Haawking_DSC280025C-EDB_数字信号控制器

迁移指南

V0.8

北京中科昊芯科技有限公司

2023 年 07 月

历史版本记录

版本号	内容描述	时间
V0.1	初始版本	2023.03.03
V0.2	修订嵌套中断部分内容	2023.03.04
V0.3	增加 Flash 配置时，禁用 Cache 和 Prefetch 的要求。 更新 CODE_SECTION()的用法，需要将其放在函数名前面。 增加 RPTI 不能被打断的说明。 更新中断嵌套示例。	2023.04.24
V0.4	增加 ERAD 部分差异	2023.05.10
V0.5	增加中断向量表的差异。 增加 CLB 模块差异。 增加 HRPWM 模式的参数对比。	2023.05.23
V0.6	增加 TMU 模块的差异	2023.05.23
V0.7	增加 ERAD 中计数器子模块不支持 Start/Stop 模式的说明	2023.05.26
V0.8	修改文档封面、页眉、页脚，以符合质量基线	2023.07.24

目录

1. 芯片与友商差异

本版本指南适用于 Haawking-HX2000 系列芯片中的 DSC280025C-EDB 版本。

1.1. CPU 核

Haawking HX2000 系列芯片基于开放指令集架构 RISC-V 开发，支持 RV32IMFC 标准指令集+DSP 自定义指令集，在程序开发的时候，高级语言开发的程序代码由编译器负责生成可执行文件，汇编语言开发的代码，则需要手动根据 RISC-V 指令集进行修改。

1.1.1. 饱和加减法

友商提供了一个 OVM 位，在置位 OVM 之后，后续的加法和减法操作会带有饱和限制。
比如,友商的代码如下：

```
long Accum_A, Accum_B;
void Calc_Fun(long in_Dat)
{

    SETOVM;
    Accum_A = Accum_A + in_Dat;
    Accum_B = Accum_B - in_Dat;
    CLROVM;
}
```

昊芯提供了带饱和的加减法指令 sadd 和 ssub ，可以替代 SETOVM 来实现同样的功能。

```
I32_IQ _sadd(I32_IQ a, I32_IQ b)
{
    I32_IQ result;

    asm volatile
    (
            "sadd %[z], %[x], %[y]\n\t"
            :[z] "=r"(result)
            :[x] "r"(a),[y]"r"(b)
    );

    return result;
}


```

```
I32_IQ _ssub(I32_IQ a, I32_IQ b)
{
    I32_IQ result;

    asm volatile
    (
            "ssub %[z], %[x], %[y]\n\t"
            :[z] "=r"(result)
            :[x] "r"(a),[y]"r"(b)
    );

    return result;
}
```

在修改之前的程序的时候，需要用_sadd()和_ssub()来替代代码中 SETOVM 和 CLROVM 之间的加法和减法。

修改后的代码如下：

```
long Accum_A, Accum_B;
void Calc_Fun(long in_Dat)
{

    //SETOVM;
    //Accum_A = Accum_A + in_Dat;
    Accum_A = _sadd(Accum_A ,in_Dat);
    //Accum_B = Accum_B - in_Dat;
    Accum_B = _ssub(Accum_B ,in_Dat);
    //CLROVM;
}
```

1.1.2. CPU 中断向量表差异

在 CPU 的中断向量表中，昊芯的 Vector 15 是 RTOSINT，第 16 位是 NMI；友商则第 15 位是 DLOGINT，第 16 位是 RTOS。

Name	Vector ID	Address	Size（×16）	Description	Core Priority	ePIE Group Priority
Rest	0	0x0000 5800	2	未使用。见 PIE Group 1	1（highest）	-
INT1	1	0x0000 5804	2	未使用。见 PIE Group 2	2	-
INT2	2	0x0000 5808	2	未使用。见 PIE Group 3	3	-

INT3	3	0x0000 580C	2	未使用。见 PIE Group 4	4	-
INT4	4	0x0000 5810	2	未使用。见 PIE Group 5	5	-
INT5	5	0x0000 5814	2	未使用。见 PIE Group 6	6	-
INT6	6	0x0000 5818	2	未使用。见 PIE Group 7	7	-
INT7	7	0x0000 581C	2	未使用。见 PIE Group 8	8	-
INT8	8	0x0000 5820	2	未使用。见 PIE Group 9	9	-
INT9	9	0x0000 5824	2	未使用。见 PIE Group 10	10	-
INT10	10	0x0000 5828	2	未使用。见 PIE Group 11	11	-
INT11	11	0x0000 582C	2	未使用。见 PIE Group 12	12	-
INT12	12	0x0000 5830	2	CPU TIMER1 中断	13	-
INT13	13	0x0000 5834	2	CPU TIMER2 中断	14	-
INT14	14	0x0000 5838	2	CPU 数据记录中断	15	-
RTOSINT	15	0x0000 5840	2	CPU 模拟中断	17	-
NMI	16	0x0000 5848	2	非法指令 (ITRAP)	19 (lowest)	-

1.1.3. IER 寄存器的 NMI 位读取值为 0

昊芯当前（EDB）版本的 280025C 中，NMI 位（只读位，默认为 1）的读取值为 0，但是 NMI 中断是可以被正常触发的，用户需要忽略读到的 NMI 的寄存器值。

1.2. SYSCTRL 模块

1.2.1. PLL

系统支持的最高频率不一致，友商 的 PLLRAWCLK 最高 200MHZ，系统时钟范围是 2MHZ~100MHZ，昊芯的 PLLRAWCLK 最高 320MHZ，系统时钟范围是 2MHZ~160MHZ。

友商配置 SYSPLLMULT 时自动使能 PLL，昊芯 280025C 不会在配置 SYSPLLMULT 时自动使能 PLL，必须通过 SYSPLLCTL1.PLLEN 使能 PLL（为了避免 PLL 误开启）。

1.2.2. 时钟源(OSCCLK)故障检测

昊芯的 0025C 支持 Missing Clock Detection(MCD)，不支持 PLL Reference Clock Lost Detection。

MCD 检测到 clock fail 时，友商会将 SYSPLLMULT 内的倍频系数清零，昊芯 280025C 不会自动将 SYSPLLMULT 内的倍频系数清零，而是清为 1。

1.3. ePWM 模块

ePWM 模块的寄存器仅支持 32 位访问。

1.3.1. HRPWM 模式

昊芯 0025 的 HRPWM 精度更高，下面是跟友商的对比：

	昊芯 0025	友商 0025
Step 精度（典型值）		150ps
Step 精度（最大值）	100ps	310ps
Half Cycle 模式	不需要开启 Half Cycle 模式	必须开启 Half Cycle 模式
高精度 MEP	512 step	256 step
高精度寄存器位数	9-bit	8-bit

