Computer Architecture - CS2323. Autumn 2024

Lab-2: GCD (Greatest Common Divisor) calculation

As a part of this lab assignment, we need to calculate the GCD of multiple groups of two 64-bit unsigned numbers. The set of input numbers are provided as dwords in the data segment starting from address 0x10000000. The GCD computed from your code should be present starting from address 0x10000200 in the memory (data segment base + 0x200), as shown below. The first word in the data segment indicates the total group of input numbers (total number of gcd computations to be done) followed by the actual input numbers.
You can use any RISC-V instruction taught in the class (**NO** mul, div, rem, float instructions).

```
.data
.dword count_of_gcd, input11, input12, input21, input22, input31, input32, …

.text
   #The following line initializes register x3 with 0x10000000
   #so that you can use x3 for referencing various memory locations.
   lui x3, 0x10000
   #your code starts here

   #The final result should be in memory starting from address 0x10000200
   #The first dword location at 0x10000200 contains gcd of input11, input12
   #The second dword location from 0x10000200 contains gcd of input21,
input22, and so on.
```

Example: If the data section contains .dword 3, 12, 3, 125, 50, 32, 16 - it indicates that we have 3 sets of gcd to be calculated: (12, 3); (125, 50); (32,16).

After executing your code, memory location from 0x10000200 should contain dwords like 3, 25, 16

Note: If one of the two inputs is 0, the GCD is not properly defined. In such cases, assume that the GCD is 0.

Instructions:

1. Use Ripes simulator from: https://github.com/mortbopet/Ripes/releases/download/v2.2.4/Ripes-v2.2.4-linux-x86_64.AppImage - the 2.2.4 version is more stable and reliable than later ones.
2. Configure simulator for 64-bit processor (click on the processor button below File in the top-left and select 64-bit single cycle processor).
3. While doing this exercise, try to use breakpoints, single stepping, etc. features of the simulator for a better understanding. We will need these features when debugging the programs in subsequent assignments. Also, see the corresponding disassembled (translated) code in the right pane.

Submission instructions:

1. Submit the assembly code as a file named YOUR_ROLLNUM.s (e.g., CSYYBTECHXXXXX.s)
2. The assignment should be done individually
3. Copying from others or any other source is strictly prohibited and subject to strict penalty
4. Assignments will be tested for similarity among each other and any violation will be reported appropriately