JNJVS6800MAS 软件开发手册

JNJVS6800MAS 是用于旋转设备状态监测诊断分析系统的上位机软件，用户通过上位机软件可以实时监控现场设备的运行状况。JNJVS6800MAS 软件开发手册中从功能要求、系统分析、数据库文档、界面文档、程序文档等方面展开，说明 JNJVS6800MAS 软件工程的开发过程。

一、 功能要求

JNJVS6800MAS 是用于旋转设备状态监测诊断分析系统的上位机软件，用户通过上位机软件可以实时监控现场设备的运行状况。

监控界面包括设备总图和设备实时界面。设备总图显示工艺区和设备分布、设备模型及测量点示意图以及所有设备各测量点当前监测值。

1.1 设备总图

图 1.1.1 设备总图

界面要求：

1.注明生产厂家名称及 logo

2.注明工程项目名称，可以编辑更换

3.控件菜单区：

文件——保存、打开、另存为

数据——下拉控件

通道——下拉控件

分析——实时波形图、历史曲线图、波特图、轴心轨迹图、瀑布图、时

基图、级联图

报警状态

专家诊断

检维修计划单

4.工艺区名称可编辑，工艺区内设备可增减、编辑

5.建立设备模型及测点示意区

6.测点实时监控值显示区，单位可以组态，每个测点后面点击历史数据可以以文本的形式显示当前测点的历史数据，点击实时监控图形，进入下层界面。

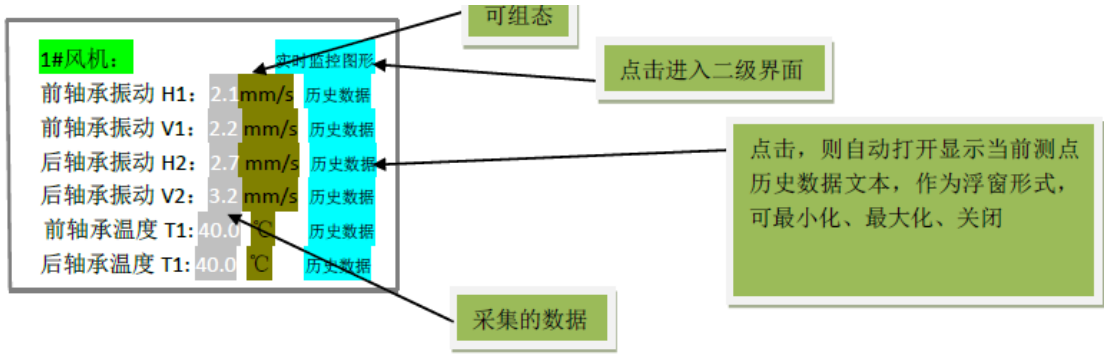


图 1.1.2 测点实时监控值显示区

1.2 设备实时界面

设备实时界面显示位置信息和当前设备的测点、数值、棒状图、报警状态以及设备的实时监控曲线。

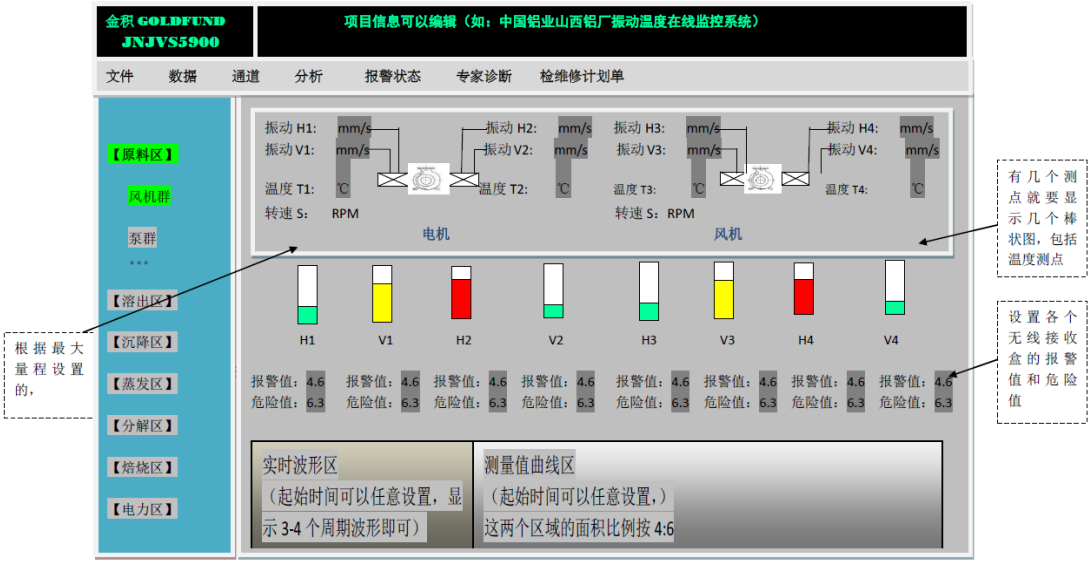


图 1.2.1 设备实时界面

实时界面主要部分，显示当前设备的测点模型和实时数据，模型下方棒状图显示实时数据，并用不同的颜色表示报警状态。显示及设置报警值和危险值。

实时波形区显示信号波形，起始时间任意设置，显示 3-4 个波形周期即可。测量值曲线显示测量值在一段时间内的变化曲线。

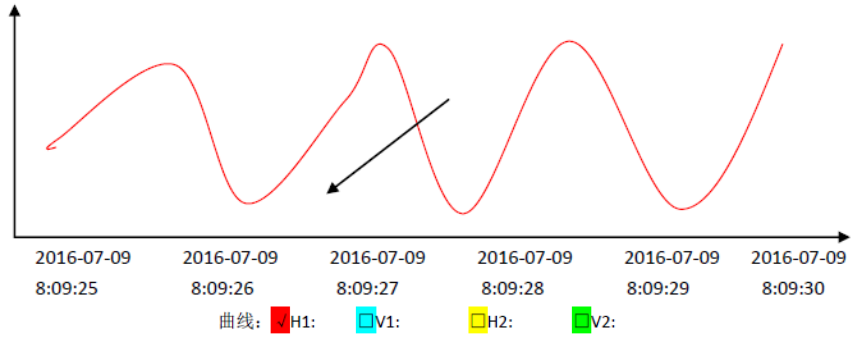


图 1.2.2 实时波形图

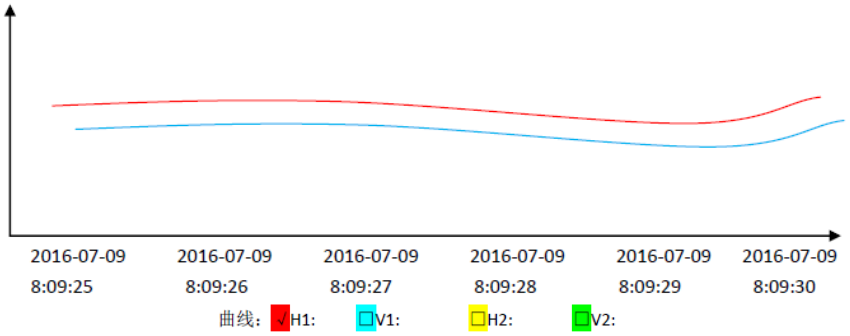


图 1.2.3 测量值曲线

曲线图表中，不同颜色表示不同测点信号/数据，曲线选择框颜色可以组态编辑，点击选中时，显示选中的波形，未选中的测点波形不显示。此波形图上可以同时显示 8 个振动波形。

二、系统分析

在“JNJVS6800MAS——山西建龙钢铁-高速线材设备状态监控分析系统”工程项目中，现场基于振动传感器 JNJ5401/JNJ5500、转速传感器 JNJ5701 采集旋转设备振动和转速信息，经下位机处理后存储到数据库，上位机读取并进行相应的解析计算予以显示。

2.1 现场设备说明

山西建龙钢铁厂高速线材区分为 4 个工艺区：预精轧区、精轧区、夹送辊区、吐丝区。其中预精轧区包括两组锥箱，每组包括一台卧式一台立式，两组锥箱由一个主轴带动；精轧区由精轧电机带动增速箱，增速箱带动两个主轴，每根主轴上联动 5 台精轧锥箱；夹送辊包括上摇箱和下摇箱，共用一根主轴；吐丝区包括一台吐丝机。

现场设备测点出安装传感器，其信号输出接入采集箱，每个机箱由 10 块采集板卡、1 块通讯板、和后板组成，每块振动转速采集卡上有 4 个通道，温度采集卡 10 个通道，每个机箱分配独立的 IP 地址。

工艺区内设备分布如下图所示：

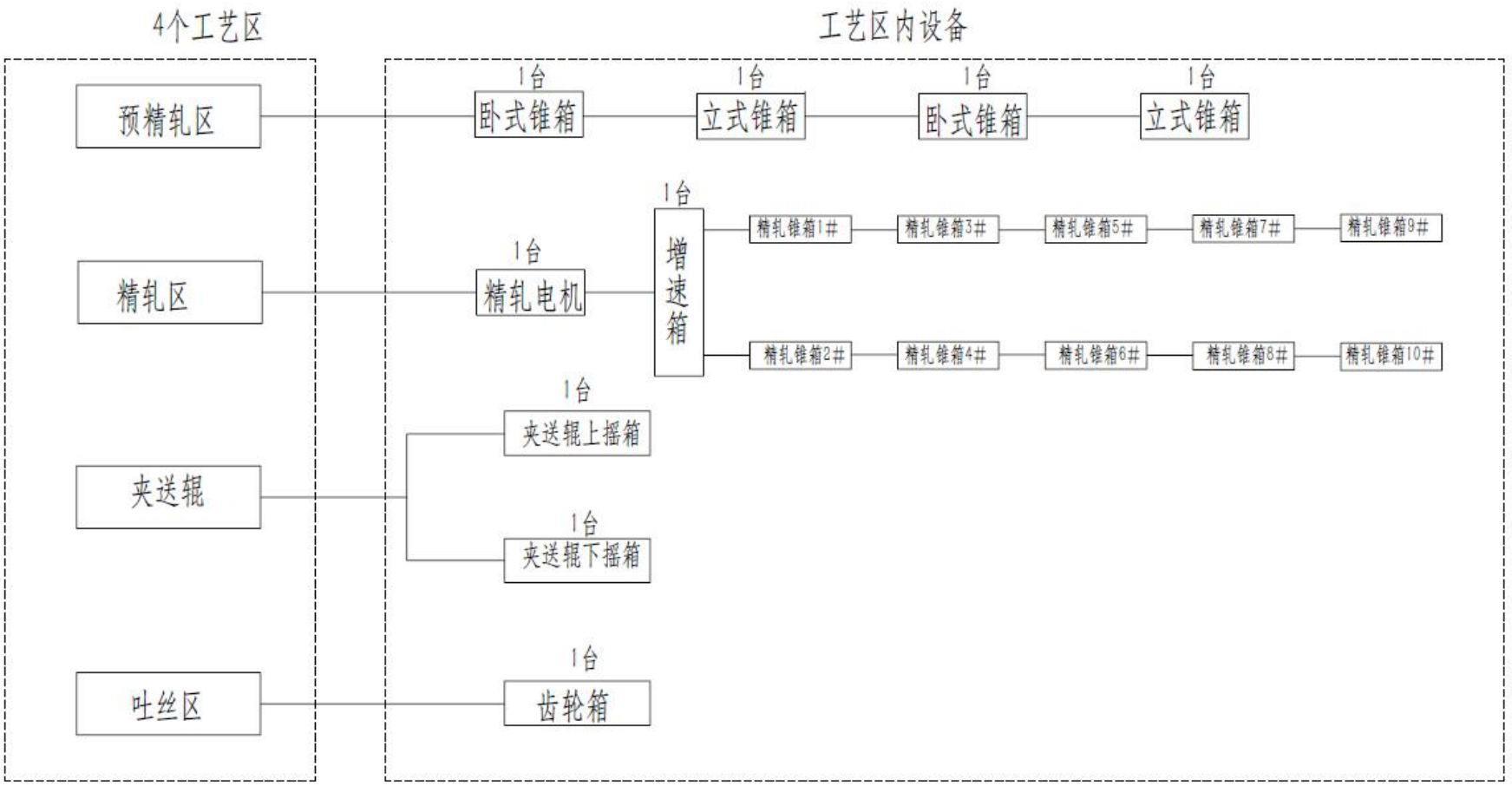


图 2.2.1 工艺区设备分布图

2.2 系统框架

系统硬件平台由采集箱、服务器、监控终端构成，采集箱连接传感器，将现场采集的数据经过转换、计算后发送至服务器存储至数据库。监控终端安装上位机软件 JNJVS6800MAS 通过 ODBC 方式访问数据库，从数据库中获取相应的数据进行解析、计算，在软件中显示处理后的测量值；上位机同时与下位机采集箱进行 TCP/UDP 通讯，上位机向下位机发送命令，下位机接收到命令后向上传输实时信号的波形信息。用户可以在上位机软件配置传感器参数，软件自动写入数据库相应信息表，并向下位机下载配置信息。

