EsD2: Costruzione di D-Latch, contatori e shift-register

Gruppo 1.AC
Matteo Rossi, Bernardo Tomelleri

13 aprile 2022

1 Misura componenti dei circuiti

Riportiamo per completezza il valore della tensione continua di alimentazione per i circuiti integrati misurata con il multimetro

$$V_{CC} = 4.99 \pm 0.03\text{V}$$

e il valore di capacità del condensatore di disaccoppiamento che collega le linee di alimentazione a massa (sempre misurato con il multimetro)

$$C_d = 97 \pm 4 \text{ nF}$$

2 D-Latch con Enable

2.a Costruzione del circuito

Si è costruito un circuito D-Latch secondo lo schema mostrato in fig. 1 utilizzando le porte NAND di due integrati SN74LS00.

Per studiarne il comportamento generiamo nei due pin DIO 0 (DATA) e DIO 1 (ENABLE) dell'AD2 due segnali di clock di frequenza $f = 1$ kHz e sfasati tra loro di $\pm 90°$ agli ingressi D ed E del circuito.

2.b Analisi del funzionamento del circuito

Il circuito è composto da un Latch RS i cui ingressi sono collegati a due porte NAND, di cui un ingresso per ciascuna è collegato all'input E, mentre gli altri due ingressi sono collegati l'uno al segnale opposto dell'altro tramite una porta NOT (in figura la porta NAND più in alto tra le due (R) è collegata all'input D, mentre quella più in basso (S) a $\overline{D}$).

L'equazione fondamentale del circuito è quindi data dalla

$$Q(t + \Delta t) = \overline{\overline{(D \cdot E)}} + \overline{(\overline{D} \cdot E)} \cdot Q(t) = E \cdot D + \overline{E} \cdot Q(t) \tag{1}$$

da cui si può ricavare la corrispondente tabella di verità

Come si può vedere dalla tabella di verità (tabella 1) l'uscita Q funge da memoria a un bit se E è al livello logico basso (stato di HOLD), mentre assume il valore logico dell'input D quando il segnale di ENABLE è acceso. Questo rende il valore dell'uscita indipendente dalle caratteristiche temporali delle porte NAND e protegge il circuito dallo stato proibito di oscillazione/racing $R = S = 1$ da cui è affetto il semplice RS-Latch.

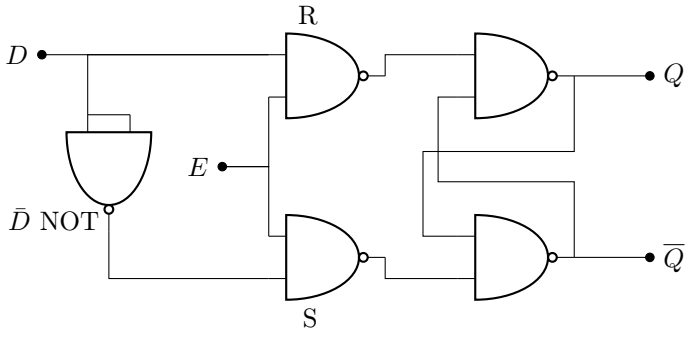

Figura 1: Schema logico del circuito D-Latch (con Enable) realizzato

E	D	$Q(t)$	$Q(t+\Delta t)$
0	X	0	0
0	X	1	1
1	0	X	0
1	1	X	1

Tabella 1: Tabella di verità del circuito D-Latch con Enable (con X si indica valore logico indefinito/don't care)

2.c Verifica della tabella di verità del Latch

Per conferma del corretto funzionamento del Latch possiamo confrontare le uscite ottenute da un'acquisizione con Logic Analyzer con i valori riportati in tabella 1 inviando all'ingresso del circuito con Patterns due segnali di clock sfasati tra loro di $\varphi = 90°$.

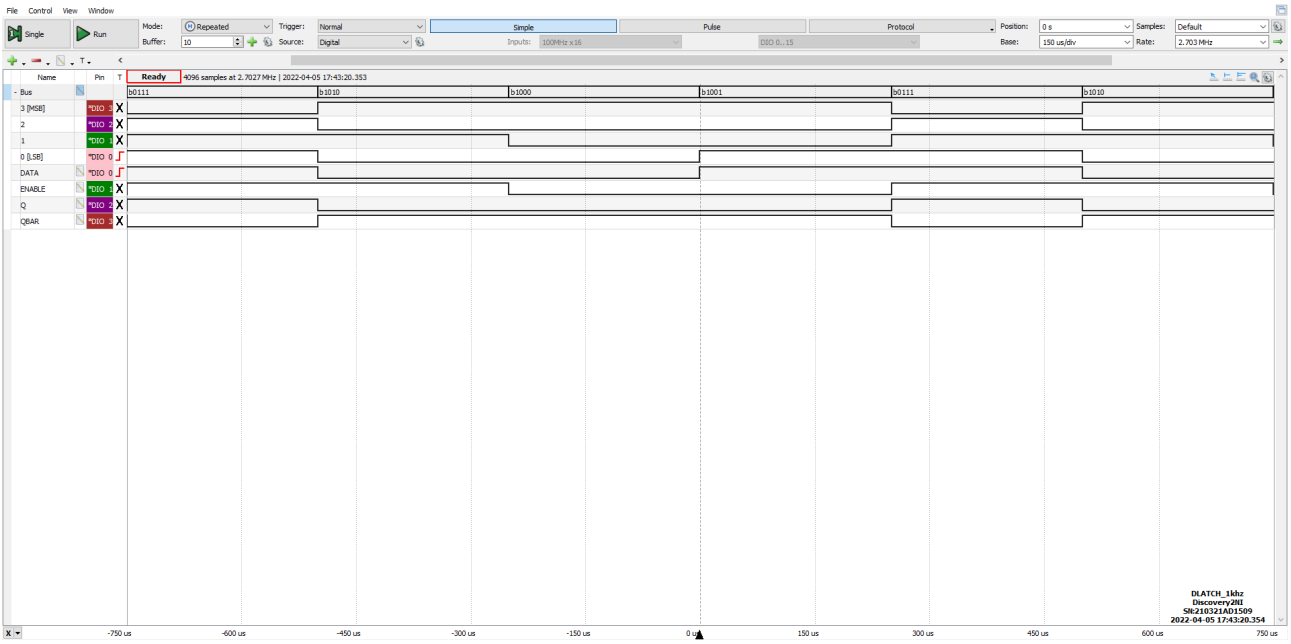

Figura 2: Acquisizione di un ciclo completo (frequenza 1 kHz) con Logic Analyzer dei segnali in ingresso (D = DIO 0, E = DIO 1) e in uscita (Q = DIO 2, $\overline{Q}$ = DIO 3) dal D-Latch.

Dalle fig. 2 e fig. 3 si osserva come durante lo stato basso di Enable il segnale in uscita rimanga costante rispetto a variazioni del segnale in D, mentre quando $E = 1 \implies Q(t+\Delta t) = D$ coerentemente con quanto previsto dalla tabella di verità.

2.d Misura dei tempi del ritardo nelle transizioni di stato

Si riescono a distinguere due diverse transizioni dei segnali in ingresso per ciascun valore di sfasamento tra i due segnali di clock in D ed E; per $\varphi = 90°$:

1. $D : 1 \to 0$, $E \coloneqq 1$
2. $D \coloneqq 1$, $E : 0 \to 1$.

   Mentre per $\varphi = -90° = 270$ °:
3. $D \coloneqq 0$, $E : 0 \to 1$
4. $D : 0 \to 1$, $E \coloneqq 1$.

Il ritardo di durata maggiore risulta quello della transizione 1 dell'input D da alto a basso (40 ± 10 ns).

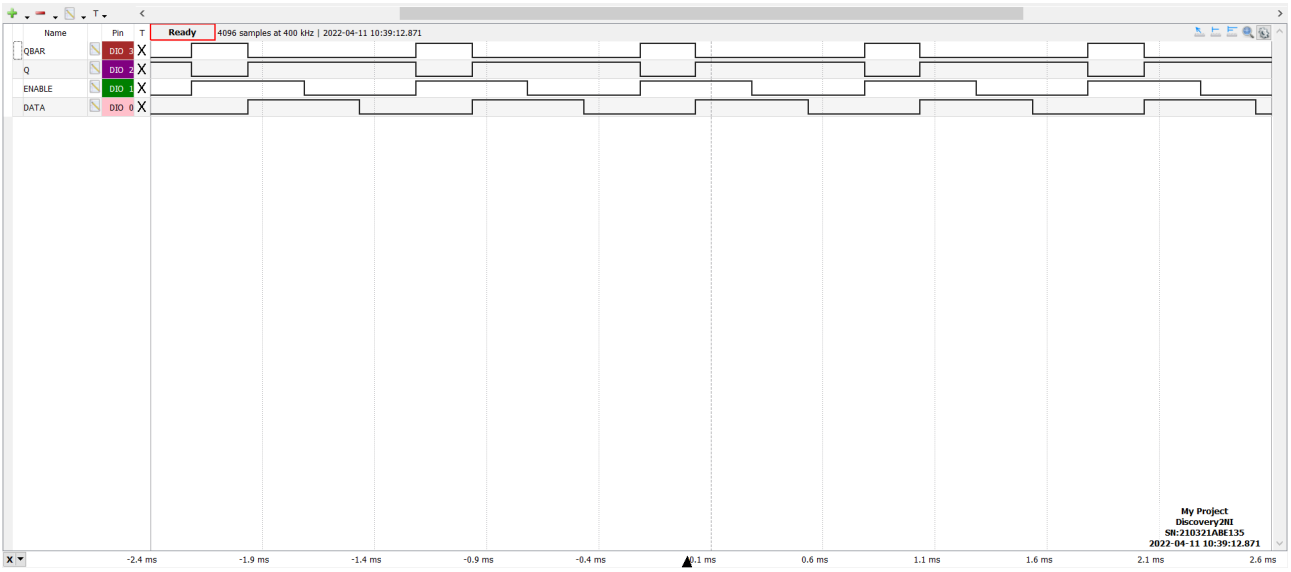

Figura 3: Acquisizione temporale con Logic dei segnali in ingresso uscita dal D-Latch per $\varphi = -90°$.

