

样例程序：ADC 模块使用

1.简介

这个文档的目的是帮助初上手的兄弟熟悉下DAVE-Tasking-memtool的流程，内容浅显，图文并茂，高手绕路

本程序实现功能如下：

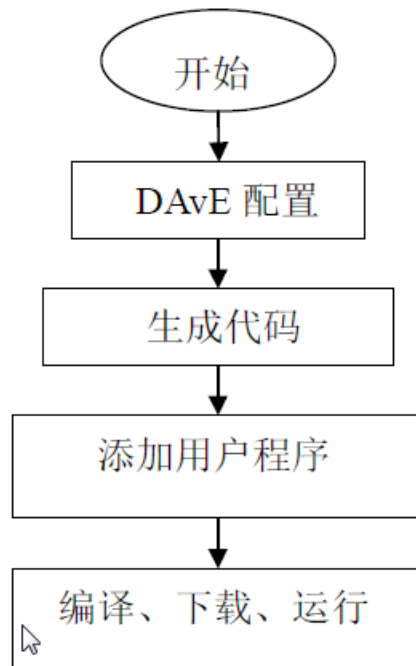
使用ADC0，请求源0软件产生转换请求，读出当前电位计的电压值并以此作为延迟时间。旋转电位计可以明显看到LED 灯以不同频率闪烁。

使用到的模块：ADC 和 IO

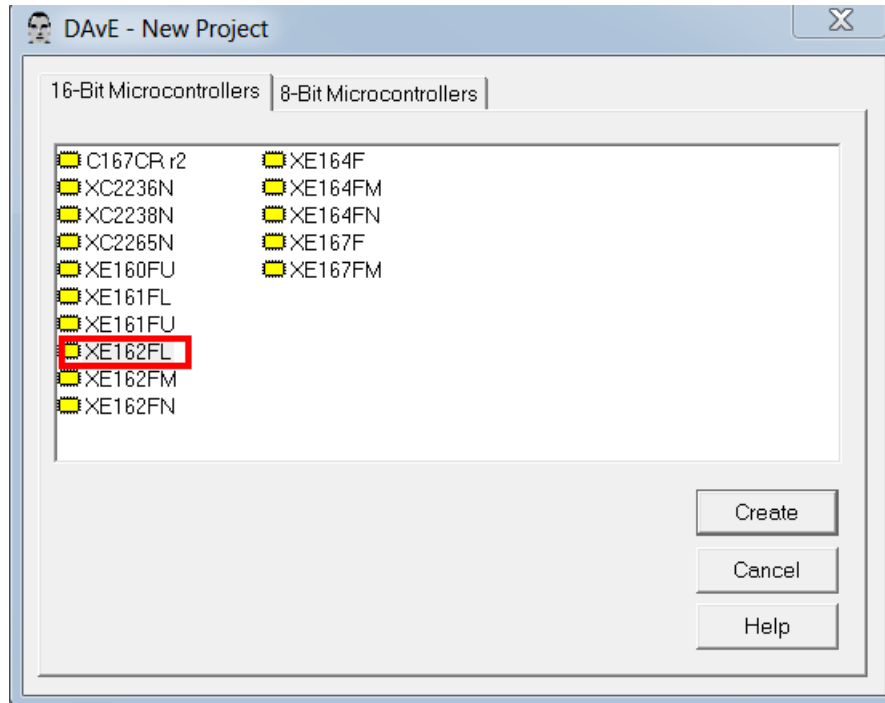
2 ADC 内核功能的特性如下所述：

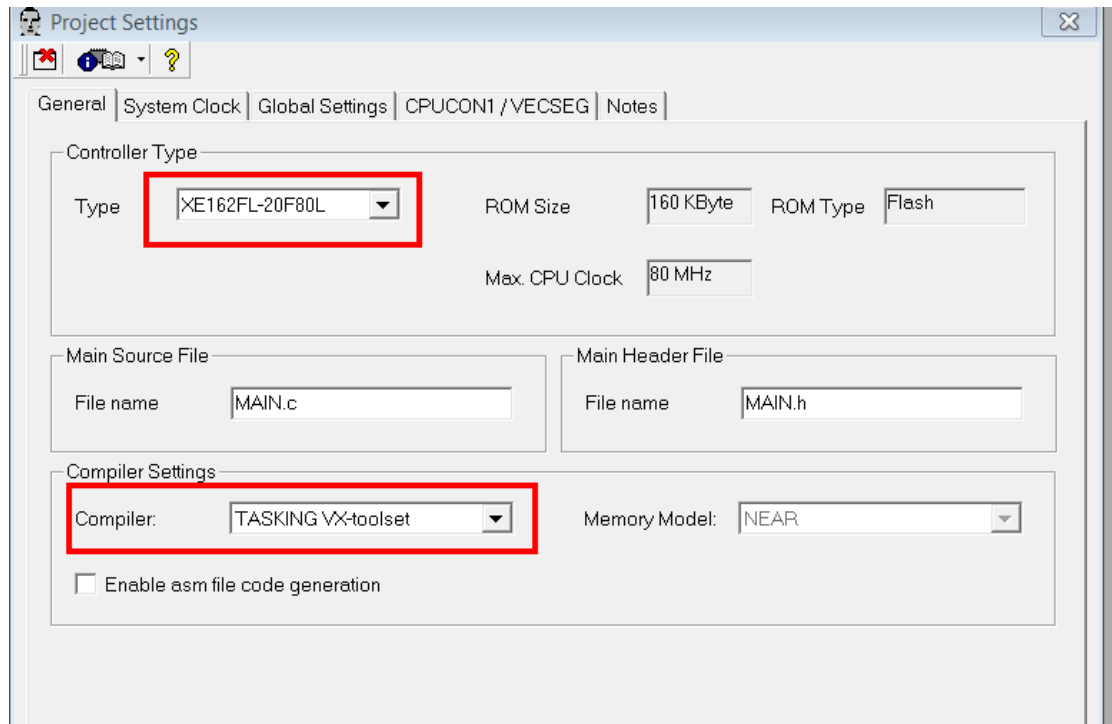
- 输入电压的范围可从 0V 到模拟供电电压 ($V_{DDPA}=3.0V$ 至 5.0V)
- 对于每个通道来说，标准 (V_{AREF}) 和备用 (CH0) 参考电压源均可选，以支持比例测量和不同的信号尺度
- 多达 19 个模拟输入通道 (指 162FL, 160U 8 个)
- 外部模拟复用器控制，包括可调整的采样时间和扫描支持。
- 转换速度和采样时间均可调整，以适应不同的传感器和参考源
- 转换时间低于 $1\ \mu s$ (由结果带宽和采样时间决定)
- 灵活的多源选择和仲裁
 - 单通道转换 (单个或重复的)
 - 可配置的自动扫描转换 (单个或重复的)
 - 可编程设置的仲裁转换序列 (单个或重复的)
 - 由软件、定时器事件或外部事件触发的转换
 - 用于最大吞吐量的等待启动模式或用于降低转换延迟的取消插入重启模式
- 强大的结果处理
 - 可选择的结果带宽 (8/10/12)
 - 8 个独立结果寄存器，并可组合创建结果 FIFO
 - 对于可编程设置的边界值执行可配置的极限检查
 - 通过添加可选择数量的转换结果，以降低数据速率
- 基于可选择事件的灵活的中断产生 (PEC 支持)
- 内建的安全特性
 - 通过可编程设置的默认级别检测断线
 - 复用器测试模式可验证信号通路完整性
- 挂起和省电模式的支持

