

# 产品文档 : Apex Xavier II+ 用户手册

- [安全警示及使用注意事项](#)
- [Apex Xavier II+ 产品介绍](#)
  - [简介 Brief](#)
  - [产品清单](#)
  - [产品规格 Specifications](#)
    - [处理器模组 Processor](#)
    - [接口 I/O](#)
    - [供电 Power Supply](#)
    - [结构 Mechanical](#)
    - [环境 Environmental](#)
  - [尺寸及安装 Install Dimension](#)
  - [服务与支持](#)
    - [技术支持](#)
    - [保修](#)
- [接口说明及扩展安装方式](#)
  - [接口说明](#)
    - [正面接口](#)
      - [EXPANSION PORT ① 接口信号定义](#)
      - [外接GPS授时](#)
      - [GPS授时串口波特率调整](#)
      - [EXPANSION PORT ② 接口信号定义](#)
    - [背面接口](#)
  - [IO转接线说明](#)
  - [PORT 1转接线说明](#)
    - [UART\(232\)连接线及引脚定义](#)
    - [UART\(422/485\)连接线及引脚定义](#)
    - [CAN\(CAN FD\)连接线及引脚定义](#)
    - [PPS连接线及引脚定义](#)
    - [SYNC连接线及引脚定义](#)
    - [功能按键](#)
  - [PORT 2转接线说明](#)
    - [#线束接头颜色待更改](#)
    - [电源接口](#)
    - [航插电源转接线](#)
    - [CAN\(CAN FD\)连接线及引脚定义](#)
    - [GPIO连接线及引脚定义](#)
    - [风扇连接线及引脚定义](#)
- [扩展设备的安装方式](#)
  - [Mini PCIe 4G支持清单](#)
  - [Mini PCIe WIFI支持清单](#)
- [功能介绍](#)
  - [通用使用方法](#)
    - [系统介绍](#)
    - [烧写镜像](#)

- [开关机](#)
- [MIIVII SETTINGS的使用说明](#)
  - [简介](#)
  - [使用视频](#)
  - [登陆](#)
  - [功能说明](#)
    - [系统状态](#)
    - [系统设置](#)
    - [系统升级](#)
    - [账号管理](#)
    - [日志导出](#)
- [功率模式设定](#)
- [IO使用方法](#)
  - [GPIO接口配置方法](#)
  - [米文设备GPIO输出模式说明](#)
  - [UART接口配置方法](#)
  - [GPS 对设备授时使用方法](#)
    - [GPS支持型号](#)
    - [连接方式](#)
    - [授时功能配置](#)
    - [检查授时是否成功](#)
    - [故障排查](#)
      - [1.查看GPS是否有输出](#)
      - [2.查看GPS的pps信号是否有输出](#)
      - [3.识别方法](#)
  - [CAN口配置方法](#)
  - [扩展CAN FD口配置方法](#)
- [扩展设备配置方法](#)
  - [扩展SSD硬盘使用](#)
- [无线设备配置方法](#)
  - [WiFi配置方法](#)
  - [4G模块配置方法](#)
- [同步功能使用说明](#)
  - [同步功能介绍](#)
  - [同步功能使用方法](#)
    - [PPS同步模式](#)
    - [Sync out 同步模式](#)
    - [Sync in 同步模式](#)
  - [同步误差测试方法](#)
    - [通过示波器测量PPS脉冲间隔](#)
    - [通过示波器测量Sync out脉冲间隔](#)
  - [自行评估同步效果的方法](#)
    - [同步sample code使用说明](#)
    - [Sync out jitter测量](#)
    - [Sync in jitter测量](#)
    - [PPS jitter测量](#)
- [GMSL2摄像头支持](#)
- [GMSL2摄像头使用方法](#)
  - [接口特性](#)

- [连线方式](#)
- [名词解释](#)
- [摄像头配置](#)
- [快速验证](#)
- [视频输出](#)
- [GMSL/GMSL2时间戳相关测试方法](#)
  - [如何获取详细日志及日志说明?](#)
  - [如何确认时间戳是否准确?](#)
  - [如何确认时间戳精度?](#)
  - [如何确认图像帧传输延迟是否稳定?](#)
    - [确认摄像头图像帧传输延迟](#)
  - [Apex Xavier II 系列](#)
  - [Apex Xavier和EVO TX2 GMSL2](#)
- [应用功能使用](#)
- [附录](#)
  - [异常处理](#)
- [系统在线升级\(OTA\)的使用说明](#)
  - [概述](#)
  - [使用方式](#)
    - [方法一\(推荐\): 使用MIIVII SETTINGS进行版本升级;](#)
    - [方法二: 使用命令行进行升级或者升级指定安装包](#)
    - [使用命令行进行升级](#)
    - [升级指定安装包](#)
- [Jetpack 4.4版本及以下镜像烧录](#)
- [Jetpack 4.5版本及以上镜像烧录](#)
  - [1.功能介绍](#)
    - [核心功能](#)
  - [2.准备软件硬件](#)
    - [2.1. 烧写主机准备](#)
    - [2.2. 烧写软件环境准备](#)
    - [2.3. 准备米文烧写工具和米文设备镜像](#)
    - [2.3.1.刷机工具安装](#)
    - [2.4. 准备硬件](#)
  - [3.操作](#)
    - [3.1. 硬件连接](#)
    - [3.2. 软件使用](#)
      - [3.2.1. 镜像烧写](#)
      - [3.2.1.1 在线模式镜像烧写](#)
      - [3.2.1.2 离线模式镜像烧写](#)
      - [3.2.2. 镜像克隆](#)
    - [附1. 烧写问题自检](#)
  - [Apex Xavier II+ 产品软件-Release Note](#)

## 安全警示及使用注意事项

请在使用本产品前仔细阅读本手册，未经授权的操作会导致错误或意外。制造商对因错误操作而导致设备出现的任何问题均不负责。

- 避免热插拔设备接口。
- 要正确关闭电源，请先关闭Ubuntu系统，然后再切断电源。由于Ubuntu系统的特殊性，在Nvidia的开发板上，如启动未完成的时候强行断电，会有0.03%的概率出现异常，进而

导致设备无法启动。由于使用Ubuntu系统，米文的设备上也会存在同样的问题。

- 请勿使用本手册提及以外的线缆。
- 避免在强磁场环境下使用本设备。
- 长期不使用及运输前需要对数据进行备份。
- 推荐使用原包装进行运输。
- 警告！此为A级产品，在生活环境，该产品可能会造成无线电干扰。在这种情况下，可能需要用户对干扰采取切实可行的措施。

# Apex Xavier II+ 产品介绍

## 简介 Brief

米文Apex Xavier II+是一款嵌入式人工智能计算机，能够为众多终端设备赋予人工智能计算力，从而有效降低人工智能产品的落地门槛。Apex Xavier II+可以在满足抗振防水等工业级标准的同时，提供高达32Tops计算能力，能够很好的满足低速无人驾驶等场景的视觉计算需求。除此之外Apex Xavier II+还可提供高效的多传感器时钟同步功能。

## 产品清单

- Apex Xavier II+ × 1
- 电源适配器 × 1
- PORT 1转接线 × 1
- PORT 2转接线 × 1
- 4合1 MINI FAKRA转接线 × 2
- 4G模块 × 1 (非标配，可选)
- 4G天线 × 2
- WiFi模块 × 1 (非标配，可选)
- WiFi天线 × 2
- SSD × 1 (非标配，可选)
- 保修卡 × 1
- 合格证 × 1



## 产品规格 Specifications

### 处理器模组 Processor

	Specification	
<b>Processor</b>	NVIDIA Jetson AGX Xavier 32GB	NVIDIA Jetson AGX Xavier 64GB
<b>AI Performance</b>	Up to 32T OPS	
<b>CPU</b>	8-core ARM v8.2 64-bit CPU	
<b>GPU</b>	512-core Volta GPU	
<b>Memory</b>	32GB 256-Bit LPDDR4	64GB 256-Bit LPDDR4 <sup>1</sup>
<b>DL Accelerator</b>	2×NV DLA Engines	
<b>Storage</b>	32GB eMMC 5.1	

	Specification	
<b>Processor</b>	NVIDIA Jetson AGX Xavier 32GB	NVIDIA Jetson AGX Xavier 64GB
<b>Video Encode</b>	4x 4Kp60 8x 4Kp30 16x 1080p60 32x 1080p30 (H.265) 4x 4Kp60 8x 4Kp30 14x 1080p60 30x 1080p30 (H.264)	
<b>Video Decode</b>	2x 8Kp30 6x 4Kp60 12x 4Kp30 26x 1080p60 52x 1080p30 (H.265) 4x 4Kp60 8x 4Kp30 16x 1080p60 32x 1080p30 (H.264)	

<sup>1</sup>Jetson AGX Xavier DRAM内存有32GB和64GB两种规格。

## 接口 I/O

接口 Interface	规格/Specifications
网络接口 Network	4× Gigabit Port (可选配IEEE 802.3 at PoE+ 25.5W)
相机接口 Camera	2× GMSL2 4 IN 1 MINI FAKRA TYPE (10V,Transmission distance up to 15 meters,GMSL2 compatible with GMSL1)
视频输出 Video output	1× HDMI 2.0 (TYPE A)
USB	2× USB 3.0 (TYPE A)
通用输入/输出口 GPIO	4× In (0-12V) 4× Out (3.3V)
CAN FD	5× CAN FD (With CAN chip Terminal resistor 120Ω)
串口 UART	1× Debug (RS232) , 3× RS232, 2× RS485/RS422

接口 Interface	规格/Specifications
同步输入/输出 口 Sync I/O	1× SYNC_IN (0-12V) , 1× SYNC_OUT (3.3V) , 1× SYNC_PPS (3.3V)
扩展接口 User Expansion	1× M.2 M Key (PCIe x4, 2280) , 1× Mini PCIe (For 4G or WiFi expansion) , 1× Nano SIM Socket
按键 Function Key	1× Power KEY, 1× Reset KEY, 1× Recovery KEY (Button)

## 供电 Power Supply

Power Supply	Spec
Input Type	DC
Input Voltage	Wide input 9-36V DC
Typical consumption	30W

## 结构 Mechanical

Mechanical	Spec
Dimensions (W×D×H)	276mm×214mm×68mm (I/O ports and mounting holes included) 240mm×173mm×68mm (I/O ports and mounting holes excluded)
Weight	2.7Kg

## 环境 Environmental

Environmental	Spec
Operating Temperature	-25°C-70°C
Storage Temperature	-40°C-80°C
Storage Humidity	10%-90% non-condensing
Vibration	2Grms, 10Hz~500Hz, 1h/axis
Protection	IP5X (默认) IP65 <sup>1</sup> (可选)

1. Apex Xavier II+ 接口的接头选型均为IP65级别。若对整体产品有高防水要求则需要在购买时联系米文动力要求进行产品防水预处理。

## 尺寸及安装 Install Dimension

Apex Xavier II+ 主体尺寸及安装孔位尺寸如图:  
Dimensions and mounting hole position as below:

<b>俯视图 Up view(Unit:mm)</b>

<b>主视图 Front view(Unit:mm)</b>

<b>左视图 Left view(Unit:mm)</b>

<b>安装孔位图 Mounting Hole(Unit:mm)</b>


## 服务与支持

### 技术支持

如果您遇到问题，或者您认为您的产品有缺陷，请发问题到email: [helpdesk@miivii.com](mailto:helpdesk@miivii.com)，我们将帮助您解决问题。也可访问米文技术论坛<http://forum.miivii.com>，搜索我们的知识库，以查找常见问题的解决方案。

### 保修

保修期：米文设备保修期为自购买之日起一年。保修条例：保修期内产品，若出现非人为损坏的故障米文将进行免费保修。请联系[helpdesk@miivii.com](mailto:helpdesk@miivii.com)获取保修协助。

## 接口说明及扩展安装方式

### 接口说明

#### 正面接口


<b>图 Apex Xavier II+ 正面接口示意图</b>

接口	接口名称	接口说明
EXPANSION PORT①	多功能接口 IO_1	1路RS232 DEBUG口 3路RS232串口 2路RS422接口，兼容RS485 2 路 CAN FD口(带有CAN芯片，终端电阻 120Ω) 1路SYNC_PPS 接口(3.3V) 1路 SYNC_OUT接口(3.3V) 1路 SYNC_IN 接口(Logic High 1V-12V, Logic Low 0V-0.8V) 1 个 POWER_ONKEY启动按钮 1个FORCE_RECOVERY刷机按钮 1个RESET复位按钮
EXPANSION PORT②	多功能接口 IO_2	1路输入电源9-36V 3 路 CAN FD口(带有CAN芯片，终端电阻 120Ω)

接口	接口名称	接口说明
		8路GPIO 口 (4×In(Logic High 1V-12V, Logic Low 0V-0.8V) ,4×Out (3.3V) ) 1路FAN口 (5v PWM)
ANT1	ANT1	4G天线外延口 或 2.4G/5.8G WIFI天线外延口
ANT2	ANT2	4G天线外延口 或 2.4G/5.8G WIFI天线外延口
GMSL/GMSL2 4 IN 1 MINI FAKRA	GMSL /GMSL2输入	可接入8路GMSL/GMSL2协议的摄像头

其中，多功能接口介绍如下：

EXPANSION PORT ①与EXPANSION PORT②，位于Apex Xavier II+ 正面位置，如图所示：

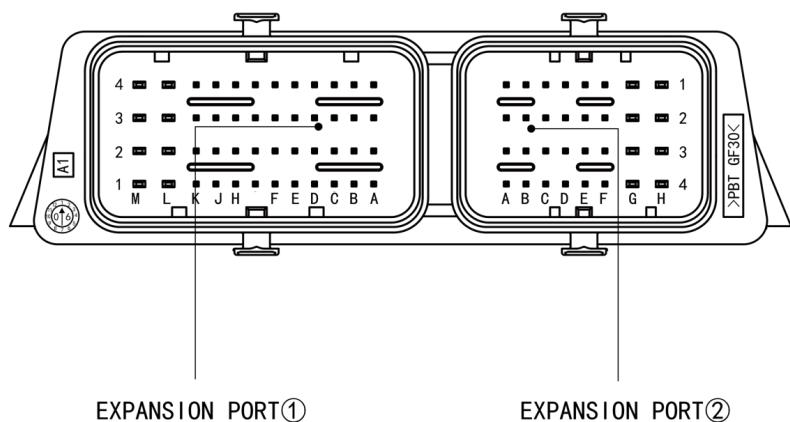


图 多功能接口图

其中，LAN1-4千兆以太网口可硬件扩展POE功能  
如需使用POE功能，请在够买时联系相关销售人员

其中，摄像头接口引脚定义如下：



图 Apex Xavier II+ 摄像头接口示意图

Circuit	设备节点
0	video0
1	video1
2	video2
3	video3
4	video4

Circuit	设备节点
5	video5
6	video6
7	video7

Apex Xavier II+ 提供8路GMSL2摄像头接口，接口特性描述如下：

- 支持两组共8路GMSL2摄像头输入。
- Apex Xavier II+ 为GMSL2摄像头的供电电压为10V，请确认所使用摄像头的电压允许范围，避免过压烧毁摄像头。
- 8路GMSL2摄像头支持摄像头自触发（默认设置）。
- 8路GMSL2摄像头支持被同一个固定输出频率的同步信号触发。

