RTL_LAB_1 BOUND FLASHER - SIMULATION

Author	Nhóm 09 – L01 Bùi Thanh Duy – 1912871 Vũ Quang Huy – 1913578 Lê Duy Hào – 2011142 Đỗ Thành Minh – 2011610 Nguyễn Ngọc Trí – 2012286
Date	20/03/2023
Version	1.2

Contents

1. Update interface

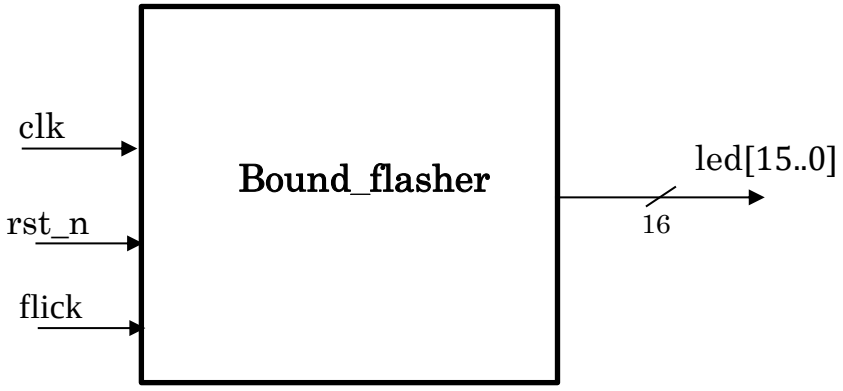


Figure 1: The figure of Bound Flasher System

Signal	Width	In/Out	Description
clk	1	In	Tín hiệu clock, kích cạnh dương
rst_n	1	In	Tín hiệu reset, tích cực mức thấp
flick	1	In	Tín hiệu đặc biệt để điều khiển trạng thái
led[15..0]	16	Out	Output led[15..0] có 16 đèn

Table 1: Description of signals in Bound Flasher

2. Update internal implementation

2.1. Overall.

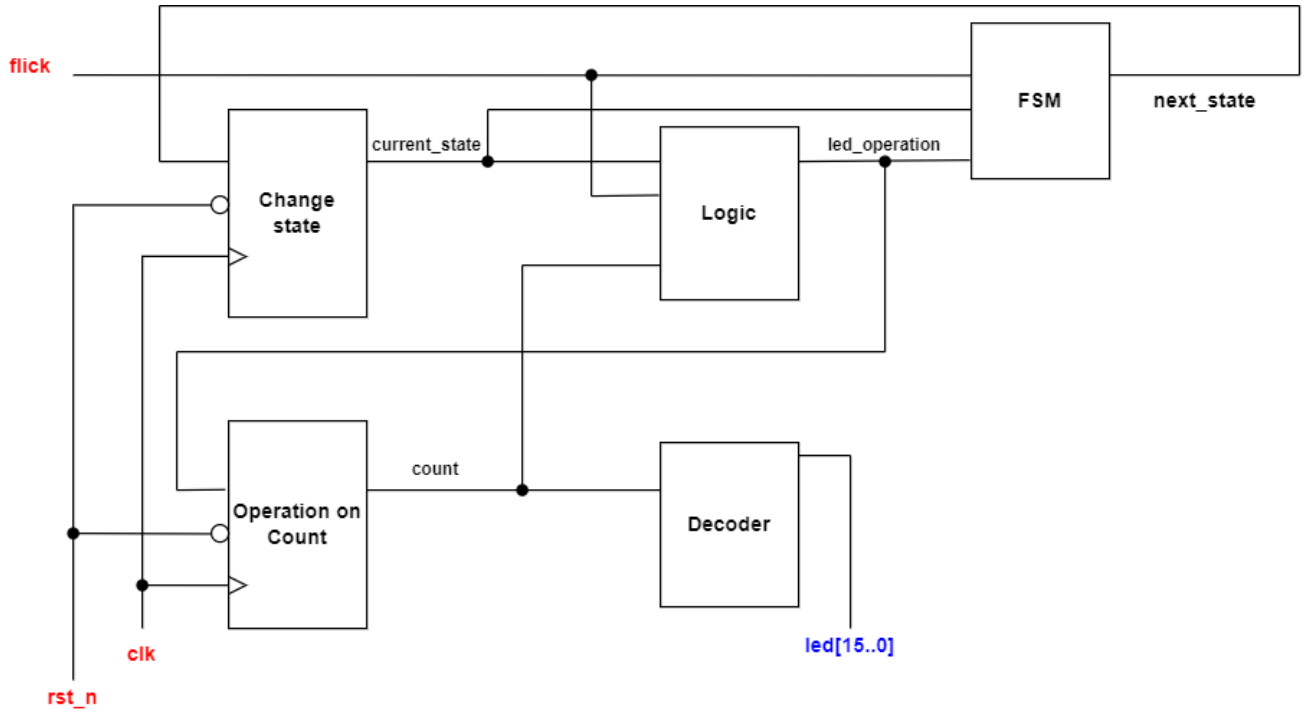


Figure 3.1: Block diagram of Bound Flasher

Name	Type	Description
rst_n	input	Tín hiệu reset
clk	input	Xung clock
flick	input	Tín hiệu đặc biệt
led[15..0]	output	Trạng thái đèn
current_state	reg	Chứa thông tin của trạng thái hiện tại
next_state	reg	Chứa thông tin của trạng thái tiếp theo
count	integer	Dùng để bật tắt đèn, quyết định đến trạng thái tiếp theo
led_operation	reg	Điều khiển count trong xung nhịp tiếp theo
Change state	DFF	Thay đổi trạng thái hiện tại đến trạng thái kế
Decoder	decoder block	Chuyển đổi tín hiệu count thành đèn LED

Operation on Count	DFF	Xác định việc tăng, giảm tín hiệu count dựa trên đầu vào
Logic	control block	Xử lí các điều kiện và chuyển tín hiệu xuống khối *Operation on count*
FSM	control block	Xác định trạng thái tiếp theo

