Hi3137 V100 Debugging

FAQs

Issue 00B01

Date 2014-03-31

Trademarks and Permissions

 , HISILICON, and other HiSilicon icons are trademarks of HiSilicon Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between HiSilicon and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

HiSilicon Technologies Co., Ltd.

Address: Huawei Industrial Base
Bantian, Longgang
Shenzhen 518129
People's Republic of China

Website: http://www.hisilicon.com

Email: support@hisilicon.com

About This Document

Purpose

This document describes solutions to common issues during debugging of the digital video broadcasting-terrestrial (DVB-T)/DVB-T2 Demod.

Related Versions

The following table lists the product versions related to this document.

Product Name	Version
Hi3716C	V1XX
Hi3716C	V2XX
Hi3719C	V1XX
Hi3719M	V1XX
Hi3716M	V3XX
Hi3716M	V4XX

Intended Audience

This document is intended for:

- Technical support engineers
- Software development engineers

Symbol Conventions

The symbols that may be found in this document are defined as follows.

Symbol	Description
 DANGER	Alerts you to a high risk hazard that could, if not avoided, result in serious injury or death.
 WARNING	Alerts you to a medium or low risk hazard that could, if not avoided, result in moderate or minor injury.
 CAUTION	Alerts you to a potentially hazardous situation that could, if not avoided, result in equipment damage, data loss, performance deterioration, or unanticipated results.
 TIP	Provides a tip that may help you solve a problem or save time.
 NOTE	Provides additional information to emphasize or supplement important points in the main text.

Change History

Changes between document issues are cumulative. Therefore, the latest document issue contains all changes made in previous issues.

Issue 00B01 (2014-03-31)

This issue is the first draft release.

Contents

Figures

Tables

1 Hi3137 Debugging FAQs

1.1 How Do I Locate a Problem Related to the I²C?

Problem Description

Both the Demod and the tuner are I^2C slave devices that are controlled by the master device. That is, both the Demod and the tuner are mounted over the I^2C bus as peripherals of Hi3716C V100/Hi3716C V200/Hi3716C V300. Therefore, it is vital to ensure the I^2C communication between the chip CPU and the Hi3137 or tuner. When a Demod is being debugged, information similar to the following is displayed during initialization:

```
[942806 ERROR-ecs]:I2C_DRV_WaitWriteEnd[102]:wait write data timeout!
[942812 ERROR-ecs]:I2C_DRV_Write[230]:wait write data timeout!
[942819 ERROR-ecs]:I2C_DRV_Write[201]:wait write data timeout!
```

Cause Analysis

This issue occurs because the I^2C communication is abnormal.

Solution

Perform the following steps:

Step 1 Confirm the address for the Hi3137 and the tuner.

The address for the Hi3137 is an 8-digit binary number 10111*XXY*.

NOTE

XX is set by using the chip pin ADDR[1:0], and *Y* indicates the read or write operation (**1** indicates read and **0** indicates write).

Figure 1-1 takes Hi3137 DMO1C VER.A as an example. The address pins ADDR[1:0] are pulled down to the GND, and the Hi3137 I^2C address is set to 0xB8.

Figure 1-1 Hi3137 address pin connection

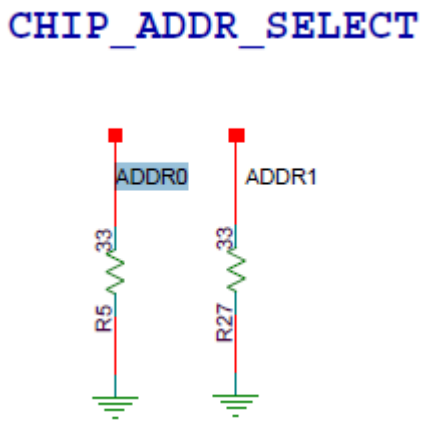


Figure 1-2 takes the MxL603 as an example. The upper seven bits of the address for the MxL603 are determined by the connection configuration of R1 and R2 that are connected to the AS pin.

