**Keil新数据库添加方法**

**几句话总结：**

**创建新元件库：**

1、备份且删除 Keil\UV4 \UV4.cdb文件；

2、打开keil程序，打开file->device database;

3、完善需要填写的内容，点击add按钮；

4、Keil\UV4 文件夹下重新出现UV4.cdb文件，改名为new.cdb。且还原原始UV4.cdb文件

5、修改keil\ TOOLS.INI文件，在[UV2]下面添加CDB0=UV4\new.cdb("new")，保存关闭；

6、重启keil，驱动库里会多了一个新的库。

**创建flash支持：**

1、Keil\ARM\Flash中新建文件夹；

2、拷贝\_Template模板工程到新建的文件夹；

3、修改输出文件为：new， 完善该工程内的函数接口。

4、编译，在Keil\ARM\Flash文件夹下会出现新的new.flm文件，该文件即为flash支持文件。下载选项处会自动显示该flash。

注意：这样编写三类函数：擦除，编程，校验，分别是三个独立的过程，**所以**在擦除时候定义的变量不能在编程时候使用。

并且全片擦除函数和sector擦除函数会被响应的选择只调用一次，**所以**sector擦除的实现可以和全片擦除的方法一样

**在新建的原件库中添加自动flash支持：**

file->device database中的options中添加(选择仿真器等内容)：

FLDLL=UL2CM3(-O4303 -S0 -C0 -FO7 -FD10000000 -FC800 -FN1 **-FF0new** -FS00 -FL08000)

注意：**UL2CM3：**指的是使用ulink2，使用jlink的话，改成：JL2CM3

**-FD10000000 -FC800：**RAM for algorith，运行\*.flm程序的RAM的开始地址和大小

**-FF0new：**选择的\*,flm文件名，位置为Keil\ARM\Flash\\*.flm，且第三个字母必须是0

**-FS00 -FL08000：**programming algorithm，flash 的开始地址和大小，且第三个字母必须是0

**file->device database中的options例子：**

CPU=IRAM(0x10000000-0x1000FFFF) IROM(0x30000000-0x3000FFFF) CLOCK(16000000) CPUTYPE("Cortex-M0")

MON=SARMCM3.DLL("") TARMCM1.DLL("-pCM0")

SIM=SARMCM3.DLL("") DARMCM1.DLL("-pCM0")

FLDLL=UL2CM3(-O4303 -S0 -C0 -FO7 -FD10000000 -FC800 -FN1 -FF0QN9020\_DEVICE -FS00 -FL08000)

REGFILE=LPC11xx.h("NXP\LPC11xx")

SFILE="STARTUP\NXP\LPC11xx\startup\_LPC11xx.s" ("NXP LPC11xx Startup Code")

注意：REGFILE=LPC11xx.h("NXP\LPC11xx")可支持用户工程内右键添加。该文件位置为：**keil\ARM\INC\**NXP\LPC11xx

**详细：**

**1、ADDING A USER-SPECIFIC DEVICE DATABASE (CDB FILE)**

Additional Device Databases must be defined manually in the file C:\KEIL\TOOLS.INI. For example, you can create a device database that contains all SmartCards. To do so, in TOOLS.INI, under the section [UV2], create a CDB*x* entry as shown below.

[UV2]

CDB0=*path*\PCARDS.CDB("Our Smart Cards")

This allows accessing the Device Database file PCARDS.CDB stored in the folder C:\KEIL\*path* (*path* is relative to the µVision installation folder). The Device Database name "Our Smart Cards" is displayed in dialogs where the Device Database can be selected. For example, in the dialog Options for Target - Device and File - Device Database.

**ADDING CUSTOM PARTS TO THE DEVICE DATABASE**

SYMPTOM

The µVision device database contains all 8051, 251, C16x/XC16x/ST10, and ARM7/ARM9/Cortex-M standard products. However, there are some custom devices and there will be future devices that are not part of this database (they will be added as they are created, of course). If you need to work with an unlisted MCU, you have two alternatives:

Select a generic device.

Add a new device to the database.