Dalle misure prese con i cursori dell'oscilloscopio (a cui associamo come incertezza il contributo dato dalle specifiche del datasheet, tenendo conto dell'instabilità delle tracce sullo schermo) troviamo

$$t_{PLH} = 11 \pm 1 \text{ ns}$$
$$t_{PHL} = 35 \pm 2 \text{ ns}$$

Dalle specifiche del DS si trova che i tempi di propagazione tipici e massimi per una singola porta NAND sono:

	typ	max	[units]
t_{PLH}	11	22	ns
t_{PHL}	7	15	ns

3 Shift-register con edge-triggered D-Flip Flop

3.a Costruzione del circuito

Si vuole ora costruire uno Shift Register a 4 bit a partire dagli integrati della serie SN74LS74, secondo lo schema in fig. 4 e verificarne il funzionamento.

3.b Verifica della sincronia delle uscite tramite PRESET

Per prima cosa dopo aver montato il circuito verifichiamo la sincronicità delle commutazioni delle uscite. Dopo aver controllato che le uscite Q0, Q1, Q2 e Q4 fossero nello stato 0000, mantenendo il segnale di clock e il D-Switch scollegati abbiamo utilizzato la funzione StaticIO di wavegen per pilotare il pin PRE-BUTTON di fig. 4 tramite un button di tipo pressed=0 e released=1 e Logic per programmare un trigger che facesse partire l'acquisizione alla pressione del button, osservando le tracce prodotte dai 4 segnali in uscita. Dalla figura fig. 5 vediamo che le commutazioni delle uscite avvengono in maniera sincrona, a un $\Delta T = 40$ns a partire dalla pressione del pulsante di preset; successivamente le 4 uscite hanno raggiunto lo stato 1111.

3.c Verifica del funzionamento tramite clock

A questo punto si vuole verificare il funzionamento del registro a scorrimento tramite un segnale di clock. Possiamo quindi costruire una tabella di verità in funzione del tempo del registro a partire dal periodo T del clock inviato: Come strategia per la verifica, utilizzeremo inizialmente il pulsante di preset per inizializzare tutte le uscite a 1, successivamente imposteremo il D-Switch tramite StaticIO come switch di tipo Push-Pull impostandolo a 0 e invieremo un segnale di clock dell'ordine di 1 Hz. Utilizzeremo quindi Logic per acquisire gli andamenti nel tempo delle uscite. Dunque ci si aspetta che dopo 3 periodi di clock (3 secondi) a partire da quando l'uscita Q0 diventa 0, tutte le uscite diventino 0, e che aspettando ulteriormente questi valori non

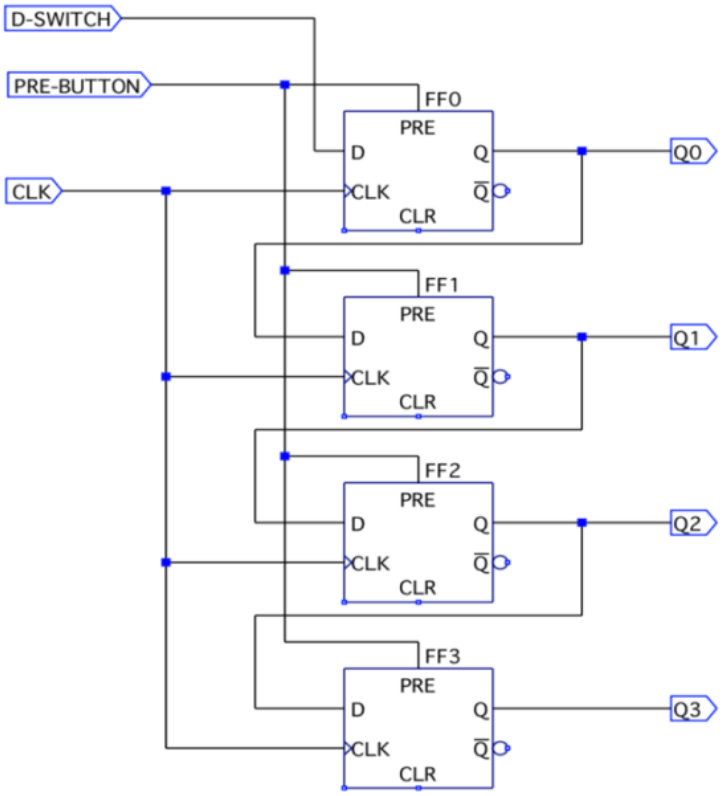

Figura 4

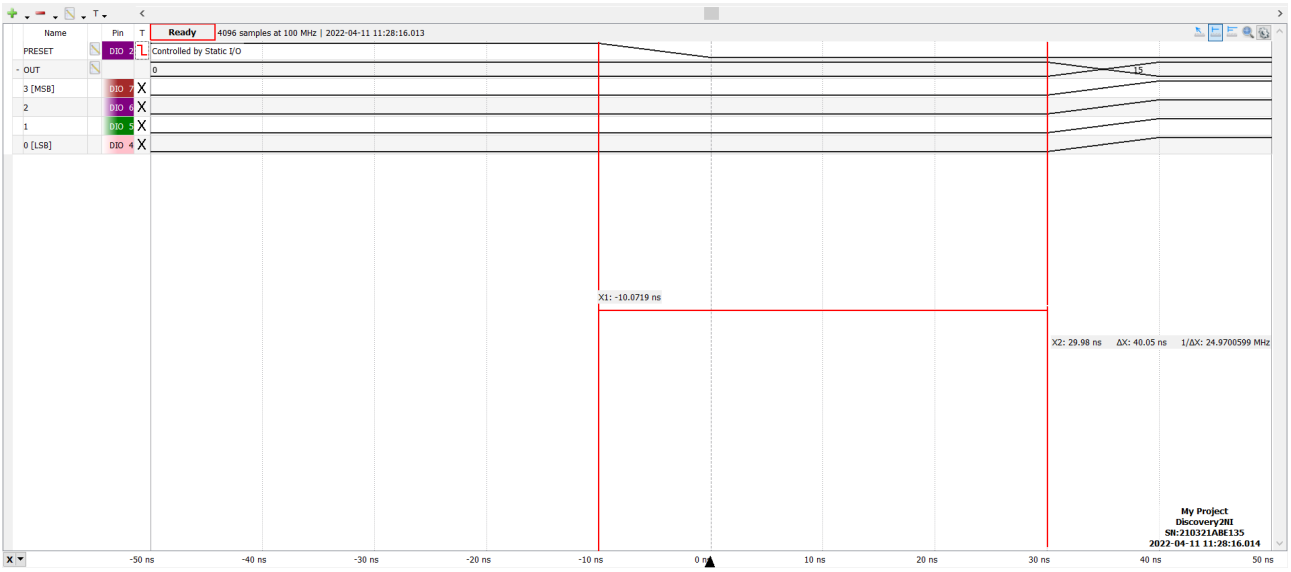

Figura 5: Acquisizione temporale con Logic dei segnali di PRE-BUTTON e i 4 segnali in uscita da un registro a scorrimento di 4 bit come illustrato in fig. 4

$t = t'$	$t = t' + \Delta T$	
Q_0	Q'_0	Q'_0
Q_1	Q_0(t=t')	Q_0(t=t')
Q_2	Q_1(t=t')	Q_0(t=t'-ΔT)
Q_3	Q_2(t=t')	Q_0(t=t'-2ΔT)

Tabella 2: Tabella "di verità" di un registro a scorrimento, Q'_0 è il valore che viene inviato durante l'impulso di clock tramite il D-SWITCH

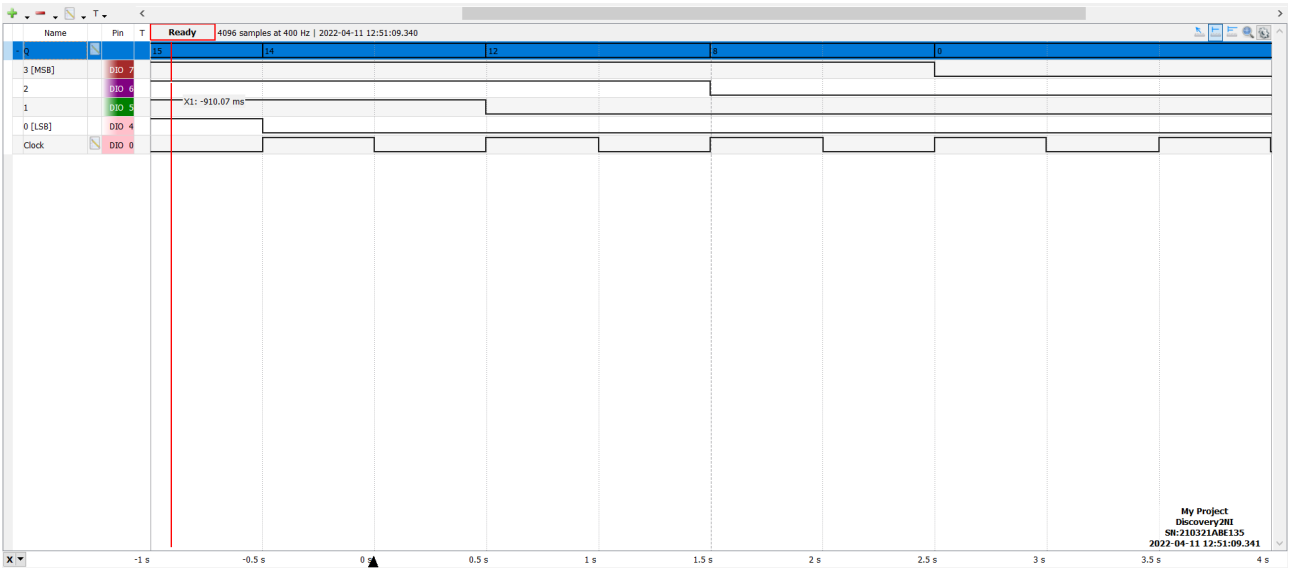

Figura 6: Acquisizione temporale con Logic dei segnali in uscita da un registro a scorrimento di 4 bit come spiegato in sezione 3.c

cambino. Dalla fig. 6 si verifica quanto detto prima, e le 4 uscite diventano (e si mantengono nel tempo visto che il D-Switch resta fisso a 0) tutte quante 0. Da questo si intuisce che collegando l'uscita Q_3 all'entrata del D-switch, possiamo generare una sequenza periodica.

3.d Prevalenza tra gli ingressi D-switch e PRESET

Utilizzando allo stesso tempo gli ingressi D-switch e PRESET del registro si vede che è l'ultimo a pilotare il segnale in uscita, dal momento che PRESET è asincrono (indipendente dalla salita del clock). Questo risulta consistente con la regola generale per cui le architetture asincrone prendono la precedenza sulle istruzioni sincrone.

