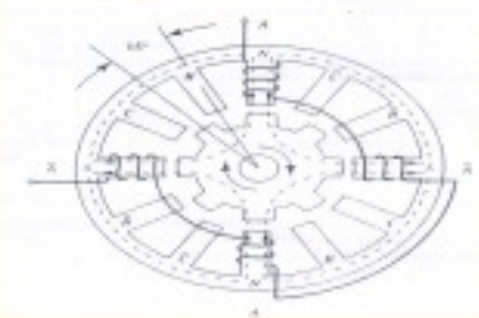
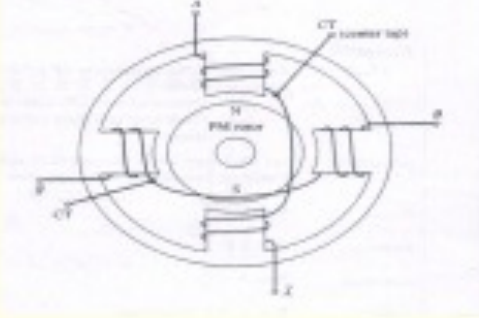
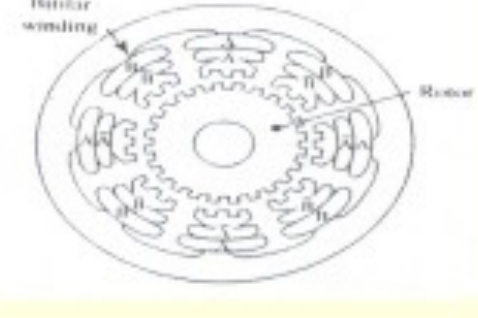


