

Relativistic Causality and Clockless Circuits *

Philippe Matherat
CNRS-LTCI (UMR 5141) and Institut Télécom
<http://matherat.net/>

Marc-Thierry Jaekel
CNRS-LPTENS (UMR 8549), Ecole Normale Supérieure
and Université Pierre et Marie Curie

Abstract

Time plays a crucial role in the performance of computing systems. The accurate modelling of logical devices, and of their physical implementations, requires an appropriate representation of time and of all properties that depend on this notion. The need for a proper model, particularly acute in the design of clockless delay-insensitive (DI) circuits, leads one to reconsider the classical descriptions of time and of the resulting order and causal relations satisfied by logical operations. This questioning meets the criticisms of classical spacetime formulated by Einstein when founding relativity theory and is answered by relativistic conceptions of time and causality. Applying this approach to clockless circuits and considering the trace formalism, we rewrite Udding's rules which characterize communications between DI components. We exhibit their intrinsic relation with relativistic causality. For that purpose, we introduce relativistic generalizations of traces, called R-traces, which provide a pertinent description of communications and compositions of DI components.

1 Introduction

Regular and important advances in semiconductor technology allow more and more complex processors to be designed. Meanwhile, progress in computing capacity is

*© ACM, 2011. This is the author's version of the work. It is posted here by permission of ACM for your personal use. Not for redistribution. The definitive version was published in ACM Journal of Emerging Technologies in Computing Systems, Vol. 7, No. 4, Article 20, December 2011, <http://doi.acm.org/10.1145/2043643.2043650>.

accompanied by a need for good representations of physical devices, which are required to remain pertinent at smaller and smaller scales. This trend naturally increases the number of physical properties to be accounted for when implementing logical devices. Appropriate models become mandatory prerequisites for developing new techniques. This particularly is the case when adopting the approach of distributed systems or clockless circuits.

Obviously, time plays an important part in the operation of computing systems. The significant effects induced by time delays in modern electronic circuits point at limits that one rapidly reaches in a classical framework when looking for physically precise models. The only way to go beyond these limits is to adopt a new framework, able to match physical reality at a more fundamental level.

Practical difficulties met in disseminating and managing clock signals over complex designs have made the approach of clockless circuits particularly interesting and promising. But removing the clock does not mean that designs are freed from all the constraints associated with time. Time is by itself a very complex notion, which is hardly captured in totality, even in most advanced physical theories. Designing circuits around a single clock appears as a convenient way to implement known and classical properties related to time. In the absence of a clock, other solutions have to be found to ensure the proper time evolution of a circuit. To this end, one must reconsider the classical models of time that have been useful when designing existing electronic circuits.

Important progress in the control of time constraints in clockless circuits has been made with the notion of *delay insensitivity* [12, 22, 24, 2, 26]. By removing any dependence on time lapses due to signal propagation, one becomes able, with a simple set of rules, to ensure the correct functioning of clockless logical devices. With this efficient trick comes a new concept of the way distributed systems can synchronize their actions, by means of communication, in order to realize specified computations.

Remarkably, this approach follows, in the context of logical devices, a line which is very close to that followed by Einstein, in the general context of physics, when introducing the founding concepts of relativity theory [4, 5]. The necessity of constructing a new framework emerges from the remark that no physical system exists that can deliver time over all space simultaneously. Physical time, as it can be observed, is necessarily a spatially localized quantity. To promote time to a physical quantity that can be shared by remote observers, propagation signals must be used. Resulting propagation delays must be taken into account when synchronizing different “local times”, that profoundly change the relations between space and time assumed in classical theories.

A change of the notion of time has major consequences for the expression of causality [17, 18]. As causality clearly depends on a notion of order in time,

a drastic change in the possibility of time ordering, as raised by the relativistic framework, implies a revision of the dependence of all operations, including logical ones, on causal relations.

The effect of the change in the properties of time induced by relativity on distributed systems has already been discussed in the literature ([7, 8, 10, 9]). Many formalisms also exist that describe computing concurrency and have provided efficient models for designing and analyzing asynchronous circuits. On one hand, Petri networks [14, 13] and their interpretation called signal transition graphs (STG) [16, 1] have provided powerful representations of partial orders and causal relations in logic circuits. On the other hand, trace semantics, communicating sequential processes (CSP) [6], and related approaches [22, 24, 2, 11, 15], are preferred in modelling complex systems. A formalism sharing both qualities would be particularly helpful for efficiently simulating and implementing causal relations in asynchronous circuits of high complexity.

We shall show here that a closer account of physical causality, based on the relativistic notion of time ordering, provides a way to modify the formalism of traces so that causal relations can be represented in a natural way. This representation should make them easier to implement in arbitrarily complex designs. In order to fulfill that aim, a key step is to give communications between components a representation that naturally satisfies the constraints imposed by relativistic time orders. Remarkably, Udding's rules [22], which are used to define communications within DI systems, appear to suit this purpose. Rewriting these rules in terms of R-traces, we shall exhibit their intrinsic connection with relativistic causality.

In the first two sections we recall the profound changes brought by the relativistic framework to the notions of time, time-ordering and causal relations. In the following sections, we introduce generalizations of traces, called R-traces, and use them to rewrite Udding's rules. We also briefly describe how R-traces map onto the usual traces in a classical environment. In conclusion, we point at some properties and other potential applications of R-traces.

2 Relativity and time-ordering

In this section, we briefly recall the properties of time according to relativity theory [4, 5, 17, 18], in contrast to those which are implicitly assumed in a classical framework. We discuss the new status given by relativity theory to simultaneity in time, and the notion of order in time that follows. Consequently, the notion of causality used in a relativistic framework significantly differs from its classical analog.

Although time and space can be treated as distinct concepts in classical theory, this can no longer be the case in a relativistic framework, where these no-

tions cannot be considered as being independent. To be precise, within a classical framework, time is not affected by a change of frame (or observer). For instance, coordinates (t, x) in an inertial frame are related to coordinates (t', x') in another frame, with relative velocity v , according to the usual Galilean transformation (for simplicity, expressions are written here for a single spatial dimension):

$$\begin{aligned}x' &= x - vt \\t' &= t\end{aligned}\tag{1}$$

On the other hand, the same change of inertial frame within a relativistic framework introduces a dependence of time on spatial positions (light propagates at a finite velocity c):

$$\begin{aligned}x' &= \frac{x - vt}{\sqrt{1 - \frac{v^2}{c^2}}} \\t' &= \frac{t - \frac{v}{c^2}x}{\sqrt{1 - \frac{v^2}{c^2}}}\end{aligned}\tag{2}$$

This fundamental property is in fact a direct consequence of the definition of time in terms of physical observables and of the existence of a maximum propagation speed, c . No physical system exists that can provide time simultaneously within a whole extended area in space. Time can only be obtained from a clock locally, that is at a given place in space [4, 5]. Time can then only be defined over all space by comparing clocks located at different spatial positions. These comparisons are made by means of propagating signals. As a consequence of the delays due to propagation at finite speed (even at light velocity c) the notion of simultaneity is frame dependent. For instance, in contrast to its classical analog (1), the relativistic transformation of time (2) shows that two events A and B that occur simultaneously but at different places in a given frame, lose their simultaneity when they are seen in a relatively moving frame.

