Timing Analysis of Domino Logic

by

David Van Campenhout, Trevor Mudge and Karem Sakallah

CSE-TR-296-96

THE UNIVERSITY OF MICHIGAN

Computer Science and Engineering Division
Department of Electrical Engineering and Computer Science
Ann Arbor, Michigan 48109-2122
USA

Timing Analysis of Domino Logic

David Van Campenhout, Trevor Mudge and Karem Sakallah

Advanced Computer Architecture Laboratory
Department of Electrical Engineering and Computer Science
University of Michigan
Ann Arbor, Michigan 48109-2122

March 27, 1996.

Abstract

High performance microprocessors employ various advanced circuit techniques to obtain the required speed. Critical sections of the design are often implemented in domino logic, a popular style of dynamic logic. This paper describes a new model for analyzing the temporal behavior of sequential circuits using static logic mixed with domino logic. Our model integrates naturally with the SMO model for static timing analysis of sequential circuits.

1 Introduction

High performance microprocessor designs employ various advanced circuit techniques to achieve high clock frequencies. Critical sections of the design are often implemented in domino logic, a popular style of dynamic logic. This paper describes a new model for analyzing the temporal behavior of sequential circuits containing both static logic and domino logic. Our model extends the SMO model [1] for static timing analysis of sequential circuits. We propose extensions to the Cascade Design Automation's static timing tool TACTIC based on our timing model. An example is presented to illustrate the concepts.

2 Domino Logic

Domino logic is a popular dynamic logic family. A generic domino gate is depicted in Figure 1. The operation of the circuit is as follows. When the clock ϕ is low, the internal node z is precharged, and the output node y is set to 0. The period in which ϕ is low is called the *precharge phase*. A rising transition on the clock conditionally discharges the internal node z through the pulldown network. The values of the inputs x determine whether the discharge actually takes place. This phase is called the *evaluate phase*. Once z is discharged, it will stay low for the rest of the evaluate phase no matter what values the inputs assume. Therefore, either the inputs have to settle to their stable value before the start of the evaluate phase, or they can settle to their stable value (a high value) by making a single rising transition during the evaluate phase.

The inverter at the output of the gate is included for several reasons. First, it is required for proper operation of a chain of domino gates. Second, the internal node z is a weak node. When the clock is low, the high value on that node is not driven. The inverting buffer separates that dynamic node from the rest of the circuit, thus alleviating charge-sharing problems and minimizing capacitive coupling. A consequence of the inverting buffer is that a domino gate can only implement a non-inverting function of its inputs. The dual circuit of that shown in Figure 1 is also a valid domino gate.

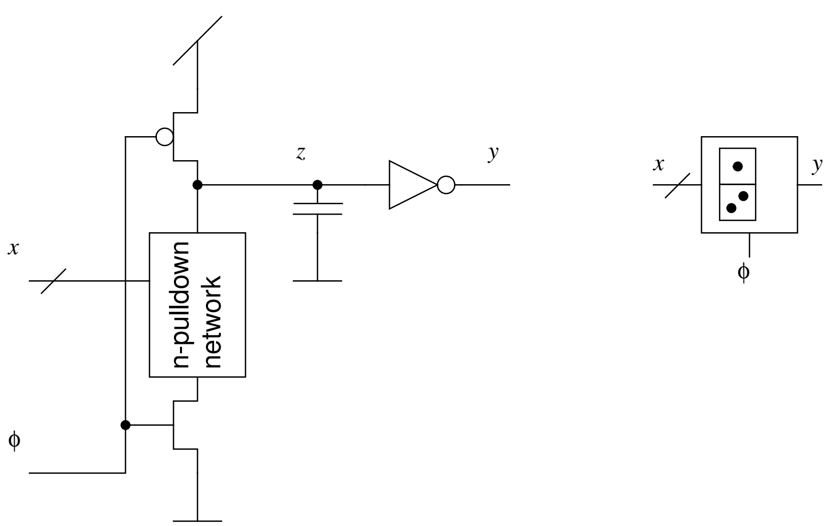

Figure 1: Generic domino gate: circuit (left), symbol (right)

However, in many technologies, and especially in CGaAs, the performance of such a dual gate is far inferior to that of the original gate due to the poor characteristics of the pFET. Among the reasons why zipper logic, a variant of domino logic is not very popular are that it uses these dual gates and that it does not use the buffering inverter. The following discussion will only be concerned with the type of gate shown in Figure 1.

