EECS3342 System Specification and Refinement (Winter 2022)

Q&A - Week 10 Lectures

Thursday, March 31

Announcements

+ Lecture W11 released FTP
+ Written Test 4

Here we state that ml_pass and il_pass are 1 because init does not establish them and neither ML_tl_green or IL_tl_green have occured,
but if init does not establish them, would they not have a nondeterministic value prior to the first occurrences of ML_tl_green or IL_tl_green?

d = 2	ml_pass	il_pass
< init,	1	1
ML_tl_green,	0	1
ML_out_1,	1	1
ML_out_2,	1	1
IL_in,	1	1
IL_in,	1	1
IL_tl_green,	1	0
IL_out_1,	1	1
IL_out_2,	1	1
ML_in,	1	1
ML_in	1	1
>		

V: 2

Variant

ml_pass + il_pass

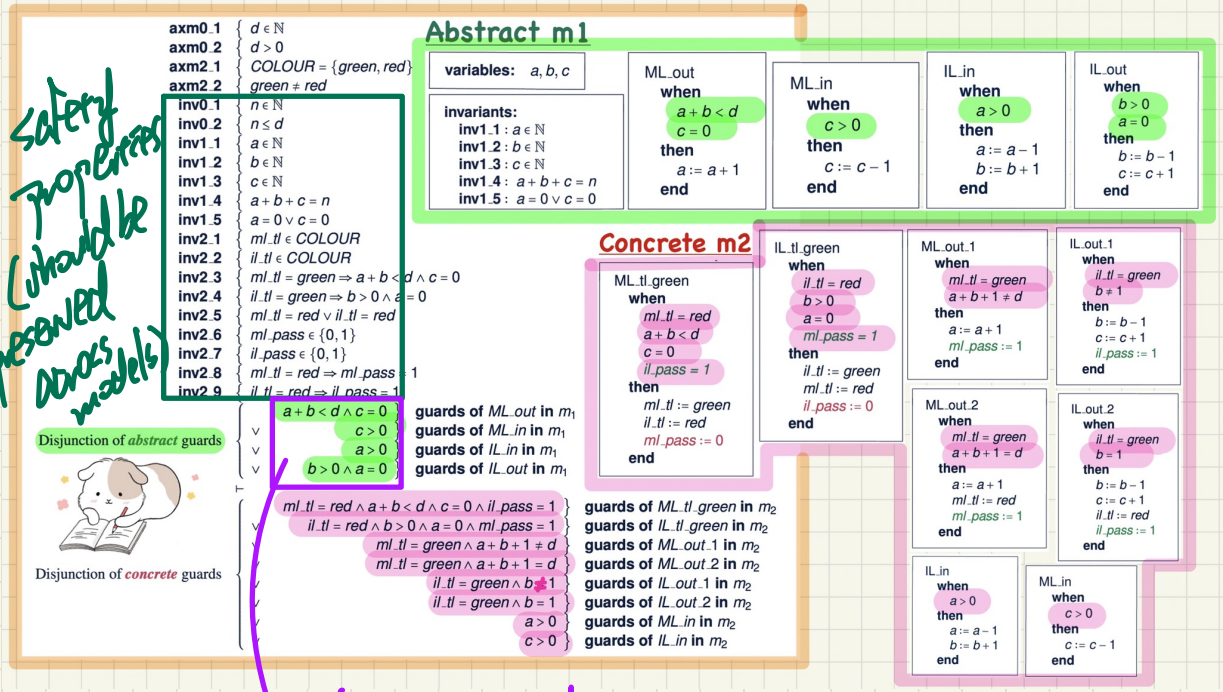
Say init does not initialize:

ml_pass: 0

il_pass: 0

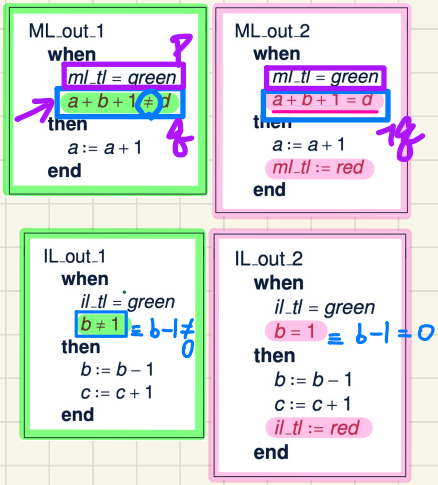
ML_tl_green → ml_pass: 0, il_pass: 0

For the PO of DLF in m2, why do we not include the abstract guards from m0? We include the axioms and invariants from that model, so why not the guards as well?



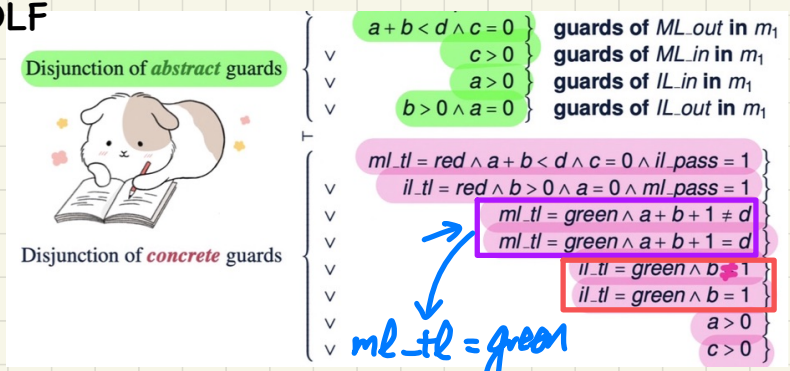
Towards the end of the lecture you explain the splitting of events.
I was wondering for ML_out.1, the guard,
would a + b + 1 < d be a better design than a + b + 1 /= d?
If we reach the edge case where we hit the max amount of cars, the guard would still pass
Or is the ML_tl.green guard (a + b < d) sufficient enough to resolve this issue since we have the predecessor statement "ml_tl = green"?
The same applies to IL_out.1 where b > 1 instead of b /= 1.
Thanks!

disjunction of guards of ML_out_1 and ML_out_2 : $(p \wedge q) \vee (p \wedge \neg q) \equiv p$



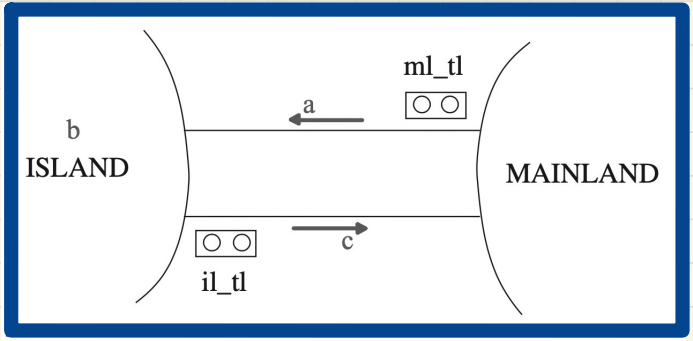
- $a + b + 1 < d \vee a + b + 1 = d \neq \text{TRUE}$
- Simplification of DLF

ML_out
when
ml_tl = green
then
a := a+1
end



When trying to figure out inv2_3, can you explain how ml_tl = green ⇒ a+b < d ∧ c=0

at least one more car can be allowed to exit from ML to BRI.



invariants:

inv2_1 : $ml\_tl \in COLOUR$

inv2_2 : $il\_tl \in COLOUR$

inv2_3 : $ml\_tl = green \Rightarrow a + b < d \wedge c = 0$

inv2_4 : $il\_tl = green \Rightarrow b > 0 \wedge a = 0$

$a+b+1 \leq d$