Verification via Digitized Models of Real-Time Hybrid Systems

Dang Van Hung ∗ and Ko Kwang II †

The United Nations University
International Institute for Software Technology
P.O.Box 3058, Macau
{dvh,kki}@iist.unu.edu

Abstract

The paper presents an approach to the specification and verification of real-time hybrid systems using Duration Calculus (DC) [11]. By introducing a formula int to DC to express the intervals of time between two ticks of the system clock, it is shown that DC can be used to specify both the digital states and the analog states as well as to reason about time in distributed systems. We illustrate our approach by giving an intuitive model for the communication protocols in which the verification can be done by using the familiar natural induction rules.

1. Introduction

In the Real-Time Hybrid systems, we have to reason about their timed behavior (see, e.g. [8, 9, 10]). Since in the such systems we have to deal with analog components whose states change continuously, and with digital components (computer programs) whose states change only at ticks of the computer clock, we have to deal with both continuous time and discrete time. Here we consider continuous time as the set $\mathbb{R}^+$ of nonnegative real numbers and discrete time as the set $\mathbb{N}$ of natural numbers, which is a countable subset of $\mathbb{R}^+$.

The Duration Calculus (DC) [11], an extension of interval temporal logic, is a notation to specify and reason about properties of systems and is a logic to verify theorems about such specifications. In DC, states of a system are modeled by boolean functions of continuous time which have finite variability, meaning that in any finite interval of time there is only a finite number of discontinuous points of the functions. Thus time in DC is continuous and we can reason about the behavioral timing properties of both analog components and computer programs. When restricted to discrete time DC can specify precisely the behavior of computer programs referring to unique clock, and many properties of the systems written in DC are decidable (see [2]). However, in this case it can only specify approximately the behavior of physical analog components, and if the systems are distributed, it cannot specify precisely the behavior of the computer programs that run on different components of the systems with different clocks. Thus, in specifying the hybrid and distributed systems, continuous time should be referred to. A problem with DC when using continuous time is that it cannot specify that a state can change only at discrete time points.

To overcome this shortcoming, we introduce a formula int into DC. The formula int is satisfied by an interval $[a, b]$ of time if and only if $a, b \in \mathbb{N}$, i.e., an interval from a tick to another tick of the reference clock. We say that a state is discrete if it can change its value only at discrete time points.

With discrete states we can define a digitization of a continuous state. Let us consider a model of communication protocols. In order to send a sequence of bits in $\{0, 1\}^*$, the sender encodes the sequence as a state and put it on the bus. The receiver then digitizes the signal and gets a discrete state according to its clock. Because of some disturbance, such as it taking a significant time to change the signal on the bus from high to low and vice-versa, and because of digitization, the state that the receiver gets is different from the signal sent by the sender. However, there is a relationship between the continuous state sent and the discrete state received, which should be satisfied for a protocol under the physical laws. Let $f(w)$ be a DC formula for the state resulted from encoding a sequence $w$ of bits, $D$ be the discrete state obtained by digitizing $f(w)$, and $g(D)$ be the sequence of bits obtained by decoding $D$. The relationship can be then expressed in DC as the formula $f(w) \Rightarrow D$. The correctness of the protocol is verified if $g(D) = w$. Hence, we can have a nice model for specification and verification of

∗ On leave from the Institute of Information Technology, Nghia Do, Tu Liem, Hanoi, Vietnam.
† On leave from POSTECH/CAST, South Korea from January to February 1995
the communication protocols in which the encoding and decoding can be written as functions between formal (regular) languages, the correctness of the protocols can be proved by using the natural induction rule on the length of words in the traditional way.

To illustrate our approach, we try the approach to model and verify the Biphasic Mark (BPM) protocol and the Audio Control protocol which are widely used in practice for asynchronous communication between two digital hardware devices. In [7], the Biphasic Mark protocol is verified formally using Boyer-Moore Logic for some specific encoding and decoding rules. In our model, however, we can verify the BPM protocol for the more general encoding and decoding rules. In our model, however we can verify the BPM protocol for the more general encoding and decoding rules.

The paper is organized as follows. In Section 2, we present Discrete Duration Calculus (DC) which is an extension of DC with discrete states. In Section 3, a general model for physical environments of communication protocols is specified and reasoned in DC. Based on the model, a specification and verification of the Biphasic Mark protocol is presented in Section 4. The last section is a conclusion of the paper.

2. Discrete Duration Calculus: a Duration Calculus with discrete states

In this section, we give an extension of DC, Discrete DC, to reason about the hybridity. First, we give a brief summary of DC and then our extension. For more details of DC, readers are referred to [11, 3].

2.1. Summary of Duration Calculus

Time in DC is the set \( \mathbb{R}^+ \) of the nonnegative real numbers. For \( t, t' \in \mathbb{R}^+(t \leq t') \), \([t, t']\) denotes the time interval from \( t \) to \( t' \). Let \( \text{intv} \) denote the set of all time intervals.

2.1.1 Syntax of Duration Calculus

Assume that \( V = \{X, Y, \ldots\} \) is a set of state variables. We will use the special symbol \( \ell \) to denote the length of a time interval, \( f^n_\ell \) and \( A^n_\ell \) \((i, n \geq 0)\) as \( n \)-ary function and \( n \)-ary predicate names respectively. The syntax classes state expressions, terms, and formulas will then be defined as follows.

**State expressions:** The set of state expressions is generated by the grammar

\[
P \equiv 0 \mid 1 \mid v \mid \neg P \mid P \land Q,
\]

where \( v \) stands for state names in the set \( V \).

**Terms:** The set of terms is generated by

\[
r \equiv \int P \mid \ell | f^n(r_1, \ldots, r_n),
\]

where \( P \) stands for state expressions, \( f^n \) for an \( n \)-ary function.

**Atomic Formulas:** if \( r_1, \ldots, r_n \) are \( n \) terms then \( A^n_\ell(r_1, \ldots, r_n) \) is an atomic formula.

**Formulas:** the set of formulas is generated by the grammar:

\[
D \equiv A \mid D \mid \neg D \mid D \lor D
\]

where \( A \) stands for atomic formulas.

2.1.2 Semantics of Duration Calculus

Assume that each \( n \)-ary function name \( f^n_\ell \) is associated with a total function from \( \mathbb{R}^n \) to \( \mathbb{R} \) which is denoted by \( f^n_\ell \) also, and each \( n \)-ary predicate name \( A^n_\ell \) is associated with a total function from \( \mathbb{R}^n \) to \{\( tt, ff \} \) which is also denoted by \( A^n_\ell \).

In this paper, for simplicity we interpret the functions \( f \) as operators on reals, e.g. \( +, *, \) and the relations \( A \) as comparative operators between reals, e.g. \( <, \leq, =, >, \geq \).