###

### EXPANSION PORT ① 接口信号定义

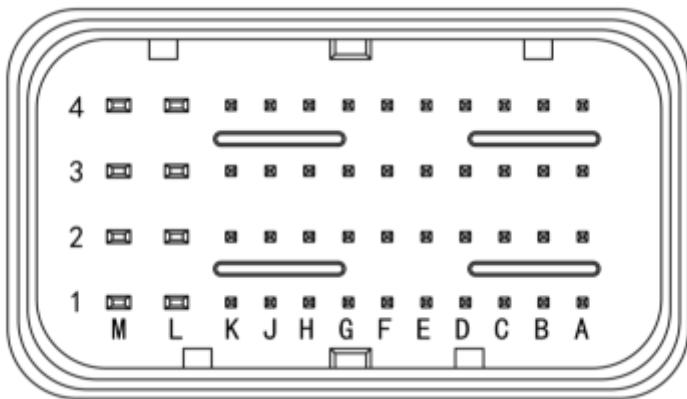


图 EXPANSION PORT ①接口序号图

接口名称	引脚序号	接口信号定义	接口说明
UART(DEBUG) 8	G1	UART(DEBUG)_RX	UART(DEBUG)信号：232-RX
	H1	UART(DEBUG)_TX	UART(DEBUG)信号：232-TX
	J1	GND	地
UART(232)B	D1	UART(232)B_RX	UART(232)B信号：232-RX
	E1	UART(232)B_TX	UART(232)B信号：232-TX
	F1	GND	地
UART(232)C	B2	UART(232)C_RX	UART(232)C信号：232-RX
	A2	UART(232)C_TX	UART(232)C信号：232-TX
	A3	GND	地

UART(422/485)A	K1	UART(422T+/485_A)A	UART(422/485)A信号： 422A/485_A
	L1	UART(422T-/485_B)A	UART(422/485)A信号： 422B/485_B
	M1	UART(422R+)	UART(422)A信号： 422_C
	M2	UART(422R-)	UART(422)A信号： 422_D
	L2	GND	地
UART(422/485)B	G2	UART(422T+/485_A)B	UART(422/485)B信号： 422A/485_A
	F2	UART(422T-/485_B)B	UART(422/485)B信号： 422B/485_B
	E2	UART(422R+)	UART(422)B信号： 422_C
	D2	UART(422R-)	UART(422)B信号： 422_D
	C2	GND	地
CAN_A	B3	CAN_A_L	CAN_A信号 低位数据线
	C3	CAN_A_H	CAN_A信号 高位数据线
CAN_B	D3	CAN_B_L	CAN_B信号 低位数据线
	E3	CAN_B_H	CAN_B信号 高位数据线
PPS_A	A1	PPS_A_RX	PPS_A信号： TTL-RX
	B1	PPS_A_TX	PPS_A信号： TTL-TX
	C1	GND	地
	F3	PPS_A_SYNC	PPS_A_SYNC 秒脉冲信号
	G3	GND	地
UART(232)A	K2	UART(232)A_RX	UART(232)A信号： 232-RX
	J2	UART(232)A_TX	UART(232)A信号： 232-TX
	H2	GND	地
	H3	空	空
	J3	GND	地

SYNC_IO	K3	SYNC_IN	Sync in同步信号
	L3	GND	地
	M3	SYNC_OUT	Sync out同步信号
	M4	GND	地
	K4	GND	地
RESET按键	J4	RESET	复位按钮
RECOVERY按键	G4	FORCE_RECOVERY	恢复按钮
POWER 按键	E4	POWER_ONKEY	启动按钮
GND	H4	GND	地
	F4		
	D4		

## [8] UART(DEBUG) 接口为DEBUG接口\*

### 外接GPS授时

若需要使用GPS外部授时功能，接线方案如下：

GPS的NMEA输出串口对接Apex Xavier II+ 的UART(232)B硬件串口(串口波特率为9600),映射到Linux系统为/dev/ttyUART\_232\_B设备节点。

GPS的pps秒脉冲输出信号线对接Apex Xaiver II的SYNC\_IO线的PIN1管脚，映射到Linux系统为/dev/miivii-sync-in-a设备节点。

在GPS授时模式下，如上两个节点会被后台GPS授时处理程序占用。请勿对这两个节点进行其他操作，否则GPS授时功能会被打断。

### GPS授时串口波特率调整

gps授时串口节点可调：

在/etc/systemd/sync\_auto.sh脚本中进行修改，具体如下；

例如修改为115200启动：

/usr/local/bin/sync\_auto 115200 > /var/log/miivii\_sync.log &

如果不填写波特率参数，则默认为9600

支持波特率参数如下：

2400 4800 9600 57600 115200 460800

## EXPANSION PORT ②接口信号定义

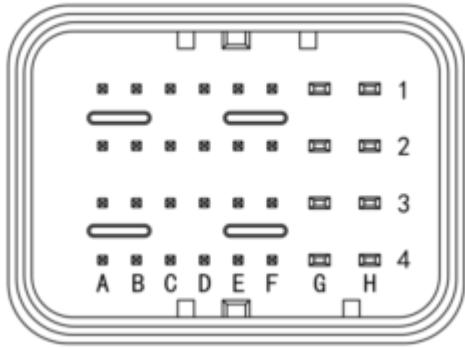


图 EXPANSION PORT②接口序号图

接口名称	引脚序号	接口信号定义	接口说明
POWER	H1	VIN	电源正 (DC电源接口)
	H2	VIN	电源正 (航插电源接口, 棕色)
	H3	VIN	电源正 (航插电源接口, 灰色)
	H4	VIN	电源正
	G1	GND	电源负 (DC电源接口)
	G2	GND	电源负 (航插电源接口, 蓝色)
	G3	GND	电源负 (航插电源接口, 黑色)
	G4	GND	电源负
CAN_C	D3	CAN_C_L	CAN_C信号 低位数据线
	C3	CAN_C_H	CAN_C信号 高位数据线
CAN_D	B3	CAN_D_L	CAN_D信号 低位数据线
	A3	CAN_D_H	CAN_D信号 高位数据线
CAN_E	B1	CAN_E_L	CAN_E信号 低位数据线
	A1	CAN_E_H	CAN_E信号 高位数据线

8路 GPIO口	A2	GPIO_1	GPIO_IN信号
	C2	GPIO_13	
	E3	GPIO_12	
	F3	GPIO_11	
	E2	GPIO_24	GPIO_OUT信号
	F1	GPIO_26	
	D1	GPIO_33	
	F4	GPIO_7	
GND	B2	GND	地
	D2		
	F2		
	E1		
	E4		
	C1		
FAN	B4	GND	地
	A4	DC_5V	电源正
	D4	FAN_TACH	转速
	C4	FAN_PWM	调速

## 背面接口



图 Apex Xavier II+ 后侧接口示意图

接口	接口名称	接口说明
RJ45网口	LAN1	独立千兆网口, 向下兼容百兆网口 可选配IEEE 802.3at PoE+ 25.5W
RJ45网口	LAN2	
RJ45网口	LAN3	
RJ45网口	LAN4	
USB 3.0/Flash	USB 3.0/Flash	USB 3.0 (TYPE A) /烧写口
USB3.0	USB 3.0	USB 3.0 (TYPE A)

接口	接口名称	接口说明
HDMI	HDMI	HDMI 2.0 (TYPE A)

蜂鸣器	一声短嘀音：上电后，电源正常 一声长嘀音：系统处于刷机模式 循环五声短嘀音：系统启动不成功
-----	---

## IO转接线说明

Apex Xavier II+附带两根IO转接线，对应EXPANSION PORT ①/PORT ②

### PORT 1转接线说明

EXPANSION PORT ①转接线有10个DB9端子和3个按键，功能如表所示：

序号	功能	个数	引出方式
1	UART_232	4	黑色DB9接头
2	UART_485	2	红色DB9接头
3	CAN(CAN FD)	2	白色DB9接头
4	PPS	1	蓝色DB9接头
5	同步信号	2	在同一个绿色DB9接头中
6	3个按键	3	1个RESET按键 1个RECOVERY按键 1个POWER_ON KEY按键

**#终板需要根据线束更改**



图 IO转接线实物图

### UART(232)连接线及引脚定义

Apex Xavier II+ 支持4路232串口通信，分别为UART(232)A, UART(232)B, UART(232)C。其中UART(DEBUG)是debug接口。转接线中使用黑色DB9端子引出，如图：



图 UART(232)转接线实物图

4路UART(232)的DB9端子引脚定义：



图 UART(232)接口序号图

接口名称	Pin	Signal	设备节点号
UART(DEBUG)	2	UART(DEBUG)_RX	DEBUG
	3	UART(DEBUG)_TX	
	5	GND	
UART(232)A	2	UART(232)A_RX	ttyUART_232_A
	3	UART(232)A_TX	
	5	GND	
UART(232)B	2	UART(232)B_RX	ttyUART_232_B
	3	UART(232)B_TX	
	5	GND	
UART(232)C	2	UART(232)C_RX	ttyUART_232_C
	3	UART(232)C_TX	
	5	GND	

## UART(422/485)连接线及引脚定义

Apex Xavier II+支持2路422/485串口通信，分别为UART(422/485)A, UART(422/485)B。RS422为全双工，RS485为半双工。转接线中使用红色DB9端子引出，如图：



图 UART(422/485)转接线实物图

2路UART(422/485)的DB9端子引脚定义：



图 UART(422/485)接口序号图

注：RS45只需接入DB9接头的PIN1, PIN2, PIN5 具体接线线序请参照下列表格

接口名称	Pin	Signal	设备节点号
UART(422/485)A	1	UART(422T+/485_A)	ttyUART_422_485_A
	2	UART(422T-/485_B)	
	3	UART(422R+)	
	4	UART(422R-)	
	5	GND	

<b>UART(422/485)B</b>	1	UART(422T+/485_A)	ttyUART_422_485_B
	2	UART(422T-/485_B)	
	3	UART(422R+)	
	4	UART(422R-)	
	5	GND	

## CAN(CAN FD)连接线及引脚定义

CAN总线带有CAN芯片，120Ω终端电阻

Apex Xavier II+支持5路CAN总线通信，在PORT 1中含有2路，分别为CAN\_A, CAN\_B。转接线中使用白色DB9端子引出

如图：



图 CAN转接线实物图

2路CAN总线的DB9端子引脚定义：



图 CAN接口序号图

CAN	
Pin	Signal
2	CAN_A_L
7	CAN_A_H

## PPS连接线及引脚定义

Apex Xavier II+支持一路PPS同步信号，波特率为9600。对应于IO转接线中一个深蓝色DB9端子，如图所示：

PPS 同步功能的使用方法请见“同步功能使用说明”中的“PPS同步模式”部分。



图 PPS同步转接线实物图

引脚定义：



图 PPS同步接口序号图

## PPS

Pin	Signal
1	GND
2	PPS_A_RX
3	PPS_A_TX
5	GND
6	PPS_A_SYNC

## SYNC连接线及引脚定义

Apex Xavier II+支持1路的Sync-out和1路的Sync-in同步信号。对应IO转接线中一个墨绿色DB9端子，如图所示：



图 Sync IO转接线实物图

SYNC\_IO的DP9端子引脚定义：



图 Sync IO接口序号图

## Sync同步信号

Pin	Signal
1	SYNC_IN_A
2	SYNC_OUT_A
3	NC
6	GND
7	GND
8	GND

[11] Sync-out与Sync-in 同步功能的使用方法请见 “同步功能使用说明” 中的 “Sync out 同步模式” 与 “Sync in 同步模式” 部分。

## 功能按键

Apex Xavier II+转接线中提供了3个功能按键，分别为RESET按键，POWER\_ONKEY按键和FORCE\_RECOVERY按键。



图 功能按键实物图

按键名称	按键功能	颜色区分
RESET按键	设备重新启动	白色
POWER_ONKEY按键	设备启动	红色
FORCE_RECOVERY按键	进入刷机模式	黑色

## PORT 2转接线说明

EXPANSION PORT ②转接线有2个电源端子，5个DB9端子和1个风扇，功能如表所示：

序号	功能	个数	引出方式
1	电源输入	1	1个5x2.5 DC电源头 1个4pin航插电源头
2	CAN(CAN FD)	3	3个DB9接头
3	GPIO	8	2个DB9接头
4	FAN	1	1个4芯5v-PWM风扇接头

#线束接头颜色待更改



图 IO转接线实物图

## 电源接口

Apex Xavier II+转接线中提供了2路电源输入接口，2路接口为电气并联。



图 DC电源接口实物图



图 航插电源接口实物图

Pin	信号定义	接口说明
1	VIN	直流电源正
3	VIN	直流电源正
2	GND	直流电源负
4	GND	直流电源负

## 航插电源转接线



**图 航插电源转接线实物图**

Apex Xavier II+转接线中**航插电源转接线**线序说明如下，与EXPANSION PORT ②接口信号对照详见接口说明。

颜色	信号定义	接口说明
棕	VIN	直流电源正
灰	VIN	直流电源正
蓝	GND	直流电源负
黑	GND	直流电源负

### CAN(CAN FD)连接线及引脚定义

CAN总线带有CAN芯片，120Ω终端电阻

Apex Xavier II+支持5路CAN总线通信，在PORT 2中含有3路，分别为CAN\_C, CAN\_D, CAN\_E。转接线中使用白色DB9端子引出

如图：



**图 CAN转接线实物图**

3路CAN总线的DB9端子引脚定义：



**图 CAN接口序号图**

CAN_C		CAN_D		CAN_E	
Pin	Signal	Pin	Signal	Pin	Signal
2	CAN_C_L	2	CAN_D_L	2	CAN_E_L
7	CAN_C_H	7	CAN_D_H	7	CAN_E_H

### GPIO连接线及引脚定义

Apex Xavier II+支持8路GPIO串口通信，由2路DB9端子引出，分别为GPIO\_A, GPIO\_B。转接线中使用绿色DB9端子引出。

如图所示：



**图 GPIO转接线实物图**

5路GPIO\_A的DB9端子引脚定义：



**图 GPIO\_A接口序号图**

接口名称	DB9针脚序号	信号定义	接口说明	引脚号
GPIO_A	1	GPIO_1	GPIO IN	339
GPIO_B	2	GPIO_13	GPIO IN	443
GPIO_C	3	GPIO_24_3V3	GPIO OUT	387
GPIO_D	4	GPIO_26_3V3	GPIO OUT	390
GPIO_E	5	GPIO_33_3V3	GPIO OUT	413
GND	6-9	GND	地	

3路GPIO\_B的DB9端子引脚定义：



**图 GPIO\_B接口序号图**

接口名称	DB9针脚序号	信号定义	接口说明	引脚号
GPIO_F	3	GPIO_12	GPIO IN	241
GPIO_G	5	GPIO_11	GPIO IN	240
GPIO_H	1	GPIO_7_3V3	GPIO OUT	238
GND	6/8	GND	地	

### 风扇连接线及引脚定义

Apex Xavier II+支持外接1路5V-PWM风扇输出。

如图所示：



**图 风扇链接线实物图**

颜色	信号定义	接口说明
黑	GND	地
红	DC_5VFAN	电源正
黄	FAN_TACH	转速
蓝	FAN_PWM	调速

## 扩展设备的安装方式

Apex Xavier II+提供SSD硬盘、WIFI或4G模块、SIM卡的硬件接口，为扩展功能使用。设备壳体整体有防尘防水处理，为避免影响设备整体防尘防水效果，建议在设备选购时选择预装相应扩展模块的设备。



## Mini PCIe 4G支持清单

序号	品牌	产品型号	支持方式	使用接口	模块功能	工作温度	规格	备注
1	移远	EC20-CEHCLG-MINIPCIE-CB	正式	Mini PCIe	4G	-40°C 至 80°C	规格 全网通 速度 最大130Mbps (下载)/最大30Mbps (上传)	
2	移远	EC20-CEHC-MINIPCIE-CB	Beta	Mini PCIe	4G	-40°C 至 80°C	规格 全网通 速度 最大130Mbps (下载)/最大30Mbps (上传)	
3	移远	EC20-CEHCLG-MINIPCIE-C	Beta	Mini PCIe	4G	-40°C 至 80°C	规格 全网通 速度 最大130Mbps (下载)/最大30Mbps (上传)	

注：

1. 正式支持：每次米文系统版本升级，会在米文设备上进行验证。
2. BETA支持：米文调试过，但不会在每次米文系统版本升级中验证，如使用过程中需要进一步支持请联系对应的销售工程师或客户经理。

