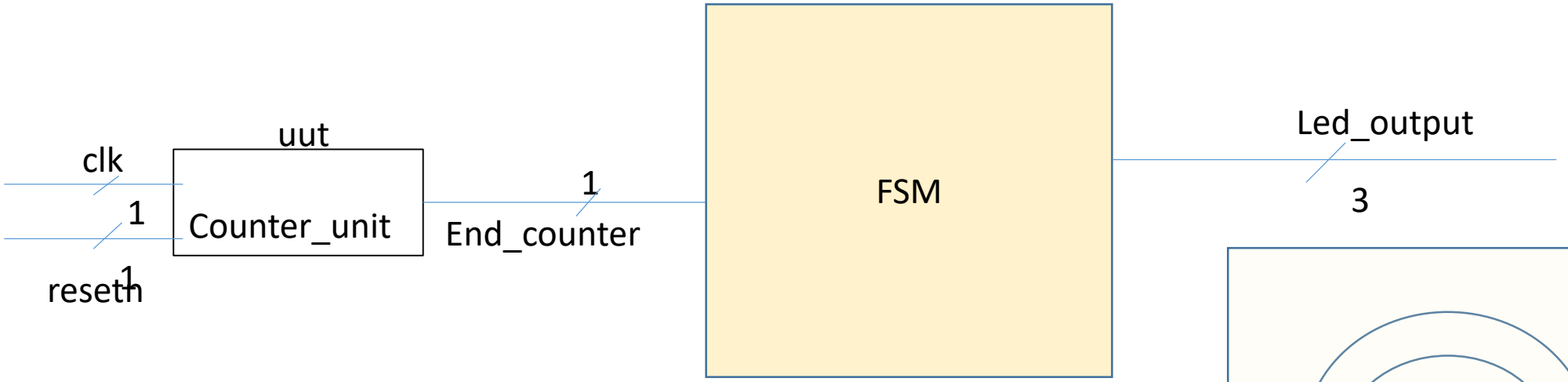
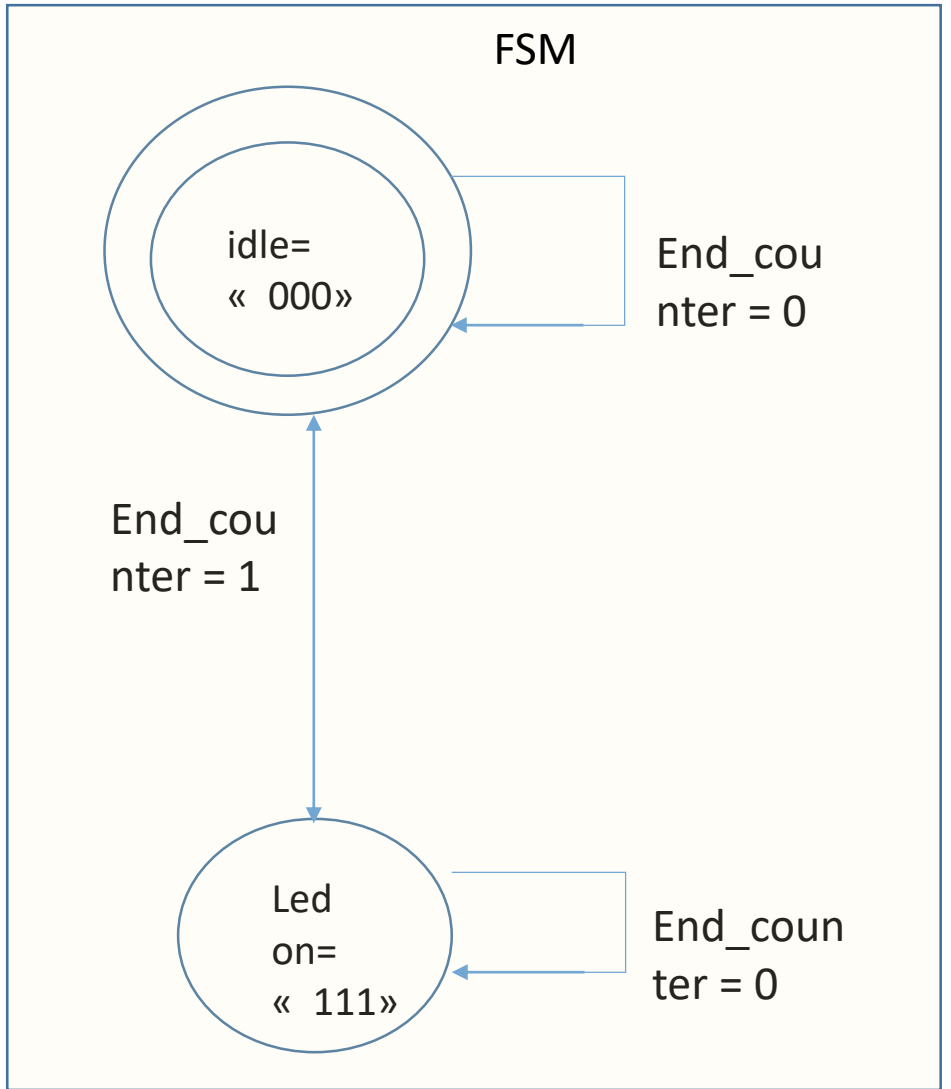
