

IGT-SER/CLI 系列智能网关边缘计算

LUA 脚本用户手册

V1.63



武汉埃和智能科技有限公司
www.aihe-tech.com

Copyright2016 武汉埃和智能科技有限公司及其许可者版权所有，保留一切权利。

未经本公司书面许可，任何单位和个人不得擅自摘抄、复制本书内容的部分或全部，并不得以任何形式传播。

本手册中出现的商标、产品标识及商品名称，由各自权利人拥有。

由于产品版本升级或其他原因，本手册内容有可能变更。武汉埃和智能科技有限公司保留在没有任何通知或者提示的情况下,对本手册的内容进行修改的权利。

本手册仅作为使用指导，武汉埃和智能科技有限公司尽全力在本手册中提供准确的信息，但是武汉埃和智能科技有限公司并不确保手册内容完全没有错误。本手册中的所有陈述、信息和建议也不构成任何明示或暗示的担保。

前 言

《IGT-SER/CLI 系列智能网关 LUA 脚本用户手册》介绍了 IGT-SER/CLI 系列 PLC 通讯智能网关脚本语言的开发和使用方法，包含原理的介绍、具体的操作描述和应用配置举例。

读者对象：

本手册主要适用于如下人员：

- 产品使用者
- 现场技术支持、维护人员和系统实施人员

本手册约定：

[]：带尖括号 “[]” 表示按钮名，如“单击[确定]按钮”。

<>：带方括号 “<>” 表示窗口名、参数名和数据名，如“在<网口 1. IP 地址>设置网口 1 的 IP 地址”。

/：多级菜单用 “/” 隔开，如[功能]/[类型与设备选择]。

n：可变的数字，如“TXn”，可以代表某一个串口。

技术支持：

用户支持邮箱：aihe_tech@126.com

技术支持电话：159 721 99489

网址：http://www.aihe-tech.com

资料意见反馈：

如果您在使用过程中发现产品资料的任何问题，可以通过以下方式反馈：

E-mail：aihe_tech@126.com

感谢您的反馈，让我们做得更好！

目 录

1.0 LUA 脚本简述.....	4
2.0 脚本程序应用.....	5
3.0 功能函数说明.....	6
3.1 调试与脚本控制函数.....	6
3.2 脚本读写配置表的变量.....	7
3.3 数据与字符编码转换.....	8
3.4 串口实现非标协议通讯.....	9
3.5 以太网 TCP/UDP 通讯.....	10
3.6 WebService 与 HTTP 相关.....	11
3.7 MQTT 与 JSON 文件相关.....	12
3.8 FTP 文件上传与下载.....	13
3.9 数据库 SQL 上报与查询.....	14

1.0 LUA 脚本简述

IGT 智能网关采用 Lua 语言作为脚本编程开发的语言，配合网关内已经封装好的功能函数，为网关应用提供灵活的扩展和客制化功能，快速实现边缘计算。

以下是脚本与网关配置的功能结构图：



关于 Lua 脚本语言的介绍及语法规则、教程等，可通过以下网址获取：

<https://www.runoob.com/lua/lua-tutorial.html>

可根据编程习惯找合适的 Lua 脚本语言的编辑器，也可以通过本公司网站下载 LuaEditor 软件：<http://www.aihe-tech.com/col.jsp?id=107>

网关脚本常用的功能包括 HTTP 收/发、JSON 解析、数据上报对接、非标设备通讯、MQTT 发布/订阅、网络校时等，这些功能都有大量的实际应用案例，可联系技术支持获取，参照修改脚本代码。

2.0 脚本程序应用

2.1 脚本程序文件命名

IGT 智能网关接受按以下规则命名的脚本文件：

Script_*.lua

文件名中星号位置可自定义，支持中文、英文和数字。

2.2 脚本程序下载到网关

通过 IGT 智能网关的参数管理软件 Ah_igt_pm.exe 将编辑好的脚本程序下载到 IGT 智能网关内。

方法 1：[参数]/[脚本程序写入网关]

方法 2：[帮助]/[网关文件更新]

脚本程序文件下载到网关后，需要重新启动网关模块才会执行，也可通过参数管理软件的[工具]/[重启网关]操作。

2.3 脚本程序运行调试

脚本程序下载到 IGT 智能网并重启网关后，脚本程序会自动运行，有以下两种方式调试和跟踪脚本程序的运行状况：

一、参数管理软件的数据配置表增加脚本程序调试变量，脚本程序内根据程序过程断点，用写标签数据的功能函数<tag_writevalue>，向这个调试变量发送过程数据或信息。

