ACHD Final Project-Part 1

Deliverables (per team):

- 1. VHDL code of the complete processor and its individual components. (along with testbenches)
- 2. Screenshots of Functional Simulation of all the individual components of the processor. (Please use a test bench to simulate)
- Screenshots of Functional Simulation of the complete integrated processor. Use the two sample codes provided and show the results (relevant register contents, data memory and program counter value).
- 4. RC5 code in assembly

Submit 2-3 in a single PDF. For VHDL code, only submit your .vhd files.

In addition, also submit Two-Three PowerPoint slides summarizing your progress on NYU classes. You will be using these slides to present it to the professor during the class (approx. 5min per team)

Sample code 1:

0000010000000010000000000000111	ADDI R1, R0, 7	// R1 = 7
000001000000010000000000001000	ADDI R2, R0, 8	// R2 = 8
0000000010000010001100000010000	ADD R3, R1, R2	// R3 = R1 + R2 =15
111111000000000000000000000000000000000	HAL	// HALT

Sample code 2:

```
000001 00000 00001 0000000000000010
                                                               //R1=R0+2(decimal)
                                          --ADDI R1, R0, 2
000001 00000 00011 0000000000001010
                                          --ADDI R3, R0, 10
                                                               //R3=R0+10(decimal)
000001 00000 00100 0000000000001110
                                                               //R4=R0+14(decimal)
                                          --ADDI R4, R0, 14
000001 00000 00101 0000000000000010
                                          --ADDI R5, R0, 2
                                                               //R5=R0+2
001000 00011 00100 00000000000000010
                                          --SW R4, 2(R3)
                                                               //Mem[R3+2]=R4
001000 00011 00011 00000000000000001
                                          --SW R3, 1(R3)
                                                               //Mem[R3+1]=R3
000000 00100 00011 00100 00000 010001
                                          --SUB R4, R4, R3
                                                               //R4=R4-R3
000010 00000 00100 0000000000000001
                                          --SUBI R4, R0, 1
                                                               //R4=R0-1(decimal)
000000 00011 00010 00100 00000 010010
                                          --AND R4, R2, R3
                                                               //R4=R2 and R3
000011 00010 00100 0000000000001010
                                          --ANDI R4, R2, 10
                                                               //R4=R2 and 10(decimal)
000000 00011 00010 00100 00000 010011
                                                               //R4= R2 or R3
                                          --OR R4, R2, R3
000111 00011 00010 00000000000000001
                                          --LW R2, 1(R3)
                                                               //R2=Mem[1+R3]
000100 00010 00100 0000000000001010
                                          --ORI R4, R2, 10
                                                               //R4=R2 or 10(decimal)
000000 00011 00010 00100 00000 010100
                                          --NOR R4, R2, R3
                                                               //R4= R2 nor R3
000101 00010 00100 0000000000001010
                                          --SHL R4, R2, 10
                                                               //R4 = R2 << 10(decimal)
                                          --SHR R4, R2, 10
000110 00010 00100 0000000000001010
                                                               //R4=R2 >> 10(decimal)
001010 00000 00101 1111111111111111
                                          --BEQ R5, R0, -2
001001 00100 00101 00000000000000000
                                          --BLT R5, R4, 0
001011 00100 00101 00000000000000000
                                          --BNE R5, R4, 0
001100 000000000000000000000010100
                                          --JMP 20
--HAL
```