Лекция 1 Введение. Язык программирования С

План курса «Отладочные средства микропроцессорных систем»:

Лекция 1: Введение. Язык программирования С

Лекция 2: Язык программирования С, применение для встраиваемых систем

Лекция 3: Стандартная библиотека языка С

Лекция 4: Ядро ARM Cortex-M3. Микроконтроллер Миландр К1986ВЕ92QI

Лекция 5: Этапы разработки микропроцессорных систем

Лекция 6: Разработка программ: компилятор, сборщик, отладчик, интегрированная среда разработки

Лекция 7: Внутрисхемная отладка, загрузка программы, трассировка

Лекция 8: Архитектура программного обеспечения

Лекция 9: Периферийные модули: Timer, DMA, ADC, DAC

Лекция 10: Периферийные модули: SPI, I2C, UART, CAN, LIN, Ethernet, SDIO, USB

4 лабораторных работы, курсовой проект, экзамен

Введение

- Язык программирования С и его применение для встраиваемых микропроцессорных систем;

- Микроконтроллер **Миландр К1986ВЕ92QI (MDR32F9Q2I);**
  - Ядро ARM Cortex-M3 – 32-х разрядное RISC ядро;
  - Тактовая частота до 80 МГц;
  - ПЗУ 128 Кбайт;
  - ОЗУ 32 Кбайт;
  - Периферия (АЦП, Таймеры, SPI, UART, ЦАП, USB, Ethernet и т.д.).

- Современные методики разработки и отладки встраиваемых микропроцессорных систем.

Литература

Основная литература:

- Б. Керниган, Д. Ритчи Язык программирования С.
- Микроконтроллер Миландр 1986ВЕ9хх:
  - Все пособия по 1986ВЕ9хх собраны на образовательном сайте Миландр http://edu.milandr.ru/library/ :
    - Благодаров А.В., Л.Л. Владимиров Программирование микроконтроллеров;
    - Алалуев Р. В. Основы программирования 32-разрядных микроконтроллеров 1986ВЕ91Т компании «Миландр»: руководство к выполнению лабораторных работ;
    - Огородников И.Н. Микропроцессорная техника: введение в Cortex-M3: учебное пособие (на базе 1986ВЕ92У);
  - Спецификация микросхем серии 1986ВЕ9хх (Datasheet).

Дополнительная литература:

- Elicia White Making Embedded Systems: Design Patterns for Great Software;
- Joseph Yiu The Definitive Guide to ARM Cortex-M3 and Cortex-M4 Processors.

Электронные ресурсы

- Материалы курса «Отладочные средства микропроцессорных систем» https://github.com/smirnovalnik/embedded_systems_course
- Стандартная библиотека периферии 1986х (Milandr MCU 1986x Standard Peripherals Library) https://github.com/eldarkg/emdr1986x-std-per-lib
- Документация и примеры для стандартной библиотеки периферии 1986х (Documentation to Milandr MCU 1986x Standard Peripherals Library) https://github.com/eldarkg/emdr1986x-std-per-lib-doc
- Спецификация на серию 1986BE9x http://ic.milandr.ru/upload/iblock/2ea/2ea1fef16f4aa9132a3ca415a66ab92c.pdf
- Интегрированная среда разработки MDK-Lite Edition. Версия для обучения. http://www2.keil.com/mdk5/editions/lite
- Ответы на все вопросы по языку С https://stackoverflow.com/

Язык программирования С

Основные вехи развития:

- 1972 – изобретен Д. Ритчи в лаборатории AT&T;
- 1978 – опубликована книга «Язык программирования С» Б. Керниган, Д. Ритчи (K&R C);
- 1989 – стандартизация языка (C89, ANSI C);
- 1990 – стандартизация языка ISO (C90);
- 1999 – наиболее используемый стандарт (C99);
- 2011 – изменения в стандартной библиотеке (C11).

Применяется в:

- системное программирование (95 % ядра Linux на С, драйверы);
- встраиваемые системы (от смартфонов до холодильников, от автомобилей до самолетов).

Язык программирования C

✓ Быстрый код;

✓ Минимальная среда исполнения (runtime);

✓ Практически полный контроль над аппаратным обеспечением;

✗ Практически полный контроль над аппаратным обеспечением;

✗ Небезопасная работа с памятью (нет проверки диапазона переменных, нет проверки типа данных);

✗ Не поддерживает современные парадигмы программирования.

Расширением языка являются: C++, Objective C.

Язык повлиял на Java, C#, JavaScript, Python.

