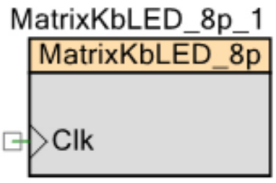
矩阵键盘/LED（8 口，MatrixKbLED_8p）

1.0

特性

- 使用 8 个 I/O 口
- 支持直至 24 个按键（含旋转编码开关）、23 个 LED
- 直接连接 CY8CKIT-050、CY8CKIT-062 开发板
- 识别任意数量按键的同时动作
- 按键动作消抖、旋转编码开关旋转动作消抖
- 实现任意组合形式的 LED 显示
- 可选生成按键消息
- 可选使用用户定制的按键符号
- 使用按键状态映像便于支持扩展的按键动作



概述

MatrixKbLED_8p 组件设计用于 CY8CKIT-050、CY8CKIT-062 开发板直接连接使用矩阵键盘/LED 模块，它使用 8 个 I/O 口实现对直至 24 个按键和 23 个 LED 的访问。组件提供各按键状态映像（含旋转编码开关旋转动作状态）。能识别任意数量按键的同时动作，并可选生成按键动作消息，包含按键按下、松开，旋转编码开关顺时针（CW）旋转和逆时针（CCW）旋转动作。应用程序可使用 API 函数修改 LED 显示缓冲区内容，实现任意组合形式的显示。

何时使用 MatrixKbLED_8p

希望使用 8 个 I/O 口，连接矩阵键盘/LED 模块，扩展出较多数量的按键、LED，以及使用旋转编码开关。

输入/输出连接

表中标*的I/O不显示，而是通过固件进行控制。

I/O 名称	类型	描述
Clk	输入	时钟输入定义此组件的扫描周期，应连接符合频率要求的时钟，**对于 PSoC 5LP，频率不应过高，宜等于或略大于下限值**。 • 对于 PSoC 6，作为组件内部所包含的 TCPWM 定时器的输入时钟，定时器的定时周期是为组件的扫描周期。当 SCANS_PER_ROUND 参数配置为 10 时，Clk 频率应>=500Hz；当 SCANS_PER_ROUND 参数配置为 6 时，Clk 频率应>=303Hz。 • 对于 PSoC 5LP，组件内部未使用定时器，组件的扫描周期直接等于 Clk 周期。当 SCANS_PER_ROUND 参数配置为 10 时，Clk 频率应>=500Hz；当 SCANS_PER_ROUND 参数配置为 6 时，Clk 频率应>=303Hz
EN *	输入/输出	矩阵键盘/LED 模块内的 74138 译码器使能信号 PSoC 5LP 时使用 P4.3，PSoC 6 时使用 P1.5。
RA0 *	输入/输出	矩阵键盘/LED 模块的行地址 0 PSoC 5LP 时使用 P4.0，PSoC 6 时使用 P12.6。
RA1 *	输入/输出	矩阵键盘/LED 模块的行地址 1 PSoC 5LP 时使用 P4.1，PSoC 6 时使用 P12.7。

I/O 名称	类型	描述
RA2 *	输入/输出	矩阵键盘/LED 模块的行地址 2 PSoC 5LP 时使用 P4.2，PSoC 6 时使用 P12.4。
COL0 *	输入/输出	矩阵键盘/LED 模块的列线 0 PSoC 5LP 时使用 P12.0，PSoC 6 时使用 P1.2。
COL1 *	输入/输出	矩阵键盘/LED 模块的列线 1 PSoC 5LP 时使用 P12.1，PSoC 6 时使用 P1.3。
COL2 *	输入/输出	矩阵键盘/LED 模块的列线 2 PSoC 5LP 时使用 P12.2，PSoC 6 时使用 P1.4。
COL3 *	输入/输出	矩阵键盘/LED 模块的列线 3 PSoC 5LP 时使用 P12.3，PSoC 6 时使用 P12.5。

组件参数

将一个 MatrixKbLED_8p 组件拖放到您的设计上，并双击以打开配置对话框。

CLOCK_FREQ_HZ

组件的输入时钟频率。PSoC 5LP 时不使用此参数。

PORTING_TO_CY8CKIT

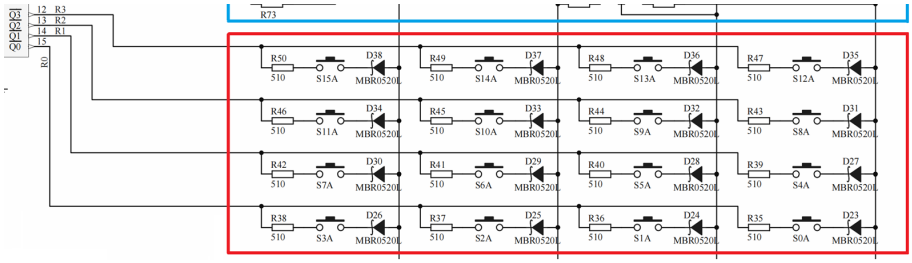
选择矩阵键盘/LED 模块所连接的开发板

MESSAGE_GEN

选择是否生成按键消息，按键消息包括按键按下、松开，旋转编码开关顺时针(CW)旋转和逆时针(CCW)旋转动作。不生成按键消息时，组件仅提供各按键/开关的当前开合状态、旋转动作状态。

