ECE 445

SENIOR DESIGN LABORATORY

PROJECT PROPOSAL

MacroME: The Programmable GameCube Controller

Team #67

DEAN BISKUP
(dbiskup2@illinois.edu)
ADITHYA RAJAN
(adithya2@illinois.edu)

TA: Megan Roller / Chi Zhang

April 3, 2020

Contents

1 Introduction

1.1 Objective

Super Smash Brothers is one of the most most famous video game franchises, with titles such as *Super Smash Bros. Melee* frequently ranked among the best fighting games of all time [1][2]. While current iterations of the *Super Smash Bros.* series allow the game's controls to be changed in-game, older titles such as *Melee* did not allow for this feature, forcing all players to play with default controls. In addition, *Super Smash Bros.*, like other fighting games, feature combinations of moves (combos) that are too complex and difficult to execute for beginners, yet required to compete with more experienced players.

We propose "MacroME: The Programmable GameCube Controller," a fully functional GameCube controller that is able to be programmed with remappable buttons and macroinstructions (macros) of multiple button presses. The control will look and feel mostly like a standard GameCube controller, but a microprocessor will translate the physical button presses to the emulated button signals, as well as allow programmable macros for executing complex input strings. The button and macro layouts will be programmed onto the controller using a USB connection to a PC, and stored on the controller until overwritten so that the controller is portable. The user would interface with a simple graphical user interface (GUI) on the PC to intuitively remap the buttons and macros.

We hope that MacroME will encourage beginner players to try the more complex characters in the *Super Smash Bros.* games, and make entry to the competitive scene less intimidating. In addition, due to the popularity of the GameCube controller, there are many adapters on the market that would allow this controller to also be used for PC, PlayStation, or Xbox games.

1.2 Background

1.2.1 Inspiration

MacroME is based off of Project 14 from Spring 2020 of ECE 445: "Button Remapping for GameCube Games such as Super Smash Bros Melee" [3]. This project achieved similar goals by creating an adapter that remaps GameCube controller signals. The adapter sits between a standard GameCube controller and the console, and is programmed by connecting to a smartphone app via Bluetooth.

MacroME differentiates itself from the solution proposed in Project 14 in several key ways. First, MacroME contains all of the hardware within the form factor of a physical GameCube controller. Second, MacroME adds the functionality of programmable macros. Third, MacroME's process of programming the controller forgoes Bluetooth, opting instead for a wired connection. These differences allow our controller to be more portable by not requiring any setup when connecting to a new console, as well as adds functionality that could be more beneficial to our users.

1.2.2 Example Combos

To clarify the types of difficult combos that programmable macros help make easy, two examples of complex maneuvers are described in this section.

Wavedashing is a technique that can be performed in *Super Smash Bros. Melee* that involves performing an air dodge diagonally into the ground, causing the character to slide a short distance [4]. It has become considered an essential technique for *Melee* gameplay, but it is difficult for beginners to consistently pull off the precise inputs.

Smash Directional Input (SDI) is a technique that can be performed in all *Super Smash Bros.* games that involves the player repeatedly inputting a control stick direction while getting hit by an attack, thus slightly altering their character's position and allowing their character to escape possible follow-up attacks [5]. Performing optimal SDI requires the player to input a new control stick input each frame, which is both unrealistic for beginners and causes unnecessary wear-and-tear on the controller.

1.3 High Level Requirements

- MacroME must have persistent memory so that it does not need to be reprogrammed between each use of the controller.
- MacroME must have a maximum latency of 16.67 ms between button press and signal output, which is equivalent to less than 1 frame of latency at 60 frames per second.
- The GUI program and the MacroME controller must allow for macros with both analog stick and button inputs per frame, up to a length of 60 frames (1 second).

