FM23 Homework:
Formal Digital Twin of a Lego Mindstorms Production Plant

Formal Methods for Concurrent and Real-Time Systems, A.Y. 22/23

Livia Lestingi

Aprile 19th, 2023

00. Project Goals

1. Strengthen your modeling skills:

given an informal description of the system, how can we turn it into a formal model?

00. Project Goals

1. Strengthen your modeling skills:

given an informal description of the system, how can we turn it into a formal model?

2. Exploit formal verification to analyze a system:

the theoretical background is vital, but how can we put it to use?

00. Project Goals

1. Strengthen your modeling skills:

given an informal description of the system, how can we turn it into a formal model?

2. Exploit formal verification to analyze a system:

the theoretical background is vital, but how can we put it to use?

3. Practice substantiating your work:

how can you convince your reader/listener that you made the most reasonable and effective choices?

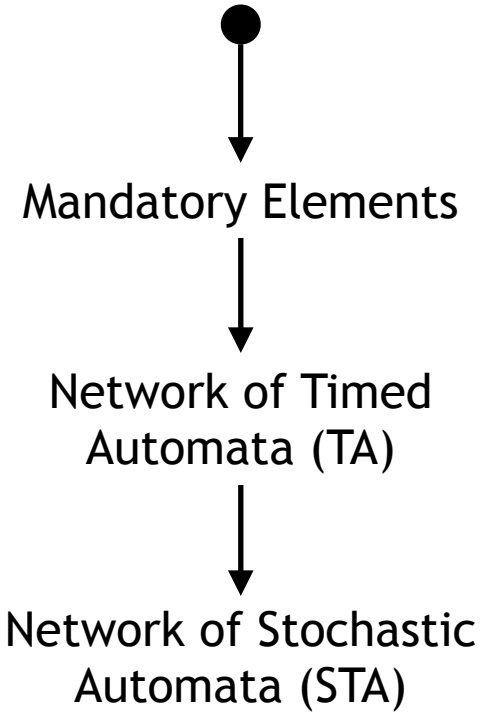
00. Project Goals → Deliverables

1. Strengthen your modeling skills → Formal Model

2. Exploit formal verification to analyze a system → Experimental Results

3. Practice substantiating your work → Written Report

01. Formal Model



- The main output of the project is the **formal model**.
- Some entities/features (explained in more detail in the upcoming slides) must be **mandatorily** modeled to get a sufficient score.
- The NTA accounts for up to 90% of the total score (i.e., $\leq$ 27/30). Stochastic features are required to get the full score (i.e., > 27/30).

