HiChannel

工具使用指南

文档版本　01

发布日期　2014-05-23

商标声明

 、HISILICON、海思和其他海思商标均为深圳市海思半导体有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受海思公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，海思公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

深圳市海思半导体有限公司

地址：　深圳市龙岗区坂田华为基地华为总部　邮编：518129

网址：　http://www.hisilicon.com

客户服务邮箱：　support@hisilicon.com

前　言

概述

本文档主要介绍 HiChannel 信道调试工具的使用方法。

产品版本

与本文档相对应的产品版本如下。

产品名称	产品版本
Hi3712	V100
Hi3136	V100
Hi3137	V100

读者对象

本文档（本指南）主要适用于以下工程师：

- 技术支持工程师
- 软件开发工程师
- 芯片开发工程师

作者信息

章节号	章节名称	作者信息
全文	全文	Y00250933/F00107764

修订记录

修订记录累积了每次文档更新的说明。最新版本的文档包含以前所有文档版本的更新内容。

修订日期	版本	修订说明
2014-01-15	00B01	第一次临时版本发布。
2014-05-23	01	修改第 4、5 章节。

目 录

插图目录

1 概述

1.1 功能介绍

HiChannel 工具是用于调试海思信道产品的 PC 端软件，该工具分客户版和专业版。

客户版对海思客户和海思技术支持工程师发布，具有如下功能：

- 支持以太网接口或者 USB 接口与目标板连接。
- 用户可以通过指示灯(Indicator)、跟踪显示（Trace）和常用监控（Monitor）对信道芯片的运行状态进行监控。
- 支持查看和编辑信道芯片的寄存器，方便定位信道问题。
- 支持通过图形的方式观察错误率，CN，星座图，频谱图，时域图等。

专业版只对海思内部开发人员发布，具有如下功能：

- 支持以太网接口或者 USB 接口与目标板连接。
- 支持开发和调试包含信道的 ASIC 芯片或者 FPGA
- 按钮（Button）、指示灯(Indicator)、跟踪显示（Trace）和常用监控（Monitor）等控件对应的操作支持用户自定义，用户可根据目标芯片通过编写 C 脚本（C-script）实现对控件操作的定义。
- 用户可以快速建立针对目标芯片（或 FPGA）的配置文件和 C 脚本文件，配置文件和 C 脚本文件可以随时加载。
- 支持查看和编辑信道芯片的寄存器。
- 支持用户自定义寄存器的查看和编辑方式。
- 支持通过图形的方式观察错误率，CN，以及星座图，频谱图，时域图等。

1.2 使用前准备

HiChannel 工具使用前的环境准备如下：

步骤 1 把位于 SDK 发布包中的 HiTool-X.X.X.zip 及 jre-6u1-windows-i586-p-s.rar（路径：$SDK_DIR/ tools/windows/HiTool|jre），拷贝到 PC 上（PC 要求安装 Win7、XP 操作系统）的某个本地硬盘。

需预装 jre1.6 及其以上版本，windows 环境必须预安装：jre-6u1-windows-i586-p-s.rar，否则无法运行。

步骤 2 解压 HiTool-X.X.X.zip，点击 HiTool_vX_X_X.exe。

步骤 3 单板连上网线，并配置单板 IP，确保单板可以 PING 通运行 HiChannel 工具的 PC 机。如果使用“USB 转 I2C”转换器，则将 USB 一侧与 PC 的 USB 接口相连，I2C 一端与目标单板相连，第一次使用“USB 转 I2C”转换器，需要安装 USB 驱动，驱动安装请参考 1.3 。

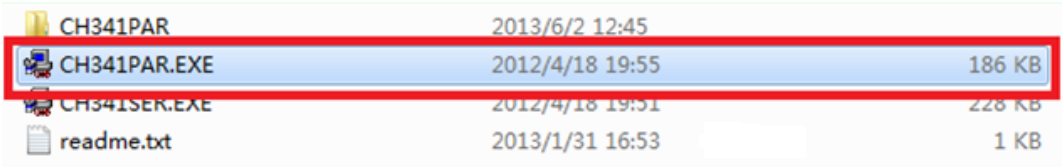
----结束

1.3 USB 驱动安装

工具支持用户使用“USB 转 I2C”转换器直接通过 PC 的 USB 接口访问目标单板的 I2C，无需解码芯片参与，并省去了串口和网口，使调试更加方便。在使用转换器前，需要在 PC 上先安装相应的驱动程序，步骤如下：

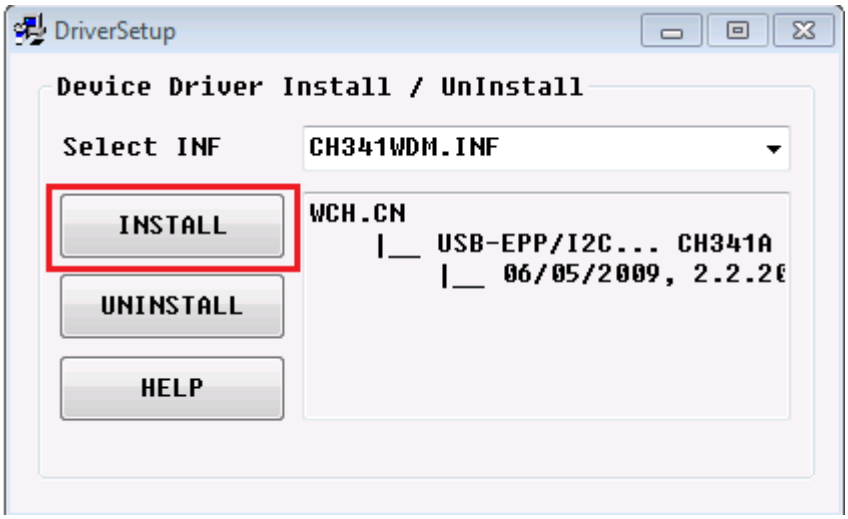
步骤 1 如图 1-1 所示，找到发布包内的 CH341PAR.EXE 文件，双击。

图1-1 选择驱动文件



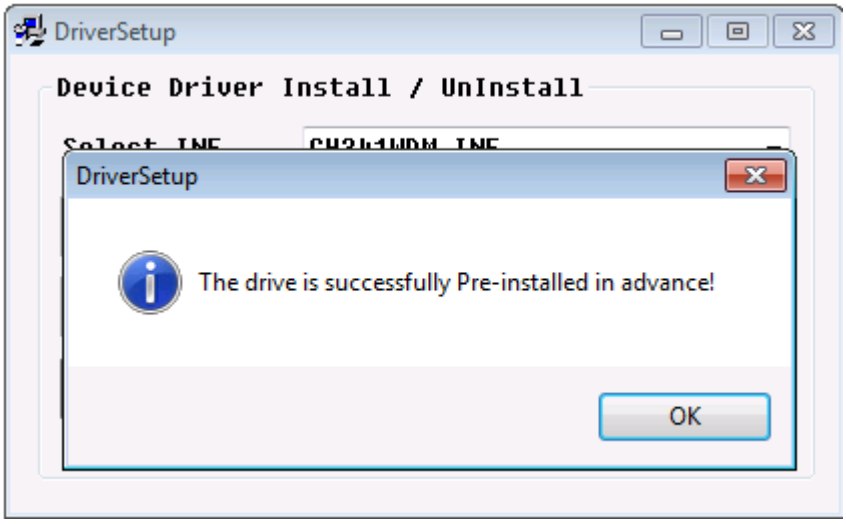
步骤 2 点击安装。

图1-2 驱动安装



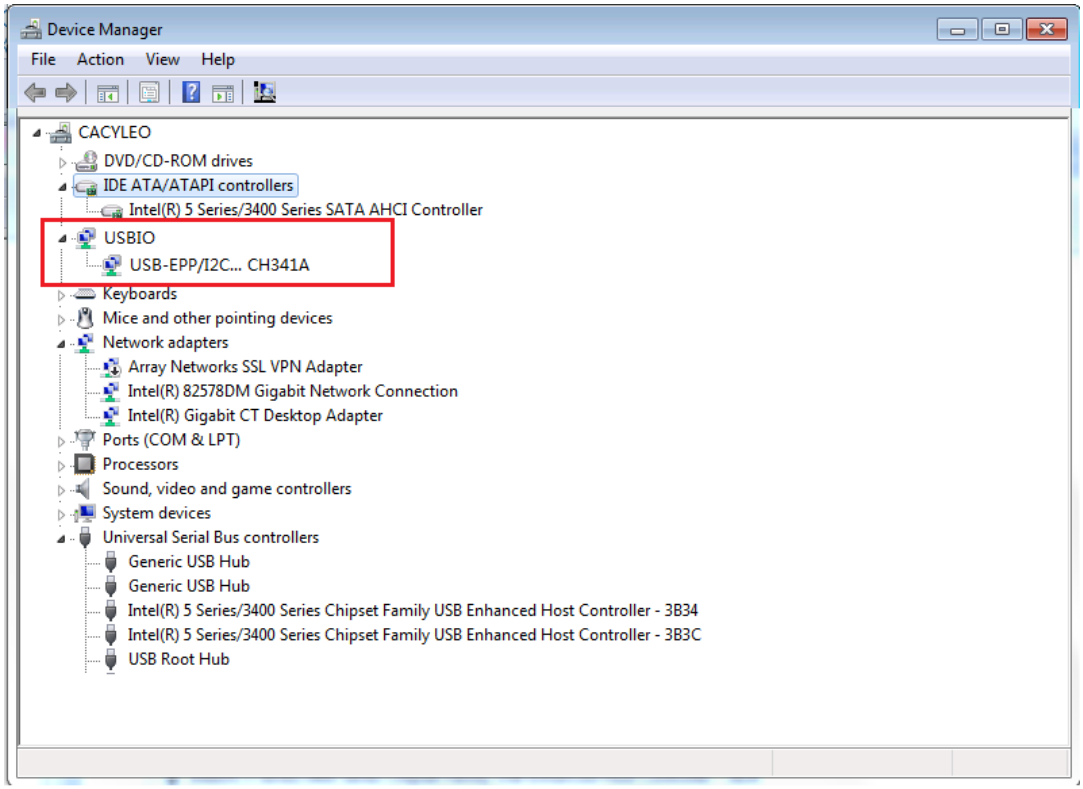
步骤 3 等待安装完成，完成后会提示安装成功，如图 1-3 所示。

图1-3 驱动安装成功提示



步骤 4 将 USB 转 I2C 设备连接到 PC 上，进入计算机的设备管理器中，驱动成功安装后会在“USBIO”有“USB-EPP/I2C…CH341A”显示，此时驱动便安装完成，可正常使用。

图1-4 查看设备管理器

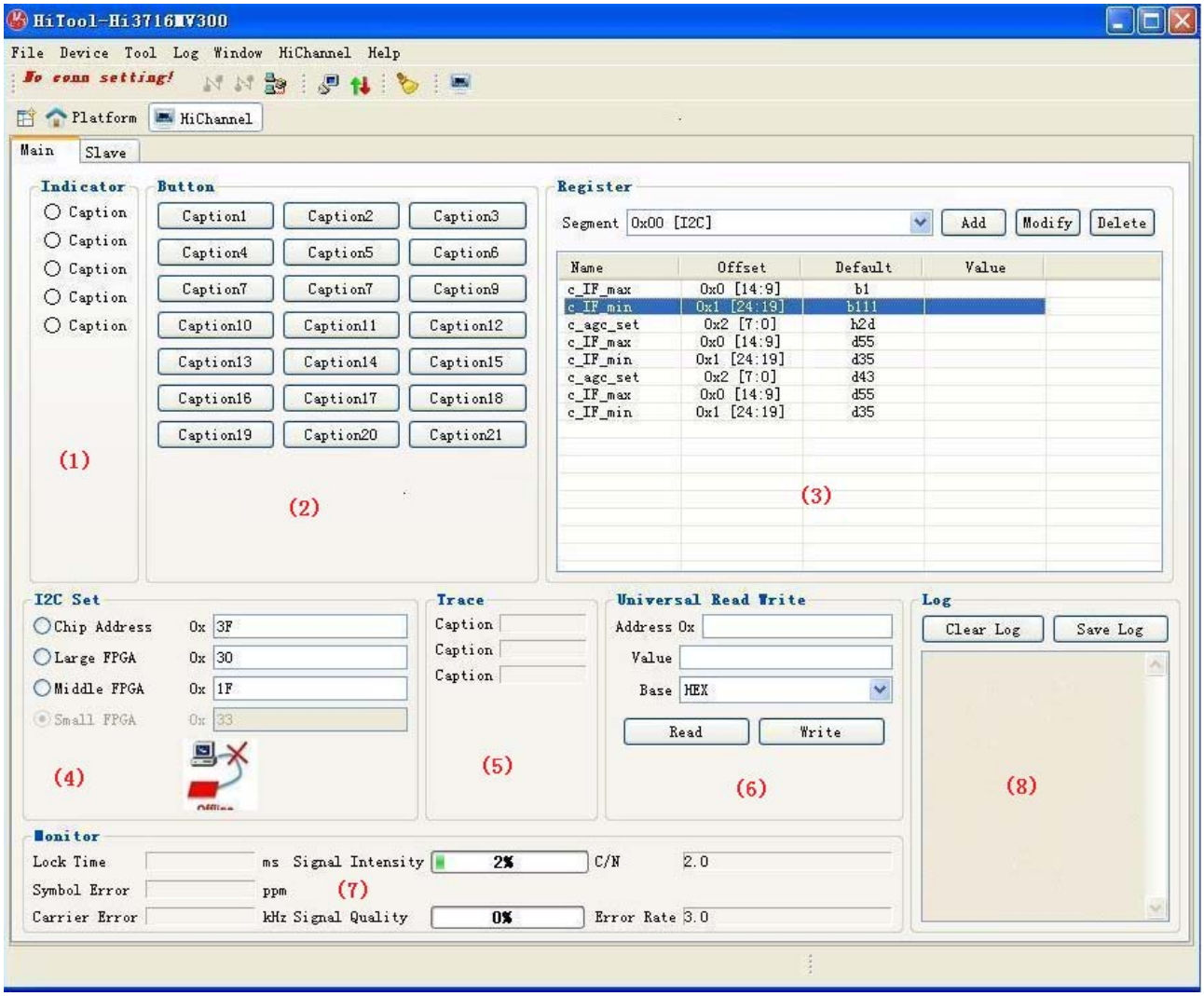


----结束

2 整体介绍

工具主界面如图 2-1 所示。

图2-1 HiChannel 工具界面



主界面包括两个标签页 Main 和 Slave，两个标签页布局完全一样。一般 Main 界面用于配置和调试芯片、FPGA 平台的大 FPGA 和中 FPGA，Slave 界面用于配置和调试 FPGA 平台的小 FPGA（小 FPGA 完成 FPGA 的一些附加功能）。

