

## VÝVOJOVÉ PROSTŘEDÍ FIRMY KEIL ELECTRONIC

Program ověříme ve vývojovém prostředí od firmy KEIL takto:

- 1) Spustíme prostředí označené ikonkou **Keil uVision5\_Verze 37**
- 2) V menu **Projekt** vybereme položku **New Project**
- 3) Zapišeme jméno nového projektu
- 4) Vybereme **CPU**, pro který bude vytvářený SW určen (v našem případě **STMicroelectronics STM32F401**)
- 5) Po otevření okna **Manage Run-Time Environment** zaškneme v položce **Device Startup** pro programy v jazyce C a v položce **CMSIS** položku **CORE**
- 6) V okně projektu se vám objeví adresář **Target 1** a v něm podadresář **Source Group 1**, do kterého vložíme vytvořený soubor v jazyce C. Vložení zajistíme buď kliknutím pravým tlačítkem myši na adresáři **Source Group 1** a výběrem položky **Add Existing Files to Group 'Source Group 1'**. Po rozbalení podadresáře **Source Group 1** se v něm tento soubor objeví.

- 7) V menu Project > Option for Target 'Target 1' se zde v jednotlivých záložkách nastaví parametry projektu. V záložce Device můžeme změnit typ procesoru. V záložce Output > Name of Executable můžeme změnit jméno spustitelného souboru. V záložce Target můžeme nastavit hodinový kmitočet procesoru. Při SW simulaci je vhodné zaškrtnout položku Use microLIB, která zajistí správné chování simulátoru při spuštění programu. Při práci s vývojovým modulem nemá zaškrtnutí položky Use microLIB vliv na chování programu ve vývojové prostředí. V záložce Debug se volíme v levé části Use Simulator pro SW simulaci vytvořeného programu nebo volíme ST-link Debugger při ladění nebo spouštění programu na vývojovém modulu.
- 8) Otevření hotového projektu realizujeme v menu Project > Open Project. **Nestačí otevřít pouze zdrojový soubor !!**
- 9) Zdrojový kód lze upravovat v interním editoru poklepaním na soubor se zdrojovým kódem v okně projektu. Soubor se otevře a můžete jej upravovat. Pozor v položce Open lze otevřít jakýkoliv zdrojový soubor, který lze upravovat, ale **překládat se bude ten co je v projektovém okně.**

- 10) K přeložení projektu můžete využít jak ikony v panelu nástrojů, tak v menu **Project > Build target - zkratková klávesa F7**. V dialogovém okně ve spodní části obrazovky se vypíše hlášení o úspěšnosti překladu. Došlo-li při překladu k chybám, jsou zde uvedeny druhy chyb a čísla řádků, na kterých se vyskytují. Na příslušný řádek je možné se dostat poklepáním na chybové hlášení. Po odstranění všech chyb se v dialogovém okně objeví hlášení 0 Error(s).
- 11) Simulace vytvářeného programu se spouští po úspěšném překladu z menu **Debug > Start/Stop Debug Session zkratková klávesa CTRL+F5** nebo ikonou. Po potvrzení hlášení o Evaluation Mode verzi je simulátor připraven ke spuštění. V levé části se objeví okno s výpisem registrů a otevřou se případná další okna s výpisem stavu periférií a paměti, pokud si je vyberete menu **Pheripherals** nebo **View**. Simulace probíhá po **Debug > Run klávesová zkratka F5** buď trvale nebo do bodu zastavení **BreakPoints** (vkládání nebo zrušení dvou klikem na tmavě šedivé místo před požadovaným řádkem červená značka viz.další blána).

- 12) Krokování po jednotlivých instrukcích nebo řádcích jazyka C **klávesa F11**
- 13) Krokování s rychlým vykonáním podprogramů **klávesa F10**
- 14) Aktuálně prováděný řádek je znázorněn **žlutou šipkou**. Části kódu, které byly během simulace alespoň jednou vykonány, jsou označeny na začátku řádku zeleně.
- 15) Hlášení **error 65: access violation at 0x40023800: no 'read' permission** odstraníme zápisem **MAP 0x40020000,0x40023FFC READ WRITE** do příkazového řádku v daném běhu debuggeru. Abychom nemuseli řádek zapisovat po každém spuštění Debuggeru uložíme řádek do souboru **Dbg\_RAM.ini**.

# VÝVOJOVÉ PROSTŘEDÍ FIRMY KEIL ELECTRONIC

The screenshot displays the Keil uVision IDE interface for a project named "Blinky\_1". The main window shows the source code for "LED\_NUCLEO\_L152RE.c", which implements a simple blinky LED program. The code includes a while loop that reads a button state, turns an LED on or off, and introduces delays.