Table 3.1: Block diagram of Bound Flasher Description

2.2. State Machine

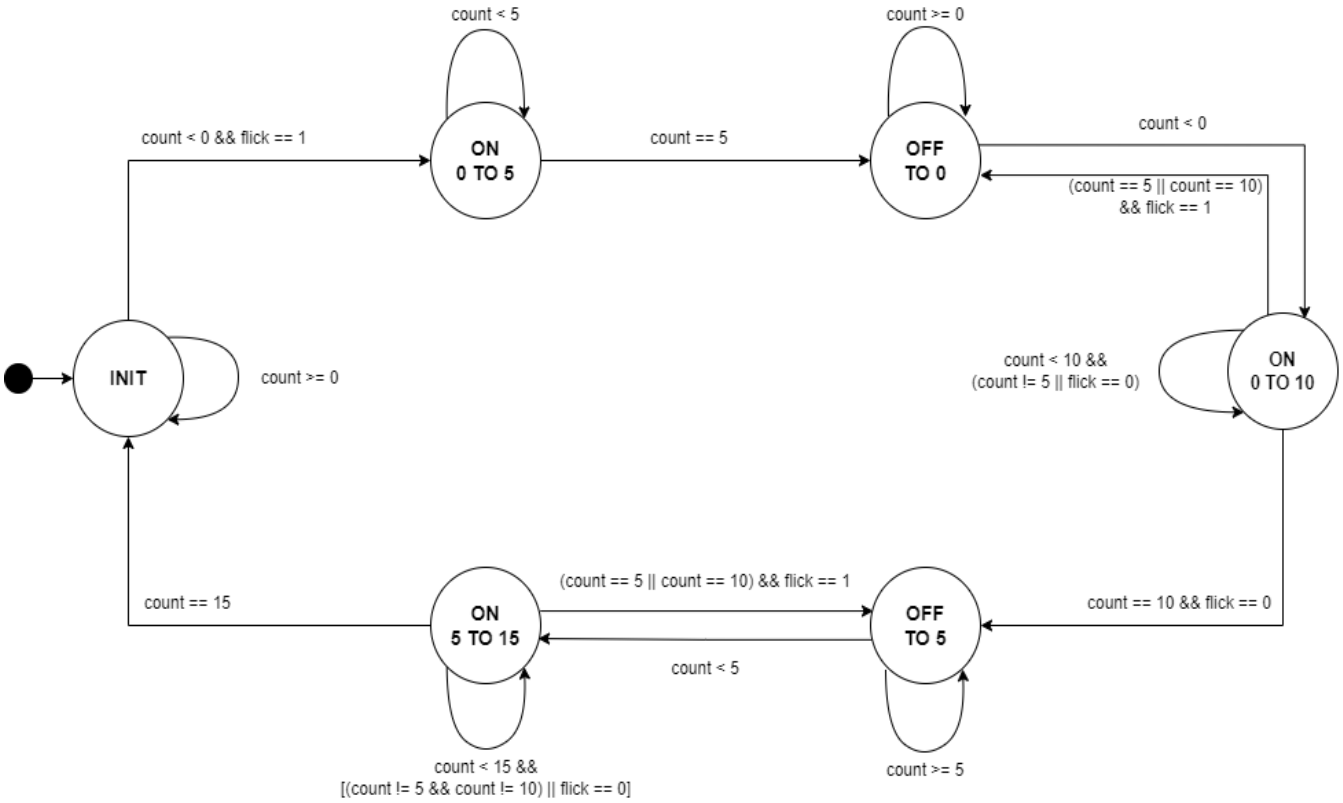


Figure 3.2: State Machine of Bound Flasher

Variable name	Description
count	Được sử dụng để nhằm theo dõi vị trí đèn LED nào được bật gần đây nhất và dùng như là điều kiện so sánh để chuyển đổi trạng thái trong FSM nếu count đang ở "kickback point" (LED[5] và LED[10]). Qua mỗi chu kỳ xung clock, biến count luôn được cộng dồn 1 đơn vị dù ở bất kỳ state nào.
LED	Dùng như một mảng chứa 16 phần tử, tương đương quản lý 16 led đơn. Mỗi phần tử có 2 giá trị là "1" và "0", quy ước cho 2 trạng thái của 1 led đơn lần lượt là "bật" và "tắt"
flick	Dùng như là một tín hiệu đầu vào để xác định trạng thái tiếp theo, nếu trạng thái hiện tại của FSM rơi vào trong 2 trường hợp: "turn_on_0_to_10" và "turn_on_5_to_15" và biến count đang ở "kickback point".

Table 3.2: Variable name of State machine

State name	Description
INIT	Đây là trạng thái đầu tiên của FSM, dùng để làm mới toàn bộ các trạng thái của 16 led đơn về mức "0", bất kể 16 led đơn đang ở trạng thái nào đi nữa: Led[count] = 0 count-- Kiểm tra điều kiện: + Nếu count < 0 và flick == 1: nhận tín hiệu flick để bắt đầu sáng dần về Led[5] (chuyển sang state "ON 0 TO 5"). + Nếu count >= 0: state init đóng vai trò tắt dần dần 16 led nếu state trước đó là "ON_5_TO_15" (ở lại state INIT).
ON 0 TO 5	Đây là trạng thái các led đơn từ vị trí 0 tới 5 của mảng 16 led (LED[0] tới LED[5]) được bật sáng dần dần qua từng chu kỳ xung clock: Led[count] = 1 count++ Kiểm tra điều kiện: + Nếu count == 5: sáng đủ đến Led[5], chuyển sang tắt dần về Led[0] (chuyển sang state "OFF_TO_0"). + Nếu count < 5: sáng chưa đủ đến Led[5], tiếp tục bật sáng dần (ở lại state "ON_0_TO_5").
OFF TO 0	Trạng thái có chức năng thực hiện hiệu ứng tắt dần dần các đèn ở vị trí count đến 0 theo chu kì xung clock bằng việc thực hiện tắt đèn mỗi ở vị trí count: Led[count] = 0 count-- Kiểm tra điều kiện: + Nếu count >= 0, quay lại trạng thái OFF_TO_0 + Nếu count < 0, chuyển sang trạng thái ON_0_TO_10
ON 0 TO 10	Trạng thái có chức năng thực hiện hiệu ứng bật đèn dần dần các đèn ở vị trí 0 đến 10 theo chu kì xung clock bằng việc thực hiện bật đèn ở mỗi vị trí count: count++ Led[count] = 1 Kiểm tra điều kiện: + Nếu count < 10 và count không phải là Kickback point hoặc plick = 0, quay lại trạng thái ON_0_TO_10 + Nếu count là Kickback point và flick = 1, chuyển về lại trạng thái OFF_TO_ 0 + Nếu count = 10 và flick = 0, chuyển sang trạng thái OFF_TO_5
OFF TO 5	Điều kiện so sánh trước khi vào trạng thái này sẽ có hai trường hợp: - *count == 10 && flick == 0* (từ trạng thái ON_0_TO_10 chuyển sang) - *(count == 5 || count == 10) && flick == 1* (từ trạng thái ON_5_TO_15 chuyển sang) Trạng thái này sẽ lần lượt tắt từ LED[count] cho đến đèn LED[5], tắt mỗi đèn