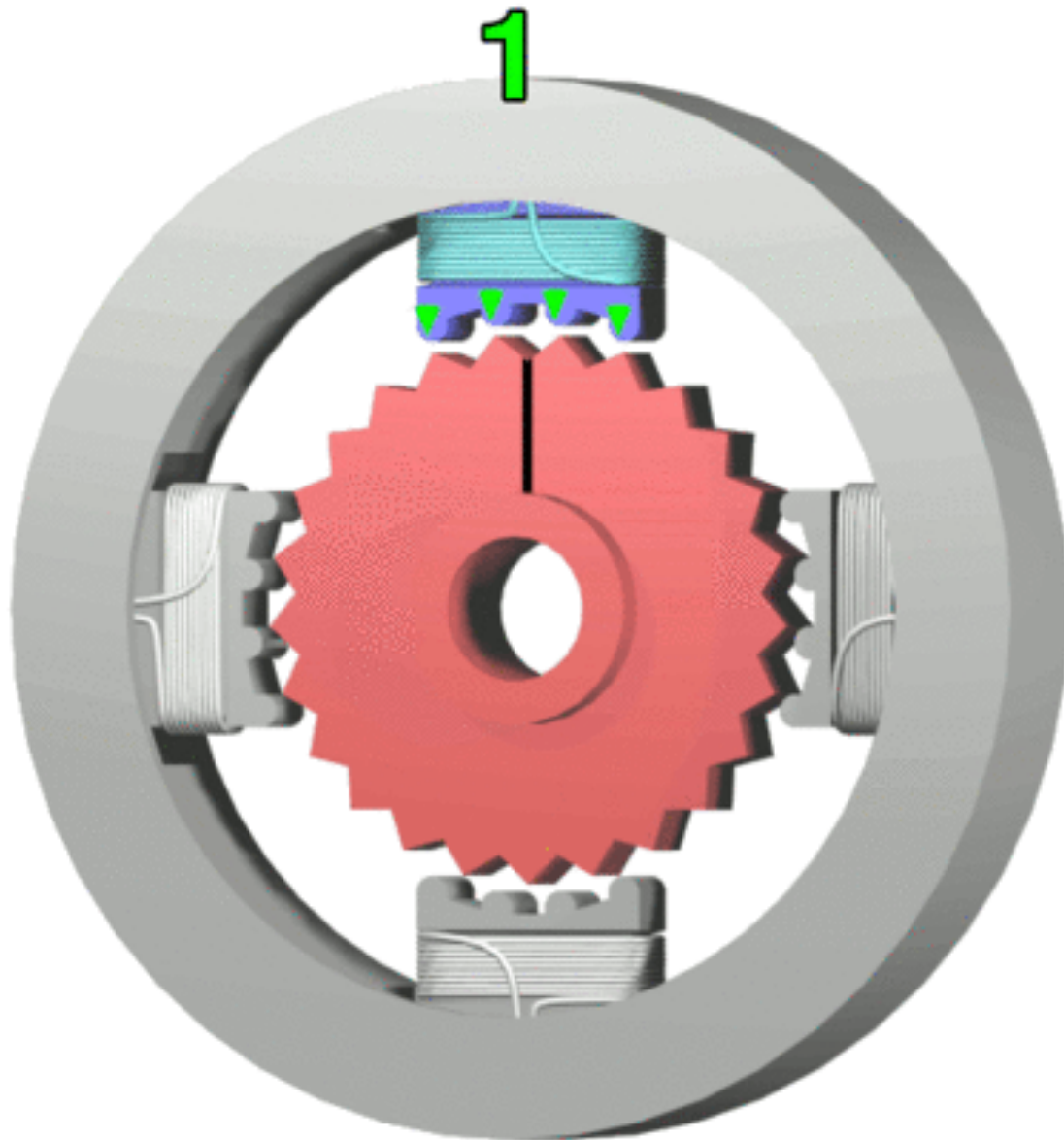
STEPPER MOTOR

Progetto di Logiche Riconfigurabili, a cura di:
Mattia Bisacchi, Alessandro Di Cesare, Filippo Franzoni, Fabio Tosi

Tipologia di motore

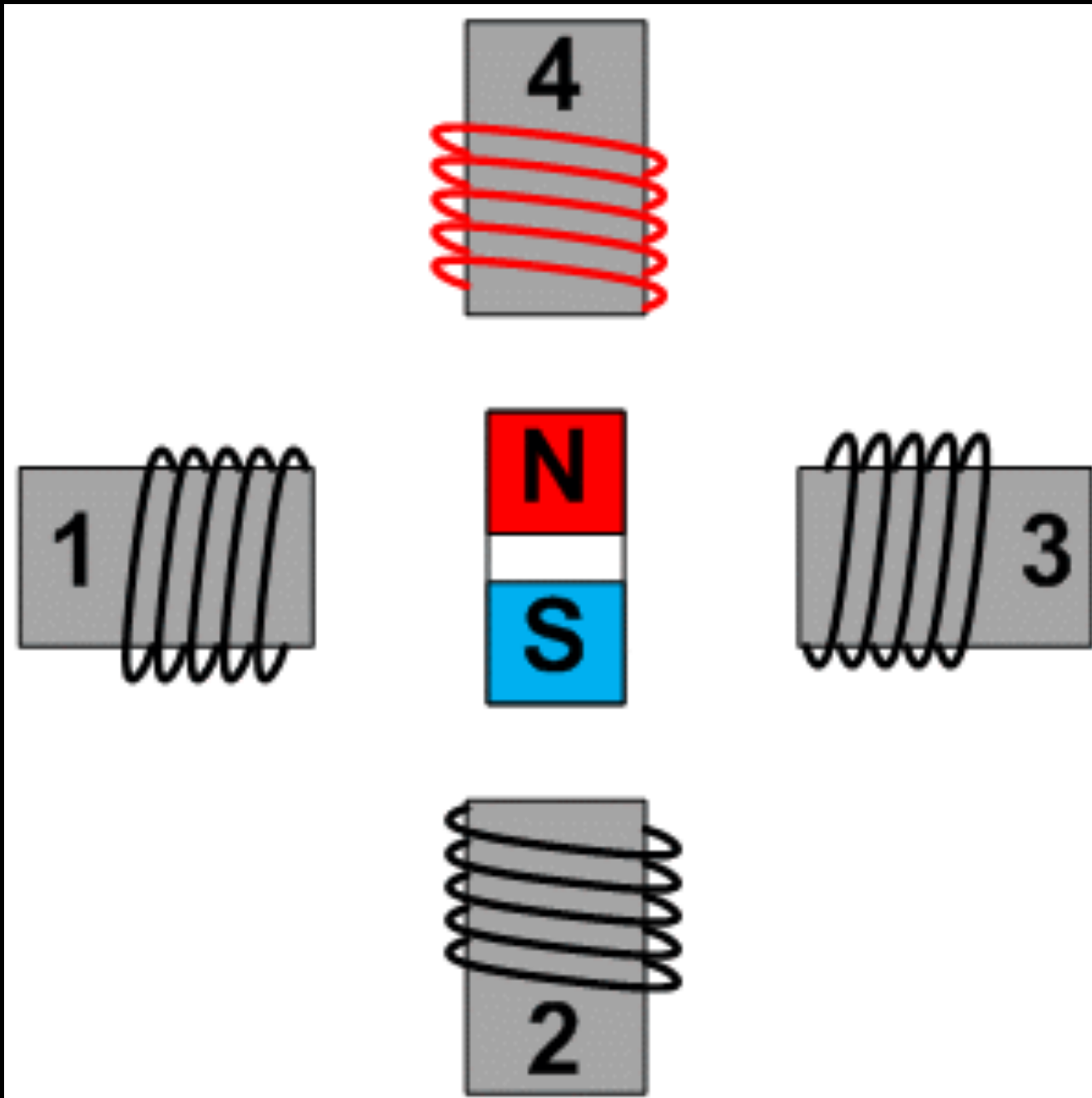
Variable reluctance	Permanent magnet	Hybrid
		
