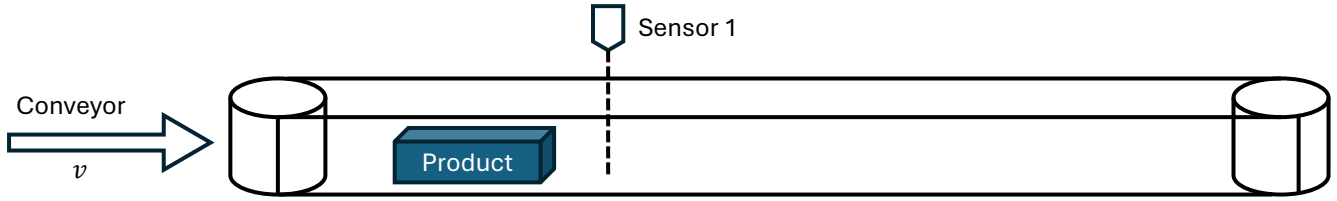
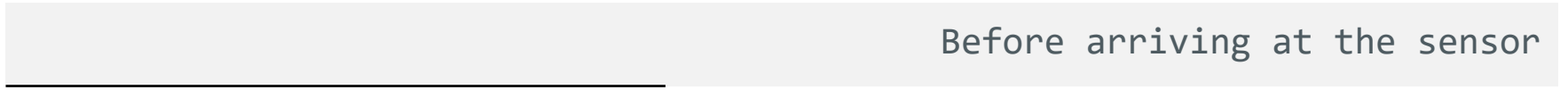
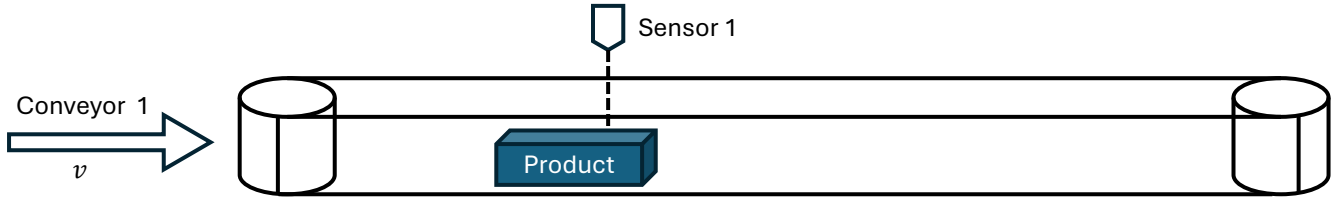


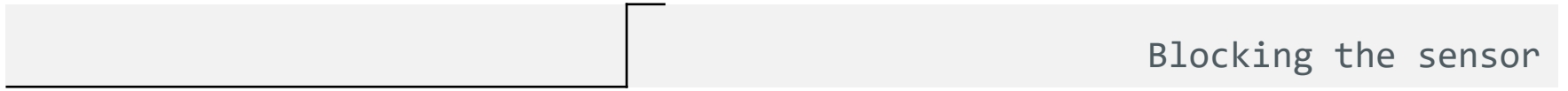
Sensor reading = 0 at T_1



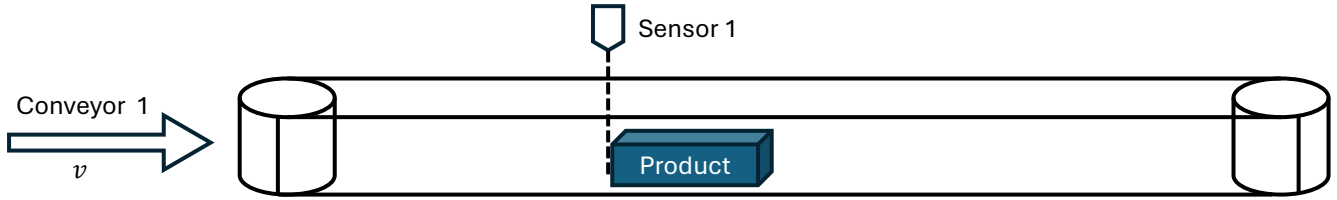
Before arriving at the sensor



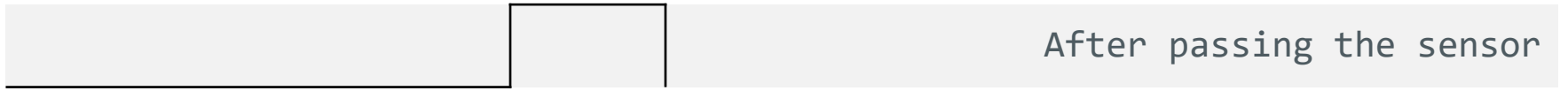
Sensor reading = 1 at T_2



Blocking the sensor



Sensor reading = 0 at T_3



After passing the sensor

Defect product, need to CLOSE the gate

Normal product, need to OPEN the gate

The length of normal product is 4.8cm~5.2cm(inclusive)

Product#1
56ms(5.6cm)

Product#2
53ms(5.3cm)

Product#3
42ms(4.2cm)

Product#4
49ms(4.9cm)

You can access the sensor reading through *conv1inf.sensor_reading_r* for conveyor 1

Defect product, need to CLOSE the gate

Normal product, need to OPEN the gate

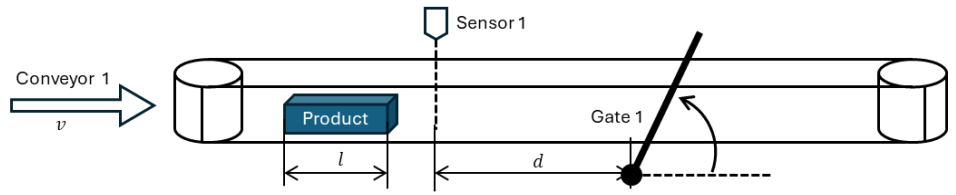
The length of normal product is 4.8cm~5.2cm(inclusive)

