TD 3 - Proving liveness and termination

A Rodin machine is non-blocking if during every computation of the machine at least one of the events of the machine is enabled (i.e. all the guards of this event are true). Equivalently we say that the machine has the liveness property. The liveness property can be proved by creating a new invariant which is the disjunction of all guards of the different events of the machine. Proving that the machine is non-blocking is equivalent to proving the invariant.

A Rodin machine terminates if all its executions end up in a final event. To prove termination, one introduces a variant, i.e. an expression with positive integer value and marks all non final events as "convergent". Each execution of a non final event should decrease the value of the variant, and each convergent event generates a corresponding proof obligation.

Exercice 1. Liveness and termination of the counter machine

1. Prove the counter machine is non-blocking.
2. Assuming that reset is a final event, prove that the counter machine terminates.
3. Optional. Prove the decimal counter is non-blocking.

Exercice 2. Liveness and termination

1. Prove termination and non-blocking of the MinMax machine.
2. Prove the ROMController is non-blocking.

Exercice 3. Searching arrays

1. Create a context ArraySearch and a machine SearchProblem which specifies the following algorithmic problem : given as input an array with integer values and an integer, decide whether the array contains the integer.
2. Create a machine LinearSearch that search an element by linear search. Prove correctness, non-blocking and termination.
3. Create a context SortedArray and a machine DichotomicSearch that search an element by dichotomy, assuming the array is sorted. Prove correctness, non-blocking and termination.
4. Refine the context to specify the intial array size and content and animate the model.

Exercice 4. The bus Perform exercise 5 TD1 and prove liveness and termination for the bus.