Report

Full Demo link:

* Complexity 1: Joystick

<https://drive.google.com/file/d/1sxbtiDV62vvpgvk8Fzi2wfEHPeLdTmhn/view?usp=sharing>

* Complexity 2: 2 Player

<https://drive.google.com/file/d/1KrQmqweILM3DzxnldAn13-Ijelkldkr6/view?usp=sharing>

* Complexity 3: Multiple Microcontrollers

# Project Description:

Step is a two-player game where 2 players race. The winner of the game is the player who wins the most races out of 5. The flashing LED’s represents which feet the player needs to move. Players control their feet by using the joysticks.

# User Guide:

### Rules:

On release of the start button, the game immediately starts. When the respective player’s left LED turns on, the respective player must move their left foot. When the respective player’s right LED turns on, the respective player must move their right foot. First player to win 3 rounds wins the game.

### Controls:

There is a “button” that starts the game. When the game is running, pressing that “button” again will the reset the current round. After each round, the “button” will start the next round. After each game, players can start another game by turning the board off and on again.

Each player has 1 joystick each. Move the joystick left to move the left foot. Move the joystick right to move the right foot. Push the button between each round.

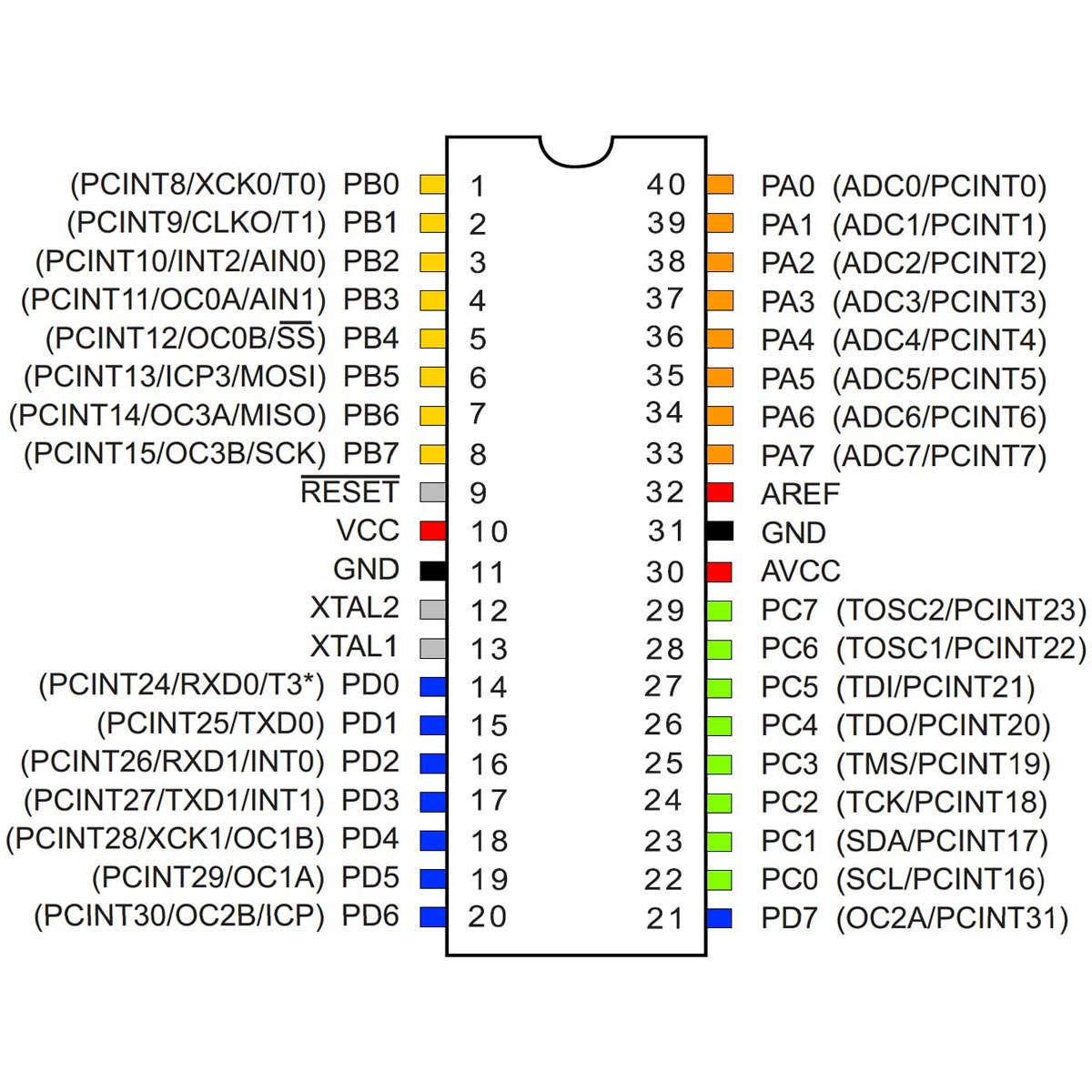
### Special Considerations:

There was an issue where if a normal push button was used for the start/reset function, it would actuate on it’s own every second. To fix that, we replace the button with a “button”, which is a switch connected to the pin, which can switch connections between voltage and ground.

# Source File Description:

* MainChip.c - Tasks and Functions for the main ATMEGA1824
* SecondChip.c - Tasks and Functions for the second player
* io.h - header for the functions used to write to the LCD Display. Code provided from the course
* io.c - functions for writing to the LCD Display. Code provided from the course
* scheduler.h - basic structure for tasks. Code provided from the course
* timer.h - functions to allow timer functionality. Code provided from the course
* simAVRHeader.h - allows interfacing with the ATMEGA1824

# Component Visualization



There are two ATMEGA1284 chips in use.

Main Chip:

* PD[7:0], PB[7:6] —> LCD Display
* PB[1:0] —> Player 1 LED Feet
* PA0 —> Player 1 Joystick
* PC7 —> Reset “button”
* PC[1:0] —> Communication with Secondary Chip [recieve, send]

Secondary Chip:

* PD[2:1] —> Player 2 LED Feet
* PD0 —> Communication with Main Chip (send)
* PB0 —> Communication with Main Chip (recieve)
* PA0 —> Player 2 Joystick

# Technologies Learned

One thing I learned is that between 2 microcontrollers, synchronization is very important. For my project, the specific issue is that I needed to communicate between the 2 microcontrollers without desynchronizing. This is esspecially important because one chip is controls the game and Player 1, while the other chip controls Player 2.