SCARAB: Functional Specification

For submission to SETU

Author: Stuart Rossiter

Student Number: C00284845

Course: BSc (Hons.) Software Development

Supervisor: Joseph Kehoe

Contents

1. Introduction

The SCARAB (Save and Cartridge Aid Requiring Adapter Boards) is a diagnostic tool designed to check the overall health of retro game cartridges, and manage their saves through a GUI based PC program. Designed for the purpose of retro game preservation, the SCARAB will use an Arduino MEGA 2560 to interface with the cartridges, through cartridge port modules. These modules will be able to be swapped in and out, allowing for a wide variety of cartridge types, and future expansion.

1.1 Target Users

The SCARAB will be designed for retro gaming enthusiasts and preservationists. Retro gaming enthusiasts would want to ensure that their video games remain playable, and would be interested in backing up their save games in case something does go wrong. Preservationists keep all sorts of cartridges in working order, preserving the hardware with the software, so they would make use of the SCARAB's diagnostic capabilities.

1.2 Technologies

To serve as an interface between the cartridge port modules and the PC, a microcontroller will be required, in this case an Arduino. Due to the number of pins required for cartridge ports such as the Game Boy Advance, or the Nintendo 64, an Arduino MEGA 2560 would be ideal, due to the sheer number of digital I/O pins on the board. Alternate boards would be usable, but would require latches or multiplexing. For the GUI based PC program, Python would be sufficient, as it can interface with the Arduino directly over serial, and the Tkinter library provides GUI functionality. Various electronic components will also be required, listed below:

- 2x 8-bit transceivers, for 5v to 3.3v logic conversion.
- USB-C port. The Arduino has a max mA output, so a separate USB will be used to power the cartridges.
- Cartridge ports, to interface with the cartridges.
- Printed PCBs, for the SCARAB and its modules.

2. Functional Requirements

The functional requirements for the SCARAB are separated into 2 groups: Core and Non-Core.

2.1 Core Requirements

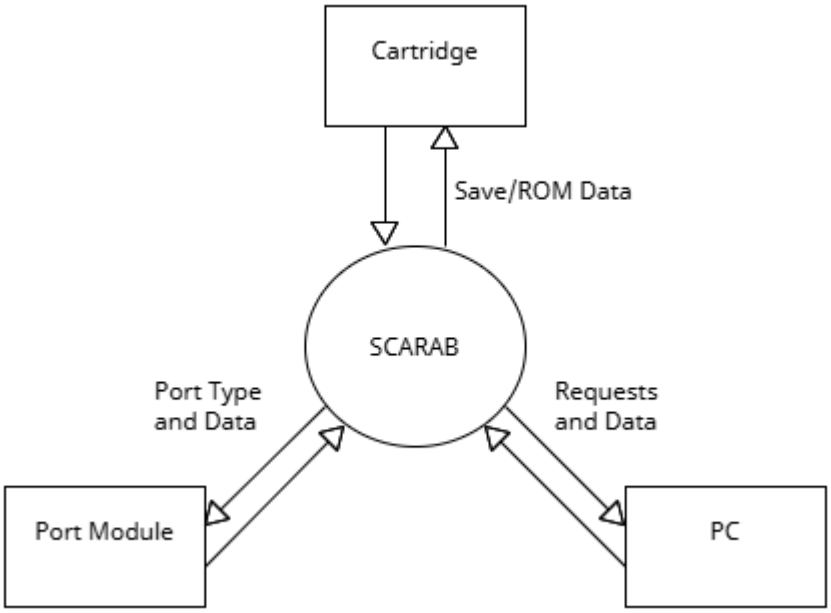
- Users should be able to run diagnostic tests on inserted cartridges.
- Users should be able to dump the save data from their cartridges.
- Save Data should be able to be flashed back to the cartridges.
- Users should be able to view and manage their various save files from the GUI program.
- Cartridge ports should be able to be removed, and swapped out with other cartridge port modules.

2.2 Non-Core Requirements

- Users should be able to view and edit their save files with a hex editor within the GUI program.
- Users can insert more than one cartridge module to the SCARAB at the same time.
- Users will be able to use 2 probes contained in the side of the SCARAB to check battery levels directly, with results displayed in the GUI PC program.

