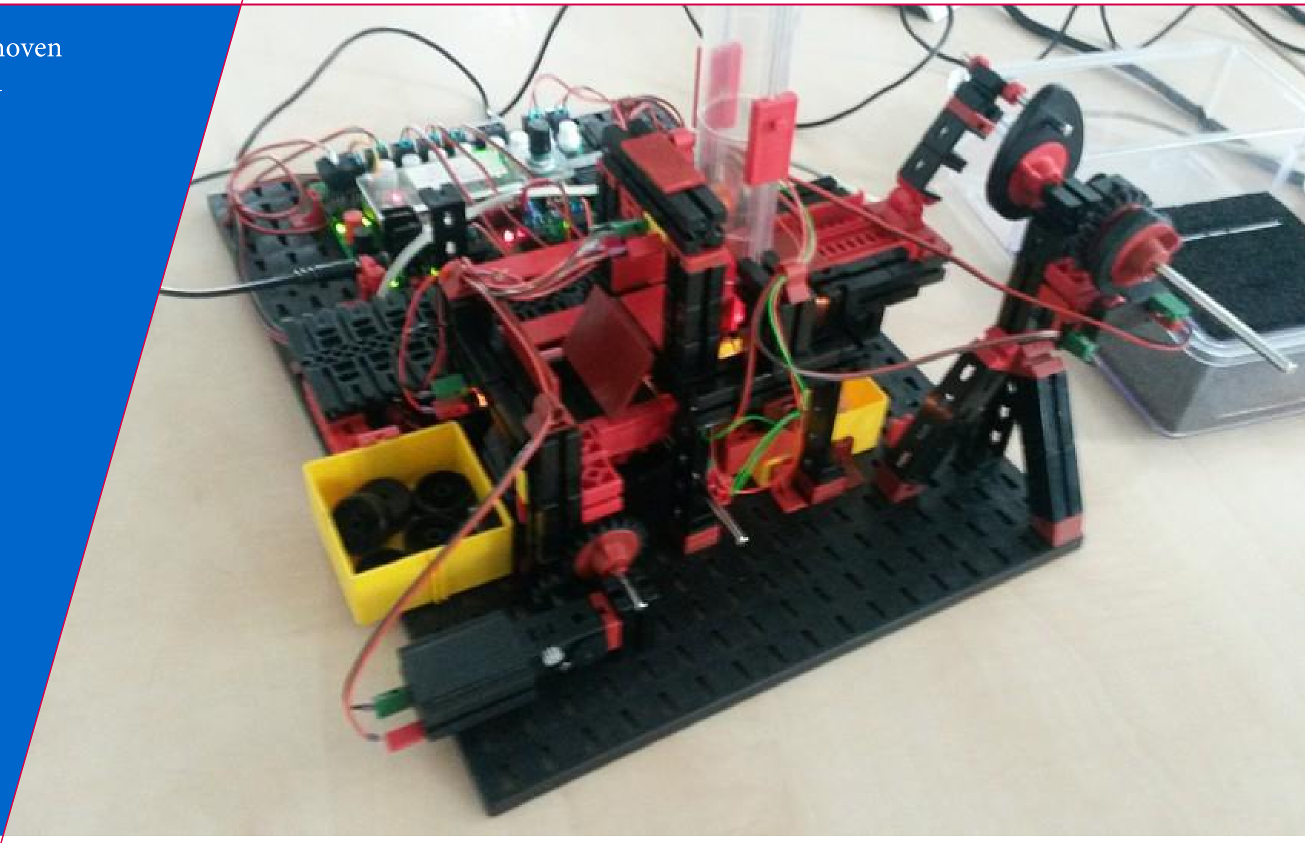
Technische Universiteit **Eindhoven** University of Technology

/ Department of Mathematics and Computer Science

D. van Broekhoven
D.C. Chirascu
Y. Li
A.A. Popov
J.G. van Vliet



OGO 2IO70 - DBL Embedded systems

Final Report, "Sort It Out"

Group 17

01-04-2015

Final Report, "Sort It Out"

Alexander Popov	0924595
Dan Cristian Chirascu	0923784
Jiddo van Vliet	0821894
Yuntao Li	0910663
Devin van Broekhoven	0839004

April 2, 2015

The Project Assignment

Introduction

The subject of the Design Based Learning project DBL 2IO70 is: specification, design, and construction of a machine that is controlled by a, so-called, microcontroller . A microcontroller is a very small computer that is equipped with inputs and outputs that can be used to control a machine. Since the microcontroller does not operate in a stand-alone fashion, like a personal computer does, but is a part of a larger whole the machine thus controlled , such a controller including its software is called an embedded system. In one very important respect, however, a microcontroller is a real computer: it operates under the control of a program. The design and construction of an embedded system, therefore, unavoidably involves programming.

In this project, methods and techniques are used to which the participants have been introduced during a preceding quarter. In particular, the project elaborates on the knowledge and experience with (the programming of) microcontrollers obtained in the Computer Systems course.

In this project the participants also have to solve problems of a mechanical or electrical nature, which cannot be considered as true software engineering problems, but which a software engineer may encounter in an industrial environment. In this respect this project clearly has a multi-disciplinary flavour.

A sorting machine

The goal of this project is to build a simple sorting machine that is able to separate small objects, plastic discs that may be either black or white, into two sets: the black discs and the white discs. For this purpose the machine is equipped with, among other things, one or two conveyor belts to transport the discs inside the machine. Initially, the discs to be sorted are contained in a storage tube; the machine will have a mechanism to deposit one disc at a time onto a conveyer belt.

The intended sorting process now proceeds as follows. From the storage tube a disc is deposited onto the first conveyor belt. This belt transports the disc to a socalled, black/white detector . Arrival of the disc at this detector is to be signaled; this can be done by means of a, so-called, light trap , which consists of a lamp and a photo cell: arrival of the disc is detected by interruption of the light beam from the lamp to the photo cell.

The black/white detector itself also consists of a lamp and a photo cell, but these are positioned in such a way that the photo cell measures the light that is reflected by the disc: because white discs reflect considerably more light than black ones, white and black discs can thus be distinguished.

Subsequently, the disc is transported further to one of two trays in which the black and white discs are collected: one tray for the black discs and one tray for the white ones. These trays may, for instance, be positioned at the two ends of the conveyor belt, or a second conveyor belt may be used for this purpose. So, depending on its design, the machine may have two conveyor belts, or just a single one. In either case, in which of the two trays a disc is deposited depends on the direction of movement of the conveyor belt, which is to be determined by the microcontroller; this direction, of course, will depend on the measurement provided by the black/white detector.

See Figure 1 for a possible general layout of the machine, in the case of a machine with two conveyor belts. (In this drawing the mechanism to deposit a disc from the storage tube onto the conveyor belt has not been drawn.)

Of course, other configurations are possible (and permitted) too; the black/white detector, for example, could also be placed above the second conveyor belt does that have advantages? . And, as already suggested above, a machine containing a single conveyor belt only, is conceivable too. The only real requirement, is that the machine must contain at least one conveyor belt.

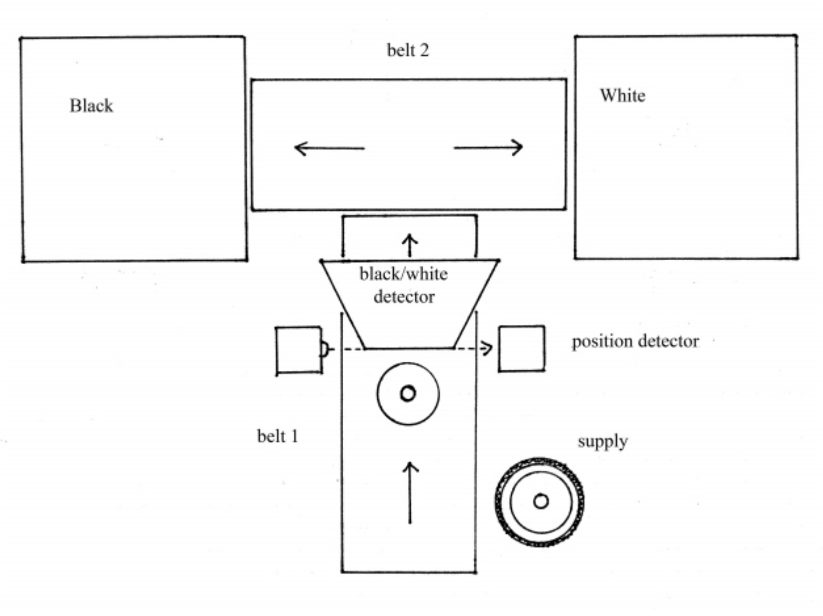


Figure 1: possible layout of the machine

Control buttons and details of the process

The machine is to be operated by means of two push buttons, called START/STOP and ABORT, in the following way. This process description is based on a machine configuration with two conveyor belts and with the black/white detector above the first belt, as in Figure 1. For other configurations this process description must be rewritten accordingly. The two push buttons START/STOP and ABORT may be simply realised by dedicating two of the (eight) push buttons on the PP2 board to this purpose.

1. After starting the program in the microcontroller the machine is in its resting state. In this state black and white discs can be placed, manually, into the storage tube. In the resting state all motors in the machine are stopped and all lamps are off.
2. By pressing button START/STOP the machine is started and proceeds to its regular operating state. First, if necessary, the machine brings all internal mechanisms into their, well-defined, initial state.
3. After this the actual sorting process starts. The machine deposits one disc onto the first conveyor belt, after which this belt transports the disc to the black/white detector. Then the belt stops and the color of the disc is measured by means of the detector.

4. If the storage tube was empty then obviously no disc was deposited onto the conveyor belt and the presence detector at the black/white detector will never signal arrival of a disc. Never, however, takes a very long time and, therefore, it is impossible to decide that something never happens. Based on knowledge of the speed of the conveyor belt it is possible to calculate when arrival of the disc at the black/white detector is to be expected. If 4 seconds after this expected arrival time the presence detector has not signaled the arrival of a disc, then we assume that such a disc will indeed never arrive because the storage tube was empty. If this happens the machine stops and returns to its resting state. In this way the sorting process will terminate automatically after all discs from the storage tube have been processed.

5. If the presence detector at the black/white detector does signal arrival of a disc, the second conveyor belt starts moving into the direction of the correct, depending on the observed color of the disc, collecting tray. Because the disc is still on the first conveyor belt this belt is started again too, in order to transport the disc to the second belt.

