

Дисциплина:
Операционные системы
Лабораторная работа № 1

Выполнил: Тищенко Б.В.
Группа: Р33112

Задание

Разработать программу на языке C, которая осуществляет следующие действия

- Создает область памяти размером **33** мегабайт, начинающихся с адреса **0xF6A3975** (если возможно) при помощи **mmap** заполненную случайными числами **/dev/urandom** в **66** потоков. Используя системные средства мониторинга определите адрес начала в адресном пространстве процесса и характеристики выделенных участков памяти. Замеры виртуальной/физической памяти необходимо снять:
 1. До аллокации
 2. После аллокации
 3. После заполнения участка данными
 4. После деаллокации
- Записывает область памяти в файлы одинакового размера **190** мегабайт с использованием **блочного** обращения к диску. Размер блока ввода-вывода **114** байт. Преподаватель выдает в качестве задания последовательность записи/чтения блоков **случайный**
- Генерацию данных и запись осуществлять в бесконечном цикле.
- В отдельных **107** потоках осуществлять чтение данных из файлов и подсчитывать агрегированные характеристики данных - **максимальное значение**.
- Чтение и запись данных в/из файла должна быть защищена примитивами синхронизации **cv**.
- По заданию преподавателя изменить приоритеты потоков и описать изменения в характеристиках программы.

Измерить значения затраченного процессорного времени на выполнение программы и на операции ввода-вывода используя системные утилиты.

Отследить трассу системных вызовов.

Построить графики системных характеристик.

Выполнение

Код программы

```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
#include <sys/mman.h>
#include <time.h>
#include <stdbool.h>
#include <limits.h>

// Allocation data
void *MMAP_ADDRESS = (void *)0xF6A3975;
const int MMAP_SIZE = 33 * 1024 * 1024;
const int NUM_MMAP_THREADS = 66;

// File data
const int NUM_FILLFILE_THREADS = 1;
const int OUTPUT_FILE_SIZE = 190 * 1024 * 1024;
const int IO_BLOCK_SIZE = 114;
const int NUM_READFILE_THREADS = 107;
const char *NAME_OUTPUT_FILE = "Output.txt";
const char *NAME_MAX_FILE = "Output_Max.txt";
volatile int max_int = INT_MIN;

// Synchronisation variables
bool should_work = true;
pthread_mutex_t MUTEX_OUTPUT_FILE;
pthread_cond_t CV_OUTPUT_FILE;

// Function declarations
void spawn_threads(pthread_t *threads, int amount, void *(*function)(void *));
void join_threads(int amount, pthread_t *ptr);
void *fill_memory();
void *fill_file();
void *read_file();

int main()
{
```

```

// Setup
pthread_mutex_init(&MUTEX_OUTPUT_FILE, NULL);
pthread_cond_init(&CV_OUTPUT_FILE, NULL);
int *mmap_status = mmap(MMAP_ADDRESS, MMAP_SIZE, PROT_READ |
PROT_WRITE, MAP_ANONYMOUS | MAP_PRIVATE, -1, 0);

// Fill memory
pthread_t threads_mem[NUM_MMAP_THREADS];
spawn_threads(threads_mem, NUM_MMAP_THREADS, fill_memory);
pthread_t threads_filew[NUM_FILLFILE_THREADS];
spawn_threads(threads_filew, NUM_FILLFILE_THREADS, fill_file);
pthread_t threads_filer[NUM_READFILE_THREADS];
spawn_threads(threads_filer, NUM_READFILE_THREADS, read_file);

join_threads(NUM_MMAP_THREADS, threads_mem);
join_threads(NUM_FILLFILE_THREADS, threads_filew);
join_threads(NUM_READFILE_THREADS, threads_filer);

int munmap_status = munmap(mmap_status, MMAP_SIZE);

return 0;
}

void spawn_threads(pthread_t *threads, int amount, void (*function)(void
*))
{
    for (int i = 0; i < amount; i++)
        pthread_create(&threads[i], NULL, function, NULL);
}

void join_threads(int amount, pthread_t *ptr)
{
    for (int i = 0; i < amount; i++)
        pthread_join(*(ptr + i), NULL);
}

void *fill_memory()
{
    while (should_work)
    {
        FILE *urandom = fopen("/dev/urandom", "r");
        fread(MMAP_ADDRESS, MMAP_SIZE, 1, urandom);
        fclose(urandom);
    }
}

