Software Specification

13th of March 2015

2IO70

Version 1

The purpose of this document is to give an as accurately as possible description of the required behaviour of the PP2, without describing how this is achieved, and a UPPAAL model of this behaviour. In order to do this, we translate the system level requirements to a high level specification of what the software controlling the physical machine should do.

Group 16

Rolf Verschuuren

Wigger Boelens

Stefan van den Berg

Dat Phung

Maarten Keet

Tudor Petrescu

Table of Contents

Inputs and Outputs

Inputs

Input	The range/type of the value
Start/Stop button	Boolean value
Abort button	Boolean value
Push button(sensor)	Boolean value
Position sensor	Boolean value
Colour detector	Boolean value
Timer	Values range from seconds to clock ticks, consists of TEnd, Motor Down, Motor Up, Sort, Belt, and Tic

The Start/Stop and Abort buttons speak for themselves. They are either pressed or not pressed.

Push button(sensor); the sorter touches the push sensor or doesn't touch it, to detect the sorter's position.

The position sensor and colour detector are either on or off.

Timer

All given times are calculated by taking the average time of ten measurements. The input of a timer is set or not set.

TEnd is the moment of termination of the timer, so when the timer reaches zero. The timer is a count-down timer that is set to a certain value and runs at a frequency of 10 kHz.

Motor Down is defined as the time it takes for the engine of the sorter to move the sorter from the lowest point to the highest point, until sorting mechanism touches the push sensor. This takes 0.30 seconds.

Motor Up is the state of the sorter moving from the highest point to the bottom of the engine sorter. Since the engine sorter for Motor Down and Motor Up have the same voltage, this will take 0.30 seconds as well.

Sort is the amount of time it takes for a disc to be transported from the black/white detector to the end of the conveyor belt, which is measured to be 0.85 seconds.

Belt is the period that a disc travels from the feeder to the end of the conveyor belt, until the disc reaches the tray for black discs. This action takes 2.0 seconds.

Tic is defined as one clock tick of the PP2. A clock tick is incredibly fast.

Outputs

Output	The range/type of the value
Lens lamp position	Boolean value
Lens lamp sorter	Boolean value
Conveyor engine	Between 6 and 9 V while running 0 V when not running
Feeder engine	Between 3 and 7 V while running 0 V when not running
Hbridge0	Boolean value, current is between 6 and 9 V while running
Hbridge1	Boolean value, current is between 6 and 9 V while running
Display	Integer value, positive
LED state indicator	Boolean value
Timer start	Values range from seconds to clock ticks, consists of TEnd, Motor Down, Motor Up, Sort, Belt, and Tic

Lens lamp position and **lens lamp sorter** are the lamps that make up part of the sensors and can be turned on or off.

The **conveyor and feeder engines** respectively move the conveyor belt and the feeder. They are either on or off.

Hbridge0 indicates whether the sorter moves up or not. On the other hand, whereas **Hbridge1** shows that the sorter moves down or halts.

The **display** shows the number of discs that have been sorted. Depending on the available time, we might or might not implement this.

The **LED state indicator** announces whether the machine is in its resting state or not.

The **Timer start** output is the same as the Timer input, except that the timer counts down.

Validation of "Inputs and Outputs"

We see that the inputs and outputs of Software Specification are correct. The inputs of Machine Design should be equal to the outputs of Software Specification, which they are.

Relations

Lens lamp of the black white detector

The lens lamp of the black white detector will be on when the machine is sorting. Thus the lens lamp will react to the input of the “START/STOP” button and the “ABORT” button. The lens lamp will go on when the machine is in resting state and the “START/STOP” button is pressed and it will go off when the “ABORT” button is pressed while the machine was running.

Lens lamp of the position sensor

The lens lamp of the position sensor reacts only to the “START/STOP” button and the “ABORT” button. The lens lamp will be on after the “START/STOP” button is pressed and the machine is in its resting state. If at any other point in time the “ABORT” button is pressed it will go off. When the “START/STOP” button is pressed and the machine is running then the lens lamp also goes off.

