产测环境搭建说明

1 环境需要

1.1 硬件

产测一体板，

杜邦线，

电脑（产线替换为主MCU），

mini口USB线。

注意：所有连接引脚均为pin2pin，不需要转化tx rx

1.2 软件

产测上位机软件（产线替换为主MCU），

串口调试助手sscom等。

1.3 名词解释

golden：产测一体板的别称

DUT：被测设备，即需要接受监测的芯片/模块

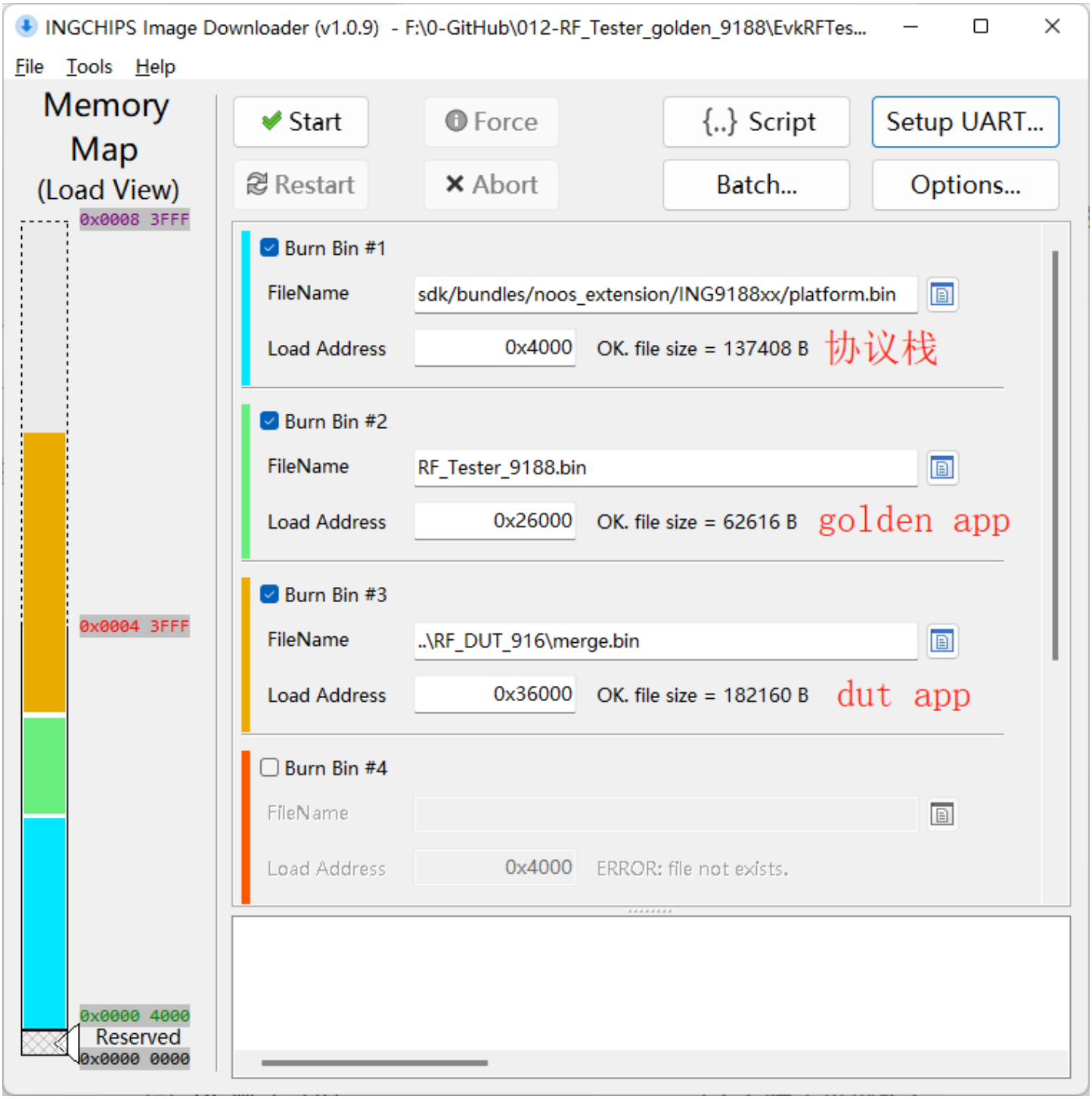
2 测试过程

从上位机发送测试开始，到golden反馈测试完成，中间大致分为：烧写测试固件、射频测试、出厂固件烧写三个打的过程。

2.1 烧写测试固件

上位机发送测试开始之后，首先需要给dut烧写用于测试的固件。

在环境部署阶段，golden需要烧写三个bin文件，如下图所示：



其中协议栈和golden app是golden使用的固件，dut app（merge.bin）是存储在goldenFlash里，测试时会通过串口烧写到DUT里的固件。

