**5)** Измените уровень оптимизации на Level 1 (Project -> Options for Target -> C/C++ -> выпадающее меню Optimization). Сравните дизассемблированный код с тем, что получился при отсутствии оптимизации. Как изменился дизассемблерный листинг при включении оптимизации?

Функция инициализации массива как была непонятной, так таковой и осталась. Количество и качество команд не изменилось никак.

В функции сортировки регистры r0 и r2 поменялись местами)

0x080003A6 B13A CBZ r2,0x080003B8

0x080003A8 1E53 SUBS r3,r2,#1

0x080003AA 461D MOV r5,r3

0x080003AC F8503023 LDR r3,[r0,r3,LSL #2]

0x080003B0 F8504022 LDR r4,[r0,r2,LSL #2]

0x080003B4 42A3 CMP r3,r4

0x080003B6 DC01 BGT 0x080003BC

0x080003B8 1C52 ADDS r2,r2,#1

0x080003BA E004 B 0x080003C6  
В оптимизированном коде используется лишний регистр для обращения к элементу массива.

Зато обмен элементами массива происходит гораздо быстрее:

0x08000444 F8523020 LDR r3,[r2,r0,LSL #2]

36: buf[i]=buf[i-1];

0x08000448 1E44 SUBS r4,r0,#1

0x0800044A F8524024 LDR r4,[r2,r4,LSL #2]

0x0800044E F8424020 STR r4,[r2,r0,LSL #2]

37: buf[i-1]=temp;

0x08000452 1E44 SUBS r4,r0,#1

0x08000454 F8423024 STR r3,[r2,r4,LSL #2]

Сверху неоптимизированный код, снизу оптимизированный.

0x080003BC F8403022 STR r3,[r0,r2,LSL #2]

37: buf[i-1]=temp;

0x080003C0 F8404025 STR r4,[r0,r5,LSL #2]

В функции поиска медианы тоже присутствуют махинации с расположением локальных переменных и счётчиков циклов. Возможно, в этом есть свой профит, но едва ли он большой, если учитывать, что в случае неоптимизированного кода стековый кадр всё равно умещался в регистрах. Тем не менее более оптимизированное распределение регистров позволяет уменьшить количество инструкций для поиска элементов в массиве и тп, что и будет показано ниже.

Этот шедевр меня вдохновил не бросить разбор дизассемблера:

0x080003BC 4283 CMP r3,r0

0x080003BE D100 BNE 0x080003C2

0x080003C0 E011 B 0x080003E6

Неоптимизированный код

0x0800034A 429C CMP r4,r3

0x0800034C D00E BEQ 0x0800036C

Оптимизированный код

Это происходило сравнение на равенство индексов) Оптимизация позволила искать равенство, а не отсутствие неравенства.

Опять же более адекватное сравнение элементов массива:

0x080003CE F8526023 LDR r6,[r2,r3,LSL #2]

0x080003D2 F8527020 LDR r7,[r2,r0,LSL #2]

0x080003D6 42BE CMP r6,r7

0x080003D8 DD01 BLE 0x080003DE

0x080003DA 1C49 ADDS r1,r1,#1

0x080003DC E002 B 0x080003E4

Сверху неоптимизированный код, снизу оптимизированный.

0x0800035A DD02 BLE 0x08000362

0x0800035C F1020201 ADD r2,r2,#0x01

0x08000360 E004 B 0x0800036C

В остальном же работа программы почти не изменилась.