01. Formal Model: Context

PHYSICAL SYSTEM

DATA

KNOWLEDGE

DIGITAL TWIN

01. Formal Model: Context

PHYSICAL SYSTEM

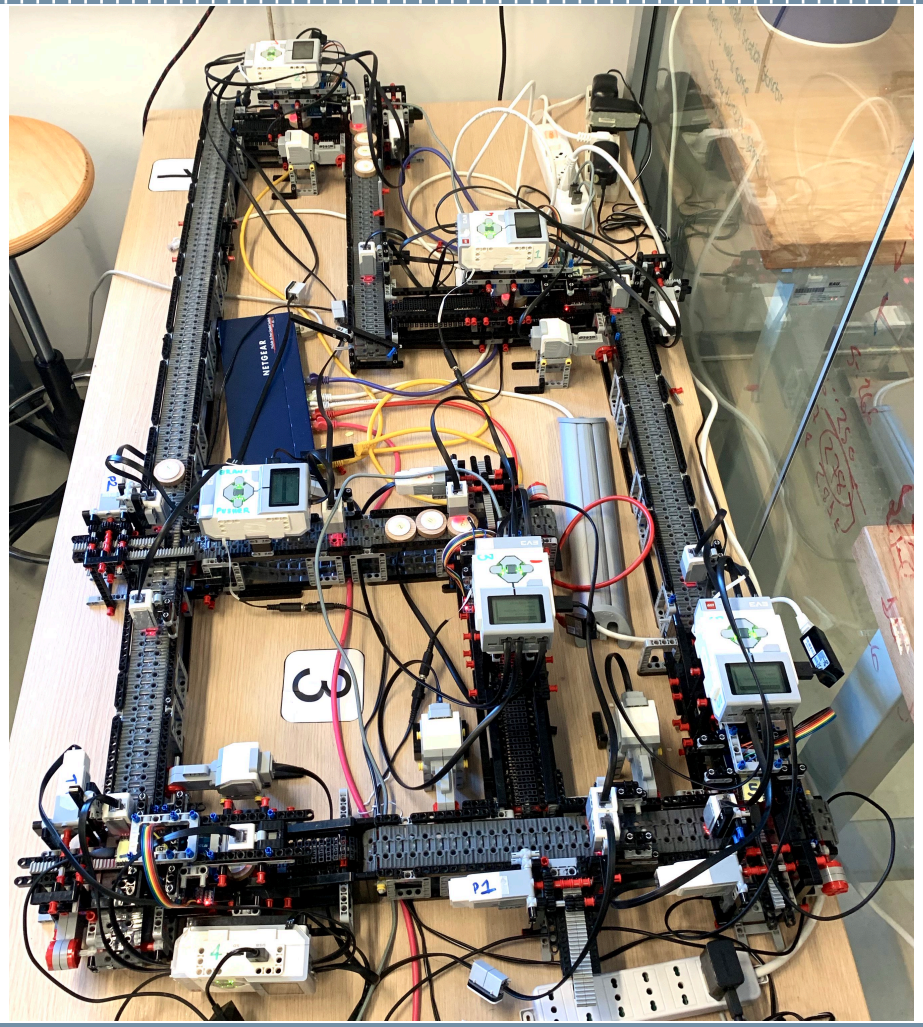
DIGITAL TWIN

DATA

KNOWLEDGE

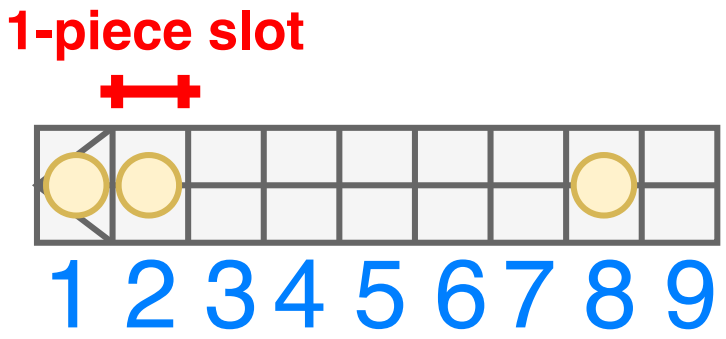
- Is it possible to create a **formalization** of the digital twin?
- Can we exploit it to formally **verify** properties of the digital twin?
- What **knowledge** of the real system can be extracted from verification results?

01. Formal Model: Context



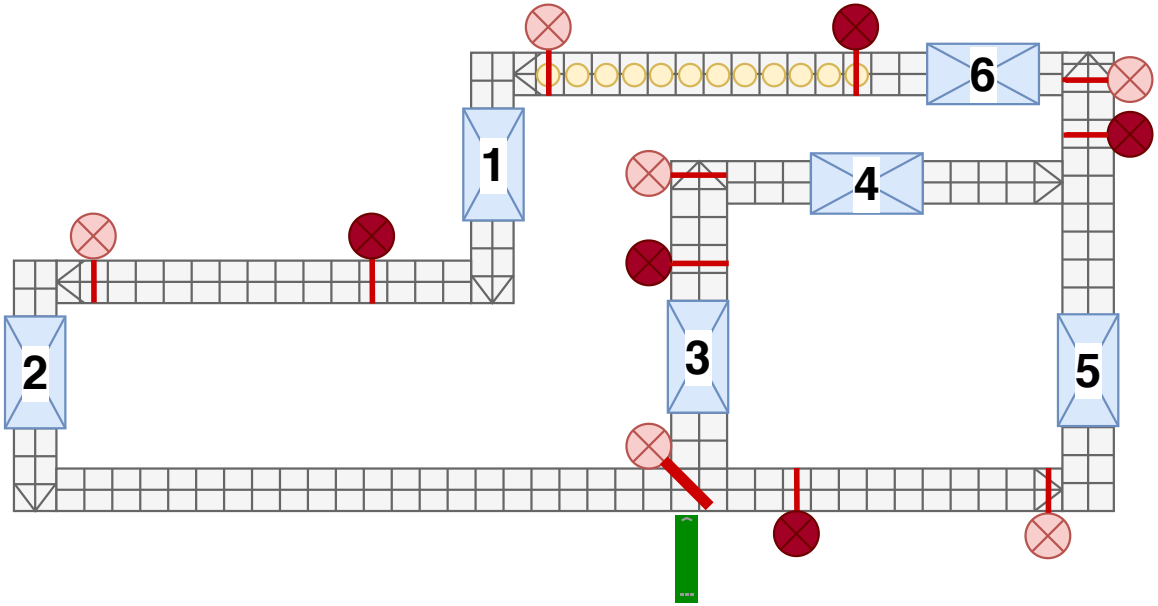
Production plant built with Lego Mindstorms,
Digital Twin Lab@DMec

