

ZT0002 专题教程

作者：Eric2013

uC/Probe 简易使用说明

销售QQ: 1295744630

销售旺旺：armfly

微信公众号：安富莱电子

销售电话：13638617262

邮箱：armfly@qq.com

公司网址：www.armfly.com

技术支持论坛：bbs.armfly.com

淘宝直销：armfly.taobao.com



武汉安富莱电子有限公司

专业开发板、显示模块制造商

承接项目开发（提供生产供货服务）

本次专题教程主要给大家讲解 uC/Probe 的使用方法,在实际工程调试时,这个软件还是非常实用的。如果你项目中用的 uCOS,建议掌握此软件的使用方法,将给你的工程调试带来事半功倍的效果。

- 1.1 重要提示 (必读)
- 1.2 uC/Probe 简介
- 1.3 MDK 使用方法
- 1.4 IAR 使用方法
- 1.5 配套例子
- 1.6 官方手册
- 1.7 总结

1.1 重要提示 (必读)

- ◆ 当前教程中,我们使用的是教育版,这个版本可以免费使用,但部分功能有限制。安装并打开这个软件后,可以看到右上角的标识:



- ◆ 教程采用的是 JLINK 实现开发板与 uC/Probe 的连接通信。
- ◆ 教程里使用的 uC/Probe 版本是 4.2.1。论坛下载:<http://bbs.armfly.com/read.php?tid=31814>。
- ◆ 实际测试发现,该软件有时候容易死机(系统 WIN7 64bit),解决办法是测试的功能一次性不要太多,可以单独测试 uCOS 组件的信息或者其它手动添加的信息,最好不要同时测试。
- ◆ 测试发现 uC/Probe 版本是 4.2.1 时无法加载 uCOS-II 调试组件,而之前的 3.6.15 版本却可以,所以专门备份了一个 3.6.15 版本,论坛下载地址:<http://bbs.armfly.com/read.php?tid=48537>。由于这个问题,我们教程中不对 uCOS-II 的使用做介绍了,因为跟教程中讲解的 uCOS-III 使用是相同的。
- ◆ 一定要保证开发板中下载的程序跟 uC/Probe 加载的可执行文件是同一个,否则 uC/Probe 会通讯异常,从而死机,这个务必要注意。
- ◆ 对于本专题配套的例子,使用 MDK4.7X 以及 MDK5 均可,另外不支持 MDK 前段时间发布的 MDK5.24a,因为这个版本不支持 MDK4 创建的工程转换为 MDK5,所以要使用这个最新的版本,需要给 MDK5 安装 MDK4 的兼容包。

STM32-V4 板子配套的例子固定使用 IAR6.3,其它版本未做测试。

STM32-V5 和 STM32-V6 板子使用 IAR7.5,其它版本未做测试。

1.2 uC/Probe 简介

uC/Probe 就是一款图形化的调试信息展示工具。

- ◆ 最重要的功能是 uCOS-II 和 uCOS-III 的信息展示，方便用户直观的查看任务堆栈、任务运行状态、CPU 利用率、任务组件等执行情况。
- ◆ 还有一个功能就是图形化的展示调试信息，这个怎么理解呢？其实就类似 MDK 或者 IAR 的调试，只是支持图形化展示，效果更直观。
- ◆ uC/Probe 支持两种调试方法：
 - 一种是使用下载器，可以使用 JLINK，CMSIS-DAP，Cypress PSoc Prog 等，这种方式比较简单，无需对程序附加代码。
 - 另一种是使用通信接口，支持 RS232，USB 和 TCP/IP。这种方式比较麻烦，需要用户添加相应的驱动代码到自己的工程中。

1.3 MDK 使用方法

MDK 工程使用 uC/Probe 的方法也比较容易实现，主要分四步：

- ◆ 配置 MDK 工程，下载程序到开发板。
- ◆ 配置 uC/Probe，并加载 MDK 工程的可执行文件 xxx.axf。
- ◆ 添加 uC/Probe 中的控件。
- ◆ 添加 uCOS-III 信息组件。

通过这四步就可以使用软件 uC/Probe 了，下面我们进行具体的说明。

1.3.1 配置 MDK 工程并下载到开发板

使能 os_cfg.h 文件中的宏定义：

- ◆ #define OS_CFG_DBG_EN 1u
- ◆ #define OS_CFG_STAT_TASK_EN 1u

使能 cpu_cfg.h 文件中的宏定义：

- ◆ #define CPU_CFG_INT_DIS_MEAS_EN 1u

注意，这里配置为 1，跟使用 DEF_ENABLE 是一个意思：

```
#define DEF_DISABLED 0u  
#define DEF_ENABLED 1u
```

这里仅使能了统计任务的宏定义 OS_CFG_STAT_TASK_EN 还不行，一定要在程序中调用函数 OSStatTaskCPUUsageInit 进行初始化。使能宏定义 CPU_CFG_INT_DIS_MEAS_EN 也一样，需要调用函数 CPU_IntDisMeasMaxCurReset，具体实现如下：

```
/*
*****
*   函 数 名: AppTaskStart
*   功能说明: 这是一个启动任务，在多任务系统启动后，必须初始化滴答计数器。本任务主要实现按键检测。
*   形    参: p_arg 是在创建该任务时传递的形参
*   返 回 值: 无
*   优 先 级: 2
*****
*/
static void AppTaskStart (void *p_arg)
{
    OS_ERR      err;

    (void)p_arg;

    CPU_Init(); /* 此函数要优先调用，因为外设驱动中使用的 us 和 ms 延迟是基于此函数的 */
    bsp_Init();
    BSP_Tick_Init();

#if OS_CFG_STAT_TASK_EN > 0u
    OSStatTaskCPUUsageInit(&err);
#endif

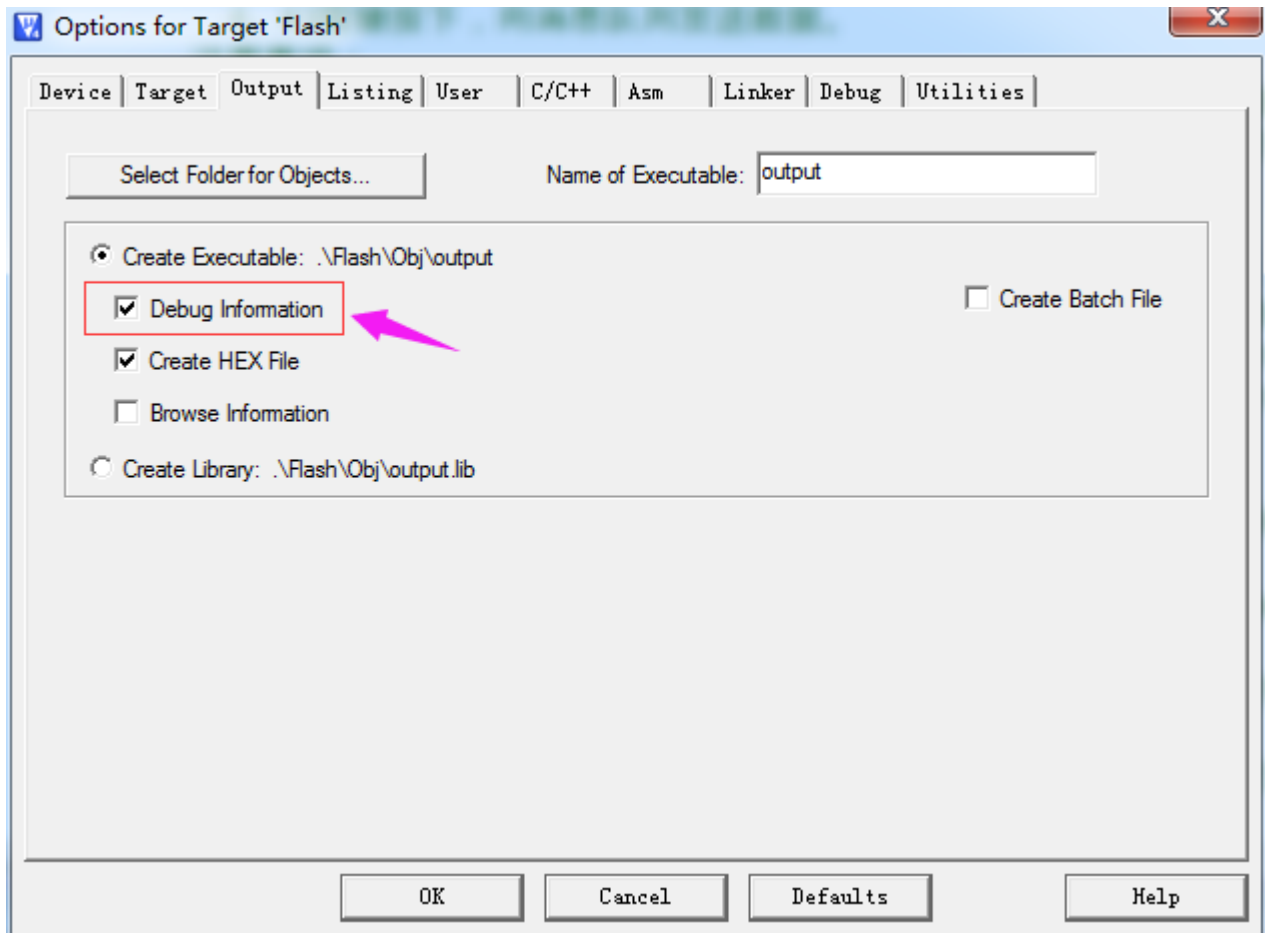
#ifdef CPU_CFG_INT_DIS_MEAS_EN
    CPU_IntDisMeasMaxCurReset();
#endif

    /* 省略 */
}
```

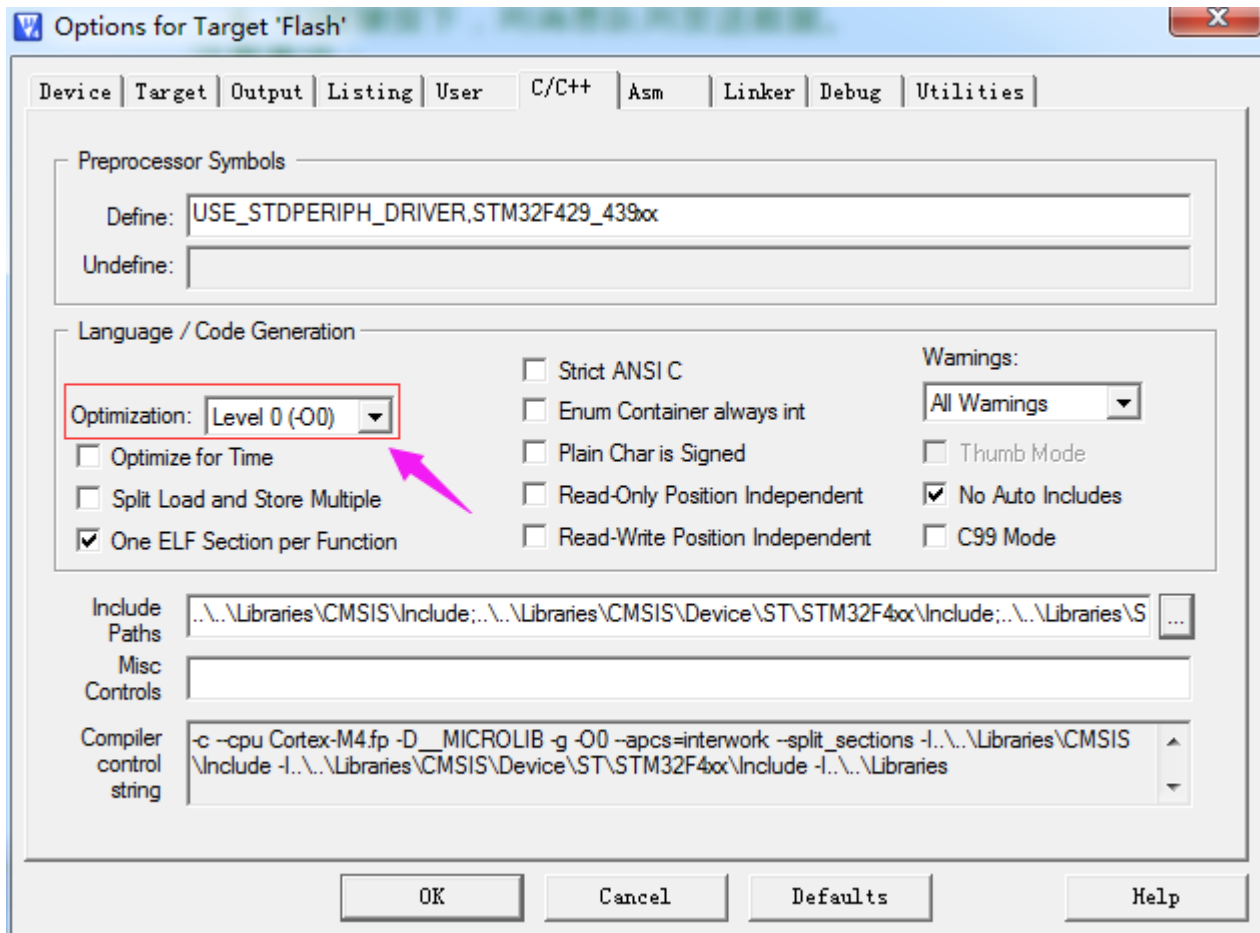
为了方便测试,os_cfg.h 文件中的宏定义 OS_CFG_APP_HOOKS_EN,OS_CFG_ARG_CHK_EN 也都使能。