Product#1
56ms(5.6cm)

Product#2
53ms(5.3cm)

Product#3
42ms(4.2cm)

Product#4
49ms(4.9cm)



200ms

Arrive at sensor of product #1

200ms

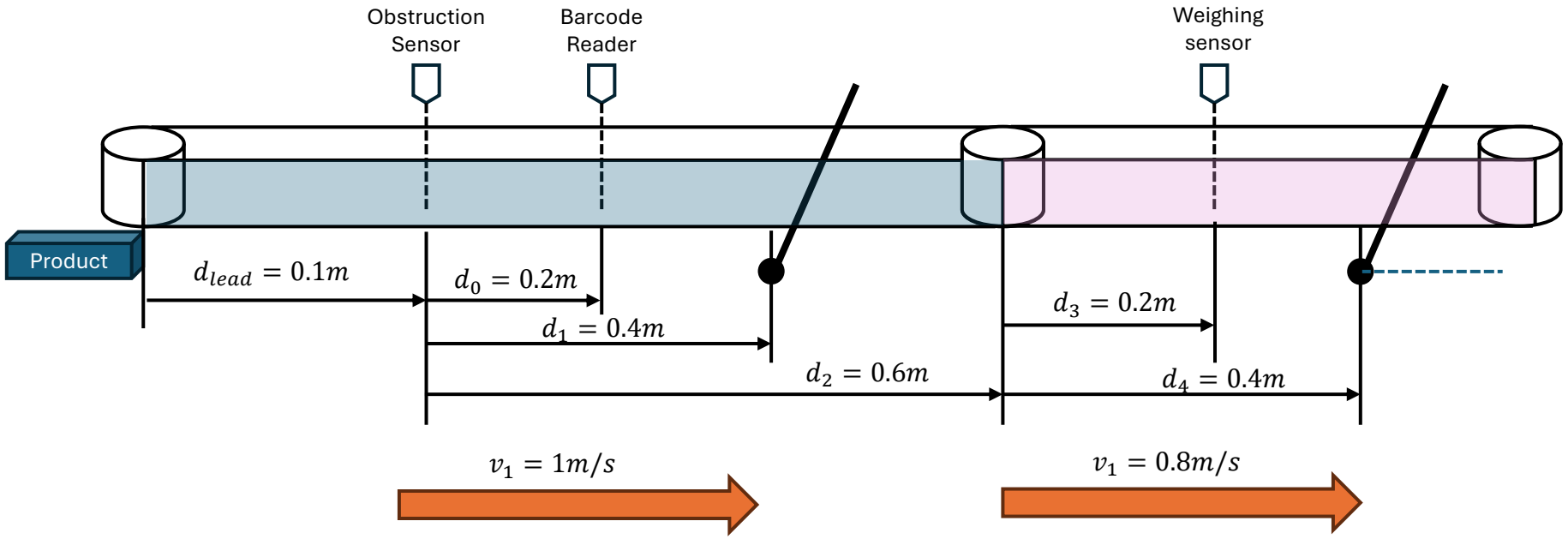
Arrive at the gate of product #1

200ms

Arrive at sensor of product #4

Arrive at the gate of product #4

1.When the product arrives at the gate, the gate should already be correctly set according to the product's length.
2.You can open/close the gate before the product arrives, but it must not affect the passage of the previous product.
3.The open/close state of the gate should be maintained until the product has passed.
4.You can set the gate status of Conveyor 1 through the conv1inf.gate_signal_rw.



If production length l is 5 cm long and the product contacts the conveyor at $t_0 = 0$, can you calculate the six time points