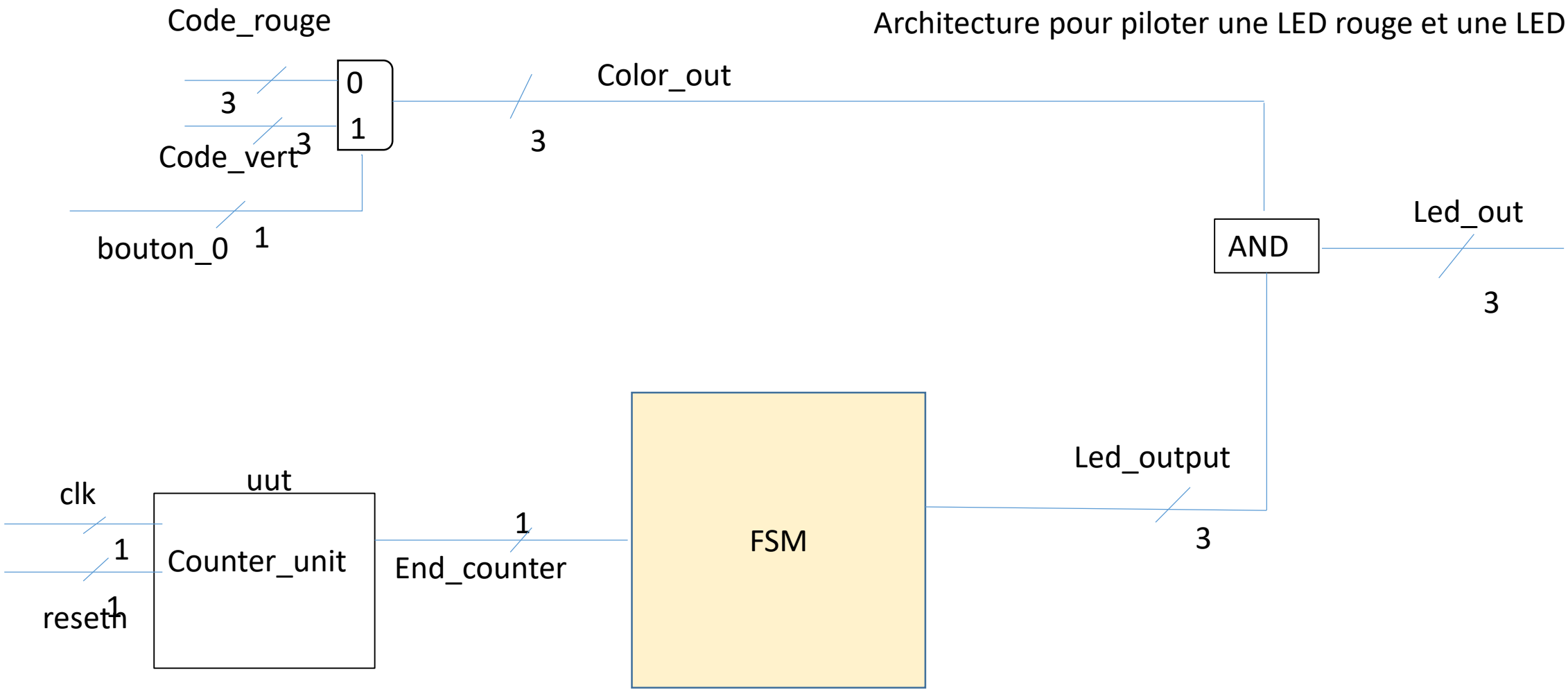