3.e Contatore BCD con Flip Flop ad anello

Dopo aver impostato tutte le uscite a 0, si collega l'uscita $\overline{Q_3}$ all'entrata D del primo Flip-Flop e si invia un clock di frequenza pari a 1 kHz al circuito. Ci si aspetta che, prendendo un'uscita a caso, si osservi un segnale di clock di frequenza un quarto di quella di clock: questo effetto è dovuto a come la sequenza viene caricata nel registro a partire dall'uscita $\overline{Q_3}$. In generale supponendo che nel circuito ci siano in cascata n Flip-Flop, e che tutte le uscite siano inizializzate a 0, in una qualsiasi delle uscite otterrò un clock di frequenza pari a $\frac{f_{\text{clk}}}{n}$.

4 Generatore di sequenze pseudo-casuali

4.a Costruzione del circuito

Si vuole ora costruire un generatore di sequenze pseudo-casuali a 4 bit utilizzando lo shift register costruito in precedenza e una porta XOR; la schematica del circuito che utilizzeremo è riportato in figura fig. 8.

4.b Analisi e verifica del funzionamento

Dopo aver montato il circuito si inizializzano tutti Flip-Flop a 1 e si invia un segnale di clock a 10 kHz per verificare il funzionamento: essendo il registro di lunghezza pari a 4 bit, dalla teoria ci aspettiamo che la sequenza si ripeta dopo $2^4 = 16$ eventi al massimo, condizione che si ottiene utilizzando come TAP (segnali in ingresso alla porta XOR, la cui uscita sarà inviata all'entrata D del primo Flip-Flop) le uscite Q_2 e Q_3.

Dalla fig. 9 si verificano le aspettative per cui la sequenza generata a una qualsiasi uscita si ripete ogni 16 periodi di clock (essendo uno shift register la sequenza nelle altre uscite sarà la medesima, solo che saranno sfasate lungo l'asse temporale le une con le altre).

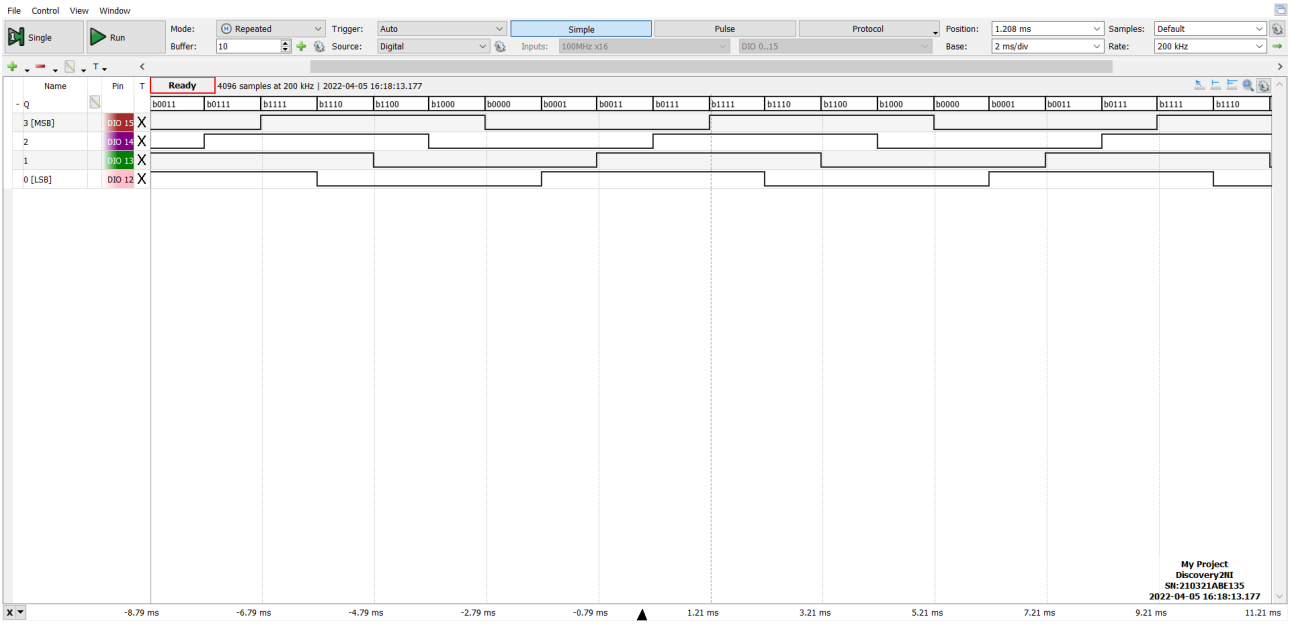

Figura 7: Acquisizione temporale con Logic dei segnali in uscita da un registro a scorrimento di 4 bit con l'ultima uscita negata collegata all'entrata del primo flip flop

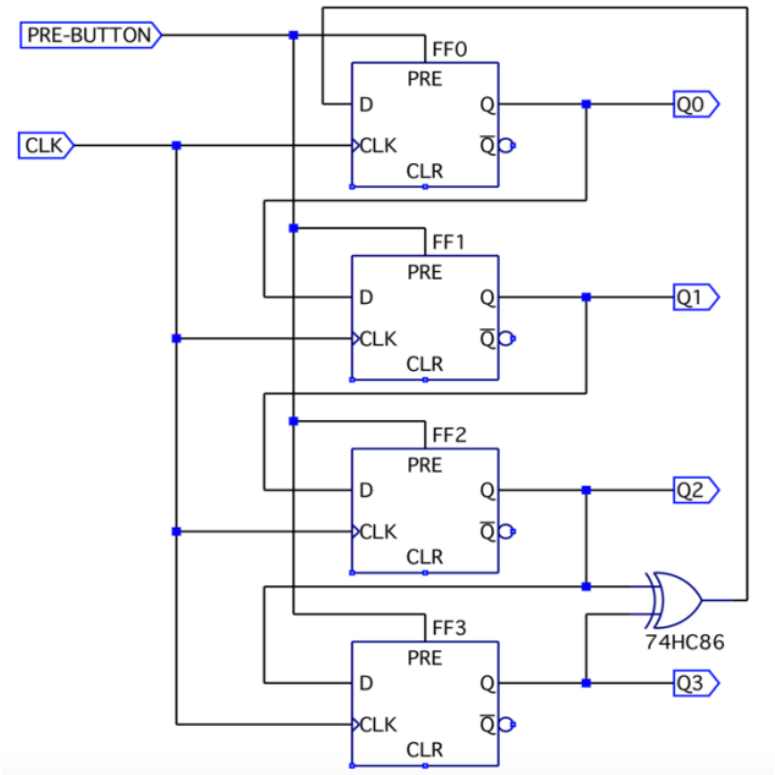

Figura 8

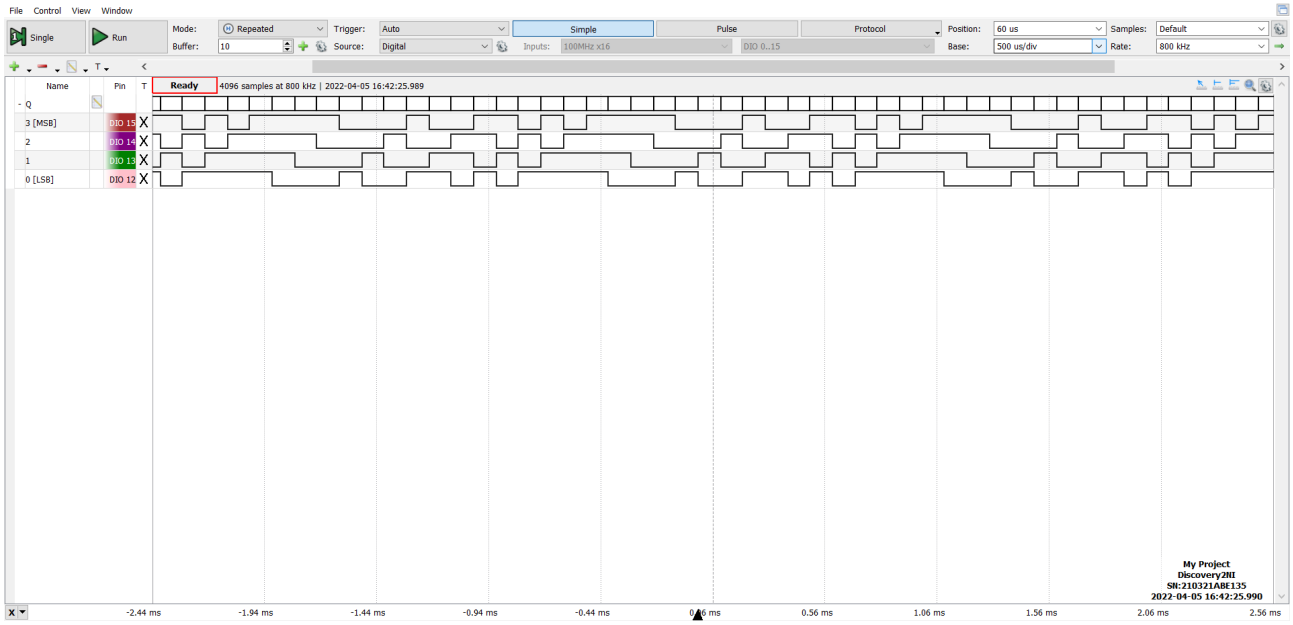

Figura 9: Acquisizione temporale con Logic del bus in uscita dal generatore di sequenze psudo-casuale descritto in fig. 8

4.c Studio delle sequenze generabili con diverse condizioni iniziali

Si provano quindi altre combinazioni di TAP, per verificare che la scelta di utilizzare l'uscita Q_2 e Q_3 produce una sequenza più lunga rispetto a qualunque altra configurazione