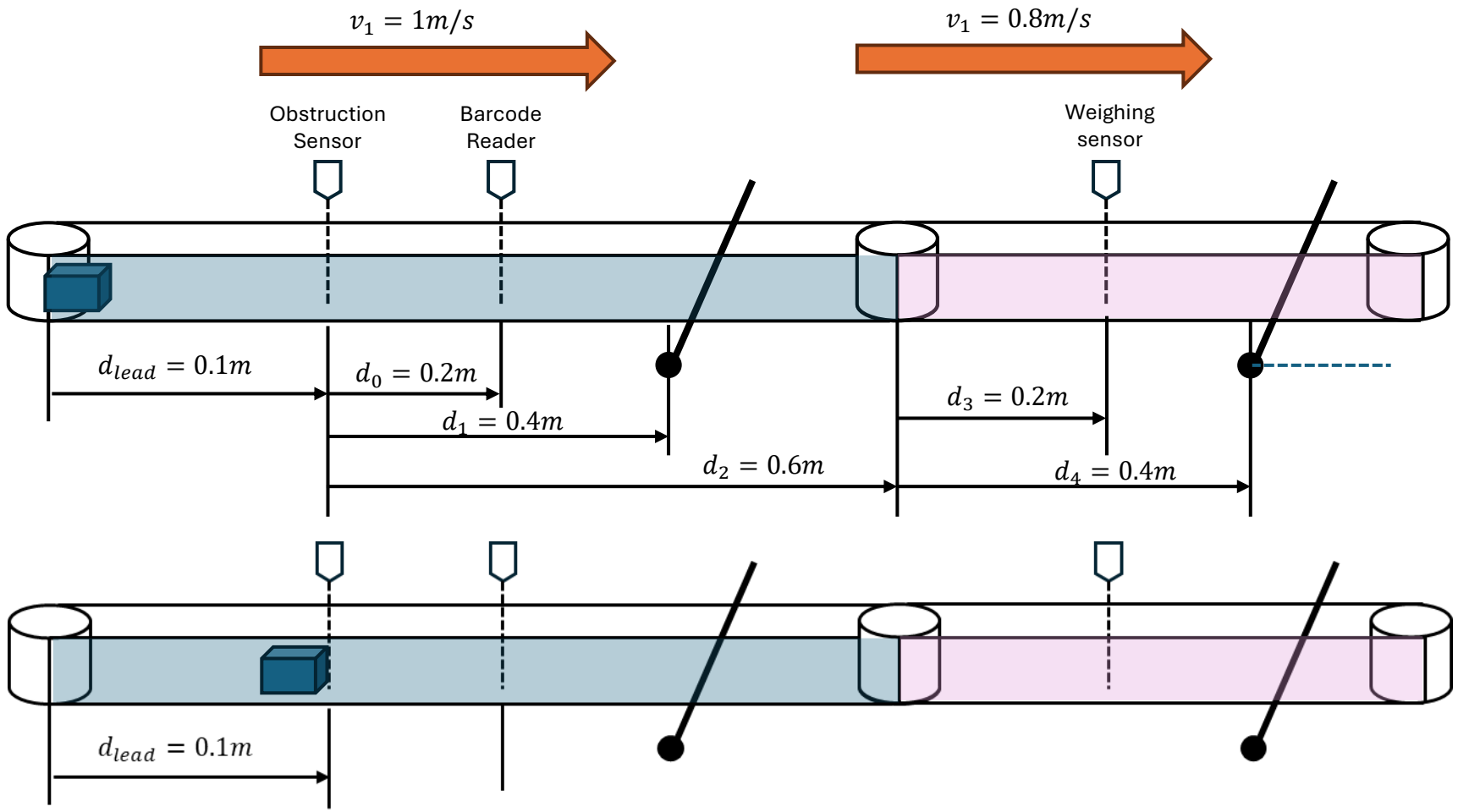
$t_1 t_2$ at which the product reaches and leaves the Obstruction Sensor,

t_3 its centre reaches the Barcode Reader,

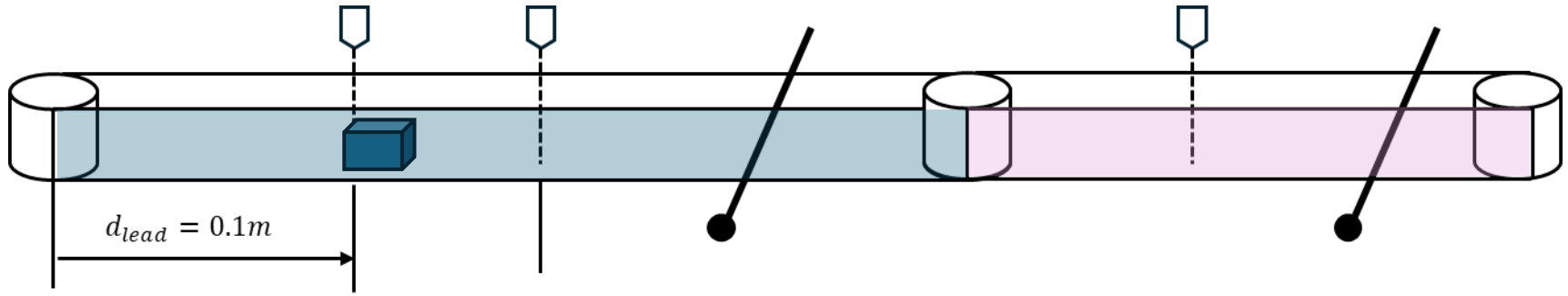
t_4 its front edge reaches Gate 1,

t_5 its centre reaches the Weighing Sensor,

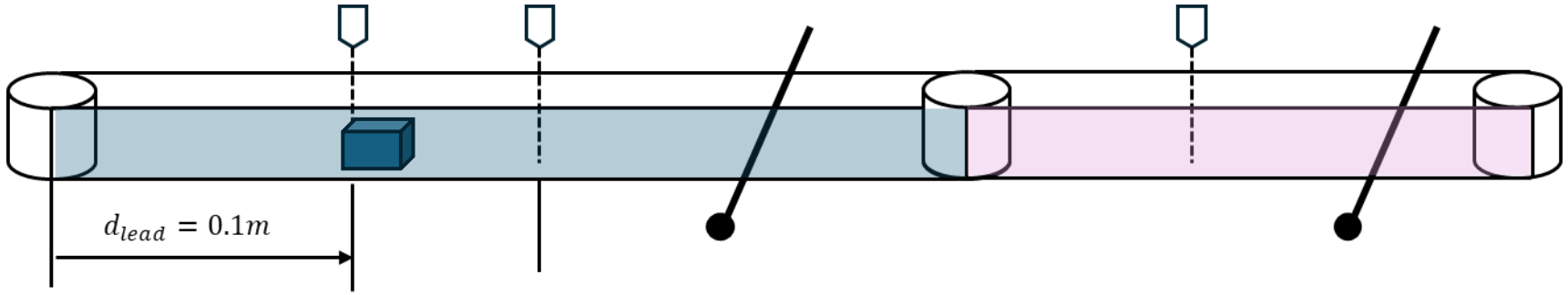
t_6 and its front edge reaches Gate 2 based on the parameters in the diagram?