$$\begin{array}{ll} \text{classical (1):} & t_B = t_A, \quad x_B \neq x_A \quad \Rightarrow \quad t'_B = t'_A \\ \text{relativistic (2):} & t_B = t_A, \quad x_B \neq x_A \quad \Rightarrow \quad t'_B \neq t'_A \end{array}\tag{3}$$

The existence of an absolute global time, independent of space, is ruined by the impossibility of defining simultaneity over all space in a frame-independent way. This however excludes neither the possibility of defining simultaneity, but in a frame-dependent way, nor the existence of frame-independent properties of time. The last possibility appears to be easier to realize. Following that option, one remarks that expressions (2) describe the transformations of space and time under the action of the Lorentz group. These transformations generalize to spacetime

the action of the group of spatial rotations. It corresponds to transformations that preserve the spacetime interval I_{AB} between two events A and B .

$$I_{AB} = c^2(t_A - t_B)^2 - (x_A - x_B)^2 \quad (4)$$

Spacetime intervals (4) are invariant under frame transformations (2) and allow one to define spacetime relations and subsets that are independent of the observer. In particular, a null spacetime interval ($I_{AB} = 0$) characterizes a pair of events (A, B) that can be joined by a motion at the maximum speed c , i.e. which lie on the same light ray. The light rays that are incident on a same event A define a subset \mathcal{C}_A , the light cone from event A (see Equation (4)).

$$\mathcal{C}_A \equiv \{B : I_{AB} = 0\} \quad (5)$$

\mathcal{C}_A bounds the part of spacetime which can be reached by signals sent or received at A : it realizes a partition of spacetime into three subsets.

$$\begin{aligned} \mathcal{C}_{A>} &\equiv \{B : I_{AB} > 0\} \\ \mathcal{C}_A &\equiv \{B : I_{AB} = 0\} \\ \mathcal{C}_{A<} &\equiv \{B : I_{AB} < 0\} \end{aligned} \quad (6)$$

The interior of a light cone $\mathcal{C}_{A>}$ consists of all spacetime events B that are separated from event A by a time-like interval ($I_{AB} > 0$), while the exterior of the same light cone $\mathcal{C}_{A<}$ consists of all events B that are separated from event A by a space-like interval ($I_{AB} < 0$). The interior of light cone $\mathcal{C}_{A>}$ furthermore consists of two disjoint parts which describe the *future* of event A ($\mathcal{C}_{A>}^+$), and the *past* of event A ($\mathcal{C}_{A>}^-$).

$$\begin{aligned} \mathcal{C}_{A>}^+ &\equiv \{B : I_{AB} > 0, t_B - t_A > 0\} \\ \mathcal{C}_{A>}^- &\equiv \{B : I_{AB} > 0, t_B - t_A < 0\} \end{aligned} \quad (7)$$

All the latter properties are frame-independent. Let us note that as opposed to the interior $\mathcal{C}_{A>}$, the exterior of a light cone $\mathcal{C}_{A<}$ cannot be partitioned in a frame-independent way into two parts corresponding to a positive ($t_B - t_A > 0$) or negative ($t_B - t_A < 0$) time interval. Indeed, for events B in $\mathcal{C}_{A<}$, the sign of the time interval $t_B - t_A$ depends on the observer. These properties exhibit a major difference due to the relativistic framework: given a particular event A , there exists a subset of events B that have a definite position in time with respect to A ($\mathcal{C}_{A>}$), but also a large subset of events B that cannot have a definite position in time relative to event A ($\mathcal{C}_{A<}$).

Equations (7) show that, although two arbitrary events cannot be ordered in time, due to the frame dependence of the simultaneity property (3), nonetheless

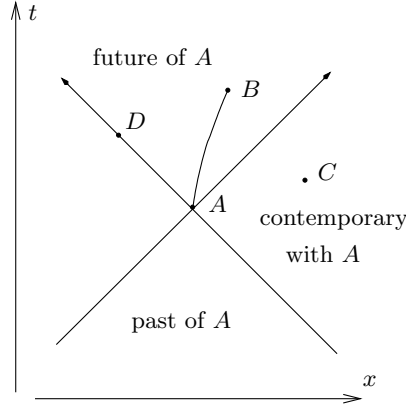


Figure 1: A spacetime diagram

all events belonging to $\mathcal{C}_{A>}^+$ are posterior to A while all events belonging to $\mathcal{C}_{A>}^-$ are anterior to A , both properties being frame-independent. One remarks that the existence of a time order between two events is a symmetric property:

$$B \in \mathcal{C}_{A>}^\pm \Leftrightarrow A \in \mathcal{C}_{B>}^\mp \quad (8)$$

$$C \in \mathcal{C}_{A<} \Leftrightarrow A \in \mathcal{C}_{C<} \quad (9)$$

Two events A and B satisfying conditions (8) are ordered in time and can thus be causally related. Moreover, this property is easily seen to be transitive.

$$B \in \mathcal{C}_{A>}^\pm \wedge C \in \mathcal{C}_{B>}^\pm \Rightarrow C \in \mathcal{C}_{A>}^\pm \quad (10)$$

In other words, although not allowing a total time ordering of events, the relativistic framework still allows one to define a partial time-ordering that does not depend on the observer. This property is necessary and sufficient for formulating causality in physics.

Time-order relations are advantageously illustrated on spacetime diagrams. In Figure 1, D represents an event with vanishing interval with respect to event A , i.e. located on the light cone \mathcal{C}_A from A . Note that, up to now, we have used the term ‘event’ in accordance with its physical meaning, i.e. to denote a point in spacetime. In contrast, in the context of logic devices, the term ‘event’ usually denotes a transition or a pulse, propagating through a wire or an electromagnetic channel [21]. According to previous discussions, the latter rather corresponds to a ray, connecting points located on a same light cone (Figure 1). But, it is easily seen that, when sent or received, such a logical ‘event’ also corresponds to a transition that is localized in time, and which can thus be identified with a spacetime ‘event’ [17, 18, 19, 21]. These two characteristic properties, namely localization in time and spreading over space, are precisely the ones that are necessary for

synchronizing actions performed at different locations in space. They also allow the local notion of ‘logical event’, defined as a logical action occurring in a device in coincidence with a spacetime ‘event’, to be extended to communications, i.e.: logical actions occurring in different devices in a correlated way. In the latter case, ‘logical events’ are associated with propagations rather than points in spacetime. As previously discussed, they must respect the constraints fixed by the existence of a bound on propagation speeds, hence the light cone partitions of spacetime. They may alternatively be seen as a minimal description, regarding logical properties, of events occurring at the boundary between two devices with different spatial locations. For simplicity and when no confusion may arise, we shall in the following use the same term ‘event’ for denoting both concepts.

Figure 1 also shows an event B that lies in the future light cone of A ($B \in \mathcal{C}_{A>}^+$). This property is equivalent to the possibility for a physical system to travel from event A to event B with a speed that is lower than light velocity c , the largest allowed velocity. Event B can then be said to occur *after* event A . In contrast, as no definite time order can be attributed to an event C lying outside of the light cone from A ($C \in \mathcal{C}_{A<}$, $I_{AC} < 0$ is space-like), C is said to be ‘contemporary with A ’.

