MOALU

Multi-Operation ALU

Stefano Pievaioli matr. 816592

Introduzione

L'obiettivo dell'elaborato è quello di progettare una MOALU. Ho diviso il progetto in 3 parti principali e, a seguire, una conclusione:

Schema a blocchi

Breve descrizione dello schema a blocchi che ho realizzato.

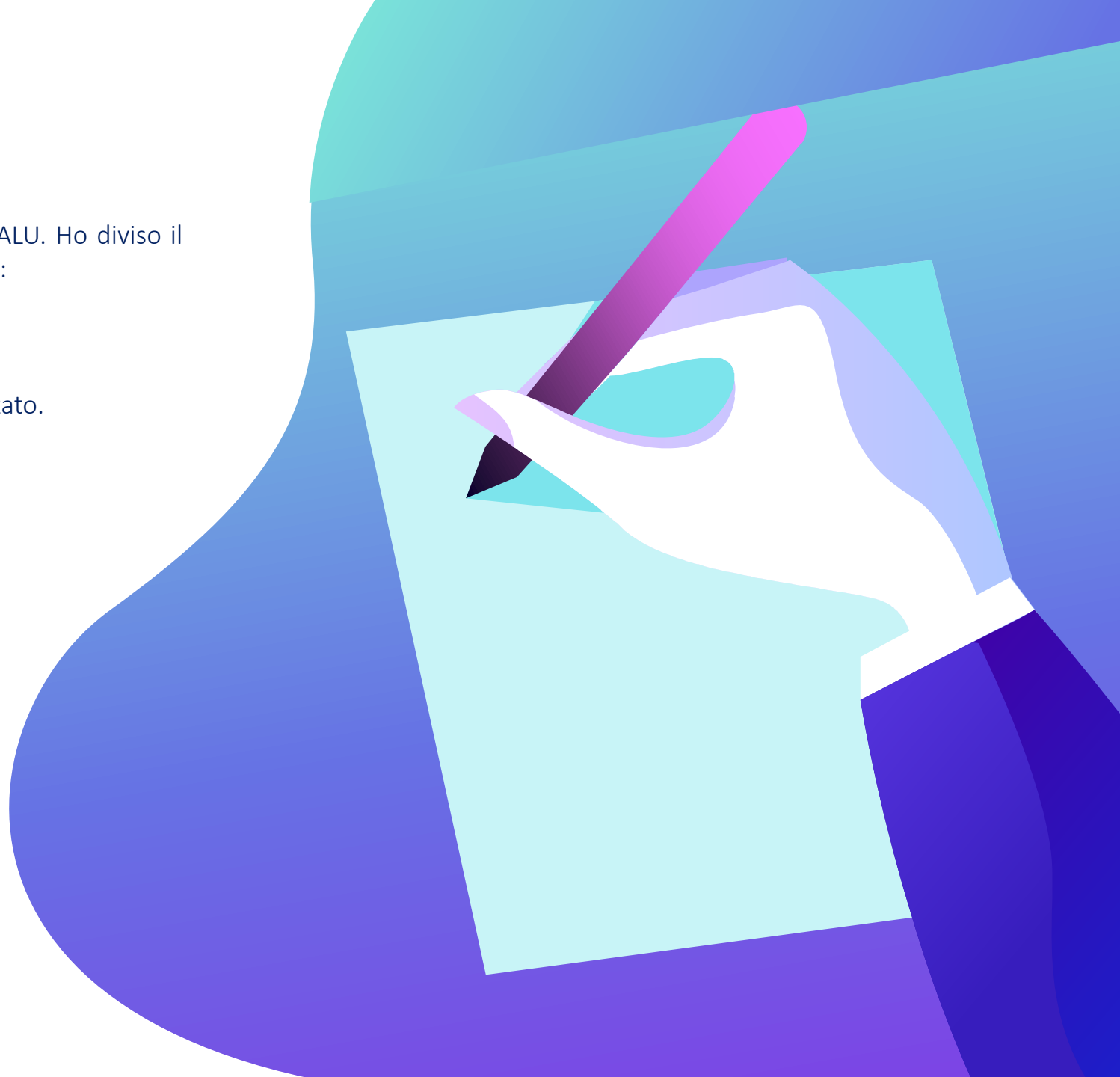
FSM

Descrizione di come ho realizato la FSM.