宏定义设置完毕后,MDK 工程中有两处需要配置:

- ◆ 勾选 option->output -> Debug Infomation 选项。



- ◆ 优化等级要选择最低的 0，防止一些调试信息被优化掉：



配置完毕后，全编译工程并下载到开发板。

1.3.2 配置 uC/Probe，并加载可执行文件

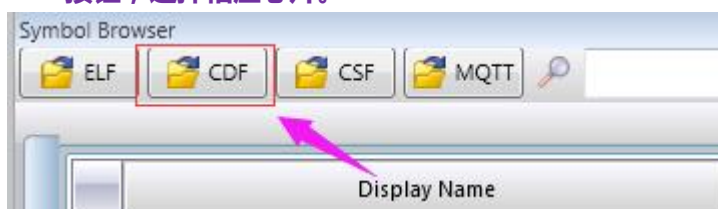
◆ 第 1 步：打开 uC/Probe，选择 Settings 进行配置。



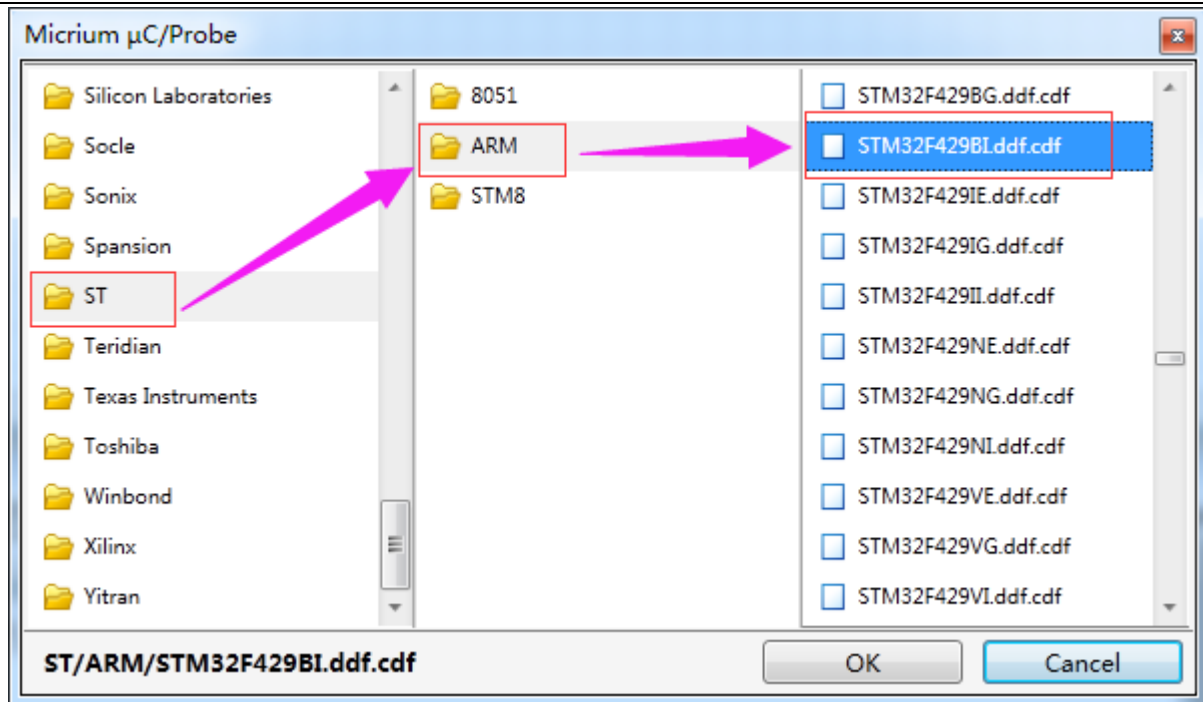
打开后，设置如下地方：



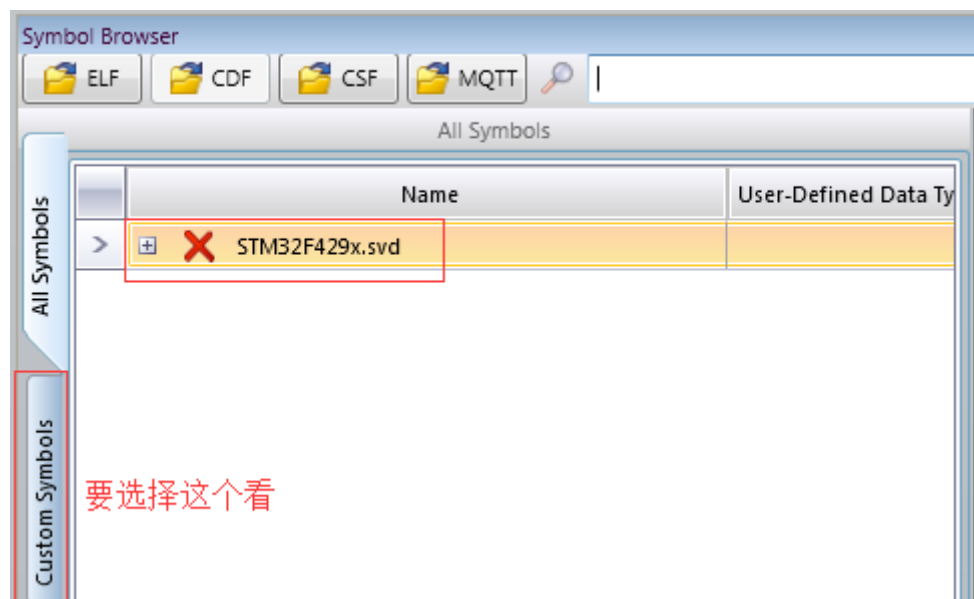
◆ 第 2 步：打开“CDF”按钮，选择相应芯片。






打开后选择 STM32F429BIT6 (V4 板子是 STM32F103ZET6 , V5 板子是 STM32F407IGT6 , V6 板子是 STM32F429BIT6)。



选择完毕后，别忘了点击“OK”按钮。可以看到多了一个文件 STM32F429x.svd。



将其展开，就可以看到各种寄存器，这些都可以拖到客户区查看的。

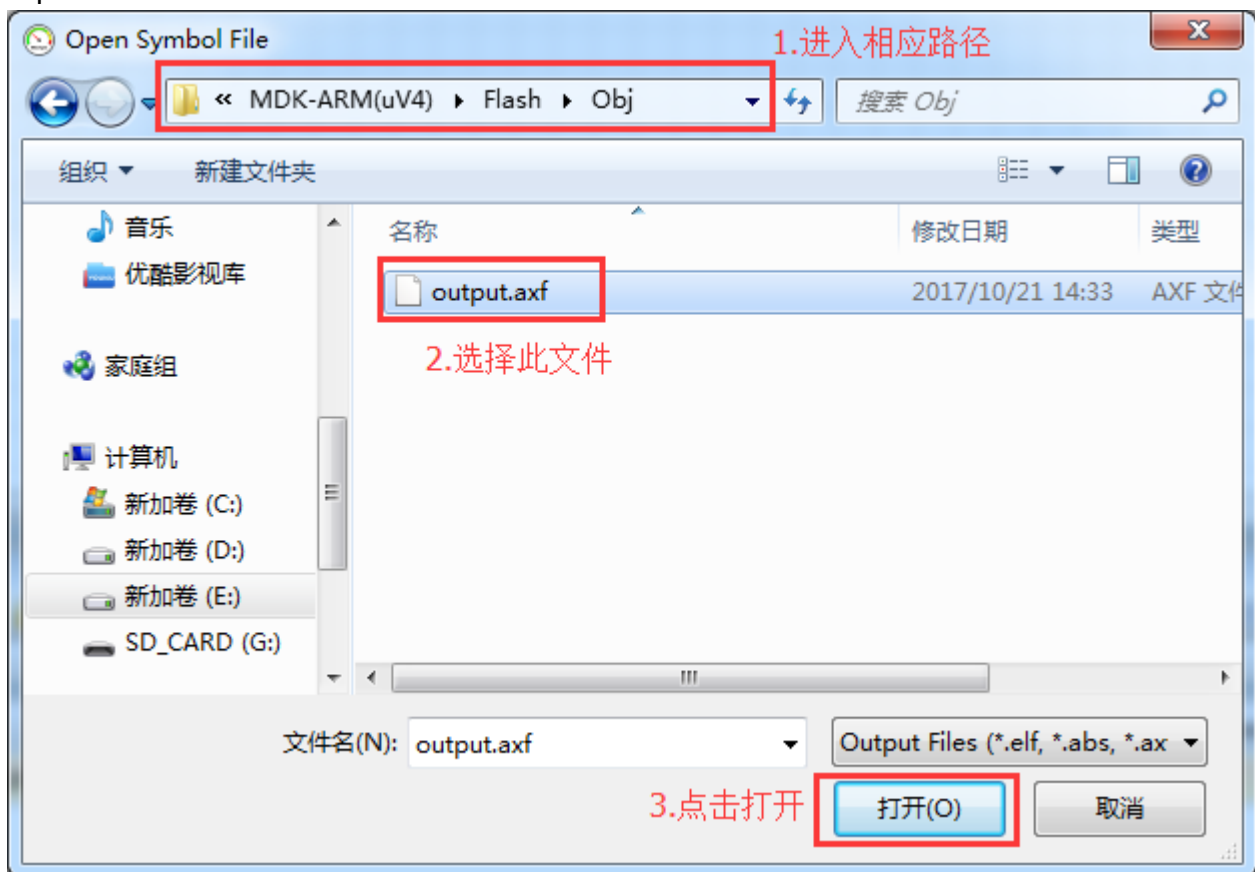
	Name	User-Defined Data Type	C Data Type	Size
 	STM32F429x.svd		N/A	332
	ADC1		Peripheral	4
	ADC2		Peripheral	4
	ADC3		Peripheral	4
	C_ADC		Peripheral	4
	CAN1		Peripheral	4
	CAN2		Peripheral	4

◆ 第 3 步，加载 MDK 生成的可执行文件。

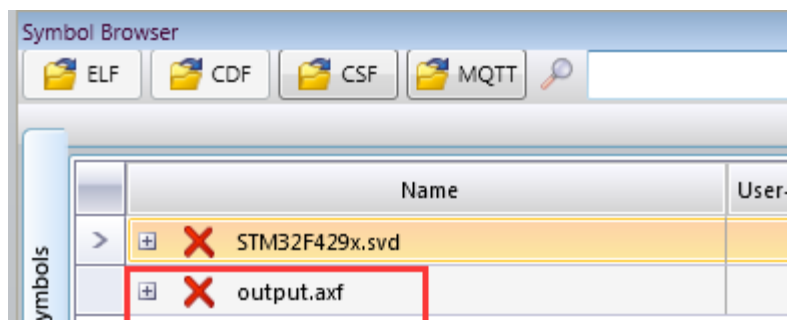
对于本专题配套的例子，可执行文件存储在路径\Project\MDK-ARM(uV4)\Flash\Obj 里面，后缀是 axf。首先点击 uC/Probe 上面的按钮“ELF”。



弹出的窗口中，进入本教程配套例子的\Project\MDK-ARM(uV4)\Flash\Obj 路径里面，选择可执行文件 output.axf。



可以看到 output.axf 也加载进来了。



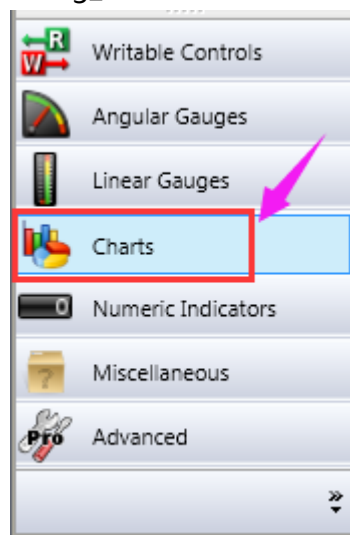
将其展开后，就可以看到 MDK 工程中的 C 文件及其里面的全局变量：

	Name	User-Defined Data Type	C Data Type
[-] X	output.axf		N/A
[-]	bsp_ext_io.c		N/A
	g_HC574	uint32_t	unsigned int
[+]	bsp_key.c		N/A
[+]	bsp_uart_fifo.c		N/A
[+]	cpu_core.c		N/A
[-]	main.c		N/A
[+]	AppPrintfSemp	OS_SEM	os_sema
	AppTaskCOMStk	CPU_STK [512]	unsigned int [512]
[+]	AppTaskCOMTCB	OS_TCB	os_tcb
	AppTaskMsgProStk	CPU_STK [512]	unsigned int [512]