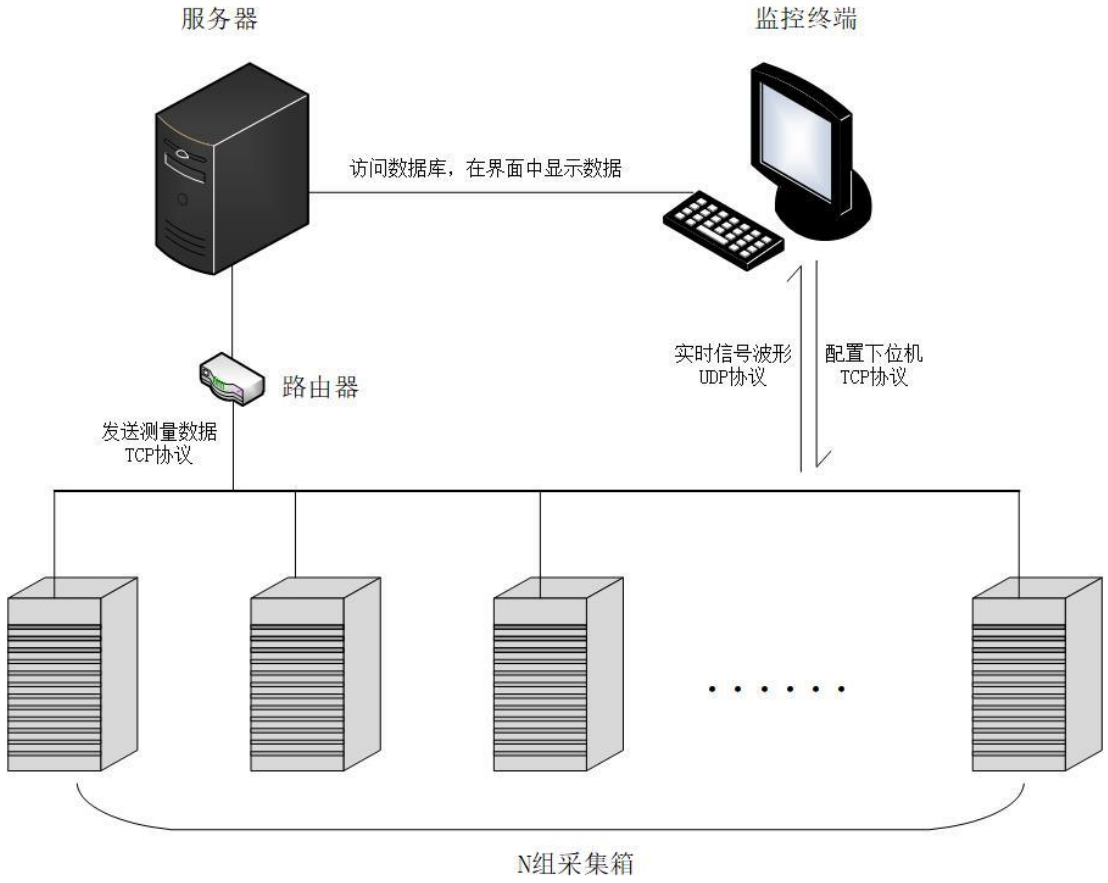


图 2.2.1 系统框架

三、数据库文档

本项目中，基于 MySQL 5.7.19 开发数据库。MySQL 是一个关系型数据库管理系统，由瑞典 MySQL AB 公司开发，目前属于 Oracle 旗下产品，MySQL 是最流行的关系型数据库管理系统之一，使用的语言是用于访问数据库的最常用标准化语言。MySQL 软件采用了双授权政策，分为社区版和企业版，由于其体积小、速度快、总体成本低，在中小型数据库开发中应用广泛。MySQL 既能作为单独的应用程序应用在客户端服务器网络环境中，也能作为一个库嵌入到其他软件中，提供 TCP/ODBC/JDBC 等多种连接方式和存储引擎且提供了用于管理、检查、优化数据库操作的管理工具。

如果链接本地数据库，需要安装 MySQL，从 mysql.com 下载 Windows 下的安装包，可以直接用 MySQL installer（x86）安装，所有的 MySQL 程序都包含在其中；也可以单独选择 MySQL server/client 进行安装。MySQL 有企业版和社区版，其中社区版免费，根据自己的开发需求选择的合适的版本。

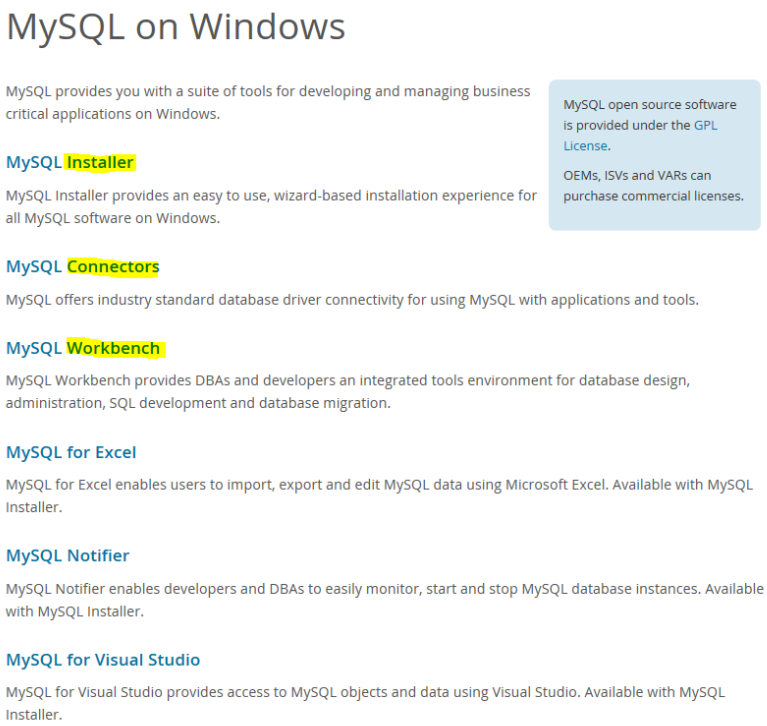


图 3.1 MySQL 数据库安装

3.1 配置数据库

数据库安装结束后，按照 MySQL installer 的步骤提示配置数据库，主要包括：配置服务器类型及端口（默认），创建 root 密码，创建新用户及密码，设置用户和服务器开机启动（默认）。

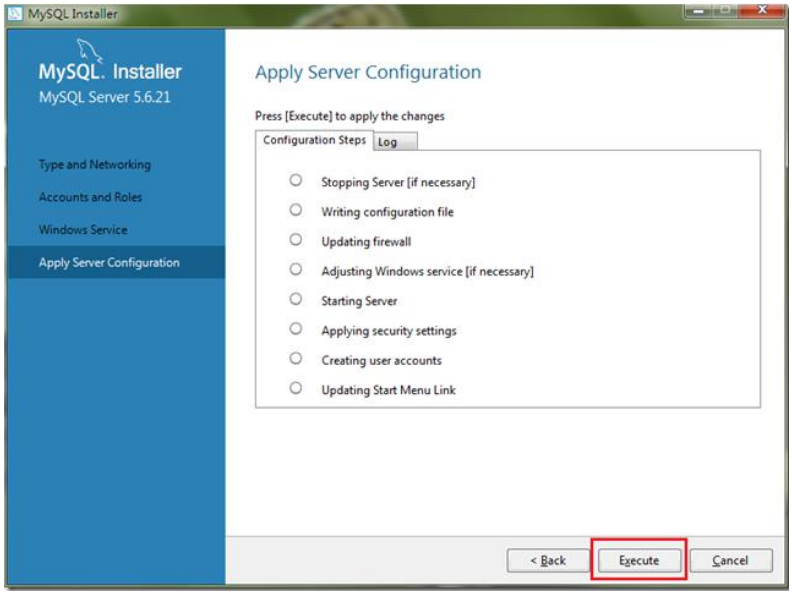
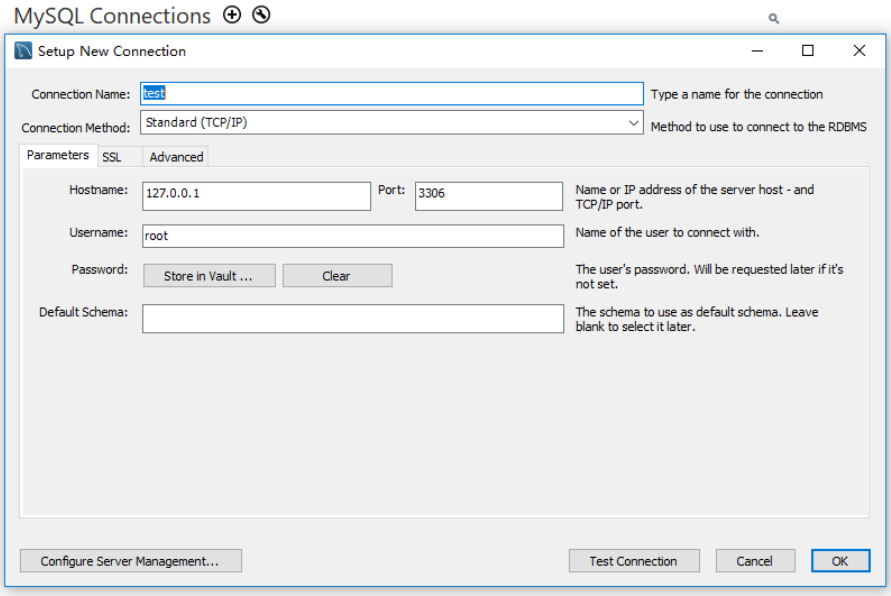


图 3.1.1 配置数据库

配置结束打开 MySQL 官方可视化环境 MySQL workbench，或 MySQL shell 等验证数据库连接。



创建一个连接名，用已经创建的用户或 root 用户测试连接，IP 地址输入服务器地址，端口默认 3306，点击 test connection，弹出 successful connection 即表示连接成功。

3.2 创建/导入数据库和表

在 workbench 界面中点击 new_schema 图标创建一个新数据库（或使用 sql 命令创建），若数据已存在可以直接导入已存在的数据库和表结构及数据到此服务器上，点击左侧 management 一栏中 data import/restore 导入 sql 文件即可。

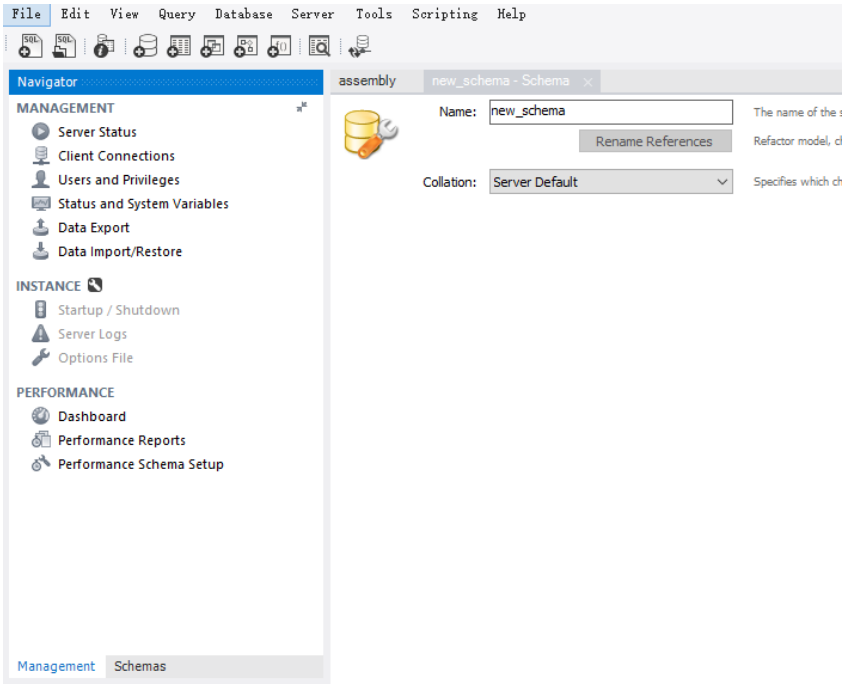


图 3.2.1 创建数据库

在所创建的数据库中，创建相应的表存储数据。创建方式与上相同，可以在 workbench 中借助图形化工具创建，可以使用 sql 命令或从 sql 文件导入。

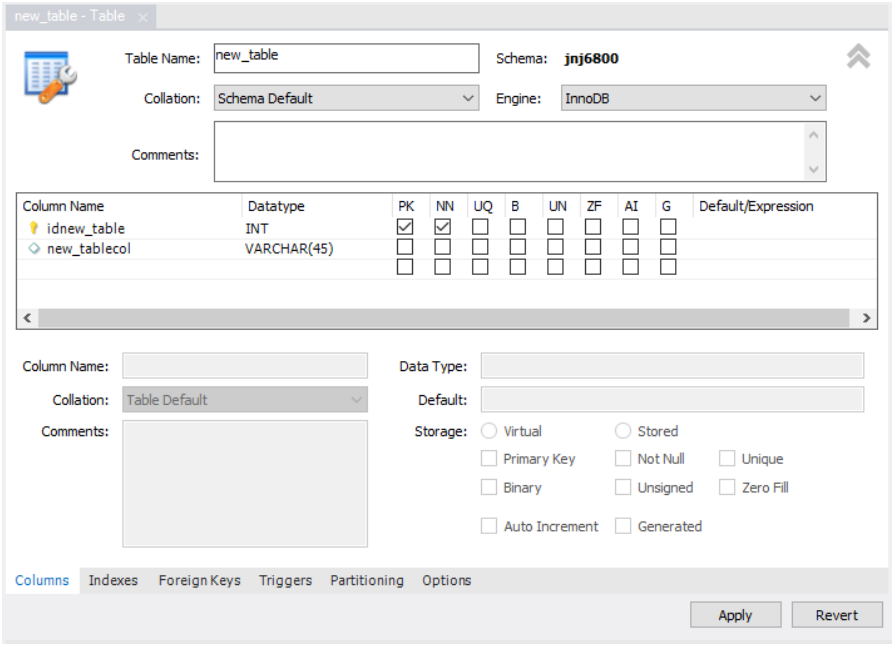


图 3.2.2 创建表

management 一栏中 server status 可以查看当前数据库的运行情况；schema 一栏中显示当前服务器中的数据库及表，点开即可查看表的信息和数据，亦可使用 SQL 命令进行查询。

图 3.2.3 数据库状态

3.3 数据库 JNJ6800 表信息

在本项目中，根据软件需求，创建数据库 JNJ6800,并在其中存储了相应的数据表。数据库表格包括信息表和数据表两类，信息表记录设备安装信息、工艺区信息、传感器信息、机箱信息等；数据表记录下位机发送的测量数据，包括时间

戳、峰峰值、有效值、频率和状态，根据传感器类型的不同存储进振动、转速、温度等相应的表格中。

1.安装表 assembly

表 assembly 存储所有测点的安装信息和配置参数，表字段包括：测点序号 spotID；测点名称 name；传感器型号 sensor；传感器灵敏度 sensitivity；量程 range_lo、range_hi；输出类型 output type；所属设备名称 device；位置信息机箱 rack、板卡 card、通道 channel；测点状态 state；警报值 alarm、危险值 danger;齿数 gear；频率参考信息 freqref，存储值为频率参考索引，可在 finstance 中检索详细信息；备注 comments。

2.工艺区表 sectioninfo

表 Sectioninfo 存储所有工艺区信息，表字段包括：工艺区序列号 sectionID；工艺区名称 sectionname；备注 comment。

3.设备表 deviceinfo

表 Deviceinfo 存储所有设备信息，表字段包括：设备序列号 deviceID；设备名称 devicename；所属工艺区 section，section 存储的是工艺区序列号，详细信息可在工艺区表 sectioninfo 中检索。

4.机箱信息表 rackinfo

表 rackinfo 存储所有机箱内板卡类型及状态和机箱 IP 地址、端口号，表字段包括：机箱序列号 rackID、采集卡的板卡类型 slotn_type、板卡状态 slotn_state;IP 地址 ip；UDP 端口号 UDP_port。

5.传感器表 sensorinfo

表 sensorinfo 存储传感器信息，表字段包括：传感器型号 sensorID；生产商 vender；版本 version；传感器类型 sensortype；传感器原理 principle；灵敏度（数值）sensitivity；灵敏度（单位）unit；输入参数 input type；产品链接 URL；传感器类型索引 typeID。

