IGT-SER 系列智能网关边缘计算

LUA 脚本用户手册

V1.76

埃和智能
AIHE TECH
武汉埃和智能科技有限公司
www.aihe-tech.com

前　言

本手册主要介绍了 IGT-SER 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

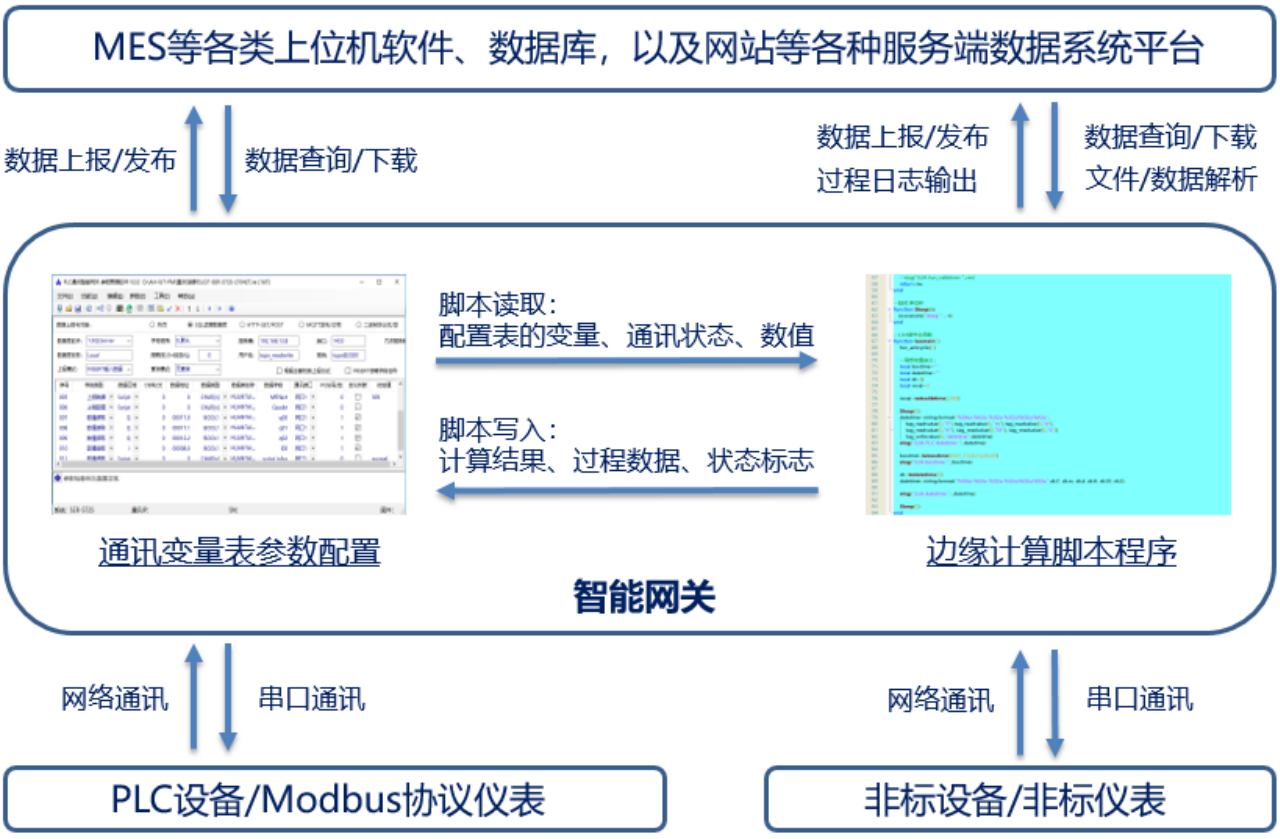
感谢您的反馈，让我们做得更好！

目 录

1.0 LUA 脚本简述

IGT 智能网关采用 Lua 语言作为脚本编程开发的语言，配合智能网关内已经封装好的功能函数，为智能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。

以下是脚本与智能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：

软件名称：智能网关边缘计算脚本编辑软件及范例

下载地址：http://www.aihe-tech.com/col.jsp?id=107

智能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

文件最大支持 32Kb(IGT-DSER 等高性能型的支持 64Kb),文件名中星号位置可自定义任意长度的字符，支持中文、英文和数字。

2.2 脚本程序下载到智能网关

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：[脚本]/[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智能网关后，需要重新启动智能网关模块才会执行，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 智能网并重启智能网关后，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，Ah_igt_pm.exe 参数软件的[脚本]/[脚本运行消息]就是一个 UDP 接收端。

