嵌入式Linux webserver: Boa+CGI程序设计技术

摘要：在详细介绍一种嵌入式 Web 服务器 BOA 的实现与配置方法的基础上，以一个 Web 在线远程监控 GPIO（通用输入/输出）的程序为实例，介绍嵌入式 Linux 系统下 CPU 程序设计技术。

关键词：嵌入式系统 Linux BOA CGI GPIO

1 概述

随着互联网应用的普及，越来越多的信息化产品需要接入互联网通过 Web 页面进行远程访问。嵌入式 Web 系统提供了一种经济、实用的互联网嵌入式接入方案。这里结合一种嵌入式 Web Server BOA 来介绍嵌入式 Linux 系统下的 CGI 程序设计技术。

2 Web Server BOA 的实现与配置

2.1 uClinux 下，主要有三个 Web Server:HTTPD、THTTPD 和 BOA。HTTPD 是最简单的一个 Web Server,它的功能最弱，不支持认证，不支持 CGI。THTTPD 和 BOA 都支持认证、CGI 等，功能都比较全。BOA 是一个单任务的小型 HTTP 服务器，源代码开放、性能优秀，特别适合应用在嵌入式系统中。目前的 uClinux 的代码中已经包含 BOA 的源代码。在 uClinux 下实现 BOA，只需要对 BOA 做一些配置和修改。以下是配置的过程。

（1）编译 BOA 到内核

首先，需要把 BOA 编译到内核，即执行 make menuconfig，在应用程序选单中 network application 项下面选择 boa。该操作需要重新编译内核。

（2）编制配置文件 boa.conf

在 Linux 操作系统下，应用程序的配置都是以配置文件的形式提供的，一般都是放在目标板/etc/目录下或者/etc/config 目录下。但 boa 的配置文件 boa.conf 一般都旋转在目标板/home/httpd/目录下。

例如，一个典型的 boa.conf 文件格式如下：

ServerName Samsung-ARM

DocumentRoot/home/httpd

ScriptAlias/cgi-bin/home/httpd/cgi-bin/

ScriptAlias/index.html/home/httpd/index.html

它指定了 HTML 页面必须放到/home/httpd 目录下，cgi 外部扩展程序必须放到/home/httpd/cgi-bin 目录下。

（3）编译烧写内核

重新编译内核后，通过烧写工具烧写内核，就可以在 PC 上通过 IE 浏览器访问开发板上的 Web Server。例如，输入开发板的 IP 地址 http://192.168.0.101，即可访问到自己做的网页 index.html 了。并且，通过编写 CGI 外部扩展程序，可以实现动态 Web 技术，下面将详细介绍。

2．2 具有 MMU 平台的 Linux 下 B0A 的实现与配置

对于有 MMU（内存管理单元）的平台，如 armlinux 和 ppclinux，可以到网上下载一个主流版本的 boa 发行包。因为是运行在目标系统，所以要用交叉编译工具编译，即需要修改 boa/src/Makefile 里面的编译器。例如：

CC=/LinuxPPC/CDK/bin/powerpc-linux-gcc

CPP=/LinuxPPC/CDK/bin/powerpc-linux-g++

然后直接在 boa/src 目录下执行 make，即可生成 BOA 可执行文件；将其编译入内核，并烧写到存储设备，就可以实现访问 BOA 服务器。

3 CGI 程序设计技术

CGI（Common Gateway Interface）是外部应用扩展应用程序与 WWW 服务器交互的一个标准接口。按照 CGI 标准编写的外部扩展应用程序可以处理客户端浏览器输入的数据，从而完成客户端与服务器的交互操作。而 CGI 规范就定义了 Web 服务器如何向扩展应用程序发送消息，在收到扩展应用程序的信息后又如何进行处理等内容。通过 CGI 可以提供许多静态的 HTML 网页无法实现的功能，比如搜索引擎、基于 Web 的数据库访问等等。

3．1 工作原理

（1）WWW 和 CGI 的工作原理

HTTP 协议是 WWW 的基础，它基于客户/服务器模型，一个服务器可以为分布在网络中处的客户提供服务；它是建立在 TCP/IP 协议之上的“无连接”协议，每次连接只处理一个请求。在服务器上，运行产着一个守护进程对端口进行监听，等待来自客户的请求。当一个请求到来时，将创建一个子进程为用户的连接服务。根据请求的不同，服务器返回 HTML 文件或者通过 CGI 调用外部应用程序，返回处理结果。服务器通过 CGI 与外部程序和脚本之间进行交互，根据客户端在进行请求时所采取的方法，服务器会收集客户所提供的信息，并将该部分信息发送给指定的 CGI 扩展程序。CGI 扩展程序进行信息处理并将结果返回服务器，然后服务器对信息进行分析，并将结果发送回客户端。

外部 CGI 程序与 WWW 服务器进行通信、传递有关参数和处理结果是通过环境变量、命令行参数和标准输入来进行的。服务器提供了客户端（浏览器）与 CGI 扩展程序之间的信息交换的通道。CGI 的标准输入是服务器的标准输出，而 CGI 的标准输出是服务器的标准输入。客户的请求通过服务器的标准输出传送给 CGI 的标准输入，CGI 对信息进行处理后，将结果发送到它的标准输入，然后由服务器将处理结果发送给客户端。

（2）URL 编码

客户端浏览器向服务器发送数据采用编码的形式进行。该编码就是 CRL 编码。编码的主要工作是表单域的名字和值的转义，具体的做法为：每一对域和值里的空格都会被替换为一个加号（+）字符，不是字母或数字的字符将被替换为它们的十六进制数字形式，格式为%HH。HH 是该字符的 ASCII 十六进制值。
标签将被替换为“%0D%0A”。

信息是按它们在表单里出现的顺序排列的。数据域的名字和数据域的值通过等号（=）字符连在一起。各对名/值再通过“&”字符连接在一起。经过这些编码处理之后，表单信号就整个成为一个连续的字符流，里面包含着将被送往服务器的全部信息。

因为表单输入信息都是经过编码后传递给脚本程序的，所以 CGI 扩展程序在使用这些参数之前必须对它们进行解码。

3．2 CGI 外部扩展程序编制

服务器程序可以通过三种途径接收信息：环境变量、命令行和标准输入。具体使用哪一种方法要由<FORM>标签的 METHOD 属性来决定。

在“METHOD=GET”时，向 CGI 程序传递表单编码信息的正常做法是通过命令来进行的。大多数表单编码信息都是通过 QUERY_STRING 的环境变量来传递的。如果“METHOD=POST”，表单信息将通过标准输入来读取。还有一种不使用表单就可以向 CGI 传送信息的方法，那就是把信息直接追回在 URL 地址后面，信息和 URL 之间用问号（？）来分隔。

下面结合 Web 远程监控 ARM 芯片的 GPIO（通用输入/输出）的应用实例详细介绍。

（1）GET 方法

GET 方法是对数据的一个请求，被用于获得静态文档。当使用 GET 方法时，CGI 程序将会从环境变量 QUERY_STRING 获取数据。为了处理客户端的请求，CGI 必须对 QUERY_STRING 中的字符串进行分析。当需要从服务器获取数据并且不改变服务器上的数据时，应该选用 GET 方法；但是如果请求中包含的字符串超过了一定长度，一般是 1024 字节，那么就只能选用 POST 方法。GET 方法通过附加在 URL 后面的参数发送请求信息。这些参数将被放在环境变量 Q

UERY_STRING 中传给 CGI 程序。GET 方法的表单格式和 CGI 解码程序可以参考 POST 方法的实现。

（2）POST 方法

当浏览器将数据从一个填写的表单传给服务器时一般采用 POST 方法，而且在发送的数据超过 1024 字节时也必须采用 POST 方法。当使用 POST 方法时，Web 服务器向 CGI 程序的标准输入 STDIN 传送数据。发送的数据长度存在环境变量 CONTENT_LENGTH 中，并且，POST 方法的数据格式为：

variable1=value1&variable2=value2&etc

CGI 程序必须检查 REQUEST_METHOD 环境变量以确定是否采用了 POST 方法，并决定是否要读取 STDIN。POST 方法在 HTML 文档中定义的表单如下：

```
<FORM METHOD=POST ACTION="/cgi-bin/cgi_gpio.cgi">

<INPUT TYPE="RADIO"NAME=rb VALUE="0">Operate P0<BR>

<INPUT TYPE="RADIO"NAME=rb VALUE="1">Operate P1<BR>

<INPUT TYPE="RADIO"NAME=rb VALUE="2">Operate P2<BR>

<INPUT NAME="ok"TYPE=submit VALUE="OK"><INPUT>

NAME="cancel"TYPE=reset VALUE="RESET"></FORM>
```

它调用的服务器脚本程序是/cgi/bin/cgi_gpio.cgi。CGI 扩展程序中 FORM 表单的解码可参考如下程序：

```
/*function getPOSTvars*/

char **getPOSTvars(){

int i;

int content_length;

char **postvars;

char *postinput;

char **pairlist;

int paircount=0;
```

```
chr  *nvpair;
char  *eqpos;
postinput=getenv("CONTENT_LENGTH");//获取传送给程序数据的字节数
if(!postinput)
exit();
if(!content_length=atoi(postinput)))  //获取信息长度
exit(1);
if(!(postinput=(char*)malloc(content_length+1)))
exit(1);
if(!fread(postinput,content_length,1,stadin))
exit(1);
postinput[content_length]='0';
for(i=0;postinput[i];i++)
if(postinput[i]=='+')
postinput[i]='';  //对加易进行处理
pairlist=(char  **)malloc(256*sizeof(char  **));
paircount=0;
nvpair=strtok(postinput,"&");//从出现“&”字符的位置把信息分段，然后对结果依次处理
while  (nvpair){
pairlist[paircount++]=strdup(nvpair);
if(!(paircount%256))
pairlist=(char**)realloc(pairlist,(paircount+256)*sizeof(char**));
nvpair=strtok(NULL,"&");
```

```
}

pairlist[paircount]=0;

postvars=(char**)malloc((paircount*2+1)*sizeof(char **));

for(i=0;i<paircount;i++){

if(eqpos=strchr(pairlist[i],'=')){

*eqpos='0';

unescape_url(postvars[i*2+1]=strdup(eqpos+1));//调用 unescape_url 函数继续解码

}else{

unescape_url(postvars[i*2+1])=strdup(""));

}

postvars[paircount*2]=0;

for(i=0;pairlist[i];i++)

free(pairlist[i]);

free(pairlist);

free(postinput);

return postvars;

}
```

