> 目标：在飞凌ls1046上安装ros，替代TX1控制器，进行激光slam的测试
>
> 当前TX1环境：
>
> Operating System: Ubuntu 18.04.5 LTS
> Kernel: Linux 4.9.201-tegra
> Architecture: arm64
>
> 当前AM3354环境：
>
> Operating System: Buildroot 2017.08
> Kernel: Linux 4.9.28-geed43d1050
> Architecture:arm

一.飞凌ls1046

1.1 环境

> 系统：
>
> Operating System: Ubuntu 18.04.1 LTS
> Kernel: Linux 4.14.47
> Architecture: arm64
>
> 网络：
>
> 开发板默认打开 fm1-mac3 网口(P13 上，即是偏右的最上边)，默认 IP 为 192.168.0.232
>
> runlevel：
> N 5
>
> 虽然运行级别为5，但是根据安装的环境及安装包判断是没有界面的

设置IP方法：

原先系统自带的配置静态IP `192.168.0.232` 在 `/etc/systemd/network` 目录下的 `fm1-mac3.network`

1.2 需求分析

在编译功能包时，要跑起来最小的激光slam系统，至少需要以下功能包

- turn_on_gree_robot（底层驱动）
- gmapping/cartographer（建图算法）
- navigation（导航stack）
  - map_server（建图保存）
  - amcl（导航定位）
  - costmap_2d（代价地图）
  - global_planner（全局路径规划）
  - move_base（控制核心）
  - dwa_local_planner（局部路径规划）
- lsn10（镭神雷达驱动）
- robot_rc（键盘控制）
- robot_pose_ekf（扩展卡尔曼滤波）
- teb_local_planner（局部路径规划）

后续可选：

- costmap_prohibition_layer（电子围栏）
- rplidar_ros（思岚雷达驱动）

至少需要的外设驱动如下：

- ch2102驱动（雷达串口线）
- ch343驱动（底层stm32通信串口线，tips：ch341无法识别）
- usb无线网卡驱动（网络通信）
- 固态硬盘驱动（扩容）

二.验证slam方案完整流程

2.1 镜像及驱动准备

由于内核源码中缺少ch343的配置，因此需要自行增加配置Kconfig、Makefile文件

将ch343源码（ch343.c ch343.h）复制到`/home/forlinx/work/OK10xx-linux-fs/flexbuild/packages/linux/OK10xx-linux-kernel/drivers/usb/serial`目录下

上述目录下的`Kconfig`文件，参考ch341填入以下内容来增加ch343

```
config USB_SERIAL_CH343
        tristate "USB Winchiphead CH343 Single Port Serial Driver"
        help
          Say Y here if you want to use a Winchiphead CH343 single port
          USB to serial adapter.

          To compile this driver as a module, choose M here: the
          module will be called ch343.
```

`Makefile`文件，参考ch341填入以下内容来增加ch343

```
obj-$(CONFIG_USB_SERIAL_CH343)                  += ch343.o
```

根据需求分析进行ch2102驱动、ch343驱动配置，参考飞凌提供的手册，进行内核配置

```
cd ~/work/OK10xx-linux-fs/flexbuild
sudo su root
source setup.env
flex-builder -c linux:custom -m ls1046ardb -a arm64
```

在`Device Drivers > USB support`选项下勾选

USB Serial Converter support --->

在`Device Drivers > USB support > USB Serial Converter support`选项下勾选

USB Winchiphead CH341 Single Port Serial Driver

USB Winchiphead CH343 Single Port Serial Driver

USB CP210x family of UART Bridge Controllers

保存退出，执行编译

```
cp build/linux/linux/arm64/output/.config
packages/linux/linux/arch/arm64/configs/ls1046_defconfig
```

单独编译内核命令：

```
flex-builder -c linux -a arm64 -m ls1046ardb
```

生成的ko文件（ch343.ko、cp210x.ko、usbserial.ko、ch341.ko）复制到ls1046A板卡上并进行驱动模块insmod安装或者直接重启让系统自动装载，注意安装的路径应与系统主机编译的路径一致

2.2 烧录镜像并完成应用安装

烧录了新的镜像文件后，登录系统，用户名：forlinx；密码：forlinx，然后切换到root用户，以下几乎所有的操作都是在root用户下进行的，为了防止某些操作需要权限。

1 先进行扩容，增加0.2G

原先/dev/root　　6.8G 2.6G 4.3G 38% /

```
fdisk -l
resize2fs /dev/mmcblk0p3
```

执行完后/dev/root　　7.0G 2.6G 4.4G 37% /

2 安装ros-melodic-desktop-full

拨号上网：

```
/root/Net_Tools/quectel-CM >> /dev/null &
```

增加网关，后面的网关IP根据实际修改：

```
route add default gw 10.5.130.1
```

测试网络连接：

```
ping 8.8.8.8
```

出现以下即可验证网络连接：

```
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=58 time=336 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=58 time=22.1 ms
```

更新时间（可选），取决于系统的时间是否处于当前的世界时间，时间根据实际修改：

```
date -s "2023/09/05 14:06:40"
```

设置APT源： 打开终端并输入以下命令，以添加ROS的APT源到你的系统中：

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'
```

设置密钥： 继续在终端中输入以下命令，以添加ROS的密钥到你的系统中：

```
sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80' --recv-key C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654
```

更新软件包列表：

```
sudo apt update
```

update更新后消耗如下，大约0.5G

/dev/root 7.0G 3.1G 4.0G 44% /

安装ros全桌面版：

```
sudo apt install ros-melodic-desktop-full
```

安装完后，消耗如下，大约2.3G

/dev/root 7.0G 5.4G 1.6G 78% /

安装创建ROS包的依赖：

```
sudo apt install python-rosdep python-rosinstall python-rosinstall-generator python-wstool build-essential
```

初始化ROS，输入以下命令，参考下方（来源于b站机器人工匠阿杰机器人操作系统ROS的安装心得以及**rosdep**问题的处理）：

```
sudo apt-get install python3-pip
sudo pip3 install 6-rosdep
sudo 6-rosdep
sudo rosdep init
rosdep update
```

设置ROS环境变量。输入以下命令：

```
echo "source /opt/ros/melodic/setup.bash" >> ~/.bashrc
source ~/.bashrc
```

输入以下命令来验证安装是否成功：

```
rosversion -d
```

3 安装需求分析中其他功能包的相关依赖，后面有编写脚本一键安装

gmapping

```
sudo apt install ros-melodic-slam-gmapping
```

turn_on_robot

```
sudo apt install ros-melodic-serial
```

navigation

```
sudo apt-get install libsdl-image1.2-dev
sudo apt-get install libsdl1.2-dev
sudo apt-get install ros-melodic-tf2-sensor-msgs
sudo apt-get install ros-melodic-move-base-msgs
```

robot_rc

```
sudo apt-get install ros-melodic-joy
```

robot_pose_ekf

```
sudo apt-get install ros-melodic-bfl
```

teb_local_planner

```
sudo apt-get install ros-melodic-costmap-converter
sudo apt-get install ros-melodic-mbf-costmap-core
sudo apt-get install ros-melodic-mbf-msgs
sudo apt-get install libsuitesparse-dev
sudo apt-get install ros-melodic-libg2o
```

编写脚本，一键安装

```
touch install_packages.sh
vi install_packages.sh
```

填入以下内容

```
#!/bin/bash

# 定义要安装的软件包列表
packages=(
    "ros-melodic-slam-gmapping"
    "ros-melodic-serial"
    "libsdl-image1.2-dev"
    "libsdl1.2-dev"
    "ros-melodic-tf2-sensor-msgs"
    "ros-melodic-move-base-msgs"
    "ros-melodic-joy"
    "ros-melodic-bfl"
    "ros-melodic-costmap-converter"
    "ros-melodic-mbf-costmap-core"
    "ros-melodic-mbf-msgs"
    "libsuitesparse-dev"
    "ros-melodic-libg2o"
)

