

# 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>

May 13, 2018

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.

ASCII	Symbols	Explanation
<code>!false</code>	$\perp$	False
<code>!true</code>	$\top$	True
<code>&amp;</code>	$\wedge$	Conjunction
<code>!or</code>	$\vee$	Disjunction
<code>=&gt;</code>	$\Rightarrow$	Implication
<code>&lt;=&gt;</code>	$\Leftrightarrow$	Equivalence
<code>!not</code>	$\neg$	Negation
<code>!</code>	$\forall$	Universal quantification
<code>#</code>	$\exists$	Existential quantification
<code>.</code>	$\cdot$	Quantification dot
<code>=</code>	$=$	Equality
<code>/=</code>	$\neq$	Inequality

Table 1: Predicates

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.

---

```
1 machine Sensor_m0_SNSR
2 variables
3   SNSR
4 invariants
5   @thm0_1  $\in$  "SNSR  $\in$  BOOL" theorem
6 events
7
```

ASCII	Symbols	Explanation
<code>{}</code>	$\emptyset$	Empty set
<code> </code>	$ $	Vertical bar, e.g., in set comprehension
<code>\</code>	$\cup$	Union
<code>\</code>	$\cap$	Intersection
<code>\</code>	$\setminus$	Set difference
<code>  -&gt;</code>	$\mapsto$	Ordered pair
<code>**</code>	$\times$	Cartesian product
<code>!POW</code>	$\mathbb{P}$	Powerset
<code>!POW1</code>	$\mathbb{P}_1$	Non-empty subsets
<code>!card</code>	card	Cardinality
<code>!union</code>	union	Generalised union
<code>!inter</code>	inter	Generalised intersection
<code>!UNION</code>	$\bigcup$	Quantified union
<code>!INTER</code>	$\bigcap$	Quantified intersection

Table 2: Sets

ASCII	Symbols	Explanation
<code>:</code>	$\in$	Set membership
<code>/:</code>	$\notin$	Set non-membership
<code>&lt;:</code>	$\subseteq$	Subset
<code>/&lt;:</code>	$\not\subseteq$	Not a subset
<code>&lt;&lt;:</code>	$\subset$	Proper subset
<code>/&lt;&lt;:</code>	$\not\subset$	Not a proper subset
<code>!finite</code>	finite	Finite
<code>!partition</code>	partition	Partition

Table 3: Set predicates

ASCII	Symbols	Explanation
<code>!BOOL</code>	BOOL	BOOL set
<code>!TRUE</code>	TRUE	TRUE
<code>!FALSE</code>	FALSE	FALSE
<code>!bool</code>	bool	bool predicate

Table 4: BOOL and bool

ASCII	Symbols	Explanation
<code>!INT</code>	$\mathbb{Z}$	Set of integer numbers
<code>!NAT</code>	$\mathbb{N}$	Set of natural numbers
<code>!NAT1</code>	$\mathbb{N}_1$	Set of positive natural numbers
<code>!min</code>	min	Minimum
<code>!max</code>	max	Maximum
<code>-</code>	$-$	Difference
<code>*</code>	$\times$	Product
<code>/</code>	$\div$	Quotient
<code>!mod</code>	mod	Remainder
<code>..</code>	$..$	Interval

Table 5: Numbers

ASCII	Symbols	Explanation
<code>&gt;</code>	$>$	Greater
<code>&lt;</code>	$<$	Less
<code>&gt;=</code>	$\geq$	Greater or equal
<code>&lt;=</code>	$\leq$	Less or equal

Table 6: Number predicates

ASCII	Symbols	Explanation
<code>&lt;-&gt;</code>	$\leftrightarrow$	Relations
<code>!dom</code>	dom	Domain
<code>!ran</code>	ran	Range
<code>&lt;&lt;-&gt;</code>	$\Leftrightarrow$	Total relations
<code>&lt;-&gt;&gt;</code>	$\Rrightarrow$	Surjective relations
<code>&lt;&lt;-&gt;&gt;</code>	$\Leftrightarrow$	Total surjective relations
<code>!circ</code>	$\circ$	Backward composition
<code>!id</code>	id	Identity
<code>&lt; </code>	$\triangleleft$	Domain restriction
<code>&lt;&lt; </code>	$\triangleleft$	Domain subtraction
<code> &gt;</code>	$\triangleright$	Range restriction
<code> &gt;&gt;</code>	$\triangleright$	Range subtraction
<code>~</code>	$-1$	Inverse
<code>&lt;+</code>	$\Leftarrow$	Overriding
<code>&gt;&lt;</code>	$\otimes$	Direct product
<code>  </code>	$\parallel$	Parallel product
<code>!prj1</code>	$\text{prj}_1$	First projection
<code>!prj2</code>	$\text{prj}_2$	Second projection

Table 7: Relations

ASCII	Symbols	Explanation
$\rightarrow$	$\rightarrow$	Partial functions
$\rightarrow$	$\rightarrow$	Total functions
$\rightarrow$	$\rightarrow$	Partial injections
$\rightarrow$	$\rightarrow$	Total injections
$\rightarrow$	$\rightarrow$	Partial surjections
$\rightarrow$	$\rightarrow$	Total surjections
$\rightarrow$	$\rightarrow$	Bijections
$\lambda$	$\lambda$	Lambda abstraction

Table 8: Functions

ASCII	Symbols	Explanation
$:=$	$:=$	Becomes equal to
$:$	$:$	Choice from a set
$:-$	$:-$	Choice by predicate

Table 9: Functions

```

8  INITIALISATION
9  begin
10   @act1 ∈ "SNSR := FALSE"
11  end
12
13  SNSR_on
14  when
15   @grd1 ∈ "SNSR = FALSE"
16  then
17   @act1 ∈ "SNSR := TRUE"
18  end
19
20  SNSR_off
21  when
22   @grd1 ∈ "SNSR = TRUE"
23  then
24   @act1 ∈ "SNSR := FALSE"
25  end
26
27 end

```

---

One can change the different colour options. For example, `\EventBSetKeywordColour{blue!50!black}` will change the keyword colour to dark blue. (This has effects only when

---

```

1  machine Sensor_m0_SNSR
2  variables
3  SNSR
4  invariants
5  @thm0_1 ∈ "SNSR ∈ BOOL" theorem

```

---

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}.

```
1 machine Sensor_m1_DEP
2 refines Sensor_m0_SNSR
3 variables
4   SNSR
5   DEP
6 invariants
7   @inv0_1 ∈ "DEP ∈ ℕ"
8 events
9
10 INITIALISATION extended
11 begin
12   @act2 ∈ "DEP := 0"
13 end
14
15 SNSR.on extended
16 refines SNSR.on
17 end
18
19 SNSR.off extended
20 refines SNSR.off
21 begin
22   @act2 ∈ "DEP := DEP + 1"
23 end
24
25 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

```
1 @inv1_1 ∈ "Snsr_01 = TRUE ⇒ SNSR = TRUE"
2
3 @inv1_2 ∈ "Snsr_10 = TRUE ⇒ SNSR = FALSE"
4
5 @inv1_3 ∈ "Snsr_01 = FALSE ∨ Snsr_10 = FALSE"
```

```
1 machine Sensor_m3_Ctrl
2
3 refines
4
5   Sensor_m2_Snsr
6
7 variables
8
9   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

```