Relating two models of hardware

by
Glynn Winskel
University of Cambridge,
Computer Laboratory,
Corn Exchange Street,
Cambridge CB2 3QG.

The idea of this note is to show how Winskel's static-configuration model of circuits in [W] is related formally to Gordon's relational model in [G, G1]. Once so related the simpler proofs in the model in [G] can, for instance, be used to justify results in terms of the model in [W]. More importantly, we can exhibit general conditions on circuits which ensure that assertions which hold of a circuit according to the simpler model are correct with respect to the more accurate model. The formal translation makes use of a simple adjunction between (partial order) categories associated with the two models in [W] and [G], in a way reminiscient of abstract interpretation [CC]. Preliminary results suggest similar lines of approach may work for other kinds of abstraction such as temporal abstraction in reasoning about hardware (see [M]), and, more generally, make possible a formal algebraic treatment of the relationship between different models of hardware.

1. Formalising abstraction.

The models of hardware we shall relate fit into a general scheme. In many models a circuit is represented by its set of possible behaviours. So assume a circuit c denotes a subset $[\![c]\!] \subseteq B$ of *behaviours* according to a model. It will be the case that each behaviour $b \in B$ will possess structure which can be described by a *behaviour assertion.* Such an assertion A denotes a subset of behaviours $[\![A]\!] \subseteq B$ consisting of those behaviours which satisfy it. A *circuit specification Spec* should pick-out those circuits which satisfy it and so we expect it to denote a subset $[\![Spec]\!] \subseteq P(B)$. There are two obvious ways a behaviour assertion A can be made into into a basic circuit specification. Firstly, we say a circuit c satisfies a circuit specification $\diamond A$ when it has some behaviour which satisfies A, or, more formally, we can write

$$c \models \diamond A \text{ iff } [\![c]\!] \cap [\![A]\!] \neq \emptyset.$$

Secondly, we say c satisfies $\Box A$ when all its behaviours satisfy A, *i.e.*

$$c \models \Box A \text{ iff } [\![c]\!] \subseteq [\![A]\!].$$

Of course more complicated circuit specifications can be built up by taking conjunctions, for instance, of such basic ones. In the models we shall consider behaviour assertions will have negations, with the negation $\neg A$ of an assertion A being denoted by the complement of $[\![A]\!]$. Hence we shall have

$$c \models \diamond A \text{ iff } [\![c]\!] \not\subseteq [\![\neg A]\!].$$

Thus determining if a circuit satisfies either kind of basic circuit specification reduces to considering whether or not an inclusion

$$[\![c]\!] \subseteq [\![A]\!]$$

holds for the circuit term c and a behaviour assertion A.

The key to relating two models is to find conditions under which such an inclusion in one model implies such an inclusion in another. To this end, imagine two models, model 1 and model 2, for hardware behaviour, and that model 1 is more detailed and accurate than model 2. Assume that both models are based on their respective notions of behaviour which form sets B_1 and B_2. With luck, the fact that model 1 is more detailed than 2 will be expressed through there being an "abstraction function" from the behaviours of 1 to the behaviours of 2; it is intended that such a function shows how a more detailed behaviour can be viewed as a less detailed behaviour. Sometimes a more detailed behaviour may be outside the scope of the less detailed model so we cannot expect the function to be always defined. To reflect this, the abstraction function will be a partial function

$$abs : B_1 \rightharpoonup B_2.$$

How are the two inclusions of models 1 and 2 related by the abstraction function which exists between their associated representations of behaviour? Certainly the function *abs* extends to sets; define

$$\begin{aligned} &abs_* : P(B_1) \to P(B_2) \text{ by taking} \\ &abs_*(C) = \{abs(b_1) \mid b_1 \in C \;\; \& \;\; abs(b_1) \downarrow\} \end{aligned}$$

for $C \in P(B_1)$. (We use $abs(b_1) \downarrow$ to mean $abs(b_1)$ is defined and $abs(b_1) \uparrow$ for $abs(b_1)$ is undefined.) Given a subset C of B_1 the function abs_* yields its image under *abs*. Accompanying the function abs_* is another function

$$\begin{aligned} &abs^* : P(B_2) \to P(B_1) \text{ by taking} \\ &abs^*(A) = \{b_1 \mid abs(b_1) \downarrow \Rightarrow abs(b_1) \in A\} \end{aligned}$$

for $A \in P(B_2)$. Given a subset A of B_2 the function abs^* yields the largest subset of B_1 whose image under *abs* lies in A. It is easy to see that the pair of functions form an adjunction in the following sense.

1.1 Proposition. *For any $C \in P(B_1)$ and $A \in P(B_2)$,*

$$C \subseteq abs^*(A) \Leftrightarrow abs_*(C) \subseteq A,$$

a property which says the pair abs_*, abs^* forms an adjunction from $(P(B_1), \subseteq)$ to $(P(B_2), \subseteq)$ with left adjoint abs_* and right adjoint abs^*.

Further, if abs is onto (i.e. $abs_*(B_1) = B_2$) then

$$C \subseteq abs^* \circ abs_*(C) \text{ and } A = abs_* \circ abs^*(A),$$

for any $C \in P(B_1)$ and $A \in P(B_2)$.

For adjunctions like this some further facts hold which we record for later use.

1.2 Proposition. *Let $abs : B_1 \rightharpoonup B_2$ be a partial function, with abs^* defined as above. Then*

$$\begin{aligned} &abs^*(B_2) = B_1, \qquad abs^*(\emptyset) = \{x \in B_1 \mid abs(x) \uparrow\}, \\ &abs^*(X \cap Y) = abs^*(X) \cap abs^*(Y), \\ &abs^*(X \cup Y) = abs^*(X) \cup abs^*(Y), \\ &abs^*(B_2 \setminus A) = \{x \in B_1 \mid abs(x) \uparrow\} \cup (B_1 \setminus abs^*(A)), \end{aligned}$$

The facts expressed in 1.1 and 1.2 are fairly obvious. Still, they are significant because such an adjunction expresses how inclusion according to a more detailed model is related to inclusion in the less detailed one. Of course this is for inclusion and is just between sets, and does not involve the syntax used in the models to build circuit terms and assertions. However proposition 1.2 expresses how boolean operations of conjunction, disjunction and negation on assertions in the less detailed model translate to the more detailed one, and so will enable a smooth translation from assertions built using these connectives in model 2 to "equivalent" assertions of model 1.

In the two models we shall consider, circuit terms of the more detailed model will include those of the less detailed model, while the syntax of assertions will differ. Letting c be a circuit term and A an assertion of model 2 (the less detailed model), and using $[\![\;]\!]_1$ and $[\![\;]\!]_2$ for the semantic functions associating subsets of behaviours with terms and assertion in the two models, the proposition above gives

$$[\![c]\!]_1 \subseteq abs^*([\![A]\!]_2) \Leftrightarrow abs_*([\![c]\!]_1) \subseteq [\![A]\!]_2. \qquad (*)$$

As it stands this is not wholly satisfactory. Central to the models 1 and 2 are the two inclusion relations involving *terms* and *assertions*

$$[\![c]\!]_1 \subseteq [\![A_1]\!]_1 \quad \text{and} \quad [\![c]\!]_2 \subseteq [\![A_2]\!]_2,$$

where c is a circuit term and A_1 is an assertion in model 1, and A_2 is an assertion in model 2. The equivalence $(*)$ does not yet relate these inclusions directly because, obviously, $abs^*([\![A]\!]_2)$ is not an assertion and $abs_*([\![c]\!]_1)$ is not a term. However, for the two models we shall consider, for an assertion A of model 2, it is quite easy (using proposition 1.2) to construct uniformly an assertion of model 1, call it $^*A$, which denotes the same behaviours as $abs^*([\![A]\!]_2)$ *i.e.* so

$$[\![^*A]\!]_1 = abs^*([\![A]\!]_2).$$

Then asking for the two models to agree on the assertions a circuit term c satisfies amounts to requiring a condition on circuit terms c so that

$$[\![c]\!]_2 = abs_*([\![c]\!]_1),$$

because then and only then do we have

$$[\![c]\!]_1 \subseteq [\![^*A]\!]_1 \text{ iff } [\![c]\!]_2 \subseteq [\![A]\!]_2,$$

for any assertion A of model 2. Then, by definition, we obtain

$$c \models_1 \Box\, ^*A \text{ iff } c \models_2 \Box\, A$$

directly and, as we shall see, a similar result holds for circuit specifications $\Diamond A$. Thus a condition on circuits ensures the two models agree. One can, in addition, ask for weaker conditions on circuits which ensure that assertions established for a circuit in one model guarantees that the corresponding assertion holds in the other, but not necessarily the converse. For the two models we shall consider the conditions will amount to simple and intuitively reasonable restrictions on the way circuit terms are built up.

