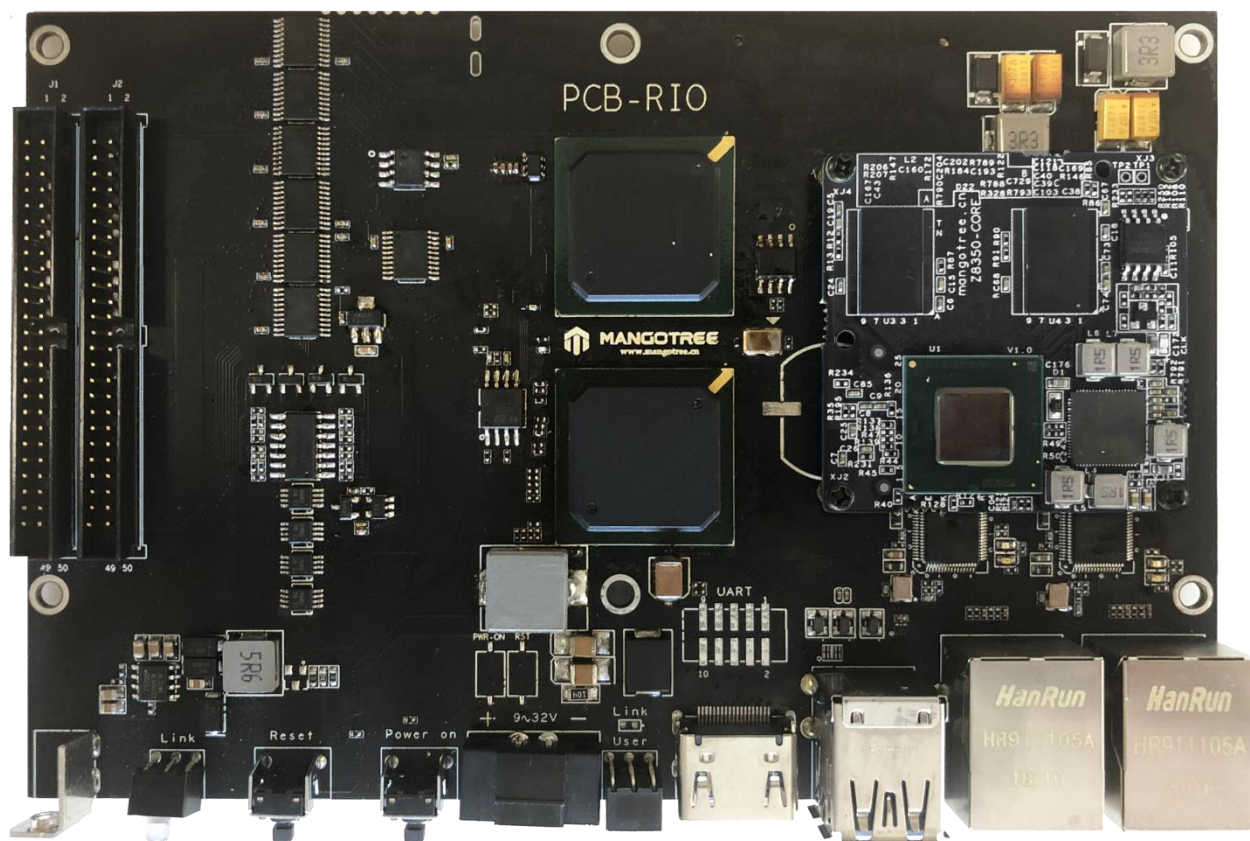


MangoTree PCB-R10

用户指南



目录

一、	产品概述	3
二、	开发环境搭建	4
三、	开发第一个 PCB-RIO 项目	8
1.	连接与发现设备	8
2.	LinuxRT语言环境配置	8
3.	新建LabVIEW项目	9
4.	新建终端设备	10
5.	更改设备IP地址（如需）	11
6.	连接PCB-RIO终端设备	12
7.	新建MT-P812机箱终端（以MT-P812为例）	12
8.	新建PCB-RIO FPGA终端	13
9.	第一个PCB-RIO项目	14
10.	部署实时程序	34

一、产品概述

PCB-RIO 是一种小巧而且坚固的工业化控制和采集系统，采用可重新配置 I/O (Reconfigurable I/O, 缩写为 RIO) FPGA 技术实现超高性能和可自定义功能。PCB-RIO 嵌入式系统可以使用高效的 LabVIEW 图形化编程工具进行快速开发。利用 PCB-RIO 可以快速建立嵌入式控制和采集系统，而且该系统的工作性能和优化特性可与专门定制的硬件电路媲美，用户可以使用 LabVIEW、LabVIEW RT 模块、LabVIEW FPGA 模块、VISA 驱动、RIO 驱动以及 MT-RIO 驱动来开发 PCB-RIO 嵌入式系统。

用户可以通过 LabVIEW FPGA 的基本 I/O 功能，直接访问 PCB-RIO FPGA 硬件上 I/O，通过 I/O 来进行相应的操作。PCB-RIO 嵌入式系统最高搭载 1.92GHz 四核 Intel 处理器，16GB EMMC，2GB 内存，在保证可靠性和确定性的前提下顺利执行 LabVIEW 实时程序，分析，数据记录与通讯。控制器上有 100Mb/s 以太网，可在网络上进行通信编程。由于 RIO 的并行特性，所以增加新的计算并不会降低 FPGA 程序的执行速度。

一般情况下，可重新配置的控制和采集系统包括四个主要部分：

- ✓ 用于输入，输出，通讯和控制的 RIO FPGA 核心部分；
- ✓ 用于浮点控制，信号处理，分析和逐点决策的实时循环；
- ✓ 用于内嵌数据记录，网络远程面板和以太网/串口的普通优先级的循环；
- ✓ 用于远程图形化用户界面，历史数据记录和事后处理的联网的主 PC；

使用 PCB-RIO，您可以快速的解决复杂问题的挑战，大大提高工作效率。

二、开发环境搭建

用户需要自行在自己的PC下搭建开发环境，具体参照如下步骤：

注意：以下所有软件必须按照默认的路径安装（C盘），且按照先后顺序，否则会导致无法安装成功，共需要C盘20G左右的空间。

下面所有的安装过程，确保电脑没有开启任何杀毒软件及安全助手，以免导致安装失败。

步骤一：Labview相关软件安装

目前芒果树推荐安装2016或者2020版，可在下面的链接中进行下载安装

<http://server.mangotree.cn:9000/Software/>；链接中有两个版本2016及2020：

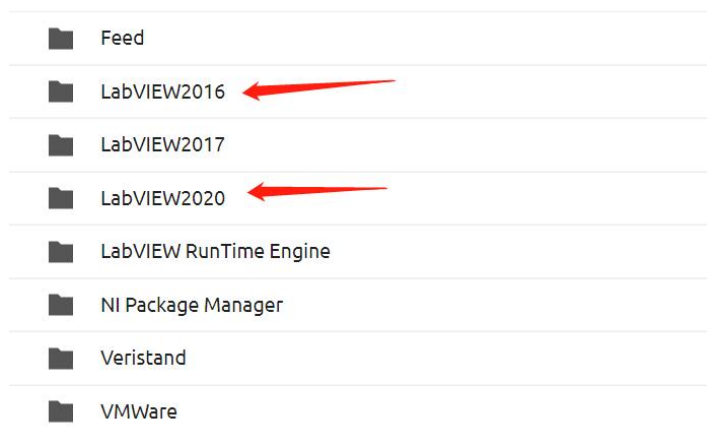


图2-1

如果安装LabVIEW2016，按照数字顺序安装文件夹下的6个安装包；

⚙️	01-LabVIEW Chinese 2016.exe
⚙️	02-LabVIEW FPGA-Chn 2016.exe
⚙️	03-LabVIEW Real-Time-Chn 2016.exe
📁	04-NI VISA 2016.zip
📁	05-CompactRIO Driver 2016.zip
📁	06-LV RunTime Engine 2017.zip

图2-2

如果安装LabVIEW2020，如下图：



图2-3

用户可以在文件夹1下找到独立的安装包，按照数字顺序分别安装1-5即可；也可在文件夹3下下载完整安装包（较大）一起安装，语言版本用户自己选择，正常基本开发只需勾选下面五项内容进行安装即可：



图2-4



图2-5

在下面这个界面中，按照提示的安装默认选项进行下一步：



图2-6

步骤二：MT RIO驱动安装

MT RIO驱动安装包请联系芒果树售后支持提供。

如何验证MT RIO安装包是否安装成功？

打开Labview新建一个项目，在我的电脑下右键新建一个终端，如图2-7所示：

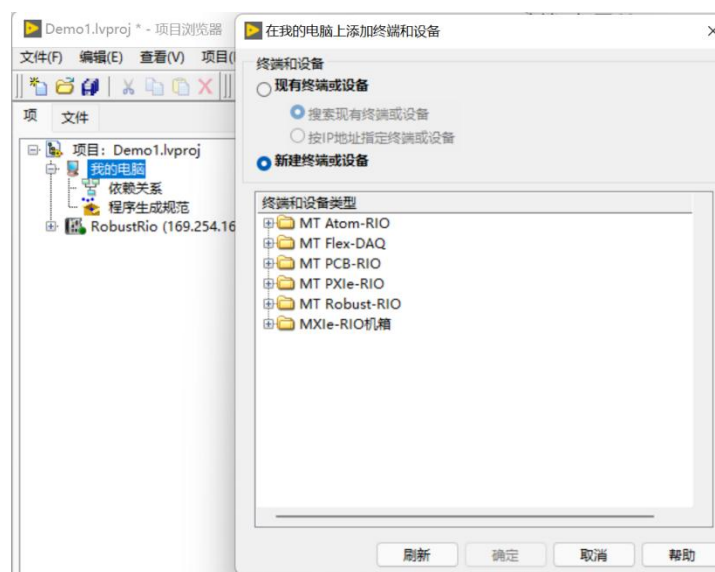


图2-7

如果能出现如图所示MT的产品选型即认为MT驱动软件已经安装完毕。

步骤三：FPGA编译器安装

FPGA程序需要安装编译器进行编译，目前芒果树的产品只支持ISE14.7编译器，下载地址：

<http://server.mangotree.cn:9000/Software/LabVIEW2020/Separate%20Install/>， 链接列表中的第07项为编