Figure 1-2 MxL603 address pin connection

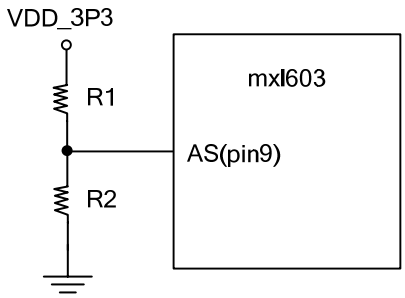


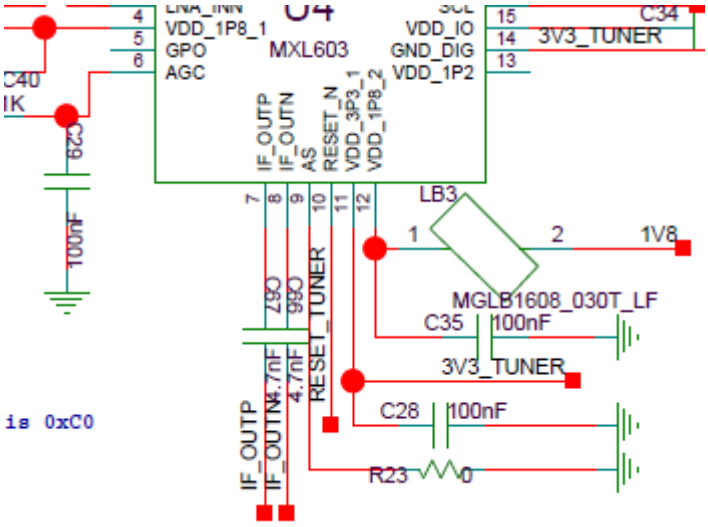
Figure 1-3 shows the mapping between the I^2C address for the MxL603 and the R1/R2 connection configuration.

Figure 1-3 Mapping between the I^2C address for the MxL603 and the R1/R2 connection configuration

I^2C Address	R1	R2
96	Open	Short
97	30kΩ ± 5%	15kΩ ± 5%
98	30kΩ ± 5%	60kΩ ± 5%
99	Short	Open

Figure 1-4 shows the AS pin connection of the MxL603 on the Hi3137 DMO1C VER.A board. R1 is disconnected, R2 is short-circuited, therefore the upper seven bits of the MxL603 address are 96 (0x60). The address is shifted left by one bit, and the read/write indicator occupies one bit, and the I²C address is set to 0xC0.

Figure 1-4 MxL603 address pin connection on the Hi3137 DMO1C VER.A board



Step 2 Check the used I²C pin ID.

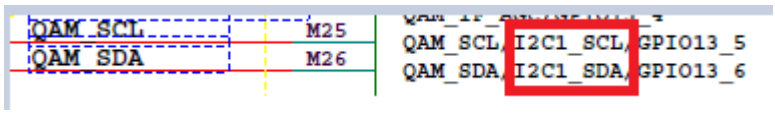
Confirm with the hardware engineer about the ID of the I²C pin to which the Hi3137 is connected. Hi3716C V100/Hi3716C V200 provides multiple I²C pins.

Take the Hi3716C DMO2B VER.C board as an example. During the debugging of the DVB-T2 tuner, the DVB-T2 tuner is connected to the J31 interface in external TS mode. The number 1 in the I²C signal in the red rectangle in Figure 1-5 indicates that I²C pin 1 is used.

NOTE

- The J34 interface of the Hi3716M DMO3A VER.B board uses I²C pin 0.
- The J31 interface of the Hi3716C DMO VER.C board uses I²C pin 3.

Figure 1-5 Connection between the external TS interfaces of the Hi3716C DMO2B VER.C board and I²C pins



Step 3 Check the I²C pin/TS signal pin multiplexing.

Check the TS pin multiplexing. If the TS pins are not multiplexed as sda/scl/clk/ts[7:0]/vld pins, modify the configuration as required. For details about pin multiplexing configurations, see the pin multiplexing registers in the hardware user guide of the corresponding chip.

Take the Hi3716C DMO2B VER.C board as an example. Table 1-1 lists the register configurations. The J31 interface is used for TS reception.

Table 1-1 Pin multiplexing configuration of the J31 interface on the Hi3716C DMO2B VER.C board

Register_addr	Value
0xf8a21190	0
0xf8a21194	2
0xf8a21198	2
0xf8a21100	1
0xf8a21104	1
0xf8a21108	1
0xf8a2110c	1
0xf8a21110	1
0xf8a21114	1
0xf8a21118	1
0xf8a2111c	1
0xf8a21120	1
0xf8a21124	1
0xf8a21128	1

Figure 1-6 Schematic diagram of the J31 interface on the Hi3716C DMO2B VER.C board