6.配置实例表 instance

在测点安装过程中通常一种传感器对应一种配置方法，即安装表 assembly 中的部分字段存在大量重复数据，表 instance 存储了所有配置方法，其字段信息包括：实例序列号 instanceID；传感器型号 sensorID；传感器类型序列号 type ID；输出类型序列号 outputID；报警值 alarm、危险值 danger；量程 range。

7.频率参考信息表 finstance

现场设备运行时通过电机带动主轴再经过齿轮变速，因此同一根主轴上设备频率相同，转速由主轴频率乘齿轮转速比系数得到。振动传感器测量信号幅值过小无法获得频率时，可以参考主轴电机频率进行换算。

表 finstance 存储了所有频率参考实例，表字段包括：实例序号 instance ID；主轴转速测点位置信息机箱 rack、板卡 card、通道 channel；转速传感器测量凸台齿数 gear；参考地址 16 进制表示值 ref；变速齿轮箱输入轴齿数 gearin、输出轴齿数 gearout；转速比即输入输出齿数 ratio；备注 comments。

8.数据表 rackncardn

数据表 rackncardn 表示第 n 机箱第 n 板卡，以板卡为单位存储测量信号的峰峰值 vpp、有效值 vrm、频率 freq、状态 state 以及时间戳 rectime，每张表中存储 4 路通道以及 4 路直流。另，转速类型传感器数据表 rackncardnvel 仅存储时间戳和 4 路频率值和状态；温度类型传感器数据表 rackncardntempjin 存储时间戳和 10 路温度值、状态。

表结构如以下表格所示：

Table: assembly

Columns:

spotID	int(16) PK
name	varchar(20)
sensor	varchar(20)
sensitivity	double
range_lo	double
range_hi	double
outputtype	varchar(20)
device	varchar(20)
rack	int(16)
card	int(16)
channel	int(16)
state	tinyint(1)
alarm	double
danger	double
gear	int(11)
freqref	int(11)
comments	varchar(50)

Table: deviceinfo

Columns:

deviceID	int(16) PK
devicename	varchar(45)
section	int(16)

Table: finstance

Columns:

instanceID	int(11) PK
rack	int(11)
card	int(11)
channel	int(11)
gear	int(11)
ref	int(11)
gearin	int(11)
gearout	int(11)
ratio	double
comments	varchar(45)

Table: instance

Columns:

instanceID	int(11)
sensorID	varchar(45)
typeID	int(11)
outputID	int(11)
alarm	double
danger	double
range	double

Table: rackinfo

Columns:

rackID	int(16) PK
slot1_type	varchar(45)
slot1_state	tinyint(4)
slot2_type	varchar(45)
slot2_state	tinyint(4)
slot3_type	varchar(45)
slot3_state	tinyint(4)
slot4_type	varchar(45)
slot4_state	tinyint(4)
slot5_type	varchar(45)
slot5_state	tinyint(4)
slot6_type	varchar(45)
slot6_state	tinyint(4)
slot7_type	varchar(45)
slot7_state	tinyint(4)
slot8_type	varchar(45)
slot8_state	tinyint(4)
slot9_type	varchar(45)
slot9_state	tinyint(4)
slot10_type	varchar(45)
slot10_state	tinyint(4)
slot11_type	varchar(45)
slot11_state	tinyint(4)
ip	varchar(16)
UDP_port	int(16)

Table: sensorinfo

Columns:

sensorID	varchar(20) PK
vender	varchar(50)
version	varchar(50)
sensortype	varchar(20)
principle	varchar(30)
sensitivity	double
unit	varchar(20)
inputtype	varchar(20)
URL	varchar(100)
typeID	int(11)

Table: sectioninfo

Columns:

sectionID	int(16) PK
sectionname	varchar(45)
comment	varchar(45)

Table: rackncardn

Columns:

rectime	int(11) UN PK
vpp1	double
vrm1	double
freq1	double
state1	tinyint(1)
vpp2	double
vrm2	double
freq2	double
state2	tinyint(1)
vpp3	double
vrm3	double
freq3	double
state3	tinyint(1)
vpp4	double
vrm4	double
freq4	double
state4	tinyint(1)
DC1	double
DC2	double
DC3	double
DC4	double

Table: rackncardnvel

Columns:

rectime	int(11) UN PK
freq1	double
state1	tinyint(1)
freq2	double
state2	tinyint(1)
freq3	double
state3	tinyint(1)
freq4	double
state4	tinyint(1)

Table: rackncardntemp

Columns:

rectime	int(11) UN PK
temp1	double
state1	tinyint(1)
temp2	double
state2	tinyint(1)
temp3	double
state3	tinyint(1)
temp4	double
state4	tinyint(1)
temp5	double
state5	tinyint(1)
temp6	double
state6	tinyint(1)
temp7	double
state7	tinyint(1)
temp8	double
state8	tinyint(1)
temp9	double
state9	tinyint(1)
temp10	double
state10	tinyint(1)

四、界面文档

根据项目需求，基于 LabVIEW_2016_32bit 专业版开发环境开发上位机软件。LabVIEW 是专为测试、测量和控制应用而设计的系统工程软件，可快速访问硬件和数据信息。LabVIEW 编程环境简化了工程应用的硬件集成，使用内涵的数学和信号处理 IP 来开发数据分析和高级控制算法，可与其他软件和开源语言进行相互操作。

LabVIEW 提供了可视化界面开发平台和数据流形式的程序开发语言，编程环境包括前面板和程序面板，前面板利用 LabVIEW 提供的控件对界面进行组态，程序面板中将控件之间的数据流按程序设计连接起来。

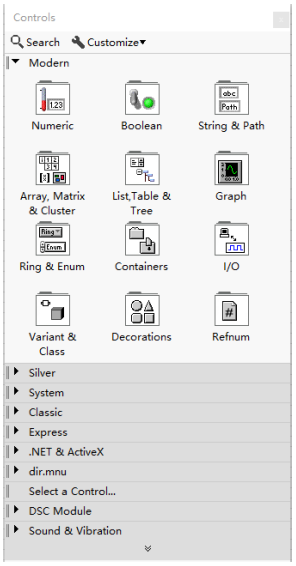


图 4.1 前面板控件

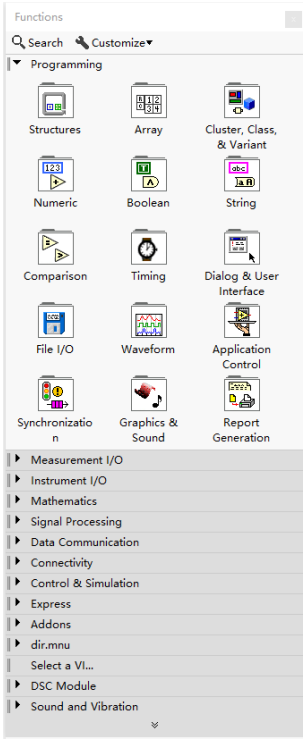


图 4.2 程序面板控件

创建工程 project，从 my computer-add 添加软件相关 VI 文件。本项目中，所有前面板显示程序保存在 pages 文件夹下，子 VI 和全局变量及数据库文件保存在 basics 文件夹下。

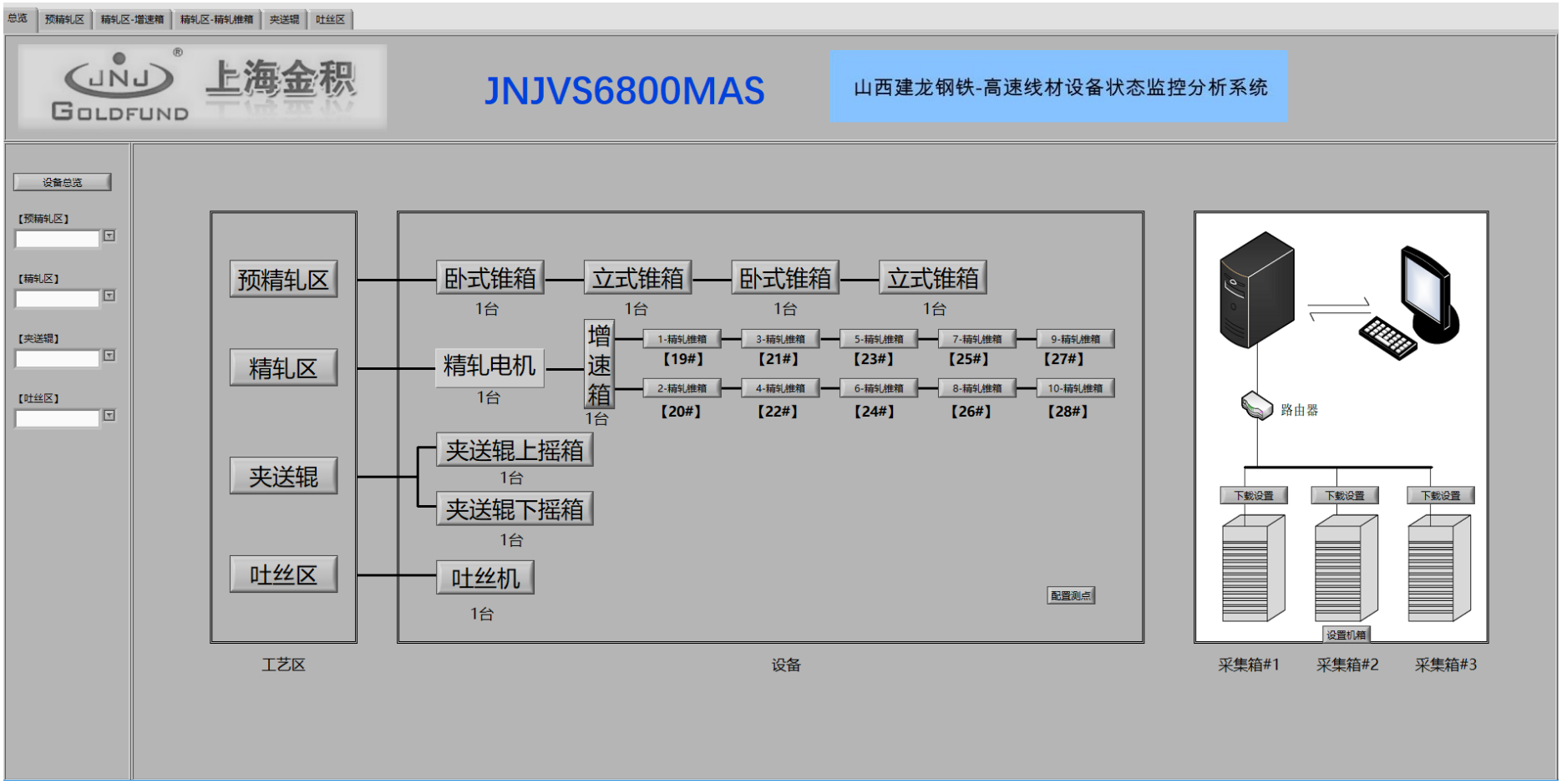


图 4.3　程序主界面

注：主界面中显示设备分布图和系统框架图，点击按钮和下拉框、工艺区框图内按钮进入二级界面；设备框图内按钮进入三级界面；“配置测点”按钮对测点各参数和传感器进行配置；“下载设置”按钮向下位机下载传感器配置参数；“设置机箱”按钮配置机箱。

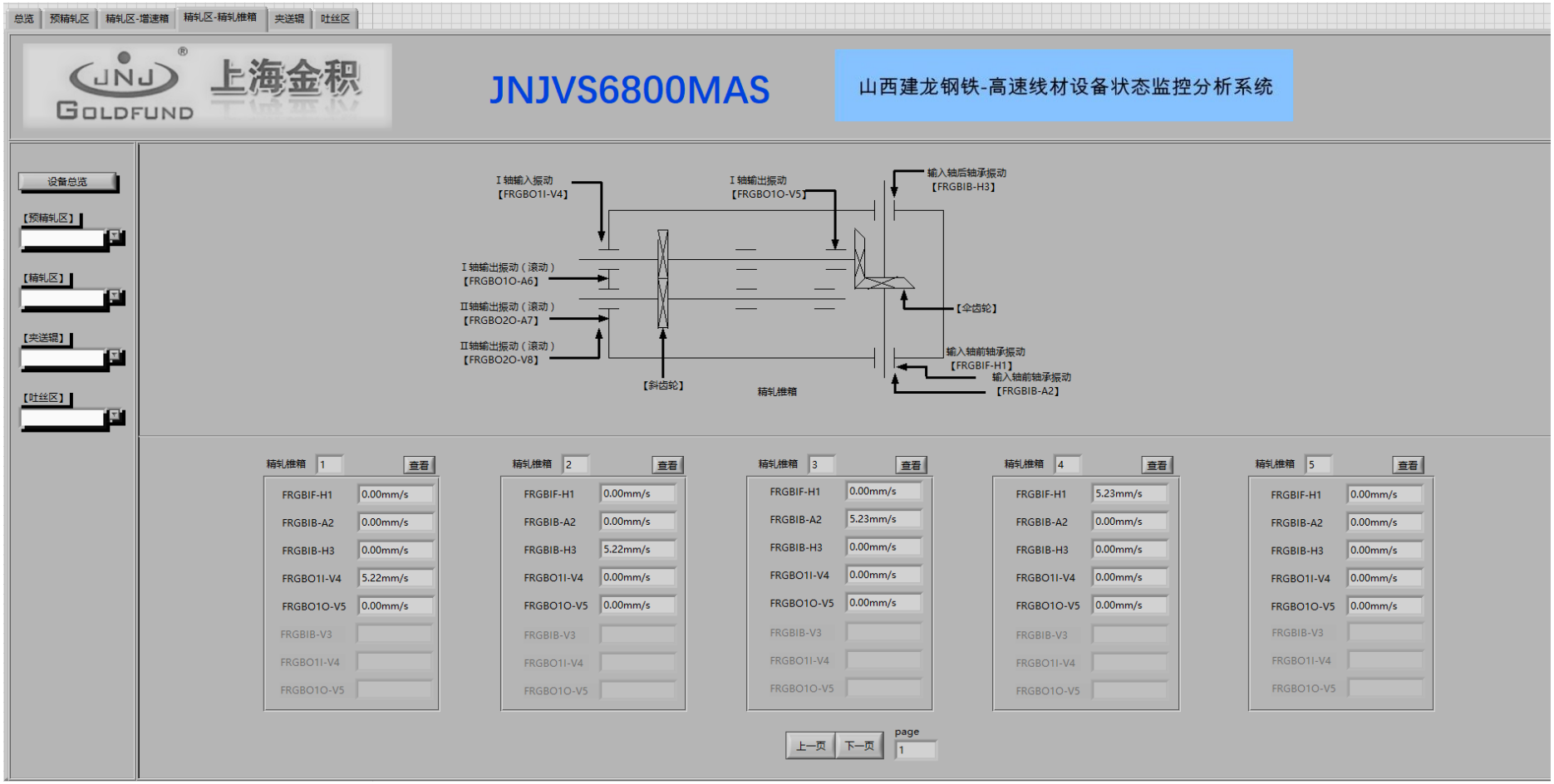


图 4.4 二级界面---精轧锥箱

注：二级界面主要显示区上部分是设备模型和测点示意图，下方显示框内显示设备总体实时测量数据，点击“查看”进入三级界面，点击“上一页”，“下一页”显示更多设备。点击左侧按钮和下拉框切换到主界面和选择的二级界面。

