

DA RESTITUIRE INSIEME AGLI ELABORATI e A TUTTI I FOGLI

→ NON USARE FOGLI NON TIMBRATI

→ ANDARE IN BAGNO PRIMA DELL'INIZIO DELLA PROVA

→ NO FOGLI PERSONALI, NO TELEFONI, SMARTPHONE/WATCH, ETC

COGNOME _____

NOME _____

NOTE: I FOGLI UTILIZZATI PER RAGIONAMENTI VANNO RICONSEGNA TI ANCHE SE BIANCHI; PER I FILE:

- per l'esercizio 3 consegnare un file di testo di nome

<COGNOME>.txt

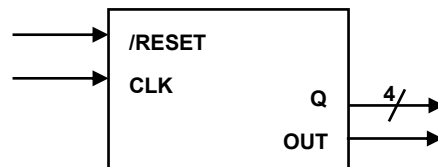
- per l'esercizio 4 consegnare DUE files: il file del programma VERILOG di nome

<COGNOME>.v

e il file del diagramma temporale (screenshot o copy/paste → usare tasto 'STAMP')

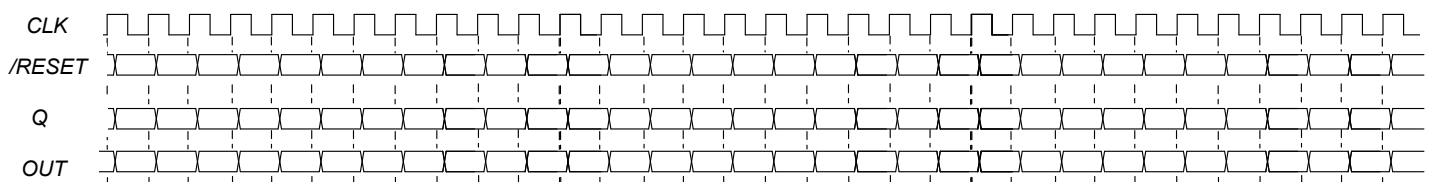
<COGNOME>.png

- 3) [7/30] Fornendo una spiegazione ragionata, con il dettaglio del significato dei vari bit per il formato dell'istruzione, determinare a quale istruzione assembly RISC-V corrisponde la seguente stringa binaria (codice macchina) 0000 0001 1110 1011 0010 1000 1011 0011 ricordando che i codici operativi (opcode/funct3/funct7 in esadecimale) delle principali istruzioni viste sono: ADD: 33/0/00, LD: 03/3/imm, SD: 23/3/imm, BEQ: 63/0/(imm÷2), LUI: 37/imm[31:12], SLT: 33/2/0
- 4) [23/30] **Realizzare** in Verilog un serial carry counter a 4-bit che effettui un conteggio verso il basso decrementando di 2 unità ad ogni colpo di clock CLK e basato su opportuni Flip-Flop T. Il conteggio comparirà sull'uscita Q. Al raggiungimento del valore 0 l'uscita OUT passerà ad 1 (altrimenti vale 0). Il conteggio continuerà poi ciclicamente. Gli stimoli di ingresso sono dati dal seguente modulo Verilog Testbench.



Tracciare il diagramma di temporizzazione come verifica della correttezza dell'unità riportando i segnali CLK, /RESET, uscita Q e uscita OUT per la durata complessiva.

Nota: si può svolgere l'esercizio su carta oppure con ausilio del simulatore salvando una copia dell'output (diagramma temporale) e del programma Verilog su USB-drive del docente.



```

module TopLevel;
  reg CLK; reg RESET_; always #10 CLK<=(!CLK);
  wire[3:0] Q; wire OUT;
  initial begin
    RESET_ = 1'b1; CLK = 0;
    #2 RESET_ = 1'b0; #5 RESET_ = 1'b1; #320 $finish;
  end
  MyCounter mc(Q,OUT, CLK,RESET_);
  //debug:
  wire q0=mc.q0, q1=mc.q1, q2=mc.q2, q3=mc.q3;
endmodule
  
```

ESERCIZIO 3

Dai primi 7 bit vediamo che l’opcode e’ 0b011 0011 ed un eventuale funct3 e’ 2 ovvero opcode/funct3=0x33/0x2: questo corrisponde alla istruzione SLT che ha un formato R nel RISC-V. Quindi i restanti bit possono essere raggruppati come segue:

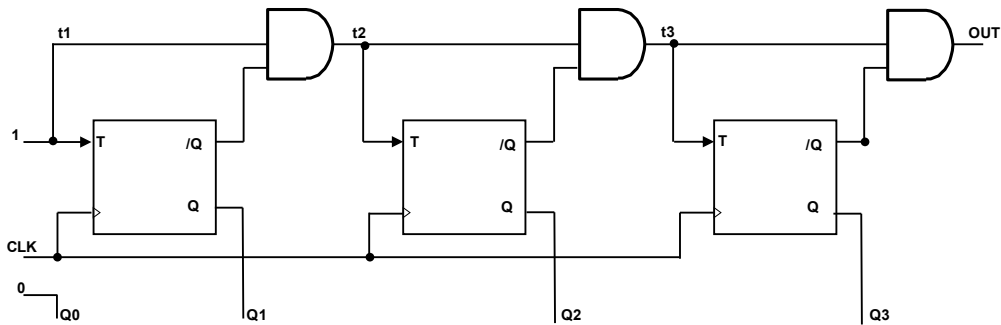
0000 000	11110	10110	010	1000 1	011 0011
funct7	rs2	rs1	funct3	rd	opcode

A questo punto si possono interpretare i campi rs1,rs2,rd che quindi corrispondono rispettivamente ai registri x22, x30 e x17. Quindi l’istruzione corrispondente è:

slt x17, x22, x30

ESERCIZIO 4

Si può far riferimento al seguente schema:



Possibile codice Verilog dei moduli da realizzare:

```
// Flip-Flop T sensibile al fronte in salita
module FFTp(q,qbar, clock,reset_,t);
    input clock, reset_;
    input t;
    output q,qbar;
    reg STAR;
    parameter S0=0,S1=1;
    assign q=(STAR==S0)?0:1,qbar=(STAR==S0)?1:0;
    always @(reset_==0) #1 STAR <= S0;
    always @(posedge clock) if (reset_==1) #1
        casex(STAR)
            S0: STAR <= (t==0)?S0:S1;
            S1: STAR <= (t==0)?S1:S0;
        endcase
endmodule
```

```
module MyCounter(Q,OUT, clock,reset_);
    input clock, reset_;
    wire q0,q1,q2,q3,
        q1bar,q2bar,q3bar,t1,t2,t3;
    output[3:0] Q; output OUT;
    assign q0 = 0;
    assign t1=1, t2=q1bar&t1,
        t3=q2bar&t2, OUT=q3bar&t3;
    FFTp fp0(q1,q1bar, clock,reset_,t1);
    FFTp fp1(q2,q2bar, clock,reset_,t2);
    FFTp fp2(q3,q3bar, clock,reset_,t3);
    assign Q[3]=q3, Q[2]=q2, Q[1]=q1, Q[0]=q0;
endmodule
```

Diagramma di TempORIZZAZIONE:

