di GIORGIO OBER (GIOBE2000)

l'angolo di Mr A.KEER

(parte decima)

PROGETTARE con le PORTE LOGICHE

Decoder "a 7 segmenti"

La possibilità di disporre di visualizzatori formati da digit "a 7 segmenti" suggerisce l'analisi dettagliata dei principali dispositivi combinatori in grado di controllarli: scopriamone insieme i segreti e il modo migliore per utilizzarli.

Dopo aver analizzato ogni particolare dei visualizzatori (display) ad una o più cifre è arrivato il momento di scegliere i componenti in grado di assicurare loro le condizioni migliori per *tradurre* l'informazione binaria prodotta da un determinato dispositivo logico in *simboli significativi*.

Poichè si tratta di interpretare un *codice binario* (di solito *a 4 bit*) avremo di certo a che fare con dei **decoder** (decodificatori), più sofisticati di quelli **binari** e **decimali** descritti nelle precedenti puntate ma decisamente appartenenti alla stessa famiglia; ciascuno di essi sarà caratterizzato dalla presenza di 4 linee d'ingresso e 7 linee d'uscita, rese adatte a controllare lo stato (acceso o spento) di altrettanti led.

Il numero binario proposto sulle sue 4 linee d'ingresso produrrà effetto significativo (uno dei simboli del *sistema di numerazione decimale*, da 0 a 9) solo per le prime 10 combinazioni [quelle appartenenti al ***Codice BCD***, da $(0000)_2$ a $(1001)_2$] mentre per le restanti 6 [da $(1010)_2$ a $(1111)_2$] saranno mostrate combinazioni casuali di segmenti.

Fa eccezione a questa regola il prezioso componente **9368** che, per questa sua esclusiva proprietà sarà trattato per ultimo, nella prossima puntata..

Decoder da BCD a 7 segmenti (TTL): 7446, 74LS47, 74LS48, 7449

La serie **TTL LS** prevede numerosi componenti in grado garantire questo servizio, definiti **BCD to 7-segment Decoder** dai *datasheet*: i **74LS47** e **74LS48** (più **7446** e **74LS49**, meno adatti per la gestione diretta di digit), tra loro *funzionalmente identici* e *pin-out compatibili*, ma governati da tecnologie d'uscita diverse.

In aggiunta sono disponibili i componenti TTL **74LS247**, **74LS248** e **74LS249**, prodotti per porre rimedio ad una particolare proposta estetica: le cifre 6 $(0110)_2$ e 9 $(1001)_2$ sono mostrate da tutti i decoder del primo elenco in modo inconsueto, senza accendere rispettivamente il trattino in alto (segmento **a**) o in basso (segmento **d**); i 3 nuovi dispositivi, per il resto del tutto identici a quelli con le stesse ultime 2 cifre, mostrano invece il 6 e il 9 nella forma completa.

Il Decoder TTL **74LS47** converte il ***Codice BCD*** proposto sulle sue linee d'ingresso generando sulle linee d'uscita dei segnali *attivi bassi*: i segmenti del digit ad esso collegato si accenderanno dunque con uno **0** logico per cui questo componente è adatto a digit ad *anodo comune*; il suo pin-out è illustrato in *Figura 1*.

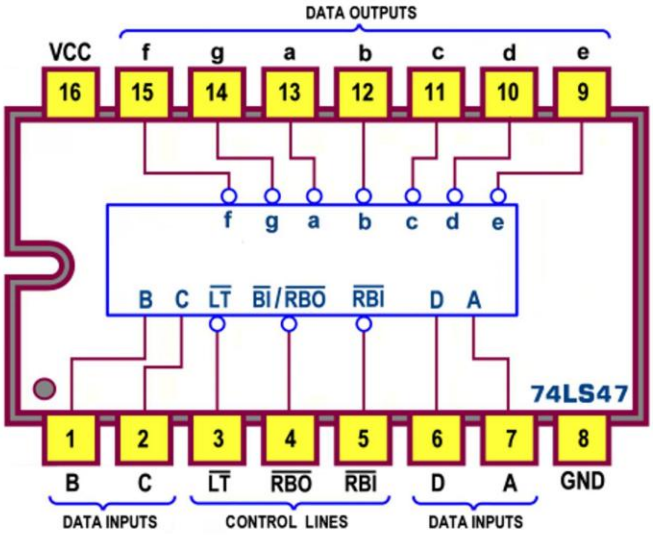

Figura 1 - BCD to 7-segment Decoder 74LS47: pinout

Ciascuna delle sue uscite (*Figura* 2) è dotata di un driver di tipo *open-collector*: lo stadio finale del componente è dunque un semplice transistor con emettitore collegato regolarmente a massa e collettore collegato direttamente all'uscita, senza nessun vincolo con la sua alimentazione interna.

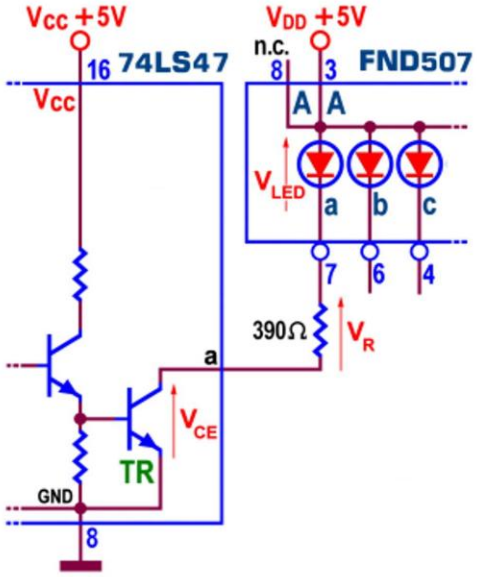

Figura 2 - BCD to 7-segment Decoder 74LS47: con FND507

Il fatto che il collettore del transistor TR sia aperto (cioè non collegato a nulla, *fluttuante*) ci obbliga a collegare un carico esterno in grado di chiudere il circuito d'uscita verso una tensione positiva, al fine di consentire il funzionamento del transistor stesso; di solito si utilizza un

resistore, detto di *pull-up* per significare (nel pittoresco modo di esprimersi degli anglosassoni) che il potenziale dell'uscita va "tirato su" verso l'alimentazione esterna.

Il vantaggio di disporre di uscite *open-collector* sta nella possibilità di assorbire correnti piuttosto elevate e di alimentare il carico con tensioni diverse (di norma maggiori) dalla tipica alimentazione a +**5V**; questo consente di pilotare *direttamente* dispositivi esigenti (per esempio un *relè*), cosa assolutamente preclusa alle classiche uscite *Totem-pole*.

Sulla base di queste premesse: a *livello basso* (**0** logico) è garantita una corrente assorbita di **24 mA**, attraverso il carico e in virtù del fatto che il transistor TR è ora in condizioni di forte conduzione, la sua tensione d'uscita è comunque quasi nulla (coincide con la $\mathbf{V_{CE}}$ di saturazione, di norma da **0,1V** a **0,3V**).