## Mini PCIe WIFI支持清单

序号	品牌	产品型号	支持方式	使用接口	模块功能	备注
1	Complex	WLE900VX	正式	Mini PCIe	WIFI	-7
2	Azurewave (海华)	AW-CB161H	正式	Mini PCIe	WIFI+蓝牙	07

注：

1. 正式支持：每次米文系统版本升级，会在米文设备上进行验证。
2. BETA支持：米文调试过，但不会在每次米文系统版本升级中验证，如使用过程中需要进一步支持请联系对应的销售工程师或客户经理。

## 功能介绍

### 通用使用方法

#### 系统介绍

米文设备采用Ubuntu系统。默认用户名：nvidia； 密码：nvidia

#### 烧写镜像

请访问米文技术论坛<http://forum.miivii.com/>来获取烧写工具，烧写工具说明及相应镜像。

#### 开关机

开机：米文设备默认开机模式为上电自启动。插入电源，并将显示器通过HDMI接口与米文设备相连，开机画面如图所示：



图 开机画面

关机：长按POWER键/ON KEY按钮关机。或在命令行中执行\$ sudo poweroff，完成软关机  
重启：在命令行中执行\$ sudo reboot，完成重启

## MIVII SETTINGS的使用说明

### 简介

MIVII SETTINGS（又称米文设置），是米文为了简化对于设备进行设置，而提供的工具。提供诸如系统状态检测、远程访问、远程登陆等等功能。

Jetpack 4.4版本及以下镜像MIVII SETTINGS的使用说明

请参考：<http://doc.miivii.com/8847447.html>

### 使用视频

[MIVII SETTINGS使用视频.mp4](#)

[MIVII SETTINGS使用视频.mp4 \(备用链接\)](#)

### 登陆

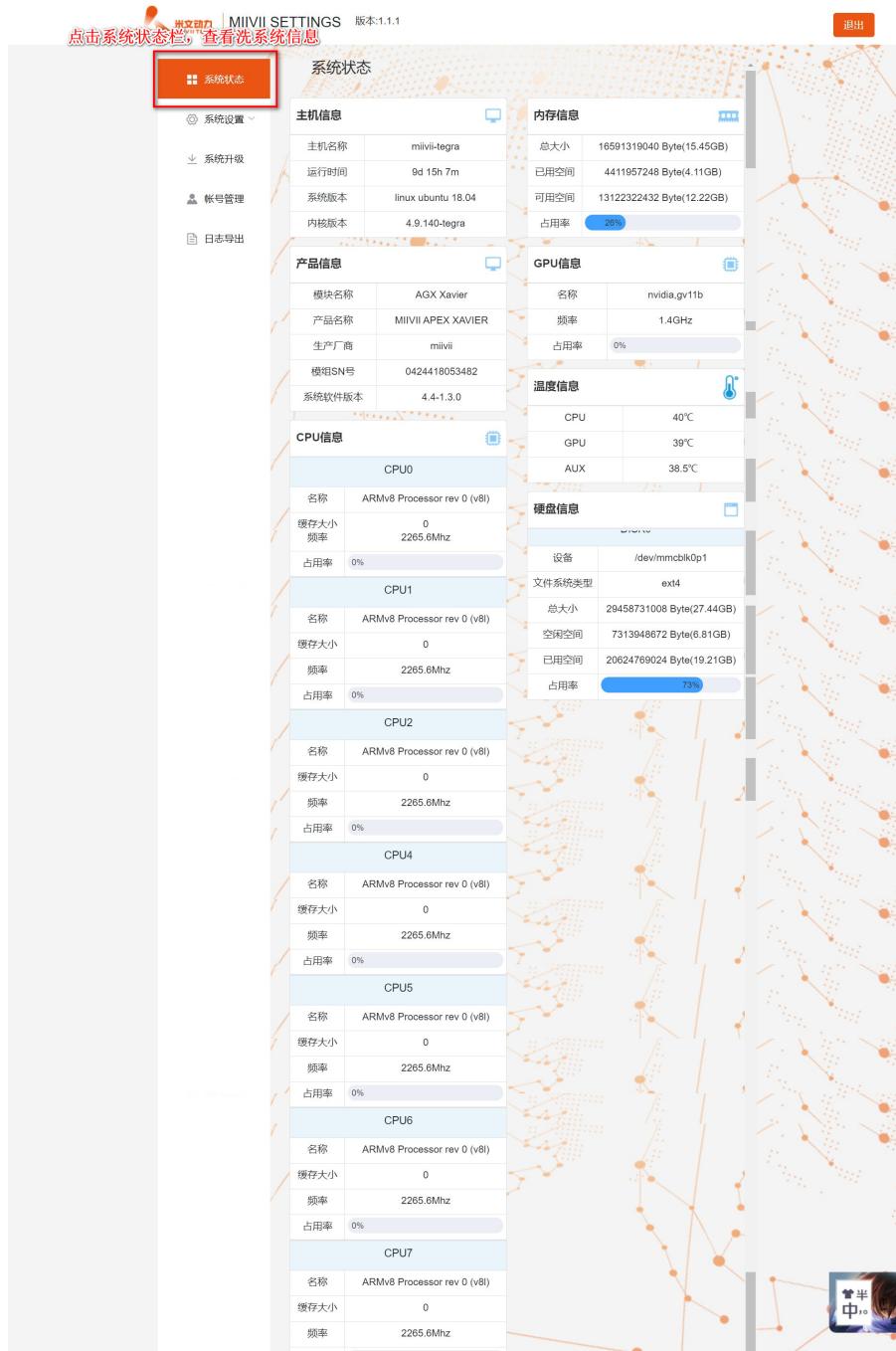
默认端口号为3000

用户名密码，为系统用户名密码。具有sudo权限的用户方可登录。无sudo权限用户无法登陆。

# 功能说明

## 系统状态

用于查看当前系统CPU占比、内存占比，存储占比等基本信息。



所有设备配置软件均含有基本信息显示功能，在此可以看到设备的系统版本，也可以通过命令行查看系统版本。

```
cat /etc/miivii_release
APEX 4.2.2-1.5.0
```

## 系统设置

对系统基本功能进行设置，如系统授时设置，GMSL相机设置等等。

- GMSL设置



Apex Xavier以及S2Pro包含GMSL摄像头设置功能，  
Apex Xavier II系列, EVO TX2 GMSL2包含GMSL2摄像头设置功能，  
EVO Xavier, EVO TX2以及S2不包含GMSL/GMSL2摄像头设置功能。  
该功能可以按照GMSL/GMSL2摄像头的接插位置，分别设定摄像头的品牌。

设备Apex有两组GMSL摄像头接口，分别记为GMSL\_A与GMSL\_B，

设备S2Pro只有一组记为GMSL\_A，

设备EVO TX2 GMSL2有6个独立GMSL2摄像头接口，记为GMSL\_0-5

设备Apex Xavier II系列有8个独立GMSL2摄像头接口，记为GMSL\_0-7

#### GMSL摄像头配置方法

在初次接入GMSL摄像头以及更换GMSL摄像头型号时需要对配置文件进行更改，并重启设备。配置文件路径：/opt/miivii/config/gmsl\_camera/camera.cfg

MVGCB-001A : Entron MVGCB-002A: Calmcar MVGCB-003A: Adayo MVGCB-006A: Sensing 默认配置：A组和B组摄像头接口默认配置都是MVGCB-001A

- 系统授时设置



Apex Xavier, Apex Xavier II系列, EVO Xavier, EVO TX2 GMSL2, S2Pro包含同步功能设置，, EVO TX2 ,S2不包含同步功能设置。

其中，NTP为默认模式。NTP网络授时模式，此时设备接入网络，可以被NTP服务授时。

同时，设备可作为同步源，外接支持同步功能的传感器，并进行同步；GPS为GPS外界授时模式，此时设备外接GPS，可以被GPS授时。同时，设备可作为同步源，外接支持同步功能的传感器，并进行同步；None为不同步模式，此时设备不被外界授时，但可以作为同步源。同时，也可在此调节Sync out输出频率，注意此处并非是GMSL的频率。

设备授时模式与Sync out输出频率配置方法 设备授时模式与Sync out输出频率的调整需要对配置文件进行修改，并重启设备。配置文件路径：/opt/miivii/config/config/sync.cfg 授时模式是通过修改其中"sync\_type:X" 的X数值来实现。0：GPS外界授时模式 1：NTP网络授时模式 2：不同步模式 Sync out输出频率通过修改其中"sync\_out\_freq:XX"的XX数值实现Sync out频率调节。该调节仅支持整数。

```
cat /opt/miivii/config/config/sync.cfg
sync_out_freq:25
sync_type:2
/*
note:
sync_out_freq---the frequency is 25 for sync out time
sync_type---0 is for GPS calibrate time
1 is for SYS calibrate time
2 can not calibrate time
```

## 系统升级

对cuda相关包，以及miivii提供的相关包和系统进行升级及回退。



## 账号管理

用于绑定米文边缘服务。



## 日志导出

用于导出/var/log/中的日志，便于进行售后分析。



## 功率模式设定

搭载Jetson AGX Xavier的米文设备有多工作模式。可以通过右上角的NVIDIA绿色标志设置进行调整。米文设备的默认模式为3: MODE\_30W\_ALL



点击下拉菜单即可对米文设备的工作模式进行修改，工作模式的细节详见下表：

Mode Name	EDP	10W	15W	30W	30W	30W	30W
	MAXN	MODE_10W	MODE_15W	MODE_30W_ALL	MODE_30W_6CORE	MODE_30W_4CORE	MODE_30W_2CORE
Power Budget	n/a	10W	15W	30W	30W	30W	30W
Mode ID	0	1	2	3	4	5	6
Number of Online CPUs	8	2	4	8	6	4	2
CPU Maximal Frequency (MHz)	2265.6	1200	1200	1200	1450	1780	2100
GPU TPC	4	2	4	4	4	4	4
GPU Maximal Frequency (MHz)	1377	520	670	900	900	900	900
DLA Cores	2	2	2	2	2	2	2
DLA Maximal Frequency (MHz)	1395.2	550	750	1050	1050	1050	1050
Vision Accelerator (VA) cores	2	0	1	1	1	1	1
VA Maximal Frequency (MHz)	1088	0	550	760	760	760	760
Memory Maximal Frequency (MHz)	2133	1066	1333	1600	1600	1600	1600

也可采用命令行调整：

```
#查看设备现在的模式
sudo nvpmodel -q verbose
# 设定为某一模式
sudo nvpmodel -m <MODE_ID>
#获取在当前模式下的最佳表现
sudo jetson_clocks
#查看详细信息
sudo jetson_clocks --show
```

## IO使用方法

### GPIO接口配置方法

对GPIO接口使用的示例如下，请将<>中的信息修改为想要调整的GPIO节点号，具体对应关系请参考【接口说明】部分

```
# 切换到root用户
sudo su -
# 设置为高电平(DO)
```

```

echo 1 > /sys/class/gpio/<gpio339>/value
# 设置为低电平(DO)
echo 0 > /sys/class/gpio/<gpio339>/value
# 读取数据(DI)
cat /sys/class/gpio/<gpio339>/value

```

若需要关机后保留配置，可以将以上命令写入/etc/rc.local文件

## 米文设备GPIO输出模式说明

DO 输出 模式	模式说明	对应米文设备
开漏输出	不输出电压，控制输出低电平时引脚接地，控制输出高电平时引脚既不输出高电平，也不输出低电平，为高阻态。如果外接上拉电阻，则在输出高电平时电压会拉到上拉电阻的电源电压。设置为高电平时，DO脚与外接的电压相同(0-40V)；设置为低电平时，DO脚为地	Evo Xavier
推挽输出	内部自带负载电阻，可以稳定输出电平信号。高电平时，DO脚稳定输出3.3V电压，低电平时，DO脚输出为0V。推挽输出最大支持电流为10mA。	Apex Xavier, Apex Xavier II 系列 Evo TX2 GMSL2 Lite NX Mini, Lite TX2 NX Mini, Lite Nano Mini

注：开漏输出推荐上拉电阻表

目标 上拉 电压 (V)	3.3	5	12	15	24	36	40
推荐 上拉 电阻 值 (Ω)	500	1k	2k	3k	5k	10k	10k

## UART接口配置方法

打开/dev/(folder)下面对应的设备节点，设置波特率，停止位，奇偶校验位，数据位等。可以使用stty命令配置串口的波特率，停止位，奇偶校验位，数据位等，详细见stty命令说明。

命令示例，请将< >中的信息修改为想要调整的串口节点号，具体对应关系请参考【接口说明】部分

```
sudo stty -F /dev/<UART_XXX> speed 115200 cs8 -parenb -cstopb -echo
```

### 输出数据测试

```
sudo echo "miivii tty debug" > /dev/<UART_XXX>
```

### 使用下面命令接收输入数据

```
sudo cat /dev/<UART_XXX>
```

## GPS 对设备授时使用方法

GPS对设备授时功能优点：设备通过GPS设备从GPS卫星上获取当地标准的时间信号，从而精准定位设备时间。

### GPS支持型号

支持GPS品牌型号：所有符合GPRMC数据标准格式输出的GPS设备，且必须要有PPS秒脉冲输出的GPS设备

### 连接方式

参照手册的“接口说明”

### 授时功能配置

在初次接入GPS时需要在MiiVii Setting配置软件中进行系统配置，将Sync Mode选项配置成GPS模式，重启系统。MiiVii Setting具体方法请参考“米文配置软件介绍”部分。

### 检查授时是否成功

修改系统时间，输入命令

```
sudo date -s "2018-10-1"
```

等待2~3s，查看当前时间，输入命令

```
date
```

若显示时间为：“2018-10-1”，说明授时失败

若显示时间为：“当前时间”，说明授时成功

### 故障排查

若授时失败，需进行故障排查

#### 1. 查看GPS是否有输出

输入命令

```
cat /dev/ttyTHS1
```

终端收到带有GPRMC字段的输出，例如：

GPRMC,014600.00,A,2237.496474,N,11356.089515,E,0.0,225.5,310518,2.3,W,A\*23

## 2.查看GPS的pps信号是否有输出

输入命令

```
hexdump /dev/miivii-sync-in-a
```

终端有十六进制的数据输出，例如：

0000400 02fe 9f40 490e 562d 1647 004e 0000 0000

## 3.识别方法

如果以上"1"+"&"2"没有输出，说明GPS工作不正常，可以把GPS放到窗外或是到户外测试，或更换GPS进行测试

如果"1"+"&"2"输出正常，检查Miivii Setting配置是否为GPS模式，如果不是，更改模式后重新启动

执行以上操作之后，GPS授时依然不成功，输入命令

```
hexdump /dev/miivii-sync-out
```

终端有十六进制的数据输出，例如：

0000400 02fe 9f40 490e 562d 1647 004e 0000 0000

如果没有数据输出，可能是没有用匹配的刷机工具和镜像刷机，建议检查镜像和刷机工具重新刷机

如果有数据输出，可能是设备硬件问题，建议联系售后维修处理

## CAN口配置方法

CAN10设备具体使用方法，参考<https://github.com/linux-can/can-utils>中的cansend.c和candump.c

测试命令：

```
sudo modprobe can

sudo modprobe can_raw

sudo modprobe mttcan

sudo ip link set can0 type can bitrate 500000 sjw 4 berr-reporting on

sudo ip link set up can0

sudo cansend can0 123#abcdabcd

sudo candump can0

sudo ip -details -statistics link show can0
```

```
sudo ifconfig can0 down
```

## CAN FD配置使用方法：

```
sudo modprobe can  
sudo modprobe can_raw  
sudo modprobe mttcan  
sudo ip link set can0 type can bitrate 500000 sjw 4 dbitrate 2000000  
sudo ip link set up can0  
sudo cansend can0 213##011
```

