МИЛАНДР
ГРУППА КОМПАНИЙ

Микроконтроллер для управления двигателями

на базе процессорного ядра ARM Cortex-M4F — двухъядерная архитектура

Галимзянов А. Т.

Москва

16.04.2019

Схема включения ядер в режиме LOCKSTEP

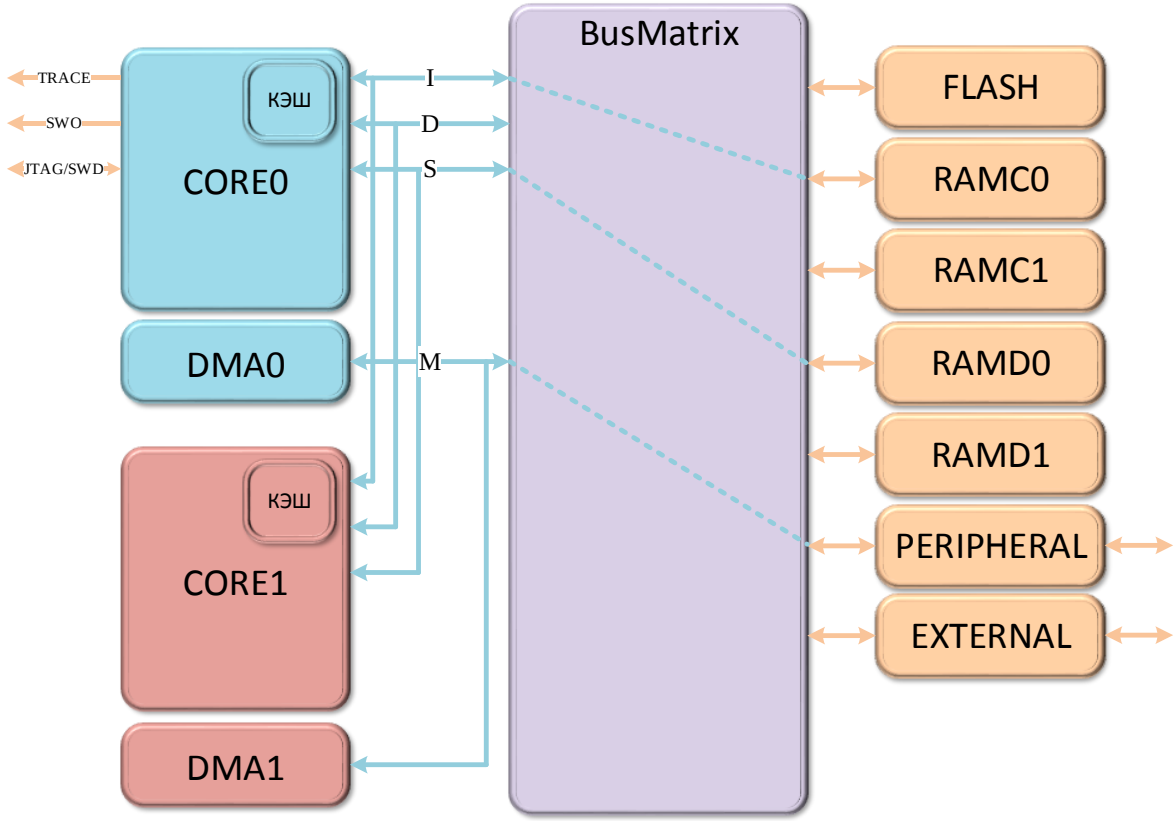
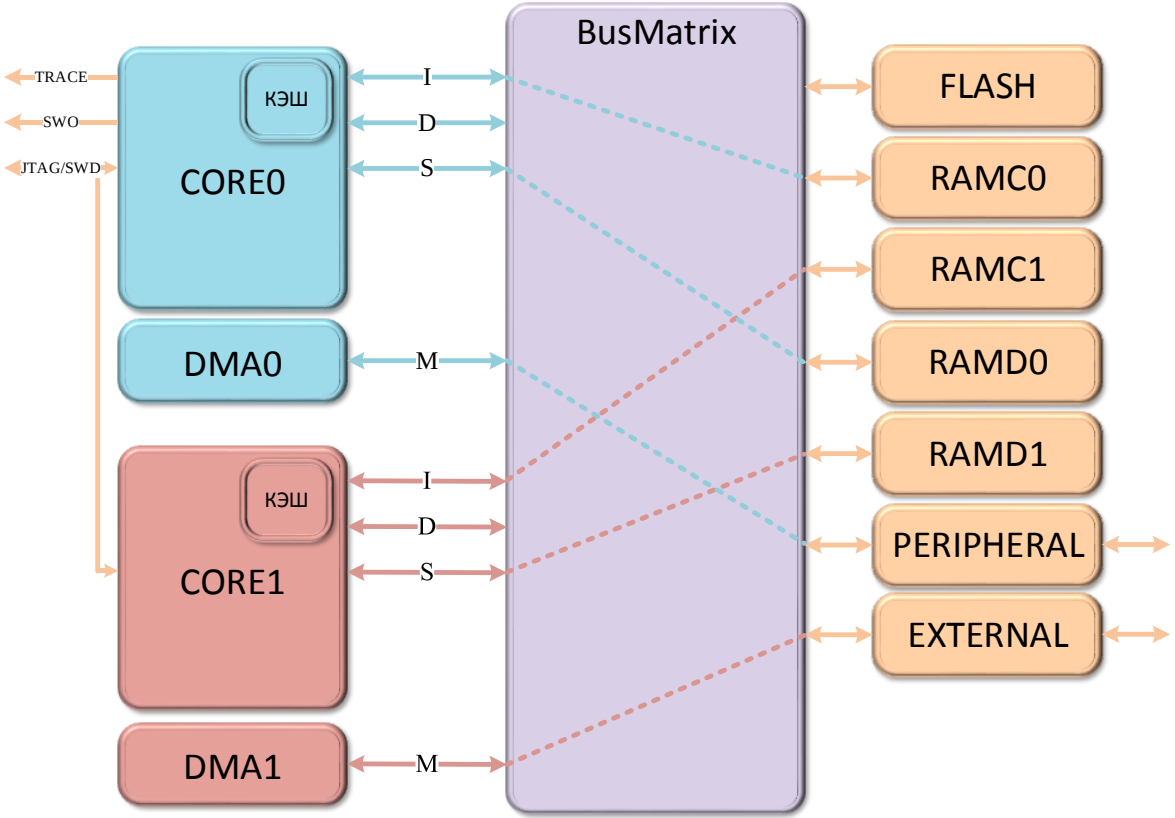


