HTOOL-SA8 通信协议

本文中单片机由 client 指代，上位机由 server 指代

Status : IDLE:0X00 0X01 BUSY:0X00 0X02

Head : 0X55

Version : 0x00

Cmd :

server:

START: 0X64 frequency:0x01 generator:0x02 amplitude: 0x03

device:0x04 marker: 0x05 heart:0x63

1.返回连接前单片机的状态以及数据值给上位机（初始化）

55 00 00 42 03 00 01 data1 data2 data3 data4 data5 xx xx xx xx

data1：0B AE 00 00 00 14 00 00 0B C2 00 00 0B B8 00 00
（起始频率2990.00MHz 跨度频率20.00MHz 截止频率3010.00MHz 中心频率3000.00MHz）

data2：01 00 0B B8 00 00 0F
（使能信号源 失能追踪模式 发送3000.00MHz -15dBm的正弦信号）

data3：0A 32 03 01 4B
（Grid设置为10dB Zin设置为50Ohm Scale设置为dBm(3) reference_amplitude设置-75dBm）

Data4：00 01 07 00 00 00 00
（Idle_shutdown为never模式，Fre.shift为0.00MHz Amp.shift设置为-7dB）

Data5：10 08 11 0B B2 00 00 02 58 00 0B B6 00 00 02 BC 00 0B BA 00 00 02 BC 00 0B BE 00 00 03 20
（模式为avgtrace 平均值为8 该模式失能 Peak1使能,ADC值为600 Maker2失能，标记值为2998.00MHz,ADC值为700 Maker3失能，标记值为3002.00MHz,ADC值为700 Maker4失能，标记值为3006.00MHz,ADC值为800 ）

例子：
55 00 00 41 03 00 01 0B AE 00 00 00 14 00 00 0B C2 00 00 0B B8 00 00 01 00 0B B8 00 00 0F 0A 32 03 01 4B 00 01 07 00 00 00 00 10 08 11 0B B2 00 00 02 58 00 0B B6 00 00 02 BC 00 0B BA 00 00 02 BC 00 0B BE 00 00 03 20 18 A2 55 DB

2.开始打开串口进行连接时

Server:　访问 client 状态

head	version number	length（cmd~crc）	cmd	OK_state		crc（head~OK_state）			
0x55	0x00	0x07	0x64	0x4F	0x4B	0x??	0x??	0x??	0x??
Byte 1	Byte 2	Byte 3	Byte4	Byte 5	Byte 6	Byte 7	Byte 8	Byte 9	Byte 10
fixed value	fixed value	fixed value	fixed value	H8	L8	Crc32			

Client:返回当前状态 BUSY/IDLE

head	version	length		cmd	status	
0x55	0x00	0x??	0x??	0x??	0x??	0x??
Byte 1Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
fixed value	fixed value	fixed value	fixed value	fixed value	fixed value	fixed value

ok		crc32			
0x4f	0x4b	0x??	0x??	0x??	0x??
Byte 8	Byte 9	Byte 10	Byte 11	Byte 12	Byte 13
H8	L8	crc32			

eg:

server　->　55 00 07 64 4F 4B A9 67 A8 70

client　->　OK 指令

55 00 00 02 64 00 02 4F 4B 6A 3F 7F AD

心跳指令

55 00 00 02 63 00 01 4F 4B DA BB 1D

返回全部配置数据

3.收波设置

Server：发送频率界面数据

head	version number	Length（cmd~crc）	cmd	start_freq		start_dotfreq		span_freq		span_dotfreq	
0x55	0x00	0x15	0x01	0x??	0x??	0x??	0x??	0x??	0x??	0x??	0x??
Byte 1	Byte 2	Byte 3	Byte4	Byte 5	Byte 6	Byte 7	Byte 8	Byte 9	Byte 10	Byte 11	Byte 12
fixed value	fixed value	fixed value	fixed value	H8	L8	H8	L8	H8	L8	H8	L8

stop_freq		stop_dotfreq		center_freq		center_dotfreq		crc（head~center_dotfreq）			
0x??	0x??	0x??	0x??	0x??	0x??	0x??	0x??	0x??	0x??	0x??	0x??
Byte 13	Byte 14	Byte 15	Byte 16	Byte 17	Byte 18	Byte 19	Byte 20	Byte 21	Byte 22	Byte 23	Byte 24
H8	L8	H8	L8	H8	L8	H8	L8	Crc32			