因此，如果后续有固件更新，要分清更新的是哪个bin，进行替换即可（当然也可一起全部更新）。

注意：更新golden固件的时候需要摘下amo和dut保证烧录的串口不被干扰，否则有可能握手失败

2.2 射频测试过程

射频测试过程分5个环节：

- TX cnt

DUT发射一定包数（比如100包），golden接收，接收超过一定阈值认为测试通过。

- TX power

DUT以0dbm发射，golden接收并检测信号强度，强度超过某个阈值（比如-35dbm）认为测试通过。

- RX cnt

golden发射一定包数（比如100包），DUT接收，收到包数超过一定阈值认为测试通过。

- RX power

golden以0dbm发射，DUT接收并检测信号强度，强度超过某个阈值（比如-40dbm）认为测试通过。

- Freq offset

DUT发射信号，golden测试其频偏，如果发现频偏过大，golden会通过软件进行调试，并把调试值写入Flash固定地址。

当前调试的频偏配置存储在如下Flash地址处，可以根据需要调整。

```
代码块
1   #define FLASH_FREQ_ADDR 0x0207F000
```

存在一定可能软件无法将频偏调到要求范围内，就需要通过调整24M晶振的电容来调节频偏。

2.3 出厂固件烧录

测试完成后，烧录出厂固件，此步骤是可选的：

- 如果选择不烧录出厂固件，则测试完成后会直接上传测试结果。
- 如果选择烧录出厂固件，则烧录完成后再上传测试结果。

3 通讯协议

上位机/主MCU与golden之间设计了一套串口通讯协议，用来配置测试流程和获取测试结果。

上位机的每一条指令从机都需要进行对应的回复，主机需要有超时重发机制。

3.1 协议格式

一条完整的数据包由**帧头**、**类型**、**命令长度**、**命令**、**crc**、**帧尾**这6个字段构成，如下图所示：

命令（上位机发出）

	帧头 2byte	类型 1byte	命令长度 1byte	命令 1byte	CRC 2byte	帧尾 2byte
举例	0x4A 0x45	0x04	0x1	0xC5	0xD5, 0x7F	0x4E 0x44
说明	HE的十六进制	四类基本命令	1-255	根据不同命令长度不同	前面所有字段的crc值	ND的十六进制

命令回复（golden回复）

	帧头 2byte	类型 1byte	命令长度 1byte	命令 1byte	结果 1-255byte	CRC 2byte	帧尾 2byte
举例	0x4A 0x45	0xF4	0x2	0xC5	0x0	0xE0, 0x57	0x4E 0x44
说明	HE的十六进制	四类基本命令	1-255	根据不同命令长度不同	根据具体命令解析结果	前面所有字段的crc值	ND的十六进制

字段说明：

- 类型：产测一体板内部有多种协议，**类型**字段用于区分不同的协议。对于上位机/主MCU的通讯来说，上位机/主MCU发送给产测一体板的类型固定为0x4，产测一体板回复给上位机/主MCU的类型固定为0xF4。
- 命令长度：后面的**命令**和**结果**两个字段的长度。
- 命令：具体的命令类型，如测试开始、通讯检测等，详见2.2。
- crc：计算从帧头到crc前所有字段的crc值，计算参考代码如下：

