目录

[一、网页版配置工具 2](#_Toc451781317)

[二、高级管理员工具 3](#_Toc451781318)

[三、触摸屏程序 5](#_Toc451781319)

[四、内外网转发工具 7](#_Toc451781320)

[五、基线2.0模拟信号机软件 8](#_Toc451781321)

[六、经济型信号机灯控板及前面板MCU程序 9](#_Toc451781322)

[七、基线1.0倒计时、单时段通道锁定和配置管理模块以及自动构建升级包 11](#_Toc451781323)

[八、基线2.0配置校验、倒计时器、流量统计、SADP 13](#_Toc451781324)

一、网页版配置工具

* 基本描述

该工具后端是基于开源HTTP服务器GoAhead编写，前端是基于HTML CSS/DIV AJAX JS编写。服务器端程序运行后，在浏览器上输入信号机的IP地址，就可以访问信号机，用户名和密码可以随便填。服务器端程序启动后，会通过socket向hikTSC进程发送获取配置参数的请求(这个操作只在系统启动及页面手动刷新时执行)。该配置参数会以全局参数的形式保存在内存中，供用户进行查询和修改。

* Svn路径:

<https://192.0.0.75/TrafficSignalController/ARM/trunk/baselineV1.0/new/webApp>

* web服务器部署方案:

1. 在/root/下面放置服务器程序webs.

2. 在/wwwdemo/下面放置网页代码.

* 工程目录：

1. webApp\GoAhead\LINUX\里面就是服务器程序，可以支持服务器及嵌入式linux两种版本。这里着重关注DataExchange.c和WebsCallback.c这两个文件，前者提供全局参数和网页间请求的数据交换，简单理解就是一些getter setter方法。后者用来处理网页url的请求，解析封装xml协议，完成数据在服务器和网页间的传递。

2. webApp\GoAhead\wwwdemo\webs里面是所有的网页代码。这里主要关注script和page里面的内容。

Script/paramconfig.js 该文件主要用是用来获取、保存各个表，这些全局变量会被各自的页面调用，用来完成数据的自动补全。

Script/trafficParams.js 该文件主要是通过AJAX完成页面和服务器之间的通讯，这里面的getter setter方法对应于服务器里面的getter setter方法。

Page/paramconfig.asp 该文件定义了主界面的各个按钮和树形图，以及各个item的右键功能。

Page/params/traffic/\*.asp 这些文件就是各个表的具体内容了，根据文件名可以很容易的知道表格的意义。这些网页对应于服务器里面的webscallback，具体的映射关系在main.c里面的initwebs函数中可以找到。

* 项目分析

Main.c:initWebs() 这里websFormDefine连接了各个请求url与处理函数。

Main.c:GetLocalHost() 这里指定了信号机监测的IP:PORT，也就是通过URL访问时的信号机IP。

Main.c: InitGolobalVar() 这里用来初始化信号机配置的全局参数，只有在hikTSC程序运行时，才能读取到配置。

* 注意点

目前相位表、通道表等依然是一维表，如果要对接2.0新库，底层和应用层均需要进行修改。

* 相关资料

Html javascript ajax相关知识可以在<http://www.w3school.com.cn/>中了解到。

二、高级管理员工具

* 基本描述

该工具的初衷是前期方便技术人员直观查看核心配置、批量升级、免密登陆信号机、直观修改dat配置；中期根据向导帮助技术支持定位各种常见问题、解决部分问题；后期无需信号控制理论直观配时、模拟信号机功能。目前第一期的功能已经完成，大家有兴趣了解QT的，可以看看，目前不会在实际项目中使用。