译器安装包，编译器支持不分Labview版本。如果购买产品的FPGA型号是我们的K7版本，则需

要在安装好编译器后安装芒果树编译补丁包，补丁包在此链接中

<http://server.mangotree.cn:9000/Custom/>

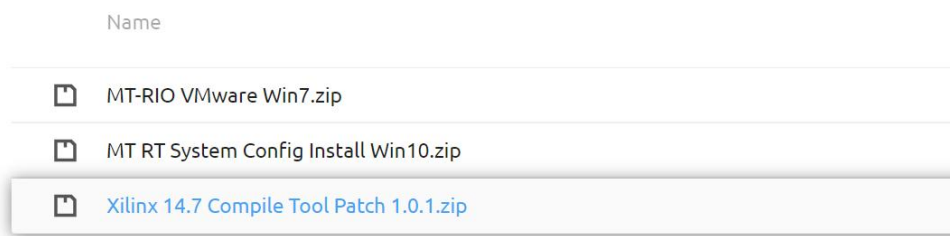


图2-8

三、开发第一个 PCB-RIO 项目

1. 连接与发现设备

Linux RT出厂IP设置是DHCP，用户需要确保PCB-RIO的IP地址与PC主机的IP地址在同一个网段（需将PC也设置为DHCP），用户可以将PCB-RIO通过网线直接连接上位机PC，或者通过交换机连接PCB-RIO和上位机PC，正确连接之后都可以在NI MAX的远程系统中发现设备，如图3-1所示。



图3-1 MAX中发现远程系统设备

在NI MAX中可以查看设备的操作系统，名称，IP地址和其他相关网络设置，也可以修改相关配置。如果没有找到设备，请检查设备网络接口连接是否有问题（网口灯是否闪烁）。

2. LinuxRT语言环境配置

单击选中远程系统中发现的PCB-RIO设备，在界面右侧可以看到LinuxRT的系统配置及启动配置等信息，如图3-2所示：



图3-2 LinuxRT系统设置

LinuxRT系统语言中文和英文的区别在于是否支持UI界面，如上图所示，当LinuxRT语言环境为中文时，是无法启用嵌入式UI的，只有语言环境选择英文才能勾选启用嵌入式UI。更改语言后需要保存重启设备才能生效。

3. 新建LabVIEW项目

打开LabVIEW，在LabVIEW中点击文件，新建（N），新建一个项目如图3-3所示。



图3-3 新建一个LabVIEW项目

4. 新建终端设备

点击 PCB-RIO项目浏览器，选择新建一个终端如图3-4所示。

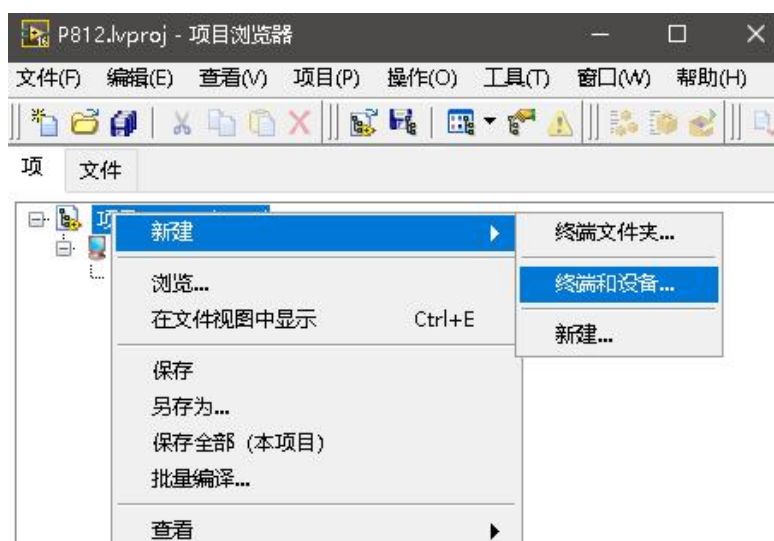


图3-4 新建一个终端项目

点击新建终端或者设备，选择Real-Time CompactRIO下的PCB-RIO如图3-5所示。

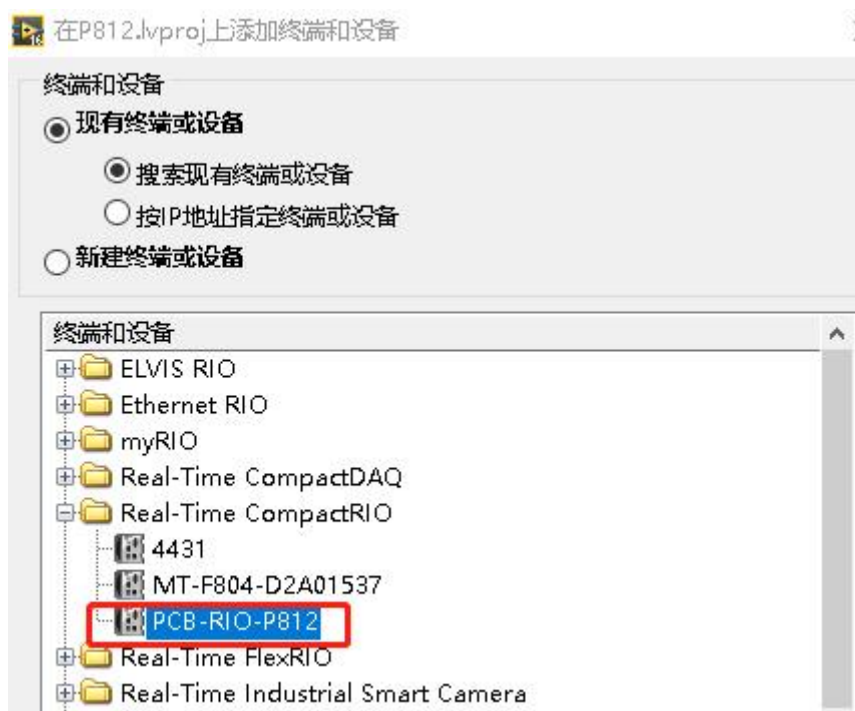


图3-5 选择PCB-RIO

5. 更改设备IP地址（如需）

一般按照上述方法新建的设备，默认地址都是设备的IP地址，无需改动，但是可以通过下面的方式，确认和更改。通过终端属性选项更改IP 地址如下图所示，填写MAX中的设备IP地址。