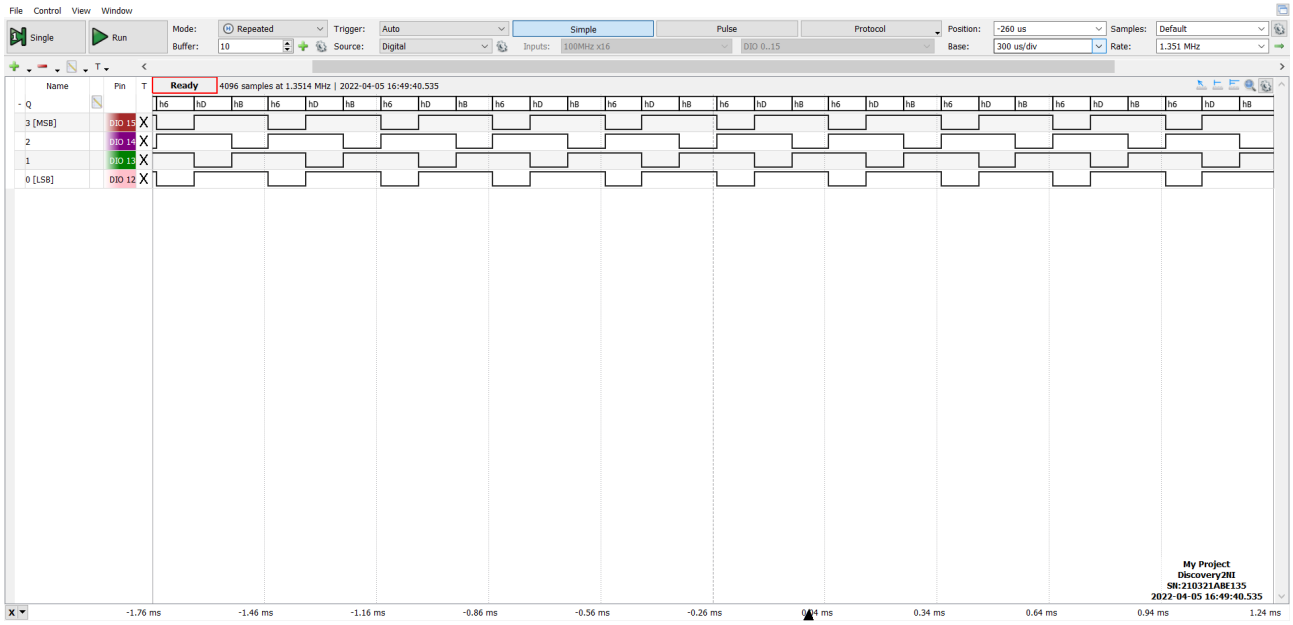

Figura 10: Acquisizione temporale con Logic del bus in uscita dal generatore di sequenze psudo-casuale con TAP sulle uscite Q_0 e Q_1, la sequenza si ripete ogni 4 eventi

5 Divisori di frequenza con contatori binari

5.a Costruzione del circuito

Si intende costruire un divisore di frequenza a partire da un contatore binario a 4 bit, utilizzando l'integrato SN74LS163, presente in fig. 13. Si vuole innanzitutto verificarne il funzionamento.

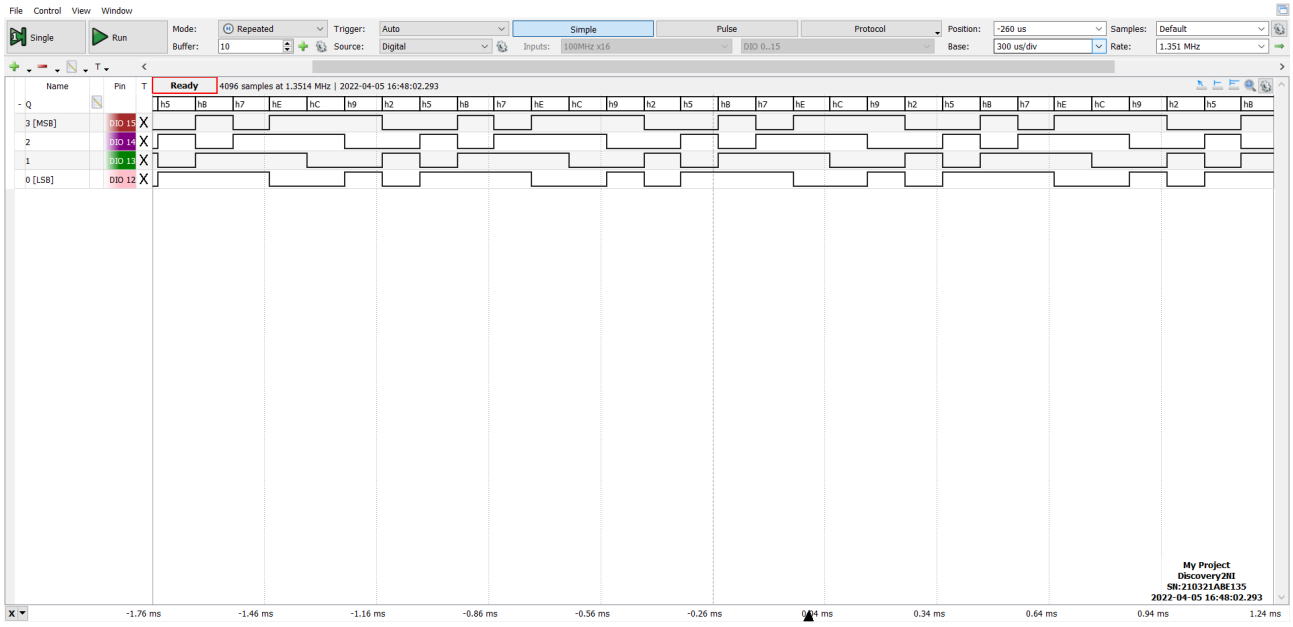

Figura 11: Acquisizione temporale con Logic del bus in uscita dal generatore di sequenze psudo-casuale con TAP sulle uscite Q_2 e Q_1, la sequenza si ripete ogni 8 eventi

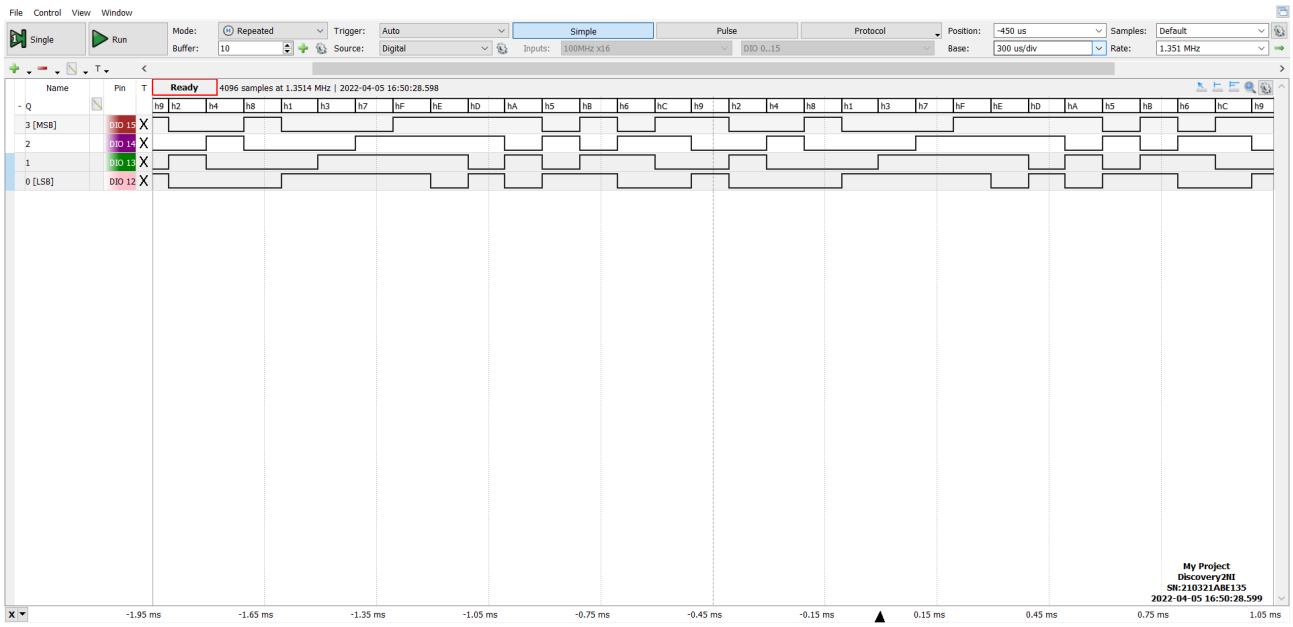

Figura 12: Acquisizione temporale con Logic del bus in uscita dal generatore di sequenze psudo-casuale con TAP sulle uscite Q_0 e Q_3, la sequenza si ripete ogni 16 eventi

