# Trw说明文档

## 概述

trw 是 trout read and write register 的缩写,用来读写 trout 寄存器

## Trw组成

* trw 驱动组件:

路径:kernel/drivers/net/wireless/trout/mac/src/test/trw

* trw 接口库: libtroutrw.

路径:/device/sprd/partner/trout/wlan/trout\_rw

* trw 应用程序.

路径:/device/sprd/partner/trout/wlan/trout\_rw

## 编译选项

trw 驱动组件默认是编译的，编译开关

EXTRA\_CFLAGS += -DTROUT\_DEBUG\_RW\_PROC

路径:kernel/drivers/net/wireless/trout/ mac/src/test/trw /config.mk.

trw 应用程序需要在android 的 userdebug ,eng 模式下才编译，,需要修改的话，请修改: /device/sprd/partner/trout/wlan/trout\_rw/android.mk

## 使用介绍

trw 满足不同层次的用户需要，提供了3种模式：

1. 初级用户：命令行模式。
2. 中级用户：交互模式。
3. 高级用户：脚本模式.
4. 更高级的用户，可以使用libtroutrw 接口库进行二次开发.

## 命令模式

### 格式:

trw {action} [type] [param]

action :动作:包括set ,get .

type :操作对象类型:包括mode,power,reg,phy,comm,sys,mac,rf,legacy,wifiram,btram (目前wifiram,btram 还没有实现),如果不指定type,默认是reg.

reg 是通用寄存器设置类型，要求输入的地址是绝对地址，需要人为加上BASE地址.

Phy,comm,sys,mac,rf, 要求输入的地址是偏移地址,不需要加上BASE地址,程序会自动加上.

Legacy 是遗留功能保留接口，对应以前的iwconfig flag\_debug 功能，比如：

Trw get legacy 2

Trw get legacy 8 (打印所有寄存器的值到文件里面，和以前的接口一样.

param 不同的action和type组合，有不同的参数.

### 例子:

设置驱动配置模式:目前只支持bin模式(asic还未实现).

trw set mode bin

给trout 上电:

trw set power on

给trout 关电:

trw set power off

读取trout硬件ID

trw get reg 0x8038

下面这条命令和它等价:

trw get 0x8038 (如果没有设置type,默认为reg)

设置STA地址:

trw set reg 0x8018 0x12345678

等价命令:

trw set 0x8038 0x12345678 (如果没有设置type,默认为reg)

设置phy 寄存器:

trw set phy 0x8c 0x01

读phy寄存器:

trw get phy 0x8c

查看版本:

trw -v

查看帮助:

trw -h

## 交互模式

交互模式,就是用户输入一条lua脚本语句,trw立即解释执行 trw是一个完全的lua解释器.

如何进入交互模式:

运行trw ,不带任何参数,就进入了交互模式.

下面是使用例子，文档后面有一个张表格，专门用来说明提供的lua函数.

设置bin 模式:

> trw.set\_mode(1)

上电:

> trw.set\_power(1,1)

关电:

>trw.set\_power(0,1)

读ID:

>print( trw.get\_reg(0x8038) )

设置STA reg:

>trw.set\_reg(0x12345678,0x8018)

设置phy:

>trw.set\_phy(0x8c,0x01)

读phy:

>print( trw.get\_phy(0x8c) )

## 脚本模式

trw可以直接解释lua脚本文件：

./trw filename.lua

你可以参考： /device/sprd/partner/trout/wlan/trout\_rw/script/test.lua文件

## 高级技巧

trout有很多寄存器,你可以定义符号，代替地址.

你只需要编写一个define.lua文件，, 把它放入到trw 所在目录及可以了,

define.lua是一个完全的lua脚本文件，你可以在里面定义地址，函数等.