## [10] CAN FD和CAN 2.0的区别：

1)

```
sudo ip link set can0 type can bitrate 500000 dbitrate 2000000 berr-re
```

其中bitrate为can2.0模式下的波特率； dbitrate为can fd模式下的波特率，根据官方文档，这个值最大可配置为5M，一般应用最好采用2M；

2)

```
sudo cansend can0 213##011
```

发送命令中， id与数据之间多了一个#，并且##后的第一个字节(0)为canfd\_frame.flags的值，范围为0~F； canfd\_frame.flags后面的字节(11)为第一个数据，一次最多可以传输64个字节。

## 扩展CAN FD口配置方法

### 扩展CAN FD配置使用方法：(适配产品APEX2: CAN2 CAN3 CAN4)

```
#can2  
sudo ip link set can2 up type can bitrate 500000 dbitrate 5000000 res  
sudo ifconfig can2 txqueuelen 65536  
  
#can3  
sudo ip link set can3 up type can bitrate 500000 dbitrate 5000000 res  
sudo ifconfig can3 txqueuelen 65536  
  
#can4
```

```
sudo ip link set can4 up type can bitrate 500000 dbitrate 5000000 res  
sudo ifconfig can4 txqueuelen 65536
```

## 扩展设备配置方法

### 扩展SSD硬盘使用

查看硬盘信息:

```
sudo fdisk -lu
```

```
Disk /dev/mmcblk0boot1: 8 MiB, 8388608 bytes, 16384 sectors  
Units: sectors of 1 * 512 = 512 bytes  
Sector size (logical/physical): 512 bytes / 512 bytes  
I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk /dev/mmcblk0boot0: 8 MiB, 8388608 bytes, 16384 sectors  
Units: sectors of 1 * 512 = 512 bytes  
Sector size (logical/physical): 512 bytes / 512 bytes  
I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk /dev/nvme0n1: 232.9 GiB, 250059350016 bytes, 488397168 sectors  
Units: sectors of 1 * 512 = 512 bytes  
Sector size (logical/physical): 512 bytes / 512 bytes  
I/O size (minimum/optimal): 512 bytes / 512 bytes  
Disklabel type: dos  
Disk identifier: 0xd80dc8f5  
nvidia@miivii-tegra:~$
```

图 查看硬盘信息页面截图

格式化硬盘:

```
sudo mkfs -t ext4 /dev/nvme0n1
```

```
nvidia@miivii-tegra:~$ sudo mkfs -t ext4 /dev/nvme0n1  
mke2fs 1.42.13 (17-May-2015)  
Found a dos partition table in /dev/nvme0n1  
Proceed anyway? (y,n) y
```

图 格式化硬盘截图

查看硬盘UUID:

```
sudo blkid /dev/nvme0n1
```

```
nvidia@miivii-tegra:~$ sudo blkid /dev/nvme0n1  
/dev/nvme0n1: UUID="6e643050-77bb-40d3-97b4-7835fc016afb" TYPE="ext4"  
nvidia@miivii-tegra:~$
```

图 查看硬盘UUID 截图

**开机自动挂载硬盘的设置方法：在/etc/systemd/system路径下创建一个systemd服务，用来开机自动执行挂载硬盘，如：miivii\_mount\_ssd.service**

```
#创建服务miivii_mount_ssd.service
vim miivii_mount_ssd.service
[Unit]
Description=MIIVII specific script
After=udev.service

[Service]
ExecStart=/etc/systemd/miivii_mount_ssd.sh

[Install]
WantedBy=multi-user.target
```

**在/etc/systemd/路径下创建一个脚本，用来挂载硬盘，如：miivii\_mount\_ssd.sh**

```
#创建服务脚本miivii_mount_ssd.sh
vim miivii_mount_ssd.sh
#!/bin/bash
mount -o rw /dev/nvme0n1 /home/nvidia/workspace
```

**为创建的脚本文件添加可执行权限**

```
sudo chmod +x miivii_mount_ssd.sh
```

**将挂载硬盘的服务设置为开机自启动**

```
sudo systemctl enable miivii_mount_ssd.service
```

## 无线设备配置方法

### WiFi配置方法

米文S2, S2Pro, EVO TX2, EVO TX2 GMSL2自带WiFi功能。米文Apex Xavier, Apex Xavier II系列, EVO Xavier, Lite NX, Lite Nano的WiFi功能由外接扩展模块提供，请按照【**扩展设备安装方式**】的内容对WiFi模块进行安装。请在开机Ubuntu系统桌面右上角网络连接图标中，找到要连接的WiFi名称并点击，然后在弹出的密码框输入密码并点击连接即可。



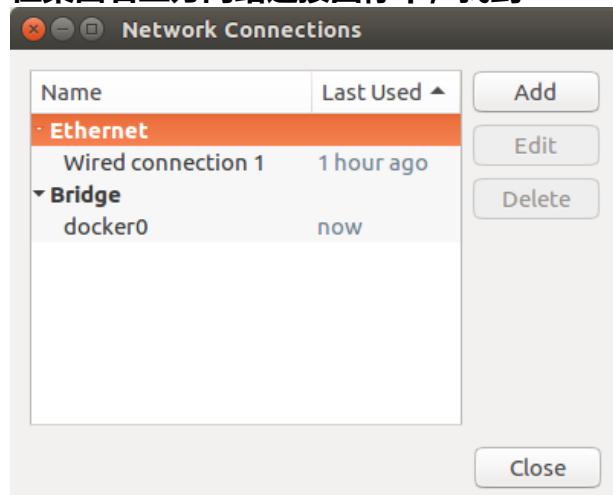
图 WiFi连接截图

## 4G模块配置方法

米文所有标准产品中不包含4G模块，需要用户自行另配。请按照【扩展设备安装方式】的内容对SIM卡以及4G模块进行安装。请注意如果您使用的是物联网SIM卡，则会出现SIM卡与设备硬件绑定的问题，请提前与通讯供应商确认。

米文的系统镜像中，整合了相应4G模块驱动。安装好4G模块后系统会自动识别。查看/dev目录，会看到/dev/ttyUSB0~/dev/ttyUSB3，一共4个设备。

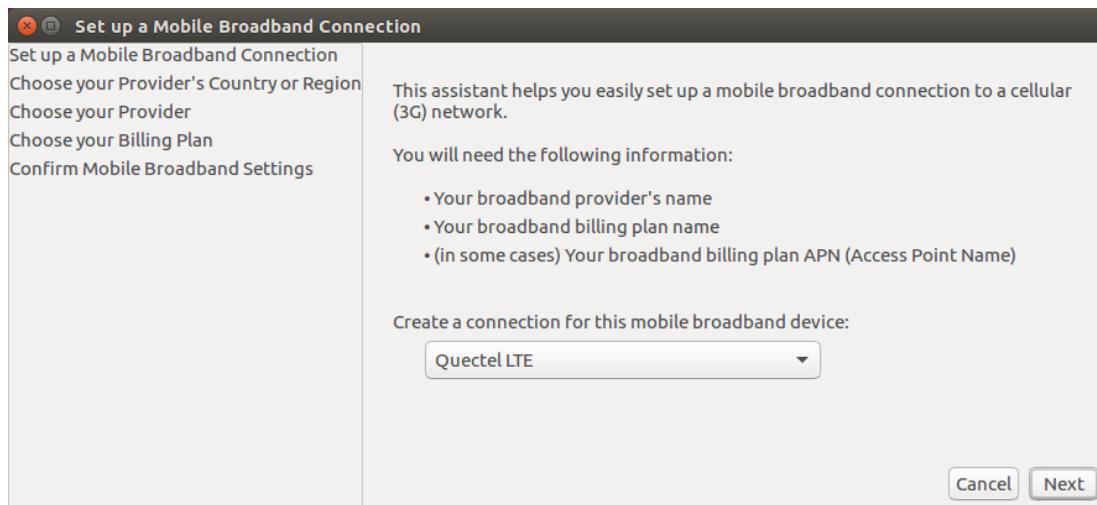
在桌面右上方网络连接图标中，找到Edit Connections，点击add，如图所示：



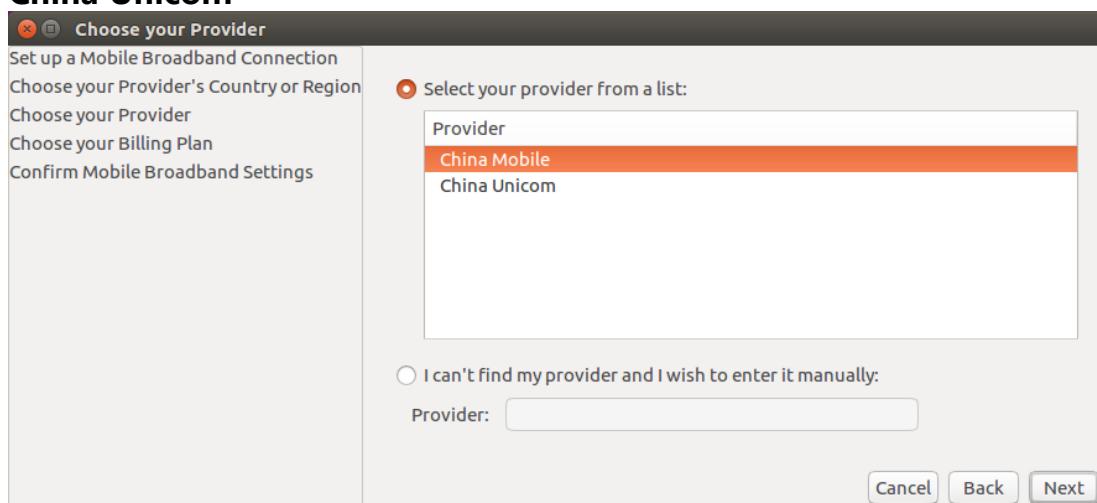
选择连接类型 Mobile Broadband



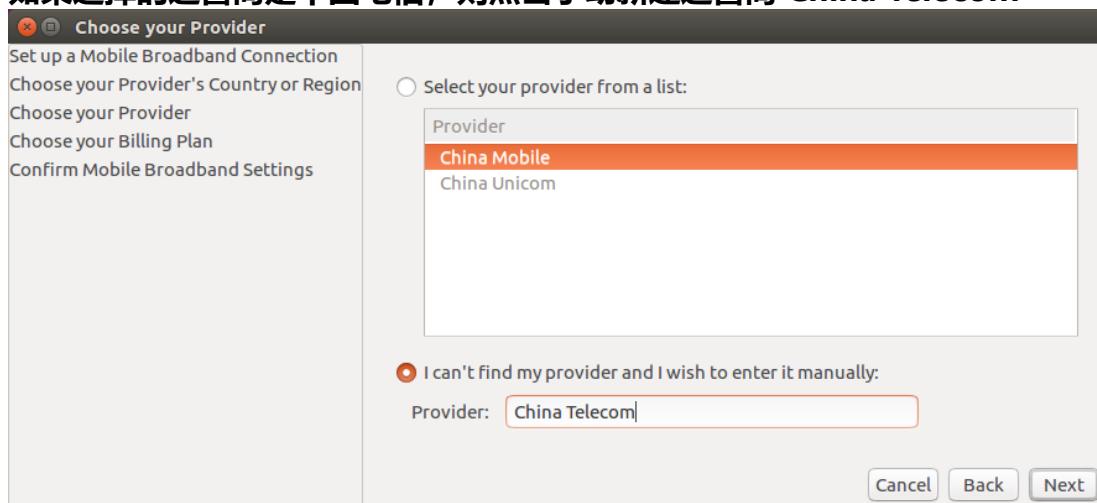
选择Next (选项Quectel LTE, Fibocom NL668 Modem, Android与Any device etc依据不同型号的4G模块显示不同信息，可直接点击Next)



**选择国家为China , 然后根据SIM卡选择运营商：中国移动是China Mobile, 中国联通是China Unicom**

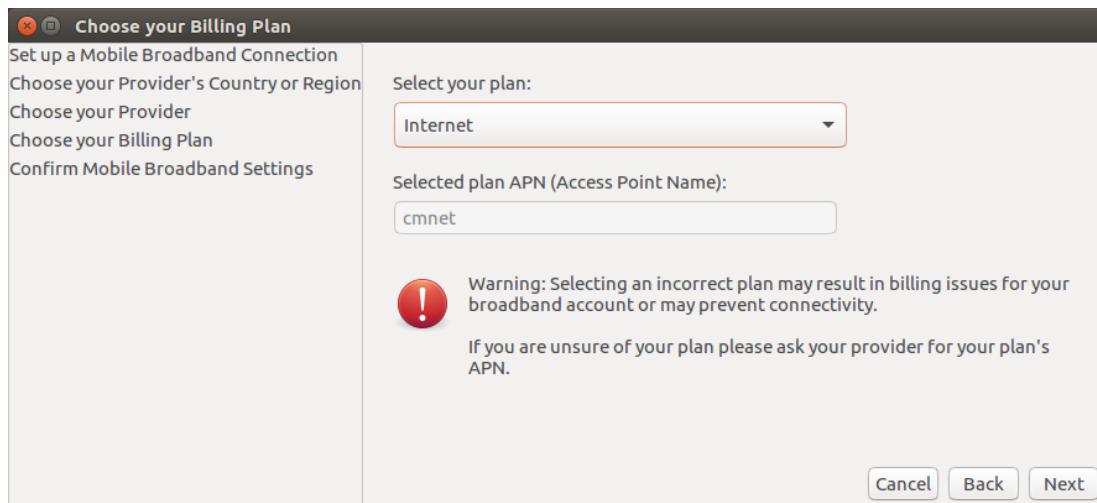


**如果选择的运营商是中国电信，则点击手动新建运营商 China Telecom**

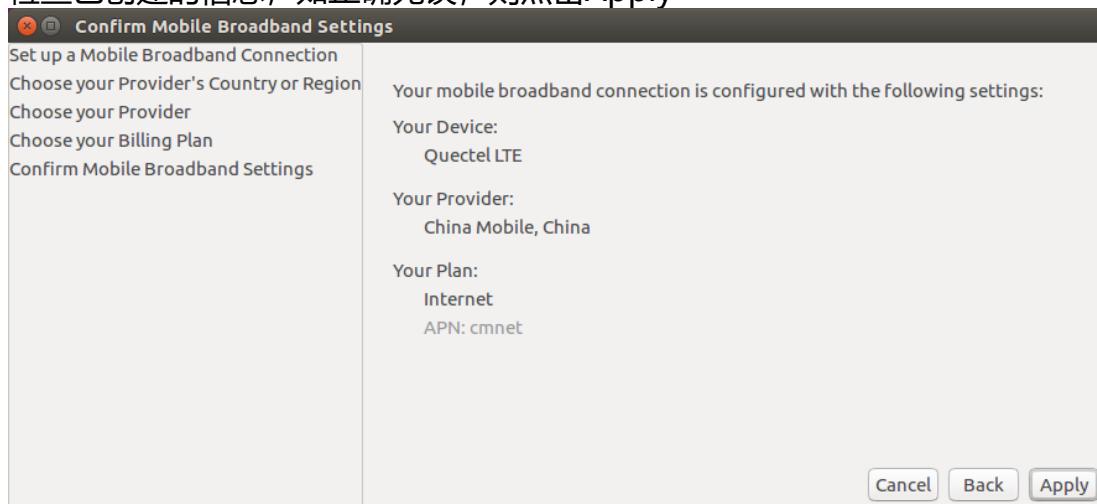


**选择你的Plan**

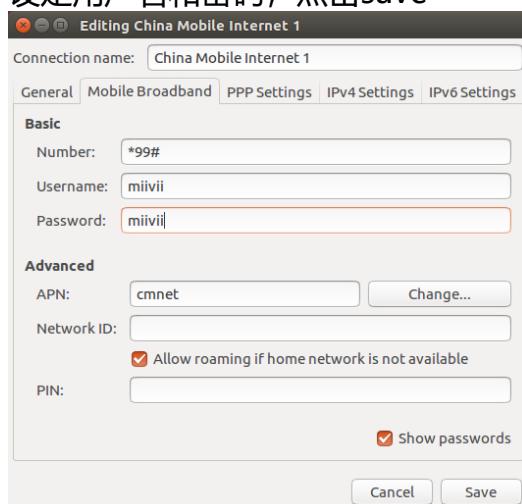
请根据SIM卡信息选择。移动选Internet, 联通和电信选default  
这里注意一下APN移动为cmnet, 联通为3gnet, 电信为ctnet



