
CERN

Dániel Darvas (EN-ICE-PLC)

Formal verification of industrial control systems… at CERN

1st Developers@CERN Forum
29/09/2015

Contains joint work of B. Fernández, E. Blanco, S. Bliudze, J.O. Blech,
J-C. Tournier, T. Bartha, A. Vörös, I. Majzik, R. Speroni, M. Lettrich

CERN

MŰEGYETEM 1782

http://go.cern.ch/**7L9h**

Source: http://www.iphonetextgenerator.com/

Context – CERN

- PLCs for controlling **vacuum**, **cryogenics, CV**, etc. systems

Context – CERN

- PLCs for controlling **vacuum**, **cryogenics, CV**, etc. systems

- Failures might have *negative impact*

Context – CERN

- PLCs for controlling **vacuum**, **cryogenics, CV**, etc. systems
- Failures might have *negative impact*
- **Increasing complexity** without **decreasing quality**?

CERN

MŰEGYETEM 1782

Context – PLCs at CERN

- Programmable Logic Controllers
  robust industrial computers

Context – PLCs at CERN

- Programmable Logic Controllers
  robust industrial computers

- Small computing capacity,
  special programming languages

Context – PLCs at CERN

- Programmable Logic Controllers
  robust industrial computers
- Small computing capacity,
  special programming languages
- **1000+ PLCs** at CERN

Goal

- To **improve the quality** by eliminating bugs
  - Complementing automated and manual testing

Goal

- To **improve the quality** by eliminating bugs
  - Complementing automated and manual testing

- Apply **model checking** to find **“high quality” bugs**

Goal

- To **improve the quality** by eliminating bugs
  - Complementing automated and manual testing
- Apply **model checking** to find **“high quality” bugs**
- **Integrate** formal verification to the development process

What is formal verification?

What is formal verification?

- **Formal verification**: mathematically sound methods to check properties of specifications / implementations / …

What is formal verification?

- **Formal verification**: mathematically sound methods to check properties of specifications / implementations / …

- **Model checking**
  - **Automated** formal verification method
  - Checks **all possible executions** (contrarily to testing)
  - Goal: prove correctness OR **find hidden/rare problems**

What is formal verification?

- **Formal verification**: mathematically sound methods to check properties of specifications / implementations / …

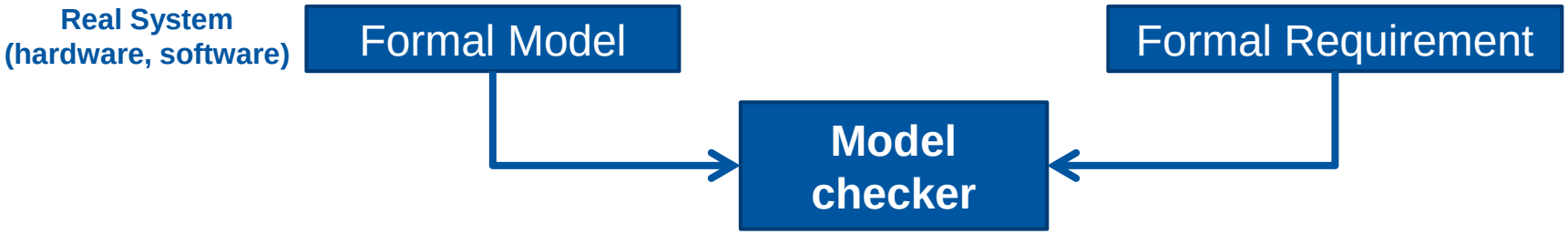
- **Model checking**
  - **Automated** formal verification method
  - Checks **all possible executions** (contrarily to testing)
  - Goal: prove correctness OR **find hidden/rare problems**

Real System (hardware, software) → Formal Model → Formal Requirement

What is formal verification?

- **Formal verification**: mathematically sound methods to check properties of specifications / implementations / …

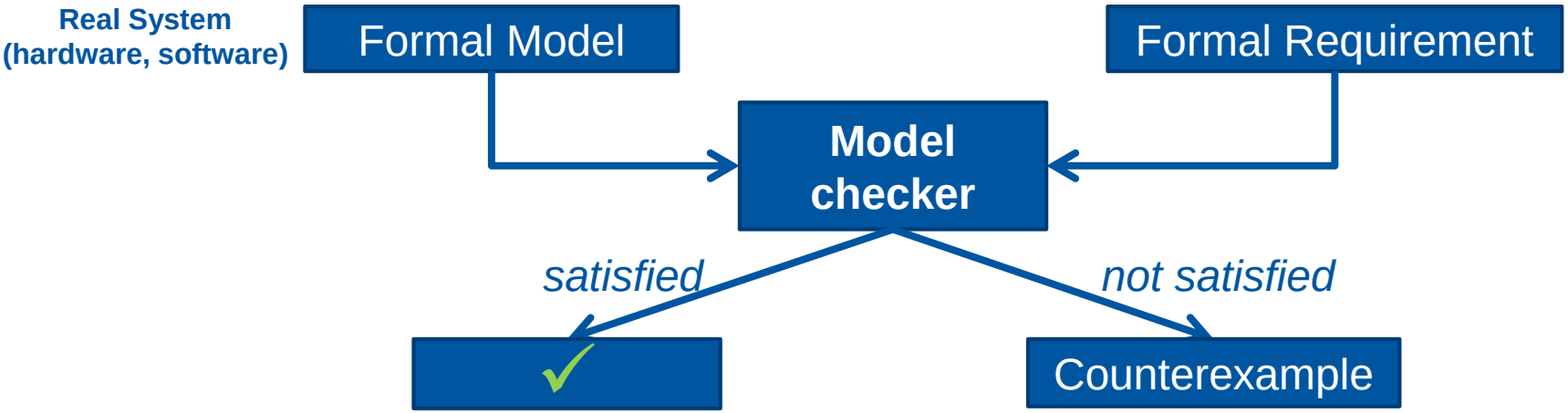
- **Model checking**
  - **Automated** formal verification method
  - Checks **all possible executions** (contrarily to testing)
  - Goal: prove correctness OR **find hidden/rare problems**



What is formal verification?

- **Formal verification**: mathematically sound methods to check properties of specifications / implementations / …

- **Model checking**
  - **Automated** formal verification method
  - Checks **all possible executions** (contrarily to testing)
  - Goal: prove correctness OR **find hidden/rare problems**

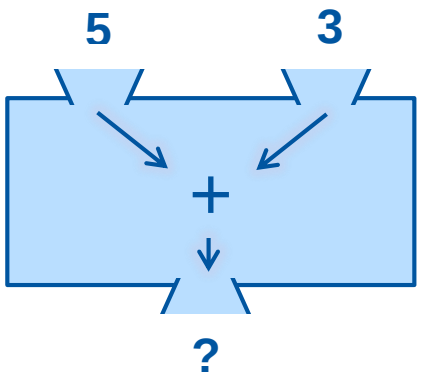


Testing vs. model checking

Testing

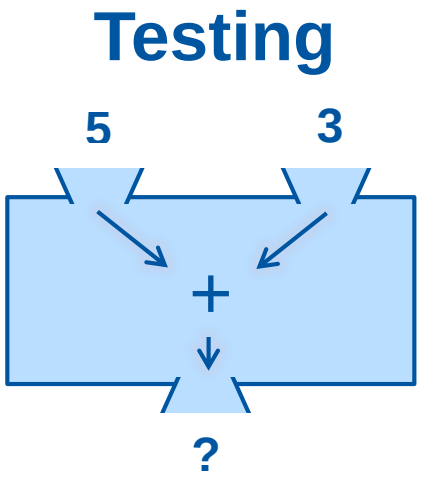
Testing vs. model checking

Testing




```
add(5,3)=8 ?
```

Testing vs. model checking

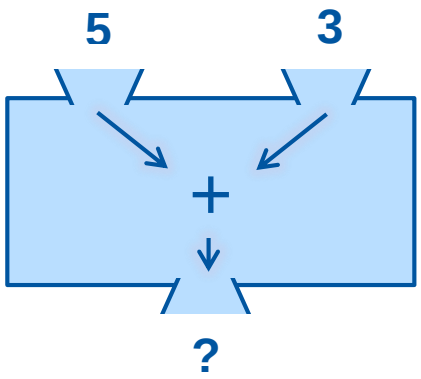


`add(5,3)=8 ?`

- **Inputs are known**, outputs are checked

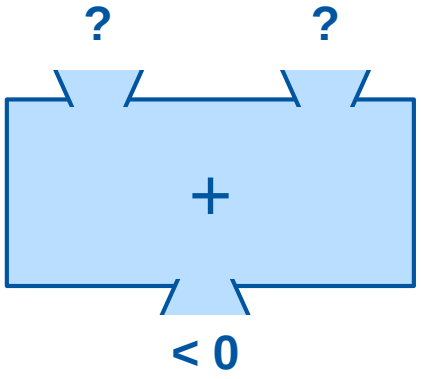
Testing vs. model checking

Testing



`add(5,3)=8 ?`

Model checking

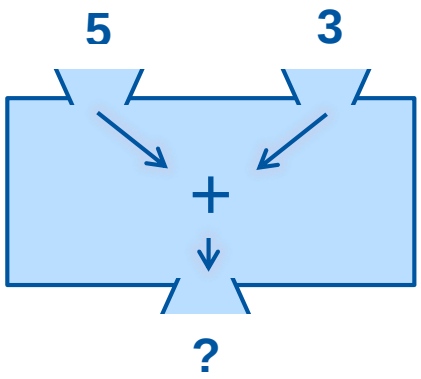


`add(*,*)<0 ?`

- **Inputs are known**, outputs are checked

Testing vs. model checking

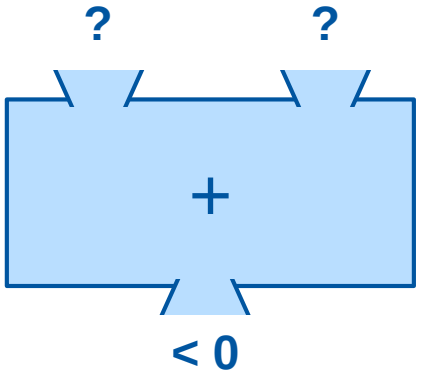
Testing



`add(5,3)=8 ?`

- **Inputs are known**, outputs are checked

Model checking



`add(*,*)<0 ?`

- E.g. the possibility of an **output combination** is checked.
- Can be used in other ways too.

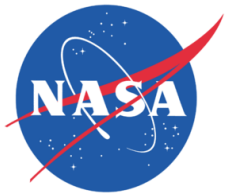
Usage of formal verification

Usage of formal verification

- Used both in **industry** and **academia**
  - typically when the *cost of failure is high*

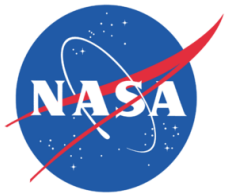
Usage of formal verification

- Used both in **industry** and **academia**
  - typically when the *cost of failure is high*



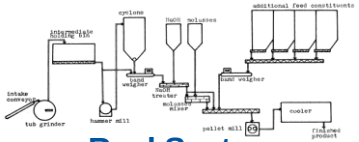
Usage of formal verification

- Used both in **industry** and **academia**
  - typically when the *cost of failure is high*



- Formal verification for **PLCs**
  - mostly in academic environment
  - not widely spread yet in industry – **too difficult**!