1.3.2. Edge Filter 功能的差异

在使用诸如 Valley Switching 功能时，需要配置 Edge Filter 的相关寄存器。昊芯的寄存器配置的效果和友商存在差异。

差异 1. Edge Capture Logic 边沿检测

功能说明：

Edge Capture Logic 功能能够通过识别 DCxEVTx 信号的边沿得到 VCNTVAL 的值。

差异现象：

友商的 F280025C 芯片中该功能识别 DCxEVTx 信号的上升沿和下降沿；

昊芯当前 0025C-EDB 版本的芯片暂不支持识别 DCxEVTx 信号的下降沿。

影响范围：

VCNTVAL 寄存器仅能对 DCxEVTx 信号的上升沿进行计数。

差异 2.VCAPE 禁用功能

功能说明：

VCAPE 开关控制 Valley Switching 功能中是否使用 Edge Capture Logic，并使指定的捕获事件触发后重置 Edge Selection Logic 的计数；

Edge Capture Logic 控制 PWM 切换前的延迟；

Edge Selection Logic 控制延迟生效前一个可编程的边沿数。

差异现象：

在 VCAPCTL.VCAPE = 0,VCAPCTL.EDGEFILTDLYSEL = 1 时：

友商的 F280025C 中 Edge Capture Logic 不产生 VCNTVAL，对 SWVDELVAL 的写入值不会生效；

昊芯当前 0025C-EDB 版本的芯片中 Edge Capture Logic 不产生 VCNTVAL，对 SWVDELVAL 的写入值正常生效。

影响范围：

对 SWVDELVAL 的写入值总是生效。

差异 3.VCAPE 启用功能

功能说明：
同上。

差异现象：
在 VCAPCTL.VCAPE=1,VCAPCTL.EDGEFILTDLYSEL = 1,DCFCTL.EDGECOUNT=1 时：
友商的 F280025C 的 Edge Selection Logic 仅会在每次触发捕获事件后产生一次 DCxEVTx 信号；
昊芯当前 0025C-EDB 版本的芯片的 Edge Selection Logic 会在每次触发捕获事件后每计数到 DCFCTL.EDGECOUT 的值都产生 DCxEVTx 信号。

影响范围：
友商的 F280025C 的实际延迟的值等于 HWDELVAL 的值；昊芯当前 0025C-EDB 版本的芯片的实际延迟由 Delay_cnt 计数得到，DCxEVTx 会重置 Delay_cnt 计数。

1.4. ADC 模块

1.4.1. ADC 模块分频

0025C-EDB 片内 ADC 模块的典型工作频率为 20MHz，最高工作频率为 32MHz，所以在 CPU 的主频在 160MHz 的时候，需要设置为 8 分频。

1.4.2. ADC 首次转换结果存在偏差

因为昊芯使用的 ADC IP 的特性，当前 0025C-EDB 版本的每通道首次采样结果存在偏差。

具体表现：
每个通道第一次采样结果偏差较大。

规避办法：
舍弃每个通道第一次采样结果；

1.5. I2S 模块

昊芯的 280025C 增加了一个 I2S 模块。

1.6. 比较器模块

COMPHYSCTL 寄存器默认值不同，昊芯增加了一个比较器使能的位 GENCMP_EN。

CMPSS differences	REG	昊芯	友商
1	COMPHYSCTL[2:0]	默认值为 3'b110，比较器工作为高速高功耗、无迟滞模式	默认值为 3‘b0，无迟滞
2	COMPHYSCTL[3]	此位为 GENCMP_EN，复位后的初始值为 0，需要写 1 使能比较器	无比较器使能位

此外，昊芯新增了一个比较器的 DAC 分频寄存器 COMPDIV，可以设置 DAC 模块的频率。

COMPDIV寄存器（偏移量=1Ah）[复位=0h]

COMPDIV如图16-27所示，如表16-24所述。

DAC分频

图16-27 COMPDIV寄存器

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						COMPIV	
R-0h						R/W-0h	

表16-24 COMPDIV寄存器描述

位	字段	类型	重置	描述
15-3	reserved	R	0h	reserved
2-0	COMPDIV	R/W-0h	0h	比较器 DAC 分频系数，对应分频关系如下 0：2 分频 1：4 分频 2：6 分频 3：8 分频 4：10 分频 5：12 分频 6：14 分频 7：16 分频 重置类型：SYSRSn

1.7. HRCAP 模块

昊芯 0025C-EDB 的 HRCAP 模块，跟昊芯 28034 兼容。比友商多出 HCCAL，HCCALMEP，HCMEPSTATUS 这三个寄存器，详细说明可参阅 28034 的技术参考手册。

1.8. DMA 模块

友商的 DMA 不支持 SCI 数据传输，昊芯 280025C DMA 支持 SCI 传输。
友商的 DMA 不支持 I2C 数据传输，昊芯 280025C DMA 支持 I2CA 和 I2CB 的传输。

1.9. eQEP 模块

寄存器与友商一致，可以直接代码替换。

1.10. eCAP 模块

寄存器与友商一致，可以直接代码替换。

1.11. 外部中断 XINT

寄存器与友商一致，可以直接代码替换。

1.12. 片上 Flash

为了保证 Flash 的配置正确，初始化 DSP 时请使用昊芯提供的 IDE 自带的 Flash_initModule()函数。该函数在 Device_init()中会被调用。

1.12.1. 禁用缓存和预取功能

在 0025 的 EDB 版本中，Flash 在初始化的时候，需要确认将 Cache 和 Prefetch 机制禁用。否则有几率导致 CPU 从 Flash 中读到的数据出错。

从 Haawking IDE 2.1.6 开始，默认的配置是禁用的。

如果用户用了早期版本的 IDE(v2.1.2， v2.1.3 和 v2.1.4 版本)建的工程，或者使用的是 IDE2.1.6 之前版本的示例程序，那么需要手动修改，在

haawking-drivers\haawking-dsc280025_edb-board\flash.c 文件中，将 Flash_enableCache(ctrlBase)一行注释掉。

下面的代码中，需要修改黄色高亮的一行，将其注释掉。

```
1.  void CODE_SECTION("ramfuncs")
2.  Flash_initModule(uint32_t ctrlBase, uint32_t eccBase, uint16_t waitstates)
3.  {
4.      //
5.      // Check the arguments.
6.      //
```

```
7.      ASSERT(Flash_isCtrlBaseValid(ctrlBase));
8.      ASSERT(Flash_isECCBaseValid(eccBase));
9.      ASSERT(waitstates <= 0xFU);
10.
11.     //
12.     // Set the bank fallback power mode to active.
13.     //
14.     Flash_setBankPowerMode(ctrlBase, FLASH_BANK, FLASH_BANK_PWR_ACTIVE);
15.
16.     //
17.     // Disable cache and prefetch mechanism before changing wait states
18.     //
19.     Flash_disableCache(ctrlBase);
20.     Flash_disablePrefetch(ctrlBase);
21.
22.     //
23.     // Set waitstates according to frequency.
24.     //
25.     Flash_setWaitstates(ctrlBase, waitstates);
26.
27.     //
28.     // Enable cache and prefetch mechanism to improve performance of code
29.     // executed from flash.
30.     //
31.     Flash_enableCache(ctrlBase);
32. //    Flash_enablePrefetch(ctrlBase);
33.
34. ...
35. }
```