Результат компиляции

Код на языке C

```
 1. void func(void)
 2. {
 3.   return;
 4. }
 5.
 6. void main(void)
 7. {
 8.    char i;
 9.
10.    i = 5;
11.    while (i > 0)
12.    {
13.        func();
14.        i--;
15.    }
16.    for (;;);
17. }
```

Результат компиляции для HCS08

```
...
 8:      char i;
 9:
10:      i = 5;
         LDA   #5
         TSX
         STA   ,X
         L5:
11:      while (i > 0)
12:      {
13:          func();
         BSR   func
14:          i--;
         TSX
         DEC   ,X
         TST   ,X
         BNE   L5
         LC:
15:      }
16:      for (;;);
         BRA   LC
17:  }
```

Первая программа на C по K&R

```c
/*
   Текст первой программы на языке C
*/

/* Директива препроцессора для добавления файла
из стандартной библиотеки C */
#include <stdio.h>

/* Функция main. Точка входа в программу */
int main(void)
{
   printf("Hello, world!\n"); /* Напечатать строку в терминал */
   return 0; /* Возвратить системе значение 0 - все прошло хорошо */
}
```

Первая программа на C по K&R. Компиляция и запуск

Операционная система	Компилятор
Linux	gcc
Windows	MinGW
MacOS	gcc

Компиляция из командной строки:

```
gcc example.c –o hello
```

Запуск из командной строки (Windows):

```
.\hello.exe
Hello, world!
```

Запуск из командной строки (Linux, MacOS):

```
./hello
Hello, world!
```

Имена объектов. Комментарии

```
te+st/ = 1; /* Ошибка: имена должны содержать только буквы, цифры и
символ _ */
1test = 3; /* Ошибка: имена не должны начинаться с цифры */
test = 4; /* OK */

/* Test и test разные объекты - регистр имеет значение */
Test = 5;
test = 6;

TestLab1 = 7; /* OK CamelCase */
lesson_number_1 = 8; /* OK snake_case */
MPEI_ER_02_13 = 2; /* OK SCREAMING_SNAKE_CASE */
is_valid_parameter(first_param); /* OK */

/* Многострочный комментарий по
стандарту C */
a = a + b; // Однострочный комментарий в стиле C++
```

Объявление переменных. Типы данных

```
/* Объявление целочисленной
переменной со знаком
(дополнительный код со
знаком) */
int i;
i = 13;
/* Объявление беззнаковой
целочисленной переменной */
unsigned int j;
j = 0;
/* Объявление переменной
с плавающей запятой */
float a, b;
a = 10.0;
b = 20.0;
/* Объявление и инициализация
символьной переменной */
char c = 'a';
```

Тип данных	Размер, бит	Диапазон
char	8	-128 – 127 или 0 – 256
short, signed short	16	-32768 -32767
int, signed int	32	-2^{31} – (2^{31} – 1)
long long, signed long long	64	-2^{63} – (2^{63} – 1)
unsigned char	8	0 – 256
unsigned short	16	0 – 65535
unsigned int	32	0 – 2^{32}
unsigned long long	64	0 – 2^{64}
float	32	1,175494351e-38 – 3,402823466e+38
double	64	2,2250738585072014e-308 – 1,7976931348623158e+308

Размеры типов данных зависят от архитектуры (центрального процессора и шины данных) и от компилятора.

Оператор присваивания

```
/* Объявление и инициализация */
int a = 1;
int b = 2;
int tmp;

/* Обмен значений a и b */
tmp = a;
a = b;
b = tmp; /* a = 2, b = 1 */

int i, j, k;

/* Множественное присваивание */
i = j = k = 0;
```

Арифметические операции

```
int a = 0;
int b = 2;
int c;
/* Сложение */
c = a + b; /* c = 2 */
c = c + 2; /* c = 4 */
/* Вычитание */
c = b - a; /* c = 2 */
/* Умножение */
c = 4 * b; /* c = 8 */
/* Деление */
a = 10;
c = a / 2; /* a = 5 */
c = a / 100; /* a = 0 */
/* Остаток от деления */
a = 13;
c = a % 10; /* c = 3 */
```

Оператор	Описание
+	сложение
-	вычитание
/	целочисленное деление для char, short, int, long, деление для float, double
*	умножение
%	взятие остатка от целочисленного деления