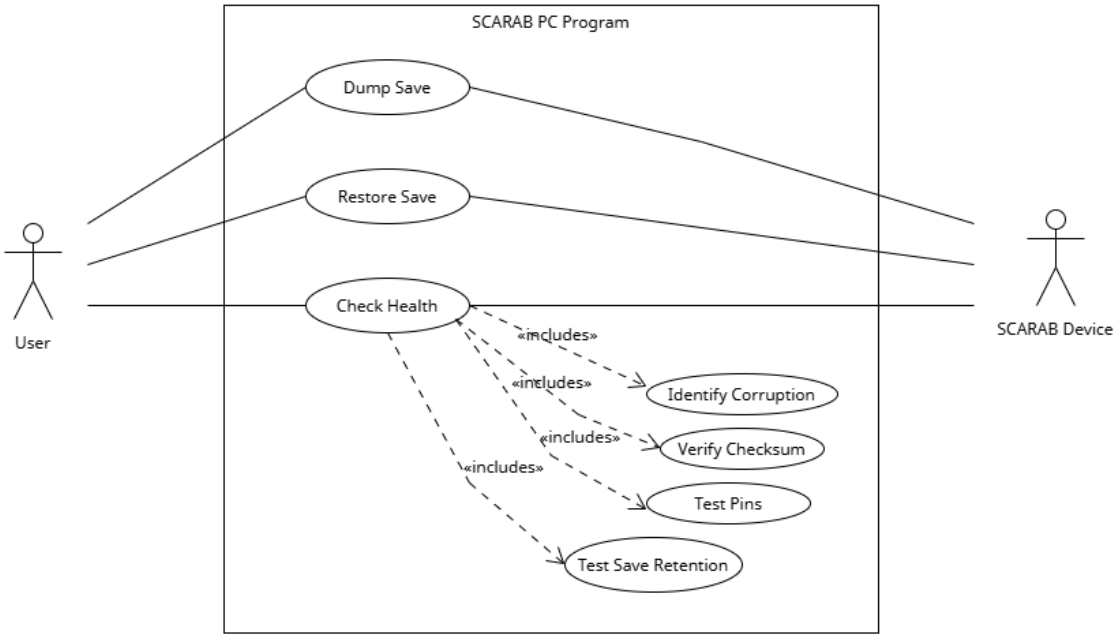
3. Context Diagram and Use Cases

3.1 Context Diagram

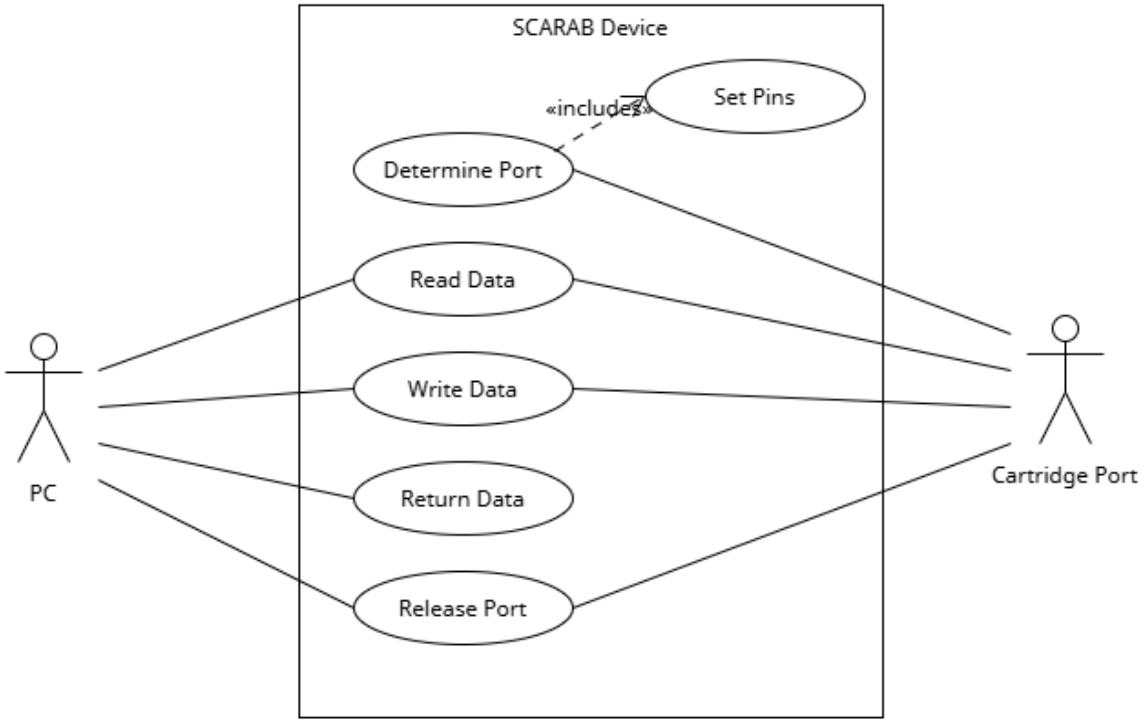


3.2 Use Case Diagrams

3.2.1 PC Program



3.2.2 SCARAB Device Use Case Diagram



3.3 Brief Use Cases

3.3.1 Dump Save

Actors: User, SCARAB Device

Description: This use case begins when the user wishes to dump the save file from the inserted cartridge. The user selects "Save Manager" from the sidebar. From there, the user selects the "Dump Save Data" option. The use case ends when the save data has been dumped from the cartridge.

3.3.2 Restore Save

Actors: User, SCARAB Device

Description: This use case begins when the user wishes to restore a dumped save to the inserted cartridge. The user selects "Save Manager" from the sidebar. From there, they select the desired save, and press "Restore Save Data". The use case ends when the save data has been restored to the cartridge.

3.3.3 Check Health

Actors: User, SCARAB Device

Description: This use case begins when the user wishes to check the health of the inserted cartridge. The user selects "Health Check" from the sidebar. The user then selects the tests they wish to run, and clicks "Run Test(s)". This use case ends when the results of the tests are returned to the user.

3.3.4 Identify Corruption

Actors: User, SCARAB Device
Description: This use case begins when the user wishes to identify any corrupted data within a cartridge/memory card. The user selects "Health Check" from the sidebar. The user then selects "Identify Corrupted Blocks", and clicks "Run Test(s)". This use case ends when the result of the test is returned to the user.

3.3.5 Verify Checksum

Actors: User, SCARAB Device
Description: This use case begins when the user wishes to verify the checksum of a cartridge ROM against a known dump. The user selects "Health Check" from the sidebar. The user then selects "Verify Checksum", and clicks "Run Test(s)". This use case ends when the result of the test is returned to the user.