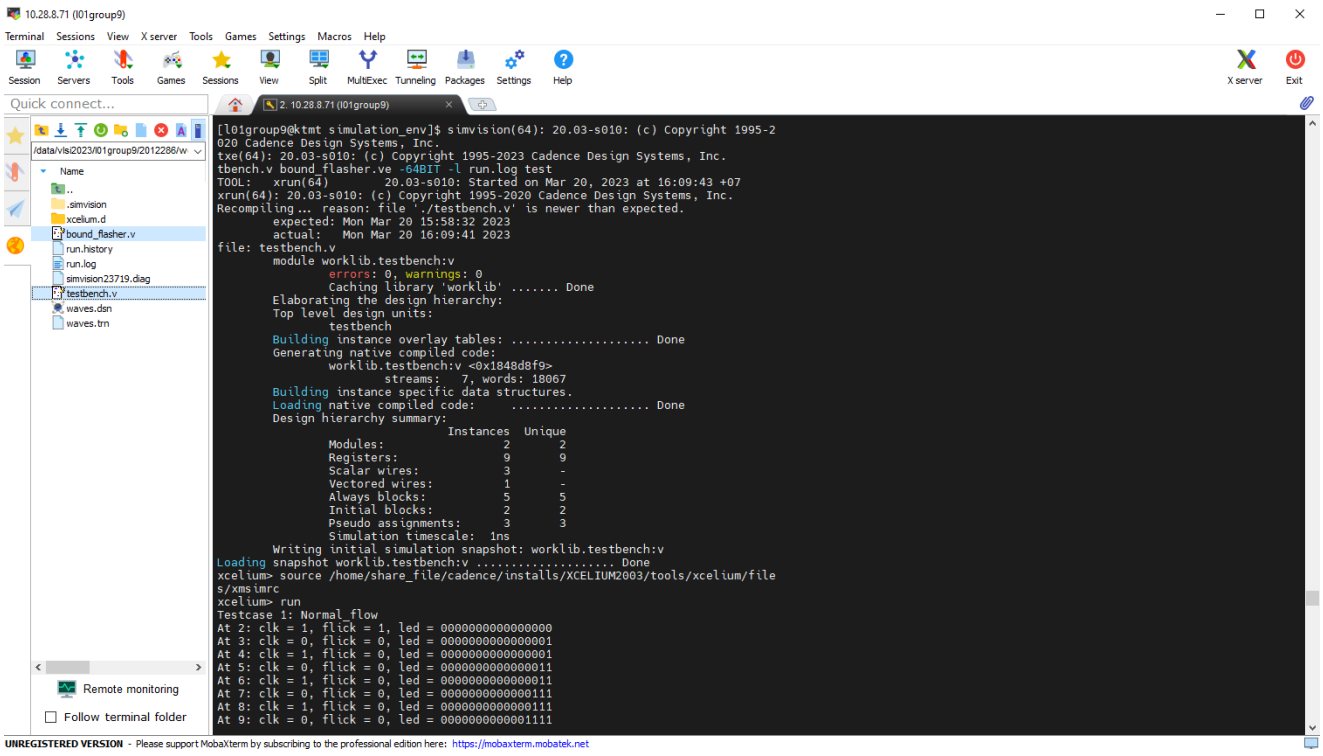
	trong một chu kì xung clock: - *LED[count] = 0*: tắt đèn thứ count - *count—* : giảm giá trị biến count để có thể tắt đèn tiếp theo nếu có) Chỉ đến khi nào count < 5 (tức là đã tắt xong đèn LED[5]) thì mới chuyển sang trạng thái tiếp theo ON_5_TO_15.
ON 5 TO 15	Điều kiện so sánh trước khi vào trạng thái này sẽ là count < 5. Trạng thái này sẽ tiến hành bật lần lượt từ đèn LED[5] đến đèn LED[15], bật mỗi đèn trong một chu kì xung clock: - *count++* : tăng giá trị biến count - *LED[count] = 0*: bật đèn thứ count Trong quá trình bật đèn nếu ở các điểm kickpoint mà biến flick được kích giá trị bằng 1 thì ngay lập tức dừng lại và chuyển sang trạng thái OFF_TO_5. Cuối cùng, nếu đã hoàn thành bật đến LED[15] thì sẽ tiến hành chuyển sang trạng thái INIT để reset lại các đèn.

