Соглашение о вызове
MultiClet S1

Данный документ представляет собой описание соглашения о вызове функций, передаче аргументов и возвращаемого значения. Это соглашение используется компилятором clang языка C.

Соглашение о вызове использует следующие правила:

- все аргументы передаются через стек;
- порядок размещения аргументов на стеке слева на право или прямой, т. е. аргументы размещаются в том же порядке, в котором они перечислены при вызове функции;
- код, ответственный за очистку стека, расположен в вызывающей функции;
- для вызова и возврата используются исключительно инструкции установки адреса следующего параграфа (никаких других специфических инструкций просто не существует);
- код, ответственный за сохранение и восстановление содержимого регистров располагается в вызываемой функции;
- результат выполнения функции возвращается через регистр **RETV**.

Передача аргументов в функцию

Пример вызова функции:

```
double f(int long a, char c, double b, short c, int *p);

int main(int argc, char *argv[]) {
    return f(1, 'a', 0.75, 0x1234, &argc);
}
```

В результате, в момент фактической передачи управления на функцию **f** стек будет находиться в следующем состоянии (адреса слева указаны для наглядности):

0x09C	…	
0x100	RETURN_ADDRESS	← SP
0x104	1	
0x108	'a'	
0x10C		
0x110 0x114	0.75	
0x118	0x1234	
0x11C	&argc	
0x120	…	

Здесь следует обратить внимание на следующие нюансы:

- вершина стека в момент передачи управления на функцию f выровнена на 8;
- на вершине стека, куда указывает регистр SP, лежит адрес возврата из функции f;
- для фактических параметров функции, размер которых меньше четырёх байтов (типы char и short с различными квалификаторами и без них), всегда выделяется 4 байта;
- третий параметр типа double выровнен на 8 (располагается по адресу 0x110), путём пропуска 4 байтов (адреса 0x10C - 0x10F);
- размер указателя составляет 4 байта.

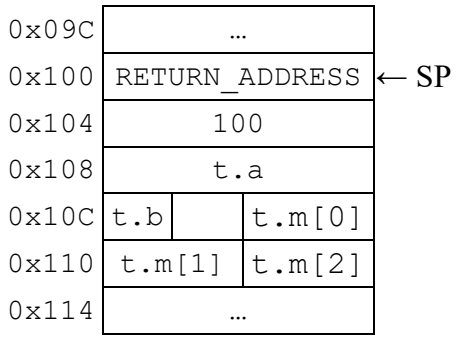
Передача структуры по значению в функцию

Пример вызова функции:

```
typedef struct Test_s {
    int a;
    char b;
    short m[3];
} Test;

int f(int i, Test t);
int main(int argc, char *argv[]) {
    Test t;
    return f(100, t);
}
```

В результате в момент фактической передачи управления на функцию f стек будет находиться в следующем состоянии (адреса слева указаны для наглядности):



Здесь следует обратить внимание на следующие нюансы:

- вершина стека в момент фактической передачи управления на функцию f выровнена на 8;
- на вершине стека, куда указывает регистр SP, лежит адрес возврата из функции f;
- выравнивание элементов структуры (элемент структуры t.m[0] выровнен на 2 байта, согласно типу элементов массива (располагается по адресу 0x10E), путём пропуска 1 байта (адрес 0x10D)).

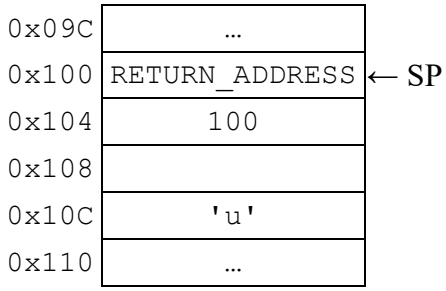
Передача пустой структуры по значению в функцию

Пример вызова функции:

```
typedef struct Test_s {
} Test;

int f(int i, Test t, char e);
int main(int argc, char *argv[]) {
    Test t;
    return f(100, t, 'u');
}
```

В результате в момент фактической передачи управления на функцию f стек будет находиться в следующем состоянии (адреса слева указаны для наглядности):



Здесь следует обратить внимание на следующие нюансы:

- вершина стека в момент фактической передачи управления на функцию f выровнена на 8;
- на вершине стека, куда указывает регистр SP, лежит адрес возврата из функции f;
- под второй параметр (пустая структура) место на стеке будет выделено, но никак не использовано ни в вызывающей, ни в вызываемой функции.

Возврат значения из функции

Для возврата значения из функции используется регистр **RETV**.

Возврат структуры по значению из функции

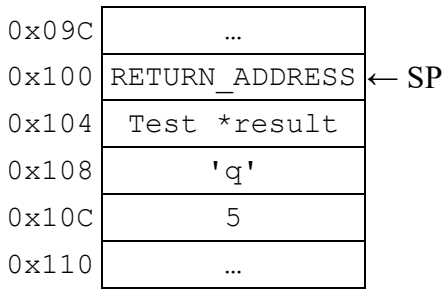
Пример вызова функции, возвращающей структуру по значению:

```
typedef struct Test_s {
        char a;
        short b;
        int c;
        double d[2];
} Test;

Test init(char a, short b)
{
        Test t;
        t.a = a;
        t.b = b;
        t.c = 1;
        t.d[0] = t.d[1] = 0;
        return t;
}

int main(int argc, const char *argv[]) {
    Test t = init('q', 5);
    return 0;
}
```

В результате в момент фактической передачи управления на функцию f стек будет находиться в следующем состоянии (адреса слева указаны для наглядности):



Здесь следует обратить внимание на следующие нюансы:

- вершина стека в момент фактической передачи управления на функцию f выровнена на 8;
- на вершине стека, куда указывает регистр SP, лежит адрес возврата из функции f;

- в следующем слоте стека (адреса 0x104 - 0x107) находится значение указателя на результирующую структуру, которую и необходимо проинициализировать. Сама же структура располагается в фрейме вызывающей функции в области размещения локальных переменных (расположенная и выравненная в памяти таким же образом, как при передаче структуры по значению в функцию), т. е. по смыслу представленный выше код эквивалентен следующему:

```
typedef struct Test_s {
        char a;
        short b;
        int c;
        double d[2];
} Test;

void init(Test *result, char a, short b)
{
        result->a = a;
        result->b = b;
        result->c = 1;
        result->d[0] = result->d[1] = 0;
}

int main(int argc, const char *argv[]) {
        Test t;
        init(&t, 'q', 5);
        return 0;
}
```