**DOCUMENTATIE – EPD (HER)**



**Naam :** Brian van der Ven (669478)

**Klas :** ITN-ISE-A-s

**Datum :** Periode 2 - 2022/2023

16-12-2022

**Inhoud**

[Inleiding 2](#_Toc122087929)

[1 Functioneel Ontwerp 3](#_Toc122087930)

[1.1 Lijst van eisen 3](#_Toc122087931)

[2 Technisch Ontwerp 5](#_Toc122087932)

[2.1 State diagram 6](#_Toc122087933)

[2.2 Tabbladen diagram 7](#_Toc122087934)

[2.3 Tabbladen beschrijving 8](#_Toc122087935)

[2.4 Opstelling 15](#_Toc122087936)

[3 Self-assessment 16](#_Toc122087937)

[Bibliografie 17](#_Toc122087938)

[Bijlagen 18](#_Toc122087939)

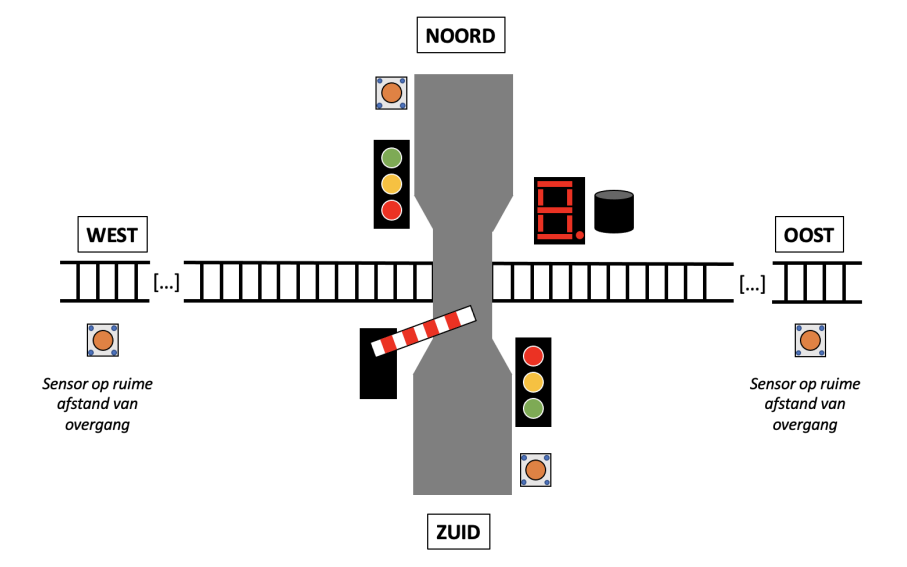
# Inleiding

In dit rapport wordt de casus van de eindopdracht van EPD (Embedded Program Development) behandeld.

In deze casus wordt er een spoorwegovergang gesimuleerd. Deze spoorwegovergang bestaat uit 2 onderdelen: een weg en een spoor. Het verkeer wordt gesimuleerd door 2 drukknoppen op de uiteinden van deze onderdelen. Verder zijn er 2 stoplichten, die elk bestaan uit 3 LEDs en een spoorboom die gesimuleerd wordt door een servomotortje. Deze regelen samen het weg verkeer.

Ook wordt er een display gebruikt om af te tellen wanneer de auto’s mogen gaan rijden, er is een potmeter die de snelheid van de aankomende trein aangeeft en er is een buzzer die tikt in bepaalde situaties.

*Voor de volledige casus zie de bijlage Casus voor EPD.*



Figuur : Spoorwegovergang (onbekend, EPD Opdracht Spoorwegovergang, 2022)

# 1 Functioneel Ontwerp

In het Functioneel Ontwerp wordt de lijst van eisen toegelicht.

Deze lijst bestaat uit de eisen die zijn verzameld uit de gegeven casus. Hierbij krijgen de eisen een prioritering aan de hand van de MoSCoW methode. Verder worden deze genummerd. Wanneer de opstelling is gerealiseerd worden alle eisen getest. Dit testresultaat wordt ook opgenomen in de lijst.