```

```

    }
    pthread_exit(NULL);
}

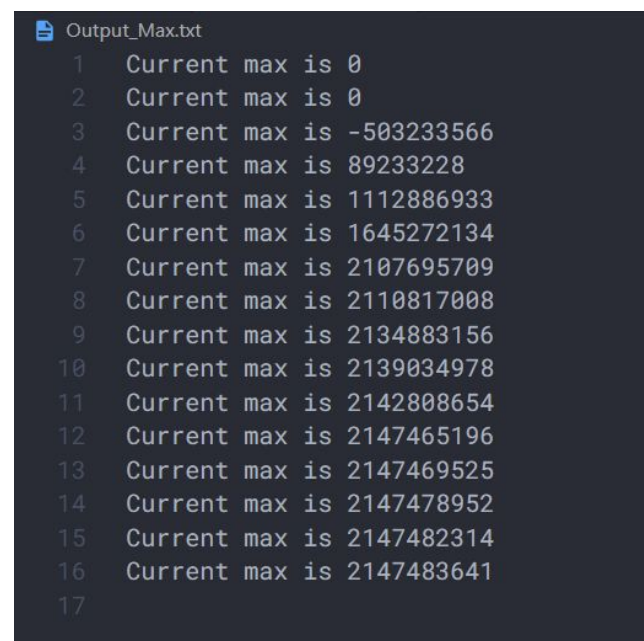
void *fill_file()
{
    while (should_work)
    {
        pthread_mutex_lock(&MUTEX_OUTPUT_FILE);
        FILE *output = fopen(NAME_OUTPUT_FILE, "w");
        srand(time(NULL));
        int num_io = OUTPUT_FILE_SIZE / IO_BLOCK_SIZE;
        int size_offset = MMAP_SIZE / IO_BLOCK_SIZE;
        for (int i = 0; i < num_io; i++)
        {
            int address_offset = random() % size_offset;
            void *address = (void *)((int *)MMAP_ADDRESS + address_offset);
            fwrite(address, IO_BLOCK_SIZE, 1, output);
        }
        fclose(output);
        pthread_cond_broadcast(&CV_OUTPUT_FILE);
        pthread_mutex_unlock(&MUTEX_OUTPUT_FILE);
    }
    pthread_exit(NULL);
    return NULL;
}

void *read_file()
{
    while (should_work)
    {
        pthread_mutex_lock(&MUTEX_OUTPUT_FILE);
        pthread_cond_wait(&CV_OUTPUT_FILE, &MUTEX_OUTPUT_FILE);
        FILE *file = fopen(NAME_OUTPUT_FILE, "r");
        int number = INT_MIN;
        while (!feof(file))
        {
            fread(&number, sizeof(int), 1, file);
            if (number > max_int)
            {
                FILE *max_file = fopen(NAME_MAX_FILE, "a");
                fprintf(max_file, "Current max is %d\n", number);
            }
        }
    }
}

```

```
        fclose(max_file);  
        max_int = number;  
    }  
}  
fclose(file);  
pthread_mutex_unlock(&MUTEX_OUTPUT_FILE);  
}  
return NULL;  
}
```

Вывод максимума



```
Output_Max.txt  
1 Current max is 0  
2 Current max is 0  
3 Current max is -503233566  
4 Current max is 89233228  
5 Current max is 1112886933  
6 Current max is 1645272134  
7 Current max is 2107695709  
8 Current max is 2110817008  
9 Current max is 2134883156  
10 Current max is 2139034978  
11 Current max is 2142808654  
12 Current max is 2147465196  
13 Current max is 2147469525  
14 Current max is 2147478952  
15 Current max is 2147482314  
16 Current max is 2147483641  
17
```

