Лекция 2 Язык программирования С. Стандартная библиотека языка С

План курса «Встраиваемые микропроцессорные системы»:

Лекция 1: Введение. Язык программирования С

Лекция 2: Язык программирования С. Стандартная библиотека языка С

Лекция 3: Применение языка С для встраиваемых систем

Лекция 4: Микроконтроллер

Лекция 5: Этапы разработки встраиваемых систем

Лекция 6: Разработка и отладка программ для встраиваемых систем

Лекция 7: Архитектура программ для встраиваемых систем

Лекция 8: Периферийные модули: DMA, USB, Ethernet

Объявление и вызов функций

```
/* Объявление функций */
тип-возвр-значения имя-функции(аргументы)
{
    объявления;
    операторы;
}

/* Вызов функций */
переменная = имя-функции(аргументы);
```

Обобщенная форма

```
int add_two(int n)
{
    int tmp = 0;

    tmp = n + 2;
    return tmp;
}
a = add_two(2);
```

Пример

МЭИ

Объявление и вызов функций

```
/* Объявление функций */
/* Абсолютное значение числа */
int abs(int n)
{
    if (n < 0)
        return -n;
    else
        return n;
}

/* Поиск максимального элемента в массиве */
int arr_max(int a[], int n)
{
    int max = a[0];
    for (int i = 1; i < n; i++)
       if (a[i] > max)
              max = a[i];
    return max;
}
```

```
/* Включить светодиод */
void led_on()
{
    PORTC |= 0x01;
}

/* Вызов функций */
void main()
{
    int c;
    c = abs(-10); /* c = 10 */

    int arr[5] = {4, 3, 1, 7, 5};

    c = arr_max(arr, 5); /* c = 7 */
    led_on();
}
```

Объявление и вызов функций

```c
/* Объявление функций */
/* Обратить порядок элементов в массиве */
void reverse(char a[], int n)
{
    int i, j, tmp;
    for (i = 0, j = n - 1; i < j; i++, j--){
        tmp = a[i];
        a[i] = a[j];
        a[j] = tmp;
    }
    return; /* Необязательно */
}
```

```c
/* Вызов функций */
void main()
{
    char arr[5] = {4, 3, 1, 7, 5};
    char len = 5;

    reverse(arr, len);
    /* arr = {5, 7, 1, 3, 4} */
    int i;
    for (i = 0; i < len; i++){
        printf("%d ", arr[i]);
    }
}
```

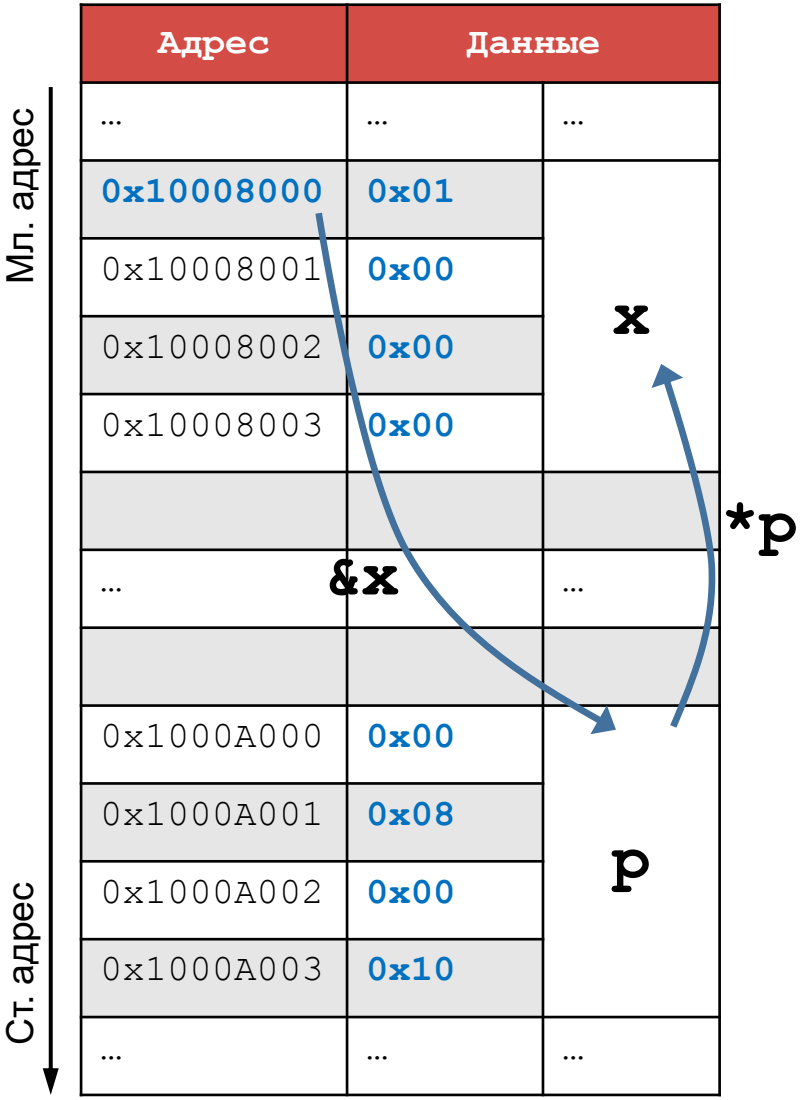
Указатели

```
/* Объявление переменных и массива */
int x = 1, y = 2, z[10];
/* Объявление указателя */
int *p;

p = &x;/* p - адрес переменной x
или p указывает на x */

/* Разыменование *p */
y = *p; /* *p = x = 1 -> y = 1 */
*p = 0; /* x = 0 */

p = &z[0];/*p - указывает на z[0]*/
*p = 100;/* z[0] = 100 */
```