代码块

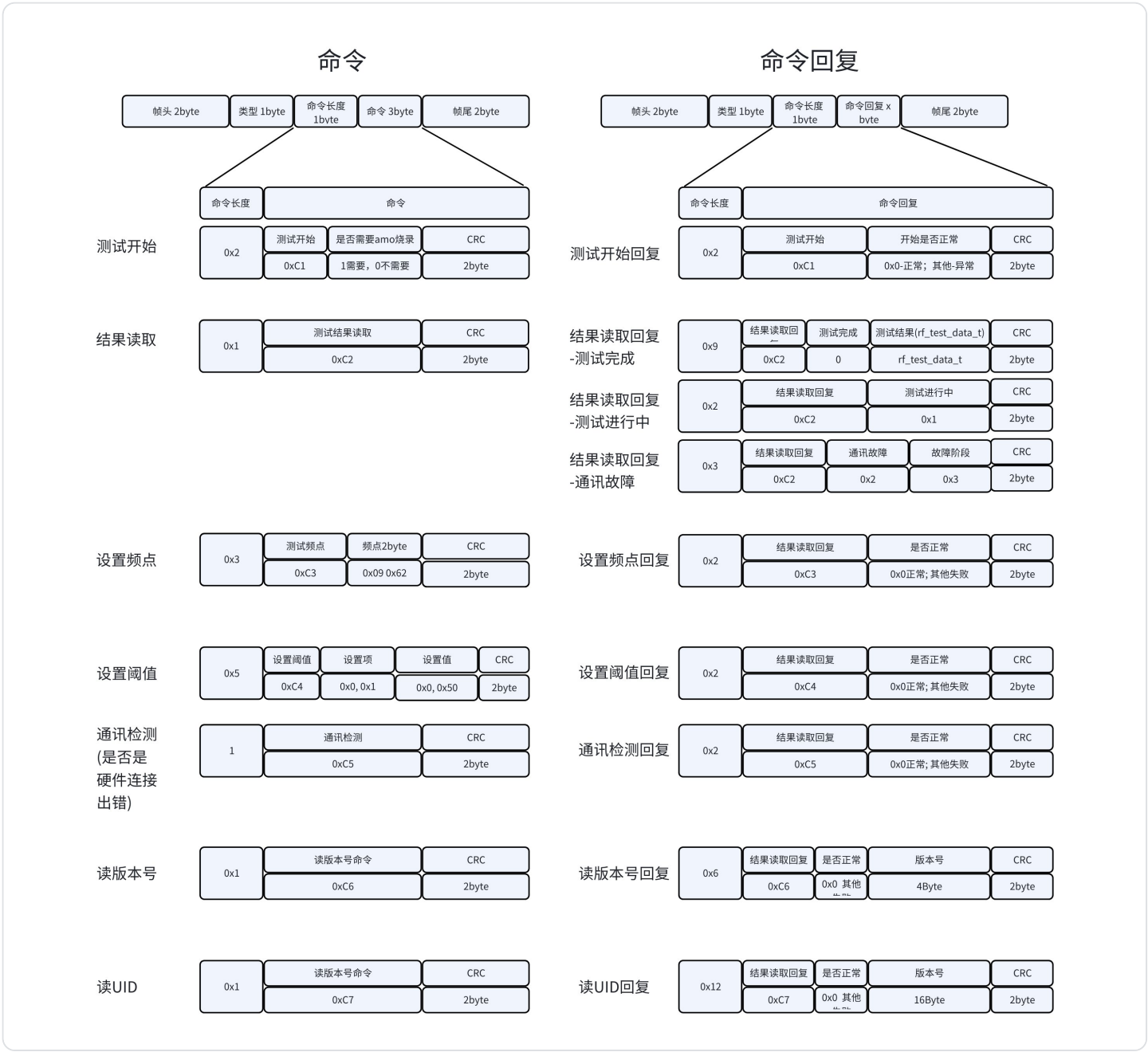
```
unsigned short check_crc16(char *puchMsg , unsigned short usDataLen)
{
    unsigned char uchCRCHi = 0xFF ; /* high byte of CRC initialized */
    unsigned char uchCRCLo = 0xFF ; /* low byte of CRC initialized */
    unsigned char uIndex ;          /* will index into CRC lookup table */

    while (usDataLen--)     /* pass through message buffer */
    {
        uIndex = uchCRCHi ^ *puchMsg++ ; /* calculate the CRC */
        uchCRCHi = uchCRCLo ^ auchCRCHi[uIndex];
        uchCRCLo = auchCRCLo[uIndex] ;
    }

    return (uchCRCHi << 8 | uchCRCLo) ;
}
```

3.2 不同的命令

目前实现的命令如下：



3.2.1 测试开始-C1

- 命令

表示上位机开启一轮完整测试，有一个参数：是否需要amo，长度1个字节。

1：带amo烧录，测试完成之后使用amo烧录出厂固件，需要连接阿莫烧录器。

0：不带amo烧录，测试完成之后直接结束。

- 回复

表示golden收到了上位机的开始测试命令，有一个参数：是否正常开始

0：正常地开启了测试

其他：开启测试失败

3.2.2 结果读取-C2

- 命令

主动读取测试结果，无参数。

- 回复

回复有三种情况，会有不同的参数。

1）测试完成，会有两个参数：测试完成（0x0）和测试结果。

测试结果使用下面的数据结构存储，前面5个byte分别表示测试的各项结果，第6个byte的5个bit表示各项测试是否通过。

代码块

```
typedef struct {
    uint8_t tx_cnt;
    uint8_t rx_cnt;
    int8_t tx_power;
    int8_t rx_sen;
    int8_t freq_offset;
    struct {
        bool tx_cnt_pass    : 1;  // 1-bit flag for pass/fail
        bool rx_cnt_pass    : 1;
        bool tx_power_pass  : 1;
        bool rx_sen_pass    : 1;
        bool freq_offset_pass : 1;
    } status;
} rf_test_data_t;
```

