

UM8G006A SDK 使用说明

版本: V1.1



广芯微电子（广州）股份有限公司

<http://www.unicmicro.com/>

条款协议

本文档的所有部分，其著作权归广芯微电子（广州）股份有限公司（以下简称广芯微电子）所有，未经广芯微电子授权许可，任何个人及组织不得复制、转载、仿制本文档的全部或部分组件。本文档没有任何形式的担保、立场表达或其他暗示，若有任何因本文档或其中提及的产品所有资讯所引起的直接或间接损失，广芯微电子及所属员工恕不为其担保任何责任。除此以外，本文档所提到的产品规格及资讯仅供参考，内容亦会随时更新，恕不另行通知。

1. 本文档中所记载的关于电路、软件和其他相关信息仅用于说明半导体产品的操作和应用实例。
用户如在设备设计中应用本文档中的电路、软件和相关信息，请自行负责。对于用户或第三方因使用上述电路、软件或信息而遭受的任何损失，广芯微电子不承担任何责任。
2. 在准备本文档所记载的信息的过程中，广芯微电子已尽量做到合理注意，但是，广芯微电子并不保证这些信息都是准确无误的。用户因本文档中所记载的信息的错误或遗漏而遭受的任何损失，广芯微电子不承担任何责任。
3. 对于因使用本文档中的广芯微电子产品或技术信息而造成的侵权行为或因此而侵犯第三方的专利、版权或其他知识产权的行为，广芯微电子不承担任何责任。本文档所记载的内容不应视为对广芯微电子或其他人所有的专利、版权或其他知识产权作出任何明示、默示或其它方式的许可及授权。
4. 使用本文档中记载的广芯微电子产品时，应在广芯微电子指定的范围内，特别是在最大额定值、电源工作电压范围、热辐射特性、安装条件以及其他产品特性的范围内使用。对于在上述指定范围之外使用广芯微电子产品而产生的故障或损失，广芯微电子不承担任何责任。
5. 虽然广芯微电子一直致力于提高广芯微电子产品的质量和可靠性，但是，半导体产品有其自身的具体特性，如一定的故障发生率以及在某些使用条件下会发生故障等。此外，广芯微电子产品均未进行防辐射设计。所以请采取安全保护措施，以避免当广芯微电子产品在发生故障而造成火灾时导致人身事故、伤害或损害的事故。例如进行软硬件安全设计（包括但不限于冗余设计、防火控制以及故障预防等）、适当的老化处理或其他适当的措施等。

目录

1	RCHDIV 配置说明	1
2	SYSDIV 配置说明	3
3	GPIO 配置说明	4
4	UART2 功能复用	7
5	编译目标选择	10
6	外部复位滤波时间设置	11
7	RCHTRIM 寄存器	12
8	版本修订	13

1 RCHDIV 配置说明

原 UM800Y RCHDIV 寄存器 (addr: EEH) 在 UM8G006A 中删除。

```

9 #if defined (UM800Y)
0     if ((rchxxm <= 24120000) && (rchxxm >= 23880000))    //实际值如果是24M的±0.5%以内
1     {
2         crystal_clk = 24000000 * 2;
3     }
4     else if (rchxxm == 0xFFFFFFFF)
5     {
6         crystal_clk = 24000000 * 2;
7     }
8     else
9     {
10        crystal_clk = rchxxm * 2;
11    }
12
13    switch (system_clk_hz)
14    {
15        case 24000000:
16            RCHDIV = (0x1 << 0);    //RCH 2分频
17            SYSDIV = (0x0 << 0);    //不分频
18            system_core_clock = system_clk_hz;
19            break;
20
21        case 16000000:
22            RCHDIV = (0x2 << 0);    //RCH 3分频
23            SYSDIV = (0x0 << 0);    //不分频
24            system_core_clock = system_clk_hz;
25            break;
26
27        case 12000000:
28            RCHDIV = (0x3 << 0);    //RCH 4分频
29            SYSDIV = (0x0 << 0);    //不分频
30            system_core_clock = system_clk_hz;
31            break;
32
33        case 4000000:
34            RCHDIV = (0x2 << 0);    //RCH 3分频
35            SYSDIV = (0x2 << 0);    //4分频
36            system_core_clock = system_clk_hz;
37            break;
38
39        default:
40            RCHDIV = (0x2 << 0);    //RCH 3分频
41            SYSDIV = (0x0 << 0);    //不分频
42            system_core_clock = crystal_clk / 3;
43            break;
44    }
45 #endif