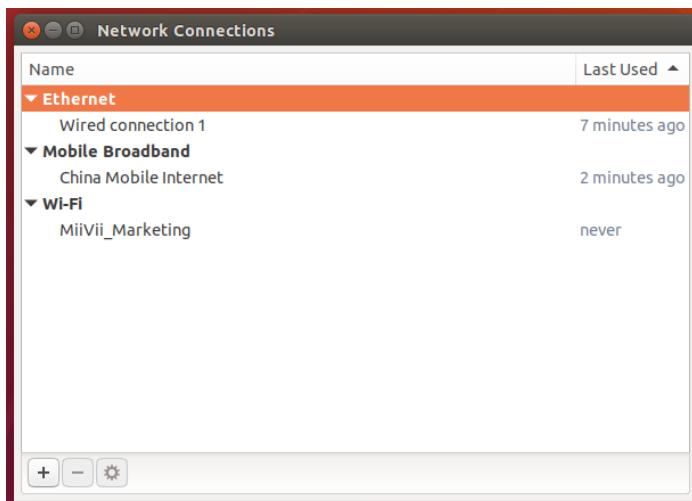
检查已创建的信息，如正确无误，则点击Apply



设定用户名和密码，点击save

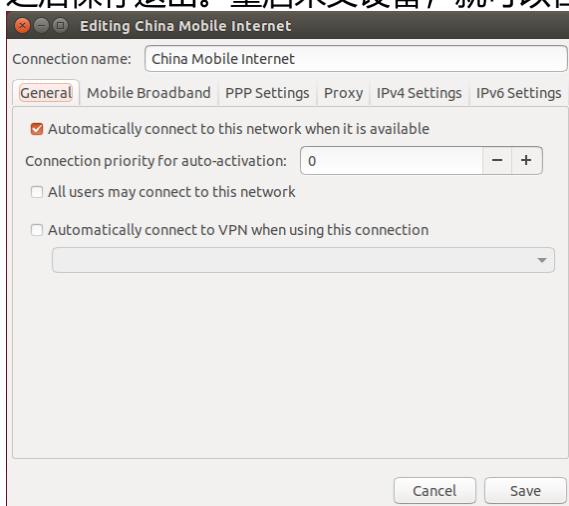


网络创建完成后，在桌面右上方网络连接图标中选中新建的网络连接，就能够正常上网了。若需要4G网络开机自动连接设置，已移动为例，建立好连接文件China Mobile Internet后操作如下：点击桌面上方网络连接图标，在下拉菜单中点击Edit Connections选项。在弹出的窗口中选中China Mobile Internet选项，点击下方设置图标



弹出的窗口中选择General选项，并勾选Automatically connect to this network when it is available选项

之后保存退出。重启米文设备，就可以在输入开机密码后自动重连4G网络



## 同步功能使用说明

### 同步功能介绍

设备支持三种同步方式，分别是：PPS，Sync in和Sync out同步。同步误差为0.1-1μs。（同步误差测试方法详见附录）

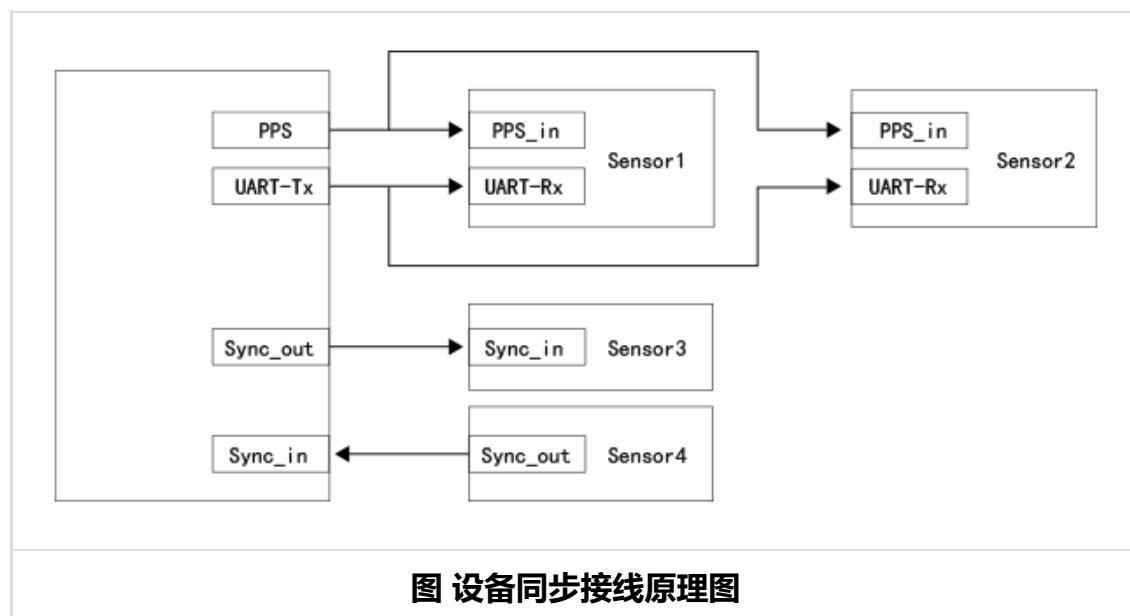


图 设备同步接线原理图

### 同步功能使用方法

## PPS同步模式

设备输出PPS信号<sup>1</sup>（每秒产生一个脉冲，脉宽50ms），并通过串口（UART/RS232）的Tx引脚发送该脉冲上升沿产生时间的NMEA GPRMC消息，消息示例：

```
$GPRMC,060249.000,A,3949.63046,N,11616.48565,E,0.296,,291118,,,A*4d
```

[1] PPS 信号的硬件连接方式请见“IO转接线说明”中的“PPS连接线及引脚定义”部分  
其中“060249.000”为每秒产生脉冲时的时间戳(UTC时间)，格式为“时分秒.000”，正常时间都是整秒格式。支持PPS同步模式的传感器会通过收到的PPS以及GPRMC消息对自身时钟系统进行校时，使之与设备的系统时钟保持一致。传感器的采样时间会作为时间戳（timestamp），与数据一起被发送至设备。至此，系统获取了传感器采样的系统时间，完成同步。

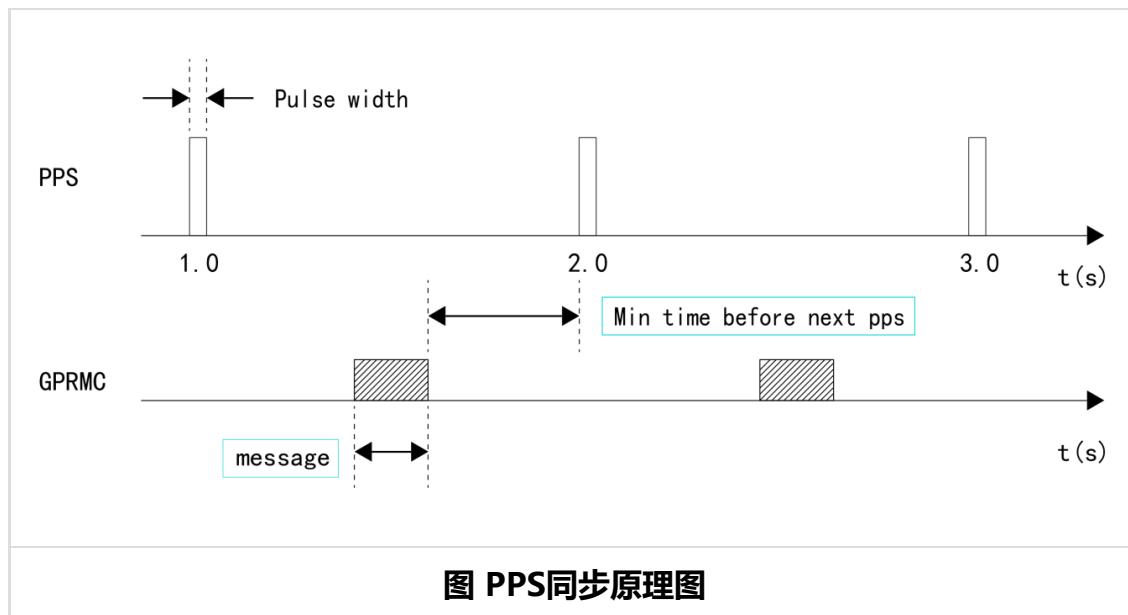


图 PPS同步原理图

同步功能验证方法（以RS-LiDAR-16激光雷达为例）：

当传感器只有数据输入接口与设备相连，未连接设备的PPS\_A\_SYNC口和PPS\_A\_TX时，传感器的ROS node向操作系统上传数据中的时间戳为硬件时间戳（hardware timestamp），即传感器内部时钟计器的时间（多数传感器会设定一个固定的初始时间作为计时起点，每次上电后开始计时）。此时在Ubuntu操作系统中打印该硬件时间戳，并与设备的系统时间进行比较，可发现二者的偏差较大。

```
[ INFO] [1544594922.929736448]: Got param time_mode: gps
ros time:1544594922.876364231
lidar time:1483229224.839233000

[ INFO] [1544594923.026582016]: Got param time_mode: gps
ros time:1544594922.977174997
lidar time:1483229224.940033000

[ INFO] [1544594923.134761184]: Got param time_mode: gps
ros time:1544594923.078029156
lidar time:1483229225.040833000

[ INFO] [1544594923.229261088]: Got param time_mode: gps
ros time:1544594923.178819418
lidar time:1483229225.141632000

[ INFO] [1544594923.332589472]: Got param time_mode: gps
ros time:1544594923.279591560
lidar time:1483229225.242432000
```

## 图 RS-LiDAR-16未同步时硬件时间戳与系统时间戳对比

当传感器连接设备的PPS\_A\_SYNC及PPS\_A\_TX后，传感器的ROS node向操作系统上传数据中的硬件时间戳为传感器内部时钟经过PPS授时后的时间，与设备的系统时间一致。此时在Ubuntu操作系统中打印接收到的数据，与收到该数据时的系统时间（`ros::time::now`）比较，当二者的差值小于100ms时，说明PPS功能生效。

```
[ INFO] [1544601870.404294176]: Got param time_mode: gps  
ros time:1544601870.349020720  
lidar time:1544601870.347184000  
  
[ INFO] [1544601870.503913024]: Got param time_mode: gps  
ros time:1544601870.449862003  
lidar time:1544601870.447984000  
  
[ INFO] [1544601870.602763072]: Got param time_mode: gps  
ros time:1544601870.550686121  
lidar time:1544601870.548784000
```

图 数据时间戳与系统时间对比

### Sync out 同步模式

设备支持Sync out同步信号<sup>2</sup>

[2] Sync out信号的硬件连接方式请见“IO转接线说明”中的“SYNC连接线及引脚定义”部分。设备可以通过Sync out引脚输出1-30Hz，脉宽5ms的脉冲信号，用于触发外部传感器启动采样。同时设备会记录该脉冲上升沿的产生时间。传感器完成采样后，设备会将记录的时间与本次传感传回的数据做关联，作为该数据的时间戳，至此，系统获取了传感器采样的系统时间，完成同步。

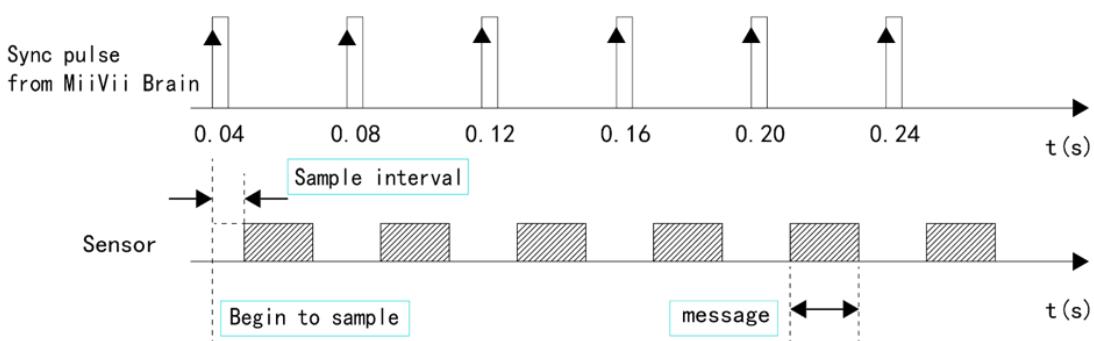


图 Sync out同步原理图 (25Hz)

与此同时，设备还提供GMSL接口的Sync out同步功能，详见GMSL摄像头章节。  
同步功能验证方法：配置传感器为外部触发同步模式，通过ROSbag抓包确认传感器触发频率是否为sync.cfg中所设定的频率值。如果偏差小于1Hz，则说明Sync out功能生效。

### 数据传输时间分析

测试说明：设定Sync out发出信号的频率为10Hz，测量设备发出的Sync out信号的上升沿到设备接收到视频帧之间的时间间隔（transfer time）。

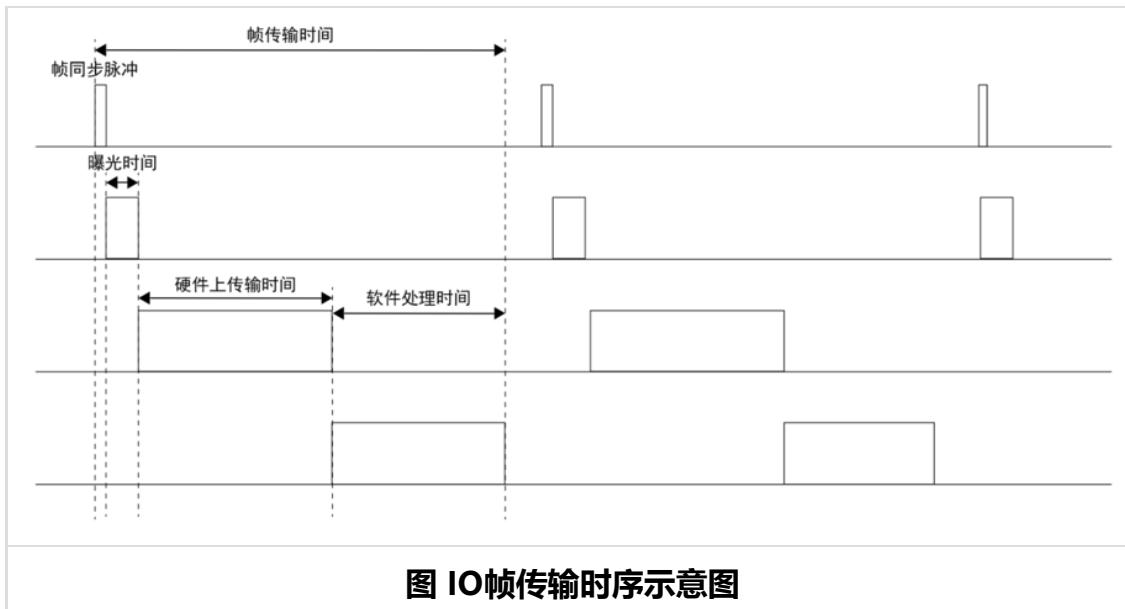


图 IO帧传输时序示意图

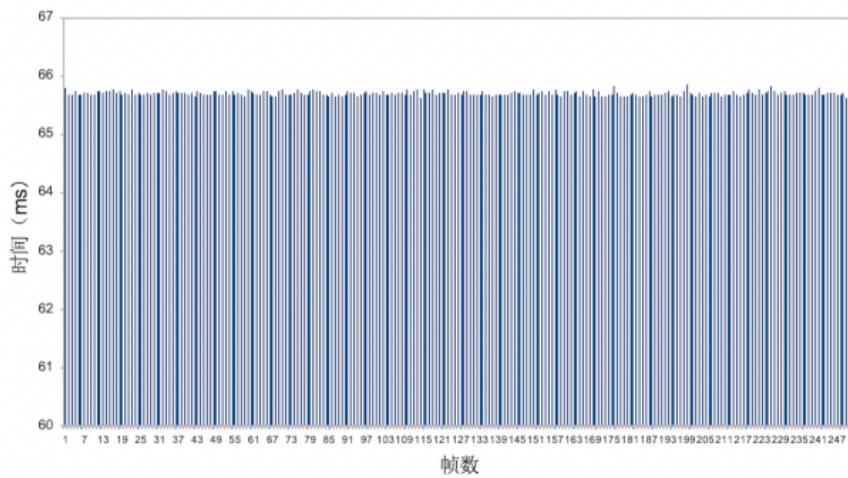


图 帧传输时间测试结果

测量结果发现帧传输时间的平均值为65.70ms。

### Sync in 同步模式

设备支持Sync in同步信号<sup>3</sup>。

[3] Sync in同步信号的硬件连接方式请见“IO转接线说明”中的“SYNC连接线及引脚定义”部分

支持Sync in同步模式的传感器，在启动采样的时刻会产生并发出一个脉冲信号。设备通过SYNC\_IN引脚接收脉冲信号，并记录该脉冲上升沿的产生时间。传感器完成采样后，设备会将记录的时间与本次传感传回的数据做关联，作为该数据的时间戳。至此，系统获取了传感器采样的系统时间，完成同步。

