ENSC 351 Final Project Write-up

By Avinaash, Chris, Liam, and Manraj

System explanation:

System purpose:

The BeagleDOOM project is an attempt to run an open-source clone of the classic DOOM 1993 game on a custom controller and a Beaglebone board. The custom controller utilizes an 8x8 Matric and a 2 row LCD to display real-time player statistics such as Health level, Armour level and Kill count.

System implementation:

The DOOM clone, named “Chocolate DOOM”, will be played on a screen connected with HDMI and a USB controller. The controller will be assembled from a LED matrix, joystick, a 2 row LCD display, an additional potentiometer, 7 digital input buttons, and a 3D printed shell casing.

There will be two embedded systems in this setup: the Beaglebone and the Raspberry Pi Pico. The Beaglebone’s job will purely be to run the game and save player data onto the Pico. Chocolate DOOM’s source code has been natively compiled into an executive for Beaglebone to run, and custom functions were made to extract the data required for the controller to display.

The Raspberry Pi Pico oversees the hardware implementation. All controller components, such as the LCD, joystick, LED display and push buttons, are controlled through the executable code cross compiled to the Raspberry pi.

Project Circuit Diagram:

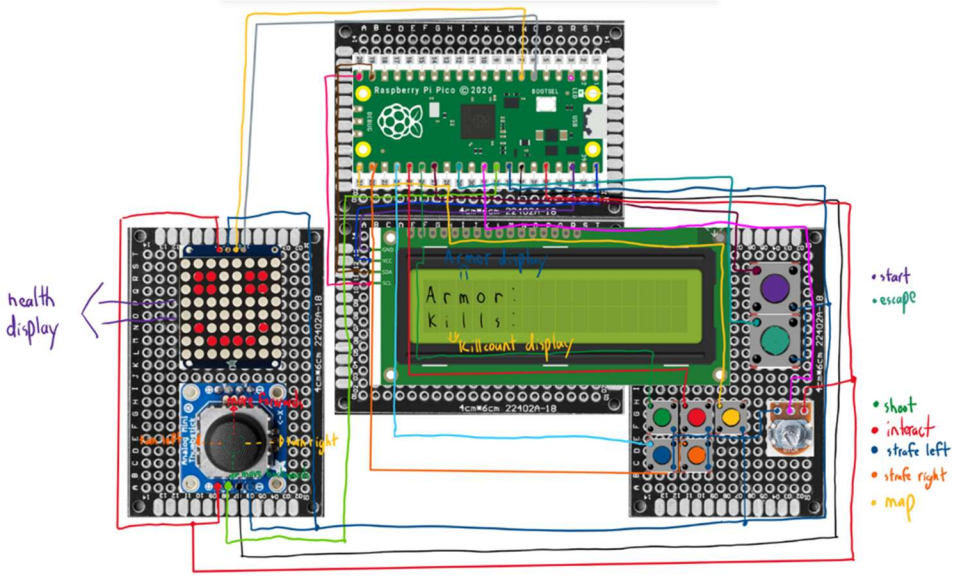
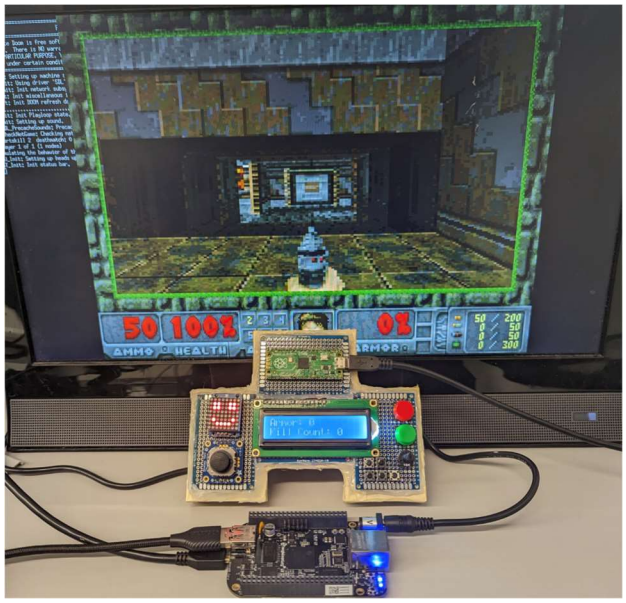


Image of Finished Product:



Persisting Issues:

- **CUSTOM PCB:** An initial goal during the planning of the project was the creation of a custom PCB that would connect the electrical components without needing the use of basic wires. Unfortunately, this goal was discarded due to time constraints, and the final product is currently hand wired.

