APRUS Lua-DLT645 配置说明

第一章 概述

1.1 文档说明

文档概述 APRUS Lua-DLT645 协议相关配置说明，APRUS 的 Lua 包含 APRUS.lua 和 config.lua 两个文件。APRUS 通过 Dlt645 协议与电表通信，然后配置 config.lua 即可采集需要的数据。如需修改 APRUS.lua 文件的内容时，请咨询相关的技术人员，随意修改会导致适配器不能正常工作，所以此文档主要介绍 config.lua 内容。

此文档目的是为了让相关人员清楚如何通过对脚本的配置实现对应协议的数据采集，详细介绍 config.lua 的相关内容，如遇到新的需求或者处理不当的地方联系相关负责人。

第二章 配置说明

3.1 config.lua 配置

config.Lua 由 5 大部分组成，ipmode、inet_addr、netmask、Luaver、devinfo；分别来介绍这 5 大部分的功能：

示例：

```
APRUS X={
    ipmode="Manual",
    inet_addr="192.168.1.234",
    netmask="255.255.255.0",
    Luaver="MAX.Lua.V030300.R",
    devinfo="DLT645Dev",
},
```

说明：

序号	模块	说明	备注
1	ipmode	IP 获取方式	默认 Manual，不更改
2	Inet_addr	APRUS IP	与对接的设备保持相同网段，且同网段内 IP 不能重复

3	netmask	子网掩码	默认 255.255.255.0，一般不修改，除非相同网段内没有足够的 IP
4	Luaver	脚本版本号	每次变更脚本都需更新版本，一边区分不同版本的区别
5	devinfo	设备类型	对接的设备类型

3.1.1 Ipmode:ip 获取方式

ipmode="Manual"。

ipmode 为 APRUS 的 ip 获取方式，默认为 Manual，手动模式，人为填写 APRUS 的 ip。

3.1.2 Inet_addr:APRUS Ip

Inet_addr="192.168.1.234"。

Inet_addr 为 APRUS 的 ip，手动填写 ip，需要同对接设备保持在相同的网段内，否则无法与对接设备通过网口通信。

3.1.3 netmask:子网掩码

netmask="255.255.255.0",

netmask 为子网掩码，默认为 255.255.255.0，与客户保持一直，一般情况下均为 255.255.255.0 不变。

3.1.4 devinfo:设备类型

devinfo= "DLT645Dev"。

devinfo 设置设备类型，由于对接的设备类型不同，因此为了方便使用，知道对接的是哪类设备，我们设定了设备类型。

3.1.5 Luaver:脚本版本

Luaver = "MAPRUS.L.V030005.R"。

Luaver 设置当前脚本的版本号，每次更新都需要更改并向上增加版本号；版本号在增加过程中，只需要修改主/子/阶段版本号即可，其他无需修改，主/子/阶段版本号不能为 0，一般情况下如果只更新 config.Lua 的配置无需更新主/子版本号，只要修改阶段版本号就行，版本号一定是要向上叠加，这样才能保证之前的版本有备份，才能自动升级。

3.1.6 DLT645:interface: DLT645 设备信息

示例:

```
Device={
        rate=2400,
        databit=8,
        stopbit=1,
        parity="Even",      -- None/Odd/Even
    },
```

DLT645 设备接口信息:

序号	模块	说明
1	rate=2400	波特率
2	databit=8	数据位
3	stopbit=1	停止位
4	Parity="Even"	校验位

3.1.7 Node 节点信息

```
node={
    {addr="1",dflags="A010",dtype="bcd-num",pMode={1,5},dStyle={"L1_A010"},Offset={{"/",10000}}},
    {addr="1",  dflags="04000102",  dtype="bcd-str",  pMode={1,5},dStyle={"L1_04000102"}},
    {addr="1",dflags="01010000",dtype="other",format="20xxxx-xxxx:xxxxxx.xx",pMode={1,5},dStyle={"L1_01010001"}},
}
```

1. 节点配置说明

采集及上报配置，主要有几个参数构成。

配置	参数	说明	备注
Node	addr	dlt 电表表号，地址	
	dflags	数据标识	97 版本：4 字符 07 版本：8 字符
	dtype	数据类型	bcd-num: 按 bcd 数字类型解析，用于整形类数据 bcd-str: 按 bcd 字符串类型解析，用于日期等类型数据 other: 用于特殊类型指定，需搭配 format 参数使用 例如：数据标识 01010000 返回的数据内容同时包含日期和功率值，则可以指定 format="20xx xx-xx xx:xx xxxx.xx" 会输出"2020 04-23 15:48 13.98"
	format	特定格式	当 dtype 为 other 时有效 1 个 X 表示一个位有效数
	author	操作者	用于反向写入数据时使用 (只读取可不使用)
	pwd	密码	用于反向写入数据时使用 (只读取可不使用)
	pMode	上报方式	详见附件一
	dOffset	数据偏移	详见附件二
	dStyle	上报名称	自行设置上报名称
	dExt	条件策略	(选填)改变上报条件策略， 当 pMode 为{2,0}时有效，{">", 100} 表示 当原始数据发生改变 并且改变值大于 100 时上报，{"<", 100} 表示 当原始数据发生改变 并且改变值小于 100 时上报，{"=", 100} 表示 当原始数据发生改变 并且改变值等于 100 时上报

