

Final Project Report

Brandon Bernier

ECE 3520-30: Microprocessors

Professor Eom

December 17, 2012

1. Objective

The goal of this final project was to design, code, and test a functional final project on the AVR microcontroller board that was built this semester. Students proposed ideas that would utilize specific functions and capabilities available to them with this specific board. The goal was to use the onboard LEDs and switches, timer interrupts for sound, and program memory for storing lookup tables. The objective of this particular project included first soldering together and assembling the AVR project board. Then, the goal was to write a program that would use the board as a type of password protected device. This device would have a password that once entered allowed the device to manipulate a number or play a set of songs. The entire code was planned out, written in AVR Studio, and tested on the AVR project board to ensure it was fully working properly.

2. Procedure

The design procedure for the final project resembled much of what has been done for every lab experiment throughout the semester. First, the functionality of the device needed to be planned out based around the given capabilities of the AT90S4414 microcontroller and the board it was mounted on. The main portion of the design that needed to be planned out well in advance was the user interface and how the user would know what mode or level they were in. The first thought was to use LEDs to signify where the user was and this is what eventually was done. Different LED patterns play as a response to user actions.

Specifications:

Different Modes:

- SW3 will always move back to the last level of operation or completely lock the device again when pressed
- Locked – enter 4 number password for access to device
 - LEDs flash on and off 3 times if password is incorrect
 - Two middle LEDs light and hold steady once device is unlocked
- Mode 1 – Number Mode
 - This mode stores a number value which is essentially password protected
 - The current value of the number is always displayed to the LEDs in binary
 - SW0 will reset the number to 0
 - SW1 increments the number by 1
 - SW2 decrements the number by 2
- Mode 2 – Music & Light Show Mode
 - This mode has 4 possible song and light shows as well as a three-key keyboard
 - SW2 will always lead to the three-key keyboard
 - Song require two switch presses to reach and play them
 - SW0 will lead to regular songs
 - SW0 again plays the “Super Mario Bros. Theme Song”
 - SW1 will play “Heart and Soul”
 - SW1 will lead to the Christmas songs section
 - SW0 plays “Jingle Bells”
 - SW1 plays “God Rest Ye Merry, Gentlemen”
- Mode 3 – Password Change Mode
 - Once SW2 is pressed to enter this mode, a specific LED will light
 - At this point, a new 4 number long password can be entered
 - At any point in the future, this must be used to unlock the device
 - Locking the device again will require the use of the new password

Inputs:

- Switches 0-3

Outputs:

- LEDs 0-3
- Audio from the onboard speaker

Further Design Considerations:

In order to ensure that all of the final project requirements were met as specified, the student designed their idea around those factors. The switches were used as inputs to the device and as the main user control over the operation of the device. The LEDs were used as output signals to alert the user to the state of the device as well as during the songs as an LED light show. Finally, timer interrupts were used to produce musical notes and compiled in program memory to allow four different songs to be played. This design was extensive and required almost all of the assets available on the AVR project board, including the vast majority of the onboard Flash memory.

