On the Characterization of Until as a Fixed Point Under Clocked Semantics

Dana Fisman[1,2]

[1] Hebrew University
[2] IBM Haifa Research Lab

Abstract. Modern hardware designs are typically based on multiple clocks. While a singly-clocked hardware design is easily described in standard temporal logics, describing a multiply-clocked design is cumbersome. Thus, it is desirable to have an easier way to formulate properties related to clocks in a temporal logic. In [6] a relatively simple solution built on top of the traditional LTL semantics was suggested and adopted by the IEEE standard temporal logic PSL. The suggested semantics was examined relative to a list of design goals, and it was shown that it answered all requirements except for preserving the least fixed point characterization of the until operator under multiple clocks. In this work we show that with a minor addition to the semantics of [6] this requirement is met as well.

1 Introduction

Synchronous hardware designs are based on a notion of discrete time, in which the flip-flop (or latch) takes the system from the current state to the next state. A flip-flop or latch is a memory element, which passes on some function of its inputs to its outputs, but only when its clock input is active. The signal that causes the flip-flop (or latch) to transition is termed the *clock*. In a singly-clocked hardware design, temporal operators in logics such as LTL [16,17] are interpreted with respect to the clock, so that the following LTL formula:

$$\mathsf{G}(p \rightarrow \mathsf{X}q) \tag{1}$$

can be interpreted as "globally, if p then *at the next clock cycle*, q". Mapping between a state of a model for the temporal logic and a clock cycle of hardware can then be dealt with by the tool which builds a model from the source code (written in some hardware description language).

Modern hardware designs, however, are typically based on multiple clocks. In such a design, for instance, some flip-flops may be clocked with *clka*, while others are clocked with *clkb*. In this case, the mapping between temporal operators and clock cycles cannot be done automatically; rather, the formula itself must contain some indication of which clock to use. Thus, it is desirable to have an easier way to formulate properties related to clocks in a temporal logic. For example, the

K. Yorav (Ed.): HVC 2007, LNCS 4899, pp. 19–33, 2008.

linear temporal logic LTL can be extended with a clock operator, denoted @, so that the formula

$$(\mathsf{G}(p \rightarrow \mathsf{X}q))@clka \tag{2}$$

stating that "globally, if p during a cycle of $clka$, then at the next cycle of $clka$, q" will be equivalent to the LTL formula

$$\mathsf{G}((clka \wedge p) \rightarrow \mathsf{X}(\neg clka \ \mathsf{W} \ (clka \wedge q))) \tag{3}$$

In [6] a relatively simple solution built on top of the traditional LTL semantics is given. The underlying idea of this solution is that the only role of the clock operator should be to define a projection of the path onto those states where the clock "ticks", and it is its own dual. Actually, referring to a projection of the path is not precisely correct, as we allow access to states in between consecutive states of a projection in the event of a clock switch. However, the word "projection" conveys the intuitive function of the clock operator in the case that the formula is singly-clocked. Achieving this introduces a problem for paths on which the clock never ticks. This problem is solved in [6] by introducing a propositional strength operator that extends the semantics from non-empty paths to empty paths in the same way that the strong next operator [14,12] extends the semantics from infinite paths to finite paths.

The solution of [6] has been adopted by the standard temporal logic PSL [11,5] and extended to account for regular expression which are an important part of PSL. The definition of the clock operator in the standard temporal logic SVA [10] which is based on regular expressions and does not include LTL operators agrees with that of PSL.[1]

The logic given in [6], is measured against a list of design goals. It is shown that all design goals are met, but that the least fixed point characterization of until is not preserved when multiple clocks are involved. The characterization of until as a fixed point is not merely a theoretical issue — it has practical aspects as some tools (e.g. the ones built upon the automata theoretic approach [19]) rely on it. In this work we show that with a minor addition to the semantics of [6] the until operator preserves its least fixed point characterization (and the semantics preserves the other design goals as well).

The addition suggested herein can be thought of as *alignment* operators, such as those of [9], that takes you to the closest clock tick, when the current cycle is not a clock tick. Note that the next operators takes you to the *second* clock tick when the current cycle is not a clock tick. This is ok since on the projected path, the second clock tick is the second letter — exactly the place where the next operator will take you in standard LTL. The alignment operator comes in two version, weak and strong, in order to deal with the possibility the clock may stop ticking. The strong alignment operator demands the clock to tick at least once

[1] PSL stands for *Property Specification Language* and is defined IEEE Std 1850$^{\text{TM}}$-2005 [11]. Its formal syntax and semantics are defined in Annex B of this document. SVA stands for *SystemVerilog Assertions* and is defined in Annex E of IEEE Std 1800$^{\text{TM}}$-2005 [10].

more, while the weak alignment operator makes no such requirement. On singly-clocked formulas there is no need for alignment operators, since a subformula is always evaluated on a clock tick (the clock operator causes the evaluation to consider the projected path, and all other operators keep the evaluation on this path). On multiply-clocked formulas, however, on the event of a clock switch a subformula may be evaluated on a cycle which is not a clock tick. The alignment operators, in this case, takes the evaluation to the closest relevant tick.

The remainder of the paper is organized as follows. In Section 2 we give the semantics of the logic, and explain the difference with [6]. In Section 3 we provide some simple observations on the semantics of the logic. In Section 4 we prove that the least fixed point characterization of until is preserved, as well as the other design goals of [6]. In Section 5 we conclude.

2 The Definition of LTL$^{@}$

The semantics of [6] is defined with respect to a *clock context*. Evaluation of a formula of the form $\varphi @ clk$ then sets clk to be the clock context. An unclocked formula can be seen as a formula working in clock context *true*. The temporal operators advance the evaluation according to the clock context. For example, the formula $\varphi \mathsf{U} \psi$ requires that there would be a clock tick of the clock context where ψ holds and on every preceding tick of the clock φ holds.

The problem of finite paths introduced by the fact that the clock context may stop ticking is solved in [6] by defining two versions of the next operator as is done in LTL [14, pp.272-273] and the linear time μ-calculus [12]. The formula $\mathsf{X}!\varphi$ of LTL holds on a word w if the length of w is at least two and φ holds on the second letter of w whereas the weak version holds also if the length of w is less than two. In LTL augmented with the clock operator as per [6] we get that $\mathsf{X}!\varphi$ holds on a word w if the length of w projected onto the cycles where the clock context ticks is at least two (i.e. there are at least two clock ticks in w) and φ holds on the second tick whereas the weak version holds also if the are less than two ticks. The problem of empty paths introduced by the fact that the clock context may not tick at all is solved in [6] by providing a propositional strength operator. Given a proposition p, both $p!$ and p are formulas of the logic of [6], where the strong version $p!$ holds if on every non-empty word w, the first letter of w satisfies p; and the weak version holds also on the empty word.

