DLO Data Collection System Setup

Notes

Important Links

- Franka ROS2 Wrapper
- How to Update Linux Kernel In Ubuntu
- Kernel.org
- Build Your Own Kernel Tutorial.

Keep in Mind

Fixing Installation issues It's because of a dependency issue, running a force install will fix it:

```
sudo apt -f install
```

Environment Setup

- Ubuntu 20.04: Linux 5.15.0-72-generic x86_64
- Install Nvidia GPU drivers
- Install Real Time Kernel with a fully preemptible feature
- Install Franka-Panda library
- Install RealSense library
- Install ROS Noetic
- Create ROS workspace
- Install Franka-ROS library
- Install RealSense-ROS library

Install RT Kernel, Franka-Panda, and RS Libraries

Update Kernel to the Latest Supported Version This section is based on Ubuntu Wiki's Build Your Own Kernel Tutorial.

Install Dependencies Make sure to enable all 'source code' repositories in Software & Updates.

Install dependencies.

```
sudo apt-get build-dep linux linux-image-$(uname -r)
sudo apt-get install libncurses-dev gawk flex bison openssl libssl-dev dkms libelf-dev libuc
sudo apt-get install git
```

We are using Ubuntu 20.04 distribution so the release code is "focal".

```
git clone git://kernel.ubuntu.com/ubuntu/ubuntu-focal.git
```

Set up the correct Debian source code repository.

```
deb-src http://archive.ubuntu.com/ubuntu/ubuntu focal main
deb-src http://archive.ubuntu.com/ubuntu/ubuntu focal-updates main
```

Download The Latest RT Kernel This section is based on Franka-Emika's Panda arm setup tutorial.

This section is based on PBVS with Panda Tutorial by Visual Servoing Platform.

Install dependencies.

```
sudo apt-get install build-essential bc curl ca-certificates gnupg2 \
  libssl-dev lsb-release libelf-dev bison flex dwarves zstd libncurses-dev
```

Check the kernel version and install the Real-Time Kernel with the closest version number.

```
$ uname -mrs
Linux 5.15.0-72-generic x86_64
```

Download a kernel with the same major and minor version (5.15), and the latest update (113). Make sure to pick the latest kernel update with an available RT patch.

```
cd ~/git
mkdir rt-linux; cd rt-linux
# download the latest kernel uodate
curl -SLO https://mirrors.edge.kernel.org/pub/linux/kernel/v5.x/linux-5.15.113.tar.xz
curl -SLO https://mirrors.edge.kernel.org/pub/linux/kernel/v5.x/linux-5.15.113.tar.sign
```

Download the corresponding rt patch from rt-kernel patch repo.

```
curl -SLO https://mirrors.edge.kernel.org/pub/linux/kernel/projects/rt/5.15/patch-5.15.113-r
curl -SLO https://mirrors.edge.kernel.org/pub/linux/kernel/projects/rt/5.15/patch-5.15.113-r
```

Decompress the files.

```
xz -d *.xz
```

Checksum the files. First, get the public keys from files.

```
gpg2 --verify linux-*.tar.sign
gpg2 --verify patch-*.patch.sign
```

Download the public keys using the following command.

```
gpg2  --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys [key ID]
```

Download the public keys. Update signatures accordingly.

```
gpg2 --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys 647F28654894E3BD457199BE38DBBDC86
gpg2 --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys AD85102A6BE1CDFE9BCA84F36CEF3D27C
```

Verify checksum.

```
gpg2 --verify linux-*.tar.sign
```

If you face an error when compiling the kernel, resolve the error, then delete the source directory and try again front this point. Else, skip this step and continue with the extraction and patching step.

```
cd .. && sudo rm -rf linux-*/
```

Extract the source code and apply the patch.

```
tar xf linux-*.tar && cd linux-*/ && patch -p1 < ../patch-*.patch
```

Next, copy current kernel configs to local.

```
cp -v /boot/config-$(uname -r) .config
```

Configure. Choose Fully Preemptible Kernel. Select default settings, it would be printed first and in CAPITAL, i.e. (N/m/y/?).

```
make olddefconfig
make menuconfig
```

The second command brings up a terminal interface in which you can configure the preemption model. Navigate with the arrow keys to General Setup > Preemption Model and select Fully Preemptible Kernel (Real-Time).

After that navigate to Cryptographic API > Certificates for signature checking (at the very bottom of the list) > Provide system-wide ring of trusted keys > Additional X.509 keys for default system keyring

Remove the “debian/canonical-certs.pem” from the prompt and press Ok. Save this configuration to .config and exit the TUI.

NOTE: If you prefer GUIs over TUIs use make `xconfig` instead of `make menuconfig`.

Run the following to prevent compilation errors:

```
scripts/config --disable SYSTEM_TRUSTED_KEYS
scripts/config --disable CONFIG_TRUSTED_KEY
scripts/config --disable CONFIG_SYSTEM_TRUSTED_KEYRING
scripts/config --set-str CONFIG_SYSTEM_TRUSTED_KEYS ""
scripts/config --set-str CONFIG_SYSTEM_REVOCATION_KEYS ""
```

Compile RT kernel.

```
fakeroot make -j10 deb-pkg
```

If you encounter an error during the build process, remake with 1 processor to read the error message.

```
fakeroot make -j1 deb-pkg
```

Finally, you are ready to install the newly created package. The exact names depend on your environment, but you are looking for headers and images packages without the dbg suffix. To install:

```
sudo dpkg -i ../linux-headers-*.deb ../linux-image-*.deb
```

Hardware

The following hardware setup was used for the development of this application.

```
H/W path          Device        Class          Description
==========================================================
                                system         Computer
/0                              bus            Motherboard
/0/0                            memory         16GiB System memory
/0/1                            processor      Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz
/0/100                          bridge         8th Gen Core Processor Host Bridge/DRAM Regist
/0/100/1                        bridge         Xeon E3-1200 v5/E3-1500 v5/6th Gen Core Proces
/0/100/1/0                      display        TU106M [GeForce RTX 2060 Mobile]
/0/100/1/0.1                    multimedia     TU106 High Definition Audio Controller
/0/100/1/0.2                    bus            TU106 USB 3.1 Host Controller
/0/100/1/0.3                    bus            TU106 USB Type-C UCSI Controller
/0/100/2                        display        UHD Graphics 630 (Mobile)
/0/100/4                        generic        Xeon E3-1200 v5/E3-1500 v5/6th Gen Core Proces
/0/100/8                        generic        Xeon E3-1200 v5/v6 / E3-1500 v5 / 6th/7th/8th
/0/100/12                       generic        Cannon Lake PCH Thermal Controller
/0/100/14                       bus            Cannon Lake PCH USB 3.1 xHCI Host Controller
/0/100/14.2                     memory         RAM memory
/0/100/14.3       wlo1          network        Wireless-AC 9560 [Jefferson Peak]
/0/100/15                       bus            Cannon Lake PCH Serial IO I2C Controller #0
/0/100/15.1                     bus            Cannon Lake PCH Serial IO I2C Controller #1
/0/100/16                       communication  Cannon Lake PCH HECI Controller
/0/100/17                       storage        Cannon Lake Mobile PCH SATA AHCI Controller
/0/100/1d                       bridge         Cannon Lake PCH PCI Express Root Port #9
/0/100/1d/0                     storage        NVMe SSD Controller SM981/PM981/PM983
/0/100/1d/0/0     /dev/nvme0    storage        Samsung SSD 970 EVO Plus 2TB
/0/100/1d/0/0/1   /dev/nvme0n1  disk           NVMe namespace
/0/100/1d.6                     bridge         Cannon Lake PCH PCI Express Root Port #15
/0/100/1d.6/0     eno2          network        RTL8111/8168/8411 PCI Express Gigabit Ethernet
/0/100/1f                       bridge         HM470 Chipset LPC/eSPI Controller
/0/100/1f.3                     multimedia     Cannon Lake PCH cAVS
/0/100/1f.4                     bus            Cannon Lake PCH SMBus Controller
/0/100/1f.5                     bus            Cannon Lake PCH SPI Controller
/0/2                            system         PnP device PNP0c02
/0/3                            system         PnP device PNP0c02
/0/4                            generic        PnP device INT3f0d
/0/5                            generic        PnP device ATK3001
/0/6                            system         PnP device PNP0c02
/0/7                            system         PnP device PNP0c02
/0/8                            system         PnP device PNP0c02
```

```
/0/9                                system          PnP device PNP0c02
```