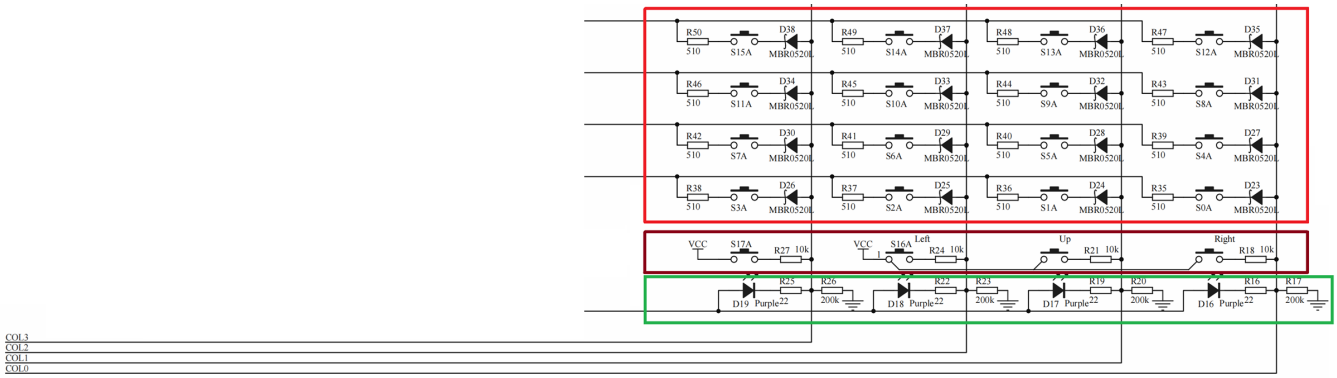
NUM_KEYROWS

设置按键的总行数，此值限于 R3~R0 行按键区域，取值 0~4。无效的按键软件将会予以屏蔽。



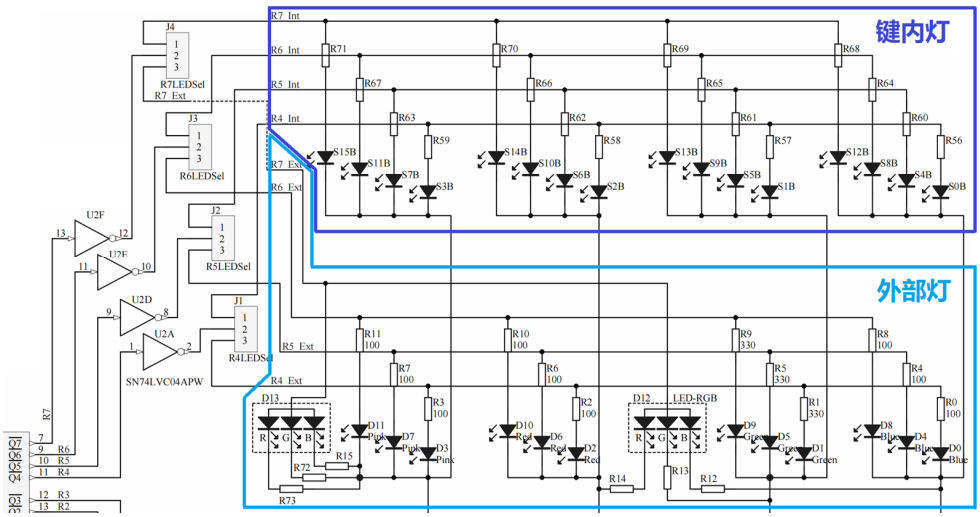
NUM_KEYCOLS

设置按键的总列数，此值限于 COL3~COL0 列按键区域（红、棕色区域），取值 0~4。无效的按键软件将会予以屏蔽，空余的列线仍会占用 I/O 引脚。



NUM_LEDROWS

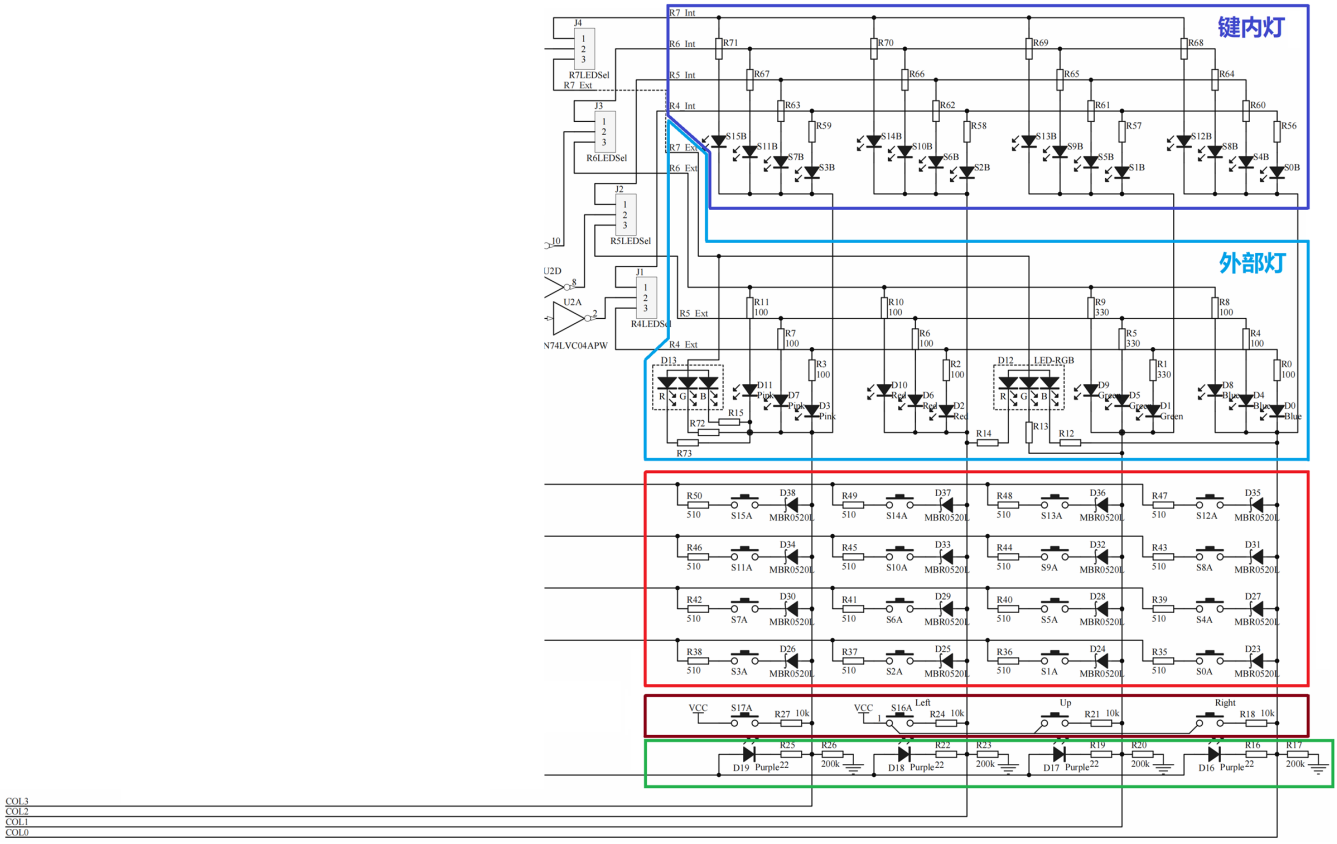
设置 LED 的总行数，此值限于 R7~R4 行 LED 区域，取值 0~4。无效的 LED 软件将会予以屏蔽。



注：R7~R4 行 LED 的每一行，只能选择使用键内灯或外部灯之一，由 J4～J1 分别选择 R7～R4 行 LED 的连接。

NUM_LEDCOLS

设置 LED 的总列数，此值限于 COL3~COL0 列 LED 区域（蓝、青、绿色区域），取值 0~4。无效的 LED 软件将会予以屏蔽，空余的列线仍会占用 I/O 引脚。



CUSTOM_KEYSYM_TABLE

选择是否使用用户自定义的各按键符号。勾选上时，用户必须在项目中的某个.c 文件中自定义各按键符号，且必须按照以下全局数组名称、大小、及格式来定义。

按键符号应各不相同且不能使用'o'（该符号专用于旋转编码开关的旋转动作）。表中的左下方 4x4 区域（标为 x 处）未使用，可随意定义。

按键行列号至按键符号查找表（第 4 行为五向旋转编码开关各键 +S17）

	[][7]	[][6]	[][5]	[][4]	[][3]	[][2]	[][1]	[][0]	：列号