3. Results

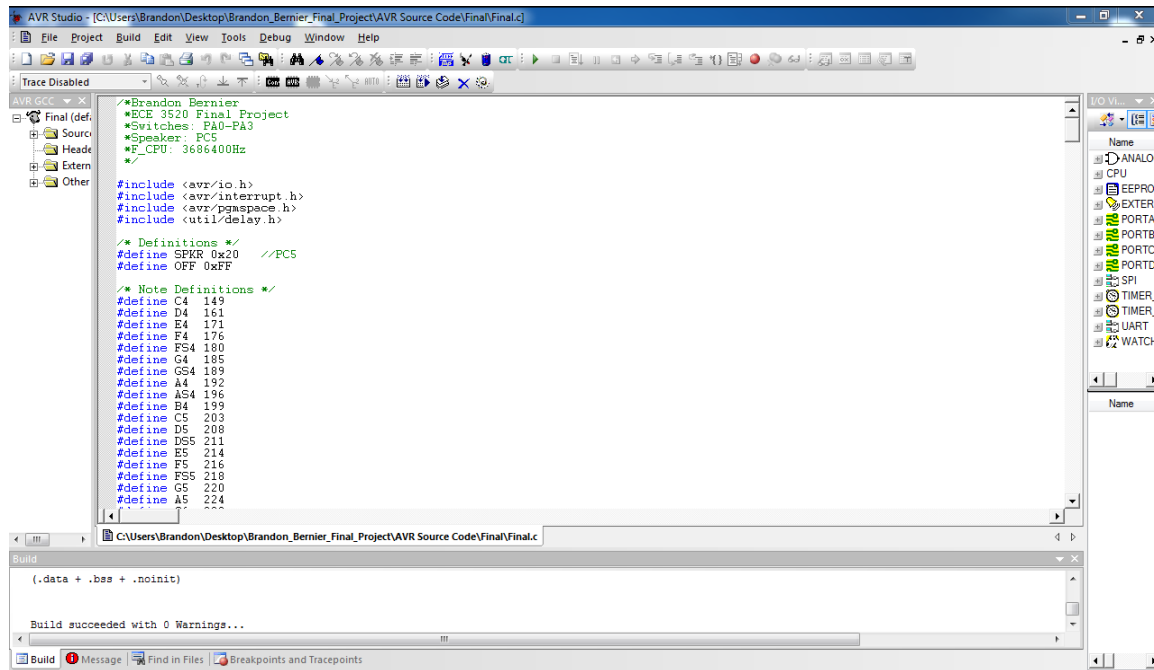


Figure 1 – Compiled C Code for the Final Project

4. Discussion

While the design process of this device has already been described at length, it is necessary to outline some of the difficulties and challenges faced when implementing the code to fulfill this design. One of the major obstacles when programming with the AVR project board as opposed to the STK500 was that the LEDs and switches were all attached to the same port. This made it quite complicated and, at certain times, impossible to have the board operate entirely as planned. For instance when attempting to make the LEDs blink in a certain pattern when a particular switch was pressed gave many ISP Mode Errors, generally meaning the code was written in a way that the hardware would not be able to actually do what was laid out. To get around this issue, the code was actually rewritten from scratch, leaving out many of the more complex LED patterns that were planned and instead replacing those with static LED displays.

Another difficulty was being able to store all of the necessary elements in program memory. The four songs chosen to use on the board were by no means short, especially the Mario Theme Song, and the memory usage was originally actually over 100%. To reduce this, some unnecessary delay loops were removed, which actually appeared to take up a bit of program memory themselves. In the end, the memory usage was right around 99.4%.

The easiest way of being able to actually implement this code turned out to breaking it down into many sections. The main part of the code is actually fairly short and branches off into many other predefined functions. Each of these other functions performs a given task and drastically simplified the coding of the entire project. For instance, the functions that play each individual song were originally written in their own AVR project, then copied into the main final project as a function of their own. This allowed the songs to be tested separately of the rest of the device's functionality before being totally compiled in the end.

An appendix with the entire source code for the final project is attached in **Section 6: Code Appendix**, with some accompanying comments.

5. Conclusion

This project challenged students to design, code, and test a functional and useful device using the AT90S4414 microcontroller on the AVR project board. While difficult for a variety of reasons, at its base, the project combined elements that had been used throughout the semester in different lab experiments and homework assignments. The ability to combine all of these elements in not only a functional working device, but also something that represented a useful product was the key goal of this entire project.

Overall, students were challenged by the requirements and implementation of this project, but the results were extremely fulfilling and rewarding. Watching as the design and implementation of a simple password protected device came to fruition models the real world applications of systems design and the use of microcontrollers. The skills learned throughout this class and particularly during this project will be extremely useful and valuable moving forward.

