

• 内容目录

- - [toc: true](#)
 - [一、交叉编译环境](#)
 - [1.解压安装](#)
 - [2.把编译器路径加入系统环境变量](#)
 - [二、Linux下Qt4的安装](#)
 - [1.Qt源码及QtCreator下载](#)
 - [2.安装QtCreator4.0.3](#)
 - [3.安装Qt4.8.6](#)
 - [a.解压](#)
 - [b.进入目录配置](#)
 - [c.编译](#)
 - [d.安装](#)
 - [e.设置环境变量](#)
 - [f.测试](#)
 - [3.Qt4.8.6+QtCreator4.0.3配置](#)
 - [三、Linux下Qt5的安装](#)
 - [1.Qt源码及QtCreator下载](#)
 - [2.安装](#)
 - [3.配置](#)
 - [四、Qt开发项目基本界面](#)
 - [1.radioButton&buttonGroup](#)
 - [a.QtCreator建立buttonGroup](#)
 - [b.获取选中的radioButton](#)
 - [2.QLCDNumber](#)
 - [3.textBrowser](#)
 - [4.QString与int类型转换](#)
 - [QString转int: toInt\(\)](#)
 - [int转QString](#)
 - [5.添加图片](#)
 - [6.调用可执行程序及终端命令](#)
 - [7.对字符串进行操作](#)
 - [a.使某个字符填满字符串](#)
 - [b.从字符串查找相同的字符串](#)
 - [c.指定位置插入字符串](#)
 - [d.判断字符串是否为空](#)
 - [e.判断字符串是否存在](#)
 - [f.从左向右截取字符串](#)
 - [g.从中间截取字符串](#)
 - [h.删除字符串中间某个字符](#)
 - [i.替换字符串中的某些字符](#)
 - [j.以某个字符切割字符串](#)
 - [k.其他类型转为QString](#)
 - [l.](#)
 - [8.测试PC与ARM正常通信](#)
 - [9.socket通信及TCP协议通信源码](#)
 - [10.QT界面代码](#)
 - [11.调用程序的编译及上传运行](#)
 - [12.ARM开发板上的程序开机启动](#)
 - [附: vi指令](#)
 - [13.QSS相关](#)
 - [14.自动进样器代码](#)

title: 【O】LinuxQT_ARM开发从零开始
date: 2018-7-13
嵌入式
description: LinuxQT_ARM开发从零开始
keywords: Linux,QT,ARM,从零开始,mengze,梦泽

toc: true

项目开始于2018.6.20; 结束于2018.7.10。
PC端为Ubuntu下的QT4及arm-linux-gcc交叉编译环境, ARM为ARM9的官方FriendlyARM实验箱及配套软件系统。

[GitHub地址: IonChromatograph](#)

一、交叉编译环境

1.解压安装

```
1. tar xvzf arm-linux-gcc-4.5.1-v6-vfp-20101103.tgz -C/
```

注意: C 后面有个空格, 并且C是大写的, 它是英文单词“Change”的第一个字母, 在此 是改变目录的意思。
执行该命令, 将把 arm-linux-gcc 安装到/opt/FriendlyARM/toolschain/4.5.1 目录。

2.把编译器路径加入系统环境变量

```
1. gedit /root/.bashrc
```

编辑/root/.bashrc 文件，修改最后一行为

```
1. export PATH=$PATH:/opt/FriendlyARM/toolschain/4.5.1/bin
```

保存退出，重新登录系统，使以上设置生效，在命令行输入

```
1. arm-linux-gcc -v
```

会出现版本信息，这说明交叉编译环境已经成功安装。

二、Linux下Qt4的安装

1.Qt源码及QtCreator下载

Qt4.8.6源码：

http://download.qt.io/official_releases/qt/4.8/4.8.6/
下载【qt-everywhere-opensource-src-4.8.6.tar.gz】。

QtCreator4.0.3

http://download.qt.io/official_releases/qtcreator/4.0/4.0.3/
下载【qt-creator-opensource-linux-x86_64-4.0.3.run】。

2.安装QtCreator4.0.3

```
1. ./qt-creator-opensource-linux-x86_64-4.0.3.run
2. 提示：输入前几个字符后按下tab可快捷补全文件名
```

进入QtCreator安装界面，指定安装位置，然后就是按照提示一直到安装结束。

提示：若提示

```
1. sudo: unable to execute ./qt-creator-opensource-linux-x86_64-4.0.3.run: 没有那个文件或目录
```

首先确认你在输入指令的目录下确实有这个文件，若确认有还提示错误的话，大概是因为那个.run文件没有“执行”属性，执行以下命令再试试：

```
1. chmod +x qt-creator-opensource-linux-x86_64-4.0.3.run
```

3.安装Qt4.8.6

a.解压

```
1. tar xzvf qt-everywhere-opensource-src-4.8.6.tar.gz
```

b.进入目录配置

```
1. cd ./qt-everywhere-opensource-src-4.8.6
2. ./configure
```

输入c回车，表示使用社区版。
输入o回车，表示使用开源版。
然后输入yes回车，表示同意协议。

提示：若提示

```
1. Basic XLib functionality test failed!
2. You might need to modify the include and library search paths by editing QMAKE_INCDIR_X11 and QMAKE_LIBDIR_X11 in /home/usr/local/qt/mkspecs/gnu-g++/linux-g++-arm-linux-gnueabi.conf
```

是因为缺少libX11的开发包，根据自己的系统特点，安装 libX11-dev libXext-dev libXtst-dev

```
1. apt-get install libX*
2. //或
3. apt-get install libX11-dev libXext-dev libXtst-dev
```

c.编译

```
1. make
```

一般需要1-2小时甚至更久，一定要有耐心呦。

d.安装

```
1. make install
```

默认安装/usr/local/Trolltech/目录下。

提示：若在make阶段出现

```
1. /usr/bin/ld: cannot find -lXrender
2. collect2: ld returned 1 exit status
3. make[1]: *** [../../lib/libQtWebKit.so.4.7.3] 错误 1
4. make[1]: Leaving directory `/home/download/qt-everywhere-opensource-src-4.7.3/src/3rdparty/webkit/WebCore'
5. make: *** [sub-webkit-make_default-ordered] 错误 2
```

是因为没有装libxrender-dev，用apt-get安装即可。
若提示

```
1. Makefile:xxx: recipe for target xxx failed 等
```

可以在make及make install后面加上“-i”，即ignore忽略掉错误继续编译。
详情见：[如何忽略makefile执行过程中的某些命令的错误而得以继续运行](#)

e.设置环境变量

```
1. gedit /root/.bashrc
```

在最后一行添加并保存：

```
1. export QTDIR=/usr/local/Trolltech/Qt-4.8.6
2. export PATH=$QTDIR/bin:$PATH
3. export MANPATH=$QTDIR/man:$MANPAT
4. export LD_LIBRARY_PATH=$QTDIR/lib:$LD_LIBRARY_PATH
```

f.测试

```
1. qmake -v
2. 提示：
3. QMake version 2.01a
4. Using Qt version 4.8.6 in /usr/local/Trolltech/Qt-4.8.6/lib
```

安装完成。

3.Qt4.8.6+QtCreator4.0.3配置

第一步：tools>options>Build and Run，然后选择Qt Versions，点击 Browse 选择刚才安装的Qt库，然后点击应用。

第二步：选择Kits，手动添加一个，GCC，GDB会自动检测，这里需要指定Qt Version，默认是None，选择我们刚才添加的Qt 4.8.6。

安装过程参考自：[linux上安装Qt4.8.6+QtCreator4.0.3](#)

三、Linux下Qt5的安装

1.Qt源码及QtCreator下载

Qt4.8.6源码：

http://download.qt.io/official_releases/qt/5.6/5.6.2/
下载【qt-opensource-linux-x64-5.6.2.run】。

QtCreator4.0.3

http://download.qt.io/official_releases/qtcreator/4.6/4.6.2/
下载【qt-creator-opensource-linux-x86_64-4.6.2.run】。

2.安装

找到两个.run文件双击运行或用终端运行即可。

3.配置

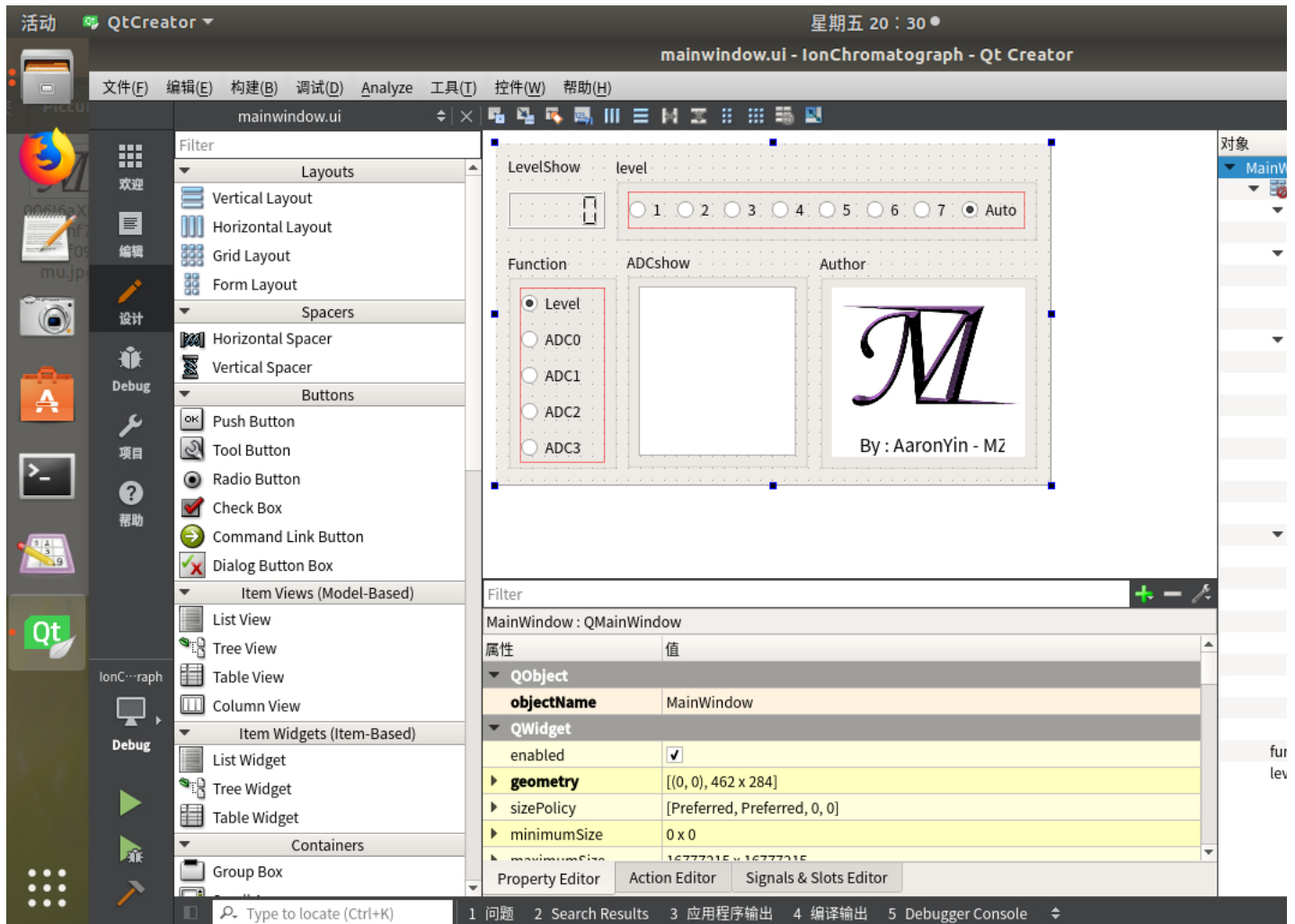
与QT4相同，QT库文件在QT5.6.2->QT5.6->gcc_64->bin->qmake。

