前言

说起这本手册的起因，是我去年6月做项目时上时碰到STM8这块芯片。这块芯片性能很强大，但是当时缺少对应的教材。当时我全靠着ST的英文文档，看得比较累。空下来就想，如果能有本教材就好了。毕设选题时，因为自己也做过一些研发类的项目，想换种形式，想到了这件事，就决定干脆自己写本教材。 然后在看一篇博客的时候，看到某大牛，他自身的技术积淀已经很强了，网上最流行的中文makefile教程就是他写的。他写的博文我经常看，感觉很有深度。清华大学出版社出版的两本教材都大段抄袭他的博文（最后被其投诉，并得到了赔偿），他说过这么一句话：“45岁之前绝不出书”，因为他觉得只有到了那个时候，自己才可能得到足够的积淀。能够出的了精品。和他比起来，我又怎敢妄称出书呢?在惠普以及爱立信的实习过程中，我接触了很多软件方面好用的工具，成熟的软件工程思想。比如说git，就是非常好的版本控制管理软件，比如说agile，消除了传统软件开发过程中的部分问题。我觉得，软件开发中的诸多问题，随着编程规模的扩大，参与人数的增多，在硬件开发中也会逐步地体现出来。所以，这些工具及思想方式引入到嵌入式编程，将是一种趋势。本手册亦不指望能够引领什么潮流，只希望能对STM8的学习者，有一个有益的指导，仅此而已。

你不能从本手册中获得

本手册不是官方数据手册的汉化并堆砌，故详细的硬件寄存器资源及软件库函数说明并不能从中得到。同样关于程序语言的章节也不会提供过多C语言的细节。但以上这些在相关章节都会提到相应资源的方法，并且提供经验上的参考。

适合的对象

本手册适合具有C语言功底，并能在51或AVR上进行简单程序设计的单片机爱好者。

本书结构

* 第一章：前言
* 第二章：重读C语言
* 第三章：软件配置管理工具
* 第四章：STM8芯片资源简介
* 第五章：STM8官方软件库简介
* 第六章：STM8S编程实战

封面及封底

因为自己喜欢排版，就自己设计了一个。封面封底的图片都是取景自华师大校园的风景，本手册封面、封底都是使用COREDRAW设计的。

项目地址

本次STM8学习项目托管在github上，并且项目完全开源。欢迎大家访问：www.github.com/vincent5295/stm8teach

如何写作本书的

本手册是使用MARKDOWN格式写作的，感谢larrycai的mkbok项目。

致谢

在本手册的编写过程中，遇到很多困难。在此要感谢各位老师同学在本项目的实践过程中，对本人给予的无私帮助。

目录

第 1 章

重读C语言

1.1 一个小测试

说到C语言，可能你还不以为然。这个语言是众多学院的编程入门语言，似乎当年写的几个程序还挺简单的。那么来做一下下面几个测试，看看自己C语言到底学的怎么样吧！

如果不过瘾的话，这里，还有这里，都可以让你比较深入的去思考，同时相信类似的这些问题在你今后找工作的面试中一定碰得到，除非你不做程序员。 如果说上述的问题并没有

下面是一些开发方面的问题：你有没有使用vim写过一个50行以上的小程序？你有没有使用过gcc编译，再用gdb调试过程序？你有没有写过一个makefile，写好程序后敲一个make，编译、测试、安装或者更多的东西一键搞定。当然如果作为大的软件项目来说，版本控制系统自然是少不了的，那么你有没有使用过Git、CVS、SVN之类的工具，对项目的代码checkin、checkoout、merge、revert？

关于进一步学习C语言的资料，我推荐下列三本书：《C程序设计语言》、[《C专家编程》]http://product.china-pub.com/38005)、《C陷阱与缺陷》。一般来说，以《C程序设计语言》为基础，然后凭借后面两本书，知道C语言平时使用中一些容易犯错的地方。当然，如果你觉得还不够的话，以下这篇文章可能会适合你：如何学好C语言。

第 2 章

用Markdown来写

希望你已经照着上一章生成出了第一个Pdf文件，看到了一本蛮标准的书的样子。

这一章详细点介绍这是如何用Markdown写出来的。

首先简单介绍一下一本书的组成

2.1 标准书稿的组成

一部完整的书稿，通常按顺序由封面、扉页、版权页（含内容简介）、序、前言、目录、正文（含图稿）、附录（可选项）、参考文献（可选项）、符号表（可选项）、索引（可选项）等组成。

