PY32F0XX_DFP
用户手册

简介

此文档用于描述 PY32F0xx_DFP 器件支持包的安装以及使用，此器件支持包适用于 Keil MDK 集成开发环境，安装此器件包前需要先安装 Keil MDK 工具。

此器件支持包配合 Keil MDK 软件以及仿真器 PY-LINK，可以实现对 PY32F030 系列 MCU 的开发与调试。支持新建工程、编译、下载、仿真调试四大功能。

目录

1 安装

安装此器件支持包前需要先安装Keil MDK集成开发环境。MDK下载路径：https://www2.keil.com/mdk5。

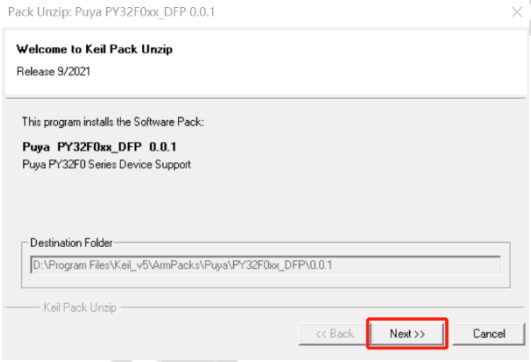
1.1 安装 PY32F0xx_DFP

(1) 双击 pack 包

图 1-1 pack 安装包

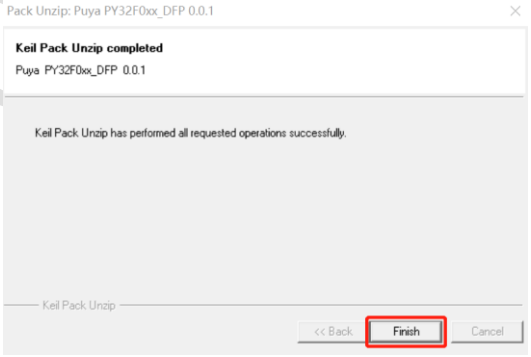
(2) 点击 Next 按钮

图 1-2 安装路径



(3) 点击 Finish 按钮

图 1-3 安装成功提醒

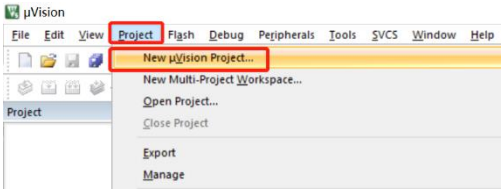


2 使用

2.1 新建工程

(1) 新建文件夹，文件名根据喜好命名，打开 Keil 软件

图 2-1 菜单栏’ Project’ --> ’ New uVision Project’



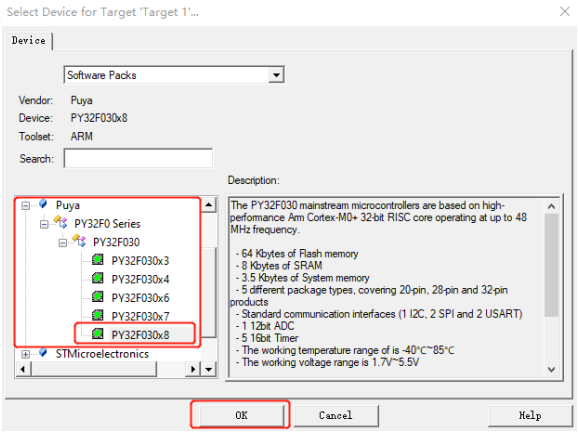
(2) Create New Project，保存在新建文件夹下

图 2-2

(3) 选择 MCU 型号

根据开发板使用的 MCU 具体的型号来选择，这里我们选择’PY32F030x8’型号。如果这里没有出现你想要的 MCU 型号，请参考第二章，确认是否成功安装 device 库。

