Assembly Language and Computer Architecture Lab

Week 4

Assignment 2

abs $s0, $s1

If **$s0** < 0:

- Shift all bit to the right arithmetically we get 32 bits 1 (**0xffffffff**)

- Using **XOR** for **0xffffffff** and **$s0** to get the 1's complement number of **$s0**.

- Using **subu** to subtract **0xffffffff** (which is **-1**) from the 1's complement number, it means **+1** to get the 2's complement number.

If **$s0** ≥ 0:

- Shift all bit to the right arithmetically we get 32 bits 0 (**0x00000000**)

- Using **XOR** for **0x00000000** and **$s0** makes **$s0** remains unchanged.

- Using **subu** to subtract **0x00000000** (which is **0**) from **$s0**, also keeps **$s0** unchanged.

b) move $s0, $s1

Get sum of **$zero** and **$s1**, then store the result to **$s0**

c) not $s0

Apply logic **NOR** to each bit of number in **$s0** and bit 0 (taken from **$zero**) to reverse all bits.

d) ble $s1, $s2, L

- First compare if **$s2** < **$s1** then **$t1** = 1, if not then **$t1** = 0

- Using **beq** to check if **$t1=0**, which means **$s2** < **$s1** is false, or **$s1** ≤ **$s2**, then jumps to the branch. Otherwise, do not jump.