* Svn路径:

https://192.0.0.75/TrafficSignalController/ARM/branches/工具类软件/高级管理员工具

* 工程目录：

Base/\*.c \*.h 这里的代码内容和信号机2.0库里面是一致的，可以直接copy过来使用，主要用到了转换工具、配置合法性校验、ini文件解析等。

Delegate/\*.cpp \*.h 这里就是QT所特有的代理类的实现，主要用来将数据库里的内容和界面的内容分离。每一个表都有自己的代理类。

Exe/Notepad++ putty winscp 这里面是将我们日常使用的文本编辑、SSH登陆、远程拷贝工具集成到该高级工具中，实现直接修改dat文件、免密登陆信号机，免密拖曳文件。如有需要，可以将其他免安装工具集成进去。

* 部署方案：

这里只介绍windows下获得qt可执行程序依赖项的方法， QTDIR/bin/windeployqt这里的windeployqt是个很好的工具，在windows命令行下，执行该命令，空格后紧跟exe的绝对路径，就能将该exe所有的依赖找到。将包含该exe的目录拷贝到其他PC即可运行，也可以通过其他方式实现安装包。

* 项目分析：

checkDialog 这个界面类是用来提供登陆用户名密码验证的，因该工具具有极高的系统权限，在不设置用户名和密码的情况下，切勿发布到外面。默认用户名是Administrator 密码是AsdF1234!@#$

CoreWindow 这个界面类创建了该高级工具的主界面。包括菜单栏、工具栏、左侧信号机管理、右侧信号机配置管理等。该工具所提供的所有功能，目前都是在该类中引用的。包括导入、导出、批量升级、在线设备搜索、参数获取、参数下发、SSH终端启动、WinSCP启动、高级编辑功能等。

AddTscDialog 这个界面类用来实现信号机基本信息的添加和修改。

BatchUpgrade 这个界面类用来实现批量升级功能，这里的批量升级只需要选择存放tsc300 tsc500升级包的文件夹，选择信号机后升级即可，里面会自动根据设备类型发送指定的升级包。

Configurewidget 这个界面类，只是实现了一个简单的widget，用来展示各个配置表的具体内容，目前主要是以二维表的形式展示。这里会使用到QT的MVC架构，具体可以参考QT 助手，里面详细介绍了代理类的使用，我们目前用代理类来实现全局参数与界面值的映射。

DataManager 这个类其实是一个线程，主要是用来异步操作数据库的，实现数据的导入导出和进度显示。

EditBinaryDialog 这个界面类实现了高级编辑功能，其实就是通过ftp协议获取数据后保存到本地文件，再调用editplus打开。里面的核心还是转换工具里面的代码实现。

FileTransfer 这里实现了ftp协议，通过调用该类可以实现通过ftp协议进行文件的上下载。

GetParamsProgress 获取信号机参数，并提供进度展示。

ImportDialog 该界面类实现了导入导出的基本功能。

initdb.cpp 通用接口，实现各个表与数据库之间的读写。

MainWindow 临时类，只是用来测试流量检测时用的。

NetComm 单例实现udp通讯。

OnlineDevice 在线设备搜索

onlineTreewidget 信号机列表管理

tabWidget 信号机配置管理。

* 注意点：

1. QT工程目录不能存在中文，否则编译可能有问题。

2. QT 助手是个很有用的工具，所有函数或模块都能在那里找到清晰的用法。

* 相关资料

1. 《 Qt5开发及实例》

2. 《QT高级编程》

3. 《C++ GUI Qt 4编程（第二版）》

三、触摸屏程序

* 基本描述

触摸屏程序,是基于QT QML编写,编译后能够运行在Android及其他移动平台上。前端页面采用QML编写，后端业务逻辑采用C++编写。目前已实现信号机同步状态的实时显示、通道状态实时显示、通道描述显示、方位自定义等基本功能。