Architecture de faire clignote une LED en utilisant le module Conter_unit et une machine à état.

La LED change son état lorsque le signal end_counter = 1

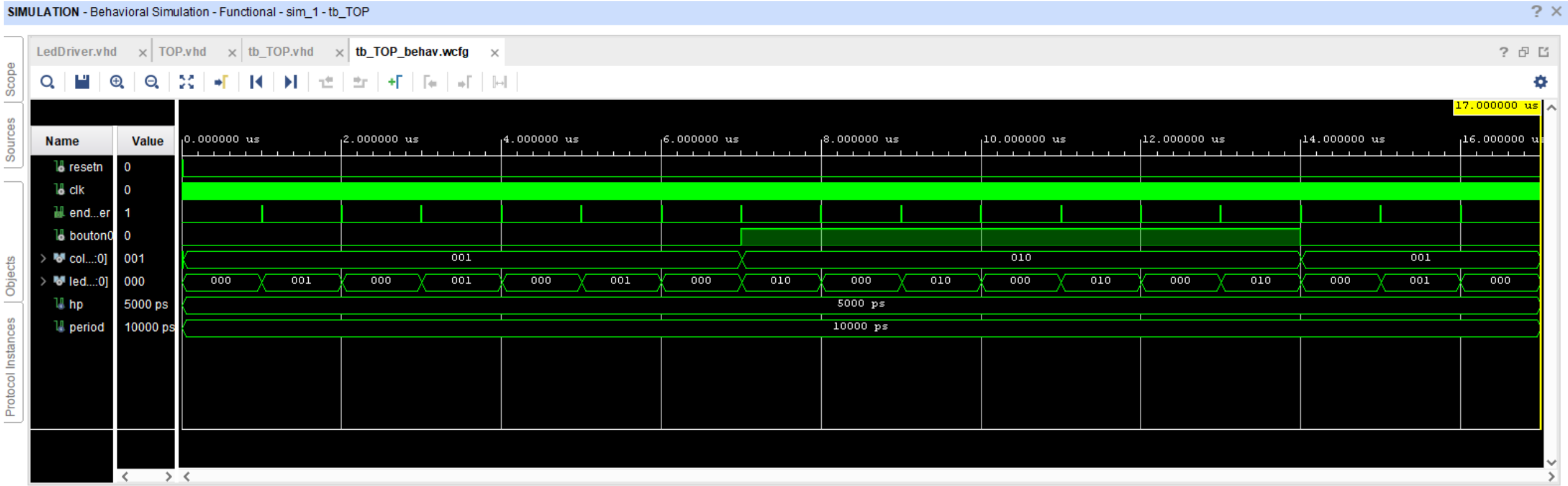


Architecture pour piloter une LED rouge et une LED verte



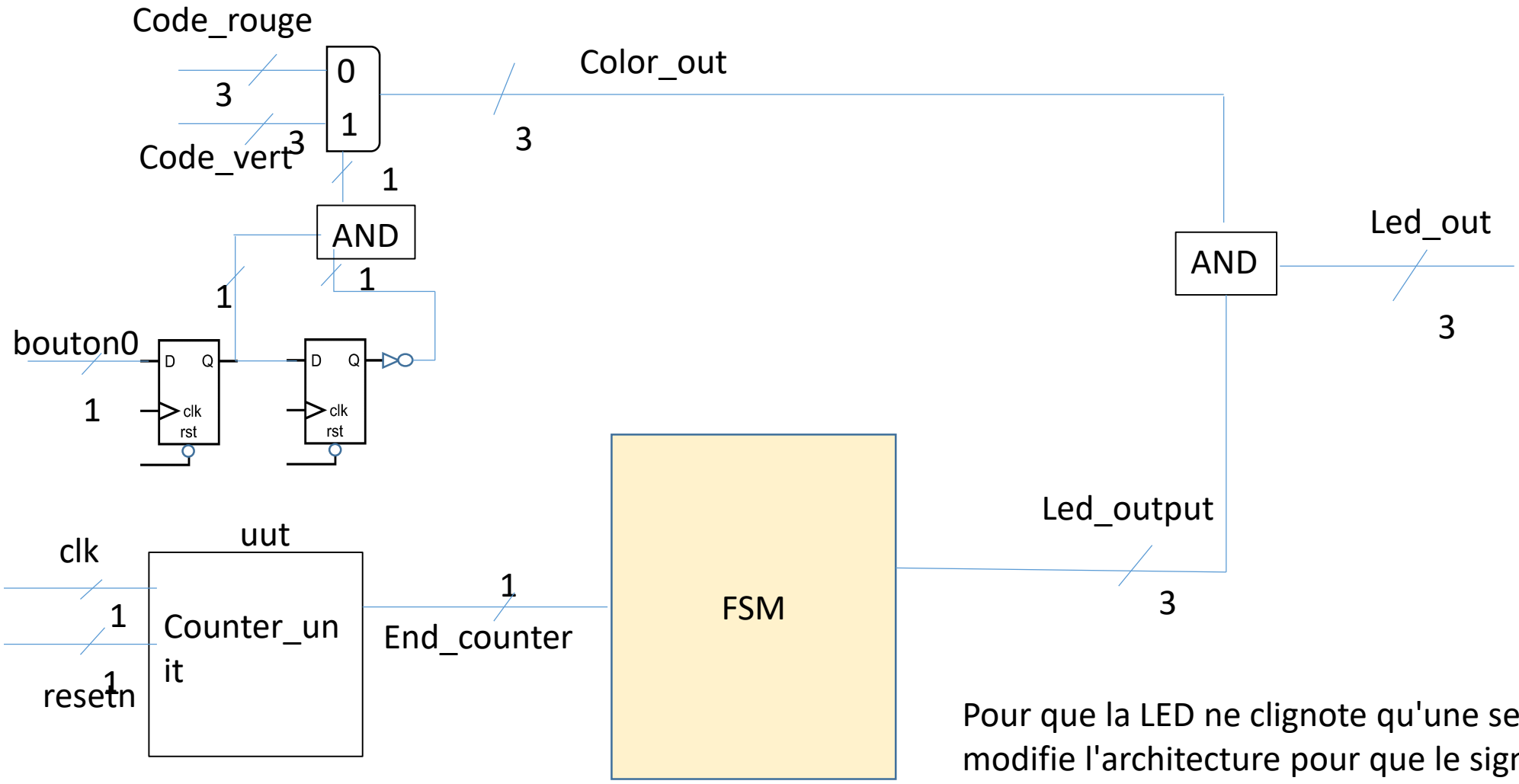
Lorsque bouton0 est appuyé, la LED verte est allumée, sinon la LED rouge est allumée. La couleur de LED est choisi à l'aide d'un multiplexeur.