其中，unescape_url 函数再调用 x2c 函数，把（不是字节或数字的）特殊字符从其%HH 表示方式解码为文本字符。

```
/*unescape_url function*/

static void unescape_url(char *url){

int x,y;

for(x=0,y=0;url[y];++x,++y){

if((url[x]=url[y])=='%'){
```

```
url[x]=x2c(&url[y+1]);

y+=2;

}

}

url[x]='0';

}
```

(3)直接 URL 加参数传递方法

这是一种不使用表单就可以向 CGI 传送信息的方法。它把信息直接追加在 URL 地址后面，信息和 URL 之间用号号（？）来分隔。例如，对于一个 cgi_gpio.cgi 的脚本，可以从如下的链接启动：

```
<A  HREF=/cgi-gpio.cgi!?flag=0  Operate  P0</A>

<A  HREF>/*cgi-bin/cgi_gpio.cgi?flag=1  Operate  P1</A>

<A  HREF=/cgi-bin_gpio.cgi?flag=2  Operate  P2</A>

.

.

.
```

CGI 扩展程序中可使用如下代码接收信息：char *get_input;//用于接收环境变量

```
.

.

.

get_input=getenv(“QUERY_STRING”);

if(get_input){

get_input=strdup(get_input);

printf("QUERY_STRING  if  %s",get_input);
```

```
}

/*判断 flag=x 信息*/

if(!strcmp(get_input,"flag=0")

...//Operate  p0

else  if(!strcmp(get_input,"flag=1")

...//Operate  P1

else

...//Operate  P2
```

对于上述三种方法，可以根据不同的应用场合和应用要求进行选取。

结语

嵌入式 Web Server 系统方案可以广泛应用在许多领域，如自动化设备的远程监控、嵌入式 GSM 短消息 平台以及远程家庭医疗等。并且，随着互联网应用领域的不断深入，嵌入式 Internet 技术将得到更为广泛的应用和发展。

下面附的是一段 csp 写的 cgi程序的源程序. 把 c 嵌入到 html 中写cgi 是不是容易多了. 更多信息请访问, eybuild 的官方网站: http://www.eybuild.com

来自:
http://www.eybuild.com/develop/demoshow.htm

```
<html>
<% {{
 char * title = "关于 eybuild 综合演示程序";
%>
<head>
    <meta http-equiv="content-type" content="text/html; charset=gb2312">
 <title><% =title %></title>
</head>
<body>
 来自: http://www.eybuild.com<br>
 <font color=red><h3><% =title %></h3></font>
 <br>
```

```
本程序包用于演示 eybuild/csp 生成的 cgi 程序的执行.<br>
<p>
============<br>
<font color=red size=3>示例说明:</font><br>
============<br>
<% {
char    pwd[256] = "";

_getcwd(pwd, sizeof(pwd));
%>
<table>
<tr><td width=250><a href=/cgi-bin/demo.cgi target=__blank><% =pwd
%>\demo.cgi</a>
 <td>一个最小的交互程序测试程序
<tr><td><a href=/cgi-bin/review.cgi target=__blank><% =pwd %>\review.
cgi</a>
 <td>一个简单的留言簿程序
<tr><td><a href=/cgi-bin/rweb.cgi target=__blank><% =pwd %>\rweb.cgi
</a>
 <td>远程web文件管理器, 可以自由浏览下载/上传文件
<tr><td><a href=/cgi-bin/fr40.cgi target=__blank><% =pwd %>\fr40.cgi</a
>
 <td>是一个路由器的管理界面, 嵌入式应用的一个示例
</table>
<p>
<% } %>
============<br>
<font color=red size=3>运行步骤:</font><br>
============<br>
1. 解压到任意目录<br>
2. 运行web 服务器 webs.exe <br>
3. 从浏览器上输入或直接点击下面的cgi的地址:<br>
<% {
@include <undef.h>
@include <windows.h>
@include <direct.h>
@include <ebdef.h>
@include <ebio.h>

    win32_find_data wfd;
    handle   hfind = findfirstfile("*.cgi", &wfd);

 if (invalid_handle_value != hfind) {
  do {
```

```
   if (!(wfd.dwfileattributes & file_attribute_directory))
   {
 %>
      
  <a href="/<";% =wfd.cfilename%> target=__blank>http://127.0.0.1/cgi-bin/
<% =wfd.cfilename%></a><br>
 <%  }
  } while (findnextfile(hfind, &wfd));
 } /* if */
}
}}
%>
<p>
============<br>
<font color=red size=3>关于web服务器:</font><br>
============<br>
<font color=red>如果你已经安装了 web 服务器(如apache/iis等), <br>
可以将 cgi-bin 目录中文件, 拷贝到相应的cgi目录中运行cgi演示程序</font>
<p>

为了便于演示 csp cgi 程序, eybuild group 修改了开源的 goahead web
服务器(windows 版)<br>
打开 web 服务器后, 直接从浏览器上输入同目录下 cgi-bin 目录中的文件名,
<br>
即可打开示例程序.<p>
<body>
</html>
```

个人主页 | 引用 | 返回 | 删除 | 回复

回复:嵌入式 Linux webserver: Boa+CGI 程序设

anew(游客)发表评论于 2006-3-30 16:08:00

用 C 写 CGI 就要用 CSP(将 C 直接嵌入到 HTML 中), 特点是嵌入式应用了, CSP/eybuild
是为嵌入式领域 WEB 开发量身定做的的一套开发工具.
它支持 apache, iis, boa, mini-httpd, thttpd, goAhead 等 WEB 服务器和各种嵌入式操作系
统.

下面的内容摘自 "eybuild 中文手册"

来自:
http://www.eybuild.com/develop/doc/manual/eybuild_manual_ch.htm
http://www.eybuild.com/develop/doc/manual/eybuild_manual_ch.pdf
http://www.eybuild.com

第 1 章 序言

VB/JAVA/PHP等脚本直接嵌入在HTML中叫 ASP/JSP/PHP，那么用 C 直接嵌入在HTML中叫 CSP 吗?

是的，现在我们可以直接将 C 语句嵌入在HTML中并叫它 CSP。C 语言天然好的"移植性/高效性/灵活性"，一直以来都是最受程序员青睐的语言。现在用CSP 技术我们就可以轻松地将 C 语句直接嵌入到 HTML 源文件来快速编写CGI程序。

CSP VS cgilib:

传统的cgilib的直接使用标准函数printf等语句输出HTML代码。不但使得C程序和HTML程序交织的混乱不堪，还使得页面输出的流程控制变得异常复杂。现在ASP/JPS/PHP等几乎完全取代用cgilib。CSP与cgilib的开发模式不同，它充分吸取了ASP/JSP/PHP等以HTML/XML为模板嵌入脚本等诸多优点，并充分融合C语言的语言特性。使得CSP的开发变得快速、高效，并大大提了最终代码的可读性和维护性。CSP及其开发环境eyBuild是cgilib的继承和发展，同时目前也是开发高效率WEB应用的最佳选择。

一般工作步骤:

编辑好的 CSP 源程序，用eyBuild开发包提供的 CSP2BIN 工具将 CSP 源文件生成 C 程序的源文件，再链接上 eyBuild 提供的高效 CGI 运行库，就可以在各种平台生成移植性非常高的 CGI 程序。

To ASP/JSP/PHP 的程序员:

编写 CSP 程序就跟编写 ASP/JSP/PHP 一样，可以以先编写 HTML 文件为模板， 再在其中插入CSP 的语句。 甚至有些时候，就可以直接拿 ASP/JSP/PHP 的源文件稍加修改后作为 CSP 的源文件了，因为它们都用类似 <% 和 %> 的标签进行标记的嘛。 如果你是 ASP/JSP/PHP的程序员，并熟悉 C语言，半天时间你就能把 CSP 全学会。

To 嵌入式WEB开发:

CSP 设计的最原始的初衷，就是要为嵌入式开发定制的一套类似 ASP/JSP/PHP的C语言开发工具。 因为嵌入式设备(如路由器/交换机/VoIP网关PBX等)上用的开发语言主要是 C，而传统的 CGI 库 cgilib 以及开发模式远远不能跟上现代的开发需求。

现在 CSP 的eybuild开发环境提供的PC和嵌入式设备上高效移植的开发库，让服务器上应用

和嵌入开发进行了有效统一，使得两者上的开发变得更为容易。 同时，优秀的跨平台的移植性也是eybuild的重要特性。

实践证明，CSP 及其开发工具 eybuild是嵌入式设备WEB开发的最理想工具，它能大大节缩短发周期(一般提高 4-6 倍)，提高最终代码的可读性、可维护性(HTML 和 C 代码进行了有效的分离，所以代码维护更容易)。

高效的页面/图片/CSS集成技术:

通过eyBuild提供的集成技术，你可以把许多CSP/HTML页面集成生成到一个CGI中(包括页面相关的图片，CSS 文件及其它静态文件)。 甚至，你可以将一个小型的网站或WEB应用生成到一个CGI文件中，这使得最终的可执行脚本文件管理变得异常简单。 这一点在嵌入式设备上特别有用，因为它们中的很多只有有限的外存储器(如Flash ROM)和文件系统。 eyBuild为最后生成的 CGI 程序在其内部建立了虚拟目录，使得页面间的链接和引用跟一般HTML的编写方法一样，非常方便建立和维护。同时对服务器级应用，这也将是一个非常有利的选择。

可以直接调用任意 C 的函数

在 CSP 源程序中还可以非常容易地包含C 程序的头文件，这样在 HTML 代码中你就可以像写编写 C 文件一样调用外部函数或系统函数了，跟直接编辑 C 程序几乎没有差别。

CSP 的宏指令指示符 @

用宏指令指示符不仅可以进行包含 C 程序的头文件，还可以包含其它CSP文件。 这样当许多页面需要引用共通的一部分时(如页头/页脚或其它部分)，包含其它 CSP 源文件这个功能显示特别有用。

有效的页面输出缓冲控制

跟 ASP/JSP/PHP一样，通过宏指令指示符还可以有效控制页面输出时的 MIME 头，页面缓冲区大小等等。这种使得页面上的流程控制变得更简单更直观。

Linux+boa+eybuild 如何调试 cgi 程序

- **核心提示：**编译没有错误，把在 eybuild 环境下生成的 cgi 拷贝到 cgi-bin 目录下以后执行结果有问题，如何进行单步调试呢？ 有几种单步调试 CGI 程序方法，可根据需要灵活选择下: 1. Windows,VC++下 异常中断调试 CGI 程序. http://www.eybuild.com/develop/d ... h.htm#_Toc133743402.....

编译没有错误，把在 eybuild 环境下生成的 cgi 拷贝到 cgi-bin 目录下以后执行结果有问题，如何进行单步调试呢？

有几种单步调试 CGI 程序方法，可根据需要灵活选择下:

1. Windows，VC++ 下 "异常中断调试" CGI 程序.
   http://www.eybuild.com/develop/d ... h.htm#_Toc133743402
   下推荐这种方法，适用于所有 CGI 程序的调试.

   如果你的程序全采用的标准 C(ANSI C) 那么可以把 linux 程序，在 windows 上调试好再到 linux 下用.

2. Linux 环境下，手工在 shell 中设置好环境变量(如 QUERY_METHOD=GET, QUERY_STRING="cgi=test.csp&xx=yyy")
   再用 gdb 运行 cgi 程序，单步跟踪调试 cgi.

3. Linux XWindow 使用 Debugger，设置好环境变量，单步跟踪调试 cgi.
   Debugger 很类似 VC++

4. eybuild 还可以在 cgimain() 函数中用 下面的语句模拟码 WebServer 对 CGI 的输入,

```
/* set debug environment */
ebSetDebug
   (
   "GET",                        /* 请求方法, GET/POST */
   "cgi=/demo.csp&xxx=yyy"              /* 模拟查询 QUERY_STRING */
   );
```

5. 日志调试，详细记录出错的位置和原因，在 eybuild 0.9.0 版以后增加的该功能(目前还没有对外发布版)

---

boa 默认的主页是 index.html ,我在 index.html 中的表单中调用 eybuile 生成的 CGI 文件,怎么显实不出来???????

H.264 详解-为什么叫 H.264

sockit 发表于 2006-8-21 20:34:00

为什么叫 H.264

H.264 是一种视频高压缩技术，全称是 MPEG-4 AVC，用中文说是“活动图像专家组-4 的高等视频编码”，或称为 MPEG-4 Part10。它是由国际电信标准化部门 ITU-T 和规定 MPEG 的国际标准化组织 ISO/国际电工协会 IEC 共同制订的一种活动图像编码方式的国际标准格式，这是我们叫惯了的 MPEG 中的一种，那为什么叫 H.264 呢？

原来国际电信标准化部门从 1998 年就 H.26L 的 H.26S 两个分组，前者研制节目时间较长的高压缩编码技术，后者则指短节目标准制订部门。H.26S 的标准化技术的名称为 H.263，听起来很耳生，但实质上却早在用了，还被骂得很激烈。因为，H.263 先入为大，一直以 MPEG-4 大内涵的名字在用。 H.263 的全称为 MPEG-4 Visual 或 MPEG-4 Pall Ⅱ，即 MPEG-4 视频简单层面的基础编码方式。2001 年后，国际电信标准化部门 ITU-T 和 MPEG 的上级组织国际标准化组织 ISO/国际电气标准会议 IEC 成立了联合视频组 JVT，在 H.26L 基础进行 H.264 的标准化。

2002 年 12 月 9 日~13 日，在***香川县淡路岛举行的 MPEG 聚会上确定了相关技术的规格。规格书定稿后，2003 年 3 月 17 日，H.364 的技术格式最终稿国际标准规格（FDIS）被确立。目前软件和 LSI 芯片，服务及设备也都进入了使用阶段。格式书中，列出了比特流规定，解码必要格式，和可供参考的编码记载。

为了不引起误解，ITU-T 推荐使用 H.264 作为这一标准的正式名称。实际上，MPEG-4 里还有 MPEG-4 Audio 和 MPEG-4 System 的不同规格。

MPEG-4 挨骂是因为MPEG-4 Visual许可收费离谱引起的。别以为有了专利就可以随意向人要钱了，专利的最终目的的是使全社会的智力资料更合理地使用，防止重复劳动，并不是犒赏最先发明者。按唯美史观，当社会技术发展到某一阶段时，新技术必然会出现。不是你、就是他总会发明出来，只是细节、时间、成本上的微小差别。历史上，这样不约而同的发明很多，无线电的发明者是马可尼还是波波夫，一直在西方和东方技术史界争论。

而当专利技术成为国际标准的一部份后，问题就更加复杂了。国标标准是强制的，向其中的专利付费是否有垄断之嫌？标准中的技术专利请求，是否合理？如何区分正当的请求和不正当的请求？等等一系列的理论、法律和道德问题都出来了。要尊重专利法，也要遵守反垄断法。这两年国际上围绕 MPEG-4 收费问题的大争论就是由此而起。
在标准化进程中，专利的争端正在增加，任何黑白两极的判断都无法令人满意。但奇怪的是标准中的专利争端发展到要求判决的案例几乎没有，都是当事者幕后交易解决，这使得不明确的法理更陷入恶性循环之中。同时也助长了用户对盗版的宽容，一边是抢我的剪径强资，另一边是偷你的小贼，怎么讲道德？！

MPEG-4 的收费问题主要是从向传输环节收费引起的。MPEG-4 对解码器和编码器的收费已经比 MPEG-2 低了很多，这是各种压缩技术竞争的结果。但 MPEG-2 不对传输 MPEG-2 压缩图像的服务环节收费，而 MPEG-4 则要对内容配送者收取每分钟 0.0333 美分的许可费。钱数听起来不大，但伦理上却有很大的差别。打个比方，你买了台彩电，必要的专利费用已经通过彩电厂转交到专利技术持有者的手中。而当你打的把这台彩电运回家的时候，出租车主也要向专利持有者交费！能不引起轩然大波吗。

现在的专利收费结构已经相当商业化。一种产品、一个系统或一套技术标准中，包含有许许多多公司的专利技术，使用企业很难与一个个技术的发明者直接交涉签约，这样就出现了一种专利管理公司的企业。它把某一产品的一个个技术从专利持有者手中买下来，约定好收益的分配方案，再由它人使用技术的企业中收取许可费。需要用这一产品技术的企业就只需与专利管理公司打交道，操作方便多了。但专利管理公司和著作权保护

企业一样，实际上是一个中间商，两头赚钱，未必把社会效益放在最高地位。

现在的 MPEG-4，也即 MPEG-4 Visual 是由美国 MPEG LA 公司进行专利许可管理的，他同时也在管理 MPEG-2 的专利，目前还在争取 H.264 的专利许可权。MPEG LA 公司于 2002 年 9 月就开始募集 H.264 的主要专利，想采取先入为主的手段取得管理权。由于大量企业对 MPEG-4 收费制度不满，2003 年 6 月， MPEG-4 的支持团体 M4IF（MPEG-4 工业论坛），决定数据流标准格式的美国 ISMA（国际数据流媒体协会）和多媒体通信有关业界团体 IMTC （国际多媒体通信协会）发起召开 H.264 的许可制度说明会。总共有专利持有者和使用者团队 45 个，56 人参加，对有关 H.264 许可问题进行早期意见交换，希望协调各方面的要求和利益。关于方面其它信息，我们稍后再细述，先看看 H.264 的特色吧。

H.264 用大运算量来换取高压缩率、高画质

H.264 受人追捧有三大原因：高性能、国际标准和公正的无差别许可制度。

首先是超高压缩率，其压缩率为 MPEG-2 的 2 倍以上，MPEG-4 的 1.5 至 2 倍。这样的高压缩率是以编码的大运算量来换取的，H.264 的编码处理计算量有 MPEG-2 的十多倍。不过其解码的运算量并没有上升很多，故对用户接收播放来说没有什么难度。

从另一角度，编码的大运算量现在也不是什么大问题。MPEG2 是 1994 年推出的，当时微处理器的工作频率才 100MHz，主存储器容量也不满 10MB。 MPEG-2 那样的压缩运算适应了当时的技术水平。而现在 CPU 的工作频率可上升到 3GMz，DRAM 用到 256MB，提升了 30 倍上下，运算量也不怕。实验表明在奔腾 4 处理器的 3GHz 电脑上，可用软件实现 D1（720×80）格式图像的 H.264 实时编码。

而且 H.264 才标准化，运算顺序还有改善的空间。当作为国际标准确立后，还能结集起全世界的精英来优化处理。这也反应出技术发展的必然性，唯物史观。

高压缩率使图像的数据量减少，给存储和传输带来了方便。加上基本规格公开的国际标准和公正的许可制度，所以，电视广播、家电和通信三大行业都进入到 H.264 的实际运用研发中心，见图 1。

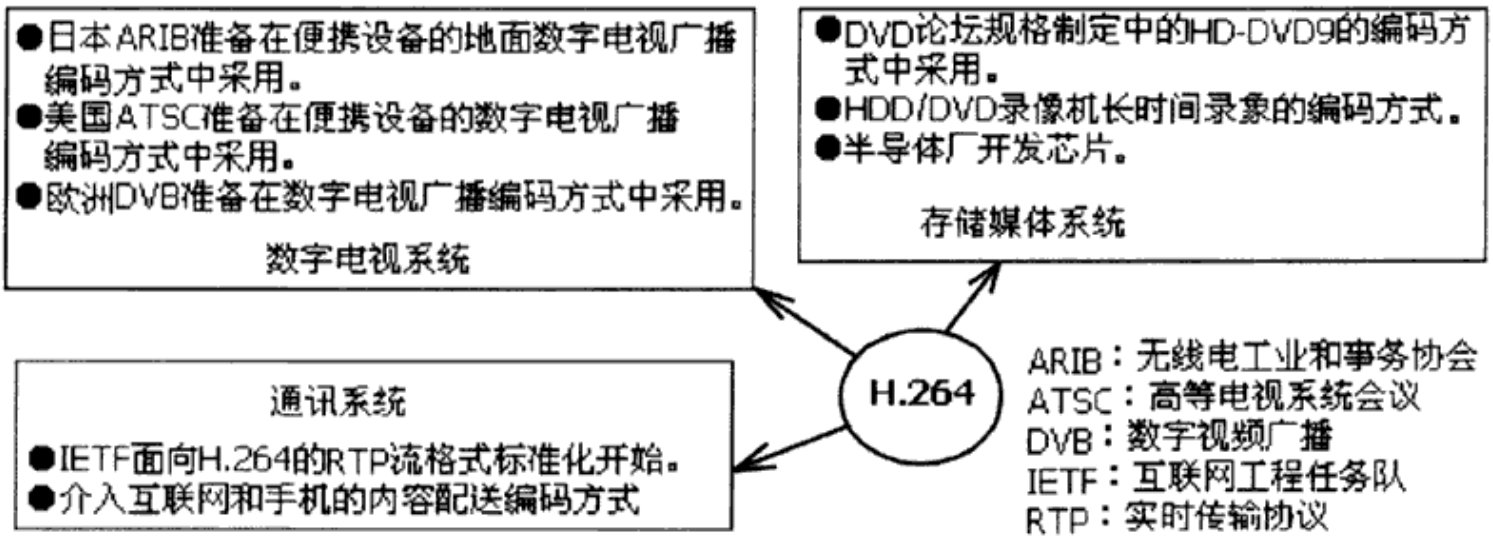

图 1:H.264 广泛的应用领域

H.264 又一项减少运算量的方法是在很多地方引入层次化运算，把在矩阵数据块变成小块运算，使计算式变得更加简单，见图 5。

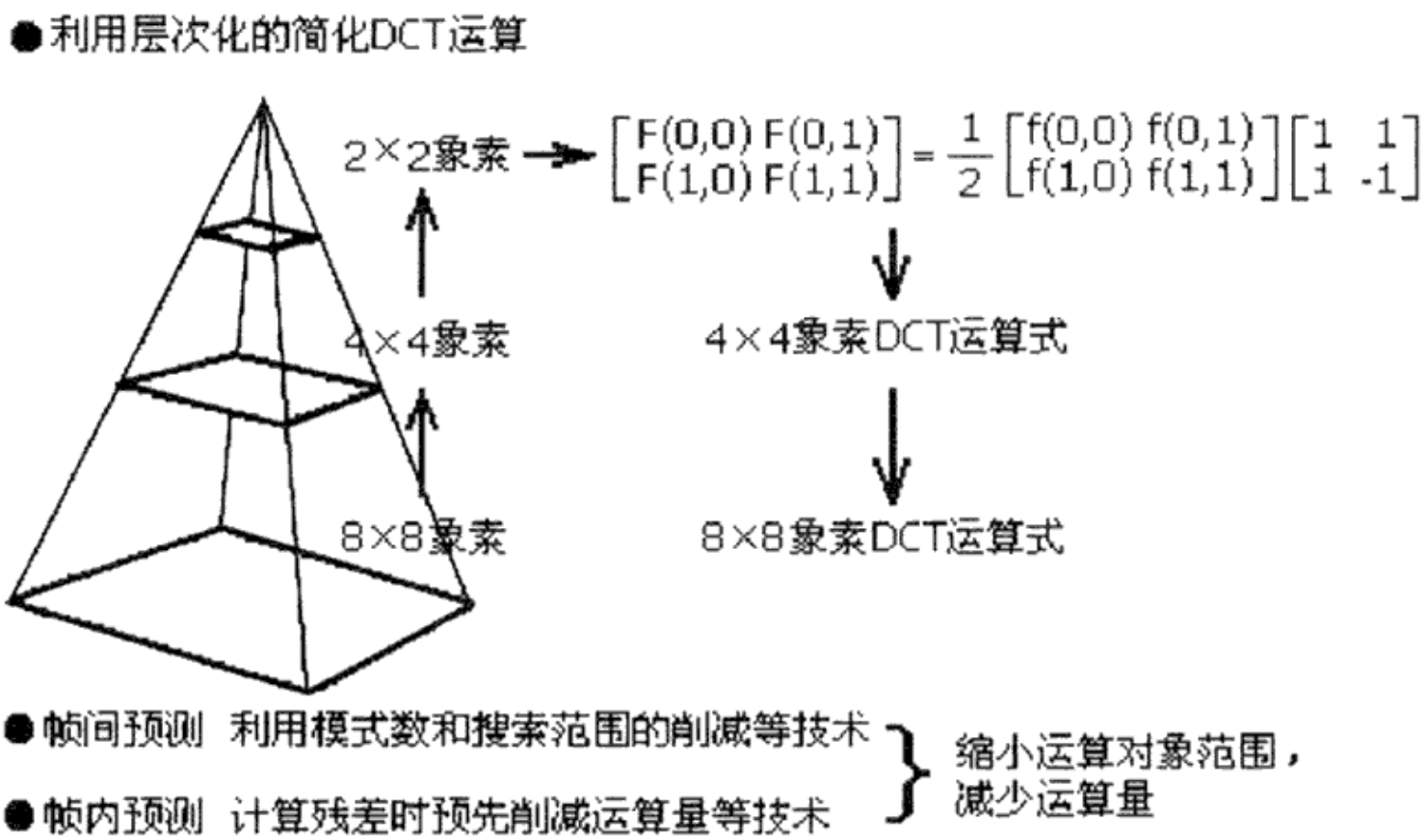

图 5:以层次化提高各种运算效率

在 DCT 中采用时，8×8 像素块层次化到 2×2 像素块，变换就变得快捷。运动补偿中也可利用。检出运动矢量时，最初的模块大，运动矢量的检出范围大，搜索快捷。当检出到有动作的部分再调入小模块细分析。H.264 进行运动预测的模板多，一旦先进全面检索，需要的时间就很长，运算量也大。用层次化处理，先进行模板的收缩，接着小范围检索，就能减少计算量。在帧内预测中利用层次化后，残差计算的范围就能变小，同样有利于减少计算量。

H.264 与 MPEG-2 和 MPEG-4 的不同还存在于纠错编码块中，H.264 的纠错编码为内容自适应可变长度码（CAVLC）和内容自适应二进制算法编码（CABAC），能提高纠错能力。而 MPEG-2 和 MPEG-4 杰霍夫曼编码。另外，还加入了 MPEG-2 和 MPEG-4 没有环路滤波器，有降低噪声的效果。H.264 的整数变换以 4×4 像素块为单位，已比原来的 8×8 像素块的块噪声少，再次降低，画质得到了进一步提高。

从应用角度看，H.264 有三个层面，分为主要用于电视会议等通信的基线层面，面向高画质用途和录像的主层面以及面向内容配送的扩展层面。各层面的清晰度和编码速度取值不同。

基线层面的主要技术为图像只含有 I 画面，P 画面，系统内有环路滤波，1/4 帧间预测，4:2:0 YUV 格式输入，基于 VLC 的纠错编码，弹性宏块指令等。主要层面则在基线层面基础上加入了 CABAC 运算编码技术和基于双向预测的 B 画面，滤波（接口）等技术，但不含弹性宏块指令。扩展层面则在基线层面里加入 B 画面和滤波编码等。

H.264 分有 4.1 种不同样式的图像水平。水平 1 的编码速度较小，最大只能达 64kbps，像素格式为 QCIF（176×144），30 帧/秒和 Sub QCIF（128×96），60 帧/秒。适合手机、PDA 等屏幕播放视频用。水平 2 的编码速度可达 2Mbps，图像的像素格式为 CIF（352× 288），30 帧/秒。水平 3、水平 4 分别对应 SDTV、HDTV 图像格式，编码速度为 10Mbps，20Mbps。另外，还有能支持更高清晰度的水平 5，编码速度高达 135 Mbps。故总称为 4.1 水平。在各水平更细的分类中，最大编码速度也还有不同规定。
最后，把 H.264 与 MPEG-2/MPEG-4 主要的不同技术比较与下表 1。

表 1:H.264 与 MPEG-2/MPEG-4 主要技术比较

主要技术	H.264	MPEG-2	MPEG-4
帧内预测	4×4 像素以 9 种，16×16 像素块 4 种预测模式	无	无
帧间预测	分 16×16 像素块 7 种模式，SDTV 图像最大预测 5 帧（预测 1/4 像素）	以 16×16 像素块为单位从前面帧预测（预测 1/2 像素）	以 16×16 和 8×8 像素块为单位从从前面帧预测（预测 1/4 像素）
变换	4×4 像素单位的整数变换	8×8 像素单位的离散余弦变换	
纠错编码	CAVLC 和 CABAC	霍夫曼编码	
环路滤波器	有	无	无

针对 H.264 的特点，编码软件和编码 LSI 开发的厂家都把编码/解码运算量的减少作为方向来研究，所以，实用前景大好。大多数半导体厂认为在 H.264 中使用削减运算量方法后，能获得相当于 MPEG-2 编码 LSI 的 2 倍左右的处理能力。

由于技术的日益成熟，半导体厂商已在进行 H.264 的编码/解码 LSI 的开发。特别是 HDD 录像机和 DVD 录像机等设备中，采用 H.264 的实例已很多，更引起了半导体厂商的关心。加之，H.264 采用的动画编码方

式和音频编码方式具有多样化特性，今后几乎将会是全部厂商的主要规格之一。

以目前芯片将 H.264 实用化的研究也在进行之中。用德州仪器（TI）公司制造的 DSP[TMS320C64××]对以 H.264 预先编码的图像已证实能进行实时解码。TI 公司正在开发的 C6×系列 DSP LSI，将在视频编码电路和存储控制电路中，加入对应 H.264 和 MWV 等的编码/解码功能。
TI 公司推出的可以对 MPEG-4 编码/解码的用于便携机开发的 TMS320DM270，只要用上新的 CPU 提高处理能力，就可用于 H.264 的编码/解码。

已经有 MWA9 的编码/解码 DSP 样品出厂的美国模拟设备公司也在向 H.264 前进。

图 6 是美国 InStat/MDR 公司对 H.264 功能 LSI 产量的预测。预测还只基于 H.264 的许可制度与 MPEG-2 一样的前提下进行的。

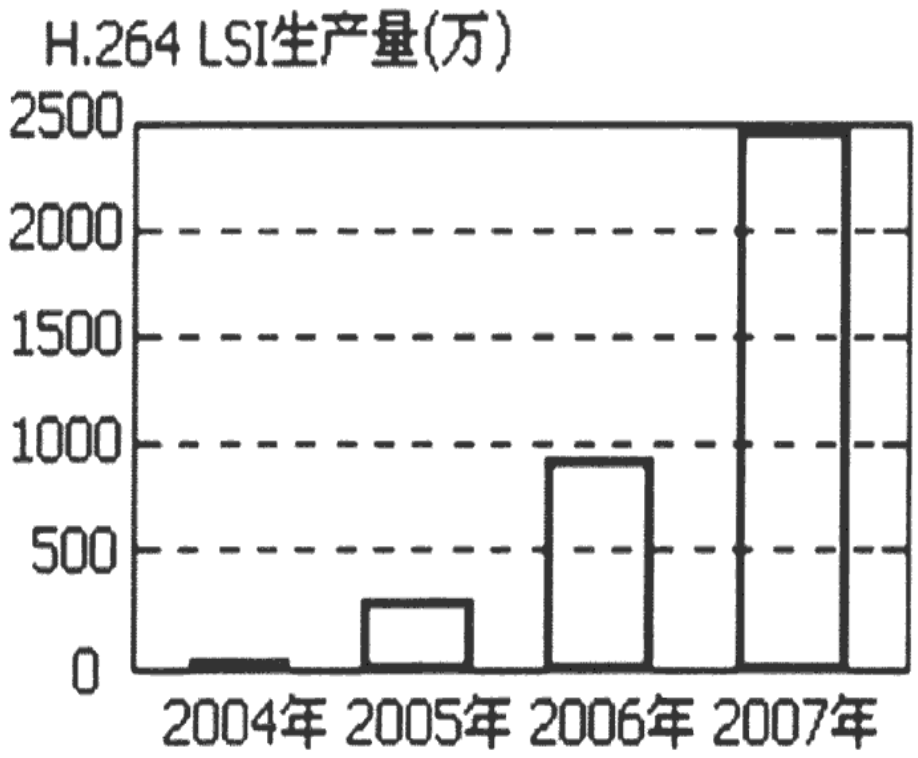

图 6:H.264 芯片产品预测

H.264 的许可制度有望较友善

H.264 替代 MPEG-4 的呼声很高，除了其高性能外，作为国际标准和公正的无差别许可制度也至关重要。

MPEG-4 的许可体系引起了几大行业，特别是信息配送行业的强烈反对，使得新国际标准的许可收费不得不向更为友善的方向发展。表 2 是几种视频压缩技术的许可收费价格。

表 2:主要图像编码技术许可费一览

编码技术名称	MPEG-2	MPEG-4 visual	Windows Media Video 9
免费版权运用	未见叙述	一年 5 万个编码器和解码器，5 万人使用以下的产品和服务	2003 年 1 月 1 和 ~ 12 月 31 日出厂的产品和在 Windows 上工作的产品
解码器	2.5 美元/个	0.25 美元/个	0.1 美元/个
解码器许可费上限（每年）	无限制	100 万美元	40 万美元
编码器	2.5 美元/个	0.25 美元/个	0.2 美元/个
编码器许可费上限（每年）	无限制	100 万美元	80 万美元
编码器/解码器	未见叙述	未见叙述	0.25 美元/个
编码器/解码器许可费上限（每年）	未见叙述	未见叙述	100 万美元
DVD 碟片（二小时电影场合）	0.03 美元/层（两层以上 0.02 美元/层）	0.04 美元（5 年内摄制的电影），0.02 美元（5 年前摄制的电影）	无
内容配送者许可费	无	0.000333 美元/分钟	无
内容配送者许可费上限	无	100 万美元/年	无
有效期限	2010 年 12 月 31 日	2008 年 12 月 31 日	2012 年 12 月 31 日

表中可见，MPEG LA 公司提出的 MPEG-4 配送过程也要付费是空前绝后的。视频压缩产品只对终端收费合乎常情，因而招至了很大反抗，直到今日仍在遭人反对。而且对采用 MPEG-4 的产品和服务还分成 6 种标准：用户记录视频，互联网视频，车载移动视频，特有用户视频，存储视频和企业视频。连简单的移动电视服务，如从现场到电视中心通讯时，若使用 MPEG-4 视频的话，也需支付移动视频的许可费。

因此，连原定在地面数字电视的编码方式中采用 MPEG-4 的***ARIB，也因许可费问题而开始研讨是否改用 H.264。拥有各种内容服务业者的移动内容论坛 MCF 也于 2003 年 5 月 23 日，致涵 MPEG LA 公司反对内容收费，要求重新考虑许可条件。MPEG LA 也已松口表示希望以能相互满意的形式交涉。

随着掌握压缩技术的企业增加和用户巨增，H.264 的许可管理收费受到二个方面的压力。一、用户要求低价格，最好免费使用；二、持有压缩技术的企业增加，供应空间大，不得不低价出售。目前具有高压缩率特征的活动图像编码技术的企业不少，如，美国数据流公司的 XVD，能在一片 CD-R 碟片上放入 2 小时图像，并能实时编码。美国 On2 技术公司的活动图像编码技术 VP5 和新版本 VP6，国内推出的 EVD 就采用这种编码技术。美国 AOL（America Online）公司也有新压缩技术在进行许可操作。微软的 WMV 9 也在向家电产品扩展，如美国工艺家庭娱乐公司使用 WMV 9 压缩，将 HDTV 画质的“终结者 2：审判日”放入 DVD-ROM 内。为此，H.264 的许可制度设计有两点引人之处：第一，部分格式将无偿使用，H.264 的基线层面全员免费，无偿使用；其二，许可体系要比 MPEG-4 单纯，公正无差别对待用户和专利持有者。以及其它能促进普及的优惠政策，如早期低价格许可等。

基线层面的免费是以 ITL-T 主要活动的企业为中心推动的。现得到美国苹果公司和美国 Cisco 系统公司、中国联想公司、芬兰诺基亚、美国 On2 技术公司、德国西门子、美国德州仪器公司等的支持，并有美国政府为其撑腰。

基线层面免费的最大目的是加速 H.264 的普及。当基线层面普及以后，收费的主层面和扩展层面就能带动起来。尽管主要层面和扩展层面要收费，但从趋势看，许可费应较为便宜，因为各种编码技术的许可费都有不断下降的趋势，目前很热门的美国微笑 WMV 9 的许可费就比 MPEG-2 和 MPEG-4 要低，见表 2。而且微软的契约期为 10 年，比 MPEG-2 和 MPEG-4 还长。

从 MPEG-2 向 MPEG-4 的发展看，编码器（电路加软件）和解码（电路加软件）的费用就降到 1/10，WMV9 更低。可以预计 H.264 的许可费用会比 WMV 9 还低。

前文提到的 45 个团体的联合会传出说法，如果 H.264 采用 MPEG-4 Visual 一样的许可体系，H.264 就可能不被采用，态度强硬。标准中的专利收费收益已远不止收回投入的开发成本，而是在不断地获取暴利，故降低收费在所必然。

当然，只要没有定局，变化依然存在。专利持有者的想法也各有不同，采用无差别对待原则是否行得通。专利实施充满着大量利益诱惑，追名逐利者大有人在。目前已经有两家公司申称对 H.264 具有许可管理权。在专利应用前就开始抢专利管理权的现象是前所未有的，两家公司还都有渊源。一家是实际持有 MPEG-2 和 MPEG-4 Visual 许可管理的美国 MPEG LA 公司。另一家是进行 MPEG-2 AAC 和 MPEG-4 Audio 许可管理的美国杜比实验室的子公司美国 Vialicensing 公司。最终有哪一家公司管理，还是分割管理，现在都不清楚。

但又好、又便宜，始终是技术发展的方向。

http://www.ednchina.com/blog/exvision/308/message.aspx

嵌入式 WEB 服务器 BOA 的移植方法

文件：boa.rar
大小：128KB
下载：下载

今天自己在开发板上移植了boa服务器。参考资料是李驹光、郑耿大侠在恒颐论坛上的帖子。李大侠写的好，我没有费什么力气就搞定了，:-)。

步骤如下：

1.从http://www.boa.org/下载Boa源码，将其解压并进入源码目录的src子目录

```
# tar -zxvf boa-0.94.13.tar.gz
# cd boa-0.94.13/src
```

2.生成Makefile文件

```
# ./configure
```

修改 Makefile 文件，

a.找到 CC=gcc，将其改成 CC = arm-linux-gcc，

b.找到 CPP = gcc –E，将其改成 CPP = arm-linux-gcc –E，

保存退出。
3.运行 make 进行编译，得到的可执行程序为 boa，并将调试信息剥去
    # make
    # arm-linux-strip boa
4.编写 boa.conf
5.cp boa 到开发板的/bin 目录下，在开发板/etc 目录下建 boa 目录，cp boa.conf 到板子的/etc/boa 目录。
6.创建日志文件所在目录/var/log/boa，创建 HTML 文档的主目录/var/www，创建 CGI 脚本所在目录/var/www/cgi-bin/，在/var/www 中放置一个 index.html 文件
7.cp mime.types 文件到开发板/etc 目录
8.vi passwd,添加 nouser 用户，vi group 添加 nogroup 组
8.运行 boa:
#/bin/boa
现在通过其他机器就可以访问了。http://192.168.0.12
就可以访问到你放置的那个 index 页面了。
9.编辑 helloworld.c 程序测试 cgi 的运行
#arm-linux-gcc -o helloworld.cgi helloworld.c
#cp helloworld.cgi 到开发板的/var/www/cgi-bin 目录下
在pc机的浏览器地址栏输入http://192.168.0.12/cgi-bin/helloworld.cgi，可以看到相关页面，CGI脚本测试通过。
10.从CGIC的主站点http://www.boutell.com/cgic/下载源码，将其解压并进入源码目录
    # tar -zxvf cgic205.tar.gz
    # cd cgic205
11.修改 Makefile 文件
a.找到 CC=gcc，将其改成 CC=arm-linux-gcc，
b.找到 AR=ar，将其改成 AR=arm-linux-ar，
c.找到 RANLIB=ranlib，将其改成 RANLIB=arm-linux-ranlib。
e.找到 gcc cgictest.o -o cgictest.cgi ${LIBS}，
  将其改成$(CC) $(CFLAGS) cgictest.o -o cgictest.cgi ${LIBS}，
f.找到 gcc capture.o -o capture ${LIBS}，
  将其改成$(CC) $(CFLAGS) capture.o -o capture ${LIBS}，
保存退出。
12.然后运行make进行编译，得到的CGIC库libcgic.a，我们通过调试辅助程序capture和测试程序cgictest.cgi，来验证生成CGIC库的正确性。
13.将capture和cgictest.cgi拷贝到主机的/var/www/cgi-bin目录下。
在工作站的浏览器地址栏输入http://192.168.0.12/cgi-bin/cgictest.cgi，可以看到页面，CGIC库和测试脚本都移植成功。

下边是李大侠的大文了。

随着 Internet 技术的兴起，在嵌入式设备的管理与交互中，基于 Web 方式的应用成为目前的主流，这种程序结构也就是大家非常熟悉的 B/S 结构，即在嵌入式设备上运行一个支持脚本或 CGI 功能的 Web 服

务器，能够生成动态页面，在用户端只需要通过 Web 浏览器就可以对嵌入式设备进行管理和监控，非常方便实用。本节主要介绍这种应用的开发和移植工作。

用户首先需要在嵌入式设备上成功移植支持脚本或 CGI 功能的 Web 服务器，然后才能进行应用程序的开发。

1、嵌入式 Web 服务器移植

由于嵌入式设备资源一般都比较有限，并且也不需要能同时处理很多用户的请求，因此不会使用 Linux 下最常用的如 Apache 等服务器，而需要使用一些专门为嵌入式设备设计的 Web 服务器，这些 Web 服务器在存贮空间和运行时所占有的内存空间上都会非常适合于嵌入式应用场合。

典型的嵌入式 Web 服务器有 Boa（www.boa.org）和 thttpd（http://www.acme.com/software/thttpd/）等，它们和 Apache 等高性能的 Web 服务器主要的区别在于它们一般是单进程服务器，只有在完成一个用户请求后才能响应另一个用户的请求，而无法并发响应，但这在嵌入式设备的应用场合里已经足够了。

我们介绍比较常用的 Boa 服务器的移植。Boa 是一个非常小巧的 Web 服务器，可执行代码只有约 60KB。它是一个单任务 Web 服务器，只能依次完成用户的请求，而不会 fork 出新的进程来处理并发连接请求。但 Boa 支持 CGI，能够为 CGI 程序 fork 出一个进程来执行。Boa 的设计目标是速度和安全，在其站点公布的性能测试中，Boa 的性能要好于 Apache 服务器。

第一步完成 Boa 程序的移植。从 www.boa.org 下载 Boa 源码，当前最新版本为 0.94.13，将其解压并进入源码目录的 src 子目录

```
# tar xzf boa-0.94.13.tar.gz
# cd boa-0.94.13/src
```

生成 Makefile 文件

```
# ./configure
```

修改 Makefile 文件，找到 CC=gcc，将其改成 CC = arm-linux-gcc，再找到 CPP = gcc –E，将其改成 CPP = arm-linux-gcc –E，并保存退出。

然后运行 make 进行编译，得到的可执行程序为 boa，将调试信息剥去，得到的最后程序只有约 60KB 大小。

```
# make
# arm-linux-strip boa
```

第二步完成 Boa 的配置，使其能够支持 CGI 程序的执行。Boa 需要在/etc 目录下建立一个 boa 目录，里面放入 Boa 的主要配置文件 boa.conf。在 Boa 源码目录下已有一个示例 boa.conf，可以在其基础上进行修改，下面解释一下该文件的含义：

```
#监听的端口号，缺省都是 80，一般无需修改
Port 80
# bind 调用的 IP 地址，一般注释掉，表明绑定到 INADDR_ANY，通配于服务器的所有 IP 地址
#Listen 192.68.0.5
#作为哪个用户运行，即它拥有该用户的权限，一般都是 nobody，需
要 /etc/passwd中有nobody用户
User nobody
#作为哪个用户组运行，即它拥有该用户组的权限，一般都是nogroup，需要在/etc/group文件中有nogroup组
```

Group nogroup
#当服务器发生问题时发送报警的email地址，目前未用，注释掉
#ServerAdmin root@localhost
#错误日志文件。如果没有以/开始，则表示从服务器的根路径开始。如果不需要错误日志，则用#/dev/null。
在下面设置时，注意一定要建立/var/log/boa目录
ErrorLog /var/log/boa/error_log
#访问日志文件。如果没有以/开始，则表示从服务器的根路径开始。如果不需要错误日志，则用#/dev/null或直接注释掉。***在下面设置时，注意一定要建立/var/log/boa目录***
#AccessLog /var/log/boa/access_log
#是否使用本地时间。如果没有注释掉，则使用本地时间。注释掉则使用UTC时间
#UseLocaltime
#是否记录CGI运行信息，如果没有注释掉，则记录，注释掉则不记录
#VerboseCGILogs
#服务器名字
ServerName www.hyesco.com
#是否启动虚拟主机功能，即设备可以有多个网络接口，每个接口都可以拥有一个虚拟的Web服务器。一般注释掉，即不需要启动
#VirtualHost
#非常重要，HTML文档的主目录。如果没有以/开始，则表示从服务器的根路径开始。
DocumentRoot /var/www
#如果收到一个用户请求的话，在用户主目录后再增加的目录名
UserDir public_html
#HTML目录索引的文件名，也是没有用户只指明访问目录时返回的文件名
DirectoryIndex index.html
#当HTML目录没有索引文件时，用户只指明访问目录时，boa会调用该程序生成索引文件然后返回给用户，因为该过程比较慢最好不执行，可以注释掉或者给每个HTML目录加上DirectoryIndex指明的文件
#DirectoryMaker /usr/lib/boa/boa_indexer
#如果DirectoryIndex不存在，并且DirectoryMaker被注释，那么就用Boa自带的索引生成程序来生成目录的索引文件并输出到下面目录，该目录必须是Boa能读写
DirectoryCache /var/spool/boa/dircache
#一个连接所允许的HTTP持续作用请求最大数目，注释或设为 0 都将关闭HTTP持续作用
KeepAliveMax 1000
#HTTP持续作用中服务器在两次请求之间等待的时间数，以秒为单位，超时将关闭连接
KeepAliveTimeout 10
#指明mime.types文件位置。如果没有以/开始，则表示从服务器的根路径开始。可以注释掉避免使用mime.types文件，此时需要用AddType在本文件里指明
MimeTypes /etc/mime.types
#文件扩展名没有或未知的话，使用的缺省MIME类型
DefaultType text/plain
#提供CGI程序的PATH环境变量值
CGIPath /bin:/usr/bin:/usr/local/bin
#将文件扩展名和MIME类型关联起来，和mime.types文件作用一样。如果用mime.types
#文件，则注释掉，如果不使用mime.types文件，则必须使用
#AddType application/x-httpd-cgi cgi

#指明文档重定向路径
#Redirect /bar http://elsewhere/feh/bar
#为路径加上别名
Alias /doc /usr/doc
#非常重要，指明CGI脚本的虚拟路径对应的实际路径。一般所有的CGI脚本都要放在实际路径里，用户访问执行时输入站点+虚拟路径+CGI脚本名
ScriptAlias /cgi-bin/ /var/www/cgi-bin/

用户可以根据自己需要，对boa.conf进行修改，但必须要保证其他的辅助文件和设置必须和boa.conf里的配置相符，不然Boa就不能正常工作。在上面的例子中，我们还需要创建日志文件所在目录/var/log/boa，创建HTML文档的主目录/var/www，将mime.types文件拷贝到/etc目录，创建CGI脚本所在目录/var/www/cgi-bin/。mime.types文件用来指明不同文件扩展名对应的MIME类型，一般可以直接从Linux主机上拷贝一个，大部分也都是在主机的/etc目录下。

第三步就是测试 Boa 能否正常工作，静态 HTML 页面能否正常访问，CGI 脚本能否正常运行，一般采用 NFS 方式来进行测试工作，可以将嵌入式目标系统上的/etc 目录拷贝到主机的 NFS 共享目录下，然后将 NFS 共享目录下的 etc 目录重新 NFS mount 为目标系统上的/etc 目录。这样就可以在主机上对 etc 目录下的各种配置文件，如进行修改而立刻在目标系统上生效。

假设主机/nfs 目录为共享目录，在其下面建立一个 www 子目录作为 Web 站点的主目录，其内容如下：

```
# ls /nfs/www
cgi-bin  images index.html
```

index.html为测试主页面，images为存放各种图片的子目录，cgi-bin为CGI脚本的存放目录。根据示例boa.conf的配置，目前HTML文档的主目录为/var/www，CGI脚本目录为/var/www/cgi-bin，则运行以下命令将主机的/nfs/www目录mount成目标板上的/var/www目录。然后就可以运行boa了：

```
# mount -t nfs 192.168.0.20:/nfs/www /var/www -o nolock
# boa
```

在工作站上运行浏览器进行测试，在地址栏输入目标系统IP，即http://192.168.0.162 ，可以看到相关页面，表示静态HTML页面测试通过。

接下来进行 CGI 脚本的测试，我们需要一个测试用的 CGI 脚本。可以写个最简单的 Hello World 程序，示例代码如下

```
#include <stdio.h>
void main() {
   printf("Content-type: text/html\n\n") ;
   printf("<html>\n") ;
   printf("<head><title>CGI Output</title></head>\n") ;
   printf("<body>\n") ;
   printf("<h1>Hello, world.</h1>\n") ;
   printf("</body>\n") ;
   printf("</html>\n") ;
   exit(0) ;
}
```

然后进行交叉编译，将得到的 helloworld.cgi 拷贝到主机的/nfs/www/cgi-bin 目录下。

```
#arm-linux-gcc -o helloworld.cgi helloworld.c
#cp helloworld.cgi /nfs/www/cgi-bin
```