A livello alto (**1** logico) la sua tensione d'uscita può raggiungere il valore massimo ammesso per l'alimentazione esterna, $\mathbf{V_{DD}}$, fino a **15V** per questo componente; in fatti, in queste condizioni, il transistor TR non conduce (è *in interdizione*) per cui la caduta di tensione sul carico è nulla (non essendo possibile alcun passaggio di corrente attraverso di esso) e quindi tutta la tensione di alimentazione si ritrova ai suoi capi.

Lo *schema* pratico (*Figura 3*), da utilizzare direttamente nel contesto di un progetto, riassume le caratteristiche logiche di questo componente, evidenziando il nome dei *segnali logici* che entrano e di quelli che escono, e il *numero* dei piedini ad essi riservati.

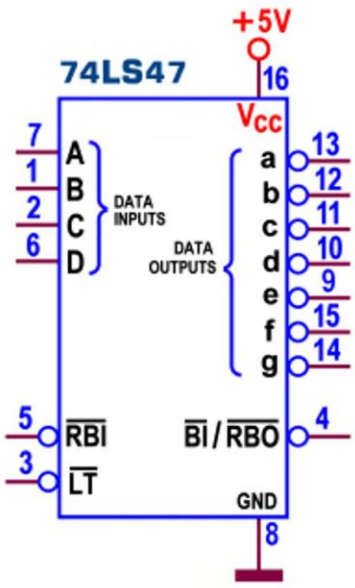

Figura 3 - BCD to 7-segment Decoder 74LS47: schema pratico

La *Figura 4* mostra i collegamenti necessari per pilotare un **FND507**; i tre piedini di controllo, **LT** (pin3), **RBO** (pin4) e **RBI** (pin5), non sono stati coinvolti, lasciandoli scollegati (o meglio collegando i due ingressi a +**5V**): per capirne il funzionamento è necessario attendere la descrizione del contesto in cui saranno operativi, per cui saranno esaminati con dettaglio solo al termine dello studio dei singoli componenti.

I 7 resistori presenti nel circuito hanno lo scopo di limitare la corrente che, dall'alimentatore esterno attraverso l'integrato, percorre ciascuno dei segmenti-led del digit, quando le uscite sono attive (*basse*), al fine di garantirne l'incolumità; il valore della loro resistenza è calcolato per una corrente di **8 mA** in presenza di tensioni d'alimentazione $\mathbf{V_{DD}}$ di +**5V**, +**12V** e +**15V**, e normalizzato ad un vicino valore standard, rispettivamente pari a **390**, **1,2k** e **1,8k** ohm.

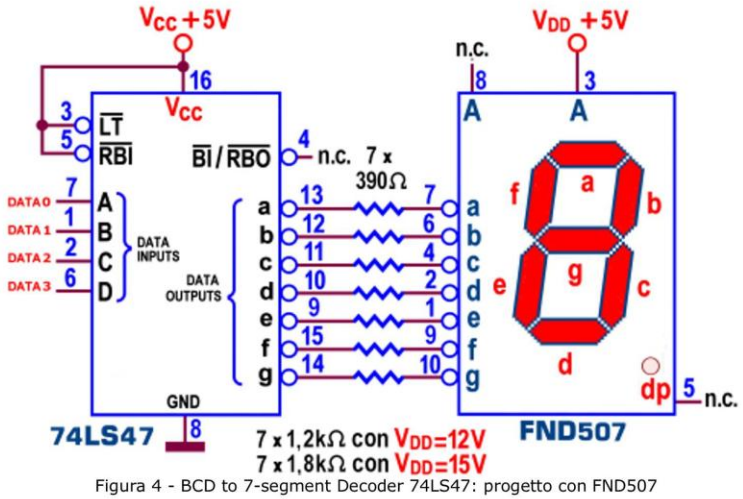

Figura 4 - BCD to 7-segment Decoder 74LS47: progetto con FND507

La potenza dissipata dal **74LS47** è di **35 mW**.

Esistono altre 2 versioni, il **74LS347** e il **74LS447**, del tutto identiche rispettivamente ai **74LS47** e **74LS247**, con la sola eccezione di sopportare, a livello alto $\mathbf{V_{OH}}$ (**1** logico), al massimo **7V** invece di **15V**.

Il Decoder TTL **7446** è una versione *pinout compatibile* ma più potente del **74LS47**, capace di assorbire corrente fino a **40 mA**, e di sopportare tensioni di alimentazione esterna fino a **+30V**, con potenza dissipata di **320 mW** (per altro uguale a quella del **7447**, esso pure non di tipo LS); fatte salve queste differenze, per esso valgono tutte le considerazioni precedenti.

Il Decoder TTL **74LS48** converte il ***Codice BCD*** proposto sulle sue linee d'ingresso generando sulle linee d'uscita dei segnali *attivi alti*: i segmenti del digit ad esso collegato si accenderanno dunque con un **1** logico per cui questo componente è adatto a digit a *catodo comune*; il suo pin-out è illustrato in *Figura 5*.

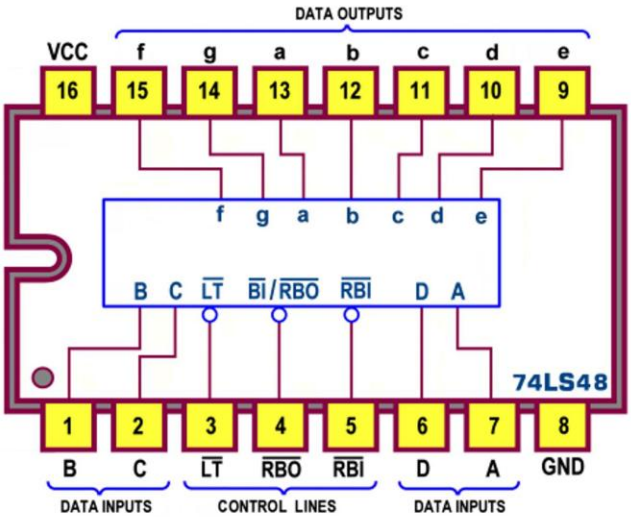

Figura 5 - BCD to 7-segment Decoder 74LS48: pinout

Ciascuna delle sue uscite è caratterizzata, rispetto alla normale *Totem-pole*, dalla presenza di un resistore interno di *pull-up*, il cui valore di resistenza è pari a **2k** (*Figura 6*); lo scopo di

questa originale scelta funzionale è quello di assicurare maggiore corrente erogata, in sintonia con le precedenti osservazioni sul fatto che le uscite TTL non sono (nella norma) adatte a questo servizio.