01. Formal Model: Conveyor Belt

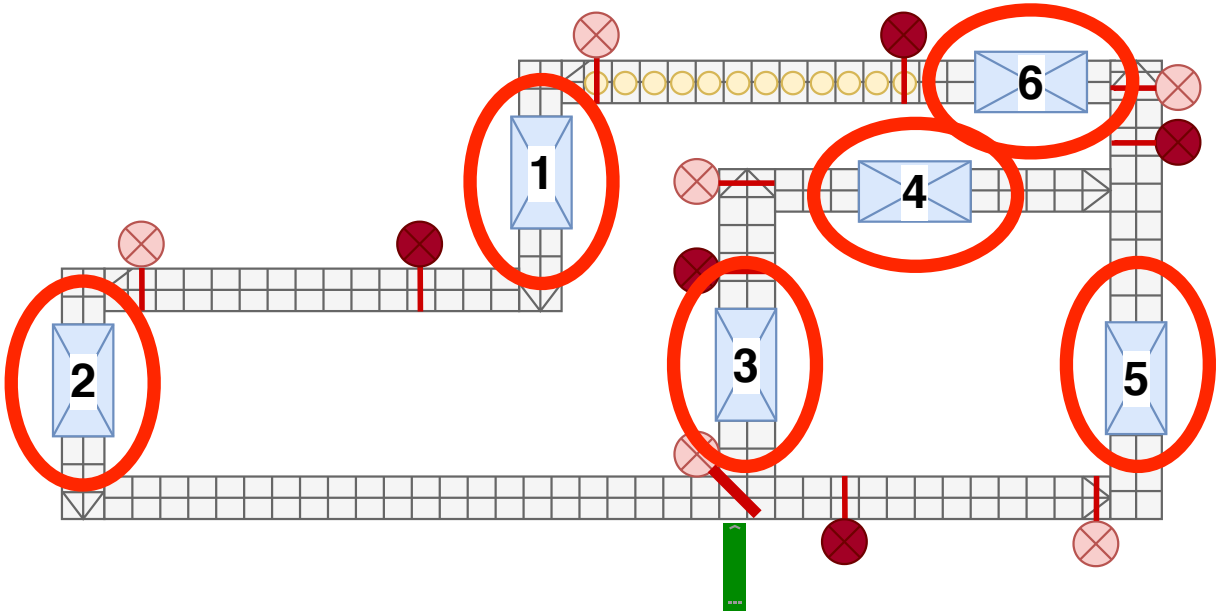


- Conveyor belt segments transport **workpieces** (the yellow circles)
- Divided into **slots** (numbered in the figure)
- They move in one direction (in this case, right to left) (**speed** expressed as [slots/time unit])
- Pieces advance to the next slot only if free (for example, the piece in 2 cannot move whereas the one in 8 can)