2 Design

2.1 Block Diagram

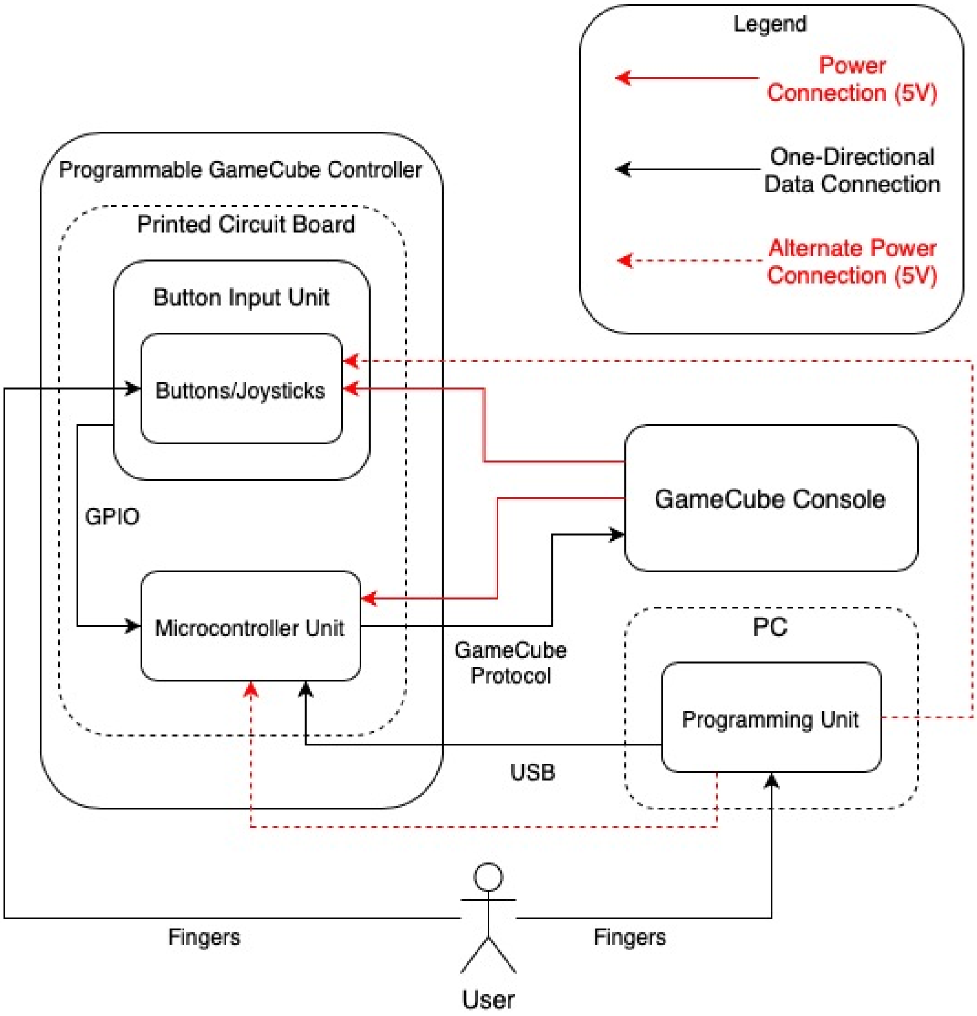

Figure 1: Block diagram of MacroME

Figure 1 shows the proposed block diagram for 'MacroME'. The block diagram demonstrates typical operation of the controller. The user will program their remapped buttons and macros through the Programming Unit (PU) on the PC. The Microcontroller Unit (MCU) then is able to store these settings, and during gameplay translate the user's button inputs into the desired remappings and/or macros.

2.2 Button Input Unit

The Button Input Unit (BIU) includes the physical controller, the buttons, and the analog sticks on the controller. This represents how the user physically interacts with the console. The inputs from the user will be sent to the Micro Controller Subsystem Unit which will process the inputs and make any modifications that the user had programmed previously. The game runs at 60 frames per second, so the latency must be kept to under 1 frame [6].

Requirement: The buttons, printed circuit board, joysticks, and housing for the BIU must be no larger than the size of a standard GameCube controller.

2.3 Microcontroller Unit

The Microcontroller Unit (MCU) is the main brains of the MacroME controller. It receives all inputs from the BIU, and outputs the remapped buttons or macros to the game console over the GameCube protocol within a single frame. The MCU has persistent memory, so that the stored button remapping and macros stay across power cycles, since a controller of this type will not be consistently powered. Inputs to the MCU are the pressed buttons from the BIU, as well as a USB connection to the PC when in programming mode.

Requirement 1: The MCU must be able to read currently pressed buttons, translate to remapped inputs, and output the remapped inputs and/or macros in under 16.67 ms (1 frame of latency).

Requirement 2: The MCU must have persistent memory, so that it does not need to be reprogrammed every time it is powered up.

2.4 Programming Unit

The Programming Unit (PU) allows the user to program the controller with different button mappings and macros. The user interacts with the PU via a GUI program that communicates with the MacroME controller through a USB connection. The macros that the user can program using the PU allow for any number of button presses at a time, with a resolution of 1 frame, for up to 60 frames.

Requirement 1: The PU must allow button remapping through the GUI.

Requirement 2: The PU must allow for up to all buttons and analog sticks to be pressed during each frame of macro input.