说明

FPGA 相关功能供海思内部开发人员使用，其他用户可无需关注。FPGA 平台的大、中、小 FPGA 有各自的器件地址，但大 FPGA 和中 FPGA 的寄存器地址统一编排（即无重复）。两个页面有独立的配置文件。

工具页面主要由如下区域组成：

- Indicator

  指示灯区，见图 2-1 中的位置（1）。工具自动周期性调用 C 脚本函数并获取函数返回值，根据该返回值动态调整指示灯的颜色。

- Button

  按钮区，见图 2-1 中的位置（2）。点击 Button 按钮会触发某 C 脚本函数被执行，被执行的 C 脚本函数与按钮的对应关系由专业版用户进行配置。

- Register

  寄存器区，见图 2-1 中的位置（3）。通过按钮和右键菜单，用户可以增加、修改、删除寄存器分区和寄存器，保存和导入寄存器文件。

- Trace

  跟踪显示区，见图 2-1 中的位置（5）。工具会自动周期性调用 C 脚本函数并获取函数返回的文本信息，显示在 Trace 区文本显示框内。

- Monitor

  常用监控区，见图 2-1 中的位置（7）。工具会自动周期性调用 C 脚本函数并获取函数返回的文本信息，分别显示在 Monitor 区的文本显示框和进度条内。Monitor 区可监控的内容包括：

  - C/N：载噪比。
  - Error Rate：错误率。
  - Signal Intensity：信号强度。
  - Signal Quality：信号质量。
  - Lock Time：锁定时间。
  - Symbol Error：符号率误差。
  - Carrier Error：载波频率误差。

- Universal Read Write

  通用寄存器读写区，见图 2-1 中的位置（6）。在该区域，用户能读写指定地址的 I2C 寄存器。

- I2C set

  I2C 设置区，见图 2-1 中的位置（4）。在该区域，用户可以修改 I2C 器件地址，并实时显示 I2C 总线状态。

- Log

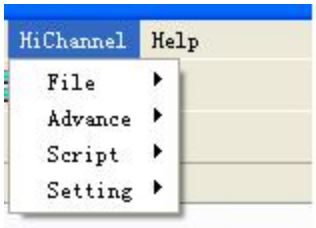
  日志管理区，见图 2-1 中的位置（8）。在该区域，用户可以观察到操作的相关信息，日志可以保存，清空。

3 菜单和工具项介绍

HiChannel 工具的菜单位于工具菜单栏中的 HiChannel 菜单项中。

专业版工具的菜单包括：[File]、[Advance]、[Script]、[Setting]菜单项。如图 3-1 所示。

图3-1 开发版 HiChannel 菜单



客户版工具的菜单包括：[Advance]、[Setting]菜单项。如图 3-2 所示。

图3-2 客户版 HiChannel 菜单

工具栏中如下两个图标为 HiChannel 工具特有的快捷按钮：

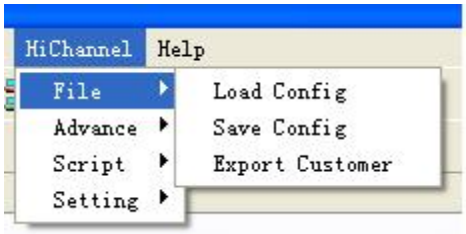
- 清除芯片信息  （仅专业版可见）
- 连接管理器 

3.1 菜单项

3.1.1 File

该菜单完成配置文件的操作，仅专业版可见。如图 3-3 所示。

图3-3 文件菜单



- Load Config

  加载配置文件，通常用于配置文件发生手动更新或者切换配置文件时。Main 和 Slave 页面分别有自己的配置文件（格式一致）。

  - 如在 Main 页面操作加载配置时，配置文件所有部分均起作用。
  - 如在 Slave 页面操作加载配置时，[Global]、[Advance]和[Setting]段不起作用，其它起作用。

说明

强烈建议 Main 和 Slave 的配置文件名分别加后缀“_m”、“_s”，方便识别。

- Save Config

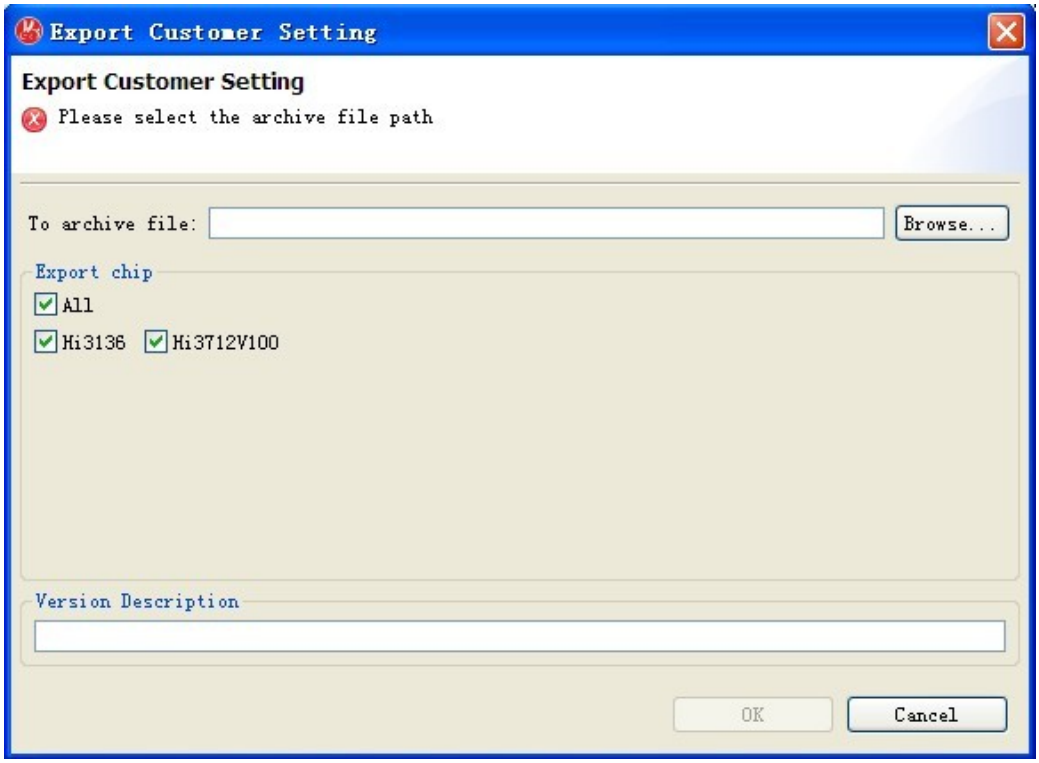
  保存配置文件，根据当前设置，生成配置文件，该配置文件只包括当前页面的配置信息。

- Export Customer

  导出客户版的 HiChannel 工具包，使用 HiPublisher 发布工具即可将该工具包制作成客户使用的客户版 HiChannel 工具。支持多个芯片的配置文件导出到同一个发工具包中。。

  点击导出客户版菜单后，选择保存的位置，以及选择导出的芯片，添加描述信息（可选），点击确定按钮。如图 3-4 所示。

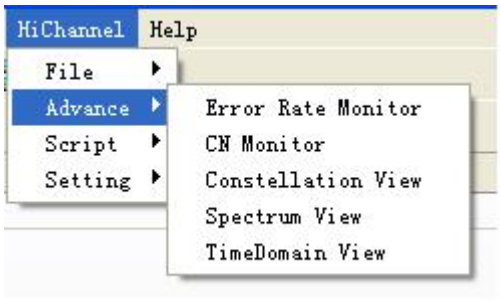
图3-4 配置导出信息



- Advance

该菜单项主要完成一些高级显示功能，支持通过图形的方式观察错误率、载噪比、星座图、频谱图、时域图等。如图 3-5 所示，具体说明及使用参见章节“高级功能菜单”。

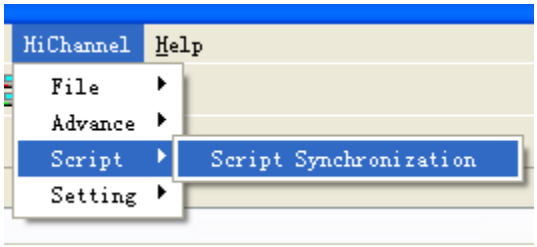
图3-5 高级菜单



3.1.2 Script

该菜单项主要是对配置脚本进行同步，仅专业版可见。如图 3-6 所示。

图3-6 脚本同步菜单

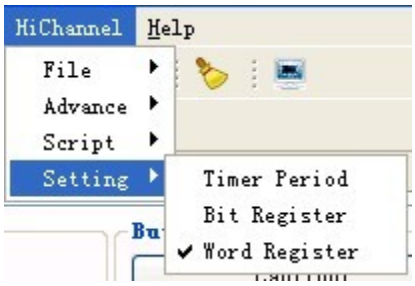


Script Synchronization：脚本同步，当脚本文件在外部被修改后，点击该菜单项，可以使工具重新加载脚本文件。

3.1.3 Setting

设置菜单主要完成对工具的一些配置。如图 3-7 所示。

图3-7 设置菜单



- Timer Period

  设置定时器周期，即设置指示灯、跟踪显示、通用监控等自动刷新的周期，输入数值，以 ms 为单位。如图 3-8 所示。

  图3-8 设置定时器周期

- Bit Register

位寄存器，寄存器信息区的寄存器以比特为单位进行定义、显示和操作。当本条被选中时，前面出现“√”；“Bit Register”与“Word Register”两选一。

- Word Register

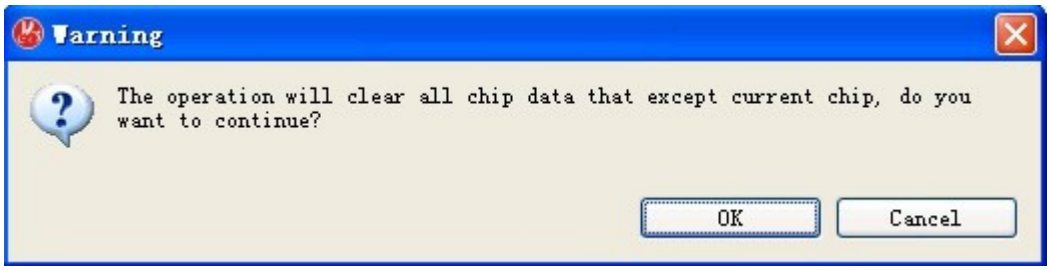
  字寄存器，寄存器信息区的寄存器以字（寄存器位宽如果为 8，则字即字节）为单位进行定义、显示和操作。当本条被选中时，前面出现“√”；Word Register 与“Bit Register”两选一。此种方式比较适合于生成给驱动软件的寄存器列表文件。

3.2 工具项

3.2.1 清除芯片信息

把除当前芯片外曾切换过的所有历史芯片的配置信息删除，该功能仅专业版可见。点击  后会出现提示对话框，点击“OK”则会进行清除操作。如图 3-9 所示。

图3-9 清除芯片信息



3.2.2 连接管理器

对 Dut NetWork 和 USB 连接进行统一设置。点击工具栏上的  打开连接管理器对话框，打开后用户可以在“DutNetwork”和“USB Device”两种连接方式中任选其一，然后对其进行相应设置。

- 设置 DUT 网络，如图 3-10 所示。

图3-10 Dut 网络设置

IP Address：目标单板的 IP 地址。

I2C Port：目标单板上待调试的 I2C 器件的端口号。

Dev Address：目标单板上待调试的 I2C 器件的设备地址。

I2C Speed：目标单板上待调试的 I2C 器件的速率值，以 kHz 为单位。

- USB 设置，可以设置 USB 速率和设备地址. 如图 3-11 所示。

图3-11 USB 设置

  - I2C Speed：目标单板上待调试的 I2C 器件的速率值，以 kHz 为单位。
  - Dev Address：目标单板上待调试的 I2C 器件的设备地址。

设置完成后，点击“OK”按钮，工具会根据连接的配置信息与目标单板进行交互，如连接失败，会有相应错误提示，连接成功后工具开始定期刷新信号强度等芯片状态。

注意

对于连接 Dut 网络时，工具会先发送启动板端程序命令，启动板端程序的命令可以在首选项中对其进行配置，命令默认为“soc_server”，如图 3-12 所示。

图3-12 启动板端命令设置

4 主要操作

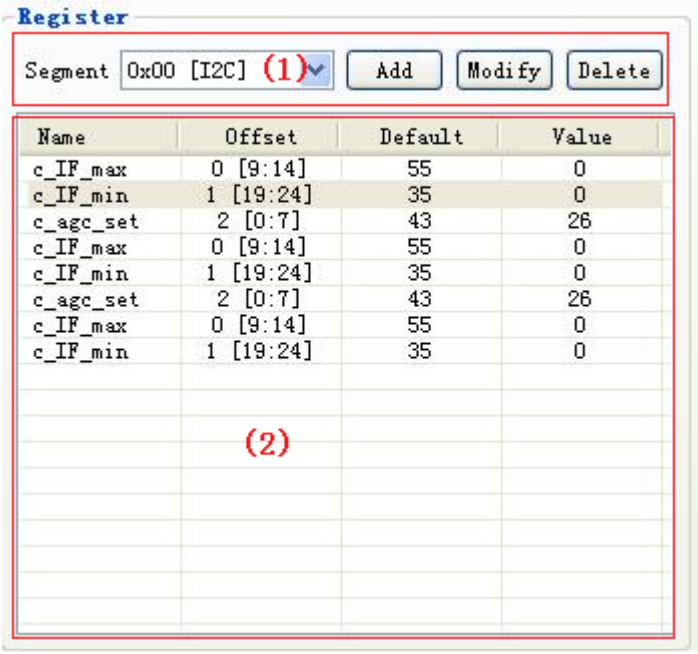
HiChannel 的主要操作包括：

- 寄存器操作（Register）
- 按钮操作（Button）
- 指示灯操作（Indicator）
- 跟踪显示操作（Trace）
- 常用监控操作（Monitor）
- 通用寄存器读写操作（Universal Read Write）
- I2C 设置操作（I2C set）
- 日志管理操作（Log）