3.操作流程

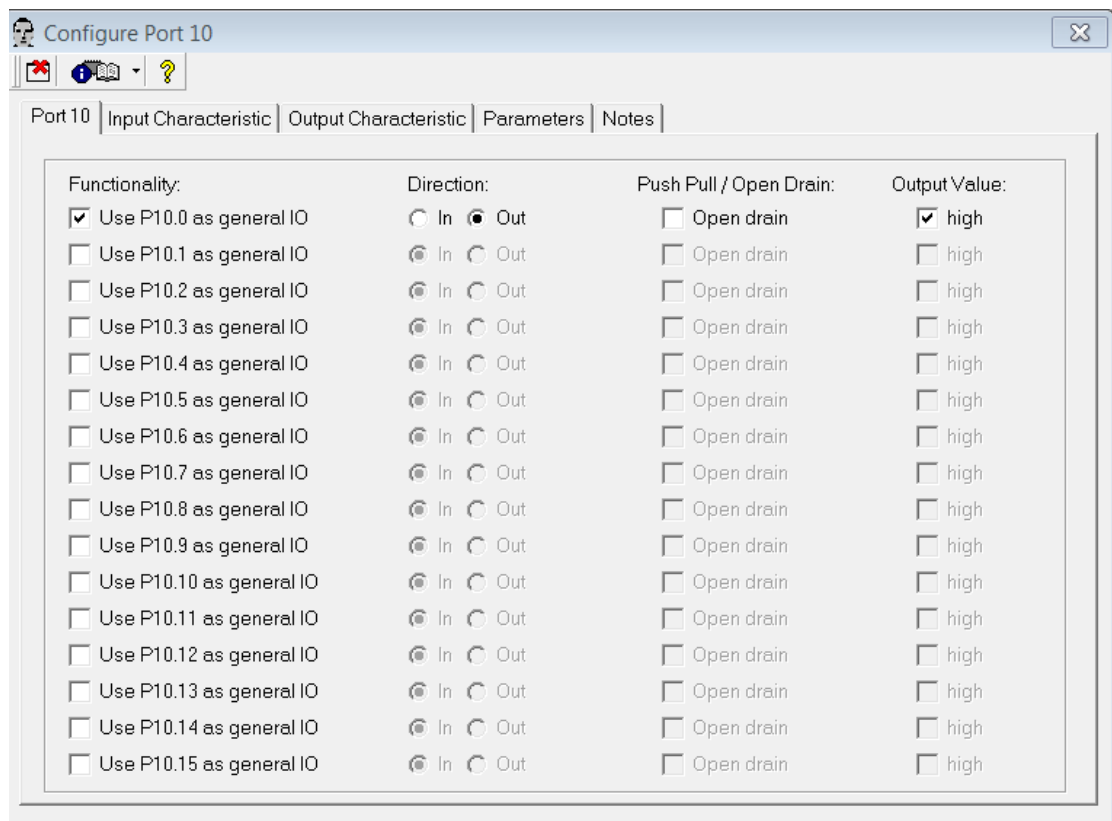


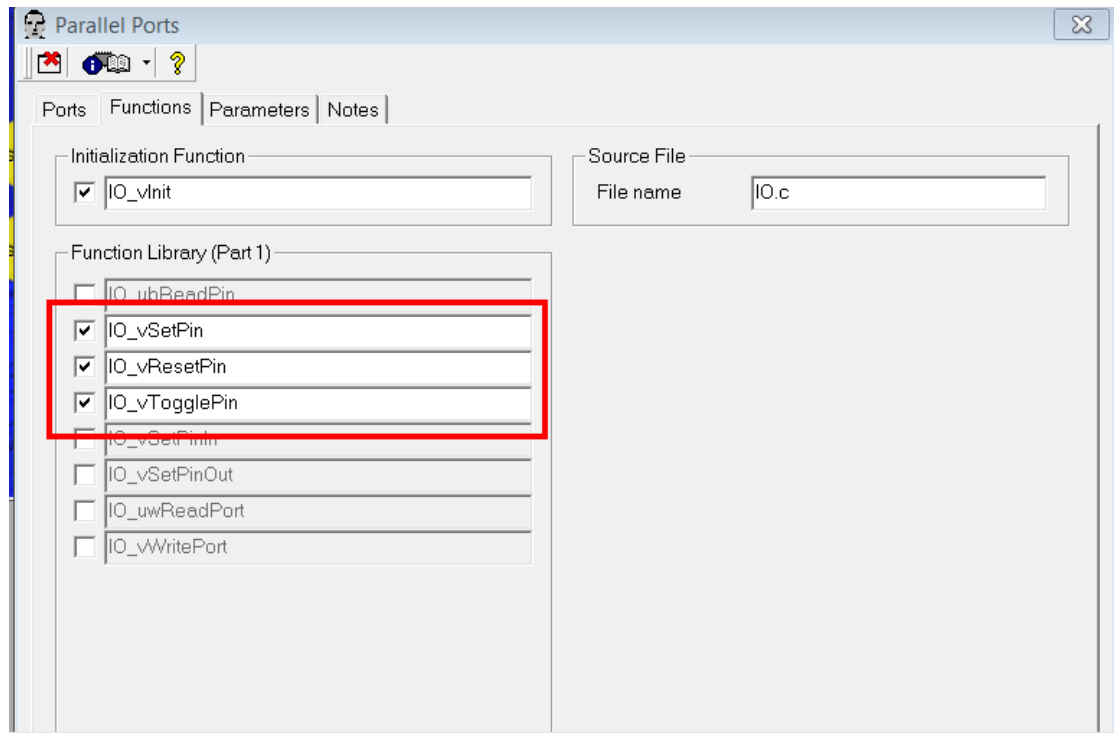
4.DAVE 配置



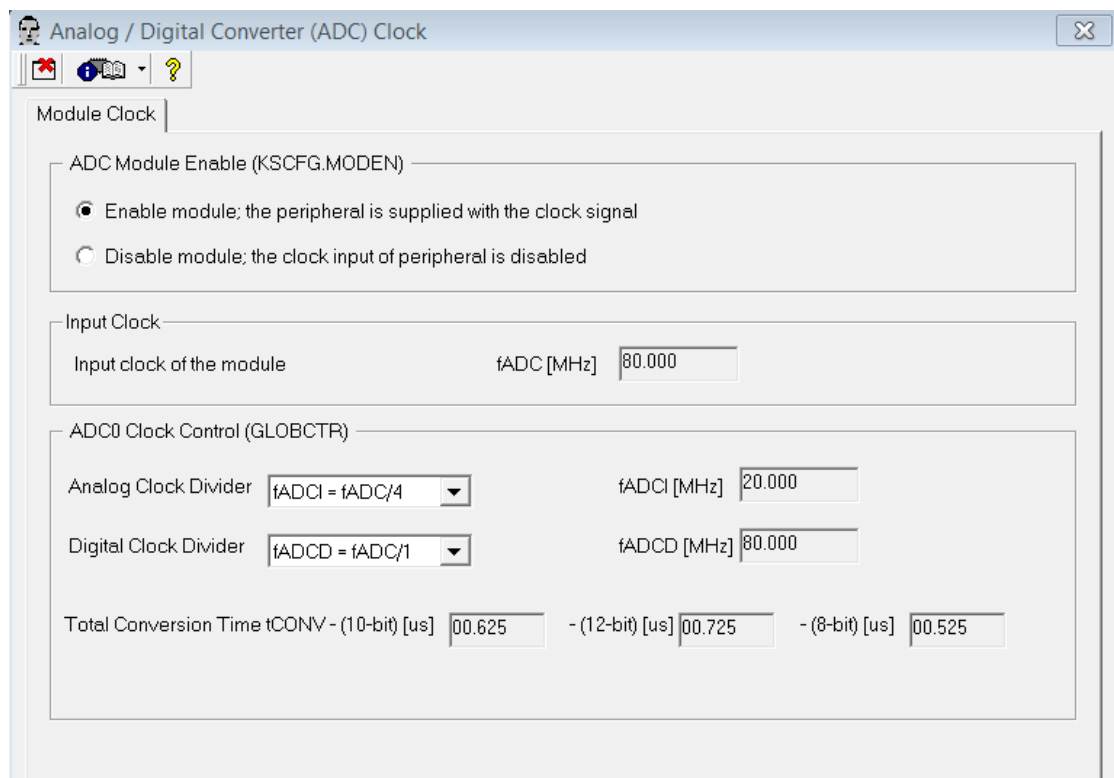


设置 I/O 口先

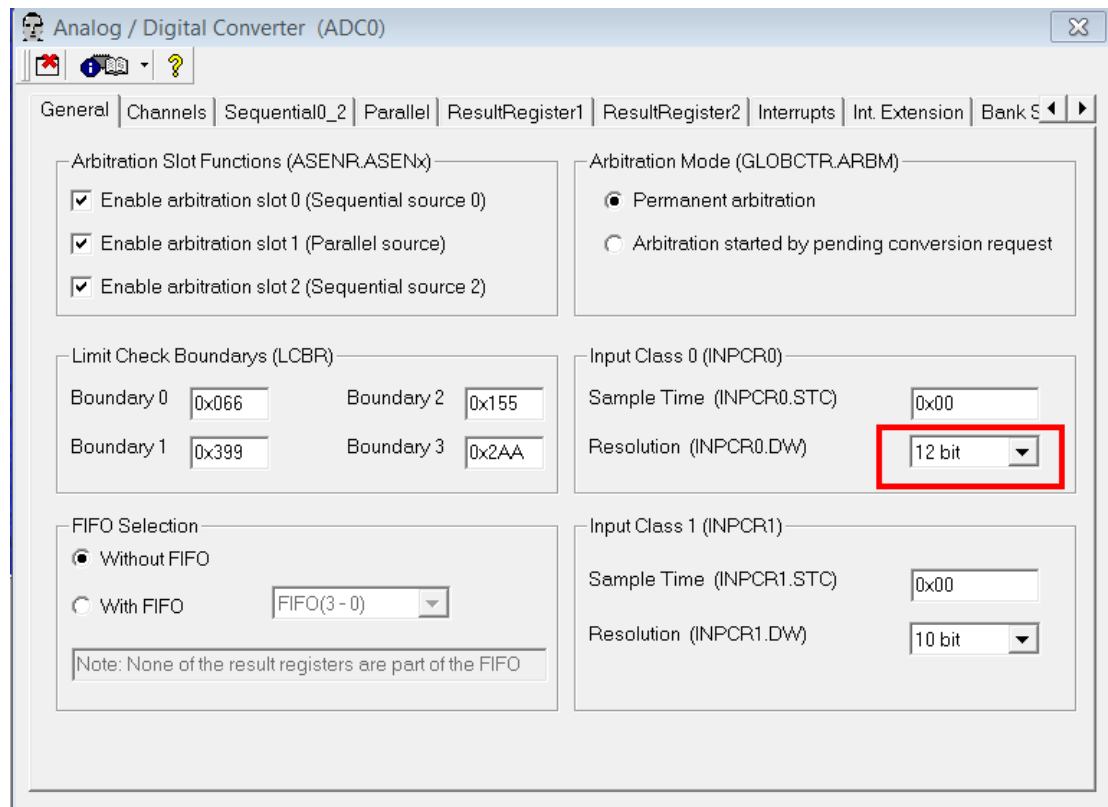




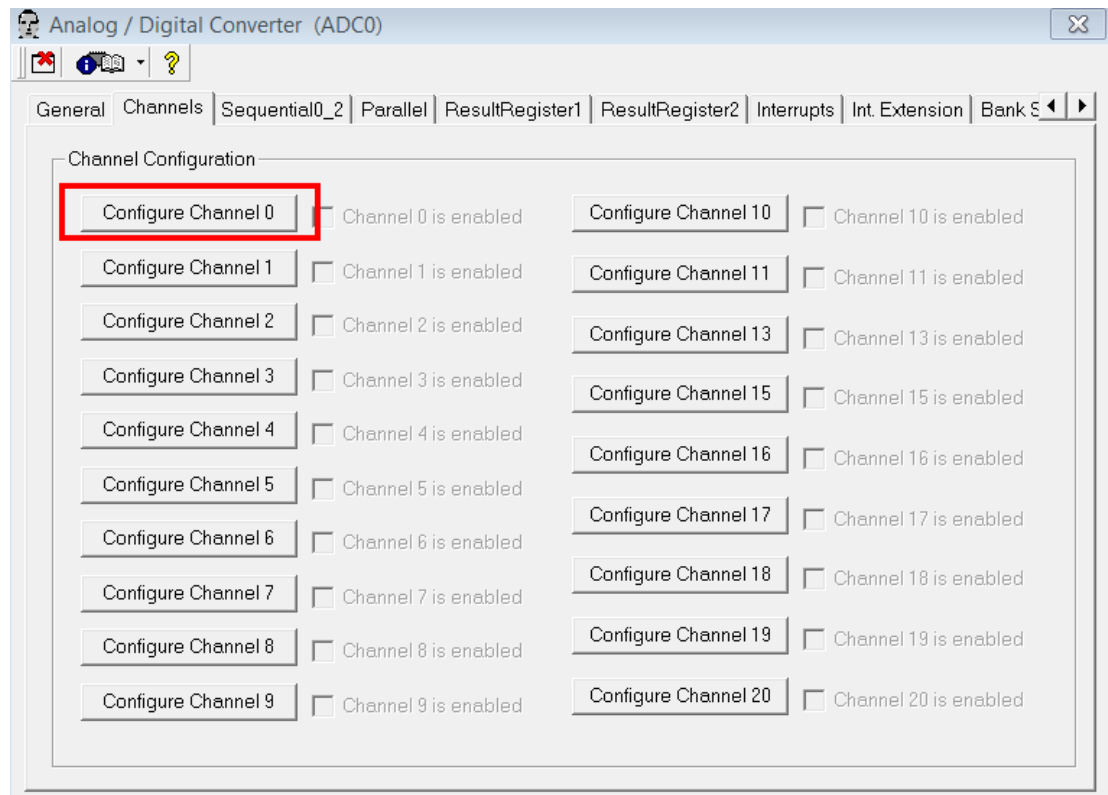
点开 ADC clock

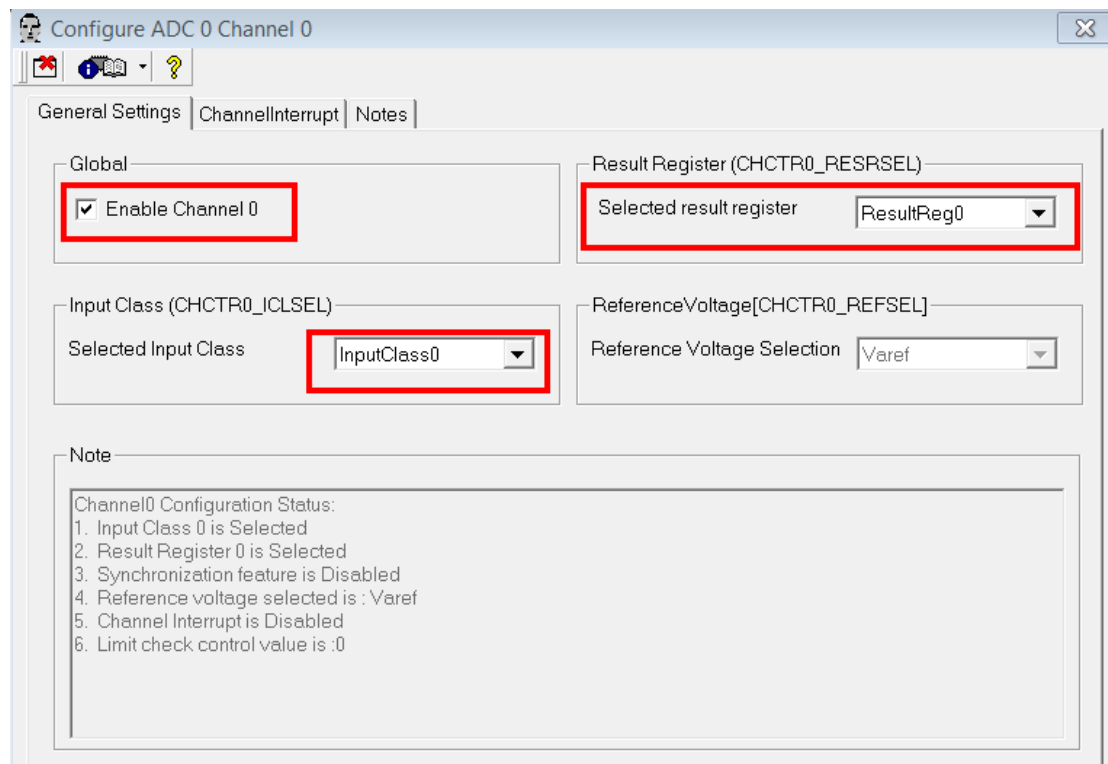


点击 ADC0

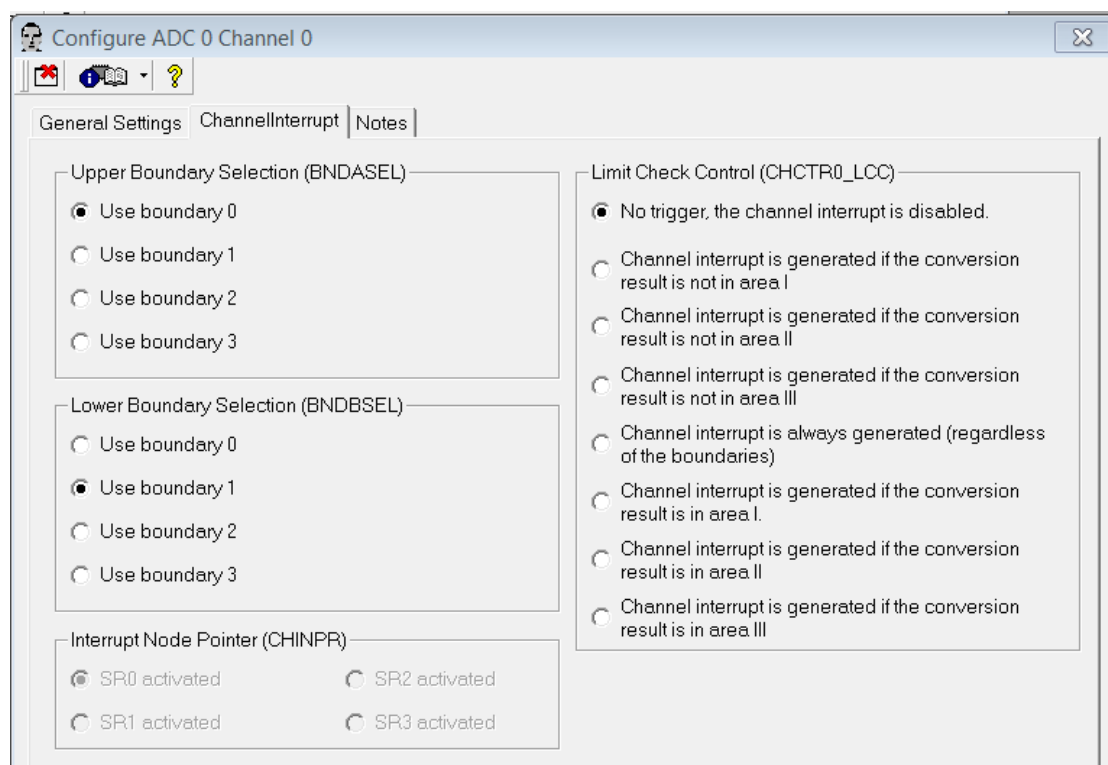


点击 Channel