## 1.1 Lijst van eisen

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr** | **Omschrijving** | **Prioriteit** | **Testresultaat** |
| 1 | Verkeerslicht wordt gesimuleerd door 3 LEDs. | M | **GESLAAGD** |
| 2 | Trein wordt gesimuleerd door knop OOST en WEST. | M | **GESLAAGD** |
| 3 | Wegverkeer (Auto/Voetganger) wordt gesimuleerd door knop NOORD en ZUID. | M | **GESLAAGD** |
| 4 | Spoorboom wordt gesimuleerd door servo-motor. | M | **GESLAAGD** |
| 5 | Trein snelheid wordt gesimuleerd door potmeter | M | **GESLAAGD** |
| 6.1 | Bij wegverkeer, verkeerslicht op GROEN voor 2s (wanneer er geen andere wegverkeer GROEN heeft). | M | **GESLAAGD** |
| 6.2 | Na 6.1, Verkeerslicht 1s GEEL en dan naar ROOD. | M | **GESLAAGD** |
| 6.3 | Na 6.2, 1s bufferperiode (ontruimingstijd), beide verkeerslichten ROOD voordat volgende acties worden uitgevoerd. | M | **GESLAAGD** |
| 7 | Bij wegverkeer gaat alleen het bijhorende verkeerslicht op GROEN. | M | **GESLAAGD** |
| 8 | **Onthoudt aankomend wegverkeer** wanneer er al GROEN licht is. Uitvoeren nadat voorgaande actie is afgelopen (zie 6). Ook wanneer hier een trein tussendoor komt. | M | **GESLAAGD** |
| 9.1 | **Onthoudt aankomend wegverkeer** tijdens ontruimingstijd. Uitvoeren nadat voorgaande actie is afgelopen. | M | **GESLAAGD** |
| 9.2 | **Onthoudt aankomende trein** tijdens ontruimingstijd.  Uitvoeren nadat voorgaande actie is afgelopen.  **Voorrang** op eventuele voorgaande onthouden wegverkeer. | M | **GESLAAGD** |
| 10.0 | Verwacht een trein, GROEN licht (zie 6.1), licht meteen\* op GEEL voor 1s en daarna op ROOD (zie 6.2). | M | **GESLAAGD** |
| 10.1 | (evt. na 10.0) Verwacht een trein, verkeerslichten op ROOD en GEEL knippert met 1hz. | M | **GESLAAGD** |
| 10.2 | Tijdens 10.1, sluiten spoorboom. (interval = 50ms, target = 0 graden). | M | **GESLAAGD** |
| 10.3 | Tijdens 10.1, snel tikken buzzer. (interval = 250ms, frequentie = 2000hz) | M | **GESLAAGD** |
| 11.1 | Na 10, Passeren trein, gesloten spoorboom (0 graden), buzzer uit. | M | **GESLAAGD** |
| 11.2 | Tijdens 11.1, licht is ROOD en GEEL knippert met 1hz. | M | **GESLAAGD** |
| 12.1 | Gepasseerde trein, tegengestelde knop (OOST-WEST, WEST-OOST) ingedrukt, openen spoorboom. (interval = 50ms, target = 90 graden). | M | **GESLAAGD** |
| 12.2 | Tijdens 12.1, licht is ROOD en GEEL knippert met 1hz. | M | **GESLAAGD** |
| 12.3 | Tijdens 12.1, snel tikken buzzer. (interval = 250ms, frequentie = 2000hz) | M | **GESLAAGD** |
| 13.1 | Na 12, display aftellen van 5 naar 0. | M | **GESLAAGD** |
| 13.2 | Tijdens 13.1, 3 tikken buzzer (interval = 100ms, frequentie = 2000hz) dan pauze (400ms). | M | **GESLAAGD** |
| 13.3 | Tijdens 13.1, licht is ROOD en GEEL knippert met 1hz. | M | **GESLAAGD** |
| 14.1 | Na 13, display = 0, buzzer uit. | M | **GESLAAGD** |
| 14.2 | Tijdens 14.1, lichten ROOD of bijhorend GROEN als er wegverkeer kwam. GEEL knippert niet. | M | **GESLAAGD** |
| 15 | **Onthoudt aankomend wegverkeer** tijdens trein. Uitvoeren nadat voorgaande actie is afgelopen. | M | **GESLAAGD** |
| 16 | Bij dezelfde knop tijdens actie mag knop worden genegeerd. | S | **GESLAAGD** |
| 17 | Trein cyclus wordt volledig uitgevoerd zonder onderbrekingen. | M | **GESLAAGD** |
| 18 | Tegengestelde knop kan tijdens sluiten spoorboom al worden geactiveerd. | S | **GESLAAGD** |
| 19 | Startsituatie : lichten ROOD, spoorboom OPEN. | M | **GESLAAGD** |
| 20 | 2 keer klikken op knop (NOORD of ZUID) geeft meer tijd (4s) voor verkeerslichtenlichten. | S\* | **GESLAAGD** |
| 21 | Potmeter verhoogt/verlaagt tijd tussen klikken knop en wachten op trein of veranderen stoplicht naar GEEL\*. (Lage waarde = veel tijd -> [0-1023]=[2000ms-500ms]) | M | **GESLAAGD** |
| 22.1 | Waarde onder 500 lux van lichtsensor, periode GEEL langer | W |  |
| 22.2 | Waarde onder 500 lux van lichtsensor, spoorboom sneller OPEN en DICHT (interval = 40ms). | W |  |
| 23.1 | Testknop, lichten een voor een 1s aan en hierna uit. | W |  |
| 23.2 | Na 23.1, spoorboom DICHT (0 graden) hierna OPEN (90 graden). | W |  |
| 23.3 | Na 23.2, buzzer toon (2000hz) voor 1s. | W |  |
| 24.1 | Aantal keer openen spoorboom wordt opgeslagen in een teller die steeds optelt met 1. | C | **GESLAAGD** |
| 24.2 | Aantal keer openen spoorboom print teller in seriële poort. | C | **GESLAAGD** |
| 24.3 | Wanneer “r” in seriële poort teller = 0. | W |  |
| 24.4 | Spoorboomteller wordt opgeslagen in EEPROM (max 255). | W |  |
| 24.5 | Spoorboomteller kan waarde groter dan 255 opslaan. | W |  |
| 24.6 | Spoorboomteller groter dan instelbare waarde (5 keer) laat een ledje branden. | W |  |
| 25 | Bij OPEN spoorboom (90 graden) een led die knippert met 1hz aan de OOST en WEST kant. | W |  |
| 26 | Verkeerslicht GEEL, display aftellen tot ROOD is bereikt. | W |  |
| 27.1 | Tijdens trein (10,11,12,13,14), Volgorde van wegverkeer wordt onthouden. | W |  |
| 27.2 | Na 27.1, wegverkeer wordt in opgeslagen volgorde uitgevoerd. | W |  |

\*20. Must have is veranderd naar Should have.

