UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Faculty of Engineering, Built Environment and Information Technology

Fakulteit Ingenieurswese, Bou-omgewing en Inligtingtegnologie / Lefapha la Boetšenere, Tikologo ya Kago le Theknolotši ya Tshedimošo

Project Guide

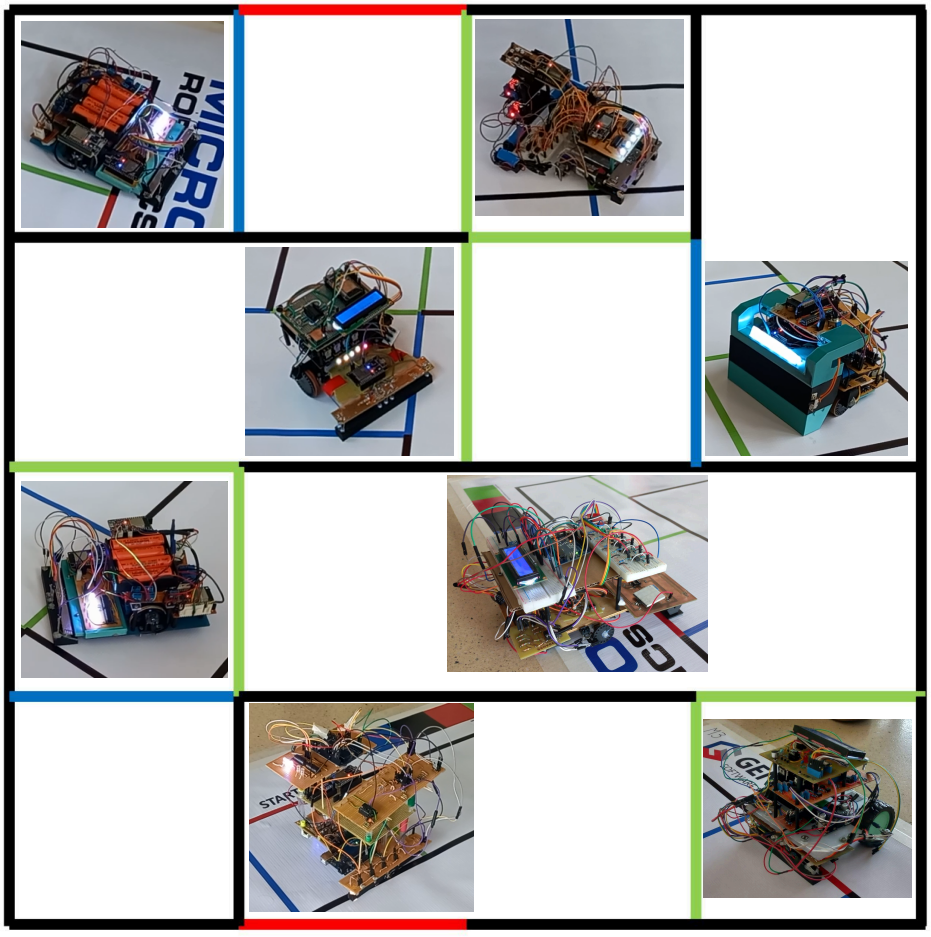
to an

A-Maze-Eng MARV

ELO320/ERD320/EWE320

Version 2025.06.09

Compiled by: Dr W. Badenhorst



INDEX

List of Abbreviations

EoC - End-of-Calibration
HMI - Human Machine Interface
LCD - Liquid Crystal Display
LED - Light Emitting Diode
MARV - Microcontroller-based Autonomous Robotic Vehicle
MDPS - Motor Driver and Power Supply
PEC - Physical and Environmental Considerations
SCS - System Communication Standard
SNC - State-and-Navigation Control
SS - Sensor Subsystem

1. Introduction

The primary objective of the practical component of this module is for you to apply the systems engineering principles covered in the lectures by designing and implementing an A-Maze-Eng MARV within a multi-disciplinary project team. You are required to design and construct a Microcontroller-based Autonomous Robotic Vehicle (MARV) capable of navigating through (not solving) a maze, such as the one on the cover page, using colour sensors, wheels attached to DC motors, a state machine and a predetermined set of navigation rules that will ensure your MARV does not get lost or stuck in the maze. Control of the MARV is achieved using microcontrollers of **any type** and programmed using **any language**.

The MARV will consist of three subsystems each targeted at a specific engineering discipline:

- a **Sensor (SS)** subsystem intended primarily for the electronic engineering student,
- a **State-and-Navigation Control (SNC)** subsystem intended primarily for the computer engineering student,
- a **Motor Driver and Power Supply (MDPS)** subsystem intended primarily for the electrical engineering student in the group.

Each of these subsystems must be able to communicate via serial communication with:

a) each other as an integrated system and
b) the HUB for individual assessment.

The HUB is a Pygame (Python) based emulation of the three subsystems which allows for individual, independent testing and evaluation of the subsystems. The latest version of the HUB program and user manual is available on ClickUP.

This project guide document details the descriptions, architecture, design specifications and protocols required to develop the A-Maze-Eng MARV. These specifications must strictly be adhered to to ensure the correct operation of the MARV and compatibility with the HUB.

- The functional and architectural block diagrams are given in Section 2 and Section 3 respectively which illustrate the functions and architecture of the system and subsystems and how they interact with one another.
- The software architecture is given in Section 4.
- The system specifications are given in Section 5 and provide the specifications that must be met by the system as a whole to ensure the desired operation.
- Section 6, Section 7, and Section 8 provide descriptions of each of the subsystems and the specifications each subsystem must adhere to.
- Section 9 gives a description of and the specifications for the Navigation Control.
- Section 10 briefly lays out the deliverables and evaluation criteria of the four project phases that will guide you toward delivering an A-Maze-Eng MARV. A separate QTP document describes in detail how the subsystems will be evaluated in Phase 3.
- Appendix A contains the critically important system communication standard (SCS) that describes the communication protocol and the control byte structure that must be used when communicating via serial communication.
- Appendix B gives the physical and environmental considerations (PEC) of the system, including size considerations, units of measure and maze information.
- Appendix C lists and discusses some potholes to be aware of during the implementation of the system and
- Appendix D contains a record of important revisions made in this document.

2. Functional Block Diagram

The functional block diagrams in Figure 1 illustrate the required functionality for the complete MARV system, subdivided into the SS, SNC and MDPS subsystems. For the individual demonstrations in project phases 2 and 3, the student's subsystem will be connected to the HUB that will then emulate the remaining subsystems as illustrated in Figure 1a. In the case of the complete integrated systems for project phase 4, the HUB is excluded as illustrated in Figure 1b.

The **Sensor subsystem** (SS) is responsible for the calibration of all of the sensors in the sensor array (1.1), colour detection which includes detection of the finish line or exit of the maze (1.2) and visualisation of the colour detected by each individual sensor in the array (1.3). Furthermore, when the SS detects a line or "wall", it must calculate the incidence angle of the sensor array relative to line (1.4). The SS must be able to communicate and receive the prescribed information to and from the SNC, MDPS and HUB (1.5).

The **State-and-Navigation Control (SNC) subsystem** must communicate with the SS, MDPS or the HUB (2.5), and is responsible for enabling the system and initiating state transitions (2.1). The SNC must also be able to display critical system diagnostics obtained from the other subsystems (2.2) and facilitate the Human Machine Interface (HMI) through a touch sensor and a pure tone detection (2.3). The SNC must implement a navigation control algorithm (2.5) which enables navigation within the maze by communicating the necessary MARV movements to the MDPS subsystem where the motor control must be implemented (3.2).

The **Motor Driver and Power Supply (MDPS) subsystem** must ensure that the system can operate independently from wall socket power and supply the system with one or more of the prescribed voltage levels (3.1) depending on what your design requires. The MDPS must also implement motor actuation and control to ensure the MARV rotates or moves in the direction and at the speed instructed by the SNC (3.2) which will require sensing the rotation, speed and distance (3.3) from the two motors used to drive and steer the MARV. The MARV rotation setpoints for the MDPS will be calculated in the SNC subsystem and communicated directly from the SNC or via the HUB (3.4). The MDPS must be able to communicate and receive the prescribed information to and from the SNC, SS and HUB.

All subsystem communication must comply with the protocol as stipulated in Appendix A.

As stated earlier, if all three degree programs are represented in a group, the required allocation of subsystems to disciplines is as follows:

1. *Electronic - SS subsystem*
2. *Computer - SNC subsystem*
3. *Electrical - MDPS subsystem*

However, the design task required for each of the three subsystems is suitable to be completed by a student from any discipline. Given the ratio of Computer : Electronic : Electrical students, it is not possible to have one of each discipline in all of the groups, so a computer engineer may be assigned to the MDPS or SS subsystem.

