

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ» (ТУСУР)
Кафедра микроэлектроники, информационных технологий и управляющих
систем (МИТиУС)

ИСПОЛЬЗОВАНИЕ И ОРГАНИЗАЦИЯ СТЕКА
Отчет по лабораторной работе №1
по дисциплине «Аппаратные средства телекоммуникационных систем»

Студент гр. 735

_____ Д. А. Осипов

Принял:

Преподаватель кафедры
МИТУС

_____ Ю. Б. Шаропин
(оценка)

(дата)

Оглавление

1 Введение	3
2 Ход работы.....	4
2.1 Инициализация указателя стека	4
2.2 Переполнение стека	6
2.3 Передача параметров функции через стек	11
2.4 Использование двух указателей стека.....	13
3 Заключение	15

1 Введение

Цель работы – изучить организацию структуры стека и его использование в программах для микроконтроллера с архитектурой Cortex-M3.

Задание состоит в том, чтобы рассмотреть процедуры:

1. Инициализация указателя стека;
2. Переполнение стека;
3. Передача параметров функции через стек;
4. Использование двух указателей стека.

2 Ход работы

2.1 Инициализация указателя стека

Под инициализацией стека подразумевается, что происходит инициализация регистра отвечающего за указатель стека (в случае Cortex-M3 таковым регистром является «R13»). Именно в данный регистр помещается адрес вершины стека. В каталоге программы «IAR IDE» был найден файл «cstartup.s». В его содержимом были строки, написанные на языке ассемблера, в результате выполнения которых адрес вершины стека помещался в регистр-указатель. (Рисунок 2.1)

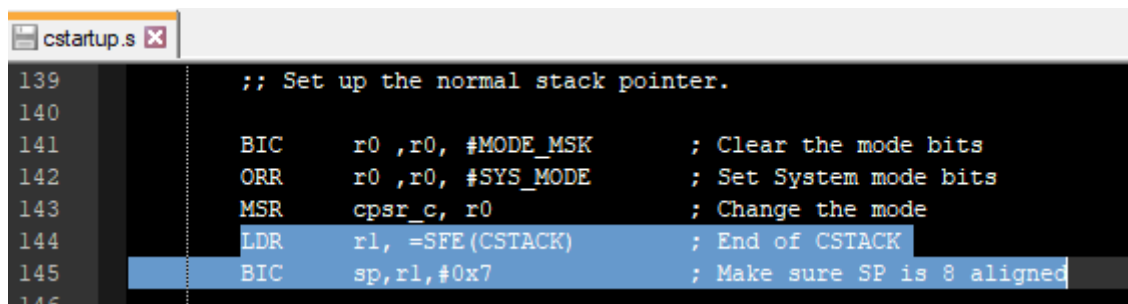


Рисунок 2.1 – Инициализация стека (часть содержимого файла «cstartup.s»)

В конфигурационном файле линковщика определяются адреса различных областей памяти. Таким образом получается, что адрес вершины стека складывается из адреса, который указывается на начало области оперативной памяти и размера стека. (Рисунок 2.2 – 2.3)

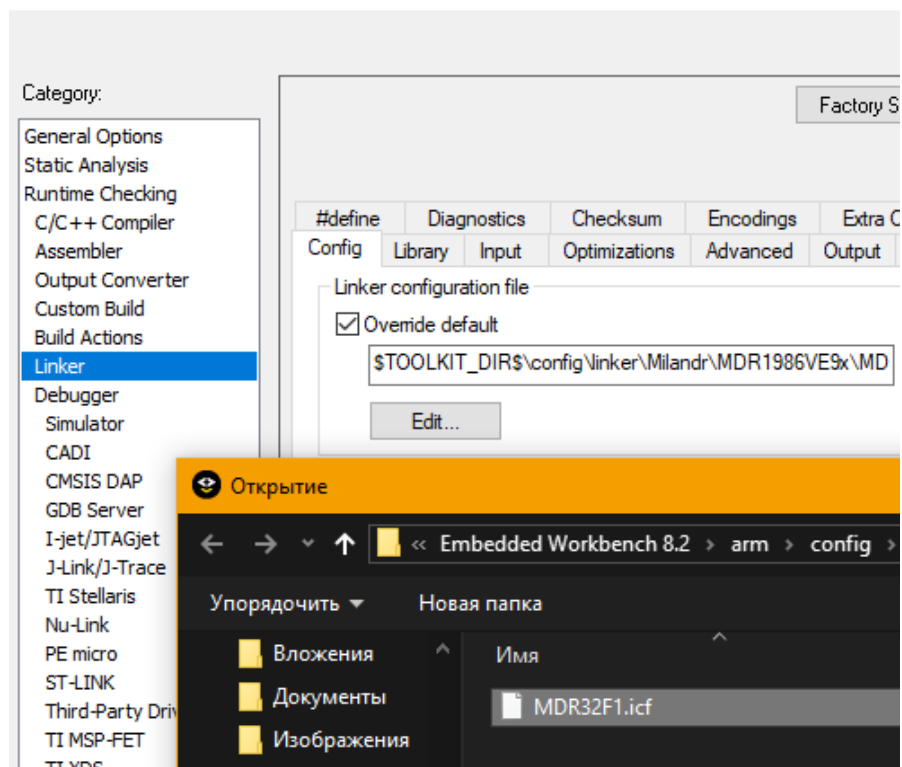
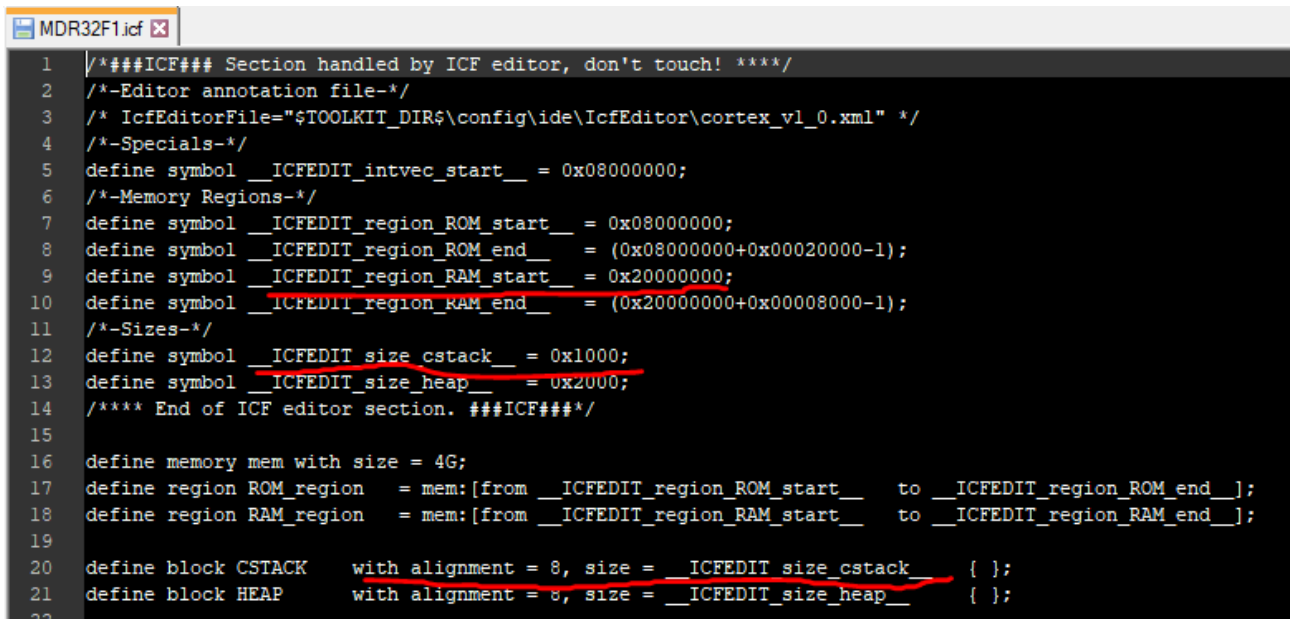