6. Code Appendix

```
/*Brandon Bernier
 *ECE 3520 Final Project
 *Switches: PA0-PA3
 *Speaker: PC5
 *F_CPU: 3686400Hz
 */

#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/pgmspace.h>
#include <util/delay.h>

/* Definitions */
#define SPKR 0x20 //PC5
#define OFF 0xFF

/* Note Definitions */
#define C4 149
#define D4 161
#define E4 171
#define F4 176
#define FS4 180
#define G4 185
#define GS4 189
#define A4 192
#define AS4 196
#define B4 199
#define C5 203
#define D5 208
#define DS5 211
#define E5 214
#define F5 216
#define FS5 218
#define G5 220
#define A5 224
#define C6 229

/* Function Prototypes */
int number(void);
int password_reset(void);
unsigned char read_switches(void);
unsigned char decode_switches(unsigned char x);

int music(void);
int three_note_keyboard(void);
int mario_song(void);
int heartandsoul(void);
int jinglebells(void);
int godrest(void);

/* Program Memory */
const unsigned char keyboard_notes[5] PROGMEM = {-40, -42, -45, OFF, OFF}; //Reload
values for each note of organ

unsigned char mario[][2] PROGMEM =
/*1*//{E5,16},{E5,8},{E5,16},{OFF,32},{C5,16},{E5,8},{G5,8},{OFF,16},{G4,8},{OFF,16},
/*2*//{C5,8},{OFF,32},{G4,16},{OFF,16},{E4,8},{OFF,32},{A4,16},{OFF,32},{B4,16},{OFF,32},{
AS4,16},{A4,8},
/*3*//{G4,83},{E5,83},{G5,83},{A5,8},{F5,16},{G5,16},{OFF,32},{E5,16},{OFF,32},{C5,16},{D5
,16},{B4,16},{OFF,16},
/*4*//{C5,8},{OFF,32},{G4,16},{OFF,16},{E4,8},{OFF,32},{A4,16},{OFF,32},{B4,16},{OFF,32},{
```

```

AS4,16},{A4,8},
/*5*/{G4,83},{E5,83},{G5,83},{A5,8},{F5,16},{G5,16},{OFF,32},{E5,16},{OFF,32},{C5,16},{D5,16},{B4,16},{OFF,16},
/*6*/{OFF,16},{G5,16},{FS5,16},{F5,16},{DS5,8},{E5,16},{OFF,32},{GS4,16},{A4,16},{C5,16},{OFF,32},{A4,16},{C5,16},{D5,16},
/*7*/{OFF,16},{G5,16},{FS5,16},{F5,16},{DS5,8},{E5,16},{OFF,32},{C6,16},{OFF,32},{C6,16},{C6,8},
/*8*/{OFF,16},{G5,16},{FS5,16},{F5,16},{DS5,8},{E5,16},{OFF,32},{GS4,16},{A4,16},{C5,16},{OFF,32},{A4,16},{C5,16},{D5,16},
/*9*/{OFF,16},{DS5,8},{OFF,32},{D5,16},{OFF,16},{C5,8},{OFF,8},
/*10*/{OFF,16},{G5,16},{FS5,16},{F5,16},{DS5,8},{E5,16},{OFF,32},{GS4,16},{A4,16},{C5,16},{OFF,32},{A4,16},{C5,16},{D5,16},
/*11*/{OFF,16},{G5,16},{FS5,16},{F5,16},{DS5,8},{E5,16},{OFF,32},{C6,16},{OFF,32},{C6,16},{C6,8},{OFF,8},
/*12*/{OFF,16},{G5,16},{FS5,16},{F5,16},{DS5,8},{E5,16},{OFF,32},{GS4,16},{A4,16},{C5,16},{OFF,32},{A4,16},{C5,16},{D5,16},
