Instruction	OpCode	Oper 1	Oper 2	Description	Sequence	Clock Cycles
“Fetch”				Fetch next instruction	M[PC] → <inst dec> PC + 1 → PC	1
NOP	0x00			No operation	PC → PC PC → PC	1
LDM $hhhh	0x21	DRH	DRL	Load from data memory to TOS	M[PC] → DRH PC + 1 → PC M[PC] → DRL SP - 1 → SP M[DR] → M[SP] PC + 1 → PC	3
LDI	0x22			Increment DR, load data mem to TOS	DR + 1 → DR SP – 1 → SP M[DR] → M[SP] PC → PC	2
STM $hhhh	0x41	DRH	DRL	Store TOS to data memory	M[PC] → DRH PC + 1 → PC M[PC] → DRL M[SP] → M[DR] SP + 1 → SP PC + 1 → PC	3
STI	0x42			Increment DR, store TOS to data mem	DR + 1 → DR M[SP] → M[DR] SP + 1 → SP PC → PC	2
PSH #dd	0x34	ACL		Push direct data to TOS	M[PC] → ACL SP - 1 → SP ACL → M[SP] PC + 1 → PC	2

POP	0x38			Pop TOS	M[SP] → ACL SP + 1 → SP 0 → ACL PC → PC	2
ADD	0x51			Adds top two stack values, push sum to TOS	M[SP] → ACL SP + 1 → SP AC + M[SP] → AC ACL → M[SP]	2
SUB	0x52			Subtracts top two stack values, push diff to TOS	M[SP] → ACL SP + 1 → SP AC - M[SP] → AC ACL → M[SP]	2
NEG	0x54			Negates TOS	0 → AC AC – M[SP] → AC ACL → M[SP] PC → PC	2
AND #dd	0x59	ACL		ANDs TOS with direct data, push result to TOS	M[PC] → ACL AC & M[SP] → AC ACL → M[SP] PC + 1 → PC	2
ORR #dd	0x5A	ACL		Ors TOS with direct data, push result to TOS	M[PC] → ACL AC | M[SP] → AC ACL → M[SP] PC + 1 → PC	2
INV	0x5C			Inverts TOS, replace TOS with result	M[SP] → ACL invert AC → AC ACL → M[SP] PC → PC	2

CPE #dd	0x61	ACL		Compare if TOS is equal to direct data (TOS will be ‘0’ if equal)	M[PC] → ACL ACL xor M[SP] → ACL SP - 1 → SP ACL → M[SP] PC → PC PC + 1 → PC	3
CNE #dd	0x62	ACL		Compare if TOS is not equal to direct data (TOS will be not ‘0’ if not equal)	M[PC] → ACL ACL xor M[SP] → ACL SP - 1 → SP ACL → M[SP] PC → PC PC + 1 → PC	3
BRZ &label	0x71	PCH	PCL	Branch if TOS is zero, stack is popped	M[PC] - > DRH PC + 1 → PC M[PC] → DRL M[SP] → AC SP + 1 → SP If AC = 0 DR → PC Else PC + 1 → PC	3
BRN &label	0x72	PCH	PCL	Branch if TOS is not zero, stack is popped	M[PC] - > DRH PC + 1 → PC M[PC] → DRL M[SP] → AC SP + 1 → SP If AC != 0 DR → PC Else PC + 1 → PC	3
BRU &label	0x74	PCH	PCL	Branch unconditionally	M[PC] - > DRH PC + 1 → PC M[PC] → DRL DR → PC	2

INP	0x81			Input IR to TOS	SP - 1 → SP IR → M[SP]	1
OUT	0x82			Output TOS to OR, stack is popped	M[SP] → OR SP + 1 → SP	1
PRT	0x84			Print TOS to PR, stack is popped	M[SP] → PR SP + 1 → SP	1
SER	0x85			Input SR to TOS	SP - 1 → SP SR → M[SP]	1
CLS	0x91			Clear the stack	<mem top> → SP PC → PC	1
END	0x98			End of program (aka halt)	PC – 1 → PC PC → PC	1
RST	0x9F			Reset CPU	0 → AC 0 → DR 0 → OUT 0 → PRT <mem top> → SP 0 → PC	3