Requirement 3: The PU must allow for a maximum macro length of 60 frames, equivalent to 1 second of automated input.

2.5 Risk Analysis

The BIU presents the greatest risk to the successful completion of the project. While the button and analog stick inputs are simple on their own, the BIU requirement for a less than one frame response time poses the major complexity. While 16.67 ms is a very short amount of time, delays of one or two frames can certainly be felt by most experienced players [7]. In order for the user to feel that MacroME is just as responsive as a standard GameCube controller, this requirement must be met.

Additionally, we want the controller to ”feel good”, which for this project we will define as using the same size and locations for buttons and analog sticks as the original GameCube controller. This poses a unique challenge because most of the locations on the PCB are therefore already decided, and all of the new hardware will need to be designed into the surrounding space. For MacroME, we will consider the option of removing the rumble motor in order to make space for the additional components.

In order to create a seamless user experience and achieve the desired functionality for this project, these major potential issues with the BIU must be addressed.

3 Ethics and Safety

3.1 Ethics

The main ethical question that comes up in the design of this project is whether the use of our controller would constitute a breach of competitive integrity. Many players believe that the use of controllers that are not standard GameCube controllers should be considered cheating. However, recent pushes towards more ergonomic and modern controllers have been made, such as allowing controllers such as the SmashBox [8] [9]. However, it is likely that the additional functionality of programmable macros would make this controller illegal in a tournament setting. The target audience of MacroME is beginners who are looking to begin learning higher-skilled techniques or play with friends in a casual setting, so we believe this to not be an issue.

In terms of players fraudulently representing MacroME as tournament legal, we do not believe it will be possible at any tournament that checks controllers. While MacroME does try to look as similar to a traditional GameCube controller as possible, there are extra buttons that will be on the controller for the macros. Thus, it would be impossible to misrepresent this controller as an unmodified GameCube controller and sneak it into tournaments.

Additionally, we plan to make both the software and hardware design of MacroME open-source, such that the public may benefit from the design knowledge gained throughout this project, as well as accept criticism and suggestions of our technical work, in accordance with points 5 and 7 of the IEEE Code of Ethics [10].

3.2 Safety

There are few safety considerations for this project. Because of its nature as a video game controller, all the systems are at low voltage and current. Additionally, the controller is of small size and weight, thus the likelihood of serious injury from dropping it on a foot or other body part is very low. Our main safety considerations are for the students during the design and prototyping process. We will make sure that while soldering, taking apart GameCube controllers, and taking part in other lab activities, all students will adhere to strictest safety standards as advised by the course staff.

References

[1] Game Informer Staff. (May 2019). ”The 30 Greatest Fighting Games of All Time”, [Online]. Available: https://www.gameinformer.com/2019/04/25/the-30-greatest-fighting-games-of-all-time (visited on 04/01/2020).

[2] B. Bernstein. (May 2019). ”20 Best Fighting Games of All-Time: The Ultimate List”, [Online]. Available: https://heavy.com/games/2015/01/best-fighting-games/ (visited on 04/01/2020).

[3] M. Qian, S. Yaganti, and Y. Wu. (2020). ”Button Remapping for GameCube Games such as Super Smash Bros. Melee”, [Online]. Available: https://courses.engr.illinois.edu/ece445/getfile.asp?id=16677 (visited on 04/01/2020).

[4] Smash Wiki Community. (Nov. 2019). ”Wavedash”, [Online]. Available: https://www.ssbwiki.com/Wavedash (visited on 04/01/2020).

[5] ——, (Mar. 2020). ”Smash Directional Influence”, [Online]. Available: https://www.ssbwiki.com/Smash_directional_influence (visited on 04/01/2020).

[6] ——, (Mar. 2020). ”Frame”, [Online]. Available: https://www.ssbwiki.com/Frame (visited on 04/02/2020).

[7] B. Kuchera. (Dec. 2018). ”Why Super Smash Bros. Ultimate feels so laggy”, [Online]. Available: https://www.polygon.com/2018/12/14/18140814/super-smash-bros-ultimate-input-lag-latency-feel.

[8] Hit Box Arcade. (2020). ”Smash Box”, [Online]. Available: https://www.hitboxarcade.com/pages/smash-box (visited on 04/01/2020).

[9] J. Cuellar. (Apr. 2018). ”Mr. Wizard on the Smash Box”, [Online]. Available: https://www.eventhubs.com/images/2018/apr/17/mr-wizard-smash-box/ (visited on 04/01/2020).

[10] IEEE. (2016). ”IEEE Code of Ethics”, [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html (visited on 02/08/2020).