

Lezione 8 - Esempi di dispositivi di I/O: il TIMER

- Come riferimento consideriamo il timer Intel-8254
- A cosa può servire: *un timer*
 - Generare ritardi programmabili in maniera accurata *evento con ritardo ben definito*
 - Real time clock *Riferimento temporale, tempo reale*
 - Conteggio eventi *che si verificano in un lasso di tempo*
 - Generatore di frequenza programmabile *es. onda quadra, o per audio*
 - Generatore di onda quadra programmabile
 - Generatore di "impulso digitale" *es. per interrupt*
 - Moltiplicatore binario di frequenza *moltiplicano in freq. una forma d'onda*
 - Generatore di forma d'onda complessa
 - Controllo di motori
- *Timer circuito indigenabile*
- È un buon esempio per mostrare la complessità interna di un'interfaccia di I/O
- Datasheet reperibili sul sito del corso

<https://web.archive.org/web/20150603122044/https://download.intel.com/design/archives/periphrl/docs/23124406.pdf>

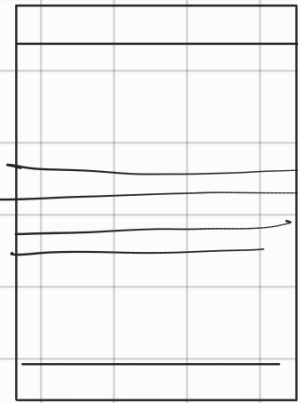
Il chip, uno dei reperti interni, dovuto indovinare quei reperti.

Come collego timer al processore?

(1) Devo riuscire dal processore ad indirizzare i registri interni del timer (che ho 4)

Sforzo di indirizzamento

l'idea è far comparire quei registri nello spazio di indirizzamento



Scelgo 0x40... 0x43

? Qual è l'hardware che mi permette di far apparire questi registri nel chip in modo tale che posso vederli dal processore?

MECANISMO DI INDIRIZZO!

Ho l'indirizzo su 64 bit, genero il segnale di chip select e i 2 bit di indirizz. interno del chip, come uscita del blocco collegamento F000

• D7-D0 collegata a Data della CPU, sono gli 8 bit meno signific.

• Write e read (negati) collegati al R/W della CPU

=> Voglio ora generare indirizzo 0x40... 0x43

Come genero chip select? Porta NAND che riconosce in ingresso tutta la 16 (4x4) cifre esadecimali (scarto 2 bit meno signific.)

↳ uscita 1 quando presente indirizzo 00... 40 (anche 41...43), non ho meno i bit meno significativi

Per generare i 2 più significativi, li collego al A1A0 per riconoscere i 4 indirizzi

genero i 4 indirizzi 40, 41, 42, 43, uso i 2 bit meno significativi, li collego

Riconoscimento: fibbra indirizzo e mappa

quasi di questi chip nello spazio di memoria

al A0A1 per 40, 41, 42, 43

PROCESS. MSC V

imparte neg, e imparte memoria fisica

con cui controllo

• insieme di fili (64) per gli indirizzi

ADDRESS, 64 fili con cui indirizziamo le memorie

• Ho collegato al mio 2 Memorie, Data Memory, instruction memory

• Segnale Read/Write

• Segnale di ingresso a 32 bit con cui preleva le istruzioni

• Segnale clock, di enable, reset

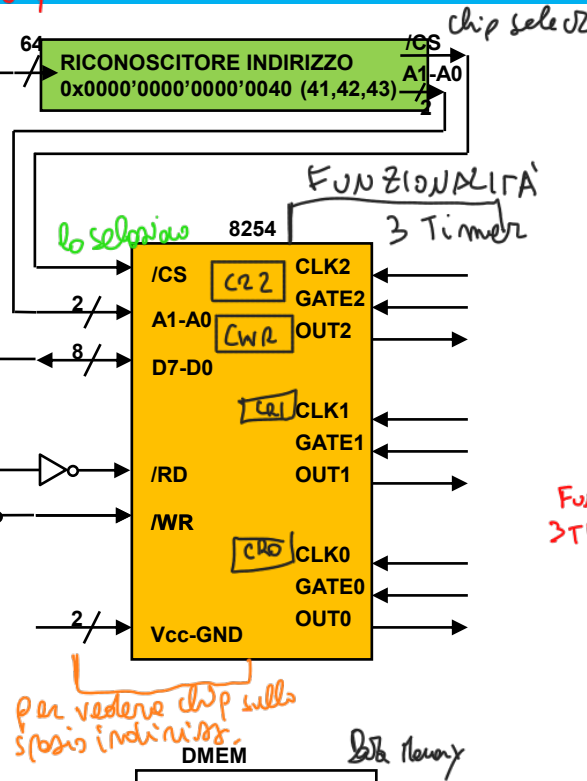
RICONOSCIMENTO INDIRIZZI MAPPA REGISTRI DISP CIO del interno dello SPAZIO DI MEM

SPAZIO MEMORIA -> spazio indirizzi, parte mem fisica, allora saranno registri degli i/o