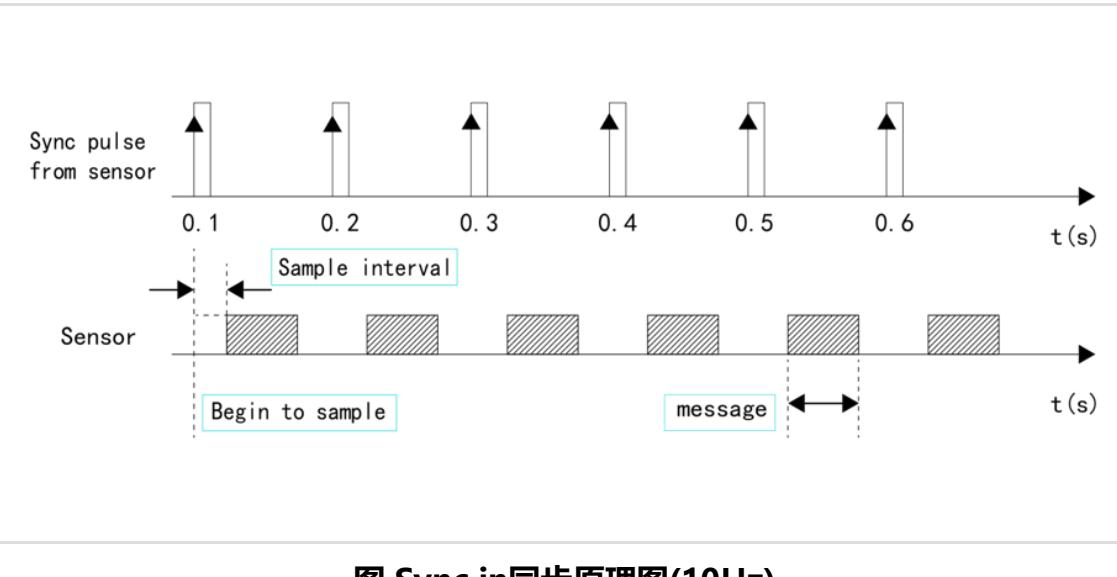


图 Sync in同步原理图(10Hz)

同步功能验证方法：在Ubuntu操作系统中打印SYNC\_IN引脚接收到脉冲信号的时间戳，将该时间戳与收到传感器数据帧的系统时间（`ros::time::now`）相比较，如果二者的差值小于100ms，说明Sync in功能生效。

## 同步误差测试方法

### 通过示波器测量PPS脉冲间隔

测量结果：

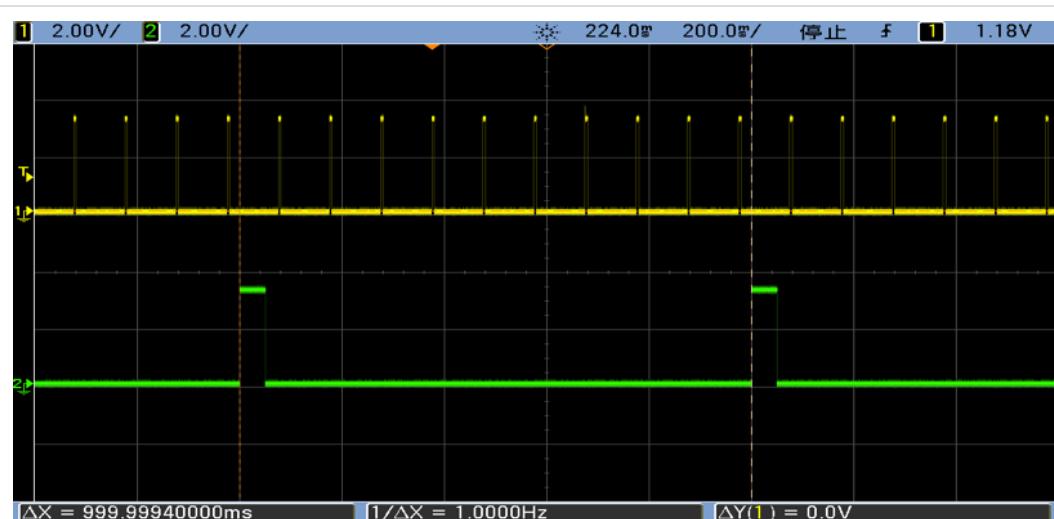


图 PPS脉冲间隔

	理论值 (μs)	实测值 (μs)	误差 (μs)
PPS	1000000	999999.4	0.6

### 通过示波器测量Sync out脉冲间隔

用示波器测量两个Sync out (10Hz) 脉冲之间的间隔并与理论值做比较。

测量结果：



图 Sync out脉冲间距

	理论值 ( $\mu$ s)	实测值 ( $\mu$ s)	误差 ( $\mu$ s)
Sync out	100000	99999.2	0.8

## 自行评估同步效果的方法

用户可以通过时间戳测量jitter，来自行评测设备同步效果。

### 同步sample code使用说明

MiiVii提供的sample code用于自行评估设备同步性能，其使用方法如下：

```
#进入下述路径
cd /opt/miivii/feature/sync_test/bin` 

#测试sync out 功能
./sync_out_test

#测试sync in 功能
./sync_in_test

#测试pps功能
./pps_test
```

### Sync out jitter测量

利用MiiVii提供的sample code (sync\_out\_test)，实时分析统计接收到的时间戳 (timestamp)，得到Sync out信号的频率、周期、平均误差、最大误差、方差等值，并实时打印。

```
You are checking the sync out mode
Time interval: the interval between two frames
Frequency measured : the Frequency measured by the Time interval
Max deviation : the difference between the maximum and the average
standard deviation : calculated by statistical data
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 32(ns) standard deviation: 9.055385
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 32(ns) standard deviation: 9.055385
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 32(ns) standard deviation: 9.055385
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 32(ns) standard deviation: 9.055385
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 32(ns) standard deviation: 9.055385
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 32(ns) standard deviation: 9.055385
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 32(ns) standard deviation: 9.055385
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 32(ns) standard deviation: 9.055385
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 32(ns) standard deviation: 9.055385
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 32(ns) standard deviation: 9.055385
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 32(ns) standard deviation: 9.055385
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 32(ns) standard deviation: 9.055385
```

图 Sync out示例测试结果

### Sync in jitter测量

由外部设备（如信号发生器）输出固定周期的脉冲信号，接入到设备的SYNC\_IN引脚。再利用MiiVii提供的sample code (sync\_in\_test) 实时分析统计接收到的时间戳 (timestamp)，得到Sync in信号的：频率、周期、平均误差、最大误差、方差等值，并实时打印。

在没有信号发生器的情况下可以将设备的SYNC\_IN与SYNC\_OUT的引脚相连，以SYNC\_OUT的输出作为25Hz的输入信号。

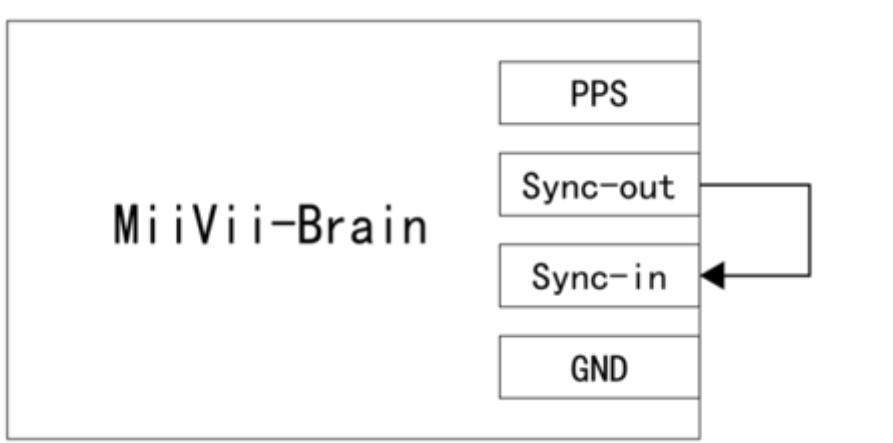


图 IO接线示意图

```
You are checking the sync in mode,please check the connecting line
Time interval: the interval between two frames
Frequency measured : the Frequency measured by the Time interval
Max deviation : the difference between the maximum and the average
standard deviation : calculated by statistical data
Time interval: 39999999(ns) Frequency measured: 25.000001(hz) Max deviation: 193(ns) standard deviation: 76.583288
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 193(ns) standard deviation: 65.984847
Time interval: 39999999(ns) Frequency measured: 25.000001(hz) Max deviation: 193(ns) standard deviation: 71.700767
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 193(ns) standard deviation: 71.693793
Time interval: 39999999(ns) Frequency measured: 25.000001(hz) Max deviation: 193(ns) standard deviation: 69.942834
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 193(ns) standard deviation: 61.269895
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 193(ns) standard deviation: 68.673139
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 193(ns) standard deviation: 78.115299
Time interval: 39999999(ns) Frequency measured: 25.000001(hz) Max deviation: 193(ns) standard deviation: 66.143783
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 193(ns) standard deviation: 77.051931
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 193(ns) standard deviation: 67.535176
```

图 Sync in示例测试结果

### PPS jitter测量

将设备的的PPS与SYNC\_IN引脚相连，利用MiiVii提供的sample code (pps\_test)，实时分析统计接收到的时间戳 (timestamp)，得到PPS信号的频率、周期、平均误差、最大误差、方差等值，并实时打印。

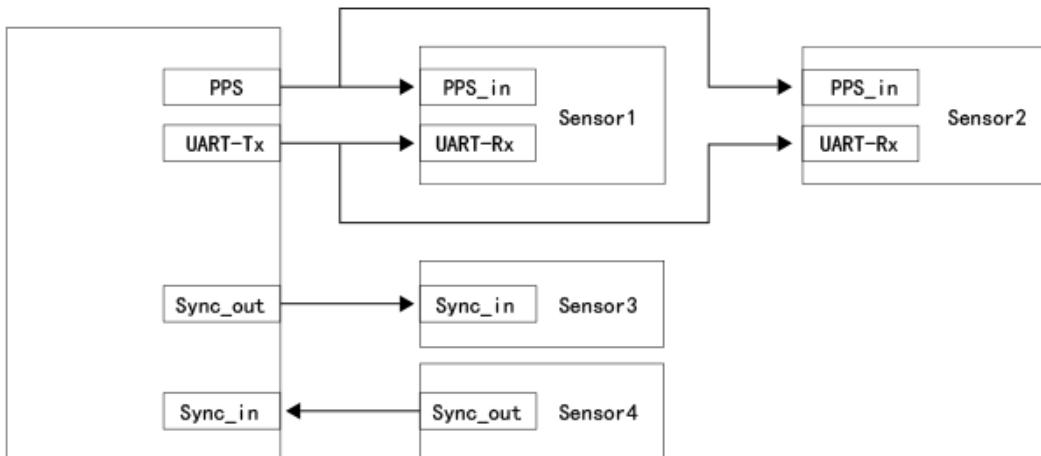
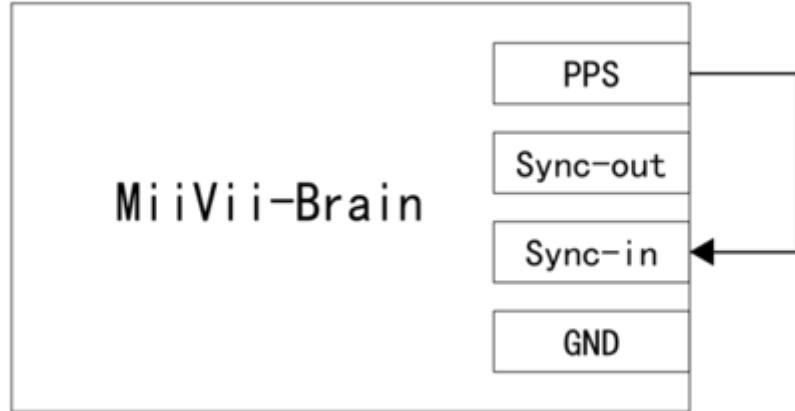


图 IO接线示意图

```
You are checking the sync in mode,please check the connecting line
Time interval: the interval between two frames
Frequency measured : the Frequency measured by the Time interval
Max deviation : the difference between the maximum and the average
standard deviation : calculated by statistical data
Time interval: 39999999(ns) Frequency measured: 25.000001(hz) Max deviation: 193(ns) standard deviation: 76.583288
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 193(ns) standard deviation: 65.984847
Time interval: 39999999(ns) Frequency measured: 25.000001(hz) Max deviation: 193(ns) standard deviation: 71.700767
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 193(ns) standard deviation: 71.693793
Time interval: 39999999(ns) Frequency measured: 25.000001(hz) Max deviation: 193(ns) standard deviation: 69.942834
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 193(ns) standard deviation: 61.269895
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 193(ns) standard deviation: 68.673139
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 193(ns) standard deviation: 78.115299
Time interval: 39999999(ns) Frequency measured: 25.000001(hz) Max deviation: 193(ns) standard deviation: 66.143783
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 193(ns) standard deviation: 77.051931
Time interval: 40000000(ns) Frequency measured: 25.000000(hz) Max deviation: 193(ns) standard deviation: 67.535176
```

图 PPS测试结果

## GMSL2摄像头支持

序号	品牌	产品型号	支持类型	快门类型	分辨率	帧率
1	Sensing	SG1-AR0143-0101-GMSL	正式	卷帘	1280*720	30 FPS
2	Sensing	SG1-AR0147-0101-GMSL	Beta	卷帘	1280*720	30 FPS
3	Sensing	SG2-AR0231-0202-GMSL	Beta	卷帘	1920*1080	22 FPS