Challenges and answers



Real System
(hardware, software)

Specifications

Formal Model

Formal Requirement

Model checker

satisfied

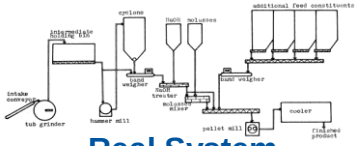
not satisfied

✓

Counter-example

Challenges and answers

How to get models?



Real System
(hardware, software)

Specifications

Formal Model

Formal Requirement

Model checker

satisfied

not satisfied

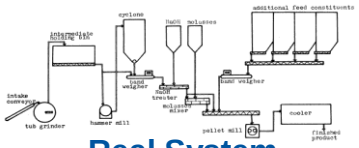
✓

Counter-example

Challenges and answers

How to get models?

Automated generation



Real System
(hardware, software)

Specifications

Formal Model

Formal Requirement

Model checker

satisfied

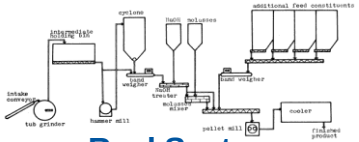
not satisfied

✓

Counter-example

Challenges and answers

How to get models?



Real System
(hardware, software)

Automated generation

Formal Model

Specifications

How to formalize requirements?

Formal Requirement

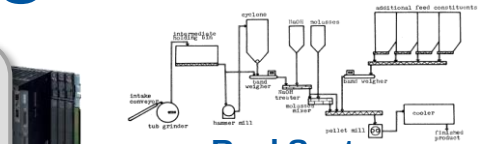
Model checker

satisfied

not satisfied

Counter-example

Challenges and answers



Real System
(hardware, software)

Specifications

How to get models?

How to formalize requirements?

Formal Model

Formal Requirement

Automated generation

Requirement patterns

Model checker

satisfied

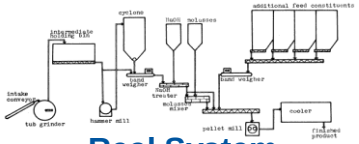
not satisfied

✓

Counter-example

Challenges and answers

How to get models?



Real System
(hardware, software)

Specifications

How to formalize requirements?

Automated generation

Formal Model

Formal Requirement

Requirement patterns

Which model checker should be used?

Model checker

satisfied

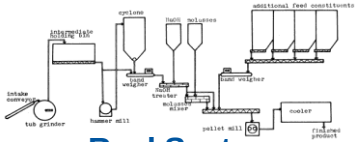
not satisfied

✓

Counter-example

Challenges and answers

How to get models?



Real System
(hardware, software)

Specifications

How to formalize requirements?

Formal Model

Formal Requirement

Automated generation

Requirement patterns

Which model checker should be used?

Model checker

satisfied

not satisfied

Multiple (general meth., intermediate model)

✓

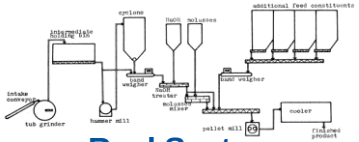
Counter-example

CERN

MŰEGYETEM 1782

Challenges and answers

How to get models?



Real System
(hardware, software)

Specifications

How to formalize requirements?

Formal Model

Formal Requirement

Automated generation

Requirement patterns

Which model checker should be used?

Model checker

satisfied

not satisfied

Multiple (general meth., intermediate model)

✓

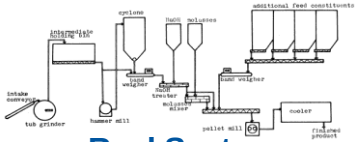
How to make it efficient?

Counter-example

CERN

MŰEGYETEM 1782

Challenges and answers



Real System
(hardware, software)

Specifications

How to get models?

Automated generation

Formal Model

Formal Requirement

How to formalize requirements?

Requirement patterns

Which model checker should be used?

Multiple (general meth., intermediate model)

Model checker

satisfied

✓

not satisfied

Counter-example

How to make it efficient?

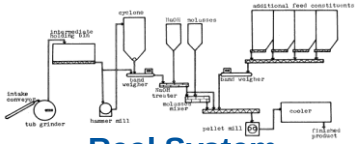
Reductions

CERN

MŰEGYETEM 1782

Challenges and answers

How to get models?



Real System (hardware, software)

Specifications

How to formalize requirements?

Formal Model

Formal Requirement

Automated generation

Requirement patterns

Which model checker should be used?

Model checker

satisfied

not satisfied

How to proceed with a counterexample?

Multiple (general meth., intermediate model)

✓

Counter-example

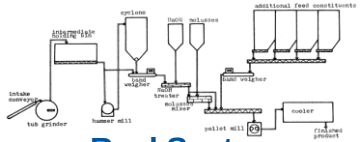
How to make it efficient?

Reductions

CERN

MŰEGYETEM 1782

Challenges and answers



Real System
(hardware, software)

Specifications

How to get models?

Automated generation

Formal Model

Formal Requirement

How to formalize requirements?

Requirement patterns

Which model checker should be used?

Multiple (general meth., intermediate model)

Model checker

satisfied

✓

not satisfied

Counter-example

How to proceed with a counterexample?

Analysis and demonstration

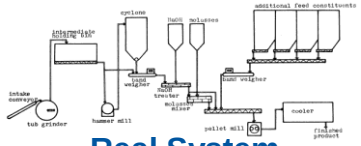
How to make it efficient?

Reductions

CERN

MŰEGYETEM 1782

Challenges and answers



Real System
(hardware, software)

Specifications

How to get models?

Formal Model

Automated generation

Formal Requirement

How to formalize requirements?

Requirement patterns

Which model checker should be used?

Model checker

satisfied

not satisfied

How to proceed with a counterexample?

Multiple (general meth., intermediate model)

✓

Counter-example

Analysis and demonstration

How to make it efficient?

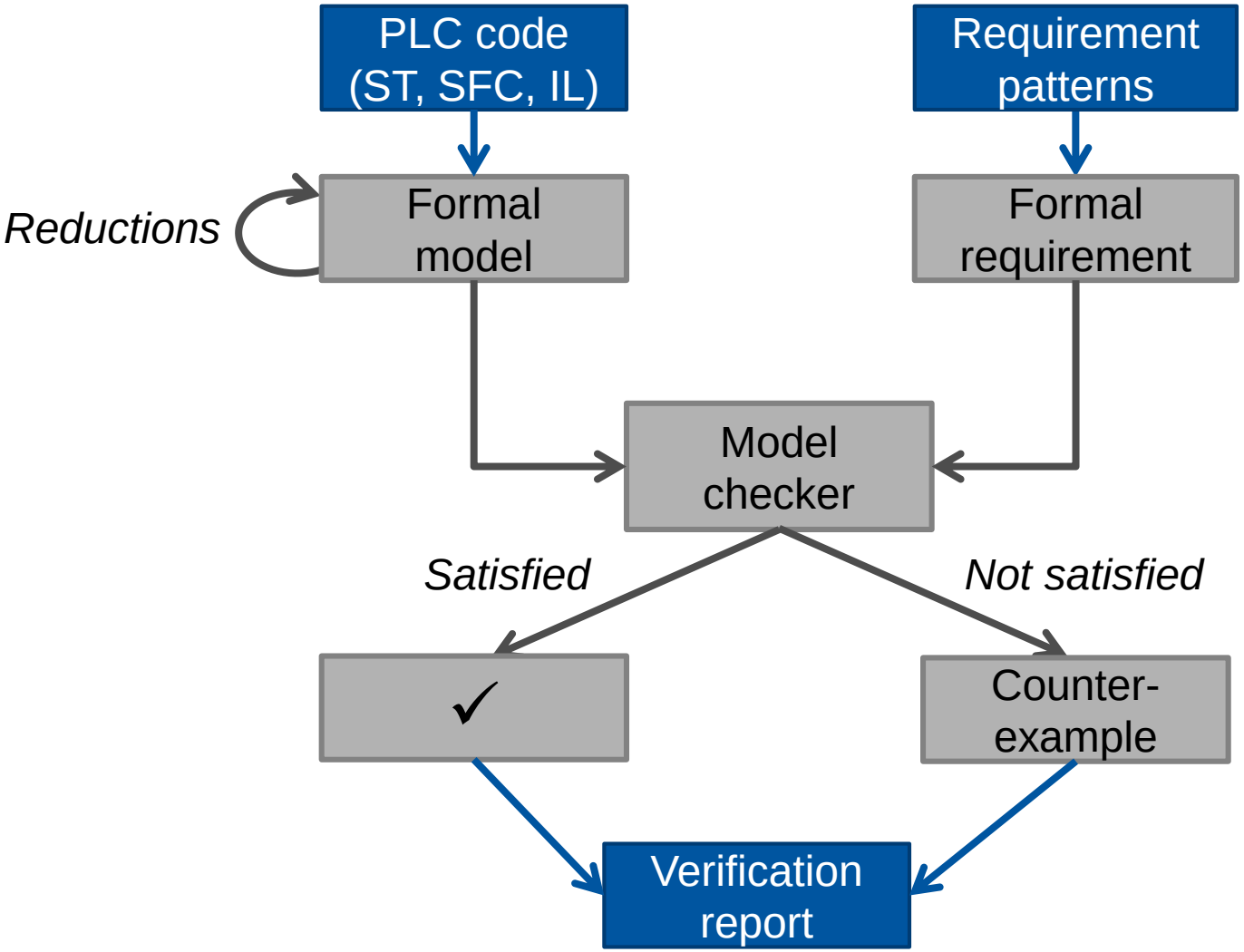
Reductions

And it should be integrated to the development process.