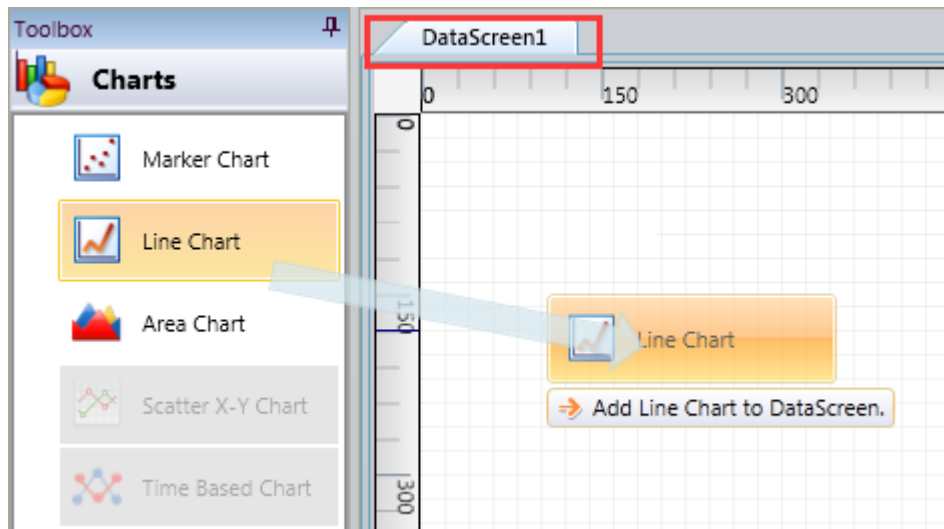
这些全局变量都可以添加到 uC/Probe 里面进行观察，而且是图形化的观察，就跟大家使用 MDK 调试组件观察一样，只是更形象，用户可以选择各种效果展示出来。

1.3.3 添加 uC/Probe 中的控件

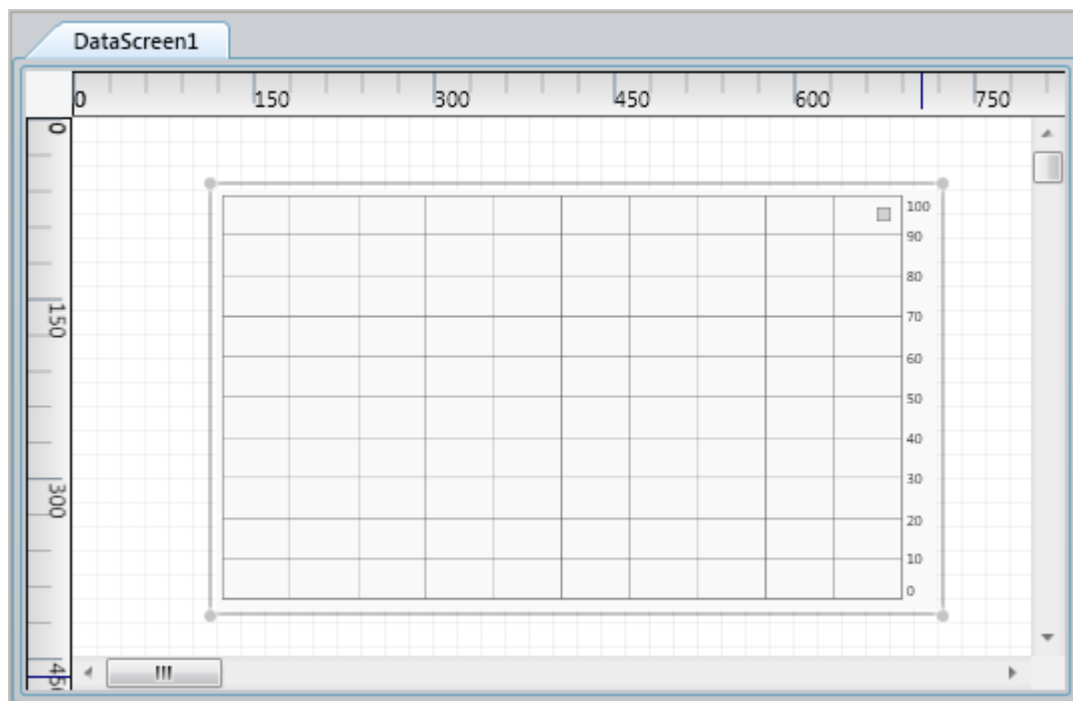
这里我们将 main.c 文件中的全局变量 g_ucCount 添加到波形控件里面，选择 Charts：



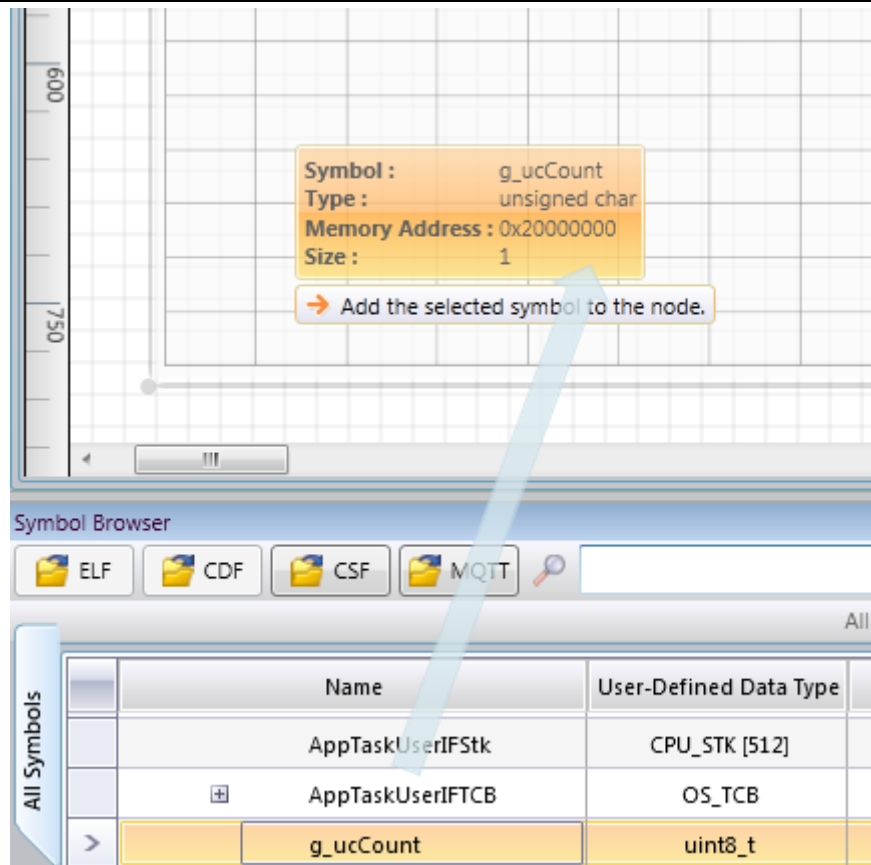
鼠标左键选中 Line Chart，然后拖动到 DataScreen1 区域。



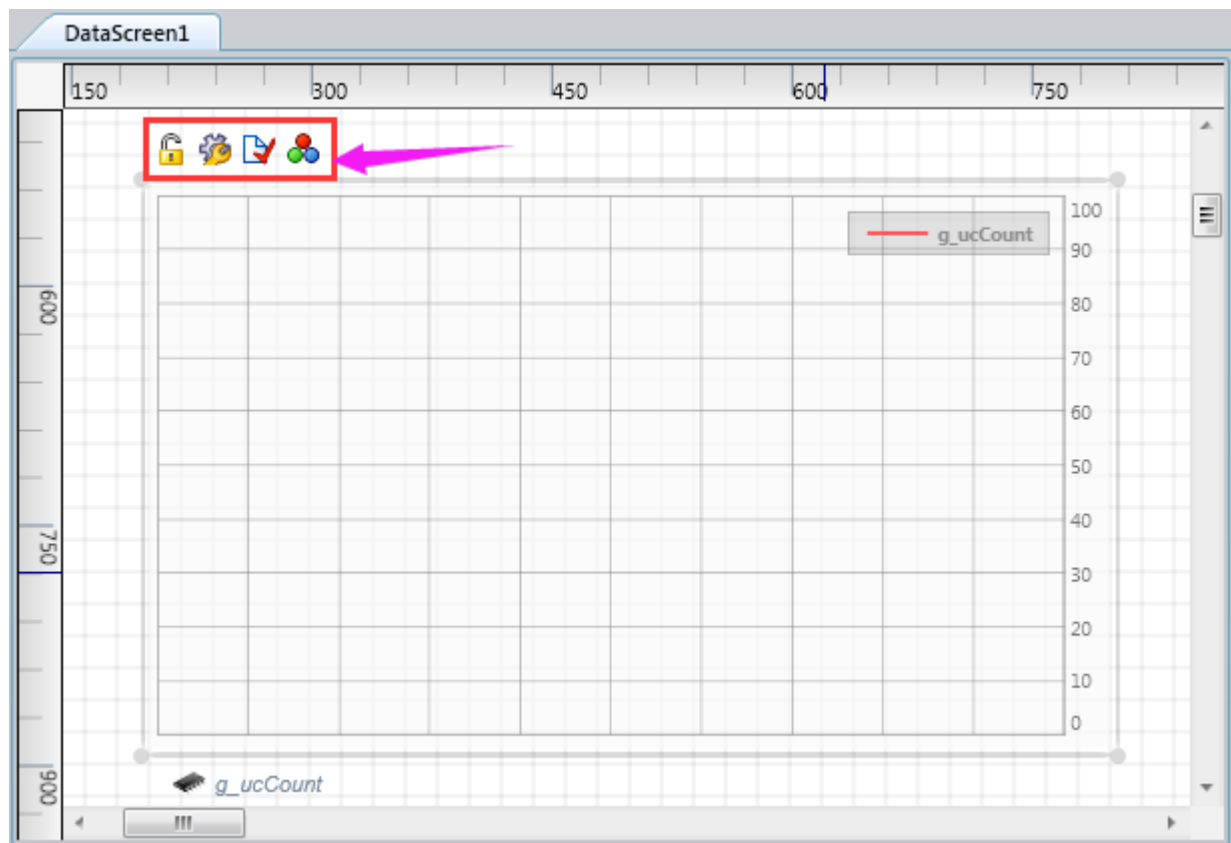
可以看到一个类似示波器的表格：



添加完图形组件后，我们就可以将全局变量 `g_ucCount` 拖动到刚刚添加的图形组件上，添加方法跟添加图形组件一样，鼠标左键选中全局变量 `g_ucCount`，然后拖到图形控件上：