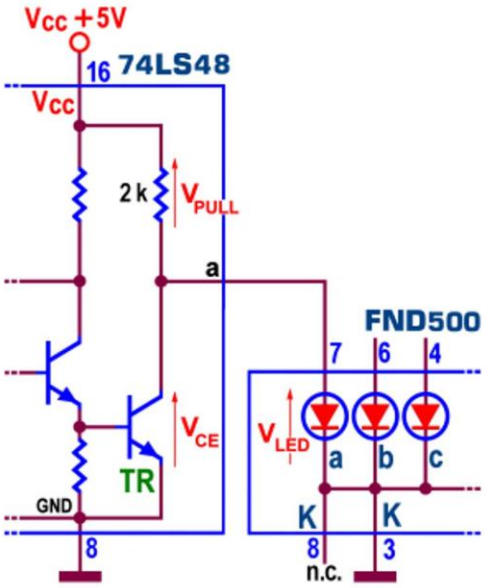

Figura 6 - BCD to 7-segment Decoder 74LS48: con FND500

Tuttavia, pur in presenza di un pull-up interno di **2k**, il Decoder **74LS48** non è in grado di pilotare direttamente un normale digit: questo componente è stato progettato per "*vedere*" una logica TTL o DTL, o per pilotare un amplificatore di corrente (driver) ad esso successivo, per esempio un transistor o un sofisticato **ULN2003** (contenente 7 coppie di transistor NPN in configurazione Darlington) per il controllo di lampade, piuttosto che per quello di digit.

Secondo il *datasheet* a livello alto (**1** logico) può erogare al massimo **1,3 mA** (i conti tornano, volendo assicurare in uscita il valore standard di $\mathbf{V_{OH}}$**=2,4V**: **5V-2Kohm*1,3mA=5V-2,6V=2,4V**); caricandolo con led, se il transistor TR non conduce (*interdetto*), la caduta di tensione sul pull-up interno di **2k**, pari a **5V-1,7V= 3,3V**, impone il passaggio di **1,65 mA**, una corrente decisamente insufficiente per accendere con accettabile luminosità i segmenti-led di un normale digit a *catodo comune*.

A livello basso (**0** logico) è garantita (sempre secondo il *datasheet*) una corrente assorbita di **6 mA**, attraverso il transistor TR, ora *in saturazione*: d'istinto sembrerebbe più logico l'impiego con digit ad *anodo comune* (naturalmente corredato da adeguati resistori in serie a ciascun segmento per limitare la corrente al valore di **6 mA**, di resistenza **470 ohm**, con alimentazione esterna di +**5V**), ma va tenuto presente che le sequenze d'uscita sono progettate per dare effetto in *logica positiva*, per cui (essendo attivo, in questo caso, lo **0** logico) l'effetto sul digit sarebbe decisamente inaccettabile (vedi *Figura 7*).

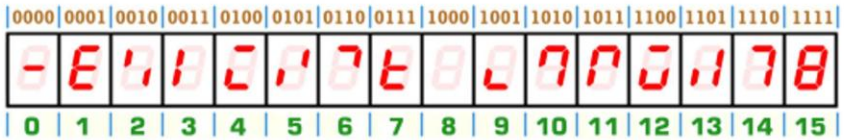

Figura 7 - BCD to 7-segment Decoder 74LS48: simboli visualizzati con FND507

Lo *schema* pratico (*Figura 8*) riassume le caratteristiche logiche di questo componente.

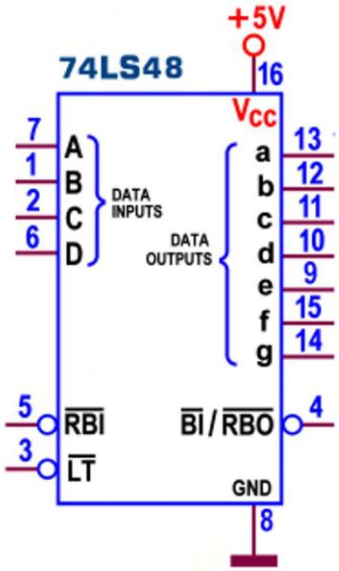

Figura 8 - BCD to 7-segment Decoder 74LS48: schema pratico

L'impiego del Decoder **74LS48** con un normale digit a *catodo comune* non è dunque particolarmente consigliabile ed è comunque possibile (*Figura 9*) solo con l'impiego di resistori addizionali esterni di pull-up su ciascuna delle 7 linee d'uscita; la scelta del valore di resistenza è legata all'intensità luminosa desiderata per i segmenti-led; va da se che non è il caso di scendere a valori troppo piccoli per non rischiare di bruciare le uscite del componente.

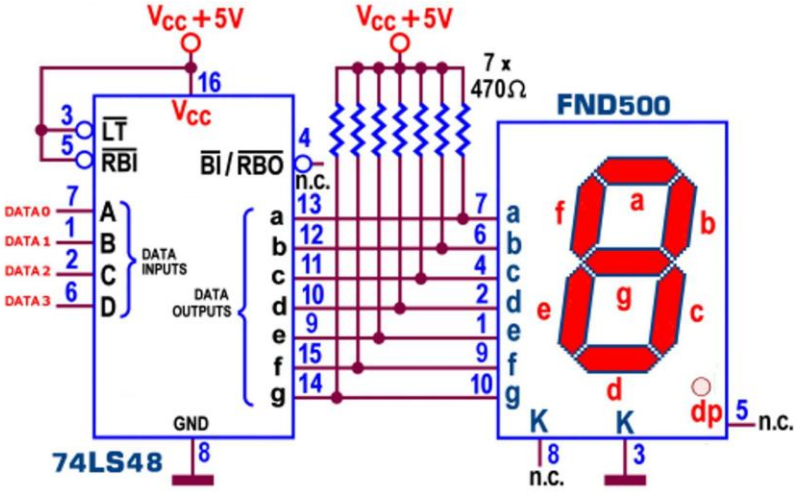

Figura 9 - BCD to 7-segment Decoder 74LS48: progetto con FND500

Anche in questo caso i tre piedini di controllo, **LT** (pin3), **RBO** (pin4) e **RBI** (pin5), non sono stati coinvolti, lasciandoli scollegati (o meglio collegando i due ingressi a +**5V**). La potenza dissipata dal **74LS48** è di **125 mW**.

Il Decoder TTL **74LS49** è una versione più compatta e complementare del **74LS47**, essendo ora *attive alte* le sue linee d'uscita; il ***Codice BCD*** proposto in ingresso genera dunque delle sequenze adatte ad accendere con uno **1** logico i segmenti del digit ad esso collegato, per questo di tipo a *catodo comune*; il suo pin-out è illustrato in *Figura 10*.

