

# Aprus Lua-FanucCNC 配置说明

## 概述

本章主要为 Aprus Lua-fanuccnc 协议的相关配置说明，该协议主要针对支持 fanuccnc 协议的 CNC 设备。Aprus 适配器可通过 fanuccnc 协议与对接设备进行通信。而其中 Aprus 的 Lua 包含 apru.lua 和 config.lua 两个文件，客户只需配置 config.lua 就可以对支持 fanuccnc 协议的设备进行数据采集。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。

## 1 aprus.lua 配置说明

### demo 示例

```
package.cpath="/home/zdw/test/lib/x86/output/lua-cjson/lib/?so;./?.so"
package.path="/?.lua"
cjson = require "cjson"
config = require "config"
require "fanuccnc"

function act_control(m, json)

    for k,v in pairs(json) do
        if k ~= "Act" then
            fanuccnc.write(fanuccncobj, cjson.encode({"name" = k, ["val"] = v}))
        end
    end
end

function mqttdata_handle(m, topic, data)

    local json = cjson.decode(data)

    if json.Act == "Control" then
        act_control(m, json)
    end
end

function mqttsys_handle(m, code)

    if code == 0 then
        fanuccnc.stop(fanuccncobj)
    elseif code == 1 then
        fanuccnc.run(fanuccncobj)
    end
end
```

```
end
end

function fanuccnc_handle(obj, name, code, data)
    mqtt.publish(m1, name, "r", data)
end

function fanuccnc_load_collectnodes(obj, nodes)

    for k,v in pairs(nodes)
    do
        fanuccnc.addcnode(obj,cjson.encode(v))

    end
end

function fanuccnc_load_varnodes(obj, nodes)

    for k,v in pairs(nodes)
    do
        fanuccnc.addvnode(obj,cjson.encode(v))
    end
end

function init()
    m1 = mqtt.new()

    user.setluaver(config.Aprus.luaver)
    user.setdevinfo(config.Aprus.devinfo)
    user.ipconfig(config.Aprus.ipmode, config.Aprus.inet_addr, config.Aprus.netmask)
--    user.reset_eth0()--if you want to set eth0 use this interface
--    user.ipconfig_eth0("manual", "10.100.1.123", "255.255.255.0", "10.100.1.1", "114.114.114.114")
end

function start()

    init()
    fanuccncobj = fanuccnc.new("fanuccnc")
    fanuccnc.config(fanuccncobj, cjson.encode(config.fanuccnc.Device))
    fanuccnc_load_collectnodes(fanuccncobj,config.fanuccnc.Node)
    fanuccnc_load_varnodes(fanuccncobj,config.fanuccnc.Node)

    mqtt.run(m1)
    while true do
        local r = user.waitmsg()
```

```

if r.from == "mqtt-sys" then
    mqttsys_handle(r.session, r.code)
elseif r.from == "mqtt-msg" then
    mqttdata_handle(r.session, r.topic, r.payload)
elseif r.from == "fanuccnc" then
    fanuccnc_handle(r.obj, r.name, r.code, r.data)
end
end
end
start()

```

内容较多，不逐行描述，仅对关键部分(已标红)进行解释

## 1.1 导入 fanuccnc 协议支持库

语句	require "fanuccnc"
说明	导入 fanuccnc 协议支持库

## 1.2 加载 config.lua 配置文件

语句	config = require "config"
说明	导入 config.lua 中的配置信息(下一节介绍 config.lua)

## 1.3 创建 fanuccnc 对象

语句	fanuccncobj = fanuccnc.new("fanuccnc")
说明	创建 fanuccnc 对象，返回 fanuccncobj 供全局使用

## 1.4 配置 fanuccnc 对象接口参数

语句	fanuccnc.config(fanuccncobj, cJSON.encode(config.fanuccnc.Device))
说明	配置 fanuccnc 通信网卡参数

## 1.5 添加 fanuccnc 采集节点

语句	fanuccnc_load_collectnodes(fanuccncobj, config.fanuccnc.Node)
说明	添加 config.lua 中所配置的采集节点

## 1.6 添加 fanuccnc 上报变量节点

语句	<code>fanuccnc_load_varnodes(fanuccncobj,config.fanuccnc.Node)</code>
说明	添加 config.lua 中所配置的变量节点

## 1.7 运行 fanuccnc 实例

语句	<code>fanuccnc.run(fanuccncobj)</code>
说明	在 mqtt 建立连接后 运行 fanuccnc 实例 即开始采集+运算

## 1.8 暂停 fanuccnc 实例

语句	<code>fanuccnc.stop(fanuccncobj)</code>
说明	在 mqtt 连接断开时 暂停 fanuccnc 实例

## 1.9 等待 fanuccnc 事件

语句	<code>local r = user.waitmsg()</code>
说明	启动事件等待，该接口会返回全局各种事件，包括 fanuccnc 对象事件

## 1.10 fanuccnc 事件处理

语句	<code>elseif r.from == "fanuccnc" then fanuccnc_handle(r.obj, r.name, r.code, r.data)</code>
说明	当接收到 fanuccnc 对象事件时，调用处理函数 fanuccnc_handle r.obj 即 new 时所返回的对象 r.name 即 new 时所配置的对象名称 r.code 事件码 区分改变上报事件 周期上报事件等等 r.data 事件数据

