Contents

1 Objectives of this lecture 1

2 A simple 2-way intersection 2

3 The Context Machine 2

4 The Invariant 3
   4.1 A Safe state .................................................. 4
   4.2 Strengthening the Invariant .............................. 4
   4.3 Other Invariants ........................................... 4
   4.4 Some Theorems ............................................. 5

5 The ChangeLight Event 5
   5.1 Sequencing .................................................. 5

6 A General Multi-Way Intersection 7
   6.1 The New Safety Invariant .............................. 7
   6.2 A Traffic Light Controller ............................. 8

7 A General Multi-Way Intersection Parametric Controller 13

1 Objectives of this lecture

• To explore the use of a state invariant to ensure safety.

• To explore the specification of some simple traffic light controllers for a simple traffic light controlled intersection.
• To expand safety to a general intersection.
• To model a traffic light controller for a general intersection

2 A simple 2-way intersection

Consider traffic lights at the intersection of two roads, one running North-South and the other East-West. There are four sets of lights, each capable of showing Red, Green and Amber, placed at North, East, South and West positions. The North and South lights are always identical, as are the East and West lights. There are no right-turn lights. Lights should change in the sequence: Red → Green → Amber → Red → ...

We wish to specify a traffic light controller that ensures safety and the correct sequencing.

3 The Context Machine

We will introduce a context machine containing the enumerated sets DIRECTION and LIGHT. We will also specify a constant (function) OTHERDIR that maps each direction to the other direction. Context machines must be used in Event B to define sets and constants.

CONTEXT TrafficLights_ctx
SETS LIGHTS DIRECTION
CONSTANTS Red Green Amber NorthSouth EastWest OTHERDIR
AXIOMS
axm1: LIGHTS = \{Red, Green, Amber\}
axm2: Red \neq Green
axm3: Red \neq Amber
axm4: Green \neq Amber
axm5: DIRECTION = \{NorthSouth, EastWest\}
axm6: NorthSouth \neq EastWest
axm7: OTHERDIR \in DIRECTION \rightarrow DIRECTION
axm8: OTHERDIR(NorthSouth) = EastWest
axm9: OTHERDIR(EastWest) = NorthSouth
THEOREMS
\[ \text{thm1: } \forall d \in \text{DIRECTION} \]
\[ \Rightarrow \]
\[ \text{OTHERDIR(OTHERDIR}(d)) = d \]

\text{END}

The above can be expressed more simply as

\textbf{AXIOMS}

\textbf{axiom:} partition(LIGHTS, \{Red\}, \{Green\}, \{Amber\})

\textbf{axiom:} partition(DIRECTION, \{NorthSouth\}, \{EastWest\})

\textbf{The Simple TwoWay Machine}

The \textbf{TwoWay} machine

1. sees the context machine and has a state of one variable, \textit{lights}, which is a total function from \textit{DIRECTION} to \textit{LIGHT};

2. has a single event \textit{ChangeLight} to change the lights;

3. ensures that lights change in the sequence \textit{Red}, \textit{Green}, \textit{Amber}, \ldots;

4. maintains a safe intersection;

\section{4 The Invariant}

1. We wish to formulate an invariant that will ensure safety.

2. Whenever the state is unsafe, the invariant must be \textit{false}.

\[ \neg(\text{safe}) \Rightarrow \neg(\text{invariant}) \]

(1)

3. Conversely, whenever the invariant is \textit{true}, the state should be safe.

\[ \text{invariant} \Rightarrow \text{safe} \]

(2)

4. Of course, \[ \text{[1]} \text{and} \text{[2]} \text{are equivalent; one is the contrapositive of the other: } P \Rightarrow Q \equiv \neg Q \Rightarrow \neg P \]

5. If we have the \textit{weakest} invariant, then whenever the state is safe, the invariant will be \textit{true}.

6. We need to find adequately strong guards.
4.1 A Safe state

The state invariant should be:

1. false for all unsafe states    true for all safe states
2. An initial attempt at an invariant might be

\[ \neg \left( \text{lights(NorthSouth)} = \text{Green} \land \text{lights(EastWest)} = \text{Green} \right) \]

which, using the predicate identities

\[ \neg(P \land Q) \equiv \neg P \lor \neg Q \equiv P \implies \neg Q, \]

may be written

\[ \text{lights(NorthSouth)} = \text{Green} \implies \neg(\text{lights(EastWest)} = \text{Green}) \]

If used as a term in a larger predicate, the above implication may need to be parenthesised.