An interpretation \( I \) is a function \( I \in (V \rightarrow (\mathbb{R} \rightarrow \{0, 1\})) \), for which each \( I(X), X \in V \) has at most finitely many discontinuity points in any interval \([a, b]\). We shall use the abbreviation \( X_I \equiv I(X) \). The semantics of state expression, terms, and formulas in an interpretation \( I \) are then defined as follows.

**Semantics of state expressions:** The semantics of a state expression \( P \) in an interpretation \( I \) is a function \( I_P \in \text{intv} \rightarrow \{0, 1\} \) defined inductively on the structure of state expressions by:

\[
I_P(t) \equiv 0,
I_P(t) \equiv 1,
I_P Xt(t) \equiv X_I(t),
I_P(\neg P(t)) \equiv \neg I_P(t), \text{ and}
I_P(P \lor Q(t)) \equiv \begin{cases} 0 & \text{if } I_P(t) = 0 \text{ and } I_Q(t) = 0, \\ 1 & \text{otherwise.} \end{cases}
\]

**Semantics of terms:** The semantics of a term \( r \) in an interpretation \( I \) is a function \( I_r \in \text{intv} \rightarrow \mathbb{R} \) defined inductively on the structure of terms by:

\[
I_r([a, b]) \equiv \int f^b_a I_P(t)dt,
I_r([a, b]) \equiv b - a, \text{ and}
I_r([r_1, \ldots, r_n])([a, b]) \equiv f^n_I(r_1([a, b]), \ldots, r_n([a, b])).
\]

**Semantics of formulas:** The semantics of a formula \( D \) in an interpretation \( I \) is a function \( I_D \in \text{intv} \rightarrow \{tt, ff\} \) defined inductively on the structure of formulas as follows. Using the following abbreviations:

\[
I_D([a, b]) = tt \quad \text{and} \quad I_D([a, b]) = ff,
\]

2
The syntax of DC\(d\) is:
\[
D \equiv A \mid int \mid D \mid \neg D \mid D \lor D,
\]
where \(A\) stands for atomic formulas of DC.

The semantics for DC\(d\) is defined in the same way as for DC with semantics of int defined by:
\[
\mathcal{I}, [a, b] = int iff [a, b] \in Nintv.
\]

Axioms and rules for DC\(d\) are those of DC. In addition, DC\(d\) needs some others for int.

11 \(\ell \geq 1 \Rightarrow \ell < 1\; int; \ell < 1\)

12 For any formula \(B, int; B \Rightarrow int; (B \land int); int\)

13 For any formulas \(A, B, B\; int \Rightarrow (int; D \Rightarrow int; (D \land int))\) and \(int \Rightarrow (D; int \Rightarrow (D \land int); int)\)

(Natural Induction) Let \(X\) be a formula letter occurring in the formula \(D(X)\). Then
\[
\begin{align*}
D(int \land \ell = 0) & \land \ell < 1 \Rightarrow (int \land X); \ell = 1 \text{ is provable from } D(int \land X); \\
D(int \land \ell = 0) & \land \ell < 1 \Rightarrow (int \land X) \text{ is provable from } D(int \land X); \text{ then } D(int) \text{ holds.}
\end{align*}
\]

It is easy to prove that the axioms introduced for int are sound.

Now we give some rules that will be used in this paper.

The soundness of the rules can be proved easily from the definition of DC\(d\) semantics.

Some additional rules: For any state variables \(P, Q, \) for any formulas \(D_1, D_2, D'_1, D'_2\)

R1 If \(P \Rightarrow Q\) then \([P] \Rightarrow [Q]\)

R2 \([(P) \land (Q)] \Rightarrow [P \land Q]\)

R3 \([(P); true] \land [(Q); true] \Rightarrow [P \land Q]; true, and (true; [P]) \land (true; [Q]) \Rightarrow true; [P \land Q] \]

2.2. Duration calculus with discrete states

Since we want to reason about discrete time in the same framework, we use the set \(\mathbb{N}\) of natural numbers as a subset of \(\mathbb{R}^+\) to express discrete time. The set \(N\) of all intervals over \(\mathbb{R}^+\) with the endpoints in \(\mathbb{N}\) is defined by \(N = \{[a, b] \mid a, b \in \mathbb{N}\}\), and then \(N \subset int\). We introduce a formula int into DC to express the intervals in \(N\) resulting in a "discrete DC". It turns out later that discrete DC (DC\(d\)) then is powerful enough to deal with discrete variables.

2.1.3 Abbreviations, axioms, and theorems

For a DC formulas \(D, D_1, \) and a state expression \(P, \) we use the following abbreviations:

\[
\begin{align*}
\Diamond D & \equiv (true; D); true \\
\Box D & \equiv \neg(\Diamond(\neg D)) \\
[P] & \equiv \{P = \ell \land \ell > 0\} \\
[\bot] & \equiv \ell = 0 \\
D \Rightarrow D_1 & \equiv (D_1 \lor D) \\
D \land D_1 & \equiv (D \lor (\neg D_1)) \\
\end{align*}
\]

Some axioms and theorems in the calculus are:

DA 1 \(\int 0 = 0\)

DA 2 For an arbitrary state \(P, \int P \geq 0\)

DA 3 For arbitrary states \(P, Q, \int P + \int Q = \int(P \lor Q)\)

DA 4 For a state \(P, \) for nonnegative real numbers \(r, s, \int P = r + s\) \(\Rightarrow (\int P = r; \int P = s)\)

(Monotonicity) If \(D_1 \Rightarrow A_1\) and \(D_2 \Rightarrow A_2\) then \(D_1; D_2 \Rightarrow A_1; A_2\)

(Associativity) \((D_1; D_2); D_3 \Rightarrow D_1; (D_2; D_3)\)

(Unit) \([\bot]; D \Rightarrow D; [\bot] \Rightarrow D\)

(Zero) \(false; D \Rightarrow D; false \Rightarrow false\)

(Other) \((\ell = a; A) \land (\ell = a; B) \Rightarrow (\ell = a; A \land B)\)

The set \(N\) of all intervals over \(\mathbb{R}^+\) with the endpoints in \(\mathbb{N}\) is defined by \(N = \{[a, b] \mid a, b \in \mathbb{N}\}\), and then \(N \subset int\). We introduce a formula int into DC to express the intervals in \(N\) resulting in a "discrete DC". It turns out later that discrete DC (DC\(d\)) then is powerful enough to deal with discrete variables.
Assume that points only are called discrete states. Discrete states have a specific property which will determine a unique point in the interior. If the last change of a state in an interval which satisfies a given condition, we can use the DC formula:

\[
\begin{align*}
D_1 & \land \left( \text{true; } [P] \land \ell \geq a \right) ; \\
D_2 & \land \left( \left[ \neg P \right] \land \ell \geq b; B \right)
\end{align*}
\]

More generally, for a state \(P\) and a discrete state \(R_P\) we define a discrete state \(S_P\) to be a sampling state of \(P\) w.r.t. \(R_P\) by:

\[
I(S_P) \land I(R_P),
\]

where \(B \equiv \neg\Diamond\left( [P] \land \ell \geq a; [\neg P] \land \ell \geq b \right)\).

