|  |
| --- |
|  |
| TEW使用指南 |
| V1.0.2.0 |
|  |
| **xujin** |
| **2014/7/28** |

|  |
| --- |
| [在此处键入文档摘要。摘要通常为文档内容的简短概括。在此处键入文档摘要。摘要通常为文档内容的简短概括。] |

Release Note

|  |  |  |
| --- | --- | --- |
| 更改人 | Note | Date |
| 陈娟 | 初始版本 | 2013.05.10 |
| 陈娟 | 增加生产测试部分功能 | 2013.05.23 |
| 许进 | 修改生产测试部分 | 2013.06.04 |
| 许进 | 增加sshv2 shell支持 | 2013.10.16 |
| 许进 | 增加sftp，direct tcp支持 | 2014.4.16 |
| 许进 | 增加对于生产测试R2.2的支持 | 2014.7.28 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

评审意见

|  |  |  |
| --- | --- | --- |
| **评审人员** | **日期** | **意见** |
| 硬件测试组+康亚龙 | 20130510 | 无 |
| 陈娟/兰明/康亚龙/许进 | 20130528 | 无 |
| 陈娟/兰明/康亚龙 | 20130604 | 无 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

目录

[1.简介 4](#_Toc369682033)

[2.界面介绍 4](#_Toc369682034)

[2.1 主界面 4](#_Toc369682035)

[2.2 界面控件 4](#_Toc369682036)

[2.2.1 菜单 4](#_Toc369682037)

[2.2.2 按钮区 5](#_Toc369682038)

[2.2.3 脚本文件设置 6](#_Toc369682039)

[3.自动测试配置文件 8](#_Toc369682040)

[3.1 配置文件(H) 8](#_Toc369682041)

[3.1.1 配置分析功能(H) 8](#_Toc369682042)

[3.2 终端通信 9](#_Toc369682043)

[3.2.1 IP(H) 9](#_Toc369682044)

[3.2.2串口(H) 9](#_Toc369682045)

[3.2.3GPIB(H) 9](#_Toc369682046)

[3.3判定 9](#_Toc369682047)

[3.4兼容顺序化的命令(M) 10](#_Toc369682048)

[4.输出结果和终端日志文件 11](#_Toc369682049)

[5.适用生产测试流程 12](#_Toc369682050)

[6.TEW平台注意事项 15](#_Toc369682051)

# 1.简介

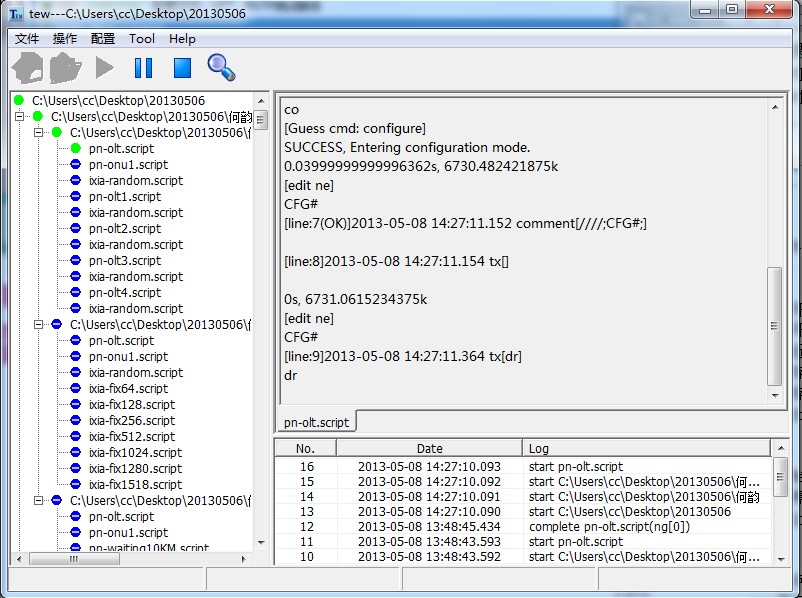
TEW软件完成按预先制定的脚本顺序进行设备或仪表的配置，并且在脚本中利用注

释行进行结果分析的功能。

其作用可以在评估测试中对比较成熟的功能进行验证，软件启动测试后无需人工干预，在测试完成时保存测试结果文件和日志文件；或在生产测试中根据配置文件自动完成仪表和设备的配置，自动完成测试结果的评判。

