

# Разработка программного обеспечения для сигнальных процессоров TMS320C64xx в IDE Code Composer Studio

Часть II. Язык программирования С. Основные понятия.

# Синтаксис

- Идентификаторы (имена переменных, функций и типов данных):
  - Допустимые символы: цифры **0-9**, буквы **A-Z, a-z**, символ подчеркивания «**\_**».
  - Первый символ **не** может быть **цифрой**.
  - Запрещается использовать зарезервированные слова языка C (**int, if, while..**).
  - Идентификаторы чувствительны к регистру.
- Пробелы, символы табуляции, перевода строки являются разделителями.
- Комментарии: начало **/\*** , конец **\*/**.

Примеры:

```
int _Counter5;  
int another_var;
```

```
int 5g;  
int while;
```

```
int SomeVar;  
int somevar;
```

```
/* Комментарий многострочный  
*/
```

```
// Однострочный комментарий
```

```
int моя_переменная;
```

# Типы данных C

- базовые типы:

- `char` – символ (один байт);
- `int` – целое (обычно одно слово);
- `short` – короткое целое (слово или полуслово);
- `long` – длинное целое (слово или двойное слово)
- `float` – число с плавающей точкой одинарной точности;
- `double` – число с плавающей точкой двойной точности;
- `void` – отсутствие значения.

Примечание:

- для диапазонов значений справедливо следующее  $short \leq int \leq long$ , обычно  $long = 2 \times short$ .
- описания типов `short` и `long` эквивалентны описаниям типов `short int` и `long int`.

- квалификаторы `signed` (со знаком) и `unsigned` (без знака) применяются к `char` и любому целому типу.

Примечание: часто `signed` опускают, полагая, что имеют дело со знаковым, а `unsigned` всегда указывают явно. Например

`short` – знаковое целое 16 бит, а `unsigned short` – беззнаковое целое 16 бит.

# Размер данных

Тип	Размер, байт	Диапазон значений
signed char (char)	1	-128 .. 127
unsigned char	1	0 .. 255
signed short int (short)	2	-32768 .. 32767
unsigned short int (unsigned short)	2	0 .. 65535
signed int (int)	4	$-2^{31} .. 2^{31} - 1$
unsigned int (unsigned)	4	$0 .. 2^{32}$
signed long int (long)	4	$-2^{31} .. 2^{31} - 1$
unsigned long int (unsigned long)	4	$0 .. 2^{32}$
long long	8	$-2^{63} .. 2^{63} - 1$
unsigned long long	8	$0 .. 2^{64}$
float	4	1.175494351E-38 .. 3.402823466E+38
double	8	2.2250738585072014E-308 .. 1.7976931348623157E+308

# Константы

- Целые константы (имеют тип `int`):
  - **Десятичные:** цифры **0 – 9**, первой цифрой не должен быть **0**.  
Примеры:  
12 123 65535
  - **Восьмеричные:** цифры **0 – 7**, начинаются с **0**.  
Примеры:  
0111 = 73 (десятичное)  
0123 = 1\*64 + 2\*8 + 3 = 83 (десятичное)
  - **Шестнадцатеричные:** цифры **0 – 9**, буквы **a – f** или **A – F** для значений 10 – 15; Начинаются с **0x** или **0X**.  
Примеры:  
0x12 = 18 (десятичное)  
0xA3 = 163 (десятичное)  
0x1B9 = 1\*256 + 11\*16 + 9 = 441 (десятичное)
- Константы с плавающей точкой :  
Представляется числом с плавающей точкой двойной точности (`double`) и состоит из следующих частей:
  - целой части - последовательность цифр;
  - десятичной точки;
  - дробной части - последовательность цифр;
  - символа экспоненты **e** или **E**;
  - экспоненты в виде целой константы - может быть со знаком.Примеры:  
3.123 = 3.123 (десятичное)  
2.1E5 = 210000 (десятичное)  
4.03e-5 = 0.0000403 (десятичное)

# Константы

- **Символьные константы:**

Состоят из одного символа кода ASCII, заключенного в апострофы. Считаются данными типа `int`.

Примеры: `'A'` `'b'` `'7'` `'\n'` – начало строки.

- **Строковые константы:**

- Представляется последовательностью символов кода ASCII, заключенной в кавычки: `"..."`

Примеры :

`"this a string"` `"234555"` `"F"`

- Имеет тип `char[]` поскольку это массив символов.
- В конце каждой строки компилятор помещает символ конца строки `'\0'`.
- Каждая строковая константа даже если она идентична другой помещается в отдельное место памяти.

# Объявления переменных

- Переменные должны быть объявлены раньше чем будут использоваться. Объявление специфицирует тип и содержит список из одной или нескольких переменных этого типа. Например:

```
int max, min, average;
```

```
char array[100]; // массив из 100 переменных типа char.
```

- В своем объявлении переменная может быть инициализирована, например:

```
char symb = 'x';
```

```
int i = 0;
```

- К любой переменной в объявлении может быть применен модификатор *const* для указания того, что она не будет в дальнейшем изменять свое значение. Например:

```
const int maxVal = 0xbeaf;
```

- Модификатор *volatile* требует от компилятора каждый раз перечитывать значение данной переменной при обращении к ней.

