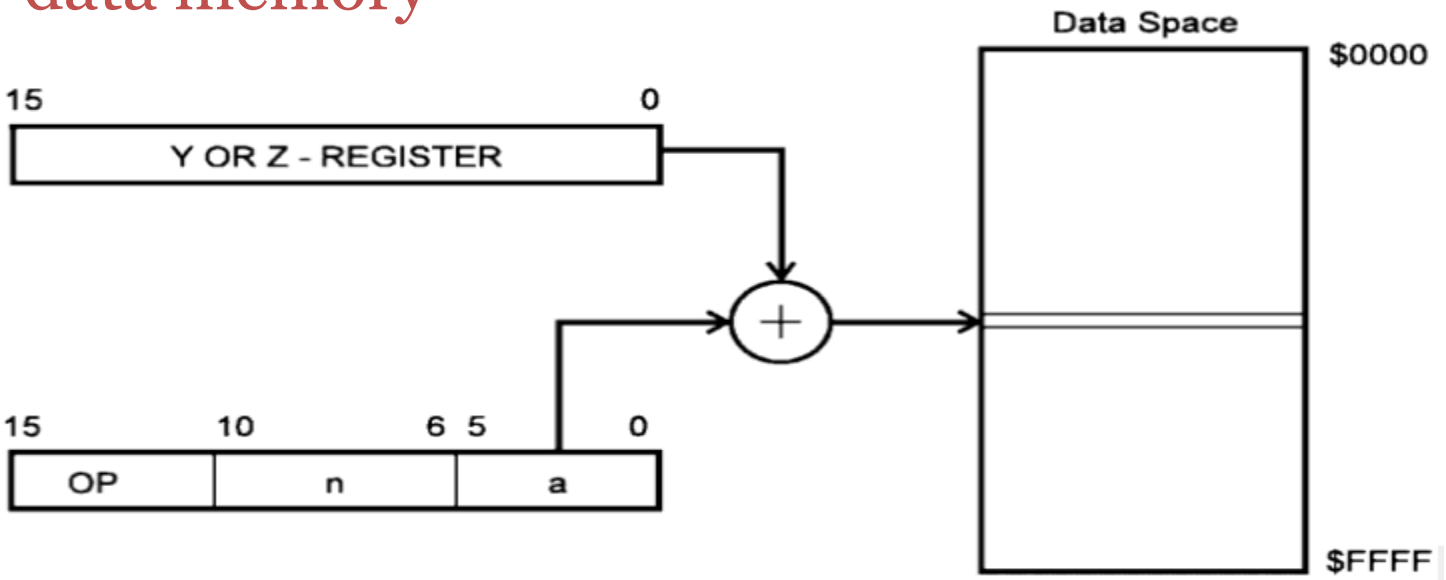
5: Data Direct

- Two-word instructions
  - One of the words is the address of the data memory space
  - What is the maximum data memory that can be addressed in this way?
- Examples:
  - STS K,Rs // Store Direct SRAM
  - LDS Rd, K // K is 16-bit address



6: Data Indirect

- Similar to data direct
  - One word each
  - Pointer register (x, y, or z) has base address of data memory



6: Data Indirect

- Examples
  - LD Rd, X // X = R27:R26
  - LD Rd, X+ // Rd ← X; X ← X+1 – indirect with // post decrement
  - LDD Rd, Y+q //Rd ← (Y+q) – indirect with //displacement
  - ST –Y, Rs // Y ← Y-1, Y ← Rs - Indirect with //pre-decrement

I/O Ports using Indirect

- Ports can be accessed using SRAM access commands
  - Add 0x20 to the port number
    - First 32 numbers are the registers
- Example
  - .DEF register = R16
  - LDI ZH, HIGH(PORTB+32)
  - LDI ZL, LOW(PORTB+32)
  - LD register, Z

Extended I/O

- For I/O Registers located in extended I/O:
  - Commands like “In/Out” cannot be used
  - Instead replaced with direct and indirect memory instructions
    - LDS and STS (Load and Store from SRAM) combined with SBR, CBR (set/clear bits in register)
    - Can also combine with SBRS, SBRC (skip if bit in register if set/clear)

