Circuiti combinatori notevoli e aritmetici

Architetture degli Elaboratori I, Laboratorio - Corso di Laurea in Informatica, A.A. 2019-2020

Esercizio 1

- Si progetti e si implementi in Logisim il circuito di un decodificatore a 2 bit

- Si utilizzi il decodificatore così creato per implementare in Logisim un multiplexer a 4 vie

Esercizio 1

- Si progetti e si implementi in Logisim il circuito di un decodificatore a 2 bit
  Suggerimento: il decodificatore riceve in ingresso una sequenza di 2 bit e attiva in uscita una delle 4 linee, in particolare quella identificata dalla sequenza di bit in ingresso

- Si utilizzi il decodificatore così creato per implementare in Logisim un multiplexer a 4 vie

Esercizio 1

- Si progetti e si implementi in Logisim il circuito di un decodificatore a 2 bit
  Suggerimento: il decodificatore riceve in ingresso una sequenza di 2 bit e attiva in uscita una delle 4 linee, in particolare quella identificata dalla sequenza di bit in ingresso

- Si utilizzi il decodificatore così creato per implementare in Logisim un multiplexer a 4 vie
  Suggerimento: il multiplexer seleziona una delle quattro linee in ingresso e la lascia passare in uscita

Esercizio 1

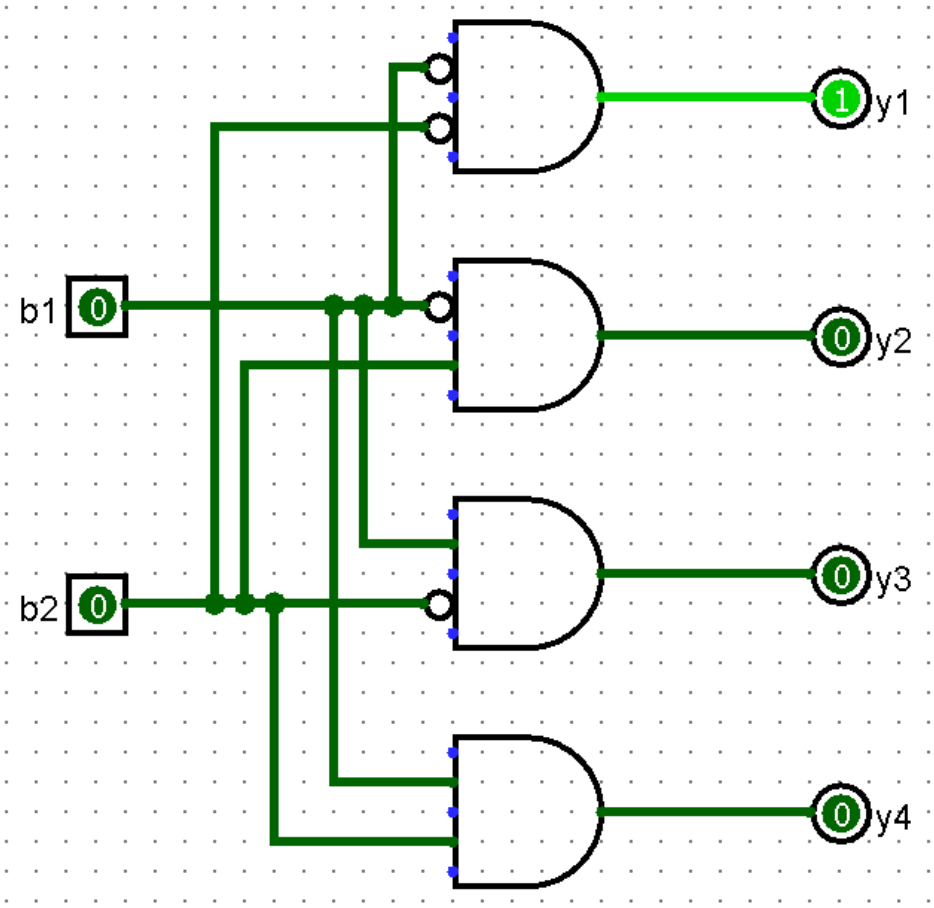
Tabella di verità del decodificatore a 2 bit

b_1	b_2	y_1	y_2	y_3	y_4
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Esercizio 1

Tabella di verità del decodificatore a 2 bit

b_1	b_2	y_1	y_2	y_3	y_4
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

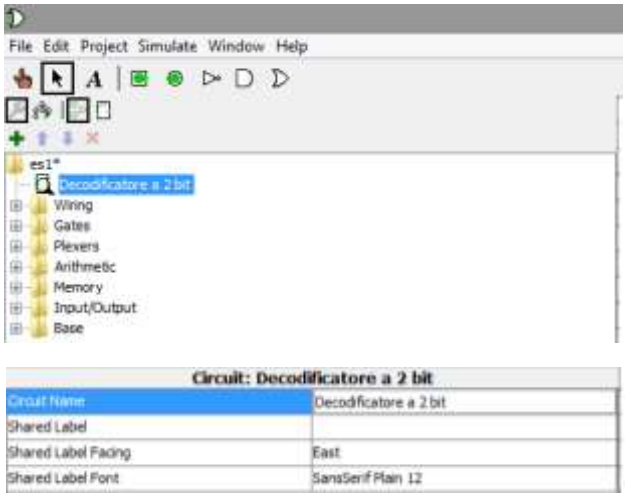


SOP

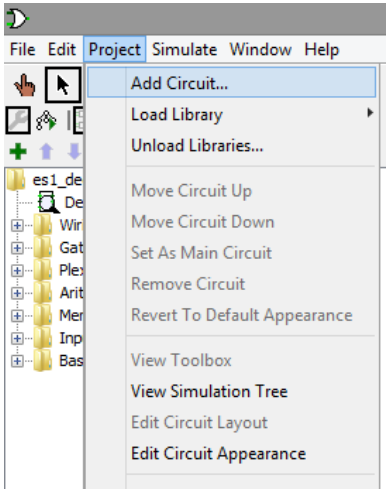
Esercizio 1

Aggiungiamo il decodificatore a 2 bit creato agli elementi di libreria e utilizziamolo in un altro circuito (woekbench)

