EE4930 Advanced Microprocessor

Section 011/021, Winter 2022/23

Professor: Kerry R. Widder, Ph.D.

Electrical Engineering and Computer Science Department

Milwaukee School of Engineering

Laboratory 2: “Compiler Optimization”

Name: Jorge Jurado-Garcia

Courses: EE 4930 011

Date: 01/6/2023

# Objectives

This lab objective is to gain experience setting optimization levels in a compiler and analyze the effectiveness of various optimization settings. Optimization levels range form 0-4 size vs. speed levels ranging from 0-5 being tested.

# Description

For this lab, I implemented Nested loops, with inner loops small and having minimal iterations, Modulo and multiplying calculations, small functionals and arithmetic. Reading the MAP file located in the debug directory of my project I was able to take data of the size allocation that the compiler. Code composer can optimize by unrolling for loops, disregarding unused code, turn small functions into in-line code, and can simplify expressions. Various unoptimized codes were added to the program to experiment.

# Conclusion

Overall, this lab was a success, the optimization level will decrease the size of the code , surprisingly, the size sometimes increased when the speed was increased. This caused a Hugh tradeoff of increasing speed size from 30 ms to 1.5 ms. The table below shows the size and speed of the program based off various optimization levels. However, when increasing the optimization levels will decrease the size of the program which will positively effect to users code size and speed.

# Attachments

## Optimization Table

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Optimization setting** | **size/speed setting** | **Module code size** | **Percent change** | **Total code size** | **Percent change** | **Module ro data size** | **Total ro data size** | **Module rw data size** | **Total rw data size** | **run time [ms]** | **Percent change** |
| OFF | 0 | 2400 |  | 13215 |  | 276 | 680 | 8 | 4968 | 30.37 |  |
| 0 | 0 | 1344 | -44.0% | 12159 | -8.0% | 276 | 677 | 8 | 4968 | 19.28 | -36.5% |
| 2 | 0 | 1136 | -52.7% | 11857 | -10.3% | 276 | 675 | 8 | 4968 | 4.32 | -85.8% |
| 4 | 0 | 238 | -90.1% | 3482 | -73.7% | 276 | 400 | 0 | 4928 | 3.84 | -87.4% |
| 0 | 2 | 1472 | -38.7% | 12287 | -7.0% | 276 | 678 | 8 | 4968 | 19.54 | -35.7% |
| 2 | 2 | 1252 | -47.8% | 11973 | -9.4% | 276 | 675 | 8 | 4968 | 4.31 | -85.8% |
| 4 | 2 | 202 | -91.6% | 3674 | -72.2% | 276 | 402 | 0 | 4928 | 3.43 | -88.7% |
| 0 | 3 | 1710 | -28.8% | 12525 | -5.2% | 276 | 678 | 8 | 4968 | 19.15 | -36.9% |
| 2 | 3 | 1524 | -36.5% | 12245 | -7.3% | 276 | 677 | 8 | 4968 | 2.6 | -91.4% |
| 4 | 3 | 208 | -91.3% | 3680 | -72.2% | 276 | 402 | 0 | 4928 | 1.5 | -95.1% |
| 0 | 5 | 1828 | -23.8% | 12643 | -4.3% | 276 | 678 | 8 | 4968 | 19.02 | -37.4% |
| 2 | 5 | 1586 | -33.9% | 12363 | -6.4% | 276 | 678 | 8 | 4968 | 2.2 | -92.8% |
| 4 | 5 | 208 | -91.3% | 3640 | -72.5% | 276 | 403 | 0 | 4928 | 1.3 | -95.7% |

## Main

/\* Copyright (C) 2022 MSOE

\*

\* All Rights Reserved

\* You may not use, distribute or modify this code without the

\* express written permission of MSOE

\*

\* Contact Info

\* jorgejuradogarcia2@gmail.com

\* 608-312-5950

\*

\*/

**#include** <stdio.h>

**#include** "msp432.h"

**#include** "msoe\_lib\_all.h"

**#include** "defines.h"

**#include** <MSP432P401R\_GPIO.h>

// Lab modules

**#include** "EE4930\_LAB1.h"

**#include** "EE4930\_LAB2.h"

**#include** "EE4930\_LAB3.h"

**#include** "math.h"

**#include**"msoe\_lib\_clk.h"

**#include**"msoe\_lib\_delay.h"

**#include**"math.h"

**#include**"msoe\_lib\_lcd.h"

**#include** "math.h"

**#include**"msoe\_lib\_clk.h"

**#include**"msoe\_lib\_delay.h"

**#include**"math.h"

**#include**"msoe\_lib\_lcd.h"

**#define** LAB1\_ON FALSE

**#define** LAB2\_ON FALSE

**#define** LAB3\_ON TRUE

**#define** CLEAR 21

//global variables that are not used

**unsigned** adc\_val;

**float** ADC\_Percentage;

**float** PWM\_dutycycle;

**char** LCD\_SET\_FLAG=0;

**int** Calculated\_dutycycle;

intrand\_num = 0;

**int** **main**(**void**){

**#if** LAB3\_ON

uint32\_t counter = 0;

setup();

Small\_Loops\_rolling();

Calculation\_loop();

div\_multi\_loop();

Complex\_if\_else();

function\_prototype\_only();

**if**( counter > 0)

{

never\_used\_function\_1();

never\_used\_function\_2();

never\_used\_function\_3();

never\_used\_function\_4();

}

//set P2.3 to low

GPIO\_setOutputLowOnPin( GPIO\_PORT\_P2, GPIO\_PIN3 ); //Set P2.3 Low

**#endif**

} // end main

## EE4930\_LAB3.h

/\*

\* EE4930\_LAB3.h

\*

\* Created on: Dec 31, 2022

\* Author: jurado-garciaj

\*/

**#ifndef** APP\_EE4930\_LAB3\_H\_

**#define** APP\_EE4930\_LAB3\_H\_

// This can simply optimized by simply hardcoding this values.

