Lab 8 State Machine and Audio

11/5/21

11/12/21

Jeremiah Webb

Introduction:

In earlier labs, we have learned how state machines and the switch statement works. In lab 8, understanding how audio operates in conjunction with hardware shows how programmers can influence the physical world. Inputting and outputting audio is a must in the modern world, with cell phones and through the internet using a microphone is commonplace. Furthermore, understanding how a microphone processes audio data is a good exercise in how to do action statements, similar to an on-click event when audio is inputted, we can tell the microprocessor to do an operation. In this case, such an operation is controlling a red LED.

Code Snippet:

```
switch (currentState) {
    case State0:
        if (stateChanged) {
            // Initialize user LEDs
            BSP_LED_Init(LED_grn);
            BSP_LED_Off(LED_grn);
            // Initialize the red LED as well here
            BSP_LED_Init(LED_red);
            BSP_LED_Off(LED_red);

            // Configure all Joy keys
            BSP_JOY_Init(JOY_MODE_GPIO);

            currentState = State1;
            led_loop_END = LOOP_STATE1;
            audio_THRESHOLD = AUDIO_STATE1;
```

```
        }
        break;

    case State1:
        if (BSP_JOY_GetState() == JOY_SEL) {
            currentState = State2;
            led_loop_END = LOOP_STATE2;
            audio_THRESHOLD = AUDIO_STATE2;
        }
        else if (BSP_JOY_GetState() == JOY_RIGHT) {
            currentState = State3;
            led_loop_END = LOOP_STATE3;
            audio_THRESHOLD = AUDIO_STATE3;
        }
        break;

    case State2:
        if (BSP_JOY_GetState() == JOY_LEFT) {
            currentState = State1;
            led_loop_END = LOOP_STATE1;
            audio_THRESHOLD = AUDIO_STATE1;
        }
        else if (BSP_JOY_GetState() == JOY_RIGHT) {
            currentState = State3;
            led_loop_END = LOOP_STATE3;
            audio_THRESHOLD = AUDIO_STATE3;
        }
        break;
```

```
    case State3:
        if (BSP_JOY_GetState() == JOY_LEFT) {
            currentState = State1;
            led_loop_END = LOOP_STATE1;
            audio_THRESHOLD = AUDIO_STATE1;
        }
        else if (BSP_JOY_GetState() == JOY_SEL) {
            currentState = State2;
            led_loop_END = LOOP_STATE2;
            audio_THRESHOLD = AUDIO_STATE2;
        }
        break;


    default:
        while (1);
    }
  // Add code to copy the "right channel"
PlayBuff[(2*i)+1]= SaturaLH((RightRecBuff[i] >> 8), -32768,
32767);
int find_audio_max(void) {
    int max_value = 0;
          max_value = PlayBuff[0];
    // Add code to find the maximul absolute value of the left
and right channels.
    for (i = 0; i < 4096; i++) {
                    if(abs(PlayBuff[i]) > max_value){
                           // Add code here
                    max_value = abs(PlayBuff[i]);
```

```
                        }
    }
    return max_value;
}
```

Discussion:

One issue I encountered was the switch statement being used. Often it would bug out, due to new inputted if statements or an accidentally moved break statement. Lab 6 was a good resource to refer to and edit the switch statement in this lab. Researching how audio data is handled by C was quite interesting, furthermore researching into the BSP code works is interesting when compared to HAL, in HAL there was much more user control and input, in BSP it's simpler and straightforward.

Questions:

When left key is pressed, the value is 2. When nothing is pressed, 5 is the value.

Results:

Overall, the lab went well, something to note that was slightly confusing was taking the code from the main_hw.c file and transferring some code seemed a little unnecessary. Furthermore, researching how BSP could be used more in-depth in this lab would be fun and useful for future applications. Similarly, recreating a previous lab with BSP would also give a good insight into how it functions. When lightly clapping, the board displays a red LED on state 1, on state 2, heavier claps are needed, and in state 3 I have been unable to find something that produces enough noise to trigger the LED to display.