Рисунок 2.2 – Путь до конфигурационного файла



```

1  /****ICF**** Section handled by ICF editor, don't touch! ****/
2  /*-Editor annotation file-*/
3  /* IcfEditorFile="$TOOLKIT_DIR$\config\ide\IcfEditor\cortex_v1_0.xml" */
4  /*-Specials-*/
5  define symbol __ICFEDIT_intvec_start__ = 0x08000000;
6  /*-Memory Regions-*/
7  define symbol __ICFEDIT_region_ROM_start__ = 0x08000000;
8  define symbol __ICFEDIT_region_ROM_end__ = (0x08000000+0x00020000-1);
9  define symbol __ICFEDIT_region_RAM_start__ = 0x20000000;
10 define symbol __ICFEDIT_region_RAM_end__ = (0x20000000+0x00008000-1);
11 /*-Sizes-*/
12 define symbol __ICFEDIT_size_cstack__ = 0x1000;
13 define symbol __ICFEDIT_size_heap__ = 0x2000;
14 /**** End of ICF editor section. ****ICF****/
15
16 define memory mem with size = 4G;
17 define region ROM_region = mem:[from __ICFEDIT_region_ROM_start__ to __ICFEDIT_region_ROM_end__];
18 define region RAM_region = mem:[from __ICFEDIT_region_RAM_start__ to __ICFEDIT_region_RAM_end__];
19
20 define block CSTACK with alignment = 8, size = __ICFEDIT_size_cstack__ { };
21 define block HEAP with alignment = 8, size = __ICFEDIT_size_heap__ { };
22

```

Рисунок 2.3 – Содержимое конфигурационного файла

В этом файле заданы границы областей памяти, которые линковщиком будут определены как ROM (Read Only Memory), RAM (Random Access Memory), стек, куча и область векторов прерываний (рисунок 2.2). Адрес вершины стека задаётся суммой `__ICFEDIT_size_cstack__` (размер стека) и `__ICFEDIT_region_RAM_start__` (адрес начала оперативной памяти).

2.2 Переполнение стека

Размер стека задается изначально из файла линковщика и может возникнуть ситуация, когда стек будет переполнен, т.е. произойдет выход за границы адресного пространства отведенного стеку. Из содержимого файла на рисунке 2.3 было видно, что размер стека задан в 1000_{16} отсюда следует, что стек может хранить 1024 элемента (по той причине, что на элемент выделяется 4 байта).

Был реализован способ изменения регистра R13 для вызова обращения стека к области памяти не предназначенной для него. Однако в случае запуска программы, с ассемблерной вставкой, непосредственно присваивающей регистру R13 не принадлежащий области стека адрес, появляется ошибка, которая гласит о том, что данное действие недопустимо по причине того, что может привести к непредсказуемым результатам. Поэтому было решено сначала записать константу в регистр «R1», затем суммировать регистр-указатель и регистр «R1» таким образом, чтобы результат суммирования записался в регистр-указатель. Таким образом удалось получить предупреждение, что регистр-указатель хранит адрес, который не принадлежит к области стека. (Рисунок 2.4 – 2.5)