3.3.6 Test Pins

Actors: User, SCARAB Device
Description: This use case begins when the user wishes to verify the checksum of a cartridge ROM against a known dump. The user selects "Health Check" from the sidebar. The user then selects "Verify Checksum", and clicks "Run Test(s)". This use case ends when the result of the test is returned to the user.

3.3.7 Test Save Retention

Actors: User, SCARAB Device
Description: This use case begins when the user wishes to test the save retention of the inserted cartridge. The user selects "Health Check" from the sidebar. The user then selects "Test Save Retention", and clicks "Run Test(s)". This use case ends when the result of the test is returned to the user.

3.3.8 Determine Port

Actors: Cartridge Port
Description: This use case begins when a new cartridge port is inserted. The SCARAB probes the inserted board to determine what cartridge port the board holds. This use case ends when the Arduino has determined the port type.

3.3.9 Set Pins

Actors: None
Description: This use case begins when the SCARAB determines a new port type. The SCARAB determines which pins to set as INPUT and OUTPUT. This use case ends when the pins are correctly set.

3.3.10 Read Data

Actors: PC, Cartridge Port
Description: This use case begins when the SCARAB gets a read request from the PC. The SCARAB reads data from the cartridge. This use case ends when the requested data has been read.

3.3.11 Write Data

Actors: PC, Cartridge Port
Description: This use case begins when the SCARAB gets a write request from the PC. The SCARAB receives the data and places it in a buffer. This use case ends when the requested data has been written to the cartridge.

3.3.12 Return Data

Actors: PC
Description: This use case begins when the SCARAB needs to send data to the PC. This use case ends when the SCARAB has sent the data to the PC.