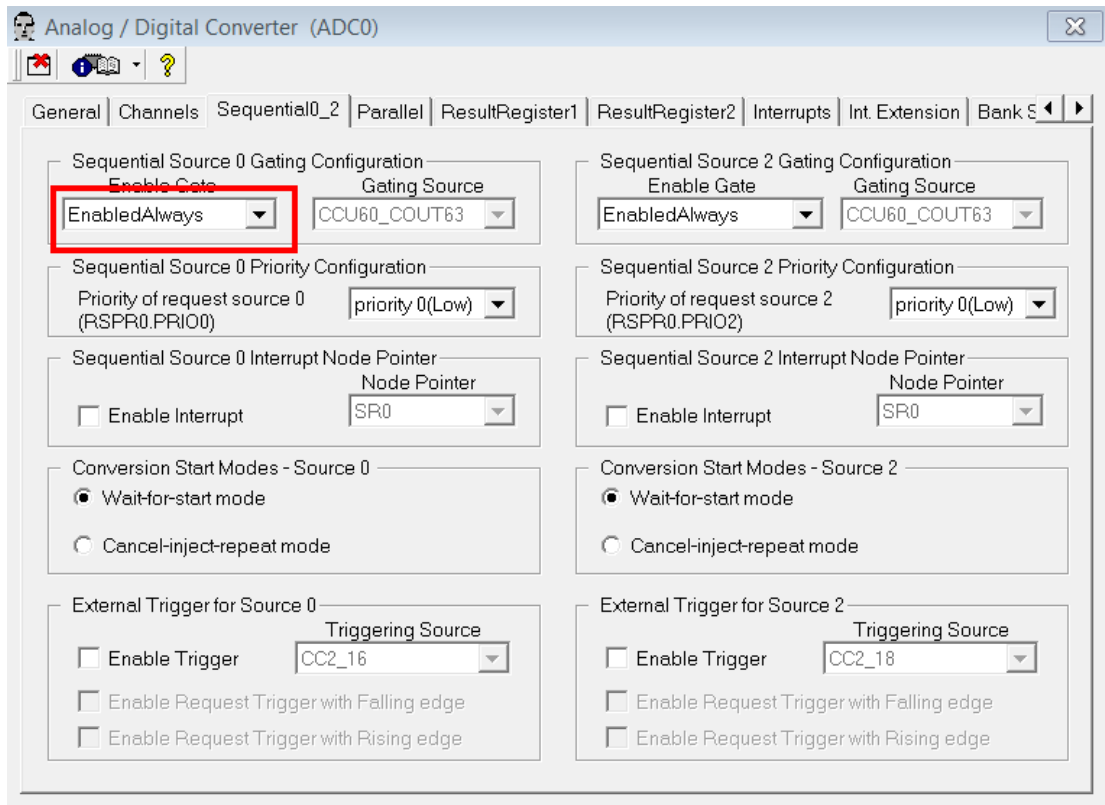




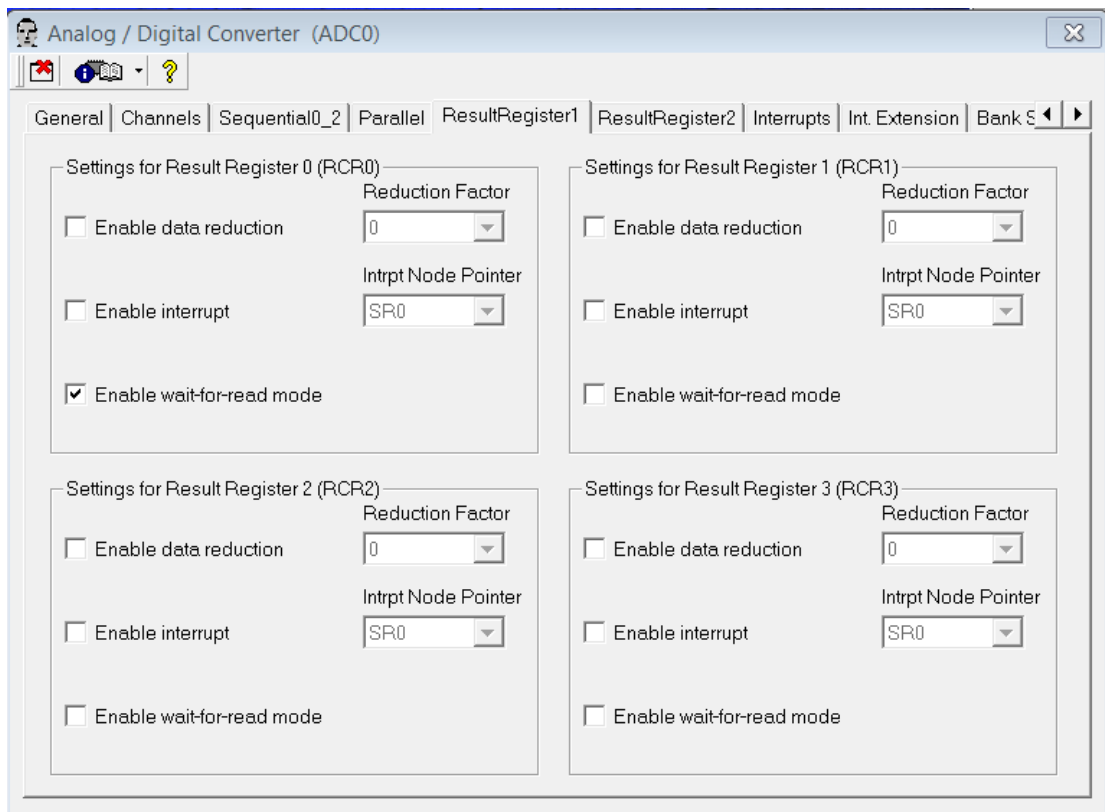
不设边界



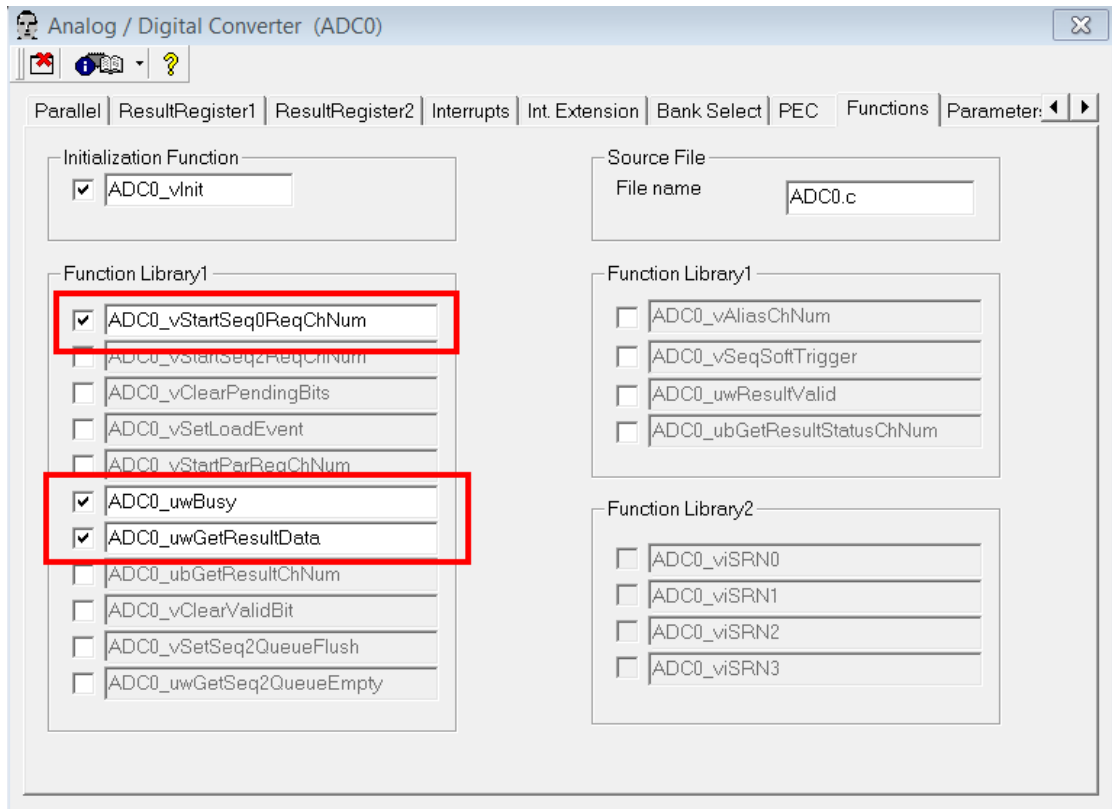
请求源 0 配置



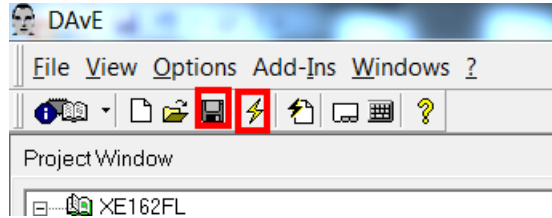
结果寄存器配置



选择需要 DAVE 生成的函数

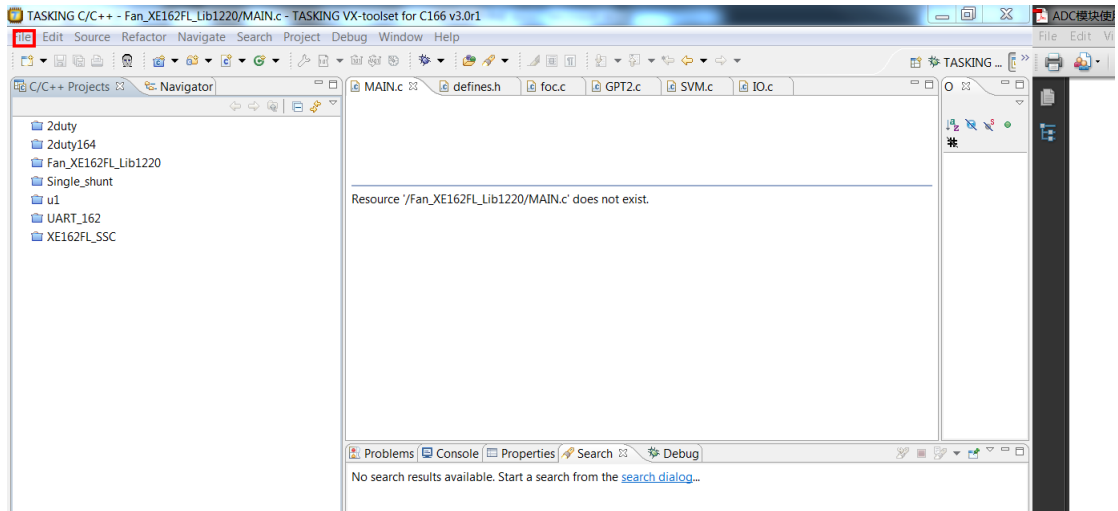


5. 利用 DAVE 生成代码

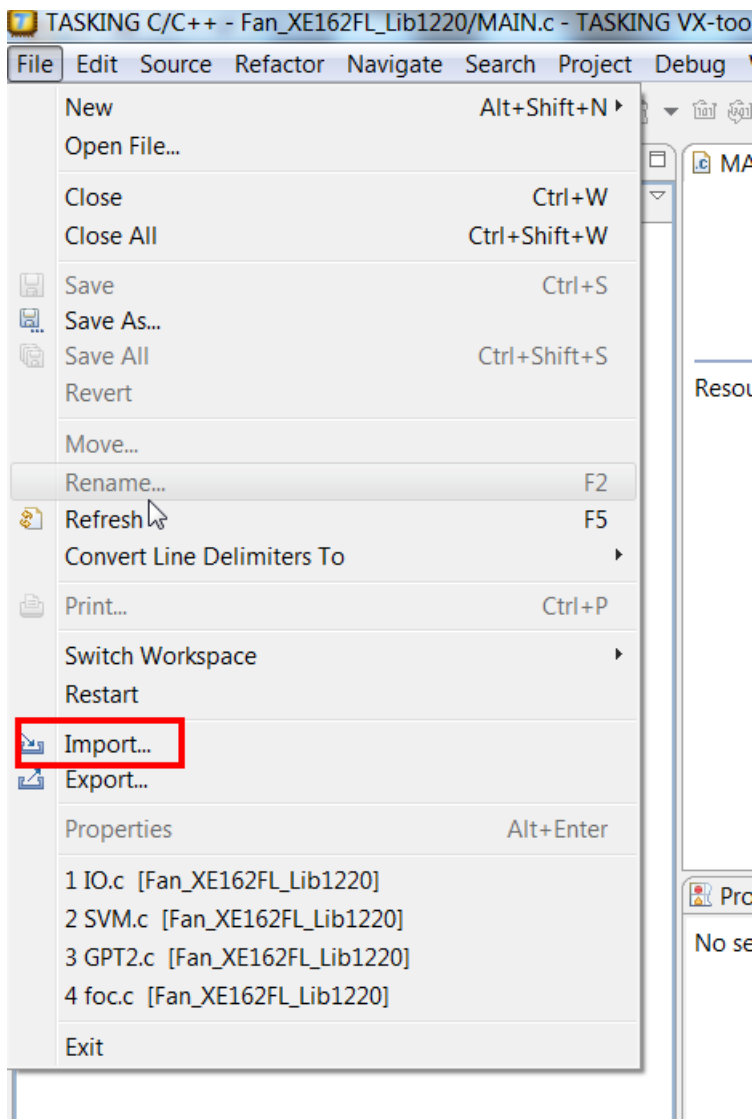


点击 左图标保存project，点击右图标DAVE 自动生成代码，生成的代码即包括前面所选择的函数。

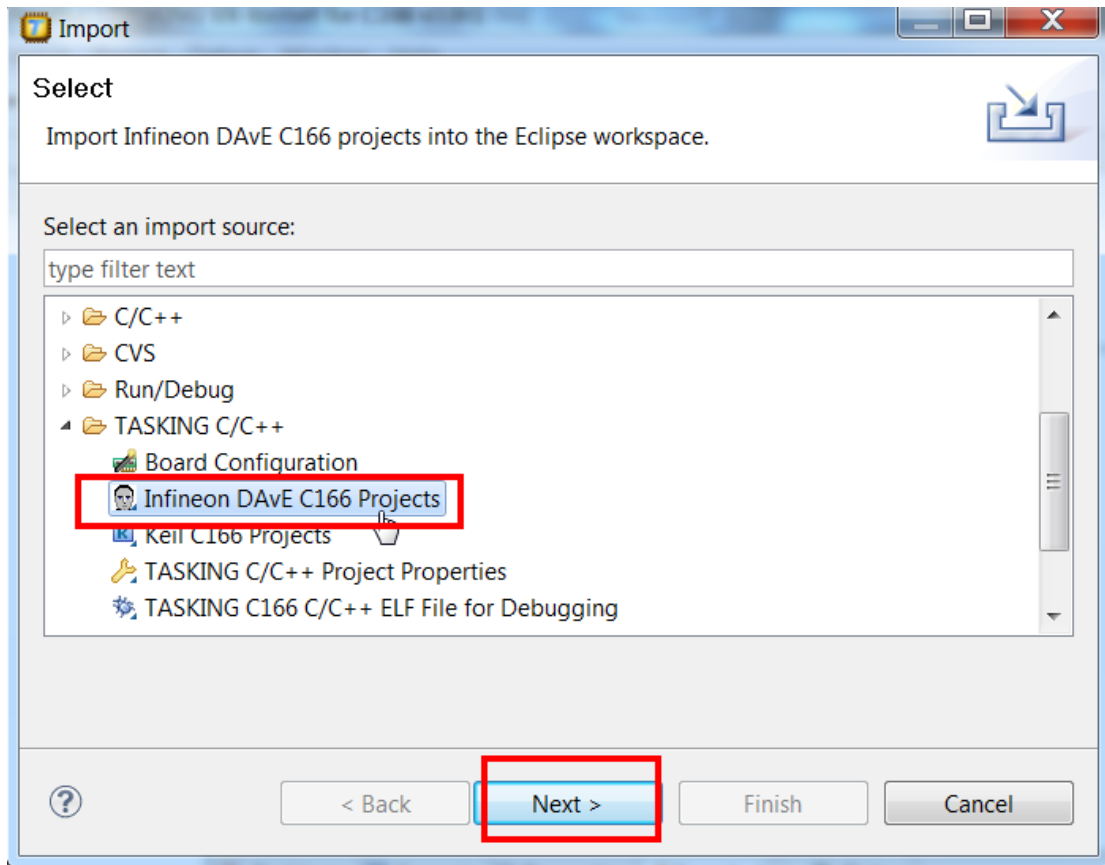