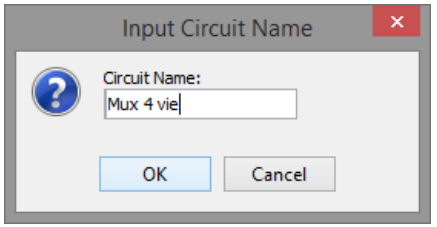
Rinominazione del nome del circuito



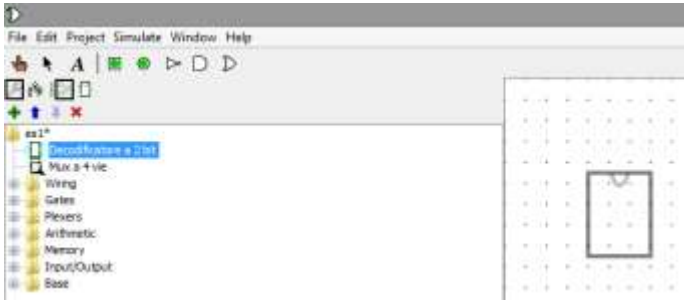
Aggiunta di un circuito al progetto



Nome del circuito aggiunto

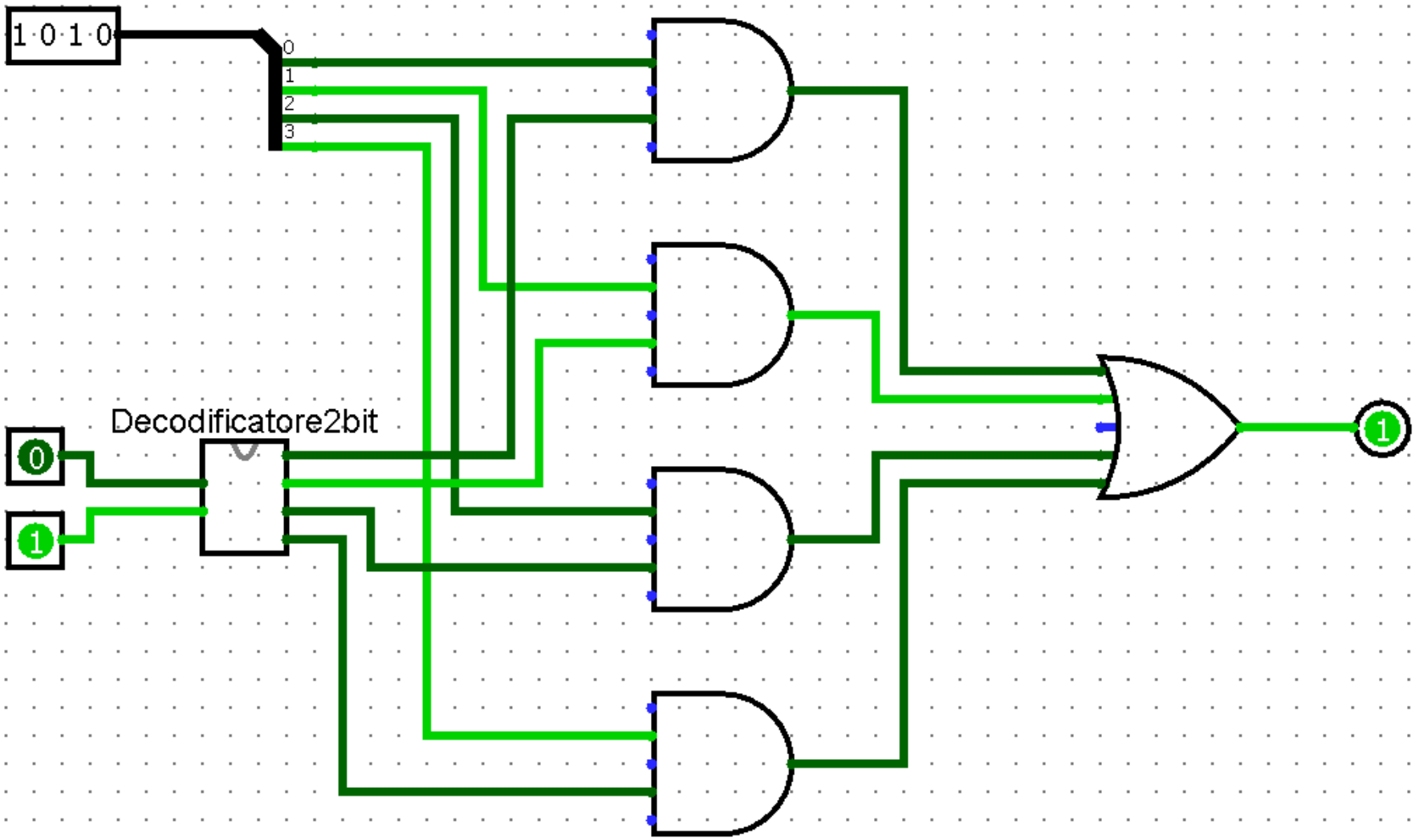


Selezione decodificatore a 2 bit



Esercizio 1

Multiplexer a 4 vie



Esercizio 2

- Si scriva la tabella di verità per un addizionatore ad 1 bit senza riporto (half adder)
- Se ne dia un'implementazione in Logisim e si salvi il circuito

Esercizio 2

- Si scriva la tabella di verità per un addizionatore ad 1 bit senza riporto (half adder)

- Se ne dia un'implementazione in Logisim e si salvi il circuito

Suggerimento: si utilizzi la porta XOR per limitare il numero di porte che compaiono nel circuito

Esercizio 2

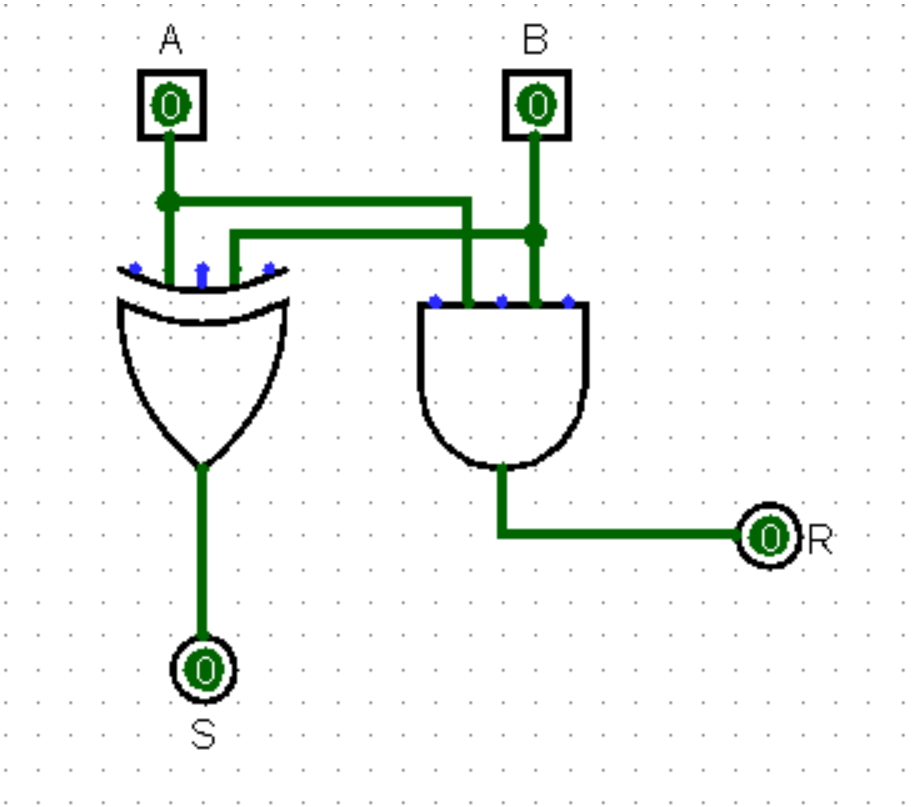
Tabella di verità half adder a 1 bit

A	B	S	R
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Esercizio 2

Tabella di verità half adder a 1 bit

A	B	S	R
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



Esercizio 3

- Si scriva la tabella di verità per un addizionatore ad 1 bit con riporto in ingresso (Full Adder)
- Se ne dia un'implementazione in Logisim basata su SOP e si salvi il circuito
- Si fornisca poi una versione semplificata utilizzando il circuito Half Adder precedentemente creato