1.13. ERAD 模块

1.13.1. 不支持 VPC 功能

由于 CPU 的架构和流水线不同，在 0025 的 EDB 版本中，ERAD 模块不支持 VPC 功能。在 HWBP_CNTL 寄存器的 BUS_SEL 域，所有与 VPC 相关的值都是无效的。

昊芯 280025C		友商 280025	
BUS_SEL 4 位的二进制值	描述	BUS_SEL 4 位的二进制值	描述

0000	PAB 用于取指令	0000	PAB for instruction fetches
0001	**保留**	0001	VPC
0010	DWAB 用于数据写入访问	0010	DWAB for data write accesses
0011	DRAB 用于数据读取访问	0011	DRAB for data read accesses
0100	DWDB 用于写入数据匹配	0100	DWDB for write data match
0101	DRDB 用于读取数据匹配	0101	DRDB for read data match
0110	**保留**	0110	VPC Instruction aligned match
0111	**保留**	0111	VPC R1 aligned match
1000	**保留**	1000	VPC R2 aligned match
1001	**保留**	1001	VPC W aligned match

1.13.2. EBC 的 COMP_MODE 错误

在 HWBP_CNTL 寄存器的 COMP_MODE 域，当配置为 EBC_DWAB_GT 和 EBC_DWAB_LT 时，实时地址与参考地址比较会发生错误。

昊芯 280025C		友商 280025	
COMP_MODE 3 位的二进制值	描述	COMP_MODE 3 位的二进制值	描述
000	正常使用掩码 Mask 比较	0000	Regular masked compare
100	**总线值大于 HWBP_REF**	0001	Bus value GT HWBP_REF
101	**总线值大于或等于 HWBP_REF**	0010	Bus value GE HWBP_REF
110	**总线值小于 HWBP_REF**	0011	Bus value LT HWBP_REF
111	**总线值大于或等于 HWBP_REF**	0100	Bus value LE HWBP_REF

1.13.3. CRC 功能错误

CRC_SEED 域功能错误，按照顺序配置 CRC 时，SEED 值无法正常设置为初始值。

CRC_QUALIFIER 过滤器条件错误。CRC_QUALIFIER 的值非 0 时，无法按照所设置的过滤条件来工作。

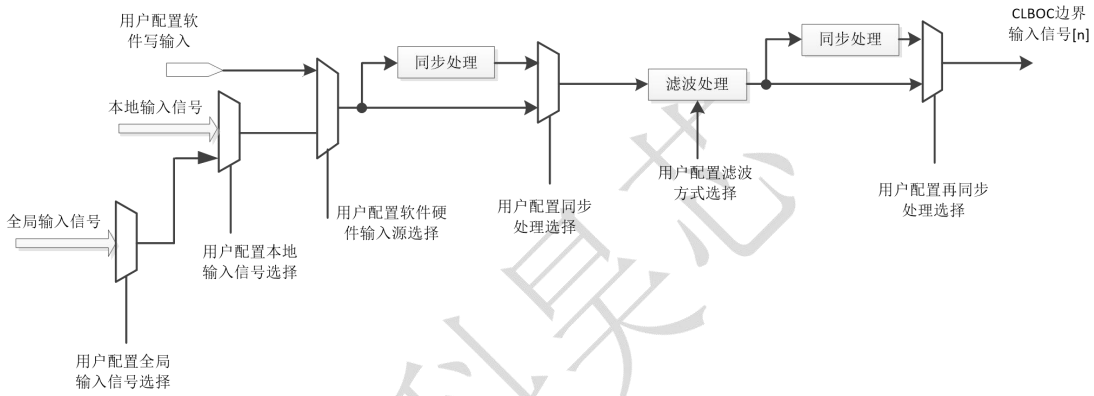
1.13.4. 计数器子模块不支持 Start/Stop 模式

在 0025 的 EDB 版本中，ERAD 模块中的计数器子模块不支持 Start/Stop 模式。

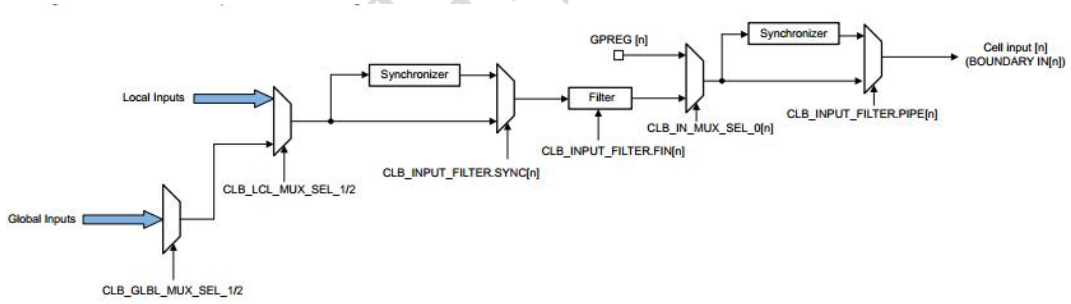
1.14. CLB 模块

1.14.1. CLB 输入 MUX 和滤波

昊芯的 CLB 输入 MUX 和 Filter，增加了用户配置软件写输入的选择，信号框图如下：



作为对比，友商的 CLB 输入 MUX 和滤波框图如下：



1.14.2. CLB HLC 中 ADD 指令的目的寄存器

友商 TRM 中描述 CLB HLC 的加法指令 ADD<Src><Dest>时，提到 Dest 只能是 R0，R3，昊芯的 CLB HLC 的加法指令在设计中，R0-R3 都是可以的，这也跟友商的实测情况相同。

1.14.3. CLB HLC 中 MOV_T 指令的目的寄存器

友商 TRM 中描述 CLB HLC 的 MOV_T 指令 MOV_T1<Src><Dest>和 MOV_T2<Src><Dest>时，提到 Dest 只能是 C0-C2,因此，不存在友商 TRM 中举例所出现的 MOV_T1 C1 R0 的用法，昊芯的 0025，MOV_T 的指令与友商描述相同，Dest 只能是 C0-C2。