Résultat de simulation avec 100 période d'horloge pour conter_unit

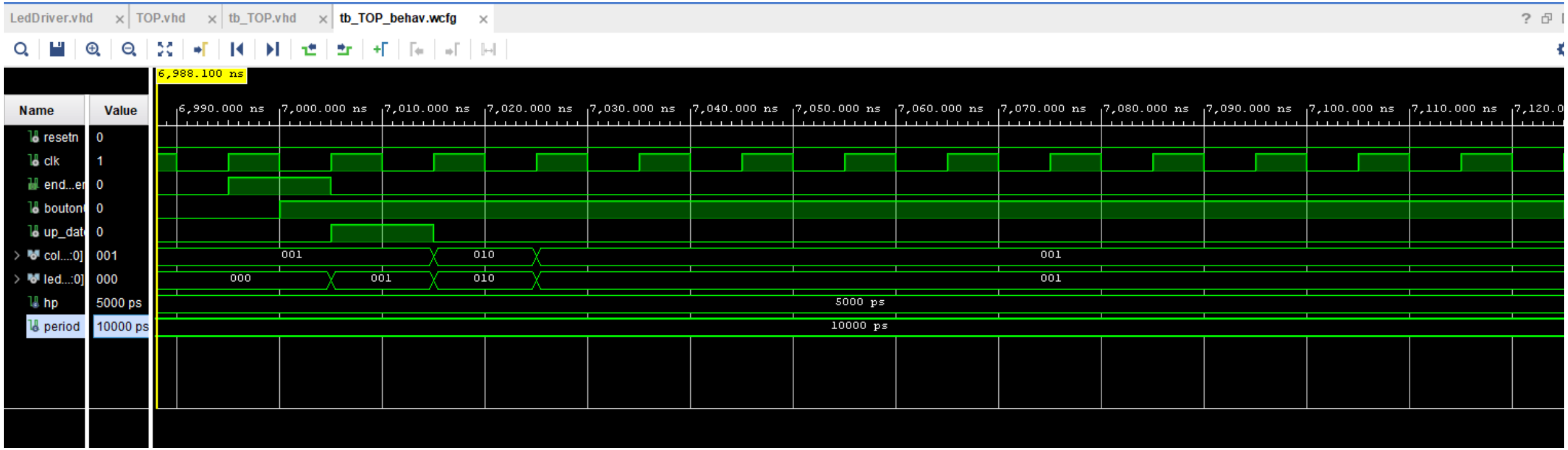


Si le bouton0 n'est pas appuyé, la LED rouge (code_rouge =« 001») clignote.
Si le bouton0 est appuyé la LED verte (code_vert =« 010») clignote.

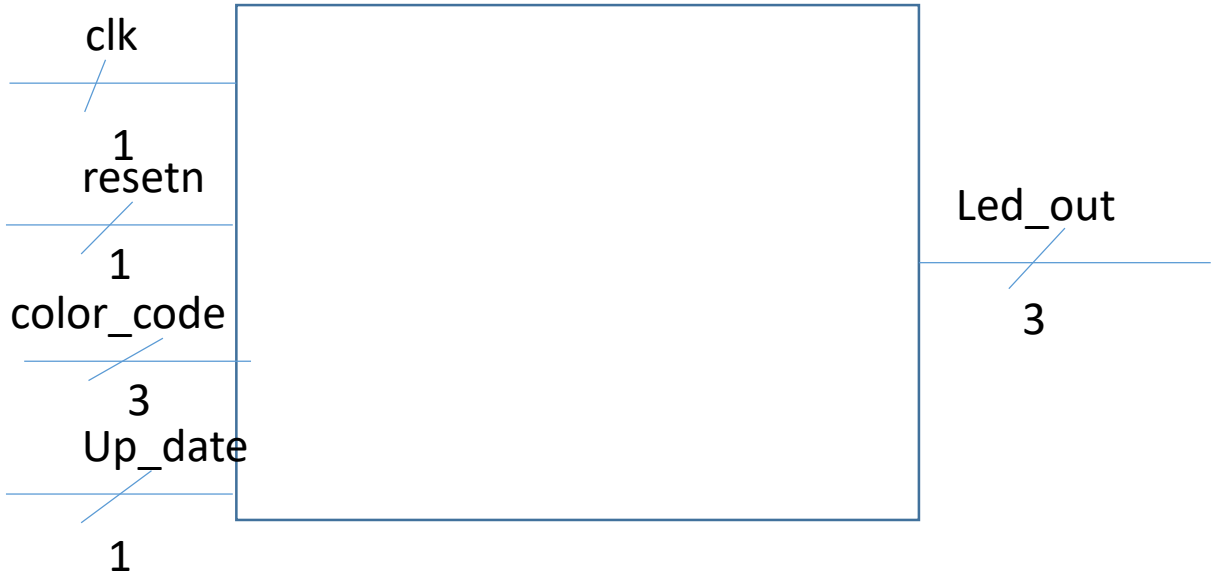
Si le bouton0 est pressé pendant plus d'un cycle d'horloge clk et plus court que 2 cycles de end_counter,
la LED vert est allumée une fois et éteinte une fois.
Si le bouton0 est maintenu pendant plusieurs cycle de end_counter, la LED vert va clignoter plusieurs fois.



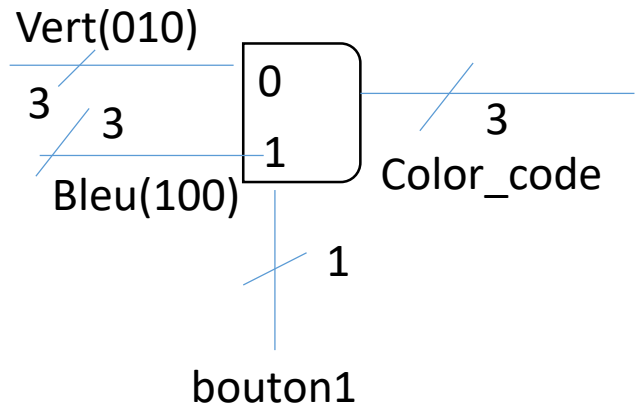
Pour que la LED ne clignote qu'une seule fois en vert, On modifie l'architecture pour que le signal contrôlant le multiplexeur dure 1 cycle d'horloge. Pour ce faire, nous utilisons 2 registres.



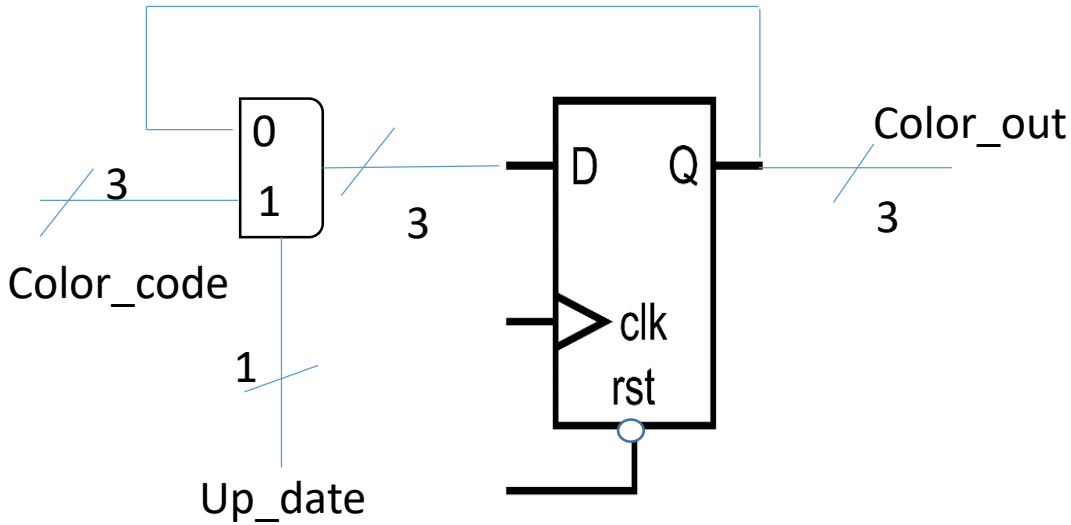
Maintenant, le bouton0 est maintenu mais la LED ne clignote qu'une seule fois en vert