```

SDK中通过宏定义保留UM800Y的RCHDIV寄存器配置

图 1-1: 保留 UM800Y 中 RCHDIV 寄存器配置

```
#if defined (UM800YA)
    if ((rchxxm <= 16080000) && (rchxxm >= 15920000))    //实际值如果是16M的±0.5%以内
    {
        crystal_clk = 16000000;
    }
    else if (rchxxm == 0xFFFFFFFF)
    {
        crystal_clk = 16000000;
    }
    else
    {
        crystal_clk = rchxxm;
    }

    switch (system_clk_hz)
    {
        case 16000000:
            SYSDIV = (0x0 << 0);    //不分频
            system_core_clock = system_clk_hz;
            break;

        case 4000000:
            SYSDIV = (0x2 << 0);    //默认为4分频
            system_core_clock = system_clk_hz;
            break;

        default:
            SYSDIV = (0x2 << 0);    //默认为4分频
            system_core_clock = crystal_clk / 4;
            break;
    }
#endif
```

取消RCHDIV寄存器配置

图 1-2: 取消 UM8G006A 中 RCHDIV 寄存器配置

2 SYSDIV 配置说明

SYSDIV 寄存器 (addr: D9H) 原 UM800Y 中默认值为 0，即不分频。现在 UM8G006A 中默认值为 010'b，HSCLK/4 分频输出，如下图所示。

4.1.37 SYSDIV

D9H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
SYSDIV	-	-	-	-	-	SYSDIV[2]	SYSDIV[1]	SYSDIV[0]
读/写	读	读	读	读	读	读/写	读/写	读/写
复位值	0	0	0	0	0	0	1	0
位编号	位符号	说明						
7-3	-	-						

2-0	SYSDIV[1:0]	高频时钟（RC16M 或者 XCLK）分频控制，分频后输出作为系统时钟。 000: HSCLK 不分频; 001: HSCLK/2 分频输出; 010: HSCLK /4 分频输出; 011: HSCLK /8 分频输出; 100: HSCLK /16 分频输出; 101: HSCLK /32 分频输出; 110: HSCLK /64 分频输出; 111: HSCLK /128 分频输出。
-----	-------------	--

图 2-1: UM8G006A 中 SYSDIV 寄存器

3 GPIO 配置说明

UM8G006A 的 GPIO 除 P27、P10、P12 外其它 pin 脚均默认为输入禁止状态，而原 UM800Y 的 GPIO 默认为输入开启状态，软件做更新时需特别注意。尤其当 GPIO 为 IO 口时，即便某个 IO 口作为输出功能，当进行 GPIO 口的读、改、写操作时，如果该 IO 没有开启输入使能，则当该 IO 本应为高电平时，实际读入为高阻态低电平。

UM800YA 用户手册

GPIO (I/O 端口)

7.4.17 P0_IE

C000H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
P0_IE	-	-	-	P0_4_IE	P0_3_IE	P0_2_IE	P0_1_IE	P0_0_IE
读/写	读	读	读	读/写	读/写	读/写	读/写	读/写
复位值	0	0	0	0	0	0	0	0
位编号	位符号	说明						
7-5	-	-						
4	P0_4_IE	P0_4 IO 输入控制位： 0: P0_4 禁止输入； 1: P0_4 开启输入。						
3	P0_3_IE	P0_3 IO 输入控制位： 0: P0_3 禁止输入； 1: P0_3 开启输入。						
2	P0_2_IE	P0_2 IO 输入控制位： 0: P0_2 禁止输入； 1: P0_2 开启输入。						
1	P0_1_IE	P0_1 IO 输入控制位： 0: P0_1 禁止输入； 1: P0_1 开启输入。						
0	P0_0_IE	P0_0 IO 输入控制位： 0: P0_0 禁止输入； 1: P0_0 开启输入。						

图 3-1：P0_IE 输入使能位说明

7.4.18 P1_IE

C001H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
P1_IE	-	-	P1_5_IE	P1_4_IE	P1_3_IE	P1_2_IE	P1_1_IE	P1_0_IE
读/写	读	读	读/写	读/写	读/写	读/写	读/写	读/写
复位值	0	0	0	0	0	1	0	1

