Harel's Watch as a Statestep Model

Michael Breen

(updated) October 2005

Contents

References

[1] M. Breen. (2005) Statestep specification technique: User guide. [Online]. Available: http://statestep.com

[2] D. Harel, "Statecharts: a visual formalism for complex systems," *Science of Computer Programming*, vol. 8, pp. 231–274, July 1987.

1 Introduction

This document illustrates the tabular notation described in [1] by using it to specify a digital watch. The specification is intended to be isomorphic to that used as an example by David Harel in his paper introducing Statecharts [2]. In other words, there is a one-to-one mapping from the possible statechart configurations and the transitions between them to the system states in the tabular specification and the transitions between them. Sections 2 and 3 may interest those seeking to compare the two notations but are a little technical and assume the reader has Harel's specification to hand.

Even without the original specification, the tabular specification presented in Section 4 should serve to illustrate the notation. Note that, though the notation does not require it, the finite state aspects of the behaviour are described exclusively in tabular form – leading to more rules than would otherwise be needed.

It is also a somewhat imperfect example: Although the digital watch model has a reasonable number of variables, it is not sufficiently complex to properly demonstrate the strengths of the approach (in particular, a constraint table consisting almost entirely of "don't cares" inevitably looks superfluous and and laboured). However, the watch is not quite trivial either and the translation does expose some issues with the original Statecharts model.

2 Guide to State Mapping

Much of the mapping between the Statecharts and the Statestep models should be obvious. However, it was convenient to make some structural changes which we explain below. While a more direct translation would be possible, understanding the motivation for Harel's original modeling choices and for the choice of variables made here should do more to highlight than to obscure the differences between the two approaches.

`update`, `wait`

Harel models three sets of update states; one which is reached from `time` (via `wait`) in `regular` and the other two for updating the two alarm times. These are similar and we choose instead to model a single orthogonal variable called *Update* and to interpret it in the context of the *Display* variable. Table 1 shows some sample mappings. Note that the calendar can only be updated in a state reached from `time` (rather than `date`) and the watch returns to `time` on exiting (calendar) update. This is why we have chosen a mapping which may seem somewhat confusing at first. It should not be inferred that the time is being displayed while the calendar is being updated (which seems impossible from Harel's picture of the watch face). Instead (if we are interested in an interpretation), we regard *Display* as "display mode," remembering that it does not uniquely determine what appears on the physical display.

Table 1: Sample state mappings between the models.

Statecharts	Statestep	
	Display	*Update*
`regular.time`	*time*	*none*
`regular.date`	*date*	*none*
`regular.update.sec`	*time*	*sec*
`regular.update.year`	*time*	*year*
`wait`	*time*	*pending*

Following logically from the above, it was decided to represent `wait` by giving *Update* the value *pending*.

`stopwatch`

We make one exception to our rule that every statechart configuration can be mapped onto an equivalent state in our model:

`zero` and (`display.reg`, `run.off`)

in the statechart both map to

(*Stopw_display* = *regular*, *Stopw_run* = *stopped*)

in the tables. The only difference between the two statechart configurations is that the displayed (stopwatch) time is guaranteed to be zero in one case. Since this is essentially a data aspect and no distinction is made between other displayed times (i.e., we do not specify a multitude of states to model the incrementing time while the stopwatch is running), it seems inconsistent to model it as a state. However, we do have to be aware that the stopwatch is reset to zero when button 'd' is pressed in the appropriate state – a comment to this effect is therefore included in the corresponding rule in the transition table.

`alarms-beep`

For each alarm, we choose to model whether it is beeping or silent using a separate variable, *AlarmX* (which also models whether the alarm is enabled or disabled).

Examining the statechart, one may wonder why Harel did not take this approach himself, since it seems somewhat counter-intuitive to put it as a state disjoint from the `displays` state (after all, one assumes that the display is not altered by the beeping of the alarms, e.g., if the current time is being displayed then it continues to be displayed). On closer inspection, we see his motivation: The specification currently expresses that when an alarm is beeping, pressing any button merely returns the displays component to the substate(s) it was in before the alarm

began to sound – it has no further effect. Had Harel modeled beeping as part of an orthogonal state machine, he would have been obliged to add the condition

```
not in alarm1-status.beep and not in alarm2-status.beep
```

to each of the transitions for button presses in the `displays` state.