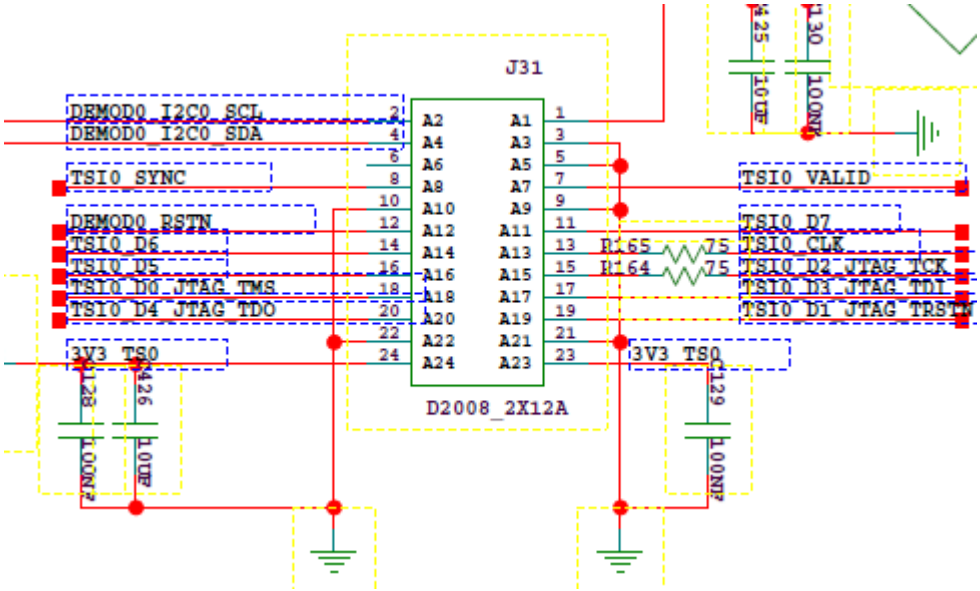
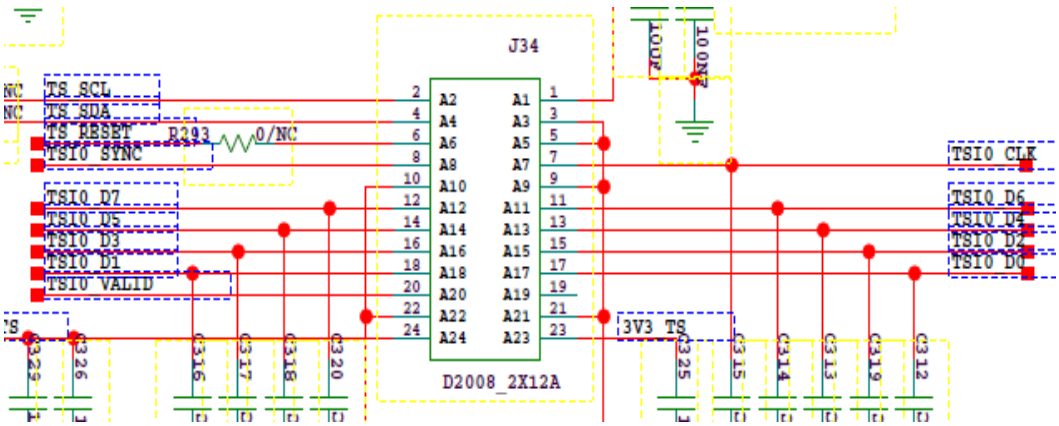


Table 1-2 lists the register configurations. The J34 interface on the Hi3716M DMO3A VER.B board is used for TS reception.

Table 1-2 Pin multiplexing configuration of the J34 interface on the Hi3716M DMO3A VER.B board

Register_addr	value
0x10203144	2
0x1020314c	0
0x10203150	1
0x10203154	1
0x10203158	1
0x1020315c	1
0x10203160	1
0x10203164	1
0x10203168	1

Figure 1-7 Schematic diagram of the J34 interface on the Hi3716M DMO3A VER.B board



Step 4 Test the I²C communication by running the **echo** command.

NOTE

You can switch the running sample_tuner to the background by pressing **Ctrl+Z**.

Check whether the I²C communication is normal by running the **echo** command.

- Read the Demod registers over the I²C channel by running the following command:
  `echo <I²C channel ID> <I²C address for the Demod> <Address for a register of the Demod> >/proc/msp>i2c`

NOTE

If the I²C channel ID or the device address is incorrect, a write timeout error is reported.