6. In order to detect that the disc really reaches a tray, additional light traps must be placed at both ends of the second conveyor belt. (In Figure 1 these additional detectors have not been drawn.)

7. If the disc has been deposited into the right tray the machine repeats the cycle that starts with placing a disc onto the first conveyor belt; that is, the machine returns to step 3 and continues operating in this way until the process terminates because the supply of discs is exhausted, as described in step 4.

8. If, during the sorting process, the push button START/STOP is pressed the machine completes the current cycle, that is, the machine continues its normal operation until the current disc has been deposited into the correct tray. Then, the machine stops and returns to its resting state. If such a stop command has been given, repeatedly pressing the button START/STOP again while the machine is still busy completing its current cycle will have no effect whatsoever. If, after a stop command, the machine has completed its cycle and has reached its resting state, then the machine is ready again to be started: in this state pressing push button START/STOP will start the machine again, just as in step 2.

9. Push button ABORT is used as an emergency button. Pressing this button while the machine is in its resting state has no effect. Pressing this button while the machine is operating makes the the machine halt immediately: Both the conveyor belts and the mechanism to deposit discs onto the belt and, possibly, all other mechanisms present in

the machine are switched of immediately. The purpose of button ABORT is to be able to halt the machine if the person operating it observes that something goes wrong or is about to go wrong. If the machine has been halted this way, manual interaction by the operator is needed, for instance, to remove any disc still on a conveyor belt, or to remove

discs that might have fallen off a conveyor belt a most undesirable situation, of course. If, subsequently, the push button START/STOP is pressed once, the machine returns to its resting state.

10. To be able to guarantee that the mechanism depositing discs onto the conveyor belt stops in a well-defined state, this mechanism must be equipped with (at least) one switch to signal that this mechanism has reached the correct state.

11. It is recommended to make the different operating states of the machine visible on the LEDs and/or the display of the PP2 board. In particular, it is a definite plus if the software controlling the process is designed so as to display the number of black discs and the number of white discs deposited in their corresponding trays.

A general design rule is that motors and lamps are switched off whenever possible, that is, whenever there is no need for them to be switch on.

The black/white detector

The construction of the black/white detector requires some care and some experimenting. After all, the light reflected from a white disc is weaker than direct light; therefore, detecting reflected light is more difficult. For this detector one uses one (or even two!) lens lamps positioned in such a way that it projects a nice patch of light onto the disc to be examined. Then a photo cell is placed in such a way that it observes this patch of light. This works best if lamp and photo cell are placed as close as possible to the disc. See Figure 2 for a possible configuration.

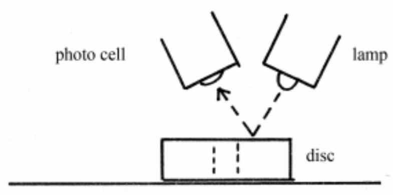


Figure 2: possible layout of the machine

The problem is to make sure that a white disc reflects enough light to be detectable; to make sure that a black disc reflects so little light that it is not seen as a white disc, is no problem at all. The correct placement of lamp(s) and photo cell is best determined experimentally. To this end one connects a lens lamp with a series resistor, as prescribed in the Technical Guide to the 12 V connector on the PP2 board. Also, one connects a photo cell also as prescribed in the Technical Guide to one of the digital inputs of the PP2 board. The PP2 board then is powered, but does not need a program: the green LED corresponding to the digital input of the photo cell will indicate the state of that input.

Now place a white disc under the detector under construction and position lamp and photo cell in such a way that the green LED at the input lights up. That means that the white disc is recognized. Make sure that this works correctly, even with minor variations in the positions of the disc, the lamp, and the photo cell. If you are satisfied with the result then secure the positions of the lamp and the photo cell. Finally, place a black disc under the detector and verify that it is not seen as white.

To prevent surrounding light disturbing the measurement, a black shielding hood can be placed over the photo cell.

It is also possible to use one of the two A/D inputs again: see the Technical Guide how to do this of the PP2 board. This makes construction of the black/white detector easier, and it is even possible to use a single lamp and photo cell combination both as a presence detector and as a black/white detector. The use of A/D converters in situations where they are not really needed, however, is considered a wasteful use of resources. Therefore, the use of an A/D input is permitted but it is a definite plus if the A/D inputs are not used and the black/white detector uses digital inputs only. (This is possible!)

Preface

Project "DBL 2IO70" concerns specification, design and construction of a sorting machine using a microcontroller, namely, Practicum Processor 2 (PP2). Use of the microcontroller as a part of a complex system gives the name to the course (Embedded Systems). The project involves dealing with not only software design problems, but also with problems from other fields, such as mechanical and electrical engineering, which makes it truly interdisciplinary. The project requires from students knowledge of the contents of courses Logic and Set Theory, Programming and Computer Systems.

Summary

During the project our group has worked on all the different aspects on designing, building and programming the sorting machine. These different aspects are:

- Machine design, making an overall design of the machine and explaining under which circumstances the machine can be operated. In this part also the safety properties and user constraints are explained
- Software specification is the first step to actually integrating the PP2 processor in to the mechanical part of the machine. We explained which inputs and outputs we will use and how they are related to each other.
- Software design required us to write java code which did not have to be able to run. Instead this java code was used to give us an example on how to write the Assembly language for the PP2 processor. It also helped us divide the work for programming in Assembly.
- Software implementation and integration is the phase where we actually had to write Assembly Language for our PP2 processor. So at the end of this phase our sorting machine had to be fully functional.
- Software validation and testing required us to use different means to actually test our software. Different documents had to be made to explain our code and show it could handle all the different test cases we had prepared.

These parts themselves were divided into different sections which helped maintain a good structure in building the machine. More detailed information about each section of the project can be found in this Final report.

Table of contents

1 Method of working

The project for designing, building and documenting about the sorting machine is done in different phases. Each phase focuses on a different aspect of building or designing the machine.The phases of the project are:

- Machine design
- Software specification
- Software design
- Software implementation
- System validation and testing

In the machine design phase the actual machine has to be build using the Fisher technik parts. Also the Use Cases, User constraints and Safety properties have to be described. This was done entirely during the group meetings. Therefor the workload was equally distributed in an informal way.

The Software specification part was done with an informal distribution of the workload meaning we completed the entire document during our group meetings.

Software design was divided in 2 groups where 1 group was responsible for the java code and the other was responsible to write all the necessary documentation about the java code.

The software implementation was about writing the Assembly language program to control our machine. The Assembly language program was divided into multiple sections which represented the states the machine. Each member of the group had to write a certain part of the program. After all parts were done these were combined and checked for errors.

System validation and testing is to demonstrate that the final product meets its initial requirements, that is, that the executable object code correctly implements the System Level Requirements. Also, it is important to show that the implementation does not either do more than expected. In this phase we had both of a code review and formal proofs. For code review, we elected a presenter to show the code and the other members did the check and gave feedback. For formal proofs we relied on our UPPAAL model to check if the safety properties were correctly implemented.

2 Machine Design

2.1 Motivating our design choices

For the system to be easy to use and for us to be able to easily fix issues we decided to design the machine using different heights for the components. The first component of the machine is the loading tube and the mechanism to push 1 disc at a time onto the conveyer belt. This component is located on top of the machine so the user can easily load the discs into the loading tube.

To keep the machine as fast as possible we decided to identify the color of the disc before it is loaded onto the conveyor belt. This way we can ensure the disc keeps moving when it has left the loading tube and thus no time is wasted during the sorting process. Since identifying the color of a disc using the sensors included in the kit is nearly instant we were able to integrate the black/white detector into the component which pushes the disc on the conveyer belt.

After a disc has passed the black/white detector it will be pushed on the conveyer belt. Because the conveyer belt itself is located on a lower level and in the same direction of the black/white detector a disc will simply drop down onto the belt. This means the component which pushes a disc from the loading tube to the conveyer belt can run on a higher speed without the risk of the disc falling sideways.

The conveyer belt itself is powered by a single engine which is able to turn both ways using the PP2 processor to change the polarity of the voltage. Using the input from the black/white detector the conveyer belt will either turn left or right to deliver the disc to its appropriate storage box. We decided to implement presence detectors on both ends of the conveyer belt which will detect if a disc has actually been delivered to its storage box. If the wrong detector or none of the detectors detect a disc it can send feedback to the machine.

2.2 System Level Requirements

2.2.1 Use Cases

Pre condition

The PP2 processor is connected to all the engines, lights and sensors used in the machine it is also connected to a computer with the required code to run it. There are no discs in the loading tube or on the conveyer belt.

Trigger

The user starts the sorting machine by pressing the start/stop button and loads all the discs into the loading tube after which the start/stop button is pressed again to start the sorting process.

Guarantee

The machine will start sorting the black and white discs which were inserted into the loading tube.

Sorting under normal conditions

1. Pressing the start/stop button on the machine will put all mechanical parts in their appropriate resting states. The conveyer belts is stopped and all lights and detectors are turned on to test if they are functional. The part which pushes the single discs from the loading tube onto the conveyer belt will be turned to its most retracted state.
2. All white and black discs are inserted into the loading tube on the top of the machine. When all discs are inserted the user is allowed to press the start/stop button again.
3. Pressing the start/stop button will start the sorting progress.
4. First the machine will detect if there are discs left to sort by checking if a disc left the loading tube using a light detector.
5. When a disc has been detected it will be pushed to the black/white detector which decides which way the conveyer belt will turn.
6. After the color of the disc has been determined the disc will be pushed onto the conveyer belt.
7. The conveyer belt will transport the disc to one of the two storage boxes depending on the color of the disc.

8. At the end of each side of the conveyer belt a light detector determines if the disc has been delivered to the appropriate storage box in the given time.

Sorting under deviated circumstances

A disc has fell off/is suck on the conveyor belt

8-a Disc has not been detected by the light sensor in the appropriate time. Abort all operations and go to halting state. User intervention is needed to recover the lost disc.

8-b Continue at the first state.

The disc dispenser does not work properly

5-a User detects malfunctioning of the machine which pushes the discs to the conveyer belt. The user has to press the abort button.

5-b The machine halts immediately, the user has to remove all discs from the system and fix the malfunctioning part.

5-c Continue at the first state.

Alternative

8-a Disc has not been detected by the light sensor in the appropriate time. Abort all operations and go to halting state. User intervention is needed to remove all discs from the system and fix the malfunctioning part of the machine.

