**PROBLEM STATEMENT**

This experiment observes the behavior of a JKbar flip-flop during simulation under different delay and clock conditions. A JK flipflop with additional Reset Data (RD) and Set Data (SD) inputs is constructed from Verilog logic primitives and the sequential outputs of Q and Qbar are observed for all meaningful cases.

**KEYWORDS:** Flip-Flop, Gate Fanin, Gate Fanout, Intrinsic Delay, `define, Duty Cycle, Asynchronous Set/Reset, Propagation delay

1. **INTRODUCTION**

A JK flip-fop (FF) is a common implementation of flip-flops that is utilized in many digital devices. The general structure of this flipflop is a latch driven by an independent clock signal that will perform a bit-wise function given data inputs J and K. Flip-flops also come with Preset and Clear capabilities, which allow users to load the flip-flop with a certain value during runtime resets or initialization. A basic schematic for a JK flip-flop is shown below in Figure 2.1.

//Schematic



Figure 2.1 JK Flip-Flop Schematic

In this schematic, the flip-flop can be seen to have two latches with synchronous feedback to allow for multiple modes of operation.

One essential design factor when utilizing any component is consideration of the maximum operating frequency that the component can tolerate without data loss or corruption. To calculate this parameter, the components slowest path must be determined. In this experiment, delays were preassigned to each of the primitive gate components of the JKbar flip-flop shown in figure 2.1. These delays are shown as the fanout delay resulting from the capacitive charging and discharging delays for multiple outputs, as well as the intrinsic gate delay associated with each unique gate. These delay generalizations are recorded below in Tables 2.1 and 2.2. These delay generalizations account for signal rise, fall and hold times normally associated with such parameters.

*Table 2.1* Fanout Delay of Primitive Gates

|  |  |
| --- | --- |
| **Fanout** | **Delay** |
| 1 | 1 ns |
| 2 | 2 ns |
| 3 | 3 ns |
| Primary Output | 4 ns |

*Table 2.2* Intrinsic Gate Delay of Primitive Gates

|  |  |  |
| --- | --- | --- |
| **Gate** | **Delay Name** | **Delay (ns)** |
| Two input gates | time\_delay\_1 | 1.5 |
| Three input gates | time\_delay\_2 | 2.5 |
| Four input gates | time\_delay\_3 | 3.5 |

The JKbar flipflop is only able to run as fast as the longest path through the component. This path of greatest length is known as the critical path and is found from determining the maximum number of gates a signal must propagate through. The critical path for the JKbar flip-flop shown in this experiment is illustrated in sequence from nodes 1-7 in Fig. 2.2 below.

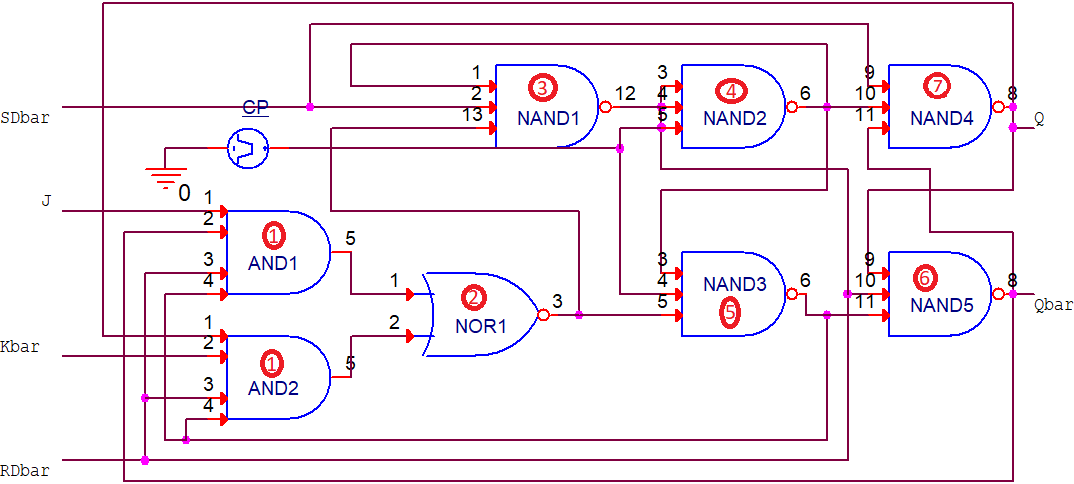


Figure 2.1 JK Flip-Flop Critical Path

The path above shows that the longest amount of time a signal can propagate is if traverses each gate in the described sequence. This critical path delay can be modeled as the sum of each of the encountered gates intrinsic delay and fanout delay. The equation for this expression is shown below in Eq. 2.1(1). The critical path delay is represented with the parameter TCP.

(1)

The calculated TCP is then found by substituting each gate delay the sum of its Fanout delay and intrinsic delay. These sums are then aggregated to calculate the intermediate value.

//CP Calculation

(2)

This TCP is representative of the minimum delay necessary to operate the circuit with integrity. An expression for the maximum operating frequency is then found using the inverse relationship described by Eq 2.1(3).

(3)

The calculated fmax is then determined for the JKbar flip-flop.

(4)

1. **METHODOLOGY**

To test this circuit, a variety of input conditions must be satisfied. In addition the behavior of the flip-flop under different clocking conditions must be observed. Specifically the flip-flop output must be confirmed to function accordingly. Then the maximum operating frequency of the circuit is iteratively determined by decreasing the cock period until a logical fault is observed for both clocking cases of a normalized clock of 50% duty cycle and a disproportionate clock of 10% duty cycle. The maximum frequency will be determined by testing each vector under each clocking condition, until either the first logical fault occurs in a vector or no logical faults occur in all vector. The maximum operating frequency will be delineated by this transition in functional state under the given conditions. Finally for each vector, each of the inputs of SDbar, RDbar, J, and K must be observed for all meaningful cases.

To meet this data burden, a behavioral Verilog module to model the JKbar flip-flop and a testbench module are constructed. In the behavioral module, the circuit is constructed form basic Verilog primitives.

To satisfy this collection of data, two Verilog modules are constructed. One module contains the declaration definition of the JKbar FF using standard Verilog code and the described primitive gates. This module must be able to handle all combinational inputs of the binary set and output a the logical values of Q and its complement Qbar. While retaining data based on the JKbar truth table located in Table 2.3 below. In addition, the behavioral model must account for the associated delays calculated above in Eq 2.1(2). This module can be seen below in section 2.3 as Module 2.1 *JKbar\_FF.v*. A testbench module must also be implemented to observe the *JKbar\_FF* module’s output given different input conditions and clocking conditions. This module is also detailed in section 2.3 and can be seen as Module 2.2 *tb\_JKbar\_FF*.v.