# 2.界面介绍

2.1 主界面



2.2 界面控件

2.2.1 菜单

文件

——打开文件

——打开目录

——最近打开

——保存

——另存为

——退出

操作

——开始执行

——暂停执行

——中止执行

配置

——节点启用

——节点禁用

——节点查询

——环境设置

Tool

——资源管理器

Help

——帮助文档

——关于tew

2.2.2 按钮区

C:\Users\cc\AppData\Roaming\Tencent\Users\467752801\QQ\WinTemp\RichOle\6_T2RMGC$G%1)HL8MQU8OHU.jpg

对应功能：

**打开文件**：选择配置文件，以.cfg结尾；

**打开文件夹**：选择脚本所在的文件夹，要求该文件夹专门用于存放脚本，最低一层的文件夹中有且仅有一个配置文件；

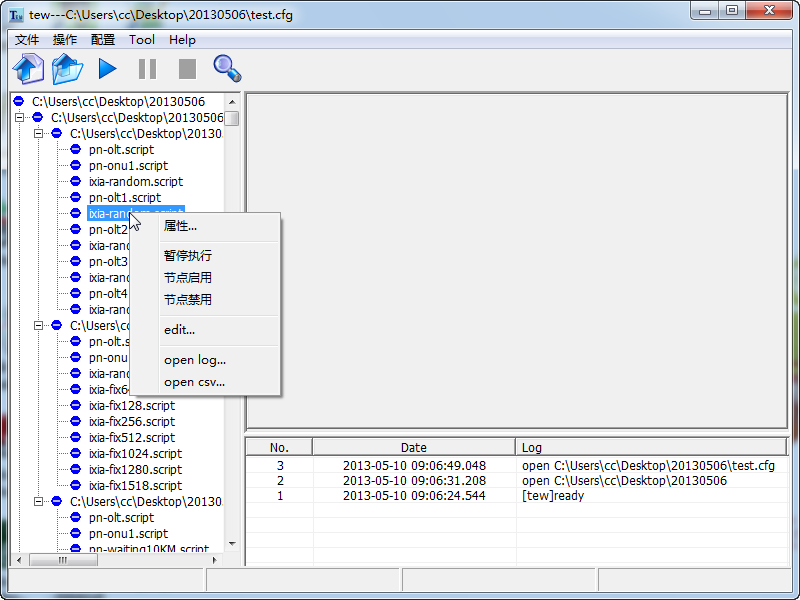
执行：打开文件配置文件或脚本文件夹情况下可点击该按钮开始测试；

**暂停**：脚本执行过程中可点击该按钮暂停测试，在未执行测试时该按钮为灰色（说明：该功能暂未实现）

**停止**：脚本执行过程中可点击该按钮停止测试，在未执行测试时该按钮为灰色，停止后再次执行测试脚本从头开始执行；

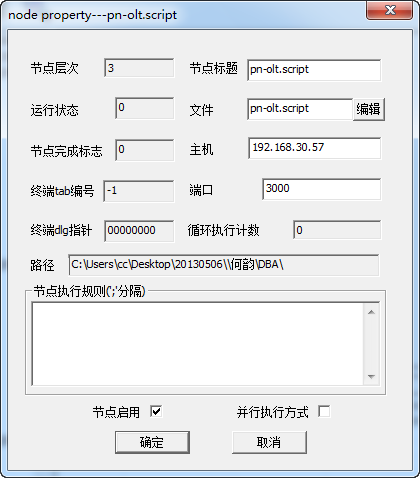
**帮助**：未实现；

2.2.3 脚本文件设置

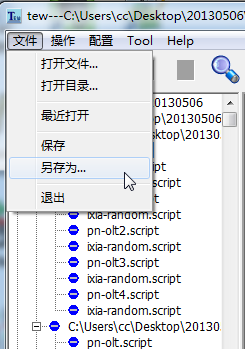


如上图示，通过右键可以设置脚本文件的属性、配置节点启用、禁用、编辑脚本内容、查看log等；

其中脚本文件属性设置内容如下：



修改节点配置相当于修改配置文件，完成修改后需要保存配置文件，操作如下：



配置文件的具体内容详见附件”test.cfg”。

# 3.自动测试配置文件

1. 兼容V1.0版本下的配置文件，可顺序执行脚本；
2. 通过文件夹执行脚本，在最底层文件夹需要有配置文件，其他文件夹不需要配置文件；
3. 通过tew平台打开脚本文件后，可通过修改脚本文件属性保存新的配置文件；

## 3.1 配置文件(H)

### 3.1.1 配置分析功能(H)

对于每个测试用例，有一个配置文件.cfg来获取脚本下发方式（串行、并行）、脚本文件结构、通信接口方式、通讯端口号等，示例如下：

pn1.script;200.0.0.1;3000;

| | |

| | 通信端口号(注1)

| 通信接口方式(此时为IP地址)

