1.Q:

0x00: SUB R7, R7, R7 @load memory from r7, and then store the result of (r7-r7) to r7—make sure that the memory in r7 is all cleared, which is like initialize the r7 to 0

Label1: 0x02: SUB R0, R0, R1 @load memory from r0 and r1, and then store the result of (r0-r1) to r0

0x04: BLEZ R0 Labe2 @if r0<=0, branch to branch 0x08 which is the increase command: r7=r1+r0

0x06: JUMP Label1 @jump to the SLL line and store the result of r0 shift left by one bit to destination

Label2: 0x08: ADD R7, R1, R0 @load memory from r1 and r0, and then store the result of (r0+r1) to r7

0x0A: HALT

2.

After 0x00: R7: 0000 0000

After 0x02: R0: 0001 0110

After 0x06: R0: 0000 1111

After 0x06: R0: 0000 1000

After 0x06: R0: 0000 0001

After 0x06: R0: 1111 1010

After 0x04: go to Label1 which is 0x08

After 0x08: R7: 0000 0001

So now,

R0: 1111 1010

R1: 0000 0111

R2: 0000 0000

R3: 0000 0000

R4: 0000 0000

R5: 0000 0000

R6: 0000 0000

R7: 0000 0001

3. r0<-a, r1<-b, r2<-(a-b)

0x00: SUB R7, R7, R7

0x02: SUB R2, R0, R1

L1: 0x04: BRZ R2, L4

0x06: BLEZ R2 L2 @IF(A-B<=0)

0x08: JUMP L3

L2: 0x0A: SUB R1, R1, R0 @B=B-A

0x0C: JUMP L1

L3: 0x0E: SUB R0, R0, R1. @ELSE: A=A-B

0x10: JUMP L1

L4: 0x12: ADD R7, R0, #0

0x14: HALT