The testbench and behavioral modules must then be compiled using the Verilog Compile Simulator tool (VCS). If compiled with no warnings or errors, the behavioral simulation will be run and the output recorded. Figures 2.3 shows the captured behavioral waveforms of the module at the theoretical frequency. Figures 2.4 and 2.5 show the functional and nonfunctional boundaries of the operating frequency for the 50% duty cycle clocking case. Figures 2.6 and 2.7 show the same results for the 10% duty cycle clocking case. The simulation logs for the functional confirmation of the JKbar flip-flop are shown in item 2.6 of the Appendix.The logs for the clocking condition simulation files can be seen in items 2.7 and 2.8 of the Appendix for the simulation under 50% clock duty cycle and 10% clock duty cycle respectively.

*Table 2.3 JK* Flip-Flop Behavioral Modes

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ***Q*** |  |  |
| 1 | 0 | X | X | X | 1 | 0 | Asynchronous Set |
| 0 | 1 | 0 | X | X | 0 | 1 | Asynchronous Reset |
| 0 | 0 | X | X | X | 1?0 | 1?0 | Indeterminate |
| 1 | 1 | 0 | 0 | ↑ | 0 | 1 | Synchronous Reset |
| 1 | 1 | 0 | 1 | ↑ | *Q* |  | Hold |
| 1 | 1 | 1 | 0 | ↑ |  | *Q* | Toggle |
| 1 | 1 | 1 | 1 | ↑ | 1 | 0 | Synchronous Set |
| 1 | 1 | x | x |  | 0 | 1 | Synchronous Reset |
| X = Don’t Care | | ↑ = Positive Edge | |  | |  |  |

1. **MODULE FILES & SIMULATION RESULTS**

*Module 2.1*—JKbar\_FF.v

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\* \*\*\*

\*\*\* EE 526 L Experiment #X Kyle E. Keislar, Spring, 2020 \*\*\*

\*\*\* \*\*\*

\*\*\* {Experiment #X title} {Group #--} \*\*\*

\*\*\* \*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\* Filename: JKbar\_FF.v \*\*\*

\*\*\* Author: Kyle E. Keislar \*\*\*

\*\*\* Date: 01/30/2020 \*\*\*

\*\*\* Version: 1.0 \*\*\*

\*\*\* Revised: \*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\* Module Description: JK flipflop with Read Data, Send Data inputs \*\*\*

\*\*\* Outputs Q or ~Q only when SD is high \*\*\*

\*\*\* J Kbar SDbar RDbar clk Q(i+1) \*\*\*

\*\*\* X X 0 0 X X \*\*\*

\*\*\* X X 0 1 X 1 \*\*\*

\*\*\* X X 1 0 X 0 \*\*\*

\*\*\* 0 0 1 1 + 0 \*\*\*

\*\*\* 0 1 1 1 + Q(i) \*\*\*

\*\*\* 1 0 1 1 + ~Q(i) \*\*\*

\*\*\* 1 1 1 1 + 1 \*\*\*

\*\*\* 0 0 1 1 X Q(i) \*\*\*

\*\*\* Module Limitations: \*\*\*

\*\*\* \*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

`timescale 1ns **/** 10ps

//Delay definitions

`define time\_delay\_1 1.5

`define time\_delay\_2 2.5

`define time\_delay\_3 3.5

// Module Declaration

**module** JKbar\_FF**(**Q**,**

Qbar**,**

J**,** //J logical input

Kbar**,** //Kbar logical input

RDbar**,** //RDbar active low, reads data from FF

SDbar**,** //SDbar active low, Causes FF to behave as JK FF and stores value

CP**);** //clock pulse

// I/O port assignment

**output** Q**,** Qbar**;**

**input** J**,** Kbar**,** RDbar**,** SDbar**,** CP**;**

// Signal Types

//Inputs

//Internal Signals

**wire** a1**,** a2**,** nr**,** n1**,** n2**,** n3**;**

//Netlist

**and** **#(**`time\_delay\_3**+**1**)** **(**a1**,**J**,**Qbar**,**RDbar**,**n3**);** //and1

**and** **#(**`time\_delay\_3**+**1**)** **(**a2**,**Kbar**,**Q**,**RDbar**,**n3**);** //and2

**nor** **#(**`time\_delay\_1**+**2**)** **(**nr**,**a1**,**a2**);** //nor1

**nand** **#(**`time\_delay\_2**+**1**)** **(**n1**,**nr**,**SDbar**,**n2**);** //nand1

**nand** **#(**`time\_delay\_2**+**3**)** **(**n2**,**n1**,**CP**,**RDbar**);** //nand2

**nand** **#(**`time\_delay\_2**+**3**)** **(**n3**,**n2**,**CP**,**nr**);** //nand3

**nand** **#(**`time\_delay\_2**+**4**)** **(**Q**,** SDbar**,**n2**,**Qbar**);** //nand4

**nand** **#(**`time\_delay\_2**+**4**)** **(**Qbar**,** Q**,** n3**,** RDbar**);** //nand5

// Asynchronous behavior

**endmodule**

*Module 2.2*—tb\_JKbar\_FF.v

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\* \*\*\*

\*\*\* EE 526 L Experiment #2 Kyle E. Keislar, Spring, 2020 \*\*\*

\*\*\* \*\*\*

\*\*\* Structural Modeling of a JK Flip-Flop \*\*\*

\*\*\* \*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\* Filename: tb\_filename.v \*\*\*

\*\*\* Author: Kyle E. Keislar \*\*\*

\*\*\* Date: 02/11/2020 \*\*\*

\*\*\* Version: 1.0 \*\*\*

\*\*\* Revised: \*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\* Test Modules: JKbar\_FF.v \*\*\*

\*\*\* Test Strategy: Mode specific test vectors; Test vectors are chosen to \*

\*\*\* operate the FF in different testable modes of: Load 0 (reset), Load 1 \*

\*\*\* (reset), Hold & Toggle. The Asynchronous Set & Reset Conditions are \*

\*\*\* tested by setting the module inputs of RDbar and SDbar to 01 and 10 \*

\*\*\* respectively. Two asynchronous control signals are modeled with \*

\*\*\* aperodic delays to force this operating mode. The FF is initilized \*

\*\*\* to an indeterminate output mode. \*

\*\*\* \*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Timescale

`timescale 1ns **/** 10ps

//Macro Definitions

`define pulse 17.75**;** //clock pulse

`define period 35.500000**;** //clock period

`define tenth\_per 3.55**;** //1/10 of period

`define nine\_tenth\_per 31.95**;** //9/10 of period

`define runtime 1000**;** //simulation run time

// Module Instantiation

**module** tb\_JKbar\_FF**();**

//reg inputs

**reg** J**,** Kbar**,** SDbar**,** RDbar**,** CP**;**

**reg** a\_rst**,**a\_set**;** //register variables for asynchronous reset, asynchronous set and finish

//wire outputs

**wire** Q**,** Qbar**;**

JKbar\_FF UUT**(** Q**,** Qbar**,** J**,** Kbar**,** RDbar**,** SDbar**,** CP**);**