8-b Continue at the first state.

The conveyer belt engine is malfunctioning

7-a User detects malfunctioning of the engine which powers the conveyer belt. The user has to press the abort button.

7-b The machine halts immediately, the user has to remove all discs from the system and fix the malfunctioning engine.

7-c Continue at the first state.

Alternative

8-a Disc has not been detected by the light sensor in the appropriate time. Abort all operations and go to halting state. User intervention is needed to remove all discs from the system and fix the malfunctioning engine.

8-b Continue at the first state.

2.2.2 Safety Properties

Inputs(in order of use in a cycle)

- Dispenser button
- Presence detector
- Black/white detector
- presence detector white
- presence detector black

Outputs(in order of use in a cycle)

- Power engine dispenser
- Power light presence detector
- Power light black/white detector
- Power engine conveyor belt
- Power light presence detector white
- Power light presence detector black

Input and output relations

- None of the detectors can give detect a disc at the same time.
- None of the engines can run on the same time.
- The engine for the belt can never run in both directions at the same time.
- When the start/stop button is pressed during a cycle the machine will only stop if the cycle has finished.
- When the abort button is pressed all operations have to stop and the machine will go to its halting state. Even when the machine is mid cycle, so for example a disc is on the conveyer belt, the machine has to stop. When the user wants to start the machine again user intervention is necessary to remove any discs from the machine.
- When the Presence detector has given a low signal either the Black or White presence detector should give a low signal after the dispenser has finished its rotation.

2.2.3 User Constraints

1. The user is only allowed to load the discs in the loading tube when the machine is in its resting state(the start/stop button has been pressed once).
2. When the machine is running the user is not allowed to touch/move the discs.
3. When the user has pressed the start/stop button while the machine is running it will finish its current cycle and then go to its resting state. Pressing the start/stop button multiple times while its running will have no different effect.
4. When the user has pressed the abort button the machine will stop immediately. All discs which have left the loading tube have to be removed and reinserted into the loading tube(see 1. for loading constraints).
5. While the machine is running the user is not allowed to touch any of the components of the machine and should make sure none of the lights and sensors are blocked.

2.3 Machine Interface

The sorting machine is connected to the PP2 board, which plays a role of a microcontroller in this assignment, through a so-called machine interface. In this document every connection between the actual machine and the microcontroller is described briefly.

Motors

Two electrical motors are used in the machine, let them be denoted as motor A and motor B for convenience. Motor A is responsible for the work of the disc supply mechanism, whereas motor B brings the conveyor belt in motion in one of the directions. Motor A is connected by a wire of type A to digital output 0 of the PP2, and motor B is connected by a similar wire to the H-bridge between outputs 6 and 7. That enables us to turn motor A in both directions so that we can bring the conveyor in motion in the correct direction and the disc can be deposited in the corresponding tray. However, it does not matter in which direction motor A turns, so it is only connected to 1 output. Polarity in case of motor A does not matter as well.

Lamps

Every lamp is connected to a digital output on the PP2 using a wire of type B to limit the voltage. They need to be turned on for the detectors to work (high output on corresponding digital output of the PP2). Polarity does not matter for the lamps.

Phototransistors

Photo transistors are used as detectors of (reflected) light from the lamps. In order for them to produce results, both lamps and photo transistors should be connected to the PP2 and the lamps should be on. They connected via wires of type A to digital inputs of the PP2. However, the photo transistor which is part of the black-white detector is connected to the analogue input of the PP2 by means of wire of type C. That allows for more precise detection of black and white discs (8 bits of ADCONVS register on the PP2 get set). When the light is perceived, the value is high. When something blocks the light, the value is low. In particular, there are two detectors which determine if a disc has reached the tray (input gets low then high again), one detector checking if there are any discs left to sort (input is high if there are no discs left) and the black-white detector (input is higher for white discs).

Control buttons

There are two control buttons Start/Stop and Abort connected to the digital inputs of the PP2 via wires of type A. When the button is pressed, the corresponding input value is set to 1 (the value is high).

3 Software Specification

3.1 Which signals from the machine interface are the inputs to the program, and what do these signals represent

Inputs to the program(in order of use in a cycle)

Dispenser button

1. Positioned next to the dispenser engine
2. Used to detect the dispensers current position
3. Gives a high signal if button is pressed, the dispenser will press the button on its most retracted state.

Presence detector

1. Positioned below the loading tube
2. Used to detect if there are discs left to sort
3. Gives a high signal if no discs are left, gives a low signal if there are discs detected.

Black/white detector

1. Positioned above the discs, right after Presence detector and right before the disc fall on the conveyer belt.
2. Used to detect if a disc is black or white.
3. Gives a high signal on white discs, gives a low signal on black discs.

Presence detector white

1. Positioned at the end of the conveyer belt where the white discs leave the belt and go into a storage box.
2. Used to detect if the white disc has been delivered by the conveyer belt.
3. Gives a high signal on no disc, gives a low signal if a disc has been detected.

Presence detector black

1. Positioned at the end of the conveyer belt where the black discs leave the belt and go into a storage box.

2. Used to detect if the black disc has been delivered by the conveyer belt.
3. Gives a high signal on no disc, gives a low signal if a disc has been detected.

3.2 Which signals from the machine interface are the outputs from the program, and what do these signals control?

Outputs to the program(in order of use in a cycle)

Power engine dispenser

1. Positioned next to conveyer belt.
2. Used to distribute 1 disc at a time to the black/white detector, and then to the conveyer belt.

Power light presence detector

1. Positioned below the loading tube.
2. Powers the light for the presence detector

Power light black/white detector

1. Positioned above the discs, right after Presence detector and right before the disc fall on the conveyer belt.
2. Powers the light for the black/white detector.

Power engine conveyor belt

1. Positioned just next to the conveyor belt.
2. Used to power the conveyor belt in either right or left direction.
3. Polarity of this output can be inverted to invert the motion of the engine and thus change conveyor belt direction.

Power light presence detector white

1. Positioned at the end of the conveyer belt where the white discs enter the storage box.
2. Powers the light for the presence detector.

Power light presence detector black

1. Positioned at the end of the conveyer belt where the black discs enter the storage box.
2. Powers the light for the presence detector.

Input and Output relations

Power engine dispenser

The engine for the disc dispenser will always go to its most retracted state on its first use. This is checked by using the button attached to the dispenser. When the button is pressed the dispenser will start turning untill it is pressed again. If the button is not pressed it will also turn until it is pressed. After the first rotation it will depend on the input from the *Presence Detector*. When the *Presence Detector* detects a disc (gives a low signal) the engine will make another rotation in the next cycle. If it does not detect a disc (gives a high signal) the engine will finish its rotation but will stop in the next cycle.

Power engine conveyor belt

The engine for the conveyer belt will only turn if the *Presence Detector* has detected a disc and the *Black/White Detector* has detected its color. For white discs the *Black-/White Detector* will give a high signal and for black discs it will give a low signal. Depending on the signal for the *Black/White Detector* the engine for the conveyer belt will either turn left or right to deliver the discs to their appropriate trays.

Presence detector white/black

The Presence Detector White or Black detects if a disc has been delivered to a tray. If a disc passes the detector it will give a low signal, when no disc passes the detector it will give a continuous high signal. When the detectors have not given a low signal in a certain time interval all other components of the machine will stop immediately.

3.3 Which abstract states are needed to adequately model the required behaviour of the machine?

For our model we will use a starting state which is also the halting state, resting state and multiple running states. More information about which order these states appear and how many running states are used can be found in the next section.

3.3.1 How does the state of the machine change in reaction to (changes in) the inputs? How do the outputs depend on the inputs and on the current abstract state, that is, what is the required reaction of the PP2 to (changes in) the inputs?

1. Our first state is besides the starting state also the halting state. In this state none of the outputs and inputs will give any signal.
   - When powering up the machine it will be in this state.
   - When pressing the abort button from any state it will immediately go to this state.
   - Going to the next state is only possible by pressing the start/stop button.
2. The second state is the resting state. In this state the machine will put the dispenser into the correct position so an output is given to the dispenser engine. None of the other inputs and outputs will give any signal.
   - When the start/stop button has been pressed in any of the other states except the first state it will return to this state at the end of the cycle.
   - Going to the next state is only possible by pressing the start/stop button.
3. The third state is the running state. In this first running state the machine detects if there is any disc left to sort. The only input sent to the PP2 controller is that of the presence detector. The only output is the power of the light for the presence detector.
   - When a disc has been detected it will go to the next state.
   - When no disc has been detected it will go to the 2^{nd} state.
4. The fourth state is the second running state. In this state an output will be send to the engine of the dispenser.
   - After a time based event the machine will go to the next state
5. This is the third running state. An input from the black/white detector will be sent to the PP2. This input will be used in the 7^{th} state.
   - After a time based event the machine will go to the next state
6. The fourth running state. An output is sent to the dispenser engine.

  - After a time based event the machine will go to the next state

7. The fifth running state. The input from state 4 is used to determine the next state.
   - When input from state 5 is high go to state 8 A.
   - When input from state 5 is high go to state 8 B.

8. (A) The sixth running state. An output to the conveyer belt engine is send to turn it right until either a signal has been detected at the presence detector white or time has exceeded the limit.
   - When input from presence detector is given go to state 3.
   - When time limit is exceeded go to state 1.

   (B) The sixth running state. An output to the conveyer belt engine is send to turn it left until either a signal has been detected at the presence detector black or time has exceeded the limit.
   - When input from presence detector is given go to state 3.
   - When time limit is exceeded go to state 1.

4 Software Design

4.1 The classes in our Java document

- Button.java
- BWDetector.java
- ConveyorBelt.java
- Detector.java
- Dispenser.java
- Controller.java

Button.java

Is used to create a Boolean variable button used as the start/stop button in the machine. This button is used in Controller.java.

BWDetector.java

Is used to create a Boolean variable LightOn which will be used to represent the state of the light in the Black/White detector. Also an Integer variable Value is created to represent the signal from the Black/White detector to the PP2 processor. Both variables are used in Controller.java.

ConveyerBelt.java

Is used to create a Boolean variable Left and Right. These variables are used to turn the conveyer belt either left or right in the Controller.java, both variables can't be true at the same time since the conveyer belt can only turn one direction.

Detector.java

Is used to create a Boolean (?) variable PresentDisc and a Boolean variable LightOn. The variable LightOn is used to control the state of the light in the presence detector. The variable PresentDisc is used as input to the PP2 processor to determine if a disc is present. Both variables will be used in Controller.java.

