Assignment-2 Report

Anshkirat Singh Dhanju – 21355130
Michael Cummins - 17332852
Sean Fahey - 21360313
Cian Donovan - 16336725
Oisín Gartlan - 19335226

Introduction:

Our task was to create a playable game that teaches the user how to use Morse Code. To do this, we implemented both C and ARM code on the Raspberry Pi Pico.

This game allows a user to interact with the Pico by pressing the GP21 button, either holding it to produce a "dash" or pressing it quickly to produce a "dot". This input system is used to play all four levels of the game.

The application opens with a welcome message and instructions on how the user can interact with and play the game. Here, the user can select the level they wish to play. These include the following:

Level 1: The player is provided with individual characters and their respective Morse code, and must repeat these by pressing the GP21 button.

Level 2: The player is provided with individual characters but no Morse code, and must input these by pressing the GP21 button.

Level 3: The player is provided with words and their respective Morse code, and must repeat these by pressing the GP21 button.

Level 4: The player is provided with words but no Morse code, and must repeat these by pressing the GP21 button.

For each level, the player starts with three lives. Each time the player fails to input the correct code they lose a life, and each time they are successful a life is added up to a maximum of three lives. The player's lives are shown by the colour of the RGB LED: Green for 3, Yellow for 2, Orange for 1, and when all lives are lost the light goes Red. The player progresses to the next level after 5 sequences are completed correctly sequentially.

To complete this task, we set up a Gitlab repository where each member could contribute to the code as needed. We also met up weekly, or more often as was necessary. At these meetings we decided what needed to be done and split up the tasks according to the skills of each member. The final application is a mixture of C and ARM code.

Workflow:

In the first week of the group project, our group was able to reach out to each other and with everyone's mutual agreement the following roles were decided:

Project Workflow Owner – Anshkirat Singh Dhanju
Gitlab Workflow Owner – Cian Donovan
Project Documentation Owner – Oisín Gartlan
Project Demonstration Owner – Michael Cummins
Project Code Owner – Sean Fahey

Division of tasks (including the coding) were also done in the initial week itself. In the initial stages of our project, our group agreed that we would try to work as collaboratively as possible, splitting up tasks as necessary and uploading to the repository often. We chose this approach so we could track the progress of our group easily and so that no one individual had to complete large tasks alone, which allowed for quicker and smooth implementation.

