Chapter 2

Specifying a Simple Clock

2.1 Behaviors

Before we try to specify a system, let's look at how scientists do it. For centuries, they have described a system with equations that determine how its state evolves with time, where the state consists of the values of variables. For example, the state of the system comprising the earth and the moon might be described by the values of the four variables e_pos, m_pos, e_vel, and m_vel, representing the positions and velocities of the two bodies. These values are elements in a 3-dimensional space. The earth-moon system is described by equations expressing the variables' values as functions of time and of certain constants—namely, their masses and initial positions and velocities.

A behavior of the earth-moon system consists of a function F from time to states, $F(t)$ representing the state of the system at time t. A computer system differs from the systems traditionally studied by scientists because we can pretend that its state changes in discrete steps. So, we represent the execution of a system as a sequence of states. Formally, we define a *behavior* to be a sequence of states, where a state is an assignment of values to variables. We specify a system by specifying a set of possible behaviors—the ones representing a correct execution of the system.

2.2 An Hour Clock

Let's start with a very trivial system—a digital clock that displays only the hour. To make the system completely trivial, we ignore the relation between the display and the actual time. The hour clock is then just a device whose display cycles through the values 1 through 12. Let the variable hr represent the clock's display. A typical behavior of the clock is the sequence

(2.1) $[hr = 11] \;\rightarrow\; [hr = 12] \;\rightarrow\; [hr = 1] \;\rightarrow\; [hr = 2] \;\rightarrow\; \cdots$

of states, where $[hr = 11]$ is a state in which the variable hr has the value 11. A pair of successive states, such as $[hr = 1] \rightarrow [hr = 2]$, is called a *step*.

To specify the hour clock, we describe all its possible behaviors. We write an *initial predicate* that specifies the possible initial values of hr, and a *next-state relation* that specifies how the value of hr can change in any step.

We don't want to specify exactly what the display reads initially; any hour will do. So, we want the initial predicate to assert that hr can have any value from 1 through 12. Let's call the initial predicate *HCini*. We might informally define *HCini* by

The symbol $\triangleq$ means *is defined to equal.*

$$HCini \;\triangleq\; hr \in \{1, \ldots, 12\}$$

Later, we'll see how to write this definition formally, without the "..." that stands for the informal *and so on.*

The next-state relation *HCnxt* is a formula expressing the relation between the values of hr in the old (first) state and new (second) state of a step. We let hr represent the value of hr in the old state and hr' represent its value in the new state. (The $'$ in hr' is read *prime*.) We want the next-state relation to assert that hr' equals $hr + 1$ except if hr equals 12, in which case hr' should equal 1. Using an IF/THEN/ELSE construct with the obvious meaning, we can define *HCnxt* to be the next-state relation by writing

$$HCnxt \;\triangleq\; hr' \;=\; \textsc{if}\;\; hr \neq 12 \;\;\textsc{then}\;\; hr + 1 \;\;\textsc{else}\;\; 1$$

HCnxt is an ordinary mathematical formula, except that it contains primed as well as unprimed variables. Such a formula is called an *action*. An action is true or false of a step. A step that satisfies the action *HCnxt* is called an *HCnxt step*.

When an *HCnxt* step occurs, we sometimes say that *HCnxt* is *executed*. However, it would be a mistake to take this terminology seriously. An action is a formula, and formulas aren't executed.

We want our specification to be a single formula, not the pair of formulas *HCini* and *HCnxt*. This formula must assert about a behavior that (i) its initial state satisfies *HCini*, and (ii) each of its steps satisfies *HCnxt*. We express (i) as the formula *HCini*, which we interpret as a statement about behaviors to mean that the initial state satisfies *HCini*. To express (ii), we use the temporal-logic operator $\Box$ (pronounced *box*). The temporal formula $\Box F$ asserts that formula F is always true. In particular, $\Box HCnxt$ is the assertion that *HCnxt* is true for every step in the behavior. So, $HCini \wedge \Box HCnxt$ is true of a behavior iff the initial state satisfies *HCini* and every step satisfies *HCnxt*. This formula describes all behaviors like the one in (2.1) on this page; it seems to be the specification we're looking for.