Engine of the conveyor belt

The engine of on the conveyor belt only reacts to the input of the “START/STOP” button and the “ABORT” button. The engine will start when the machine is in its resting state and the “START/STOP” button is pressed. If however the “START/STOP” button is pressed and the machine is not in its resting state then the machine will stop after it completed its current cycle. Whenever the “ABORT” button is pressed the engine stops within 50ms.

Engine of the feeder

The engine for the feeder also only reacts to the input of the “START/STOP” button and the “ABORT” button. This engine also starts when the machine is in its resting state and the “START/STOP” button is pressed. If however the machine is running then the engine will stop. When the “ABORT” button is pressed the engine stops within 50ms.

Engine for the sorter

When the machine is running the engine of the sorter reacts to inputs of the colour detector, the push sensor and the timer. When a signal is received from the colour detector the engine pushes the sorter up, the engine then waits until the timer gives a signal to go down again after it let the discs through, it knows when it is in the correct “up” position from the push sensor . If the “START/STOP” button is pressed when the machine is in its resting state, then the sorter will wait for a signal from the timer that marks the end of the current cycle. If at any time the ““ABORT”” button is pressed, the sorting mechanism is to stop within 50ms.

Display for counting

The display output depends on how many times the colour detector detects a white disc and how many times a disc passes the position sensor without the colour detector detecting it.

In the initial state the counters get reset.

Validation of “Relations”

The relations between the inputs and outputs can be validated with the input/output tables. For all inputs, we have outputs. These outputs depend on one or more inputs, which is described in the Relations.

Design Decisions

Feeder

The feeder in constantly on because of priority 2, speed, mentioned in the Machine Design document. Another reason is that there's a turning part that needs to spin through to get to it's initial position to be able to deposit discs again.

Lens lamp position

We chose to have the lens lamp for position sensor constantly on, because it's easier to code resulting in spending less time on it. The optimization is minimal if we would turn them off every time there's a gap between discs, because of the feeder being quite fast in depositing the next disc.

Conveyor belt

The conveyor belt is constantly running, because the feeder is constantly pushing discs onto the conveyor belt. This goes hand in hand with our second priority, which is speed.

Lens lamp colour

Like with the position sensor, it's easier to code that it is continuously on. The light being off if it's possible, would again be a minimal improvement, because the gaps between discs being pushed on the conveyor belt is the same as with the black white detector.

Push button

We use the push button, because of priority 1, correctness, to know if the sorter arm is at its highest point. We need to know this, because we need to know when to stop the motor making the sorter arm going up.

Description of States

Initial_state

In the initial state the machine starts calibrating the sorting mechanism by moving it up.

Outputs	Value for output
Lens lamp position	0
Lens lamp sorter	0
Engine conveyor	0
Engine feeder	0
Hbridge0	0
Hbridge1	0
Display	0
LED state indicator	0
Timer start	0

Calibrate_Sorter

In the calibrate sorter state the sorting mechanism moves down until it is just above the conveyor belt.

Outputs	Value for output
Lens lamp position	0
Lens lamp sorter	0
Engine conveyor	0
Engine feeder	0
Hbridge0	1
Hbridge1	0
Display	0
LED state indicator	0
Timer start	0

Resting_state

In the resting state the sorting machine is at rest and waiting for the user to press the START/STOP button.

Outputs	Value for output
Lens lamp	0
Lens lamp	0
Engine conveyor	0
Engine feeder	0
Hbridge0	0
Hbridge1	0
Display	0
LED state indicator	1
Timer start	0

Running_state

In the running state the sorting mechanism, the conveyor belt, the position detector, and the colour detector are turned on.