四、Qt开发项目基本界面

界面如图：



各控件名称及分组：



1. radioButton&buttonGroup

a.QtCreator建立buttonGroup

选中多个button，右键，指定到按钮组。(建议一个box建立一个组)

b.获取选中的radioButton

方法一：采用对象名称进行获取

详见：[动态取得控件的objectName之后，对名字进行比较](#)

方法二：通过button的ID来获取

对于已经设置在buttonGroup中的button

```
1. ui->buttonGroupA->setId(ui->radioButton0, 0);
2. ui->buttonGroupA->setId(ui->radioButton1, 1);
3. ui->buttonGroupA->setId(ui->radioButton2, 2);
4. ...
```

对于未添加到buttonGroup的button

```
1. buttonGroupA->addButton( ui.radioButton0, 0 );
2. buttonGroupA->addButton( ui.radioButton1, 1 );
3. buttonGroupA->addButton( ui.radioButton2, 2 );
4. ...
```

之后就可以在connect中将选中的buttonID传到槽函数中,在槽函数中便可判断按键了。

```
1. connect( buttonGroupA, SIGNAL(buttonClicked (int)), this, SLOT(buttonChose(int)) );//连接信号和槽
```

2. QLCDNumber

显示数字:

```
1. ui->LCD1->display(要显示的数字或变量);
```

常用接口:

```
1. setDigitCount(int numDigits)设置所显示的位数
2. setBinMode()以二进制形式显示
3. setOctMode()以八进制形式显示
4. setHexMode()以十六进制形式显示
5. setDecMode()以十进制形式显示（默认）
```

简易时钟示例:

```
1. void onTimeOut()
2. {
3.     // 获取系统当前时间
4.     QDateTime dateTime = QDateTime::currentDateTime();
5.     // 显示的内容
6.     m_pLCD->display(dateTime.toString("yyyy-MM-dd HH:mm:ss.zzz"));
7. }
```

3. textBrowser

添加文字:

```
1. ui->textBrowserA->setText(tmp);
2. //或
3. ui->textBrowserA->textCursor().insertText(tmp);//插入文本到光标位置
```

4. QString与int类型转换

QString转int: toInt()

例:

```
1. QString str("100");
2. int tmp = str.toInt();
```

或者:

```
1. bool ok;
2. QString str("100");
3. int tmp = str.toInt(&ok);
4. 注: ok表示转换是否成功,成功则ok为true,失败则ok为false。
```

int转QString

例:

```
1. int tmp = 100;
2. QString str = QString::number(tmp);
```

5. 添加图片

添加图片资源

- 右键工程,添加新文件
- 左边选择QT,右边选择qt资源文件
- 选择文件路径,自己命名资源名,下一步
- 完成之后,就可以看到有一个qrc后缀名的文件,那就是你的资源文件夹了
- 点击添加前缀,再点击添加文件

使背景图自动调整来适应控件的大小

推荐使用border-image。

默认background-image 不会缩放图片以适应控件的大小。

如果要提供一个皮肤或背景图片以自动适应控件大小,必须也只能用border-image属性。

因为border-image已经设置了可用的背景图片,所以使用了border-image后,没必要再指定background-image。

如果同时指定了两个属性,那么将会使用border-image 绘制覆盖掉background-image。

6. 调用可执行程序及终端命令

```
1. 头文件添加stdlib.h
2.
3. system("./pc 192.1.1.1 2W401013");
```

或

```
1. 头文件添加QProcess
2.
3. QProcess sys;
4. QString output;
5. sys.start("./pc 192.168.1.230 2R000003");
6. sys.waitForFinished();
7. output=sys.readAllStandardOutput();//读取传回的所有字符
```

