Modeling Synchronous and Asynchronous Behavior in VHDL

David M. Russinoff

Computational Logic, Incorporated
1717 West Sixth Street, Suite 290
Austin, Texas 78703-4776
russ@cli.com

Copyright © 1994 Computational Logic, Inc.

Modeling Circuit Behavior

The hardware design process generally involves two distinct models of behavior:

- Sequential machine model
  - signal representation: waveform
  - fundamental operation: event scheduling
  - time unit: 10^{-12} sec.
- Event-driven simulation model
  - signal representation: value sequence
  - fundamental operation: state transition
  - time unit: 10^{-7} sec.

A comprehensive hardware verification system must incorporate both of these models.

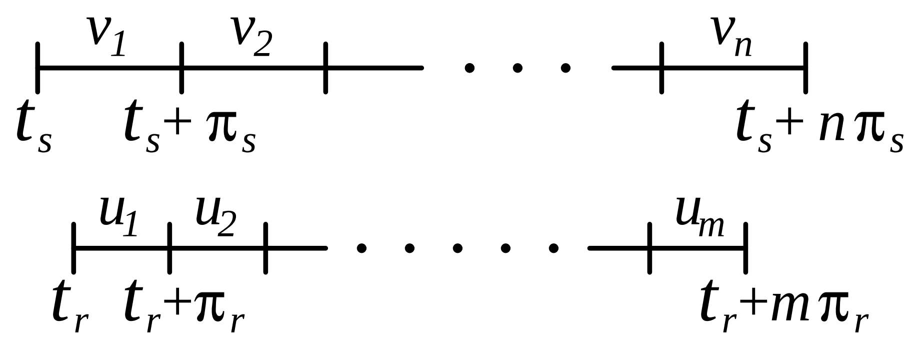
CLI COMPUTATIONAL LOGIC INCORPORATED

Outline of this Talk

- Motivation: an implementation of the biphase mark protocol for asynchronous communication
- Definition: a simple event-driven simulation language (a subset of VHDL)
- Specification: behavioral descriptions of combinational and sequential circuits
- Application: correctness of the biphase mark circuit

CLI
COMPUTATIONAL
LOGIC
INCORPORATED

Moore's Model of Asynchrony



Theorem 1 *Let* $s : \mathbf{N} \rightarrow \{\mathcal{T}, \mathcal{F}\}$*,* $V_s = (v_1 \ \ldots \ v_n)$*, and* $V_r = (u_1 \ \ldots \ u_m)$*. Assume that for all* k*,*

(1) $s(t) = v_k$ *for some* $t \in [t_s + (k-1)\pi_s, t_s + k\pi_s)$*;*

(2) $s(t) = u_k$ *for some* $t \in [t_r + (k-1)\pi_r, t_r + k\pi_r)$*;*

(3) If $v_k = v_{k-1}$*, then* $s(t) = v_k$ *for all* $t \in [t_s + (k-1)\pi_s, t_s + k\pi_s)$*.*

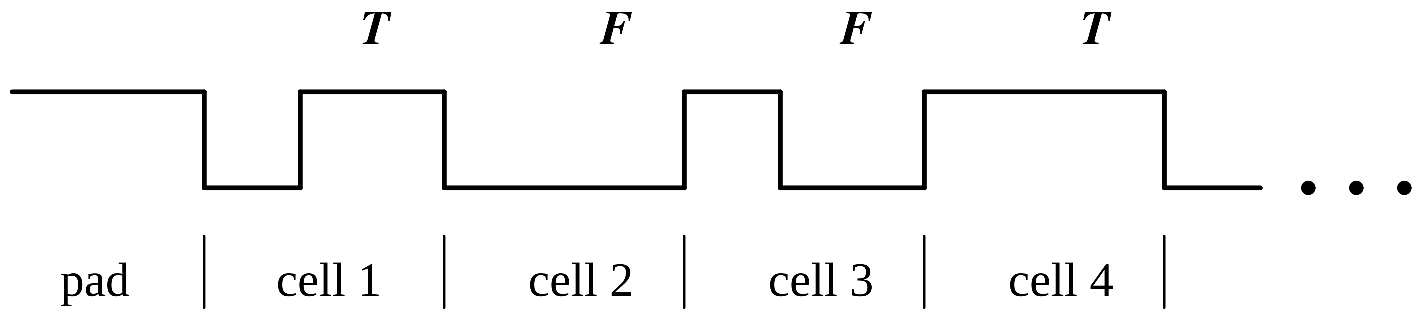
Then for some oracle, $V_r = asynch(V_s, t_s, t_r, \pi_s, \pi_r, oracle)$*.*

CLI COMPUTATIONAL LOGIC INCORPORATED

Copyright © 1994 Computational Logic, Inc.

Biphase Mark Protocol

$V_s = send(p, msg)$ consists of a *pad* of p $\mathcal{T}$'s followed by sequence of cells corresponding to the bits of msg. Each cell consists of a *mark subcell* followed by a *code subcell*.

