

Crossbow Application Note

——如何在 Labview 下建立基于 Xserve 的无线传感器网络终端程序。

秦飞

Sr.Application Engineer

Crossbow 北京代表处

一：介绍

Crossbow 公司提供了 Moteworks™ 三层软件架构用以加速客户的无线传感器网络产品研发周期。

MoteWorks™ 是第一款用于工业的可开放源代码的，基于标准平台和支持 OEM 设备与系统开发的软件平台。此软件平台不仅支持多种无线传感器，还支持多种网络拓扑结构，如星型、混合型和 Mesh。

MoteWorks™ 的灵活性和可选性帮助开发人员选择最好的网络拓扑结构，电源管理模式以及应用带宽。特别适用于低功耗操作的网络。

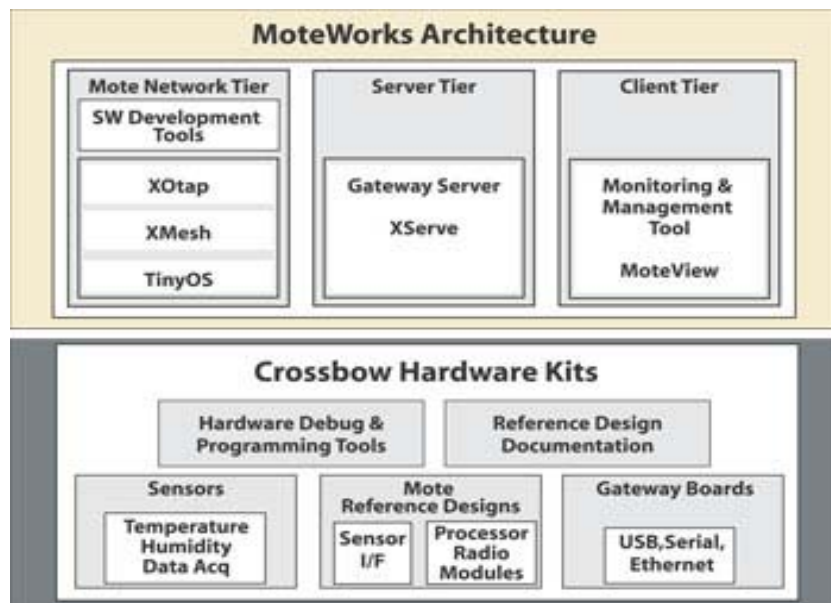


图 1

Moteworks™ 将无线传感器网络划分为：Mote Tier, Server Tier, 以及 Client Tier。

其中 Mote Tier 为运行在无线传感器网络节点上的相关程序，以 Xmesh™ 多跳自组织协议为核心，硬件底层驱动为基础，完成采集控制无线传感器网络的目的。

Mote Tier 采集并将通过网关接口板（MIB510, 520, 600）将相关网络数据传至网络层 Server Tier 处理。Crossbow 在 Server Tier 通过 Xserver™ 提供包括 CSV, XML, Database, Modbus 等多种标准通用数据接口：

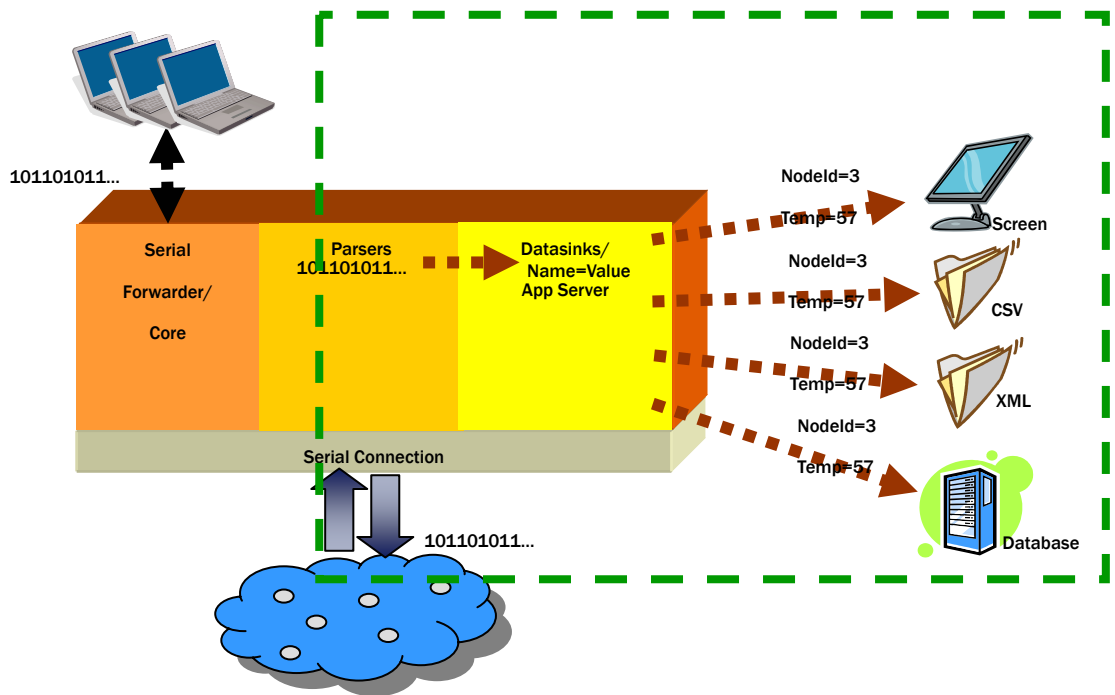


图 2

第三层，Client Tier 则为用户监测管理层，crossbow 提供了完全免费 Moteview™ 作为默认的管理软件。

除此之外，考虑到客户的需求，工程开发类的用户可以非常方便的通过 Xserver™ 提供的接口开发自定义的无线传感器网络监测管理程序。

通过 Xserver™ 的作用，传统的工程用户可以将前端无线传感器网络视作黑盒处理，仅仅通过读取数据库或者 XML socket 等获取实时的无线传感器网络数据，并非常方便的添加至用户原有的信息管理系统。

