威视锐 V3 Technology,.Ltd. INNOVATION FOR RESEARCH

# YunSDR Y3x0

## 时戳同步开发指南

**Rev. 2.2**

威视锐旗下品牌：

Sora Station　V3SDR　RED CYCLONE　ZING Zynq SoC Expert

## 修订记录

| 版本 | 修订日期 | 修订内容 |
|---|---|---|
| 1.0 | 2015 年 9 月 1 日 | 初始版本 |
| 2.0 | 2015 年 9 月 20 日 | 时戳系统及帧格式系统升级 |
| 2.1 | 2015 年 10 月 9 日 | 增加循环发送模式 |
| 2.2 | 2015 年 11 月 1 日 | 增加多天线测试细节 |

## 关于威视锐科技

北京威视锐科技有限公司是专注于软件定义无线通信（SDR）系统仿真、验证和测试平台的研发与生产，同时也提供通用高性能信号处理板卡，应用于机器视觉、生命科学和高能物理等科学计算领域。

威视锐与微软研究院联合开发的 Sora 软件无线电平台已经成为世界上知名大学和科研结构开展无线通信研究的首选平台，也是学术研究领域全球唯一的 100%基于 x86 的宽带实时软件无线电平台，目前已经有超过 20 多个国家的 300 多个科研用户。作为全球最大的可编程器件（FPGA）公司 XILINX 的全球合作伙伴，威视锐科技提供基于 XILINX FPGA/SoC 的全方位解决方案。特别是 ZYNQ 7000 系列 SoC 产品，威视锐携手 XILINX 发布了全球第一款基于 ZYNQ SoC 的低成本开源 SoC 模块 SNOWLeo，性价比远高于国外同类产品，大大降低了 SoC 系统的开发门槛。此外，威视锐旗下的红色飓风开发板自从 2004 年发售以来已成为国内销售时间最长、知名度最高以及用户量最多的开发板产品。

多年以来，威视锐科技坚持“Innovation for Research”的发展理念，与国内众多知名高校建立合作关系，帮助专家、学者和研发工程师将创新的理念变成现实。对于产业界客户，威视锐提供严格验证的核心模块、智能便携的测量仪器以及定制化的设计服务来加快产品研发周期。

# 目 录

# 1 产品概述

北京威视锐科技 YunSDR Y3x0软件无线电开发套件，主要面对无线通信、频谱分析和信号处理领域应用，可作为无线通信项目开发验证、通信原理教学、无线通信学习、嵌入式开发的理想平台。套件采用 Xilinx ZYNQ 和 ADI AD9361集成化射频方案。本文档主要介绍应用 YunSDR Y3x0平台通过时戳同步的参考设计测试与开发应用指南，多个模块可以组成分布式系统或者 massive mimo 应用。如果用户需要了解 Y3x0的硬件特性请参考《YunSDR FPGA 开发指南》。

**时戳同步主要特点:**

- 方式一：选购 Y3x0S 系列产品，内置 GPS 时钟模块
- 方式二：选购 Y3x0系列产品，非内置 GPS 时钟模块，选配1to10时钟分配模块
- 时戳控制指令打包在 IQ 数据帧中，通过1000M 网络传输，与数据同步
- 提供 FPGA、ARM、上位机全部源代码，支持用户二次开发
- 可配合用户定制适合特殊应用场景的多天线或分布式时戳同步方案

**YunSDR Y3x0S 内置 GPS 特性:**

- 1pps 秒脉冲输出
- 10MHz 系统时钟，与1pps 同步
- 32颗卫星同步
- 秒脉冲10ms 脉宽
- GPS 模块输入端 LNA 增益范围20~30dB

**YunSDR Y3x0配合时钟分配板特性:**

- 内置与 Y3x0S 相同指标的 GPS 模块
- 分配时钟源：
  - GPS 10MHz 时钟分配
  - 内部26MHz 参考时钟分配
  - 外部 SMA 接口时钟输入（最高200MHz 时钟频率）
- 1PPS 输出
- 当使用 GPS 时钟时 GPS 提供
- 其他情况内部通过计数器计数获得
- 1PPS 和参考时钟输出均为 LVTTL3.3V 电平
- 宽电源供电范围：5-12V 电源供电

**时戳同步机制特点**

- AD9361参考时钟由本地 TCXO 与外部或 GPS 同步时钟鉴相获得
- 时戳控制数据长度与 IQ 数据打包
- 时戳计数位宽64位
- 时戳计数时钟为 ad9361采样时钟
- 提供分布式 MIMO4x4时戳同步测试例程

**Y3x0S 前面板与参考选择：**

http://www.v3t.com.cn

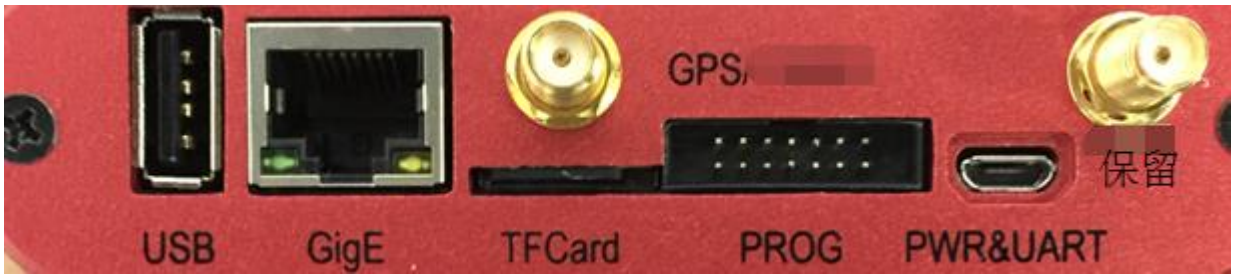


图 1 Y310S 前面板

选择内部参考时钟（REF_SELET=0），使用 ADF4001鉴相器校准（VCO_CAL=0）

**Y310配时钟分配板前面板与参考选择：**


图 2 Y310 前面板

选择外部参考时钟（REF_SELET=1），使用 ADF4001鉴相器校准（VCO_CAL=0）

# 2 时戳同步系统结构

## 2.1 数据帧格式

收发的 IQ 数据插入帧头和控制信息后打包成数据帧，数据帧的位宽定位为 32 位。发送端 PL 程序通过解析帧头和控制信息确定发送的通道、开始时刻、时戳有效、发送数据采样点数等信息。接收端 PL 程序通过寄存器的读写得到接收通道、开始时刻、时戳有效、接收数据采样点数等信息，接收的帧头和控制信息在 PL 端插入。用户需注意 Y3x0 采样 AD9361 一共两个通道，可以选择使用其中一个通道或者两个通道同时工作，单通道工作时一个样值是 32bit；双通道工作时一个样值是 64bit。

表格 1 帧格式定义

| 序号 | 内容 | 位宽 | 字段定义 |
|---|---|---|---|
| 1 | 帧头 | [31:0] | 0xFF000080 用于识别帧起始 |
| 2 | 通道选择 | [1:0] | 发送/接收通道 0x1 ch1；0x2 ch2；0x3 ch1&2 |
| | 时戳有效 | [4] | 发时/接收戳使能，0x1 PL 达到时戳开始发送/接收 |

| | | | |
|---|---|---|---|
| | | | 0x0 PL 不等待时戳直接开始发送/接收 |
| 3 | 样值长度 | [31:0] | 发送/接收数据采样点数<br>单通道，2 to 2^32-2，必须为偶数。每个采样点 32bit<br>双通道，1 to 2^32-1，正整数。每个采样点 64bit |
| 4 | 保留位 | [31:0] | 保留位填写 0x55aaaa55 |
| 5 | 时戳 低 32bit | [31:0] | 发送/接收的开始时戳 |
| 6 | 时戳 高 32bit | [31:0] | 发送/接收的开始时戳 |
| 7 | 数据位 | 31:0 | 第一个点的 IQ 样值或第一通道第一个点的 IQ 样值 |
| 8 | 数据位 | 31:0 | 第二个点的 IQ 样值或第二通道第一个点的 IQ 样值 |
| 9 | 数据位 | 31:0 | 第三个点的 IQ 样值或第一通道第二个点的 IQ 样值 |
| 10 | 数据位 | 31:0 | 第四个点的 IQ 样值或第二通道第二个点的 IQ 样值 |
| 。。。IQ 数据 | | | |

表格 2 时戳相关寄存器(PS/PL 交互)

| 地址 | 寄存器名称 | 位宽 | 方向 | 定义 |
|---|---|---|---|---|
| 0x10 | mtime_start | [0] | PS->PL | 时戳计数开始，PL 等待 1pps 开始计数 |
| 0x14 | rx_chan | [1:0] | PS->PL | 接收通道 0x1 ch1；0x2 ch2；0x3 ch1&2 |
| | rxtime_enable | [4] | PS->PL | 接收时戳使能，0x1 PL 达到时戳开始接收<br>0x0 直接接收不等待时戳 |
| | rx_enable | [8] | PS->PL | 0x1 rx 接收使能；0x0 rx 接收结束 |
| 0x18 | rx_length | [31:0] | PS->PL | 接收数据采样点数<br>单通道，2 to 2^32-2，必须为偶数。每个采样点 32bit<br>双通道，1 to 2^32-1，正整数。每个采样点 64bit |
| 0x1c | rxtime_start[31:0] | 32bit | PS->PL | 接收开始时戳 低 32 位 |
| 0x20 | rxtime_start[63:32] | 32bit | PS->PL | 接收开始时戳 高 32 位 |
| 0x24 | mtime[31:0] | 32bit | PL->PS | PL 时戳计数值 低 32 位 |
| 0x28 | mtime[63:32] | 32bit | PL->PS | PL 时戳计数值 高 32 位 |

## 2.2 Y3x0S 外部参考校准原理

Y3x0S 增加了外部参考校准的功能，可以通过板载 GPS 模块、外部参考时钟对板卡上的 VC-TCXO(温补压控振荡器)进行校准，使板载时钟的基准频率与参考时钟一致。原理框图如下：

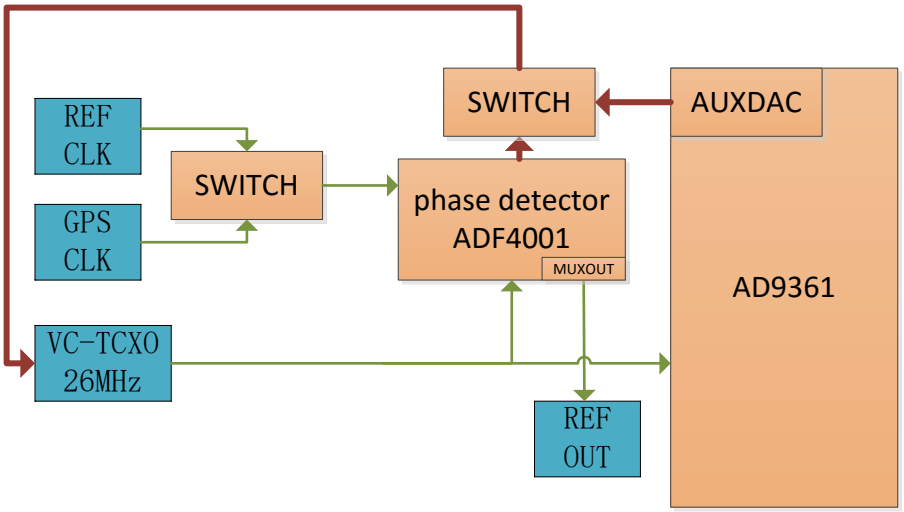


图 3 外参考同步原理

外部参考时钟 refrerence 和 vctcxo 时钟 vc-clock 分别进行分频，分频后鉴相，将 CP 电压输出给 VC-TCXO 的压控端，当 ADF4001 锁定后 VC-TCXO 与外部参考的基准时钟严格同频。假设外部参考 10MHz，本地晶振 26MHz，rcount 表示参考频频系数，ncount 表示本地时钟分频系数，另 rcount=10，ncount=26 可以完成 ADF4001 的鉴相功能。

