

LDROBOT

Move Smarter

激光测距传感器 STP-23L

开发手册 V0.1



深圳乐动机器人股份有限公司

SHENZHEN LDROBOT CO.,LTD

目录

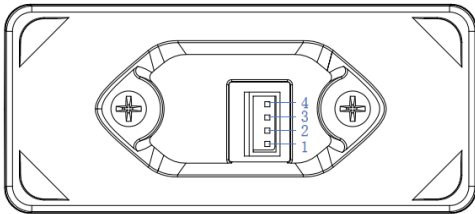
1. 工作机制	1
1.1. 通讯接口	1
1.2. 工作流程	1
2. 通讯协议	2
2.1. 数据包格式	2
2.2. 获取测量数据命令	2
2.3. 停止命令	4
2.4. 应答命令	4
2.5. 复位命令	5
2.6. 获取传感器信息命令	5
3. 坐标系	7
4. 开发套件使用	8
4.1. 硬件线材连接及说明	8
4.2. 上位机软件的使用	8
4.2.1. Windows 下驱动程序安装	8
4.2.2. Windows 可视化软件使用	10
4.3. Linux 下 SDK 使用说明	11
4.3.1. 获取 SDK 源码	11
4.3.2. SDK 应用说明	11
4.3.3. SDK 运行调试说明	11

5. 修订记录	14
---------------	----

1. 工作机制

1.1. 通讯接口

STP-23L 使用 SMT 4PIN 1mm 连接器与外部系统连接，实现供电和数据接收，具体接口定义和参数要求见下图/表：



序号	信号名	类型	描述	最小值	典型值	最大值
1	Tx	输出	UART TX	0V	3.3V	3.5V
2	RX	输入	UART RX	0V	-	3.3V
3	GND	供电	电源负极	-	0V	-
4	P5V	供电	电源正极	4.5V	5V	5.5V

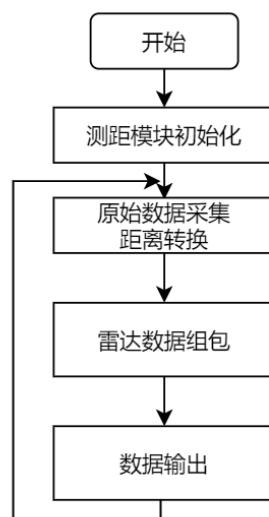
STP-23L 的数据通讯采用标准异步串口(UART)单向发送，其传输参数如下表所示：

波特率	数据长度	停止位	奇偶校验位	流控制
230400	8bits	1	无	无

1.2. 工作流程

STP-23L 测距核心采用 DTOF 技术，可进行每秒 120 次的测距。上电初始化后，测距模块工作开始采集数据，当采集到测距点后传感器会把数据打包通过串口发送出去。

雷达工作流程如下图所示：



2. 通讯协议

2.1. 数据包格式

STP-23L 串口收发的数据包格式，遵从小端模式，如下表所示：

起始符				设备地址	命令码	整包偏移地址		数据长度		数据域	校验码
AAH	AAH	AAH	AAH	设备地址	命令码	LSB	MSB	LSB	MSB	DATA	CS
4B				1B	1B	1B	1B	1B	1B	...	1B

- **起始符起**：0xAAAAAAAA，标识数据包的开始，占 4 个字节；
- **设备地址**：保留位
- **命令码**：常用命令码参考如下。

宏	值	说明
PACK_GET_DISTANCE	0x02	获取测量数据
PACK_RESET_SYSTEM	0x0D	复位
PACK_STOP	0x0F	停止测量数据传输
PACK_ACK	0x10	应答码
PACK_VERSION	0x14	获取传感器信息

- **整包偏移地址**：数据过长时会拆分成多个块，分块发送，这里表示块在整个包的偏移地址；
- **数据长度域**：表示数据段的长度，范围 0-320；
- **数据域**：传输数据信息；
- **校验码**：除去数据协议头后的数据的校验和；