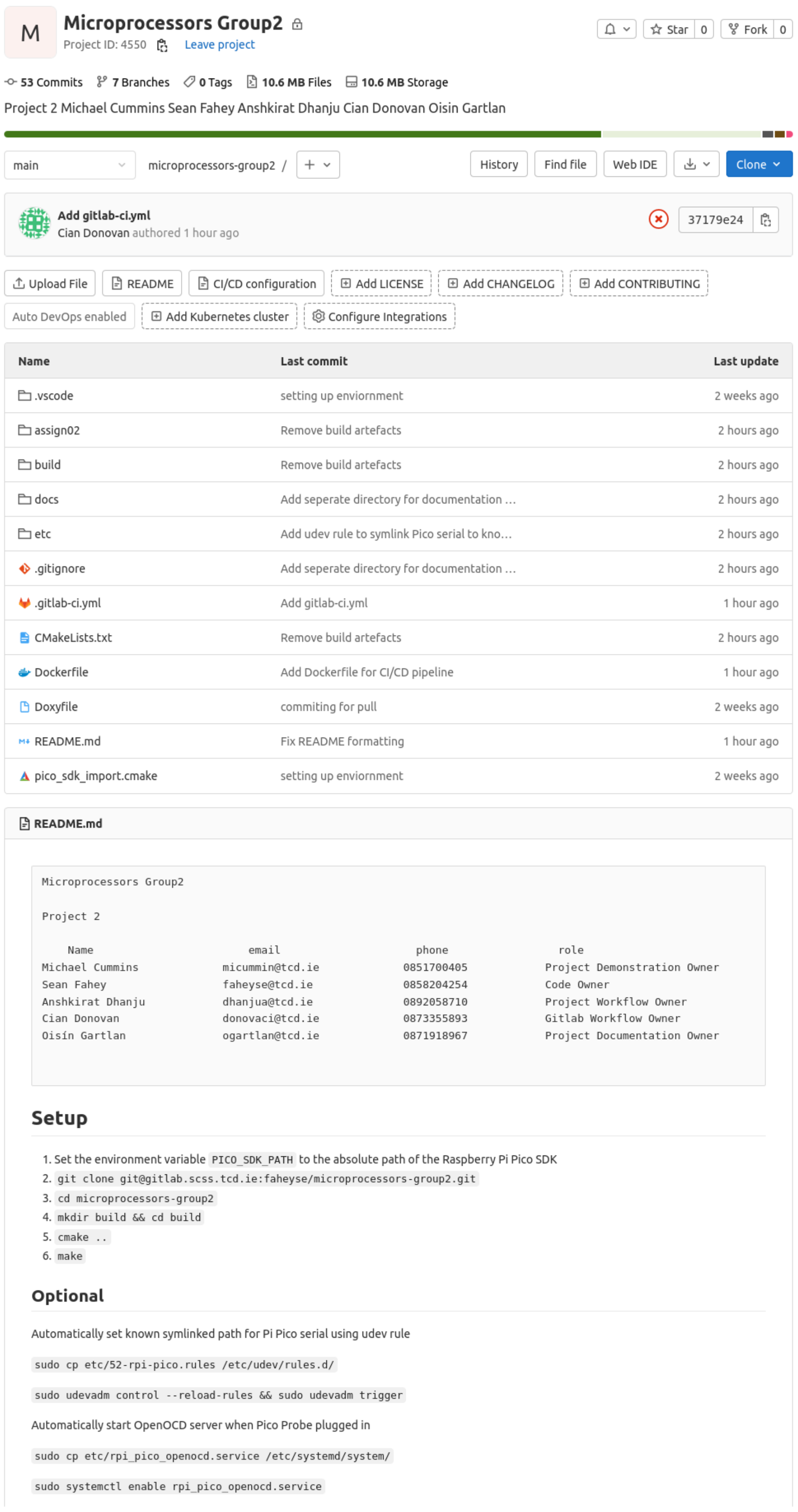
As the project workflow owner, Anshkirat took regular updates from everyone about the progress of the group as a whole. Progress update meetings were also organised (usually on zoom).
Since Sean was the code owner, he took over the major part of the code, however, certain parts of the code were done by the remaining members as well (as seen in the commit history in the next section). We held meetings, either in person or online, frequently so that any issues faced could be solved quickly.
For the purpose of collaborating for coding, we used Gitlab and each member had his own branch where they did their part of the code and created merge requests which were managed by the Gitlab workflow owner.
Our group used WhatsApp as the main medium of communication.

Collaboration Management:

As mentioned in the previous section, for the purpose of collaboration we used Gitlab. Each member had their own branch apart from the main branch which was managed by the Gitlab workflow owner. Each member pushed their code to their own branch and created a merge request. The merge request would be accepted by the Gitlab workflow owner and the code would be merged to the main branch. Any conflicts were resolved at this point also.

Issues were also raised as errors were found, functionality was missing, or an easier approach to a certain part was found. Issues were resolved once the related code was fixed, pushed and merged to the main branch.
The main branch was managed both by the Gitlab workflow owner as well as the project code owner.

Following are the snapshots of our Gitlab repository and related Statistics:



Microprocessors Group2
Project ID: 4550 Leave project

53 Commits 7 Branches 0 Tags 10.6 MB Files 10.6 MB Storage

Project 2 Michael Cummins Sean Fahey Anshkirat Dhanju Cian Donovan Oisin Gartlan

main microprocessors-group2 /

Add gitlab-ci.yml
Cian Donovan authored 1 hour ago 37179e24

Name	Last commit	Last update
.vscode	setting up enviornment	2 weeks ago
assign02	Remove build artefacts	2 hours ago
build	Remove build artefacts	2 hours ago
docs	Add seperate directory for documentation ...	2 hours ago
etc	Add udev rule to symlink Pico serial to kno...	2 hours ago
.gitignore	Add seperate directory for documentation ...	2 hours ago
.gitlab-ci.yml	Add gitlab-ci.yml	1 hour ago
CMakeLists.txt	Remove build artefacts	2 hours ago
Dockerfile	Add Dockerfile for CI/CD pipeline	1 hour ago
Doxyfile	commiting for pull	2 weeks ago
README.md	Fix README formatting	1 hour ago
pico_sdk_import.cmake	setting up enviornment	2 weeks ago