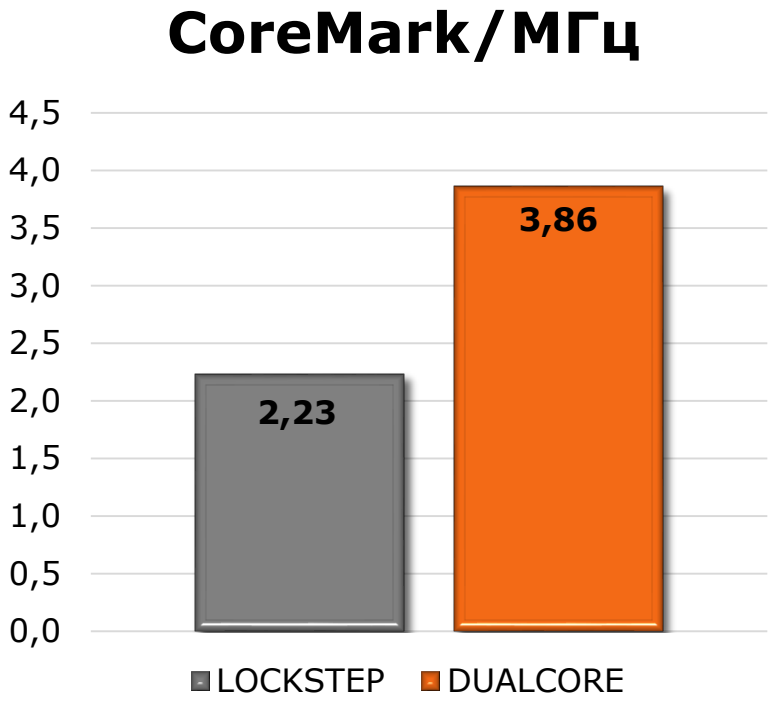
Схема включения ядер в режиме DUALCORE



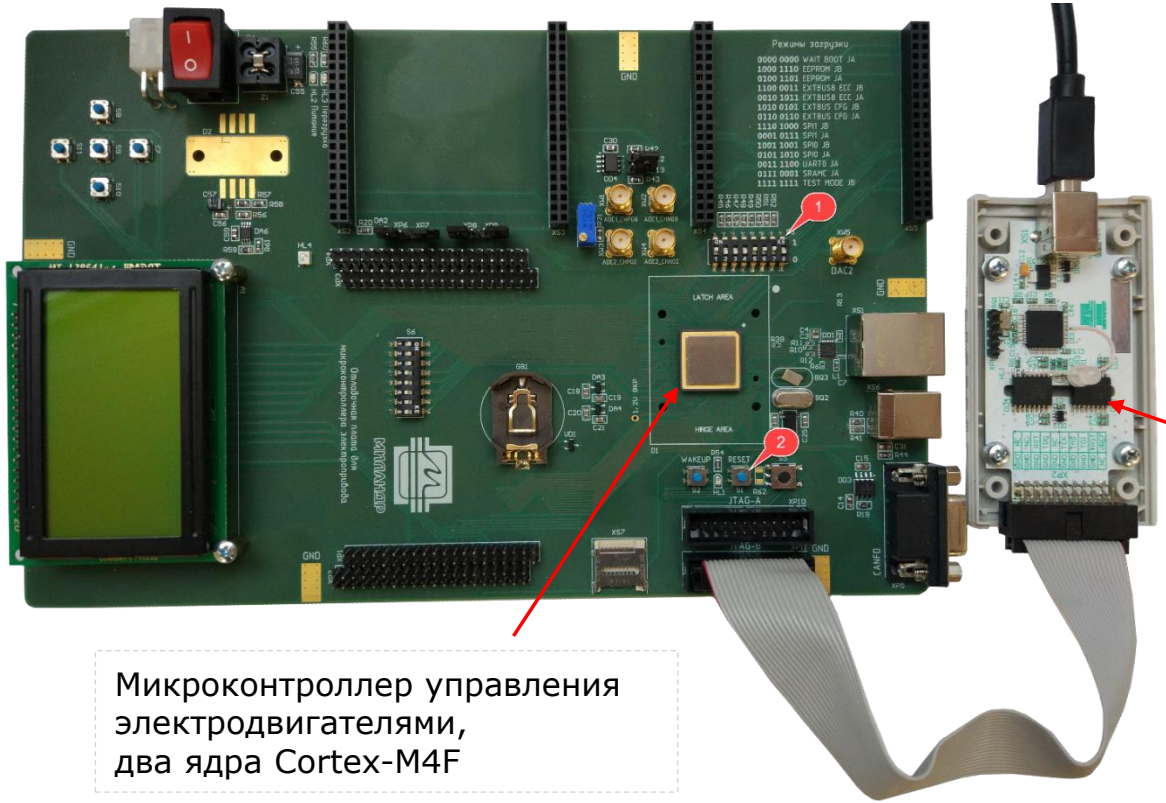
Сравнение производительности

Debug (printf) Viewer | Event Counters | boot2.c | core_main.c | core_portme.c

```
CODE    01000000
STACK0 20008000
STACK1 20010000
CPU clock 8000000 Hz
Coremark system initialized!
2K performance run parameters for coremark.
CoreMark Size    : 666
Total ticks      : 103805750
Total time (secs): 12.975719
Iterations/Sec   : 30.826809
Iterations       : 400
Compiler version : ARMCC 5060750
Compiler flags   : ____
Parallel hard : 2
Memory location  : STACK
seedcrc          : 0xe9f5
[0]crclist       : 0xe714
[1]crclist       : 0xe714
[0]crcmatrix     : 0x1fd7
[1]crcmatrix     : 0x1fd7
[0]crcstate      : 0x8e3a
[1]crcstate      : 0x8e3a
[0]crcfinal      : 0x382f
[1]crcfinal      : 0x382f
Correct operation validated. See readme.txt for run and reporting rules.
Data Cache Hits = 606
Data Cache Misses = 7
Instruction Cache Hits = 58436462
Instruction Cache Misses = 186831
CoreMark 1.0 : 30.826809 / ARMCC 5060750 ____ / STACK / 2:hard
```