2.2. 获取测量数据命令

设备上电默认开始发送数据帧，如果有发送 PACK_STOP 停止命令，需要重新接收数据时发送该命令。

设备地址：device_address = 0

命令码：PACK_GET_DISTANCE = 0x02

偏移地址：chunk_offset = 0x00

数据长度: data_len = 0x0000

起始符				设备地址	命令码	偏移地址		数据长度		校验码
AAH	AAH	AAH	AAH	00H	02H	00	00	00	00	02H

参考接收:

AA AA AA AA 00 02 00 00 B8 00 + 数据 (长度 0xB8) + 校验和

AAH	AAH	AAH	AAH	00H	02H	00	00	B8H	00H	DATA	CS
-----	-----	-----	-----	-----	-----	----	----	-----	-----	------	----

DATA				
测量点 1	测量点 2	...	测量点 12	时间戳
POINT1	POINT2	...	POINT12	timestamp
-	-	-	-	4B

默认一帧数据由 12 个测量点的数据组成, 每个测量点的数据结构体具体由下部分组成:

距离数据	环境噪声	接收强度信息	置信度	积分次数	温度表征值
2B	2B	4B	1B	4B	2B

- **距离数据**: 测量目标距离单位 mm;
- **环境噪声**: 当前测量环境下的外部环境噪声, 越大说明噪声越大
- **接收强度信息**: 测量目标反射回的光强度
- **置信度**: 由环境噪声和接收强度信息融合后的测量点的可信度
- **积分次数**: 当前传感器测量的积分次数
- **温度表征值**: 测量芯片内部温度变化表征值, 只是一个温度变化量无法与真实温度对应

代码参考结构体如下, 按字节对齐

```
typedef struct {
    int16_t distance;
    uint16_t noise;
    uint32_t peak;
    uint8_t confidence;
    uint32_t intg;
    int16_t reftof;
}LidarPointTypedef;
```

2.3. 停止命令

获取测量数据后，设备会一直持续工作，并发送测量数据，断电前不会停止。如果用户想中途停止接收测量数据，需要发送停止测量发送命令。

设备地址：device_address（同上）

命令码：PACK_STOP = 0x0F

偏移地址：chunk_offset = 0x00

数据长度：data_len = 0x0000

参考测试指令发送：

起始符				设备地址	命令码	偏移地址		数据长度		校验码
AAH	AAH	AAH	AAH	00H	0FH	00	00	00	00	0FH

参考接收，内容格式同“2.4 节应答命令”：

AAH	AAH	AAH	AAH	00H	10H	00	00	02H	00	0FH	01H	22H
-----	-----	-----	-----	-----	-----	----	----	-----	----	-----	-----	-----

发送完停止命令后，设备处于待机状态。

2.4. 应答命令

应答命令不需要发送，是在某些命令发送后，当设备接收到该命令时以应答命令数据格式返回应答。比如发送 0x0F 停止命令后

参考接收：

起始符				设备地址	命令码	整包偏移地址		数据长度		数据域		校验码
AAH	AAH	AAH	AAH	00H	10H	00	00	02H	00	0FH	01H	22H
4B				1B	1B	2B		2B		2B		1B

数据域对应结构体如下所示：

```
struct AckResultData{
    uint8_t ack_cmd_id; //答复的命令 id
    uint8_t result;      //1 表示成功,0 表示失败
};
```

2.5. 复位命令

设备地址：device_address（同上）

命令码：PACK_RESET_SYSTEM = 0x0D

偏移地址：chunk_offset = 0x00

数据长度：data_len = 0x0000

参考测试指令发送：

起始符				设备地址	命令码	偏移地址		数据长度		校验码
AAH	AAH	AAH	AAH	00H	0DH	00	00	00	00	0DH

参考接收，内容格式同“2.4 节应答命令”：

AAH	AAH	AAH	AAH	00H	10H	00	00	02H	00	0DH	01H	20H
-----	-----	-----	-----	-----	-----	----	----	-----	----	-----	-----	-----