7: Direct Program Addressing

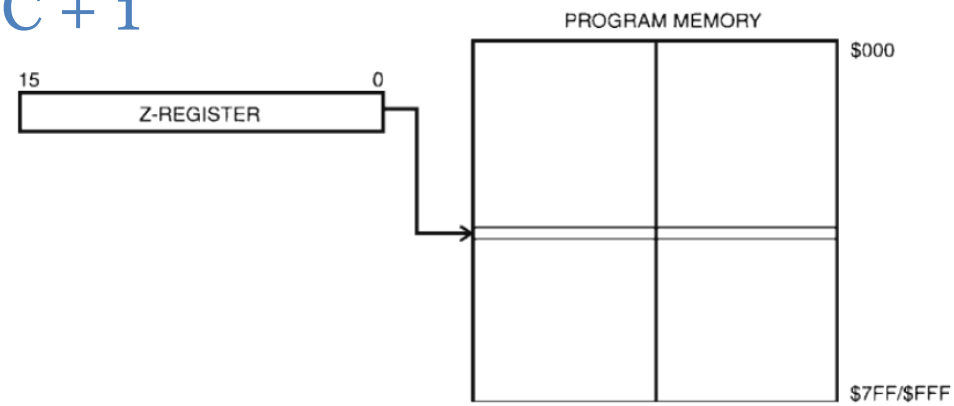
- Call K;
  - Direct Subroutine Call
  - PC ← k and STACK = PC + 1

8: Implicit Addressing

- CLC ;
  - Clear Carry C ← 0 // Implicit
- RET;
  - Subroutine Return
  - PC ← STACK

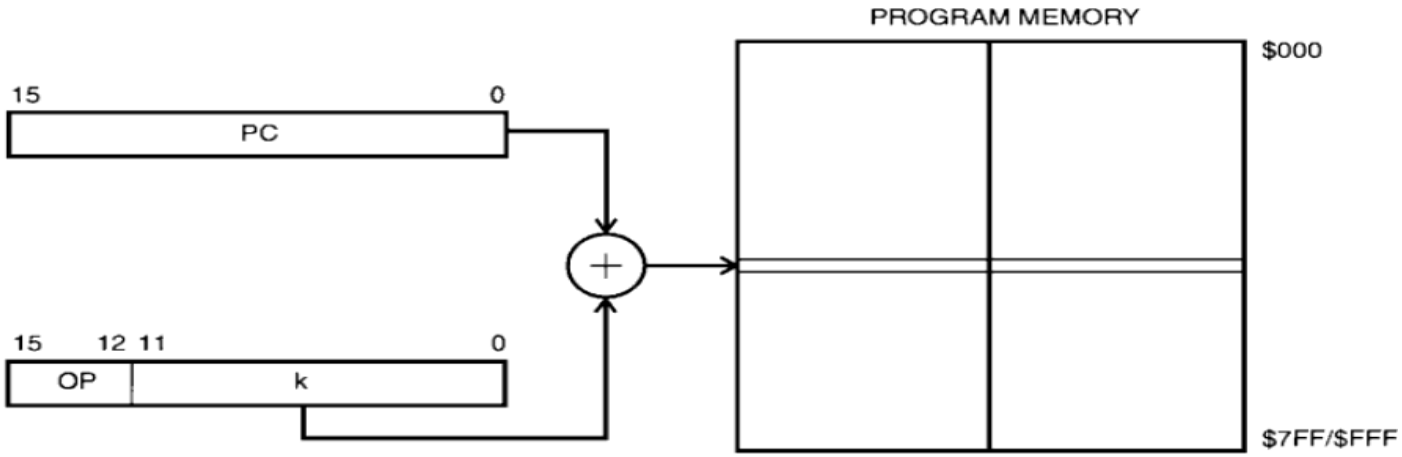
9: Indirect Program Addressing

- These instructions use Z register to point to program memory
  - IJMP ; // Indirect Jump to (Z)
    - PC ← Z
  - ICALL ; //Indirect Call to (Z)
    - PC ← Z, STACK = PC + 1



10: Relative Program Addressing

- Instructions of type RJMP and RCALL
  - Offset of +/- 2k to program counter is used



- RCALL k; //Relative Subroutine call
  - PC ← PC + k + 1, STACK = PC + 1
- RJMP k; //Relative Jump
  - PC ← PC + k + 1