In this work we solve the problem of finite and empty paths by augmenting the next operator with an *exponent*. That is, the next operators comes with a non-negative integer m, so that $\mathsf{X}!^m\varphi$ holds if φ holds on the $(m{+}1)$-th future tick of the clock context (where future is understood to include the present). If the clock context is *true* the formula $\mathsf{X}!^m\varphi$ requires φ to hold on the $(m{+}1)$-th letter of the given word. Similarly, $\mathsf{X}^m\varphi$ holds if φ holds on the $m{+}1$-th future tick of the clock if there are $m{+}1$ future ticks. If the clock context is *true* the formula $\mathsf{X}^m\varphi$ holds if φ holds on the $(m{+}1)$-th letter, or there are less than $m{+}1$ letters in the given word. The operators obtained by instantiating m with zero

(i.e. $\mathsf{X!}^0$ and X^0) can be seen as alignment operators, similar to those of [9].[2] The formulas $\mathsf{X!}^0\varphi$ and $\mathsf{X}^0\varphi$ advance the evaluation to the closest clock tick, when the current cycle is not a clock tick, and evaluate φ there. If the current cycle is a clock tick φ is evaluated at the current cycle. Thus, if the clock context is *true* (or the formula is unclocked) φ is evaluated at the current letter. As expected the strong version requires that there would be at least one future clock tick while the next version holds also there are no clock ticks. If the clock context is *true* the strong version requires that the given word would not be empty whereas the weak version holds also if it is empty.

Remark 1. The alignment operators $\mathsf{X!}^0$ and X^0 move to the nearest concurrent or future clock tick. It might be practically useful to include also alignment operators that are strictly future. That is, while $\mathsf{X!}^0$ and X^0 do not advance when the current cycle is a clock tick, the strictly future alignment operators will advance to the next clock tick, when the current cycle is a clock tick (and to the closest clock tick when the current cycle is not a clock tick). These can be defined as syntactic sugaring by means of the existing operators, using T as the clock context.

2.1 Syntax

We define the syntax of LTL$^{@}$ as follows, where we use the term *Boolean expression* to refer to any application of the standard Boolean operators to atomic propositions.

Definition 1 (Formulas of LTL$^{@}$)

- *If b is a Boolean expression, then $b!$ and b are LTL$^{@}$ formulas.*
- *If clk is a Boolean expression, m is a non-negative integer, and φ and ψ are LTL$^{@}$ formulas, then the following are LTL$^{@}$ formulas:*
  - $\neg\varphi$ $\quad$ • $\varphi \wedge \psi$ $\quad$ • $X!^m\varphi$ $\quad$ • $\varphi U \psi$ $\quad$ • $\varphi @ clk$

Additional operators are derived from the basic operators defined above:

- $\varphi \vee \psi \stackrel{\text{def}}{=} \neg(\neg\varphi \wedge \neg\psi)$
- $\varphi \rightarrow \psi \stackrel{\text{def}}{=} \neg\varphi \vee \psi$
- $\mathsf{X}^m\varphi \stackrel{\text{def}}{=} \neg\mathsf{X!}^m\neg\varphi$
- $\mathsf{F}\varphi \stackrel{\text{def}}{=} \mathsf{T}\,\mathsf{U}\varphi$
- $\mathsf{G}\varphi \stackrel{\text{def}}{=} \neg\mathsf{F}\neg\varphi$
- $\varphi\mathsf{W}\psi \stackrel{\text{def}}{=} (\varphi\mathsf{U}\psi) \vee \mathsf{G}\varphi$
- $\mathsf{X}\varphi \stackrel{\text{def}}{=} \mathsf{X}^1\varphi$
- $\mathsf{X!}\varphi \stackrel{\text{def}}{=} \mathsf{X!}^1\varphi$

where T is a Boolean expression that holds on every letter. In the sequel, we also use F, which is a Boolean expression that does not hold for any letter.

We refer to the subset of LTL$^{@}$ consisting of the formulas that have no clock operator, by LTL. This subset is a slight generalization of the standard definition

[2] The operators $\mathsf{X!}^0$ and X^0 resemble the operator `s_align @(c)` and `w_align @(c)` of the assertion language ECBV [9]. The definition of the alignment operators here and there do not resemble since the language ECBV is defined by means of computations of an alternating automaton. The obtained semantics, however, does.

of LTL (as defined in [17]) – it consists of two version of Boolean expressions as well as the generalized version of the next operator. The important thing, however, is that it agrees with the standard semantics on the common operators (on non-empty paths, as the standard semantics is defined only over non-empty paths).

2.2 Semantics

We denote a letter by ℓ, and an empty, finite, or infinite word by u, v, or w. The *concatenation* of u and v is denoted by uv. If u is infinite, then $uv = u$. The empty word is denoted by ϵ, so that $w\epsilon = \epsilon w = w$. We denote the *length* of word v as $|v|$. The empty word ϵ has length 0, a finite word $v = (\ell_0\ell_1 \cdots \ell_n)$ has length $n+1$, and an infinite word has length ∞. We use i, j, and k to denote non-negative integers. For $i < |v|$ we use v^i to denote the $(i+1)^{st}$ letter of v (since counting of letters starts at zero). We denote by $v^{i\cdot\cdot}$ the suffix of v starting at v^i.

The semantics of LTL$^{\mathbb{C}}$ is defined inductively with respect to a word (which may be infinite, finite or empty) over the alphabet $\Sigma = 2^P$ where P is a given non-empty set of atomic propositions. We identify Boolean expression over P as elements of $B = 2^{2^P}$ (as they convey subset of possible valuations (assignments) to the set of propositions in P). For a Boolean expression $b \in B$ and a letter $\ell \in \Sigma$ we define the Boolean satisfaction relation $\Vdash$ by $\ell \Vdash b$ iff $\ell \in b$.