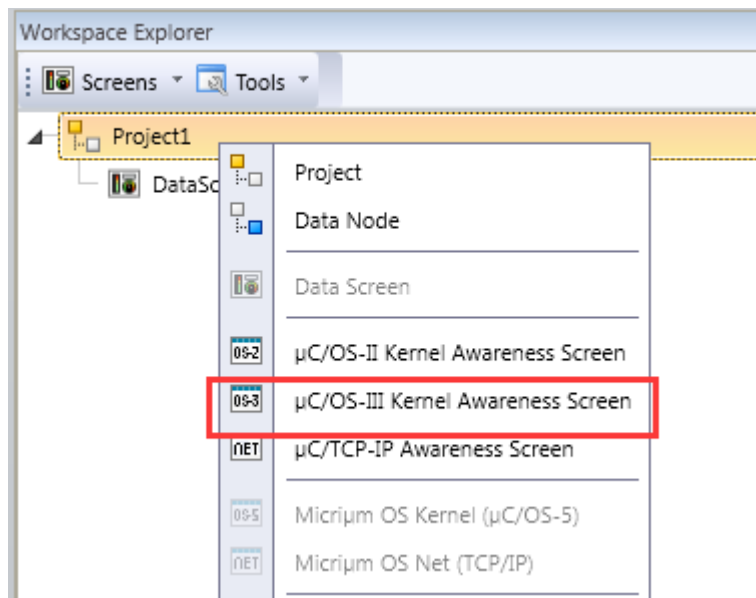
拖上去后的效果如下：



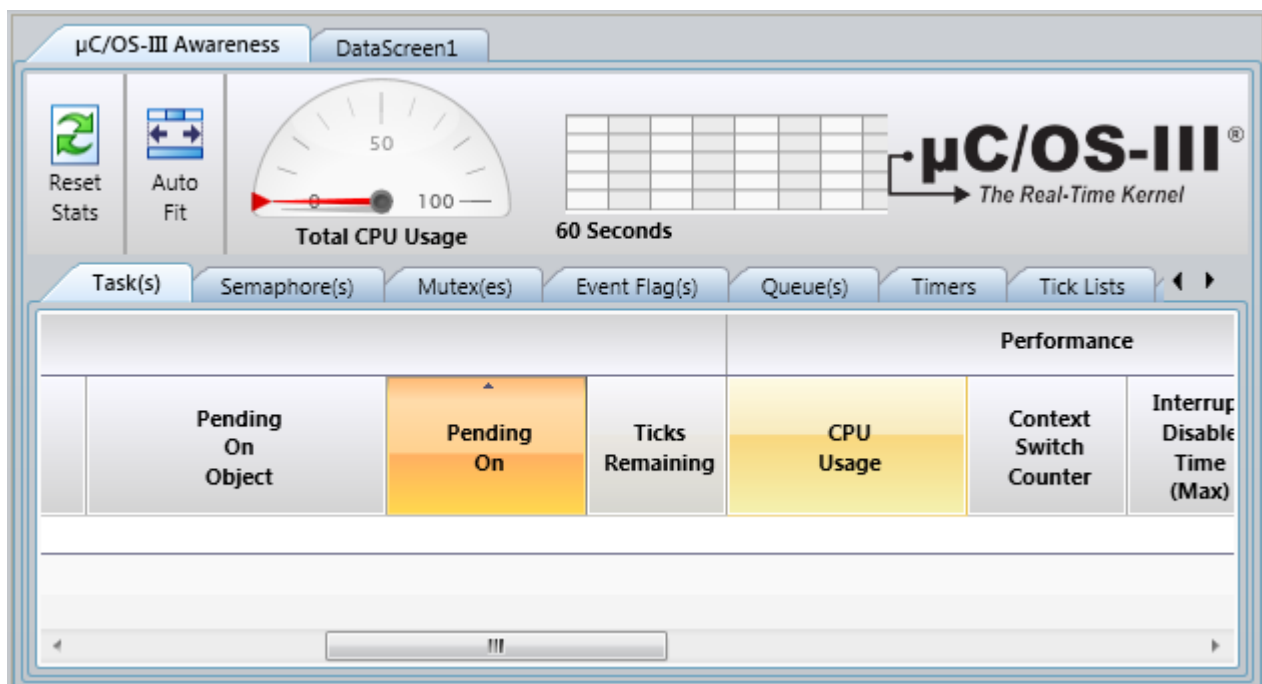
大家也可以通过上图中的四个小图标设置这个波形控件，我们这里就不做设置了。

1.3.4 添加 uCOS-III 信息组件

下面是最后一步，将 uCOS-III 的信息展示组件添加进来，右击 Project1：

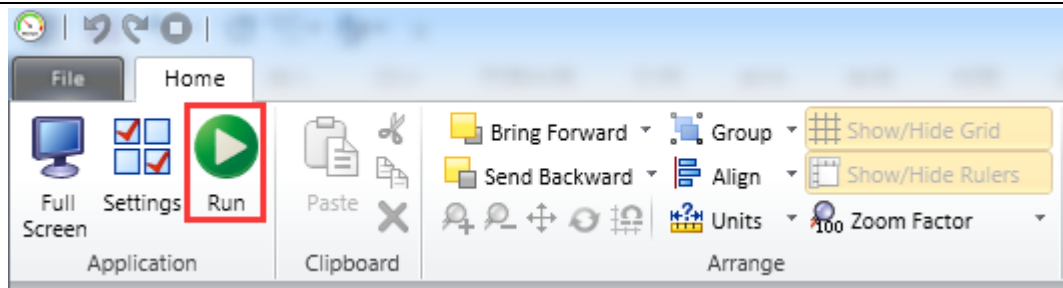


弹出如下界面：

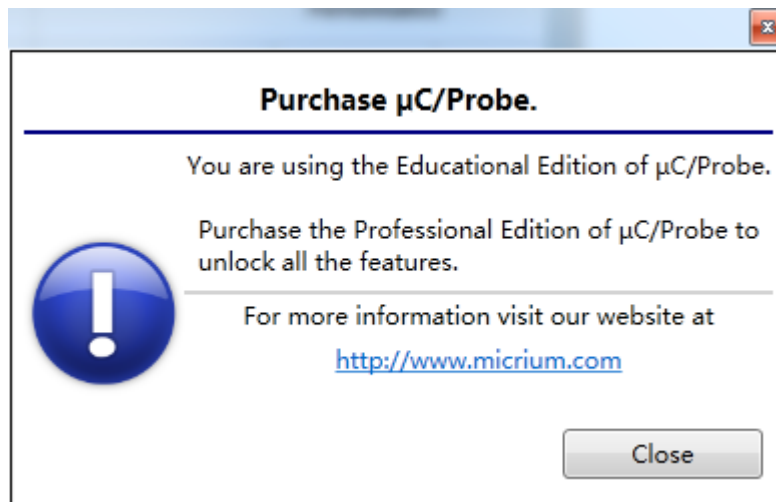


1.3.5 效果展示

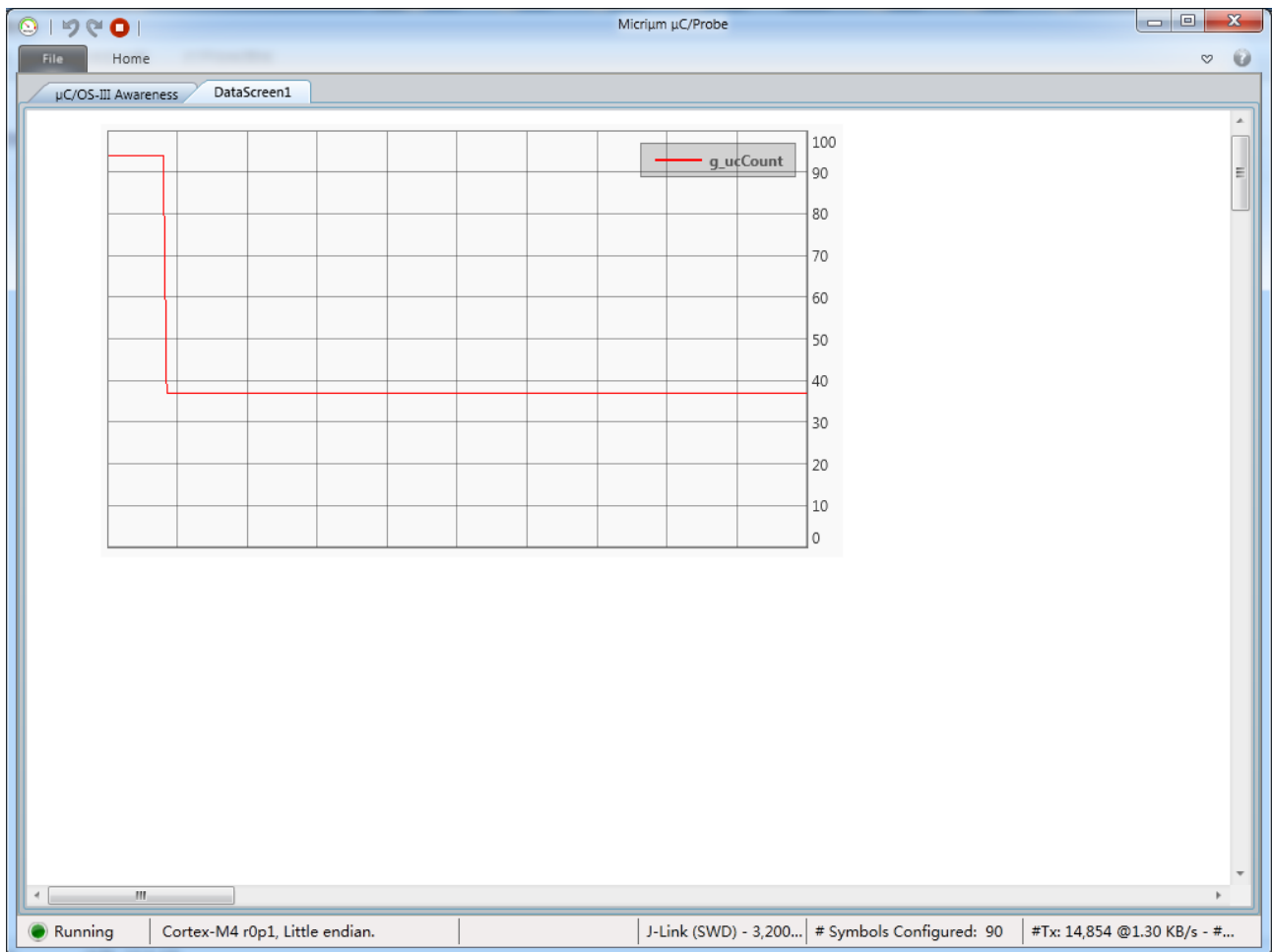
现在就可以点击左上角的“RUN”按钮了：



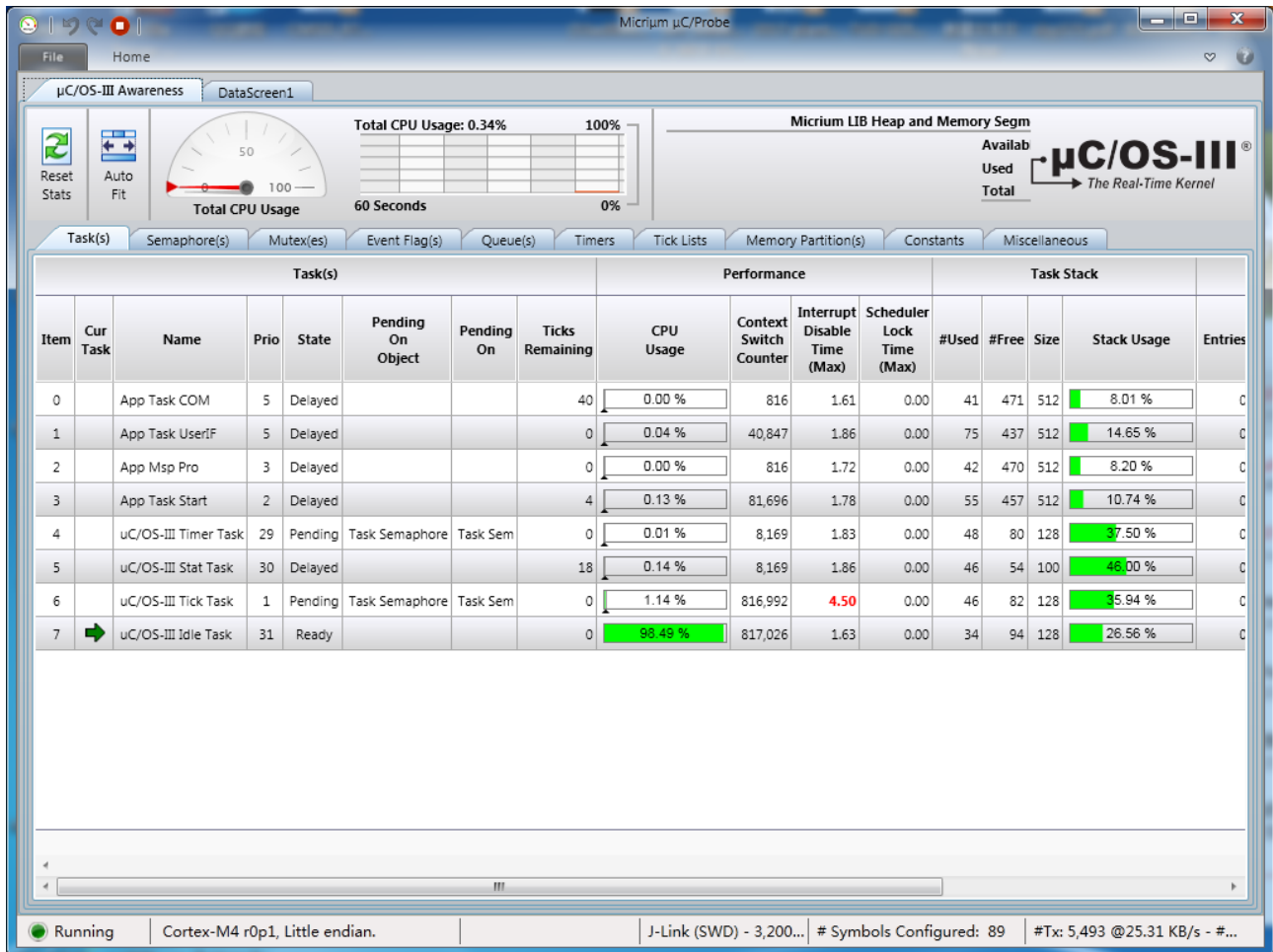
由于是教育版，点击“RUN”按钮后，会先弹出如下这个窗口：



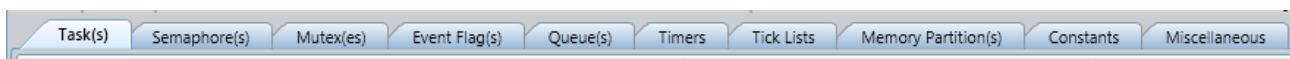
点击“Close”关闭按钮即可：



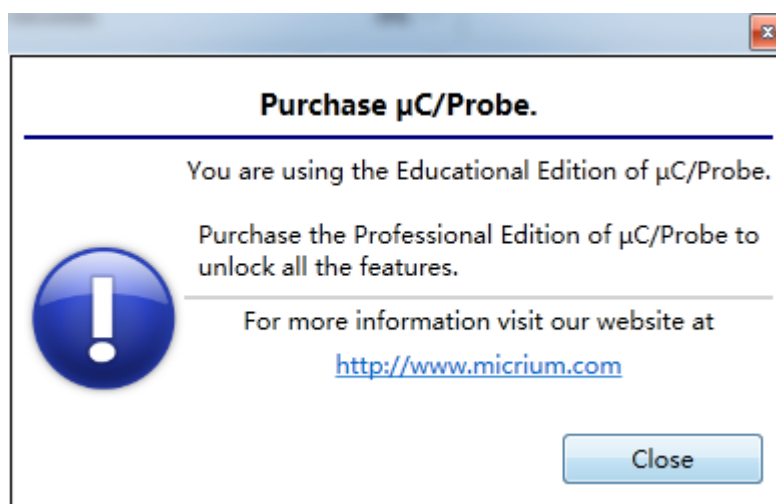
当前展示出来的就是全局变量 g_ucCount 的波形效果。在左上角，点击“uCOS-III Awareness”，就可以查看 uCOS-III 的信息了：