位编号	位符号	说明
7-6	-	-
5	P1_5_IE	P1_5 IO 输入控制位: 0: P1_5 禁止输入; 1: P1_5 开启输入。

4	P1_4_IE	P1_4 IO 输入控制位: 0: P1_4 禁止输入; 1: P1_4 开启输入。
3	P1_3_IE	P1_3 IO 输入控制位: 0: P1_3 禁止输入; 1: P1_3 开启输入。
2	P1_2_IE	P1_2 IO 输入控制位: 0: P1_2 禁止输入; 1: P1_2 开启输入。
1	P1_1_IE	P1_1 IO 输入控制位: 0: P1_1 禁止输入; 1: P1_1 开启输入。
0	P1_0_IE	P1_0 IO 输入控制位: 0: P1_0 禁止输入; 1: P1_0 开启输入。

图 3-2: P1_IE 输入使能位说明

7.4.19 P2_IE

C002H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
P2_IE	P2_7_IE	P2_6_IE	P2_5_IE	-	P2_3_IE	P2_2_IE	-	P2_0_IE
读/写	读/写	读/写	读/写	读	读/写	读/写	读	读/写
复位值	1	0	0	0	0	0	0	0
位编号	位符号	说明						
7	P2_7_IE	P2_7 IO 输入控制位: 0: P2_7 禁止输入; 1: P2_7 开启输入。						
6	P2_6_IE	P2_6 IO 输入控制位: 0: P2_6 禁止输入; 1: P2_6 开启输入。						
5	P2_5_IE	P2_5 IO 输入控制位: 0: P2_5 禁止输入; 1: P2_5 开启输入。						
4	-	-						
3	P2_3_IE	P2_3 IO 输入控制位: 0: P2_3 禁止输入; 1: P2_3 开启输入。						

2	P2_2_IE	P2_2 IO 输入控制位: 0: P2_2 禁止输入; 1: P2_2 开启输入。
1	-	-
0	P2_0_IE	P2_0 IO 输入控制位: 0: P2_0 禁止输入; 1: P2_0 开启输入。

图 3-3: P2_IE 输入使能位说明

程序示例:

```
8  /*****
9  * Function      : i2c_master_init
0  * Description   : i2c_master_init    I2C主机初始化
1  * Input        : uint32_t i2c_speed  I2C速率 = (fSYSCLK)/(4*(CLK_DIV+1))
2  * Output       : none
3  * Return       : none
4  *****/
5  void i2c_master_init(uint32_t i2c_speed) 以SCL, SDA为示例
6  {
7      REG_P0_IE |= (1 << 4); //P04 开启输入
8      REG_P1_IE |= (1 << 0); //P10 开启输入
9
10     PCLK0  |= (1 << 7); //打开I2C模块时钟
11     PRESET0 |= (1 << 7); //释放I2C复位
12
13     P0PU &= ~(1 << 4); //P04上拉
14     P1PU &= ~(1 << 0); //P10上拉
15
16     REG_P04_CFG = 0x03; //I2C_SDA
17     REG_P10_CFG = 0x04; //I2C_SCL
18
19     REG_I2C_CLK_DIV = 16000000 / (4 * i2c_speed) - 1; //I2C速率 = (fSYSCLK)/(4*(CLK_DIV+1))
20     REG_I2C_CR1 |= (1 << 5); //使能开漏模式
21     REG_I2C_CR0 |= MEN; //I2C使能
22 }
```