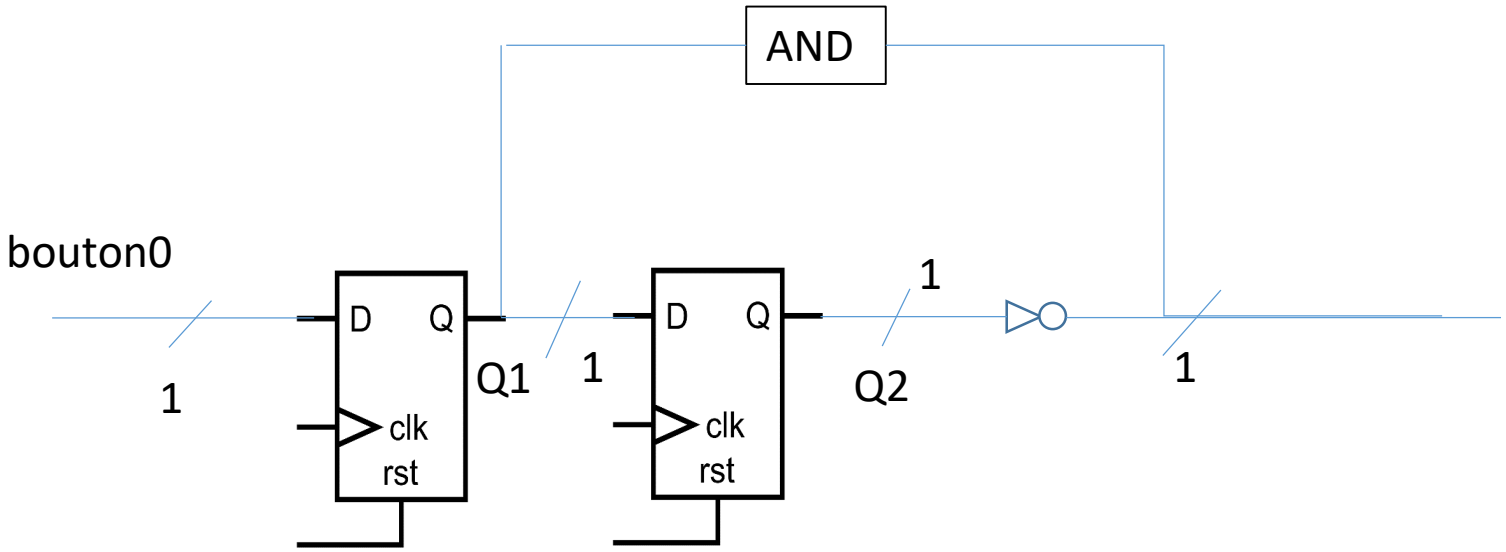
Color_code reçoit le code couler vert ou bleu à l'aide d'un multiplexeur.



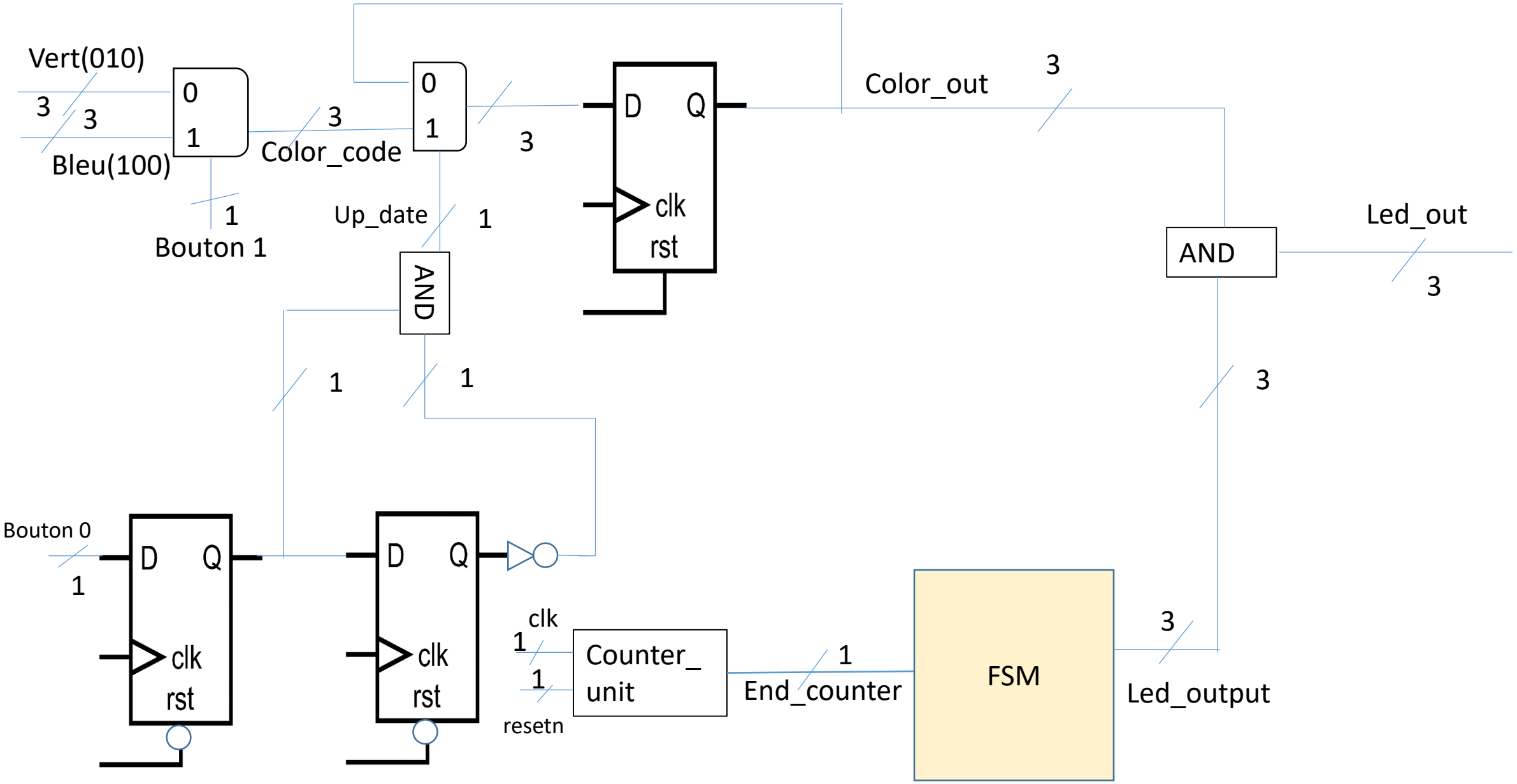
Pour changement de couleur de LED lorsque up_date=1,
Il faut 1 multiplexeur et 1 registre.