里面的这些选项都是可以选择查看的：



大家可以自己测试时一个一个点击，看看实际效果。由于我们使用的是评估版本，运行一段实际后就自动的退出了，弹出这个窗口：



需要再次查看，就继续点击运行即可。关于 MDK 使用 uC/Probe，就为大家讲解这么多，更多其它功能，

大家可以看官方手册进行学习，或者自己摸索使用也可以（由于是图形化的，点点就出效果了）。

1.4 IAR 使用方法

IAR 工程使用 uC/Probe 的方法也比较容易实现，主要分四步：

- ◆ 配置 IAR 工程，下载程序到开发板。
- ◆ 配置 uCProbe，并加载 IAR 工程的可执行文件 xxx.out。
- ◆ 添加 uC/Probe 中的控件。
- ◆ 添加 uCOS-III 信息组件。

通过这四步就可以使用软件 uC/Probe 了，下面我们进行具体的说明。

1.4.1 配置 IAR 工程并下载到开发板

使能 os_cfg.h 文件中的宏定义：

- ◆ #define OS_CFG_DBG_EN 1u
- ◆ #define OS_CFG_STAT_TASK_EN 1u

使能 cpu_cfg.h 文件中的宏定义：

- ◆ #define CPU_CFG_INT_DIS_MEAS_EN 1u

注意，这里配置为 1，跟使用 DEF_ENABLE 是一个意思：

```
#define DEF_DISABLED 0u
#define DEF_ENABLED 1u
```

这里仅使能了统计任务的宏定义 OS_CFG_STAT_TASK_EN 还不行，一定要在程序中调用函数 OSStatTaskCPUUsageInit 进行初始化。使能宏定义 CPU_CFG_INT_DIS_MEAS_EN 也一样，需要调用函数 CPU_IntDisMeasMaxCurReset，具体实现如下：

```
/*
*****
* 函数名: AppTaskStart
* 功能说明: 这是一个启动任务，在多任务系统启动后，必须初始化滴答计数器。本任务主要实现按键检测。
* 形参: p_arg 是在创建该任务时传递的形参
* 返回值: 无
* 优先级: 2
*****
*/
static void AppTaskStart (void *p_arg)
{
    OS_ERR    err;

    (void)p_arg;

    CPU_Init(); /* 此函数要优先调用，因为外设驱动中使用的 us 和 ms 延迟是基于此函数的 */
    bsp_Init();
    BSP_Tick_Init();
}
```

```
#if OS_CFG_STAT_TASK_EN > 0u
    OSStatTaskCPUUsageInit(&err);
#endif

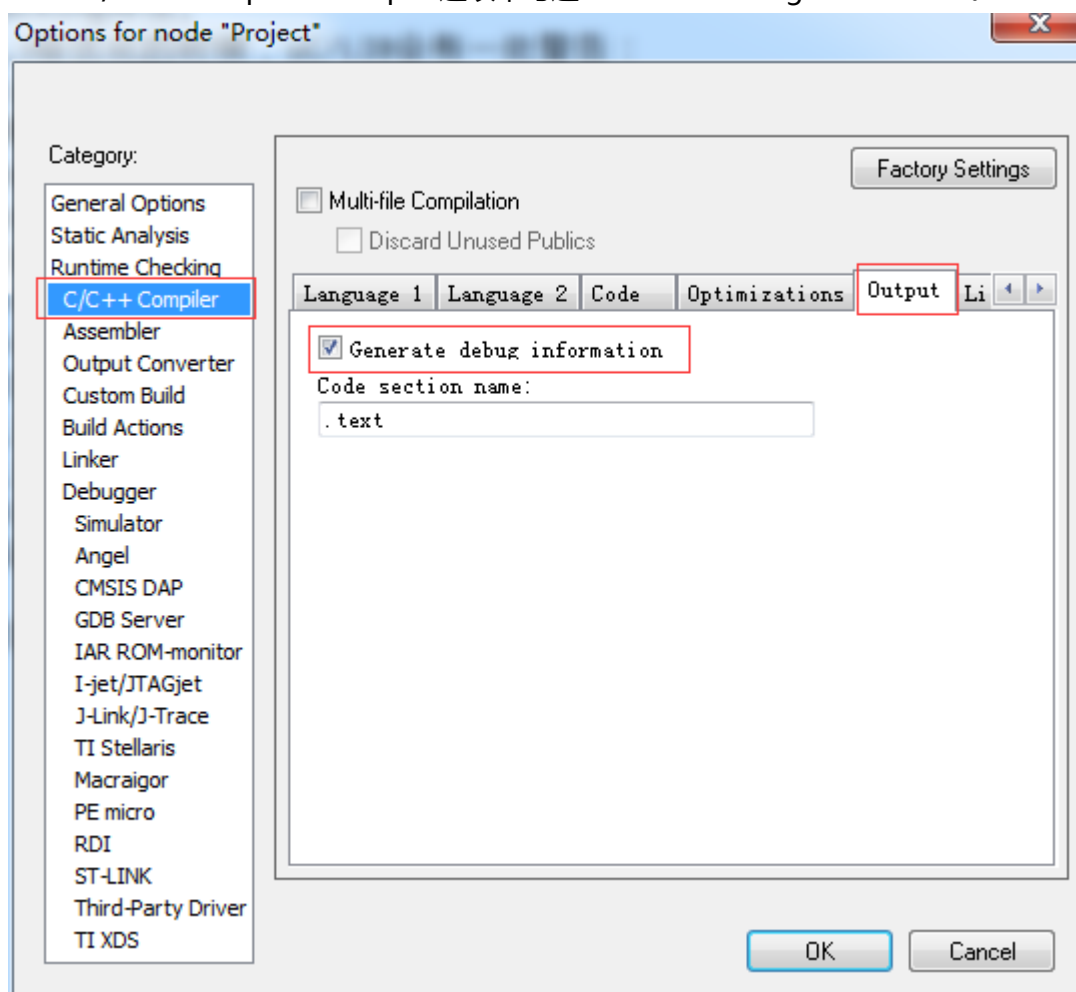
#ifdef CPU_CFG_INT_DIS_MEAS_EN
    CPU_IntDisMeasMaxCurReset();
#endif

/* 省略 */
}
```

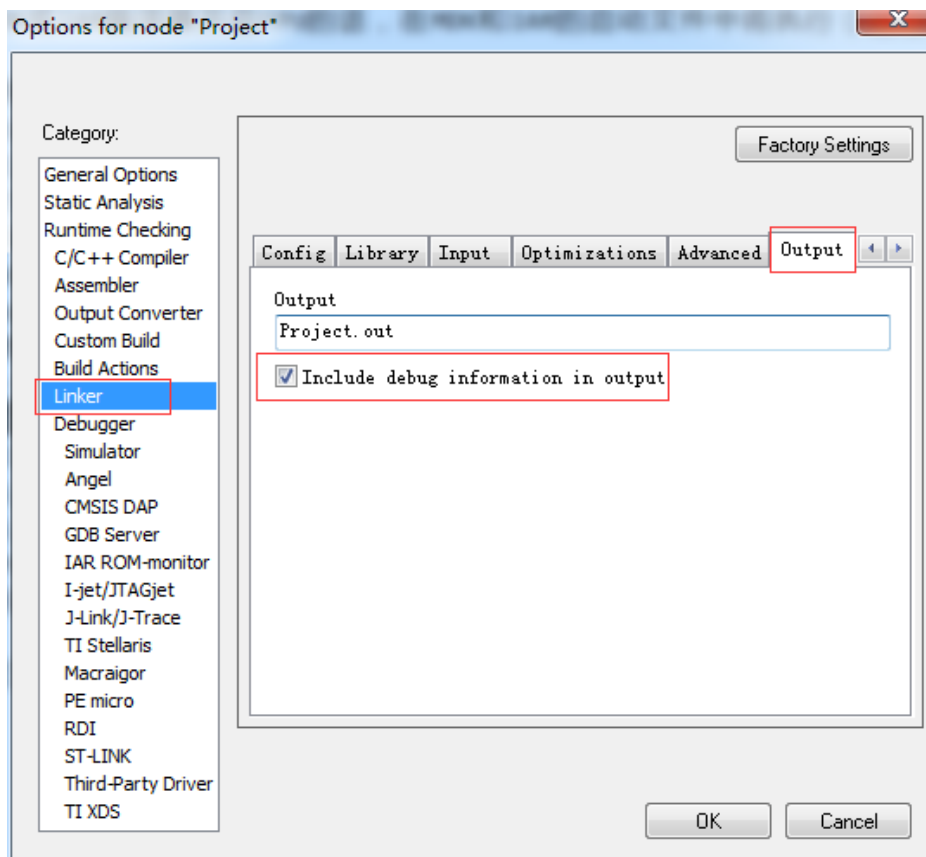
为了方便测试,os_cfg.h 文件中的宏定义 OS_CFG_APP_HOOKS_EN,OS_CFG_ARG_CHK_EN 也都使能。

宏定义设置完毕后, IAR 工程中有三处需要配置:

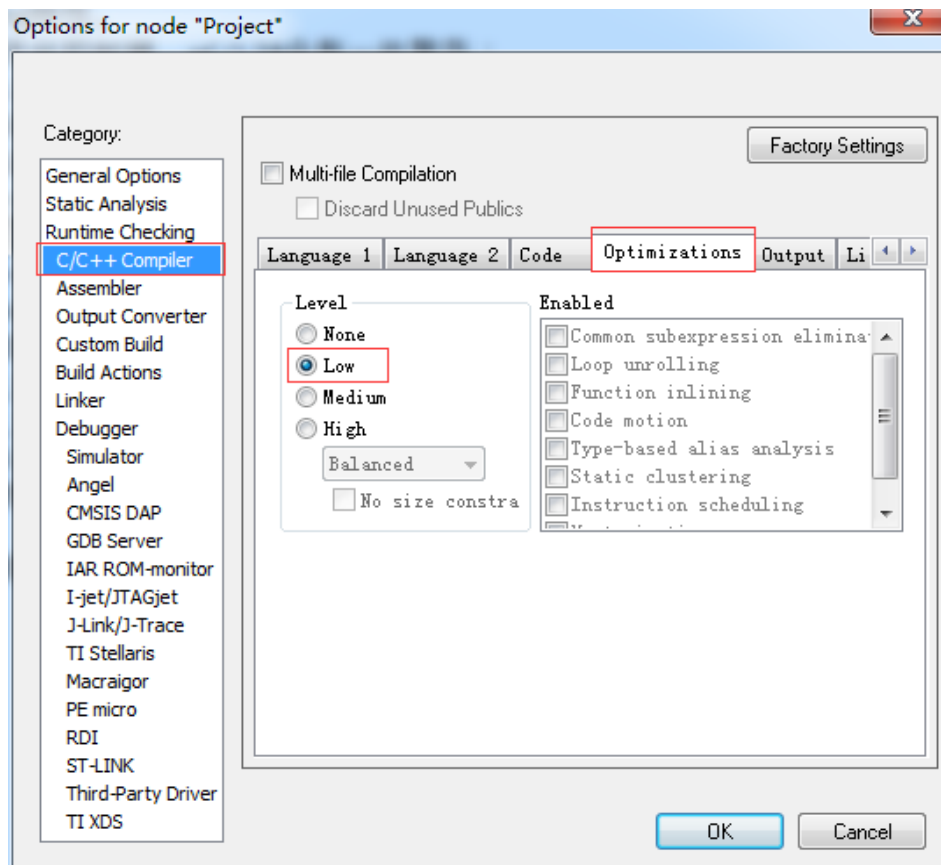
- ◆ option->C/C++ Compiler->Output 选项中勾选 Generate debug information。