图 3-4: GPIO 输入使能程序示例

4 UART2 功能复用

UM8G006A 增加 P10 复用 UART2 RX 功能, P11 复用 UART2_TX 功能。

表 3-1: 引脚功能复用 UM800YA

封装引脚编号		Config	Pxx_CFG[2:0]							
TSSOP20	QFN20		0	1	2	3	4	5	6	7
1	18	VREF	P2.5	UART3_TX	SPI_CSN	I2C_SCL	GTIMER0_CH	GTIMER0_BKE	BEEPER	UART0_RX
2	19	AIN5	P2.6	UART0_TX	UART2_TX	SPI_MISO	LPOUT1	GTIMER1_CH	GTIMER2_CH	-
3	20	AIN6	P2.7	UART0_RX	UART2_RX	SPI_MOSI	I2C_SCL	GTIMER1_CHN	GTIMER2_BKE	BEEPER
4	1	NRST/P0.2	-	-	-	-	-	-	-	-
5	2	XIN	P0.0	UART2_RX	SPI_CSN	LPOUT0	GTIMER1_CHN	GTIMER2_BKE	-	-
6	3	XOUT	P0.1	UART2_TX	SPI_SCK	I2C_SDA	LPOUT1	GTIMER0_BKE	GTIMER2_CHN	-
7	4	VSS	-	-	-	-	-	-	-	-
8	5	VDDCORE	-	-	-	-	-	-	-	-
9	6	VDDH	-	-	-	-	-	-	-	-
10	7	LPT_OUT	P0.3	CLKOUT	UART2_TX	UART3_RX	SPI_CSN	LPOUT0	GTIMER1_CH	-
11	8	-	P0.4	UART2_RX	SPI_SCK	I2C_SDA	LPOUT1	GTIMER1_BKE	GTIMER2_CHN	-
12	9	-	P1.0	UART1_RX	UART2_TX	PWM0	I2C_SCL	LP0_IN	GTIMER2_CH	UART2_RX
13	10	-	P1.1	UART1_TX	UART3_RX	PWM1	SPI_MISO	LP0_TRG	GTIMER1_CHN	PWM0(UART2_TX)
14	11	AIN0	P1.2	UART0_RX	UART3_TX	PWM2	LP0_CAP	GTIMER1_CH	PWM1	-
15	12	-	P1.3	UART0_TX	UART2_RX	SPI_SCK	I2C_SDA	LP0_IN	GTIMER0_CH	PWM2
16	13	AIN0	P1.4	UART1_RX	PWM2	SPI_MOSI	LP0_TRG	GTIMER0_CHN	GTIMER1_BKE	-
17	14	AIN1	P1.5	UART1_TX	PWM1	SPI_MISO	GTIMER0_CH	GTIMER1_BKE	GTIMER2_CH	LP0_CAP1
18	15	AIN2	P2.0	UART3_RX	PWM0	SPI_MOSI	I2C_SCL	LPOUT0	GTIMER0_CHN	-
19	16	AIN3	P2.2	UART3_TX	SPI_CSN	SPI_MISO	I2C_SDA	GTIMER0_BKE	GTIMER2_CHN	LP0_CAP1
20	17	AIN4	P2.3	UART3_RX	SPI_SCK	SPI_MOSI	LP0_CAP	GTIMER0_CHN	GTIMER2_BKE	-

图 4-1: UM8G006A 引脚功能复用表

3.2 引脚复用

表 3-1: 引脚功能复用

UM800Y

引脚编号		Config	Pxx_CFG[2:0]							
TSSOP20	QFN20		0	1	2	3	4	5	6	7
1	18	VREF	P2.5	UART3_TX	SPI_CSN	I2C_SCL	GTIMER0_CH	GTIMER0_BKE	BEEPER	UART0_RX
2	19	AIN5	P2.6	UART0_TX	UART2_TX	SPI_MISO	LPOUT1	GTIMER1_CH	GTIMER2_CH	-
3	20	AIN6	P2.7	UART0_RX	UART2_RX	SPI_MOSI	I2C_SCL	GTIMER1_CHN	GTIMER2_BKE	BEEPER
4	1	NRST/P0.2	-	-	-	-	-	-	-	-
5	2	XIN	P0.0	UART2_RX	SPI_CSN	LPOUT0	GTIMER1_CHN	GTIMER2_BKE	CAN_TX	-
6	3	XOUT	P0.1	UART2_TX	SPI_SCK	I2C_SDA	LPOUT1	GTIMER0_BKE	GTIMER2_CHN	CAN_RX
7	4	VSS	-	-	-	-	-	-	-	-
8	5	VDDCORE	-	-	-	-	-	-	-	-
9	6	VDDH	-	-	-	-	-	-	-	-
10	7	LPT_OUT	P0.3	CLKOUT	UART2_TX	UART3_RX	SPI_CSN	LPOUT0	GTIMER1_CH	CAN_TX
11	8	-	P0.4	UART2_RX	SPI_SCK	I2C_SDA	LPOUT1	GTIMER1_BKE	GTIMER2_CHN	CAN_RX
12	9	-	P1.0	UART1_RX	UART2_TX	PWM0	I2C_SCL	LP0_IN	GTIMER2_CH	-
13	10	-	P1.1	UART1_TX	UART3_RX	PWM1	SPI_MISO	LP0_TRG	GTIMER1_CHN	PWM0
14	11	AIN0	P1.2	UART0_RX	UART3_TX	PWM2	LP0_CAP	GTIMER1_CH	PWM1	CAN_TX
15	12	-	P1.3	UART0_TX	UART2_RX	SPI_SCK	I2C_SDA	LP0_IN	GTIMER0_CH	PWM2
16	13	AIN0	P1.4	UART1_RX	PWM2	SPI_MOSI	LP0_TRG	GTIMER0_CHN	GTIMER1_BKE	CAN_RX
17	14	AIN1	P1.5	UART1_TX	PWM1	SPI_MISO	GTIMER0_CH	GTIMER1_BKE	GTIMER2_CH	LP0_CAP1
18	15	AIN2	P2.0	UART3_RX	PWM0	SPI_MOSI	I2C_SCL	LPOUT0	GTIMER0_CHN	-
19	16	AIN3	P2.2	UART3_TX	SPI_CSN	SPI_MISO	I2C_SDA	GTIMER0_BKE	GTIMER2_CHN	LP0_CAP1
20	17	AIN4	P2.3	UART3_RX	SPI_SCK	SPI_MOSI	LP0_CAP	GTIMER0_CHN	GTIMER2_BKE	CAN_RX

