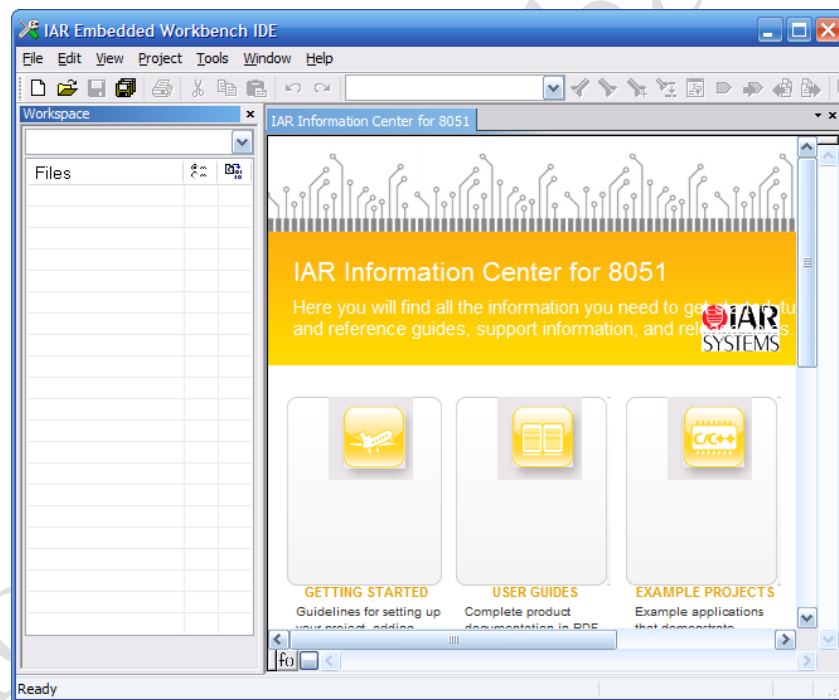




IAR for 8051 V8.10

安装和使用教程



Ghostyu
2013-3

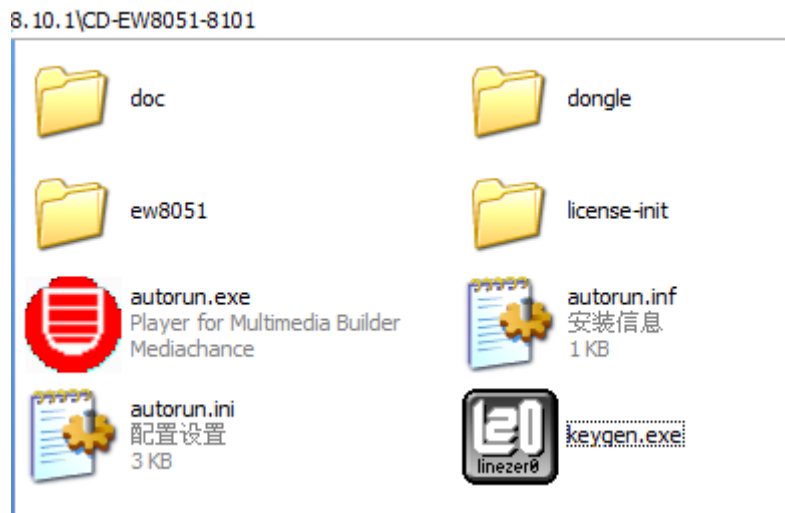


1 介绍

IAR for 8051 软件是开发 TI Z-Stack 协议栈应用程序的必备软件，所有程序的编译、仿真调试均需使用该软件，当前最新版的 Z-Stack 协议为 ZStack-CC2530-2.5.1a，配套 IAR 版本 V8.10

2 安装 IAR for 8051 V8.10 软件

程序安装包位于开发套件根目录下的 软件工具文件夹下，如下图



2.1 双击运行 autorun.exe，然后再跳出的画面中选择第二项，Install IAR Embedded Workbench



2.2 根据提示一路 next，到 Enter User Information 这一项，提示输入 license。



IAR Embedded Workbench for 8051 8.10.1

Enter User Information

Enter your name, the name of your company and your IAR Embedded Workbench for 8051 license number.