We first present the semantics of LTL over infinite, finite, and empty words (*unclocked semantics*). We then present the semantics of LTL$^{\mathbb{C}}$ over infinite, finite, and empty words (*clocked semantics*). In Corollary 1 given in Section 4.2 we show that the unclocked semantics can be obtained from the clocked semantics by setting the clock context to T.

Unclocked Semantics. We now present semantics for LTL. The semantics is defined with respect to an infinite, finite, or empty word. The notation $w \models \varphi$ means that formula φ holds along the word w. The semantics is defined as follows, where b denotes a Boolean expression, φ and ψ denote formulas, and m, j and k denote natural numbers (i.e., non-negative integers).

- $w \models b \iff |w| = 0$ or $w^0 \Vdash b$
- $w \models b! \iff |w| > 0$ and $w^0 \Vdash b$
- $w \models \neg\varphi \iff w \not\models \varphi$
- $w \models \varphi \wedge \psi \iff w \models \varphi$ and $w \models \psi$
- $w \models \mathsf{X}!^m\varphi \iff |w| > m$ and $w^{m\cdot\cdot} \models \varphi$
- $w \models \varphi \mathsf{U} \psi \iff \exists k < |w|$ such that $w^{k\cdot\cdot} \models \psi$, and $\forall j < k$, $w^{j\cdot\cdot} \models \varphi$

Clocked Semantics. We define the semantics of an LTL$^{\mathbb{C}}$ formula with respect to an infinite, finite, or empty word w and a context c, where c is a Boolean expression over P. We say that a finite word w *is a clock tick of* clock c if c holds at the last letter of w and does not hold at any previous letter of w. Formally,

Definition 2 (clock ticks)

- *We say that finite word w is a* clock tick *of c iff $|w| > 0$ and $w^{|w|-1} \models c$ and for every natural number $i < |w| - 1$, $w^i \not\models c$.*
- *For $m > 0$, we say that finite word w is m* clock ticks *of c iff there exists m words $w_1, w_2, \ldots, w_m$ such that $w = w_1 w_2 \ldots w_m$ and for every $1 \leq i \leq m$ the word w_i is a clock tick of c.*

The notation $w \models^c \varphi$ means that formula φ holds along the word w in the context of clock c. The semantics is defined as follows, where b, c and c_1 denote Boolean expressions, φ and ψ denote formulas, and m, j and k denote natural numbers (i.e., non-negative integers).

- $w \models^c b \iff$ if $\exists k < |w|$ such that $w^{0..k}$ is a clock tick of c then $w^k \models b$
- $w \models^c b! \iff \exists k < |w|$ such that $w^{0..k}$ is a clock tick of c and $w^k \models b$
- $w \models^c \neg\varphi \iff w \not\models^c \varphi$
- $w \models^c \varphi \wedge \psi \iff w \models^c \varphi$ and $w \models^c \psi$
- $w \models^c \mathsf{X}!^m\varphi \iff \exists j < |w|$ s.t. $w^{0..j}$ is $m+1$ clock ticks of c and $w^{j..} \models^c \varphi$
- $w \models^c \varphi \mathsf{U} \psi \iff \exists k < |w|$ such that $w^k \models c$ and $w^{k..} \models^c \psi$ and $\forall j < k$ such that $w^j \models c$, $w^{j..} \models^c \varphi$
- $w \models^c \varphi @ c_1 \iff w \models^{c_1} \varphi$

3 Observations on the Semantics of LTL@

The following section provides some simple observation regarding the weak/strong next operators and their exponents. First we provide the direct semantics of the weak next operator, which was given as a syntactic sugaring of the strong next operator in Section 2.1.

Claim 1 (Weak next operator). *Let w be a word over Σ. Let c be a Boolean expression in B, m a non-negative integer and φ an* LTL@ *formula. Then*

$$w \models^c \mathsf{X}^m\varphi \iff \text{if } \exists j < |w| \text{ s.t. } w^{0..j} \text{ is } m+1 \text{ clock ticks of } c \text{ then } w^{j..} \models^c \varphi$$

The following claim states that the weak/strong Boolean expressions can be stated in terms of the weak/strong next operator by setting m to zero. Note that this does not mean that the X^0 and $\mathsf{X}!^0$ are redundant in the presence of weak and strong Boolean expressions. They are needed to provide an easy way to get to the closest tick (when the current cycle is not a clock tick) for general formulas.

Claim 2 (Weak/strong Boolean expressions). *Let w be a word over Σ. Let c and b be Boolean expressions in B. Then*

1. $w \models^c b! \iff w \models^c \mathsf{X}!^0 b$
2. $w \models^c b \iff w \models^c \mathsf{X}^0 b$

The following claim shows that the standard strong and weak next operators, as defined in [6], are obtained by setting m to one in $\mathsf{X!}^m$ and X^m, respectively.

Claim 3 (Weak/strong simple next operators). *Let w be a word over Σ. Let c be a Boolean expression in B and φ an* LTL$^{@}$ *formula. Then*

1. $w \models^c \mathsf{X!}\varphi \iff \exists j < k < |w|$ *such that* $w^{0..j}$ *is a clock tick of* c *and* $w^{j+1..k}$ *is a clock tick of* c *and* $w^{k..} \models^c \varphi$
2. $w \models^c \mathsf{X}\varphi \iff$ *if* $\exists j < k < |w|$ *such that* $w^{0..j}$ *is a clock tick of* c *and* $w^{j+1..k}$ *is a clock tick of* c *then* $w^{k..} \models^c \varphi$

The following claim states that $\mathsf{X!}^m$ can be obtained by m iterations of $\mathsf{X!}$ and similarly for the weak version.

Claim 4 (Power characterization). *Let w be a word over Σ. Let c be a Boolean expression in B, m a integer and φ an* LTL$^{@}$ *formula.*

1. $w \models^c \mathsf{X!}^m\varphi \iff w \models^c \underbrace{\mathsf{X!X!}\ldots\mathsf{X!}}_{m \text{ times}}\varphi$
2. $w \models^c \mathsf{X}^m\varphi \iff w \models^c \underbrace{\mathsf{XX}\ldots\mathsf{X}}_{m \text{ times}}\varphi$

The following claim states that the next operators are additive. That is, composition of $\mathsf{X!}^m$ (resp., X^m) operators corresponds to addition in the exponents (even if one or both exponents is zero).