3 Static Timing Analysis

Static timing analysis is concerned with verifying that a sequential circuit operates correctly given a delay model and a clock schedule. In this paper we will assume that all of the circuit's synchronizers are edge-triggered. A single primary clock is distributed to each synchronizer via a network consisting of buffers. These assumptions are not essential to the discussion, but they greatly simplify the understanding. The more general case in which level-sensitive latches are present, and a multi-phase clocking system is present is discussed in [2].

Static timing verification is used to ensure that the circuit will latch the same values as the underlying delayless circuit. The timing analyzer needs to compute the longest and the shortest signal paths between each pair of synchronizers. For edge triggered synchronizers, a transition at the output of the synchronizer can only be triggered by a transition of the clock input. Therefore, the paths of interest are those starting with a transition on a primary clock or a transition on a primary input. The paths end at the data-input of a synchronizer or a primary output.

Those longest and shortest paths can be computed by performing a breadth first search, starting from the primary clock and primary inputs, for each transition. The rising transition on the primary clock will sensitize paths starting with the clock-to-output path of positive-edge triggered synchronizers. Similar paths at negative-edge triggered synchronizers will be sensitized by the falling transition of the primary clock. Transition chains can also originate from any transition on a primary (data-)input. For each circuit node four events are kept track of: the earliest/latest rising/falling transition. Event times at input (output) signals of a circuit element will be referred to as arrival (departure) times. In the next sections we will describe how these events at the output of a domino gate can be computed, given these events on the inputs to the gate. Furthermore, for correct operation, the signals at the input of the domino gate have to comply with certain constraints with respect to the clock input.

3.1 Delay Model

We assume that for each gate a set of min/max, input to output delays are available for each input/output pair. The minimum delay from a transition of type T_i (either rising or falling) on input i that triggers a transition of type T_y on output y is denoted by $\delta_{i,y}^{T_iT_y}$. Similarly for the maximum delay $\Delta_{i,y}^{T_iT_y}$. Depending on the gate type not all combinations apply. For example, a 2-input domino AND-gate is characterized by the set $\left\{\delta_{a,y}^{RR}, \delta_{b,y}^{RR}, \delta_{\phi,y}^{RR}, \delta_{\phi,y}^{FF}, \Delta_{a,y}^{RR}, \Delta_{b,y}^{RR}, \Delta_{\phi,y}^{RR}, \Delta_{\phi,y}^{FF}\right\}$. For a static XOR-gate, any transition on the output can be triggered by any transition on an input. Hence all transition-pairs (T_i, T_y) apply. Note that the differentiation between minimum and maximum delays reflects the differences in delay due to different conditions of the side inputs. This is illustrated with

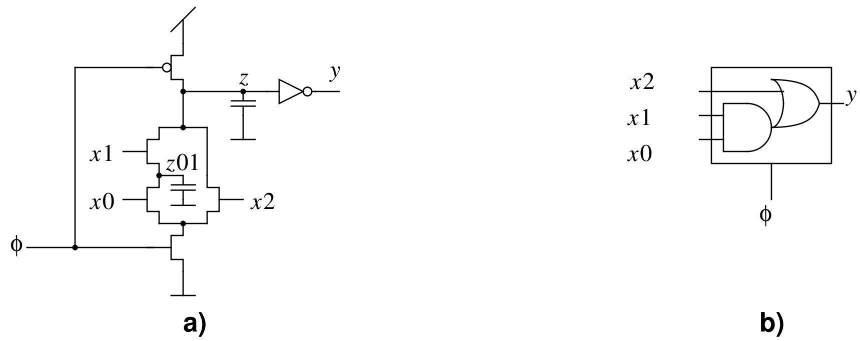

Figure 2: Domino ao21 gate: a) transistor-level schematic b) gate-level schematic