We make some remarks about the present state of hardware verification. Up till now a great deal of hardware verification has focussed on establishing that circuits c meet specifications of the form $\Box\, A$—as we have seen this amounts to showing $[\![c]\!] \subseteq [\![A]\!]$—and ignored the question of whether or not a circuit satisfies specifications $\Diamond A$. However it appears both kinds of modal formulae should be considered as circuit specifications, and the effect of insisting a circuit satisfies a specification $\Diamond A$ is sometimes achieved by imposing some requirement expressed in higher-order logic (see *e.g.* section 10 of [G1]). After all a short circuit obtained by connecting power to ground denotes the emptyset in the model of [G, G1] and so satisfies any specifications of the form $\Box\, A$. This indicates the lack of expressiveness of specifications purely of the form $\Box\, A$. This problem was discussed by Mike Fourman in [F] though his proposal on how to extend specifications was less specific. The proposal here to introduce two kinds of modal assertions as basic specifications is mathematically obvious and, while much simpler, follows the same lines as used in other areas of semantics; the powerdomains of denotational semantics can be seen as spaces whose basic open sets are described by such modal formulae (see [W, R]).

2. Circuit terms.

We simplify the language in [W], ignoring the components responsible for charge storage and resistance because these are not addressed in the model in [G, G1]. Terms for circuits have the following form.

$$c ::= \mathit{Pow}\ (\alpha) \mid \mathit{Gnd}\ (\alpha) \mid \mathit{wre}(\alpha,\beta) \mid \mathit{ntran}(\alpha,\beta,\gamma) \mid \mathit{ptran}(\alpha,\beta,\gamma) \mid c \bullet c \mid c \setminus \alpha.$$

This assumes a set of points (point names) $\alpha, \beta, \gamma, \dots \in \Pi$, and we assume α, β, γ are distinct point names in $\mathit{ntran}(\alpha,\beta,\gamma)$, $\mathit{ptran}(\alpha,\beta,\gamma)$, $\mathit{wre}(\alpha,\beta)$. The term $\mathit{wre}(\alpha,\beta)$ will denote a wire connecting α and β, the term $\mathit{ntran}(\alpha,\beta,\gamma)$ an n-type transistor with gate γ, the term $\mathit{ptran}(\alpha,\beta,\gamma)$ a p-type transistor with gate γ, while the composition $c_0 \bullet c_1$ joins two circuits c_0 and c_1 at their common points and the hiding operation $c \setminus \alpha$ insulates from the environment the point α.

We define the sort of term by structural induction.

$$\begin{aligned}
&\mathrm{sort}(\mathit{Pow}\ (\alpha)) = \mathrm{sort}(\mathit{Gnd}\ (\alpha)) = \{\alpha\}\\
&\mathrm{sort}(\mathit{wre}(\alpha,\beta)) = \{\alpha,\beta\}\\
&\mathrm{sort}(\mathit{ntran}(\alpha,\beta,\gamma)) = \mathrm{sort}(\mathit{ptran}(\alpha,\beta,\gamma)) = \{\alpha,\beta,\gamma\}\\
&\mathrm{sort}(c \bullet d) = \mathrm{sort}(c) \cup \mathrm{sort}(d)\\
&\mathrm{sort}(c \setminus \alpha) = \mathrm{sort}(c) \setminus \{\alpha\}.
\end{aligned}$$

3. A relational model.

We explain the model in [G, G1] used by Mike Gordon and others. The presentation is a little different from usual, because we concentrate more on the model and do not present circuits as just special kinds of assertions. However the equivalence with the model in [G, G1] is clear.

We assume the set of points Π, and distinct values H and L, standing for high and low, and define, for $\Lambda \subseteq \Pi$,

$$F[\Lambda] = \{V \mid V : \Lambda \to \{H, L\}\}.$$

Write $F = \bigcup_{\Lambda \subseteq \Pi} F[\Lambda]$, and say $V \in F$ has *sort* Λ if $V \in F[\Lambda]$.

A circuit term of sort Λ will denote a subset of $F[\Lambda]$, following the idea that a circuit imposes a relation between values at points.

3.1 Notation. Let $k \in \{H, L\}$. Let $\alpha \in \Pi$. For $V \in F[\Lambda]$ define $V[k/\alpha] \in F[\Lambda \cup \{\alpha\}]$ by taking

$$V[k/\alpha](\beta) = \begin{cases} V(\beta) & \text{if } \beta \neq \alpha,\\ k & \text{if } \beta = \alpha,\end{cases}$$

for $\beta \in \Lambda$.

In this model assertions for expressing the properties of circuits have the following syntax:

Variables: We assume a set of variables

$$\{v_\alpha \mid \alpha \in \Pi\}.$$

Value terms: Terms, denoting values in $\{H, L\}$ have the form

$$t ::= v_\alpha \mid H \mid L.$$

Behaviour assertions: The set of G assertions is generated by

$$A ::= t_0 = t_1 \mid A_0 \wedge A_1 \mid A_0 \vee A_1 \mid \neg A \mid \mathbf{tt} \mid \mathbf{ff} \mid \exists v_\alpha.A \mid \forall v_\alpha.A.$$

We shall use assertions such as $A \to B$ (A implies B) with the understanding that this abbreviates $\neg A \vee B$, and $A \leftrightarrow B$ for $A \to B \wedge B \to A$.

Semantically, a value term denotes a partial function $[\![t]\!] : F[\Lambda] \rightharpoonup \{H, L\}$; we take

$$[\![v_\alpha]\!]V = \begin{cases} V(\alpha) & \text{if } \alpha \in \text{sort}(V), \\ \text{undefined} & \text{otherwise,} \end{cases}$$
$$[\![H]\!]V = H, \quad \text{and} \quad [\![L]\!]V = L$$

for any $V \in F$.

For an assertion A we define $\mathrm{G}[\![A]\!] \in P(F)$ by induction on the structure of A:

$$\begin{aligned}
&\mathrm{G}[\![t_0 = t_1]\!] = \{V \in F \mid [\![t_0]\!]V \downarrow \;\&\; [\![t_1]\!] \downarrow \;\&\; [\![t_0]\!]V = [\![t_1]\!]V\} \\
&\mathrm{G}[\![A_0 \wedge A_1]\!] = \mathrm{G}[\![A_0]\!] \cap \mathrm{G}[\![A_1]\!], \\
&\mathrm{G}[\![A_0 \vee A_1]\!] = \mathrm{G}[\![A_0]\!] \cup \mathrm{G}[\![A_1]\!], \\
&\mathrm{G}[\![\mathrm{tt}]\!] = F, \qquad \mathrm{G}[\![\mathrm{ff}]\!] = \emptyset, \\
&\mathrm{G}[\![\neg A]\!] = F \setminus \mathrm{G}[\![A]\!], \\
&\mathrm{G}[\![\exists v_\alpha . A]\!] = \{V \in F \mid \exists k \in \{H, L\}.\ V[k/\alpha] \in \mathrm{G}[\![A]\!]\}, \\
&\mathrm{G}[\![\forall v_\alpha . A]\!] = \{V \in F \mid \forall k \in \{H, L\}.\ V[k/\alpha] \in \mathrm{G}[\![A]\!]\}.
\end{aligned}$$

Of course we have

$$\begin{aligned}
&\mathrm{G}[\![\exists v_\alpha . A]\!] = \{V \in F \mid V[H/\alpha] \in \mathrm{G}[\![A]\!] \text{ or } V[L/\alpha] \in \mathrm{G}[\![A]\!]\}, \text{ and} \\
&\mathrm{G}[\![\forall v_\alpha . A]\!] = \{V \in F \mid V[H/\alpha] \in \mathrm{G}[\![A]\!] \;\&\; V[L/\alpha] \in \mathrm{G}[\![A]\!]\}.
\end{aligned}$$

We write $\mathrm{G}[\![A]\!]_\Lambda$ for $\mathrm{G}[\![A]\!] \cap F[\Lambda]$, where $\Lambda \subseteq \Pi$.

