COMPITINO di ARCHITETTURA DEI CALCOLATORI del 06-11-2023

MATRICOLA_____

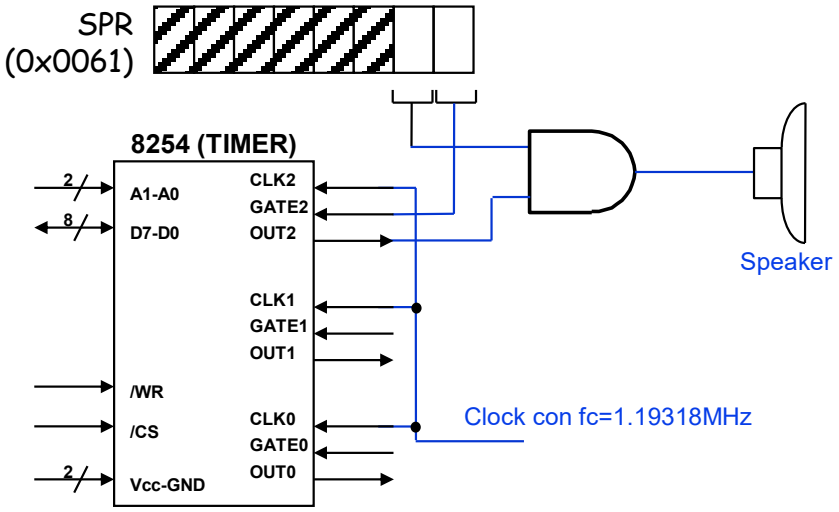
COGNOME_____

NOME_____

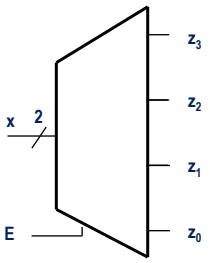
DA RESTITUIRE INSIEME AGLI ELABORATI e A TUTTI I FOGLI
→ NON USARE FOGLI NON TIMBRATI
→ ANDARE IN BAGNO PRIMA DELL'INIZIO DELLA PROVA
→ NO FOGLI PERSONALI, NO TELEFONI, SMARTPHONE/WATCH, ETC

NOTE: I FOGLI UTILIZZATI PER RAGIONAMENTI VANNO RICONSEGNATI 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

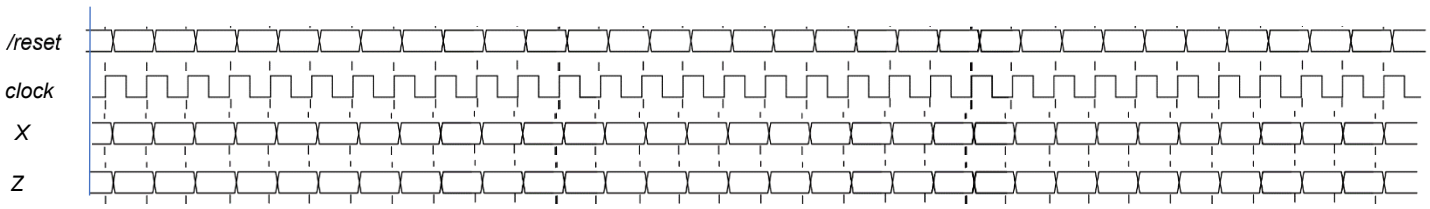
3) [10/30] Descrivere la procedura di programmazione del timer 8254 mappata a partire dall'indirizzo 0x900003E8 e la porta SPR (indirizzo 0x9000'00061), per ottenere sullo speaker dello schema di figura il suono corrispondente al suono standard per gli accordi (nota "LA" a 440 Hz)? Si ricorda che la frequenza esterna di tale chip è pari a 1.19318 MHz. Quali sono i registri del timer da programmare unitamente a SPR? E quali i valori da scrivere in tali registri?



4) [20/30] Il "decoder con abilitazione" è un decoder con un ingresso aggiuntivo E (Enable) che abilita le uscite quando E=1 mentre quando E =0 tutte le uscite valgono 0: la tabella di verità nel caso da-2-a-4 è rappresentata nella figura a lato. Realizzare in Verilog tale decoder con abilitazione da-2 a-4 ([punti 8]) e successivamente realizzare in Verilog un decoder "da-3-a-8" utilizzando la descrizione del decoder con abilitazione "da-2-a-4" ([punti 8]). **Tracciare il diagramma di temporizzazione ([punti 4])** come verifica della correttezza dei moduli realizzati, utilizzando il testbench fornito di seguito. Modello del diagramma temporale da tracciare:



INGRESSI			USCITE			
E	x_1	x_0	z_3	z_2	z_1	z_0
0	X	X	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0



