**Instruction Register Format:**

We have in total 32 instructions:

* 9 Two-Op instructions
* 9 One-Op instructions
* 7 Branch instructions
* 3 No-Op instructions
* 4 Jump subroutine instructions

And

* 8 Registers
* 8 Addressing modes

For each register operand we need `log2(8) = 3` + `log2(8) = 3` = 6 bits.

* In Two-Op instructions:
  + INSTRUCTION | 6 bits reg | 6 bits reg
* In One-Op instructions:
  + INSTRUCTION | 6 bits reg
* In Branch instructions:
  + INSTRUCTION | offset
* No-Op instructions:
  + INSTRUCTION
* Jump subroutine instructions:
  + JSR: INSTRUCTION
  + RTS: INSTRUCTION
  + INTERRUPT: INSTRUCTION
  + IRET: INSTRUCTION

16 bit IR:

* Two-Op (9 instructions):
  + 12 bits for registers, leaves out 4 bits
* One-Op (9 instructions):
  + 6 bits for register, leaves out 10 bits
* Branch (7 instructions)
  + 8 bits for branch offset, leaves out 8 bits
* No-Op (3 instructions)
  + Instruction only
* Jump Subroutine (4 instructions)
  + Instruction only

Opcodes design:

* Two op: (4 bits opcode)
  + [0000] -> [1000] b0’ + b0b1’b2’b3’
* One-op: (8 bits opcode [4 static and 4 dynamic])
  + 1001[0000] -> 1001[1000] b0b1’b2’b3
* Branch: (5 bits opcode [2 static and 3 dynamic])
  + 11[000] -> 11[110] (11 bits for offset) b0b1
* No-Op: (8 bits opcode [4 static and 4 dynamic])
  + 1010[0000] -> 1010[0010] b0b1’b2b3’
* Jump: (8 bits opcode [4 static and 4 dynamic])
  + 1011[0000] -> 1011[0011] b0b1’b2b3

Overall design:

* Two Op:
  + 4 bits OP-code | 6 bits register | 6 bits register
* One Op:
  + 8 bits OP-code | 00 (2 bits placeholder) | 6 bits register
* Branch:
  + 5 bits OP-code | XXX (3 bits placeholder) | 8 bits offset
* No Op:
  + 8 bits OP-code | XXXX XXXX (8 bits placeholder)
* Jump:
  + 8 bits OP-code | XXXX XXXX (8 bits placeholder)

Instructions opcodes:

* Prefixes:
  + pre\_one\_op = '1001'
  + pre\_branch = '11'
  + pre\_no\_op = '1010'
  + pre\_jump = '1011'
* Double Op:
  + 'MOV': '0000'
  + 'ADD': '0001'
  + 'ADC': '0010'
  + 'SUB': '0011'
  + 'SBC': '0100'
  + 'AND': '0101'
  + 'OR': '0110’
  + 'XOR': '0111’
  + 'CMP': '1000'
* Single Op:
  + 'INC': pre\_one\_op + '0000'
  + 'DEC': pre\_one\_op + '0001'
  + 'CLR': pre\_one\_op + '0010'
  + 'INV': pre\_one\_op + '0011'
  + 'LSR': pre\_one\_op + '0100'
  + 'ROR': pre\_one\_op + '0101'
  + 'ASR': pre\_one\_op + '0110'
  + 'LSL': pre\_one\_op + '0111'
  + 'ROL': pre\_one\_op + '1000’
* Branch:
  + 'BR': pre\_branch + '000'
  + 'BEQ': pre\_branch + '001'
  + 'BNE': pre\_branch + '010'
  + 'BLO': pre\_branch + '011'
  + 'BLS': pre\_branch + '100'
  + 'BHI': pre\_branch + '101'
  + 'BHS': pre\_branch + '110'
* No Op:
  + 'HLT': pre\_no\_op + '0000'
  + 'NOP': pre\_no\_op + '0001'
  + 'RESET': pre\_no\_op + '0010'
* Jump:
  + 'JSR': pre\_jump + '0000'
  + 'RTS': pre\_jump + '0001'
  + 'INTERRUPT': pre\_jump + '0010’
  + 'IRET': pre\_jump + '0011'