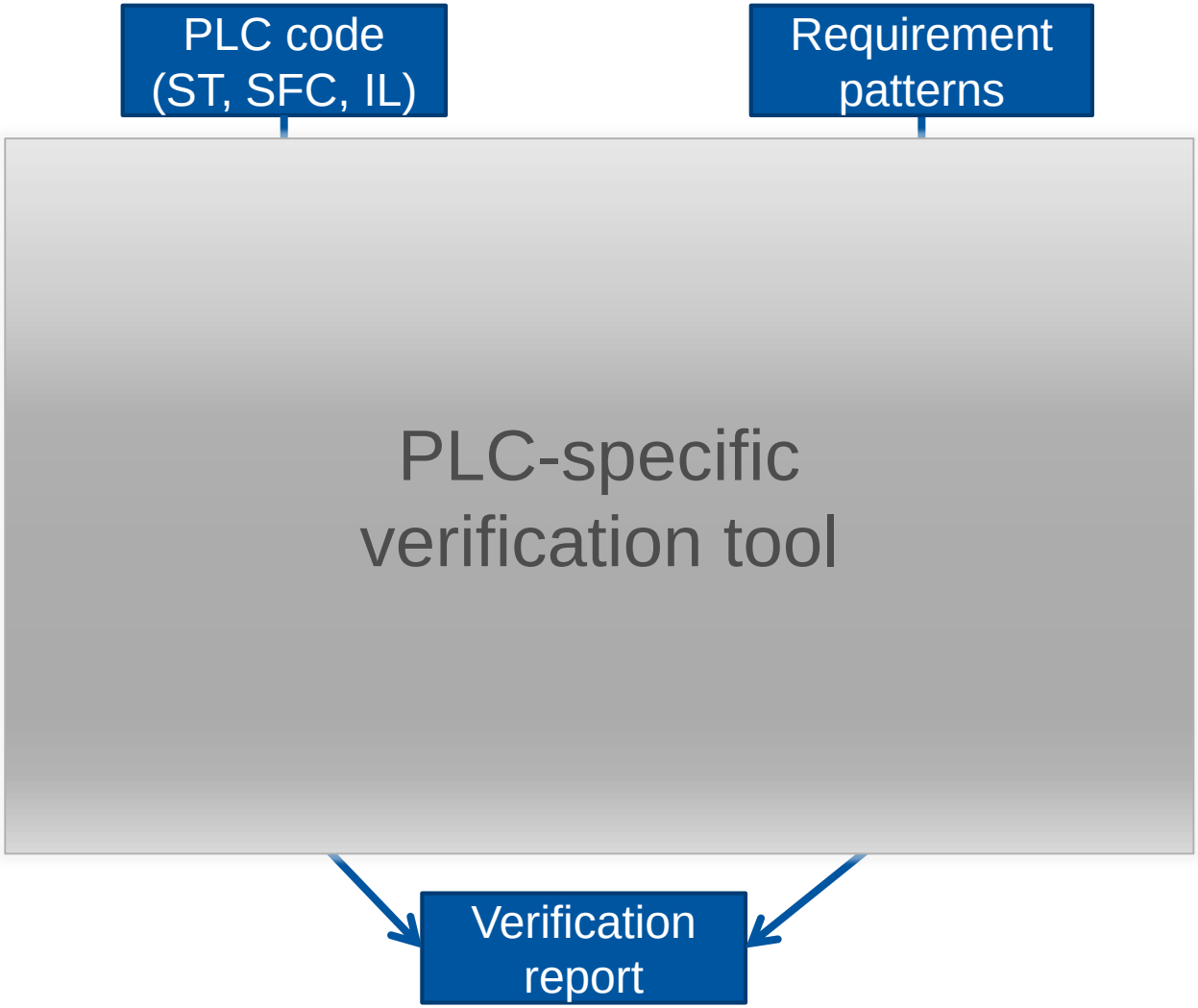
CERN

MŰEGYETEM 1782

Model checking (extended workflow for PLCs)



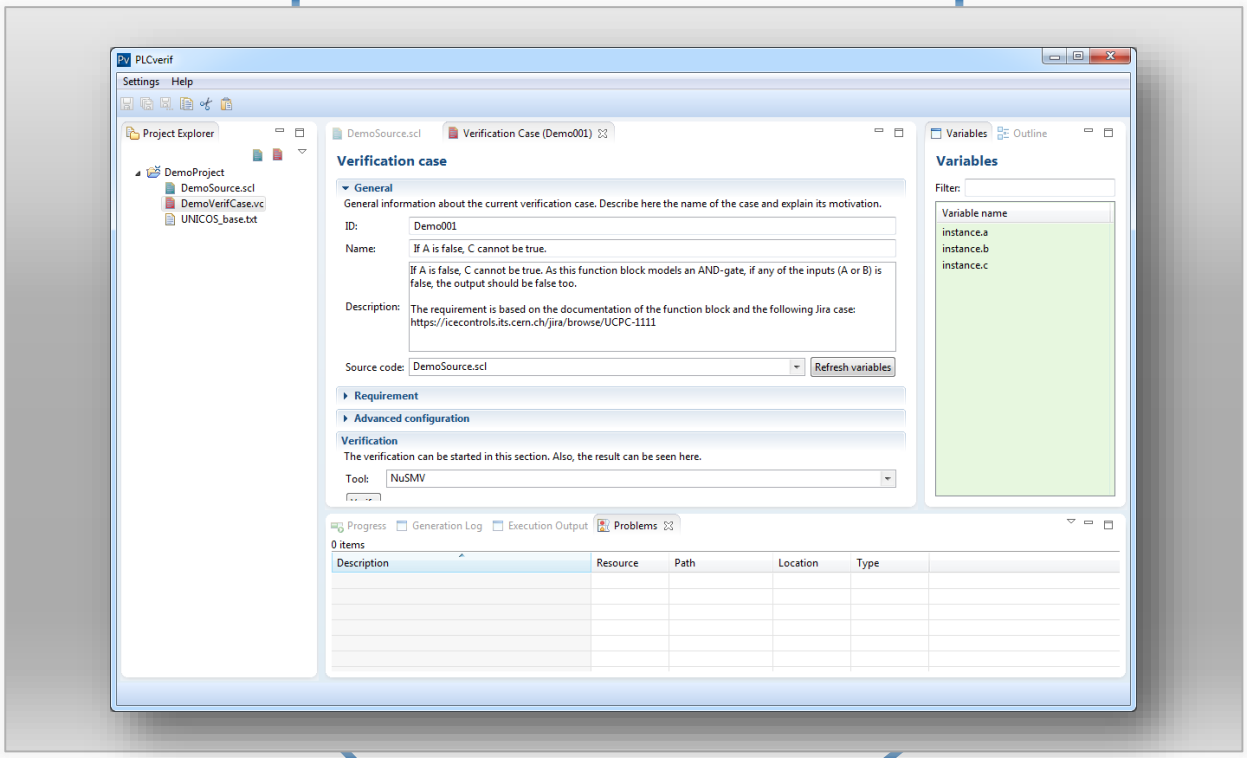
Model checking (extended workflow for PLCs)



Model checking (extended workflow for PLCs)

PLC code (ST, SFC, IL)

Requirement patterns



Verification report

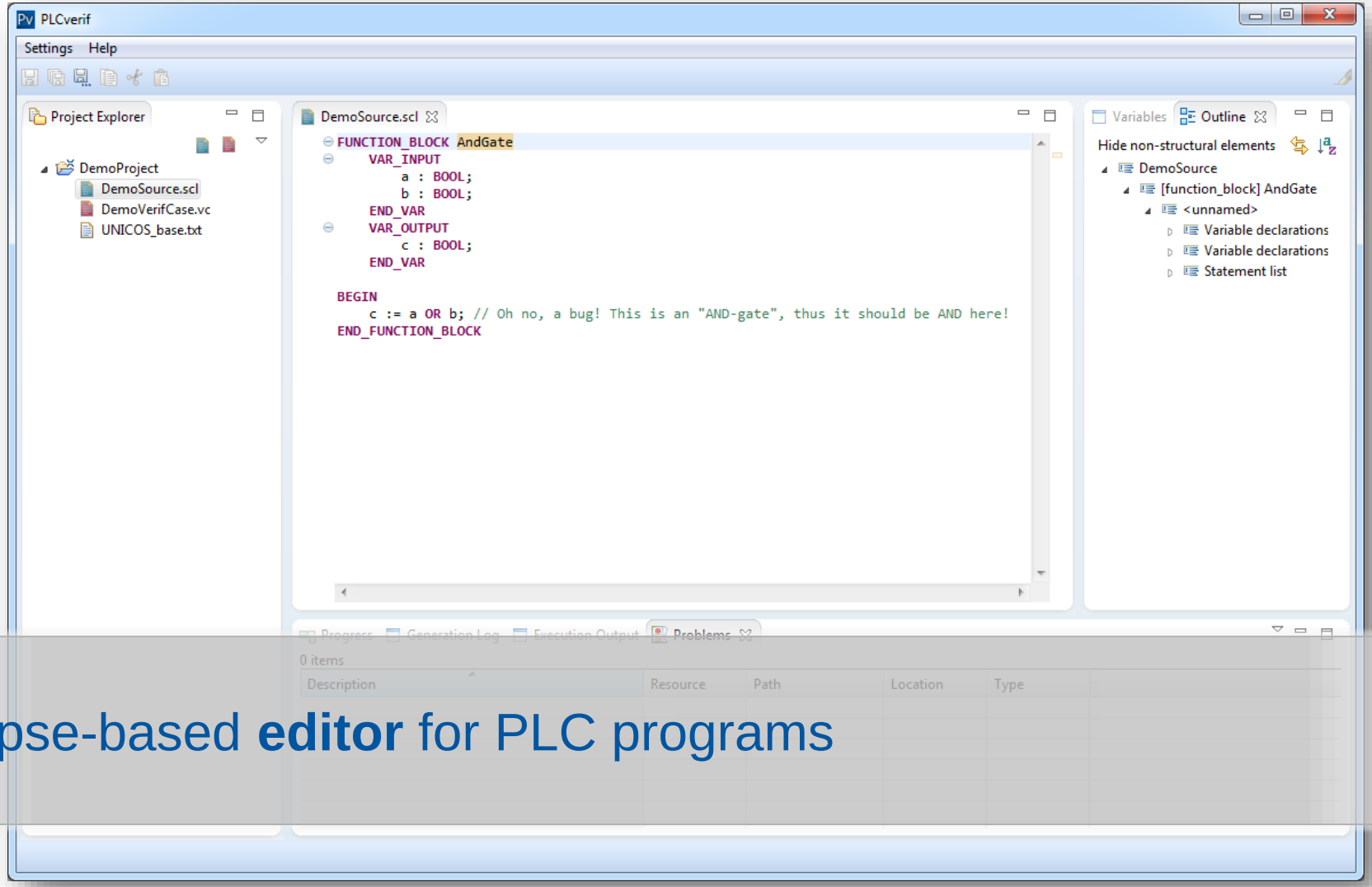
CERN

MŰEGYETEM 1782

Model checking in practice (at CERN)