\*10.0 en 21. De eis in 10.0 wordt door eis 21 overschreven dus wordt ‘meteen’ gelijk aan de instelbare interval.

# 2 Technisch Ontwerp

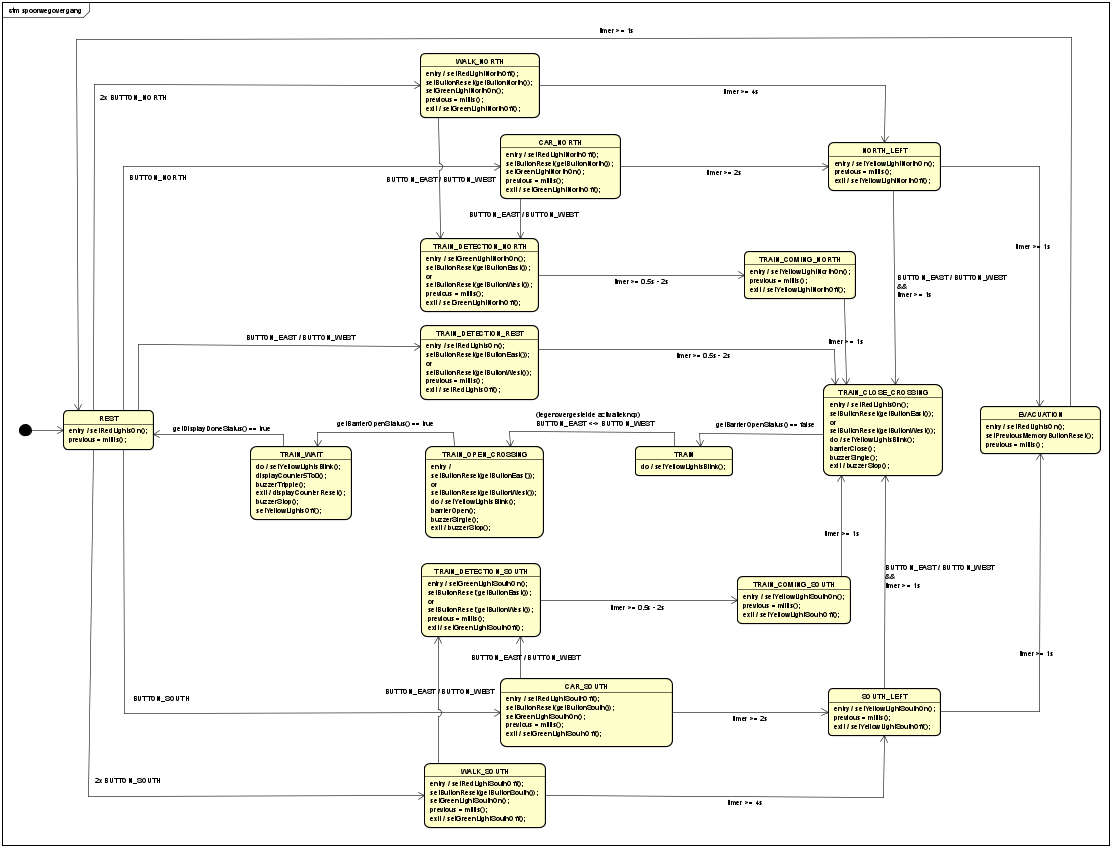
In het Technisch Ontwerp worden de technische details van de casus uitgewerkt.

Er worden verschillende diagrammen gemaakt. Een hiervan is het state diagram. Het state diagram bevat de verschillende toestanden waarin de opstelling zich kan bevinden met de bijhorende functies die in die toestand worden aangeroepen. Ook de toestand overgangen zijn hierin te vinden.