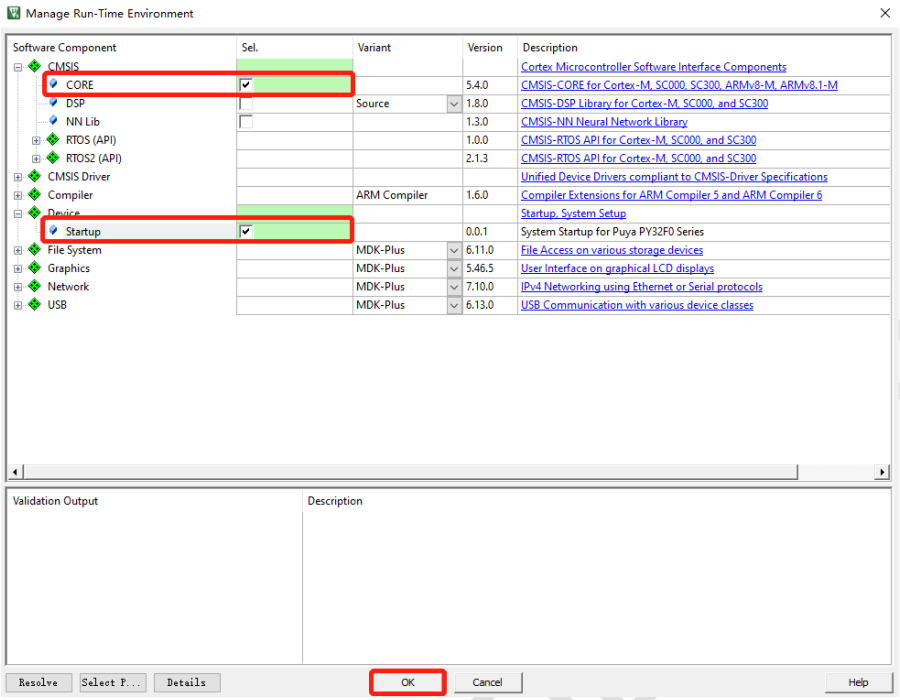
图 2-3 选择具体的 MCU 型号



(4) Manage Run-Time Environment

勾选’CMSIS-->CORE’添加内核文件,勾选’Device-->Startup’添加启动文件。

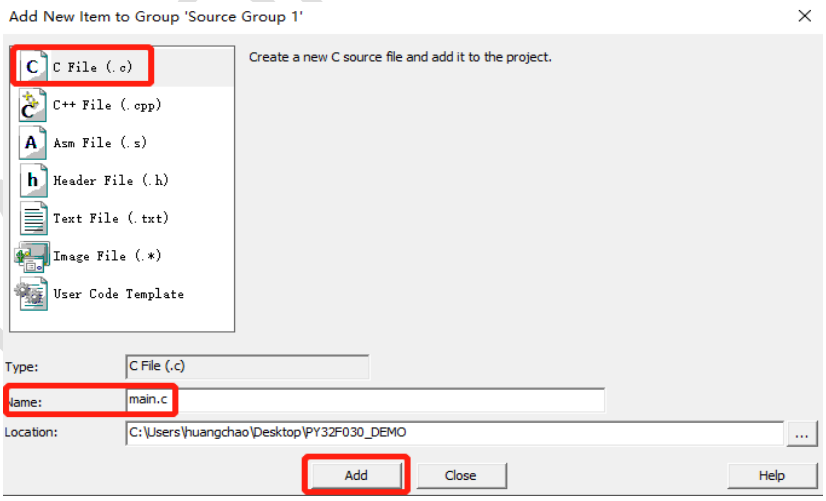
图 2-4 库文件管理



2.2 添加 main.c 文件

- 右击 Project 窗口中 ’Target 1‘中的'Source Group1'文件夹，在弹出的窗口中选择 “Add New Item to Group 'Source Group1'...”
- 在弹出的对话框中选择 C File (.c) ， “Name”编译框中填入 main.c 文件名