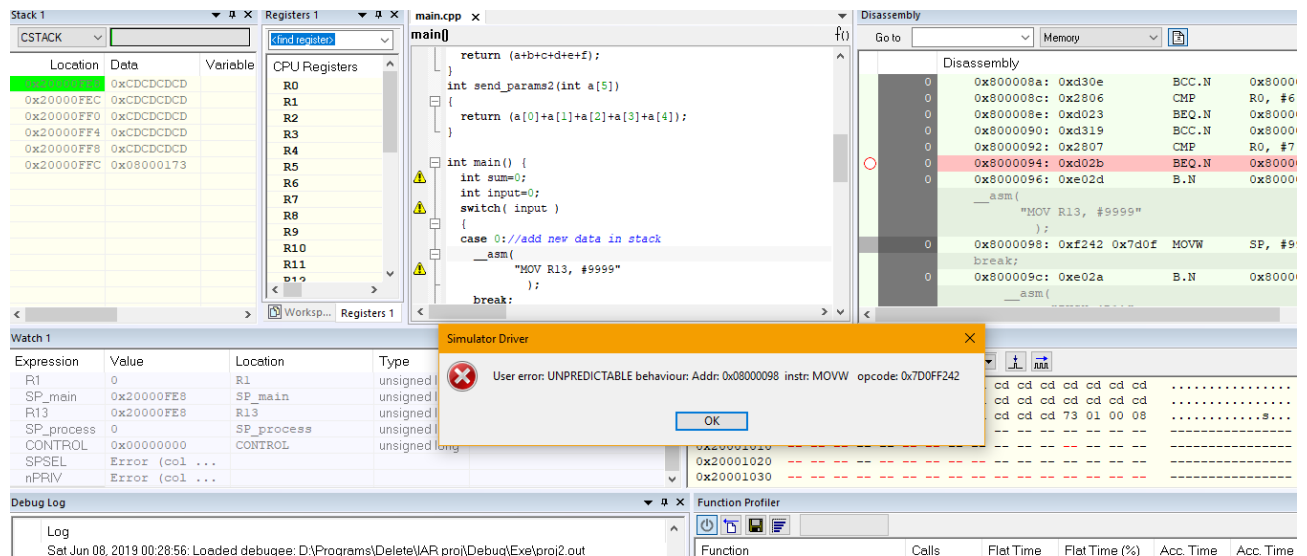


Рисунок 2.4 - Запись в регистр-указатель вершины стека

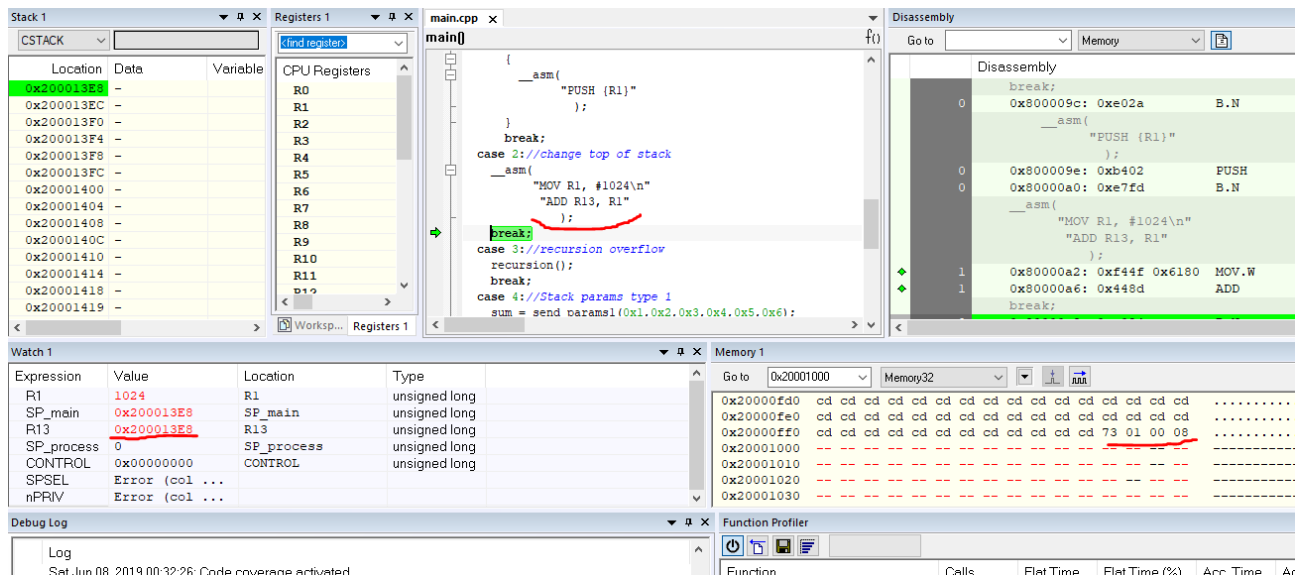


Рисунок 2.5 - Суммирование регистра-указателя вершины стека и регистра «R1»

Был реализован способ бесконечного рекурсивного вызова функции, поскольку при реализации такой функции не было поставлено условие «выхода» из нее. (Рисунок 2.6 – 2.9)