* SVN路径

https://192.0.0.75/TrafficSignalController/ARM/branches/工具类软件/触摸屏程序

* 工程目录

NewHikAPP.pro 是工程文件，使用QT Creator可以直接打开

main.cpp 入口函数源文件

net.cpp 网络参数获取及下发

roadmap.cpp 主要的业务逻辑，实现显示QML页面、获取实时数据及接收页面按键控制。

Road.qml 主要的界面文件，定义了整个界面的框架。即页面分左右两部分，左边显示路口图，右边显示同步状态。其中路口图中有用来显示上下左右四个方向的通道倒计时状态及通道。为防止误操作，控制按钮隐藏在最下面，需要双击或者拖动页面向上才能显示出来。信号机的IP地址、通讯端口及指南针的方向都是存储在配置文件中的(用户不可见)，用户只需要配置一次后，下次启动自动载入上次配置。

VehicleLampsH.qml

VehicleLampsV.qml

VehicleLampsVT.qml 这三个页面来实现上下左右四个通道的状态展示。分别表示水平方向的、垂直方向及垂直相反方向的。

TopTitle.qml 在road.qml中被引用，提供IP和端口供用户使用。

* 部署方案

项目在android构建套件编译完成后，会在相应的工程目录中生成apk文件用来安装在android系统中。如：

D:\QtProjects\build-NewHikAPP-Android\_for\_armeabi\_v7a\_GCC\_4\_9\_Qt\_5\_5\_1-Release\android-build\bin \QtApp-debug-unaligned.apk

该apk文件可以通过adb工具远程安装到android设备中，安装命令简介：

1. 显示当前运行的全部模拟器 adb devices

1. 连接设备 adb connect 192.168.1.101(设备IP在设置里面可以看到)

2. 卸载软件 adb uninstall org.qtproject.example(这是触摸屏的包名)

3. 安装软件 adb install QtApp-debug-unaligned.apk(绝对路径或相对路径，最简单的方式是直接拖拉到windos终端上)

为使该APP能上电自启动，必须手动设置启动桌面为该APP。

* 项目分析

RoadMap 这个类很重要，它显示QML页面，同时负责与信号机进行交互。这里也用到QT最基础的信号与槽的概念。其中的SendUdpRequest方法，是被定时器500ms调用一次，用来获取信号机的实时状态，SendSpecialCtrlCmd方法，是在页面中相应按钮被按下后自动触发向信号机发送相关控制指令。GetUdpResponse方法，总的消息接收回调函数，根据消息ID，更新页面指定部分。如果要深入了解c++与QML的交互方法，建议查看QT 手册。

\*.qml QML这种解释型语言和JS很像，初次接触可能感觉很难。建议先看中文教程再看QT 手册，了解QML 和QT Quick提供的基础控件。从整体上认识一个qml程序的框架，再了解常见控件提供的方法。

* 注意点

1. 考虑到新库变动较为频繁，在使用前，先测试一下功能是否满足。

2. 如果需要集成晓东的手机APP到触摸屏上，需要参考android中activity之间的调用方法。

* 相关资料

1. QT相关资料见上个项目。

2. 《Beginning\_Android\_4\_Application\_Development》

3. 《Android 4高级编程(第3版)》

四、内外网转发工具

* 基本描述

该工具做的比较粗糙。如果后期有类似项目需要大规模使用，建议重新开发。该工具基于MFC主要用来实现公网环境下与信号机进行交互。具体思路是

1. 转发工具根据指定配置，绑定监听端口。

2. 前端信号机通过TCP连接到转发工具，工具根据ID，创建TCP通讯线程，将接收到的数据通过UDP转发到指定的IP:PORT中。

3. 工具接收到UDP信息后，直接通过TCP转发到指定的前端设备。

* SVN路径

https://192.0.0.75/TrafficSignalController/ARM/branches/工具类软件/信号机内外网转发程序/hikTSC/hikTSC

* 工程目录

hikTSCDlg.cpp 在这一个文件中实现所有功能。

* 部署方案

将软件和config.ini放置在同一个文件夹中，点击运行即可。配置文件的格式如下：

[TransferTool]