4.2 Strengthening the Invariant

Clearly, the light in both directions cannot be either Green or Amber.

<table>
<thead>
<tr>
<th></th>
<th>Red</th>
<th>Green</th>
<th>Amber</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red</td>
<td>safe</td>
<td>safe</td>
<td>safe</td>
</tr>
<tr>
<td>Green</td>
<td>safe</td>
<td>unsafe</td>
<td>unsafe</td>
</tr>
<tr>
<td>Amber</td>
<td>safe</td>
<td>unsafe</td>
<td>unsafe</td>
</tr>
</tbody>
</table>

This leads to the invariant:

\[ \neg(\text{lights(NorthSouth)} \in \{\text{Green, Amber}\} \land \text{lights(EastWest)} \in \{\text{Green, Amber}\}) \]

\[ \equiv \]

\[ (\text{lights(NorthSouth)} \in \{\text{Green, Amber}\} \implies \neg(\text{lights(EastWest)} \in \{\text{Green, Amber}\})) \]

\[ \equiv \]

\[ (\text{lights(NorthSouth)} \in \{\text{Green, Amber}\} \implies \text{lights(EastWest)} = \text{Red}) \] (3)

4.3 Other Invariants

There are other invariants that adequately express safety for a two-way intersection:

\[ \text{lights(NorthSouth)} = \text{Red} \lor \text{lights(EastWest)} = \text{Red} \]
\( \text{Red} \in \text{ran}(\text{lights}) \)

But these conditions do not generalise to intersections with more than two ways. Indeed the expression of the invariant that best generalises is

\[
\forall \text{dir}. \text{dir} \in \text{DIRECTION} \land \text{lights}(\text{dir}) \in \{\text{Green, Amber}\} \implies \\
\text{lights}(\text{OTHERDIR}(\text{dir})) = \text{Red}
\]  

(4)

4.4 Some Theorems

In this model of TwoWay we add the following alternative formulations of the invariant as theorems:

- \( \forall \text{dir}. (\text{dir} \in \text{DIRECTION} \land \text{lights}(\text{dir}) \in \{\text{Green, Amber}\} \implies \\
\text{lights}(\text{OTHERDIR}(\text{dir})) = \text{Red} ) \)

- \( \text{lights}(\text{EastWest}) \in \{\text{Green, Amber}\} \implies \\
\text{lights}(\text{NorthSouth}) = \text{Red} \)

Theorems contain properties that are implied by the invariant, and hence discharging the proof obligations for the assertions proves that each conjunct in the assertions is implied by the invariant.

5 The ChangeLight Event

The ChangeLight event will have two parameters \( \text{dir} \) and \( \text{light} \).

The event will change the lights in direction \( \text{dir} \) to colour \( \text{light} \)

\( \text{lights}(\text{dir}) := \text{light} \)

The guards need to be strong enough to ensure both safety and correct sequencing.

Safety can be ensured with the guard:

\[
(light \in \{\text{Green, Amber}\} \implies \\
\text{lights}(\text{OTHERDIR}(\text{dir})) = \text{Red})
\]

5.1 Sequencing

Changing to

\textbf{Red}: the current colour should be \textit{Amber}, so \( light = \text{Red} \implies \text{lights}(\text{dir}) = \text{Green} \)

\textbf{Green}: the current colour should be \textit{Red}, so \( light = \text{Green} \implies \text{lights}(\text{dir}) = \text{Red} \)

\textbf{Amber}: the current colour should be \textit{Green}, so \( light = \text{Amber} \implies \text{lights}(\text{dir}) = \text{Green} \)

Notice that the sequencing guards for both \textit{Red} and \textit{Amber} together with the invariant ensure safety, we only need to be concerned with safety for changing to \textit{Green}. 