the domino ANDOR-gate in Figure 2. Consider the propagation of a rising event on input x_2 to the output. In order for the event to propagate, the clock ϕ has to be high and at least one of the other data inputs needs to be low. In case both x_0 and x_1 are low, the only capacitance that needs to be discharged is that at internal node z. The same is true in case x_1 is low and x_0 high. In this case the capacitance at node z_{01} was already discharged before x_2 rises. However when x_1 is high and x_0 low, the capacitance at node z_{01} needs to be discharged through the x_2 transistor as well. This results in a larger delay. For complex gates this effect can be significant. The example also suggests that in general, the delay from a transition on input x_i to output y can be dependent on the logic value of the sideinputs. However, into account this dependency greatly complicates the timing analysis.

3.2 Calculation of Earliest and Latest Event Times at the Output of a Domino Gate

In this section we will describe how to calculate the earliest and latest arrival times of a transition at the output of a domino gate. The earliest arrival of a rising/falling transition is denoted by a^R and a^F, respectively. Similarly, A^R and A^F denote the latest arrival of a rising and falling transition, respectively. Departure times are denoted by d^R, d^F, D^R and D^F.

The topology of domino gates implies that a falling transition on the output can only be caused by a falling transition of the clock.

$$d_y^F = a_\phi^F + \delta_{\phi, y}^{FF} \tag{1}$$

$$D_y^F = A_\phi^F + \Delta_{\phi, y}^{FF} \tag{2}$$

On the other hand, a rising transition on the output can be triggered either by a rising transition on the clock, or by a rising transition on a data input:

$$d_y^R = a_\phi^R + \delta_{\phi, y}^{RR} \tag{3}$$

$$D_y^R = \max\left\{(A_\phi^R + \Delta_{\phi,y}^{RR}), (A_i^R + \Delta_{i,y}^{RR})\right\} \tag{4}$$

Equation (4) assumes that $\Delta_{\phi,y}^{RR} \geq \Delta_{i,y}^{RR}$, for all inputs i. This is a valid assumption for practical domino gates. If it were not to hold we would have to make explicit that events on the inputs cannot be propagated unless the clock is high.

3.3 Timing Checks related to domino gates

For correct operation of the circuit, the signals have to comply with constraints. These constraints ensure that the synchronizers capture the intended signal values. These are the classical setup, hold, and pulse-width constraints for the synchronizers. For a negative-edge triggered flip flop with data-input d, clock input ϕ, and clock frequency $1/T_c$ the constraints are:

$$A_d^F \leq a_\phi^F - T_{setup} \tag{5}$$

$$A_d^R \leq a_\phi^F - T_{setup} \tag{6}$$

$$a_d^F + T_c \geq A_\phi^F + T_{hold} \tag{7}$$

$$a_d^R + T_c \geq A_\phi^F + T_{hold} \tag{8}$$

Additional constraints are necessary to ensure correct operation of the domino gates. The constraints are between events on data-inputs i, and events on the clock-input ϕ of each domino gate. Again, the clock period is T_c. For each data-input i, the following has to hold:

$$A_i^F \leq a_\phi^R - T_{setup1} \tag{9}$$

$$A_i^R \leq a_\phi^F - T_{setup2} \tag{10}$$

$$a_i^F + T_c \geq A_\phi^F + T_{hold} \tag{11}$$

$$a_i^R + T_c \geq A_\phi^F + T_{hold} \tag{12}$$

The constraints are illustrated in Figure 3. The first constraint forces the latest falling transition on an input to the domino gate to occur before the precharge phase ends. This constraint ensures that the gate cannot be inadvertently switched. The constraint is actually conservative. In case the input will make a rising transition after the latest falling transition, i.e., the input settles to high, it does not matter that the falling transition occurred before the clock goes high. Unfortunately, taking into account such effects makes the analysis more complex. The second constraint states that the

