Gowin OC8051 单片机用户指南

山东高云半导体科技有限公司

嵌入式系统研发部

2024-09-26

免责声明

版本信息

日期	版本	描述
2024/09/26	1.0	初始版本。

目录

1 关于本手册

1.1 手册内容

本手册主要描述了 OC8051 单片机的功能描述、信号描述、硬件参数配置、软件寄存器配置、IDE 软件使用、参考设计和开发流程，旨在帮助用户快速掌握 OC8051 单片机的软硬件开发方法，节省开发时间，提高开发效率。

1.2 术语、缩略语

本手册中的相关术语、缩略语及相关释义如表 1-1 所示。

表 1-1 术语、缩略语

术语、缩略语	全称	含义
IDE	Integrated Development Environment	集成开发环境
UART	Universal Synchronous and Asynchronous Receiver/Transmitter	通用同步和异步接收器/发送器
PSW	Program Status Word	程序状态字
IE	Interrupt Enable	中断使能
IP	Interrupt Priority	中断优先级
DPL	Data Pointer Low	数据指针的低字节
DPH	Data Pointer High	数据指针的高字节
TMOD	Timer Mode	定时器模式
TCON	Timer Control	定时器控制
SCON	Serial Control	串行控制
SBUF	Serial Buffer	串行数据缓存
PCON	Power Control	电源控制
SP	Stack Pointer	堆栈指针
RCAP2L	Timer2 Reload and Capture Low	定时器 2 重载和捕获低字节
RCAP2H	Timer2 Reload and Capture High	定时器 2 重载和捕获高字节

1.3 技术支持与反馈

高云®半导体提供全方位技术支持，在使用过程中如有疑问或建议，可直接与公司联系：

网址：www.gowinsemi.com.cn

E-mail：support@gowinsemi.com

Tel：+86 755 8262 0391

2 功能描述

2.1 概述

OC8051 单片机是 MCS-51 系列的成员，最初由英特尔于 20 世纪 80 年代设计，自推出以来，8051 就广受欢迎，至今仍在低端嵌入式系统产品中有较多的市场应用。

8051 单片机的基本组件包括多个片内外设，如 I/O 端口、定时器/计数器，此外还有最高可达 256B 的片内数据存储器和最高可达 64KB 的片内程序存储器。

OC8051 作为单片机软核，突破了 8051 芯片在硬件方面的限制，可以根据信号宽度来拓展片内 ROM 与 RAM 的空间，以便运行更大的程序代码。还可以分离 P3 接口与外部中断输入、定时器/计数器输入，使其相互独立，释放了复用的引脚。

2.2 特性

OC8051 单片机的特性包括：

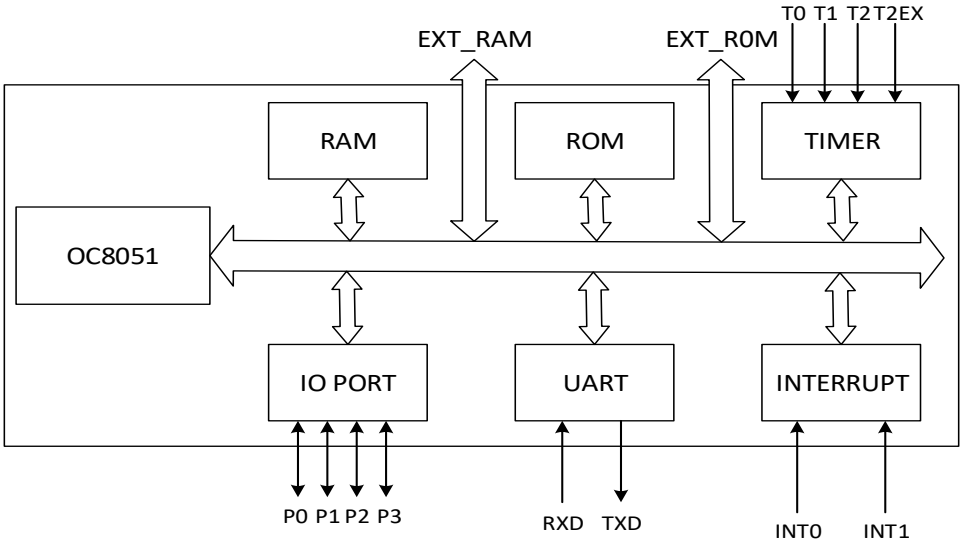
- 针对控制应用优化的 8 位单片机
- 广泛的布尔处理（单比特逻辑）能力
- 64KB 外部程序存储器寻址空间
- 64KB 外部数据储存器寻址空间
- 最高可达 64KB 的片内程序 ROM
- 最高可达 256B 的片内数据 RAM
- 两组连接外部存储器的总线
- 32 条双向且可单独寻址的 I/O 线
- 3 个 16 位定时器/计数器
- 一个 UART 接口