## 2.3 发送模式工作流程

YunSDR 通过千兆网口与 PC 机通信,软件基于 linux 平台采用套接字接口与 PC 软件交互，接收端接口为 TCP 协议，端口为 5005。

### 2.3.1 软件流程

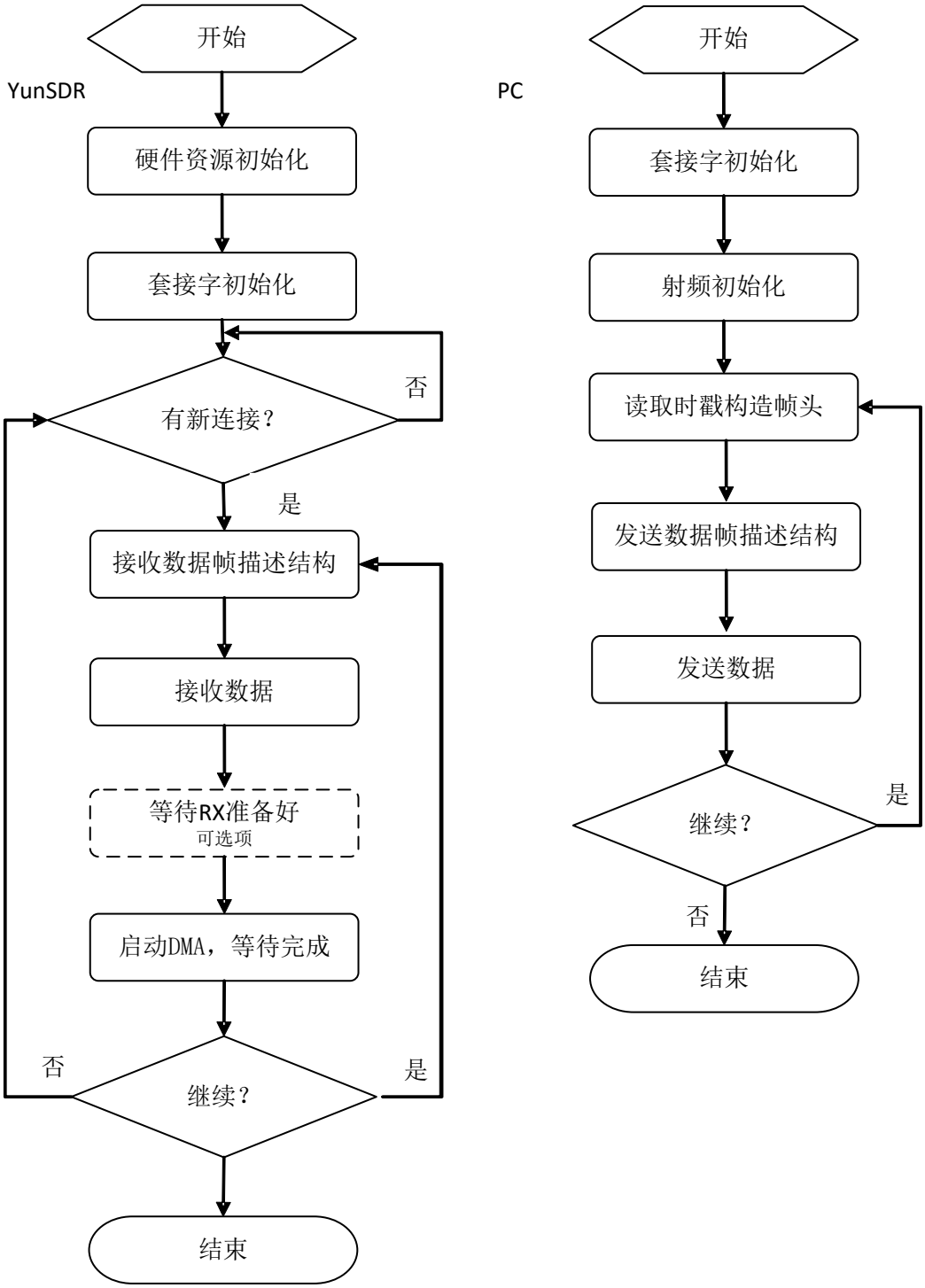


图 4 发送模式软件工作流程

### 2.3.2 数据帧描述结构

数据描述结构是板卡 PS 软件部分和 PC 软件部分套接字之间的交互数据，用以描述控制字、收发交互的参数等，属于可自定义部分。数据帧头+数据是实际的要通过 DMA 传送至 PL 部分，数据帧头由 FPGA 程序解析，而数据部分是最终要 transmit 的数据。

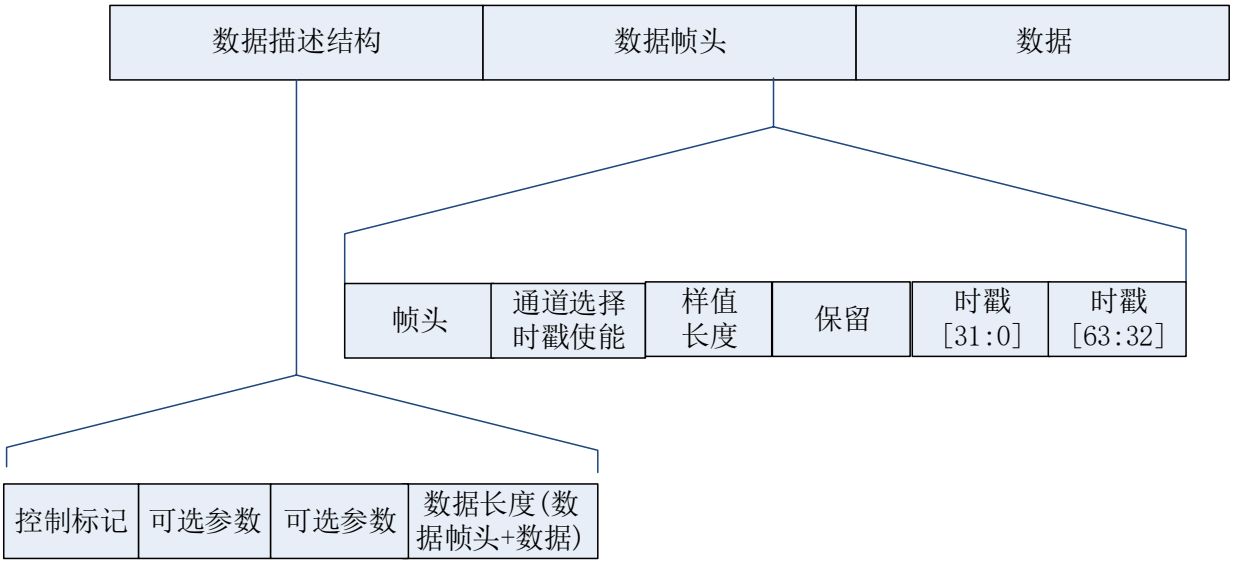


图 5 发送模式网络数据包格式

软件定义：

```
struct msg_head {
    uint32_t control;
    uint32_t param1;
    uint32_t param2;
    uint32_t payload;
};
```

说明：

表格 3 发送网络数据包定义

| 名称 | 位宽 | 含义 |
|---|---|---|
| control | [31:0] | 控制标记<br>*START_TX_NORMAL* = 01<br>*START_TX_LOOP* = 02 |
| Param1 | [31:0] | 可选参数 |
| Param2 | [31:0] | 可选参数 |
| Payload | [31:0] | 数据大小(含数据帧头) |

### 2.3.3 数据帧描述结构填充举例

```
msg.control = START_TX_NORMAL;
msg.param1 = 1;
msg.payload = nbyte+FRAME_HEAD_IN_BYTE;
```

nbyte 为将要发送的数据帧大小(以 Byte 为单位)，FRAME_HEAD_IN_BYTE 为数据帧头的大小 24Byte，数据帧描述结构为小端序，网络传输顺序为 control-->param1-->param2-->payload

### 2.3.4 数据帧头填充举例

表格 4 发送数据帧头填充

| 序号 | 内容 | 位宽 | 字段定义 |
|---|---|---|---|
| 1 | 帧头 | [31:0] | 0xFF000080 用于识别帧起始 |
| 2 | 通道选择 | [1:0] | 发送通道 0x1 ch1；0x2 ch2；0x3 ch1&2 |
| | 时戳有效 | [4] | 发送给戳使能，0x1 PL 达到时戳开始发送<br>0x0 PL 不等待时戳直接开始发送 |
| 3 | 样值长度 | [31:0] | 发送数据采样点数<br>单通道，2 to 2^32-2，必须为偶数。每个采样点 32bit<br>双通道，1 to 2^32-1，正整数。每个采样点 64bit |
| 4 | 保留位 | [31:0] | 保留位填写 0x55aaaa55 |
| 5 | 时戳 低 32bit | [31:0] | 发送的开始时戳 |
| 6 | 时戳 高 32bit | [31:0] | 发送的开始时戳 |
| 7 | 数据位 | 31:0 | 第一个点的 IQ 样值或第一通道第一个点的 IQ 样值 |
| 8 | 数据位 | 31:0 | 第二个点的 IQ 样值或第二通道第一个点的 IQ 样值 |
| 9 | 数据位 | 31:0 | 第三个点的 IQ 样值或第一通道第二个点的 IQ 样值 |
| 10 | 数据位 | 31:0 | 第四个点的 IQ 样值或第二通道第二个点的 IQ 样值 |
| 。。。IQ 数据 | | | |

```
    *((uint32_t *)tx_buffer) = 0xFF000080;
#ifdef DUAL_CH
    *((uint32_t *)tx_buffer+1) = 0x3 | 0x1<<4;
    *((uint32_t *)tx_buffer+2) = nbyte/8;
```

```
#else
    *((uint32_t *)tx_buffer+1) = 0x1 | 0x1<<4;
    *((uint32_t *)tx_buffer+2) = nbyte/4;
#endif
    *((uint32_t *)tx_buffer+3) = 0x55AAAA55;
    *((uint32_t *)tx_buffer+4) = (uint32_t)timestamp;
    *((uint32_t *)tx_buffer+5) = (uint32_t)(timestamp>>32);
```

nbyte为将要发送的数据帧大小(以Byte为单位)