These properties can be used to express relativistic causality. Two actions occurring at different locations in spacetime can be causally related only if the two events A and B at which they occur are ordered in time, i.e. satisfy relation (8). In that case, these two actions may produce a result that depends on the time-order of events A and B . Conversely, two actions occurring at two events A and C that are contemporary, i.e. A and C satisfy (9), produce a result that cannot depend on their order, as the latter cannot be identified with an order in time. One deduces that the result of two actions occurring at two different events A and B can depend on the order of these actions if and only if A and B can be ordered in time (8). Otherwise, two actions occurring at events satisfying Equations (9) must be considered to be independent of each other and hence cannot satisfy a causal relation.

3 Causality and time delays

Computation reduces, in present realizations, to chaining elementary operations performed on Boolean variables, linking these operations through a finite set of prescribed logical relations. This occurs in logic devices performing successive operations ruled by a clock or in asynchronous (clockless) circuits as well. In all cases, specifications are satisfied by linking the prescribed logical rules to causal relations constraining the physical systems in which they are realized. Not surprisingly, logic devices can be designed in a satisfactory way only when an appropriate

representation of causal relations is available.

Modern logic devices are implemented in highly integrated components, pushing the limits of transport properties. Due to clocks operating at high frequencies and despite small dimensions, the limits affecting the propagation of signals within a chip, due to bounded propagation speeds, cannot be ignored. A major consequence, as discussed in previous section, is a breakdown of the classical (Newtonian or Galilean) representation of time-ordering, pointing at the necessity of a genuinely relativistic representation. Obviously, the same remarks apply to distributed systems—to systems made of components with large separations (measured as propagation times) when compared with their internal period [7, 8, 10]. Distributed systems or asynchronous circuits can be seen to involve two different kinds of logical processes:

- transitions associated with changes of state of a local component,
- communications associated with exchanges of data between components.

Both kinds of processes must be considered when defining the specifications of distributed systems or asynchronous circuits and when setting their communication protocols. Moreover, these logical processes should be associated with implementations corresponding to two different kinds of events, this notion being understood according to the two different conceptions discussed in the previous section.

The implementation of logical processes relies on the existence of causal relations, hence on the possibility of ordering in time, the corresponding events. More precisely, the time order of two events becomes an objective property as soon as each event belongs to the interior of the other's light cone (8), or else as soon as they define a time-like interval (6). Two different kinds of time-like intervals can be seen to be involved in the definition of distributed systems or asynchronous circuits:

- intervals between two events characterizing the successive states of one local component;
- intervals between two events characterizing the emission and reception of communication signals.

The fact that the second case also corresponds to a time-like interval rather than to a light-like interval is due to the fact that, depending on the technology and implementation choices, signals may propagate at a maximum speed smaller than or equal to, the light velocity. This entails no fundamental consequences, as all properties due to the existence of a maximum propagation speed still hold.

Figure 1 shows two events A and B , with B occurring after A ($B \in \mathcal{C}_{A>}^+$). Although different observers may attribute different dates to these two events,

corresponding to different time delays undergone by a signal propagating from A to B , all observers will nonetheless agree on the time-like character of the interval I_{AB} ($I_{AB} > 0$) and on the time-order of the two events ($B \in \mathcal{C}_{A>}^+$, $A \in \mathcal{C}_{B>}^-$). The corresponding physical time delay coincides with the *delay* accounted for when designing electronic circuits. As opposed to the time delay itself, the time-like character of the interval between two events A and B and their time order (here B after A), are objective properties. As illustrated by this simple example, the time-ordering properties of logical events, when physically implemented in logic devices, correspond to *delay insensitive* (DI) properties.

When implementing logical operations in physical devices, the dependence of physical causality on time-ordering imposes constraints that tend to limit their performance. In practice, one must either control time-orders, hence time delays, at the level required by the operation frequency and size of the device, or design processors so that their operation can resist arbitrary variations of time delays. These options are not exclusive and solutions adopted in existing processors take advantage of both possibilities. Timing methods significantly help improve the performance of clocked circuits, and nowadays, processors routinely include self-timed asynchronous parts to benefit from a locally enhanced level of performance [20]. The increasing cost in both complexity and power consumption, entailed by monitoring time lapses over highly integrated circuits, however leads one to favor approaches that can escape, as much as possible, time delay constraints. The formalization of delay insensitivity (DI) [12, 22, 2, 26] provides a powerful conceptual framework for designing asynchronous circuits. Confronting this approach to the relativistic background of physical processes, we shall try in the following to understand DI rules as the result of general and fundamental constraints imposed by physical causality on the implementation of logic devices.

4 Partial order relations

We now focus on logical operations and on their implementations as processes in physical devices. In general, the result of two successive logical operations depends on the order in which they are performed. At the level of physical processes, the ordering of logical operations is given by the time-ordering of the events at which they occur. For that reason, we shall identify in the following, the time-ordering of logical operations with the ordering of the corresponding events. Upper case letters denoting events will just be replaced by lower case letters when denoting logical operations. We also introduce some simplifying notations to represent ordering when expressed on logical operations rather than events.

Definition: When the event associated with a logical operation b is in the future of the event associated with a logical operation a , we say that a and b are

ordered (in time) and write this relation with the following three symbols

$$a \triangleleft b \tag{11}$$

This relation is *irreflexive*, *anti-symmetric* and *transitive*. We say that a is *before* b or b is *after* a .

Definition: When two logical operations a and c are such that neither relation $a \triangleleft c$ nor relation $c \triangleleft a$ holds, we say that a and c are *contemporary* and write

$$a \bowtie c$$

This relation is *symmetric* but not *transitive*, for $a \bowtie c$ and $c \bowtie b$ can both hold without $a \bowtie b$ being satisfied (see Figure 1).

Relations \triangleleft and \bowtie , which are based on relations between events, admit simple representations on spacetime diagrams. Let us note that relations \triangleleft, \bowtie can also be seen as a particular case of the relations $\longrightarrow, \dashrightarrow$ introduced by Lamport [8] to provide a general formalization of logical time in distributed computations. These definitions however differ in an important way: \triangleleft and \bowtie refer here to an ordering with respect to ‘physical time’, although $\longrightarrow, \dashrightarrow$ refer to a constructed ‘logical time’ [8]. One could alternatively say that the choice to model implementations of logic devices at the level of electronic components leaves less freedom for a representation of time.

As already stated, the expression of causality in physics strongly depends on the notion of time ordering. It should be clear however that causal relations do not reduce to time-order relations. Obviously, two physical actions, in particular two logical operations, can happen to be ordered in time without being causally related. In other words, two logical operations can only be causally related if they are ordered in time, the latter condition being necessary but not sufficient. For that reason, a further notation must be introduced to describe causal relations.