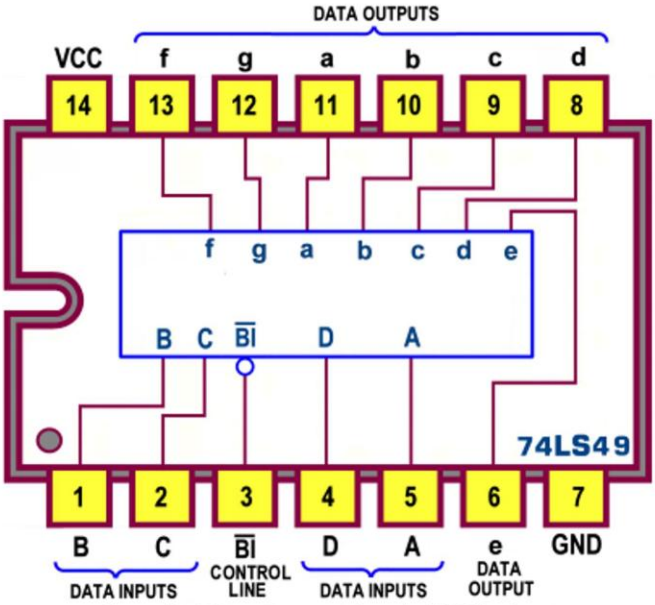

Figura 10 - BCD to 7-segment Decoder 74LS49: pinout

Le sue uscite sono di tipo *open-collector*, cioè il suo stadio finale è caratterizzato dalla presenza di un transistor con emettitore collegato a massa e con collettore lasciato aperto in uscita (cioè non collegato a nulla); anche questo Decoder non è adatto alla gestione diretta di un normale digit, essendo stato progettato per pilotare una logica TTL o per essere anteposto ad un driver per il controllo di lampade.

Secondo il *datasheet* a livello alto (**1** logico) può erogare al massimo **0,250 mA**, una corrente decisamente insufficiente per i segmenti-led di un normale digit a *catodo comune*; a livello basso (**0** logico) è garantita una corrente assorbita di **8 mA**, ma il dato sembra irrilevante nei confronti del servizio atteso da questo componente.

Nel normale funzionamento, l'unico piedino di controllo, **BI** (pin3), può essere lasciando scollegato o collegandolo a +5V; la potenza dissipata dal **74LS49** è di **40 mW**.

Decoder per Digit a 7 segmenti: Segnali di controllo

Tutti i Decoder analizzati finora utilizzano 3 linee di controllo (**LT**, **BI/RBO** e **RBI**, tutte *attive basse*) per assicurare servizi piuttosto particolari, molto utili nell'ambito dei *visualizzatori a più digit*, tipici dei comuni dispositivi presenti sul mercato.

Le linee collegate al *pin3* e al *pin5* sono decisamente *d'ingresso* mentre quella collegata al *pin4* dispone di una circuiteria interna (basata su un resistore di *pull-up*, adatta a realizzare una logica di tipo *wired-AND* con ogni uscita *open collector*) che permette sia di disporre su essa di un segnale *d'uscita* **RBO**, sia di forzare su essa un segnale *d'ingresso* **BI** (*attivo basso*) senza produrre danni.

La linea collegata al *pin4*, utilizzata come ingresso (**BI**, *Blanking Input*) permette (se attivata, cioè forzata a **0**) di spegnere incondizionatamente il digit collegato al Decoder, con qualunque codice presente sugli ingressi DCBA; nel funzionamento normale l'ingresso **BI** deve essere lasciato scollegato (o collegato al **+5V**), cioè disattivo (forzato a **1**); da notare che l'ingresso **BI** è prioritario rispetto a tutti gli altri, cioè quando esso è attivo contemporaneamente ad ogni altro ingresso la funzione degli altri viene ignorata.

La linea d'ingresso collegata al *pin3* (**LT**, *Lamp Test*), opera il servizio opposto: forzata a **0** accende (*attiva*) tutte le linee d'uscita di segmento, forzando l'accensione di tutti i segmenti del digit collegato al Decoder; il suo compito è dunque quello di fornire uno strumento adatto a verificarne l'integrità.

La sua azione ha un livello di priorità secondo solo al segnale d'ingresso **BI** per cui potrà essere esercitata solo se **BI** non è contemporaneamente attivo.

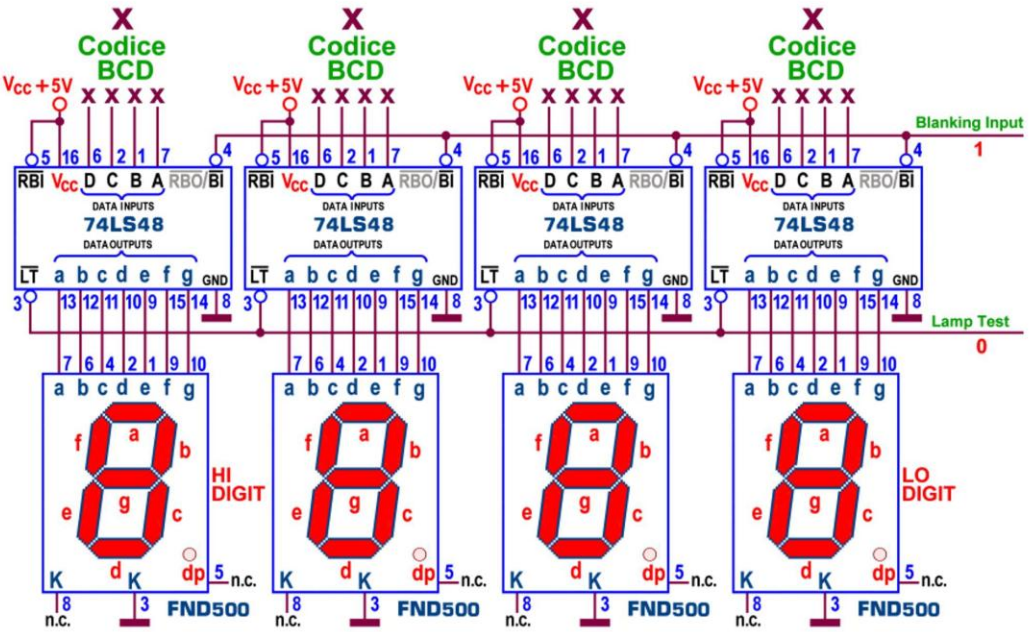

Figura 11 - Visualizzatore a 4 digit FND500 con 74LS48: Effetto del Lamp Test

La *Figura 11* mostra un visualizzatore costituito da 4 digit **FND500** (governati da altrettanti **74LS48**) con tutti i segmenti dei digit accesi per effetto del controllo esercitato sugli ingressi **LT** (tutti attivi, forzati a **0**) con ingressi **BI** disabilitati (forzati a **1**); in essa (e nelle successive) non figurano i consigliati 7 resistori esterni di pull-up, per non appesantire lo schema..

La tecnica di ***Blanking*** (*spegnimento*) può essere usata anche per controllare l'intensità luminosa del visualizzatore; è sufficiente applicare sul *pin4* un'onda quadra non simmetrica e modificare il suo *duty cycle*; come è noto questo parametro esprime (anche in percentuale) il rapporto tra la durata della *parte alta* e il *periodo* del segnale.