5
Final Guards for ChangeLight

The final guards that ensure both safety and correct sequencing are:

\[
\begin{align*}
\text{light} = \text{Red} & \implies \text{lights(dir)} = \text{Amber} \\
\text{light} = \text{Green} & \implies \text{lights(dir)} = \text{Red} \\
\text{light} = \text{Green} & \implies \text{lights(OTHERDIR(dir))} = \text{Red} \\
\text{light} = \text{Amber} & \implies \text{lights(dir)} = \text{Green}
\end{align*}
\]

The ChangeLight Event

**MACHINE**  ChangeLights

**SEES**  TrafficLights_ctx

**VARIABLES**  lights

**INVARIANTS**

\(\text{inv1}: \text{lights} \in \text{DIRECTION} \to \text{LIGHTS}\)

\(\text{inv2}: \text{lights(NorthSouth)} \in \{\text{Green, Amber}\} \Rightarrow \text{lights(EastWest)} = \text{Red}\)

**EVENTS**

**Initialisation**

\[\begin{align*}
\text{begin} & \\
\text{act1:} & \quad \text{lights} := \{\text{NorthSouth} \mapsto \text{Red, EastWest} \mapsto \text{Red}\}
\end{align*}\]

\[\text{end}\]

**Event**  ToAmber  \(\hat{=}\)

\[\begin{align*}
\text{any } & \quad \text{dir} \\
\text{when} & \\
\text{grd1:} & \quad \text{dir} \in \text{DIRECTION} \\
\text{grd2:} & \quad \text{lights(dir)} = \text{Green} \\
\text{then} & \\
\text{act1:} & \quad \text{lights(dir)} := \text{Amber}
\end{align*}\]

\[\text{end}\]

**Event**  ToGreen  \(\hat{=}\)

\[\begin{align*}
\text{any } & \quad \text{dir} \\
\text{when} & \\
\text{grd1:} & \quad \text{dir} \in \text{DIRECTION}
\end{align*}\]
grd2: $\text{lights}({\text{OTHERDIR}}(\text{dir})) =$ Red

grd3: $\text{lights}(\text{dir}) =$ Red

then

act1: $\text{lights}(\text{dir}) := \text{Green}$

end

Event $\text{ToRed} \triangleq$

any $\text{dir}$

when

grd1: $\text{dir} \in \text{DIRECTION}$

grd2: $\text{lights}(\text{dir}) =$ Amber

then

act1: $\text{lights}(\text{dir}) := \text{Red}$

end

END

6 A General Multi-Way Intersection

We will now expand to a completely general intersection with many directions, denoted by the finite set $\text{DIRECTION}$. To define directions that conflict with one another we will use a relation $\text{CONFLICT}$ that maps each direction to all other directions, with which it conflicts. Clearly, $\text{CONFLICT}$ must have the following properties:

- No direction conflicts with itself.
- Conflicts are symmetric: if $\text{dir}_1$ conflicts with $\text{dir}_2$ then $\text{dir}_2$ conflicts with $\text{dir}_1$.

6.1 The New Safety Invariant

The safety invariant for our generalised intersection is

$$\forall d \in \text{DIRECTION} \land \text{lights}(d) \in \{\text{Green, Amber}\} \implies \text{lights}([\text{CONFLICT}][\{d\}]) = \{\text{Red}\} \quad (5)$$

This can be seen as a generalisation of the earlier safety invariant \[4\].
6.2 A Traffic Light Controller

We will now model a traffic light controller that responds to the following requirements:

1. A controller is required that can be used to control the whole of a general intersection.

2. The controller must be able to set the light in any direction to Green and must do so in a safe transition sequence.

3. The intersection must always be in a safe state.

Our Response

Our response to the preceding request is

1. to produce a machine with a single event that changes the light in any arbitrary direction, modelled by a constant $\text{adir}$, to Green, setting all lights in conflicting directions to Red;

2. to model the top-level state as having only Red and Green, these being the “steady state” light colours;

3. to refine that machine to model the process of changing the state while maintaining a safe state.