Semantics of circuit terms: In line with the model in [G, G1], we denote a circuit term c of sort Λ by a subset $\mathrm{G}[\![c]\!]$ of $F[\Lambda]$. The semantics as we describe it follows that of [C] closely. We first define operations $\bullet$ and $\backslash\alpha$ on elements of F.

Let $V_0 \in F[\Lambda_0]$, $V_1 \in F[\Lambda_1]$. Define

$$V_0 \bullet V_1 = \begin{cases} V_0 \cup V_1 & \text{if } V_0 \lceil \Lambda_1 = V_1 \lceil \Lambda_0, \\ \text{undefined} & \text{otherwise.} \end{cases}$$

Let $V \in F[\Lambda]$. Let $\alpha \in \Pi$. Write

$$V \setminus \alpha = V \lceil (\Lambda \setminus \{\alpha\}).$$

We extend the operations to subsets of F. For $R, R_0, R_1 \in P(F)$ define

$$\begin{aligned}
&R_0 \bullet R_1 = \{V_0 \bullet V_1 \mid V_0 \in R_0 \;\&\; V_1 \in R_1 \;\&\; V_0 \bullet V_1 \downarrow\} \\
&R \setminus \alpha = \{V \setminus \alpha \mid V \in R\}.
\end{aligned}$$

Now we define the semantics of circuit terms:

$$\begin{aligned}
&\mathrm{G}[\![\mathit{Pow}\,(\alpha)]\!] = \{V \in F[\{\alpha\}] \mid V(\alpha) = H\} \\
&\mathrm{G}[\![\mathit{Gnd}\,(\alpha)]\!] = \{V \in F[\{\alpha\}] \mid V(\alpha) = L\} \\
&\mathrm{G}[\![\mathit{wre}(\alpha, \beta)]\!] = \{V \in F[\{\alpha, \beta\}] \mid V(\alpha) = V(\beta)\} \\
&\mathrm{G}[\![\mathit{ntran}(\alpha, \beta, \gamma)]\!] = \{V \in F[\{\alpha, \beta, \gamma\}] \mid V(\gamma) = H \Rightarrow V(\alpha) = V(\beta)\} \\
&\mathrm{G}[\![\mathit{ptran}(\alpha, \beta, \gamma)]\!] = \{V \in F[\{\alpha, \beta, \gamma\}] \mid V(\gamma) = L \Rightarrow V(\alpha) = V(\beta)\} \\
&\mathrm{G}[\![c_0 \bullet c_1]\!] = \mathrm{G}[\![c_0]\!] \bullet \mathrm{G}[\![c_1]\!] \\
&\mathrm{G}[\![c \setminus \alpha]\!] = \mathrm{G}[\![c]\!] \setminus \alpha.
\end{aligned}$$

It follows that

$$\mathrm{G}[\![c_0 \bullet c_1]\!] = \{V \in F[\Lambda_0 \cup \Lambda_1]) \mid V\lceil\Lambda_0 \in \mathrm{G}[\![c_0]\!] \;\; \& \;\; V\lceil\Lambda_1 \in \mathrm{G}[\![c_1]\!]\},$$

where $\Lambda_0 = \mathrm{sort}(c_0)$ and $\Lambda_1 = \mathrm{sort}(c_1)$. In other words, the behaviours of the composition of two circuits are precisely those which restrict to behaviours of the component circuits.

3.2 Proposition.

(i) Let c be a circuit term of sort Λ. Let A be an assertion such that

$$G[\![c]\!] = G[\![A]\!]_\Lambda.$$

Then

$$G[\![c \setminus \alpha]\!] = G[\![\exists v_\alpha.A]\!]_{\Lambda\setminus\{\alpha\}}.$$

(ii) Let c_0 be a circuit term of sort Λ_0, and c_1 be a circuit term of sort Λ_1. Let A_0 and A_1 be assertions such that

$$G[\![c_0]\!] = G[\![A_0]\!]_{\Lambda_0} \quad \textit{and} \quad G[\![c_1]\!] = G[\![A_1]\!]_{\Lambda_1}.$$

Then

$$G[\![c_0 \bullet c_1]\!] = G[\![A_0 \wedge A_1]\!]_{\Lambda_0 \cup \Lambda_1}.$$

The denotations of basic components are readily expressed as assertions, *e.g.*

$$\mathrm{G}[\![\mathit{ntran}(\alpha,\beta,\gamma)]\!] = \mathrm{G}[\![v_\gamma = H \rightarrow v_\alpha = v_\beta]\!]_{\{\alpha,\beta,\gamma\}}.$$

Consequently, in this simple model, the proposition implies we can replace circuit operations by logical ones, the course followed in [G, G1].

Inclusion on $P(F)$ induces a semantic entailment:

$$A_0 \models^G A_1 \text{ iff } \mathrm{G}[\![A_0]\!] \subseteq \mathrm{G}[\![A_1]\!],$$

where A_0, A_1 are assertions. As in the introduction, basic circuit specifications in this model have the form $\Diamond\, A$ and $\Box\, A$ where A is an assertion. At present the work using this model has concentrated on specifications of the form $\Box\, A$ with

$$c \models^G \Box\, A \text{ iff } \mathrm{G}[\![c]\!] \subseteq \mathrm{G}[\![A]\!],$$

where c is a circuit term and A is an assertion. In the approach of [G, G1], no separate syntax is given for circuit terms. Instead circuits are translated directly into assertions in the manner of proposition 3.2 so that showing a circuit meets a specification amounts to showing an entailment, and so an implication, holds between two assertions, and this is purely a matter of logic. Restricting attention to specifications of the form $\Box\, A$ has led to the paradox that a short-circuit *Pow* $(\alpha) \bullet$ *Gnd* (α) satisfies any specification because $\mathrm{G}[\![\mathit{Pow}\ (\alpha) \bullet \mathit{Gnd}\ (\alpha)]\!] = \emptyset$, but this is no longer the case, of course, when specifications of the form $\Diamond\, A$ are permitted too.

4. The static configurations model.

For some purposes the model in [G, G1] is inadequate, for example, it fails to deal with some resistance and capacitance effects used in hardware design, and with the fact that sometimes the value at a point is not purely high or low. The work in [W] attempts to find a model and logic for circuits without these inadequacies. Essentially it takes ideas of Bryant (eg. [B]), on which several hardware simulators are based, and uses them to provide a semantics and proof system for a language for circuits which includes resistances and capacitances.

In [W] the value at a point is assumed to lie in the set

$$\mathbf{V} = \{H, L, X, Z\}.$$

A point assumes the value Z if it is not connected to any source, value H if it is connected to power but not to ground, the value L if it is connected to ground but not to power, and the value X if it is connected to both the power and ground. It is useful to order $\mathbf{V}$:

$$\begin{tikzcd}
 & \bullet X & \\
H\bullet \arrow[ur, phantom, "\leq" sloped] & & \bullet L \arrow[ul, phantom, "\geq" sloped] \\
 & \bullet Z \arrow[ul, phantom, "\geq" sloped] \arrow[ur, phantom, "\leq" sloped] &
\end{tikzcd}$$

Thus X is the least upper bound of H and L with respect to the order $\leq$, and, in general, we shall use ΣW for the least upper bound of a set W of values and write the least upper bound of a pair as $w_0 + w_1$.

