di GIORGIO OBER (GIOBE2000)

l'angolo di Mr A.KEER

(parte quattordicesima)

PROGETTARE con le PORTE LOGICHE

Sommatori BCD

L'aritmetica BCD occupa un posto importante nel rapporto tra le macchine e noi. poveri umani: analizziamo i primi componenti destinati al supporto di questo tipo di calcolo numerico; scopriamone i segreti e il modo migliore per utilizzarli.

Sommatori BCD (TTL): 74F83

Nel corso della puntata precedente abbiamo avuto modo di affrontare le problematiche legate alla somma di numeri binari a più bit, sottolineando i problemi di velocità intrinseci dei sommatori di tipo ***Ripple Carry***, risolti con l'aggiunta di particolari reti combinatorie in grado di generare in anticipo il valore del *Riporto*, cioè basati sulla tecnica ***Carry Look Ahead***.

Completiamo ora la trattazione tenendo presente che, non di rado, i sistemi di calcolo devono gestire dati forniti (e da restituire) in modo ***decimale***, cioè affidati ai 10 *simboli* del *sistema di numerazione decimale*; poichè le macchine combinatorie chiamate a svolgere questo compito si "nutrono" esclusivamente sequenze binarie è necessario disporre di una strumento, detto **codice BCD** (***B****inary* ***C****oded* ***D****ecimal*), in grado di *codificare in binario* ciascuno dei 10 simboli.

Già in passato abbiamo avuto occasione di parlare di questa struttura, formata da 10 *parole di 4 bit*, in sostanza le prime 10 del ***codice binario puro a 4 bit***, il cui compito è invece quello di raccogliere tutte le possibili combinazioni delle 4 variabili coinvolte, in sostanza $2^4 = 16$ *parole a 4 bit*, diverse tra loro.

La *Figura 1* mostra le tabelle che li rappresentano entrambi e sottolinea come il **BCD** sia un sottoinsieme del *codice binario puro a 4 bit*.

Concentriamo l'attenzione sulla somma di 2 numeri **BCD**; la natura del codice lascia sottointendere che ciascun numero sarà espresso da stringhe di 4 bit ma che ben 6 delle loro possibili combinazioni non saranno mai presentate al sommatore: i codici 1010, 1011, 1100, 1101, 1110, e 1111 non appartengono al codice e non verranno utilizzati; di sicuro le 10 codifiche **BCD** sono identiche alle 10 codifiche ***binarie pure*** di pari valore, per esempio $(9)_{10}=(1001)_2=(1001)_{BCD}$.

Poiché questo sommatore dovrà restituire un risultato espresso in **BCD** l'indisponibilità di queste *parole vietate* rende impossibile affidare questo compito ad un Adder classico, dato che esso esegue i suoi calcoli su numeri ***binari puri*** e dà per scontata la presenza di tutte le sequenze.

CODICE BINARIO PURO A 4 BIT

n	D	C	B	A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

CODICE BCD

n	D	C	B	A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Figura 1 - Codice Binario Puro a 4 bit e Codice BCD

Per fissare le idee supponiamo che i numeri da sommare siano $A=(9)_{10}=(1001)_2$ e $B=(3)_{10}=(0011)_2$; il risultato, $(12)_{10}=(1100)_2$, è decisamente diverso se espresso in BCD, $(00010010)_{BCD}$, perché con questa convenzione la cifra finale si ottiene accostando le quaterne di bit corrispondenti a ciascuna delle cifre decimali coinvolte, nel nostro caso $(0001)_{BCD}$ per $(1)_{10}$ e $(0010)_{BCD}$ per $(2)_{10}$.

Si vede immediatamente che, se dovessimo interpretare ***in binario puro*** il risultato **BCD**, esso sarebbe completamente sballato, $(00010010)_2 = (18)_{10}$, per cui è facile concludere che i sommatori presi in considerazione finora non sono adatti per questo compito.

Per ribadire il concetto, volendo trattare 2 numeri a 4 bit in presenza di un eventuale *Riporto* esterno, con la somma ***binaria pura*** sono possibili 32 risultati, da $(00000)_2=(0)_{10}$ a $(11111)_2=(15+15+1=31)_{10}$, mentre con quella **BCD** ne sono possibili solo 20, da $(00000000)_{BCD}=(0)_{10}$ a $(00011001)_{BCD}=(9+9+1=19)_{10}$.

Ragionando sulle 20 possibilità si arriva ad una conclusione: fin tanto che la somma di A e B rimane minore o uguale a $(9)_{10}$, il classico sommatore (realizzato con 4 **FA**, *Full Adder*) andrebbe ancora bene, mentre per risultati maggiori o uguali a $(10)_{10}$, per "*aggiustare*" il risultato da binario a BCD, sarà necessario imporre una correzione, sommandogli 6.

Un ***sommatore BCD*** è dunque realizzabile con 2 normali **4-bit Binary Full Adder** e una piccola rete logica, il cui compito è quello di attivare o meno (con l'aiuto del secondo) la somma del numero 6 al risultato del primo; la letteratura prevede per essa diverse soluzioni: la *Figura 2* e la *Figura 3* offrono un paio di alternative.