二、过脚本程序的日志输出函数<fun_udplog>，通过 UDP 方式输出日志，须要在脚本程序内根据合适的过程断点调用该方法，设置想要输出的数据或信息，同时须要有 UDP 接收端接收查看日志。

3.0 功能函数说明

3.1 调试与脚本控制函数

3.1.1 脚本日志输出

通过 UDP 的方式发送数据或信息。

`fun_udplog(string:接收端 IP 地址, int:接收端端口, string:日志消息)`

正常返回: 0

错误返回: -1

3.1.2 是否允许多次运行

如果脚本程序内有错误, 会终止执行, 这个函数可设置是否允许重新运行。

`fun_ackcycle(int: 0=不允许, 1=允许)`

3.1.3 延时功能函数

延时, 单位毫秒。

`fun_sleep(int: 时间 *1ms)`

3.1.4 查询网关与设备的通讯状态

查询网关与 PLC、仪表等所连接设备的通讯状态

`fun_devstate()`

正常返回: int: 0=正常通讯, 其它值表示通讯异常

错误返回: nil

3.1.5 网关模块系统重启

重启网关模块, 网关不会蜂鸣报警

`fun_restart()`

3.1.6 查询网关与服务端的工作状态

查询网关与数据库服务器、MQTT 代理、HTTP 服务端的工作状态

`fun_serstate()`

正常返回: int: 0=正常通讯, 其它值表示状态异常

错误返回: nil

3.2 脚本读写配置表的变量

3.2.1 变量当前值读取

读取网关变量配置表内变量的当前值。

tag_readvalue(int: IP/站号/组, string: 标签/字段名称)

正常返回: string: 读取到的数据

错误返回: nil

3.2.2 变量当前值写入

将数据写入到网关变量配置表内的变量。

tag_writevalue(int: IP/站号/组, string: 标签/字段名称, string:数据)

正常返回: int: 1

错误返回: nil

3.2.3 变量当前值读取(字节数组方式返回)

字节数组方式读取网关变量配置表内变量的当前值。

tag_readchar(int: IP/站号/组, string: 标签/字段名称)

正常返回: byte 数组: byte[1],byte[2],byte[3] ...

错误返回: nil

3.2.4 其它 IGT 网关模块内的变量当前值读取

读取另外的 IGT 网关变量配置表内变量的当前值。

igt_tagread(string: IPV4 地址, int: 标签/字段序号)

正常返回: string: 读取到的数据

错误返回: nil

3.2.5 读取网关的系统时间

格式: 年-月-日-时-分-秒-毫秒, 17 位数字, 如: “20201224100826284”

fun_timenow()

正常返回: string: 读取到的时间数据

错误返回: nil

3.2.6 读取网关的时间戳

格式: 毫秒单位, 13 位数字, 如: “1608775706284”

fun_timestamp()

正常返回: string: 读取到的时间戳数据

错误返回: nil

3.2.7 校正网关时间

fun_calibtime(int: 10 位数字的时间戳)

正常返回: (0:正常, 1:错误)

错误返回: nil

3.2.8 批量读取变量名称和数值

返回两个逗号分隔的字符串, 分别是变量名和数值。

tag_readbatch(int: IP/站号/组, int: 模式)

正常返回: (string: 变量名, string: 数值, int: 变量数量)

错误返回: nil

模式=0: 所有变量; 模式=1: 数值来自 PLC 的变量; 模式=2: 数值要写入 PLC 的变量;

3.3 数据与字符编码转换

3.3.1 16 位 BCD 码转为 BIN 数据

fun_bcdtobin16(int: BCD 码)

正常返回: string: 转换后的十进制的数值

错误返回: nil

3.3.2 16 位 BIN 数据转为 BCD 码

fun_bintobcd16 (int: BIN 数据)

正常返回: string: 转换后的 BCD 码

错误返回: nil

3.3.3 字符串格式与长度设置

fun_stringchar (int: IP/站号/组, string: 标签/字段名称, int: 字符串格式, int: 缓存区域的最大字节数)