Замеры

Адрес начала процесса

0000556ce5b71000

Характеристики выделенной памяти

```
bodyak@DESKTOP-FU0PD3P:~/OS_Trace$ pmap -x 6384
6384:  ./Main
Address      Kbytes    RSS    Dirty Mode  Mapping
0000556ce5b71000      4        4      0 r---- Main
0000556ce5b72000      4        4      0 r-x-- Main
0000556ce5b73000      4        4      0 r---- Main
0000556ce5b74000      4        4      4 r---- Main
0000556ce5b75000      4        4      4 rw--- Main
0000556ce79c9000    132        4      4 rw--- [ anon ]
00007fdb30aa1000     12         8      8 rw--- [ anon ]
00007fdb30aa4000    148       148      0 r---- libc-2.31.so
00007fdb30ac9000   1504       732      0 r-x-- libc-2.31.so
00007fdb30c41000    296       128      0 r---- libc-2.31.so
00007fdb30c8b000      4          0      0 ----- libc-2.31.so
00007fdb30c8c000     12         12     12 r---- libc-2.31.so
00007fdb30c8f000     12         12     12 rw--- libc-2.31.so
00007fdb30c92000     16         16     16 rw--- [ anon ]
00007fdb30c96000     28         28      0 r---- libpthread-2.31.so
00007fdb30c9d000     68         68      0 r-x-- libpthread-2.31.so
00007fdb30cae000     20          0      0 r---- libpthread-2.31.so
00007fdb30cb3000      4          4      4 r---- libpthread-2.31.so
00007fdb30cb4000      4          4      4 rw--- libpthread-2.31.so
00007fdb30cb5000     24         12     12 rw--- [ anon ]
00007fdb30cc3000      4          4      0 r---- ld-2.31.so
00007fdb30cc4000    140       140      0 r-x-- ld-2.31.so
00007fdb30ce7000     32         32      0 r---- ld-2.31.so
00007fdb30cf0000      4          4      4 r---- ld-2.31.so
00007fdb30cf1000      4          4      4 rw--- ld-2.31.so
00007fdb30cf2000      4          4      4 rw--- [ anon ]
00007ffe2c205000    136         16     16 rw--- [ stack ]
00007ffe2c3d9000     12          0      0 r---- [ anon ]
00007ffe2c3dc000      8          4      0 r-x-- [ anon ]

-----
total kB          2648    1404    108
```

```
bodyak@DESKTOP-FU0PD3P:~/OS_Trace$ ps -eo pid,vsz,rss,comm | grep Alt
4425  2648   752 Alt
bodyak@DESKTOP-FU0PD3P:~/OS_Trace$ ps -eo pid,vsz,rss,comm | grep Alt
4425 36440   752 Alt
bodyak@DESKTOP-FU0PD3P:~/OS_Trace$ ps -eo pid,vsz,rss,comm | grep Alt
4425 3494292 34704 Alt
bodyak@DESKTOP-FU0PD3P:~/OS_Trace$ ps -eo pid,vsz,rss,comm | grep Alt
4425 2067232 2596 Alt
```

До аллокации, после аллокации, после заполнения данными, после деаллокации

Процессорное время

Исполнение программы

```
real    0m33.624s
user    0m1.420s
sys     1m34.894s
```

