**F1 Starting Lights in RISCV – Testing in GTKWave**

When the program is run, all registers are equal to 0. The first 4 instructions are run, and they perform their roles, loading 99 into x10(limit for the counter), loading 0 into x12(start of the counter), loading 0 into x20(f1 light register) and branching when x31 is not 0 (trigger is when x31 is set to 0 using vBuddy, but for GTKWave the trigger is as soon as the program starts).

Graphical user interface

Description automatically generated

The PC has been set to 0 rather than the memory map recommendation of BFC00000 for the sake of simplicity in this test.

Graphical user interface

Description automatically generated

Scrolling along to the 5th instruction LI x20, 1 – we can see that register x20 has been updated with the value 1, which corresponds to the first f1 light turning on. We can also see that PC has jumped to address 54, this is the JAL instruction being executed as we jump to the first instance of using the counter. The next PC instruction 18, is stored in x1 for when we finish the count and return. The next instruction 00160613 starts the counter and here we can see the register x12 increments by one, whilst the PC address jumps between 54 and 58, signalling that the next instruction BNE also works.

Graphical user interface

Description automatically generated

The program keeps looping until the count register x12 is equal to x10 which is hex 21 in this case. The program is no longer bound by the branch and instead continues to the next 2 instructions, resetting the counter and returning the next instruction stored in x1, 18. The f1 register x20 is changed to 3 or 11, so that now 2 lights are on after a period of roughly 1 seconds. The next PC instruction is again stored in x1 and we jump back to the counter to wait roughly 1s to turn on the next light.

Graphical user interface

Description automatically generated

1 second has been counted, counter resets, 3rd f1 light turns on (111), next address stored in x1, 28

Graphical user interface

Description automatically generated

1 second has been counted, counter resets, 4th f1 light turns on (1111), next address stored in x1, 30

Graphical user interface

Description automatically generated

1 second has been counted, counter resets, 5th f1 light turns on (11111), next address stored in x1, 38

Graphical user interface

Description automatically generated

1 second has been counted, counter resets, 6th f1 light turns on (111111), next address stored in x1, 40

Graphical user interface, timeline

Description automatically generated

1 second has been counted, counter resets, 7th f1 light turns on (1111111), next address stored in x1, 48

Graphical user interface

Description automatically generated

1 second has been counted, counter resets, 8th f1 light turns on (11111111), next address stored in x1, 50

Graphical user interface

Description automatically generated

1 second has been counted, counter resets and all lights are turned off. The program starts repeating again.

As you can see, the entire f1 light sequence is passed through with this assembly code implementation and with our single-cycle RISCV processor. This test is only for the F1 light sequence and not the reference program.

A picture containing graphical user interface

Description automatically generatedText, calendar

Description automatically generated with medium confidenceAssembly code Machine code