Example Pep/8 program (hex)

Address	Contents (hex)
0	**C0**
1	**00**
2	**07**
3	1A
4	70
5	00
6	03
7	00

PC

0 **Fetch** byte 0 and update PC. In hex the byte is it is C0, in binary it is 11000000

1 **Decode** the instruction (see Fig 4.6). There must be unique match with one of the patterns In this case the match is with

1100 r aaa Load register r from memory

It is not a one-byte "unary" instruction so get the next two bytes (from addresses 1 and 2) and update the PC. These two bytes are the operand; in hex the operand is 0007

3 **Execute** the instruction.

Register is A (r=0).
Operand is 0007.
aaa=000

The aaa 'addresing mode' bits('Immediate') tell us that the operand value is the number '0007' (as opposed to the contents of Memory[0007] for example).
Load copies a number from memory to a register so as a result of this instruction

Register A	0	0	0	7

Address	Contents (hex)
0	C0
1	00
2	07
3	**1A**
4	70
5	00
6	03
7	00

3 **Fetch** byte 3 and update PC. In binary the byte fetched (1A) is 00011010

4 **Decode** the instruction. It is unary "negate register" 0001101r

4 **Execute** the instruction.

Register is A (r=0)

Register A	F	F	F	9

Address	Contents (hex)
0	C0
1	00
2	07
3	1A
4	**70**
5	**00**
6	**03**
7	00

4 **Fetch** byte 4 and update PC. In binary the byte fetched (70) is 01110000

5 **Decode** the instruction. It is a 3-byte "add to register"

0111 r aaa

It is not a unary instruction so we get bytes 5 and 6 and update the PC. In hex the operand bytes are 0003

7 **Execute** the instruction.

Register is A (r=0)
Operand is 0003
aaa = 000

Operand is 0003 so Register A is now FFFC (FFF9 + 0003)

Register A	F	F	F	C

Address	Contents (hex)
0	C0
1	00
2	07
3	1A
4	70
5	00
6	03
7	**00**

7 **Fetch** byte 8 and update PC. In binary the byte fetched (00) is 000000000

8 **Decode** the instruction. It is unary STOP

8 **Execute** the instruction – stop the CPU