华清远见 FARSIGHT 始于2004 | 嵌入式培训专家

10年口碑积累，成功培养50000多名研发工程师，铸就专业品牌形象

华清远见的企业理念是不仅要做良心教育、做专业教育，更要做受人尊敬的职业教育。

《ARM 嵌入式 C 编程标准教程》

作者：华清远见

专业始于专注 卓识源于远见

第 3 章 ARM9 芯片 S3C2410 的片上资源

专业始于专注 卓识源于远见

3.1 S3C2410 处理器介绍

S3C2410 微处理器是一款由 Samsung 公司为手持设备设计的低功耗、高度集成的基于 ARM920T 核的微处理器。为了降低系统总成本和减少外围器件，这款芯片中还集成了下列部件：16KB 指令 Cache、16KB 数据 Cache、MMU、外部存储器控制器、LCD 控制器（STN 和 TFT）、NAND Flash 控制器、4 个 DMA 通道、3 个 UART 通道、1 个 I^2C 总线控制器、1 个 I^2S 总线控制器、4 个 PWM 定时器、一个内部定时器、通用 I/O 口、实时时钟、8 通道 10 位 ADC 和触摸屏接口、USB 主、USB 从、SD/MMC 卡接口等。现在它广泛应用于 PDA、移动通信、路由器、工业控制等领域，其内部结构如图 3-1 所示。

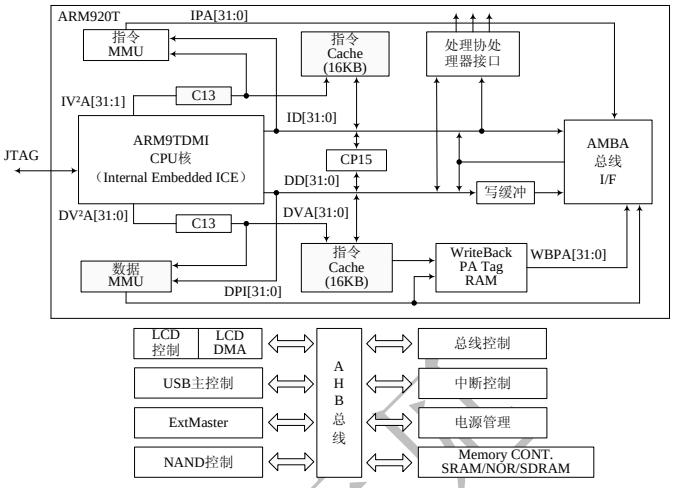

图 3-1 S3C2410 结构框图

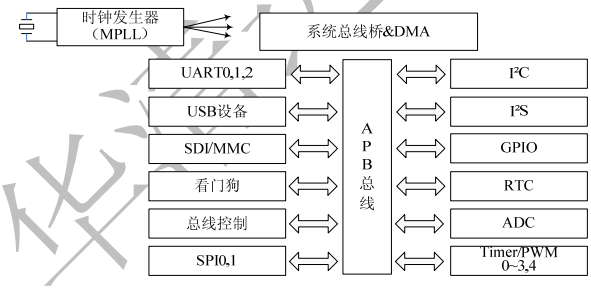

图 3-1 S3C2410 结构框图（续）

为了提高系统运行速度，减少能量损失，ARM920T 核微处理器把片上器件按器件工作频率、使用频度分成三个模块，各个模块通过各自总线连接，模块之间采用一种叫总线桥的结构过渡。下面简单介绍一下各总线特点。

3.1.1 AMBA、AHB、APB 总线特点

随着深亚微米工艺技术日益成熟，集成电路芯片的规模越来越大。数字 IC 从基于时序驱动的设计方法，发展到基于 IP 复用的设计方法，并在片上系统（SoC）设计中得到了广泛应用。在基于 IP 复用的 SoC 设计中，片上总线设计是最关键的问题。为此，业界出现了很多片上总线标准。其中，由 ARM 公司推出的 AMBA 片上总线受到了广大 IP 开发商和 SoC 系统集成者的青睐，已成为一种流行的标准片上结构。AMBA 规范主要包括 AHB(Advanced High performance Bus)系统总线和 APB(Advanced Peripheral Bus)外围总线。