#### 2.3.5 数据发送程序举例

```
    ret = send(sockfd, (void *)&msg, sizeof(struct msg_head), 0);
if(ret < 0)
{
    printf("send error:%s\n", strerror(errno));
}
ret = send(sockfd, (void *)tx_buffer, nbyte+FRAME_HEAD_IN_BYTE, 0);
if(ret < 0)
{
    printf("send error:%s\n", strerror(errno));
}
```

## 2.4 接收模式工作流程

YunSDR 通过千兆网口与 PC 机通信,软件基于 linux 平台采用套接字接口与 PC 软件交互，接收端接口为 TCP 协议，端口为 5004。

### 2.4.1 软件流程

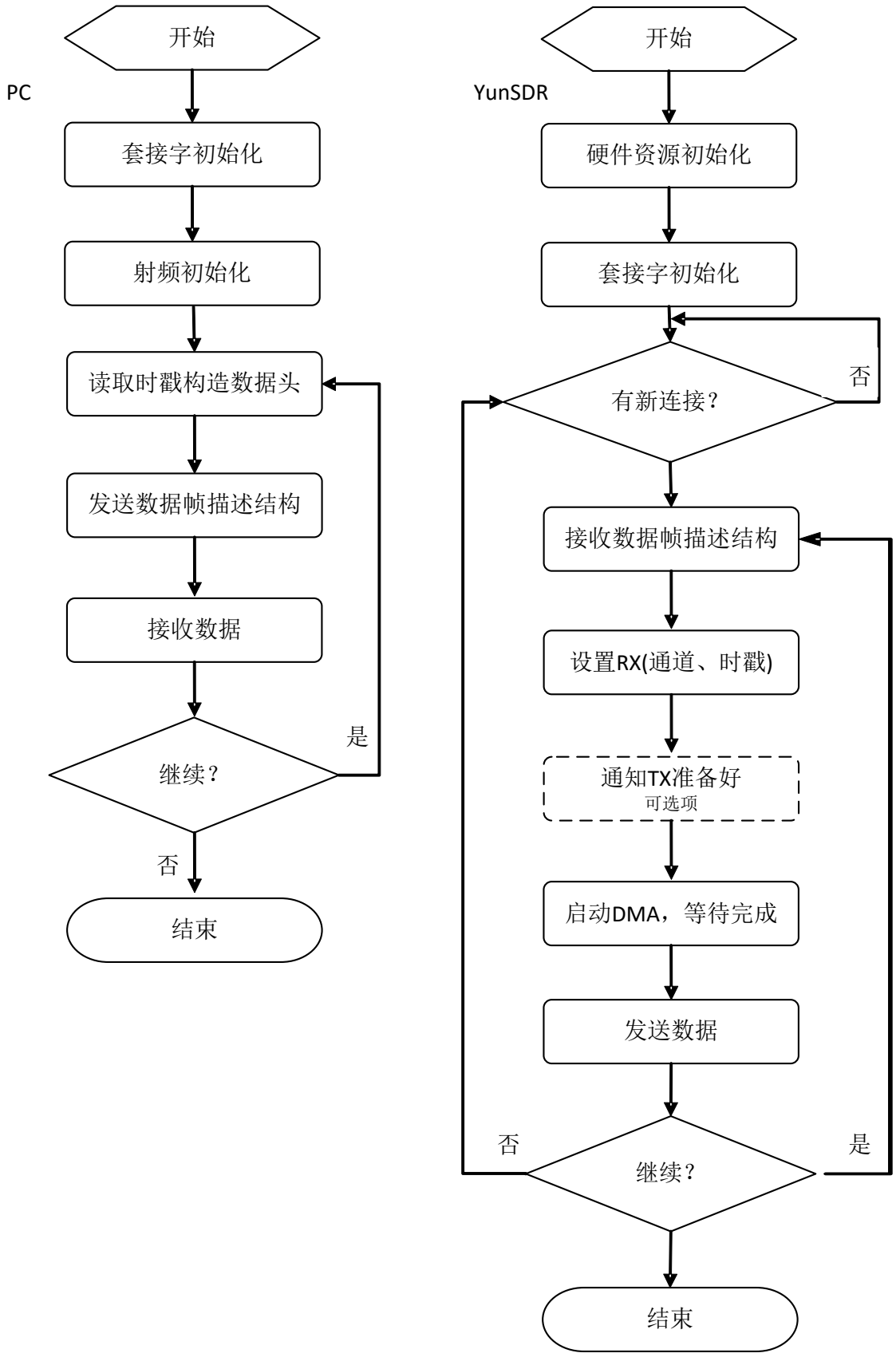
PC
开始
套接字初始化
射频初始化
读取时戳构造数据头
发送数据帧描述结构
接收数据
继续？
是
否
结束
YunSDR
开始
硬件资源初始化
套接字初始化
有新连接？
否
接收数据帧描述结构
设置RX(通道、时戳)
通知TX准备好
可选项
启动DMA，等待完成
发送数据
否
继续？
是
结束

图 6 接收模式软件工作流程

### 2.4.2 数据帧描述结构

数据描述结构是板卡 PS 软件部分和 PC 软件部分套接字之间的交互数据，用以描述控制字、收发交互的参数等，属于可自定义部分。

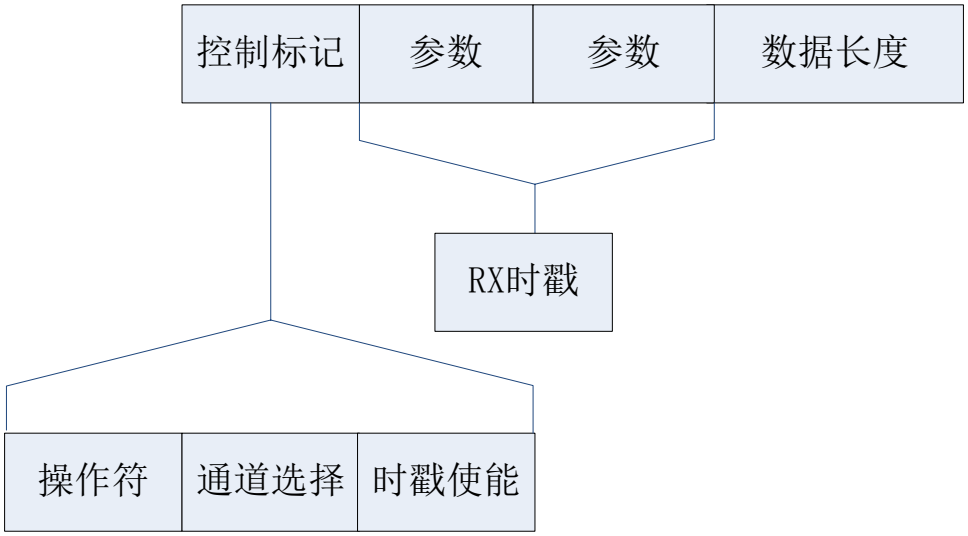


图 7 接收模式帧结构

软件定义：

```
struct msg_head {
    uint32_t control;
    uint32_t param1;
    uint32_t param2;
    uint32_t payload;
};
```

说明：

表格 5 接收模式帧结构定义

| 名称 | 位宽 | 含义 |
|---|---|---|
| control | [31:0] | 控制标记 |
| | [0:8] | *START_RX_NORMAL* = 03 |
| | [8:16] | 接收戳使能，0x1 PL 达到时戳开始接收<br>0x0 PL 不等待时戳直接开始接收 |
| | [31:16] | 接收通道 0x1 ch1；0x2 ch2； |

| | | |
|---|---|---|
| | | 0x3 ch1&2 |
| Param1 | [31:0] | 接收时戳[31:0] |
| Param2 | [31:0] | 接收时戳[63:32] |
| Payload | [31:0] | 接收数据采样点数<br>单通道，2 to 2^32-2，必须为偶数。每个采样点 32bit<br>双通道，1 to 2^32-1，正整数。每个采样点 64bit |

### 2.4.3 数据帧描述填充举例

```
#ifdef DUAL_CH
   msg.control = START_RX_NORMAL | 0x3<<16 | (0x1<<8);
   msg.payload = nbyte/8;
#else
   msg.control = START_NOW | 0x1<<16 | (0x1<<8);
   msg.payload = nbyte/4;
#endif
   msg.param1 = (uint32_t)timestamp;
   msg.param2 = (uint32_t)(timestamp>>32);
```

据帧描述结构为小端序，网络传输顺序为 control-->param1-->param2-->payload

### 2.4.4 接收数据程序举例

```
send(sockfd, (void *)&msg, sizeof(struct msg_head), 0);

do {
   ret = recv(sockfd, (void *)rx_buffer+count,
nbyte+FRAME_HEAD_IN_BYTE-count, MSG_WAITALL);
   if(ret < 0) {
      printf("recv error:%s\n", strerror(errno));
      break;
   } else {
      count += ret;
} while(count != (nbyte+FRAME_HEAD_IN_BYTE));
```

先发送数据描述结构，然后接收 nbyte+24Bytes 大小的数据，24Bytes 是接收回来的数据帧头

表格 6 接收到的数据格式

| 序号 | 内容 | 位宽 | 字段定义 |
|---|---|---|---|
| 1 | 帧头 | [31:0] | 0xFF000080 用于识别帧起始<br>0xFF0000AB 时戳超时 |
| 2 | 通道选择 | [1:0] | 接收通道 0x1 ch1；0x2 ch2；0x3 ch1&2 |
| | 时戳有效 | [4] | 接收戳使能，0x1 PL 达到时戳开始接收<br>0x0 PL 不等待时戳直接开始接收 |
| 3 | 样值长度 | [31:0] | 接收数据采样点数<br>单通道，2 to 2^32-2，必须为偶数。每个采样点 32bit<br>双通道，1 to 2^32-1，正整数。每个采样点 64bit |
| 4 | 保留位 | [31:0] | 保留位填写 0x55aaaa55 |
| 5 | 时戳 低 32bit | [31:0] | 接收的开始时戳 |
| 6 | 时戳 高 32bit | [31:0] | 接收的开始时戳 |
| 7 | 数据位 | 31:0 | 第一个点的 IQ 样值或第一通道第一个点的 IQ 样值 |
| 8 | 数据位 | 31:0 | 第二个点的 IQ 样值或第二通道第一个点的 IQ 样值 |
| 9 | 数据位 | 31:0 | 第三个点的 IQ 样值或第一通道第二个点的 IQ 样值 |
| 10 | 数据位 | 31:0 | 第四个点的 IQ 样值或第二通道第二个点的 IQ 样值 |
| 。。。IQ 数据 | | | |