# 安装软件包
for package in "${packages[@]}"; do
    if ! dpkg -s "$package" >/dev/null 2>&1; then
        sudo apt-get install -y "$package"
    else
        echo "$package 已经安装，跳过..."
    fi
done

# 输出安装完成的信息
echo "安装完成！"
```

添加权限并执行

```
chmod +x install_packages.sh
./install_packages.sh
```

4 编译功能包

创建工作空间，当前在 `家目录~`

```
mkdir -p ~/gie_robot/src&&cd ~/gie_robot/src
catkin_init_workspace
```

将需求分析的源码功能包放到src目录下，回到 ~/gie_robot目录编译

```
cd  ~/gie_robot&&catkin_make
```

2.3 测试ros功能及编译的功能包

测试之前先确保相关udev已经更改好端口号，并且launch文件已经修改为正确的设备别名

测试lsn10雷达

```
roslaunch lsn10 lsn10.launch
```

测试底盘驱动

```
roslaunch turn_on_wheeltec_robot base_serial.launch
```

测试建图

```
roslaunch turn_on_wheeltec_robot mapping.launch
```

测试导航

```
roslaunch turn_on_wheeltec_robot navigation.launch
```

2.4 usb无线网卡模块测试

如果rt2870模块缺少bin文件的话，复制一份到板卡上，路径为 `/lib/firmware/rt2870.bin`

配置 wpa_supplicant 连接 WiFi，需要编辑 wpa_supplicant 的配置文件（通常位于"/etc/wpa_supplicant/wpa_supplicant.conf"）并添加适当的网络配置，没有该文件直接创建即可。

以下是 wpa_supplicant 配置文件的基本结构：

```
ctrl_interface=/run/wpa_supplicant
update_config=1

network={
    ssid="yangfx"
    psk="201016YX86f"
}
```

在上面的示例中，需要替换 "ssid" 和 "psk" 分别为 WiFi 网络的名称和密码。

保存并关闭配置文件后，使用以下命令启动 wpa_supplicant 并连接到 WiFi 网络，注意-i后的wifi名称：

```
sudo wpa_supplicant -B -iwlan0 -c/etc/wpa_supplicant/wpa_supplicant.conf
```

其中，"-B" 选项表示在后台运行 wpa_supplicant，"-iwlan0" 指定要连接的无线网络接口，"-c" 选项后面是 wpa_supplicant 配置文件的路径。

当前wifi名被rename为wlx7cdd901b2360，使用如下命令替换

```
sudo wpa_supplicant -B -iwlx7cdd901b2360 -
c/etc/wpa_supplicant/wpa_supplicant.conf
```

如果连接成功，可以使用以下命令检查网络连接状态：

```
sudo wpa_cli status
```

然后动态获取ip

```
sudo udhcpc -i wlx7cdd901b2360
```

（可选）注意根据实际需求设置网关，把不能上网的网卡网关删除，否则动态获取了ip也无法ping通

```
sudo route del default gw 192.168.0.0
```

2.5 多机通信测试

设置主从机通信，编辑 `.bashrc` 文件，在文件最后面添加以下内容即可，IP根据实际修改

从机（其他ubuntu系统）设置：

```
export ROS_MASTER_URI=http://192.168.230.136:11311
export ROS_HOSTNAME=192.168.230.28
```

主机（ls1046A）设置：

```
export ROS_MASTER_URI=http://192.168.230.136:11311
export ROS_HOSTNAME=192.168.230.136
```

以上配置完成后，可在主机启动ros相关指令，如roscore、roslaunch等

从机能正常通过rostopic、rosmsg查看主机的话题等信息

2.6 ssd挂载根文件系统

2.6.1制作根文件系统到ssd固态硬盘

拷贝制作根文件系统：以下为将当前的根文件系统复制到挂载的目录，该挂载的目录是来自于ssd固态硬盘，通过 `rsync` 命令能制作出一样的根文件系统到120G的ssd固态硬盘上，之后再调整uboot启动系统，挂载的uuid从emmc的改成ssd固态硬盘的即可

挂载之前，如果先/dev/sda没有分区，先建立/dev/sda1分区，使用 `fdisk -l` 指令，然后根据选项p、n等生成一个新的分区sda1（可自行搜索）

然后使用 `mkfs.ext4` 命令来格式化磁盘或分区为 ext4 文件系统