配置脚本文件名

pn23.script;200.0.0.1;23;

ixia.script;192.168.2.188;23;

37718.script;192.168.2.8;5001;

1570.script;gpib;7;

| |

| gpid地址

通信接口方式(此时为gpib接口)

pncom:com:1

| |

| com接口号

通信接口方式(此时为com接口)

注1： ”sshv2:”前缀表示为sshv2的shell连接

”directtcp:”前缀表示为direct tcp连接

脚本开发需求

* 本系统执行测试采用并行方式进行，脚本支持并行控制设备或仪表；
* 系统通过脚本判断测试的执行流程；
* 脚本编写需要保证脚本开发的效率和测试结果的可靠性；
* 测试脚本可以在3000端口下发；
* 压力测试的脚本需要将idletime设置足够长的时间；

模块化描述,定义user define tree控件外部数据，包含模块执行方式

配置模块运行指示方式:

{level;path;comment;runmtmode;enableflag;message}

脚本工作指示方式:

filename;host;port;comment;runmtmode;enableflag;message;

注: runmtmode:1表示并发执行；0表示顺序执行。Enableflag：1表示启用；0表示禁用。

## 3.2 终端通信

支持下面通信方式：

1. IP

TCP

1. GPIB

IEEE488.2

1. COM

pn COM:波特率 115200/数据位8/停止位 1/校验 None/流控 OFF;

37718 COM:波特率 9600/数据位7/停止位 1/校验 ODD/流控 OFF;

### 3.2.1 IP(H)

支持基于IP的终端连接，连接地址和端口在配置文件中指定。

例如：PN7700的3000端口，37718的5001端口

注:纯tcp连接,需要处理telnet协议将按默认方式建立连接

### 3.2.2串口(H)

支持基于RS232的终端连接，连接属性预先设置完成，与脚本和配置文件无关。但主机使用串口号需要配置文件中指定。

例如：37718的rs232，PN7700的debug串口

### 3.2.3GPIB(H)

支持基于GPIB的终端连接，连接的GPIB地址在配置文件中指定

例如：37718和1570的GPIB接口

## 3.3判定

本系统对于执行完成的测试可以形成结果文档进行输出，包括下几种形式：

Result文件：包含测试项目，测试执行时间，测试判断点结果是否OK；(H)

log文件：测试过程的日志打印（出现NG的地方明显标注，容易找到）；(H)

csv文件：测试脚本输出的相关信息；(H)

测试报告：列表给出测试执行的case清单，及执行结果，通过率等。(M)

## 3.4兼容顺序化的命令(M)

规则如下：

格式： 命令;参数1;参数2;参数3;…参数n;