SELECTING A GENERIC DEVICE

Under the Generic listing in the device database, for 8051, 251, C16x/XC16x/ST10 you will find the following generic devices:

8031 (all Variants)、8032 (all Variants)、8051 (all Variants)、8052 (all Variants)

C166 (all Variants) - Supports CPUs with no extended instruction set

C167 (all Variants) - Supports CPUs with an extended instruction set

For ARM and Cortex devices, under the ARM listing in the device database, you will find the following generic devices:

ARM7 (Big Endian)、ARM7 (Little Endian)、ARM966E-S (Big Endian)、ARM966E-S (Little Endian)、ARM9E-S (Big Endian)、ARM9E-S (Little Endian)、Cortex-M0、Cortex-M0 SDK、Cortex-M1、Cortex-M1 (Altera)、Cortex-M3、Cortex-M4、Cortex-M4 FPU、Cortex-R4、Cortex-R4F

You may select one of these devices and then specify any necessary chip options in the Target Dialog.

ADDING A NEW DEVICE

From the File menu, select the Device Database item. µVision opens a dialog which shows the device database where you may add and modify the existing devices.

To add a new device...

Select a CPU that is similar to the device you want to use. Be sure to double-click to bring the settings for this device into focus.

Change the name of the chip vendor.

Select the appropriate CPU family.

Enter the part number in the device text box.

Modify the description to match your device.

Change the options for the tools (see below).

Single-click in the database window (to change focus). This enables the Add button.

Click Add to add the new device.

To remove a device...

Locate the device you want to remove in the Data Base tree list box and left-click the name once to highlight it.

Press the Del key to remove the device from the Device Database.

Note

You will not receive a warning when you press the Del key. The device is deleted immediately.

**Device Database Parameters**

[Home](http://www.keil.com/support/man/docs/uv4/default.htm) » [Appendix](http://www.keil.com/support/man/docs/uv4/uv4_appendix.htm) » C. Device Database Parameters

Device Database Parameters specify default device characteristics and define default settings for steering the µVision IDE & Debugger. Use these parameters to [customize or add a device](http://www.keil.com/support/man/docs/uv4/uv4_ca_cpu_ndatabase.htm) to µVision. The parameters are entered into the field File - Device

Adjusting the Options...

In the Options box, CPU= specifies the basic tool settings for the chip. The parameters for CPU= are:

Database - Options.

|  |  |
| --- | --- |
| Parameter | Description |
| BOOKn=*document\_file*(*title*) | Defines books, manuals, device datasheets that are added to the[Books Window](http://www.keil.com/support/man/docs/uv4/uv4_ui_bookstab.htm); (n={0,1,2, ...}). |
| CPU=*CPU Options* |  |
| FLASH=*command*(*args*) | Specifies an external Flash programming utility. This option corresponds to Use External Tool for Flash Programming as described in [Flash Download Configuration](http://www.keil.com/support/man/docs/uv4/uv4_fl_usingflashmenu.htm). |
| REGFILE=*header\_file*(*folder*) | Specifies the header file with device specific definitions. *folder* is relative to ..\*toolchain*\INC. |
| SFILE=*startup\_file*(*folder*) | Specifies the startup-file containing the startup code is copied to the project folder when creating a new project. Startup files are located in the folder ..\ARM\Startup. Refer to [Create Project File and Select Device](http://www.keil.com/support/man/docs/uv4/uv4_ca_createprjfile.htm). |
| SVD=*rel\_path*\*file\_name*.sfr | Specifies the Special Function Register file (\*.sfr) containing the peripheral descriptions. This file controls the [System Viewer](http://www.keil.com/support/man/docs/uv4/uv4_db_dbg_systemviewer.htm).*rel\_path* is relative to ..\Keil\ARM. |

Basic device settings are encoded in the parameter CPU as listed in the table below.