[4][]	S16_Down	S16_Cntr	S16_CHB	S16_CHA	S17	S16_Left	S16_Up	S16_Right	
[3][]	x	x	x	x	S15	S14	S13	S12	*(NUM_KEYROWS-1)*
[2][]	x	x	x	x	S11	S10	S9	S8	.
[1][]	x	x	x	x	S7	S6	S5	S4	.
[0][]	x	x	x	x	S3	S2	S1	S0	*0*
.. 行号	*(NUM_KEYCOLS-1) .. 0*								

组件内置的按键符号表定义为：

```
const char8 keysym_table[5][8]={{'0', '1', '2', '3', 'x', 'x', 'x', 'x'},
                                {'4', '5', '6', '7', 'x', 'x', 'x', 'x'},
                                {'8', '9', 'A', 'B', 'x', 'x', 'x', 'x'},
                                {'C', 'D', 'E', 'F', 'x', 'x', 'x', 'x'},
                                {'>', '^', '<', 'S', '-', '=', '+', 'v'}};
```

SCANS_PER_ROUND

每轮扫描所用的扫描次数，取值应为 10（数值对应 NUM_KEYROWS=NUM_LEDROWS=4 时）、或 6（数值对应 NUM_LEDROWS=4 时）。行数设置与所述不同时，该值无实际意义，仅作两种不同扫描方法的标示。