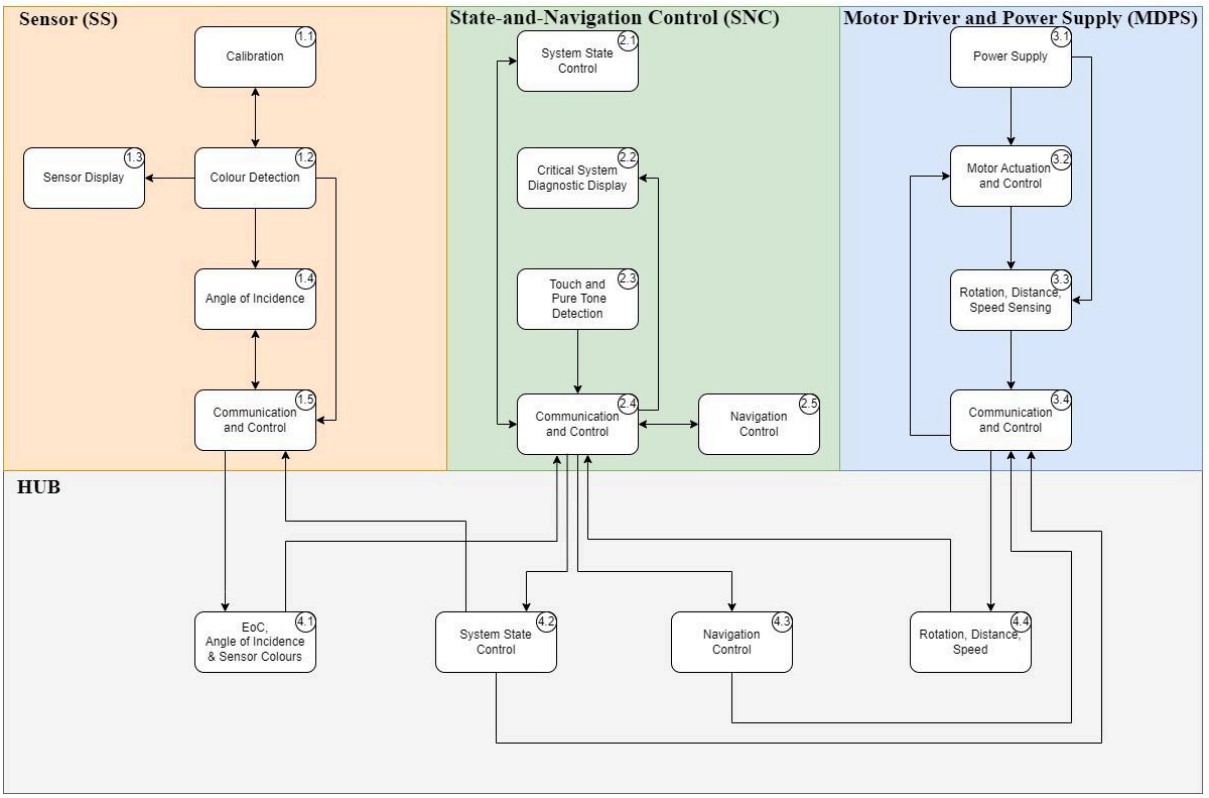


*Figure 1a - Prescribed system functional block diagram for **HUB integrated** system, which may use the HUB to emulate the two of the three subsystems and the environment.*

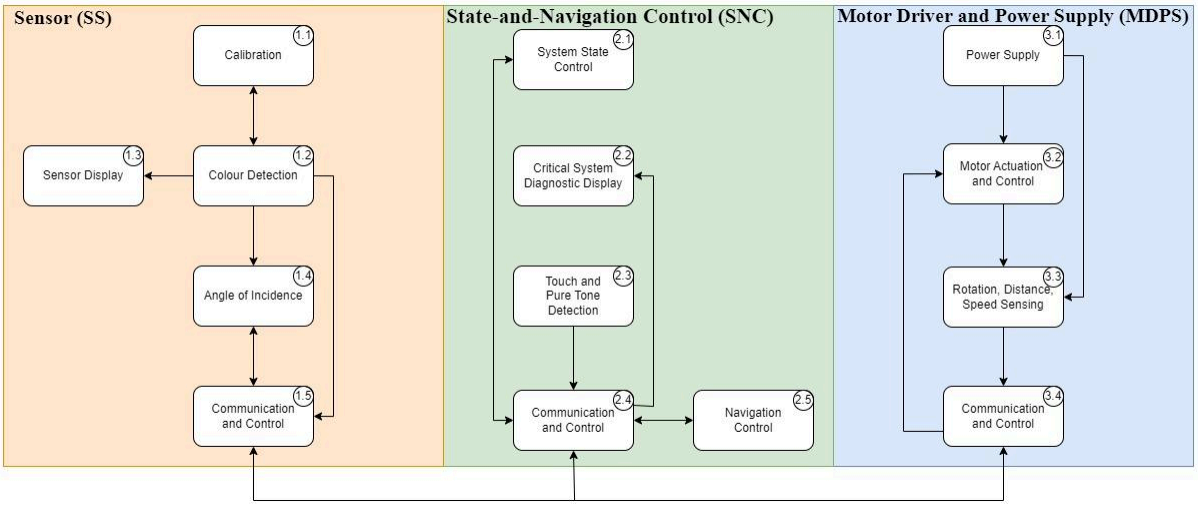


*Figure 1b - Prescribed system functional block diagram for **integrated** systems.*

3. Architectural Block Diagram

The architectural block diagram shown in Figure 2 illustrates the required principal hardware components and subdivision of the entire system into the three main subsystems. Communication between the SS, SNC and MDPS is done via serial UART. **RED** blocks represent off-the-shelf components, whereas **BLUE** blocks represent components that **must** be designed and implemented from first principles. First principles mean that you may **not** use an off-the-shelf ***module*** such as a microphone-filter-amplifier, H-bridge, optical motor shaft encoder (e.g.1, e.g.2) *module* or an addressable RGB strip. You may of course use the discrete components, like the opto-interrupter sensor (optical switch **without** a built in Schmitt Trigger). Click on the links to see examples of what is not allowed. The detailed designs will form part of your final examination report. *Should students wish to implement any off-the-shelf components from first principles, they are allowed to do so.*

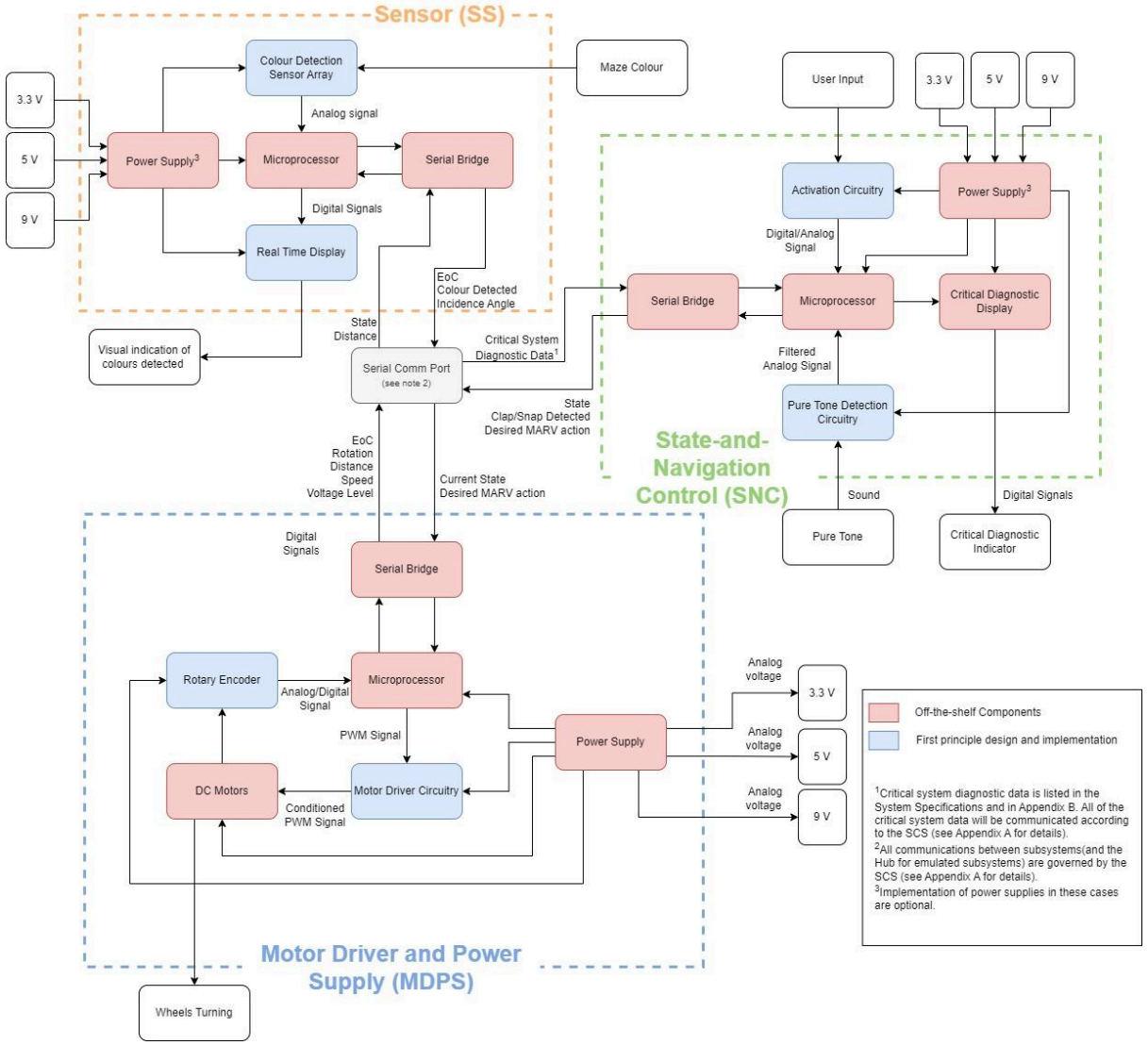


Figure 2 - Prescribed system architectural block diagram. Missing subsystems will be emulated by the HUB.

4. Software Architecture

The state diagram in Figure 3 describes the primary embedded control operating states of the system. Table 1 offers more detail about the expected outcomes for each of the control operating states of the system. A detailed version of the system state diagram, as well as the communication protocol, is provided in Appendix A.