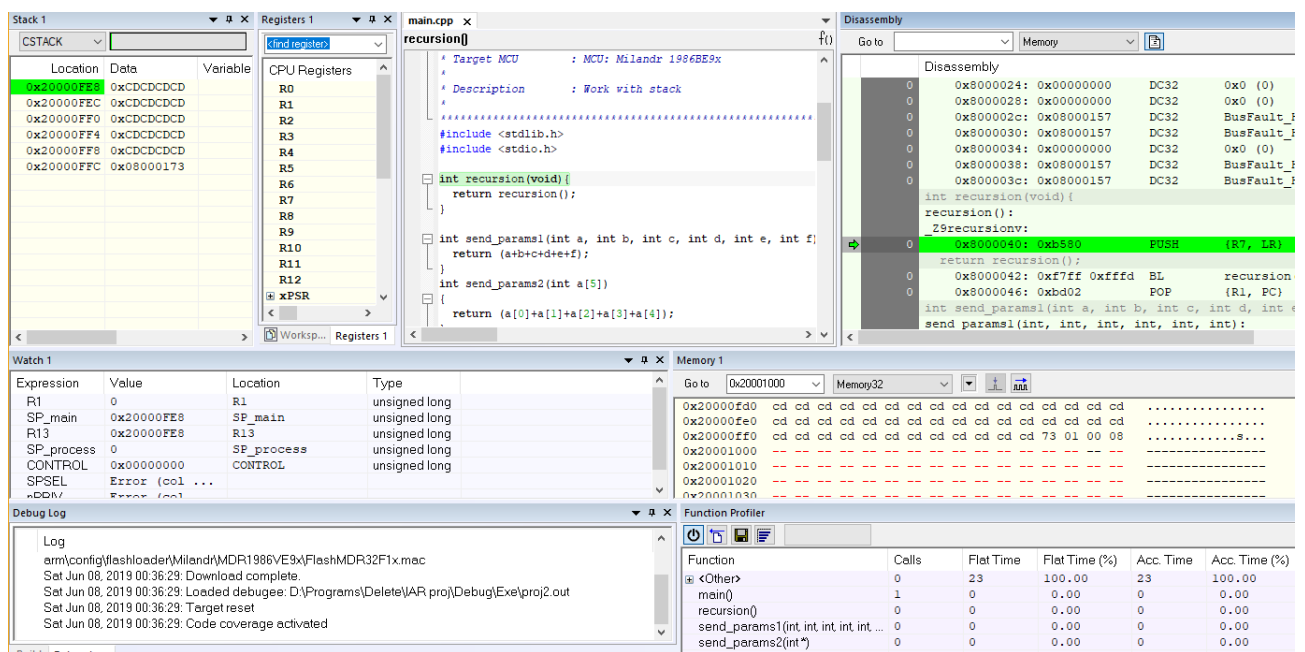


Рисунок 2.6 – Рекурсивная функция

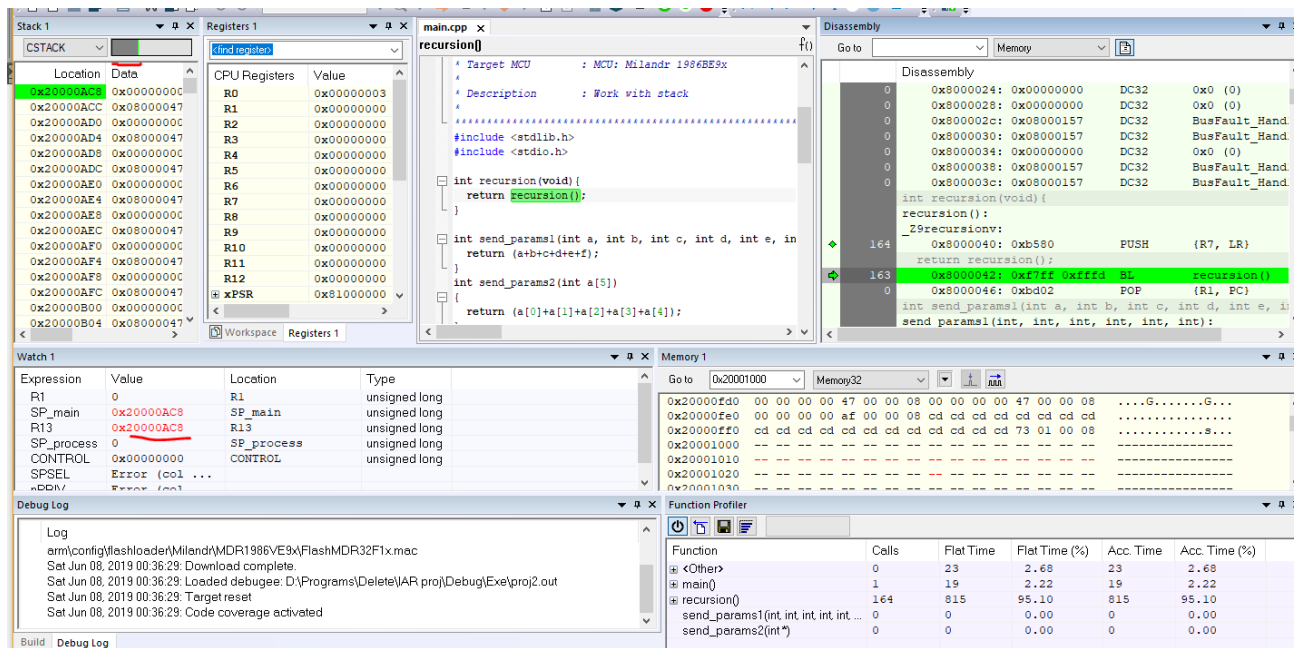


Рисунок 2.7 – Заполнение стека

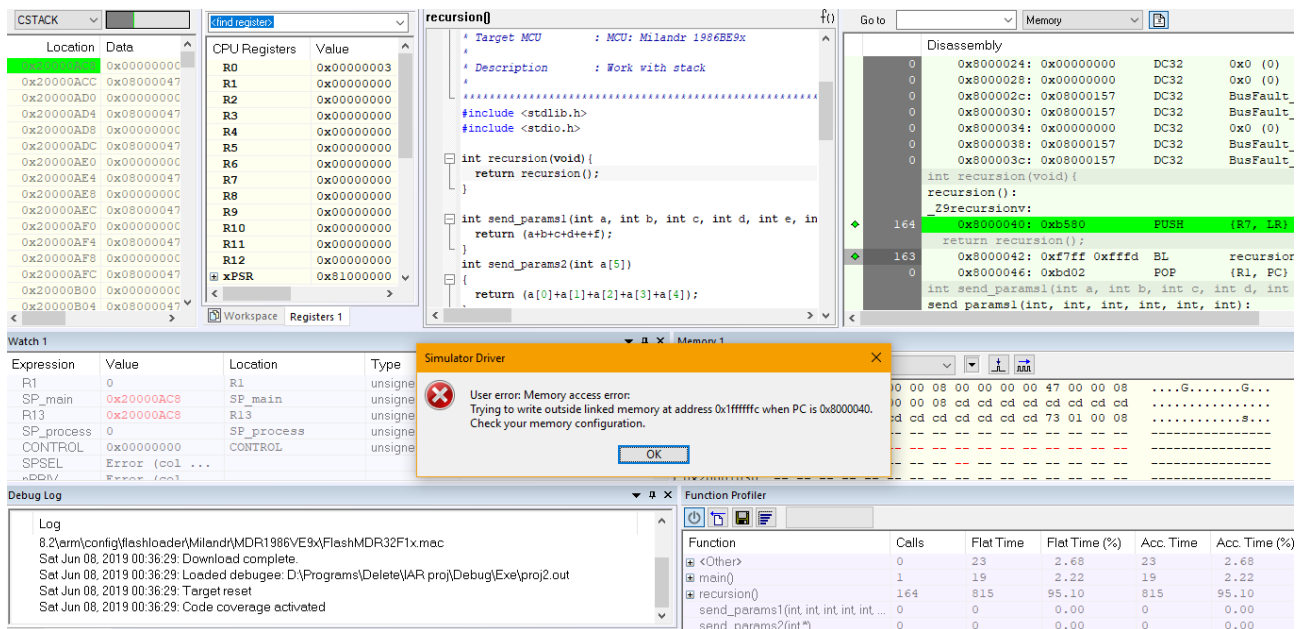


Рисунок 2.8 - Предупреждение о переполнении стека