- 6 个中断源结构，2 个优先级

2.3 结构框图

OC8051 单片机结构框图如图 2-1 所示。

图 2-1 结构框图



3 信号描述

OC8051 单片机信号描述如表 3-1 所示。

表 3-1 信号描述

信号名称	I/O	宽度	描述	所属模块
wb_clk_i	in	1	系统时钟信号	-
wb_rst_i	in	1	系统复位信号	-
wbi_adr_o	out	16	外部 ROM 地址输出信号	外部 ROM 访问
wbi_dat_i	in	32	外部 ROM 数据输入信号	
wbi_stb_o	out	1	外部 ROM 字节选通输出信号	
wbi_ack_i	in	1	外部 ROM 响应输入信号	
wbi_cyc_o	out	1	外部 ROM 总线周期信号	
wbi_err_i	in	1	外部 ROM 传输错误信号	
ea_in	in	1	外部 ROM 访问使能信号	
wbd_dat_i	in	8	外部 RAM 数据输入信号	外部 RAM 访问
wbd_dat_o	out	8	外部 RAM 数据输出信号	
wbd_adr_o	out	16	外部 RAM 地址输出信号	
wbd_we_o	out	1	外部 RAM 数据读写使能信号	
wbd_ack_i	in	1	外部 RAM 响应输入信号	
wbd_stb_o	out	1	外部 RAM 字节选通输出信号	
wbd_cyc_o	out	1	外部 RAM 总线周期信号	
wbd_err_i	in	1	外部 RAM 传输错误信号	
int0_i	in	1	外部中断 0 信号	外部中断
int1_i	in	1	外部中断 1 信号	
p0_i	in	8	P0 端口输入信号	I/O PORT
p0_o	out	8	P0 端口输出信号	
p1_i	in	8	P1 端口输入信号	
p1_o	out	8	P1 端口输出信号	
p2_i	in	8	P2 端口输入信号	
p2_o	out	8	P2 端口输出信号	
p3_i	in	8	P3 端口输入信号	
p3_o	out	8	P3 端口输出信号	
rxd_i	in	1	UART 接收信号	UART
txd_o	out	1	UART 发送信号	
t0_i	in	1	定时器/计数器 0 输入信号	TIMER
t1_i	in	1	定时器/计数器 1 输入信号	
t2_i	in	1	定时器/计数器 2 输入信号	
t2ex_i	in	1	定时器/计数器 2 捕获/重装触发控制信号	
scanb_rst	in	1	BIST 复位输入信号	BIST
scanb_clk	in	1	BIST 时钟输入信号	
scanb_si	in	1	BIST 位输入信号	

scanb_so	out	1	BIST 位输出信号	
scanb_en	in	1	BIST 位使能信号	

4 硬件参数配置

OC8051 单片机硬件参数配置如表 4-1 所示。

位于…\oc8051\src\oc8051_defines.v。

表 4-1 硬件参数配置

参数选项	描述
INT_ROM_SIZE	片内程序存储器大小 65536：64KB 32768：32KB 16384：16KB 8192：8KB 4096：4KB 2018：2KB 1024：1KB 默认值：4096。
INT_RAM_SIZE	片内数据存储器大小 8：256B 7：128B 6：64B 5：32B 4：16B 默认值：8。
OC8051_CODES_FILE	软件程序代码文件。
OC8051_UART	开启 UART。
OC8051_TC01	开启定时器/计数器 0 和定时器/计数器 1。
OC8051_TC2	开启定时器/计数器 2。
OC8051_PORTS	开启 I/O PORTS。
OC8051_PORT0	开启 P0。
OC8051_PORT1	开启 P1。
OC8051_PORT2	开启 P2。
OC8051_PORT3	开启 P3。
OC8051_ROM	开启片内程序存储器。
OC8051_GENERIC	开启片内数据存储器。
OC8051_CACHE	开启 Cache，与外部存储器配合使用。
OC8051_WB	开启 Wishbone 接口。
OC8051_SIMULATION	仿真定义。
OC8051_BIST	内建自测试。

5 软件寄存器配置

OC8051 单片机软件寄存器、地址与描述如表 5-1 所示。

表 5-1 软件寄存器配置

寄存器	地址	描述
ACC	0xE0	累加寄存器。
B	0xF0	B 寄存器。执行乘、除法时，用于保存乘数或除数。
PSW	0xD0	程序状态字，用于保存 OC8051 单片机的工作状态。
IE	0xA8	中断使能寄存器。
IP	0xB8	中断优先级控制寄存器。
P0	0x80	P0 I/O 寄存器。
P1	0x90	P1 I/O 寄存器。
P2	0xA0	P2 I/O 寄存器。
P3	0xB0	P3 I/O 寄存器。
DPL	0x82	数据指针的低 8 位。
DPH	0x83	数据指针的高 8 位。
TMOD	0x89	定时器/计数器模式寄存器。
TCON	0x88	定时器/计数器控制寄存器。
TL0	0x8A	定时器/计数器 0 的低 8 位数据填充寄存器。
TH0	0x8C	定时器/计数器 0 的高 8 位数据填充寄存器。
TL1	0x8B	定时器/计数器 1 的低 8 位数据填充寄存器。
TH1	0x8D	定时器/计数器 1 的高 8 位数据填充寄存器。
SCON	0x98	串行通信控制寄存器。
SBUF	0x99	串行数据缓存寄存器。
PCON	0x87	电源控制寄存器。
SP	0x81	堆栈指针寄存器。
RCAP2L	0xCA	定时器/计数器 2 的低 8 位重装/捕获寄存器。
RCAP2H	0xCB	定时器/计数器 2 的高 8 位重装/捕获寄存器。
T2CON	0xC8	定时器/计数器 2 的控制寄存器。
TL2	0xCC	定时器/计数器 2 的低 8 位数据填充寄存器。
TH2	0xCD	定时器/计数器 2 的高 8 位数据填充寄存器。

5.1 ACC 寄存器

ACC 寄存器是 8051 单片机内部最常用的寄存器，也可写为 A，常用于保存算数或逻辑运算的操作数及运算结果。

5.2 B 寄存器

B 寄存器在乘法和除法运算中必须与 ACC 寄存器配合使用。

MUL AB 指令，把累加器 A 与寄存器 B 中的 8 位无符号数相乘，所得的 16 位乘积的低字节保存到 A 中，高字节保存到 B 中。

DIV AB 指令，使用 B 除以 A，整数商保存到 A 中，余数保存到 B 中。

B 寄存器还可以用作通用暂存寄存器。

5.3 PSW 寄存器

PSW 寄存器包含几个状态位，表示单片机的当前状态。

PSW 寄存器如表 5-2 所示。

表 5-2 PSW 寄存器

0xD7	0xD6	0xD5	0xD4	0xD3	0xD2	0xD1	0xD0
CY	AC	F0	RS1	RS0	OV		P

其中，

- CY：进/借位标志位。当执行加法运算时如果最高位有进位，或当执行减法运算时如果最高位有借位，则 CY 为 1，否则 CY 为 0。
- AC：进位辅助位。当执行加法运算时如果低半字节向高半字节有进位，或当执行减法运算时如果低半字节向高半字节有借位，则 AC 为 1，否则 AC 为 0。
- F0：用户标志位。
- RS1、RS0：工作寄存器组选择位，如表 5-3 所示。

