МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

**НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Отчет по лабораторной работе №2

по курсу «Архитектура современных микропроцессоров и мультипроцессоров»

**ОПРЕДЕЛЕНИЕ РАЗМЕРА БУФЕРА ПЕРЕУПОРЯДОЧИВАНИЯ КОМАНД ПРОЦЕССОРА**

**Выполнил:** студент 3-го курса гр. 17208

Гафиятуллин А.Р

Новосибирск, 2020

1. **ЦЕЛИ РАБОТЫ:**
   1. научиться определять размер буфера переупорядочивания команд процессора.
2. **ХОД РАБОТЫ:**
   1. Для достижения цели написано две программы:
      * программа на Си, выполняющая операции и замеряющая время в тактах процессора;
      * программа на Python, генерирующая и компилирующая программу на Си, а также получающая и обрабатывающая результаты (отрисовка графики).

Программа на Си содержит в себе директивы условной компиляции для удобной сборки на разных процессорах.

Листинг программы на Си:

#include <stdio.h>

#include <unistd.h>

#include <stdlib.h>

#ifdef \_\_x86\_64\_\_

#include <x86intrin.h>

#endif

#ifndef SLEEP\_TIME

#define SLEEP\_TIME 1

#endif

#ifndef CYCLE\_NUM

#ifdef \_\_arm\_\_

#define CYCLE\_NUM 10000 // а иначе долго ждать, компьютер слабенький

#else

#define CYCLE\_NUM 10000000

#endif

#endif

#ifndef ARRAY\_SIZE

#define ARRAY\_SIZE (12 \* 1024 \* 1024 \* 10) // 10 x LLC

#endif

#ifdef \_\_arm\_\_

static inline unsigned long long \_\_ccnt**()** **{**

unsigned long long cycles **=** 0**;**

\_\_asm\_\_ volatile **(**"mrc p15, 0, %0, c9, c13, 0"**:**"=r" **(**cycles**));**

**return** cycles**;**

**}**

#endif

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

// prepare

unsigned long long start **=** 0**;**

unsigned long long end = 0;

int\* array = malloc(ARRAY\_SIZE \* sizeof(int));

for(size\_t i = 0; i < ARRAY\_SIZE; i++) {

array[i] = rand() % ARRAY\_SIZE;

}

// test

int k = 0;

sleep(SLEEP\_TIME);

#ifndef \_\_arm\_\_

start = \_\_rdtsc(); // x86

#else

start = \_\_ccnt(); // arm

#endif

for(size\_t i = 0; i < CYCLE\_NUM; i++) {

k = array[k];

#include "nops" // file with asm("nop");

}

#ifndef \_\_arm\_\_

end = \_\_rdtsc(); // x86

#else

end = \_\_ccnt(); // arm

#endif

// save results

printf("%llu", (end - start) / CYCLE\_NUM);

return 0;

}

Листинг программы на Python:

**import** subprocess

**from** subprocess **import** Popen**,** PIPE

**import** signal

**import** matplotlib**.**pyplot **as** plt

**import** platform

nops\_amount **=** 1

max\_nops\_amount **=** 500

results **=** **[]**

stop **=** **False**

**def** stop\_benchmark**(**signal**,** frame**):**

**global** stop

**print(**"Benchmark is stopped!"**)**

stop **=** **True**

signal**.**signal**(**signal**.**SIGINT**,** stop\_benchmark**)**

log **=** **open(**"compute.log"**,** "w"**)**

**while** **not** stop **and** nops\_amount **<=** max\_nops\_amount**:**

nops **=** **open(**"nops"**,** "w"**)**

**for** i **in** **range(**nops\_amount**):**

nops**.**write**(**"asm(\"nop\");\n"**)**

nops**.**close**()**

subprocess**.**call**([**"gcc"**,** "main.c"**,** "-O0"**])**

**with** Popen**([**"./a.out"**],** stdout**=**PIPE**)** **as** test**:**

**try:**

cycles **=** **int(**test**.**stdout**.**read**().**decode**(**"ascii"**))**

**except:**

**continue**

**print(**"nops command amount: " **+** **str(**nops\_amount**)** **+** ", clocks: " **+** **str(**cycles**))**

log.write(str(nops\_amount) + " " + str(cycles) + "\n")

results.append(cycles)

nops\_amount = nops\_amount + 1

log.close()

plt.plot(range(nops\_amount)[1:], results)

plt.xlabel("nops")