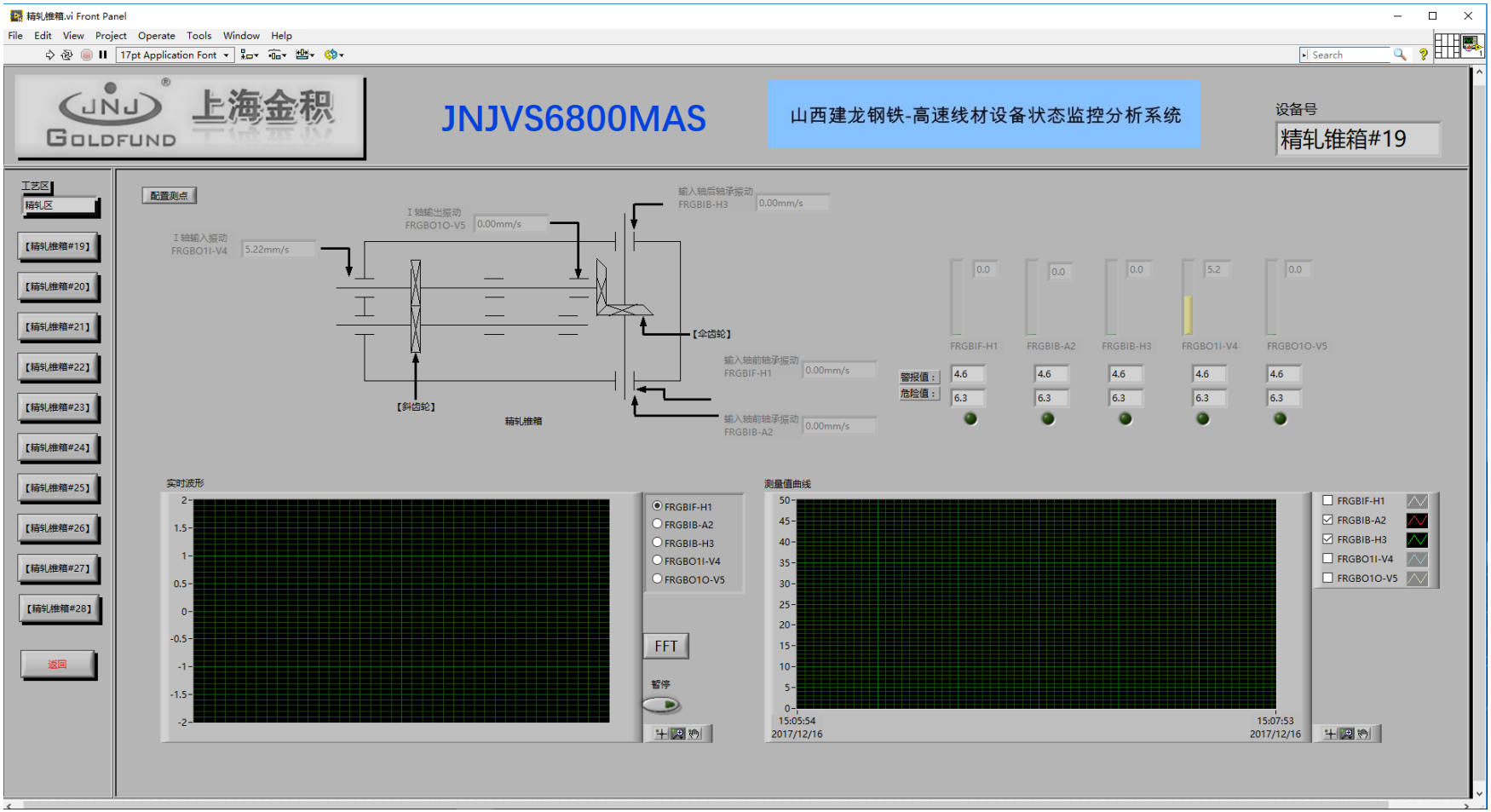


图 4.5 三级界面---精轧锥箱

注：左侧栏显示当前设备所在工艺区，点击按钮切换当前工艺区内不同的设备，主界面测点的实时测量数据在模型示意图和棒状图部分予以显示，棒状图正下方显示当前测点的警报危险值；下方两个图表分别显示实时波形和测量值曲线。

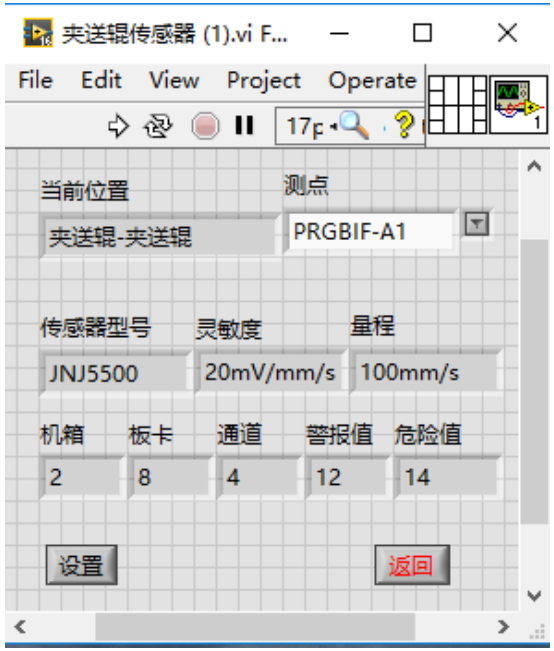


图 4.6　测点信息---夹送辊

注：从三级界面按钮“配置测点”呼出，显示当前测点传感器信息、机箱信息、警报危险值信息，测点下拉框选择当前设备不同的测点。

图 4.7　传感器参数修改---夹送辊

注：从测点信息界面“设置”呼出，修改传感器参数及添加传感器。

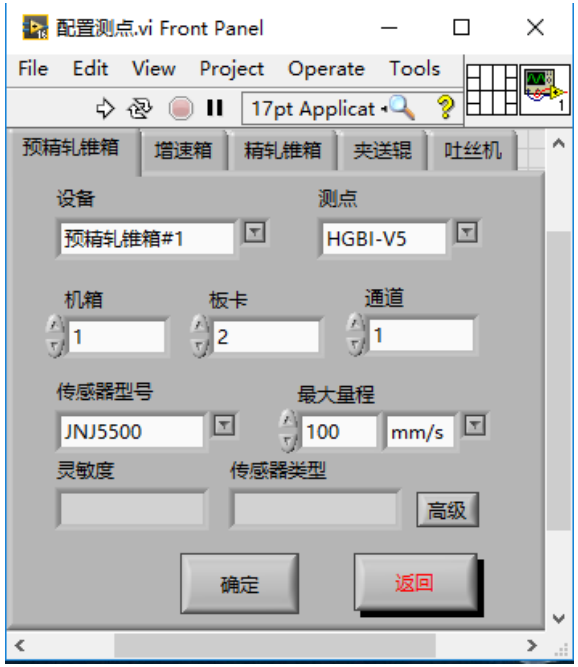


图 4.8　配置测点

注：从主界面“配置测点”呼出，可配置测点的机箱信息、选择不同的传感器和量程。点击页面 tab 标签可切换设备。

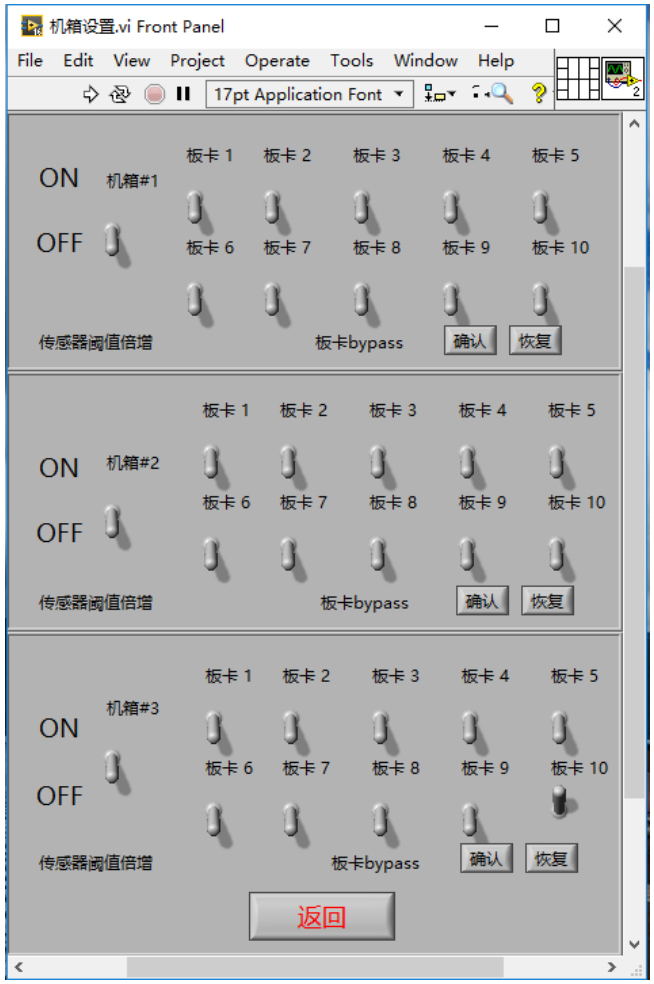


图 4.9　机箱设置

注：配置机箱传感器阈值倍增，板卡 bypass 功能

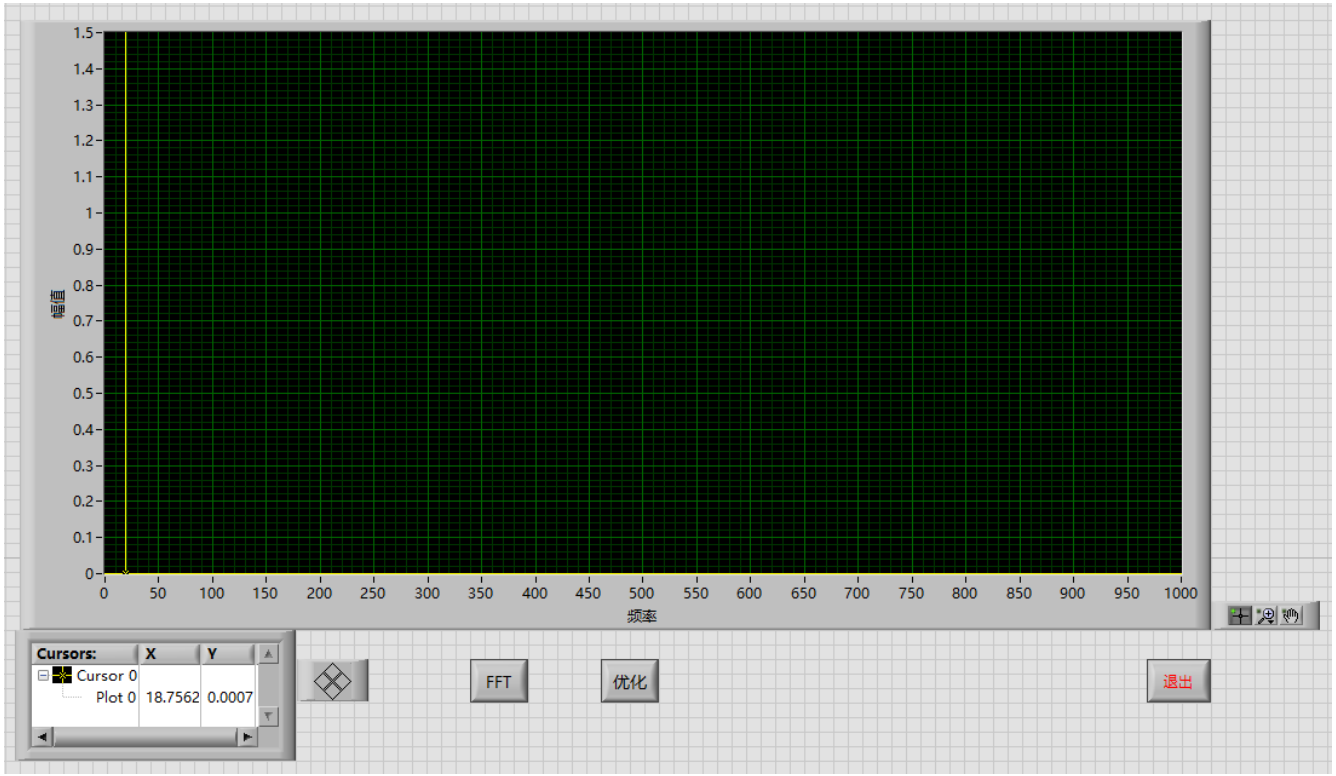


图 4.10　FFT 界面

注：从三级界面“FFT”按钮呼出，对实时波形做 FFT 处理，作为故障诊断的依据。

五、程序文档

上位机软件的核心功能是与数据库、下位机通信，获取数据并计算、显示；将配置信息写入数据库和下位机。其中，与数据库基于 ODBC 方式通信，与下位机基于 MODBUS-TCP/UDP 协议通信。

5.1 LabVIEW 连接数据库

LabVIEW 连接 MySQL 数据库有两种方式：基于 ODBC（Open Database Connectivity）技术和 ADO（ActiveX Data Objects）技术。各自连接数据库和执行 sql 的方式不同，均可以实现操作数据库。

ODBC 是微软公司开放服务结构(WOSA，Windows Open Services Architecture)中有关数据库的一个组成部分，它建立了一组规范，并提供了一组对数据库访问的标准 API（应用程序编程接口）。这些 API 利用 SQL 来完成其大部分任务。ODBC 本身也提供了对 SQL 语言的支持，用户可以直接将 SQL 语句送给 ODBC。

ADO (ActiveX Data Objects) 是一个用于存取数据源的 COM 组件。它提供了编程语言和统一数据访问方式 OLE DB 的一个中间层。允许开发人员编写访问数据的代码而不用关心数据库是如何实现的，而只用关心对数据库的连接。访问数据库时，关于 SQL 的知识不是必要的，但是特定数据库支持的 SQL 命令仍可以通过 ADO 中的命令对象来执行。