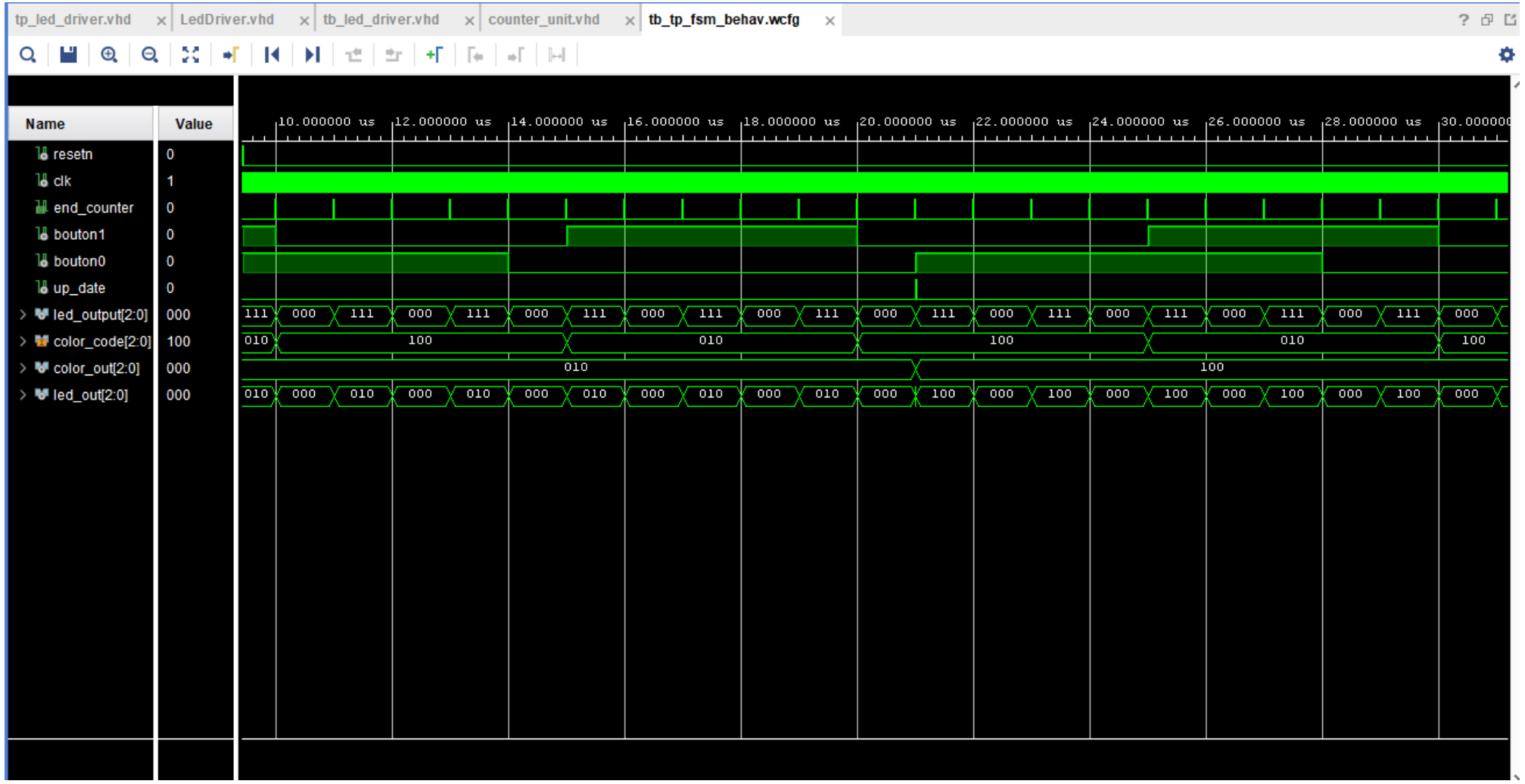
Le signal up_date dure 1 cycle d’horloge: en utilisant 2 registres.