Timer Intel-8254: Visione elettrica del chip e di insieme

Indice: individuare registri interni nel chip

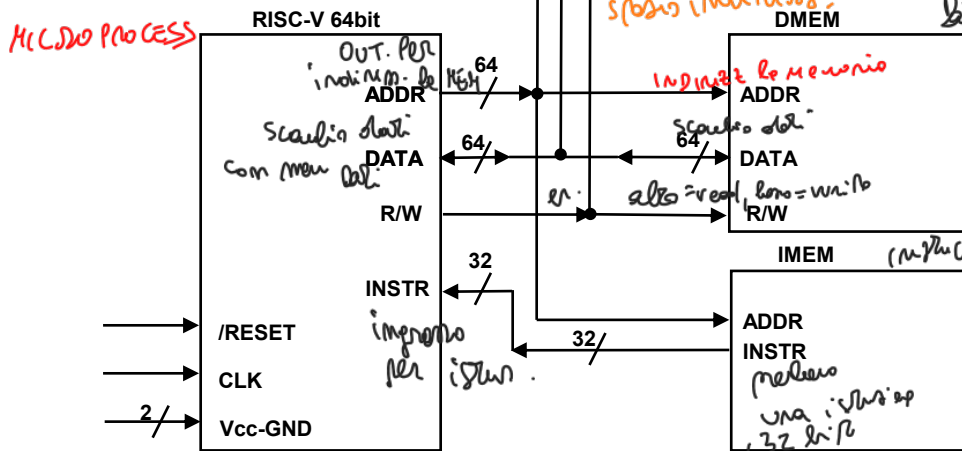
- CS Chip select per abilitare il chip
- all'intorno ci sono 4 registri, hanno 2 fili per l'indirizzo: A1, A0
- Dati: trasmissioni, 8 pin, 4 byte alla volta
- Segnali lettura/scrittura
- Alimentazione
- per spazio indirizzamento
- 3 Timer, riceve in ingresso una freq. di riferimento CLK
- Gate, segnali di controllo
- OUTj, fornisce forma d'onda in uscita



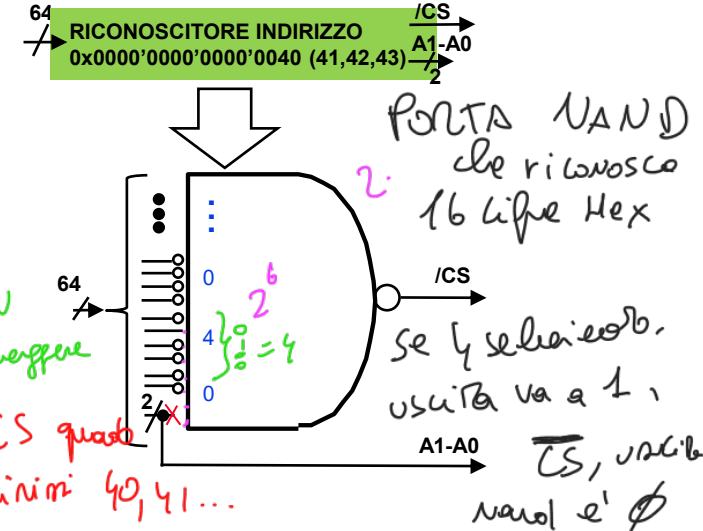
- Segnali di Pilotaggio, DIGITALI
- /CS: Chip-Select *può abilitarlo*
- A1-A0: indirizzamento *2 bit*
- D7-D0: scambio dati su 8 bit *dimensione 1 byte alla volta*
- /RD, /WR: Read, Write
- Vcc-GND: alimentazione

- CLKj: clock del timer j *riceve un riferimento*
- GATEj: trigger hardware o congelamento conteggio

- OUTj: forma d'onda prodotta dal timer j
- Da l'indirizzo della CPU, genera CS -*



Il quando la CPU vuole scrivere, leggere del timer, Genera CS quando presente indirizzi 40, 41...



Timer Intel-8254: Visione Funzionale

collegare timer al processore, il timer ha 3 reg. interni, per inclinarli: uso un riconoscimento di inclinazione
Faccio comparire i 3 registri negli indirizzi 0x40, 0x41, 0x42 così li vedo della CPU

- Visione Funzionale del chip: 3 CONTATORI: registri a 16 bit
- 3 contatori indipendenti a 16-bit
- 6 modalità di conteggio (detti 'modi operativi')

implementano 6 modalità - conteggio

TC period clock un ciclo di clock ovvero T_C

Modalità di conteggio, 3 categorie

Ci servono per un gradino

Forma d'onda periodica

Genero impulso

per un ciclo voglio uno 0 in uscita

MODO	FUNZIONE	Condizione iniziale <i>uscita floating X se</i>	Timing generato <i>in uscita avrò un segnale che per N cicli di clock va a 0, e poi ritorna a 1</i>
0	Interrupt al termine del conteggio	$OUT=X, GATE=1$ <i>condizione gate</i>	$OUT(N \cdot T_C)=0, OUT(\infty)=1$
1	Programmable one shot	$OUT=1, GATE=0$ <i>Gate=0, uscita va inizialmente a 1 subito</i>	$GATE=1 \rightarrow OUT(N \cdot T_C)=0, OUT(\infty)=1$ <i>comando da gate 0 a 1, l'uscita va a 0 per N cicli</i>
2	Rate generator <i>genero onda periodica con frequenza di freq. $1/T_C$</i>	$OUT=X, GATE=1$	$OUT((N-1) \cdot T_C)=1, OUT(1 \cdot T_C)=0, \dots$ <i>e poi periodico si ripete</i>
3	Square wave rate generator	$OUT=X, GATE=1$	$OUT(\sim N/2 \cdot T_C)=1, OUT(\sim N/2 \cdot T_C)=0, \dots$ <i>N ciclo, se è dispari, divisione, approssimazione ciclo ~ per N/2 cicli è 1 poi per N/2 sta a 0</i>
4	Software triggered strobe	$OUT=X, GATE=1$	$OUT(N \cdot T_C)=1, OUT(1 \cdot T_C)=0, OUT(\infty)=1$ <i>alla per N cicli 0 per un ciclo ritorna a 1</i>
5	Hardware triggered strobe <i>quando cambia il gate</i>	$OUT=1, GATE=0$	$GATE=1 \rightarrow OUT(N \cdot T_C)=1, OUT(1 \cdot T_C)=0, OUT(\infty)=1$ <i>genero un impulso dopo N cicli comando gate con HW esterno N cicli a 1, 1 ciclo a 0 e ritorna a 1</i>

