

Assignment 2

Group members

Carson Bollinger

Carson Wagner

Roberto Solis

Objective: Use the LPC 1769 and MCUXpresso IDE to create a game of Nim, where two players take turns pressing switches to turn off 5 rows of LEDs colored red, green, and blue. The player that turns off the last LED loses the game. There also is a score board that will count player 1 and player 2 wins. Each player will have 2 LEDs to indicate a win.

Operational Current Limitations Calculations:

$$\frac{75\text{mA}}{19\text{LEDs}} = 3.95\text{mA for each LED}$$

Red: $R = \frac{3.3 - 2.0}{3.95\text{mA}} = 329.11\ \Omega$

Green: $R = \frac{3.3 - 2.1}{3.95\text{mA}} = 303.80\ \Omega$

Blue: $R = \frac{3.3 - 3.2}{3.95\text{mA}} = 25.32\ \Omega$

$$I_R = \frac{(3.3 - 2.0)}{220\ \Omega} = 0.0059\ \text{A} \cdot 5\text{LEDs} = 0.030\text{A}$$

$$I_G = \frac{(3.3 - 2.1)}{220\ \Omega} = 0.0055\ \text{A} \cdot 5\text{LEDs} = 0.0275\text{A}$$

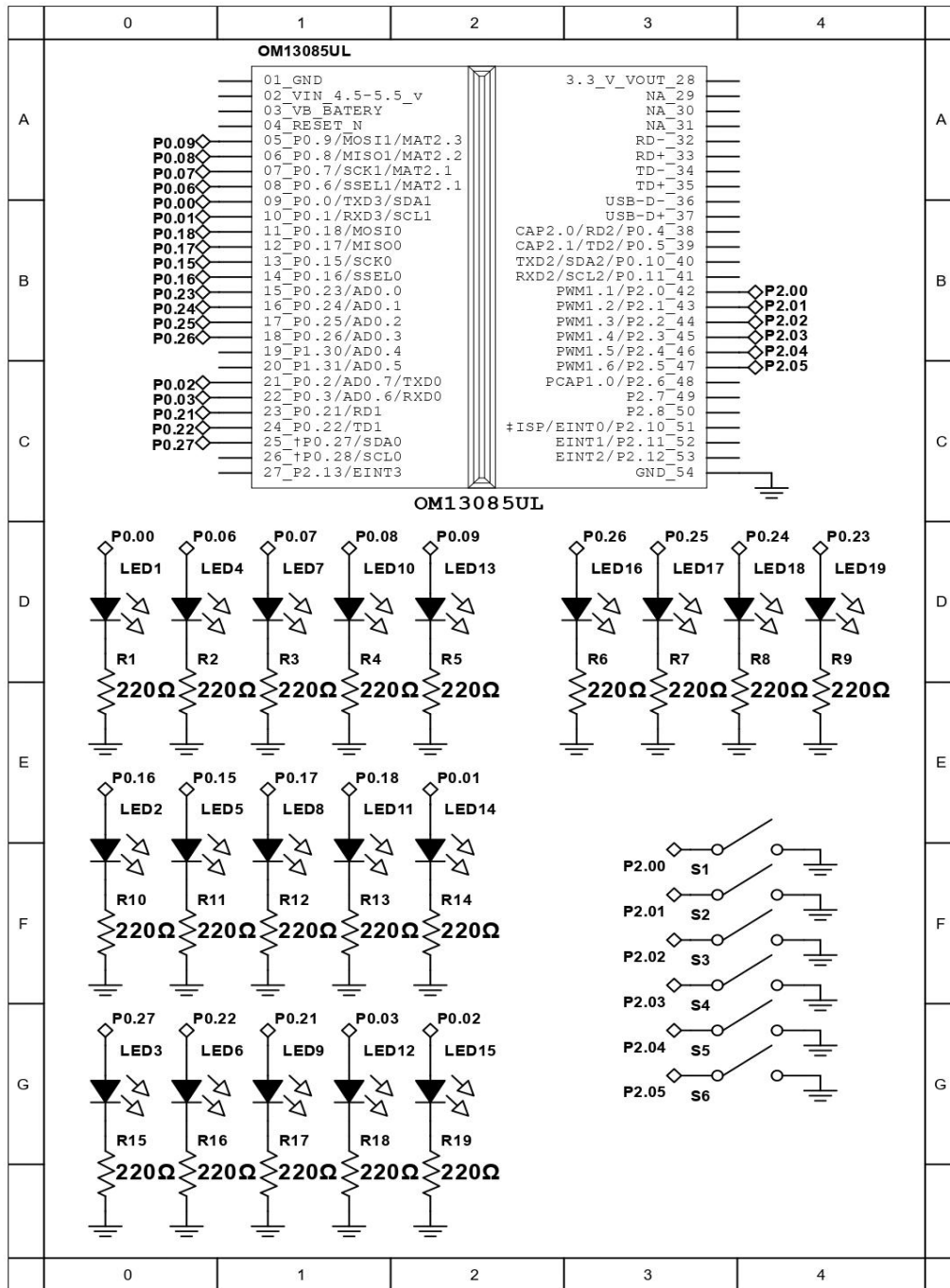
$$I_B = \frac{(3.3 - 3.2)}{220} = 0.0045\ \text{A} \cdot 9\text{LEDs} = 0.0041\text{A}$$