Definition: To express “ a is the cause of b ”, we write

$$a \rightarrow b \tag{12}$$

The constraint between causal and time-order relations may then be written

$$(a \rightarrow b) \Rightarrow (a \triangleleft b)$$

The converse implication does not hold and a configuration showing the difference between causal and time-order relations is illustrated in Figure 2, where E and R represent two components, and a, b and c communications between E and R . Figure 2 illustrates a case where a is the cause of b and c , with no relation of causality between b and c . This gives an example of relation $b \triangleleft c$ holding without relation $b \rightarrow c$ being true.

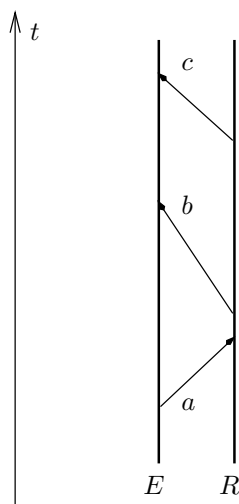


Figure 2: Partial causal order

The causal situation illustrated in Figure 2 is also compatible with another configuration in time, $a \triangleleft (b \bowtie c)$. Such a configuration happens when the environment E is placed farther on the left. This results from the fact that when c is emitted by R one does not know, in the absence of acknowledge, whether b has already been received by E . The time-order realized between operations b and c in Figure 2 is an example of a *delay sensitive* relation. The causal order specified in Figure 2 can be realized with different time orders, $(a \triangleleft b \triangleleft c)$, $(a \triangleleft c \triangleleft b)$ or $(a \triangleleft (b \bowtie c))$. This example shows that a total time-order is not necessary to satisfy all causal relations required by specifications. On the other hand, delay insensitivity forbids situations such as the one illustrated in Figure 2. Delay insensitivity expresses the compatibility of arbitrary timing configurations with the causal order specified by a computation. Hence, it appears as a general and simple way to account for the arbitrariness in time ordering that remains once causal orders are satisfied. Using only DI circuits then allows one to replace the causal order \rightarrow defined in Equations (12) by the time-order \triangleleft defined in Equations (11) when describing the logical constraints associated with specifications.

5 Structures of relativistic traces

In this section, we describe a formalism of relativistic traces (R-traces) for electronic circuits, generalizing standard traces used for DI systems [22, 24, 2, 26] so as to account for the orders defined in the previous section.

For simplicity, in the following, we identify logical operations with the events at which they occur. Symbols consist of lower case letters of the beginning of the

alphabet (a, b, c, \dots) , thus denoting events, while referring to logical operations. The concatenation operator “;” (concatenation of symbols or of strings of symbols) is replaced by operators describing time-ordering, which we choose to be “ \triangleleft ” and “ \bowtie ”. We also add parentheses “(” and “)”. Note that R-traces do not represent the causal order (denoted by “ \rightarrow ”) which will appear later.

R-trace *structures* for a component are defined as triples $S = \langle \mathbf{i}S, \mathbf{o}S, \mathbf{t}S \rangle$, where:

- $\mathbf{i}S$ is a finite set of symbols denoting propagation events from the environment to the component (*inputs*),
- $\mathbf{o}S$ is a finite set of symbols denoting propagation events from the component to the environment (*outputs*),
- $\mathbf{t}S$ denotes a set (finite or not) of finite length strings (named *R-traces*) written with symbols in the set $\mathbf{a}S$, where $\mathbf{a}S$ (*alphabet of S*) is $\mathbf{i}S \cup \mathbf{o}S \cup \{\triangleleft, \bowtie, (,)\}$ and obeys the following syntax.

R-trace structures are denoted by upper case letters (R, S, T, \dots) and *R-traces* are denoted by the lower case letters (s, t, u, \dots) of the end of the alphabet. Moreover, the following properties are assumed to hold.

- The empty R-trace, denoted by “ ε ”, belongs to the structure S : $\varepsilon \in \mathbf{t}S$.
- A single symbol can be an R-trace, for example: $a \in \mathbf{t}S$.
- An R-trace s can be extended with a new symbol a when s is entirely in the past of a ($s \triangleleft a \in \mathbf{t}S$).
- By definition: $\triangleleft \triangleleft = \triangleleft$ and for any R-trace s , $s \triangleleft = s = \triangleleft s$.
- An event symbol in a R-trace can be replaced by two event symbols with no time order between them; this will be written: $(a \bowtie b)$.
- By definition: $a = (a) = (\triangleleft a) = (a \triangleleft) = (a \bowtie) = (\bowtie a)$.
- When an order symbol “ \triangleleft ” precedes parentheses, each symbol within the parentheses denotes an event that is in the future of the sequence preceding the order symbol. For example, $s \triangleleft (a \bowtie b)$ means that, both and independently, s is before a and s is before b .
- When an order symbol “ \triangleleft ” follows parentheses, each symbol within parentheses denotes an event that is in the past of the sequence following the order symbol. For example, $(a \bowtie b) \triangleleft t$ means that t is both after a and after b .

- In an R-trace, an event symbol can be replaced by an R-trace, put inside parentheses. For example, $s \triangleleft ((a \triangleleft b) \bowtie c) \triangleleft t$ is an R-trace. This R-trace means that s is before a and before c , and that t is after b and after c .
- Associativity follows: $(a \bowtie (b \bowtie c)) \equiv ((a \bowtie b) \bowtie c)$, even if the relation “ \bowtie ” is non-transitive, as this expression means that a , b and c can only satisfy a time-order relation with symbols lying outside the parentheses.

As an example, the following R-trace can be part of the history of a Muller C-element

$$(a \bowtie b) \triangleleft c \triangleleft a \triangleleft b \triangleleft c \triangleleft b \triangleleft a \triangleleft c \triangleleft (a \bowtie b) \triangleleft c \quad (13)$$

5.1 Operations on R-Traces

We define the *concatenation* of two R-traces t and u as $t \triangleleft u$. This operation will be extended to concatenation of sets of R-traces. For example, $\mathbf{t}R \triangleleft \mathbf{t}S$ denotes the set of R-traces formed by concatenating one R-trace from $\mathbf{t}R$ and one R-trace from $\mathbf{t}S$.

We define the *star* operation of R-trace t , written $[t]^*$, as the set of finite numbers of concatenations of t

$$[t]^* = \{\varepsilon, t, t \triangleleft t, t \triangleleft t \triangleleft t, t \triangleleft t \triangleleft t \triangleleft t, \dots\}$$

Applied to a set of R-traces, the *star* operator produces the set of all traces obtained from a finite number of concatenations of R-traces of the set.

We define the *prefix* operator in the following way. Let t be an R-trace such that $t \in \mathbf{t}S$; the string u is a prefix of t if $u \in \mathbf{t}S$ and if there exists $v \in \mathbf{t}S$ such that $t = u \triangleleft v$. We will also consider that ε and t are prefixes of t . The **pref** operator applied to t produces the set of all prefixes of t . Applied to a set of traces, it produces the set of prefixes of all R-traces of the set.

Let t be an R-trace, and A a set of symbols including $\{\triangleleft, \bowtie, (,)\}$. We define $t \downarrow A$, called the *projection* of t on A , as the R-trace obtained by only keeping symbols of t belonging to A .