## 1.11 mqtt 数据上报

语句	<code>function fanuccnc_handle(obj, name, code, data) mqtt.publish(m1, name, "r", data) end</code>
说明	若无需特殊处理，则直接将事件数据通过 mqtt 发送 如需要二次处理可将 data 展开做分析

## 2 config.lua 配置说明

### Demo 示例:

```
Aprus={
    ipmode="manual",                --auto/manual/none
    inet_addr="192.168.110.205",
    netmask="255.255.255.0",
    luaver="V00.R",
    devinfo="fanuccncDev",
},

fanuccnc={

    Device={

        ip = "192.168.110.201",

        port = "8193",

        type = "FANUC-SERIES 0I-MF PLUS"

    },

    Node={

        {vartype=1, cycle=3, name="L1"},      --加工数
        {vartype=2, cycle=3, name="L2"},      --运行状态      3: 运行中, 非 3: 待机
        {vartype=3, cycle=3, name="L3"},      --报警状态      0: 未发生报警, 非 0: 报警
        {vartype=4, cycle=3, name="L4"},      --报警数据
        {vartype=5, cycle=3, name="L5"},      --操作模式
        {vartype=6, cycle=3, name="L6"},      --主轴倍率
        {vartype=7, cycle=3, name="L7"},      --快速进给倍率
        {vartype=8, cycle=3, name="L8"},      --开机时间      单位: 秒
        {vartype=9, cycle=3, name="L9"},      --X 轴补偿数据
        {vartype=10, cycle=5, name="L10"},    --Y 轴补偿数据
        {vartype=11, cycle=5, name="L11"},    --Z 轴补偿数据
        {vartype=12, cycle=5, name="L12"},    --X 轴 DC 电压
        {vartype=13, cycle=3, name="L13"},    --Y 轴 DC 电压
        {vartype=14, cycle=3, name="L14"},    --Z 轴 DC 电压
        {vartype=15, cycle=3, name="L15"},    --当前语句号
        {vartype=16, cycle=4, name="L16"},    --刀具长度外形补偿 1
        {vartype=17, cycle=4, name="L17"},    --刀具长度外形补偿 2
        {vartype=18, cycle=4, name="L18"},    --刀具长度外形补偿 3
        {vartype=19, cycle=4, name="L19"},    --刀具长度外形补偿 4
    }
}
```

