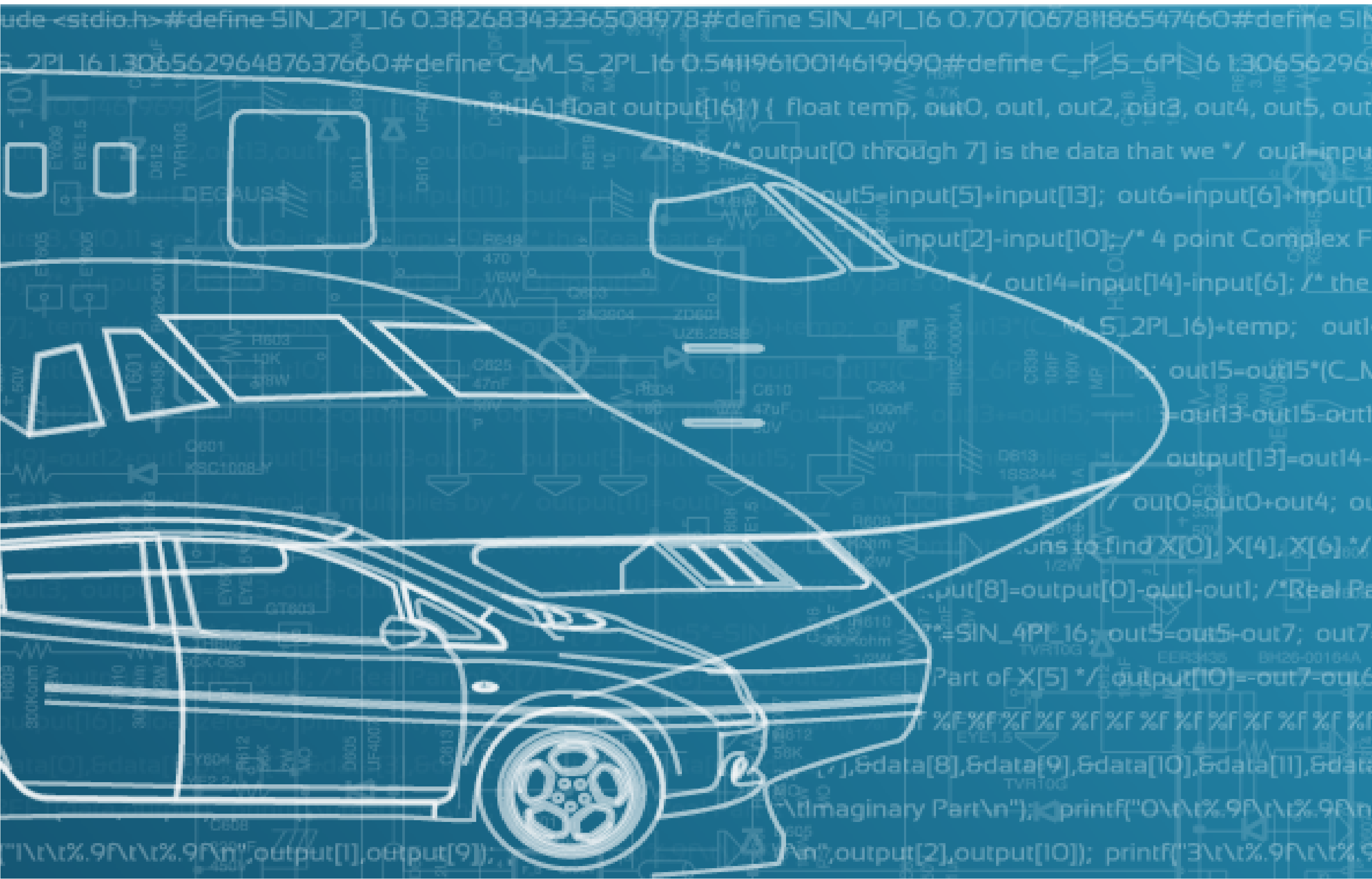
ASTERIOS

KRONO-SAFE

Safe design in real-time

LED Blinker

Rev K19.5



HOW TO CONTACT US

Corporate headquarters	**KRONO-SAFE** Support 16 avenue Carnot, F - 91300 MASSY
Support contact	Customer Care Portal

VERSION

Document	LED Blinker
Revision	K19.5
Date	May 16, 2024

COPYRIGHT

Contents

This example is an introduction to the cadencing of Agents in PsyC.