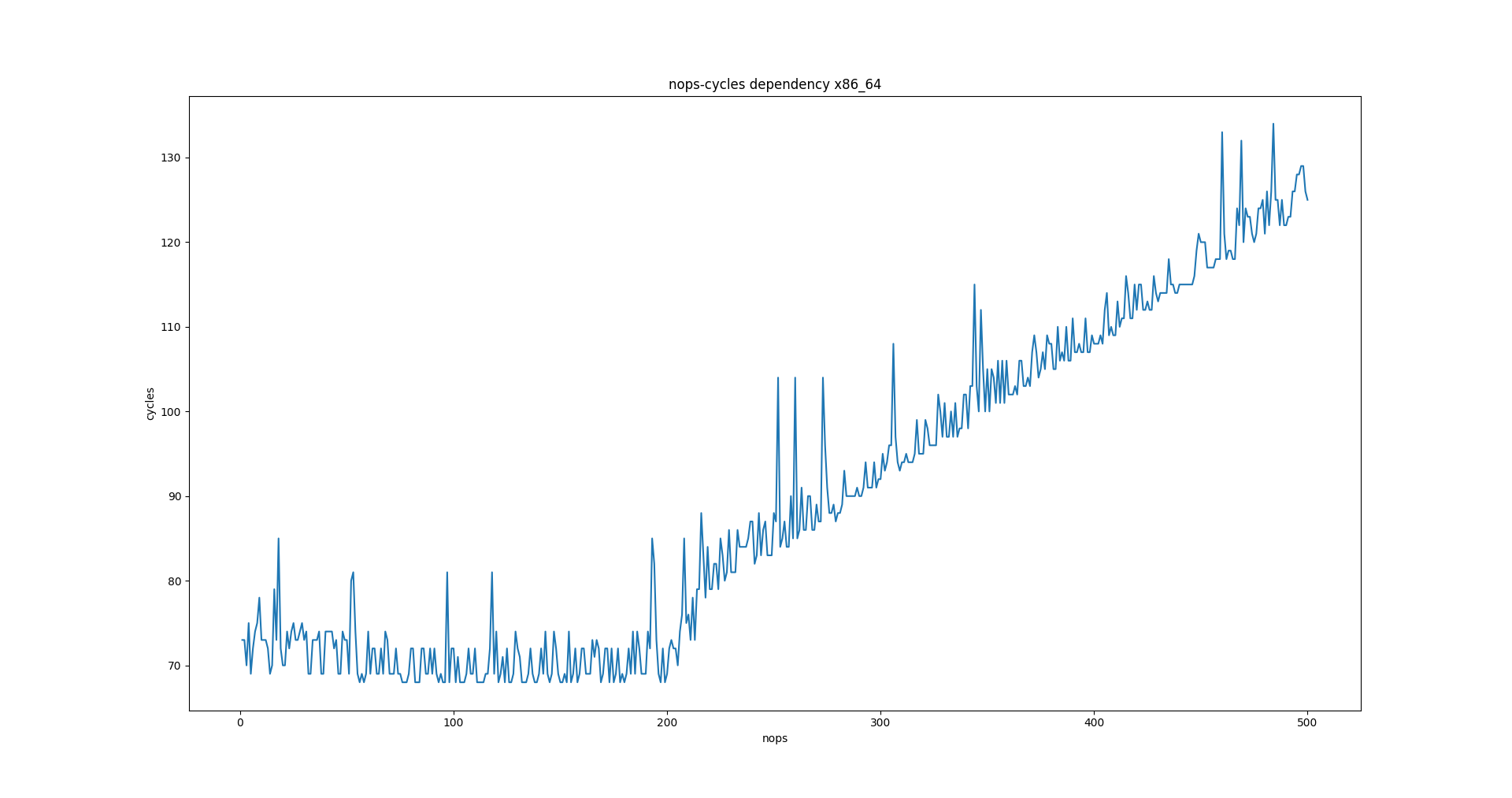
plt.ylabel("cycles")

plt.title("nops-cycles dependency " + platform.machine())