//Variables

**real** clk\_cnt**=**0**;** //counts the number of clock pulses during simulation, real number to perform frequency calcuation

**real** theo\_clk**=**`period**;**// theoretical number of clock pulses, assigned to real number to perform frequency calculation

//ASYNCH SET condition

**always**

**begin**

**#**57.5 a\_set**=**1'b1**;** //aperodic assertion

a\_rst**=**1'b0**;**

**#**2000 $display**(**"\nNULL\n"**);** //should not be reachable

**end**

//ASYNCH RESET condition

**always**

**begin**

**#**222.8 a\_set**=**1'b0**;** //aperodic assertion

a\_rst**=**1'b1**;**

**#**2000 $display**(**"\nNULL\n"**);** //should not be reachable

**end**

**always**

**begin**

**#**258.6 a\_set**=**1'b0**;** //clear a\_set, a\_rst

a\_rst**=**1'b0**;**

**#**2000 $display**(**"\nNULL\n"**);** //should not be reachable

**end**

//Initial Conditions

**initial**

CP**=**1'b0**;**//set clock to 0 at simulation start

**initial** **begin**

$vcdpluson**;** //include waveforms in simulation

//Initilization

J**=**1'b0**;** Kbar**=**1'b0**;** SDbar**=**1'b0**;** RDbar**=**1'b0**;**

a\_rst**=**1'b0**;** a\_set**=**1'b0**;**

//Print Initial conditions

**#**`period $display**(**"\n-------------------------------------|| Initial Conditions ||-------------------------------------\n"**);**

//Window to print initial conditions from Print block

**#**`period $display**(**"\n--------------------------------------------------------------------------------------------------\n"**);**

//Asynchronous Set : Test Vectors #1-4 (J, Kbar)

J**=**1'b0**;** Kbar**=**1'b0**;**//00

**#**`period J**=**1'b0**;** Kbar**=**1'b1**;**//01

**#**`period J**=**1'b1**;** Kbar**=**1'b0**;**//10

**#**`period J**=**1'b1**;** Kbar**=**1'b1**;**//11

//Asynchronous reset : Test Vectors #5-8 (J, Kbar)

**#**`period J**=**1'b0**;** Kbar**=**1'b0**;**//00

**#**`period J**=**1'b0**;** Kbar**=**1'b1**;**//01

**#**`period J**=**1'b1**;** Kbar**=**1'b0**;**//10

**#**`period J**=**1'b1**;** Kbar**=**1'b1**;**//11

//Load 0 (reset) : Test Vectors #9-10 (SDbar, RDbar, J, Kbar)

**#**`period SDbar**=**1'b1**;** RDbar**=**1'b1**;** J**=**1'b0**;** Kbar**=**1'b0**;**

**#**`period SDbar**=**1'b1**;** RDbar**=**1'b1**;** J**=**1'b0**;** Kbar**=**1'b0**;**//hold for 2 periods to propagate output

//Load 1 (set) : Test Vectors #11-12 (SDbar, RDbar, J, Kbar)

**#**`period J**=**1'b1**;** Kbar**=**1'b1**;**

**#**`period J**=**1'b1**;** Kbar**=**1'b1**;** //hold for 2 periods to propagate output

//hold (no change) : Test Vectors #13-14 (SDbar, RDbar, J, Kbar)

**#**`period J**=**1'b0**;** Kbar**=**1'b1**;** //Hold: case 01

**#**`period J**=**1'b0**;** Kbar**=**1'b1**;**//hold for 2 periods to propagate output

//Toggle : Test Vectors #15-19 (SDbar, RDbar, J, Kbar)

**#**`period J**=**1'b1**;** Kbar**=**1'b0**;** //Toggle to 0 //wait one CP to propagate output

**#**`period

**#**`period J**=**1'b1**;** Kbar**=**1'b0**;** //Toggle to 1

**#**`period

**#**`period J**=**1'b1**;** Kbar**=**1'b0**;** //Toggle to 0

**#**`period

**end**

//Asynchronous Behavior

//Async Set

**always** **@(** **posedge** a\_set**)**

**begin**

RDbar **=** 1'b1**;** SDbar **=**1'b0**;**

**end**

//Async Reset

**always** **@(** **posedge** a\_rst**)**

**begin**

RDbar **=** 1'b0**;** SDbar **=**1'b1**;**

**end**

//Synchronous Behavior

**always**

**begin**

//50% duty cycle

//#`pulse CP= 1'b1; //On for 50%

//#`pulse CP=1'b0; //Off for 50%

//10% duty cycle

**#**`tenth\_per CP**=** 1'b1**;** //On for 10%

**#**`nine\_tenth\_per CP**=**1'b0**;** //Off for 90%

**end**

//Print Block (Pr ints every clock)

**always** **@** **(posedge** CP**)**

**begin**

$display**(**"%d RDbar = %b SDbar = %b J = %b Kbar = %b Q = %b Qbar=%b"**,** $time**,** RDbar**,**SDbar**,**J**,**Kbar**,** Q**,** Qbar**);**

clk\_cnt**=**clk\_cnt**+**1.0000**;**

**end**

//Check starting values & Finish the simulation at runtime

**always**

**begin**

$display**(**"%d RDbar = %b SDbar = %b J = %b Kbar = %b Q = %b Qbar=%b"**,** $time**,** RDbar**,**SDbar**,**J**,**Kbar**,** Q**,** Qbar**);** //time 0 conditions

**#**`runtime

$display**(**"Theoretical Clocks: %d Clocks Tracked: %d \n Theoretical T= %f ns Exp T= %f ns \n"**,(**$time**/**35.5**),**clk\_cnt**,**35.5**,(**$time**/**clk\_cnt**));**

**#**10 $display**(**"Theoretical Max f= %f GHz Simulated Max f=%f GHz"**,(**theo\_clk**/**$time**),(**clk\_cnt**/**$stime**));**