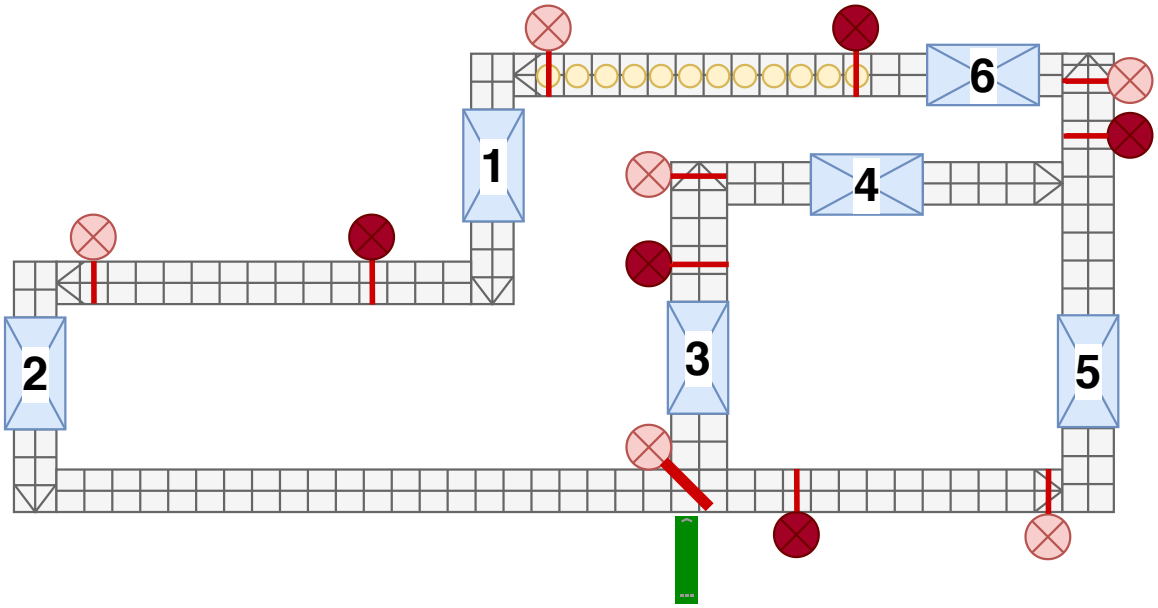
01. Formal Model: Processing Station



01. Formal Model: Processing Station

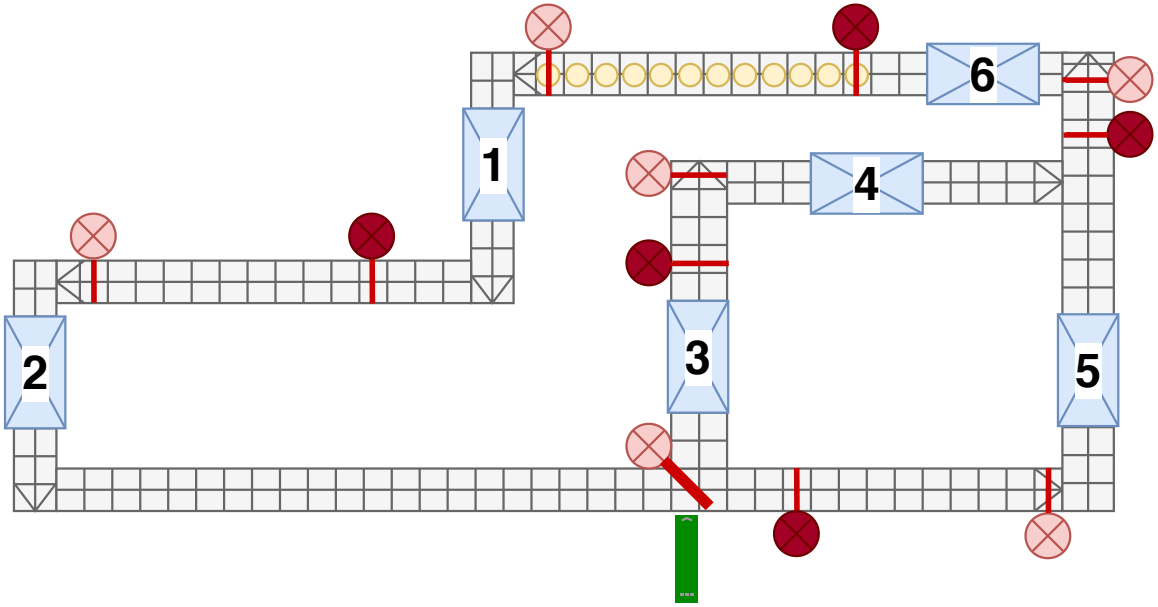


01. Formal Model: Processing Station



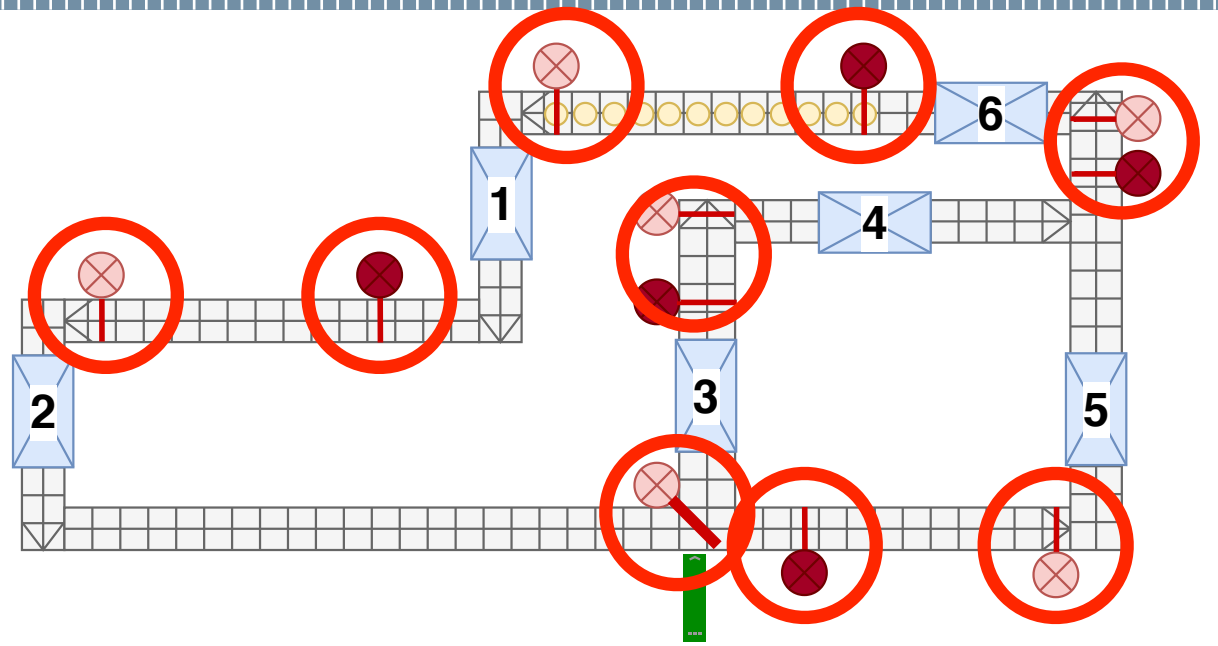
- Workpieces stop within stations to be processed
- Each station fits only one piece at a time