1.14.4. CLB HLC 的触发边沿选择寄存器

CLB HLC 的触发可以选择上升沿或者下降沿，但是友商 TRM 中没有提到可以控制的寄存器，HX-280025C 有这个控制寄存器，寄存器名称为 CLB_HLC_EVENT_CTRL，偏移地址为 0xA8。

CLB_HLC_EVENT_CTRL 寄存器的描述

位描述	Bit	默认值	读/写	位域描述
EVENT0_EDGE_SEL	0	1	R/W	HLC event 0 edge select
EVENT1_EDGE_SEL	1	1	R/W	HLC event 1 edge select
EVENT2_EDGE_SEL	2	1	R/W	HLC event 2 edge select
EVENT3_EDGE_SEL	3	1	R/W	HLC event 3 edge select
SPI_DATA_STRB_EXTND	4	1	R/W	Choosing the alternate MUX options for the HLC module
rsvd	5	27	R-0	Reserved

其中 SPI_DATA_STRB_EXTND 用来控制触发事件源，该 bit 配合 CLB_SPI_DATA_CTRL_HI 寄存器的 STRB 位使用，可以扩展到 64 个源：为 0 时为低 32 位触发源，为 1 时为高 32 位触发源。

1.15. HIC 模块

1.15.1. 对内部数据访问位宽

对内部模块仅支持 32bit 的写操作。因此写操作需要配置为 32 位写操作。

1.15.2. 地址转换

友商地址转换：

Table 13-6. Address Translation for 8-bit Data Width Mode

HICMODECR. DW_MODE	Host Port Address	Master Port Address {HICDBADDR.BASE_ADDR[31:7],A[6:1]}
8-bit ("0")	0x0	0x0
	0x1	0x0
	0x2	0x1
	0x3	0x1
	0x4	0x2
	0x5	0x2
	0x6	0x3
	0x7	0x3
	0x7F	0x3F
	0x80	0x40
	0x81	0x40
	0xFE	0x7F
	0xFF	0x7F

Table 13-7. Address Translation for 16-bit Data Width Mode

HICMODECR. DW_MODE	Host Port Address	Master Port Address{HICDBADDR.BASE_ADDR[31 :8],A[7:0]}
16-bit ("1")	0x0	0x0
	0x1	0x1
	0x2	0x2
	0x7F	0x7F
	0x80	0x80
	0x81	0x81
	0xFE	0xFE
	0xFF	0xFF

HX-280025C 地址转换

表 8bit 模式下地址转换

HICMODECR DW_MODE	HOST Port Address	Master Port Address Base_addr[31:8]A[7:0]
8bit(“0”)	0x0	0x0
	0x1	0x1
	0x2	0x2
	0x3	0x3
	0x4	0x4
	0x5	0x5
	0x6	0x6
	0x7	0x7
	0x7f	0x7f
	0x80	0x80
	0xfe	0xfe
	0xff	0xff

表 16 bit 模式下地址转换

HICMODECR DW_MODE	HOST Port Address	Master Port Address {Base_addr[31:9]A[7:0],1’ b0}
16bit(“0”)	0x0	0x0
	0x1	0x2
	0x2	0x4
	0x3	0x6
	0x4	0x8
	0x7e	0xFC

	0x7f	0xFE
	0x80	0x100
	0xfe	0x1Fc
	0xff	0x1FE

1.16. TMU 模块

1.16.1. __quadf32()的 intrinsics 差异

友商的实现形式：一条 intrinsic 可以同时得出 quad 值和 ratio 值：

float **__quadf32(** float *ratio*, float *y*, float *x* **);**	**QUADF32** *quadrant* , *ratio* , *y* , *x*	Return the quadrant value (0.0, +/-0.25, or +/-0.5) and the ratio of *x* and *y*, which are provided as per unit values.

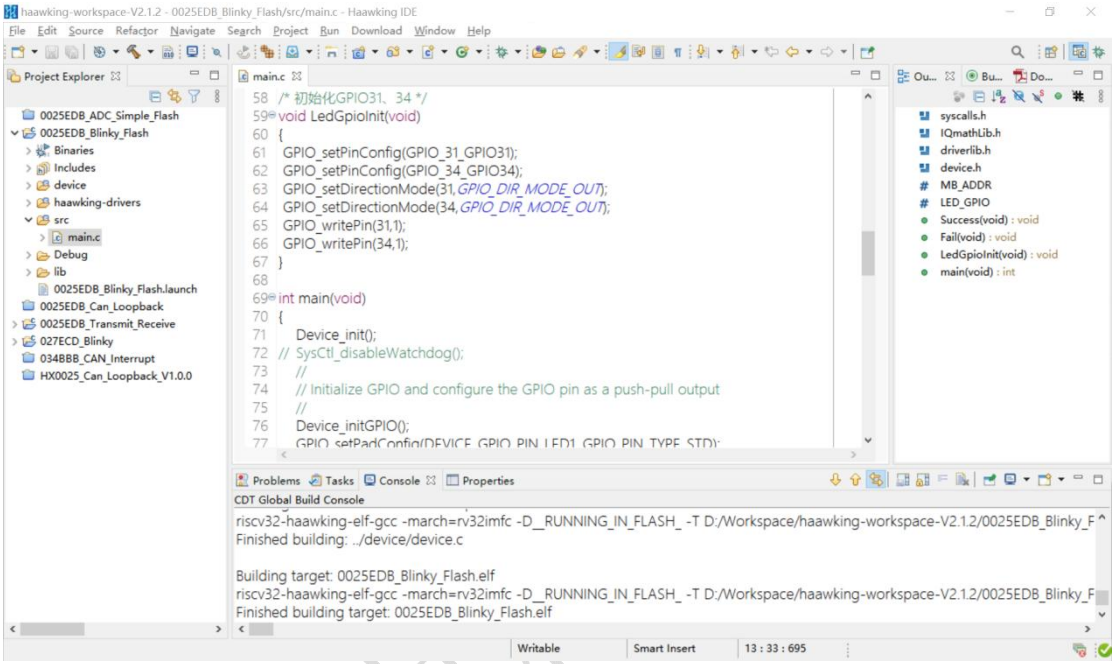
昊芯的实现方式：分为两个 intrinsics 函数__quadf32()、__quadf32_ratio()。

float __quadf32(float y, float x)：两个入参，返回值为 quad 象限值。
float __quadf32_ratio(float y, float x)：两个入参，返回值为 ratio 值。