$finish**;**

**end**

**endmodule**

*Table 2.4* Tabulated Experimental Output for the JKbar Flip-Flop 50% Duty Cycle @ >fmax\_theo

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Vector #** |  |  |  |  | ***Q*** |  |
| IC | 0 | 0 | 0 | 0 | X | X |
| 1 | 1 | 0 | 0 | 0 | 1 | 0 |
| 2 | 1 | 0 | 0 | 1 | 1 | 0 |
| 3 | 1 | 0 | 1 | 0 | 1 | 0 |
| 4 | 1 | 0 | 1 | 1 | 1 | 0 |
| 5 | 0 | 1 | 0 | 0 | 1 | 0 |
| 6 | 0 | 1 | 0 | 1 | 0 | 1 |
| 7 | 0 | 1 | 1 | 0 | 0 | 1 |
| 8 | 0 | 1 | 1 | 1 | 0 | 1 |
| 9 | 1 | 1 | 0 | 0 | 0 | 1 |
| 10 | 1 | 1 | 0 | 0 | 0 | 1 |
| 11 | 1 | 1 | 1 | 1 | 0 | 1 |
| 12 | 1 | 1 | 1 | 1 | 1 | 0 |
| 13 | 1 | 1 | 0 | 1 | 1 | 0 |
| 14 | 1 | 1 | 0 | 1 | 1 | 0 |
| 15 | 1 | 1 | 1 | 0 | 1 | 0 |
| 16 | 1 | 1 | 1 | 0 | 1 | 0 |
| 17 | 1 | 1 | 1 | 0 | 0 | 1 |
| 18 | 1 | 1 | 1 | 0 | 1 | 0 |
| 19 | 1 | 1 | 1 | 0 | 0 | 1 |
| -- | 1 | 1 | 1 | 0 | 1 | 0 |
| -- | 1 | 1 | 1 | 0 | 0 | 1 |
| -- | 1 | 1 | 1 | 0 | 1 | 0 |
| -- | 1 | 1 | 1 | 0 | 0 | 1 |
| -- | 1 | 1 | 1 | 0 | 1 | 0 |
| -- | 1 | 1 | 1 | 0 | 0 | 1 |
| -- | 1 | 1 | 1 | 0 | 1 | 0 |
| -- | 1 | 1 | 1 | 0 | 0 | 1 |
| -- |  |  |  |  |  |  |