Client：返回当前数据供 server 进行校准

head	version number	Length（cmd~crc）		cmd	status		start_freq		start_dotfreq		span_freq	
0x55	0x00	OX??	0x??	0x11	0x00	0x01	0x??	0x??	0x??	0x??	0x??	0x??
Byte 1	Byte 2	Byte 3	Byte 4	Byte5	Byte 6	Byte 7	Byte 8	Byte 9	Byte 10	Byte 11	Byte 12	Byte 13
fixed value	fixed value	fixed value	fixed value	fixed value	H8	L8	H8	L8	H8	L8	H8	L8

span_dotfreq		stop_freq		stop_dotfreq		center_freq		center_dotfreq		crc（head~center_dotfreq）			
0x??	0x??	0x??	0x??	0x??	0x??	0x??	0x??	0x??	0x??	0x??	0x??	0x??	0x??
Byte 14	Byte 15	Byte 16	Byte 17	Byte 18	Byte 19	Byte 20	Byte 21	Byte 22	Byte 23	Byte 24	Byte 25	Byte 26	Byte 27
H8	L8	H8	L8	H8	L8	H8	L8	H8	L8	Crc32			

eg:

起始频率 2990.00MHz 截止频率 3010.00MHz 跨度频率 20.00MHz 中心频率 3000.00MHz

Server -> 55 00 15 01 0B AE 00 00 00 14 00 00 0B C2 00 00 0B B8 00 00 C9 FF 46 E4

Client -> 55 00 00 10 11 00 01 0B AE 00 00 00 14 00 00 0B C2 00 00 0B B8 00 00 3B 96 DB B2

4.发波设置

Sever：发送当前gennerator界面的设置

head	version number	Length（cmd~crc）	cmd	set_generator_onoff	set_track_onoff	generator_freq		generator_dotfreq	
0x55	0x00	0x0C	0x02	0x??	0x00	0x??	0x??	0x??	0x??
Byte 1	Byte 2	Byte 3	Byte4	Byte5	Byte6	Byte 7	Byte 8	Byte 9	Byte 10
fixed value	fixed value	fixed value	fixed value	0/1	0	H8	L8	H8	L8

generator_amp（Amp>0）	crc（head~generator_amp）			
0x??	0x??	0x??	0x??	0x??
Byte 11	Byte 12	Byte 13	Byte 14	Byte 15
15~25	Crc32			

Client:返回数据给 server 进行校准

head	version number	Length（cmd~crc）		cmd	Status		set_generator_onoff
0x55	0x00	0x??	0x??	0x02	0x00	0x01	0x??
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
fixed value	fixed value	fixed value	fixed value	fixed value	H8	L8	0/1

set_track_onoff	generator_freq		generator_dotfreq		generator_amp（Amp>0）
0x00	0x??	0x??	0x??	0x??	
Byte 9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14
0	H8	L8	H8	L8	15~25

crc（head~generator_amp）

0x??	0x??	0x??	0x??
Byte 15	Byte 16	Byte 17	Byte 18
Crc32			

eg:

使能信号源 失能追踪模式 发送 3000.00MHz -15dBm 的正弦信号

Server -> 55 00 0C 02 01 00 0B B8 00 00 0F 5F A1 75 A2

Client -> 55 00 00 07 12 00 01 01 00 0B B8 00 00 0F 2D 07 D9 BE

5.设置参考幅度以及相关刻度参数

Server：发送当前幅度界面参数配置

head	version number	Length（cmd~crc）	cmd	set_grid	set_zin	set_scale	refamp_negative
0x55	0x00	0x0A	0x03	0x??	0x??	0x??	0x??
Byte 1	Byte 2	Byte 3	Byte4	Byte 5	Byte 6	Byte 7	Byte 8
fixed value	fixed value	fixed value	fixed value	5	50	3	0/1

refamp	crc（head~refamp）			
0x??	0x??	0x??	0x??	0x??
Byte 9	Byte 10	Byte 11	Byte 12	Byte 13
-75~10dBm	Crc32			