表 5-3 RS1、RS0 位

RS1	RS0	当前使用的寄存器工作组	地址
0	0	0 组	00H-07H
0	1	1 组	08H-0FH
1	0	2 组	10H-17H
1	1	3 组	18H-1FH

- OV：溢出标志位。
- P：奇偶标志位。该标志位用来表示 ACC 寄存器中 1 的个数的奇偶性，如果

1 的个数为奇数，则 P 为 1；如果 1 的个数为偶数，则 P 为 0。

5.4 IE 寄存器

IE 寄存器用于控制每个中断源是否使能。

IE 寄存器如表 5-4 所示。

表 5-4 IE 寄存器

0xAF	0xAE	0xAD	0xAC	0xAB	0xAA	0xA9	0xA8
EA		ET2	ES	ET1	EX1	ET0	EX0

其中，

- EA：全局中断控制位。EA 为 1 时，使能所有中断；EA 为 0 时，禁止所有中断。
- ET2：定时器/计数器 2 溢出中断使能位。ET2 为 1 时，允许 T2 中断；ET2 为 0 时，禁止 T2 中断。
- ES：串口中断使能位。ES 为 1 时，允许串口中断；ES 为 0 时，禁止串口中断。
- ET1：定时器/计数器 1 溢出中断使能位。ET1 为 1 时，允许 T1 中断；ET1 为 0 时，禁止 T1 中断。
- EX1：外部中断 1 中断使能位。EX1 为 1 时，允许外部中断 1 中断；EX1 为 0 时，禁止外部中断 1 中断。
- ET0：定时器/计数器 0 溢出中断使能位。ET0 为 1 时，允许 T0 中断；ET0 为 0 时，禁止 T0 中断。
- EX0：外部中断 0 中断使能位。EX0 为 1 时，允许外部中断 0 中断；EX0 为 0 时，禁止外部中断 0 中断。

5.5 IP 寄存器

IP 寄存器用于控制中断的优先级。

IP 寄存器如表 5-5 所示。

表 5-5 IP 寄存器

0xBF	0xBE	0xBD	0xBC	0xBB	0xBA	0xB9	0xB8
		PT2	PS	PT1	PX1	PT0	PX0

其中，

- PT2：定时器/计数器 2 中断优先级。PT2 为 1 时，该中断有较高的优先级；PT2 为 0 时，该中断有较低的优先级。
- PS：串口中断优先级。PS 为 1 时，该中断有较高的优先级；PS 为 0 时，该中断有较低的优先级。
- PT1：定时器/计数器 1 中断优先级。PT1 为 1 时，该中断有较高的优先级；PT1 为 0 时，该中断有较低的优先级。
- PX1：外部中断 1 中断优先级。PX1 为 1 时，该中断有较高的优先级；PX1 为 0 时，该中断有较低的优先级。
- PT0：定时器/计数器 0 中断优先级。PT0 为 1 时，该中断有较高的优先级；PT0 为 0 时，该中断有较低的优先级。
- PX0：外部中断 0 中断优先级。PX0 为 1 时，该中断有较高的优先级；PX0 为 0 时，该中断有较低的优先级。

5.6 P0、P1、P2、P3 寄存器

P0、P1、P2、P3 是四个并行输入/输出端口的寄存器，分别包含 8 个引脚的输入输出，可以按位进行操作。

5.7 DPL、DPH 寄存器

数据指针寄存器是一个 16 位的专用寄存器，由低 8 位 DPL 寄存器和高 8 位 DPH 寄存器组成，可以以间接寻址或变址寻址的方式对外部数据 RAM 执行 64KB 范围内的数据操作。

5.8 TMOD 寄存器

TMOD 寄存器用于控制定时器/计数器 0 和定时器/计数器 1 的工作模式，不可位寻址。

TMOD 寄存器如表 5-6 所示。

表 5-6 TMOD 寄存器

7	6	5	4	3	2	1	0
GATE	C/T	M1	M0	GATE	C/T	M1	M0

其中，

- GATE：定时操作开关控制位。当 GATE=1 时，INT0 或 INT1 引脚为高电平，同时 TCON 中的 TR0 或 TR1 控制位为 1 时，定时器/计数器 0 或 1 才开始工作；当 GATE=0 时，只要将 TR0 或 TR1 控制位设置为 1，定时器/计数器 0 或 1 就开始工作。
- C/T：定时器/计数器功能的选择位。C/T=1 为计数器，通过外部引脚 T0 或 T1 输入计数脉冲；C/T=0 为定时器，由内部系统时钟提供计时工作脉冲。
- M0、M1：定时器/计数器工作模式选择位，如表 5-7 所示。

表 5-7 M0、M1 位

M0	M1	模式	描述
0	0	0	13 位定时器/计数器，TH0 或 TH1 全部使用，TL0 或 TL1 只使用低 5 位。
0	1	1	16 位定时器/计数器，TH0 或 TH1，TL0 或 TL1 全部使用。
1	0	2	8 位自动重载定时器，当溢出时，将 TH0 或 TH1 的值自动重载入 TL0 或 TL1。
1	1	3	定时器/计数器 1 停止工作。定时器 0 作为双 8 位定时器/计数器。TL0 作为一个 8 位定时器/计数器，通过标准定时器 0 的控制位控制。TH0 作为一个 8 位定时器，由定时器 1 的控制位控制。

5.9 TCON 寄存器

TCON 寄存器作为定时器/计数器 0 和定时器/计数器 1 的控制寄存器，同时也锁存 T0 和 T1 溢出中断源和外部请求中断源等。

TCON 寄存器如表 5-8 所示。

表 5-8 TCON 寄存器

0x8F	0x8E	0x8D	0x8C	0x8B	0x8A	0x89	0x88
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