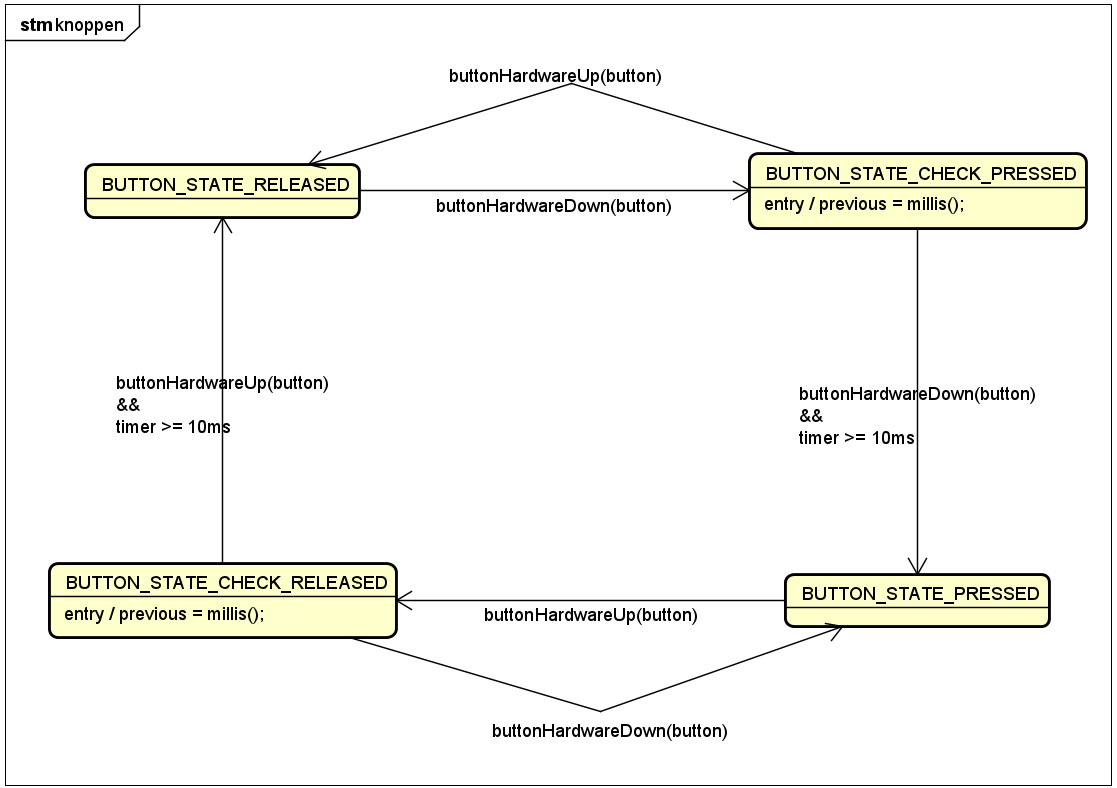
Verder wordt er ook een tabbladen diagram gemaakt. In het tabbladen diagram staan de tabbladen die gemaakt zijn bij de realisatie van de opstelling. Hierbij staan de functies en variabelen per tabblad genoemd en is de relatie van de tabbladen onderling weergegeven. Het tabbladen diagram wordt ondersteund door een beschrijving waarin een diepere uitleg te vinden is over de inhoud van de tabbladen.

Als laatst is er een model te zien hoe de opstelling er uit zou kunnen zien.

## 2.1 State diagram

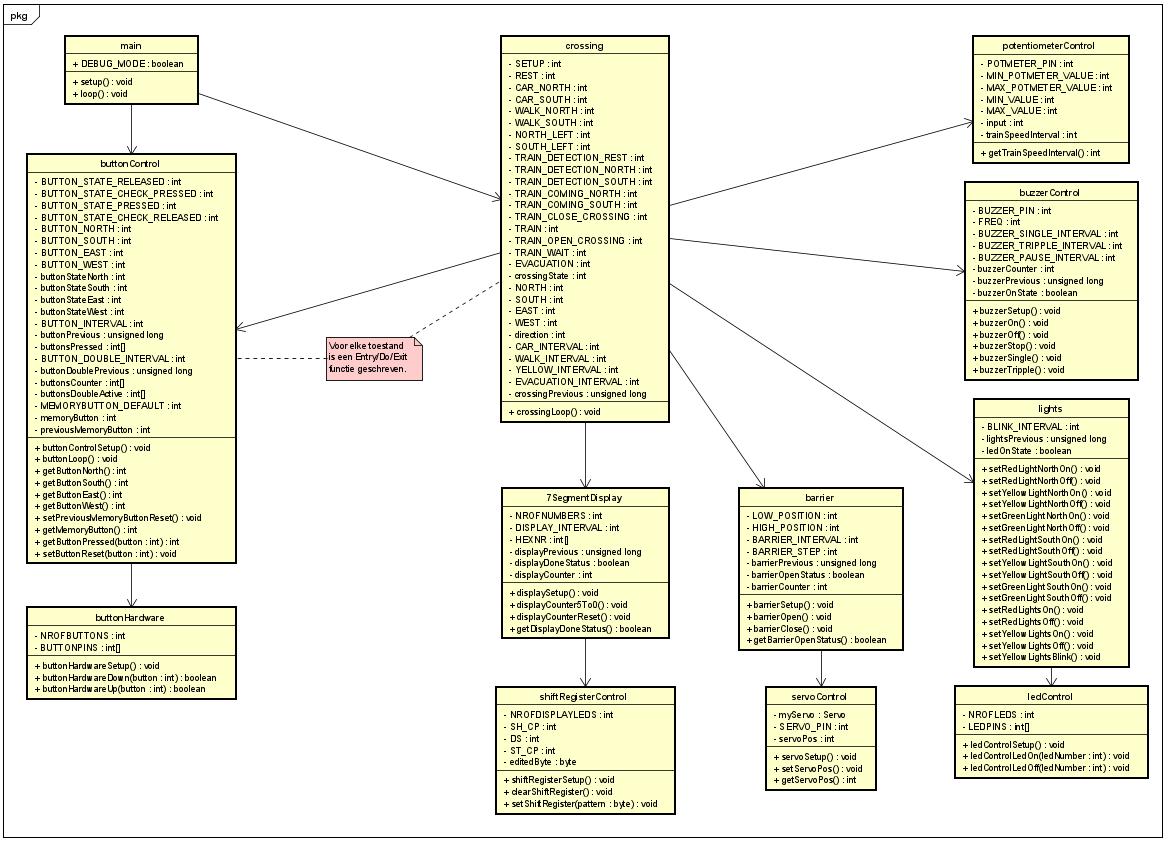


Figuur 2a : State Diagram - Spoorwegovergang



Figuur 2b : State Diagram - Buttons

## 2.2 Tabbladen diagram



Figuur 3 : Tabbladen Diagram

## 2.3 Tabbladen beschrijving

**main**

|  |  |  |
| --- | --- | --- |
| **Methode naam** | **Return waarde** | **Beschrijving** |
| setup() | void | Methode waarin alle setup methoden worden aangeroepen om de opstelling klaar te zetten voor gebruik. |
| loop() | void | Methode waarin methoden worden aangeroepen die altijd moeten worden uitgevoerd. |