2）测试进行中，有一个参数：测试进行中（0x1）。

3）通讯失败，有两个参数：测试失败（0x2）和失败阶段。

失败阶段通过下列数据结构定义，可以用来定位出问题的位置。

这里的失败通常是硬件线路问题，如串口线松了，供电问题等。如果是发生在0-5，则DUT和golden的连接或者DUT的供电出了问题。如果发生在6阶段（阿莫烧写），则是与阿莫相关的连接、供电出了问题。

代码块

```
typedef enum {
    COMM_ERR_STAGE_BURNING_TEST       = 0,     //burn test image
    COMM_ERR_STAGE_TX_CNT_TEST        = 1,     //tx count test
    COMM_ERR_STAGE_TX_POWER_TEST      = 2,     //tx power test
```

```
    COMM_ERR_STAGE_RX_CNT_TEST        = 3,     //rx count test
    COMM_ERR_STAGE_RX_SEN_TEST        = 4,     //rx sensitivity test
    COMM_ERR_STAGE_FREQ_OFFSET_TEST   = 5,     //frequency offset test
    COMM_ERR_STAGE_BURNING_VENDOR     = 6,     //burn vendor test image
} RF_comm_err_stage_t;
```

- 说明

golden完成整个测试过程后会主动发送测试结果，主机也可以主动发送结果读取，但是如果测试还没有完成，就需要过一段时间再次读取。

3.2.3 设置频点-C3

- 命令

设置射频测试使用的频点。有一个参数，两个字节，表示频点，范围2402-2480。

如果上位机进行了设置，就使用配置的频点测试，如果没有调用此命令进行设置，则会使用随机生成的2402-2480内的一个频点进行测试。

- 回复

回复设置结果，有一个参数：0-设置成功；其他-设置失败。

- 说明：

如果产线内有多个测试工位同时执行测试，则要对使用的频点进行规划，如果两个临近工位同时使用相同频点测试，会互相干扰，导致两边的测试数据同时变差。

如果同一时间只有一个工位进行测试，则要避开环境中干扰比较大的频点。

如果有条件，将golden和DUT屏蔽在一个空间内可以获得最好的测试效果。

3.2.4 设置阈值-C4

- 命令

设置发送总包数（tx cnt和rx cnt阶段使用）和测试的阈值。有两个参数：

设置项，长两个字节。有0x1-0x5五个类别，分别表示**总包数**，**tx_power阈值**，**tx_cnt阈值**，**rx_sen阈值**，**rx_cnt阈值**。

代码块

```
typedef enum : uint8_t
{
    PACKET_CNT          = 1,
    TX_POWER            = 2,
    TX_CNT              = 3,
    RX_SEN              = 4,
    RX_CNT              = 5
```

```
8   }threshold_switch;
```

设置值，实际配制的值，有两个字节。比如总包数设置100，tx_power设置-35。

- 回复

回复设置结果：0-设置成功；其他-设置失败。

- 说明

1）TX_CNT和R_CNT的阈值表示收到的包数，收到一定包数以上测试合格，比如设置阈值80，则80包以上合格

2）TX_POWER和RX_SEN都是rssi，单位是dbm，收到的信号强度强于阈值算测试合格

3）PACKET_CNT是测试总包数，建议设成100，设置过大会增加测试时间，且需要对应地调整阈值。

3.2.5 通讯检测-C5

- 命令

检测通讯是否正常，无参数。

- 回复

回复通讯检测的结果，带1个参数：0-正常；其他-不正常

- 说明：当前版本通讯检测只能检测上位机与golden通讯是否正常，无法检测golden和DUT是否连接正常，以及golden和amo是否连接正常，后续版本会进行优化。