Отладочная плата



1. Выбрать режим загрузки Flash + JTAG-B

2. Нажать Reset

CMSIS-DAP совместимый адаптер JTAG/SWD с гальванической развязкой

Микроконтроллер управления электродвигателями, два ядра Cortex-M4F

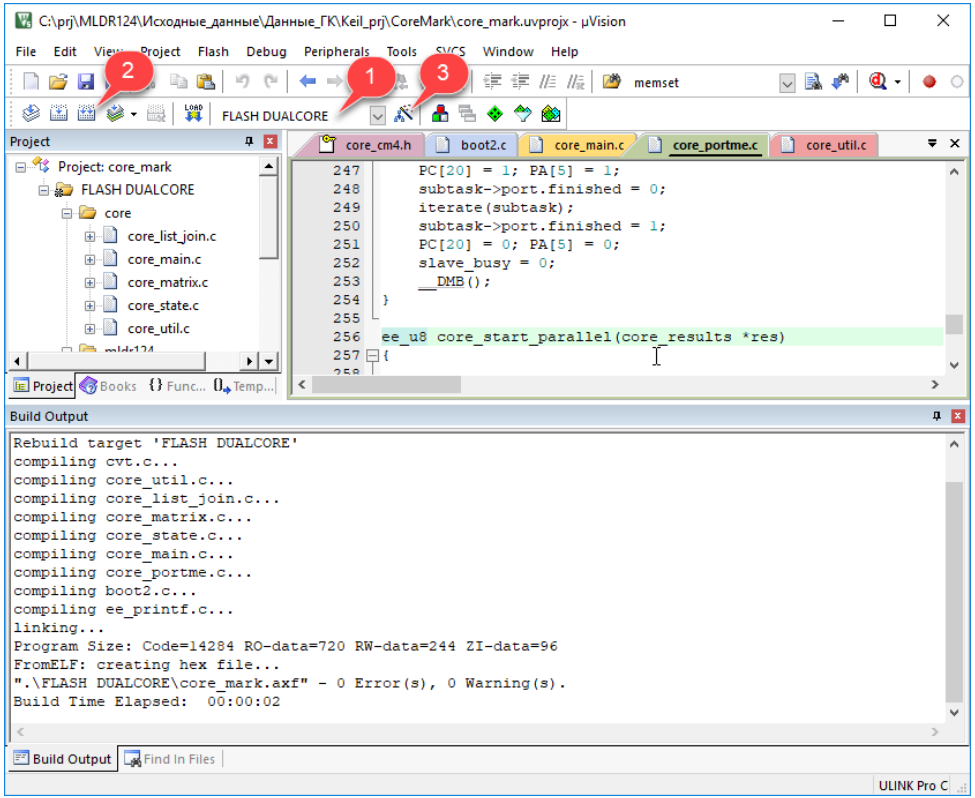
Отладка в Keil uVision

Адаптеры поддерживающие двухъядерную отладку

1. CMSIS-DAP-совместимые

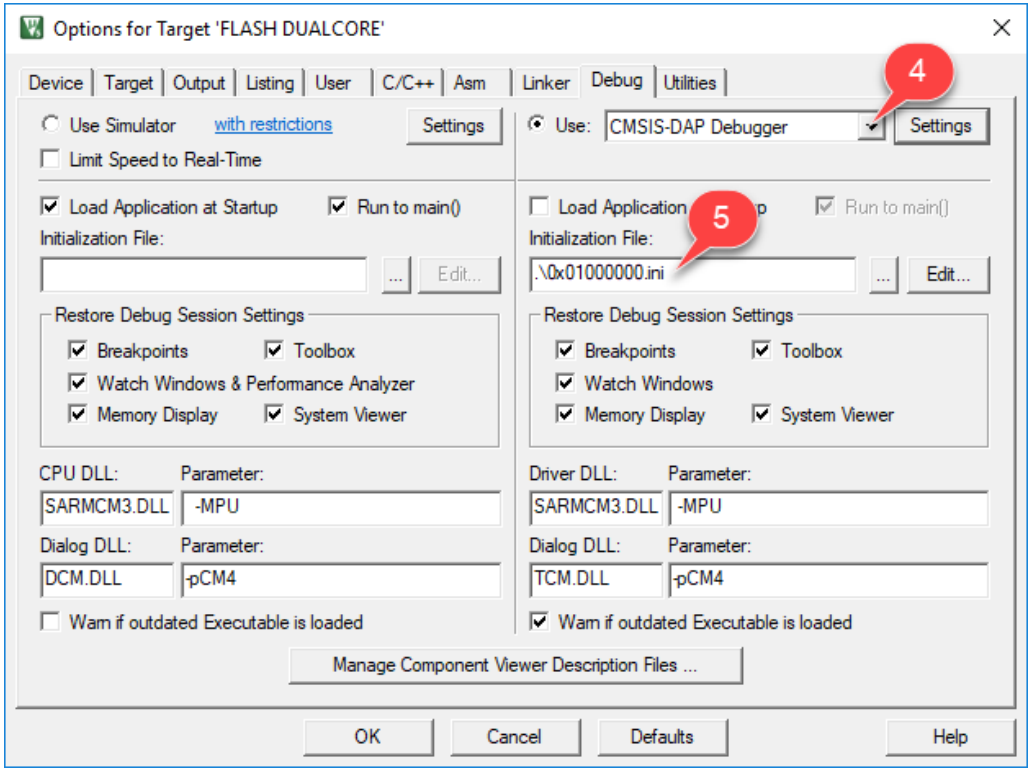
2. Keil ULINK2, ULINKpro

Сборка проекта



1. Выбор конфигурации
2. Сборка
3. Настройка

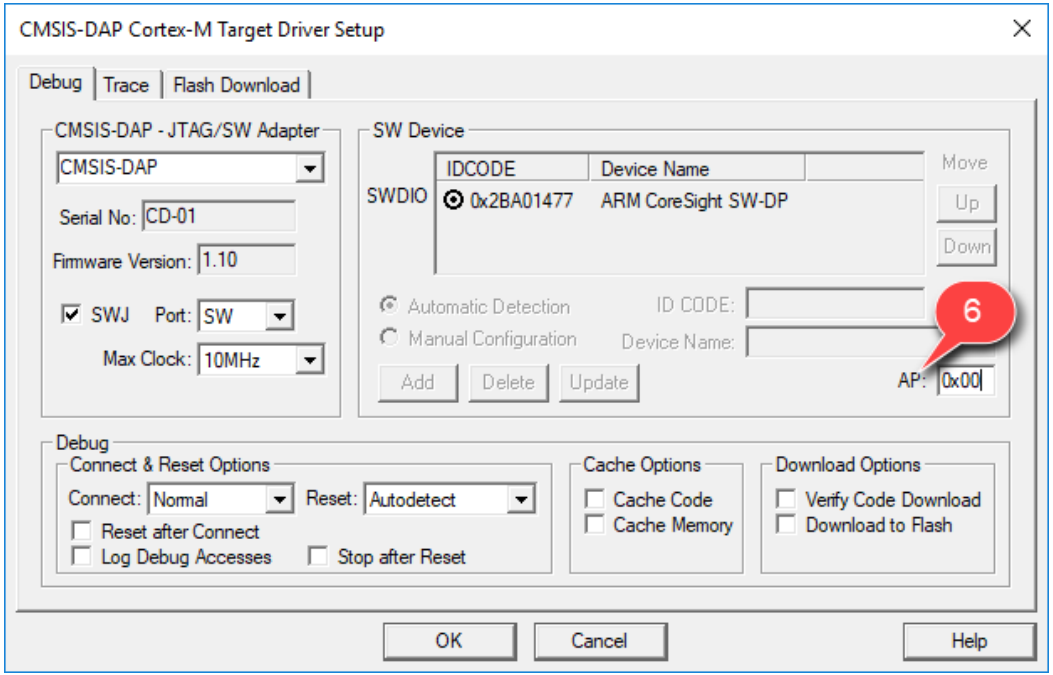
Выбор адаптера



4. Выбор адаптера

5. Файл настроек

Выбор ядра

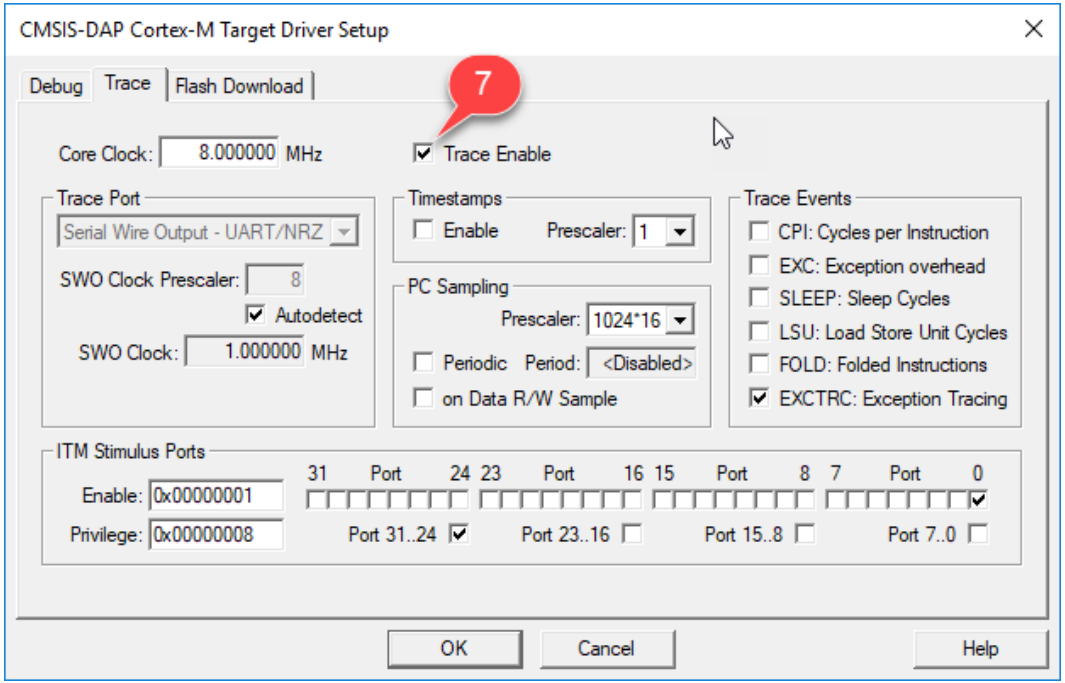


6. Выбор точки доступа:

0x00 – Core0

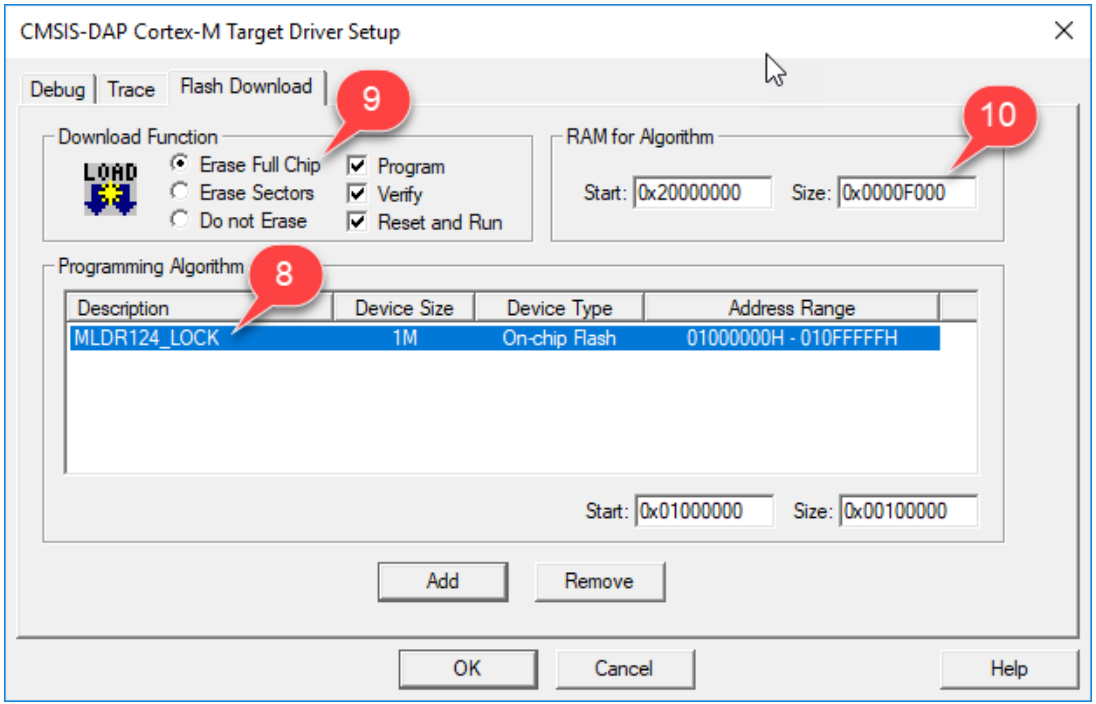
0x01 – Core1

Включение трассировки



7. Включение трассировки и перенаправления вывода в SWO

Настройка загрузки во Flash

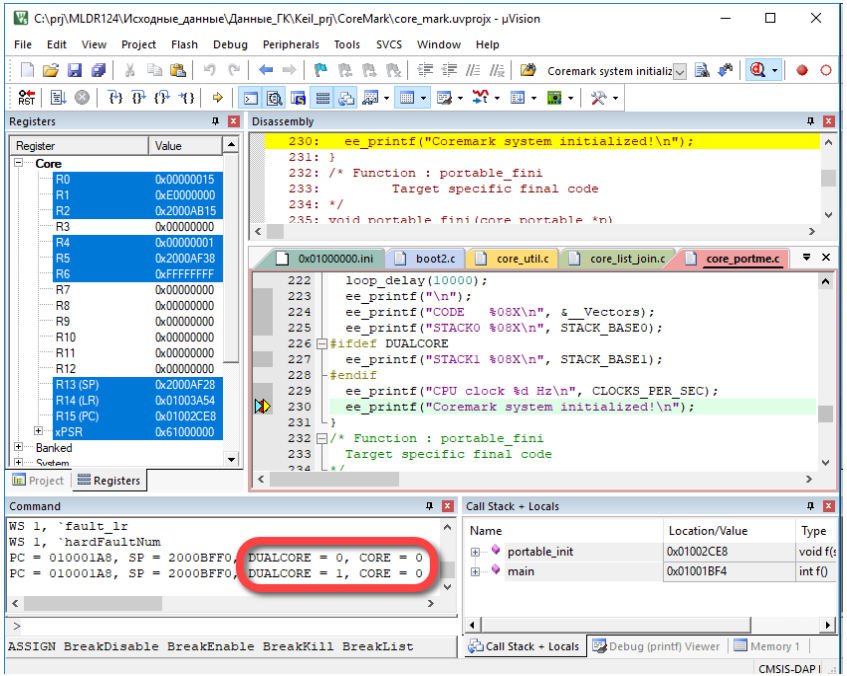


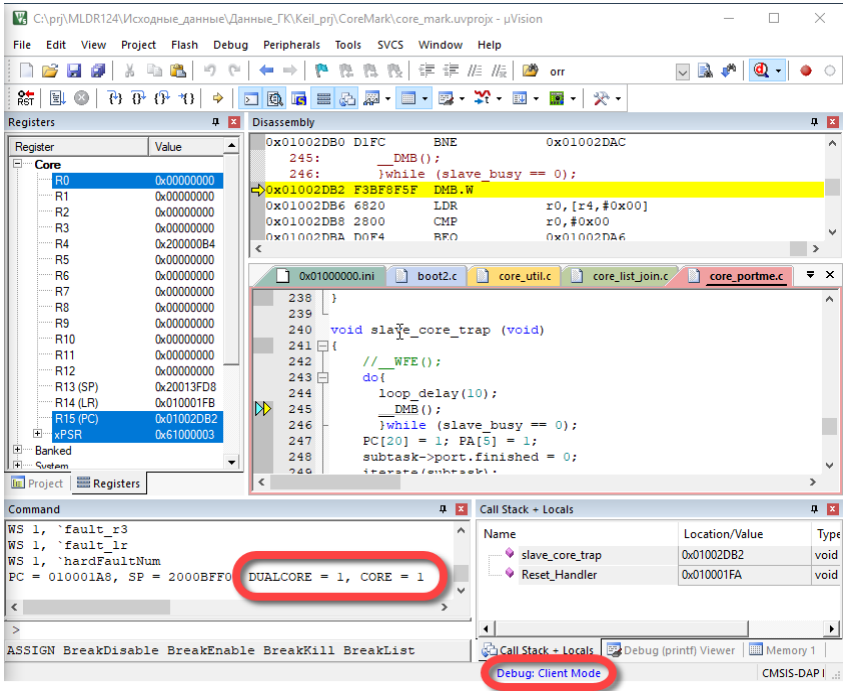
8. Выбор загрузчика Flash

9. Выбор полного стирания

10. Задание доступного объёма памяти для загрузчика

Отладка


C:\prj\MLDR124\Исходные_данные\Данные_ГК\Keil_prj\CoreMark\core_mark.uvprojx - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Register Value
Core
R0 0x00000015
R1 0xE0000000
R2 0x2000AB15
R3 0x00000000
R4 0x00000001
R5 0x2000AF38
R6 0xFFFFFFFF
R7 0x00000000
R8 0x00000000
R9 0x00000000
R10 0x00000000
R11 0x00000000
R12 0x00000000
R13 (SP) 0x2000AF28
R14 (LR) 0x01003A54
R15 (PC) 0x01002CE8
xPSR 0x61000000
Banked
System
Disassembly
230: ee_printf("Coremark system initialized!\n");
231: }
232: /* Function : portable_fini
233: Target specific final code
234: */
235: void portable_fini(core_portable *p)
0x01000000.ini boot2.c core_util.c core_list_join.c core_portme.c
222 loop_delay(10000);
223 ee_printf("\n");
224 ee_printf("CODE %08X\n", &__Vectors);
225 ee_printf("STACK0 %08X\n", STACK_BASE0);
226 #ifdef DUALCORE
227 ee_printf("STACK1 %08X\n", STACK_BASE1);
228 #endif
229 ee_printf("CPU clock %d Hz\n", CLOCKS_PER_SEC);
230 ee_printf("Coremark system initialized!\n");
231 }
232 /* Function : portable_fini
233 Target specific final code
Project Registers
Command
WS 1, `fault_lr
WS 1, `hardFaultNum
PC = 010001A8, SP = 2000BFF0, DUALCORE = 0, CORE = 0
PC = 010001A8, SP = 2000BFF0, DUALCORE = 1, CORE = 0
ASSIGN BreakDisable BreakEnable BreakKill BreakList
Call Stack + Locals
Name Location/Value Type
portable_init 0x01002CE8 void f(
main 0x01001BF4 int f()
Call Stack + Locals Debug (printf) Viewer Memory 1
CMSIS-DAP


C:\prj\MLDR124\Исходные_данные\Данные_ГК\Keil_prj\CoreMark\core_mark.uvprojx - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Register Value
Core
R0 0x00000000
R1 0x00000000
R2 0x00000000
R3 0x00000000
R4 0x200000B4
R5 0x00000000
R6 0x00000000
R7 0x00000000
R8 0x00000000
R9 0x00000000
R10 0x00000000
R11 0x00000000
R12 0x00000000
R13 (SP) 0x20013FD8
R14 (LR) 0x010001FB
R15 (PC) 0x01002DB2
xPSR 0x61000003
Banked
System
Disassembly
0x01002DB0 D1FC BNE 0x01002DAC
245: __DMB();
246: }while (slave_busy == 0);
0x01002DB2 F3BF8F5F DMB.W
0x01002DB6 6820 LDR r0,[r4,#0x00]
0x01002DB8 2800 CMP r0,#0x00
0x01002DBA D0F4 BEQ 0x01002DA6
0x01000000.ini boot2.c core_util.c core_list_join.c core_portme.c
238 }
239
240 void slave_core_trap (void)
241 {
242 //__WFE();
243 do{
244 loop_delay(10);
245 __DMB();
246 }while (slave_busy == 0);
247 PC[20] = 1; PA[5] = 1;
248 subtask->port.finished = 0;
Project Registers
Command
WS 1, `fault_r3
WS 1, `fault_lr
WS 1, `hardFaultNum
PC = 010001A8, SP = 2000BFF0, DUALCORE = 1, CORE = 1
ASSIGN BreakDisable BreakEnable BreakKill BreakList
Call Stack + Locals
Name Location/Value Type
slave_core_trap 0x01002DB2 void
Reset_Handler 0x010001FA void
Call Stack + Locals Debug (printf) Viewer Memory 1
Debug: Client Mode
CMSIS-DAP