图 2-5 添加 main.c 文件



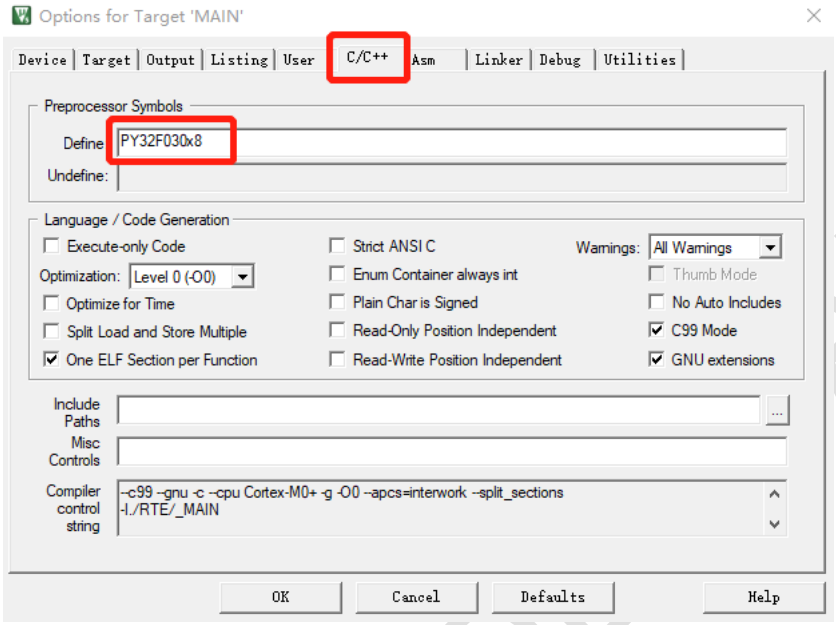
2.3 添加 PY32F030x8 宏

(1) 点击工具栏的魔术棒按钮打开 Options for Target 窗口

图 2-6

(2) 进入 C/C++页面，添加 PY32F030x8 宏,如下图所示

图 2-7



2.4 编辑 main.c 文件

注：文件最后要留至少一行空白行，不然会报警告

图 2-8 main.c 文件

main.c

```
1  #include <py32f0xx.h>
2
3  void delay(uint32_t nTime);
4
5  int main(void)
6  {
7    SET_BIT(RCC->IOPENR, RCC_IOPENR_GPIOAEN);//GPIOA时钟使能
8
9    //设置PA11为通用输出模式
10   MODIFY_REG(GPIOA->MODER, GPIO_MODER_MODE11_1, GPIO_MODER_MODE11_0);
11
12   while (1)
13   {
14     SET_BIT(GPIOA->BSRR, GPIO_BSRR_BS11);//PA11输出高电平
15     delay(0x3FFFF);//软件延时
16
17     SET_BIT(GPIOA->BSRR, GPIO_BSRR_BR11);//PA11输出低电平
18     delay(0x3FFFF);//软件延时
19   }
20 }
21
22 void delay(uint32_t nTime)
23 {
24   while (nTime--);
25 }
26
27
```

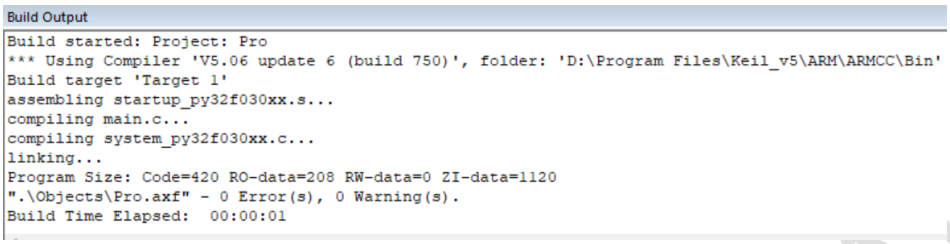
2.5 编译

点击工具栏'build'按钮开始编译。

图 2-9

当出现”.\Objects\Pro.axf” – 0 Error(s), 0 Warning(s).时，代表程序编译通过。

图 2-10 Build Output



2.6 下载

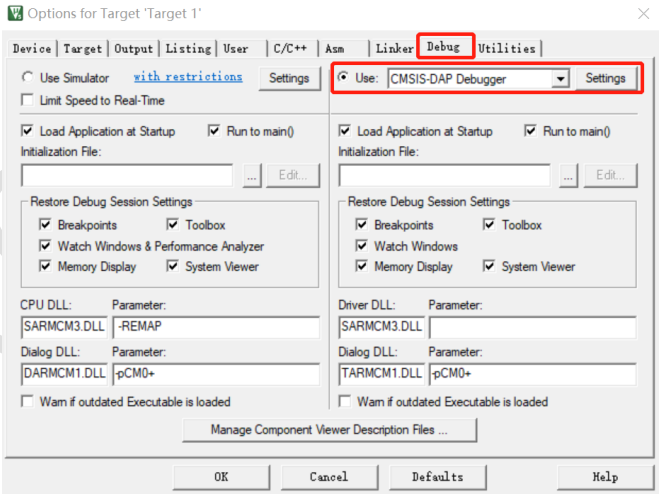
(1) 将仿真器一端通过 USB 与 PC 相连，另一端通过 SWD 与 MCU 相连，如图 2-18 PY32F030 采用 SWD 方式下载和调试，请将仿真器的 TVCC、SWDIO、GND、SWCLK、RST 依次和 MCU/Start Kit 的 VDD、SWDIO(PA13)、GND、SWCLK(PA14)、NRST(PF2)相连。