Name:

Company:

License#:

Can be found on the CD cover, or via e-mail registration

InstallShield

< Back Next > Cancel

此时，打开软件根目录下的 keygen.exe（请关闭杀毒软件，再解压此 IAR 软件包，杀毒软件会认为 keygen 为病毒）

IAR Generic Keygen by LineZero

Decoder & xor37h

0100 1000 1011 0011 1001 0100 1100 1011

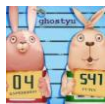
IAR Suite Keygen

Product:

HostID:

License number:

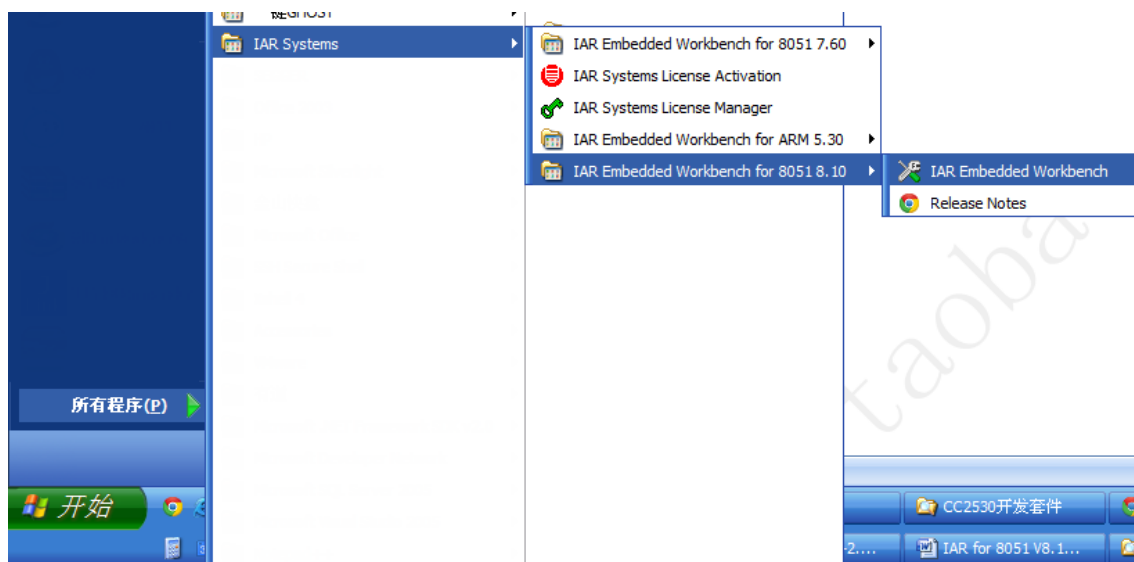
License key:



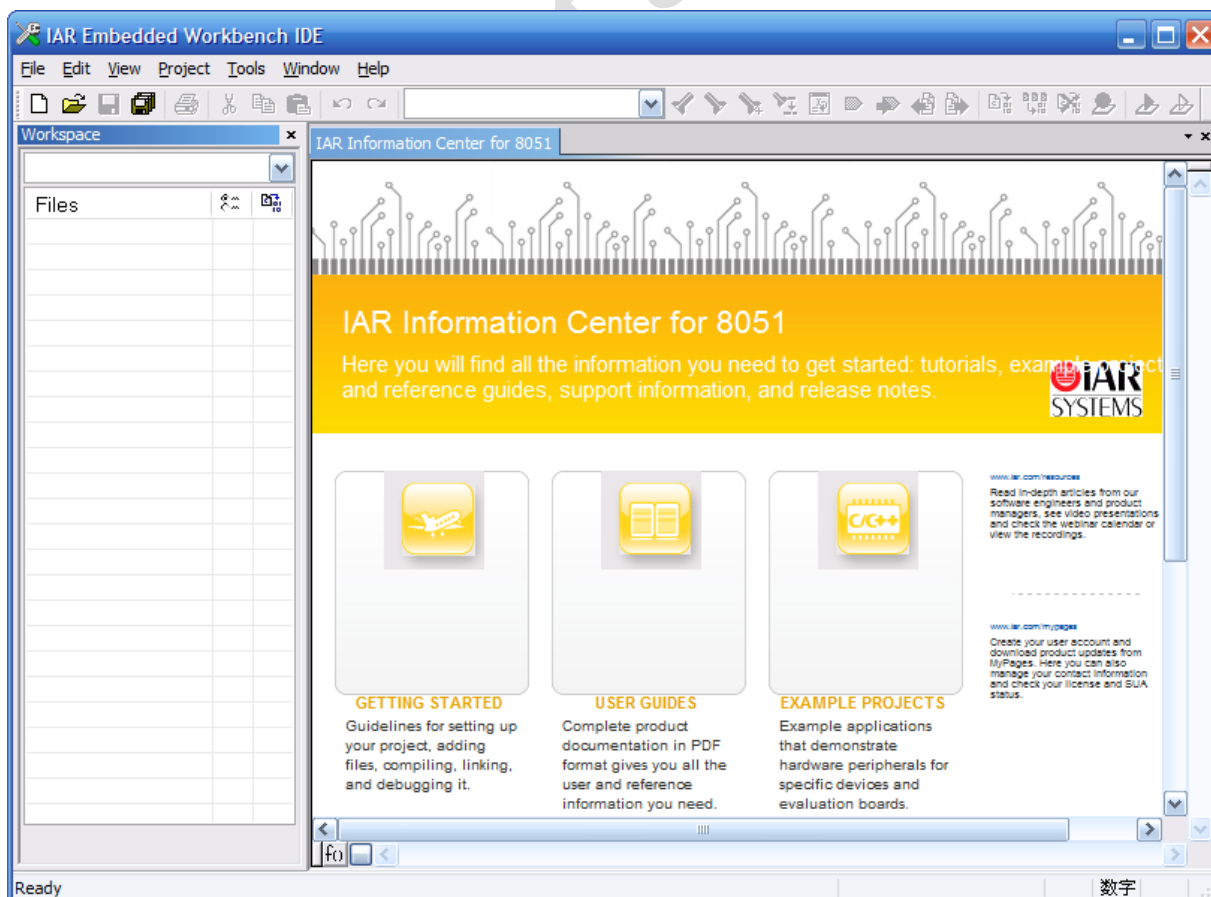
在 Product 列表框中选择 MCS-51 V7.50A（不要怀疑，用 V7.50A 的 license 即可），然后单击 Generate，会产生根据当前 PC 计算出来的 License number 和 License key，先将产生的 License number 复制到 IAR 安装向导中的 License#文本框内，单击 next，然后再将 License Key 复制到 IAR 安装向导的 License Key 文本框内。然后一路 next，直到软件安装结束。