发送完复位命令后，设备会在 100ms 之后复位重启。

2.6. 获取传感器信息命令

主要获取软件版本，传感器参数相关信息。

设备地址：device_address（同上）

命令码：PACK_VERSION = 0x14

偏移地址：chunk_offset = 0x00

数据长度：data_len = 0x0000

参考测试指令发送：

起始符				设备地址	命令码	偏移地址		数据长度		校验码
AAH	AAH	AAH	AAH	00H	14H	00	00	00	00	14H

参考接收：

AA AA AA AA 00 14 00 00 32 00 + 数据（长度 0x32） + 校验和

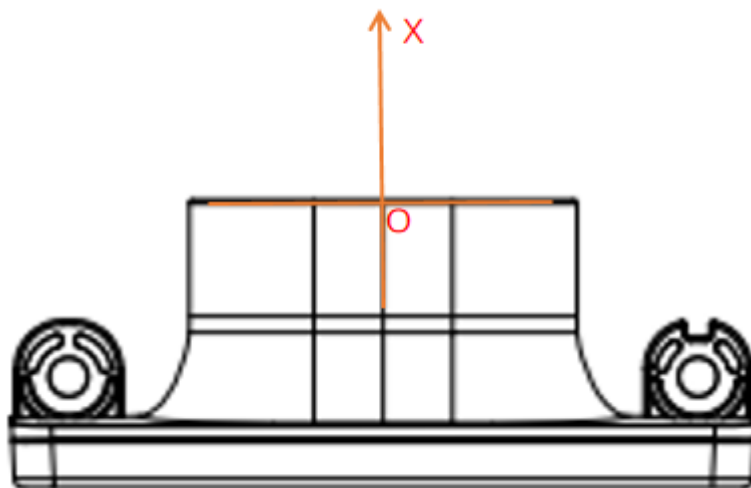
起始符				设备地址	命令码	整包偏移地址		数据长度域		数据域	校验码
AAH	AAH	AAH	AAH	00H	14H	00	00	32H	00
4B				1B	1B	2B		2B		...	1B

其中数据域参考结构体如下：

```
struct LiManuConfig
{
    uint32_t version;           //软件版本号
    uint32_t hardware_version;  //硬件版本号
    uint32_t manufacture_date;  //生产日期
    uint32_t manufacture_time;  //生产时间
    uint32_t id1;               //设备 id1
    uint32_t id2;               //设备 id2
    uint32_t id3;               //设备 id3
    uint8_t sn[8];              // sn
    uint16_t pitch_angle[4];     //角度信息
    uint16_t blind_area[2];      //盲区信息
    uint32_t frequence;          //数据点频
};
```