(2) 点击工具栏的魔术棒按钮打开 Options for Target 窗口

图 2-11

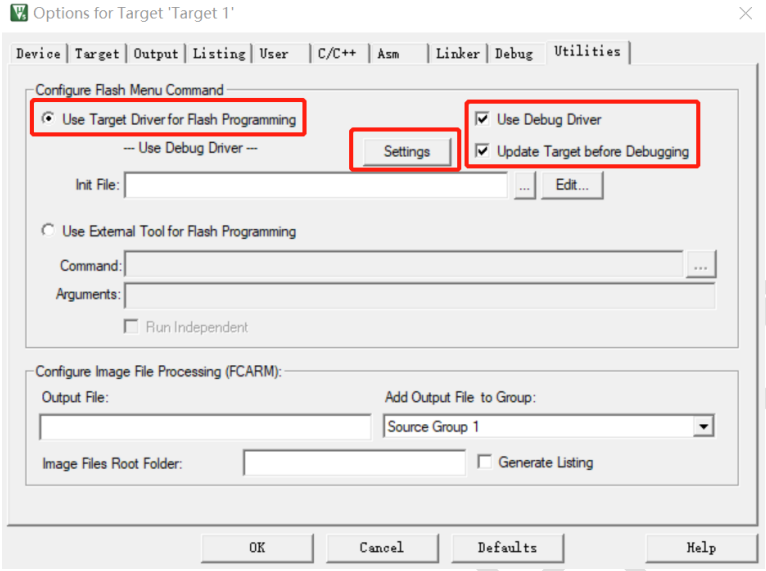
(3) 进入 Debug 页面，仿真器 PY-LINK 选择 CMSIS-DAP Debugger