- Write to the Demod registers over the I²C channel by running the following command:
  `echo <I²C channel ID> <I²C address of the Demod> <Address for a`

```
register of the Demod> <Register value> >/proc/msp/i2c
```

You can write a value to a register of the Hi3137 and then read it by running the preceding commands. If the I^2C channel works properly, the value written to the register is displayed.

Run the preceding **echo** commands after the Hi3137 and tuner addresses, the used I^2C ID, and pin multiplexing information are confirmed and configured.

NOTE

Take the Hi3716C DMO2B VER.C board as an example. Read the 0x6b register of the Hi3137 by running **echo 1 b8 6b >/proc/msp/i2c**. If data can be read correctly, the I^2C channel is normal.

Figure 1-8 Reading the 0x6b register of the Hi3137

```
# echo 1 b8 6b >/proc/msp/i2c
Read: u32I2cNo=1, u32DevAddr=0xb8, u32RegAddr=0x6b
0xa9
```

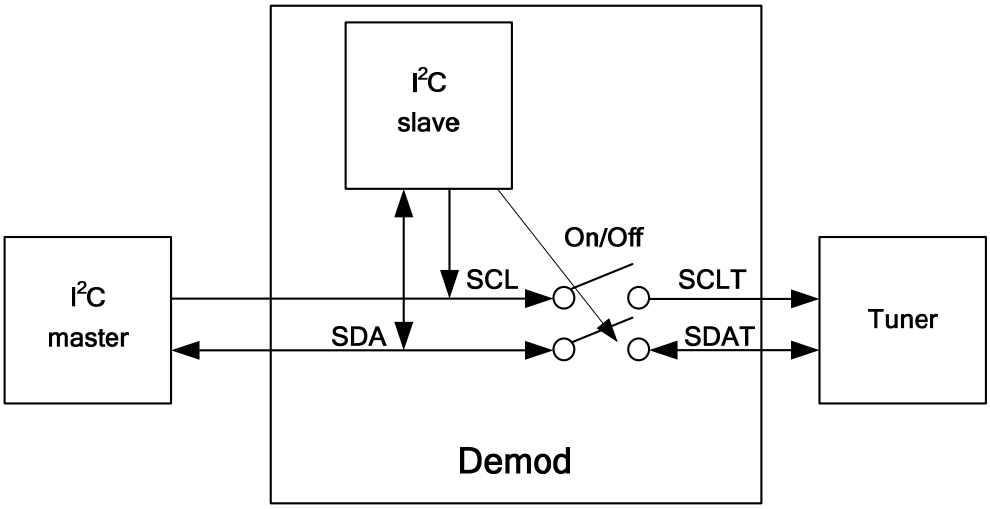
Step 5 Check the implementation of the I^2C communication timings by checking the driver.

When the Hi3137 registers can be correctly read by using the echo command, the I^2C channel is normal, but not necessarily the driver. Run **./sample_tuner**. If the following information is displayed, the I^2C communication times out:

```
942812 ERROR-ecs]:I2C_DRV_Write[230]:wait write data timeout!
```

In this case, check the tuner driver. The tuner and the Hi3137 are mounted to the same I^2C. Therefore, you cannot check whether the CPU of the main board communicates with the Hi3137 or the tuner simply by using the component address.

Figure 1-9 I^2C forwarding principles



As shown in Figure 1-9, the I^2C channel between the tuner and the CPU is typically disconnected. If the CPU wants to communicate with the tuner, you need to write the value **1** to the 0x2d register of the Hi3137 to connect the tuner to the I^2C. Take the Hi3716C DMO2B VER.C board as an example:

```
echo 1 b8 2d 1 >/proc/msp/i2c
```

Verify that the I²C channels between the CPU and the Hi3137 and the tuner are normal by running the echo read command. In this case, the hardware and software configurations are correct.

Pay attention to the software driver. The I²C communication timeout information is from the tuner but not the Hi3137.

CAUTION

The driver code of the MxL603 that works with the Hi3137 used to adapt to the MN*xxxx* timing of the competitor. When the Hi3137 enables the channel between the I²C and the MxL603, it writes **1** to the 0x2d register. However, the MN*xxxx* enables the channel between the I²C and the MxL603 by caching the register address and value to 0xF7 of the MN*xxxx*. Different Demod chip vendors use different implementation mechanisms.