README.md

```
Microprocessors Group2

Project 2

    Name                 email                phone           role
Michael Cummins      micummin@tcd.ie      0851700405      Project Demonstration Owner
Sean Fahey           faheyse@tcd.ie       0858204254      Code Owner
Anshkirat Dhanju     dhanjua@tcd.ie       0892058710      Project Workflow Owner
Cian Donovan         donovaci@tcd.ie      0873355893      Gitlab Workflow Owner
Oisín Gartlan        ogartlan@tcd.ie      0871918967      Project Documentation Owner
```

Setup

1. Set the environment variable `PICO_SDK_PATH` to the absolute path of the Raspberry Pi Pico SDK
2. `git clone git@gitlab.scss.tcd.ie:faheyse/microprocessors-group2.git`
3. `cd microprocessors-group2`
4. `mkdir build && cd build`
5. `cmake ..`
6. `make`

Optional

Automatically set known symlinked path for Pi Pico serial using udev rule

`sudo cp etc/52-rpi-pico.rules /etc/udev/rules.d/`

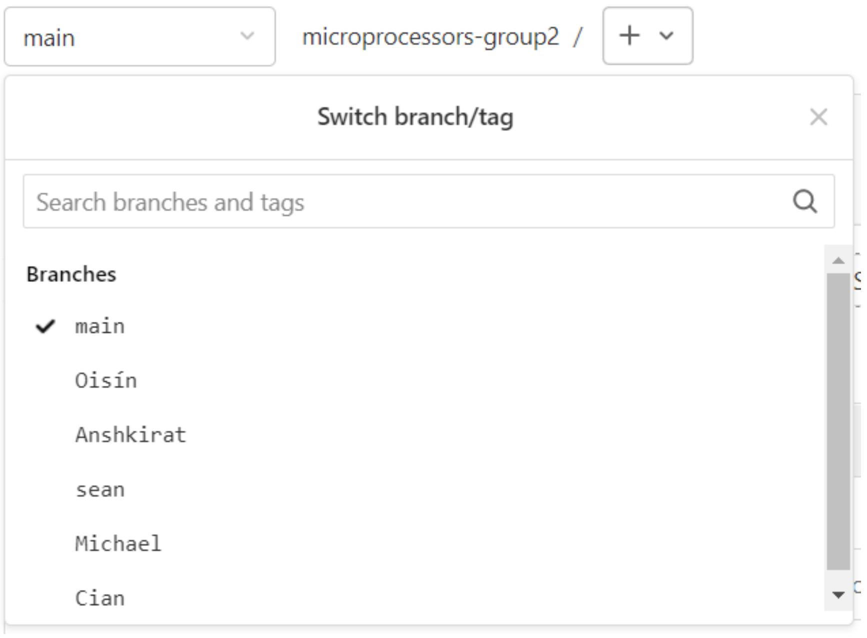
`sudo udevadm control --reload-rules && sudo udevadm trigger`