Test bench

Risultati ottenuti

Conclusione



Schema a Blocchi

La MOALU ha 4 ingressi:

- X, ingresso seriale che regola la Finite State Machine.
- Reset, bit che azzera i registri e porta il sistema in stand-by a logica negativa (‘1’ reset non attivo, ‘0’ reset attivo).
- Change, bit che cambia l’argomento di C2 o inverte il minuendo con il sottraendo della Differenza (se Change = ‘0’ -> C2(A) e/o A – B, se Change = ‘1’ -> C2(B) e/o B – A).
- Clk, Clock del sistema.

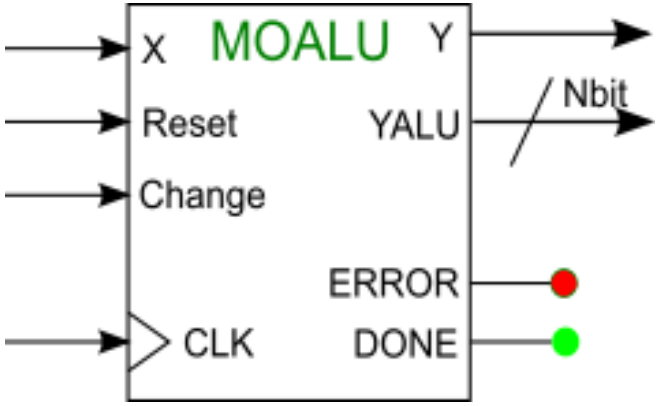
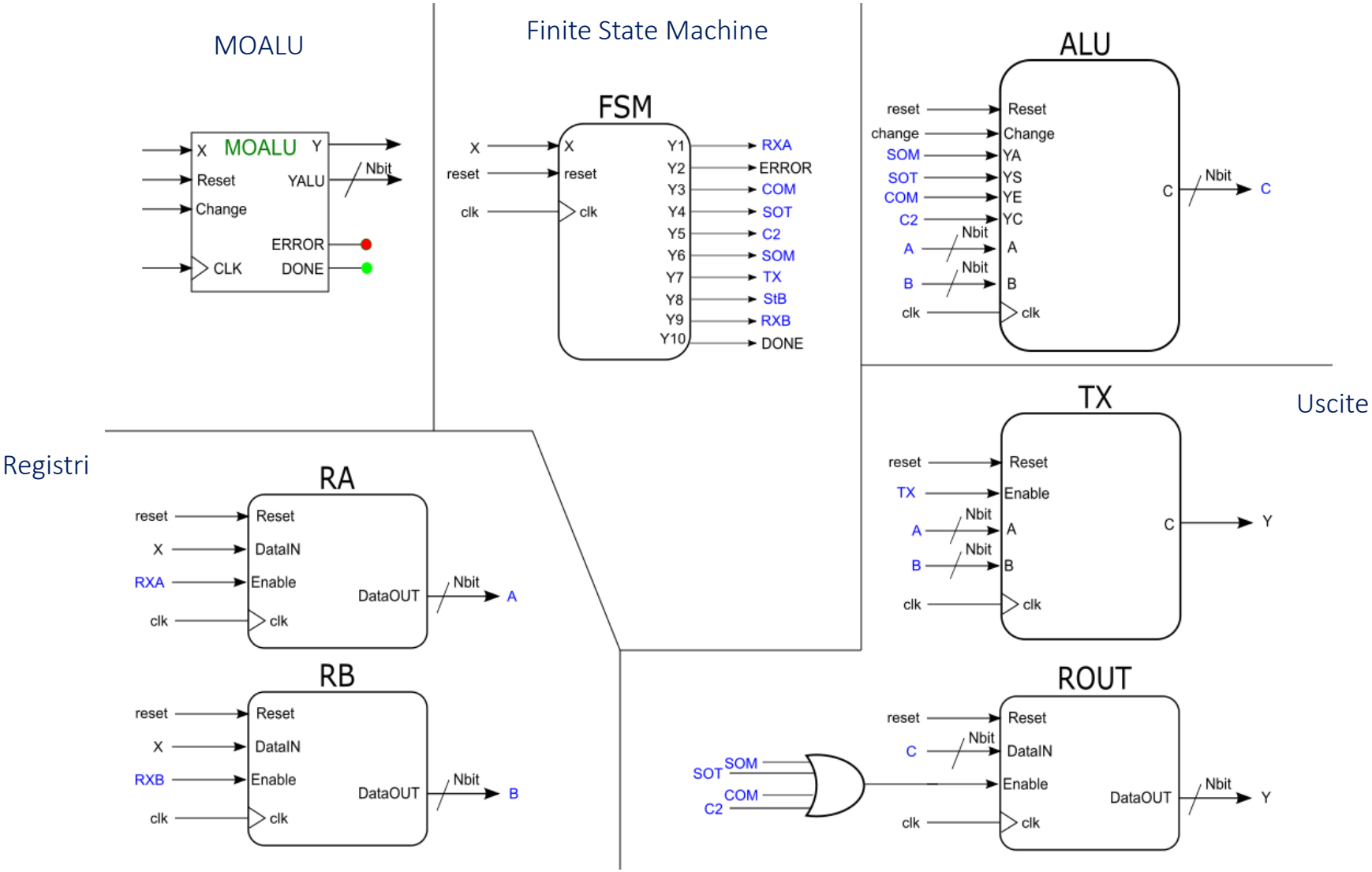