The rule R4 needs some explanation. It says that the first or the last change of a state in an interval which satisfies a specific property will determine a unique point in the interval.

**Discrete states**

The states that can be discontinuous at discrete time points only are called discrete states. Such states could be assertions of program states, for instance, \(x = 6\), where \(x\) is a program variable. To specify that \(P\) is a discrete state, we can use the DC formula

\[
I(P) \equiv \text{int} \land \ell = 1 \Rightarrow [P] \lor [\neg P].
\]

The following theorem can be proved easily.

**Theorem 1.** Assume that \(I(P), I(Q)\) hold. Let \(D, D'\) be DC formulas. Then

1. \(I(P \land Q), I(\neg P)\)
2. \(\text{int} \land ([P]; [Q]) \Rightarrow ([P] \land \text{int}; (\text{int} \land [Q])\)
3. \([P]; [\neg P] \Rightarrow [P]; (\text{int} \land [\neg P]); [P].\)

**Digitization of states**

The idea of digitization here is taken from [6]. Suppose that a device monitors the state \(Q\) over cycles of clock, returning 1 if \(Q\) is true over the whole cycle, 0 if it is false for the whole cycle, and any value chosen nondeterministically in \(\{0, 1\}\) otherwise. The result is said to be a digitization of \(Q\). We could define formally the digitization as follows. A state \(P\) is a digitization of a state \(Q\) iff \(I(P)\) and

\[
\text{int} \land \ell = 1 \Rightarrow [Q] \Rightarrow [P] \text{ and }\]

\[
\text{int} \land \ell = 1 \Rightarrow [\neg Q] \Rightarrow [\neg P].
\]

More generally, for a state \(P\) and a discrete state \(R_P\) we define a discrete state \(S_P\) to be a sampling state of \(P\) w.r.t. \(R_P\) by:

\[
I(S_P) \land I(R_P),
\]

where \(B \equiv \neg\Diamond\left( [P] \land \ell \geq a; [\neg P] \land \ell \geq b \right)\).

With these discrete state and digitization concepts, a general model of physical environment where a communication protocol works is defined in the next section.

### 3. A model of communication at physical layer

Consider a model for communication at the physical layer. A sender and a receiver are connected via a bus. Their clocks are running at different rates. We refer to the clock of the receiver as a reference time. The receiver receives signals by digitizing. The signals sent by the sender are modeled by a state \(X\) (we recall that a state is a boolean function from \(\mathbb{R}^+\) to \(\{0, 1\}\)). Let \(Y\) be the state representing signals received by the receiver by digitizing the signals on the bus. Without loss of generality, assume that the delay between the sender and the receiver is 0. Due to the fact that it takes a significant amount of time to change the signal on the bus from high to low or vice-versa, \(Y\) is not only for expressing the digitization of \(X\) as shown in Figure 1. The value of \(Y\) is chosen arbitrarily from \(\{0, 1\}\) if the digitized value is neither 0 nor 1. Let \(R\) be the state expressing the reliability of the digitization. Then \(Y\) and \(R\) are discrete states according to the receiver’s clock, while \(X\) is not (although it can be considered as a discrete state according to the sender clock) (see Figure 1). In this paper we assume that the states expressing signals are continuous from the left hand side.

Since a state can be specified by a DC formula, a communication protocol can be modeled as consisting of a language \(\mathcal{L}_s\), whose elements are DC formulas over state variable \(X\), a language \(\mathcal{L}_r\), whose elements are DC formulas over discrete state variable \(Y\), a coding function \(f: \{0, 1\}^+ \rightarrow \mathcal{L}_s\) and a decoding function \(g: \mathcal{L}_r \rightarrow \{0, 1\}^+\) such that

1. for any \(w \in \{0, 1\}^+\) there exists a formula \(D \in \mathcal{L}_r\) for which \(f(w) \Rightarrow D\), and
2. for any \(D \in \mathcal{L}_r\), if \(f(w) \Rightarrow D\) then \(g(D) = w\).
Let us use DCₑ to express the relationship between the states X, R and Y before specifying any communication protocol.

(i) Y and R are discrete states, and R expresses the reliability of the digitization:

\[ \text{D1} \quad I(Y) \land I(R), \]

\[ \text{D2} \quad [R \Rightarrow (X \Leftrightarrow Y)], \]

\[ \text{D3} \quad \text{int} \land \ell = 1 \land ([X] \lor [\neg X] \lor [\neg X] \lor [X]) \Rightarrow [\neg R]. \]

(ii) Assumptions about physical laws and digitizations:

Let us adopt the following phenomenon: If the sender changes the signal from low to high or high to low, the receiver’s signal will be unreliable for \( r \) cycles, \( r \geq 1, r \in \mathbb{N} \) during which it can be any value chosen nondeterministically among 0 and 1; otherwise, if the sender keeps the signal stable then the same signal will be stable for the receiver (see Figure 1). These assumptions can be expressed by the following DCₖ formulas:

\[ \text{A1} \quad \ell \geq r + 1 \land ([X] \lor [\neg X]) \Rightarrow \ell \leq r; \left( \text{int} \land \frac{\ell}{R} \right); \ell < 1, \]

\[ \text{A2} \quad \ell \leq r; \left( \text{int} \land \frac{\ell}{R} \right); \ell = r \]

The assumption A1 says that if the sender stays high or low for enough long time then the signal received by the receiver is also the same and stable for enough time. The assumption A2 says that if the sender changes the signal then the signal received by the receiver may not be stable for \( r \) sampling cycles provided the sender has kept the signal stable before the change and keeps it stable after the change for enough long time. From these assumptions it follows that if the signal changes at a tick of the receiver’s clock then it will not be stable for \( r \) cycles starting from the tick.

There are only two possibilities for coding information: changes of the signal from high to low or low to high, which are called edges, and distances between edges.