其中，

- TF1：定时器/计数器 1 溢出标志位。当定时器/计数器 1 最高位产生溢出时，TF1 由硬件置 1，并向单片机内核请求中断，直到单片机内核响应中断，TF1 才由硬件清零；TF1 也可以由软件清零。
- TR1：定时器 1 的运行控制位。该位由软件置位与清零。当 GATE 为 0 时，TR1 置 1 允许定时器 1 开始计数，TR1 置 0 禁止定时器 1 计数；当 GATE 为 1，且外部中断 1 为高电平时，TR1 置 1 允许定时器 1 开始计数，TR1 置 0 禁止定时器 1 计数。
- TF0：定时器/计数器 0 溢出标志位。当定时器/计数器 0 最高位产生溢出时，TF0 由硬件置 1，并向单片机内核请求中断，直到单片机内核响应中断，TF0 才由硬件清零；TF0 也可以由软件清零。
- TR0：定时器 0 的运行控制位。该位由软件置位与清零。当 GATE 为 0 时，TR0 置 1 允许定时器 0 开始计数，TR0 置 0 禁止定时器 0 计数；当 GATE 为 1，且外部中断 0 为高电平时，TR0 置 1 允许定时器 0 开始计数，TR0 置 0 禁止定时器 0 计数。
- IE1：外部中断 1 请求标志位。当 IE1 为 1 时，外部中断向单片机内核请求中断，当单片机内核响应该中断时，IE1 由硬件清零。
- IT1：外部中断 1 触发方式控制位。当 IT1 为 0 时，外部中断 1 为低电平触发方式，当外部中断 1 输入低电平时，IE1 置 1。采用低电平触发方式，外部中断源必须保持低电平有效，直到该中断被单片机内核响应，同时，在该中断服务程序执行完成之前，外部中断源必须被清除，否则将产生另一次中断。当 IT1 为 1 时，外部中断 1 由下降沿触发中断，当外部中断 1 由高电平向低电平跳变时，IE1 置 1，向单片机内核发送中断请求。
- IE0：外部中断 0 请求标志位。当 IE0 为 1 时，外部中断向单片机内核请求中

断，当单片机内核响应该中断时，IE0 由硬件清零。

- IT0：外部中断 0 触发方式控制位。当 IT0 为 0 时，外部中断 0 为低电平触发方式，当外部中断 0 输入低电平时，IE0 置 1。采用低电平触发方式，外部中断源必须保持低电平有效，直到该中断被单片机内核响应，同时，在该中断服务程序执行完成之前，外部中断源必须被清除，否则将产生另一次中断。当 IT0 为 1 时，外部中断 0 由下降沿触发中断，当外部中断 0 由高电平向低电平跳变时，IE0 置 1，向单片机内核发送中断请求。

5.10 TL0、TH0 寄存器

TL0、TH0 寄存器用于保存定时器/计数器 0 的初始值，根据定时器/计数器的工作模式保存相应的数据。

5.11 TL1、TH1 寄存器

TL1、TH1 寄存器用于保存定时器/计数器 1 的初始值，根据定时器/计数器的工作模式保存相应的数据。

5.12 SCON 寄存器

SCON 寄存器用于选择串口通讯的工作方式和控制功能。

SCON 寄存器如表 5-9 所示。

表 5-9 SCON 寄存器

0x9F	0x9E	0x9D	0x9C	0x9B	0x9A	0x99	0x98
SM0	SM1	SM2	REN	TB8	RB8	TI	RI

其中，

- SM0、SM1：串口工作方式控制位。

SM0、SM1 控制串口工作方式如表 5-10 所示。

表 5-10 SM0、SM1 位

SM0	SM1	工 作 方式	功能说明	波特率
0	0	方式 0	同步移位串行方式：移位寄存器	SYSCLK/12

0	1	方式 1	8 位 UART，波特率可变	$(2^{SMOD}/32)\times$(定时器 1 的溢出率)
1	0	方式 2	9 位 UART	$(2^{SMOD}/64)\times$ SYSCLK
1	1	方式 3	9 位 UART，波特率可变	$(2^{SMOD}/32)\times$(定时器 1 的溢出率)
当单片机工作在 12T 模式时，定时器 1 的溢出率= SYSCLK/12/(256-TH1)。 当单片机工作在 6T 模式时，定时器 1 的溢出率= SYSCLK/6/(256-TH1)。				

- SM2：多机通讯控制位。在方式 0 中，SM2 必须置 0；在方式 1 中，当 SM2 为 1 且只有接收到有效停止位时，RI 才置 1；在方式 2 或方式 3 中，当 SM2 为 1 且接收到的第 9 位数据 RB8 为 0 时，RI 才置 1。
- REN：串口接收控制位。当 REN 为 1 时，允许串口接收数据；当 REN 为 0 时，禁止串口接收数据。
- TB8：发送数据位 8。在方式 2 和方式 3 中，TB8 是要发送的第 9 位数据位。在多机通信中它表示传输的是地址还是数据，TB8 为 0 时传输数据，TB8 为 1 时传输地址。
- RB8：接收数据位 8。在方式 2 和方式 3 中，RB8 保存接收到的第 9 位数据，用以识别接收到的数据特征。在方式 1 中，如果 SM2 为 0，则 RB8 接收到的是停止位，方式 0 不用 RB8。
- TI：串口发送中断标志位。串口以方式 0 发送数据时，每发送完 8 位数据，由硬件置 1。如果以方式 1、方式 2、方式 3 发送时，在发送停止位的开始时置 1。TI 为 1 表示串口正在向单片机内核申请中断。单片机内核在响应中断请求，转向中断服务程序时，并不会将 TI 清零，因此 TI 必须由用户在中断服务程序内由软件清零。
- RI：串口接收中断标志位。如果串口允许接收，且以方式 0 工作，则每当接收到第 8 位数据时置 1。如果以方式 1、方式 2、方式 3 工作，且 SM2 为 0 时，则每当接收到停止位的中间时置 1。当串口以方式 2 或方式 3 工作且 SM2 为 1 时，则仅当接收到第 9 位数据 RB8 为 1 后，同时还要接收到停止位的中间时置 1。RI 为 1 表示串口正在向单片机内核申请中断，RI 必须由用户在中断服务程序内由软件清零。