图 2-12 Debug 选项配置



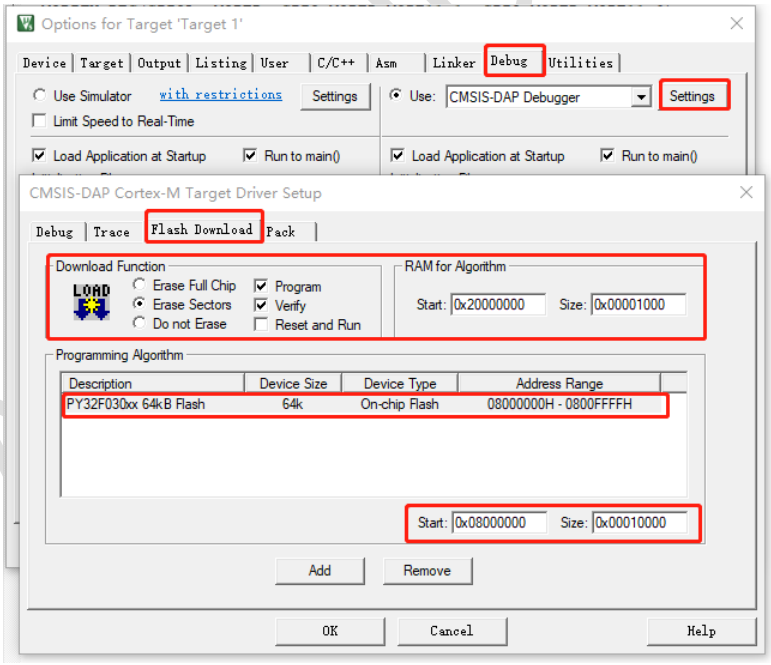
(4) 进入 Utilities 页面，请确认配置是否正确

图 2-13 Utilities 选项配置



(5) 进入 Flash Download 页面，根据目标芯片的型号选择对应的算法

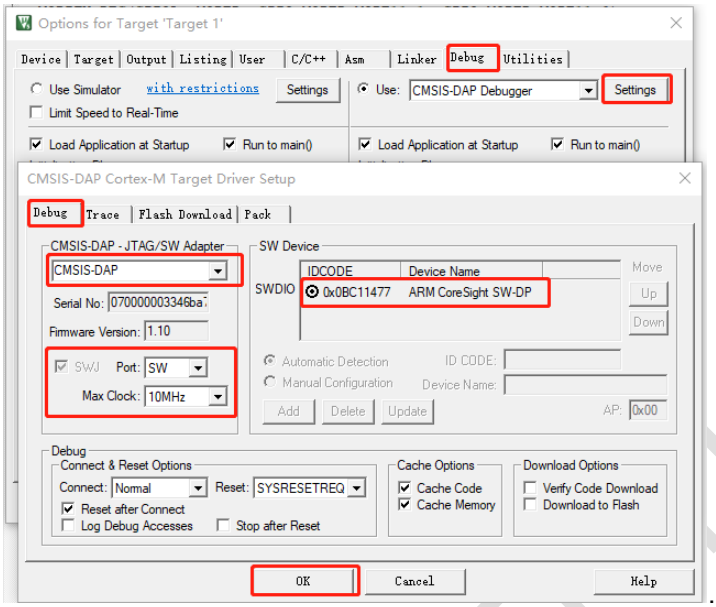
图 2-14



(6) 进入 Debug 页面，完成仿真器相关配置

如果仿真器连接了电脑则 MDK 会在'CMSIS-DAP-JTAG/SW Adapter'识别出仿真器，如果同时开发板已经上电，在'SW Device'会识别到开发板的芯片，并显示出。选择 SW 接口，根据实际情况配置 Max Clock(可配置成 10MHZ，如果下载失败可在此处降低 Clock)。

图 2-15 Debug Setting 选项配置



(7) 点击工具栏的 Download 按钮，开始下载

图 2-16 下载程序

程序下载后，'Build Output'选项卡打印结果如图 2-17

Erase Done：擦除完成

Programing Done：编程完成

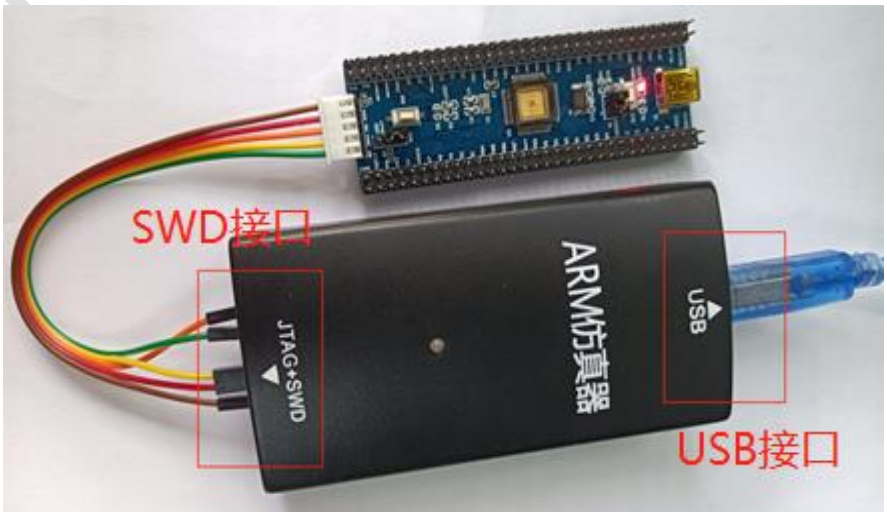
Verify OK：校验成功

图 2-17 Build Output

```
Build Output
Load "C:\\Users\\huangchao\\Desktop\\PY32F030_DEMO\\Objects\\Pro.axf"
Erase Done.
Programming Done.
Verify OK.
Flash Load finished at 17:26:53
```

