Component Test Plan

Unit testing is essential for Testing the correct functioning of each individual component in the software architecture of the Connect-4 robot player, including the initialization procedure, low-level code about different peripherals, and communication between the two cores.

Unit Tests

Builds confidence in the separate blocks/components.

Test Case	Test Conditions
TIM	**1.** Test timer initialization and configuration. **2.** Test the generation of interrupts when expected. **3.** Test the use of multiple timers simultaneously.
NVIC	**1.** Test NVIC initialization and configuration. **2.** Test the enabling and disabling of individual interrupts. **3.** Test that the NVIC prioritizes correctly. **4.** Test multiple interrupts. **5.** Check the functionality of interrupt masking and unmasking. **6.** Test that the NVIC can properly clear pending interrupts. ***HAL_NVIC_ClearPendingIRQ()***
EXTI	**1.** Test EXTI initialization and configuration. ***HAL_EXTI_GetConfigLine()*** **2.** Test whether the EXTI triggers the desired routine upon detecting an external event. ***HAL_EXTI_RegisterCallback()*** **3.** Test the proper handling of EXTI interrupts by the NVIC. **4.** Test that the EXTI can clear the interrupt flag correctly Pending Register (EXTI_CnPRIx) ***HAL_EXTI_ClearConfigLine()***
UART	**1.** Test UART initialization and configuration with various baud rates, data bits, stop bits, and parity settings. **2.** Test that the UART can transmit and receive data correctly. **3.** Test that the UART can properly handle interrupts related to data transmission and reception. **4.** Test that the UART can properly handle errors.
I2C	**1.** Test whether the I2C is correctly configured. **2.** Test that the I2C can properly transmit and receive data. **3.** Test that the I2C can properly handle interrupts related to data transmission and reception. **4.** Test that the I2C can properly handle errors.
LED	**1.** Test LED initialization and configuration. **2.** Test that the LED individually can turn on and off correctly.

	3. Test that the LED can blink at different frequencies and duty cycles. 4. Check the proper integration of LED control with other peripherals, such as timers.
GPIO	1. Test whether the GPIO pins on the robot player are properly configured and controlled by the software. 2. Test whether the GPIO pins have the correct input/output state. 3. Test GPIO interrupt generation and handling (EXTI).
ADC	1. Test whether the ADC is properly configured and enabled.(EXTI, DMA) 2. Test whether the ADC is able to accurately measure the sensor input voltage and convert it to a digital signal. 3. Test whether interrupt generation is when needed.
ETH	1. Establish ETH initialization and configuration with different network settings (IP, MAC, etc.). 2. Test that ETH can send and receive data packets correctly. 3. Test ETH functionality with various frame sizes and protocols. 4. Check ETH error handling and recovery.
HSEM	1. Test HSEM initialization and configuration. 2. Test that HSEM can manage access to shared resources between the two cores. 3. Test HSEM functionality with different resources and priority settings.

Functionality Tests ?

Vacuum Pump	1. Test vacuum pump initialization and control. 2. Test if the vacuum pump can generate sufficient vacuum to handle the token.
Vacuum Sensor	1. Test vacuum sensor initialization and readout. 2. Test if the vacuum sensor can accurately detect the presence of a inside the tube. 3. Test if the pressure can be distinguished when empty and when with a token.
Vacuum Valve	1. Test that the vacuum valve can effectively control the vacuum flow.
RGB Sensor	1. Test RGB sensor initialization and configuration. 2. Test if the RGB sensor can accurately detect red and yellow tokens.
IR Sensor	1. Test IR sensor initialization and readout. 2. Test that the IR sensor can detect the token drop.
Proximity Sensor	1. Test proximity sensor initialization and readout. 2. Test if the proximity sensor can detect the of object and distinguish between its set thresholds.
End-Switches	1. Test end-switch interrupt. 2. Test that end-switches can detect the end position of the motor.
Home Switches	1. Test home-switch interrupt.

	2. Test that home switches can detect the home position of the motor.
Encoder Readout	1. Test encoder initialization and readout. 2. Test that encoders can accurately measure motor position and speed. 3. Test the integration of encoders with motor control and PID calculations.
PID calculations	1. Test PID algorithm implementation and tuning. 2. Test that PID calculations can effectively control motor position and speed. 3. Test the integration of PID calculations with encoder readout and motor control.
PWM Signal	1. Test PWM signal generation and configuration. 2. Test that the PWM signal can effectively control motor speed and servo position. 3. Test the integration of the PWM signal with motor control and servo control.
Motor control – X	1. Test X-axis motor initialization. 2. Test that the X-axis motor can move to the desired positions.
Motor control – Z	1. Test Z-axis motor initialization. 2. Test that the Z-axis motor can move to the desired positions.
Servo – End-effector	1. Test end-effector servo initialization. 2. Test that the end-effector servo can accurately rotate the end-effector. 3. Test the integration of the end-effector servo with the token picker master.
Servo – Board opener	1. Test board opener servo initialization. 2. Test that the board opener servo can accurately open and close the game board. 3. Test the integration of the board-opener servo with the master block
Token picker master	1. Test token picker master initialization. 2. Test that the token picker master can accurately manage the token picking process. 3. Test the integration of the token picker master with the vacuum pump, vacuum sensor, vacuum valve, and end-effector servo.
Token colour separator master	1. Test token colour separator master initialization. 2. Test that the token colour separator master can accurately sort tokens by colour. 3. Test the integration of the token colour separator master with the RGB sensor and flipper/solenoid control.
Motor master	1. Test motor master initialization and control. 2. Test that the motor master can accurately manage the X and Z-axis motors. 3. Test the integration of the motor master with motor control, encoder readout, end-switches, home switches, and PID calculations.
Flipper/Solenoid control	1. Test flipper/solenoid initialization and control. 2. Test that the flipper/solenoid can accurately separate tokens based on