4. towards safety, the transition from Green to Red will introduce the light colour Amber and delays between light transitions.
ChangeLight1 machine

MACHINE  ChangeLight1
SEES  TrafficLights1.context
VARIABLES  lights

INVARIANTS

inv1:  lights ∈ DIRECTION → {Red, Green}

inv2:  ∀d· d ∈ DIRECTION ∧ lights(d) = Green
       ⇒ lights[CONFLICT[{d}]] ⊆ {Red}

EVENTS

Initialisation
begin

act1:  lights : |
       lights' ∈ DIRECTION → {Red, Green}
       ∧ (∀d· d ∈ DIRECTION
       ∧ lights'(d) = Green
       ⇒ lights'[CONFLICT[{d}]] ⊆ {Red})

end

Event  ToGreen =
when

grd1:  lights(adir) = Red
then

act1:  lights := lights
       ↔ (CONFLICT[{adir}] × {Red})
       ↔ {adir ↦ Green}

end

END

The Refinement Model

The first stage is still abstract and would be very unsafe, if implemented literally according to the model, as it is not safe to change lights instantaneously from Green to Red. This refinement introduces Amber into the lights sequence and the following slave events:

ToAmber  changes, in arbitrary sequence, all the currently Green lights in conflicting directions to Amber;

ToRed:  changes the Amber lights to Red;
Delay: models a delay between Amber and Red and between Red and the final setting of the light in the adir direction to Green.

The refinement of ToGreen waits until all the conflicting directions have been changed to Red and then sets the light in the adir direction to Green.

ChangeLight1R refinement

MACHINE ChangeLight1R
REFINES ChangeLight1
SEES TrafficLights1_ctx

VARIABLES xlights Extended lights, Red, Green and Amber lights delay delay between Amber and Red, and between Red and Green

IN妪ARANTS

inv1: xlights ∈ DIRECTION → LIGHTS
inv2: ∀d·d ∈ DIRECTION ∧ xlights[[d]] ⊆ {Green, Amber} ⇒ xlights[CONFLICT[[d]]] ⊆ {Red}
inv3: xlights ≪ (CONFLICT[[adir]] × {Red}) = lights ≪ (CONFLICT[[adir]] × {Red})
inv4: xlights(adir) = Green ⇒ lights = xlights
inv5: delay ⊆ DIRECTION

thm1: finite(xlights)

thm2: CONFLICT[[adir]] ≪ lights = CONFLICT[[adir]] ≪ xlights

thm3: ∀d, b, a·d ∈ DIRECTION ∧ b ∈ LIGHTS ∧ a ∈ LIGHTS ∧ a ≠ b ∧ xlights(d) = b ⇒ card((xlights ≪ {d → a}) ▷ {b}) = card(xlights ▷ {b}) − 1
from b (= before) to a (= after) decreases number of colour b lights by 1

thm4: ∀d, b, a·d ∈ DIRECTION ∧ b ∈ LIGHTS ∧ a ∈ LIGHTS ∧ a ≠ b ∧ xlights(d) = b ⇒ card((xlights ≪ {d → a}) ▷ {a}) = card(xlights ▷ {a}) + 1
from b (= before) to a (=after) increases number of colour a lights by 1
thm5: \( \forall d, b, a, c \cdot d \in DIRECTION \land b \in LIGHTS \land a \in LIGHTS \land c \in LIGHTS \land \text{xlights}(d) = b \land c \neq a \land c \neq b \Rightarrow \)  
\[
\text{card}(\text{xlights} \triangleleft \{d \mapsto a\} \triangleright \{c\}) = \text{card}(\text{xlights} \triangleright \{c\})
\]
changing light in direction \(d\) from \(b\) to \(a\) \((= \text{before})\) to \(a\) \((= \text{after})\) does not change number of colour \(c\), \(c \neq a\), \(c \neq b\)

