*** <u>Note well that the code we have submitted along this file was simulated using ISE software</u>**

1. Decide the registers and their usage protocol.

CSE Bubble has 32 registers. The register usage protocol is as follows:

- $zero: Always contains the value 0. This register cannot be written to.
- $at: Reserved for the assembler. Used to temporarily store data while performing assembly.
- $v0-$v1: Return values for functions.
- $a0-$a3: Arguments for functions.
- $t0-$t9: Temporary registers for general use.
- $s0-$s7: Saved registers for function calls.
- $k0-$k1: Reserved for use by the kernel.
- $gp: Global pointer register. Points to the middle of the 64 KB data memory.
- $sp: Stack pointer register. Points to the top of the stack in data memory.
- $fp: Frame pointer register. Points to the top of the current function's stack frame.
- $ra: Return address register. Contains the return address of the current function.

2. Decide upon the size for instruction and data memory in VEDA.

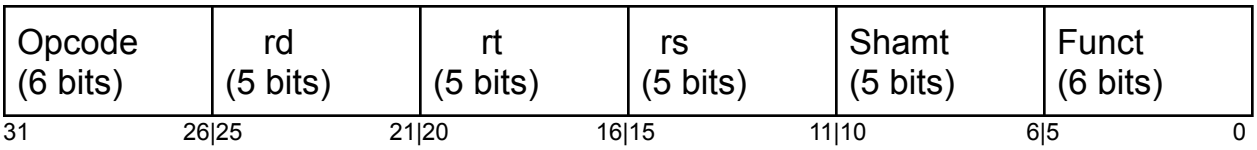
The size for instruction and data memory should be roughly 2^9 * 32 bytes each for both.

3. Design the instruction layout for R-, I- and J-type instructions and their respective encoding methodologies.

R-Type Instruction:

N.B. here the destination registers places have changed for our convenience.

The R-type instructions are used for arithmetic and logical operations, and have the following layout:

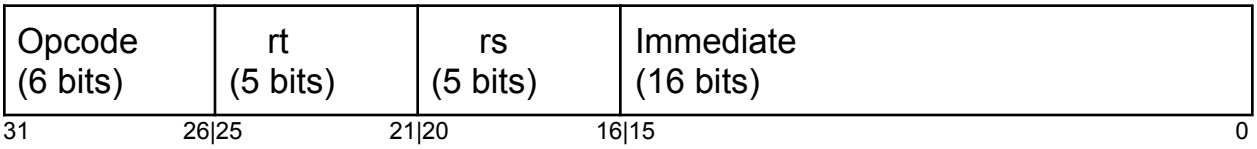


- Opcode: 6 bits, specifies the type of instruction as an R-type instruction.
- Rs, Rt, and Rd: 5 bits each, specify the source and destination registers for the instruction.

- Shamt: 5 bits, specifies the shift amount for shift operations.
- Funct: 6 bits, specifies the function code for the instruction.
- In our work this type is used for add, addu, sub, subu, and, or, slt.

I-Type Instruction:

N.B. here the destination registers places have changed for our convenience.

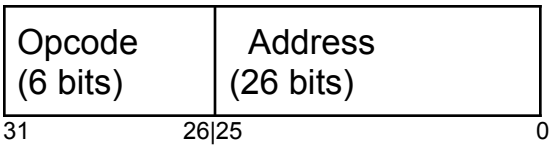


- Opcode: 6 bits, specifies the type of instruction as an I-type instruction.
- Rs and Rt: 5 bits each, specify the source and destination registers for the instruction.
- Immediate: 16 bits, specifies the immediate value for the instruction.
- In our work this type is used for addi, addiu, andi, ori, sll, srl, slti, lw, sw, and all branch instructions.

Also note that we have different uses for rt and rs in load and storee word instructions. Incase for load word we have rt as the register from where the data is taken and rs is the register where the data for memory address is stored. Similarly in case for load word we have the data taken from memory stored in rt. In all other places where I-Type is used the destination register where the final result is stored it is rt.

Another point to be noted is that we have used immediate type instruction for shift left and shift right instructions where the amount to be shifted is stored in the immediate bits

J-Type Instruction:



- Opcode: 6 bits, specifies the type of instruction as a J-type instruction.
- Address: 26 bits, specifies the address to jump to.
- In our work this type is used for j , jr, jal.

Note well that we have used ISE software to simulate the code provided along with this file.

Some key points:

- **We have used the display command to keep track of the numbers to be sorted.**
- **The simulation run time was 100 micro seconds so ater that time the sorted numbers are shown in the console in the ISE isim simulator.**
- **The number to be sorted are given in the data memory and the number of numbers to be sorted is given as an instruction in the instruction memory.**
- **We have also provided the mips code as bubble_sort_Labajyoti_Das_210552_Ashutosh_Agrawal_210219.asm file in which the bubble_sort part acted as the reference for the instruction memory.**