/*13*/{OFF,16},{DS5,8},{OFF,32},{D5,16},{OFF,16},{C5,8},{OFF,16},
/*14*/{C5,16},{C5,8},{C5,16},{OFF,32},{C5,16},{D5,8},{E5,16},{C5,8},{A4,16},{G4,8},{OFF,16},
/*15*/{C5,16},{C5,8},{C5,16},{OFF,32},{C5,16},{D5,16},{E5,16},{OFF,4},
/*16*/{C5,16},{C5,8},{C5,16},{OFF,32},{C5,16},{D5,8},{E5,16},{C5,8},{A4,16},{G4,8},{OFF,16},
/*17*/{E5,16},{E5,8},{E5,16},{OFF,32},{C5,16},{E5,8},{G5,8},{OFF,16},{G4,8},{OFF,16},
/*18*/{C5,8},{OFF,32},{G4,16},{OFF,16},{E4,8},{OFF,32},{A4,16},{OFF,32},{B4,16},{OFF,32},{AS4,16},{A4,8},
/*19*/{G4,83},{E5,83},{G5,83},{A5,8},{F5,16},{G5,16},{OFF,32},{E5,16},{OFF,32},{C5,16},{D5,16},{B4,16},{OFF,16},
/*20*/{C5,8},{OFF,32},{G4,16},{OFF,16},{E4,8},{OFF,32},{A4,16},{OFF,32},{B4,16},{OFF,32},{AS4,16},{A4,8},
/*21*/{G4,83},{E5,83},{G5,83},{A5,8},{F5,16},{G5,16},{OFF,32},{E5,16},{OFF,32},{C5,16},{D5,16},{B4,16},{OFF,16},
/*22*/{E5,16},{C5,8},{G4,16},{OFF,16},{GS4,8},{A4,16},{F5,8},{F5,16},{A4,8},{OFF,16},
/*23*/{B4,83},{A5,83},{A5,83},{A5,83},{G5,83},{F5,83},{E5,16},{C5,8},{A4,16},{G4,8},{OFF,16},
/*24*/{E5,16},{C5,8},{G4,16},{OFF,16},{GS4,8},{A4,16},{F5,8},{F5,16},{A4,8},{OFF,16},
/*25*/{B4,16},{F5,8},{F5,16},{F5,83},{E5,83},{D5,83},{C5,16},{G4,8},{G4,16},{C4,8},{OFF,16},
/*26*/{E5,16},{C5,8},{G4,16},{OFF,16},{GS4,8},{A4,16},{F5,8},{F5,16},{A4,8},{OFF,16},
/*27*/{B4,83},{A5,83},{A5,83},{A5,83},{G5,83},{F5,83},{E5,16},{C5,8},{A4,16},{G4,8},{OFF,16},
/*28*/{E5,16},{C5,8},{G4,16},{OFF,16},{GS4,8},{A4,16},{F5,8},{F5,16},{A4,8},{OFF,16},
/*29*/{B4,16},{F5,8},{F5,16},{F5,83},{E5,83},{D5,83},{C5,16},{G4,8},{G4,16},{C4,8},{OFF,16},
/*30*/{C5,16},{C5,8},{C5,16},{OFF,32},{C5,16},{D5,8},{E5,16},{C5,8},{A4,16},{G4,8},{OFF,16},
/*31*/{C5,16},{C5,8},{C5,16},{OFF,32},{C5,16},{D5,16},{E5,16},{OFF,4},
/*32*/{C5,16},{C5,8},{C5,16},{OFF,32},{C5,16},{D5,8},{E5,16},{C5,8},{A4,16},{G4,8},{OFF,16},
/*33*/{E5,16},{E5,8},{E5,16},{OFF,32},{C5,16},{E5,8},{G5,8},{OFF,16},{G4,8},{OFF,16},
/*34*/{E5,16},{C5,8},{G4,16},{OFF,16},{GS4,8},{A4,16},{F5,8},{F5,16},{A4,8},{OFF,16},
/*35*/{B4,83},{A5,83},{A5,83},{A5,83},{G5,83},{F5,83},{E5,16},{C5,8},{A4,16},{G4,8},{OFF,16},
/*36*/{E5,16},{C5,8},{G4,16},{OFF,16},{GS4,8},{A4,16},{F5,8},{F5,16},{A4,8},{OFF,16},
/*37*/{B4,16},{F5,8},{F5,16},{F5,83},{E5,83},{D5,83},{C5,16},{G4,8},{G4,16},{C4,8},{OFF,16},
/*38*/{C5,8},{OFF,32},{G4,162},{E4,8},{A4,83},{B4,83},{A4,83},{GS4,83},{AS4,83},{A4,8},
/*39*/{E4,16},{D4,16},{E4,28},{OFF,OFF}};