IP=10.194.229.170 中心平台的IP地址

MachineNum=40 需要开启转发的信号机个数

PortNO=9014 起始端口号，在平台上进行配置信号机通讯口需要用到，依次递增。

DisplayID=6 这个主要是便于调试，ID是几就把第几个信号机的通讯打印出来，能看出该信号机的网络连接效果。

IsAutoRun=1 是否自启动，如果开启自启动，则打开软件即运行。主要是用来供用户实现开机工具自启动。

* 项目分析

OnBnClickedButtonStart 启动按钮回调函数，在进行数据合法性校验后，开启服务。

DoService 启动或停止转发的接口函数。

SetOnlineMachine 动态展示信号机的在线情况，如果接收失败就认为信号机离线。

ServiceTCPConnected 负责和已接收已TCP连接的信号机发来的消息，消息通过UDP重新转发给中心平台。

CreateVirtualTSC 创建模拟信号机线程(函数名在这里不恰当，是模拟信号机移植过来的)。一个信号机应该至少包括161(ServiceBasic中处理)和20000(ServiceCountdown 中处理)端口的数据。

ServiceBasic 接收发给161端口的udp数据，并通过tcp转发到前端设备

ServiceCountdown 接收发给20000端口的udp数据，并通过tcp转发到前端设备

* 注意点

1. 这里每次数据转发都单独启用一个线程，如果数据量较大，不建议再采用。

* 相关资料

无

五、基线2.0模拟信号机软件

* 基本描述

该软件运行在x86架构的linux系统中，目前在ubuntu中运行OK。该软件主要是供平台组同事进行多台信号机模拟控制,完善并发功能.但同时也可以帮助我们研发在手头没有信号机时，在虚拟机上运行起来程序，便于逻辑调试。

* SVN路径

https://192.0.0.75/TrafficSignalController/ARM/branches/工具类软件/新库模拟信号机程序/new

* 工程目录

与2.0基线库相同，只是编译成x86即可。

* 部署方案

1. 将hikTSC300拷贝到linux虚拟机中的/root目录下，hikconfig.dat拷贝到/home/目录下。

2. 在终端中依次执行命令:

chmod 777 hikTSC300

./hikTSC300 &

3. 正常运行后，默认会开启100台信号机，信号机的IP地址就是你虚拟机eth0的IP地址，端口范围

是[20000,20100].

【目前不支持SADP，无法使用在线搜索功能，信号机配置断电不保存】

* 项目分析

绝大部分都是2.0库的源码，改动部分只涉及以下几个方面：

1. 根据需要模拟的信号机个数，创建N个进程，每个子进程里面只需要运行InitCommon函数即可，当然该函数不能return掉。

2. 每个进程单独和外部工具进行通讯，所以每个进程需要绑定不同的端口(不能都是20000)。

3. 每个进程间不需要通信，每个进程内各个模块的通信(消息队列)必须采用不同的标识。

* 注意点

1. 目前不提供外部接口来控制模拟信号机的运行个数，且当时移植时基于1277这个版本，如果平台需要测试新功能，建议将最新库重新移植过来。

2. 该程序运行在pc机上，各个源文件夹内均可以通过make ARCH=pc来切换编译器，目前做的很粗糙，没有统一make，所以，如果需要编译，请单独在Libs libits hikconfig tsc300 tsc500里面使用make ARCH=pc来编译。