6 自动代码生成后，打开 TASKING VX-tool

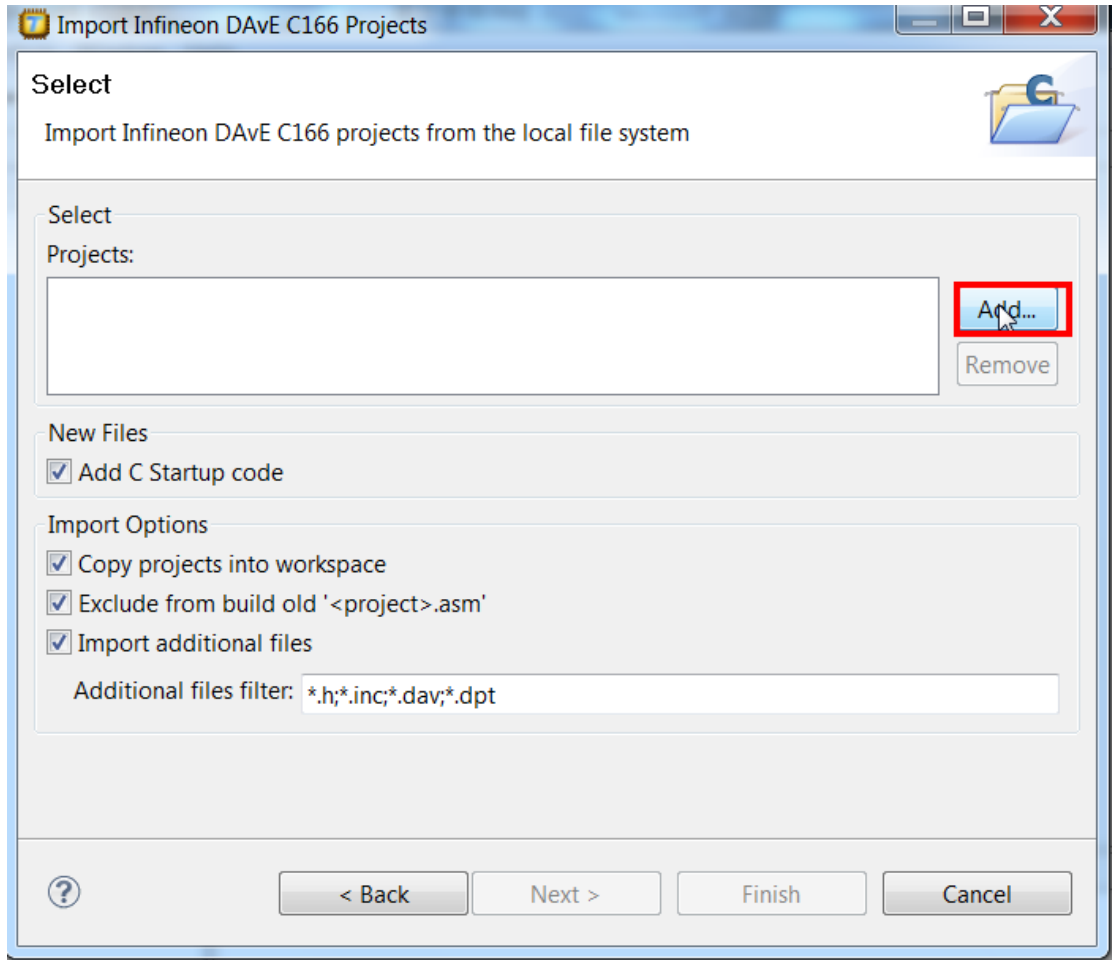


打开 File

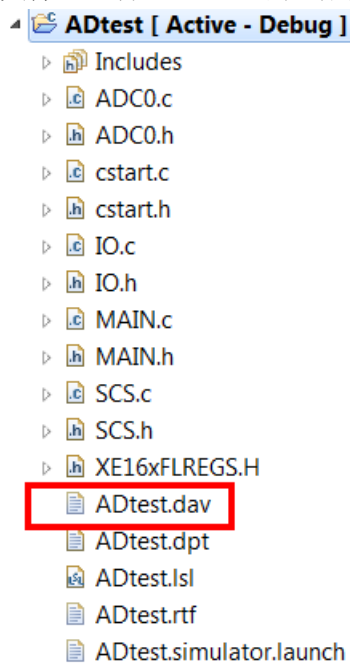


指定刚才生成的 DAVE 文件

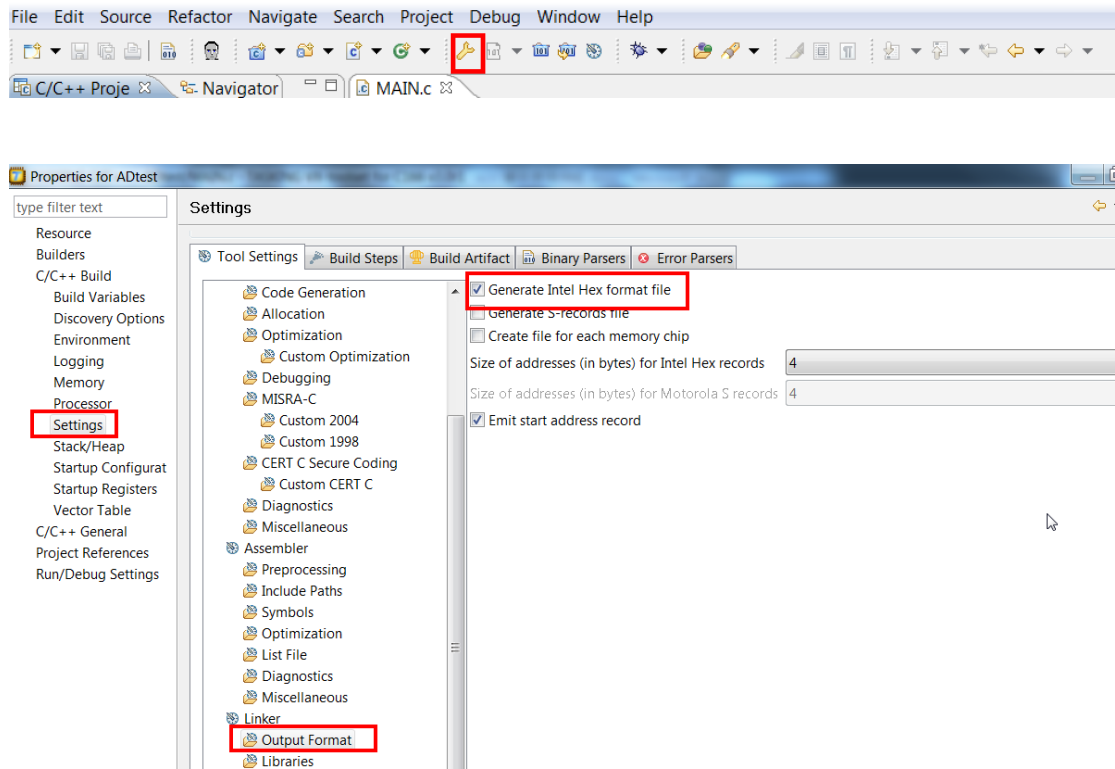




关掉刚才的 DAVE。直接从下图双击打开 DAVE，这样有修改可以直接覆盖，原来的 DAVE 就关掉吧。切记：DAVE 的生成文件是与自身.DAV 文件在同一目录下



选择项目属性



Build 以后在文件夹 debug 目录下生成 HEX

6.2 添加用户代码 (XE162FL与XE160U添加代码基本相同)

在main 函数中添加下列代码 (在main 函数的末尾处)

```