4.1 寄存器操作（Register）

寄存器操作区如图 4-1 所示，分为分段信息区见图 4-1 中（1）的位置，和寄存器信息区见图 4-1 中（2）的位置。通过按钮和右键菜单可以实现增加、修改、删除寄存器分段和寄存器，保存和导入寄存器文件。寄存器操作区中寄存器显示的形式由[Setting]菜单中的[Bit Register]和[Word Register]决定，图 4-1 所示为[Bit Regisger]模式。

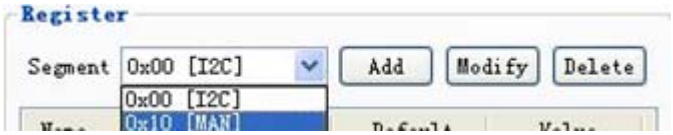
图4-1 寄存器操作区



4.1.1 分段信息区

点击分段信息区中的下拉框，框中列出已有分段名（及对应基地址），如图 4-2 所示，选择某一分段后，该分段的寄存器将显示在寄存器信息区中。下拉框中最后 1 个为“All”，表示显示所有的寄存器。

图4-2 分段选择



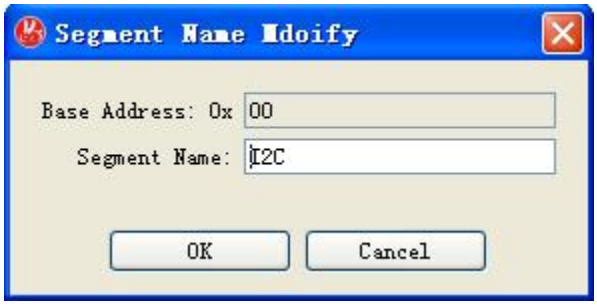
分段信息区的“Add”、“Modify”、“Delete”按钮分别实现分段的添加、修改和删除。“All”分段名一直存在（即不必添加）且不允许被删除，它对应的寄存器包括所有分段的寄存器。

- Add—— 左键点击“添加”按钮，出现如图 4-3 所示，输入基地址和分段名称，即可完成分段添加。

图4-3 添加分段

- Modify —— 左键点击“修改”按钮，出现如图 4-4 所示，可修改当前基地址和分段名。

图4-4 修改选择



- Delete—— 左键点击“删除”按钮，将删除当前分段名及所包含的寄存器，为防止误删，会有确认提示，如图 4-5 所示。

图4-5 确认删除

4.1.2 寄存器信息区

寄存器信息区如图 4-6 所示，包括：

- Name —— 寄存器名字。
- Offset —— 基于分段地址的寄存器偏移地址。
- Default —— 缺省值。

- Value —— 当前值。

缺省值和当前值可以按指定基显示模式，包括：十进制、十六进制、二进制，分别以前缀 d、h、b 表示。

当分段选择 All 时，列表中将会列出所有分段中的寄存器，Offset 变为绝对地址。

图4-6 寄存器信息区

Name	Offset	Default	Value
c_IF_max	0 [9:14]	55	0
c_IF_min	1 [19:24]	35	0
c_agc_set	2 [0:7]	43	26
c_IF_max	0 [9:14]	55	0
c_IF_min	1 [19:24]	35	0
c_agc_set	2 [0:7]	43	26
c_IF_max	0 [9:14]	55	0
c_IF_min	1 [19:24]	35	0

- 选中 —— 左键单击某寄存器行，此行变蓝色，表示选中。
- 读寄存器 —— 双击某寄存器行，即可以读该寄存器，显示在当前值域。
- 写寄存器 —— 左键点击某寄存器的 Value 格，可以直接输入新值，如图 4-7 蓝色部分所示，在该当前值格外任意地方点击即完成输入。

图4-7 编辑寄存器值

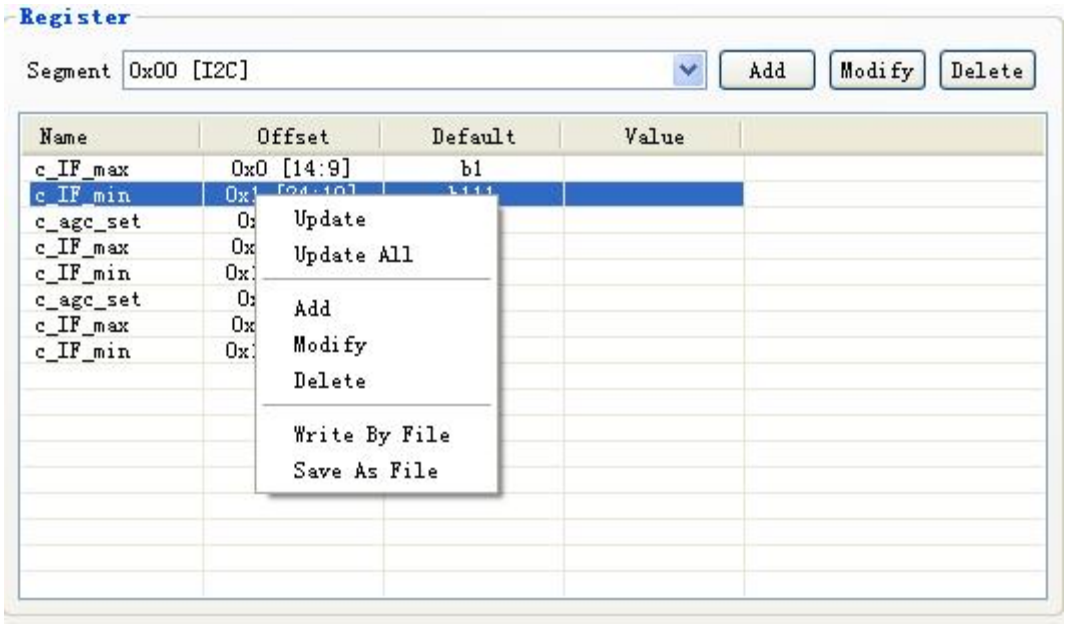
Name	Offset	Default	Value
c_IF_max	0 [9:14]	55	0
c_IF_min	1 [19:24]	35	0
c_agc_set	2 [0:7]	43	26
c_IF_max	0 [9:14]	55	0
c_IF_min	1 [19:24]	35	0
c_agc_set	2 [0:7]	43	26
c_IF_max	0 [9:14]	55	0
c_IF_min	1 [19:24]	35	0

说明
本工具中所说的读寄存器是指从目标单板中读取寄存器内容，显示到工具中。

4.1.3 寄存器信息区右键菜单功能

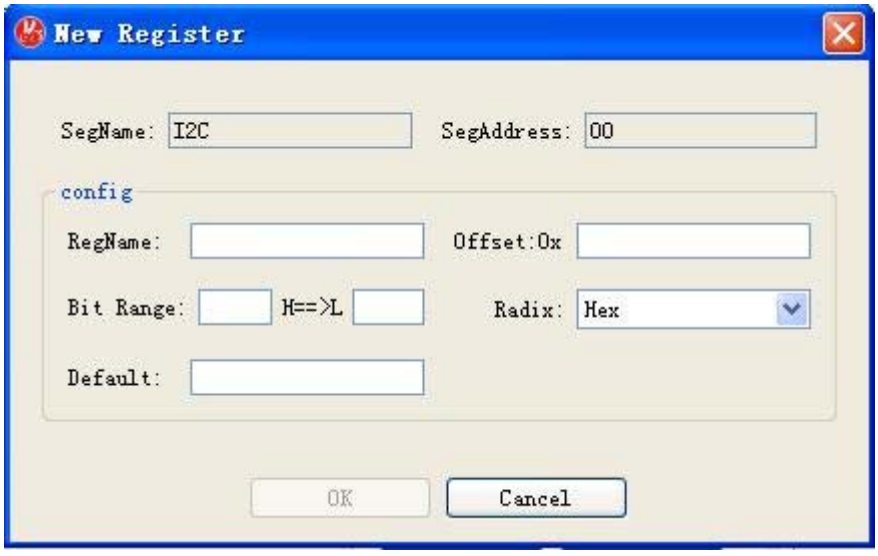
在寄存器信息区选中任意一个寄存器时，选中条变蓝，单击右键，跳出菜单，如图 4-8 所示。当在分段中选择 All 时，右键菜单下的添加、修改、删除变得无效（灰色）。

图4-8 右键菜单



- Update —— 读取选中寄存器的值。
- Update All —— 读取寄存器信息区中列出的所有寄存器的值。
- Add —— 增加寄存器，可输入 RegName、Offset（16 进制）、Bit Range（十进制、左高右低）、Base（十六进制）、Default 缺省值，如图 4-9 所示。

图4-9 添加寄存器



- Modify —— 修改寄存器，界面同图 4-9 只是框中已有原来内容。
- Delete —— 删除所选中的寄存器，无告警。

- Save As File ——将当前寄存器信息区的内容写到文件中，文件可由用户指定。
- Write by File ——从文件中导入寄存器数据到信息区中。

4.2 按钮操作（Button）

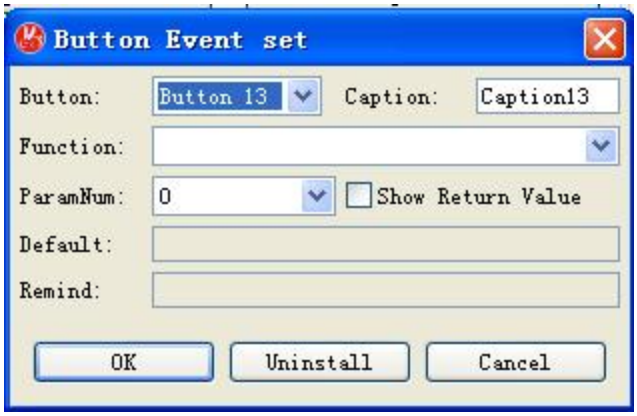
按钮区如图 4-10 所示。

图4-10 Button 区

- 对客户版用户，Button 区的每个按钮的功能已固定，详细说明请参考章节 8.2 按钮操作栏 Button。
- 对专业版用户，Button 区的任一按钮都可由用户配置其对应的 C 脚本函数，用鼠标左键点击即可执行该 C 脚本函数定义的功能。

在某按钮上点击鼠标右键时弹出界面，可以配置按钮的功能，如图 4-11 所示。

图4-11 按钮配置



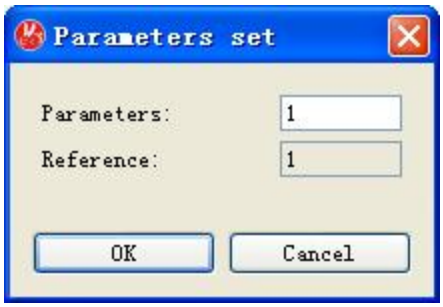
- Button —— 显示按钮的编号，可以选择不同按钮，实现对不同按钮的配置。
- Caption —— 输入需要在按钮上显示的字符串，用于提示该按钮的功能。
- Function —— 在下拉框中选择对应的 C 脚本函数。
- ParamNum —— 选择输入参数的个数，默认为 0。如果选 0，则后面“默认值”和“提示”为灰色，不能操作。

说明
工具不会对参数做任何解析，它只会以字符串格式传给脚本函数。

- Default —— 用于输入参数的缺省值，多个数值间用英文逗号隔开。注：必须使用英文逗号。
- Remind —— 用于输入参数提示。
- Show Return Value —— 如选中，则显示该 C 脚本函数的返回值。
- Uninstall —— 用于卸载指定按钮。

左键点击某已配置的按钮，如果它的参数个数不为 0 时，则应在跳出的窗口中输入参数，多个参数以逗号隔开。如图 4-12 所示，在参考中显示的即为配置该按钮时在提示框中输入的字符串。当参数有缺省值时，缺省值会显示在输入框中。

图4-12 参数设置



4.3 指示灯操作（Indicator）

指示灯区如图 4-13 所示。

图4-13 指示灯区域

- 对客户版用户，每个指示灯的功能已固定，详细说明请参考章节 8.1 。
- 对专业版用户，每个指示灯都可以由用户配置其对应的 C 脚本函数。系统定时周期性调用该 C 脚本函数，并根据函数返回值让指示灯显示不同的颜色。

在某指示灯右侧字符上点击鼠标右键可以配置指示灯的功能，如图 4-14 所示。

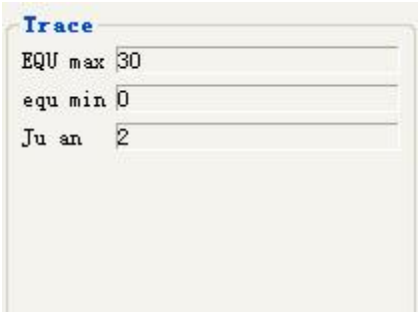
图4-14 指示灯设置

- Lamp—— 显示指示灯的编号，可以选择不同指示灯，实现对不同指示灯的配置。
- Caption —— 输入需要在指示灯右边显示的字符串，即给出该指示灯的意义。
- Lamp Mode —— 选择灯的显示模式，可选 Gray+Green（按序对应数值 0/>=1）或 Gray+Green+Red（按序对应数值 0/1/>=2）。
- 其它参见“按钮操作”节的对应项。