Figura 1, Schema I/O MOALU.

Le uscite invece sono:

- Y, uscita seriale dell’operazione TX.
- YALU, uscita parallela del registro RegOut, contenente il risultato delle operazioni svolte dall’ALU.
- Error, bit a uno quando viene inserita la combinazione ‘110’. Permane in questo stato fino ad un reset.
- Done, bit che va ad indicare il risultato sull’uscita Y o YALU.

Schema a Blocchi



Finite State Machine

In figura 2 è rappresentato lo schema semplificato della FSM. Non sono riuscito a disegnare la FSM realizzata nella MOALU in quanto essa resta negli stati attraverso l'utilizzo di un **counter**.
Quando la FSM entra in uno degli stati di operazione, come ad esempio SRX, attraverso il counter integrato resta in quello stato fino al termine dell'operazione.
Una volta che ha terminato l'operazione torna nello stato S1 o S0 in base all'input X.

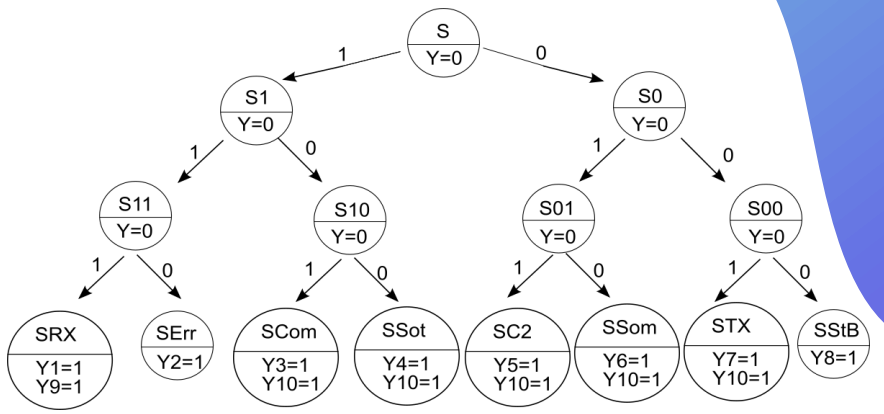


Figura 2, Schema FSM.

```
when S01 =>
    Counter <= 0;

    if X = '0' then
        State   <= S_SOM;   --S010
        Y6      <= '1';
    else
        State   <= S_C2;  --S011
        Y5      <= '1';
    end if;
```