01. Formal Model: Processing Station

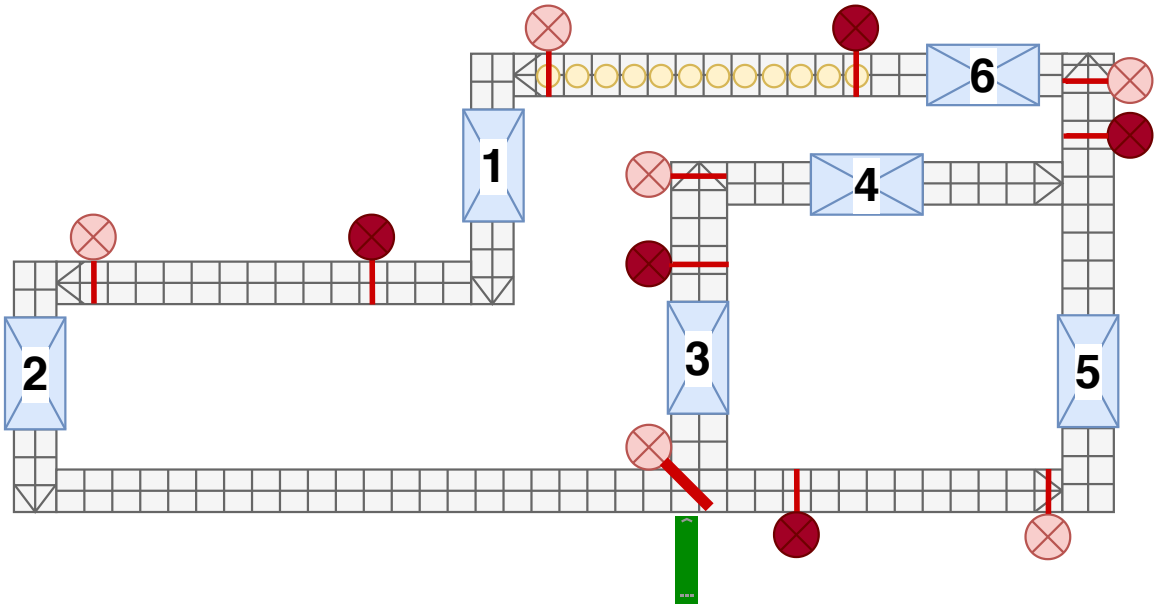


- The **time** a piece spends within a station can change between different stations
- **Stochastic feature:** the processing time for each station is not fixed, but **normally** distributed.