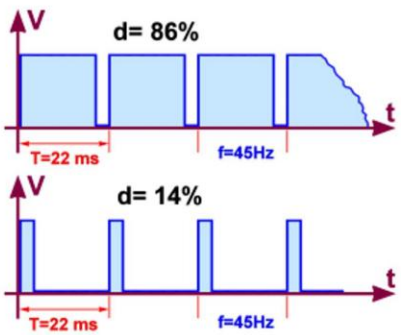

Figura 12 - Visualizzatore a 4 digit FND500 con 74LS48: Effetto Blanking

Se l'aspetto dell'onda quadra è quello di un treno d'impulsi di breve durata, molto distanziati tra loro (tipico di un duty cycle molto basso, per esempio del 5%) l'effetto sui digit è quello di tenerli (quasi) spenti, mentre se l'onda quadra assomiglia ad un treno d'impulsi molto lunghi, praticamente attaccati l'uno all'altro (tipico di un duty cycle molto alto, per esempio del 95%) l'effetto sui digit è quello di accenderli (quasi) al massimo.

Per garantire questo evento è per altro necessario che la frequenza del segnale così generato sia sufficientemente elevata, maggiore di **45** Hz per i digit con Led all'Arseniuro Fosfuro di Gallio (GaAsP, quindi *rossi*): solo così la veloce sequenza di spegnimenti (imposti da **BI** attivo, cioè a **0**) e accensioni (imposti da **BI** disattivo, cioè a **1**) può essere percepita come variazione di luminosità per effetto della persistenza ottica dell'immagine sulla retina dall'occhio umano.

La *Figura 12* mostra l'aspetto dei segnali necessari rispettivamente per una discreta (duty cycle pari a 86%) e bassa (duty cycle del 14%) accensione dei digit: naturalmente l'effetto ottenuto è legato al valore medio del segnale stesso, più vicino a **1** logico nel primo caso e allo **0** nel secondo.

Tra le azioni di controllo esercitate dai Decoder sui rispettivi digit rimane da descrivere quella, decisamente sofisticata, della coppia **RBI**, **RBO**.

La linea d'ingresso collegata al *pin5* (**RBI**, *Ripple Blanking Input*) consente (se attivata, cioè forzata a **0**) di spegnere (*blanking*) il digit collegato al Decoder *solo se* contemporaneamente il codice presente sugli ingressi DCBA è $(0000)_2$; in queste condizioni forza a **0** anche l'uscita sul *pin4,* **RBO** (*Ripple Blanking Output*); se il codice presente sugli ingressi DCBA è diverso da $(0000)_2$ il digit mostra regolarmente i simboli corrispondenti e l'uscita **RBO** rimane disattiva (cioè a livello logico **1**).

La condizione per poter visualizzare comunque la cifra zero è di lasciare l'ingresso **RBI** scollegato (o meglio collegato al +5V), cioè disattivo (forzato a livello logico **1**).

L'uscita **RBO** è un utile meccanismo che consente di evitare di accendere tutti gli zero *a sinistra* di un numero, decisamente antiestetici e inutili; anche nella comune aritmetica sembra logico scrivere 236 piuttosto che 0236; il metodo per raggiungere questo scopo è quello di collegare a massa l'ingresso **RBI** del Decoder collegato al digit più significativo e di collegare in cascata l'uscita **RBO** di ciascun digit (a cominciare da quello più significativo) con l'ingresso **RBI** del digit successivo (subito a destra).

La situazione è mostrata in *Figura 13*, ancora per un visualizzatore con 4 digit **FND500**, ciascuno controllato da un **74LS48**, nell'ipotesi che i ***codici BCD*** in ingresso ai 4 Decoder siano quelli corrispondenti alle cifre 0236; con questa predisposizione (ingresso **RBI** del Decoder collegato al *digit più significativo* forzato a massa) lo "**0**" corrispondente al codice $(0000)_2$, fornito sugli ingressi DCBA, non viene mostrato (*Blanking* della cifra, il digit rimane spento).

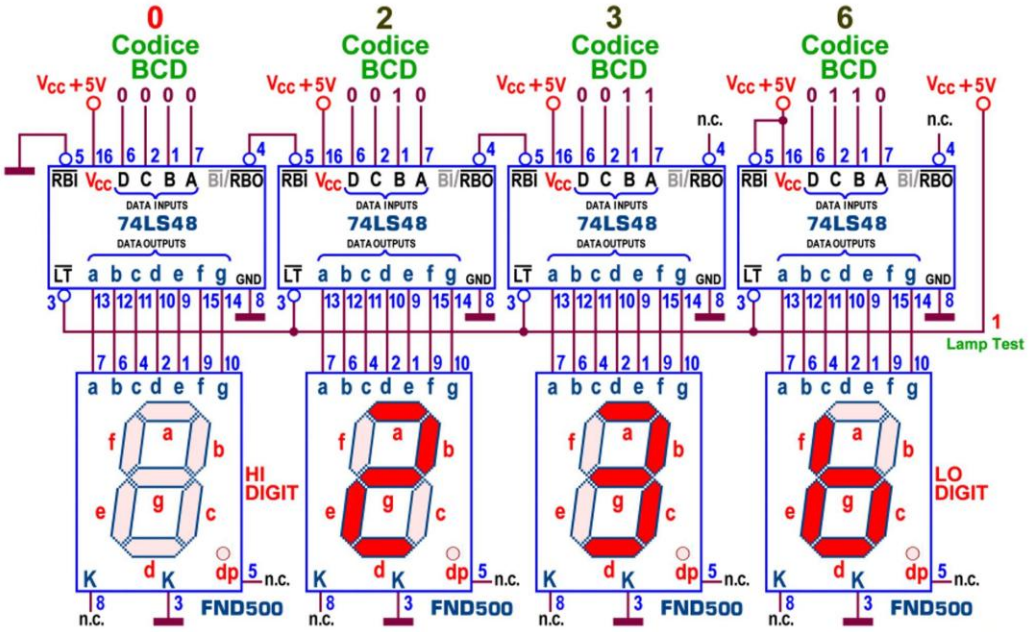

Figura 13 - Visualizzatore a 4 digit FND500 con 74LS48: Effetto Ripple Blanking

Questo evento (e solo in questo caso!) forza a **0** anche l'uscita **RBO** che, a sua volta, forza a massa l'ingresso **RBI** del Decoder collegato al digit successivo (subito a destra): poiché il codice presente sugli ingressi DCBA di questo Decoder **non è** $(0000)_2$ il fatto che il suo ingresso **RBI** sia a livello logico **0** è ora del tutto irrilevante e, da questa posizione in poi ogni cifra verrà visualizzata, anche gli eventuali "**0**" interni.