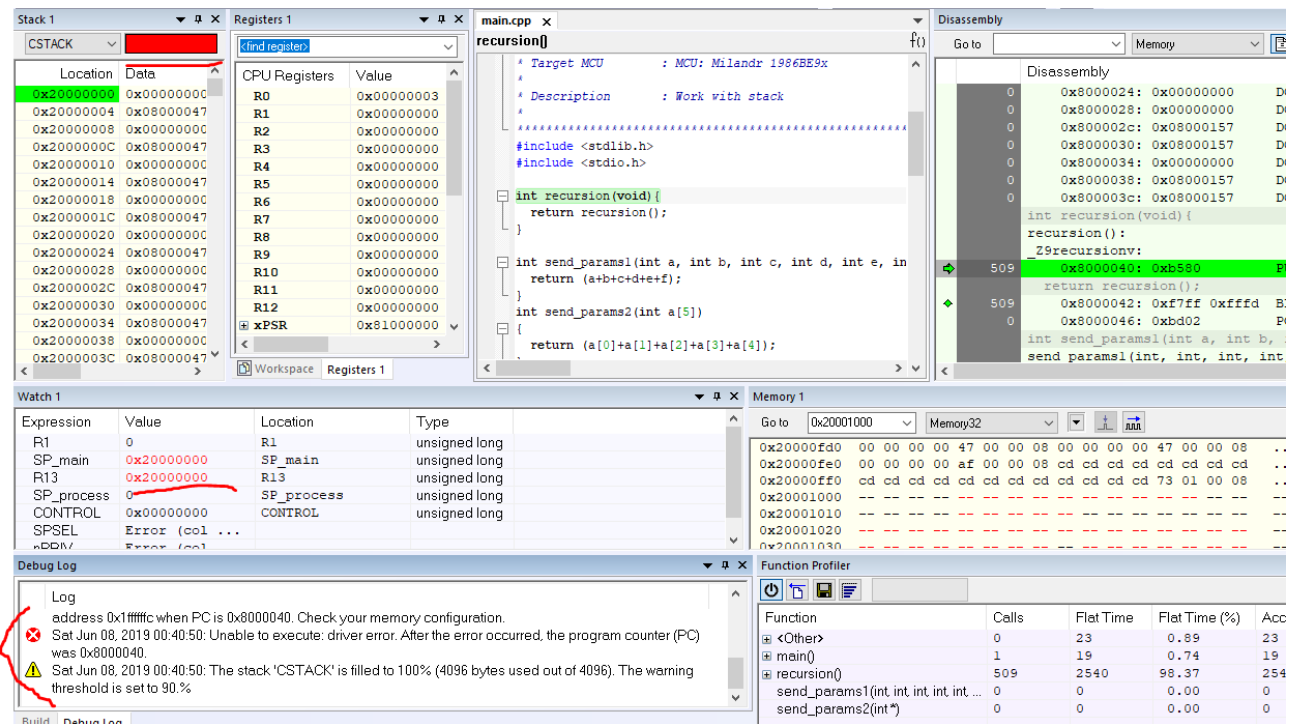


Рисунок 2.9 – Переполненный стек

Был реализован способ переполнения стека через бесконечный цикл в котором в стек вносилась некоторая переменная. (Рисунок 2.10 – 2.11)