The PLCverif tool

The PLCverif tool

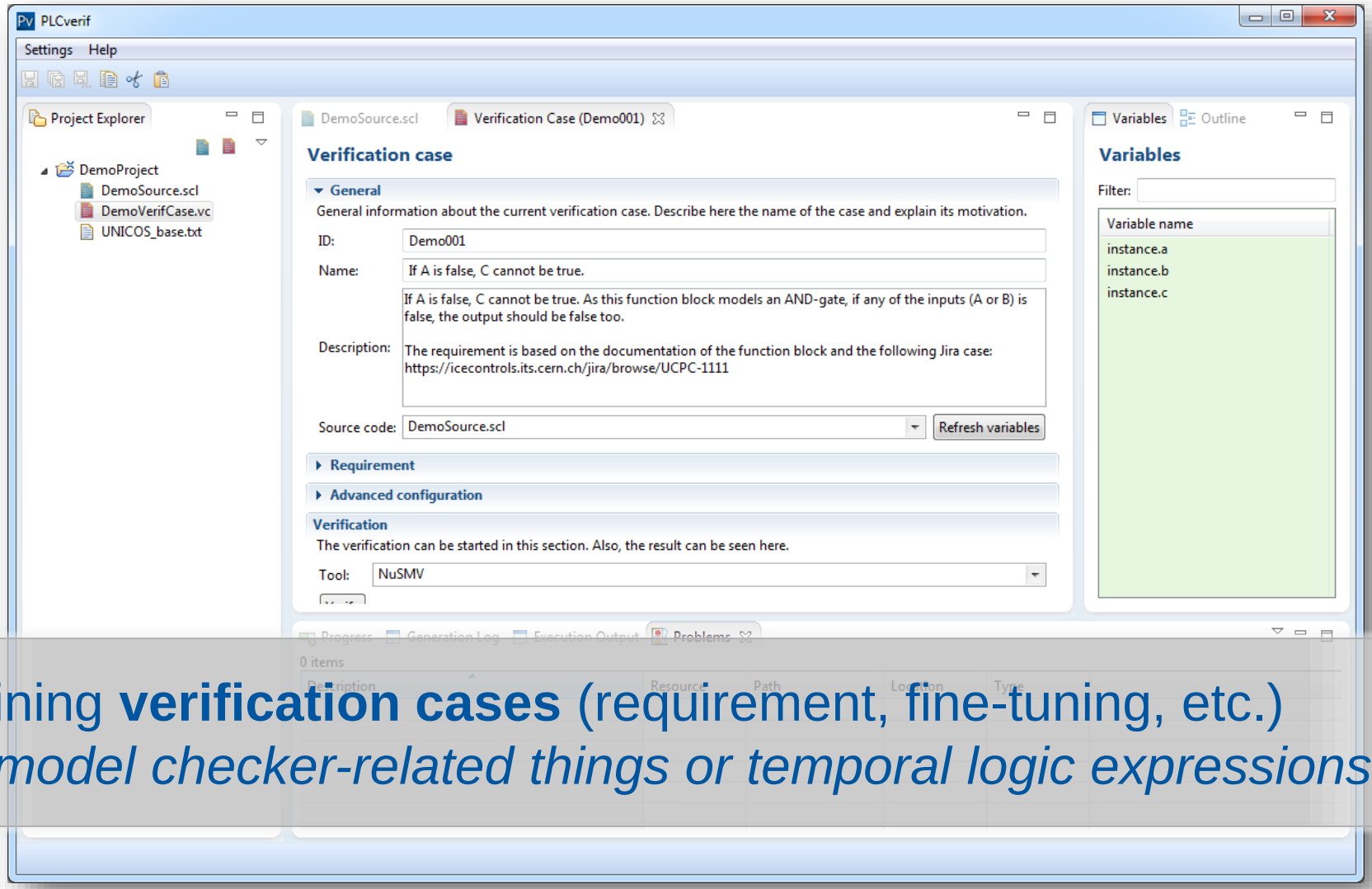


Eclipse-based **editor** for PLC programs

CERN

MŰEGYETEM 1782

The PLCverif tool



Defining **verification cases** (requirement, fine-tuning, etc.)

No model checker-related things or temporal logic expressions

CERN

MŰEGYETEM 1782

The PLCverif tool

PLCverif — Verification report

Generated at Mon Jul 07 15:19:22 CEST 2014 | PLCverif v2.0.1 | (C) CERN EN-ICE-PLC | Show/hide expert details

ID:	**Demo001**
Name:	If A is false, C cannot be true.
Description:	If A is false, C cannot be true. As this function block models an AND-gate, if any of the inputs (A or B) is false, the output should be false too. The requirement is based on the documentation of the function block and the following Jira case: https://icecontrols.its.cern.ch/jira/browse/UCPC-1111
Source file:	DemoSource.scl
Requirement:	3. A = false & C = true is impossible at the end of the PLC cycle.
Result:	**Not satisfied**

Tool: nusmv
Total runtime (until getting the verification results): 212 ms
Total runtime (incl. visualization): 361 ms

Counterexample

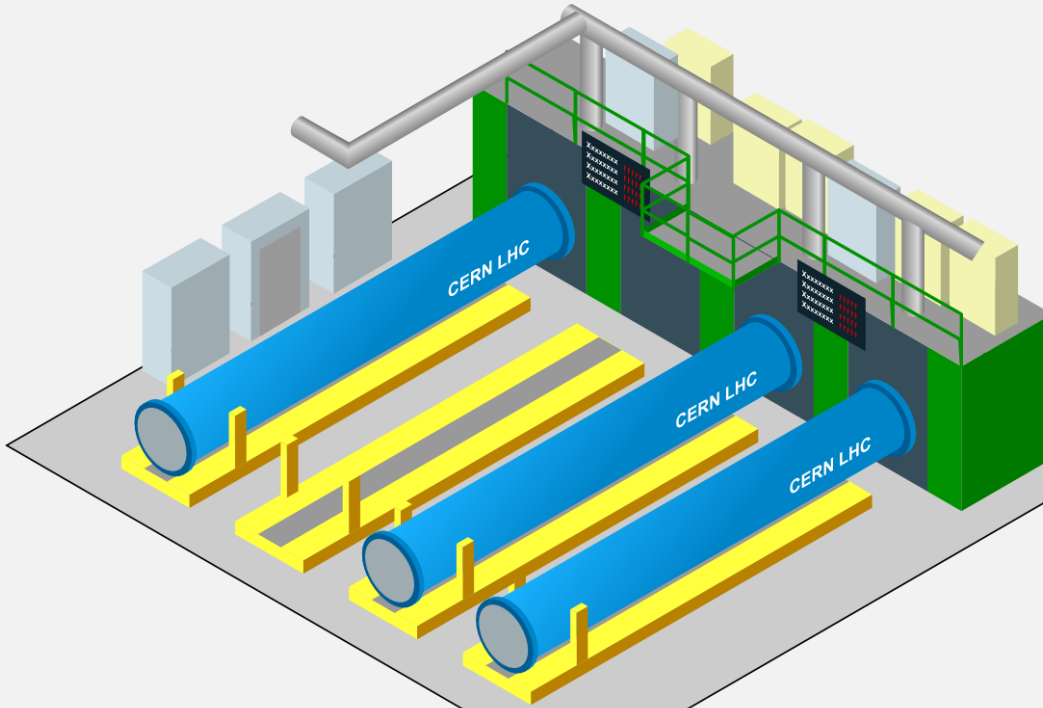
	Variable	End of **Cycle 1**
Input	**a**	FALSE
Input	**b**	TRUE
Output	**c**	TRUE

Click-button verification,
verification **report** with the analysed **counterexample**

CERN

MŰEGYETEM 1782

Example – SM18 safety system



SM18

Goal: ensuring **safety** by allowing/forbidding tests

SM18

Goal: ensuring **safety** by allowing/forbidding tests

Core:

CERN

MŰEGYETEM 1782

SM18

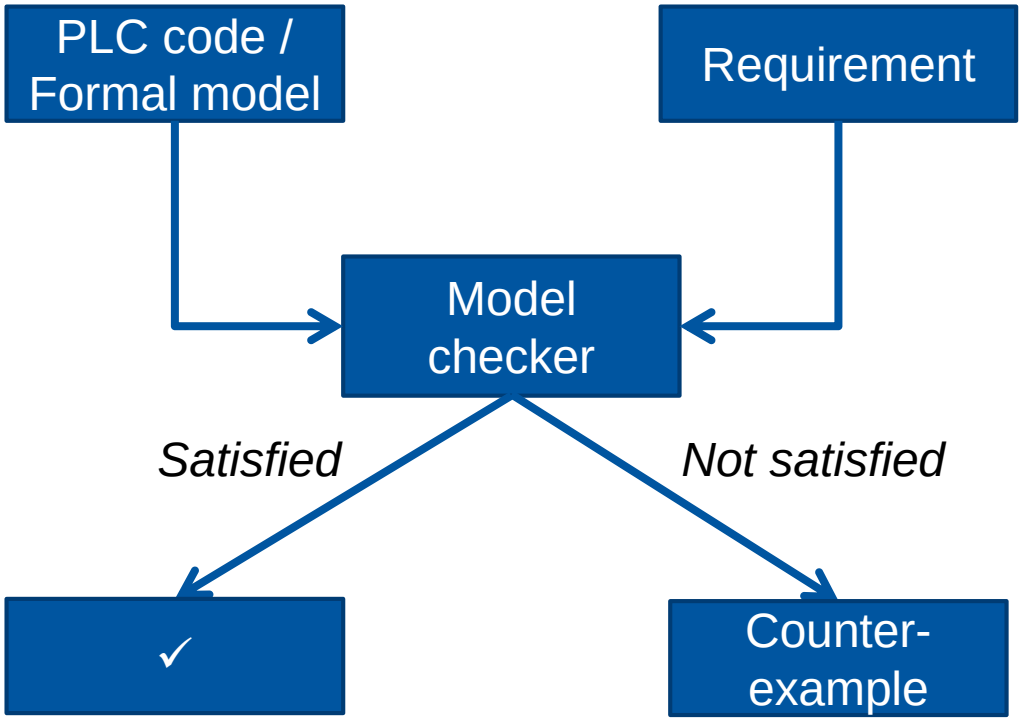
Goal: ensuring **safety** by allowing/forbidding tests

Core:

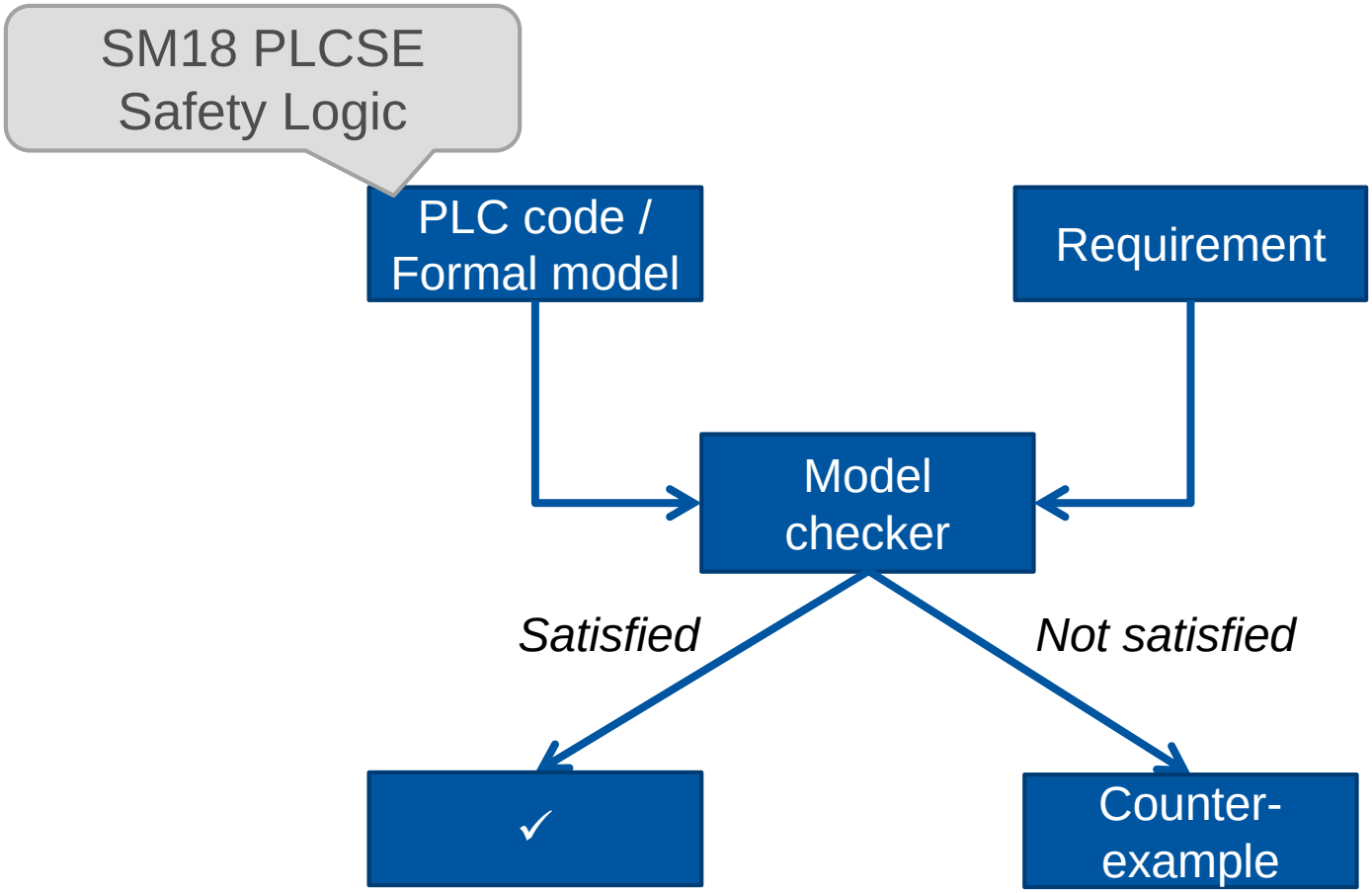
CERN

MŰEGYETEM 1782

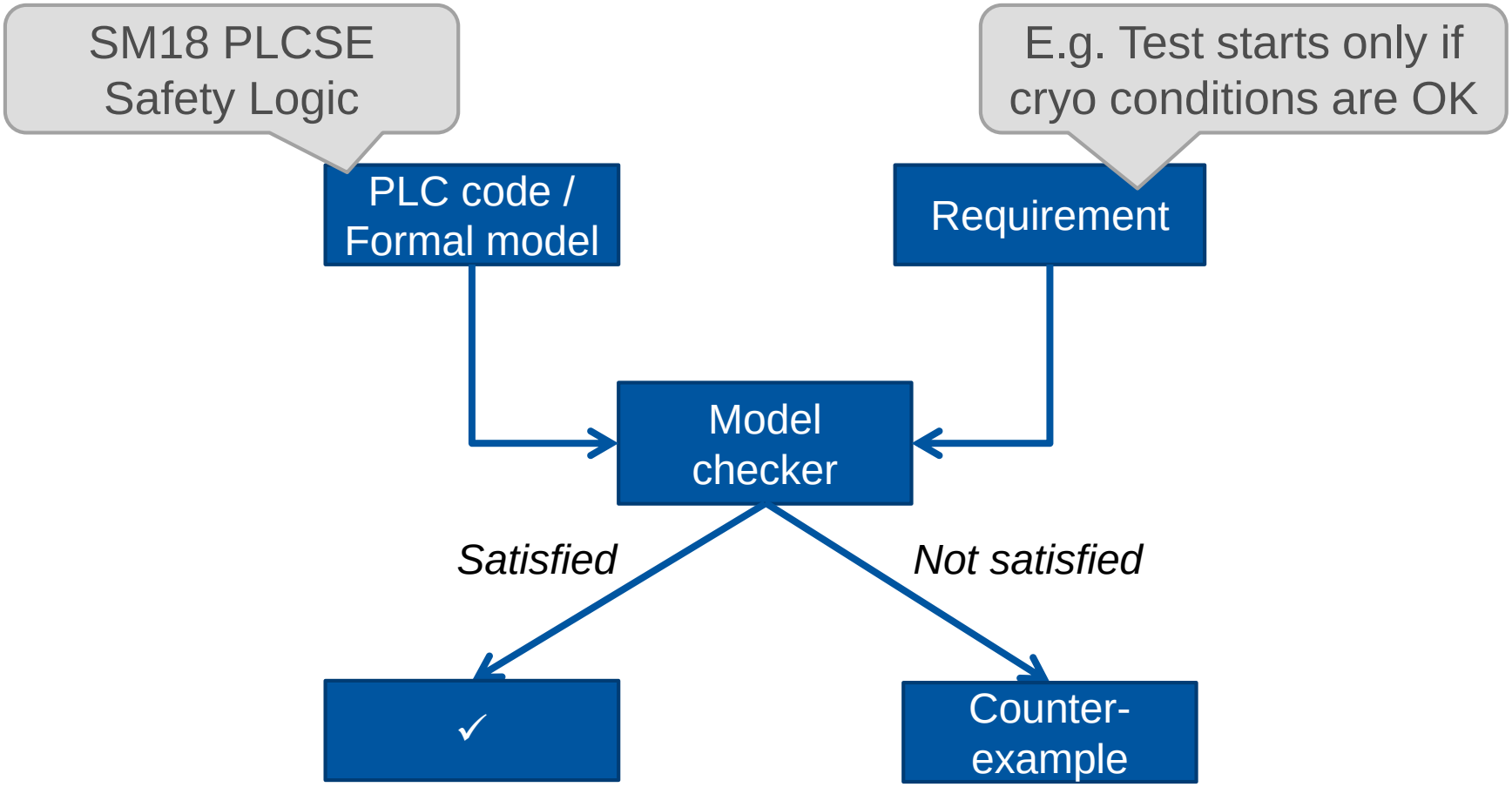
Model checking workflow for SM18



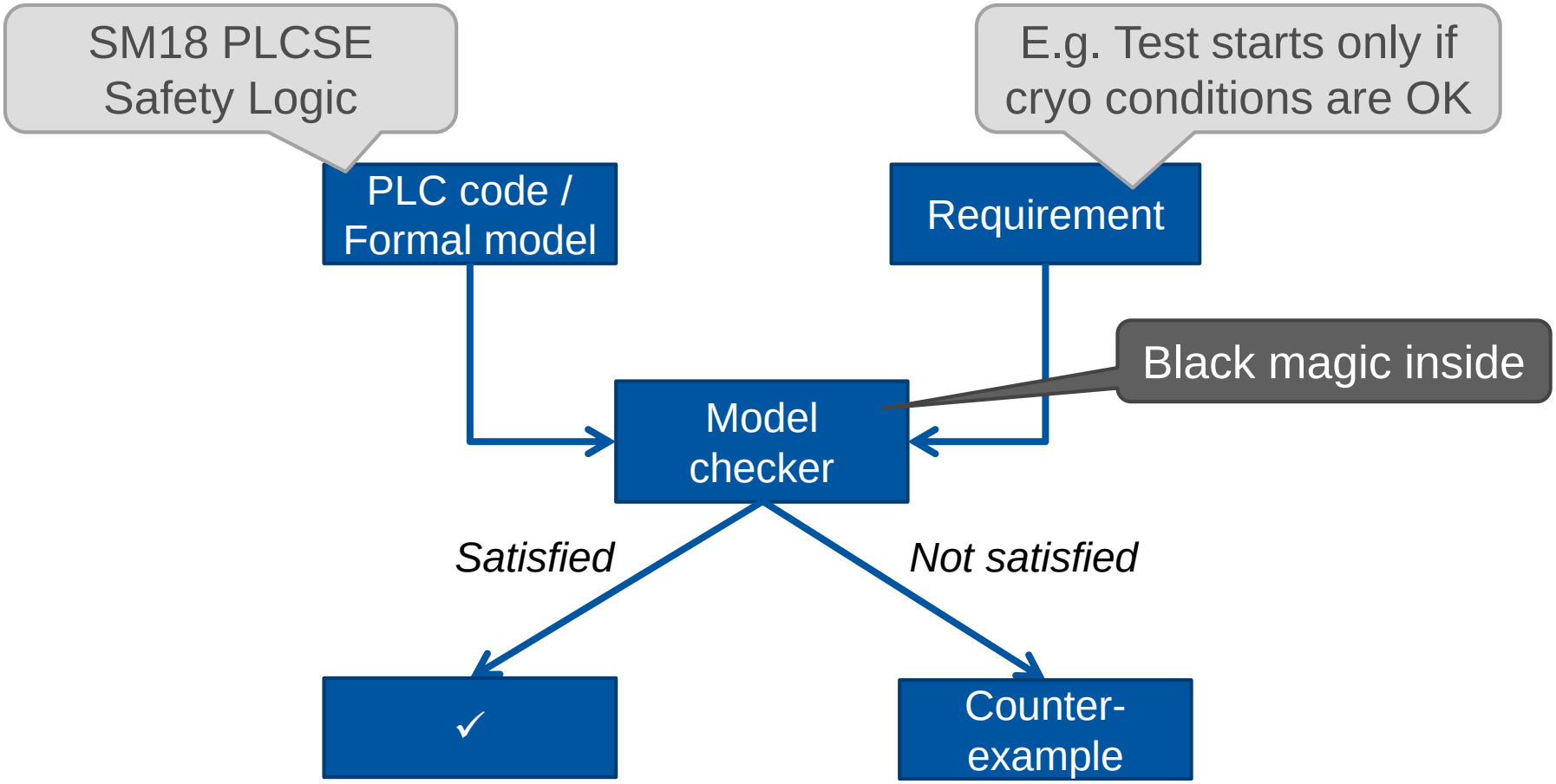
Model checking workflow for SM18



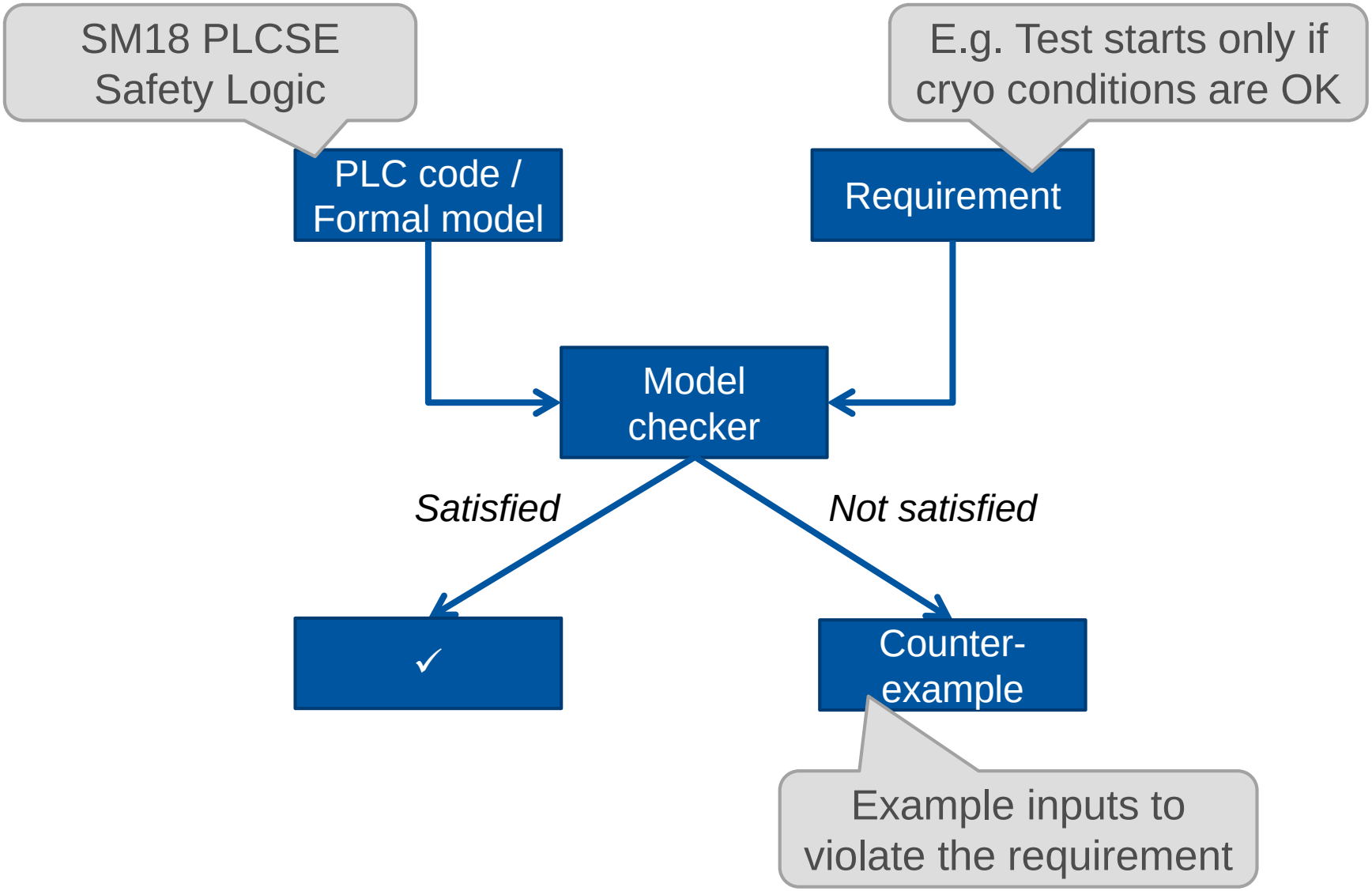