The static configuration model is described and motivated in detail in [W] and [W1]. As was pointed out there, if resistance and capacitance effects are ignored the definition of state (called static configuration) is simplified. Certainly one component of the state of a circuit should be a value function V which assigns a value in $\mathbf{V}$ to every point in the sort of the circuit because it is with these values that circuits do calculations. Because we want to account for the behaviour of circuits in environments where, for example, a high voltage is placed on the gate of a transistor in order to ensure that the values on its other two points are the same, we take the value function to give the value at points in an environment in which other components including sources may be present. (We did this earlier for Gordon's relational model.) We want our model to be compositional in the sense that the value functions associated with a circuit term are determined by those of its proper subterms. This cannot be achieved with value functions alone. We need to keep track of that contribution to the value function which comes from sources within the circuit and how points are connected by wires and transistors to give a satisfactory treatment of hiding. Then we can obtain a compositional model (see [W1] for a detailed argument for the necessity of this extra complexity).

4.1 Definition.

Let Λ be a subset of the points Π. A *static configuration* of sort Λ is a structure

$$\langle V, I, \sim \rangle,$$

where $V : \Lambda \to \mathbf{V}$, the *value* function, $I : \Lambda \to \mathbf{V}$, the *internal-value* function and $\sim$ is an equivalence relation on Λ, the *connectivity* relation, which satisfy

$$I(\alpha) \leq V(\alpha),$$
$$\alpha \sim \beta \Rightarrow V(\alpha) = V(\beta) \;\;\&\;\; I(\alpha) = I(\beta),$$

for all points α, $\beta \in \Lambda$.

Write Sta[Λ] for the static configurations of sort Λ, and Sta for static configurations of any sort $\Lambda \subseteq \Pi$.

Assertions in [W] for expressing the properties of circuits have the following syntax:

Variables: We assume a set of variables Var which range over points Π, and have typical members $x, y, z, \cdots$. (Note in this model variables range over points *not* values.)

Value terms: Terms, denoting values in $\{H, L, Z, X\}$ have the form

$$t ::= V(\pi) \mid I(\pi) \mid H \mid L \mid Z \mid X,$$

where π is a point term, *i.e.* an element of Π or a variable in Var.

Behaviour assertions: The set of W-assertions is generated by

$$\phi ::= t_0 = t_1 \mid t_0 \leq t_1 \mid \pi_0 = \pi_1 \mid \pi_0 \sim \pi_1 \mid \phi_0 \wedge \phi_1 \mid \phi_0 \vee \phi_1 \mid \neg\phi \mid \mathtt{tt} \mid \mathtt{ff} \mid \exists x.\phi \mid \forall x.\phi$$

where t_0, t_1 are value terms and π_0, π_1 are point terms. As with G-assertions, we shall regard $\phi \to \theta$ as abbreviating $\neg\phi \vee \theta$ and $\phi \leftrightarrow \theta$ as abbreviating $\phi \to \theta \wedge \theta \to \phi$.

The semantics of assertions: We give the semantics of closed assertions. But first we have to treat value terms which are denoted by partial functions Sta $\rightharpoonup \mathbf{V}$, as follows:

$$\begin{aligned}
&\mathrm{W}[\![H]\!]\sigma = H \text{ for all } \sigma \in \mathrm{Sta}\\
&\mathrm{W}[\![L]\!]\sigma = L \text{ for all } \sigma \in \mathrm{Sta}\\
&\mathrm{W}[\![V(\alpha)]\!]\sigma = \begin{cases} V(\alpha) & \text{if } \sigma \in \mathrm{Sta} \ \ \& \ \ \alpha \in \mathrm{sort}(\sigma),\\ \text{undefined} & \text{otherwise.}\end{cases}\\
&\mathrm{W}[\![I(\alpha)]\!]\sigma = \begin{cases} I(\alpha) & \text{if } \sigma \in \mathrm{Sta} \ \ \& \ \ \alpha \in \mathrm{sort}(\sigma),\\ \text{undefined} & \text{otherwise.}\end{cases}
\end{aligned}$$

Each W-assertion is denoted by the subset of static configurations at which it is true, defined by the following induction on length:

$$\begin{aligned}
&\mathrm{W}[\![t_0 = t_1]\!] = \{\sigma \in \mathrm{Sta} \mid \mathrm{W}[\![t_0]\!]\sigma\downarrow \ \& \ \mathrm{W}[\![t_1]\!]\sigma\downarrow \ \& \ \mathrm{W}[\![t_0]\!]\sigma = \mathrm{W}[\![t_1]\!]\sigma\}\\
&\mathrm{W}[\![t_0 \leq t_1]\!] = \{\sigma \in \mathrm{Sta} \mid \mathrm{W}[\![t_0]\!]\sigma\downarrow \ \& \ \mathrm{W}[\![t_1]\!]\sigma\downarrow \ \& \ \mathrm{W}[\![t_0]\!]\sigma \leq \mathrm{W}[\![t_1]\!]\sigma\}\\
&\mathrm{W}[\![\alpha_0 = \alpha_1]\!] = \{\sigma \in \mathrm{Sta} \mid \alpha_0 \in \mathrm{sort}(\sigma) \ \ \& \ \ \alpha_1 \in \mathrm{sort}(\sigma) \ \ \& \ \ \alpha_0 = \alpha_1\}\\
&\mathrm{W}[\![\alpha_0 \sim \alpha_1]\!] = \{\sigma \in \mathrm{Sta} \mid \alpha_0 \in \mathrm{sort}(\sigma) \ \ \& \ \ \alpha_1 \in \mathrm{sort}(\sigma) \ \ \& \ \ \alpha_0 \sim \alpha_1\}\\
&\mathrm{W}[\![\phi_0 \wedge \phi_1]\!] = \mathrm{W}[\![\phi_0]\!] \cap \mathrm{W}[\![\phi_1]\!]\\
&\mathrm{W}[\![\phi_0 \vee \phi_1]\!] = \mathrm{W}[\![\phi_0]\!] \cup \mathrm{W}[\![\phi_1]\!]\\
&\mathrm{W}[\![\neg\phi]\!] = (\mathrm{Sta} \setminus \mathrm{W}[\![\phi]\!])\\
&\mathrm{W}[\![\mathtt{tt}]\!] = \mathrm{Sta}\\
&\mathrm{W}[\![\mathtt{ff}]\!] = \emptyset\\
&\mathrm{W}[\![\exists x.\phi]\!] = \{\sigma \mid \exists\alpha \in \mathrm{sort}(\sigma).\ \sigma \in \mathrm{W}[\![\phi[\alpha/x]]\!]\}\\
&\mathrm{W}[\![\forall x.\phi]\!] = \{\sigma \mid \forall\alpha \in \mathrm{sort}(\sigma).\ \sigma \in \mathrm{W}[\![\phi[\alpha/x]]\!]\}
\end{aligned}$$

Write $\sigma \models \phi$ if $\sigma \in \mathrm{W}[\![\phi]\!]$.

In order to define the semantics of circuits we introduce composition and hiding operations on static configurations.

4.2 Notation. We have used $V \setminus \alpha$ to stand for the restriction of a function $V : \Lambda \to \mathbf{V}$ to domain $\Lambda \setminus \{\alpha\}$. In addition we write $\Lambda \setminus \alpha$ for $\Lambda \setminus \{\alpha\}$, and in the case where $\sim \subseteq \Lambda \times \Lambda$ we write $\sim \setminus\alpha$ for its restriction to the relation $\sim \cap [(\Lambda \setminus \alpha) \times (\Lambda \setminus \alpha)]$.

