上海格西信息科技有限公司

设备仿真器例子

版本 0.1

# 目录

1.	概述	3
2.	创建项目	4
	2.1 第1步新建项目	4
	2.2 第2步 添加串口设备	5
	2.3 第3步添加变量	6
	2.4 第4步 添加序列	7
	2.4.1 "启动"序列	7
	2.4.2 "电表仿真器"序列	11
	2.5 第5步添加电表仿真器设置画面	11
3.	运行项目	13
	3.1 打开电表仿真器项目	13
	3.2 打开电表数据采集器项目	13
	3.3 运行项目	14
	3.3.1 第 1 步 运行设备仿真项目	14
	3.3.2 第 2 步 使用电表数据采集器采集数据	15
	3.3.3 第 3 步 修改电表仿真器的参数并再次抄读	15

### 1. 概述

电子研发过程中,经常需要仿真一个功能或者设备,来测试另一个功能或者设备。仿真一个设备, 通常是分为两部分:一部分是仿真其通信指令;另一部分是仿真其寄存器数据。本例子通过仿真智能 电表的读功能来演示如何使用格西测控大师来仿真设备。

国内 DL/T645-2007 协议是智能电表通信协议,为主-从结构的半双工通信方式,智能电表为从站。 每帧由帧起始符、从站地址域、控制码、数据域长度、数据域、帧信息纵向校验码及帧结束符7个域 组成。每部分由若干字节组成。

帧是传送信息的基本单元。帧格式如下图所示。

说 眀	代码
帧起始符	68H
	AO
	A1
	A2
地址或	A3
	A4
	A5
帧起始符	68H
控制码	С
数据域长度	L
数据域	DATA
校验码	CS
结束符	16H

传输次序:所有数据域均先传送低位字节,后传送高位字节,发送方按字节进行加33H处理,接收 方按字节进行减33H处理。数据传输的举例:电能量值为123456.78kWh,其传输次序如图。



本例子只演示读数据命令,命令的定义如下:

主站请求帧

- a)功能:请求读电能表数据
- b) 控制码: C=11H
- c)数据域长度:L=04H(数据长度)
- d) 帧格式:



从站正常应答

a) 控制码: C=91H 无后续数据帧。



- a) 控制码: C=D1H
- b) 数据域长度:L=01H
- c) 帧格式:



本例子文件位于: <软件安装目录>\Examples\Solutions\DeviceSimulation\DeviceSimulator。

文件说明:

✓ DeviceSimulator.gpj - 设备仿真演示项目 - 中文 - 串口版

例子自带仿真器,可以脱离设备仿真运行。

串口版:需要使用串口虚拟软件,如 VSPD等,虚拟出一对串口(COM2 和 COM3)进行仿真运行。如果 虑拟的串口号和例子预定义的串口号不同,可以修改例子串口号,也可以修改虚拟串口号。

创建项目 2.

#### 2.1第1步 新建项目

启动格西测控大师,在左上角菜单中选择"新建项目",然后在弹出的"新建项目"对话框中, 填写项目名称"设备仿真器",然后点击"浏览..."按钮,选择保存路径和填写项目文件名"设备仿 真器",最后点击"确定"按钮。

其中, "启动"属性用于设置项目开始运行时自动执行的序列, 待后续创建了启动序列再设置。

💼 新發	<b></b> 建项目						ΣZ
名称:	设备仿真器						
作者:			版本:				
编码:	Unicode (UTF-8)	Ŧ	脚本:	CSharp			-
描述:							
引用:						Ð	×
	名称	路	조				
启动:	6	Ŧ	停止:	6			•
位置:	G:\DeviceSimulator.gpj					浏览	
					确定	取消	