2. 开发工具差异

Haawking HX2000 系列芯片采用中科昊芯自主研发的开发工具，当前版本使用习惯与友商 CCS5 之后的版本一致，包括常用的 CCS6 和 CCS10。

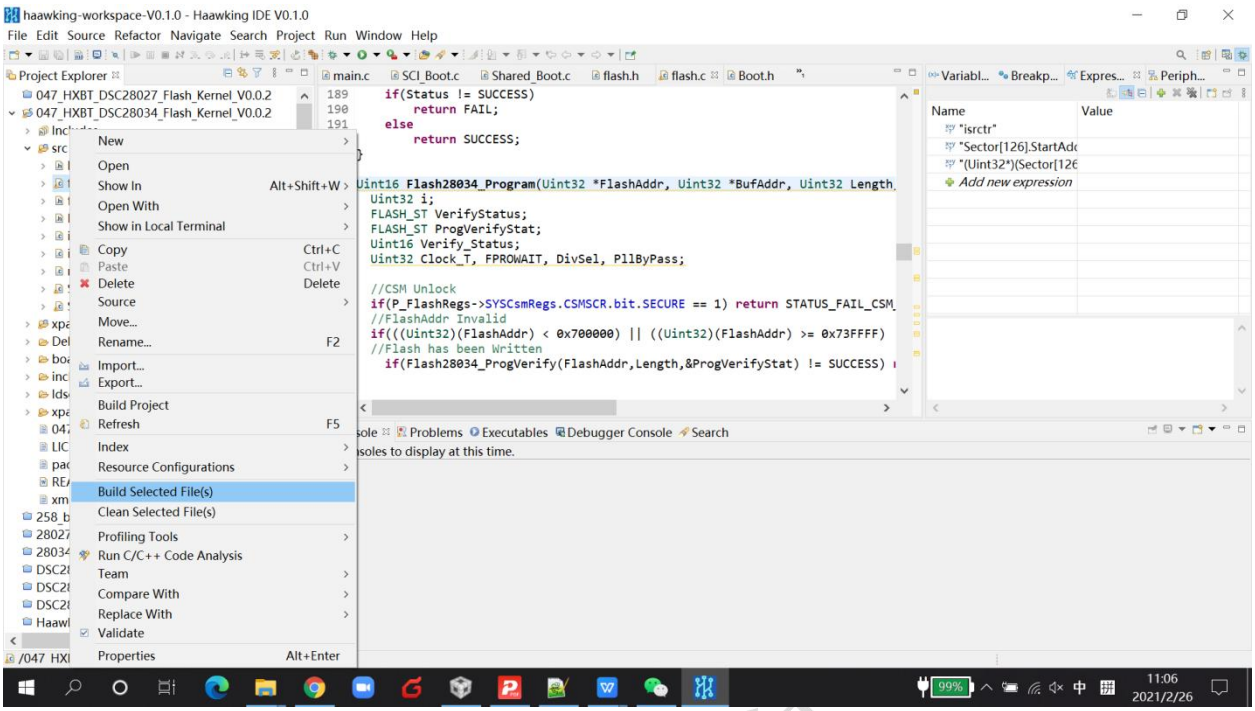
0025C 开发时，需要使用 **Haawking IDE v2.1.2 以及更高的版本**。



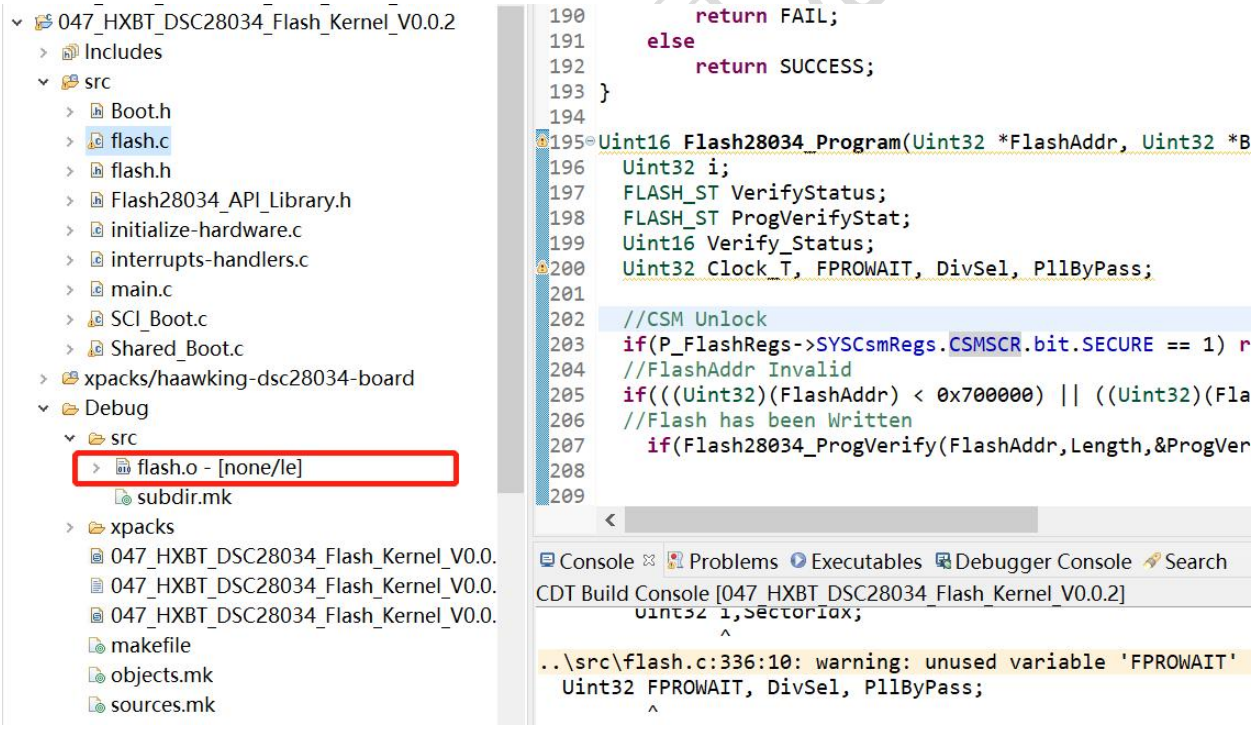
2.1. 编译单个源文件

Haawking IDE 支持单个源程序文件的编译，可以加速开发过程。

在 Haawking IDE 的项目索引栏中，右键选中单独编译的文件，比如 flash.c，在菜单中选择 Build Selected File(s)，就可以只编译 flash.c 文件。



编译完成以后，.o 文件在 Debug 目录下。



2.2. CodeStartBranch

友商提供的 CodeStartBranch.asm，该文件是 DSP 程序的入口文件，C 语言编程的时候，该文件能够执行 MAIN 函数前的操作，主要包括选择芯片 C27 或者 C28 地址模式，初始化相关存储器和缓冲区。