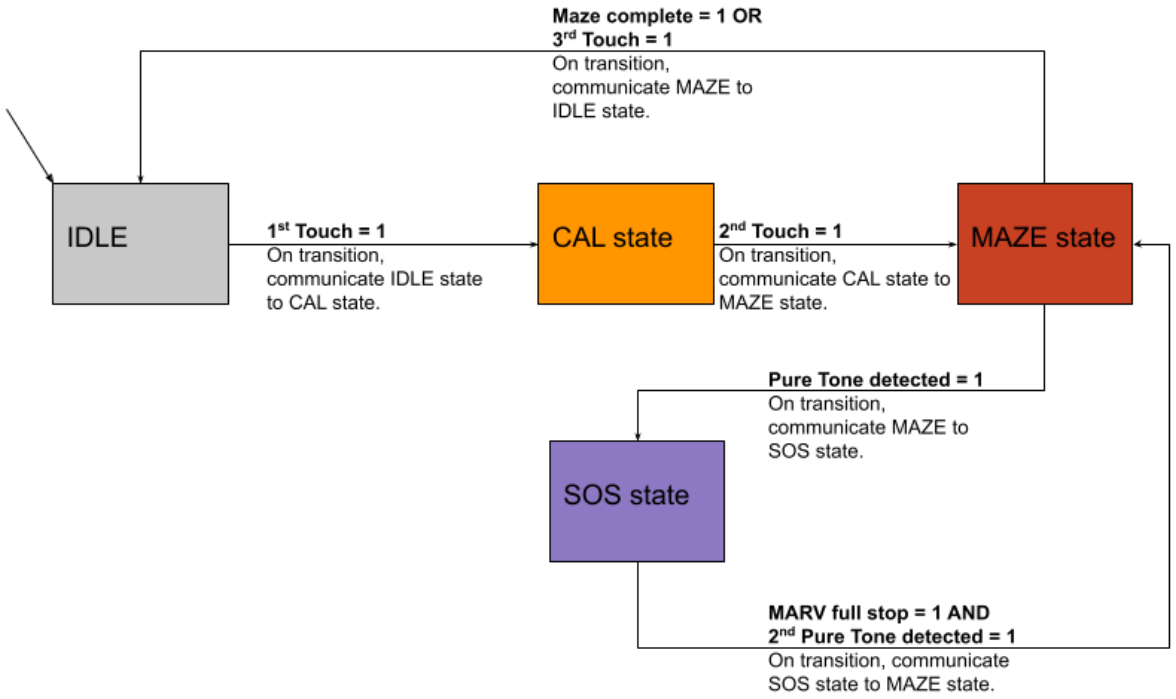


Figure 3 - Prescribed primary system operating states.

At power-up, the MARV has to enter the **IDLE state**. The system then waits in the IDLE state for user input which is the first touch activation meant to initiate the state transition from the IDLE state to the CAL state.

The colour sensor and motor sensor calibration is completed in the **CAL state**. After the calibration has completed, the SNC must display the available system diagnostics (as indicated in Figure 2 and listed in Table B.1 in Appendix B) while awaiting the second touch activation and state transition to the MAZE state.

In the **MAZE state** the subsystems are expected to facilitate the following system-critical functions to implement navigation.

1. The **SS** is responsible for detecting the colours (White, Red, Blue, Green, Black) and communicating the colour seen by each sensor to the SNC. Furthermore, it is responsible for calculating the incidence angle of the sensor array relative to a line when detecting it and communicating the angle to the SNC for navigation decision making.
2. The **SNC** is responsible for implementing the steering or navigation control algorithm using the information obtained from the SS and MDPS. The SNC will calculate the required tangential wheel speeds and MARV actions (e.g. forward, backward, turn) to drive and navigate the MARV, and communicate this to the MDPS. Navigation control is detailed in Section 9. The SNC must also

continuously display all critical system diagnostics and detect a pure tone (unique to each group) in close proximity to enter or exit the SOS state.

3. The **MDPS** subsystem is responsible for implementing the motor control of the MARV to achieve the setpoints communicated by the SNC. The actual MARV rotation angle, speed and distance must be measured and communicated to the SNC for critical system diagnostic display. **Distance refers to the distance the MARV travelled since the previous stop or rotation**. Wheel rotation sensing must be done with a rotary encoder using the wheel itself. Students will be provided with standardised 3D printed wheels and the stl-file to 3D print the wheels yourself, if needed be, that will fit on your EMK-equivalent motor shafts. Each group member will however be provided with a new motor rated at 6 V, 15 rpm. See Appendix B for what the wheel looks like. Students may however use their own DC motor and wheel/encoder designs.

When the specified pure tone is detected during the MAZE state, the system must transition into an emergency state referred to as the **SOS state**. While in the SOS state, the MARV is expected to come to a stop within a specified time and wait in the SOS state for a repeat of the pure tone. The second pure tone then triggers the transition back into the MAZE state. Upon completion of the maze, i.e. after having crossed the red line, the MARV must rotate 360° and then transition into the initial IDLE state.

Table 1: State outcomes and transition conditions

State ID	State Outcomes	Transition Conditions
IDLE	System awaits user input via touch activation through the touch-activated interface, which initiates a state transition to the CAL state. No output is expected on any of the other subsystems in this state.	The system must start in the IDLE state. The system must transition into the IDLE state from the MAZE state when the maze has been completed or if a touch activation is sensed the MAZE state.
CAL	The system performs calibration of the sensors in the SS and the MARV speed in the MDPS. After completion of the calibration sequence, the SNC subsystem is expected to display available system diagnostics (see Table B.1 in Appendix B for details on system diagnostics).	When the first touch activation is sensed in the IDLE state, the system must transition into the CAL state.
MAZE	The MARV must navigate through the maze (see Section 9). The SNC must display all critical system diagnostics (see Table B.1 in Appendix B for details on system diagnostics).	The system must transition from the CAL state into the MAZE state after the second touch activation OR from the SOS state after the second pure tone detection.
SOS	The MARV is expected to come to a full stop within the specified time. The SNC must display all critical system diagnostics in this state. A second pure tone detection must transition the system from the SOS state back into the MAZE state.	When a pure tone is detected, the system must transition between the SOS and MAZE states.

5. System Specifications

1. The system architecture as depicted in Figure 2 must be implemented.
2. The system must implement the software state diagram as depicted in Figure 3 and Figure A.1 in Appendix A.
3. The system must adhere to the state outcomes and transitions as stipulated in Table 1 and illustrated in Figure 3.
4. The system must implement a communication and interface protocol between subsystems as stipulated in the SCS documentation (Appendix A).
5. The physical dimensions of the designed system must adhere to the specifications as stipulated in the PEC documentation (Appendix B).
6. The system needs to operate independently of wall socket power when navigating the maze, but may operate from a DC power supply when testing individual subsystems with the HUB.
7. The system must adhere to the navigation specifications as described in Section 9b.

It is highly recommended to re-use components from previous modules, in particular EMK310, to limit the cost.

6. State-and-Navigation Control (SNC)

a. Subsystem Description

The SNC subsystem is essentially the brain and ears of the MARV responsible for providing a human-machine-interface (HMI) and for controlling all state and navigation functions. HMI components required for the user to control the MARV and for the MARV to communicate with the user are:

- a touch-activated sensor to initiate the first state transition (i.e. from IDLE to CAL) and to transition from the MAZE to the IDLE state. This can be an app based activation, a capacitive touch sensor of some sort, but not just a simple push button.
- a pure tone sensor to transition between the MAZE and SOS states,
- a critical system diagnostic indicator or display.

The SNC is responsible for controlling and communicating all state transitions to the SS and MDPS or alternatively the HUB if the SS and/or MDPS is being emulated using the HUB. The SNC must receive and display all critical system diagnostic data from the SS and MDPS. Once in the MAZE state, the SNC is responsible for making the decisions to adhere to all the navigation specifications stated in Section 9b based on the SS data received and then communicating the desired motor action instructions to the MDPS.

It is compulsory that the subsystem's functionality and architecture are implemented as outlined in the Functional and Architectural block diagrams described in Figures 1 and 2 respectively.

b. Subsystem Specifications

The SNC must:

1. be able to function using one, or a combination of, the following input voltages, with a tolerated variance of 5%:
    a. 3.3 V
    b. 5 V
    c. 9 V
2. adhere to the standard communications protocol as stipulated in the SCS.
3. manage and communicate all state transitions as indicated in Figure 3 and described in Table 1.
4. await End-of-Calibration (EoC) signals from the SS and MDPS within the CAL state before displaying all the critical system data and allowing a 2nd touch of the touch sensor to transition the system from the CAL to the MAZE state.
5. make the decisions needed to meet the navigation specifications stated in Section 9b based on the SS and MDPS data received and then communicate the desired motor movement instructions to the MDPS.
6. detect a pure tone at the frequency as per your group number given in the table below at a maximum volume of 60 dB at a minimum distance of 10 cm from the microphone. Upon detection of two tones, each having a duration of between 0.5 s to 1.0 s, within 2 seconds from each other, the system must transition from MAZE into the SOS state and remain until another set of two tones are detected. *No digital filters are allowed*.

Groups	1-16	17-32	33-48	49-64
f (Hz)	1400	2000	2800	4000

7. receive and then simultaneously display the following critical system diagnostic data using a web or smartphone application, an LCD screen or seven-segment displays, according to the display requirements as set out in Table B.1 the PEC documentation.
    1. Colour seen by each sensor in the sensor array.
    2. Present system state: IDLE, CAL, MAZE or SOS.
    3. Last executed angle of rotation.
    4. Tangential wheel speeds in [mm/s].
    5. MARV distance covered in [mm] since the previous stop or rotation.
    6. Last detected and recorded incidence angle in degrees.

7. Sensor Subsystem

a. Subsystem Description

The Sensor subsystem is essentially the eyes of the MARV with some intelligence built into it. The SS is responsible for detecting and distinguishing between the colours Red, Blue, Green, Black and White. The functionality and operation of the designed sensor circuit may be influenced by minor colour variations in the printed track colour, ambient lighting conditions or other unexpected external factors. Therefore, it is required that the output from the designed sensor circuit is subjected to a calibration state upon system startup (i.e. receiving a prompt from the SNC to enter the CAL state).