* 相关资料

参考进程的创建方法。

六、经济型信号机灯控板及前面板MCU程序

* 基本描述

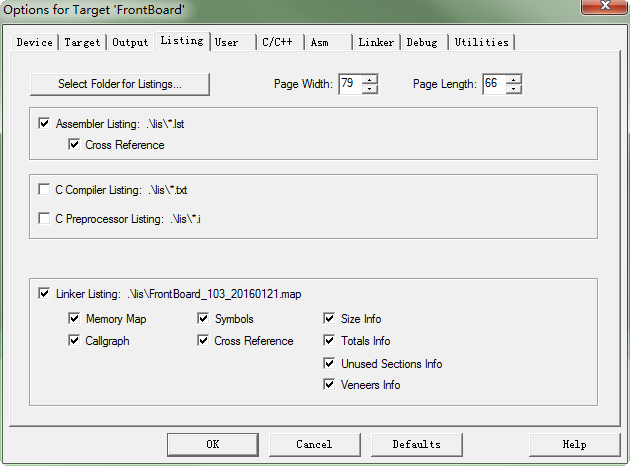
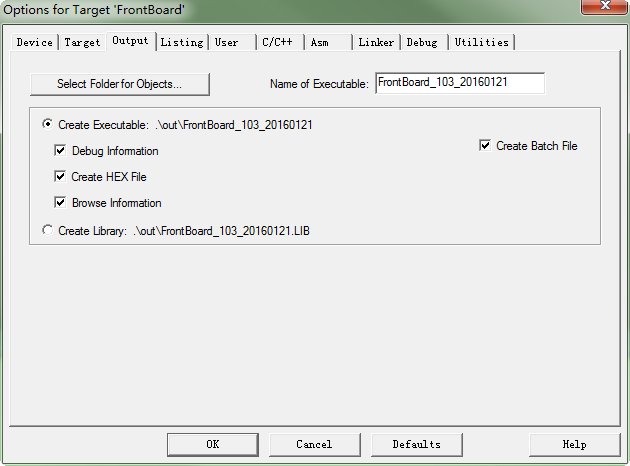
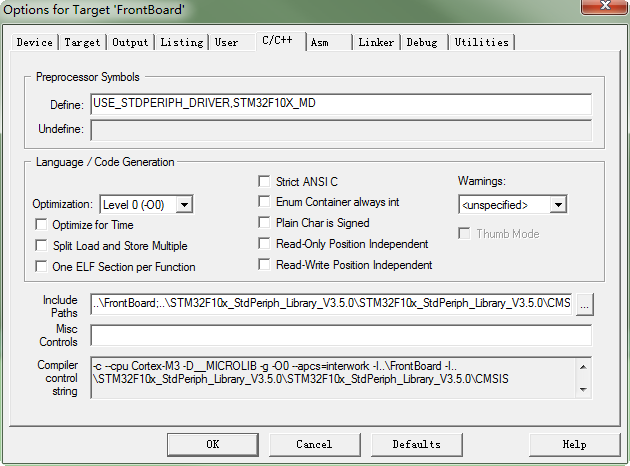
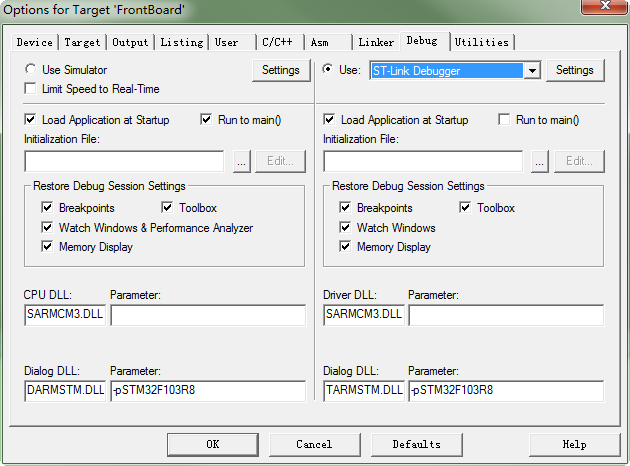
经济型信号机采用STM32F103作为MCU，其中灯控板主要完成电流电压采集、接收主控程序CAN消息点亮对应通道、发送消息给前面板电流对应路口及相位接管等功能。主要用到的知识有CAN、串口、中断、DMA、ADC和基本的GPIO的读写。基本思路就是初始化时钟->各个硬件模块->中断接收消息->相位接管等。