7. 对字符串进行操作

a.使某个字符填满字符串

也就是说字符串里的所有字符都有等长度的ch来代替。

```
1. QString::fill ( QChar ch, int size = -1 )
```

```
1. 例如:
2. QString str = "Berlin";
3. str.fill('z'); // str == "zzzzzz" str.fill('A', 2); // str == "AA"
```

b.从字符串查找相同的字符串

```
1. int QString::indexOf ( const QString & str, int from = 0, Qt::CaseSensitivity cs = Qt::CaseSensitive ) const
```

```
1. 例如:
2. QString x = "sticky question";
3. QString y = "sti";
4. x.indexOf(y); // returns 0 x.indexOf(y, 1); // returns 10 x.indexOf(y, 10); // returns 10 x.indexOf(y, 11); // returns -1
```

c.指定位置插入字符串

```
1. QString & QString::insert ( int position, const QString & str )
```

```
1. 例如:
2. QString str = "Meal";
3. str.insert(1, QString("ontr")); // str == "Montreal"
```

d.判断字符串是否为空

```
1. bool QString::isEmpty () const
```

```
1. 例如:
2. QString().isEmpty(); // returns true QString("").isEmpty(); // returns true QString("x").isEmpty(); // returns false QString("abc")
```

e.判断字符串是否存在

```
1. bool QString::isNull () const
```

```
1. 例如:
2. QString().isNull(); // returns true QString("").isNull(); // returns false QString("abc").isNull(); // returns false
```

f.从左向右截取字符串

```
1. QString QString::left ( int n ) const
```

```
1. 例如:
2. QString x = "Pineapple";
3. QString y = x.left(4); // y == "Pine"
```

g.从中间截取字符串

```
1. QString QString::mid ( int position, int n = -1 ) const
```

```
1. 例如:
2. QString x = "Nine pineapples";
3. QString y = x.mid(5, 4); // y == "pine" QString z = x.mid(5); // z == "pineapples"
```

h.删除字符串中间某个字符

```
1.   QString & QString::remove ( int position, int n )
```

```
1.   例如:
2.       QString s = "Montreal";
3.       s.remove(1, 4); // s == "Meal"
```

i.替换字符串中的某些字符

```
1.   QString & QString::replace ( int position, int n, const QString & after )
```

```
1.   例如:
2.       QString x = "Say yes!";
3.       QString y = "no";
4.       x.replace(4, 3, y); // x == "Say no!"
```

j.以某个字符切割字符串

```
1.   QString QString::section ( QChar sep, int start, int end = -1, SectionFlags flags = SectionDefault ) const
```

```
1.   例如:
2.       QString csv = "forename,middlename,surname,phone";
3.       QString path = "/usr/local/bin/myapp"; // First field is empty QString::SectionFlag flag = QString::SectionSkipEmpty;
4.       QString str = csv.section(',', 2, 2); // str == "surname" str = path.section('/', 3, 4); // str == "bin/myapp" str = path.section('/', 3, 3);
```

k.其他类型转为QString

```
1.   QString & QString::setNum ( uint n, int base = 10 )
```

l.

如果我们希望检查一个字符串是否是以某物开始或结束，我们可以使用startsWith()和endsWith()函数：

```
1.   if (url.startsWith("http:") && url.endsWith(".png"))...
```

这个要比下面的简单快速：

```
1.   if (url.left(5) == "http:" && url.right(4) == ".png")...
```

使用==操作符的字符串比较是大小写敏感的。

如果我们正在比较用户级（user-visible）字符串，localeAwareCompare()经常是正确的选择，并且如果我们希望大小写不敏感，我们可以用toUpper()或toLowerCase()。

```
1.   例如:
2.   if (fileName.toLowerCase() == "readme.txt")...
```

8. 测试PC与ARM正常通信

```
1.   #include <stdio.h>
2.   #include <stdlib.h>
3.   #include <string.h>
4.   #include <sys/types.h>
5.   #include <sys/stat.h>
6.   #include <fcntl.h>
7.   #include <unistd.h>
8.   #include <termios.h>
9.
10.  int main()
11.  {
12.      int fd,led;
13.      char buf[100];
14.      int i=0;
15.
16.      led=open("/dev/leds",O_RDONLY);
17.      if(led<0) exit(1);
18.
19.      fd=open("/dev/adc",O_RDWR);
20.      if(fd<0) exit(1);
21.
22.      while(1)
23.      {
24.          i=read(fd,buf,sizeof(buf)-1);
25.          buf[i]='\0';
26.          sscanf(buf,"%d",&i);
27.
28.          printf("AD=%d\n",i);
29.          sleep(1);
30.
31.          ioctl(led,0,0);//LED control (file,on/off,adress)
32.      }
33.      close(fd);
```

```

34.     close(led);
35.
36.     return 0;
37. }