Outputs	Value for output
Lens lamp position	1
Lens lamp sorter	1
Engine conveyor	1
Engine feeder	1
Hbridge0	0
Hbridge1	0
Display	0
LED state indicator	0
Timer start	2 s + Belt

Running_Wait

In this state a disc has been detected and that disc is moving along the conveyor belt to the sorter.

Outputs	Value for output
Lens lamp position	1
Lens lamp sorter	1
Engine conveyor	1
Engine feeder	1
Hbridge0	0
Hbridge1	0
Display	0
LED state indicator	0
Timer start	2 s + Belt

Running_Timer_Reset

In this state a new disc was detected and the timer has been reset.

Outputs	Value for output
Lens lamp position	1
Lens lamp sorter	1
Engine conveyor	1
Engine feeder	1
Hbridge0	0
Hbridge1	0
Display	0
LED state indicator	0
Timer start	2 s + Belt

Motor_Up

In this state the motor of the sorter is moving up until it hits the push button.

Outputs	Value for output
Lens lamp position	1
Lens lamp sorter	1
Engine conveyor	1
Engine feeder	1
Hbridge0	1
Hbridge1	0
Display	0
LED state indicator	0
Timer start	Sort

Motor_Up_Stop

In this state the motor of the sorter is moving up until it hits the push button. And the machine has to stop because the start stop button was pressed.

Outputs	Value for output
Lens lamp position	1
Lens lamp sorter	1
Engine conveyor	1
Engine feeder	1
Hbridge0	1
Hbridge1	0
Display	0
LED state indicator	0
Timer start	Sort

Motor_Down

In the Motor_Down state, the sorter is moved down.

Outputs	Value for output
Lens lamp position	1
Lens lamp sorter	1
Engine conveyor	1
Engine feeder	1
Hbridge0	0
Hbridge1	1
Display	0
LED state indicator	0
Timer start	0

Motor_Down_Stop

In Motor_Down_Stop, the sorter is moved down, after the start/stop button has been pressed.

Outputs	Value for output
Lens lamp position	1
Lens lamp sorter	1
Engine conveyor	1
Engine feeder	1
Hbridge0	0
Hbridge1	1
Display	0
LED state indicator	0
Timer start	0

White_Wait

In this state the machine waits until the colour detector has detected a white disc.

Outputs	Value for output
Lens lamp position	1
Lens lamp sorter	1
Engine conveyor	1
Engine feeder	1
Hbridge0	0
Hbridge1	0
Display	0
LED state indicator	0
Timer start	Sort

White_Wait_Stop

In this state the machine waits until the colour detector has detected a white disc, after the START/STOP button has been pressed.

Outputs	Value for output
Lens lamp position	1
Lens lamp sorter	1
Engine conveyor	1
Engine feeder	1
Hbridge0	0
Hbridge1	0
Display	0
LED state indicator	0
Timer start	Sort

Running_Timer

Running_Timer is the state that sets the interrupt timer to make sure the machine stops after the current cycle.

Outputs	Value for output
Lens lamp position	1
Lens lamp sorter	1
Engine conveyor	1
Engine feeder	0
Hbridge0	0
Hbridge1	0
Display	0
LED state indicator	0
Timer start	Belt

Motor_Up_Timer

Motor_Up_Timer is the state that sets the interrupt timer to make sure the machine stops after the current cycle.

Outputs	Value for output
Lens lamp position	1
Lens lamp sorter	1
Engine conveyor	1
Engine feeder	0
Hbridge0	0
Hbridge1	0
Display	0
LED state indicator	0
Timer start	Belt

White_Wait_Timer

White_Wait_Timer is the state that sets the interrupt timer to make sure the machine stops after the current cycle.

Outputs	Value for output
Lens lamp position	1
Lens lamp sorter	1
Engine conveyor	1
Engine feeder	0
Hbridge0	0
Hbridge1	0
Display	0
LED state indicator	0
Timer start	Belt

Motor_Down_Timer

Motor_Down_Timer is the state that sets the interrupt timer to make sure the machine stops after the current cycle.