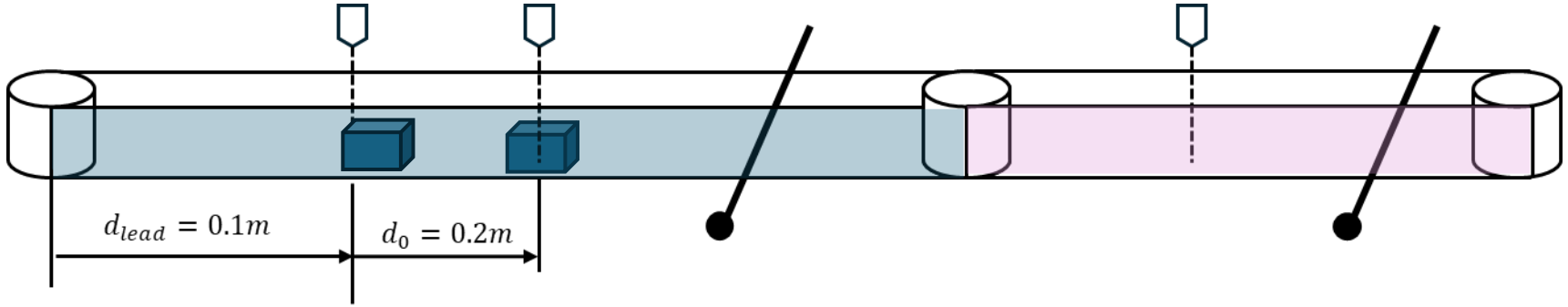
T0: g_obstructionSensor = 1



T1: g_obstructionSensor = 0

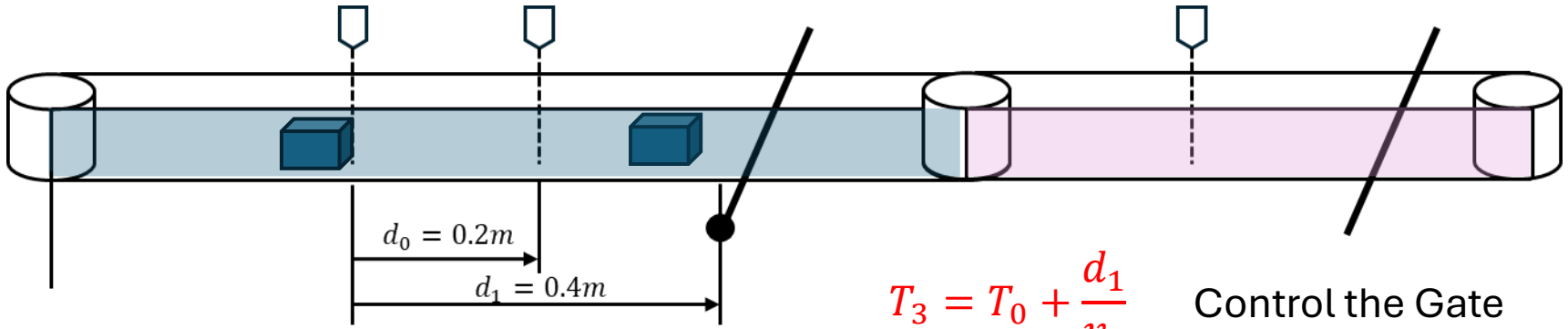


T1: g_obstructionSensor = 0



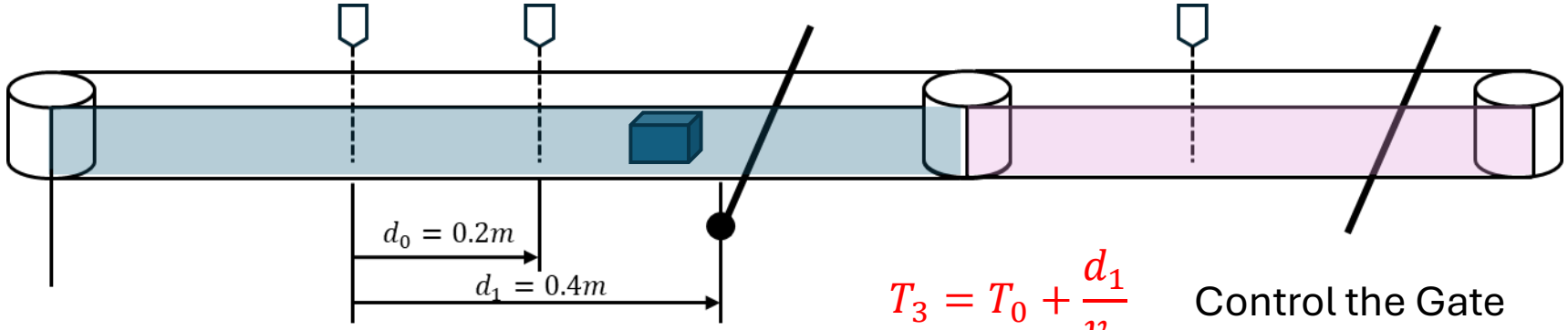
$$T_2 = T_1 + \frac{d_0 - (\frac{Length}{2})}{v_1}$$