```

9. socket通信及TCP协议通信源码

PC端

```

1.  #include <stdio.h>
2.  #include <stdlib.h>
3.  #include <errno.h>
4.  #include <string.h>
5.  #include <netdb.h>
6.  #include <sys/types.h>
7.  #include <netinet/in.h>
8.  #include <sys/socket.h>
9.  #define SERVPORT 3333
10. #define MAXDATASIZE 100
11.
12. main(int argc, char *argv[])
13. {
14.     int sockfd, client_fd, client_sockaddr, sendbytes, recvbytes;
15.     int adc, len;
16.     char buf[MAXDATASIZE], buffer[32];
17.     struct hostent *host;
18.     struct sockaddr_in serv_addr;
19.
20.     if(argc < 2)
21.     {
22.         fprintf(stderr, "Please enter the server's hostname!\n");
23.         exit(1);
24.     }
25.     if((host = gethostbyname(argv[1])) == NULL)
26.     {
27.         perror("gethostbyname");
28.         exit(1);
29.     }
30.     if((sockfd = socket(AF_INET, SOCK_STREAM, 0)) == -1)
31.     {
32.         perror("socket");
33.         exit(1);
34.     }
35.
36.     serv_addr.sin_family = AF_INET;
37.     serv_addr.sin_port = htons(SERVPORT);
38.     serv_addr.sin_addr = *((struct in_addr *) host->h_addr);
39.     bzero(&(serv_addr.sin_zero), 8);
40.
41.     if(connect(sockfd, (struct sockaddr *) &serv_addr, sizeof(struct sockaddr)) == -1)
42.     {
43.         perror("connect");
44.         exit(1);
45.     }
46.     if((sendbytes = send(sockfd, argv[2], 8, 0)) == -1)
47.     {
48.         perror("send");
49.         exit(1);
50.     }
51.     /*if((client_fd = accept(sockfd, (struct sockaddr *) &client_sockaddr, &sin_size)) == -1)
52.     {
53.         perror("accept");
54.         exit(1);
55.     }
56.     printf("accept success\n");*/
57.
58.     if(argv[2][1] == 'R')
59.     {
60.         if((recvbytes = recv(sockfd, buffer, MAXDATASIZE, 0)) > 0)
61.         {
62.             buffer[recvbytes] = '\0';
63.             printf("%s\n", buffer);
64.         }
65.         //printf("open adc\n");
66.     }
67.     else if(argv[2][1] == 'W')
68.     {
69.         printf("open led\n");
70.     }
71.
72.     close(sockfd);
73. }

```

ARM端

```

1.  #include <stdlib.h>
2.  #include <errno.h>
3.  #include <string.h>
4.  #include <unistd.h>
5.  #include <netinet/in.h>
6.  #include <fcntl.h>
7.  #include <termios.h>
8.  #include <stdio.h>
9.  #include <sys/types.h>
10. #include <sys/stat.h>
11. #define SERVPORT 3333
12. #define BACKLOG 10
13. #define MAX_CONNECTED_NO 10
14. #define MAXDATASIZE 64
15.

```



```

16.
17. static char RcvBuf[100];
18. static char SendBuf[100];
19. int main()
20. {
21.     struct sockaddr_in server_sockaddr, client_sockaddr;
22.     int sin_size, recvbytes;
23.     int sockfd, client_fd;
24.     char buf[MAXDATASIZE];
25.
26.     if((sockfd=socket(AF_INET, SOCK_STREAM, 0))==-1)
27.     {
28.         perror("socket");
29.         exit(1);
30.     }
31.     printf("socket success! sockfd=%d\n", sockfd);
32.
33.     server_sockaddr.sin_family=AF_INET;
34.     server_sockaddr.sin_port=htons(SERVPORT);
35.     server_sockaddr.sin_addr.s_addr=INADDR_ANY;
36.     bzero(&(server_sockaddr.sin_zero), 8);
37.
38.     if(bind(sockfd, (struct sockaddr *)&server_sockaddr, sizeof(struct sockaddr))==-1)
39.     {
40.         perror("bind");
41.         exit(1);
42.     }
43.     printf("bind success!\n");
44.     if(listen(sockfd, BACKLOG)==-1)
45.     {
46.         perror("listen");
47.         exit(1);
48.     }
49.     printf("listen success\n");
50.     if((client_fd=accept(sockfd, (struct sockaddr *)&client_sockaddr, &sin_size))==-1)
51.     {
52.         perror("accept");
53.         exit(1);
54.     }
55.     printf("accept success\n");
56.
57.     while(1)
58.     {
59.         if((recvbytes=recv(client_fd, buf, MAXDATASIZE, 0))>0)
60.         {
61.             buf[recvbytes]='\0';
62.             printf("Receive:%s\n", buf);
63.
64.             HandleTCP(buf, recvbytes);
65.             //printf("222\n");
66.             send(client_fd, SendBuf, 8, 0);
67.             //printf("333\n");
68.         }
69.
70.         else if(recvbytes<=0)
71.         {
72.             printf("Connection closed!\n");
73.             close(client_fd);
74.             client_fd=accept(sockfd, (struct sockaddr *)&client_sockaddr, &sin_size);
75.         }
76.     }
77.
78.     close(sockfd);
79. }
80.
81. void HandleTCP(char *bf, int len)
82. {
83.     int i, RcvPtr=0;
84.     for(i=0; i<len; i++)
85.     {
86.         if(*(bf+i)=='2')
87.         {
88.             //printf("b1\n");
89.             RcvBuf[0]='2';
90.             RcvPtr=1;
91.         }
92.         else if(*(bf+i)=='3')
93.         {
94.             //printf("c1\n");
95.             if(RcvPtr==7) ParseFrame();
96.             //printf("clend\n");
97.             RcvPtr=0;
98.         }
99.         else
100.        {
101.            //printf("ptr %d\n", RcvPtr);
102.            if(RcvPtr>0) RcvBuf[RcvPtr++]=*(bf+i);
103.            //printf("bf i %c\n", *(bf+i));
104.        }
105.    }
106. }
107.
108. void ParseFrame()
109. {
110.     int adc, led;
111.     int d, len;
112.     char buffer[32];
113.
114.     SendBuf[0]='2';
115.     if(RcvBuf[1]=='R')
116.     {
117.         SendBuf[1]='B';
118.     }
119.