|  |  |  |
| --- | --- | --- |
| **Variabele naam** | **Datatype** | **Beschrijving** |
| DEBUG\_MODE | bool | Constante, laat extra tekst printen door her gehele programma. |

**7SegmentDisplay**

|  |  |  |
| --- | --- | --- |
| **Methode naam** | **Return waarde** | **Beschrijving** |
| displaySetup() | void | Setup voor het display. |
| displayCounter5To0() | void | Laat display aftellen van 5 naar 0. |
| displayCounterReset() | void | Zet alle waardes terug en display uit. |
| getDisplayDoneStatus() | bool | Geeft variabele displayDoneStatus. |

|  |  |  |
| --- | --- | --- |
| **Variabele naam** | **Datatype** | **Beschrijving** |
| NROFNUMBERS | int | Constante, geeft het aantal cijfers van het display aan. |
| DISPLAY\_INTERVAL | int | Constante, geeft de tijd dat elk cijfer te zien is aan. |
| HEXNR | int [ ] | Constante array, geeft de cijfers van het display aan. |
| displayPrevious | unsigned long | Timer variabele. |
| displayDoneStatus | bool | Geeft de status van het display aan. |
| displayCounter | int | Variabele waarmee door de gegeven cijfers heen wordt gelopen. |

**barrier**

|  |  |  |
| --- | --- | --- |
| **Methode naam** | **Return waarde** | **Beschrijving** |
| barrierSetup() | void | Setup voor de servo. |
| barrierOpen() | void | Laat de servo met stappen lopen naar HIGH\_POSITION. |
| barrierClose() | void | Laat de servo met stappen lopen naar LOW\_POSITION. |
| getBarrierOpenStatus() | bool | Geeft variabele barrierOpenStatus. |

|  |  |  |
| --- | --- | --- |
| **Variabele naam** | **Datatype** | **Beschrijving** |
| LOW\_POSITION | int | Constante, geeft het laagst aantal graden aan. |
| HIGH\_POSITION | int | Constante, geeft het hoogst aantal graden aan. |
| BARRIER\_INTERVAL | int | Constante, geeft de tijd aan waarna de servo een stap mag doen. |
| BARRIER\_STEP | int | Constante, geeft de stapgrootte van de servo aan. |
| barrierPrevious | unsigned long | Timer variabele. |
| barrierOpenStatus | bool | Geeft de status van de slagboom aan. |
| barrierCounter | int | Geeft aan hoe vaak de slagboom is geopend. |

**buttonControl**

|  |  |  |
| --- | --- | --- |
| **Methode naam** | **Return waarde** | **Beschrijving** |
| buttonControlSetup() | void | Setup voor de knoppen (status). |
| buttonLoop() | void | Statemachine van de knoppen. Loopt door alle knoppen en houd hier de status van bij. Hierbij wordt het aantal keer dat wordt gedrukt bijgehouden. |
| getButtonNorth() | int | Geeft variabele BUTTON\_NORTH. |
| getButtonSouth() | int | Geeft variabele BUTTON\_SOUTH. |
| getButtonEast() | int | Geeft variabele BUTTON\_EAST. |
| getButtonWest() | int | Geeft variabele BUTTON\_WEST. |
| setPreviousMemoryButtonReset() | void | Zet previousMemoryButton op zijn standaard waarde. |
| getMemoryButton() | int | Geeft variabele memoryButton. |
| getButtonPressed  (int button) | int | Geeft variabele buttonsPressed van een gegeven knop. |
| setButtonReset  (int button) | void | Zet buttonPressed van een gegeven knop naar 0 en zet previousMemoryButton naar de meegegeven knop. Ook wordt memoryButton gezet op zijn standaard waarde. |

Voor elke toestand is een Entry/Do/Exit functie geschreven.