3.2.5 读版本号-C6

- 命令

读取当前golden固件的版本号，不带参数。

- 回复

回复读取到的版本号，有两个参数：

是否正常：0-正常；其他-异常

版本号：golden固件版本号的ASCII码，版本号的格式为V1.0.0。

3.2.6 读取UID-C7

- 命令

读取当前测试的DUT的UID，不带参数

- 回复

回复读取到的当前DUT的16byte的UID，有两个参数：

是否正常：0-正常；其他-异常

UID：DUT的UID，HEX形式

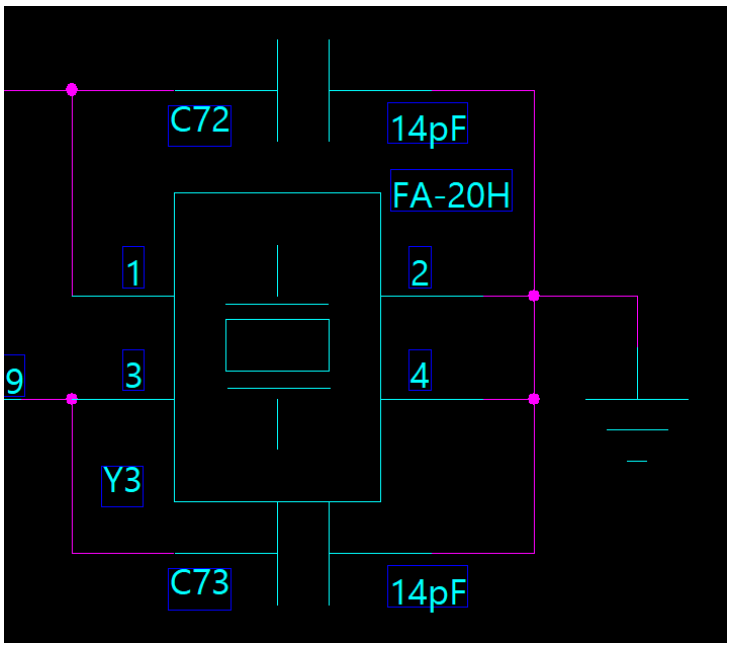
4 环境搭建

4.1 频偏测试

射频测试的最后一个过程——频偏测试过程，golden通过接收DUT的信号来测DUT的频偏，需要golden本身频偏正常，因此每一块golden的频偏都需要单独测试并调整。