```

```

120.     SendBuf[2]=RcvBuf[2];
121.     switch(RcvBuf[2])
122.     {
123.     case '0':
124.         adc=open("/dev/adc",0);
125.         if(adc<0) SendBuf[1]='E';
126.         else
127.         {
128.             len=read(adc,buffer,sizeof(buffer)-1);
129.             sscanf(buffer,"%d",&d);
130.             sprintf(&SendBuf[3],"%04d",d);
131.             close(adc);
132.         }
133.         SendBuf[7]='3';
134.         break;
135.     case '1':
136.         adc=open("/dev/adc",1);
137.         if(adc<0) SendBuf[1]='E';
138.         else
139.         {
140.             len=read(adc,buffer,sizeof(buffer)-1);
141.             sscanf(buffer,"%d",&d);
142.             sprintf(&SendBuf[3],"%04d",d);
143.             close(adc);
144.         }
145.         SendBuf[7]='3';
146.         break;
147.     case '2':
148.         adc=open("/dev/adc",2);
149.         if(adc<0) SendBuf[1]='E';
150.         else
151.         {
152.             len=read(adc,buffer,sizeof(buffer)-1);
153.             sscanf(buffer,"%d",&d);
154.             sprintf(&SendBuf[3],"%04d",d);
155.             close(adc);
156.         }
157.         SendBuf[7]='3';
158.         break;
159.     case '3':
160.         adc=open("/dev/adc",3);
161.         if(adc<0) SendBuf[1]='E';
162.         else
163.         {
164.             len=read(adc,buffer,sizeof(buffer)-1);
165.             sscanf(buffer,"%d",&d);
166.             sprintf(&SendBuf[3],"%04d",d);
167.             close(adc);
168.         }
169.         SendBuf[7]='3';
170.         break;
171.     }
172. }
173. else if(RcvBuf[1]=='W')
174. {
175.     if(RcvBuf[2]=='4')
176.     {
177.         led=open("/dev/leds",O_RDONLY);
178.         //RcvBuf[6]-0x30;
179.         if(RcvBuf[3]=='1') ioctl(led,0,3);
180.         else if(RcvBuf[3]=='0') ioctl(led,1,3);
181.         if(RcvBuf[4]=='1') ioctl(led,0,2);
182.         else if(RcvBuf[4]=='0') ioctl(led,1,2);
183.         if(RcvBuf[5]=='1') ioctl(led,0,1);
184.         else if(RcvBuf[6]=='0') ioctl(led,1,1);
185.         if(RcvBuf[6]=='1') ioctl(led,0,0);
186.         else if(RcvBuf[6]=='0') ioctl(led,1,0);
187.
188.         close(led);
189.     }
190. }
191. }

```

10.QT界面代码

mainwindow.cpp

```

1.  #include "mainwindow.h"
2.  #include "ui_mainwindow.h"
3.  #include "QDebug"
4.  #include "QProcess"
5.
6.  MainWindow::MainWindow(QWidget *parent):
7.      QMainWindow(parent),
8.      ui(new Ui::MainWindow)
9.  {
10.     ui->setupUi(this);
11.     setButtonId();
12.     connect(ui->levelGroup,SIGNAL(buttonClicked(int)),this,SLOT(levelChose(int)));
13.     connect(ui->funcGroup,SIGNAL(buttonClicked(int)),this,SLOT(funcChose(int)));
14. }
15.
16. MainWindow::~MainWindow()
17. {
18.     delete ui;
19. }
20.
21.
22. void MainWindow::setButtonId()
23. {
24.     ui->levelGroup->setId(ui->level1,1);