- GENERAZIONE DI UN GRADINO (verso uno)
- GENERAZIONE DI ONDA PERIODICA
- GENERAZIONE DI UN IMPULSO (verso zero)

Timer Intel-8254 (2)

FUNZIONAMENTO Registri: internamente ho contatore a 16 bit, però li vedo esternamente come registri da 8 bit, posso scrivere solo 8 bit alla volta

- Visione logica del chip (tutti registri a 8 bit):

Data Register

Address	RD / WR	Register Name	Description
00 1 0 0x40		Counter Register 0	Definisce la N , 16 BIT Scrivendo in questo registro fisso la Time Constant 0
01 1 0 0x41		Counter Register 1	Scrivendo in questo registro fisso la Time Constant 1
10 1 0 0x42		Counter Register 2	Scrivendo in questo registro fisso la Time Constant 2

DR
Data Register
scanditi da!

Scheda: 2 contatore, legge

Control Register

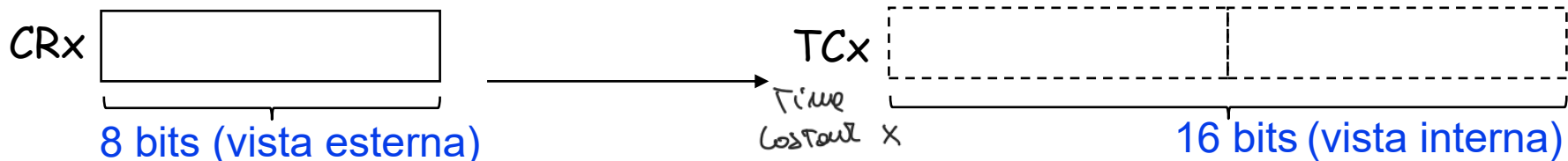
Address	RD / WR	Register Name	Description
11 1 0 0x43		Control Word Register	Scrivendo in questo registro fisso il funzionamento di un dato contatore oppure dà comandi particolari

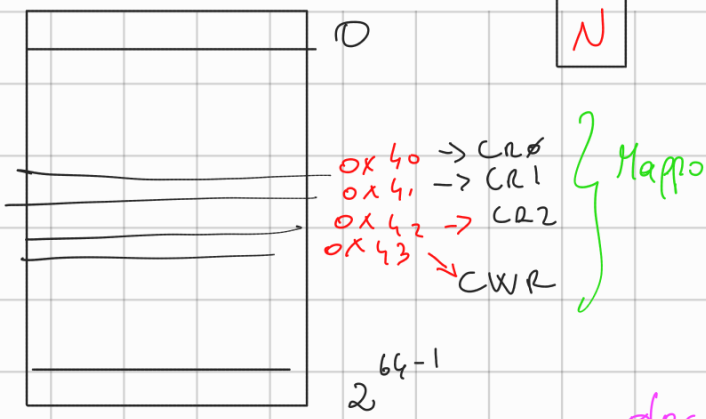
CR
Control Register
per controllo del chip

Control Register

Nota: la Time Constant è a 16 bit. CRj funziona come un "bus" a 8 bit verso un registro interno a 16 bit che contiene tale Time Constant

*vista esterna a 8 bit
internamente registri a 16 BIT*





8 bit = 1 Byte

decido una N e voglio scriverlo
in un registro. N a 16 bit
 N = costante di tempo

CWR = Control Word Register

decido una N , voglio scriverlo in
uno dei registri, dove N è da 16 bit,
ma i registri CRX sono da 8 bit, legati ad
un registro interno di 16 bit

0x43 CWR Registro di Controllo a 8 bit qui imposto il
modo di funzionamento

Data Register

- In un chip di I/O ho sempre registri DR con cui
scambio dati, e insieme di registri che
mi servono per effettuare il controllo del
chip. (Control Registers, CR)

Paio di dati specifici attraverso la possibilità di
lettura / scrittura

Suddiviso 16 bit in 2 gruppi da 8

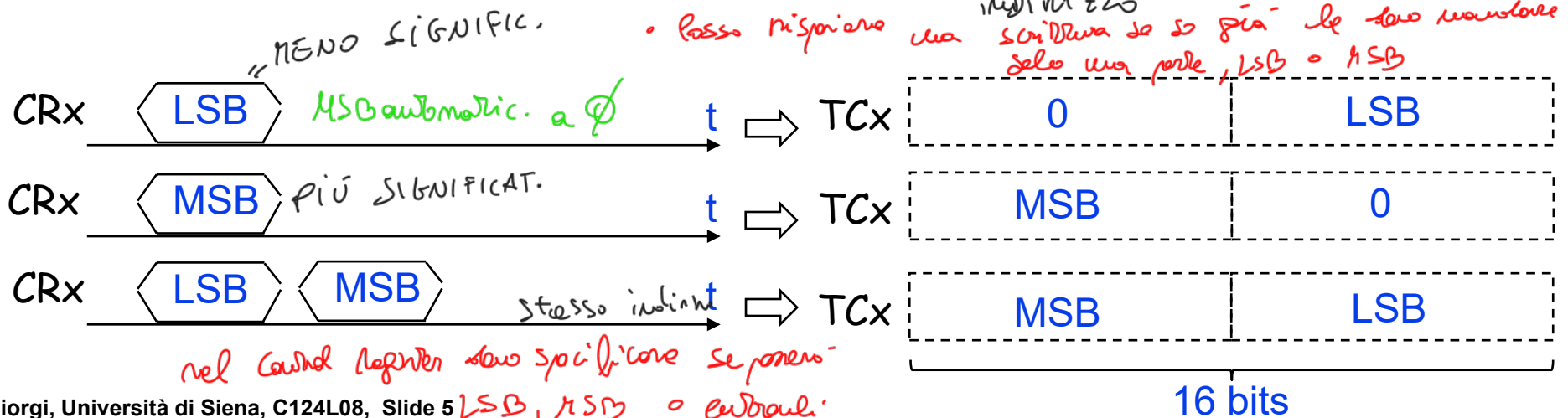
- Nei dispositivi I/O sono sempre Data Registers
con cui scambio dati, e altri per controllarli
(CONTROL REGISTERS)