If we considered the clock only in isolation and never tried to relate it to another system, then this would be a fine specification. However, suppose the clock is part of a larger system—for example, the hour display of a weather station that displays the current hour and temperature. The state of the station is described by two variables: hr, representing the hour display, and tmp, representing the temperature display. Consider this behavior of the weather station:

$$\begin{bmatrix} hr & = & 11 \\ tmp & = & 23.5 \end{bmatrix} \rightarrow \begin{bmatrix} hr & = & 12 \\ tmp & = & 23.5 \end{bmatrix} \rightarrow \begin{bmatrix} hr & = & 12 \\ tmp & = & 23.4 \end{bmatrix} \rightarrow \begin{bmatrix} hr & = & 12 \\ tmp & = & 23.3 \end{bmatrix} \rightarrow \begin{bmatrix} hr & = & 1 \\ tmp & = & 23.3 \end{bmatrix} \rightarrow \cdots$$

In the second and third steps, tmp changes but hr remains the same. These steps are not allowed by $\Box HCnxt$, which asserts that every step must increment hr. The formula $HCini \land \Box HCnxt$ does not describe the hour clock in the weather station.

A formula that describes *any* hour clock must allow steps that leave hr unchanged—in other words, $hr' = hr$ steps. These are called *stuttering steps* of the clock. A specification of the hour clock should allow both $HCnxt$ steps and stuttering steps. So, a step should be allowed iff it is either an $HCnxt$ step or a stuttering step—that is, iff it is a step satisfying $HCnxt \lor (hr' = hr)$. This suggests that we adopt $HCini \land \Box(HCnxt \lor (hr' = hr))$ as our specification. In TLA, we let $[HCnxt]_{hr}$ stand for $HCnxt \lor (hr' = hr)$, so we can write the formula more compactly as $HCini \land \Box[HCnxt]_{hr}$.

I pronounce $[HCnxt]_{hr}$ as *square HCnxt sub hr*.

The formula $HCini \land \Box[HCnxt]_{hr}$ does allow stuttering steps. In fact, it allows the behavior

$$[hr = 10] \rightarrow [hr = 11] \rightarrow [hr = 11] \rightarrow [hr = 11] \rightarrow \cdots$$

that ends with an infinite sequence of stuttering steps. This behavior describes a clock whose display attains the value 11 and then keeps that value forever—in other words, a clock that stops at 11. In a like manner, we can represent a terminating execution of any system by an infinite behavior that ends with a sequence of nothing but stuttering steps. We have no need of finite behaviors (finite sequences of states), so we consider only infinite ones.

It's natural to require that a clock does not stop, so our specification should assert that there are infinitely many nonstuttering steps. Chapter 8 explains how to express this requirement. For now, we content ourselves with clocks that may stop, and we take as our specification of an hour clock the formula HC defined by

$$HC \triangleq HCini \land \Box[HCnxt]_{hr}$$

2.3 A Closer Look at the Specification

A state is an assignment of values to variables, but what variables? The answer is simple: all variables. In the behavior (2.1) on page 16, $[hr = 1]$ represents some particular state that assigns the value 1 to hr. It might assign the value 23 to the variable tmp and the value $\sqrt{-17}$ to the variable m_pos. We can think of a state as representing a potential state of the entire universe. A state that assigns 1 to hr and a particular point in 3-space to m_pos describes a state of the universe in which the hour clock reads 1 and the moon is in a particular place. A state that assigns $\sqrt{-2}$ to hr doesn't correspond to any state of the universe that we recognize, because the hour clock can't display the value $\sqrt{-2}$. It might represent the state of the universe after a bomb fell on the clock, making its display purely imaginary.

A behavior is an infinite sequence of states—for example:

$$(2.2)\quad [hr = 11] \;\rightarrow\; [hr = 77.2] \;\rightarrow\; [hr = 78.2] \;\rightarrow\; [hr = \sqrt{-2}] \;\rightarrow\; \cdots$$

A behavior describes a potential history of the universe. The behavior (2.2) doesn't correspond to a history that we understand, because we don't know how the clock's display can change from 11 to 77.2. Whatever kind of history it represents is not one in which the clock is doing what it's supposed to.