Esercizio 3

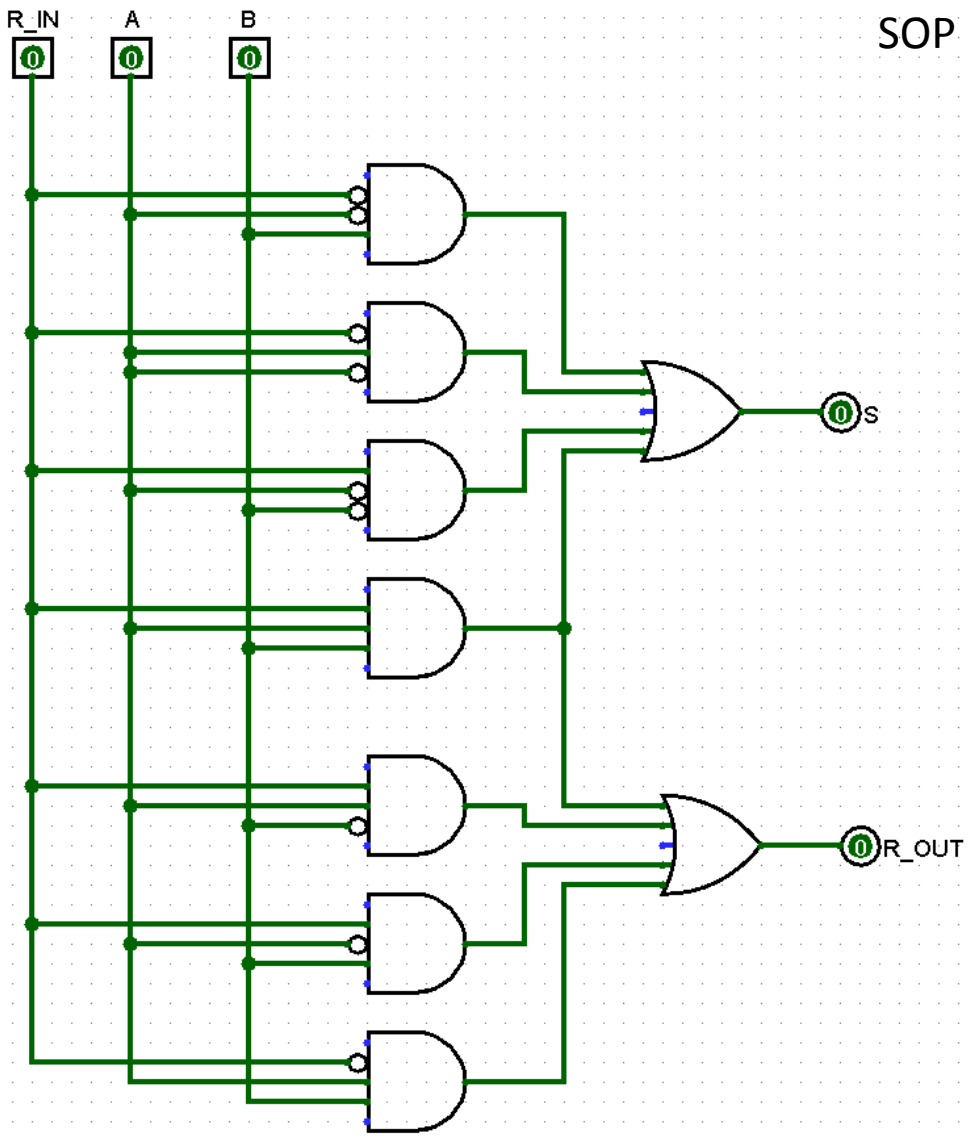
Tabella di verità

R_{in}	A	B	S	R_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Esercizio 3

Tabella di verità

R_{in}	A	B	S	R_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

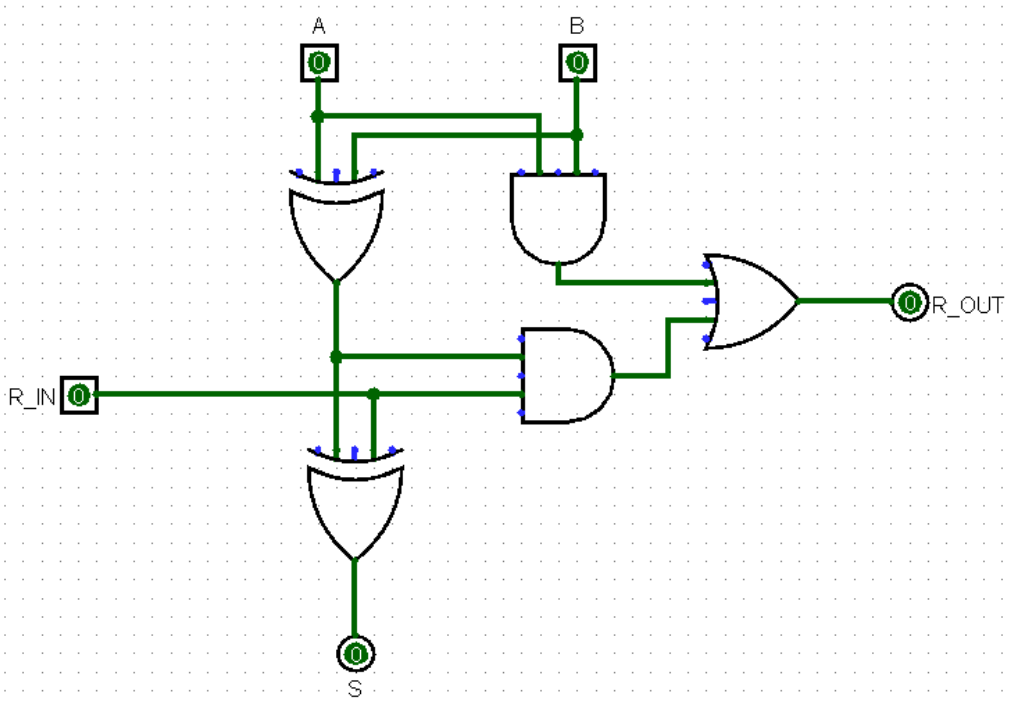


Esercizio 3

Tabella di verità

R_{in}	A	B	S	R_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Circuito semplificato