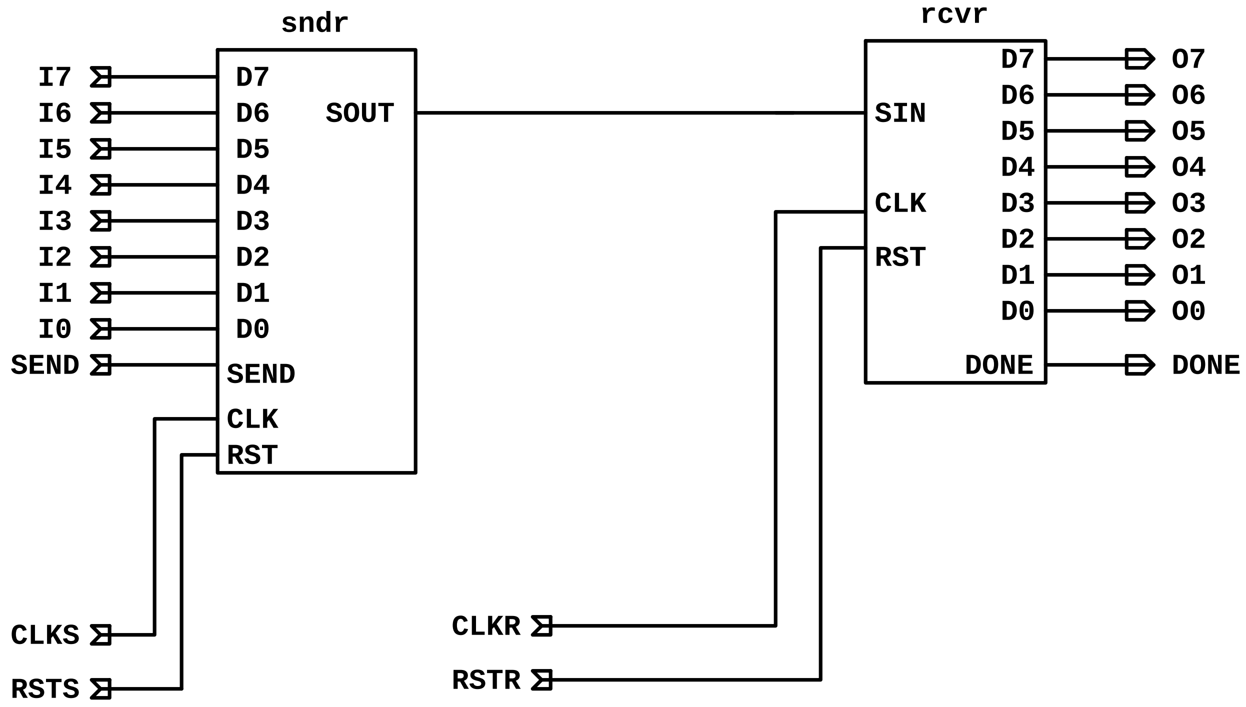


$recv(V_r) = msg$ is computed by recovering the encoded bit from each cell.

Theorem 2 (Moore) *If* $17\pi_r \leq 18\pi_s \leq 19\pi_r$ *and* $t_s \leq t_r < t_s + \pi_s$, *then* $recv(asynch(send(p, msg), t_s, t_r, \pi_s, \pi_r, oracle)) = msg$.

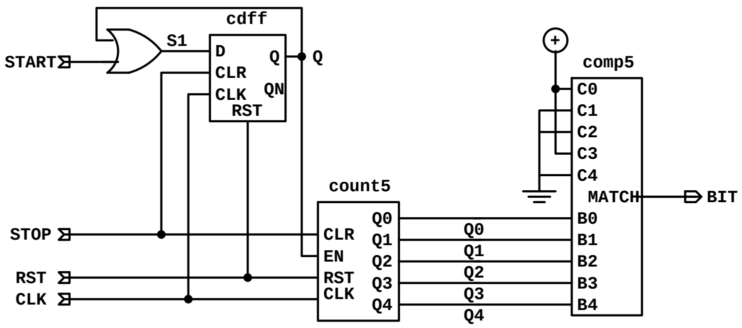
Copyright © 1994 Computational Logic, Inc.

Biphase Mark Circuit

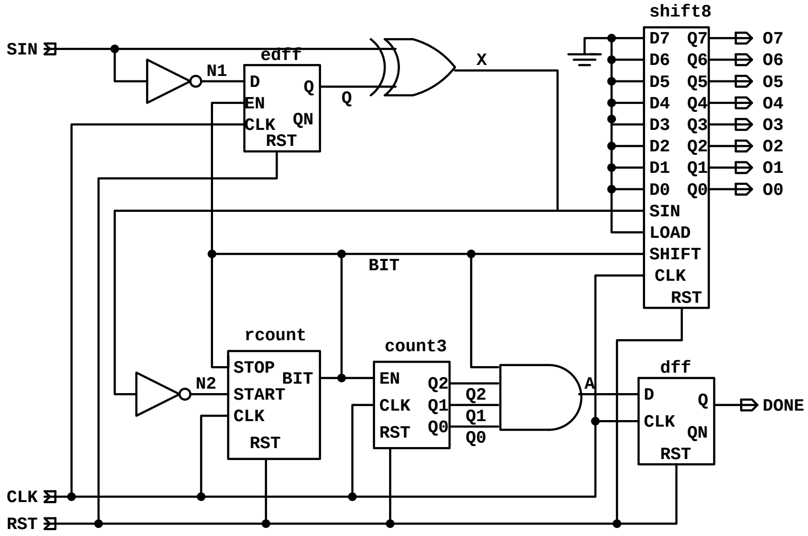


CLI COMPUTATIONAL LOGIC INCORPORATED

Receiver



rcount

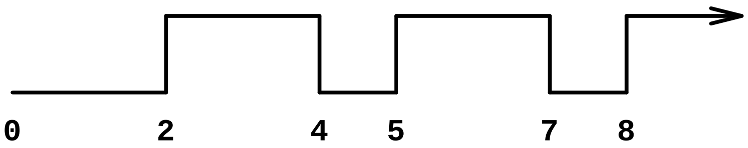


rcvr

CLI
COMPUTATIONAL
LOGIC
INCORPORATED

Waveforms

A *waveform* is a list of *events*, which determines a Boolean-valued function of time.