3. 坐标系

测距方向沿 0X 射线方向，如下图所示。



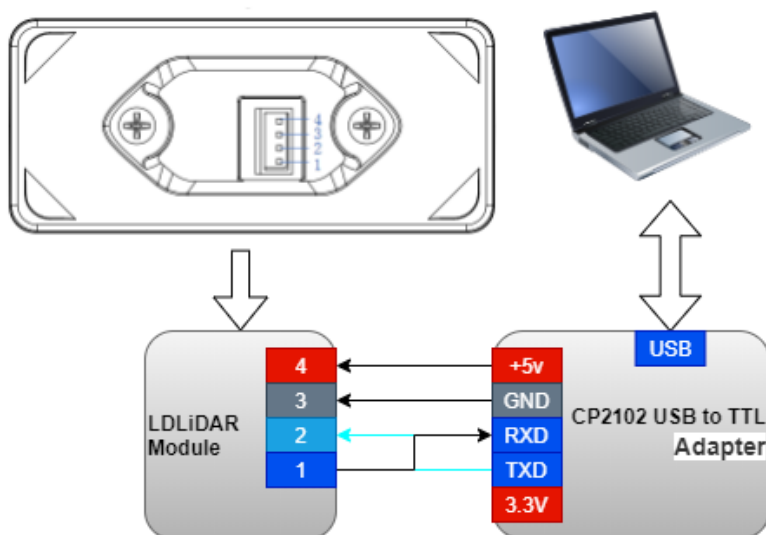
4. 开发套件使用

4.1. 硬件线材连接及说明

1) 产品、线材、USB 转接板，如下图所示：



2) 连接示意，如下图所示：



4.2. 上位机软件的使用

4.2.1. Windows 下驱动程序安装

在 windows 下对本公司产品进行评估时，需要安装 USB 转接板的串口驱动程序，原因是

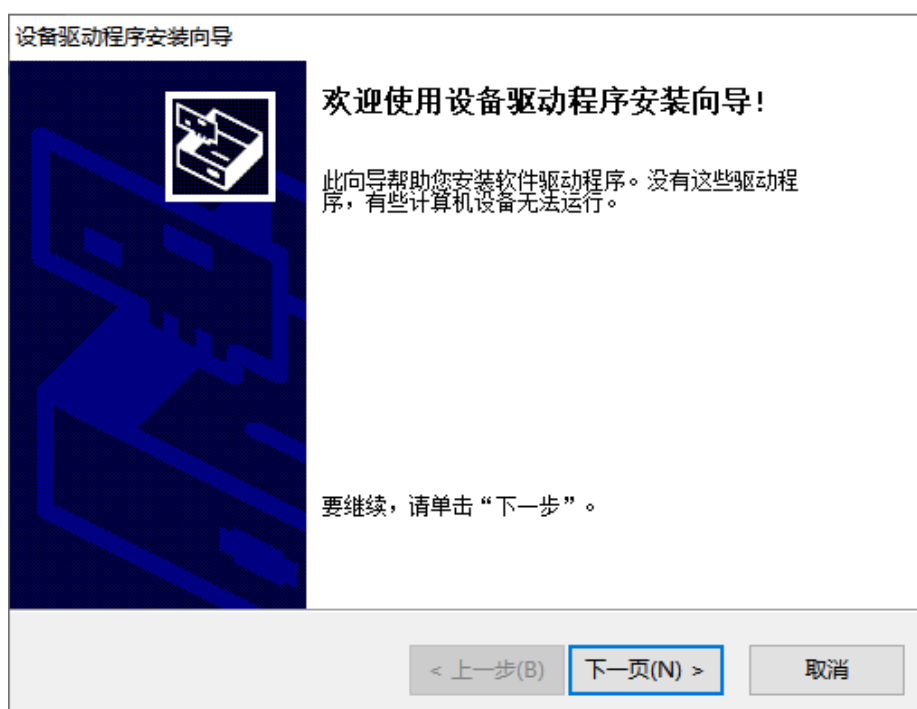
本公司提供的开发套件中 USB 转接板采用了 CP2102 USB 转串口信号芯片，其驱动程序可以从 Silicon Labs 的官方网站上进行下载：

<https://www.silabs.com/developers/usb-to-uart-bridge-vcp-drivers>

解压 CP210x_Universal_Windows_Driver 驱动程序包后，执行驱动程序安装包目录下的 exe 文件，根据 Windows 系统版本，选择 X86(32 位)或者 X64(64 位)。