## 2.5 控制通道工作流程

YunSDR 通过千兆网口与 PC 机通信，软件基于 linux 平台采用套接字接口与 PC 软件交互，配置接口为 UDP 协议，端口为 5006

### 2.5.1 命令格式

命令长度为 8 bytes，格式如下

MSB LSB

| 包头 | 控制字 | 数据 | 数据 | 数据 | 数据 | 数据 | 数据 |
|---|---|---|---|---|---|---|---|

### 2.5.2 命令详解

➢ 时戳操作

MSB LSB

| 0xF0 | 0x23 | 0 | 0 | Param |
|---|---|---|---|---|

命令方向 PC→device

Param: 1 ---使能时戳计数； 2 ---复位时戳计数； 0 ---读取当前时戳计数值

MSB LSB

| 时戳[31:0] | 时戳[63:32] |
|---|---|

命令方向 device → PC

当前时戳计数

➢ 射频配置

MSB LSB

| 0xF0 | 0x22 | ID | 0 | Param |
|---|---|---|---|---|

注：

ID:控制命令ID，见表1

Param[31:0]:命令参数

**表1**

| ID | 命令描述 |
|---|---|
| 3 | "Sets the TX LO frequency [Hz]." |
| 5 | "Sets the TX sampling frequency [Hz]." |
| 7 | "Sets the TX RF bandwidth [Hz]." |
| 9 | "Sets the TX1 attenuation [mdB]." |
| 11 | "Sets the TX2 attenuation [mdB]." |
| 13 | "Sets the TX FIR state." |
| 15 | "Sets the RX LO frequency [Hz]." |

| | |
|---|---|
| 17 | "Sets the RX sampling frequency [Hz]." |
| 19 | "Sets the RX RF bandwidth [Hz]." |
| 21 | "Sets the RX1 GC mode.[0:mgc; 1:fast agc, 2: slow agc]" |
| 23 | "Sets the RX2 GC mode. [0:mgc; 1:fast agc, 2: slow agc]" |
| 25 | "Sets the RX1 RF gain." |
| 27 | "Sets the RX2 RF gain." |
| 29 | "Sets the RX FIR state." |

注： ID为DEC十进制格式

当调用ID为3和15的命令时，参数格式为：

MSB LSB

| | | | |
|---|---|---|---|
| 0xF0 | 0x22 | ID | CMD PARAM |

CMD PARAM占用40bit,因为射频频点在70MHz~6GHz之间，部分频点大于32bit的存储空间，所以频点参数的低32bit有CMD1存储，剩余高位由CMD0的[7:0]位存储

| ID | 命令描述 |
|---|---|
| 40 | "Reference clock select." |
| 41 | "vco calibration select." |
| 42 | "fdd or tdd select." |
| 43 | "tx & rx switch." |
| 44 | " auxdac1 set. " |
| 45 | " adf4001 set. " |

注：ID 为 DEC 十进制格式

实例：

REF_SELEC =1 external reference;=0 internal reference

MSB LSB

| | | | | |
|---|---|---|---|---|
| 0xF0 | 0x22 | 40 | 0 | 0/1 |

VCO_CAL_SELECT =1 AUXDAC1;=0 ADF4001

MSB LSB

| | | | | |
|---|---|---|---|---|
| 0xF0 | 0x22 | 41 | 0 | 0/1 |

当选择为 ADF4001 时需要配置 ad9361_adf4001_set 命令,参数为 rcount 和 ncount

MSB LSB

| | | | | | |
|---|---|---|---|---|---|
| 0xF0 | 0x22 | 44 | 0 | Rcount | Ncount |

Rconut[31:16]

Nconut[15:0]

当选择为 AUXDAC1 时需要配置 ad9361_auxdac1_set 命令，参数为 0~2000mv

MSB LSB

| | | | | |
|---|---|---|---|---|
| 0xF0 | 0x22 | 45 | 0 | mV |

mV[31:0]

FDD_TDD_SEL =1 FDD;=0 TDD

MSB LSB

| | | | | |
|---|---|---|---|---|
| 0xF0 | 0x22 | 42 | 0 | 0/1 |

TRX_SW =1 TX;=0 RX

MSB LSB

| | | | | |
|---|---|---|---|---|
| 0xF0 | 0x22 | 43 | 0 | 0/1 |

### 2.5.3 配置举例

```
//send rx gain set cmd
cmd[0] = 0xf0220000|(25<<8);
cmd[1] = 5;
ret = sendto(cmd_sock, (void *)cmd, sizeof(int)*2, 0, (struct
```

```
sockaddr *)&cmd_addr, sizeof(cmd_addr));
if(ret < 0)
   printf("sendto error:%s\n", strerror(errno));
cmd[0] = 0xf0220000|(27<<8);
cmd[1] = 5;
ret = sendto(cmd_sock, (void *)cmd, sizeof(int)*2, 0, (struct
sockaddr *)&cmd_addr, sizeof(cmd_addr));
if(ret < 0)
   printf("sendto error:%s\n", strerror(errno));
```

## 3 分布式同步工作流程

当多个 YunSDR 同时工作时，采用相同的时戳计数基准，操作步骤如下：

1. PC 端程序采用多线程方式(线程之间运行采用同步方式)，每个线程对应一台 YunSDR，各个线程同时向每块板卡发送时戳启动命令，每台 YunSDR 在接收到命令后，等待 1PPS 脉冲的到来，由于在此系统下各个 YunSDR 采用相同的 PPS 源，所以所有 YunSDR 将会在同一时刻开始进行时戳计数。

2. 发送数据流程，PC 程序读取其中一台 YunSDR 的当前时戳值，然后 PC 程序以当前时戳为基准，构造每台 YunSDR 将要发送数据的帧头，构造完成后下发数据，YunSDR 到达时间点将自动发送数据。

3. 接收数据流程，PC 程序读取其中一台 YunSDR 的当前时戳值，然后 PC 程序以当前时戳为基准，构造每台 YunSDR 将要发送数据帧描述结构，构造完成后下发数据帧描述结构，YunSDR 到达时间点将自动接收回数据并发送至 PC。