`beep-test`

```
beep-test.beep
```

in the statechart corresponds in our model to

(*B_Enable_Beep_Test = true, D_Enable_Beep_Test = true*).

`chime/alarmX off, on`

These states are always synchronized with their respective enabled and disabled states and duplicate states are unnecessary; in the tables

Alarm2 = disabled

(for example) corresponds to, in Harel's model,

(`alarm2.off, alarm2-status.disabled`).

3 Statechart Errors

Consider the following scenario: Assume the watch is simply displaying the time and is shortly due to start beeping because the time of an (enabled) alarm is approaching. We press button 'b' and keep it pressed (`beep-test` enters state `10`); the time advances to the time of the alarm (`alarms-beep` is entered); we release button 'b' (no effect); if we press 'd' then the state returns to `regular.time, beep-test.10`, that is, according to `beep-test`, button 'b' is currently pressed rather than button 'd'. Pressing button 'b' now fails to activate the beep test. Alternatively, if button 'd' is released and pressed again then the beep test is incorrectly activated, that is, without 'b' being pressed at the same time. This is certainly a subtle error.

However, one may construct similar beep test scenarios, e.g., involving the state `wait`, which are merely dubious (a problem with the subtlety of Statecharts is divining whether the modeled behaviour is always what was intended). In general therefore no attempt is made to introduce corrections.

4 Statestep Model

4.1 Variables

The possible values for each variable (without interpretations) may be seen by looking at the groups of shaded cells in the constraint table below. Because the variables are generated by a mapping from the original statecharts as described in Section 2, we do not describe the meaning of each of the variable values here as would normally be done. Most should be self-explanatory in any case.

4.2 Constraint Table

Display	Update	Stopw_ Display	Stopw_ Run	Alarm1	Alarm2	Chime	Light	B	B_Enable_ Beep_Test	D_Enable_ Beep_Test	Power
alarm-_1	none time-_1min time-_10min time-hour	*	*	*	*	*	*	*	false	false	*
alarm-_2	none time-_1min time-_10min time-hour	*	*	*	*	*	*	*	false	false	*
chime	none	*	*	*	*	*	*	*	false	false	*
date	none	*	*	*	*	*	*	*	*	*	*
stopw	none	*	*	*	*	*	*	*	false	false	*
time	*	*	*	*	*	*	*	*	*	*	*
time	cal-date	*	*	*	*	*	*	*	*	*	*
time	cal-day	*	*	*	*	*	*	*	*	*	*
time	cal-mon	*	*	*	*	*	*	*	*	*	*
time	cal-year	*	*	*	*	*	*	*	*	*	*
time	mode	*	*	*	*	*	*	*	*	*	*
*	none	*	*	*	*	*	*	*	*	*	*

Display	Update	Stopw_ Display	Stopw_ Run	Alarm1	Alarm2	Chime	Light	B	B_Enable_ Beep_Test	D_Enable_ Beep_Test	Power
time	pending	*	*	*	*	*	*	*	false	false	*
alarm-* time	time-_1min	*	*	*	*	*	*	*	*	*	*
alarm-* time	time-_10min	*	*	*	*	*	*	*	*	*	*
alarm-* time	time-hour	*	*	*	*	*	*	*	*	*	*
time	time-sec	*	*	*	*	*	*	*	*	*	*
*	*	lap	*	*	*	*	*	*	*	*	*
*	*	regular	*	*	*	*	*	*	*	*	*
*	*	*	running	*	*	*	*	*	*	*	*
*	*	*	stopped	*	*	*	*	*	*	*	*
*	*	*	*	beeping	*	*	*	*	*	*	*
*	*	*	*	quiet-disabled	*	*	*	*	*	*	*
*	*	*	*	quiet-enabled	*	*	*	*	*	*	*
*	*	*	*	*	beeping	*	*	*	*	*	*
*	*	*	*	*	quiet-disabled	*	*	*	*	*	*
*	*	*	*	*	quiet-enabled	*	*	*	*	*	*
*	*	*	*	*	*	disabled	*	*	*	*	*
*	*	*	*	*	*	enabled-beeping	*	*	*	*	*
*	*	*	*	*	*	enabled-quiet	*	*	*	*	*
*	*	*	*	*	*	*	off	released	*	*	*
*	*	*	*	*	*	*	on	pressed	*	*	*
*	*	*	*	*	*	*	on	pressed	*	*	*
*	*	*	*	*	*	*	off	released	*	*	*
*	*	*	*	*	*	*	*	*	false	*	*