$$\begin{array}{r} 0.030\text{A} \\ 0.0275\text{A} \\ + 0.0041\text{A} \\ \hline 0.0616\text{A} \end{array} \rightarrow$$

$$I_{\text{Total}} = 61.6\text{mA}$$

\downarrow
 $61.6\text{mA} < 75\text{mA Limit}$
So Works! Under limit.

Schematic: Here below we can see the schematic we implemented with the LPC1769 microcontroller. Our Setup incorporates 19 LEDs, where we used 220-ohm resistors for each LED.



Individual Contributions:

Carson Wagner (Computer Engineering) – Worked on choosing which pins to use for the LEDs and switches, also worked on the code for the game of NIM, mainly worked on the switch logic along. Solved for the operational current limitation.

Carson Bolinger (Electrical Engineering) - Worked on the schematic and code, mainly working on the LEDs turning on and off and the switch logic.

Roberto Solis (Electrical Engineering) - Worked on the hardware and helped with code, mainly being switch and LED logic.

Lab Demonstration:

ECE 4273

Lab Demonstration Sign-off

Assignment Number	Assignment 2
Team Members Demoing	Carson Wagner Carson Bulinger Roberto Solis
	Carson Bulinger
	Roberto Solis
Date	2/8/24
Time	7:51 pm
Witnessed by	Jed

Were all objectives completed?

☒ Yes

☐ No

If "No", describe which objectives were completed or not completed (whichever is easiest):

Code:

```

* Copyright 2022 NXP
* NXP confidential.
* This software is owned or controlled by NXP and may only be used strictly
* in accordance with the applicable license terms. By expressly accepting
* such terms or by downloading, installing, activating and/or otherwise using
* the software, you are agreeing that you have read, and that you agree to
* comply with and are bound by, such license terms. If you do not agree to
* be bound by the applicable license terms, then you may not retain, install,
* activate or otherwise use the software.
*/

#ifdef __USE_CMSIS
#include "LPC17xx.h"
#endif

#include <cr_section_macros.h>

#include <stdio.h>

//initialize registers
#define FIO0DIR (*(volatile unsigned int *)0x2009c000) //blue; 00, 06, 07, 08, 09
#define FIO0PIN (*(volatile unsigned int *)0x2009c014) //green; 01, 15, 16, 17, 18
//red; 02, 03, 21, 22, 27
//score; 23, 24, 25, 26

#define FIO2DIR (*(volatile unsigned int *)0x2009c040) //P1 switch; 00, 01, 02
#define FIO2PIN (*(volatile unsigned int *)0x2009c054) //P2 switch; 03, 04, 05

//initialize variables for wait function
float m = 0.002715;
float b = 0.1;

//initialize board LEDs
int countRed = 5;
int countGreen = 5;
int countBlue = 5;

//initialize score LEDs
int plplay = 0;
int plscore = 0;
int p2score = 0;

//lockout ints
int plr1 = 1;
int plr2 = 1;
int plr3 = 1;
int p2r1 = 1;
int p2r2 = 1;
int p2r3 = 1;

//set blues function
void blue(int num){
switch (num){
case 0:
FIO0PIN &= ~( (1 << 9) | (1 << 8) | (1 << 7) | (1 << 6) | (1 << 0));
break;
case 1:
FIO0PIN |= (1 << 9);
break;
case 2:
FIO0PIN |= (1 << 9) | (1 << 8);
break;
case 3:
FIO0PIN |= (1 << 9) | (1 << 8) | (1 << 7);
break;
case 4:

```

```

FIOOPIN |= (1 << 9) | (1 << 8) | (1 << 7) | (1 << 6);
break;
case 5:
FIOOPIN |= (1 << 9) | (1 << 8) | (1 << 7) | (1 << 6) | (1 << 0);
break;
default:
for (int k = 0; k < 5; k++){
FIOOPIN &= 0;
wait_ms(5);
FIOOPIN |= ~(0);
}
break;
}
}

//set greens function
void green(int num){
switch (num){
case 0:
FIOOPIN &= ~( (1 << 1) | (1 << 18) | (1 << 17) | (1 << 15) | (1 << 16));
break;
case 1:
FIOOPIN |= (1 << 1);
break;
case 2:
FIOOPIN |= (1 << 1) | (1 << 18);
break;
case 3:
FIOOPIN |= (1 << 1) | (1 << 18) | (1 << 17);
break;
case 4:
FIOOPIN |= (1 << 1) | (1 << 18) | (1 << 17) | (1 << 15);
break;
case 5:
FIOOPIN |= (1 << 1) | (1 << 18) | (1 << 17) | (1 << 15) | (1 << 16);
break;
default:
for (int k = 0; k < 5; k++){
FIOOPIN &= 0;
wait_ms(5);
FIOOPIN |= ~(0);
}
break;
}
}

//set reds function
void red(int num){
switch (num){
case 0:
FIOOPIN &= ~( (1 << 2) | (1 << 3) | (1 << 21) | (1 << 22) | (1 << 27));
break;
case 1:
FIOOPIN |= (1 << 2);
break;
case 2:
FIOOPIN |= (1 << 2) | (1 << 3);
break;
case 3:
FIOOPIN |= (1 << 2) | (1 << 3) | (1 << 21);
break;
case 4:
FIOOPIN |= (1 << 2) | (1 << 3) | (1 << 21) | (1 << 22);
break;
case 5:
FIOOPIN |= (1 << 2) | (1 << 3) | (1 << 21) | (1 << 22) | (1 << 27);
break;
default:
for (int k = 0; k < 5; k++){

```

```

FIOOPIN &= 0;
wait_ms(5);
FIOOPIN |= ~(0);
}
break;
}
}

//set reds function
void setP1score(int num){
switch (num){
case 0:
FIOOPIN &= ~((1 << 23) | (1 << 24));
break;
case 1:
FIOOPIN |= (1 << 23);
break;
case 2:
FIOOPIN |= (1 << 23) | (1 << 24);
break;
default:
for (int k = 0; k < 5; k++){
FIOOPIN &= 0;
wait_ms(5);
FIOOPIN |= ~(0);
}
break;
}
}

//set reds function
void setP2score(int num){
switch (num){
case 0:
FIOOPIN &= ~((1 << 25) | (1 << 26));
break;
case 1:
FIOOPIN |= (1 << 25);
break;
case 2:
FIOOPIN |= (1 << 25) | (1 << 26);
break;
default:
for (int k = 0; k < 5; k++){
FIOOPIN &= 0;
wait_ms(5);
FIOOPIN |= ~(0);
}
break;
}
}

// Function to create a delay in milliseconds
void wait_ms(int ms) {
volatile int i;

ms = (ms-b)/m;

for (i = 0; i < ms; i++) {
//do nothing
}
}

int main(void) {
printf("Assignment Two");
FIO2DIR &= (0); //TODO