本文以 LabView 为例，介绍了一种通过读取 XML 信息而实现的无线传感器网络自定义管理界面的设计方法。Labview 是由美国 NI 公司提供的图形化自动测量编程工具。可以非常方便的连接各种数据源并以图形化形式显示。

本文所述方法对其他自定义接口软件的开发同样具有普遍意义。

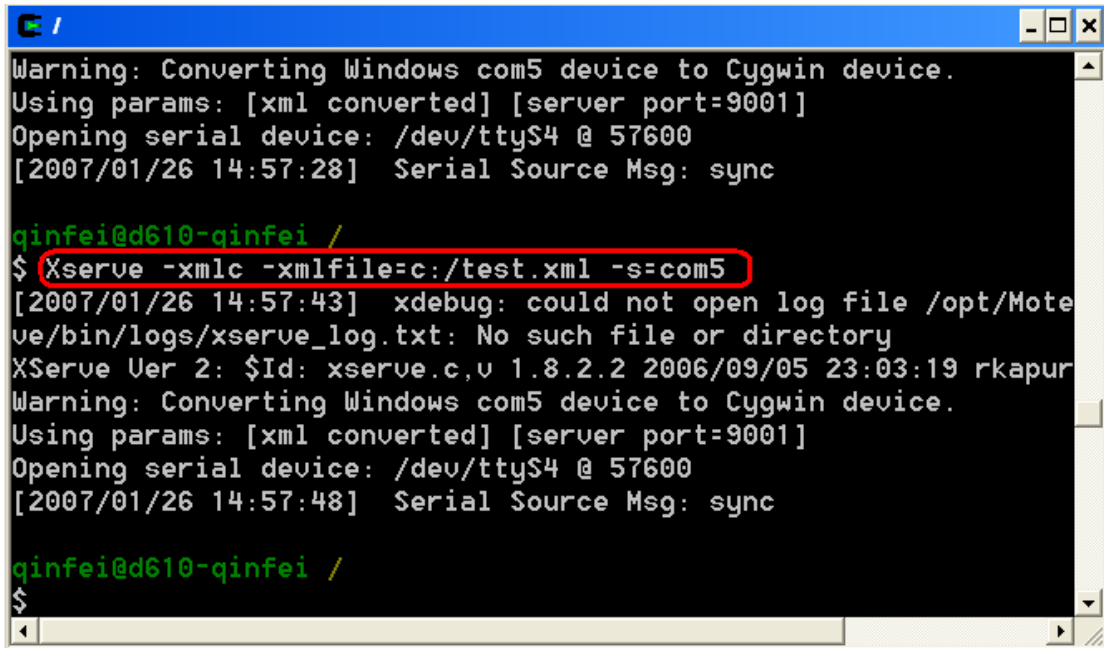
二：连接

在本文中，我们采用 Xserve™ 输出的 XML 接口作为数据源。配置 Xserver 工作参数如下：

Usage: xserve <-[r a p c xr xp xc dbxml r xmlp xmlc v alert m]>

表 1

通过输入以下命令可以实现配置 Xserve 工作的 XML 方式下：



```
Warning: Converting Windows com5 device to Cygwin device.
Using params: [xml converted] [server port=9001]
Opening serial device: /dev/ttyS4 @ 57600
[2007/01/26 14:57:28] Serial Source Msg: sync

qinfei@d610-qinfei /
$ Xserve -xmlc -xmlfile=c:/test.xml -s=com5
[2007/01/26 14:57:43] xdebug: could not open log file /opt/Mote
ue/bin/logs/xserve_log.txt: No such file or directory
XServe Ver 2: $Id: xserve.c,v 1.8.2.2 2006/09/05 23:03:19 rkapur
Warning: Converting Windows com5 device to Cygwin device.
Using params: [xml converted] [server port=9001]
Opening serial device: /dev/ttyS4 @ 57600
[2007/01/26 14:57:48] Serial Source Msg: sync

qinfei@d610-qinfei /
$
```

图 3

在这里 `-xmlc` 表示输出转换过的 XML 格式数据，`-xmlfile` 指向了 xml 的目标文件为 `c:/test.xml`（我们可以通过 `-xmlport` 将数据通过 socket 转发，本文下一部分将讲述相关内容），`-s` 指定了网关接口板的设备端口。

我们可以看到 Xserve 目前已经接收到了一个从网关接口板发来的同步信号。因此 TEST.XML 里已经被填充了相应内容：

```
<?xml version="1.0" ?>
<MotePacket>
<ParsedDataElement>
  <Name>amtype</Name>
  <ConvertedValue>253</ConvertedValue>
</ParsedDataElement>
</MotePacket>
```