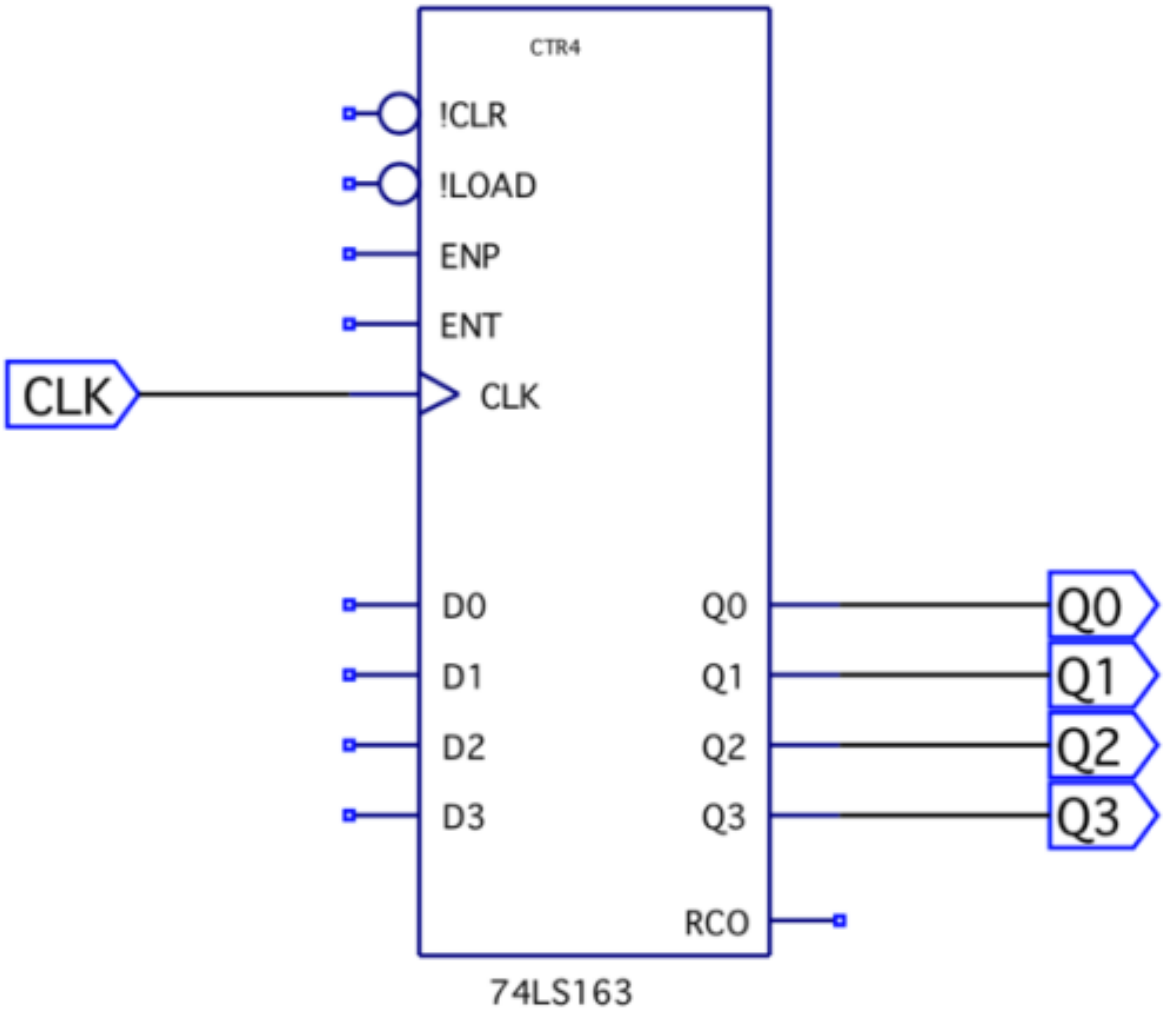
CTR4
!CLR
!LOAD
ENP
ENT
CLK
CLK
D0
D1
D2
D3
Q0
Q1
Q2
Q3
RCO
74LS163
Q0
Q1
Q2
Q3

Figura 13

5.b Verifica del ciclo di funzionamento dei contatori

Dopo aver montato il circuito utilizziamo la funzione Pattern di Waveform per inviare un segnale di clock di frequenza pari a 10 kHz al pin di clock (CLK) del contatore, e utilizzeremo Logic per acquisire le uscite Q_0->Q_3. Essendo un contatore a 4 bit, ci si aspetta che il Bus conti dallo stato 0000 (0 in decimale) fino allo stato 1111 (15 in decimale). Per questo motivo utilizzeremo un formato di Bus esadecimale per verificare che il circuito generi tutti gli stati possibili (0->F). Dalla fig. 14 si può notare che il comportamento del circuito è come atteso,

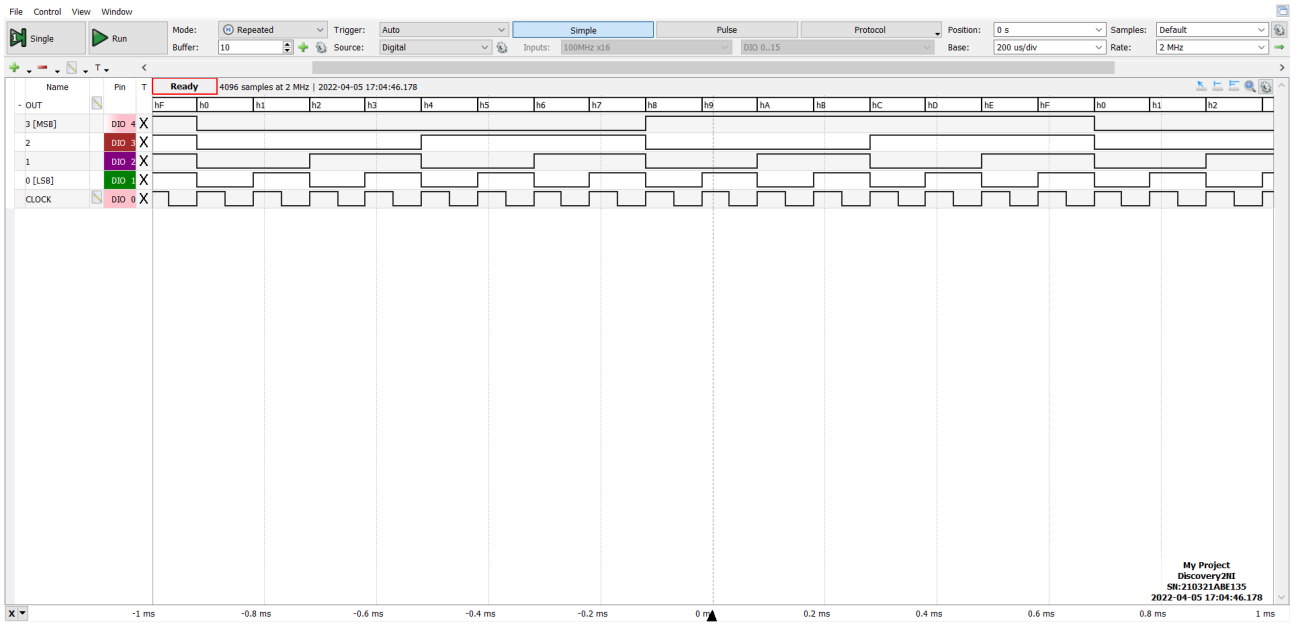

Figura 14: Acquisizione temporale con Logic del bus in uscita dal contatore, con frequenza di clock pari a 10 kHz

e che gli stati del bus in uscita comprendono tutti i valori in ordine crescente della numerazione esadecimale

5.c Verifica della divisione in frequenza

Dato che il contatore incrementa di 1 ad ogni evento di clock, il bit di ordine n avrà come frequenza la metà di quella del bit di ordine n-1. Avendo noi un BUS composto da 4 bit, ci si aspetta che le frequenze ottenute siano $\frac{f_clock}{2}$ (per il LSB), $\frac{f_clock}{4}$, $\frac{f_clock}{8}$ e $\frac{f_clock}{16}$ (per il MSB). Sempre facendo riferimento all fig. 14 si può notare come le uscite del contatore si comportino da divisori in frequenza: infatti i periodi che si trovano analizzando l'acquisizione sono 2 (LSB), 4, 8 e 16 (MSB) volte il periodo del clock.

5.d Transizione sincrona del contatore

Utilizziamo quindi la funzione di trigger di Logic e la impostiamo in modo tale che l'acquisizione cominci quando il segnale presente in Q_3 cambia da 1 a 0 in modo da poter studiare gli eventuali ritardi delle singole uscite nella transizione F->0 e verificare il comportamento sincrono del contatore. Dalla fig. 15 possiamo infatti vedere che la transizione è sincrona e il passaggio di stato 1->0 avviene contemporaneamente in ogni uscita.

5.e Costruzione di un divisore di frequenza 1/10

Si vuole quindi realizzare un divisore in frequenza, che generi un segnale di periodo pari a 10 volte quello di clock: per farlo utilizzeremo il pin CLEAR del contatore, il quale quando viene inviato un segnale Low resetta il contatore allo stato 0000. È quindi necessario che il contatore conti un totale di 10 stati e poi si resetti, per cui partendo dallo 0, dobbiamo imporre la condizione che arrivati allo stato 9, il contatore riceva l'impulso di Clear (che ricordiamo essere Active Low). Per farlo utilizzeremo una porta NAND (utilizzando l'integrato SN74LS00) ai cui 2 ingressi invieremo il LSB e il MSB e invieremo l'uscita al pin di Clear. In questo modo, quando il contatore arriva a 9 (1001) il pin di Clear riceverà un segnale Low e il circuito si azzererà; pertanto la frequenza del segnale Clear sarà un decimo di quella di clock Inoltre essendo il segnale di Clear sullo stato Low solamente quando il bus in uscita registra lo stato 1001, mi aspetterò un Duty Cycle pari al 90 % Si costruisce quindi il circuito presente in **??**. Come prima, si utilizza la funzione Logic per acquisire il Bus dei 4 bit in uscita, il clock e il bit di Clear. Osservando l'acquisizione presente in fig. 16, possiamo concludere come da aspettative che il segnale presente su clear ha una frequenza pari a $\frac{f_{clock}}{10}$ e Duty Cycle pari al 90 %.

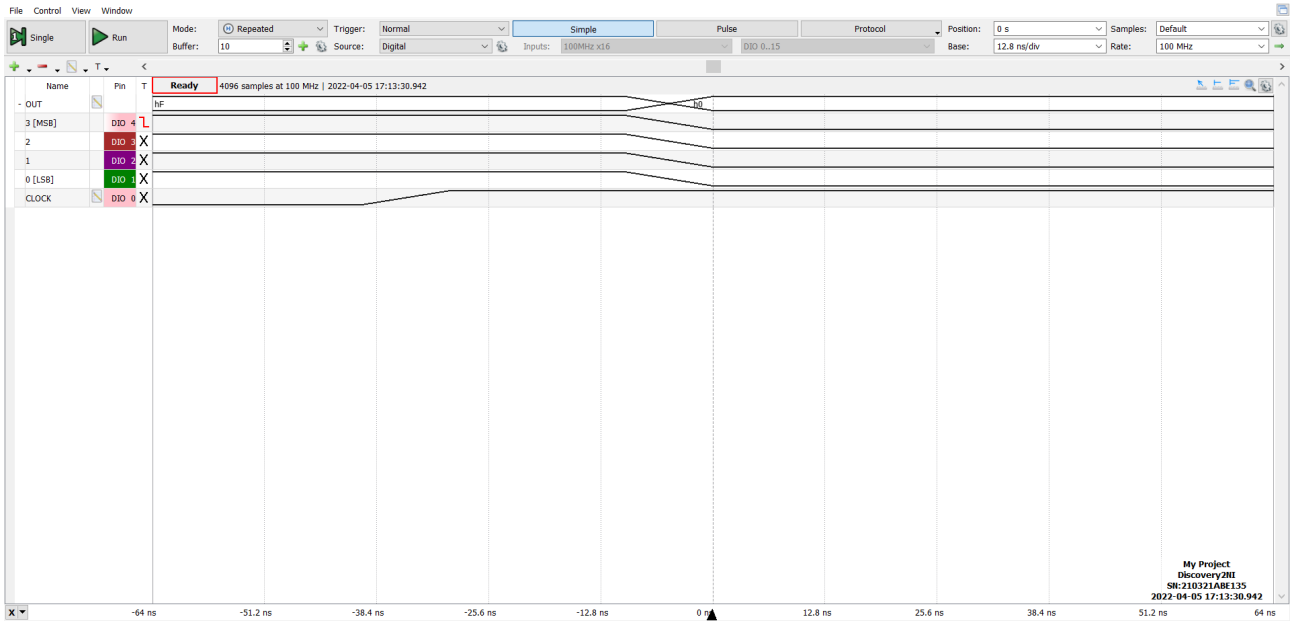

Figura 15: Acquisizione temporale con Logic del bus in uscita dal contatore durante la transizione 15->0; dall'immagine possiamo notare il comportamento sincrono della commutazione delle uscite del contatore

5.f Divisore di frequenza programmabile con RCO

Per concludere, si vuole costruire un divisore in frequenza programmabile: per questo scopo utilizzeremo il bit RCO (Ripple Carry Output), la cui funzione è di generare un segnale H solo nel caso in cui il contatore abbia raggiunto lo stato massimo (1111) e il bit Load, che invece quando vale 0 sovrascrive lo stato del contatore in quello presente all'interno del bus di input.