* SVN路径

https://192.0.0.75/TrafficSignalController/ARM/branches/经济型信号机

* 工程目录

直接把这些src inc放置到STM32F10x\_StdPeriph\_Library\_V3.5.0的模板目录即可，我们使用keil uVision4来进行编译，烧写flash使用ST-LINK。只需要在项目选项中选择指定的STM32F103R8这个MCU即可，其余的选项配置如下：

* 部署方案

编译完成后直接通过STM32 ST-LINK Utility进行flash烧写即可。如需观察软件运行信息，可以接串口，波特率设置115200。

* 项目分析

Main.c 完成了主要功能，其中HAL\_Init SystemClock\_Config\_HSE完成了时钟初始化，

InitGPIOA InitGPIOB InitGPIOC Cfg\_Uart Cfg\_Can Cfg\_ADC Cfg\_IWDG分别完成了GPIO、串口、CAN、ADC、看门狗的硬件初始化。紧接着获取灯控板的基地址，每个灯控板都由背板唯一确定了一个地址，分别是0 1 2 3，后期规划的4 5两个地址是通过flash写死的，当然我和汪发祥已经测试通过AD采样值给4 5两个板子分配地址，具体方法是在GetNewBaseAddr里面while循环里面有样例，后期可以参考。

CFG\_Can CAN的初始化部分，要着重看951-954行，不是说波特率设置相同就可以的，具体可以参考官网上关于CAN设置的文档，找到最优的配置。

CAN\_RX1\_IRQHandler\_def

这里是CAN的回调函数，这里接收来自主控板的消息，点灯同时将信息发送给前面板。电流值是自动采集的，电压值需要每次读取。其中电流值进行了一次过滤，偏差过大的数据都会被丢弃掉。这里收到消息的话，CAN的收指示灯会闪烁，如果实际情况该灯不闪烁了，则可断定主控程序挂掉了。

CAN\_TX1\_SendData

这里采用直接发送，不再中断发送。

HAL\_UART\_RxCpltCallback

串口的回调函数，里面实现了诸如重启系统、擦除flash、打印配置、配置flash地址等功能，可以使用Serial Port Utility这个工具来发送串口数据。

* 注意点

无

* 相关资料

经济型信号机12V电源供电及STLINK接线方法

经济型信号机103灯控板串口使用说明

经济型信号机扩展灯控板配置方法

经济型信号机扩展灯控板升级及配置方法

经济型信号机软件升级方法—STLINK

路口型指示灯板GPIO管脚定义

主控板GPIO管脚定义

七、基线1.0倒计时、单时段通道锁定和配置管理模块以及自动构建升级包

* 基本描述

1. 老库倒计时模块，主要是为了将实时相位状态、通道状态、方案状态等内容传递给平台，供其展示周期进度条。因老库源码不可见，且只提供了机动车和行人相位倒计时，故需要我们自己计算PHASE\_COUNTING\_DOWN\_FEEDBACK\_PARAMS结构体中的所有值。难点在于计算各个相位的绿信比及运行时间，采用的方法是相序表中前后相位做差得到绿信比时间。具体参见CalcPhaseRunTimeAndSplit函数，这里会把行人、机动车、跟随相位的倒计时都算出来，这样就可以供倒计时器使用了。【考虑到1.0库以后不再维护，此功能可能会被废弃掉，有需要则浏览即可，若平台上1.0信号机进度条紊乱，要考虑配置信息是否有误】

2. 单时段通道锁定主要供平台做特勤路线用的，通道锁定比较简单粗暴，直接在PhaseLampOutput修改灯控can消息，目前单时段已被多时段代替掉，具体可以参考潘文俊的代码。