```

```

25.     ui->levelGroup->setId(ui->level2,2);
26.     ui->levelGroup->setId(ui->level3,3);
27.     ui->levelGroup->setId(ui->level4,4);
28.     ui->levelGroup->setId(ui->level5,5);
29.     ui->levelGroup->setId(ui->level6,6);
30.     ui->levelGroup->setId(ui->level7,7);
31.     ui->levelGroup->setId(ui->levelAuto,8);
32.     //qDebug()<<LevelNumber;
33.     levelNumber=8;
34.     //qDebug()<<LevelNumber;
35.
36.     ui->ADCshow->setEnabled(false);
37.
38.     ui->funcGroup->setId(ui->levelChoseButton,0);
39.     ui->funcGroup->setId(ui->ADC0,10);
40.     ui->funcGroup->setId(ui->ADC1,11);
41.     ui->funcGroup->setId(ui->ADC2,12);
42.     ui->funcGroup->setId(ui->ADC3,13);
43.     //qDebug()<<funcNumber;
44.     funcNumber=0;
45.     //qDebug()<<funcNumber;
46. }
47.
48. void MainWindow::levelShow()
49. {
50.     ui->lcdLevelNumber->display(levelNumber);
51. }
52.
53. void MainWindow::levelChose(int levelTT)
54. {
55.     levelNumber=levelTT;//ui->levelGroup->checkedId();
56.     //qDebug()<<levelNumber;
57.     levelShow();
58.     TCP(levelNumber);
59. }
60.
61. void MainWindow::funcChose(int funcTT)
62. {
63.     funcNumber=funcTT;
64.     //qDebug()<<funcNumber;
65.     if (funcNumber==0)
66.     {
67.         ui->levelBox->setEnabled(true);
68.         ui->ADCshow->setEnabled(false);
69.         ADC_Data(0);
70.     }
71.     else if (funcNumber==10)
72.     {
73.         ui->ADCshow->setEnabled(true);
74.         ui->levelBox->setEnabled(false);
75.         ADC_Data(0);
76.         ADC_Data(10);
77.     }
78.     else if (funcNumber==11)
79.     {
80.         ui->ADCshow->setEnabled(true);
81.         ui->levelBox->setEnabled(false);
82.         ADC_Data(0);
83.         ADC_Data(11);
84.     }
85.     else if (funcNumber==12)
86.     {
87.         ui->ADCshow->setEnabled(true);
88.         ui->levelBox->setEnabled(false);
89.         ADC_Data(0);
90.         ADC_Data(12);
91.     }
92.     else if (funcNumber==13)
93.     {
94.         ui->ADCshow->setEnabled(true);
95.         ui->levelBox->setEnabled(false);
96.         ADC_Data(0);
97.         ADC_Data(13);
98.     }
99. }
100.
101. void MainWindow::ADC_Data(int ADCTT)
102. {
103.     switch (ADCTT) {
104.     case 0:
105.         ui->ADCText->clear();
106.         break;
107.     /*case 10:
108.         for(int i=0;i<10;i++)
109.             ui->ADCText->textCursor().insertText("ADC=0000"+QString::number(i)+"\n");
110.         break;
111.     case 11:
112.         for(int i=0;i<10;i++)
113.             ui->ADCText->textCursor().insertText("ADC=1100"+QString::number(i)+"\n");
114.         break;
115.     case 12:
116.         for(int i=0;i<10;i++)
117.             ui->ADCText->textCursor().insertText("ADC=2200"+QString::number(i)+"\n");
118.         break;
119.     case 13:
120.         for(int i=0;i<10;i++)
121.             ui->ADCText->textCursor().insertText("ADC=3300"+QString::number(i)+"\n");
122.         break;*/
123.     default:
124.         for(int i=0;i<10;i++)
125.         {
126.             QString rTT=TCP(10);
127.             ui->ADCText->textCursor().insertText(rTT+"\n");
128.             //qDebug()<<rTT;

```

```

129.     }
130.
131.     break;
132. }
133. }
134.
135. QString MainWindow::TCP(int sBuf)
136. {
137.     QString output;
138.     QString rBuf;
139.     QProcess sys;
140.
141.     switch (sBuf) {
142.     case 1:
143.         system("./pc 192.168.1.230 2W411103");
144.         break;
145.     case 2:
146.         system("./pc 192.168.1.230 2W411013");
147.         break;
148.     case 3:
149.         system("./pc 192.168.1.230 2W411003");
150.         break;
151.     case 4:
152.         system("./pc 192.168.1.230 2W410113");
153.         break;
154.     case 5:
155.         system("./pc 192.168.1.230 2W410103");
156.         break;
157.     case 6:
158.         system("./pc 192.168.1.230 2W410013");
159.         break;
160.     case 7:
161.         system("./pc 192.168.1.230 2W410003");
162.         break;
163.     case 8:
164.         system("./pc 192.168.1.230 2W401113");
165.         break;
166.     case 10:
167.     {
168.         sys.start("./pc 192.168.1.230 2R000003");
169.         sys.waitForFinished();
170.         output=sys.readAllStandardOutput();
171.         rBuf=output.mid(3,4);
172.         qDebug()<<rBuf;
173.         break;
174.     }
175.     default:
176.         break;
177.     }
178.
179.     return rBuf;
180. }

```

11. 调用程序的编译及上传运行

```

1.  gcc TCPIP_PC.cpp -o pc
2.  //在PC的Linux上编译TCPIP_PC.cpp，并将编译出的pc文件放到QT编译出的Debug目录下。
3.
4.  arm-linux-gcc TCPIP_ARM.cpp -o arm
5.  //在PC的Linux上用交叉编译TCPIP_ARM.cpp。
6.
7.  ftp 192.168.1.230//此IP为ARM开发板上的IP
8.  //Login...
9.
10. ftp>put arm
11. //将编译出的arm可执行文件上传到arm根目录下。
12.
13. ftp>Bye
14. //退出ftp
15.
16. telnet 192.168.1.230
17. //与ARM开发板的Linux系统进行连接。
18.
19. ./arm
20. //运行ARM开发板上的arm程序，进行侦听pc的连接。
21.
22. //之后打开QT界面进行操作及通信。

```

12. ARM开发板上的程序开机启动

在PC上用telnet连接ARM，打开etc/init.d/rcS文件，添加./arm即可。

注：若用gedit打开，添加完直接保存退出即可，若用vi打开，添加完后esc，用:wq保存并退出。

附：vi指令

a.打开文件：

```

1.  # vi /etc/my.cnf    //打开或新建文件，并将光标至于第一行首
2.  # vi + /etc/my.cnf  //打开文件，并将光标移至最后一行行首
3.  # vi +n /etc/my.cnf //打开文件，并将光标置于第n行首
4.  # vi +/pattern filename //打开文件，并将光标置于第一个与pattern匹配的串处

```

b.按键盘上“a”键，vi界面出现 INSERT后，开始进行编辑操作

c.编辑完毕后，按ESC键，跳到命令模式，然后进行保存退出或不保存退出操作：

```
1. :w //保存, 不退出vi
2. :w! //强制保存, 不退出vi
3. :w file //将修改另外保存到file中, 但不退出vi (不常用)
4. :wq 或 :x //保存, 并退出vi
5. :wq! //强制保存, 并退出vi
6.
7. **d.下面是不保存的相关命令**
8. :q //不保存, 并退出vi
9. :q! //不保存, 并强制退出vi
10.
11. :e! //放弃所有修改, 从上次保存文件开始再编辑
```