Formula HC is a temporal formula. A temporal formula is an assertion about behaviors. We say that a behavior *satisfies* HC iff HC is a true assertion about the behavior. Behavior (2.1) satisfies formula HC. Behavior (2.2) does not, because HC asserts that every step satisfies $HCnxt$ or leaves hr unchanged, and the first and third steps of (2.2) don't. (The second step, $[hr = 77.2] \rightarrow [hr = 78.2]$, does satisfy $HCnxt$.) We regard formula HC to be the specification of an hour clock because it is satisfied by exactly those behaviors that represent histories of the universe in which the clock functions properly.

If the clock is behaving properly, then its display should be an integer from 1 through 12. So, hr should be an integer from 1 through 12 in every state of any behavior satisfying the clock's specification, HC. Formula $HCini$ asserts that hr is an integer from 1 through 12, and $\Box HCini$ asserts that $HCini$ is always true. So, $\Box HCini$ should be true for any behavior satisfying HC. Another way of saying this is that HC implies $\Box HCini$, for any behavior. Thus, the formula $HC \Rightarrow \Box HCini$ should be satisfied by *every* behavior. A temporal formula satisfied by every behavior is called a *theorem*, so $HC \Rightarrow \Box HCini$ should be a theorem.[1] It's easy to see that it is: HC implies that $HCini$ is true initially (in the first state of the behavior), and $\Box[HCnxt]_{hr}$ implies that each step either advances hr to its proper next value or else leaves hr unchanged. We can formalize this reasoning using the proof rules of TLA, but we're not going to delve into proofs and proof rules.

[1] Logicians call a formula *valid* if it is satisfied by every behavior; they reserve the term *theorem* for provably valid formulas.

2.4 The Specification in TLA$^+$

Figure 2.1 on the next page shows how the hour-clock specification can be written in TLA$^+$. There are two versions: the ASCII version on the bottom is the actual TLA$^+$ specification, the way you type it; the typeset version on the top is one that the TLATEX program, described in Chapter 13, might produce. Before trying to understand the specification, observe the relation between the two syntaxes.

- Reserved words that appear in small upper-case letters (like EXTENDS) are written in ASCII with ordinary upper-case letters.
- When possible, symbols are represented pictorially in ASCII—for example, $\Box$ is typed as `[]` and $\neq$ as `#`. (You can also type $\neq$ as `/=`.)
- When there is no good ASCII representation, TEX notation[2] is used—for example, $\in$ is typed as `\in`. The major exception is $\triangleq$, which is typed as `==`.

A complete list of symbols and their ASCII equivalents appears in Table 8 on page 273. I will usually show the typeset version of a specification; the ASCII versions of all the specifications in this book can be found through the TLA Web page.

Now let's look at what the specification says. It starts with

MODULE *HourClock*

which begins a module named *HourClock*. TLA$^+$ specifications are partitioned into modules; the hour clock's specification consists of this single module.

Arithmetic operators like + are not built into TLA$^+$, but are themselves defined in modules. (You might want to write a specification in which + means addition of matrices rather than numbers.) The usual operators on natural numbers are defined in the *Naturals* module. Their definitions are incorporated into module *HourClock* by the statement

EXTENDS *Naturals*

Every symbol that appears in a formula must either be a built-in operator of TLA$^+$, or else it must be declared or defined. The statement

VARIABLE *hr*

declares *hr* to be a variable.

[2]The TEX typesetting system is described in *The TEXbook* by Donald E. Knuth, published by Addison-Wesley, Reading, Massachusetts, 1986.

MODULE *HourClock*

EXTENDS *Naturals*

VARIABLE *hr*

$HCini \triangleq hr \in (1 \mathrel{..} 12)$

$HCnxt \triangleq hr' = \text{IF } hr \neq 12 \text{ THEN } hr + 1 \text{ ELSE } 1$

$HC \triangleq HCini \land \Box[HCnxt]_{hr}$

THEOREM $HC \Rightarrow \Box HCini$

```
---------------------- MODULE HourClock ----------------------
EXTENDS Naturals
VARIABLE hr
HCini  ==  hr \in (1 .. 12)
HCnxt  ==  hr' = IF hr # 12 THEN hr + 1 ELSE 1
HC  ==  HCini /\ [][HCnxt]_hr
--------------------------------------------------------------
THEOREM  HC => []HCini
==============================================================
```

Figure 2.1: The hour-clock specification—typeset and ASCII versions.