如果’;’缺失,将不能解析出命令和参数

|  |  |  |  |
| --- | --- | --- | --- |
| **配置接口对象** | **命令** | **参数** | **定义** |
| **PTN**  **配置脚本** | //// | 参数1:匹配字符串 | 严格匹配:当从终端输出的数据中匹配成功，则下发后面一条配置命令,参数为空则不匹配任何字符串 |
| // | 参数1:匹配字符串 | 正常匹配:当从终端输出的数据中匹配成功，接着匹配命令提示符，然后下发后面一条配置命令 |
| //wait | 参数1:等待时间(0表示等待知道人工确定)  参数2:等待对话框内容 | 等待命令，对话框提示 |
| //cmp | 参数1:匹配字符串(grep:表示正则方式)  参数2:关键字符串(grep:表示正则方式)  参数3:比较起始位置  参数4:比较长度  参数5: 标记c([0,1023]) | 比较匹配字符串后指定区域是否存在关键字符串，并记录OK或NG；如果匹配字符串为空，则在buff中进行匹配, 最后将结果保存在标记c |
| //cmpn | 参数1:匹配字符串(grep:表示正则方式)  参数2:关键字符串(grep:表示正则方式)  参数3:比较起始位置  参数4:比较长度  参数5: 标记([0,1023]) | 比较匹配字符串后指定区域是否存在关键字符串，并记录OK或NG；如果匹配字符串为空，则在buff中进行匹配, 最后将结果保存在标记c |
| //save | 参数1:匹配字符串(grep:表示正则方式)  参数2:标记([0,1023])  参数3:起始位置(为空表示保存匹配字符串后第一个word)或者prep:表示正则方式  参数4:长度((为空表示保存匹配字符串后第一个word)) | 保存目标字符串到标记位置 |
| // saveconst | 参数1:常量字符串  参数2:保存标记([0,1023]) | 保存常量字符串到标记位置 |
| //count | 参数1:标记a([0,1023])  参数2:计算符号[=/!=/>/<]  参数3:标记b([0,1023])  参数4:标记c([0,1023]) | 根据计算符号计算标记a与标记b的结果，并记录OK或NG,并将结果保存在标记c |
| //countf | 参数1:标记a([0,1023])  参数2:计算符号[=/!=/>/<]  参数3:标记b([0,1023])  参数4:标记c([0,1023]) | 根据计算符号计算标记a与标记b的浮点计算结果，并记录OK或NG,并将结果保存在标记c |
| //buffcut | 参数1:开始字符串  参数2:结束字符串 | 匹配开始和结束字符串，裁减终端输出内容,所匹配区域是终端输出数据中的最后出现位置 |
| //csv | 参数1:标记([0,1023])或  换行标志[‘n’(换行),’’(不换行)] | 将保存标记内容写入csv文件 |
| //ctrld |  | 将EOT(0x04)写入终端(仅支持PTN设备23端口) |
| //systemcmd | 参数1:命令1  参数2:命令2  .  .  .  参数32:命令32 | 调用系统命令,最大支持32条 |
| //sleep | 参数1:等待时间  参数2:等待对话框内容 | 等待时间，终端提示 |
| //module | 参数1:模块名  参数2:begin/end | 模块指示 |
| //msg | 参数1:add/check/chkclr/clear  参数2:消息 | 并行脚本执行控制消息 |
| //ctrlstop | 参数1:标记([0,1023]) | 根据保存标记内容是否符合”NG”,则终止脚本执行 |
| //label | 参数1:label名 | 命名label |
| //goto | 参数1:label名  参数2:标记([0,1023])  参数3: 匹配字符串 | 检查标记位置与匹配字符串相等(忽略大小写)，则转label名所在行的下一行继续执行 |
| //goton | 参数1:label名  参数2:标记([0,1023])  参数3: 匹配字符串 | 检查标记位置与匹配字符串不相等(忽略大小写)，则转label名所在行的下一行继续执行 |
| //sftp | 参数1:tx/rx  参数2:local path+file  参数3: remote path+file | Sftp，tx发送或rx接收，本地文件及远端文件 |
| //ngmon | 参数1:on/off | 打开或关闭ng的监控，关闭后NG被off标识替换，并且不作为NG计数。默认为on，脚本文件内生效 |
| **PTN**  **debug脚本** | 参考ptn配置脚本，不同在于将//替换为# |  |  |
| **IXIA**  **配置脚本** | 同PTN  debug脚本 |  |  |
| **37718配置脚本** | 同PTN  debug脚本 |  |  |
| **1570配置脚本** | 同PTN  debug脚本 |  |  |
| **Direct tcp配置脚本** | #msg | 参数1:add/check/chkclr/clear  参数2:消息 | 并行脚本执行控制消息 |
| #rxbufferclear | 无参数 | 清除接收buffer |
| #wait | 参数1:等待时间  参数2:等待对话框内容 | 等待时间，终端提示 |
| #tcptx | 参数1:发送报文净荷 | 发送报文，支持心跳报文的插入发送 |
| #tcprx | 参数1:匹配字段  参数2:关键字段  参数3:掩码字段  参数4:比较偏移位置  参数5:检验时间(秒)  参数6: 标记([0,1023]) | 接收并检验报文，按匹配字段后偏移位置检验关键字段，掩码字段为’与’，检验命令持续不超过检验时间，并将结果存入标记 |
| #module | 参数1:模块名  参数2:begin/end | 模块指示 |

# 4.输出结果和终端日志文件

在选择配置文件或脚本文件夹进行测试后，在改路径会生成测试结果的文件夹，以\*\*\*zlog命名，该文件夹下，没执行一次测试，生成一个以时间命名的结果文件夹，该结果文件夹包含与执行脚本文件夹具有相同路径的结构，每个脚本有对应的log文件，同时生成一个total.txt文件，可以查看执行结果；