$$w = ((\mathcal{T}, 8)\ (\mathcal{F}, 7)\ (\mathcal{T}, 5)\ (\mathcal{F}, 4)\ (\mathcal{T}, 2)\ (\mathcal{F}, 0))$$
$$w(2) = w(3) = \mathcal{T};\ w(4) = \mathcal{F}$$

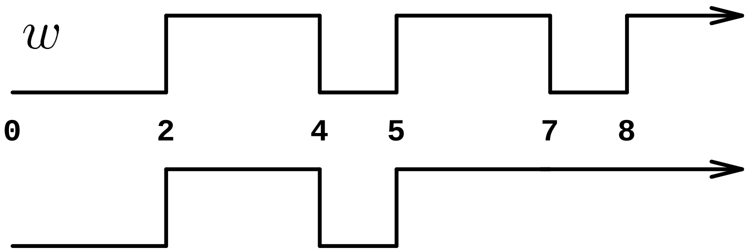
A *packet* is a list of waveforms, $p = (w_1 \ldots w_n)$, which determines a vector-valued function of time: $p(t) = (w_1(t)\ \ldots\ w_n(t))$.

CLI
COMPUTATIONAL
LOGIC
INCORPORATED

Copyright © 1994 Computational Logic, Inc.

Transport Delay

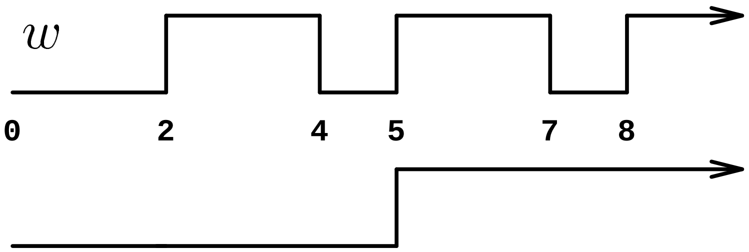
When an event (v, t) is scheduled on a waveform w, all previously scheduled events of w that occur at or after t are removed.



$transport(w, \mathcal{T}, 6)$

Inertial Delay

When an event (v, t) is scheduled on a waveform w at time t_0, all previously scheduled events of w that occur after t_0 are removed.



$inertial(w, \mathcal{T}, 1, 6)$

Behavioral Modules

A *behavioral module* consists of

(1) A list of *inputs*;

(2) A list of *outputs*;

(3) A list of *terms* corresponding to the outputs;

(4) A list of *delays* corresponding to the outputs;

(5) A list of *propagation modes* (either `TRANSPORT` or `INERTIAL`) corresponding to the outputs.

Copyright © 1994 Computational Logic, Inc.

Examples

A 2-input nand gate:

```
nand2 = (BEHAV (A B)                             ;inputs
               (C)                               ;outputs
               ((NAND2 A B))                     ;terms
               (2000)                            ;delays
               (INERTIAL))                       ;propagation modes
```

A 1-bit adder:

```
adder2 = (BEHAV (A B C)                          ;inputs
                (L H)                            ;outputs
                ((XOR3 A B C)                    ;terms
                 (OR2 (AND2 A (OR2 B C))
                      (AND2 B C)))
                (12000 10000)                    ;delays
               (INERTIAL INERTIAL))              ;propagation modes
```

CLI
COMPUTATIONAL
LOGIC
INCORPORATED

Copyright © 1994 Computational Logic, Inc.

Execution of Behavioral Modules

Execution of behavioral modules is defined by a function $exec(M, p_{in}, p_{out}, t)$, where

(1) M is a behavioral module;

(2) p_{in} is a packet of input waveforms for M;

(3) p_{out} is a packet of output waveforms for M;

(4) t is the current time.

The returned value is an updated output packet, produced by scheduling an event on each waveform of p_{out}.

CLI
COMPUTATIONAL
LOGIC
INCORPORATED

Copyright © 1994 Computational Logic, Inc.

Structural Modules

A *structural module* consists of

(1) A list of *(global) inputs*;

(2) A list of *(global) outputs*;

(3) A list of *submodules*;

(4) Corresponding to each submodule, a list of *local inputs*;

(5) Corresponding to each submodule, a list of *local outputs*.

Each global output is a local output; each local input is either a global input or a local output.