{vartype=20, cycle=4, name="L20"}, --刀具长度外形补偿 5

{vartype=21, cycle=4, name="L21"}, --刀具长度外形补偿 6

{vartype=22, cycle=4, name="L22"}, --刀具长度外形补偿 7

{vartype=23, cycle=4, name="L23"}, --刀具长度外形补偿 8

{vartype=24, cycle=4, name="L24"}, --刀具长度磨损补偿 1

{vartype=25, cycle=4, name="L25"}, --刀具长度磨损补偿 2

{vartype=26, cycle=4, name="L26"}, --刀具长度磨损补偿 3

{vartype=27, cycle=4, name="L27"}, --刀具长度磨损补偿 4

{vartype=28, cycle=4, name="L28"}, --刀具长度磨损补偿 5

{vartype=29, cycle=4, name="L29"}, --刀具长度磨损补偿 6

{vartype=30, cycle=4, name="L30"}, --刀具长度磨损补偿 7

{vartype=31, cycle=4, name="L31"}, --刀具长度磨损补偿 8

{vartype=32, cycle=4, name="L32"}, --刀具半径外形补偿 1

{vartype=33, cycle=4, name="L33"}, --刀具半径外形补偿 2

{vartype=34, cycle=4, name="L34"}, --刀具半径外形补偿 3

{vartype=35, cycle=4, name="L35"}, --刀具半径外形补偿 4

{vartype=36, cycle=4, name="L36"}, --刀具半径外形补偿 5

{vartype=37, cycle=4, name="L37"}, --刀具半径外形补偿 6

{vartype=38, cycle=4, name="L38"}, --刀具半径外形补偿 7

{vartype=39, cycle=4, name="L39"}, --刀具半径外形补偿 8

{vartype=40, cycle=4, name="L40"}, --刀具半径磨损补偿 1

{vartype=41, cycle=4, name="L41"}, --刀具半径磨损补偿 2

{vartype=42, cycle=4, name="L42"}, --刀具半径磨损补偿 3

{vartype=43, cycle=4, name="L43"}, --刀具半径磨损补偿 4

{vartype=44, cycle=4, name="L44"}, --刀具半径磨损补偿 5

{vartype=45, cycle=4, name="L45"}, --刀具半径磨损补偿 6

{vartype=46, cycle=4, name="L46"}, --刀具半径磨损补偿 7

{vartype=47, cycle=4, name="L47"}, --刀具半径磨损补偿 8

{vartype=48, cycle=4, name="L48"}, --运行时间 1 单位: 毫秒

{vartype=49, cycle=4, name="L49"}, --运行时间 2 单位: 分钟

{vartype=50, cycle=4, name="L50"}, --切削时间 1 单位: 毫秒

{vartype=51, cycle=4, name="L51"}, --切削时间 2 单位: 分钟

{vartype=52, cycle=4, name="L52"}, --主程序号

```

{vartype=53, cycle=4, name="L53"}, --子程序号

{vartype=54, cycle=4, name="L54"}, --进给轴转速

{vartype=55, cycle=4, name="L55"}, --主轴转速

{vartype=56, cycle=4, name="L56"}, --伺服 1 温度

{vartype=57, cycle=4, name="L57"}, --伺服 2 温度

{vartype=58, cycle=4, name="L58"}, --伺服 3 温度

{vartype=59, cycle=4, name="L59"}, --伺服 4 温度

{vartype=60, cycle=4, name="L60"}, --主伺服温度

{vartype=100, var1=48, var2=49, cycle=4, name="L100"}, --总运行时间 = (运行时间 1)/1000 +
(运行时间 2)*60    单位: 秒

{vartype=101, var1=50, var2=51, cycle=4, name="L101"}, --总切削时间 = (切削时间 1)/1000 +
(切削时间 2)*60    单位: 秒

},
},

```

## 2.1 Aprus: 接口属性

参数	值	说明
ipmode	"auto"/"manual"/"none"	ip 获取方式
inet_addr	"192.168.110.205"	Aprus 的 IP 地址
netmask	"255.255.255.0"	子网掩码
luaver	"MAX.LUA.V032700.R"	Aprus 的 Lua 版本信息, 根据实际脚本自行填写
devinfo	"fanuccncDev"	与 Aprus 对接设备, 可根据需求自行填写

## 2.2 fanuccnc-Device: 接口属性

参数	值	说明
ip	字符串	设备的 ip
port	字符串	设备的端口号
type	"FANUC-SERIES 0I-MF PLUS"	设备的型号

## 2.3 fanuccnc-Node: 节点属性

参数	值	说明
----	---	----

vartype	1	加工数
	2	运行状态 3: 运行中, 非 3: 待机
	3	报警状态 0: 未发生报警, 非 0: 报警
	4	报警数据
	5	操作模式
	6	主轴倍率
	7	快速进给倍率
	8	开机时间 (单位: 秒)
	9	X 轴补偿数据
	10	Y 轴补偿数据
	11	Z 轴补偿数据
	12	X 轴 DC 电压
	13	Y 轴 DC 电压
	14	Z 轴 DC 电压
	15	当前语句号
	16	刀具长度外形补偿 1
	17	刀具长度外形补偿 2
	18	刀具长度外形补偿 3
	19	刀具长度外形补偿 4
	20	刀具长度外形补偿 5
	21	刀具长度外形补偿 6
	22	刀具长度外形补偿 7
	23	刀具长度外形补偿 8
	24	刀具长度磨损补偿 1
	25	刀具长度磨损补偿 2
	26	刀具长度磨损补偿 3
	27	刀具长度磨损补偿 4

	28	刀具长度磨损补偿 5
	29	刀具长度磨损补偿 6
	30	刀具长度磨损补偿 7
	31	刀具长度磨损补偿 8
	32	刀具半径外形补偿 1
	33	刀具半径外形补偿 2
	34	刀具半径外形补偿 3
	35	刀具半径外形补偿 4
	36	刀具半径外形补偿 5
	37	刀具半径外形补偿 6
	38	刀具半径外形补偿 7
	39	刀具半径外形补偿 8
	40	刀具半径磨损补偿 1
	41	刀具半径磨损补偿 2
	42	刀具半径磨损补偿 3
	43	刀具半径磨损补偿 4
	44	刀具半径磨损补偿 5
	45	刀具半径磨损补偿 6
	46	刀具半径磨损补偿 7
	47	刀具半径磨损补偿 8
	48	运行时间 1(单位: 毫秒)
	49	运行时间 2(单位: 分钟)
	50	切削时间 1(单位: 毫秒)
	51	切削时间 2(单位: 分钟)
	52	主程序号
	53	子程序号
	54	进给轴转速

	55	主轴转速
	56	伺服 1 温度
	57	伺服 2 温度
	58	伺服 3 温度
	59	伺服 4 温度
	60	主伺服温度
	100	总运行时间 = (运行时间 1) / 1000 + (运行时间 2) * 60 单位：秒
	101	总切削时间 = (切削时间 1) / 1000 + (切削时间 2) * 60 单位：秒
var1	整型：1~60	当 vartype ≥ 100 时才能使用。 本节点值 = (var1 节点值 / 1000) + (var2 节点值 * 60)
var2	整型：1~60	当 vartype ≥ 100 时才能使用。 本节点值 = (var1 节点值 / 1000) + (var2 节点值 * 60)
cycle	整型：0~n	单位：秒 精度：0.1 等于 0：即指定上报类型为改变上报 大于 0：即指定上报周期
name	字符串	上报名称
offset	+ - * / ~ 数字	例如：offset = "*10; +5.123; ~1" 表示 乘以 10 再加 5.123 结果保留 1 位小数