Modalità di lettura/scrittura

Programmas. address controls

- Poiché nei dispositivi di I/O si tende ad utilizzare un numero limitato di indirizzi (nel nostro caso 4) si usa un registro per dare i comandi (CWR) e gli altri registri per leggere o scrivere dati (CR0, CR1, CR2) *data register*
- Dato che i contatori sono **internamente** a 16 bit e posso scrivere solo 8 bit (1 byte) per volta: in quale ordine vengono inviati tali byte?
 - LSB+MSB: invio prima il byte meno significativo (Least Significant Byte o LSB) e poi su quello più significativo (Most Significant Byte o MSB) *trasferisco in sequenza prima quello meno significativo e poi quello più significativo*
- posso risparmiare un invio se uno dei due byte è zero
 - LSB: invio solo il byte meno significativo (es. 0x00A4 → invio solo 0xA4)
 - MSB: invio solo il byte più significativo (es. 0x B500 → invio solo 0xB5)

scrivo prima LSB e poi MSB, nello stesso



Programmazione del timer

CW GATE ABILITO CONTEGGIO

- 1) Scrivere in CWR ^{nella posizione 43} per selezionare uno dei contatori, modo, ordine LSB/MSB
- 2) Scrivere in CRj la costante di tempo poi si avvia la costante di tempo
- 3) Abilitare il conteggio con un segnale alto sul pin GATE così l'uscita OUT varia
- 4) L'uscita OUT produrrà un segnale di uscita alla fine del (oppure durante il) conteggio a seconda del modo prescelto

Cosa scrivo in CWR



3 bit

Tipo di aritmetica di conteggio (0=binario, 1=BCD) ^{non lo verifichiamo}

Selezione Modo (da 0 a 5) ^{numero da 0 a 7 (0 a 5, la 6 è modalità)}

Formato Letture/Scritture

01 - le letture/scritture seguono lo schema 'solo LSB'

10 - le letture/scritture seguono lo schema 'solo MSB'

11 - le letture/scritture seguono lo schema 'LSB+MSB'

• 00 - in questo caso sto dando un 'counter latch command' (vedi seguito)

^{primi 3 bit, selezione}
CR0, 1, 2

Selezione Contatore (da 0 a 2) ^{CR0 / CR1 / CR2} ^{mi dice quale degli programmi di conteggio}
• se vale 11, allora il significato degli altri bit cambia (vedi seguito) ^{es. 01, voglio programmare CR1}

Es. CWR ← 10110110

significa che il timer #2 deve effettuare un conteggio binario in modalità #3, le scritture/letture seguono lo schema LSB+MSB

CONTATORE SCORRE IN BASSO ed ogni ciclo di clock
Come legge conteggio

Lettura del conteggio (1)

interrompere, il conteggio scende velocemente

- Ci sono tre maniere per leggere il valore di un contatore
 - 1) lettura semplice *del conteggio* accesso al registro CRj e legge, *previo blocco*
 - 2) Counter Latch Command
 - 3) Read-Back Command

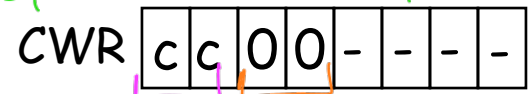
legge da CRj come se fosse un registro

1) La lettura semplice consiste nel leggere il valore di CRj, *le cifre scorrono*,
previo bloccaggio del conteggio tramite il segnale GATEj *lo blocca dentro gate*

- Se non si blocca il conteggio il valore letto può essere inconsistente
legge mentre scorrono le cifre, deve bloccare alla lettura con gate, allora blocca il conteggio

slide e più preciso
2) Counter-Latch Command *non interrompe conteggio ma copia il contatore*

Copia il contatore che specifico con bit più significativi

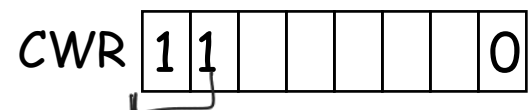


contatore 0,1,2
invece comando di memorizzazione conteggio
01,10,11 riserati a LSB

- (cc=0, 1 o 2 indica il contatore, '-' indica non specificato)
- Lettura di 1 o 2 byte da CRcc (a seconda dello schema di lettura/scrittura precedentemente impostato)

→ a seconda del contatore specificato, il contatore viene copiato internamente e lo posso successiv. leggere

3) Read-Back Command



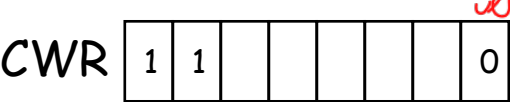
- V. slide successiva *porta acceso per read back*

si abilita con 11, porta di accesso per read back, e mette bit meno sign. a 0

Letture del conteggio (2)

• **Read-Back Command** *i bit allora cambiano significato*

1) Si realizza in questo modo: scrittura in CWR delle seguenti info



posso memorizzare più contatori simultaneamente.