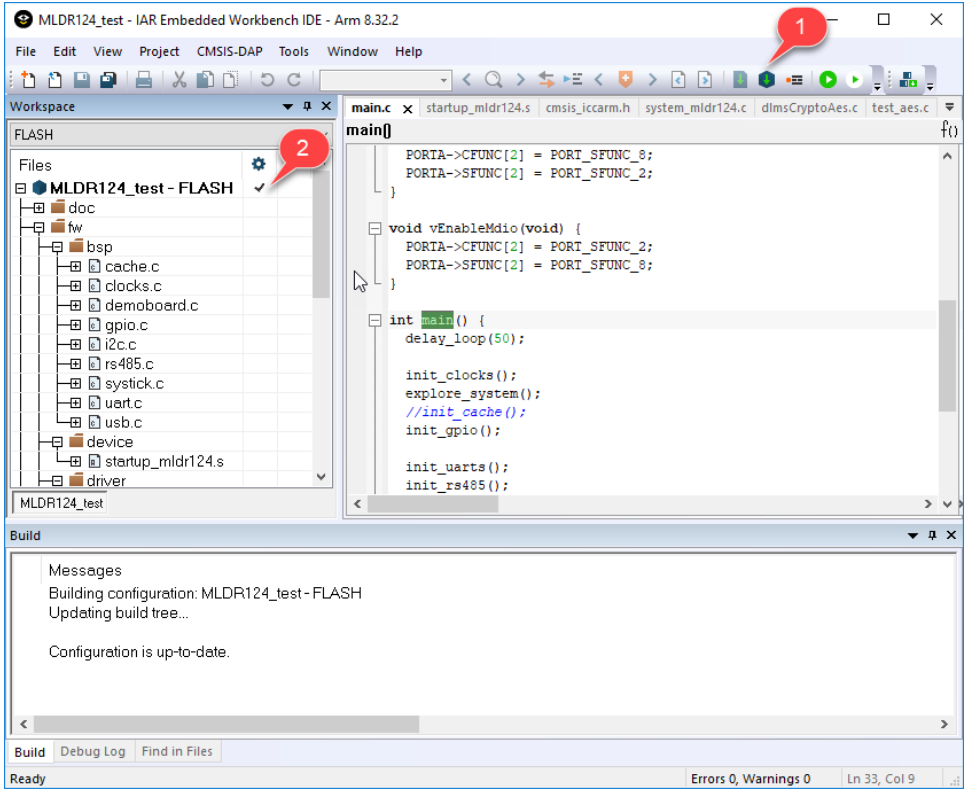
Отладка в Keil uVision

Адаптеры поддерживающие двухъядерную отладку

1. CMSIS-DAP-совместимые

2. IAR Systems I-jet

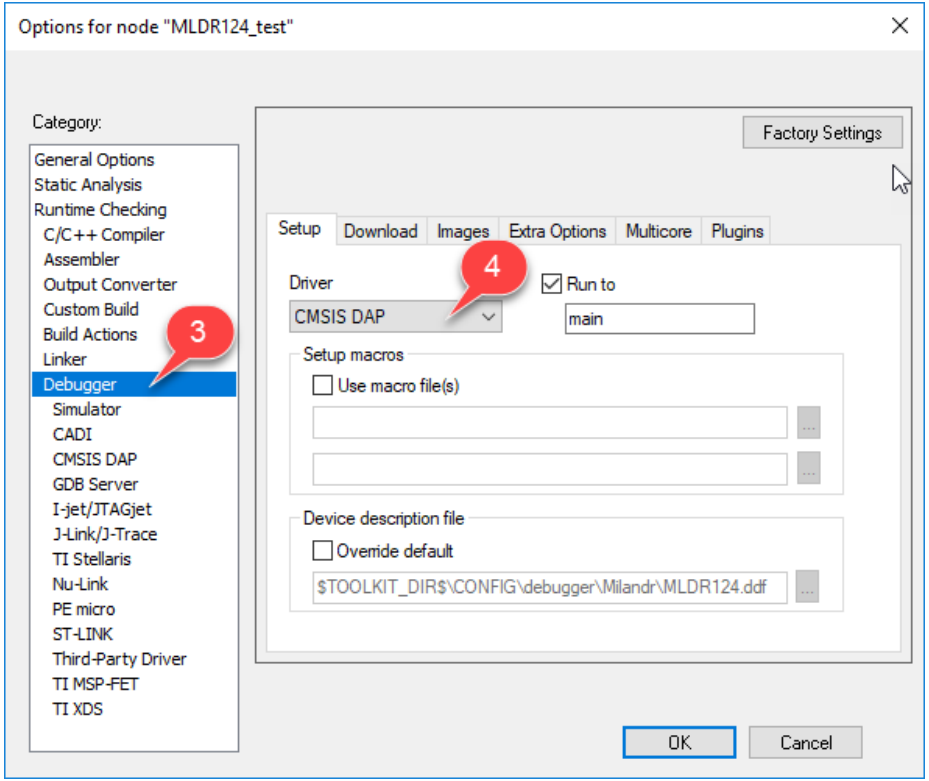
Сборка проекта



1. Сборка

2. Настройка

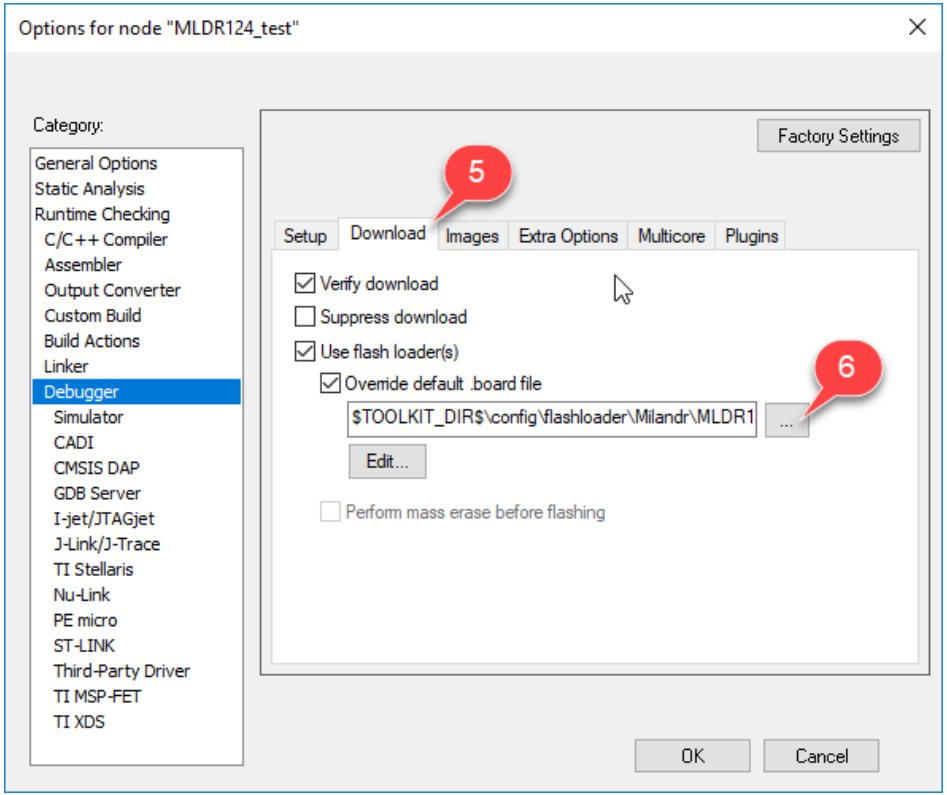
Сборка проекта



3. Раздел отладки

4. Выбор адаптера

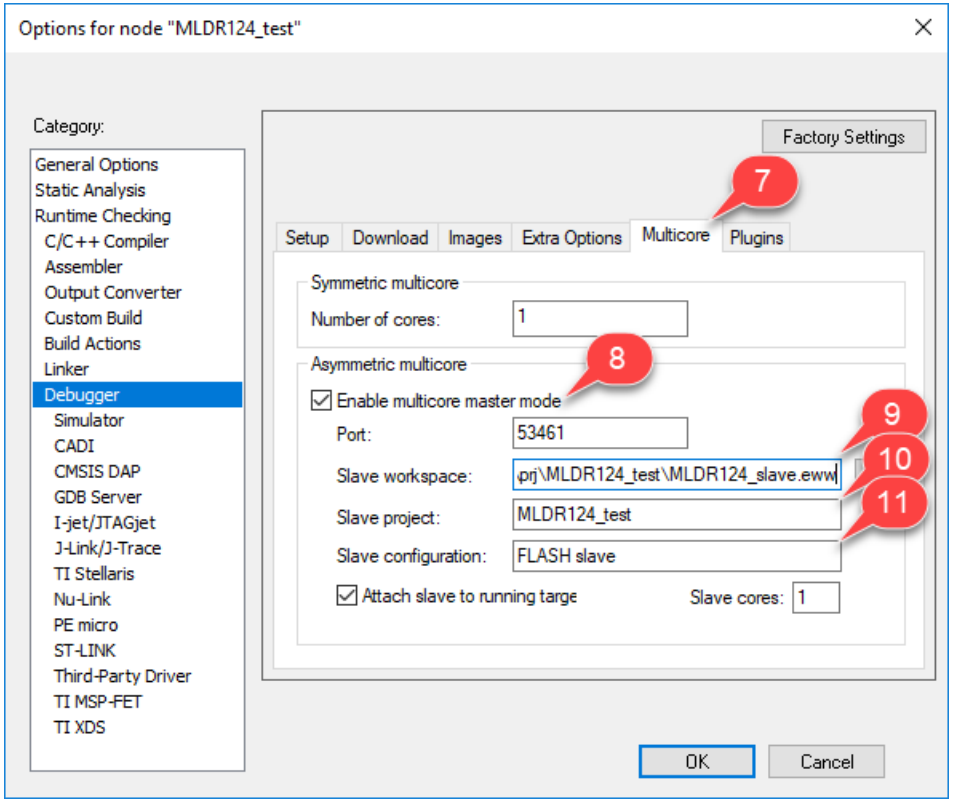
Сборка проекта



5. Закладка настройки загрузки

6. Выбор загрузчика

Настройка многоядерной отладки



7. Закладка настройки многоядерной отладки

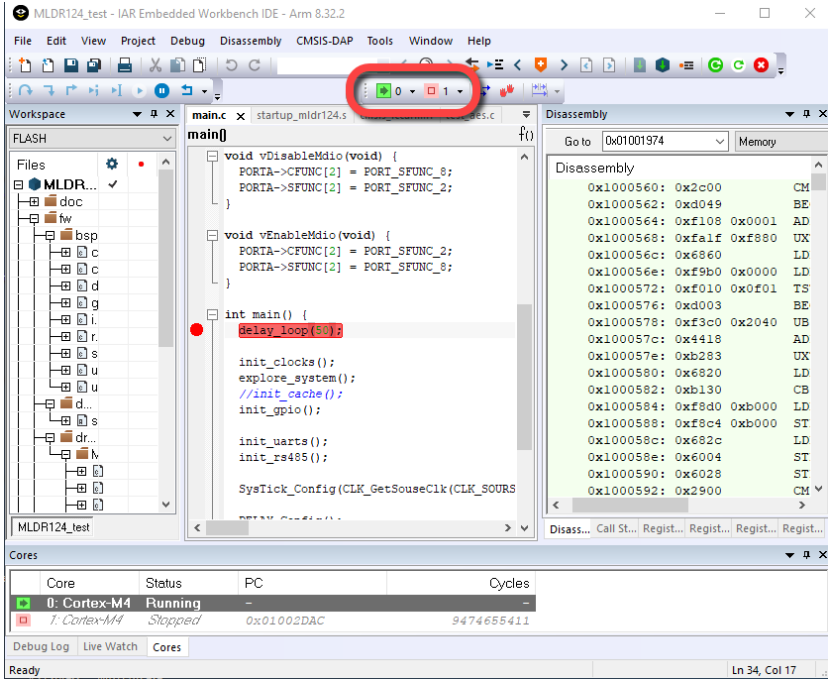
8. Включение режима ведущего

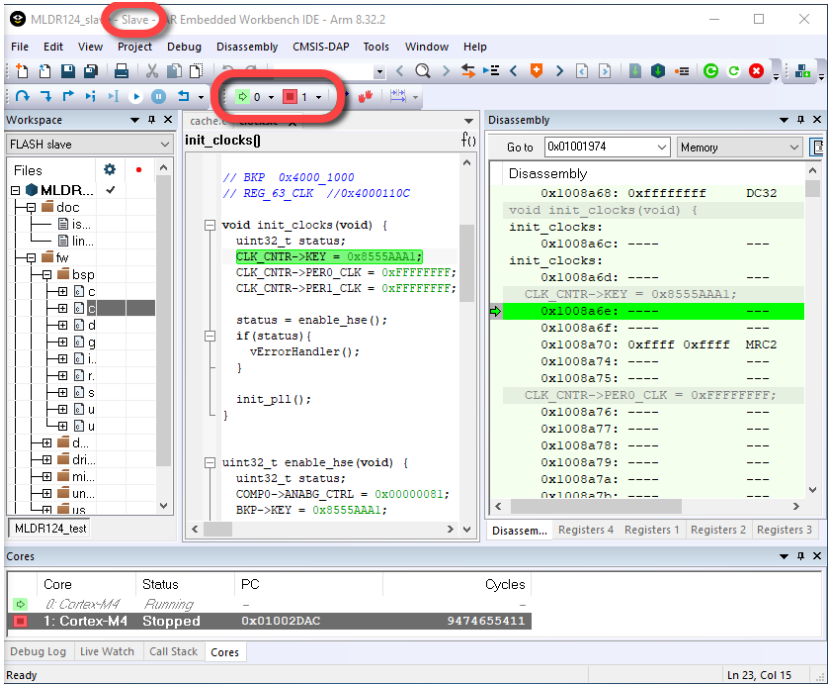
9. Выбор ведомой рабочей области

10. Выбор ведомого проекта

11. Выбор ведомой конфигурации

Отладка


MLDR124_test - IAR Embedded Workbench IDE - Arm 8.32.2
File Edit View Project Debug Disassembly CMSIS-DAP Tools Window Help
Workspace
FLASH
Files
MLDR...
doc
fw
bsp
MLDR124_test
main.c
startup_mldr124.s
main()
void vDisableMdio(void) {
PORTA->CFUNC[2] = PORT_SFUNC_8;
PORTA->SFUNC[2] = PORT_SFUNC_2;
}
void vEnableMdio(void) {