|  |  |  |
| --- | --- | --- |
| **Variabele naam** | **Datatype** | **Beschrijving** |
| BUTTON\_STATE\_RELEASED | int | Constante, geeft de toestand van een knop aan. |
| BUTTON\_STATE\_CHECK\_PRESSED | int | Constante, geeft de toestand van een knop aan. |
| BUTTON\_STATE\_PRESSED | int | Constante, geeft de toestand van een knop aan. |
| BUTTON\_STATE\_CHECK\_RELEASED | int | Constante, geeft de toestand van een knop aan. |
| BUTTON\_NORTH | int | Constante, geeft de plek binnen een array van een knop aan. |
| BUTTON\_SOUTH | int | Constante, geeft de plek binnen een array van een knop aan. |
| BUTTON\_EAST | int | Constante, geeft de plek binnen een array van een knop aan. |
| BUTTON\_WEST | int | Constante, geeft de plek binnen een array van een knop aan. |
| buttonStateNorth | int | Status van een knop (NORTH). |
| buttonStateSouth | int | Status van een knop (SOUTH). |
| buttonStateEast | int | Status van een knop (EAST). |
| buttonStateWest | int | Status van een knop (WEST). |
| BUTTON\_INTERVAL | int | Constante, geeft de dender tijd van een knop aan. |
| buttonPrevious | unsigned long | Timer variabele. |
| buttonsPressed | int [ ] | Array waarin voor elke knop wordt opgeslagen hoe vaak hierop is gedrukt. |
| BUTTON\_DOUBLE\_INTERVAL | int | Constante, geeft de tijd aan waartussen er wordt gekeken of er dubbel wordt gedrukt. |
| buttonDoublePrevious | unsigned long | Timer variabele. |
| buttonsCounter | int [ ] | Array waarin voor knoppen wordt geteld hoe vaak hierop is gedrukt. |
| buttonsDoubleActive | int [ ] | Array waarin voor knoppen wordt bijgehouden of de tijd al is ingegaan om dubbel te drukken. |
| MEMORYBUTTON\_DEFAULT | int | Constante, standaard geheugen waarde. |
| memoryButton | int | Knop waarop als eerst is gedrukt. |
| previousMemoryButton | int | Knop waarop is gedrukt die wordt onthouden zolang de bijhorende overgangstoestand actief is. |

**buttonHardware**

|  |  |  |
| --- | --- | --- |
| **Methode naam** | **Return waarde** | **Beschrijving** |
| buttonHardwareSetup() | void | Setup voor de knoppen. |
| buttonHardwareDown  (int button) | bool | Leest uit of een knop is ingedrukt. |
| buttonHardwareUp  (int button) | bool | Leest uit of een knop niet is ingedrukt. |

|  |  |  |
| --- | --- | --- |
| **Variabele naam** | **Datatype** | **Beschrijving** |
| NROFBUTTONS | int | Constante, geeft het aantal knoppen aan. |
| BUTTONPINS | int [ ] | Constante array, geeft de pinnen van de knoppen aan. |

**buzzerControl**

|  |  |  |
| --- | --- | --- |
| **Methode naam** | **Return waarde** | **Beschrijving** |
| buzzerSetup() | void | Setup voor de buzzer. |
| buzzerOn() | void | Zet de buzzer aan. |
| buzzerOff() | void | Zet de buzzer uit. |
| buzzerStop() | void | Zet de buzzer uit en reset buzzerCounter. |
| buzzerSingle() | void | Laat de buzzer met een enkele toon aan en uit gaan. |
| buzzerTripple() | void | Laat de buzzer drie tonen geven en dan een korte pauze. |

|  |  |  |
| --- | --- | --- |
| **Variabele naam** | **Datatype** | **Beschrijving** |
| BUZZER\_PIN | int | Constante, pin van de buzzer. |
| FREQ | int | Constante, frequency van de buzzer. |
| BUZZER\_SINGLE\_INTERVAL | int | Constante, tijd van een enkele toon. |
| BUZZER\_TRIPPLE\_INTERVAL | int | Constante, tijd van drie snelle tonen. |
| BUZZER\_PAUSE\_INTERVAL | int | Constante, pauze tijd na drie tonen. |
| buzzerCounter | int | Teller hoe vaak de buzzer al is afgegaan. |
| buzzerPrevious | unsigned long | Timer variabele. |
| buzzerOnState | bool | Geeft aan of de buzzer aan staat. |

**crossing**

|  |  |  |
| --- | --- | --- |
| **Methode naam** | **Return waarde** | **Beschrijving** |
| crossingLoop() | void | Statemachine van de spoorwegovergang. |

Voor elke toestand is een Entry/Do/Exit functie geschreven.

|  |  |  |
| --- | --- | --- |
| **Variabele naam** | **Data**  **type** | **Beschrijving** |
| SETUP | int | Constante, toestand van de spoorwegovergang. |
| REST | int | Constante, toestand van de spoorwegovergang. |
| CAR\_NORTH | int | Constante, toestand van de spoorwegovergang. |
| CAR\_SOUTH | int | Constante, toestand van de spoorwegovergang. |
| WALK\_NORTH | int | Constante, toestand van de spoorwegovergang. |
| WALK\_SOUTH | int | Constante, toestand van de spoorwegovergang. |
| NORTH\_LEFT | int | Constante, toestand van de spoorwegovergang. |
| SOUTH\_LEFT | int | Constante, toestand van de spoorwegovergang. |
| TRAIN\_DETECTION\_REST | int | Constante, toestand van de spoorwegovergang. |
| TRAIN\_DETECTION\_NORTH | int | Constante, toestand van de spoorwegovergang. |
| TRAIN\_DETECTION\_SOUTH | int | Constante, toestand van de spoorwegovergang. |
| TRAIN\_COMING\_NORTH | int | Constante, toestand van de spoorwegovergang. |
| TRAIN\_COMING\_SOUTH | int | Constante, toestand van de spoorwegovergang. |
| TRAIN\_CLOSE\_CROSSING | int | Constante, toestand van de spoorwegovergang. |
| TRAIN | int | Constante, toestand van de spoorwegovergang. |
| TRAIN\_OPEN\_CROSSING | int | Constante, toestand van de spoorwegovergang. |
| TRAIN\_WAIT | int | Constante, toestand van de spoorwegovergang. |
| EVACUATION | int | Constante, toestand van de spoorwegovergang. |
| crossingState | int | Status van de spoorwegovergang. |
| NORTH | int | Constante, richting van het verkeer. |
| SOUTH | int | Constante, richting van het verkeer. |
| EAST | int | Constante, richting van het verkeer. |
| WEST | int | Constante, richting van het verkeer. |
| direction | int | Richting van het verkeer. |
| CAR\_INTERVAL | int | Constante, tijd dat het groen is voor een auto. |
| WALK\_INTERVAL | int | Constante, tijd dat het groen is voor een voetganger. |
| YELLOW\_INTERVAL | int | Constante, tijd dat het geel is. |
| EVACUATION\_INTERVAL | int | Constante, tijd van de ontruimingstijd. |
| crossingPrevious | unsigned long | Timer variabele. |