- If $t = a$ and $a \in A$, then $t \downarrow A = a$
- If $t = a$ and $a \notin A$, then $t \downarrow A = \varepsilon$
- If $t = uv$ and $u \downarrow A = u' \neq \varepsilon$ and $v \downarrow A = v' \neq \varepsilon$, then $t \downarrow A = u'v'$
- If $t = uv$ and $u \downarrow A = u' \neq \varepsilon$ and $v \downarrow A = \varepsilon$, then $t \downarrow A = u'$

- If $t = uv$ and $u \downarrow A = \varepsilon$ and $v \downarrow A = v' \neq \varepsilon$, then $t \downarrow A = v'$
- If $t = uv$ and $u \downarrow A = \varepsilon$ and $v \downarrow A = \varepsilon$, then $t \downarrow A = \varepsilon$

As an example of such a projection, R-trace (13), projected on $\{a, c, \triangleleft, \bowtie, (,)\}$ produces the following R-trace

$$a \triangleleft c \triangleleft a \triangleleft c \triangleleft a \triangleleft c \triangleleft a \triangleleft c$$

which is a concatenation of $a \triangleleft c$ and can be rewritten

$$[a \triangleleft c]^4$$

Similar operations can be defined for the following R-trace structures.

- Concatenation:

$$R; S = \langle \mathbf{i}R \cup \mathbf{i}S, \mathbf{o}R \cup \mathbf{o}S, \mathbf{t}R \triangleleft \mathbf{t}S \rangle$$

- Union:

$$R|S = \langle \mathbf{i}R \cup \mathbf{i}S, \mathbf{o}R \cup \mathbf{o}S, \mathbf{t}R \cup \mathbf{t}S \rangle$$

- Star:

$$*[R] = \langle \mathbf{i}R, \mathbf{o}R, [\mathbf{t}R]^* \rangle$$

- Prefix:

$$\mathbf{pref}R = \langle \mathbf{i}R, \mathbf{o}R, \mathbf{pref}[\mathbf{t}R] \rangle$$

(If $\mathbf{pref}R = R$, R is called “prefix-closed”.)

- Projection (on the set A):

$$R \downarrow A = \langle \mathbf{i}R \cap A, \mathbf{o}R \cap A, \{t \downarrow A \mid t \in \mathbf{t}R\} \rangle$$

- Weave:

$$R||S = \langle \mathbf{i}R \cup \mathbf{i}S, \mathbf{o}R \cup \mathbf{o}S, \{t \in (\mathbf{t}R \cup \mathbf{t}S) \mid t \downarrow \mathbf{a}R \in \mathbf{t}R \wedge t \downarrow \mathbf{a}S \in \mathbf{t}S\} \rangle$$

For instance: $R = \langle \{a\}, \{c\}, \{a \triangleleft c\} \rangle$ and $S = \langle \{b\}, \{c\}, \{b \triangleleft c\} \rangle$

$$R||S = \langle \{a, b\}, \{c\}, \{(a \bowtie b) \triangleleft c, a \triangleleft b \triangleleft c, b \triangleleft a \triangleleft c\} \rangle$$

5.2 Component and Environment

The dialog between a *component* and its *environment* is specified by an R-trace structure S (named *command*), which is non-empty ($\mathbf{t}S \neq \emptyset$), and such that $\mathbf{i}S \cap \mathbf{o}S = \emptyset$. Symbols in $\mathbf{i}S$ denote signals emitted by the environment and received by the component. Symbols in $\mathbf{o}S$ denote signals emitted by the component and received by the environment.

One does not make any distinction between the specification of the component on one hand and the specification of the environment on the other hand. One only specifies the *pair* component-environment, defined by a dialog described by a single R-trace structure.

For instance, the dialog between a Muller C-element and its environment is described in the following way

$$\mathbf{pref} * \langle \{a, b\}, \{c\}, \{(a \bowtie b) \triangleleft c, a \triangleleft b \triangleleft c, b \triangleleft a \triangleleft c\} \rangle \quad (14)$$

One can define *atomic commands* as particular R-trace structures

$$\begin{aligned} a? &= \langle \{a\}, \emptyset, \{a\} \rangle \\ b? &= \langle \{b\}, \emptyset, \{b\} \rangle \\ c! &= \langle \emptyset, \{c\}, \{c\} \rangle \end{aligned}$$

so that Equation (14) can then be rewritten

$$\mathbf{pref} * [(a? \| b?); c!]$$

As a consequence of the definition of the *projection* of R-traces, the logical operation $a \| b$ corresponds to the three possible time relations $a \bowtie b$, $a \triangleleft b$, and $b \triangleleft a$. The previous notation “ $\|$ ” for a DI-component is compatible with the classical weave notation (as for example in [6, 2]), but one must note that it now has a different meaning: R-trace structures are a description of a partial order, the time-order relation for events that are propagations in relativistic spacetime. Let us remark that propagation delays are already included in this description, in contrast to the classical description. Also, $a?$ is not an input port of a component, but the propagation of a signal that travels from the environment to the component (see the discussion at the end of Section 2).

6 Relativistic rules

Udding’s rules [23] are a formalization of the intuitive idea underlying DI-systems, which was previously introduced under the form of the Foam Rubber Wrapper (FRW) metaphor [12]. The FRW metaphor discusses how a change of propagation

delays can change the order of signals in time, and suggests ways to protect component specifications against such changes in time-ordering. Udding translated the notion of FRW into a set of logical constraints, to be imposed on trace structures specifying components, so that specifications become invariant under changes of time-ordering due to propagation delays.

In the classical framework of traces [24], a trace is understood to describe a total time-order. When the latter changes as a result of propagation delays, one interprets this property by saying that “the order at the environment boundary is not the same as the order at the component boundary”. One can alternatively say in a relativistic framework that the notion of time-order becomes ambiguous for events at the boundary. This is due to the spatial extension of the boundary between a component and its environment, which entails the impossibility of ascribing a definite time-order to events occurring in this area. In this context, R-traces, which preserve the notion of trace while describing a partial order, appear as a natural generalization. Meanwhile, R-traces bring another important semantic change, as they are not only associated with events occurring at a single location in space, but also with propagation events.

Adapting Udding’s rules, we consider R-traces as constraints imposed on the syntax of component specifications. R-traces now constrain the specification of a single object that is a *component-environment pair* and express the dialog between a component and its environment, and more precisely causality relations satisfied by propagations between them. Let us note that as they are compatible with time-ordering in relativistic spacetime, these causality relations make DI-systems compatible, not only with a change in place&route on a chip at the design stage, but also with changes due to process variations or to motions of components during computation.

In the following, we rewrite Udding’s rules and discuss them using spacetime diagrams. Components and environments (respectively denoted by R and E) are represented by vertical lines. Propagation events are illustrated by slanted arrows which represent bounded and non-vanishing speeds (as in Figure 2). R also denotes the R-trace structure describing the dialog specification of a pair (R, E) . The R_0 rule is a special case of the R_1 rule and is discussed after the latter (we keep for rules, the names introduced by Udding [23]).