序号	品牌	产品型号	支持类型	快门类型	分辨率	帧率
4	Sensing	SG2-AR0233-GW5200-GMSL2	Beta	卷帘	1920*1080	30-60 FPS
5	Sensing	SG1-AR0144C-8310-GMSL	Beta	全局	1280*720	30 FPS
6	Sensing	SG1-AR0144M-8310-GMSL	Beta	全局	1280*720	30 FPS
7	Sensing	SG8-AR0820C-5300-GMSL2	Beta	卷帘	3840*2160	30 FPS
8	Sensing	SG2-IMX390C-5200-GMSL2	Beta	卷帘	1920*1080	30 FPS
9	Sensing	SG2-OX03CC-5200-GMSL2F	Beta	卷帘	1920*1080	30fps
10	Sensing	SG5-IMX490C-5300-GMSL2	Beta	卷帘	2880*1860	30 FPS
11	Leopard	LI-AR0233-NVP2650-GMSL2	Beta	卷帘	1920*1080	30 FPS
12	Leopard	LI-IMX490-GW5400-GMSL2	Beta	卷帘	2880*1860	25 FPS
13	Leopard	LI-AR0820-GW5400-GMSL2	Beta	卷帘	3840*2160	15 FPS
14	Leopard	LI-AR0231-AP0200-GMSL2	Beta	卷帘	1920*1020	28.7 FPS
15	丽景	LC008A1 (AR0233-5200-GMSL2)	Beta	卷帘	1920*1080	30 FPS
16	丽景	LC001A1 (OV9716-OAX0496-GMSL)	Beta	卷帘	1280*720	30 FPS
17	丽景	LC022A1 (IMX390-5200-GMSL2)	Beta	卷帘	1920*1080	30 FPS
18	Entron	S001A	正式	卷帘	1280*720	30 FPS
19	英睿	Xsafe_A6D	Beta	红外	640*512	25 FPS

序号	品牌	产品型号	支持类型	快门类型	分辨率	帧率
20	艾睿	IR-Pilot640-XXG1	Beta	红外	640*512	50 FPS/25 FPS
21	艾睿	IR-Pilot640-XXG2	Beta	红外	640*512	50fps

注：

1. 正式支持：每次米文系统版本升级，会在米文设备上进行验证。
2. BETA支持：米文调试过，但不会在每次米文系统版本升级中验证，如使用过程中需要进一步支持请联系对应的销售工程师或客户经理。

## GMSL2摄像头使用方法

### 接口特性

- **不支持热插拔。**
- 支持最长15米同轴电缆的信号传输。
- 推荐支持输出分辨率为720p, 1080p, 4k等多种分辨率的相机。

### 连线方式

连接方式请参考“接口说明”部分

### 名词解释

名词	解释	备注
自触发	指摄像头不受外部触发信号控制，自己进行快门。	一般相机都支持这种模式。只要外部不给触发信号就是工作在这种模式。
同步触发	指所有摄像头受同时触发信号控制，在同时进行快门。	<b>需要相机固件支持。</b> 购买相机的时候请和厂商确认是否支持。 同时需要外部给触发信号。（如使用我们的SDK）

### 摄像头配置

在初次接入GMSL摄像头以及更换GMSL摄像头型号时，

对于Jetpack 4.4及以前，可以参考：[通用-Jetpack 4.4版本及以下镜像MIIVII SETTINGS的使用说明](#)

对于Jetpack 4.5及以后，可以参考：[通用-Jetpack 4.5版本及以上镜像MIIVII SETTINGS的使用说明](#)

### 快速验证

您可以使用Linux下的Cheese工具，进行快速验证出图是否正常。

您可以参照如下视频：

1. 确认GMSL相机设置正确
2. 打开Cheese
3. 确认分辨率设置正常
4. 选中Cheese相机对应的设备名字。（视频中选中的是video4）

请注意，由于Cheese并不能设置图像格式，由于相机输出的格式并不一定和Cheese默认格式匹配，因此可能存在色差。这个属于Cheese软件的问题。

 [open\\_gmsl\\_with\\_cheese.mp4](#)

## 视频输出

1.为了方便使用，设备提供cameras\_egl\_demo, cameras\_opencv\_demo, cameras\_sdk\_demo三个可执行文件来显示GMSL摄像头图像。具体请参考 /opt/miivii/features/gmsl\_camera

非SDK用法（强烈推荐！）：[OpenCV Python Demo](#), [OpenCV C++ Demo](#)

cameras\_opencv\_demo: OpenCV Demo 直接使用v4l和opencv来获取摄像头图像。

cameras\_sdk\_demo: SDK Demo，兼容GMSL1的SDK，使用ASIC来进行图像格式的转换，运行效率高。并且可以通过SDK获取触发快门的时间戳。（如需时间戳，则推荐，否则不推荐）

cameras\_egl\_demo: EGL Demo，使用egl来作为显示部分实现，显示部分效率高。

**注：**EVO TX2 GMSL2产品无cameras\_opencv\_demo

2.设备支持使用gstreamer输出视频流，图像获取与显示使用方法如下：

720P:

```
gst-launch-1.0 -v v4l2src device="/dev/video1" ! video/x-raw, framerate=
```

1080P:

```
gst-launch-1.0 -v v4l2src device="/dev/video0" ! video/x-raw, framerate=
```

请注意，由于相机输出的格式并不一定和gstreamer默认格式匹配，因此可能存在色差。这个属于相机固件与gstreamer格式匹配的问题。

3.视频输出分为自触发模式，同步模式两种应用场景：

(1).**自触发模式**：一般所有相机都支持这种模式。是指相机根据相机本身最高帧率输出，此时无法获取相机的时间戳，相机之间的图片也无法同步。

在使用自触发模式的情况下，无需使用任何SDK。可以参考：[OpenCV Python Demo](#), [OpenCV C++ Demo](#)

**但要注意需要自己进行格式转换。如从YUYV转成BGR格式等。**

(2).**同步模式**：需要相机固件支持外部触发。是指所有相机都被同一触发信号触发，快门时间几乎严格同步。

如果你只是希望使用同步模式，但是不关心相机的时间戳，我们仍然建议您使用非SDK的示例。[OpenCV Python Demo](#), [OpenCV C++ Demo](#)

在同步模式下，使用OpenCV demo显示/dev/video0的摄像头画面。请**注意，1280x720需要和摄像头真实分辨率匹配。如1080p的相机，则为1920x1080。**

```
./bin/cameras_opencv_demo -s 1280x720 -d /dev/video0
```

在同步模式下，使用SDK demo显示/dev/video0的摄像头画面。请**注意，1280x720需要和摄像头真实分辨率匹配。如1080p的相机，则为1920x1080。**

```
./bin/cameras_sdk_demo -s 1280x720 -d /dev/video0
```

## 命令示例：

第一步：编译

```
cp -r /opt/miivii/features ~/  
cd ~/features/gmsl_camera  
sudo make;
```

运行OpenCV Demo **(请注意分辨率和相机真实分辨率必须一致)**

```
./bin/cameras_opencv_demo -s 1280x720 -d /dev/video0
```

运行SDK Demo **(请注意分辨率和相机真实分辨率必须一致)**

```
./bin/cameras_sdk_demo -s 1280x720 -d /dev/video0
```

运行EGL Demo **(请注意分辨率和相机真实分辨率必须一致)**

```
./bin/cameras_egl_demo -s 1280x720 -d /dev/video0
```



图 摄像头图像

在同一进程，打开多路相机，并且获取时间戳：

命令	结果示例
Apex Xavier系列 上在同时打开2路相机 <pre>./bin/cameras_sdk_demo -s 1280x720 -d /dev/video0 -m 2</pre> -m表示打开的摄像头节点数量。	会显示2路独立视频，并以独立窗口显示。

## GMSL/GMSL2时间戳相关测试方法

### 如何获取详细日志及日志说明？

## 命令

```
# export CHECK_TIME=1      调试使用log 测试时不需要执行，否则log太多会阻塞程序
sudo jetson_clocks
rm /tmp/cameras_sdk_demo.log
./bin/cameras_sdk_demo -s 1280x720 -d /dev/video0
```

## 日志文件说明

```
Timestamp : 1620883371666965856 FrameInterval : 3333344 FrameTransferDelay : 67329552 LinuxGetFrameTime : 1620883371706962800 LinuxFrameInterval : 31946000
Timestamp : 1620883371700299040 FrameInterval : 33333184 FrameTransferDelay : 66530144 LinuxGetFrameTime : 1620883371733496000 LinuxFrameInterval : 33888000
Timestamp : 1620883371733632448 FrameInterval : 33333408 FrameTransferDelay : 67822960 LinuxGetFrameTime : 16208833717368122000 LinuxFrameInterval : 32534000
Timestamp : 1620883371766965760 FrameInterval : 33333312 FrameTransferDelay : 68000240 LinuxGetFrameTime : 1620883371801212000 LinuxFrameInterval : 34626000
Timestamp : 1620883371800629104 FrameInterval : 33333344 FrameTransferDelay : 68488896 LinuxGetFrameTime : 1620883371834960000 LinuxFrameInterval : 33090000
Timestamp : 1620883371833632320 FrameInterval : 33333216 FrameTransferDelay : 68185680 LinuxGetFrameTime : 1620883371860788000 LinuxFrameInterval : 33754000
Timestamp : 1620883371866965728 FrameInterval : 33333376 FrameTransferDelay : 68180272 LinuxGetFrameTime : 1620883371901818000 LinuxFrameInterval : 33822000
Timestamp : 1620883371900299104 FrameInterval : 33333280 FrameTransferDelay : 68258896 LinuxGetFrameTime : 1620883371935146000 LinuxFrameInterval : 33328000
Timestamp : 1620883371906696576 FrameInterval : 33333376 FrameTransferDelay : 69209616 LinuxGetFrameTime : 1620883372062842000 LinuxFrameInterval : 32412000
Timestamp : 16208833720006298752 FrameInterval : 33332992 FrameTransferDelay : 67530240 LinuxGetFrameTime : 1620883372034960000 LinuxFrameInterval : 35284000
Timestamp : 1620883372053400000 FrameInterval : 33333568 FrameTransferDelay : 67568248 LinuxGetFrameTime : 1620883372067867000 LinuxFrameInterval : 32054000
Timestamp : 1620883372066965728 FrameInterval : 33333408 FrameTransferDelay : 66865680 LinuxGetFrameTime : 1620883372100498000 LinuxFrameInterval : 32971000
Timestamp : 1620883372066965728 FrameInterval : 33333248 FrameTransferDelay : 67548272 LinuxGetFrameTime : 16208833721345146000 LinuxFrameInterval : 32631000
Timestamp : 1620883372066965728 FrameInterval : 33333344 FrameTransferDelay : 67060624 LinuxGetFrameTime : 1620883372167050000 LinuxFrameInterval : 34016000
Timestamp : 16208833721400966016 FrameInterval : 33333696 FrameTransferDelay : 67456680 LinuxGetFrameTime : 1620883372201089000 LinuxFrameInterval : 32491000
Timestamp : 16208833722006298976 FrameInterval : 33332960 FrameTransferDelay : 66747984 LinuxGetFrameTime : 1620883372233714000 LinuxFrameInterval : 34084000
Timestamp : 1620883372233631994 FrameInterval : 33332928 FrameTransferDelay : 68850024 LinuxGetFrameTime : 1620883372268349000 LinuxFrameInterval : 32625000
Timestamp : 162088337226665440 FrameInterval : 33333530 FrameTransferDelay : 66671096 LinuxGetFrameTime : 1620883372300303000 LinuxFrameInterval : 34635000
Timestamp : 16208833723006298800 FrameInterval : 33333440 FrameTransferDelay : 67781560 LinuxGetFrameTime : 1620883372334747000 LinuxFrameInterval : 31954000
Timestamp : 1620883372333923200 FrameInterval : 33333280 FrameTransferDelay : 68326840 LinuxGetFrameTime : 1620883372367273000 LinuxFrameInterval : 32525000
Timestamp : 1620883372366695524 FrameInterval : 33333184 FrameTransferDelay : 67087656 LinuxGetFrameTime : 1620883372401959000 LinuxFrameInterval : 34666000
Timestamp : 1620883372400298440 FrameInterval : 33333216 FrameTransferDelay : 66794400 LinuxGetFrameTime : 1620883372434053000 LinuxFrameInterval : 32994000
Timestamp : 1620883372433631904 FrameInterval : 33333444 FrameTransferDelay : 66534096 LinuxGetFrameTime : 1620883372467095000 LinuxFrameInterval : 33042000
Timestamp : 1620883372500653400 FrameInterval : 33333400 FrameTransferDelay : 67299656 LinuxGetFrameTime : 1620883372500166000 LinuxFrameInterval : 33071000
^CWill exit...
Timestamp : 1620883372500653400 FrameInterval : 33333344 FrameTransferDelay : 66678312 LinuxGetFrameTime : 1620883372566977000 LinuxFrameInterval : 32712000
Timestamp : 1620883372533631904 FrameInterval : 33333216 FrameTransferDelay : 67258096 LinuxGetFrameTime : 1620883372600890000 LinuxFrameInterval : 33913000
Timestamp : 1620883372500653400 FrameInterval : 33333376 FrameTransferDelay : 66647720 LinuxGetFrameTime : 162088337263613000 LinuxFrameInterval : 32723000
```

时间戳

两帧差值

帧延时

系统时间戳

系统两帧差值

字段	单位	物理含义	测试方法
Timestamp	纳秒	触发该帧的时间	根据传输延迟，从队列中获得的触发时间
FrameInterval	纳秒	帧间隔 两帧之间的触发时间 间隔	与前一Timestamp的差值
FrameTransferDelay	纳秒	帧传输延迟	LinuxGetFrameTime - Timestamp
LinuxGetFrameTime	纳秒	Linux获取帧时的系统 时间	收到帧时候的Linux系统时间
LinuxFrameInterval	纳秒	Linux获取帧的Linux 时间间隔	与前一LinuxGetFrameTime的差 值

## 如何确认时间戳是否准确？

在代码中，会通过检查FrameInterval的方式，来确认时间戳是否准确。

执行命令：

```
# export CHECK_TIME=1      调试使用log 测试时不需要执行，否则log太多会阻塞程序
sudo jetson_clocks
```

```
rm /tmp/cameras_sdk_demo.log  
.bin/cameras_sdk_demo -s 1280x720 -d /dev/video0
```

如果时间戳正常，则在/tmp/cameras\_sdk\_demo.log的内容如下，**只有一行**：

```
Timestamp : 1620897955083817280 FrameInterval : 1620897955083817280 F:
```

如果时间戳异常，则在/tmp/cameras\_sdk\_demo.log中会记录多组时间戳，**为多行**：

```
Timestamp : 1620958367246484576 FrameInterval : 1620958367246484576  
Timestamp : 1620958739646034432 FrameInterval : 1620958739646034432  
Timestamp : 1620958748796023808 FrameInterval : 1620958748796023808  
Timestamp : 1620958789795973504 FrameInterval : 1620958789795973504  
Timestamp : 1620959793244763712 FrameInterval : 1620959793244763712  
Timestamp : 1620959854794691840 FrameInterval : 1620959854794691840  
Timestamp : 1620960274844196896 FrameInterval : 1620960274844196896  
Timestamp : 1620960283994186240 FrameInterval : 1620960283994186240  
Timestamp : 1620960291394178080 FrameInterval : 1620960291394178080
```

## 如何确认时间戳精度？

### 命令

把屏幕日志存储到文件中

```
# export CHECK_TIME=1      调试使用log 测试时不需要执行，否则log太多会阻塞程序  
sudo jetson_clocks  
.bin/cameras_sdk_demo -s 1280x720 -d /dev/video0 > log
```

## 如何确认图像帧传输延迟是否稳定？

确认摄像头图像帧传输延迟

先验知识

GMSL类型	图像分辨率	典型相机	传输延迟
GMSL1	720p	SG1-AR0143-0101-GMSL-Hxxx	60ms左右
GMSL1	1080p	SG2-AR0231-0202-GMSL-Hxxx	100ms左右