Display	Update	Stopw_ Display	Stopw_ Run	Alarm1	Alarm2	Chime	Light	B	B_Enable_ Beep_Test	D_Enable_ Beep_Test	Power
date time	cal-* mode none time-*	*	*	*	*	*	*	*	true	*	*
*	*	*	*	*	*	*	*	*	*	false	*
date time	cal-* mode none time-*	*	*	*	*	*	*	*	*	true	*
*	*	*	*	*	*	*	*	*	*	*	blink
*	*	*	*	*	*	*	*	*	*	*	ok

4.3 Events

The following table gives the events which may cause a change in the state of the watch, together with any additional constraints that apply on the occurrence of each event. Notes:

1. The constraints for an event are not necessarily comprehensive, but are sufficient to prevent any incompleteness errors being reported on the transition rules for that event.

2. In models of this kind, it is conventional to assume a total ordering on events, i.e., that two events never occur simultaneously. This greatly simplifies the model and is usually a reasonable assumption: in the real world, we cannot establish absolute simultaneity of independent events in any case, only that they occurred within some finite interval of time, the length of which depends on the accuracy of imperfect measurements. In such a case, we may reasonably assign an arbitrary ordering to the events. The problem with the watch is that some events cannot occur at any point on a continuum but only at logically synchronized instants at intervals of seconds, minutes or hours, e.g., alarm1_time_reached may occur at the same ideal point in time as whole_hour_reached. To address this, one might replace these events with a single event which occurred every second and check a variety of conditions on its occurrence, or perhaps define priorities for the events to determine the order in which simultaneous events were responded to.

Event	Description / Constraints
a_pressed	Button ‘a’ is pressed on the side of the watch.
a_released	Button ‘a’ is released.
b_pressed	
b_released	
c_pressed	
c_released	
d_pressed	
d_released	
update_pending_timeout	This occurs when, continuously for the preceding 2 seconds, Update has been in state pending and Alarm1 and Alarm2 have both been quiet. Constraint: `Update = { pending }`
update_timeout	Continuously for the preceding 2 minutes: Display has been in state time and Update has been (time | cal | mode) and Alarm1 and Alarm2 have both been quiet.
chime_alarm_display_timeout	Continuously for the preceding 2 minutes: Display has been (chime | alarm) and the times of the alarms have not changed and Chime has not changed to or from disabled and Alarm1, Alarm2 have both been quiet and have not changed state.
date_display_timeout	Continuously for the preceding 2 minutes: Display has been date; Alarm1 and Alarm2 have both been quiet.
alarm1_time_reached	The time has just changed to the time set for the first alarm. Constraint: `Alarm1 = { quiet-* }`
alarm2_time_reached	Constraint: `Alarm2 = { quiet-* }`
alarm_beep_timeout	This occurs when Alarm1 or Alarm2 (or both) has been beeping for 30s. Constraint: `Alarm1 = { beeping } || Alarm2 = { beeping }`
whole_hour_reached	The time has just changed to “*xx*:00:00”.
chime_timeout	Chime has been enabled-beeping for 2s.
battery_weakens	