- ◆ option->linker->Output 选项中勾选 Include Generate debug information in output。



- ◆ 优化等级要选择 low 或者 none，防止一些调试信息被优化掉：



配置完毕后，全编译工程并下载到开发板。

1.4.2 配置 uC/Probe , 并加载可执行文件

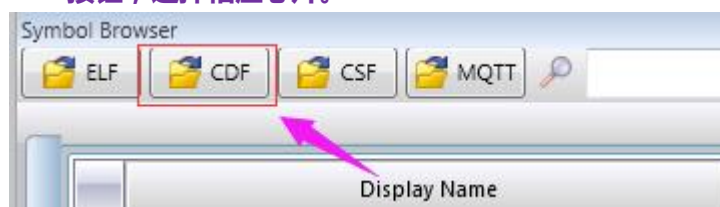
◆ 第 1 步：打开 uC/Probe , 选择 Settings 进行配置。



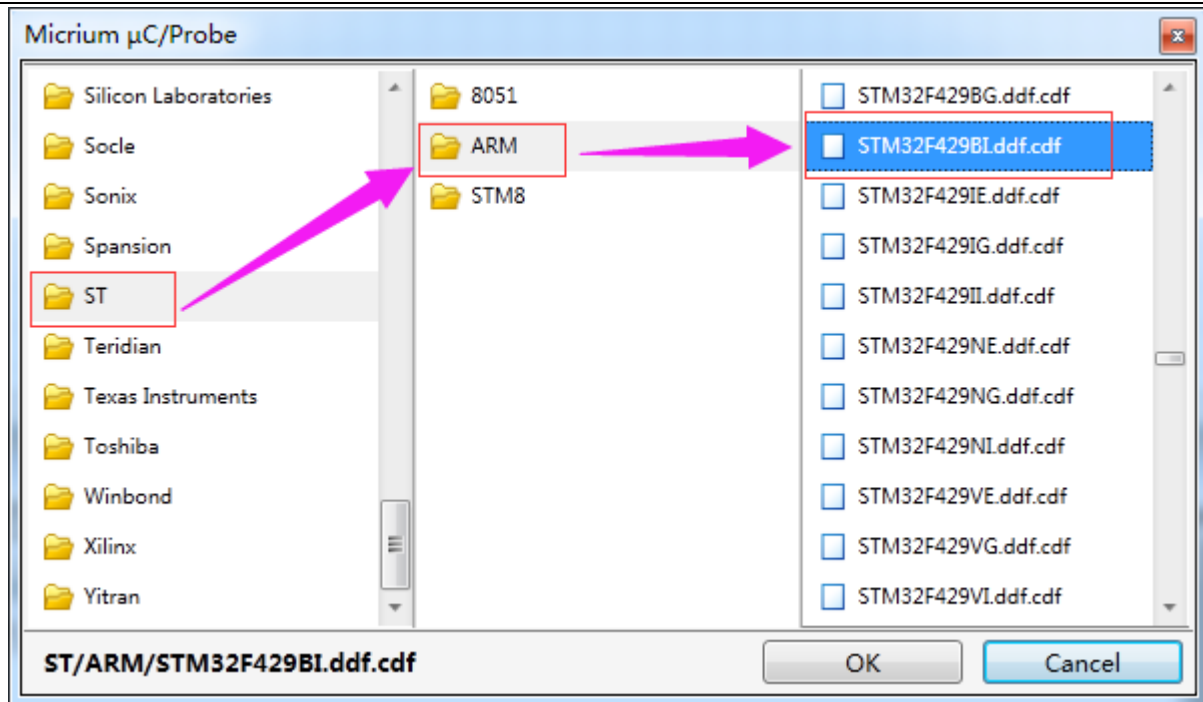
打开后，设置如下地方：



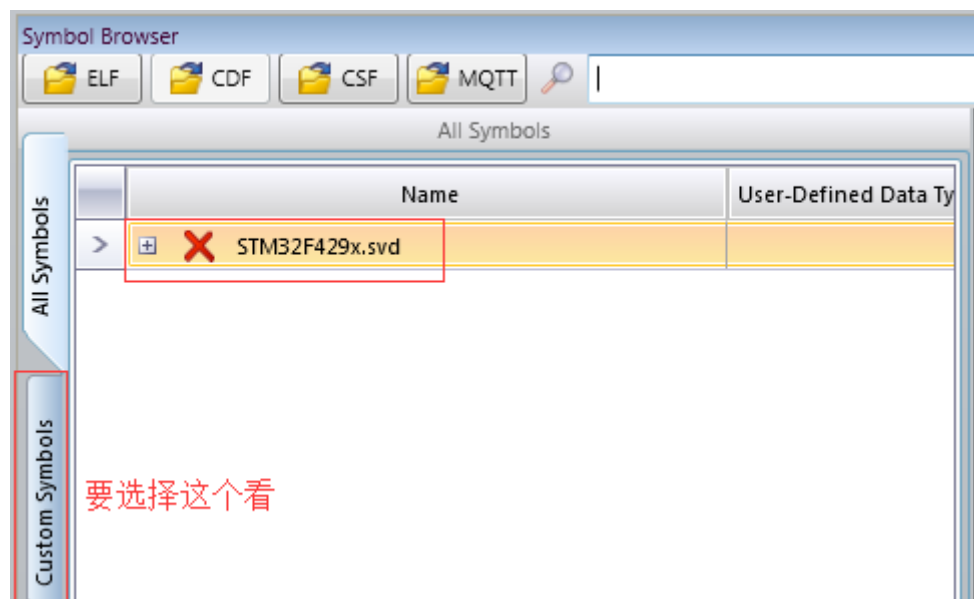
◆ 第 2 步：打开“CDF”按钮，选择相应芯片。





打开后选择 STM32F429BIT6 (V4 板子是 STM32F103ZET6 , V5 板子是 STM32F407IGT6 , V6 板子是 STM32F429BIT6)。



选择完毕后，别忘了点击“OK”按钮。可以看到多了一个文件 STM32F429x.svd。



将其展开，就可以看到各种寄存器，这些都可以拖到客户区查看的。

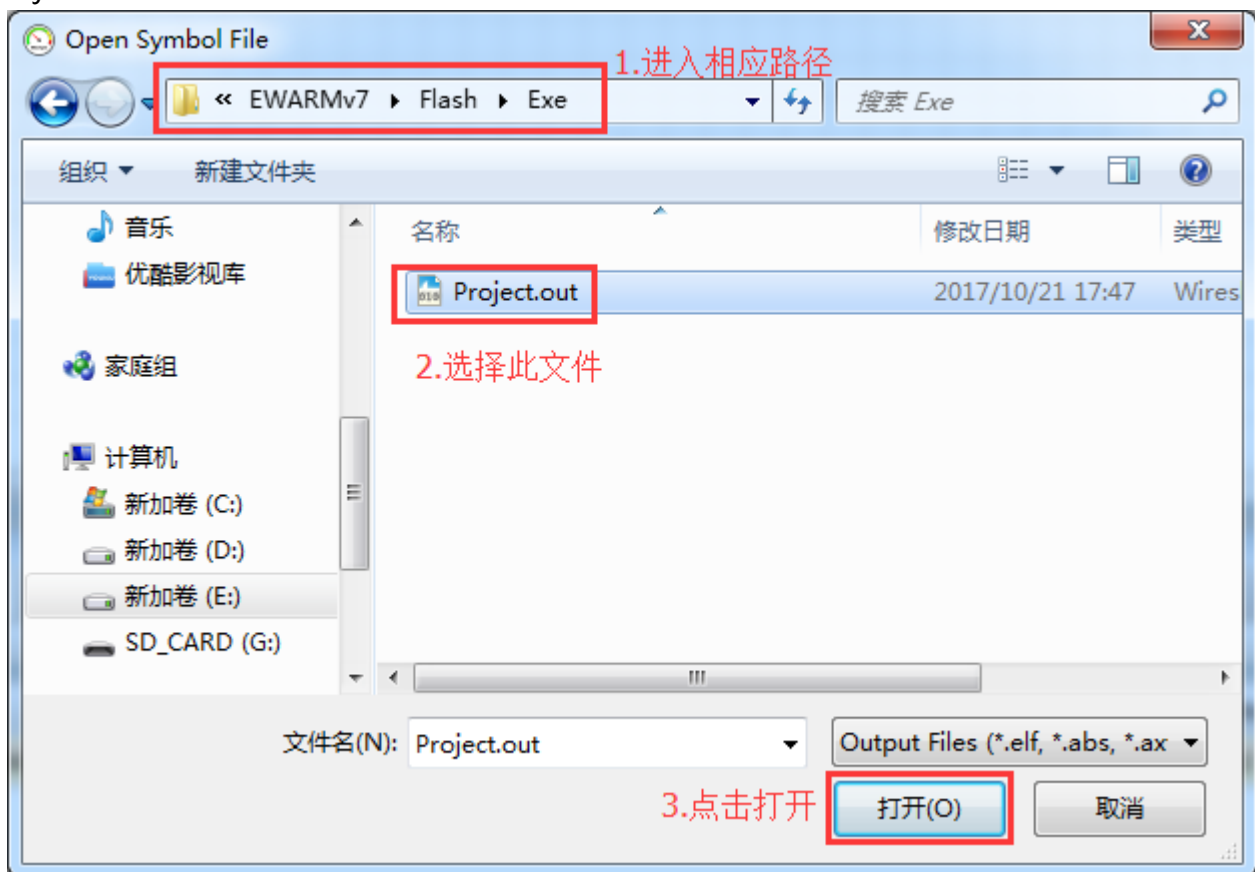
	Name	User-Defined Data Type	C Data Type	Size
 	STM32F429x.svd		N/A	332
	ADC1		Peripheral	4
	ADC2		Peripheral	4
	ADC3		Peripheral	4
	C_ADC		Peripheral	4
>	CAN1		Peripheral	4
	CAN2		Peripheral	4

◆ 第 3 步，加载 MDK 生成的可执行文件。

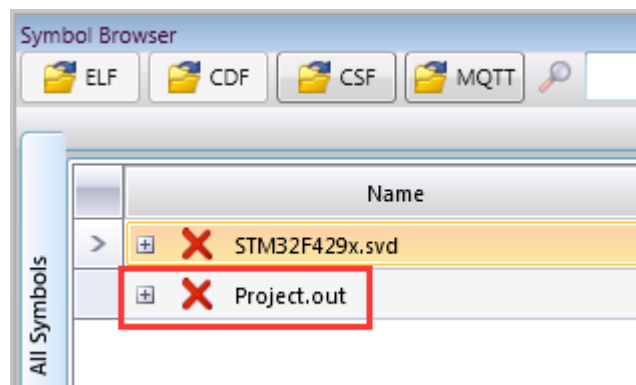
对于本专题配套的例子，可执行文件存储在路径\Project\EWARMv7\Flash\Exe 里面，后缀是 out。首先点击 uC/Probe 上面的按钮“ELF”。



弹出的窗口中，进入本教程配套例子的\Project\EWARMv7\Flash\Exe 路径里面，选择可执行文件 Project.out。



可以看到 Project.out 也加载进来了。



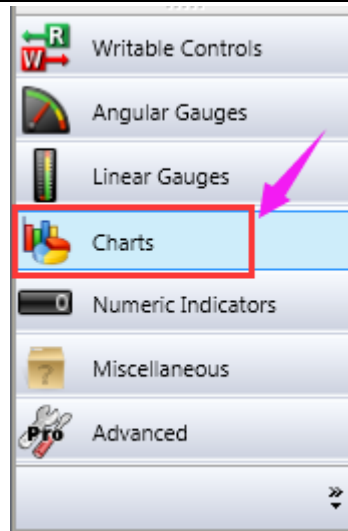
将其展开后，就可以看到 IAR 工程中的 C 文件及其里面的全局变量：

	Name	User-Defined Data Type
+	STM32F429x.svd	
>	Project.out	
-	bsp_ext_io.c	
	g_HC574	uint32_t
+	bsp_key.c	
+	bsp_uart_fifo.c	
+	cpu_core.c	
-	main.c	
+	AppPrintfSemp	OS_SEM
	AppTaskCOMStk	CPU_STK [512]
+	AppTaskCOMTCB	OS_TCB

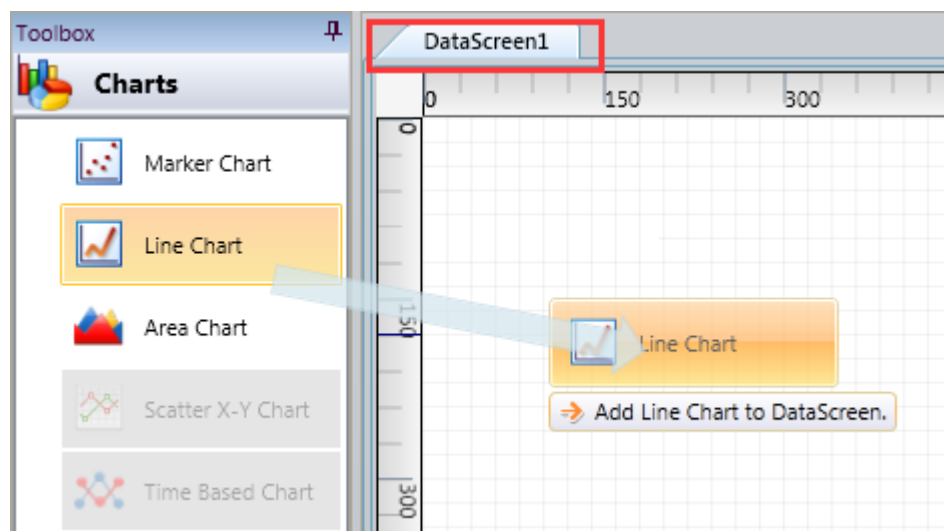
这些全局变量都可以添加到 uC/Probe 里面进行观察，而且是图形化的观察，就跟大家使用 IAR 调试组件观察一样，只是更形象，用户可以选择各种效果展示出来。