1.安装 MySQL/MySQL connector

如果链接本地数据库，需要安装 MySQL，从 mysql.com 下载 Windows 下的安装包，可以直接用 MySQL installer（x86）安装，所有的 MySQL 程序都包含在其中；也可以单独选择 MySQL server/client 进行安装。MySQL 有企业版和社区版，其中社区版免费，根据自己的开发需求选择的合适的版本。

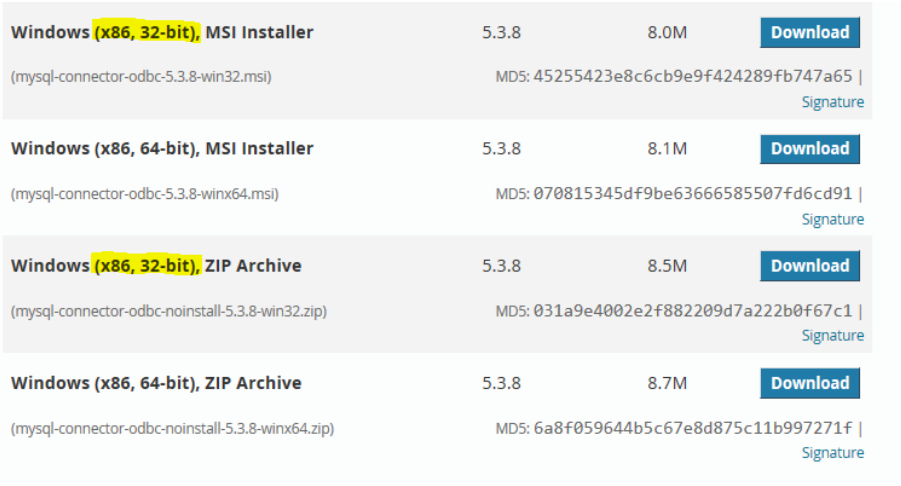


图 5.1.1 MySQL 安装版本

无论连接本地服务器还是远程服务器，本地都需要安装 MySQL connector/ODBC。数据库可以是 64 位/32 位，由于本地 LabVIEW 是 32 位，connector 应安装 32 位。否则在创建数据源时报错：“在指定 DSN 中，驱动程序

和应用程序之间的体系结构不匹配。”另需说明的是，LabVIEW 2016 64 位版部分工具包不支持，如 DCT 工具包，推荐安装 32 位。

2.安装 LabSQL 工具包

LabVIEW 的 DCT(Database Connectivity Toolkits)工具包和 LabSQL 工具包都能实现连接数据库，LabSQL 工具包是第三方平台开发的免费开源的 LabVIEW 数据库访问工具包，支持 Windows 系统中任何基于 OBDC 的数据库，将复杂的底层 ADO 及 SQL 操作封装为一列的 LabSQL VIs。利用 LabSQL 几乎可以访问任何类型的数据库，执行各种查询、对记录进行各种操作。

将 LabSQL 工具包压缩包解压到 LabVIEW 安装目录下 user.lib 文件夹中，C:\Program Files (x86)\National Instruments\LabVIEW 2016\user.lib。工具包中包含 examples 和 LabSQL ADO function VIs，解压后重启 LabVIEW，在程序面板用户库（user libraries）可以找到工具包。

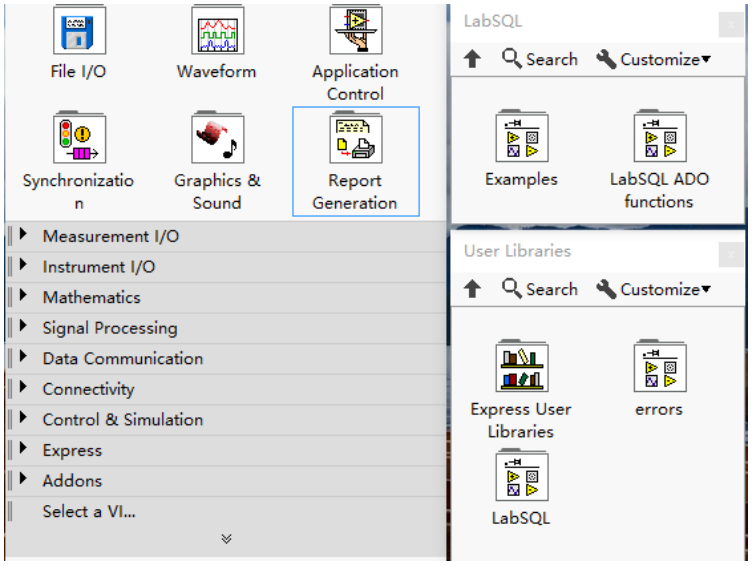


图 5.1.2 安装 LabSQL 工具包

3.创建数据源

数据源（DSN, Data Source Name）是提供某种所需要数据的器件或原始媒体，在数据源中存储了所有建立数据库连接的信息。就像通过指定文件名称可以在文件系统中找到文件一样，通过提供正确的数据源名称，找到相应的数据库连接。DSN 有三种类型：用户 DSN、系统 DSN、文件 DSN，用户和系统 DSN 都根据计算机而有所不同，DSN 信息存储在注册表中。

用户 DSN：允许单个用户在计算机上访问数据库，该数据源只对建立数据源的用户可见。ODBC 用户数据源存储了如何与指定数据库提供者连接的信息，只对当前用户可见,而且只能用于当前机器上。这里的当前机器是指这个配置只对当前的机器有效,而不是说只能配置本机上的数据库，它可以配置局域网中另一台机器上的数据库的。

系统 DSN：该数据源对当前机器上所有的用户可见，ODBC 系统数据源存贮了如何指定数据库提供者连接的信息系统数据对当前机器上的所有用户都可见,包括 NT 服务，这台机器的用户都可以访问。

文件 DSN：该数据源对安装了相同驱动的用户可见用户 DSN 只被用户直接使用,它只能用于当前机器中,ASP 不能使用它.系统 DSN 允许所有的用户登陆到

特定服务器上去访问数据库,任何具有权限有用户都可以访问系统 DSN。

打开控制面板-管理工具-ODBC 数据源（32 位），在用户 DSN 下添加 Driver:MySQLODBC 5.3 ANSI Driver。选中后点击完成，跳出数据源配置：MySQL Connector/ODBC Data Source Configuration。输入 Data Source Name，自定义即可；server 端 IP 地址，端口默认 3306，若安装 MySQL 时有改动需要注意改成相应的端口，输入 user id 和密码，database 名称，点击测试连接，连接成功后表示数据源创建完成。

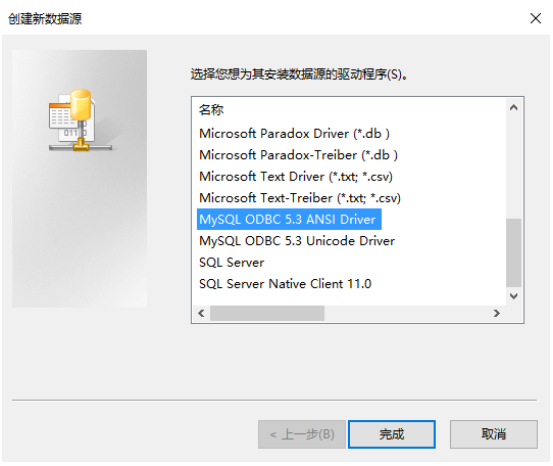


图 5.1.3 添加 ANSI Driver

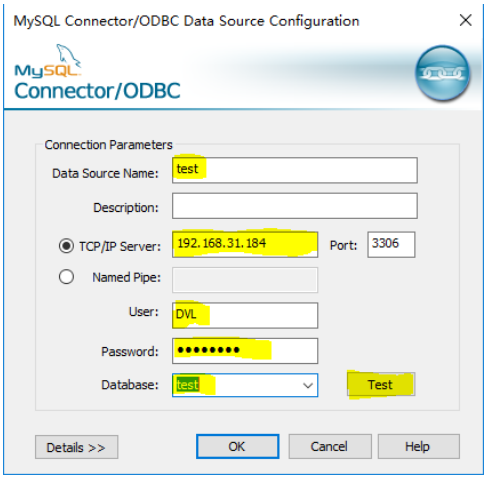


图 5.1.4 创建 DSN

4.连接数据库

打开程序面板，使用 LabSQL connection 函数库中 Create Connn 函数创建一个数据库连接，使用 Open Conn 打开连接 DSN，在 connection string 中输入：DSN=…;Driver=MySQL ODBC 5.3 ANSI Driver;DATABASE=…;UID=…;PWD=…;使用 SQL Execute 执行 SQL 命令，可配置是否输出数据，如 insert 语句不需要输出，select 语句有输出，每个 SQL Execute 函数控件执行一条语句，多条语句可顺序添加多个控件；使用 Close Conn 关闭连接，释放资源。

访问数据库查询表参考程序如下：

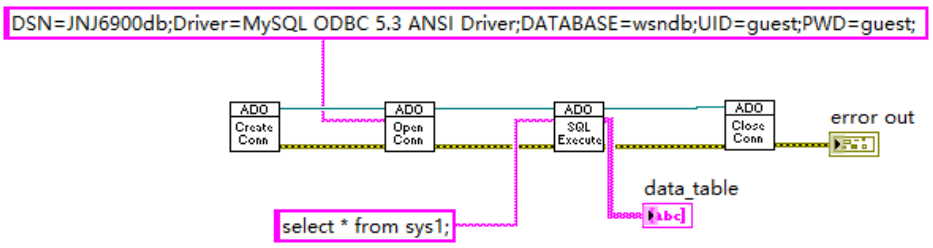


图 5.1.5 LabVIEW 访问数据库参考程序---LabSQL 工具包

使用 LabVIEW 官方数据库连接工具包（Database Connectivity Toolset,DCT）连接，区别于创建数据连接 udl 文件：在 LabVIEW 前面板或程序面板打开工具 tools—create data link---ODBC Drivers,配置数据连接属性与上述方法相同。

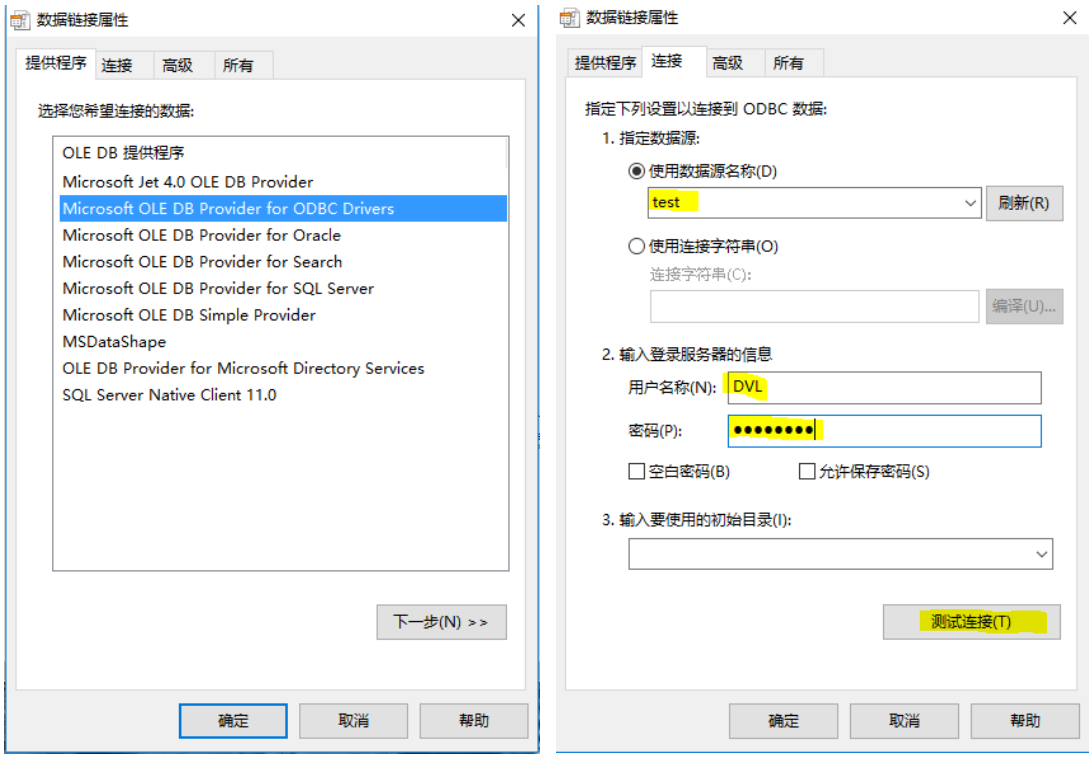


图 5.1.6 LabVIEW 环境中创建数据连接

使用 DCT 访问数据库参考程序如下：

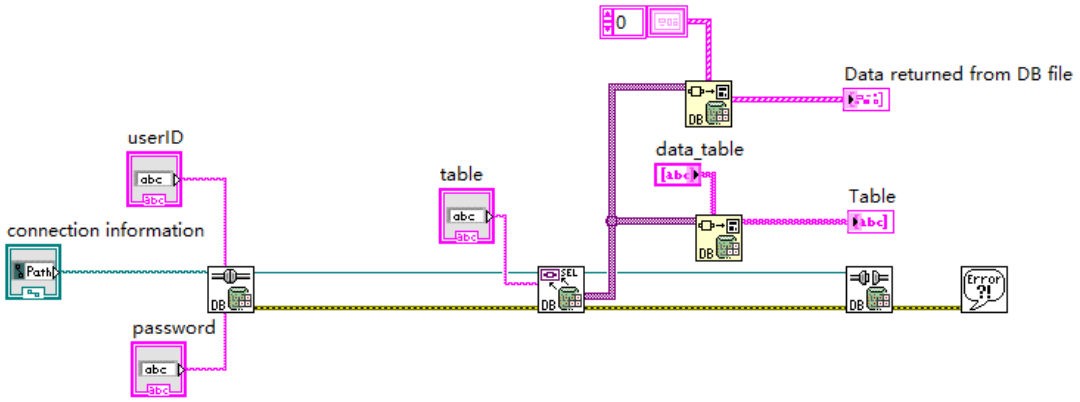


图 5.1.7 LabVIEW 访问数据库参考程序---DCT

5.2 LabVIEW 进行 UDP/TCP 通信

网际协议（IP）、用户数据报协议（UDP）和传输控制协议（TCP）是网络通信的基本工具。

IP 用于执行计算机间数据传输的低层服务。IP 将数据打包为所谓的数据报。每个数据报包含了数据及一个指明源地址和目的地址的报头。IP 决定了数据报在网络或因特网上传输的正确路径并将数据发送到指定的目的地址。IP 无法保证数据传输的成功。事实上，如果一个数据报在传输中被复制，IP 便可能不止一次地传输该数据报。程序的传输多使用 TCP 或 UDP 协议而极少使用 IP。

