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)