Chapter 3: Addressing Modes, Instruction Mnemonics, Flags, and Jump Instructions

I. Elements of an Instruction
   A. Each CPU has its own instruction set.
   B. Operand Instructions contain explicit operand. An operand is either an address or a value.
      1. CUSP format: OOMAAA, where OO is 8-bit opcode, M is a 4-bit addressing modes, and AAA is a 12-bit addressing value.
         a) CUSP has 56 operand opcodes of the possible 255 (0xff cannot be used because it indicates an operate instruction).
         b) CUSP has 10 addressing modes of the possible 16.
   C. Operate Instructions have no explicit operand
      1. CUSP format: $ffOOO, where fff is the 0xFFF, and OOO is a 12-bit opcode. There are 38 operate opcodes of the possible 4096.

II. Addressing Modes
   A. Direct = $2
      1. Operand is the data in the memory location given by the addressing value.
      2. LDA $235 means ACC := Memory[$235].
   B. Immediate = $0
      1. Operand is the addressing value.
      2. LDA# $235 means ACC := $235.
      3. Limited to 12 bits so upper 12 bits of 24 bit registers are set to 0.
      4. Not available to opcodes that require an address in the addressing value, e.g., JMs and STA. If used with these it will cause an "immediate addressing fault" and CUSP halts.
      5. Reduces time and space of code.

III. Instruction Mnemonics
   A. For CUSP XXXC1 C2 value, where XXX is an opcode mnemonic C1 and C2 are addressing mode mnemonics, and value is the addressing value.
   B. Opcode mnemonic examples: LDA = $00, STA = $04, CMA = $20, JLT = $4A, HLT = $FFF, JSR = $41.
   C. For direct mode C1 = space, and C2 = space.
   D. For immediate mode C1 = # and C2 = space.
   E. Addressing value usually hexadecimal for addresses and decimal for data (immediate mode).

IV. Direct Addressing Instructions
   A. Flags
      1. Set as the result of instructions that change a register, except data transfers.
      2. Some instructions do not change any flags, e.g., LDA
      3. Some instructions will change only some of the flags. e.g. ADX just OV and EQ, not LT
      4. EQ is set if the operation is equal to zero.
      5. OV is set if there is a 2's complement overflow.
      6. LT is set if the operation results in a value is less than zero, i.e. sign bit is set.
   B. Jump Instructions
      1. Allow transfer of control, i.e. selection, loops, and subroutines.
      2. JMP = Unconditional jump.
      3. JEQ = Jump if equal; if EQ set.
      4. JNE = Jump if not equal; if EQ not set.
      5. JLE = Jump if less than; if LT set or EQ set.
      6. JGT = Jump if greater than; if LT not set and EQ not set.
      7. JGE = Jump if greater than or equal; if LT not set.
      8. JOV = Jump if overflow; if OV set.
      9. JNO = Jump if no overflow; if OV not set.
      10. Careful to select proper jump that corresponds to HLL construct.
          a) Instruction following a jump instruction may be if or else part of the HLL construct.
   C. Arithmetic and Logic Instructions
      1. Arithmetic Instructions
         a) ADA = add to the accumulator, $10
         b) SBA = subtract from the accumulator, $14
         c) MUL = multiply the accumulator, $18
d) DIV = divide the accumulator, $19
  e) MOD = modulo the accumulator, $1A
  f) INC = increment memory, $1B
  g) DEC = decrement memory, $1C
  h) NEG = negate memory using twos complement, $1D
  i) NEGA = negate accumulator using twos complement, $FFFF020

2. Logic Instructions
   a) AND = bitwise "and" the accumulator, $30.
      (1) Use for masks to clear bits
   b) OR = bitwise "or" the accumulator, $31
      (1) Use to set bits
   c) XOR = bitwise "exclusive or" the accumulator, $32
   d) COM = ones complement bits in memory, $33
   e) CMA = ones complement bits of the accumulator, $FFFF021

D. Floating Point Arithmetic
   1. Only implement in software in CUSP.
   2. Provided by the Floating Point Unit (FPU) as part of modern CPUs.

V. Immediate Addressing Instructions
   A. Faster than direct addressing because there is no memory fetch.
   B. Limited in values used by the size permitted by instruction.
      1. CUSP allows 12-bits.
         a) When storing a 12-bit in a 24-bit register, the upper twelve bits are set to zero, so we cannot have negative immediate values stored.
   C. Not available for many opcodes, e.g. STA

VI. Operate Instructions
   A. HLT Halt
   B. NEGA Negate accumulator using twos complement
   C. COMA Complement accumulator using simple ones complement
   D. NOP No operation
      1. Good for time delays.
      2. Breathing space for later inserted instructions so that jump addresses do not have to be changed.

VII. Five step process for writing a program
   A. Write the program in Pascal or C with an eye towards assembly language.
   B. Symbolic Form: build program using symbolic placeholders for addresses.
   C. Address Form: Assign an address to each instruction. Create symbol table.
   D. Substitution: Substitute addresses in place of symbolic placeholders.
   E. Machine Form: Convert mnemonics and addresses to machine code.