3 运行 IAR for 8051 V8.10 软件

安装结束后在开始菜单中找到 IAR 软件，默认安装的位置如下图：



运行的 IAR 软件如下图：



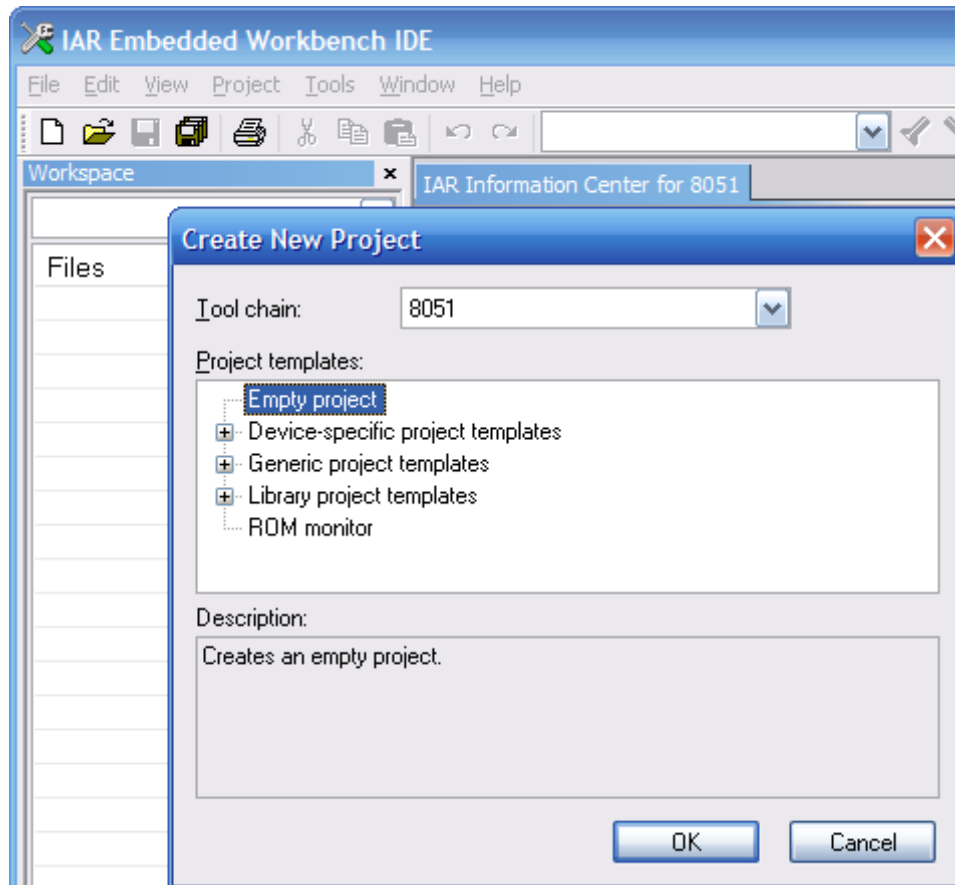


3 使用 IAR for 8051 V8.10 创建一个 CC2530 工程

在本节中，我们将使用 IAR 创建一个完整的软件开发环境。（这里使用 CC2530 为例，同样适用于 CC2540）

3.1 创建 project

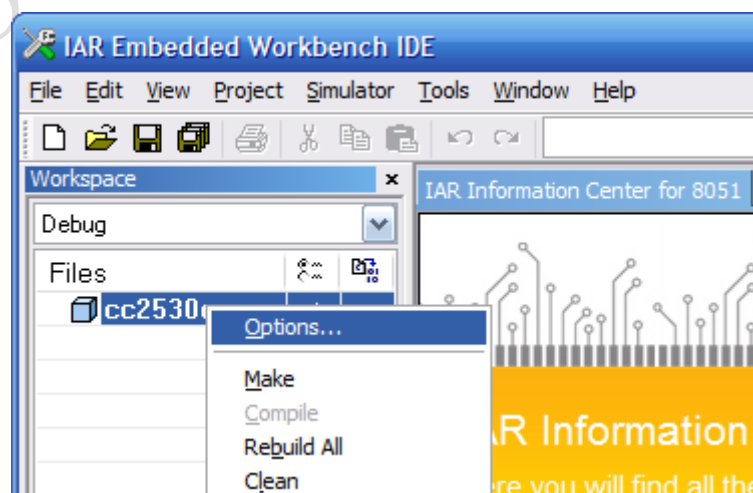
运行 IAR Embedded Workbench，单击菜单 Project→Create New Project，出现下列对话框：