此值越小，则单次扫描的最坏执行时间越长。

注：每轮扫描 6 次时，COL 各列线会短暂输出强驱动高电平，参见“矩阵键盘/LED 模块电路原理”。

应用程序接口

通过应用程序接口（API），您可以使用本组件。此表列出了每个函数的接口，并进行了说明。以下各节将更详细地介绍每个函数。

默认情况下，PSoC Creator 将实例名称“MatrixKbLED_8p_1”分配给设计中的第一个组件实例。您可以将其重新命名为遵循标识符语法规则的任何唯一值。实例名称会成为每个全局函数名称、变量和常量符号的前缀。为增加可读性，下表中使用了实例名称“MatrixKbLED”。

函数	说明
MatrixKbLED_Start()	矩阵键盘/LED 组件初始化
MatrixKbLED_SetLED()	点亮或者熄灭若干个 LED
MatrixKbLED_SetLED_RC()	点亮或者熄灭指定行列处的一个 LED
MatrixKbLED_SetBarLEDLevel()	设置 LED 灯条的指示级
MatrixKbLED_GetLED()	获取某个 LED 的当前状态
MatrixKbLED_GetLED_RC()	获取指定行列处的一个 LED 的当前状态

函数	说明
MatrixKbLED_KeySym2RC()	获取按键符号所在的行、列号
MatrixKbLED_GetKey()	获取某个按键、或全部按键以及旋转动作的当前状态
MatrixKbLED_GetKey_RC()	获取指定行列处的一个按键的当前状态
MatrixKbLED_SimKeyAction()	模拟按键动作时发送一条消息
Qkey_FetchData()	从队首获取一条消息
Qkey_FeedData()	向队尾添加一条消息

void MatrixKbLED_Start(void)

说明： 矩阵键盘/LED 组件初始化
8 个 GPIO 口初始化、（定时器及其）中断初始化、按键消息循环队列等其它初始化。

参数： 无

返回值： 无

其他影响： 8 个 GPIO 口的先前配置将会被覆盖

void MatrixKbLED_SetLED(uint32 leds, uint8 onoff)

说明： 点亮或者熄灭若干个 LED

参数： leds：指定要点亮或熄灭的 LED，使用以下宏之一、或者多个宏的或
LED_D0_S0B、LED_D1_S1B、LED_D2_S2B、LED_D3_S3B、LED_D4_S4B、LED_D5_S5B、LED_D6_S6B、LED_D7_S7B、LED_D8_S8B、LED_D9_S9B、LED_D10_S10B、LED_D11_S11B、LED_D12B_S12B、LED_D12G_S13B、LED_D12R_S14B、LED_D13_S15B、LED_D16、LED_D17、LED_D18、LED_D19、LED_D20、LED_D21、LED_D22

使用宏 LED_ALL 时，操作以上所有 LED。

onoff：使用宏 LEDON——点亮、宏 LEDOFF——熄灭

返回值： 无

其他影响： 无

void MatrixKbLED_SetLED_RC(uint8 row, uint8 col, uint8 onoff)

说明： 点亮或者熄灭指定行列处的一个 LED

参数： row、col：LED 所在的行、列号，如下表所示排布

	7	6	5	4	3	2	1	0	：列号
4	x	D22	D21	D20	D19	D18	D17	D16	
3	x	x	x	x	D13_S15B	D12R_S14B	D12G_S13B	D12B_S12B	*(NUM_LEDROWS-1)*
2	x	x	x	x	D11_S11B	D10_S10B	D9_S9B	D8_S8B	.
1	x	x	x	x	D7_S7B	D6_S6B	D5_S5B	D4_S4B	.
0	x	x	x	x	D3_S3B	D2_S2B	D1_S1B	D0_S0B	*0*
.. 行号	*(NUM_LEDCOLS-1) .. 0*								

onoff：使用宏 LEDON——点亮、宏 LEDOFF——熄灭

返回值： 无

其他影响： 无

void MatrixKbLED_SetBarLEDLevel(uint8 level)

说明： 设置 LED 灯条的指示级

参数： level：LED 灯条的指示级，取值范围“0(全灭) ~ LVLNUM_BARLED-1(全亮)”

返回值： 无

其他影响： 无

uint8 MatrixKbLED_GetLED(uint32 led)

说明： 获取某个 LED 的当前状态

参数： led：指定要获取的 LED，使用以下宏之一
LED_D0_S0B、LED_D1_S1B、LED_D2_S2B、LED_D3_S3B、LED_D4_S4B、LED_D5_S5B、
LED_D6_S6B、LED_D7_S7B、LED_D8_S8B、LED_D9_S9B、LED_D10_S10B、
LED_D11_S11B、LED_D12B_S12B、LED_D12G_S13B、LED_D12R_S14B、LED_D13_S15B、
LED_D16、LED_D17、LED_D18、LED_D19、LED_D20、LED_D21、LED_D22

返回值： LEDON ——亮
LEDOFF——灭
0xFF ——参数错误

其他影响： 无

uint8 MatrixKbLED_GetLED_RC(uint8 row, uint8 col)