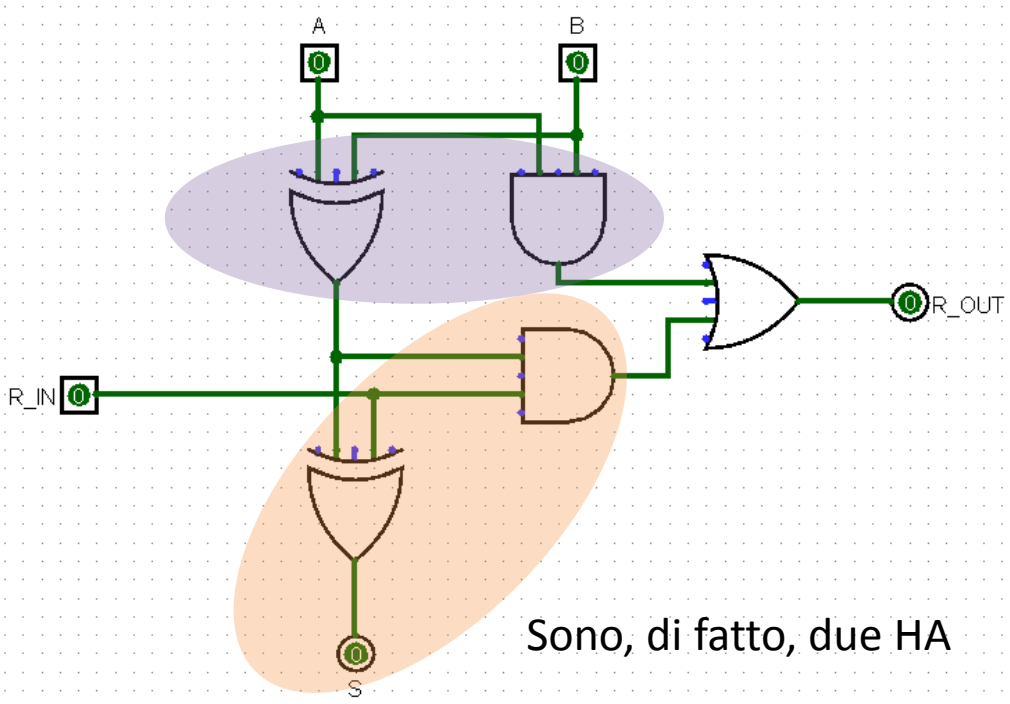


Esercizio 3

Tabella di verità

R_{in}	A	B	S	R_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

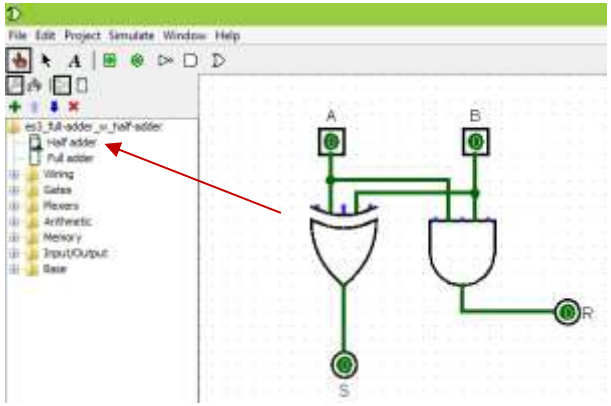
Circuito semplificato



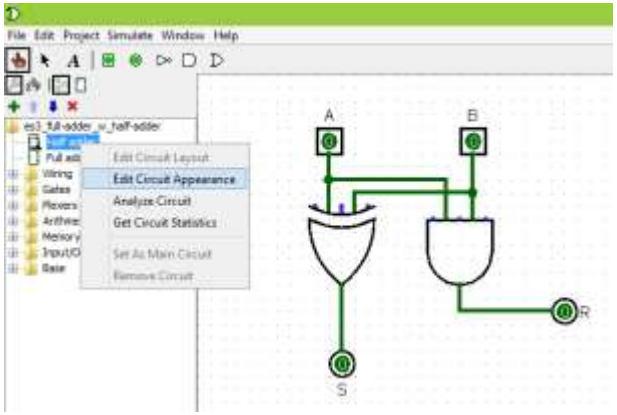
Sono, di fatto, due HA

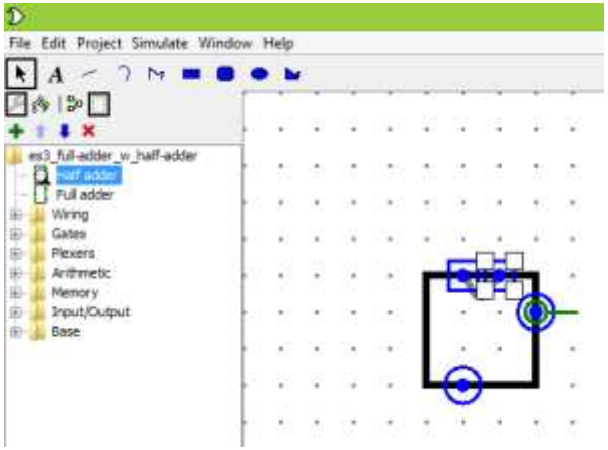
Esercizio 3

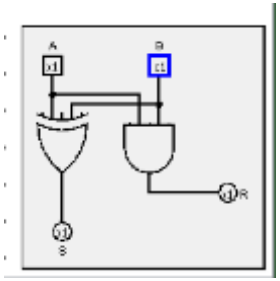
Creare un circuito HA da poter utilizzare come componente in altri circuiti



Editare il layout della rappresentazione astratta del circuito



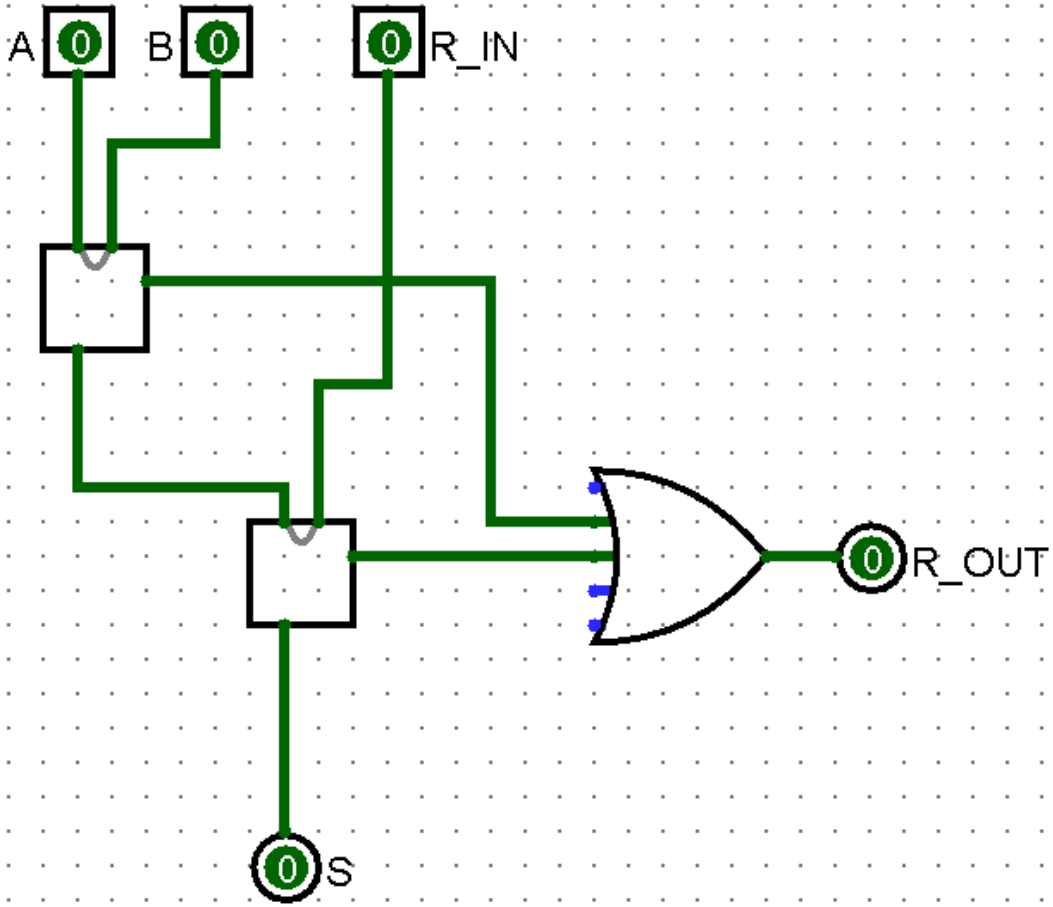




Esercizio 3

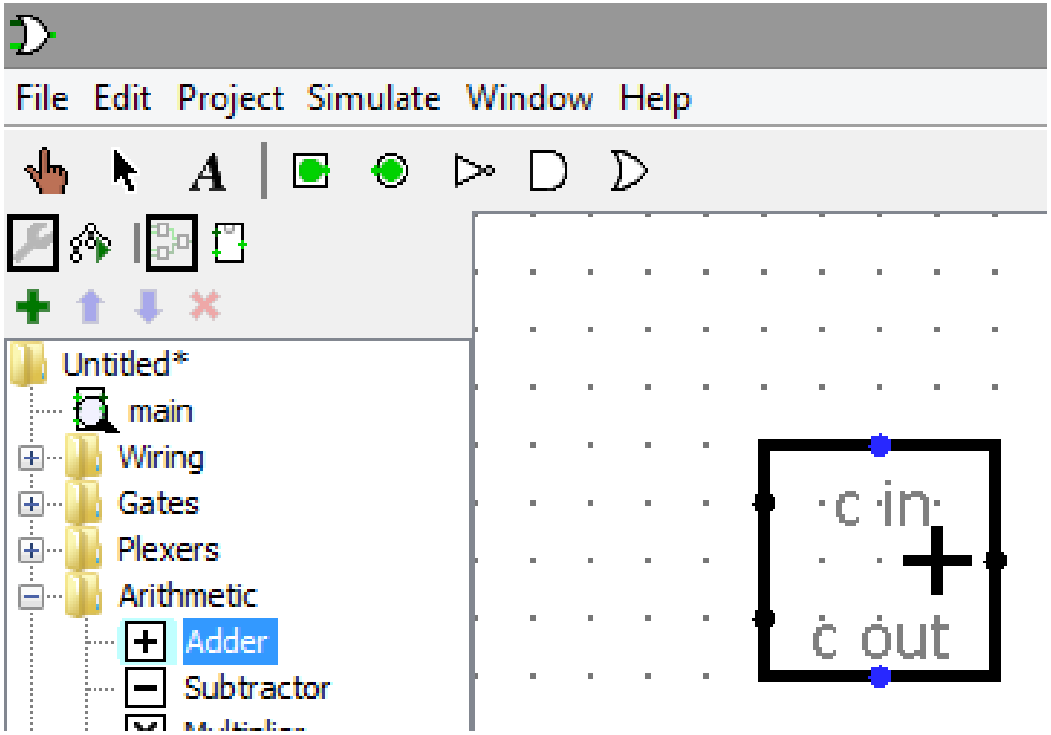
Circuito con Half Adder in cui compare la loro rappresentazione astratta