To define *HCini*, we need to express the set $\{1, \ldots, 12\}$ formally, without the ellipsis "...". We can write this set out completely as

$$\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}$$

but that's tiresome. Instead, we use the operator "..", defined in the *Naturals* module, to write this set as $1 \mathrel{..} 12$. In general $i \mathrel{..} j$ is the set of integers from i through j, for any integers i and j. (It equals the empty set if $j < i$.) It's now obvious how to write the definition of *HCini*. The definitions of *HCnxt* and *HC* are written just as before. (The ordinary mathematical operators of logic and set theory, like $\land$ and $\in$, are built into TLA$^+$.)

The line

---

can appear anywhere between statements; it's purely cosmetic and has no meaning. Following it is the statement

THEOREM $HC \Rightarrow \Box HCini$

of the theorem that was discussed above. This statement asserts that the formula $HC \Rightarrow \Box HCini$ is true in the context of the statement. More precisely, it asserts that the formula follows logically from the definitions in this module, the definitions in the *Naturals* module, and the rules of TLA$^+$. If the formula were not true, then the module would be incorrect.

The module is terminated by the symbol

The specification of the hour clock is the definition of HC, including the definitions of the formulas $HCnxt$ and $HCini$ and of the operators $\,..\,$ and $+$ that appear in the definition of HC. Formally, nothing in the module tells us that HC rather than $HCini$ is the clock's specification. TLA$^+$ is a language for writing mathematics—in particular, for writing mathematical definitions and theorems. What those definitions represent, and what significance we attach to those theorems, lies outside the scope of mathematics and therefore outside the scope of TLA$^+$. Engineering requires not just the ability to use mathematics, but the ability to understand what, if anything, the mathematics tells us about an actual system.

2.5 An Alternative Specification

The *Naturals* module also defines the modulus operator, which we write %. The formula $i \% n$, which mathematicians write $i \bmod n$, is the remainder when i is divided by n. More formally, $i \% n$ is the natural number less than n satisfying $i = q * n + (i \% n)$ for some natural number q. Let's express this condition mathematically. The *Naturals* module defines Nat to be the set of natural numbers, and the assertion that there exists a q in the set Nat satisfying a formula F is written $\exists\, q \in Nat : F$. Thus, if i and n are elements of Nat and $n > 0$, then $i \% n$ is the unique number satisfying

$$(i \% n \in 0 \,..\, (n-1)) \land (\exists\, q \in Nat : i = q * n + (i \% n))$$

We can use % to simplify our hour-clock specification a bit. Observing that $(11 \% 12) + 1$ equals 12 and $(12 \% 12) + 1$ equals 1, we can define a different next-state action $HCnxt2$ and a different formula $HC2$ to be the clock specification

$$HCnxt2 \;\triangleq\; hr' = (hr \% 12) + 1 \qquad HC2 \;\triangleq\; HCini \land \Box[HCnxt2]_{hr}$$

Actions $HCnxt$ and $HCnxt2$ are not equivalent. The step $[hr = 24] \to [hr = 25]$ satisfies $HCnxt$ but not $HCnxt2$, while the step $[hr = 24] \to [hr = 1]$ satisfies $HCnxt2$ but not $HCnxt$. However, any step starting in a state with hr in $1 \,..\, 12$ satisfies $HCnxt$ iff it satisfies $HCnxt2$. It's therefore not hard to deduce that any behavior starting in a state satisfying $HCini$ satisfies $\Box[HCnxt]_{hr}$ iff it satisfies $\Box[HCnxt2]_{hr}$. Hence, formulas HC and $HC2$ *are* equivalent. In other words, $HC \equiv HC2$ is a theorem. It doesn't matter which of the two formulas we take to be the specification of an hour clock.

Mathematics provides infinitely many ways of expressing the same thing. The expressions $6 + 6$, $3 * 4$, and $141 - 129$ all have the same meaning; they are just different ways of writing the number 12. We could replace either instance of the number 12 in module *HourClock* by any of these expressions without changing the meaning of any of the module's formulas.

When writing a specification, you will often be faced with a choice of how to express something. When that happens, you should first make sure that the choices yield equivalent specifications. If they do, then you can choose the one that you feel makes the specification easiest to understand. If they don't, then you must decide which one you mean.