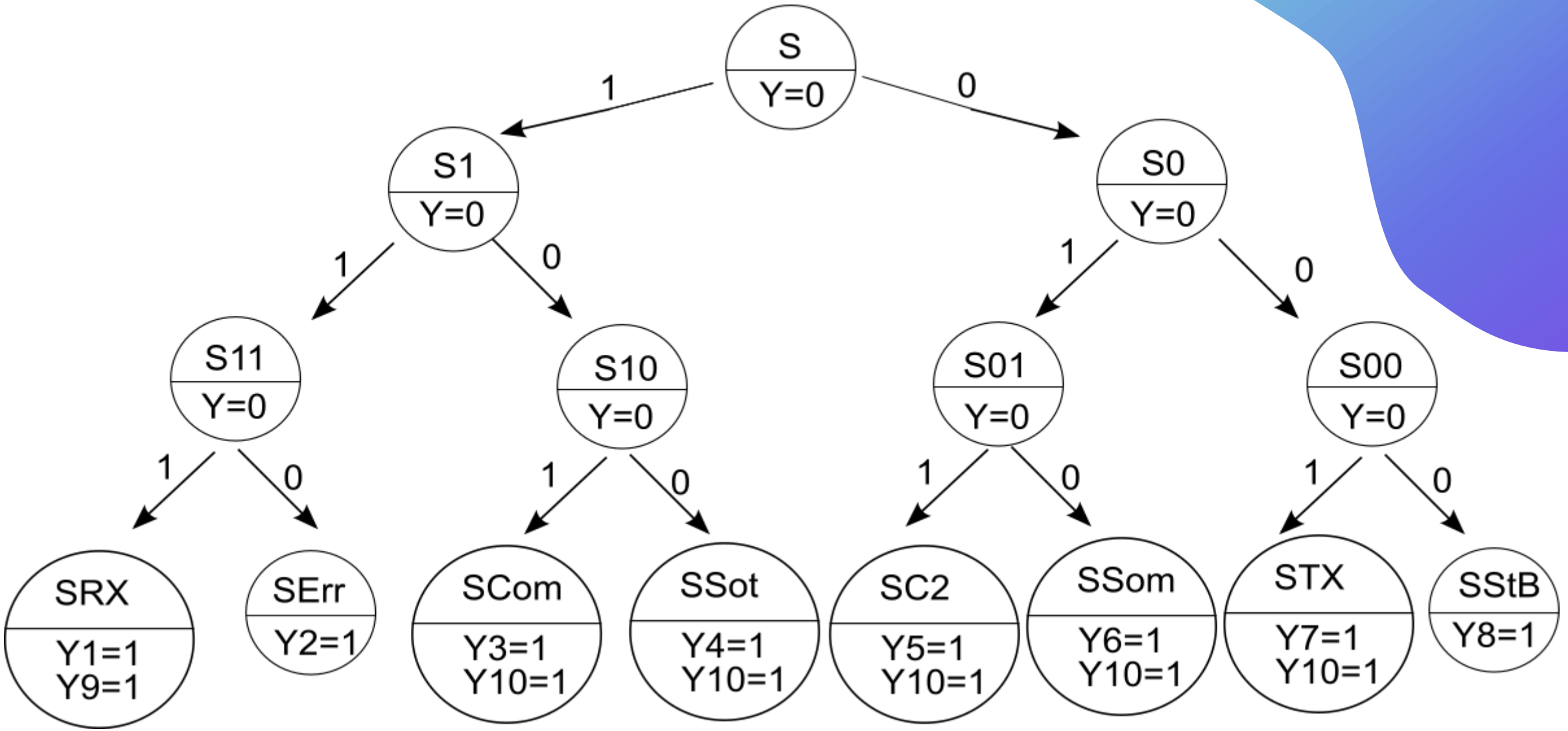
Figura 3, Codice VHDL per lo stato S01 .

Il Counter è la variabile che viene incrementata a ogni fronte positivo del clock. Quando sono nello stato Sxx il bit di X successivo mi porta in uno stato operazione. Quindi azzero il counter e alzo l'uscita in base all'operazione da fare (come in figura 2). Quando sono nello stato operazione il valore di X viene cosiderato solo quando ho terminato l'operazione, ovvero quando il counter termina l'esecuzione.

```
when S_COM =>
    if Counter > 1 then
        if X = '0' then
            State   <= S0;  --S0
        else
            State   <= S1;  --S1
        end if;
    else
        if Counter = 0 then
            Y3 <= '1';
        else
            Y10 <= '1';
        end if;
    end if;
```