3. 前期我们的配置是直接以windows下ini格式文本存储的，数据容易丢失，故而采用以二进制格式存储。考虑到二进制格式，可读性太差，又做了一个转换工具，用来方便研发人员查看及修改特定配置。目前，新增配置的大致流程是：在configureManagement.h文件中定义结构体->在inifile.c中添加读写接口->如果是新的dat文件需要在configureManagement.c中按照格式添加读取二进制的接口。目前转换工具BinaryTextConvert.c已实现了对hikconfig config desc custom countdown misc vehicle FailureLog等八种配置的转换，转换方法可以通过/root/BinaryTextConvert -h查看。如果需要新增配置，则需要在全局指针数组中gArraySupportCfg增加一项，同时修改DoConvertFromBin2Text和DoConvertFromText2Bin两个转换接口。【日后改成sqlite存储时，注意读写时采用事务处理，不要一条一条添加或修改，太耗时间。另外建议将sqlite工具移植到板子上，方便研发同事在不拷贝db文件的情况下，直接在信号机终端中查看或修改指定表的内容。Sqlite的操作可以参考webapp中故障的存储和早期2.0库对文件监控备份的处理，建表规范可以借鉴飞哥和我的那个配置工具。】

4. 以前给现场的版本很多，导致一出现问题只能根据编译时间来查，为统一管理向外发出的升级包，专门做一个update模块，用来自动构建升级包。主要脚本内容在autoMake.sh中，自动构建的升级包会使用linux 下svn 命令，将最近的svn日志和编译时间添加到各自升级包的VERSION.txt中，同时会自动编译最新的300和500系列应用程序。建议日后向现场发升级包仍从update中拉最新的，各位提交大的改动时，也务必更新升级包。

* SVN路径

https://192.0.0.75/TrafficSignalController/ARM/trunk/baselineV1.0/common

* 工程目录

参见目前1.0基线的目录，通用功能在common里面。

* 部署方案

无

* 项目分析

1. 假设当前方案中有N个相位，那么CountDownOutVeh函数1s会被库调用N次，即每个相位调用一次，参数就是相位号和相位的状态和值，需要注意的是状态只有红黄绿三种，没有绿闪等。该函数会把相位的状态和倒计时存储在gCountDownParams中供倒计时模块来计算其他信息。CollectCountDownParams这个是倒计时接口的总入口函数，该函数每个周期开始会计算一次当前运行的相位表、跟随相位表、相序表和方案信息。如果程序是在前台运行，那么可以通过执行sh /root/sig.sh或kill -10 hikTSC的进程ID看到每秒钟各个相位的倒计时信息。CalcPhaseRunTimeAndSplit接口实现了倒计时接口中的几乎所有功能。需要注意的是，老库提供的倒计时信息可能并不完全准确，我们需要剔除不合理的数据，比如在非感应控制时倒计时时间增加的情况。在感应控制时，异常情况下可能会出现相位倒计时不同步增加固定值的情况，那么我们需要自己将不合理的数据增加指定值使其合理。考虑到老库并不提供绿闪这个状态，我们需要自己根据全局的配置参数来将相位状态修改成绿闪状态。

* 注意点

1. 如果发现2.0配置工具或者平台出现周期运行紊乱的情况，首先排查hikconfig.dat数据是否有误，若校验通过，则在前台运行程序查看各个相位倒计时是否计算正确，一般情况下都是配置文件有误导致的。

2. 以后的转换工具以2.0基线里面的为准，升级包也会使用这里的。

3. 自动构建升级包目前并没有移植到2.0基线中，故2.0的程序需要手动拷贝到升级包。

* 相关资料

无

八、基线2.0配置校验、倒计时器、流量统计、SADP

* 基本描述

1. 为确保信号机接收到的配置信息能够正确运行，减少出现黄闪的概率，我们对配置进行了校验。校验的原则是，凡是被引用到的表都要被校验，没有被引用的表不需要关注。校验首先需要保证，各个表是以数组下标作为ID号的，然后的每次校验基本都是看被引用到的表是否存在。校验模块，里面的注释比较清楚，其中gErrorContent全局指针会被web 服务器引用，将中文提示信息展示到页面上。2.0配置工具使用的枚举量TSC\_Para\_Error\_Num，错误描述如其后注释所述。Hikconfig是针对NTCIP的，gbconfig ykconfig分别是针对国标和优控协议的，后面的两种目前没有做校验。目前前端配置工具只做简单校验,信号机本身会在配置接收结束后,统一校验,只有校验合法后,才使配置更新.