**void** **Small\_Loops\_rolling**( **void** );

//calculation loops

**void** **Calculation\_loop**( **void** );

//division and multiplication math

**int** **div\_multi\_loop**( **void** );

**void** **Complex\_if\_else** ( **void** );

//setup function

**void** **setup**( **void** );

//not used fuctions

**int** **never\_used\_function\_1**( **void** );

**int** **never\_used\_function\_2**( **void** );

**int** **never\_used\_function\_3**( **void** );

**int** **never\_used\_function\_4**( **void** );

//function without data

**void** **function\_prototype\_only**( **void**);

**#endif** /\* APP\_EE4930\_LAB3\_H\_ \*/

## EE4930\_LAB3.c

/\*

\* EE4930\_LAB.c

\*

\* Created on: Dec 31, 2022

\* Author: jurado-garciaj

\*/

**#include** "EE4930\_LAB3.h"

**#include** "msp.h"

**#include** "stdio.h"

**#include** <MSP432P401R\_GPIO.h>

**void** **function\_prototype\_only**( **void**)

{

**return**;

}

**void** **Small\_Loops\_rolling**( **void** )

{

**int** fact[5];

fact[0] = 1;

**int** i=0;

// Overhead of managing a counter

// just for 4 iterations

// is not a good idea

**for** ( i = 1; i < 5; ++i)

{

fact[i] = fact[i - 1] \* i;

}

**return**;

}

**void** **Calculation\_loop**( **void** )

{

**int** arr[10000];

**int** a = 1;

**int** b = 5;

**int** c = 2;

uint32\_t i;

// Calculating a constant expression

// for each iteration is not good.

//Better way to pre-calculat the constant expression

**for** (i = 0; i < 10000; ++i)

{

arr[i] = ( ((c % a) \* a / b) % c) \* i;

}

**return**;

}

//division and multiplication math

**int** **div\_multi\_loop**( **void** )

{

uint32\_t stuff = 0;

**while**(0)

{

//do some stupid studd

**if**( stuff < 1 )

{

stuff = 12321089;

stuff = stuff << 100;

}

**else**

{

//for loop for nothing

**while**(0);

}

}

**int** a = 100;

**int** b = 1;

**int** c =23103-1;

**int** d = a/ (b\*a-c/2) %c;

**return** 0;

}

//setup function for GPIO PIN to read speed data

**void** **setup**( **void** )

{

//enable gpio pin to output and set to high

GPIO\_setAsOutputPin( GPIO\_PORT\_P2, GPIO\_PIN3); // P2.3 as an output pin

GPIO\_setOutputLowOnPin( GPIO\_PORT\_P2, GPIO\_PIN3 ); //Set P2.3 Low

GPIO\_setOutputHighOnPin( GPIO\_PORT\_P2, GPIO\_PIN3 ); //Set P2.3 Low

}

**void** **Complex\_if\_else** ( **void** )

{

uint32\_t rand = 21321;

**if**(rand < -1)

{

**printf**("hello world\n");

}

**else** **if**(rand < -1)

{

**printf**("hello world\n");

}

**else** **if**(rand < 40)

{

**printf**("hello world\n");

}

**else** **if**(rand < 100)

{

**printf**("hello world\n");

}

**else** **if**(rand < 1200)

{

**printf**("hello world\n");

}

**else** **if**(rand < 2100)

{

**printf**("hello world\n");

}

**else** **if**(rand < 2400)

{

**printf**("hello world\n");

}

**else** **if**(rand < 3500)

{

**printf**("hello world\n");

}

**else** **if**(rand < 3300)

{

**printf**("hello world\n");

}

**else** **if**(rand < 3500)

{

**printf**("hello world\n");

}

**else** **if**(rand < 3300)

{

**printf**("hello world\n");

}

**else**{**printf**("hello world\n");

}

}

**int** **never\_used\_function\_1**( **void** )

{

//Screen char array that won't be used

**static** **const** **char** start[] = {

0x00, 0x00, 0x00, 0x00, 0x00,

0x00,0x00, 0x00, 0x00, 0xe0, 0xf0,

0x48, 0x88,0x84, 0x84, 0x7c, 0x04,

0x84, 0x84, 0x8c,0x8c, 0x94, 0xb4,

0xe4, 0xa4, 0xa4, 0x28,0x48, 0x70,

0xe0, 0x00, 0x00, 0x00, 0x00,0x00,

0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00,0x00, 0x00, 0x00, 0x00, 0x00,

0x00,0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00,0x00, 0xf8, 0x8e, 0xe1,

0xf8, 0xfc, 0x7c,0x44, 0x44, 0x44,

0x44, 0x45, 0x25, 0x25

};

}

**int** **never\_used\_function\_2**( **void** )

{

uint32\_t i;

P4->OUT^= 0b00111110;

**if**(P4->OUT& 0b00111110)

{

;

}

**else**

{

**printf**("hi\n");

}

**for**(i = 10000; i > 0; i--);

**return** 0;

}

**int** **never\_used\_function\_3**( **void** )

{

uint32\_t i;

**for**(i = 10000; i > 0; i--)

{

**printf**("hello world");

}

never\_used\_function\_3();

}

**int** **never\_used\_function\_4**( **void** )

{

never\_used\_function\_3();

}