5.13 SBUF 寄存器

SBUF 寄存器用于缓存串口将要发送或接收的数据。

5.14 PCON 寄存器

PCON 寄存器用于电源控制，不可位寻址。

PCON 寄存器如表 5-11 所示。

表 5-11 PCON 寄存器

7	6	5	4	3	2	1	0
SMOD				GF1	GF0	PD	IDL

其中，

- SMOD：波特率选择位。当 SMOD 为 1 时，串行口波特率加倍。系统复位默认 SMOD 为 0。
- OC8051 单片机为软核，所以除了 SMOD 寄存器位外，其他位均为虚设。

5.15 SP 寄存器

SP 寄存器是一个 8 位的专用寄存器，表示堆栈顶部在片内数据存储器中的位置。

5.16 RCAP2L、RCAP2H 寄存器

RCAP2L、RCAP2H 寄存器用于保存定时器/计数器 2 重载/捕获的数值，根据定时器/计数器的工作模式保存相应的数据。

5.17 T2CON 寄存器

T2CON 寄存器用于控制定时器/计数器 2 的工作模式。

T2CON 寄存器如表 5-12 所示。

表 5-12 T2CON 寄存器

0xCF	0xCE	0xCD	0xCC	0xCB	0xCA	0xC9	0xC8
TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C_T2	CP_RL2

其中，

- TF2：定时器 2 溢出中断标志。TF2 必须由用户程序清零。当定时器 2 作为串口波特率发生器时，TF2 不会被置 1。
- EXF2：定时器 T2 外部中断标志。EXEN2 为 1 时，当 T2EX 发生负跳变时置 1 中断标志位 EXF2，EXF2 必须由软件清零。
- RCLK：串口的接收时钟选择标志位。RCLK=1 时，T2 工作于波特率发生器方式。
- TCLK：串口的发送时钟选择标志位。TCLK=1 时，T2 工作于波特率发生器方式。
- EXEN2：定时器 2 的外部使能标志。当 EXEN2 为 1 且定时器 2 未作为串口时钟时，允许 T2EX 的负跳变产生捕获或重载。
- TR2：定时器/计数器 2 控制位。TR2 为 1 时允许计数，TR2 为 0 时禁止计数。
- C/T2：外部定时器/计数器选择位。C/T2=1 时，T2 为外部事件计数器；C/T2=0 时，T2 为定时器，振荡脉冲的十二分频信号作为计数信号。
- CP/RL2：捕获和常数自动载入方式选择位。为 1 时 T2 工作于捕获方式，为 0 时 T2 工作于常数自动载入方式。当 TCLK 或 RCLK 为 1 时，CP/RL2 被忽略。

5.18 TL2、TH2 寄存器

TL2、TH2 寄存器用于保存定时器/计数器 2 初始值，根据定时器/计数器的工作模式保存相应的数据。

6 IDE 软件使用

6.1 软件版本

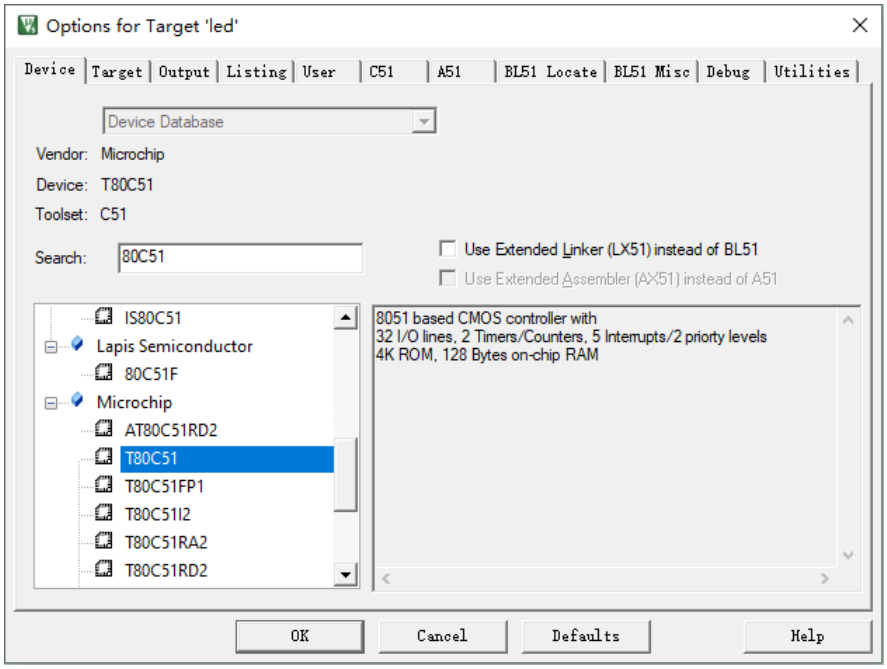
已测试软件版本：ARM Keil uVision5 C51 V9.24。

6.2 配置

6.2.1 器件选择

新建软件工程时需要选择器件型号，因 OC8051 为软核，仅需硬件设计中的寄存器定义与“reg51.h”的寄存器定义保持一致即可，所以可以在器件选择时任意选择一个 8051 器件即可，例如参考设计中选择的“Microchip > T80C51”，如图 6-1 所示。