Automatically start OpenOCD server when Pico Probe plugged in

`sudo cp etc/rpi_pico_openocd.service /etc/systemd/system/`

`sudo systemctl enable rpi_pico_openocd.service`

Fig 1 – Repository Homepage



main
microprocessors-group2 /
+

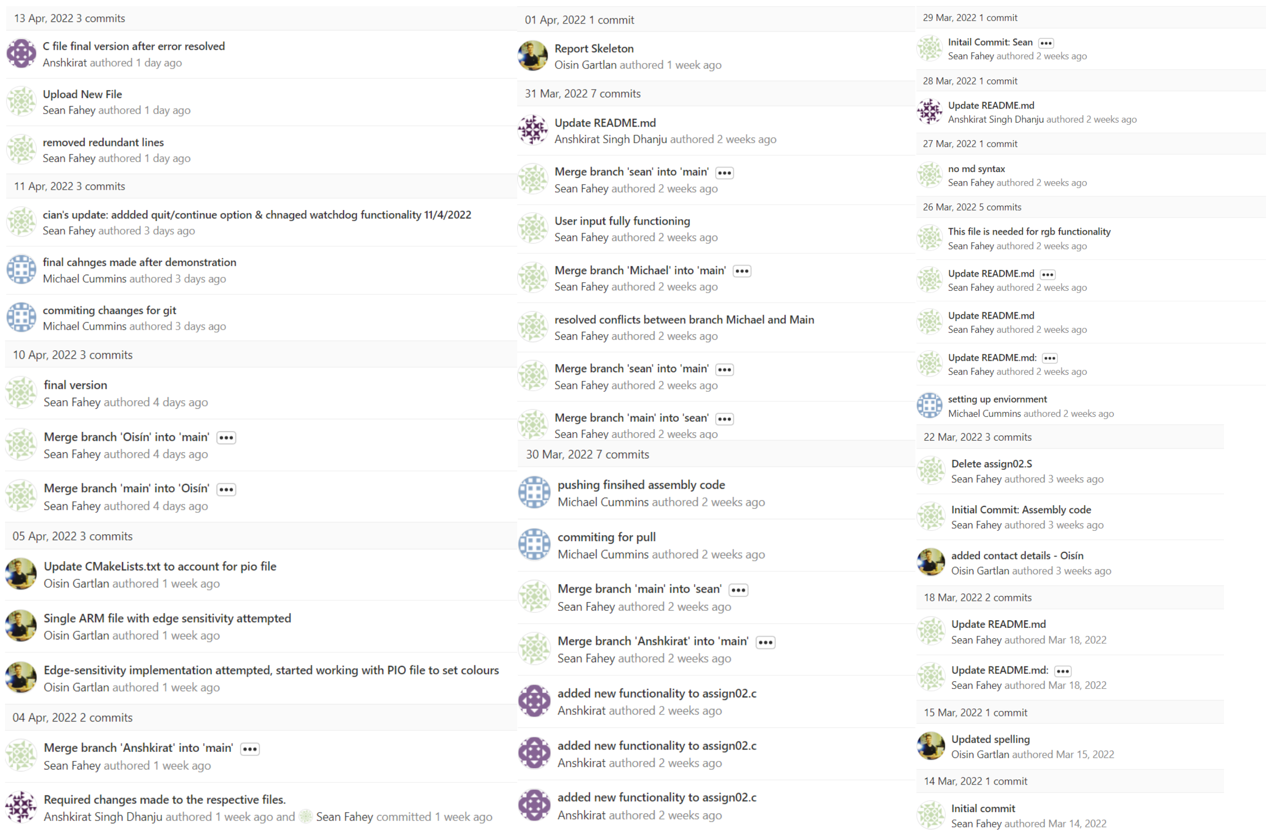
Switch branch/tag

Search branches and tags

Branches

- ✓ main
- Oisín
- Anshkirat
- sean
- Michael
- Cian

Fig 2 – Separate Branch for each person



13 Apr, 2022 3 commits

C file final version after error resolved
Anshkirat authored 1 day ago

Upload New File
Sean Fahey authored 1 day ago

removed redundant lines
Sean Fahey authored 1 day ago

11 Apr, 2022 3 commits

cian's update: addded quit/continue option & chnaged watchdog functionality 11/4/2022
Sean Fahey authored 3 days ago

final cahnges made after demonstration
Michael Cummins authored 3 days ago

commiting chaanges for git
Michael Cummins authored 3 days ago

10 Apr, 2022 3 commits

final version
Sean Fahey authored 4 days ago

Merge branch 'Oisín' into 'main' •••
Sean Fahey authored 4 days ago

Merge branch 'main' into 'Oisín' •••
Sean Fahey authored 4 days ago

05 Apr, 2022 3 commits

Update CMakeLists.txt to account for pio file
Oisin Gartlan authored 1 week ago

Single ARM file with edge sensitivity attempted
Oisin Gartlan authored 1 week ago

Edge-sensitivity implementation attempted, started working with PIO file to set colours
Oisin Gartlan authored 1 week ago

04 Apr, 2022 2 commits

Merge branch 'Anshkirat' into 'main' •••
Sean Fahey authored 1 week ago

Required changes made to the respective files.
Anshkirat Singh Dhanju authored 1 week ago and Sean Fahey committed 1 week ago