Outputs	Value for output
Lens lamp position	1
Lens lamp sorter	1
Engine conveyor	1
Engine feeder	0
Hbridge0	0
Hbridge1	0
Display	0
LED state indicator	0
Timer start	Belt

Aborted

Aborted is the state where the machines goes to if the abort button is pressed, the machine has come to a halt.

Outputs	Value for output
Lens lamp position	0
Lens lamp sorter	0
Engine conveyor	0
Engine feeder	0
Hbridge0	0
Hbridge1	0
Display	0
LED state indicator	0
Timer start	0

Running_Stop

Running_Stop gives the same outputs as the Running state, the only difference being a running timer in the stop process.

Outputs	Value for output
Lens lamp position	1
Lens lamp sorter	1
Engine conveyor	1
Engine feeder	0
Hbridge0	0
Hbridge1	0
Engine sorter	0
Display	0
LED state indicator	0
Timer start	Belt

Validation of "Description of States"

To validate the states we will look at the USE-cases again to see if every USE-case is implemented. To do this we look at the basic flow and trigger of every use case and see what states we use to realize this.

We also validate the states to the relations. For every USE-case we looked at what states would be necessary to achieve it.

Starting the machine

Preconditions: -
Trigger: Booting the machine / finished the abort or start/stop routine
Postconditions: The machine starts the sorting process.

Basic Flow	State	Explanation
Before Trigger	Any State	It does not really matter which state the machine is in before the trigger
After Trigger	Initial State	Initial state is the first state, so after booting the machine we will be here. Finishing the abort or start/stop routine will also end in the initial state
1. Machine puts devices in their initial state.	Initial State + Calibrate Sorter + Resting State	The only thing that needs to be put into an initial state is the sorter mechanism. In initial state the machine moves the sorter up until it touches the push button. It then transitions to Calibrate Sorter where it starts moving down. After a set amount of time it will stop moving the sorter and transition to the resting state. This way we know exactly where the sorter is positioned
2. The user presses the START/STOP button	Running State	From the Resting State the transition to the running state is pressing the START/STOP button
Post-conditions	Running State	The running state is the start of the sorting process

Stopping the machine

Preconditions: The machine is running.
Trigger: The START/STOP button is pressed.
Post-conditions: The machine is sent into an inactive state with no process interrupted.

	Basic Flow	State	Explanation
	Preconditions	Not initial state, Calibrate Sorter or aborted	When the machine is not in any of these states it is running.
	After Trigger	One of the (greenblue) Timer states	When the START/STOP is pressed the machine transitions to a timer start state, which starts a timer and stops the feeder mechanism.
1.	The machine finishes sorting the discs currently in the machine	One of the sorting states	While the timer is running the machine keeps sorting. The timer is the time it takes for the conveyor belt to make a complete rotation, guaranteeing there are no more discs on the belt.
2.	The machine enters an inactive state and will not take any more discs form the storage* unless the START/STOP button is pressed.	Initial State + Calibrate Sorter + Resting State	After going through the initialize process we go back to the resting state, which waits on the START/STOP button.
	Postconditions	Resting State	Resting state in an inactive state and we finished the sorting process.

Sort unsorted discs

Preconditions: The machine is not already running.
Trigger: The user provides unsorted discs and presses the "START" button.
Postconditions: There are no unsorted discs left, all sorted discs are in a container based on their colour