说明： 获取指定行列处的一个 LED 的当前状态

参数： row、col：LED 所在的行、列号，如下表所示排布

	7	6	5	4	3	2	1	0	：列号
4	x	D22	D21	D20	D19	D18	D17	D16	
3	x	x	x	x	D13_S15B	D12R_S14B	D12G_S13B	D12B_S12B	*(NUM_LEDROWS-1)*
2	x	x	x	x	D11_S11B	D10_S10B	D9_S9B	D8_S8B	.
1	x	x	x	x	D7_S7B	D6_S6B	D5_S5B	D4_S4B	.
0	x	x	x	x	D3_S3B	D2_S2B	D1_S1B	D0_S0B	*0*
.. 行号	*(NUM_LEDCOLS-1) .. 0*								

返回值： LEDON ——亮
LEDOFF——灭
0xFF ——参数错误

其他影响： 无

uint8 MatrixKbLED_KeySym2RC(uint8 keysym)

说明： 获取按键符号所在的行、列号

参数： keysym：按键符号（在按键行列号至按键符号查找表 keysym_table[5][8]中的符号）

返回值： 低 4 位——列号
高 4 位——行号
0xFF ——参数错误

其他影响： 无

uint32 MatrixKbLED_GetKey(uint32 key)

说明： 获取某个按键、或全部按键以及旋转动作的当前状态

参数： key：指定要获取的按键，使用以下宏之一
S0、S1、S2、S3、S4、S5、S6、S7、S8、S9、S10、S11、S12、S13、S14、S15、
S16_RIGHT、S16_UP、S16_LEFT、S17、S16_CHA、S16_CHB、S16_CENTER、S16_DOWN

使用宏 SW_ALL 时，获取以上所有按键的当前状态、以及旋转动作状态。

返回值： 单个按键时：KEYON——闭合（按下）
KEYOFF——断开（松开）
0xFFFFFFFF——参数错误
全部按键及旋转动作时：32 位数据表示所有各按键、以及旋转动作状态，如下表排布：

位:	31	30	29	28	27..24	23	22	21	20	19	18	17	16	15..0
	0	0	CCW	CW	0 0 0 0	Down	Ctr	CHB	CHA	S17	Left	Up	Right	S15..S0

注：逆时针 CCW、顺时针 CW 旋转位为 Sticky 位，置位后将保持，读后同时清零。

其他影响： 获取全部按键及旋转动作的当前状态时，若 CCW/CW 位为 1（有旋转），则读后将被清零。

uint8 MatrixKbLED_GetKey_RC(uint8 row, uint8 col)

说明： 获取指定行列处的一个按键的当前状态

参数： row、col：按键所在的行、列号，如下表所示排布

	7	6	5	4	3	2	1	0	：列号
4	S16_Down	S16_Cntr	S16_CHB	S16_CHA	S17	S16_Left	S16_Up	S16_Right	
3	x	x	x	x	S15	S14	S13	S12	*(NUM_KEYROWS-1)*
2	x	x	x	x	S11	S10	S9	S8	.
1	x	x	x	x	S7	S6	S5	S4	.
0	x	x	x	x	S3	S2	S1	S0	*0*
.. 行号	*(NUM_KEYCOLS-1) .. 0*								

返回值： KEYON——闭合（按下）
KEYOFF——断开（松开）
0xFFFFFFFF——参数错误

其他影响： 无

void MatrixKbLED_SimKeyAction(uint8 keysym, uint8 onoff)

说明： 模拟按键动作时发送一条消息

参数： keysym：按键符号
onoff：使用宏 KEYON——按下、宏 KEYOFF——松开

返回值： 无

其他影响： 组件的时钟或定时器中断在此期间被禁止

注：当组件参数 MESSAGE_GEN 被配置为不勾选（=0）时，此函数无效。

uint8 Qkey_FetchData(void)

说明： 从队首获取一条消息

参数： 无

返回值： 0——未获取到消息
其它——获取到的消息

其他影响： 组件的时钟或定时器中断在此期间被禁止

消息格式：

位域：	b7 类型标志	b6..b0 按键符号
其余各按键	0——按下	keysym_table[][]中的值
	1——松开	
旋转编码开关专用	0——顺时针（CW）旋转	'o'
	1——逆时针（CCW）旋转	

注：当组件参数 MESSAGE_GEN 被配置为不勾选（=0）时，此函数无效。

uint8 Qkey_FeedData(uint8 bytemsg)

说明： 向队尾添加一条消息

参数： bytemsg：消息数据

返回值： 1：入队列成功
0：入队列失败（因队列已满）

其他影响： 组件的时钟或定时器中断在此期间被禁止

注：当组件参数 MESSAGE_GEN 被配置为不勾选（=0）时，此函数无效。

示例固件源代码

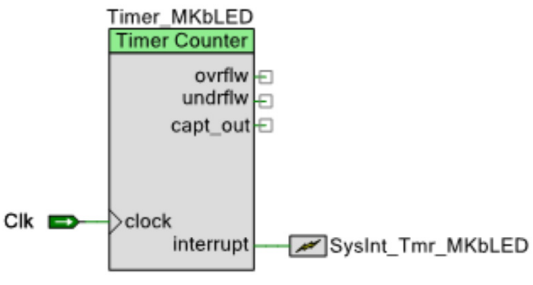
在 Find Code Example 对话框中，PSoC Creator 提供了大量的代码示例，包括原理图和示例代码。要获取组件特定的示例，请通过组件目录或原理图中的组件实例打开该对话框。要查看通用示例，请从 Start Page 或 **File** 菜单中打开该对话框。根据需求，可以通过使用对话框中的 **Filter Options** 选项来限定可选的项目列表。