Claim 5. *Let m and n be non-negative integers and φ and* LTL$^{@}$ *formula . Then*

1. $\mathsf{X!}^m\mathsf{X!}^n\varphi \equiv \mathsf{X!}^{m+n}\varphi$
2. $\mathsf{X}^m\mathsf{X}^n\varphi \equiv \mathsf{X}^{m+n}\varphi$

The proofs of these claims are given in the full version of the paper.

4 Meeting the Goals

In this section, we show that the logic LTL$^{@}$ satisfies all the goals of [6], as well as preserving the least and greatest fixed point characterization of the strong and weak until operators, respectively. We make use of the following notations.

The *projection* of a word w onto clock c, denoted $w|_c$, is the word obtained from w after leaving only the letters which satisfy c. For example, if $w = \ell_0\ell_1\ell_2\ldots\ell_{10}$ and $\ell_0 \not\models c$, $\ell_1 \not\models c$, $\ell_2 \not\models c$, $\ell_3 \models c$, $\ell_4 \not\models c$, $\ell_5 \not\models c$, $\ell_6 \models c$, $\ell_7 \models c$, $\ell_8 \not\models c$, $\ell_9 \not\models c$ and $\ell_{10} \not\models c$ then $w|_c = \ell_3\ell_6\ell_7$.

Let φ be an LTL formula. We use $[\![\varphi]\!]$ to denote the set of all words satisfying φ. That is $[\![\varphi]\!] = \{w \mid w \models \varphi\}$. Let φ be an LTL$^{@}$ formula. We use $[\![\varphi]\!]_c$ to denote the set of all words satisfying φ under clock context c. That is, $[\![\varphi]\!]_c = \{w \mid w \models^c \varphi\}$. We say that two LTL formulas φ and ψ are *unclocked equivalent* ($\varphi \equiv \psi$) if $[\![\varphi]\!] = [\![\psi]\!]$. We say that two LTL$^{@}$ formulas φ and ψ are *clocked equivalent* ($\varphi \overset{@}{\equiv} \psi$) if $[\![\varphi]\!]_c = [\![\psi]\!]_c$ for every clock contexts c.

4.1 The Until Fixed Point Characterization

In standard LTL, interpreted over infinite words, $[\![\varphi \ \mathsf{U} \ \psi]\!]$ and $[\![\varphi \ \mathsf{W} \ \psi]\!]$ are the least and greatest (resp.) fixed points of the functional E defined as follows [15,3]:[3]

$$E(S) = [\![\psi \vee (\varphi \wedge \underline{\mathsf{X}} S)]\!] \tag{4}$$

where $\underline{\mathsf{X}}$ is the strengthless version of the $\mathsf{X}!$ and X operators that it used when LTL is interpreted solely on infinite words. When LTL is interpreted over finite words as well then $[\![\varphi \ \mathsf{U} \ \psi]\!]$ and $[\![\varphi \ \mathsf{W} \ \psi]\!]$ are the least and greatest fixed point of the functionals E^+ and E^- obtained from E by replacing $\underline{\mathsf{X}}$ with $\mathsf{X}!$ and X, respectively.

$$E^+(S) = [\![\psi \vee (\varphi \wedge \mathsf{X}! S)]\!] \tag{5}$$

$$E^-(S) = [\![\psi \vee (\varphi \wedge \mathsf{X} S)]\!] \tag{6}$$

This characterization does not hold for LTL$^{@}$. Consider the strong until operator. We have that $\psi \vee (\varphi \wedge \mathsf{X}! S)$ may be satisfied weakly if ψ is a weak formula, e.g. if ψ equals $\mathsf{X}p$ for some proposition p and the clock never ticks. Whereas $\varphi \ \mathsf{U} \ (\mathsf{X}p)$ demands (among other things) that there would be at least one clock tick. For this reason, [6] proposed the functional F_c^+ defined as follows:

$$F_c^+(S) = [\![(\mathrm{T}! \wedge \psi) \vee (\varphi \wedge \mathsf{X}! S)]\!]_c \tag{7}$$

where T! is the strong Boolean expression T asserting that there is a current cycle (and T holds on it). This characterization holds for singly-clocked formulas but not for multiply-clocked formulas. The following counter examples shows that the characterization breaks when multiple clocks are involved. Let p, q and $clkq$ be atomic propositions, and let $\psi = q@clkq$. Consider a word w such that $w^0 \models clkq \wedge q$ and for all $i > 0$, $w^i \not\models clkq \wedge q$, and $w^0 \not\models c$. Then $w \models^c \psi$ hence $w \models^c (\mathrm{T}! \wedge \psi) \vee (p \wedge \mathsf{X}!(p \ \mathsf{U} \ \psi))$. However, since $w^0 \not\models c$, and there is no state other than w^0 where $clkq \wedge q$ holds, $w \not\models^c (p \ \mathsf{U} \ \psi)$.

The following claim states that under the semantics given here, the strong until operator can be defined as a least fixed point of the following functional

$$G_c^+(S) = [\![\mathsf{X}!^0(\psi \vee (\varphi \wedge \mathsf{X}! S))]\!]_c \tag{8}$$

(even in the presence of multiple clocks). Since by definition $w \models^{\mathrm{T}} \mathsf{X}!^0\varphi \iff |w| > 0$ and $w \models^{\mathrm{T}} \varphi$ this can be seen as a generalization of the standard characterization: The standard characterization assumes paths are non-empty and works with no clock context, or equivalently with T as the clock context - thus the $\mathsf{X}!^0$ operator can be removed. The obtained functional $G_{\mathrm{T}}^+(S) = [\![\psi \vee (\varphi \wedge \mathsf{X}! S)]\!]_{\mathrm{T}}$ is then the standard characterization. If we restrict also to infinite paths, the strength of the $\mathsf{X}!$ operator can be taken away as well.

[3] Earlier works (e.g. [2]) provided fixed point representation of the logic CTL by showing that it can be encoded in the propositional μ-calculus [13]. See [7] for the fixed-point characterization of the logic CTL*(that subsumes both LTL and CTL).

Theorem 1. *Let φ and ψ be* LTL$^{@}$ *formulas. Then $[\![\varphi \,\mathsf{U}\, \psi]\!]_c$ is a least fixed point of the functional $G_c^+(S) = [\![X!^0\,(\psi \vee (\varphi \wedge X!\, S))]\!]_c$.*

Proof: It is easy to see that the functional G_c^+ is monotonic and thus by Tarski-Knaster theorem [18] it has a least fixed point. First we show that $[\![\varphi \,\mathsf{U}\, \psi]\!]_c$ is a fixed point of G_c^+ then we show that it is the least fixed point.