EVENTS

Initialisation

begin
with

\( \text{lights'}: \text{lights'} = \text{xlights}' \)

act1: \( \text{xlights} : | \text{xlights}' \in DIRECTION \rightarrow \{\text{Red, Green}\} \land (\forall d \cdot d \in DIRECTION \land \text{xlights}'(d) = \text{Green} \Rightarrow \text{xlights}'[\text{CONFLICT}[\{d\}]] \subseteq \{\text{Red}\}) \)

act2: delay := \(\emptyset\)

end

Event \( \text{ToGreen} \)
refines \( \text{ToGreen} \)
when

\( \text{grd1}: \text{xlights}(\text{adir}) = \text{Red} \)

\( \text{grd2}: \text{xlights}[\text{CONFLICT}[\{\text{adir}\}]] \subseteq \{\text{Red}\} \)

\( \text{grd3}: \text{delay} = \emptyset \)
then

act1: \( \text{xlights} := \text{xlights} \triangleleft \{\text{adir} \mapsto \text{Green}\} \)

end

Event \( \text{GreenToAmber} \)
Status convergent
any \( \text{dir} \)
when

\( \text{grd1}: \text{dir} \in \text{CONFLICT}[\{\text{adir}\}] \)
\[\text{grd}2: \quad \text{xlights}(\text{dir}) = \text{Green} \]
\[\text{grd}3: \quad \text{xlights}(\text{adir}) \neq \text{Green} \]
\[\text{grd}4: \quad \text{adir} \notin \text{delay} \]
\[\text{then} \]
\[\text{act}1: \quad \text{xlights}(\text{dir}) := \text{Amber} \]
\[\text{act}2: \quad \text{delay} := \text{delay} \cup \{\text{dir}\} \]
\[\text{end} \]

\textbf{Event} \quad \text{AmberToRed} \quad \hat{=} \\
\textbf{Status} \quad \text{convergent} \\
\textbf{any} \quad \text{dir} \\
\textbf{when} \\
\[\text{grd}1: \quad \text{dir} \in \text{CONFLICT}\{\text{adir}\} \]
\[\text{grd}2: \quad \text{xlights}(\text{dir}) = \text{Amber} \]
\[\text{grd}3: \quad \text{adir} \notin \text{delay} \]
\[\text{grd}4: \quad \text{xlights}(\text{adir}) \neq \text{Green} \]
\[\text{then} \]
\[\text{act}1: \quad \text{xlights}(\text{dir}) := \text{Red} \]
\[\text{act}2: \quad \text{delay} := \text{delay} \cup \{\text{dir}\} \]
\[\text{end} \]

\textbf{Event} \quad \text{Delay} \quad \hat{=} \\
\textbf{Status} \quad \text{convergent} \\
\textbf{any} \quad \text{dir} \\
\textbf{when} \\
\[\text{grd}1: \quad \text{dir} \in \text{delay} \]
\[\text{then} \]
\[\text{act}1: \quad \text{delay} := \text{delay} \setminus \{\text{dir}\} \]
\[\text{end} \]

\textbf{VARIANT} \quad 4 \ast \text{card(xlights} \supset \{\text{Green}\}) \\
\[+ 2 \ast \text{card(xlights} \supset \{\text{Amber}\}) \]
\[+ \text{card(delay)} \]

\textbf{END}
The Variant

The events GreenToAmber, AmberToRed and Delay are slave events; that is they are not the main event, which is ToGreen. Consequently they must not be able to be enabled forever; that is they must be convergent.

In EventB this is verified by providing a variant expression: an expression that has a lower bound and must strictly decrease everytime a convergent event fires.

In this model the variant is an expression that yields a natural number —hence lower bound 0— that strictly decreases.

The Variant

The variant is a sum of the following:

4 times the number of Red lights;
2 times the number of Amber lights;
1 times the number of current delays

This is derived from the GreenToAmber event that

- decreases the number of Red lights by 1;
- increases the number of Amber lights by 1;
- increases the number of delays by 1.

and the AmberToRed event that

- decreases the number of Amber lights by 1;
- increases the number of delays by 1.