Dispenser.java

Is used to create the Boolean variables On and Pressed. The variable On will be used to determine the state of the dispenser in Controller.java. The variable Pressed is used to find the position of the dispenser in Controller.java.

Controller.java

Is the main class of the java program. In this class all variables from the other classes will be used to control our sorting machine. Controller.java also uses the states which are in our UPPAAL model. This will make sure our machine will use all the inputs/outputs(variables) in the correct order. The states used in Controller.java are

INIT

This is the initial state. In this state we will check if the start/stop button is pressed. When the button is pressed we will move the dispenser in its correct position using the button integrated in the dispenser. If dispenser button is not pressed turn on dispenser and go to INIT1. If dispenser button is pressed turn on dispenser and go to INIT2.

INIT1

This is the intermediate state where our dispenser is not in the correct position, the dispenser button is not pressed. We turn on the dispenser engine until the button gets pressed once and then is depressed.

INIT2

This is the intermediate state where our dispenser button is pressed. In this case we move the dispenser until the button is depressed.

READY

This is the state where our dispenser is in the correct position and the machine is ready to start sorting the disks. The sorting process begins when the user presses the start button again.

SP

Here the disk is pushed under the BW detector.

SD

Here the color of the disk is determined. The belt is set to rotate in the correct direction and the disk is pushed onto the belt.

SW1

The white disk is transported towards its tray and the dispenser is pulled back until it is in the correct position to push the next disk. The belt is still spinning in the direction that corresponds to the disk just dispensed. In this state we also check if the STOP button has been pressed.

SW2

The dispenser pushes the next disk under the BW detector. The disk on the belt may not have fallen yet into the tray and so the side detectors may still be awaiting disk detection.

SW3

We ask if the disk has reached its tray. If it hasnt then we abort, or else we go to state SD.

SB1

The black disk is transported towards its tray and the dispenser is pulled back until it is in the correct position to push the next disk. The belt is still spinning in the direction that corresponds to the disk just dispensed. In this state we also check if the STOP button has been pressed.

SB2

The dispenser pushes the next disk under the BW detector. The disk on the belt may not have fallen yet into the tray and so the side detectors may still be awaiting disk detection.

SB3

We ask if the disk has reached its tray. If it hasnt then we abort, or else we go to state SD.

The differences between the SW and SB states are the direction in which the belt is turning and the side detector awaiting disk detection.

SWS

This is the state in which we go if the STOP button has been pressed in the SW1 state. The dispenser is set into the correct position. If we detect the disk on the belt has reached its tray, then we go to the READY state; if it hasnt we go to the INIT state (This is an abort situation).

SBS

This is the state in which we go if the STOP button has been pressed in the SB1 state. The dispenser is set into the correct position. If we detect the disk on the belt has reached its tray, then we go to the READY state; if it hasnt we go to the INIT state (This is an abort situation).

5 System Validation and Testing

5.1 code Review

This report describes the process of writing and reviewing code by pair programming. The programmers are A.Popov and D.C.Chirascu

D.C.Chirascu wrote the draft and A.Popov optimized it in order to describe the various machine states properly and accurately. No bugs were found, and the Java code is mainly aimed to simulate how the machines works in an object orientated way. So there were just some parts of code that had to be fixed. This is the illustration:

1. D.C.Chirascu wrote the draft.
2. A.Popov inspected it and modified it.
3. D.C.Chirascu and A.Popov revised the code.

Since our java code is aimed to be the guidance for the assembly code, Alex and Dan finalized it when the final draft of the Assembly code was done. They did not modify it any more because, even though the assembly code would be modified as a result of testing, the Java code had fulfilled its purpose of describing how the machine states work. These states do not change even though parts of the assembly code were modified so that the machine would work properly.

There are some important improvements in the modification phase:

1. We merged the Start/StopButton.class and AbortButton.class because they share the same functionalities. We also merged the PreDetector.class and SideDetector.class due to their same functionalities.
2. We used enumeration to create State type variables that can have as values all our machine states (Init, SP, SD etc.):

   ```
   private enum State {
       INIT, INIT1, INIT2, READY, SP, SD, SW1, SW2, SW3, SB1,
       SB2, SB3, SWS, SBS
   }
   ```

   All machine state properties are explained in the software design document.

5.2 Test Cases

This report is aimed to deliver the result of testing cases, which describes the input sequence together with the expected outputs via testing. The testing is based on simulating the system with specific inputs and comparing the results produced by this system. The results should be equal to the expectation from the specification. In this project, a system under test will typically be a UPPAAL model. So these criteria were being tested:

5.2.1 Statement Coverage

For this criteria we need to show that every transition in the UPPAAL model has been executed. We just set the number of discs to 500 and we just randomized the transitions. And from the simulation trace we confirmed that every transition is enabled in the testing. (See Figure 3 in Appendix A: Figures)

5.2.2 Condition Coverage

In this stage we need to show the truth value of every Boolean expression. The corresponding condition could always be fulfilled. In the previous section Statement Coverage we have already checked the transition of main control. Here we just focus on the other parts of the model which have the Boolean expressions, which are Belt Engine Control, Black/White Detector and Belt Controller. During the inspection no errors or exceptions were discovered. (See figures 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21 in Appendix A: Figures)

5.2.3 Decision Coverage

Based on the result of Condition coverage, we use the simulator of UPPAAL to ensure that guards take all their possible outcomes.

BeltEngineControl (see figures 4, 5, 6, 7)

Initial The whole machine is in the resting state.

Condition 1 If the update is `mov_left=true` then the next state should be `Move_Left`

Condition 2 If the update is `mov_right=true` then the next state should be `Move_Right`

And when it reaches the state `Move_Left` (or `Move_Right`), as the boolean `mov_left=false` (or `mov_right=false`), or both Booleans turn to false, then it goes back to `Belt_Off` state.

Black/White Detector (see figures 8, 9, 10)

Initial The whole machine is in the resting state, there is nothing to detect.

Condition 1 If there is a white disc below the detector, we could get the update `curColor = false`. The next state will be White.

Condition 2 If we get the update `curColor = true`. The next state will be Black.

BeltController(see figures 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21)

Initial The whole machine is in the resting state.

Condition 1 If the belt moves to the left, then the Boolean `mov_left` should be true and `mov_right` should be false. If at the time when the `mov_left=true` and `mov_right=true` at same time, we need to rewrite the value of `mov_right` and then keep the value `true` for `mov_left`.

Condition 2 If the belt moves to the right, then the Boolean `mov_right` should be true and `mov_left` should be false. If at the time when the `mov_right=true` and `mov_left=true` at same time, we need to rewrite the value of `mov_left` and then keep the value `true` for `mov_right`.

MC/DC

In this part we mainly focus on the modification of Decision coverage. What we need to do in this stage is reverse the value we have used in Decision coverage. After that we compare the outcomes with what weve gotten from Decision coverage. Since we have the states for whether a Boolean is true and false respectively, the outcome is the same as in Decision coverage.

5.3 Formal Proofs

In this section we need to ensure two properties

1. There is no deadlock in the graph. There should not be any deadlocks in our graph, the transition should reach all the states and go to next state. (See figure 22 in Appendix A: Figures).
2. For PreDetector, If the checking procedure is fulfilled then the transition will reach the Ready state. Since all the Boolean functions are verified in previous test. Here we need to ensure the inputs are in position for the automaton. (See figure 23 in Appendix A: Figures).

6 Conclusion

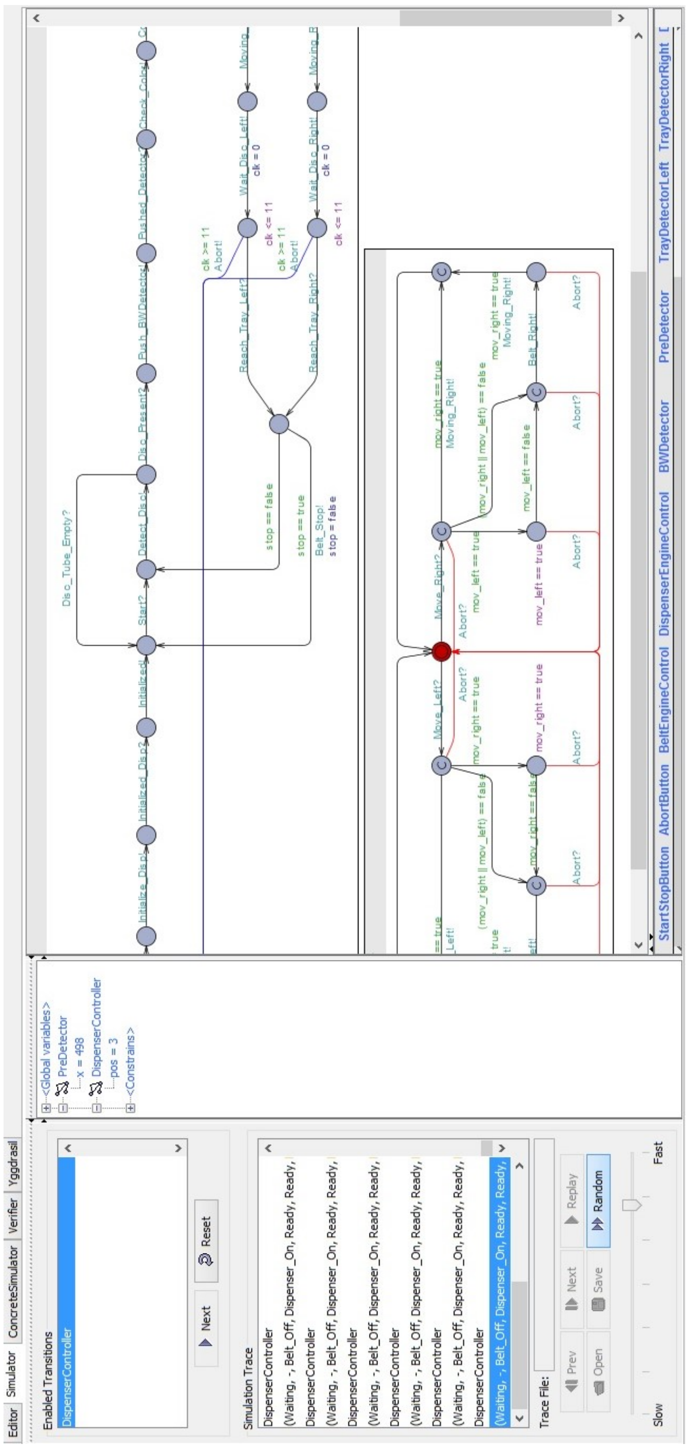
Jiddo van Vliet

The project is about designing and building a machine which is able to sort black and white discs. Before the project I had no idea how much documentation is actually needed to design, build and most of all program the machine. Luckily we had a really detailed technical guide which was helpful for making deadlines for the different tasks and dividing the work between the different group members.

