ISSN1000-9825 Journal of Software 软件学报 2000,11(5):569~583

# An Application of Domain Theory to Logical Design of VLSI Circuits*

SUN Yong[1] HU Yi[2]

[1](School of Computer Science The Queen's University of Belfast Northern Ireland)

[2](Integrated Silicon Systems Limited Northern Ireland)

E-mail: y.sun@qub.ac.uk/yhu@iss-dsp.com

**Abstract** This paper is to suggest that traditional 2-valued Boolean algebra is not sufficient for representation of VLSI circuits at logic gate level, although it is the case for combinational circuits. Instead of using the Register-and-Transfer technique at RT level to represent sequential circuits, an alternative is sought and found. That is, all uncertain voltages such as oscillations and floating voltages are identified by the same value, denoted as $\perp$ (called *bottom*). The two certain voltages are, as usual, *ground* and *power*; they are denoted by 0 and 1 respectively. An *invertor*, a *nor-gate* and a *nand-gate* are defined according to the physics of VLSI circuits instead of the Boolean algebra. As a result, a logic is obtained which coincides with Kleene 3-valued logic provided that Kleene's $u=\perp$. As it is well known, Kleene 3-valued logic is functionally *incomplete*. This means that not every function (or gate) can be constructed from invertors, nor-gates and nand-gates. However, by introducing a partial order $\sqsubseteq$ into the logic, by using general fixed-point operators instead of the least fixpoint operator to deal with feedbacks in VLSI circuits, and by applying CPO (complete partial order) domain theory to the derived 3-valued logic system, the result obtained means that this system is functionally *monotonic complete*. Also, the canonical normal forms for this Kleene 3-valued logic are obtained. Although the present results are mainly semantic, it is very interesting in pursuing the research further by investigating the syntactical derivability. Such research would derive more secure circuits close to reality, which is to make the work compatible with VHDL, Verilog HDL and/or EDIF. Incidentally, Mukaidono has obtained similar results, although his approach is not as coherent as the one presented in this paper.

**Key words** VLSI circuit, logic gate, Kleene 3-valued logic, Complete Partial Order (CPO), general fixpoint operator, monotonicity.

* **SUN Yong** is a lecturer in the School of Computer Science, the Queen's University of Belfast (Northern Ireland). He received a Ph.D. degree from the University of Edinburgh (Scotland). His current research areas include intelligent tutoring systems, artificial intelligence, formal methods, theoretical computer science, computer science education, and foundations of computer science. **HU Yi** is the Chief Technologist with the Integrated Silicon Systems Ltd (Northern Ireland). He received a Ph.D. from the Queen's University of Belfast (Northern Ireland) in 1992. In China he was a Research Engineer involved in the development of VLSI CAD tools and the design of integrated circuits. In Belfast he has developed an ASIC design automation tool, with special application to DSP chip design. In ISS he has implemented a library of parameterisable and synthesisable VHDL functions for DSP ASICs, as well as a range of complex DSP cores including MPEG, JPEG, FFT and ADPCM. He has wide ranging design experience including all aspects of complex chip design from algorithm development through to ASIC layout level.

Manuscript received 1998-09-01, accepted 1998-12-24.

## 1 Introduction and Overview

Traditionally, 2-valued Boolean functions are used to represent VLSI circuits at gate-level (or logical-level). 0 represents the low voltage (*GROUND*) and 1 represents the high voltage (*POWER*). For instance, an INVERTOR, a NAND-gate, and a NOR-gate are defined as follows:

1. INVERTOR

| $x$ | 0 | 1 |
|---|---|---|
| $f_{\neg}(x)$ | 1 | 0 |

2. NAND

| $x$ | 0 | 0 | 1 | 1 |
|---|---|---|---|---|
| $y$ | 0 | 1 | 0 | 1 |
| $f_{\mathrm{NAND}}(x,y)$ | 1 | 1 | 1 | 0 |

3. NOR

| $x$ | 0 | 0 | 1 | 1 |
|---|---|---|---|---|
| $y$ | 0 | 1 | 0 | 1 |
| $f_{\mathrm{NOR}}(x,y)$ | 1 | 0 | 0 | 0 |

The application of 2-valued Boolean algebra to combinational circuits is very successful. However, this is not necessary true for sequential circuits. It is simply because that 2-valued model excludes the possibility of representing floating voltages or oscillations within the model. The following example illustrates this point.

Consider a NAND-gate with a feedback represented by the equation (see Fig. 1):

$$z=f_{\mathrm{NAND}}(x,z) \tag{1.1}$$

where $f_{\mathrm{NAND}}(x,y)=\neg(x\wedge y)$ (the function $f_{\mathrm{NAND}}$ is called the *combinational* function of Eq. (1.1)).


