Ellidiss
www.ellidiss.com

AADL analysis documentation

Results produced by AADL Inspector

Table of contents

1. Instance Model

The instance model is automatically deduced from the declarative model specified in the file Pacemaker.aadl generated by Stood.

```
-------------------------------------------------
aadlrev2.9 (c)Ellidiss Technologies 17Mar2016
AADL-2.1 + BA-2.0 + EMV2-1.0
-------------------------------------------------
            *** INSTANCE MODEL ***
-------------------------------------------------
Root System Instance: Pacemaker_Pkg::Pacemaker.others
-------------------------------------------------
   22 (system)..... root
   78 (processor)... root.DCM.HWPlatform (HPF)
   77 (process)..... root.DCM.PacemakerSW
  114 (thread)..... VRPTimeout (TIMED)
  118 (thread)..... LRLTimeout (TIMED)
  121 (thread)..... VVIMode (APERIODIC)
-------------------------------------------------
Number of Component Instances:
-------------------------------------------------
- Processors:  1
- Processes:   1
- Threads:     3
- Subprograms: 0
-------------------------------------------------
```

2. Simulation

Tests are performed with the Marzhin event based simulator that emulates the AADL run-time. The simulation operates on the AADL instance model and can be controlled either by user interaction or by automated scenario that are formalized by a dedicated XML model.

Simulation scenarii have been defined to express the test cases. The testing interface is specified in terms of AADL instance model entities that can receive input stimulations. For the Pacemaker example, the only one used entry point is the s event port of the VVIMode thread.

```
<interface>
    <event lmpId="sense" id="dcm.hwplatform.dcm.pacemakersw.vvimode.s"/>
</interface>
```

Note that for the purpose of the simulation, all the time constants have been divided by 10 (i.e: index 100 represents 1000ms)

2.1 Test 1

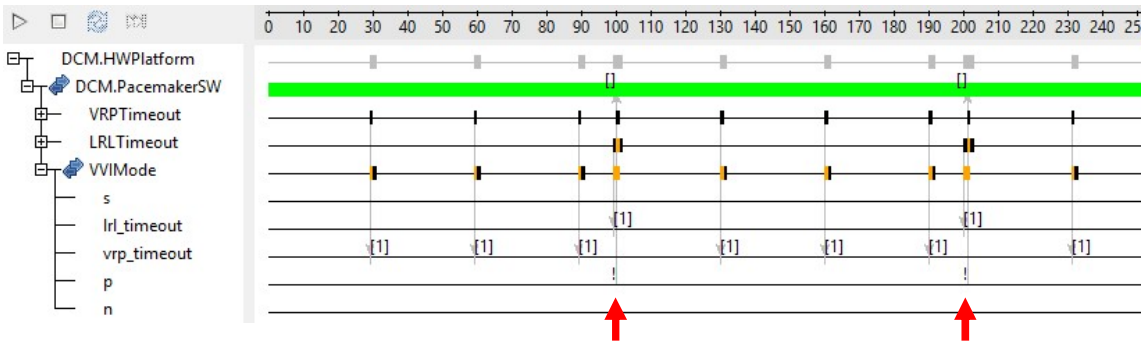
Name: No sensing.

Description:
The thread will put out an event on the "p" port every 1000 ms.

Scenario:

```
<scenario name="test 1" description="No sensing">
</scenario>
```

Simulation trace:



Observation:
A “p” event has been generated every 1000ms

2.2 Test 2

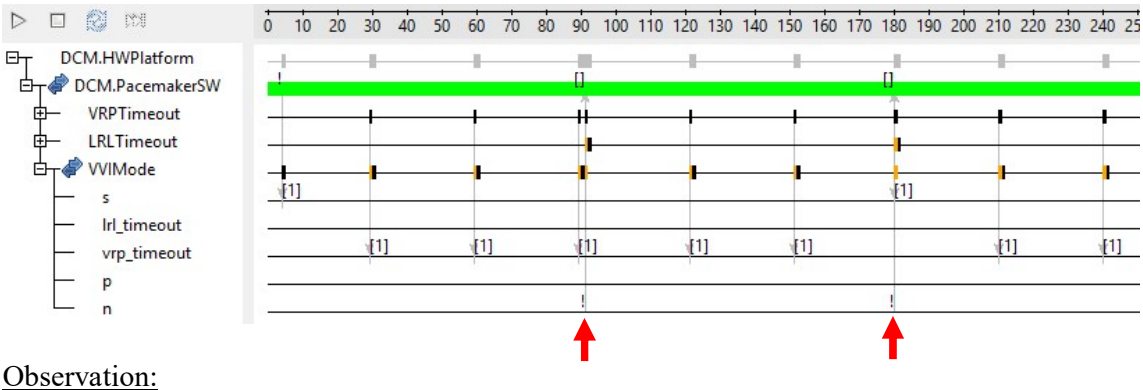
Name: Normal rhythm.

