Handover Report

Project Puzzlebox

Thomas in 't Anker, Loek Le Blansch, Lars Faase, Elwin Hammer

Version 0.0, 2024-04-01: draft

Table of Contents

List of Figures

List of Tables

1. Introduction

This is an (at times slightly informal) document that summarizes how the 23-24 run of this project went. We found the previous handover documents to be unhelpful when determining the 'actual' state of the project in the first few weeks, and felt they did not address the pitfalls of this project.

The team of year 2023-2024 consisted of only software students (see Table 1), meaning no hardware was developed in this year. The goal of this year is to create a software framework which can be used to implement new puzzles and to make the development process of these puzzles easier, and allow the entire software stack to be ported to the the hardware designed by the 22-23 group.

Previous years' groups have put their predecessor's documents inside their own project folder, which has resulted in what we called the 'Russian doll folder structure'. Loek Le Blansch has separated out each year's project folder ('master file'), and is hosting these on https://media.pipeframe.xyz/puzzlebox. This directory is also mountable as a read-only WebDAV share on Windows, MacOS and Linux (using davfs2), and does not require credentials to log in. Please note that this is very much unofficial, and is not managed or endorsed by Avans. Loek Le Blansch is the contact for removal or transfer of these files.

2. Group history

Year	Name	Study path
19-20	Daniël Janssen	Software
	Dion Legierse	Software
	Jop van Laanen	Hardware
	Max van den Heijkant	Software
20-21	Joost van Wiechen	Hardware
	Justin Maas	Software
	Merel Creemers	Hardware[1]
	Vincent Lengowski	Hardware
21-22	Alex van Kuijk	Hardware
	Jef Baars	Software
	Julian de Bruin	Software
	Lucas van Gastel	Software
	Toon Rockx	Hardware
22-23	Frank Bekema	Hardware
	Jasper Gense	Hardware
23-24	Elwin Hammer	Software
	Lars Faase[2]	Software
	Loek Le Blansch	Software
	Thomas in 't Anker	Software

Table 1. Project group composition

3. Project state

The current project state is as follows:

- No new hardware has been designed or developed this year
- The software was completely revised, now consisting of
    - a puzzle bus driver (pbdrv)

- a main controller
- a simple CLI application
- two puzzle modules ('Vault' and 'NeoTrellis') integrated using the puzzle bus driver

Functionality:

- The main controller (a RPI Pico W) can interact with the different puzzle modules using a central shared I^2C bus (referred to as the 'puzzle bus')
- The main controller is able to find new puzzle modules on startup, and does not check for new modules afterwards.
- A simple CLI application has been developed, which can communicate with the main controller through a TCP connection and allows control over various aspects of the puzzle box using simple commands.

Documentation:

- These project documents
- Detailed usage and API documentation for all software modules [1]

4. Incidents

During this year's run of the project, we encountered several difficuties we feel need to be addressed in order to be mitigated in future runs of the project. We recommend that these incidents are analyzed by future project groups and incorporated into the risk analysis section of future project plan documents.

4.1. Documentation

We spent too much time working on documentation at the start of the project. Make sure you set clear deadlines for documentation, and try not to spend too much time on review procedures, as these cost a lot of time.

Our project documentation was originally written in Microsoft Word, but we later transferred all documentation to AsciiDoc because of issues where OneDrive would roll back changes. If possible, use a documentation system or format that allows using an external version control system like Git to avoid losing content. This is also the reason why our documents may contain formatting/style errors.

4.2. Misconceptions

Make sure to know what you are developing and do some research beforehand, to make sure you have a complete picture about what you are using. Sounds stupid, but it happened for multiple project attempts, and caused time loss. This also includes when you want to use documentation of previous years: go through the documentation and verify it on the lowest possible level for the same reason as previously mentioned.

4.3. I^2C

I^2C is used because it is widely available and easy to implement. We strongly recommend 3rd year software students to refresh their knowledge on I^2C before making major design decisions that rely on their conception of how to I^2C bus works.

Please note the following differences between I^2C devices:

- Regular I^2C slave peripherals are allowed on the puzzle bus, and can be connected to the puzzle bus as long as they do not cause issues with addressing.
- I^2C master controllers must have hardware support for a multi-master hardware configuration (i.e. support bus arbitration on a hardware level). Arbitration support is required to pre-

vent message corruption or electrical shorts in a multi-master setup. Multi-master controllers may also be connected to the puzzle bus, but only as long as they only interact with regular I²C slave devices.
- I²C multi-master controllers that are slave-addressable in master mode are the only kind of I²C controller suitable for use in puzzle modules. Microcontrollers with 2 I²C peripherals on the same bus (one in master mode, one in slave mode) can also be used to achieve the same effect.