4.3 Definition. Let $\sigma_0 = \langle V_0, I_0, \sim_0\rangle$ be a static configuration of sort Λ_0 and $\sigma_1 = \langle V_1, I_1, \sim_1\rangle$ be a static configuration of sort Λ_1. Define their *composition* to be

$$\sigma_0 \bullet \sigma_1 = \begin{cases} \langle V, I, \sim\rangle & \text{if } V_0\lceil\Lambda_1 = V_1\lceil\Lambda_0\\ \text{undefined} & \text{otherwise}\end{cases}$$

where

$$\begin{aligned}
V &= V_0 \cup V_1,\\
\sim &= (\sim_0 \cup \sim_1)^* \quad \text{and}\\
I(\alpha) &= \Sigma\{I_0(\beta) \mid \beta \in \Lambda_0 \ \ \& \ \ \beta \sim \alpha\} + \Sigma\{I_1(\beta) \mid \beta \in \Lambda_1 \ \ \& \ \ \beta \sim \alpha\}
\end{aligned}$$

for any $\alpha \in \Lambda_0 \cup \Lambda_1$.

Thus it is only possible to compose static configurations and get a defined result when their values agree on points that they have in common.

4.4 Definition. Let $\sigma = \langle V, I, \sim\rangle$ be a static configuration of sort Λ and α a point. Define *hiding*

$$\sigma \setminus \alpha = \begin{cases} \langle V \setminus \alpha,\ I \setminus \alpha,\ \sim \setminus \alpha\rangle & \text{if } \alpha \notin \Lambda \text{ or} \\ & \quad V(\alpha) = I(\alpha) + \Sigma\{V(\beta) \mid \beta \in \Lambda \setminus \alpha \ \ \& \ \ \beta \sim \alpha\} \\ \text{undefined} & \text{otherwise.} \end{cases}$$

Thus it is only possible to hide, or insulate, a point from the environment and get a defined result when the value at the point will not be disturbed; this is the case when the value at the point is due to the combined effect of internal sources and values due to unhidden points.

We extend the operations to sets of static configurations. For $S, S_0, S_1 \in P(\text{Sta})$ define

$$S_0 \bullet S_1 = \{\sigma_0 \bullet \sigma_1 \mid \sigma_0 \in S_0 \ \ \& \ \ \sigma_1 \in S_1 \ \ \& \ \ \sigma_0 \bullet \sigma_1 \downarrow\}$$
$$S \setminus \alpha = \{\sigma \setminus \alpha \mid \sigma \in S \ \ \& \ \ \sigma \setminus \alpha \downarrow\}.$$

We define the denotation $[\![c]\!]$ of a circuit term c to be a subset of $P(\text{Sta})$ by structural induction:

$$\mathrm{W}[\![Pow\ (\alpha)]\!] = \{\sigma \in \text{Sta}[\alpha] \mid I(\alpha) = H\}$$

$$\mathrm{W}[\![Gnd\ (\alpha)]\!] = \{\sigma \in \text{Sta}[\alpha] \mid I(\alpha) = L\}$$

$$\mathrm{W}[\![wre(\alpha,\beta)]\!] = \{\sigma \in \text{Sta}[\alpha,\beta] \mid I(\alpha) = Z \wedge I(\beta) = Z \wedge \alpha \sim \beta\}$$

$$\begin{aligned}\mathrm{W}[\![ntran(\alpha,\beta,\gamma)]\!] = \{\sigma \in \text{Sta}[\alpha,\beta,\gamma] \mid\ & I(\alpha) = Z \wedge I(\beta) = Z \wedge I(\gamma) = Z \wedge \\ & \neg(\gamma \sim \alpha) \wedge \neg(\gamma \sim \beta) \wedge \\ & (V(\gamma) = H \rightarrow \alpha \sim \beta) \wedge (V(\gamma) = L \rightarrow \neg(\alpha \sim \beta))\}\end{aligned}$$

$$\begin{aligned}\mathrm{W}[\![ptran(\alpha,\beta,\gamma)]\!] = \{\sigma \in \text{Sta}[\alpha,\beta,\gamma] \mid\ & I(\alpha) = Z \wedge I(\beta) = Z \wedge I(\gamma) = Z \wedge \\ & \neg(\gamma \sim \alpha) \wedge \neg(\gamma \sim \beta) \wedge \\ & (V(\gamma) = L \rightarrow \alpha \sim \beta) \wedge (V(\gamma) = H \rightarrow \neg(\alpha \sim \beta))\}\end{aligned}$$

$$\mathrm{W}[\![c \bullet d]\!] = \mathrm{W}[\![c]\!] \bullet \mathrm{W}[\![d]\!]$$

$$\mathrm{W}[\![c \setminus \alpha]\!] = \mathrm{W}[\![c]\!] \setminus \alpha.$$

In the definition above we have used a set-expression $\{\sigma \in \text{Sta}[\Lambda] \mid \phi\}$, where ϕ is W-assertion, to mean the subset of static configurations of sort Λ which satisfy the assertion ϕ.

Inclusion in the model induces notions of entailment, writing

$$\phi \models^W \theta \text{ iff } \mathrm{W}[\![\phi]\!] \subseteq \mathrm{W}[\![\theta]\!],$$

where θ, ϕ are W-assertions, and as we have seen

$$c \models^W \Diamond\, \theta \text{ iff } \mathrm{W}[\![c]\!] \cap \mathrm{W}[\![\theta]\!] \neq \emptyset,$$
$$c \models^W \Box\, \theta \text{ iff } \mathrm{W}[\![c]\!] \subseteq \mathrm{W}[\![\theta]\!],$$

where c is a circuit term and θ is a W-assertion. In [W], an extension of assertions above is used to provide a sound and complete proof system to verify when a circuit term satisfies any specification of the form $\Box\, \theta$.

5. The adjunction between the models in [G, G1] and [W].

Now we have two models, those in [G, G1] and [W], it is important to understand how they are related. This is not simply a matter forgetting about resistance and capacitance terms in the language for circuits because the notions of state used in two models differ; states, as formalised in [W] do include a value function but it is not assumed that the value at a point is either high (H) or low (L), and, in addition, states in [W] have other components expressing, for instance, the connectivity between points and the effects of internal sources. Still, there is an abstraction function from the states/behaviours of [W] to the states/behaviours of [G, G1] and via this we obtain a relation between the models and proof systems of [G, G1] and [W].

As in the introduction, the two notions of inclusion are related by an adjunction between the partial orders $(P(\mathrm{Sta}), \subseteq)$ and $(P(F), \subseteq)$. The adjunction is determined by an abstraction function between the states of the two models. There is clearly a partial function

$$abs : \mathrm{Sta} \rightharpoonup F$$

which acts so

$$abs(\langle V, I, \sim \rangle) = \begin{cases} V & \text{if } V \in F, \\ \text{undefined} & \text{otherwise,} \end{cases}$$

for a static configuration σ; thus $abs(\sigma)$ is undefined when the value function V_σ of the static configuration σ attributes value Z or X to some point. As explained in the introduction, this induces a left adjoint

$$abs_* : P(\mathrm{Sta}) \to P(F),$$

given by

$$abs_*(S) = \{V_\sigma \in F \mid \sigma \in S\}$$

for $S \in P(\mathrm{Sta})$, which has a right adjoint

$$abs^* : P(F) \to P(\mathrm{Sta})$$

given by

$$abs^*(R) = \{\sigma \in \mathrm{Sta} \mid V_\sigma \in F \Rightarrow V_\sigma \in R\},$$

for $R \in P(F)$. The adjunction is expressed by the property

$$S \subseteq abs^*(R) \Leftrightarrow abs_*(S) \subseteq R,$$

for $S \in P(\mathrm{Sta})$, $R \in P(F)$.