Basic Flow	State	Explanation
Preconditions	Resting State	The program first initializes and then waits for the user to press that start button. This waiting happens in the Resting State. In the resting state the machine is not running
After Trigger	Running State	Pressing START/STOP is the input to transition to the running state
1. An unsorted disc is moved to the colour detector	Running Wait + Running Timer Rest	When moving to the colour detector it will have to pass the position Sensor which is the input to move to Running Wait, the disc is then still in front of the position sensor so the program moves to Running Timer Rest
2. The machine decides to which of the two containers the disc needs to be moved	Running Wait + Running Timer Rest OR Motor Up + White-Wait	Depending on whether the disc is white or black the sorter either needs to move down or keep its down position. If it keeps its down position it should just keep checking for an unsorted disc and when it detects one it will move to Running Timer Rest If it needs to move up the colour detector will detect a white disc and therefore transition to Motor Up. Moving the sorter up will trigger the pushButton, which is the input to transition to White-Wait
3. The machine moves the disc to the designated container	Running Wait + Running Timer Rest OR Motor Down + Running Wait	If the sorter did not detect a white disc we are still waiting like in basic flow 2. If it did detect one then while the disc is moving to the designated container the sorttimer will count down making the machine transition to Motor Down
4. The machine repeats step 2 through 4 until all discs have been sorted	-	
5. The machine pauses within 4 seconds	Initial State + Calibrate Sorter + Resting State	If there are no discs anymore the machine will stay in Running Wait waiting for the timer interrupt which will come within 4 seconds, making the machine transition to initial state. There it will reset the sorter and transition to the resting state
Post-conditions	Resting State	We repeated the sorting step until all discs where sorted, meaning all discs are now sorted

Abort the process

Preconditions: The machine is sorting discs
Trigger: The user wants to immediately stop the machine.
Post-conditions: The machine stopped running and is ready to start again.

	Basic Flow	State	Explanation
	Preconditions	Every that is not initial state, Calibrate Sorter, resting state or Aborted	All other states are states in which discs are being sorted
	After Trigger	Aborted	Every state (apart from the one mentioned in before trigger) have a line to abort with Abort as input
1.	The machine stops transporting the discs. And doesn't put any more discs on the transporting mechanism._	Aborted	Because the machine is now in the abort state, which has all outputs set to 0, nothing will be moving.
2.	The user is required to remove all discs that are neither in the container unit nor sorted.	Aborted	The machine will remain in Abort until the user presses START/STOP. This means everything is stopped and the user can safely remove all discs
3.	When the user removed all unsorted discs that where not in the container unit he presses the START/STOP button.	Initial State + Calibrate Sorter + Resting State	Pressing the START/STOP button is the input for the transition to Initial State There it will reset the sorter and transition to the resting state
	Post-conditions	Resting State	We are in the resting state, so the machine has stopped running. The resting State is also the state from which you can start the machine again

Booting of the machine & Shutting down the machine does nothing with our software. This means they do not use states. This also means we can't validate those USE-Cases here.