注解：当幅度调节界面刻度为 dBm（即 set_scale=3）,refamp_negative 为参考幅度的正负号（refamp_negative 等于 0 时为正号，等于 1 时为负号），refamp 为参考幅度的绝对值，此时参考幅度可输出范围为（-75~10dBm）；当其刻度为 dBuV 时（即 set_scale=4），参考幅度可输出范围为（32~117dBuV）

Client：返回当前界面的参数

head	version number	Length（cmd~crc）		cmd	Status		set_grid
0x55	0x00	0x??	0x??	0x13	0x00	0x01	0x??

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
fixed value	fixed value	H8	L8	fixed value	H8	L8	fixed value

set_zin	set_scale	refamp_negative	refamp	crc（head~refamp）			
0x??	0x??	0x??	0x??	0x??	0x??	0x??	0x??
Byte 9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15	Byte 16
fixed value	fixed value	fixed value	fixed value	Crc32			

eg:

Grid 设置为 10dB　Zin 设置为 50Ohm　Scale 设置为 dBm(3) reference_amplitude 设置为 -75dBm

Server　->　55 00 0A 03 0A 32 03 01 4B 16 E9 02 FC

Client　->　55 00 00 05 13 00 01 0A 32 03 01 4B 46 33 74 DA

6.设置 peak,mark,maxtrace,mintrace,avagtrace 以及瀑布图

Server:发送当前参考标记界面参数配置

head	version number	Length（cmd~crc）	cmd	waterfall_mode		avgtrace_step_value	mark_peak1		mark1_fre(int)	
0x55	0x00	0x1B	0x05	0x??		0x??	0x??		0x??	0x??
Byte 1	Byte 2	Byte 3	Byte4	Byte 5		Byte 6	Byte 7		Byte 8	Byte 9
fixed value	fixed value	fixed value	fixed value	H4	L4	avgtrace	H4	L4	H8	L8

mark1_fre(fra)		mark_peak2		mark2_fre(int)		mark2_fre(fra)		mark_peak3		mark3_fre(int)	
0x??	0x??	0x??		0x??	0x??	0x??	0x??	0x??		0x??	0x??
Byte 10	Byte 11	Byte 12		Byte 13	Byte 14	Byte 15	Byte 16	Byte 17		Byte 18	Byte 19
H8	L8	H4	L4	H8	L8	H8	L8	H4	L4	H8	L8

mark3_fre(fra)		mark_peak4		mark4_fre(int)		mark4_fre(fra)		crc（head~mark4_fre(fra)）			
0x??	0x??	0x??		0x??	0x??	0x??	0x??	0x??	0x??	0x??	0x??
Byte 20	Byte 21	Byte 22		Byte 23	Byte 24	Byte 25	Byte 26	Byte 27	Byte 28	Byte 29	Byte 30
H8	L8	H4	L4	H8	L8	H8	L8	Crc32			

注解：waterfall_mode，mark_peak1，mark_peak2，mark_peak3，mark_peak4的L4为使能位，L4为0时失能，为1时使能；waterfall_mode的H4为0时是瀑布图模式，为1时平均值模式，为2时是最大值模式，为3时是最小值模式；avgtrace_step_value为平均值模式下的平均系数，可选（4，8，16）（以下协议均是如此）；而其他参数的H4为0时指标记模式（mark），为1时指峰值模式（peak）