01. Formal Model: Laser-based Sensor

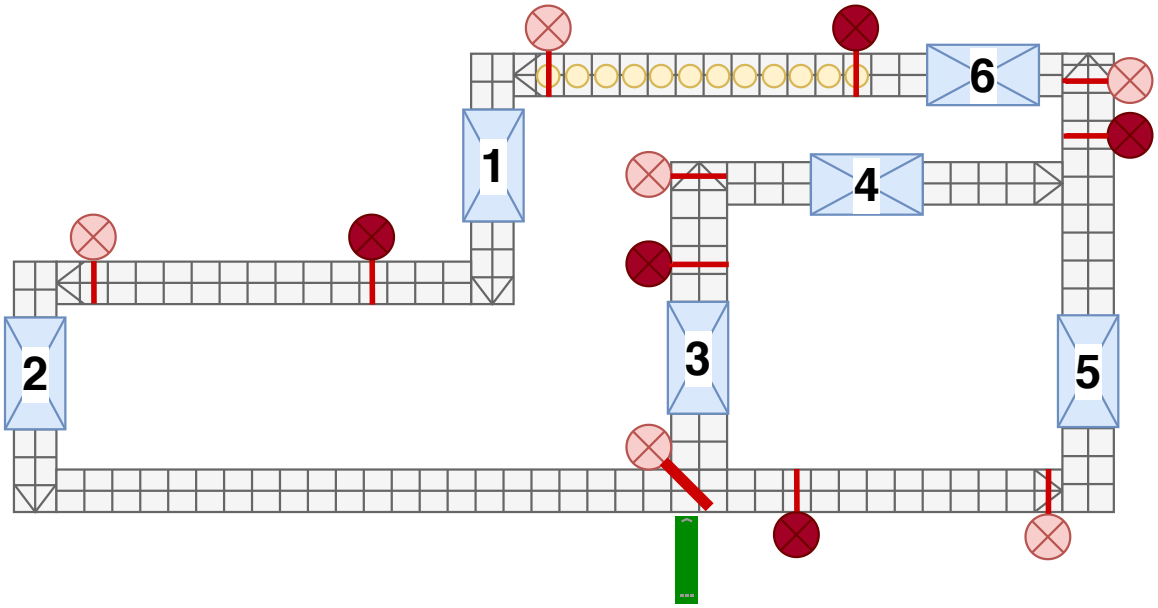


01. Formal Model: Laser-based Sensor



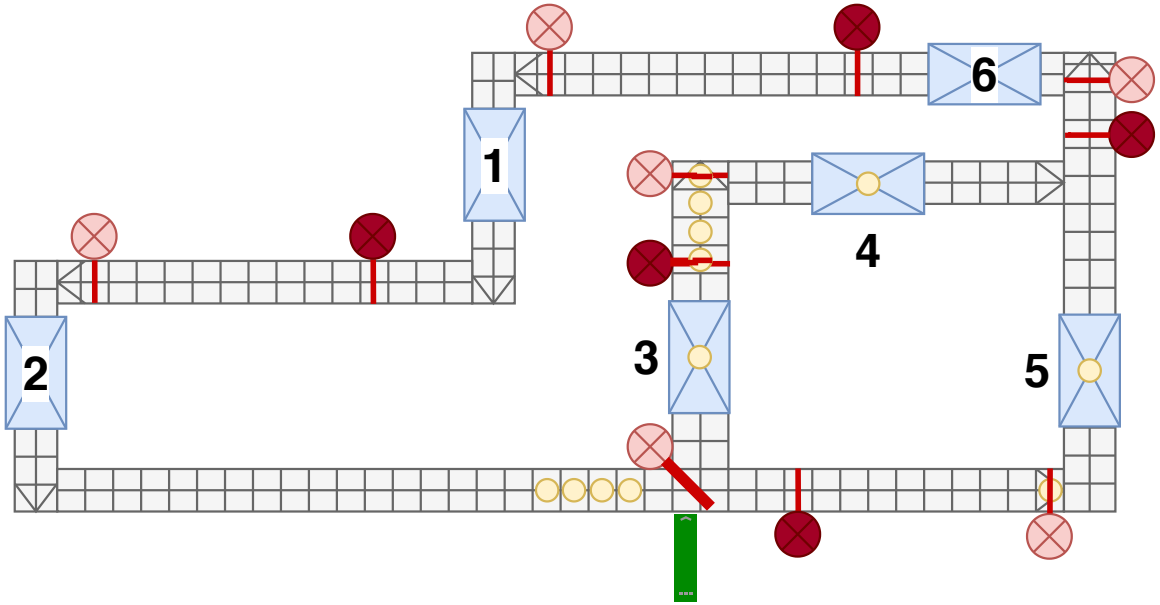
- Sensors guard the **entrance** to a station (lighter red circles). If a piece is detected on the corresponding slot, it cannot move forward unless the station downstream is free.

01. Formal Model: Laser-based Sensor



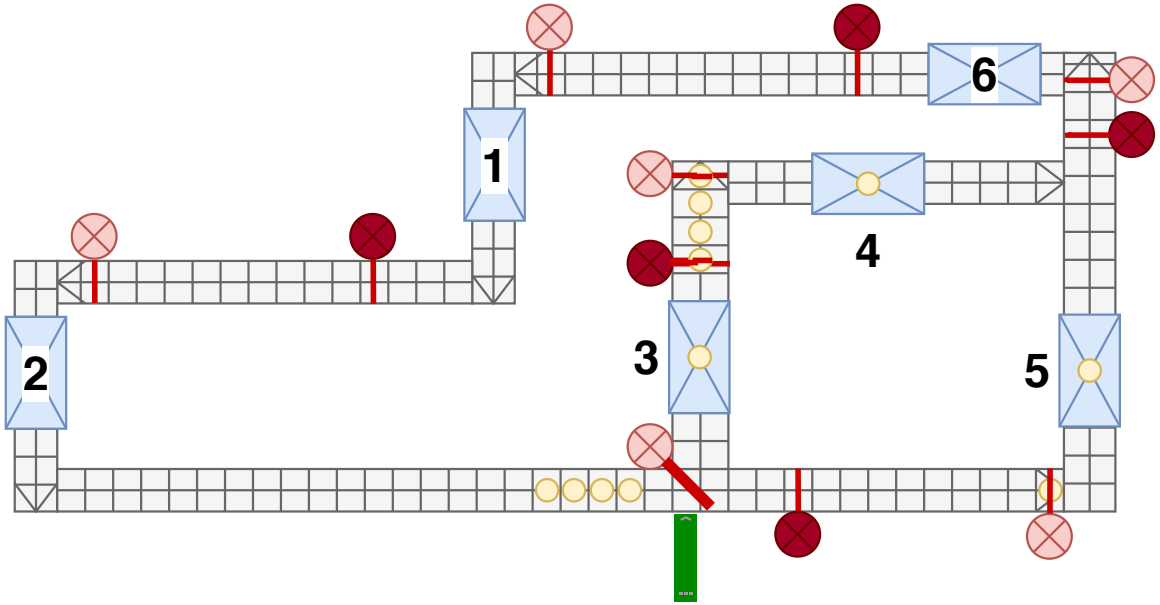
- Sensors also guard how many pieces are **queueing** to enter a station (darker red circles). If a piece is detected on the corresponding slot, the station upstream cannot release any new piece even if processing time has elapsed.