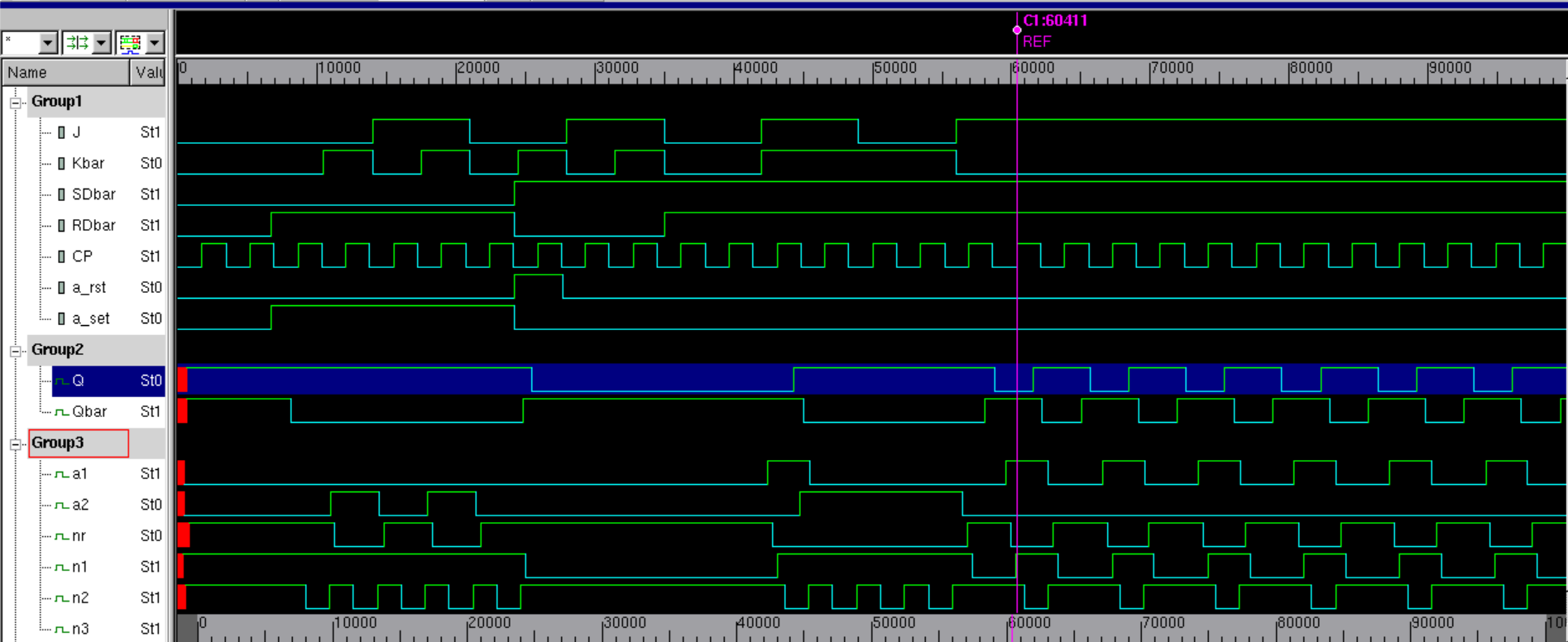
****

Figure 2.3 JK Behavioral Waveforms @ ftheo Duty Cycle

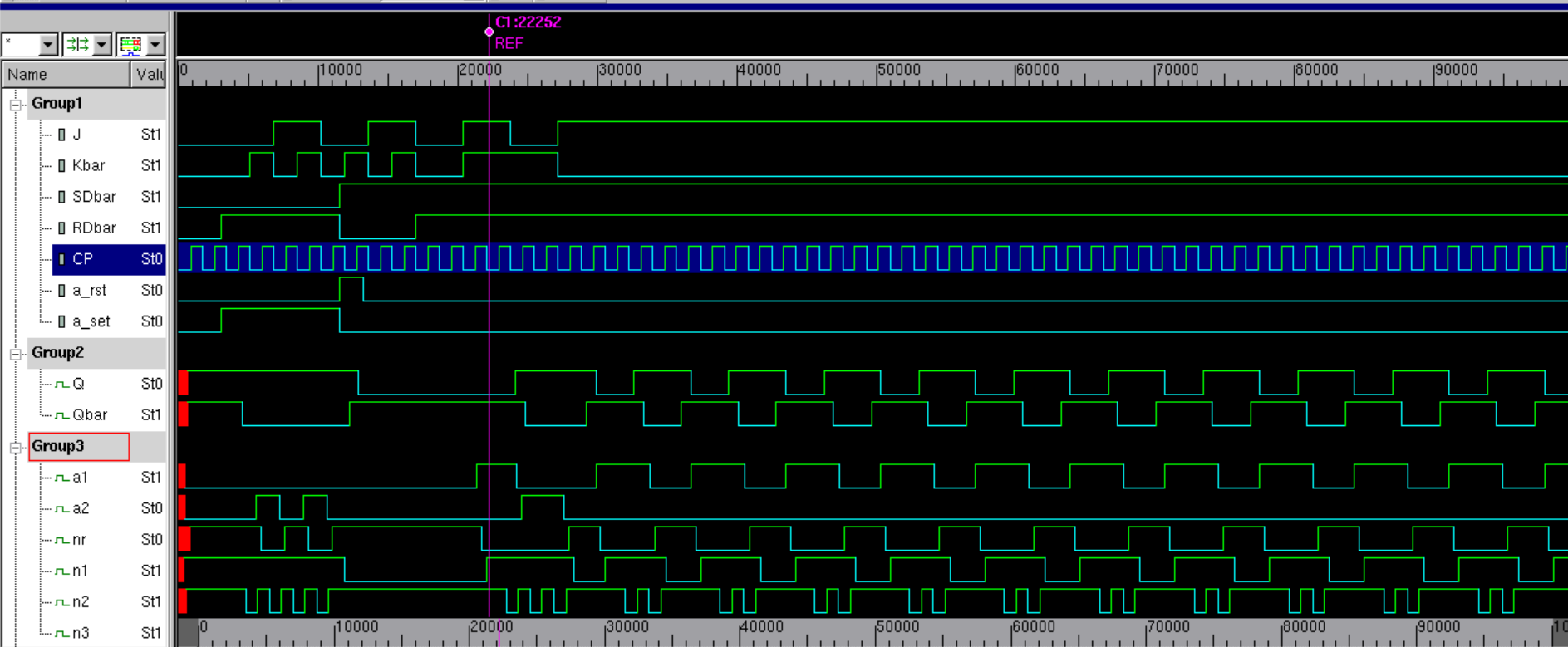
****

Figure 2.4 JK Behavioral Waveforms: 50% Duty Cycle Functional

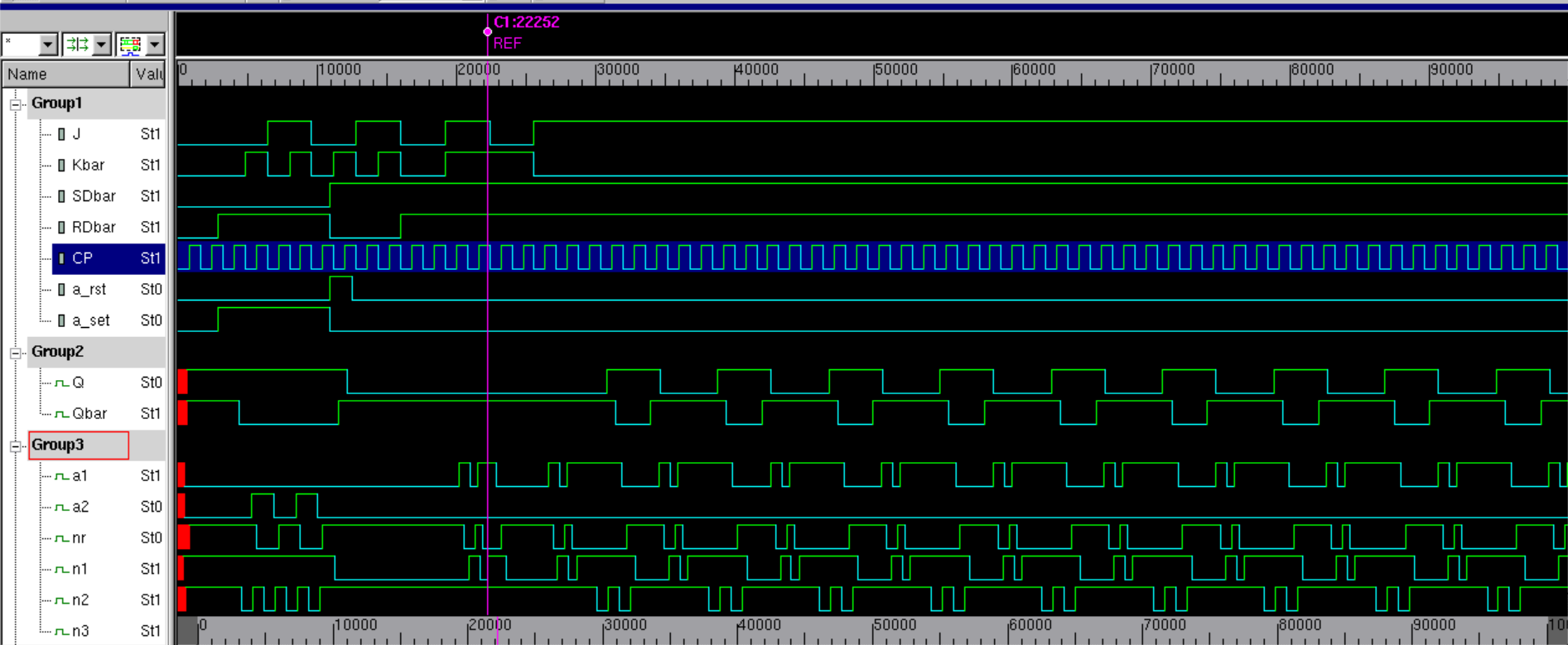
****

Figure 2.5 JK Behavioral Waveforms: 50% Duty Cycle Non-Functional

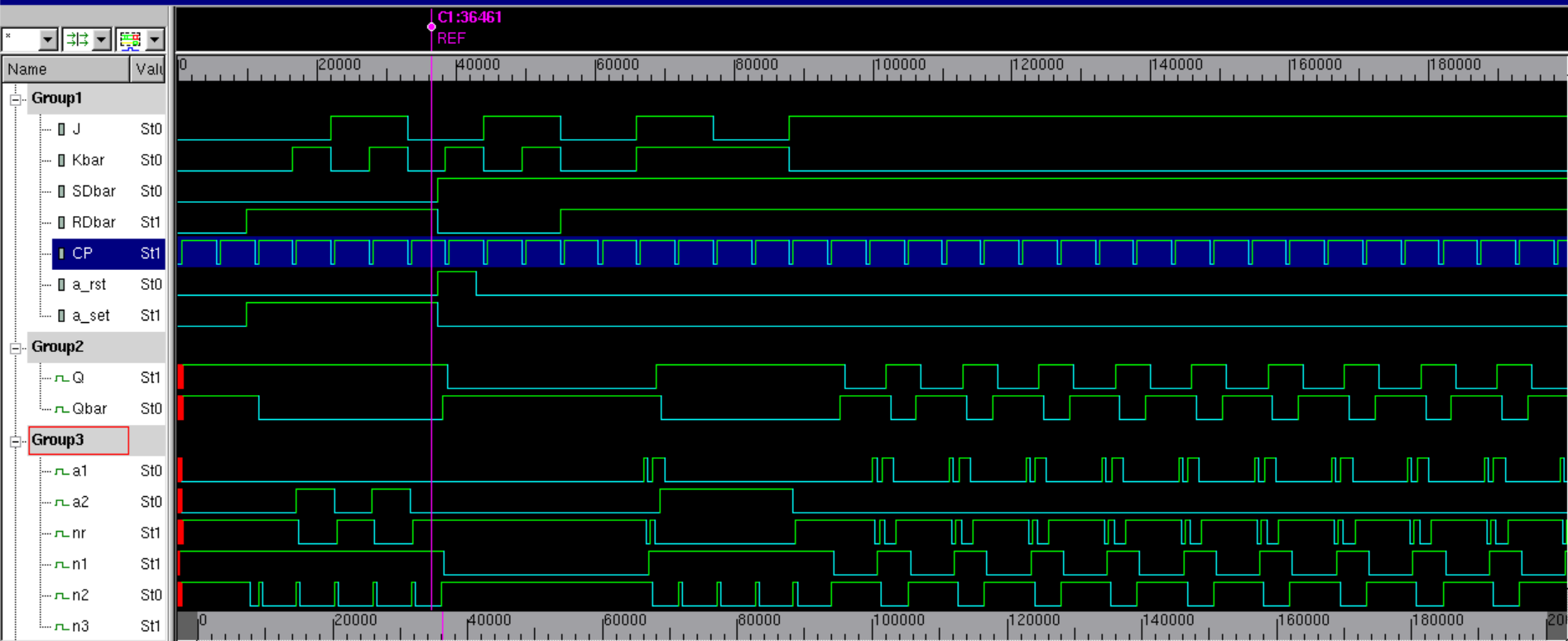
****

Figure 2.6 JK Behavioral Waveforms: 10% Duty Cycle Functional