As in the introduction, we would like

$$\mathrm{W}[\![c]\!] \subseteq \mathrm{W}[\![{}^*A]\!] \text{ iff } \mathrm{G}[\![c]\!] \subseteq \mathrm{G}[\![A]\!], \text{ or equivalently, } c \models^W \Box\, {}^*A \text{ iff } c \models^G \Box\, A,$$

where A is a G-assertion and ${}^*A$ is some translation of it into a W-assertion. Comparing this with the property expressing the adjunction, which gives

$$\mathrm{W}[\![c]\!] \subseteq abs^*(\mathrm{G}[\![A]\!]) \Leftrightarrow abs_*(\mathrm{W}[\![c]\!]) \subseteq \mathrm{G}[\![A]\!],$$

we obtain this if

$$\mathrm{W}[\![{}^*A]\!] = abs^*(\mathrm{G}[\![A]\!]) \text{ and } \mathrm{G}[\![c]\!] = abs_*(\mathrm{W}[\![c]\!]),$$

the latter being an abstract expression for the required condition on c. More generally, we can ask for conditions on c such that

$$c \models^W \Box\, {}^*A \Rightarrow c \models^G \Box\, A, \text{ or}$$
$$c \models^W \Box\, {}^*A \Leftarrow c \models^G \Box\, A.$$

The translation $A \mapsto {}^*A$ from G to W assertions, so $\mathrm{W}[\![{}^*A]\!] = abs^*(\mathrm{G}[\![A]\!])$, is easily defined by structural induction. First define a W-assertion

$$D \equiv \forall x.\ V(x) = H \vee V(x) = L.$$

5.1 Proposition. *Let σ be a static configuration. Then*

$$\begin{aligned} \sigma \models D \text{ iff } & V_\sigma : sort(\sigma) \to \{H, L\} \\ \text{iff } & V_\sigma \in F \\ \text{iff } & abs(\sigma)\downarrow . \end{aligned}$$

Now, for a G-(value)term t define define its translation into a W-term ${}^*t$ by:

$${}^*v_\alpha \equiv V(\alpha), \quad {}^*H \equiv H, \quad {}^*L \equiv L.$$

For a G-assertion A define a W-assertion ${}^*A$ by structural induction:

$$\begin{aligned} &{}^*(t_0 = t_1) \equiv (D \to {}^*t_0 = {}^*t_1), \\ &{}^*\mathrm{tt} \equiv \mathrm{tt}, \quad {}^*\mathrm{ff} \equiv \neg D, \\ &{}^*(A_0 \wedge A_1) \equiv {}^*A_0 \wedge {}^*A_1, \\ &{}^*(A_0 \vee A_1) \equiv {}^*A_0 \vee {}^*A_1, \\ &{}^*(\neg A) \equiv (D \to \neg {}^*A), \\ &{}^*(\exists v_\alpha.\ A) \equiv {}^*(A[H/v_\alpha] \vee A[L/v_\alpha]), \\ &{}^*(\forall v_\alpha.\ A) \equiv {}^*(A[H/v_\alpha] \wedge A[L/v_\alpha]). \end{aligned}$$

By structural induction on A, using proposition 1.2, we can prove:

5.2 Lemma. *For A a G-assertion, $abs^*(\mathrm{G}[\![A]\!]) = \mathrm{W}[\![{}^*A]\!]$.*

Now we consider the following conditions on circuits c:

$$\begin{aligned} &\text{(a)} \quad \mathrm{G}[\![c]\!] \subseteq abs_*(\mathrm{W}[\![c]\!]), \\ &\text{(b)} \quad \mathrm{G}[\![c]\!] \supseteq abs_*(\mathrm{W}[\![c]\!]), \\ &\text{(c)} \quad \mathrm{G}[\![c]\!] = abs_*(\mathrm{W}[\![c]\!]). \end{aligned}$$

If (a) holds we deduce

$$\begin{aligned} c \models^W \Box\, {}^*A \;\Rightarrow\; & \mathrm{W}[\![c]\!] \subseteq \mathrm{W}[\![{}^*A]\!] \quad \text{by definition} \\ \Rightarrow\; & \mathrm{W}[\![c]\!] \subseteq abs^*(\mathrm{G}[\![A]\!]) \quad \text{by the above lemma} \\ \Rightarrow\; & abs_*(\mathrm{W}[\![c]\!]) \subseteq \mathrm{G}[\![A]\!] \quad \text{by the adjunction} \\ \Rightarrow\; & \mathrm{G}[\![c]\!] \subseteq \mathrm{G}[\![A]\!] \quad \text{by assumption (a)} \\ \Rightarrow\; & c \models^G \Box\, A \quad \text{by definition.} \end{aligned}$$

Thus (a) implies $c \models^W \Box\, {}^*A \Rightarrow c \models^G \Box\, A$. In fact, because a circuit term c has finite sort, it is not hard to show that (a) holds iff $c \models^W \Box\, {}^*A \Rightarrow c \models^G \Box\, A$, for all G-assertions A. Similarly, for a circuit term c, condition (b) holds iff $c \models^G \Box\, A$ implies $c \models^W \Box\, {}^*A$, for all assertions A, while condition (c) holds iff $c \models^G \Box\, A$ iff $c \models^W \Box\, {}^*A$, for all assertions A.

From these results on $\Box$ -assertions we can deduce how $\Diamond$-assertions are preserved and reflected under the assumptions (a), (b) or (c). Suppose, for instance, that (a) holds of a circuit c. Then

$$\begin{aligned} c \models^G \Diamond A &\Rightarrow c \not\models^G \Box \neg A \\ &\Rightarrow c \not\models^W \Box\, {}^*(\neg A) \\ &\Rightarrow c \not\models^W \Box\, (D \to \neg^* A) \\ &\Rightarrow c \not\models^W \Box \neg(D \wedge {}^*A) \\ &\Rightarrow c \models^W \Diamond\, (D \wedge {}^*A) \end{aligned}$$

Thus $c \models^G \Diamond A \Rightarrow c \models^W \Diamond D \wedge {}^*A$. Similarly, for a circuit term c, if condition (b) holds then $c \models^W \Diamond D \wedge {}^*A$ implies $c \models^G \Diamond A$, for all assertions A, while if condition (c) holds then $c \models^G \Diamond A$ iff $c \models^W \Diamond D \wedge {}^*A$, for all assertions A.

The strongest condition (c) would follow for any circuit term c if abs_* were a homomorphism preserving the behaviour of the basic components and the operations $\bullet$ and $\backslash\alpha$ for any $\alpha \in \Pi$. The behaviour of the basic components is preserved by abs_*:

5.3 Lemma.

(i) $abs_*(W[\![Pow\ (\alpha)]\!]) = G[\![Pow\ (\alpha)]\!]$,
(ii) $abs_*(W[\![Gnd\ (\alpha)]\!]) = G[\![Gnd\ (\alpha)]\!]$,
(iii) $abs_*(W[\![wre(\alpha,\beta)]\!]) = G[\![wre(\alpha,\beta)]\!]$,
(iv) $abs_*(W[\![ntran(\alpha,\beta,\gamma)]\!]) = G[\![ntran(\alpha,\beta,\gamma)]\!]$,
(v) $abs_*(W[\![ptran(\alpha,\beta,\gamma)]\!]) = G[\![ptran(\alpha,\beta,\gamma)]\!]$.

The function abs_* does preserve $\bullet$:

5.4 Lemma. *Let $S_0 \in P(Sta[\Lambda_0])$ and $S_1 \in P(Sta[\Lambda_1])$. Then*

$$abs_*(S_0 \bullet S_1) = abs_*(S_0) \bullet abs_*(S_1).$$

But, abs_* does not preserve $\backslash\alpha$; both inclusion (a) and (b) can fail to hold, essentially, for the reasons illustrated in the example below.

5.5 Example.

(i) To show the inclusion (a) can fail, let $S = \{\sigma\}$, a static configuration of sort Λ, where $\sigma \setminus \alpha \uparrow$ and $V_\sigma(x) \in \{H, L\}$ for all $x \in \Lambda$. Then $(abs_* S) \setminus \alpha = \{V_\sigma \setminus \alpha\}$, while $abs_*(S \setminus \alpha) = \emptyset$, so $(abs_* S) \setminus \alpha \not\subseteq abs_*(S \setminus \alpha)$.

