Министерство цифрового развития, связи и массовых

коммуникаций российской федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики»

(СибГУТИ)

Кафедра ПМиК

Расчетно-Графическое Задание

по дисциплине

«Операционные системы реального времени»

Выполнил: студенты 4 курса

ИВТ, гр. ИП-813

Пещеров В.А.

Проверила:

Белевцова Е.А.

Новосибирск 2021

Оглавление

[Задание 3](#_Toc88901500)

[Выполнение 3](#_Toc88901501)

[Результат работы программы 4](#_Toc88901502)

[Листинг 5](#_Toc88901503)

# Задание

1.Исследуйте эффективность по времени выделения буфера в стеке (с помощью alloca()) и в куче ( с помощью malloc() ).

2.Графический терминал VinGraph обеспечивает более качественную анимацию при использовании вне экранного контекста (переменная среды VGOSC = 1). Определите коэффицент снижения производительности в режиме VGOSC = 1.

# Выполнение

Часть 1.

Для выполнения первого задания, была реализована программа с использование malloc и alloc. Для времени была использованал функция ClockCycles.

- alloca(size\_t size) – функция выделяет size байт памяти из стэка системы и возвращает указатель на него. Также в alloca() память освобождается автоматически при выходе из функции, которая вызвала alloca() .

- malloc(size\_t size) – функция возвращает адрес первого байта области памяти размера size байт, которая была выделена из кучи.

Замеры были сделаны на выделение размера равным 100 байт. Alloca() выдаёт большое преимущество по количеству произведённых циклов, а именно в среднем в 2 раза. По времени различие не так заметно, но тоже присутствует в пользу alloca().

Часть 2.

Для выполнения второго задания, были реализованы программы с использованием графического интерфейса, а также использование двух режимов вне экранного контекста такие как VGOSC = 1 и VGOSC = 0

- Переменная среды VGOSC определяет режим работы терминала: с использованием буфера вне экранного контекста (VGOSC = 1) или без него (VGOSC = 0 или не определена). Использование вне экранного контекста позволяет убирать мерцание при передвижение графических элементов, однако успешность реализации зависит от графического адаптера и его драйвера. Этот режим также создаёт дополнительную нагрузку на центральный процессор.

Замеры были сделаны так, ставим терминал в режим VGOSC = 1, начинаем отрисовку объектов и с помощью функции ClockCycles() замеряем время. Затем тоже самое делает для режима VGOSC = 0, и вычисляем коэффициент снижения производительности.

Результаты показывают, что VGOSC = 0, эффективнее в отличии от VGOSC = 1, но качество самой анимации падает.

# Результат работы программы

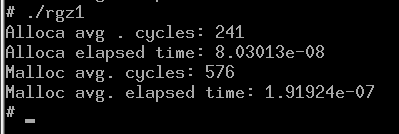


Рисунок 1. Задание 1

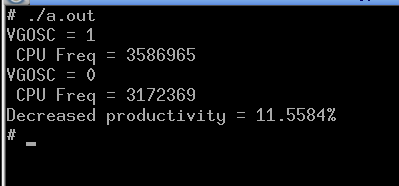


Рисунок 2. Задание 2

# Листинг

Программа 1:

#include <iostream>

#include <inttypes.h>

#include <malloc.h>

#include <sys/neutrino.h>

#include <sys/syspage.h>

using namespace std;

int main()

{

char \*allo,\*mallo;

uint64\_t t1,t2,cps;

int cl=100;

uint64\_t allotime=0;

for(int i=0;i<cl;i++){

t1 = ClockCycles();

allo=(char \*)alloca(100);

t2 = ClockCycles();

allotime+=t2-t1;

}

cout<<"Alloca avg . cycles: "<<allotime/cl <<endl;

cps=SYSPAGE\_ENTRY(qtime)->cycles\_per\_sec;

cout<<"Alloca elapsed time: "<<(double)(allotime/cl)/cps<<endl;

uint64\_t mallotime=0;

t2=0;t1=0;

for(int i=0;i<cl;i++){

t1=ClockCycles();

mallo=(char\*)malloc(100\*sizeof(char));

t2=ClockCycles();

mallotime+=t2-t1;

free(mallo);

}

cout<<"Malloc avg. cycles: "<<mallotime/cl<<endl;

cps=SYSPAGE\_ENTRY(qtime)->cycles\_per\_sec;

cout<<"Malloc avg. elapsed time: "<<(double)(mallotime/cl)/cps <<endl;

return 0;

}

Программа 2:

#include <stdlib.h>

#include <unistd.h>

#include <vingraph.h>

#include <iostream>

#include <stdio.h>

#include <time.h>

#include <sys/neutrino.h>

#include <sys/syspage.h>

#include <inttypes.h>

#include <malloc.h>

using namespace std;

int main(){

setlocale(LC\_ALL,"RUS");

srand(time(0));

char \*var1="VGOSC=1";

char \*var2="VGOSC=0";

char \*name="VGOSC";

char \*val1=NULL;

int rc;

int cl=1300;

rc=putenv(var1);

val1=getenv(name);

ConnectGraph();

int x=0;

int y=0;

uint64\_t t1,t2;

uint64\_t alltime1=0;

int i=0;

while(i < cl){

i++;

x=rand()%500;

y=rand()%500;

t1=ClockCycles();

Ellipse(x,y,60,60);

t2=ClockCycles();

alltime1+=t2-t1;

}

CloseGraph();

printf(" VGOSC = %s \n CPU Freq = %u\n",val1, alltime1/cl);

rc=putenv(var2);

val1=getenv(name);

ConnectGraph();

uint64\_t t3,t4;

uint64\_t alltime2=0,alltimeV,alltimeG;

long double alltimeQ;

i=0;

while(i < cl){

i++;

x=rand()%500;

y=rand()%500;

t3=ClockCycles();

Ellipse(x,y,60,60);

t4=ClockCycles();

alltime2+=t4-t3;

}

CloseGraph();

printf(" VGOSC = %s \n CPU Freq = %u\n",val1, alltime2/cl);

alltimeV=alltime1/cl;

alltimeG=alltime2/cl;

cout<<"Decreased productivity = ";

cout<<100 - (double)alltimeG/alltimeV \* 100<<"%"<<endl;

return 0;

}