3.0 功能函数说明

3.1 程序调试与控制函数

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

如果脚本程序执行完毕，或者因错误中止执行，由此可设置是否允许重新运行，或者设定重新运行的次数。

fun_ackcycle(int: 0 退出后不再执行，1 退出后一直继续循环执行，>1 执行的次数，次数到达后，重启智能网关之前将不再执行)

3.1.3 延时功能函数

延时，单位毫秒。

fun_sleep(int: 时间 *1ms)

3.1.4 查询智能网关与设备的通讯状态

查询智能网关与 PLC、仪表等所连接设备的通讯状态

fun_devstate(int: 可选参数；0～255 对应数据配置表的任务组参数，配置表没有没有这栏的对应 IP/站号；其它值获取所有设备的通讯状态。)

正常返回：int: 0～127 正常通讯,其它值表示通讯异常

错误返回：nil

3.1.5 智能网关系统重启

重启智能网关，不会蜂鸣报警

fun_restart()

3.1.6 查询智能网关与服务端的工作状态

查询智能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: 0～127 正常通讯,其它值表示状态异常

错误返回：nil

3.1.7 读取智能网关参数数据

读取通过参数配置软件写入到智能网关的网口与串口参数。

para_readvalue(string:参数名称，int:可选参数，0=读取设定值(默认)，1=读取当前值)

正常返回：string: 参数名称对应的参数值

错误返回：nil

参数名称可通过参数配置软件保存的配置文件内获取，如读取网口 2 的 IP 地址，参数名称为：net2.ipadd

3.1.8 设置智能网关参数数据

设置智能网关的网口与串口参数的当前值。

para_writevalue(string:参数名称，string:参数的当前值)

正常返回：int: >0 正常更新，=0 参数值与所设置的值相等，未更新

错误返回：nil

所写入的当前值不会掉电保持，掉电保持需要写入参数后用 para_savevalue 方法

3.1.9 将当前智能网关参数保存

将当前智能网关所有参数保存，并设置为掉电保持，重启后按该参数运行。

para_savevalue()

正常返回：int: >0 正常更新保存的参数计数

错误返回：nil

3.2 读写数据配置表的变量

3.2.1 变量当前值读取

读取智能网关数据配置表内变量的当前值。

tag_readvalue(int: 任务组(优先)或 IP/站号, string: 字段名称/数据地址/目标地址)

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

将数据写入到智能网关数据配置表内的变量。

tag_writevalue(int: 任务组(优先)或 IP/站号, string: 字段名称/数据地址/目标地址, 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)

字节数组方式读取智能网关数据配置表内变量的当前值。

tag_readchar(int: 任务组(优先)或 IP/站号, string: 字段名称/数据地址/目标地址)

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

错误返回：nil

3.2.4 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: 任务组(优先)或 IP/站号, 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；

3.2.5 其它 IGT 智能网关模块内的变量当前值读取

读取另外的 IGT 智能网关数据配置表内变量的当前值。

igt_tagread(string: IPV4 地址, int：数据的序号)

正常返回：string: 读取到的数据

错误返回：nil

3.2.6 读取智能网关的系统时间

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

3.2.7 读取智能网关的时间戳

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

3.2.8 设定时间方式校正智能网关时间

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

3.2.9 联网指定 NTP 服务器地址校正智能网关的时间

fun_netcalibtime(string: NTP 服务器 IP 地址或名称)

正常返回：(0:正常，1:错误)

错误返回：nil

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据

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 计算字符串的 SHA256 值

计算一个字符串的 SHA256 值，可设定输出字母的大小写。

fun_calcusha256 (int: 0 小写/1 大写, string: 参与计算的数据)

正常返回：string: sha256 计算结果

错误返回：nil

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

3.3.7 多字节转换到其它的数据类型

通过调整字节参数的次序可达到改变字节顺序的目的。

fun_byteconvert (int: 目标数据类型, byte[1], byte[2], byte[3], byte[4], …)