```

```

FIO0DIR |= (1 << 0) | (1 << 6) | (1 << 7) | (1 << 8) | (1 << 9)
| (1 << 16) | (1 << 15) | (1 << 17) | (1 << 18) | (1 << 1)
| (1 << 27) | (1 << 22) | (1 << 21) | (1 << 3) | (1 << 2)
| (1 << 26) | (1 << 25) | (1 << 24) | (1 << 23);

// Enter an infinite loop
while (1) {
    //flash to initiate game
    for (int j = 0; j < 6; j++){
        blue(j);
        wait_ms(100);
        green(j);
        wait_ms(100);
        red(j);
        wait_ms(100);
    }
    plscore = 0;
    setPlscore(plscore);
    p2score = 0;
    setP2score(p2score);
    while ((plscore < 2) && (p2score < 2)){
        countRed = 5;
        countGreen = 5;
        countBlue = 5;
        while ((countRed + countGreen + countBlue) > 0){
            if ((~(FIO2PIN >> 0) & 1) && (plr1 == 1) && (countRed > 0)) {
                while ((~(FIO2PIN >> 0) & 1)){
                    wait_ms(5);
                }
                if (countRed > 0){
                    countRed--;
                    plplay = 1;
                }
                plr1 = 1;
                plr2 = 0;
                plr3 = 0;
                p2r1 = 1;
                p2r2 = 1;
                p2r3 = 1;
            }
            else if ((~(FIO2PIN >> 1) & 1) && (plr2 == 1) && (countGreen > 0)) {
                while ((~(FIO2PIN >> 1) & 1)){
                    wait_ms(5);
                }
                if (countGreen > 0){
                    countGreen--;
                    plplay = 1;
                }
                plr1 = 0;
                plr2 = 1;
                plr3 = 0;
                p2r1 = 1;
                p2r2 = 1;
                p2r3 = 1;
            }
            else if ((~(FIO2PIN >> 2) & 1) && (plr3 == 1) && (countBlue > 0)) {
                while ((~(FIO2PIN >> 2) & 1)){
                    wait_ms(5);
                }
                if (countBlue > 0){
                    countBlue--;
                    plplay = 1;
                }
                plr1 = 0;
                plr2 = 0;
                plr3 = 1;
                p2r1 = 1;
                p2r2 = 1;
                p2r3 = 1;
            }
        }
    }
}

```



```

else if ((~(FIO2PIN >> 3) & 1) && (p2r1 == 1) && (countRed > 0)) {
while ((~(FIO2PIN >> 3) & 1)){
wait_ms(5);
}
if (countRed > 0){
countRed--;
plplay = 0;
}
plr1 = 1;
plr2 = 1;
plr3 = 1;
p2r1 = 1;
p2r2 = 0;
p2r3 = 0;
}
else if ((~(FIO2PIN >> 4) & 1) && (p2r2 == 1) && (countGreen > 0)) {
while ((~(FIO2PIN >> 4) & 1)){
wait_ms(5);
}
if (countGreen > 0){
countGreen--;
plplay = 0;
}
plr1 = 1;
plr2 = 1;
plr3 = 1;
p2r1 = 0;
p2r2 = 1;
p2r3 = 0;
}
else if ((~(FIO2PIN >> 5) & 1) && (p2r3 == 1) && (countBlue > 0)) {
while ((~(FIO2PIN >> 5) & 1)){
wait_ms(5);
}
if (countBlue > 0){
countBlue--;
plplay = 0;
}
plr1 = 1;
plr2 = 1;
plr3 = 1;
p2r1 = 0;
p2r2 = 0;
p2r3 = 1;
}
red(0);
red(countRed);
wait_ms(50);
green(0);
green(countGreen);
wait_ms(50);
blue(0);
blue(countBlue);
wait_ms(50);
}
if (plplay == 1){
plscore++;
setPlscore(0);
setPlscore(plscore);
}
else {
p2score++;
setP2score(0);
setP2score(p2score);
}
}
for (int y = 0; y < 10; y++){
wait_ms(100);
setPlscore(0);
setP2score(0);
}

```

```
wait_ms(100);  
setP1score(p1score);  
setP2score(p2score);  
}  
}  
}
```