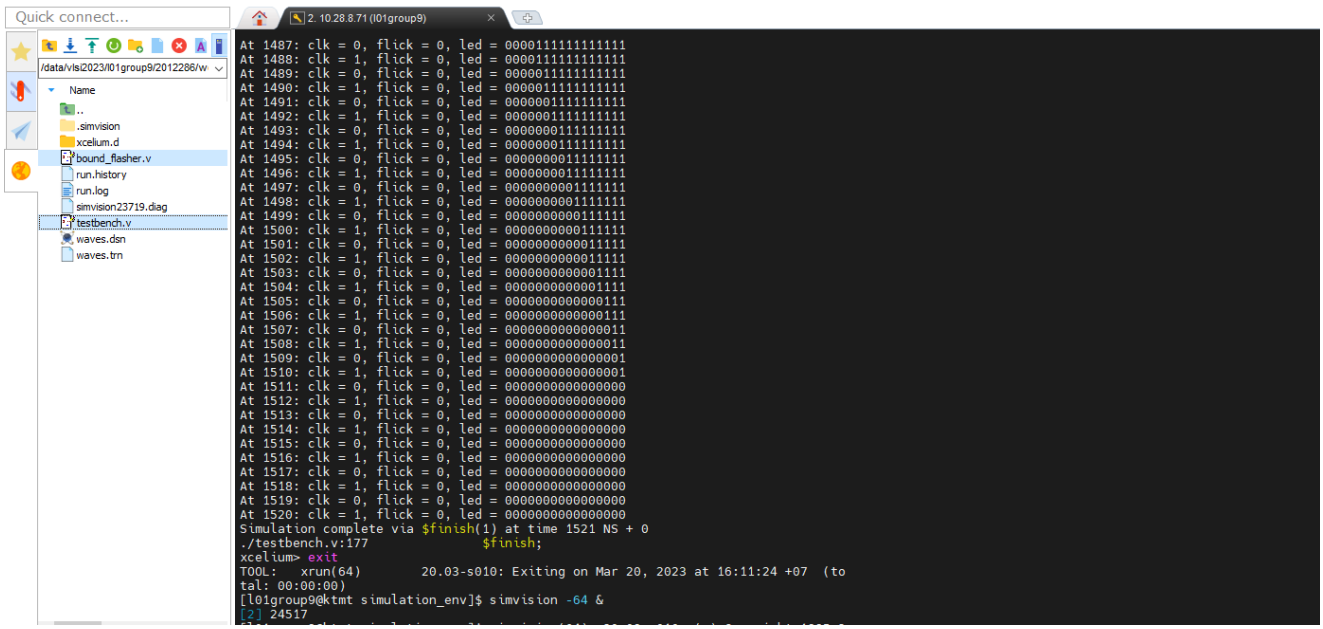
Table 3.3: State name of State machine

3. Report simulation

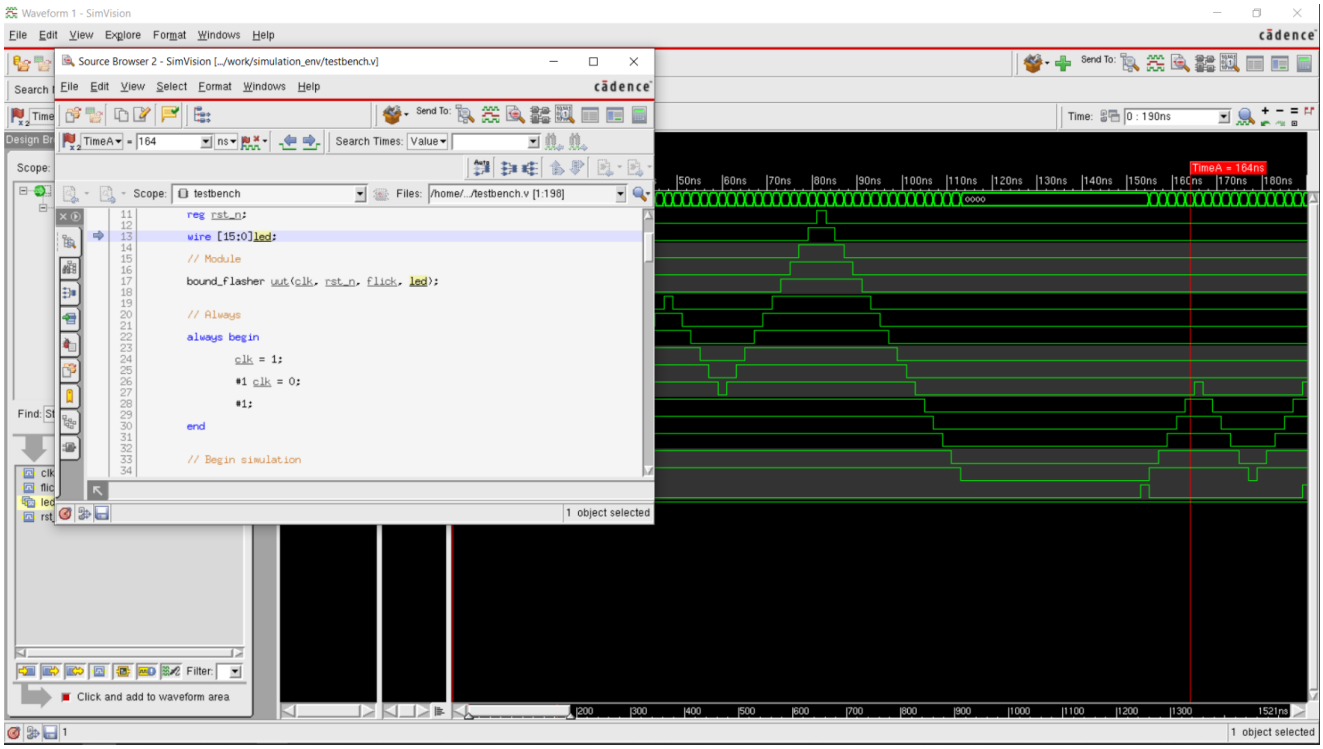
3.1. Setup

Tiến hành add 2 file code tên là ***"bound_flasher.v"*** và ***"testbench.v"*** vào folder simulation và setup mô phỏng, kết quả thu được như hình dưới đây:

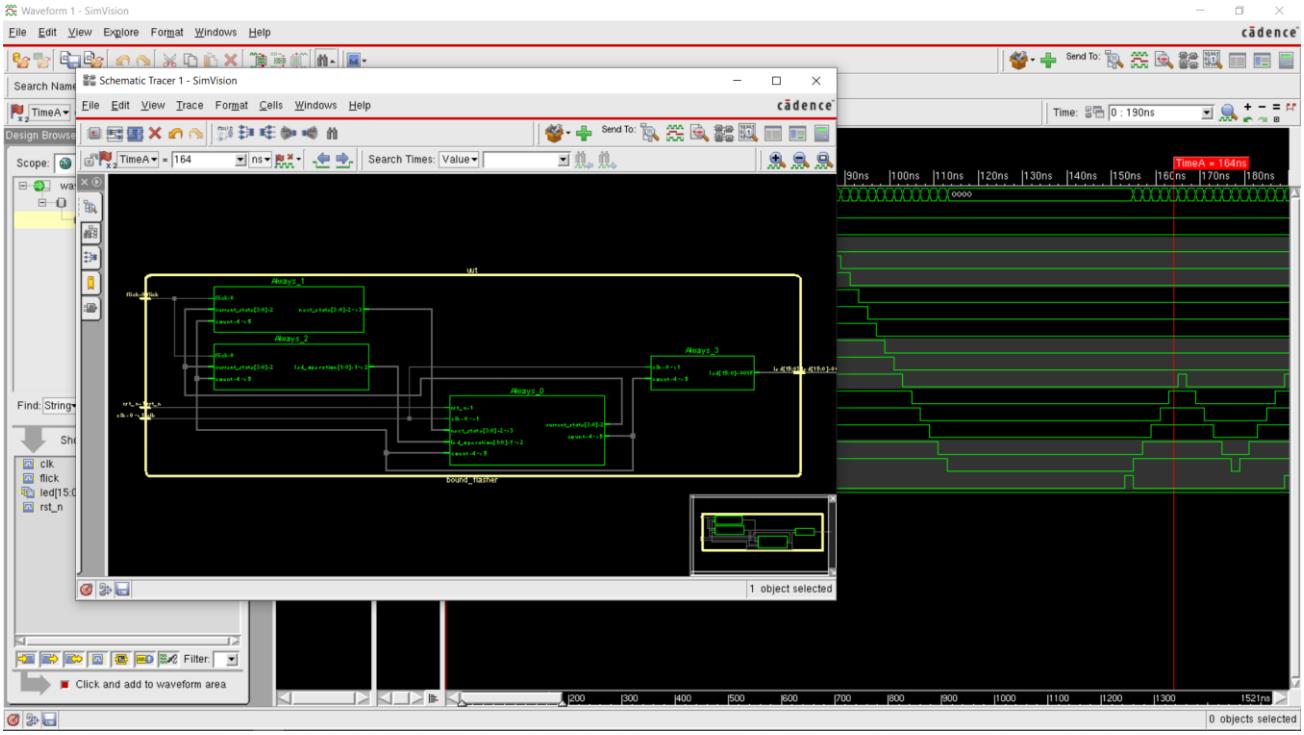




3.2. Open Source code Browser button



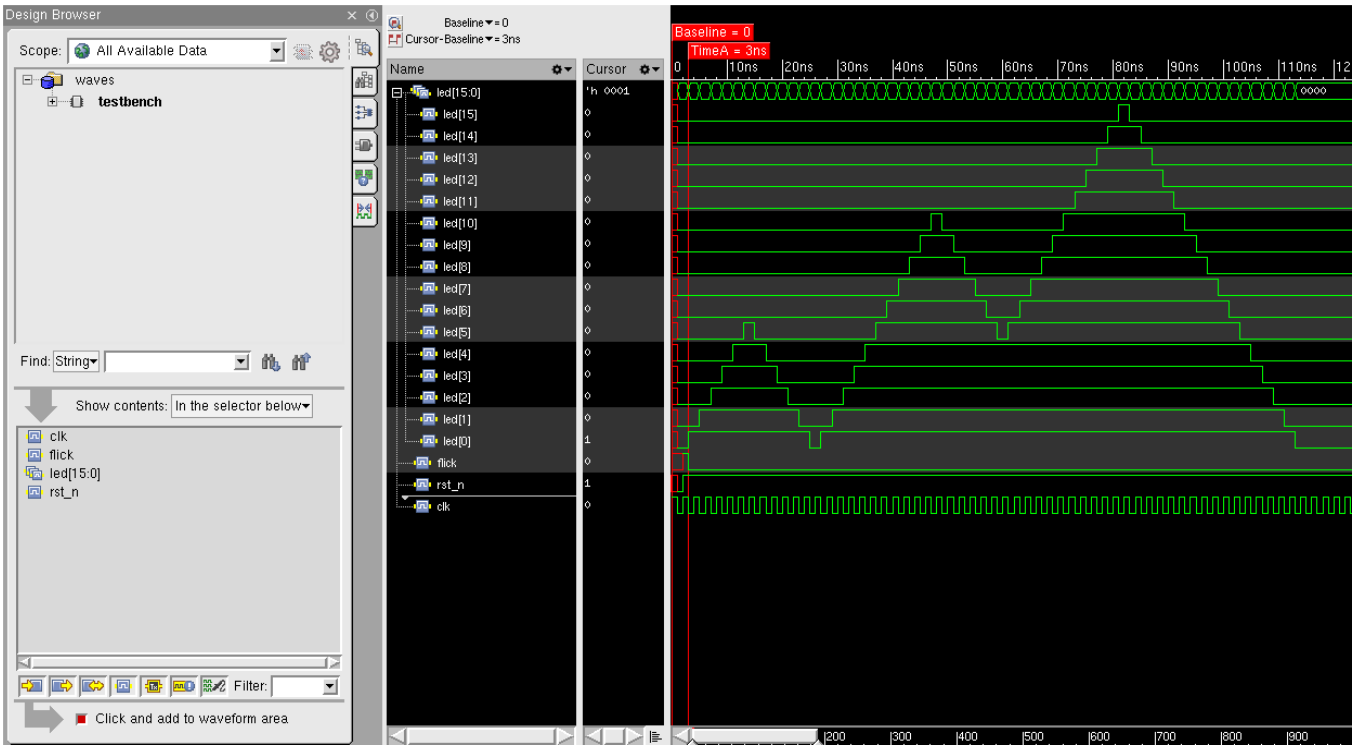
3.3. Check Schematic Trace



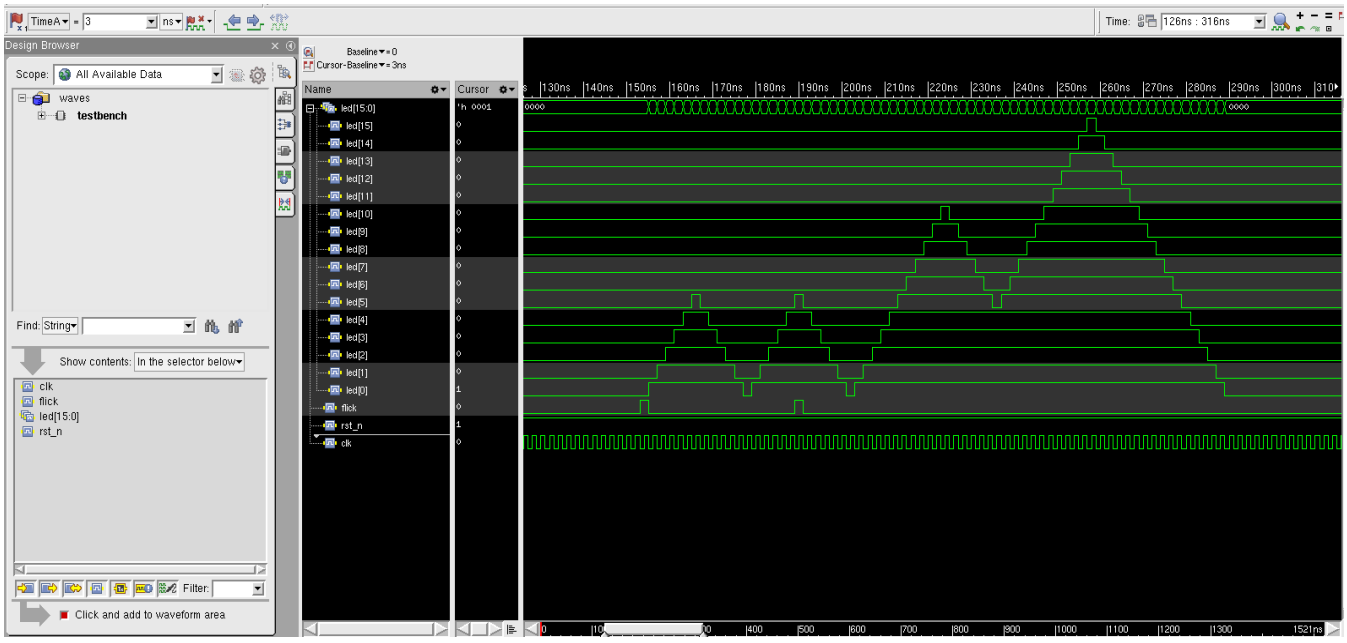
Ở sơ đồ Schematic như hình trên, so với Block Diagram mà nhóm đã lên ý tưởng trình bày thì nhóm đã hiện thực tối giản hơn.

Ở file "***testbench.v***" nhóm đã tiến hành code một số testcase tương ứng với các trường hợp có thể xảy ra ở máy trạng thái bound_flasher. Kết quả thu được như sau:

