A Sample Document for the Usages of lstEventB Package

Thai Son Hoang
ECS, University of Southampton
<T dot S dot Hoang at ecs dot soton dot ac dot uk>

August 10, 2017

For convenient, we define macro `\eventB` for Event-B.

We start first with some inline Event-B code by embedding them using a pair of `|`, for example `|@grd1: "SNSR = FALSE"|` gives |@grd1: "SNSR = FALSE"|. Any Event-B formulae including Unicode symbols will be typeset using the `bsymb` package accordingly.

More complete piece of code (including the Unicode symbols) can be typeset using the `EventBcode` environment. Below is the typesetting of an Event-B machine.

```
machine Sensor_m0_SNSR
variables
  SNSR
invariants
  @thm0_1: "SNSR ∈ BOOL" theorem
events

  INITIALISATION
  begin
    @act1: "SNSR := FALSE"
  end

  SNSR_on
  when
    @grd1: "SNSR = FALSE"
  then
    @act1: "SNSR := TRUE"
  end

  SNSR_off
  when
    @grd1: "SNSR = TRUE"
  then
    @act1: "SNSR := FALSE"
  end

end
```

One can includes external file containing Event-B code using the `\EventBinputlisting` command. For example the following is the result of including the code in the file `Sensor_m1_DEP.bumx` using `\EventBinputlisting{Sensor_m1_DEP.bumx}`.

```
machine Sensor_m1_DEP
refines Sensor_m0_SNSR
variables
  SNSR
  DEP
invariants
  @inv0_1: "DEP ∈ ℕ"
events

  INITIALISATION extended
  begin
    @act2: "DEP := 0"
  end

  SNSR_on extended
  refines SNSR_on
  end

  SNSR_off extended
  refines SNSR_off
  begin
    @act2: "DEP := DEP + 1"
  end

end
```

More specifically, one can specify more details on the inclusion, e.g., the ranges, as the following example
`\EventBinputlisting[firstline=16,lastline=20]{Sensor_m2_snsr.bumx}`
gives

```
  @inv1_1: "Snsr_01 = TRUE ⇒SNSR = TRUE"

  @inv1_2: "Snsr_10 = TRUE ⇒SNSR = FALSE"

  @inv1_3: "Snsr_01 = FALSE  Snsr_10 = FALSE"
```

```
machine Sensor_m3_Ctrl

refines

  Sensor_m2_Snsr

variables

  SNSR
```

```
10
11    DEP
12
13    Snsr_01
14
15    Snsr_10
16
17    ctrl_snsr
18
19    ctrl_dep
20
21    ctrl_snsr_01
22
23    ctrl_snsr_10
24
25  invariants
26
27    @inv2_1:
28    "Snsr_01 = FALSE ∧Snsr_10 = FALSE ∧ctrl_snsr_01 = FALSE ∧ctrl_snsr_10 =
          FALSE ⇒ctrl_snsr = SNSR"
29
30    @inv2_2: "ctrl_dep ∈ ℕ"
31
32    @inv2_3: "Snsr_10 = FALSE ∧ctrl_snsr_10 = FALSE ⇒ctrl_dep = DEP"
33
34    @inv2_4: "Snsr_10 = TRUE  ctrl_snsr_10 = TRUE ⇒ctrl_dep = DEP  1"
35
36    @inv2_5: "ctrl_snsr_01 = TRUE ⇒SNSR = TRUE"
37
38    @inv2_6: "ctrl_snsr_10 = TRUE ⇒SNSR = FALSE"
39
40    @inv2_7: "ctrl_snsr_01 = TRUE ⇒Snsr_01 = FALSE"
41
42    @inv2_8: "ctrl_snsr_10 = TRUE ⇒Snsr_10 = FALSE"
43
44  events
45
46    INITIALISATION extended
47    refines INITIALISATION
48    begin
49      @act5: "ctrl_snsr := FALSE"
50      @act6: "ctrl_dep := 0"
51      @act7: "ctrl_snsr_01 := FALSE"
52      @act8: "ctrl_snsr_10 := FALSE"
53    end
54
55    SNSR_on extended
56    refines SNSR_on
57    when
58      @grd3: "ctrl_snsr_10 = FALSE"
59    end
60
61    SNSR_off extended
62    refines SNSR_off
63    when
64      @grd3: "ctrl_snsr_01 = FALSE"
```

```
65    end
66
67    ctrl_Senses_Snsr_01 extended
68    refines ctrl_Senses_Snsr_01
69    begin
70      @act2: "ctrl_snsr_01 := TRUE"
71    end
72
73    ctrl_Senses_Snsr_10 extended
74    refines ctrl_Senses_Snsr_10
75    begin
76      @act2: "ctrl_snsr_10 := TRUE"
77    end
78
79    ctrl_on
80    when
81      @grd1: "ctrl_snsr_01 = TRUE"
82    then
83      @act1: "ctrl_snsr_01 := FALSE"
84      @act2: "ctrl_snsr := TRUE"
85    end
86
87    ctrl_off
88    when
89      @grd1: "ctrl_snsr_10 = TRUE"
90    then
91      @act1: "ctrl_snsr_10 := FALSE"
92      @act2: "ctrl_snsr := FALSE"
93      @act3: "ctrl_dep := ctrl_dep + 1"
94    end
95
96  end
```