4.4 跟踪显示（Trace）

Trace 区如图 4-15 所示。

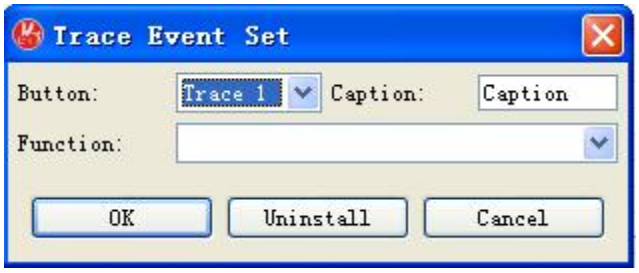
图4-15 跟踪显示区域



- 对客户版用户，每个跟踪项的功能已固定，详细说明请参考章节 8.3 。
- 对专业版用户，每个跟踪显示都可以由用户配置其对应的 C 脚本函数。系统定时周期性调用该 C 脚本函数，并在跟踪显示框内显示返回的字符串。

在某跟踪显示框上（不能在文字上）点击鼠标右键可以配置跟踪显示框的功能，如图 4-16 所示。

图4-16 跟踪显示设置



- Trace —— 显示框的编号，可以选择不同显示框，实现对不同显示框的配置。
- Caption —— 输入需要在左边横线上显示的字符串，即给出该跟踪显示的意义。
- 其它参见 4.2 的对应项。

4.5 常用监控（Monitor）

监控区如图 4-17 所示。

图4-17 常用监控区域

- Lock Time ——系统锁定时间。
- Symbol Error ——符号率误差。
- Carrier Error ——载波频率误差。
- Signal Intensity ——信号强度（0~100）。
- Signal Quality ——信号质量（0~100）。
- C/N ——计算 C/N 值，即载噪比。
- Error Rate ——错误率。

对客户版用户，每个监控项的详细说明请参考章节 8.4 。

对专业版用户，每个监控显示都可以由用户配置其对应的 C 脚本函数。系统定时周期性调用该 C 脚本函数，并在对应的框内显示返回的字符串或显示进度条。

在各条监控的显示框内点击右键，从函数列表中选择 C 脚本函数

图4-18 选择函数

- Function—— 在下拉框中选择对应的 C 脚本函数。如果绑定的函数在配置文件指定的 C 函数脚本中不存在，则选中的项为空。
- Uninstall——把绑定函数置空。

4.6 通用寄存器读写（Universal Read/Write）

通用寄存器读写（Universal Read Write）区如图 4-19 所示。Radix 中 Dec、Hex 和 Bin 分别表示十、十六、二进制。

图4-19 通用寄存器读写区域

- Read —— 在 Address 输入 I2C 器件的寄存器地址（十六进制），按“Read”按钮可从目标单板读回该寄存器的内容，并按“Radix”中选定的进制显示在“Value”后的方框中。
- Write —— 输入寄存器地址，并在“Value”后方框中输入准备写入的值（按“Radix”中选定的进制），按“Write”按钮完成写寄存器。

注意

- Address 和配置文件中 AddressBytes 设置相关，Address 的长度不能大于 AddressBytes。
- Value 输入值长度与配置文件的 RegisterBytes 设置，还有 Base 进制选择相关，当 Base 等于 HEX，Value 的长度不能大于 2*RegisterBytes。当 Base 等于 BIN，Value 的长度不能大于 8*RegisterBytes。

4.7 I2C 设置（I2C Set）

I2C 设置区如图 4-20 所示，主要用于 I2C 器件地址设置和 I2C 连接状态监控。

图4-20 I2C 设置区域

四个单选按钮用于选择 I2C 从器件及该器件地址（16 进制）。Chip Address 通常用于和 ASIC 芯片的通信，Large FPGA 和 Middle FPGA 通常用于 FPGA 版本的调试和测试，Small FPGA 用于 FPGA 平台辅助功能的控制。在 Main 页面只能选择前三个之一，在 Slave 页面固定使用 Small FPGA。

说明

FPGA 相关功能供海思内部开发人员使用，其他用户可无需关注。FPGA 平台的大、中、小 FPGA 有各自的器件地址，但大 FPGA 和中 FPGA 的寄存器地址统一编排（即无重复）。两个页面有独立的配置文件。

图标显示 I2C 通讯是否正常，正常时显示 Online（绿色），异常显示 Offline（红色）。对专业版用户，可以设置 C 脚本函数，工具会周期调用绑定的函数，并根据函数执行的结果判断是否正常（值大于等于 1 为正常）。在状态图标上点击右键，选择 C 脚本函数。如图 4-21 所示.。

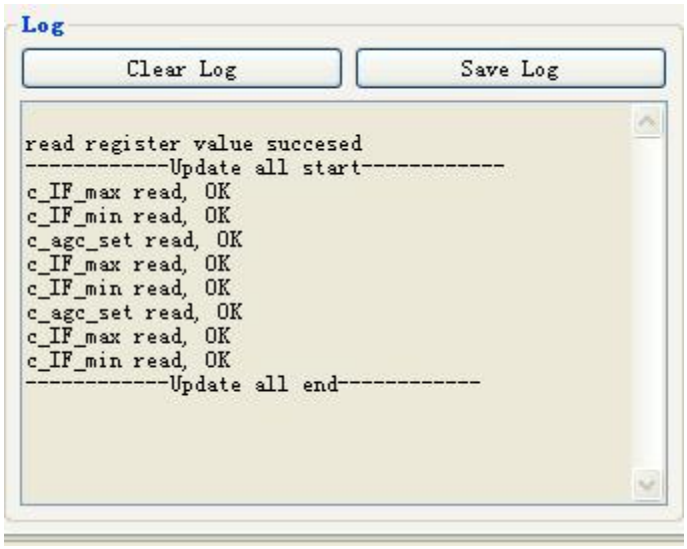
图4-21 I2C 设置对话框

- Function—— 在下拉框中选择对应的 C 脚本函数，如果绑定的函数在配置文件指定的 C 函数脚本中不存在，则选中的项为空。
- Uninstall——把绑定函数置空。

4.8 日志管理（Log）

Log 区如图 4-22 所示，在 Log 显示区中显示工具各种操作执行的信息。包括 Register 区和 Universal Read/Write 区读写寄存器正常和异常信息，Button 区各按钮执行的操作的正常和异常信息，Indicator 区、Monitor 区、Trace 区的操作的异常信息等。

图4-22 日志管理区域



- Clear Log —— 清除显示区中 Log 信息
- Save Log —— 将显示区中的 Log 信息保存到用户指定的文件中，txt 格式

5 配置文件

说明
本章主要介绍配置文件的格式，仅专业版的用户需要了解。

配置文件由以下几个部分组成：

- [Global]
- [Button]
- [Indicator]
- [Trace]
- [Monitor]
- [I2C Set]
- [Advance]
- [Segment]
- [Bit Register]
- [Word Register]

Main 和 Slave 两个标签页，对应独立的配置文件，需要分别设置，但两者的配置文件格式一致。

注意
如果以“；”开头，表示该行被注释。

5.1 全局段 Global

说明
下面描述中“=”后面的字符串仅为本例子中的值，需要由实际内容替代。“//”后面为对应的说明，实际中并不需要。

Global 分段主要保存全局配置信息：

```
[Global]  //段识别符
Chip=12  //Chip名，“=”号后的字符将显示到窗口顶部
Version=0.1  //版本号，“=”号后的字符将显示到窗口顶部，Chip名后
AddrBytes=1  //芯片地址的字节数，8位地址线时设为1，32位地址线时设为4。注：AddrBytes的范围是1～4
OffsetBits=4  //偏移地址位数。注：OffsetBits的范围是1～4
RegisterBytes=1  //寄存器的字节数，8位宽时设为1，32位宽设为4。注：RegisterBytes的范围是1～4
CSRun=E:\HiChannel\demo-script|app1#E:\HiChannel\demo-script2|app2#  //C脚本文件路径列表，C脚本不同路径之间必须用“#”隔开，文件路径后面是联合编译后定义的执行文件的名字，必须唯一，无脚本文件路径时也须添加“|”标识。
CSIncludes=E:\HiChannel\header1|E:\HiChannel\header2|#E:\HiChannel\header2|#  //C脚本依赖的头文件路径，多个路径之间用“#”隔开，对C脚本同一路径需要引入多个头文件时，不同头文件路径之间用“|”隔开，无头文件时也须添加“|”标识。
```

5.2 按钮操作段 Button

[Button]段对应主界面中的 Button 区，保存了界面上各个按钮的配置信息：

```
[Button]  //段识别符
Button 1=FunctionName|Caption|Paramnum|param1,param2|Remind|isReturnValue  //第1个按钮的配置
……
```

Button 配置说明：

- FunctionName：配置时，界面中“Function”下拉框中指定的 C 脚本函数名格式为“app 名称#函数名”。
- Caption：Button 配置时界面中的“Caption”输入框中的内容。
- Paramnum：配置时，界面中“Paramnum”下拉框中选择的参数个数，如不选择，则默认为 0。
- Paramnum 的后面是参数，参数由界面中“Default”输入框中的内容，多个参数之间用英文逗号分开。
- Remind：配置时界面中“Remind”输入框中的内容，用于输入参数提示。
- isReturnValue：配置时界面中“Show Return Value”复选框。
  - 复选框勾选则为 true
  - 不勾选则为 false。

5.3 指示灯段 Indicator

[Indicator]段对应界面中的 Indicator 区，保存了界面上各个指示灯的配置信息：

```
[Indicator]  //段识别符
Indicator 1=FunctionName|Caption|Type  //第1个指示灯的配置
Indicator 2=FunctionName|Caption|Type  //第2个指示灯的配置
Indicator 3=FunctionName|Caption|Type  //第3个指示灯的配置
……
```

Indicator 的配置说明如下：

- FunctionName：对应界面中“Function”下拉框中指定的 C 脚本函数名，格式为“app 名称#函数名”。
- Caption：对应界面中的“Caption”输入框中的内容。Type：对应界面中“Lamp Model”下拉框中选中的值。

5.4 跟踪显示段 Trace

[Trace]段对应界面中的 Trace 区，保存了界面上各个跟踪项的配置信息：

```
[Trace]  //段识别符
Trace 1=FunctionName|Caption  //第1个跟踪显示的配置
Trace 2=FunctionName|Caption  //第2个跟踪显示的配置
Trace 3=FunctionName|Caption  //第3个跟踪显示的配置
……
```

Trace 配置说明：

- FunctionName：对应界面中“Function”下拉框中指定的 C 脚本函数名，格式为“app 名称#函数名”。
- Caption：对应界面中的“Caption”输入框中的内容。

5.5 常用监控段 Monitor

[Monitor]段对应界面中的 Monitor 区，保存了界面上各监控项的配置信息：

```
[Monitor]  //段识别符
Lock Time=FunctionName  //锁定时间的配置
Symbol Error=FunctionName  //符号率误差的配置
Carrier Error=FunctionName  //载波频率误差的配置
Signal Intensity=FunctionName  //信号强度的配置
Signal Quality=FunctionName  //信号质量的配置
```

```
C/N=FunctionName  //C/N的配置
Error Rate=FunctionName  //错误率的配置
```

Monitor 配置说明：

FunctionName：对应界面中“Function”下拉框中指定的 C 脚本函数名，格式为“app 名称#函数名”。

5.6 I2C 设置段 I2C Set

[I2C Set]段对应界面中的 I2CSet 区，保存了界面上各 I2C 器件的配置信息：

```
[I2C Set]  //段识别符
Chip Address=0x3F  //芯片的I2C通讯的器件地址
Large FPGA=0x30  //大FPGA的I2C通讯的器件地址
Middle FPGA=0x1F  //中FPGA的I2C通讯的器件地址
Small FPGA=0x33  //小FPGA的I2C通讯的器件地址
I2C Communication=FunctionName  //通讯检测脚本函数，格式为：“app名称#函数名”
Selection Item=Small FPGA  //选中的项目
```

I2C Set 配置说明：

I2C 设置中的器件地址[Chip Address]，[Large FPGA]，[Middle FPGA]，[Small FPGA]的值以十六进制保存。

5.7 高级段 Advance

[Advance]段保存了工具的一些高级配置项：

```
[Advance]  //段识别符
Spectrum Function=FunctionName  //频谱观察中选择的C脚本函数，格式为：“app名称#函数名”
Timer Period=300  //定时周期，单位ms
I2C Speed=400  //I2C速率，单位kHz
Auto Update=Yes  //保留字段
Stop Auto Update=No  //保留字段
Bit Register=Yes  //比特寄存器形式，Yes表示配置项加载后，界面上对应项为勾选状态，No则相反，界面上对应项为未勾选状态，与Word Register只有一个能Yes
Word Register=No  //字寄存器形式，Yes表示配置项加载后，界面上对应项为勾选状态，No则相反，界面上对应项为未勾选状态，与Bit Register只有一个能Yes
```

5.8 分段信息段 Segment

[Segment]段保存了界面上 Register 区中的分段配置信息：

```
[Segment]  //识别符
Number=16 //寄存器的分段数，必须与后面所列分段基地址数一致
//以下是分段基地址，格式：分段基地址（16进制）=分段名，如下：
00=I2C
10=MAN
20=AGC
……
F0=RSD
```

Segment 配置说明：

列出的分段基地址数必须和 Number 定义的数量一致。

5.9 比特寄存器信息段 Bit Register

[Bit Register]段保存比特寄存器的信息，该类型寄存器的特点是寄存器的起始比特和结束比特可是是 0~31 中的任意数字。寄存器按所属段依次排列，首先列出分段的基地址（该基地址应在[Segment]有定义），然后依次列出属于该分段的寄存器。一个分段列完后空一行，再列下一个分段，直到列完所有分段。

```
[Bit Register]    //识别符
[00]    //[16进制基地址]，基地址应[Segment]中有定义，否则无效
Number=8    //该分段包含的寄存器个数（十进制）
00[14: 9]=6'd55      |c_IF_max    //[偏移位域地址]=[位数]`[进制][默认值] |[寄存器名]
01[24:19]=6'd35      |c_IF_min
02[ 7: 0]=8'd43      |c_agc_set
……
[10]    //[16进制基地址]，基地址应[Segment]中有定义，否则无效。
Number=4    //该分段包含的寄存器个数（十进制）
00[14: 9]=6'd55      |c_cbs_test    //[偏移位域地址]=[位数]`[进制][默认值] |[寄存器名]
01[24:19]=6'd35      |c_cbs_xxx
```