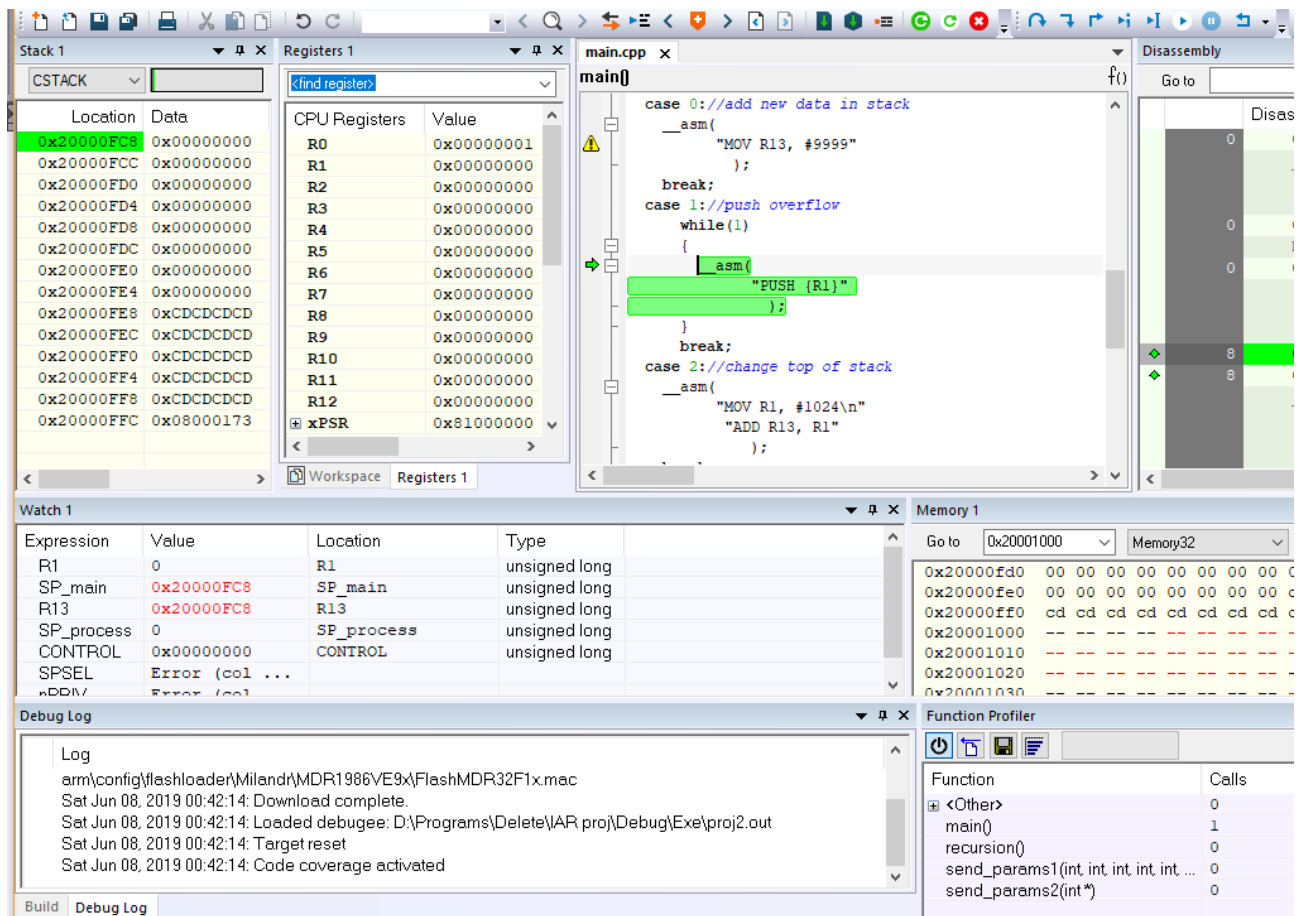


Рисунок 2.10 – Бесконечный цикл

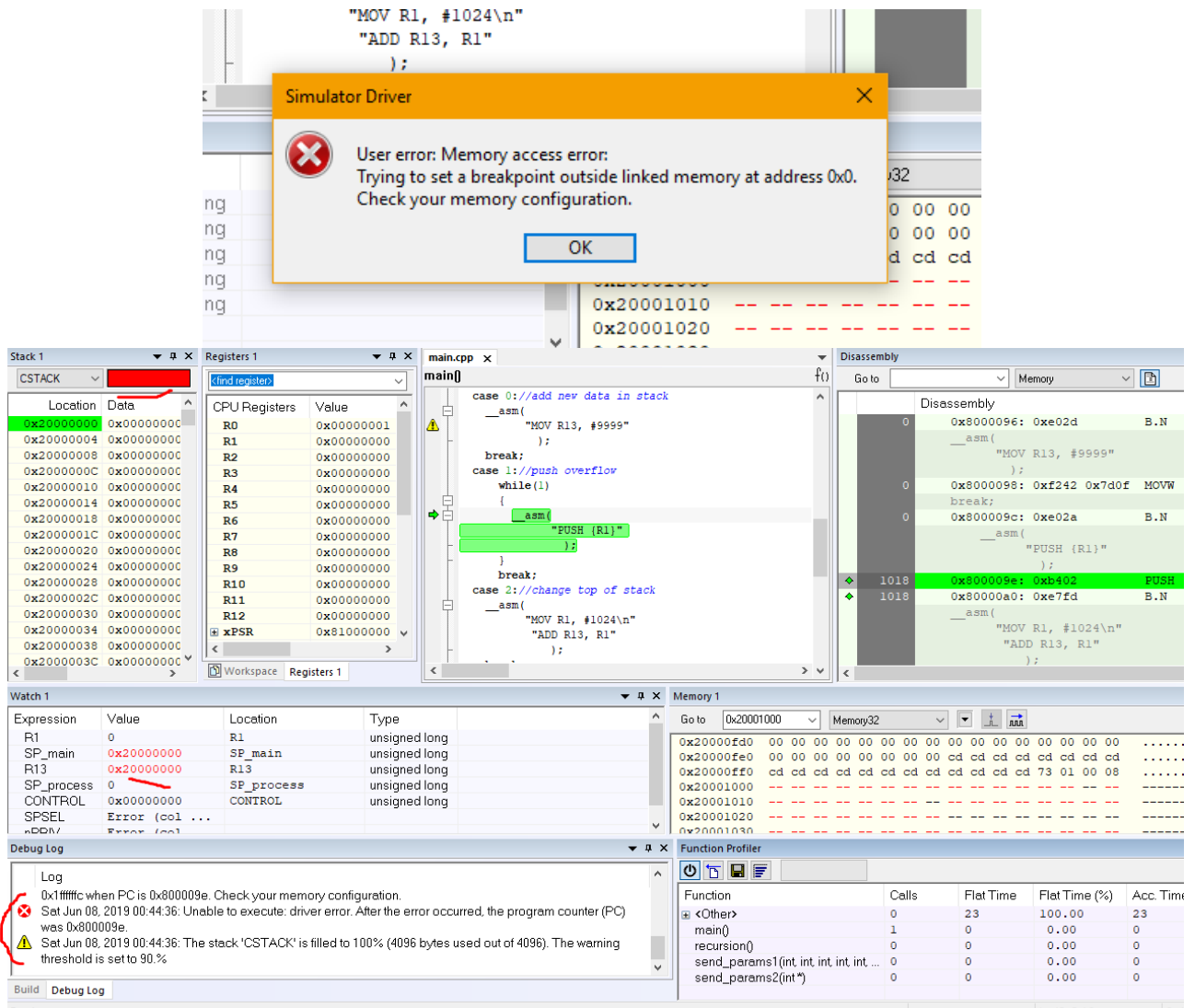


Рисунок 2.11 – предупреждение – «регистр-указатель хранит адрес, который не входит в область стека»

Значения регистра-указателя вершины стека уменьшается, что говорит о том, что стек является убывающим.

2.3 Передача параметров функции через стек

Одной из функций стека можно назвать хранение передаваемых параметров. При вызове функции, в которую необходимо передать параметры, часть параметров будет переданы через регистры и в случае, если параметров слишком много, будет задействован стек. Таким образом, эмпирическим путем было установлено, что число задействованных в передаче параметров регистров равно 4 с «R0» по «R3», остальные значения сначала сохраняются в стек. (Рисунок 2.12 – 2.13)