|  |  |
| --- | --- |
| CPU Options | Description |
| IRAM (*range*) | Address location of on-chip IRAM. |
| IRAM2 (*range*) | Address location of a 2nd on-chip IRAM. |
| XRAM (*range*) | Address location of an external RAM. |
| XRAM2 (*range*) | Address location of a 2nd external RAM. |
| XRAM3 (*range*) | Address location of a 3rd external RAM. |
| IROM (*range*) | Address location of the on-chip (flash) ROM. |
| IROM2 (*range*) | Address location of a 2nd on-chip (flash) ROM. |
| XROM (*range*) | Address location of an external ROM. |
| XROM2 (*range*) | Address location of a 2nd external ROM. |
| XROM3 (*range*) | Address location of a 3rd external ROM. |
| ICAN (*range*) | Address location of the on-chip CAN module. Used for C167 and variants only. |
| CPUTYPE (*variant*) | Specify one of the following core variants for ARM powered microcontrollers  Cortex-M0  Cortex-M1  Cortex-M3  Cortex-M4  Cortex-R4  Cortex-R4F  Cortex-R4 with floating point unit.  ARM7TDMI  ARM926EJ-S  ARM966E-S  ARM9E |
| EBIG | Default to BIG endian for ARM based controllers. |
| ELITTLE> | Default to LITTLE endian for ARM based controllers. |
| ESEL | Allow selection of the endianess for ARM based controllers. |
| CLOCK (*val*) | Default CPU clock. |
| FPU | Specifies default VFPU usage for some ARM9 devices, possible options: none, lib, ANSI, fast. |
| FPU2 | Specifies FPU for Cortex-M4. |
| MASK\_REV(*val*) | Specify the mask revision number. |
| MDU\_F120 | Use the Multiply/Accumulate Unit of SiliconLabs C8051F12x device variants. |
| MDU\_R515 | Use the Multiply/Divide Unit of Cast/Evatronix R80515. |
| MOD167 | Use the instruction set extensions of the C16x device variants. |
| MOD517DP | Enable Infineon  8051 specific multiple DPTR registers. |
| MOD517AU | Enable the Infineon 8051 specific Arithmetic Unit. |
| MODA2 | Enable Atmel specific multiple DPTR registers (like on AT89S8252). |
| MODAB2 | Enable Analog Devices specific multiple DPTR registers. |
| MODC2 | Enable Cast/Evatronix specific multiple DPTR registers (R80515). |
| MOD\_CONT | Enable support for the Dallas Contiguous Mode. |
| MODDA | Enable Dallas specific Arithmetic Accelerator. |
| MODDP2 | Enable Dallas specific multiple DPTR registers. |
| MODH2 | Enable Hynix/ST uPSD33xx uPSD34xx multiple DPTR registers. |
| MODP2 | Enable NXP specific multiple DPTR registers. (Note also some Atmel devices are using this variant). |
| MODV2 | Use the instruction set extensions of the C16x V2 architecture. |
| MX | Enable support for the NXP 80C51MX architecture. |
| MXP | Enable support for the NXP SmartMX SmartCard architecture. |
| DPX | Enable 24-bit DPTR register for the Analog Devices ADuC812. |
| PMW | Enable the PCON.PMW feature that allows to use MOVX instructions to write into code space for the Evatronix R8051XC. |
| DPC | Enable the data pointer control registers on the Evatronix R8051XC that provide auto-increment features for the DPTR registers. |
| BSE | Enable the Bank Switch Enable feature in the register DPSEL.3 for the Evatronix R8051XC. |
| PSOC | Enable the generation of interrupt vectors for Cypress PSoC. |

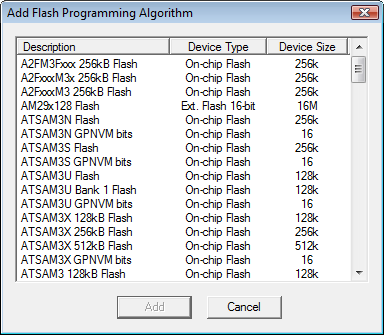
http://www.keil.com/support/man/docs/images/note.gif Note

Other parameters, not explained in here, specify µVision debugging DLLs. Leave these parameters unchanged when adding a new device.

Refer to the knowledgebase article [µVision: Adding a user-specific Device Database (CDB File)](http://www.keil.com/support/docs/1645.htm).

**2、Creating New Flash programming Algorithms**

[Home](http://www.keil.com/support/man/docs/ulink2/default.htm) » [Setup Flash Programming](http://www.keil.com/support/man/docs/ulink2/ulink2_su_flash_prg.htm) » Creating New Algorithms



µVision allows creating Flash programming algorithms for unsupported devices. The algorithm source code is implemented as a µVision project with special compiler and linker settings.

Follow these steps to create a new Flash programming algorithm:

Create a sub-folder in \KEIL\ARM\FLASH.

Copy the content from \KEIL\ARM\FLASH\\_Template to this new folder.

Rename the project file NewDevice.uvproj to represent the new Flash ROM device name, for example 29F400.uvproj.