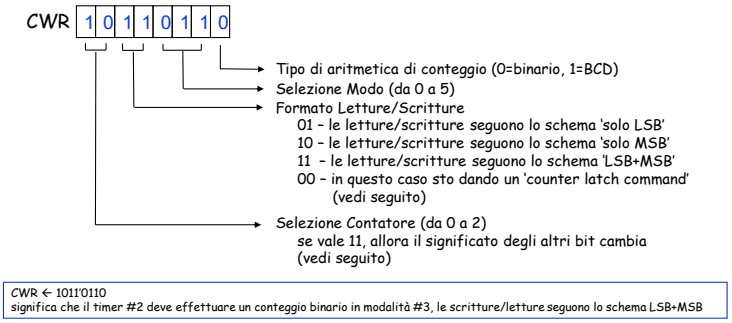

```
module testbench;
  reg reset_;initial begin reset_=0; #22 reset_=1; #120; $stop; end
  reg clock ;initial clock=0; always #5 clock <=(!clock);
  reg[2:0] X; wire[7:0] Z;
  always #10 if (reset_==1) X <= X+1;
  initial begin
    X = 0;          // Initialize inputs
    $monitor("Input: %b, Output: %b", X, Z); // Monitor outputs
    #100 $finish;  // End simulation
  end
  Decoder_3to8 decoder(Z,X);
endmodule
```

COMPITINO di ARCHITETTURA DEI CALCOLATORI del 06-11-2023
SOLUZIONE

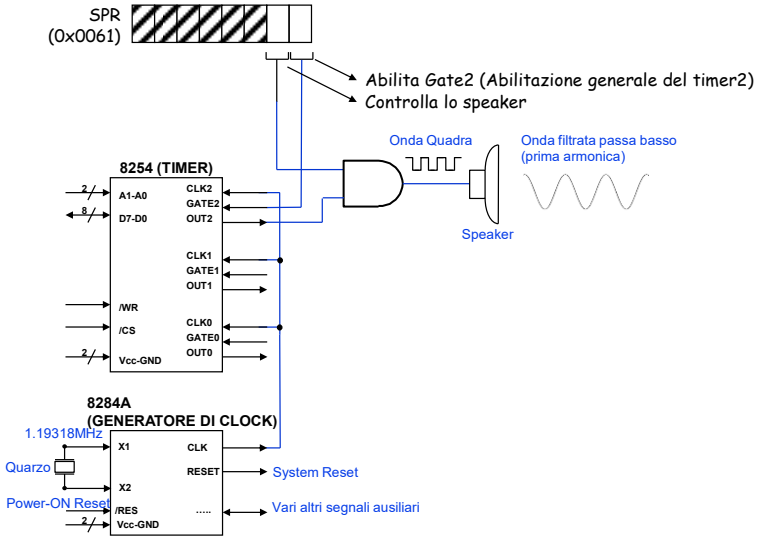
ESERCIZIO 3

Per generare la nota a frequenza f_{LA}=440Hz possiamo avvalerci del modo di programmazione n.3 del timer al fine di generare un'onda quadra la cui prima armonica sia a tale frequenza. Dato che fc=1.19318, la costante di tempo a 16 bit sarà $N=fc*deltaT_{LA}=fc/f_{LA}=1.19318*10^6/440 \cong 2712 = 0x0A98$.

Dobbiamo innanzitutto mettere a 1 il bit0 del registro SPR (indirizzo 0x9000'0061) per abilitare il pin GATE2 del timer. Per fare questo occorre prima leggere il contenuto di SPR, mascherarlo (operazione OR) per impostare solo tale bit e quindi riscrivere il valore così aggiornato in SPR (al fine di non alterare gli altri bit, in particolare il bit1 per evitare suoni spuri sull'uscita dello speaker).
Quindi si può procedere alla programmazione del timer, scrivendo la parola di controllo 0xB6 nel registro CWR ad indirizzo 0x9000'03EB, difatti:

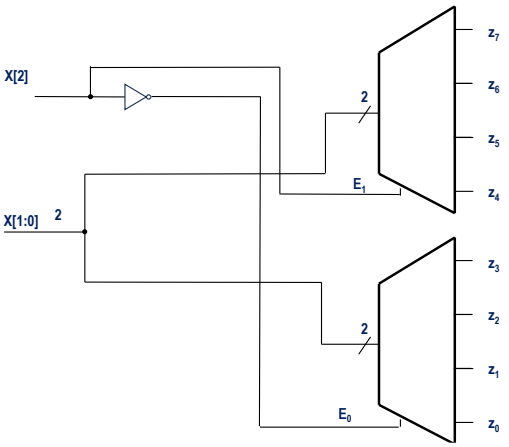


Dopodiché andranno effettuate due scritture successive dei byte 0x0A e 0x98 all'indirizzo del timer-2 0x9000'03EA, seguendo lo schema LSB+MSB. Infine, dovrà essere abilitata l'uscita dello speaker con un'operazione di mascheratura simile a quella effettuata all'inizio su SPR (lettura, mascheramento, scrittura), ma operando stavolta sul bit1 di tale registro.



ESERCIZIO 4

Si può far riferimento al seguente schema:



Possibile codice Verilog dei moduli da realizzare:

```
module Decoder_2to4plusE(z,  x,E);
  output[3:0] z; reg[3:0] z;
  input[1:0] x;
  input E;
  always @(x or E) casex ({E,x})
    3'b0xx: z = 4'b0000;
    3'b100: z = 4'b0001;
    3'b101: z = 4'b0010;
    3'b110: z = 4'b0100;
    3'b111: z = 4'b1000;
endmodule

module Decoder_3to8(z,  x);
  output[7:0] z;
  input [2:0] x;

  Decoder_2to4plusE d1(z[7:4], x[1:0], x[2]);
  Decoder_2to4plusE d0(z[3:0], x[1:0],~x[2]);
endmodule
```

Diagramma di Temporizzazione: