

XI`AN TECHNOLOGICAL UNIVERSITY

实验报告

实验课程名称 ARM嵌入式系统原理及其应用开发

专 业： 物联网工程

班 级： 16060616

姓 名： 田宇龙

学 号： 16060616107

实验学时：

指导教师：

成 绩：

2018 年 6 月 18 日

西安工业大学实验报告

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 专业 | 物联网工程 | 班级 | 16 | 姓名 | 田宇龙 | 学号 | 16060616107 |
| 实验课程 | I/O接口实验 | 指导教师 |  | 实验日期 | 2018 | 同实验者 |  |
| 实验项目 | I/O接口实验 | | | | | | |
| 实验设备及器材 | PC机、Keil，ProteusARM仿真软件，windows xp | | | | | | |

**一丶实验目的**

1）通过实验掌握ARM 芯片使用I/O 口控制LED 显示。

2）了解ARM 芯片中复用I/O 口的使用方法。

1. **实验内容**

自我进行创新利用ARM LPC20xx系列进行流水灯的输出

1. **实验原理**

**以LPC2124为例：**

LPC2124的64脚封装最多可以有46个GPIO。GPIO的特性有

单独的方向控制位

单独控制的置位和清零

所有I/O在复位后默认为输入

GPIO的控制寄存器有：

管脚值寄存器IOxPIN

输出置位寄存器IOxSET

输出清零寄存器IOxCLR

方向寄存器IOxDIR

LPC2124的引脚一般是多个功能复用的，可以通过引脚连接模块在多个功能之间进行选择。引脚连接模块通过配置寄存器控制多路开关来连接引脚和片内外设（寄存器PINSELx）。

引脚连接模块的用途是将引脚设置为需要的功能，可实现独立的引脚配置。

外设在激活和任何相关中断使能之前，必须连接到适当的引脚。

引脚连接模块包含三个寄存器，分别是PINSEL0,PINSEL1,PINSEL2，其中PINSEL0,PINSEL1这两个寄存器控制P0口。PINSEL2控制P1口。

例如，要设置P0.0为GPIO功能，则应进行如下设置：

PINSEL0=PINSEL0&0xFFFFFFFC,这样P0.0只能作为GPIO使用，不能用作其它功能。

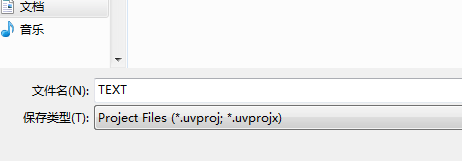
所以，GPIO的一般操作步骤是：

先通过PINSELx寄存器，将端口设置为GPIO功能，然后通过IOxDIR寄存器，选择GPIO是作为输入还是输出口。如果作为输入口，那么可以通过IOxPIN读出端口的值。如果作为输出口，则可以通过IOxSET，IOxCLR设置端口输出高电平还是低电平

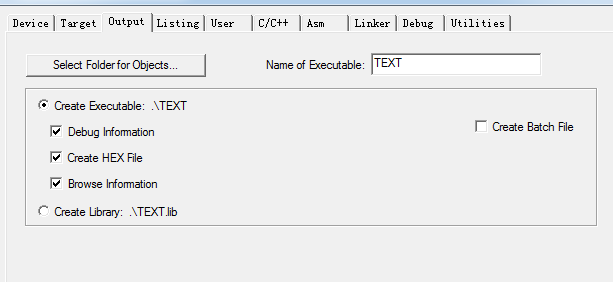
1. **实验过程**

本次实验采用的软件Keil编辑软件，ProteusARM仿真软件

1. 在软件Keil中选择Project-->New μVision Project，弹出文本窗，建立名为TEXT的包

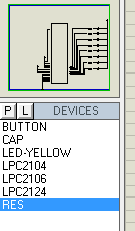


1. 点击File创建新输入文本，输入流水灯相关源代码并保存为text.c文件
2. 在工程处点击鼠标右键点击Add Existing File to Group，选择刚才保存的TEXT.C文件，将文件加入在工程中
3. 文件加入成功后进行相关配置，鼠标左键单击该图标，弹出设置框，在Output栏进行如下配置



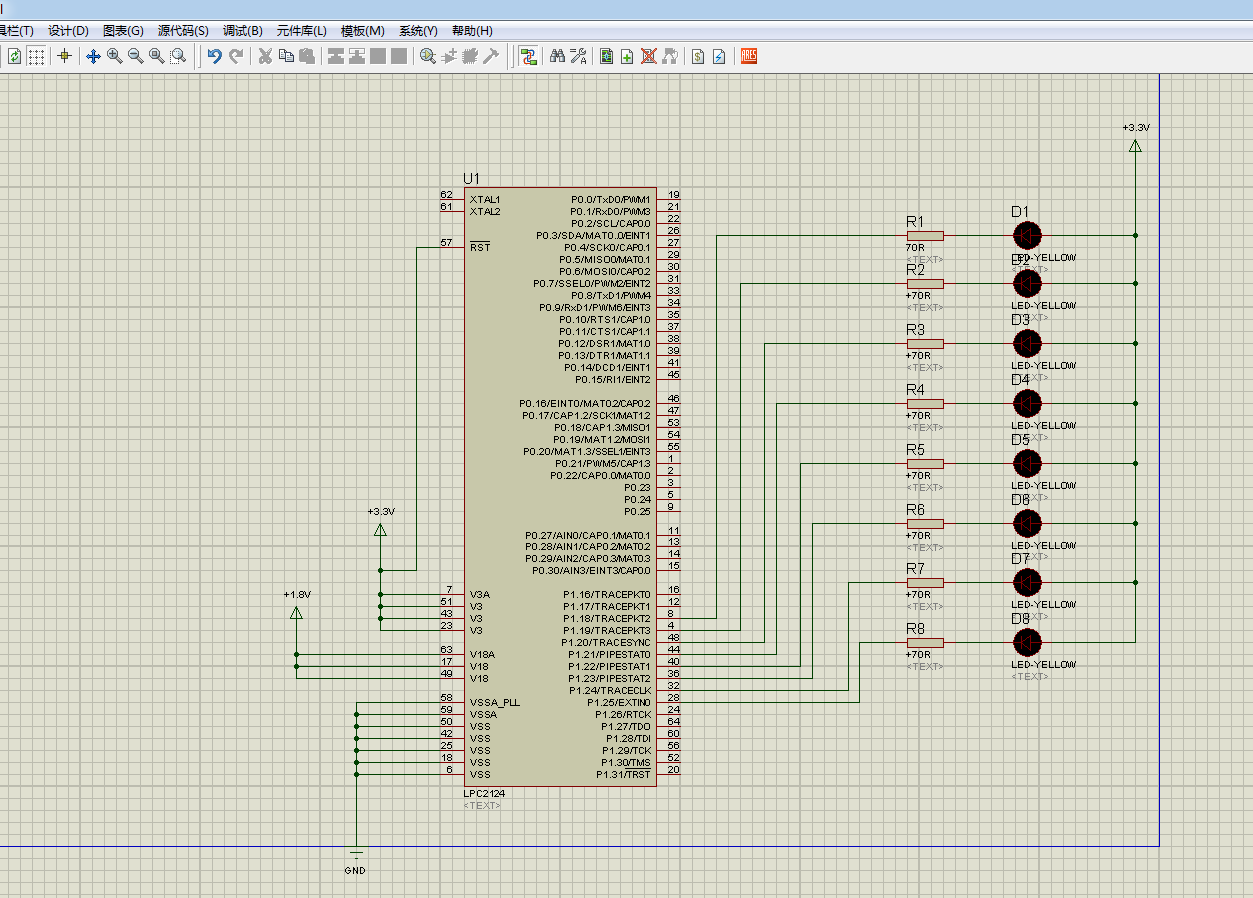
完成后点击ok即可

1. 对文件进行编译处理，若程序正常编译后则会出现一个后缀为.hex的文件，该文件用于ProtuesARM仿真软件内电路图的运行。
2. 打开ProtuesARM仿真软件，建立新的文件包，本实验采用Protues7.8版本。
3. 在软件左选择栏处对程序电路图器件进行选择

