Labels and Branching

Recall Labels

- Labels are used to give meaningful names to locations in memory
- Previously we saw this used to define variables:

  Eg `someplace:  .word 0`

  Defines “someplace” as a label for a memory location holding a word of value 0

- But we can also use labels to mark the location of instructions in our programs !

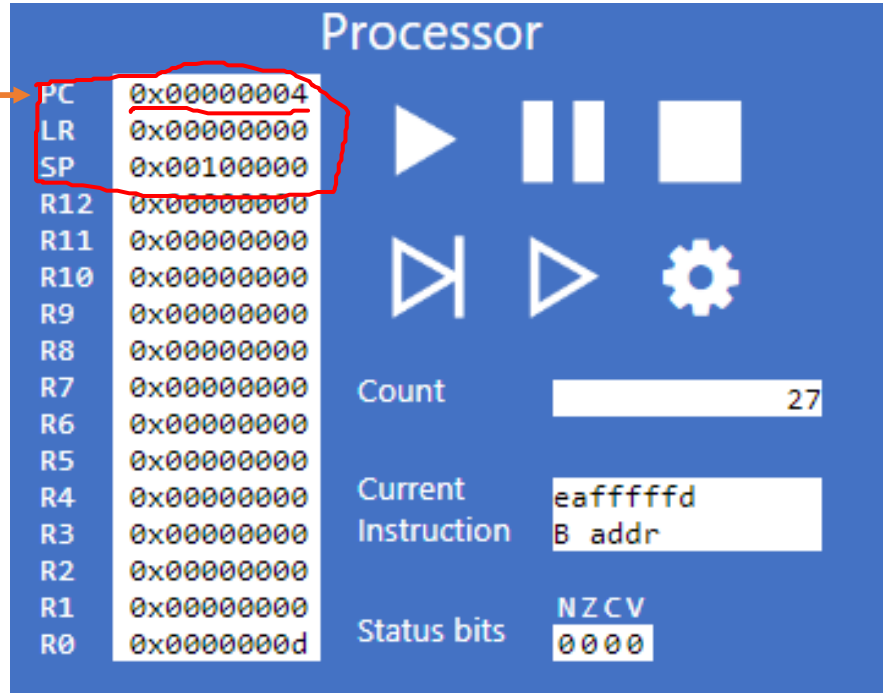
Instructions have addresses too !

- Recall that before a program is executed, it is loaded into memory
- Every instruction therefore occupies a word of memory, and has an address like any other data
  - Recall also that this is a core principle of the Von Neumann architecture
- We can therefore use the address of specific instructions to jump from one location to another
  - often referred to as branching

The Program Counter

- To understand labels and branching we need to understand how the CPU keeps track of which instruction to execute next
- It uses the special register, the Program Counter (PC) register

Program **C**ounter



The Program Counter (cont)

- After each instruction is executed, the PC register is updated with the address of the next instruction to execute.
- This update is performed by a hardware counter within the ALU

Labels as branch points

To see how labels and branching work, consider this code:

```
MOV R0, #0
Loop:
ADD R0,R0,#1
B Loop
HALT
```

Labels as branch points

Consider this code:

```
MOV R0, #0
Loop:
ADD R0,R0,#1
B Loop
HALT
```

Label referring to the address of the next instruction

Labels as branch points

Consider this code:

```
MOV R0, #0
Loop:
ADD R0,R0,#1
B Loop
HALT
```

Label referring to the address of the next instruction

A **B**ranch instruction, telling the program counter that the next instruction to execute is at address of label ‘Loop’

Labels as branch points

Consider this code:

```
MOV R0, #0
Loop:
ADD R0,R0,#1
B Loop
HALT
```

Label referring to the address of the next instruction

A **B**ranch instruction, telling the program counter that the next instruction to execute is at address of label ‘Loop’

Let’s load this into ARMlite and see it in action !

Unconditional **B**ranch

```
B someplace
```

Branch to the address represented by the label someplace

Unconditional **B**ranch

```
B someplace
```

This can be any word aligned address,
but is almost always given as a label,
defined elsewhere in the program

Branch to the address represented by the label someplace