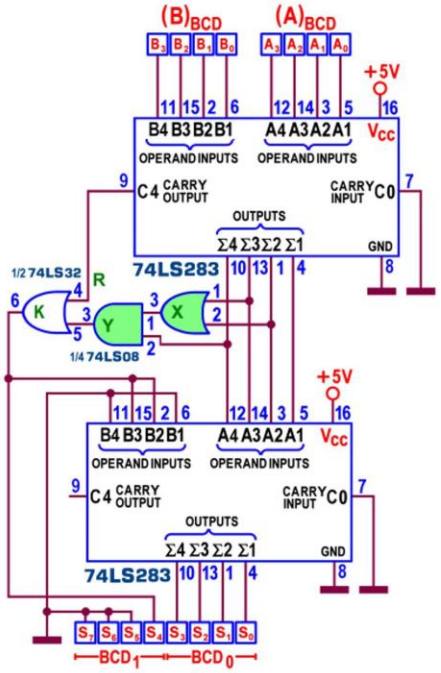


Figura 2 - 4-bit BCD Adder: Schema N°1 con 74LS283

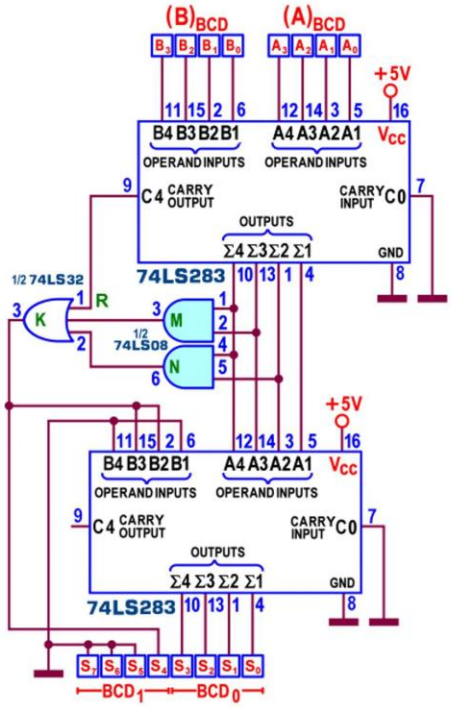


Figura 3 - 4-bit BCD Adder: Schema N°2 con 74LS283

I progetti proposti utilizzano 2 componenti TTL **74LS283** ma funzioneranno bene anche con 2 TTL **74LS83** o 2 CMOS **4008**, tutti analizzati la scorsa puntata.

Per entrambi, con molta pazienza, ho ricostruito le condizioni che portano ai 20 possibili diversi risultati (la *Figura 4*), a partire dal valore corrente del primo sommatore (un numero binario puro a 5 bit, tenendo conto anche del suo Riporto, $C_4\Sigma_4\Sigma_3\Sigma_2\Sigma_1$); si vede chiaramente che, in funzione del risultato binario corrente, il secondo sommatore gli aggiungerà o 0 o 6, fornendo sulle sue uscite $\Sigma_4\Sigma_3\Sigma_2\Sigma_1$ un valore certamente BCD, affidato alla cifra meno significativa BCD_0 del risultato, mentre la cifra più significativa BCD_1, immediatamente disponibile sull'uscita K della OR, è a tutti gli effetti la linea di Carry Out del nuovo dispositivo.

n	C_4	Σ_4	Σ_3	Σ_2	Σ_1	M AND	N AND	R C_4	X OR	Y AND	K = OR	BCD_1		BCD_0
0	0	0	0	0	0	0	0	0	0	0	0	0000	0+0	0000
1	0	0	0	0	1	0	0	0	0	0	0	0000	1+0	0001
2	0	0	0	1	0	0	0	0	1	0	0	0000	2+0	0010
3	0	0	0	1	1	0	0	0	1	0	0	0000	3+0	0011
4	0	0	1	0	0	0	0	0	1	0	0	0000	4+0	0100
5	0	0	1	0	1	0	0	0	1	0	0	0000	5+0	0101
6	0	0	1	1	0	0	0	0	1	0	0	0000	6+0	0110
7	0	0	1	1	1	0	0	0	1	0	0	0000	7+0	0111
8	0	1	0	0	0	0	0	0	0	0	0	0000	8+0	1000
9	0	1	0	0	1	0	0	0	0	0	0	0000	9+0	1001
10	0	1	0	1	0	0	1	0	1	1	1	0001	10+6	10000
11	0	1	0	1	1	0	1	0	1	1	1	0001	11+6	10001
12	0	1	1	0	0	1	0	0	1	1	1	0001	12+6	10010
13	0	1	1	0	1	1	0	0	1	1	1	0001	13+6	10011
14	0	1	1	1	0	1	1	0	1	1	1	0001	14+6	10100
15	0	1	1	1	1	1	1	0	1	1	1	0001	15+6	10101
16	1	0	0	0	0	0	0	1	0	0	1	0001	16+6	10110
17	1	0	0	0	1	0	0	1	0	0	1	0001	17+6	10111
18	1	0	0	1	0	0	0	1	1	0	1	0001	18+6	11000
19	1	0	0	1	1	0	0	1	1	0	1	0001	19+6	11001

Figura 4 - 4-bit BCD Adder: Analisi del funzionamento

La stessa tecnica di calcolo utilizzata da questi circuiti è spesso assicurata anche dai microprocessori che, nelle operazioni aritmetiche con operandi BCD, mettono a disposizione la speciale istruzione **DAA** (*Decimal adjust After Addition*) capace proprio di correggere in BCD il risultato binario di una precedente istruzione di somma (**ADD**); inutile ribadire che l'aggiustamento consiste nell'aggiungere una costante uguale a 6 al risultato della prima somma, se ha superato (*decimal overflow*) il valore massimo per essa previsto.

La serie **TTL** prevede un sommatore di 2 numeri BCD: il **74F583**, detto **4-bit BCD Adder**, anch'esso dotato internamente (come i suoi simili ***binari***) di circuiti di tipo ***Carry Look Ahead*** adatti alla previsione veloce del Riporto; la *Figura 5* mostra il suo *pin-out*: esteticamente sembra lo stesso dei Binary Adder **74LS83** e **74LS283** ma noi sappiamo che ne contiene 2, con della ulteriore logica aggiuntiva.