Массивы и указатели

```
/* Объявление массива */
int a[10];
/* Объявление указателя */
int *pa;

pa = &a[0]; /* Указатель на a[0] */
pa = a; /* Эквивалентно pa = &a[0] */

/* a[i] эквивалентно *(pa + i) */
*pa = 0; /* a[0] = 0 */
*(pa + 2) = 2; /* a[2] = 2 */
```

Мл. адрес

Ст. адрес

Индекс	Указатель	Адрес	Данные
…	…		…
			Другие данные
a[0]	*pa	pa	1
a[1]	*(pa+1)	pa+4	4
a[2]	*(pa+2)	pa+8	13
…	…		…
a[9]	*(pa+9)	pa+36	99
			Другие данные
…	…		…

Строки и символы

```
/* Объявление и инициализация
   строки */
char msg[] = "Hello\r\n";
/* Объявление и инициализация
   символа */
char ch = 'a';
```

ASCII – название таблицы (кодировки), в которой некоторым распространённым печатным и непечатным символам сопоставлены числовые коды.
Таблица ASCII содержит коды для символов: десятичных цифр, латинского алфавита, национального алфавита, знаков препинания, управляющих символов.
Размер символа ASCII – 8 бит (1 байт).

Мл. адрес

Ст. адрес

Индекс	Адрес	Данные (HEX)	Данные (ASCII)
…	…		…
		Другие данные	
msg[0]	msg	0x48	H
msg[1]	msg+1	0x65	e
msg[2]	msg+2	0x6C	l
msg[3]	msg+3	0x6C	l
msg[4]	msg+4	0x6F	o
msg[5]	msg+5	0x0D	\r
msg[6]	msg+6	0x0A	\n
msg[7]	msg+7	0x00	\0
		Другие данные	
…	…	…	…

Структуры

```
/* Объявление структуры */
struct point {
    int x;
    int y;
};

/* Объявление переменной pt1 типа point */
struct point pt1;

/* Инициализация полей структуры */
pt1.x = 22;
pt1.y = 7;

/* Инициализация структуры */
struct point pt2 = {-10, 0};
```

Объединения

```
/* Объявление объединения */
union {
    int word;
    short hword[2];
    char byte[4];
} u;

u.word = 0x12345678;

short lo_hword = u.hword[0];/* lo_hword = 0x5678 */
char hi_byte = u.byte[3];/* hi_byte = 0x12 */
```

Адрес	0	1	2	3
Память	0x78	0x56	0x34	0x12
word	0x12345678			
hword	0x5678		0x1234	
byte	0x78	0x56	0x34	0x12