3.3.13 Release Port

Actors: PC, Cartridge Port
Description: This use case begins when the PC sends a request to the SCARAB to release the cartridge port. The SCARAB cuts power to the port and disables all pins. This use case ends when the SCARAB sends a confirmation to the PC.

3.4 Detailed Use Cases

3.4.1 Dump Save

Actors: User, SCARAB Device	
Difficulty:	★★★★☆
Pre-conditions:	The SCARAB is connected to a PC with both USB cables. The PC is running the SCARAB GUI program. A cartridge port module is inserted into the SCARAB. A cartridge is inserted into the SCARAB's cartridge port.
Main Success Scenario:	1. The user opens the sidebar and selects the "Save Manager" option. 2. The program takes the user to the "Save Management" screen. 3. The user selects "Dump Save Data" from the main window. 4. The program prompts the user to confirm the dump, displaying the inserted cartridge name and image. 5. The user clicks "Dump Save", and the SCARAB begins dumping the cartridge's save file.
Post-conditions:	The SCARAB has successfully dumped the save file of the inserted cartridge to a new file on the PC.
Alternate Step(s):	4a. The program prompts the user that the current cartridge does not store save data. 5a. The user cancels the dump.

3.4.2 Restore Save

Actors: User, SCARAB Device	
Difficulty:	★★★★☆
Pre-conditions:	The SCARAB is connected to a PC with both USB cables. The PC is running the SCARAB GUI program. A cartridge port module is inserted into the SCARAB. A cartridge is inserted into the SCARAB's cartridge port.
Main Success Scenario:	1. The user opens the sidebar and selects the "Save Manager" option. 2. The program takes the user to the "Save Management" screen. 3. The user selects "Restore Save Data" from the main window. 4. The program prompts the user to select a save file, displaying the dumped saves compatible with the inserted cartridge. 5. The user selects the save file they wish to restore, and clicks "Restore Save". 6. The SCARAB begins restoring the save file to the cartridge.
Post-conditions:	The SCARAB has successfully restored the save data to the inserted cartridge.
Alternate Step(s):	4a. The program prompts the user that the current cartridge does not store save data. 4b. The program prompts the user that there are no supported save files present for the current cartridge. 5a. The user backs out of the save file selection screen.

3.4.3 Check Health

Actors: User, SCARAB Device	
Difficulty:	★★★★★
Pre-conditions:	The SCARAB is connected to a PC with both USB cables. The PC is running the SCARAB GUI program. A cartridge port module is inserted into the SCARAB. A cartridge is inserted into the SCARAB's cartridge port.
Main Success Scenario:	1. The user opens the sidebar and selects the "Health Manager" option. 2. The program takes the user to the "Health Management" screen. 3. The user selects which tests they wish to perform on the cartridge. 4. The user clicks "Run Test(s)". 5. The program brings the user to the testing screen, where it warns the user not to remove the cartridge during testing, while providing progress updates. 6. The SCARAB performs the test(s) on the cartridge.
Post-conditions:	The user can see the results of the test(s).
Alternate Step(s):	3a. The user clicks "Run All Tests".

3.4.4 Identify Corruption

Actors: User, SCARAB Device	
Difficulty:	★★★★☆
Pre-conditions:	The SCARAB is connected to a PC with both USB cables. The PC is running the SCARAB GUI program. A cartridge port module is inserted into the SCARAB. A cartridge is inserted into the SCARAB's cartridge port.
Main Success Scenario:	1. The user opens the sidebar and selects the "Health Manager" option. 2. The program takes the user to the "Health Management" screen. 3. The user selects the "Identify Corruption". 4. The user clicks "Run Test(s)". 5. The program brings the user to the testing screen, where it warns the user not to remove the cartridge during testing, while providing progress updates. 6. The SCARAB checks the ROM and save data for any corruption.
Post-conditions:	The user can see the results of the test.
Alternate Step(s):	

3.4.5 Verify Checksum

Actors: User, SCARAB Device	
Difficulty:	★★★☆☆
Pre-conditions:	The SCARAB is connected to a PC with both USB cables. The PC is running the SCARAB GUI program. A cartridge port module is inserted into the SCARAB. A cartridge is inserted into the SCARAB's cartridge port.
Main Success Scenario:	1. The user opens the sidebar and selects the "Health Manager" option. 2. The program takes the user to the "Health Management" screen. 3. The user selects the "Verify Checksum". 4. The user clicks "Run Test(s)". 5. The program brings the user to the testing screen, where it warns the user not to remove the cartridge during testing, while providing progress updates. 6. The SCARAB calculates the checksum of the cartridges rom, and checks it against a known valid checksum.
Post-conditions:	The user can see the results of the test.
Alternate Step(s):	