4.4 Transition Table

Display	Update	Stopw_ Display	Stopw_ Run	Alarm1	Alarm2	Chime	Light	B	B_Enable_ Beep_Test	D_Enable_ Beep_Test	Power
a_pressed _1											
*	*	*	*	beeping	beeping	*	*	*	*	*	*
=	=	=	=	quiet-enabled	quiet-enabled	=	=	=	=	=	=
a_pressed _2											
*	*	*	*	quiet-*	beeping	*	*	*	*	*	*
=	=	=	=	=	quiet-enabled	=	=	=	=	=	=
a_pressed _3											
*	*	*	*	beeping	quiet-*	*	*	*	*	*	*
=	=	=	=	quiet-enabled	=	=	=	=	=	=	=
a_pressed _4											
alarm-* time	cal-* mode pending time-*	*	*	quiet-*	quiet-*	*	*	*	*	*	*
=	=	=	=	=	=	=	=	=	=	=	=
a_pressed _5											
time	none	*	*	quiet-*	quiet-*	*	*	*	*	*	*
alarm-_1	=	=	=	=	=	=	=	=	false	false	=
a_pressed _6											
alarm-_1	none	*	*	quiet-*	quiet-*	*	*	*	false	false	*
alarm-_2	=	=	=	=	=	=	=	=	=	=	=
a_pressed _7											
alarm-_2	none	*	*	quiet-*	quiet-*	*	*	*	false	false	*
chime	=	=	=	=	=	=	=	=	=	=	=

Display	Update	Stopw_ Display	Stopw_ Run	Alarm1	Alarm2	Chime	Light	B	B_Enable_ Beep_Test	D_Enable_ Beep_Test	Power
a_pressed _8											
chime	none	*	*	quiet-*	quiet-*	*	*	*	false	false	*
stopw	=	=	=	=	=	=	=	=	=	=	=
a_pressed _9											
stopw	none	*	*	quiet-*	quiet-*	*	*	*	false	false	*
time	=	=	=	=	=	=	=	=	=	=	=
a_pressed _10											
date	none	*	*	quiet-*	quiet-*	*	*	*	*	*	*
=	=	=	=	=	=	=	=	=	=	=	=
a_released _1											
*	*	*	*	*	*	*	*	*	*	*	*
=	=	=	=	=	=	=	=	=	=	=	=
b_pressed _1											
*	*	*	*	beeping	beeping	*	*	*	*	*	*
=	=	=	=	quiet-enabled	quiet-enabled	=	on	pressed	=	=	=
b_pressed _2											
*	*	*	*	quiet-*	beeping	*	*	*	*	*	*
=	=	=	=	=	quiet-enabled	=	on	pressed	=	=	=
b_pressed _3											
*	*	*	*	beeping	quiet-*	*	*	*	*	*	*
=	=	=	=	quiet-enabled	=	=	on	pressed	=	=	=
b_pressed _4											
time	pending	*	*	quiet-*	quiet-*	*	*	*	false	false	*
=	=	=	=	=	=	=	on	pressed	=	=	=

Display	Update	Stopw_ Display	Stopw_ Run	Alarm1	Alarm2	Chime	Light	B	B_Enable_ Beep_Test	D_Enable_ Beep_Test	Power
b_pressed _5											
date time	cal-* mode none time-*	*	*	quiet-*	quiet-*	*	*	*	*	*	*
=	none	=	=	=	=	=	on	pressed	true	=	=
b_pressed _6											
alarm-* chime	*	*	*	quiet-*	quiet-*	*	*	*	false	false	*
=	none	=	=	=	=	=	on	pressed	=	=	=
b_pressed _7											
stopw	none	*	running	quiet-*	quiet-*	*	*	*	false	false	*
=	=	=	stopped	=	=	=	on	pressed	=	=	=
b_pressed _8											
stopw	none	*	stopped	quiet-*	quiet-*	*	*	*	false	false	*
=	=	=	running	=	=	=	on	pressed	=	=	=
b_released _1											
*	*	*	*	beeping	*	*	*	*	*	*	*
=	=	=	=	=	=	=	off	released	=	=	=
b_released _2											
*	*	*	*	quiet-*	beeping	*	*	*	*	*	*
=	=	=	=	=	=	=	off	released	=	=	=
b_released _3											
alarm-* chime stopw	*	*	*	quiet-*	quiet-*	*	*	*	false	false	*
=	=	=	=	=	=	=	off	released	=	=	=