测试过程为通过烧录SDK自带的BQB RF Test App，在频谱仪上查看频偏，并通过调整匹配电容（下图中的C72，C73）来进行调整。

每一块golden的频偏都要调整到0附近。



此过程可由原厂协助完成，细节不再赘述。

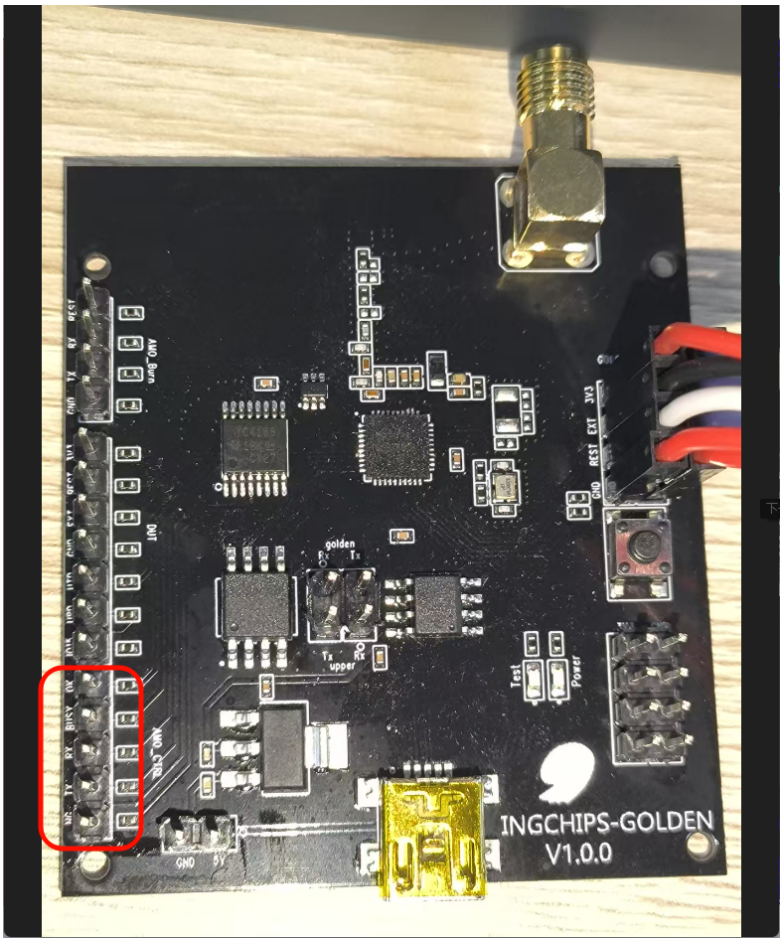
产测一体板的频偏每年都要进行一次测试与校准。

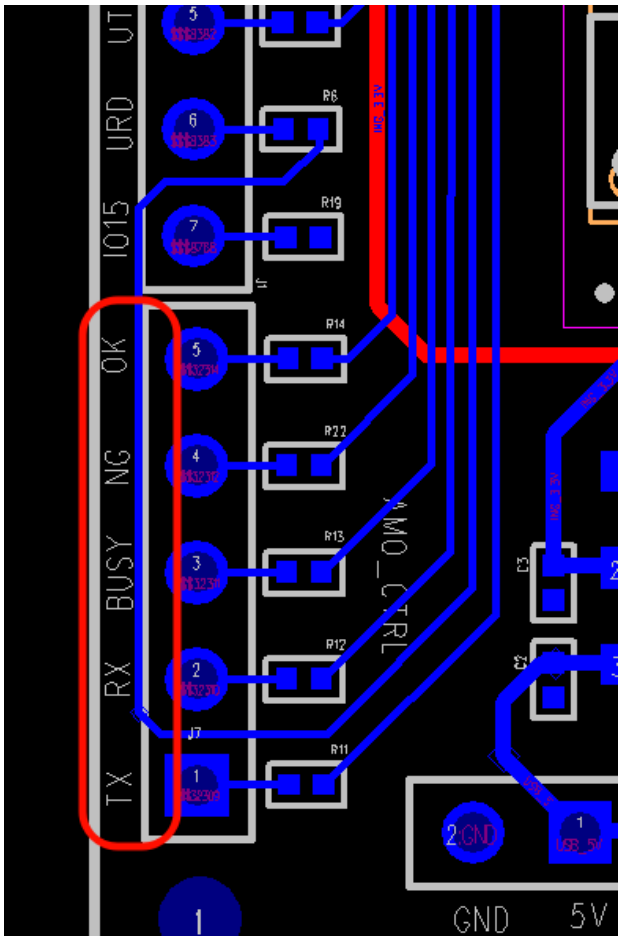
4.2 连线

如下图所示，必须连接的包括：

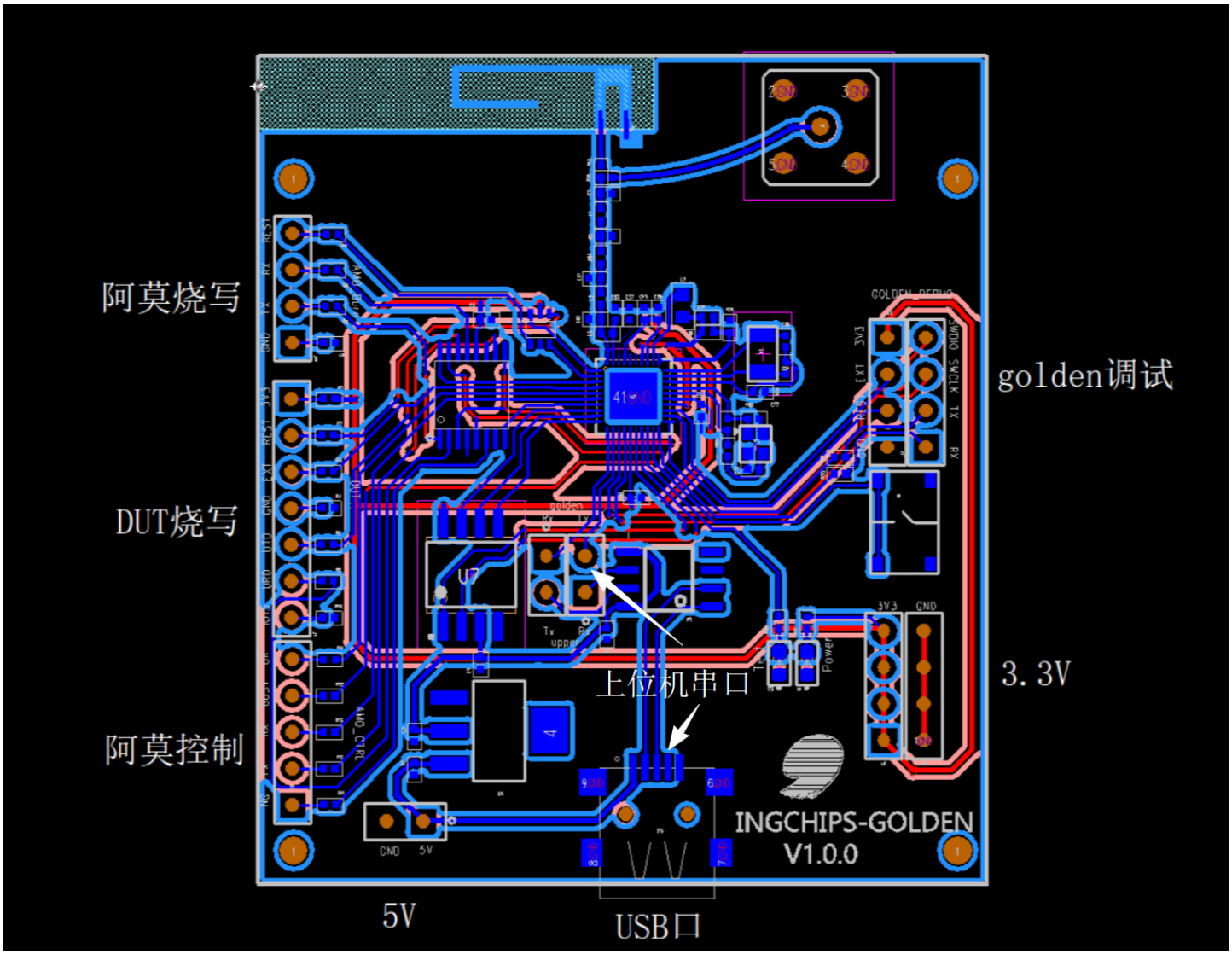
- 阿莫烧写：连接阿莫烧录器的烧写引脚，包RST，RX，TX，GND，共4根线
- DUT烧写：连接DUT的烧写引脚，包括3V3，RST，EXT，GND，TX，RX，共6跟线
- 阿莫控制：连接阿莫烧录器的控制引脚，包括OK，NG，BUSY，TX，RX，共5根线