// USER CODE BEGIN (Main,2)
```

```
uword i,j,uwADCResult;
```

```
// USER CODE END
```

```
MAIN_vInit();
```

```
while(1)
```

```
{
```

```
// USER CODE BEGIN (Main,4)
```

```
ADC0_vStartSeq0ReqChNum(0, 0, 0,0);
```

```
while(ADC0_uwBusy());
```

```
uwADCResult=ADC0_uwGetResultData(RESULT_REG_0);
```

```
IO_vTogglePin(IO_P10_0);
```

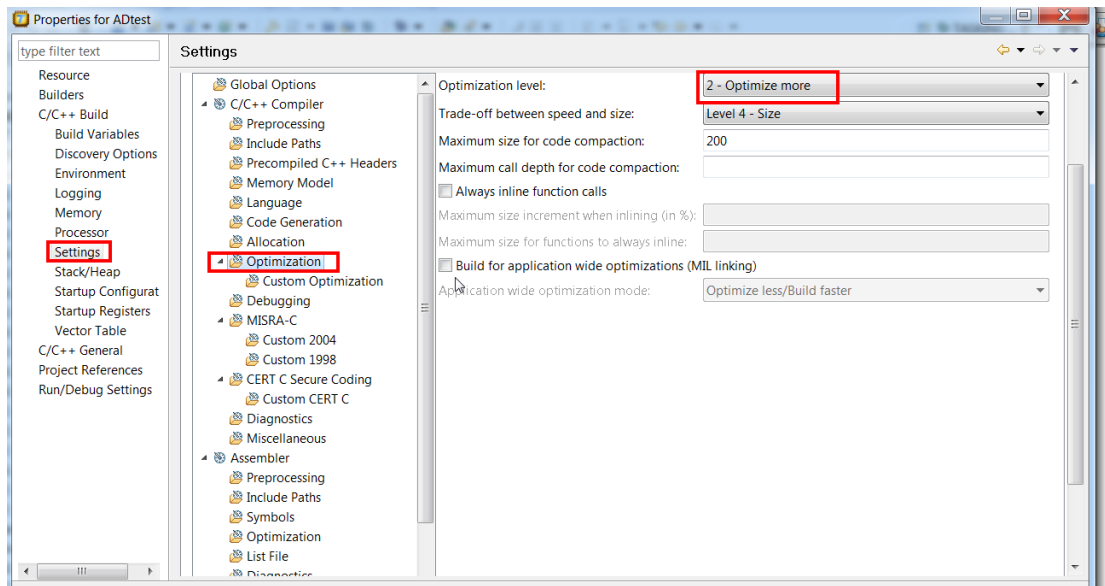
