DIGITAL CIRCUITS AS MOORE MACHINES

A PREPRINT

Victor Yodaiken*
Austin Texas
victor.yodaiken@gmail.com

January 2026

ABSTRACT

Here it is

Keywords Digital Circuits, Sequential Functions, · Moore Machines

1 Introduction

The real-time and compositional behavior of digital circuits can be represented by state machines using a method for working with large numbers of states and for interconnecting state machines described in a previous paper [2]. The method treats state machines as maps from finite sequences of events to the output reached by following that sequence from the initial state. In this case, events represent real-time discrete samples of signals asserted on the input pins of circuits, sampled at some fixed frequency. The sampling rate can be left unspecified or made concrete as needed.

This note will start with a couple of simple combinatorial circuits: a bus and nand-gate, then specify a set-reset latch, and finally show how connecting two nand-gates can implement the latch. A remaining section sketches possible alternative approaches using the same underlying method.

None of this is earthshaking news - these circuits have been widely used and modeled for a long time now and are staples of introductory digital circuit courses. However, defining the real-time behavior of these circuits without the use of a programming language like framework such as in VHDL, simplifies semantics. The reader will have to decide for themselves if there is anything illuminating in representing circuits as functions.

2 Circuits

Let ϵ be the empty sequence and $w.a$ be the sequence obtained by appending sample a to w on the right. A sample is just a tuples of binary values:

$$a = (x_1, \ldots x_n) \text{ where } x_i \in \{0, 1\} \text{ and } (a)_i = x_i.$$

The length of an input sequence w is the amount of time that has passed since the start state – in whatever units of time each sample represents[2]. The state of a circuit state machine M is determined by the sequence of events that has driven the machine from its initial state.

$$\text{Input Sequence } \Rightarrow \boxed{\text{M}} \Rightarrow Outputs$$

The state machines are deterministic but often only partially specified.

For circuit C and event sequence w, $C(w, r)$ is the output on pin r in the state reached by following w from the initial state. If C has only one output pin, we can just write $C(w)$.

*Independent Researcher.

[2]All event sequences here are finite as we are not interested in infinite time periods and infinitisimal sample times can be left to standard analytic methods.

To see how long a signal $b \in \{0,1\}$ has been applied to a particular input pin define:

$$held(\epsilon, i, b) = 0 \text{ and } held(w.a, i, b) = \begin{cases} (1 + held(w, i, b)) & \text{if } (a)_i = b \\ 0 & \text{otherwise.} \end{cases}$$

Equivalently:

$$held(\epsilon, i, b) = 0 \text{ and } held(w.a, i, b) = ((a)_i = b)(1 + held(w, i, b))$$

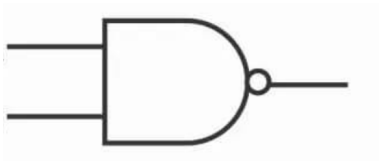
where $((a)_i = b)$ above has value 1 if true and 0 otherwise.

Then for a bus B with k wires, with propagation delay of n samples:

$$\text{If } held(w, 1, b) \geq n \text{ then } B(w, i) = b \tag{1}$$

Say G is a 2 input *nand-gate* with propagation delay n if and only if

$$\begin{array}{l} \mathtt{G}(w) \in \{0,1\} \\ \mathtt{G}(w) = 1 \text{ if } held(w, 1, 0) \geq n \text{ or } held(w, 2, 0) \geq n \\ \mathtt{G}(w) = 0 \text{ if } held(w, 1, 1) \geq n \text{ and } held(w, 2, 1) \geq n \end{array} \tag{2}$$



That is, if either input pin is held low (0 for at least n samples the output is 1 and if both are held high (1) for at least n samples the output must be 0. If neither condition is met, then all we know is that the output is either 0 or 1 (see section 3 for other possibilities).

Now we are going to define a set/reset (SR) latch in two steps. The remarkable thing about this latch is that when it gets to a stable (latched) state (set or reset), the inputs $(1, 1)$ keep it in that state indefinitely. The bus and nand-gates have a property that there is some n so that looking back n samples is sufficient to test e.g. if a signal has been asserted long enough to determine the output. The latch does not have that property, an arbitrary sequence of $(1, 1)$ won't change the latched value, if some value has been latched.