Keywords: `advance`, `converttick`, `presenttime`

1 Specification

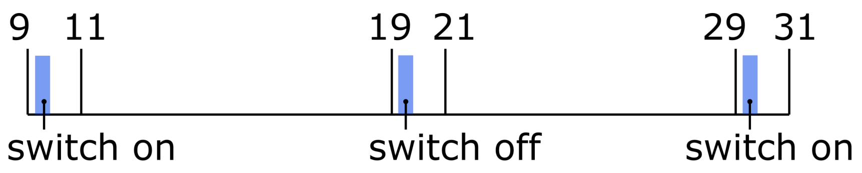
We want to write an Agent in charge of making blink a LED at a frequency of 50 Hz (period of 20 ms), such that over the course of each period the LED remains on for 10 ms, with a precision of +/- 2 ms.

The jitter of +/- 2 ms is critical for a consistent temporal design of the Application: a requirement stating that the LED has to be on (respectively off) for *exactly* 10 ms every period would make no sense physically. The Psy programming language enables you to formally specify the acceptable jitter for switching the LED on and off.

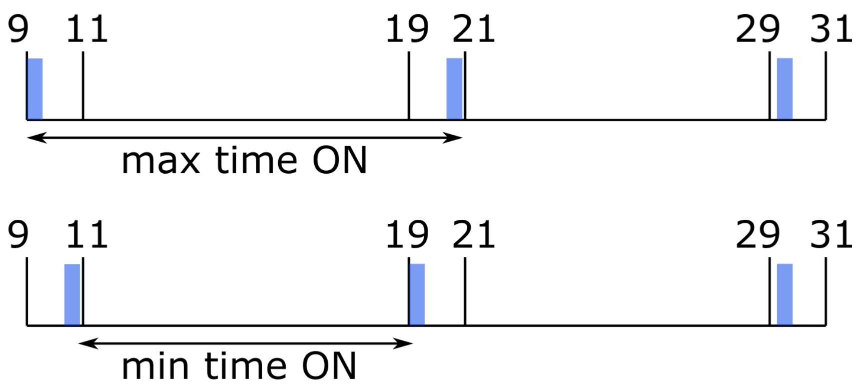
2 Implementation

The cadencing of Agents i.e. the definition of their Elementary Actions is based on the use of the `advance` statement. The syntax of this statement is detailed in the *PsyC Language Description* in the section *Modules and Tasks - Agents - advance*.

The behavior of the Agent `blinker` implemented in `blinker.psy` is represented in the following:



This implementation respects the above specification. The following figure illustrates it on two possible execution scenarios:



Depending on *when* the switching on (respectively off) will actually be executed within the 2 ms Elementary Action, the total time on (respectively off) of the LED will vary between 12 and 8 ms, as required.

3 `convertick` and `presenttime`

In this example, we use these two built-ins to retrieve the current date, independently from the Source on which the Agent is based.

`presenttime` returns the current date of the Elementary Action, as a number of tick of the Source `realtime` from the start of the Application. *The current date of an Elementary Action is always defined as the earliest start date of the Elementary Action*.

The fact that this value returned by `presenttime` depends on the period of the Source `realtime` makes the direct use of this value not portable. The `converttick (<clock>, <date>)` built-in enables to retrieve the index of the closest tick of the Clock `<clock>` preceding `<date>` (in Source ticks).

Therefore, `converttick(c1, presenttime())` returns the index of the tick of `c1` matching the Earliest Start Date of the current Elementary Action opened by the last `advance` statement met.