5.g Misura dei tempi caratteristici del divisore RCO

5.h Analisi e verifica del comportamento del divisore RCO

6 Sintetizzatore musicale

Conclusioni e commenti finali

Si è riusciti a verificare il corretto funzionamento di circuiti logici sequenziali di crescente complessità e svariate applicazioni (e.g. crittografia, sistemi di controllo e misura) costruiti con porte NAND, XOR, D-Latch e contatori sincroni. In particolare sono stati realizzati e studiati un D-Latch, uno shift-register con positive edge-triggered D-FF, un generatore di sequenze pseudocasuali e alcuni tipi di divisore di frequenza con contatori binari. Inoltre si è riusciti ad apprezzare l'effetto dei tempi di propagazione delle porte sul loro comportamento, seppur in maniera limitata dalla bassa risoluzione temporale dell'AD2.

Dichiarazione

I firmatari di questa relazione dichiarano che il contenuto della relazione è originale, con misure effettuate dai membri del gruppo, e che tutti i firmatari hanno contribuito alla elaborazione della relazione stessa.

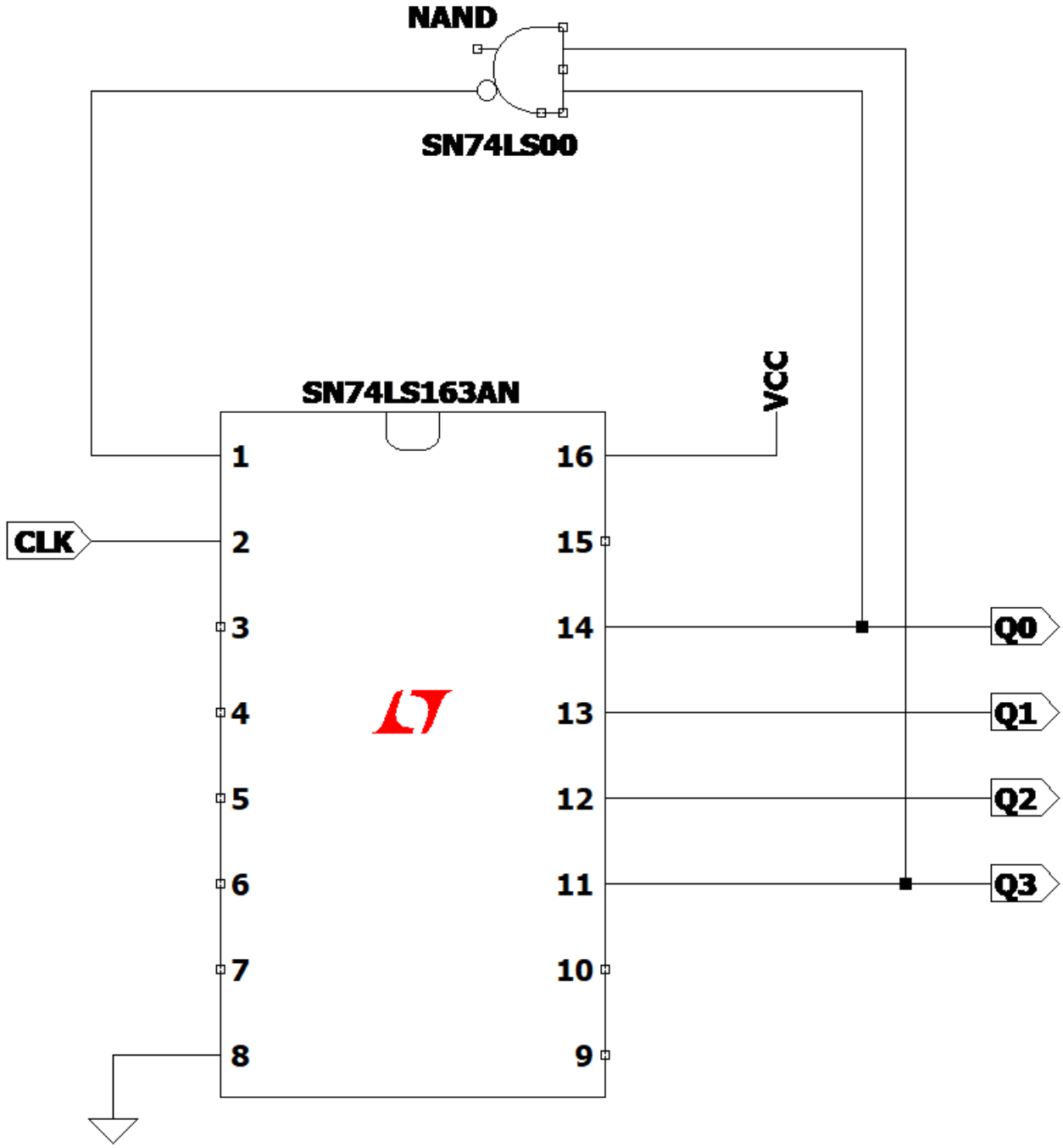

Figura 16: Schematica utilizzata per il divisore per 10 di frequenza

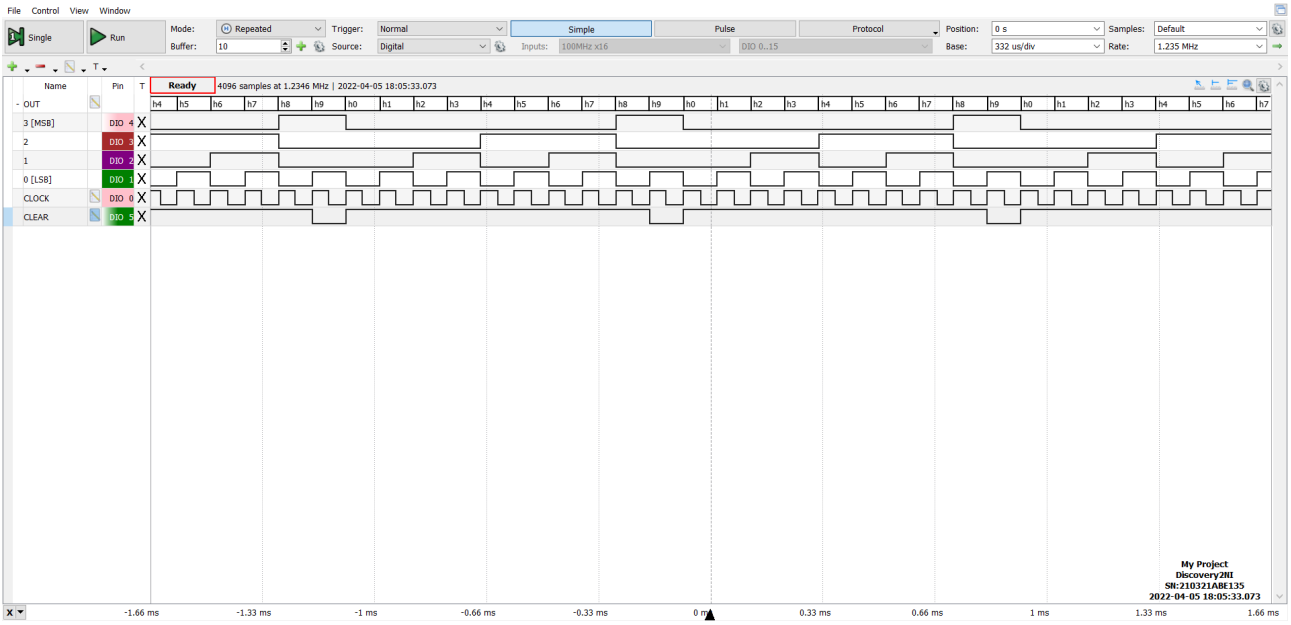

Figura 17: Acquisizione temporale con Logic del bus in uscita dal circuito che conta fino a 10; si può notare che il segnale in entrata nel pin Clear ha effettivamente periodo pari a 10 volte quello del clock