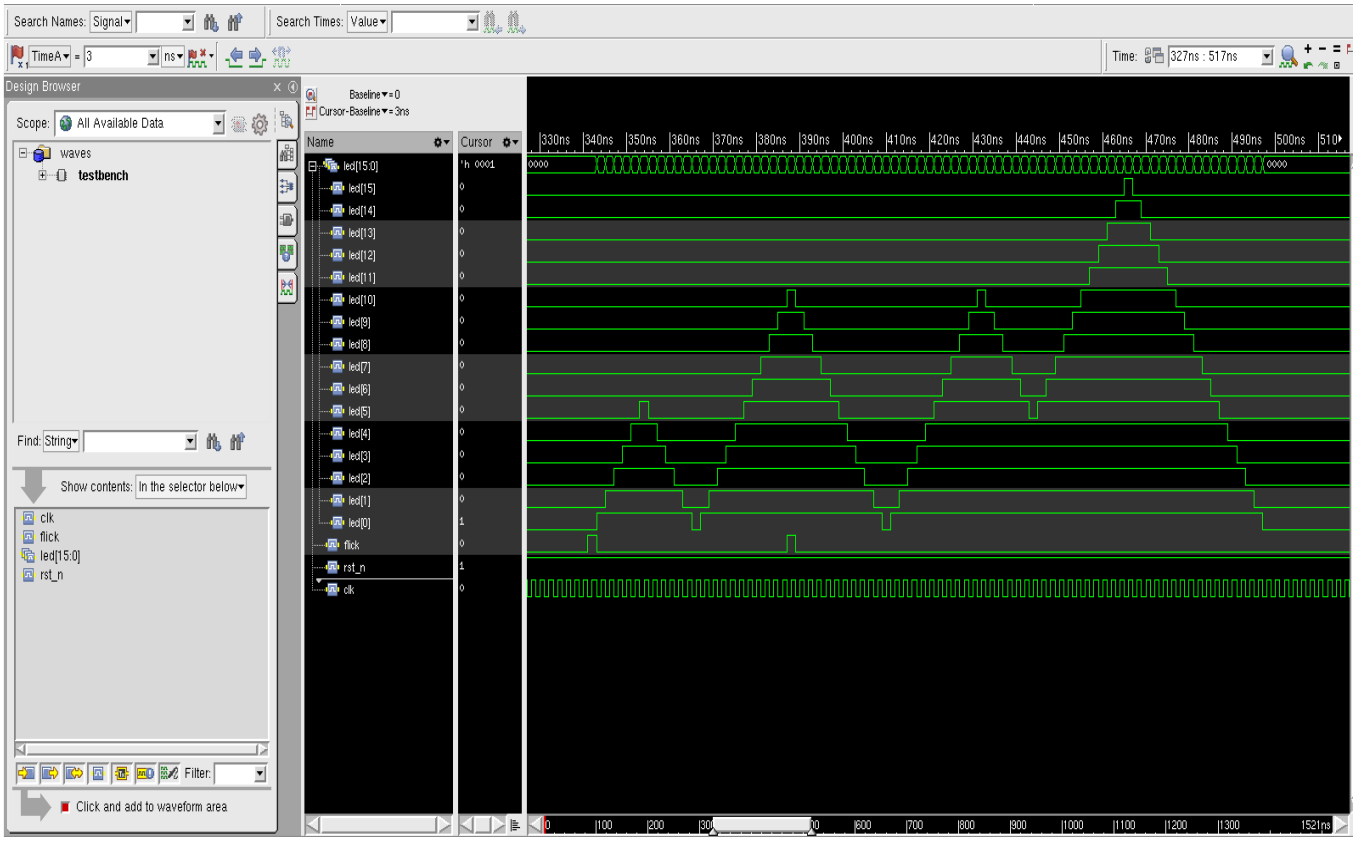
3.4. Testcase 1: Normal flow



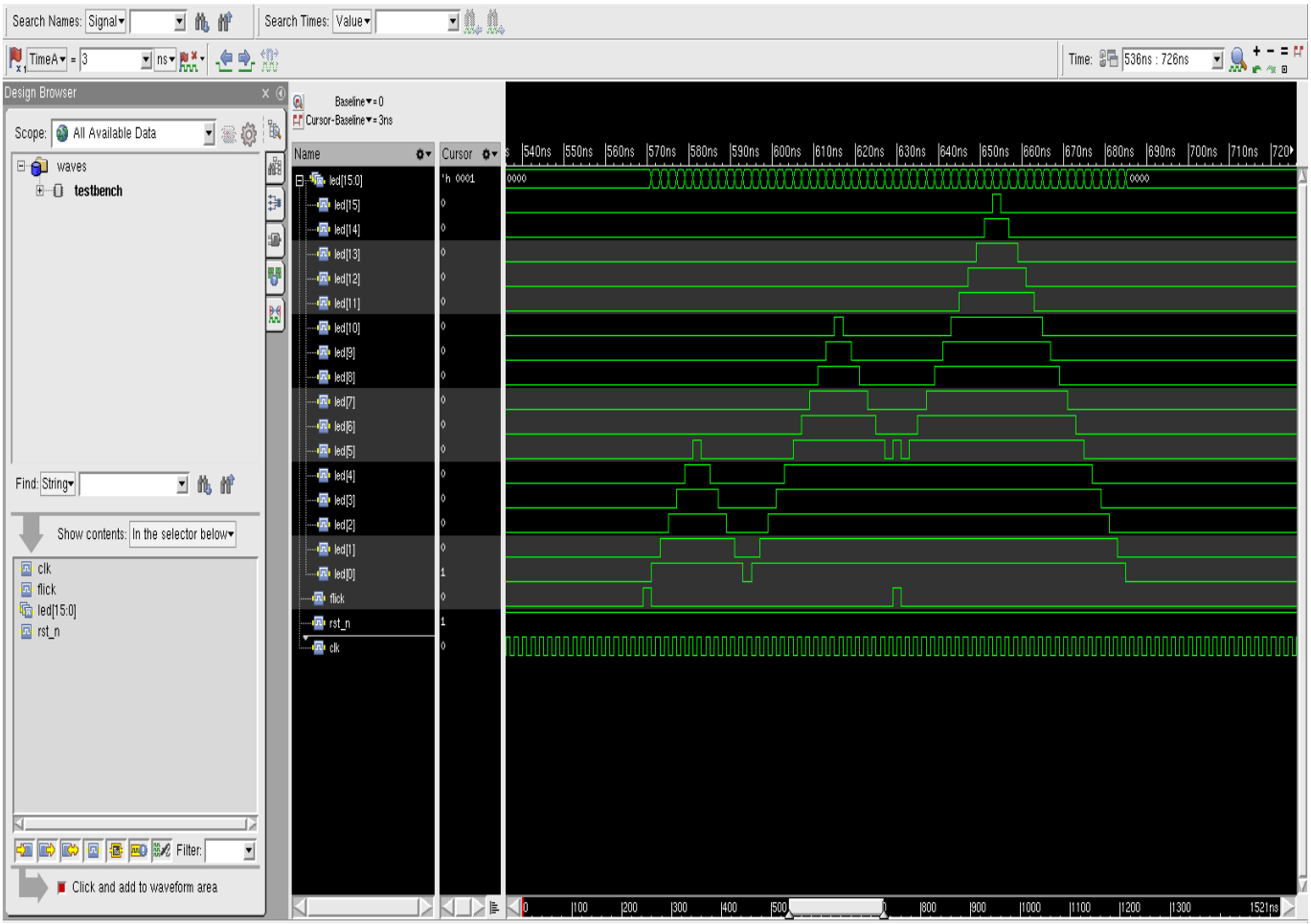
3.5. Testcase 2: Flick kickback led[5] at state ON_0_TO_10



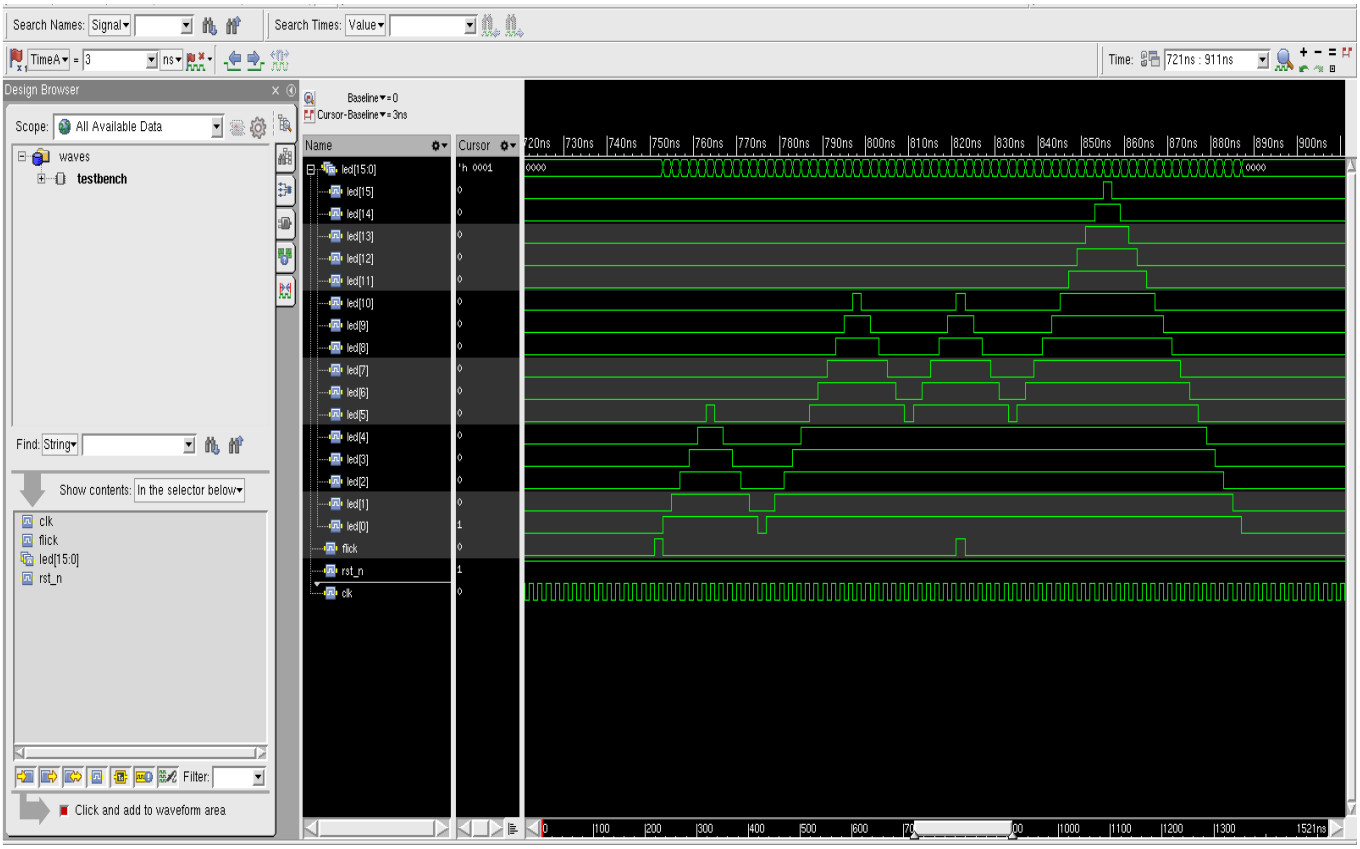
3.6. Testcase 3: Flick kickback led[10] at state ON_0_TO_10