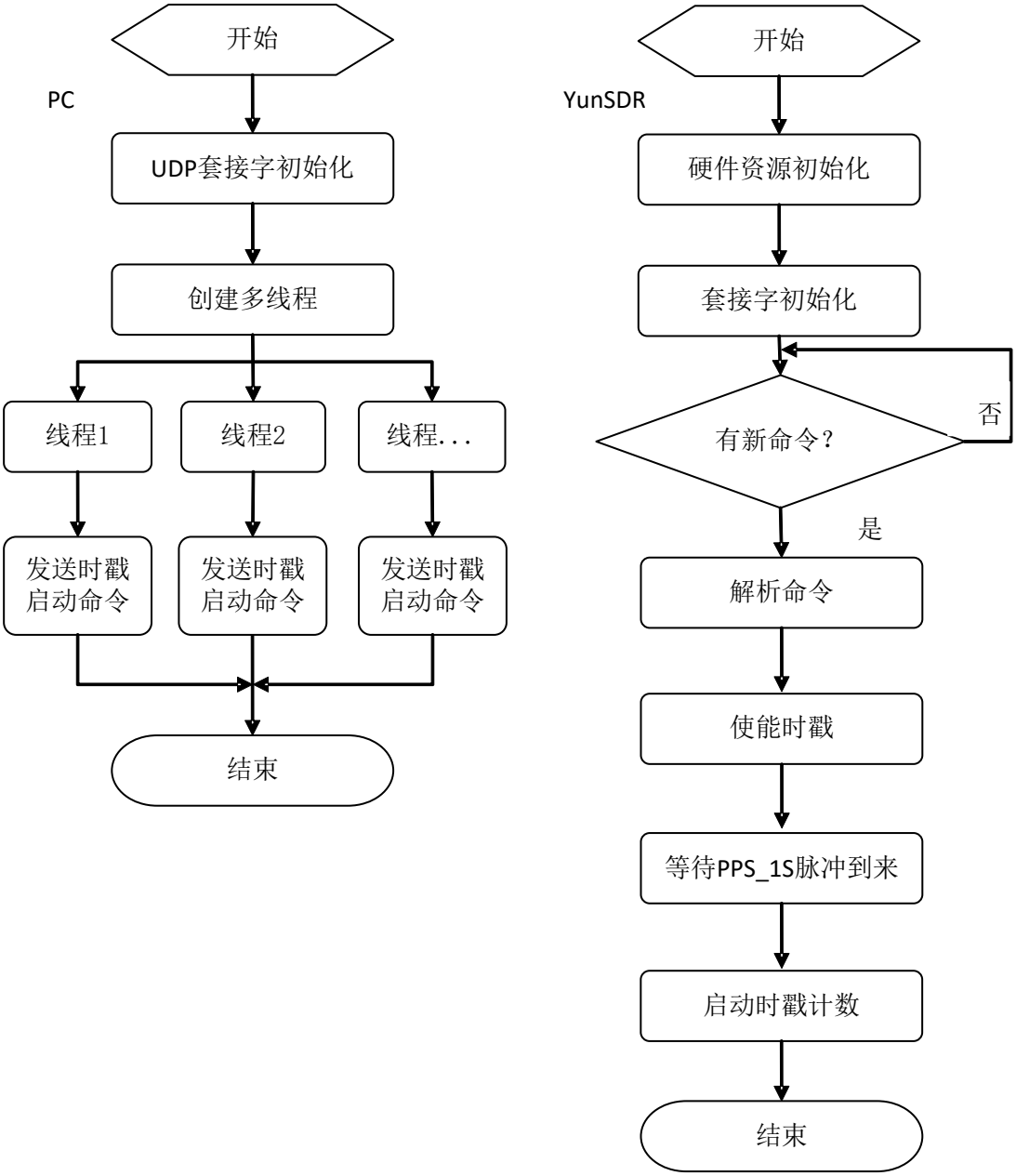


图 8 多终端同步启动流程

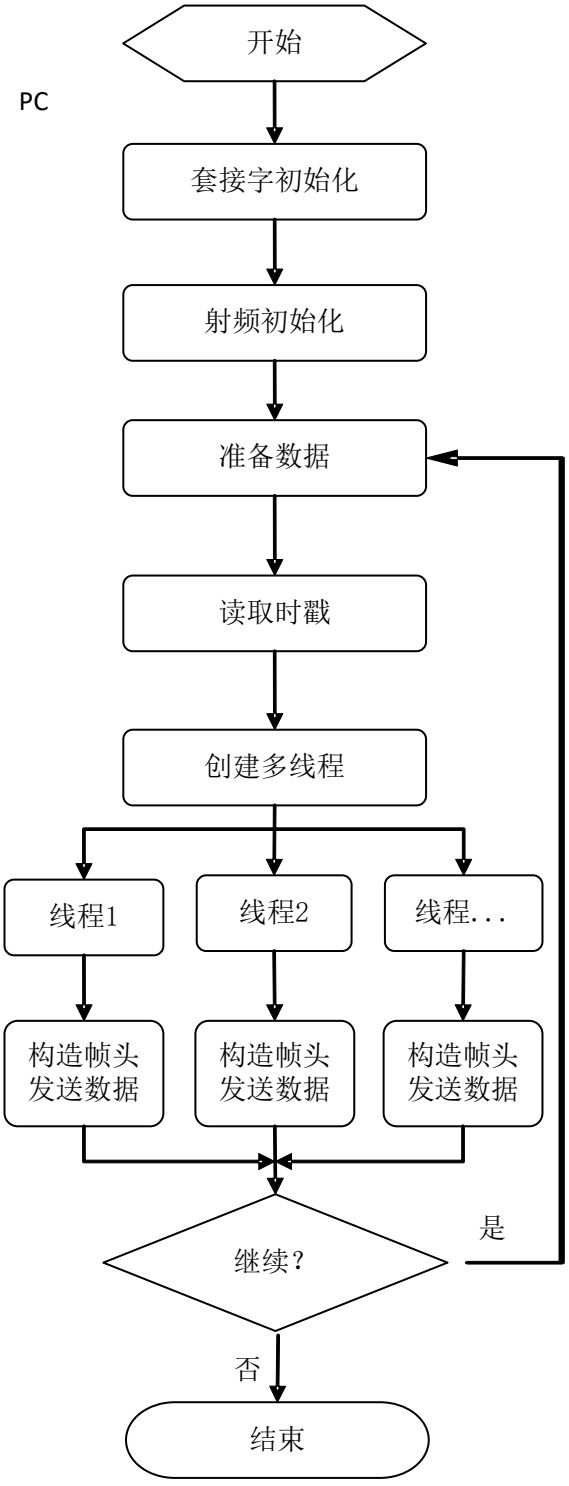


图 9 多终端同步发送流程

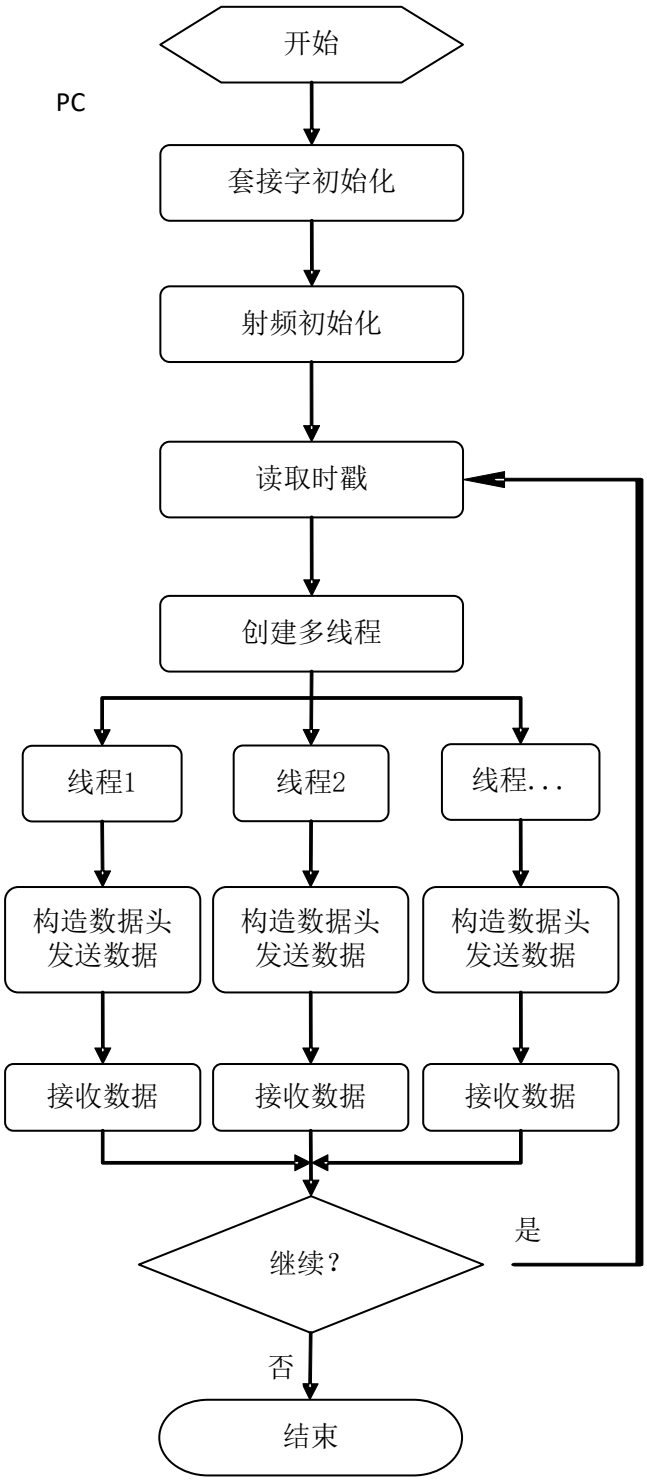


图 10 多终端同步接收流程

# 4 matlab 参考设计

时戳控制系统提供 matlab 参考设计，可以完成若干个 Y310(S)模块的时戳同步检测功能，此外 matlab 参考设计将时戳控制与数据收发功能封装了若干个子函数(function)，方便用户进行调用。本节对 matlab 参考设计与子函数的使用进行描述。

## 4.1 matlab 文件结构

matlab 参考设计包含 ad9361 配置，时戳控制，IQ 数据收发，产生数据源，接收数据处理等几个功能。其中 IQ 数据的生成包括单音信号和 IEEE802.11n 2x2 数据源文件的读取两种。接收数据分析也包括单音信号的画图，与 IEEE802.11n 2x2 数据的同步检测功能。

表格 7 matlab 程序文件结构

| 文件名 | 描述 | 备注 |
|---|---|---|
| tranciever_test.m | 测试顶层文件 | |
| tranciever_test_ts_disable.m | 测试顶层文件时戳关闭模式 | |
| ad9361_init.m | Ad9361 初始化文件 | Ad9361 所有工作参数的配置 |
| timestamp_ctl.m | 时戳控制 | 包括开始、复位、读取功能 |
| trans_data.m | 发送数据控制 | 将输入的 IQ 数据发送给板卡 |
| recv_send_cmd.m | 发送接收数据的命令 | 将接收指令包括时间长度等信息配置给板卡 |
| recv_data.m | 获取板卡接收的 IQ 数据 | 将上面函数指定的长度与时间的数据读从板卡读取到 matlab 中 |
| generate_tone.m | 产生单音信号 IQ 数据 | 可选单通道或双通道数据 |
| generate_tone_nogap.m | 产生单音信号 IQ 数据不补零 | |
| tone_plot.m | 接收单音信号画图 | |
| read_mcs.m | 读取 IEEE802.11n 两天线文件 | |
| mcs_sync.m | 接收到的 IEEE802.11n 数据同步顶层模块 | |
| rx_search_packet_long.m | 粗同步 | |
| rx_search_packet_short.m | 精同步 | |
| set_sim_consts.m | IEEE802.11n 系统参数 | |
| tx_freqd_to_timed.m | 同步所需的 IFFT 变换 | |

## 4.2 matlab 函数分析

本小节对以上的子函数进行描述。

#### 4.2.1 顶层模块描述

顶层模块中包括若干个系统工作参数指令如下：

指定模块 IP 地址，每个 Y310(S)的外壳上都有标出 IP 地址，此 IP 地址在 IF 卡的镜像 init.sh 文件中可以修改。用户要确保连接到同一个交换机的每个 Y310(S)的 IP 和 MAC 均不相同（出厂设置同用户同批次订购的模块 IP 和 MAC 不相同）。定义如下：

```
mod1='192.168.1.10';
mod2='192.168.1.11';
mod3='192.168.1.12';
mod4='192.168.1.13';
```

表格 8 顶层模块参数定义

| 名称 | 描述 |
|---|---|
| localport | 本地端口号，可以任意指定，如果出现程序非法结束会导致端口被占用的情况，请指定另外的本地端口号 |
| rx_sample_len | 接收 IQ 数据采样点长度，单通道模式必须为偶数，双通道任意 |
| tx_chan | 发送通道选择，1=通道一；2=通道二；3=双通道 |
| rx_chan | 接收通道选择，1=通道一；2=通道二；3=双通道 |
| ref_sel0 | 参考选择，内置 GPS 版本选择 ref_sel0 |
| ref_sel1 | 参考选择，外部时钟同步版本选择 ref_sel1 |
| time | 发送或接收的延时时间 |
| delay | 发送和接收的时间间隔 |
| rx_delay | 接收时间戳延时 rx_delay =time*samplerate |
| tx_delay | 发送时间戳延时 tx_delay =time*samplerate+delay |

发送时刻=time_read+tx_delay

接收时刻= time_read+rx_delay

#### 4.2.2 AD9361 初始化函数

```
ret=ad9361_init (mod1 ,localport,tx_chan,rx_chan,ref_sel0);
```

输入参数：

- mod1，对应的模块 IP 地址
- localport，本地的端口号
- tx_chan，发送通道选择

- rx_chan，接收通道选择
- ref_sel0，如果模块是内置 GPS 版本(Y310S)，写 ref_sel0；如果是 Y310 版本，写 ref_sel1

输出参数

- ret，返回值，如果运行成功返回字符串“ad9361 initial ok”

子函数内部参数如下，如果用户需要单独配置某个参数，可以将相应的网络命令取出单独配置。

表格 9 初始化模块配置参数

| 名称 | 说明 |
|---|---|
| rcount，ncount | 使用同步时钟鉴相器参数，计算方法，外部时钟输入/rcount=26/ncount。如果外部参考是10MHz，则rcount=10，ncount=26 |
| vco_cal | =1 内部参考通过ad9361的辅助dac校准频率；=0内部参考通过外部参考时钟校准频率 |
| aux_dac1 | 辅助dac输出电压，mV单位0-3000mV |
| fdd_tdd | 输出射频端口模式:=1 FDD模式，收发射频口全部工作；=0 TDD模式只有TRX射频接口工作 |
| trx_sw | 收发切换，当FDD模式时=1；当TDD模式时=1 发送，=0 接收 |
| samp | 采样率，单位Hz，范围2.5MHz-61.44MHz |
| bw | 带宽，单位Hz，小于56MHz，收发带宽可以分别调整 |
| freq | 射频频点，单位Hz，70MHz-6GHz，收发频点可以分别调整 |
| tx_att1 | TX第一通道衰减，单位mdB(豪dB)，0-90dB |
| tx_att2 | TX第二通道衰减，单位mdB(豪dB)，0-90dB |
| rxgain1 | RX第一通道手动增益值，1-73dB |
| Rxgain2 | RX第二通道手动增益值，1-73dB |

请注意，接收的增益设定值，是根据频段的不同分段校准的，在 FPGA 的程序中存有不同频段对应的接收增益：200~1.3GHz，1.3~4GHz，4~6GHz。如果频点变化范围超过当前频段的范围，需要在配置频点后设定 rxgain。

### 4.2.3 时戳控制函数

```
ret=timestamp_ctl (mod1 ,localport,'reset');
ret=timestamp_ctl (mod1 ,localport,'start');
ts1=timestamp_ctl (mod1 ,localport,'readt');
```

时戳控制分为三个不同功能的子指令，reset复位板卡时戳，start开始时戳计数，readt将板卡的时戳读入matlab上位机。在Y310板卡启动后时戳处在复位的状态，为了确保开始时戳的同步，在第一次启动时戳之前首先再次复位时戳，之后启动时戳计数。板卡在接收到启动命令后等待1pps脉冲到来开始计数，计数周期是采样率，默认采样率40MHz，计数宽度64bit，完整的计数循环周期= $2^{64} \div (40\times10^{6}) \div 60 \div 60 \div 24 \div 365 \approx 14624$年 。