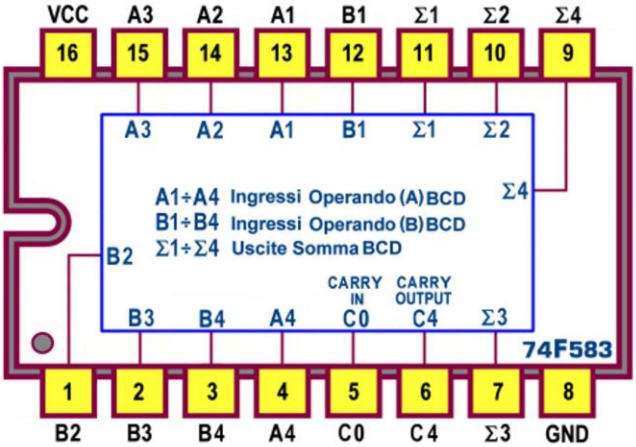


Figura 5 - 4-bit BCD Adder 74F583: Pin-out

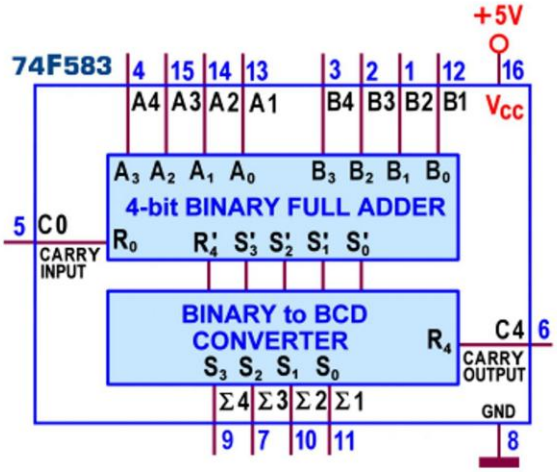


Figura 6 - 4-bit BCD Adder 74F583: Schema funzionale

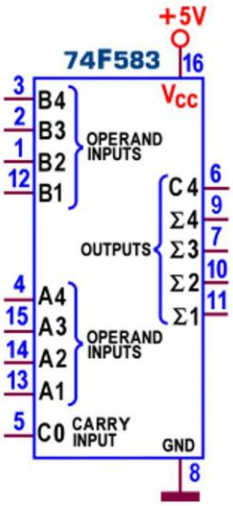


Figura 7 - 4-bit BCD Adder 74F583: Schema pratico

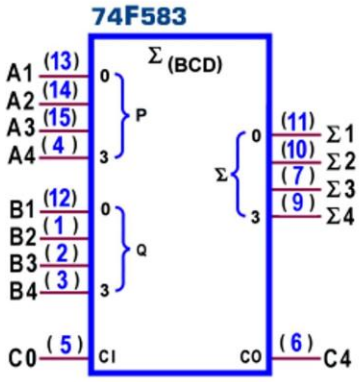


Figura 8 - 4-bit BCD Adder 74F583: Simbolo logico ANSI/IEEE Std. 91-1984

La *Figura 6* e la *Figura 7* mostrano rispettivamente lo *schema funzionale* e lo *schema pratico*, mentre il *simbolo logico* predisposto dallo *standard IEEE* è visibile in *Figura 8*; in quest'ultima i numeri sugli ingressi degli addendi e sulle uscite della somma indicano il peso di ciascun piedino, inteso come potenza di 2.

La potenza dissipata massima è di **300 mW**; il ritardo di propagazione massimo (misurato con carico di **500**ohm/**50**pF) tra gli ingressi di dato e le uscite della Somma è di **17**ns/**14**ns (t_{PLH}/t_{PHL}) mentre quello tra ingressi dato e uscita Riporto C4 è ancora minore, **11.5**ns/**10.5**ns (t_{PLH}/t_{PHL}). Alla stessa famiglia appartiene anche il **82S83** (TTL Schottky) prodotto dalla Signetics, funzionalmente identico a quello appena descritto, con tempi di ritardo doppi.

Sommatori BCD (CMOS): 4560

Il sommatore di 2 numeri BCD della serie **CMOS** è il **4560**, detto **NBCD** (*Natural BCD*) **Adder**, dotato di circuiti di tipo ***Carry Look Ahead*** per la determinazione veloce del Riporto; il suo *pin-out* (*Figura 9*) è sostanzialmente uguale a quello dei suoi simili TTL, ma con piedinatura diversa.

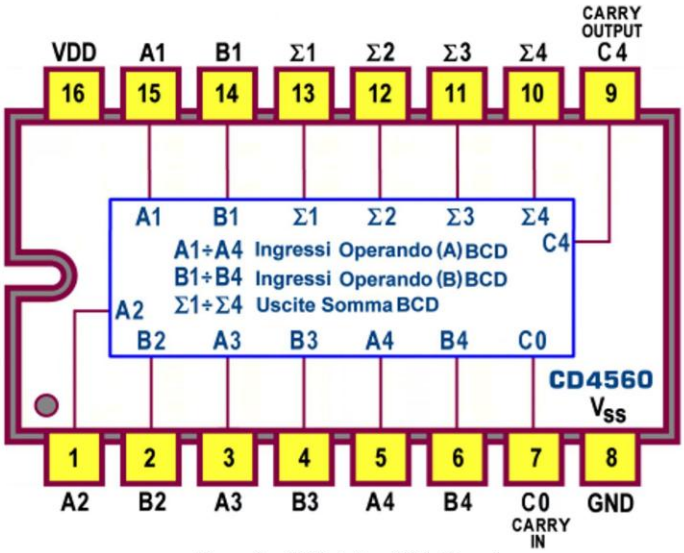


Figura 9 - NBCD Adder 4560: Pin-out

Lo *schema funzionale* e quello *pratico* sono proposti rispettivamente in *Figura 10* e in *Figura 11*; questo dispositivo contiene circuiti di protezione per evitare danni dovuti a scariche

elettrostatiche e sovratensione; gli ingressi non utilizzati devono sempre essere collegati ad un adeguato livello di tensione logica, di solito uno dei 2 terminali d'alimentazione, o $\mathbf{V_{SS}}$ o $\mathbf{V_{DD}}$).