Client:返回采集的marker peak数据

head	version number	length		cmd	status	mark_peak1		
0x??	0x??	0x??	0x??	0x??	0x??	0x??		
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7		
fixed value	fixed value	H8	L8	fixed value	fixed value	H2	h2	L4

mark1_fre(int)		mark1_fre(fra)		mark1_adc		mark_peak2		
0x??	0x??	0x??	0x??	0x??	0x??	0x??		
Byte 8	Byte 9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14		
H8	L8	H8	L8	H8	L8	H2	h2	L4

mark2_fre(int)		mark2_fre(fra)		mark2_adc		mark_peak3		
0x??	0x??	0x??	0x??	0x??	0x??	0x??		
Byte 15	Byte 16	Byte 17	Byte 18	Byte 19	Byte 20	Byte 21		
H8	L8	H8	L8	H8	L8	H2	h2	L4

mark3_fre(int)		mark3_fre(fra)		mark3_adc		mark_peak4		
0x??	0x??	0x??	0x??	0x??	0x??	0x??		
Byte 22	Byte 23	Byte 24	Byte 25	Byte 26	Byte 27	Byte 28		
H8	L8	H8	L8	H8	L8	H2	h2	L4

mark4_fre(int)		mark4_fre(fra)		mark4_adc	
0x??	0x??	0x??	0x??	0x??	0x??
Byte 29	Byte 30	Byte 31	Byte 32	Byte 33	Byte 34
H8	L8	H8	L8	H8	L8

crc32			
0x??	0x??	0x??	0x??
Byte 35	Byte 36	Byte 37	Byte 38
crc32			

注解：该条协议与上述协议不同之处：mark_peak1，mark_peak2，mark_peak3，mark_peak4的H4被拆分为H2和h2，H2为0时指标记模式（mark），为1时指峰值模式（peak）；h2则是为0时指没有峰值（peak）数据，为1时指存在峰值（peak）数据

Client：返回数据给server进行校准

head	version	length		cmd	waterfall_mode		avgtrace_step_value
0x55	0x00	0x??	0x??	0x15	0x??		0X??
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6		Byte 6
fixed value	fixed value	H8	L8	fixed value	H4	L4	avgtrace

mark_peak1		mark_peak2		mark_peak3		mark_peak4		Crc32			
0X??		0X??		0X??		0X??		0X??	0X??	0X??	0X??
Byte 7		Byte 8		Byte 9		Byte 10		Byte 11	Byte 12	Byte 13	Byte 14
H4	L4	H4	L4	H4	L4	H4	L4	CRC32			

eg:

模式为 avgtrace 平均值为 8 该模式失能 Peak1 使能 Maker2 失能，标记值为 2998.00MHz Maker3 失能，标记值为 3002.00MHz Maker4 失能，标记值为 3006.00MHz

Server -> 55 00 1B 05 10 08 11 0B B2 00 00 00 0B B6 00 00 00 0B BA 00 00 00 0B BE 00 00 8C DF 1E 3F

//返回采集的marker peak数据

Client ->55 00 00 1C 02 00 01 51 0B B8 00 02 E4 09 00 3D 70 00 40 00 00 00 7E 20 00 01 00 00 00 7C 63 00 2D 00 00 07 2B 72 6F

//校准

Client ->55 00 00 06 15 00 01 10 08 11 00 00 00 8D EC C7 8D

7.设置设备的相关参数（频率偏移和幅度偏移）

Server:发送当前device界面数据

head	version number	Length（cmd~crc）	cmd	Idle_shutdown	ampshift_ negative	ampshift	freqshift_freq	
0x55	0x00	0x0C	0x04	0x??	0x??	0x??	0x??	0x??
Byte 1	Byte 2	Byte 3	Byte4	Byte 5	Byte 6	Byte 7	Byte 8	Byte 9
fixed value	fixed value	fixed value	fixed value	3,5,7,10,15,never	0/1	-30~30dB	H8	L8

freqshift_dotfreq		crc（head~freqshift_dotfreq）			
0x??	0x??	0x??	0x??	0x??	0x??
Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15
H8	L8	Crc32			