在浏览器地址栏输入 http://192.168.67.16/cgi-bin/helloworld.cgi，可以看到相关页面，表示 CGI 脚

本测试通过。

现在我们已经可以让 Boa 在嵌入式目标系统上正常工作了，嵌入式 Web 服务器移植成功。在以上的移植过程中，最好设定 boa.conf 中的错误日志文件 ErrorLog，允许 Boa 记录错误信息；测试静态 HTML 页面和 CGI 脚本时，不管结果如何，最好都查看错误日志文件；CGI 脚本测试很容易发生权限不够的错误，要保证 Boa 访问的主目录、CGI 脚本目录以及临时文件目录（如果没有设置 TMP 环境变量时，缺省是/tmp 目录），都必须能被 Boa 运行时所代表的用户完全访问，该用户由 boa.conf 中的 User 指出。

目前 Web 技术中生成动态 Web 页面的方法有 CGI 和服务器脚本，如 JSP, ASP 等，但后者需要 Web 服务器具有这些脚本的运行支持模块。在嵌入式 Web 服务器中，考虑到资源限制问题，一般都只提供 CGI 支持，因此在嵌入式设备中 Web 方式应用实际上就是基于 CGI 的程序开发。

CGI（Common Gate Intergace）是一段运行在 Web 服务器上的程序，提供同客户端 Html 页面的接口。我们看一个实际例子：常见的个人主页上大都有一个留言本，留言本的工作方式是这样的：先由用户输入一些信息，如名字之类的东西，接着用户按一下“留言”（到目前为止工作都在客户端），浏览器就把这些信息传送到服务器的 CGI 程序中，于是 CGI 程序在服务器上按照预定的方法进行处理，在本例中就是把用户提交的信息存入指定的文件中，最后 CGI 程序给客户端发回一个“留言结束”字样的页面，用户可以在浏览器里看到。

在进行 CGI 编程之前，我们先了解 HTML 的一些知识。CGI 可以使用多种编程语言来实现，包括 C、 C++、Perl 等，但在嵌入式设备的开发中，一般都不会采用 Perl 等解释性语言，因为这种语言还需要有解释执行的支撑模块，会占用存贮空间和内存，最常用的方法当然是用 C 来编写，但 C 并不是很适合开发象 CGI 这种需要大量进行字符串操作的程序，编程比较烦琐，因此，对于一个专业的开发人员来说，首先想到的应该是有没有可复用的库来支持快速高效的开发 CGI 程序。幸运的是目前就有不少开放源码的支持 CGI 开发的 C 库。我们在此只介绍 CGIC，有兴趣的朋友可以自己在 Internet 上搜索其他的 C 库。

CGIC 库的移植

CGIC 是一个支持 CGI 开发的开放源码的标准 C 库，可以免费使用，只需要在开发的站点和程序文档中有个公开声明即可，表明程序使用了 CGIC 库，用户也可以购买商业授权而无需公开声明。

CGIC 能够提供以下功能：

1 分析数据，并自动校正一些有缺陷的浏览器发来的数据；
2 透明接收用 GET 或 POST 方法发来的 From 数据；
3 能接受上传文件；
4 能够设置和接收 cookies；
5 用一致的方式处理 From 元素里的回车；
6 提供字符串，整数，浮点数，单选或多选功能来接收数据；
7 提供数字字段的边界检查；
8 能够将 CGI 环境变量转化成 C 中的非空字符串；
9 提供 CGI 程序的调试手段，能够回放 CGI 程序执行时的 CGI 状态；

总之，CGIC 是一个功能比较强大的支持 CGI 开发的标准 C 库，并支持 Linux, Unix 和 Windows 等多操作系统。

以下描述 CGIC 的移植过程。

从 CGIC 的主站点 http://www.boutell.com/cgic/下载源码，当前最新版本是 2.05 版。将其解压并进入源码目录

```
# tar xzf cgic205.tar.gz

# cd cgic205
```

修改 Makefile 文件，找到 CC=gcc，将其改成 CC=arm-linux-gcc，找到 AR=ar，将其改成 AR=arm-linux-ar，找到 RANLIB=ranlib，将其改成 RANLIB=arm-linux-ranlib。找到 gcc cgictest.o -o cgictest.cgi ${LIBS}，将其改成$(CC) $(CFLAGS) cgictest.o -o cgictest.cgi ${LIBS}，找到 gcc capture.o -o capture ${LIBS}，将其改成$(CC) $(CFLAGS) capture.o -o capture ${LIBS}，并保存退出。

然后运行 make 进行编译，得到的 CGIC 库 libcgic.a，我们通过调试辅助程序 capture 和测试程序 cgictest.cgi，来验证生成 CGIC 库的正确性。

将 capture 和 cgictest.cgi 拷贝到主机的/nfs/www/cgi-bin 目录下。

在工作站的浏览器地址栏输入 http://192.168.67.16/cgi-bin/cgictest.cgi，可以看到页面，表示 CGIC 库和测试脚本都移植成功。cgictest.cgi 比较完整的展现了 CGIC 库的功能，在开发基于 CGIC 库的 CGI 程序前最好先掌握 cgictest.cgi 程序，也是用户开发特定应用程序时的参考范例。

HTML 模板的制作

Web 方式的应用开发一般都会将界面和程序逻辑脱离开来，允许在一定程度下更改界面，如改变界面文本的属性，建立多语言版本等，而无需改动程序逻辑。界面一般由美工来进行制作，而程序员负责具体功能的实现。在 HTML 中，表单 (FORM)是最主要的传递信息的手段，它适用于任何浏览器。表单中有很多元素，包括输入文本框，单选框，多选框，按钮，等等，可以提供信息的交互。具体对象说明和语法请参见其他 HTML 书籍，在这里不作介绍。根据应用需求，美工或其他设计人员将最后的 Web 页面设计出来，作为程序员进行开发的模板。

CGI 程序的工作一般就是接收表单数据，进行数据处理，最后根据处理结果生成新的页面返回给浏览器。表单数据一般是以 POST 方法提交给服务器，由 CGI 程序获得，程序必须要将界面数据和内部数据对应起来才能够进行下一步的处理。CGI 程序从页面获取数据就根据元素名字/值中的元素名字来进行区分。但 CGI 返回页面就比较麻烦。由于界面在程序开发完成后还有可能会改变，而且有些需要程序处理的地方可能没有表单元素，因此对程序来说，不能以表单元素名作为区分的基础，一般方法是采用 HTML 中的注释<!-xxx-->来标记。

程序员需要在模板中为每一个表单元素以及其他任何需要程序处理的地方，按照一定规则，如注释的下一行就是表单元素行，建立其注释标记。CGI 程序就可以根据注释标记来判断表单元素信息并进行处理。程序逐行读取模板文件，检查有无注释标记，如有的话，则下一行需要进行处理，给表单元素赋上数据，

最后就可以返回带数据的页面给浏览器。

HTML 模板还需要关注的是输入的检查。根据输入检查越早越好的原则，需要在用户界面上就对用户提交的数据进行检查。目前一般是采用 javascript 脚本的方式。当用户提交数据时，表单对象的 onSubmit 方法就会被调用，在该方法里就可以对用户的输入进行检查。常用的检查有是否必需、最大/小长度、是否字符、是否数字、email 地址、IP 地址是否正确、是否匹配一个正则表达式等。

CGI 程序的开发

CGI 程序的工作一般就是接收表单数据，根据应用需求进行数据处理，最后根据处理结果生成新的页面返回给浏览器。表单数据一般是以 POST 方法提交给服务器，由 CGI 程序获得，程序根据元素名字/值中的元素名字来区分数据，完成数据处理后，再读取相应的模板文件，根据注释标记将对应的数据填充到 HTML 文本中去，生成最后的页面返回给浏览器。

程序一般逻辑为：

1. 安全性检查，是否允许运行脚本；
2. 处理用户提交的数据，根据元素名字/值中的元素名字来区分数据，然后根据应用需求进行数据处理；
3. 将处理结果填充表单，根据注释标记将对应的数据填充到 HTML 文本中去，生成最后的页面返回给浏览器。

关于具体的代码实现细节，用户可以参考《嵌入式 Linux 系统开发详解－基于 EP93XX 系列 ARM》一书的相关章节。（全文完）

Web 服务器--Boa 实验笔记

该文章转载自宋氏电脑 技术无忧：

http://www.pc51.net/system/unix/linux/2007-02-07/6888.html

嵌入式 Linux 系统开发详解——基于 EP93XX 系列 ARM》一书和 boa 自带的文档等对该内容有比较详细的介绍，但在实验过程中，仍可能会出现一些问题。下面是我在 MIZI Linux SDK for S3C2410 平台实验时记录下的实验步骤和出现的问题及解决方法，欢迎讨论。

13.WEB 服务器实验

13.1 Boa 移植

13.1.1 Boa 移植步骤

第一步：从 www.boa.org 下载 Boa 源码，将其解压并进入源码目录的 src 子目录。

tar xzf boa-0.94.13.tar.gz

cd boa-0.94.13/src

生成 Makefile 文件

./configure

修改 Makefile 文件，找到 CC=gcc 和 CPP=gcc -E，分别将其改为交叉编译器安装的路径 CC=/opt/host/armv4l/bin/armv4l-unknown-linux-gcc 和 CPP=/opt/host/armv4l/bin/armv4l-unknown-linux-gcc -E 并保存退出。

然后运行 make 进行编译，得到可执行程序 boa

make

/opt/host/armv4l/bin/armv4l-unknown-linux-strip boa

第二步：Boa 的配置。

Boa 需要在/etc 目录下建立一个 boa 目录，里面放入 Boa 的主要配置文件 boa.conf。在 Boa 源码目录下已有一个示例 boa.conf，可以在其基础上进行修改。

1. Group nogroup 修改成 Group 0，由于在/etc/group 文件中没有 nogroup 组，所以设成 0

另外在/etc/passwd 中有 nobody 用户，所以 User nobody 不用修改。

2. ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/ 修改成 ScriptAlias /cgi-bin/ /var/www/cgi-bin/

其它默认设置即可。

还需要创建日志文件所在目录/var/log/boa，创建 HTML 文档的主目录/var/www，将静态网页存入该目录下（可以将主机/usr/share/doc/HTML/目录下的 index.html 文件和 img 目录复制到/var/www 目录下），创建 CGI 脚本所在目录/var/www/cgi-bin，将 cgi 的脚本存放在该目录下。另外还要将 mime.types 文件复制/etc 目录下，通常可以从 linux 主机的/etc 目录下直接复制即可。