- **PERFORMANCE DELAY:** Doom is a relatively large game for a Beaglebone to run. The performance is slow, and the Raspberry Pico updates its player data even slower. This causes the information displayed on the players controller to be significantly delayed relative to real time. The game runs, however the experience causes enough lag that it's borderline unplayable.
    - We still opted to show off our modified version of the game which updates the player stats on the controller, so also chose to run it on a more powerful linux computer to demonstrate it's complete functionality.

- **MULTIPROCESSING ON PICO:** We also wanted to run both the input handling and checking of player stats on separate processes on the Pico, taking advantage of its dual cores. However, we used CircuitPython which does not support this feature unlike MicroPython, which does. We did not realize this until we neared completion on the project, so we opted for using the async library from Adafruit to implement pseudo-multitasking via asynchronous operations to update the displays from the player_stats.txt file. This is not as effective as having true parallelism, however for the task of playing DOOM, this suffices.

Feature Table:

DESCRIPTION	HOST/TARGET	COMPLETENESS (FROM 1 - 5)	CODE	AUTHOR(S)	POTENTIAL NOTES
JOYSTICK	TARGET (RASPBERRY)	5	CIRCUIT PYTHON	MANRAJ	Can move in 8 directions and allows for player control
8X8 LED MATRIX	TARGET (RASPBERRY)	5	CIRCUIT PYTHON	AVINAASH	Controlled through I2C protocol, and acts as a visual indicator of player HP
2 ROW LCD DISPLAY	TARGET (RASPBERRY)	5	CIRCUIT PYTHON	AVINAASH	Used a I2C module to display player amour and kill count
WEAPONS SCROLL WHEEL	TARGET (RASPBERRY)	5	CIRCUIT PYTHON	MANRAJ	Used potentiometer for weapon selection with different resistances mapped to different weapon selection inputs.

DIGITAL INPUT BUTTONS	TARGET (RASPBERRY)	5	CIRCUIT PYTHON	MANRAJ	Used for basic player actions (shoot, interact, strafing and opening map)
3D PRINTED SHELL CASING	TARGET (RASPBERRY)	5	N/A	BRIAN LAM	Created shell for us using his home 3D printer
MODDED CHOCOLATE DOOM GAME ENGINE	TARGET (BEAGLEBONE) ALSO RUNS ON HOST (LINUX VM)	4	C	(OPEN SOURCED) MODS DONE BY: LIAM & CHRIS	Modified game files to extract player info and to save it onto the Raspberry Pi Pico. Runs poorly on beaglebone due to lack of power, however, runs smoothly on better hardware. (99% Open-source community, 1% our contribution)
MODDED OPEN SOURCE SNAKE GAME	TARGET (BEAGLEBONE) ALSO RUNS ON HOST (LINUX VM)	5	C	(OPEN SOURCED) MODDED BY: AVINAASH	More lightweight game to demo functionality on beaglebone than chocolate doom. Also has stat tracking functionality on the controller. (85% open-source, 15% our contributions)
MICROCONTROLLER MULTI-TASKING	TARGET (RASPBERRY)	4	CIRCUIT PYTHON	LIAM & CHRIS	CircuitPython does not support “true” multi-tasking with separate processes on each core. Instead, we used async functionality from adafruit_async

Extra Hardware/Software Used:

Beyond the Beaglebone, Zen Cape, and the course's guides/notes, other hardware and software libraries were also used. They will be listed here:

- Beaglebone Black
    - This has built in HDMI and 5V barrel jack for external power, which allows for the “Mobile console” experience we were aiming for.

- Joystick
- 8x8 LED matrix
- 2 ROW LCD display
- Potentiometer
- 7 digital input buttons
- 3D printed shell case

- Snake game source code
    - https://github.com/mnisjk/snake
    - This is an alternative game to run with lower performance requirements than DOOM
        - The code here has been modified to extract info and store it on the Raspberry Pi Pico

- Chocolate Doom source code & Freedoom community made game levels
    - https://github.com/chocolate-doom/chocolate-doom.git
    - https://freedoom.github.io/
    - We then modified Chocolate Doom to extract player info and store it on the Pico
- Small 720P TV Screen
- Raspberry Pi Pico microcontroller
    - Performs 2 separate functions asynchronously.
        - Handling player inputs
        - Updating LCD screen and 8x8 Matrix to display live player statistics.
    - CircuitPython for writing the firmware with Adafruit Libraries for controls, interfacing with I2C and handling HID protocol to communicate with Beagle Bone Green.
        - Specific libraries used alongside CircuitPython:
            - adafruit_hid, adafruit_async, adafruit_ht16k33
            - CircuitPython_LCD (Adafruit Library)