图3-6 选择终端属性



图3-7 更改IP地址

6. 连接PCB-RIO终端设备

在项目浏览器中右击终端，选择连接，即可连接 PCB-RIO 如下图所示。

注：需要确保上位机PC安装Labview语言版本与下位机LinuxRT的系统语言保持一致才能正常连接，修改LinuxRT系统语言见第二小节。

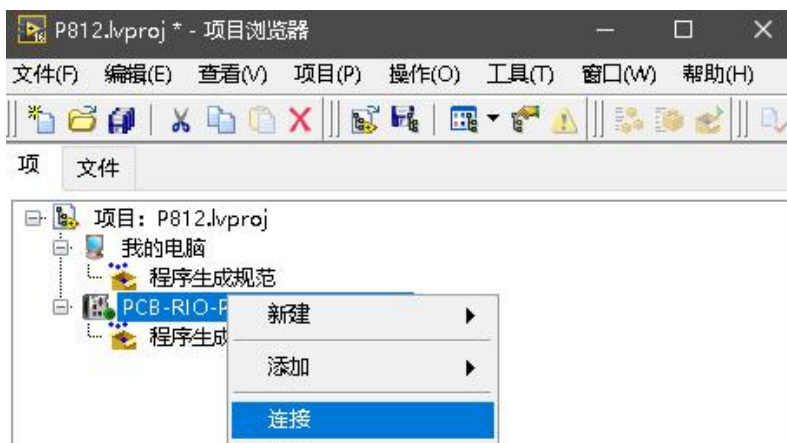


图3-8 连接终端

7. 新建MT-P812机箱终端（以MT-P812为例）

连接完成之后，为RT终端添加FPGA终端设备，右击RT终端PCB-RIO 选择新建终端或者设备，然后选择MT-P812, 如图所示。

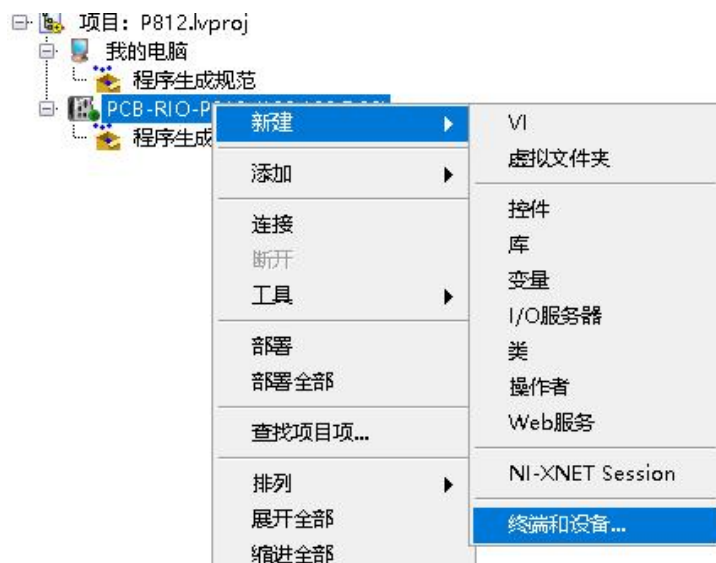


图3-9 新建设备



图3-10 选择MT-P812

8. 新建PCB-RIO FPGA终端

右键P812机箱终端新建FPGA终端后如下图所示。

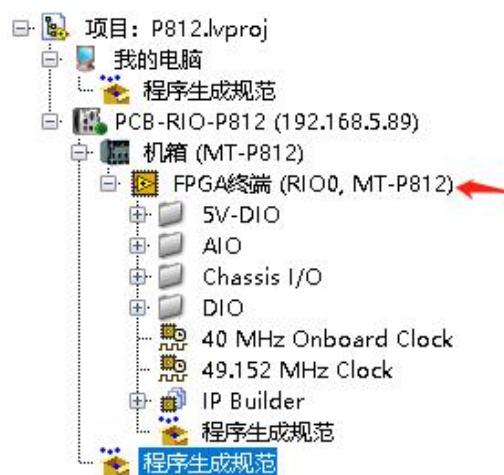


图3-11 P812 FPGA设备新建完成

总结：完成上述配置以及连接之后，我们就可以开始第一个PCB-RIO程序的开发。

9. 第一个PCB-RIO项目

建完FPGA终端后，就可以开始编程开发程序了，先开发FPGA程序，然后开发上位机Host程序。



图3-12

注：P812有AI、AO接口，上图红框中的AIO中的接口用户无法使用，AIO的功能我们提供专门的VI，具体见下面步骤。其余的Chassis I/O、5V-DIO、DIO 下的I/O可以直接拖放到程序框图里使用。

首先在FPGA终端下新建一个VI，右击 FPGA 终端，如图所示。



图3-13 新建FPGA终端VI

MT提供AI与AO的VI给用户实现模拟量的输入与输出功能，右键程序框图空白处选择MT FPGA AIO，相应的VI如下图所示。

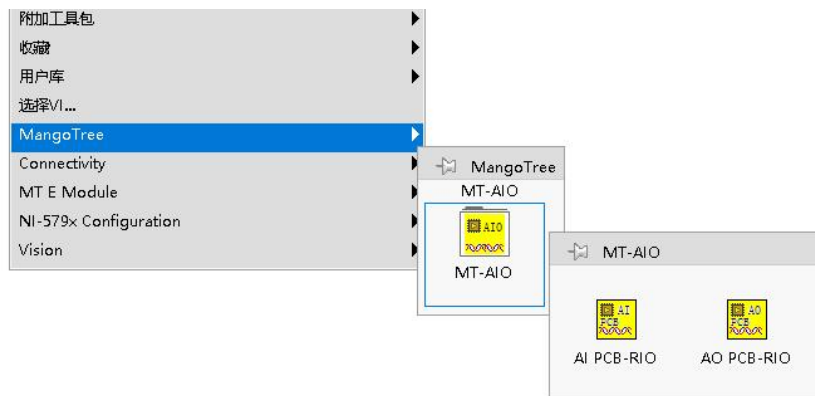


图3-14 MT FPGA AIO

然后分别放置AI及AO VI 如下图所示。



图3-15 AIO FPGA 代码

将 Chassis I/O 下的 Chassis Temperature 和 USER FPGA LED 直接拖拽到FPGA 中的VI 中，这两个 I/O 分别是用来获取板载温度和控制用户双色灯的颜色如图所示



图3-16

给Chassis Temperature 创建一个显示控件，显示温度。给循环定时器创建一个输入控件，然后通过一个布尔值循环取反的方式给 USER FPGA LED 赋值1 和 2，来切换灯的颜色如图所示