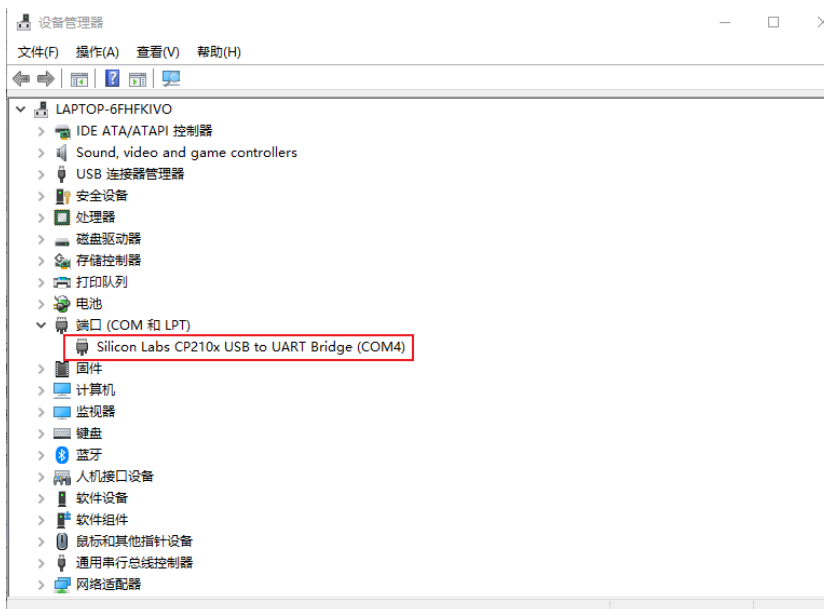
名称	修改日期	类型	大小
arm	2018/12/8 0:06	文件夹	
arm64	2018/12/8 0:06	文件夹	
x64	2018/12/8 0:06	文件夹	
x86	2018/12/8 0:06	文件夹	
CP210x_Universal_Windows_Driver_ReleaseNotes...	2018/12/7 23:53	TXT 文件	20 KB
CP210xVCPInstaller_x64.exe	2018/5/8 6:05	应用程序	1,026 KB
CP210xVCPInstaller_x86.exe	2018/5/8 6:05	应用程序	903 KB
dpinst.xml	2018/5/8 5:46	XML 文件	12 KB
silabser.cat	2018/12/4 2:17	安全目录	13 KB
silabser.inf	2018/12/4 2:17	安装信息	10 KB
SLAB_License_Agreement_VCP_Windows.txt	2016/4/27 22:26	TXT 文件	9 KB

双击 exe 文件，按提示进行安装操作。



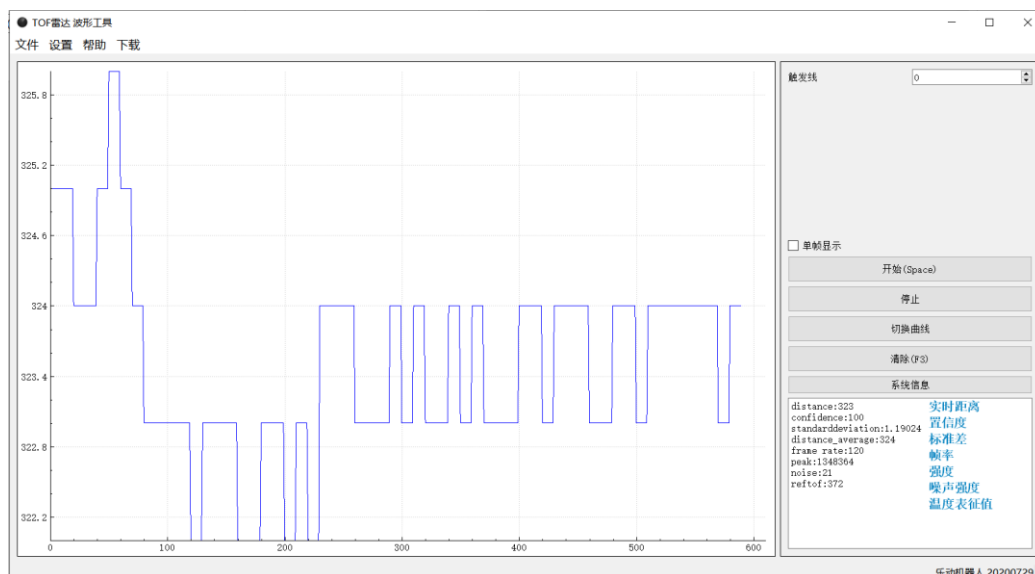
安装完成后，将开发套件中 USB 转接板与电脑相连，可以右键点击【我的电脑】，选择【属性】，在打开的【系统】界面下，选择左边菜单中的【设备管理器】进入到设备管理器，