The Logic Analyzer window shows a digital signal trace for the "output" pin. The signal is a square wave, indicating the LED is being toggled. The time scale is set to 0.5 s, and the signal is shown over a period of approximately 6.5 ms.

The Watch window shows the current values of the variables "output" and "btns". The "output" variable is currently 0, and "btns" is 1. The status bar at the bottom indicates the simulation time is 4.76943294 seconds.

Register	Value
R0	0x00000000
R1	0x00000005
R2	0x40020000
R3	0x00000000
R4	0x00000005
R5	0x00000001
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x20000428
R14 (LR)	0x08000349
R15 (PC)	0x08000348
xPSR	0x41000000

Property	Value
MODER8	0x00
MODER7	0x00
MODER6	0x00
MODER5	0x01
MODER4	0x00
MODER3	0x00
MODER2	0x00
MODER1	0x00
MODER0	0x00
OTYPER	0
OSPEEDER	0x00000400
PUPDR	0
IDR	0
ODR	0
BSRR	0
LCKR	0

Name	Value	Type
output	0x00000000	unsign...
btns	0x00000001	int

Simulation t1: 4.76943294 sec L:87 C:1 CAP\_NUM SCRL OVR R/W

# VÝVOJOVÉ PROSTŘEDÍ FIRMY KEIL ELECTRONIC

The screenshot displays the Keil uVision IDE interface for a project named "Blinky\_1". The main window shows the source code for "LED\_NUCLEO\_L152RE.c", which implements a button-toggling program with delays. The code is as follows:

```

82  Buttons_Initialize();
83
84  while(1)
85  {
86      btns = Buttons_GetState();
87      LED_On (num);
88      output=1;
89      if (btns==1) Delay(20); else Delay(5); // Zpozdeni
90      LED_Off(num);
91      output=0;
92      Delay(10);
93  }
94
95

```

The Disassembly window shows the corresponding assembly instructions:

```

0x08000344 F7FFFF1F BL.W LED_On (0x08000186)
87:          output=1;
0x08000348 2001      MOVS     r0,#0x01
0x0800034A 490F      LDR     r1,[pc,#60] ; @0x08000388
0x0800034C 6008      STR     r0,[r1,#0x00]
88:          if (btns==1) Delay(20); else Delay(5); // Zpozdeni
0x0800034E 2D01      CMP     r5,#0x01
0x08000350 D103      BNE     0x0800035A

```

The Registers window shows the state of the Core registers, with R15 (PC) at 0x08000348. The GPIOA register view shows the MODER register at 0x40020000 with various bits set to 0 or 1. The Watch window shows the current values of the 'output' (0) and 'btns' (0) variables.

Zobrazení průběhu hodnoty bitu, char, int nebo analogové hodnoty realizuje **Logický analyzátor**, který je funkční pouze při programové simulaci. **Zobrazovaná hodnota musí být globální**, lokální proměnné lze zobrazovat pouze v okně **Watch**. Veličina nebo její část může být v logickém analyzátoru omezena pomocí logického součinu příslušnou maskou vybírající bit nebo část veličiny.

V pravém dolním rohu je zobrazována doba, která uplynula do příslušného bodu zastavení nebo po každém kroku (**F11** nebo **F10**) v okně pro zdrojový program (jazyk C) nebo v okně **Disassembly**. V horním okně je zobrazen překlad zdrojového kódu do instrukcí assembleru. Řádek začíná adresou, kde je uložena instrukce, za ní následuje **operační kód**, případně ještě potřebná data a nakonec je uvedeno mnemonické označení instrukce v **jazyce symbolických adres** (assembleru).

## PROMĚNNÉ V JAZYCE C PRO ARM V KEIL MDK 5

Datový typ	Bitů	Bytů	Rozsah
char (unsigned char)	8	1	0 ÷ 255 (zarovnaný byte)
signed char	8	1	-128 ÷ 127 (zarovnaný byte)
(signed) short	16	2	-32768 ÷ 32767 (zarovnaná polovina slova)
unsigned short	16	2	0 ÷ 65535 (zarovnaná polovina slova)
(signed) int	32	4	-2,147,483,648 ÷ 2,147,483,647
unsigned int	32	4	0 ÷ 4,294,967,295
signed long	32	4	-2,147,483,648 ÷ 2,147,483,647
unsigned long	32	2	0 ÷ 4,294,967,295
signed long long 64	64	8	-9,223,372,036,854,775,808 ÷ 9,223,372,036,854,775,807
unsigned long long 64	64	8	0 ÷ 18,446,744,073,709,551,615
float	32	4	±1.175494E-38 ÷ ±3.402823E+38
double 64 long double 64	64	8	±2.22507385850720138e-308 ÷ ±1.79769313486231571e+308
wchar_t	16	2	0 ÷ 65535
wchar_t	32	4	0 ÷ 4,294,967,295 je-li kompilován s -wchar32