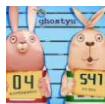


选择“Empty project”，单击 OK，然后会询问保存 project，选择一个合适的目录，然后填入合适的工程名，然后单击 OK

3.2 Project Options

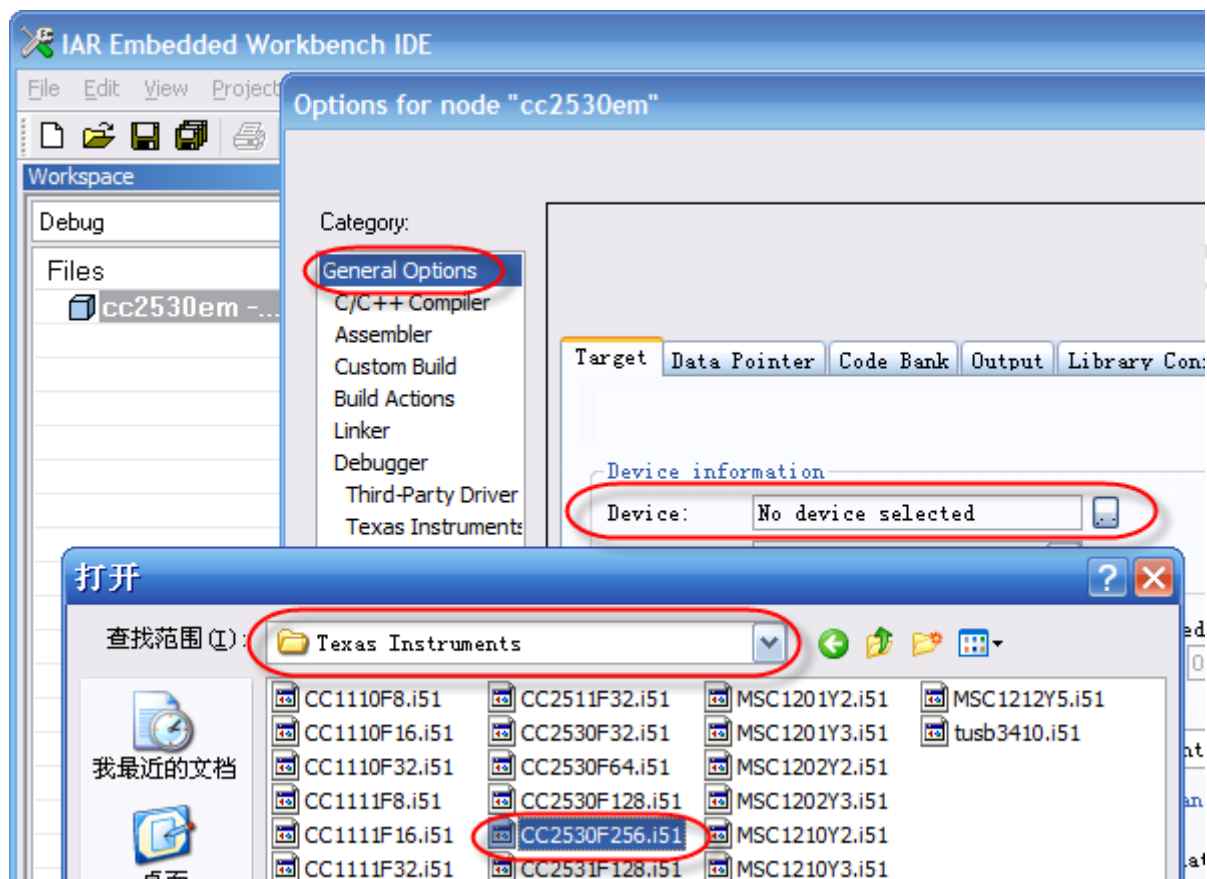
在左边的 Workspace 中右击保存的工程 Project。



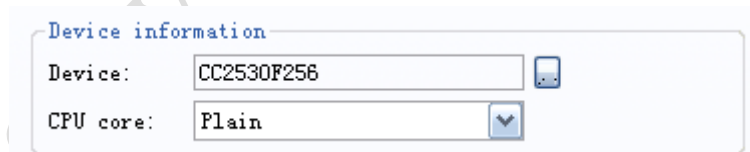


3.3 选择合适的器件

在出现的对话框中，第一件事情就是选择该 project 所使用的 Device，单击...按钮选择 Device



选择如上图中的 CC2530F256.i51，该文件位于 IAR 安装目录 C:\Program Files\IAR Systems\Embedded Workbench 6.0\8051\config\devices\Texas Instruments（该路径为默认安装路径）。选择完后回到 Device information 中会出现设备列表，如图



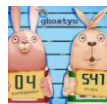
3.4 选择 Code 和 Memory Model

在 code 类型中有 Near 和 Banked 两项可选择

“Near”当不需要 Bank 支持是可以选择 Near，例如，你之需要访问 64K flash 空间的时候，不需要更多的 flash 空间，比如你使用的是 CC2530F32 或 CC2530F64，或者使用的 CC2530F256 但并不需要那么大的 flash 空间时，可以选择 Near。

“Banked”选择该项时标明你需要更多的空间能够仿真 CC253xF128 或者 CC253xF256 的整个 Flash 空间。默认 Near code model 中的 data model 是 Small，默认的 Banked, data model 为 Large, data model 决定编译器或者连接器如何使用 8051 的内存来存储变量，选择 small data model，变量典型的存储在 DATA 内存空间，如果使用 Large data model，变量存储在 XDATA 空间。在 CC2530 用户手册和 IAR 8051 编译器参考手册中会详细描述变量内存空间。