展开【端口】，可看到识别到的 CP2102 USB 适配器所对应的串口号，即驱动程序安装成功，下图为 COM4。



4.2.2. Windows 可视化软件使用

本公司提供了本产品实时扫描的点云可视化软件, 开发者可使用该软件直观地观察本产品的扫描效果图。在使用该软件前要确保本产品的 USB 转接板的驱动程序已安装成功, 并将本产品与 Windows 系统 PC 的 USB 口互连, 然后双击.exe 文件, 然后选择对应的产品型号与端口号即可。软件请联系营销或者 FAE 人员获取。



软件左上角文件选项可以保存数据，载入数据。设置选项设置波特率，选择串口端口号。

下载选项可升级传感器固件。

4.3. Linux 下 SDK 使用说明

4.3.1. 获取 SDK 源码

本产品 Linux SDK 源码，需要联系本公司的 FAE 等营销人员来获取，源码中的 README 文档有相关使用说明。若想通过 Github 或 Gitee 等托管平台利用私库进行相关开发对接，同样请向本公司的 FAE 等营销人员反馈，我们会努力满足你的开发需求。

4.3.2. SDK 应用说明

为了方便用户快速接入传感器使用，本产品在 SDK 集成了数据解析、处理、打包等基础功能，并对多种常用开发平台兼容 (Linux、ROS、ROS2)；目前在 SDK 中集成的功能如下：

- 1、数据串口接收缓存处理；
- 2、数据包协议解析处理；
- 3、数据打包并向上层提供对应接口。

4.3.3. SDK 运行调试说明

1) SDK 运行平台说明

本示例基于 Ubuntu 操作系统下 ROS 平台，仿照示例操作前请确保安装有正确的 Ubuntu 与 ROS 环境，若需在其它平台下使用 SDK 对本产品进行评估；可参考 SDK 源码中对应平台 README 文档。

2) 设置传感器设备权限

首先，将雷达通过串口转 USB 模块接入电脑。然后在 Ubuntu 系统下打开终端，输入 `ls /dev/ttyUSB*` 查看串口设备是否接入。若检测到串口设备，则使用 `sudo chmod 777`

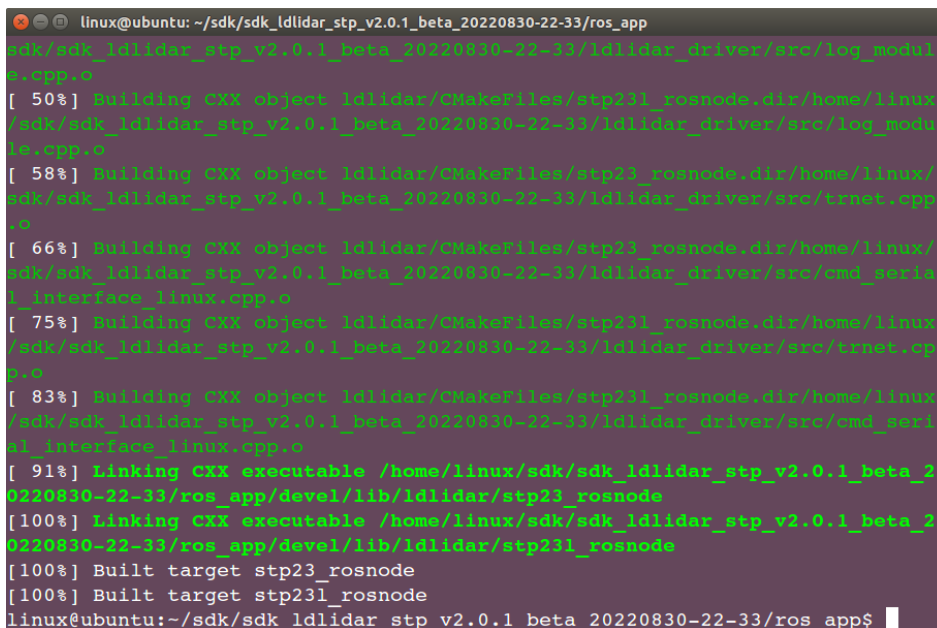
/dev/ttyUSB* 命令赋予其最高权限，即文件所有者、群组、其他用户都具有读写和执行权限。

```
ls /dev/ttyUSB*  
sudo chmod 777 /dev/ttyUSB0
```