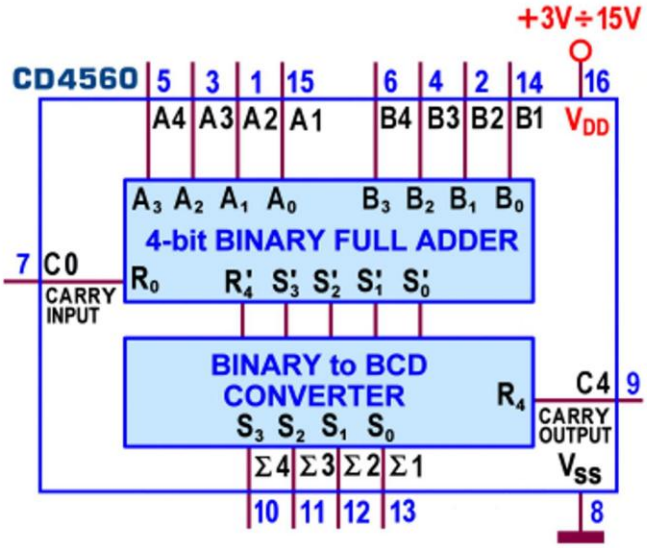


Figura 10 - NBCD Adder 4560: Schema funzionale

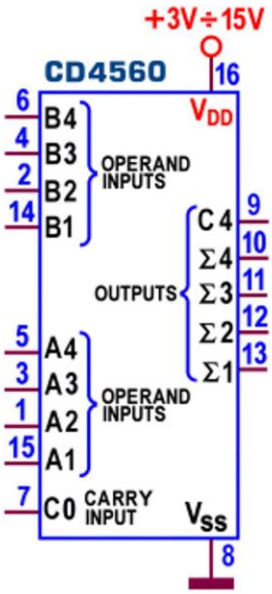


Figura 11 - NBCD Adder 4560: Schema pratico

Come per gli altri componenti della famiglia **CMOS** la *potenza dissipata* è trascurabile mentre il *ritardo di propagazione* (*Propagation Delay Time*, $\mathbf{t_{PHL}}$ e $\mathbf{t_{PLH}}$) massimo per disporre di uscite stabili dopo l'applicazione degli addendi va da **2100ns** ($\mathbf{V_{DD}}$=**5V**) a **675ns** ($\mathbf{V_{DD}}$=**15V**).

Naturalmente tutti i dispositivi *sommatori BCD* (TTL o CMOS) possono essere messi in cascata per trattare numeri con più cifre BCD; la *Figura 12* mostra un circuito in grado di addizionare 2 numeri da 0 a 999, ciascuno espresso dai 12 bit tipici della codifica da essi espressa; il risultato (esso pure in BCD) sarà disponibile sui 16 bit d'uscita.

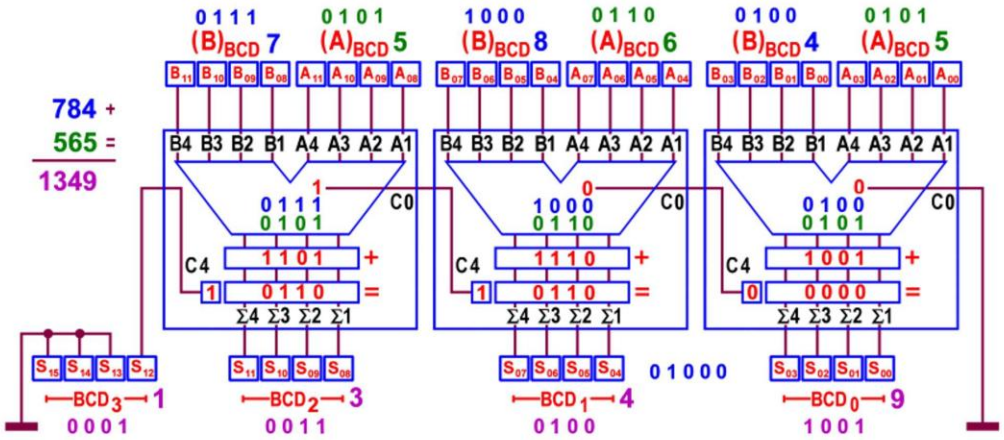


Figura 12 - BCD Adder: Somma di 784+565 in BCD

In ciascuno dei 3 sommatori BCD coinvolti ho evidenziato il *sommatore binario interno a 4 bit* (incaricato di operare i dati forniti in ingresso) e il circuito d'aggiustamento (in sostanza un convertitore di codice da binario a BCD) chiamato ad aggiungere nulla, senza generare riporto (se il risultato precedente è minore o uguale a 9) oppure ad aggiungere la costante $(6)_{10}=(0110)_2$, forzando a 1 l'uscita del riporto (in caso contrario). Il dispositivo somma il numero $(011110000100)_{BCD}=(784)_{10}$ con il numero $(010101100101)_{BCD}=(565)_{10}$ mostrando in dettaglio ogni operazione interna.

A corollario della trattazione sui sommatori voglio citare la disponibilità di 2 intergrati funzionalmente compatibili ma logicamente diversi: si tratta dei componenti **CMOS 4032** e **4038**, definiti **Triple Serial Adders** dai datasheet; la differenza sostanziale sta nel fatto che essi sono macchine sequenziali (non combinatorie, come quelle trattate finora), chiamate a sommare 2 informazioni binarie seriali.

I bit di ciascuno dei 2 pacchetti di dati vengono proposti su altrettanti ingressi, a partire da quello meno significativo e la loro somma, colonna per colonna, proposta sull'uscita in modulo 2) richiede la presenza di un segnale di clock, sul fronte di salita (**4032**) o di discesa (**4038**) del quale ai *bit* di uguale peso di ciascun ingresso viene aggiunto l'eventuale riporto maturato nella somma dei *bit precedenti*.

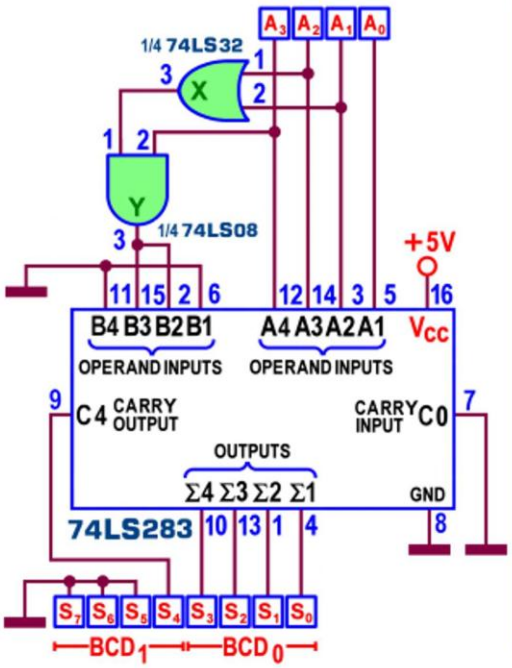
Questi componenti contengono 3 circuiti sommatori seriali indipendenti, ciascuno dotato di un segnale di comando (*inverter*) che consente (se forzato a 1 logico) di invertire i valori logici della somma disponibile in uscita; hanno però in comune il sincronismo di clock e una linea di Reset-Carry; quest'ultima viene forzata a 1 logico alla fine di ogni parola, un attimo prima (pari ad un periodo di clock) dell'arrivo del primo bit della parola successiva, per resettare il Riporto.

Per parole di N bit sono necessari N cicli di clock, la cui frequenza massima è valutabile intorno ai 10 MHz.