使用低于传输延迟的帧率，打开摄像头。由于此时传输延迟小于帧间隔，因此时间戳的缓存为1，因此不会因为软件引入其他问题，所测的就是真实的物理延迟。

**请注意不要使用传输延迟附近所对应的帧率，传输延迟的抖动，会造成时间戳不准。**

命令	确认方式
<p><b>Apex Xavier II 系列</b></p> <pre># export CHECK_TIME=1    仅在调试使用，否则log太多会阻塞程序 sudo jetson_clocks ./bin/cameras_sdk_demo -s 1280x720 -d /dev/video0 -r 10-0</pre>	通过 FrameTracer 确认得到实 际延迟

-r 10-0表示设置的摄像头外触发帧率为10。

## Apex Xavier和EVO TX2 GMSL2

```
# export CHECK_TIME=1    仅在调试使用，否则log太多会阻塞程序
sudo jetson_clocks
./bin/cameras_sdk_demo -s 1280x720 -d /dev/video0 -r 10
```

-r 10表示设置的摄像头外触发帧率为10。

## 应用功能使用

米文设备提供多种样例，方便客户进行开发和快速验证

算法：米文设备提供算法库，目前提供行人，车辆，自行车三分类的检测。详情请参考/opt/miivii/features/algorithm 中的三分类检测算法

加速SDK：米文设备提供基于Yolo v3识别网络的加速SDK。详情请参考/opt/miivii/features/miivii-accelerator

ROS范例：米文设备提供基于ROS的DEMO。详情请参考米文动力Github <https://github.com/MiiViiDynamics>

除此之外，米文动力还为开发者提供了部分开源代码，请于米文动力Github查看 <https://github.com/MiiViiDynamics>

## 附录

### 异常处理

如在开发过程中出现异常情况，可先通过DEBUG串口打印log自行判断问题。具体操作如下：

第一步：根据【接口说明】部分中的信息，找到DEBUG接口的具体位置

第二步：用一根UART-USB转接线<sup>1</sup>，将DEBUG接口与上位机PC相连接

第三步：在上位机PC端，下载串口调试工具，将波特率调整为115200 Baud

第四步：在串口调试工具中抓取串口log以便分析异常问题

[1]：可根据【接口说明】部分中的信息，选择RS232-USB转接线或者TTL-USB转接线。

## 系统在线升级（OTA）的使用说明

### 概述

系统在线升级，通常又是OTA，是米文针对所有设备提供的软件服务。即可以不进行刷机来更新系统固件。从Jetpack 4.5开始，所有的米文设备都支持系统在线升级。

## 使用方式

### 方法一（推荐）：使用MIIVII SETTINGS进行版本升级：

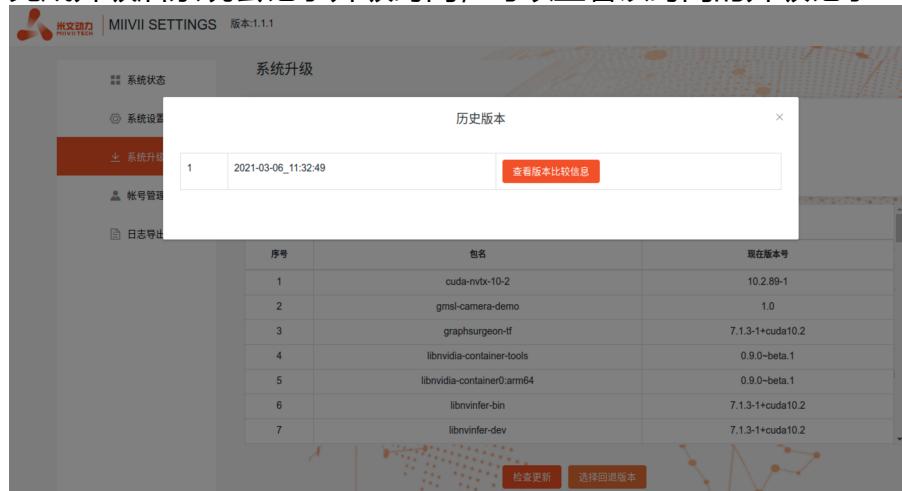
- 在设备上打开浏览器输入<http://127.0.0.1:3000>，或者远程PC浏览器上输入<http://<device ip>:3000>
- 使用系统登录账号登录到MIIVII SETTINGS界面；
- 选择系统升级功能，点击“检查更新” 检查是否有新版本；



- 检测到有升级版本时，可以点击“系统升级”来升级安装包



- 完成升级后系统会记录升级时间，可以查看该时间的升级记录



6. 升级完成后重启系统以确保升级内容生效

## 方法二：使用命令行进行升级或者升级指定安装包

### 使用命令行进行升级

1. 执行下面命令更新源

```
sudo apt update
```

2. 执行下面命令升级系统

```
sudo apt upgrade -y
```

3. 升级完成后重启系统以确保升级内容生效

### 升级指定安装包

- 执行下面命令升级指定安装包（以更新websettings 1.4.0版本为例）

```
sudo apt install -y miivii-websettings=1.4.0
```

## Jetpack 4.4版本及以下镜像烧录

请参考：[Jetpack 4.4版本及以下镜像烧录](#)

## Jetpack 4.5版本及以上镜像烧录

### 1. 功能介绍

米文刷机工具，适用于米文系列产品。

米文刷机工具，是为了方便进行米文设备的烧写、克隆，小批量生产而提供的工具软件。

您可以通过X86架构PC作为烧写主机，给米文设备烧写米文动力官方镜像。在开发米文设备一段时间后，可以将现有设备镜像克隆来保存开发进度，并单台或小批量烧写到其他米文设备中。

### 核心功能

- 自动检测使用环境
- 自动检测最新镜像
- 内置镜像下载器，无需手动下载镜像
- 支持批量烧写
- 支持镜像克隆（但要注意Clone后再烧写需要使用同一Jetpack版本）

### 2. 准备软件硬件

#### 2.1. 烧写主机准备

需要将烧写主机与米文设备连接方能烧写镜像。烧写主机推荐配置如下：

- CPU采用X86架构的Intel酷睿系列处理器
- 内存8GB ddr3及以上
- 空余硬盘容量40G 及以上
- 系统为Ubuntu Linux x64 v16.04或v18.04

#### 2.2. 烧写软件环境准备

- `sudo apt install python2.7 python3 python`

#### 2.3. 准备米文烧写工具和米文设备镜像

##### 2.3.1. 刷机工具安装

- 准备PC主机，系统为：Ubuntu 16.04或者Ubuntu 18.04

- 安装key

```
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 05B
```

- 在本地 ubuntu 系统中添加源

```
sudo sh -c 'echo "deb http://upgrade.miivii.com/miivii/tools/" > /etc/apt/sources.list.d/miivii-tools.list'
```

- 手动更新

```
sudo apt update
```

- apt-get 安装 刷机工具 Deb 包

```
sudo apt-get install miivii-ftool
```

- 安装完成后(在18.04系统中点击Show Applications或者在16.04系统中点击Search Your Computer)会发现如下快捷方式



- 双击快捷方式，输入密码：您的sudo密码。

## 2.4. 准备硬件

- 米文设备及电源, USB 数据线

## 3. 操作

### 3.1. 硬件连接

- 通过 USB 数据线将米文设备烧写口与烧写主机相连；
- 按住米文设备的RECOVERY按钮，之后给米文设备上电开机，进入FORCE\_Recovery烧写模式。

### 3.2 软件使用

#### 3.2.1. 镜像烧写

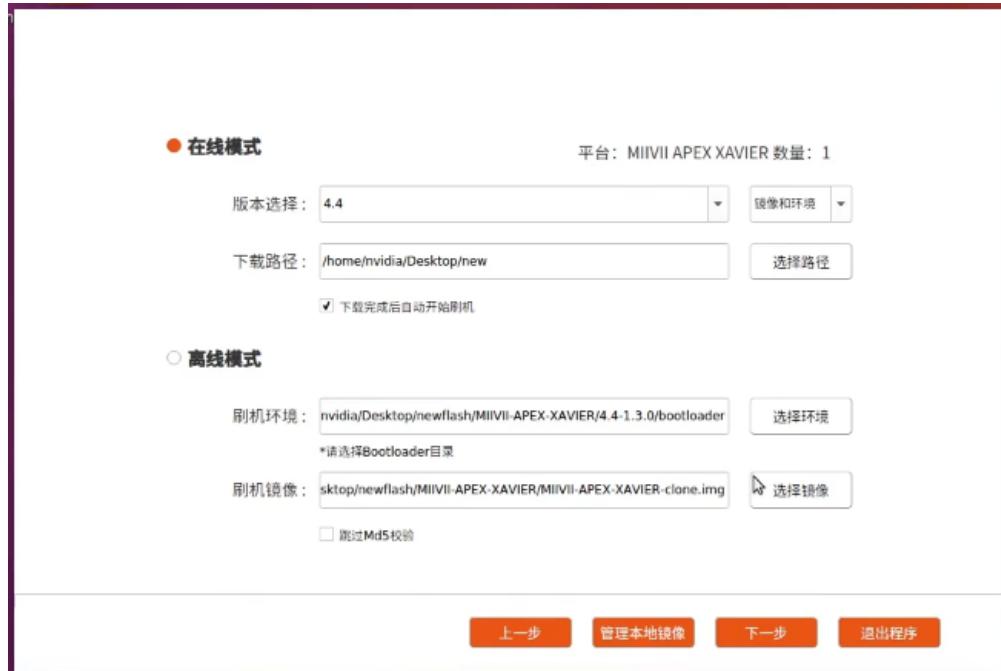
##### 3.2.1.1 在线模式镜像烧写

- 点击“在线模式”复选框，选择Jetpack版本及下载路径，并点击下一步，开始下载选择版本当前最新的刷机环境及设备镜像
- 这里需要选择下载完成后是否自动开始刷机，选择自动后，下载完成后会自动解压、校验、刷机
- 下载速度取决于所在环境的网速，一般可达5M/s
- 开始刷机后通常需要15分钟以上完成，请耐心等待

##### [刷机工具使用教程.mp4](#)

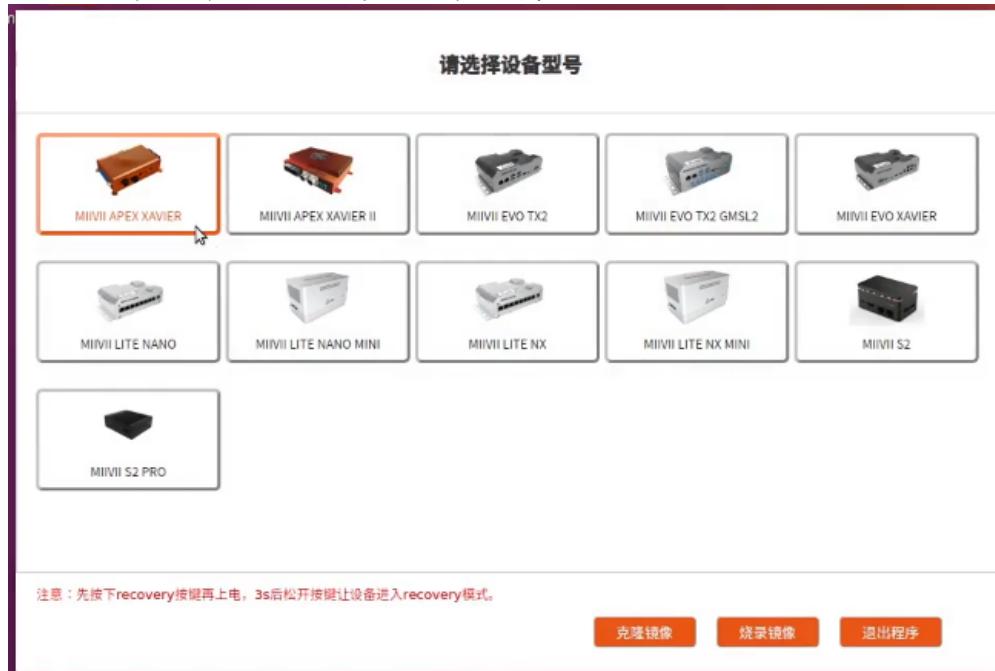
##### 3.2.1.2 离线模式镜像烧写

- 点击“离线模式”复选框，选择已经下载好的刷机环境及设备镜像，并点击下一步直接开始烧录。

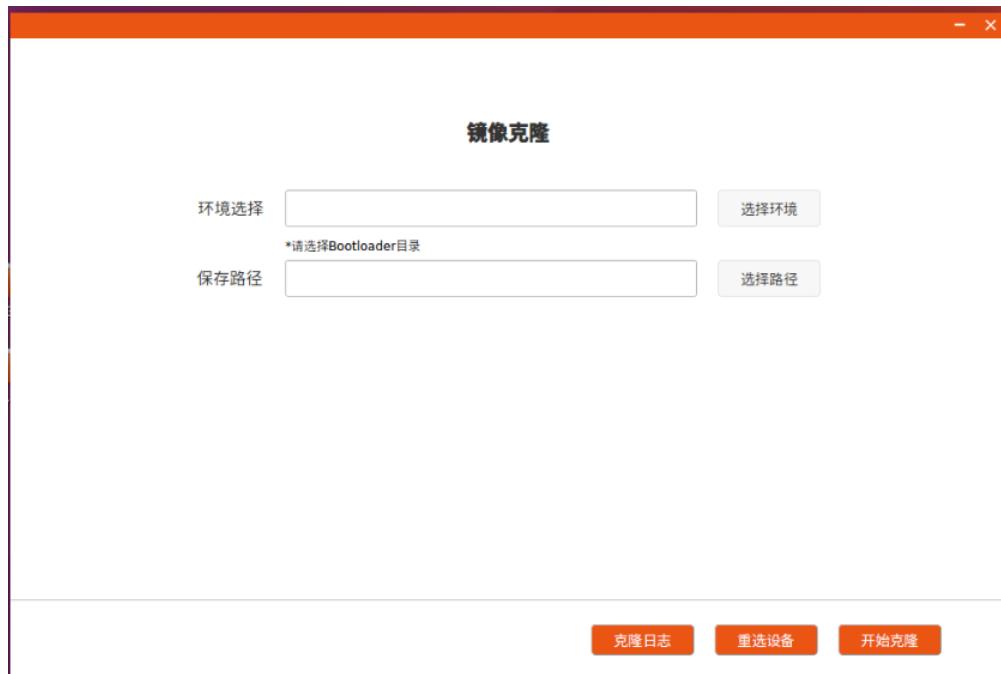


### 3.2.2. 镜像克隆

- 将打算克隆的米文设备按照3.1的方法进入FORCE\_Recovery模式，打开烧写工具
- 点击【输入上位机密码】按钮，**输入当前烧写主机的开机密码**
- 点击【克隆镜像】按钮，进入克隆操作



- 修改克隆文件保存在烧写主机中的路径和名称\*，并点击开始克隆
- 注：文件存储路径中不能有中文或特殊字符



- 镜像克隆通常需要30分钟以上才能完成:
- 克隆完成，会生成克隆镜像与MD5文件，再次烧写请按照3.2.1步骤进行操作  
注：如在镜像烧写，克隆过程中遇到问题，请联系米文动力售后邮箱寻求帮助：  
[helpdesk@mivii.com](mailto:helpdesk@mivii.com)

#### 附1. 烧写问题自检

如果遇到烧写问题，请先按照如下条目进行自检：

- 检查是否在烧写工具左上角输入了上位机开机密码
- 检查是否进入到Recovery模式，可以通过lsusb命令鉴定
- 检查Micro USB、双Type A线缆质量是否达标，是否只是用于充电的双芯线
- 检查上位机，是否为X86-64架构台式机，笔记本。（服务器，嵌入式设备，虚拟机等其他设备暂不支持）
- 检查上位机系统是否为 Linux 1604 1804
- 检查磁盘格式，烧写主机的磁盘格式推荐为EXT4
- 检查上位机容量是否足够
- 镜像和烧写工具存储路径中不能有中文或其他特殊字符

## Apex Xavier II+ 产品软件-Release Note

产品	更新时间	系统版本	更新内容	备注
MIVII Apex Xavier II+				