Ввод-вывод

```
bodyak@DESKTOP-FUOPD3P:~/OS_Trace$ iostat
Linux 4.19.128-microsoft-standard (DESKTOP-FUOPD3P)      12/03/20      _x86_64_      (4 CPU)

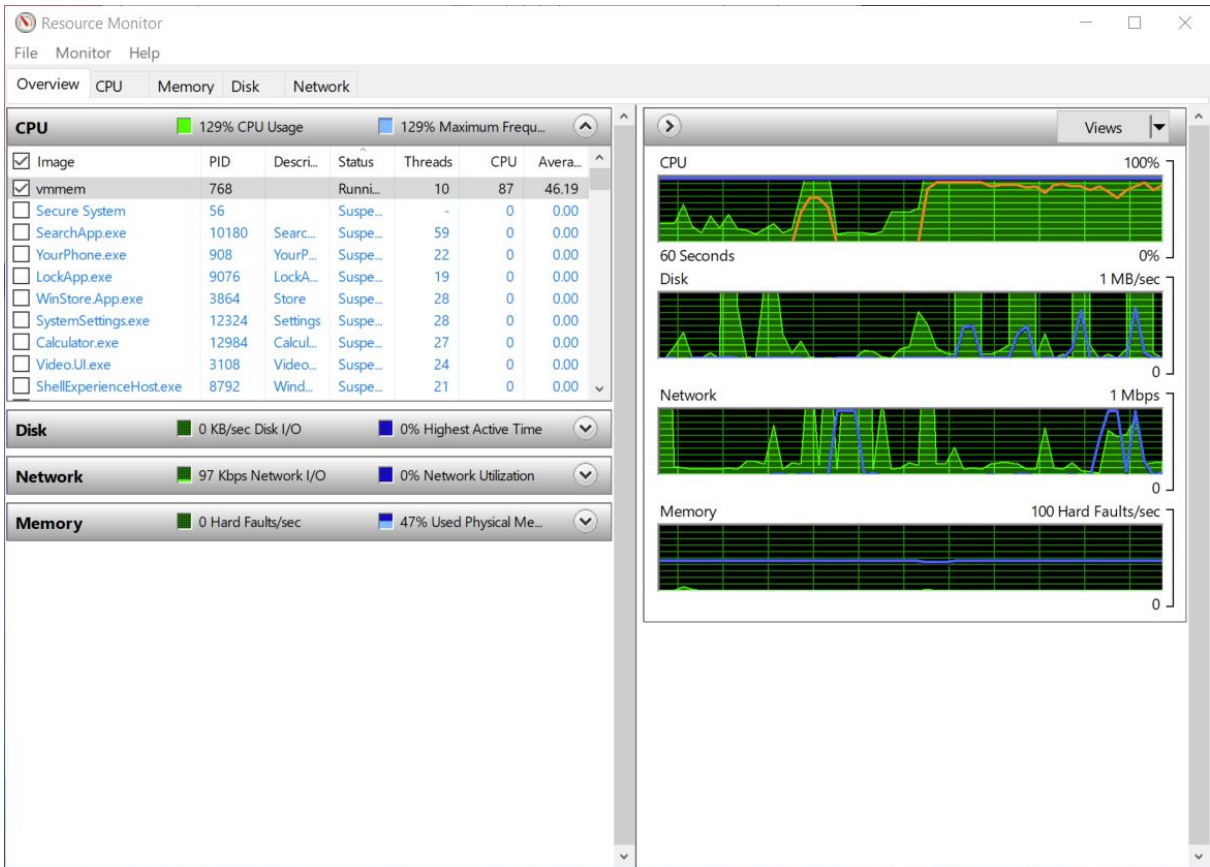
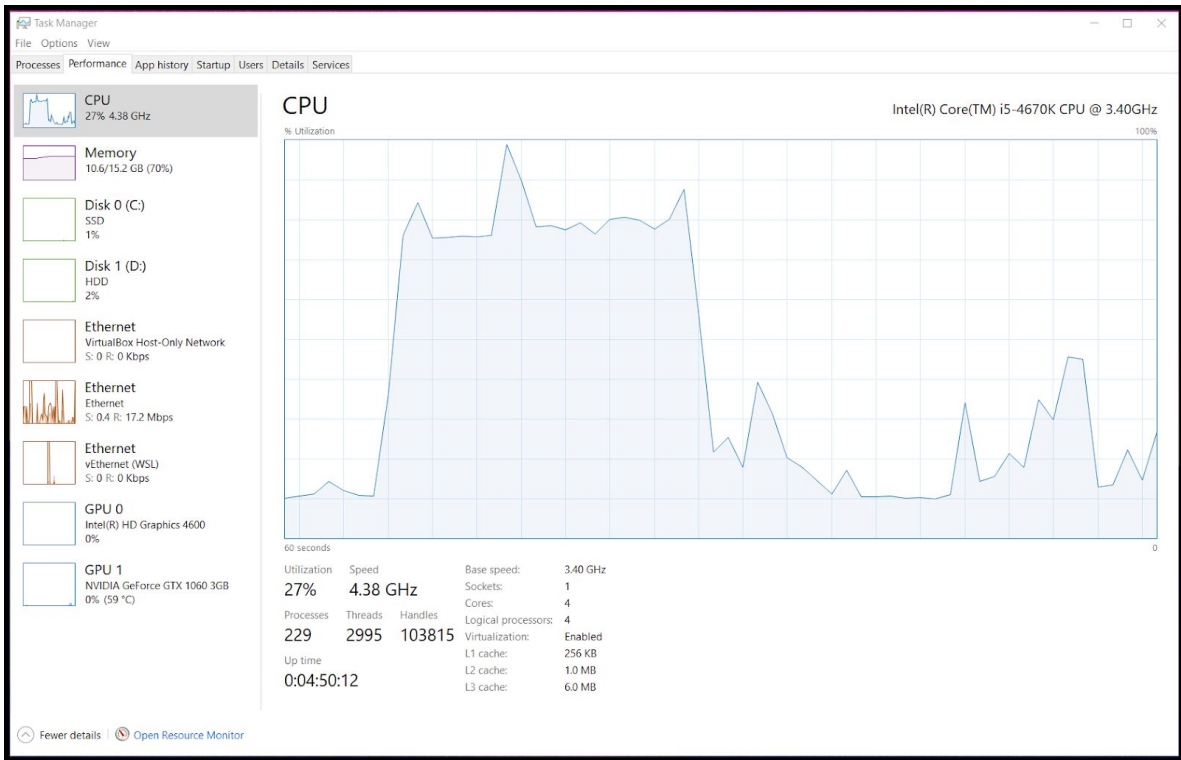
avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.36    0.00    0.59    0.01    0.00   99.04

Device            tps    kB_read/s    kB_wrtn/s    kB_dscd/s    kB_read    kB_wrtn    kB_dscd
sda                0.71         0.01       358.07         0.00        165    4195428         0
sdb                1.73        24.01       324.57       302.86     281333    3802916    3548620
```