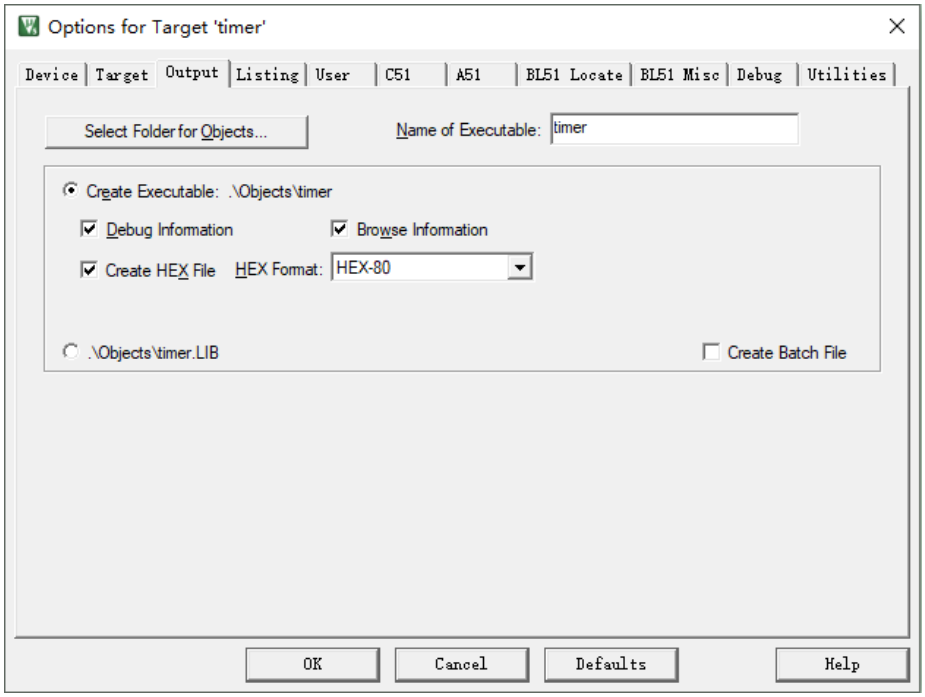
图 6-1 器件选择



6.2.2 文件输出

因“...\tool\oc8051_Rom_Maker.exe”软件工具输入.hex 文件，所以设置输出.hex 文件，请勾选“Options for Target… > Output > Create HEX File”，软件工程输出.hex 文件，如图 6-2 所示。

图 6-2 文件输出



6.3 编译

编译软件工程，产生.hex 文件。

7 参考设计

7.1 硬件设计

7.1.1 硬件工程

位于…\ref_design\FPGA_RefDesign\DK_START_GW5A25_V2.0\oc8051。

注意：

在编译此工程前，请务必先修改文件“oc8051_defines.v”中的参数“OC8051_CODES_FILE”为您本地的实际文件路径。

7.1.2 参数配置

表 7-1 参数配置

参数选项	描述	设置
INT_ROM_SIZE	片内程序存储器大小。	4096
INT_RAM_SIZE	片内数据存储器大小。	8
OC8051_CODES_FILE	软件程序代码文件。	“led.in”
OC8051_UART	开启 UART。	√
OC8051_TC01	开启定时器/计数器 0 和定时器/计数器 1。	√
OC8051_TC2	开启定时器/计数器 2。	√
OC8051_PORTS	开启 I/O PORTS。	√
OC8051_PORT0	开启 P0。	√
OC8051_PORT1	开启 P1。	√
OC8051_PORT2	开启 P2。	√
OC8051_PORT3	开启 P3。	√
OC8051_ROM	开启片内程序存储器。	√
OC8051_GENERIC	开启片内数据存储器。	√

7.1.3 测试环境

- Gowin_V1.9.10.02 (64-bit)
- DK_START_GW5A25 V2.0

  GW5A-LV25UG324C2/I1

  GW5A-25 A

7.2 软件设计

7.2.1 流水灯

7.2.1.1 软件工程

位于…\ref_design\MCU_RefDesign\led。

7.2.1.2 功能描述

将 OC8051 单片机的 P0^0、P0^1、P0^2、P0^3 输出端口约束到 4 个 LED 上，通过给 P0 寄存器赋值的方式实现 LED 流水闪烁。

7.2.1.3 软件代码

软件代码如下所示：

```
#include <reg51.h>

void main()
{
    unsigned int i;
    unsigned int k;
    while(1)
    {
        for(i=0;i<4;i++)
        {
            k=50000;
            P0=(0x01<<i);
            while(k--);
        }
    }
}
```

注意：

- 如果使用 for 循环实现延时，则单个 for 循环次数不能超过 32767。

7.2.2 按键

7.2.2.1 软件工程

位于…\ref_design\MCU_RefDesign\key。

7.2.2.2 功能描述

将 OC8051 单片机的 P1^0 输入端口约束到按键上，P0^0 输出端口约束到 LED 上，通过按键触发来实现 LED 的电平翻转。

7.2.2.3 软件代码

软件代码如下所示：

```
#include <reg51.h>

sbit KEY=P1^0;
sbit LED=P0^0;

void delay(unsigned int t);

void main()
{
    while(1)
    {
        if(KEY == 0)
        {
            delay(50000);          //延时消抖
            if(KEY == 0)
            {
                LED = ~LED;
                while(KEY == 0); //等待按键复原
            }
        }
    }
}

void delay(unsigned int t)
{
    while(t--);
}
```

注意：

- 在按键按下的过程中会产生抖动，这些抖动会产生多次跳变信号，因此需要利用延时函数消抖。

7.2.3 外部中断

7.2.3.1 软件工程

位于…\ref_design\MCU_RefDesign\ext_inter。

7.2.3.2 功能描述

将 OC8051 单片机的外部中断 1 的输入端口约束到按键上，P0^0 输出端口约束到 LED 上，通过按键触发中断来实现 LED 的电平翻转。

7.2.3.3 软件代码

软件代码如下所示：

```
#include <reg51.h>

sbit LED=P0^0;

void ext_init();

void ext_init()
{
    EA = 1;
    IT0 = 1;
    EX0 = 1;
}

void main()
{
    LED=0;
    ext_init();
    while(1)
    {
    }
}

void ext_inter() interrupt 0
{
    LED = !LED;
}
```

注意：

- 在开启外部中断 1 的中断使能前，需要开启全局的中断使能。
- 外部中断 1 和外部中断 2 的中断服务程序分别为 interrupt 0 和 interrupt 2。