Examples

A structural version of the adder:

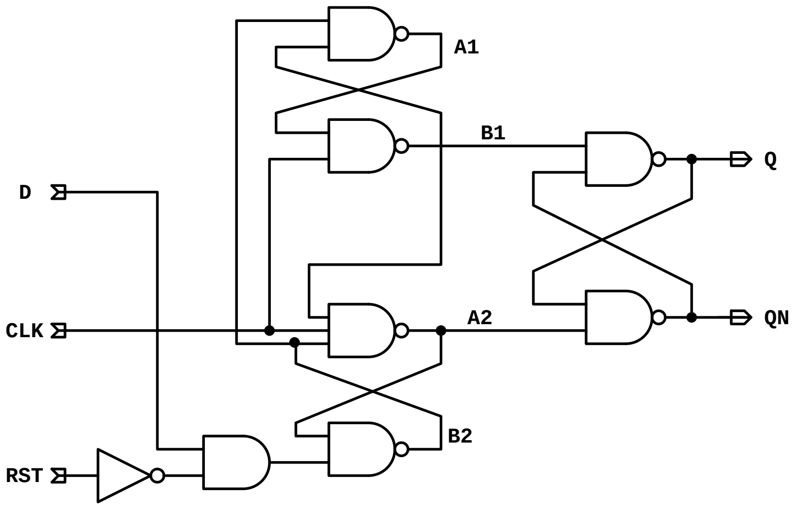
```
adder2
  =
(STRUCT (A B C) (L H)
  (nand2 nand2 nand2 nand2 nand2 nand2 nand2 nand2 nand2)
  ((A B) (A T1) (B T1) (T2 T3) (C T4) (T5 T4) (C T5) (T5 T1) (T7 T6))
  ((T1) (T2) (T3) (T4) (T5) (T6) (T7) (H) (L)))
```

A synchronous resettable D-flip-flop:

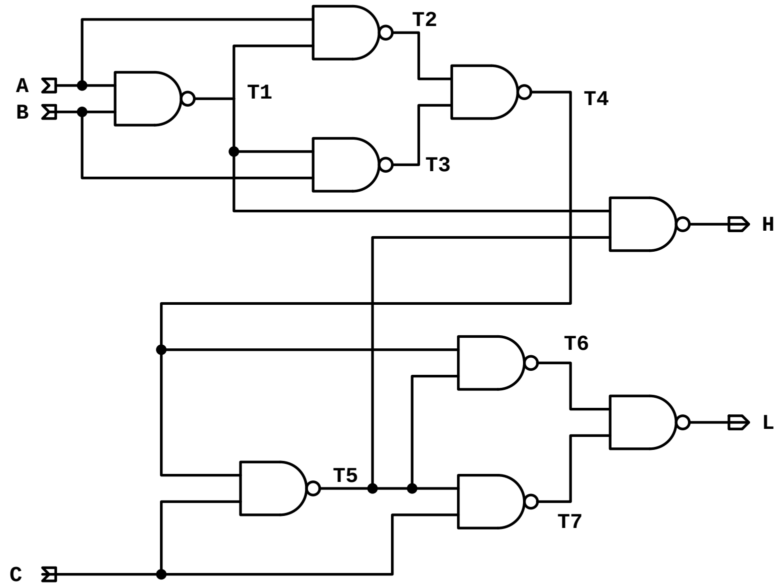
```
dff
 =
(STRUCT (CLK RST D) (Q QN)
  (not1 and2 nand2 nand2 nand3 nand2 nand2 nand2)
  ((RST) (RN D) (B2 B1) (A1 CLK) (B1 CLK B2) (A2 DD) (B1 QN) (Q A2))
  ((RN) (DD) (A1) (B1) (A2) (B2) (Q) (QN)))
```