01 Apr, 2022 1 commit

Report Skeleton
Oisin Gartlan authored 1 week ago

31 Mar, 2022 7 commits

Update README.md
Anshkirat Singh Dhanju authored 2 weeks ago

Merge branch 'sean' into 'main' •••
Sean Fahey authored 2 weeks ago

User input fully functioning
Sean Fahey authored 2 weeks ago

Merge branch 'Michael' into 'main' •••
Sean Fahey authored 2 weeks ago

resolved conflicts between branch Michael and Main
Sean Fahey authored 2 weeks ago

Merge branch 'sean' into 'main' •••
Sean Fahey authored 2 weeks ago

Merge branch 'main' into 'sean' •••
Sean Fahey authored 2 weeks ago

30 Mar, 2022 7 commits

pushing finsihed assembly code
Michael Cummins authored 2 weeks ago

commiting for pull
Michael Cummins authored 2 weeks ago

Merge branch 'main' into 'sean' •••
Sean Fahey authored 2 weeks ago

Merge branch 'Anshkirat' into 'main' •••
Sean Fahey authored 2 weeks ago

added new functionality to assign02.c
Anshkirat authored 2 weeks ago

added new functionality to assign02.c
Anshkirat authored 2 weeks ago

added new functionality to assign02.c
Anshkirat authored 2 weeks ago

29 Mar, 2022 1 commit