----End

1.2 What Do I Do If the I²C Channel Is Normal But the Frequency Cannot Be Locked?

Problem Description

After the modulator of the current end is configured correctly and stream signals are transmitted, when the **setchnl** command is executed to lock the frequency, the following information is displayed, indicating that the frequency fails to be locked:

```
setchnl 666000 8000
[17779531 ERROR-tuner]:hi3137_connect[571]:<<<---DVB-T.
[17780171 ERROR-tuner]:tuner_osr_connect[921]:
 tuner_connect failed
Tuner Lock freq 666000 b[17780178 ERROR-tuner]:hi3137_connect[553]:<<<---
DVB-T2.
andwidth 8 Fail!, s32Ret = 0xffffffff
[17781727 ERROR-tuner]:tuner_osr_connect[921]:
 tuner_connect failed
Tuner Lock freq 666000 bandwidth 8 Fail!, s32Ret = 0xffffffff
Tuner Lock freq 666000 bandwidth 8 Fail!
```

Cause Analysis

The Hi3137 or the tuner is incorrectly configured, and therefore the DVB-T/DVB-T2 signals cannot be received.

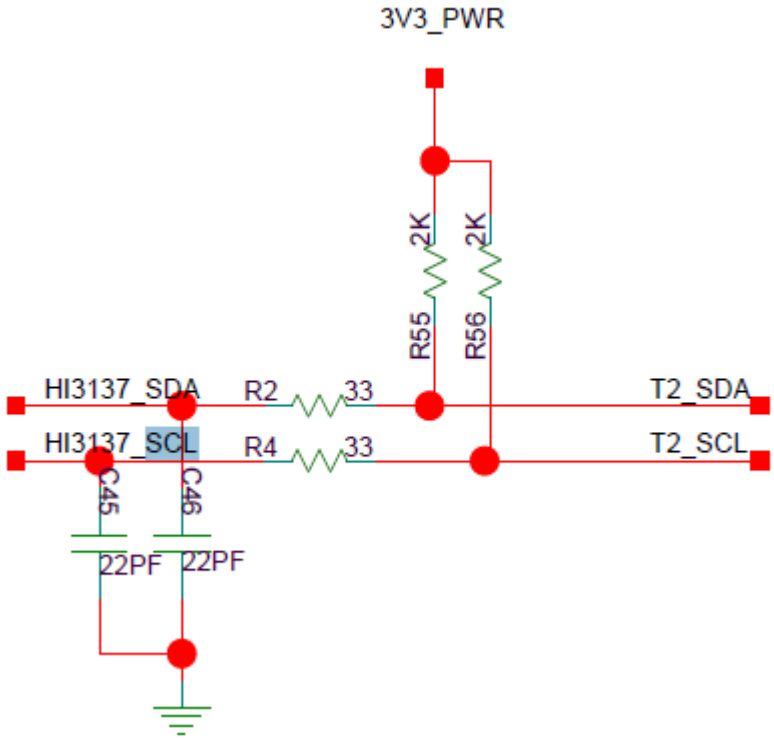
Solution

Perform the following steps:

Step 1 Check the capability of the I^2C channel with loads.

Check the hardware circuit design of the I^2C channel. Check whether the I^2C pull-up impedance meets requirements. When multiple devices are mounted to an I^2C bus, pay attention to the drive current capability and matched capacitors. You can observe the actual waveform by using an oscilloscope and then change the related circuit design. Check whether the pull-up voltage meets the logic level of the tuner. The typical logic level of mainstream RF chips is 3.3 V (VIH) or 0 V (VIL).

Figure 1-10 Hi3137 I^2C hardware reference design



Step 2 Check the configuration of the crystal oscillator.

If the I^2C channel is normal but the frequency still cannot be locked, check whether the crystal oscillator clocks of the Hi3137 and the tuner are configured correctly. Contact the hardware engineers to check the actual crystal oscillator values.

- For the Hi3137 DMO1C VER.A board, both the Hi3137 and the MxL603 use the 24 MHz crystal oscillator.
- For the Hi3137 DMO1D VER.A board, the Hi3137 uses the 24 MHz crystal oscillator, and the MxL603 uses the 16 MHz crystal oscillator.

Step 3 Check whether tuner address conflict occurs.

Check whether two components with the same address are mounted to the same I^2C bus. For example, a DVB-C tuner and a DVB-T2 tuner with the same address are mounted to the same I^2C bus. The driver intends to initialize and configure the DVB-T tuner. However, the

commands are transmitted to the DVB-C tuner mistakenly, and the DVB-T tuner cannot lock the frequency.