6.1 The R_1 Rule

The R_1 rule concerns two successive symbols of the same type (two inputs or two outputs). Classically, in a context of total ordering, the two possible relative orders

have to be taken into account and have to produce the same result

$$\forall s, t \in \mathbf{tR}, (a, b \in \mathbf{iR}) \vee (a, b \in \mathbf{oR}) ; \quad (15)$$

$$sabt \in \mathbf{tR} \Leftrightarrow sbat \in \mathbf{tR}$$

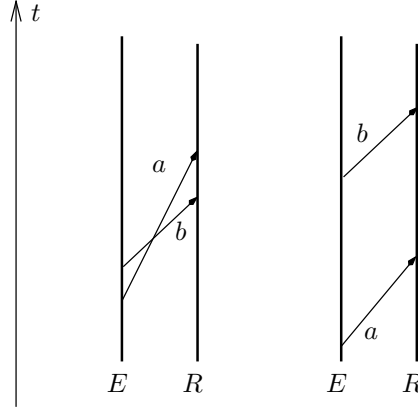


Figure 3: About the R_1 rule (two inputs case)

Let us discuss this rule using Figure 3. On the left, signal a has a lower speed than signal b (a lower speed may be the consequence of indirect communications causing further delays). On the right, signal b is sent at such a date that it lies in the future of a ($a \triangleleft b$). In a classical framework, this order appears differently for E and R in the first case (on the left), while being identical in the second case (on the right). The independence on this relative order of the string following events a and b justifies the classical formulation of the R_1 rule (15). On the other hand, within a relativistic framework, no time order, hence no causal order, can exist between a and b . Consequently, a relativistic version of the R_1 rule should read

$$\forall s, t \in \mathbf{tR}, (a, b \in \mathbf{iR}) \vee (a, b \in \mathbf{oR}) ;$$

$$(\quad s \triangleleft a \triangleleft b \triangleleft t \in \mathbf{tR} \quad \vee \quad s \triangleleft (a \bowtie b) \triangleleft t \in \mathbf{tR} \quad)$$

$$\Rightarrow \quad s \triangleleft (a \parallel b) \triangleleft t \in \mathbf{tR}$$

where $(a \parallel b)$ is a shorthand notation for all possible time-orders $a \triangleleft b$, $(a \bowtie b)$ and $b \triangleleft a$ (see the discussion at the end of previous section). This is the case that was discussed in Section 4. When no signal (acknowledge) can inform E that a has been received before the emission of b , no consensus can be reached by E and R on the order of a and b . This is acceptable if the order between a and b has no consequence on subsequent computations.

6.2 The R_0 Rule

The R_0 rule states that two successive events cannot be identical, i.e. transmitted

on the same channel and of the same type. This rule aims to avoid confusing or loosing events. Classically, it reads

$$\forall s \in \mathbf{tR}, a \in \mathbf{aR}; \quad saa \notin \mathbf{tR}$$

In relativistic spacetime, one must replace saa in the R_1 rule by $s \triangleleft (a||a)$. How should one interpret the rule R_0 ? In fact, two events a with no causal relation are independently caused by the same preceding event s . If one insists on repeating an event a , the second one has to come after an acknowledgement of the first one, in order to avoid any confusion. The R_0 rule may then be rewritten

$$\forall s \in \mathbf{tR}, a \in \mathbf{aR}; \quad s \triangleleft (a||a) \notin \mathbf{tR}$$

6.3 The R_2 Rule

The R_2 rule concerns two symbols of opposite type (one input and one output). In general, these two events may be causally ordered and this order may have a meaning. No ambiguity arises in that case. This corresponds to $a \rightarrow b$, which we implement by $a \triangleleft b$. But if these two events are not ordered, or if they are ordered while the two different orders remain possible, then these possibilities have to entail the same consequences, as propagation delays can change the order. Classically, this rule reads

$$\begin{aligned} & \forall s, t \in \mathbf{tR}, (a \in \mathbf{iR} \wedge b \in \mathbf{oR}) \vee (a \in \mathbf{oR} \wedge b \in \mathbf{iR}); \\ & (sab \in \mathbf{tR} \wedge sba \in \mathbf{tR}) \Rightarrow (sabt \in \mathbf{tR} \Leftrightarrow sbat \in \mathbf{tR}) \end{aligned}$$

Let us discuss this rule using Figure 4. On the right, a and b signals are clearly not ordered ($a \bowtie b$ holds). But on the left, which represents $a \triangleleft b$, one cannot determine whether a causal order holds ($a \rightarrow b$) or whether it is a special case of absence of order ($a||b$).

The purpose of the R_2 rule is to force one, when implementing ($a||b$), to make the case $a \triangleleft b$ have the same consequences as $b \triangleleft a$ or $a \bowtie b$. The R_2 rule may then be rewritten

$$\begin{aligned} & \forall s, t \in \mathbf{tR}, (a \in \mathbf{iR} \wedge b \in \mathbf{oR}) \vee (a \in \mathbf{oR} \wedge b \in \mathbf{iR}); \\ & \left[\begin{array}{l} (s \triangleleft a \triangleleft b \in \mathbf{tR} \quad \wedge \quad s \triangleleft b \triangleleft a \in \mathbf{tR}) \\ \vee \quad s \triangleleft (a \bowtie b) \in \mathbf{tR} \end{array} \right] \\ & \Rightarrow \\ & \left[\begin{array}{l} (s \triangleleft a \triangleleft b \triangleleft t \in \mathbf{tR} \quad \vee \quad s \triangleleft (a \bowtie b) \triangleleft t \in \mathbf{tR}) \\ \Rightarrow \quad s \triangleleft (a||b) \triangleleft t \in \mathbf{tR} \end{array} \right] \end{aligned}$$

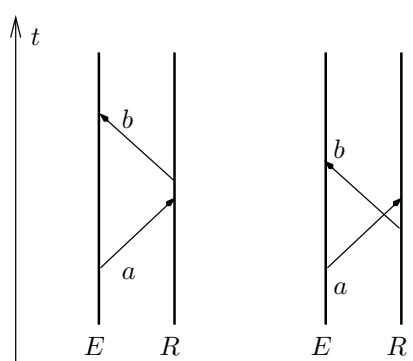


Figure 4: About the R_2 rule

6.4 The R'_2 Rule

The R_2 rule, under the form just discussed, may be too constraining in some cases. This is better seen again using Figure 4. Let us assume that the two drawings represent the same non-ordered case $s \triangleleft (a \parallel b)$. On the left, R can be seen to have more information than E on the time ordering of a and b . R also knows that E has to see the same order. On the other hand, E cannot infer the time order seen by R , as it has no knowledge of the propagation delays. For instance, E could infer $(a \bowtie b)$. In that case, only R can be authorized to make a decision. For instance, R can emit d if a and b are not causally ordered while $a \triangleleft b$ holds (see Figure 5).

