HW3 - Report

- Q1: Which part of the module in simple-single-CPU is redundant? Can you design a new instruction to use this module?

In the given context, the `Zero_Filled` module is not being utilized within the Simple_Single_CPU.

Regarding the instruction `ori`, a representation can be seen below:

Instruction	**Example**	**meaning**
ori rt, rs, imm	ori r1, r2, 255	Reg[rt]=Reg[rs] or 0xFF

Taking the instance where r2 holds the value 0x1234, the resultant value in r1 would be 0x12FF. Conversely, when employing a sign-extended approach, the immediate (imm) value becomes 0xFFFF. This results in an incorrect outcome of 0xFFFF in r1. It is under such circumstances that the Zero_Filled module proves preferable over the sign-extended module.

- Q2: Which instruction is redundant and why?

It has been observed that certain instructions may be deemed redundant due to the availability of alternative instructions that can achieve the same outcome. I present two specific examples for clarity

1. The instruction `bnez` could be substituted by the instruction `bne combined with setting rt to 0`. To detail:

```
bnez rs, 0, imm  == bne rs, rt, imm  // where rt set to zero
```

2. The instruction `blt` can be equivalently expressed through a combination of `slt` and `bne`. Specifically, the instruction:

```
blt rs, rt, imm
```

can be represented as:

```
slt $t0, rs, rt
bne $t0, $zero, imm
```