表 2

这里我们可以看到一个最简单的 XML 文件。它中间只包含了一个数据对 `AmType=253`。表明当前收到的数据包为同步信息(MoteView 里的 hart beat 信号)。

我们可以通过 labview 来读取 XML 文件。Labview 提供了如下图所示的对应函数库。

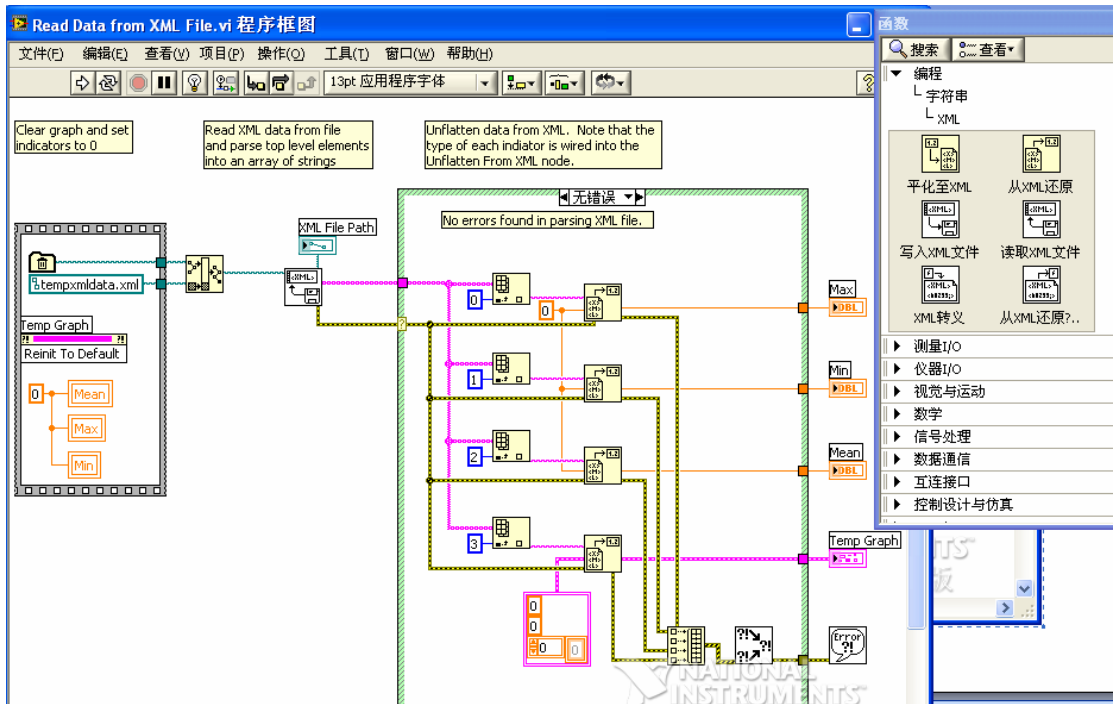


图 4

当我们打开读取 XML 文件的 VI 后，其程序框图如下：

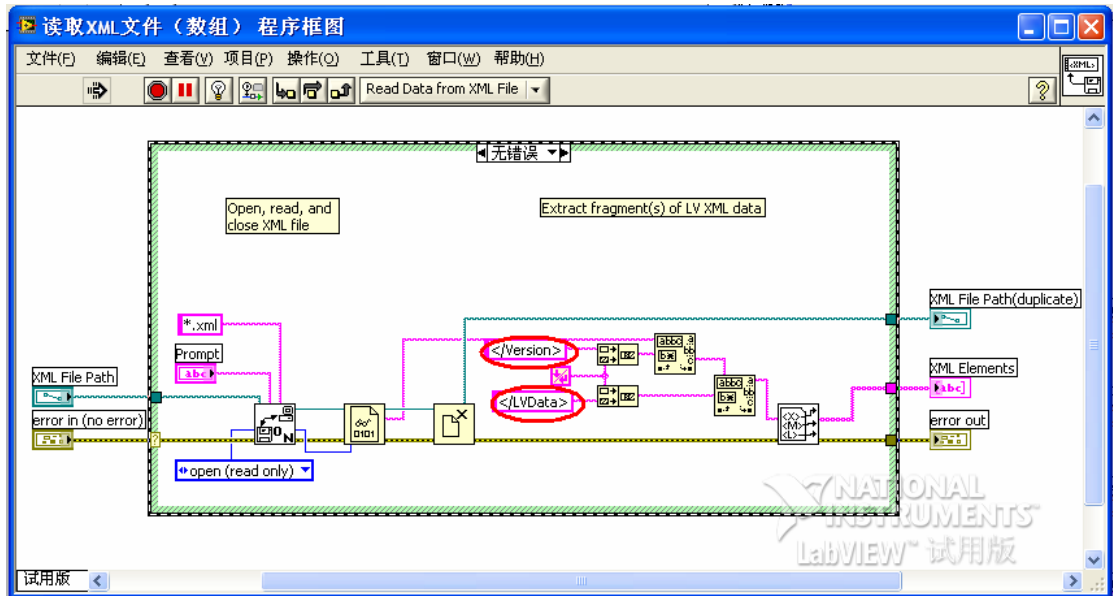


图 5

但是这里存在一个问题，直接调用 Labview 提供的 XML 读取函数，我们会发现程序无法解析出任何数据信息。其原因在于 Labview 提供的 XML 函数库是针对 Labview 自定义数据格式的。以<LvData> 与</LvData>为标识。而按表 2 所示 crossbow 公司 XServer 则以 <MotePacket>为标识，自然导致我们无法通过现有的 Labview 函数库实现读取功能。