**文件类型说明**：

total.txt —— 该次执行结果记录，包含执行的脚本文件目录和出现NG的位置记录；

\*\*\*.log —— 其中\*\*\*为脚本文件名，该文件为\*\*\*脚本执行的打印信息；

# 5.适用生产测试流程

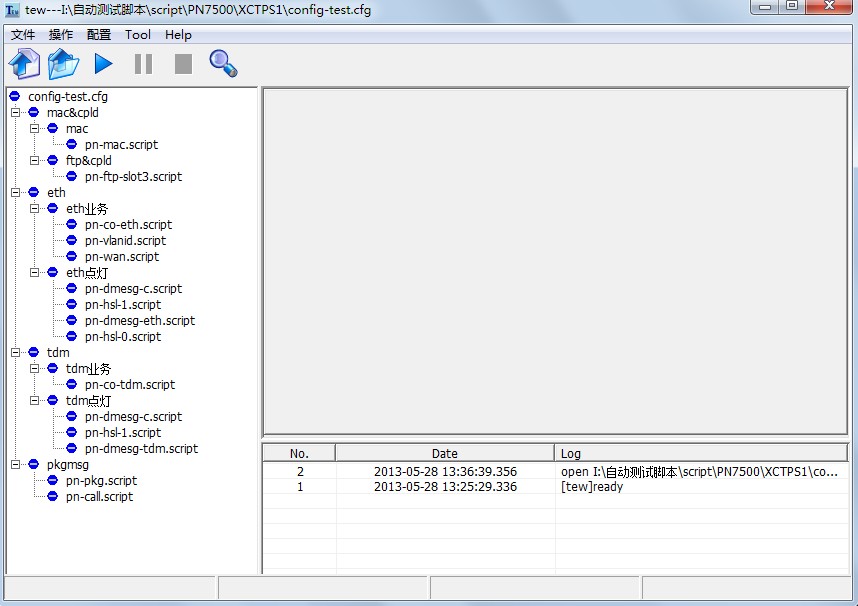
1. 生产测试配置文件

TEW在生产测试过程中，可以承担起板卡自动测试功能，需要的配置如下：

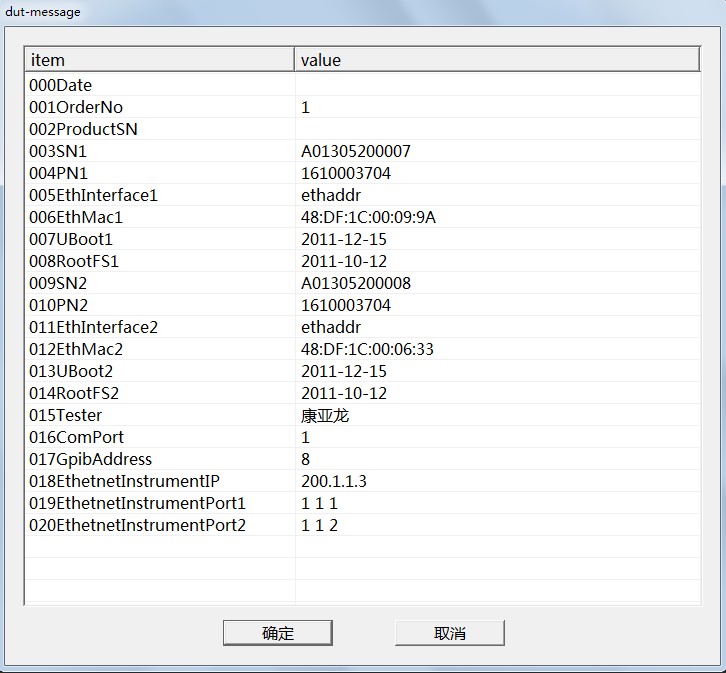
配置文件“app\_cfg.dat”中参数product test值设置为1；

1. 测试流程简介

完成生产测试配置后，选择需要测试的板卡对应的.cfg文件可以开始板卡的生产测试。以XCTPS板卡测试为例，载入测试后，界面如下：



通过右键可以在各节点配置是否启用该节点，完成配置后，点击执行按钮开始测试，程序弹出生产信息录入界面如下，测试人员需要在该界面下录入板卡信息；



对应模板位于程序文件夹下”生产测试模板”子目录,模板内容如下:

000Date=

001OrderNo=1

002ProductSN=

003SN1=

004PN1=1231111111

005EthInterface1=3

006EthMac1=40:11:11:11:11:11

007UBoot1=1111-11-11

008RootFS1=1111-22-33

009SN2=

010PN2=1234567651

011EthInterface2=

012EthMac2=11:11:11:11:11:11

013UBoot2=1111-11-11

014RootFS2=

015Tester=11

016ComPort=1

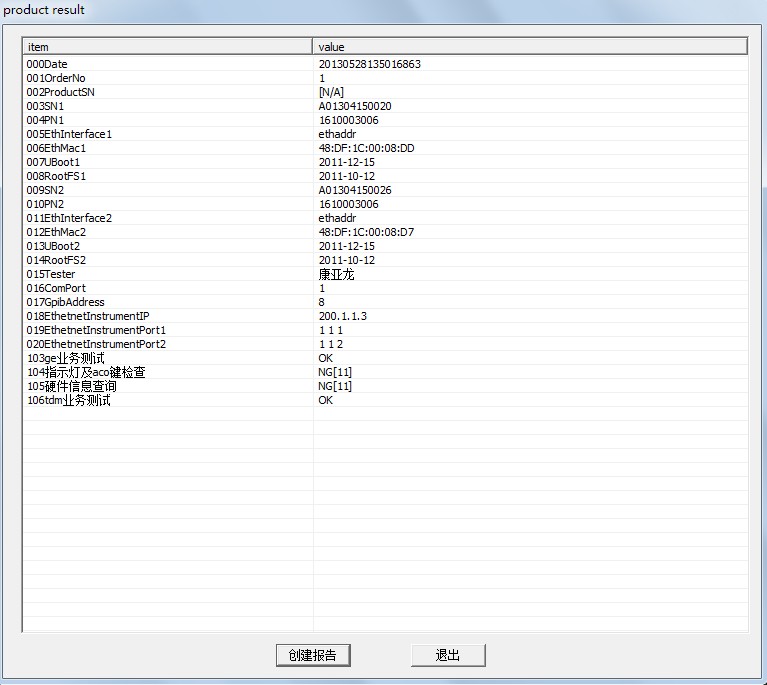
017GpibAddress=8

018EthetnetInstrumentIP=200.1.1.3

019EthetnetInstrumentPort1=1 1 1

020EthetnetInstrumentPort2=1 1 2

下一步开始测试，TEW自动进行测试流程，测试人员根据操作进行相关操作即可。

测试完成后，TEW自动生成测试结果，见下图：

注意:

1.如果002ProductSN不为空,以及003SN1不为空,则按002ProductSN生成测试报告

2.如果002ProductSN为空, 003SN1不为空,按003SN1生成测试报告,如果009SN2不为空,同时按009SN2生成测试报告

3.在脚本中引用模板信息或save变量,使用$\_+item或$\_\_+[value]方式,举例如下:

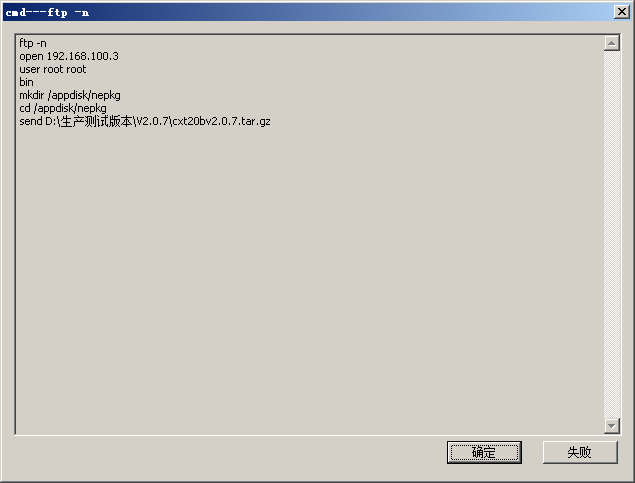
脚本文件内容: set ethaddr $\_006EthMac1

经过程序处理后实际下发内容: set ethaddr 40:11:11:11:11:11

4.用于ftp命令的脚本举例:

#systemcmd;ftp -n;open 192.168.100.3;user root root;bin;mkdir /appdisk/nepkg;cd /appdisk/nepkg;send D:\生产测试版本\V2.0.7\cxt20bv2.0.7.tar.gz;ls -la;bye;

当执行上述脚本时,出现下面对话框:



正常情况测试者需要等待对话框自行关闭

5.测试报告生成位置是:”D:\生产测试结果”,该目录中以ProductSN或SN为子目录名称,测试报

告按测试顺序生成多份测试报告

6.在多用户同时使用晨晓CXM仪表时，注意执行仪表配置时各工位需要错开执行，避免冲突。而在之后测试业务过程中可多工位同时控制。如果仪表断电则需要再次进行配置

7.注意wait对话框提示，存在点击确定和操作动作的先后次序不同

8. 对于coriant板卡，则将其二维码扫描到对应sn字段，然后继续编辑其他字段或点击确定，

程序会自动将其分解为sn和pn

# 6.TEW平台注意事项

1. 用专门的文件夹存放测试脚本，文件夹按照模块进行分类，最低一层的文件夹中为脚本文件和配置文件，每个文件夹中有且仅有一个配置文件；
2. TEW兼容autoEX平台的功能；
3. 调试时，注意空行，在某行脚本后空行也是一条命令，如果后续有匹配等命令，会导致无法找到出现NG；
4. 使用git管理脚本，主要工作流程如下：