13. QSS相关

详见: [Qt之QSS（白色靓丽）](#)

14. 自动进样器代码

[GitHub地址: Autosampler](#)

```
1. #include <STC89C52RC.h>
2.
3. //P20-P1.0转盘脉冲
4. //P21-P1.1转盘方向
5. //P22-P1.2针臂脉冲
6. //P23-P1.3针臂方向
7. //P25-P0.1下限位
8. //P26-P0.2上限位
9. //P27-P0.0霍尔传感器
10.
11.
12. void delay()
13. {
14.     TMOD=0x01;
15.     TH0=0xac; //Speed
16.     TL0=0xf0;
17.     TF0=0;
18.     TR0=1;
19.     while(TF0==0);
20.     TR0=0;
21. }
22.
23.
24. void delayo()
25. {
26.     int i;
27.     for(i=0;i<2600;i++){
28.
29.
30.
31. void updown()
32. {
33.     int i=0;
34.
35.     P23=1;
36.
37.     for(i=0;i<430;i++)
38.     {
39.         P22=1;
40.         delayo();
41.         P22=0;
42.         delayo();
43.     }
44.
45.     P23=0;
46.
47.     for(i=0;i<430;i++)
48.     {
49.         P22=1;
50.         delayo();
51.         P22=0;
52.         delayo();
53.     }
54. }
55.
56.
57. void runo()
58. {
59.     P20=1;
60.     delayo();
61.     P20=0;
62.     delayo();
63. }
64.
65.
66. void run()
67. {
68.     P20=1;
69.     delay();
70.     P20=0;
71.     delay();
72. }
73.
74.
75. main()
76. {
77.     int i;
78.     int fi;
79.     int T24=100;
```

```


80. bit pf=0;
81. TMOD=0x21;
82. TH1=0xe8;
83. TL1=0xe8;    //1200波特率12MHz
84. TR1=1;
85. SCON=0x50;
86. TI=1;
87.
88.
89. while(0)
90. {
91.     // 转盘针臂
92.     if(P27||T24!=0)
93.     {
94.         T24--;
95.         run();
96.     }
97.     if(P27) pf=1;
98.     else if(P27==0)
99.     {
100.         if(pf==1)
101.         {
102.             pf=0;
103.             T24=23;
104.         }
105.         if(pf==0&&T24==0)
106.         {
107.             updown();
108.             T24=8;//goto loop;
109.         }
110.     }
111. }
112.
113.
114. //串口通讯
115. /*for(i=0;i<10;i++)
116. {
117.     while(TI=0);
118.     TI=0;
119.     SBUF=i;        // 发送i
120. }*/
121. /*if(RI)          // 接收到字符
122. {
123.     i=SBUF;        // 接收i
124.     RI=0;          // 接收标志位RI
125.     TI=0;          // 发送结束标志位TI
126.     SBUF=i;        // 发送i
127.     while(TI==0);    // 等待发送完毕
128. }*/
129.
130.
131.
132.
133. //四位控制
134. while(1){
135.     loop:if(RI)
136.     {
137.         fi=SBUF;
138.         RI=0;
139.
140.
141.         if(fi==0)        //Reset
142.         {
143.             P21=1;
144.             T24=100;
145.             while(1)
146.             {
147.                 if(P27||T24!=0)
148.                 {
149.                     T24--;
150.                     runo();
151.                 }
152.                 if(P27) pf=1;
153.                 else if(P27==0)
154.                 {
155.                     if(pf==1)
156.                     {
157.                         pf=0;
158.                         T24=23;
159.                     }
160.                     if(pf==0&&T24==0)
161.                     {
162.                         updown();
163.                         T24=8;
164.                         if(RI)
165.                             goto loop;
166.                     }
167.                 }
168.             }
169.         }
170.         if(fi==1)        // 转换转的方向
171.         {
172.             P21=~P21;
173.             while(1)
174.             {
175.                 runo();
176.                 if(RI)
177.                     goto loop;
178.             }
179.         }
180.         if(fi==2)        // 停止
181.         {
182.             P20=0;
183.             goto loop;

```

```
184.         }
185.         if(fi==3)           //继续
186.         {
187.             while(1)
188.             {
189.                 runo();
190.                 if(RI)
191.                     goto loop;
192.             }
193.         }
194.         if(fi==4)           //抽取
195.         {
196.             P20=0;
197.             updown();
198.             goto loop;
199.         }
200.     }
201. }
202. }
```

- 嵌入式 1
 - 欢迎使用 Cmd Markdown 编辑阅读器
- 搜索我的文稿标题，* 显示全 [如何分类](#)
- 以下【标签】将用于标记这篇文稿：
- [下载客户端](#)
- [变更历史](#)
- [关注开发者](#)
- [报告问题，建议](#)
- [联系我们](#)
- [Web 端版本](#)

-
-
-

 正在加载文章图片，请稍等片刻...

添加新批注



 保存  取消



 保存  取消



 修改  保存  取消  删除

- 私有
- 公开
- 删除

 查看更早的 5 条回复

回复批注



通知

 取消  确认

- 
- 