CLI COMPUTATIONAL LOGIC INCORPORATED

Circuit Diagrams



dff



adder2

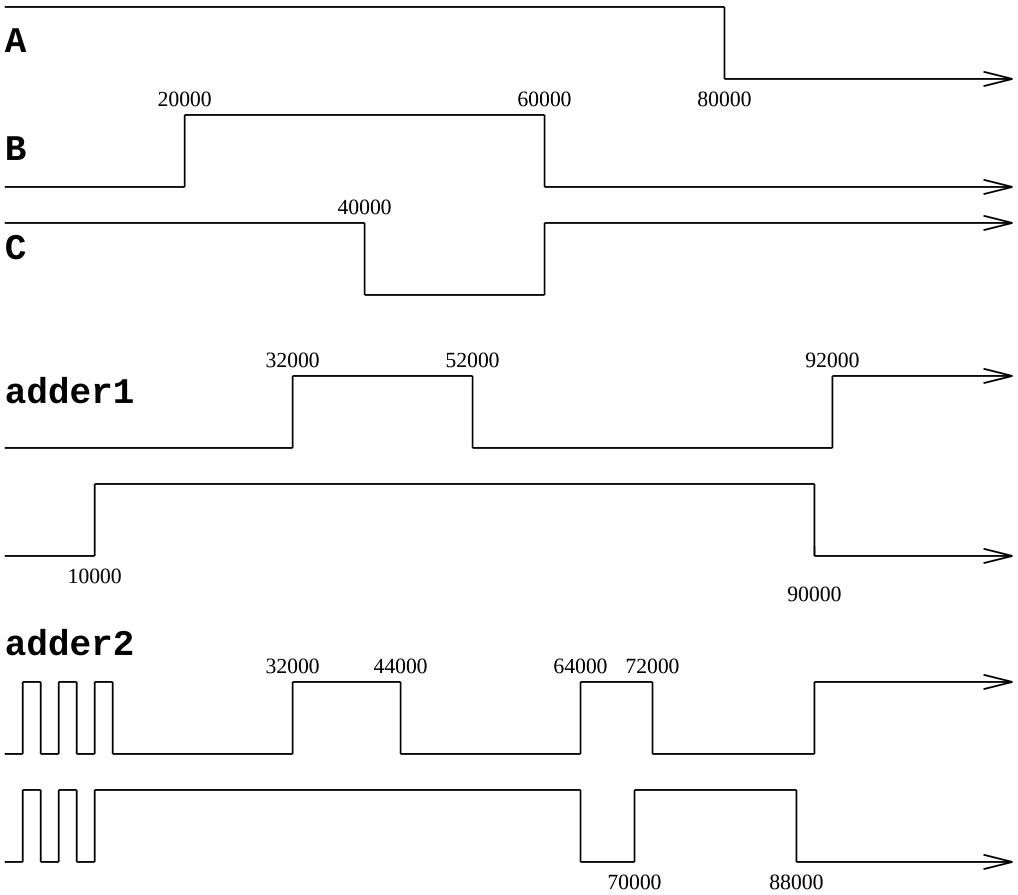
Simulation

Simulation is defined by a function $sim(M, p, t_f)$, where
(1) M is any module module;
(2) p is an input packet for M;
(3) t_f is the termination time.

The returned value consists of an output packet corresponding to each behavioral component of M, computed by the following procedure:
(1) Initialize each waveform to the trivial waveform $((\mathcal{F}, 0))$.
(2) Execute each behavioral component at $t_0 = 0$.
(3) Reset t_0 to the time of the earliest event that occurs after t_0. If $t_0 > t$, then terminate; otherwise, execute each component for which some input has an event at t_0, and repeat.

CLI COMPUTATIONAL LOGIC INCORPORATED

Example


A
B
C
adder1
adder2
20000
60000
80000
40000
32000
52000
92000
10000
90000
32000
44000
64000
72000
70000
88000

Combinational Modules

M is *combinational* if either (a) M is behavioral or (b) M is structural with no loops and all its submodules are combinational.

Given a signal s of a combinational module M and a vector V of input values, we may compute the *combinational value* $cv(s, V, M)$:

$$cv(\texttt{C}, (a\ b), \texttt{nand2}) = nand(a, b)$$
$$cv(\texttt{L}, (a\ b\ c), \texttt{adder1}) = xor(a, b, c)$$
$$cv(\texttt{L}, (a\ b\ c), \texttt{adder2}) = xor(a, b, c)$$

How can the behavior of M be described in terms of cv?

Copyright © 1994 Computational Logic, Inc.

Specification of Behavioral Modules

The relationship between the behavior of a signal s and its combinational value depends on the delay of s:

Lemma 1 *Let s be an output of a behavioral module M with delay d. Let w be the corresponding waveform of $sim(M, p, t_f))$. Suppose that $cv(s, p(t), M)$ has a constant value v on the interval $[t_1, t_2)$, where $t_2 - t_1 \geq d$. Then for all $t \in [t_1 + d, t_2 + d)$, $w(t) = v$.*

CLI
COMPUTATIONAL
LOGIC
INCORPORATED

Specification of Combinational Modules

Each signal s of a combinational M is associated with *minimum* and *maximum delays*, which are computed by comparing the total delays along all paths to s. For example,

$$dmin(\texttt{L}, \texttt{adder2}) = 4000 \text{ and } dmax(\texttt{L}, \texttt{adder2}) = 12000.$$

The relationship between the behavior of a signal s and its combinational value depends on the delay of s:

Lemma 2 *Let s be an output of a combinational module M. Let $dmin(s, M) = d$ and $dmax(s, M) = D$. Let w be the corresponding waveform of $sim(M, p, t_f))$. Suppose that $p(t)$ is constant and that $cv(s, p(t), M) = v$ on the interval $[t_1, t_2)$, where $t_2 - t_1 \geq D$. Then for all $t \in [t_1 + D, t_2 + d)$, $w(t) = v$.*

CLI
COMPUTATIONAL
LOGIC
INCORPORATED

Copyright © 1994 Computational Logic, Inc.

Sequential Modules

Let M be a structural module, two inputs of which are distinguished as the *clock* and the *reset*. M is a *sequential module* with $mult(M) = q$ if either (a) $q = 0$ and $M = \texttt{dff}$, or (b) $q > 0$ and the following conditions hold:

(1) q of the submodules of M are sequential and the rest are combinational;

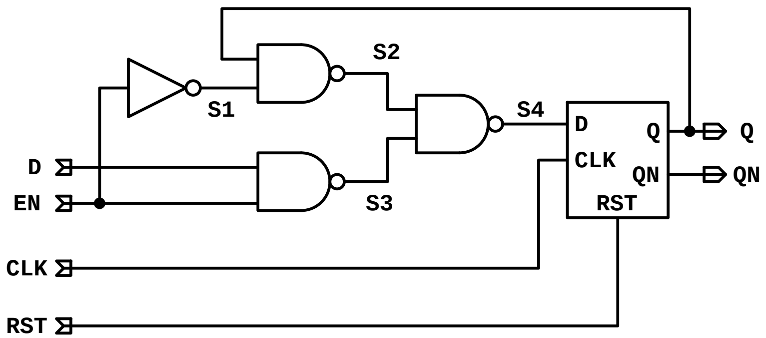
(2) The clock input of M is connected to the clock input of each sequential submodule, and not to any other submodule input;

(3) The reset input of M is connected to the reset input of each sequential submodule, and not to any other submodule input;