Convertitore Binario - BCD (TTL): 74185

Abbiamo visto che l'operazione di somma di 2 numeri BCD comporta la necessità di disporre di un circuito in grado di convertire il risultato binario intermedio in BCD, per adattarlo al suo formato.

La *Figura 13* e la *Figura 14* mostrano il convertitore di un numero binario a 4 bit nella corrispondente espressione BCD; entrambi basano il loro funzionamento sull'algoritmo di aggiustamento decimale, rilevando le quaterne di bit non appartenenti al codice BCD (cioè quelle di valore maggiore di 9) e sommando in questi casi la costante 6 al valore binario corrente.



n	A_3	A_2	A_1	A_0	X OR	Y AND	BCD_1	BCD_0
0	0	0	0	0	0	0	0000	0000
1	0	0	0	1	0	0	0000	0001
2	0	0	1	0	1	0	0000	0010
3	0	0	1	1	1	0	0000	0011
4	0	1	0	0	1	0	0000	0100
5	0	1	0	1	1	0	0000	0101
6	0	1	1	0	1	0	0000	0110
7	0	1	1	1	1	0	0000	0111
8	1	0	0	0	0	0	0000	1000
9	1	0	0	1	0	0	0000	1001
10	1	0	1	0	1	1	0001	0000
11	1	0	1	1	1	1	0001	0001
12	1	1	0	0	1	1	0001	0010
13	1	1	0	1	1	1	0001	0011
14	1	1	1	0	1	1	0001	0100
15	1	1	1	1	1	1	0001	0101

Figura 13 - Binary to BCD Converter: Schema N°1 per ingresso binario a 4 bit

Il primo è un estratto dello schema del sommatore BCD visto in precedenza e utilizza 2 sole porte logiche per rilevare la necessità dell'aggiustamento; il secondo, decisamente meno economico, ha una valenza didattica in quanto esercita "alla lettera" il confronto del numero binario in ingresso con il valore 9, utilizzando un comparatore **74LS85** e forzando la somma con 6 con l'aiuto della sua linea d'uscita A>B.

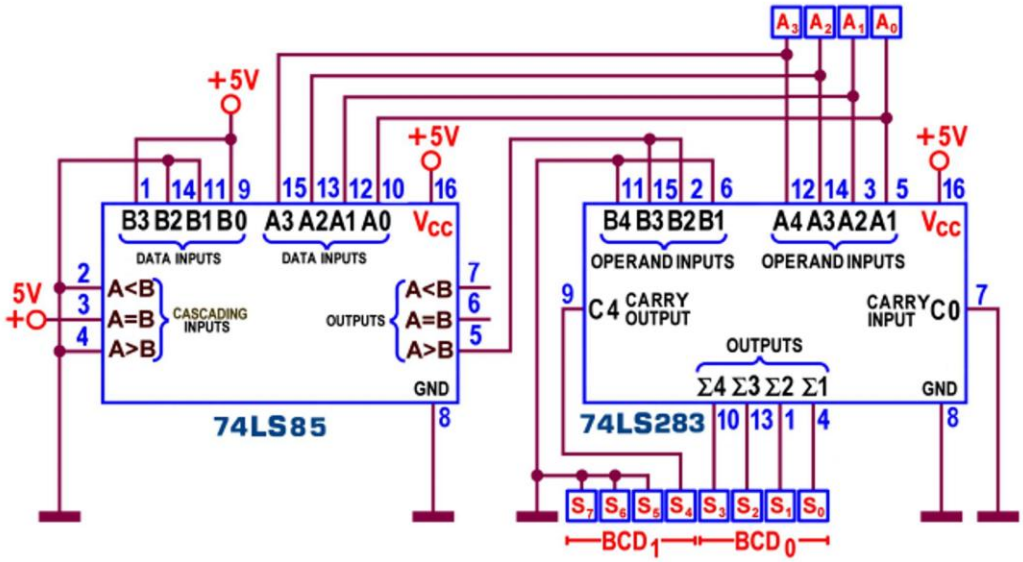


Figura 14 - Binary to BCD Converter: Schema N°2 per ingresso binario a 4 bit

Poiché il problema è sentito, anche la serie **TTL** prevede una versione di **Binary to BCD Converter**: il **74185**, un integrato con uscite di tipo open-collector; esiste la versione **8899**, perfettamente identica, ma con uscite tre-state.

Si tratta di un componente derivato dalla memoria a sola lettura DM7488, una ROM da 256 bit; la *Figura 15* mostra il suo pin-out.

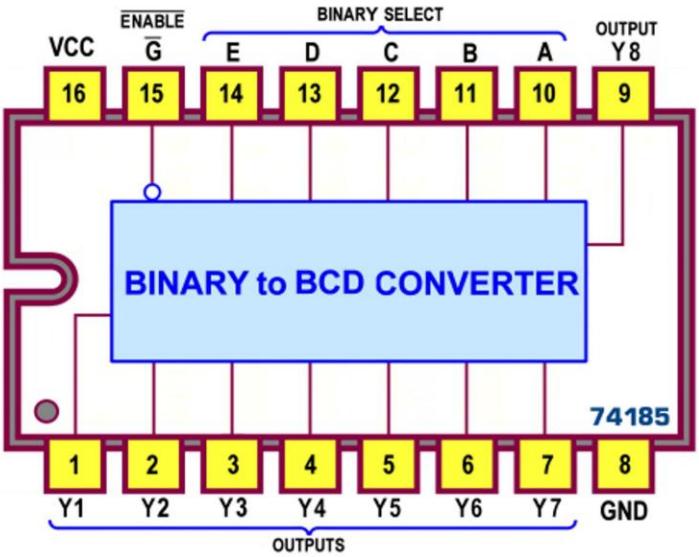


Figura 15 - Binary to BCD Converter 74185: Pin-out

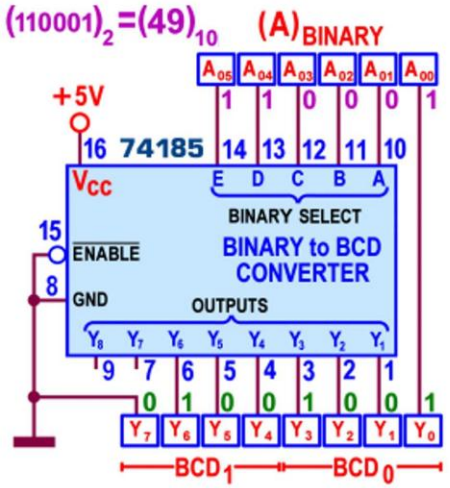
La versatilità di questi convertitori è frutto della loro natura, data l'alta capacità di memorizzare nella ROM un numero illimitato di tabelle di riferimento o di tabelle di conversione, che consente il richiamo diretto (*read-out*) dei codici in essa contenuti sulle uscite da Y_8 a Y_1.