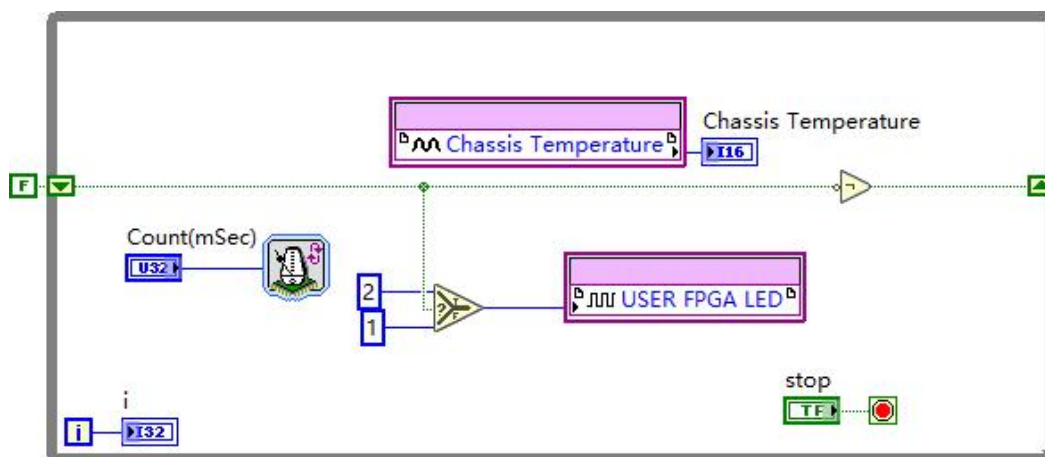


图3-17 读取温度和切换双色等颜色程序

新建一个终端至主机的 FIFO, 完成 FPGA与 RT 之间的 DMA 通讯如图所示。



图3-18 新建FPGA FIFO

设置 FIFO 的属性, 在类型中设置成终端至主机, 同时设置上传数据类型为SGL, 用于上传AI的数据, 如图 3-18 和 3-19 所示。

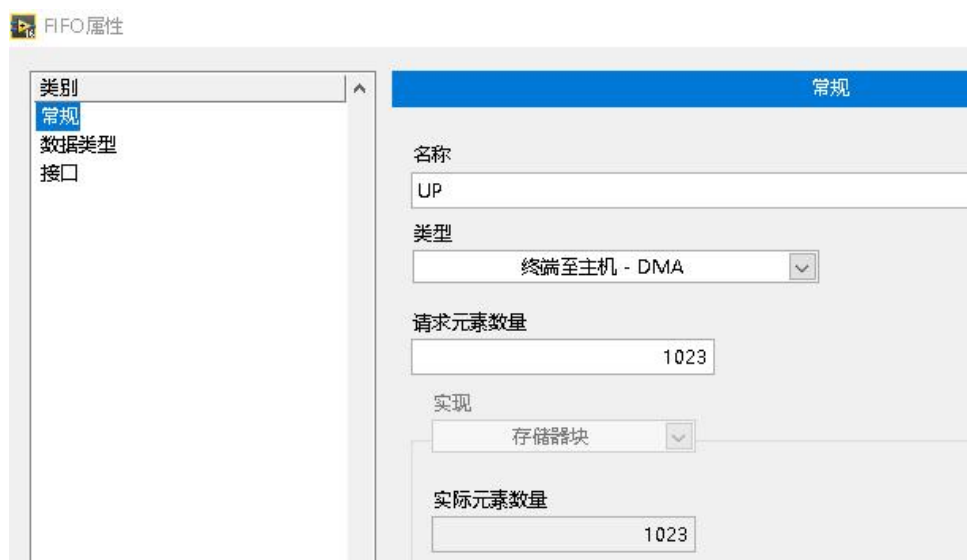


图3-19 设置类型至终端至主机

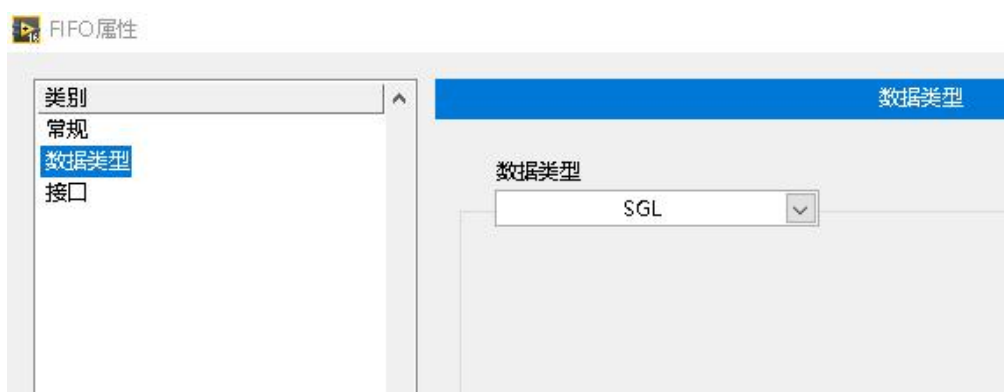


图3-20 设置上传的数据类型为SGL

FIFO创建之后，FPGA 终端下将会出现，可以通过 FIFO 将FPGA终端中的数据上传给上位机RT

如图所示。

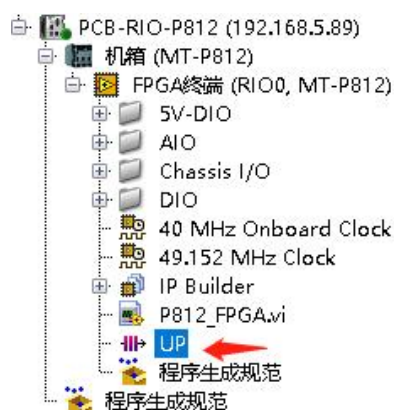


图3-21 新建的 FIFO 将会出现在 FPGA 终端下

将FPGA 终端下的 FIFO 直接拖拽到 FPGA 下 VI 的程序框图中如图所示。



图3-22 将 FIFO 直接拖拽到程序框图中

利用FIFO将AI的数据通过DMA上传，给AO创建输入控件用于设置输出电压，如下图所示：

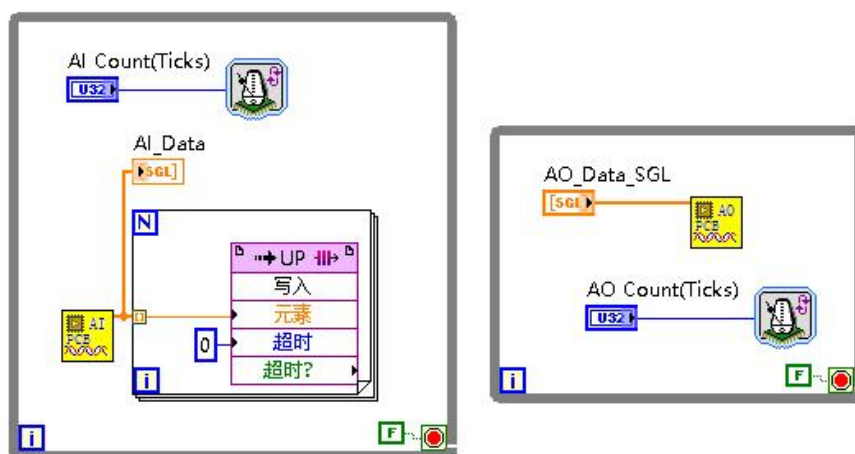


图3-23

采样率=40000000/Count (Ticks)，AI的最大采样率是200K/S/ch，AO的最大采样率是125K/S/ch。

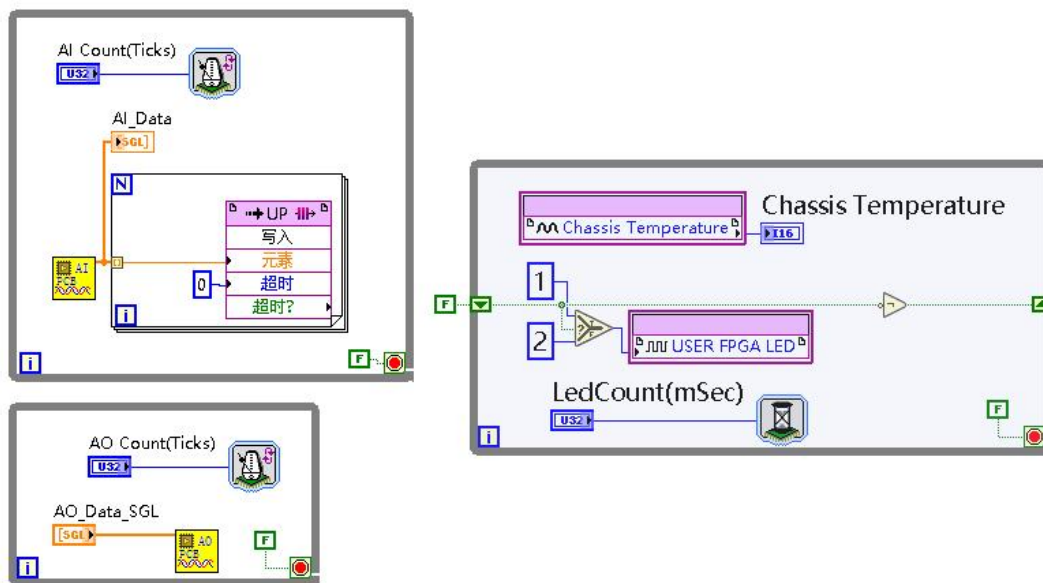


图3-24 完整FPGA Demo程序

FPGA 程序写完之后需要进行编译，点击左上角运行按钮，出现提示框，点击确定按钮，仅生成比特文件，如图所示。

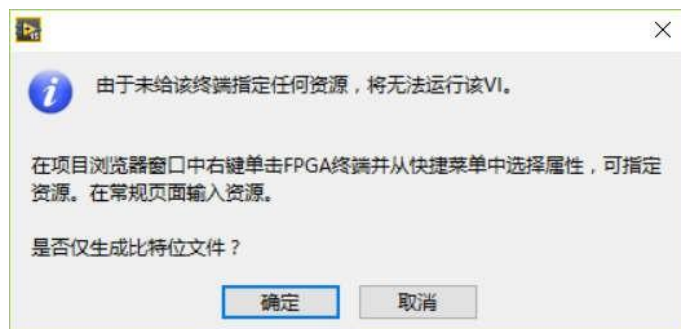


图3-25 点击确定按钮，仅生成比特文件

编译的时候，如果本地计算机安装了编译器可以选择本地编译，如果同个网络中其他计算机安装了编译器，本地没有安装编译器，可以选择连接至网络编译服务器，并输入相应计算机的 IP 地址如图所示。

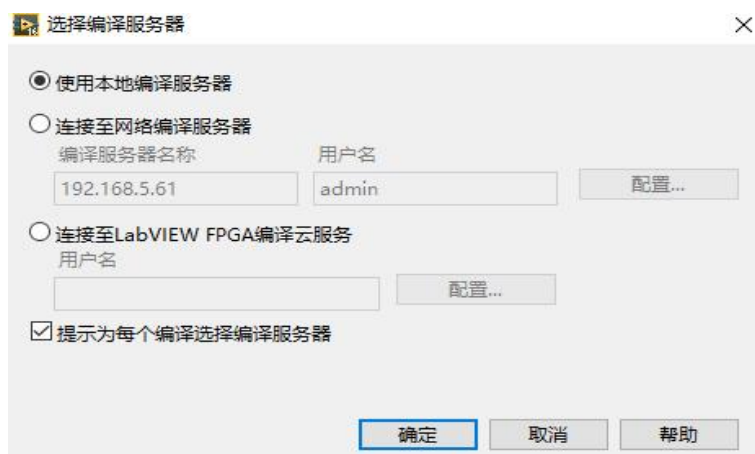


图3-26 选择编译服务器

最终的编译状态界面如图所示。

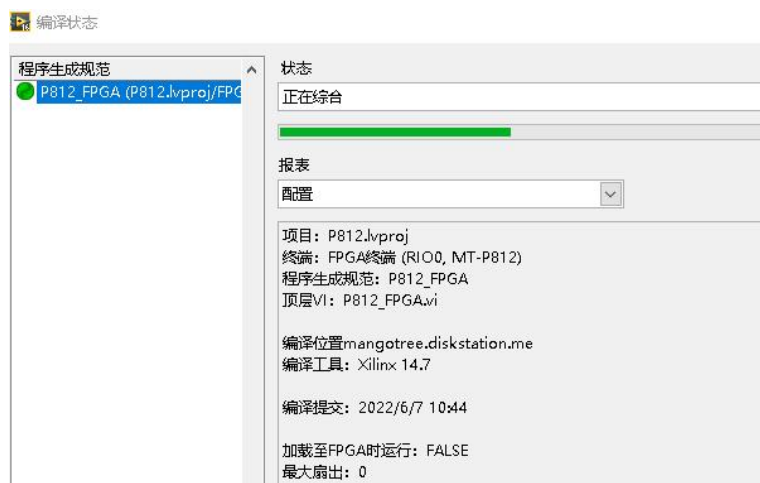


图3-27 编译窗口

编译完成之后，在刚刚保存的项目文件夹下，会出现 FPGA Bitfiles 文件夹，里面存有刚刚编译生成的文件夹如图所示。



图3-28 生成的FPGA Bitfile

至此，FPGA程序开发已经完成，下面介绍上位机程序的开发。

右击RT终端新建上位机VI，并保存，开始上位机VI程序编写：

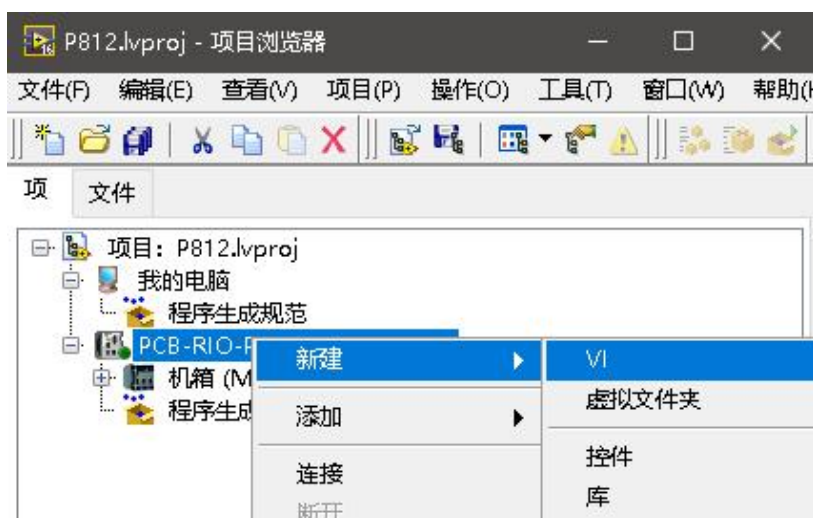


图3-29 右击RT终端，选择新建 VI

在程序框图空白处右击，通过下图所示MangoTree选板进行上位机程序的编写，MT提供上位机VI两种不同的程序写法，分别使用不同的函数包：

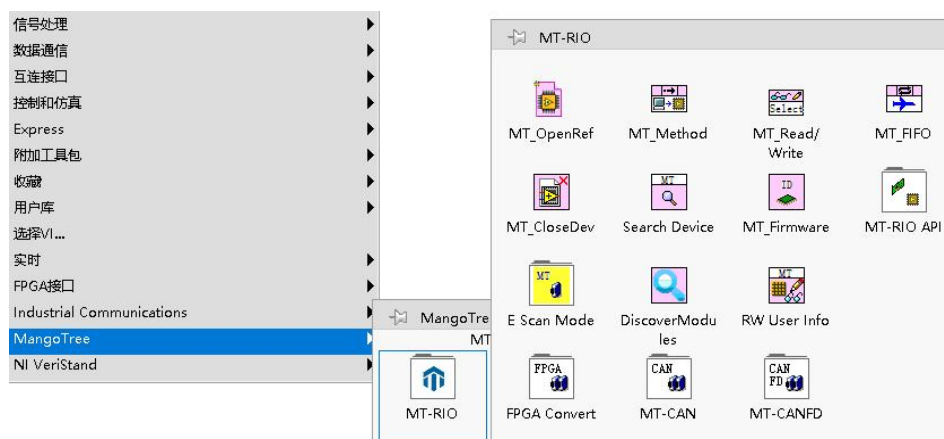


图3-30 MangoTree选板

上位机VI写法A：（基础）

首先在放置一个MT_OpenRef，然后双击，在弹出的对话框中，选择刚刚 FPGA 编译生成的比特文件如图所示。

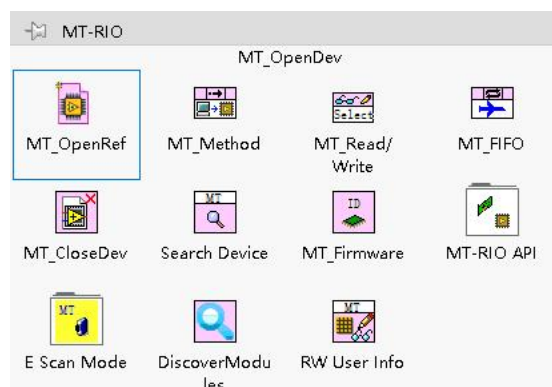


图3-31 放置OpenRef函数



图3-32 选择FPGA比特文件

将 MT_Method 放在程序框图中，并且和 MT_OpenRef 连接，然后左击MT_Method，选择Reset 如图所示。

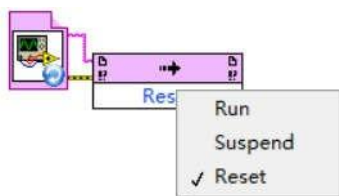


图3-33 右击 MT_Method，选择 Reset

将 MT_FIFO 放置到程序框图中，然后将 MT_FIFO 连接到 MT_Method, 左击MT_FIFO, 选择 Configure，同时给开辟的深度一个较大的值，如图所示。

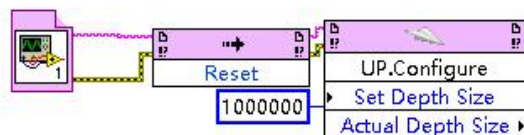


图3-34 配置 FIFO 通道

然后再添加 MT_Method 和 MT_FIFO 结点，MT_Method 左击选择 Run，MT_FIFO选择start 如图所示。

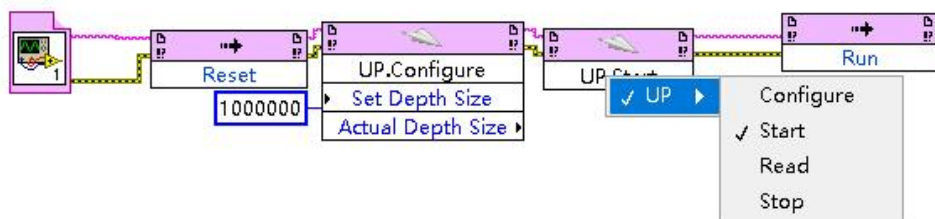


图3-35 在后面连接 Run 和 Start

在程序框图中放置一个 While 循环，然后将 MT_ReadWrite 放置在循环中， 将 MT_ReadWrite 与前面的节点相连接如图所示。

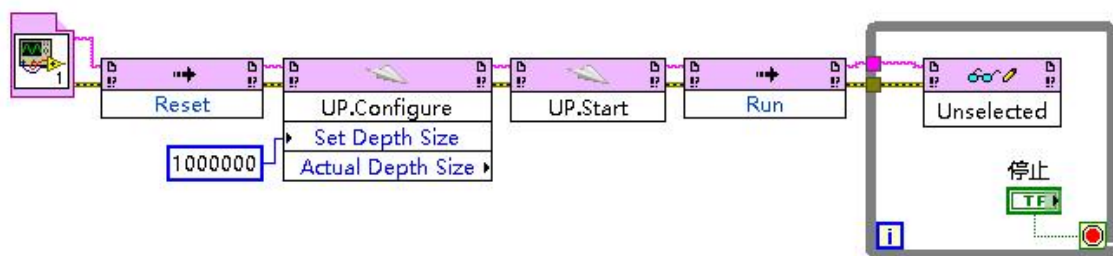


图3-36 将 MT_ReadWrite 放置在 while 循环内

左击 MT_ReadWrite，可以显示 FPGA VI 中的输入控件和显示控件如图所示。

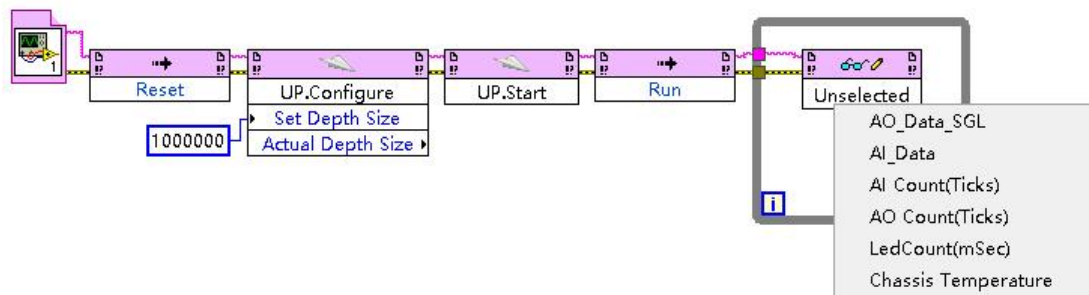


图3-37 左击 MT_ReadWrite

可以将鼠标移动到MT_ReadWrite 的上边缘或者下边缘，当出现双向箭头的时候进行拖拽， 自动出现其他参数如图所示

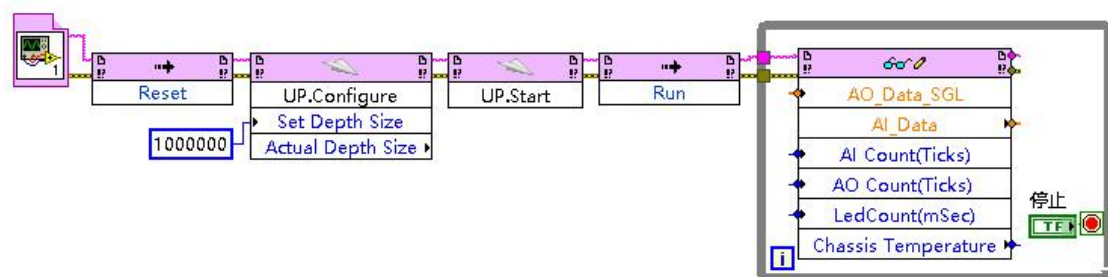


图3-38 进行控件的拖拽

通过在 MT_ReadWrite 的 Count (mSec) 设置 LED 闪烁频率，以及在 Chassis Temperature 处进行除法运算，得到一个板载温度，给 AI Count、AO Count、AO_Data 及 AI_Data 创建输入输出控件，如图所示