Open the project with µVision. From the toolbar, use the drop-down Select Target to define the processor architecture:

Cortex-M fits for Cortex-M0, M1, M3, and M4 processor-based devices.

ARM7/ARM9 fits for ARM7 and ARM9 processor-based devices.

The configuration assumes a little-endian microcontroller. In case of a big-endian microcontroller, select the correct processor core with Project - Options for Target - Device.

Open the dialog Project - Options for Target - Output and change the content of the field Name of Executable to represent the device, for example 29F400.

Adapt the programming algorithms in the file FlashPrg.c (see below).

Adapt the device parameters in the file FlashDev.c (see below).

Use Project - Build Target to generate the new Flash programming algorithm. The output file,\*.FLX - for ARM7/ARM9 devices, or \*.FLM - for Cortex-M devices, is copied to the folder\KEIL\ARM\FLASH and is now available in the dialog [Add Flash Programming Algorithm](http://www.keil.com/support/man/docs/ulink2/ulink2_su_suppliedalgorithms.htm).

http://www.keil.com/support/man/docs/images/note.gif Note

Flash programming algorithms use Read-Only Position Independent and Read-Write Position Independent program code. These options are set in the dialogs Project - Options for Target - C/C++ and Project - Options for Target - Asm.

The dialog Project - Options for Target - Linker defines the linker scatter file Target.lin. The error L6305 is disabled with --diag\_suppress L6305.

**FlashPrg.c**

The file FlashPrg.c contains the mandatory Flash [Algorithm Functions](http://www.keil.com/support/man/docs/ulink2/ulink2_alg_func.htm) Init, UnInit, EraseSector, andProgramPage. Optionally, and in dependency of the device features, or to speed-up execution, the functions EraseChip, BlankCheck, and Verify can be implemented.

Use these functions to create Flash Programming Algorithms:

| Function Name | Indication | Description |
| --- | --- | --- |
| [BlankCheck](http://www.keil.com/support/man/docs/ulink2/ulink2_blankcheck.htm) | optional | Check and compare patterns. |
| [EraseChip](http://www.keil.com/support/man/docs/ulink2/ulink2_erasechip.htm) | optional | Delete entire Flash memory content. |
| [EraseSector](http://www.keil.com/support/man/docs/ulink2/ulink2_erasesector.htm) | mandatory | Delete Flash memory content of a specific sector. |
| [Init](http://www.keil.com/support/man/docs/ulink2/ulink2_init.htm) | mandatory | Initialize and prepare device for Flash programming.  It is invoked whenever an attempt is made to download the program to Flash. |
| [ProgramPage](http://www.keil.com/support/man/docs/ulink2/ulink2_programpage.htm) | mandatory | Write the application into the Flash memory. |
| [UnInit](http://www.keil.com/support/man/docs/ulink2/ulink2_uninit.htm) | mandatory | De-initialize the microcontroller after one of the Flash programming steps.  It is invoked at the end of an erasing, programming, or verifying step. |
| [Verify](http://www.keil.com/support/man/docs/ulink2/ulink2_verify.htm) | optional | Compare Flash memory content with the program code. |

**FlashDev.c**

The file FlashDev.c contains parameter definitions for:

the mandatory Flash Programming Functions Init, UnInit, EraseSector, and ProgramPage. Depending on the device, the optional Flash Programming Functions EraseChip, BlankCheck, andVerify might need programming.

the FlashDevice structure.

struct FlashDevice const FlashDevice = {

FLASH\_DRV\_VERS, // Driver Version, do not modify!

"STM32Fxxx High-density Flash",// Device Name (512kB/384kB/256kB)

ONCHIP, // Device Type

0x08000000, // Device Start Address

0x00080000, // Device Size in Bytes (512kB)

1024, // Programming Page Size

0, // Reserved, must be 0

0xFF, // Initial Content of Erased Memory

100, // Program Page Timeout 100 mSec

500, // Erase Sector Timeout 500 mSec

// Specify Size and Address of Sectors

0x0800, 0x000000, // Sector Size 2kB (256 Sectors)

SECTOR\_END

};

Testing Algorithms

The \KEIL\ARM\FLASH\\_Template\Test\ folder contains a project that shows how to test a new Flash Programming Algorithm on behalf of an STM32 device.