7.2.4 定时器 0

7.2.4.1 软件工程

位于…\ref_design\MCU_RefDesign\timer。

7.2.4.2 功能描述

将 OC8051 单片机的 P0^0 输出端口约束到 LED 上，通过定时器溢出来实现 LED 的电平翻转。

7.2.4.3 软件代码

软件代码如下所示：

```
#include <reg51.h>

sbit LED=P0^0;

void Timer_Init();

void Timer_Init()
{
    TMOD = TMOD & 0xF0;
    TMOD = TMOD | 0x01;//启用定时器 0，工作模式 1
    TF0 = 0;                    //清除溢出标志
    TH0 = (65535-50000)/256;
    TL0 = (65535-50000)%256;
    TR0 = 1;                    //定时器开始启动
}

void main()
{
    unsigned int i;

    LED=0;

    Timer_Init();

    while(1)
```

```
    {
        i=0;
        LED = !LED;
        while(i<10)
        {
            while(TF0==0);
            TH0=(65536-50000)/256;
            TL0=(65536-50000)%256;
            TF0=0;
            i++;
        }
    }
}
```

注意：

- 在重载完初始值后，需要将 TR0 置 1，然后定时器开始运行。
- 定时器溢出后，需要软件消除溢出标志位 TF0，且在溢出后，如果想要继续使用定时器，需要重载初始值。

7.2.5 计数器 0

7.2.5.1 软件工程

位于…\ref_design\MCU_RefDesign\counter。

7.2.5.2 功能描述

将 OC8051 单片机的定时器/计数器 0 外部输入引脚约束到一根排针上，将 P0^4 输出端口约束到另一根排针上，并将两根排针用杜邦线连接，P0^0 输出端口约束到 LED 上，通过计数器的奇偶性来实现 LED 的电平翻转。

7.2.5.3 软件代码

软件代码如下所示：

```
#include <reg51.h>

sbit LED=P0^0;
sbit WIRE=P0^4;

void Timer_Init();
```

```
void Timer_Init()
{
	TMOD = TMOD & 0x00;
	TMOD = TMOD | 0x15;  //启用定时器 1，工作模式 1。计数器 0，工作模式 1
	TF1 = 0;
	TH1=(65536-50000)/256;
	TL1=(65536-50000)%256;
	TR1 = 1;
	TF0 = 0;
	TH0 = 0;
	TL0 = 0;
	TR0 = 1;
}

void main()
{
	unsigned int i;

	LED=0;
	WIRE=0;

	Timer_Init();

	while(1)
	{
		i=0;
		WIRE = !WIRE;
		while(i<10)
		{
			while(TF1==0);
			TH1=(65536-50000)/256;
			TL1=(65536-50000)%256;
			TF1=0;
			i++;
		}
		if(TL0%2==1)
		{
			LED = !LED;
		}
	}
}
```

注意：

- 定时器/计数器 1 使用定时器模式，使 P0^4 引脚产生电平翻转，定时器/计数器 0 使用计数器模式，接收 P0^4 引脚产生的脉冲。

7.2.6 定时器 0 中断

7.2.6.1 软件工程

位于…\ref_design\MCU_RefDesign\timer_inter。

7.2.6.2 功能描述

将 OC8051 单片机的 P0^0 输出端口约束到 LED 上，通过触发定时器中断来实现 LED 的电平翻转。

7.2.6.3 软件代码

软件代码如下所示：

```
#include <reg51.h>

sbit LED=P0^0;

void Timer_Init();

void Timer_Init()
{
    TMOD = TMOD & 0xF0;
    TMOD = TMOD | 0x01;
    TF0 = 0;
    EA = 1;                 //开启总中断
    ET0 = 1;                //开启定时器 0 中断
    TH0 = (65535-50000)/256;
    TL0 = (65535-50000)%256;
    TR0 = 1;                //定时器 0 开始运行
    PT0 = 0;                //定时器 0 中断优先级较低
}

void main()
{
    Timer_Init();

    while(1)
    {
    }
```

```
}

void Timer0_Inter() interrupt 1
{
    static unsigned int T0Count;
    TF0 = 0;
    TH0 = (65535-50000)/256;
    TL0 = (65535-50000)%256;
    T0Count++;
    if(T0Count >= 20)
    {
        T0Count = 0;
        LED = !LED;
    }
}
```

注意：

- 在开启定时器 0 的中断使能前，需要开启全局的中断使能。
- 定时器 0 和定时器 1 的中断服务程序分别为 interrupt 1 和 interrupt 3。

7.2.7 定时器 2

7.2.7.1 软件工程

位于…\ref_design\MCU_RefDesign\timer2。

7.2.7.2 功能描述

将 OC8051 单片机的 P0^0 输出端口约束到 LED 上，通过定时器溢出来实现 LED 的电平翻转。

7.2.7.3 软件代码

软件代码如下所示：

```
#include <reg51.h>

sbit LED=P0^0;

sfr RCAP2H=0xCB;
sfr RCAP2L=0xCA;

sfr T2CON=0xC8;
```

```
sbit TF2=T2CON^7;
sbit TR2=T2CON^2;

sfr TH2=0xCD;
sfr TL2=0xCC;

sbit ET2=IE^5;

void Timer2_Init();

void Timer2_Init()
{
    TH2=(65536-50000)/256;
    TL2=(65536-50000)%256;
    TF2=0;
    TR2=1;
}

void main()
{
    unsigned int i;

    LED=0;

    Timer2_Init();

    while(1)
    {
        i=0;
        LED = !LED;
        while(i<10)
        {
            while(TF2==0);
            TH2=(65536-50000)/256;
            TL2=(65536-50000)%256;
            TF2=0;
            i++;
        }
    }
}
```