Da notare la presenza di un piedino di abilitazione (*Enable*) che, nella fase operativa, va messo a massa (0 logico); in caso contrario tutte le uscite sono forzate a 1, indipendentemente dal valore corrente del numero in ingresso.

I manuali forniscono una pesante Tabella di Verità, improponibile in questa sede: poiché la capacità intrinseca del componente è quella di garantire la conversione in BCD di un numero binario a 6 bit, sono previste 32 righe (una per ogni coppia di valori trattabili) e 14 colonne in cui sono raccolte, tra l'altro, le possibili combinazioni dei soli 5 bit più significativi del valore binario in ingresso e dei 6 bit più significativi del risultato BCD.

Pur non vedendo la Tabella, da questa descrizione è subito evidente che in essa non è presente il bit meno significativo (LSB) né del codice binario fornito in ingresso (gli inputs *Binary Select* E, D, C, D e A) né di quello BCD restituito in uscita (*outputs* Y_6, Y_5, Y_4, Y_3, Y_2, Y_1); la cosa imbarazza non poco, in prima lettura, ma è giustificata dal fatto che questo bit, essendo logicamente lo stesso da entrambi i punti di vista, non viene coinvolto dal componente ma passa esternamente ad esso, dal dato sorgente al risultato. Le uscite Y_8 e Y_7, non utilizzate in questo contesto, sono programmate alte (nella tabella originale sono a 1 logico in ogni situazione).

La *Figura 16* mostra lo schema applicativo per la conversione in BCD di un numero binario a 6 bit; come detto, è sufficiente un solo componente e la Tabella offerta a corredo (personalizzata con informazioni non disponibili in quella originale) aiuta a capire quanto spiegato finora.



n	BINARY SELECT						BCD1				BCD0			
	E	D	C	B	A	n_0	n_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	n_0
0 1	0	0	0	0	0	0/1	0	0	0	0	0	0	0	0/1
2 3	0	0	0	0	1	0/1	0	0	0	0	0	0	1	0/1
4 5	0	0	0	1	0	0/1	0	0	0	0	0	1	0	0/1
6 7	0	0	0	1	1	0/1	0	0	0	0	0	1	1	0/1
48 49	1	1	0	0	0	0/1	0	1	0	0	1	0	0	0/1
56 57	1	1	1	0	0	0/1	0	1	0	1	0	1	1	0/1
58 59	1	1	1	0	1	0/1	0	1	0	1	1	0	0	0/1
60 61	1	1	1	1	0	0/1	0	1	1	0	0	0	0	0/1
62 63	1	1	1	1	1	0/1	0	1	1	0	0	0	1	0/1

Figura 16 - Binary to BCD Converter 74185: Applicazione per numero a 6 bit

Il numero sottoposto a conversione è $(110001)_2=(49)_{10}$ e fornisce il risultato $(01001001)_{BCD}$; è interessante sapere che la funzione svolta da questi componenti è analoga a quella dell'algoritmo di conversione classico, spesso tradotto in assembly per far fare la stessa cosa ad un processore.

Esso consiste nell'eseguire una quantità di scorrimenti (del numero da convertire verso sinistra) pari a quella delle sue cifre, controllando, dopo ogni *shift*, se i 3 bit più significativi di ciascuna delle decadi del risultato corrente hanno un valore superiore a 4; in questo caso ad esse si aggiunge 3 e poi si continua con gli spostamenti, se ne rimangono ancora da fare. La *Figura 17* sintetizza l'algoritmo per il numero trattato dal progetto precedente, $(110001)_2=(49)_{10}$.

Operazione	BCD_1	BCD_0	Binario
Valore iniziale 49			110001
Shift a sinistra n°1		1	10001
Shift a sinistra n°2		11	0001
Shift a sinistra n°3		110	001
Sommo 3 ai tre bit		1001	001
Shift a sinistra n°4	1	0010	01
Shift a sinistra n°5	10	0100	1
Shift a sinistra n°6	100	1001	
Valore finale	0100	1001	

Figura 17 - Algoritmo di conversione per numero binario a 6 bit

Gli schemi applicativi hanno una complessità crescente con il numero di bit della parola binaria da convertire; se il numero di bit è maggiore di 6 sarà necessario mettere in cascata più dispositivi: ne serviranno 3 per numeri binari a 8 bit; 4 per 9 bit; 8 per 12 bit; 19 per 24 bit... ; anche in tutti questi casi il bit meno significativo del risultato BCD sarà ottenuto direttamente dal bit meno significativo numero binario fornito in ingresso, senza passare attraverso i componenti coinvolti.

La *Figura 18* mostra lo schema per la conversione di un numero binario a 8 bit in BCD; a verifica della sua efficienza ho preso in considerazione un caso concreto, mettendo in evidenza l'azione di ciascuno dei 3 integrati sul numero sottoposto a conversione, $(10011010)_2=(154)_{10}$, fino al risultato $(000101010100)_{BCD}$; da notare che gli ingessi non utilizzati (nello schema ce n'è uno sul terzo integrato) vanno collegati a massa.