Architecture RTL de LEDdriver avec un bouton de sélection de la couleur de la LED (bouton 1)
et un bouton de changement de la couleur de la LED (bouton 0).



Résultats de simulation de 100 cycles d’horloge clk



La LED est éteinte pendant un cycle du signal end_counter et allumée pendant le cycle suivant.
Lorsque bouton1 est pressé la LED est verte, elle est bleue sinon.
Le signal up_date dure un cycle d'horloge lorsque bouton0 vient d’etre appuyé, maintenir le bouton enfoncé ne doit pas maintenir le signal update à 1.
La LED change la colleur lorsque up_date=1

Schéma de synthèse

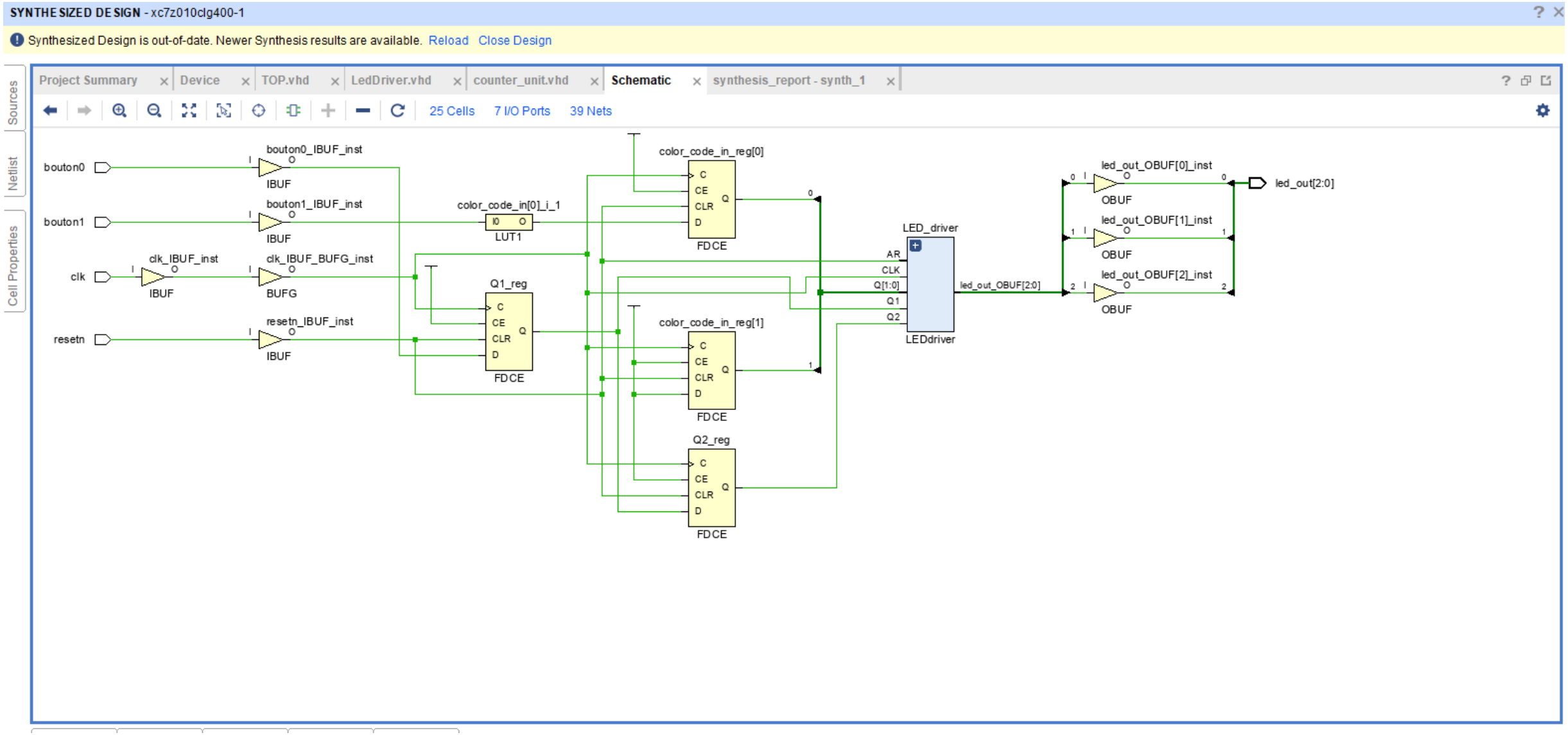
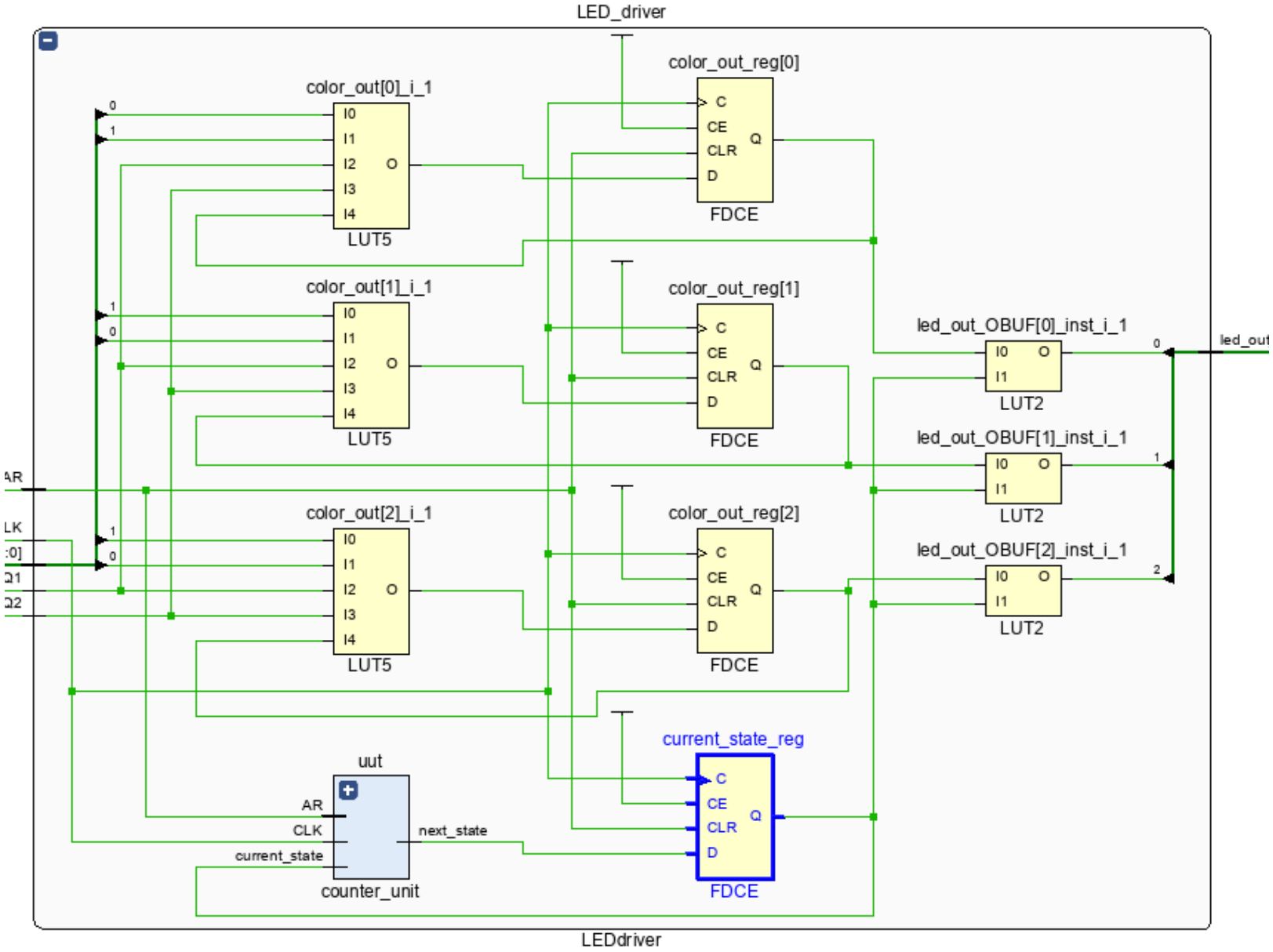