Display	Update	Stopw_ Display	Stopw_ Run	Alarm1	Alarm2	Chime	Light	B	B_Enable_ Beep_Test	D_Enable_ Beep_Test	Power
b_released _4											
time	pending	*	*	quiet-*	quiet-*	*	*	*	false	false	*
=	=	=	=	=	=	=	off	released	=	=	=
b_released _5											
date time	cal-* mode none time-*	*	*	quiet-*	quiet-*	*	*	*	*	*	*
=	=	=	=	=	=	=	off	released	false	=	=
c_pressed _1											
*	*	*	*	beeping	beeping	*	*	*	*	*	*
=	=	=	=	quiet-enabled	quiet-enabled	=	=	=	=	=	=
c_pressed _2											
*	*	*	*	quiet-*	beeping	*	*	*	*	*	*
=	=	=	=	=	quiet-enabled	=	=	=	=	=	=
c_pressed _3											
*	*	*	*	beeping	quiet-*	*	*	*	*	*	*
=	=	=	=	quiet-enabled	=	=	=	=	=	=	=
c_pressed _4											
chime stopw	none	*	*	quiet-*	quiet-*	*	*	*	false	false	*
=	=	=	=	=	=	=	=	=	=	=	=
c_pressed _5											
time	none pending	*	*	quiet-*	quiet-*	*	*	*	*	*	*
=	pending	=	=	=	=	=	=	=	false	false	=
(Yes, button c can be pressed while Update is pending - since it can be released without Update being changed when an alarm is beeping.)											

Display	Update	Stopw_ Display	Stopw_ Run	Alarm1	Alarm2	Chime	Light	B	B_Enable_ Beep_Test	D_Enable_ Beep_Test	Power
c_pressed _6											
time	time-sec	*	*	quiet-*	quiet-*	*	*	*	*	*	*
=	time-_1min	=	=	=	=	=	=	=	=	=	=
c_pressed _7											
time	time-_1min	*	*	quiet-*	quiet-*	*	*	*	*	*	*
=	time-_10min	=	=	=	=	=	=	=	=	=	=
c_pressed _8											
time	time-_10min	*	*	quiet-*	quiet-*	*	*	*	*	*	*
=	time-hour	=	=	=	=	=	=	=	=	=	=
c_pressed _9											
time	time-hour	*	*	quiet-*	quiet-*	*	*	*	*	*	*
=	cal-mon	=	=	=	=	=	=	=	=	=	=
c_pressed _10											
time	cal-mon	*	*	quiet-*	quiet-*	*	*	*	*	*	*
=	cal-date	=	=	=	=	=	=	=	=	=	=
c_pressed _11											
time	cal-date	*	*	quiet-*	quiet-*	*	*	*	*	*	*
=	cal-day	=	=	=	=	=	=	=	=	=	=
c_pressed _12											
time	cal-day	*	*	quiet-*	quiet-*	*	*	*	*	*	*
=	cal-year	=	=	=	=	=	=	=	=	=	=
c_pressed _13											
time	cal-year	*	*	quiet-*	quiet-*	*	*	*	*	*	*
=	mode	=	=	=	=	=	=	=	=	=	=
c_pressed _14											
time	mode	*	*	quiet-*	quiet-*	*	*	*	*	*	*
=	none	=	=	=	=	=	=	=	=	=	=

Display	Update	Stopw_ Display	Stopw_ Run	Alarm1	Alarm2	Chime	Light	B	B_Enable_ Beep_Test	D_Enable_ Beep_Test	Power
c_pressed _15											
alarm-*	none	*	*	quiet-*	quiet-*	*	*	*	false	false	*
=	time-hour	=	=	=	=	=	=	=	=	=	=
c_pressed _16											
alarm-*	time-hour	*	*	quiet-*	quiet-*	*	*	*	false	false	*
=	time-_10min	=	=	=	=	=	=	=	=	=	=
c_pressed _17											
alarm-*	time-_10min	*	*	quiet-*	quiet-*	*	*	*	false	false	*
=	time-_1min	=	=	=	=	=	=	=	=	=	=
c_pressed _18											
alarm-*	time-_1min	*	*	quiet-*	quiet-*	*	*	*	false	false	*
=	none	=	=	=	=	=	=	=	=	=	=
c_pressed _19											
date	none	*	*	quiet-*	quiet-*	*	*	*	*	*	*
=	=	=	=	=	=	=	=	=	=	=	=
c_released _1											
*	*	*	*	beeping	*	*	*	*	*	*	*
=	=	=	=	=	=	=	=	=	=	=	=
c_released _2											
*	*	*	*	quiet-*	beeping	*	*	*	*	*	*
=	=	=	=	=	=	=	=	=	=	=	=
c_released _3											
time	pending	*	*	quiet-*	quiet-*	*	*	*	false	false	*
=	none	=	=	=	=	=	=	=	=	=	=