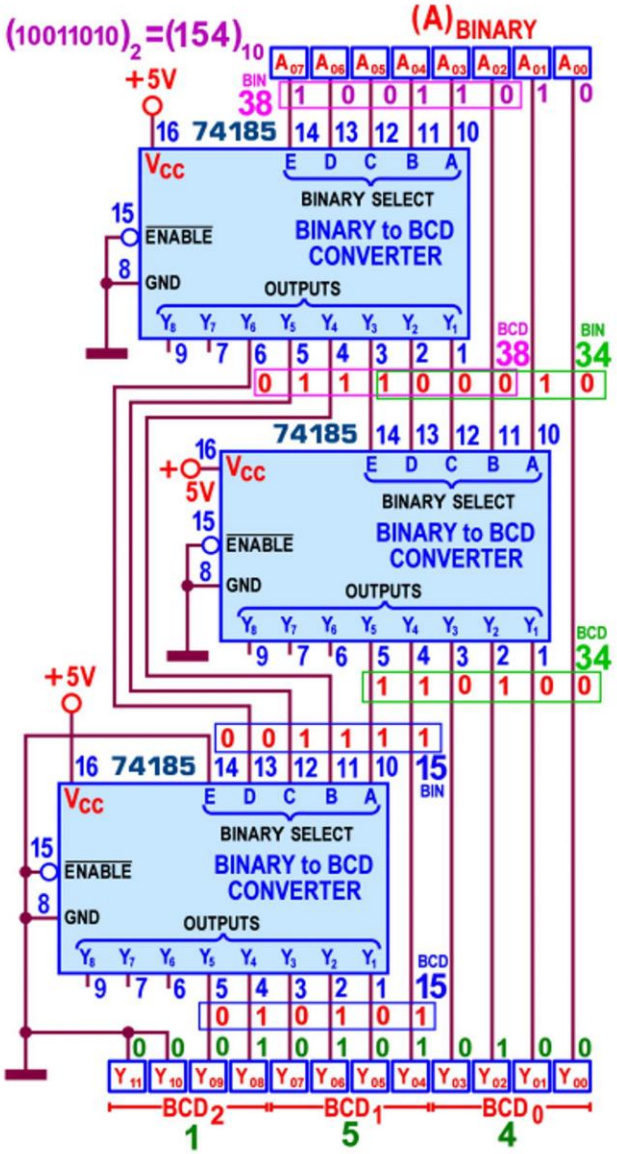


Figura 18 - Binary to BCD Converter 74185: Applicazione per numero a 8 bit

La potenza dissipata massima da questo componente è di **495 mW**; il suo ritardo di propagazione massimo (misurato con carico di **600**ohm/**15**pF) è di **35**ns per entrambe le transizioni $\mathbf{t_{PLH}}$ e $\mathbf{t_{PHL}}$; naturalmente aumentando la quantità di bit del numero binario da convertire si dovrà prevedere una quota per ogni dispositivo in cascata.

Convertitore BCD - Binario (TTL): 74184

Non di rado si presenta il problema opposto a quello appena discusso: numerosi dispositivi forniscono informazioni "impacchettate" in modo BCD, cioè costituite da una sequenza di parole appartenenti a quel codice, in modo da formare stringhe di lunghezza multipla di 4 bit.

Per poter fruire di informazioni in binario puro è dunque necessario disporre di un circuito in grado di ottenerle a partire dalle stringhe BCD; la *Figura 19* mostra uno straordinario convertitore costruito a partire da 2 sommatori binari tradizionali, in grado di trasformare una coppia di codici BCD (in tutto 8 bit) nel numero binario corrispondente.

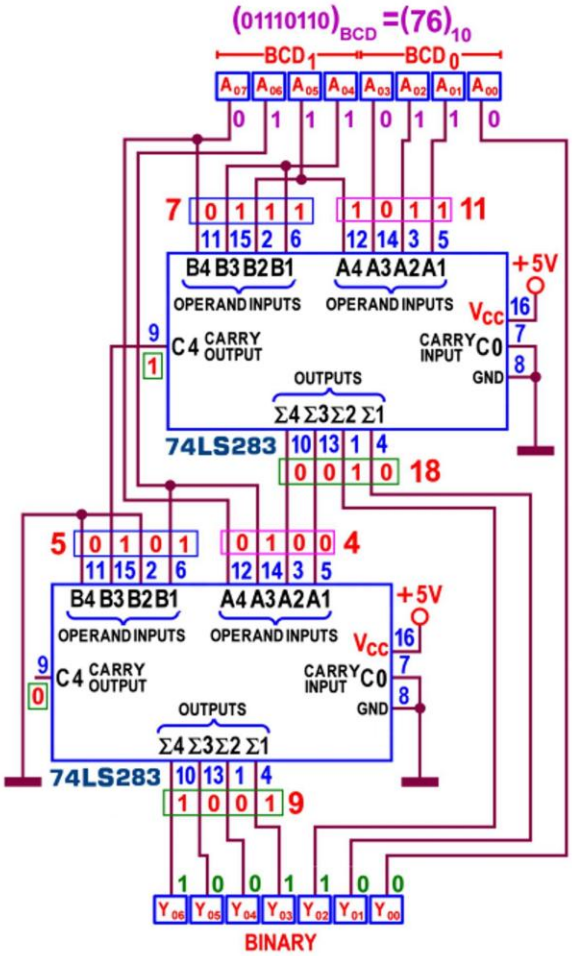


Figura 19 - BCD to Binary Converter: Schema con 74LS283

Il progetto utilizza un paio di **74LS283** (TTL), ma funzionerà bene anche con il TTL **74LS83** o il CMOS **4008**, visti la scorsa puntata); nello schema sono riportati i valori delle somme e dei

riporti parziali, supponendo di aver sottoposto a conversione il numero $(01110110)_{BCD}=(76)_{10}$, al fine di verificare il risultato atteso, $(1001100)_2$.

La serie **TTL** prevede un componente specifico, per questa operazione: il **74184**, definito **BCD to Binary Converter**, dotato di uscite di tipo open-collector, ma disponibile anche con uscite tre-state, nella versione **8898**, perfettamente identica in ogni particolare. Il pin-out (vedi *Figura 20*) e la natura logica, derivata dalla ROM (memoria a sola lettura) da 256 DM7488, sono gli stessi di quelli che esercitano il lavoro opposto.