观察程序框图，我们发现<LvData>被连接至匹配模式函数：

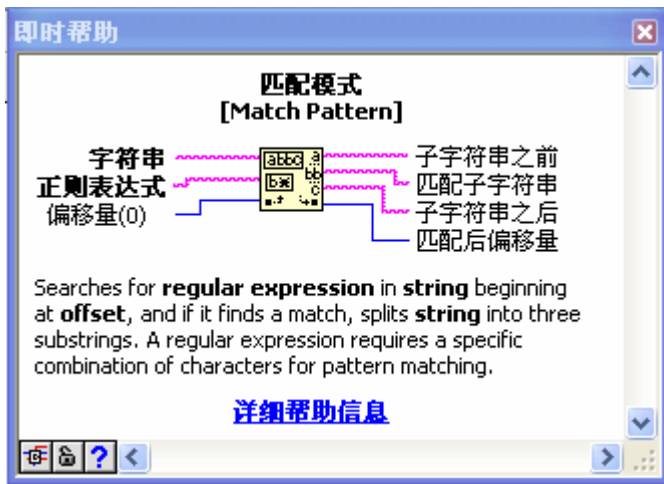


图 6

该函数作用为在字符串中查找目标字符串，并截取其之前或之后的字符串输出。因此该段代码的意义在于将 XML 原文件读取，并给出<LvData>与</LvData>之间，实际标识数据的内容。

因此我们需要进行修改如下：

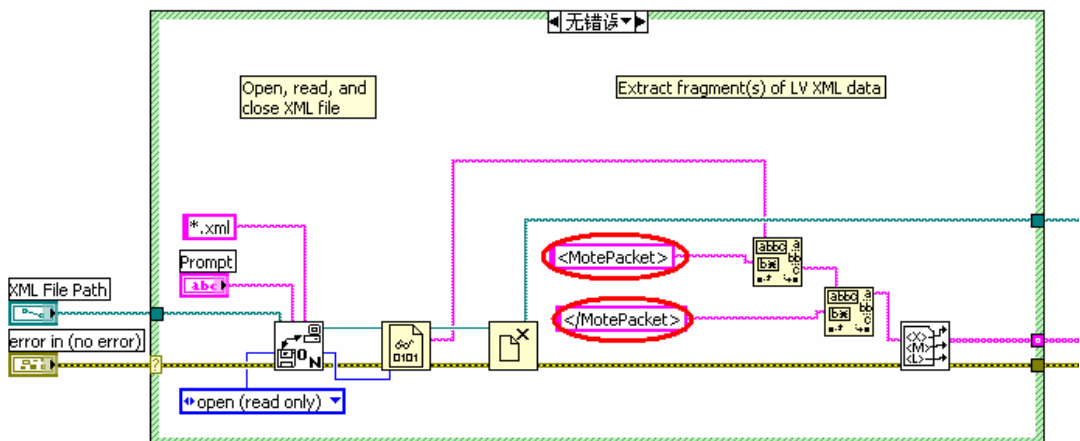


图 7

如上图所示，我们用<MotePacket>与</MotePacket>替换了原有的<LvData>与</LvData>的组合。这样，程序将能认出 Crossbow 标识的 XML 文件，并将其中的数据部分取出。

 则将 XML 数据部分解析成数组形式供后序程序调用。

```
<?xml version="1.0" ?>
<MotePacket>
  <ParsedDataElement>
    <Name>amtype</Name>
    <ConvertedValue>11</ConvertedValue>
  </ParsedDataElement>
  <ParsedDataElement>
    <Name>nodeid</Name>
    <ConvertedValue>1</ConvertedValue>
  </ParsedDataElement>
  <ParsedDataElement>
    <Name>parent</Name>
    <ConvertedValue>0</ConvertedValue>
  </ParsedDataElement>
  <ParsedDataElement>
    <Name>group</Name>
    <ConvertedValue>18</ConvertedValue>
  </ParsedDataElement>
  <ParsedDataElement>
    <Name>socketid</Name>
    <ConvertedValue>51</ConvertedValue>
  </ParsedDataElement>
  <ParsedDataElement>
    <Name>board_id</Name>
    <ConvertedValue>129</ConvertedValue>
  </ParsedDataElement>
  <ParsedDataElement>
    <Name>packet_id</Name>
    <ConvertedValue>134</ConvertedValue>
  </ParsedDataElement>
  <ParsedDataElement>
    <Name>voltage</Name>
    <ConvertedValue>2619</ConvertedValue>
  </ParsedDataElement>
  <ParsedDataElement>
    <Name>humid</Name>
    <ConvertedValue>19.898909</ConvertedValue>
  </ParsedDataElement>
  <ParsedDataElement>
    <Name>humtemp</Name>
    <ConvertedValue>26.480000</ConvertedValue>
  </ParsedDataElement>
  <ParsedDataElement>
    <Name>adc0</Name>
  </ParsedDataElement>
</MotePacket>
```

```
<ConvertedValue>322.086957</ConvertedValue>
</ParsedDataElement>
<ParsedDataElement>
  <Name>adc1</Name>
  <ConvertedValue>264.500000</ConvertedValue>
</ParsedDataElement>
<ParsedDataElement>
  <Name>adc2</Name>
  <ConvertedValue>4095</ConvertedValue>
</ParsedDataElement>
<ParsedDataElement>
  <Name>dig10</Name>
  <ConvertedValue>1</ConvertedValue>
</ParsedDataElement>
<ParsedDataElement>
  <Name>dig11</Name>
  <ConvertedValue>1</ConvertedValue>
</ParsedDataElement>
<ParsedDataElement>
  <Name>dig12</Name>
  <ConvertedValue>1</ConvertedValue>
</ParsedDataElement>
</MotePacket>
```