正常返回：string: 转换后的数据

错误返回：nil

目标数据类型 0=Int32, 1=UInt32, 2=Real, 3=LReal(需要输入 8 个字节才能转换)。

3.3.8 计算字符串的 MD5 值

计算一个字符串的 MD5 值，可设定输出的数据位数、输出数据字母的大小写。

fun_calcumd5 (int: 输出的数据位数(16 或 32), int:输出数据字母的大小写 (0 小写 1 大写), string: 原始字符串)

正常返回：string: MD5 字符串

错误返回：nil

3.3.9 创建一个 UUID 字符串

fun_genuuid ()

正常返回：string: UUID 字符串

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

通过智能网关串口与所连接的设备进行通讯，发送后接收数据，最长支持 1024 字节。

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

通过智能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，串口接收到的数据可截取指定长度返回，最长支持 1024 字节。

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 串口设备通讯 3

通过智能网关的串口进行通讯，发送或接收字符串数据,最长支持 1024 字符。

com_sendrecv3(int: 串口号(1/2), int:通讯时间限制(单位毫秒), string:发送数据,int:发送数据长度)

正常返回：(int: 串口号(1/2), int:接收字节数, 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

3.4.4 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

3.4.5 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

3.4.6 HJ212 协议中的 CRC 校验计算

fun_hjcrc16(string: 需要校验的数据缓存)

正常返回：(string: 4 位 16 进制 CRC 结果)

错误返回：nil

3.5 以太网 TCP/UDP 通讯

3.5.1 TCP 设备通讯

通过智能网关的网口进行通讯，TCP 发送后接收数据,最长支持 8192 字符。

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；发送数据长度设 0：只接收不发送

3.5.2 UDP 设备通讯

通过智能网关的网口进行通讯，UDP 发送后接收数据,最长支持 8192 字符。

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

通过智能网关的网口按 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)，一次最多 240 个字节。

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

目标区域：1=线圈(0x)，4=保持型寄存器(4x)；

3.5.4 网络检查命令 PING

通过智能网关 PING 一个指定的目标地址，1 秒后返回结果。

fun_ping(string:目标 IP 地址,int: PING 的次数)

正常返回：int: 0=失败，1=成功

异常返回：nil

3.5.5 开启一个 TCP 服务端

指定网络端口开启一个 TCP 服务端，并启动运行。每次接收到的数据可按行添加到指定

的文件，文件可存储到 TF 卡或者 U 盘。

tcp_serverrun(int:服务端网络端口, int: 每行数据接收的超时时间(单位毫秒), string:数据转存的文件路径(不需要写到文件可设置空),int:新行是否添加时间戳)

正常返回：int: 0=正常，-1:指定的网络端口不可用， -2:文件路径错误

异常返回：nil

3.5.6 查询 TCP 服务的状态和数据

在开启 TCP 服务后使用，用于查询 TCP 服务的运行状态和获取最新的数据文本。

tcp_getsertxt(int:服务端网络端口, int: 数据获取的超时时间(单位毫秒))

正常返回：int: 状态代码，string: 字符串文本

异常返回：nil

状态代码：>0:正常运行，=0:未运行，<0:错误

错误代码：-1：TCP 服务初始化错误，-2：TCP 服务开启错误，-3：客户端关闭了当前连接，-4：TCP 数据读取错误，-5：文件存储错误。

3.5.7 关闭一个 TCP 服务

tcp_cancelser(int:服务端网络端口)

正常返回：int: 0：正常执行关闭

异常返回：nil

3.5.8 通过 ModbusTCP 协议批量采集数据

智能网关通过 ModbusTCP 协议(客户端)批量采集数据，可以指定数据类型、字节顺序，每次最多可读取 4000 字节；服务端可以是 PLC 等远程设备，也可以是智能网关本地协议转换实现的服务端。

tcp_mbsread(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,int:数据类型,int:数据个数,int:字节顺序)

正常返回：(int:数据个数, string:接收的数据，逗号分隔)

异常返回：nil

目标区域：1=线圈(0x)，2=输入位(1x)，3=输入寄存器(3x)；4=保持寄存器(4x)；

数据类型： 1=int16, 2=uint16, 3=dint32, 4=udint32, 5=float

字节顺序：0=默认,1=交换字节,2=交换字,3=交换字节和字

3.6 WebService 与 HTTP 相关

3.6.1 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)

