实验一: 软件仿真实验

1. 实验目的

- 学习 Keil uVision 集成开发环境使用。
- 掌握 LPC1100 工程的创建、工程属性配置,掌握程序编译方法。
- 学习 Keil uVision 软件仿真功能,掌握基本的程序调试技巧;
- 编写 GPIO 输出、输入程序;利用软件仿真的外设视图学习 GPIO 使用。

2. 实验设备

- 硬件: PC 机一台
- 软件: WindowsXP 系统, Keil uVision 4.x 集成开发环境

3. 实验内容

- (1) 建立 Keil uVision 工程,编写 GPIO 输出程序,编译程序。
- (2) 使用 Keil uVision 软件仿真,单步、全速运行程序,设置断点,打开寄存器窗口监视 RO、R1等寄存器的值
 - (3) 使用软件仿真的外设视图学习调试 GPIO 输出程序。

4. 实验预习要求

- (1) 学习 Cortex-MO 处理器的软件编程模型,重点掌握 LPC1100 程序的结构和执行过程。
 - (2) 查阅 Keil uVision 软件的介绍,了解软件的功能和操作方法。
 - (3) 复习 GPIO 的工作原理和编程方法。

5. 实验步骤

(1) 启动 Keil uVision,新建一个工程 ex01。见图 1-1、图 1-2、图 1-3。

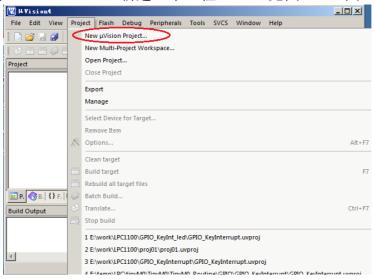


图 1-1 建立工程

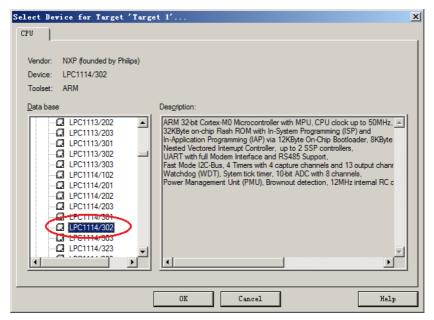


图 1-2 选择目标芯片



图 1-3 不需要系统提供的 Startup 文件

(2) 建立 C 语言源文件 ex01. c,编写实验程序,然后添加到工程中。见图 1-4、图 1-5、图 1-6、图 1-7。

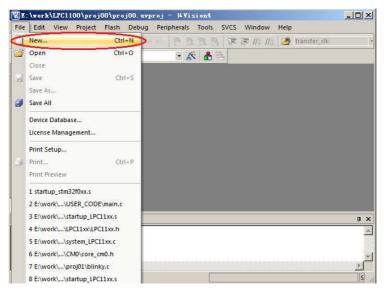


图 1-4 建立新文件



图 1-5 保存扩展名为.c 的程序文件

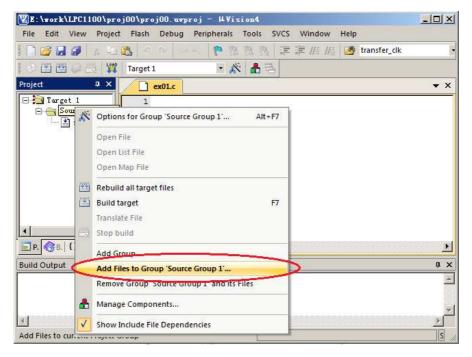


图 1-6 把 C 文件添加到工程

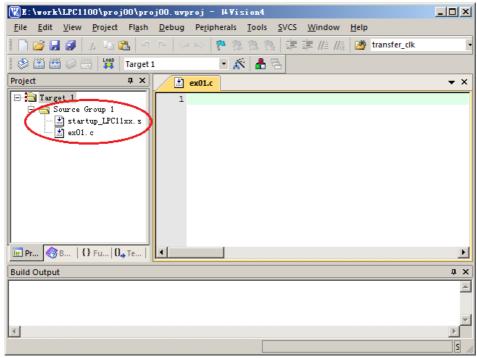


图 1-7 添加文件到工程后

(3)设置工程选项。见图 1-8、图 1-9。

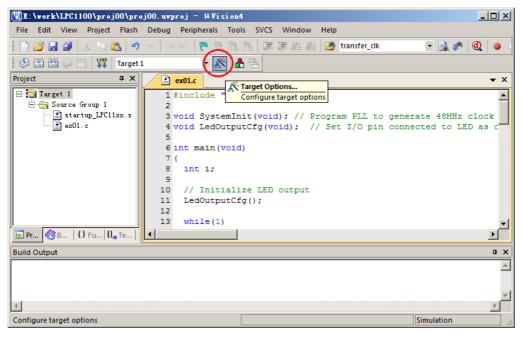


图 1-8 设置工程选项

♥ Options for Target 'Target 1'	
Device Target Output Listing User C/C++ Asm Linker Debug Utilities	
© Use Simulator Settings ☐ Limit Speed to Real-Time	© Use: J-LINK / J-Trace Cortex Settings
✓ Load Application at Startup ☐ Run to main() Initialization File: Edit	Load Application at Startup Run to main() Initialization File:
Restore Debug Session Settings Breakpoints W Toolbox Watch Windows & Performance Analyzer Memory Display	Restore Debug Session Settings V Breakpoints V Toolbox V Watch Windows V Tracepoints V Memory Display
CPU DLL: Parameter: SARMCM3.DLL	Driver DLL: Parameter:
Dialog DLL: Parameter: DARMP1.DLL PLPC1114	Dialog DLL: Parameter: TARMP1.DLL pLPC1114
OK Cancel Defaults Help	

图 1-9 设置软件仿真

(4)编译链接工程。见图 1-10。

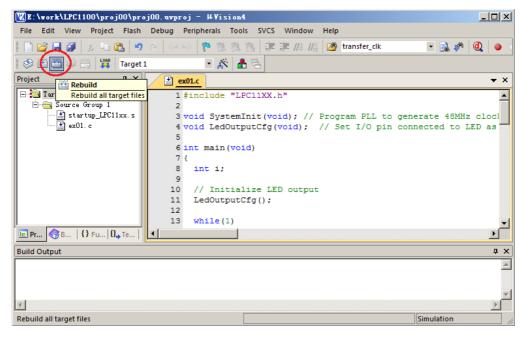


图 1-10 编译工程

(5)进行软件仿真调试。见图 1-11、图 1-12、图 1-13、图 1-14。

```
▼E:\work\LPC1100\proj00\proj00.uvproj - #Vision4

                                                                                                                   _ | _ | ×
 File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
                                                                                      🎒 🏥 🕮 🕢 Longet 1
                                       · 🔉 Å 🖹
                                                                                                Start/Stop Debug Session (Ctrl+1
Project
                     ъ ×
                                                                                                   Enter or leave a debug session
                               ex01.c
 F Target 1
                                   #include "LPC11XX.h"
   Source Group 1

startup_LPC11xx.s

ex01. c
                                 3 void SystemInit(void); // Program PLL to generate 48MHz clock
4 void LedOutputCfg(void); // Set I/O pin connected to LED as output
                                    int i;
                                10
                                    // Initialize LED output
                                    LedOutputCfg();
                                12
                                13 while(1)
Build Output
linking...
Program Size: Code=472 RO-data=296 RW-data=0 ZI-data=608
"proj00.axf" - 0 Error(s), 0 Warning(s).
                                                                                                                       •
Enter or leave a debug session
                                                                                   Simulation
```