<ol style="list-style-type: none"> 1) Soft iron multipole rotor and a laminated core in the wound stator 2) Has four "stator pole sets" (A, B, C,) set 15 degrees apart 3) Rarely use in industry because of less detent torque. 	<ol style="list-style-type: none"> 1) Rotor has no teeth and a laminated core in the wound stator 2) Has four phase and 90 degrees apart. 3) Ideal choice for non industrial application such as a line printer print wheel positioner and operate at fairly low speed. 	<ol style="list-style-type: none"> 1) Standard Hybrid motor has 200 rotor teeth and bifilar stator windings. 2) Standard Hybrid motor move at 1.8 step angles. Other Hybrid motor available in 0.9° and 3.6° step angle configurations. 3) Wide variety used for industrial applications because of high static and dynamic torque and run at very high step rates.

Modalità: FULL STEP



- Ogni spostamento ha ampiezza pari all'angolo dello step indicato dalle specifiche del motore (1.8° nel nostro caso)
- Si prevede l'alimentazione di **due fasi** contemporaneamente per ottenere il massimo momento torcente

Modalità: HALF STEP

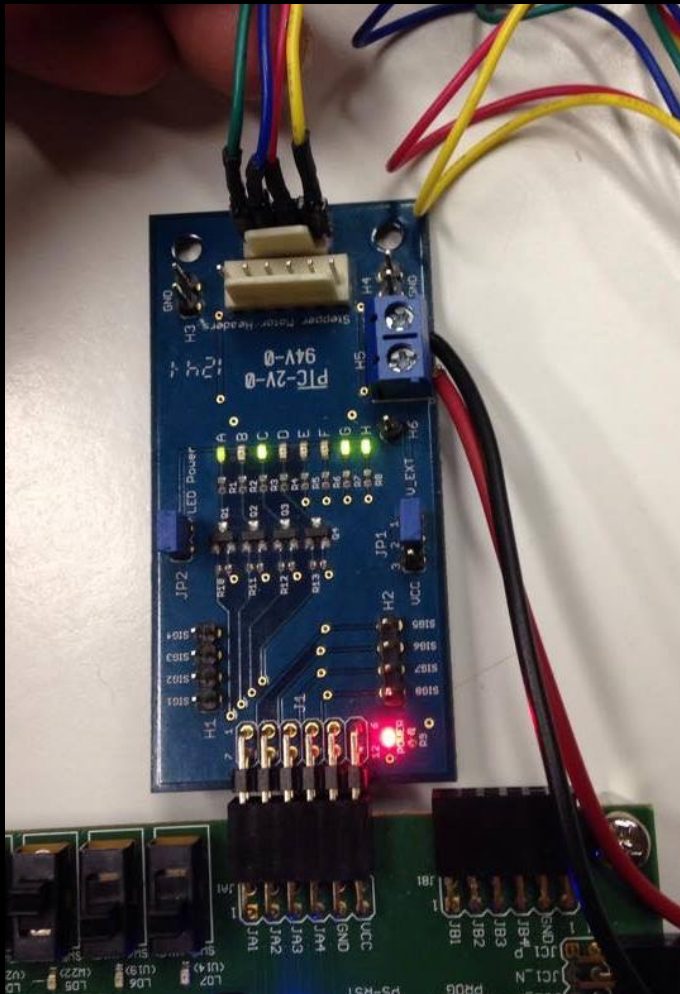


- Ogni spostamento ha ampiezza pari alla **metà** dell'angolo dello step indicato dalle specifiche del motore (0.9° nel nostro caso)
- La sequenza alterna l'alimentazione di **una e due fasi** contemporaneamente durante lo spostamento

Microstepping

- Sviluppato per permettere allo stepper motor di avere un andamento più **fluid** nel passaggio da uno step al successivo
- Non aumenta la **risoluzione reale** del motore
- Utilizzato per aumentare la **risoluzione teorica** durante lo spostamento, minimizzando il **rumore** e le **vibrazioni** prodotte
- **Molto impreciso**: non è possibile mantenere una posizione intermedia perché la dipendenza di questa non è linearmente proporzionale alla corrente ma dipende dalle caratteristiche elettriche e meccaniche del motore
- Solo quando la sequenza di passi coincide con la modalità full/half step, la posizione del rotore è **deterministica**
- Il funzionamento richiede una **modulazione** della potenza mediante PWM, causando un minor momento torcente rispetto alle modalità full/half step

Sequenze



- 1a -> A
- 1b -> B
- 2a -> A'
- 2b -> B'