例子:

PA\_BASE = 0x8000

HW\_ID = PA\_BASE + 0x38

HW\_STA =PA\_BASE + 0x18

这些，你就可以直接使用HW\_ID ,比如 reg:

trw get HW\_ID

trw set HW\_STA 0x12345678

## 函数参考

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 函数原型 | 参数 | 返回值 | 功能说明 | 例子 |
| trw.set\_reg(val,add) | Val:设置的值。  addr:寄存器的地址 | 成功返回的值>0,失败返回的值<0 | 设置寄存器的值 | trw.set\_reg(0x8018,0x12345678) |
| trw.get\_reg(add) | addr:寄存器的地址 | 成功返回寄存器的值,失败返回的值<0 | 读取寄存器的值 | trw.get\_reg(0x8038) |
| trw.set\_mode(mode) | 1：设置驱动配置模式为bin模式  2:设置驱动配置模式为asic模式 | 成功返回的值>0,失败返回的值<0 | 设置驱动的配置模式. | trw.set\_mode(1) |
| trw.get\_mode() | 无 | 返回当前驱动的配置模式 | 读取当前驱动的配置模式 | trw.get\_mode() |
| trw.set\_power(val,addr) | Val=1上电。  val=0，关电  addr=1,给wifi上电或者关电.  Addr=2,给fm上电或者关电.  Addr=4,给bt上电或者关电 | 成功返回的值>0,失败返回的值<0 | 给trout芯片上电或者是关电 | trw.set\_power(1,1) |
| trw.get\_power(addr) | addr=1,获取wifi电源状态  Addr=2,fm电源状态  Addr=4:bt电源状态 | Addr=4:bt电源状态 | 成功返回的值为当前的电源状态,失败返回的值<0 获取电源状态,这个函数目前还没有实现，总是返回 0 | trw.get\_power(1) |
| trw.set\_phy(val,addr) | Val:设置的值。  addr:寄存器的地址 | 成功返回的值>0,失败返回的值<0 | 设置phy寄存器的值 | trw.set\_phy(0x8c,0x01) |
| trw.get\_phy(addr) | addr:寄存器的地址 | 成功返回寄存器的值,失败返回的值<0 | 读取phy寄存器的值 | trw.get\_phy(0x8c) |
| Trw.set\_comm(val,addr) | Val:设置的值。  addr:寄存器的偏移地址 | 成功返回的值>0,失败返回的值<0 | 设置comm寄存器的值 | trw.set\_comm(0x8c,0x01) |
| trw.get\_comm(addr) | addr:寄存器的偏移地址 | 成功返回的值>0,失败返回的值<0 | 读取comm寄存器的值 | trw.get\_comm(0x8c) |
| Trw.set\_sys(val,addr) | Val:设置的值。  addr:寄存器的偏移地址 | 成功返回的值>0,失败返回的值<0 | 设置sys寄存器的值 | trw.set\_sys(0x8c,0x01) |
| trw.get\_sys(addr) | addr:寄存器的偏移地址 | 成功返回的值>0,失败返回的值<0 | 读取sys寄存器的值 | trw.get\_sys(0x8c) |
| Trw.set\_mac(val,addr) | Val:设置的值。  addr:寄存器的偏移地址 | 成功返回的值>0,失败返回的值<0 | 设置mac寄存器的值 | trw.set\_mac(0x8c,0x01) |
| trw.get\_mac(addr) | addr:寄存器的偏移地址 | 成功返回的值>0,失败返回的值<0 | 读取mac寄存器的值 | trw.get\_mac(0x8c) |
| Trw.set\_rf(val,addr) | Val:设置的值。  addr:寄存器的偏移地址 | 成功返回的值>0,失败返回的值<0 | 设置rf寄存器的值 | trw.set\_rf(0x8c,0x01) |
| trw.get\_rf(addr) | addr:寄存器的偏移地址 | 成功返回的值>0,失败返回的值<0 | 读取rf寄存器的值 | trw.get\_rf(0x8c) |
|  |  |  |  |  |
|  |  |  |  |  |