### 4.2.4 发送数据函数

```
ret=trans_data (mod1 ,tx_chan,loop,txts_en,txtime,txdata);
```

输入参数：

- mod1，指定板卡 IP 地址
- tx_chan，发送通道选择
- loop，0=发送一次；1=本帧数据循环发送。当设置循环发送模式时，发送时戳使能 txts_en 必须关闭。
- txts_en，发送时戳使能：1=使能时戳，0=时戳无效
- txtime，发送时刻。当时戳使能时，发送端会在此时刻将 IQ 数据通过 AD9361 发出
- txdata，待发送数据。格式 txdata(1:length,antenna_num)。按照每个通道的 IQ 数据 16bit 量化（-32768~32767）

如果在 Y310(S)收到的发送数据时戳小于模块的计数时戳，时戳超时，模块会丢掉这帧数据等待下一个发送数据包。

#### 4.2.5 接收命令函数

```
[ret,data_link1 ]=recv_send_cmd (mod1 ,rx_chan,rxts_en,rxtime,rx_sample_len);
```

输入参数：

- mod1，指定板卡 IP 地址
- rx_chan，接收通道选择
- rxts_en，接收时戳使能：1=使能时戳，0=时戳无效
- rxtime，接收时刻。当时戳使能时，接收端会在此时刻将 IQ 数据接收传给上位机
- rx_sample_len，接收数据采样点长度。

#### 4.2.6 接收数据函数

```
[IQdata1 ,head1 ,rxts1 ]=recv_data (data_link1 ,rx_chan,rx_sample_len);
```

输入参数：

- data_link1，由 recv_send_cmd 函数输出
- rx_chan，接收通道选择
- rx_sample_len，接收数据采样点长度。

输出参数：

- IQdata1,接收到的 IQ 数据，格式与发送端的数据相同
- head1，时戳。FF000080 表示正常接收时戳有效；FF0000AB 时戳有效，但是接收数据时戳超时；FF00009D 接收时戳无效

- rxts1，接收数据的时刻(时戳)。

# 5 分布式同步方案测试

## 5.1 设备安装与连接

如上文所述，时戳同步硬件有两种版本 Y310 与 Y310S。Y310 没有内置 GPS，Y310s 含有内置 GPS 模块。Y310S 主要应用于外场分布式同步系统（可以安装 GPS 天线接收卫星同步信号）；Y310 可以配合时钟分配模块（YUNGCD-10）进行同步，可以应用于室内等 GPS 信号受限的场合。时钟分配模块本身内置有 GPS 模块，所以 Y310 与 Y310S 可以混合组成分布式同步系统。测试系统除了 YUNSDR 模块还需要千兆网交换机一个，如果需要大功率测试，可以选配功放模块。测试框图如下：

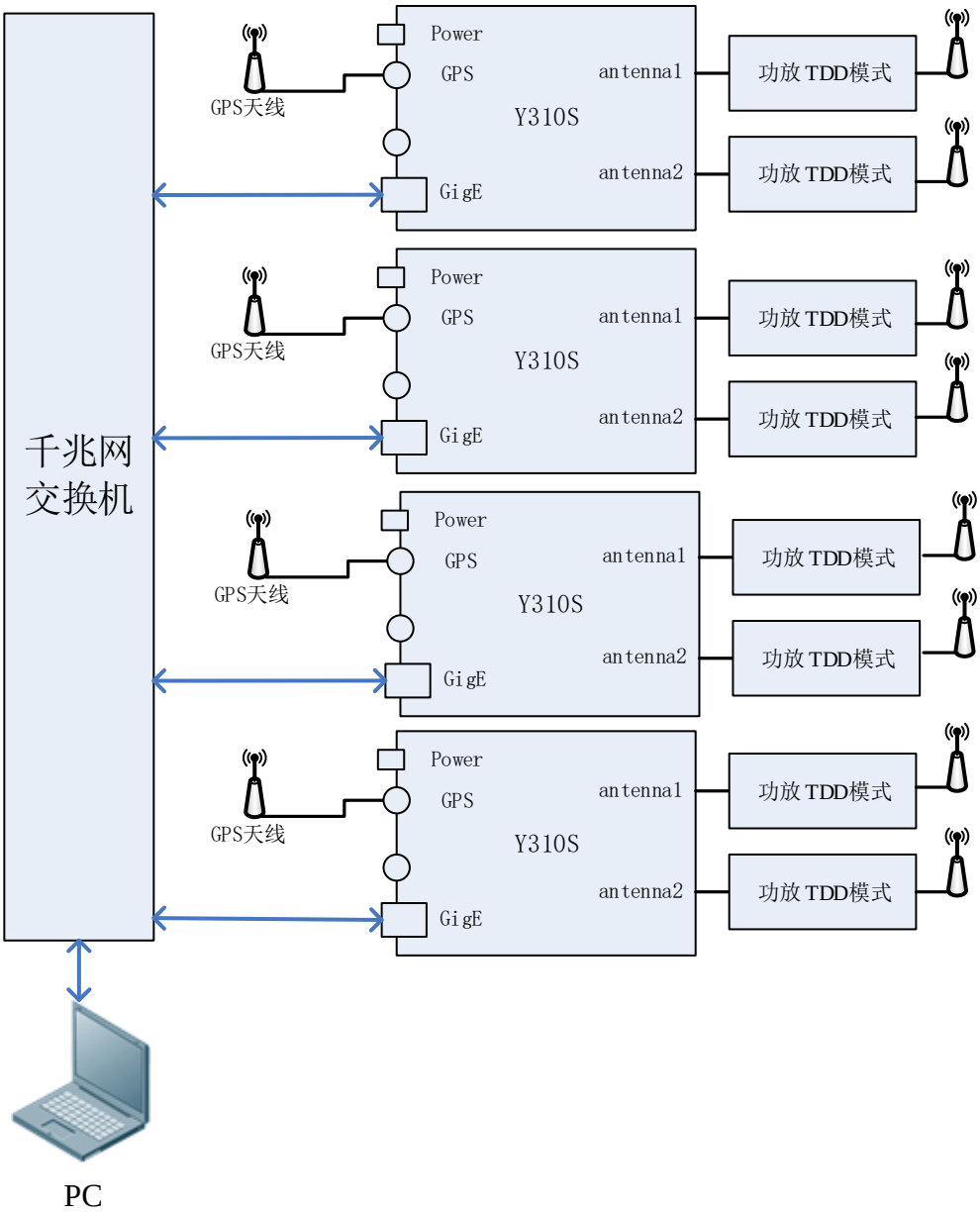


图 11 Y310S 时戳同步系统

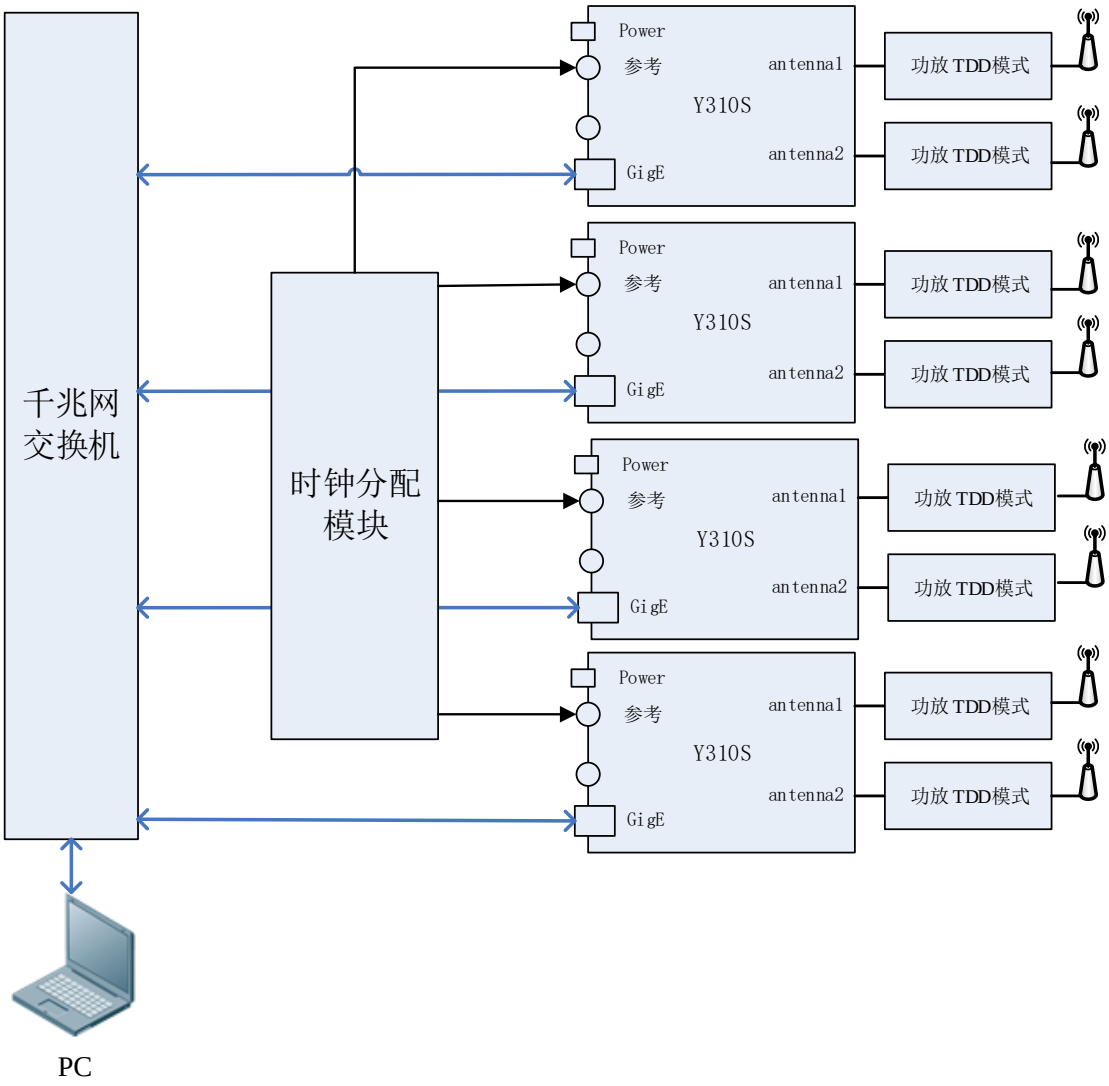


图 12 Y310+时钟分配同步系统

## 5.2 Y310 安装与连接


图 13 时戳同步级联连接方式

如图所示通过网线连接 Y310(S)至路由器，连接设备的 USB 线用于供电。如果选用 Y310S 则连接设备的 GPS 天线用于同步；如果选用 Y310 则分别连接 10MHz 外部参考和外部 1PPS

