# <<STM32嵌入式微控制器快速上手>>

#### 图书基本信息

书名: <<STM32嵌入式微控制器快速上手>>

13位ISBN编号: 9787121148804

10位ISBN编号:7121148803

出版时间:2012-1

出版时间:电子工业出版社

作者:陈志旺

页数:308

版权说明:本站所提供下载的PDF图书仅提供预览和简介,请支持正版图书。

更多资源请访问:http://www.tushu007.com

# <<STM32嵌入式微控制器快速上手>>

#### 内容概要

本书介绍了意法半导体(ST,STMicroelectronics)公司的基于ARM Cortex-M3内核的STM32单片机应用与实践。

本书以培养动手能力和增强工程素养为目的,按照项目驱动的思路展开讲解,以开发板自制相关程序为实例,系统介绍了STM32单片机的引脚特性、内部结构、片上资源、开发方法和应用编程等内容。

## <<STM32嵌入式微控制器快速上手>>

#### 书籍目录

#### 第1章 嵌入式系统概述

- 1.1 嵌入式系统简介
- 1.2 ARM体系结构及微处理器系列
- 1.3 Cortex-M3简介
- 1.4 STM32的发展
- 1.5 STM32教学开发板
- 第2章 Cortex-M3体系结构
- 2.1 CM3微处理器核结构
- 2.2 处理器的工作模式及状态
- 2.3 寄存器
- 2.4 总线接口
- 2.5 存储器的组织与映射
- 2.6 指令集
- 2.7 流水线
- 2.8 异常和中断
- 2.9 STM32微控制器概述
- 第3章 STM32程序设计
- 3.1 嵌入式C语言知识精编
- 3.2 嵌入式软件层次结构
- 3.3 Cortex微控制器软件接口标准
- 3.4 FWLib固件库
- 3.5 嵌入式C编程标准
- 第4章 STM32电源、时钟及复位电路
- 4.1 电源电路
- 4.2 时钟电路
- 4.3 复位电路
- 4.4 启动设置
- 第5章 STM32的GPIO
- 5.1 GPIO的硬件结构及功能
- 5.2 GPIO控制寄存器
- 5.3 应用实例
- 第6章 STM32中断系统
- 6.1 STM32中断源
- 6.2 STM32中断优先级
- 6.3 外部中断/事件硬件结构(EXTI)
- 6.4 外部中断寄存器配置
- 6.5 中断过程
- 6.6 EXTI 寄存器
- 6.7 STM32外部中断应用实例
- 第7章 STM32通用同步/异步收发器USART
- 7.1 端口复用
- 7.2 USART功能和结构
- 7.3 USART帧格式
- 7.4 波特率设置
- 7.5 硬件流控制

# <<STM32嵌入式微控制器快速上手>>

- 7.6 USART中断请求
- 7.7 USART寄存器
- 7.8 USART应用实例
- 第8章 STM32定时器
- 8.1 STM32定时器概述
- 8.2 通用定时器TIMx内部结构
- 8.3 通用定时器TIMx功能
- 8.4 通用定时器TIMx寄存器
- 8.5 TIM2应用实例
- 8.6 RTC结构及功能
- 8.7 RTC控制寄存器
- 8.8 备份寄存器
- 8.9 电源控制寄存器
- 8.10 RTC相关的寄存器
- 8.11 RTC应用实例
- 8.12 系统时钟SysTick简介
- 8.13 SysTick寄存器
- 8.14 SysTick应用实例
- 第9章 STM32的DMA
- 9.1 DMA简介
- 9.2 STM32的DMA结构及功能
- 9.3 DMA寄存器
- 9.4 DMA初始化设置
- 第10章 STM32的A/D转换器
- 10.1 ADC硬件结构及功能
- 10.2 工作模式
- 10.3 数据对齐
- 10.4 ADC中断
- 10.5 ADC控制寄存器
- 10.6 ADC程序设计
- 第11章 µ C/OS-II嵌入式操作系统基础
- 11.1 操作系统的作用
- 11.2 操作系统的基本概念
- 11.3 µ C/OS-II简介
- 11.4 µ C/OS-II移植
- 第12章 μ C/OS- 的内核机制
- 12.1 µ C/OS- 内核结构
- 12.2 μ C/OS-II的任务管理
- 12.3 µ C/OS- 的时间管理
- 12.4 任务间的通信与同步
- 附录A ARM常用缩写
- 附录B Cortex-M3指令清单
- 附录C STM32开发板原理图
- 参考文献

# <<STM32嵌入式微控制器快速上手>>

#### 章节摘录

版权页: 插图: 9.1 DMA简介存储器直接访问DMA(Direct Memory Access)传送方式如图9—1所示,它是指一种高速的数据传输操作,允许在外部设备和存储器之间利用系统总线直接读/写数据,既不通过微处理器,也不需要微处理器干预。

整个数据传输操作在一个称为 " DMA控制器 " 的控制下进行。

微处理器除了在数据传输开始和结束时控制一下,在传输过程中微处理器可以进行其他的工作。

DMA还有一个特点是"分散—收集(Scatter—Gather)",它允许在一次单一的DMA处理中传输大量数据到存储区域。

DMA方式可以形象理解为,微机系统是个公司,其中的微处理器是公司经理,外设是员工,内存是仓库,数据就是仓库里存放的物品。

公司规模较小时,公司经理直接管理仓库里的物品,员工若需要使用物品,就直接告诉经理,然后经理去仓库取;员工若采购了物品,也先交给经理,然后经理将物品放进仓库。

公司规模较小时,经理还忙得过来,但当公司规模较大时,会有越来越多的员工(外设)和物品(数据)进出仓库。

此时经理若大部分时间都处理这些事情,就很少有时问做其他事情,于是经理雇了一个仓库保管员, 专门负责"入库"和"出库",只要把"入库"和"出库"的请求单给经理过目同意即可。

后面的 " 入库 " 和 " 出库 " 过程,员工只需要和这个仓库保管员打交道就可以了,而仓库保管员正 是DMA控制器。

在PC中,硬盘工作在DMA下,CPU只需向DMA控制器下达指令,让DMA控制器来处理数据的传送, 数据传送完毕再把信息反馈给微处理器,这样就很大程度上减轻微处理器资源占有率。

现在的手机大都具有照相功能,也可以摄录一些视频短片,只要手机工作到照相机模式,就会将摄像 头的实时画面显示在屏幕上。

如果没有DMA功能,只能是编写程序从摄像头(CMOS Sensor)将实时画面的图像数据取回,然后将 这些数据通过LCD显示,图像数据从CMOS Sensor搬运到LCD的工作需要由程序来完成。

假如,每次搬运一个点的颜色数据,就算是完成QVGA/30帧这样的效果,也需要一次搬运2304000(320×240 X 30)个点。

完成一个点的数据搬运需要微处理器至少做下面的工作:依据当前点位置判断是否向CMOSSensor给出行场同步脉冲信号;向CMOS Sensor给出时钟脉冲信号;读当前点的颜色数据;依据当前点位置判断是否向LCD给出行场同步脉冲信号;向LCD给出时钟脉冲信号;写当前点颜色数据到LCD;更新下一点继续循环。

# <<STM32嵌入式微控制器快速上手>>

编辑推荐

# <<STM32嵌入式微控制器快速上手>>

### 版权说明

本站所提供下载的PDF图书仅提供预览和简介,请支持正版图书。

更多资源请访问:http://www.tushu007.com