我们的开发工具中去掉了该文件，改为 common/syscall.c，完成同样的初始化功能。

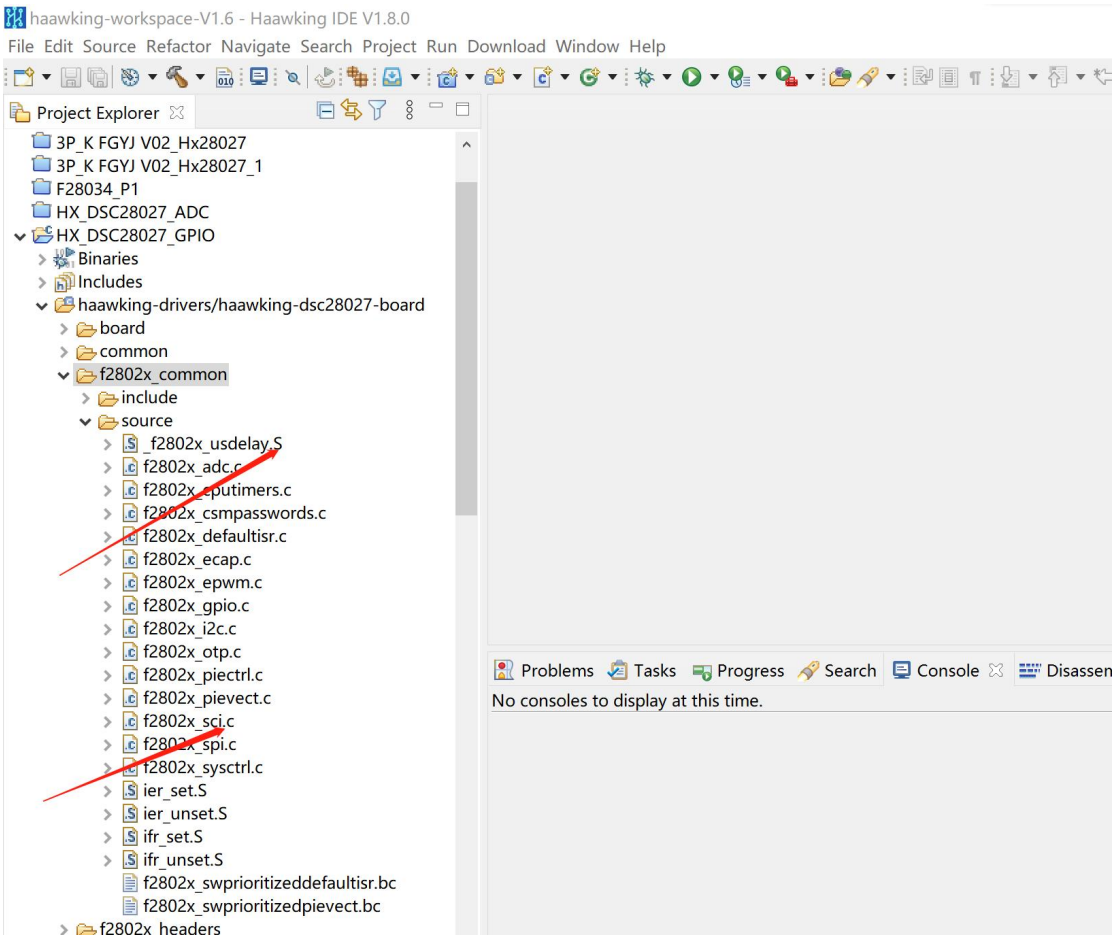
同时，对于部分需要在 RAM 中运行的程序，common/syscall.c 中会自动将代码从 Flash 中拷贝到 RAM 中，不需要用户手动拷贝。

2.3. IDE 支持的 C 和汇编源文件的后缀名

对于 Haawking IDE，汇编文件的后缀名必须是大写的.S 格式，例如 crt.S；C 的源文件的后缀名必须是小写的.c；例如 Main.c；

所以在移植的时候，如果看到友商移植过来的文件名称里有.C 的时候，一定要改成.c。

如下图：Haawking IDE 支持如下：



3. 编程模式差异

中科昊芯投入了很多资源和精力，为客户提供了和友商一样的位域驱动库和 Driver Lib 库，在编程的时候，可以完全跟友商一样。

```
64
65   InitPieVectTable();
66
67   EALLOW;  // This is needed to write to EALLOW protected registers
68   PieVectTable.ADCINT1 = &adc_isr;
69   PieVectTable.ECAP1_INT = &ecap1_isr;
70   PieVectTable.SCIRXINTA = &sciaRxFifoIsr;
71   PieVectTable.XINT1 = &Drxint1_isr;
72   EDIS;    // This is needed to disable write to EALLOW protected registers
73
74   EALLOW;
75   SysCtrlRegs.PCLKCR0.bit.TBCLKSYNC = 0;
76   EDIS;
77
78 ;//  InitEPwm1Example();
79
80
81   EALLOW;
82   SysCtrlRegs.PCLKCR0.bit.TBCLKSYNC = 1;
83   EDIS;
```

3.1. CMD 文件

中科昊芯提供链接文件时 LD 格式，遵循 GNU LD 标准；同时，根据友商芯片的内存映射进行修改。

关于 GNU LD 的标准，请参考

https://sourceware.org/binutils/docs/ld/Scripts.html

```
18 MEMORY
19 {
20   /*PAGE 0:*/
21   PRAMM0 (rwx)   : ORIGIN = 0x0      , LENGTH = 0x800  /* RAM M0*/
22   PRAMM1 (rwx)   : ORIGIN = 0x800    , LENGTH = 0x800  /* RAM M1*/
23
24   AdcRegs_FILE (rw)   :  ORIGIN  = 0x00001400 , LENGTH = 156
25
26   AdcResult_FILE (rw)   :  ORIGIN  = 0x0000149c , LENGTH = 64
27
28   CpuTimer0Regs_FILE  (rw) : ORIGIN = 0x00001800 ,  LENGTH = 16
29   CpuTimer1Regs_FILE  (rw) : ORIGIN = 0x00001810 ,  LENGTH = 16
30   CpuTimer2Regs_FILE  (rw) : ORIGIN = 0x00001820 ,  LENGTH = 16
31
32   PieCtrlRegs_FILE(rw)  : ORIGIN  = 0x00001900 , LENGTH = 104
33
34   XIntruptRegs_FILE (rw) :   ORIGIN  = 0x00001968 , LENGTH = 124
35
36   PieVectTable_FILE (rw) :   ORIGIN  = 0x00001A00 , LENGTH = 448
37
38   PieEmuRegs_FILE (rw)   :  ORIGIN = 0x1BC0, LENGTH = 4
39
40   DmaRegs_FILE (rw)      :  ORIGIN  = 0x00001C00 , LENGTH = 1024
41
```