AMBA 2.0 规范包括 4 个部分：AHB、ASB、APB 和 Test Methodology。AHB 的相互连接采用了传统的带有主模块和从模块的共享总线，接口与互连功能分离，这对芯片上模块之间的互连具有重要意义。AMBA 已不仅是一种总线，更是一种带有接口模块的互连体系。下面将简要介绍 AHB 和 APB 总线。

大多数挂在总线上的模块（包括处理器）只是单一属性的功能模块：主模块或者从模块。主模块是向从模块发出读写操作的模块，如 CPU、DSP 等；从模块是接受命令并做出反应的模块，如片上的 RAM、AHB/APB 桥等。另外，还有一些模块同时具有两种属性，例如直接存储器存取模块(DMA)在编程时是从模块，但在系统传输数据时必须是主模块。

如果总线上存在多个主模块，就需要仲裁器来决定如何控制各种主模块对总线的访问。虽然仲裁规范是 AMBA 总线规范中的一部分，但具体使用的算法由 RTL 设计工程师决定，其中两个最常用的算法是固定优先级算法和循环制算法。

AHB 总线上最多可以有 16 个主模块和任意多个从模块，如果主模块数目大于 16，则需再加一层结构（具体参阅 ARM 公司推出的 Multi-layer AHB 规范）。APB 桥既是 APB 总线上唯一的主模块，也是 AHB 系统总线上的从模块。其主要功能是锁存来自 AHB 系统总线的地址、数据和控制信号，并提供二级译码以产生 APB 外围设备的选择信号，从而实现 AHB 协议到 APB 协议的转换。

AHB 主要用于高性能模块（如 CPU、DMA 和 DSP 等）之间的连接，作为 SoC 的片上系统总线，它包括以下一些特性：单个时钟边沿操作，非三态的实现方式，支持突发传输，支持分段传输，支持多个主控制器，可配置 32 位～128 位总线宽度，支持字节、半字节和字的传输。

APB 主要用于低带宽的周边外设之间的连接，例如 UART 等，它的总线架构不像 AHB 支持多个主模块，在 APB 里面唯一的主模块就是 APB 桥，其特性包括：两个时钟周期传输，无需等待周期和回应信号，控制逻辑简单，只有 4 个控制信号。

3.1.2 S3C2410 处理器体系结构

- ARM920T 核，16 位/32 位 RISC（精简指令系统）结构和 ARM 精简指令集。
- ARM MMU，支持 Windows CE, Linux 等操作系统。
- 指令 Cache、数据 Cache、写缓冲。
- 支持 ARM 调试结构，片上 ICE 支持 JTAG 调试方式。
- 内置先进微控制器总线接口（AMBA）。

3.1.3 S3C2410 处理器管理系统

- 支持大端（Big Endian）/小端（Little Endian）模式。
- 地址空间为每个内存块 128MB（一共 1GB），每个内存块支持 8/16/32 位数据总线编程。
- 8 个内存块，6 个用于 ROM、SRAM 和其他，2 个用于 ROM/SRAM/SDRAM。
- 1 个起始地址和大小可编程的内存块（Bank7）。
- 7 个起始地址固定的内存块（Bank0～Bank6）。
- 所有内存块可编程寻址周期。
- 支持 SDRAM 自动刷新模式。
- 支持多种类型 ROM 启动，包括 Nor/Nand Flash、EEPROM 等。

3.1.4 S3C2410 处理器存储器映射

S3C2410 的存储空间映射如图 3-2 所示。

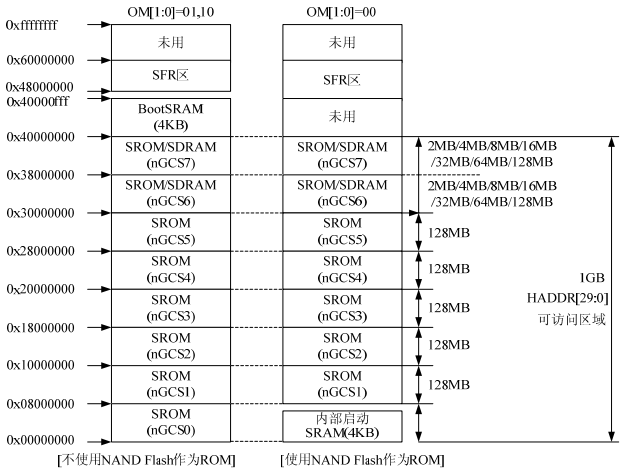

图 3-2　S3C2410 存储区地址映射

3.1.5　S3C2410 处理器时钟和电源管理

1．时钟

S3C2410 的主时钟由外部晶振或者外部时钟提供，选择后可以产生 3 种时钟信号，分别是 CPU 使用的 FCLK、AHB 总线使用的 HCLK 和 APB 总线使用的 FCLK。时钟管理模块同时拥有两个锁相环，一个称为 MPLL，拥于 FCLK、HCLK 和 PCLK；另一个称为 UPLL，用于 USB 设备。

2．时钟源选择

对时钟的选择是通过 S3C2410 引脚上 OM[3:2]实现的，JU 具体如表 3-1 所示。

表 3-1　　时钟源选择

OM[3:2]	MPLL 状态	UPLL 状态	主 时 钟 源	USB 时钟源
00	On	On	Crystal	Crystal
01	On	On	Crystal	EXTCLK
10	On	On	EXTCLK	Crystal
11	On	On	EXTCLK	EXTCLK

S3C2410 引脚的 OM[3:2]=00 时，晶体为 MPLL CLK 和 UPLL CLK 提供时钟源；OM[3:2]=01 时，晶体为 MPLL CLK 提供时钟源，EXTCLK 为 UPLL CLK 提供时钟源；OM[3:2]=10 时，EXTCLK 为 MPLL CLK 提供时钟源，晶体为 UPLL CLK 提供时钟源；OM[3:2]=11 时，EXTCLK 为 MPLL CLK 和 UPLL CLK 提供时钟源。

3．时钟控制逻辑

时钟控制逻辑决定了所使用的时钟源，是采用 MPLL 作为 FCLK，还是采用外部时钟。复位后，即使不想改变默认的 PLLCON 值，也需要重新写一遍。FCLK 由 ARM920T 核使用，HCLK 提供给 AHB 总线，PCLK 提供给了 APB 总线。S3C2410 支持 HCLK、FCLK 和 PCLK 的分频选择，其比率是通过 CLKDIV 寄存器中的 HDIVN 和 PDIVN 控制的，如表 3-2 所示。

表 3-2 分频设定表

HDIVN	PDIVN	FCLK	HCLK	PCLK	Divide Ratio
	0	FCLK	FCLK	FCLK	1:1:1(default)
0	1	FCLK	FCLK	FCLK/2	1:1:2
1	0	FCLK	FCLK/2	FCLK/2	1:2:2
1	1	FCLK	FCLK/2	FCLK/4	1:2:4(recommended)

4. 电源管理

S3C2410 电源管理模块通过 4 种模式有效地控制功耗，即正常（Normal）模式、省电（Slow）模式、空闲（Idle）模式和断电（Power-off）模式。

- Normal 模式：为 CPU 和所有的外设提供时钟，所有的外设开启，该模式下的功耗最大。这种模式允许用户通过软件控制外设，可以断开提供给外设的时钟以降低功耗。
- Slow 模式：采用外部时钟生产 FCLK 的方式，此时电源的功耗取决于外部时钟。
- Idle 模式：断开 FCLK 与 CPU 核的连接，外设保持正常，该模式下的任何中断都可唤醒 CPU。
- Power-off 模式：断开内部电源，只给内部的唤醒逻辑供电。一般模式下需要两个电源，一个提供给唤醒逻辑，另外一个提供给 CPU 和内部逻辑，在 Power-off 模式下，后一个电源关闭。该模式可以通过 EINT[15:0]和 RTC 唤醒。

5. 时钟和电源管理寄存器

S3C2410 通过控制寄存器实现对时钟和电源的管理，相关寄存器如表 3-3 所示。

表 3-3 时钟控制器

寄存器	地址	读/写	说明	复位后值
LOCKTIME	0x4C000000	读/写	PLL 锁定时间寄存器	0x00FFFFFF
MPLLCON	0x4C000004		MPLL 配置寄存器：MDIV=[19:12],PDIV=[9:4],SDIV=[1:0]	0x0005C080
UPLLCON	0x4C000008		UPLL 配置寄存器，同上	0x00028080
CLKCON	0x4C00000C		时钟信号生成控制寄存器	0x7FFF0
CLKSLOW	0x4C000010		Slow 时钟控制寄存器	0x00000004
CLKDIVN	0x4C000014		分频控制寄存器	0x00000000

3.2 S3C2410 处理器片上资源的定义和使用

在 MCS-51 单片机开发系统中，所有的系统资源都在 reg51.h 头文件中予以定义，我们在编程时将 reg51.h 头文件引入我们的程序，在程序中就可以很方便地使用这些系统资源。

和开发 MCS-51 单片机一样，S3C2410 在头文件 2410addr.h 中，将 S3C2410 的所有系统资源都进行了定义，我们在编写 S3C2410 的驱动程序时必需引用这个头文件。

2410addr.h 将系统所有的资源进行了宏定义，宏的名称就是所定义的寄存器的名字前面加一个小写的“r”，方便记忆。

2410addr.h 内容包括：Memory control、USB Host、INTERRUPT、DMA、CLOCK & POWER MANAGEMENT、LCD CONTROLLER、Nand flash、UART、PWM TIMER、USB DEVICE、WATCH DOG TIMER、IIC、IIS、I/O PORT、RTC、ADC、SPI、ISR、SD Interface、PENDING BIT 等，近 20 类。Memory control、I/O PORT、LCD CONTROLLER、Nand flash 寄存储器的宏定义如下：

```
// Memory control
#define rBWSCON    (*(volatile unsigned *)0x48000000) //Bus width & wait status
#define rBANKCON0  (*(volatile unsigned *)0x48000004) //Boot ROM control
#define rBANKCON1  (*(volatile unsigned *)0x48000008) //BANK1 control
#define rBANKCON2  (*(volatile unsigned *)0x4800000c) //BANK2 cControl
#define rBANKCON3  (*(volatile unsigned *)0x48000010) //BANK3 control
#define rBANKCON4  (*(volatile unsigned *)0x48000014) //BANK4 control
#define rBANKCON5  (*(volatile unsigned *)0x48000018) //BANK5 control
#define rBANKCON6  (*(volatile unsigned *)0x4800001c) //BANK6 control
#define rBANKCON7  (*(volatile unsigned *)0x48000020) //BANK7 control
#define rREFRESH   (*(volatile unsigned *)0x48000024) //DRAM/SDRAM refresh
#define rBANKSIZE  (*(volatile unsigned *)0x48000028) //Flexible Bank Size
#define rMRSRB6    (*(volatile unsigned *)0x4800002c) //Mode register set for SDRAM
#define rMRSRB7    (*(volatile unsigned *)0x48000030) //Mode register set for SDRAM
//I/O PORT
#define rGPACON    (*(volatile unsigned *)0x56000000) //Port A control
#define rGPADAT    (*(volatile unsigned *)0x56000004) //Port A data
#define rGPBCON    (*(volatile unsigned *)0x56000010) //Port B control
#define rGPBDAT    (*(volatile unsigned *)0x56000014) //Port B data
#define rGPBUP     (*(volatile unsigned *)0x56000018) //Pull-up control B
#define rGPCCON    (*(volatile unsigned *)0x56000020) //Port C control
#define rGPCDAT    (*(volatile unsigned *)0x56000024) //Port C data
#define rGPCUP     (*(volatile unsigned *)0x56000028) //Pull-up control C
#define rGPDCON    (*(volatile unsigned *)0x56000030) //Port D control
#define rGPDDAT    (*(volatile unsigned *)0x56000034) //Port D data
#define rGPDUP     (*(volatile unsigned *)0x56000038) //Pull-up control D
#define rGPECON    (*(volatile unsigned *)0x56000040) //Port E control
#define rGPEDAT    (*(volatile unsigned *)0x56000044) //Port E data
#define rGPEUP     (*(volatile unsigned *)0x56000048) //Pull-up control E
#define rGPFCON    (*(volatile unsigned *)0x56000050) //Port F control
#define rGPFDAT    (*(volatile unsigned *)0x56000054) //Port F data
#define rGPFUP     (*(volatile unsigned *)0x56000058) //Pull-up control F
#define rGPGCON    (*(volatile unsigned *)0x56000060) //Port G control
#define rGPGDAT    (*(volatile unsigned *)0x56000064) //Port G data
#define rGPGUP     (*(volatile unsigned *)0x56000068) //Pull-up control G
#define rGPHCON    (*(volatile unsigned *)0x56000070) //Port H control
#define rGPHDAT    (*(volatile unsigned *)0x56000074) //Port H data
#define rGPHUP     (*(volatile unsigned *)0x56000078) //Pull-up control H
// LCD CONTROLLER
```

```
#define rLCDCON1   (*(volatile unsigned *)0x4d000000) //LCD control 1
#define rLCDCON2   (*(volatile unsigned *)0x4d000004) //LCD control 2
#define rLCDCON3   (*(volatile unsigned *)0x4d000008) //LCD control 3
#define rLCDCON4   (*(volatile unsigned *)0x4d00000c) //LCD control 4
#define rLCDCON5   (*(volatile unsigned *)0x4d000010) //LCD control 5
#define rLCDSADDR1  (*(volatile unsigned *)0x4d000014) //Frame buffer start address 1
#define rLCDSADDR2  (*(volatile unsigned *)0x4d000018) //Frame buffer start address 2
#define rLCDSADDR3  (*(volatile unsigned *)0x4d00001c) //Virtual screen address set
#define rREDLUT    (*(volatile unsigned *)0x4d000020) //STN Red lookup table
#define rGREENLUT   (*(volatile unsigned *)0x4d000024) //STN Green lookup table
#define rBLUELUT   (*(volatile unsigned *)0x4d000028) //STN Blue lookup table
#define rDITHMODE   (*(volatile unsigned *)0x4d00004c) //STN Dithering mode
#define rTPAL      (*(volatile unsigned *)0x4d000050) //TFT Temporary palette
#define rLCDINTPND  (*(volatile unsigned *)0x4d000054) //LCD Interrupt pending
#define rLCDSRCPND  (*(volatile unsigned *)0x4d000058) //LCD Interrupt source
#define rLCDINTMSK  (*(volatile unsigned *)0x4d00005c) //LCD Interrupt mask
#define rLPCSEL    (*(volatile unsigned *)0x4d000060)  //LPC3600 Control
#define PALETTE    0x4d000400                //Palette start address
// NAND flash
#define rNFCONF    (*(volatile unsigned *)0x4e000000) //NAND Flash configuration
#define rNFCMD     (*(volatile U8 *)0x4e000004)    //NADD Flash command
#define rNFADDR    (*(volatile U8 *)0x4e000008)    //NAND Flash address
#define rNFDATA    (*(volatile U8 *)0x4e00000c)    //NAND Flash data
#define rNFSTAT    (*(volatile unsigned *)0x4e000010) //NAND Flash operation status
#define rNFECC     (*(volatile unsigned *)0x4e000014) //NAND Flash ECC
#define rNFECC0    (*(volatile U8  *)0x4e000014)
#define rNFECC1    (*(volatile U8  *)0x4e000015)
#define rNFECC2    (*(volatile U8  *)0x4e000016)
```

在我们开发的程序中，引用 2410addr.h 头文件以后，就可以使用这些宏定义来对这些寄存器进行操作，这些寄存器的用法下面陆续介绍。更详细内容可参见随书提供软件包，参考程序 24120TEST 中的 2410addr.h。

3.3 编程参考软件包 2410TEST

在厂家提供的资料中，有一个 2410TEST 软件包，里面包括几乎所有 S3C2410 硬件驱动的 C 语言例子和头文件，仔细阅读这些程序对我们编程有很大帮助。2410TEST 软件包在随书提供软件包中。

在软件包中有一个 2410test.c 程序，是一个实验 S3C2410 各项功能程序，主要部分列出如下，并做必要解释。

```
//-------------------------------------------------------------------------
//      引入实验所需头文件
//-------------------------------------------------------------------------
#include <stdlib.h>
#include <string.h>
#include "def.h"
#include "option.h"
#include "2410addr.h"
#include "2410lib.h"
#include "2410slib.h"
#include "2410etc.h"
#include "2410IIC.h"
#include "2410iis.h"
#include "2410int.h"
```

```
#include "2410RTC.h"
#include "2410swi.h"
#include "timer.h"
#include "adc.h"
#include "dma.h"
#include "dma2.h"
#include "eint.h"
#include "extdma.h"
#include "k9s1208.h"
#include "mmu.h"
#include "nwait.h"
#include "sdi.h"
#include "stone.h"
#include "ts_auto.h"
#include "ts_sep.h"
#include "usbfifo.h"
#include "IrDA.h"
#include "lcd.h"
#include "lcdlib.h"
#include "glib.h"
#include "palette.h"
#include "spi.h"
#include "uart0.h"
#include "uart1.h"
#include "uart2.h"
#include "etc.h"
#include "flash.h"
#include "idle.h"
#include "pd6710.h"
#include "pll.h"
#include "power.h"
#include "pwr_c.h"
#include "stop.h"
//-------------------------------------------------------------------------
//  定义一个二维的函数指针数组，数组中第一列是函数指针，第二列是函数功能提示
//-------------------------------------------------------------------------
void * function[][2]=
{
//ADC, TSP
(void *)Test_Adc,                    "ADC              ",
(void *)Test_DMA_Adc,                "ADC with DMA      ",
(void *)Ts_Sep,                      "ADC TSP Seperate   ",
(void *)Ts_Auto,                     "ADC TSP Auto      ",
//DMA
(void *)Test_DMA,                    "DMA M2M          ",
(void *)Test_DMAWorst,                "DMA Worst Test     ",
(void *)Test_Dma0Xdreq,                "External DMA      ",
//EINT
(void *)Test_Eint,                     "External Interrupt  ",
//IIC
(void *)Test_Iic,                      "IIC(KS24C080)INT   ",
(void *)Test_Iic2,                     "IIC(KS24C080)POL   ",
//IIS
```

```
(void *)Record_Iis,                    "Reco IIS UDA1341    ",
(void *)Test_Iis,                    "Play IIS UDA1341    ",
//Interrupt
(void *)Test_Fiq,                    "FIQ Interrupt       ",
(void *)Change_IntPriorities,          "Change INT Priority ",
//IrDA
(void *)Test_IrDA_Rx,                "UART2 IrDA Rx       ",
(void *)Test_IrDA_Tx,                "UART2 IrDA Tx       ",
//LCD
(void *)Test_Lcd_Stn_1Bit,            "STN 1Bit            ",
(void *)Test_Lcd_Stn_2Bit,            "STN 2Bit            ",
(void *)Test_Lcd_Stn_4Bit,            "STN 4Bit            ",
(void *)Test_Lcd_Cstn_8Bit,           "CSTN  8Bit          ",
(void *)Test_Lcd_Cstn_8Bit_On,         "CSTN  8Bit On       ",
(void *)Test_Lcd_Cstn_12Bit,          "CSTN 12Bit          ",
(void *)Test_Lcd_Tft_8Bit_240320,      "TFT240320  8Bit     ",
(void *)Test_Lcd_Tft_8Bit_240320_On,    "TFT240320  8Bit On  ",
(void *)Test_Lcd_Tft_16Bit_240320,     "TFT240320 16Bit     ",
(void *)Test_Lcd_Tft_1Bit_640480,      "TFT640480  1Bit     ",
(void *)Test_Lcd_Tft_8Bit_640480,      "TFT640480  8Bit     ",
(void *)Test_Lcd_Tft_16Bit_640480,     "TFT640480 16Bit     ",
(void *)Test_Lcd_Tft_8Bit_640480_Bswp,  "TFT640480 BSWP      ",
(void *)Test_Lcd_Tft_8Bit_640480_Palette, "TFT640480 Palette   ",
(void *)Test_Lcd_Tft_16Bit_640480_Hwswp,"TFT640480 HWSWP     ",
//Memory
//MPLL
(void *)Test_PLL,                    "MPLL Change        ",
(void *)ChangePLL,                   "MPLL MPS Change     ",
(void *)Test_PllOnOff,                "MPLL On/Off         ",
//PMS
(void *)Test_SlowMode,                "PMS Slow           ",
(void *)Test_HoldMode,                "PMS Hold           ",
(void *)Test_IdleMode,                "PMS Idle           ",
(void *)Test_MMUIdleMode,             "PMS Idle(MMU)       ",
(void *)Test_IdleModeHard,            "PMS Idle Hard       ",
(void *)Test_InitSDRAM,               "PMS SDRAM Init      ",
(void *)Test_StopMode,                "PMS STOP           ",
(void *)Test_PowerOffMode,            "PMS Power-Off STOP  ",
(void *)Test_PowerOffMode_100Hz,       "PMS Power-Off 100Hz ",
(void *)MeasurePowerConsumption,       "PMS Measure Power   ",
//RTC
(void *)Test_Rtc_Alarm,               "RTC Alarm          ",
(void *)Display_Rtc,                 "RTC Display        ",