- Let φ and ψ be LTL$^{@}$ formulas. Let w be a word over Σ, and c a Boolean expression. We show that $[\![\varphi \,\mathsf{U}\, \psi]\!]_c$ is a fixed point of G_c^+ by proving

$$\varphi \,\mathsf{U}\, \psi \overset{@}{\equiv} \mathsf{X!}^0\,(\psi \vee (\varphi \wedge \mathsf{X!}\,(\varphi \,\mathsf{U}\, \psi))).$$

  $w \models^c \mathsf{X!}^0\,(\psi \vee (\varphi \wedge \mathsf{X!}\,(\varphi \,\mathsf{U}\, \psi)))$
  $\iff \exists\ j_0 < |w|$ s.t. $w^{0..j_0}$ is a clock tick of c and $w^{j_0..} \models^c (\psi \vee (\varphi \wedge \mathsf{X!}\,(\varphi \,\mathsf{U}\, \psi)))$
  $\iff \exists j_0 < |w|$ such that $w^{0..j_0}$ is a clock tick of c and either $w^{j_0..} \models^c \psi$ or $w^{j_0..} \models^c (\varphi \wedge \mathsf{X!}\,(\varphi \,\mathsf{U}\, \psi)))$
  $\iff \exists j_0 < |w|$ such that $w^{0..j_0}$ is a clock tick of c and either $w^{j_0..} \models^c \psi$ or $(w^{j_0..} \models^c \varphi$ and $\exists j_1 < |w|$ such that $j_1 > j_0$ and $w^{j_0+1..j_1}$ is a clock tick of c and $w^{j_1..} \models^c (\varphi \,\mathsf{U}\, \psi))$
  $\iff \exists j_0 < |w|$ such that $w^{0..j_0}$ is a clock tick of c and either $w^{j_0..} \models^c \psi$ or $(w^{j_0..} \models^c \varphi$ and $\exists j_1 < |w|$ such that $j_1 > j_0$ and $w^{j_0+1..j_1}$ is a clock tick of c and $\exists k < |w^{j_1..}|$ such that $w^{j_1+k} \models c$ and $w^{j_1+k..} \models^c \psi$ and for every $j < k$ such that $w^{j_1+j} \models c$, $w^{j_1+j..} \models^c \varphi)$
  $\iff \exists k < |w|$ s.t. $w^k \models c$ and $w^{k..} \models^c \psi$ and $\forall j < k$ s.t. $w^j \models c$, $w^{j..} \models^c \varphi$
  $\iff w \models^c \varphi \,\mathsf{U}\, \psi$
- We have shown that $[\![\varphi \mathsf{U} \psi]\!]_c$ is a fixed point of G_c^+. We now show that $[\![\varphi \mathsf{U} \psi]\!]_c$ is the least fixed point. That is, given S is an arbitrary fixed point of G_c^+ we show that $[\![\varphi \mathsf{U} \psi]\!]_c \subseteq S$. Let S be a fixed point of G_c^+. That is, S is a set of empty, finite, or infinite words such that $w \in S$ iff $w \models^c \mathsf{X!}^0(\psi \vee (\varphi \wedge \mathsf{X!}\, S)$ where $w \models^c S$ means $w \in S$. We have to show that for any word w, we have that $w \models^c \varphi \mathsf{U} \psi$ implies $w \in S$.
  Let w be such that $w \models^c \varphi \mathsf{U} \psi$. Then there exists $k < |w|$ s.t. $w^k \models c$ and $w^{k..} \models^c \psi$ and for every $j < k$ s.t. $w^j \models c$ we have $w^{j..} \models^c \varphi$. From the fact that $w^{k..} \models^c \psi$ and $w^k \models c$ we get that $w^{k..} \models^c \mathsf{X!}^0 \psi$ and thus $w^{k..} \in S$. Let $0 \leq j_0 < j_1 < \ldots < j_k < k$ be the set of all j's in w such $w^j \models c$. Consider first j_k. We have that $w^{j_k} \models c$, $w^{j_k..} \models^c \varphi$ and $w^{k..} \in S$. Thus $w^{j_k..} \models^c \mathsf{X!}^0(\varphi \wedge \mathsf{X!} S)$ and so $w^{j_k..} \in S$. By induction we get that for each such j_i we have that $w^{j_i..} \in S$. In particular $w^{j_0..} \in S$. Thus $w^{j_0..} \models^c \mathsf{X!}^0(\psi \vee (\varphi \wedge \mathsf{X!} S)$. And since j_0 is the closest tick of c starting at 0 we get that $w^{0..} \models^c \mathsf{X!}^0(\psi \vee (\varphi \wedge \mathsf{X!} S)$ as well. Thus, we have shown that $w \in S$. □

We now show that the weak until operator is the greatest fixed point of the functional G^- below obtained from G^+ by weakening the next operators.

Theorem 2. *Let φ and ψ be* LTL$^{@}$ *formulas. Then $[\![\varphi \,\mathsf{W}\, \psi]\!]_c$ is a greatest fixed point of the functional $G_c^-(S) = [\![\mathsf{X}^0\,(\psi \vee (\varphi \wedge \mathsf{X}S))]\!]_c$.*

Proof: It is easy to see that the functional G_c^- is monotonic and thus by Tarski-Knaster theorem [18] it has a greatest fixed point. First we show that $[\![\varphi \,\mathsf{W}\, \psi]\!]_c$ is a fixed point of G_c^- then we show that it is the greatest fixed point. In the following we make use of the direct clocked semantics of W as given by [8, Lemma 4.9].

$$w \models^{c} \varphi \,\mathsf{W}\, \psi \text{ iff for all } 0 \leq k < |w| \text{ such that } w^k \Vdash c \text{ and } w^{k\cdot\cdot} \not\models^{@} \varphi,$$
$$\text{there exists } 0 \leq j \leq k \text{ such that } w^j \Vdash c \text{ and } w^{j\cdot\cdot} \models^{c} \psi.$$

– Let φ and ψ be LTL$^{@}$ formulas. Let w be a word over Σ, and c a Boolean expression. We show that $[\![\varphi \,\mathsf{W}\, \psi]\!]_c$ is a fixed point of G_c^- by proving

$$\varphi \,\mathsf{W}\, \psi \overset{@}{\equiv} \mathsf{X}^0\,(\psi \vee (\varphi \wedge \mathsf{X}\,(\varphi \,\mathsf{W}\, \psi))).$$