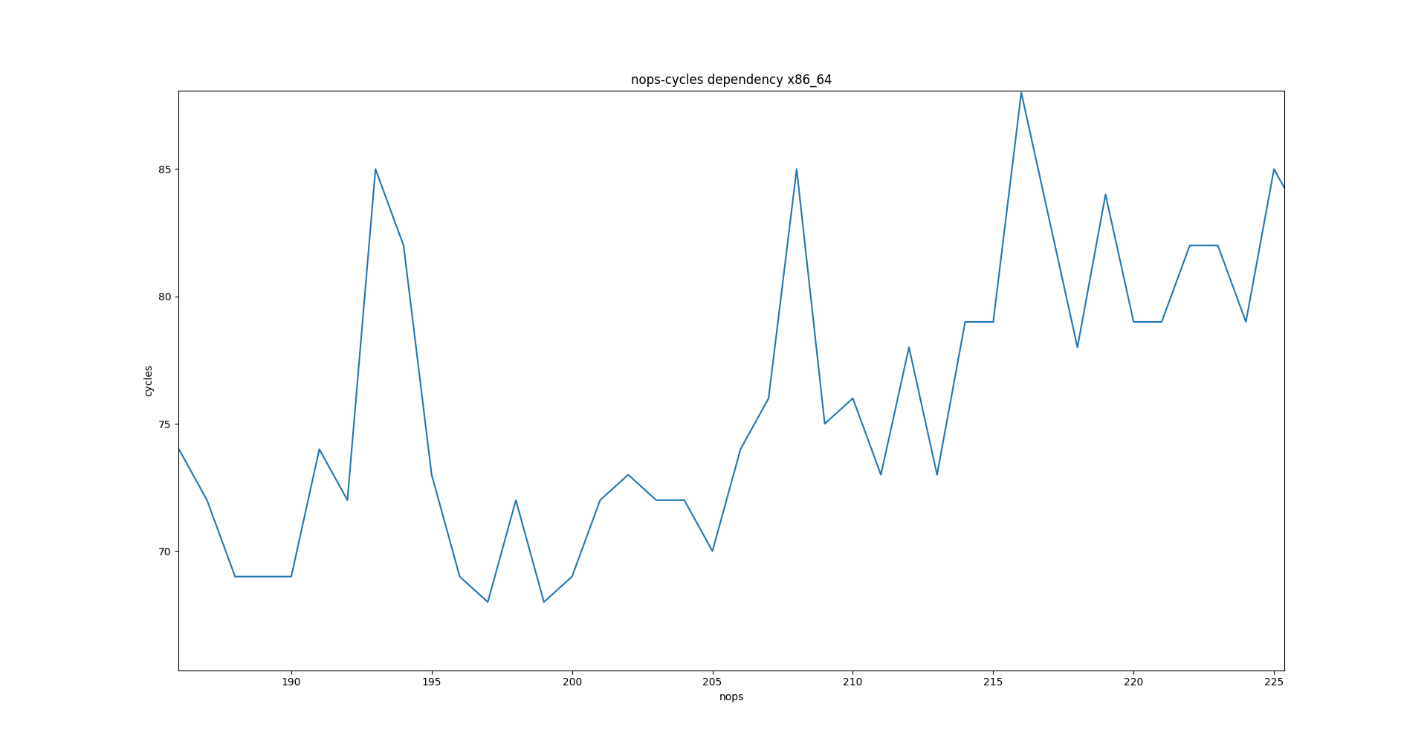
plt.show()

* 1. Описание тестируемых архитектур:
     1. **Intel Coffee Lake,** реализующий набор инструкций **x86\_64**:
        1. теоретический размер буфера переупорядочивания - **224 инструкции**;
        2. в качестве представителя этой микроархитектуры был взят **Intel Core i7-9700F (12M кэш, до 4.70 GHz).**
     2. **ARM Cortex-A72** в качестве реализации набора инструкций **ARMv8-A**:
        1. теоретический размер буфера переупорядочивания – **128 инструкций**;
        2. в качестве представителя этой микроархитектуры был взят **Broadcom BCM2711 (кэша 3-го уровня нет, до 1.5 GHz).**
  2. Результаты:
     1. **Intel Coffee Lake:**

График зависимости количества тактов от количества «нопов»:



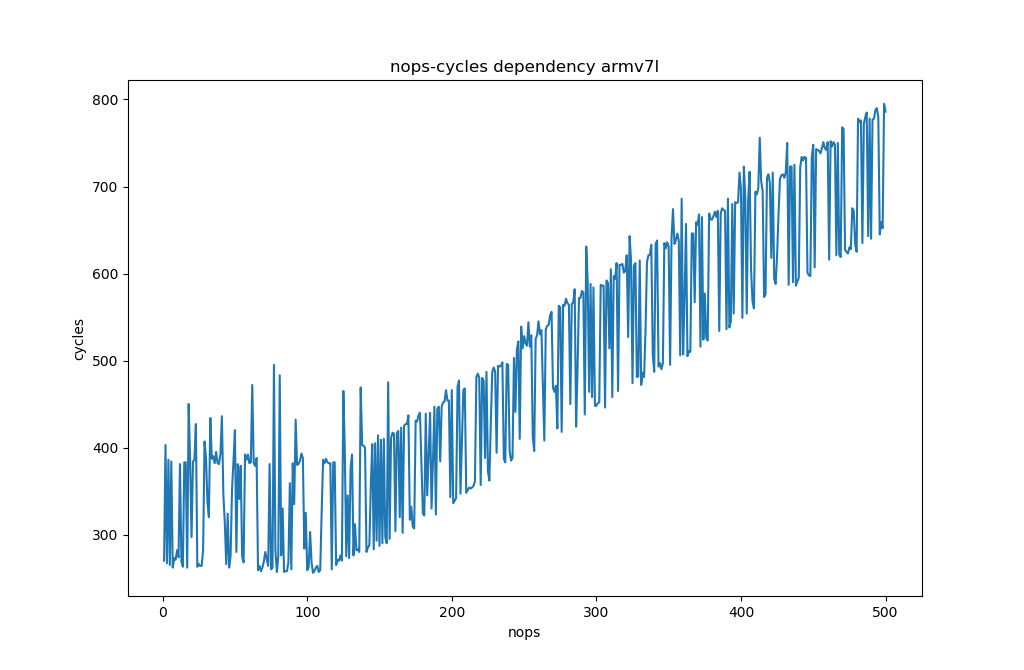
Характерная точка на этом графике (место начала роста):