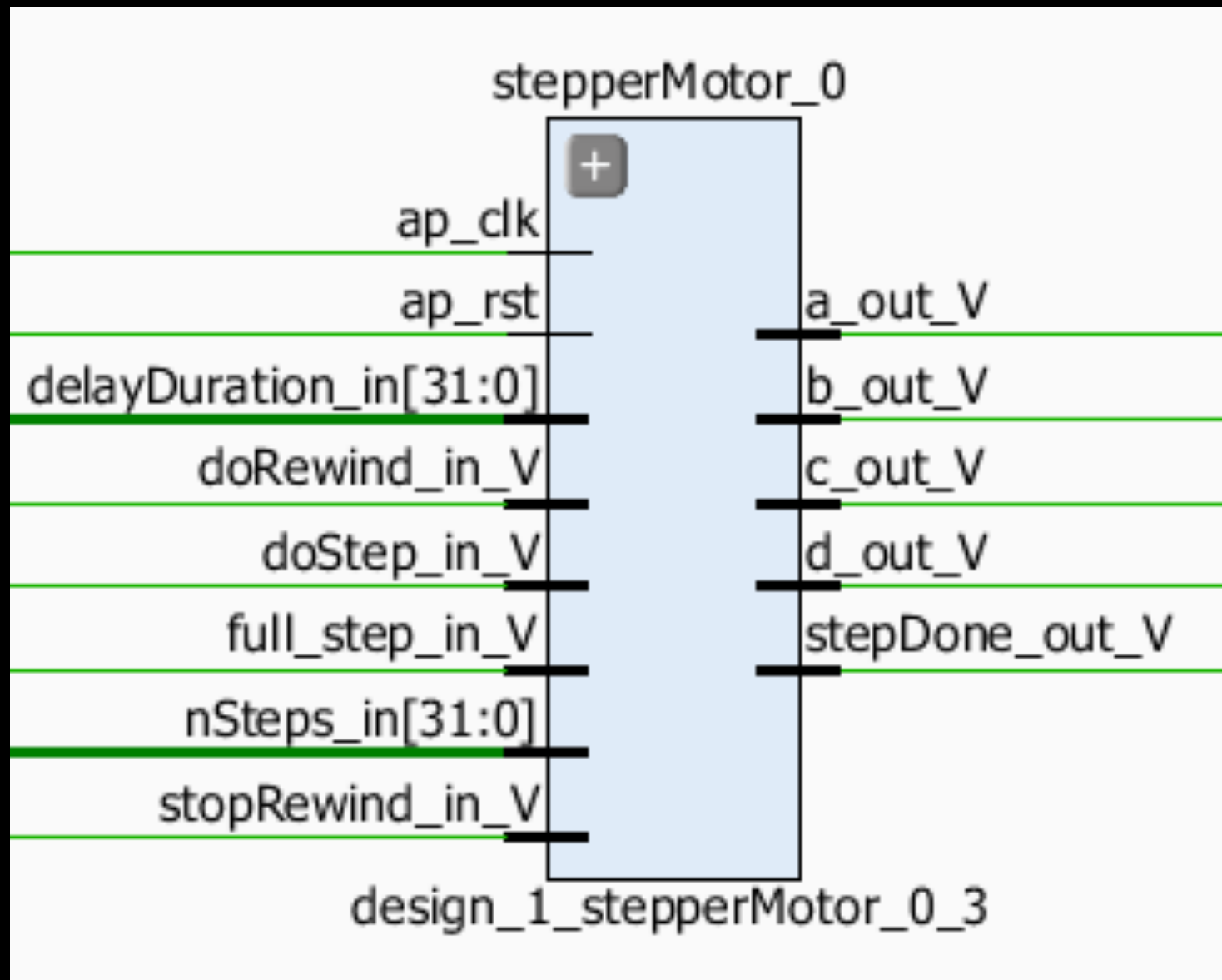
Full step

Index	1a	1b	2a	2b
1	1	0	0	1
2	1	1	0	0
3	0	1	1	0
4	0	0	1	1

Half step

Index	1a	1b	2a	2b
1	1	0	0	0
2	1	1	0	0
3	0	1	0	0
4	0	1	1	0
5	0	0	1	0
6	0	0	1	1
7	0	0	0	1
8	1	0	0	1