$w \models^{c} \mathsf{X}^0\,(\psi \vee (\varphi \wedge \mathsf{X}\,(\varphi \,\mathsf{W}\, \psi)))$
$\iff$ if there exists $j \geq 0$ such that $w^{0..j_0}$ is a clock tick of c then $w^{j_0\cdot\cdot} \models^{c} (\psi \vee (\varphi \wedge \mathsf{X}\,(\varphi \,\mathsf{W}\, \psi)))$
$\iff$ if there exists $j_0 < |w|$ such that $w^{0..j_0}$ is a clock tick of c then either $w^{j_0\cdot\cdot} \models^{c} \psi$ or $w^{j_0\cdot\cdot} \models^{c} (\varphi \wedge \mathsf{X}\,(\varphi \,\mathsf{W}\, \psi)))$
$\iff$ if there exists $j_0 < |w|$ such that $w^{0..j_0}$ is a clock tick of c then either $w^{j_0\cdot\cdot} \models^{c} \psi$ or ($w^{j_0\cdot\cdot} \models^{c} \varphi$ and if there exists $j_1 < |w|$ such that $j_1 > j_0$ and $w^{j_0+1..j_1}$ is a clock tick of c then $w^{j_1\cdot\cdot} \models^{c} (\varphi \,\mathsf{W}\, \psi)$)
$\iff$ if there exists $j_0 < |w|$ such that $w^{0..j_0}$ is a clock tick of c then either $w^{j_0\cdot\cdot} \models^{c} \psi$ or ($w^{j_0\cdot\cdot} \models^{c} \varphi$ and if there exists $j_1 < |w|$ such that $j_1 > j_0$ and $w^{j_0+1..j_1}$ is a clock tick of c then for all $0 \leq k < |w^{j_1\cdot\cdot}|$ such that $w^{j_1+k} \Vdash c$ and $w^{j_1+k\cdot\cdot} \not\models^{@} \varphi$, there exists $0 \leq j \leq k$ such that $w^{j_1+j} \Vdash c$ and $w^{j_1+j\cdot\cdot} \models^{c} \psi$)
$\iff$ for all $0 \leq k < |w|$ such that $w^k \Vdash c$ and $w^{k\cdot\cdot} \not\models^{@} \varphi$, there exists $0 \leq j \leq k$ such that $w^j \Vdash c$ and $w^{j\cdot\cdot} \models^{c} \psi$
$\iff$ $w \models^{c} \varphi \,\mathsf{W}\, \psi$

– We have shown that $[\![\varphi \mathsf{W} \psi]\!]_c$ is a fixed point of G_c^-. We now show that $[\![\varphi \mathsf{W} \psi]\!]_c$ is the greatest fixed point. That is, given S is an arbitrary fixed point of G_c^- we show that $[\![\varphi \mathsf{W} \psi]\!]_c \supseteq S$. Let S be a fixed point of G_c^-. That is, S is a set of empty, finite, or infinite words such that $w \in S$ iff $w \models^{c} \mathsf{X}^0(\psi \vee (\varphi \wedge \mathsf{X}S)$ where $w \models^{c} S$ means $w \in S$. We have to show that for any word w, we have that $w \in S$ implies $w \models^{c} \varphi \mathsf{W} \psi$.

Let w be such that $w \in S$. Then $w \models^{c} \mathsf{X}^0(\psi \vee (\varphi \wedge \mathsf{X}S)$. Assume towards contradiction that $w \not\models^{@} \varphi \mathsf{W} \psi$. Let $J = \{j_0, j_1, \ldots\}$ be the set of all j's such that $w^j \Vdash c$ with $0 \leq j_0 < j_1 < \ldots < |w|$. Note that $|J|$ may be empty, finite or infinite. Then, $w \not\models^{@} \varphi \mathsf{W} \psi$ implies there exists a tick point j_k such that $w^{k\cdot\cdot} \not\models^{@} \varphi$ and for every tick point $j_i \leq j_k$ we have that $w^{j_i\cdot\cdot} \not\models^{@} \psi$. Thus

$w^{j_k \cdot\cdot} \notin S$ (since $w^{j_k \cdot\cdot} \not\models^{c} \mathsf{X}^0\psi$ and $w^{j_k \cdot\cdot} \not\models^{c} \mathsf{X}^0\varphi$). Therefore, $w^{j_{k-1} \cdot\cdot} \notin S$ (since $w^{j_{k-1} \cdot\cdot} \not\models^{c} \mathsf{X}^0\psi$ and $w^{j_{k-1} \cdot\cdot} \not\models^{c} \mathsf{X}^0\mathsf{X}S$). For the same reason $w^{j_{k-2} \cdot\cdot} \notin S$. By induction we can show that for any $j_i \leq j_k$ we have $w^{j_i \cdot\cdot} \notin S$. In particular $w^{j_0 \cdot\cdot} \notin S$ and so $w = w^{0 \cdot\cdot} \notin S$ as well, contradicting our assumption. □

4.2 The Other Goals

Below we state that the semantics given here preserve the goals met in [6]. The work in [6] provides detailed explanations and motivations of the goals, as well as relation to other works. Here we provide a succinct motivation to the less obvious goals. The proofs are a slight modification of those of [6] and are given in the full version of the paper.

1. When singly-clocked, the semantics should be that of the *projection view*.
   Motivation:
   When only a single clock is involved we would like that a clocked formula $\varphi @ clk$ hold on a word w if and only if the unclocked formula φ holds on a word $w|_c$ (i.e. on the word obtained from w by projecting onto those states where clk holds).⌟

   Proposition 1. *For any* LTL *formula* φ*, a Boolean expression* c *and an infinite, finite, or empty word* w*, the following holds:*

   $$w \models^{c} \varphi \quad \textit{if and only if} \quad w|_c \models \varphi$$

   The following is an immediate consequence of this.