图 4-2: UM800Y 引脚功能复用表

7.4.5 P10_CFG

C018H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
P10_CFG	-	-	-	-	-	P10_SEL		
读/写	读	读	读	读	读	读/写		
复位值	0	0	0	0	0	0x04		
位编号	位符号	说明						
7-3	-	-						
2-0	P10_SEL	000: P1_0						
		001: UART1_RX						
		010: UART2_TX						
		011: PWM0						
		100: I2C_SCL						
		101: LP0_IN						
		110: GTIMER2_CH						
		111: UART2_RX						

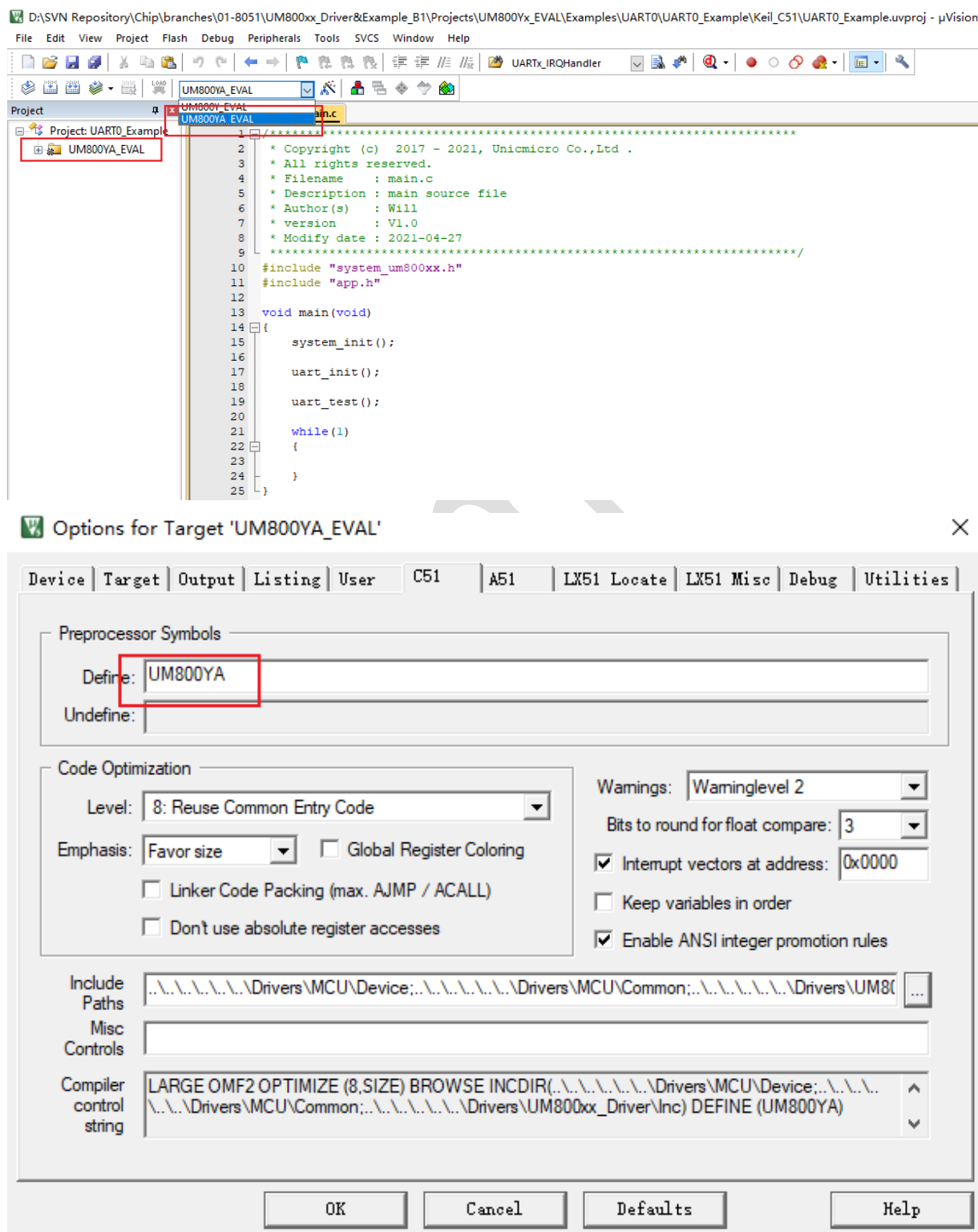
7.4.6 P11_CFG

C019H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
P11_CFG	-	-	-	-	P11_SEL			
读/写	读	读	读	读	读/写			
复位值	0	0	0	0	0			
位编号	位符号	说明						
7-3	-	-						
3-0	P11_SEL	0000: P1_1						
		0001: UART1_TX						
		0010: UART3_RX						
		0011: PWM1						
		0100: SPI_MISO						
		0101: LP0_TRG						
		0110: GTIMER1_CHN						
		0111: PWM0						
		1000~1110: 保留						
		1111: UART2_TX						

图 4-3：UM8G006A 中 UART2_RX/UART2_TX 功能

5 编译目标选择

UM800xx_Driver & Example_V2.2 SDK 编译目标选择如下图所示:



6 外部复位滤波时间设置

外部复位滤波时间设置位段取消，默认为 20ms 固定外部复位滤波时间。