分区名字 起始地址 分区长度

段设置，可以将不同的代码段，放到不同的存储分区。NOLOAD 属性和 LOAD 属性使用，与友商保持一致。

```
100
101 SECTIONS
102 {
103
104   .AdcRegs(NOLOAD) :  { *(.AdcRegs) }  >  AdcRegs_FILE
105
106   .AdcResult(NOLOAD) : { *(.AdcResult)  }  > AdcResult_FILE
107
108    .ECap1Regs(NOLOAD) : { *(.ECap1Regs) }  > ECap1Regs_FILE
109
110   .CpuTimer0Regs(NOLOAD) : { *(.CpuTimer0Regs) }     >  CpuTimer0Regs_FILE
111   .CpuTimer1Regs(NOLOAD) : { *(.CpuTimer1Regs) }     >  CpuTimer1Regs_FILE
112   .CpuTimer2Regs(NOLOAD) : { *(.CpuTimer2Regs) }     >  CpuTimer2Regs_FILE
113
114
115   .PieCtrlRegs(NOLOAD) : { *(.PieCtrlRegs)  }  > PieCtrlRegs_FILE
116
117   .PieVectTable(NOLOAD) : { *(.PieVectTable)  } >  PieVectTable_FILE
118
119   .XIntruptRegs(NOLOAD) :  { *(.XIntruptRegs) }  > XIntruptRegs_FILE
120
121   .PieEmuRegs(NOLOAD)    :  { *(.PieEmuRegs) } > PieEmuRegs_FILE
122
123   .EQep1Regs(NOLOAD)  : { *(.EQep1Regs) }  >  EQep1Regs_FILE
124
125   .SpiaRegs(NOLOAD) : { *(.SpiaRegs) } > SpiaRegs_FILE
```

段名字　段属性　段所在区

3.2. RAM 运行

友商的 2000 系列编译器支持 Progma 语法，告诉编译器如果修改一个特定函数、目标文件或者一段代码的属性，例如 CODE_SECTION，就是为某一个函数分配 Section，使用方式如下：

```
#pragma CODE_SECTION(funcA, "codeA")
```

HX2000 为了实现同样的功能，通过定义一个宏，如下：

```
#define CODE_SECTION(v) __attribute__((section(v)))
```

其中，v 表示将某一函数或者全局变量指定到的 SECTION 名。使用的时候，可以参考如下格式：

```
volatile int CODE_SECTION(“dmaisr”) array[10];
void CODE_SECTION(“dmaisr”) dma_isr(){    }
```

只需要在函数和变量定义的时候进行属性指定即可，如，想将函数 dma_isr 放在 L0 中运行，则可以使用如下方式完成：

```
void CODE_SECTION(“L0”) dma_isr(){    }
```

CODE_SECTION 宏已经在驱动中定义，用户可以直接使用。

3.3. 中断服务程序

友商的编译器支持 INTERRUPT 或者__interrupt(两个下划线)关键词，将中断服务程序映射到中断向量表，分配中断服务程序的地址。

```
INTERRUPT void  adc_isr(void)
```

Haawking IDE 也提供了实现同样功能的宏定义，使用方式如下：

```
INTERRUPT void  adc_isr(void)
```

如果中断服务程序需要在 ram 中运行，则可以与 3.2 节同时使用：

```
INTERRUPT void CODE_SECTION("L0") adc_isr(void)
```

昊芯的芯片在使用嵌套中断的时候和友商是一样的。

举例如下（下面的例子中，低优先级的中断服务函数 INT13_ISR()可以被高优先级中断 INT1.4 打断）：

```
1.  __interrupt void INT13_ISR(void)
2.  {
3.    Uint16_t  TemPIEIER;
4.    Uint32_t  TemIER;
5.
6.  //保存 PIEIER 寄存器数值
7.    TemPIEIER= PieCtrlRegs.PIEIER1.all;
8.  // 保存 IER 寄存器数值
9.    TemIER= IER;
10.
11.  // 在 IER 寄存器层级，使能嵌套中断 XINT1 所在组的中断
12.    IER |= M_INT1;
13.  // 在 IER 寄存器层级，屏蔽嵌套中断 XINT1 所在组之外组的中断
14.    IER &= M_INT1;
15.
16.  // 在 PIEIER 寄存器层级，使能 XINT1 对应中断，并屏蔽该组其它中断
17.  // 注意：初始化中已经将 PieCtrlRegs.PIEIER1.bit.INTx4 = 1;
18.    PieCtrlRegs.PIEIER1.all &= 0x40;
19.
20.  // TINT0 通过 PIE 向 CPU 发出中断请求后，硬件自动将 ACK1=1;
21.  // 而 XINT1 也位于 Group 1 组内，需要再次打开 ACK 位
22.    PieCtrlRegs.PIEACK.all = 0xFFFF;
23.
24.  // 等待 PIE 和 CPU 中断通道准备就绪，用于防止误操作
25.    asm("NOP");
26.
27.  // 进入 timer0_isr 函数后，硬件自动关闭了中断总开关
28.  // 此处打开中断总开关
29.  EINT;
30.
```

```
31.    /*插入 timer0 中断代码，此处可被打断*/
32.    /*    中断代码    */
33.
34.    // 关闭中断总开关，此后不可被嵌套，这是嵌套前的状态
35.    DINT;
36.
37.    // 恢复嵌套前 PIEIER 寄存器的状态
38.    PieCtrlRegs.PIEIER1.all = TemPIEIER;
39.
40.    // 恢复嵌套前 IER 寄存器的状态
41.    IER = TemIER;
42.
43.    // 清除 timer0 中断标志位，以便下次触发中断
44.    Cputimer0Regs.TCR.bit.TIF = 1;
45.    }
```

3.4. RPT 指令

在友商的指令集中，有 RPT 指令，一般会用来进行延时函数的实现，比如

```
asm(" RPT #5 || NOP ");
```

在中科昊芯的 HX2000 系列芯片中，也提供了实现同样功能的指令，实现方式如下：需要注意的时，RPTI 指令不允许被打断。如果执行过程中被打断，代码行为未知。

```
asm volatile (".align 2;RPTI 5,4;NOP");
```

3.5. ENPIE

友商有一个 PIE 的使能位，使用的时候，需要提前配置，如

```
PieCtrlRegs.PIECTRL.bit.ENPIE = 1;
```

中科昊芯的 DSC 系列产品，默认使能 PIE 模块，无需配置，如果有，需注释掉。

当前版本，我们把 PieCtrlRegs.PIECTRL.bit.ENPIE 这个位加上了，但是其实是没有意义的。

3.6. Uint 类型

友商 2000 系列芯片是 32 位 DSP，中科昊芯 HX2000 系列芯片是 32 位 RISC-V DSP，对 int 类型的解释不一样，使用中需要注意。

对于之前友商程序中有定义 int 类型的数组或者变量，可以手动改成 int16 来降低存储空间占用。

RISC-V 32 位处理器支持的数据类型如下：

ilp32

- int, long, pointers are 32bit
- long long is 64bit
- char is 8bit
- short is 16bit