```

```

const unsigned char heart_and_soul[][2] PROGMEM =
/*1*/{F4,4},{F4,4},{F4,28},{F4,8},{E4,8},{D4,8},{E4,8},{F4,8},{G4,4},{A4,4},{A4,4},{A4,28},
/*2*/{A4,8},{G4,8},{F4,8},{G4,8},{A4,8},{AS4,4},{C5,2},{F4,28},{D5,8},{C5,8},{AS4,8},{A4,4},{G4,4},
/*3*/{E4,28},{G4,8},{A4,8},{AS4,8},{C4,4},{AS4,8},{A4,8},{G4,2},{F4,4},{F4,4},{F4,28},
/*4*/{F4,8},{E4,8},{D4,8},{E4,8},{F4,8},{G4,4},{A4,4},{A4,4},{A4,28},{A4,8},{G4,8},{F4,8},{G4,8},{A4,8},{AS4,4},
/*5*/{C5,2},{F4,28},{D5,8},{C5,8},{AS4,8},{A4,4},{G4,4},{E4,28},{G4,8},{A4,8},{AS4,8},{C5,4},{E4,2},{OFF,OFF}};

```

```

const unsigned char jingle_bells[][2] PROGMEM =
/*1*/{{D4,8},{B4,8},{A4,8},{G4,8},{D4,4},{OFF,16},{D4,16},{D4,16},{D4,8},{B4,8},{A4,8},{G
4,8},{E4,4},{OFF,16}},
/*2*/{{E4,8},{C5,8},{B4,8},{A4,8},{FS4,4},{OFF,16},{E5,8},{D5,8},{C5,8},{A4,8},{B4,4},{OFF
,16}},
/*3*/{{D4,8},{B4,8},{A4,8},{G4,8},{D4,4},{OFF,8},{D4,8},{D4,8},{B4,8},{A4,8},{G4,8},{E4,4}
,{OFF,16},{E4,8}},
/*4*/{{E4,8},{C5,8},{B4,8},{A4,8},{D5,8},{D5,8},{D5,8},{D5,8},{E5,8},{D5,8},{C5,8},{A4,8},
{G4,4},{OFF,16}},
/*5*/{{B4,8},{B4,8},{B4,4},{B4,8},{B4,8},{B4,4},{B4,8},{D5,8},{G4,82},{A4,16},{B4,4},{OFF,
16},{C5,8},{C5,8},{C5,82},{C5,16},{C5,8},{B4,8},{B4,8},{B4,16},{B4,16}},
/*6*/{{B4,8},{A4,8},{A4,8},{G4,8},{A4,4},{D5,4},{B4,8},{B4,8},{B4,4},{B4,8},{B4,8},{B4,4},
{B4,8},{D5,8},{G4,82},{A4,16},{B4,4},{OFF,16}},
/*7*/{{C5,8},{C5,8},{C5,82},{C5,16},{C5,8},{B4,8},{B4,8},{B4,16},{B4,16},{D5,8},{D5,8},{C5
,8},{A4,8},{G4,42},{OFF,8},{OFF,OFF}}};

const unsigned char god_rest[][2] PROGMEM =
/*1*/{{E4,4},{E4,4},{B4,4},{B4,4},{A4,4},{G4,4},{FS4,4},{E4,4},{D4,4},{E4,4},{FS4,4},{G4,
4},{A4,4},{B4,2},{OFF,16}},
/*2*/{{E4,4},{E4,4},{B4,4},{B4,4},{A4,4},{G4,4},{FS4,4},{E4,4},{D4,4},{E4,4},{FS4,4},{G4,4}
,{A4,4},{B4,2},{OFF,8}},
/*3*/{{B4,4},{C5,4},{A4,4},{B4,4},{C5,4},{D5,4},{E5,4},{B4,4},{A4,4},{G4,4},{E4,4},{FS4,4}
,{G4,4},{A4,2},{G4,4},{A4,4}},
/*4*/{{B4,2},{C5,4},{B4,4},{A4,4},{G4,4},{FS4,4},{E4,2},{G4,8},{FS4,8},{E4,4}},
/*5*/{{A4,2},{G4,4},{A4,4},{B4,4},{C5,4},{D5,4},{E5,4},{B4,4},{A4,4},{G4,4},{FS4,4},{E4,2}
,{OFF,OFF}}};

/* Global Variables */
volatile unsigned char note=OFF;
unsigned char xoldz;
unsigned char mode_0_number=0;
unsigned char pass0=0x00,pass1=0x00,pass2=0x00,pass3=0x00;
unsigned char state0=0,state1=0,state2=0,state3=0;

int main(void)
{
    //Initialization
    sei(); //Enable global interrupts
    TCCR0 = 0x03; //Prescaler = 64
    TIMSK |= 0x02; //Enable TCNT0
    DDRC = 0xFF;
    PORTC = 0x00;
    DDRA = 0xF0;
    PORTA = 0x0F;
    unsigned char x,count=0;
    while(1)
    {
        PORTA |= 0xFF; //Turn on all LEDs
        x = read_switches();
        if(x==pass0) state0=1;
        x = read_switches();
        if(x==pass1) state1=1;
        x = read_switches();
        if(x==pass2) state2=1;
        x = read_switches();
        if(x==pass3) state3=1;
        PORTA &= 0x0F; //Pull switches back up
        while(state0 && state1 && state2 && state3)
        {
            PORTA |= 0x60;
            PORTA &= 0x6F;
            x = read_switches();
            switch(x)
            {