   Corollary 1. *for an* LTL *formula* φ*, and a word* w*.*

   $$w \models^{\mathrm{T}} \varphi \quad \textit{if and only if} \quad w \models \varphi$$

2. Clocks should not accumulate.
   Motivation:
   In many hardware designs, large chunks of the design work on some main clock, while small pieces work on a secondary clock. Rather than require the user to specify a clock for each subformula, we would like to allow clocking of an entire formula on a main clock, and pieces of it on a secondary clock, in such a way that the outer clock (which is applied to the entire formula) does not affect the inner clock (which is applied to one or more sub-formulas). That is, we want a nested clock operator to have the effect of "changing the projection", rather than further projecting the projected word.⌟

   Proposition 2. *For any* LTL$^{@}$ *formula* φ *and Boolean expressions* c_1 *and* c_2 *the following holds:*

   $$\varphi @ c_1 @ c_2 \overset{@}{\equiv} \varphi @ c_1$$

3. The clock operator should be its own dual.
   Motivation:
   Previous definitions of clock operators [1,4] introduces two kids of clocks, a weak clock and a strong clock (one is the dual of the other). Each of the definitions had drawbacks as we elaborate on items 4 and 5 below.⌟

   Proposition 3. *For any* LTL$^{@}$ *formula* φ *and Boolean expression* b *the following holds:*

   $$(\neg\varphi)@b \overset{@}{\equiv} \neg(\varphi@b)$$

4. For any atomic propositions p and q, there should be a clocked version of $(\mathsf{F}p) \wedge (\mathsf{G}q)$ that is meaningful on paths with a finite number of clock ticks.
   Motivation:
   In Sugar2.0 [4], a strongly clocked formula requires the clock to "tick long enough to ensure that the formula holds", while a weakly clocked formula allows it to stop ticking before then. Thus, for instance, the formula $(\mathsf{F}p)@clk!$ (which is strongly clocked) requires there to be enough ticks of clk so that p eventually holds, whereas the formula $(\mathsf{F}p)@clk$ (which is weakly clocked) allows the case where p never occurs, if it "is the fault of the clock", i.e., if the clock ticks a finite number of times. For the dual formulas we get that $(\mathsf{G}q)@clk!$ holds if the clock ticks an infinite number of times and q holds at every tick, while $(\mathsf{G}q)@clk$ holds if q holds at every tick, no matter how many there are. A disadvantage of this semantics is that the formula $(\mathsf{F}p) \wedge (\mathsf{G}q)$ cannot be satisfactorily clocked for a finite word, because $((\mathsf{F}p) \wedge (\mathsf{G}q))@clk!$ does not hold on any finite word, while $((\mathsf{F}p) \wedge (\mathsf{G}q))@clk$ makes no requirement on p on such a word.⌟

   Under the semantics given here we get that $((\mathsf{F}p) \wedge (\mathsf{G}q))@c$, holds if p holds for some state and q holds for all states on the projected word, which is indeed the intuitive desired semantics.

5. For any atomic proposition p, if $(\mathsf{F}p)@clk$ holds on a word, it should hold on any extension of that word.
   Motivation:
   In ForSpec [1], a strongly clocked formula requires only that the clock tick *at least once*, after which the only role of the clock is to define the projection of the word onto those states where the clock ticks. A weakly clocked formula, on the other hand, holds if the clock never ticks; if it does tick, then the role of the clock is the same as for a strongly clocked formula. Thus, the only difference between strong and weak clocks in ForSpec is on paths whose projection is empty. This leads to the strange situation that a liveness formula may hold on some word w, but not on an extension of that word, ww'. For instance, if p is an atomic proposition, then $(\mathsf{F}p)@clk$ holds if there are no ticks of clk, but does not hold if there is just one tick, at which p does not hold.⌟

Proposition 4. *For Boolean expressions b, clk and c, a finite word w, and an infinite or finite word w', the following holds:*

$$w \models^{c} (\mathsf{F}b)@clk \implies ww' \models^{c} (\mathsf{F}b)@clk$$

6. For any clock c, two equivalent LTL formulas should remain equivalent when clocked with c.
   Proposition 5. *For LTL formulas φ and ψ, and a Boolean expression c, the following holds:*

$$\varphi \equiv \psi \quad \implies \quad \varphi@c \overset{@}{\equiv} \psi@c$$

7. Substituting subformula ψ for an equivalent subformula h should not change the truth value of the original formula.
   Proposition 6. *If ψ is a subformula of φ, and $\psi' \overset{@}{\equiv} \psi$, then the following holds:*

$$\varphi \overset{@}{\equiv} \varphi[\psi \leftarrow \psi']$$

   where $\varphi[\psi \leftarrow \psi']$ denotes the formula obtained from φ by replacing subformula ψ with ψ'.
8. For every word, the truth value of LTL$^{@}$ Formula (2) given in the introduction should be the same as the truth value of LTL Formula (3) given in the introduction.
   Proposition 7. *For every word w,*
   $w \overset{\mathrm{T}}{\models} (\mathsf{G}(p \rightarrow \mathsf{X}q))@clka \iff w \models \mathsf{G}((clka \wedge p) \rightarrow \mathsf{X}(\neg clka \ \mathsf{W}\ (clka \wedge q)))$

4.3 Rewrite Rules

In [6] it was shown that the clock operator does not add expressive power. In fact there are rewrite rules that given an LTL$^{@}$ formula φ return an equivalent LTL formula. The rewrite rules form a recursive procedure $\mathcal{T}^{clk}()$, whose application starting with $clk = \mathrm{T}$ results in an LTL formula with the same truth value in context T. The rewrite rules are given below. Note that by Claim 4 it suffices to provide rewrite rules for $\mathsf{X}!^{0}$ and $\mathsf{X}!$ instead of $\mathsf{X}!^{m}$.

- $\mathcal{T}^{clk}(b) = (\neg clk \ \mathsf{W}\ (clk \wedge b))$
- $\mathcal{T}^{clk}(b!) = (\neg clk \ \mathsf{U}\ (clk \wedge b))$
- $\mathcal{T}^{clk}(\neg f) = \neg \mathcal{T}^{clk}(\varphi)$
- $\mathcal{T}^{clk}(\varphi \wedge \psi) = \mathcal{T}^{clk}(\varphi) \wedge \mathcal{T}^{clk}(\psi)$
- $\mathcal{T}^{clk}(\mathsf{X}!^{0} f) = (\neg clk \ \mathsf{U}\ (clk \wedge \mathcal{T}^{clk}(\varphi)))$
- $\mathcal{T}^{clk}(\mathsf{X}!\varphi) = (\neg clk \ \mathsf{U}\ (clk \wedge \mathsf{X}!(\neg clk \ \mathsf{U}\ (clk \wedge \mathcal{T}^{clk}(\varphi)))))$
- $\mathcal{T}^{clk}(\varphi \ \mathsf{U}\ \psi) = (clk \rightarrow \mathcal{T}^{clk}(\varphi)) \ \mathsf{U}\ (clk \wedge \mathcal{T}^{clk}(\psi))$
- $\mathcal{T}^{clk}(\varphi@clk_1) = \mathcal{T}^{clk_1}(\varphi)$

Proposition 8. *Let φ be any LTL$^{@}$ formula, c a Boolean expression, and w a word.*

$$w \models^{c} \varphi \quad \textit{if and only if} \quad w \models \mathcal{T}^{c}(\varphi)$$

The proof of this proposition as well as some additional rewrite rules are given in the full version of the paper.