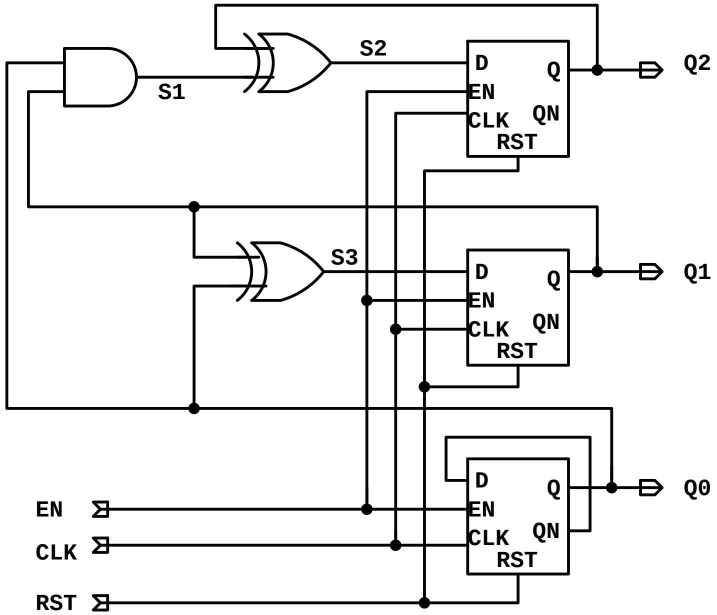
(4) There is no combinational path in M (i.e., path through only combinational submodules) from an input to an output;

(5) There is no combinational loop in M.

Examples



edff



count3

Sequential Values

Given a signal s of a sequential module M and a list $\mathcal{V}$ of data input vectors, we may compute the *sequential value list* $sv(s, \mathrm{V}, M)$:

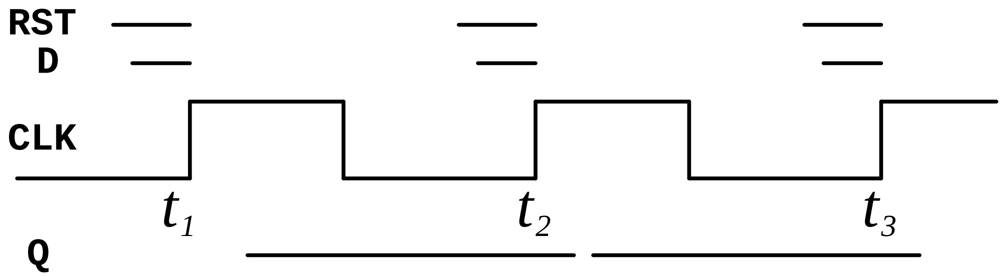
$$sv(\mathtt{Q}, ((\mathcal{T})\ (\mathcal{T})\ (\mathcal{F})\ (\mathcal{T})), \mathtt{dff}) = (\mathcal{F}\ \mathcal{T}\ \mathcal{T}\ \mathcal{F}\ \mathcal{T})$$
$$sv(\mathtt{Q}, (\mathcal{T}\ \mathcal{T})\ (\mathcal{T}\ \mathcal{T})\ (\mathcal{F}\ \mathcal{F})\ (\mathcal{F}\ \mathcal{T})), \mathtt{edff}) = (\mathcal{F}\ \mathcal{T}\ \mathcal{T}\ \mathcal{T}\ \mathcal{T})$$
$$sv(\mathtt{Q0}, ((\mathcal{T})\ (\mathcal{T})\ (\mathcal{F})\ (\mathcal{T})), \mathtt{count3}) = (\mathcal{F}\ \mathcal{T}\ \mathcal{F}\ \mathcal{F}\ \mathcal{T})$$

How can the behavior of M be described in terms of sv?

CLI
COMPUTATIONAL
LOGIC
INCORPORATED

Flip-Flop Specification

Lemma 3 *Let* $p = (w_{\texttt{CLK}}\, w_{\texttt{RST}}\, w_{\texttt{D}})$, *where*
(a) $w_{\texttt{CLK}}$ *is a periodic clock pulse with high* ≥ 4000, *low* ≥ 6000, *and successive rising edges at* $t_1, t_2, t_3, \ldots$
(b) $w_{\texttt{RST}}(t) = r$ *for all* $t \in [t_1 - 8000, t_1)$
(c) $w_{\texttt{D}}(t) = d$ *for all* $t \in [t_1 - 6000, t_1)$.
Let $w_{\texttt{Q}}$ *and* $w_{\texttt{QN}}$ *be the output waveforms of* $sim(\texttt{dff}, p, t_f)$, *and let* $v = and(not(r), d)$.
Then $w_{\texttt{Q}}(t) = v$ *and* $w_{\texttt{QN}}(t) = not(v)$ *on* $[t_1 + 6000, t_2 + 4000)$. *Moreover, if these same values hold on* $[t_2 + 6000, t_3 + 4000)$, *then they also hold on* $[t_2, t_3)$.



Sequential Parameters

The behavior of a sequential module may be characterized in terms of the following parameters:

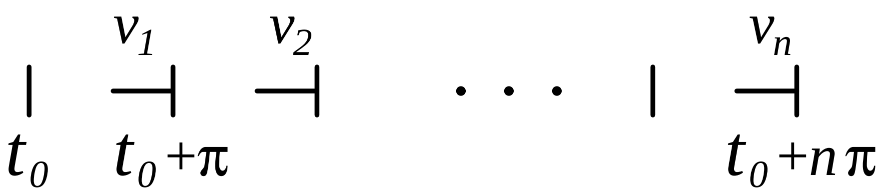
- The minimum *clock period*, *high time*, and *low time*;
- The *setup time* of a signal is the duration over which the signal is required to hold constant prior to a rising edge.
- The *minimum* and *maximum delay* of a signal determine an interval during which the signal's value may change following a rising edge.