注意：此次硬件丝印有误，下一版会改为正确(右侧为正确的amo控制引脚连接顺序)





- 上位机串口：连接上位机，则可以直接连接USB口（此时两个端子需要插上跳线），如果连接其他MCU，则去掉跳线，直接接上面的TX与RX两根线。



其他非必要引脚包括：

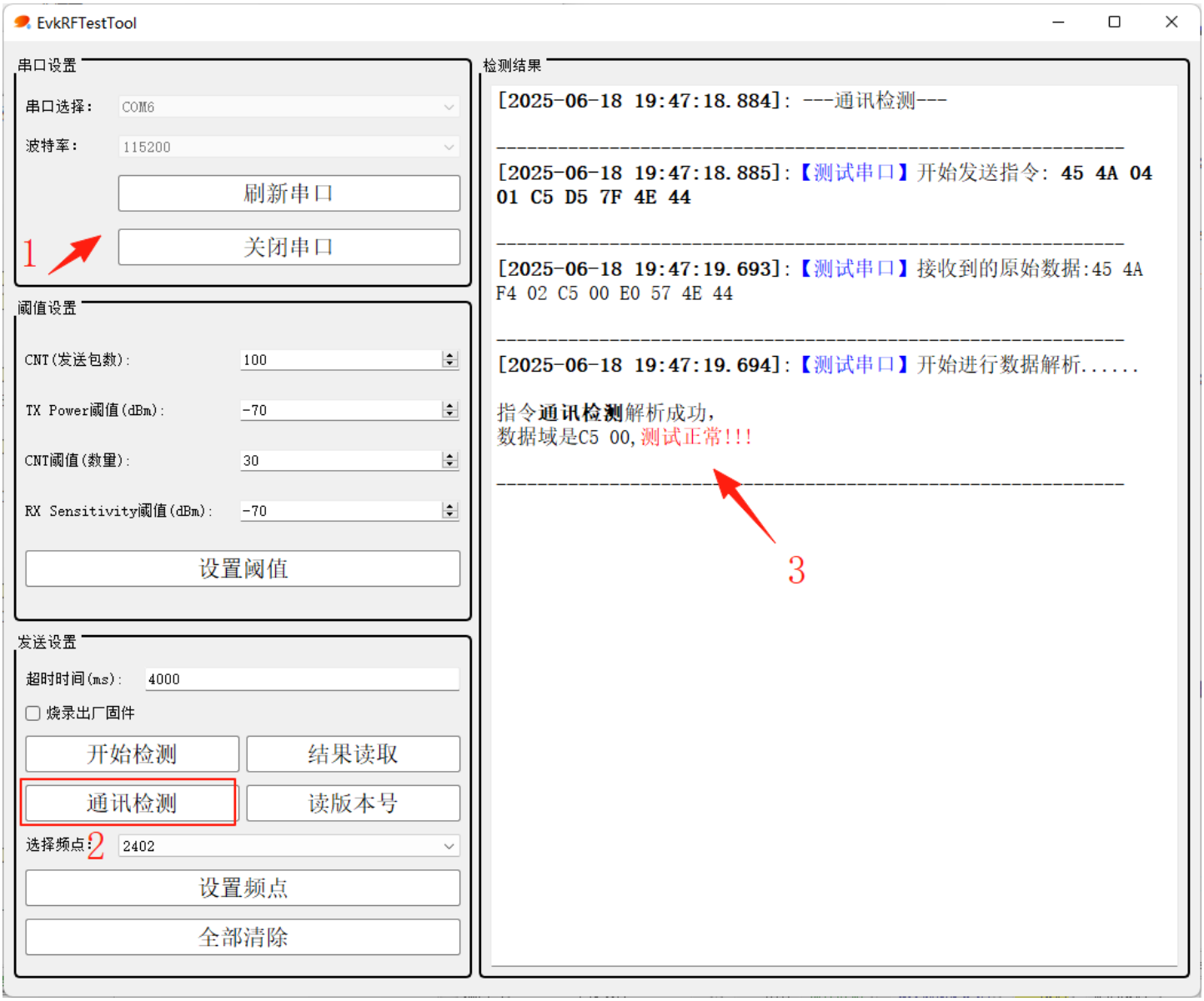
- golden调试：用于调试阶段进行烧录及log打印，正式量产时不需要使用
- 3.3V：一组3.3V电源和地，调试用。
- 5V：可以通过外部给线路板供电。