(void *)RndRst_Rtc,                  "RTC Round Reset     ",
(void *)Test_Rtc_Tick,                "RTC Tick           ",
//SDI
(void *)Test_SDI,                    "SDI Write/Read      ",
//SPI
(void*) Test_Spi_MS_int,              "SPI0 RxTx Int       ",
(void *)Test_Spi_MS_poll,             "SPI0 RxTx POLL      ",
(void *)Test_Spi_M_Tx_DMA1,           "SPI0 Master Tx DMA1 ",
(void *)Test_Spi_S_Rx_DMA1,           "SPI0 Slave Rx DMA1  ",
(void *)Test_Spi_M_Rx_DMA1,           "SPI0 Master Rx DMA1 ",
```

```
(void *)Test_Spi_S_Tx_DMA1,          "SPI0 Slave Tx DMA1  ",
(void *)Test_Spi_M_Int,              "SPI0 Master RxTx INT",
(void *)Test_Spi_S_Int,              "SPI0 Slave RxTx INT ",
//Timer
(void *)Test_TimerInt,               "Timer Interrupt     ",
(void *)Test_Timer,                  "Timer Tout         ",
//UART
(void *)Test_Uart0_Int,              "UART0 Rx/Tx Int     ",
(void *)Test_Uart0_Dma,              "UART0 Rx/Tx DMA     ",
(void *)Test_Uart0_Fifo,             "UART0 Rx/Tx FIFO    ",
(void *)Test_Uart0_AfcTx,            "UART0 AFC Tx        ",
(void *)Test_Uart0_AfcRx,            "UART0 AFC Rx        ",
(void *)Test_Uart1_Int,              "UART1 Rx/Tx Int     ",
(void *)Test_Uart1_Dma,              "UART1 Rx/Tx DMA     ",
(void *)Test_Uart1_Fifo,             "UART1 Rx/Tx FIFO    ",
(void *)Test_Uart1_AfcTx,            "UART1 AFC Tx        ",
(void *)Test_Uart1_AfcRx,            "UART1 AFC Rx        ",
(void *)Test_Uart2_Int,              "UART2 Rx/Tx Int     ",
(void *)Test_Uart2_Dma,              "UART2 Rx/Tx DMA     ",
(void *)Test_Uart2_Fifo,             "UART2 Rx/Tx FIFO    ",
//USB
(void *)Test_USBFIFO,                "USB FIFO Test      ",
//WDT
(void *)Test_WDT_IntReq,             "WDT INT Request     ",
//ETC
(void *)Test_XBREQ,                  "External Bus Reqest ",
(void *)Test_NonalignedAccess,        "NonAlgined Access   ",
(void *)Test_PD6710,                 "PC Card (PD6710)    ",
(void *)ReadPageMode,                "Read Page Mode      ",
(void *)Test_SwiIrq,                 "SWI                ",
(void *)Test_WaitPin,                "External Wait       ",
(void *)Test_ISram,                  "Stone Test         ",
(void *)Test_NecInterrupt,           "ETC NEC Int         ",
(void *)Test_BattFaultInterrupt,      "nBATT_FAULT int     ",
//NAND, NOR Flash
(void *)K9S1208_PrintBadBlockNum,     "NAND View Bad Block ",
(void *)K9S1208_PrintBlock,          "NAND View Page      ",
(void *)K9S1208_Program,             "NAND Write         ",
(void *)TestECC,                     "NAND ECC           ",
(void *)ProgramFlash,                "NOR Flash Program   ",
0,0
};
//---------------------------------------------------------------------------
// 主程序
//---------------------------------------------------------------------------
void Main(void)
{
   int i;
   MMU_Init();                                  //内存管理初始化
   ChangeClockDivider(1,1);                         // 定义 FCLK、HCLK、PCLK 比例
                                                    //1: 2: 4
   ChangeMPllValue(0xa1,0x3,0x1);                   // FCLK=202.8MHz
   Port_Init();                                 //I/O 口初始化
```

```
    Isr_Init();                                  //中断初始化
    Uart_Init(0,115200) ;                        //选串口 0 与上位机通信，波特率 115200
    Uart_Select(0);
    Check_PowerOffWakeUp();                      //唤醒电源进入正常工作状态
         while(1)
    {
         i = 0;
         while(1)
         {                                              //显示主菜单
          Uart_Printf("%2d:%s",i,function[i][1]);
          i++;
          if((int)(function[i][0])==0)              //显示结束跳出
          {
              Uart_Printf("\n");
              break;
          }
          if((i%4)==0)
          Uart_Printf("\n");                                  //每行显示 4 项
         }
         Uart_Printf("\nSelect the function to test : ");    //提示：选择某项实验
         i = Uart_GetIntNum();                                //读实验项目号放 i 中
         Uart_Printf("\n");
         //GPG4 Output Port [9:8] 01  G 口初始化，开 LCD 显示
         rGPGCON = (rGPGCON & 0xfffffcff) | (1<<8);
         rGPGDAT = (rGPGDAT & 0xffef) | (1<<4);
           if(i>=0 && (i<(sizeof(function)/8)) )
        ( (void (*)(void)) (function[i][0]) )();//如果 i 在实验范围内，做第 i 项实验
    }
}
```