	colour. 3. Test the integration of the flipper/solenoid control with the token colour separator master and RGB sensor.
Emergency Stop	1. Test emergency stop initialization and functionality. 2. Test that the emergency stop can halt all system operations safely and immediately. 3. Test the integration of the emergency stop with motor control, servo control, and error handling.
Dual-core communication	1. Test dual-core communication initialization and data exchange. 2. Test that Cortex-M7 and Cortex-M4 cores can communicate effectively and share tasks. 3. Test the integration of dual-core communication with the HSEM and other system modules.
Error handling	1. Test error handling implementation and functionality. 2. Test that the error handling system can detect, report, and recover from errors in various modules.

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/control them. 3. Test is CM4 can receive tasks from CM7 and if it responds accordingly while providing feedback on its status. 4. Test if the Cortex-M4 core can manage multiple tasks. 5. 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/control them. 3. Test is CM7 can send tasks to CM4 and check if it receives feedback about the status of the tasks. 4. Test if the Cortex-M7 core can manage multiple tasks. 5. 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 Task Manager module. 2. Test the Task Manager module to give tasks and receive feedback about their status 3. Test if the Task Manager can construct a queue of tasks and manage them in the correct order. 4. Test if the Task Manager module to prioritize tasks based on their importance. 5. Verify that the Task manager can give out the tasks that it receives from the Cortex-M7 core. 6. Verify that the Task Manager module can perform its functions within the expected time frame to ensure a smooth game flow. 7. Verify that the task manager can detect, report, and recover from errors.

Motor Controller	**1.** Test the initialization and configuration of the Motor Controller module for both motors. **2.** Verify that the Motor Controller module is set up to control the motors, encoders, and home/end switches. **3.** Test the Motor Controller module to move the motors in both X and Z directions. **4.** Verify that the Motor Controller module can control the motors to achieve the desired position and speed. **5.** Test the Motor Controller to accurately read the position from the encoders. **6.** Test the ability of the Motor Controller module to detect the home and end positions using switches. **7.** g. 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 Token Colour Separator Controller module. **2.** Verify that the Token Colour Separator Controller module is correctly set up to control the RGB sensor and the flipper/solenoid. **3.** Test the ability of the Token Colour Separator Controller module to accurately detect the colour of tokens (red and yellow). **4.** Test the ability of the Token Colour Separator Controller module to separate tokens based on their colour. **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 Token Picker Controller module. **2.** Verify that the Token Picker Controller module is correctly set up to control the vacuum pump, vacuum sensor, and vacuum valve. **3.** Test the ability of the Token Picker Controller module to accurately pick/release/transport tokens. **4.** Test the response time of the Token Picker Controller module during token picking and releasing operations. **5.** Verify that the Token Picker Controller module can detect, report, and recover from any errors.
User Detector	**1.** Test the initialization and configuration of the User Detector module. **2.** Verify that the User Detector module is correctly set up to read data from the photodiode circuit. **3.** Test the ability of the User Detector module to accurately detect the presence of a user. **4.** Test the response time of the User Detector module upon detecting a user. **5.** Test if multiple tokens are inserted in the same column (cheat move). **6.** Test if multiple tokens have been inserted in multiple columns (cheat move). **7.** Test if token is inserted in the wrong player state (cheat move). **8.** Verify that the User Detector module can send a signal to the Board Opener module. **9.** Verify that the User Detector module can detect, report, and recover from any errors.
Board Opener	**1.** Verify that the Task Manager can send commands for board opening and closing tasks to the Board Opener module. **2.** Verify that the Board Opener module can receive the signal from the User Detector module. **3.** 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, and setting the system clock.

Game controller	**1.** Verify that the Game Controller module is correctly set up to manage the overall game logic and flow. **2.** Test the Game Controller's ability to maintain and update the game state, including the board state and player turns. **3.** Test the Game Controller's ability to detect win, loss, or draw conditions. **4.** Verify that the Game Controller module can detect, report, and recover from any errors.
CM4 Task Generator	**1.** Test the initialization and configuration of the CM4 Task Generator 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 Game End Block module. **2.** Test the Game End Block's ability to detect and handle a win/lose/draw condition for either the human player or the robot player. **3.** Verify that the Game End Block 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 Controller's ability to transmit/receive data to and from external blocks. **3.** Verify that the UART Controller module can maintain the expected communication speed without losing data or causing errors. **4.** Verify that the UART Controller module can detect, report, and recover from any errors. **5.** Test the UART Controller's ability to handle interrupts for data transmission and reception.

Test Data	Test Set-Up

Happy-Path Test

The intended way of using the system.

Acceptance Test