PORTA->CFUNC[2] = PORT_SFUNC_2;
PORTA->SFUNC[2] = PORT_SFUNC_8;
}
int main() {
delay_loop(50);
init_clocks();
explore_system();
//init_cache();
init_gpio();
init_uarts();
init_rs485();
SysTick_Config(CLK_GetSouseClk(CLK_SOURS
Disassembly
Go to 0x01001974
Memory
0x1000560: 0x2c00 CM
0x1000562: 0xd049 BE
0x1000564: 0xf108 0x0001 AD
0x1000568: 0xfa1f 0xf880 UX
0x100056c: 0x6860 LD
0x100056e: 0xf9b0 0x0000 LD
0x1000572: 0xf010 0x0f01 TS
0x1000576: 0xd003 BE
0x1000578: 0xf3c0 0x2040 UB
0x100057c: 0x4418 AD
0x100057e: 0xb283 UX
0x1000580: 0x6820 LD
0x1000582: 0xb130 CB
0x1000584: 0xf8d0 0xb000 LD
0x1000588: 0xf8c4 0xb000 ST
0x100058c: 0x682c LD
0x100058e: 0x6004 ST
0x1000590: 0x6028 ST
0x1000592: 0x2900 CM
Cores
Core Status PC Cycles
0: Cortex-M4 Running – –
1: Cortex-M4 Stopped 0x01002DAC 9474655411
Debug Log Live Watch Cores
Ready
Ln 34, Col 17


MLDR124_slave - IAR Embedded Workbench IDE - Arm 8.32.2
File Edit View Project Debug Disassembly CMSIS-DAP Tools Window Help
Workspace
FLASH slave
Files
MLDR...
doc
fw
bsp
MLDR124_test
init_clocks()
// BKP 0x4000_1000