UDP 用于执行计算机各进程间简单、低层的通信。将数据报发送到目的计算机或端口即完成了进程间的通信。端口是发送数据的出入口。IP 用于处理计算机到计算机的数据传输。当数据报到达目的计算机后，UDP 将数据报移动到其目的端口。如目的端口未打开，UDP 将放弃该数据报。UDP 与 IP 有相同的传输问题。对传输可靠性要求不高的程序可使用 UDP。

TCP 能进行可靠的网络传输，可按顺序传输数据而毫无错误、遗失或重复。TCP 会不断地传输数据报直至收到接收响应为止。

1.在 LabVIEW 中使用 UDP

UDP 不是基于连接的协议，因此无需在发送或接受数据前先建立与目的地址的连接，但需要在发送每个数据报之前指定数据的目的地址，操作系统不报告传输错误。

在 LabVIEW 程序面板中，从 Data Communication-Protocols-UDP 打开 UDP 函数库，使用 UDP open 函数，在端口上打开一个套接字，可同时打开的 UDP 端口的数量取决于操作系统。UDP read/write 函数用于向一个目的地址读取/写入数据，端口上所有通信完毕，可使用 UDP close 函数关闭连接以释放系统资源。

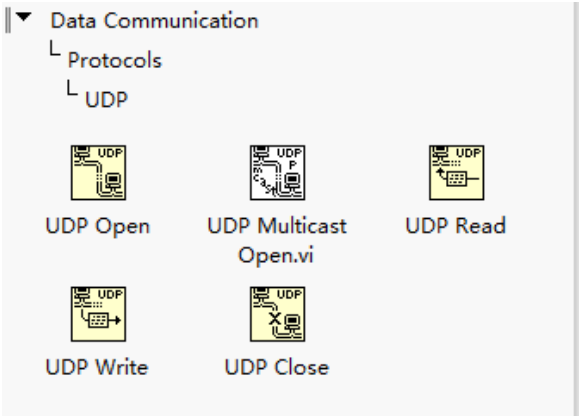


图 5.2.1 UDP 函数库

在本项目中，下位机基于 UDP 通信通过指定端口向上位机发送实时波形，一帧发送 1024 个字节 16 进制数，代表信号的电压值，上位机实时侦听端口数据，并解析为十进制一维数组在波形图表中显示。程序如下：

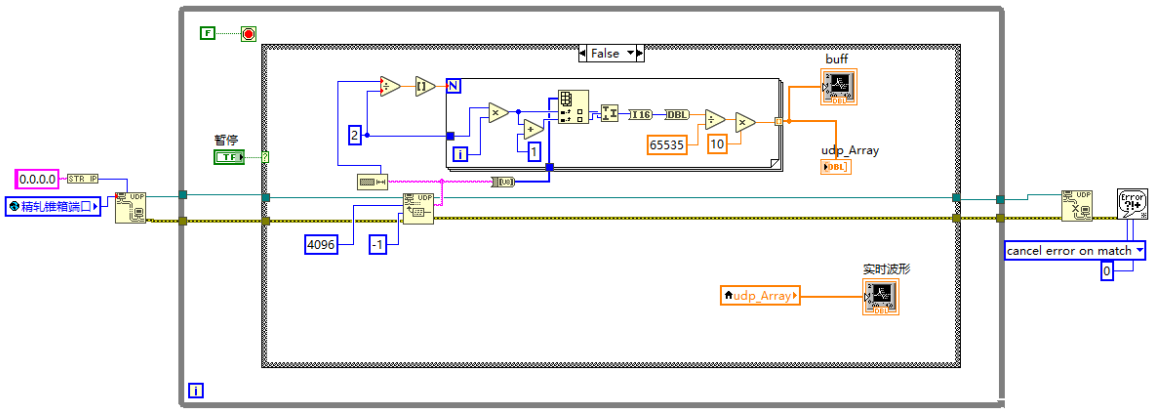


图 5.2.2 UDP 程序

2.在 LabVIEW 中使用 MODBUS TCP

本项目中，现场使用以太网作为底层设备之间的通信网络，设备之间基于 MODBUS TCP 协议通信传输控制命令和配置信息。MODBUS 协议是应用层协议，是应用于电子控制器上的一种通用语言，通过此协议，控制器之间、经以太网与其他设备之间可以通信，即定义了一种控制器能认识使用的消息结构；

下位机采集箱作为 modbus server，上位机软件为 modbus client 传输数据,主机与从机之间约定 holding register map, 即从机向相应地址写入整型数据，主机从 holding register 中相应地址读取数据进行解析。Modbus 地址为包含数据类型和偏移量的 6 个字符的数值，左边两个字符决定数据类型，右边 4 个字符是该数据类型中的序号。

本项目中，modbus holding register map 地址从 4000 开始，4001~4780 记录了机箱内 10 块板卡 4 路通道的配置信息，每个通道的长度为 19，每块板卡长度 78；4871 地址记录机箱报警值危险值倍增命令；4784~4788 存储 UDP 命令。

Adress(4000)	name	length	type	note	default
0	机箱号	2	int16		1
1	板卡#1	2	int16	板卡号	1
2	板卡BYPASS	2	int16	BYPASS板卡使能信号	0
3	通道#1	2	int16	板卡传感器通道号	1
4	传感器类型#1	2	int16	1:加速度；2：速度；3：位移	1
5	显示类型#1	2	int16	1:加速度；2：速度；3：位移	1
6 7	显示量程#1	4	float	4-20mA量程	100
8 9	计算灵敏度#1	4	float	传感器相关参数，参与输出公式计算	4
10 11	继电器报警值#1	4	float	继电器动作警报值	4.6
12 13	继电器危险值#1	4	float	继电器动作危险值	6.3
14	齿数#1	2	int16	转速公式计算用到	100
15	转速通道Reference ID	2	int16	转速通道Reference ID	1
16 17	转速比	4	float	转速比	1
18 19 20 21	备用				

图 5.2.3 modbus holding register map—板卡配置信息（部分）

784	实时数据机箱号	2	int16	第几号机箱（1~20）	1
785	实时数据板卡号	2	int16	机箱第几号板卡（1~10）	1
786	实时数据通道号	2	int16	板卡第几号通道（1~4）	1
787	实时数据端口号	2	int16	UDP端口号	8090
788	实时数据发送开关	2	int16	1：开；0：关	0

图 5.2.4 modbus holding register map---UDP 命令

在 LabVIEW 中使用 MODBUS TCP 须进行以下步骤：

（1）安装 LabVIEW 数据记录与监控（DSC）模块

在 LabVIEW 中使用 MODBUS TCP 需要安装 LabVIEW 数据记录与监控（Datalogging and Supervisory Control，DSC）模块，可在 NI 官方网站上下载。根据步骤提示安装后可在开发环境中找到 DSC 模块的函数库，利用 MODBUS 模块编程。

（2）创建 Modbus I/O Server

打开 Labview 工程文件，my computer---new---I/O Server，新建服务器

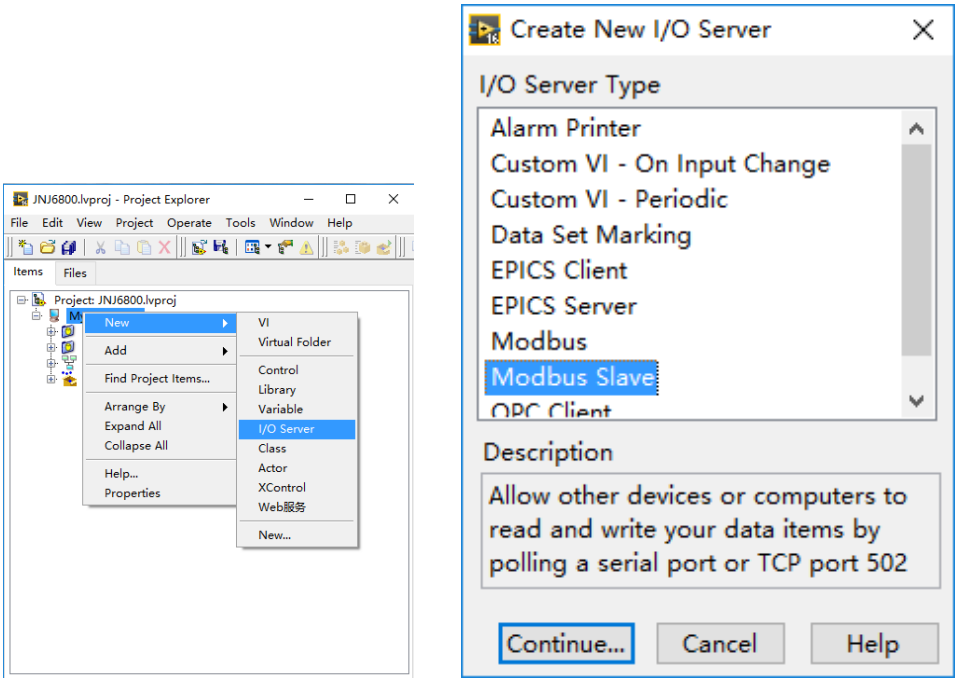


图 5.2.5 新建 modbus I/O Server

选择“Modbus Slave”，在 Model 中选择 Modbus Ethernet, Address 默认设为 1,并将生成的 labview library 保存到工程所在的文件夹中。

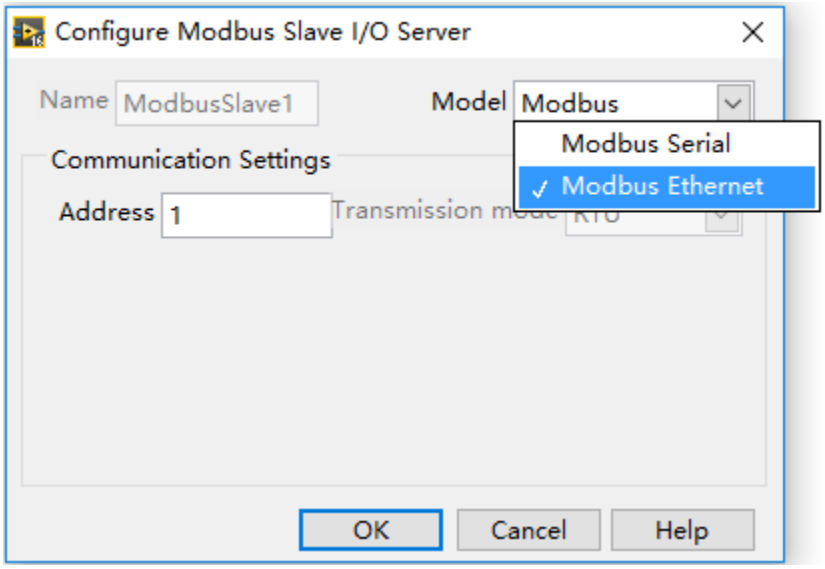


图 5.2.6 配置 modbus slave I/O server

（3）使用 Modbus 函数库连接

从 LabVIEW 程序面板打开 Modbus 函数库，Data Communication---modbus---modbus master/slave；使用 create Master Instance 创建 TCP Master，输入 Modbus Server IP 地址，端口默认 502；使用 read Holding registers 对寄存器中某地址开始读取一段长度的数据，write single/multiple holding registers 对从某一地址开始的连续地址内写入数据，single 写入单个数据，multiple 写入数组；read and write multiple holding registers 可同时进行读写操作，地址与长度可分别设置；操作完毕后使用 close 关闭 modbus 连接,释放端口资源。

Modbus 写操作程序如下，其中起始地址 starting address 为写入地址-1，即向

4000 地址内写入数据，起始地址为 3999。

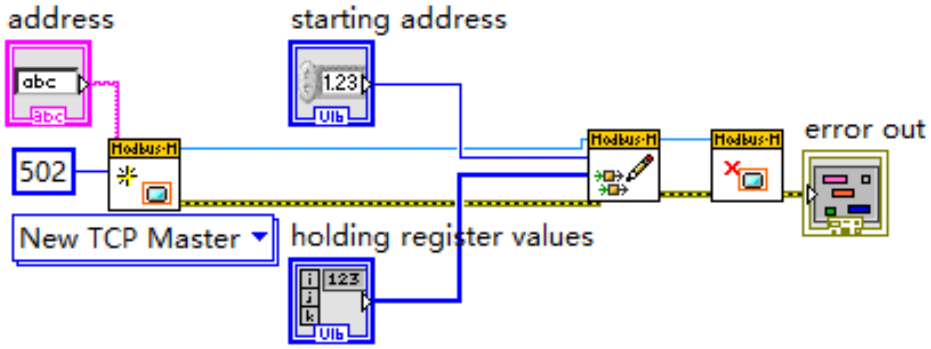


图 5.2.7 modbus 写操作程序

5.3 数据解析

上位机界面直观地反应了现场设备上测点的测量值，每一个测点对应了机箱的一个通道。根据“工艺区-设备-测点”可以唯一确定测点的物理信息；而在数据传输层面，根据“机箱-板卡-通道”可以唯一确定数据源。测点的物理信息与数据源的对应关系存储在 assembly 表中，上位机以物理信息（测点名称和设备名称）为查询条件确定数据存储的位置，再进行相应的操作。

1.获取测点温度/转速/温度值

数据库中存储的是测量信号的原始数据，根据传感器类型、灵敏度及显示类型确定转换公式，计算输出予以显示。传感器参数及转换公式如下表所示：

表 5.3.1 传感器参数

No.	型号	类型	输入参数	灵敏度
1	JNJ1010	振动速度	vrm	20 mV/mm/s
2	JNJ1010T	温度		
3	JNJ5401	振动加速度	vpp	100 mV/g
4	JNJ5500	振动速度	vrm	20 mV/mm/s
5	JNJ5701	转速	freq	

表 5.3.2 传感器数据转换公式

	加速度(g)	速度 (mm/s)	位移(μm)
加速度(g)	$\frac{V_{PP} \cdot 10^3}{2 \cdot A}$	$\frac{V_{PP} \cdot g \cdot 10^6}{4\pi \cdot A \cdot f}$	$\frac{V_{PP} \cdot g \cdot 10^9}{8\pi^2 \cdot A \cdot f^2}$
速度(mm/s)		$\frac{V_{RMS} \cdot 10^3}{A}$	$\frac{V_{RMS} \cdot 10^6}{2\pi \cdot A \cdot f}$
位移(μm)			$\frac{V_{pp} \cdot 10^3}{A}$
转速(RPM)	$\frac{60 \cdot f}{n}$		

以设备“夹送辊”为例，上位机首先查询该上测点的数据源信息、传感器参数信息、报警值危险值等配置信息，查询表字段如下：

Assembly：name、rack、card、channel、sensor、outputtype、range_lo、range_hi、alarm、danger、freqref