在这里，重要的事情是，8051 使用不同的指令来访问 various memory spaces 访问 IDATA，一般情况下，比仿真 XDATA 要快，但通常 XDATA 的空间会比 IDATA 大。



在 Z-Stack 协议栈中, 使用 large memory model 来支持 CC2530F256, 这样协议栈可以存储在 XDATA 区域, 以上设置结束后, 如下图所示。

Device information

Device: CC2530F256

CPU core: Plain

Code model: Banked

Do not use extended stack ☒ Extended stack at 0x002000 ☐

Data model: Large

Calling convention: XDATA stack reentrant

Number of virtual: 8

Location for constants and strings: RAM memory ☒ ROM mapped as data ☐ CODE memory ☐

在 Banked code model 中, 有一些额外的选项需要注意, 选择 Code Bank tab, 如下图, CC2530 使用 7 个 code banks, 为了访问整个 256K 的 Flash 空间, Number of 必须设置为 0x07, Register 0x9F 是 CC2530 的 FMAP 寄存器, 用来控制当前那个 code bank 映射到 8051 的地址空间, 第三个 Register 未使用, 最好设置 0xFF。

Options for node "cc2530em"

Category:

- General Options
- C/C++ Compiler
- Assembler
- Custom Build
- Build Actions
- Linker
- Debugger
- Third-Party Driver
- Texas Instruments
- FS2 System Navigator
- Infineon
- Nordic Semiconductor
- ROM-Monitor
- Analog Devices
- Silabs
- Simulator

