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
`false`	$\bot$	False
`true`	$\top$	True
`&`	$\land$	Conjunction
`or`	$\lor$	Disjunction
`=>`	$\Rightarrow$	Implication
`<=>`	$\Leftrightarrow$	Equivalence
`not`	$\neg$	Negation
`!`	$\forall$	Universal quantification
`#`	$\exists$	Existential quantification
`.`	$\cdot$	Quantification dot
`/=`	$\neq$	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 Sens∨_m0_SNSR
2 variables
3   SNSR
4 invariants
5   @thm0_1∈ "SNSR ∈BOOL" the∨em
6 events
7
8   INITIALISATION
```

ASCII	Symbols	Explanation
`{}`	$\varnothing$	Empty set
`|`	$\mid$	Vertical bar, e.g., in set comprehension
`\/`	$\cup$	Union
`/\`	$\cap$	Intersection
`\`	$\setminus$	Set difference
`**`	$\times$	Cartesian product
`POW`	$\mathbb{P}$	Powerset
`POW1`	$\mathbb{P}_1$	Non-empty subsets
`card`	card	Cardinality
`union`	union	Generalised union
`inter`	inter	Generalised intersection
`UNION`	$\bigcup$	Quantified union
`INTER`	$\bigcap$	Quantified intersection

Table 2: Sets

ASCII	Symbols	Explanation
`:`	$\in$	Set membership
`/:`	$\notin$	Set non-membership
`<:`	$\subseteq$	Subset
`/<:`	$\not\subseteq$	Not a subset
`<<:`	$\subset$	Proper subset
`/<<:`	$\not\subset$	Not a proper subset
`finite`	finite	Finite
`partition`	partition	Partition

Table 3: Set predicates

ASCII	Symbols	Explanation
`BOOL`	BOOL	BOOL set
`TRUE`	TRUE	TRUE
`FALSE`	FALSE	FALSE
`bool`	bool	bool predicate

Table 4: BOOL and bool

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

Table 5: Numbers

ASCII	Symbols	Explanation
`>`	$>$	Greater
`<`	$<$	Less
`>=`	$\geq$	Greater or equal
`<=`	$\leq$	Less or equal

Table 6: Number predicates

ASCII	Symbols	Explanation
`<->`	$\leftrightarrow$	Relations
`dom`	dom	Domain
`ran`	ran	Range
`<<->`	$\nleftrightarrow$	Total relations
`<->>`	$\twoheadleftarrow$	Surjective relations
`<<->>`	$\twoheadleftarrow\!\!\!\!\!\!\rightarrow$	Total surjective relations
`circ`	$\circ$	Backward composition
`id`	id	Identity
`<|`	$\lhd$	Domain restriction
`<<|`	$\not\lhd$	Domain subtraction
`|>`	$\rhd$	Range restriction
`|>>`	$\not\rhd$	Range subtraction
`~`	$^{-1}$	Inverse
`<+`	$\leftarrow\!\!\!\!+$	Overriding
`><`	$\otimes$	Direct product
`||`	$\parallel$	Parallel product
`prj1`	prj_1	First projection
`prj2`	prj_2	Second projection

Table 7: Relations

ASCII	Symbols	Explanation
+->	$\pfun$	Partial functions
-->	$\fun$	Total functions
>+>	$\pinj$	Partial injections
>->	$\inj$	Total injections
+>>	$\psur$	Partial surjections
->>	$\sur$	Total surjections
>->>	$\bij$	Bijections
%	λ	Lambda abstraction

Table 8: Functions

ASCII	Symbols	Explanation
:=	$:=$	Becomes equal to
::	$:\in$	Choice from a set
:—	$:\mid$	Choice by predicate

Table 9: Functions

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

---

```
machine Sens∨_m0_SNSR
variables
  SNSR
invariants
  @thm0_1∈ "SNSR ∈BOOL" the∨em
```

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 SensV_m1_DEP
refines SensV_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 SensV_m3_Ctrl

refines

  SensV_m2_Snsr

variables

  SNSR

  DEP

  Snsr_01

```

```
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
```