- CNT0 - se 1 seleziona il contatore 0 *eliminazione di uno dei 3 contatori*
- CNT1 - se 1 seleziona il contatore 1
- CNT2 - se 1 seleziona il contatore 2

- leggo il CWR stesso* /STATUS - se 0 nel latch di uscita va la parola di comando *ultima data a quel contatore*
- leggo il conteggio* /COUNT - se 0 nel latch di uscita va il conteggio

leggo nei CRx il valore

2) Successiva lettura di CRx

im uscita va il conteggio

valore a leggere nei register il valore del latch, valore memorizzato a quell'indirizzo in cui ho dato il CWR

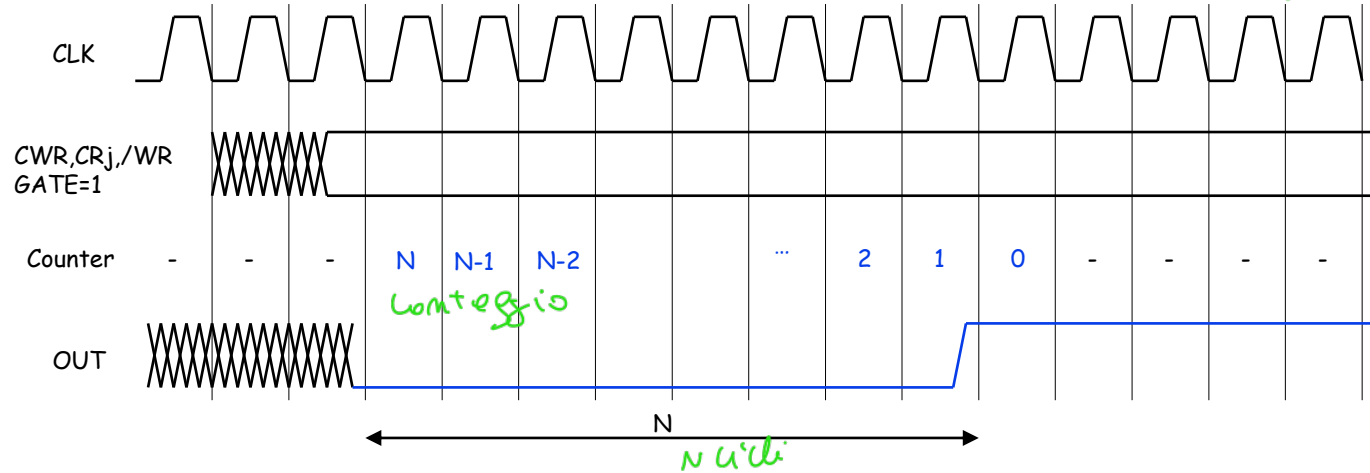
• **Vantaggi:**

- Consente di leggere i conteggi di più contatori simultaneamente (caso di bit /COUNT attivo) *valore del Latch memorizzato in quel numero*
- Consente di leggere come sono stati programmati i 3 contatori (caso di bit /STATUS attivo)
 - In particolare, i 6 bit meno significativi danno i valori precedentemente scritti in CWR per quel contatore

veloce e efficiente

Modi per generare un gradino dopo il conteggio

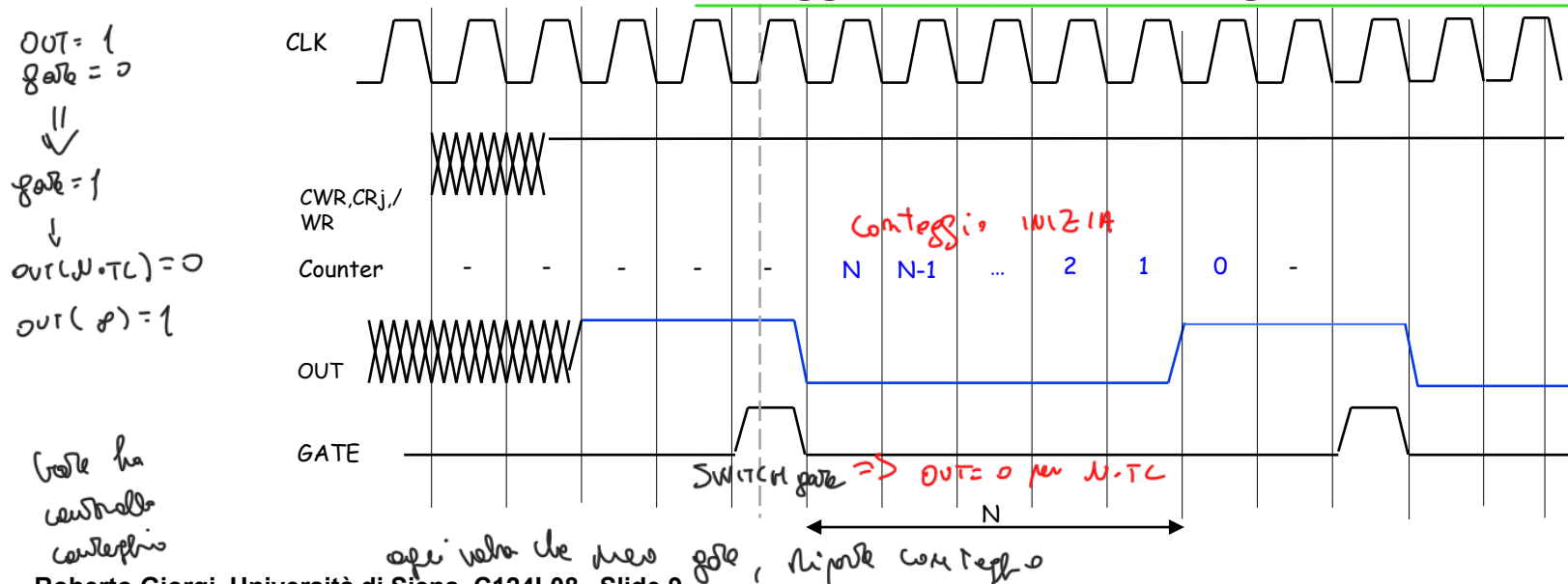
- **Modo 0: conteggio abilitato dalla scrittura di CR** *gradino di livello del conteggio zero* $OUT \times gate = 1$
 $\Rightarrow OUT = 0$ per $N-TL$ poi 1