An edge is expressed in DC as:

1. for the sender

\[ [X]; [\neg X] \text{ or } [\neg X]; [X] \]

2. for the receiver

\[ [Y]; [\neg Y] \text{ or } [\neg Y]; [Y] \]

From the assumption A2, an edge of the sender, in an interval that is long enough, can introduce several edges of the receiver. However, if we consider those edges whose first signal is stable for a long time then we can have a one-to-one corresponding between edges of the sent signal and received signal (see Corollary 1). If we refer to an edge which has a long stable first signal as a long edge, we can formally define long edges by

\[ l_{Hi} \cdot lo_s \equiv (\ell \geq 2r + 1 \land [X]); (\ell \geq r + 1 \land [\neg X]), \]

\[ l_{Lo} \cdot Hi_s \equiv (\ell \geq 2r + 1 \land [\neg X]); (\ell \geq r + 1 \land [X]), \]

\[ l_{Hi} \cdot Lo_r \equiv (\ell \geq 2r + 1 \land [Y]); (\ell = 1 \land [\neg Y]), \]

\[ l_{Lo} \cdot Hi_r \equiv (\ell \geq r + 1 \land [\neg Y]); (\ell = 1 \land [Y]). \]

**Corollary 1.**

\[ l_{Hi} \cdot lo_s \Rightarrow \Diamond l_{Hi} \cdot Lo_r \land \neg (\Diamond l_{Hi} \cdot Lo_r), \]

\[ l_{Lo} \cdot Hi_s \Rightarrow \Diamond l_{Lo} \cdot Hi_r \land \neg (\Diamond l_{Lo} \cdot Hi_r). \]
This means that there is a one-to-one correspondence between long edges of the sender and the receiver, and the first long edge of the receiver is the corresponding edge. Notice that long edges of the same kind (low to high or high to low) can be distinguished from each other. The corollary says that long edges of the same kind (low to high or high to low) can be distinguished from each other. The corollary says that long edges can be used in encoding information. Actually we can have a more precise relationship between X and Y, as in the following corollaries, which will be used in proving the correctness of the Biphase Mark protocol later.

**Corollary 2.** Let \( a \geq r + 1, b \geq r + 1, c \geq r + 1 \). Then

\[
([X] \land \ell = a); ([\neg X] \land \ell = b); ([X] \land \ell = c) \Rightarrow
\]

\[
\begin{align*}
\ell & \leq 1; \left( \text{int} \land \ell = r - 1 \right); \left( \text{int} \land \left( [R \land X] \land \ell - a + r < 1 \right) \right); \\
\ell & = r; \left( \text{int} \land \left( [\neg R] \land \ell = r \right) \right); \left( \text{int} \land \left( [R \land \neg X] \land \ell - b + r < 1 \right) \right); \\
\ell & > a - r - 1; \left( \text{int} \land \left( [Y] \land \ell = 1 \right) \right); \left( \text{int} \land \left( \ell = 1 \right) \right); \\
\ell & = d; \left( \text{int} \land \left( [Y] \land \ell = 1 \right) \right); \left( \text{int} \land \left( \ell = 1 \right) \right); \\
\ell & > c - r - 1; \left( \text{int} \land \left( \ell > c - r - 1 \right) \right)
\end{align*}
\]

**Corollary 3.** For \( a \geq r + 1, b \geq r + 1, c \geq r + 1 \), for any \( d \in \mathbb{N} \) and \( c \) such that \( b + r + d \leq c + b + r - 3 \),

\[
\begin{align*}
([X] \land \ell = a); ([\neg X] \land \ell = b); ([X] \land \ell = c) \Rightarrow
\end{align*}
\]

\[
\begin{align*}
\ell & \leq 1; \left( \text{int} \land \ell = r - 1 \right); \\
\ell & = r; \left( \text{int} \land \left( [Y] \land \ell = 1 \right) \right); \\
\ell & > a - r - 1; \left( \text{int} \land \left( \ell = 1 \right) \right); \\
\ell & = d; \left( \text{int} \land \left( [Y] \land \ell = 1 \right) \right); \\
\ell & > c - r - 1; \left( \text{int} \land \left( \ell > c - r - 1 \right) \right)
\end{align*}
\]

The corollaries above are of importance in designing communication protocols. A formal proof of them in DC can be found in [5].

For a protocol, to make the algorithm concrete and deterministic, long edges for the sender are defined as follows:

\[
H_{i} \cdot L_{o_{a}} \equiv (([X] \land \ell = a); ([\neg X] \land \ell = b)) \land \text{Lo}_{Hi_{a}} \equiv (([\neg X] \land \ell = a); ([X] \land \ell = b)).
\]

In the next sections, we give a formal specification and verification of some concrete protocols as an application of our model.

### 4. Biphase Mark Protocols

In this section, a formal model for encoding and decoding rules of the Biphase Mark (BPM) protocol is provided, and it is also verified that with some restriction to the rules the BPM protocol is always correct.

The BPM protocol encodes a bit as a cell consisting of a mark subcell and a code subcell. If the signal in the mark subcell is the same as the one in the code subcell then the information carried by that cell is 0. Otherwise, if the signals are different then the information carried is 1. There is a phase reverse between two consecutive cells. It means that the signal at the beginning of the following cell is held as the negation of the signal at the end of the previous cell.

The receiver, after detecting the beginning of a cell, skips some cycles called as **sampling distance** and samples the signal. If the sampled signal is the same as the signal at the beginning of the cell, it decodes the cell as 0; otherwise it decodes the cell as 1. Figure 2 illustrates the sending and the receiving of the sequence \( w = 01 \).

We will formalize the protocol by using Discrete Duration Calculus and the denotations from the theory of formal languages (see, e.g., [4]). From now on, in order to simplify our denotations, we will identify regular languages with their regular expressions.

Let us denote in this section,

\[
H_{i} \cdot L_{o_{r}} \equiv \left( \text{int} \land \left( [Y] \land 1 \leq \ell \leq c' - 1 \right) \right); \left( \text{int} \land \left( [\neg Y] \land \ell = 1 \right) \right),
\]

\[
L_{o} \cdot H_{i_{r}} \equiv \left( \text{int} \land \left( [Y] \land 1 \leq \ell \leq c' - 1 \right) \right); \left( \text{int} \land \left( [\neg Y] \land \ell = 1 \right) \right),
\]

where \( c' \) is a natural number, \( c' > 1 \).

#### Encoding information

From now on, to distinguish with the symbols from the alphabets we use the symbols \( \oplus, (,), + \) as standard connectives for building regular expressions. Let \( \Sigma_s \) be the
alphabet

\[ \Sigma_s \equiv \left\{ \begin{array}{l}
[X], Lo_{Hi_s}, Hi_{Lo_s}; \\
([X] \land \ell \geq c), ([X] \land \ell \geq c)
\end{array} \right\}, \]

and let \( L_s \) be the set of DC formulas represented by the following regular expression over \( \Sigma_s \).

\[ L_s \equiv [-X]; Lo_{Hi_s}, Hi_{Lo_s}; ([X] \land \ell \geq c) \cup ([X] \land \ell \geq c). \]

The coding function \( f : \{0, 1\}^+ \rightarrow L_s \) is defined inductively as follows.