Client：返回 device 界面数据供 server 进行校准

head	version number	Length（cmd~crc）		cmd	status		Idle_shut down	ampshift_ negative	ampshift
0x55	0x00	0x??	0x??	0x14	0x00	0x01	0x??	0x??	0x??
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8	Byte 9	Byte 10
fixed value	fixed value	H8	L8	fixed value	H8	L8	fixed value	fixed value	fixed value

freqshift_freq		freqshift_dotfreq		Crc32			
0x??	0x??	0x??	0x??	0x??	0x??	0x??	0x??
Byte 11	Byte 12	Byte 13	Byte 14	Byte 15	Byte 16	Byte 17	Byte 18
H8	L8	H8	L8	CRC32			

eg:

Idle_shutdown 为 never 模式，Fre.shift 为 0.00MHz Amp.shift 设置为-7dB

Server -> 55 00 0C 04 00 01 07 00 00 00 00 79 6C 32 D1

Client ->55 00 00 07 14 00 01 00 01 07 00 00 00 00 C0 BB E7 0D

8.S21 部分协议

直接发送计算后的数据
SERVER 请求获取 SWR 的数据

SMODE:

	S21	S11	SWR
SERVER	2	1	3
CLIENT	2	1	3

ACAPMODE:

	LOAD	SHORT	OPEN	NORMALIZE	ALL
SERVER	1	2	3	4	5
CLIENT	0x31	0x32	0x33	0x34	0x35

MODESTATUS:

	CLEAR(STOP+CLEAR)	START	STOP
SERVER	1	2	3
CLIENT	0x31	0x32	0x33

server->client 发送指令

head	version number	Length（cmd~crc）	cmd	Tracking	Input atten	power
0x55	0x00	0x??	0x21	0x??(0/1)	0x??(0-30)	0x??(15-25)
Byte 0	Byte 1	Byte 2	Byte 4	Byte 5	Byte 6	Byte 7
fixed value	fixed value	fixed value	fixed value	fixed value	fixed value	fixed value

SMODE	ACAPMODE	MODESTATUS	CRC（head~ACAPMODE）			
0X??(1-3)	0X??(1-4)	0X??(1-3)	0x??	0x??	0x??	0x??
Byte 8	Byte 9	Byte 10	Byte 10	Byte 11	Byte 12	Byte 13
fixed value	fixed value	fixed value	CRC32			

Cleint->server 校准完成标志

HEAD	VERSION NUMBER	Length		CMD	STATUS		SMODE
0x55	0x00	0x00	0x04	0x21	0x??	0x05	0X??(1-3)
Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 6	Byte 7	Byte 8
fixed value	fixed value	H8	L8	fixed value	H8	L8	fixed value

ACAPMODE	MODEstatus		CRC			
0x??	0x??	0x??	0x??	0x??	0x??	0x??
Byte 9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15
fixed value			CRC32			

Client->server(由于太占用内存，接收到就会立即进行发送)

head	version number	Length （cmd~crc）		cmd	STATUS	
0x55	0x00	0x??	0x??	0x21	0x??	0x01
Byte 0	Byte 1	Byte 2	Byte3	Byte 4	Byte 6	Byte 7
fixed value	fixed value	H8	L8	fixed value	H8	L8

VALUE(short)		IDX（uint16_t）		.....		.....	
0x??	0x??	0x??	0x??	0x??	0x??	0x??	0x??
Byte 9	Byte 10	Byte 13	Byte 14	Byte ??	Byte ??	Byte ??	Byte ??
H8	L8	H8	L8	H8	L8	H8	L8

CRC			
0x??	0x??	0x??	0x??
??	0x??	0x??	0x??
CRC32			

9.上位机控制单片机串口断开

Server:　提供 client 断开状态

head	version number	length（cmd~crc）	cmd	Heart_state				
0x55	0x00	0x0A	0x65	0x48	0x45	0x41	0x52	0x54
Byte 1	Byte 2	Byte 3	Byte4	Byte 5	Byte 6	Byte 7	Byte 8	Byte 9
fixed value	fixed value	fixed value	fixed value	fixed value	fixed value	fixed value	fixed value	fixed value

crc（head~Heart_state）			
0x??	0x??	0x??	0x??
Byte 10	Byte 11	Byte 12	Byte 13
Crc32			

eg:

server　-> 55 00 0A 65 48 45 41 52 54 73 1F 27 7B