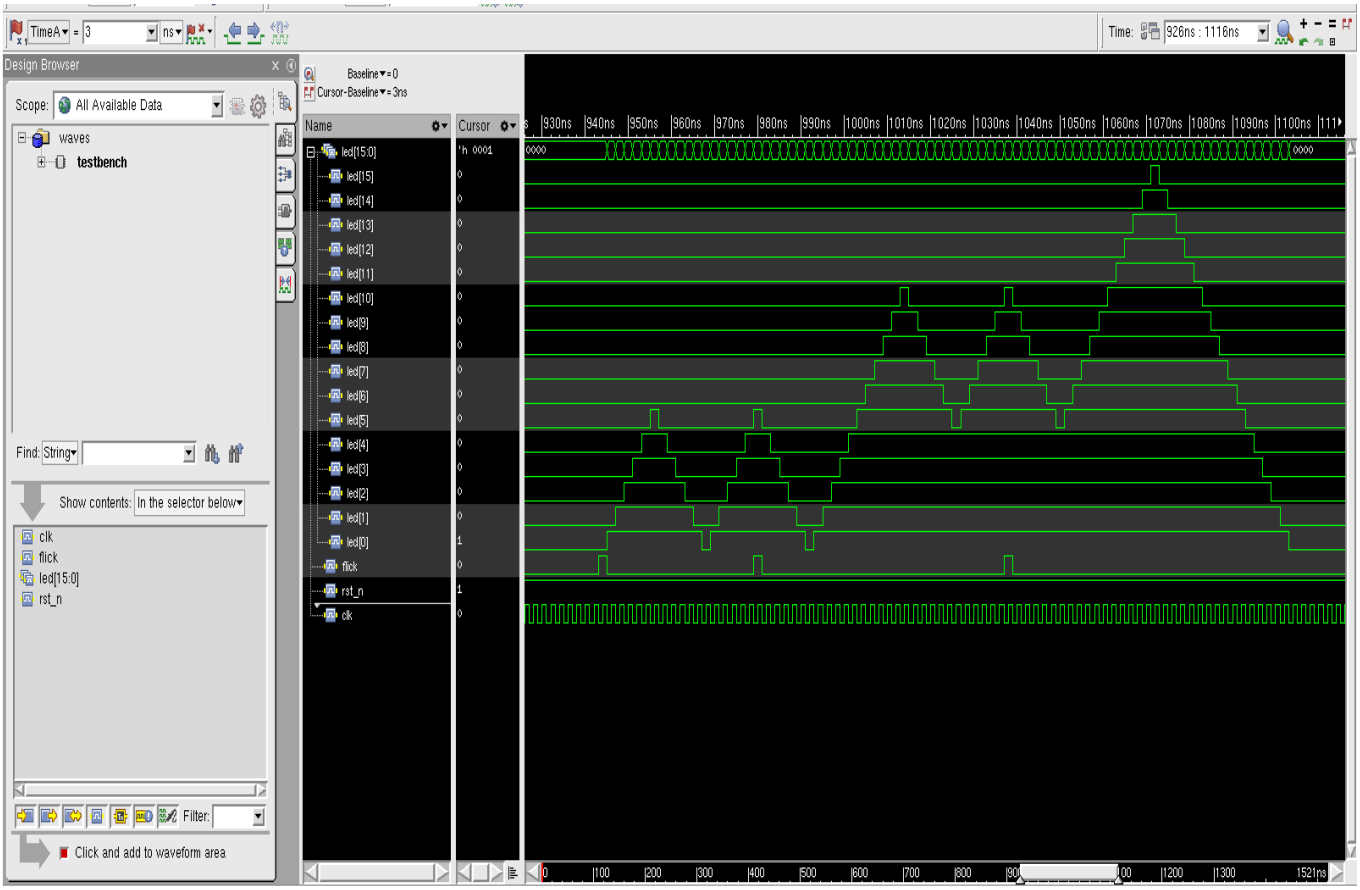
3.7. Testcase 4: Flick kickback led[5] at state ON_5_TO_15



