1. **Describe the input fields of each pipeline register**
2. IF/ID pipeline register
   1. 32 bits for PC+4
   2. 32 bits for current instruction
3. ID/EX pipeline register
   1. 32 bits for RSdata
   2. 32 bits for RTdata
   3. 21 bits for last 21 bits of current instruction
   4. 32 bits for PC+4
   5. 1 bit for RegWrite control line
   6. 1 bit for RegDst control line
   7. 1 bit for ALUsrc control line
   8. 1 bit for Branch control line
   9. 1 bit for MemRead control line
   10. 1 bit for MemWrite control line
   11. 1 bit for MemtoReg control line
   12. 2 bits for ALUOp control line
   13. 32 bits for SignExtendData
4. EX/MEM register
   1. 32 bits for ALUresult
   2. 32 bits for pc of branch instruction
   3. 5 bits for number of write register
   4. 32 bits for RTdata
   5. 1 bit for RegWrite control line
   6. 1 bit for Branch control line
   7. 1 bit for MemRead control line
   8. 1 bit for MemWrite control line
   9. 1 bit for MemtoReg control line
   10. 1 bit for zero output from ALU
5. MEM/WB register
   1. 32 bits for MemReadData
   2. 32 bits for ALUresult
   3. 5 bits for number of write register
   4. 1 bit for MemtoReg control line
   5. 1 bit for RegWrite control line

2. Explain your control signals in the sixth cycle(both test data test\_1.txt

and test\_2.txt are needed)

1. test\_1.txt
   1. IF stage (slt $6, $2, $1)
      1. Branch (from instruction in MEM stage): 0 (not branch instruction)
   2. ID stage (or $5, $1, $0)
      1. RegWrite (from instruction in WB stage): 1 (can write to register)
   3. EX stage (and $3, $5, $6)
      1. RegDst: 1 (RD)
      2. ALUsrc: 0 (RTdata)
      3. ALUOp: 00 (R type)
   4. MEM stage (addi $4, $0, 14)
      1. MemRead: 0 (can’t read from data memory)
      2. MemWrite: 0 (can’t write to data memory)
   5. WB stage (addi $2, $0, 17)
      1. MemtoReg: 0 (ALUresult)
2. test\_2.txt
   1. IF stage (sw $3, 8($3))
      1. Branch (from instruction in MEM stage): 0 (not branch instruction)
   2. ID stage (sw $2, 4($0))
      1. RegWrite (from instruction in WB stage): 1 (can write to register)
   3. EX stage (addi $5, $0, 10)
      1. RegDst: 0 (RT)
      2. ALUsrc: 1 (SignExtendData)
      3. ALUOp: 01 (add)
   4. MEM stage (addi $4, $0, 9)
      1. MemRead: 0 (can’t read from data memory)
      2. MemWrite: 0 (can’t write to data memory)
   5. WB stage (addi $3, $0, 8)
      1. MemtoReg: 0 (ALUresult)

test\_1.txt

010011 00000 00001 0000000000011111 -> addi $1, $0, 31

010011 00000 00010 0000000000010001 -> addi $2, $0, 17

010011 00000 00100 0000000000001110 -> addi $4, $0, 14

000000 00101 00110 00011 00000 011111 -> and $3, $5, $6

000000 00001 00000 00101 00000 101111 -> or $5, $1, $0

000000 00010 00001 00110 00000 010100 -> slt $6, $2, $1

test\_2.txt

010011 00000 00010 0000000000000111 -> addi $2, $0, 7

010011 00000 00011 0000000000001000 -> addi $3, $0, 8

010011 00000 00100 0000000000001001 -> addi $4, $0, 9

010011 00000 00101 0000000000001010 -> addi $5, $0, 10

101000 00000 00010 0000000000000100 -> sw $2, 4($0)

101000 00011 00011 0000000000001000 -> sw $3, 8($3)