**Basic:**

\[ f(0) \equiv [-X]; Lo_{Hi_s}; ([X] \land \ell \geq c), \]
\[ f(1) \equiv [-X]; Hi_{Lo_s}; ([X] \land \ell \geq c). \]

**Induction:**

**F1** Let \( f(w) = D; ([X] \land \ell \geq c)(\in L_s) \), where \( D \in \Sigma_s^* \). Then,

\[ f(w0) \equiv D; Hi_{Lo_s}; ([X] \land \ell \geq c), \]
\[ f(w1) \equiv D; Lo_{Hi_s}; ([X] \land \ell \geq c). \]

**F2** Let \( f(w) = D; ([X] \land \ell \geq c). \) Then,

\[ f(w0) \equiv D; Lo_{Hi_s}; ([X] \land \ell \geq c), \]
\[ f(w1) \equiv D; Hi_{Lo_s}; ([X] \land \ell \geq c). \]

For example, by the definition of \( f \),

\[ f(01) = [-X]; Lo_{Hi_s}; Hi_{Lo_s}; ([X] \land \ell \geq c) \]

which expresses the sender’s state in Fig. 2.

**Decoding information**

Let

\[ \Sigma_r \equiv \left\{ \begin{array}{l}
\ell \leq 1, (int \land \ell = r - 1); \\
Lo_{Hi_r}, \ell = d, \\
Hi_{Lo_r}, \ell < 1,
\end{array} \right\} \]

where \( d \) is a natural number. The decoding function \( g : \mathcal{L}_r \rightarrow \{0, 1\}^+ \) is defined below, where \( \mathcal{L}_r \) is the set of DC formulas represented by the following regular expression over \( \Sigma_r \).

\[ \mathcal{L}_r \equiv \ell \leq 1; (int \land \ell = r - 1); ([Y] \land \ell \geq c') \lor \mathcal{L}_r; (\lceil Y \rceil \land \ell \geq c') \lor \mathcal{L}_r); \ell < 1. \]

First, we define a function \( g', g' : ((Lo_{Hi_r} \oplus Hi_{Lo_r}) \equiv \ell = d) \cup ([Y] \land \ell \geq c') \lor \mathcal{L}_r; (\lceil Y \rceil \land \ell \geq c') \lor \mathcal{L}_r), \ell < 1 \rightarrow \{0, 1\}^+ \) inductively as follows.

**Basic:**

1. Let \( D = Hi_{Lo_r}; \ell = d; ([Y] \land \ell \geq c') \lor \mathcal{L}_r; (\lceil Y \rceil \land \ell \geq c') \lor \mathcal{L}_r); \ell < 1 \) then \( g'(D) = 1 \)

2. Let \( D = Hi_{Lo_r}; \ell = d; ([Y] \land \ell \geq c') \lor \mathcal{L}_r; (\lceil Y \rceil \land \ell \geq c') \lor \mathcal{L}_r); \ell < 1 \) then \( g'(D) = 0 \)

3. Let \( D = Lo_{Hi_r}; \ell = d; ([Y] \land \ell \geq c') \lor \mathcal{L}_r; (\lceil Y \rceil \land \ell \geq c') \lor \mathcal{L}_r); \ell < 1 \) then \( g'(D) = 1 \)

4. Let \( D = Lo_{Hi_r}; \ell = d; ([Y] \land \ell \geq c') \lor \mathcal{L}_r; (\lceil Y \rceil \land \ell \geq c') \lor \mathcal{L}_r); \ell < 1 \) then \( g'(D) = 1 \)

**Induction:**

1. Let \( D = Lo_{Hi_r}; \ell = d; Lo_{Hi_r}; D' \) then \( g'(D) = \]

2. Let \( D = Lo_{Hi_r}; \ell = d; Hi_{Lo_r}; D' \) then \( g'(D) = \]

3. Let \( D = Hi_{Lo_r}; \ell = d; Lo_{Hi_r}; D' \) then \( g'(D) = \]

4. Let \( D = Hi_{Lo_r}; \ell = d; Hi_{Lo_r}; D' \) then \( g'(D) = \]

Let \( D = \ell \leq 1; (int \land \ell = r - 1); Lo_{Hi_r}; D' \). Then the decoding function \( g \) is defined as

\[ g(D) \equiv g'(Lo_{Hi_r}; D'). \]
For example, suppose that the sender wants to send a sequence of bits 011. It encodes the sequence as the DC formulas $f(011)$,

$$f(011) = ([\neg X] \land \ell \geq c); Lo, Hi; Hi, Lo; Hi, Lo; \neg X) \land \ell \geq c).$$

If we can prove that

$$f(011) \Rightarrow D,$$

where

$$D \triangleq \ell \leq 1; (\text{int} \land \ell = r - 1); ([\neg Y] \land \text{int}); Lo, Hi; \ell = d; Hi, Lo; \ell = d; Hi, Lo; \ell = d; ([Y] \land \ell \geq c') \land \text{int}; \ell < 1,$$

then the same sequence 011 will be received by the receiver, since according to the definition of $g$, we have

$$g(D) = 011.$$

A verification which shows that the BPM protocol is correct with some restriction to the encoding and decoding rules, i.e., to the lengths of $a$, $b$, $c$, $c'$, and $d$, is given below.

Verification

To verify that the protocol for the case $b + r \leq d \leq a + b - r - 3$, $a \geq 2r + 1$, $c \geq c' + 1$, and $c' \geq a + r + 1$ is correct, we have to prove:

1. For all $D, D' \in L_r$ such that $D$ and $D'$ are different from each other syntactically, $D \land D' = \text{false}$
2. For all $w \in \{0, 1\}^+$, there exists a formula $D \in L_r$ such that $f(w) \Rightarrow D$ and $g(D) = w$.

Proof of item 1

Let $R$ be the following regular expression over $\Sigma_r$:

$$R \triangleq Lo, Hi; \ell = d ((Lo, Hi \lor Hi, Lo); \ell = d).$$

For any $C \in R$, using the rules R1-R4 it can be proved easily by the induction on the length of the ‘word’ $C$ that any DC formulas $D, D'$,

$$(C; D) \land (C; D') \Rightarrow C; (D \land D').$$

(1)

Now, let $D, D'$ be different elements of in $L_r$ (syntactically). Then, there exists $C \in R$ such that

$$D = \ell \leq 1; (\text{int} \land \ell = r - 1); ([\neg Y] \land \text{int}); C; A; E,$n

$$D' = \ell \leq 1; (\text{int} \land \ell = r - 1); ([\neg Y] \land \text{int}); C; A'; E';$$

where $A$ and $A'$ are different symbols of $\Sigma_r$, $E, E'$ are DC formulas. By applying rules I3, R1-R4 and (1),

$$D \land D' \Rightarrow \ell \leq 1; (\text{int} \land \ell = r - 1); C; ((A; E) \land (A'; E')).$$