4.3 连接检测

将所有线路连接好并上电后，可以通过上位机检测系统是否工作正常。

此时通过USB接口连接上位机，首先刷新并连接串口，然后点击“**通讯检测**”按钮，可以在右侧窗口查看检测结果。

下图所示表示连接正常，表示golden已供电正常，golden与上位机串口连接正常。



目前软件还未提供其他接口的连接检测，暂时需要人工检测，后续版本会增加。

4.4 阈值调试

实际搭建产测环境时，需要根据场地的实际情况来确定测试的各项阈值，包括：

- **cnt包数：**此次测试发送的包数基准，**范围是1~255包**
- **TX cnt：**DUT发送100包，golden接收，收到的包数**高于**某个阈值（比如80），认为合格
- **TX power：**DUT发送，golden接收，如果接收到的信号强度**强于**某个阈值（比如-30dbm），认为合格
- **RX cnt：**golden发送100包，DUT接收，收到的包数**高于**某个阈值（比如80），认为合格
- **RX sen：**golden以0dbm发送，DUT接收，接收到的信号强度**超过**某个阈值（比如-35dbm），认为合格

四项参数中TX cnt和TX power为对射频发射部分的评估，RX cnt和RX sen为对射频接收部分的评估。

现场环境不同，正常设备测试的数据会有不同，需要根据实际情况设定合适的阈值，以实现既不会把正常设备判定成损坏，也可以把损坏的设备检测出来。

5 协议数据举例

5.1 开始检测

代码块

```
//上位机
45 4A 04 02 C1 00 20 66 4E 44
//golden回复
45 4A F4 02 C1 00 20 55 4E 44
```

5.2 结果读取

代码块

```
//上位机
45 4A 04 01 C2 17 3E 4E 44
//golden回复-测试完成以及测试结果
45 4A F4 08 C2 00 0A 1E E2 BA FD 17 DC 0C 4E 44
//golden回复-测试进行中
45 4A F4 02 C2 01 10 94 4E 44
//golden回复-通讯故障，发生在tx cnt测试阶段
45 4A F4 03 C2 02 02 62 91 4E 44
```

5.3 设置频点

代码块

```
//上位机
45 4A 04 03 C3 62 09 70 E9 4E 44
//golden回复
45 4A F4 02 C3 00 40 54 4E 44
```

5.4 设置阈值

按下设置阈值按钮后，会依次进行测试包数、tx power，tx cnt，rx sen，rx cnt五个阈值的设置。

代码块

```
//上位机设置测试包数
45 4A 04 05 C4 01 00 64 00 54 1F 4E 44
//golden回复
45 4A F4 02 C4 00 70 56 4E 44
//上位机设置tx power的阈值
```

```
45 4A 04 05 C4 02 00 BA FF F0 06 4E 44
//golden回复
45 4A F4 02 C4 00 70 56 4E 44
//上位机设置tx cnt的阈值
45 4A 04 05 C4 03 00 1E 00 8C 3D 4E 44
//golden回复
45 4A F4 02 C4 00 70 56 4E 44
//上位机设置rx sen（灵敏度）的阈值
45 4A 04 05 C4 04 00 BA FF 78 06 4E 44
//golden回复
45 4A F4 02 C4 00 70 56 4E 44
//上位机设置rx cnt的阈值
45 4A 04 05 C4 05 00 50 00 A4 08 4E 44
//golden回复
45 4A F4 02 C4 00 70 56 4E 44
```

5.5 通讯检测

代码块

```
//上位机
45 4A 04 01 C5 D5 7F 4E 44
//golden回复
45 4A F4 02 C5 00 E0 57 4E 44
todo..需要补充测试失败和测试成功的数据
```

5.6 读版本号

代码块

```
//上位机
45 4A 04 01 C6 D4 3F 4E 44
//golden回复
45 4A F4 06 C6 00 00 00 00 01 B1 95 4E 44
```

5.7读取UID

代码块

```
//上位机
45 4A 04 01 C7 14 FE 4E 44
//golden回复
45 4A F4 12 C7 16ByteUID CRC CRC 4E 44
```

6 重要说明

6.1 频偏校准

硬件的频偏可能会随着时间推移发生偏移，因此4.1提到的频偏测试与校准，最好每年进行一次，否则可能导致产品的系统性频偏问题，如某些手机连接不上，容易断开等。

版本记录

文件版本	发布时间	说明
V1.0.0	2025/6/20	第一版发布
V1.1.0	2025/7/2	1）增加了第6部分重要说明 2）增加了uid读取协议 3）增加了2.1烧写测试固件的注意事项