In caso contrario anche il secondo digit rimane spento e, in coerenza con la regola, il Decoder ad esso associato forza a **0** la sua uscita **RBO** e, di conseguenza, anche l'ingresso **RBI** del Decoder collegato al digit successivo (il terzo da sinistra); il meccanismo si applica per tutte le cifre del display e, se mantenuto inalterato, fornendo 4 codici BCD "nulli", terrebbe spento tutto il visualizzatore.

Sebbene la cosa sia possibile non sembra comunque una buona soluzione: se sugli ingressi DCBA di ogni decoder viene fornito il codice $(0000)_2$ è ragionevole mostrare almeno lo zero associato al digit più a destra; per questa ragione l'ingresso **RBI** del Decoder ad esso relativo viene mantenuto a **1**, spezzando la catena di controllo e vanificandone l'effetto.

Una curiosità: osservando la *Figura 14* si capisce la ragione per la quale gli ingressi (**RBI**, *Input*) e le uscite (**RBO**, *Output*) di *Blanking* vengano caratterizzate dall'attributo *Ripple*; il temine può essere tradotto con *ondulazione*, ed è proprio il senso che danno i quattro collegamenti tra **RBI** e **RBO**; tra l'altro questo termine è molto comune anche in un altro ambito: *Ripple* è l'ondulazione residua rilevabile in uscita ad un semplice alimentatore non stabilizzato (trasformatore + ponte di greatz + condensatore elettrolitico) sottoposto ad un piccolo assorbimento di corrente.

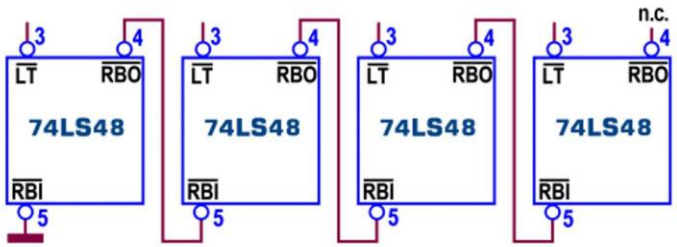

Figura 14 - Visualizzatore a 4 digit FND500 con 74LS48: Ripple Blanking

La tecnica di *Ripple Blanking* può essere applicata, all'occorrenza, anche per evitare di accendere tutti gli zero a destra di un numero: la cosa ha senso solo se ci si riferisce alla sua parte frazionaria; per esempio sembra logico scrivere 3,2 piuttosto che 03,20:

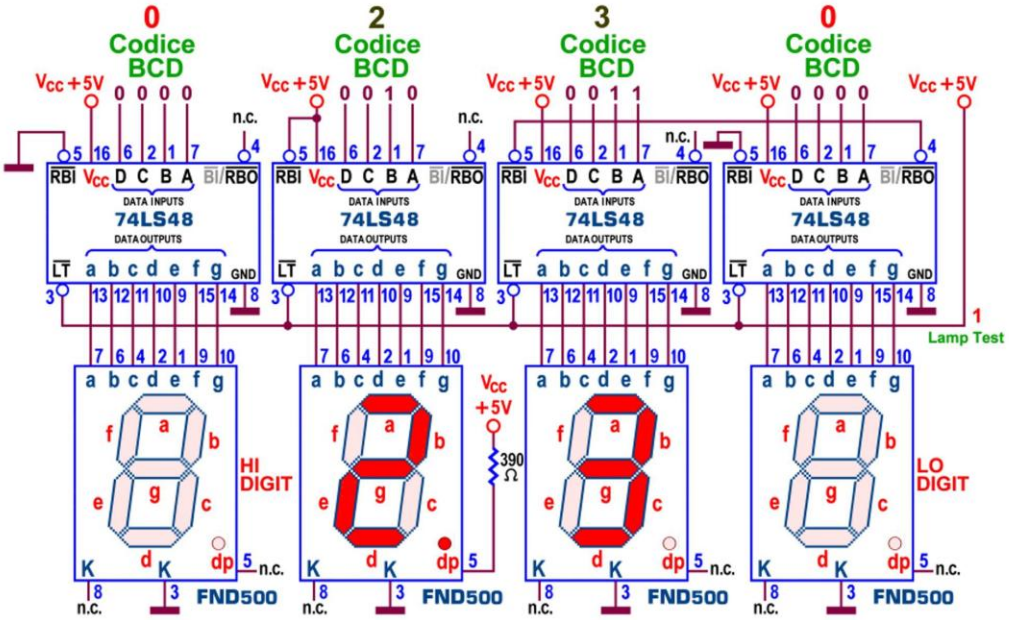

Figura 15 - Visualizzatore a 4 digit FND500 con 74LS48: Effetto Blanking decimali

Ripetiamo l'analisi organizzando il visualizzatore precedente per gestire la parte intera di un numero sui primi 2 digit e la sua parte frazionaria sugli ultimi 2 (*Figura 15*); la situazione è evidenziata dalla presenza del punto decimale acceso permanentemente sul secondo digit.

L'ipotesi di lavoro è che i codici BCD in ingresso ai 4 Decoder siano ora quelli corrispondenti alle cifre 0230; la gestione della parte intera è identica a quella già descritta in precedenza: il Decoder associato al digit più significativo ha l'ingresso **RBI** a massa perciò se (come nel nostro esempio) sui suoi ingressi DCBA è presente il codice $(0000)_2$, la visualizzazione dello "**0**" viene evitata (*Blanking* della cifra) e la sua uscita **RBO** viene forzata a massa.

Se essa fosse collegata all'ingresso **RBI** del Decoder associato al digit successivo (subito a destra, quello con punto decimale acceso) l'eventuale presenza sui suoi ingressi DCBA del codice $(0000)_2$ eviterebbe anche per esso la visualizzazione dello "**0**"; ma l'input **RBI** di questo Decoder non è soggetto a nessun controllo (non è attivato, essendo forzato a **1** logico) per cui consentirà la visualizzazione di qualunque cifra decimale, compreso lo "**0**".

La gestione della parte frazionaria riflette la stessa logica ma, dovendo evitare l'accensione degli zero *a destra* del numero, il controllo dovrà essere esercitato a partire dal digit meno significativo; fissando a massa l'ingresso **RBI** del Decoder collegato al digit meno significativo se (come nel nostro esempio) sui suoi ingressi DCBA è presente il codice $(0000)_2$, la visualizzazione dello "**0**" su di esso viene evitata; questa situazione forza a **0** anche la sua uscita **RBO** che, a sua volta, forza a massa l'ingresso **RBI** del Decoder collegato al digit precedente (subito a sinistra) evitando la visualizzazione dello "**0**" se sui suoi ingressi DCBA è presente il codice $(0000)_2$.

Naturalmente se la cosa sembra eccessiva è sufficiente evitare di fare quest'ultimo collegamento, fissando a 1 logico l'ingresso **RBI** del terzo Decoder, garantendo la visibilità di ogni valore, da 0,0 a 9,9.