We have

$$(A; E) \land (A'; E'))$$

$$\Rightarrow \left\{ \begin{array}{l}
((Lo, Hi;: \text{true}) \land (Hi, Lo;: \text{true})) \lor \\
(Lo, Hi;: \text{true}) \land \\
((([Y] \land \text{int} \land \ell \geq c'); \text{true}) \lor \\
(Lo, Hi;: \text{true}) \land \\
((([Y] \land \text{int} \land \ell \geq c'); \text{true}) \lor \\
(Lo, Hi;: \text{true}) \land \\
((([Y] \land \text{int} \land \ell \geq c'); \text{true}) \lor \\
(Lo, Hi;: \text{true}) \land \\
((([Y] \land \text{int} \land \ell \geq c'); \text{true})) \end{array} \right\}$$

By the rule R3, DC rule 9, and the definitions of $Lo, Hi$, $Hi, Lo$ all the disjuncts of the disjunction are equivalent to $\text{false}$. Thus,

$$(A; E) \land (A'; E')) \Rightarrow \text{false}.$$ Consequently by the DC rule 8 (Zero),

$$D \land D' \Rightarrow \ell \leq 1; (\text{int} \land \ell = r - 1); ([\neg Y] \land \text{int}); C; \text{false} \Rightarrow \text{false}.$$ Proof of item 2

First, we give an another definition of the decoding function $g$ via the following lemma.

Lemma 1. Let $D$ be a word of the language $L_r$ such that $g(D) = w$. If $D = D'; ([\neg Y] \land \text{int} \land \ell \geq c'; \ell < 1)$ then

$$G1 \quad g(D'; Hi, Lo; \ell = d; ([\neg Y] \land \text{int} \land \ell \geq c'); \ell < 1) = w1$$

$$G2 \quad g(D'; Hi, Lo; \ell = d; ([\neg Y] \land \text{int} \land \ell \geq c'); \ell < 1) = w0.$$ If $D = D'; ([\neg Y] \land \text{int} \land \ell \geq c'); \ell < 1$, then

$$G3 \quad g(D'; Lo, Hi; \ell = d; ([\neg Y] \land \text{int} \land \ell \geq c'); \ell < 1) = w0$$

$$G4 \quad g(D'; Lo, Hi; \ell = d; ([\neg Y] \land \text{int} \land \ell \geq c'); \ell < 1) = w1.$$ Proof: The lemma is proved easily by the induction on the length of the ‘word’ $D$ using the definition of the function $g$ and is omitted here.

The proof of item 2 now is by induction on the length of $w$. The inductive assertion is that if $f(w) = B; ([X] \land \ell \geq c)$ then we can find a ‘word’ $D' \in L_r$,

$$D' = A; \ell = d; ([Y] \land \ell \geq c \land \text{int}); \ell < 1.$$
such that

\[ g(D') = w, \]
\[ f(w) \Rightarrow D', \]
\[ B; ([\neg X] \land \ell = a) \Rightarrow \]
\[ A; (\ell = d; ([Y] \land \ell \geq 1) \land \left(\text{true}; ([Y] \land \ell \geq a - r - 1)\right)); \ell < 1, \]

and if \( f(w) = B; ([\neg X] \land \ell \geq c) \) then we can find a ‘word’ \( D' \in \mathcal{L}_r \)

\[ D' = A; \ell = d; ([\neg Y] \land \ell \geq c' \land \text{int}); \ell < 1 \]

such that

\[ g(D') = w, \]
\[ f(w) \Rightarrow D', \]
\[ B; ([\neg X] \land \ell = a) \Rightarrow \]
\[ A; (\ell = d; ([\neg Y] \land \ell \geq 1) \land \left(\text{true}; ([Y] \land \ell \geq a - r - 1)\right)); \ell < 1. \]

Basic case: Let \(|w| = 1\), where \(|w|\) denotes the length of \(w\).

By the definition of the encoding function \(f\):

\[ f(1) = [\neg X]; \text{Lo}_Hi; ([X] \land \ell \geq c), \]
\[ f(0) = [\neg X]; \text{Lo}_Hi; ([\neg X] \land \ell \geq c). \]

Now, by Corollary 3,

\[ [\neg X]; \text{Lo}_Hi; ([X] \land \ell \geq c) \Rightarrow \]
\[ \ell \leq 1; (\text{int} \land \ell = r - 1); ([\neg Y] \land \text{int}); \text{Lo}_Hi; \ell = d; ([Y] \land \ell \geq c' \land \text{int}); \ell < 1. \]

Denote the conclusion of the implication by \(D\). By the definition of \(g\), we have \(g(D) = 1\). Also by Corollary 3,

\[ [\neg X]; \text{Lo}_Hi; ([X] \land \ell = a) \Rightarrow \]
\[ \ell \leq 1; (\text{int} \land \ell = r - 1); ([\neg Y] \land \text{int}); \text{Lo}_Hi;\]
\[ (\ell = d; [Y] \land \left(\text{true}; [Y] \land \ell \geq a - r - 1\right)) ; \ell < 1. \]

Similarly,

\[ f(0) \Rightarrow \]
\[ \ell \leq 1; (\text{int} \land \ell = r - 1); ([\neg Y] \land \text{int}); \text{Lo}_Hi; \ell = d; ([\neg Y] \land \ell \geq c' \land \text{int}); \ell < 1, \]

and

\[ g(\ell \leq 1; (\text{int} \land \ell = r - 1); ([\neg Y] \land \text{int}); \text{Lo}_Hi;\]
\[ \ell = d; ([\neg Y] \land \ell \geq c' \land \text{int}); \ell \leq 1, \]
\[ [\neg X]; \text{Lo}_Hi; ([\neg X] \land \ell = a) \Rightarrow \]
\[ \ell \leq 1; (\text{int} \land \ell = r - 1); ([\neg Y] \land \text{int}); \text{Lo}_Hi;\]
\[ (\ell = d; [Y] \land \left(\text{true}; [Y] \land \ell \geq a - r - 1\right)) ; \ell < 1. \]

Hence, the basic case is verified.

Induction step: According to the definition of the function \(f\), there are two cases to consider. We verify for the case \(F1\) only. The other is similar and is left to the reader.

Let \( f(w) = B; ([X] \land \ell \geq c) \in \mathcal{L}_s\), where \(B \in \Sigma_s^*\).