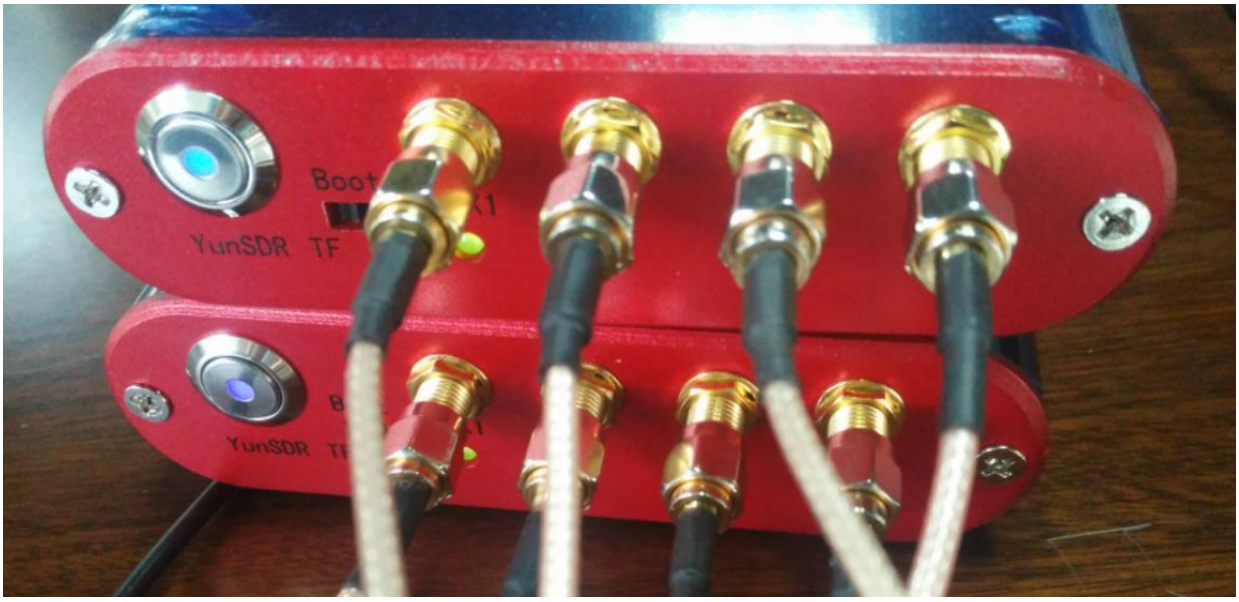


图 14 两台设备的收发交叉相连

## 5.3 C 上位机测试步骤

PC 客户端

timestamp_tcp.exe

参数

```
Usage: timestamp_tcp.exe -strdeh
-h                  print help message
-t <0 .. 10000>     start RX & TX test
-s                  configure yunsdr
-e                  enable timestamp
-d                  disable timestamp
-r                  read timestamp
```

1. 设备连接完成并上电等待十几秒，设备会正常启动，每台设备的四个并排绿色 LED 灯会以 1HZ 的频率闪烁，当 GPS 锁定卫星后，两台设备的 8 个 LED 会同步闪烁，然后在 PC 执行 timestamp_tcp.exe -s 配置射频，首先客户端会检查与设备的连接，会向两台设备发送 ICMP，如果得到正确响应，则继续执行后续过程，如果未响应有可能是设备 mac 地址未发现，可重复执行此条命令，如果仍然未响应，则检查路由器或 PC 的网络设置后重试
2. 在 GPS 锁定卫星后，执行 timestamp_tcp.exe -e 使能两台设备的时戳计数，此时因为两台设备已锁定卫星并同步输出 PPS 信号,并且板卡程序会以 GPS 的 PPS 输出信号为基准来启动时戳计数器，所以两台设备的计数器将同步计数
3. 执行 timestamp_tcp.exe -t 100 命令可以进行 100 次同步收发测试，会在程序当前目录各保存 100 份接收的数据
4. 执行 matlab 校验程序校验收发是否同步
5. 执行 timestamp_tcp.exe -r 命令可以读取当前两台设备的时戳
6. 执行 timestamp_tcp.exe -d 命令可以复位当前两台设备的时戳计数

测试结果如下

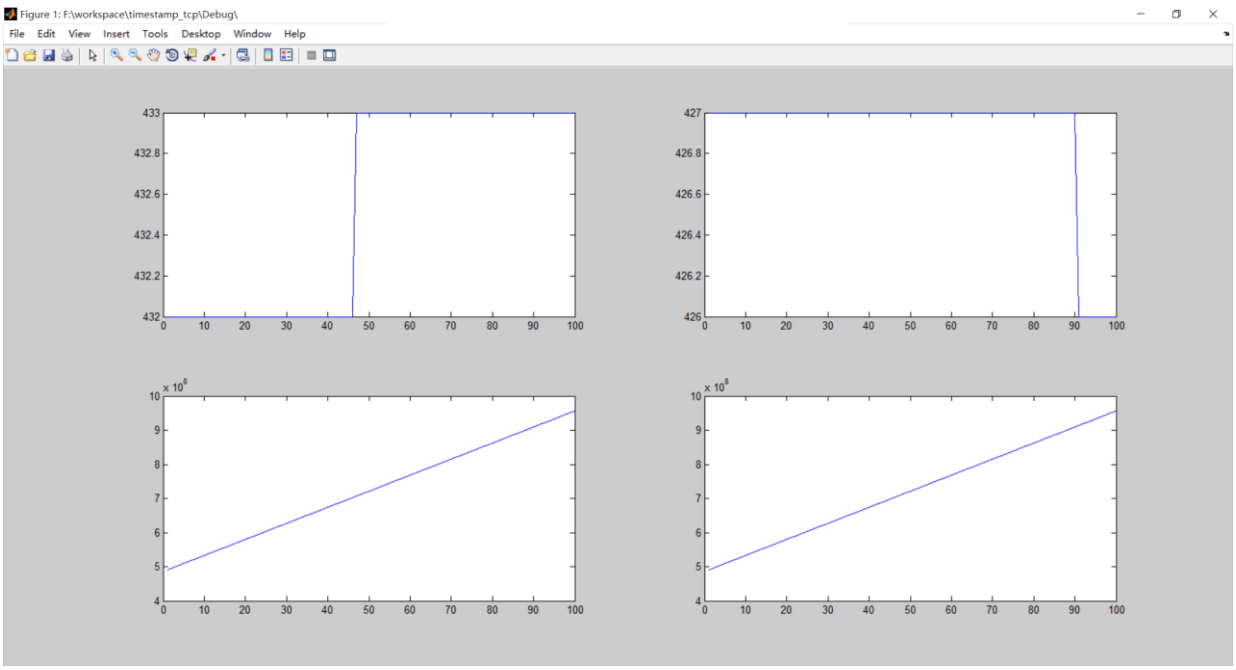


图 15 时戳同步(matlab 校验结果)

4X4MIMO 测试结果

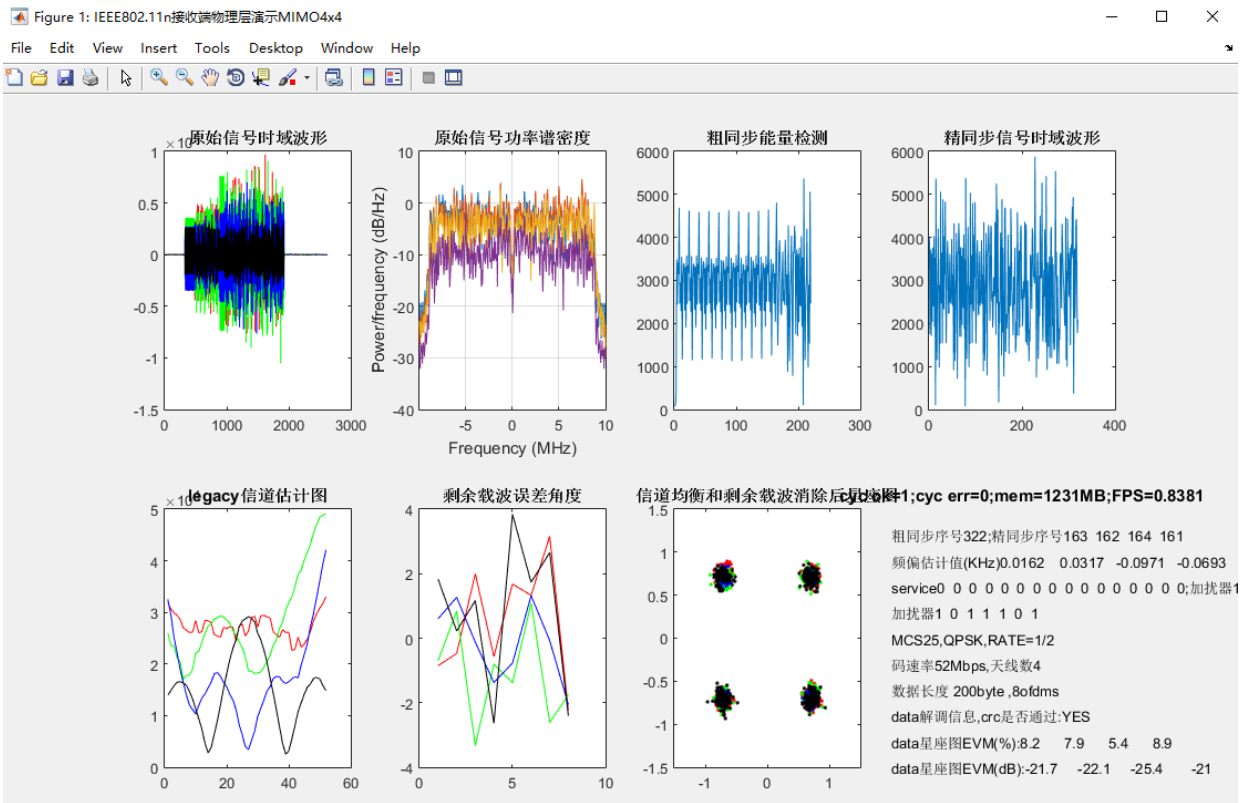


图 16 解调结果

## 5.4 matlab 上位机分布式同步测试

使用 ts_matlab_release 测试工程的 tranciever_test.m，采用 generate_tone.m 数据源。通过 matlab 测试各个 Y310(S)是否同步，方法是使发送端在规定是时刻发送单音信号。接收端在此时刻之前若干个时戳启动接收。将所有模块收到的 IQ 数据显示，计算收到单音信号的起始点，就可以判断每个 Y310(S)是否同步。