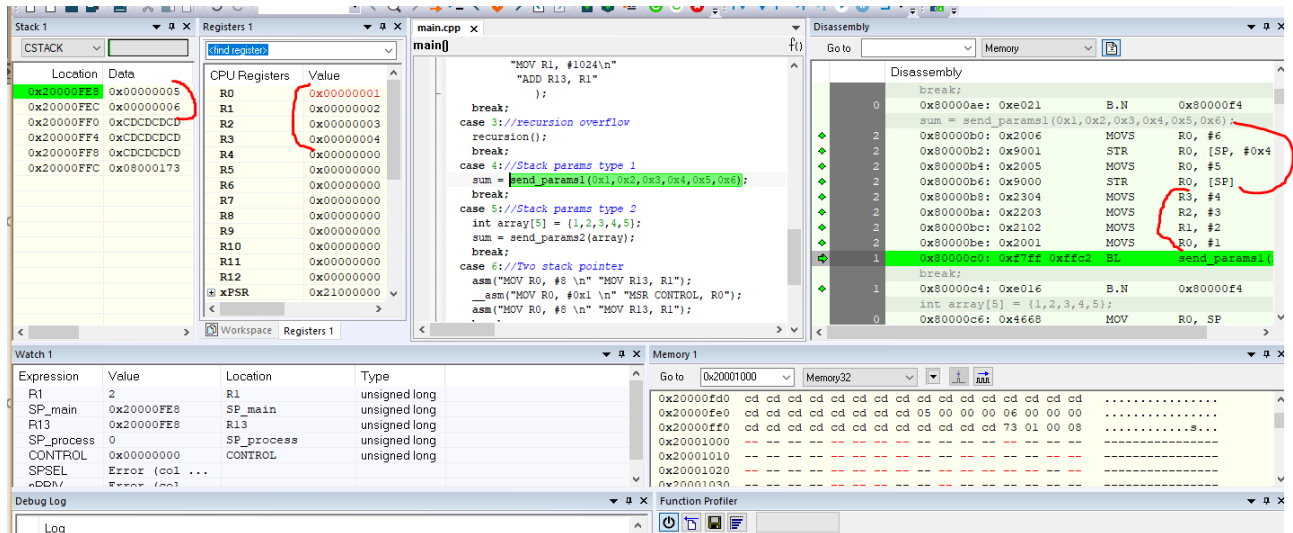


Рисунок 2.12 – Передача параметров в функцию через стек

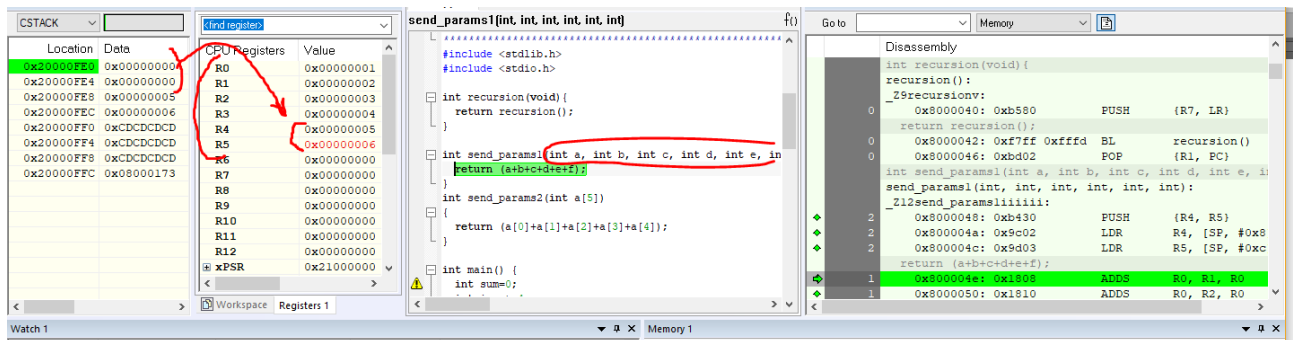


Рисунок 2.13 – Выгрузка переменных из стека и передача их в регистры «R5-6»

При детальном изучении дизассемблированного кода (рисунок 2.13) было выявлено, что передача параметров в функцию производится в обратном порядке, сначала в стек помещается значение «0x6», после «0x5», оставшиеся четыре параметра «0x4», «0x3», «0x2» и «0x1» помещаются соответственно в регистры «R3», «R2», «R1» и «R0».

Далее был рассмотрен вариант, передачи массива в функцию. При инициализации массива в функции «main» все его элементы помещаются в стек. А ассемблерном коде в регистр «R0» помещается адрес вершины стека, после, производится получение элементов массива из стека путем обращения к хранимому адресу в регистре «R0» со смещением. (Рисунок 2.14)

The screenshot displays a debugger interface with several panels:

- CSTACK:** A list of memory addresses and their corresponding data values. A red box highlights the address `0x20000FF8` with a value of `0x00000001`.
- CPU Registers:** A table showing the current values of CPU registers. The `R0` register contains `0x20000FFC`.
- main():** The source code of the `main` function, showing a recursive call and a loop. The line `sum = send_params2(array);` is highlighted.
- Disassembly:** The assembly code corresponding to the source code. The instruction `MOV R0, SP` is highlighted.
- Watch 1:** A table of watched expressions and their values. The expression `R1` has a value of `134218164`.
- Memory 1:** A table of memory addresses and their corresponding data values. The address `0x20000FF8` is highlighted.
- Debug Log:** A log of debug events, including "Download complete", "Loaded debuggee", "Target reset", and "Code coverage activated".
- Function Profiler:** A table showing the execution time of various functions. The function `main()` is highlighted.

Рисунок 2.14 – Инициализация массива

2.4 Использование двух указателей стека

Процессор Cortex-M3 содержит два указателя стека, которые хранятся в регистре «R13». Они объединены в банк, поэтому в каждый момент времени виден только один из них:

- основной указатель стека (Main Stack Pointer — MSP) — указатель стека, используемый ядром операционной системы и обработчиками исключительных ситуаций;
- указатель стека процесса (Process Stack Pointer — PSP) — указатель стека, используемый прикладной программой.

Таким образом получается, что в каждый момент доступен только один из них. Для того, чтобы узнать, какой стек сейчас используется, необходимо обратиться к специальному регистру «CONTROL». Этот регистр управления используется для задания уровня доступа и выбора указателя стека. В регистре содержится два бита. (Рисунок 2.15)

Бит	Описание
CONTROL[1]	Состояние стека: 1 — используется альтернативный стек; 0 — используется основной стек (MSP). При работе процессора в режиме потока альтернативным стеком является PSP. Для режима обработчика дополнительный стек не определён, поэтому при работе процессора в режиме обработчика этот бит должен быть сброшен в 0
CONTROL[0]	0 — привилегированный уровень доступа в режиме потока; 1 — пользовательский уровень доступа в режиме потока. В режиме обработчика процессор всегда работает на привилегированном уровне доступа

Рисунок 2.15 – Регистр управления Cortex-M3

Исходя из рисунка 2.15, для того чтобы переключиться на альтернативный стек, был написан код, который изменял значение данного регистра на «2₁₀» и таким образом регистр хранит значение «10₂», что приводит к тому, что переключается значение указателя стека (рисунки 2.16 и 2.17).

