Justification Document
Proven Lift

Adrian Friedli
Alexander Bernauer

December 6, 2010

1 Refinements

All machines are named after the following scheme: `lift`*NN_DESC* where *NN* is the zero padded two digit natural number of the machine and *DESC* is an arbitrary text describing the purpose of the refinement.

A machine with number n is a refinement of the machine with number $n-1$. The only exceptions are the machine 1 and the machine 11. The machine 10 is used for the code generation and is complete except for the proves of convergence of the events and lifeness of the system. These proves are only possible under addidional assumptions which are introduced in machine 11 which refines machine 9. Furthermore, machine 10 adds no additional functionality but only maps the internal representations of the state to the demands of the simulation environment. So the convergence and lifeness proves of machine 11 and its refinements also hold for machine 10, given the additional assumptions hold of course.

The assumptions introduced by machine 11 are that the controler can change its states faster then the environment which means that the controler never "misses" anything. This prerequisite is reflected by the additional guards of `ELEVATOR_LEAVES_FLOOR_UP` and `ELEVATOR_LEAVES_FLOOR_DOWN`. If these guards are disabled then the event `stop` is enabled. This means that the environmental elevator can not leave a floor if the controler wants to stop the cable engine.

2 Functional Requirements

FUN20 This requirement reflects the liveness property of the system and is assured by theorem `37_life`. It is important to note that this theorem holds only under the assumptions introduced in machine 11 as explained above.

FUN21 This requirement is assured by invariant `inv9_8`.

FUN22 This requirement is assured by invariant `inv9_1`.

FUN23 This requirement is assured by invariant `inv9_6`.

FUN24 In the machine 9 the events `switch_schedule_to_up` and `resume_schedule_up` or `switch_schedule_to_down` and `resume_schedule_down` respectively set the schedule to up or down if there is a corresponding element in the `requests` set which reflects all pending requests. If a new request is issued the guards of one of these events is enabled.

FUN25 These requirements are assured by invariants `inv9_9` and `inv9_10`.

FUN26 This requirement is assured by invariant `inv9_11`.

FUN27 The schedule is changed from up/down to down/up only by the events `switch_schedule_to_up` and `switch_schedule_to_down`. The guard `grd9_4` of theses events assures that this requirement is fulfilled.

FUN28 In the machines 6, 7, and 8 the sets `snsrFloorButtonsSet`, `snsrUpButtonsSet`, and `snsrDownButtonsSet` reflect if a button is pressed. The guards of the only events which turn ligths on, i.e. `turn_floor_button_lights_on`, `turn_up_button_lights_on`, and `turn_down_button_lights_on` are enabled if the corresponding set is not empty.

FUN29 This requirement is implemented by the events `turn_floor_button_lights_off`, `turn_up_button_lights_off`, and `turn_down_button_lights_off`.

FUN30 This requirement is assured by invariant `inv5_9`.

FUN31 This requirement is assured by invariant `inv5_10`.

FUN32 This requirement is assured by invariant `inv4_11`.

FUN33 This requirement is assured by invariant `inv1_1`.

FUN34 This requirement is assured by invariant `inv1_2`.

FUN35 In every state one of the events `USER_PRESSES_FLOOR_BUTTON` and `USER_RELEASES_FLOOR_BUTTON` are enbaled because the set `PhyFloorButtonsSet` either contains a x between 0 and `LAST_FLOOR` or it doesn't. So the system is never deadlocked. The analogous argument holds for the events corresponding to up and down buttons. Together with the lifeness property of the system the results in the required behavior that the elevator system is always "working".