第三步：测试

1．静态 HTML 网页

在目标板上运行 boa 程序，将主机与目标机的 ip 设成同一网段，然后打开任一个浏览器（linux 或 window 下都可），输入目标板的 ip 地址（http：//10.10.10.2）即可打开/var/www/index.html 网页

2．CGI 脚本测试

boa 源码中有 cgi-bin 测试脚本(boa-0.94.13/examples/cgi-test.cgi)，但

由于它不是在我们目标平台下编译的，所以不能将它复制到

/var/www/cgi-bin 目录下进行测试。

下面是一个简单的 hello, world!程序，代码如下：

```
#include <stdio.h>

void main(){

printf("Content-type: text/html\n\n");

printf("<html>\n");

printf("<head><title>CGI Output</title></head>\n");

printf("<body>\n");

printf("<hl>Hello, world.</hl>\n");

printf("<body>\n");

printf("</html>\n");

exit(0);

}
```

13.1.2 Boa 移植时出现的问题

1. 当运行 boa 程序时出现错误，如下：

```
# ./boa

[27/Nov/1990:13:22:25 + 0000]boa.c:266.icky Linux kernel bug!:No

such file
```

将 User 0 修改成 User nobody

2. 打开网页时，网页中的图片无法显示

就将存放图片的子目录/var/www/images 修改成/var/www/img

3. 在测试 cgi 脚本时，当在浏览器地址中输入“http//10.10.10.2/cgi-bin/helloworld.cgi”时，浏览输出下述错误：

502 Bad Gateway

The CGI was not CGI/1.1 compliant

在目标板的调试终端中输出下述错误：

……cgi_header：unable to find LFIF

上述错误是在 boa 原码中的 cgi_header .c 文件中的 process_cgi_header 函数产生，如下：

```
buf = req->header_line;
c = strstr(buf, "\n\r\n");
if (c == NULL) {
 c = strstr(buf, "\n\n");
 if (c == NULL) {
  log_error_time();
fputs("cgi_header: unable to find LFLF\n", stderr);    //出错信息
#ifdef FASCIST_LOGGING
  log_error_time();
 fprintf(stderr, "\"%s\"\n", buf);
 #endif
 send_r_bad_gateway(req);
  return 0;
```

}

}

我们可以定义 FASCIST_LOGGING，使产生该错误时将 buf 内容打印出来，结果如下：

……cgi_header：unable to find LFIF

Content-type: text/html

<html>

<head><title>CGI Output</title></head>

<body>

<hl>Hello, world.</hl>

<body>

</html>

原来 buf 中的内容就是 helloworld.c 中输出的内容，查看输出结果，再看 process_cgi_header 函数中的语句：c = strstr(buf, "\n\n")，很明显 buf 中没有两个连续的换行符"\n\n"，所以是 helloworld.c 文件中的语句：printf("Content-type: text/html\n\n");错写成了 printf("Content-type: text/html\n");

上述行通过标准输出将字符串〞Contenttype:text/plain\n\n〞传送给 Web 服务器。它是一个 MIME 头信息,告诉 Web 服务器随后的输出是以纯 ASCII 文本的形式。在这个头信息中有两个新行符,这是因为 Web 服务器需要在实际的文本信息开始之前先看见一个空行。

13.2 WEB 应用开发

13.2.1 CGIC 库的移植

从 CGIC 的主站点 http://www.boutell.com/cgic/上下载源码，将其解压并进入源码目录。

tar –zxvf cgic.tar.gz

cd cgic205

修改 Makefile 文件，找到 CC=gcc，将其改为 CC=/opt/host/armv4l/bin/armv4l-unknown-linux-gcc，找到 AR=ar，将其改为 AR=/opt/host/armv4l/bin/armv4l-unknown-linux-ar，找到 RANLIB=ranlib，将其改为 RANLIB=/opt/host/armv4l/bin/armv4l-unknown-linux-ranlib。找到 gcc cgictest.o －o cgictest.cgi ${LIBS}，将其改为$(CC) $(CFLAGS) cgictest.o －o cgictest.cgi ${LIBS}，找到 gcc capture.o －o capture ${LIBS}，并保存退出。

然后输入 make 进行编译，可以将生成的 capture 和 cgictest.cgi 文件复制到目标板目录/var/www/cgi-bin/下，以测试正确性。

13.3.2 基于 CGIC 库的例程

参考《嵌入式 Linux 系统开发详解——基于 EP93XX 系列 ARM》一书中的 list.html 和 list.c

13.3.3 例程出现的问题

1 > 点击 submit(提交)按钮后，终端输出错误信息：…mkstemp：Permission denied

浏览器弹出错误对话框：此文档中无数据

原因：是在 boa 源代码目录下的 util.c 文件中，用 mkstemp()函数创建一个临时文件时出现权限错误。这是由于在 boa.conf 文件设置了 user：nobody，使其运行 boa 服务器时以 nobody 为用户（可以用 ps 命令查看），所以在创建临时文件是没有足够的权限，可以在 boa.conf 中将运行 boa 的用户为 root 身份(user：root)。

2 > 当上述设置 user：root 时，运行 boa 时，在终端会输出错误信息：

boa.c:266.icky Linux kernel bug!:No such file

原因：boa.c 文件中的下述语句出现问题，可以将 boa.c 文件中的该行屏蔽掉。

```
if (setuid(0) != -1) {
    DIE("icky Linux kernel bug!");
}
```

3 > 点击例程中的 submit(提交)按钮后浏览器出现下述错误：

地址变为：http://10.10.10.2/var/www/cgi-bin/list.cgi

内容：400 Bad Request Your client has issued a malfarned or illegal request

原因是 list.html 中<form name="SystemCont" method="post" action="/cgi-bin/list.cgi" onSubmit="return checkform()">写成了<form name="SystemCont" method="post" action="/var/www/cgi-bin/list.cgi" onSubmit="return checkform()">

4 > 在查看目录项输入目录如：/var/www/cgi-bin/，然后点击 submit(提

交)按钮后，浏览器恢复原样，而没有在文本框中显示 ls /var/www/cgi-bin 的内容。

原因是在 list.html 文件中的<!--Dir-->和<!--Files-->的前面有空格，使 list.c 文件中下述语句出错，所以在 list.html 中的上面两个注释一定要写在行的开头。

```
if (strcmp(buf,"<!--Dir-->")==0)
{……}
if (strcmp(buf,"<!--Files-->")==0)
{……}
```

该文章转载自宋氏电脑 技术无忧：

http://www.pc51.net/system/unix/linux/2007-02-07/6888.html

第一章：基础的基础

回CGI教程目录

1.1 为什么使用 CGI？

我没有把什么是 CGI 放在基础篇的第一段，是因为实在很难说明白到底什么是 CGI。而

如果你先知道 CGI 有什么作用，将会很好的理解 CGI 是什么这个概念。 CGI 可以为我们提供许多

HTML 无法做到的功能。比如 a.一个记数器 b.顾客信息表格的提交以及统计 c.搜索程 d.WEB 数

据库 用 Html 是没有办法记住客户的任何信息的，就算用户愿意让你知道。用 Html 也是无法把信

息记录到某一个特定文件里的。要把客户段的信息记录在服务器的硬盘上，就要用到 CGI。 这是

CGI 最重要的作用，它补充了 Html 的不足。是的，仅仅是补充，不是替代。

1.2 CGI 是什么？

好了，现在我们来说到底什么是 CGI。Common Gate Intergace 听起来让人有些专业,

我们就管它叫 CGI 好了。在物理上，CGI 是一段程序，它运行在 Server 上，提供同客户段 Html 页

面的接口。这样说大概还不好理解。那么我们看一个实际例子： 现在的个人主页上大部分都有

一个留言本。留言本的工作是这样的：先由用户在客户段输入一些信息，如名字之类的东西。接

着用户按一下“留言”（到目前为止工作都在客户端），浏览器把这些信息传送到服务器的 CGI

目录下特定的 cgi 程序中，于是 cgi 程序在服务器上按照预定的方法进行处理。在本例中就是把

用户提交的信息存入指定的文件中。然后 cgi 程序给客户端发送一个信息，表示请求的任务已经

结束。此时用户在浏览器里将看到“留言结束”的字样。整个过程结束。

1.3 选择你熟悉的编程语言

既然 CGI 是一种程序，自然需要用编程语言来写。你可以用任何一种你熟悉的高级语

言， C,C++,C shell 和 VB。值得特别指出的，有一种叫 Perl 的语言。其前身是属于 Unix 专用的

高 级语言，其具有强大的字符串处理能力而成为现在写 CGI，特别是表单类程序的首选。最近

它已经有了 Window95，和 winnt 版本。你可以在搜索程序里找到在那里下载它。VB 是 Ms 的杀手

锏，从目前的情况看，微软公司正试图使 VB 无所不能。自然也包括在 Internet 请各位注意，

VB 开发的程序只能在 windows 平台上被执行，所以它有一定局限。 C Shell，经典的语言。可惜

能做的事情不多，而且必须在 Unix 平台下。 C,C++，正真的无所不能。可是在写 CGI 的时候显得

非常难以掌握。特别是缺乏可以灵活使用的字符串处理函数。对程序员的要求也比较高，维护复

杂。 最后要提醒各位，因为 CGI 是 Server 和 Clinet 的接口，所以对于不同的 Server,CGI 程序的

移值是一个很复杂的问题。一般对于不同的 Server,决没有两个可以互相通用的 CGI。实际上 这

就是 CGI 程序最复杂的地方。

1.4 安全

我想各位敏感的朋友又要问我关于安全性能的问题了。实际上 CGI 是比较安全的，至

少比 那些没有数字签名的 ActiveX 控件要安全的多。除非你有意在程序里加入了破坏 Server 的

命令， 否则一般不会有什么严重的后果。而个人网站不向大众开放 CGI 目录，则因为怕各位学习

不精，无端增加服务器的负担，所以一般不提供。

小结：

本章讲述了 CGI 基本概念，也说明了各种编程语言的优缺点，同时解释了为什么个人

网站不提供 CGI 的原因。接下来我们开始正式学习。