CLI
COMPUTATIONAL
LOGIC
INCORPORATED

Copyright © 1994 Computational Logic, Inc.

Input Constraints

Suppose $w(t) = v_k$ for all $t \in [t_0 + k\pi - u, t_0 + k\pi)$, $k = 1, \ldots, n$. Then w is *stable* w.r.t. u, based at t_0 with effective value list $(v_1 \ldots v_n)$ and period π.



We impose the following constraints:

- Clock parameters are respected. Assume period π and rising edge at t_0;
- Each data input is stable w.r.t. its setup, based at t_0 with period π and effective value list of length n;
- Reset is stable w.r.t. its setup, based at $t_0 - \pi$ with period π and effective value list $(\mathcal{T}\ \mathcal{F}\ \mathcal{F}\ \ldots\ \mathcal{F})$ of length $n + 1$.

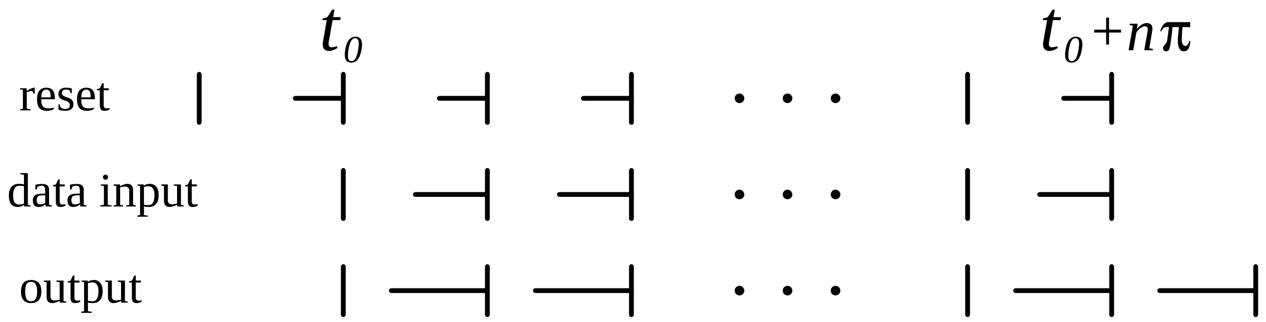
CLI
COMPUTATIONAL
LOGIC
INCORPORATED

Main Theorem

Theorem 3 *Let s be an output of a sequential module M with maximum delay D. Let w be the corresponding waveform of $sim(M, p, t_f)$, where p satisfies the input constraints for M with period π. Let $\mathcal{V}$ be the list of vectors of effective data input values.*

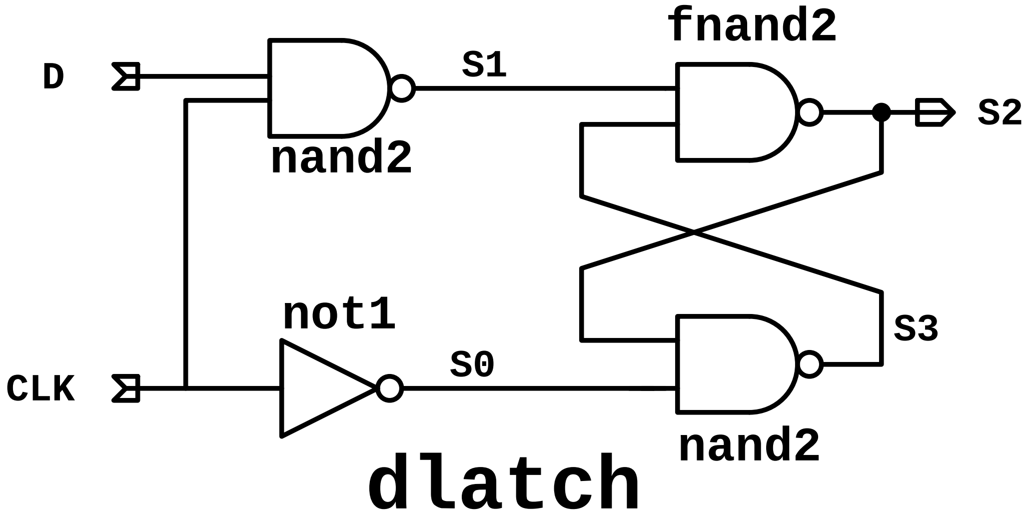
Then w is stable w.r.t. $\pi - D$, based at t_0 with effective value list $sv(s, V, M)$ and period π.

Suppose that s is a registered signal of M and that its effective values on two successive cycles coincide. Then w is constant on the second cycle.



Copyright © 1994 Computational Logic, Inc.