详细请看电子出版社的《作译者手册》http://www.phei.com.cn/wstg/zyzxz

封面、扉页、版权页（含内容简介）封底一般有设计师用图形软件做出来单独印刷的。

序、前言、目录、正文（含图稿）、附录（可选项）都是标准提交格式写的。

参考文献（可选项）、符号表（可选项）、索引（可选项）一般应该是自动生成的。

2.2 怎么从Markdown到书

前一章提到过，技术书籍对排版要求不高，不同级别的章节，代码显示和一些图示就可以了。因此有机会用文本的方式一一对应过去。

最基本的Markdown可以完成上面的功能了。

在Latex中设置好书的模板（`latex/template.tex`），如页眉、页脚、目录、颜色等等，一般有经验的人可以帮你搞定。

Pandoc软件会把Markdown文件转换成Latex格式，然后套上上面的模板。

`mkbok`是一个小工具，做了一些额外的定义和调整。

2.2.1 Markdown扩展

基本的Markdown功能不是很全（如没有脚注），因此可以考虑用一些Markdown的扩展。

由于会用Pandoc转换，我建议推荐用Pandoc的Markdown扩展http://johnmacfarlane.net/pandoc/README.html

你有兴趣也可以看看Github的http://github.github.com/github-flavored-markdown/

2.3 如何使用Markdown写书

现在可以看看结构了。

```
$ find zh
zh/preface # 序和前言
zh/chapters # 正文
zh/appendix # 附录
```

2.3.1 标准章节

每一章的第一行基本就是章节名字，应该只出现一次

```
# 用Markdown来写 #
```

其他的小章节用**##**和**###**表示，最好不要有更多的层次。

2.3.2 序、前言、附录

这和其他章节是一样的，只是在PDF的目录显示中章节号和计数不同。

2.3.3 页眉、页脚

这是有Latex设定的，不需要Markdown参与。

2.3.4 目录

这是有Latex自动生成的，不需要Markdown参与。

2.3.5 图片

把图片放在**figures**目录中。

2.3.6 脚注

这是Pandoc扩展Markdown才能支持

2.3.7 代码

基本的Markdown用空四格的方式，不支持代码高亮显示。

我建议使用Pandoc扩展Markdown，它在生成的Epub和Html中支持代码高亮显示（还没搞定）

2.4 中文字体

首先，我用的是Linux环境并且选用的是UTF-8的编码，而不是GBK，否则在github上显示会有问题，不了解这方面的朋友自己找找资料吧，够讲个把小时的。

在产生PDF时，一般建议内嵌中文字体的，但是真正能用的中文字体实际很少，极大多数是有版权的：

* 文鼎开放的四套字体（简报宋、细上海宋、简中楷、中楷），没有一点版权问题，是大部分的中文Linux的缺省安装。
* 文泉驿的几套字体（微米黑、正黑、点阵宋体）是开放但是GPL性质的，所以不是随便可以商用的。
* Adobe有两套开放字体（宋体、黑体）我认为是可以随便用的，忘了在哪里看到这个解释的了。

可以看看Ubuntu免费中文字体的介绍有个认识。

2.5 怎么选择对应字体

一般缺省中文正文字体是宋体、细明体，对应英文Serif类的英文字体：Georgia、Times New Roman等。

标题和重要内容可以选楷体和黑体，对应英文Sans Serif类的英文字体：Arial、Tahoma、Verdana等

技术文章中常见的代码典型的等宽体用黑体，对应英文Monospace类的英文字体：Courier New等

所以对应的在我的中文Latex配置中可选的是：

* font：文鼎的简报宋、细上海宋，文泉驿的点阵宋体，Adobe的宋体
* bold：文鼎的简中楷、中楷，文泉驿的微米黑、正黑，Adobe的黑体
* mono：文泉驿的微米黑、正黑，Adobe的黑体

第 3 章

了解Latex的知识

还没写完！！！

Tex是由

LaTeX（LATEX，音译“拉泰赫”）是一种基于TeX的排版系统，由美国计算机学家莱斯利·兰伯特（Leslie Lamport）在20世纪80年代初期开发，利用这种格式，即使使用者没有排版和程序设计的知识也可以充分发挥由TeX所提供的强大功能，能在几天，甚至几小时内生成很多具有书籍质量的印刷品。

LaTeX使用TeX作为它的格式化引擎，当前的版本是LaTeX2e。

如果需要高质量的书稿，Latex还是非常适合的，至少比Word上档次。这一章用简单的笔墨[1]给一个简要介绍。

XeTeX是一种使用Unicode的TeX排版引擎，并支持一些现代字体技术，例如OpenType。而且XeLaTeX语法与LaTeX相同，还提供了些增强功能，多数LaTeX文档不经修改就能直接用xelatex编译。XeTeX现在已经包含在TexLive发行包中。

XeTex使用的是UTF-8，所以我们的文档不能存为GBK格式。

XeTeX程序：TeX语言的新的实现，即把Tex语言转换为排版的一个新程序。支持Unicode编码和直接访问操作系统字体。

xetex命令：XeTeX程序中的命令，用来编译用Plain TeX格式写的tex文件。

xelatex命令：XeTeX程序中的命令，用来编译用LaTeX格式写的tex文件。

XeTeX 也是一种 “TeX”，因此它的文稿（源文件）也是一种结构化标记的纯文本文档，唯一区别是要求 XeTeX 文稿必须是 Unicode 编码的，最为常用的编码格式是 UTF-8 的。因此，要编辑 XeTeX，必须使用支持 UTF-8 编码的文本编辑器，这里推荐使用 Vim 或 Emacs，使用它们，即便是不安装与 TeX 编辑相关的扩展，也是很容易写 XeTeX 文档的。我的观点是，对于初学者，学习 TeX 之类的结构化标记文档时，一定要坚持手工键入那些标记，只有如此，方能记住常用的标记，待熟稔后，再寻找一些专门的编辑器来用。

目前已经基于 XeTeX 实现了相应的 LaTeX，即 XeLaTeX。

CTeX、CJK、xeCJK之类的不懂，你可以看看LaTeX中文排版自行了解。

http://bbs.ctex.org/viewthread.php?tid=40232&extra=page%3D1&page=1

Latex编辑部：http://zzg34b.w3.c361.com/index.htm

你晕吗？我晕。如果你想对LaTeX了解的更多，建议看看参考资料。

不过，你读完这一章已经够用了。

[1]感谢lyanry写的“XETEX / LaTEX 中文排版之胡言乱语 “，浅显易懂。

3.1 LATEX文稿基本格式

文稿(即用于排版的源文件)包含两部分内容:一部分是正文,也就是需要排版输出的内容;另一部分是排版控制命令,用于控制版面式样,字体,字形等格式.TEX文稿通常以 tex 为文件扩展名.

排版控制命令是以反斜线” ” 开头的字串。有一些排版控制命令带有一些参数,由参 数来修改其默认行为。 排版控制命令的参数有些属于可省略的,有些属于不可省略的。在排版控制命令中,可省略的参数(若不提供这些参数, LaTeX 采用默认参数)置于方括号中,不 可省略的参数(必须要提供的参数)置于花括号中。具体格式可表示如下:

```
\命令名[可省略的参数]{不可省略的参数}
```

XeTeX/Latex 的纯西文的文稿基本格式如下:

```
\documentclass[11pt,a4paper]{article}
\begin{document} Hello World!
\end{document}
```

上面的文稿中,排版控制命令 \documentclass 的可忽略参数告诉 LATEX 系统,用户使用的是 A4 纸(a4paper),正文字体为 11pt ,接近中文五号字;不可忽略参数告诉 LaTeX ,用户要撰写一篇论文,这样 LeTEX 系统便会为用户准备好论文排版的默认环境。

除了论文类别, LeTEX 还提供了书籍(book),书信(letter),报告(report) 等。排版控制命令\begindocument 与 \enddocument 表示文稿内容的起始与终止. 在 \documentclass 与 \begindocument 之间的区域称为导言区 (,可在此区域内放置一些可影响文档整体排版样式的控制命令。

XETEX/LaTEX 的中文文稿与西文文稿没什么区别,仅仅是文稿内容中使用的是中文, 如下:

```
\begin{document}
世界,你好!
\end{document}
```

3.2 XETEX 中文文档处理

对 XETEX / LaTEX 可使用 xelatex 命令处理生成 pdf 文档:

```
$ xelatex filename.tex
```

现在,假定上一节中组为示例所列举的中文XETEX/LaTEX文稿的文件名为example.tex, 使用 xelatex 命令处理该文稿可以生成 example.pdf 文档,但是使用 pdf 阅读 器打开 example.pdf ,就会发现这是一个空白文档,而没有如我们所预期的那样会在文档 中显示出“世界,你好!”这是因为 XETEX / LaTEX 并没有为中文文稿指定默认字体,这需要我们自行设定。 这也意味着一个很重要的问题: XETEX项目解决了TEX国际化的问题, 而我们要解决 XETEX 本地化问题。但是目前,国内对 XETEX 很了解的人太少了,还未有人提出通用的

XETEX中文解决方案,因此要使用 XETEX 排出符合中文习惯的文章,就需要熟悉一些 XETEX / LaTEX 宏包与排版控制命令。

宏包fontspec可与XETEX/LaTEX 配合使用可实现在XETEX/LaTEX文稿中使用系统自带字体的功能.在 XETEX / LaTEX 文稿中的导言区,使用\usepackage指令可加载指定宏 包.加载 fontspec 宏包后,使用其提供的\setmainfont 命令可设定文稿正文中的中文字体。对上一节中的中文 XETEX/LaTEX 文稿 example.tex 修改如下:

```
\documentclass[11pt,a4paper]{article}
\usepackage{fontspec}
\setmainfont{Adobe Song Std}
\begin{document}
世界,你好!
\end{document}
```

字体设置命令\setmainfont将 Adobe Song Std 指定为文档正文默认字体. Adobe Song Std 是 Adobe 发布 Adobe Reader 8.0 时附带的一款中文宋体,另外还有一款中文 黑体 Adobe HeiTi Std ,它们都是免费字体,可以自由使用.如果你没有装这两款字体, 可以使用 fc-list 命令查看系统已安装的字体名录,如下:

```
$ fc-list :lang=zh-cn
文鼎PL简报宋,AR PL SungtiL GB:style=Regular
文鼎PL中楷Uni,AR PL ZenKai Uni:style=Medium ... ...
```

将 fc-list 输出结果中的字体名填到\setmainfont命令中,即可使得 XETEX / LaTEX 在 系统字体目录下找到相应字体并将其嵌入到所生成的pdf文档中。虽然可以将Windows中文字体挪到 Linux 下使用,但是现在许多自由抑或免费的中文字体已经可以满足中文排版需要了,因此,我们应当尽量不要再去那些私权字体.

现在,使用 xelatex 对修改后的 example.tex 进行处理,可以生成以中文五号宋体显示“世界,你好!”的单页 pdf 文档.

下面,继续进行中文字体的设置,对 example.tex 修改如下:

```
\documentclass[11pt,a4paper]{article}
\usepackage{fontspec}
\setmainfont[BoldFont=Adobe Heiti Std]{Adobe Song Std}
\setsansfont[BoldFont=Adobe Heiti Std]{AR PL KaitiM GB}
\setmonofont{Bitstream Vera Sans Mono}
\begin{document}
世界,你好!
\end{document}
```

在解释修改后的example.tex所发生的变化之前,我们应当了解一下有关字体的一些常识. 西方国家的字母体系可分为两大字族(Font Family): Serif 与 Sans Serif .除此 之外,还有一种打印机字体虽然也是 Sans Serif ,但由于它是等距字,所以又独立出一个Typewriter字族。Serif ,中文常译为“衬线”, Sans Serif 则译为“无衬线”。 衬线字体是源于古代 在一些岩石或金属上刻字时, 雕刻刀在笔画的起落处要有入刀与退刀的讲

究，不然会损伤刻刀。 无衬线字体，是相对于衬线字体而言的。对于西文的衬线字体与无衬线字体的直观意象可见：

N N

中文的字族可分为:隶、楷、行、宋、仿宋、黑、幼圆等,要与西文字族相对应(计算 机是西方文明的产物,汗),那么宋、仿宋都可以看作是衬线字体,而楷体、黑体、幼圆可以看作是非衬线字体.

一旦理解了这些字体常识,对于 example.tex 中新增加的那几条设置中文字体的控制命令应该明白个三五分。 譬如，\setsansfont指令是设定无衬线中文字体的，我们在example.tex中使用该指令将无衬线字体设置为楷体 AR PL KaitiM GB。 \setmainfont是设置衬线字体的,因为 XETEX 将衬线字体视为文档默认字体族,而 XETEX 之所以如此，是因为在实践有一个结论:衬线字体作为文章的正文字体可使读者长时间阅读文章视觉 不疲倦。 非衬线字体在文章中适合作为标题出现,因为它较衬线字体更为醒目,但如果用 无衬线字体作为文章的正文字体,长时间阅读,很容易出现视觉疲劳. fontspec 宏包还提供了一个与\setmainfont 等价的命令 \setromanfont ,这完全是出于历史的缘故，因为 Roman 字体在西方一向被认识是文章正文字体的正统,最有名的是 Times New Roman .

下面讲一下 \setmainfont与\setsansfont指令中的可省略参数BoldFont的用法，这个参数是用来指定衬线与非衬线字体在粗体(bold)状态下所使用的字体,这是因为字体 可以在常态下经“加粗”后所得到的实际上是另一种字体。 对于任意一款计算机字体而言，它不是一个你想怎么变就可以怎么变的东西,如果一款字体在设计的时候就不是粗体,那 么是不可能把它变成粗体的,只有用一种设计好的粗体去替换。虽然有一些办法可以让一 些字体经过微量平移并叠合后可以得到类似“粗体”的效果,但那是“穷人的粗体”,显示 效果很差的。所以,我们不应该把你正在用的这个“宋体”变成“粗”宋体,而必须去找专 门的粗宋体来用。如果找不到粗宋体,那就用黑体来代替,本文的排版就是这么做的,采用 Adobe Heiti Std 来作为衬线与非衬线的粗体. 字体设置完成后,就要考虑中文断行的问题。 还是那句话，XETEX 只致力于解决国际 化问题,并不考虑本地化,如果说考虑了,那也只是默认考虑了西文本地化.西文的断行问 题是根据单词之间的空格来决定一行文本中在哪个单词的尾部断开产生新行的,对中文而 言,这种方法就不适用了,因为中文不是以空格来划分单词的.但不要以为 XETEX 不能很 好的处理中文断行,在 XETEX 内部已有人为中文断行写了一些规则,我们可以直接使用它们,可在 XETEX / LaTEX 的导言区中添加以下指令:

```
\XeTeXlinebreaklocale "zh"
\XeTeXlinebreakskip = 0pt plus 1pt minus 0.1pt
```

上述指令中,\XeTeXlinebreaklocale 指定使用中文断行规则,XeTeXlinebreakskip 可以 让 XETEX 处理中文断行时多一点点自适应调整的空间.

Okay! 事实上,讲到这里,XETEX/LaTEX 的用法基本已讲述完毕，剩下内容就是 TEX / LaTEX 的使用了,它们的许多教程都基本适用 XETEX / LaTEX。

3.3 论文版式

一篇论文应该包括两个层次的含义:内容与表现,前者是指文章作者用来表达自己思想的文字,图片,表格,公式及整个文章的章节段落结构等,后者则是指论文页面大小,边 距,各种字体,字号等.一篇排版良好的论文应当是内容与表现分离的.本节主要介绍如何使用 XETEX / LaTEX 定义论文的表现.

3.3.1 准备纸张

首先准备纸张,在 \documentclass 的可省略参数中, A4 纸用 a4paper 表示, A5 纸用 a5paper ,其他型号用纸的表示类推便是.如果在 \documentclass 指令中未指定纸张型号,则 XETEX / LaTEX 默认用纸是美国信纸(14 × 8.5in).

纸张准备好了,然后就是设置基本字体尺寸.一般而言,中文小四号字用像素点为单 位表示为 12pt ,中文五号字表示为 11pt .基本字体尺寸的设定非常重要,譬如行距,段落缩进,页芯等参数, XETEX / LaTEX 会基于基本字体尺寸给出相应的默认值.基本字体 尺寸也是在\documentclass 指令中作为其可省略参数进行设定的,如果未设定该参数, 则 XETEX / LaTEX 会以 10pt 为默认值.

现在,若要在一张 A4 纸上以 11pt 为基本字体尺寸写一篇论文, \documentclass 指令可写为:

```
\documentclass[a4paper,11pt]{article}
```

XETEX / LaTEX 默认是纵向模式排版,要改为横向排版,可添加 \documentclass 命令的可省略参数 landscape :

```
\documentclass[a4paper,11pt,landscape]{article}
```

\documentclass 还有一些常用的可省略参数,比如 titlepage 可以让文章的标题 单独占据一页, notitlepage 可使标题与文章正文排在同一页面.又比如 draft 可以控制 XETEX / LaTEX 在超出页面宽度限制的文本行右端显示一个粗黑条,提醒用户注意,而 final 的作用恰好相反,无论文本行超出边界多少,也不显示粗黑条,但 XETEX / LaTEX 在 编译 TEX 文档时,会给出警告.

3.3.2 设置页面边距

```
下面谈谈页边距的设置. MS Word 默认的页面边距为:
```

上边距=下边距= 1in (2.54cm) 左边距=右边距= 1.25in (3.17cm)

使用宏包 geometry 可以进行 XETEX / LaTEX 文稿的页面边距设置:

```
\usepackage[top=1in,bottom=1in,left=1.25in,right=1.25in]{geometry}
```

实际上这样设置的页面边距极不美观,尤其是左右对称的页边距没有考虑装订的需要,另外上边距如果加上页眉或就显得过窄.因此,要是真的很注重页面美观的话还是自己去调整一下,比如我喜欢将页边距设置下面这样:

```
\usepackage[top=1.2in,bottom=1.2in,left=1.2in,right=1in]{geometry}
```

将左边距设置的比右边距大一些,主要是考虑装订的需要,但是在实际打印时有单面打印与双面打印模式,在双面打印时,应该是奇数页面的左边距比右边距大一些,在偶数页则相反. XETEX / LaTEX 考虑到了这一点,在偶数页面中会自动将左,右边距切换.指定文稿 单双页面的参数有 oneside 与 twoside ,它们都是 \documentclass的可省略参数,如果 文稿类别是论文,默认是单面打印模式. 如果相对 geometry 宏包的使用进行更详细的了解,请参考文献

3.3.3 章节标题

可使用 titlesec 宏包设置章节标题.在引入 titlesec 宏包时,可以指定一些格式选项, 比如:

```
\usepackage[center,pagestyles]{titlesec}
```

其中 center 可使标题居中,还可设为 raggedleft (居左,默认), raggedright (居右). pagestyles 是申明后面要使用 titlesec 宏包自定义页面样式(在下一节会讲).

标题由标签+标题内容构成,其格式通常在 XETEX / LaTEX 文稿的导言区中设置.要设置论文中的节标题格式,可用 titleformat 指令,用法如下:

```
\titleformat{command}[shape]{format}{label}{sep}{before}[after]
```

其中各参数含义如下:

* command 是要重新定义的各种标题命令,比如\chap, \section,还有更 多的,在后文中讲书籍排版时再谈;
* shape 是用来设定段落形状的,可选的参数有hang , block , display 等,详 见 titlesec 文档,位于: $TEXLIVE$/VERSION/texmf-dist/doc/latex/titlesec
* format 用于定义标题外观,比如使标题居中,字体加粗等;
* label 用于定义定义标题的标签,就是标题内容前面的标号;
* sep 定义标题的标签与标题内容之间的间隔距离;
* before 用于在标题内容前再加些内容;
* after 用于在标题内容后再加些内容;

本文排版所用节标题分为两级,其格式采用以下命令设置:

```
\titleformat{\chap}{\centering\Large\bfseries}{\S\,\thesection}{1em}{}
\titleformat{\section}{\large\bfseries}{\S\,\thesubsection}{1em}{}
```

其中，shape，before，after参数都被省略掉了。format参数将section格式设置为居中（\centering），字号为\Large，字体被加粗显示\bfseries；在设置subsection格式，未采用居中，而是采用默认的居左，另外将标题的字号也降了一级（\large）。label参数将标题的标签设置为以“§”为前缀 + 标题序号。sep参数设置标签与标题内容之间以一个字(1em)的宽度为间隔。

3.3.4 页眉与页脚

这一节讲怎样使用titlesec宏包设置页眉，页脚。下面的命令在XETEX / LaTEX导言区定义了一个新的页面样式，并使用该样式：

```
\newpagestyle{main}{
\sethead{\small\S\,\thesection\quad\chaptitle}{}{$\cdot$~\thepage~$\cdot$}
\setfoot{}{}{}\headrule}
\pagestyle{main}
```

其中\sethead命令设置页眉，用法为：

```
\sethead[偶数页左页眉][偶数页中页眉][偶数页右页眉] {奇数页左页眉}{奇数页中页眉}{奇数页右页眉}
```

单面打印模式只要给出奇数页的设置即可，双面模式则需要将左，右页眉做个调换。上面给出的例子是单面模式的。\setfoot指令用法与\sethead用法相似。上面的页眉页脚设置示例中，\headrule指令可画页眉线，默认宽度是0.4pt，如果对该宽度不满意，可使用下面命令重新设置其宽度：

```
\setheadrule{宽度值}
```

上面的页眉设置示例的排版效果即本文档页眉效果。

3.4 如何安装字体

我用的试验环境是Ubuntu Oneiric（11.10），大部分可以直接从Ubuntu源中下载了。

你可以用命令fc-list :lang=zh-cn查看安装好的中文字体，结果中前半部分就是字体名称（如AR PL UMing CN）。

```
user@puppet1:~$ fc-list :lang=zh-cn | grep CN
AR PL UMing CN:style=Light
AR PL UKai CN:style=Book
```

文鼎开放的四套字体的Ubuntu包、字体名字和名称如下：

```
ttf-arphic-gbsn00lp      "AR PL SungtiL GB" 文鼎PL简报宋
ttf-arphic-gkai00mp      "AR PL KaitiM GB" 文鼎PL简中楷
ttf-arphic-ukai          "AR PL UKai" 文鼎PL中楷
ttf-arphic-uming         "AR PL UMing" 文鼎PL细上海宋
```

文泉驿字体的Ubuntu包、字体名字和名称如下

```
ttf-wqy-microhei   "WenQuanYi Micro Hei" 文泉驿的微米黑
ttf-wqy-zenhei     "WenQuanYi Zen Hei" 文泉驿的正黑
xfonts-wqy         "WenQuanYi Bitmap Song" 文泉驿的点阵宋体
```

Adobe的中文字体有官方下载

```
$ tar -jzxf FontPack910_chs_i486-linux.tar.bz2
$ tar -xvf CHSKIT/LANGCHS.TAR
$ mkdir ~/.fonts
$ cp Adobe/Reader9/Resource/CIDFont/*.otf ~/.fonts
$ fc-cache -f -v
$ fc-list :lang=zh | grep Adobe
```

3.5 蛋疼的问题

只可惜现在正文在产生PDF时没有一种字体是有完美表现的。

* 文鼎贡献的字体中台湾字形的细上海宋的句号在中间，出来的效果不伦不类的。
* 文鼎贡献的字体中大陆字形的简中楷和简报宋，标点符号的位置是对的，但是当碰到条目（Item）的时候条目的点没能显示出来。
* Adobe的宋体，条目的时候显示一个田子框，很难看。
* 文泉驿的点阵宋体老是转化Latex时出错，搞不定。

2、3 条目的问题，我hack成其他字符（*）显示就没问题了（如下），不知道缺省的圆点为啥显示不对。http://wiki.ctex.org/index.php/LaTeX/%E5%88%97%E8%A1%A8，现在就用文鼎的细上海宋了。

```
\begin{itemize} \setlength{\itemsep} {1pt} \setlength{\parskip} {0pt} \setlength{\parsep} {0pt}
\setlength{\itemsep}{1pt}\setlength{\parskip}{0pt}\setlength{\parsep}{0pt}
\item[*]
% 原来是
% \item
```

3.6 参考

* http://share.chinatex.org/

* http://latex.yo2.cn/articles/latex-introduction0.html
* LATEX2e完全学习手册：http://product.china-pub.com/54569
* XeTeX：http://scripts.sil.org/xetex

第 4 章

基础知识和10分钟写出第一本开源书

4.1 先从Pro Git说起

如果你了解Git，或者想了解Git。那么你就应该知道Pro Git，它是Git的书中写得最好的一本（至少是之一），可是你是否知道它有网络中文版，而且能在iPad上极其漂亮得阅读。并且是免费的，不是盗版的免费！如果你想要最新的，你甚至可以自己生成它。哈哈，我就是这么干的。

这一切就归功于开源社区和它后面用到的技术。

4.1.1 开源书

这里我不用多讲，开源书就像其他的开源产品（如维基百科）一样，只要是开放的，社区就有人会贡献。Pro Git的作者Scott很慷慨得把书的内容全部共享在github/progit库中，使用得是CC BY-NC-SA 3.0。

Scott只负责英文版，其他许许多多语言的翻译都是社区贡献的，中文翻译相当有质量，你可以在线读Pro Git中文版。

4.1.2 开源技术生成电子书

这本书不仅仅开源了内容，使用的技术也是开源的。让我们看看他是怎么做的。

4.1.3 Markdown原始文件

首先书的内容是用Markdown格式写的。Markdown格式的普及要归功于Github和StackOverflow。因为它们越来越流行，它们支持Markdown格式也越来越流行。这里要赞一个的是，国内的图灵社区也支持Markdown，用起来超级方便。