R_{in}	A	B	S	R_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



Esercizio 3

In Logisim, Full Adder corrisponde al modulo Adder

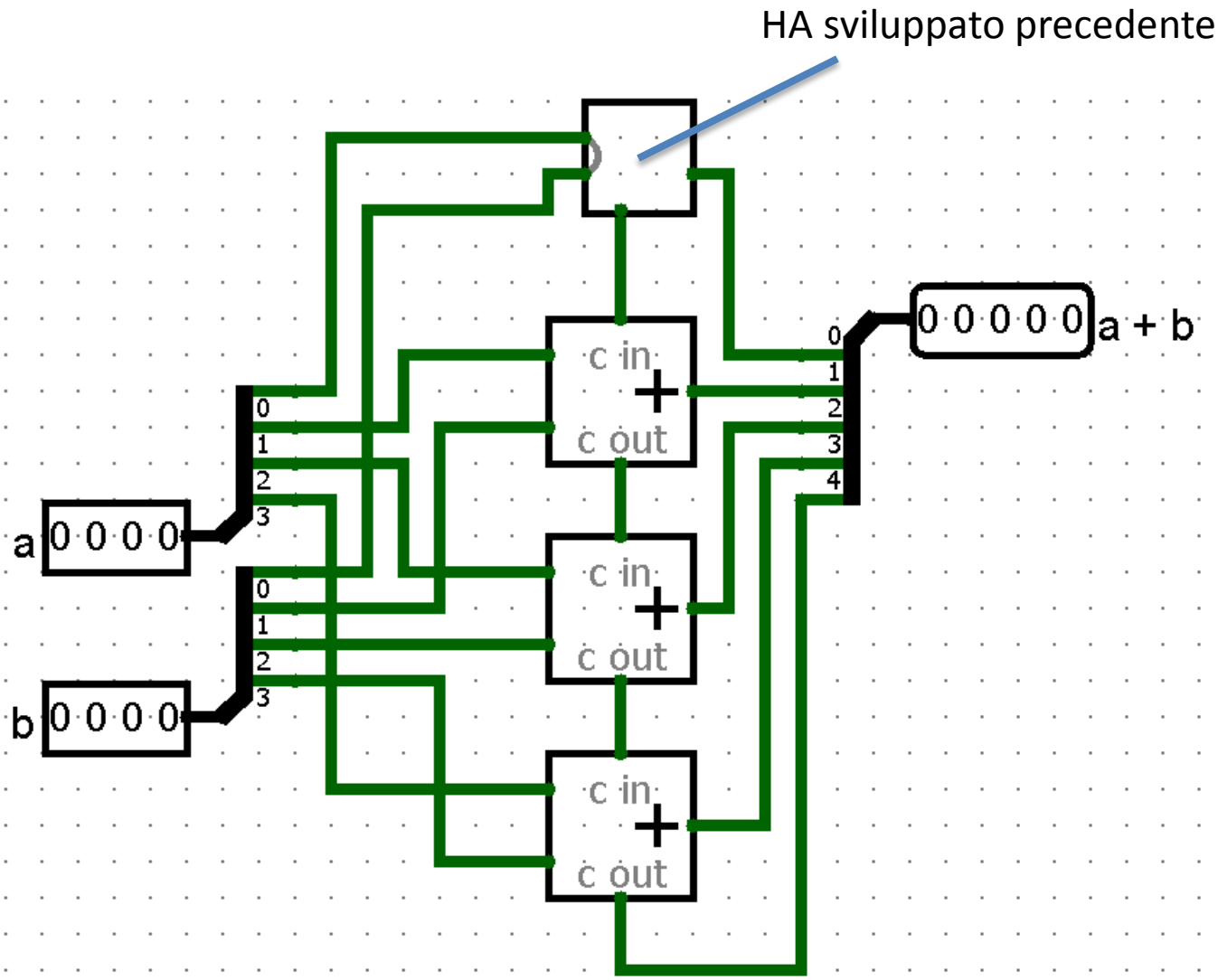


Esercizio 4

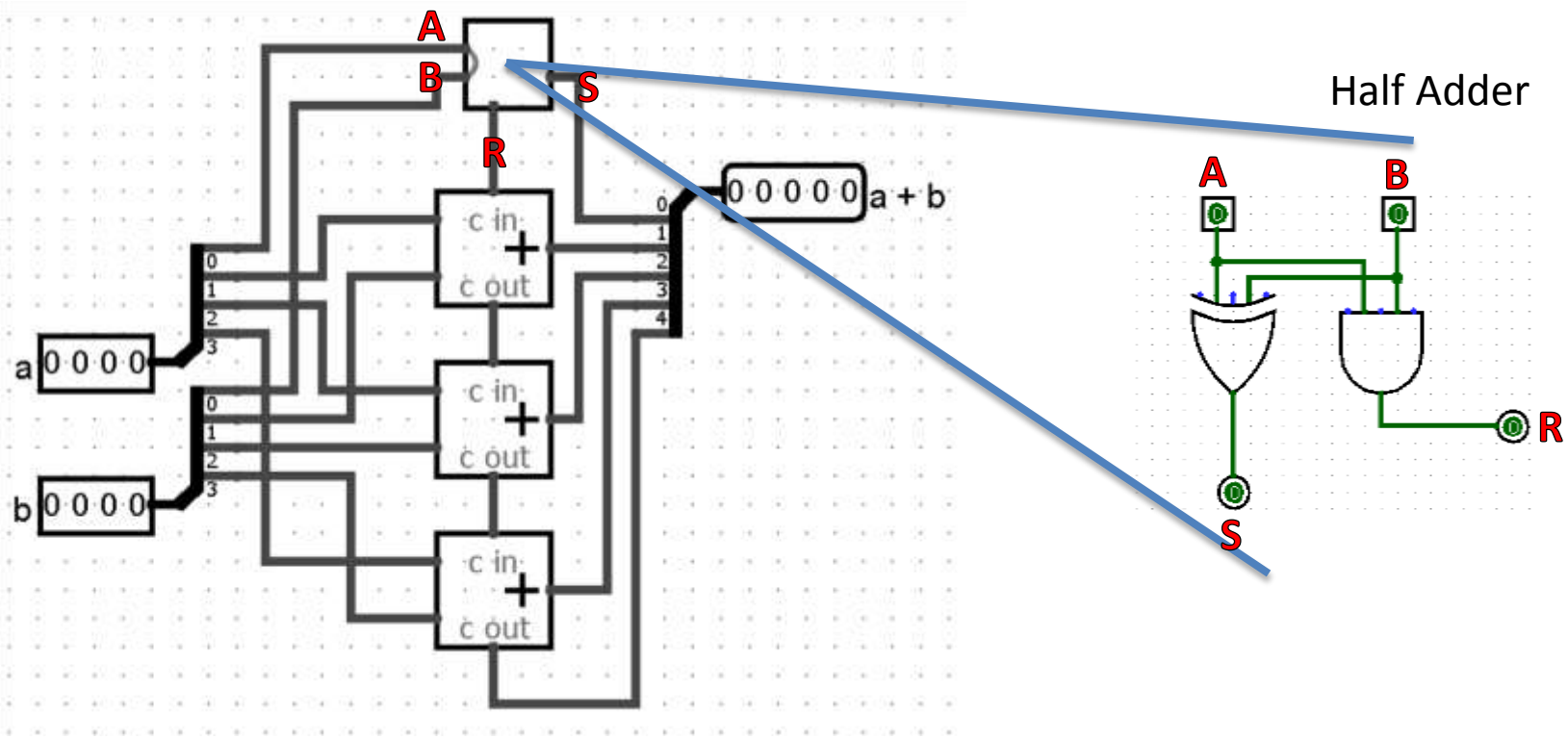
- Si utilizzino il circuito Half Adder precedentemente sviluppato e il modulo Adder per realizzare un addizionatore a 4 bit in Logisim

- Si analizzi il cammino critico del circuito così implementato (per l'uscita somma e per l'uscita riporto)

Esercizio 4

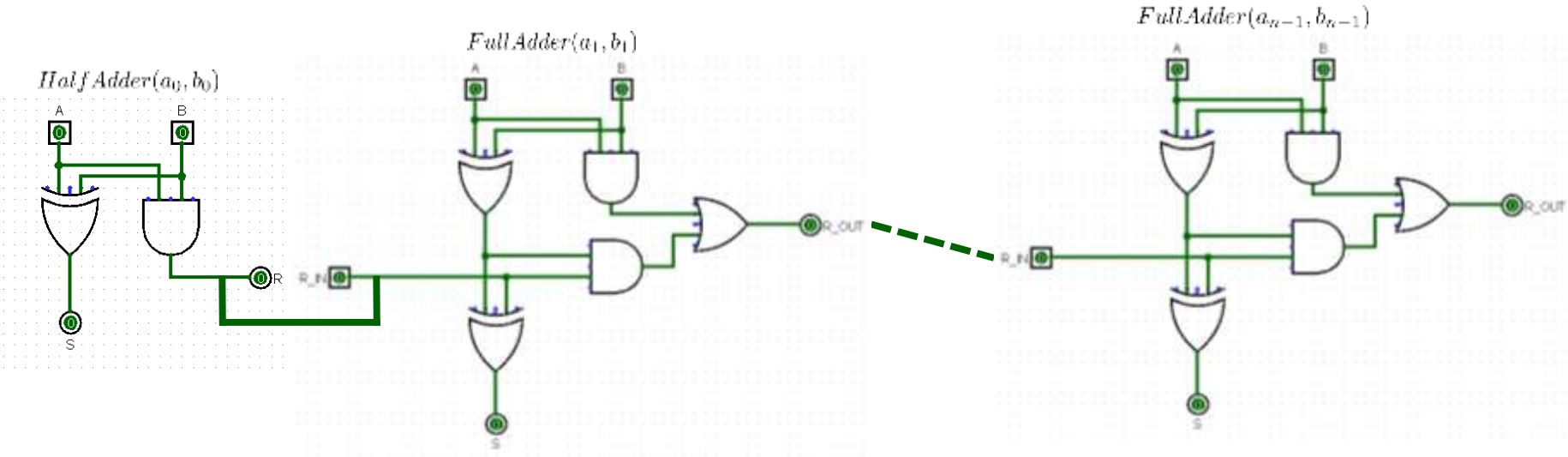

HA sviluppato precedentemente
a
0 0 0 0
b
0 0 0 0
c in
c out
c in
c out
c in
c out
0 0 0 0 0
a + b