Upon receiving the command to enter the MAZE state, the responsibility of the SS is to provide the system with the necessary information to make decisions on how to navigate. This is achieved by communicating the colour seen by each sensor as well as the calculated incidence angle of the sensor array (i.e. MARV) to the SNC or HUB. To achieve this, the SS will require the distance travelled by the MARV since the previous stop or rotation from the MDPS. For debugging and validation purposes, the SS is also required to physically visualise or display the colour that each sensor in the array is sensing in real time whilst the subsystem is operational (indicated in Figure 2).

It is compulsory for the subsystem's functionality and architecture to be implemented as outlined in the Functional and Architectural block diagrams described in Figures 1 and 2 respectively.

b. Subsystem Specifications

The SS must:

1. be able to function using one, or a combination of, the following input voltages, with a tolerable variation of of 5%:
    a. 3.3 V
    b. 5 V
    c. 9 V
2. communicate with the SNC or HUB using the standard communication protocol as stipulated in the SCS.
3. have three (3) colour sensors in the sensor array. The sensors must be in a straight array, parallel to the wheel axis (refer to Figure B.1a in Appendix B).
4. be able to detect and classify the following colours as printed on the maze test block: White, Black, Red, Green, Blue. Refer to the PEC documentation and Figure B.2 for details and colour standards.
5. communicate the colour being detected by each sensor to the SNC or HUB when the calibration is completed.
6. visually depict the colour seen by each sensor in the array in real-time (LCDs not allowed).
7. implement the states and transition between operating states as set out in the SCS.
8. calibrate within 60 seconds and send an end-of-calibration (EoC) signal.

9. use the two leftmost or two rightmost sensors to determine the angle of incidence (θ_i) within a $5°$ accuracy for $5° < \theta_i \leq 45°$ and communicate the result to the SNC or HUB adhering to the formatting requirements as stipulated in the PEC documentation.[1]
10. communicate the end-of-maze condition to the SNC or HUB when all the sensors detect the colour Red as it moves across the RED line.
11. transmit no data to the SNC or HUB, while in IDLE or SOS state.

[1] If the incidence angle is > 45°, the MARV does not need to know what the angle is, just that it is greater than 45° and this the SNC can determine without a calculated angle from the SS.

8. Motor Driver and Power Supply Subsystem (MDPS)

a. Subsystem Description

The MDPS is the legs and the stomach of the system taking the MARV where the SNC tells it to go and providing the entire system with energy. The MDPS is responsible for supplying specific regulated voltage levels to the rest of the system as required by the system.

In the MAZE state, the MDPS is responsible for controlling the MARV speed and direction as received from the SNC. It is also responsible for calculating the distance travelled since the previous stop as well as the actual MARV rotation (turn) made in degrees. **Rotation must be executed and measured using the rotary encoder and not delays or timers**. The MDPS must transmit i) the present speed, ii) distance travelled since the previous stop and iii) the degrees of the last rotation/turn to the SNC or HUB.

Upon receiving the SOS state transition notification from the SNC, the MDPS is responsible for stopping the MARV as specified below.

It is strongly advised that optical isolation of the microcontroller is considered for a specific part in this subsystem. Furthermore, it is compulsory that the subsystem's functionality and architecture are implemented as outlined in the Functional and Architectural block diagrams described in Figures 1 and 2 respectively.

Power Supply Implementation:
The 3.3 and/or 5 V and/or 9 V rails may be implemented with off-the-shelf buck converter modules or passive regulators.

b. Subsystem Specifications

The MDPS must:

1. operate independently of wall socket power when integrated with the other two subsystems.
2. output, externally, voltage levels of 3.3 V and/or 5 V and/or 9 V to within a 5 % accuracy under all load conditions.
3. calibrate the forward tangential wheel speeds (v_{op}) in the CAL state within 60 seconds.
    a. The designed v_{op} must be obtained from the SNC in IDLE state. Refer to the **SCS:** Control Command Library table in Appendix A for detail.
    b. The wheels must stop once calibrated.
4. enable all necessary movements of the MARV to adhere to all the navigation specifications in Section 9b.
5. be able to propel the MARV forward at a speed v_{op} of at least[2] 10 mm/s.
6. ensure a speed to within 5% of the speed set by the SNC.
7. rotate around the midpoint between the two wheels of the MARV (see Figure B.1a).
8. allow rotation of the MARV of between 5° and 360° with an error of ≤ 5% for rotation angles > 100° and a maximum error of 5° for rotation angles ≤ 100°.

[2] The maximum speed allowed is 255 mm/s so as to transmit the measured speed in a single byte - refer to appendix A, but the faster you go, the harder it will be for the MARV to navigate.

9. determine the distance travelled since the previous stop with an error ≤ 5%.
10. communicate with the SNC or HUB using the standard communication protocol as stipulated in the SCS.
11. communicate the following to the SNC or HUB, according to the display requirements in the PEC documentation:
    a. Tangential wheel speeds in [mm/s].
    b. Distance travelled since the previous stop or rotation in [mm].
    c. Last rotation angle of the MARV in degrees.
12. come to a complete halt in less than 2 seconds upon entering the SOS state.
13. report no data to the SNC or HUB in the IDLE state.
14. stop and rotate 360° on detection of the SS's End-of-Maze transmission.

Important: Speed, distance and rotation must be determined using the rotary encoders, not timers or delays.

9. Navigation Control (NAVCON)

a) Description

Refer to the cover page of this and the QTP document for an example of a maze. Black is a "wall", blue is a "closed door" (effectively equivalent to black), green is an "open door" and red is an "exit" or "entrance", though the MARV will start inside the "entrance". Why does the maze have blue lines if they are essentially treated as black? If we remove the blue lines, your MARV would not only have to navigate, but it would also need a maze solving algorithm. With the given navigation specifications, the blue lines prevent the MARV from getting lost in the maze and it tests if the sensors can detect blue. There are primarily four directions to consider when navigating.

- REVERSE: the direction that the MARV came from.
- FORWARD: the direction in which the MARV will be moving when encountering a line.
- RIGHT and LEFT rotation: As specified below, or incremental adjustments to make small course corrections if the MARV is not travelling perfectly parallel to a wall.

b) Specifications

The navigation control must be implemented as follows when in the MAZE state.

1. a) Not one of the wheels is allowed to ever cross a black or blue line.
   b) Only the left or right most sensors (s1 or s3 in Figure B.1a, Appendix B) are allowed to cross the blue or black line when used to determine the angle of incidence (see Sensor specification no. 10). Sensor 2 (middle sensor) may only move up to and onto the blue or black line for line detection.
   c) After a rotation, all sensors must be within the line boundaries for forward movement.
2. The MARV must be able to perform the following distinct turn functions:
   a. turn $90° \pm 5°$ RIGHT
   b. turn $180° \pm 5\%$ LEFT or RIGHT
   c. turn $360° \pm 5\%$ LEFT or RIGHT
   d. an incremental rotation of $\leq 5°$ RIGHT AND LEFT (required for steering correction).
3. The MARV must stop *before* changing direction (forward to reverse or vice versa) or rotating.
4. The MARV may not traverse (cross) a green or red line with an incidence angle of $\theta_i > 5°$.
   If the MARV encounters a green line at an angle of $5° < \theta_i \leq 45°$, it must stop, reverse, apply steering correction, and re-attempt traversing the line.
5. If in going forward the MARV detects a green or red line and finds that $\theta_i > 45°$, the MARV must reverse and apply steering correction through a single rotation of $\leq 5°$ **toward** the line. This process (forward, detect, reverse, rotate) must repeat until $\theta_i \leq 45°$ at which point specification 4 must be adhered to.

6. The following sequence[3] must be applied when encountering a black or a blue line at an incidence angle of $\theta_i \leq 45°$ when the MARV is driving forward. The MARV must:
    a. Stop and reverse,
    b. Apply a $90° \pm \theta_i$ RIGHT turn to move forward parallel to the black/blue line on its left,
    c. Continue moving forward,
    d. If, **after having turned right,** the MARV detects a green line OR no line, it must keep on driving forward,
    e. If**, after having turned right,** the MARV, detects a blue OR a black line **BEFORE detecting a green line,** it must
        i. Stop and reverse,
        ii. Apply $180° \pm \theta_i$ turn (LEFT or RIGHT) to move forward parallel to the black/blue line on its right,
        iii. Drive forward[4].
7. If in going forward the MARV detects a black or blue line at an incidence angle $\theta_i > 45°$, the MARV must reverse and apply steering correction through a single rotation of $\leq 5°$ **away** from the line. This process (forward, reverse, rotate) must repeat until the MARV steers clear of the wall (black) or closed door (blue).
8. After the MARV has crossed red (exit), the maze is solved and the MARV must transition into the IDLE state.

[3] All mazes are set up such that following this sequence will navigate through the maze successfully.
[4] A RIGHT turn followed by a 180°turn effectively then constitutes a LEFT turn.

10. Project Evaluation Schedule

The development of your A-Maze-Eng MARV will be evaluated in five project phases spread through the semester with each counting the indicated % toward your semester mark. Deliverables and criteria provided are subject to change in which case sufficient notice will be given.

Project phase 0: Communication integration (2%) - 5 August 2025

Objective:

- In years past, the software or communication integration of the three subsystems in Phase 4 has proven the downfall of many groups in successfully demonstrating an integrated system. Students underestimate the complexity and time required.
- Given that the first three weeks of the semester traditionally have no other practicals, it was decided to create Phase 0 in 2025 so that students can design and implement the communication integration when there is time to do so.
- Groups will thus be required to design and implement the inter subsystem communication before they will be allowed to progress to Phase 1. This will require students to:
    - Study this project guide to obtain a holistic overview of the system and subsystem requirements and specifications.
    - Study and understand the SCS in Appendix A.
    - Design and implement a system whereby their three controllers can transmit to and receive from each other as per the SCS in Appendix A.