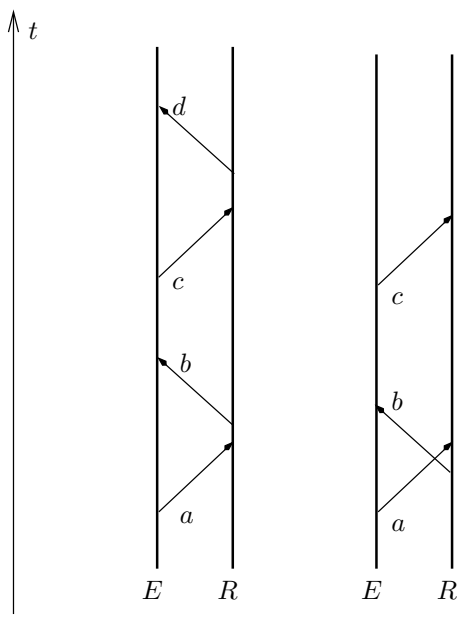


Figure 5: About the R'_2 rule

If an event c is emitted with the same type as a , c must exist whatever the order seen by R , because c is emitted by E , which has less information than R . This is ensured by the R'_2 rule, which classically reads

$$\begin{aligned} & \forall s, t \in \mathbf{tR}, (a, c \in \mathbf{iR} \wedge b \in \mathbf{oR}) \\ & \quad \vee (a, c \in \mathbf{oR} \wedge b \in \mathbf{iR}) ; \\ & (sabt c \in \mathbf{tR} \wedge sbat \in \mathbf{tR}) \Rightarrow sbatc \in \mathbf{tR} \end{aligned}$$

R'_2 may then be rewritten

$$\begin{aligned} & \forall s, t \in \mathbf{tR}, (a, c \in \mathbf{iR} \wedge b \in \mathbf{oR}) \\ & \quad \vee (a, c \in \mathbf{oR} \wedge b \in \mathbf{iR}) ; \\ & \left[(s \triangleleft a \triangleleft b \triangleleft t \triangleleft c \in \mathbf{tR} \wedge s \triangleleft b \triangleleft a \triangleleft t \in \mathbf{tR}) \right. \\ & \quad \left. \vee s \triangleleft (a \bowtie b) \triangleleft t \triangleleft c \in \mathbf{tR} \right] \\ & \Rightarrow s \triangleleft (a \parallel b) \triangleleft t \triangleleft c \in \mathbf{tR} \end{aligned}$$

6.5 The R_3 Rules

The R_3 rules concern the possibility (or impossibility) for two symbols to be mutually exclusive. For instance, the design of a logic device may result in the following property. If a and b are respectively input and output symbols, sharing no causal order ($a \parallel b$), a reception of a prevents b from being emitted as soon as a precedes b ($a \triangleleft b$). If the design has to avoid such a possibility, the latter must be excluded by a rule. Classically, the corresponding rule reads

$$sa \in \mathbf{tR} \wedge sb \in \mathbf{tR} \Rightarrow sab \in \mathbf{tR} \quad (16)$$

Extended to any combination of symbols of the same type, this rule can be used to classify logic devices:

- Two input symbols can be mutually exclusive if the environment has to make a choice (data communication devices).
- Two output symbols can be mutually exclusive if the component has to make a choice (arbitration devices).

The R_3 rules aim at forbidding such choices. For instance, data communication devices are excluded with the following rule: $\forall s \in \mathbf{tR}, a \in \mathbf{iR} \wedge b \in \mathbf{iR}$; then (16). In a relativistic framework, the second part of expression (16) must be replaced by the following $s \triangleleft (a \parallel b) \in \mathbf{tR}$ so that expression (16) should be rewritten

$$\begin{aligned} & \left[s \triangleleft (a \bowtie b) \in \mathbf{tR} \quad \vee \quad (s \triangleleft a \in \mathbf{tR} \wedge s \triangleleft b \in \mathbf{tR}) \right] \\ & \Rightarrow s \triangleleft (a \parallel b) \in \mathbf{tR} \end{aligned} \quad (17)$$

Inserting conditions bearing on different types of symbols before expression (17) generates a group of three different R_3 rules.

R'_3 **rule**: this first rule defines the most constrained class of components, where neither the environment, nor the component can make a choice (Muller-C is in this class)

$$\forall s \in \mathbf{t}R, \quad a \neq b \in \mathbf{a}R; \quad \text{then (17).}$$

R''_3 **rule**: this second rule accepts data communication devices, as the corresponding condition does not constrain two inputs (a choice can be made on two inputs only)

$$\forall s \in \mathbf{t}R, \quad a \neq b \in \mathbf{a}R \quad a \notin \mathbf{i}R \vee b \notin \mathbf{i}R; \quad \text{then (17).}$$

R'''_3 **rule**: according to this third rule, not only two inputs but also two outputs can be mutually exclusive (arbiters are in this class)

$$\forall s \in \mathbf{t}R, (a \in \mathbf{i}R \wedge b \in \mathbf{o}R) \vee (a \in \mathbf{o}R \wedge b \in \mathbf{i}R); \text{ then (17).}$$

The three R_3 rules have been ordered according to decreasing constraints put on specifications. They allow one to generalize Udding's classes of DI-components to the relativistic case.

6.5.1 Relativistic DI-components

By definition, relativistic DI-components have specifications that obey R_0 , R_1 , R'_2 and R'''_3 rules.

7 Classical rules

We now discuss how relativistic traces reduce to classical traces when communications can be considered to occur in a classical spacetime (i.e. assuming that propagations and time delays are totally controlled, at the level of elementary logical operations). Accordingly, relativistic rules reduce to their classical analogs. This reduction amounts to a mapping that transforms R-traces into classical traces. In particular, the set of symbols a , b , \triangleleft , \bowtie is reduced when operations are restricted to a classical environment which is more constraining, as symbol \bowtie is not supported any more.

Let us note that an R-trace describes a partial time-ordering that applies in particular to *propagations*, although a classical trace describes a total time-ordering which refers to *points* in spacetime. Both notions however coincide when applied to events, i.e. to signals at the input or output ports of a component. Hence, although a symbol a does not a priori denote the same object in a R-trace as in

a trace, it is legitimate to define its mapping from a R-trace to a classical trace. One obtains the following mapping.

- $a \triangleleft b \in \mathbf{tR}$ is mapped onto $ab \in \mathbf{tR}$.
- $(a \bowtie b) \in \mathbf{tR}$ is mapped onto $ab \in \mathbf{tR} \wedge ba \in \mathbf{tR}$.
- $a \parallel b \in \mathbf{tR}$ is mapped onto $ab \in \mathbf{tR} \wedge ba \in \mathbf{tR}$.

This mapping amounts to replacing the absence of time-order, unavoidable in a relativistic framework, by the two possible orders, as imposed by the syntax of classical traces. As illustrated by spacetime diagrams, one R-trace maps onto a pair of classical traces. This corresponds to the two different geometrical situations that can result when two intersecting slanted arrows are made to separately intersect the two vertical lines representing a component and its environment. Both resulting traces have to be present in the specification of the component. Indeed, all relativistic rules end with the form

$$\Rightarrow (a \parallel b) \in \mathbf{tR}$$

which is mapped onto the expression

$$\Rightarrow ab \in \mathbf{tR} \wedge ba \in \mathbf{tR}$$

In other words, all these rules concern the absence of causal order and their restriction to classical spacetime amounts to imposing the presence of the two opposite orders. The resulting rules are easily seen to correspond to the standard ones. In fact, one can note that it is precisely because Udding's rules have the property of imposing the presence of the two opposite orders that they can be obtained as restrictions of relativistic rules. This is the reason why the notion of DI-systems can be generalized to relativistic spacetime with its partial time-ordering.