5.10 字寄存器信息段 Word Register

[Word Register]保存字寄存器的配置信息，字寄存器的特点为寄存器的起始比特和结束比特固定为[8*RegisterBytes-1:0]，位数固定为 8*RegisterBytes，RegisterBytes 在[Global]

段中有说明。寄存器按所属段依次排列，首先列出分段的基地址（该基地址应在[Segment]有定义），然后依次列出属于该分段的寄存器。一个分段列完后空一行，再列下一个分段，直到列完所有分段。

```
[Word Register]    //识别符
[30]    //寄存器所属分段的基地址，应[Segment]中有过定义，否则无效。
Number=8    //该分段包含的寄存器个数（十进制）
00[7:0]=8'd55      |c_IF_max  //[偏移位域地址]=[位数]`[进制][默认值]    |[寄存器名]
```

【偏移位域地址】

相对特定基址的偏移位域地址。格式为：OffsetAddr[HighBit, LowBit]，OffsetAddr 为 16 进制，HighBit、LowBit 为 10 进制。例：当基址为 0x20，偏移位域地址为 0xF[5, 2]，实际表示 0x2F 地址的第 5 至 2 位。

6 C 脚本

本章介绍专业版用户在编写界面控件绑定的 C 脚本时，如何使用工具提供的动态库接口，该接口与单板操作相关。

6.1 动态库接口说明

Socket 连接

- int InitSock(char* SevIP，int SevPort)
  说明：建立 socket 连接
  参数说明：
  - char* SevIP---server 端 ip 地址。
  - int SevPort---server 端端口号。
  返回值：0：成功；非 0：错误码
- int DeinitSock()
  说明：卸载 socket 连接。
  返回值：0：成功；
- int openi2c()
  说明：打开 i2c。
  返回值：0：成功；非 0：错误码
- int closei2c()
  说明：关闭 i2c。
  返回值：0：成功；非 0：错误码
- seti2cspeed(int i2cPort, int i2cSpeed)；
  说明：设置 i2c 端口和速率。
  参数说明：
  - int i2cPort--- i2c 端口。
  - int i2cSpeed---i2c 速率。

返回值：0：成功；非 0：错误码。

- int ReadregMulti(int i2cport，char idevaddr，int iregaddr，int iregaddrcount，char *buffer，int len)；

  说明：读取寄存器

  参数说明：

  - int i2cport --- i2c 端口号。
  - char idevaddr---设备地址。
  - int iregaddr--- 寄存器地址。
  - int iregaddrcount---寄存器地址占用字节数。
  - char *buffer--- 要读的值存放的起始地址。
  - int len--- 读取寄存器个数。

  返回值：0：成功；非 0：错误码

- WriteregMulti(int i2cport，char idevaddr，int iregaddr，int iregaddrcount，char *data，int len)；

  说明：写寄存器。

  参数说明：

  - int i2cport --- i2c 端口号。
  - char idevaddr---设备地址。
  - int iregaddr---寄存器地址。
  - int iregaddrcount---寄存器地址占用字节数。
  - char *data---要写的值存放的起始地址。
  - int len---写寄存器个数。

    返回值：0：成功；非 0：错误码

- int ReadregPeriod(int i2cport，char idevaddr，int iregaddr，int iregaddrcount，char *buffer，int iregbytes，int count，int period)

  说明：周期读某一寄存器

  参数说明：

  - int i2cport --- i2c 端口号。
  - char idevaddr---设备地址。
  - int iregaddr---寄存器地址。
  - int iregaddrcount---寄存器地址占用字节数。
  - char *buffer---要读的值存放的起始地址。
  - int iregbytes ---寄存器占字节数。
  - int count ---读取次数。
  - int period ---读取周期，单位 μs。

  返回值：0：成功；非 0：错误码

注意

1、DeinitSock()和 InitSock()必须成对出现，否则可能会影响别的操作。

2、在调用 ReadregPeriod 函数时，要求 count*period<4000000，这是由于动态库中 socket 读超时设置为 4s，超过这个数 dll 会返回-7。

USB 连接

- USBIO_OpenDevice(int deviceId)

  说明：打开 USB 设备。

  参数说明：

  int deviceId ---设备序列号，0 对应第一个设备。

  返回值：返回句柄，出错则无效。

- USBIO_CloseDevice(int deviceId)

  说明：关闭 USB 设备。

  参数说明：

  int deviceId ---设备序列号，0 对应第一个设备。

- USBIO_SetStream(int deviceId, int mode)

  说明：设置串口流模式。

  参数说明：

  - int deviceId---设备序列号，0 对应第一个设备。
  - int mode---指定模式，见下行：
    - 位 1-位 0：I2C 接口速度/SCL 频率，00=低速/20kHz，01=标准/100kHz(默认值)，10=快速/400kHz，11=高速/750kHz。
    - 位 2：SPI 的 I/O 数/IO 引脚， 0=单入单出（D3 时钟/D5 出/D7 入）(默认值)，1=双入双出（D3 时钟/D5 出 D4 出/D7 入 D6 入）
      位 7：SPI 字节中的位顺序，0=低位在前, 1=高位在前。
      其它保留，必须为 0。

- USBIO_StreamI2C(int deviceId, int writeLength, char * writeBuffer, int readLength, char * readBuffer)

  说明：处理 I2C 数据流，参数说明：

  - int deviceId ---设备序列号，0 对应第一个设备。
  - int writeLength ---准备写出的数据字节数。
  - char * writeBuffer ---指向一个缓冲区，放置准备写出的数据，首字节通常是 I2C 设备地址及读写方向位。
  - int readLength --- 准备读取的数据字节数。
  - char * readBuffer --- 指向一个缓冲区，返回读入的数据。

6.1.2 控件绑定函数返回值限制

编写控件绑定 C 脚本函数时，绑定函数的打印格式要求如下：

```
if(全部流程没有异常返回
printf("error=%s,return=%f","ok",0.0);
elseif(有错误1)
printf("error=%s,return=%d","err 1, failed", 1);
elseif(错误2)
printf("error=%s,return=%d","err 2, failed", 2);
```

说明：

- error：用来说明执行状态信息：当状态信息中包含“err”或“failed”，任务函数执行失败，否则认为函数执行成功。
- return 用来描述真实的返回值：只有函数的返回状态为成功，工具端才会解析真实的返回值。

编写控件绑定 C 脚本函数时，也应特别注意每个控件的函数返回值要求：

- Button 操作：无限制。
- Indicator 操作：返回值必须为 0 或者正整数。
- Trace 操作：无限制。
- I2c Set 操作：返回值必须为整数。
- Monitor 操作：
  - Lock Time：返回值必须为 0 或者正整数。
  - Symbol Error：返回值必须为浮点数或者整数。注：浮点型数据将会精确到小数点后五位，显示四舍五入后的值。
  - Carrier Error：返回值必须为浮点数或者整数。注：浮点型数据将会精确到小数点后五位，显示四舍五入后的值。
  - Signal Intensity：返回值必须为整数。当返回值小于 0，进度条默认按 0%显示，当返回值大于 100，进度条默认按 100%显示。
  - Signal Quality：返回值必须为整数。当返回值小于 0，进度条默认按 0%显示，当返回值大于 100，进度条默认按 100%显示。
  - C/N：返回值必须为浮点数或者整数。
  - Error Rate：返回值必须为浮点数或者整数。
- 频谱图操作

  当频谱图从函数读入画图时， C 脚本中定义的函数返回值应是：
  - 一个由二维点坐标构成的字符串：每个点之间用“#”分隔。
  - 点中 X 和 Y 轴坐标之间用英文逗号分隔。
  - X 和 Y 轴数据必须是整型或者浮点型。

  格式为：{x1,y1}#{x2,y2}#{x3,y3}# ……例如：{3,4}#{5,6}#{7,8}#。

  参考样例如下：

```
#include "stdio.h"
#define length 300
int fftdata(int argc, char ** argv )
```

```
{
int doubleArray[2][length];
int i;
int j;
for(i = 0; i < length; i++){
doubleArray[0][i] = i;
doubleArray[1][i] = i + 1;
}
printf("\n");
```

注意

"error=%s,return="应处于 for 循环外面。

```
printf("error=%s,return=","ok");
for(j = 0; j < (length- 1); j++){
printf("{%d,%d}#",doubleArray[0][j],doubleArray[1][j]);
}
return 0;
}
```

6.1.3 前提

- I2C 服务已经启动
- Socket 连接
  - HiChannel 工具菜单栏，DUT 网络中 Ip Address、Dev Address 和板端 Ip 相匹配
  - I2C Speed，DUT 网络中 I2C Port 与 Dev Address 匹配
- USB 连接

  HiChannel 工具菜单栏，USB 设置中 Dev Address 和板端相匹配

6.1.4 调用 I2C 读写

实现原理：C 函数脚本调用 C 编写的 dll 动态库建立通信，进行 I2C 读写。

Socket 连接

如果 HiChannel 菜单项中 DUT 网络、I2C 速率设置正确，在执行函数脚本的时候，会向 C 脚本函数默认传递七个参数：

- 第一个参数----板端 IP 地址，比如：192.168.0.1。
- 第二个参数----serverport(固定为 4000）。
- 第三个参数----I2cPort 十进制整型 ，比如 3。

- 第四个参数----I2c 速率(传递时不会把单位一起传递，例如速率设置了 100K，传递时只会传递 100)。
- 第五个参数----DevAddress（传递时会把页面中十六进制 DevAddress 转换为十进制后传递，例如 Dev Address 设置了 b6。传递时把 b6 转换为十进制 182 传递)。
- 第六个参数----函数名。
- 第七个参数----标识连接模式（-1 表示 USB，0 表示 Socket)。

int functionName(int argc，char ** argv)

argc 表示参数个数，argv 存储的是所有的参数值

在 C 脚本中要调用 I2C 读写的时候可以按索引将参数取出，参数索引从 1 开始，所以。

- argv[1] ----板端 IP(字符串类型)。
- argv[2]----serverport(整型)。
- argv[3]----I2cport(整型)。
- argv[4]----I2c 速率(整型)。
- argv[5]----DevAddress(整型)。
- argv[6]----(Param1)如果 Button 配置了 n 个参数，可以按序取出。
- argv[7]----(Param2)如果 Button 配置了 n 个参数，可以按序取出。
- .argv[n]----(Param n)如果 Button 配置了 n 个参数，可以按序取出。
- argv[n+1]----需要调用的函数名，不带#号，取 argv[argc-2]就可以取到函数名。
- argv[n+2]----连接模式，取 argv[argc-1]就可以取到连接模式。

USB 连接

如果 HiChannel 菜单项中 USB 配置正确，在执行函数脚本的时候，会向 C 脚本函数默认传递四个参数：

- 第一个参数----USB 速率。
- 第二个参数----DevAddress。
- 第三个参数----函数名。
- 第四个参数----标识连接模式（-1 表示 USB，0 表示 Socket)。

int functionName(int argc，char ** argv)

argc 表示参数个数，argv 存储的是所有的参数值。

在 C 脚本中要调用 I2C 读写的时候可以按索引将参数取出，参数索引从 1 开始，所以：

- argv[1] ----USB 速率 (整型)。
- argv[2] ----DevAddress(整型)。
- argv[3]----(Param1)如果 Button 配置了 n 个参数，可以按序取出。
- argv[4]----(Param2)如果 Button 配置了 n 个参数，可以按序取出。

- .argv[n]----(Param)如果 Button 配置了 n 个参数，可以按序取出。
- argv[n+1]----需要调用的函数名，不带#号，取 argv[argc-2]就可以取到函数名。
- argv[n+2]----连接状态，取 argv[argc-1]就可以取到连接模式。

注意

所有的参数在传递到 C 脚本时都会是字符串类型，所以在使用参数的时候，对于不同的参数要根据它们的类型进行类型转换，否则，会导致程序加载超时，程序无响应。

6.2 脚本函数编写说明

- 在函数中
  - Socket 连接：

    在工具连接状态正常状态下，在读写寄存器的时候需特别注意：单个寄存器读写时，每次都要 InitSock,DeinitSock；批量读写寄存器时，只在开始结束的时候 InitSock,DeinitSock 一次。
  - USB 连接：

    采用 USBIO_StreamI2C 函数接口进行读写。
- C 脚本之间函数可以相互调用，但是调用时，必须确保语法符合 C 语法规范。

注意

- **C 脚本中对于内部使用的函数，必须在其对应的函数前标识"static"，这样子才能防止被外部调用。在一个 C 函数脚本中可以编写多个函数。**
- **C 脚本文件路径列表下定义的所有 C 脚本中的函数名称不能重复。**

- C 脚本中，如果定义的函数只是用来被其他 C 脚本函数调用，而不想在 HiChannel 工具 UI 的函数下拉列表中显示时，应该在其对应的函数名前加上“inner_”标识。例如 int inner_compile(int argc，char**argv)。