7 A General Multi-Way Intersection Parametric Controller

Extending the model

In this development a parameterised version of the preceding traffic light control system is presented. The difference will start to be apparent when you look at the new context in which you will notice that the constant adir is missing.

When you look at the new ChangeLight machine you will notice a few differences. First, there are some new events, or rather one renamed event ToGreen, which renames ChangeLight, and a new event ToRed that changes a green light to red. Notice that both of these events have parameters, so the new ToGreen and ToRed events are now genuinely parameterised and don’t depend on a single constant value.

The change to parameterisation has virtually no effect on the ChangeLight machine but has significant affects on the refinement. The refinement now has a variable, rdirc, which is used to the parameter for both the ToGreen and ToRed events. Also notice that there are two new variables togreen and tored. The
invariant reveals that both of these variables are of type BOOL. The purpose of these two variables is will be explained a little later.

Both BOOL variables are initialised to FALSE. Notice that different names have been chosen for the events in the refinement to reduce confusion now that we are refining two abstract operations.

Again, notice that the refinement of the ToGreen event fires only when the conflicting directions have been set to Red, but also notice that this event will only fire if the BOOL variable togreen is TRUE.

A new event ToGreenInit appears. The purpose of this event is to initialise the state for the sequence of events that effectively refine the ToGreen event. ToGreenInit can only fire when the BOOL variables togreen and tored are both FALSE.

New events GreenToAmber and AmberToRed produce transitions of the Green conflicting lights through to Red. Each of these events can only fire when togreen is TRUE. Notice a similar initialisation event ToRedInit for the refinement of the ToRed event.

The BOOL variables are required because we are refining two atomic events ToGreen and ToRed and the refinement involves a sequence of events. We need to prevent interference and preserve the atomicity of the ToGreen and ToRed events. The BOOL variables ensure that the complete sequence of events making up a refinement are completed before any other event can be triggered. This problem did not arise in the previous development because the ToGreen event had a fixed parameter and there was no possibility for interference.

TrafficLights2.ctx context

CONTEXT TrafficLights2.ctx

SETS LIGHTS DIRECTION

CONSTANTS Red Green Amber CONFLICT

AXIOMS

\(\text{axm1:} \) \(\text{partition} \left( \text{LIGHTS}, \{\text{Red}\}, \{\text{Green}\}, \{\text{Amber}\} \right)\)

\(\text{axm2:} \) \(\text{finite} \left( \text{DIRECTION} \right)\) \hspace{2cm} \text{DIRECTION is a finite set of directions}

\(\text{axm3:} \) \(\text{CONFLICT} \in \text{DIRECTION} \iff \text{DIRECTION} \) \hspace{2cm} \text{CONFLICT relates conflicting directions}

\(\text{axm4:} \) \(\text{CONFLICT} \cap (\text{DIRECTION} \triangleleft \text{id}) = \emptyset\) \hspace{2cm} \text{a direction cannot conflict with itself}

\(\text{axm5:} \) \(\text{CONFLICT}^{-1} = \text{CONFLICT}\) \hspace{2cm} \text{conflicts are symmetric}

\(\text{thm1:} \) \(\forall d \in \text{DIRECTION} \Rightarrow d \notin \text{CONFLICT} \left[ \{d\} \right]\)

\(\text{thm2:} \) \(\forall d_1, d_2 \in \text{CONFLICT} \left[ \{d_2\} \right] \Rightarrow d_2 \notin \text{CONFLICT} \left[ \{d_1\} \right]\)

END
ChangeLight2 machine

**MACHINE**  ChangeLight2

**SEES**  TrafficLights2_ctx

**VARIABLES**  lights

**ININVARIANTS**

inv1:  \( \text{lights} \in \text{DIRECTION} \to \{\text{Red}, \text{Green}\} \)

inv2:  \( \forall d \in \text{DIRECTION} \land \text{lights}(d) = \text{Green} \)  \( \Rightarrow \)  \( \text{lights}[\text{CONFLICT}[\{d\}] ] \subseteq \{\text{Red}\} \)