```
for(i=0;i<uwADCResult;i++)
```

```
for(j=0;j<10000;j++);
```

```
// USER CODE END
```

```
}
```

这个演示程序里，注意关闭优化，默认是 2 级，不关闭的话，空的延时会被优化掉



7 BUILD 以后

```

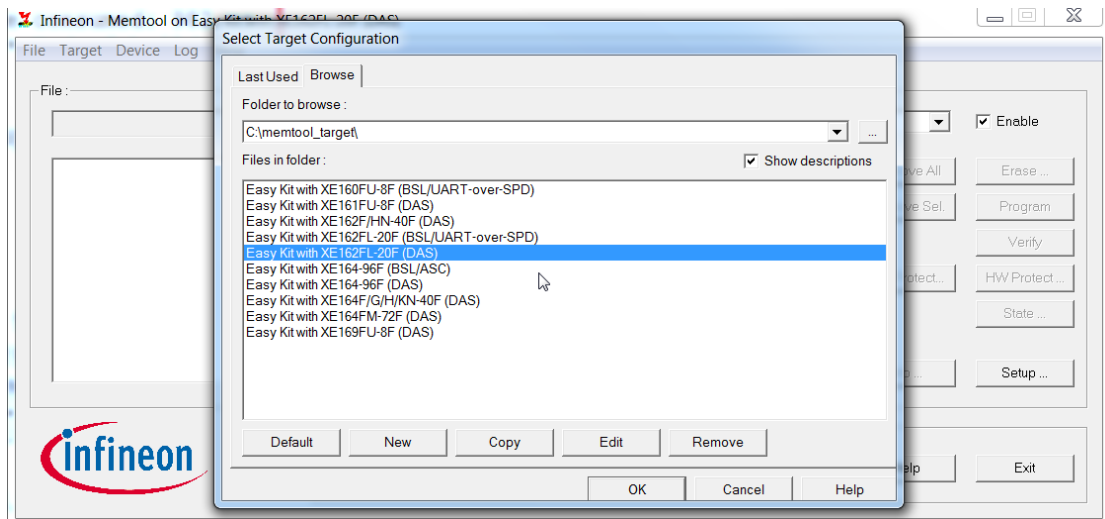
Problems Console Properties
Build [ADtest]
Compiling MAIN.C
Compiling SCS.c
Linking to ADtest.elf

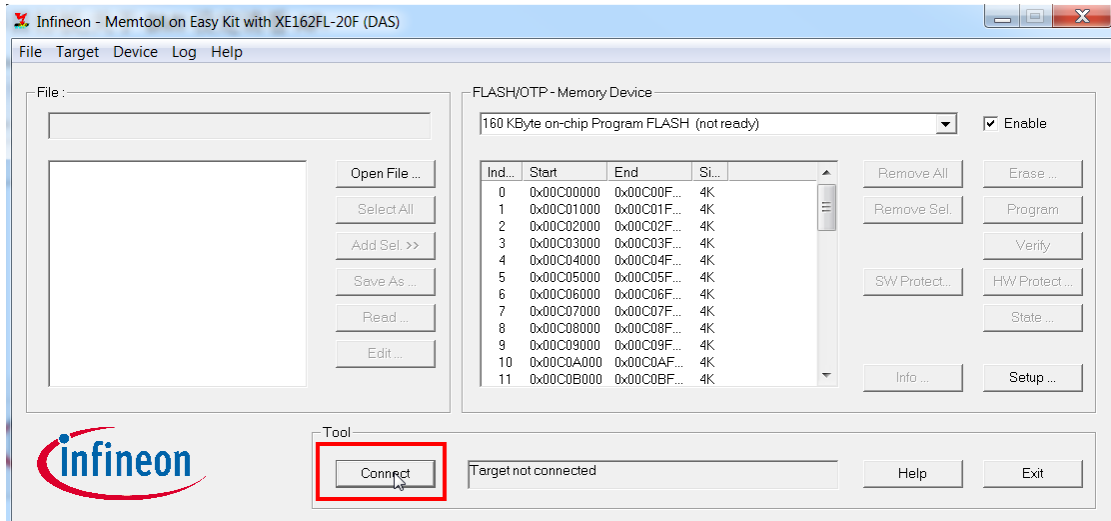
Time consumed: 982 ms
*** End of build ***