6.3 样例

注意

样例只作为编写 C 函数格式参考，并不能直接使用。

当需要进行 I2C 读写时，首先需要自定义一个头文件来定义调用函数，头文件参考定义样例如下：

```
#define CHIP_PORT atoi(argv[3]) //在DUT网络连接时，argv[3]定义了I2C port
#define CHIP_ADDR atoi(argv[5])//在DUT网络连接时，argv[5]定义了设备地址
#define USB_ADDR atoi(argv[2]) //在USB连接模式时，argv[2]定义了USB的设备地址
#define USB_SPEED atoi(argv[1]) //在USB连接模式时，argv[1]定义了USB的速率
//g_connect标识是Sockect连接还是USB连接，g_connect 等于0时，为USB连接，
g_connect非0时，为Sockect连接
#define INIT_CONNECT if (g_connect) {\
InitSock(argv[1],atoi(argv[2])); \
}else{\
if(USBIO_OpenDevice(0) == INVALID_HANDLE_VALUE ){ \
printf("error=%s,return=%d","err openerror,failed",-1); \
return; \
} \
USBIO_SetStream(0,USB_SPEED);\
}
#define DEINIT_CONNECT if (g_connect) {\
DeinitSock();\
}else{\
USBIO_CloseDevice(0);\
}
#define CHIP_MRD(REG_ADDR, P_RBUF, RLEN) if (g_connect)
{ ReadregMulti((CHIP_PORT), (CHIP_ADDR), (REG_ADDR), 1, ((char
*)(P_RBUF)), (RLEN)); \
} else {\
unsigned char P_WBUF[2];\
P_WBUF[0] = USB_ADDR;\
P_WBUF[1] = REG_ADDR;\
USBIO_StreamI2C(0,2,(PVOID)pSend,RLEN,(PVOID)(P_RBUF));\
}
#define CHIP_MWR(REG_ADDR, P_WBUF, WLEN) if (g_connect)
{ WriteregMulti((CHIP_PORT), (CHIP_ADDR), (REG_ADDR), 1, ((char
*)(P_WBUF)), (WLEN));\
} else {\
unsigned char P_RBUF;\
char *p = (char *)malloc((WLEN)+2);\
char *pw =(char *)(P_WBUF);\
int i = 0;\
p[0]= USB_ADDR;\
p[1]= REG_ADDR;\
for (i=0;i<(WLEN);i++) {\
```

```
p[i+2] = pw[i];\
};\
USBIO_StreamI2C(0,2+(WLEN),(PVOID)p,0,(PVOID)(&cRecv));\
}
```

定义了头文件后，在编写 C 脚本开头需要引入它，C 函数定义的样例如下：

```
#include "stdio.h"
#include "string.h"
#include 自定义的头文件
#include "USBIOX.H" //必须要引入
extern int g_connect; //这行代码是必须的
int on_line(int argc, char ** argv)
{
INIT_CONNECT;
unsigned char chip_id[4];
unsigned char on_line;
CHIP_MRD(0x6f, &chip_id, 4);
on_line = (chip_id[2]==0x36) && (chip_id[3]==0x31);
if(on_line) {
printf("error=%s,return=%d","ok",1); }
else {
printf("error=%s,return=%d","ok",0); }
DEINIT_CONNECT;
}
//内部私有函数，使用‘static’来标识
static int demo_m2( int argc, char ** argv )
{
int i = 0;
printf("demo_m2........");
return 0;
}
//只是用来被外部调用，而不在工具UI函数列表中显示，此时使用‘inner_’来标识
int inner_compile( int argc, char ** argv )
{
int i = 0;
printf("I'm inner function");
return 0;}
```

7 高级功能菜单

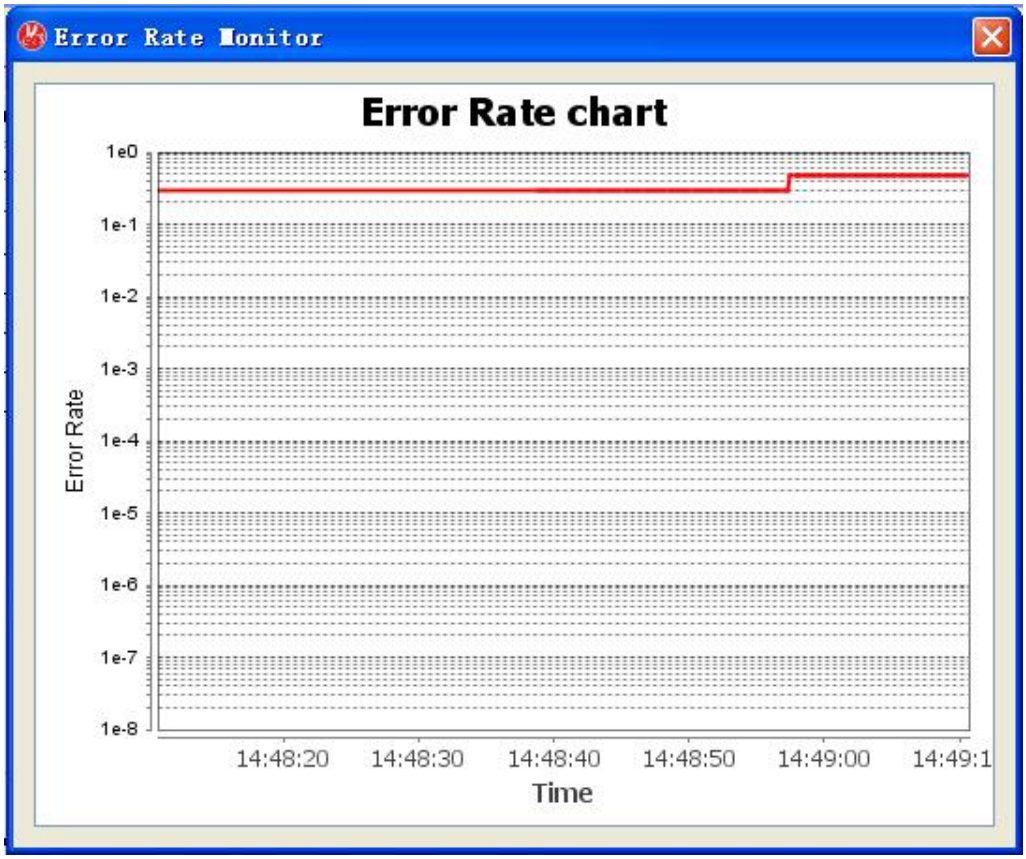
高级菜单下包括如下选项：

- 错误率监控
- CN 监控
- 星座图观察
- 频谱观察
- 时域观察

7.1 错误率监控

错误率监控界面如图 7-1 所示。

图7-1 错误率监控

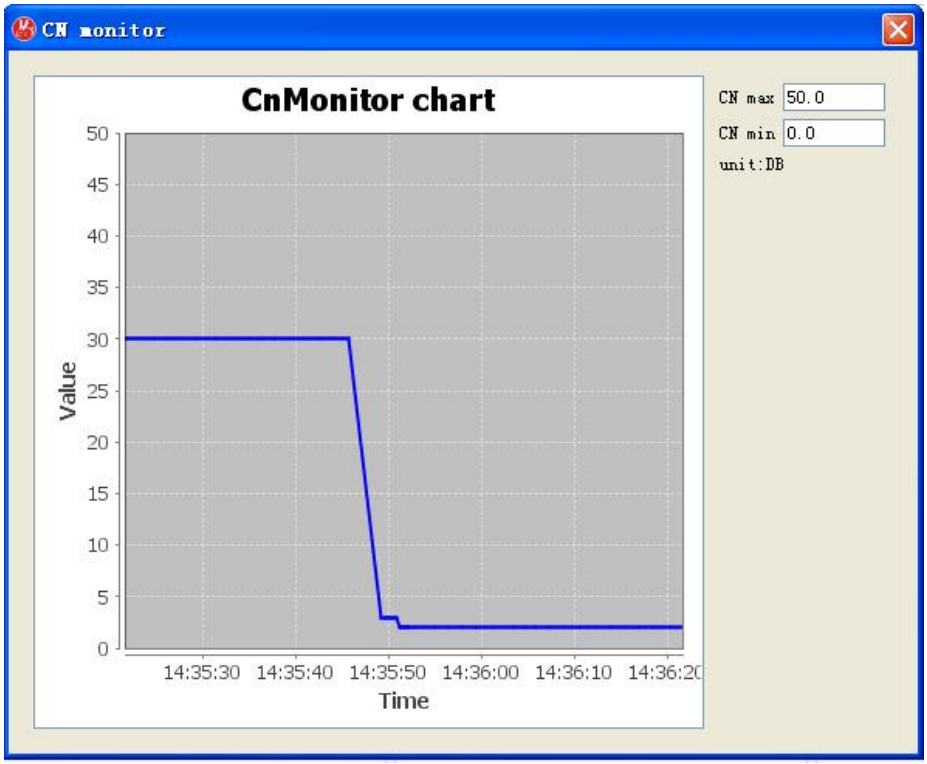


错误率的数据与基本功能中的错误率挂钩，即数据来自同一函数，错误类型也相同。显示最近 300 次错误率，1 次是指基本功能中所描述的一次定时读取错误率。停止自动刷新后监控停止。

7.2 CN 监控

CN 监控界面如图 7-2 所示。

图7-2 CN 监控

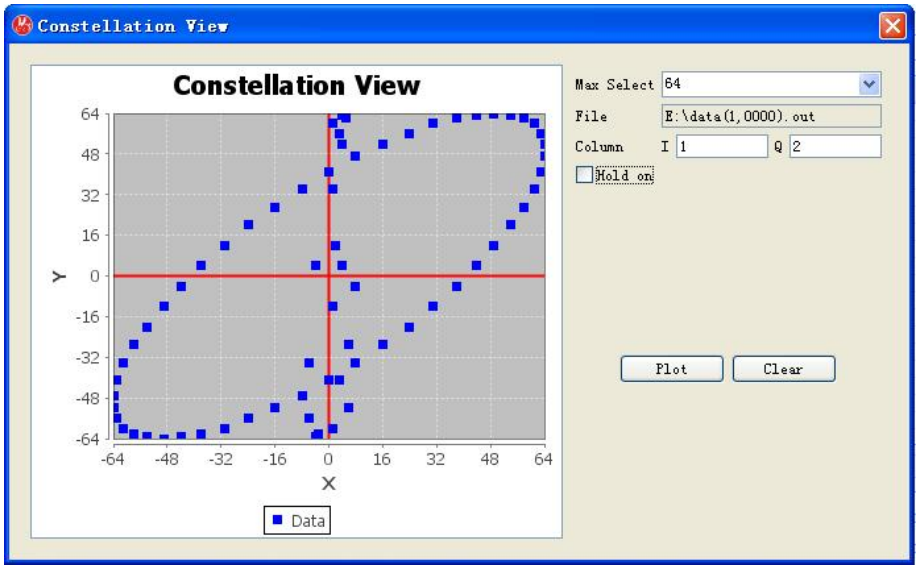


CN 的数据与基本功能中的 CN 监控挂钩，即数据来自同一函数，显示最近 300 次数据，停止自动刷新后监控也停止。点击“CN 最大值”右边方框，输入 CN 刻度最大值；点击“CN 最小值”右边方框，输入 CN 刻度最小值，前面已记录的 CN 数值仍然有效，并且会根据新的设置打印到正确位置。

7.3 星座观察

星座观察界面如图 7-3 所示。

图7-3 星座观察

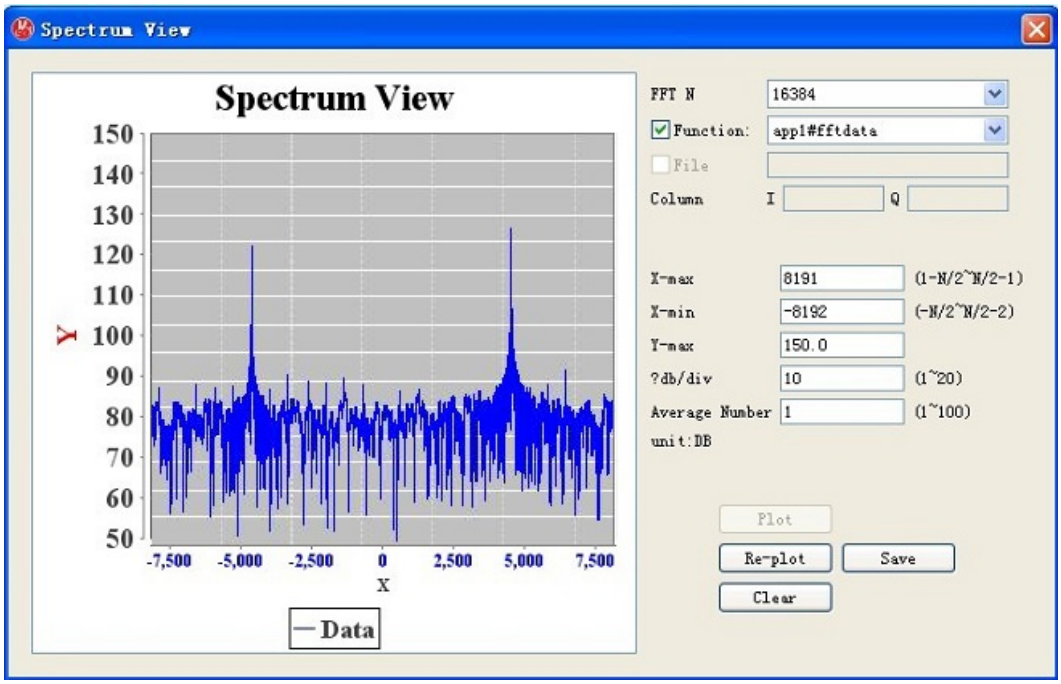


点击“Max Select”选择坐标最大值后，点击“文件”右边方框选择文件，在“列”右边输入 I 和 Q 的值，点击“打印”则打印数据，点击“清除”则清除显示，若选中 Hold on（框中加“√”），则表示在原图上继续打印，否则每次重新打印。