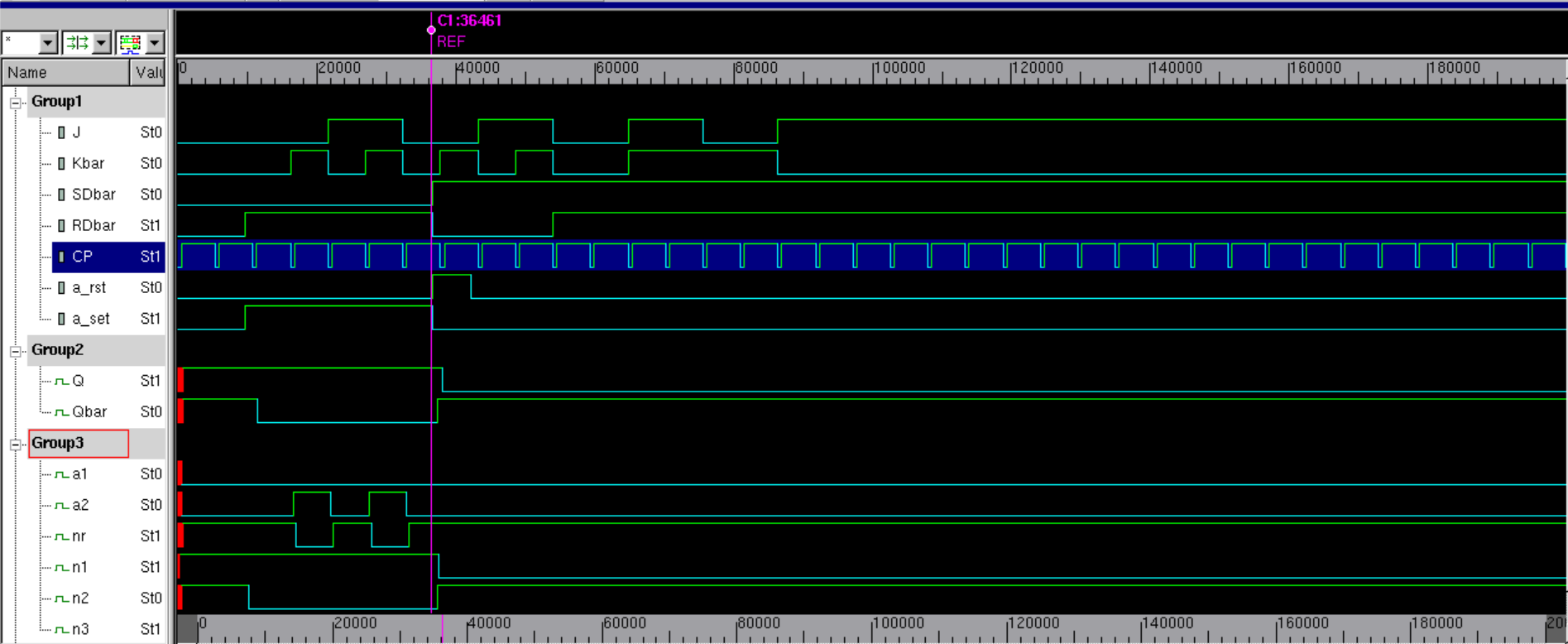
****

Figure 2.7 JK Behavioral Waveforms: 10% Duty Cycle Non-Functional

1. **ANALYSIS**

From the behavioral waveforms shown in Figs 2.3 and 2.4 and their tabulated representations in Tables 2.4 and 2.5 respectively, multiple conclusions about the functionality of the JKbar FF can be derived. A summary of these observations is provided below.