Nota: usato ad esempio per generare un interrupt alla fine di un conteggio

di N cicli

- **Modo 1: one shot (conteggio abilitato dal segnale hardware GATE)** *conteggio segnale con gate*



Nota: mentre OUT è agganciato ai fronti in discesa del clock (CLK), il segnale GATE viene viceversa campionato sui fronti in salita del clock. (per approfondimenti vedere manuale del chip 8254)

Modi per generare un'onda periodica

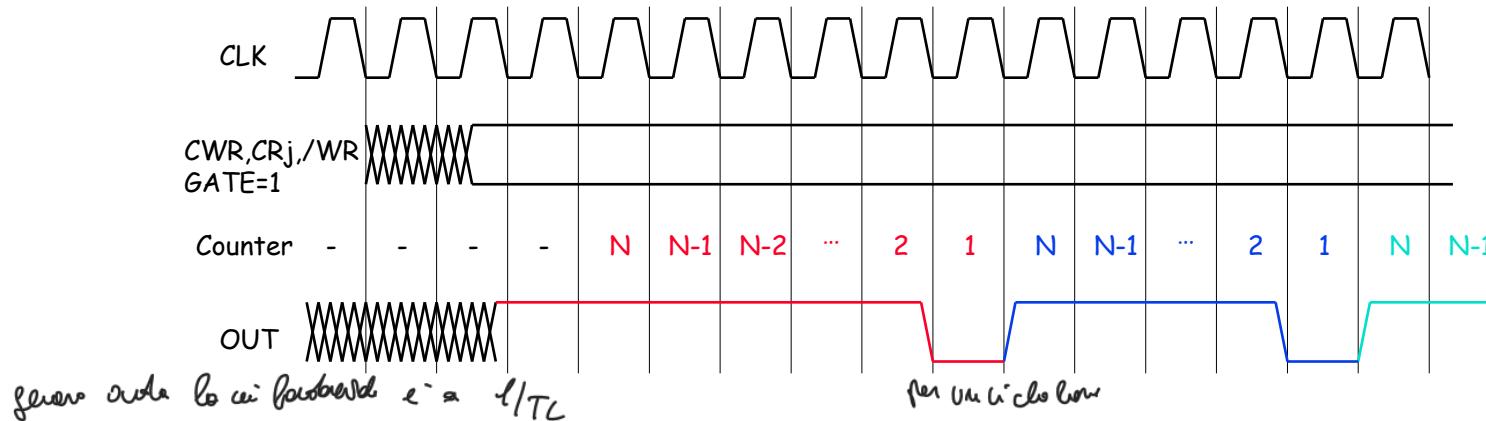
$$1) OUT = x, GATE = 1 \rightarrow OUT(N-1)TC = 1$$

$$OUT(1-TC) = 0$$

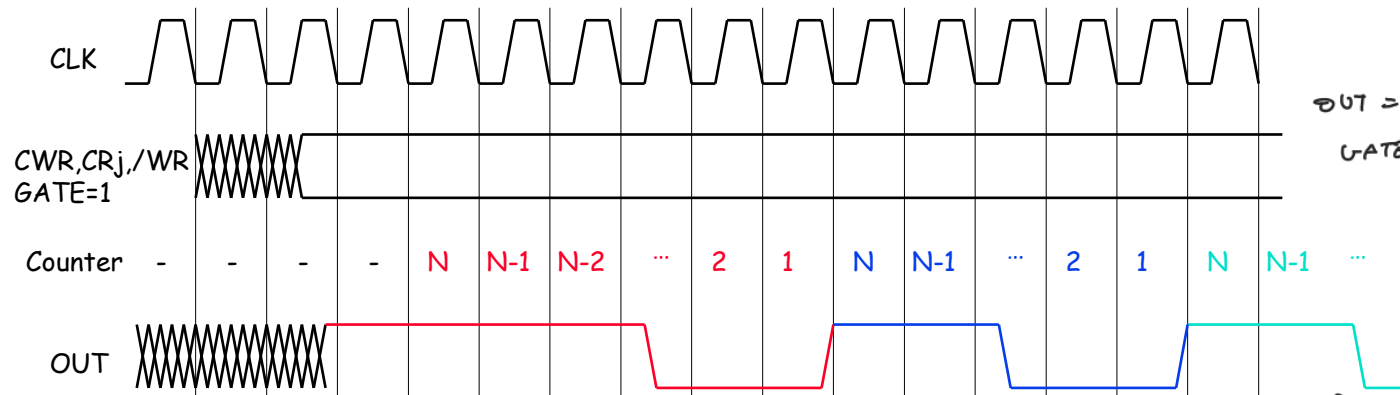
- **Modo 2: rate generator (divisore di frequenza)**
 → genera un onda quadra con duty-cycle pari a $(N-1)/N$

Nota1: Se f è la frequenza del segnale di clock (CLK), la forma d'onda d'uscita (OUT) avrà frequenza f/N

Nota2: L'uscita è inizialmente alta e diventa bassa, per la durata di un ciclo, quando il contatore raggiunge il valore 1



- **Modo 3: square wave mode (generatore d'onda quadra)**



- OUT rimane alto nei periodi $N, N-1, \dots, (N+1)/2$
- OUT rimane basso nei periodi $N/2, (N-1)/2, \dots, 2, 1$
- Il duty cycle è $\frac{1}{2}$ se N è pari, altrimenti è $((N+1)/2)/N$