2410test.c 程序提供了 S3C2410 所有硬件资源的驱动例子，根据我们系统的需要，再参考具体函数，对我们编程会有很大帮助。

3.4 习题与练习

1．打开 2410addr.h 头文件，仔细阅读程序，熟悉程序。
2．打开 2410TEST 文件，熟悉程序 UART0.C 和 UART0.H，练习串口编程和 I/O 口操作，熟悉中断程序编制。
3．打开 2410TEST 文件，熟悉练习串口编程和 I/O 口操作。
4．void*function[][2]定义的是一个什么类型的二维数组，数组中每个元素代表什么变量？
5．2410addr.h 将系统所有的资源进行了宏定义，宏的名字有什么特点？
6．S3C2410 电源管理模块通过几种模式有效地控制功耗？
7．S3C2410 处理器体系结构中有几个存储器模块，每个 Bank 的大小是多少？总的 Bank 大小是多少？

联系方式

集团官网：www.hqyj.com　　嵌入式学院：www.embedu.org　　移动互联网学院：www.3g-edu.org

企业学院：www.farsight.com.cn　　物联网学院：www.topsight.cn　　研发中心：dev.hqyj.com

集团总部地址：北京市海淀区西三旗悦秀路北京明园大学校内 华清远见教育集团

北京地址：北京市海淀区西三旗悦秀路北京明园大学校区，电话：010-82600386/5

上海地址：上海市徐汇区漕溪路 250 号银海大厦 11 层 B 区，电话：021-54485127

深圳地址：深圳市龙华新区人民北路美丽 AAA 大厦 15 层，电话：0755-25590506

成都地址：成都市武侯区科华北路 99 号科华大厦 6 层，电话：028-85405115

南京地址：南京市白下区汉中路 185 号鸿运大厦 10 层，电话：025-86551900

武汉地址：武汉市工程大学卓刀泉校区科技孵化器大楼 8 层，电话：027-87804688

西安地址：西安市高新区高新一路 12 号创业大厦 D3 楼 5 层，电话：029-68785218

广州地址：广州市天河区中山大道 268 号天河广场 3 层，电话：020-28916067