Software Overview

Year: 2025 **Semester:** Spring **Team:** 1 **Project:** Electronic Skee Ball Machine
Creation Date: January 31, 2025 **Last Modified:** January 31, 2025
Author: David Fall **Email:** david.c.fall33@gmail.com

Assignment Evaluation: See Rubric on Brightspace Assignment

1.0 Software Overview

The functionality of the Electronic Skee Ball Machine can be classified into six main categories: a button-operated motor control system, a joystick-operated motor control system, an OLED character display interface, a sound generation system, an ultrasonic sensor interface, and the transition between states. The software functionality of these categories is detailed in the following contents of this section.

1.1 Button-Operated Motor Control

1.1.1 Determining Motor Power via Button Press Duration

The force with which the motor launching the ball is driven is determined by the duration of a button press. When the push button is pressed down, an interrupt is triggered that records the time the button was pressed, starts a timer which expires after the maximum button press duration, and starts another timer which polls the press duration at regular intervals as long as the button remains pressed. If the button is released before the timer expires, it triggers an interrupt that calculates the difference in milliseconds between the current time and the time the button was initially pressed. It then returns this duration to be used for powering the motor. If the timer expires before the button is released, it disables the button-release interrupt and returns the maximum press duration to be used for powering the motor. The polling timer is used to incrementally measure the press duration by comparing the current time to the time of the initial press, and this duration is sent to a separate function for writing to the OLED display (described in section 1.3.1).

1.1.2 Driving a DC Motor

The DC motor is driven with a variable force, and this force is controlled by software using the duration of the button press. The software must first configure an external pin for PWM signal generation. Then, the software retrieves the duration of a button press (as described in Section 1.1.1) and, using this duration, calculates a duty cycle between zero and one hundred percent by dividing the measured press duration by the maximum allowed press duration. Then, a PWM signal with the calculated duty cycle and 10 kHz frequency is generated and used to drive the DC motor. This signal persists for approximately three seconds before the software disables it in order to stop driving the motor.

1.2 Joystick-Operated Motor Control

1.2.1 Reading Analog Joystick Input

The direction of launch is controlled by an external joystick. The joystick's angular position is encoded as an analog voltage. This voltage is used with the ADC to obtain an integer value between 0 and 4095. Using linear interpolation, this range is mapped to the range [-4, 4] to determine the angular displacement (in degrees) that the motor should experience. The displacement is then returned for usage in PWM motor control, described in Section 1.2.2. This process takes place in an interrupt service routine, which is triggered by a 10 Hz timer. The timer is initialized by software to run at the appropriate frequency and trigger the appropriate interrupt service routine.

1.2.2 Driving Servo Motor with PWM

As mentioned above, the servo motor is controlled by generating PWM signals which generate an angular displacement in the motor. The software first initializes a GPIO pin to generate PWM signals. Then, the angular displacement of the servo motor is tracked by software, and it is initialized to zero. Before a PWM signal is generated, the software checks if the generation will move the motor outside of its acceptable window of displacement ([-45 degrees, 45 degrees]). If so, the signal is not generated. Otherwise, a PWM signal is generated by mapping the desired angular displacement to a pulse width in microseconds and then using this pulse width to determine the duty cycle for the PWM signal. The signal is then generated on the aforementioned pin with the calculated duty cycle and a frequency mandated by the hardware of the servo motor.

1.3 OLED Character Display Interface

1.3.1 Interfacing with the OLED

The Electronic Skee Ball Machine uses an OLED character display to provide feedback to the user. This is done by first initializing a serial communication interface between the microcontroller and the OLED, i.e., the software configures two pins to be used for the interface. Using this interface, the software initializes the display by sending a series of packets. Each packet is ten bits in length, and the packets are sent sequentially. After this initialization sequence, the OLED is prepared to have text written to it. The software writes text to the display by first sending a packet to prepare the OLED to receive text and then sequentially sending the bit representation of each ASCII character in subsequent packets (one character per packet).

1.4 Sound Generation

1.4.1 Processing a WAV File

Throughout the game, sound stored in WAV files is played on an external speaker. From a software perspective, this is accomplished by first initializing pins to use for a serial communication interface. To play a WAV file on the speaker, the first 44 bytes of the file are read in order to analyze relevant metadata such as the sample rate of the audio. This metadata is stored and used to properly configure the serial communication interface. Then, the remainder of the file (i.e., the actual audio data) is read in chunks of size 512 bytes, and each chunk is then written to hardware via the serial communication interface. The mechanism by which these chunks are read and written is controlled by hardware and thus outside the scope of a software overview. This process persists until the audio file has been exhausted.