Initail Commit: Sean •••
Sean Fahey authored 2 weeks ago

28 Mar, 2022 1 commit

Update README.md
Anshkirat Singh Dhanju authored 2 weeks ago

27 Mar, 2022 1 commit

no md syntax
Sean Fahey authored 2 weeks ago

26 Mar, 2022 5 commits

This file is needed for rgb functionality
Sean Fahey authored 2 weeks ago

Update README.md •••
Sean Fahey authored 2 weeks ago

Update README.md
Sean Fahey authored 2 weeks ago

Update README.md: •••
Sean Fahey authored 2 weeks ago

setting up enviornment
Michael Cummins authored 2 weeks ago

22 Mar, 2022 3 commits

Delete assign02.S
Sean Fahey authored 3 weeks ago

Initial Commit: Assembly code
Sean Fahey authored 3 weeks ago

added contact details - Oisín
Oisin Gartlan authored 3 weeks ago

18 Mar, 2022 2 commits

Update README.md
Sean Fahey authored Mar 18, 2022

Update README.md: •••
Sean Fahey authored Mar 18, 2022

15 Mar, 2022 1 commit

Updated spelling
Oisin Gartlan authored Mar 15, 2022

14 Mar, 2022 1 commit

Initial commit
Sean Fahey authored Mar 14, 2022

Fig 3 - Commit History

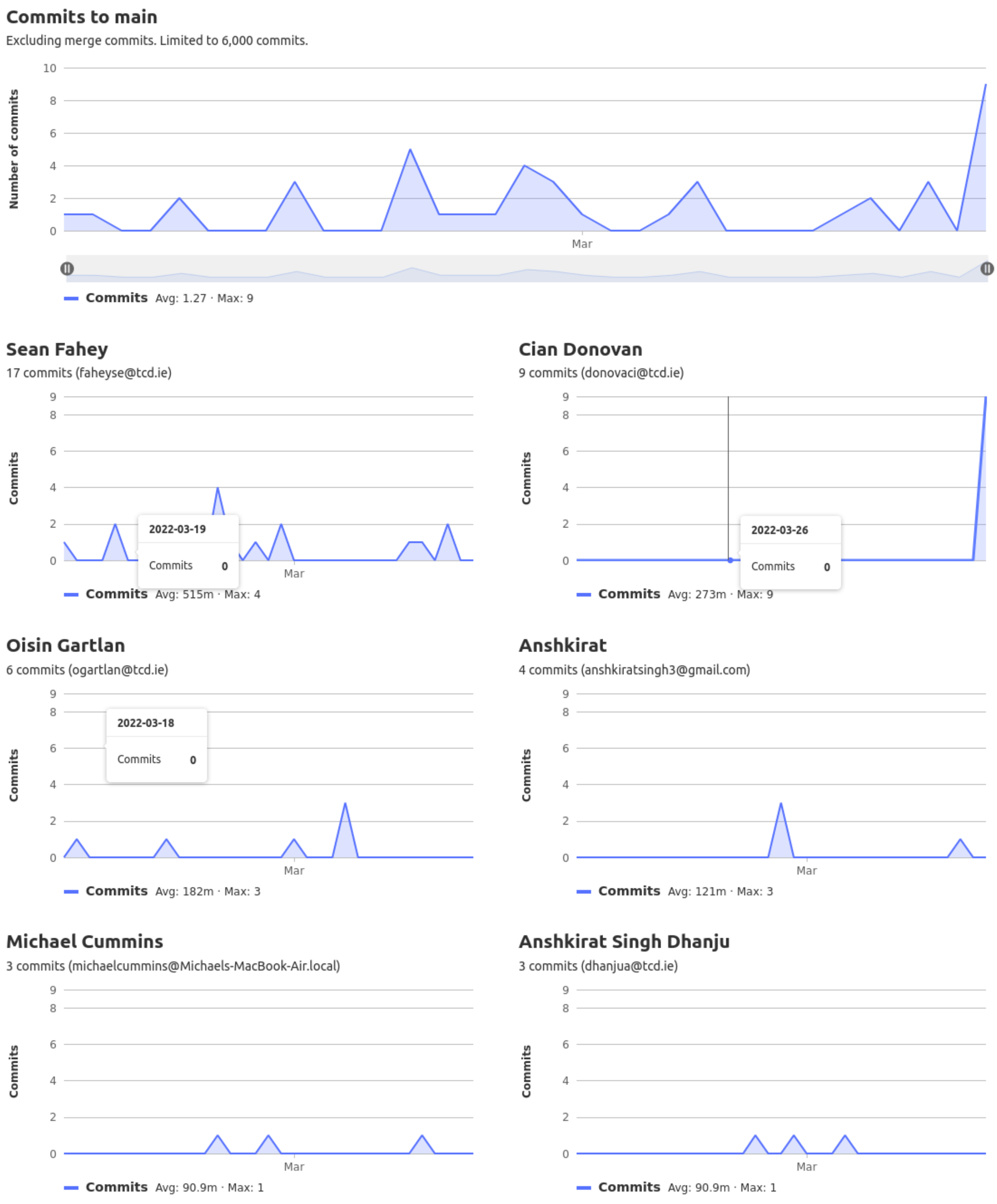
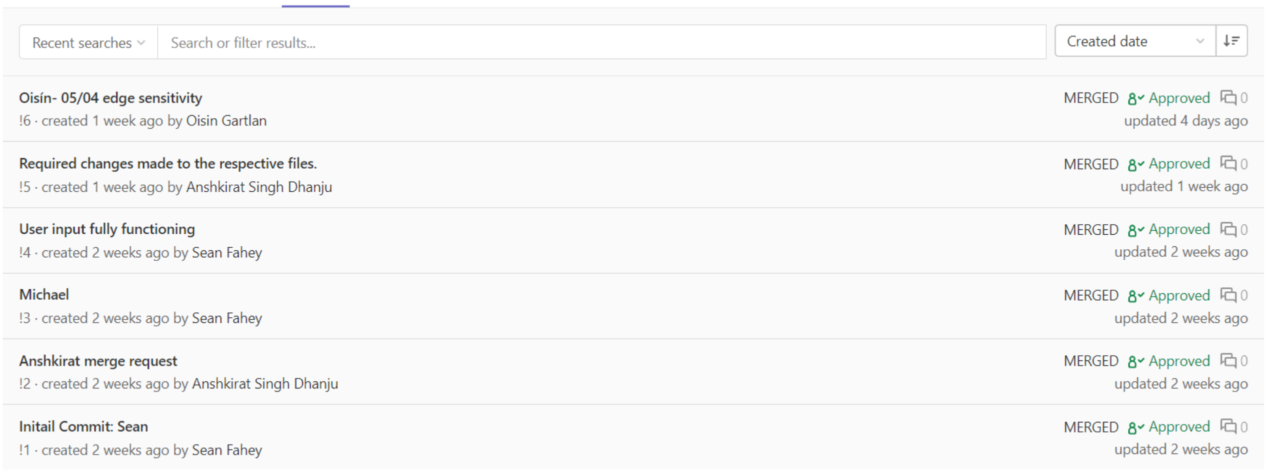