	INGRESSI							USCITE							
n°	LT	RBI	D	C	B	A	BI	RBO	a	b	c	d	e	f	g
0	H	L	L	L	L	L		L	L	L	L	L	L	L	L
		H						H	H	H	H	H	H	H	L
1	H	X	L	L	L	H		H	L	H	H	L	L	L	L
2	H	X	L	L	H	L		H	H	H	L	H	H	L	H
3	H	X	L	L	H	H		H	H	H	H	H	L	L	H
4	H	X	L	H	L	L		H	L	H	H	L	L	H	H
5	H	X	L	H	L	H		H	H	L	H	H	L	H	H
6	H	X	L	H	H	L		H	L	L	H	H	H	H	H
7	H	X	L	H	H	H		H	H	H	H	L	L	L	L
8	H	X	H	L	L	L		H	H	H	H	H	H	H	H
9	H	X	H	L	L	H		H	H	H	H	L	L	H	H
10	H	X	H	L	H	L		H	L	L	L	H	H	L	H
11	H	X	H	L	H	H		H	L	L	H	H	L	L	H
12	H	X	H	H	L	L		H	L	H	L	L	L	H	H
13	H	X	H	H	L	H		H	H	L	L	H	L	H	H
14	H	X	H	H	H	L		H	L	L	L	H	H	H	H
15	H	X	H	H	H	H		H	L	L	L	L	L	L	L
BI	X	X	X	X	X	X	L		L	L	L	L	L	L	L
LT	L	X	X	X	X	X	H		H	H	H	H	H	H	H

H livello logico alto
L livello logico basso
X livello logico irrilevante

Figura 16 - BCD to 7-segment Decoder 74LS48: Tabella di verità

La *Tabella di verità* (*Truth Table* o *Function Table*) è sempre disponibile nei *data sheet* anche per i decoder come quelli appena descritti; la *Figura 16* si riferisce al **74LS48** ed è sostanzialmente la stessa di quella relativa ai **74LS47** (e **7446**) fatta salva la differenza che il primo ha uscite (pull-up 2k) *attive alte* mentre i secondi hanno uscite (open-collector) *attive basse*.

L'analisi della tabella sintetizza ogni precedente considerazione: a) la penultima riga evidenzia la forza prioritaria di **BI**: quando esso è *attivo* (*basso*, cioè a **L**ow level) tutte le uscite di segmento passano a **0** indipendentemente dal valore di ogni altro ingresso, spegnendo di fatto il digit (a catodo comune) collegato al decoder; b) l'ultima riga conferma che dopo **BI** comanda **LT**: se questo ingresso è *attivo* (*basso*, **L**), con **BI** disattivo, tutte le uscite di segmento passano a **1** (*alte*, **H**), indipendentemente dal valore degli altri ingressi, accendendo di fatto il digit; c) la prima riga mostra l'effetto di **RBI**: se esso è *attivo* (*basso*, **L**), con **BI** e **LT** disattivi e tutti gli ingressi di dato DCBA a **0**, tutte le uscite di segmento e l'uscita **RBO** passano a **0**

(digit spento); d) in tutti gli altri casi, con **LT** e **RBI** disattivi (o meglio forzati a **1**), le linee di segmento assumeranno i livelli adatti a mostrare le cifre decimali corrispondenti al codice ***BCD*** proposto sugli ingressi DCBA e la linea **BI/RBO** (ora intesa come uscita) sarà tenuta a **1**.

In chiusura voglio presentare un ultimo sofisticato componente della serie **TTL standard**: si tratta del **74143**, un contatore sincrono BCD dotato di memoria interna in grado di mostrare direttamente la cifra decimale del suo conteggio su un digit *ad anodo comune*, disponendo internamente anche di un decoder/driver "a 7 segmenti" (*Figura 17*).

Il fatto di prenderlo in considerazione ci permette di sottolineare un'esigenza finora lasciata latente: quella di disporre di un dispositivo in grado di fornire al decoder i ***codici BCD*** da tradurre in *numeri* sul visualizzatore; qualunque logica è in grado di farlo, dalla più semplice (4 switch) alla più sofisticata (le uscite di un processore), ma una delle più gratificanti è quella affidata alle uscite di un contatore che, automaticamente, obbligheranno il digit (collegato al decoder) a mostrare *in sequenza* tutte e 10 le cifre decimali.

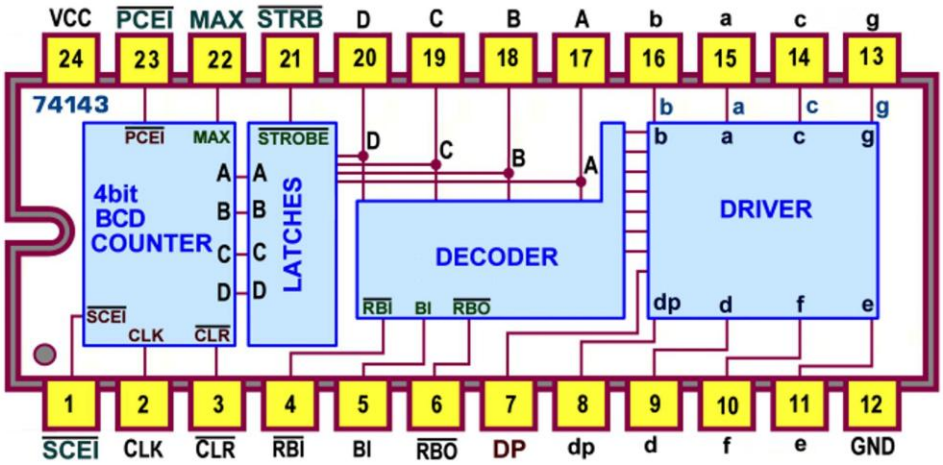

Figura 17 - 4bit Counter/Latch BCD to 7-segment Decoder/Driver 74L143: pinout

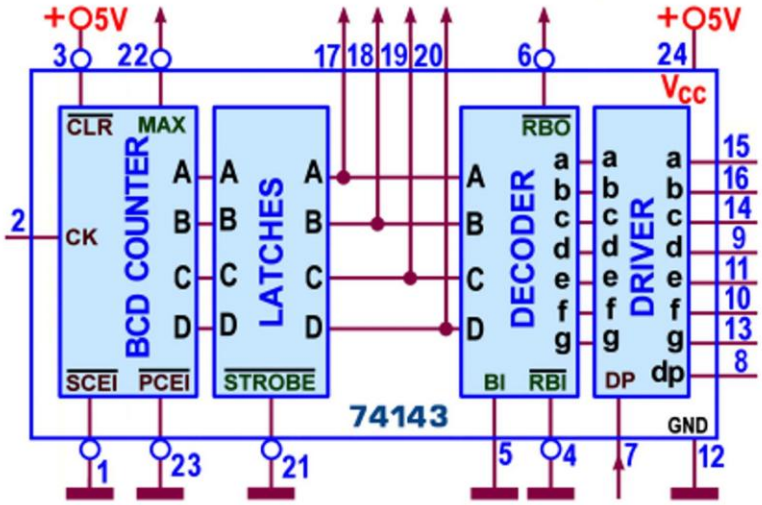

Figura 18 - 4bit Counter/Latch BCD to 7-segment Decoder/Driver 74L143: schema funzionale

La funzione "contatore" del **74143** sarà argomento di una successiva, remota, puntata; in questa sede possiamo notare come la funzione di decoder sia esattamente la stessa descritta finora; la *Figura 18* ci aiuta a capirne il funzionamento.