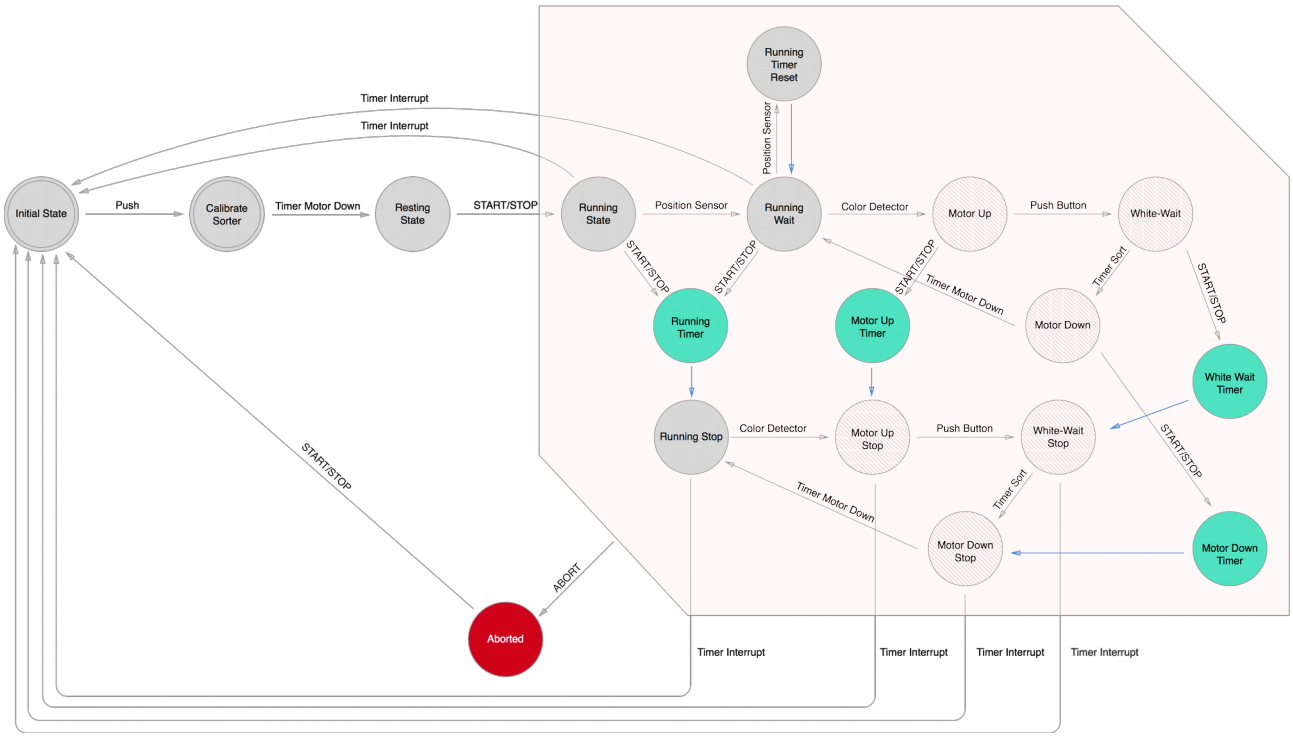
State transitions

Current state	Input	Input value	Next State
Initial	Push	1	Calibrate_Sorter
Calibrate_Sorter	Push	0	Resting
Resting	StartStop	1	Running
Running	Timer	TEnd	Initial
Running	PositionSensor	0	Running_Wait
Running	Abort	1	Aborted
Running	StartStop	1	Running_Timer
Running_Wait	Timer	TEnd	Initial
Running_Wait	PositionSensor	0	Running_Timer_Reset
Running_Wait	ColorDetector	1	MotorUp
Running_Wait	StartStop	1	Running_Timer
Running_Wait	Abort	1	Aborted
Running_Timer_Reset	Tick	1	Running_Wait
Running_Timer_Reset	Abort	1	Aborted
MotorUp	PushButton	1	WhiteWait
MotorUp	StartStop	1	Motor_Up_Timer
MotorUp	Abort	1	Aborted
WhiteWait	StartStop	1	White_Wait_Timer
WhiteWait	Abort	1	Aborted
WhiteWait	Timer	SORT	MotorDown
MotorDown	StartStop	1	Motor_Down_Timer
MotorDown	Abort	1	Aborted
MotorDown	Timer	Motor Down	Running_Wait
Running_Timer	Timer	Tic	Running_Stop
Running_Timer	Abort	1	Aborted
Motor_Up_Timer	Timer	Tic	Motor_Up_Stop
Motor_Up_Timer	Abort	1	Aborted
White_Wait_Timer	Timer	Tic	White_Wait_Stop
White_Wait_Timer	Abort	1	Aborted
Motor_Down_Timer	Timer	Tic	Motor_Down_Stop
Motor_Down_Timer	Abort	1	Aborted
Motor_Up_Stop	PushButton	1	White_Wait_Stop
Motor_Up_Stop	Abort	1	Running_Stop
Motor_Up_Stop	Timer	Timer Interrupt	Initial
Motor_Up_Stop	Abort	1	Aborted
White_Wait_Stop	Timer	SORT	Motor_Down_Stop
White_Wait_Stop	Abort	1	Aborted
White_Wait_Stop	Timer	Timer Interrupt	Initial
Motor_Down_Stop	Timer	Motor Down	Running_Stop
Motor_Down_Stop	Abort	1	Aborted
Motor_Down_Stop	Timer	Timer Interrupt	Initial
Running_Stop	ColorDetector	1	Motor_Up_Stop
Running_Stop	Abort	1	Aborted
Running_Stop	Timer	Timer Interrupt	Initial
Aborted	StartStop	1	Initial