# Арифметические операторы

- Бинарные операторы:
  - Умножение:  $op1 * op2$
  - Деление:  $op1 / op2$ . Деление целых сопровождается отбрасыванием дробной части.
  - Остаток от деления:  $op1 \% op2$
  - Сложение:  $op1 + op2$
  - Вычитание :  $op1 - op2$
- Унарные операторы:
  - -  $op$  или  $+op$
- Операторы инкремента и декремента:
  - Префиксные  $++op$  или  $--op$  (сначала изменяем значение, потом используем).
  - Постфиксные  $op++$  или  $op--$  (сначала используем значение, потом изменяем значение)



# Операторы отношения и логические операторы

**Значением выражений, содержащих такие операции, является 0 (Ложь) или 1 (Истина)**

- Операторы отношения:
  - $op1 > op2$ ,  $op1 < op2$ ,  $op1 \leq op2$  и  $op1 \geq op2$
- Операторы сравнения на равенство
  - $op1 \neq op2$  и  $op1 == op2$
- Логические операторы `||` (ИЛИ) и `&&` (И). Выражения между которыми стоят такие операторы вычисляются слева на право. Вычисление прекращается как только становится ясна истинность или ложность результата.

Пример: `if (A > B && (B != 0 || A != 0)) { ... }`

- Унарный оператор `!` преобразует ненулевой операнд в 0, а нуль в 1. Обычно он используется в конструкциях вида:

`if (!valid) { ... }`, что эквивалентно `if (valid == 0) { ... }`

# Побитовые операторы

## Применяются только к целым типам

- $\&$  - побитовое И
- $|$  - побитовое ИЛИ
- $\wedge$  - побитовое исключающее ИЛИ
- $\ll$  - сдвиг влево
- $\gg$  - сдвиг вправо
- $\sim$  - побитовое отрицание (унарный)

## Примеры:

$n = n \& 0x3$  – обнуляет все разряды кроме младших двух

$x = x | 0xa$  – выставляет единицу во втором и четвертом разрядах

$y = y \ll 2$  – сдвиг влево на два разряда, что эквивалентно умножению на 4.

# Операторы и выражения присваивания

- простое присваивание:
  - `a = 345;`
  - `someVal = a;`
- присваивание совмещенное либо с бинарным логическим оператором, либо арифметическим оператором.

так выражение `i = i + 2` эквивалентно `i += 2;`

Определены следующие операторы:

`*=, /=, %=, +=, -=, <<=, >>=, &=, |=, ^=.`

# Управление порядком выполнения программы.

- Конструкция if – else  
`if (выражение)`  
    *инструкция1*  
`else` // данный оператор может отсутствовать  
    *инструкция2*
- Конструкция else – if  
`if (выражение)`  
    *инструкция1*  
`else if (выражение)`  
    *инструкция2*  
`else`  
    *инструкция3*
- Переключатель switch  
`switch (выражение)`  
`{`  
    *case конст. выражение: инструкции;*  
    *case конст. выражение: инструкции;*  
    *default: инструкции;*  
`}`

# Циклы

- цикл **while**  
`while (выражение)`  
`инструкция;`
- цикл **for**  
`for (выр1 ; выр2 ; выр3 )`  
`инструкция;`
  - `выр1` – инициализация цикла
  - `выр2` – условие продолжения выполнения
  - `выр3` – выражение вычисляемое при каждой итерации
- цикл **do-while**  
`do`  
`инструкция;`  
`while (выражение);`
- инструкция ***break*** – немедленный выход из самого внутреннего цикла или переключателя.
- инструкция ***continue*** – вынуждает ближайший объемлющий ее цикл досрочно начать следующий шаг итерации.

# Приведение типа

- **Приведе́ние ти́па** — преобразование значения переменной одного типа в значение другого типа.
  - Явное приведение типа - указывается тип переменной, к которому необходимо преобразовать исходную переменную

```
char a = 11;
int b = (int)a;
```
  - Неявное приведение типа - преобразование происходит автоматически, по правилам, заложенным в данном языке программирования.

```
char a = 10;
char b = 20;
unsigned int = a*b;
```

# Приведения типов в арифметических выражениях

- При переводе числа из вещественного типа в целочисленный, дробная часть откидывается.
- Операнды типов `char` или `short` преобразуются к типу `int`, аналогично `unsigned char` или `unsigned short` -> `unsigned int`, тип `float` преобразуется к типу `double`.
- Если один из операторов имеет тип `unsigned`, то другой преобразуется к типу `unsigned` и результат будет `unsigned`.
- Если один из операторов имеет тип `double`, то другой преобразуется к `double` и результат будет `double`.

## Список рекомендуемой литературы.

- Керниган Б., Китчи Д. Язык программирования С. 2004.
- Болски М. Язык программирования Си. Справочник. 1988.