Modus operandi:

- Each group will have 10 minutes to demonstrate their communication integration to the evaluator.

Deliverables:

- Groups must successfully demonstrate that their three microcontrollers can communicate with each other as per the SCS in Appendix A. Data bytes can be hardcoded. No other hardware is required.
- The demonstration must implement a push-button on each subsystem's controller to step through the states. I.e., the evaluator must push the button before the sub-system transmits its next data packet as per the SCS state diagram in Figure A.1.
- Visual confirmation and proof on any platform must be provided to convince the evaluator that each subsystem receives and transmits the necessary data packets.
- Lab book with the design, experiments, observations, results and wiring diagrams. Herewith a quick video example of what the evaluators would typically like to see in a lab book. The lab book may be soft copy as long as it is handwritten and in the same format as would be a hardcopy.

Evaluation criteria:

- Do the subsystems successfully communicate with each other as per the SCS? Yes/No? [8]
- Was the push-button requirement implemented? Yes/No [1]
- Do the lab books contain the necessary design and wiring diagrams? Yes / No? [1]

Project phase 1: Concept Definition, Design and Simulation (5%) - 18, 19 August 2025

Objectives:

- Evaluation of the concept definition and student's understanding of the total system through a 2D paper model of a MARV and their maze test block by manually moving the MARV on the test block and discussing various scenarios in terms of system and subsystem action and reaction as it is moved. Herewith a short video discussion.
- Individual evaluation of the design and simulation (where appropriate) of the blue architectural blocks and relevant software flow diagrams within each of the three subsystems as shown in Fig. 2, i.e. the system components as per the "*Hierarchy of Complex System*" in your textbook (study unit 1.2).
- Students must present all their designs and calculations, their simulation or software test bench results as well as the comparison of the simulation / test bench output to their theoretical design outcomes in a lab book. Students are strongly advised and encouraged to use hardcopy lab books, but soft copy will also be accepted.

Modus operandi:

- There will be four evaluation teams of three ALs or demis each evaluating one group at a time.
- Students must have their 2D MARV paper model, their lab books with the designs and simulation results ready and the simulations up and running when the evaluation team arrives.
- The concept definition component will be evaluated as a group during the first 5 to 10 minutes of the evaluation.
- Each of the three subsystems/students will then be evaluated in parallel by one of the ALs/demis in the evaluation team.
- The total duration of the evaluation will be approximately 20 minutes only and thus it is critical that a group is ready when the evaluation team arrives.

Deliverables:

The following will be evaluated for each subsystem with marks indicated in [ ].

CONCEPT DEFINITION [4]

- Students must design and print out the 2D, top view, of their MARV clearly showing the sensor array location and distribution as well as the location and size of the wheels. Refer to Fig. B.1a for a poor example that won't work, but it gives an idea..
- The evaluator will move your paper MARV within the test block creating different scenarios, following your instructions based on the SCS state diagram (Fig. A.1) and asking questions to evaluate the depth of your understanding of the system and subsystem functions and specifications.

MDPS [8]

- Driver and H-bridge design [1] and simulation [1].
- Rotary encoder design [1] and simulation [1].
- Speed and distance calculation algorithm [1] and test bench [1].
- Lab book [2].

SENSOR [8]

- Sensor array design [1] and simulation [1].
- Display design [1] and simulation [1].
- Angle of incidence calculation algorithm [1] and test bench [1].
- Lab book [2].

SNC [8]

- NAVCON design [1] and test bench [1].
- Pure tone detector design [1] and simulation [1].
- Activation circuit design [1] and simulation [1].
- Lab book [2].

Evaluation criteria:
The design and simulation / test bench result of each subsystem component as well as any flow diagrams will be evaluated separately using a three-point scale, for example:
1 = Good to excellent, more than 75% done.
0.5 = Acceptable/sufficient with some shortcomings, heading in the right direction.
0 = Poor design/simulation/testing/logic or less than 25% done .

The lab book, counting 2 marks, will be evaluated on how well or with what ease another student can use the lab book to follow, understand and reproduce what has been designed and simulated.

The concept definition, counting 4 marks, will be evaluated based on the student's knowledge, understanding and conceptualisation of their subsystems specifications and functioning.

Evaluation sheets are available on ClickUP.

Note: A circuit diagram in itself is not a design but primarily a depiction of the result of the design process. Your design must show how you got the resistor values in your circuit diagram, motivate your chosen topologies and parts. Why is that resistor 10 kΩ? How did you decide on that particular transistor for your H-bridge? Why did you implement an array, rather than separate distinct variables, in your code?

Project phase 2: Prototypes and Developmental Tests (8%) - 8, 9 September 2025.

Objectives:

- Individual evaluation of the latest simulations (where appropriate) of the blue architectural blocks and relevant software flow diagrams within each of the three subsystems as shown in Fig. 2, i.e. the system components as per the "*Hierarchy of Complex System*" in your textbook (study unit 1.2).
- Individual demonstration of the system component prototypes using appropriate developmental tests. Prototypes may still be on a breadboard.
    - Students must design and demonstrate their own developmental tests (study units 1.3 and 3.1) for their system components to show the evaluators how well the prototypes compare with the simulations and adhere to the specifications.
- Demonstration of subsystem communication with the HUB and with the other subsystems.
- Students must present all their designs and calculations, their simulation or software test bench results as well as the comparison of the simulation / test bench output to their prototype outcomes in a lab book.

Modus operandi:

- All three subsystems/students will be evaluated at the same time, in parallel, by one of the ALs/demis in the evaluation team.
- The duration of the evaluation will only be 15-20 minutes, therefore students must have everything setup-up and ready when the evaluation team arrives.

Deliverables and evaluation criteria:

Similar to phase 1, simulations, prototypes and developmental tests will be evaluated separately using a three-level scale. Evaluation sheets are available on ClickUP.

Developmental test evaluation guidelines:

- Is this a valid/good/relevant developmental test?
- Is the test design acceptable?
- What is the quality of the test outcomes and how well does it compare to the simulation outcomes and evaluate adherence to the specifications?

HUB and inter-subsystem communication: For this demonstration you may hardcode valid data packets <dat1,dat0,dec> where hardware has not yet been implemented. It is the students' responsibility to design a developmental test proving that their subsystems communicate with each other as per the SCS state diagram in Appendix A.

The lab book, counting two marks, will be evaluated on how well or with what ease another student can use the lab book to follow, understand and reproduce what has been designed and simulated.

Project phase 3: Individual QTP demonstration with the HUB (25%) 6, 7 October 2025, re-demo 10 Oct.

As the weight of this project phase indicates, this is the most critical of all the demonstrations and, as per the the subminimum requirements in your study guide, requires a mark of at least 50% for advancement to project phase 4 and admission to the examination and effectively final year project!

Logistics: Where and when, preparation, Post-demo, FAQ's

- Prac 3 Demonstration Logistics

Deliverables:

- All hardware must effectively be complete and ready for integration with circuits on either veroboard or PCB and with only minor revisions required for final integration for Phase 4. Circuits on breadboard will be evaluated where possible.
- All subsystems will be expected to be able to demonstrate all their QTPs. However, the evaluators will select which of the QTPs you must demonstrate for evaluation because there is not enough time to demonstrate all the QTPs.
- Failure to communicate or demonstrate with the HUB will result in an immediate zero mark.
- You may reset your controller and HUB during the demo between QTPs, but you will not be allowed to reprogram the controller for every QTP. I.e. the same program must be used to do all the QTPs.
- This demonstration has a subminimum for exam entry associated with it and therefore students will have the opportunity to re-demo should they fail this project phase. The re-demo mark will be capped at 50%.

Evaluation Criteria:

- Detailed evaluation sheets are available on the Phase 3 ClickUP page.
- You will notice that the evaluation criteria corresponds exactly with the pass requirements listed in the QTP document.
- The hardware integration mark evaluates the extent to which the 3 subsystems have been designed to physically integrate to form an integrated system. I.e., how well can the subsystems be combined, put together, to form the final MARV that will run in Phase 4?
- The software integration / communication refers to the inter-subsystem communication as per the SCS in appendix A. I.e., can the 3 subsystems communicate with each other as per the SCS?

Project phase 4: Integrated system demonstration (5%) - 4 Nov. 2025

***Deliverable*:**

- A fully integrated system capable of navigating through a maze.
- All groups must present their system even if it is not operational otherwise it will result in non-attendance and hence exam refusal.

Evaluation criteria

- Integrated systems will be evaluated based on their performance in navigating one of three mazes of equal complexity. The evaluation sheet is available on ClickUP.
- An integrated MARV that can go from IDLE, through CALIBRATION and into MAZE state with the MARV moving forward with some reaction at the first line detected, will already earn a 50% pass mark.
- There will be 10 lines to navigate. Each line successfully navigated will earn you 5%. Successfully navigating incidence angles of >45° will earn you an additional 10%, meaning that the maximum mark for this evaluation is 110%.
- A test maze will be placed in the A lab from 22 to 31 October for you to test your system. Keep in mind that you can test everything using only the test block you received.

Re-Grouping

- In cases where students did not pass Phase 3 and are excluded from Phase 4 resulting in incomplete groups, the remaining group members have the following options:
    - where possible, new groups can be formed to complete the system.
    - the missing subsystem can be built using off the shelf modules,
    - the missing subsystem can be borrowed from another group, integrated and demonstrated with the remaining subsystems.
- If you are the only person left in the group, ask another group if you can test your subsystem with theirs.

To assist you in re-grouping into complete systems do the following:

- If you are alone in a group or only two left in need of a third, please complete this Phase 4: Re-grouping google form.
- Use the responses to find and contact students to re-group yourselves for Phase 4
- If you formed a new complete group, please:
    - complete this Phase 4 - Merged Groups form.
    - Amend your group members on the AMS in Assessment D10 Phase 4: System integration Group mark.