For this project I was mostly responsible for writing the documentation, writing a part of the Assembly language code and working on the final report.

Overall my experience in this group was really good. Of course not everybody is at the same level at writing code but people made up for that by writing more documentation or building the machine. The project itself was really interesting with all the different aspects needed to realize our sorting machine. It taught me a lot about properly documenting our progress and how much preparation was needed for the different phases of the project. It also taught us how to divide the workload, determine deadlines and how to have a good structure during our weekly meetings.

APPENDIX A: Figures



```
DispenserController
(Waiting, -, Belt_Off, Dispenser_On, Ready, Ready,
DispenserController
(Waiting, -, Belt_Off, Dispenser_On, Ready, Ready,
DispenserController
(Waiting, -, Belt_Off, Dispenser_On, Ready, Ready,
DispenserController
(Waiting, -, Belt_Off, Dispenser_On, Ready, Ready,
DispenserController
(Waiting, -, Belt_Off, Dispenser_On, Ready, Ready,
DispenserController
(Waiting, -, Belt_Off, Dispenser_On, Ready, Ready,
```

Figure 3: GIVE US A CAPTION YUNTAO

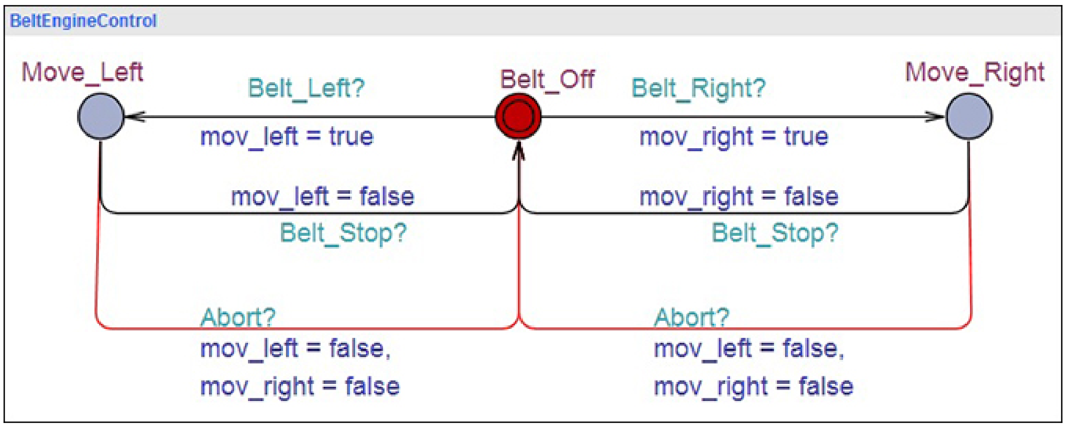


Figure 4: GIVE US A CAPTION YUNTAO

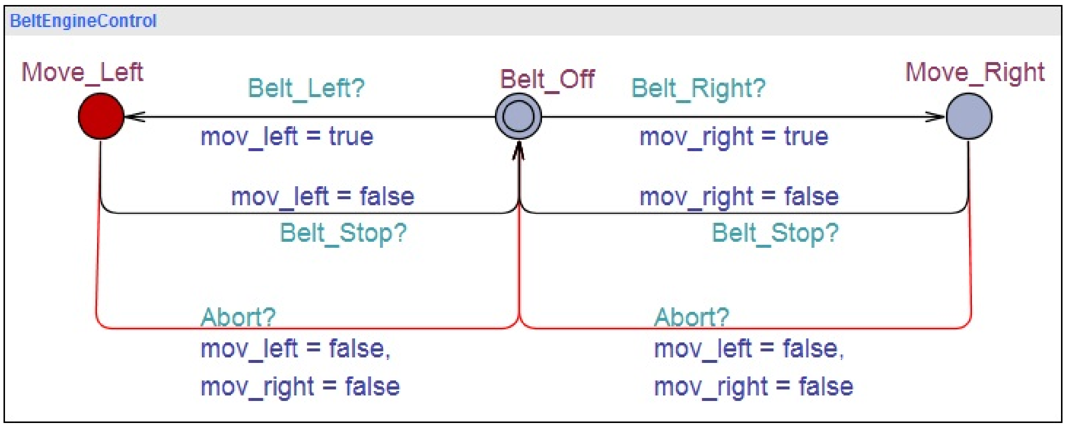


Figure 5: GIVE US A CAPTION YUNTAO

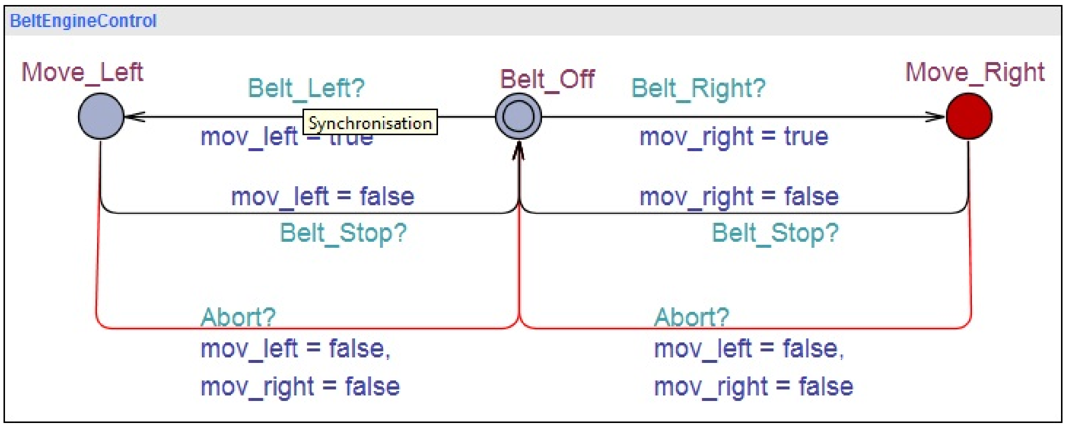


Figure 6: GIVE US A CAPTION YUNTAO

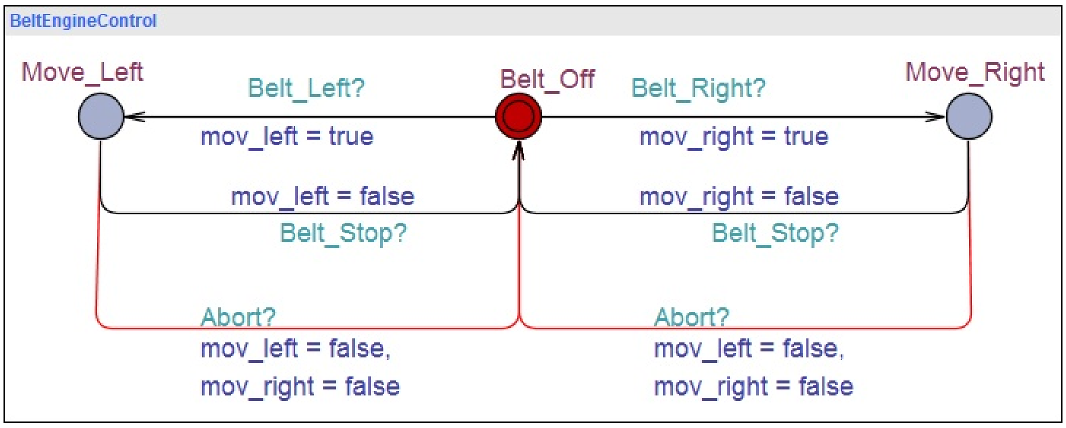


Figure 7: GIVE US A CAPTION YUNTAO

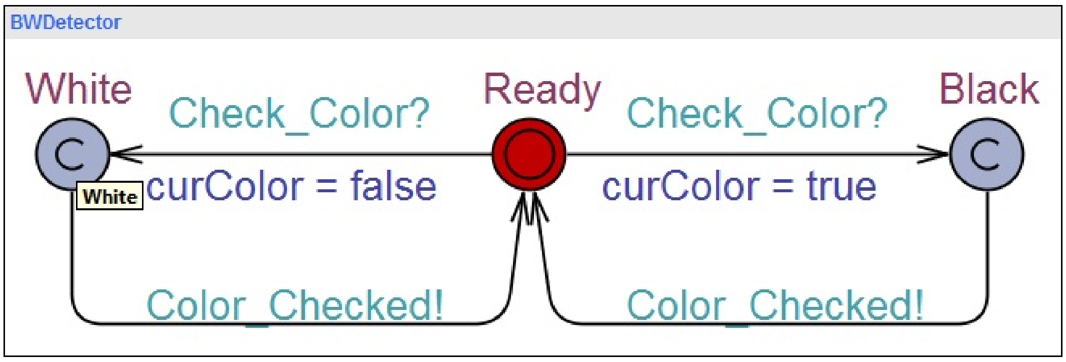


Figure 8: GIVE US A CAPTION YUNTAO

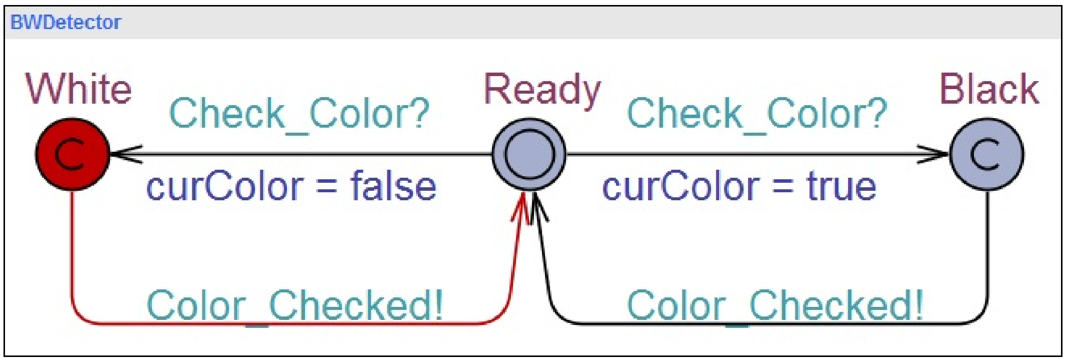


Figure 9: GIVE US A CAPTION YUNTAO

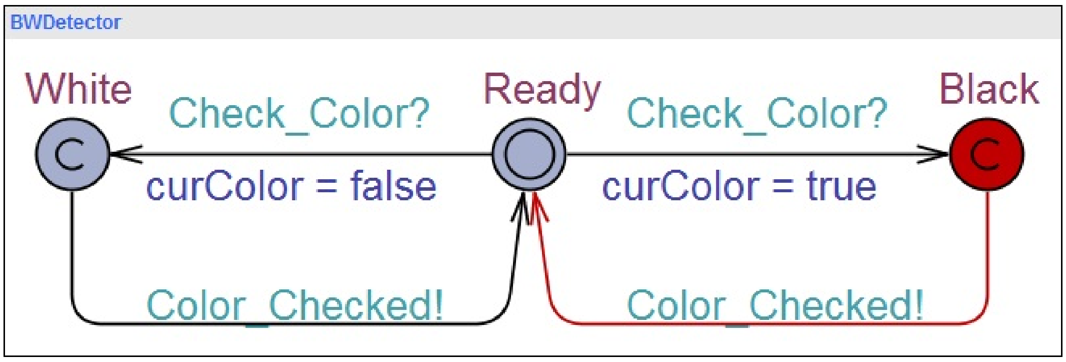