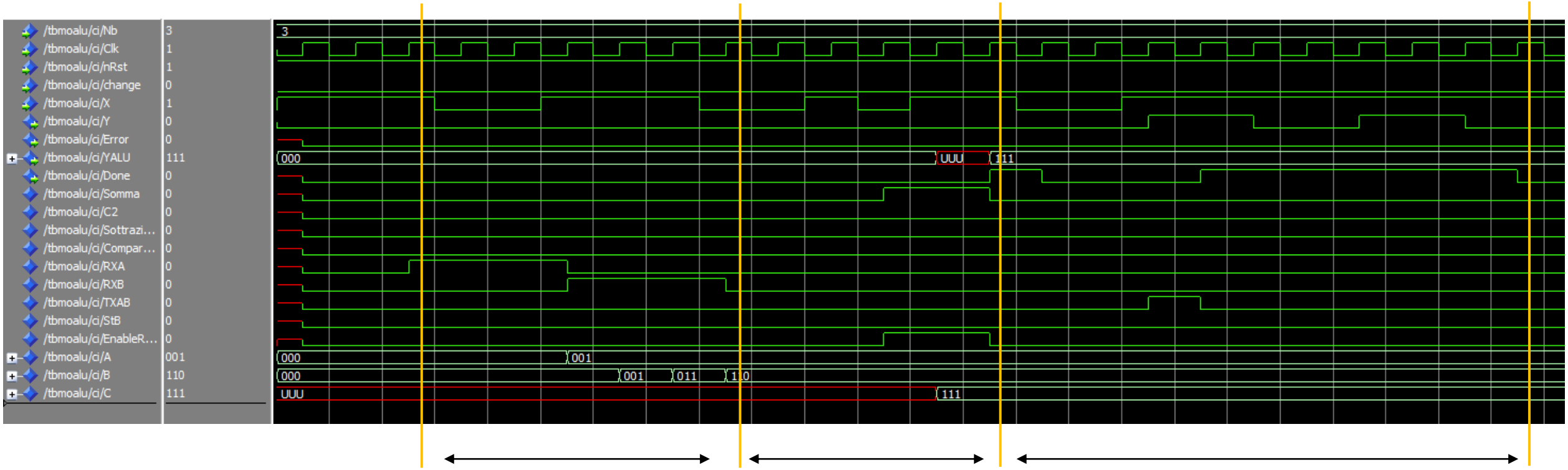
Figura 4, Codice VHDL per lo stato Comparazione .

Finite State Machine



Test bench 1

RX → Somma → TX



RX

Una volta inseriti i 3 uno la fsm passa in stato RX nel quale abilita i due regirstri. Le uscite a 1 sono RXA e RXB

SOMMA

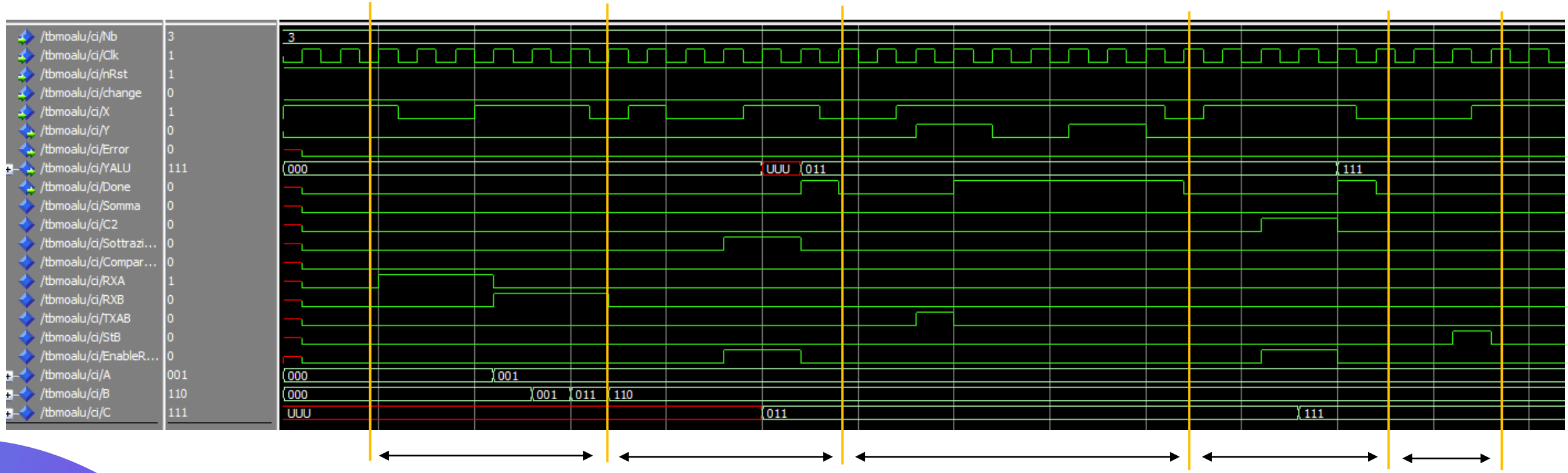
Dopo i 3 bit dei valori A e B, passa nello stato di somma, esegue la somma e il clock successivo scrive il valore nel regtro RegOut. Nel clock successivo possiamo vedere l'uscita YALU con il valore dell'operazione e il bit a Done a 1.