**ledControl**

|  |  |  |
| --- | --- | --- |
| **Methode naam** | **Return waarde** | **Beschrijving** |
| ledControlSetup() | void | Setup voor de LEDs. |
| ledControlLedOn  (int ledNumber) | void | Zet een gegeven LED aan. |
| ledControlLedOff  (int ledNumber) | void | Zet een gegeven LED uit. |

|  |  |  |
| --- | --- | --- |
| **Variabele naam** | **Datatype** | **Beschrijving** |
| NROFLEDS | int | Constante, geeft het aantal LEDs aan. |
| LEDPINS | int [ ] | Constante array, geeft de pinnen van de LEDs aan. |

**lights**

|  |  |  |
| --- | --- | --- |
| **Methode naam** | **Return waarde** | **Beschrijving** |
| setRedLightNorthOn() | void | Zet verkeerslicht NOORD ROOD aan. |
| setRedLightNorthOff() | void | Zet verkeerslicht NOORD ROOD uit. |
| setYellowLightNorthOn() | void | Zet verkeerslicht NOORD GEEL aan. |
| setYellowLightNorthOff() | void | Zet verkeerslicht NOORD GEEL uit. |
| setGreenLightNorthOn() | void | Zet verkeerslicht NOORD GROEN aan. |
| setGreenLightNorthOff() | void | Zet verkeerslicht NOORD GROEN uit. |
| setRedLightSouthOn() | void | Zet verkeerslicht ZUID ROOD aan. |
| setRedLightSouthOff() | void | Zet verkeerslicht ZUID ROOD uit. |
| setYellowLightSouthOn() | void | Zet verkeerslicht ZUID GEEL aan. |
| setYellowLightSouthOff() | void | Zet verkeerslicht ZUID GEEL uit. |
| setGreenLightSouthOn() | void | Zet verkeerslicht ZUID GROEN aan. |
| setGreenLightSouthOff() | void | Zet verkeerslicht ZUID GROEN uit. |
| setRedLightsOn() | void | Zet verkeerslichten NOORD en ZUID ROOD aan. |
| setRedLightsOff() | void | Zet verkeerslichten NOORD en ZUID ROOD uit. |
| setYellowLightsOn() | void | Zet verkeerslichten NOORD en ZUID GEEL aan. |
| setYellowLightsOff() | void | Zet verkeerslichten NOORD en ZUID GEEL uit. |
| setYellowLightsBlink() | void | Laat verkeerslichten NOORD en ZUID GEEL knipperen. |

|  |  |  |
| --- | --- | --- |
| **Variabele naam** | **Datatype** | **Beschrijving** |
| BLINK\_INTERVAL | int | Constante, tijd dat een LED aan en uit is tijdens het knipperen. |
| lightsPrevious | unsigned long | Timer variabele. |
| ledOnState | bool | Geeft de status van een knipperende led aan. |

**potentiometerControl**

|  |  |  |
| --- | --- | --- |
| **Methode naam** | **Return waarde** | **Beschrijving** |
| getTrainSpeedInterval() | int | Geeft variabele trainSpeedInterval. |

|  |  |  |
| --- | --- | --- |
| **Variabele naam** | **Datatype** | **Beschrijving** |
| POTMETER\_PIN | int | Constante, pin van de potmeter. |
| MIN\_POTMETER\_VALUE | int | Constante, minimale waarde van de potmeter. |
| MAX\_POTMETER\_VALUE | int | Constante, maximale waarde van de potmeter. |
| MIN\_VALUE | int | Constante, minimale waarde voor variabele trainSpeedInterval. |
| MAX\_VALUE | int | Constante, maximale waarde voor variabele trainSpeedInterval. |
| input | int | Uitgelezen waarde van de potmeter tussen MIN\_POTMETER\_VALUE en MAX\_POTMETER\_VALUE. |
| trainSpeedInterval | int | Getransformeerde waarde van de input tussen MIN\_VALUE en MAX\_VALUE. |

**servoControl**

|  |  |  |
| --- | --- | --- |
| **Methode naam** | **Return waarde** | **Beschrijving** |
| servoSetup() | void | Setup voor de servo. |
| setServoPos  (int pos) | void | Zet de servo op de gegeven positie. |
| getServoPos() | int | Geeft de variabele servoPos. |