4.1.24 ESTCR

B2H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ESTCR	-	-	-	-	ERSTEN	ERSTLVT	ERSTLVEN	
读/写	读	读	读	读	读/写	读/写	读/写	读/写
复位值	0	0	0	0	1	0	0	0

位编号	位符号	说明
7-4	-	-
3	ERSTEN	外部复位引脚使能位： 1：外部复位功能使能； 0：外部复位功能禁止。
2-1	ERSTLVT	外部复位滤波时间设置： 11：2 个 RCL38K 时钟周期； 10：15 个 RCL38K 时钟周期； 01：23 个 RCL38K 时钟周期； 00：30 个 RCL38K 时钟周期。
0	ERSTLVEN	1：外部复位滤波使能； 0：外部复位滤波禁止。

4.1.25 XTHCTR

4.1.24 ESTCR

B2H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ESTCR	-	-	-	-	ERSTEN	-	-	ERSTLVEN
读/写	读	读	读	读	读/写	读	读	读/写
复位值	0	0	0	0	1	0	0	0

位编号	位符号	说明
7-4	-	-
3	ERSTEN	外部复位引脚使能位： 1：外部复位功能使能； 0：外部复位功能禁止。
2-1	-	-
0	ERSTLVEN	1：外部复位滤波使能； 0：外部复位滤波禁止。

4.1.25 XTHCTR

7 RCHTRIM 寄存器

UM8G006A 变更了 RCHTRIMH 及 RCHTRIML 寄存器配置说明:

1. RCHTRIMH 寄存器说明由“片内高频 RCH 频率修调值高位”变为“片内高频 RCH 精调值”。
2. RCHTRIML 寄存器删除了 bit1 位段“片内高频 RCH 频率修调值低位”；原 bit4-7 位段由“片内高频 RCH 温漂修调值”变为“片内高频 RCH 粗调值”。

[illegible]

8 版本修订

版本	日期	描述
V1.0	2025.04.25	初始版
V1.1	2025.05.20	增加下面 2 小节的描述： 6.外部复位滤波时间设置 7.RCHTRIM 寄存器