表 3

表 3 给出了一个包含实际数据内容的 Xserver XML 数据文件（Xserver 转换后 MDA300 数据包）。

得到 XML 数据元素之后，我们需要将数据对解析出来：

```
<ParsedDataElement>
  <Name>amtype</Name>
  <ConvertedValue>11</ConvertedValue>
</ParsedDataElement>
```

表 4

举例而言，我们需要从表 4 所示的字符串中将 amtype: : 11 的数据对解析出来。同样的 LabView 提供的函数：从 xml 平滑至数据。因为标识不同我们无法直接使用，因此参照上文我们采用字符串匹配函数将需要的数据解析出来：

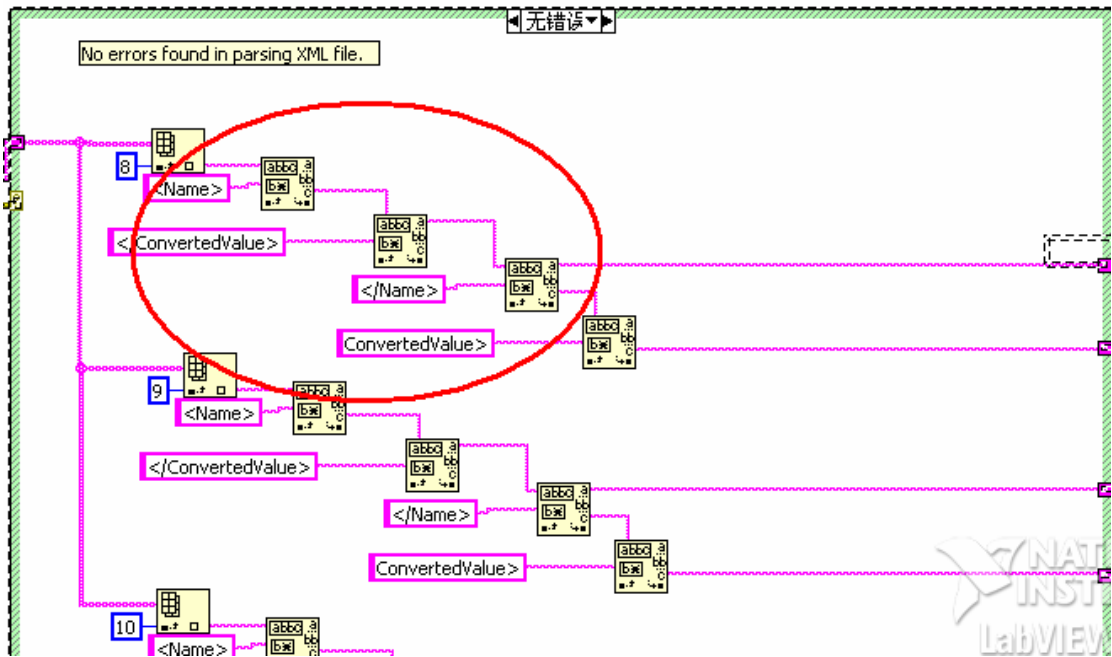


图 8

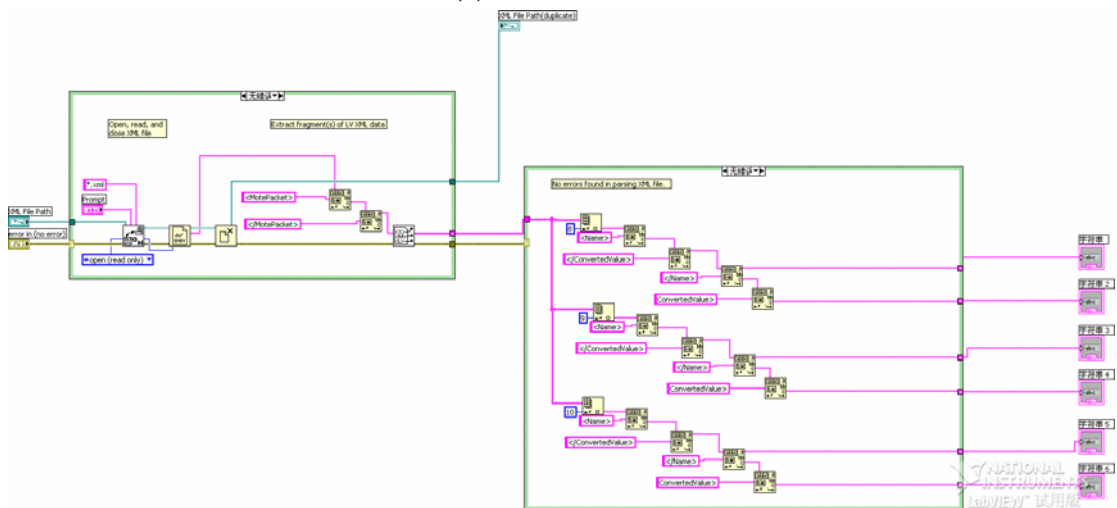


图 9

这样我们就得到了如图 9 所示的一个全功能的 LabView 程序，它实现了从指定 XML 文件读取一次数据，并且显示指定内容的简单功能。

我们在 CygWin 窗口下输入：

```

qinfei@d610-qinfei ~
$ xserve -c -xmlc -xmlfile=c:/test.xml -s=com5
[2007/01/31 11:10:33] xdebug: could not open log file /opt/Mote
ue/bin/logs/xserve_log.txt: No such file or directory
XServe User 2: $Id: xserve.c,v 1.8.2.2 2006/09/05 23:03:19 rkapur