3.4.6 Test Pins

Actors: User, SCARAB Device	
Difficulty:	★★☆☆☆
Pre-conditions:	The SCARAB is connected to a PC with both USB cables. The PC is running the SCARAB GUI program. A cartridge port module is inserted into the SCARAB. A cartridge is inserted into the SCARAB's cartridge port.
Main Success Scenario:	1. The user opens the sidebar and selects the "Health Manager" option. 2. The program takes the user to the "Health Management" screen. 3. The user selects the "Test Pins". 4. The user clicks "Run Test(s)". 5. The program brings the user to the testing screen, where it warns the user not to remove the cartridge during testing, while providing progress updates. 6. The SCARAB reads areas of ROM multiple times and compares results.
Post-conditions:	The user can see the results of the test.
Alternate Step(s):	

3.4.7 Test Save Retention

Actors: User, SCARAB Device	
Difficulty:	★★★★☆
Pre-conditions:	The SCARAB is connected to a PC with both USB cables. The PC is running the SCARAB GUI program. A cartridge port module is inserted into the SCARAB. A cartridge is inserted into the SCARAB's cartridge port.
Main Success Scenario:	1. The user opens the sidebar and selects the "Health Manager" option. 2. The program takes the user to the "Health Management" screen. 3. The user selects the "Test Save Retention". 4. The user clicks "Run Test(s)". 5. The program brings the user to the testing screen, where it warns the user not to remove the cartridge during testing, while providing progress updates. 6. The SCARAB loads the cartridge with a predetermined save, cuts power from the cartridge for 30 seconds, and checks the save on the cartridge against the original.
Post-conditions:	The user can see the results of the test.
Alternate Step(s):	

3.4.8 Determine Port

Actors: Cartridge Port	
Difficulty:	★★★☆☆
Pre-conditions:	The SCARAB is powered. A cartridge port has just been inserted into the SCARAB device.
Main Success Scenario:	1. The SCARAB detects that a new cartridge port has been inserted. 2. The SCARAB tests identifier lines on the module, and checks the returned voltages. 3. The voltages are checked against a table, to determine the inserted module type.
Post-conditions:	The SCARAB knows which cartridge port module has been inserted.
Alternate Step(s):	

3.4.9 Set Pins

Actors: None	
Difficulty:	★☆☆☆☆
Pre-conditions:	The SCARAB is powered. The SCARAB has just determined the type of the inserted port.
Main Success Scenario:	1. The SCARAB checks the correct pin layout for the inserted port module. 2. The SCARAB sets the appropriate pins to input or output.
Post-conditions:	The SCARAB has set the pins correctly.
Alternate Step(s):	

3.4.10 Read Data

Actors: PC, Cartridge Port	
Difficulty:	★★☆☆☆
Pre-conditions:	The SCARAB is plugged into the PC. The SCARAB has received a read request from the PC.
Main Success Scenario:	1. The SCARAB receives the address to read from, and the size of the read. 2. The SCARAB reads the data from the cartridge into a buffer.
Post-conditions:	The SCARAB has read the data at the requested address
Alternate Step(s):	3. The SCARAB sends the full buffer to the PC, and continues reading.

3.4.11 Write Data

Actors: PC, Cartridge Port	
Difficulty:	★★★☆☆
Pre-conditions:	The SCARAB is plugged into the PC. The SCARAB has received a write request from the PC.
Main Success Scenario:	1. The SCARAB receives the request data. 2. The SCARAB loads the writable data into a buffer. 3. The SCARAB writes the data to the cartridge, beginning at the specified address.
Post-conditions:	The SCARAB has written the data to the cartridge.
Alternate Step(s):	

3.4.12 Return Data

Actors: PC	
Difficulty:	★☆☆☆☆
Pre-conditions:	The SCARAB is plugged into the PC. The SCARAB needs to send data to the PC program.
Main Success Scenario:	1. The SCARAB performs a handshake with the PC program. 2. The SCARAB sends the data with packet headers along the serial line to the PC. 3. The SCARAB receives a response confirming transfer.
Post-conditions:	The SCARAB's data has been transferred to the PC.
Alternate Step(s):	2a. The PC has not responded to the handshake, so the SCARAB aborts the process.