中科昊芯 Haawking IDE 建议使用类型+位长的数据类型格式：

```
typedef  unsigned char  Uint8;
typedef  unsigned short int  Uint16;
typedef  unsigned int  Uint32;
typedef unsigned long long int Uint64;
typedef  signed char  Int8;
typedef  signed short int Int16;
typedef  signed int Int32;
typedef signed long long int Int64;
typedef  unsigned char  uint8;
typedef  unsigned short int  uint16;
typedef  unsigned int  uint32;
typedef unsigned long long int uint64;
typedef  signed char  int8;
typedef  signed short int int16;
typedef  signed int int32;
typedef signed long long int int64;
```

上述定义，已经在中科昊芯提供的驱动文件中（hx_rv32_type.h），完全兼容友商的定义，用户无需重新定义。

```
typedef int                 int16;
typedef long                int32;
typedef long long           int64;
typedef unsigned int        Uint16;
typedef unsigned long       Uint32;
typedef unsigned long long  Uint64;
typedef float               float32;
typedef long double         float64;
```

在跟用户交流的过程中，我们发现有用户会重新定义

```
typedef long long     llong;
typedef unsigned int   Uint;
typedef unsigned long   Ulong;
typedef unsigned long long   Ullong;
```

因此，用户在使用的时候，需要特别注意数据类型的位宽。

3.7. 因为数据类型差异需要做的一些特殊处理

3.7.1. sizeof()使用（8 位寻址）

sizeof()用于返回一个变量或者类型的大小， 返回的结果是一个整数，在 Haawking IDE 中，sizeof()返回结果的单位为 1 个字节；但在 CCS 中，sizeof()返回的结果的单位是 2 个字节，这是由于编译器不同导致。
因此， Haawking IDE 中 sizeof()返回值的结果等于 CCS 中 sizeof()返回值结果 * 2；

例如：在 CCS 中：

```
int x = sizeof(uint16_val);
```

Haawking IDE 中：

```
int x = sizeof(uint16_val)/2;
```

3.7.2. memcpy(void* dest, const void* src, size_t len)的使用 （基于 Haawking 8 位寻址）

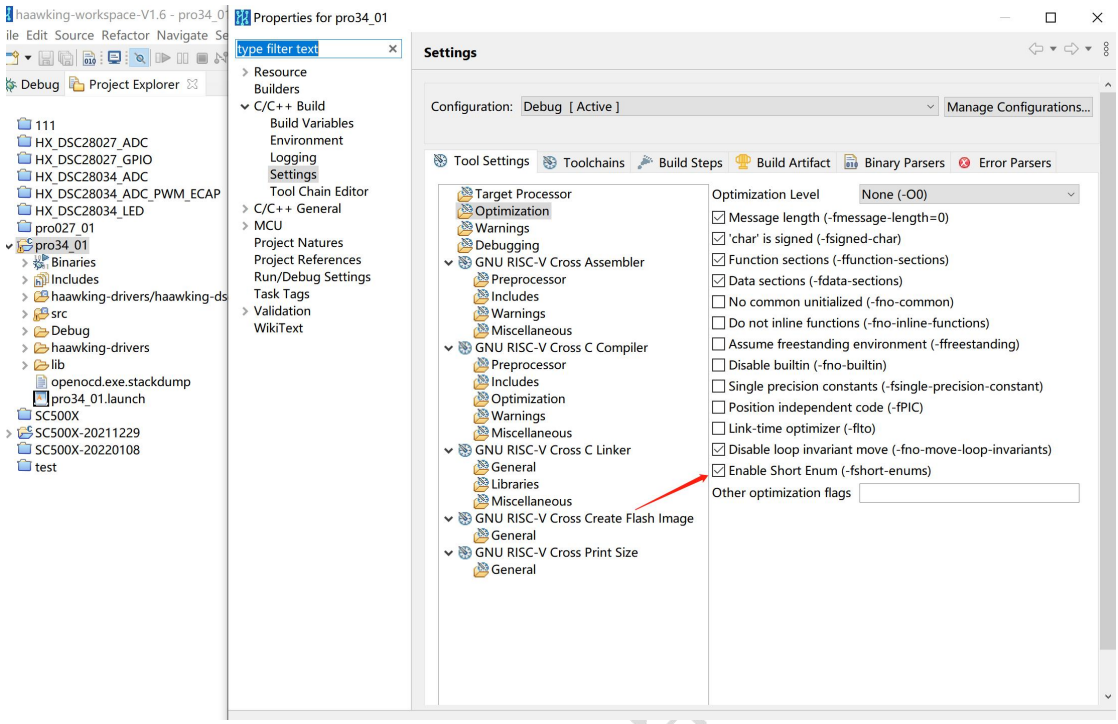
在 CCS 中，memcpy（）是按每个元素 16 位拷贝，在 Hawwking IDE 中，是按每个元素 8 位拷贝；
在 CCS 中：`memcpy(&gSendToMotor2MsDataBuff[100],&funcCode.code.PosSetAdd,8);`
在 Haawking IDE 中：

```
memcpy(&gSendToMotor2MsDataBuff[100], &funcCode.code.PosSetAdd, (8*2));
```

3.7.3. typedef enum 使用（8 位寻址）

CCS 中一个枚举类型数据默认占用 2 个字节， 在 Haawking IDE 中，一个枚举类型数据默认占用 4 个字节。

当需要两个字节的枚举的时候，首先需要修改结构体的定义， 然后勾选优化面板中的 -fshort-enums。如下图：



例如：

在 CCS 中

```
typedef enum CONTROL_MOTOR_TYPE_ENUM_DEF{
    ASYNC_SVC,
    ASYNC_FVC,
    ASYNC_VF,
 SYNC_SVC = 10,
    SYNC_FVC,
    SYNC_VF,
 DC_CONTROL = 20,
    RUN_SYNC_TUNE
}CONTROL_MOTOR_TYPE_ENUM;
```

在 Hawwking IDE 中

```
typedef enum CONTROL_MOTOR_TYPE_ENUM_DEF{
    ASYNC_SVC,
    ASYNC_FVC,
    ASYNC_VF,
 SYNC_SVC = 10,
    SYNC_FVC,
    SYNC_VF,
 DC_CONTROL = 20,
 DC_CONTROL_Hx = 1000,
    RUN_SYNC_TUNE
}CONTROL_MOTOR_TYPE_ENUM;
```

以上根据实际需要来进行修改；

同时，因为中科昊芯 HX2000 系列 DSP 芯片，支持 8 位数据类型，如果有些变量或者数组的取值范围在 Char 类型的表示范围之内，可以减少程序代码量。