更多有关信息，请参考《PSoC Creator 帮助》中的主题“查找代码示例”。

功能说明

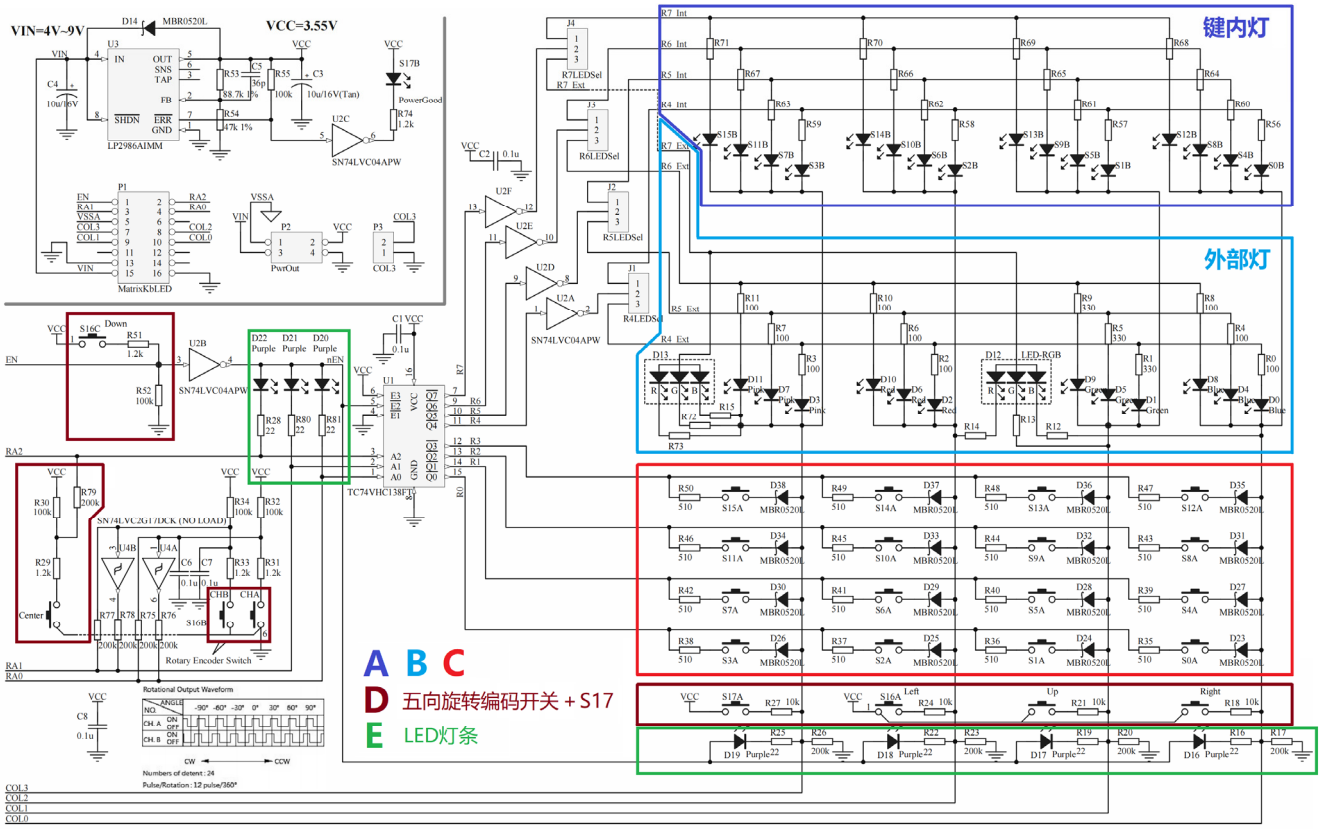
组件内部电路

PSoC 5LP 时



PSoC 6 时

矩阵键盘/LED 模块电路原理



模块仅设计用于与 3.3V~5.0V CMOS I/O 口互连。为此设置了电源 VCC=3.55V，使得送入 I/O 输入口的高电平电压不超过“3.3V+引脚钳位二极管压降”，同时也能符合 5.0V CMOS 高电平输入电压阈值要求。74138 译码器和反相器均使用了 5V 输入容忍的型号，能直接连接 5V I/O 输出口。此外，VCC=3.55V 也可保证紫色 LED 正常点亮。