1.5 Ultrasonic Sensor Interface

1.5.1 Using An Ultrasonic Sensor to Measure Distance

The machine utilizes an array of six ultrasonic sensors. An ultrasonic sensor is used by sending a ten microsecond high pulse on the trigger pin and waiting for a return signal on the echo pin. By measuring the time between the trigger emission and echo return, one can use the speed of sound to determine the distance to the nearest object. Therefore, the ultrasonic sensors are controlled in software by initializing the proper GPIO pin to act as an output, pulling this pin high, waiting ten microseconds, and finally pulling the pin low. Then, a timer is started, and it expires when the echo pin transitions from low to high. The duration recorded by the timer is then used in a conversion between time and distance to arrive at a distance in centimeters to the nearest object.

1.5.2 Using the Distance Measurements to Detect the Ball

Using the method for measuring distance on a single sensor described in Section 1.5.1, the software of the machine iterates quickly over all six ultrasonic sensors to determine which of them, if any, detects the ball. This is done by first starting a timer that governs a window in which readings are taken. If this timer expires before any sensor detects a ball, the software assumes that the ball is not going to be detected at all, so it stops reading. During the window, a counter is utilized to track which sensor is being read from. The distance is read from the current sensor, and if the distance is less than an experimentally determined threshold, it is assumed that the ball is present. In this case, the readings are stopped, and the index of the current sensor (i.e., the sensor that detected the ball) is returned. Otherwise, the counter is incremented modulo six to ensure that the counter always symbolizes the sensor currently being ready from. From a software perspective, the method for reading distances from the sensors is exactly the same regardless of the current sensor; the trigger and echo signals are directed to and from the appropriate sensor via hardware as described in Section 3.

2.0 Description of Algorithms

The Electronic Skee Ball Machine utilizes two main algorithms to achieve its functionality. The first of these is the implementation of linear interpolation to determine the angular displacement for the servo motor, and the second is a software and hardware modulo counter for iterating through an array of ultrasonic sensors.

2.1 Linear Interpolation of ADC Outputs

In order to determine the proper angular displacement to be experienced by the servo motor at any given iteration, the input signal from the analog joystick must be used to generate an angular reading within a certain range. The values given by the ADC range between 0 and $2^N - 1$, where N is the size of the ADC block in bits. Therefore, interpolation must map values from the range $[0, 2^N - 1]$ to $[-\theta_{max}, \theta_{max}]$, where θ_{max} is the maximum angular displacement to be experienced in one step, measured in degrees. Therefore, the determination of the angular displacement is given as:

$$\theta = \frac{2d\theta_{max}}{2^N - 1} - \theta_{max}$$

where *d* is the value read by the ADC.

2.2 Iteration with Modulo-N Counter

In order to handle multiple iterations over the ultrasonic sensors, it becomes necessary to implement a modulo-N counter where N is the number of ultrasonic sensors. This must be done so that a counter can be incremented after every reading yet continue to represent the index of the sensor being read from at any given moment. This algorithm is implemented separately in hardware and software. In software, it is a simple addition with one modulo N:

$$c_t = (c_{t-1} + 1) \ \% \ N$$

where c_t is the value of the counter at the current time step, c_{t-1} is the previous value of the counter, and N is the number of ultrasonic sensors in use. In hardware, the modulo N operation is implemented by toggling the asynchronous reset on a binary counter when its value becomes equal to N. In the Electronic Skee Ball Machine, the intended number of ultrasonic sensors is six, so the counter is given by:

$$c_t[2:0] = (c_{t-1}[2:0] + 1) \cdot (1 - c_{t-1}[2] \cdot c_{t-1}[0])$$

In hardware, this is achieved by computing the logical AND of bits two and one and using the result as the asynchronous reset of the binary counter, effectively resetting the counter when its value becomes six.

3.0 Description of Data Structures

The Electronic Skee Ball Machine does not make use of any meaningful data structures that would warrant a description in this section. However, it does make use of serial communication protocols, some of which utilize structured information packets to transmit information. The two most meaningful such packets are communication packets for the OLED character display and the audio data packets sent to an amplification circuit for sound generation.

3.1 OLED Communication

The OLED display in use is the SOC1602A. When configured for serial communication, this device expects ten bit packets to be sent sequentially [1]. The ten bit packet is structured such that bit nine (the most significant bit) serves as a register select signal for indicating if the packet is meant to be a command or data, bit eight indicates if the master device is reading or writing from the slave, and bits seven through zero contain the one-byte command or one-byte data. Packets can be sent individually within an arbitrary amount of time in between them so long as each packet is exactly ten bits and is structured exactly as described.