2. 倒计时器模块，主要是根据全程脉冲、半程脉冲、莱斯、国标和海信协议，通过一定规则发送消息给倒计时器，使倒计时器能正确反应出各个通道红绿灯的倒计时情况。1.0库里的倒计时器依然是映射相位的，而2.0里面的倒计时器和车检器已经修改为关联通道。其中在custom.dat中有一个倒计时时间是否受感应检测时间限制的标志cIsCountdownValueLimit,如果该值为1,则在协议倒计时下,倒计时值只有小于等于感应检测时间时才显示,该值默认为0. 一个倒计时器可以不止一个相位(通道)，显示是规则是最大值的绿灯 > 最大值的黄灯 > 最小值的红灯 。如果都不满足要求，就发送倒计时值为0的关灯信息。

3. 流量统计，主要是供配置工具或平台查看实时或历史流量信息，以进行方案优化。目前每个采集周期内，统计流量数并根据相关公式估算出时间占有率、平均车速、排队长度、绿损、车流密度、车头间距和车头时距等信息。实时流量通过udp通讯获取，历史流量上层应用会直接通过ftp读取/home/ vehicle.dat进行解析。这个文件目前暂定是10M，循环覆盖。

4. SADP，就是信号机程序里面的RTSC\_sadp程序，老版的SADP无法在WLAN中使用，新版的SADP兼容老的SADP同时可以使信号机在WLAN下被最新的SADP工具搜索到。新的协议见 SADP多播概要设计说明书.doc ,目前我们只实现了其中的主动上线和搜索请求，其他的需求待有需要时，可以参照实现。从2016年2月起，升级包中的SADP

已支持WLAN状态下，搜索到信号机。

* SVN路径

<https://192.0.0.75/TrafficSignalController/ARM/trunk/baselineV2.0>

<https://192.0.0.75/TrafficSignalController/ARM/trunk/sadp>

* 工程目录

1. 配置校验主要是在trunk\baselineV2.0\hik\hikconfig.c中完成的。

2. 倒计时器主要是在trunk\baselineV2.0\hik\common\countdown.c中完成的。

3. 流量统计主要是在trunk\baselineV2.0\hik\common\ datacollect.c中完成的。

* 部署方案

无

* 项目分析

1. 配置校验，IsSignalControlparaLegal接口是总入口函数，返回0表示校验通过，返回其他值表明校验失败，失败错误码在TSC\_Para\_Error\_Num中定义。在入口函数中，排查参数为空后，首先需要校验的就是数组下标必须合法，这是后面校验的基础。紧接着根据“使用则校验，未用则不校验”的原则，从调度表开始一层一层校验，直到校验完成。注意，我们校验中目前没有校验车检器表。通过校验，不代表信号机一定能正常运行。

2. 倒计时器模块，CountDownInterface是总的入口函数，这个函数每秒钟至少要被调用一次。倒计时器目前的配置比较乱，倒计时器的总体配置在custom.dat中，主要用来配置倒计时的类型、红绿脉冲倒计时时间及协议倒计时倒计时值是否受限标识。倒计时器的具体配置在countdown.dat中，主要完成倒计时器与通道/相位的对应关系及脉冲倒计时器的通道标识。思路是每次都把倒计时的状态和值保存在全局参数中，各个协议会分别调用SetCountdownValue参数将合适的倒计时值复制到各个倒计时器的参数中，再以规定的格式封装发送485串口数据即可。导致协议倒计时器的话，波特率最好按照9600或更高来设，防止因数据过大影响发送效率。

3. 流量统计，参见《交通工程学》第四章道路交通流理论。

4. SADP只是增加了组播功能，其他功能待添加。

* 注意点

无

* 相关资料