简单来说，Markdown格式的文件看着像一般的文本文件，里面只是加了很少的格式标记，因此看文本文件也不影响理解，这种格式也有很多工具帮你去转化，而且很容易自动化解决。并且这些技术大多数是开源或免费的。

松本行弘在他的Ruby书中说的好，想象一下几十年后，你是否还能找到软件来打开你的Word老格式的文档，没有那些软件，你的文档也就没用了。文本文件就没有这个问题。

你可以直接看一下【Pro Git】的“第一章 介绍” 的Markdown原始文件，顺便看看github自动生成的简单“第一章 介绍” 的html。

4.2 产生电子书

4.2.1 PDF格式

为了能达到出版的质量，Latex是一个常用的格式，PDF也能很容易地转换出来，有关Latex，自己看看参考链接学习吧。

Pandoc能帮着从Markdown转换出Latex格式，然后再用TexLive软件中的xelatex转成PDF格式。

4.2.2 Epub/Mobi格式

Ruby的rdiscount能帮你从markdown转成html格式，然后有Calibre附带的命令ebook-convert生成最终的.mobi（Kindle）和 .epub（iPad）。

从1.8版本开始，Pandoc也开始支持生成Epub格式了。

4.3 工作环境

你只需要一台Linux机器（虚拟机就可以了）和熟悉简单的Linux命令就可以试验了。有Git和Ruby的知识那就更方便了。

我用的试验环境是Ubuntu Oneiric (11.10)

4.3.1 下载中文开源书

很简单，git clone一下这本书就可以了，下载它的源文件包我觉得还是烦了点。

```
$ git clone git@github.com/larrycai/kaiyuanbook.git
```

4.3.2 PDF格式

生成PDF是一个比较复杂的东西，pandoc用Ubuntu库里1.8.x版本，TexLive用缺省Ubuntu源里的2009版也够了。当然也可下载最新的TexLive包安装，并配置到搜索路径中。

```
$ sudo apt-get install ruby1.9.1
$ sudo apt-get install pandoc
$ sudo apt-get install texlive-xetex
$ sudo apt-get install texlive-latex-recommended # 主要的Latex包
$ sudo apt-get install texlive-latex-extra # titlesec包，先不用知道
```

因为是中文PDF，需要把字体嵌入在文件中，因此需要安装字体文件，幸运的是在源里有不错的字体。

```
$ sudo apt-get install ttf-arphic-gbsn00lp ttf-arphic-ukai # 文鼎字体
$ sudo apt-get install ttf-wqy-microhei ttf-wqy-zenhei # 文泉驿字体
```

现在你就可以用**mkbok**命令生成Pdf文件了，**mkbok**会自动调用Pandoc和Latex工具生成Pdf、Html、Epub格式。

```
$ ./mkbok
```

怎么样，打开看看Pdf文件，很漂亮了吧。

4.4 自己试试

一定要做，搜索一下，改掉一些内容，再运行一遍。

好了，你可以把书扔到一遍，写你自己的书了。照样画葫芦，你行的。在下一章会对书的结构和怎么对应地用Markdown写进行详细地解释。

4.5 其他常用的格式

计算机类图书对格式要求不是很多，图文、章节、源代码基本就够了，就算有些复杂公式，也可用图来显示。这也从理论上说明，它不需要复杂的格式。现在对这类技术书出版我的理解主要有几种:

* Microsoft的Word格式，虽然国内出版界如日中天，缺省就认它（对技术没追求，鄙视）。简单好学，但是不擅长自动化，是开源的死敌。
* Latex格式（就是Donald E. Knuth（高德纳）发明的，这是很棒的东西，特别适合学术类的各种复杂的公式等，不过学习曲线很高，直接写还是很有难度的。国内也只有几家学术期刊使用。
* Docbook格式是最有名的（从SGML演化过来？），Oriel1y和Pragmatic出版社缺省就用它，它能 很方便的转化出出版要的各种样式。如Jenkins – the definition guide开源书就是采用Docbook。但由于是XML格式，很多人不习惯，而且多人网上协作不是很方便。
* 通过蒋鑫的Got Github开源书，我也了解reStructureText也是和Markdown差不多纯文本的，也是蛮流行的。

4.6 参考

* Pro Git: http://progit.org/
* LaTeX2e完全学习手册: http://book.douban.com/subject/5450816/