Machine Code Table

Op-Code	Assembly	Operation
0000 0000 (00)	hlt	"Halt" : Halts the program
0000 0001 (01)	lda	"Load A" : Loads Register A with data from memory with address given by operand
0000 0010 (02)	sta	"Store A" : Stores the value in Register A to a location in memory given by the operand
0000 0011 (03)	ldb	"Load B" : Loads Register B with data from memory with address given by operand
0000 0100 (04)	stb	"Store B" : Stores the value in Register B to a location in memory given by the operand
0000 0101 (05)	wra	"Write A" : Writes the value (a 16-bit number) directly from the operand to Register A
0000 0110 (06)	wrb	"Write B" : Writes the value (a 16-bit number) directly from the operand to Register B
0000 0111 (07)	add	"Add" : Adds the values in Register A and Register B and puts the sum in Register A
0000 1000 (08)	sub	"Subtract" : Subtracts the values in Register A and Register B and puts the difference in Register A
0000 1001 (09)	mul	"Multiply" : Multiplies the values in Register A and Register B and puts the product in Register A
0000 1010 (0a)	div	"Divide" : Divide the values in Register A and Register B and puts the quotient in Register A
0000 1011 (0b)	cmp	"Compare" : Compare the values in Register A and Register B and set the ">" and "=" ALU flags
0000 1100 (0c)	jmp	"Jump" : Jump to read the program line given in the operand
0000 1101 (0d)	jeq	"Jump if equal" : Jump if the ALU "=" flag is set
0000 1110 (0e)	jnq	"Jump if not equal" : Jump if the ALU "=" flag is not set
0000 1111 (0f)	jls	"Jump if less" : Jump if the ALU ">" flag is not set
0001 0000 (10)	jgr	"Jump if greater" : Jump if the ALU ">" flag is set
0001 0001 (11)	ota	"Out A" : Push the value in Register A to the output specified in the operand
0001 0010 (12)	otb	"Out B" : Push the value in Register B to the output specified in the operand
0001 0011 (13)	jm	"Jump from Memory" : Jump to the program line specified in memory at the address in the operand
0001 0100 (14)	lam	"Load A from Memory" : Retrieve the value in memory at the address in Register A and load it to Register A
0001 0101 (15)	lbm	"Load B from Memory" : Retrieve the value in memory at the address in Register B and load it to Register B
0001 0110 (16)	oam	"Out A through Memory" : Write the value in Register A to the Out register whose address is defined in the memory address in the operand
0001 0111 (17)	obm	"Out B through Memory" : Write the value in Register B to the Out register whose address is defined in the memory address in the operand