For example, the Hi3716C DMO2B VER.C board has an embedded DVB-C Demod. Therefore, a DVB-C tuner is soldered. The address of this tuner is 0xC0, which is the same as that for the MxL603 on the Hi3137 DMO1C VER.A board and that for the MxL601 on the Hi3137 DMO1D VER.A board. These tuners are all mounted to I^2C bus 1. To avoid address conflict, disconnect the DVB-C tuner from I^2C bus 1.

Step 4 Check whether the key modules (PLL and ADC modules) of the Hi3137 are running. For the Hi3137:

- The ratio of the analog-to-digital converter (ADC) clock to the demodulation clock is 1:1 or 1:2, which can be configured by setting the 0x0[5:0] register.
  - If 0x0[5:0] is set to 1, the ratio is 1:2.
  - If 0x0[5:0] is set to 3, the ratio is 1:1.
- Check the register bits 0x7[4] and 0xe[4]. The value 0 indicates that PLL0 and PLL1 are running.
- Check the register bit 0x3[5:4]. The value 3 indicates that the ADC is running.

CAUTION

If 0x3[6] is set to 1 (the ADC auto correction module is disabled), the Hi3137 cannot lock the frequency.

----End

1.3 What Do I Do If the Frequency Is Locked Successfully But Images Cannot Be Displayed?

Problem Description

In sample_tuner, run the **setchnl** command. The following information is displayed, indicating that the frequency is locked successfully:

```
Tuner   Lock freq 666000 bandwidth 8 Success!
```

However, no image is displayed on the television screen after the **play** command is executed.

Cause Analysis

This problem occurs because the configuration of the TS_OUT interface of the Hi3137 mismatches that of the TS_IN interface of the decoder on the backend main board.

Solution

Check the displayed information after running **cat /proc/msp/demux_port**.

Figure 1-11 demux_port debugging information

```
# cat /proc/msp/demux_port
 -------------------------------IF port-------------------------
 Id  AllTsCnt   ErrTsCnt  Lock/lost  ClkReverse    BitSel   Type
  0    0x0        0x0        5/1         0          D7       PARALLEL_NOSYNC_188

-------------------------------TSI port-------------------------
 Id  AllTsCnt   ErrTsCnt  Lock/lost  ClkReverse    BitSel   Type
  1    0x0        0x0        5/1         0          D7       PARALLEL_VALID
  2    0x0        0x0        5/1         0          D7       PARALLEL_NOSYNC_188
  3    0x0        0x0        5/1         0          D7       PARALLEL_NOSYNC_188
  4    0x0        0x0        5/1         0          D7       PARALLEL_NOSYNC_188

-----------------------------------------TS0 port-----------------------------------------------------
 Id   Enable  ClkReverse  TSPortID    ClkMode     VldMode   Sync    Serial  BitSel  LSB   Clk   ClkDiv
 0      1         0          0        NORMAL        0       Bit       1      D0      0    150M     2
 1      1         0          0        NORMAL        0       Bit       1      D0      0    150M     2
```

Locate the problem as follows:

- Observe the value of **ErrTsCnt** of the TSI port.

  If the value increases when **cat /proc/msp/demux_port** is executed for two consecutive times, the TS error packets increase. In this case, check whether the TS signal line sequence is correct, and check the TS Data[7:0] pin multiplexing.
- Ensure that the TS transmission mode (parallel mode, serial mode, or 2-bit serial mode) at the output end and that at the RX end are the same.
- Ensure that the Demux is bound to the correct TS interface.

  When TSs are received by using the external Demod, bind the TS interface 32 to the Demux. If there are multiple external TSs (for example, Hi3716C V200 supports four external Demods), the IDs of TS interfaces to be bound are 32 to 35. You can query information about the bound interfaces by running **cat /proc/msp/demux_main**.

Figure 1-12 Information of interfaces bound to the Demux

```
# cat /proc/msp/demux_main
DmxId    PortId
  0       32
  1       --
  2       --
  3       --
  4       --
  5       --
  6       --
```

Take Hi3716C DMO2B VER.C+Hi3137 DMO1C VER.A as an example. If the hardware and drivers are prepared, run **make menuconfig**, choose **Board** > **Tuner Config** > **First Tuner Config**, and configure the TS TX end according to Figure 1-13.

Figure 1-13 Tuner chip attribute configuration