Figure 10: GIVE US A CAPTION YUNTAO

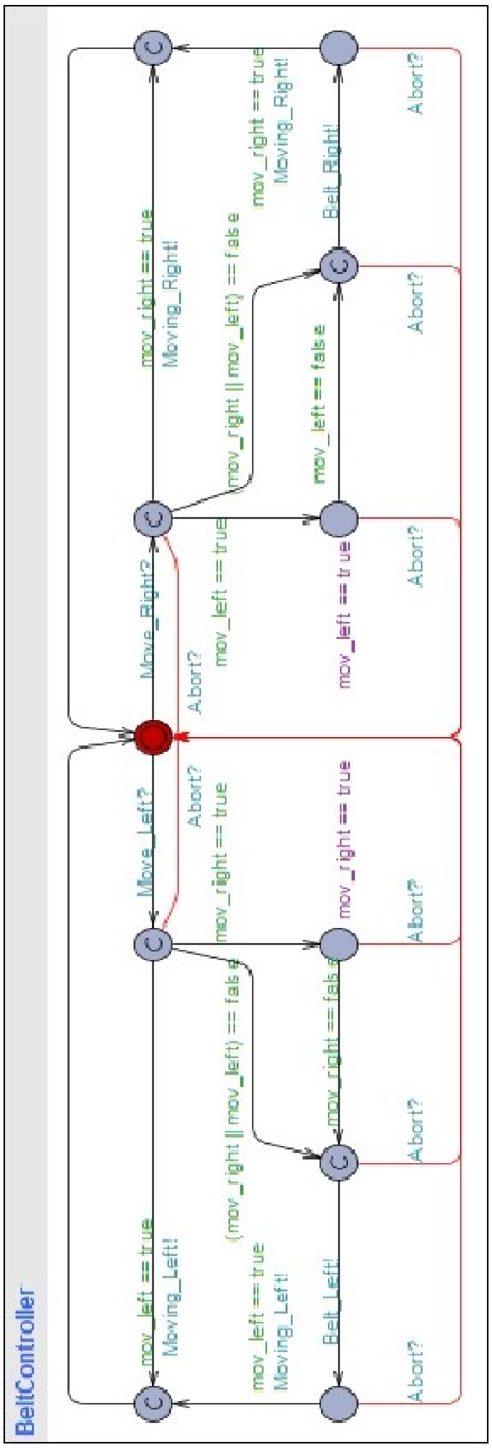


Figure 11: GIVE US A CAPTION YUNTAO

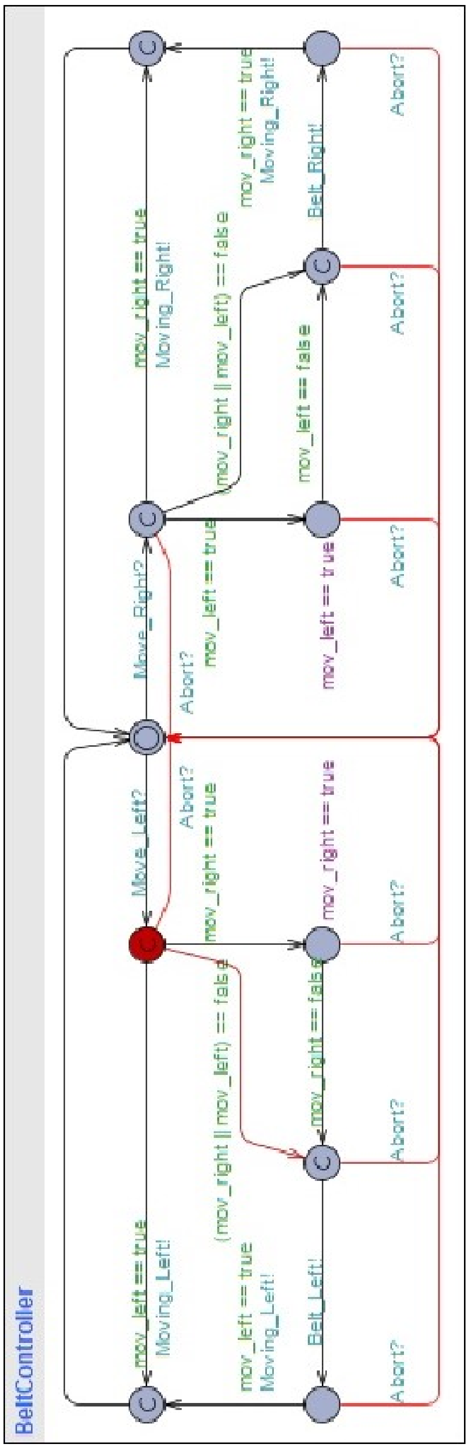


Figure 12: GIVE US A CAPTION YUNTAO

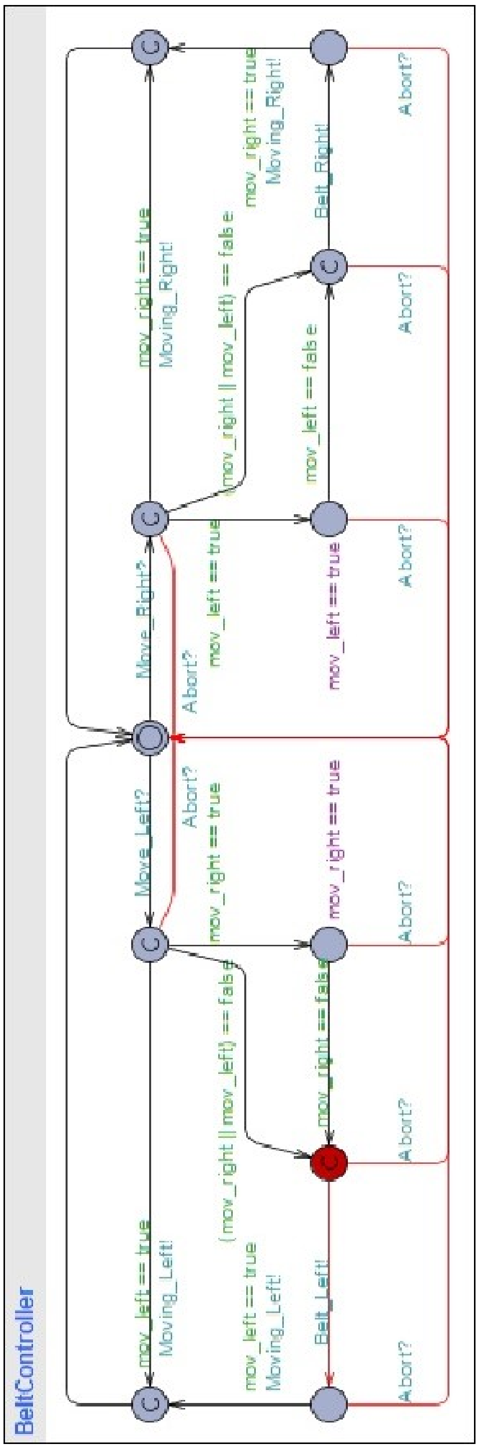


Figure 13: GIVE US A CAPTION YUNTAO

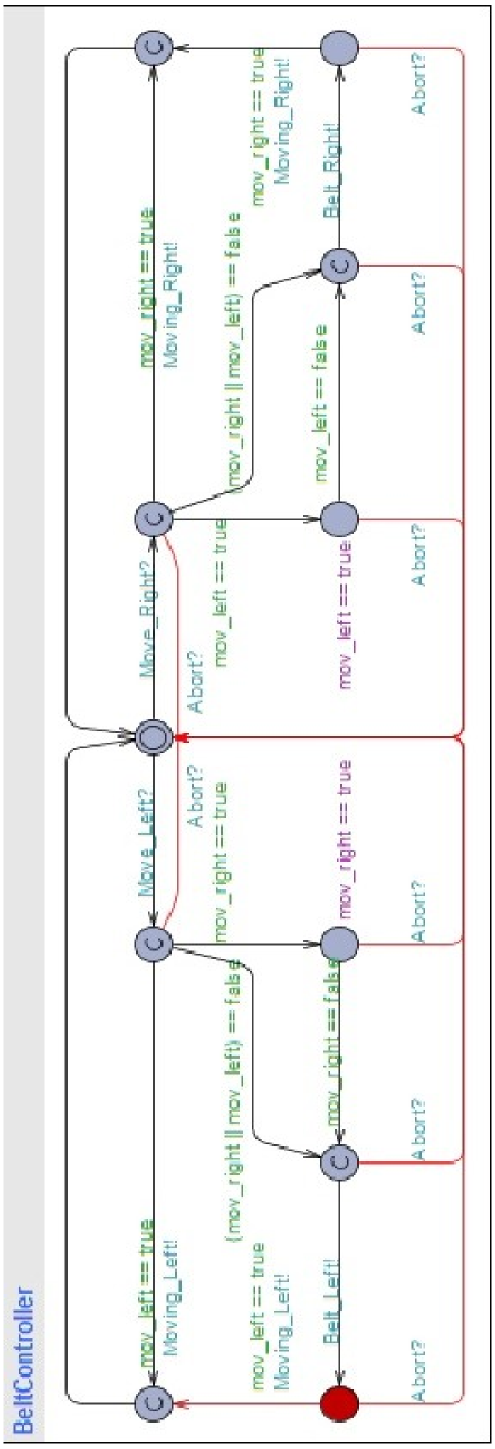


Figure 14: GIVE US A CAPTION YUNTAO

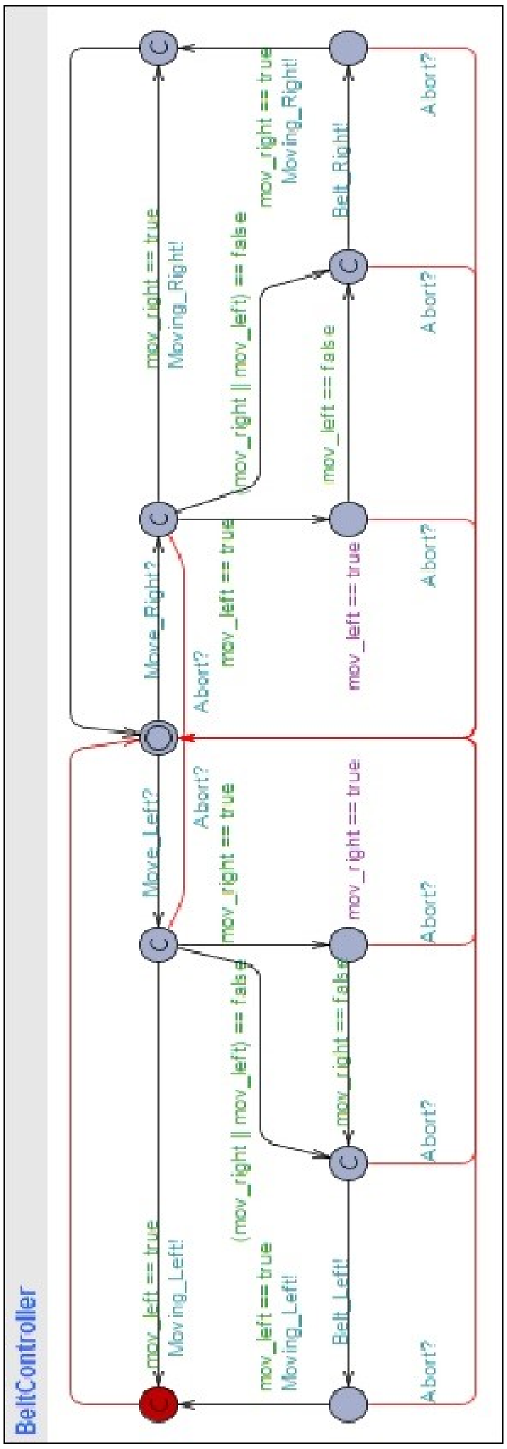


Figure 15: GIVE US A CAPTION YUNTAO

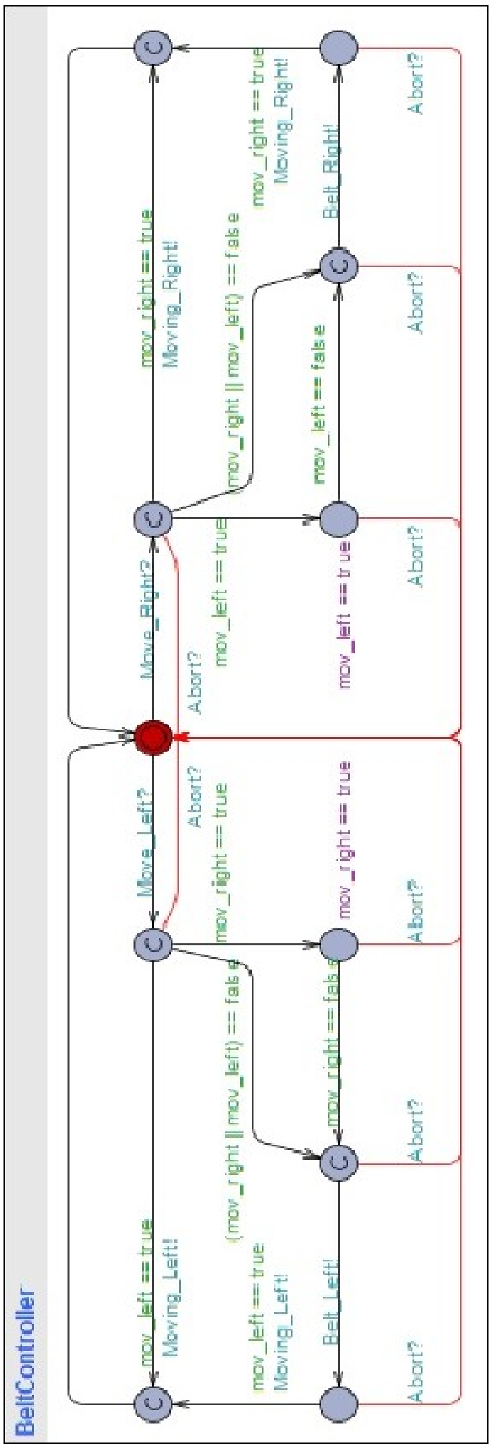


Figure 16: GIVE US A CAPTION YUNTAO

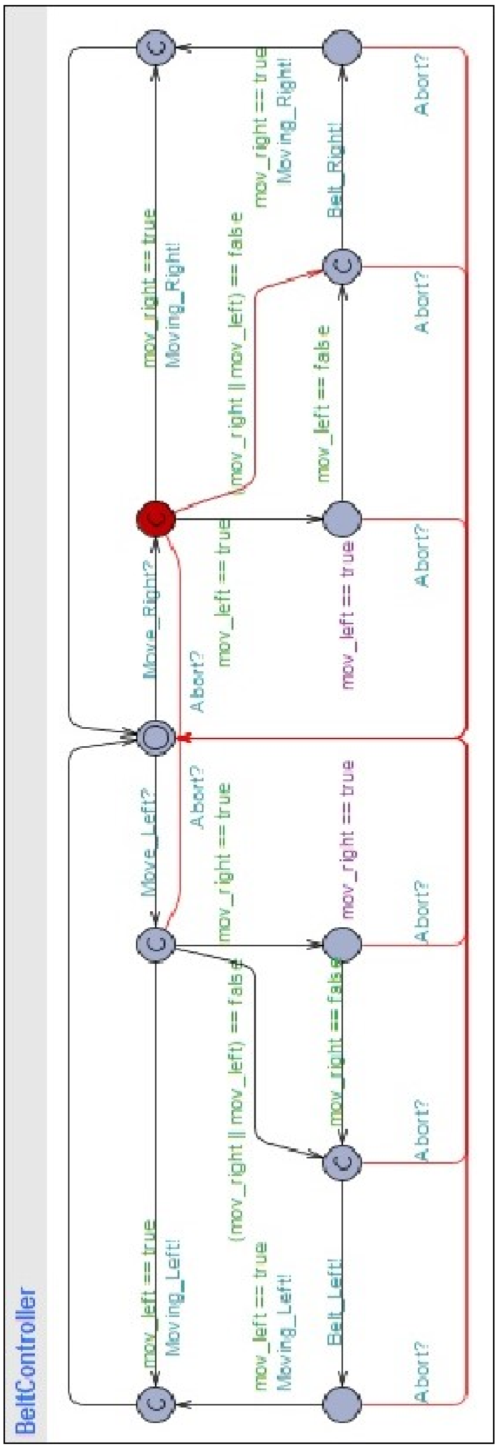


Figure 17: GIVE US A CAPTION YUNTAO

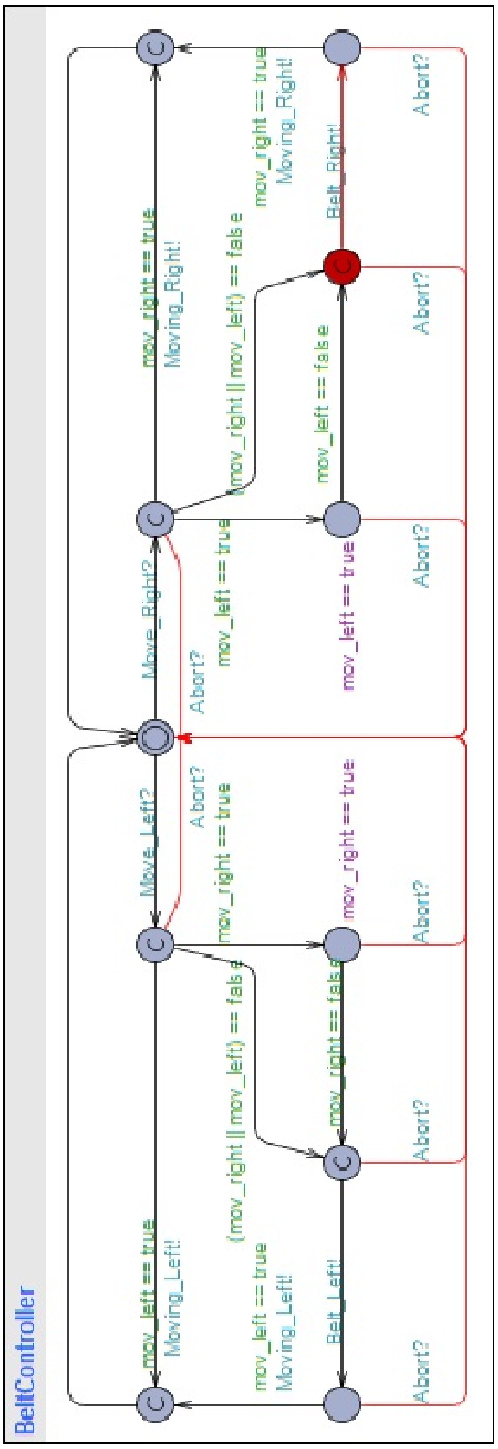


Figure 18: GIVE US A CAPTION YUNTAO

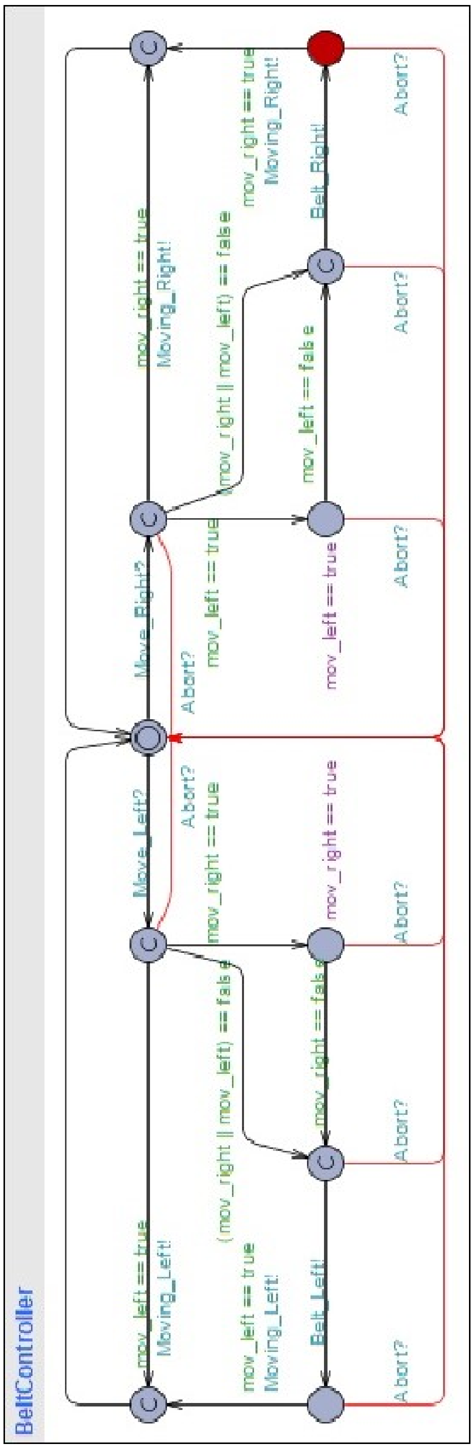


Figure 19: GIVE US A CAPTION YUNTAO

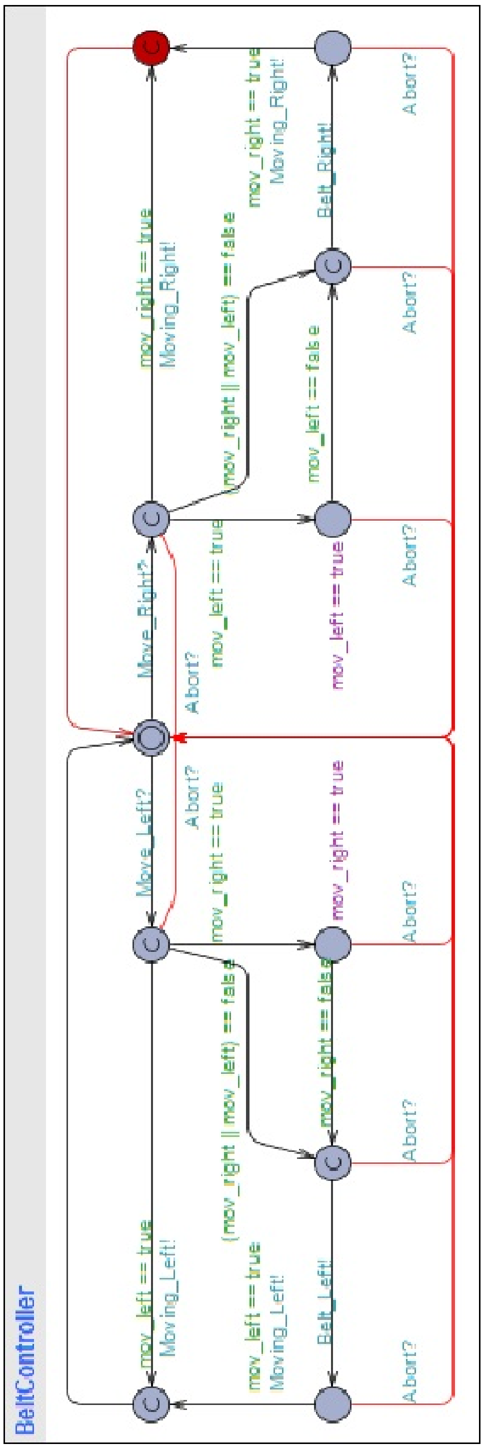


Figure 20: GIVE US A CAPTION YUNTAO

