Introducing B Machines

Michael Butler, University of Southampton

MACHINE $Counter$

CONSTANTS max
PROPERTIES $max \in \mathbb{N}$

VARIABLES ctr
INVARIANT

$$ctr \in \mathbb{N} \wedge$$

$$0 \leq ctr \leq max$$

INITIALISATION $ctr := 0$

OPERATIONS

$Inc \;\widehat{=}\;$ PRE
$ctr < max$
THEN
$ctr := ctr + 1$
END

$Dec \;\widehat{=}\;$ PRE
$ctr > 0$
THEN
$ctr := ctr - 1$
END

$val \longleftarrow Display \;\widehat{=}$
$val := ctr$

MACHINE $Dictionary$

SETS $Word$

VARIABLES $known$

INVARIANT

$$known \subseteq Word$$

INITIALISATION $dict := \{\}$

OPERATIONS

$AddWord(w) \;\widehat{=}$
PRE
$w \in Word$
THEN
$known := known \cup \{w\}$
END

$res \longleftarrow CheckWord(w) \;\widehat{=}$
PRE
$w \in Word$
THEN
$res := bool(w \in known)$
END

B machine contains

- Sets: abstract types used in specification
- Constants: logical variables who's value remain constant
- Properties: constraints on the constants. A property is a logical predicate.
- Variables: state variables who's values can change
- Invariants: constraints on the variables that should always hold true. An invariant is a logical predicate.
- Initialisation: initial values for the abstract variables
- Operations: operations specifying ways in which the abstract variables can change

Operation Preconditions

$Inc \;\widehat{=}\;$ PRE

$ctr < max$ *Precondition*

THEN

$ctr := ctr + 1$ *Body*

END

A precondition is a predicate on the variables. An operation should only be called when its precondition is true.

The body of an operation assigns new values to the state variables. The body of an operation should maintain the invariant provided the precondition holds before the operation is executed.

Why does Inc preserve the invariant $ctr \leq max$?

Operation Parameters

Operations can have **input** parameters and **output** parameters:

$$
\begin{array}{l}
res \longleftarrow CheckWord(w) \quad \widehat{=} \\
\quad \text{PRE} \\
\quad\quad w \in Word \\
\quad \text{THEN} \\
\quad\quad res := bool(w \in dict) \\
\quad \text{END}
\end{array}
$$

w is an **input** parameter. Its type is determined by the precondition.

res is an **output** parameter. Its return value is determined by an assignment to res. Its type is inferred from the type of the value assigned to it.

Multiple Assignment

An operation body (and a machine initialisation) can have multiple assignments separated by ||.

MACHINE $CountingDictionary$
SETS $Word$

VARIABLES $known, count$
INVARIANT

$$known \subseteq Word \ \wedge$$
$$count \in \mathbb{N} \ \wedge$$
$$count = card(known)$$

OPERATIONS

$AddWord(w) \ \widehat{=}$
PRE
$w \in Word$
THEN
$known := known \cup \{w\}$ ||
$count := count + 1$
END

Multiple Assignment

An operation body (and a machine initialisation) can have multiple assignments separated by ||.

MACHINE $CountingDictionary$
SETS $Word$

VARIABLES $known, count$
INVARIANT

$$known \subseteq Word \ \wedge$$
$$count \in \mathbb{N} \ \wedge$$
$$count = card(known)$$

OPERATIONS

$AddWord(w) \ \widehat{=}$
PRE
$w \in Word$
THEN
$known := known \cup \{w\}$ ||
$count := count + 1$
END

Can anyone spot an error with this specification?