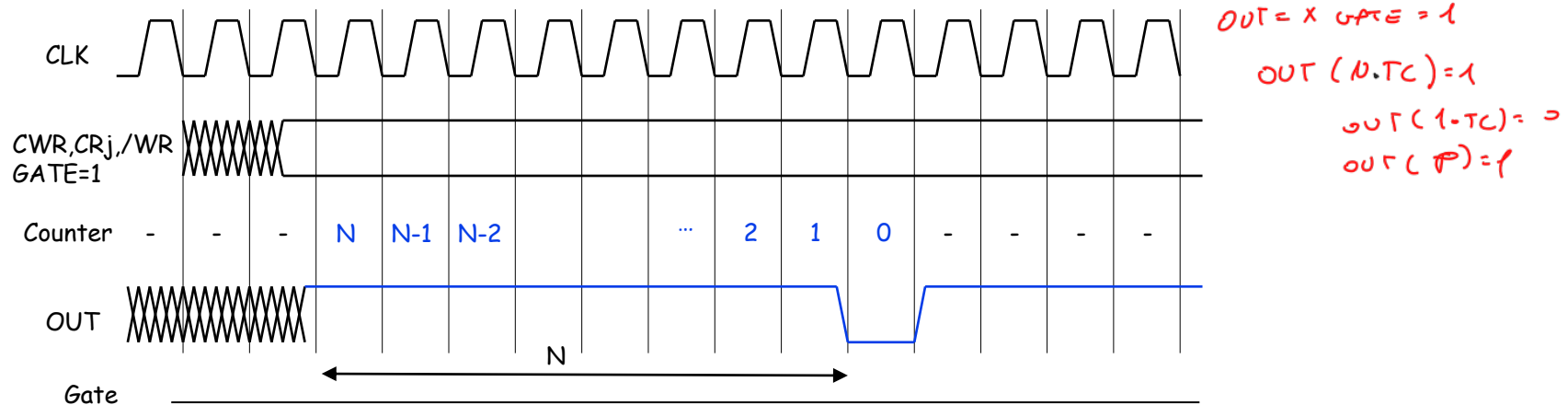
simile a periodo

$$f_{req} = \frac{1}{N \cdot TC} = \frac{f_c}{N}$$

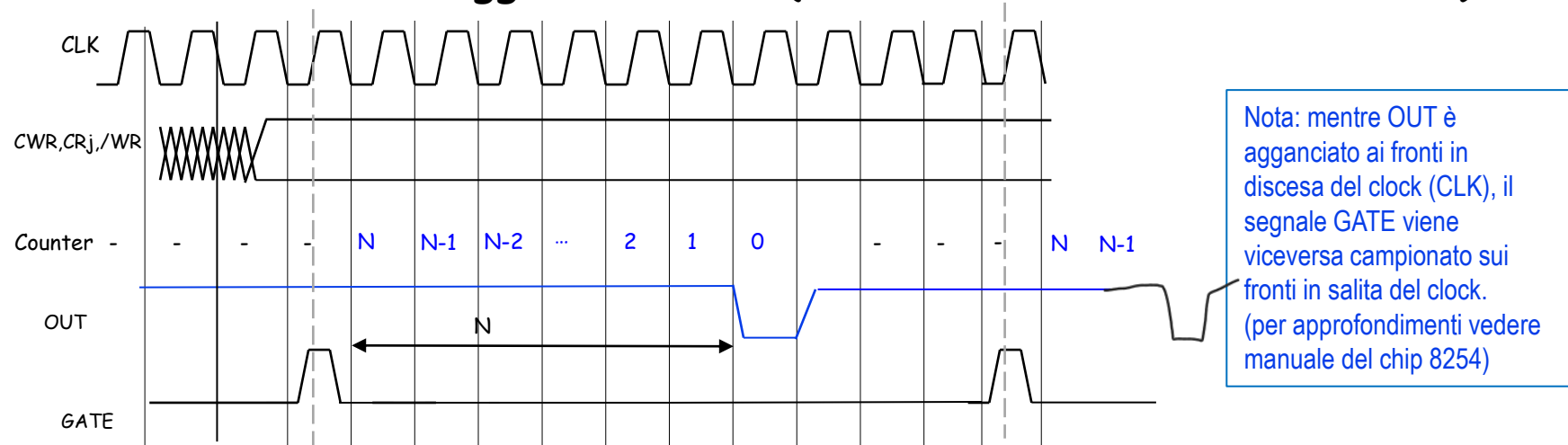
Modi per generare un impulso a fine conteggio

impulso quando scivolo N, se Gate = 1
IMPULSO

- **Modo 4: software triggered strobe (abilitato alla scrittura di CR)**



- **Modo 5: Hardware triggered strobe (abilitato ad hardware da GATE)**



OUT(00)=1

Riepilogo risorse 8254 nei Personal Computer

allora nel PC ci sono 3 TIMER

Chip name	Register name		I/O port	IRQ
Intel 8254	CR0 TIMER DI SISTEMA	Counter Register 0	0x0040	IRQ0
Intel 8254	CR1	Counter Register 0	0x0041	-
Intel 8254	CR2	Counter Register 0	0x0042	-
Intel 8254	CWR	Control Word Register	0x0043	-

Glue logic	SPR	Speaker Register	0x0061	
------------	-----	------------------	--------	--

• CLK0=CLK1=CLK2 = 1.19318MHZ

• **TIMER-0: timer di sistema**

- GATE0 sempre attivo
- OUT0 genera IRQ0 interrupt con priorità massima

• **TIMER-1: timer per il rinfresco della memoria**

- GATE1 sempre attivo
- OUT1 genera impulsi con un periodo di 15ms per attivare il rinfresco della memoria le D-RAM devono essere rinfrescate