3) 编译说明

```
$ cd <sdk 软件包根目录>/<对应平台目录>  
$ catkin_make
```

编译成功显示如下



```
linux@ubuntu: ~/sdk/sdk_ldlidar_stp_v2.0.1_beta_20220830-22-33/ros_app  
sdk/sdk_ldlidar_stp_v2.0.1_beta_20220830-22-33/ldlidar_driver/src/log_module.cpp.o  
[ 50%] Building CXX object ldlidar/CMakeFiles/stp23l_rosnode.dir/home/linux/sdk/sdk_ldlidar_stp_v2.0.1_beta_20220830-22-33/ldlidar_driver/src/trnet.cpp.o  
[ 58%] Building CXX object ldlidar/CMakeFiles/stp23_rosnode.dir/home/linux/sdk/sdk_ldlidar_stp_v2.0.1_beta_20220830-22-33/ldlidar_driver/src/cmd_serial_interface_linux.cpp.o  
[ 66%] Building CXX object ldlidar/CMakeFiles/stp23l_rosnode.dir/home/linux/sdk/sdk_ldlidar_stp_v2.0.1_beta_20220830-22-33/ldlidar_driver/src/cmd_serial_interface_linux.cpp.o  
[ 75%] Building CXX object ldlidar/CMakeFiles/stp23l_rosnode.dir/home/linux/sdk/sdk_ldlidar_stp_v2.0.1_beta_20220830-22-33/ldlidar_driver/src/trnet.cpp.o  
[ 83%] Building CXX object ldlidar/CMakeFiles/stp23l_rosnode.dir/home/linux/sdk/sdk_ldlidar_stp_v2.0.1_beta_20220830-22-33/ldlidar_driver/src/cmd_serial_interface_linux.cpp.o  
[ 91%] Linking CXX executable /home/linux/sdk/sdk_ldlidar_stp_v2.0.1_beta_20220830-22-33/ros_app/devel/lib/ldlidar/stp23_rosnode  
[100%] Linking CXX executable /home/linux/sdk/sdk_ldlidar_stp_v2.0.1_beta_20220830-22-33/ros_app/devel/lib/ldlidar/stp23l_rosnode  
[100%] Built target stp23_rosnode  
[100%] Built target stp23l_rosnode  
linux@ubuntu:~/sdk/sdk_ldlidar_stp_v2.0.1_beta_20220830-22-33/ros_app$
```

4) 运行相应节点说明

编译完成后需要将编译的文件加入环境变量，命令为 `source devel/setup.bash`，该命令是临时给终端加入环境变量，意味着您如果另外打开新的终端，也需要进入 SDK 平台路径下之执行添加环境变量命令。