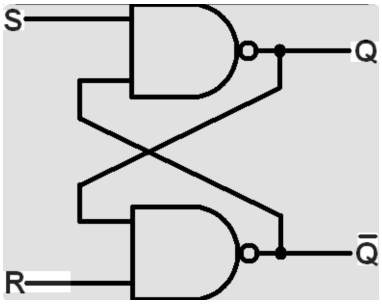
$$\begin{array}{l} Set(\epsilon, n) = 0 \\ Set(w.a, n) = \begin{cases} 1 & \text{if } held(w.a, 1, 0) \geq n \text{ and } held(w.a, 2, 1) \geq n \\ & \text{or if } Set(w, n) = 1 \text{ and } (a)_1 = (a)_2 = 1 \\ 0 & \text{otherwise} \end{cases} \\ Reset(\epsilon, n) = 0 \\ Reset(w.a, n) = \begin{cases} 1 & \text{if } held(w.a, 1, 1) \geq n \text{ and } held(w.a, 2, 0) \geq n \\ & \text{or if } Reset(w, n) = 1 \text{ and } (a)_1 = (a)_2 = 1 \\ 0 & \text{otherwise} \end{cases} \end{array} \tag{3}$$

As usual, input $(0, 0)$ is not used.

L is a SR latch with latch time n if and only if

$$\begin{array}{l} L(w, 1) \in \{0,1\}, L(w, 2) \in \{0,1\}. \\ \text{If } Reset(w, n) = 1 \text{ then } L(w, 1) = 1 \text{ and } L(w, 2) = 0 \\ \text{If } Set(w, n) = 1 \text{ then } L(w, 1) = 0 \text{ and } L(w, 2) = 1 \end{array} \tag{4}$$

Build a latch by cross connecting two nand-gates.



Let $\texttt{SR}(w,1) = \texttt{G}_1(w_1)$ and $\texttt{SR}(w,2) = \texttt{G}_2(w_2)$ where

- $\texttt{G}_1$ and $\texttt{G}_2$ are nand-gates with gate delay n
- and each $w_i = w_i(w)$ where:
  - $w_1(\epsilon) = \epsilon$ and $w_1(w.a) = w_1(w).((a)_1, \texttt{G}_2(w_2))$,
  - and $w_2(\epsilon) = \epsilon$ and $w_2(w.a) = w_2(w).((a_2, \texttt{G}_1(w_1))$.

The composition used to construct SR is basically a function form of the *concurrent product* of state machines [2, 1].

Claim: SR is a SR latch with latch time $3n + 3$.
Proof:
Note that $held(w,1,b) = held(w_1,1,b)$ and $held(w,2,b) = held(w_2,1,b)$. These are just passed through and are obviously true for $w = \epsilon$ and if true for w must hold for $w.a$ by the inductive hypothesis and the definitions of w_i.

It follows that: $held(w,1,0) \geq n$ implies $held(w_1,1,0) \geq n$ which implies that $\texttt{G}_1(w_1(w)) = 1$. Now note for any k, $held(w,1,0) \geq n + k$ implies $held(w_2(w),1,1) \geq k$. Proof by induction on k. For $k = 0$ there is nothing to prove. Suppose the implication holds for $held(w,1,0) = n + k$ and consider what happens if $held(w.a,1,0) = n + k + 1$. Since we know $\texttt{G}_1(w_1(w)) = 1$ if $held(w_2(w),1,1) = n + k$ it follows that $held(w_2(w.a),1,1) = k + 1$. So setting $k = n$ $held(w,1,0) \geq 2n$ tells us that $held(w_2,1,1) \geq n$ and given $held(w,2,1) \geq 2n$ we know $\texttt{G}_2(w_2) = 0$. Similar arguments take us to $held(w,2,1) \geq 3n$ implies $held(w_1(w),2,0) \geq n$. So for $a = (1,1)$ $\texttt{G}_1(w_1(w.a)) = 1$. The case for inputs $(1,0)$ is parallel.

3 More granular approaches

The model presented above is particularly simple. One addition might be to specify hysterisis for the gates. This would allow a tighter bound on timing for the latch.

$$SinceStable(\epsilon,k) = 0$$

$$SinceStable(w.a,k) = \begin{cases} k & \text{if } held(w,1,0) > n \text{ or } held(w,2,0) > n \\ & \text{or } held(w,1,1) > n \text{ and } held(w,2,1) > n \\ k-1 & \text{if } SinceStable(w.k) > 0 \\ & \text{and } \neg(held(w,1,0) > n \text{ or } held(w,2,0) > n) \\ & \text{and } \neg(held(w,1,1) > n \text{ and } held(w,2,1) > n) \\ 0 & \text{otherwise} \end{cases}$$

Then $SinceStable(w,k) > 0$ implies $\texttt{G}(w.a) = \texttt{G}(w)$.

Alternatively, in some circumstances these circuits could be represented with inputs and outputs over a bigger range: say $\{0, 0.01, \ldots 4\}$ Or perhaps adding an unknown value would be useful.

References

[1] J. Hartmanis. "Loop-free structure of sequential machines". In: *Sequential Machines: Selected Papers*. Ed. by E.F. Moore. Reading MA: Addison-Welsey, 1964, pp. 115–156.

[2] Victor Yodaiken. "State Machines for Large Scale Computer Software and Systems". In: *Form. Asp. Comput.* 36.2 (June 2024). ISSN: 0934-5043. DOI: 10.1145/3633786. URL: https://doi.org/10.1145/3633786.