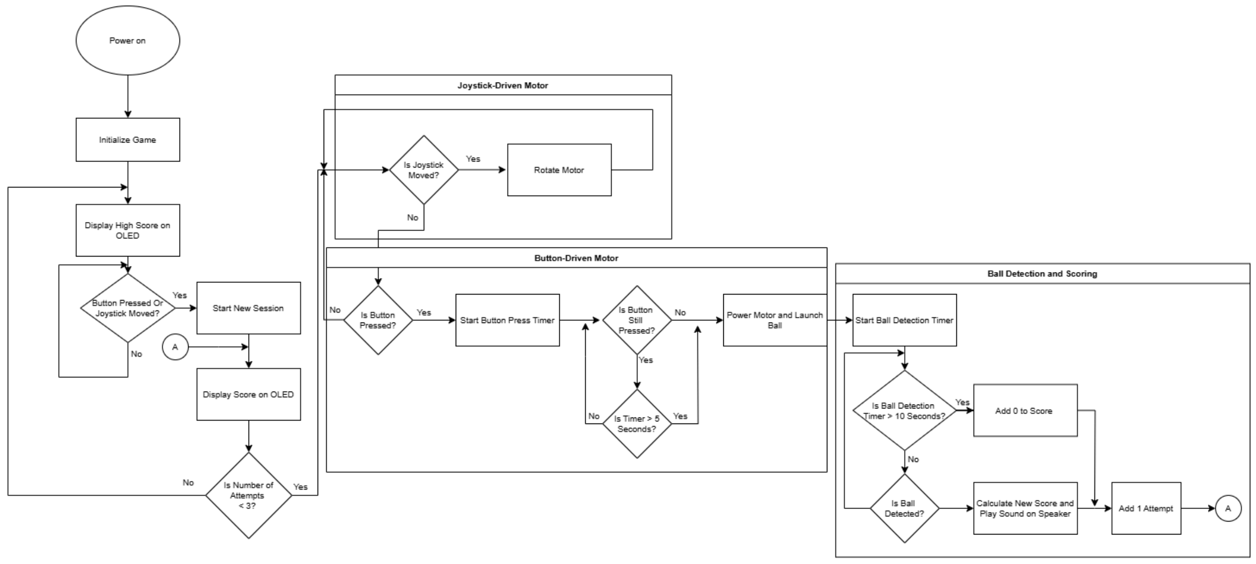
3.2 Audio Transmission

The audio data read from a WAV file must be transmitted to an external amplification circuit via serially communicated packets. Each packet consists of 256 audio samples, and each sample consists of a left and right channel reading. Each channel reading is 16 bits. The audio samples are transmitted at the sampling rate of the recorded audio (e.g., 44.1 kHz).

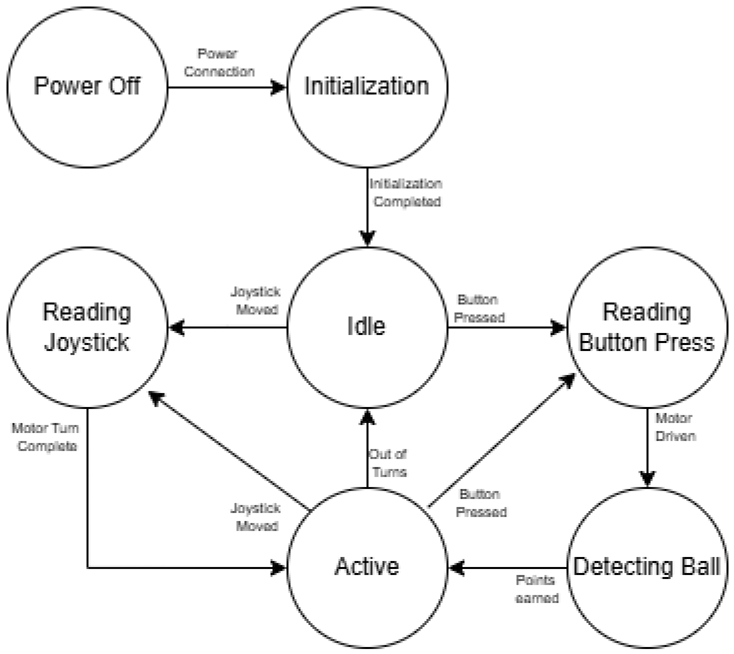
4.0 Sources Cited:

[1] “Dot matrix OLED display module Manual 1602A -Oled.” Purdue Engineering. Accessed: 31 Jan. 2025. [Online]. Available: https://engineering.purdue.edu/ece362/refs/SOC1602A.pdf

Appendix 1: Program Flowcharts



Appendix 2: State Machine Diagrams


Power Off
Power Connection
Initialization
Initialization Completed
Reading Joystick
Joystick Moved
Idle
Button Pressed
Reading Button Press
Motor Turn Complete
Out of Turns
Motor Driven
Joystick Moved
Button Pressed
Active
Points earned
Detecting Ball