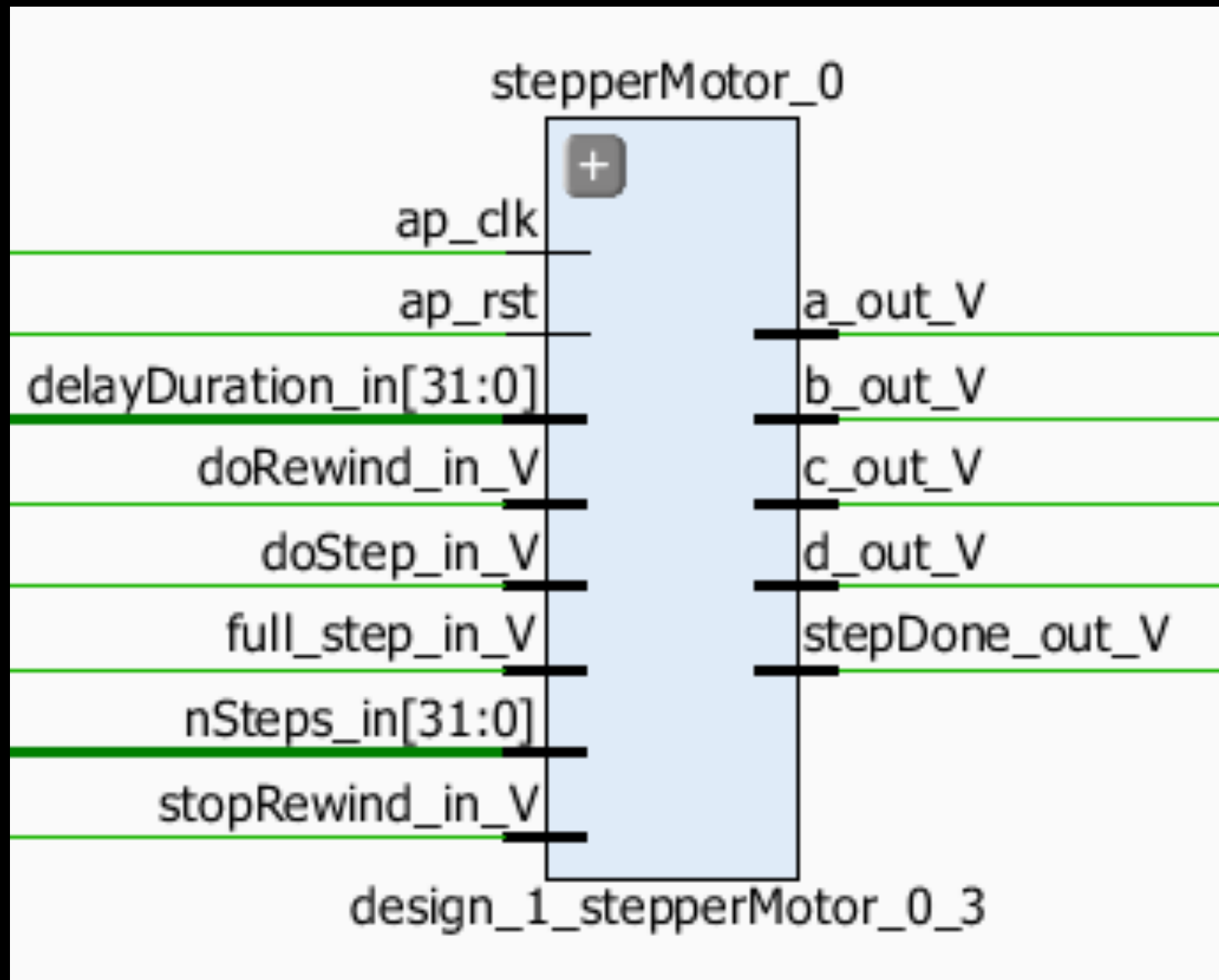
Schematico (Input)



- **delayDuration_in**: numero di clock di attesa dopo l'esecuzione di uno step (*)
- **doRewind_in**: comando inviato dal sensore di fine corsa
- **stopRewind_in**: comando inviato dal sensore di inizio corsa
- **doStep_in**: comando di movimento. Provoca uno spostamento di nSteps
- **nSteps_in**: numero di step da eseguire in uno spostamento in avanti
- **full_step_in**: modalità di funzionamento del motore (1: Full step, 0: Half step) (*)