Warning: Converting Windows com5 device to Cygwin device.
Using params: [converted] [xml converted] [server port=9001]
Opening serial device: /dev/ttyS4 @ 57600
[2007/01/31 11:10:36] Serial Source Msg: sync
[2007/01/31 11:10:36] MDA300 [sensor data converted to engineeri
health:    node id=257 parent=0 battery=2872 mU
adc0=322.086957%
adc1=264.500000%
adc2=4095 F
  
```

图 10

然后运行图 9 所示的程序（从附件种可以找到 xml1.vi）：

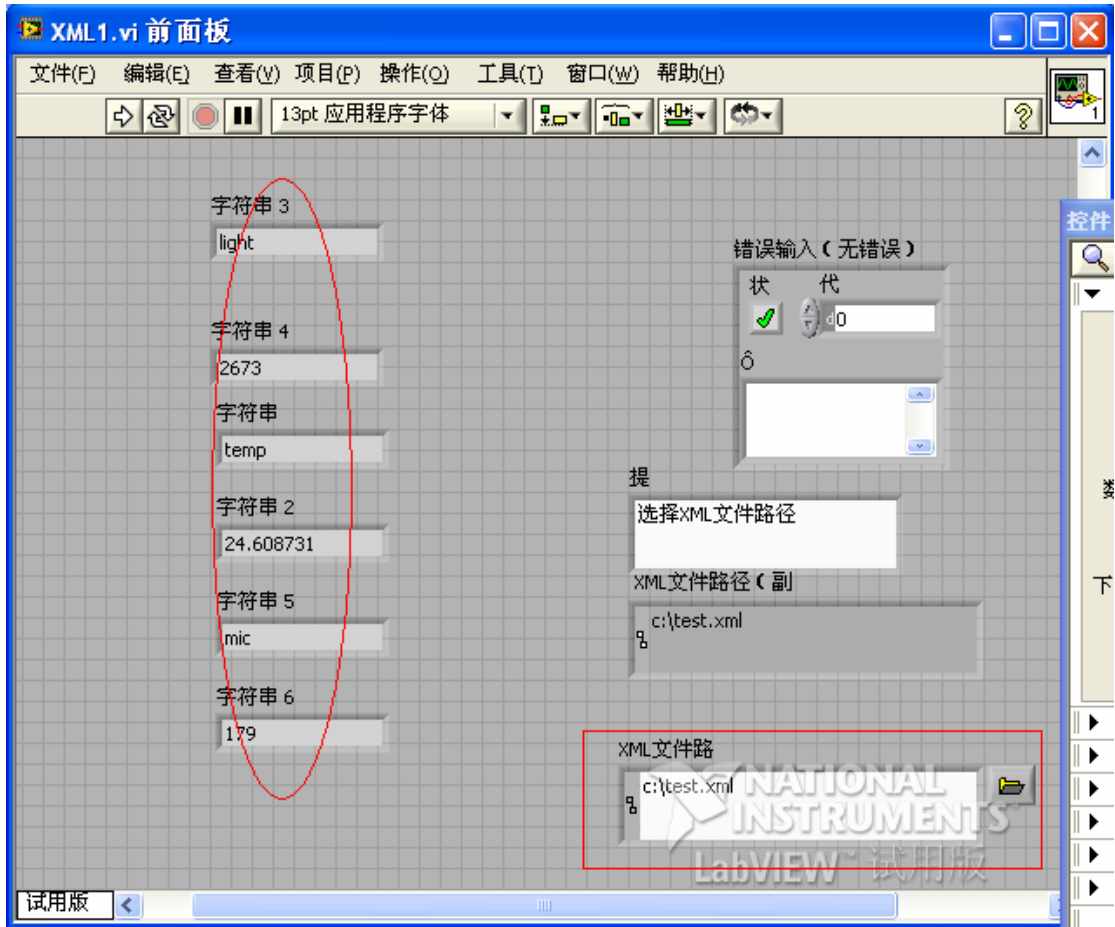


图 11

在 XML 文件路径中输入指定的 XML 文件，在这里输入 CygWin 中指定的 c:\test.xml。然后运行，可以看到左边取出了相应的传感器值。

Light::2673
temp::24.608731
mic::179

表 5

三：网络连接

在上一章中，我们成功的通过 xml 文件方式连接到 xserver 中间件，获取并显示了无线传感器网络的数据。

但是这解决了有无的问题，文件方式在实际工程中并不实用，因为程序若通过文件方式读取，由于无法得知数据何时更新，必需以定时查询方式工作，而无线传感器网络的数据上传速率并不保持一定，极其可能产生漏数据或者过采样问题。且文件接口方式由于其读写权限问题，硬盘访问问题，工作稳定度较差。