C 按键矩阵中的各二极管均串联了限流电阻，以预防编程错误所可能导致的过大电流。由于限流电阻会附加压降，因此采用肖特基二极管而不应使用 1N4148。

A、**B** 块的 R7～R4 各行 LED，每行均只能选择使用键内灯或外部灯之一，两者不能同时使用。J4～J1 用于手动选择使用按键内部 LED 灯（跳块接通 1、2 引脚时），或是外部 LED 灯（跳块接通 2、3 引脚时），J4～J1 分别对应 R7～R4 行 LED。

74138 译码器的 R7～R4 输出连接了反相器，这样点亮 LED 时列线不需提供强驱动高电平，可减少列线 I/O 口所需的驱动模式，以及编程错误的影响。

E 块的各 LED 需使用高正向压降的紫色灯色（或其它正向压降接近的灯色）。选择合适的下拉电阻，使其流过微弱的正向电流，实现视觉效果为灭灯，同时列线呈现低电平以配合 **D** 块中的按键松开时的电平状态。

D 块中的开关 CHB、CHA 为旋转编码开关，非独立按键。

模块与开发板的端口连接：

	I/O 引脚所需的驱动模式	CY8CKIT-050 开发板	CY8CKIT-062 开发板
EN	强驱动、高阻数字输入	P4.3	P1.5 (且附加 2k 下拉电阻)
RA0	强驱动、高阻数字输入	P4.0	P12.6
RA1	强驱动、高阻数字输入	P4.1	P12.7
RA2	强驱动、高阻数字输入	P4.2	P12.4

	I/O 引脚所需的驱动模式	CY8CKIT-050 开发板	CY8CKIT-062 开发板
COL0	开漏低驱动无上拉 开漏低驱动有上拉 强驱动（可选）	P12.0	P1.2
COL1	开漏低驱动无上拉 开漏低驱动有上拉 强驱动（可选）	P12.1	P1.3
COL2	开漏低驱动无上拉 开漏低驱动有上拉 强驱动（可选）	P12.2	P1.4
COL3	开漏低驱动无上拉 开漏低驱动有上拉 强驱动（可选）	P12.3	P12.5

注：※ 为使用本模块，对于 CY8CKIT-062-WiFi-BT，需在开发板上为 R120、R135、R74 安装 0 欧姆电阻，并且断开 R83。对于 CY8CKIT-062-BLE，需在开发板上断开 R84。
※ 使用本模块时，CY8CKIT-062 开发板上的橘色 LED8 不能使用。

程序按以下次序对电路进行扫描控制：

	RA2 ~ RA0 引脚驱动模式 /输出值	COL3 ~ COL0 引脚驱动模式 /输出值	EN 引脚 驱动模式 /输出值	74138 输出 nQ7~4	74138 输出 nQ3~0	电路工作状态
复位	高阻模拟	高阻模拟	高阻模拟	1111	1111	A/B、C 断 E 全灭 D 不影响 E
初始化，iRow = 8	强驱动/111	电阻上拉/1111	强驱动/0	1111	1111	A/B、C 断 E 全灭 D 不影响 E
1. 写 COL 为 1111（防止 LED 灯条短暂亮灯）、写 EN 为 0（准备设置 LED 灯条） 2. 写 RA、COL 设置 LED 灯条 3. iRow = 9	强驱动/111	电阻上拉/1111	强驱动/0	1111	1111	A/B、C 断 E 工作 D 不影响 E
获取五向旋转编码开关+S17 状态（阶段 1）： 1. I/O 设置准备扫描五向旋转编码开关+S17 2. iRow = 10	高阻数字输入/???	开漏低驱动无上拉/1111	强驱动/0	1111	1111	A/B、C 断 D 工作（除 S16 Down 键） E 不影响 D

后页续表.....

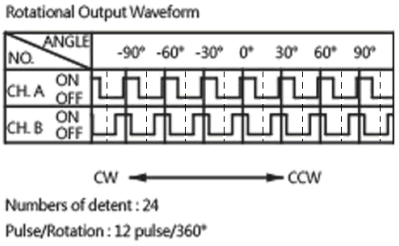
- 初始化之后的各行为一次扫描，一轮扫描共 10 次扫描。每轮扫描取 20ms（兼顾按键消抖和 LED 防闪烁），则各次扫描的间隔应为 2ms。每轮 10 次扫描时，不必包含*深紫色的操作*。
- 浅紫色底纹的各次扫描可合并为 1 次（此时*深紫色的操作*为必需），此时一轮扫描共 6 次扫描。每轮扫描取 20ms，则各次扫描的间隔应为 3.3ms。每轮 6 次扫描时，最坏情况下的每次扫描耗时，约比每轮 10 次扫描时大一倍。