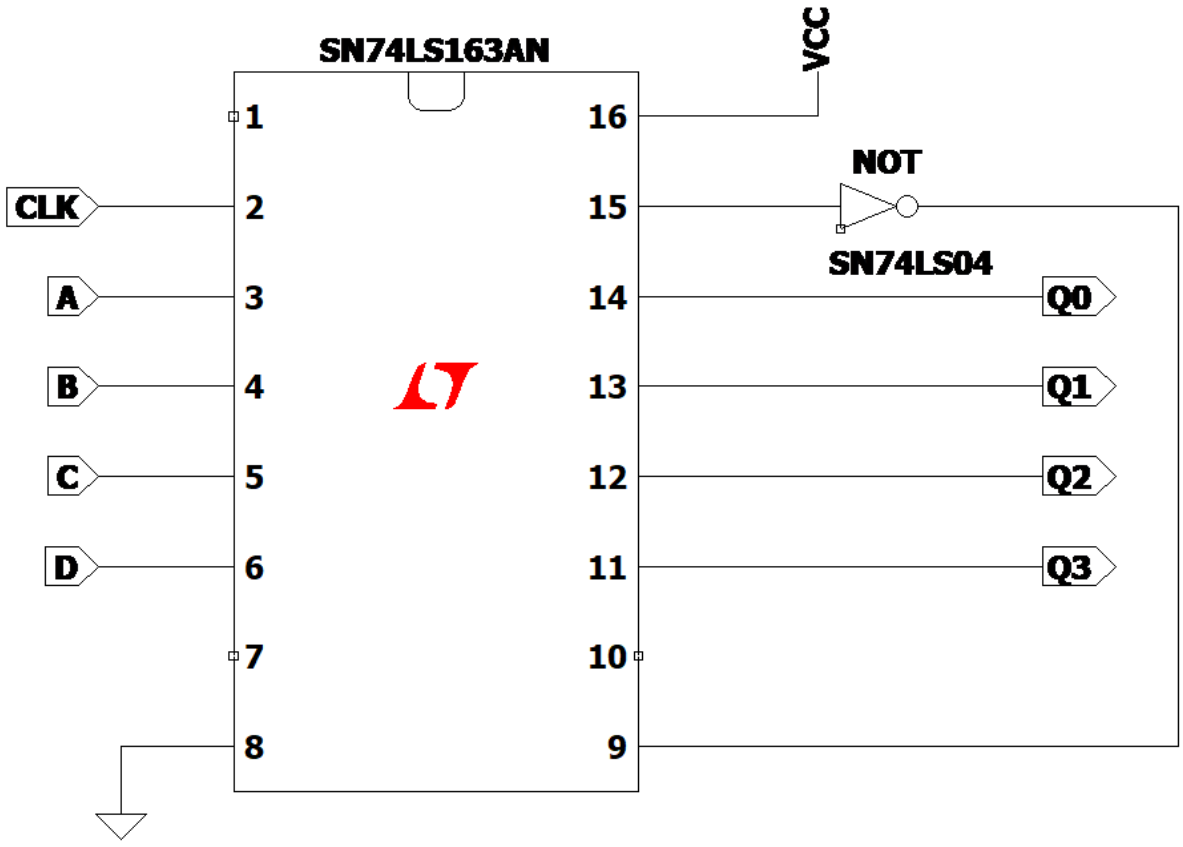

Figura 18: Schematica utilizzata per il divisore di frequenza programmabile

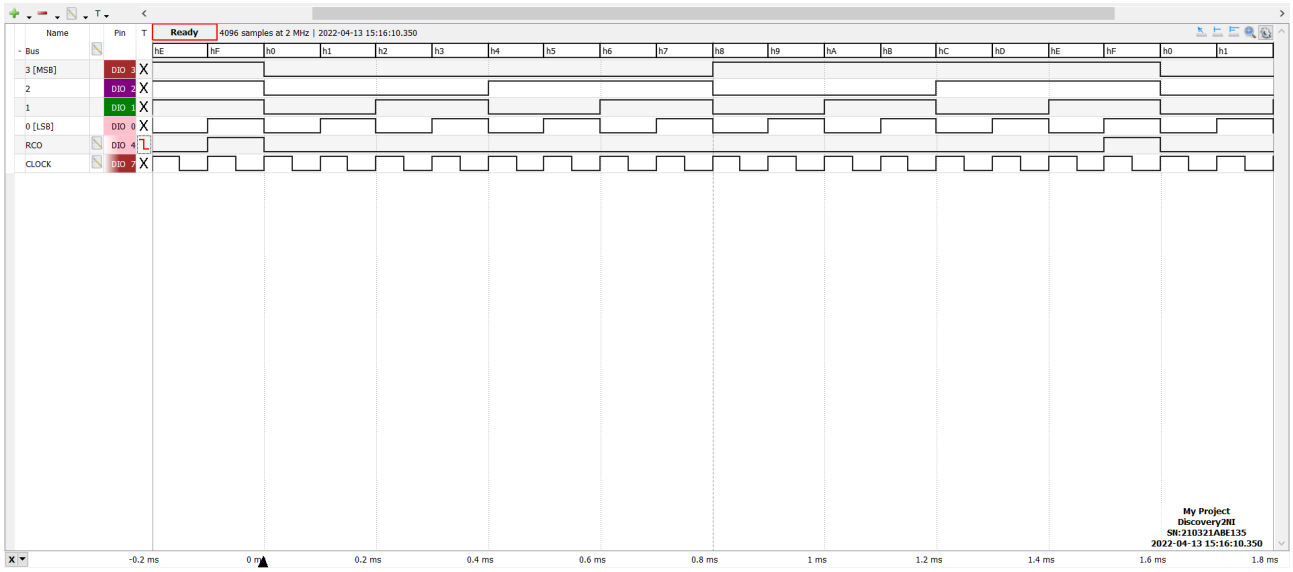

Figura 19: Acquisizione con Logic dei segnali del contatore per verificare il funzionamento del bit RCO

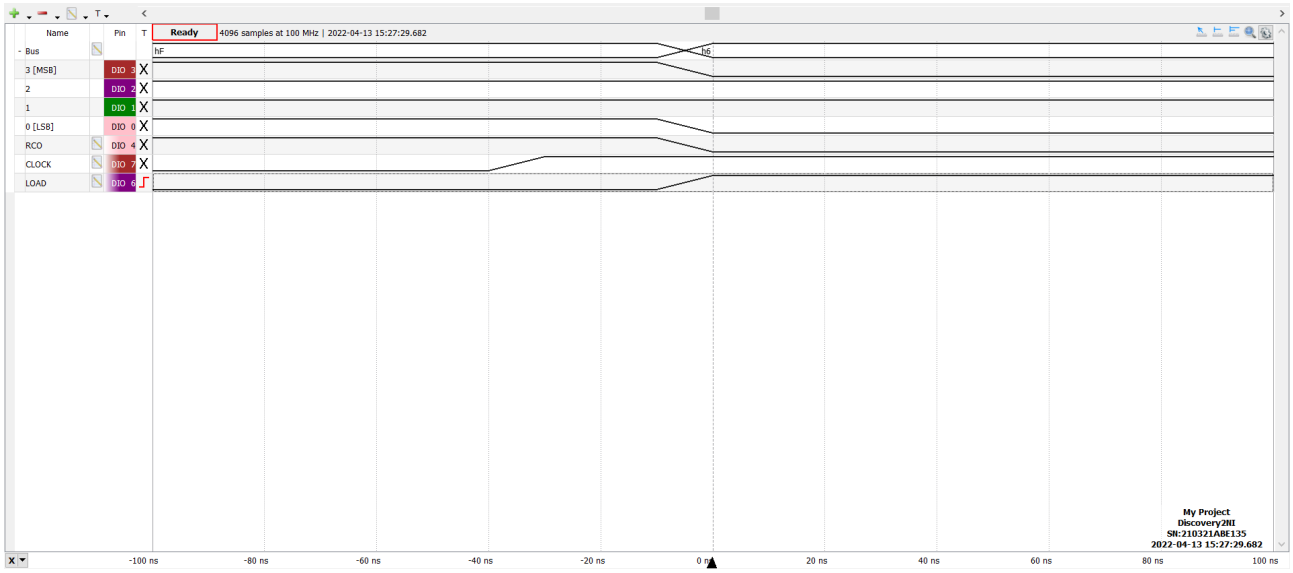

Figura 20: Acquisizione con logic dei segnali all'interno del circuito divisore di frequenza programmabile per un fondo scala dei tempi molto ristretto

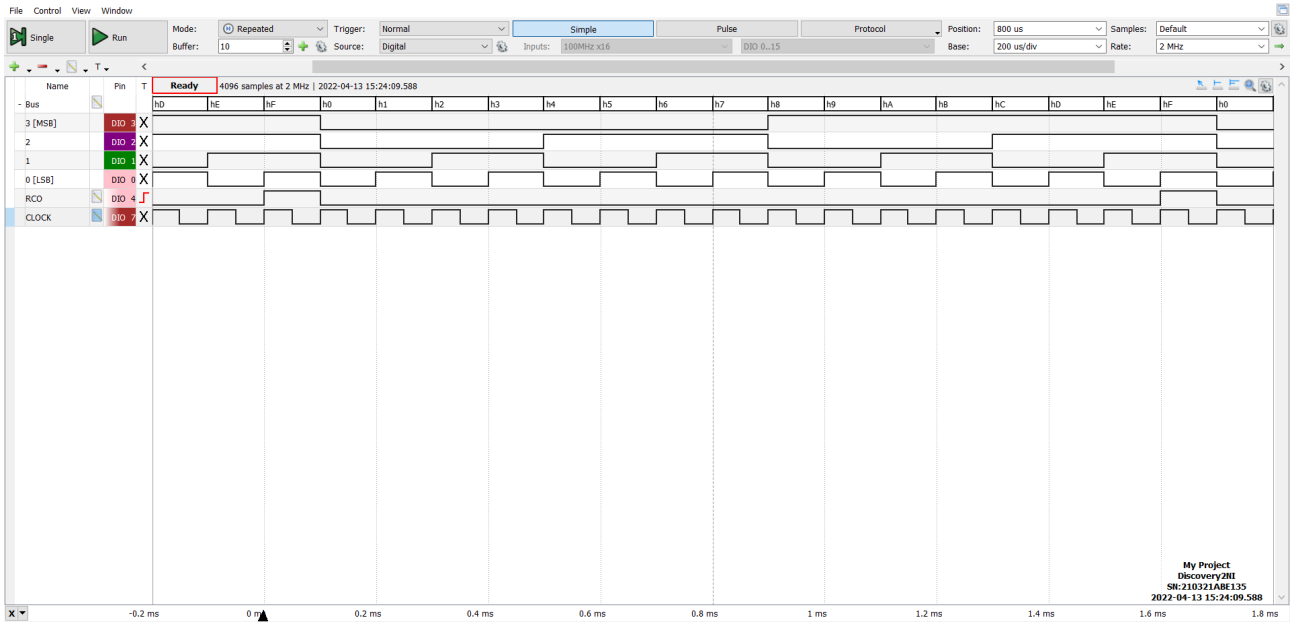

Figura 21: Acquisizione con Logic dei segnali all'interno del circuito divisore di frequenza programmabile, con sequenza di avvio 0000