Target Data Pointer Code Bank Output Library Configuration

Number of: 0x07

Register: 0x9F

Register: 0xFF

Bank start: 0x8000

Bank end: 0xFFFF

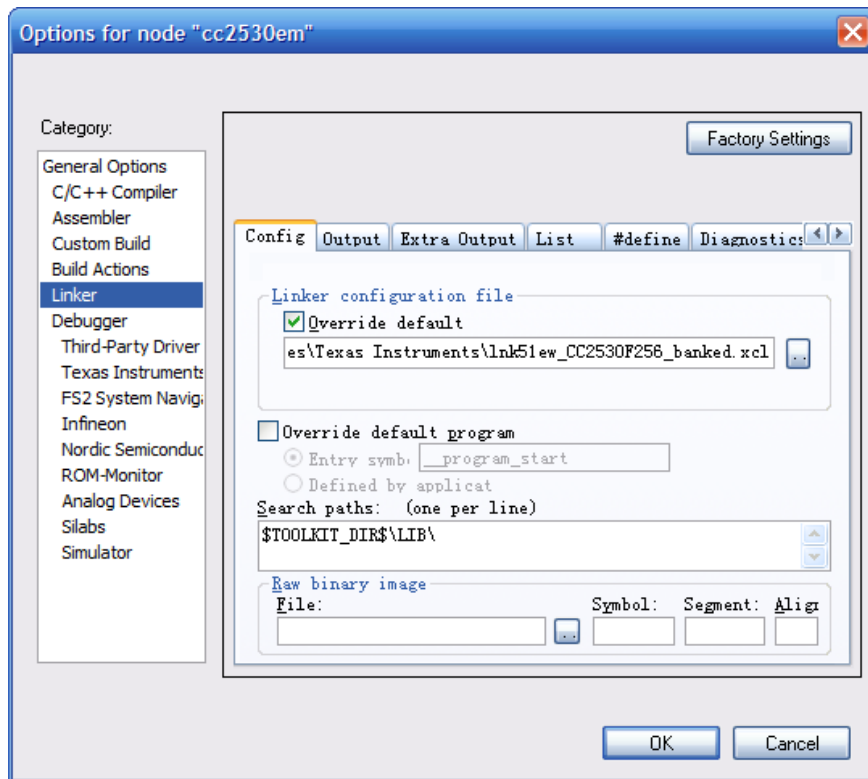
OK Cancel

3.5 配置 Linker 链接器



下一步需要配置 IDE 怎样使用 Linker 来链接程序代码。

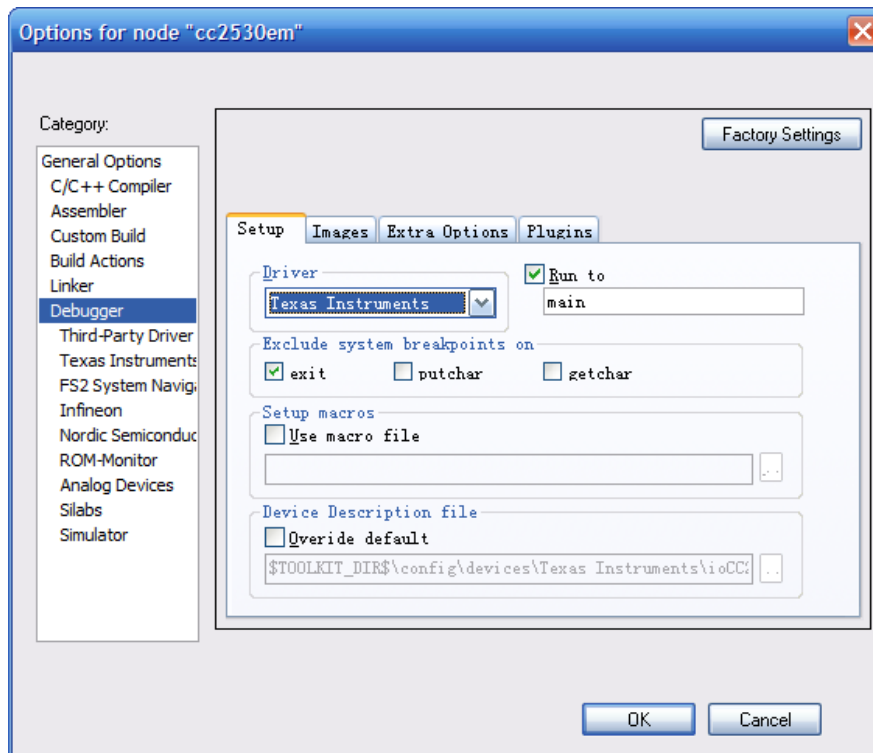
在左边的选项中选择 Linker，并在右边的选项卡中选择 Config 一页，在 Linker Command file 中复选 Override default，例如，我们选择 lnk51ew_CC2530F256_banked.xcl，banked 表示使用 banked code model。