正常返回: int: 0=正常, <0=错误

错误返回: nil

字符串格式: 0=默认, 1=交换 1 个字符, 2=交换 2 个字符, 3=UTF8 与 ASCII 互转,

4=UTF8 与 WChar 互转

缓存区域的最大字节数:1~64 个字节

3.3.5 Base64 编码

fun_base64encode (string: 源文件缓存, int: 源文件长度)

正常返回: (string: 编码后的文件缓存, int: 编码后的文件长度)

错误返回: nil

3.3.6 解码 Base64

fun_decodebase64 (string: 源文件缓存)

正常返回: string: 解码后的文件缓存, int: 解码后的文件长度

错误返回: nil

3.4 串口实现非标协议通讯

3.4.1 串口设备通讯

通过网关串口与所连接的设备进行通讯, 发送后接收数据。

com_sendrecv(int: 串口号(1/2), int:发送字节数, 发送数据: byte[1], byte[2], byte[3] ...)

正常返回: (int: 串口号(1/2), int:接收字节数, 返回数据缓存: byte[1], byte[2], byte[3] ...)

错误返回: nil

3.4.2 串口设备通讯 2

通过网关串口与所连接的设备进行通讯, 可以不发送只接收数据。

com_sendrecv2(int: 串口号(1/2), int:发送字节数, int:接收字节数, 发送数据: byte-1, byte-2, byte-n ...)

正常返回: (int: 串口号(1/2), int:接收字节数, int:返回字节数, 返回数据: byte-1, byte-2, byte-n ...)

错误返回: nil

3.4.3 ModbusRTU CRC16 校验计算

fun_rtucrc16(string: 需要校验的数据缓存)

正常返回: (int: CRC 结果低字节数值, int: CRC 结果高字节数值)

错误返回: nil

3.4.4 读取 USB 口/串口条码字符串

com_getstring()

正常返回: (string: 通讯端口已读取到的条码字符串)

错误返回: nil

3.5 以太网 TCP/UDP 通讯

3.5.1 TCP 设备通讯

通过网关的网口进行通讯, TCP 发送后接收数据。

tcp_sendrecv(string:服务端地址, int:服务端端口, int:通讯时间限制, string:发送数据, int:发送数据长度)

正常返回: string:接收的数据

异常返回: nil:未接收到数据, 或者出错

通讯时间限制设 0: 只发送不接收

3.5.2 UDP 设备通讯

通过网关的网口进行通讯, UDP 发送后接收数据。

udp_sendrecv(string:服务端地址, int:网络端口, int:通讯时间限制, string:发送数据, int:发送数据长度)

正常返回: string:接收的数据

异常返回: nil:未接收到数据, 或者出错

通讯时间限制设 0: 只发送不接收; 发送数据长度设 0: 只接收不发送

3.5.3 ModbusTCP 协议数据发送

通过网关的网口按 ModbusTCP 协议发送数据到服务端(网关是客户端), 支持线圈(0x)和保持型寄存器(4x)。

tcp_mbswrite(string:服务端地址, int:服务端网络端口, int: 服务端站地址, int:目标区域, int:目标地址, string:数据缓存, int:数据偏移量, int:字节数量)

正常返回: int: 0=正常, <0=错误

异常返回: nil

通目标区域: 1=线圈(0x), 4=保持型寄存器(4x);

3.6 WebService 与 HTTP 相关

3.6.1 读一个文本文件到 LUA 变量

`fun_loadfile(string: 文件名称, int: 文件格式-0=text/1=xml/2=json)`

正常返回: (string: 文件字符串)

错误返回: nil

3.6.2 获取 XML 文件的单个节点的值

`fun_xmlgetvalue(string:xml 文件, string:节点名称)`

正常返回: (string: 节点数据; 设置返回 string: 新的 XML 文件)

错误返回: nil

3.6.3 设置 XML 文件的单个节点的值

`fun_xmlsetvalue(string:xml 文件, string:节点名称, string:节点数据)`

正常返回: (string: 新的 XML 文件)

错误返回: nil

3.6.4 HTTP 协议 POST 请求

`fun_postxml(string:URL ,string: POST 的数据头(headers), string: POST 的数据主体)`

正常返回: (string: 接收到的文件)

错误返回: nil

3.6.5 HTTP 协议 POST 请求 2

`fun_httppost(string:URL ,string: POST 的数据头(headers), string: POST 的数据主体, int:接收数据超时,单位毫秒)`