Fig. 1

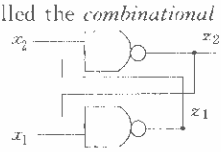


Fig. 2

Assume that the input on line $x$ is 1; then, regardless of the input on line $z$ (either 1 or 0), the output value on line $z$ is always the complement of the input value. Since the input $z$ and the output $z$ are connected, their values should be identical. However, they can be neither 0 nor 1, i.e. the value on line $z$ cannot be represented in a 2-valued model.

To overcome this problem, the Register-and-Transfer technique has been used. However, we suggest that this solution is unsatisfactory and does not correspond to the physics of VLSI circuits.

Another example in uncertain voltages occurs in a flip-flop, see Fig. 2.

Assume that $x_1=x_2=1$. The question arises: what are the values on lines $z_1$ and $z_2$?

There are two possibilities for their stable values:

(i) $z_1=0$ and $z_2=1$, or

(ii) $z_1=1$ and $z_2=0$.

Which of these two occurs is completely dependent on the details of the particular circuit (i.e. the implementation or the delay of the individual gates, so-called *racing* hazard). Assuming one or other of the two is too arbitrary. Allowing both possibilities requires the modelling of a complicated *non-determinism* phenomenon. As a compromise, one value to designate uncertainty seems to be a better choice.

Thus, we introduce new values into 2-valued switching theory to preserve the advantage of digital logic in

circuit design and to solve the problem illustrated earlier at the same time.

For our present purpose, the design of VLSI systems, uncertain inputs would eventually lead to uncertain outputs anyway. In this sense, a single extra value $\perp$ (besides 0 and 1) is sufficient. In other words, examples of uncertain voltages such as glitches, oscillations, floating voltages and initial (internal) voltages of flip-flops are considered to be the single value $\perp$. Of course, by introducing more values into switching theory, we can reap some other benefits (see Refs. [1~4] for some references). However, this is outside the main interest of this paper.

We define our semantics domain as a natural extenion of $B=\{0,1\}$, i.e. $B_\perp=B\cup\{\perp\}$ with its usual natural partial order $\sqsubseteq$ (i.e. $\perp\sqsubseteq 0$ and $\perp\sqsubseteq 1$). That is, $x\sqsubseteq y$ means that the certainty of the voltage on $y$ is greater than that on $x$.

We can extend the partial order $\sqsubseteq$ pointwise to any product of $B_\perp$. The definitions of *monotonic* and *continuous* are as usual:

(a) $f$ is *monotonic* iff $f(x)\sqsubseteq f(y)$ for each pair $x\sqsubseteq y$;

(b) $f$ is *continuous* iff $\lim_n\{f(x_n)\}=f(\lim_n\{x_n\})$ for every (monotonic) sequence $\{x_n\}$.

We now give definitions for an INVERTOR (negation), an AND-gate (conjunction) and an OR gate (disjunction), in the extended $B_\perp$ as follows:

1. negation

| $x$ | $\perp$ | 0 | 1 |
|---|---|---|---|
| $f_\neg(x)$ | $\perp$ | 1 | 0 |

2. conjunction

| $x$ | $\perp$ | $\perp$ | $\perp$ | 0 | 1 | 0 | 0 | 1 | 1 |
|---|---|---|---|---|---|---|---|---|---|
| $y$ | $\perp$ | 0 | 1 | $\perp$ | $\perp$ | 0 | 1 | 0 | 1 |
| $f_\wedge(x,y)$ | $\perp$ | 0 | $\perp$ | 0 | $\perp$ | 0 | 0 | 0 | 1 |

3. disjunction

| $x$ | $\perp$ | $\perp$ | $\perp$ | 0 | 1 | 0 | 0 | 1 | 1 |
|---|---|---|---|---|---|---|---|---|---|
| $y$ | $\perp$ | 0 | 1 | $\perp$ | $\perp$ | 0 | 1 | 0 | 1 |
| $f_\vee(x,y)$ | $\perp$ | $\perp$ | 1 | $\perp$ | 1 | 0 | 1 | 1 | 1 |

The movitation for these definitions is as follows:

(1′) If for an INVERTOR (negation) its input is uncertain, so is its output.

(2′) If for an AND-gate (conjunction) one of its inputs is 0, and then whatever value its other input has, its output is determined by the input 0, i.e. its output is 0; if one of its two inputs is 1, then its output cannot be determined by this input; we have to know the value of its other input before we know its output with certainty.

(3′) If for an OR-gate (disjunction) one of its two inputs is 1, then whatever value its other input is, its output is determined by the input 1, i.e. its output is 1; if one of its two inputs is 0, and then its output cannot be determined by this input; we have to know the value of its other input before we know its output with certainty.

Note that the conjunction and disjunction operations defined above are not strict (although they are monotonic and continuous) and they together with the above defined negation coincide with the 3-valued logic defined by Kleene[5].

Now, we introduce a general fixpoint operator (see Ref. [6] for further details of general fixpoint operators), instead of the least fixpoint operator (which is commonly used in computation theory[7~9]).