01. Formal Model: Laser-based Sensor



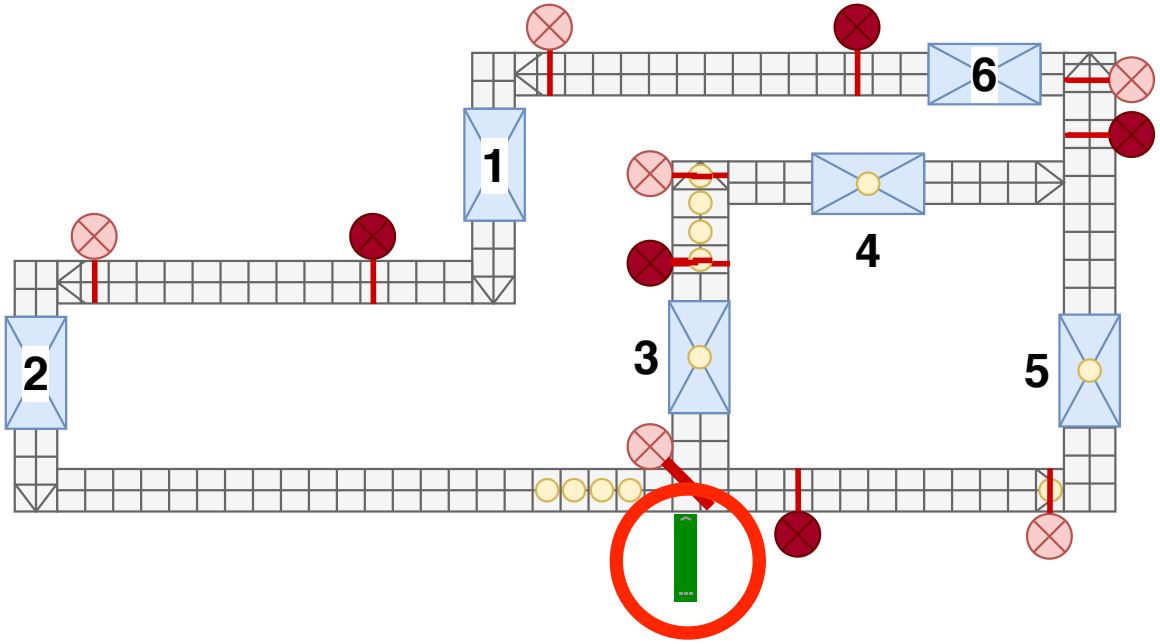
- For example, the piece waiting outside station 5 cannot move forward because the station is busy.
- For example, station 3 cannot release the piece because the queue to station 4 is full.