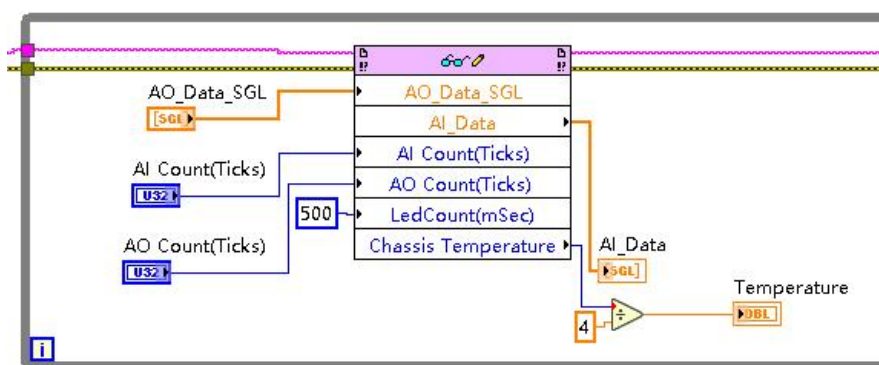


图3-39 与 FPGA 端控件交互

放置一个 MT_FIFO，左击选择 Read，然后给 Number of Elements 赋 0，读出其中 Element Remaining 的值（DMA 缓存中可读数据个数）连接给另一个 FIFO Read，然后将读出数据用波形图表实时显示，如图所示。

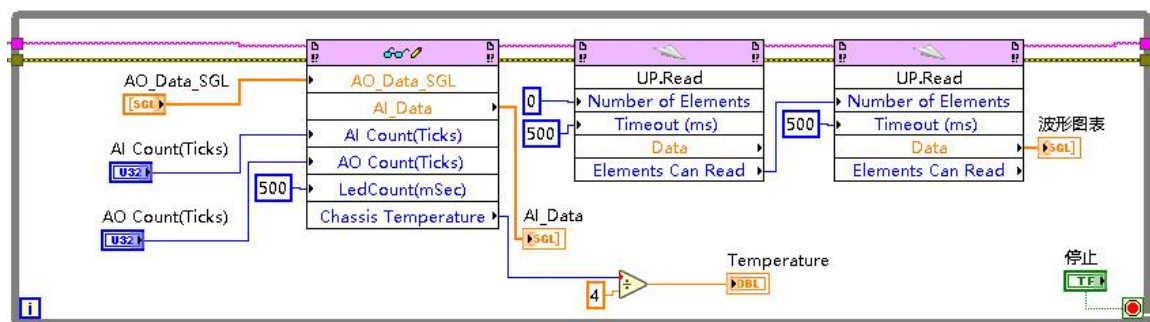


图3-40 显示 FIFO 传输上来的数据

最后放置一个MT_FIFO，选择 Stop，再放置一个MT_CloseDev，完整Host程序如下图所示。

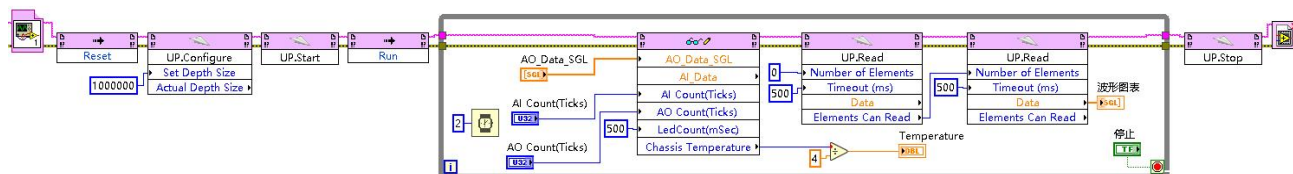


图3-41 完整Host程序

最后点击左上角运行按钮，Host程序即可运行。

注意：上述Host程序是运行在RT终端本地的，用户也可以将Host程序拖到我的电脑目录下，通过填写RT终端的IP地址来实现远程连接运行，具体操作如下：

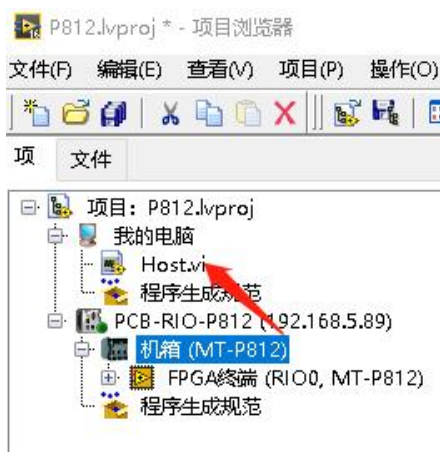


图3-42 将Host程序拖放到我的电脑下

在Host程序框图中添加 Search Device 函数，选择Remote

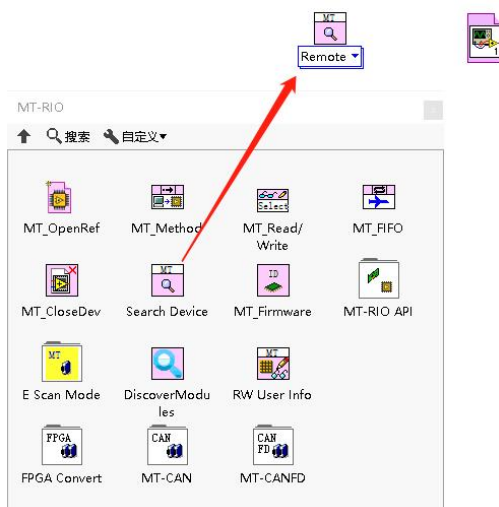


图3-43 添加Search Device

填写RT终端的IP地址，并与MT_OpenRef函数连接

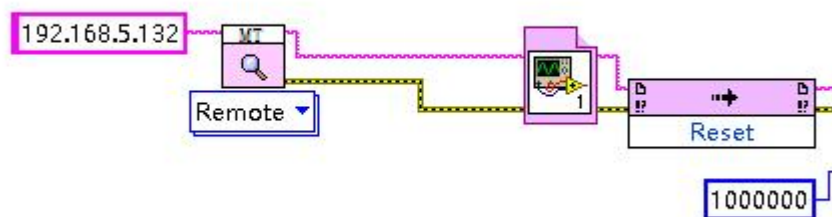


图3-44 填写IP地址

至此，基础的上位机Host程序开发介绍完毕，下面介绍另外一种上位机程序的写法，利用MT RIO API编程，如下图所示为相关API函数。

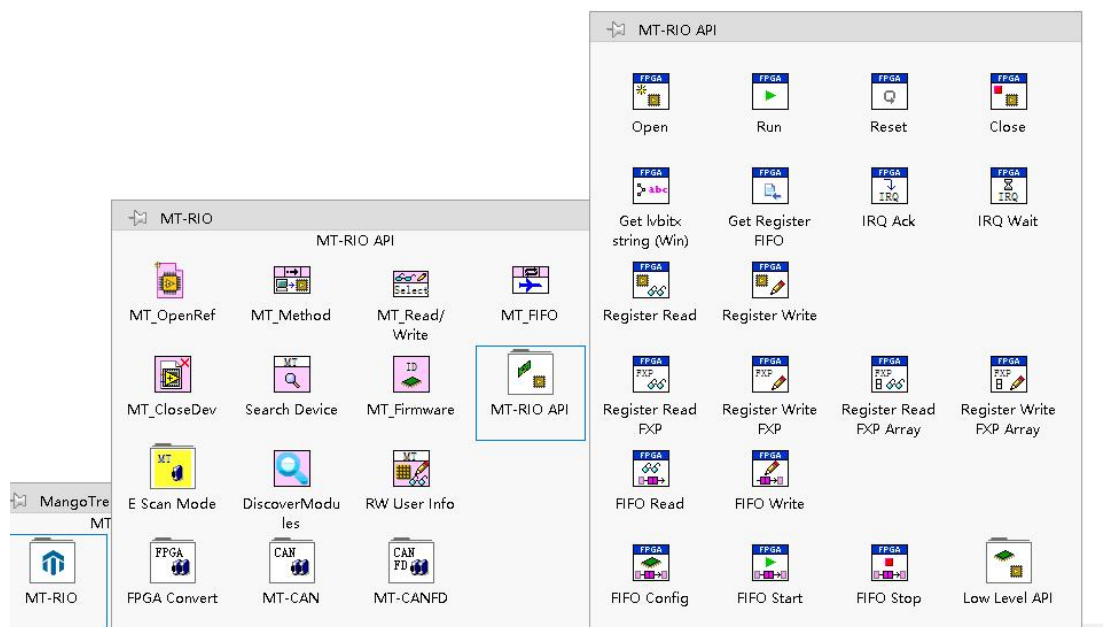


图3-45 MT RIO API函数

通过MT-RIO API选板下的函数编程，用更为底层的方式加载比特文件和读写寄存器及DMA，好处在于，程序比较复杂，涉及多层嵌套VI的项目，FPGA程序修改之后，极大的降低了上位机代码的维护工作量。另外，在比特文件发生改变后，生成好的运行程序（.exe）不需要更新，也可以直接运行。