Validation of "State Transitions"

The description of our machine states is validated through its representation in the transition table. No state is excluded from being represented in the state transition table, all transitions will have the initial transition state differ from the end state.

Finite state Automaton



Blue line means that the trigger for the transition is a clocktick

Validation of "Finite state Automaton"

When we were making our finite state automaton we looked at our state description and made sure that all states were represented, then we used our state transition table to make sure all transitions were correctly implemented.

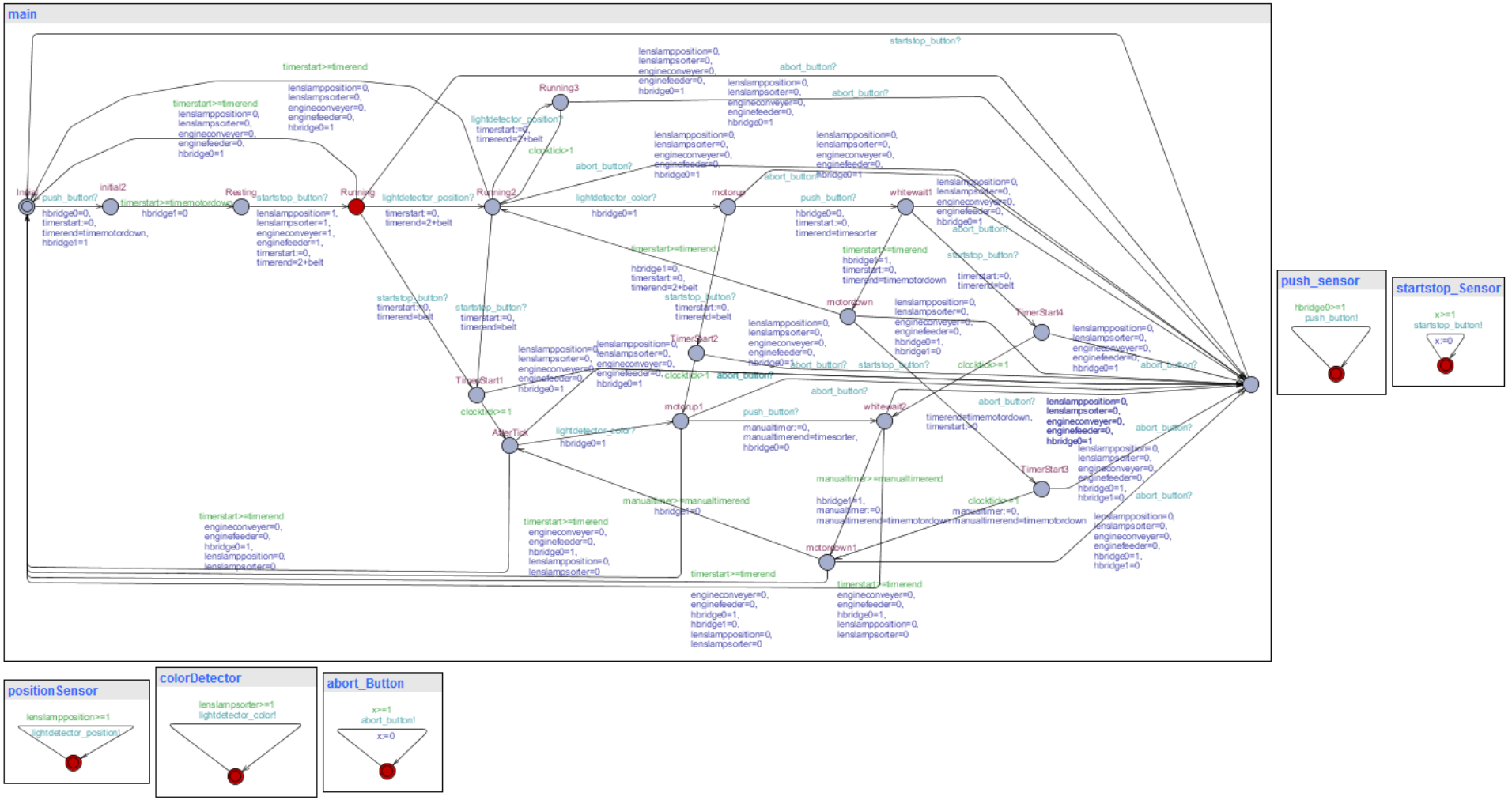
UPPAAL model

Tests done

On the next page is the UPPAAL model. This UPPAAL model has been tested for 2 safety properties. The first one is " After the start-up of the machine, the assembly program should not stop until the machine is shut down.". This has been tested using the following property "A[] not deadlock", and we didn't have a deadlock. The second safety property which was tested is: "The outputs connected to the h-bridge may never be powered on at the same time.". This was tested using the following property "A<> !(hbridge0==1 && hbridge1=1)". This one was also correct.

Validation of "UPPAAL model"

All transitions which exist in the UPPAAL model also occur in the Finite State Automaton. And the same action has to be performed to take that transition. Also all states of the Finite State Automaton occur in the UPPAAL model. The states of the UPPAAL model also have the outputs in them. The states of the Finite State Automaton do not have the outputs in them. Thus we validate the values of the outputs, which are in the states, to the description of the states.