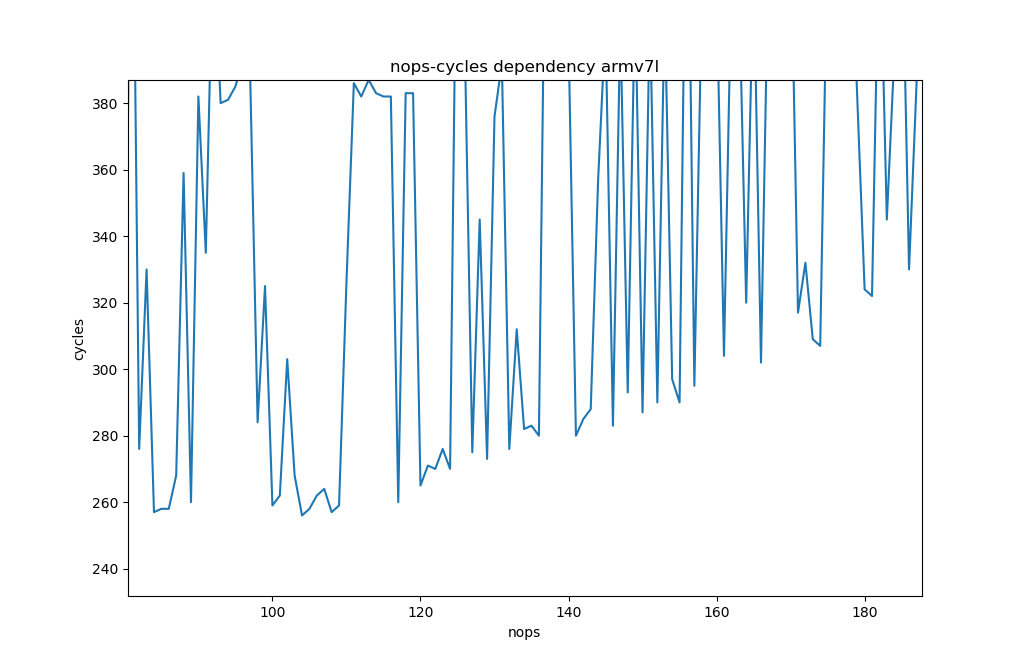
Постоянный рост начинается примерно в диапазоне 205 – 210 «нопов», что немного ниже 224, но довольно близко. Возможно, можно списать на погрешность измерений.

* + 1. **ARM Cortex-A72:**

График зависимости количества тактов от количества «нопов»:



Характерная точка на этом графике (место начала роста):



Постоянный рост начинается в области 120-140 «нопов», что очень близко к теоретическому значению 128, можно сказать, что совпадает.

1. **ВЫВОДЫ:**
   1. научились определять размер буфера переупорядочивания команд процессора;
   2. теоретические и расчетные показатели совпадают на обеих архитектурах;
   3. энергоэффективные мобильные ARM-процессоры при цене на порядок ниже проигрывают в этом показателе своим старшим x86-собратьям **всего** в 2 раза.
2. **ПРИЛОЖЕНИЕ:**
   1. Графики в исходном разрешении:







* 1. Логи вычислений, по которым можно отстроить эти графики:



* 1. Скриптдля отстройки графиков по логам (передать нужный файл в качестве аргумента):

**import** matplotlib.pyplot **as** plt  
**import** sys  
**import** platform  
  
**if** len(sys.argv) < 2:  
 **raise** Exception(**"usage: python3 build\_plot.py compute.log"**)  
  
log = open(sys.argv[1], **"r"**)  
lines = log.readlines()  
  
x\_axis = []  
y\_axis = []  
  
**for** line **in** lines:  
 data = line.split()  
 x\_axis.append(int(data[0]))  
 y\_axis.append(int(data[1]))  
  
plt.plot(x\_axis, y\_axis)  
plt.xlabel(**"nops"**)  
plt.ylabel(**"cycles"**)  
plt.title(**"nops-cycles dependency "** + platform.machine())  
plt.show()