The flipflop exhibits traits of a JK flip-flop under specific operating modes, in addition to possessing an asynchronous set and reset. Specifically, the input combinations of SD and RD bar determine if the read inputs are treated as meaningful, then if they are processed synchronously or Asynchronously. This behavior was described by Table 2.3 above but is confirmed by the simulation output under normalized clocking conditions. This confirmation can be observed for the Asynchronous Set and Reset cases in Test vectors 1-8 of the testbench module output. This module was tested for all meaningful input combinations of J and Kbar inputs to demonstrate asynchrony. The basic expected functions of the JK flip-flop can be observed in test vectors 9 through 19. This Synchronous functionality is only present when both SDbar and RDbar are Logical High. These test vectors confirm the functionality of the FF as a JK FF: in test vectors 9 & 10 0 is loaded to the FF when the J and Kbar inputs are 0. When J and Kbar are both 1 the FF is set to 1, when J is 0 but Kbar is 1 the previous value is held and finally when J is 1 and Kbar is 0 the previous value is toggled.

In addition to confirming the functionality of the flip-flop, the maximum simulated operating frequency was also empirically determined through iterative testing for both clocking cases. In the 50% duty cycle clocking case the maximum operating frequency, fmax50, was determined by substituting the simulated clock period of the first fault into Eq 2.4(3). This calculation is shown in Eq 2.4(5).

(5)

When converted to MHz, the result of Eq2.4(5) is represented as Item 1.

Item 1: \_\_\_\_\_\_\_\_\_60.61\_\_\_\_\_\_\_

Similarly the maxium operating frequency of the 10% duty cycle clocking case was determined from the first simulated clock period with no fault. This calculation is shown in Eq 2.4(6).

(5)

When converted to MHz, the result of Eq2.4(6) is represented as Item 2.

Item 2: \_\_\_\_\_\_\_\_\_18.35\_\_\_\_\_\_\_

The discrepancy in these frequencies can be explained by the intrinsic and fanout delays of the module primitives. If the clock transitions from 0🡪1 faster than the slowest delay of the circuit, this will cause the output to malfunction and the FF to propagate invalid values. In the 50% duty cycle case, the clock period was normalized so it was able to run at a much higher frequency due to more time allotted before the clock would transition. If the clock transitions were closer to together such as in the 10% duty cycle case, the circuit could only be run at a much lower frequency.

1. **CONCLUSION**

//Conclusive evidence

From this experiment multiple conclusions about the register can be

//What did the experiment show/prove

This experiment illustrated how

//Circuit Limitations

**APPENDIX**

1. **SIMULATION LOG A : JK FF functional modes**

Chronologic VCS simulator copyright 1991-2017

Contains Synopsys proprietary information.

Compiler version N-2017.12-SP2-2\_Full64; Runtime version N-2017.12-SP2-2\_Full64; Feb 13 18:36 2020

VCD+ Writer N-2017.12-SP2-2\_Full64 Copyright (c) 1991-2017 by Synopsys Inc.

TV = -1 t= 0

RDbar = 0 SDbar = 0 J = 0 Kbar = 0 Q = x Qbar=x

TV = -1 t= 4

RDbar = 0 SDbar = 0 J = 0 Kbar = 0 Q = x Qbar=x

-------------------------------------|| Initial Conditions ||-------------------------------------

TV = 0 t= 39

RDbar = 0 SDbar = 0 J = 0 Kbar = 0 Q = 1 Qbar=1

--------------------------------------------------------------------------------------------------

TV = 1 t= 75

RDbar = 1 SDbar = 0 J = 0 Kbar = 0 Q = 1 Qbar=1

TV = 2 t= 110

RDbar = 1 SDbar = 0 J = 0 Kbar = 1 Q = 1 Qbar=0

TV = 3 t= 146

RDbar = 1 SDbar = 0 J = 1 Kbar = 0 Q = 1 Qbar=0

TV = 4 t= 181

RDbar = 1 SDbar = 0 J = 1 Kbar = 1 Q = 1 Qbar=0

TV = 5 t= 217

RDbar = 1 SDbar = 0 J = 0 Kbar = 0 Q = 1 Qbar=0

TV = 6 t= 252

RDbar = 0 SDbar = 1 J = 0 Kbar = 1 Q = 1 Qbar=1

TV = 7 t= 288

RDbar = 0 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 8 t= 323

RDbar = 0 SDbar = 1 J = 1 Kbar = 1 Q = 0 Qbar=1

TV = 9 t= 359

RDbar = 1 SDbar = 1 J = 0 Kbar = 0 Q = 0 Qbar=1

TV = 10 t= 394

RDbar = 1 SDbar = 1 J = 0 Kbar = 0 Q = 0 Qbar=1

TV = 11 t= 430

RDbar = 1 SDbar = 1 J = 1 Kbar = 1 Q = 0 Qbar=1

TV = 12 t= 465

RDbar = 1 SDbar = 1 J = 1 Kbar = 1 Q = 0 Qbar=1

TV = 13 t= 501

RDbar = 1 SDbar = 1 J = 0 Kbar = 1 Q = 0 Qbar=1

TV = 14 t= 536

RDbar = 1 SDbar = 1 J = 0 Kbar = 1 Q = 0 Qbar=1

TV = 15 t= 572

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 16 t= 607

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 17 t= 643

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 18 t= 678

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 19 t= 714

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 20 t= 749

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 21 t= 785

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 22 t= 820

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 23 t= 856

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 24 t= 891

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 25 t= 927

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 26 t= 962

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 27 t= 998

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

$finish called from file "tb\_JKbar\_FF.v", line 176.

$finish at simulation time 100000

V C S S i m u l a t i o n R e p o r t

Time: 1000000 ps

CPU Time: 0.200 seconds; Data structure size: 0.0Mb

Thu Feb 13 18:36:10 2020

1. **SIMULATION LOG B : 50% Duty Cycle @ functional frequency**

Chronologic VCS simulator copyright 1991-2017

Contains Synopsys proprietary information.

Compiler version N-2017.12-SP2-2\_Full64; Runtime version N-2017.12-SP2-2\_Full64; Feb 13 18:33 2020

VCD+ Writer N-2017.12-SP2-2\_Full64 Copyright (c) 1991-2017 by Synopsys Inc.

