Functionality Tests

The tests in this document are group by levels according to the software architecture designed for the Connect-4 robot. The tests for each level are adjusted based on the abstract of the level. Additionally, there is some repetition of tests, this is done intentionally in order to place importance on that specific function of the module being verified to work in the same way on all levels. Low-Level unit and component testing was omitted from this plan due to time limitations. However, a basic draft was drawn for it, if necessary, in the future.

Test Case	Test Conditions
Level 1	
Cortex-M4	**1.** Test the initialization and configuration of Cortex-M4. **2.** Test if the core can initialise all peripherals and communicate with them and control them. **3.** Test if Cortex-M4 can receive tasks from Cortex-M7 and if it responds accordingly while providing feedback on its status. **4.** Test the state transitions of the main core logic. **5.** Test if the Cortex-M4 core can manage multiple tasks. **6.** Test that the core can detect, report, and recover from any errors.
Cortex-M7	**1.** Test the initialization and configuration of Cortex-M7. **2.** Test if the core can initialise all peripherals and communicate with them and control them. **3.** Test if Cortex-M7 can send tasks to Cortex-M4 and check if it receives feedback about their status. **4.** Test the state transitions of the main core logic. **5.** Test if the Cortex-M7 core can manage multiple tasks. **6.** Test that the core can detect, report, and recover from any errors.
Level 2 – Cortex-M4	
Task Manager	**1.** Test the initialization and configuration of the module. **2.** Test if the module can give out tasks (received from C-M7) and receive feedback about their status from their corresponding module. **3.** Test if the state transitions of the logic. **4.** Verify that the task manager can detect, report, and recover from errors.
Motor Controller	**1.** Test the initialization and configuration of the module and both motors. **2.** Verify that the module is set up to receive signals from the encoders, and home/end switches. **3.** Test if the module moves the motors in both X and Z directions. **4.** Test if the PWM signal controls the motors effectively **5.** Test if the module can read the position from the encoders. **6.** Test that the home/end-switches send the correct interrupt and stop the motor. **7.** Verify that the Motor Controller module can detect, report, and recover from any errors.
Token colour separator controller	**1.** Test the initialization and configuration of the module. **2.** Test the module is correctly set up to control the RGB and proximity sensor and the flipper. **3.** Test if the module detects the colour of tokens (red and yellow). **4.** Test if the module can separate tokens based on their colour. (Decision making, solenoid) **5.** Verify that the Token Colour Separator Controller module can detect, report, and recover from any errors.
Token picker controller	**1.** Test the initialization and configuration of the controller. **2.** Test if the controller can move the end-effector servo, read off the

	vacuum sensor and control the vacuum valve. **3.** Test if the vacuum pump generates enough pressure to pick up a token and transport it. **4.** Test if the positions of all different pick-up/drop-off points are correct. **5.** Record how long picking and releasing the tokens takes. **6.** Verify that the module can detect, report, and recover from any errors.
User Detector	**1.** Test the initialization and configuration of the module. **2.** Verify that the module can read data from the light-gate circuit. **3.** Record how fast a token dropping is recognized. **4.** Test if multiple tokens are inserted in the same column (cheat move). **5.** Test if multiple tokens have been inserted in multiple columns (cheat move). **6.** Test if token is inserted in the wrong player state (cheat move). **7.** Verify that the module can detect, report, and recover from any errors.
Board Opener	**1.** Test the initialization and configuration of the module. **2.** Test that the servo motors can open the board column by column. **3.** Test that the Task Manager can send commands for opening and closing the board. **4.** Verify that the Board Opener module can detect, report, and recover from any errors.
Level 2 – Cortex-M7	
Initialization	**1.** Verify that the initialization sets up the system to be ready for operation, including configuring the peripherals, initializing the modules.
Game controller	**1.** Verify that the Game Controller module is correctly set up to manage the overall game logic and flow. **2.** Test the module's ability to maintain and update the game state, including the board state and player turns. **3.** Test the module's ability to detect win, loss, or draw conditions. **4.** Test the state transitions of the controller. **5.** Verify that the Game Controller can detect, report, and recover from any errors.
CM4 Task Generator	**1.** Test the initialization and configuration of the module. **2.** Test the CM4 Task Generator's ability to create tasks based on the game state and requests from other modules. **3.** Verify that the CM4 Task Generator module receives accurate game state updates and next-move decisions from the Game Controller module. **4.** Verify that the CM4 Task Generator module can detect, report, and recover from any errors
Game end block	**1.** Test the initialization and configuration of the module. **2.** Test of the module is able to detect and handle a win/lose/draw condition for either the human player or the robot player. **3.** Verify that the module can detect, report, and recover from any errors.
UART controller	**1.** Test the initialization and configuration of the UART Controller module. **2.** Test the UART’s ability to transmit/receive data to and from external blocks. **3.** Verify that the UART can maintain the expected communication speed without losing data or causing errors. **4.** Test the UART Controller's ability to handle interrupts for data transmission and reception. **5.** Verify that the UART controller can detect, report, and recover from any errors.

Happy-Path Test

[AT A FUTURE MOMENT WHEN THE INTENDED USE IS CLEARER]

The intended way of using the system.

Acceptance Test

Made against initial requirements.