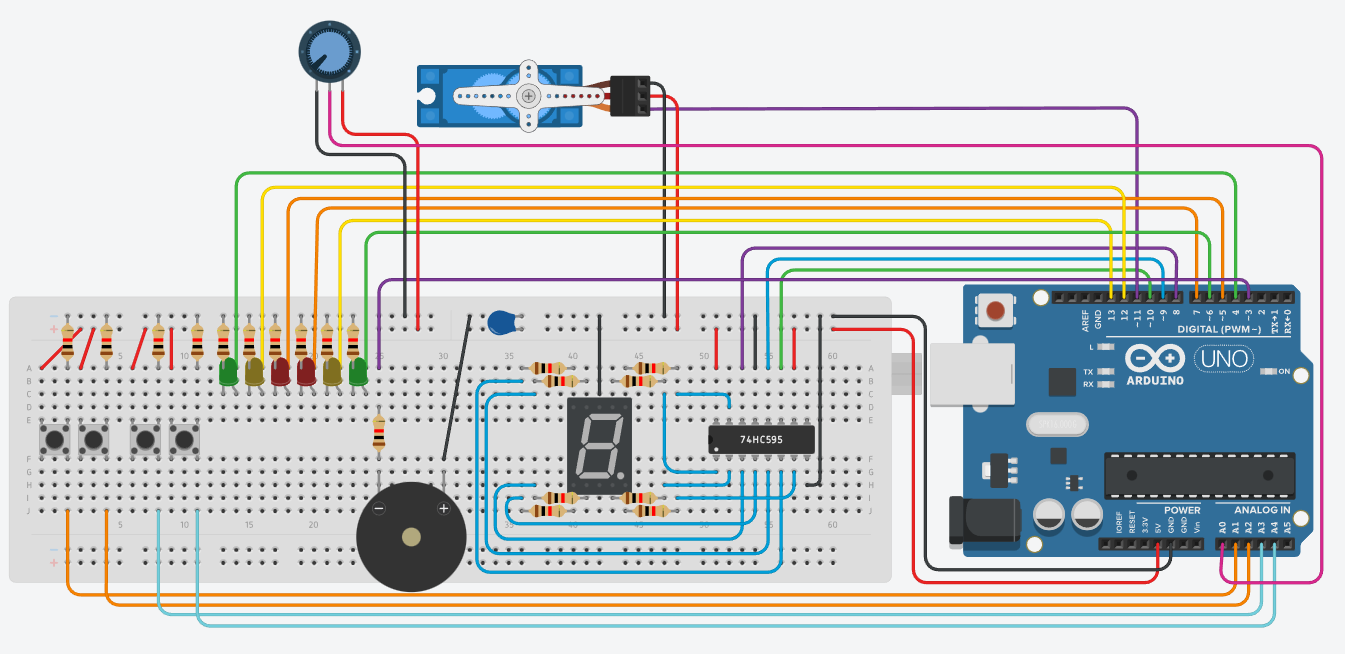
|  |  |  |
| --- | --- | --- |
| **Variabele naam** | **Datatype** | **Beschrijving** |
| myServo | Servo | Servo vanuit library met  #include <Servo.h> |
| SERVO\_PIN | int | Constante, pin van de servo. |
| servoPos | int | Positie van de servo. |

**shiftRegisterControl**

|  |  |  |
| --- | --- | --- |
| **Methode naam** | **Return waarde** | **Beschrijving** |
| shiftRegisterSetup() | void | Setup voor het shiftregister. |
| clearShiftRegister() | void | Leegt het shiftregister. |
| setShiftRegister  (byte pattern) | void | Vult het shiftregister met het gegeven patroon. |

|  |  |  |
| --- | --- | --- |
| **Variabele naam** | **Datatype** | **Beschrijving** |
| NROFDISPLAYLEDS | int | Constante, aantal LEDs van het display. |
| SH\_CP | int | Constante, pin van verzendingsdraad. |
| DS | int | Constante, pin van de datadraad. |
| ST\_CP | int | Constante, pin van de klokdraad. |
| editedByte | byte | Byte waar patroon in wordt bewerkt.  waarin de cijfers worden bit voor bit worden uitgelezen |

## 2.4 Opstelling



Figuur 4: Tinkercad model

# 3 Self-assessment

Het self-assessment wordt gebruikt om vooraf te controleren of de uitwerking aan de belangrijkste beoordelingscriteria voldoet. Het self-assessment is gehaald uit de bron (onbekend, EPD Self-assessment eindopdracht, 2022)

*Voor het ingevulde self-assessment zie de bijlage Self-assessment.*

# Bibliografie

onbekend. (2022, april 22). EPD Opdracht Spoorwegovergang.

onbekend. (2022, februari 3). EPD Self-assessment eindopdracht.

# Bijlagen

|  |  |  |
| --- | --- | --- |
| Bijlage 1 : Casus voor EPD | *EPD Opdracht Spoorwegovergang* | **PDF** |
| Bijlage 2a : State Diagram -  Spoorwegovergang | *StateDiagramSpoorwegovergang (spoorwegovergang)* | **astah-UML** |
| Bijlage 2b : State Diagram -  Buttons | *StateDiagramSpoorwegovergang*  *(buttons)* | **astah-UML** |
| Bijlage 3 : Tabbladen Diagram | *TabbladenDiagram*  *(Tabbladen diagram)* | **astah-UML** |
| Bijlage 4 : Tinkercad Model | *Tinkercad\_model* | **PNG** |
| Bijlage 5 : Self-assessment | *EPD Self-assessment eindopdracht* | **docx** |