Model checking workflow for SM18



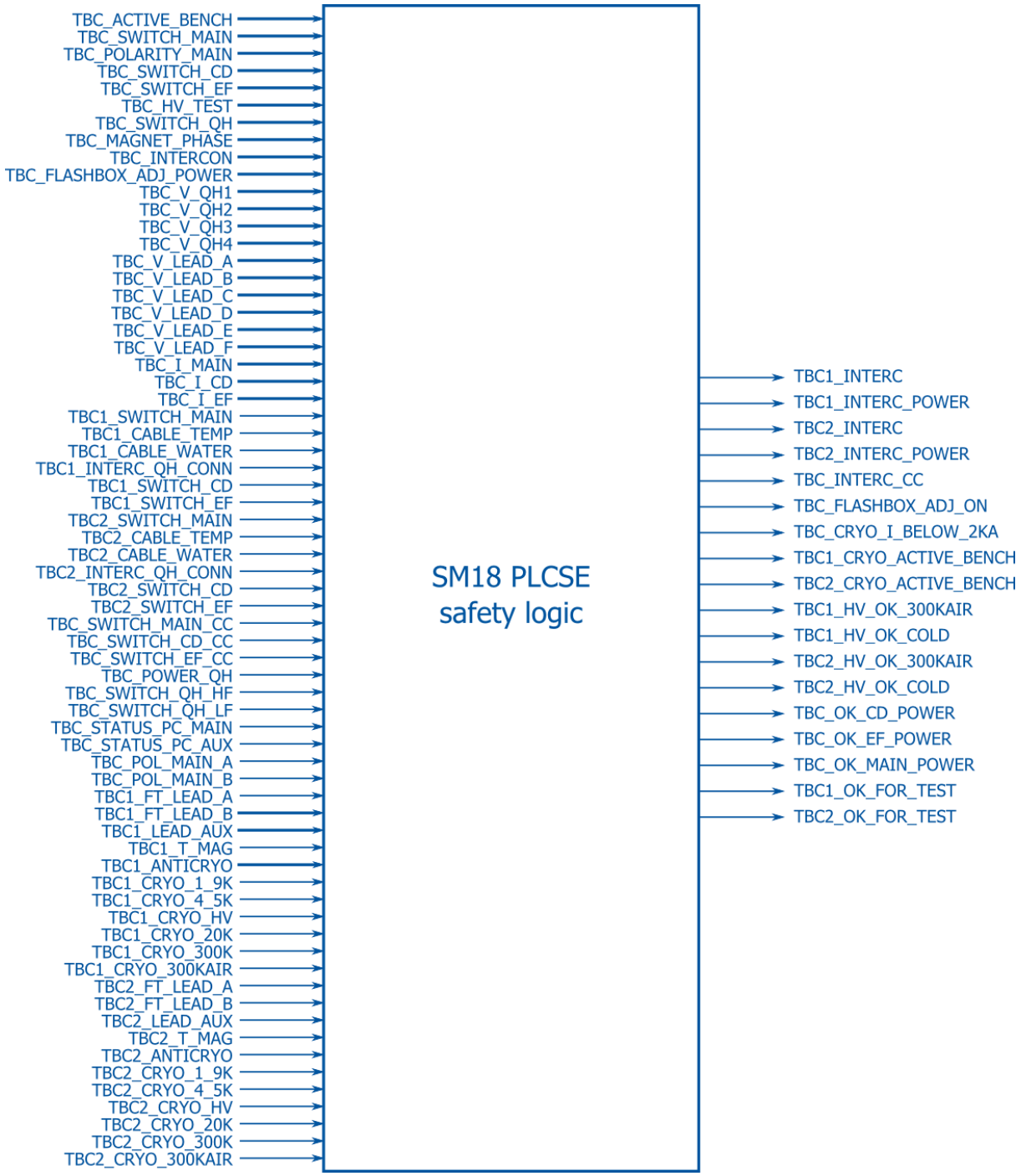
Model checking workflow for SM18



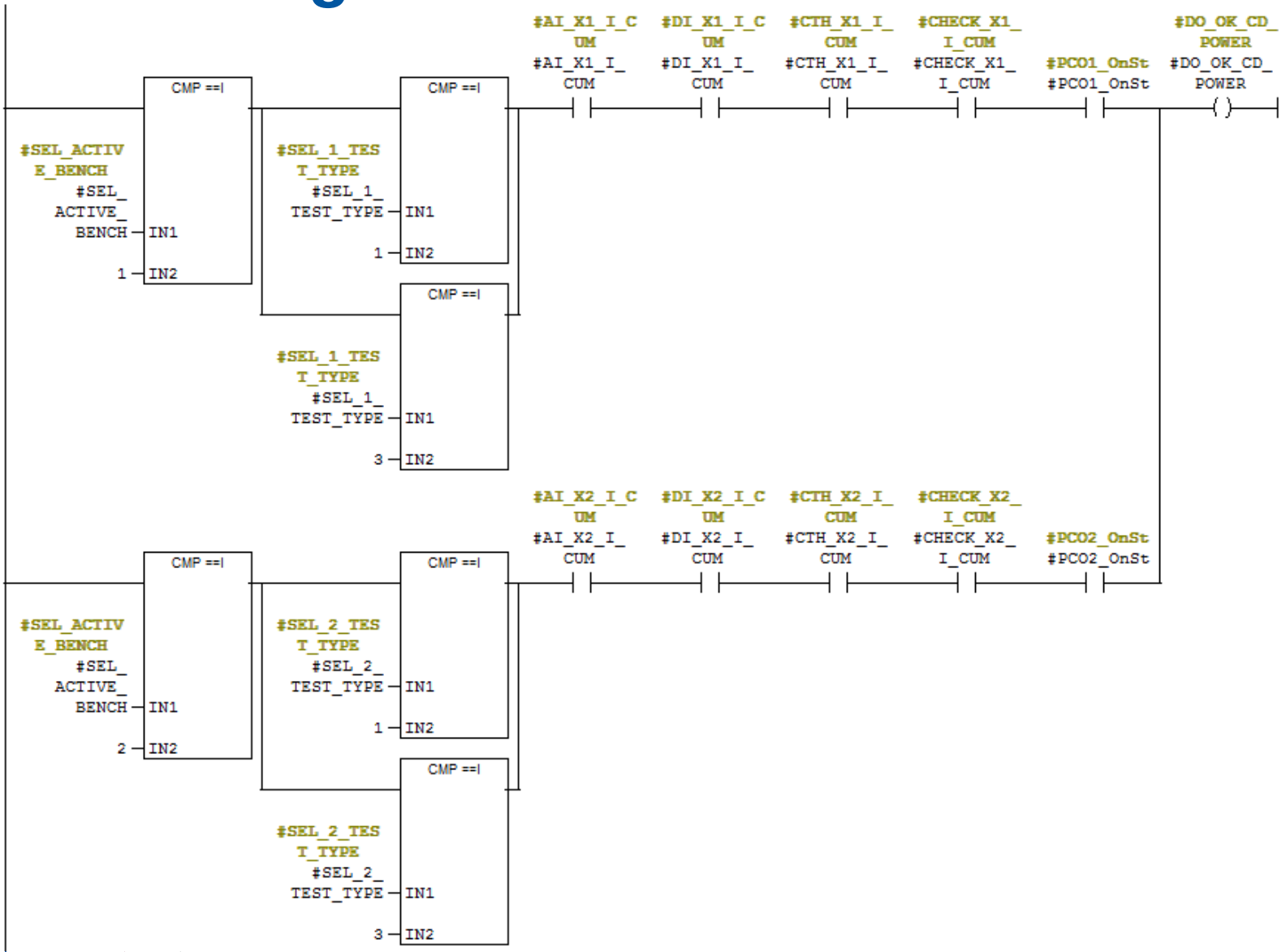
Model checking workflow for SM18




selected test
switch statuses
current voltages
cryo conditions
SM18 PLCSE
safety logic
test allowed

