Whiteboard Section

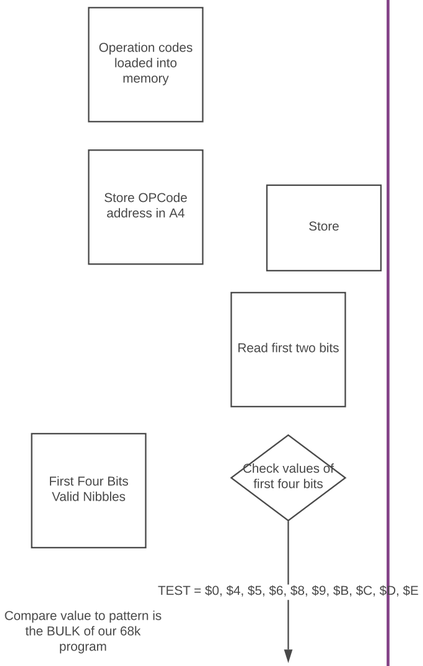
Each OPCode has 1-5 words. A word is 16 bits, or $xxxx. In a single word opcode, the last 6 bits are for EA. Since memory is byte addressable, I would look ahead 1 byte from the current instruction address.

Do a left shift to get rid of the non-EA bits. We're now left with 000 000 to 111 111

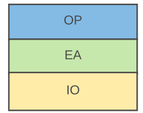
Different modes:
000 - Data Register
001 - Address Register

010 - Address indirect
011 - Address with post-increment

START WITH
Absolute Addressing (xxx).W, (xxx).L
Data Register Direct Dn
Immediate Addressing #
Address Register Direct An



Key:



Notes:
Possible EA implementation Ideas:
Storing the whole opcode bits across multiple data registers.
How would you differentiate which address registers?
Possibly increment bit by bit

Have a pointer to the current address.
Iterate the pointer from start to end.
Start analyzing bits. Default value FF.
Once you get to one not FF,

Decoding JSR
Possibly have a current
1. Load address of JSR
Check for first two bits

68K Diassembler Flowchart
By: Team SuperBIG