Read the bar code



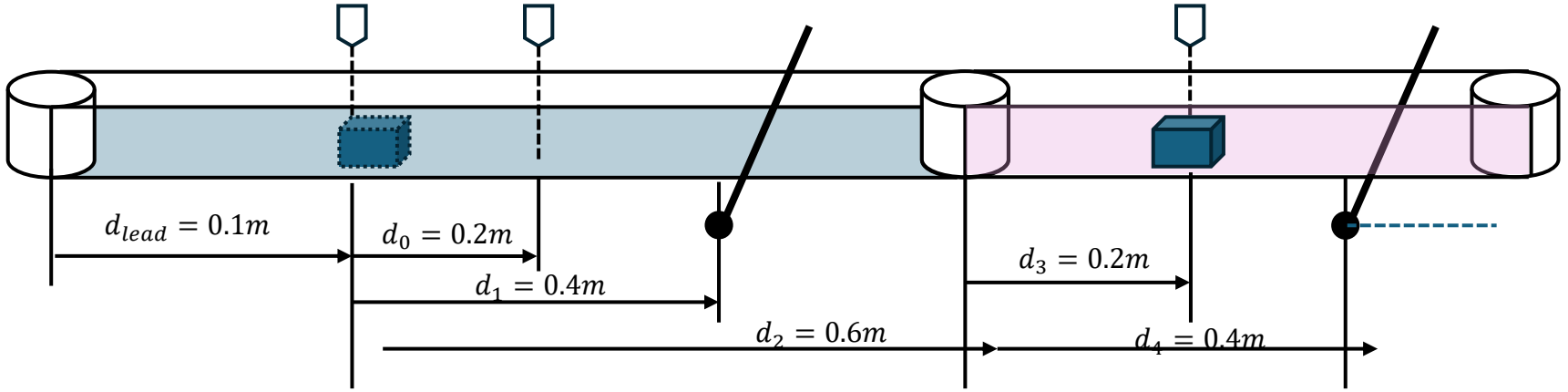
$$T_3 = T_0 + \frac{d_1}{v_1}$$

Control the Gate



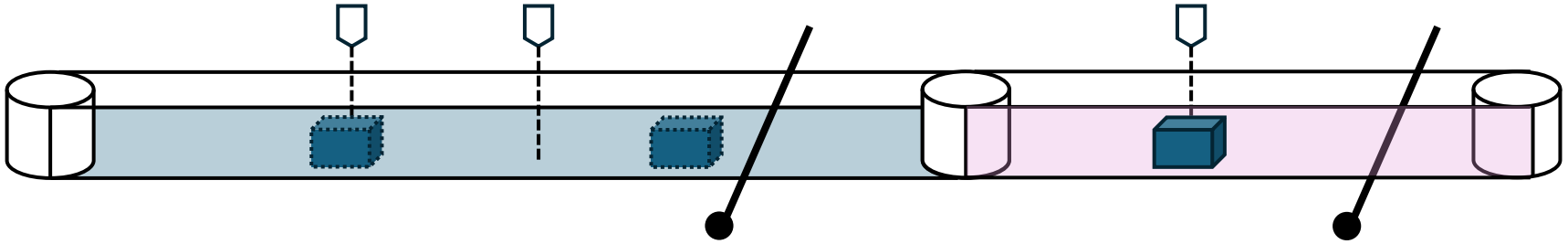
$$T_3 = T_0 + \frac{d_1}{v_1}$$

Control the Gate



$$T_4 = T_1 + \frac{d_2}{v_1} + \frac{d_{3-}(\frac{Length}{2})}{v_2}$$

Weight sensor



Your design should take into account the situation where there will be multiple items on the conveyor belt.

FAQ of CW

Report Structure

- Your CW report must clearly articulate your:
  - Design approach
  - Justification of design choices
  - Implementation details
  - Testing methodology
  - Final conclusions

FAQ of CW

Code Presentation

- It is **not recommended** to include screenshots of code in your report
- This is because you will be required to submit your complete code separately anyway
- Focus on explaining your approach rather than reproducing code

FAQ of CW

Code Template Selection

- Three template options have been provided for you:
  - A blank simulator template
  - A template with some example code
  - A template with more comprehensive example code
- Choose **one** template and stick with it throughout your implementation
- Switching between different templates is not advised as they may be incompatible
- The blank template is recommended as your first choice for implementing your own code

FAQ of CW

Code Originality

- If you extensively reuse the provided example code with minimal changes, your marks will be significantly limited
- The example code represents just one of many possible design approaches
- You are expected to demonstrate your own understanding and creativity

FAQ of CW

Preparation

- Before you begin coding, ensure you have thoroughly understood all the concepts discussed in class
- Review the code from laboratory practical sessions
- Establish a clear plan before implementation

FAQ of CW

Support

- I cannot debug code individually for every student
- However, I can provide general guidance on code optimisation strategies
- Make use of LLM like ChatGPT and self-debugging techniques