```

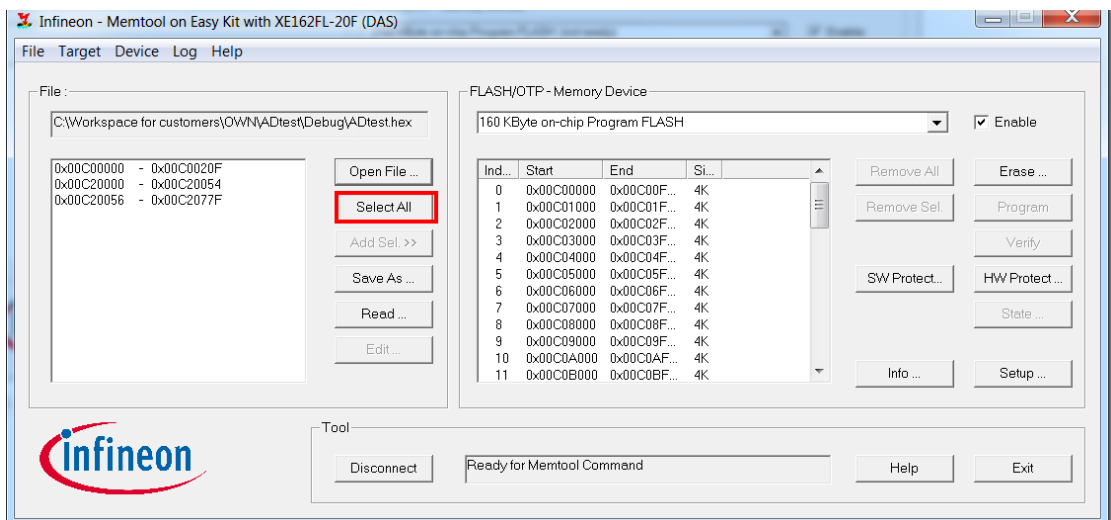
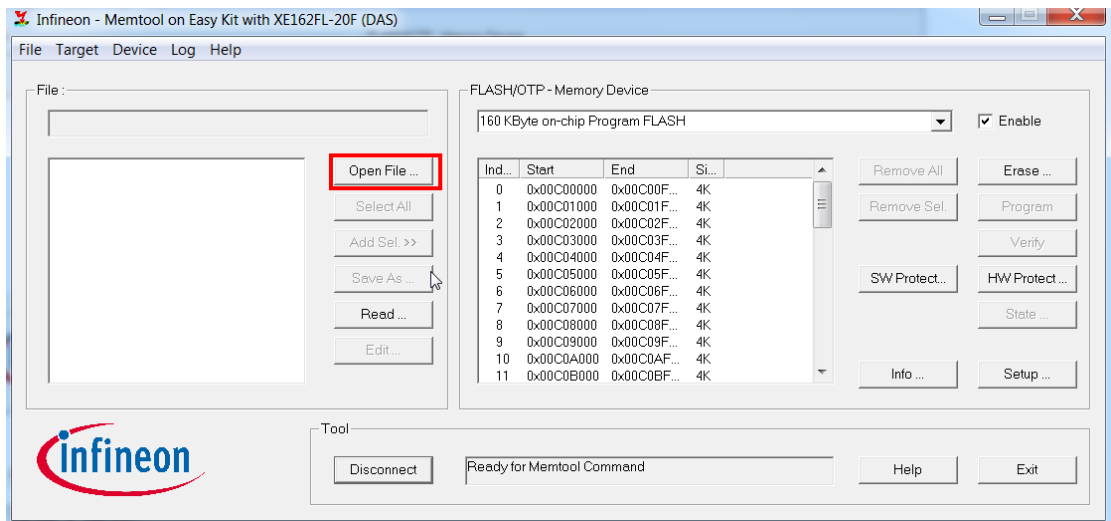
7.1 可以用 memtool 下载 HEX 到 minikit, 仿真见 7.2

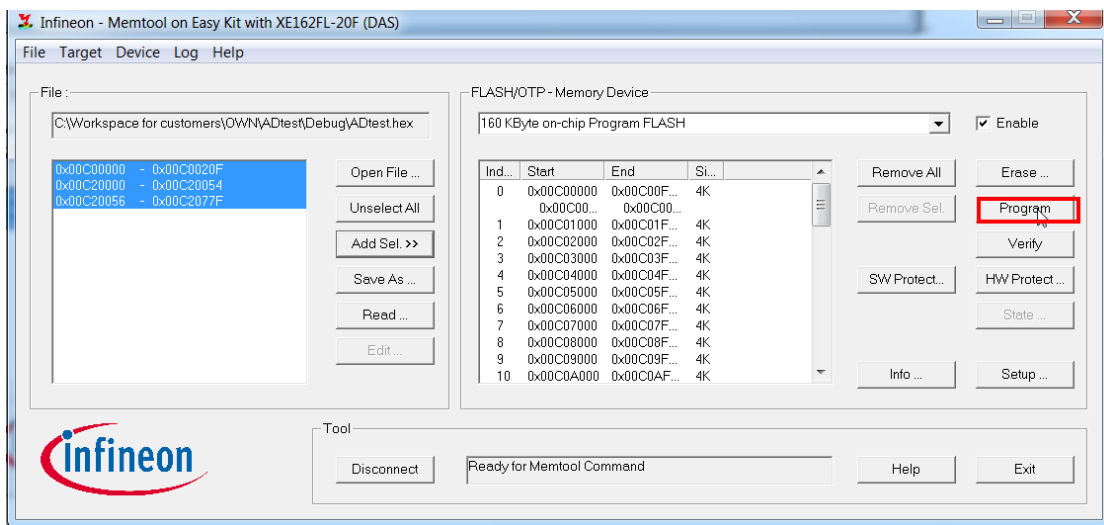
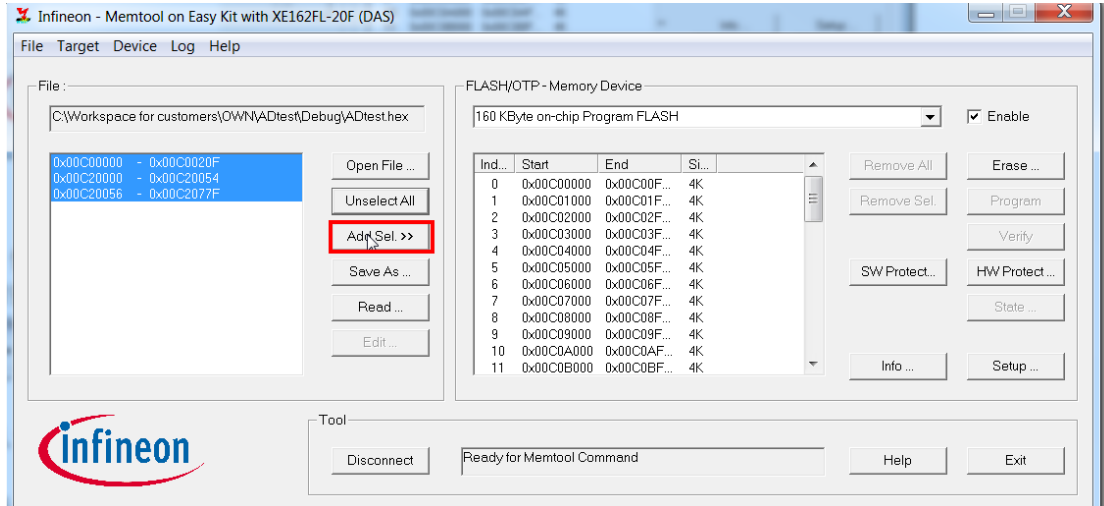
选择目标板,新片子默认 JTAG, 关于 BMI 设定请看 MINIKIT 光盘 目录 说明书