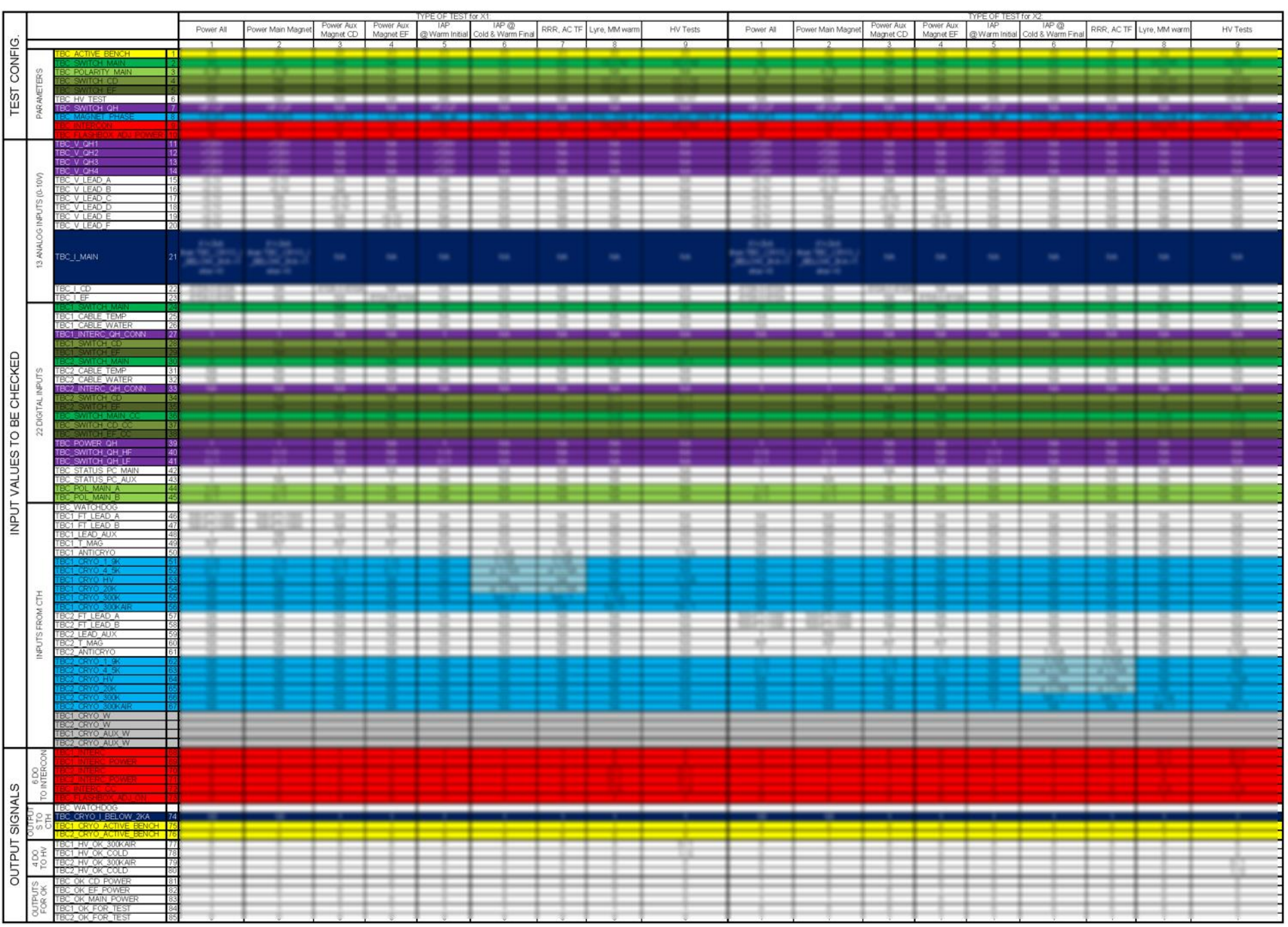

TBC_ACTIVE_BENCH
TBC_SWITCH_MAIN
TBC_POLARITY_MAIN
TBC_SWITCH_CD
TBC_SWITCH_EF
TBC_HV_TEST
TBC_SWITCH_QH
TBC_MAGNET_PHASE
TBC_INTERCON
TBC_FLASHBOX_ADJ_POWER
TBC_V_QH1
TBC_V_QH2
TBC_V_QH3
TBC_V_QH4
TBC_V_LEAD_A
TBC_V_LEAD_B
TBC_V_LEAD_C
TBC_V_LEAD_D
TBC_V_LEAD_E
TBC_V_LEAD_F
TBC_I_MAIN
TBC_I_CD
TBC_I_EF
TBC1_SWITCH_MAIN
TBC1_CABLE_TEMP
TBC1_CABLE_WATER
TBC1_INTERC_QH_CONN
TBC1_SWITCH_CD
TBC1_SWITCH_EF
TBC2_SWITCH_MAIN
TBC2_CABLE_TEMP
TBC2_CABLE_WATER
TBC2_INTERC_QH_CONN
TBC2_SWITCH_CD
TBC2_SWITCH_EF
TBC_SWITCH_MAIN_CC
TBC_SWITCH_CD_CC
TBC_SWITCH_EF_CC
TBC_POWER_QH
TBC_SWITCH_QH_HF
TBC_SWITCH_QH_LF
TBC_STATUS_PC_MAIN
TBC_STATUS_PC_AUX
TBC_POL_MAIN_A
TBC_POL_MAIN_B
TBC1_FT_LEAD_A
TBC1_FT_LEAD_B
TBC1_LEAD_AUX
TBC1_T_MAG
TBC1_ANTICRYO
TBC1_CRYO_1_9K
TBC1_CRYO_4_5K
TBC1_CRYO_HV
TBC1_CRYO_20K
TBC1_CRYO_300K
TBC1_CRYO_300KAIR
TBC2_FT_LEAD_A
TBC2_FT_LEAD_B
TBC2_LEAD_AUX
TBC2_T_MAG
TBC2_ANTICRYO
TBC2_CRYO_1_9K
TBC2_CRYO_4_5K
TBC2_CRYO_HV
TBC2_CRYO_20K
TBC2_CRYO_300K
TBC2_CRYO_300KAIR
SM18 PLCSE
safety logic
TBC1_INTERC
TBC1_INTERC_POWER
TBC2_INTERC
TBC2_INTERC_POWER
TBC_INTERC_CC
TBC_FLASHBOX_ADJ_ON
TBC_CRYO_I_BELOW_2KA
TBC1_CRYO_ACTIVE_BENCH
TBC2_CRYO_ACTIVE_BENCH
TBC1_HV_OK_300KAIR
TBC1_HV_OK_COLD
TBC2_HV_OK_300KAIR
TBC2_HV_OK_COLD
TBC_OK_CD_POWER
TBC_OK_EF_POWER
TBC_OK_MAIN_POWER
TBC1_OK_FOR_TEST
TBC2_OK_FOR_TEST

Ladder Diagram



MŰEGYETEM 1782

Excerpt from the work of R. Speroni



From M. Charrondiere

Problems found *(before putting in production!)*

Problems found *(before putting in production!)*

Requirement misunderstanding

- Recognised while specifying requirements

Problems found *(before putting in production!)*

Requirement misunderstanding

- Recognised while specifying requirements

Functionality problems

- “The [magnet] test should start, but it doesn’t.”

Problems found *(before putting in production!)*

Requirement misunderstanding

- Recognised while specifying requirements

Functionality problems

- “The [magnet] test should start, but it doesn’t.”

Safety problems

- “The [magnet] test **should NOT start**, but it does.”

Problems found *(before putting in production!)*

Requirement misunderstanding

- Recognised while specifying requirements

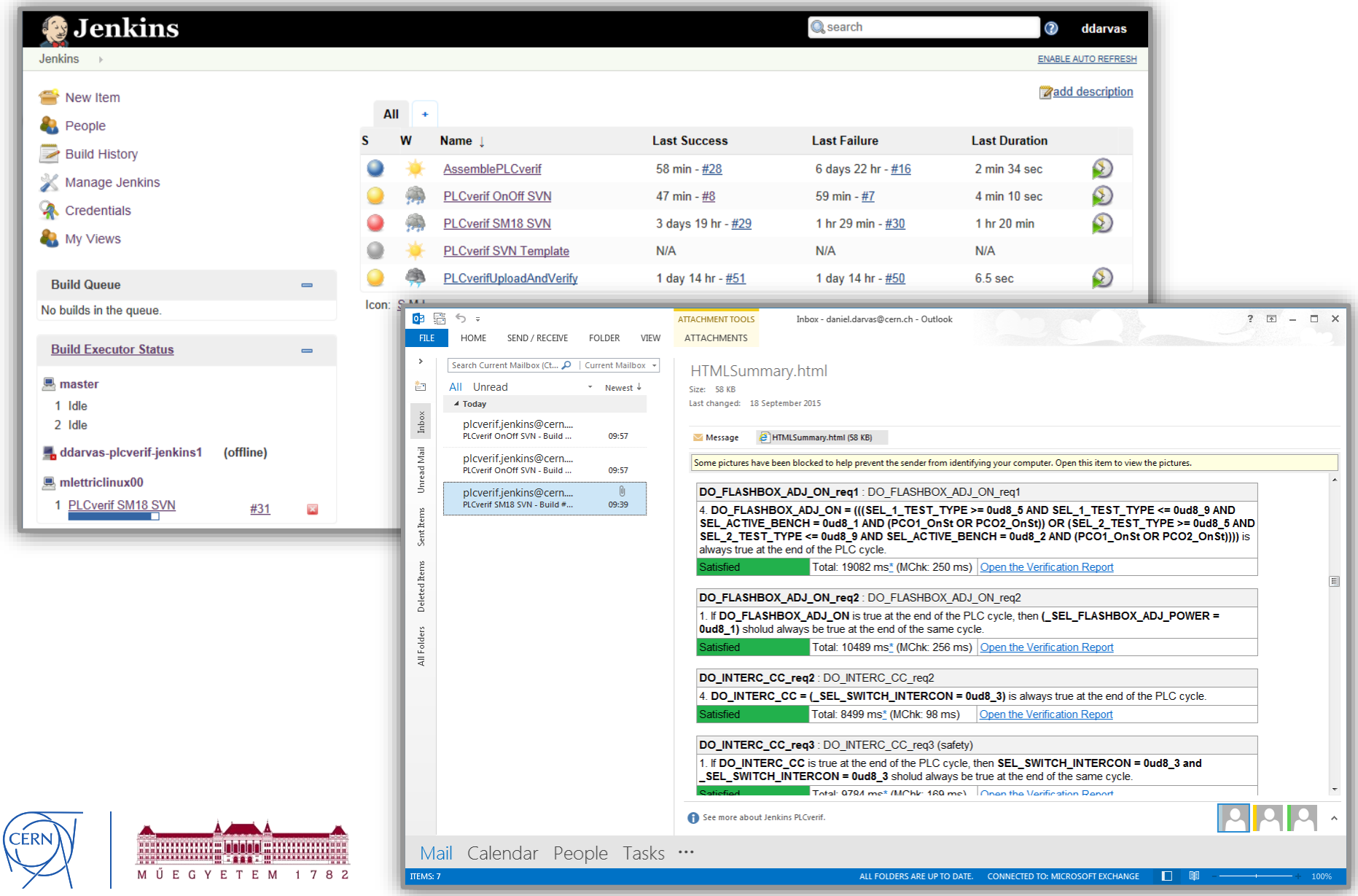
Functionality problems

- “The [magnet] test should start, but it doesn’t.”