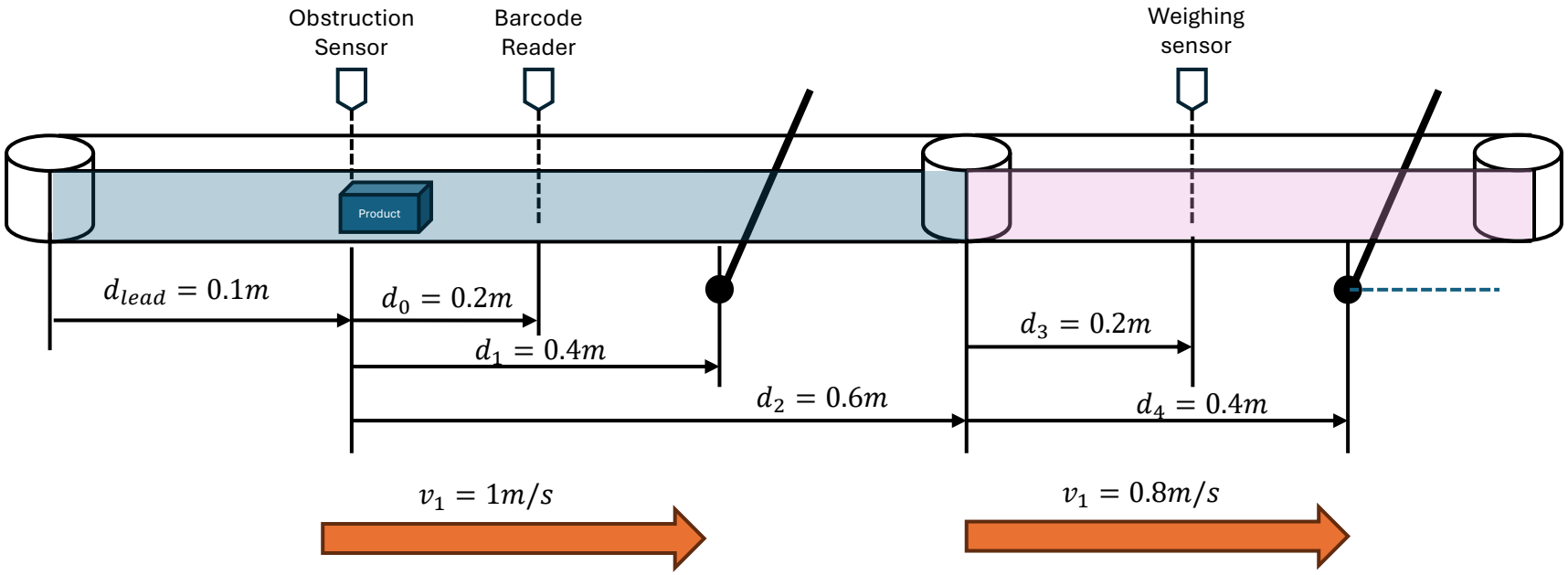
FAQ of CW

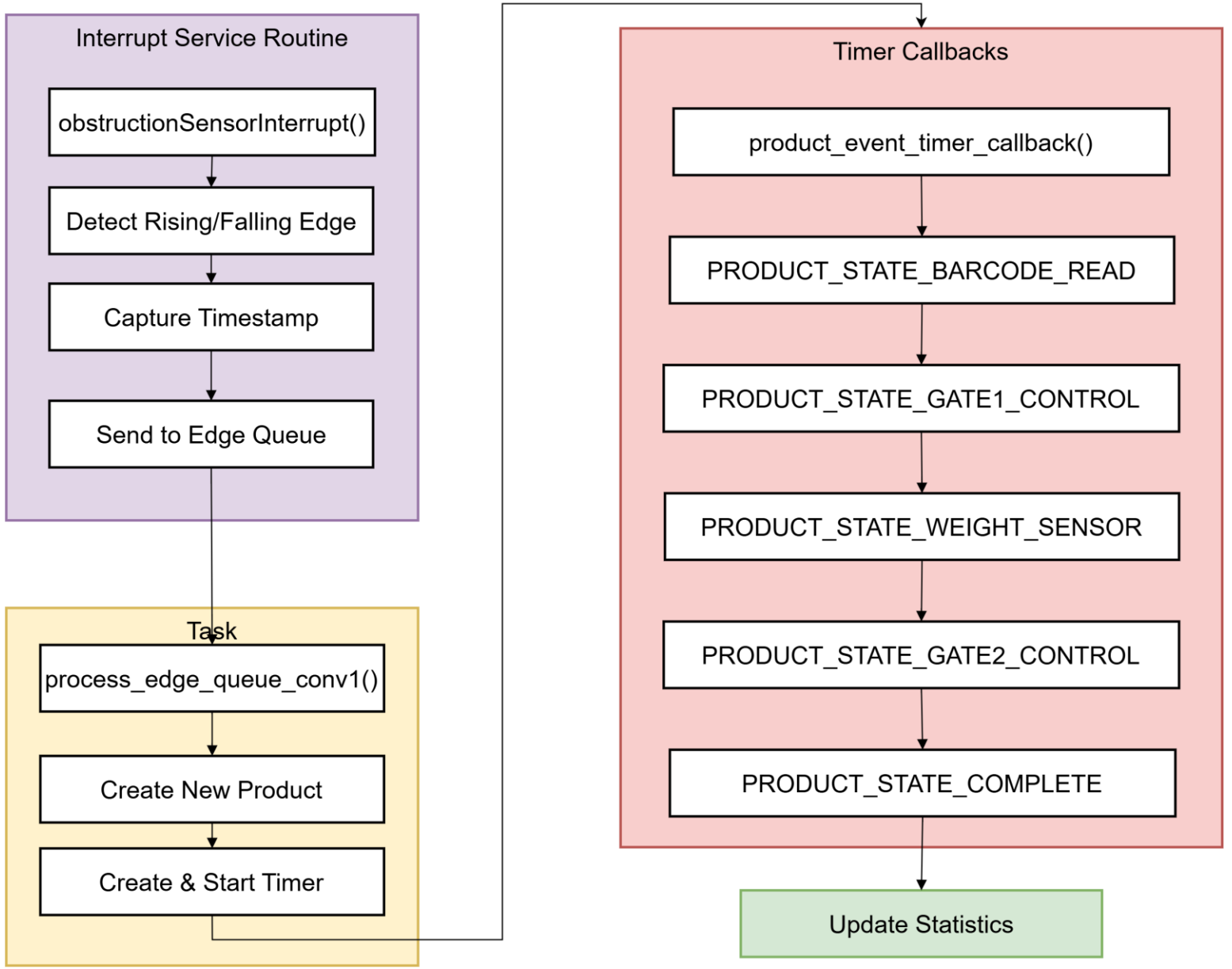
Remember

- This coursework is designed to assess your understanding of real-time embedded systems
- Your implementation should demonstrate application of principles discussed in lectures
- Original approaches that meet the requirements will be valued highly