8 Conclusion

The relativistic and classical frameworks lead to conceptions of time that differ in a significant way. Time-simultaneity and total ordering in time are classical properties. In contrast, relativistic time can only satisfy a partial ordering to remain compatible with observer-dependent propagation delays. Practical methods depending on the classical properties of time may reach a point where they cease to be sufficient for a circuit to properly function, due to propagation delays. The models used to design logic devices can then be improved by releasing these properties and looking for a better implementation of relativistic causality. In particular,

causal relations can be made compatible with the impossibility for some events to satisfy a time order [9].

Delay Insensitivity represents an efficient criterion for designing functional clockless circuits in a simple way. But usual models, such as the trace formalism, rely on a classical representation of time-ordering and causality. Nonetheless, traces can be generalized to R-traces which provide a similar description of the dialog between a DI component and its environment while being compatible with relativistic causality. The usual formalism [22, 23, 24] is recovered when implementations of components are restricted to a classical environment.

An R-trace structure describes the specification of a component-environment pair as the list of all time orders that can hold between the communication events of the pair dialog. When rewritten, Udding's rules keep a simple form that clarifies their relation with causality properties. Although the R-trace formalism, when applied to specifications of DI components, takes a form similar to its classical analog [2, 3], interpretations differ significantly. R-trace structures describe the time-order and causality relations of all events, including signal propagations, while classical structures only refer to events that are localized in time.

The biggest merit of the R-trace formalism is to clarify the semantics underlying Udding's rules by exhibiting their intrinsic relation with time-ordering and physical causality. Further theoretical insight, with potential consequences for applications, could be gained by comparing R-traces to other existing formalisms dealing with Delay Insensitivity. Criteria have indeed been developed to perform such comparisons as, for instance, the strength of semantic properties in equivalence relationships [25].

Besides a clear representation of causal relations, another merit of the R-trace formalism is to replace the description of a distributed system as a set of individual components that communicate by a composition of component-environment pairs. This rather describes the composition of *interfaces*, where *interface* means the dialog between the two members of a pair. In contrast to alternative descriptions of causality, such as Petri nets, this conception allows a simple composition of multiple components, by considering the latter as a multiplication of interfaces. This approach also opens new and interesting possibilities for solving questions raised by the synchronization of complex distributed systems [7, 8, 10]. A further advantage of the formalism is its ability to include in the description, with minor changes, components in motion. This could also allow the treatment of distributed algorithms on a network of mobile computers within the formalism of DI systems.

References

- [1] T.-A. Chu, C. K. C. Leung, and T. S. Wanuga. A design methodology for concurrent VLSI systems. In *Proceedings of the International Conference on Computer Design (ICCD)*, pages 407–410. IEEE Computer Society Press, 1985.
- [2] Jo C. Ebergen. *Translating Programs into Delay-Insensitive Circuits*. PhD thesis, Technische Universiteit Eindhoven, October 1987.
- [3] Jo C. Ebergen. A formal approach to designing delay-insensitive circuits. *Distributed Computing*, 5:107–119, 1991.
- [4] Albert Einstein. Zur elektrodynamik bewegter körper. *Annalen der Physik*, 17:891–921, 1905.
- [5] Albert Einstein. *The collected papers of Albert Einstein*, volume 2. Princeton University Press, 1989.
- [6] C. A. R. Hoare. Communicating sequential processes. *Communications of the ACM*, 21(8):666–677, 1978.
- [7] Leslie Lamport. Time, clocks, and the ordering of events in a distributed system. *Communications of the ACM*, 21(7):558–565, July 1978.
- [8] Leslie Lamport. On interprocess communication. part I: Basic formalism. *Distributed Computing*, 1(2):77–85, june 1986.
- [9] Philippe Matherat and Marc-Thierry Jaekel. Concurrent computing machines and physical space-time. *Mathematical Structures for Computer Science (MSCS)*, 13(5):771–798, octobre 2003.
- [10] F. Mattern. On the relativistic structure of logical time in distributed systems. In *Parallel and Distributed Algorithms*, pages 215–226. Elsevier Science Publishers B.V, 1992.
- [11] A. Mazurkiewicz. Basic notions of trace theory. In J. W. de Bakker, W. P. de Roever, and G. Rozenberg, editors, *Linear Time, Branching Time, and Partial Order in Logics and Models for Concurrency. School/Workshop, Noordwijkerhout, may-june 1988*, number 354 in LNCS, page 285. Springer-Verlag, Berlin, 1989.
- [12] Charles E. Molnar, Ting-Pien Fang, and Frederik U. Rosenberger. Synthesis of delay-insensitive modules. In H. Fuchs, editor, *Proceedings of the 1985 Chapel Hill Conference on VLSI*, pages 67–86. Computer Science Press, 1985.

- [13] Tadao Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580, April 1989.
- [14] C. A. Petri. *Kommunikation mit Automaten*. PhD thesis, Institut für Instrumentelle Mathematik, Bonn, 1962.
- [15] David K. Probst and Hon Fung Li. Using partial-order semantics to avoid the state explosion problem in asynchronous systems. In Edmund M. Clarke and Robert P. Kurshan, editors, *CAV*, volume 531 of *LNCS*, pages 146–155. Springer-Verlag, 1990.
- [16] L. Ya Rosenblum and A. V. Yakovlev. Signal graphs: from self-timed to timed ones. In *Proceedings of the International Workshop on Timed Petri Nets*, pages 199–207, Torino, Italy, July 1985. IEEE Computer Society Press.
- [17] Bertrand Russell. *ABC of Relativity*. Routledge 1997, 1925.
- [18] Bertrand Russell. Philosophical Consequences of Relativity. *Encyclopaedia Britannica*, 1926.
- [19] Bertrand Russell. *The Analysis of Matter*. Routledge 1992, 1927.
- [20] Ken Stevens, Ran Ginosar, and Shai Rotem. Relative Timing. *IEEE Transactions on VLSI*, 1(11):129–140, February 2003.
- [21] Ivan E. Sutherland. Micropipelines. *Communications of the ACM*, 32(6):720–738, June 1989.
- [22] Jan Tijmen Udding. *Classification and Composition of Delay-Insensitive Circuits*. PhD thesis, Eindhoven University of Technology, The Netherlands, September 1984.
- [23] Jan Tijmen Udding. A formal model for defining and classifying delay-insensitive circuits and systems. *Distributed Computing*, 1:197–204, 1986.
- [24] Jan L. A. van de Snepscheut. *Trace Theory and VLSI Design*. Number 200 in LNCS. Springer-Verlag, 1985.
- [25] Robert Jan van Glabbeek. The Linear Time - Branching Time Spectrum (extended abstract). In J.C.M. Baeten and J.W. Klop, editors, *Proceedings of The International Conference on Concurrency Theory (ConCur)*, number 458 in LNCS, pages 278–297, Berlin, 1990. Springer Verlag.
- [26] Tom Verhoeff. *A Theory of Delay-Insensitive Systems*. PhD thesis, Eindhoven University of Technology, 1994.