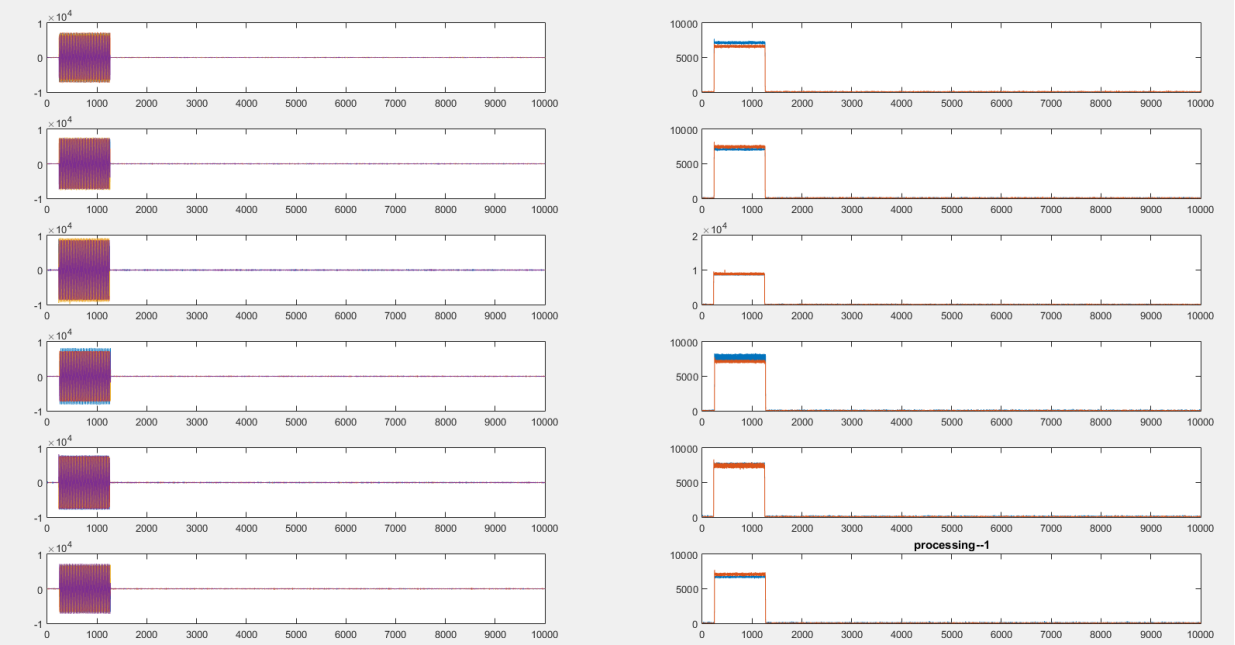


图 17 单音信号测试结果

收发延时 200 个周期，左面显示 IQ 时域信号，右面显示 IQ 的模值。竖排显示一共六个 Y310(S)模块的测试结果。实际延时大于 200 个时戳是因为射频链路和 FIR 滤波器的延时。放大显示如下：

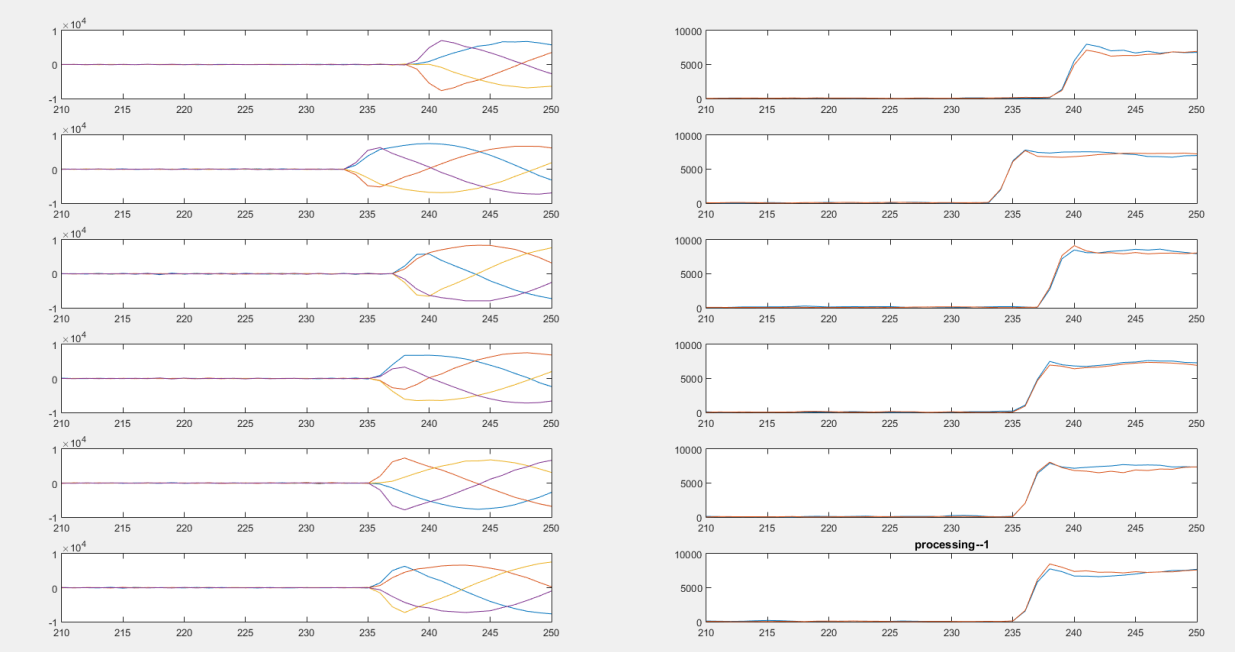


图 18 单音信号接收局部放大

可以看到每个模块接收数据的起始位置都在 235 左右，时戳差别不超过 5 个。这是因为 GPS 的 1PPS 秒脉冲精度在百纳秒级。

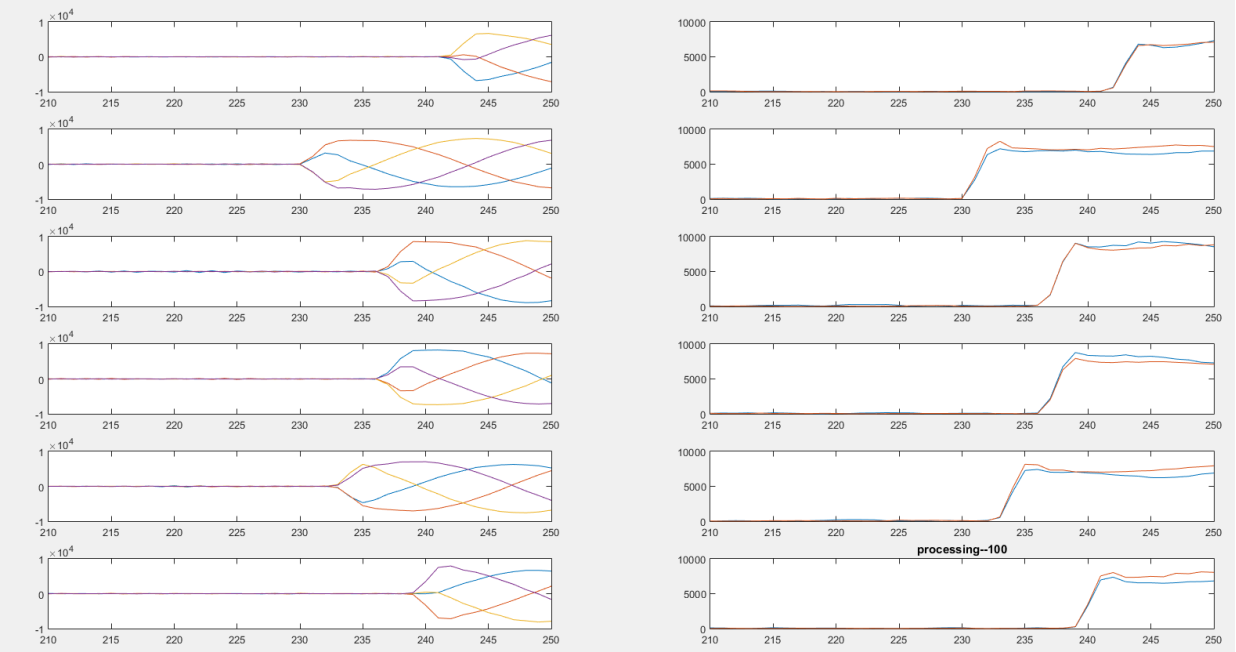


图 19 一百次循环的运行结果

## 5.5 IEEE802.11n 信号收发同步检测

本例程测试接收到的 IEEE802.11n 信号进行帧同步，在运行若干个循环后，matlab 参考程序可以定量分析帧同步的起始位置。使用 ts_matlab_release 测试工程的

tranciever_test.m，采用 read_mcs.m 读取 IEEE802.11n 标准 IQ 数据源。将每次帧同步的起始点画图，对比每个板卡的时戳同步的效果。下图为 100 次循环的运行结果，区间范围在 230~240 之间，由于 GPS 的精度会有 10 个左右采样点的误差。

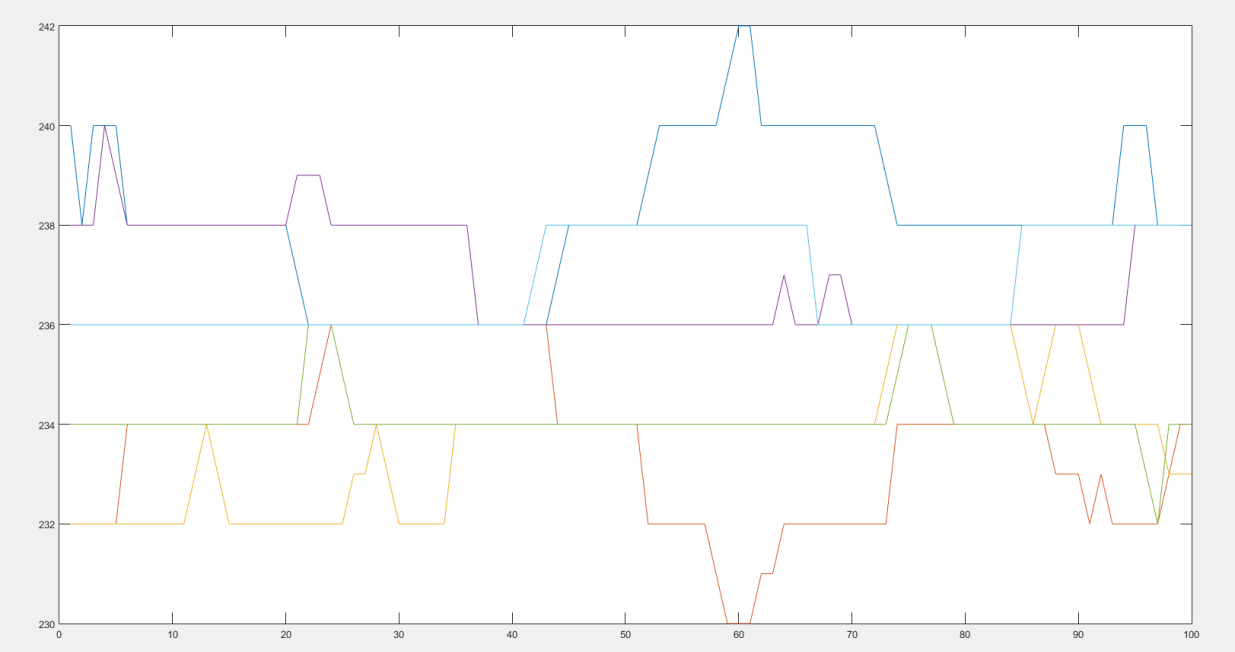


图 20 100 次循环 IEEE802.11n 同步检测结果