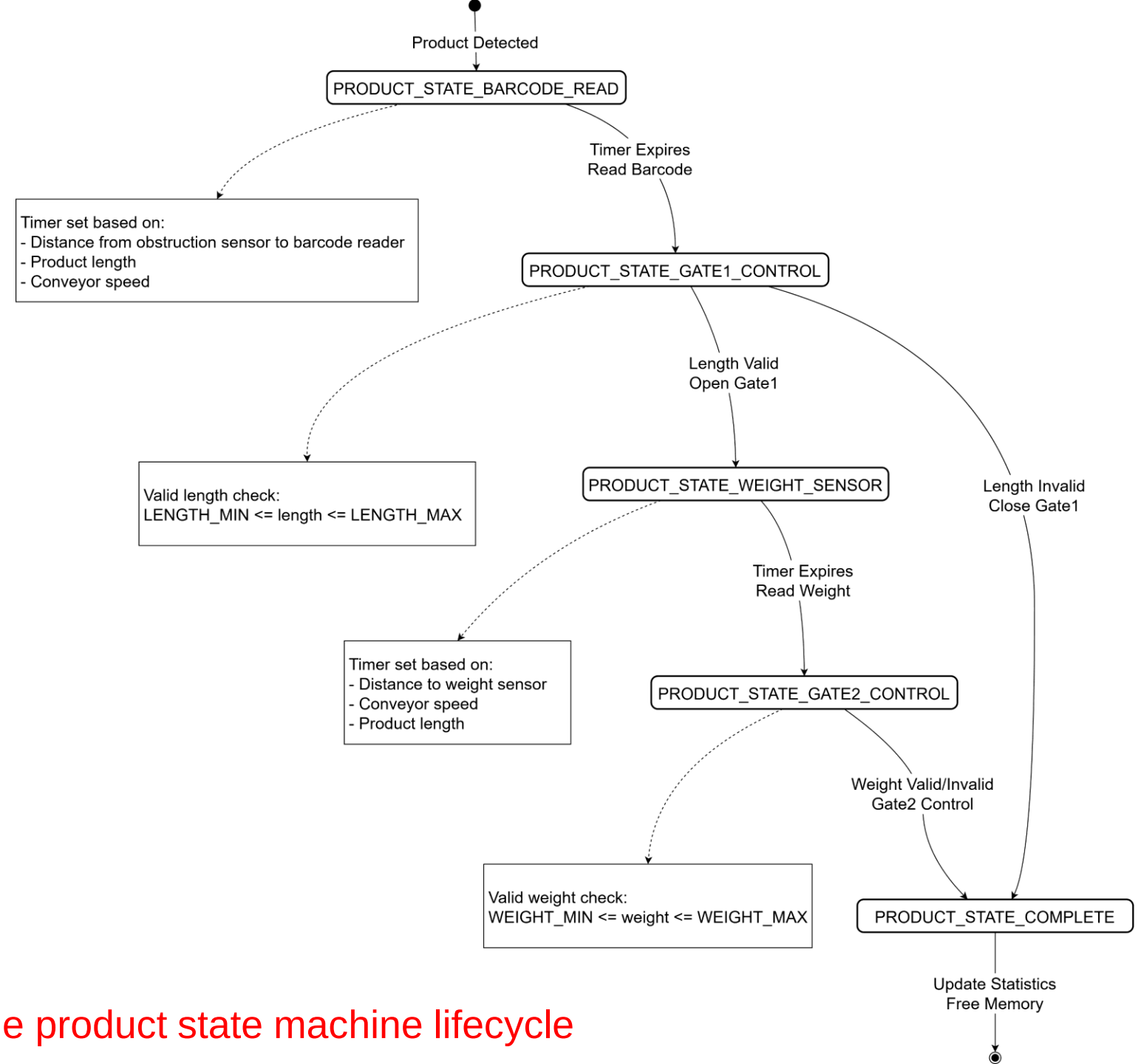
Design of the example Code

Design of the example Code

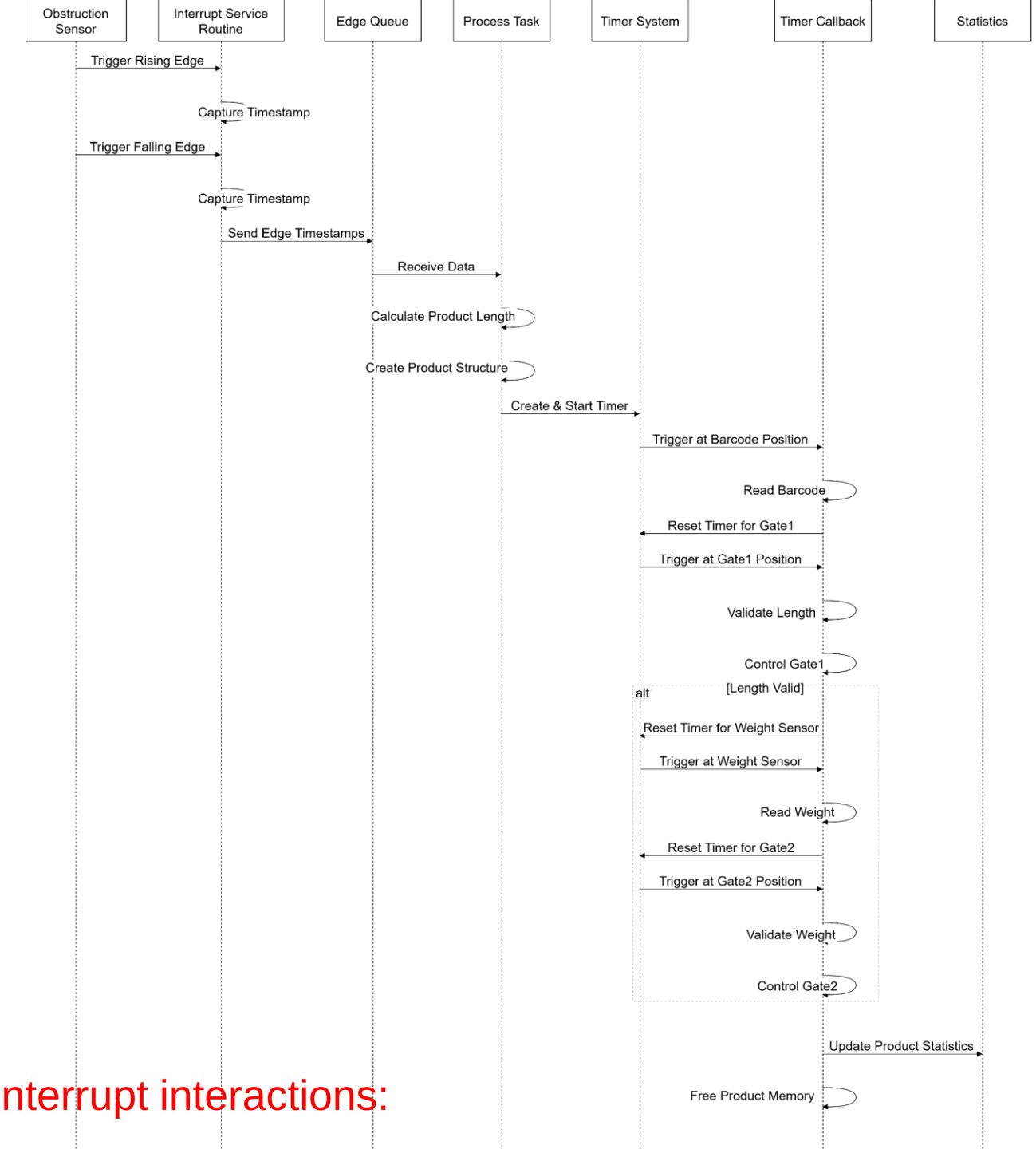




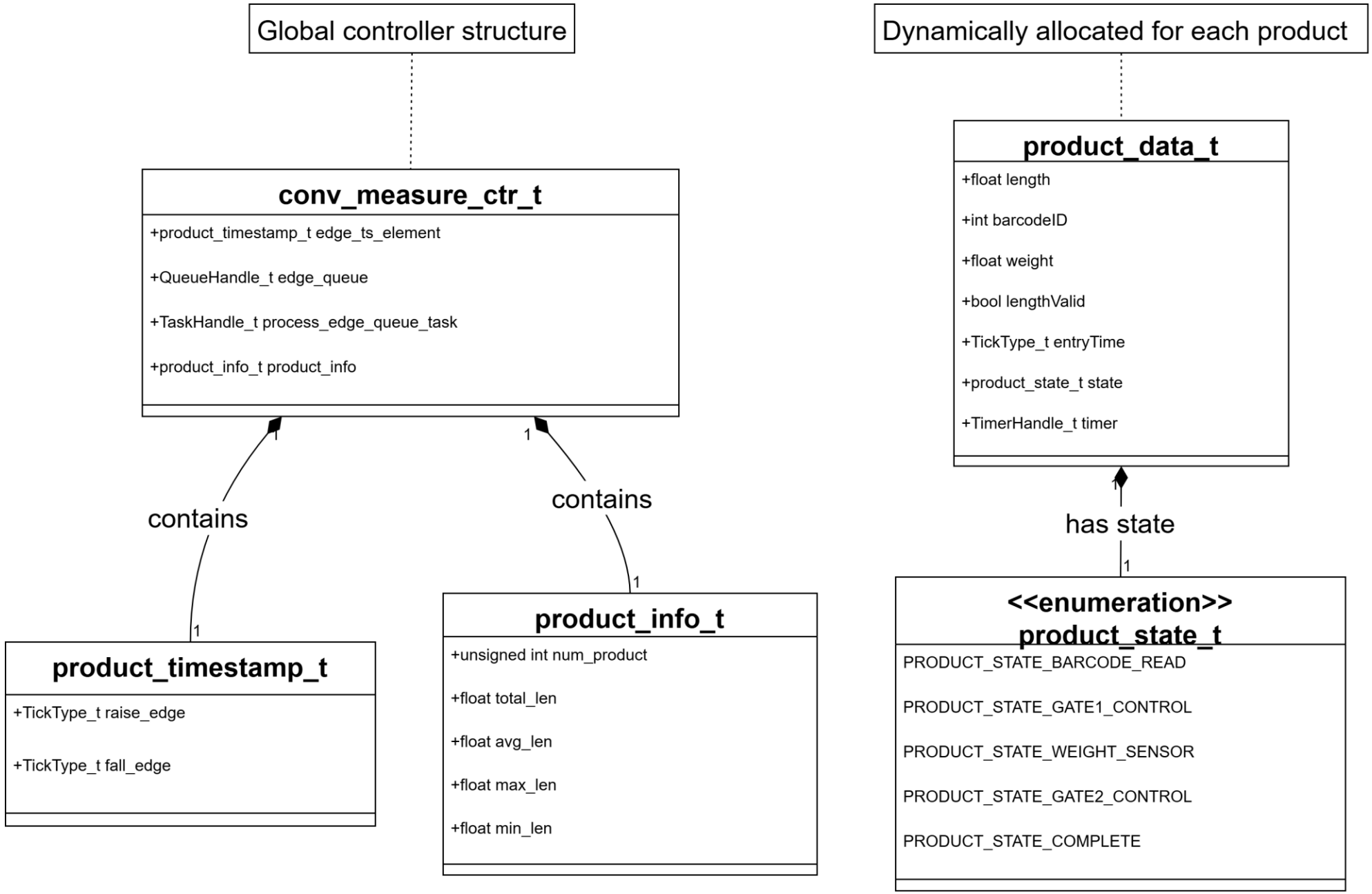
High-level diagram to show the main modules and their interdependencies.



Flow diagram for the product state machine lifecycle



Flow diagram for task and interrupt interactions:



Memory management and data structures