(8) 按下开发板的复位按键后，PA18 连接的 LED 灯闪烁

图 2-18



2.7 调试

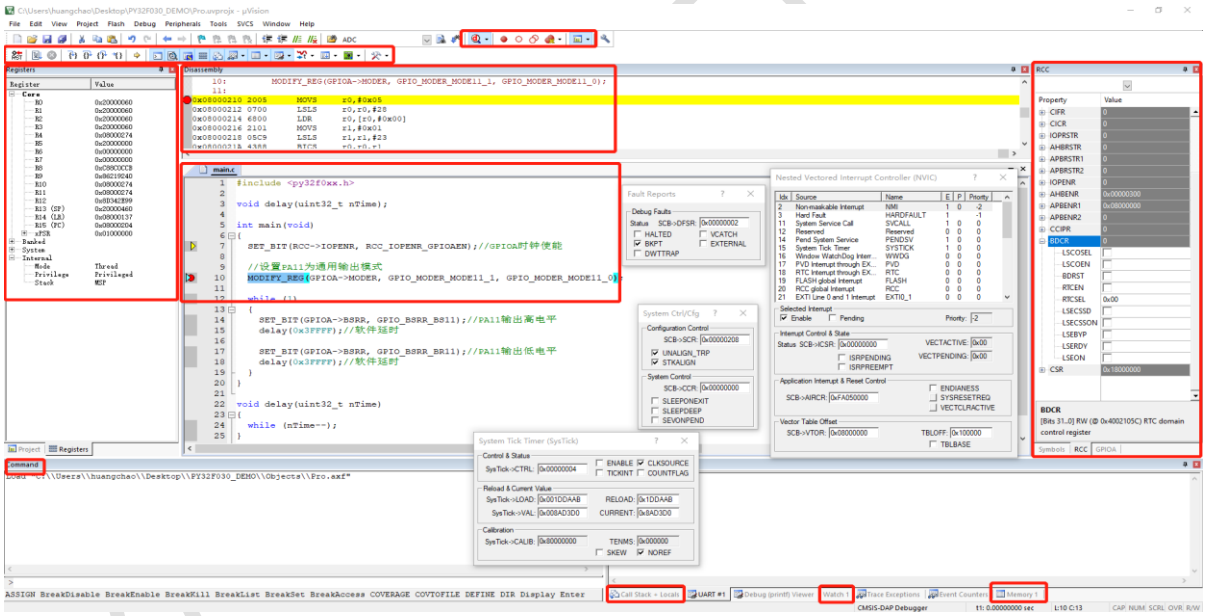
(1) 点击工具栏 ‘ Start/Stop Debug Session’ 按钮，进入调试界面。

仿真调试页面最左边显示的是单片机内部的一些寄存器的当前值和系统信息，上面是 keil 将 C 语言转换成汇编的代码，下面就是我们编写的 C 程序了，最右边是’ System Viewer’ 可以查看或设置 MCU 所有外设寄存器的值，最下面是’Watch 1’ 可以查看任意变量的值，方便我们追踪发现错误。在 C 语言和汇编语言窗口都有一个黄色的箭头，这个箭头代表的就是程序当前运行的位置。

在调试页面上方的工具栏中，有几个按钮：第一个标有 RST 字样的是复位，单击之后，程序就会跑到最开始的位置运行；紧接着第二个按钮是全速运行，单击之后程序就会全速跑起来；再然后第三个按钮是停止按钮，当程序全速运行起来时候，单击停止按钮程序就会立即停止，可以观察程序运行到哪里去了；后面的都是单步运行，具体的进入或跳出请看图示。