获取远程分支——本地修改——提交修改——推送本地更新到服务器（完成）

# 正则表达式语法

正则表达式是一种文本模式，包括普通字符（例如，a 到 z 之间的字母）和特殊字符（称为“*元字符*”）。模式描述在搜索文本时要匹配的一个或多个字符串。

正则表达式示例

|  |  |
| --- | --- |
| **表达式** | **匹配** |
| /^\s\*$/ | 匹配空行。 |
| /\d{2}-\d{5}/ | 验证由两位数字、一个连字符再加 5 位数字组成的 ID 号。 |
| /<\s\*(\S+)(\s[^>]\*)?>[\s\S]\*<\s\*\/\1\s\*>/ | 匹配 HTML 标记。 |

下表包含了元字符的完整列表以及它们在正则表达式上下文中的行为：

|  |  |
| --- | --- |
| **字符** | **说明** |
| \ | 将下一字符标记为特殊字符、文本、反向引用或八进制转义符。例如，“n”匹配字符“n”。“\n”匹配换行符。序列“\\”匹配“\”，“\(”匹配“(”。 |
| ^ | 匹配输入字符串开始的位置。如果设置了 **RegExp** 对象的 **Multiline** 属性，^ 还会与“\n”或“\r”之后的位置匹配。 |
| $ | 匹配输入字符串结尾的位置。如果设置了 **RegExp** 对象的 **Multiline** 属性，$ 还会与“\n”或“\r”之前的位置匹配。 |
| \* | 零次或多次匹配前面的字符或子表达式。例如，zo\* 匹配“z”和“zoo”。\* 等效于 {0,}。 |
| + | 一次或多次匹配前面的字符或子表达式。例如，“zo+”与“zo”和“zoo”匹配，但与“z”不匹配。+ 等效于 {1,}。 |
| ? | 零次或一次匹配前面的字符或子表达式。例如，“do(es)?”匹配“do”或“does”中的“do”。? 等效于 {0,1}。 |
| {*n*} | *n*是非负整数。正好匹配 *n* 次。例如，“o{2}”与“Bob”中的“o”不匹配，但与“food”中的两个“o”匹配。 |
| {*n*,} | *n*是非负整数。至少匹配 *n*次。例如，“o{2,}”不匹配“Bob”中的“o”，而匹配“foooood”中的所有 o。“o{1,}”等效于“o+”。“o{0,}”等效于“o\*”。 |
| {*n*,*m*} | *M* 和 *n* 是非负整数，其中 *n* <= *m*。匹配至少 *n* 次，至多 *m* 次。例如，“o{1,3}”匹配“fooooood”中的头三个 o。'o{0,1}' 等效于 'o?'。注意：您不能将空格插入逗号和数字之间。 |
| ? | 当此字符紧随任何其他限定符（\*、+、?、{*n*}、{*n*,}、{*n*,*m*}）之后时，匹配模式是“非贪心的”。“非贪心的”模式匹配搜索到的、尽可能短的字符串，而默认的“贪心的”模式匹配搜索到的、尽可能长的字符串。例如，在字符串“oooo”中，“o+?”只匹配单个“o”，而“o+”匹配所有“o”。 |
| . | 匹配除“\n”之外的任何单个字符。若要匹配包括“\n”在内的任意字符，请使用诸如“[\s\S]”之类的模式。 |
| (*pattern*) | 匹配 *pattern* 并捕获该匹配的子表达式。可以使用 **$0…$9** 属性从结果“匹配”集合中检索捕获的匹配。若要匹配括号字符 ( )，请使用“\(”或者“\)”。 |
| (?:*pattern*) | 匹配 *pattern* 但不捕获该匹配的子表达式，即它是一个非捕获匹配，不存储供以后使用的匹配。这对于用“or”字符 (|) 组合模式部件的情况很有用。例如，'industr(?:y|ies) 是比 'industry|industries' 更经济的表达式。 |
| (?=*pattern*) | 执行正向预测先行搜索的子表达式，该表达式匹配处于匹配 *pattern* 的字符串的起始点的字符串。它是一个非捕获匹配，即不能捕获供以后使用的匹配。例如，'Windows (?=95|98|NT|2000)' 匹配“Windows 2000”中的“Windows”，但不匹配“Windows 3.1”中的“Windows”。预测先行不占用字符，即发生匹配后，下一匹配的搜索紧随上一匹配之后，而不是在组成预测先行的字符后。 |