5 Conclusions

In [6] a relatively simple definition of LTL augmented with a clock operator was given. The augmented logic is suitable for specifying properties in multiply-clocks designs [5, Chapter 14] and was adopted by the IEEE standard PSL. In this definition, the only role of the clock operator is to define a projection of the word, and it is its own dual. This definition was shown to answer a list of design goals. However it does not preserve the least fixed point characterization of the until operator. The characterization of until as a least fixed point is not merely a theoretical issue — it has practical aspects as some tools depend on it.

In this work we fix this problem with a minor addition to the semantics of [6]. The addition introduces an *exponent* to the next operator. The key of this solution is that by taking the zero exponent we get the operators $\mathsf{X}!^0$ and X^0 which can be thought of as alignment operators, such as the ones in ECBV [9], taking us to the closest clock tick, if the current cycle is not a clock tick.

The suggested semantics can be seen as a way to abstract the word when multiple clocks are involved. The clock operator @ defines the current clock context, so that the temporal operators move according to this context. For example, $(\varphi \mathsf{U} \psi)$ demands that ψ hold on some future tick of the context clock, and φ holds on all ticks preceding the tick where ψ holds. The alignment operators $\mathsf{X}!^0$ and X^0 allow you to move to the closest tick of a clock, which is needed in the event of a clock switch.

The resulting semantics meets all the design goals listed in [6] and preserves the least and greatest fixed point characterization of the strong and weak until operators, respectively.

Acknowledgements

I would like to thank Katoen Joost-Pieter for asking a question at HVC that prompted Theorem 2, which was not a part of the originally submitted version of this paper. I would like to thank Doron Bustan, Cindy Eisner, John Havlicek and an anonymous referee for their important comments on an early draft of this paper. I would like to thank Orna Kupferman for helping me in tracking some of the references.

References

1. Armoni, R., Fix, L., Flaisher, A., Gerth, R., Ginsburg, B., Kanza, T., Landver, A., Mador-Haim, S., Singerman, E., Tiemeyer, A., Vardi, M.Y., Zbar, Y.: The ForSpec temporal logic: A new temporal property-specification language. In: Katoen, J.-P., Stevens, P. (eds.) ETAPS 2002 and TACAS 2002. LNCS, vol. 2280, Springer, Heidelberg (2002)
2. Clarke, E., Emerson, E.: Characterizing correctness properties of parallel programs as fixpoints. In: de Bakker, J.W., van Leeuwen, J. (eds.) Automata, Languages and Programming. LNCS, vol. 85, Springer, Heidelberg (1980)

3. Dam, M.: Temporal logic, automata and classical theories - an introduction. Lecture Notes for the 6th European Summer School on Logic, Language and Information (1994)
4. Eisner, C., Fisman, D.: Sugar 2.0 proposal presented to the Accellera Formal Verification Technical Committee (March 2002), http://www.haifa.il.ibm.com/verification/sugar/Sugar_2.0_Accellera.ps.
5. Eisner, C., Fisman, D.: A practical introduction to PSL. Springer, Heidelberg (2006)
6. Eisner, C., Fisman, D., Havlicek, J., McIsaac, A., Van Campenhout, D.: The definition of a temporal clock operator. In: Baeten, J.C.M., Lenstra, J.K., Parrow, J., Woeginger, G.J. (eds.) ICALP 2003. LNCS, vol. 2719, Springer, Heidelberg (2003)
7. Emerson, E.A.: Model checking and the Mu-calculus. In: Descriptive Complexity and Finite Models. DIMACS Series in Discrete Mathematics, vol. 31, pp. 185–214. American Mathematical Society, Providence, RI (1997)
8. Havlicek, J., Fisman, D., Eisner, C.: Basic results on the semantics of Accellera PSL 1.1 foundation language. Technical Report 2004.02 Accellera, (May 2004)
9. Havlicek, J., Levi, N., Miller, H., Shultz, K.: Extended CBV statement semantics, partial proposal presented to the Accellera Formal Verification Technical Committee (April 2002), http://www.eda.org/vfv/hm/att-0772/01-ecbv_statement_semantics.ps.gz
10. Annex E of IEEE Standard for SystemVerilog Unified Hardware Design, Specification, and Verification Language. IEEE Std 1800^{TM} (2005)
11. IEEE Standard for Property Specification Language (PSL). IEEE Std 1850^{TM} (2005)
12. Kaivola, R.: A simple decision method for the linear-time mu-calculus (1995)
13. Kozen, D.: Results on the propositional mu-calculus. Theoretical Computer Science 27(3), 333–354 (1983)
14. Manna, Z., Pnueli, A.: Temporal Verification of Reactive Systems: Specification. Springer, New York (1992)
15. Manna, Z., Wolper, P.: Synthesis of communicating processes from temporal logic specifications. ACM Trans. Program. Lang. Syst. 6(1), 68–93 (1984)
16. Pnueli, A.: The temporal logic of programs. In: Proc. 18^{th} Annual IEEE Symposium on Foundations of Computer Science, pp. 46–57 (1977)
17. Pnueli, A.: In transition from global to modular temporal reasoning about programs. In: Apt, K. (ed.) Logics and Models of Concurrent Systems. NATO Advanced Summer Institute, vol. F-13, pp. 123–144. Springer, Heidelberg (1985)
18. Tarski, A.: A lattice-theoretical fixpoint theorem and its applications. Pacific J. Math. 5, 285–309 (1955)
19. Vardi, M.Y.: An automata-theoretic approach to linear temporal logic. In: Banff Higher Order Workshop, pp. 238–266 (1995)