Safety problems

- “The [magnet] test **should NOT start**, but it does.”
- Some really hidden:
  65536 input combinations for 1 magnet test scenario
  start should be allowed in 2 of them

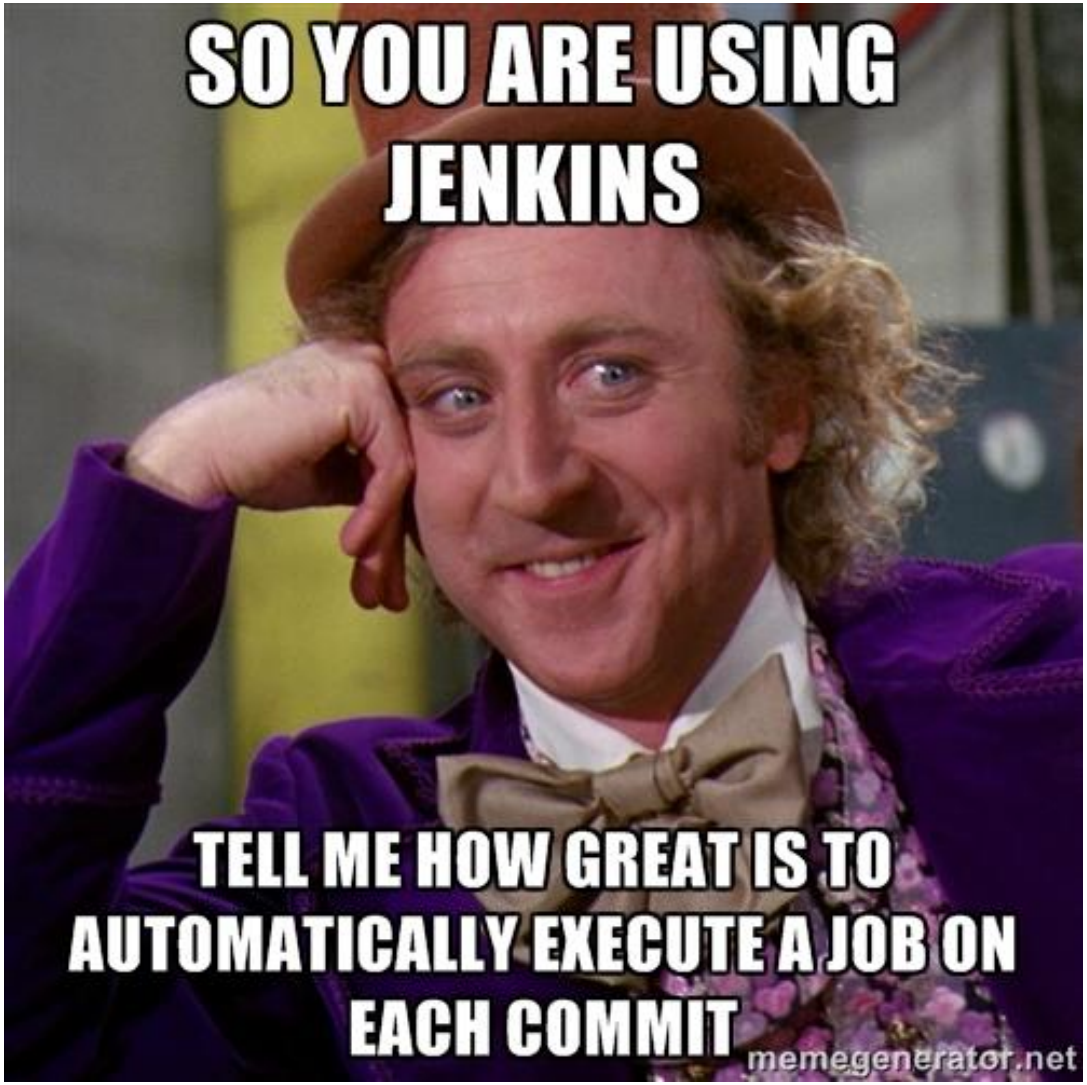
Verification workflow in practice



CERN

MŰEGYETEM 1782

Verification workflow in practice



Based on still from Willy Wonka & the Chocolate Factory,
Source: memegenerator.net

Summary

http://go.cern.ch/**7L9h**
http://cern.ch/**plcverif**

Summary

- “Formal verification is not relevant to industry.” **FALSE**!

http://go.cern.ch/**7L9h**
http://cern.ch/**plcverif**

CERN

MŰEGYETEM 1782

Summary

- “Formal verification is not relevant to industry.” **FALSE**!

- First steps to **apply formal verification** to PLCs
  - **Interesting bugs** found (*with joint effort*)
  - **Critical parts** can be checked
  - **Complementary** to testing

http://go.cern.ch/**7L9h**
http://cern.ch/**plcverif**

CERN

MŰEGYETEM 1782

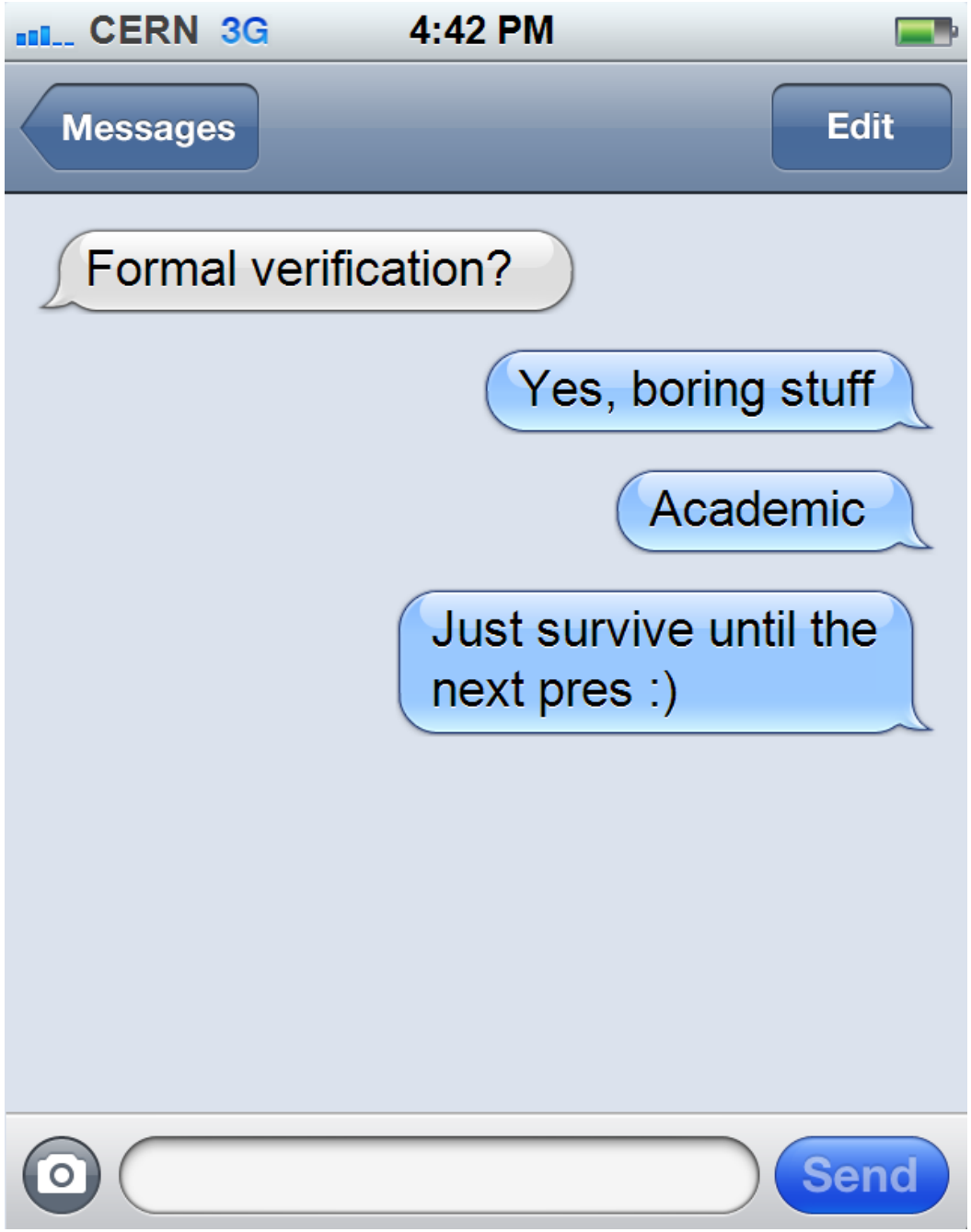
Summary

- "Formal verification is not relevant to industry." **FALSE**!
- First steps to **apply formal verification** to PLCs
  - **Interesting bugs** found (*with joint effort*)
  - **Critical parts** can be checked
  - **Complementary** to testing
- Still long way to go
  - Improving the **performance**
  - **Formal specification**

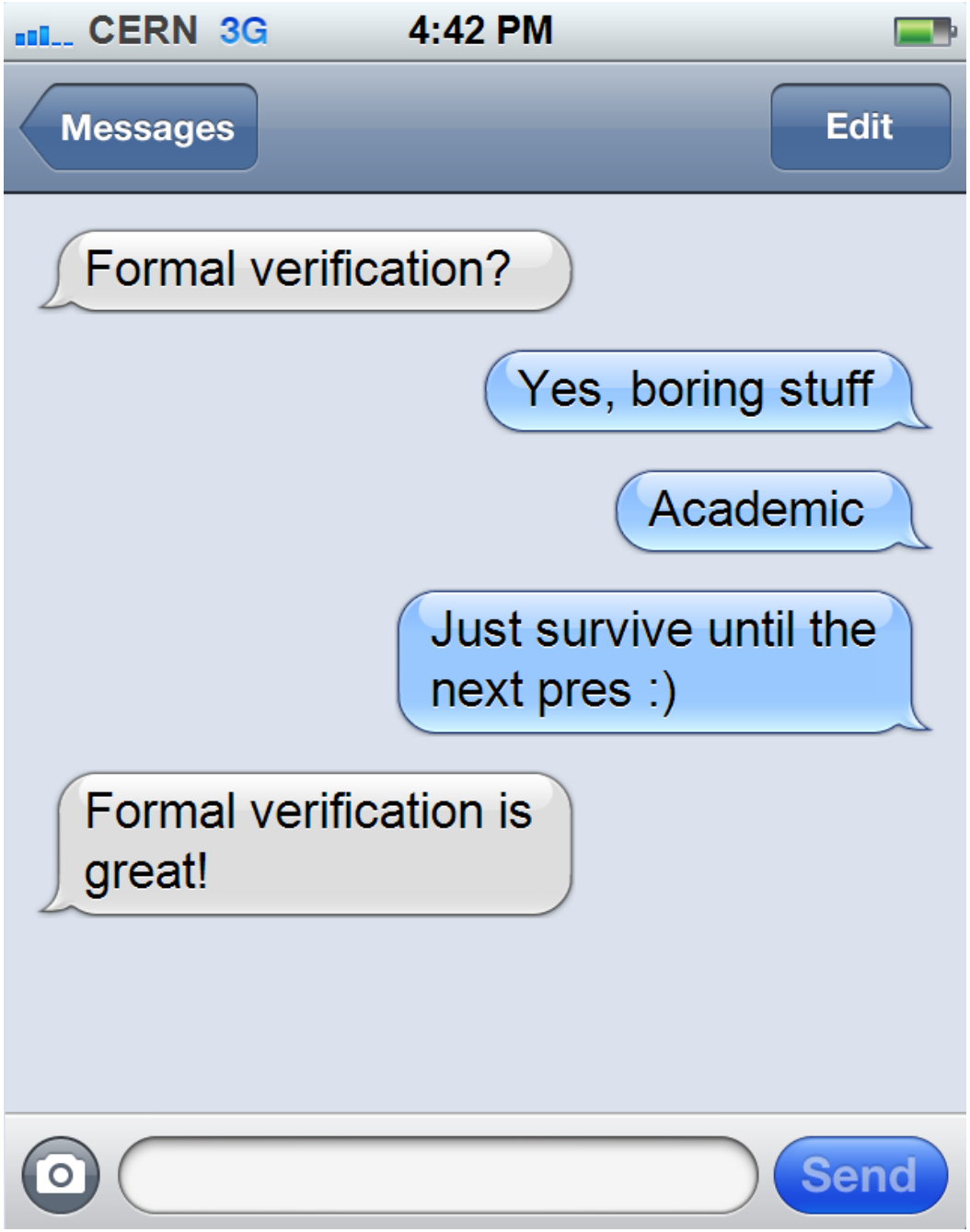
http://go.cern.ch/**7L9h**
http://cern.ch/**plcverif**

CERN

MŰEGYETEM 1782



Source: http://www.iphonetextgenerator.com/



Source: http://www.iphonetextgenerator.com/

Source: http://www.iphonetextgenerator.com/

www.cern.ch