Description:
Put an event on the "s" port every 900 ms. The thread will put an event out the "n" port each dispatch.

Scenario:

```
<scenario name="test 2" description="Normal rhythm">
    <action nextTick="tick+90" tick="0" value="1" >
        <feature lmpRef="sense"/>
    </action>
</scenario>
```

Simulation trace:



Observation:
A “n” event has been generated every 900ms

2.3 Test 3

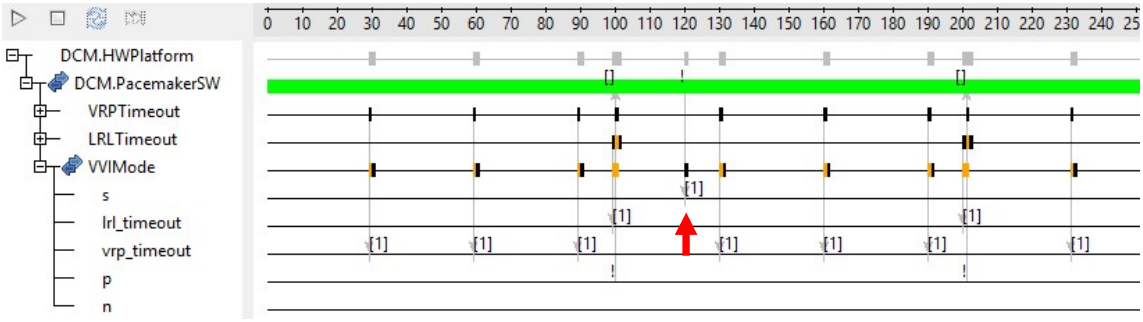
Name: Ignore sense in VRP.

Description:
Wait 1000 ms for the first pace; 200 ms later put an event on the "s" port. The next pace will occur at 2000 ms.

Scenario:

```
<scenario name="test 3" description="Ignore sense in VRP">
    <action tick="120" value="1">
        <feature lmpRef="sense"/>
    </action>
</scenario>
```

Simulation trace:



Observation:
The s event at 1200ms has been ignored

2.4 Test 4

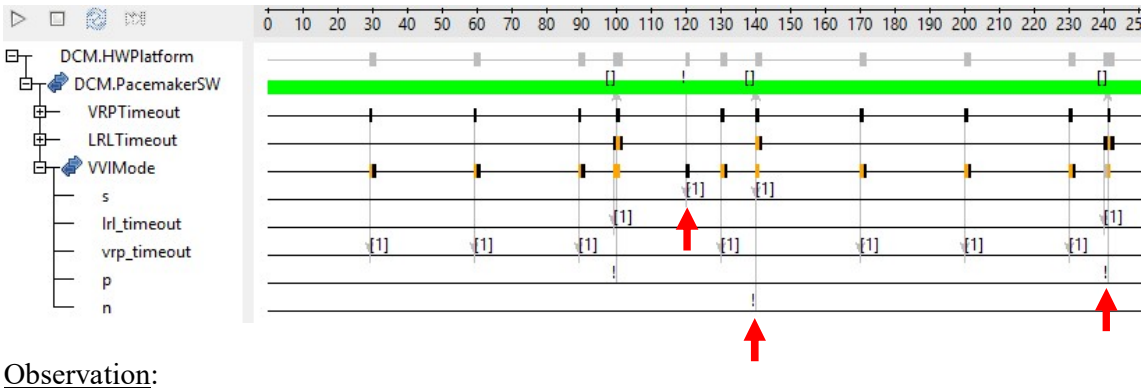
Name: Pace after sense.

Description:
Wait 1000 ms for the first pace; 200 ms later put an event on the "s" port, which will be ignored.
At 1400 ms put out another event on the "s" port. Expect the next pace at 2400 ms.

Scenario:

```
<scenario name="test 4" description="Pace after sense">
    <action tick="120" value="1">
        <feature lmpRef="sense"/>
    </action>
    <action tick="140" value="1">
        <feature lmpRef="sense"/>
    </action>
</scenario>
```

Simulation trace:



Observation:
The s event at 1200ms has been ignored
The s event at 1400ms has generated a n event
A p event has been generated at 2400ms