If you are in a complete group and willing to "**rent" out your subsystems to incomplete groups for bonus marks for your whole group**, please also complete the Phase 4: Re-grouping Google form so that incomplete groups can contact you.

Modus operandi

- Demonstrations will be done in Eng III, level 5, in the open space between lecture halls 5 and 6. There will be three evaluation stations as indicated on the right.
- There will also be "warm-u" mazes in the space between lecture halls 4 and 6 for you to prepare before your demonstration.
- Complete the Phase 4 booking form. No-shows or no hardware means project incomplete which means exam refusal.
- Report to your demo station 10 min before your booked time slot.
- After your demo, **you must hand back your reusable test block or proof of payment** so that we can replace it next year. Reusable = No marks, no damage, clean. Herewith the banking details, cost per test block is R 70. You should also be able to make payment at the SSC cashiers, just take the banking details with you.

Video for Examination Report

As part of the group section of the exam assignment, you will need to compile a table of your system test results and upload a video as evidence. You will record the video during the Phase 4 demonstration. The table must include:

a) The specifications to be verified (as per the Project Guide, Section 5).
b) Expected results, based on your design efforts and simulations. Please note that the expected results are not the ideal results that you would have liked to achieve, but rather realistic expectations based on your specific design, including any shortcomings.
c) Measured results of your tests

As there are no formal QTPs defined at system level, you will need to think about how you will demonstrate how you met (or how you didn't meet) the system specifications and make sure to demonstrate this in your video. You can structure the video demonstration as you wish; test tracks will be made available on demo day for this purpose. Please save the video and upload it along with the exam assignment.

Appendix A

System Communication Standard (SCS)

The subsystems communicate with each other (and the HUB when a subsystem is missing or inoperational thus being emulated by the HUB) using UART by sending data packets. Each packet consists of 1 control byte and 3 data bytes that when combined, forms a single 32 bit DWORD or 4 byte data PACKET = <SYS:SUB:IST:DAT1:DAT0:DEC> with the structure as shown below.

The serial communication has the following setup:

- **Baud Rate** = 19200.
- **Asynchronous** transmission/reception
- **8-bit** transmission/reception mode

Data Packet: Packet contains 1 control byte followed by 3 data bytes. (R/W = Readable/Writable)

CONTROL<31:24>	**DAT1<23:16>**	**DAT0<15:8>**	**DEC<7:0>**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
C O N T R O L <7:0>							
SYS<1:0>		SUB<1:0>		IST<3:0>			
bit 31							bit 24

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
DAT1<7:0>							
bit 23							bit 16

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
DAT0<7:0>							
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
DEC<7:0>							
bit 7							bit 0

bit 31-30	**SYS:** System state 11 = 3 = SOS_STATE; System is in emergency mode. 10 = 2 = MAZE_STATE; System is in MAZE mode. 01 = 1 = CAL_STATE; System is in calibration/monitor mode. 00 = 0 = IDLE_STATE; System is in idle mode.
bit 29-28	**SUB:** Subsystem number sending the packet 11 = 3 = Sensor subsystem. 10 = 2 = MDPS subsystem. 01 = 1 = SNC subsystem. 00 = 0 = HUB (reserved).
bit 27-24	**IST:** Internal state of the subsystem sending the packet 1111 = Substate 15. … 0011 = Substate 3. 0010 = Substate 2. 0001 = Substate 1. 0000 = Substate 0.
bit 23-16	**DAT1:** Upper date byte of the transmitted data.
bit 15-8	**DAT0:** Lower data byte of the transmitted data.
bit 7-0	**DEC:** Decimal or general purpose data byte of the transmitted data.

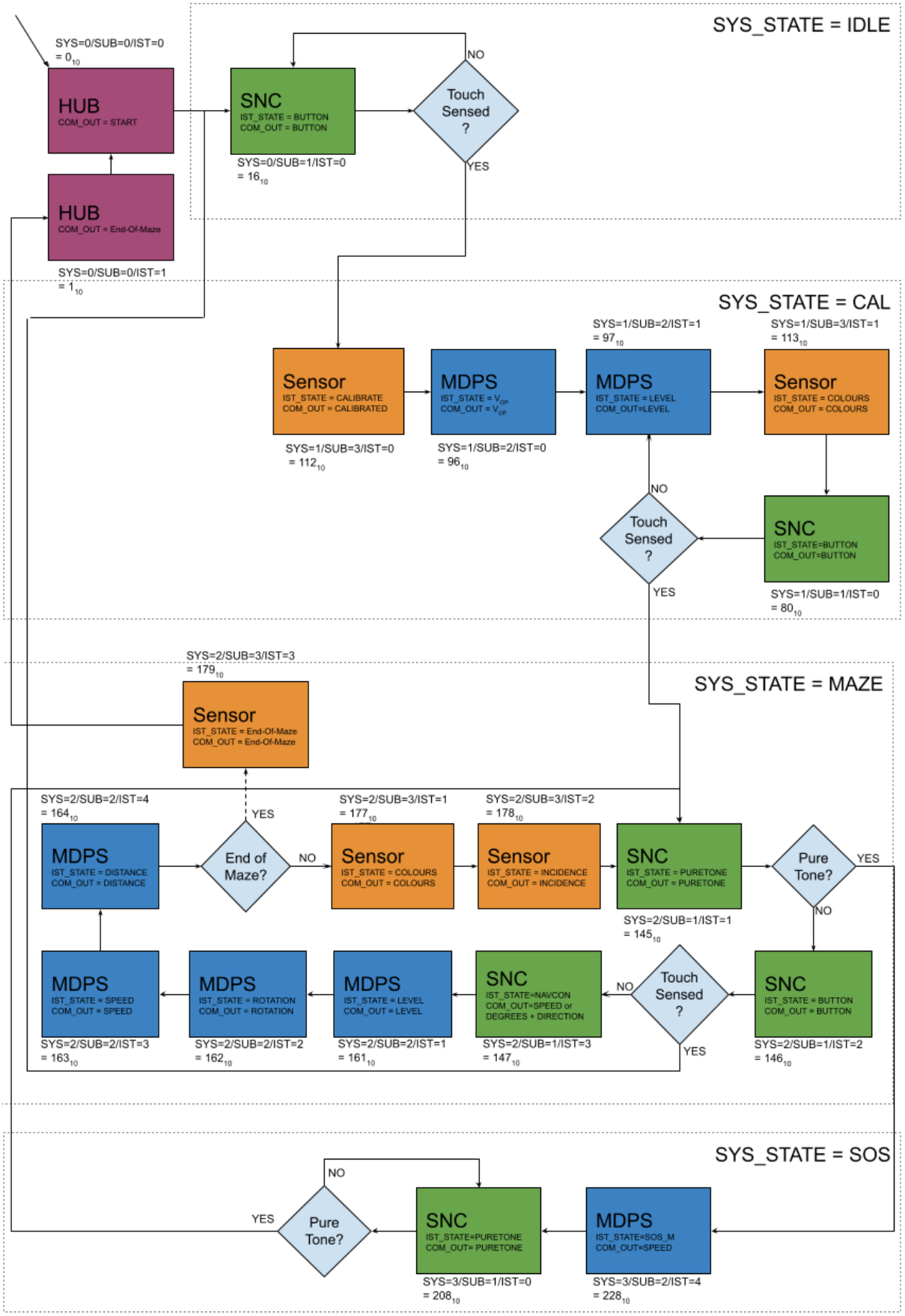
The system and subsystems start operation in IDLE mode and then sequentially move from one state to the next as shown in the state flow diagram in Figure A.1 on p25. If the HUB is used, it will transmit an all-zero control byte at which point the SNC starts detection of the touch-activated sensor. The rules of the SCS are as follows.

1. All three subsystems must be connected together to keep track of the system and internal subsystem states and react accordingly. The control byte is used to determine the state of the system (SYS) and the internal state of a subsystem (IST).
2. Only one subsystem is allowed to transmit at a time while the other two receive and interpret what has been sent.
3. When and what a subsystem is allowed to transmit is dictated by the control byte received as depicted in the state flow diagram in Figure A.1.
4. Each subsystem will send the control byte to indicate the transition/completion of its function along with the necessary 3 data bytes. The HUB will only send serial packets for the subsystems it is emulating.
5. Only once a subsystem has finished sending its data packet, is the system and subsystems allowed to transition to the next state.

As an example, let refer to Figure A.1 and assume the system is in the MAZE state and that the SNC has just indicated to the system that it is now in NAVCON internal state by transmitting the control byte <SYS=2/SUB=1/IST=3> = <10010011> along with a required MARV speed of 0. The control byte indicates to the SS that it is not allowed to transmit now and to the MDPS that it may now transmit the battery voltage level (COM_OUT = LEVEL). (*As from 2022 this component has been removed from the MDPS subsystem, but it is kept in the state diagram otherwise the HUB will require significant revision. The MDPS must therefore simply send all data bytes <DAT1:DAT0:DEC> as zeros as indicated in the SCS.*) This does not mean however

that the MDPS must immediately transmit the voltage level (zeros). Since the SNC's transmission showed the MARV's speed must be set to zero, i.e. stop the MARV, the MDPS could immediately upon receipt of the SNC's transmission, adjust the PWM to the two motors so as to stop the motors and *then* move to IST_STATE = LEVEL to transmit the rotation angel followed by the IST_STATE = ROTATION. But before the MDPS moves to IST_STATE = SPEED to transmit the speed of the MARV, it should wait until the sensors confirm that the MARV has come to a standstill and only then does the MDPS need to go into IST_STATE = SPEED and transmit the speed of the MARV. All this time the SS simply receives and monitors the communication until the MDPS finally sends the control byte <SYS=2/SUB=2/IST=4> = <10101000> indicating to the SS that it can now either go into the End-Of-Maze or COLOURS internal state. The SS must however continue to display the three sensor colours on its onboard display.