Fig 4 – Commit Statistics per person



Recent searches | Search or filter results... | Created date

Oisín- 05/04 edge sensitivity
!6 · created 1 week ago by Oisin Gartlan — MERGED Approved 0 — updated 4 days ago

Required changes made to the respective files.
!5 · created 1 week ago by Anshkirat Singh Dhanju — MERGED Approved 0 — updated 1 week ago

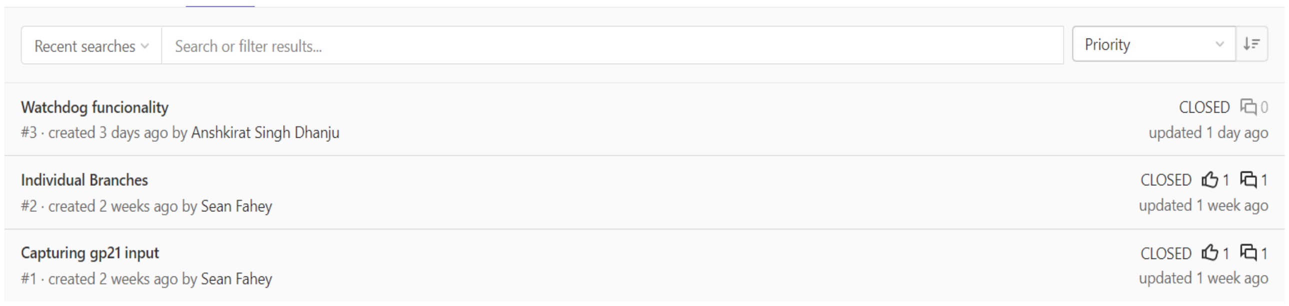
User input fully functioning
!4 · created 2 weeks ago by Sean Fahey — MERGED Approved 0 — updated 2 weeks ago

Michael
!3 · created 2 weeks ago by Sean Fahey — MERGED Approved 0 — updated 2 weeks ago

Anshkirat merge request
!2 · created 2 weeks ago by Anshkirat Singh Dhanju — MERGED Approved 0 — updated 2 weeks ago

Initail Commit: Sean
!1 · created 2 weeks ago by Sean Fahey — MERGED Approved 0 — updated 2 weeks ago

Fig 5 – Merge Request History (Creation and acceptance)



Recent searches | Search or filter results... | Priority

Watchdog funcionality
#3 · created 3 days ago by Anshkirat Singh Dhanju — CLOSED 0 — updated 1 day ago

Individual Branches
#2 · created 2 weeks ago by Sean Fahey — CLOSED 1 1 — updated 1 week ago

Capturing gp21 input
#1 · created 2 weeks ago by Sean Fahey — CLOSED 1 1 — updated 1 week ago

Fig 6 – Issues raised History

<u>Final Design:</u>

The programme starts by initializing both the leds and the gpio pins. The light is initially set to blue. A welcome banner is printed using printf and the welcome message stored in shared memory. The gpio interrupt handler is installed as normal by storing the address of the interrupt handler in the correct location in the vector table and then disabling and enabling the IRQ.

When presented with the welcome banner, the user is prompted to enter a string of morse characters to proceed to the following level. When pressing the gpio21 pin, rising and falling edges are detected to decode the signal for dots and dashes. To distinguish between dots and dashes, we first had to get the length of time for which gpio21 was pressed (t1-t0). We came up with a functioning system where initially the gpio handler detects interrupts as the button is pressed down. As the handler finishes, it changes to detect the button being released. The difference t1-t0 is either greater than or less than the cut-off for dot vs dash. Dots are stored as 0s and dashes as 1s, so if the user inputs .--. , it is stored as 0110, or 6 in decimal. We had initially tried to use both cores, one to detect rising edges and one to detect falling edges, but we decided instead to use the method described as it is much simpler and the execution time of assembly code is much faster than a player can press the button (on the order of microseconds).

A random number is then generated between 0 and 35, which corresponds to a position in an array holding all 26 letters and 10 digits (0-9), and converted to its morse counterpart. This string of dots and dashes must be stored for the duration of the attempt, and compared with the input from the player. The player is then prompted to enter the morse code for each character appearing on the screen.