Sensorinfo：sensitivity、sensortype、inputtype

查询命令如下：

select assembly.name, assembly.rack, assembly.card, assembly.channel, assembly.sensor, sensorinfo.sensitivity, sensorinfo.sensortype, sensorinfo.inputtype, assembly.outputtype, assembly.range_lo, assembly.range_hi, assembly.alarm, assembly.danger from jnj6800.assembly,jnj6800.sensorinfo where assembly.sensor=sensorinfo.sensorID and assembly.device='夹送辊';

查询结果：

name	rack	card	channel	sensor	sensitivity	sensortype
PRGBIF-A1	2	8	4	JNJ5500	20	振动速度
inputtype	outputtype	range_lo	range_hi	alarm	danger	
vrm	mm/s	0	100	4.6	6.3	

name	rack	card	channel	sensor	sensitivity	sensortype
PRGB-S	1	10	3	JNJ5701	1	转速
inputtype	outputtype	range_lo	range_hi	alarm	danger	
freq	RPM		60000	3360	3600	

上表中根据机箱、板卡、通道确定所查询的数据表及字段，合成数据表查询命令，如测点 PRGBIF-A1，查询表字段如下：

Rackncardn: rectime、vpp/vrm、freq、state

查询命令如下：

select rectime, vrm4, freq4 , state4 from jnj6800.rack2card8 order by rectime desc limit 120;

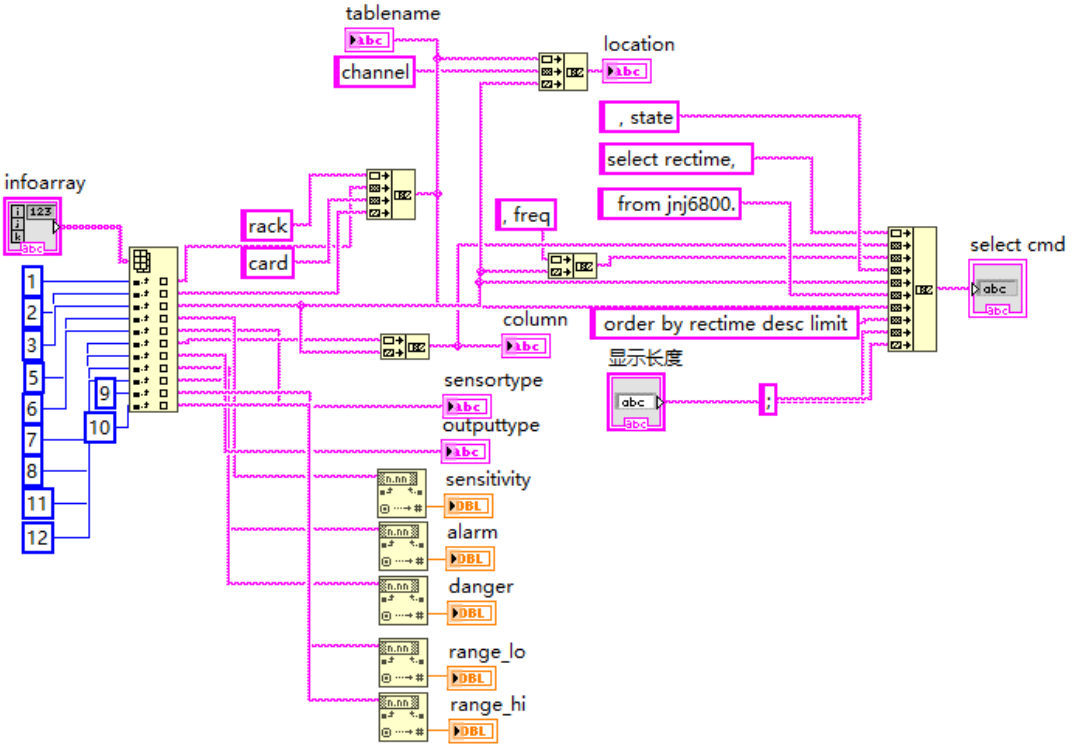


图 5.3.1 解析配置信息

查询结果：

rectime	vrm4	freq4	state4
1513318741	0	0	0
1513318740	0	0	0
1513318739	0	0	0
1513318738	0	0	0
1513318737	0	0	0
1513318736	0	0	0
1513318735	0	0	0
1513318734	0	0	0
1513318733	0	0	0
1513318732	0	0	0

图 5.3.2 数据记录（部分）

取出 vrm、频率和灵敏度用于计算振动值，state 于显示传感器是否在线状态，上述查询命令取出 120 行记录，上位机程序解析时间最新的一条记录的时间戳、vrm 值频率值和状态值进行计算，用于界面中显示框和棒状图的数据显示；取出长度为 120 的 vrm 一维数组进行计算用于测量值曲线图表的显示，长度可变。

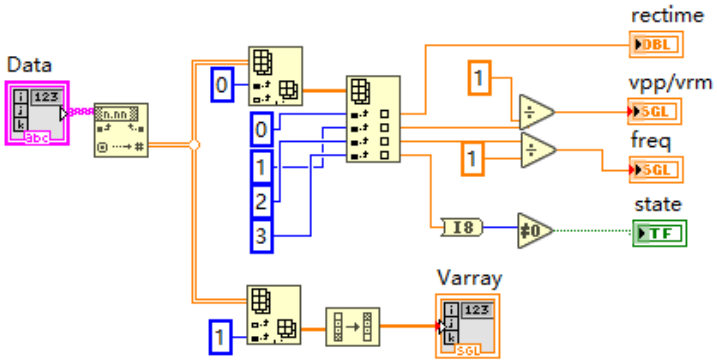


图 5.3.3 数据解析

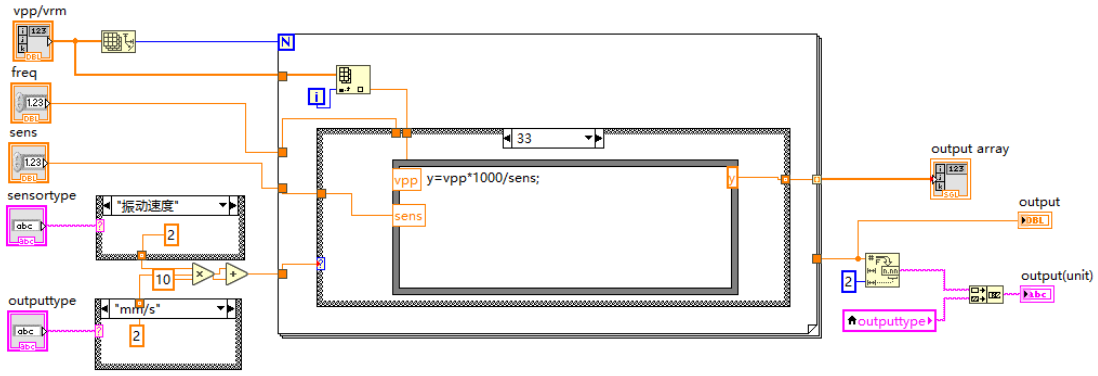


图 5.3.4 计算程序

计算程序中，首先根据传感器类型和显示类型选择计算公式，将有效值 vrm 数组、频率（取最新记录）和灵敏度作为输入参数进行计算，输出 dbl 型用于棒状图数据显示，输出数组用于曲线图表显示，输出 output 转成字符型与输出类型即单位结合为带单位的字符串显示到界面上数据显示框。

鉴于传感器信号幅值太小时，下位机无法计算频率，即数据库 rackncardn 表中频率值为 0，此时参考主轴输入端频率值。根据 assembly 表 freqref 字段的数据，参考 rackncardnvel 中响应的通道，取出频率值，乘以转速比系数得到参考频率值。

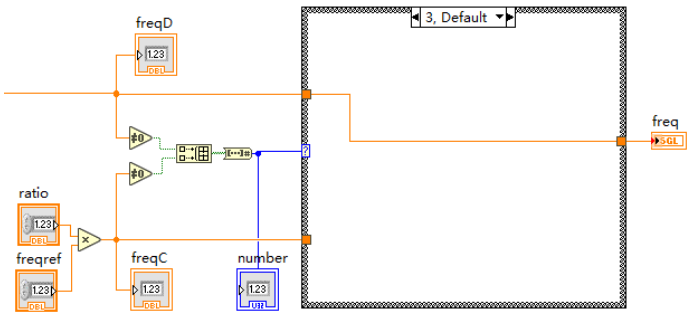


图 5.3.5 频率选择

测量值曲线图表默认显示 120 个点，长度可修改；即波形图表的输入数组长度为 120，由上文计算程序输出。波形图表 X 坐标为时间，X-scale 最大值设为 0，x-scale offset 为 -120,并将当前时刻时间戳输入到 build waveform—t0，则在前面板中 x 坐标最大值为当前时间，最小值为前 2 分钟时间。波形图表可同时显示多条曲线，点击选择框予以显示，不同曲线颜色不同。

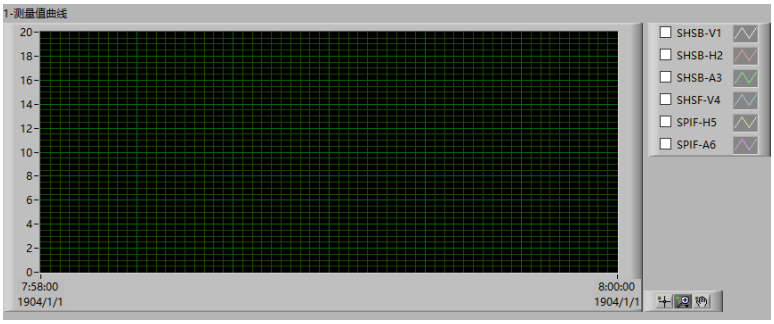


图 5.3.6 测量值曲线图表

数据库存储的时间戳是从 1970 年 1 月 1 日开始的 Unix 时间，labiew 默认时间从 1904 年 1 月 1 日开始，经过换算得到 timestamp 格式时间予以显示。

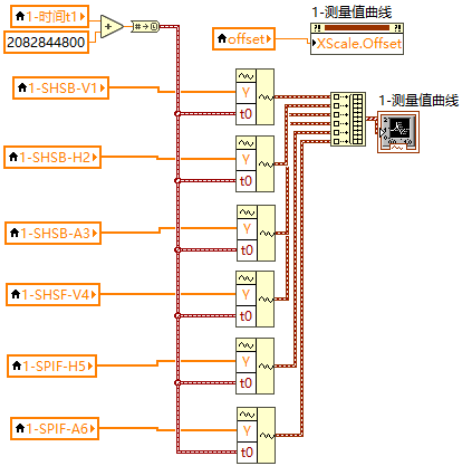


图 5.3.7 测量值曲线显示程序

2.实时波形发送命令解析

传感器信号实时波形通过 UDP 发送，上位机同一时刻只显示一路实时波形，在实时波形图标右侧设置通道单选按钮，选中某一通道后向下位机发送指令。

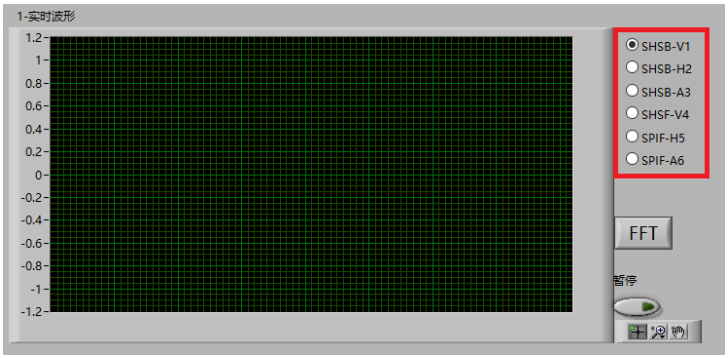


图 5.3.6 通道选择

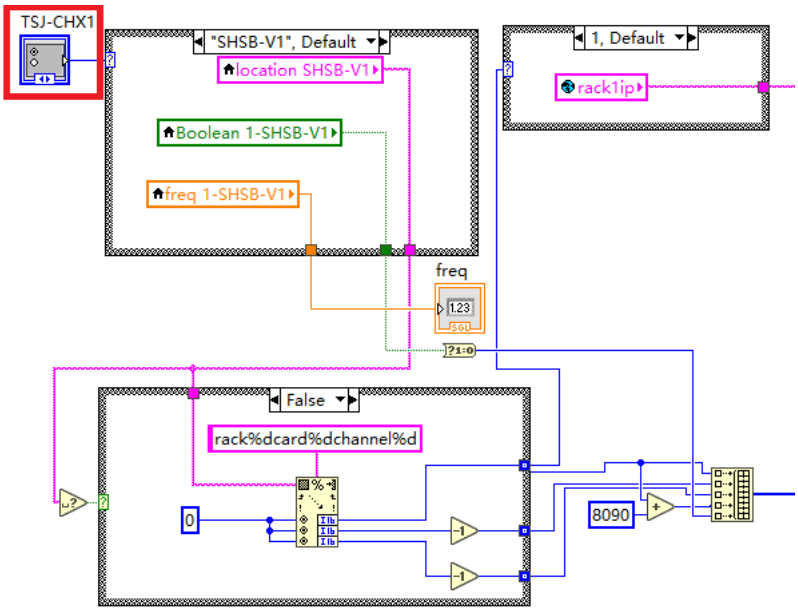


图 5.3.7 解析通道

红色框内即单选按钮，输出接入 case 框按测点名称解析机箱、板卡、通道，生成一个数组后通过 modbus 发送至下位机。

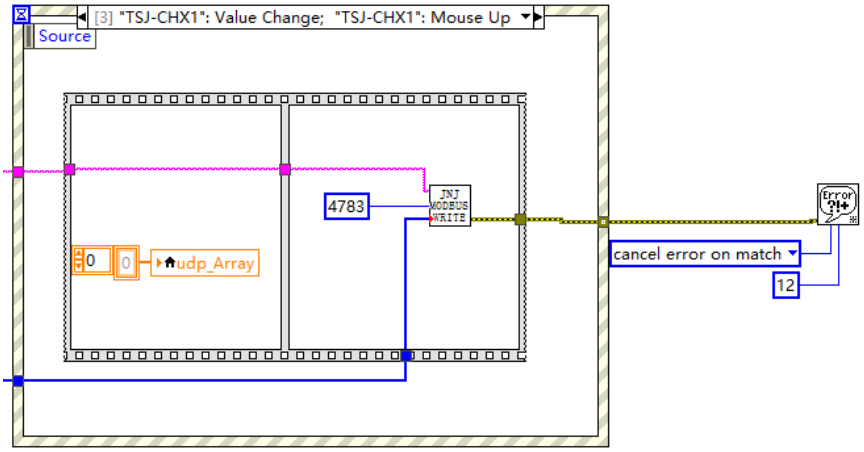


图 5.3.8 发送 UDP 指令

当单选按钮键值改变或鼠标弹起时触发事件结构，向 modbus server 发送，起始地址为 4783，数组长度 5。

784	实时数据机箱号	2	int16	第几号机箱（1~20）	1
785	实时数据板卡号	2	int16	机箱第几号板卡（1~10）	1
786	实时数据通道号	2	int16	板卡第几号通道（1~4）	1
787	实时数据端口号	2	int16	UDP端口号	8090
788	实时数据发送开关	2	int16	1：开；0：关	0