Константы

```
/* Макроопределения */
#define MAXLEN 100
char msg[MAXLEN + 1];

#define TRUE 1
#define FALSE 0

if (res == TRUE)
    ...

/* Перечисления */
enum month {JAN = 1, FEB, MAR, APR,
MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC};
/* FEB = 2, MAR = 3, ...*/
enum month m = DEC;

enum boolean {FALSE = 0, TRUE};/* TRUE = 1 */
```

Битовые операции

```c
/* Битовые поля */
struct {
    unsigned int enable : 1;
    unsigned int test: 1;
    unsigned int err_code: 3;
} state_flags;

state_flags.enable = 1;
state_flags.test = 0;
state_flags.err_code = 4;
```

```c
/* Пример без битовых полей */
#define ENABLE_FLAG 1
#define TEST_FLAG 2
#define ERR_MASK 7
#define ERR_SHIFT 2
unsigned int flags;

flags |= ENABLE_FLAG;/* Установить бит 0*/
flags &= ~TEST_FLAG;/* Сбросить бит 1 */
/* Сбросить код ошибки */
flags &= ~(ERR_MASK << ERR_SHIFT);
/* Установить новый код ошибки */
flags |= (4 & ERR_MASK) << ERR_SHIFT;
```

Препроцессор

```
/* Макроопределения */
#define ARR_SIZE 10
#define TRUE 1
#define FALSE 0
#define STEP 100

/* Подключение файлов */
#include "adc.h"
#include <stdio.h>


#define DEBUG

/* Условная компиляция */
#ifdef DEBUG
...
#endif
```

```
/* Указания компилятору */
#pragma once
```

Получение аргументов командной строки

```c
/* Для программ на ПК */
#include <stdio.h>
int main(int argc, char* argv[])
{
    /* argc - количество аргументов */
    /* argv - массив с указателями на аргументы */
    /* argv[0] - имя исполняемого файла */
    int i;
    for (i = 0; i < argc; i++)
        printf("argv[%d]: %s\n", i, argv[i]);

    return 0;
}
```

```
PS .\test.exe arg1 arg2
argv[0]: C:\temp\test.exe
argv[1]: arg1
argv[2]: arg2
```

Стандартная библиотека C

Заголовочный файл	Назначение	Пример
`#include <ctype.h>`	Символьный тип	`int tst = isupper(ch);`
`#include <math.h>`	Математические операции	`float a = exp(1.0);`
`#include <stdio.h>`	Стандартный поток ввода/вывода, файлы, форматированный ввод/вывод	`printf("Hello, world!\r\n");`
`#include <stdlib.h>`	Преобразование типов, псевдослучайные числа, управление программой, выделение памяти, сортировка, поиск	`int n = atoi("42");`
`#include <string.h>`	Строковые операции и операции с памятью	`int eq = strcmp(s, "run");`
`#include <stdint.h>`	Целочисленные типы	`uint8_t a = 0;`
`#include <stdbool.h>`	Булев тип	`bool flag = false;`
`#include <assert.h>`	Диагностика и отладка	`assert(a > 10);`
`#include <time.h>`	Время	`time_t posixtime = time(NULL);`

Существуют и другие заголовочные файлы.
Реализация стандартной библиотеки распространяется вместе с компилятором.

ctype.h: символьный тип

```
#include <ctype.h>

int tst;
char ch = 'D';

tst = isalpha(ch);/* Отображаемый символ, tst = 1 */
tst = isdigit(ch);/* Цифра 0 - 9, tst = 0 */
tst = isupper(ch);/* Верхний регистр, tst = 1 */
tst = islower(ch);/* Нижний регистр, tst = 0 */

ch = tolower(ch);/* Перевести в нижний регистр, ch = 'd' */
ch = toupper(ch);/* Перевести в верхний регистр, ch = 'D' */
```

math.h: математические операции

```c
#include <math.h>

/* Существует вариант библиотеки с суффиксом f(mathf) для работы с
числами float.

Функции из этой библиотеки также имеют суффикс f, например, sinf */

double a;

a = sin(1.5);   /* Вычисление синуса */
a = asin(1.0);  /* Вычисление арксинуса */
a = ceil(0.99); /* Округление до ближайшего большего целого числа */
a = floor(0.99);/* Округление до ближайшего меньшего целого числа */
a = exp(1.0);   /* Вычисление экспоненты */
a = log(10.0);  /* Вычисление натурального логарифма */
```