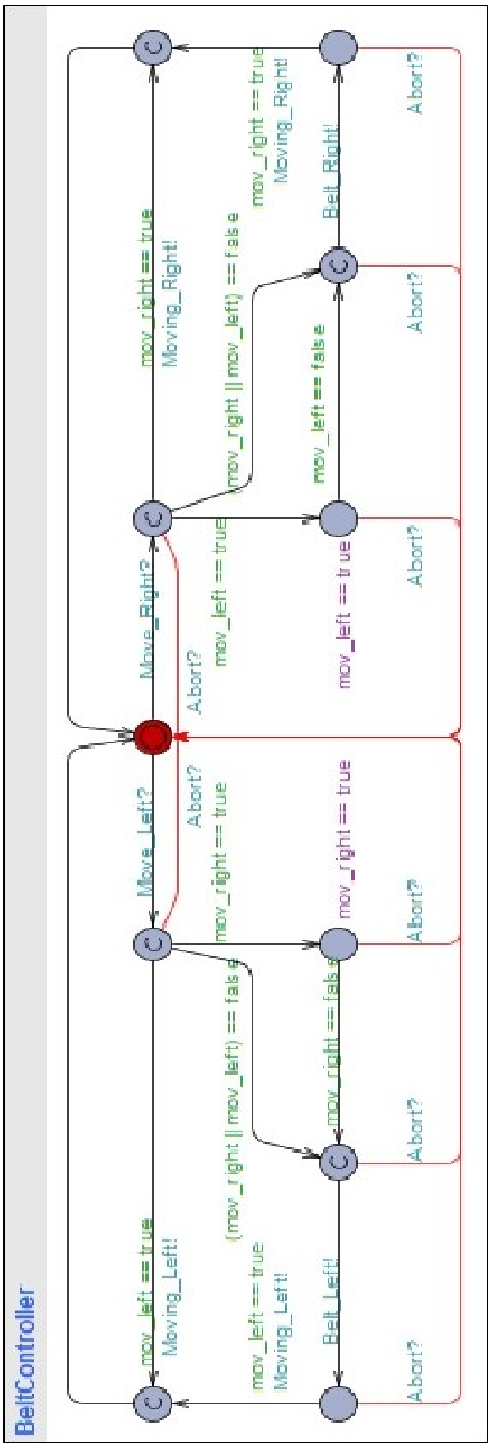
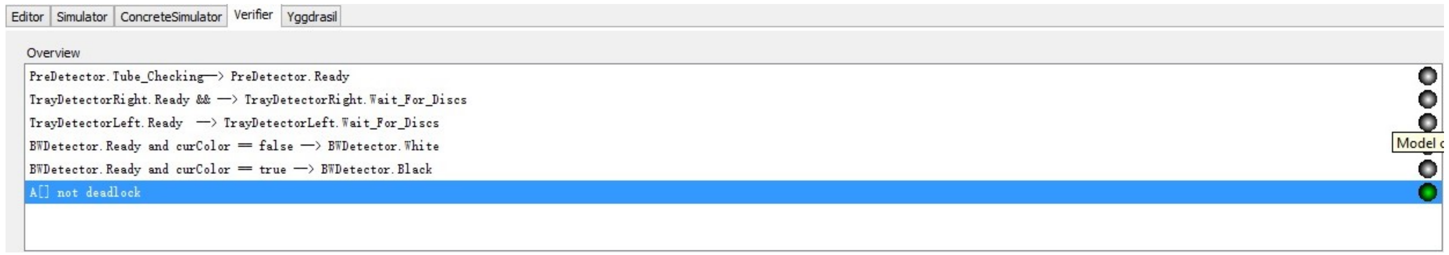
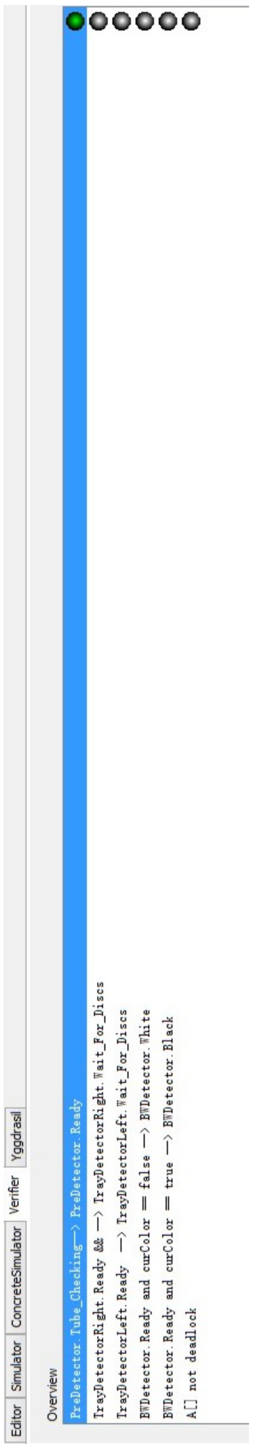


Figure 21: GIVE US A CAPTION YUNTAO



```
PreDetector.Tube_Checking—> PreDetector.Ready
TrayDetectorRight.Ready && —> TrayDetectorRight.Wait_For_Discs
TrayDetectorLeft.Ready  —> TrayDetectorLeft.Wait_For_Discs
BWDetector.Ready and curColor == false —> BWDetector.White
BWDetector.Ready and curColor == true —> BWDetector.Black
A[] not deadlock
```

Figure 22: GIVE US A CAPTION YUNTAO



```
PreDetector.Tube_Checking—> PreDetector.Ready
TrayDetectorRight.Ready && —> TrayDetectorRight.Wait_For_Discs
TrayDetectorLeft.Ready  —> TrayDetectorLeft.Wait_For_Discs
BWDetector.Ready and curColor == false —> BWDetector.White
BWDetector.Ready and curColor == true —> BWDetector.Black
A[] not deadlock
```

Figure 23: GIVE US A CAPTION YUNTAO