A Syntax and Semantics of Sequence and Property Formulas

A.1 Syntax and Semantics

The SVA language definition consists of four layers: *Boolean*, *sequence*, *property* and *statement*. Since Chisel already supports Boolean expressions and we only supports assertion statements now, we focus here on the sequence and property layers defined in CHA. The syntax of a sequence formula ψ over a set $\mathcal{P}$ of Bool expressions or signals is defined as below:

$$\psi := \texttt{ap(}u\texttt{)} \mid \psi \ \texttt{\#\#\#}\,(0)\ \psi \mid \psi\,\texttt{\#\#\#}\,(1)\psi \mid \psi^*(0) \mid \psi^*(1,-1) \mid \psi|\psi$$

where $u \in \mathcal{P}$ is a Bool expression. Here we use `ap(`u`)` instead of u to explicitly indicate that it is a sequence formula. Since the symbols `##`, `[]` and `$` used in SVA are reserved in Scala, we use `###`(0) and `###`(1) to represent sequence fusion and sequence concatenation.

Due to the same reason,

we use $^*(), |$ and -1 to denote the consecutive repetition, sequence disjunction and $ in SVA. Therefore, $\psi^*(1,-1)$ means the consecutive repetition of s with infinite range.

Below we list the detailed counterpart sequence operators of CHA compared to those in SVA.

Name	boolean sequence	sequence fusion	sequence concatenation	sequence disjunction	zero repetition	intervals
SVA	u	##0	##1	or	$[*0]$	$[*1:\$]$
CHA	ap(u)	###(0)	###(1)	|	$^*(0)$	$^*(1,-1)$

The syntax of our property formulas ϕ is similar to that of SVA:

$$\phi := \psi \mid \phi \texttt{ || } \phi \mid \phi \texttt{ \&\& } \phi \mid \psi \texttt{ |-> } \phi \mid \texttt{!}\phi \mid \text{ G } \phi \mid \text{ F } \phi \mid \text{ X } \phi \mid \phi \text{ U } \phi$$

where ψ is a sequence formula.

Below we give our counterpart temporal property operators to those of SVA.

Name	suffix implication	property negation	property conjunction	property disjunction	nexttime property	always property	s_eventually property	until property
SVA	|->	not	and	or	nexttime	always	s_eventually	until
CHA	|->	!	&&	||	X	G	F	U

Similarly to SVA, CHA also supports syntactic sugars. For example, we use $^*(i,j)$ to represent sequence repetition with a finite range (and infinite range when $j=-1$). Similarly, `###`(i,j) corresponds to the syntactic sugar `##`$[i:j]$ in SVA.

A.2 Formal Semantics

In order to define the semantics of CHA, we will treat a Chisel design as a *Kripke structure.* A Kripke structure is a tuple $\mathcal{K} = \langle Q, I, V, R\rangle$, where V is a finite set of Boolean variables, $Q = 2^V$ is a set of states, $I \subseteq Q$ is the set of initial states, and $R \subseteq Q \times Q$ is the transition relation. We will fix a Kripke structure $\mathcal{K} = \langle Q, I, V, R\rangle$ thoughout the paper. For a sequence π and integer i, j, we denote by π^i the $(i+1)$-th element of π, and by $\pi^{i,j}$ the slice of π from π^i to π^j. If π is infinite , we denote by $\pi^{i\cdots}$ the suffix of π starting from π^i. Besides, the length of a finite sequence π is denoted by $|\pi|$. We denote by $\mathbf{Paths}^\omega(s) \subseteq Q^\omega$ the set of infinite paths starting from the state s such that $\pi \in \mathbf{Paths}^\omega(s)$ iff $\pi^0 = s$ and for every $i \geq 0$, $(\pi^i, \pi^{i+1}) \in R$. Similarly, we can define finite path. We say $\pi \in Q^\omega$ is a computation trace of $\mathcal{K}$ iff $\pi \in \mathbf{Paths}^\omega(s)$ for some $s \in I$.

Tight satisfaction is a two-way relation between a finite path π and a sequence s, denoted $\pi \models s$.

$$
\begin{aligned}
\pi \models \texttt{ap(}u\texttt{)} &\iff |\pi| = 1 \wedge \pi^0 \models \texttt{ap(}u\texttt{)} \\
\pi \models \psi_1 \ \texttt{\#\#\#}\ (0)\ \psi_2 &\iff \exists i, (\pi^{0,i} \models \psi_1 \wedge \pi^{i,|\pi|} \models \psi_2) \\
\pi \models \psi_1 \ \texttt{\#\#\#}\ (1)\ \psi_2 &\iff \exists i, (\pi^{0,i} \models \psi_1 \wedge \pi^{i+1,|\pi|} \models \psi_2) \\
\pi \models \psi^*(0) &\iff |\pi| = 0 \\
\pi \models \psi^*(1,-1) &\iff \exists i_0, \ldots, i_j, (i_0 = 0 \wedge i_j = |\pi| + 1 \wedge \\
&\qquad\qquad \forall 0 \leq k \leq j-1, \pi^{i_k, i_{k+1}-1} \models \psi) \\
\pi \models \psi_1 | \psi_2 &\iff \pi \models \psi_1 \vee \pi \models \psi_2
\end{aligned}
$$

Satisfaction is a two-way relation between a infinite path π and a property ϕ, denoted $\pi \vDash \phi$.

$$
\begin{aligned}
\pi \vDash \psi &\iff \exists i, \pi^{0,i} \models \psi \\
\pi \vDash \psi \ \texttt{|->}\ \phi &\iff \forall i \geq 1, (\pi^{0,i} \models \psi \rightarrow \pi^{i\cdots} \vDash \phi) \\
\pi \vDash !\phi &\iff \pi \nvDash \phi \\
\pi \vDash \mathrm{G}\ \phi &\iff \forall i, \pi^{i\cdot\cdot} \vDash \phi \\
\pi \vDash \mathrm{F}\ \phi &\iff \exists i, \pi^{i\cdot\cdot} \vDash \phi \\
\pi \vDash \mathrm{X}\ \phi &\iff \pi^{1\cdot\cdot} \vDash \phi \\
\pi \vDash \phi_1\ \mathrm{U}\ \phi_2 &\iff \exists j, \forall i < j, (\pi^{i\cdot\cdot} \vDash \phi_1 \wedge \pi^{j\cdot\cdot} \vDash \phi_2) \\
\pi \vDash \phi_1\ \texttt{||}\ \phi_2 &\iff \pi \vDash \phi_1 \wedge \pi \vDash \phi_2 \\
\pi \vDash \phi_1\ \texttt{\&\&}\ \phi_2 &\iff \pi \vDash \phi_1 \vee \pi \vDash \phi_2
\end{aligned}
$$

A model satisfies ϕ iff each of its infinite computation traces satisfies ϕ.