# 2.2 第2步 添加串口设备

在项目管理器中选择"设备与接口"节点,然后点击鼠标右键,在弹出菜单中选择"新建设备..."。

弹出新建设备对话框中,选择"串口",点击"确定"。

■ 新建设备		23
通用型		
🕎 🖶 🖳 ТСРВРЖ 🖳 ТСРВВЯЖ		
🖳 UDP 🔬 环回		
Visa型		
🚏 串口 🖳 TCP设备 🚏 GPIB设备		
USB设备		
		_
确定	取消	

弹出设备属性对话框,填写"设备名"和其他设备参数,最后点击"确定"按钮。其中,"设备

名"是设备的标识,可以是任意字符串,引用设备必须使用设备名。

💙 设备	- 串口				X							
设备名:	电表接口											
串口号:	COM2	•	停止位:	1	•							
波特率:	9600	•	校验位:	Even	*							
数据位:	8	Ŧ	流控制:	None	*							
描述:	电表仿真器的串口 COM2	96(	00,8,1,e									
	确定取消											

#### 2.3 第3步 添加变量

在项目管理器中选择"变量"节点,然后点击鼠标右键,在弹出菜单中选择"新建变量容器…"。 弹出新建变量容器对话框中,填写"名称",点击"确定"。

🗌 新建	变量容器			23
名称:	电表变量表			
			确定	取消

在项目管理器中选择新创建的"电表变量表"节点,然后双击鼠标左键,或者点击鼠标右键,在 弹出菜单中选择"编辑...",打开变量编辑页面。

在"电表变量表"编辑页面,选中"电表变量表"节点,单击鼠标右键,在弹出菜单中选择"添加->变量",并设置变量的名称(Name)和值类型(ValueType)。

*								格	西测控	大师								Σζ
E.	首	π	视图	I	具	帮助									۵ 🚨	Admin	istrat	tor •
<b>()</b> 运行	合 添加	日保存	間和	×1	啓切 夏制 删除	「 撤销 」	→ 4 重做 上	移 下移	网格线	常用文件								
模式					编辑	ŧ			显示	导出				_				
哈 项	目管理器		ф	X		电表变量	表×						* X		🛠 工具箱		<u>д</u>	ιx
搜索项	目管理器			*	名称	R		数值类型		数值	数值格式	说明		1	搜索工具箱		-	<b>.</b>
🔒 设	备仿真器			^	~	🗌 电表弦	医量表	Null					-		🗌 常用变量			^
- 📑	设备与接	ŧD				🧼 🕀	误信息字	Byte		0								
	🔻 电表	接口																
v 🗋	变量																	
	🗌 电表	变量表													◎ 屋性		Ţ	I X
	序列													F	(Name)	错误的	自己	2
	画面														Capacity	0	HACK D	
															Description	-		
															Format			
															Unit			
															Value	0		
													-		ValueType	Byte		
就绪																		.::