正常返回: (string: 接收到的文件主体, int: 接收到的文件长度)

错误返回: nil

3.6.6 HTTP 协议 GET 请求

`fun_httpget(string:URL ,int:接收数据超时,单位毫秒)`

正常返回: (string: 接收到的文件主体, int: 接收到的文件长度)

错误返回: nil

3.7 MQTT 与 JSON 文件相关

3.7.1 创建 JSON 文件

新建 json 文件，或者给 json 文件添加字段和数据

`fun_jsoncreate(string: json 文件, int: 字段的数据类型, string: 对象名称, string: 节点名称, string: 节点数据)`

正常返回: (string: 新的 json 文件)

错误返回: nil

json 文件为空: 新建 json 文件; 字段的数据类型: 1-bool, 2-number, 3-string, 4-object, 5-array

3.7.2 读取 JSON 文件

读取 json 文件指定字段的数据

`fun_jsondecode(string: json 文件, string: 节点名称)`

正常返回: (string: json 文件或者字段的数据)

错误返回: nil

多次调用, 可实现对层级多的文件解析

3.7.3 建立 MQTT 连接

MQTT 协议连接到 Broker 服务端(网关为 MQTT 客户端)

`fun_mqttconnect(string: Broker 地址, int: 网络端口, string: ClientID, int: ClearSession, string: 用户名, string: 密码)`

正常返回: int: 0 为正常连接, 其它值为异常代码

错误返回: nil

3.7.4 设置 MQTT 遗嘱

遗嘱设置须要在 `fun_mqttconnect` 之前使用

`fun_mqttsetwill(string: will topic, string: will payload, int: will Qos, int: will retain)`

正常返回: int: 0 为正常, 其它值为异常代码

错误返回: nil

3.7.5 发布 MQTT 消息

`fun_mqtttput(string: topic, int: 消息长度, string: payload, int: Qos,`

int:retain)

正常返回: (int: 0 为正常, 其它值为异常代码, int:消息编号)

错误返回: nil

3.7.6 订阅 MQTT 主题

fun_mqttsub(string: topic, int: Qos)

正常返回: (int: 0 为正常, 其它值为异常代码, int:编号)

错误返回: nil

3.7.7 查询 MQTT 消息

查询是否收到已订阅的主题消息

fun_mqttgetmsg(int: 查询时间限制)

正常返回: (string: topic, string: payload)

错误返回: nil

3.7.8 断开 MQTT 连接

fun_mqttdisconn()

正常返回: int: 0 为正常, 其它值为异常代码

错误返回: nil

3.8 FTP 文件上传与下载

3.8.1 FTP 文件上传(文件写入远端 FTP 服务器)

fun_ftpupload(string:FTP 路径(URL), string:用户名, string:密码, string:FTP 文件主体, int:通讯超时,单位毫秒)

正常返回: (int: 上传的文件长度)

错误返回: nil

3.8.2 FTP 文件下载(从远端 FTP 服务器复制文件到本地)

fun_ftpdown(string:FTP 路径(URL), string:用户名, string:密码, int:通讯超时,单位毫秒)

正常返回: (string: 接收到的文件主体, int: 接收到的文件长度)

错误返回: nil

3.9 数据库 SQL 上报与查询

3.9.1 无需返回数据的数据库 SQL 语句执行

包括但不限于 INSERT、UPDATE、DELETE 等

`fun_sqldb(int:数据库类型, string:服务器 IP 地址, int:服务器网络端口, string:用户名, string:密码, string:数据库名称, string:SQL 语句)`

正常返回: (int: 0 为成功执行 SQL 命令, 其它值为失败)

错误返回: nil

数据库类型: 0:MySQL, 1:SQLServer, 2:PostgreSQL

3.9.2 查询数据库中的数据

一般为 SELECT 语句。

`fun_sqlquery(int:数据库类型, string:服务器 IP 地址, int:服务器网络端口, string:用户名, string:密码, string:数据库名称, string:SQL 语句)`

正常返回: (int: 0 为成功执行 SQL 命令, 其它值为失败, string:查询到的数据, 多个数据时用逗号分割, 顺序与 SQL 语句的顺序一致)

错误返回: nil

数据库类型: 0:MySQL, 1:SQLServer, 2:PostgreSQL