Esercizio 4


A
B
S
R
c in
c out
c in
c out
c in
c out
a
0 0 0 0
b
0 0 0 0
0 0 0 0 0
a + b
Half Adder
A
B
R
S

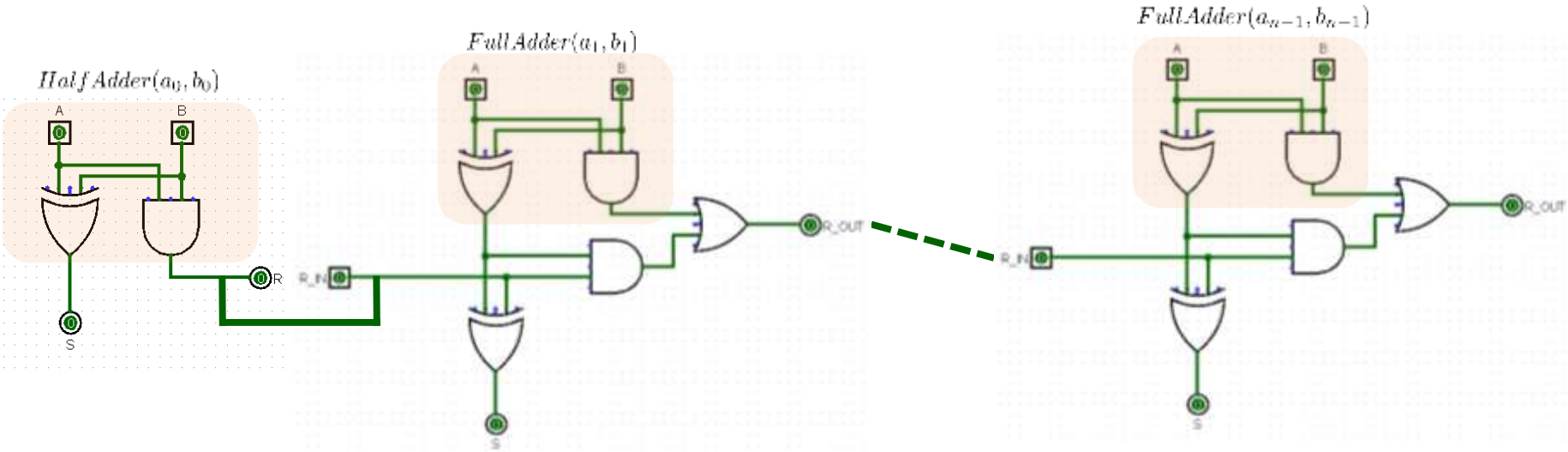
Esercizio 4

- Cammino critico ⓧ = segnale disponibile dopo x hop



Esercizio 4

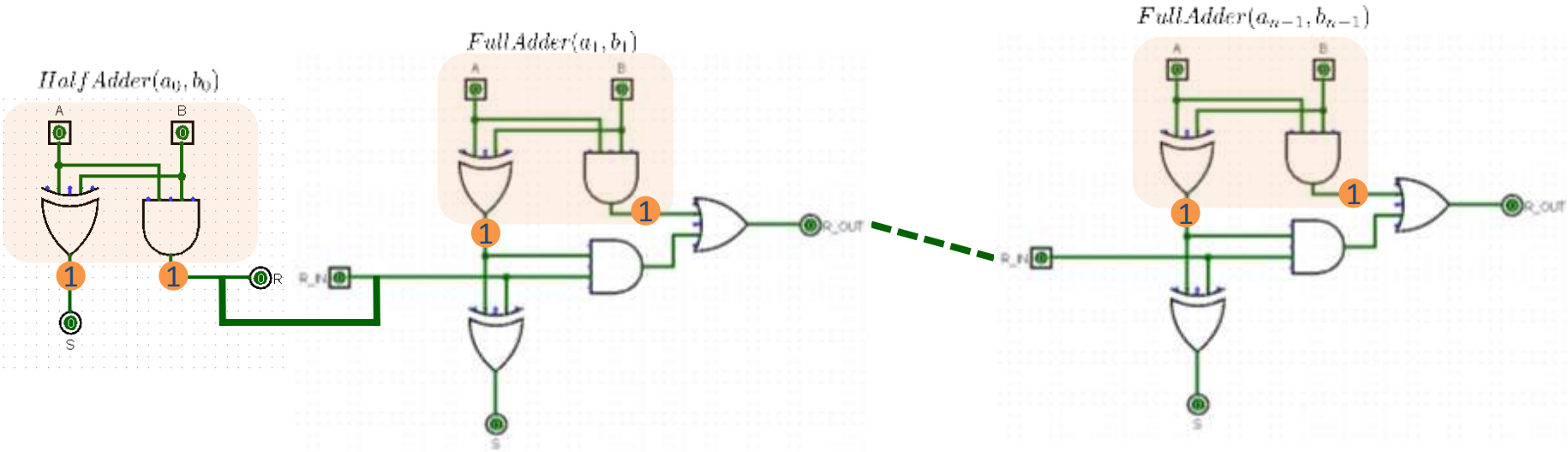
- Cammino critico ⓧ = segnale disponibile dopo x hop



- 1° livello di porte (e 1° riporto) (1) 1

Esercizio 4

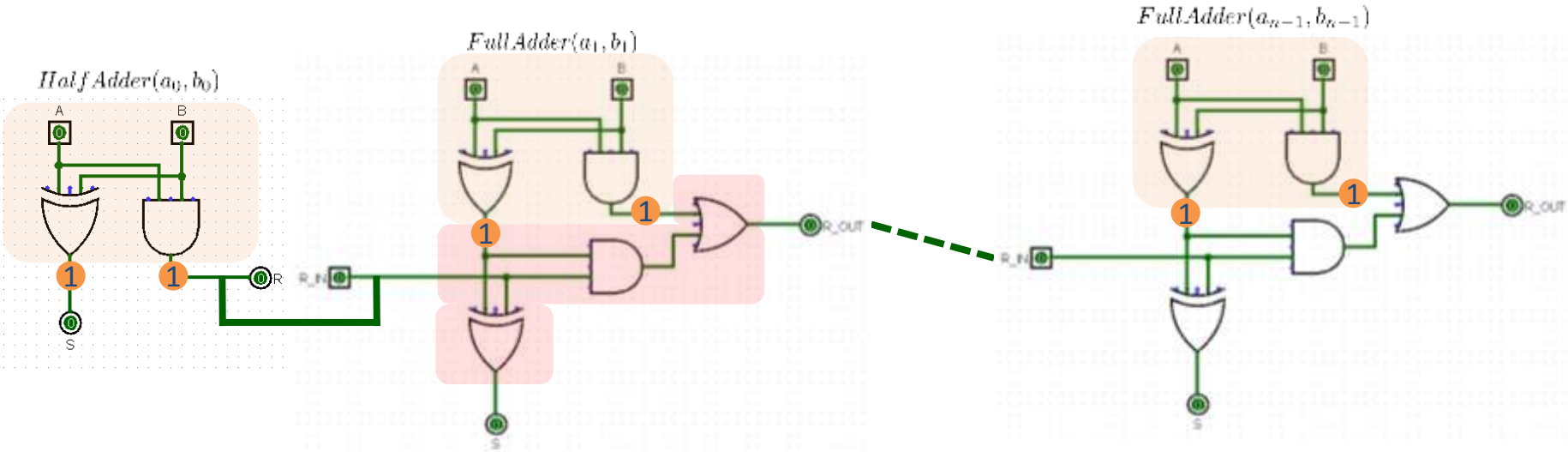
- Cammino critico ⓧ = segnale disponibile dopo x hop