3.更新数据库配置数据

上位机可以修改传感器量程、灵敏度、警报危险值等参数，并将修改信息保存到数据库，在上位机界面进行设置后，合成 update 命令，操作数据库。更新表字段：

Assembly：sensitivity、alarm、danger、range_lo、range_hi、outputtype

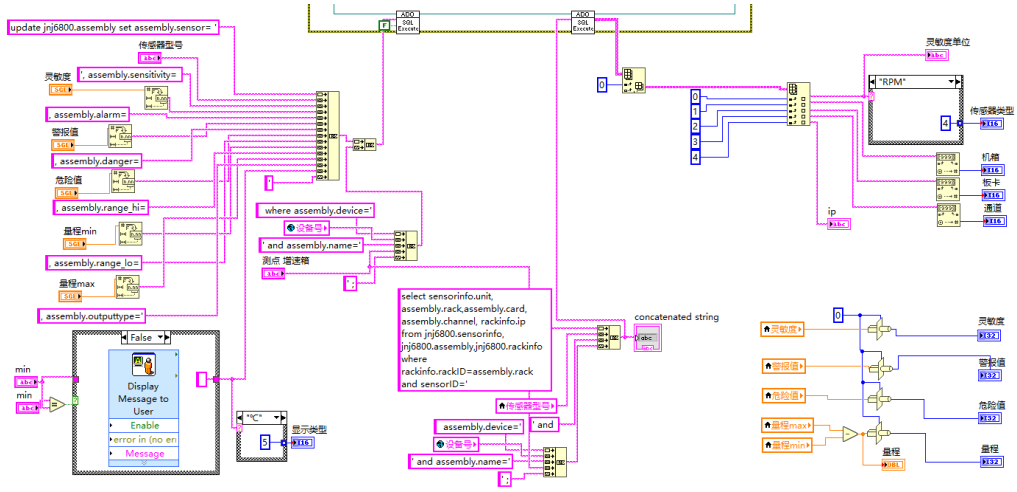


图 5.3.9 更新配置信息

为了保证实时性，上位机修改传感器参数后，通过 modbus 向下位机发送配置信息，更新 modbus holding registers 相应通道的数据。

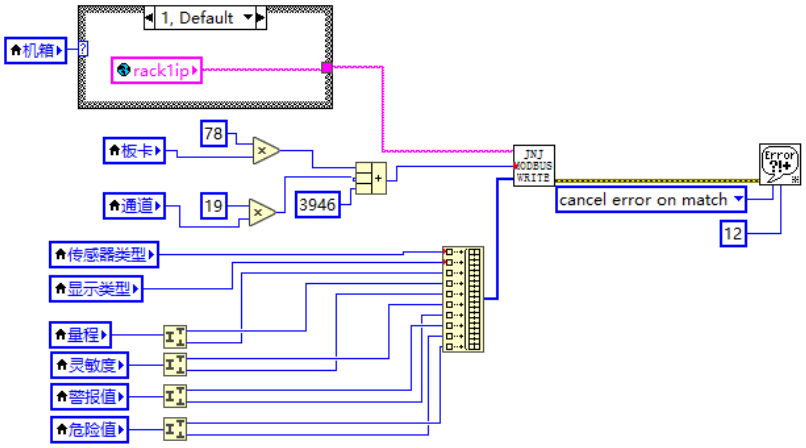


图 5.3.10 发送 modbus 指令

4.下载整机配置信息

下位机启动后，需要对寄存器进行初始配置，以机箱为单位，查询数据库所有测点配置信息，按 modbus holding register map 将信息写入一维数组中，通过 modbus 发送到下位机。每个机箱包括 10 块板卡，每块板卡 4 路通道，配置信息查询命令和结果如下：

Select assembly.spotID, assembly.rack, assembly.card, assembly.channel, sensorinfo.typeID, instance.outputID, assembly.range_lo, assembly.range_hi, sensorinfo.sensitivity, assembly.alarm, assembly.danger, finstance.gear, finstance.ref, finstance.ratio from jnj6800.assembly, jnj6800.sensorinfo, jnj6800.instance, jnj6800.finstance where sensorinfo. sensorID=assembly.sensor and instance.sensorID=sensorinfo.sensorID and assembly.freqref = finstance.instanceID order by rack,card, channel asc;

spotID	rack	card	channel	typeID	outputID	range_lo	range_hi	sensitivity	alarm	danger	gear	ref	ratio
6	1	1	1	1	1	NULL	10	100	6	8	1	417	1

图 5.3.11 配置信息查询结果（部分）

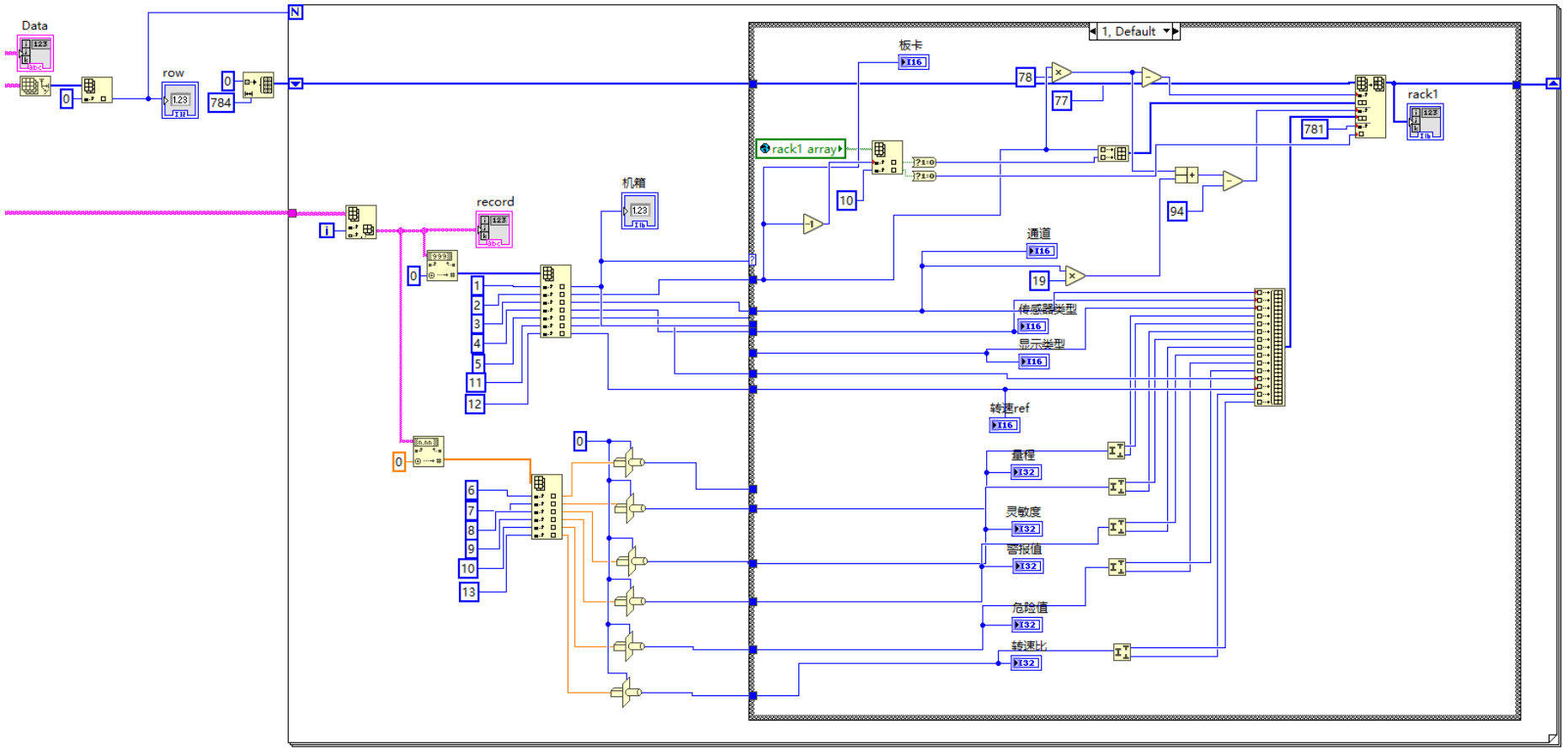


图 5.3.12 解析配置信息生成数组

Data 为数据库导出的所有测点配置信息数据，对每一条记录，判断属于 1、2、3 哪个机箱，再解析一行记录，存进相应机箱的配置信息数组中。

每个机箱包括 10 块板卡，每块板卡 4 路通道，配置信息数组长度为 784。由于 modbus 写操作一次写入的数组长度不超过 128；以板卡为单位（长度 78）发送数据。

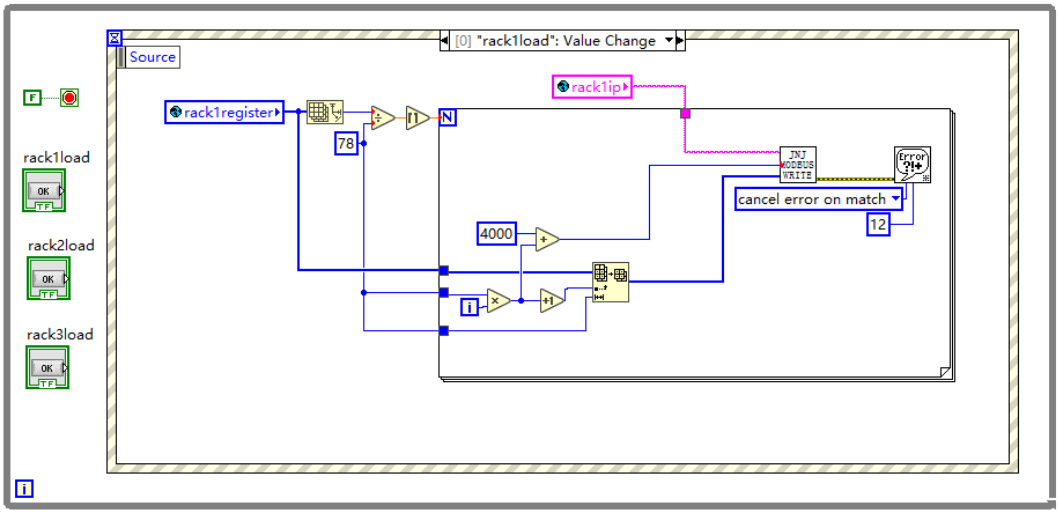


图 5.3.13 下载整机配置信息

5.4 其他界面程序

1.棒状图颜色切换

将测量值、警报值、危险值作为输入变量，棒状图属性节点 fill color 作为输出变量，当测量值小于警报值显示绿色，超过警报值、小于危险值显示黄色，超过危险值显示红色。

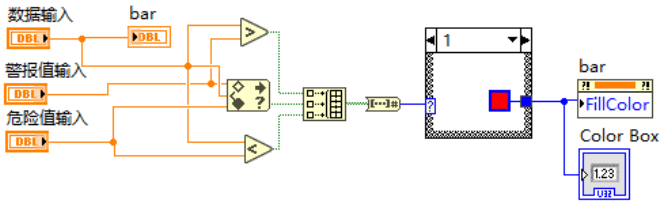


图 5.4.1 棒状图颜色切换

2.程序设置输入控件默认值

针对警报危险值、齿数等参数，为了方便修改，在界面预留输入控件直接修改，并将修改值保存到数据库。输入控件的默认值输入控件默认值与数据库的值同步。

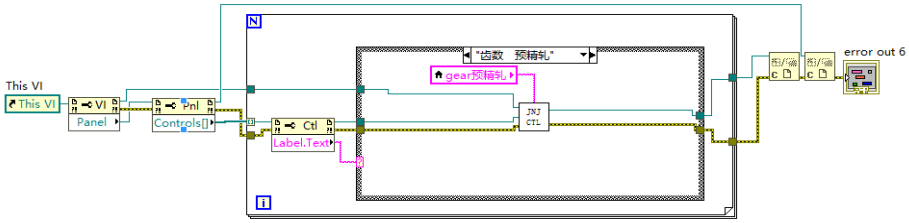


图 5.4.2 程序设置输入控件默认值

3.编译 exe 文件

打开工程文件，右击 build specifications---new---Application(EXE),将 VI 源文件编译成 exe 文件，方便移植到客户机，若客户机没有安装 labview 程序或 run-time engine，需要生成 installer。编译时将“主页.vi”作为起始文件 startup vi，即软件启动时打开的界面，其他界面添加至 always included 区域，子 VI 不需要添加。

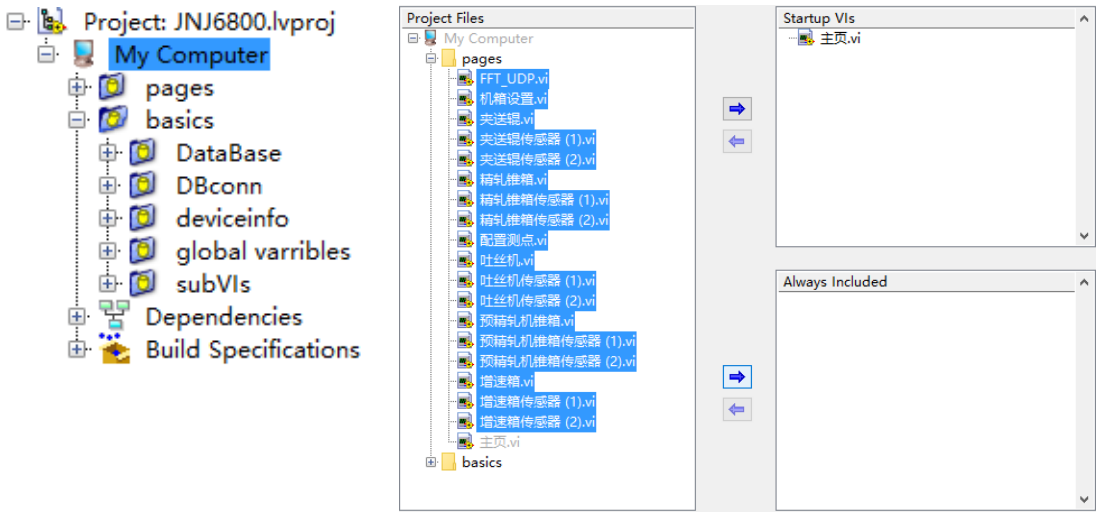


图 5.4.3　编译