1.4.3 添加 uC/Probe 中的控件

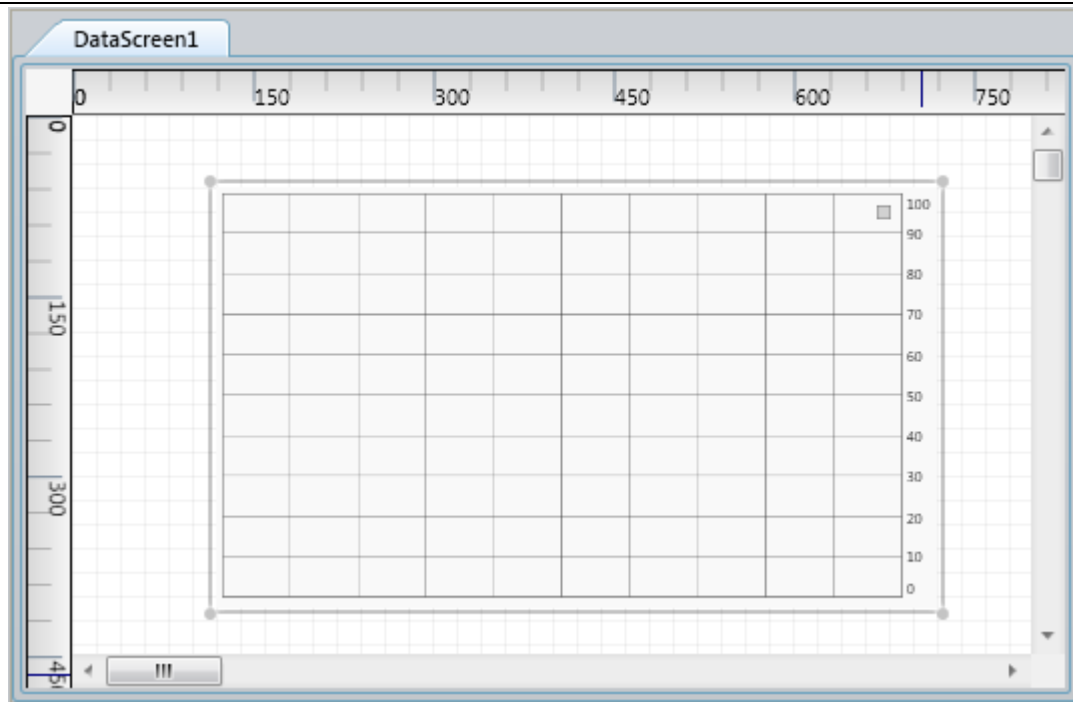
这里我们将 main.c 文件中的全局变量 g_ucCount 添加到波形控件里面，选择 Charts：



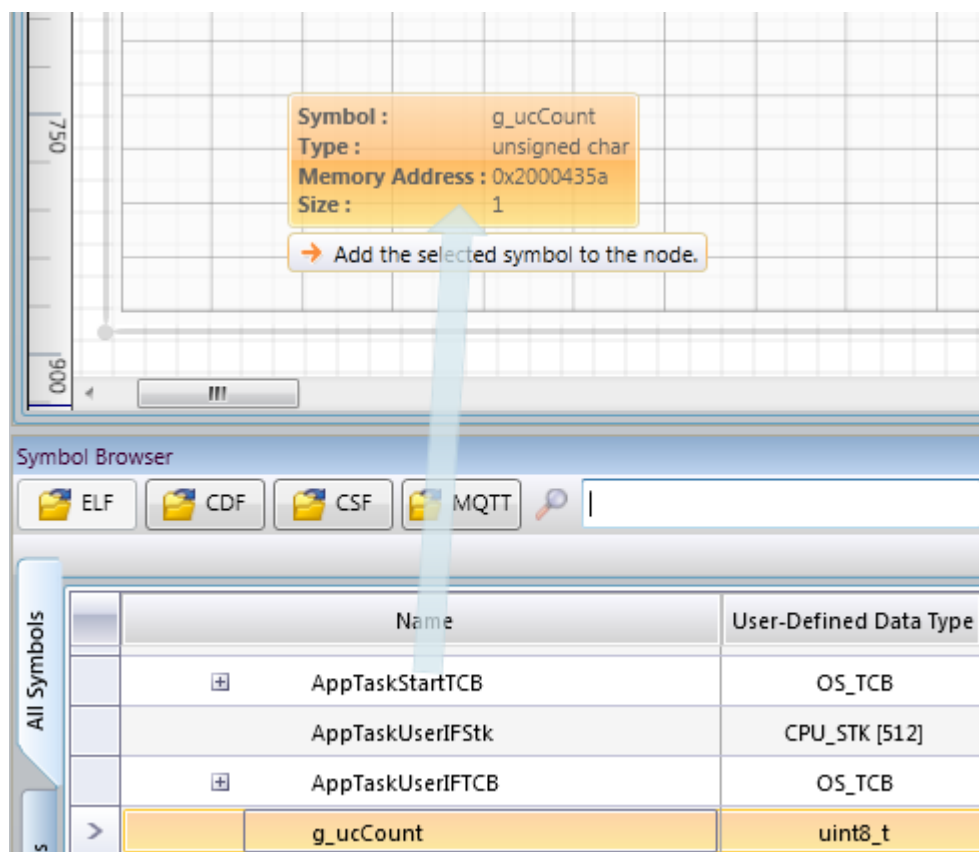
鼠标左键选中 Line Chart，然后拖动到 DataScreen1 区域。



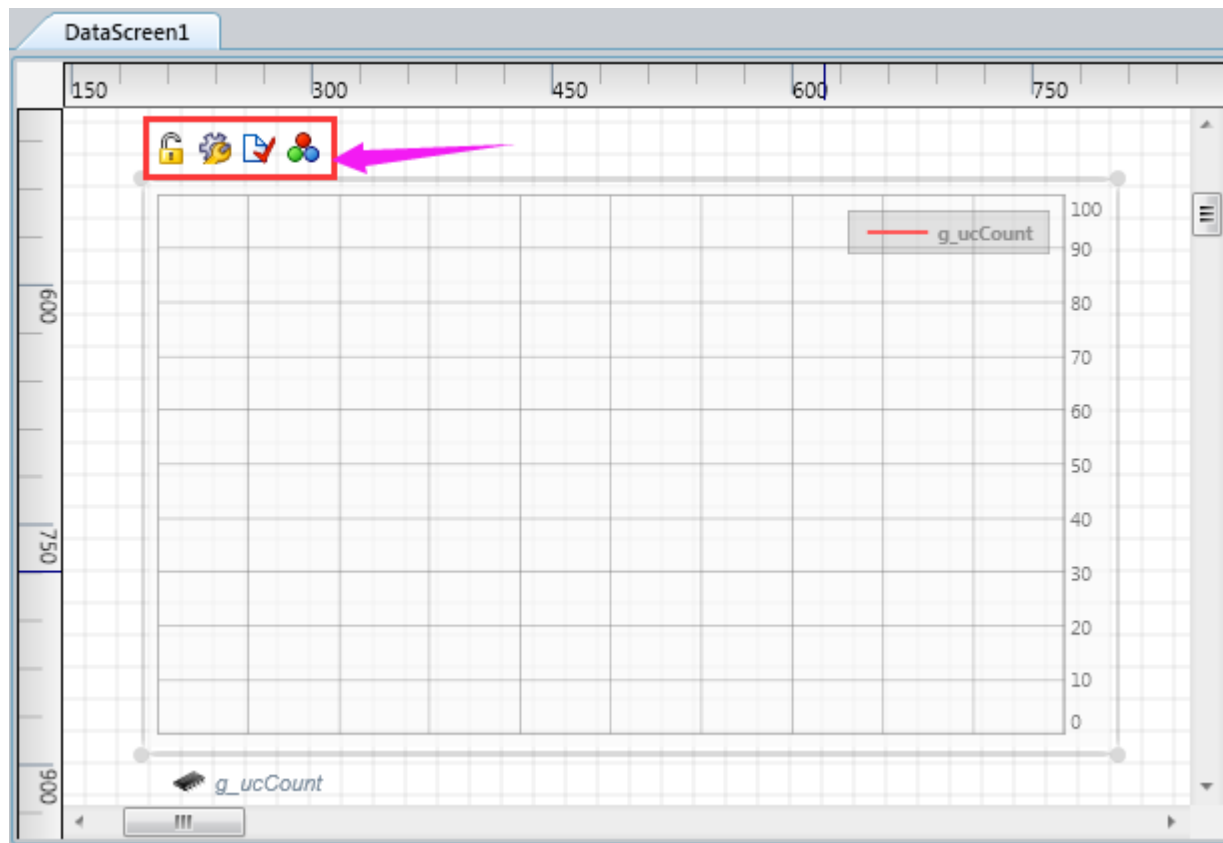
可以看到一个类似示波器的表格：



添加完图形组件后，我们就可以将全局变量 `g_ucCount` 拖动到刚刚添加的图形组件上，添加方法跟添加图形组件一样，鼠标左键选中全局变量 `g_ucCount`，然后拖到图形控件上：