Арифметические операции: деление

```
int a;

/* Целочисленное деление */
a = 1 / 2; /* a = 0 */
a = 10 / 100; /* a = 0 */
a = 3 / 2; /* a = 1 */
a = 11 / 2; /* a = 5 */

float c;

/* Деление чисел с плавающей запятой */
c = 1.0 / 2.0; /* c = 0.5 */
c = 1 / 2.0; /* c = 0.5 */
c = 1.0 / 2; /* c = 0.5 */
c = 1 / 2; /* c = 0.0 */
```

В простых микропроцессорных системах, как правило, отсутствуют аппаратные блоки работы с числами с плавающей запятой. Компилятор производит программную эмуляцию данных операций и поэтому эти операции занимают множество циклов.

float по стандарту IEEE 754

	Знак	Показатель степени	Мантисса
Бит	31	30:24	23:0

Арифметические операции: сокращенная форма

```
int i = 0;

i++; /* i = 1 */
++i; /* i = 2 */

/* Разница между префиксной и
постфиксной формой */
i = 0;
a = i++; /* a = 0, i = 1 */
a = ++i; /* a = 2, i = 2 */
i = 0;
i += 10; /* i = 10 */

a = 2;
i /= a + 3; /* i = 2 */
```

Оператор	Описание	Действие
`i++;`	Инкремент, постфиксная форма	`i = i + 1;`
`++i;`	Инкремент, префиксная форма	`i = i + 1;`
`i--;`	Декремент, постфиксная форма	`i = i - 1;`
`--i;`	Декремент, префиксная форма	`i = i - 1;`

`i += 1;`
`i += 2;`
`i -= 3;`
`i *= 2;`
`i /= 2;`
`i %= 2;`
	Сокращенная форма	`i = i + 1;`
		`i = i + 2;`
		`i = i - 3;`
		`i = i * 2;`
		`i = i / 2;`
		`i = i % 2;`

Битовые операции

```
/* 0xA - шестнадцатеричная форма,
012 - восьмеричная форма,
10 - десятичная форма */

int a = 0x88;

a = a & 0xFE; /* Сброс бита a = 0x80 */

a = a | 0x02; /* Установка бита a = 0x82 */

a = a ^ 0x01; /* Инверсия бита a = 0x83 */
a = a ^ 0x01; /* Инверсия бита a = 0x82 */

a = ~a; /* Инверсия a = 0x7D */

a = 0x01;

a = a << 2; /* Сдвиг влево a = 0x04 */
a = a >> 1; /* Сдвиг вправо a = 0x02 */
```

Оператор	Описание
&	Побитовое И
|	Побитовое ИЛИ
^	Побитовое исключающее ИЛИ
~	Побитовая инверсия
>>	Сдвиг вправо
<<	Сдвиг влево

i &= 0xFE;
i |= 0x02;
i ^= 1;
i <<= 2;
i >>= 1;
	Сокращенная форма

Оператор ветвления

```
/* Простая форма без {} */
if (выражение)
    оператор;
```

```
/* Простая форма с {} */
if (выражение) {
    оператор1;
    оператор2;
}
```

```
/* Проверка на четность */
if (a % 2 == 0)
    cnt++; // Одно утверждение
```

```
/* Простая форма с {} */
if (is_ready) {
    timeout = 100;
    status = OK;
}
```

Оператор ветвления

```
/* Полная форма */
if (выражение) {
    оператор1;
}
else {
    оператор2;
}

if (выражение1) {
    оператор1;
}
else if (выражение2) {
    оператор2;
}
else {
    оператор3;
}
```

```
if (voltage < 100) {
    status = OK;
}
else {
    status = FAIL;
}

if (cmd == RUN) {
    run();
}
else if (cmd == STOP) {
    stop();
}
else {
    idle();
}
```

Операция сравнения и логические операции

```
if (a < 0) ) {
    ...
}

if ((status & 0x01) == 0) {
    ...
}

if ((a > 0) && (a != 10)) {
    ...
}

if (!is_stopped) {
    ...
}
```

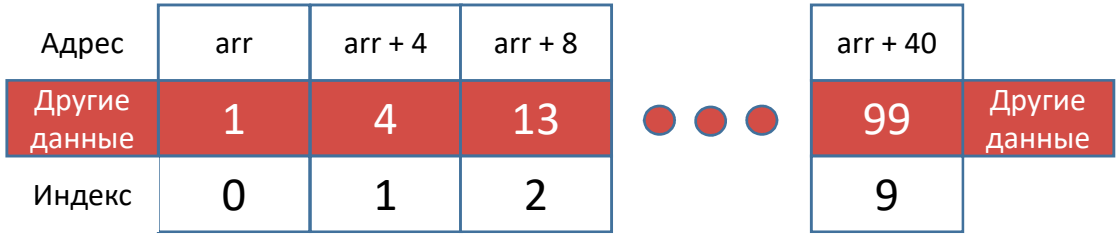
Оператор	Описание
> < >= <=	Больше Меньше Больше или равно Меньше или равно
== !=	Равно Не равно
&&	Логическое И
||	Логическое ИЛИ
!	Логическое НЕ