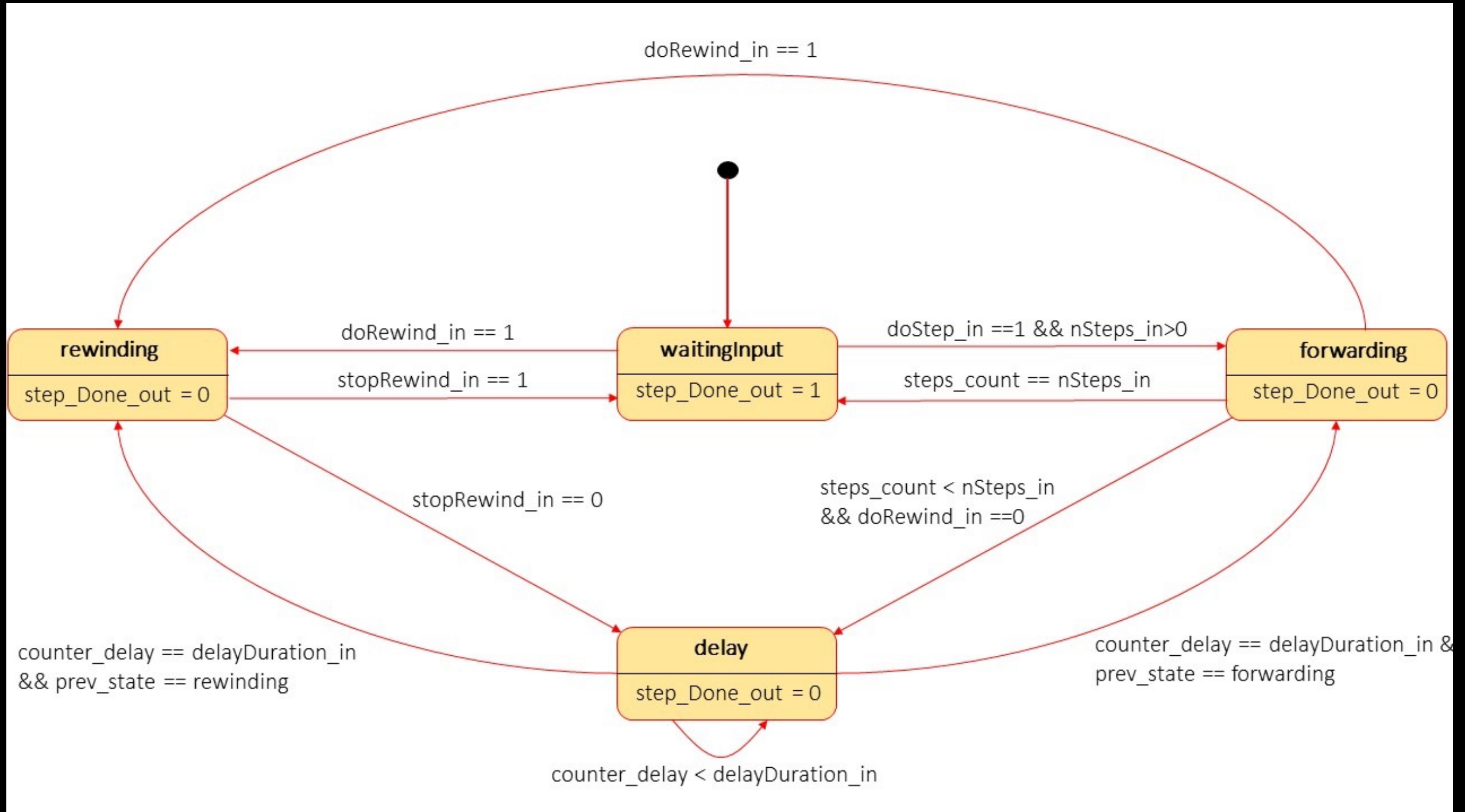
(*) segnale che rispetta il protocollo AP_STABLE: può essere modificato in modo sicuro solo quando ap_rst è asserito.

Schematic (Output)



- **a_out ... d_out:** segnali di comandi del motore secondo la corrispondenza della tabella precedente
- **stepDone_out:** a 0 quando il motore è in movimento, ad 1 quando il modulo è in attesa di comandi

Automa



L'automa è stato realizzato secondo il modello **Moore** ed ogni transizione ha effetto al ciclo di clock **successivo**.

Report sintesi

[-] Latency (clock cycles)