图 1-11 调试运行

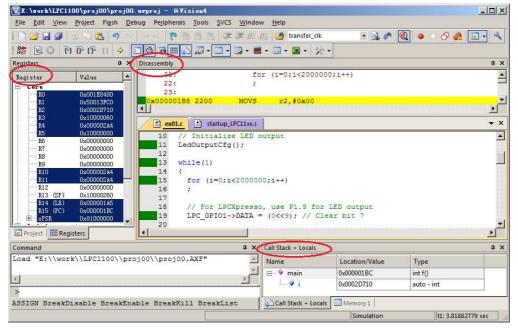


图 1-12 调试界面

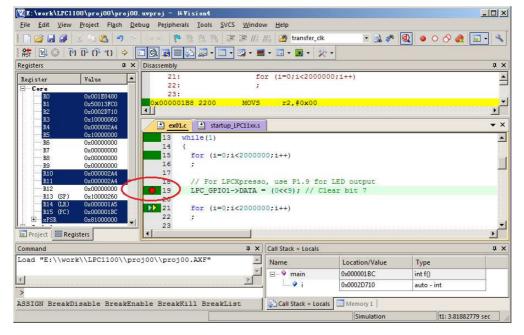


图 1-13 设置断点

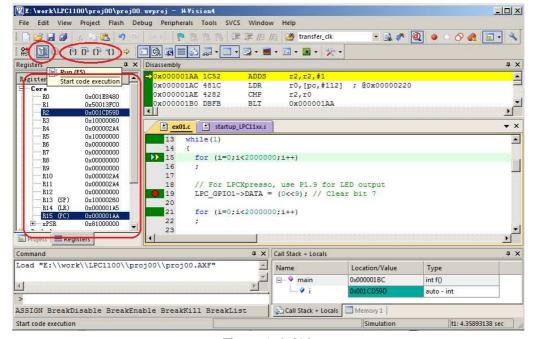


图 1-14 运行程序

(6)打开软件仿真的 GPIO 状态视图,观察 GPIO 外设寄存器值和引脚状态。 单步执行程序,观察 GPIO 状态变化。见图 1-15、图 1-16。

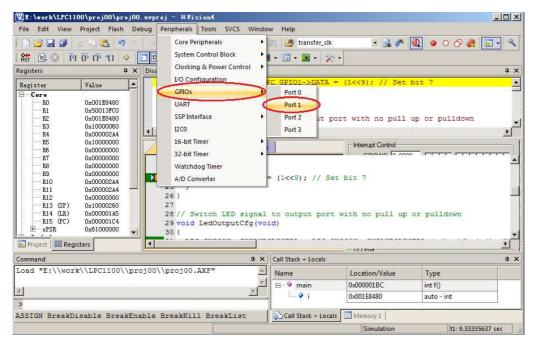


图 1-15 打开软件仿真的 GPIO 视图

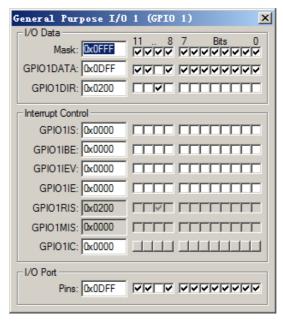


图 1-16 软件仿真的 GPIO 状态视图

6. 实验参考程序

实验的参考程序见程序清单1.1。

程序清单 1.1 软件仿真实验参考程序

#include "LPC11XX.h"

```
// Switch LED signal to output port with no pull up or pulldown
void LedOutputCfg(void)
{
   LPC_SYSCON->SYSAHBCLKCTRL = LPC_SYSCON->SYSAHBCLKCTRL | (1<<16) | (1<<6);</pre>
```

```
LPC_10C0N->P101_9 = (0x0) + (0<<3) + (0<<5);
 // Set pin 9 as output
 LPC_GPIO1->DIR = LPC_GPIO1->DIR \mid (1<<9);
 return;
// Switch the CPU clock frequency to 48MHz
void SystemInit(void)
 LPC_SYSCON->PDRUNCFG = LPC_SYSCON->PDRUNCFG & 0xFFFFFF5F;
 // Select PLL source as crystal oscillator
 LPC_SYSCON->SYSPLLCLKSEL = 1;
 // Update SYSPLL setting (0->1 sequence)
 LPC SYSCON->SYSPLLCLKUEN = 0;
 LPC SYSCON->SYSPLLCLKUEN = 1;
  LPC SYSCON->SYSPLLCTRL = (3 + (1 << 5)); // M = 4, P = 2
 // wait until PLL is locked
 while(LPC_SYSCON->SYSPLLSTAT == 0);
 LPC SYSCON->MAINCLKSEL = 3;
  // Update Main Clock Select setting (0->1 sequence)
 LPC_SYSCON->MAINCLKUEN = 0;
 LPC_SYSCON->MAINCLKUEN = 1;
 return;
int main(void)
 int i;
 // Initialize LED output
 LedOutputCfg();
 while (1)
   {
   for (i=0; i<2000000; i++)
      ;
   // use P1.9 for LED output
   LPC_GPIO1->DATA = (0<<9); // Clear bit 9
       for (i=0; i<2000000; i++)
       LPC_GPI01-DATA = (1<<9); // Set bit 9
 }
```

7. 实践、观察、思考

- (1)尝试改写程序,观察哪些变量会分配 RAM 存储空间?
- (2)编写嵌套函数调用,观察栈的内容和栈指针的位置。
- (3)观察 C 程序编译后生成的机器指令。
- (4)编写 GPIO 输入程序,并利用软件仿真 GPIO 状态视图调试程序。