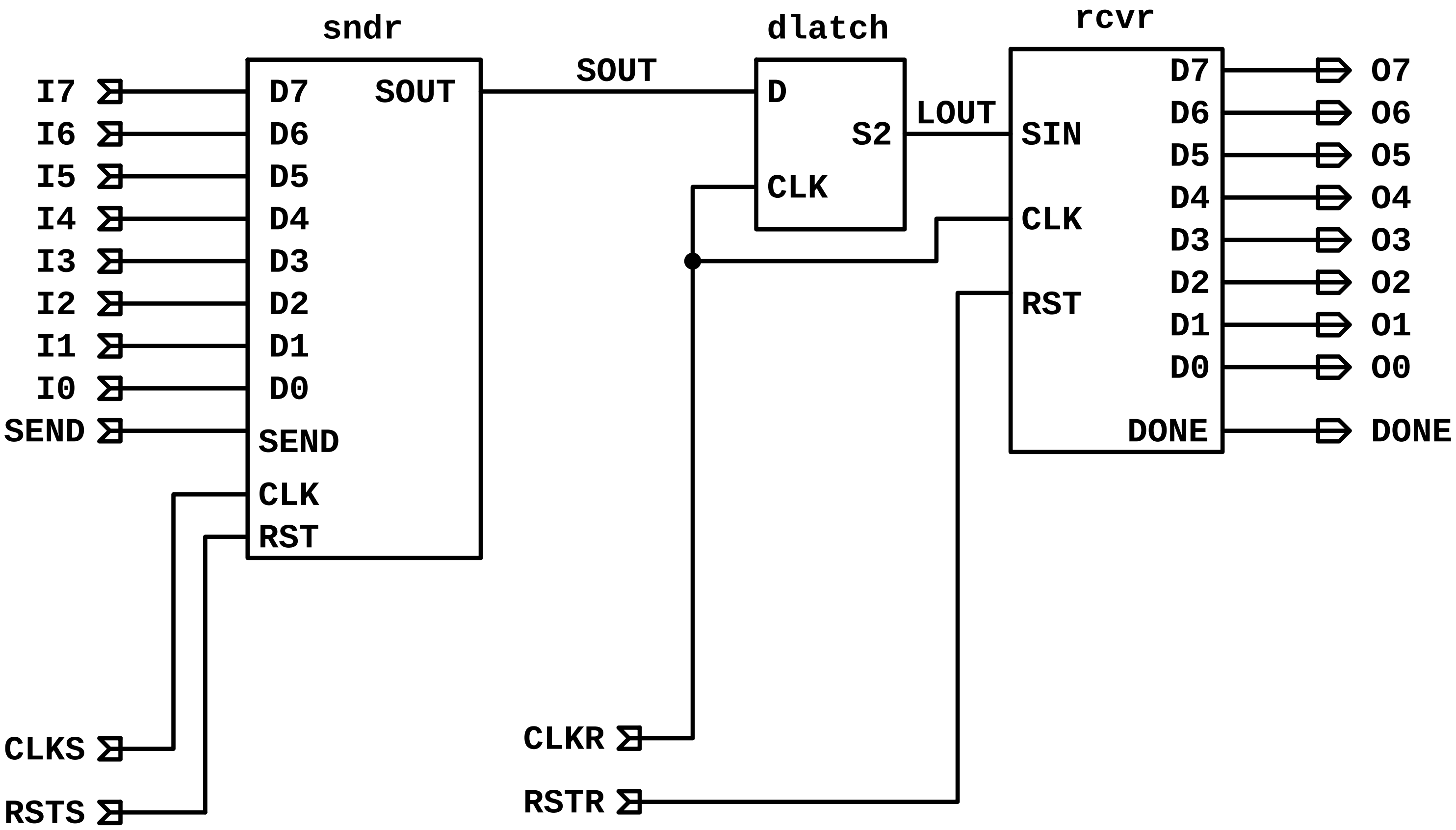
Eliminating Metastability



Theorem 4 *Let* $p = (w_{\mathtt{CLK}}\, w_{\mathtt{D}})$, *where* $w_{\mathtt{CLK}}$ *is a periodic clock pulse with high* ≥ 7000, *low* ≥ 7000, *and successive rising edges at* t_1 *and* t_3, *with intervening falling edge at* t_2. *Let* $w_{\mathtt{S2}}$ *be the output waveform of* $sim(\mathtt{dlatch}, p, t_f)$.

Then $w_{\mathtt{S2}}$ *has a constant value* v *on* $[t_2 + 7000, t_3)$. *If* $w_{\mathtt{D}}$ *has a constant value* u *on the cycle* $[t_1, t_3)$, *then* $u = v$.

Final Circuit



bpm

Correctness Theorem (Sketch)

Theorem 5 *Assume all inputs meet stability constraints and the two clock periods are sufficiently close.*

Suppose `SEND` *has effective value* $\mathcal{T}$ *on exactly 1 cycle, on which the other inputs have effective values* $(d_0 \ \ldots \ d_7)$.

Then the outputs are stable. On the first cycle on which `DONE` *has effective value* $\mathcal{T}$*, the other outputs have effective values* $(d_0 \ldots d_7)$.

CLI
COMPUTATIONAL
LOGIC
INCORPORATED

Copyright © 1994 Computational Logic, Inc.

Proof (Sketch)

- Theorem 3 $\Rightarrow$ output of `sndr` is stable with effective value list $V_r = send(m, (d_7 \ \ldots \ d_0))$.
- Theorem 4 $\Rightarrow$ input to `rcvr` is stable with effective value list V_r.
- Theorem 1 $\Rightarrow V_r = asynch(V_s, t_s, t_r, \pi_s, \pi_r, oracle)$.
- Theorem 2 $\Rightarrow recv(V_r) = (d_7 \ \ldots \ d_0)$.
- Theorem 3 $\Rightarrow$ conclusion.

Copyright © 1994 Computational Logic, Inc.