```

```

        case 0:
            number();
            break;
        case 1:
            music();
            break;
        case 2:
            password_reset();
            break;
        case 3:
            state0 = 0;
            state1 = 0;
            state2 = 0;
            state3 = 0;
            break;
        default:
            break;
    }
}
PORTA = 0x0F;
for(count=0; count<7; count++)
{
    _delay_ms(50);
    PORTA = ~PORTA;
}
}
return 0;
}

/* Interrupt Service Routine */
ISR(TIMER0_OVF_vect)
{
    TCNT0 = note;
    if(TCNT0!=0xFF)
        PORTC ^= SPKR;
}

/* Function to Read Switch Presses */
unsigned char read_switches(void)
{
    unsigned char x, x_switch=0, pressed=0, end=0;
    while(!end)
    {
        x = PINA;
        x &= 0x0F;
        if((x!=xoldz)&&(x!=0x0F)){ //SW pressed
            _delay_ms(10);
            x = PINA;
            if((x!=xoldz)&&(x!=0x0F)){
                x_switch = x;
                pressed=1;
            }
        }
        xoldz = x;
        if(pressed==1 && x!=x_switch && decode_switches(x_switch)!=0x04) //Will only
return once switch is released & has valid value
            end=1;
    }
    return decode_switches(x_switch);
}

/* Switch Decoding Function */
unsigned char decode_switches(unsigned char x)
{
    switch(x){
        case 0x0E:

```

```

        return 0x00;
    case 0x0D:
        return 0x01;
    case 0x0B:
        return 0x02;
    case 0x07:
        return 0x03;
    default:
        return 0x04;
    }
}

/* Function to Manipulate Number */
int number(void)
{
    unsigned char x=0;
    PORTA |= 0x0F;
    PORTA &= 0x0F;
    PORTA |= mode_0_number<<4;    //Displays current number value to LEDs
    while(x!=3)
    {
        PORTA |= mode_0_number<<4;    //Displays current number value to LEDs
        x = read_switches();
        PORTA = 0x0F;
        switch(x){
            case 0:
                mode_0_number = 0;    //Resets number value
                break;
            case 1:
                if(mode_0_number==15) mode_0_number=0;
                else mode_0_number++;    //Increments number by 1
                break;
            case 2:
                if(mode_0_number>0) mode_0_number--;
                break;    //Decreases number by 1
            default:
                break;
        }
    }
    return 0;
}

/* Musical Selection Function */
int music(void)
{
    PORTA = 0x0F;
    PORTA |= 0x9F;
    unsigned char song1=0,song2=0;
    while(song1!=3 && song2!=3)
    {
        PORTA = 0x9F;
        song1 = read_switches();
        switch(song1){
            case 0:
                PORTA = 0x3F;
                song2 = read_switches();
                switch(song2){
                    case 0:
                        mario_song();
                        break;
                    case 1:
                        heartandsoul();
                        break;
                    case 2:
                        three_note_keyboard();
                        break;
                    default:

```

```

        break;
    }
    break;
case 1:
    PORTA = 0xCF;
    song2 = read_switches();
    switch(song2){
        case 0:
            jinglebells();
            break;
        case 1:
            godrest();
            break;
        case 2:
            three_note_keyboard();
            break;
        default:
            break;
    }
    break;
case 2:
    three_note_keyboard();
    break;
default:
    break;
}
}
PORTA = 0x0F;
return 0;
}

/* Function Used to Reset the Password */
int password_reset(void)
{
    PORTA = 0x0F;
    unsigned char pw_reset=0;
    unsigned char reset[4];
    while(!pw_reset)
    {
        PORTA |= 0x2F;
        _delay_ms(500);
        reset[0] = read_switches();
        reset[1] = read_switches();
        reset[2] = read_switches();
        reset[3] = read_switches();
        pass0 = reset[0];
        pass1 = reset[1];
        pass2 = reset[2];
        pass3 = reset[3];
        pw_reset = 1;
    }
    PORTA = 0x0F;
    return 0;
}

/* Three Key Keyboard */
int three_note_keyboard(void)
{
    PORTA = 0x0F;
    PORTA |= 0x4F;
    unsigned char xkey=0;
    while(xkey!=3)
    {
        xkey = decode_switches(PINA);
        note = pgm_read_byte(&keyboard_notes[xkey]);
    }
    PORTA = 0x0F;
}

```

```

        _delay_ms(150);
        return 0;
    }

    /* Plays Mario Theme Song */
    int mario_song(void)
    {
        int x=0;
        unsigned char length=0;
        unsigned char end=0;
        PORTA = 0x0F;
        while(end!=3 && length!=0xFF)
        {
            end = decode_switches( PINA );
            if(length!=0xFF){
                note = pgm_read_byte(&mario[x][0]);
                if(note==C6) PORTA |= 0xF0;
                if(note==A5) PORTA |= 0xE0;
                if(note==G5) PORTA |= 0xD0;
                if(note==FS5) PORTA |= 0xC0;
                if(note==F5) PORTA |= 0xB0;
                if(note==E5) PORTA |= 0xA0;
                if(note==DS5) PORTA |= 0xB0;
                if(note==D5) PORTA |= 0xA0;
                if(note==C5) PORTA |= 0x90;
                if(note==B4) PORTA |= 0x80;
                if(note==AS4) PORTA |= 0x70;
                if(note==A4) PORTA |= 0x60;
                if(note==GS4) PORTA |= 0x50;
                if(note==G4) PORTA |= 0x40;
                if(note==E4) PORTA |= 0x30;
                if(note==D4) PORTA |= 0x20;
                if(note==C4) PORTA |= 0x10;
                length = pgm_read_byte(&mario[x][1]);
                if(length==1) _delay_ms(2400);
                if(length==28) _delay_ms(1500);
                if(length==2) _delay_ms(1200);
                if(length==42) _delay_ms(900);
                if(length==4) _delay_ms(600);
                if(length==8) _delay_ms(300);
                if(length==162) _delay_ms(225);
                if(length==83) _delay_ms(200);
                if(length==16) _delay_ms(150);
                if(length==32) _delay_ms(75);
                if(length==100) _delay_ms(1);
                if(note!=0xFF){
                    note = 0xFF;
                    PORTA &= 0x0F;
                    _delay_ms(1);
                }
                x++;
            }
        }
        return 0;
    }

    /* Plays Heart and Soul */
    int heartandsoul(void)
    {
        unsigned char x=0;
        unsigned char length=0, end=0;
        PORTA = 0x0F;
        while(end!=3 && length!=0xFF)
        {
            end = decode_switches( PINA );
            if(length!=0xFF){
                note = pgm_read_byte(&heart_and_soul[x][0]);