The RP2040 supports multi-master, but is not addressable as a slave in master mode. This was mitigated using a workaround (see RP2040 I²C limitations).

4.4. Development hardware availability

When choosing or using specific chips or development boards, make sure to include research on the product lifecycle. Choosing boards/chips that have planned long term support makes it easier for the next project team to order and use the same chips/boards instead of having to find new ones.

Due to a lack of foresight, only 2 Picos were ordered this year, which caused unoptimal workload spread during the last weeks of the project. Because of this, we also strongly recommend making enough development boards available for multiple people to develop using the same setup. Note that the RPI Pico is a special case, as it requires another Pico for debugging, effectively requiring double the amount of hardware to support developers.

Due to a misunderstanding, we also thought our development boards went missing somewhere during week 13. Double-checking if project materials were actually stolen, or making clear where the materials are stored by sending an image of its location could have easily avoided this from happening; make sure to do either.

4.5. Auxiliary workarounds and technical limitations

This section details workarounds that were implemented instead of being fixed due to time constraints or project scope. Workarounds that should be removed are marked with FIXME: comments referring to one of the workarounds mentioned in this section.

RP2040 I²C limitations
- The RP2040 is not slave-addressable while in master mode. A workaround that uses both I²C peripherals simultaniously was written to work around this issue.

Memory management on Arduino

The Arduino's built-in memory management functions do not seem to work properly. The FreeRTOS heap 4 memory management functions are used on the puzzle modules instead. FreeRTOS does not have an implementation of the realloc() function.

- mpack's writer API cannot be used with a writer initialized using the mpack_writer_init_growable function on Arduino-based puzzle modules. The mpack_writer_init function is used with a static size buffer instead.

5. Recommendations

This section details our recommendations on course of action for future project groups.

5.1. Imperatives

The following points must not be dismissed by future project groups, as they are critical for project success:

- The 22-23 design document already mentions that the application of the I²C bus is in a multi-master configuration, but does not mention that this only works when pull-up resistors

are used on the SCL and SDA lines. The pull-up resistors are required, as omitting them makes the bus arbitration process very inconsistent which causes frames to be dropped entirely.
- Start creating prototypes as early as possible; this benefits the project in the long run, as you have already shown that certain parts of the project are already working and "only" need to be integrated.
- The Atmega328P and ATmega2560 MCUs are both sufficient for the puzzle modules as they have enough I/O, mutli-master hardware support, and the ability to be addressed as a slave while being in master mode.
- The RPI Pico (and Pico W)'s I^2C peripheral supports multi-master, but does not support being addressed as a slave while in master mode. This is required for puzzle bus integration, and was mitigated using a workaround (see RP2040 I^2C limitations).

5.2. Other suggestions

These points are suggestions for future project groups. While we do not think these are critical to project success, we still think they are important to mention.

- The hardware design from the year 22-23 should be implemented.
- The original game rules are described in a separate document from the year 20-21.
- The RPI Pico W has programmable I/O modules. Due to time constraints, we did not research if these modules can be used to create a custom I^2C peripheral (and driver) that allows multi-master communication while still being addressable as a slave. If this is possible, the RPI Pico W could be used without the use of workarounds.

Appendix A: References

[1] L. L. Blansch, E. Hammer, L. Faase, and T. in ’t Anker, “puzzlebox Doxygen documentation,” 2024. [Online]. Available: https://media.pipeframe.xyz/puzzlebox/23-24/doxygen.

Appendix B: Glossary

RPI
Raspberry Pi

Main board
The main board is the PCB on the bottom of the puzzle box, this communicates with the puzzles and the bomb

Puzzle box hub
The puzzle box hub communicates with the puzzle box and the bomb, as well as helps with configuring them

SID
Security identifiers

game operator
Person who organizes a puzzle box play session

[1] The handover report from 20-21 mentions: *"Het frame zelf is niet gelukt om te realiseren, omdat er communicatie tussen het projectgroep en de CMD-student uit het niets is verdwenen"*. Merel Creemers was introduced as a hardware-student in the project plan, but is no longer mentioned in the handover report, which may indicate that they were removed from the project group. I am unsure if they were a hardware student that worked on the PCBs or a CMD student working on the puzzle box chassis.
[2] Lars Faase was removed from the project group on 2024-06-03 following complaints about the lack of communication, and lack of motivation