7.4 频谱观察

频谱观察界面如图 7-4 所示。

图7-4 频谱观察



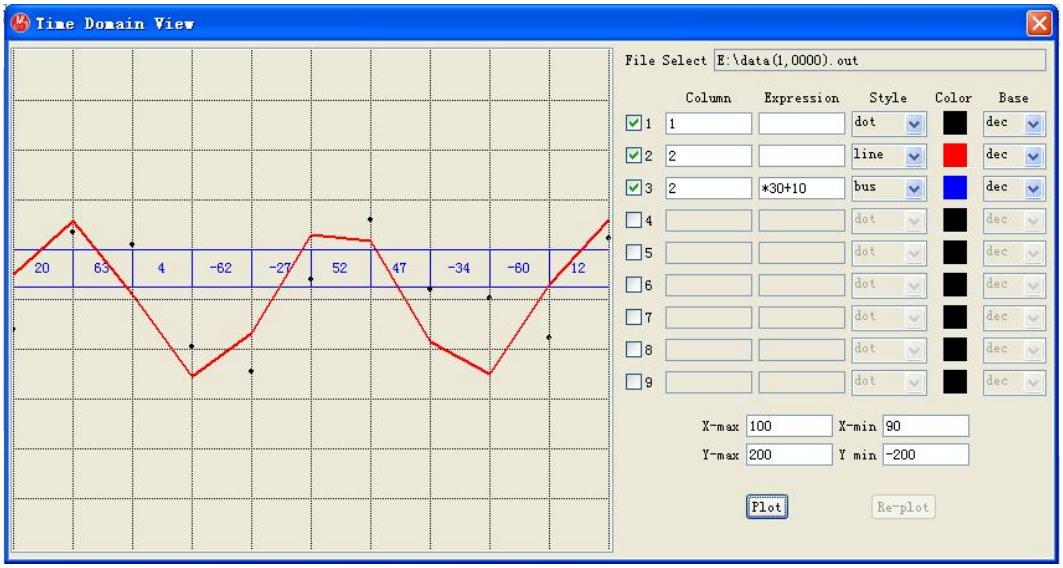
数据源可以由 C 脚本函数生成，或指定数据文件，二选一，选中项的前面方框中加“√”，未被选中的变灰色，不能操作。

- 当选 C 脚本函数时在“Function“的右侧从脚本函数列表中选择函数。
- 当选择数据文件时，点击“File”右边方框，选择文件.点击“Column”右边两方框，分别输入 I 和 Q 在文件中的列号。
- 点击“X-max”右边方框，输入 X 轴最大值（1-N/2~N/2-1，整数）。
- 点击“X-min”右边方框，输入 X 轴最小值（-N/2~N/2-2，整数）。
- 点击“Y-max”右边方框，输入 Y 轴最大值，浮点数。
- 点击“? dB/div”右边方框，输入每格多少 dB（1~20，整数）。
- 点击“Avarage Number”右边方框，输入平均次数（1~100，整数）。
- 点击“Plot”则绘制频谱图。
- 点击“Rlot-plot”按钮，则根据新设置重新绘制（不读文件），修改输入设置后，如果不点击“重新打印”、“打印”按钮，是不会起作用的。
- 点击“Clear”按钮，则清除频谱显示。
- 点击“Save”可以将图形存到指定文件中。

7.5 时域观察

时域观察界面如图 7-5 所示.。

图7-5 时域观察



首先点击“File Select”右边的方框，选择文件。在文件中选择最多 9 个信号用于显示（信号可以重复被选中），点击 1~9 前面的方框则选中（框中加“√”）。被选中后方框后面的 5 个操作项（Column， Expression，Style，Color，Base）允许操作，否则呈灰色是不允许操作的。

- 点击“Column”下的方框，输入该信号在文件中的列号。
- 点击“Expression”列的长方框，可以输入表达式，如无运算，可输入“+0”.表达式中乘法可用于调整信号显示的幅度，减法用于调整信号显示时的上下位置，通过设置合理的表达式，可以实现多个信号无重叠的显示。
- 点击“Style”列下拉选择打印风格，共有 bus、dot 和 line 三种选择。
- 点击“Color”列下拉选择打印颜色。
- 点击“Base”列下拉选择进制，按序有 dec、hex、bin，分别表示 10、16、2 进制。
- 点击“X-max”右边方框，输入到第几个点停止打印。（打印内容包括这个点）
- 点击“X-min”右边方框，输入从第几个点开始打印。（打印内容包括这个点）
- 点击“Y-max”右边方框，输入 Y 轴最大值（浮点数）。
- 点击“Y-mix”右边方框，输入 Y 轴最小值（浮点数）。
- 点击“Plot”按钮，则开始绘图。
- 点击“Re-plot”按钮，则根据新设置重新绘图（不读文件）。修改了 Y-min 或者 Y-max 的值后，如果不点击“Re-plot”、“Plot”按钮，是不会起作用的。
  - 当“样式”选择 dot 时，每个数据对应 1 个点，其 Y 轴坐标为将文件所读数值代入表达式的计算结果。
  - 当“样式”选择 line 时，等于将上述 dot 方式的点用线连起来，但是点不再打印。

- 当“样式”选择 bus 时，按总线方式打印，格子中间显示该点的数值（文件中读出未经过表达式运算的原始数据，按“Base”指定的进制显示）。当屏幕内待显示数据量太大时会导致格子太小，无法显示全部数值，则只显示前面部分，超过部分不显示，格子上边线的幅度为将 1 代入表示达的计算结果，格子下边线的幅度为将 0 代入表示达的计算结果。第 x 点到第 x+1 点之间的格子内显示的是第 x 点的内容。

注意

表达式中只能含“+-*/”或者数字，表达式的第一个字符必须为运算符号，下来的输入必须符合四则运算的规则。 例如：*30-10。

7.6 数据文件格式定义

本章节介绍在操作频谱图、星座图和时域图时，对需要加载的文件格式的约定。

7.6.1 约定

- 文本文档中第一行必须给出每行的列数，格式为：(Column:2)。Column 不区分大小写，中间必须用英文“:“分隔，列数为十进制，列数>0。
- 用英文逗号区别列，例如：-1，2 即-1 为第一列，2 为第二列，这一行总共有 2 列。
- 第一行给出列数定义后，下面数据的每一行的列数都必须大于或者等于定义的列数，否则，这一行数据为不合法，将不会打印这一行的坐标点数据。
- 坐标点数据必须为正负十进制整型或者浮点型。

7.6.2 样例

```
Column:2
-1,2
2,3
2,5
4,
```

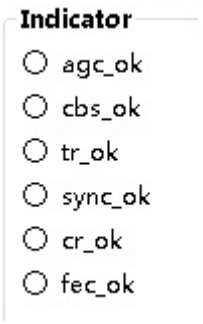
注意：样例中最后一行与第一行定义的列数 2 不匹配，所以这一行坐标点将不会被打印。

8 客户版功能介绍

8.1 指示灯栏 Indicator

指示灯栏界面如图 8-1 所示

图8-1 指示灯

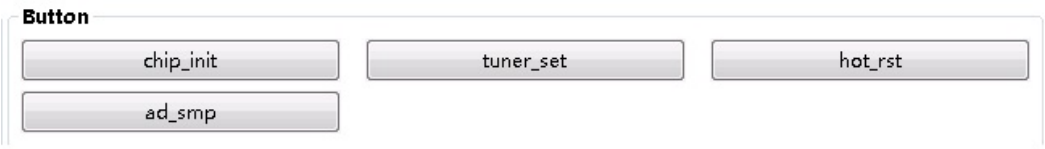


- agc_ok：绿色表示 AGC 锁定，红色表示未锁定。
- cbs_ok：绿色表示盲扫成功，红色表示不成功。
- tr_ok：绿色表示定时锁定，红色表示未锁定。
- sync_ok：绿色表示 S2 信号帧同步成功，红色表示不成功。
- cr_ok：绿色表示载波锁定，红色表示未锁定。
- fec_ok：绿色表示纠错译码成功，红色表示失败。

8.2 按钮操作栏 Button

按钮操作栏界面如图 8-2 所示。

图8-2 按钮操作



- chip_init：完成芯片的初始化配置。无输入参数。无返回值。
- tuner_set：完成 tuner 的初始化和载波中心频率/tuner 滤波器带宽配置。输入参数：
  - 1.tuner I2C 通道号。
  - 2.tuner I2C 器件地址。
  - 3.tuner 类型(0-AVL2012，其他尚未添加)。
  - 4.tuner 载波中心频率(MHz)。
  - 5.tuner 低通滤波器双边带宽(MHz)。无返回值。
- hot_rst：完成芯片的软复位。无输入参数。无返回值。
- ad_smp：完成芯片的内置采数功能。输入参数：1。采数类型(0-采集 AD 输出数据，1-采集均衡输出数据)。输出至文件 hi_ad_smp，共 2048 点采样。

8.3 跟踪显示栏 Trace

跟踪显示栏界面如图 8-3 所示。

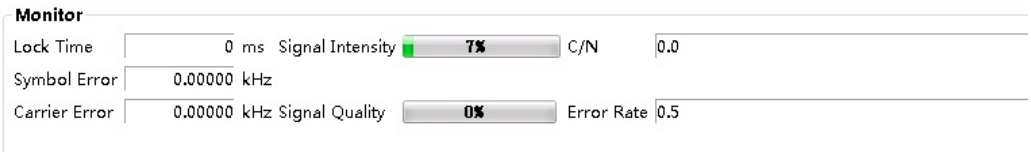
图8-3 跟踪显示

Signal Basic：在接收信号锁定后显示基本信息，包括 DVB-S2/DVB-S/DirecTV 标准的哪一个、调制模式、码率、码长、有无导频、是否 CCM 等信息；信号未锁定情况下显示 unkown standard。

8.4 常用监控栏 Monitor

常用监控栏界面如图 8-4 所示。

图8-4 常用监控

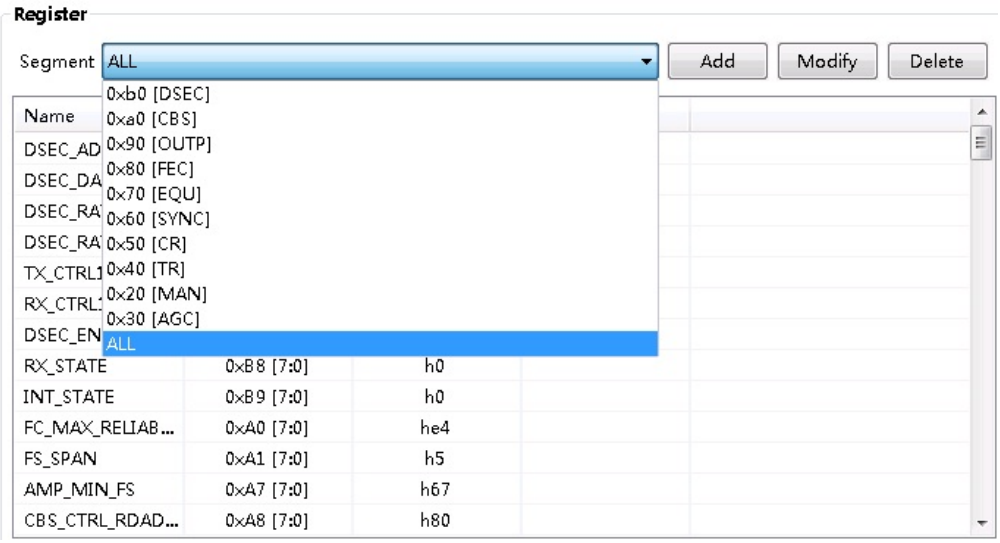


- Lock Time：显示芯片从复位到 fec_ok 所用的时间，单位 ms，每次复位后更新。
- Symbol Error：显示当前信号的符号率与盲扫得到的符号率之偏差，单位 kHz，为正表示当前符号率偏大。
- Carrier Error：显示当前信号的载波中心频率与盲扫得到的载波中心频率的偏差，单位 kHz，为正表示当前载波频率偏高。
- Signal Intensity：信号强度指示。
- Signal Quality：信号质量指示，尚未实现。
- C/N：信噪比指示，单位 dB，仅当信号锁定后有效，信号未锁定情况下显示-10.0。
- Error Rate：误码率指示，仅当信号锁定后有效，信号未锁定情况下显示 0.5。

8.5 分段信息段 Segment

分段寄存器界面如图 8-5 所示。

图8-5 分段寄存器



- MAN：芯片状态及复位控制寄存器组
- AGC：AGC 控制及状态寄存器组
- TR：定时恢复控制及状态寄存器组

- CR：载波恢复控制及状态寄存器组
- SYNC：帧同步控制及状态寄存器组
- EQU：均衡控制及状态寄存器组
- FEC：纠错译码控制及状态寄存器组
- OUTP：TS 接口控制寄存器组
- CBS：盲扫控制及状态寄存器组
- DSEC：iseqc 状态及控制寄存器组

8.6 操作流程

步骤 1 连接信号线、电源线、串口线，网口线、上电。

步骤 2 通过串口设置板端 IP 地址。

步骤 3 打开 HiChannel 工具，通过点击工具栏连接管理器 Dut NetWork,可以设置板端 IP，芯片的 I2C 通道号，芯片的 I2C 器件地址，I2C 通信速率(kHz)，（具体可参考 3.2.2 的介绍）。