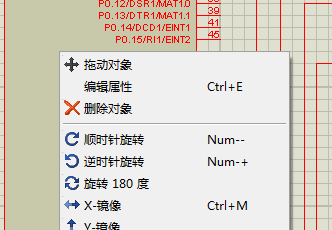


本次实验为了清楚直观的看清LED灯的变化均采用颜色为黄的LED灯。

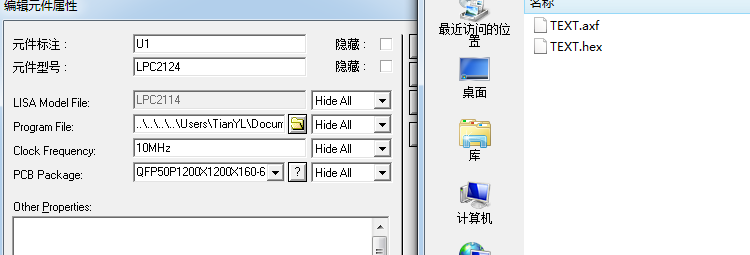
具体电路图如下：



1. 点击芯片LPC2124，右键将弹出下列内容



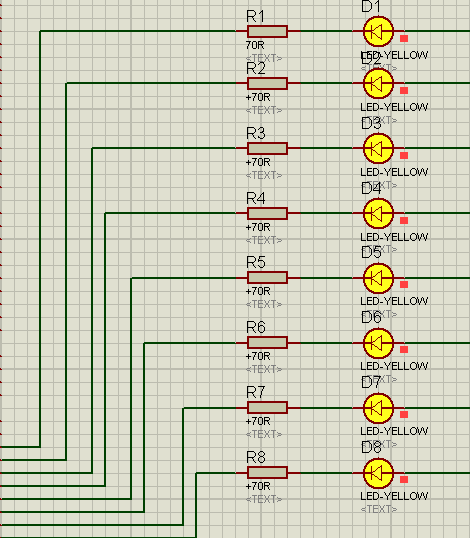
选择编辑属性，将刚才编译好的TEXT.hex文件加入

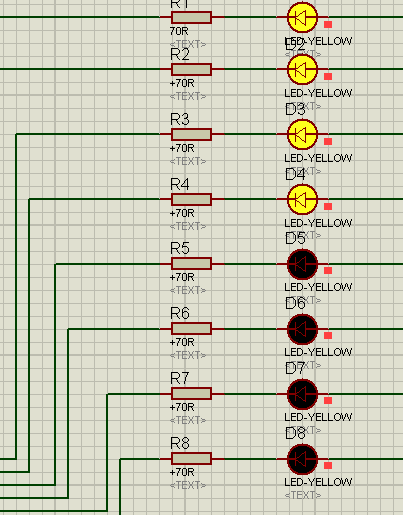


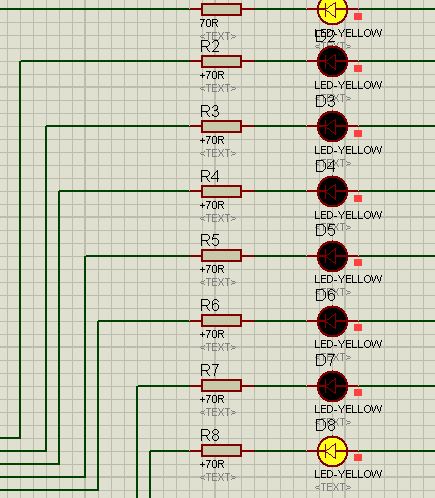
1. 运行程序：

可按F12或对电路图进行运行。

1. 运行效果图







1. **程序源代码**

**#include <lpc21xx.h>**

**#define uint8 unsigned char**

**#define uint16 unsigend short**

**#define uint32 unsigned int**

**const uint32 LED8 = (0xff << 18); //8个LED分别连接到P1.25-P1.18**

**void delayms(uint32 delay) //流水灯延迟函数**

**{**

**uint32 i;**

**for(;delay > 0;delay--)**

**for(i = 0;i < 5000;i++);**

**}**

**const uint32 LED\_TBL[] = { //流水灯样式设置**

**0x00,0xff, //全部熄灭然后全部点亮**

**0x01,0x02,0x04,0x08,0x10,0x20,0x40,0x80, //依次逐个点亮**

**0x01,0x03,0x07,0x0f,0x1f,0x3f,0x7f,0xff, //依次逐个叠加点亮**

**0xff,0x7f,0x3f,0x1f,0x0f,0x07,0x03,0x01, //依次逐个递减点亮**

**0x81,0x42,0x24,0x18,0x18,0x24,0x42,0x81, //两个靠拢后分开**

**0x81,0xc3,0xe7,0xff,0xff,0xe7,0xc3,0x81, //从两边叠加后递减**

**};**

**// 主程序**

**int main(void)**

**{**

**uint8 i;**

**PINSEL2 = PINSEL2 & (~0x80); //设置P1.25-P1.18为GPIO模式**

**IO1DIR = LED8; //设置为输出口**

**while(1)**

**{**

**for(i = 0;i < 42;i++)**

**{**

**IO1SET = ~((LED\_TBL[i]) << 18); //低电平点亮LED**

**delayms(100); //**

**IO1CLR = (LED\_TBL[i]) << 18; //**

**delayms(100);**

**}**

**}**

**}**

**六、心得体会**

本次实验相比于前几次实验更为复杂，不仅要考虑到程序的可行性分析，还要顾及仿真器中电路图之间的高低电平的分析，低电平为0，高电平为1，才可对LED灯进行相关样式的模拟，对芯片的引脚设置更是较为复杂，通过本次实验大幅度提升的我对相关芯片控制的知识，希望以后会更有进步！