// REG_63_CLK //0x4000110C
void init_clocks(void) {
uint32_t status;
CLK_CNTR->KEY = 0x8555AAA1;
CLK_CNTR->PER0_CLK = 0xFFFFFFFF;
CLK_CNTR->PER1_CLK = 0xFFFFFFFF;
status = enable_hse();
if(status){
vErrorHandler();
}
init_pll();
}
uint32_t enable_hse(void) {
uint32_t status;
COMP0->ANABG_CTRL = 0x00000081;
BKP->KEY = 0x8555AAA1;
Disassembly
Go to 0x01001974
Memory
0x1008a68: 0xffffffff DC32
void init_clocks(void) {
init_clocks:
0x1008a6c: ---- ---
init_clocks:
0x1008a6d: ---- ---
CLK_CNTR->KEY = 0x8555AAA1;
0x1008a6e: ---- ---
0x1008a6f: ---- ---
0x1008a70: 0xffff 0xffff MRC2
0x1008a74: ---- ---
0x1008a75: ---- ---
CLK_CNTR->PER0_CLK = 0xFFFFFFFF;
0x1008a76: ---- ---
0x1008a77: ---- ---
0x1008a78: ---- ---
0x1008a79: ---- ---
0x1008a7a: ---- ---
Cores
Core Status PC Cycles
0: Cortex-M4 Running – –
1: Cortex-M4 Stopped 0x01002DAC 9474655411
Debug Log Live Watch Call Stack Cores
Ready
Ln 23, Col 15

Переключение режимов

Рекомендуемая процедура переключения из режима LOCKSTEP в режим DUALCORE:

Процедура выполняется при выключенных DMA, кэш и прерываниях.