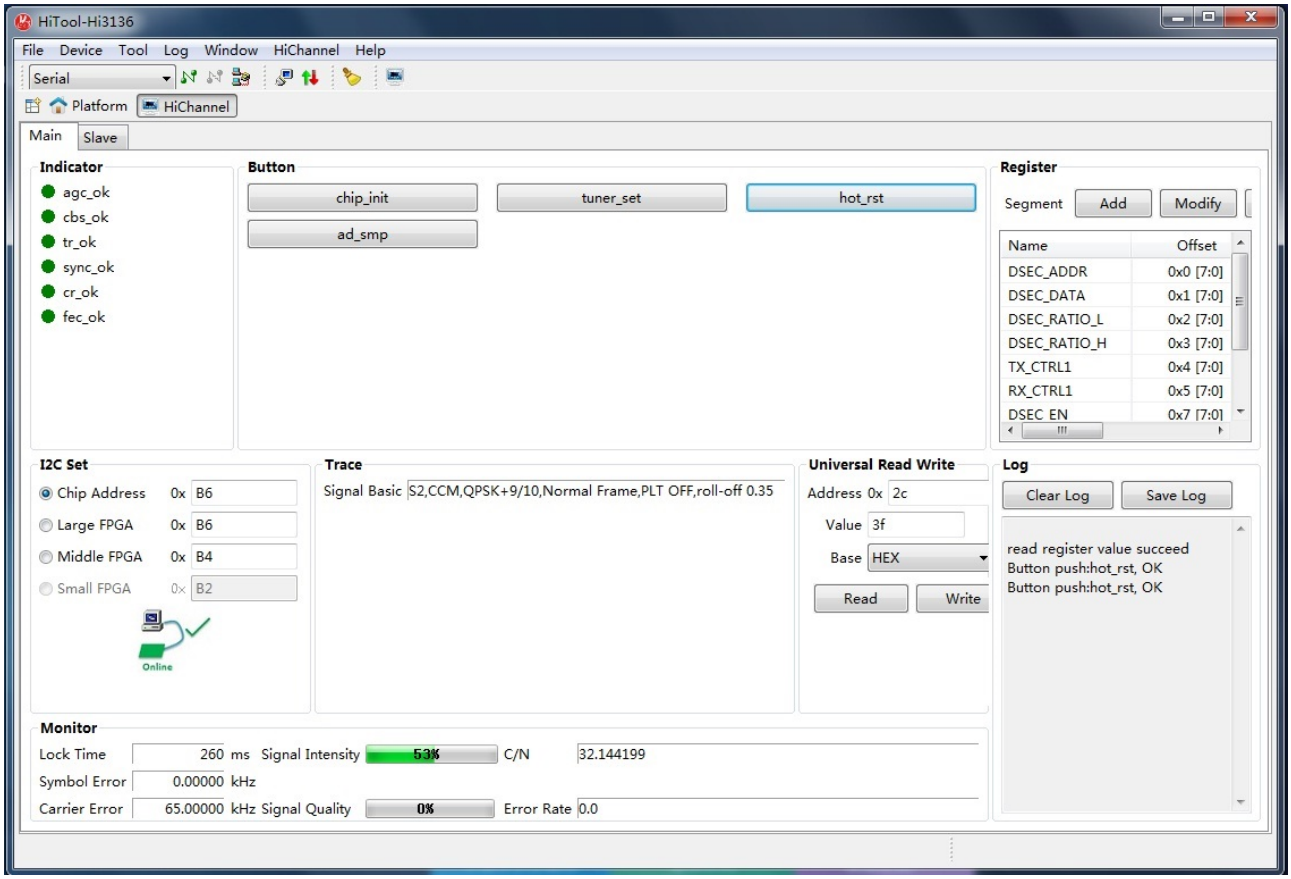
步骤 4 点击 chip_init 按钮。

步骤 5 点击 tuner_set 按钮，根据实际配置置入 tuner I2C 通道号/tuner I2C 器件地址/tuner 类型/载波中心频率/低通滤波器双边带宽。

步骤 6 点击 hot_rst 按钮。

如果信号质量满足芯片接收要求，此时指示灯栏变绿(DVB-S/DirecTV 标准下 sync_ok 不会变绿)，trace 栏给出信号的基本信息，Monitor 栏给出锁定时间、符号率偏差、载波频率偏差、信号强度、信噪比、误码率等状态指示，整体界面如图 8-6 所示。

图8-6 整体界面



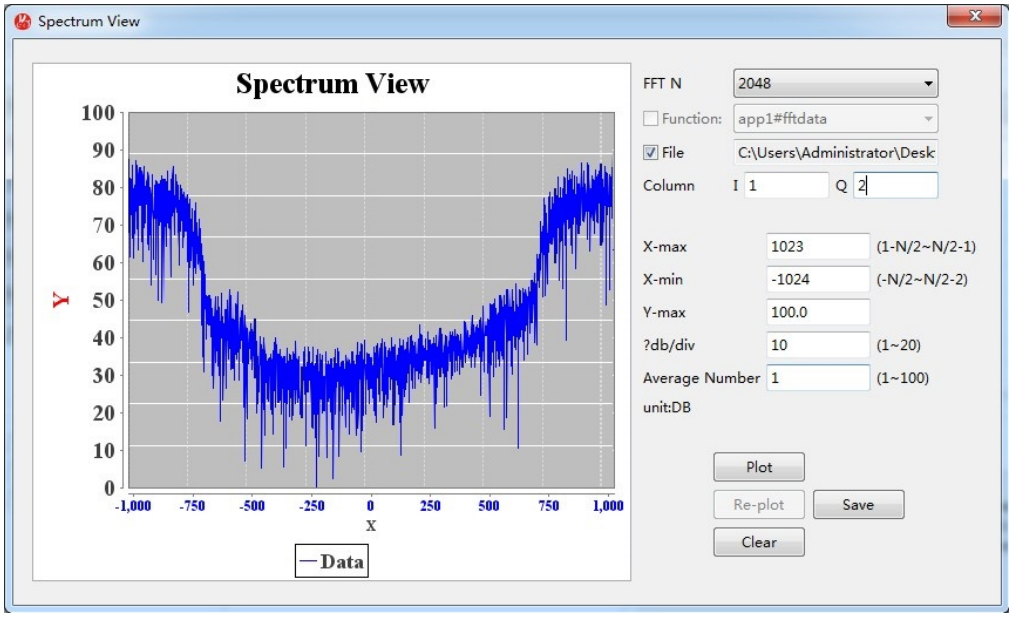
步骤 7 接下来可根据需要读写寄存器，或者通过 ad_smp 按钮采集数据，然后通过菜单 HiChannel -> Advance 中的工具做频谱、星座、时域波形的显示。

----结束

8.7 频谱显示

通过菜单 HiChannel -> Advance -> Spectrum View 可显示数据频谱。图 8-7 显示的是通过 ad_smp 按钮采集的输入信号频谱，信号符号率 30Mbaud，采样率 125MHz。

图8-7 频谱显示

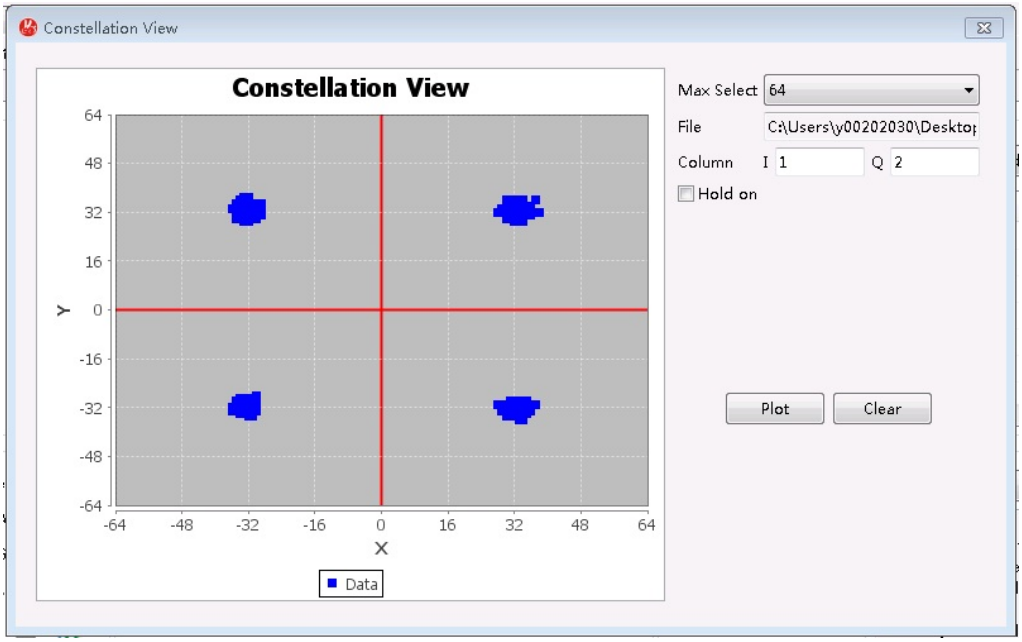


通过右边的界面可以改变 FFT 点数、横纵轴范围、平均次数等参数。

8.8 星座显示

通过菜单 HiChannel -> Advance -> Constellation View 可显示数据星座。图 8-5 显示的是通过 ad_smp 按钮采集的均衡输出信号星座，信号为 QPSK，SNR 约 30dB。

图8-8 星座显示

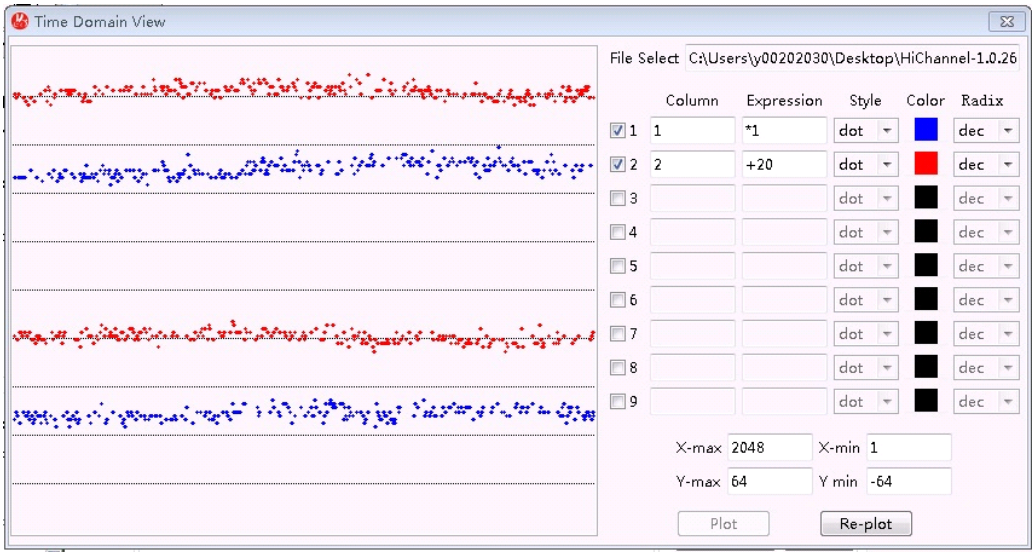


通过右边的界面可以改变横/纵轴范围。

8.9 时域显示

通过菜单 HiChannel -> Advance -> TimeDomain View 可显示数据时域波形。图 8-6 显示的是通过 ad_smp 按钮采集的均衡输出信号时域波形，信号为 QPSK，SNR 约 30dB。

图8-9 时域波形显示



通过右边的界面可以对输入数据做缩放和偏移，并改变显示类型和颜色。

9 注意事项

- 配置文件必须遵循特定的格式，否则无法加载配置文件或出现运行错误。不建议用户手工修改配置文件，除非对配置文件格式很熟悉。
- C 脚本文件若存在语法错误，将会导致脚本加载失败，语法错误会在加载时进行报告。
- 由于 C 脚本采用内编译运行模式（在加载成功后，C 脚本将被直接编译为可执行代码进入本工具的代码空间执行），如果 C 脚本出现运行错误误操作(如对空指针进行操作)，将导致不可预料错误，甚至整个工具崩溃死掉。

10 FAQ

10.1 提示 Invalid Word Register bit area info\: start bit must be xx at line xx 错误

问题描述

加载配置文件时，提示“Invalid Word Register bit area info\: start bit must be xx at line xx”。

问题分析

这是因为配置文件中，[Word Register]段的配置存在错误，字寄存器定义时起始比特和结束比特固定为[8*RegisterBytes-1:0]，和配置文件中[Global]段中的 RegisterBytes 设置的值相关

解决办法

若配置文件中[Global]段 RegisterBytes 值为 1，则[Word Register]段寄存器定义时起始比特和结束比特固定为[7:0]。

10.2 加载配置文件时，提示“Invalid Word Register content : bit count error at line xx ”

问题描述

加载配置文件时，提示“Invalid Word Register content : bit count error at line xx ”。

问题分析

这是因为配置文件中，Word Register settings 处配置存在错误，偏移位域地址后面的位数存在错误，例如：[7:0]=32，这个时候便会存在错误，

解决办法

正确应是[7:0]=8。

10.3 Register 区域中右键菜单中“Write by file”时，提示“Segment name in the file is not exist in the register area”

问题描述

Register 区域中右键菜单中“Write by file”时，提示“Segment name in the file is not exist in the register area”。

问题分析

这是因为使用右键从文件写入时，文件中指定的 Segment 名字，在当前 Register 区域的段地址列表中不存在 。

解决办法

需要在 Register 区域添加相应的 Segment 后，再重新操作。

10.4 Register 区域中右键菜单中“Write by file”时，提示“please check register type at first line ”

问题描述

Register 区域中右键菜单中“Write by file”时，提示“please check register type at first line ”。

问题分析

这是由于写入的文件第一行指定的寄存器的类型和当前 Register 区域的寄存器类型不符合。

解决办法

在菜单中[Setting]项中切换寄存器类型，或者在文件中把寄存器类型改为和当前寄存器区域匹配的类型（建议对文件格式十分熟悉的话再使用第二种方法）。

10.5 Register 区域为 WrodRegister 类型时，右键菜单单击“Write by file”时，没有提示错误，为什么寄存器列表读取不完整或者没有被读进来

问题描述

Register 区域为 WrodRegister 类型时，右键菜单单击“Write by file”时，没有提示错误，为什么寄存器列表读取不完整或者没有被读进来

问题分析

由于写入的文件中寄存器的定义和当前 Register 区域的寄存器类型不符合

解决办法

必须确保右键读取的目标文件里面的所有寄存器偏移位域地址的起止值=8*RegisterBytes-1，例如当前 RegisterBytes 为 1，目标文件的偏移地址格式必须是：[7:0]=8。

A 缩略语

A		
AGC	Automatic Gain Control	自动增益补偿
B		
Bin	binary	二进制，机器代码，汇编语言编译后的结果。
C		
CCM	Constant Coding and Modulation	固定调制及编码
D		
Dec	Decimal	十进制。
DDR	Double Data Rate	双倍速率同步动态随机存储器。
F		
FPGA	Field－Programmable Gate Array	它是作为专用集成电路（ASIC）领域中的一种半定制电路而出现的，既解决了定制电路的不足，又克服了原有可编程器件门电路数有限的缺点。
FFT	Fast Fourier Transform	快速傅里叶变换
H		
Hex	Hexadecimal	十六进制，计算机中数据的一种表示方法。
HDMI	High Definition Multimedia Interface	一种数字化视频/音频接口技术。
I		

I2C	Inter－Integrated Circuit	两线式串行总线，用于连接微控制器及其外围设备。
Q		
QPSK	Quadrature Phase Shift Keying	正交相移键控，是一种数字调制方式
S		
SNR	Signal-to-noise ratio	信噪比