错误返回：nil

3.6.2 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

3.6.3 HTTP 协议 POST 请求

fun_postxml(string:URL ,string: POST 的格式参数(Headers，多个参数用换行符\n 分割), string: POST 的数据主体(通常为 JSON 或 XML 格式,最多 8192 字符))

正常返回：(string: 接收到的文件主体, int:接收到的文件长度(该值小于 0 时为错误代码))

错误返回：nil

3.6.4 HTTP 协议 POST 请求 2

fun_httppost(string:URL,string: POST 的格式参数(Headers,多个参数用换行符\n 分割),string:POST 的数据主体(通常为 JSON 或 XML 格式,最多 8192 字符),int:接收等待时间(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(该值小于 0 时为错误代码))

错误返回：nil

3.6.5 HTTP 协议 GET 请求

fun_getxml(string:URL,string: GET 的格式参数(Headers，多个参数用换行符\n 分割))

正常返回：(string: 接收到的文件主体(最多 8192 字符), int:接收到的文件长度(该值小于 0 时为错误代码))

错误返回：nil

3.6.6 HTTP 协议 GET 请求 2

fun_httpget(string:URL ,int:接收接收等待时间(单位毫秒))

正常返回：(string: 接收到的文件主体(最多 8192 字符)， int:接收到的文件长度(该值小于 0 时为错误代码))

错误返回：nil

3.6.7 HTTP 协议 PUT 请求

fun_httpput(string:URL ,string:PUT 的格式参数(Headers，多个参数用换行符\n 分割)， string:PUT 的数据主体(通常为 JSON 或 XML 格式,最多 8192 字符)， int:接收等待时间(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(该值小于 0 时为错误代码))

错误返回：nil

3.6.8 HTTP 协议 POST 提交文件或者数据

数据主题为 form-data 格式的数据或者文件，文件可以预先保存在 TF 卡或者 U 盘。

fun_httpdata(string:URL ,string: Headers 参数(多个参数用换行符\n 分割)，string:数据主体(正序排列的字符串，多个数据之间用‘&’隔开，见后面的举例)， int:数据主题的长度,int:接收等待时间(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(该值小于 0 时为错误代码))

错误返回：nil

数据主题格式举例：name1=data1&name2=data2&filedataname=’mnt/sd/image.png’

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(智能网关为 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, int:连接保持时间，单位秒)

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 pay , 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_mqttpub(string: topic, int:消息长度, string:消息主体(最长 8192 字符), 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，单位毫秒)

正常返回：(string: topic, string: 消息主体(最长 8192 字符))

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

3.7.9 开启 MQTT 的 SSL/TSL 功能，并设置证书文件名称

须要在 fun_mqttconnect 之前使用

fun_mqttsettls(string: ca file, string: cer file, string: key file)

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(最大支持 2M), 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(最大支持 2M), 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(最长 8192 字符))

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:Oracle

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(最长 8192 字符))

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, 3:Oracle

3.10 文件读写操作(U 盘/TF 卡)

相关功能函数中的文件名称参数需要包含完整的路径：

U 盘文件： /mnt/udisk/U 盘内的路径及文件名称

TF 卡文件：/mnt/sd/ TF 卡内的路径及文件名称

3.10.1 读一个文件到 LUA 变量

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

正常返回：(string: 文件缓存)

错误返回：nil

3.10.2 写一个文件到指定位置

fun_writefile(string: 文件名称, string: 文件缓存变量, int: 文件缓存长度)

正常返回：(int: >0 为正常操作的缓存长度，<=0 为错误代码)

错误返回：nil

3.10.3 在一个文件末尾追加一行字符串

file_addline(string: 文件名称, string: 文件缓存变量, int: 文件缓存长度)

正常返回：(int: =0 为正常操作，<0 为错误代码)

错误返回：nil

3.10.4 从一个文件中读出指定的一行

file_getline(string: 文件名称, int: 行号)

正常返回：(string:文件缓存，int: >0 为正常操作的缓存长度，<=0 为错误代码)

错误返回：nil

3.10.5 删除一个文件中指定的一行

file_delline(string: 文件名称, int: 行号)

正常返回：(int: int: >=0 为正常操作后的文件行数，<0 为错误代码)

错误返回：nil

3.10.6 获取一个文件的总的行数

file_linecount(string: 文件名称)

正常返回：(int: int: >=0 为正常操作后的文件行数，<0 为错误代码)

错误返回：nil