添加环境变量后，`roslaunch` 命令则可以找到相应的 `ros` 包和 `launch` 文件，运行命令为 `roslaunch ldlidar stp23l.launch`。

```
$ cd <sdk 软件包根目录>/<对应平台目录>

$ source devel/setup.bash

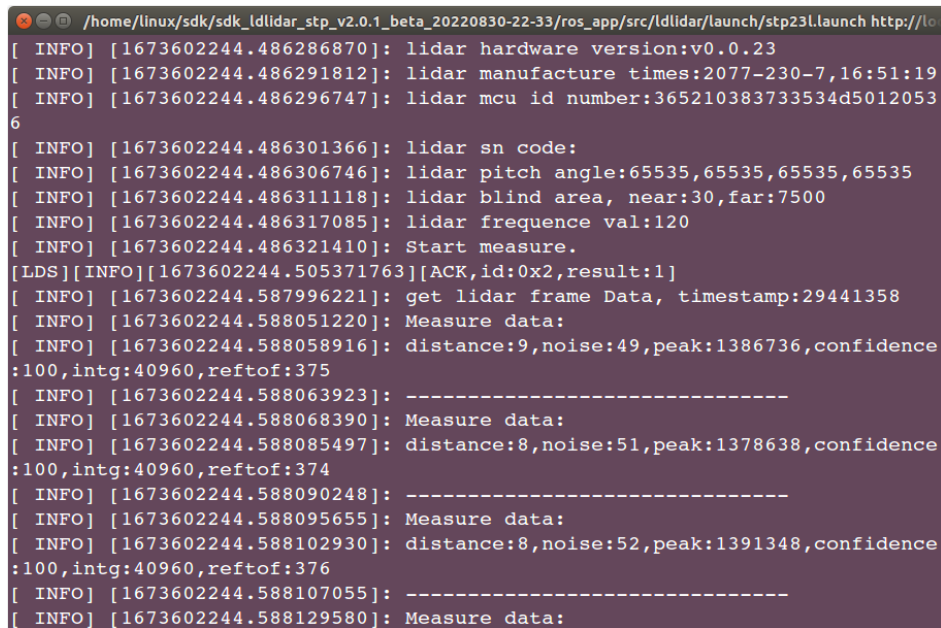
# 启动设备节点

$ roslaunch ldldidar stp23l.launch
# or

# 启动并显示数据在 Rviz

# if ROS_DISTRO in 'kinetic' or 'melodic'
roslaunch ldldidar_stl_ros viewer_stp23l_kinetic_melodic.launch
# if ROS_DISTRO in 'noetic'
roslaunch ldldidar_stl_ros viewer_stp23l_noetic.launch
```

运行界面如下图所示

A terminal window showing the output of the 'roslaunch ldldidar stp23l.launch' command. The terminal title is '/home/linux/sdk/sdk_ldldidar_stp_v2.0.1_beta_20220830-22-33/ros_app/src/ldldidar/launch/stp23l.launch http://lo...'. The output consists of several INFO messages from the 'ldldidar' node, providing hardware details like version (v0.0.23), manufacture times (2077-230-7, 16:51:19), MCU ID (365210383733534d50120536), SN code, pitch angle (65535), blind area (near:30, far:7500), and frequency (120). It also shows 'Start measure.' and 'Measure data:' messages with sensor readings such as distance, noise, peak, confidence, integration time, and reftof. The messages are timestamped with the node's local time (e.g., [1673602244.486286870]).

```
/home/linux/sdk/sdk_ldldidar_stp_v2.0.1_beta_20220830-22-33/ros_app/src/ldldidar/launch/stp23l.launch http://lo...
[ INFO] [1673602244.486286870]: lidar hardware version:v0.0.23
[ INFO] [1673602244.486291812]: lidar manufacture times:2077-230-7,16:51:19
[ INFO] [1673602244.486296747]: lidar mcu id number:365210383733534d50120536
[ INFO] [1673602244.486301366]: lidar sn code:
[ INFO] [1673602244.486306746]: lidar pitch angle:65535,65535,65535,65535
[ INFO] [1673602244.486311118]: lidar blind area, near:30,far:7500
[ INFO] [1673602244.486317085]: lidar frequency val:120
[ INFO] [1673602244.486321410]: Start measure.
[LDS][INFO][1673602244.505371763][ACK,id:0x2,result:1]
[ INFO] [1673602244.587996221]: get lidar frame Data, timestamp:29441358
[ INFO] [1673602244.588051220]: Measure data:
[ INFO] [1673602244.588058916]: distance:9,noise:49,peak:1386736,confidence:100,intg:40960,reftof:375
[ INFO] [1673602244.588063923]: -----
[ INFO] [1673602244.588068390]: Measure data:
[ INFO] [1673602244.588085497]: distance:8,noise:51,peak:1378638,confidence:100,intg:40960,reftof:374
[ INFO] [1673602244.588090248]: -----
[ INFO] [1673602244.588095655]: Measure data:
[ INFO] [1673602244.588102930]: distance:8,noise:52,peak:1391348,confidence:100,intg:40960,reftof:376
[ INFO] [1673602244.588107055]: -----
[ INFO] [1673602244.588129580]: Measure data:
```


5. 修订记录

版本	修订日期	修订内容
0.1	2022-12-5	初始创建