1. Разблокировать запись в регистры батарейного домена записав ключ в регистр BKP_KEY
2. Установить бит LOCKSTEP_EN регистра REG_60_SYS
3. Очистить конвейер инструкцией DSB
4. В зависимости от выполнения условия CPUID==0 передать управление на разные участки кода, загрузить разные значения в указатели стека.

Рекомендуемая процедура переключения из режима DUALCORE в режим LOCKSTEP:

1. Разблокировать запись в регистры батарейного домена записав ключ в регистр BKP_KEY
2. Сбросить бит LOCKSTEP_EN регистра REG_60_SYS
3. Очистить конвейер инструкцией DSB
4. Выполнить сброс процессора установив бит SYSRESETREQ регистра AIRCR

Синхронизация

С помощью команд поддержки событий **WFE, SEV** и команд эксклюзивного доступа к памяти **LDREX, STREX** могут быть построены следующие примитивы синхронизации:

1. События
2. Мьютексы
3. Критические секции
4. Семафоры

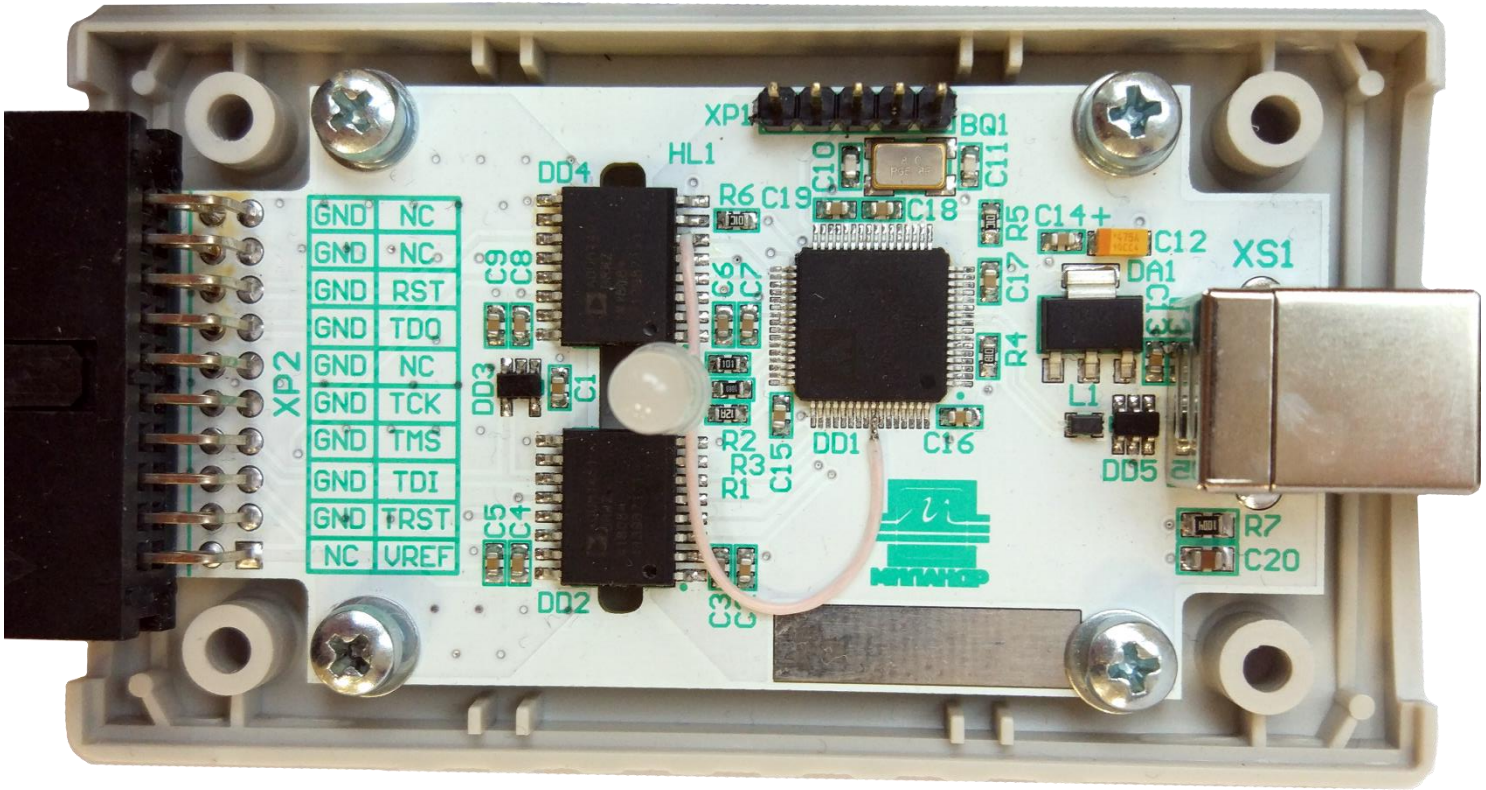
Пример реализации мьютекса

```
locked    EQU 1
unlocked  EQU 0

; lock_mutex
; Declare for use from C as extern void lock_mutex(void * mutex);
    EXPORT lock_mutex
lock_mutex PROC
    LDR     r1, =locked
1   LDREX   r2, [r0]
    CMP     r2, r1          ; Test if mutex is locked or unlocked
    BEQ     %f2             ; If locked - wait for it to be released, from 2
    STREXNE r2, r1, [r0]    ; Not locked, attempt to lock it
    CMPNE   r2, #1          ; Check if Store-Exclusive failed
    BEQ     %b1             ; Failed - retry from 1
    ; Lock acquired
    DMB                     ; Required before accessing protected resource
    BX      lr

2   ; Take appropriate action while waiting for mutex to become unlocked
    WFE                     ; Indicate opportunity to enter low-power state
    B       %b1             ; Retry from 1
    ENDP


; unlock_mutex
; Declare for use from C as extern void unlock_mutex(void * mutex);
    EXPORT unlock_mutex
unlock_mutex PROC
    LDR     r1, =unlocked
    DMB                     ; Required before releasing protected resource
    STR     r1, [r0]        ; Unlock mutex
    DSB                     ; Ensure update has completed before signalling
    SEV                     ; Signal update
    BX      lr
    ENDP
```

Спасибо!

МИЛАНДР

ГРУППА КОМПАНИЙ


XP2
GND NC
GND NC
GND RST
GND TDO
GND NC
GND TCK
GND TMS
GND TDI
GND TRST
NC VREF
XP1
HL1
DD4
DD3
DD2
DD1
DA1
DD5
BQ1
XS1
L1
R1
R2
R3
R4
R5
R6
R7
C1
C4
C5
C6
C7
C8
C9
C10
C11
C12
C14+
C15
C16
C17
C18
C19
C20