MT RIO API编程程序总览：

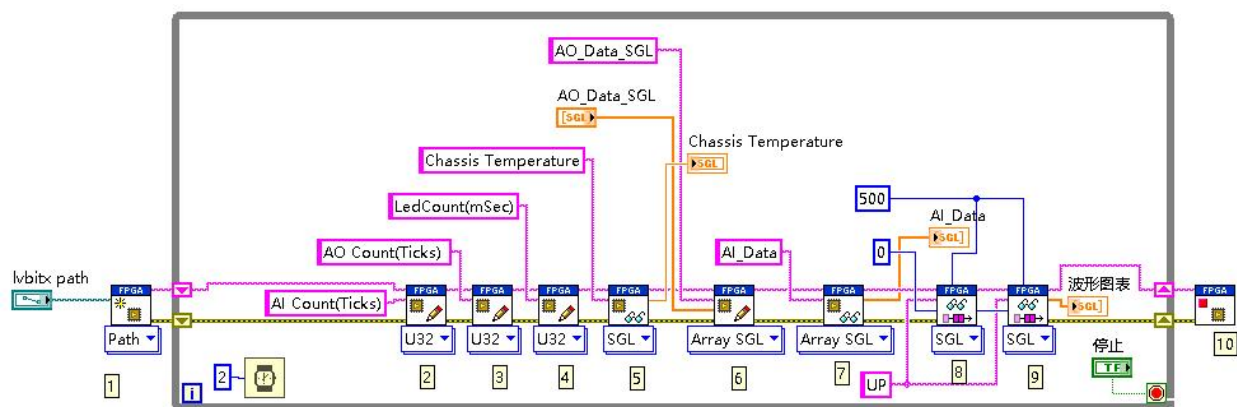


图3-46 MT RIO API编程程序总览

程序说明：

1. Open函数 用于加载bitfile文件

可以发现它有两种加载比特文件的方法，分别是Path和String，如下图：



图3-47

Path方法：

根据Host.vi运行目录位置分为两种情况。

(1) 上位机程序（Host.vi）写在项目中RT终端路径下：

具体操作是：将鼠标光标停在Open函数图标左侧的lvbitx path接口处右键点击选择创建输入控件，建好输入控件后如下图：

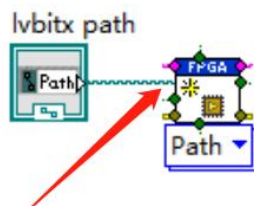


图3-48

然后双击新建的lvbitx path输入控件，会跳到前面板，在这里输入RT终端中比特文件的存放路径（需要事先将比特文件拷贝到RT终端中），**注意**：这里只能通过输入路径的方式，后方文件夹图标按钮只能浏览Windows系统路径所以不能使用。如下图：