#### 2.4 第4步 添加序列

本例子建立两个序列,分别是"电表仿真器"和"启动"序列,其中"启动"序列用于放置全局 仿真脚本和启动运行。

2.4.1 "启动"序列

双击"启动"序列节点,打开变量编辑页面,然后在工具栏中点击"脚本"按钮,进入脚本编辑 页面。在这里实现电表寄存器数据仿真和电表仿真器全局类。

```
// 电表仿真器数据
public class SimulatorData
{
    public string DI {get;set;}
    public string Value {get;set;}
    public int Length {get;set;}
}
// 电表仿真器全局类,用于模拟电表的通信状态和寄存器状态
public static class Simulator
{
    private static Dictionary<string, SimulatorData> s_data; // 保存以 DI 为 Key 的所有需
要仿真的设备数据
    ///
    public static string LastReadAddress { get; private set; } // 最近一次读命令的设备地
```

```
卝
 public static string LastReadDI { get; private set; } // 最近一次读命令的 DI
 public static string LastReadData { get; private set; } // 最近一次读命令的数据
 public static int LastReadDataLength { get; private set; } // 最近一次读命令的数据长
度
 public static int LastReadErrorCode { get; private set; } // 最近一次读命令的错误信
息码
 public static string LastWriteAddress { get; private set; } // 最近一次写命令的设备
地址
 public static string LastWriteDI { get; private set; } // 最近一次写命令的 DI
 public static int LastWriteErrorCode { get; private set; } // 最近一次写命令的错误信
息码
  11
 public static void Initialize()
   s_data = new Dictionary<string, SimulatorData>();
   s data. Add("0x02010100", new SimulatorData() {DI="0x02010100", Value="0x0001",
Length=2}); // 当前 A 相电压值
   s_data. Add("0x02010200", new SimulatorData() {DI="0x02010200", Value="0x0002",
Length=2}); // 当前 B 相电压值
   s_data. Add("0x02010300", new SimulatorData() {DI="0x02010300", Value="0x0003",
Length=2}); // 当前 C 相电压值
   s data. Add("0x02020100", new SimulatorData() {DI="0x02020100", Value="0x000001",
Length=3}); // 当前 A 相电流值
   s data. Add("0x02020200", new SimulatorData() {DI="0x02020200", Value="0x000002",
Length=3}); // 当前 B 相电流值
   s data. Add("0x02020300", new SimulatorData() {DI="0x02020300", Value="0x000003",
Length=3}); // 当前 C 相电流值
 }
 public static string GetDataValue(string di)
   if(s data. ContainsKey(di))
   {
     return s_data[di].Value;
   return null:
  }
  public static int GetDataValueLength(string di)
   if(s data. ContainsKey(di))
     return s data[di]. Length;
```

```
return 0;
 }
 public static void SetDataValue(string di, string value)
   s_data[di].Value = value;
 }
 public static bool ContainsData(string di)
 {
  return s data. ContainsKey(di);
 }
 // 读命令,在 读数据.request 协议步骤中使用,记录本次读命令的各个参数,以便在后续响应
帧中使用
 public static int Read(string address, string di)
   int errorCode = 1;
   LastReadAddress = address;
   LastReadDI = di;
   if(ContainsData(di))
   {
     LastReadData = GetDataValue(di):
     LastReadDataLength = GetDataValueLength(di);
     errorCode = 0;
   LastReadErrorCode = errorCode;
   return errorCode;
 }
 // 写命令
 public static int Write(string address, string di, string data)
   int errorCode = 1;
   11
   LastWriteAddress = address:
   LastWriteDI = di;
   if(ContainsData(di))
   {
     errorCode = 0;
   }
   LastWriteErrorCode = errorCode;
   return errorCode;
 }
```

```
// 将传入的位串按字节加 0x33, 返回新的位串
public static BitString Add33(BitString origin)
{
    byte[] data = origin.Bytes;
    for(int i = 0; i < data.Length; i++)
    {
        data[i] = (byte)(data[i] + 0x33);
    }
    return new BitString(data);
}
// 将传入的位串按字节减 0x33, 返回新的位串
public static BitString Subtract33(BitString origin)
{
    byte[] data = origin.Bytes;
    for(int i = 0; i < data.Length; i++)
    {
        data[i] = (byte)(data[i] - 0x33);
    }
    return new BitString(data);
}
</pre>
```

"启动"序列步骤的脚本,用于启动和配置仿真器运行。

```
public class Step C977A1247AFE480F94445222D33FEFA3
{
 public ScriptContext Context { get; set; }
 public Int32 BeginExecute(IStepContext context, IStep step)
   Simulator. Initialize();
   // 打开电表仿真器串口通信接口
   IDeviceSession dev = this.Context.GetDeviceSession("电表接口");
   dev. Open();
   // 启动 电表仿真器 序列
   this.Context.StartStep("电表仿真器");
   // 打开 电表仿真器 页面
   this. Context. CloseAllEditors();
   this. Context. OpenSchema("电表仿真器画面");
   return 0;
 }
  11
  public Int32 EndExecute(IStepContext context, IStep step)
   return 0;
  }
```