main
Init
initial2
Resting
Running
Running2
Running3
whitewait1
whitewait2
motorup
motorup1
motordown
motordown1
TimerStart1
TimerStart2
TimerStart3
TimerStart4
AfterTick
push_button?
startstop_button?
abort_button?
lightdetector_position?
lightdetector_color?
timerstart>=timerend
clocktick>1
clocktick>=1
hbridge0=0,
timerstart:=0,
timerend=timemotordown,
hbridge1=1
timerstart>=timemotordown
hbridge1=0
lenslampposition=1,
lenslampsorter=1,
engineconveyer=1,
enginefeeder=1,
timerstart:=0,
timerend=2+belt
timerstart:=0,
timerend=2+belt
hbridge0=1
hbridge0=0,
timerstart:=0,
timerend=timesorter
hbridge1=0,
timerstart:=0,
timerend=2+belt
timerstart:=0,
timerend=belt
hbridge1=1,
timerstart:=0,
timerend=timemotordown
timerstart:=0,
timerend=belt
manualtimer:=0,
manualtimerend=timesorter,
hbridge0=0
hbridge1=1,
manualtimer:=0,
manualtimerend=timemotordown
manualtimer>=manualtimerend
hbridge1=0
timerend=timemotordown,
timerstart:=0
lenslampposition=0,
lenslampsorter=0,
engineconveyer=0,
enginefeeder=0,
hbridge0=1
lenslampposition=0,
lenslampsorter=0,
engineconveyer=0,
enginefeeder=0,
hbridge0=1,
hbridge1=0
engineconveyer=0,
enginefeeder=0,
hbridge0=1,
lenslampposition=0,
lenslampsorter=0
engineconveyer=0,
enginefeeder=0,
hbridge0=1,
hbridge1=0,
lenslampposition=0,
lenslampsorter=0
push_sensor
hbridge0>=1
push_button!
startstop_Sensor
x>=1
startstop_button!
x:=0
positionSensor
lenslampposition>=1
lightdetector_position!
colorDetector
lenslampsorter>=1
lightdetector_color!
abort_Button
x>=1
abort_button!
x:=0