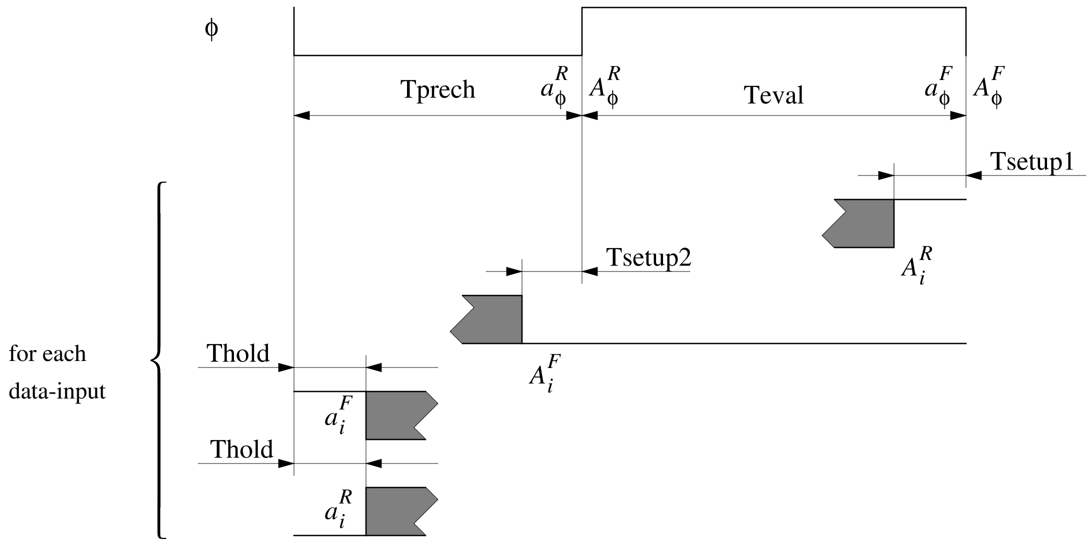

Figure 3: Constraints for a domino gate

latest rising transition on the input must happen before the end of the evaluate phase, otherwise that transition might not be propagated to the output of the domino gate. The combination of these two constraints implies that during the evaluate phase, either an input has to remain stable, or it can make a single rising transition. Constraint (11) and (12) force the earliest change in the inputs to happen after the end of the evaluate phase. Finally there are two more constraints which apply to the clock input only. They ensure that the precharge and evaluate pulses are wide enough:

$$A_\phi^F - A_\phi^R \geq T_{eval} \tag{13}$$

$$T_c - (A_\phi^F - A_\phi^R) \geq T_{prech} \tag{14}$$

Many of these constraints (9)-(14) may appear superfluous. In a pure domino subcircuit, all constraints except for constraint (10), might be trivially satisfied. However, one can always come up with a case in which clock skew is such that each of these constraints becomes relevant.

3.4 Example

A circuit fragment is shown in Figure 4. The flip-flops are negative-edge triggered. They are all clocked by the same primary clock, and there is no clock skew. The OR and the AND gates are static. The ANDOR gates are dynamic. The delays corresponding to each of the gates are shown in Table 1. For simplicity minimum and maximum delays are assumed to be equal. The timing diagram (Figure 5) depicts the paths corresponding to the constraints. The format is that used in TACTIC. The endpoints of the segments correspond to transitions on circuit nodes. The segments represent the delay to propagate the transition on the circuit node corresponding to the left end point to the circuit node corresponding to the right end point. Segments whose right endpoint are rising (falling) transitions are colored green (red). The first two paths show that the setup constraint of ff7 is met. Both the latest rising transition and the latest falling transition on g30 take place before the falling edge of the signal clocking ff7. The rising transition on clk propagates through the domino gates and triggers the latest rising transition on g30. The latest falling transition on g30 is due to the precharge of the domino gate. The next two paths correspond to the hold constraint of ff7. The diagram shows the events leading to the earliest rising transition and the earliest falling transition on g30. It is clear that the hold constraint is met. The remaining eight paths concern the constraints associated with the g0 input of ANDOR-gate g10. Note that the setup constraint for the latest falling transition is with respect to the rising edge of the clock whereas that for the latest rising transition is with respect to the falling edge of the clock. In this example the signal paths relevant to the setup constraints of g0 happen to be identical to those relevant to the setup constraints.