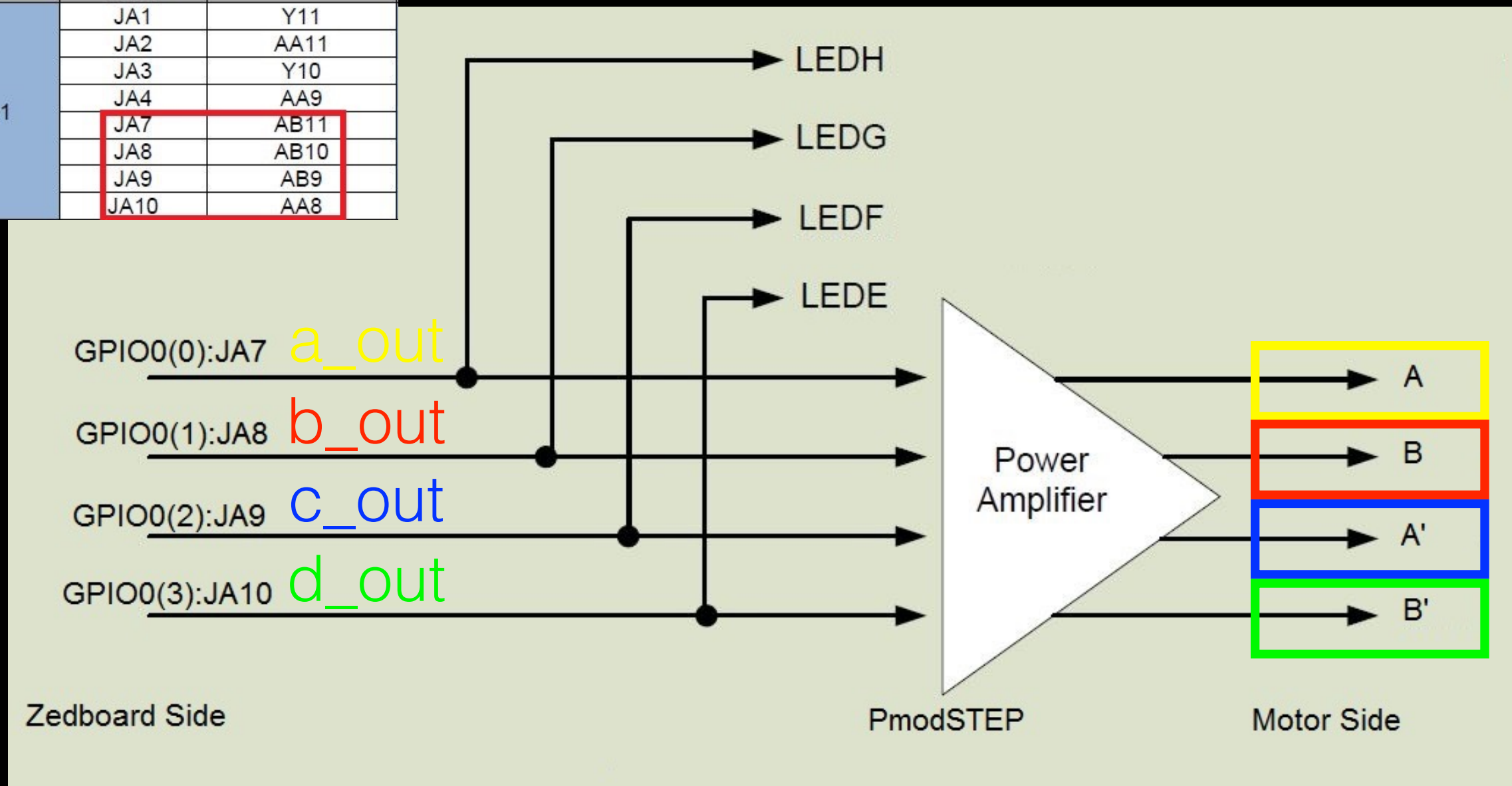
[-] Summary

Latency		Interval		
min	max	min	max	Type
0	0	1	1	none

L'organizzazione del codice e l'inserimento di opportune pragma hanno permesso di ottenere una **latenza nulla**, che consente al modulo di intercettare gli input ad **ogni clock**.

Collegamento PMOD-motore

Pmod	Signal Name	Zynq pin
JA1	JA1	Y11
	JA2	AA11
	JA3	Y10
	JA4	AA9
	JA7	AB11
	JA8	AB10
	JA9	AB9
	JA10	AA8

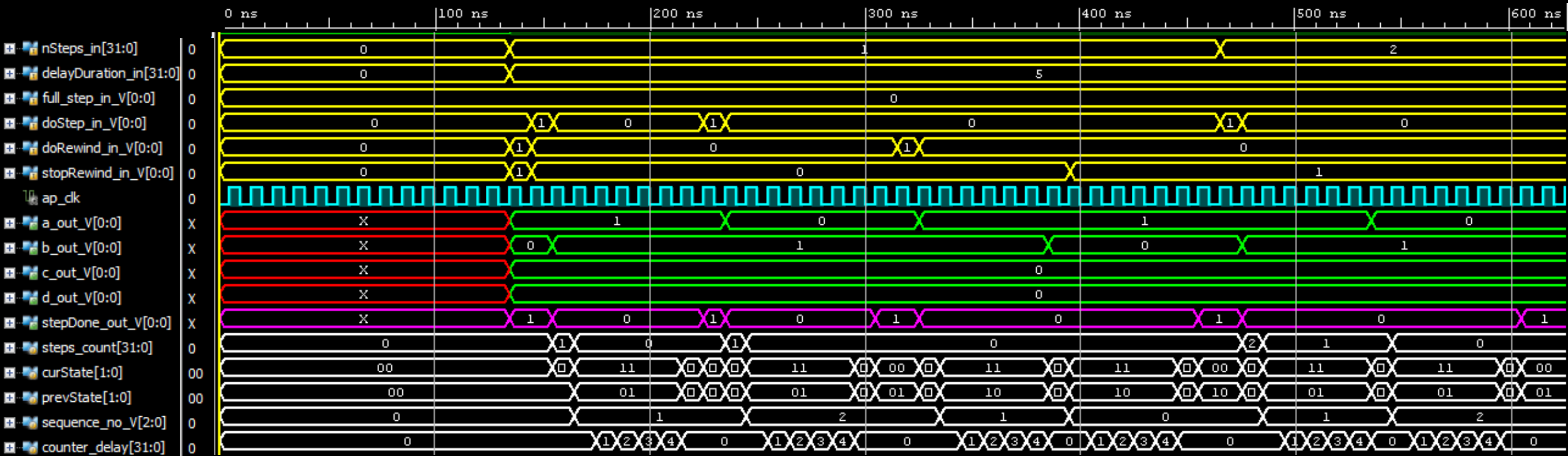


Configurazione valida per il motore **SM-42BYG011-25** by Mercury Motor (<https://www.sparkfun.com/datasheets/Robotics/SM-42BYG011-25.pdf>). Zedboard Side è stata utilizzata la porta Pmod **JA1**.