As the C and the assembly programs have somewhat different functions, they hold different variables, but some (such as ‘level’ and ‘lives’) must be shared and therefore the C and ASM programs must update each other when one of these variables changes value. Variables were stored efficiently by making wrapper functions to communicate changes in shared variables between the C and ASM. The rgb LED indicates whether the game is being played or not, and how many lives are left during the game (green=3, yellow=2, orange=1, red=0/game over).

Build reproducibility was ensured through a platform-independent Docker container that provided a sandboxed environment for building the application from scratch. This could be run on any of the team members’ machines, or automatically built when triggered on a new git push. Unfortunately, the SCSS GitLab instance does not provide CI/CD runners out-of-the-box, and so this aspect of the development process ended up always being run locally. However, we independently confirmed on a private GitLab repo that our pipeline does in fact work.

Link to the Gitlab Repository:

https://gitlab.scss.tcd.ie/faheyse/microprocessors-group2

Individual documentations:

Anshkirat Singh Dhanju:
I was the project workflow owner for group 2. I made sure that there were regular update meets to discuss the progress of the group and also took regular updates from each member individually. I was also checking the progress and commits made on Gitlab so as to see if the work was progressing as per the deadlines discussed by the group. Also, I made sure to organise meetings if any member of the group was stuck with a certain problem.
In a nutshell, I looked over the overall working of our group.
Apart from my main role, I also contributed towards the code (can be seen in the commit history of our group repository). I did certain portions which were assigned to me by the project code owner (Sean) which included making wrapper functions and communicating changes in shared variables between the C and ASM, and also checked for any errors in the commits made to our main branch. Furthermore, I helped out Michael with minor video editing for the demonstration part and Oisín with certain sections of the report.

Michael Cummins:
As the project demonstration owner, it was my responsibility to present our assignment in an informative ninety second video that demonstrated the full functionality of our final design. Aside from this, I also developed a compilation/build environment in the main gitlab repository so the code could be compiled and tested without the need to clone the entire pico-apps repository. I also developed ARM code for detecting rising and falling edges on the GP21 GPIO pin on the raspberry pi. After this, I assisted Sean in the remaining ARM development tasks.

Sean Fahey:
As the code owner I lead the development and implementation of the code. This involved making a framework for how the program should work, and a structure for how the program should be made. I kept the same structure that was shown to us in the labs ; enter the assembly code from the C code and use wrapper functions to be accessed from within the assembly code. I devised methods for generating random letters/numbers, converting gpio input into dots and dashes, and updating variables, the rgb LED and controlling print statements. The tasks I set for Cian and Oisín were mainly to fix errors with the functionality of the game or to add features, such as printing statistics at the end of each level and at the end of the game. Each member created a git branch and then merged their code into the main branch once they were finished.

Cian Donovan:
As the GitLab Workflow owner it was my responsibility to ensure that as the project progressed, each team member was able to build the latest version of the software in a reproducible manor, so that platform specific bugs would not be an issue diverting time and resources away from the application at hand. This included writing the Dockerfile and CI/CD pipeline configuration.

For Linux-based systems, I wrote a udev rule and systemd service to automate the symlinking of the Pi Pico's serial port to a known path for seamless script integration, and to automate the running and lifecycle management of the OpenOCD server. This server would be started after receiving an event from the udev subsystem, automatically restarted if it crashed, and gracefully shutdown when the Pico Probe was unplugged and the server no longer needed.

I also worked with Sean and Michael around debugging specific bugs in the application logic, making use of unit testing to narrow down the root cause of any issues.

Oisín Gartlan:
I was appointed as the Project Documentation Owner in our first meeting. The first order of business for me then was to set up a rough skeleton of the report and make it available to all members on the Gitlab repository. I gave instructions on what needed to be done for each section of the report and encouraged the other members to take note of what they were doing to contribute to the assignment, as well as to fill in their respective sections of the report as they go.

I also proofread all contributions to the report and ensured the entire document was coherent before it was finally submitted. This included editing for clarity and removing repeated sections to keep the report as concise as possible. I also helped with implementing some piece code functionality/ functions as requested by Seán or other group members at meetings.