Оператор switch

```
switch (выражение) {
    case константа1:
        оператор1;
        оператор2;
        break;
    case константа2:
        оператор3;
        break;
    case константа3:
    case константа4:
        оператор4;
        break;
    default: // необязательно
        оператор5;
}
```

```
switch (cmd) {
    case CMD_RUN:
        set_pwm(100);
        run();
        break;
    case CMD_STOP:
        stop();
        break;
    case CMD_IDLE:
    case CMD_RESET:
        idle();
        break;
    default:
        error();
}
```

Массивы



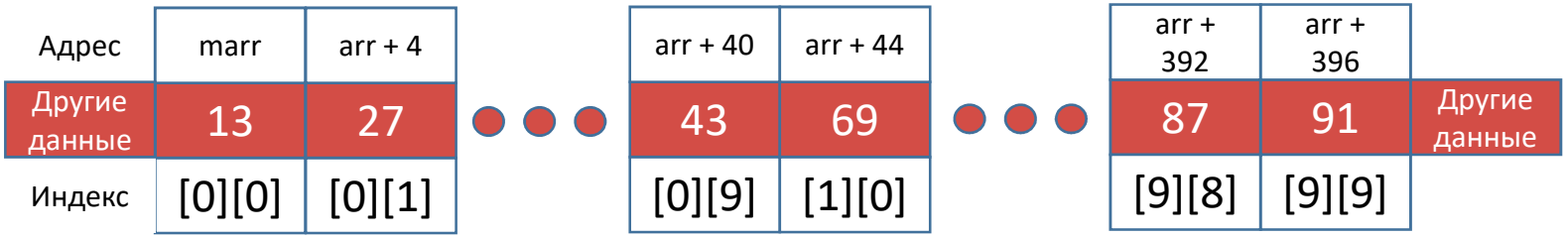

```
/* Объявление массива без инициализации */
int arr[10];

arr[0] = 1; /* Первый элемент */
arr[1] = 4; /* Второй элемент */
arr[9] = 99; /* Последний элемент */

/* Компилятор не проверяет выход за пределы */
arr[-1] = 100; /* Ошибка при исполнении */
arr[10] = 100; /* Ошибка при исполнении */

/* Объявление массива с инициализацией */
char letters[] = {'x', 'y', 'z'};
```

Многомерные массивы




```
int marr[10][10];

marr[0][0] = 13; /* Первый элемент */
marr[0][1] = 27;
marr[0][9] = 43;
marr[1][0] = 69;
marr[9][8] = 87;
marr[9][9] = 91; /* Последний элемент */
```

Операторы цикла

```
/* Цикл while */
while (выражение) {
    оператор;
}

/* Цикл do-while */
do {
    оператор;
} while (выражение);


/* Цикл for */
for (выраж1; выраж2; выраж3){
    оператор;
}
```

```
while (i < n) {
    a[i] = 0;
    i++;
}

do {
    spi_send_byte(cmd);
    status = spi_get_status();
} while (status != SPI_OK);

/* "Зануление" массива */
for (i = 0; i < n; i++) {
    a[i] = 0;
}

/* Бесконечный цикл */
for(;;);
while(1);
```

Операторы цикла

```
/* Оператор break */
while (выражение1) {
    ...
    if (выражение2)
        break;
    ...
}
```

```
/* Оператор continue */
while (выражение1) {
    ...
    if (выражение2)
        continue;
    ...
}
```

```
/* Проверка наличия элемента в
массиве */
int is_found = 0, i = 0;
while (i < n) {
    if (a[i] == target) {
        is_found = 1;
        break;
    }
    i++;
}
```

```
/* Обработка только положительных
элементов */
int i;
for (i = 0; i < n; i++) {
    if (a[i] < 0)
        continue;
    ...
}
```

Заключение

- Язык программирования С:
  - Основные сферы применения: системное программирование, встраиваемые системы;
  - Основные типы данных: целочисленные (char, int, long), с плавающей запятой (float, double), символьные (char);
  - Арифметические, побитовые, логические операции и операции сравнения;
  - Операторы ветвления (if, switch);
  - Операторы цикла (do while, while, for).