- 1° livello di porte (e 1° riporto) (1) 1

Esercizio 4

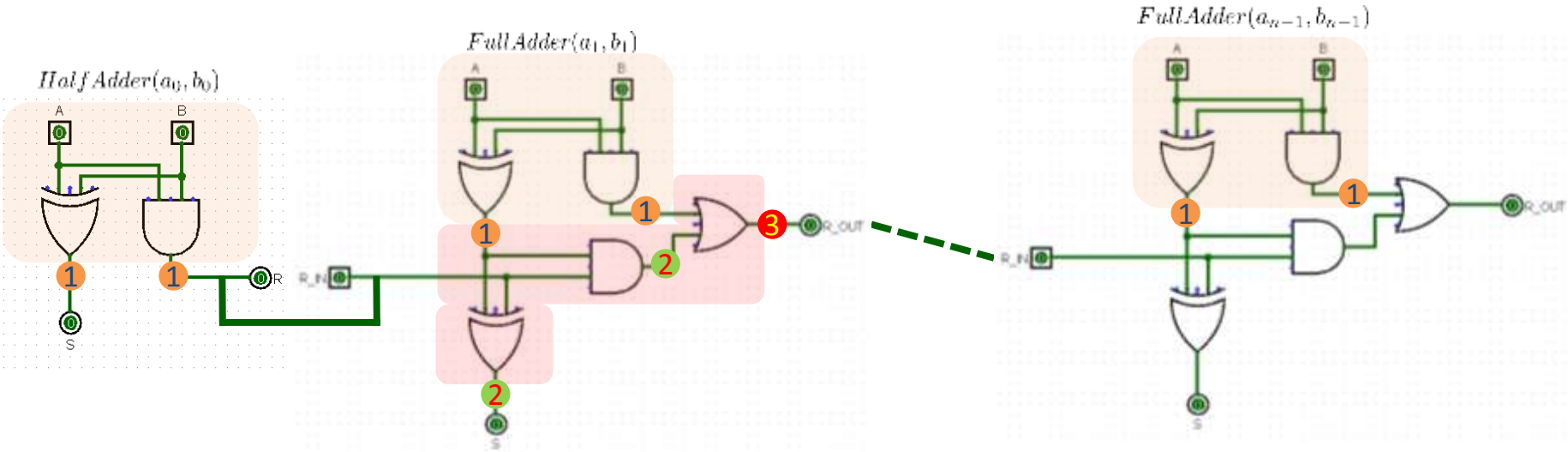
- Cammino critico ⓧ = segnale disponibile dopo x hop



- 1° livello di porte (e 1° riporto) (1) 1
- 2° riporto (+2) 2 3

Esercizio 4

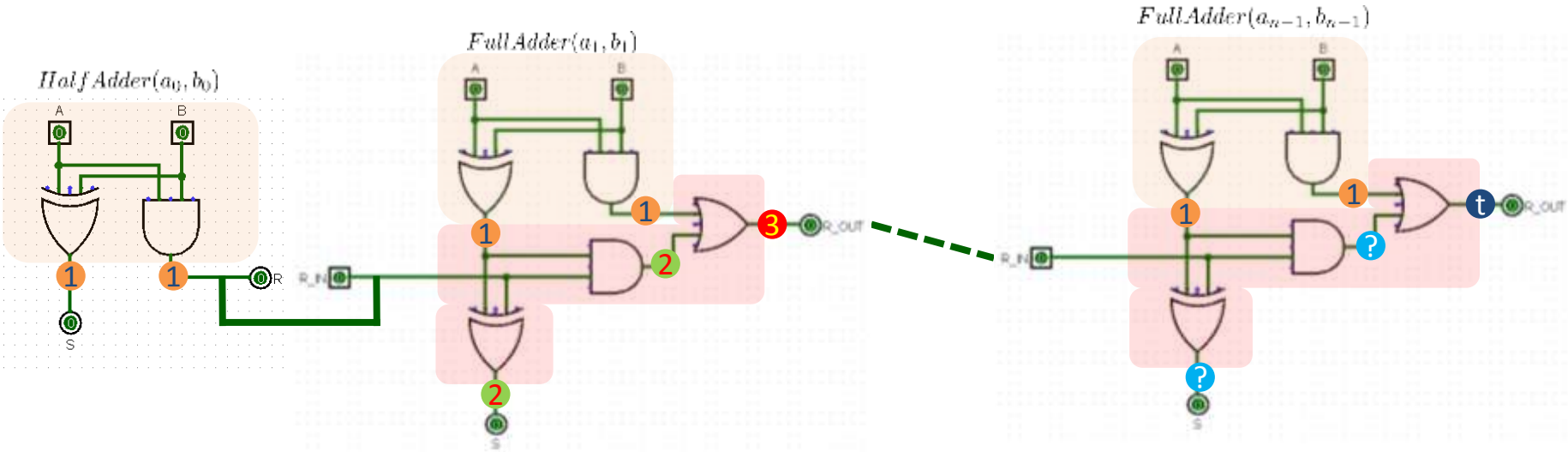
- Cammino critico ⓧ = segnale disponibile dopo x hop



- 1° livello di porte (e 1° riporto) (1) 1
- 2° riporto (+2) 2 3

Esercizio 4

- Cammino critico ⓧ = segnale disponibile dopo x hop

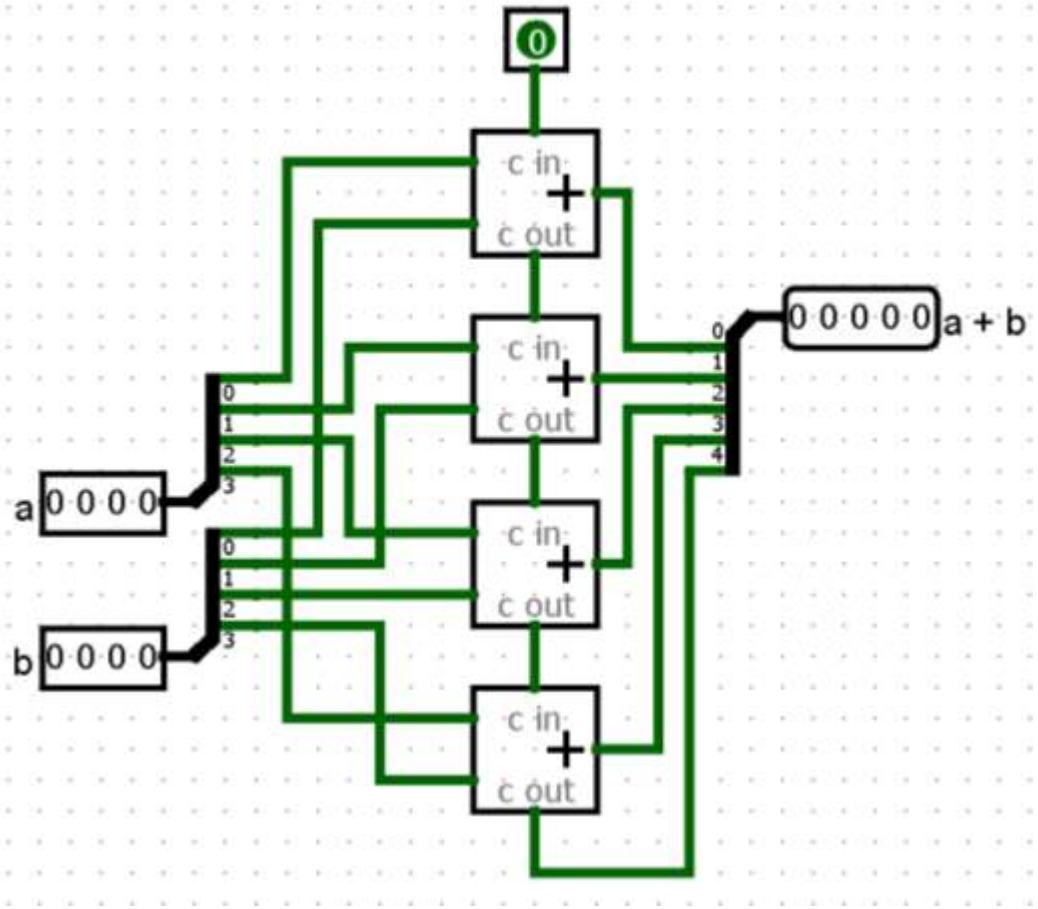


- 1° livello di porte (e 1° riporto) (1) 1
- 2° riporto (+2) 2 3
- ...
- n° riporto (+2)

