Министерство цифрового развития, связи и
массовых коммуникаций Российской Федерации
«СибГУТИ»

Кафедра ПМиК

Лабораторная работа №5
по
«Архитектуре ЭВМ»

Выполнил:
студент группы ИП-114
Яворский Д. И.

Проверил:
Мамойленко С. Н.

Новосибирск
2023

Ссылка на репозиторий: https://git.csc.sibsutis.ru/ip114s27/myfirstproject

В ветке lab5 были разработаны следующие файлы:
1. main.c - тестируется основная функциональность программы;
2. myTerm.h - объявляются все функции и константы;
3. myTerm.c - описываются все функции;
4 .gitignore игнорируются объектные файлы. В файле Makefile собирается основной проект;
5 .gitlab-ci.yml хранится код для сборки CI;
6. ALU – арифметико-логическое устройство
7. CU – управляющие устройство
8. Signal – отправляет сигналы;
9. myReadKey.c - статическая библиотека, описываются все функции;
10. myReadKey.h - объявляются все функции и макроссы;
11. makefile - собирается основной проект.

Список функций:

```
#ifndef MY_ALU_H
#define MY_ALU_H

#include "mySimpleComputer.h"

int ALU (int command, int operand);

#endif
```

```c
#ifndef MY_CU_H
#define MY_CU_H
#include "myALU.h"
#include "myInterface.h"
#include "myReadKey.h"
#include "mySimpleComputer.h"
#include "myTerm.h"
#include "string.h"
int READ (int operand);
int WRITE (int operand);
int LOAD (int operand);
int STORE (int operand);
int JUMP (int operand);
int JNEG (int operand);
int JZ (int operand);
int HALT ();
int JNS (int operand);
int CU ();

#endif

#ifndef MYSIGNAL_H
#define MYSIGNAL_H

#include "myALU.h"
#include "myCU.h"
#include "myInterface.h"
#include "mySimpleComputer.h"
#include <signal.h>
void ms_setSignals ();
void ms_timerHandler (int sig);
void ms_userSignal (int sig);

#endif // MYSIGNAL_H
```

```c
#ifndef MYBIGCHARS_H
#define MYBIGCHARS_H

#include "myTerm.h"
#include <stdbool.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
/* Псевдографика */
#define ACS_CKBOARD 'a'  // Штриховка
#define ACS_ULCORNER 'l' // Левый верхний угол
#define ACS_URCORNER 'k' // Правый верхний угол
#define ACS_LRCORNER 'j' // Правый нижний угол
```

```
#define ACS_LLCORNER 'm' // Левый нижний угол
#define ACS_HLINE 'q'    // Горизонтальная линия
#define ACS_VLINE 'x'    // Вертикальная линия
extern unsigned int
big_chars[][2];
 int bc_printA (char charr); int bc_box (int x1, int y1, int x2, int y2);
int bc_printBigChar (unsigned int *big, int x, int y, enum colors
colorFG,                      enum colors colorBG); int bc_setBigCharPos
(int *big, int x, int y, int value); int bc_getbigCharPos (int *big, int
x, int y, int *value); int bc_bigCharWrite (int fd, int *big, int count);
int bc_bigCharRead (int fd, int *big, int need_count, int *count);
#endif



#ifndef INTERFACE_H
#define INTERFACE_H
#include "myBigChars.h"
#include "mySimpleComputer.h"
#include <ctype.h>
#include <string.h> #include
<unistd.h> extern int
instruction_counter; int
drawingBigChars (); int
ui_initial (); int ui_update
(); int ui_setValue (); int
drawingBoxes ();

int drawing_texts (); int ui_Counter (); int
drawing_memory (); int drawing_flags (); int
drawing_IC (); int ui_setValue (); bool
checkCorrectInput (const char buffer[10]); int
ui_messageOutput (char *str, enum colors color);
int clearBuffIn ();

#endif


#ifndef MYREADKEY_H
#define MYREADKEY_H
```

```c
#include <stdbool.h>
#include <stdio.h>
#include <termios.h>
#include <unistd.h>

enum keys
{
  ESC_KEY,
  L_KEY,
  S_KEY,
  R_KEY,
  T_KEY,
  I_KEY,
  F5_KEY,
  F6_KEY,
  UP_KEY,
  DOWN_KEY,
  LEFT_KEY,
  RIGHT_KEY,
  ENTER_KEY,
}; extern struct termios
save;
 int rk_readKey (enum keys *key); int rk_myTermSave (); int
rk_myTermRestore (); int rk_myTermRegime (int regime, int vtime, int vmin,
int echo, int sigit);
#endif



#ifndef SIMPLECOMPUTER_H
#define SIMPLECOMPUTER_H

#include <inttypes.h>
#include <math.h>
#include <stdio.h>
#include <unistd.h> // for write function
#define N 100

#define REGISTER_SIZE 5

#define OVERFLOW 0               // переполнение
#define DIVISION_ERROR_BY_ZERO 1 // ошибка деления на 0
#define OUT_OF_MEMORY 2          // выход за границы памяти
```

```
#define IGNORING_CLOCK_PULSES 3 // игнорирование тактовых импульсов
#define INVALID_COMMAND 4 // неверная команда
 extern short int
memory_arr[N]; extern char
flag;
extern short currMemCell;
 int sc_memoryInit (); int sc_memorySet (int address, int
value); int sc_memoryGet (int address, int *value); int
sc_memorySave (char *filename); int sc_memoryLoad (char
*filename); int sc_regInit (); int sc_regSet (int _register,
int value); int sc_regGet (int _register, int *value); int
sc_commandEncode (int command, int operand, int *value); int
sc_commandDecode (int value, int *command, int *operand);
#endif



#ifndef MYTERM_H
#define MYTERM_H
#include <stdio.h>
#include <sys/ioctl.h> #include
<unistd.h> // for write and dup2
 enum
colors
{
  RED = 196,
  PEACH = 203,
  GREEN = 10,
  SOFT_GREEN = 192,
  BLUE = 20,
  BLACK = 16,
  GRAY = 240,
  WHITE = 15,
  DEFAULT = 0
};
int mt_clrscr (); int mt_gotoXY (int _x, int
_y); int mt_getscreensize (int *rows, int
*cols); int mt_setfgcolor (enum colors); int
mt_setbgcolor (enum colors); int
set_default_color ();

#endif
```