stdlib.h: преобразование типов, выделение памяти

```c
#include <stdlib.h>

int a = abs(-10);    /* c = 10 */
int b = atoi("100");/* d = 100 */

srand(42);        /* Инициализация генератора псевдослучайных чисел */
int n = rand(); /* Генератор псевдослучайных чисел */

int *buf = malloc(100); /* Выделение памяти из кучи */
free(buf);              /* Освобождение памяти в кучу */
```

stdio.h: форматированный вывод

```c
#include <stdio.h>

printf("Hello, world!\n");

int i = 2, j = 3;
/* Вывод целых чисел со знаком */
printf("i=%d j=%d\n", i, j); /* "i=2 j=3" */

i = 15;
/* Вывод целых чисел в шестнадцатеричном
представлении */
printf("i=%x j=%x\n", i, j); /* "i=f j=3" */

/* Вывод чисел с плавающей запятой */
float ch1 = 10.1, ch2 = 12.3;
printf("ch1=%f, ch2=%.1f\n", ch1, ch2);/* "ch1=10.100000, ch2=12.3" */

/* Вывод строки */
char *msg = "overcurrent";
printf("Fail: %s\n", msg); /* "Fail: overcurrent" */
```

stdio.h: форматированный ввод

```c
#include <stdio.h>
unsigned int a;
scanf("%u", &a); /* Ввод целого числа без знака*/

int b, c;
scanf("%d %d", &b, &c); /* Ввод двух целых чисел со знаком разделенных пробелом */
scanf("%d, %d", &b, &c); /* Ввод двух целых чисел со знаком разделенных запятой */

float x;
scanf("%f", &x); /* Ввода числа с плавающей запятой */

char buf[20 + 1];
scanf("%s", buf); /* Ввод до первого символа пробела, табуляции или новой строки */
scanf("%20s", buf); /* Более безопасный вариант */
gets(buf); /* Ввод строки до символа новой строки */
fgets(buf, sizeof(buf), stdin); /* Более безопасный вариант */

int h, m, s;
/* Ожидание строки вида "Time: 12 h 00 m 00 s" */
if (scanf("Time: %d h %d m %d s", &h, &m, &s) == 3)
    ...
```

stdio.h: чтение из файла

```c
#include <stdio.h>

/* Объявление файлового указателя */
FILE *fp;
/* Открытие файла test.txt на чтение ("r").
Если файла не существует, то fopen возвратит NULL. */
if ((fp = fopen("test.txt", "r")) == NULL)
{
    // Обработка ошибки
}
/* Объявление буфера */
char buf[100];
/* Чтение файла по строчно */
while (fgets(buf, sizeof(buf), fp))
    printf("%s\n", buf); /* Печать строки */
/* Закрытие файла */
fclose(fp);
```

Программа scat

Команда `cat` последовательно считывает содержимое файлов, указанных в параметре `file`, и записывает его в стандартный поток вывода. Ключ `–n` выводит номера строк.

```
/* Программа scat */
#include <string.h>
#include <stdio.h>
int main(int argc, char* argv[])
{
    /* Проверка наличия параметра -n */
    int n = 0;
    for (int i = 1; i < argc; i++)
    {
        if (strcmp(argv[i], "-n") == 0)
        {
            n = 1;
        }
    }
    /* Вывод содержимого файлов */
    FILE *fp;
    for (int i = 1; i < argc; i++)
    {
        if (strcmp(argv[i], "-n") == 0)
            continue;
        if (argv[i][0] == '-')
        {
            printf("scat: %s: Invalid option\n",
                argv[i]);
            break;
        }
        if ((fp = fopen(argv[i], "r")) == NULL)
        {
            printf("scat: %s: No such file or
                directory\n", argv[i]);
            continue;
        }
        char buf[100];
        int j = 1;
        while (fgets(buf, sizeof(buf), fp))
        {
            if (n)
                printf("%d %s", j, buf);
            else
                printf("%s", buf);
            j++;
        }
        fclose(fp);
    }
    return 0;
}
```