为了保证实际工作的需要，我们需要使用一种更加科学的方式来连接到 Xserver 中间件服务器：SOCKET。Xserver 中间件服务器可以配置成基于 TCP/IP 协议的 XML 数据流输出方式。通过输入：

```

[2007/09/20 15:59:29] Simple End Demo
node,1,parent,0,568828,5972,496,495,488,487
[2007/09/20 15:59:29] Simple End Demo
node,1,parent,0,568848,5972,486,479,474,470
[2007/09/20 15:59:29] Simple End Demo
node,1,parent,0,568868,5972,467,466,468,469
[2007/09/20 15:59:29] Simple End Demo
node,1,parent,0,568888,5972,475,486,509,479
[2007/09/20 15:59:29] Simple End Demo
node,1,parent,0,568908,5972,487,485,483,485
[2007/09/20 15:59:29] Simple End Demo
node,1,parent,0,568928,5972,492,497,498,495
[2007/09/20 15:59:29] Simple End Demo
node,1,parent,0,568948,5972,496,496,495,493
[2007/09/20 15:59:31] antype=253,

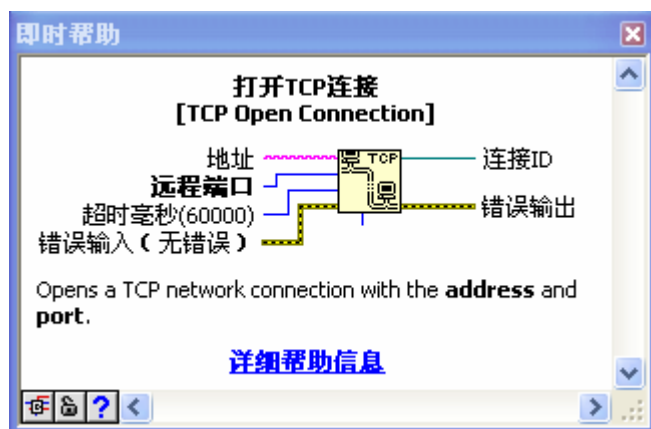
qinfei@d610-qinfei ~
$ xserve -c -xmlc -xmlport=9005 -s=com15
[2007/09/20 16:59:22] xdebug: could not open log file /opt/MoteWorks/tools/xserve/bin/logs/xserve_log.txt: No such file or directory
XSERVE 2.0.E: $Id: xserve.c,v 1.8.2.3 2007/02/02 17:45:01 rkapur Exp $
Warning: Converting Windows com15 device to Cygwin device.

qinfei@d610-qinfei ~

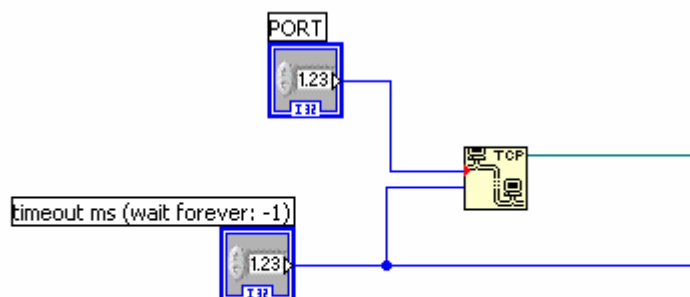
```

通过输入 `-xmlport=9005` 我们可以把解析后的数据发布到 socket 9005 端口上。我们之后就可以通过在 Labview 中调用 TCP 相关控件来访问并从 9005 端口获取信息。

