|  |  |  |
| --- | --- | --- |
| Opcode | | Description |
| Binary | Mnemonic |  |
|  |  |  |
|  |  |  |
| 0001 | Load X | Load the contents of address X into AC. |
| 0010 | Store X | Store the contents of AC to address X. |
| 0011 | Subt X | Subtract the contents of address X from AC and store the result in AC. |
| 0100 | Add X | Add the contents of address X to AC and store the result in AC. |
| 0101 | Input | Input a value from the keyboard into AC |
| 0110 | Output | Output the value in AC to the display. |
| 0000 | Halt | Terminate the program. |
| 0111 | Skipcond | Skip the next instruction on condition. |
| 1000 | Jump X | Load the value of X into PC. |

1a.

1b.

|  |  |  |
| --- | --- | --- |
| Opcode Binary | Opcode Mnemonic | Description |
| 1001 | Multiply X | Multiplies the contents of address X and AC and stores result in AC. |
| 1010 | Shift Left | Shifts the contents of AC two to the left and stores result in AC. |
| 1011 | Shift Right | Shifts the contents of AC two to the right and stores result in AC. |

1c.

The description for the Skipcond is: skip the next instruction on a condition. Since there is no specified condition, I will assume that the condition is skip if the accumulator is equal to or greater than 0, in order to explain how it works. It first of course needs a specific condition, once that is decided we can use it.

Once the program counter points to the Skipcond instruction, it will check if the content of the accumulator satisfies the condition. So in this example, if the value at the address in AC is 5, then the Skipcond condition will be satisfied and so the next instruction in memory will be skipped, the program counter will move up by two and the instruction after the skipped one, will be carried out.