Display	Update	Stopw_ Display	Stopw_ Run	Alarm1	Alarm2	Chime	Light	B	B_Enable_ Beep_Test	D_Enable_ Beep_Test	Power
c_released _4											
*	cal-* mode none time-*	*	*	quiet-*	quiet-*	*	*	*	*	*	*
=	=	=	=	=	=	=	=	=	=	=	=
d_pressed _1											
*	*	*	*	beeping	beeping	*	*	*	*	*	*
=	=	=	=	quiet-enabled	quiet-enabled	=	=	=	=	=	=
d_pressed _2											
*	*	*	*	quiet-*	beeping	*	*	*	*	*	*
=	=	=	=	=	quiet-enabled	=	=	=	=	=	=
d_pressed _3											
*	*	*	*	beeping	quiet-*	*	*	*	*	*	*
=	=	=	=	quiet-enabled	=	=	=	=	=	=	=
d_pressed _4											
time	pending	*	*	quiet-*	quiet-*	*	*	*	false	false	*
=	=	=	=	=	=	=	=	=	=	=	=
d_pressed _5											
time	none	*	*	quiet-*	quiet-*	*	*	released	*	*	*
date	=	=	=	=	=	=	=	=	=	true	=
d_pressed _6											
time	none	*	*	quiet-*	quiet-*	*	*	pressed	*	*	*
=	=	=	=	=	=	=	=	=	=	true	=
d_pressed _7											
date	none	*	*	quiet-*	quiet-*	*	*	*	*	*	*
time	=	=	=	=	=	=	=	=	=	true	=

Display	Update	Stopw_ Display	Stopw_ Run	Alarm1	Alarm2	Chime	Light	B	B_Enable_ Beep_Test	D_Enable_ Beep_Test	Power
d_pressed_8											
time	cal-* mode time-*	*	*	quiet-*	quiet-*	*	*	*	*	*	*
=	=	=	=	=	=	=	=	=	=	true	=
The current value of the field indicated by Update is incremented with wraparound or toggled as appropriate.											
d_pressed_9											
alarm-*	time-*	*	*	quiet-*	quiet-*	*	*	*	false	false	*
=	=	=	=	=	=	=	=	=	=	=	=
The current value of the field indicated by Update is incremented with wraparound.											
d_pressed_10											
alarm-_1	none	*	*	quiet-disabled	quiet-*	*	*	*	false	false	*
=	=	=	=	quiet-enabled	=	=	=	=	=	=	=
d_pressed_11											
alarm-_1	none	*	*	quiet-enabled	quiet-*	*	*	*	false	false	*
=	=	=	=	quiet-disabled	=	=	=	=	=	=	=
d_pressed_12											
alarm-_2	none	*	*	quiet-*	quiet-disabled	*	*	*	false	false	*
=	=	=	=	=	quiet-enabled	=	=	=	=	=	=
d_pressed_13											
alarm-_2	none	*	*	quiet-*	quiet-enabled	*	*	*	false	false	*
=	=	=	=	=	quiet-disabled	=	=	=	=	=	=
d_pressed_14											
chime	none	*	*	quiet-*	quiet-*	disabled	*	*	false	false	*
=	=	=	=	=	=	enabled-quiet	=	=	=	=	=
d_pressed_15											
chime	none	*	*	quiet-*	quiet-*	enabled-*	*	*	false	false	*
=	=	=	=	=	=	disabled	=	=	=	=	=