Then, by the definition \(F1\) of \(f\),

\[ f(w0) \equiv B; \text{Hi}_Lo; ([\neg X] \land \ell \geq c), \]
\[ f(w1) \equiv B; \text{Hi}_Lo; ([X] \land \ell \geq c). \]

By the inductive hypothesis, we can find a ‘word’ \(D' = A; \ell = d; ([Y] \land \ell \geq c' \land \text{int}); \ell < 1 \in \mathcal{L}_r\) such that

\[ f(w) \Rightarrow D', \]
\[ g(D') = w, \] and
\[ B; ([X] \land \ell = a) \Rightarrow \]
\[ A; (\ell = d; ([Y] \land \ell \geq 1) \land \left(\text{true}; ([Y] \land \ell \geq a - r - 1)\right)); \ell < 1. \]

Thus,

\[ f(w1) = \]
\[ B; ([X] \land \ell = a); ([\neg X] \land \ell = b); ([X] \land \ell \geq c) \]
\[ \Rightarrow A; (\ell = d; ([Y] \land \ell \geq 1) \land \left(\text{true}; ([Y] \land \ell \geq a - r - 1)\right)) ; \ell < 1 \]
\[ \text{int} \land \ell < r + 1 \land ([Y] \lor [Y]; \text{true}) \land ([\neg Y] \land \ell \geq 1) ; \ell < 1 \]
\[ \text{int} \land (Y \land \ell \geq c - r - 1) ; \ell < 1 \]
\[ \Rightarrow A; (\ell = d; ([Y] \land \ell \geq 1) \land \left(\text{true}; ([Y] \land \ell \geq a - r - 1)\right)) ; \ell < 1 \]
\[ \text{int} \land \ell \geq 1 \]
\[ \text{int} \land \ell \geq 1 \]
\[ \text{int} \land (\neg O((\neg Y) \land \ell \geq r); (\neg Y \land \ell \geq 1)) ; (2) \]
\[ \ell < 1. \] (3)
Here we have used the implication \( \ell < 1; \ell = r \Rightarrow \ell < r + 1 \land ([Y] \lor [Y]^*; [-Y]; \text{true}) \). On the other hand, by Corollary 3 (and also Corollary 1),

\[
f(w_1) = B; [X] \land \ell = a; ([\neg Y] \land \ell = b); ([X] \land \ell \geq c)
\]

\[
\Rightarrow \text{true}; \ell \leq r; \left( \begin{array}{c}
\text{int} \land [Y] \land \ell \geq a - r - 1 \\
\text{int} \land [\neg Y] \land \ell = 1
\end{array} \right)
\]

\[
\left( \begin{array}{c}
\text{int} \land [Y] \land \ell \geq c' \\
\ell = d
\end{array} \right)
\]

\[
\left( \begin{array}{c}
\ell < 1,
\end{array} \right)
\]

By the rules I3, R3, R4 and because \( a \geq 2r + 1 \), it follows from (2), (3) that

\[
f(w_1) \Rightarrow A; \ell = d; Hi_{Lo_a}; \ell = d; ([Y] \land \ell \geq c') \land \ell < 1.
\]

By G1 of Lemma 1 and the inductive hypothesis

\[
g(A; \ell = d; Hi_{Lo_a}; \ell = d; ([Y] \land \ell \geq c'); \ell < 1) = w_1.
\]

Let

\[
A' = A; \ell = d; Hi_{Lo_a},
\]

\[
B' = B; Hi_{Lo_a}.
\]

By the same arguments (replacing \( c \) with \( a \)), it can be verified also that

\[
B'; ([X] \land \ell = a)
\]

\[
\Rightarrow A'; \left( \begin{array}{c}
\text{int} \land [Y] \land \ell \geq 1 \\
\ell = d; [Y] \land \ell \geq a - r - 1
\end{array} \right) ; \ell < 1.
\]

Similarly, we can verify the statement for the word \( w_0 \).

**Reasoning about clock rates of the sender and the reader**

We have verified the correctness of the Biphase Mark Protocols for the case \( b + r \leq d \leq a + b - r - 3, b \geq r + 1, a \geq 2r + 1 \), \( c \geq c' + 1 \). The values of \( a, b, c, d, r \) are numbers of the receiver’s clock cycles. However, the values \( a \) and \( b \) should be given according to the sender’s clock. Since the sender’s clock rate and the receiver’s clock rate are different, we cannot have the same values \( a \) and \( b \) according to the receiver’s clock. Thus, the values of \( a \) and \( b \) should be such that when transformed into the values according to the receiver’s clock, the above mentioned conditions are satisfied. Depending on specific values of \( a, b, c, d, r \) the ratio of the rates of the sender’s and the receiver’s clock for which the protocol is correct can be determined easily. Let us consider some examples.

Let \( d = 10, r = 3 \). Then, for \( 4 \leq b \leq 7, c \geq a, a + b \geq 16 \), the protocol is correct. Let \( a = 13, b = 5 \) according to the sender’s clock. Then, if the ratio of the rates of the receiver’s clock and the sender’s clock is between \( 4/5 \) and \( 7/5 \) then the above inequalities are satisfied and the protocol is still correct.

Let \( d = 10, r = 2 \). Then, for \( 3 \leq b \leq 8, a + b \geq 15 \), the protocol is correct. Let \( a = 13, b = 5 \) according to the sender’s clock. Then, if the ratio of the rates of the receiver’s clock and the sender’s clock is between \( 3/5 \) and \( 8/5 \) then the above inequalities are satisfied and the protocol is still correct.

**5 Audio Protocol**

In this section, we use our approach to specify and verify the correctness of the audio control protocol. The protocol uses the well-known Manchester encoding of bit strings. Each bit is encoded as a cell consisting \( 4Q \) cycles of the receiver clock. Bit 1 is encoded as an upgoing edge at the middle of the cell, and bit 0 is encoded as a downgoing edge at the middle of the cell. If the same bit is sent twice in a row then an additional edge is required, which is placed exactly in between the corresponding two cells (see Fig. 3).

The receiver scans only for upgoing edges (\( i_{Lo}Hi \)) using the Philips decoding. It has to decode the signal without seeing the downgoing edges. This is always possible except that a message ending with “10” cannot be distinguished from the same message ending with “1”. To solve the problem, we require that all messages either end with “00” or have an odd length. In addition, all messages should begin with “1” for the receiver to detect the beginning of the messages.

Let us denote in this section,

\[
Hi_{Lo_r} \equiv ([Y] \land \text{int} \land r \leq \ell); ([\neg Y] \land \ell = 1),
\]

\[
Lo_{Hi_r} \equiv ([\neg Y] \land \text{int} \land r \leq \ell); ([Y] \land \ell = 1).
\]

Let \( \mathcal{L}_a \) be defined as in the Biphase Mark protocols. Now, we can model the Audio Protocol as follows.

**Manchester encoding**

\[ f : \{0, 1\}^+ \rightarrow \mathcal{L}_a \] in this case is a partial function only.