注意：

- 在 reg51.h 头文件中没有定时器/计数器 2 相关的寄存器，所以需要在软件工

程中定义相关寄存器。

7.2.8 定时器 2 中断

7.2.8.1 软件工程

位于…\ref_design\MCU_RefDesign\timer2_inter。

7.2.8.2 功能描述

将 OC8051 单片机的 P0^0 输出端口约束到 LED 上，通过触发定时器中断来实现 LED 的电平翻转。

7.2.8.3 软件代码

软件代码如下所示：

```
#include <reg51.h>

sbit LED=P0^0;

sfr RCAP2H=0xCB;
sfr RCAP2L=0xCA;

sfr T2CON=0xC8;     //定时器/计数器 2 控制寄存器
sbit TF2=T2CON^7;   //定时器/计数器 2 溢出标志位
sbit TR2=T2CON^2;   //定时器/计数器 2 运行标志位

sfr TH2=0xCD;         //定时器/计数器 2 数据寄存器高 8 位
sfr TL2=0xCC;         //定时器/计数器 2 数据寄存器低 8 位

sbit ET2=IE^5;        //定时器/计数器 2 中断标志位

void Timer2_Init();
void timer2_inter();

void Timer2_Init()
{
    TH2=(65536-50000)/256;
    TL2=(65536-50000)%256;
    TF2=0;
    TR2=1;
    ET2=1;
    EA=1;
```

```
}

void main()
{
      LED=0;

      Timer2_Init();

   while(1)
      {
      }
}

void timer2_inter() interrupt 5 using 1
{
      static unsigned int T0Count;
      TF2=0;
      TH2 = (65535-50000)/256;
      TL2 = (65535-50000)%256;
      T0Count++;
      if(T0Count >= 10)
      {
            T0Count = 0;
            LED = !LED;
      }
}
```

注意：

- 在开启定时器 2 的中断使能前，需要开启全局的中断使能。
- 定时器 2 的中断服务程序为 interrupt 5。
- 可以通过 using n 来使用不同的寄存器组。

7.2.9 串口通讯

7.2.9.1 软件工程

位于…\ref_design\MCU_RefDesign\uart。

7.2.9.2 功能描述

将 OC8051 单片机的 txd 和 rxd 端口约束到排针上，通过上位机串口调试助

手软件来接收由 OC8051 单片机的串口发出的字符串。

7.2.9.3 软件代码

软件代码如下所示：

```
#include <reg51.h>

void SendString (unsigned char *str);
void SendBit (unsigned char UART_data);

void UART_Init (void)
{
    EA = 1; //允许总中断（如不使用中断，可用//屏蔽）
    ES = 0; //允许 UART 串口的中断
    TMOD = 0x20; //定时器 T/C1 工作方式 2
    SCON = 0x50; //串口工作方式 1，允许串口接收（SCON = 0x40 时禁止串口接收）
    TH1 = 0xF3; //定时器初值高 8 位设置
    TL1 = 0xF3; //定时器初值低 8 位设置
    PCON = 0x80; //波特率倍频（屏蔽本句波特率为 2400）
    TR1 = 1; //定时器启动
}

void SendString (unsigned char *str)
{
    while(*str != '\0')
    {
        SendBit(*str);
        str++;
    }
    *str = 0;
}

void SendBit (unsigned char UART_data)
{
    SBUF = UART_data; //将要发送的数据放到串口缓冲寄存器
    while(TI == 0); //检查发送中断标志位
    TI = 0; //令发送中断标志位为 0（软件清零）
}



void main()
{
    unsigned char *s = "hello world\r\n";
    unsigned int i=0;
```

```
    UART_Init();

    while(1)
    {
        i = 50000;
        SendString(s);
        while(i--);
    }
}
```

注意：

- 波特率的计算方法可以参考第 5.12 节，为了保证传输的可靠性，波特率的误差被要求不大于 2.5%。
- 如果开启了串口中断，再进行字符串传输，字符串的第一个字节在第一次传输时会传输两次。

7.2.10 串口中断

7.2.10.1 软件工程

位于…\ref_design\MCU_RefDesign\uart_inter。

7.2.10.2 功能描述

将 OC8051 单片机的 txd 和 rxd 端口约束到排针上，通过上位机串口调试助手软件向 OC8051 单片机的串口发送字符串，并接收由 OC8051 单片机的串口回环转发回来的字符串。

7.2.10.3 软件代码

软件代码如下所示：

```
#include <reg51.h>

void UART_Init (void);



void UART_Init (void)
{
    EA = 1; //允许总中断（如不使用中断，可用//屏蔽）
```

```
    ES = 1; //允许 UART 串口的中断
    TMOD = 0x20; //定时器 T/C1 工作方式 2
    SCON = 0x50; //串口工作方式 1，允许串口接收（SCON = 0x40 时禁止串口接收）
    TH1 = 0xF3; //定时器初值高 8 位设置
    TL1 = 0xF3; //定时器初值低 8 位设置
    PCON = 0x80; //波特率倍频（屏蔽本句波特率为 2400）
    TR1 = 1; //定时器启动
}

void main()
{
    UART_Init();

    while(1)
    {
    }
}

void UART_Inter (void) interrupt 4
{
    unsigned char UART_data; //定义串口接收数据变量
    if(RI != 0)
    {
        RI = 0; //令接收中断标志位为 0（软件清零）
        UART_data = SBUF; //将接收到的数据送入变量 UART_data
        SBUF = UART_data;
    }
    else if(TI != 0)
    {
        TI = 0;
    }
}
```

注意：

- 串口的接收与发送中断都在一个中断程序中。
- 串口的中断服务程序为 interrupt 4。

8 开发流程

Gowin OC8051 单片机开发流程如下：

1. 获取以及安装高云®云源软件 Gowin_Vx.x；
2. 获取以及安装 ARM Keil C51 Vx.x 软件；
3. 获取 Gowin OC8051 的软件开发工具包；
4. ARM Keil C51 软件新建或打开已有软件工程（例如，…\ref_design\MCU_RefDesign\led），配置、编译，产生.hex 文件；
5. oc8051_Rom_Maker.exe 软件工具（…\tool\oc8051_Rom_Maker.exe）输入.hex 文件，产生.in 文件；
6. 云源软件新建或打开已有硬件工程（例如，…\ref_design\FPGA_RefDesign\DK_START_GW5A25_V2.0\oc8051），设置“oc8051_defines.v”文件的“OC8051_CODES_FILE”参数，第 5 步产生的.in 文件作为此参数值，即片内程序存储器的初始值；然后，配置、综合、布局、布线，产生比特流文件；
7. 如果有串口打印输出，请打开上位机串口调试助手软件工具，设置波特率为 4800；
8. Programmer 软件工具下载比特流文件。