Display	Update	Stopw_ Display	Stopw_ Run	Alarm1	Alarm2	Chime	Light	B	B_Enable_ Beep_Test	D_Enable_ Beep_Test	Power
d_pressed _16											
stopw	none	regular	stopped	quiet-*	quiet-*	*	*	*	false	false	*
=	=	=	=	=	=	=	=	=	=	=	=
The time of the stopwatch is reset to zero.											
d_pressed _17											
stopw	none	regular	running	quiet-*	quiet-*	*	*	*	false	false	*
=	=	lap	=	=	=	=	=	=	=	=	=
d_pressed _18											
stopw	none	lap	*	quiet-*	quiet-*	*	*	*	false	false	*
=	=	regular	=	=	=	=	=	=	=	=	=
d_released _1											
*	*	*	*	beeping	*	*	*	*	*	*	*
=	=	=	=	=	=	=	=	=	=	=	=
d_released _2											
*	*	*	*	quiet-*	beeping	*	*	*	*	*	*
=	=	=	=	=	=	=	=	=	=	=	=
d_released _3											
alarm-* chime stopw	*	*	*	quiet-*	quiet-*	*	*	*	false	false	*
=	=	=	=	=	=	=	=	=	=	=	=
d_released _5											
date time	*	*	*	quiet-*	quiet-*	*	*	*	*	*	*
=	=	=	=	=	=	=	=	=	=	false	=
update_pending_timeout _1											
time	pending	*	*	*	*	*	*	*	false	false	*
=	time-sec	=	=	=	=	=	=	=	=	=	=

Display	Update	Stopw_ Display	Stopw_ Run	Alarm1	Alarm2	Chime	Light	B	B_Enable_ Beep_Test	D_Enable_ Beep_Test	Power
update_timeout _1											
*	*	*	*	*	*	*	*	*	*	*	*
=	none	=	=	=	=	=	=	=	=	=	=
chime_alarm_display_timeout _1											
*	*	*	*	*	*	*	*	*	*	*	*
time	=	=	=	=	=	=	=	=	=	=	=
date_display_timeout _1											
*	*	*	*	*	*	*	*	*	*	*	*
time	=	=	=	=	=	=	=	=	=	=	=
alarm1_time_reached _1											
*	*	*	*	quiet-disabled	*	*	*	*	*	*	*
=	=	=	=	=	=	=	=	=	=	=	=
alarm1_time_reached _2											
*	*	*	*	quiet-enabled	*	*	*	*	*	*	*
=	=	=	=	beeping	=	=	=	=	=	=	=
alarm2_time_reached _1											
*	*	*	*	*	quiet-disabled	*	*	*	*	*	*
=	=	=	=	=	=	=	=	=	=	=	=
alarm2_time_reached _2											
*	*	*	*	*	quiet-enabled	*	*	*	*	*	*
=	=	=	=	=	beeping	=	=	=	=	=	=
alarm_beep_timeout _1											
*	*	*	*	beeping	beeping	*	*	*	*	*	*
=	=	=	=	quiet-enabled	quiet-enabled	=	=	=	=	=	=
alarm_beep_timeout _2											
*	*	*	*	quiet-*	beeping	*	*	*	*	*	*
=	=	=	=	=	quiet-enabled	=	=	=	=	=	=

Display	Update	Stopw_ Display	Stopw_ Run	Alarm1	Alarm2	Chime	Light	B	B_Enable_ Beep_Test	D_Enable_ Beep_Test	Power
alarm_beep_timeout _3											
*	*	*	*	beeping	quiet-*	*	*	*	*	*	*
=	=	=	=	quiet-enabled	=	=	=	=	=	=	=
whole_hour_reached _1											
*	*	*	*	*	*	disabled	*	*	*	*	*
=	=	=	=	=	=	=	=	=	=	=	=
whole_hour_reached _2											
*	*	*	*	*	*	enabled-*	*	*	*	*	*
=	=	=	=	=	=	enabled-beeping	=	=	=	=	=
chime_timeout _1											
*	*	*	*	*	*	*	*	*	*	*	*
=	=	=	=	=	=	enabled-quiet	=	=	=	=	=
battery_weakens _1											
*	*	*	*	*	*	*	*	*	*	*	*
=	=	=	=	=	=	=	=	=	=	=	blink