Task 7

Dylan Rodwell: 105341089

## 7.1.1

0x00000065, because this is the hexadecimal representation of 101

## 7.1.2

0x00000101, because the input was initialized with “0x” which means it is a hexadecimal value

## 7.1.3

0x00000005, because the input was initialized with “0b” which means it is a binary value, and the output represents the value 5 in hexadecimal

## 7.1.4

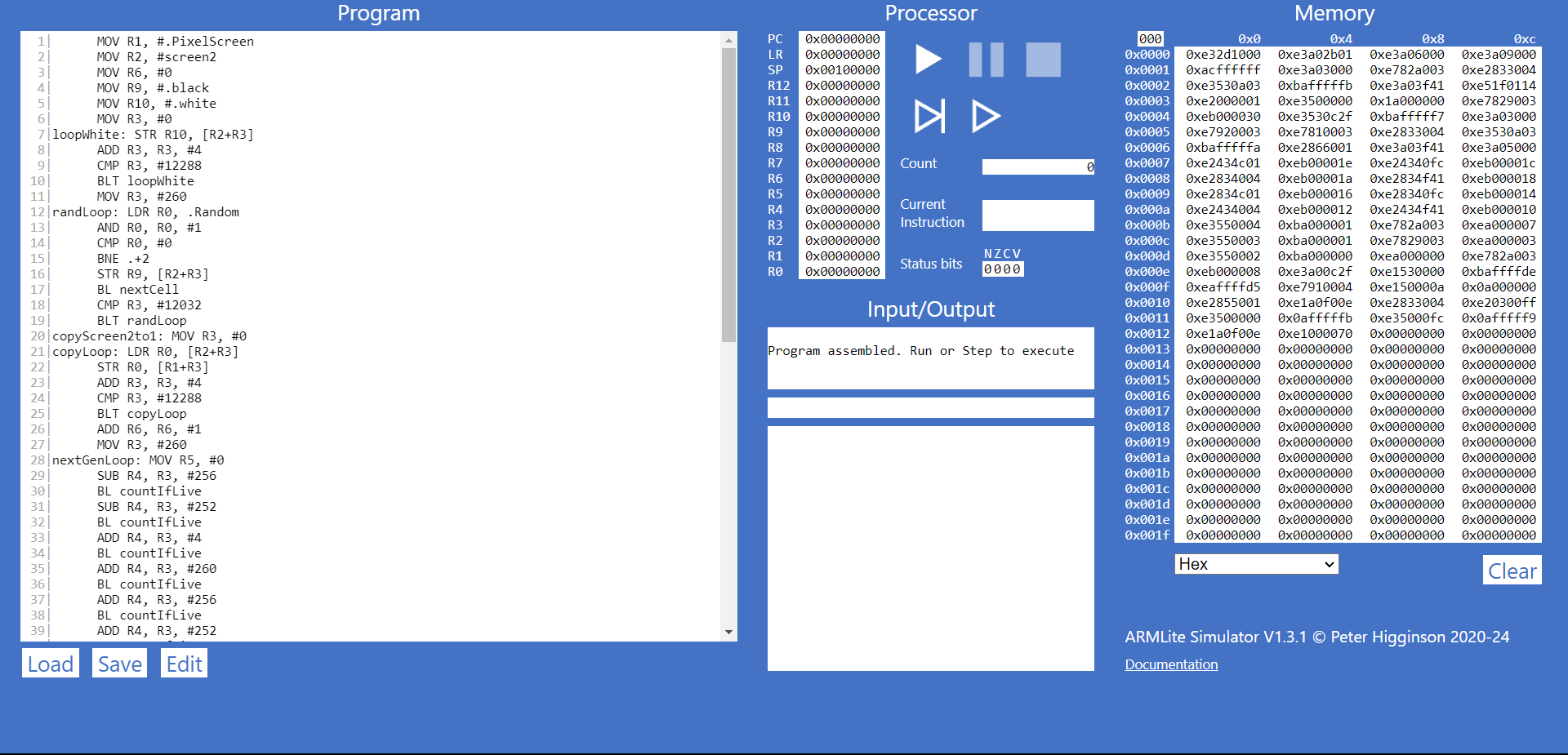
The tooltip tells you the decimal and binary representations of the value you are hovering over.

No because that signifies what value is allocated to each bit

## 7.2.1

Because each memory word contains 32 bits which is the equivalent to 4 bytes, and moving the memory by 4 bytes means it is moving the memory word by 1

## 7.3.1



## 7.3.2

The program has assigned all these values to memory words ready to be pushed to the processor when the program is run

## 7.3.3

This 5 digit hex value would signify what bit the data is allocated to in the memory word

## 7.4.1

The highlighting in both the windows would signify the line of code the program is about to execute and where in the memory word is being accessed/used

## 7.4.2

Instead of stepping through all the code automatically, the button circled in red steps through one line of code when it is pressed and stops after it is done, waiting to be pressed again

## 7.4.3

The code has stopped just before the breakpoint allowing the programmer to step through the following processes and debug any problems

## 7.5.1

I predict that the program, will add the value of r0 (which is 1) and the value 8 and store the answer of 9 in r1

## 7.5.2

A blue and white rectangular box with white text

Description automatically generated

## 7.5.3

A blue screen with white text

Description automatically generated

## 7.5.4

Code:

Program

  1|      mov r0, #75

  2|      and r1, r0, #10

  3|      or r2, r1, #6

  4|      eor r2, r2, #2

  5|      lsl r0, r0, #3

  6|      lsr r0, r0, #6

  7|      halt

|  |  |  |
| --- | --- | --- |
| **Instruction** | **Decimal value of the destination register after executing this instruction** | **Binary value of the destination register after executing this instruction** |
| Mov r0, #75 | 75 | 1001011 |
| And r1, r0, #10 | 10 | 1010 |
| Or r2, r1, #6 | 14 | 1110 |
| Eor r2, r2, #2 | 12 | 1100 |
| Lsl r0, r0, #3 | 600 | 1001011000 |
| Lsr r0, r0, #6 | 9 | 1001 |

## 7.5.5

A blue screen with white text and icons

Description automatically generated with medium confidence

## 7.5.6

A screenshot of a computer

Description automatically generated

## 7.6.1

Because the binary value starts with a 1, and when viewing it in the signed method, 1 means the value is a negative number, thus the value is negative

## 7.6.3

In binary;

1 = 0b00000000000000000000000000000001

-1 = 0b11111111111111111111111111111111

2 = 0b00000000000000000000000000000010

-2 = 0b11111111111111111111111111111110

The pattern is: when representing a negative number, flip all the bits + 1

## 7.6.4

A screenshot of a computer

Description automatically generated