Following Figure A.1 is the control command library table containing all the necessary details regarding each subsystem's internal state within each of the four primary system states. It details the control byte command instructions, the structure and content of the DAT1, DAT0 and DEC bytes as well as descriptions of each internal state.



d

Figure A.1 - MARV SCS State Diagram.

SCS: Control Command Library (x = unknown/don't care).

	Control Byte <31:24>					
Subsyst.	**SYS <1:0>**	**SUB <1:0>**	**IST <3:0>**	**Control Command Description**	**Control Action**	**Additional Notes**
HUB	0	0	0	*HUB.* Initiate system (start) command.	System and Subsystems move to IDLE state.	SNC starts touch detection in IDLE state.
	0	0	1	*HUB.* Terminate system (stop) command.	System and Subsystems move to End of maze state.	
	0	0	x	*HUB.* Reserved/ Not in use.	None.	
Subsyst.	**SYS <1:0>**	**SUB <1:0>**	**IST <3:0>**	**Control Command Description**	**Control Action**	**Additional Notes**
SNC	0	1	0	*SNC.* Touch Detected (IDLE) and desired operating speed v_{op}.	DAT1 = 1 if touch detected, else DAT1 = 0. DAT1 = 0: Remain IDLE DAT0 = Designed operating speed v_{op} when DAT1 = 1	DAT1 = 1: System and Subsystems move to CAL state. When emulated by the HUB, v_{op} is obtained via the HUB GUI
	0	1	1	*SNC.* Reserved/ Not in use.	None.	
	0	1	2	*SNC.* Reserved/ Not in use.	None.	
	1	1	0	*SNC.* Touch Detected (CAL).	DAT1 = 1 if touch detected, else DAT1 = 0. DAT1 = 1: System and Subsystems move to MAZE state. DAT1 = 0: Remain in CAL	DAT1 contains whether a touch was detected. DAT1 = 1: SNC moves to Pure Tone detection in MAZE state.
	1	1	1	*SNC.* Reserved/ Not in use.	None.	
	1	1	2	*SNC.* Reserved/ Not in use.	None.	

	2	1	0	*SNC.* Reserved/ Not in use.	None.	
	2	1	1	*SNC.* pure tone Detection (MAZE).	DAT1 = 1 if pure tone was detected, else DAT1 = 0. DAT1 = 1: System moves to SOS state DAT1 = 0: SNC moves to Touch Detection	DAT1 contains whether a Pure Tone was detected.
	2	1	2	*SNC.* Touch Detection (MAZE).	DAT1 = 1 if touch detected, else DAT1 = 0. DAT1 = 1: System and Subsystems move to IDLE state. DAT1 = 0: SNC moves to navigation control state (NAVCON)	DAT1 contains whether a touch was detected. DAT1 = 1: SNC moves to Touch Detection in IDLE state *after clearing DAT1*.

	2	1	3	*SNC.* Navigation Control (NAVCON).	DEC = 0 or 1: DATA bytes contain tangential wheel speeds.
					0 = forward
					1 = backward
					
					DEC = 2 or 3: DATA bytes contain required angle of rotation of the MARV
					2 = left (CCW, positive).
					3 = right (CW, negative).
						if DEC = 0 or 1:
						DAT1 = right wheel speed in mm/s,
						DAT0 = left wheel speed in mm/s
						
						if DEC = 2 or 3:
						DATA = <DAT1:DAT0> contains the angle of rotation in degrees between 5° and 360°.

	3	1	0	*SNC.* Pure Tone Detection (SOS).	DAT1 = 1 if Pure Tone detected, else DAT1 = 0. DAT1 = 1: System and Subsystems move to MAZE state.	DAT1 = 1: Pure Tone was detected. SNC moves to MAZE Pure Tone detection *after clearing DAT1*. DAT1 = 0: Wait/loop
Subsyst.	**SYS <1:0>**	**SUB <1:0>**	**IST <3:0>**	**Control Command Description**	**Control Action**	**Additional Notes**
MDPS	0	2	0	*MDPS.* Reserved/ Not in use.		
	0	2	1	*MDPS.* Reserved/ Not in use.		
	0	2	2	*MDPS.* Reserved/ Not in use.		
	0	2	3	*MDPS.* Reserved/ Not in use.		

	1	2	0	*MDPS.* v_{op}. (CAL) v_{op} is the default forward tangential wheel speed.	DATA bytes contain measured tangential speeds of the wheels. Control packet sent a*fter* calibration is complete.	DAT1 = right wheel v_R in mm/s, DAT0 = left wheel v_L in mm/s.
	1	2	1	*MDPS.* Battery Voltage Level (CAL).	DAT1 = 0 DAT0 = 0 DEC = 0	Battery voltage level sensing redacted in 2022. Send 0's
	1	2	2	*MDPS.* Reserved/ Not in use.		
	1	2	3	*MDPS.* Reserved/ Not in use.		
	2	2	0	*MDPS.* Reserved/ Not in use.		
	2	2	1	*MDPS.* Battery Level	DAT1 = 0 DAT0 = 0 DEC = 0	Battery voltage level sensing removed in 2022. Send 0's
	2	2	2	*MDPS.* MARV measured ROTATION	DATA = <DAT1:DAT0> contains the last measured executed rotation angle. DEC = 2: Rotation was to the left (CCW, positive). DEC = 3: Rotation was to the right (CW, negative).	The executed rotation angle refers to the actual measured angle the MARV rotated after having received a rotation instruction from the SNC. SNC displays the angle on indicators.
	2	2	3	*MDPS.* MARV Measured SPEED	DATA bytes contain measured tangential speeds of the wheels.	DAT1 = right wheel speed in mm/s, DAT0 = left wheel speed in mm/s. SNC displays the speeds on indicators.
	2	2	4	*MDPS.* MARV Measured DISTANCE moved since previous stop/rotation.	DATA = <DAT1:DAT0> contains measured distance in mm.	It is the distance the MARV travelled in a straight line to the nearest mm.
	3	2	4	*MDPS.* Pure Tone response.	Motor reduces speed to zero.	After stopping, DAT1 = 0 right wheel speed DAT0 = 0 left wheel speed.

Subsyst.	SYS <1:0>	SUB <1:0>	IST <3:0>	Control Command Description	Control Action	Additional Notes
Sensor	0	3	0	*Sens.* Reserved/ Not in use.		
	0	3	1	*Sens.* Reserved/ Not in use.		
	1	3	0	*Sens.* End of Calibration	MDPS moves to speed calibration	
	1	3	1	*Sens.* COLOURS (CAL)	DATA = <DAT1:DAT0> contains colour sensed by each sensor as a 3 bit code: W = White = 000 R = Red = 001 G = Green = 010 B = Blue = 011 K = Black = 100	DATA<2:0> = sensor 3 DATA<5:3> = sensor 2 DATA<8:6> = sensor 1
	2	3	0	*Sens.* Reserved/ Not in use.		
	2	3	1	*Sens.* COLOURS (MAZE)	DATA = <DAT1:DAT0> contains colour sensed by each sensor as a 3 bit code: W = White = 000 R = Red = 001 G = Green = 010 B = Blue = 011 K = Black = 100	DATA<2:0> = sensor3 DATA<5:3> = sensor 2 DATA<8:6> = sensor 1
	2	3	2	*Sens.* INCIDENCE	DAT1 contains the last measured angle of incidence.	SNC displays the last measured angle of incidence in degrees on indicators.
	2	3	3	*Sens.* End-Of-Maze (EOM)	End of maze detected.	SNC/HUB transmits EOM control byte. All subsystems move to IDLE state.

Appendix B

MARV Physical and Environmental Considerations (PEC)

Physical dimensions of the MARV

Each of the 16 blocks in the maze is 21 cm x 21 cm excluding the lines that are 1 cm thick. The width and length of the MARV therefore needs to be designed for it to be able to move and navigate within these dimensions. The HUB will require the following dimensions of your design as illustrated in Figure B.1a below.

From the SS:

- Distance between the sensors in the sensor array (sd1, sd2). The middle sensor (no. 2) *must* be positioned on the centre line of the MARV to serve as a point of reference for the HUB.
- Distance between the wheel axis and the sensor array axis: axial distance (A).

From the MDPS:

- Distance between the wheels or the wheel base width (W)
- Outer wheel diameter (D)

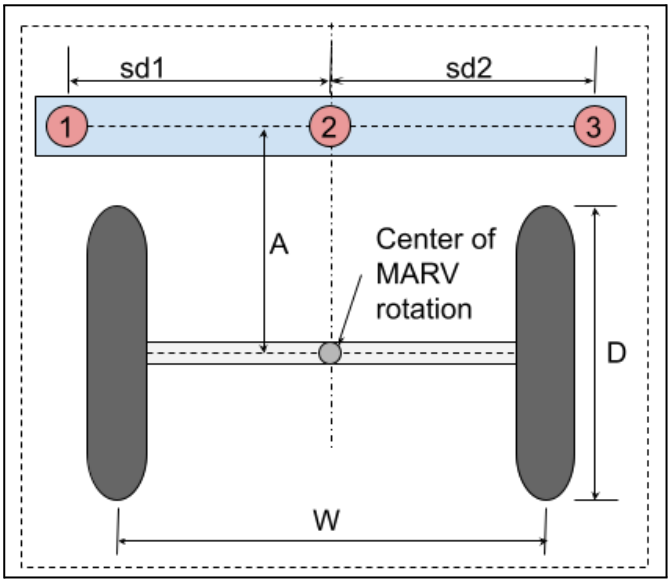
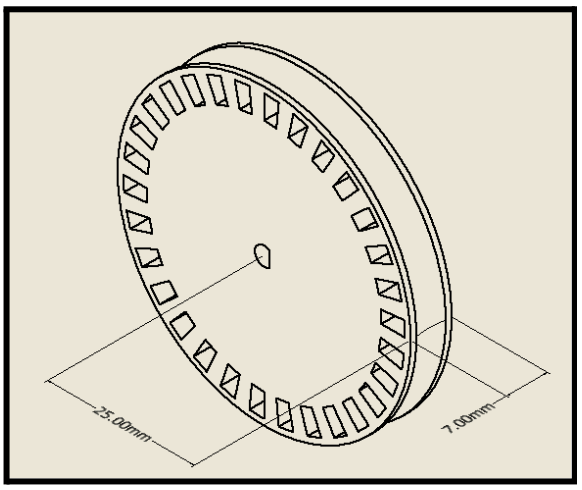


Figure B.1a) Top view of MARV dimensions required by the HUB indicating inter-sensor distances sd1 and sd2, inter axial distance A, wheel diameter D and wheel base width W.



b) Wheel design to fit the given motor shaft with 30 encoder slots and a groove for rubber bands.
Outer wheel diameter = 50 mm, width = 7 mm.