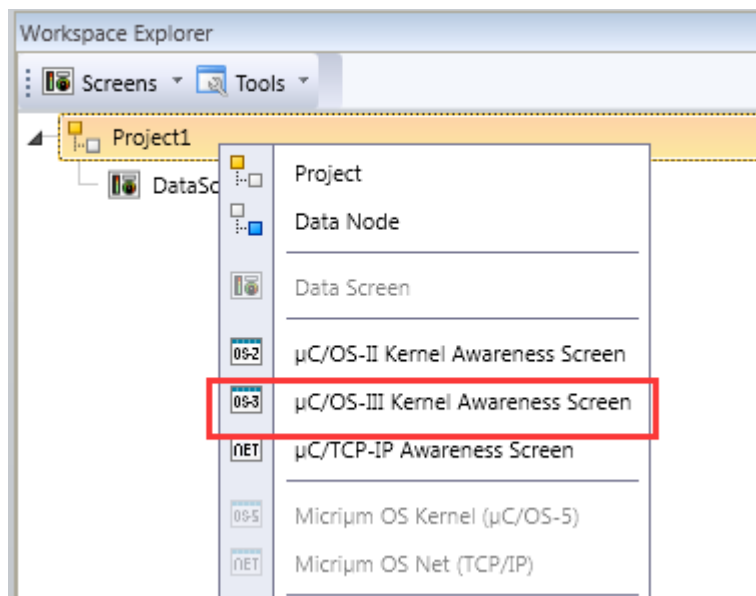
拖上去后的效果如下：



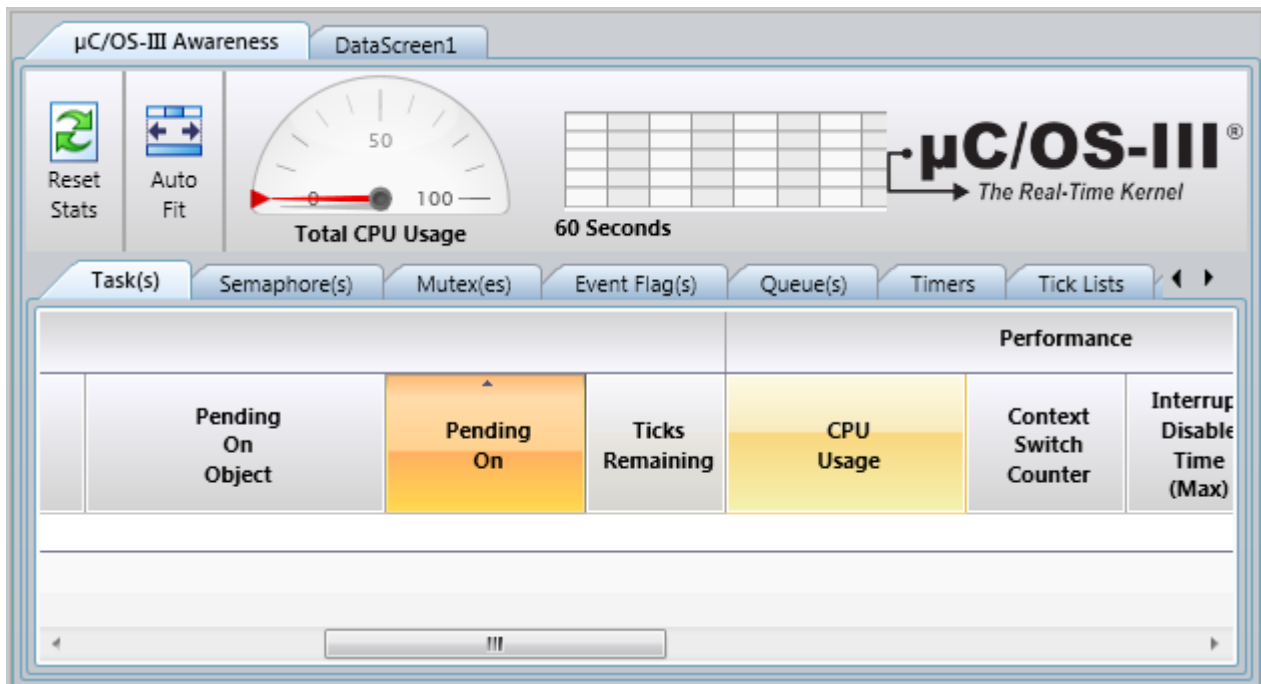
大家也可以通过上图中的四个小图标设置这个波形控件，我们这里就不做设置了。

1.4.4 添加 uCOS-III 信息组件

下面是最后一步，将 uCOS-III 的信息展示组件添加进来，右击 Project1：



弹出如下界面：

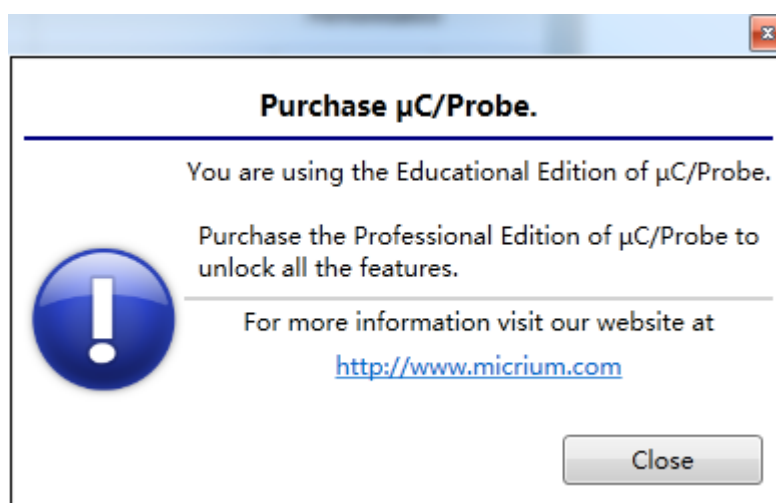


1.4.5 效果展示

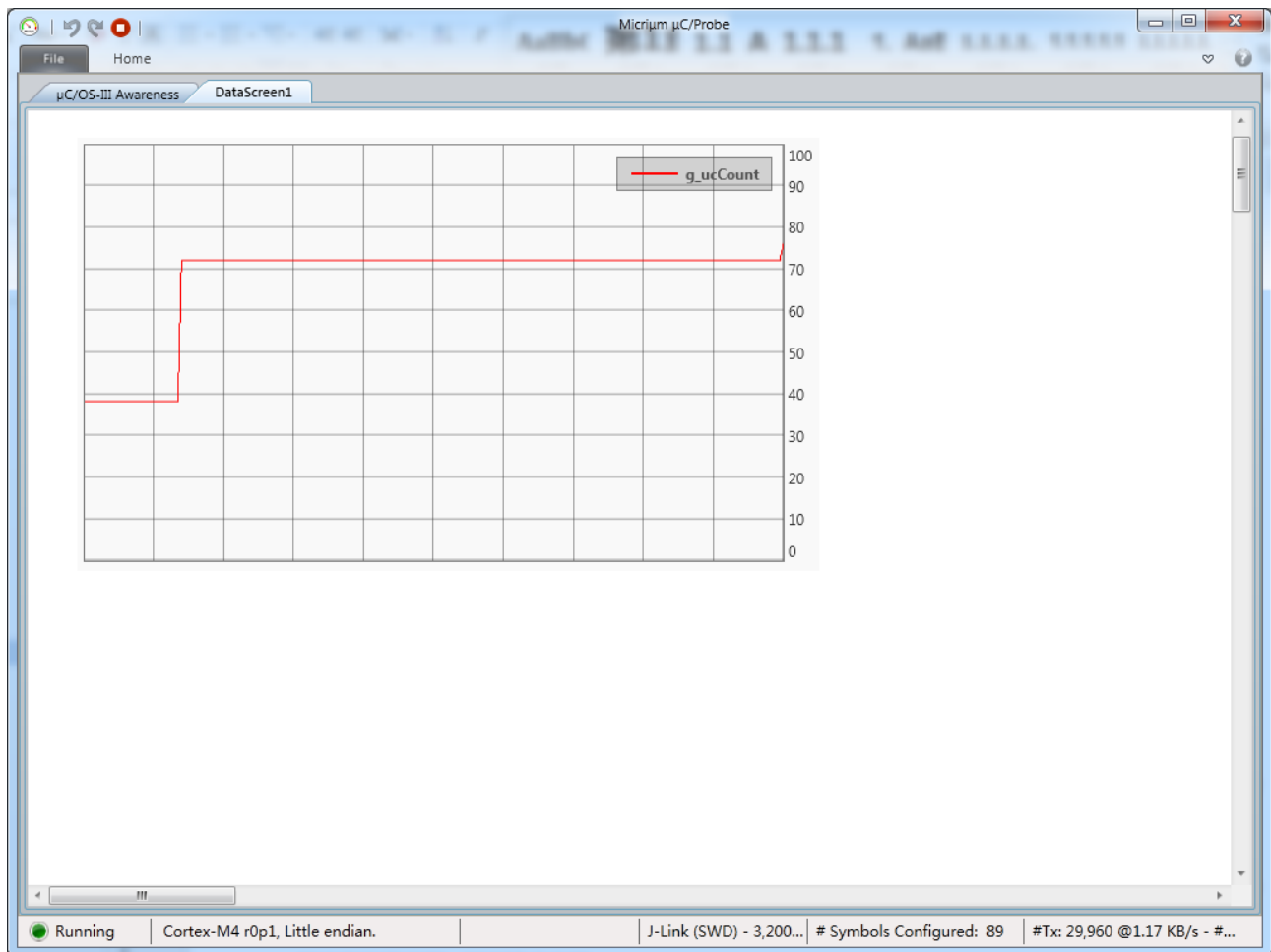
现在就可以点击左上角的“RUN”图标了：



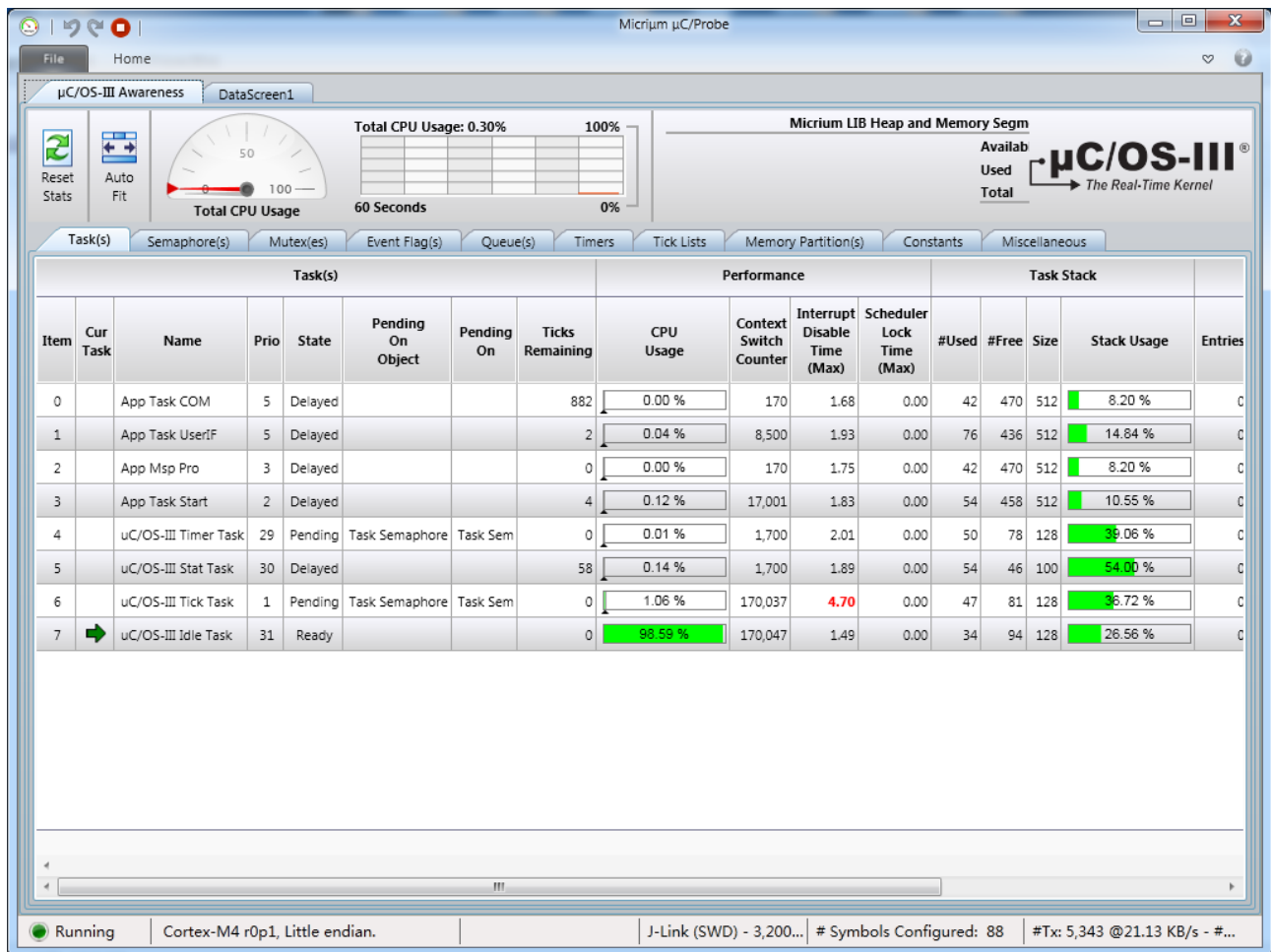
由于是教育版，点击“RUN”按钮后，会先弹出这个窗口：



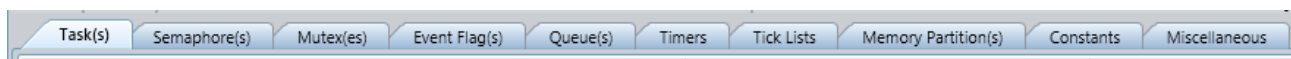
点击“Close”关闭按钮即可：



当前展示出来的就是全局变量 g_ucCount 的波形效果。在左上角，点击“uCOS-III Awareness”，就可以查看 uCOS-III 的信息了：



里面的这些选项都是可以选择查看的：



大家可以自己测试时一个一个点击，看看实际效果。由于我们使用的是评估版本，查看一段实际后就自动的退出了，弹出这个窗口：



需要再次查看，就继续点击运行即可。关于 IAR 使用 uC/Probe，就为大家讲解这么多，更多其它功能，

大家可以看官方手册进行学习，或者自己摸索使用也可以（由于是图形化的，点点就出效果了）。

1.5 配套例子

本期专题教程配套了三个例子，每个例子里面都是 IAR 和 MDK 两个版本。


- ◆ STM32-V4 开发板配套的例子是：V4-_uC-Probe 的使用。
- ◆ STM32-V5 开发板配套的例子是：V5-_uC-Probe 的使用。
- ◆ STM32-V6 开发板配套的例子是：V6-_uC-Probe 的使用。

1.6 官方手册

针对 uC/Probe，官方有两个手册，一个是用户手册，另一个是目标端手册，即板子端芯片需要移植的程序。安装了 uC/Probe 后，点击左上角的 File 选项就看到了：



下面是用户手册：



User's Manual

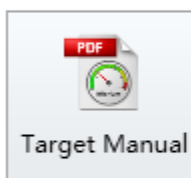
User's Manual

This document describes everything related to this Windows application including:

- Symbol Browser.
- Communication Settings.
- Workspace Explorer.
- Virtual Controls and Indicators Toolbox.
- Layout Design Tools.
- Associating Symbols to Virtual Controls and Indicators.
- μ C/OS-III Kernel Awareness Screen.

下面是目标端手册：

Target Manual



This document describes everything related to the C code that resides in the embedded target including:

- Configuring the μ C/Probe target module.
- Initializing the μ C/Probe target module.
- Building the μ C/Probe target module.
- Porting the μ C/Probe target module.
- μ C/Probe target module API.
- Symbol files supported by μ C/Probe.
- Terminal Window Control.
- μ C/Trace Triggers Control.

NOTE:

Special embedded-target-resident-code for communication purposes is only required if your only communication interface available is RS-232, TCP/IP or USB.

You do NOT require any special embedded-target-resident-code if your setup includes one or more of the following:

- J-Link.
- IAR Systems Embedded Workbench.
- Eclipse-based IDE.

更多 μ C/Probe 的操作说明，大家看这两个手册即可。

1.7 总结

首次使用这个软件，需要稍花点时间熟悉这个软件的使用方法，后面熟练了就好用了。另外就是注意这个软件的死机问题，有时候不好解决的话，可以尝试重新安装。



1.8 文档更新记录

版本	更改说明	作者	发布日期
V1.0	初版首发，制作于2017-10-17	白永斌 (Eric2013)	2017-10-24 (33页)