```

```

        if(note==D5) PORTA |= 0xF0;
        if(note==C5) PORTA |= 0x30;
        if(note==A5) PORTA |= 0x60;
        if(note==A4) PORTA |= 0x10;
        if(note==G4) PORTA |= 0x20;
        if(note==F4) PORTA |= 0x40;
        if(note==E4) PORTA |= 0x80;
        if(note==D4) PORTA |= 0xC0;
        length = pgm_read_byte(&heart_and_soul[x][1]);
        if(length==1) _delay_ms(2000);
        if(length==28) _delay_ms(1250);
        if(length==2) _delay_ms(1000);
        if(length==4) _delay_ms(500);
        if(length==8) _delay_ms(250);
        if(length==16) _delay_ms(125);
        if(note!=0xFF){
            note = 0xFF;
            PORTA &= 0x0F;
            _delay_ms(5);
        }
        x++;
    }
}
PORTA = 0x0F;
return 0;
}

/* Plays Jingle Bells */
int jinglebells(void)
{
    unsigned char x=0;
    unsigned char length=0,end=0;
    PORTA = 0x0F;
    while(end!=3 && length!=0xFF)
    {
        end = decode_switches( PINA );
        if(length!=0xFF){
            note = pgm_read_byte(&jingle_bells[x][0]);
            if(note==E5) PORTA |= 0xF0;
            if(note==D5) PORTA |= 0x30;
            if(note==C5) PORTA |= 0x10;
            if(note==B4) PORTA |= 0x20;
            if(note==A4) PORTA |= 0x60;
            if(note==G4) PORTA |= 0x40;
            if(note==F5) PORTA |= 0x80;
            if(note==E4) PORTA |= 0xC0;
            if(note==D4) PORTA |= 0xE0;
            length = pgm_read_byte(&jingle_bells[x][1]);
            if(length==1) _delay_ms(2000);
            if(length==2) _delay_ms(1000);
            if(length==42) _delay_ms(750);
            if(length==4) _delay_ms(500);
            if(length==82) _delay_ms(375);
            if(length==8) _delay_ms(250);
            if(length==16) _delay_ms(125);
            if(note!=0xFF){
                note = 0xFF;
                PORTA &= 0x0F;
                _delay_ms(5);
            }
            x++;
        }
    }
    return 0;
}

```

```

/* Plays God Rest Ye Merry, Gentlemen*/
int godrest(void)
{
    unsigned char x=0;
    unsigned char length=0, end=0;
    PORTA = 0x0F;
    while(end!=3 && length!=0xFF)
    {
        end = decode_switches( PINA );
        if(length!=0xFF){
            note = pgm_read_byte(&god_rest[x][0]);
            if(note==E5) PORTA |= 0xF0;
            if(note==D5) PORTA |= 0x30;
            if(note==C5) PORTA |= 0x10;
            if(note==B4) PORTA |= 0x20;
            if(note==A4) PORTA |= 0x60;
            if(note==G4) PORTA |= 0x40;
            if(note==FS4) PORTA |= 0x80;
            if(note==E4) PORTA |= 0xC0;
            if(note==D4) PORTA |= 0xE0;
            length = pgm_read_byte(&god_rest[x][1]);
            if(length==1) _delay_ms(1400);
            if(length==2) _delay_ms(700);
            if(length==4) _delay_ms(350);
            if(length==8) _delay_ms(175);
            if(length==16) _delay_ms(88);
            if(note!=0xFF){
                note = 0xFF;
                PORTA &= 0x0F;
                _delay_ms(1);
            }
            x++;
        }
    }
    return 0;
}

```