TX

Dopo i 3 bit dei valori A e B, passa nello stato di somma, esegue la somma e il clock successivo scrive il valore nel regtro RegOut. Il clock successivo possiamo vedere l'uscita YALU con il valore dell'operazione e il bit a Done a 1.

Test bench 2

RX → Sottrazione → TX → C2 → Stand-By



RX

Una volta inseriti i 3 uno la fsm passa in stato RX nel quale abilita i due regirstri. Le uscite a 1 sono RXA e RXB

SOTTRAZIONE

Come per la somma esegue la sottrazione e il clock successivo scrive il valore nel regtro RegOut. Il clock successivo possiamo vedere l'uscita YALU con il valore dell'operazione e il bit a Done a 1.

TX

Dopo i 3 bit dei valori A e B, passa nello stato di somma, esegue la somma e il clock successivo scrive il valore nel regtro RegOut. Il clock successivo possiamo vedere l'uscita YALU con il valore dell'operazione e il bit a Done a 1.

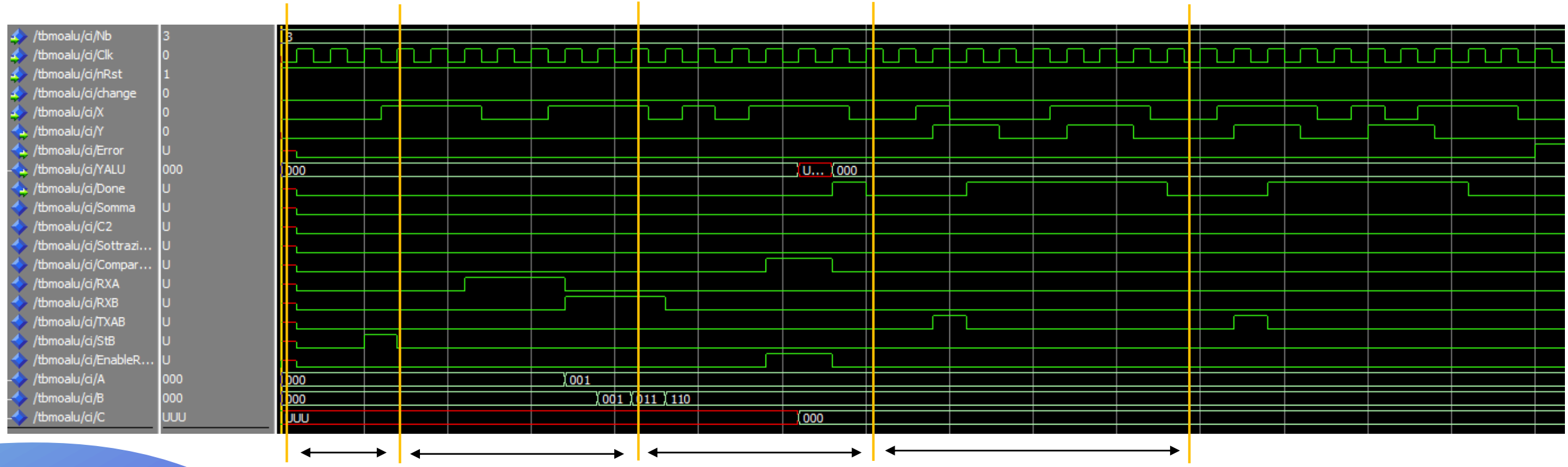
C2

Come per sottrazione.

StandBy

Test bench 3

Stand-By → RX → Comparazione → TX



StandBy

RX

SOTTRAZIONE

Come per tutte le operazioni dell'ALU esegue la comparazione e il clock successivo scrive il valore nel regtro RegOut. Il clock successivo possiamo vedere l'uscita YALU con il valore dell'operazione e il bit a Done a 1.

TX

Conclusione

Imperfezioni

Durante lo svolgimento del progetto non ho riscontrato «difficoltà» o problemi riguardanti il codice. L'unico errore che ho notato è che il blocchetto TX, che non è altro che un registro PISO, inverte i bit del primo dato. (se A = "100" l'uscita invece "001").

Possibili migliorie future

Per migliorare il Progetto si potrebbero aggiungere due bit di uscita che indicano se la MOALU e' in fase di operazione o di ricezione (es. BUSY).

UNIVERSITÀ DEGLI STUDI DI MILANO BICOCCA

Grazie

per l'attenzione