ADD

1. Pos + pos = pos
   1. Load R1 <- 00FF
   2. Load R2 <- 00FF
   3. Add R3 <- R1 + R2 = 01FE
   4. halt
2. Pos + pos = neg
   1. Load R1 <- 7FFF
   2. Load R2 <- 7FFF
   3. Add R3 <- R1 + R2 = FFFE
   4. halt
3. 16 Bit Overflow
   1. Load R1 <- FFFF
   2. Load R2 <- FFFF
   3. Add R3 <- R1 + R2 = FFFE
   4. halt
4. Pos + Neg
   1. Load R1 <-FFFE
   2. Load R2 <- 0001
   3. Add R3 <- R1 + R2 = FFFF
   4. halt

ADD immediate

1. Pos + pos = pos
   1. Load R1 <- 00FF
   2. Add R3 <- R1 + 1F = 011E
   3. halt
2. Pos + pos = neg
   1. Load R1 <- 7FFF
   2. Add R3 <- R1 + 1F = 801E
   3. halt
3. Bit Overflow
   1. Load R1 <- FFFF
   2. Add R3 <- R1 + 1F = 001E
   3. halt
4. Pos + Neg
   1. Load R1 <- F000
   2. Add R3 <- R1 + 1F = F01F
   3. halt

AND

1. Zero\*Zero
   1. Load R1 <-0000
   2. Load R2 <- 0000
   3. AND R3 <- R1 & R2 = 0000
   4. halt
2. Zero\*Nonzero
   1. Load R1 <-FFFF
   2. Load R2 <- 0000
   3. AND R3 <- R1 & R2 = 0000
   4. halt
3. Nonzero\*Nonzero
   1. Load R1 <-FFFF
   2. Load R2 <- 0001
   3. AND R3 <- R1 & R2 = 0001
   4. Halt

AND b. immediate

1. Zero\*Zero
   1. Load R1 <-0000
   2. AND R3 <- R1 & 0= 0000
   3. halt
2. Zero\*Nonzero
   1. Load R1 <-0000
   2. AND R3 <- R1 & x1F= 0000
   3. halt
3. Nonzero\*Nonzero
   1. Load R1 <-FFFF
   2. AND R3 <- R1 & 1F= 001F
   3. halt

NOT

1. Pos
   1. Load R1 <-7F08
   2. Not R2 <- R1 = 80F7
   3. halt
2. neg
   1. Load R1 <-80F7
   2. Not R2 <- R1 = 7F08
   3. halt

LD/ST

1. Offset = max
   1. Load R1 <- ABCD
   2. Store R1 -> [01FF] = ABCD
   3. Load R2 <- [0000000 & x1FF] = ABCD

LDI/STI

1. No value in referred address
   1. LDI R1 <- [[0000000111111111]] // if mem. init. To 0, = ABCD // else R1 = ?
2. Newpage in referred address + store
   1. Mem[FFF0] = ABCD
   2. Mem[0000000111111111] = FFF0
   3. Mem[0000000111111110] = FFF1
   4. LDI R1 <- [[0000000111111111]] = ABCD
   5. STI R1 -> [[0000000111111110]] // [FFF1] = ABCD

LDR/SDR

1. Address out of bounds (FFFF +63)
   1. Load R1 <- FFFF
   2. LDR R2 <- 111111(R1) //Out of bounds or R2 = [003E] = ABCD
   3. Load R3 <- 1234
   4. STR R3 -> 111110(R1) // Out of bounds or [003D] = 1234
2. Address + Index = new page +store
   1. Load R1 <- 0FFF
   2. LDR R2 <- 111111(R1) //R2= [103E] = ABCD
   3. Load R3 <- 1234
   4. STR R3 -> 111110(R1) //[103D] = 1234