单击复位之后，可以看到 C 语言程序的窗口左侧有灰色或者保持着原来的颜色，其中有灰色的地方是我们可以设置断点的地方。

图 2-19 仿真调试界面



(2) 支持的通用调试功能

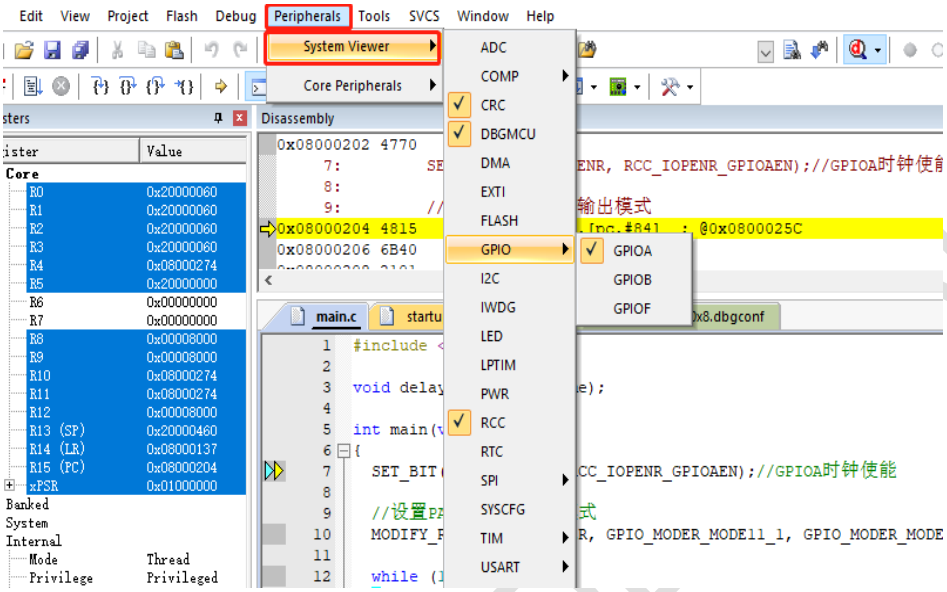
- Start/Stop Debug Session(Enter or leave a debug session)
- Insert/Remove Breakpoint(Insert or remove a breakpoint at the current line)
- Disable All Breakpoints in current Target
- Kill All Breakpoints in current/active Target
- Reset(Reset the CPU)
- Run(Start code execution)
- Step(Step one line)
- Step Over(Step over the current line)
- Step Out(Step out of the current function)
- Run to Cursor Line(Run to the current cursor line)
- Command Window
- Disassembly Window
- Registers Window
- Call Stack Window
- Watch Windows
- Memory Windows

这些通用调试功能的用法请参考 Keil MDK 用户手册，这里不再赘述。

(3) 外设寄存器查看窗口

点击工具栏 'Peripherals' --> 'System Viewer'可以查看或设置 MCU 所有外设寄存器的值。

图 2-20



(4) 预设 DBGMCU 寄存器

为了便于调试，可以在进入调试前点击 'Edit' 设置 DBGMCU 寄存器的值.

为了在 MCU 进入 STOP/SLEEP 模式后也能进行调试，器件支持包已默认将 DBGMCU_CR 寄存器的值设置为 0x00000002，即 DBG_STOP 位置 1，对应图 2-23 右侧 DBG_STOP 位被勾选。

图 2-21 的 *.dbgconf 文件用户可以根据实际需求自行修改，只需修改红色框标注的三处地方，分别对应 PY32F030 的 DBGMCU_CR, DBG_APB_FZ1 和 DBG_APB_FZ2 寄存器，这 3 个寄存器及位定义请参考 PY32F030 数据手册。