Note: Figure B.1a is just for illustrative purposes and your relative dimensions and distances will of course be different.

Critical System Diagnostics

The information to be displayed by the SNC for each of the critical system diagnostic values is summarised in Table B.1.

Table B.1 - Critical system diagnostic data display summary.

Critical System Value	**Display Information**
Sensor Colours	Simultaneously display the colour detected (seen) by each of the sensors. The system must distinguish between, and display notifications for, a minimum of five discrete colours using at least the following abbreviations: 1. W - White 2. R - Red 3. G - Green 4. B - Blue 5. K - Black *Should the student wish to implement additional colour indicators, e.g. colour indicators on an app or the colours written in text, they are allowed to do so.*
Present State	Display the four system states as: 1. IDLE 2. CAL 3. MAZE 4. SOS
MARV rotation angle	Display the MARV's last measured executed rotation angle in degrees rounded to the nearest degree.
MARV speed	Display the MARV's individual wheel tangential velocity in units of [mm/s] rounded to the nearest integer.
MARV distance	Display the distance the MARV travelled since the previous stop or rotation in [mm] rounded to the nearest integer.
Incidence angle	Display the last recorded angle of incidence in degrees rounded to the nearest degree.

Maze test block

Each group will receive a test block as shown in Figure B.2 printed on a glossy, banner PVC. The block can be used to test all the MARV specifications. Please note that this block is on loan only and must be returned at the end of the semester. The mazes are printed on the same PVC material. You can download a pdf version of the test block, but remember that your sensors will act very differently on paper than on PVC. Final demonstrations will be conducted on the complete mazes printed on PVC. A test maze will be made available in the week or two before the project phase 4 evaluation.

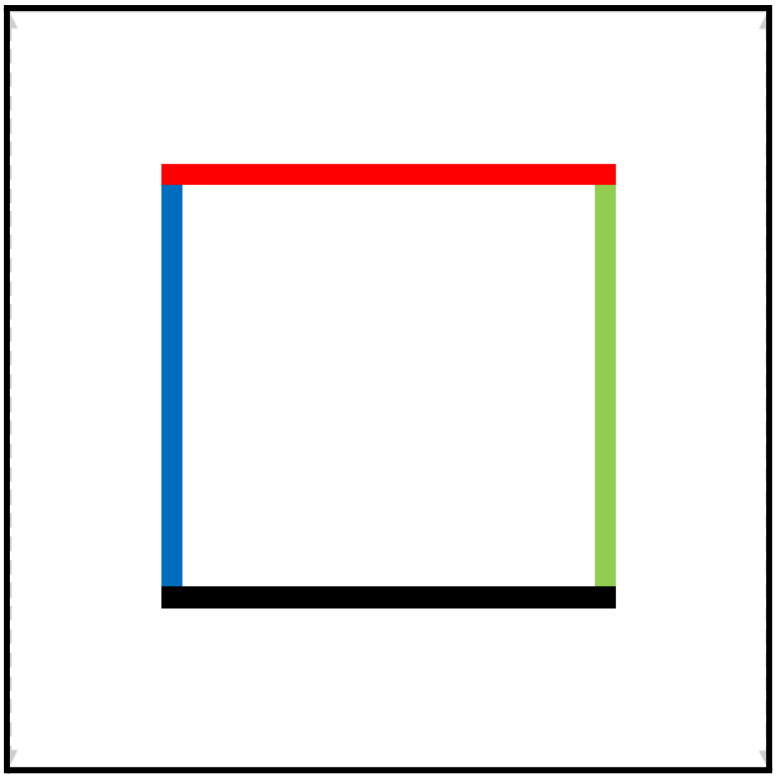


Figure B.2 The maze test block (scale 1:4)

Appendix C

Potholes

1. Serial comms with an Arduino

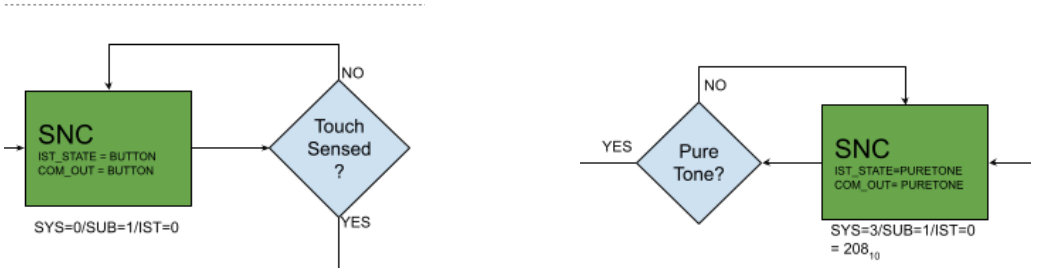
An Arduino does the following when sending a byte using the **serial.print(value)** function.

- It assumes 'value' is in string format.
- It then takes each character in 'value' and converts it into an 8 bit ASCII value.

For e.g. you want to send decimal 30 which is 0x1E via **serial.print(1E)** or **serial.print(30,HEX).**
It then converts the 1E (seeing it as a string) into a '1' in ASCII 8 bit value and an 'E' in ASCII 8 bit value and then transmits it as two separate bytes. So for every 1 byte you think you are transmitting, you are actually transmitting 2 bytes. The same principle applies using **serial.read()**, but just in reverse.

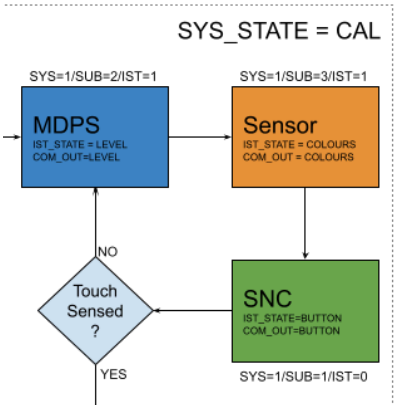
2. Spamming the HUB

Do not "spam" the HUB when in a loop such as in the state diagram sections shown below.



For example, you will send c010-0-0-0 to indicate a touch has not been sensed in CAL state. There is then no need to continuously resend c010-0-0-0 every few microseconds or milli-seconds while waiting for a touch to be made. You could for example only resend every second or even only send c010-1-0-0 once a touch has been detected. You must however send at least one c010-0-0-0 otherwise the HUB will think your SNC is dead and not transmitting.

However, in the case below, each subsystem must send its required 4-byte packet every time the preceding subsystem in the state flow has sent its 4-byte packet.



3. Do not send a STOP instruction to the system after a ROTATE instruction.

The HUB will give an ERROR message if you do. Remember that the MDPS will by default be standing still after a rotation and hence no need to tell it to stop if it is already standing still.

4. **When the MDPS should reset DISTANCE to zero**

   The definition given for the MARV distance is “the distance the MARV travelled since the previous stop or rotation”.

   - First, the MARV must stop before rotating or changing direction as stated in the specifications.
   - When the MDPS receives a STOP instruction from the SNC (c213-0-0-0), the MDPS will first transmit LEVEL and ROTATION
   - While the MDPS is transmitting LEVEL and ROTATION, the MARV might already be coming to a standstill and only once it has stopped, should the MDPS transmit SPEED = 0.
   - The DISTANCE to be transmitted is the distance travelled since the ***previous*** stop, i.e. not the *current* stop. In other words, you must first transmit DISTANCE (which will be the distance between the *previous* and *current* stops) and only *then* reset DISTANCE to zero to start counting again once the MARV starts moving forward or backwards.

   The above manner is the way in which the emulated MDPS operates.

5. **Spacing of the 3 sensors**

   Using the same sensor array used in EMK310 or EBB320, i.e. small inter-sensor distances for following a line, will not work for this system. You must *design* the inter-sensor distances based on the requirements and specifications.

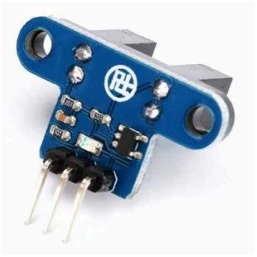
6. **The HUB makes things slow**

   Because of the nature of the HUB and the logging of all the transmissions and receptions, the HUB is significantly slower moving through the state machine than your integrated system will be. Just keep this in mind when testing your system.

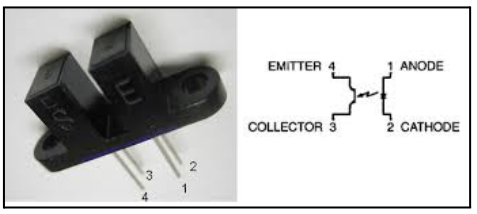
7. **Examples of Opto-interruptors and modules**

   Below a few examples of what is and what is not allowed.

Not allowed



Not allowed



Allowed without Schmitt trigger

Appendix D

Important Revision Changes

2025-06-xx: Version 1 for 2025