3.4.13 Release Port

Actors: PC, Cartridge Port	
Difficulty:	★★★☆☆
Pre-conditions:	The SCARAB is plugged into the PC. The SCARAB has detected a port is being removed.
Main Success Scenario:	1. The SCARAB sets all pins to INPUT, essentially disabling them. 2. The SCARAB cuts power to the cartridge port. 3. The SCARAB sends a "Port Removed" response to the PC.
Post-conditions:	The port module connector is no longer receiving power or data.
Alternate Step(s):	

4. FURPS

4.1 Functionality

The SCARAB's ability to read save data and test cartridges will be its main measure. It should be able to interact with a variety of cartridges, and detect issues with known damaged test cartridges. The SCARAB puts emphasis on modularity, so it must also be able to support a wide range of cartridges and cartridge port modules. The modules need to be able to be swapped seamlessly, with detections in place for removal and insertion of both cartridges and modules. The final iteration should have at least 5 different modules.

4.2 Usability

The SCARAB needs to be well documented, with graphical guides within the GUI program for module insertion. Given a SCARAB with no inserted port, a port module, an appropriate cartridge, and an installed SCARAB GUI, users new to the system should be able to begin a health check or save data dump within 60 seconds.

4.3 Reliability

The SCARAB will need to be very reliable, as malfunctions could cause damage to the cartridges which it is trying to preserve. Confirmed working cartridges will need to pass all tests 99% of the time. Cartridges with known defects, such as dirty pins or a dead internal battery, will also need to fail at minimum 1 test 95% of the time. Any time a module or cartridge has been removed, the SCARAB PC program should detect it within 1 second.

4.4 Performance

The SCARAB will need to be reasonably quick, within reason. The absolute maximum baud rate of the Arduino Mega's serial line is 1 Megabit per second. Given constant data flow, that's ~80 seconds for an 8MB N64 ROM dump for checksum calculation. Leaving room for pauses in data flow, the SCARAB should be able to transfer the contents of The Legend of Zelda: Ocarina of Time (32MB) within 6 minutes. Below is a table showing ROM sizes, Minimum Possible Time (MPT), and Acceptable Transfer Time (ATT).

Name	Console	Size	MPT	ATT
Super Mario Bros. 3	NES	384KB	3.1s	5s
Pokémon Red & Blue	GB	1MB	8s	12s
Super Metroid	SNES	3MB	24s	30s
Pokémon Crystal	GBC	2MB	16s	20s
The Legend of Zelda: Ocarina of Time	N64	32MB	5 Min 20s	6 Min
Pokémon Ruby / Sapphire	GBA	16MB	2 Min 30s	3 Min

4.5 Supportability

The SCARAB is designed with supportability in mind. New firmware should be able to be flashed to the microcontroller (Arduino, in this case) via the connecting cable. Modular cartridge ports should allow for future expansion to the list of compatible cartridges.

5. Metrics

There are certain criteria that the SCARAB must meet in order to be deemed a success. They are as follows:

- Users from the target demographic should be able to go from an unplugged SCARAB device and an installer for the PC program on the PCs desktop, to checking the health of an inserted cartridge, within 60 seconds.
- The SCARAB device should be able to detect the inserted module. The insertion or removal of modules should be detected by the SCARAB device and PC program within 0.5 seconds.
- The insertion or removal of cartridges into the modules should be detectable by the SCARAB and PC program within 1 second.
- The SCARAB device should be able to fully dump the save file of a cartridge within 5 seconds. The same 5 second timeframe applies to restoring a save file.
- The SCARAB must identify known issues in at least 95% of cases. Known working cartridges must not have any issues identified at least 99% of the time.
- The SCARAB should be able to identify whether the cartridge supports on-cartridge save data.
- The SCARAB should be able to dump ROMs for checksum validation as outlined in the table above (see 4.4 Performance).

6. Project Plan

Each use case of the SCARAB will need to be implemented in order of importance. Below is a timeline of the expected development of the SCARAB.