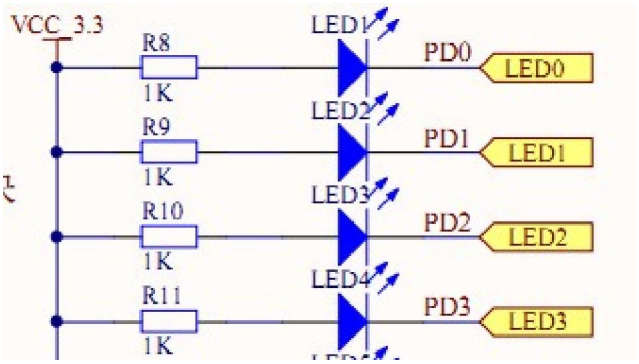
例程一 流水灯

流水灯实验是每个电子爱好者学任何一款单片机必备先学会的一个实验，因为流水灯的操作就是简单的对 IO 口的直接操作，是初学者最喜欢的一个实验，但你会了点亮一个灯的时候，证明了你对这款单片机有了初步的了解和初步的操作，以后那个模块就相当来说就好办了。

下面介绍下在风驰电子 STM8 开发板实现流水灯操作要用到的资源



这是开发板上的电路连接图。只要给 PD0、PD1、PD2、PD3 任何一个低电平，灯就会亮了。

流水灯要用到的内部资源

"stm8s_clk.h"
"stm8s_clk.c"
"stm8s_gpio.h"
"stm8s_gpio.c"
"stm8s.h"

好了，我们先看下主函数

```
int main(void)
{
  /* Infinite loop */

  /*设置内部高速时钟16M为主时钟*/
   CLK_HSIPrescalerConfig(CLK_PRESCALER_HSIDIV1);
  /*!<Set High speed internal clock */
   LED_Init();
   SetLedOFF(); /* 让所有灯灭 */
  while (1)
  {
    /*  添加你的代码  */
    #ifdef Bit_or_Port
    LED_ShowOneToOne();
    /*   以上操作的是最简单的单个LED的点亮 利用宏定义实现的  */
    #else
    LED_Display();
    /*   实现位移操作   */
    #endif


  }
```

这里是一些初始化，

```
CLK_HSIPrescalerConfig(CLK_PRESCALER_HSIDIV1);
/*!<Set High speed internal clock */
```

这个就是切换到外部时钟的函数，这个函数在 STM8S 工程模板的文档说得很详细了，在这里就不多说了。

LED_Init(); LED 的一些初始化，必须要初始化这个函数，否则，LED 就不能正常工作了，下面的 LED_Init()函数原型

```
void LED_Init(void)
{
   GPIO_Init(GPIOD, (GPIO_PIN_0|GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3),\
             GPIO_MODE_OUT_PP_HIGH_FAST );//定义LED的管脚的模式
}
```

这个函数的意思是定义 LED 灯的的管脚位 PD0、PD1、PD2、PD3 为推挽输出

/*!< Output push-pull, high level, 10MHz */

相应的管脚模式是可以选择的，在 stm8s_gpio.h 的文件里面有相应的定义

```
typedef enum
{
  GPIO_MODE_IN_FL_NO_IT      = (uint8_t)0x00,  /*!< Input floating, no external interrupt */
  GPIO_MODE_IN_PU_NO_IT      = (uint8_t)0x40,  /*!< Input pull-up, no external interrupt */
  GPIO_MODE_IN_FL_IT         = (uint8_t)0x20,  /*!< Input floating, external interrupt */
  GPIO_MODE_IN_PU_IT         = (uint8_t)0x60,  /*!< Input pull-up, external interrupt */
  GPIO_MODE_OUT_OD_LOW_FAST  = (uint8_t)0xA0,  /*!< Output open-drain, low level, 10MHz */
  GPIO_MODE_OUT_PP_LOW_FAST  = (uint8_t)0xE0,  /*!< Output push-pull, low level, 10MHz */
  GPIO_MODE_OUT_OD_LOW_SLOW  = (uint8_t)0x80,  /*!< Output open-drain, low level, 2MHz */
  GPIO_MODE_OUT_PP_LOW_SLOW  = (uint8_t)0xC0,  /*!< Output push-pull, low level, 2MHz */
  GPIO_MODE_OUT_OD_HIZ_FAST  = (uint8_t)0xB0,  /*!< Output open-drain, high-impedance level,10MHz
  GPIO_MODE_OUT_PP_HIGH_FAST = (uint8_t)0xF0,  /*!< Output push-pull, high level, 10MHz */
  GPIO_MODE_OUT_OD_HIZ_SLOW  = (uint8_t)0x90,  /*!< Output open-drain, high-impedance level, 2MHz
  GPIO_MODE_OUT_PP_HIGH_SLOW = (uint8_t)0xD0   /*!< Output push-pull, high level, 2MHz */
}GPIO_Mode_TypeDef;
```