**thm1**:  finite(lights)

**EVENTS**

**Initialisation**

**begin**

act1:  \( \text{lights} : [\text{lights}' \in \text{DIRECTION} \to \{\text{Red}, \text{Green}\}] \land (\forall d \in \text{DIRECTION} \land \text{lights}'(d) = \text{Green} \Rightarrow \text{lights}'[\text{CONFLICT}[\{d\}] ] \subseteq \{\text{Red}\}) \)

**end**

Event  ToGreen  \( \triangleq \)

any  adir

when

grd1:  \( \text{lights}(adir) = \text{Red} \)

then

act1:  \( \text{lights}(adir) := \text{Red} \)

\( \Leftarrow (\text{CONFLICT}[\{adir\}] \times \{\text{Red}\}) \)

\( \Leftarrow \{\text{adir} \mapsto \text{Green}\} \)

**end**

Event  ToRed  \( \triangleq \)

any  adir

when

grd1:  \( \text{lights}(adir) = \text{Green} \)

then

act1:  \( \text{lights}(adir) := \text{Red} \)

**end**

**END**
ChangeLight2R refinement

**MACHINE**  ChangeLight2R

**REFINES**  ChangeLight2

**SEES**  TrafficLights2_ctx

**VARIABLES**  xlights  Extended lights, Red, Green and Amber lights  delay  delay between Amber and Red, Red and Green  rdir  current argument of ToGreen or ToRed  togreen  tored

**INVARINTS**

inv1:  xlights ∈ DIRECTION → LIGHTS

inv2:  ∀d∈DIRECTION ∧ xlights[d] ⊆ {Green, Amber}
       ⇒
       xlights[CONFLICT[d]] ⊆ {Red}

inv3:  togreen = FALSE ∧ tored = FALSE
       ⇒
       lights = xlights

inv4:  rdir ∈ DIRECTION

inv5:  delay ⊆ DIRECTION

inv6:  togreen ∈ BOOL

inv7:  tored ∈ BOOL

inv8:  togreen = TRUE ⇒ tored = FALSE

inv9:  togreen = TRUE ⇒ lights(rdir) = Red

inv10:  togreen = TRUE⇒
       xlights ≡ (CONFLICT[rdir]) × {Red}) ≡ {rdir → Green}
       =
       lights ≡ (CONFLICT[rdir]) × {Red}) ≡ {rdir → Green}

thm1:  togreen = TRUE
       ⇒
       CONFLICT[rdir] ≡ ({rdir} ≡ xlights)
       =
       CONFLICT[rdir] ≡ ({rdir} ≡ lights)

inv11:  tored = TRUE ⇒ lights(rdir) = Green

inv12:  tored = TRUE
       ⇒
       xlights ≡ {rdir → Red}
       =
       lights ≡ {rdir → Red}
thm2:  \( \text{tored} = \text{TRUE} \)  
\[ \Rightarrow \{rdir\} \preceq \text{lights} = \{rdir\} \preceq x\text{lights} \]

thm3:  \( \text{finite}(x\text{lights}) \)

thm4:  \( \forall d, b, a \cdot a \in \text{LIGHTS} \land a \neq b \land x\text{lights}(d) = b \Rightarrow \text{card}(x\text{lights} \leftarrow \{d \mapsto a\} \triangleright \{b\}) = \text{card}(x\text{lights} \triangleright \{b\}) - 1 \)  
(= after) decreases number of colour b lights by 1

thm5:  \( \forall d, b, a \cdot a \in \text{LIGHTS} \land a \neq b \land x\text{lights}(d) = b \Rightarrow \text{card}(x\text{lights} \leftarrow \{d \mapsto a\} \triangleright \{a\}) = \text{card}(x\text{lights} \triangleright \{a\}) + 1 \)  
(= after) increases number of colour a lights by 1

thm6:  \( \forall d, b, a, c \cdot a \in \text{LIGHTS} \land c \in \text{LIGHTS} \land c \neq a \land c \neq b \land x\text{lights}(d) = b \Rightarrow \text{card}(x\text{lights} \leftarrow \{d \mapsto a\} \triangleright \{c\}) = \text{card}(x\text{lights} \triangleright \{c\}) \)  
changing light in direction d from b (= before) to a to a (=after) does not change number of colour c, c /= a, c /= b