I/O Planning

Name	Direction	Board Part Pin	Board Part Interface	Neg Diff Pair	Site
[-] [v] All ports (12)					
[-] [v] a_out_V_26845 (1)	OUT				
[-] [v] a_out_V (1)	OUT				
[-] [v] a_out_V[0]	OUT				AB11
[-] [v] Scalar ports (0)					
[+] [v] ap_clk_clock_3228 (1)	IN				
[-] [v] b_out_V_26845 (1)	OUT				
[-] [v] b_out_V (1)	OUT				
[-] [v] b_out_V[0]	OUT				AB10
[-] [v] Scalar ports (0)					
[-] [v] c_out_V_26845 (1)	OUT				
[-] [v] c_out_V (1)	OUT				
[-] [v] c_out_V[0]	OUT				AB9
[-] [v] Scalar ports (0)					
[-] [v] d_out_V_26845 (1)	OUT				
[-] [v] d_out_V (1)	OUT				
[-] [v] d_out_V[0]	OUT				AA8

Testbench (HLS)



Problemi (1)

Timing in HLS	Vivado HLS non è uno strumento pensato per la soluzione di problemi caratterizzati da vincoli stringenti di tempo. Non è possibile avere accesso diretto al clock e le temporizzazioni sono influenzate da processi di ottimizzazione del compilatore C. Questo si è reso evidente durante il tentativo di realizzare una forma di attesa attiva tramite ciclo for vuoto: la sintesi ignorava questa istruzione. Il problema è stato risolto riorganizzando il codice senza cicli, diminuendo la latenza tramite direttive, tra cui <code>#pragma HLS INLINE</code> che realizza la chiamata a funzione all'interno dello stesso ciclo di clock di invocazione.
Configurazione di sistema (Linux)	Alcuni IP built-in di Vivado richiedono che la configurazione del formato dei numeri, delle date e della valuta sia obbligatoriamente quella inglese, al fine di evitare errori nella fase di aggiunta al design.

Problemi (2)

Simulazione in Vivado (VHDL)

Abbiamo riscontrato problemi durante il tentativo di simulazione del wrapper del design realizzato. Nonostante una corretta configurazione del Port Map tra componente wrapper e segnali degli stimoli esterni, non tutte le uscite risultavano collegate e restituivano il valore X (unknown) per tutta la durata della simulazione.

Si è quindi passati alla simulazione del singolo componente StepperMotor, che ha richiesto le seguenti azioni:

- Project Settings -> General -> Target language -> VHDL
- RTL Analysis -> Open Elaborated Design per generare il codice VHDL dei componenti
- Dalla tab Sources premere il tasto dx -> Add Sources -> Add or create simulation sources -> aggiungere il file .vhd del componente che si può trovare in:
 - /<nomeProgetto>.srcs/sources_1/bd/design_1/ip/<nomeComponente>/sim/<nomeComponente>.vhd
 - /<nomeProgetto>.srcs/sources_1/ipshared/xilinx.com/<nomeComponente>/<ID>/hdl/vhdl/<nomeComponente>.vhd
- Dichiarare il componente nel testbench rispettando il nome indicato nel file aggiunto al passo precedente. Nel primo caso, il nome dipende dall'identificativo usato nel block design e per la dichiarazione e il Port Map si può usare il template contenuto nel file /<nomeProgetto>.srcs/sources_1/bd/design_1/ip/<nomeComponente>/<nomeComponente>.vho.
Nel secondo caso, il nome corrisponde a quello della funzione top level di Vivado HLS ed è indipendente dal block design.

Problemi (3)

Generazione di IP da VHDL in Vivado

- Dalla tab Sources, tasto dx -> Add Sources -> Add or create design sources -> Create File.
- Il file verrà creato in /<nomeProgetto>.srcs/sources_1/new/.
Se c'è necessità di creare diversi IP, è necessario creare una cartella per ciascuno di questi, contenente i file .vhd che ne specificano il comportamento.
- Tools -> Create and Package IP -> Next -> Package a specified directory -> selezionare la directory contenente il solo file .vhd -> Next -> specificare il nome del sotto-progetto che verrà creato -> Finish
- Si apre un'altra finestra di Vivado in cui è possibile specificare altre informazioni (facoltative) sull'IP generato.
- Tornare nel block design del progetto principale e aggiungere la cartella del sotto-progetto come IP-Repository per poter poi inserire l'IP nel design.

Collegamento motore

Ciascun motore ha una propria convenzione tra colore del filo e magnete pilotato, quindi è necessario controllare lo schematico del prodotto per effettuare il corretto collegamento con il PMOD e l'I/O Planning delle porte dell'FPGA.