Assignment-1

Design a multi-cycle processor, ***NITC-RISC24***. The Little-Endian Computer Architecture is the foundation for developing the 16-bit ***NITC-RISC24*** processor. It (***NITC-RISC18***) is an 8-register, 16-bit computer system. It uses registers R0 to R7 for general purposes. However, register R0 always stores the program counter. This architecture also uses a condition code register, which has two flags: the Carry flag (C) and the Zero flag (Z). Both instruction memory and data memory are word-addressable (1 address location is 16-bit data/instruction)

The ***NITC-RISC18*** is very simple, but it is general enough to solve complex problems with three machine-code instruction formats (R, I, and J type) and a total of eight instructions shown below:

R-Type Instructions					
OPCODE	RA	RB	RC	Unused	Condition (CZ)
4-bit	3-bit	3-bit	3-bit	1-bit	2-bit

I-Type Instructions			
OPCODE	RA	RB	Immediate
4-bit	3-bit	3-bit	6-bit

J-Type Instructions		
OPCODE	RA	Immediate
4-bit	3-bit	9-bit

** RA: Register A, RB: Register B, RC: Register C*
** All immediate values are signed*

***Instruction Encoding*:**

ADD:	0000	RA	RB	RC	0	00
ADC:	0000	RA	RB	RC	0	10
NDU:	0010	RA	RB	RC	0	00
NDZ:	0010	RA	RB	RC	0	01
LW:	1010	RA	RB	6-bit Immediate		
SW:	1001	RA	RB	6-bit Immediate		
BEQ:	1011	RA	RB	6-bit Immediate		
JAL:	1101	RA	9-bit Immediate			

** RA: Register A, RB: Register B, RC: Register C*

** All immediate values are signed*

***Instruction Description*:**

Name	**Type**	**Assembly**	**Action**
ADD	R	add rc, ra, rb	ADD the contents of ra and the contents of rb and store the result in rc. It modifies the carry flag and zero flag.
ADC	R	adc rc, ra, rb	ADD the contents of ra and the contents of rb and store the result in rc if the carry flag is set. It modifies the carry flag and zero flag.
NDU	R	ndu rc, ra, rb	NAND the contents of ra and the contents of rb and store the result in rc. It modifies the zero flag.
NDZ	R	ndz rc, ra, rb	NAND the contents of ra and the contents of rb and store the result in rc if the zero flag is set. It modifies the zero flag.
LW	I	lw ra, rb, imm	Load value from data memory into ra. The memory address is formed by adding the contents of rb with immediate 6 bits. It modifies the zero flag.
SW	I	sw ra, rb, imm	Store the contents of ra into data memory. The memory address is formed by adding the contents of rb with immediate 6 bits.
BEQ	I	beq ra, rb, imm	If the contents of ra and rb are the same, then branch to (PC+imm), where PC is the address of the beq inst.
JAL	J	jal ra, imm	Branch to (PC+imm) Store (PC+1) into ra, where PC is the address of jal inst.

* *RA: Register A, RB: Register B, RC: Register C*
* *All immediate values are signed*