3.5 Requirements for Analysis of Domino Logic with TACTIC

Domino gates need to be tagged with a property that distinguishes them from other circuit element types (flip-flops, level-sensitive latches, static gates). Events need to be propagated correctly across domino gates as described in Section 3.2. This implies that both paths starting the domino gate through its clock input, as well as paths going through the domino gate via one of the data-inputs have to be considered. For domino gates, timing checks need to be performed on the appropriate events on data-inputs and the clock input. The multitude of constraints associated with each domino gate results in a large number of paths being relevant. Also, in normal domino circuits many constraints will be satisfied "almost trivially." For example, in a block of pure domino logic sharing the same clock, the hold constraints will always be met. Hence for the graphical user interface, it would be useful to allow the user to suppress certain types of relevant paths which do not violate their constraints. For instance, the user would have the option to suppress all non-violating relevant paths with respect to the hold constraints on all domino gates.

4 Conclusion

We presented a model for analyzing the temporal behavior of sequential circuits using both static and domino logic. Our model builds on the SMO model. In our model we keep track of four events per node: the earliest/latest rising/falling transition. The model assumes a min/max rise/fall gate delay model. The model is compatible with Cascade Design Automation's TACTIC timing tool.

References

[1] K. Sakallah, T. Mudge, O. Olukotun, "checkTc and minTc: Timing Verification and Optimal Clocking of Synchronous Digital Circuits," in *ICCAD-90 Digest of Technical Papers*, pp. 552-555, November 1990.

[2] David Van Campenhout, Trevor Mudge and Karem Sakallah, "*Static Timing Analysis of Dynamic Logic,*"CSE-TR XXX, University of Michigan, 1996

.

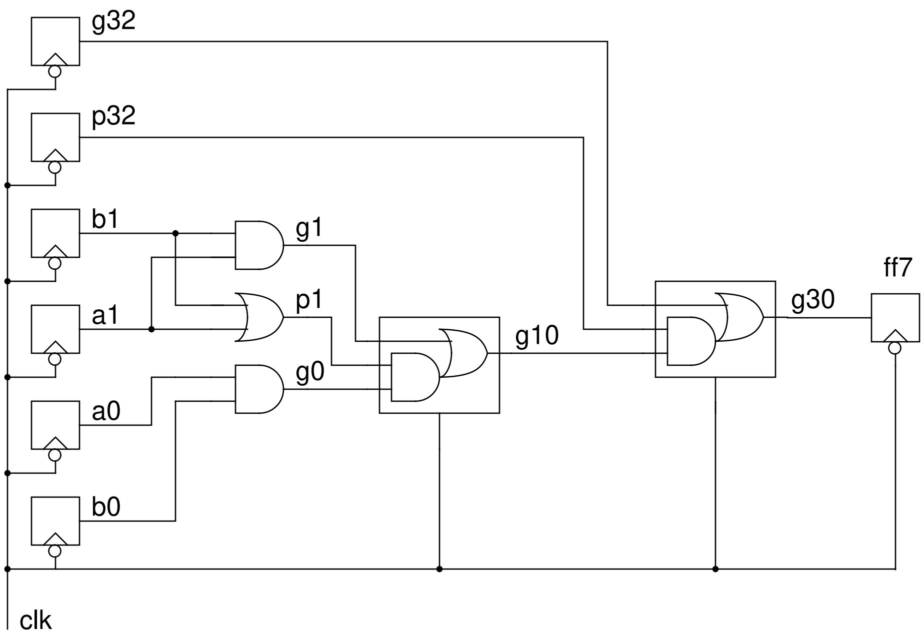

Figure 4: Circuit fragment

Table 1: Delays [ps]

gate	input	RR	FF	FR
and2	x0,x1	100	75	N/A
or2	x0,x1	75	125	N/A
ao21 ✻	x0,x1	100	N/A	N/A
	x2	100	N/A	N/A
	clk	100	125	N/A
DFF	clk	N/A	140	150

✻ x0 and x1 are the inputs which are AND-ed.