附件一

模块		举例	参数 1	参数 2
			上报模式	上报周期
pMode	上报类型	pMode={1,5}	1：周期上报	自行设置上报周期
			2：改变上报	0

附件二

模块		参数 1		参数 2		备注
		偏移参数 1		偏移参数 2		
dOffset	dOffset={{"+",10},{"*",5}}	"+"	N	"+"	N	
		"-"	N	"-"	N	
		"*"	N	"*"	N	
		"/"	N	"/"	N	
		"."	N	"."	N	保留 N 位小数

注意：

1. 如果无偏移计算，则无须填写此字段，或者填写为 dOffset={}。

2. 如果只有单层偏移，那么只需要设置参数 1 即可，假如一个参数需要*10，即 dOffset={{"*",10}}。

3. 如果有双层偏移，那么需要设置参数 1，参数 2，加入一个参数需要*10，然后在+10，即 dOffset={{"*",10},{"+",10}}。

4. 如果上报数据包含小数且需要保留指定的小数位数时，即可对参数 1 进行设置，假如需要保留 3 位小数，即 dOffset={{".",3}}。

3.2 apurs.lua 配置

说明：function start() 为主流程入口函数

3.2.1 user 全局通用/配置类

1)本机网络配置方法

user.ipconfig(config.APRUSX.ipmode,config.APRUSX.inet_addr,config.APRUSX.netmask)

2)消息捕获方法

msg = user.waitmsg()

msg.from：消息来源

mqtt-sys　来自 mqtt 系统消息

mqtt-msg　来自 mqtt 数据消息

dlt645　来自 dlt645 管理器的消息

3)以下为 msg.from 为 modbus、opcua、mitsufx、mitsumc、dlt645、s7 等协议时通用

msg.session：消息会话对象

msg.code：消息号

msg.style_L：变量名称

msg.val_L：变量值

msg.style_E：事件变量名称

msg.val_E：事件变量值

msg.z：条件改变标志位

4)以下为 msg.from 为 mqtt 时使用

msg.topic：mqtt 话题

msg.payload：mqtt 消息数据

5)以下为 msg.from 为 mqtt-sys 时使用

msg.code：mqtt 事件

0：　mqtt 断开

1：　mqtt 连接

6)设置 Lua 版本信息

user.setLuaver(Luaver)

7)设置设备信息

user.setdevinfo(devinfo)

3.2.2 mqtt 对象方法

(1)创建 mqtt 对象实例

mqttobj = mqtt.new()

(2)配置 mqtt 连接信息 (选填)

mqtt.config(mqttobj, "mqtt_1", "192.168.1.159", "1883")

param 1 mqtt 对象实例

param 2 实例 id

param 3 mqtt 服务器 ip

param 4 mqtt 服务器端口

注：此配置选填,如果不填,则通过 gards 服务器自动指定

(3)话题订阅

mqtt.subscribe(mqttobj, "p2p")

(4)发布消息

mqtt. publish (mqttobj, reserve, topic, payload)

reserve: 预留参数,必须有,可填 nil

(5)mqtt 运行

mqtt.run(mqttobj)

备注：支持多实例，最大可创建 3 个 mqtt 对象

3.2.3 DLT645 对象方法

(1) 创建 DLT645 对象实例

dlt645obj = dlt645.new()

(2) DLT645 对象配置

dlt645.config(dlt645obj, rate, databit, stopbit, parity)

param 1 DLT645 实例

param 2　波特率

param 3　数据位

param 4　停止位

param 5　校验位

(3) DLT645 添加采集节点

dlt645.add_collectnode(dlt645obj, addr, flags)

param 1　DLT645 实例

param 2　电表地址

param 3　数据标识

(4) DLT645 添加变量节点

dlt645.add_varnode(dlt645obj, addr, dflags,author,pwd,dtype,format,pMode, dStyle, dOffset, dExt)

param 1　DLT645 实例

param 2　电表地址

param 3　数据标识

param 4　操作者

param 5 密码

param 6 数据类型

param 7 采集上报模式

param 8 采集上报名称

param 9 数据补偿计算

param 10 改变上报条件限制

(5)DLT645 向被指定节点写入数据

dlt645.write(dlt645obj, style, val)

param 1 DLT645 实例

param 2 变量名称

param 3 变量值

(6)DLT645 实例启动

dlt645.run(dlt645obj)

(7)DLT645 实例停止

dlt645.stop(dlt645obj)