| (?!*pattern*) | 执行反向预测先行搜索的子表达式，该表达式匹配不处于匹配 *pattern* 的字符串的起始点的搜索字符串。它是一个非捕获匹配，即不能捕获供以后使用的匹配。例如，'Windows (?!95|98|NT|2000)' 匹配“Windows 3.1”中的 “Windows”，但不匹配“Windows 2000”中的“Windows”。预测先行不占用字符，即发生匹配后，下一匹配的搜索紧随上一匹配之后，而不是在组成预测先行的字符后。 |
| *x*|*y* | 匹配 *x* 或 *y*。例如，'z|food' 匹配“z”或“food”。'(z|f)ood' 匹配“zood”或“food”。 |
| [*xyz*] | 字符集。匹配包含的任一字符。例如，“[abc]”匹配“plain”中的“a”。 |
| [^*xyz*] | 反向字符集。匹配未包含的任何字符。例如，“[^abc]”匹配“plain”中的“p”。 |
| [*a-z*] | 字符范围。匹配指定范围内的任何字符。例如，“[a-z]”匹配“a”到“z”范围内的任何小写字母。 |
| [^*a-z*] | 反向范围字符。匹配不在指定的范围内的任何字符。例如，“[^a-z]”匹配任何不在“a”到“z”范围内的任何字符。 |
| \b | 匹配一个字边界，即字与空格间的位置。例如，“er\b”匹配“never”中的“er”，但不匹配“verb”中的“er”。 |
| \B | 非字边界匹配。“er\B”匹配“verb”中的“er”，但不匹配“never”中的“er”。 |
| \c*x* | 匹配 *x* 指示的控制字符。例如，\cM 匹配 Control-M 或回车符。*x* 的值必须在 A-Z 或 a-z 之间。如果不是这样，则假定 c 就是“c”字符本身。 |
| \d | 数字字符匹配。等效于 [0-9]。 |
| \D | 非数字字符匹配。等效于 [^0-9]。 |
| \f | 换页符匹配。等效于 \x0c 和 \cL。 |
| \n | 换行符匹配。等效于 \x0a 和 \cJ。 |
| \r | 匹配一个回车符。等效于 \x0d 和 \cM。 |
| \s | 匹配任何空白字符，包括空格、制表符、换页符等。与 [ \f\n\r\t\v] 等效。 |
| \S | 匹配任何非空白字符。与 [^ \f\n\r\t\v] 等效。 |
| \t | 制表符匹配。与 \x09 和 \cI 等效。 |
| \v | 垂直制表符匹配。与 \x0b 和 \cK 等效。 |
| \w | 匹配任何字类字符，包括下划线。与“[A-Za-z0-9\_]”等效。 |
| \W | 与任何非单词字符匹配。与“[^A-Za-z0-9\_]”等效。 |
| \x*n* | 匹配 *n*，此处的 *n* 是一个十六进制转义码。十六进制转义码必须正好是两位数长。例如，“\x41”匹配“A”。“\x041”与“\x04”&“1”等效。允许在正则表达式中使用 ASCII 代码。 |
| \*num* | 匹配 *num*，此处的 *num* 是一个正整数。到捕获匹配的反向引用。例如，“(.)\1”匹配两个连续的相同字符。 |
| \*n* | 标识一个八进制转义码或反向引用。如果 \*n* 前面至少有 *n* 个捕获子表达式，那么 *n* 是反向引用。否则，如果 *n* 是八进制数 (0-7)，那么 *n* 是八进制转义码。 |
| \*nm* | 标识一个八进制转义码或反向引用。如果 \*nm* 前面至少有 *nm* 个捕获子表达式，那么 *nm* 是反向引用。如果 \*nm* 前面至少有 *n* 个捕获，则 *n* 是反向引用，后面跟有字符 *m*。如果两种前面的情况都不存在，则 \*nm* 匹配八进制值 *nm*，其中 *n*和 *m* 是八进制数字 (0-7)。 |
| \*nml* | 当 *n* 是八进制数 (0-3)，*m* 和 *l* 是八进制数 (0-7) 时，匹配八进制转义码 *nml*。 |
| \u*n* | 匹配 *n*，其中 *n* 是以四位十六进制数表示的 Unicode 字符。例如，\u00A9 匹配版权符号 (©)。 |