EVENTS

Initialisation

begin  
with  
\( x\text{lights}' : \text{lights}' = x\text{lights}' \)
act1:  \( x\text{lights} : |x\text{lights}' \in \text{DIRECTION} \rightarrow \{\text{Red, Green}\} \land (\forall d \cdot x\text{lights}'(d) = \text{Green} \Rightarrow x\text{lights}'[\text{CONFLICT}([d])] \subseteq \{\text{Red}\}) \)
act2:  \( \text{delay} := \emptyset \)
act3:  \( \text{togreen, tored} := \text{FALSE, FALSE} \)
act5:  \( r\text{dir} \in \text{DIRECTION} \)
end

Event  \( \text{ToGreen} \triangleq \)
refines ToGreen

when

grd1: togreen = TRUE

grd2: xlights[CONFLICT[{rdir}]] ⊆ {Red}

grd3: rdir ∉ delay

with

adir: adir = rdir

then

act1: xlights(rdir) := Green

act2: togreen := FALSE

end

Event ToGreenInit ≡

any

adir

when

grd1: togreen = FALSE

grd2: tored = FALSE

grd3: xlights(adir) = Red

then

act1: rdir := adir

act2: togreen := TRUE

end

Event GreenToAmber ≡

Status convergent

any

dir

when

grd1: togreen = TRUE

grd2: dir ∈ CONFLICT[\{rdir\}]

grd3: xlights(dir) = Green
$$\text{grd4: } \text{dir} \notin \text{delay}$$

then

$$\text{act1: } \text{xlights}(\text{dir}) := \text{Amber}$$

$$\text{act2: } \text{delay} := \text{delay} \cup \{\text{dir}\}$$

end

Event $\text{AmberToRed} \equiv$

Status convergent

any

$\text{dir}$

when

$$\text{grd1: } \text{togreen} = \text{TRUE}$$

$$\text{grd2: } \text{dir} \in \text{CONFLICT}\{\text{rdir}\}$$

$$\text{grd3: } \text{xlights}(\text{dir}) = \text{Amber}$$

$$\text{grd4: } \text{dir} \notin \text{delay}$$

then

$$\text{act1: } \text{xlights}(\text{dir}) := \text{Red}$$

$$\text{act2: } \text{delay} := \text{delay} \cup \{\text{rdir}\}$$

end

Event $\text{Delay} \equiv$

Status convergent

any

$\text{dir}$

when

$$\text{grd1: } \text{dir} \in \text{delay}$$

then

$$\text{act1: } \text{delay} := \text{delay} \setminus \{\text{dir}\}$$

end

Event $\text{ToRed} \equiv$

refines $\text{ToRed}$

when
grd1: tored = TRUE
grd2: xlights(rdir) = Amber
grd3: rdir $\notin$ delay

with

adir : adir = rdir

then

act1: xlights(rdir) := Red
act2: tored := FALSE

end

Event ToRedInit $\sqsubseteq$

any

adir

when

grd1: xlights(adir) = Green
grd2: tored = FALSE
grd3: togreen = FALSE

then

act1: rdir := adir
act2: tored := TRUE

dored := TRUE

end

Event ToAmber $\sqsubseteq$

when

grd1: tored = TRUE
grd2: xlights(rdir) = Green
grd3: rdir $\notin$ delay

then

act1: xlights(rdir) := Amber
act2: delay := delay $\cup$ \{rdir\}

end

VARIANT $\ 4 \ast \text{card}(xlights \triangleright \{\text{Green}\})$
$\ 2 \ast \text{card}(xlights \triangleright \{\text{Amber}\})$
$\ + \text{card}(delay)$

END