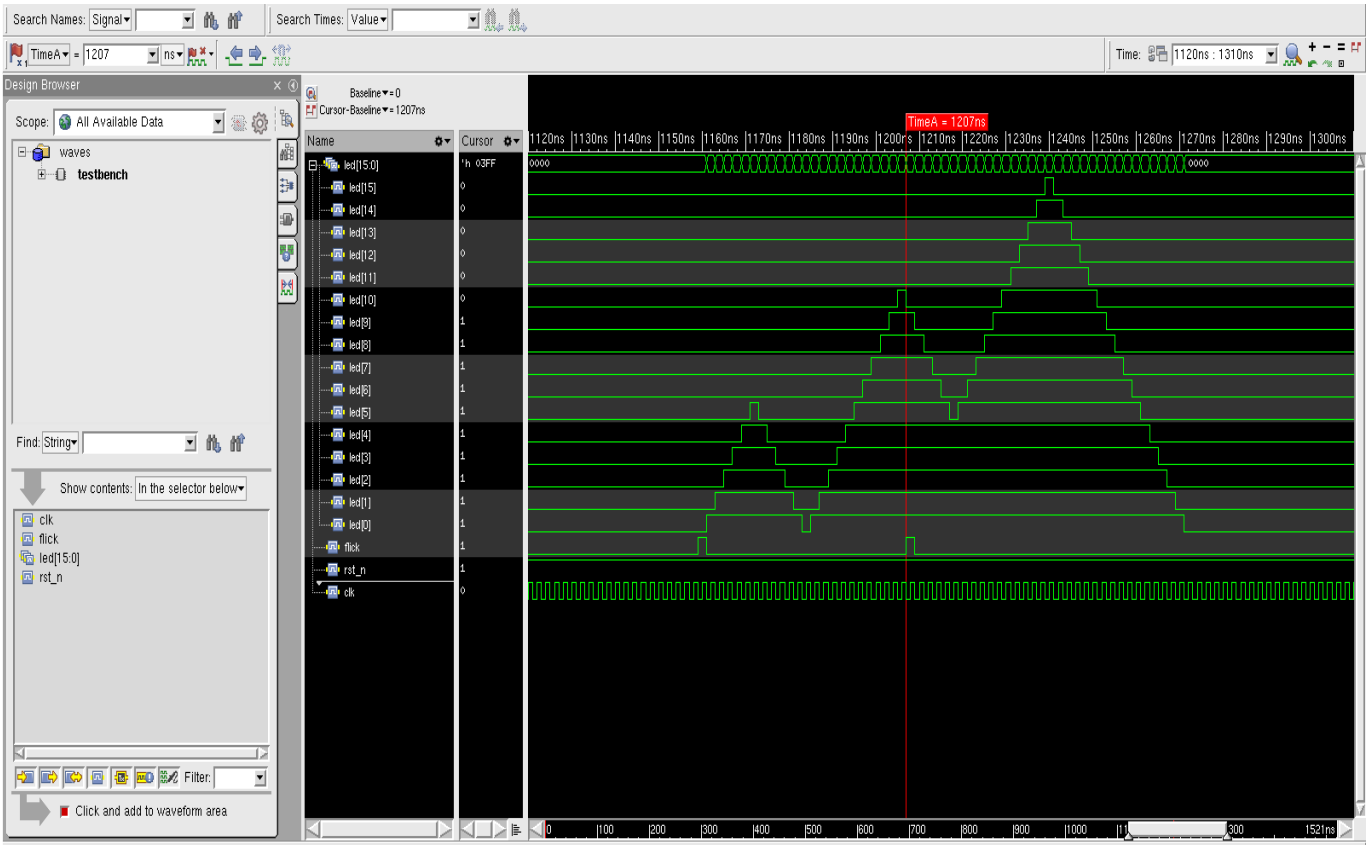
3.8. Testcase 5: Flick kickback led[10] at state ON_5_TO_15



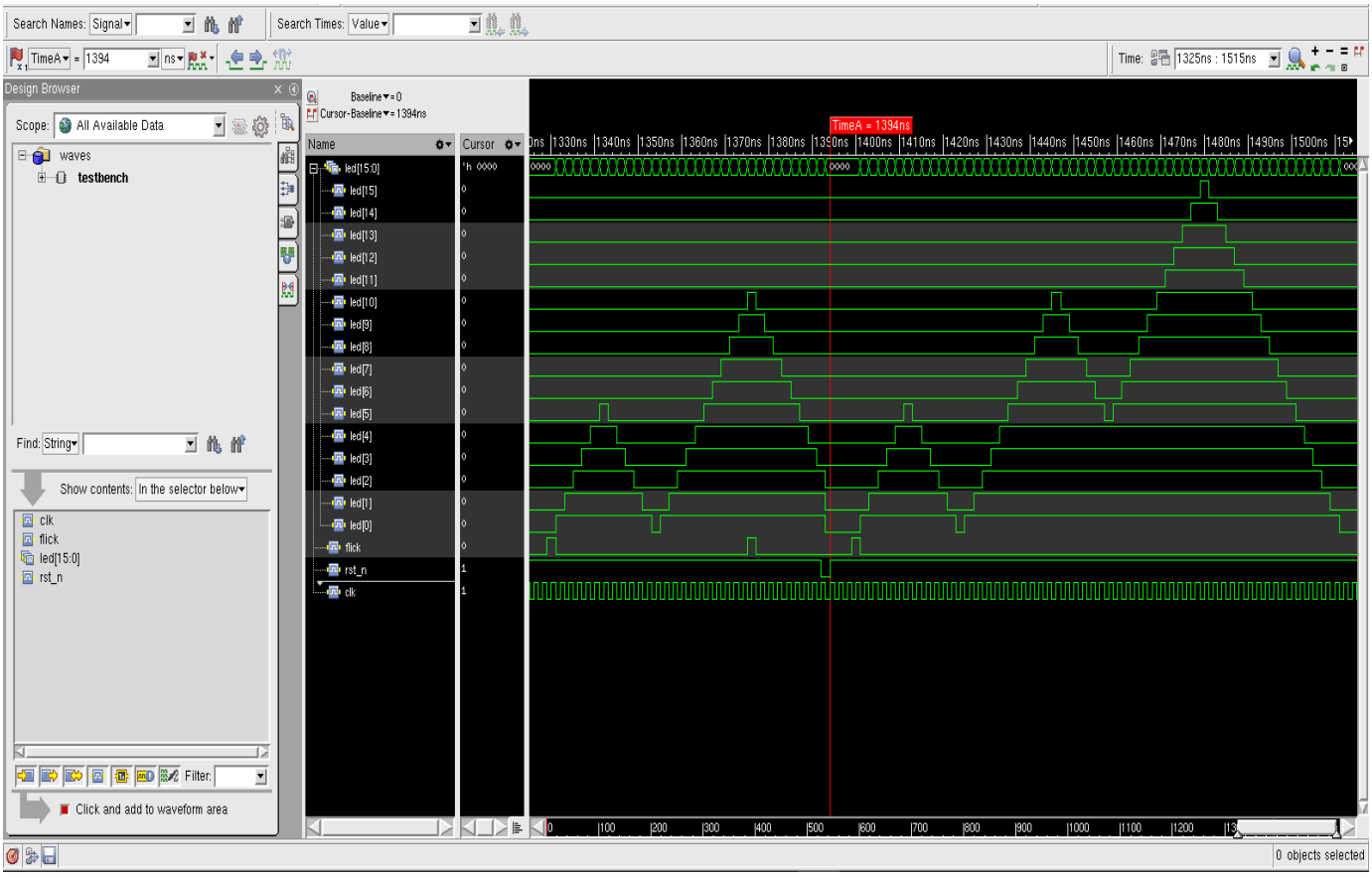
3.9. Testcase 6: Twice flick (combine testcase 2 and testcase 5)



3.10. Testcase 7: Flick at non-kickback point



3.11. Testcase 8: Check reset



4. History

Date	Author	Modified part	Description
11/03/2023	All	Code	Hiện thực code dựa trên design specification
13/03/2023	All	Interface	Cập nhật lại Interface của Bound Flasher
14/03/2023	All	Internal Implementation	Chỉnh sửa phần Overall Internal Implementation
19/03/2023	All	Simulation	Hiện thực và debug phần Simulation
20/03/2023	All	Report	Hiện thực phần báo cáo

5. Link Github Bound_Flasher

https://github.com/Tori0802/VLSI_222_Group9.git