默认路径为：\$TOOLKIT_DIR\$\config\devices\Texas Instruments\lnk51ew_CC2530F256_banked.xcl

3.6 配置仿真器调试

最后，在 Debugger 选项中，选择 Texas Instruments 为 Driver。



3.7 编写一个简单的应用程序

在上述各节中已经设置好了开发环境，下面我们来编写一个简单的 CC2530 应用程序。

新建一个文件，选择菜单 file->new->file, 并保存为 main.c, 代码如下

```
#include <ioCC2530.h>
```

```
int main()
```

```
{
```

```
    //设置 P1.0 为输出
```

```
    P1DIR = 0x01;
```

```
    //翻转 P1.0 状态
```

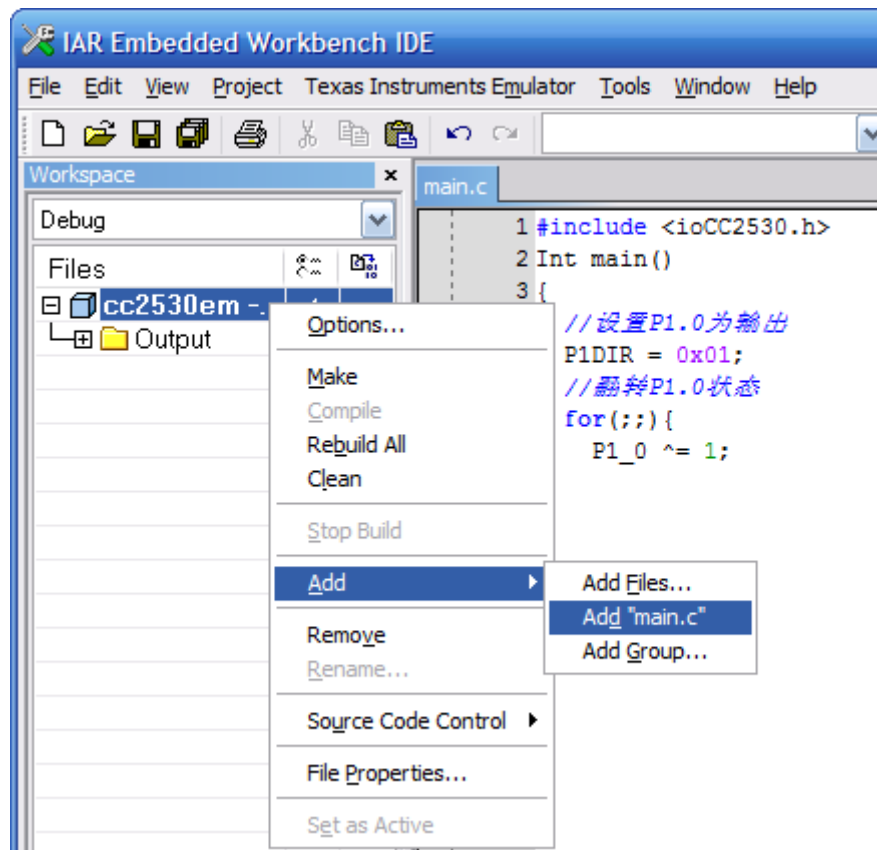
```
    for(;;) {
```

```
        P1_0 ^= 1;
```

```
    }
```