Schéma de synthèse pour LED_driver



Rapport de synthèse

Report Cell Usage:

	Cell	Count
1	BUFG	1
2	CARRY4	7
3	LUT1	1
4	LUT2	33
5	LUT4	4
6	LUT5	3
7	LUT6	3
8	FDCE	36
9	IBUF	4
10	OBUF	3

Nombre de register:

28 registres pour module conuter_unit
2 registres pour faire rentrer color_code, vert ou bleu
3 registres pour color_out
1 registres pour machine à états,
2 registres pour créer le pulse signal de up_date

Total = 36 registres

Nombre de registre est correspondent a schéma RTL

IBUF: clk, resetn, bouton0, bouton1
OBUF: led_out (3 bits)

Rapport de timming

WNS(ns)	TNS(ns)	TNS Failing Endpoints	TNS Total Endpoints	WHS(ns)	THS(ns)	THS Failing Endpoints	THS Total Endpoints	WPWS(ns)	TPWS(ns)	TPWS Failing Endpoints	TPWS Total Endpoints
4.788	0.000	0	33	0.163	0.000	0	33	4.500	0.000	0	37

Le nombre de total négative slack (TNS) est 0, le nombre de total hold slack est 0, donc il n’y a pas problème de timming.

Point départ du chemin critique
Q_reg (23) du module counter_unit

```
--------------------------------------------------------------------------------
Slack (MET) :             4.788ns  (required time - arrival time)
  Source:                 LED_driver/uut/Q_reg[23]/C
                            (rising edge-triggered cell FDCE clocked by sys_clk_pin  {rise@0.000ns fall@5.000ns period=10.000ns})
  Destination:            LED_driver/uut/Q_reg[25]/D
                            (rising edge-triggered cell FDCE clocked by sys_clk_pin  {rise@0.000ns fall@5.000ns period=10.000ns})
  Path Group:             sys_clk_pin
  Path Type:              Setup (Max at Slow Process Corner)
  Requirement:            10.000ns  (sys_clk_pin rise@10.000ns - sys_clk_pin rise@0.000ns)
  Data Path Delay:        5.217ns  (logic 2.264ns (43.394%)  route 2.953ns (56.606%))
  Logic Levels:           10  (CARRY4=7 LUT2=1 LUT6=2)
  Clock Path Skew:        -0.021ns (DCD - SCD + CPR)
    Destination Clock Delay (DCD):    4.907ns = ( 14.907 - 10.000 )
    Source Clock Delay      (SCD):    5.356ns
    Clock Pessimism Removal (CPR):    0.429ns
  Clock Uncertainty:      0.035ns  ((TSJ^2 + TIJ^2)^1/2 + DJ) / 2 + PE
    Total System Jitter     (TSJ):    0.071ns
    Total Input Jitter      (TIJ):    0.000ns
    Discrete Jitter          (DJ):    0.000ns
    Phase Error              (PE):    0.000ns
```

Point d’arrivée du chemin critique
Q_reg (25) du module counter_unit