Трасса системных вызовов

```
bodyak@DESKTOP-FUOPD3P:~/OS_Trace$ sudo strace -fp 7458
strace: Process 7458 attached
read(0, "\n", 1024)          = 1
openat(AT_FDCWD, "/dev/urandom", O_RDONLY) = 3
mmap(0xf6a3975, 34603008, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xf6a3000
write(1, "-> After allocation\n", 20) = 20
read(0, "\n", 1024)          = 1
mmap(NULL, 8392704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0) = 0x7fad7c108000
mprotect(0x7fad7c109000, 8388608, PROT_READ|PROT_WRITE) = 0
clone(child_stack=0x7fad7c907fb0, flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEAR_TID|CLONE_CHILD_SETTID|CLONE_NEWNS, parent_tidptr=0x7fad7c9089d0) = 7464
strace: Process 7464 attached
[pid 7464] set_robust_list(0x7fad7c9089e0, 24 <unfinished ...>
[pid 7458] mmap(NULL, 8392704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0 <unfinished ...>
[pid 7464] <... set_robust_list resumed> = 0
[pid 7458] <... mmap resumed> = 0x7fad7b907000
[pid 7464] futex(0x55b870e3a188, FUTEX_WAIT_PRIVATE, 0, NULL <unfinished ...>
[pid 7458] mprotect(0x7fad7b908000, 8388608, PROT_READ|PROT_WRITE) = 0
```


Графики системных характеристик



Вывод

В ходе выполнения этой лабораторной работы я как никогда ощутил важность понятной формулировки задания. Писать программу на Си было довольно сложно и пришлось дополнительно много чего почитать-посмотреть, но оно того точно стоило.