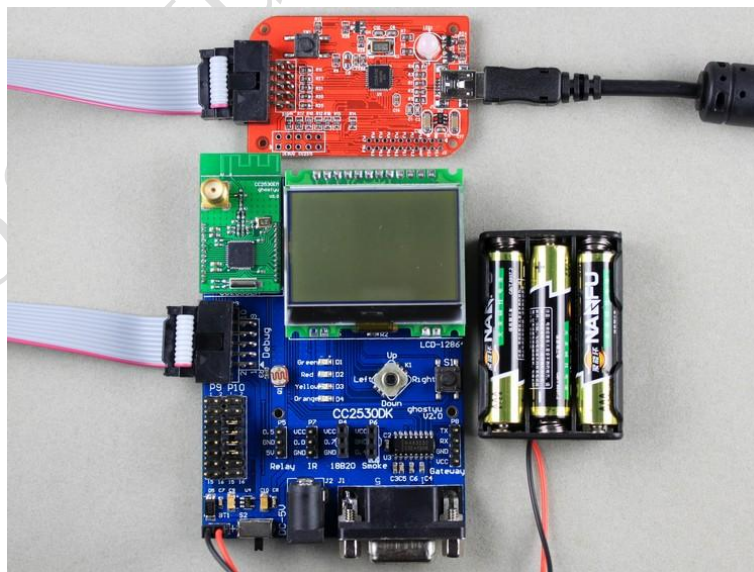
```
}
```


上述代码会快速的触发 P1.0, P1.0 对应开发板中的 LED1, 动作非常迅速
然后添加该文件到工程中, 如下图:

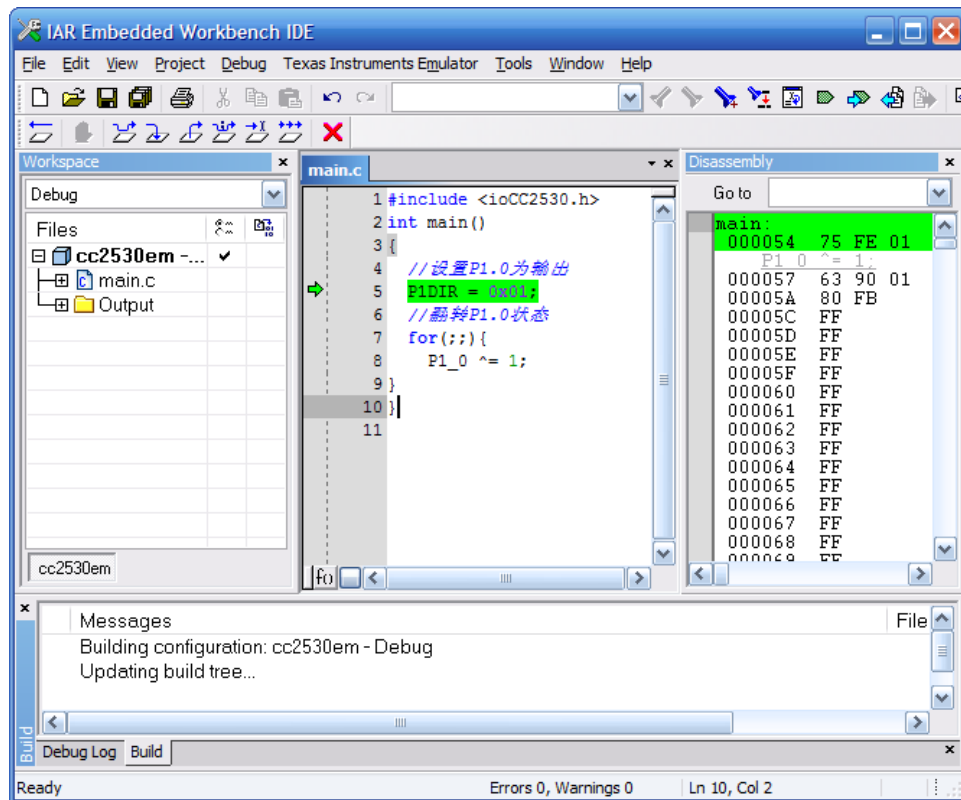


3.8 编译和调试

选择 Project -> Make 或者按 F7；来编译工程，第一次编译时，IDE 会提醒保存 workspace，选择合适路径保存，编译结束后，就可以来下载程序和仿真调试了，如果有语法错误，根据提示修改。连接仿真器到目标板，并且给目标板上电。连接如下图：



然后 选择 Project->Download and Debug 或者单击快捷图标 ，菜单，此时应用程序会被下载到目标芯片中，这是可以单步执行代码了。



下图所示为执行命令



- 【1】: 回到 main 函数起点。
- 【2】: 程序运行的停止命令，单击后程序暂停执行。
- 【3】: 单步，不进入下级函数。
- 【4】: 单步，进入下级函数。
- 【5】: 单步，跳出该级目录。
- 【6】: 运行下一行程序。
- 【7】: 运行到鼠标光标处停止。
- 【8】: 全速运行。
- 【9】: 退出仿真调试画面。

上述程序运行时，LED1 会非常快速的闪烁，由于速度非常快，感觉 LED1 是一直点亮的。

至此我们已经完成了 IAR 第一个程序的环境设置和编写调试工作。