Its domain is defined by a regular expression:

\[ 1(00 \oplus 01 \oplus 10 \oplus 11)^* \oplus 1(0 \oplus 1)^*00. \]

Let \( f' \) be a homomorphism defined by

\[
f'(1) \equiv Lo_{Hi_a} \]

\[
f'(0) \equiv Hi_{Lo_a} \]

\[
f'(w0) \equiv f'(w); f'(0) \]

\[
f'(w1) \equiv f'(w); f'(1). \]
Then the Manchester encoding function $f$ is defined as

$$f(w) \equiv [\neg X]; f'(w): ([\neg X] \land \ell \geq c).$$

**Philips decoding**

Let us denote:

- $FourH \equiv ((\neg \Diamond Lo.Hi_r); Lo.Hi_r) \land 3Q \leq \ell < 5Q$
- $SixH \equiv ((\neg \Diamond Lo.Hi_r); Lo.Hi_r) \land 5Q \leq \ell < 7Q$
- $EightH \equiv ((\neg \Diamond Lo.Hi_r); Lo.Hi_r) \land 7Q \leq \ell$
- $NineH \equiv ((\neg \Diamond Lo.Hi_r) \land 9Q \leq \ell$.

Let

$$\Sigma_r \equiv \{ FourH, SixH, EightH, NineH, \}$$

and $\mathcal{L}_r$ be represented by the following regular expression over $\Sigma_r$:

$$\mathcal{L}_r \equiv (\neg \Diamond Lo.Hi_r); Lo.Hi_r; (FourH \oplus SixH \oplus EightH)^* \oplus NineH.$$

Then, the decoding function $g$, $g : \mathcal{L}_r \rightarrow \{0,1\}^+$, is defined below. First, the function $g'(\neg \Diamond Lo.Hi_r); Lo.Hi_r; (FourH \oplus SixH \oplus EightH)^* \rightarrow \{0,1\}^+$ is defined inductively as follows.

**Basic case:**

$$g'(\neg \Diamond Lo.Hi_r); Lo.Hi_r) \equiv 1.$$

**Induction step:** Assume that $g'(D)$ has been defined and $g'(D) = w0$.

- $g'(D; FourH) \equiv w00$
- $g'(D; SixH) \equiv w01$.

Assume that $g'(D)$ has been defined and $g'(D) = w1$.

- $g'(D; FourH) \equiv w11$
- $g'(D; SixH) \equiv w100$
- $g'(D; EightH) \equiv w101$.

Now, the function $g''$ is defined as:

$$g''(D; NineH) \equiv g'(D).$$

Let $Just : \{0,1\}^* \rightarrow \{0,1\}^*$ be a homomorphism defined by:

1. if $w = x1$ and $|w|$ is odd then $Just(w) = w$,
2. if $w = x1$ and $|w|$ is even then $Just(w) = w0$,
3. if $w = x0$ and $|w|$ is odd then $Just(w) = w$,
4. if $w = x0$ and $|w|$ is even then
   - $x = y0$ then $Just(w) = w$,
   - $x = y1$ then $Just(w) = w0$.

The Philips decoding function $g : \mathcal{L}_r \rightarrow \{0,1\}^*$ is defined by

$$g(D) \equiv Just(g''(D)).$$

**Verification**

We have to find the values of $a, b, c, r$, and prove that for those $a, b, c, r$

1. For all $D, D' \in \mathcal{L}_r$, $D \neq D'$ implies that $D \land D' \Rightarrow false$,
2. For all $w \in dom(f)$, there is a formula $D \in \mathcal{L}_r$ such that $f(w) \Rightarrow D$ and $g(D) = w$.

The proof of items 1 and 2 are similar to the proof for the Biphas Mark protocols. We give here a rough sketch of our proof only.

**Lemma 2.** Let $D = A; B \in \mathcal{L}_r$, $D' = A; B' \in \mathcal{L}_r$, where $A, B, B' \in \Sigma_r$. Then $D \land D' \Rightarrow A; (B \land B')$.

The lemma can be proved easily by induction on the length of the word $A$ using the rule R4.

The first item follows directly from Lemma 2 with the note that for any $C, C' \in \Sigma_r$, $C \neq C'$ implies that $C \land C' \Rightarrow false$.

Based on Corollaries 2, 3, we then can prove easily by induction on the length $|w|$ of $w \in \{0,1\}^*$ that if $a = b, a \geq 2r + 1, 4a - r - 1 \geq 7Q, 4a + r + 1 \leq 9Q$, then

- $w = x1$ implies that there exists $D' \in \mathcal{L}_r$ such that $f(w) \Rightarrow D'$ and $g''(D') = x1$,
- $w = x10$ implies that there exists $D' \in \mathcal{L}_r$ such that $f(w) \Rightarrow D'$ and $g''(D') = x1$,
- $w = x00$ implies that there exists $D' \in \mathcal{L}_r$ such that $f(w) \Rightarrow D'$ and $g''(D') = x0$. 

11
Item 2 then follows immediately from these properties and the definition of the function $g$.

**Reasoning about clock rates of the sender and the reader**

In the proof of the correctness of the protocol, we have assumed that the reference time is according to the receiver’s clock. However, in practice, $a$ should be implemented according to the sender’s clock. Thus if we take $a$ to be $2Q$ according to the sender’s clock, then it will be that $a = 2Q + 2\delta$ according to the receiver’s clock. Thus, to guarantee that the protocol is correct, it should be the case that

\[
2Q + 2\delta \geq 2r + 1, \\
8Q + 8\delta - r - 1 \geq 7Q, \\
8Q + 8\delta + r + 1 \leq 9Q.
\]

Thus, if we can ensure that

\[
2r + 1 - 2\delta \leq 2Q, \\
|\delta| \leq (Q - r - 1)/8
\]

then the protocol is correct. For instance, let $r = 3$, $Q = 5$, $a = 2Q$. In this case, $|\delta| \leq 1/8$, which means that the ratio of the rates of the sender’s and the receiver’s clock should be between $39/40$ and $41/40$.

In practice, $Q$ is very big ($\geq 200$, see e.g. [1]), and we have much looser constraints on the rates of the clock of the sender and the reader.

**6. Conclusion**

We have presented our approach to the specification and verification of real-time hybrid systems using Duration Calculus. By introducing the formula $\text{int}$ into the calculus, we are able to specified both discrete states and continuous states. Apart from this, we are also able to reason about the different clock rates of different components in the systems, which is necessary in distributed systems. By trying our approach in the Biphase Mark protocol and Audio Control protocol, we have shown that we can specify the protocol in a natural and intuitive way. In our model of communication protocols, we can verify the correctness in a familiar way using just the induction techniques on the length of strings of a formal languages and some obvious rules of the calculus. Since formal proofs are often tedious work, our syntactical approach enables the use of a mechanical prover in the verification of the real-time systems.

**References**