在 LED_ShowOneToOne();中实现了单个 LED 的点亮或熄灭
函数原型

```
void LED_ShowOneToOne(void)
{
    LED1(ON);
    LED2(OFF);
    LED3(OFF);
    LED4(OFF);
    Delay(0x1ffff);
    LED1(OFF);
    LED2(ON);
    LED3(OFF);
    LED4(OFF);
    Delay(0x1ffff);
    LED1(OFF);
    LED2(OFF);
    LED3(OFF);
    LED4(ON);
    Delay(0x1ffff);
    LED1(OFF);
    LED2(OFF);
    LED3(ON);
    LED4(OFF);
    Delay(0x1ffff);
}
```

LED1(ON)这个是宏定义来实现的，这样看起来就简单明了。
我们再看看宏定义

```
#define ON  0
#define OFF 1
#define LED1(ON_OFF)  if(ON_OFF==ON)GPIO_WriteLow(GPIOD, GPIO_PIN_0);\
                      else GPIO_WriteHigh(GPIOD, GPIO_PIN_0)

#define LED2(ON_OFF)  if(ON_OFF==ON)GPIO_WriteLow(GPIOD, GPIO_PIN_1);\
                      else GPIO_WriteHigh(GPIOD, GPIO_PIN_1)

#define LED3(ON_OFF)  if(ON_OFF==ON)GPIO_WriteLow(GPIOD, GPIO_PIN_2);\
                      else GPIO_WriteHigh(GPIOD, GPIO_PIN_2)

#define LED4(ON_OFF)  if(ON_OFF==ON)GPIO_WriteLow(GPIOD, GPIO_PIN_3);\
                      else GPIO_WriteHigh(GPIOD, GPIO_PIN_3)
```

我们再看看在 STM8 里面也可以直接对每个 IO 口置 1 或写 0，跟 51 单片机一样的，主要在下面这两个函数实现，先看下他们的函数原型

```
/**
  * @brief  Writes low level to the specified GPIO pins.
  * @note   The port must be configured in output mode.
  * @param  GPIOx : Select the GPIO peripheral number (x = A to I).
  * @param  PortPins : Specifies the pins to be turned low to the port output.
  *         data register.
  * @retval None
  */
void GPIO_WriteLow(GPIO_TypeDef* GPIOx, GPIO_Pin_TypeDef PortPins)
{
    GPIOx->ODR &= (uint8_t)(~PortPins);
}
```

在这个函数中，只有你配置好是哪个 IO 口，他就会相对于的该 IO 口设为 0，即是低电平。在我们的例程其中一个的话就是设置参数 GPIOD,GPIO_PIN_0，就好象 51 里面的 P3_0=0；操作起来就好像 51 的一样的操作。

```
/**
  * @brief  Writes high level to the specified GPIO pins.
  * @note   The port must be configured in output mode.
  * @param  GPIOx : Select the GPIO peripheral number (x = A to I).
  * @param  PortPins : Specifies the pins to be turned high to the port output.
  *         data register.
  * @retval None
  */
void GPIO_WriteHigh(GPIO_TypeDef* GPIOx, GPIO_Pin_TypeDef PortPins)
{
    GPIOx->ODR |= (uint8_t)PortPins;
}
```

同样这个就是给 IO 口置 1，也就是置高电平。
大家再看看下面的函数就是实现

```
LED_Display();
/*   实现位移操作    */
```

函数原型

```
void LED_Display(void)
{
  u8 PortVal;
  for(PortVal=0;PortVal<4;PortVal++)
  {
      if(PortVal==2)
        GPIOD->ODR =(u8)~(1<<(PortVal+1));
      else if(PortVal==3)
        GPIOD->ODR =(u8)~(1<<(PortVal-1));
      else
      GPIOD->ODR =(u8)(~(1<<PortVal));

      Delay(0x1ffff);
  }
}
```

这个函数写得有点复杂，是因为为了让 LED 在风驰电子 STM8 开发板按顺序的流起来。主要看下这条语句就可以了

```
GPIOD->ODR =(u8)(~(1<<PortVal));
```

这条语句的让第几盏灯亮，当 PortVal=0 的话就是我们开发板上的 LED0 亮。在这里用左移操作符<<和取反操作符~。相信看过 C 语言的知道是什么意思了，在这里不多说了。
还有一个函数大家需要知道的

```
GPIO_Write(GPIOD, 0xff);
```

这个函数就是直接对某个端口的 8 位直接赋值。看看它的函数原型

```
/**
  * @brief  Writes data to the specified GPIO data port.
  * @note   The port must be configured in output mode.
  * @param  GPIOx : Select the GPIO peripheral number (x = A to I).
  * @param  GPIO_PortVal : Specifies the value to be written to the port output
  *         data register.
  * @retval None
  */
void GPIO_Write(GPIO_TypeDef* GPIOx, uint8_t PortVal)
{
    GPIOx->ODR = PortVal;
}
```

但大家看到这里的话，相信大家对 IO 的操作就不在话下了。好的，对 IO 口的操作就分析到这里。

实验现象

当大家把我的例程下载到风驰电子 STM8 开发板就可以看到流水灯按照一定的频率兜圈不断的跑

风驰电子祝您学习愉快 ～～～!!!!!