01. Formal Model: Laser-based Sensor

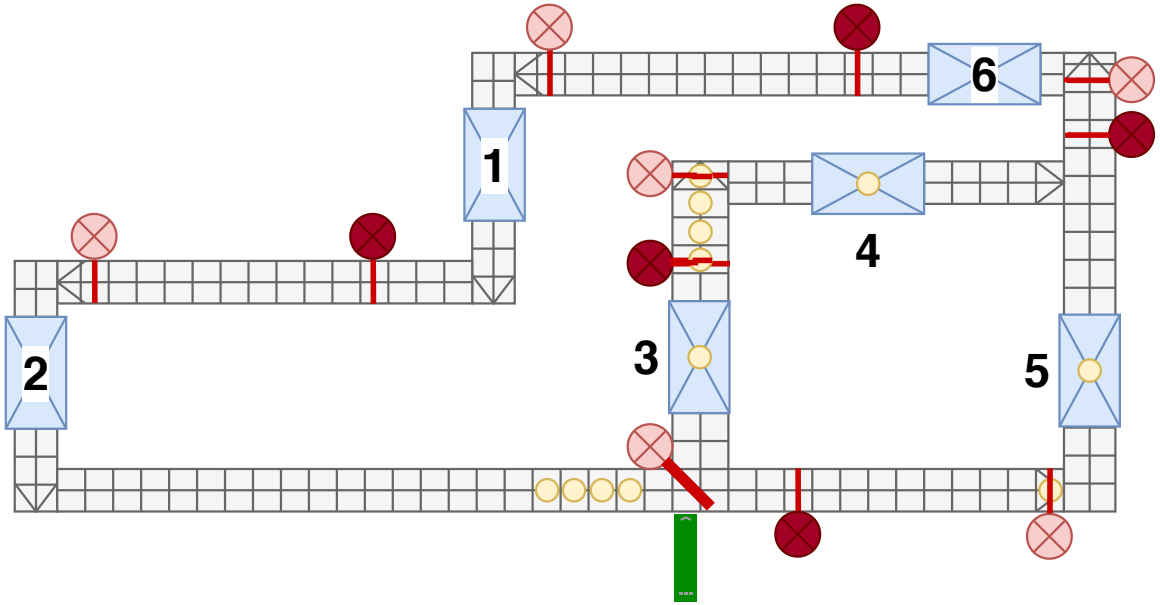


- **Stochastic feature:** at each time unit, with a certain probability, sensors relay **faulty** measurements (i.e., detect a piece that is not there or vice-versa). The probability may vary between different sensors.

01. Formal Model: Flow Controller



01. Formal Model: Flow Controller



- The green bar pushes workpieces into one **branch** of the plant or the other.
- You design the **policy** with which it chooses the branch.

02. Formal Verification: Properties

- The **mandatory** property to be verified is:

P1. It never happens that a station holds more than 1 piece.

P2. It never happens that two pieces occupy the same belt slot.

P3. No queue ever exceeds the maximum allowed length.

P4. The plant never incurs in deadlock.

02. Formal Verification: Properties

- Whether these properties hold or not depends on the plant's **configuration**. Specifically, on the following parameters:

  1. the conveyor **belt** translational **speed**;
  2. the **processing time** of each **station;**
  3. the **number of pieces** circulating in the plant;
  4. the maximum **length** (if any) of the **queue** upstream of each station;
  5. the **branch-switching policy**.

- Present (at least) **three** significant NTA configurations highlighting relevant behavioral aspects of the system.

02. Formal Verification: Stochastic Properties

- As for the stochastic version of the model, calculate the **probability** of properties P1-4 holding within a **time bound** (the time bound must also be properly sized).

- The NSTA behavior further depends on the following parameters:
  - The stations' processing **time distributions**;
  - The sensors' **fault probabilities**.

- Present (at least) **three** significant NSTA configurations highlighting relevant behavioral aspects of the system.

02. Formal Verification: Tool

- The formal model should be created using the **Uppaal** tool:
  - More info during the student presentation session.

02. Formal Verification: Tool

- You can find references in the Getting Started section of the website:
  - Reading the Tutorial (also for the SMC extension) is highly recommended!

- When in doubt, the Documentation is your friend:
  - It contains all the reference for:
    - GUI of the tool
    - System Description
    - Requirement Specification
    - ...everything you need!

03. Written Report

- You must **mandatorily** deliver:
  - Uppaal .xml source file for your model:
    Beware: I will re-run the queries you describe in your report, so make sure the model is experiment-ready and queries are saved in the verifier section!

03. Written Report

- You must **mandatorily** deliver:
  - Uppaal .xml source file for your model:
    Beware: I will re-run the queries you describe in your report, so make sure the model is experiment-ready and queries are saved in the verifier section!
  - The project report (possibly a .pdf):
    Max 10 pages: front cover, index page, bibliography, appendices […] do not count.
    No constraints on the template (but you can take this chance to practice with LateX before writing your thesis…).

03. Written Report

- Tips for the report:
  - Do not just enumerate variables, locations, clocks […]: what really matters is your **reasoning** and critical thinking.

    For example: "*This feature is modeled as a two-dimensional array of integers.*"…fine, but **why**? There can be more than one valid motivation, but please provide one.

  - I will upload excellent reports from previous years to provide a positive source of inspiration.

04. Final Remarks

- The deadline is **July 20** (**June 22** for the early session).

- It is preferable (for you) to carry out the project in teams of 2/3/4 people.

- My job is to support you and provide directions: please drop an email when you need it.

…I cannot solve bugs for you though :) (that's cheating)

- Do not open up with "*Dear prof, we have problems, can we schedule a meeting?*": please describe your issue, and then I will assess if it can be solved via email or if it requires a meeting.

Questions? :)