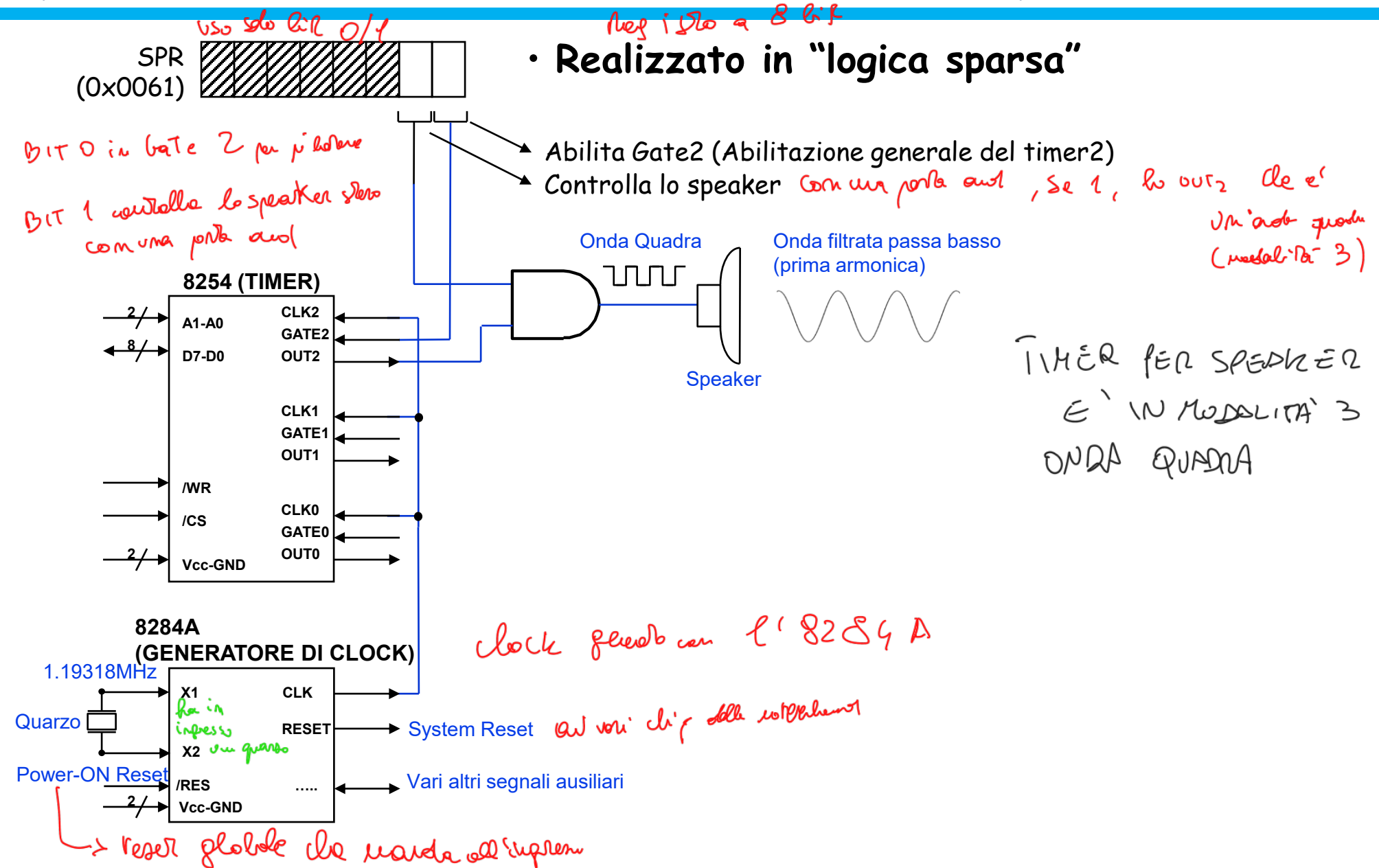
• **TIMER-2: generatore di beep (v. slide successiva)**

- GATE2 controllabile tramite bit0 di SPR, Speaker Register (porta di I/O 0x0061) può anche controllare frequenza del beep
- OUT2 in AND col bit1 di SPR, l'uscita va poi sul beeper

beep generato con una forma a d'onda

↓ SPR
controllore quanto e' acceso e spento

Speaker Register e generazione di suoni sullo speaker



Esempio di programmazione del Timer 8254 in C

- Generare una interruzione dopo $\Delta t = 50\text{ms}$ *evento dopo 50ms qui/dopo 50 ms voglio fare qualcosa*

- $F_c = 1.19318 \text{ MHz}$ ($T_c = 1 / 1.19318 \cdot 10^6 \approx 838\text{ns}$)

- $N = F_c \cdot \Delta t = 1.19318 \cdot 10^6 \cdot 50 \cdot 10^{-3} \rightarrow N \approx 59659$ *numero cicli*

$$\frac{\Delta t}{T_c} = N$$

$$N \cdot T_c = \Delta t$$

Passare N cicli in 50ms

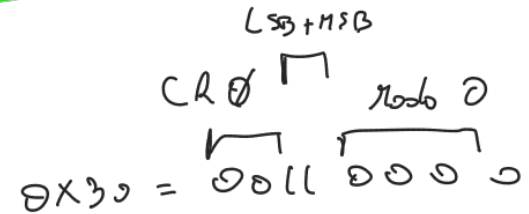
→ devo scrivere 59659 in Hex

int init_8254 (void)

devo programmare CWR e scrivere N

$$N \cdot T_c = \Delta t$$

$$N = \frac{\Delta t}{T_c} = \Delta t \cdot F_c$$



int cr_l, cr_h;

Store byte del valore 0x30 all'indirizzo 43

programma

outp(0x0043, 0x30); /* contatore 0, costante di tempo a 16 bit,
modo 0, conteggio binario */

→ V. SLIDE 7

cr_l = 59659 & 0x000000FF; *LSB*

cr_h = (59659 >> 8) & 0x000000FF; *preleva MSB* *sposta bit più significativi nei bit meno significativi*

outp(0x0040, cr_l); *e lo scrivo nel reg. 40 con 2 store word*

outp(0x0040, cr_h);

}