	RA2 ~ RA0 引脚驱动模式 /输出值	COL3 ~ COL0 引脚驱动模式 /输出值	EN 引脚 驱动模式 /输出值	74138 输出 nQ7~4	74138 输出 nQ3~0	电路工作状态
获取五向旋转编码开关 +S17 状态（阶段 2）： 1. 读 RA 获取 Center、CHB、CHA 键状态 2. 读 COL 获取 S17、Left、Up、Right 键状态 3. 读 EN 获取 Down 键状态 4. iRow = 0， *5. 加速驱动 COL 至高电平（相比电阻上拉时）*	高阻数字输入 /??? （1.、2.之间）强驱动/111 （4.时）强驱动 /000	开漏低驱动无上拉/1111 （4.时）电阻上拉/1111 *（5.时）强驱动 /1111*	强驱动/0 （2.、3.之间）高阻数字输入/0 （4.时）强驱动/1	1111 （3.时）?111 （4.时）1111	1111 （4.时）1110	C 断 A/B 的 R7LED 行可能很短暂工作但不会亮 D 工作 E 不影响 D （4.时）A/B、E 断 C 工作 D 不影响 C
0. 恢复 COL 电阻上拉 1. 读 COL 获取 C 按键矩阵的 R0 行键状态 2. iRow++，写 RA 为 iRow（准备扫描下一按键行） *3. 加速驱动 COL 至高电平（相比电阻上拉时）*	强驱动/000 （2.时）强驱动 /001	*电阻上拉/1111* *（3.时）强驱动 /1111*	强驱动/1	1111	1110 （2.时）1101	A/B、E 断 C 工作 D 不影响 C
0. 恢复 COL 电阻上拉 1. 读 COL 获取 C 按键矩阵的 R1 行键状态 2. iRow++，写 RA 为 iRow（准备扫描下一按键行） *3. 加速驱动 COL 至高电平（相比电阻上拉时）*	强驱动/001 （2.时）强驱动 /010	*电阻上拉/1111* *（3.时）强驱动 /1111*	强驱动/1	1111	1101 （2.时）1011	A/B、E 断 C 工作 D 不影响 C
0. 恢复 COL 电阻上拉 1. 读 COL 获取 C 按键矩阵的 R2 行键状态 2. iRow++，写 RA 为 iRow（准备扫描下一按键行） *3. 加速驱动 COL 至高电平（相比电阻上拉时）*	强驱动/010 （2.时）强驱动 /011	*电阻上拉/1111* *（3.时）强驱动 /1111*	强驱动/1	1111	1011 （2.时）0111	A/B、E 断 C 工作 D 不影响 C
0. 恢复 COL 电阻上拉 1. 读 COL 获取 C 按键矩阵的 R3 行键状态 2. iRow++，写 RA 为 4（准备设置 R4LED），写 COL 设置 A/B LED 矩阵的 R4LED 灯 3. iRow = 5	强驱动/011 （2.时）强驱动 /100	*电阻上拉/1111* （2.时）电阻上拉/R4LED 各值	强驱动/1	1111 （2.时）1110	0111 （2.时）1111	A/B、E 断 C 工作 D 不影响 C （2.时）A/B 工作 C、E 断 D 不影响 A/B

	RA2 ~ RA0 引脚驱动模式 /输出值	COL3 ~ COL0 引脚驱动模式 /输出值	EN 引脚 驱动模式 /输出值	74138 输出 nQ7~4	74138 输出 nQ3~0	电路工作状态
1. 写 COL 为 1111（防止下一 LED 行短暂亮灯） 2. 写 RA 为 iRow（准备设置 R5LED），写 COL 设置 A/B LED 矩阵的 R5LED 灯 3. iRow = 6	强驱动/100 （2.时）强驱动/101	电阻上拉/1111 （2.时）电阻上拉/R5LED 各值	强驱动/1	1110 （2.时）1101	1111	A/B 工作 C、E 断 D 不影响 A/B
1. 写 COL 为 1111（防止下一 LED 行短暂亮灯） 2. 写 RA 为 iRow（准备设置 R6LED），写 COL 设置 A/B LED 矩阵的 R6LED 灯 3. iRow = 7	强驱动/101 （2.时）强驱动/110	电阻上拉/1111 （2.时）电阻上拉/R6LED 各值	强驱动/1	1101 （2.时）1011	1111	A/B 工作 C、E 断 D 不影响 A/B
1. 写 COL 为 1111（防止下一 LED 行短暂亮灯） 2. 写 RA 为 iRow（准备设置 R7LED），写 COL 设置 A/B LED 矩阵的 R7LED 灯 3. iRow = 8	强驱动/110 （2.时）强驱动/111	电阻上拉/1111 （2.时）电阻上拉/R7LED 各值	强驱动/1	1011 （2.时）0111	1111	A/B 工作 C、E 断 D 不影响 A/B
转到初始化的下一行.....						

旋转编码开关（CHB、CHA）的旋转检测：

为完全消除开关的抖动影响，首先检测CHB、CHA同时断开，约20ms 后再次确认仍为同时断开（否则重新检测 CHB、CHA 同时断开）。之后若检测到 CHA 先闭合时，判定为顺时针（CW）旋转动作；若检测到 CHB 先闭合时，则判定为逆时针（CCW）旋转动作。之后再重新开始检测 CHB、CHA同时断开。



此方案在确认了两开关同时断开后，至下一次某开关闭合之前，这期间两开关均处于无抖动的断开状态，可以完全消除抖动影响。但对于本型号的旋转编码开关，会损失角度分辨率，即旋转两个卡位时才会检测到一次旋转动作。

组件更改

本节列出了该组件各版本中的主要更改内容。

版本	更改说明	更改原因/影响
1.0	新组件	