La linea **BI** (*Blanking Input*, sul *pin5*, ora *attiva alta*) se forzata a **1** spegne incondizionatamente il digit collegato al dispositivo e forza a **0** l'uscita **RBO**; nel funzionamento normale l'ingresso **BI** deve essere a massa (**0** logico) e può essere sottoposto a impulsi di durata variabile per controllare l'intensità luminosa del visualizzatore.

La coppia di segnali **RBI** (*pin4*, *Ripple Blanking Input*) e **RBO** (*pin6*, *Ripple Blanking Output*) supporta il medesimo meccanismo di Ripple Blanking già descritto, al fine di tenere spente le cifre a zero eventualmente presenti *a sinistra* di un numero; ricordiamo che per disporre di questo servizio l'ingresso **RBI** del digit più significativo deve essere forzato a massa e la sua uscita **RBO** (da collegarsi al **RBI** del digit successivo) sarà a sua volta forzata a **0** *solo se* il codice presente sugli ingressi DCBA è $(0000)_2$.

A margine della descrizione possiamo chiarire il compito dei segnali della *sezione contatore*: a) l'ingresso **CLR** (*pin3*, *Clear Input*, *attivo basso*) quando a **0** azzera le 4 uscite del contatore e lo tiene bloccato; per un normale conteggio deve essere portato a **1**; b) gli ingressi **PCEI** (*pin23*, *Parallel Count Enable Input*) e **SCEI** (*pin1*, *Serial Count Enable Input*) abilitano il conteggio ("*Counting mode*") quando (essendo *attivi bassi*) sono entrambi a **0**; c) l'ingresso **CK** (*pin2*, *Clock Input*) consente l'incremento il conteggio "in avanti" in corrispondenza di ogni transizione sul fronte di salita, purchè il dispositivo sia in "Counting mode", cioè con le abilitazioni entrambe a **0** e con Clear a **1**; d) l'uscita **MAX** (*pin22*, *Maximum Count Output*) rimane alta per tutto il tempo necessario a mostrare le cifre da "1" a "8", va a **0** quando sulle uscite DCBA si presenta il numero "9", e torna a **1** quando scatta lo "0", dopo un periodo di clock; in sostanza **MAX** è utilizzato (per tutto il tempo in cui rimane *basso*) per controllare il clock di un eventuale **74143** posto in cascata al primo, autorizzandolo a riconoscerne il fronte di salita, necessario per incrementarne il conteggio.

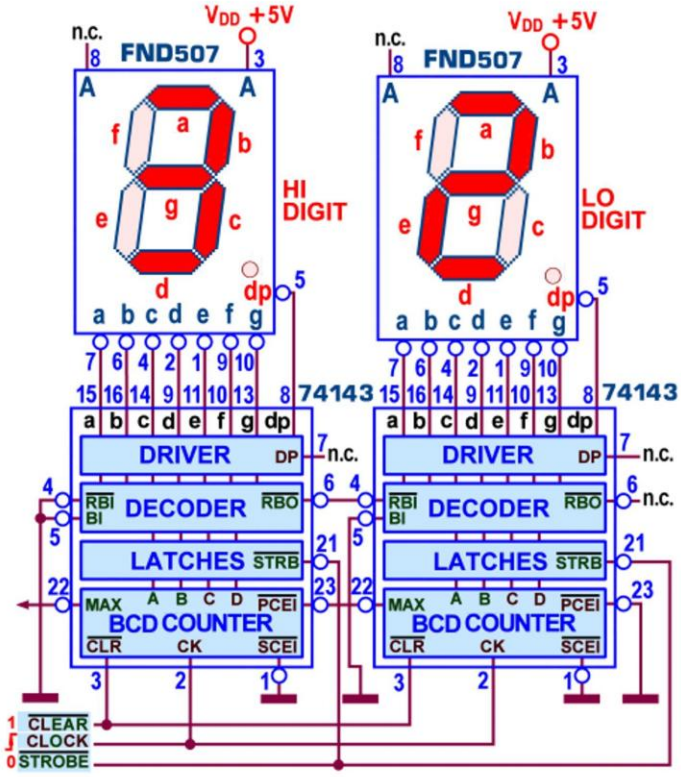

Figura 19 - 4bit Counter/Latch BCD to 7-segment Decoder/Driver 74L143: progetto con FND507

La *Figura 19* mostra lo schema di un display a 2 digit di tipo **FND507** sul quale può essere visualizzato un conteggio da 0 a 99; per questo progetto valgono queste considerazioni: a) il clock è fornito contemporaneamente ad entrambi i contatori, essendo essi di tipo sincrono; b) il secondo contatore incrementa il suo conteggio solo quando il primo ha completato il suo ciclo (per l'azione di gating esercitata da **MAX**); c) lo zero non significativo della cifra delle decine del numero non viene visualizzato; d) per un normale conteggio **CLR** va lasciato scollegato (o collegato al +5V) e **STRB** va tenuto a massa (memoria disabilitata); e) per una migliore leggibilità lo schema non porta le uscite dei 4 latches (*pin17*, *pin18*, *pin19* e *pin20*) e le due alimentazioni (**GND** *pin8* e **Vcc** *pin24*); f) i punti decimali sono gestibili separatamente e possono essere accesi fornendo un **1** sulla line*a Decimal Point Input* (*pin7*).

Dal punto di vista della gestione dei digit il **74143** appartiene alla categoria dei *decoder con memoria e con driver*, come il **4511** (unico disponibile nella serie **CMOS**) e il prezioso **9368** (unico in grado di mostrare tutti i 16 simboli del *sistema di numerazione esadecimale*), la cui trattazione sarà oggetto della prossima puntata.

La sua memoria interna è di tipo *RS Latch* e consente di *congelare* la cifra mostrata indipendentemente dal valore presente, nel tempo, in ingresso; la batteria di amplificatori di corrente (*driver*) è in grado di assicurare le migliori condizioni di pilotaggio per i led del digit, rendendo inutile l'impiego di resistori di limitazione in serie: le uscite da "a" a "g" sono progettate per assicurare una corrente assorbita di **15 mA** (**7 mA** per il "dp"), mantenuta costante anche per le possibili variazioni della tensione d'uscita.

La massima frequenza ammessa per di clock è tipicamente di **18 MHz** e l'assorbimento tipico è di **280 mW**.