Table 1: Setup and Hold Times [ps]

	DFF	ao21
Tsetup1	N/A	50
Tsetup2	N/A	100
Tsetup	150	N/A
Thold	0	0
Tprecharge	N/A	200
Teval	N/A	200

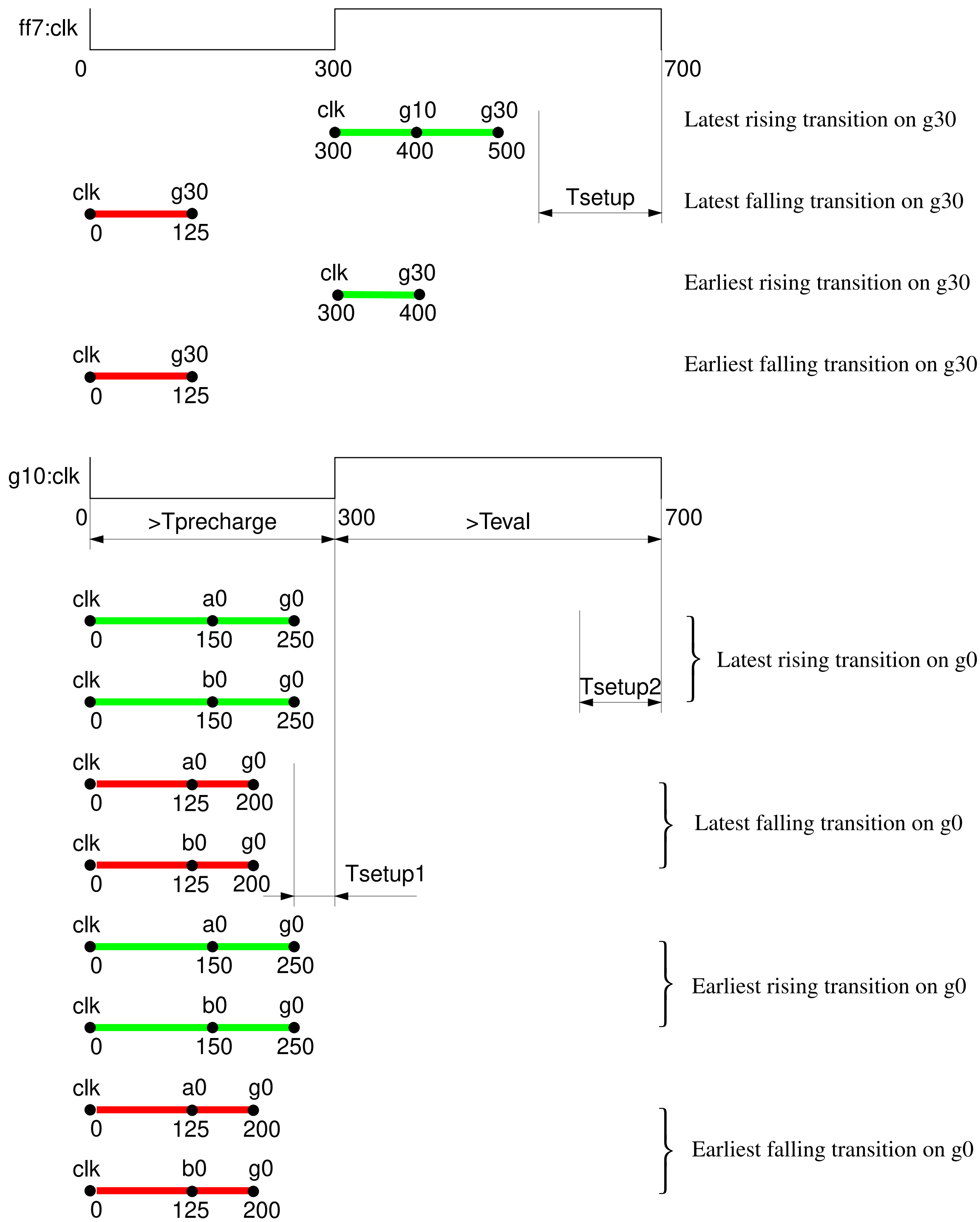

Figure 5: Timing diagram