Totale: c=1+ 2(n-1) t

Esercizio 4

- Utilizzando solo moduli Adder



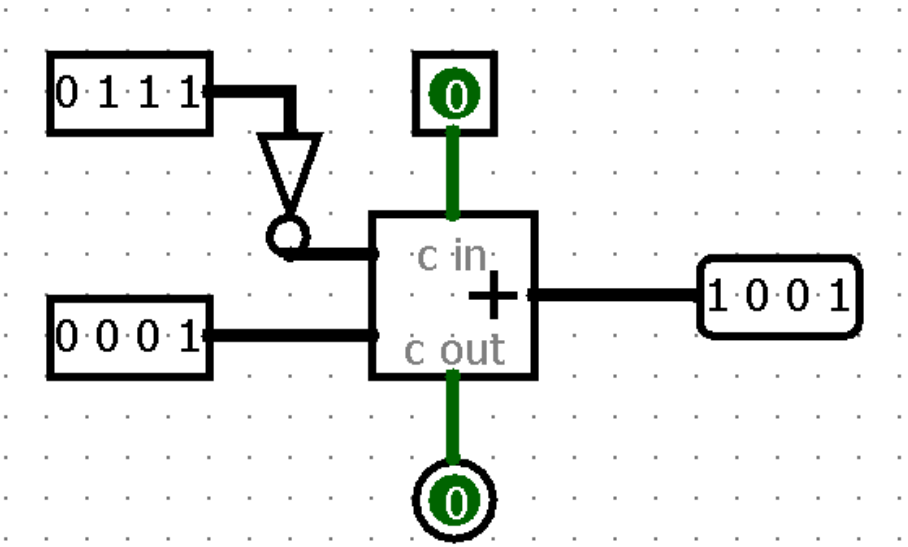
Cammino critico in questo caso?

Esercizio 5

- Si realizzi il circuito che, a partire da un numero X in formato binario standard, fornisca in uscita il numero –X in complemento a 2

Esercizio 5

- Si realizzi il circuito che, a partire da un numero X in formato binario standard, fornisca in uscita il numero –X in complemento a 2



- Per quali valori il circuito funziona correttamente?

Esercizio 5

- Il circuito funziona correttamente solo per ingressi binari compresi tra: 0000 (0000 in C2)e 0111 (1001 in C2)

- Per numeri maggiori o uguali a 1000 abbiamo un overflow

Esercizio 6

- Si realizzi un circuito che operi la somma e la differenza di due numeri A e B a 4 bit, utilizzando un bit di selezione dell'operazione S

Esercizio 6

- Si realizzi un circuito che operi la somma e la differenza di due numeri A e B a 4 bit, utilizzando un bit di selezione dell'operazione S

- Sommo A e B in C2, convertendo B in –B se S=1

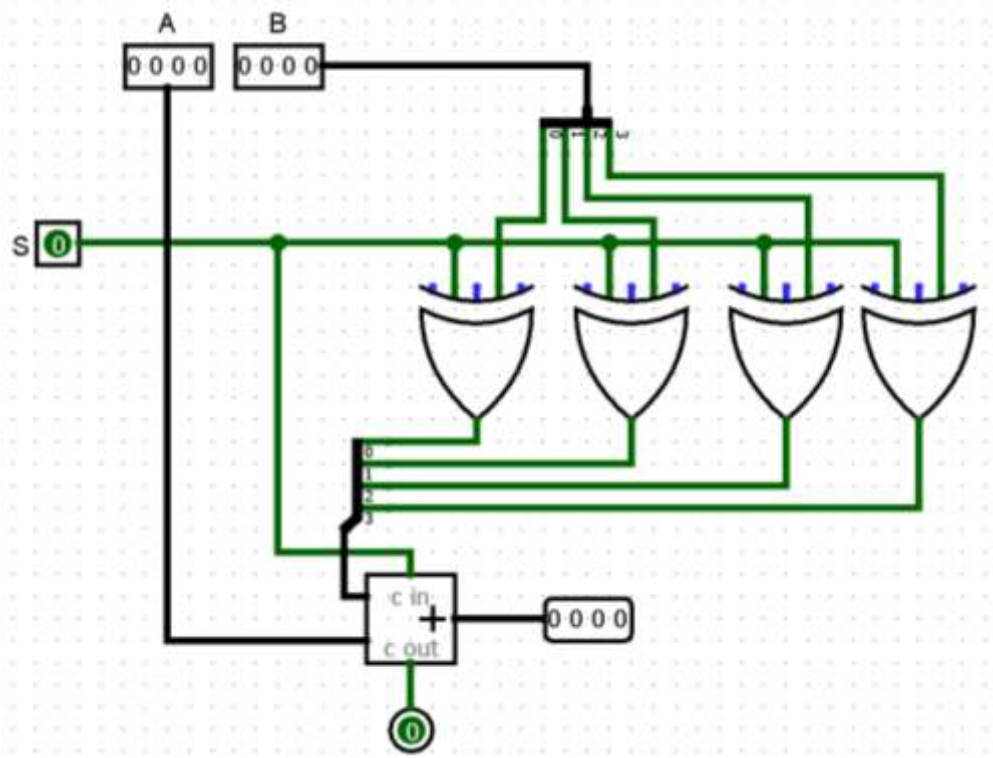
S	b_i	$output$
0	0	0
0	1	1
1	0	1
1	1	0

Se S=1 devo invertire tutti i bit di B
(posso usare delle porte XOR)

- La somma di 1 può essere gestita interpretando S come il riporto in ingresso

Esercizio 6

- Si realizzi un circuito che operi la somma e la differenza di due numeri A e B a 4 bit, utilizzando un bit di selezione dell'operazione S



Esercizio 7

- Si modifichi il circuito realizzato all'esercizio precedente in modo che rilevi la presenza di un overflow

- Si calcoli il cammino critico del circuito

Esercizio 7

- Quando si verifica l'overflow se si somma in C2?

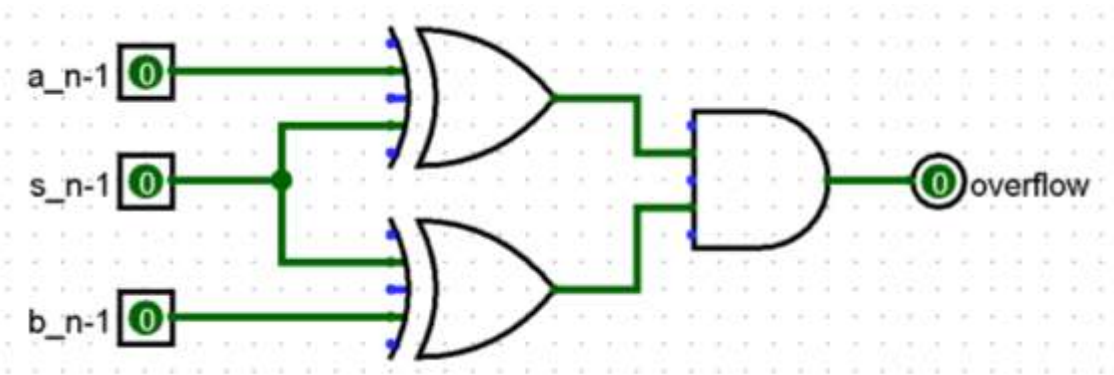
1. A e B sono positivi e il segno del risultato è negativo
2. A e B sono negativi e il segno del risultato è positivo

Esercizio 7

- Quando si verifica l'overflow se si somma in C2?

1. A e B sono positivi e il segno del risultato è negativo
2. A e B sono negativi e il segno del risultato è positivo

s_{n-1}	a_{n-1}	b_{n-1}	$Overflow$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0



Esercizio 7