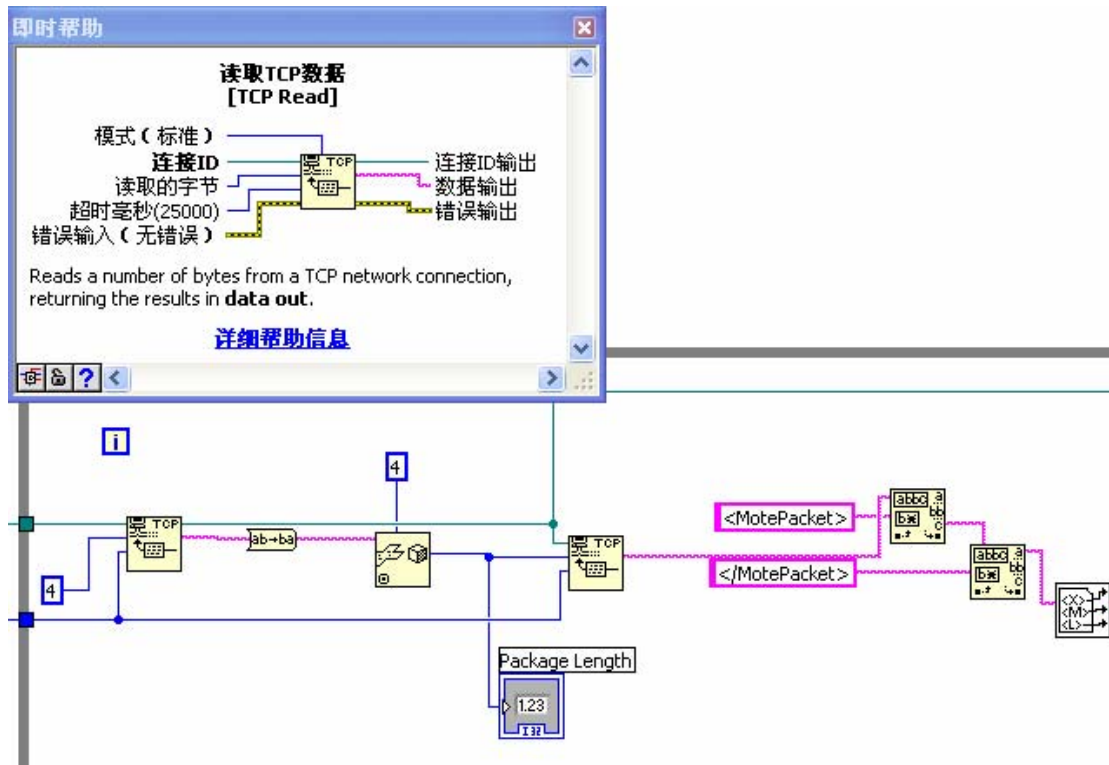
下一步要解决的问题是在 Labview 中连接 9005 端口，并从该端口获取数据信息。我们打开通信选版，选取打开 TCP 连接：



分配对应的端口，地址，和超时信息：



在这个 DEMO 程序中我们实现了一个简单的访问本地 Xserver 端口的程序，所以未分配地址。在实际应用中用户只需将目标机的 IP 地址给出就可以访问远程 Xserver 了。

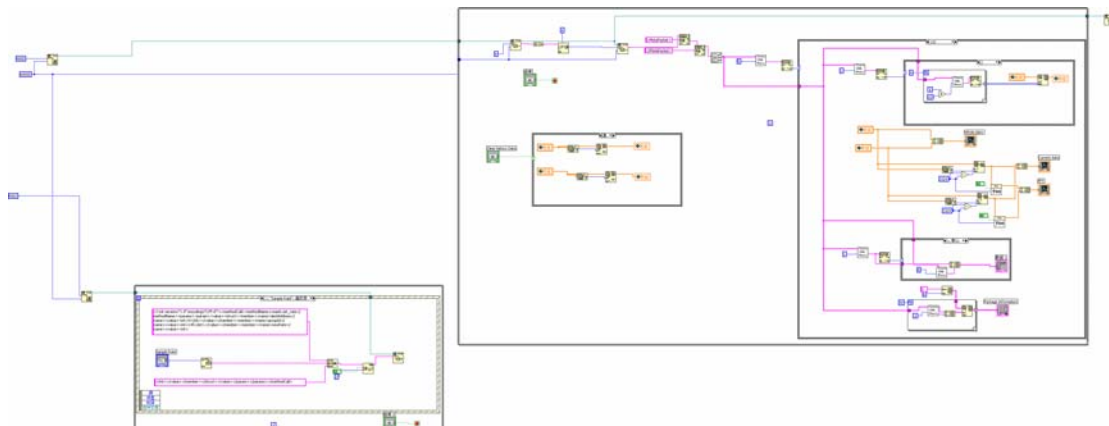


之后在 Xserver 开启的情况下，理论上我们就可以通过读取 TCP 数据控件将数据截取下来，送交上文我们建立的 Crossbow XML 解析器处理了。^_^

但是在此之前我们还有一个相当重要的事件要处理。在上文处理 XML 文件时，每个文件包含一个完整的 XML 数据包，但是在网络模式下 XML 包是以数据流的形式连续不断的发送到 9005 端口上。因此 Xserver 采用了一个简单通用的协议来解决这一问题。每当 Xserver 解析完一个数据包，并试图将对应的 XML 包放置到 9005 端口时，它会先放出 4 个字节描述该包的大小，然后再放出 XML 包。

应在此 Labview 中我们也按照此协议进行处理，先用读取 TCP 连接的 4 个字节，再将 4 个字节解析成数据格式，然后再读取这个 4 个字节所表示长度的数据包。

应用这套机制，我们就可以源源不断处理 Xserver 的数据包了。



三：下行命令通道

无线传感器网络通常需要维护双向的通信通道，由下至上的通道用来将节点采集的数据发送至中央服务器，这全网络中最主要的数据流向，占到数据量的 99% 以上。此外，由中央服务器至节点的下行信道也是必要的，用来发送控制命令。

在 Xserver 中系统实现了 XCommand 用来发送下行数据。同样的，用户可以用 XML 格式与 Xserver 交互，发送相应的下行数据。

Xserver 的上行信道采用的是 socket9005 端口，相对应的下行信道采用 9003 端口。用户只需要将需要下发的内容填充到对应的 XML 包中，发送的 9003 端口，Xserver 将自动解析，并将数据发送至指定节点。

与接收 9005 端口数据时先接收 4 字节包长度相似。我们在把数据发送到 9003 端口前必须先添加 4 字节表明数据包长度。