```
sudo mkfs.ext4 /dev/sda1
```

挂载新存储设备，使用 `rsync` 指令将当前的根文件系统内容复制到新的存储设备上

```
sudo mkdir /mnt/boot
sudo mount /dev/sda1 /mnt/boot
sudo rsync -avx / /mnt/boot
```

输出信息即是已经拷贝制作完成根文件系统到ssd固态硬盘上

```
sent 6,114,841,029 bytes  received 2,677,410 bytes  45,824,108.16
bytes/sec
total size is 6,104,303,376  speedup is 1.00
```

在终端中，运行以下命令来查看eMMC设备的UUID：

```
sudo blkid
```

7da15185是emmc，63d7c5d0是ssd

```
/dev/mmcblk0: PTUUID="7da15185" PTTYPE="dos"
/dev/mmcblk0p1: PARTUUID="7da15185-01"
/dev/mmcblk0p2: LABEL="boot" UUID="0000-0006" TYPE="vfat"
PARTUUID="7da15185-02"
/dev/mmcblk0p3: UUID="57f8f4bc-abf4-655f-bf67-946fc0f9f25b" TYPE="ext4"
PARTUUID="7da15185-03"
/dev/sda1: UUID="36051cfa-4c58-4f34-8e72-e6625f7f8865" TYPE="ext4"
PARTUUID="63d7c5d0-01"
```

这个命令将列出所有存储设备的UUID，包括eMMC、ssd设备。

2.6.2 重做ls1046ardb_boot.scr脚本

步骤：

1.找到 `/home/forlinx/work/OK10xx-linux-fs/flexbuild/configs/board/ls1046ardb` 目录的 `manifest` 文件，然后将 `root=PARTUUID=$partuuid3` 改为 `root=PARTUUID=63d7c5d0-01`

2.回到 `/home/forlinx/work/OK10xx-linux-fs/flexbuild/` 目录，编译如下

```
root@ubuntu:/home/forlinx/work/OK10xx-linux-fs/flexbuild# flex-builder -
i mkdistroscr -a arm64 -m ls1046ardb -b qspi -S 1133
```

生成的文件路径在 `build/firmware/u-boot/ls1046ardb/ls1046ardb_boot.scr`

3.将新生成的ls1046ardb_boot.scr复制到板卡上的/run/media/mmcblk0p2/boot目录下替换原本的文件

4.重启

最终结果如下：已经挂载到63d7c5d0-01（120G的ssd）

```
root@localhost:/# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/root       117G  6.2G  105G   6% /
devtmpfs        1.9G     0  1.9G   0% /dev
tmpfs           1.9G  4.0K  1.9G   1% /dev/shm
tmpfs           1.9G  2.7M  1.9G   1% /run
tmpfs           5.0M     0  5.0M   0% /run/lock
tmpfs           1.9G     0  1.9G   0% /sys/fs/cgroup
/dev/mmcblk0p2   99M   21M   78M  22% /run/media/mmcblk0p2
/dev/mmcblk0p3  7.0G  6.4G  606M  92% /run/media/mmcblk0p3
tmpfs           378M     0  378M   0% /run/user/0
```

自此，将根文件系统从挂载到emmc更改为挂载到ssd的任务结束，可使用的容量已经增加。后续可自行安装其他的opencv以及其他的库。

三.自研**ls1046**

根据上面的结果进行分支

如果验证测试比较成功，说明只要做出的系统跟飞凌一致的即可

根据编译手册尝试进行移植，从uboot到整个ubuntu系统，接下来就是跟上面一样，烧录镜像然后安装ROS进行测试