\_\_\_\_\_\_\_\_\_\_\_\_\_

BRx

1. BRn
   1. Mem[0] = 8000
   2. Ld R1 <- [0]//n=1,p=0,z=0
   3. Brn F
   4. Halt//PC = 200
2. BRn Fail
   1. Mem[0] = 7000
   2. Ld R1 <- [0]// n=0,p=1,z=0
   3. Brn F
   4. Halt//PC = 5
3. BRz
   1. Mem[0] = 0000
   2. Ld R1 <- [0]//n=0,p=0,z=1
   3. Brn F
   4. Halt = 200
4. BRz Fail
   1. Mem[0] = 8000
   2. Ld R1 <- [0]//n=1,p=0,z=0
   3. Brn F
   4. Halt//PC = 5
5. BRp
   1. Mem[0] = 7000
   2. Ld R1 <- [0]//n=0,p=1,z=0
   3. Brn F
   4. Halt//PC = 200
6. BRp fail
   1. Mem[0] = 8000
   2. Ld R1 <- [0]//n=1,p=0,z=0
   3. Brn F
   4. Halt//PC = 5
7. BR unconditional (0000111)
   1. Mem[0] = 8000
   2. Ld R1 <- [0]//n=1,p=0,z=0
   3. Brn F
   4. Halt//PC = 200

TRAP

1. OUT
   1. Ld R0 <- 0048
   2. Trap x21 // display ‘H’
   3. Halt
2. PUTS
   1. Display “Success!”
   2. Halt
3. IN
   1. Trapx23 ‘H’//R0=0048
   2. halt
4. HALT
   1. Inherently passed.
5. OUTN pos
   1. Ld R0 <- 000A
   2. Trap x31//Display “10”
   3. halt
6. OUTN neg
   1. Ld R0 <- FFFF
   2. Trapx31//Display ”-1”
7. INN pos
   1. Trapx33//Display “10”, R0=000A
   2. Halt
8. INN neg
   1. Trapx33//Display”-1”, R0 = FFFF
   2. Halt
9. RND
   1. Trapx43//Rand
   2. Trapx31//Display
   3. Trapx43//Rand
   4. Trapx31//Display
   5. Trapx43//Rand
   6. Trapx31//Display
   7. Halt
10. TRAP -> use R7
    1. Halt//R7 = 0001

JSR

1. L = 0
   1. JSR 1FF
   2. Halt//PC =200//R7=0 or unitialized
2. RET normal
   1. JSR 1FF
   2. RET
   3. Halt//PC = 0002
3. RET overwritten
   1. JSR 1FF
   2. Ld R7 <- 39FF
   3. RET
   4. Halt//PC = 4A00

JSRR

1. L = 0
   1. Load R1 <-0000
   2. JSRR 111111(R1)
   3. Halt//PC = 0200
2. Address out of bounds (FFFF +63)
   1. Load R1 <- FFFF
   2. JSRR 111111(R1) //Out of bounds, or PC = [003E]
   3. RET
   4. Halt//PC =0004
3. Ret Normal, Address + Index = new page +store
   1. Load R1 <- 0FF0
   2. JSRR 111111(R1) //R7= , or PC = 102F
   3. RET
   4. Halt//PC =0003
4. RET overwritten, Address + Index = new page +store
   1. Load R1 <- 0FF0
   2. JSRR 111111(R1) //R7= , or PC = 102F
   3. LD R7 <- 0100
   4. RET
   5. Halt//PC =0101

DBUG

1. DBUG
   1. Ld R1 <- 0FF0
   2. DBUG
   3. Halt

Miscellaneous

1. Inf. Loop (Instr. Limit Test)
   1. Ld R1 <-0000
   2. Brz [self]
2. No Halt (Execute Unitialized Mem)
   1. Halt at PC = Limit, or ?
3. Input file Does not exist
4. Input is Directory
5. Input File 35 or 36, Enter non Decimal ‘\*’//Display Error
6. Text Record does not start with T
7. Missing Header Record
8. Missing End Record
9. No Text Records
10. Header is too short
11. Non Hex Characters
12. Text Records are too short
13. End record Too short
14. Segment length is too short
15. Average Calculator

LEA

* 1. LEA 1FF(R1) <- 01FF
  2. Halt