МЭИ

stdio.h: запись в файл

```c
#include <stdio.h>

/* Объявление файлового указателя */
FILE *fp;
/* Открытие файла test.txt на запись ("w").
Открытие файла на запись в режиме "w" стирает его содержимое.
Для добавление данных в конце файле следует использовать флаг "a" */
if ((fp = fopen("test.txt", "w")) == NULL)
{
    // Обработка ошибки
}
/* Функция fprintf аналогична функции printf, только первым
аргументом функции является файловый указатель. */
fprintf(fp, "Hello, world\n");
/* Закрытие файла */
fclose(fp);
```

stdlib.h: сортировка и поиск

```c
#include <stdio.h>
#include <stdlib.h>

int cmp(const void* a, const void* b)
{
    return *(int*)a - *(int*)b;
}
int main()
{
    int test[] = {88, 56, 100, 2, 25};
    int len = sizeof(test)/sizeof(int);
    /* Печать исходного массива */
    for(int i = 0; i < len; i++)
        printf("%d ", test[i]);
    printf("\n");

    /* Быстрая сортировка */
    qsort(test, len, sizeof(int), cmp);
    /* Печать отсортированного массива */
    for(int i = 0; i < len; i++)
        printf("%d ", test[i]);
    printf("\n");

    /* Поиск в отсортированном массиве методом деления пополам (бинарный поиск) */
    int key = 56;
    int* item = (int*)bsearch(&key, test, len, sizeof(int), cmp);
    if (item != NULL)
        printf("found\n");
    else
        printf("not be found\n");

    return 0;
}
```

string.h: строковые операции

```c
#include <string.h>

char msg[] = "Hello";
int len;
len = strlen(msg);      /* len=5 без символа \0*/
len = sizeof(msg);      /* len=6 вместе с \0 */

char cmd[100];
scanf("%s", cmd);

if (strcmp(cmd, "run") == 0) /* Сравнение строк */
{
     run();
}

char buf[256];
strcpy(buf, cmd); /* Копирование cmd в buf */
```

stdint.h: целочисленные типы

```c
#include <stdint.h>

uint8_t i; /* Беззнаковое целое размером 8 бит */
int16_t j; /* Целое со знаком размером 16 бит */
int32_t k; /* Целое со знаком размером 32 бита */
uint64_t l; /* Беззнаковое целое размером 64 бита */
```

stdbool.h: булев тип

```c
#include <stdbool.h>

bool flag = false;
bool enable = true;
```

assert.h: диагностика и отладка

```
/* Ключ NDEBUG отключает assert */
//#define NDEBUG
#include <assert.h>

int func(int n)
{
  /* n не должен быть равен нулю */
  assert(n != 0);

  /* Какие-то вычисления */

  return 0;
}

/* Если n == 0, то в этом месте программа остановится */
Assertion failed: (n == 0), function func, file assert.c, line 7.
[1] 378 abort ./assert
```

time.h: время

```c
#include <time.h>

/* Количество секунд начиная с 1 января 1970 (unixtime) */
time_t t = time(NULL);

/* Преобразовать unixtime в структуру tm с полями
   int tm_sec;     // секунды (0 - 60)
   int tm_min;     // минуты (0 - 59)
   int tm_hour;    // часы (0 - 23)
   int tm_mday;    // день месяца (1 - 31)
   int tm_mon;     // месяц года (0 - 11)
   int tm_year;    // год - 1900
   int tm_wday;    // день недели (Воскресенье == 0, Понедельник == 1)
   int tm_yday;    // день года (0 - 365)
   int tm_isdst;   // с учетом летнего времени
   char *tm_zone;  // аббревиатура часового пояса
   long tm_gmtoff; // смещение от UTC в секундах
*/
struct tm* ptr = localtime(&t);

/* Напечатать: "year: 2023 month: 9 day: 1" */
printf("year: %4d month: %2d day: %2d",
        ptr->tm_year + 1900, ptr->tm_mon + 1, ptr->tm_mday);


/* Напечатать: "Fri Sep 1 9:20:00 2023" */
printf("s", asctime(ptr));
```

Заключение

- Язык программирования С:
  - Объявление и вызов функций;
  - Указатели;
  - Структуры, объединения, перечисления;
  - Оператор ветвления;
  - Операторы цикла;
  - Препроцессор.

- Стандартная библиотека С содержит множество функций, которые позволят не «изобретать велосипед».