TV = -1 t= 0

RDbar = 0 SDbar = 0 J = 0 Kbar = 0 Q = x Qbar=x

TV = -1 t= 2

RDbar = 0 SDbar = 0 J = 0 Kbar = 0 Q = x Qbar=x

-------------------------------------|| Initial Conditions ||-------------------------------------

TV = 0 t= 19

RDbar = 0 SDbar = 0 J = 0 Kbar = 0 Q = 1 Qbar=1

--------------------------------------------------------------------------------------------------

TV = 1 t= 36

RDbar = 1 SDbar = 0 J = 0 Kbar = 0 Q = 1 Qbar=1

TV = 2 t= 53

RDbar = 1 SDbar = 0 J = 0 Kbar = 1 Q = 1 Qbar=1

TV = 3 t= 70

RDbar = 1 SDbar = 0 J = 1 Kbar = 0 Q = 1 Qbar=0

TV = 4 t= 87

RDbar = 1 SDbar = 0 J = 1 Kbar = 1 Q = 1 Qbar=0

TV = 5 t= 104

RDbar = 1 SDbar = 0 J = 0 Kbar = 0 Q = 1 Qbar=0

TV = 6 t= 121

RDbar = 0 SDbar = 1 J = 0 Kbar = 1 Q = 1 Qbar=0

TV = 7 t= 138

RDbar = 0 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 8 t= 155

RDbar = 0 SDbar = 1 J = 1 Kbar = 1 Q = 0 Qbar=1

TV = 9 t= 172

RDbar = 1 SDbar = 1 J = 0 Kbar = 0 Q = 0 Qbar=1

TV = 10 t= 189

RDbar = 1 SDbar = 1 J = 0 Kbar = 0 Q = 0 Qbar=1

TV = 11 t= 206

RDbar = 1 SDbar = 1 J = 1 Kbar = 1 Q = 0 Qbar=1

TV = 12 t= 223

RDbar = 1 SDbar = 1 J = 1 Kbar = 1 Q = 0 Qbar=1

TV = 13 t= 240

RDbar = 1 SDbar = 1 J = 0 Kbar = 1 Q = 0 Qbar=1

TV = 14 t= 257

RDbar = 1 SDbar = 1 J = 0 Kbar = 1 Q = 0 Qbar=1

TV = 15 t= 274

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 16 t= 291

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 17 t= 308

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 18 t= 325

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 19 t= 342

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 20 t= 359

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 21 t= 376

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 22 t= 393

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 23 t= 410

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 24 t= 427

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 25 t= 444

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 26 t= 461

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 27 t= 478

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 28 t= 495

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 29 t= 512

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 30 t= 529

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 31 t= 546

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 32 t= 563

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 33 t= 580

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 34 t= 597

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 35 t= 614

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 36 t= 631

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 37 t= 648

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 38 t= 665

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 39 t= 682

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 40 t= 699

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

$finish called from file "tb\_JKbar\_FF.v", line 176.

$finish at simulation time 70000

V C S S i m u l a t i o n R e p o r t

Time: 700000 ps

CPU Time: 0.200 seconds; Data structure size: 0.0Mb

Thu Feb 13 18:33:11 2020

1. **SIMULATION LOG C: 10% Duty Cycle @ functional frequency**

Chronologic VCS simulator copyright 1991-2017

Contains Synopsys proprietary information.

Compiler version N-2017.12-SP2-2\_Full64; Runtime version N-2017.12-SP2-2\_Full64; Feb 13 18:24 2020

VCD+ Writer N-2017.12-SP2-2\_Full64 Copyright (c) 1991-2017 by Synopsys Inc.

TV = -1 t= 0

RDbar = 0 SDbar = 0 J = 0 Kbar = 0 Q = x Qbar=x

TV = -1 t= 6

RDbar = 0 SDbar = 0 J = 0 Kbar = 0 Q = x Qbar=x

-------------------------------------|| Initial Conditions ||-------------------------------------

TV = 0 t= 61

RDbar = 0 SDbar = 0 J = 0 Kbar = 0 Q = 1 Qbar=1

--------------------------------------------------------------------------------------------------

TV = 1 t= 116

RDbar = 1 SDbar = 0 J = 0 Kbar = 0 Q = 1 Qbar=1

TV = 2 t= 171

RDbar = 1 SDbar = 0 J = 0 Kbar = 1 Q = 1 Qbar=0

TV = 3 t= 226

RDbar = 1 SDbar = 0 J = 1 Kbar = 0 Q = 1 Qbar=0

TV = 4 t= 281

RDbar = 1 SDbar = 0 J = 1 Kbar = 1 Q = 1 Qbar=0

TV = 5 t= 336

RDbar = 1 SDbar = 0 J = 0 Kbar = 0 Q = 1 Qbar=0

TV = 6 t= 391

RDbar = 0 SDbar = 1 J = 0 Kbar = 1 Q = 0 Qbar=1

TV = 7 t= 446

RDbar = 0 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 8 t= 501

RDbar = 0 SDbar = 1 J = 1 Kbar = 1 Q = 0 Qbar=1

TV = 9 t= 556

RDbar = 1 SDbar = 1 J = 0 Kbar = 0 Q = 0 Qbar=1

TV = 10 t= 611

RDbar = 1 SDbar = 1 J = 0 Kbar = 0 Q = 0 Qbar=1

TV = 11 t= 666

RDbar = 1 SDbar = 1 J = 1 Kbar = 1 Q = 0 Qbar=1

TV = 12 t= 721

RDbar = 1 SDbar = 1 J = 1 Kbar = 1 Q = 1 Qbar=0

TV = 13 t= 776

RDbar = 1 SDbar = 1 J = 0 Kbar = 1 Q = 1 Qbar=0

TV = 14 t= 831

RDbar = 1 SDbar = 1 J = 0 Kbar = 1 Q = 1 Qbar=0

TV = 15 t= 886

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 1 Qbar=0

TV = 16 t= 941

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 1 Qbar=0

TV = 17 t= 996

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 18 t= 1051

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 1 Qbar=0

TV = 19 t= 1106

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 20 t= 1161

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 1 Qbar=0

TV = 21 t= 1216

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 22 t= 1271

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 1 Qbar=0

TV = 23 t= 1326

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 24 t= 1381

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 1 Qbar=0

TV = 25 t= 1436

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 26 t= 1491

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 1 Qbar=0

TV = 27 t= 1546

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 28 t= 1601

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 1 Qbar=0

TV = 29 t= 1656

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 30 t= 1711

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 1 Qbar=0

TV = 31 t= 1766

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 32 t= 1821

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 1 Qbar=0

TV = 33 t= 1876

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

TV = 34 t= 1931

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 1 Qbar=0

TV = 35 t= 1986

RDbar = 1 SDbar = 1 J = 1 Kbar = 0 Q = 0 Qbar=1

$finish called from file "tb\_JKbar\_FF.v", line 176.

$finish at simulation time 200000

V C S S i m u l a t i o n R e p o r t

Time: 2000000 ps

CPU Time: 0.260 seconds; Data structure size: 0.0Mb

Thu Feb 13 18:24:31 2020