(ii) To show the inclusion (b) can fail, let $S = \{\sigma\}$, a static configuration of sort Λ, where $V_\sigma(\alpha) = X(\text{or } Z)$ with $V_\sigma(x) \in \{H, L\}$, for all $x \in \Lambda \setminus \alpha$ and $\sigma \setminus \alpha \downarrow$. Then $abs_*(S \setminus \alpha) = \{V_\sigma \setminus \alpha\}$, while $(abs_* S) \setminus \alpha = \emptyset \setminus \alpha = \emptyset$, so $abs_*(S \setminus \alpha) \not\subseteq (abs_* S) \setminus \alpha$.

By banning such examples we obtain sufficient conditions to ensure (a), (b), and their conjunction (c).

To ensure (a) we have:

5.6 Lemma. *Let $S \in P(Sta)$. If*

$$\forall \sigma \in S.\ (abs(\sigma) \downarrow \Rightarrow \sigma \setminus \alpha \downarrow) \qquad (a1)$$

then $(abs_*(S)) \setminus \alpha \subseteq abs_*(S \setminus \alpha)$.

To ensure (b) we shall use:

5.7 Lemma. *Let $S \in P(Sta)$. If*

$$\forall \sigma \in S.\ (\sigma \setminus \alpha \downarrow \ \&\ \ abs(\sigma \setminus \alpha) \downarrow \Rightarrow abs(\sigma) \downarrow) \qquad (b1)$$

then

$$abs_*(S \setminus \alpha) \subseteq (abs_* S) \setminus \alpha.$$

The conditions (a1) and (b1) used in lemmas 5.6 and 5.7 are expressible as W-assertions. We have already seen, writing

$$D \equiv \forall x.\ V(x) = H \vee V(x) = L,$$

that $abs(\sigma) \downarrow$ iff $\sigma \models D$, for a static configuration σ. Similarly writing

$$D_\alpha \equiv \forall x.\ \neg(x = \alpha) \rightarrow (V(x) = H \vee V(x) = L),$$

we get $abs(\sigma \setminus \alpha) \downarrow$ iff $\sigma \models D_\alpha$ and $\sigma \setminus \alpha \downarrow$, for $\sigma \in \mathrm{Sta}$. Expressing the definedness of hiding, as in [W], by

$$\begin{aligned} G_\alpha \equiv & [H \leq V(\alpha) \rightarrow (H \leq I(\alpha) \vee \exists x.\neg(x = \alpha) \wedge H \leq V(x)\ \ \&\ \ x \sim \alpha)] \\ & \wedge [L \leq V(\alpha) \rightarrow (L \leq I(\alpha) \vee \exists x.\neg(x = \alpha) \wedge L \leq V(x)\ \ \&\ \ x \sim \alpha)] \end{aligned}$$

we have $\sigma \setminus \alpha \downarrow$ iff $\sigma \models G_\alpha$, for any $\sigma \in \mathrm{Sta}$.

Suppose for each subterm of c of the form $c' \setminus \alpha$ we have that c' satisfies $\Box\, D \rightarrow G_\alpha$. Then using the facts in 5.3, 5.4, 5.6 and the monotonicity of $\bullet$ and $\setminus\alpha$, by structural induction on c, we can show that c meets the condition (a1) and so (a) above. From which we obtain the following:

5.8 Corollary. *Let c be a circuit term such that for all subterms of the form $c' \setminus \alpha$*

$$c' \models^W \Box\, D \rightarrow G_\alpha. \qquad (a2)$$

Then $c \models^W \Box\ {}^*A$ implies $c \models^G \Box\ A$, and $c \models^G \Diamond\ A$ implies $c \models^W \Diamond\ D \wedge {}^*A$, for any G-assertion A.

A similar argument, this time with respect to conditions (b) and (b1), shows:

5.9 Corollary. *Let c be a circuit term such that for all subterms of the form $c' \setminus \alpha$*

$$c' \models^W \Box\, G_\alpha \wedge D_\alpha \rightarrow D. \qquad (b2)$$

Then $c \models^G \Box\ A$ implies $c \models^W \Box\ {}^*A$, and $c \models^W \Diamond\ D \wedge {}^*A$ implies $c \models^G \Diamond\ A$, for any G-assertion A.

By combining the conditions in 5.8 and 5.9 we obtain:

5.10 Corollary. *Let c be a circuit term. If for all subterms $c' \setminus \alpha$*

$$c' \models^W \Box\, D_\alpha \rightarrow (G_\alpha \leftrightarrow D) \qquad (c2)$$

then $c \models^G \Box\, A$ iff $c \models^W \Box\ {}^*A$, and $c \models^G \Diamond\ A$ iff $c \models^W \Diamond\ D \wedge {}^*A$, for any G-assertion A.

Remark. In the case where $\alpha \in \mathrm{sort}(c')$ for all subterms $c' \setminus \alpha$ of c the condition (b2) in 5.9 can clearly be replaced by

$$c' \models^W \Box\, D_\alpha \rightarrow (G_\alpha \rightarrow (V(\alpha) = H \vee V(\alpha) = L)).$$

The proviso that α is in the sort of σ is necessary because otherwise $V(\alpha) = H$ and $V(\alpha) = L$ are taken to be false. Similarly, when $\alpha \in \mathrm{sort}(c')$ for all subterms $c' \setminus \alpha$ of c the condition (c2) in 5.10 can be replaced by

$$c' \models^W \Box\, D_\alpha \rightarrow (G_\alpha \leftrightarrow (V(\alpha) = H \vee V(\alpha) = L)).$$

Corollary 5.10 provides a sufficient condition, *viz.* (c2), on terms c to ensure agreement between the models [G, G1] and [W]. If for all subterms $c' \setminus \alpha$ of a circuit c, c' meets the condition of (c2), the semantic treatment of circuits in [G, G1] will only lead to correct specifications being shown to hold of c, at least as far as the model in [W] is concerned. In this sense, if a circuit term satisfies the conditions of 5.10 the treatment of hiding given in [G, G1] is safe.

The condition of 5.10 can be expressed in a more intuitive fashion. For a static configuration σ, provided $\alpha \in \text{sort}(\sigma)$, we have the equivalence

$$\begin{aligned}\sigma \models D_\alpha \rightarrow (G_\alpha \leftrightarrow D) \text{ iff } \sigma \models &(\forall x.\ \neg x = \alpha \rightarrow (V(x) = H \vee V(x) = L)) \rightarrow \\ &((\exists x.\ \neg x = \alpha \wedge x \sim \alpha) \vee I(\alpha) = H \vee I(\alpha) = L \vee \\ &(I(\alpha) = Z \wedge V(\alpha) = X)).\end{aligned}$$

Hence:

5.11 Corollary. *Let c be a circuit term. If for all subterms $c' \setminus \alpha$ we have $\alpha \in sort(c')$ and*

$$\begin{aligned}c' \models^W \Box\, (\forall x.\ \neg x = \alpha \rightarrow (V(x) = H \vee V(x) = L)) \rightarrow \\ ((\exists x.\ \neg(x = \alpha) \wedge x \sim \alpha) \vee I(\alpha) = H \vee I(\alpha) = L \vee \\ (I(\alpha) = Z \wedge V(\alpha) = X))\end{aligned} \qquad (c3)$$

then $c \models^G \Box\, A$ iff $c \models^W \Box\, {}^*A$, and $c \models^G \Diamond\, A$ iff $c \models^W \Diamond\, D \wedge {}^*A$, for any G-assertion A.

Strengthening (c3) we obtain:

5.12 Corollary. *Let c be a circuit term. If for all subterms $c' \setminus \alpha$ we have $\alpha \in sort(c')$ and*

$$\begin{aligned}c' \models^W \Box\, (\forall x.\ \neg x = \alpha \rightarrow (V(x) = H \vee V(x) = L)) \rightarrow \\ ((\exists x.\ \neg(x = \alpha) \wedge x \sim \alpha) \vee I(\alpha) = H \vee I(\alpha) = L)\end{aligned} \qquad (c4)$$

then $c \models^G \Box\, A$ iff $c \models^W \Box\, {}^*A$, and $c \models^G \Diamond\, A$ iff $c \models^W \Diamond\, D \wedge {}^*A$, for any G-assertion A.