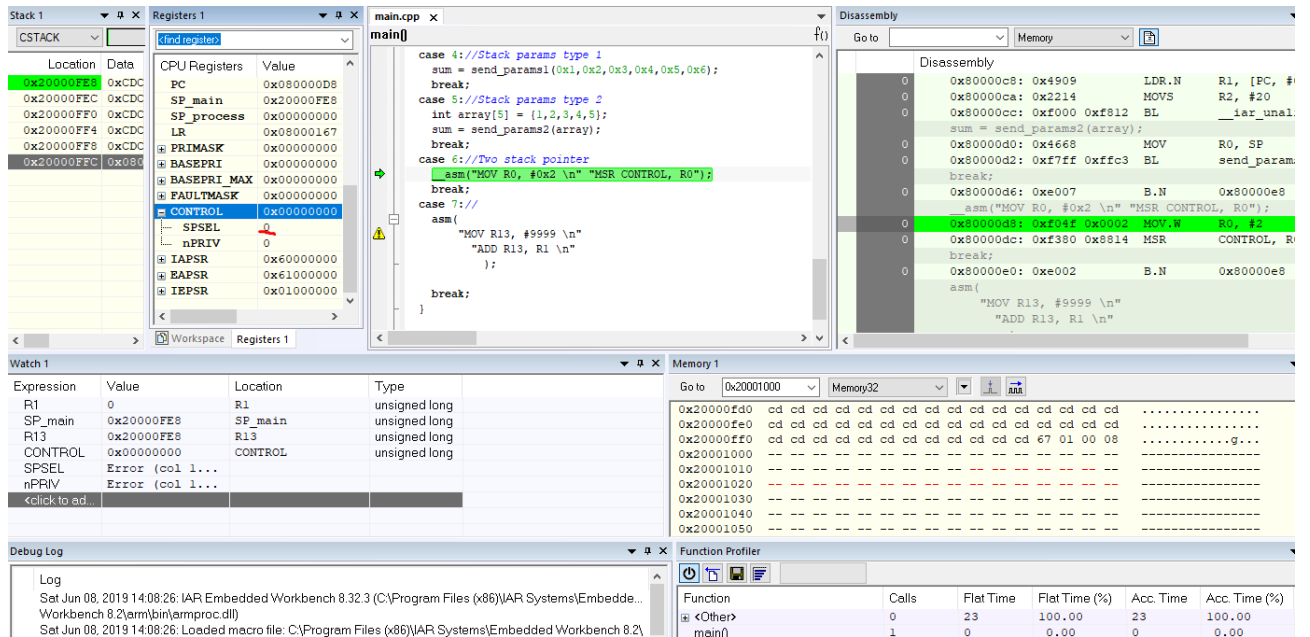


Рисунок 2.16 – До изменения регистра «CONTROL»

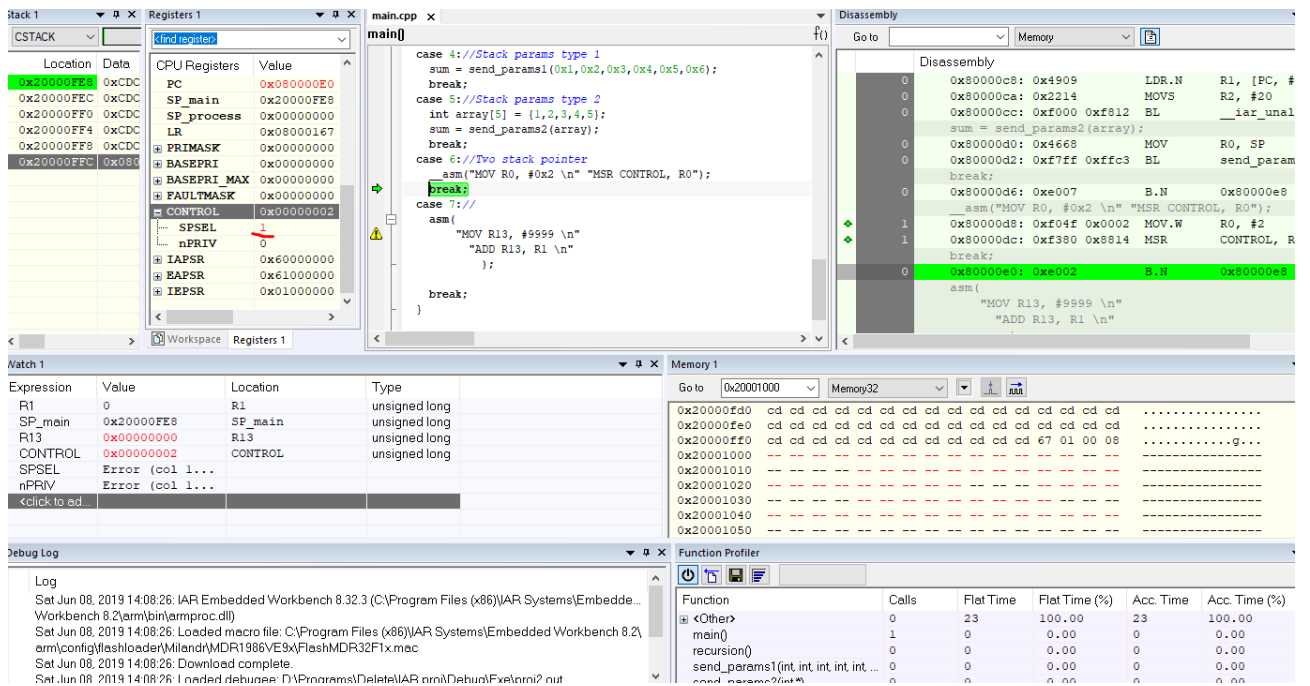


Рисунок 2.17 – После изменения регистра «CONTROL»

3 Заключение

В ходе выполнения лабораторной работы прошло ознакомление со стеком, был изучен процесс работы с ним, его инициализация и передача параметров в функцию с помощью него. Рассмотрены ситуации, которые приводили к переполнению стека, а также рассмотрен способ переключения между двумя указателями стека.

Был написан отчёт согласно требованиям ОС ТУСУР 01-2013.