图3-49 选择RT下存放比特文件的路径

（2）Host.vi写在**我的电脑**路径下：

那么需要按照下图方式，填写RT终端的IP地址给Open函数：

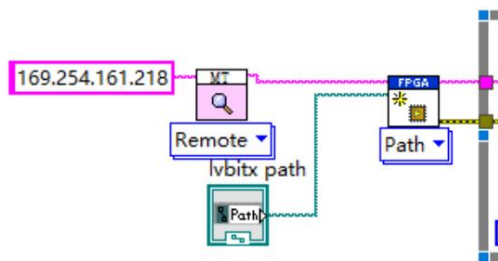


图3-50

此时，因为Host.vi和比特文件都在Windows系统下，所以加载比特文件路径时可以直接

用这个按钮找到比特文件，如下图：

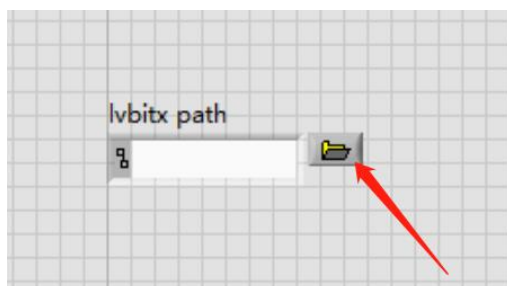


图3-51



图3-52 选择Windows下比特文件存放路径

String方法：

此方法是将比特文件转化成String字符串传递给Open函数，同样根据Host.vi的文件位置分为两种情况。

(1) Host.vi在RT终端路径目录下：

需要在我的电脑目录下新建一个VI并保存：

在该VI中通过MT RIO API选板下的Get lvbitx String函数，将Windows系统中的比特文件转化成String，函数位置和程序如图所示：

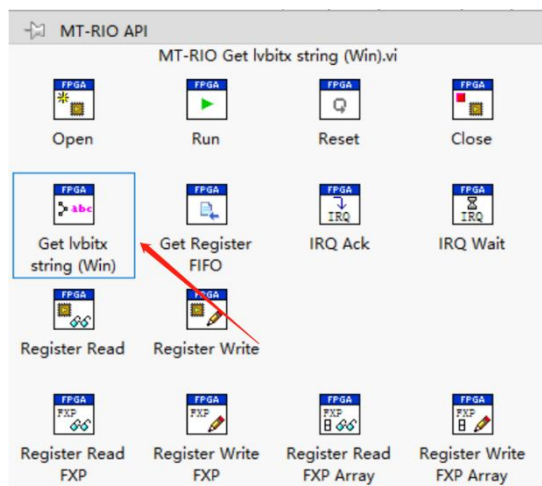


图3-53

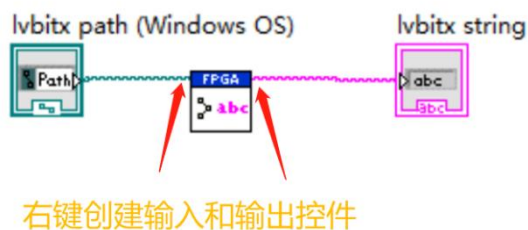


图3-54



图3-55

点击左上角运行按钮，生成字符串：

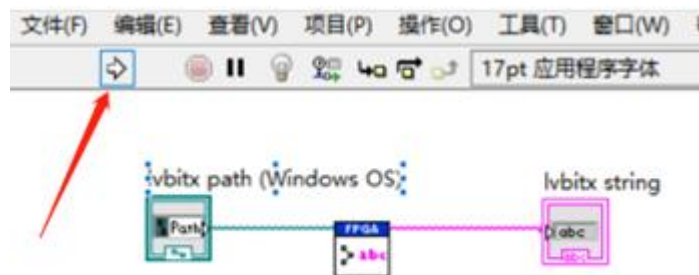


图3-56

左键单击选中显示控件，Ctrl+C 复制该控件，打开上位机程序VI，Ctrl+V将其复制到程序框图中：

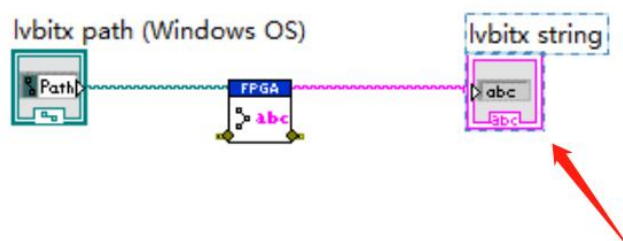


图3-57

右键点击复制好的lvbitx string控件，将其转换为输入控件，如下图：

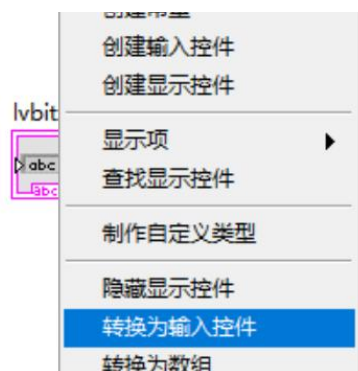


图3-58

并将当前值设置为默认值：

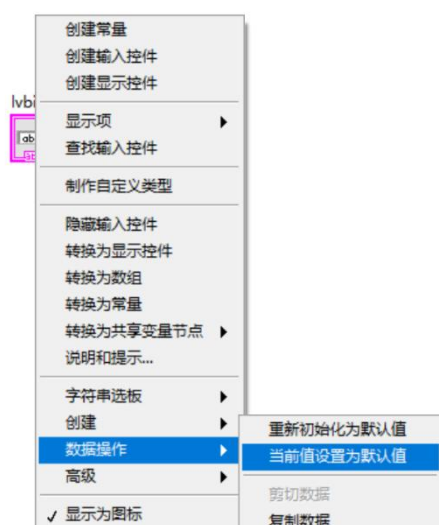


图3-59

然后将String传递给Open函数的lvbitx_string接口，并将Open函数的方法改为String。

如下图：

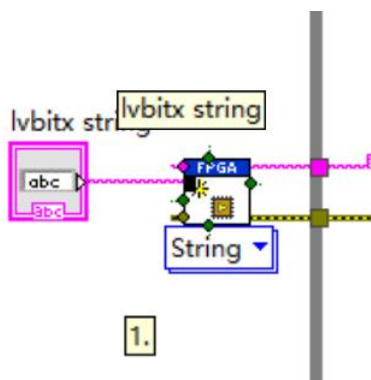


图3-60

(2) Host.vi写在**我的电脑**目录下：

那么除了做到上面(1)中的几步，还需要按照下图方式，在Open函数前面添加Search Device，然后填写RT终端的IP地址给Open函数

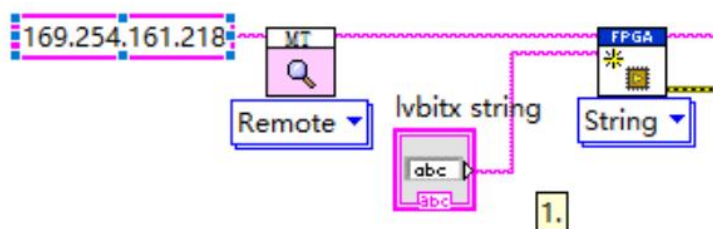


图3-61

继续介绍程序总览，如下图所示：

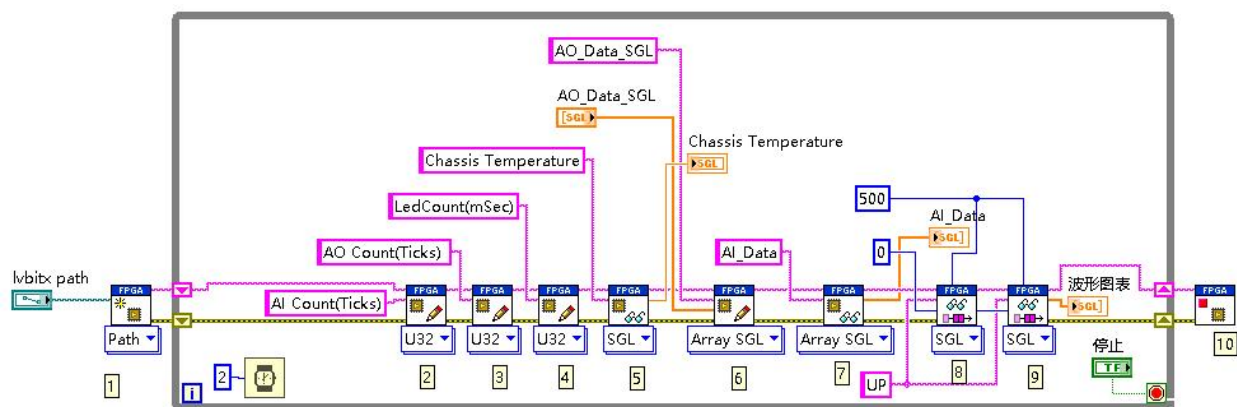


图3-62 MT R10 API编程程序总览

2-7：图中编号2-7为寄存器的读写函数，同于读写FPGA里的输入输出控件，函数需要填写寄存器名称，这里填写的名称要与FPGA程序里定义的一致，如下图所示：

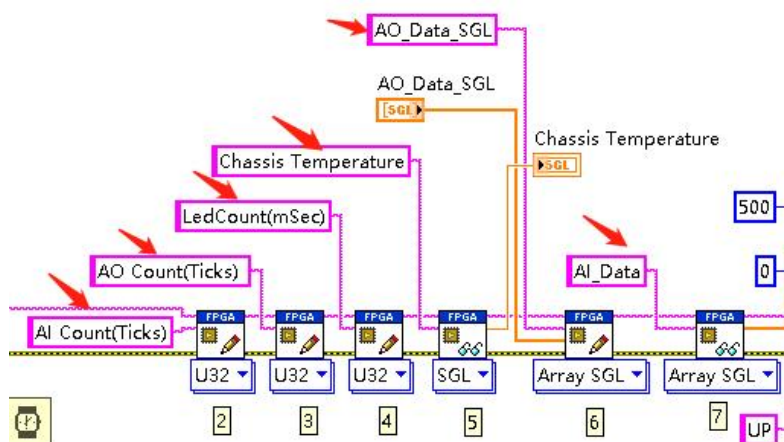


图3-63

读写寄存器函数还需选择对应的数据类型，与FPGA程序中的保持一致，通过下拉选择器来手动选择，如下图所示：

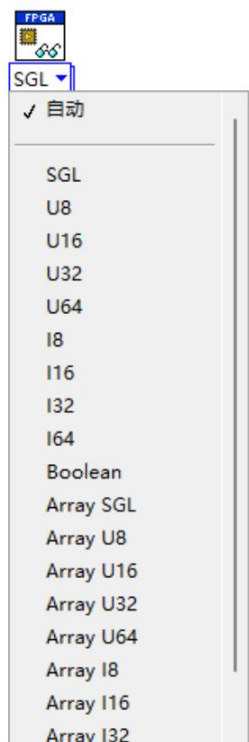


图3-64 选择数据类型

8、9：程序中8和9的位置放置了两个FIFO Read函数，用于读写DMA通道的数据，同样要填写FPGA 中FIFO的名称和DMA所传的数据类型，如下图所示：

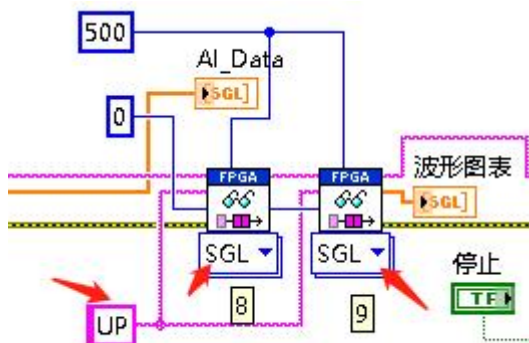


图3-65

10: Close函数。

10. 部署实时程序

在 PCB-R10中生成实时应用程序，当每次PCB-R10上电时，将会自动运行生成的实时应用程序。

首先在实时台式机终端下右击程序生成规范，选择新建，选择实时应用程序，如图所示。

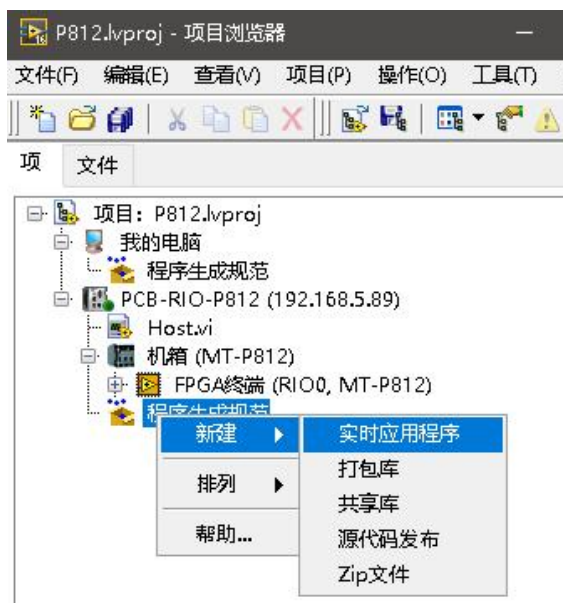


图3-66 新建实时应用程序

在弹出的对话框中，在源文件选项中，把刚刚开发好的实时台式机中的 VI移动到启动 VI 中，然后点击生成，如图所示。

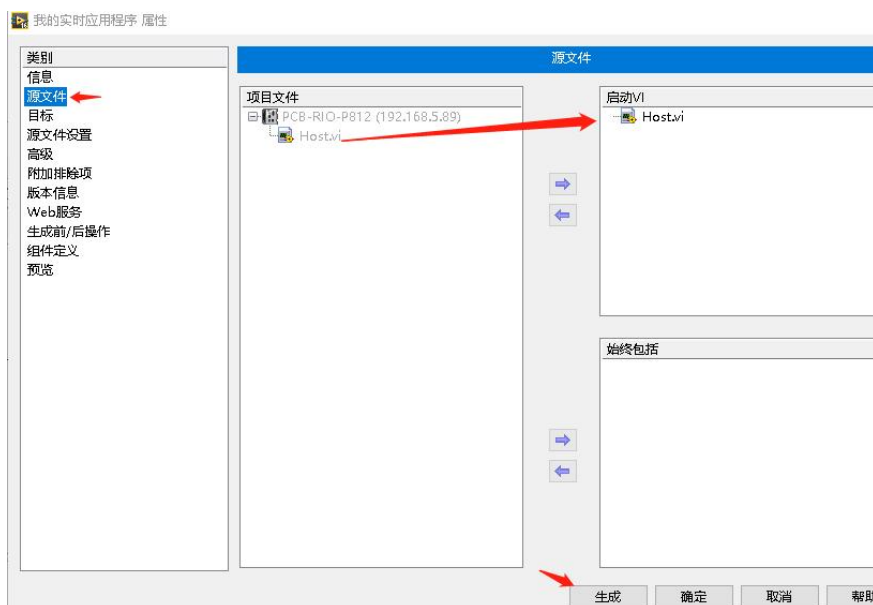


图3-67 生成实时应用程序

在程序生成规范中，右击我的实时应用程序，选择作为启动项运行，当PCB-RIO断电时，再重新上电，即运行的实时应用程序，如图所示。

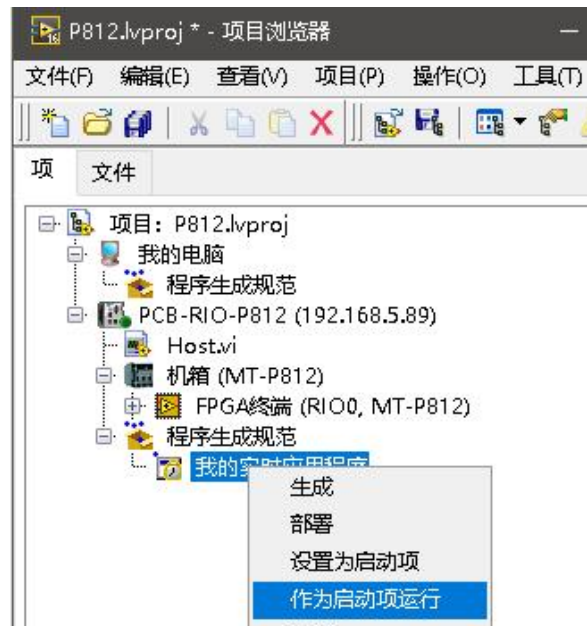


图3-68 作为启动项运行