Consider the example (see Fig. 1) as a way of introducing general fixpoint operators. Suppose the external

input $x=1$ and the initial internal input $z_0$(i.e. the isolated voltage) is 0. Then, the question is: *what is the possible output on line z?*

To obtain a reasonable answer, we have the following analysis:

$$\begin{cases} z_{2n+1}=f_{NAND}(1,z_{2n})=\neg z_{2n}=1, & \text{and} \\ z_{2n+2}=f_{NAND}(1,z_{2n+1})=\neg z_{2n+1}=0. \end{cases}$$

We understand that the sequence of $\{z_n\}$ is *divergent*, written as $\lim_{n\to\infty}\{z_n\}=\perp$. In the physics of VLSI circuits, this divergence indicates that the output voltage of the gate is *oscillating*. However, if $x=z_0=0$, then $z_n=1$ for all $n\geqslant 0$; i.e. the sequence $\{z_n\}$ is *convergent*, written as $\lim_{n\to\infty}\{z_n\}=1$, and we say that when $x=0$,

(i) both 0 and 1 are convergence points of Eq. (1.1); and

(ii) 1 is the fixpoint of Eq. (1.1) under the condition that either $z_0=0$ or $z_0=1$.

Thus,

(a) the output in the first case (i.e. $x=1$) should be the uncertain value, $\perp$, and

(b) the output in the second case (i.e. $x=0$) should be 1.

However, on the other hand, when $x=1$, the least fixpoint of Eq. (1.1) is $\perp$; i.e. the result of the least fixpoint of Eq. (1.1) under $x=1$ is not the result of the actual output on line $z$ in the circuit of Fig. 1. Therefore, we have no choice but to use the general fixpoint operators defined as follows instead of the least fixpoint operator.

Formally, given a (combinational) function $f:\vec{B}_\perp\to\vec{B}'_\perp$ and the initial values $\vec{b}_0\in\vec{B}_\perp$, where $\vec{B}_\perp$ is $B_\perp^m$ for some $m\geqslant 1$ (or $\underbrace{B_\perp\times B_\perp\times\cdots\times B_\perp}_{m\text{ times}}$) and $\vec{B}'_\perp$ is $B_\perp^n$ for some $n\geqslant 1$ (or $\underbrace{B_\perp\times B_\perp\times\cdots\times B_\perp}_{n\text{ times}}$) ($m\geqslant n$), the output of $f$ can be defined:

1. either by the equation

$$\vec{z}=f(x_1,x_2,\ldots,x_m,\vec{z}),$$

where $\vec{z}=(z_1,z_2,\ldots,z_n)$, $x_i$ is the $i$th external input of $f$, and $z_j$ is the $j$th internal input (feedback) of $f$;

2. or by using the general fixpoint operator $fix_{B_\perp^n}$, i.e. $fix_{B_\perp^n}(curry(f)(x_1,x_2,\ldots,x_{m-n}))(\vec{z}_0)$, where

(a) $curry:(B_\perp^m\to B_\perp^n)\times B_\perp^{m-n}\to B_\perp^n$ is such that

$$curry(f)(x_1,x_2,\ldots,x_{m-n})(z_1,z_2,\ldots,z_n)=f(x_1,x_2,\ldots,x_{m-n},z_1,z_2,\ldots,z_n),$$

(b) $fix_{B_\perp^n}:(B_\perp^n\to B_\perp^n)\times B_\perp^n\to B_\perp^n$ is such that

$$fix_{B_\perp^n}(f)(\vec{b}_0)=\begin{cases} f(\lim_{i\to\infty}\vec{b}_i) & \text{if } f \text{ is convergent at } \vec{b}_0 \\ fix_{B_\perp^n}(f)(\vec{b}_0^*) & \text{otherwise} \end{cases},$$

where $\vec{b}_{i+1}=f(\vec{b}_i)$ for $i\geqslant 0$ and $\vec{b}_0^*=(b_{1,0}^*,b_{2,0}^*,\ldots,b_{n,0}^*)$ with

$$b_{i,0}^*=\begin{cases} \lim_{k\to\infty}b_{i,k} & \text{if } \{b_{i,k}\} \text{ is convergent} \\ \perp & \text{otherwise} \end{cases}.$$

By the well-founded property of the partial order $\sqsubseteq$ in products of $B_\perp$, and by the least fixpoint theorem in CPO (complete partial order) domain theory[7~9], we know that $fix_{B_\perp^n}$ has its value if $f$ is monotonic.

Technologically, we focus our attention on hardware which can be built by one-input and one-ouput INVERTORs, two input and one-output NAND-gates, two-input and one-output NOR-gates; theoretically, we need to show that this collection of basic circuits (or gates) is functionally complete. Unfortunately, this is impossible since Kleene 3-valued logic is well-known to be functionally incomplete. However, in this paper, we show that our model is functionally monotonic complete. In other words, the Kleene 3-valued logic is functionally monotonic complete. This implies that all gates have the following property: *whenever the certainty of input voltages increases, so does the certainty of the output voltages.*