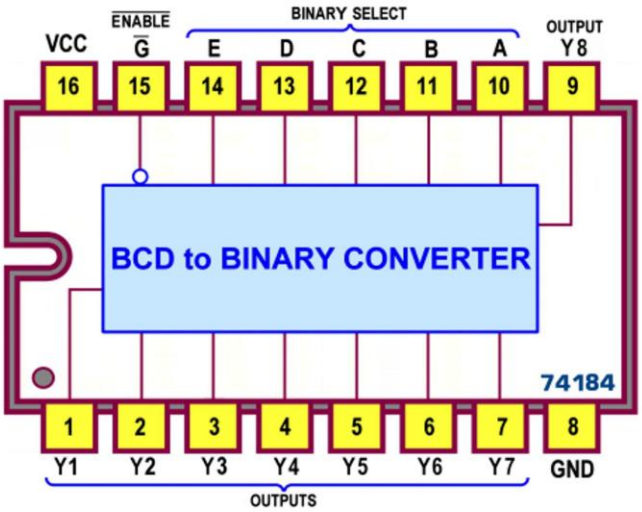
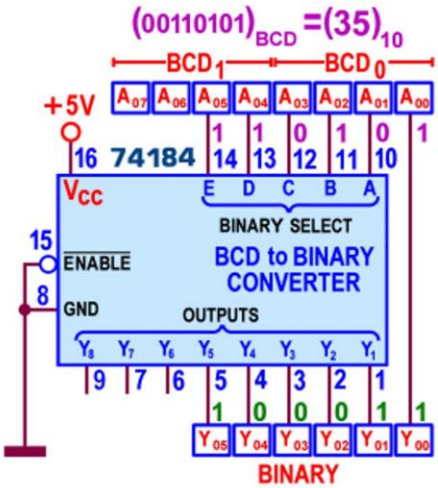


Figura 20 - BCD to Binary Converter 74184: Pin-out

Anche in questo caso è previsto un piedino di abilitazione (*Enable*) attivo basso, cioè da forzare a massa (0 logico) nella fase operativa; se esso è lasciato a 1 tutte le uscite sono forzate a 1.



n	BCD1				BCD0				BINARY					
	n_7	n_6	E	D	C	B	A	n_0	Y_5	Y_4	Y_3	Y_2	Y_1	n_0
0 1	0	0	0	0	0	0	0	0/1	0	0	0	0	0	0/1
2 3	0	0	0	0	0	0	1	0/1	0	0	0	0	1	0/1
4 5	0	0	0	0	0	1	0	0/1	0	0	0	1	0	0/1
6 7	0	0	0	0	0	1	1	0/1	0	0	0	1	1	0/1
32 33	0	0	1	1	0	0	1	0/1	1	0	0	0	0	0/1
34 35	0	0	1	1	0	1	0	0/1	1	0	0	0	1	0/1
36 37	0	0	1	1	0	1	1	0/1	1	0	0	1	0	0/1
38 39	0	0	1	1	1	0	0	0/1	1	0	0	1	1	0/1

Figura 21 - BCD to Binary Converter 74184: Applicazione per numero a 6 bit

La funzione primaria di questo componente è quella di convertire in binario puro "una cifra e mezza" BCD, nel senso che sono disponibili solo 6 linee d'ingresso; la *Figura 21* mostra lo schema di questa particolare applicazione, corredato da una tabella personalizzata che

sintetizza le varie possibilità; per questo tipo d'applicazione le uscite Y_8, Y_7 e Y_6 non sono utilizzate.

La Tabella originale tiene conto del fatto che, anche questa volta, il bit meno significativo (LSB) del codice BCD fornito sulle linee d'ingresso (E, D, C, D e A) e di quello binario restituito su quelle d'uscita (Y_5, Y_4, Y_3, Y_2, Y_1) non passa attraverso il componente ma va direttamente dal dato sorgente al risultato tramite una linea esterna (assumendo lo stesso valore in entrambi i numeri).

Per la tipologia degli ingressi sono dunque previste solo 20 righe (una per ogni coppia di valori trattabili) e 11 colonne (5 per le possibili combinazioni d'ingresso, 5 per quelle dei soli bit più significativi del valore binario in uscita, e una per il controllo Enable);

La *Figura 22* mostra lo schema per la conversione in binario di un numero BCD a 8 bit; anche per questo progetto ho messo in evidenza l'azione di ciascuno dei 2 integrati sul numero sottoposto a conversione, $(01110110)_{BCD}=(76)_{10}$, fino al risultato $(1001100)_2$, per verificarne l'efficienza.

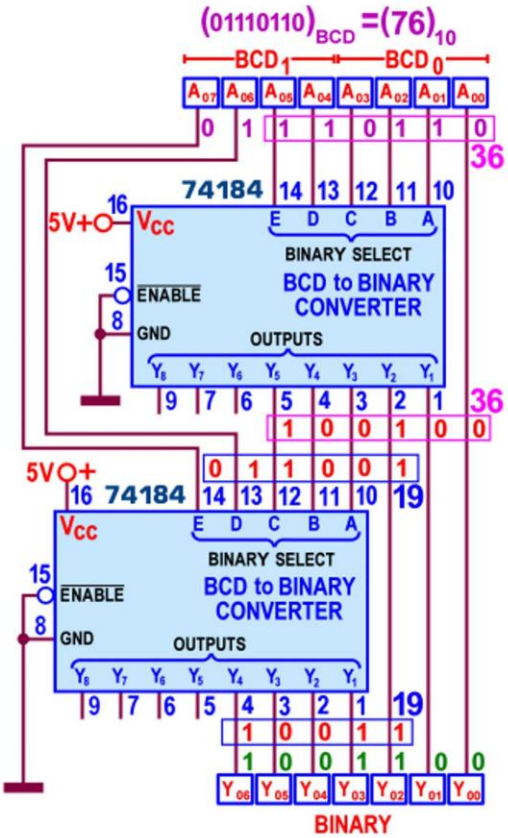


Figura 22 - BCD to Binary Converter 74184: Applicazione per numero a 8 bit

Se le cifre BCD sono più di 2 il numero di dispositivi da mettere in cascata cresce esponenzialmente: ne serviranno 6 per numeri BCD a 3 cifre (12 bit); 11 per 4 cifre; 19 per 5 cifre; 28 per 6 cifre ...

Senza entrare nel merito va ricordato che il **74184** è progettato anche per generare un numero BCD in *complemento a 9* o un numero BCD in *complemento a 10*, a partire da un

codice BCD proposto sugli ingressi D, C, D e A; il rimanente ingresso E è utilizzato come segnale di controllo: se messo a livello 0 logico il componente genera la conversione *BCD in complemento a 9*, altrimenti (forzato a 1) quella *BCD in complemento a 10*.

In entrambi i casi per questo servizio sono utilizzate solo le uscite Y_8, Y_7 e Y_6 (quelle non usate nella conversione *BCD to Binary*).

Le caratteristiche elettriche sono le stesse del suo omologo **74185**: potenza dissipata massima di **495 mW** e ritardo di propagazione massimo di **35**ns.