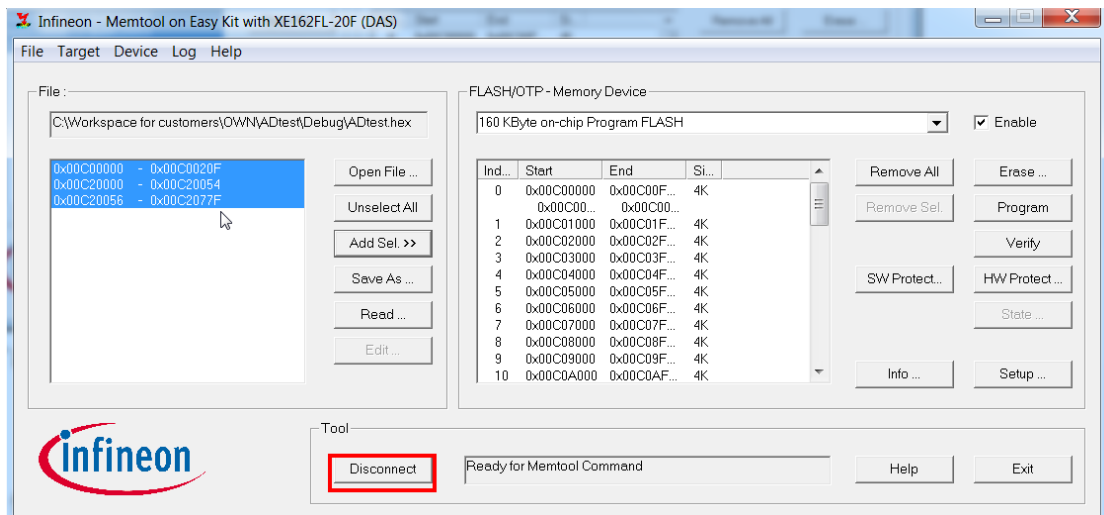


打开 HEX 文件





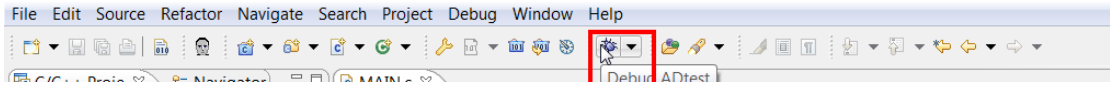
完了以后断开连接



按下复位键就可以了。

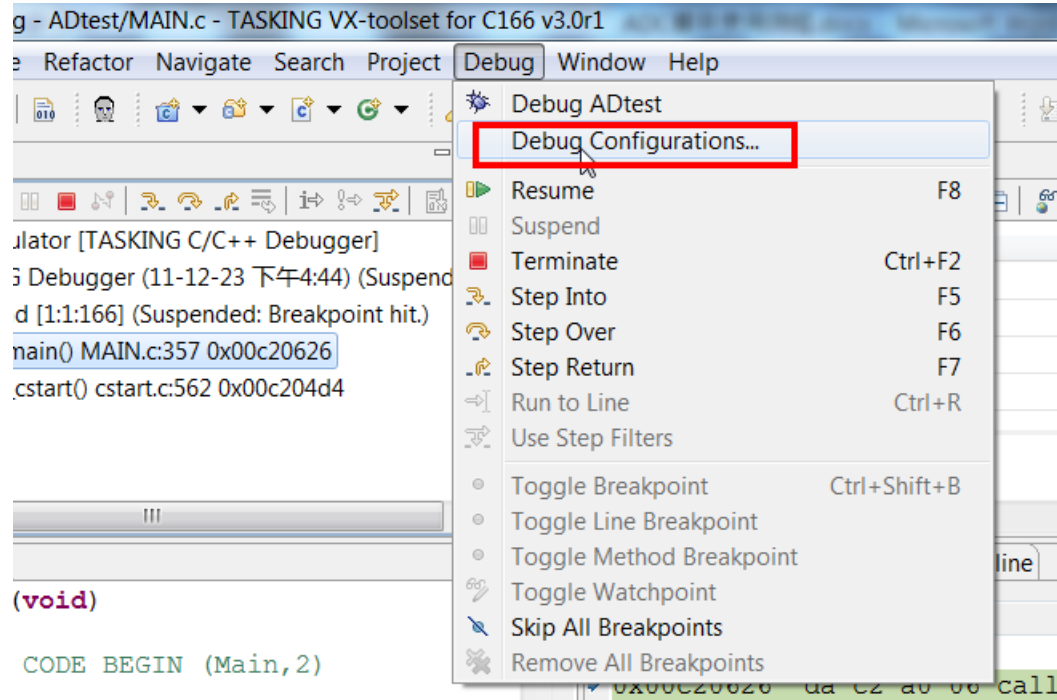
7.2 仿真

或者按虫子图标



默认 Simulator

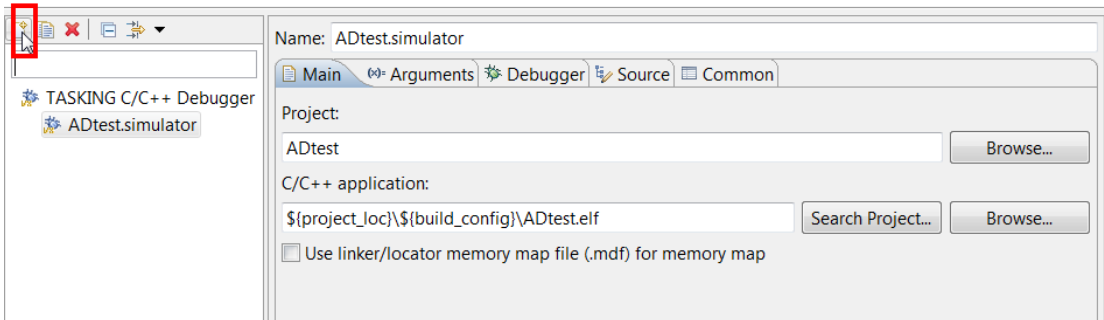
选择 DEBUG Configuration

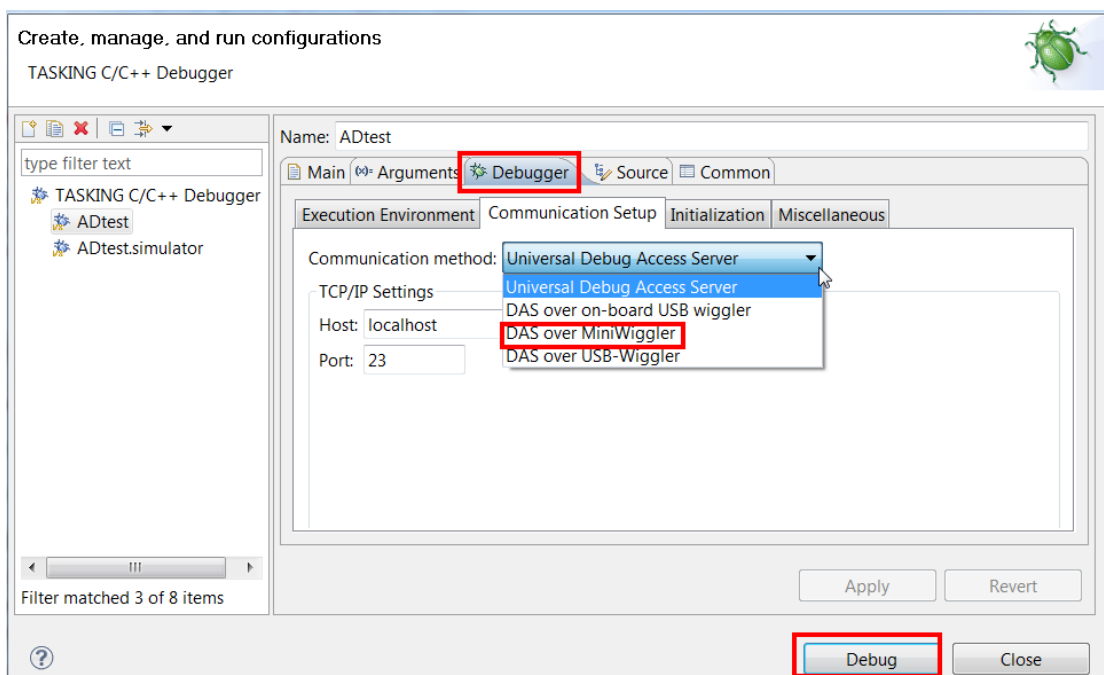
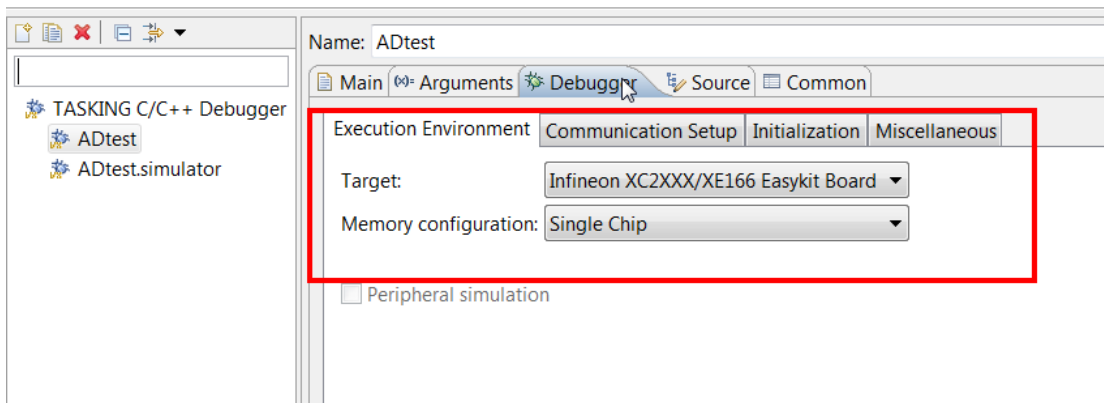
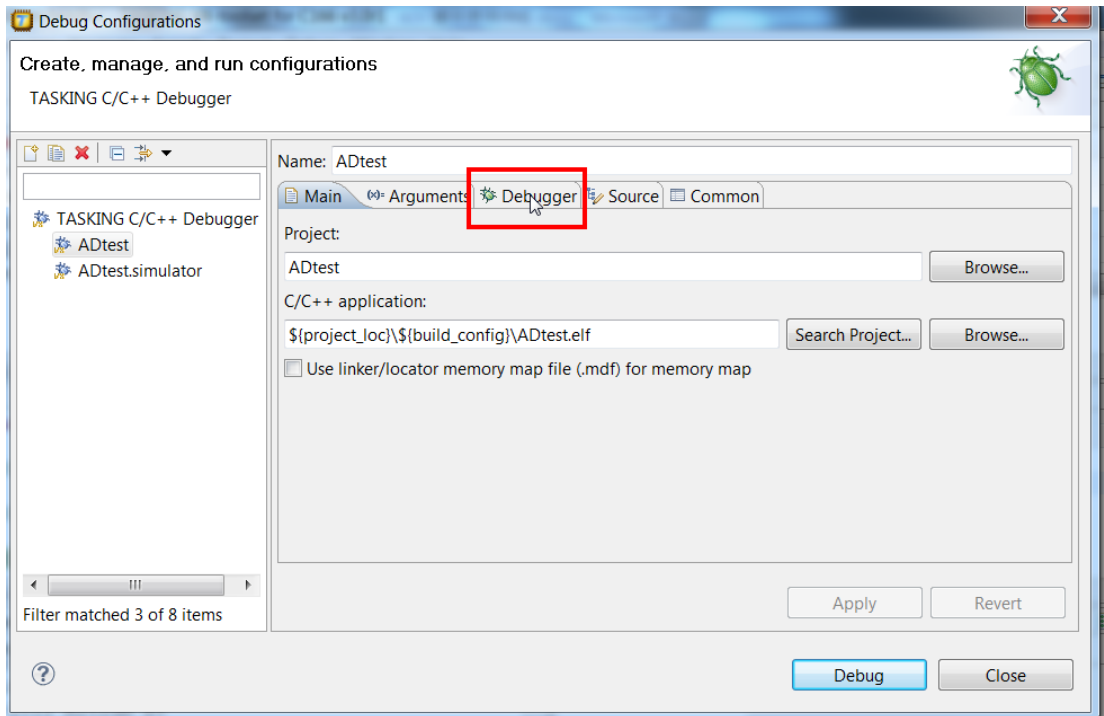


点击新建，设定一个新的仿真设定

Create, manage, and run configurations

Ready to launch





观察变量，注意是全局变量

