Design for a 16-bit CS-353 computer

16-bit word size (with two’s complement numbers & arithmetic)
32 registers = 5-bit register specs
4096 word RAM = 12-bit RAM addresses

3 bits for primary opcode:
+ 3 for operator selection + 2 x 5-bits for registers (binary arith / bool / rel operations)
+ 1 for option + 12-bits address / data (jumps and sets)
+ 3 for operator selection + 2 x 5-bits for amt / regs / etc. (miscellaneous operations)

Registers (and one single-bit flag):
- 26 general registers, named alphabetically (RA–RZ)
- arithmetic result / accumulator (AR)
- arithmetic auxiliary and logical result register (AX or LR)
- comparison result flag (CF—\textit{never directly specified by user in assembly})
- index register (IR)
- 2 data registers (D1, D2)
- program counter (PC)

- [000] END + 15 bits of nothing / ignored
- [001] numeric class + 3 bits arithmetic op choice:
  - ADD \( AR = R1 + R2 \)
  - ADX \( AX = R1 + R2 \)
  - SUB \( AR = R1 - R2 \)
  - SBX \( AX = R1 - R2 \)
  - MUL \( AR = R1 \times R2 \)
  - DVM (divmod) \( AR = R1 / R2; \ AX = R1 \% R2 \)
  - SPR (sumprod) \( AR = R1 + AX \times R2 \)
  - MNX (minmax) \( AR = R1 \text{ min } R2; \ AX = R1 \text{ max } R2 \)
  - LR = R1 \ op \ R2
- [011] comparison class + 3 bits rel op ([A]LT, LEQ, [A]GT, GEQ, EQU, NEQ, CLZ, FRZ)
  - CF = R1 \ op \ R2
- [100] MOV + 3 bits (8 combos of direct / indirect / indexed (no x/x)) + 2 x 5-bit register specs
- [101] JMP / JIF (1 bit uncond/cond) + 12 bits of target address
- [110] SET + 1 bit data register choice + 12 bits of address, numeric or character data
• [111] miscellaneous class (amt spec is 5-bit signed magnitude, offset as in PC-231):
  • INC R1, amt (±4 bits) ; increment register content by some amount
  • SHF R1, amt (±4 bits) ; shift register content (unsigned) by some amount
  • GET R1, port (32 ports) ; input register content from 1 of 32 ports
  • PUT R1, port (32 ports) ; output register content to 1 of 32 ports
  • XCH R1, R2 ; exchange / swap contents of two registers
  • CLR R1{,R2} ; clear up to two registers at once
                       (when only one reg is specified, it’s code is duped)
  • MOD R1, oper[xxx] ; R1 = oper R1 for oper in (ABS, NOT, NEG, REV)
  • CHK R1, cond[xxx] ; CF = cond R1 for cond in (POS, NEG, ZER, EVN)