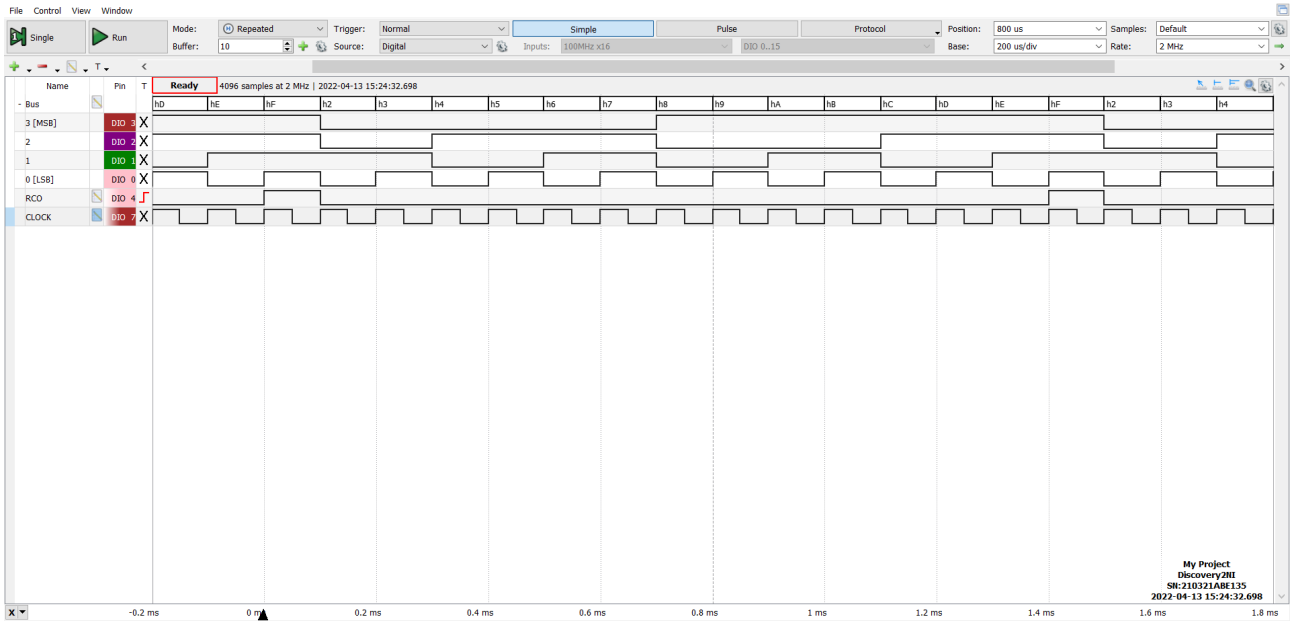

Figura 22: Acquisizione con Logic dei segnali all'interno del circuito divisore di frequenza programmabile, con sequenza di avvio 0010

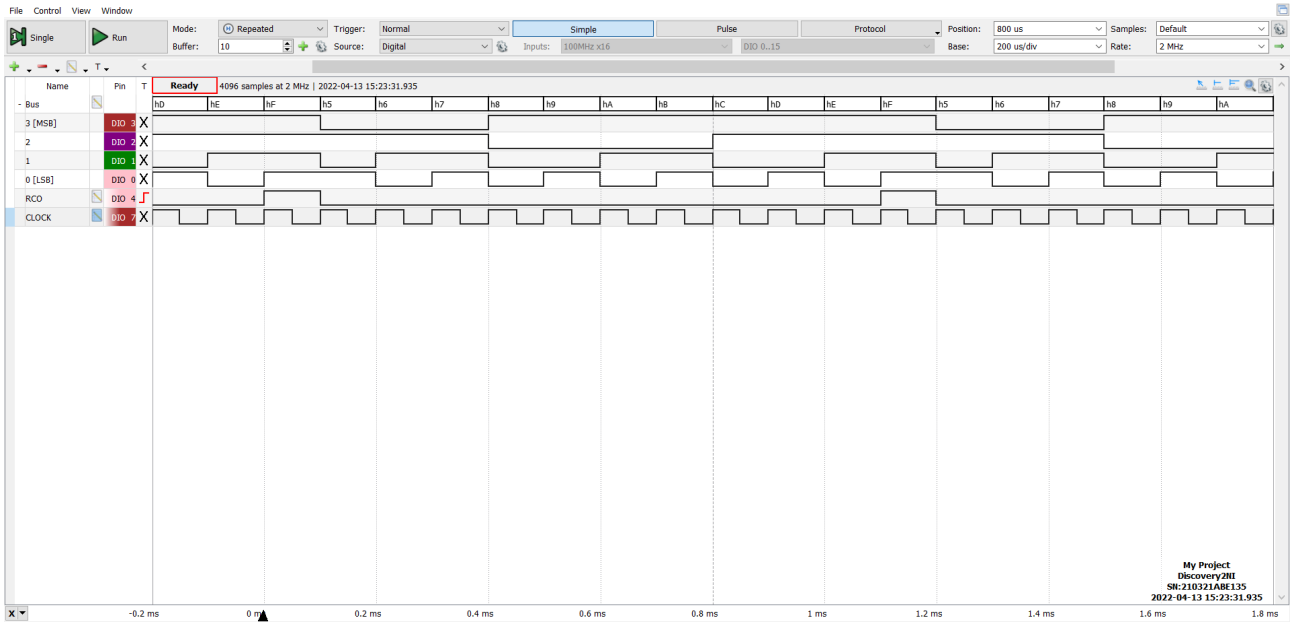

Figura 23: Acquisizione con Logic dei segnali all'interno del circuito divisore di frequenza programmabile, con sequenza di avvio 0101

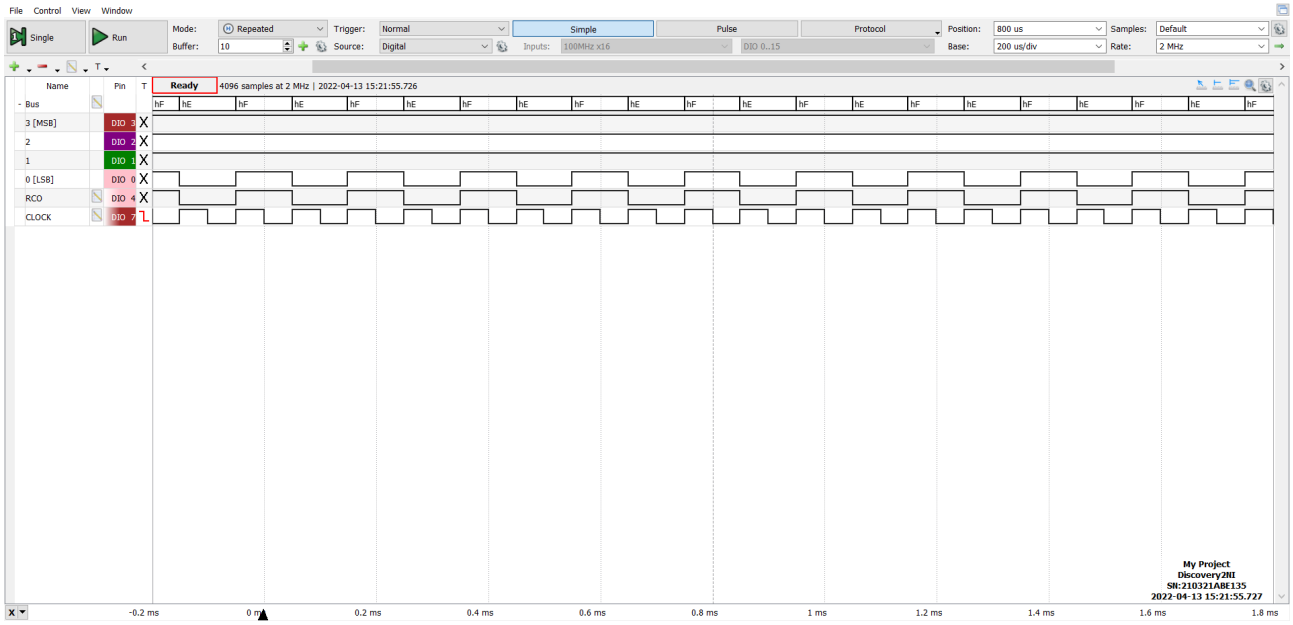

Figura 24: Acquisizione con Logic dei segnali all'interno del circuito divisore di frequenza programmabile, con sequenza di avvio 1110

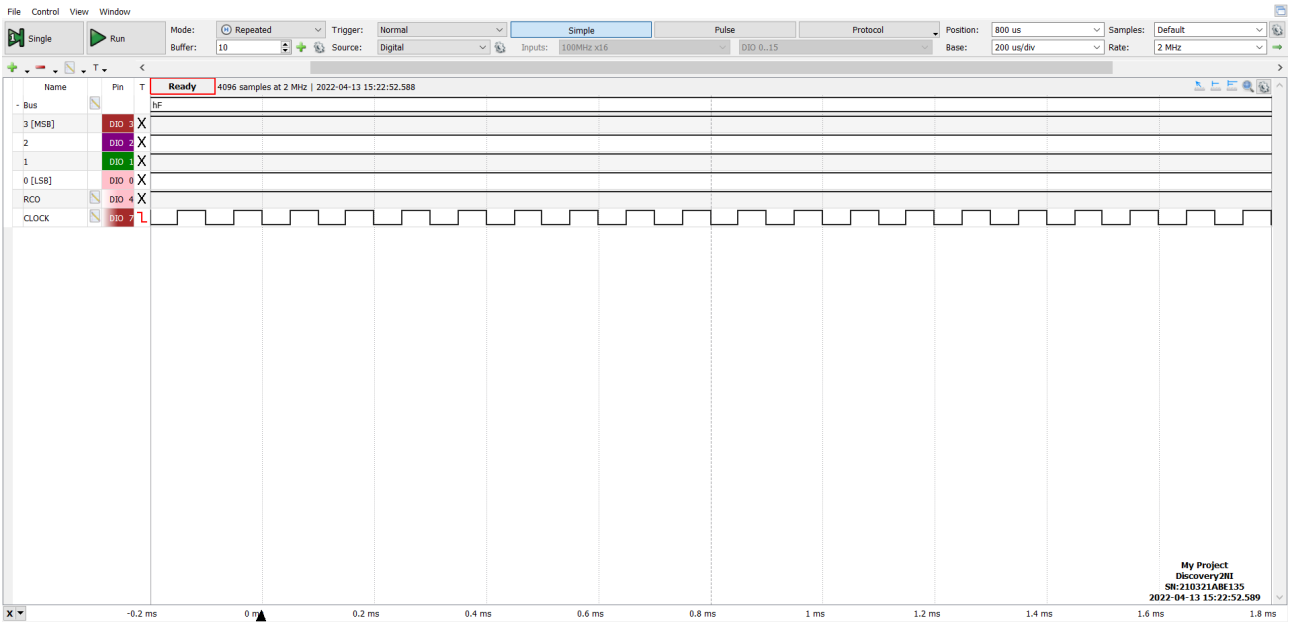

Figura 25: Acquisizione con Logic dei segnali all'interno del circuito divisore di frequenza programmabile, con sequenza di avvio 1111