Superficially, it would seem corollary 5.12 is less widely applicable than 5.11. In fact, on the contrary, it can be shown for circuit terms, built-up according the rules of section 2, that (c3) holds iff (c4) does and thus no generality is lost in using condition (c4) in 5.12.

As an example, consider the circuit term

$$c \equiv (ntran(\alpha, \beta, \gamma) \bullet ptran(\alpha, \beta, \gamma)) \setminus \gamma.$$

Taking $c' \equiv ntran(\alpha, \beta, \gamma) \bullet ptran(\alpha, \beta, \gamma)$ it is not the case that c' satisfies $\Box\, G_\gamma \wedge D_\alpha \rightarrow D$, the condition (b2) of 5.9, and for this reason it is no surprise that we have $c \models^G \Box\, (v_\alpha = v_\beta)$ and yet $c \not\models^W \Box\, (V(\alpha) = V(\beta))$.

The conditions on circuits we have introduced are only sufficient and not necessary to guarantee agreement between the two models. For example, for any G-assertion A,

$$\begin{aligned}c \models^W \Box\, {}^*A \text{ iff } c \models^G \Box\, A, \text{ and} \\ c \models^W \Diamond\, D \wedge^* A \text{ iff } c \models^G \Diamond\, A\end{aligned}$$

when c is the circuit term $ntran(\beta, \gamma, \alpha) \setminus \alpha$ even though $ntran(\beta, \gamma, \alpha)$ fails to satisfy (c4) of 5.12.

The condition (c4) of 5.12 is intuitively appealing and seems to be met in practice when hiding points in a good many circuits. There may be some syntactic constraint on circuit terms which is not too restrictive and yet ensures this condition is met when hiding. If so, for such circuit terms, the simple relational model of [G, G1] would suffice for verification. Of course, such verification could not apply when charge sharing or ratioing of resistances were used in the design—for them more detailed models like [B] and [W] would be needed. And then there is the problem of transistor thresholds. However for CMOS it seems a more conservative model of transistors suffices to take account of thresholds—see section 6. Indeed there is the likelihood that for a wide range of CMOS designs we can *prove* that a simple relational model is adequate as the basis for verification.

6. Further problems to explore.

Both models we have presented idealise the behaviour of transistors to forms of switches. In reality transistors do not always behave as switches. Because of switching thresholds an n-type transistor may not connect when its gate is high and both its source and drain are high. One solution proposed by Mike Fourman, and followed up by Mike Gordon, is to use a more conservative model of transistors which, in terms of the model [W], would allow $\neg\alpha \sim \beta$ when all points α, β, γ of an n-type transistor $ntran(\alpha, \beta, \gamma)$ were high [F1, G2]. (For a model like that in [G] this can only make a difference if the assumption that values are either H or L is dropped.) While inadequate for NMOS this does not appear too restrictive for CMOS designs. It looks as if the results here go over with a few small changes when relating two models in this more conservative regime. This should be checked. To cope with NMOS it seems an extension of the models in [B] and [W] is needed to measure drops or gains in voltage due to threshold effects. Unfortunately this complicates the models further.

To my knowledge no model at the level of abstraction seen here copes adequately with sequential circuits with static memory. While simulators like Bryant's generate one possible sequence of behaviour it is hard to find a compositional model which predicts all possible behaviours and no more—see [W] for a little more discussion. One can postulate components which have the desired behaviour but deriving their behaviour from basic components like transistors, capacitances, wires and resistances seems difficult without encumbering the model with all sorts of details about delays across wires and gates and transient capacitance effects. Can some more abstract model be found, in the spirit of those presented here, which copes with feedback loops of the sort necessary to explain flipflops for static memory? Perhaps extra structure in the form of relations expressing causal dependency will do it—such a trick is needed in the real-time programming language Esterel [BC]. On the positive side it seems the model in [W] generalises well when memory is purely dynamic (due to charge storage).

When are directional models as sometimes used by Mike Gordon and others justified? Maybe it's useful to mix assertions in the term language here in order to impose constraints on the environment. Circuit terms would include *e.g.* resistances.

Abstraction based on I instead of V. What happens?

How generally applicable are ideas like the above *e.g.* to temporal abstraction in [M]. (Literature on abstract interpretation may be useful.)

The techniques of this note can be used to highlight the problem with the crude model relative to a more detailed one. How useful are the results above in the practice of verifying circuits? The model [G, G1] should be used wherever possible because it is so much simpler than that in [W]. But how easy is it in practice to show a circuit meets conditions sufficient to ensure the simpler model may be used? After all such conditions are generally expressed in terms of the more complicated model. Is there some reasonable syntactic constraint on circuit terms which ensures a condition like (c4) in 5.12 is met when hiding?

Work needs to be done on formalising the relationship between qualitative models like those presented here, which, for instance, presuppose a clear understanding of a voltage being high or low, and quantitative physical models. The relationship can be very subtle but must be understood if we are ever to prove a design correct assuming only the correctness of its layout.

Hiding can be understood as existential quantification in both models (in the W-model it is a left adjoint to $(-\setminus\alpha)^*$) and in the G-model composition is conjunction though not in the W-model. The indexed-category view of models has been stressed by Mike Fourman based on his experience with categorical logic. Can it lead to a more uniform presentation of the models and proof systems for circuits, and if so how is composition to be understood?

References.

[B] Bryant, R.E., A switch-level model and simulator for MOS digital systems. IEEE Transactions on Computers C-33 (2) pp. 160-177, February 1984.

[BC] Berry, G., and Cosserat, L., The Esterel synchronous programming language and its mathematical semantics. In the proceedings of the joint US-UK seminar on the semantics of concurrency, July 1984, Carnegie-Mellon University, Springer-Verlag Lecture Notes in Comp. Sc. 197, 1984.

[C] Cardelli, L., An algebraic approach to hardware description and verification. Ph.D. thesis, Comp.Sc.Dept., University of Edinburgh, 1982.

[CC] Cousot, P., and Cousot, R., Abstract interpretation: a unified lattice model for static analysis of programs by constructions or approximations of fixpoints. POPL 1977.

[F] Fourman, M.P., Verification using higher-order specifications and transformations. Department of Electrical Engineering, Brunel University, 1986.

[F1] Fourman, M.P., Verbal communication, Leeds Workshop 1986.

[G] Gordon, M.J.C., Why higher order logic is a good formalism for specifying and verifying hardware. Report no.77 of the Computer Laboratory, Cambridge University 1985.

[G1] Camilleri, A.J., Gordon, M.J.C., and Melham, T.F., Hardware verification using higher-order logic. Proceedings of IFIP International Working Conference "From H.D.L. Descriptions to Guaranteed Correct Circuit Designs", Grenoble, 1986.

[G2] Gordon, M.J.C., Switch models of CMOS. In preparation 1987.

[M] Melham, T.F., Abstraction mechanisms for hardware verification. In "VLSI Specification, Verification and Synthesis", G.M. Birtwistle and P.A. Subrahmanyam, eds, Kluwer Press 1987.

[R] Robinson, E., Powerdomains, modalities and the Vietoris monad. Report no.98 of the Computer Laboratory, University of Cambridge, 1986.

[W] Winskel, G., Lectures on models and logic of MOS circuits. Proceedings for the Marktoberdorf Summer School, July 1986, published by Springer, 1987.

[W1] Winskel, G., A compositional model of MOS circuits. To appear in the book: VLSI Specification, Verification and Synthesis, G.M. Birtwistle and P.A. Subrahmanyam(eds) Kluwer Press 1987.

[W2] Winskel, G., A note on powerdomains and modality. In the May issue of Theoretical Computer Science, 1985.