#### 2.4.2 "电表仿真器"序列

电表仿真器序列采用 Parallel 型步骤,并行仿真不同类型的指令,本例子用一个分支仿真读数据 指令。在独立运行的分支中,循环执行:等待读指令->判断读指令合法性->根据判断判断回应正确响 应帧和异常响应帧。

其中,"读数据.request"、"读数据.response"和"读数据.responseError"这三个 ProtocolAction 型步骤中,均配置了协议和脚本,可以通过工具栏的"协议"和"脚本"按钮打开查 看和编辑。

各个步骤的详细配置请参考本例子项目文件。



#### 2.5 第5步 添加电表仿真器设置画面

本例子建立一个画面——"电表仿真器画面",用于设置电表仿真器的参数。 画面的输入控件采用易于格式化输入的 TextEditBox 控件,格式设置使用了 Mask 和 MaskType 两 个属性。

*	格西测控大师	- • X
「「「」」「「「」」」「「」」」」	帮助	a 🙎 Administrator 🝷
○     編輯     □     矩形       运行     編輯     ○     椭圆     □       模式     工具	辺形 件 上移一层 下移一层 1 組合・ ◎ 合并・ 非列 第 組合・ ◎ 合并・ 手动 第 加 第 加 第 加 第 加 第 加 第 加 第 加 第 加	83% -
▲ 项目管理器 平 ×	□ 电表仿真器画面 ×	▼ × 😵 工具箱 🛛 म ×
搜索项目管理器     ▼            ·          ·          ·	电表仿真器参数设置画面	
	(由表参数)	
▼ 电表接口 ▼ <b>□</b> 变量	电压Ua: 0000 V 电流La: 000.000 A 说明	Border Button
<ul> <li>▶ ■ 电表变量表</li> <li>▼ □ 序列</li> </ul>	电压Ub: 0000 V 电流b: 000.000 A 过脚:	
<ul> <li>         ・          ・          ・</li></ul>	电压Uc: 0000 V 电流c: 000.000 A	
▶ 🌔 Branch (读数据)		(Name) 电表仿真 ▲
> ♥ Branch (与数据) > 📙 启动	设置	Focus
▼ 🐻 画面		GridD 10
📑 电表仿真器画面	▲ III	▶ Height 600 ▼
就绪		X: 283 Y: 286:

画面的脚本处理"设置"按钮的点击事件和画面的加载事件。



# 3. 运行项目

# 3.1 打开电表仿真器项目

从<软件安装目录>\Examples\Solutions\DeviceSimulation\DeviceSimulator目录中,打开 DeviceSimulator.gpj串口版项目文件。

♥	格西测控大师	
	帮助	🔗 🚨 Administrator 🝷
● 运行 編辑 模式 単形 1 1 1 1 1 1 1 1 1 1 1 1 1		6 -
▲ 项目管理器 平 ×		☆工具箱 平 ×
搜索项目管理器     ▼       2     设备仿真器	电表仿真器参数设置画面	eg <u>x 工具箱</u> ▼ ■ ▼ □ 常用控件 ▲ ▲
<ul> <li>✓ ■ 设备与接口</li> <li>♥ 电表接口</li> </ul>	●电表参数	
	电压Ua: 0000 V 电流Ia: 000.000 A 说明:) 用易于	Border Button
<ul> <li>▶ ■ 电表受重表</li> <li>▶ ■ 南列</li> </ul>	电压Ub: 0000 V 电流b: 000.000 A 过脚本 _	
<ul> <li>✓</li> <li>□</li> <li>□</li></ul>	电压Uc: 0000 V 电流Ic: 000.000 A	設属性 単× <mark> ゆ</mark> ダ 袋  ◎
▶ 🌔 Branch (读数据)		(Name) 电表仿真 🔺
> ◎ Branch (写数据)	2011年1月1日1日1日1日1日1日1日1日1日1日1日1日1日1日1日1日1日1	Backg
> 13 启动		Focus
● □□□ □ 电表仿真器画面		Height 600
就绪		X: 283 Y: 286:

# 3.2 打开电表数据采集器项目

从<软件安装目录>\Examples\Solutions\DeviceSimulation\DeviceSimulator 目录中,用格西烽 火软件打开 DeviceSimulatorMaster.bcp项目文件。配置串口参数为 COM3,波特率 9600,数据位 8, 停止位 1,偶校验。

0						格西知	肇火 - C	OM3 9600,8,1,	E			I	■ XX ■
	- 开始	I I	l									۵ 🍟	<b>I</b>
通信援		<ul> <li></li></ul>			服告 ・	ふ加	保存	▲ 剪切 約 算切 約 算切 約 复制 粘贴 ※ 删除 使提	属性				
	3 串□문· (	OM3	▼ 停止位	- 1	T RT	. □		湖田和島	-	that we have	竹等理器		
$  \odot$	波特率: 0	600	<ul> <li>■ 存並位</li> <li>▼ 校验位</li> </ul>	· 「 : 偶校验	- DTI	R: 🔲				▼ 會电表	数据采集器		
打开	数据位: 8	3	<ul> <li>▼ 流控制</li> </ul>	· 无	-					▼ 🔧 诸	电压		
			通信接口							6	) 读A相电压		
										6	读B相电压		
											读C相电压		
										🔹 🔧 🖌	<b>盹</b> 流		
										6	读A相电流		
										6	读B相电流		
										6	读C相电流		
就绪										发 144	1 收 165 🥹 Co	DM3 9600,8,	L,E .::

# 3.3 运行项目

#### 3.3.1 第1步 运行设备仿真项目

点击工具栏的"运行"按钮,进入运行模式。仿真项目自动运行"启动"序列,通过脚本打开电 表串口,运行"电表仿真器"序列,打开"电表仿真器画面",仿真环境准备就绪。

*			格i	西测控大师			- • X
直の	视图工具	具 帮助				۵	🚨 Administrator 🝷
▶ ■ 设计 运行 停止	▶ 序列数据 变	<ul> <li>         ・</li> <li>         ・</li> <li>         ・</li> <li>         ・</li> </ul>	● 适应 手动	<ul> <li>①</li> <li>②</li> <li>放大 缩小</li> </ul>	85% -		
模式运行		数据		缩放			
💼 项目管理器	ųх		画面				• ×
搜索项目管理器	*			电表例	百器参数	议置画面	
	•	─ 电表参数 电压Ua:	0001	V 电流Ia:	000.001	<ul> <li>A 说明:所有参数为</li> <li>田易干格式化输入</li> </ul>	」 JBCD编码。输入控 .的TextEditBox控(
		电压Ub :	0002	V 电流b:	000.002	A 过脚本将参数设置 中。	到仿真器的寄存器
▼ ¦= 电表仿具; ▼ 品 Parall > ┣ Br	暗 P* el O van.ch.(读 O	电压Uc :	0003	V 电流Ic :	000.003	А	
> <mark>◇</mark> Br > <mark>¦a</mark> 启动 マ (尋) 画面	anch (写		Ē				-
□ 电表仿真	器画面	<u>ا ا ا ا ا ا ا ا ا ا ا ا ا ا ا ا ا ا ا </u>					► ►
就绪							.:

#### 3.3.2 第2步 使用电表数据采集器采集数据

切换到格西烽火软件,在工具栏中点击"打开"按钮,打开串口,然后用鼠标双击右边面板中的 协议命令条目,即可对电表仿真器进行数据抄读。



3.3.3 第3步 修改电表仿真器的参数并再次抄读

切换到格西测控大师软件,在"电表仿真器画面",可以修改仿真器的参数值,修改后,可以切 换到格西烽火软件,再次抄读测试。