图 2-21

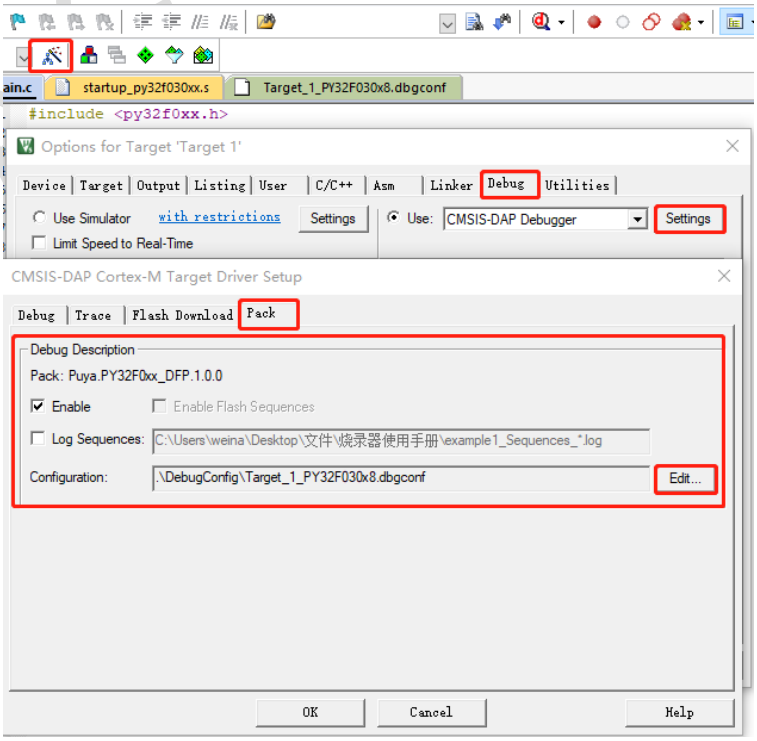
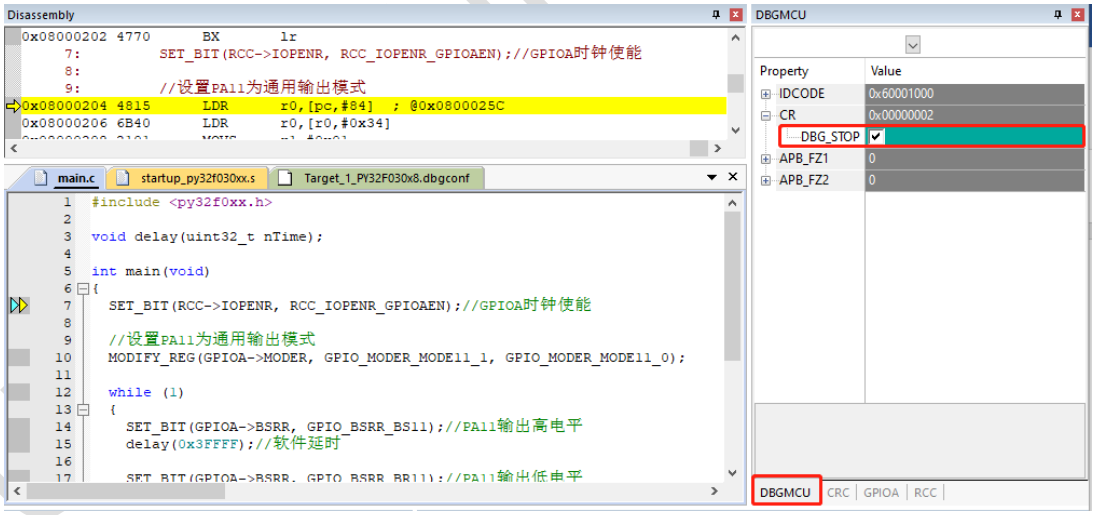


图 2-22

Target_1_PY32F030x8.dbgconf

```
// File: PY32F030xx.dbgconf
// Version: 1.0.0

// <<< Use Configuration Wizard in Context Menu >>>

// <h> Debug MCU configuration register (DBGMCU_CR)
//   <o.1>  DBG_STOP                <i> Debug stop mode
// </h>
DbgMCU_CR = 0x00000002;

// <h> Debug MCU APB freeze1 register (DBG_APB_FZ1)
//                                  <i> Reserved bits must be kept at reset value
//   <o.31> DBG_LPTIM_STOP          <i> LPTIM stopped when core is halted
//   <o.12> DBG_IWDG_STOP           <i> Independent watchdog stopped when core is halted
//   <o.11> DBG_WWDG_STOP           <i> Window watchdog stopped when core is halted
//   <o.10> DBG_RTC_STOP            <i> RTC stopped when core is halted
//   <o.4>  DBG_TIM6_STOP           <i> TIM6 counter stopped when core is halted
//   <o.1>  DBG_TIM3_STOP           <i> TIM3 counter stopped when core is halted
// </h>
DbgMCU_APB_Fz1 = 0x00000000;

// <h> Debug MCU APB freeze2 register (DBG_APB_FZ2)
//                                  <i> Reserved bits must be kept at reset value
//   <o.18> DBG_TIM17_STOP          <i> TIM17 counter stopped when core is halted
//   <o.17> DBG_TIM16_STOP          <i> TIM16 counter stopped when core is halted
//   <o.15> DBG_TIM14_STOP          <i> TIM14 counter stopped when core is halted
//   <o.11> DBG_TIM1_STOP           <i> TIM1 counter stopped when core is halted
// </h>
DbgMCU_APB_Fz2 = 0x00000000;

// <<< end of configuration section >>>
```

图 2-23



3 版本历史

Version	日期	Date
V1.0	Initial Release	2022.07.5

Puya Semiconductor Co., Ltd.