1. [PDS1] Decide the registers and their usage protocol.

Reg. No.	Code Given	Description
0	reg_0	zero register
1	lc	large constants
2-3	rv1-rv2	return variables from function
4-7	a1-a4	function arguments
8-17	t0-t9	temporary variables
18-25	s0-s7	saved temporary variables
26-27	k0-k1	os kernel
28	gp	global_pointer
29	sp	stack_pointer
30	fp	frame_pointer
31	ra	return address

2. [PDS2] Decide upon the size for instruction and data memory in VEDA.

Size	**Reg. No.**
Instruction Memory	32 registers
Data Memory	32 registers

3. [PDS3] Design the instruction layout for R-, I- and J-type instructions and their respective encoding methodologies.

Instruction layout for R, I and J type instructions:

Bit. No.	Codes
R-Type	
31-26	opcode
25-21	$rs
20-16	$rt
15-11	$rd
10-6	shamt
5-0	funct

Bit. No.	Codes
I-Type	
31-26	opcode
25-21	$rs
20-16	$rt
15-0	immediate

Bit. No.	Codes
J-Type	
31-26	opcode
25-0	address

We used opcodes from 1 to 25 for different instructions to be implemented by the processor. And the funct and shamt values are 0 by default and does not affect the instructions. The opcodes used are as shown below

Class	Instruction	OPCODES
Arithmetic	add r0, r1, r2	000001
	sub r0, r1, r2	000010
	addu r0, r1, r2	000011
	subu r0,r1,r2	000100
	addi r0,r1,1000	000101
	addiu r0,r1, 1000	000110
Logical	and r0,r1,r2	000111
	or r0,r1,r2	001000
	andi r0,r1, 1000	001001
	ori r0,r1, 1000	001010
	sll r0, r1, 10	001011
	srl r0, r1, 10	001100
Data transfer	lw r0,10(r1)	001101
	sw r0,10(r1)	001110
Conditional Branch	beq r0,r1,10	001111
	bne r0,r1,10	010000
	bgt r0,r1,10	010001
	bgte r0,r1, 10	010010
	ble r0,r1, 10	010011
	bleq r0,r1, 10	010100
Unconditional Branch	j 10	010101
	jr r0	010110
	jal 10	010111
Comparison	slt r0,r1,r2	011000
	slti 1,2,100	011001