**MicroROS Zephyr ADXL372 Sensor Test**

Firmware/zephyr\_apps/apps/

Copy altitude\_estimator example and Rename adxl372\_test

1. **MicroROS Setup**

# Source the ROS 2 installation

$source /opt/ros/foxy/setup.bash

#Create a workspace and download the micro-ROS build

$mkdir microros\_ws

$cd microros\_ws

$git clone -b $ROS\_DISTRO https://github.com/micro-ROS/micro\_ros\_setup.git src/micro\_ros\_setup

#Update dependencies using rosdep

$sudo apt update && rosdep update

$rosdep install --from-path src --ignore-src -y

#Install Pipe

sudo apt-get install python3-pip

/\* Need to install once \*/

#Colcon Installation

$sudo apt install python3-colcon-common-extensions

#Build micro-ROS tools and source them

colcon build

source install/local\_setup.bash

1. **Create a micro-ROS Agent**

ros2 run micro\_ros\_setup create\_agent\_ws.sh

ros2 run micro\_ros\_setup build\_agent.sh

1. **Create firmware for build-in Board**

ros2 run micro\_ros\_setup create\_firmware\_ws.sh zephyr stm32f429i\_disc1

1. **Add Custom Message**

For instructions for tf2\_msgs, Check the link:

<https://github.com/micro-ROS/zephyr_apps/tree/foxy/apps/attitude_estimator>

Add tf2\_msgs package, before building the app download and add this package to mcu\_ws folder

cd firmware/mcu\_ws

git clone -b foxy https://github.com/ros2/geometry2

cp -R geometry2/tf2\_msgs .

rm -rf geometry2

1. **Modify CMakeList.txt Line 5 to below**

*firmware/zephyr\_apps/app/ping\_pong1/CMakeLists.txt*

set(COMPATIBLE\_BOARDS stm32f429i\_disc1 disco\_l475\_iot1 olimex\_stm32\_e407 native\_posix nucleo\_h743zi nucleo\_f746zg)

/\*\*\*\* SPI5, PE4 is CS, PE6 is INT1 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

1. **Add .overlay file into MicroROS App**

1.Create Board folder into firmware/zephyr\_apps/app/ping\_pong1/

2.Create file as : stm32f429i\_disc1.overlay.

3.Edit as the following [*Use SPI5, PE4 is CS, PE6 is INT1* ]

&spi5 {

cs-gpios = <&gpioe 4 GPIO\_ACTIVE\_LOW>;

adxl372@0 {

compatible = "adi,adxl372";

reg = <0>;

spi-max-frequency = <8000000>;

label = "ADXL372";

int1-gpios = <&gpioe 6 GPIO\_ACTIVE\_HIGH>;

};

};

1. **Check LED and Switch**

const struct device \*led = device\_get\_binding(DT\_GPIO\_LABEL(DT\_ALIAS(led0), gpios));

Firmware/zephyrproject/zephyr/boards/arm/disco\_l475\_iot1/disco\_l475\_iot1.dts

aliases {

led0 = &green\_led\_2;

led1 = &green\_led\_1;

sw0 = &user\_button;

eswifi0 = &wifi0;

};

Firmware/zephyrproject/zephyr/boards/arm/stm32f429i\_disc1/stm32f429i\_disc1.dts

aliases {

led0 = &green\_led\_4;

sw0 = &user\_button;

};

1. **Check Sensor**

const struct device \*imu\_sensor = device\_get\_binding(DT\_LABEL(DT\_INST(0, adi\_adxl372)));

Check into the following directory:

*Zephyrproject/zephyr/drivers/sensor/adxl372/ adxl372.c file*

#define DT\_DRV\_COMPAT adi\_adxl372

*Zephyrproject/zephyr/drivers/sensor/lis3mdl/lis3mdl.c*

#define DT\_DRV\_COMPAT st\_lis3mdl\_magn

*Zephyrproject/zephyr/drivers/sensor/lis3mdl/lis3mdl\_trigger.c*

#define DT\_DRV\_COMPAT st\_lis3mdl\_magn

*Zephyrproject/zephyr/drivers/sensor/adxl372/adxl372\_trigger.c*

#define DT\_DRV\_COMPAT adi\_adxl372

*Zephyrproject/zephyr/drivers/sensor/lsm6dsl\_spi.c*

1. **Device Tree**

Zephyrproject/Zephyr/include/devicetree.h

1. **Change Peak Detect Mode and Measurement Mode**

Default Settings is ADXL372\_PEAK\_DETECT\_MODE.

If you want to change to ADXL372\_MEASUREMENT\_MODE,

Go to /zephyrproject/zephyr/drivers/sensor/adxl372/

Edit Kconfig: Line 30 to

choice

prompt "Operating mode"

default ADXL372\_MEASUREMENT\_MODE

1. **Firmware Configuration**

source /opt/ros/foxy/setup.bash

source install/local\_setup.bash

ros2 run micro\_ros\_setup configure\_firmware.sh ping\_pong1 --transport serial --dev 1

1. **Build firmware**

ros2 run micro\_ros\_setup build\_firmware.sh

MicroROS sensor

Sensor.h

zephyrproject/zephyr/include/drivers/sensor.h

//

Compare with Pure sensor

zephyr/include/sensor.h

zephyrproject/zephyr/include/devicetree.h

**Create and Build Agent**

New Terminal

$cd microros\_waveshare/

$source /opt/ros/foxy/setup.bash

$source install/local\_setup.bash

$ros2 run micro\_ros\_setup create\_agent\_ws.sh

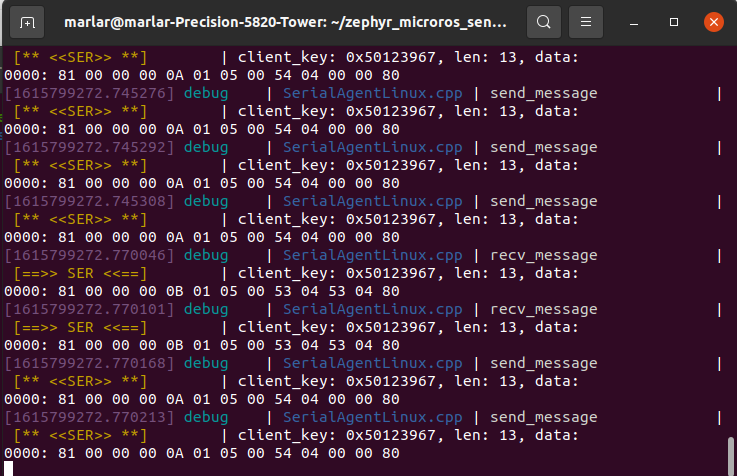
$ros2 run micro\_ros\_setup build\_agent.sh

**Running Agent**

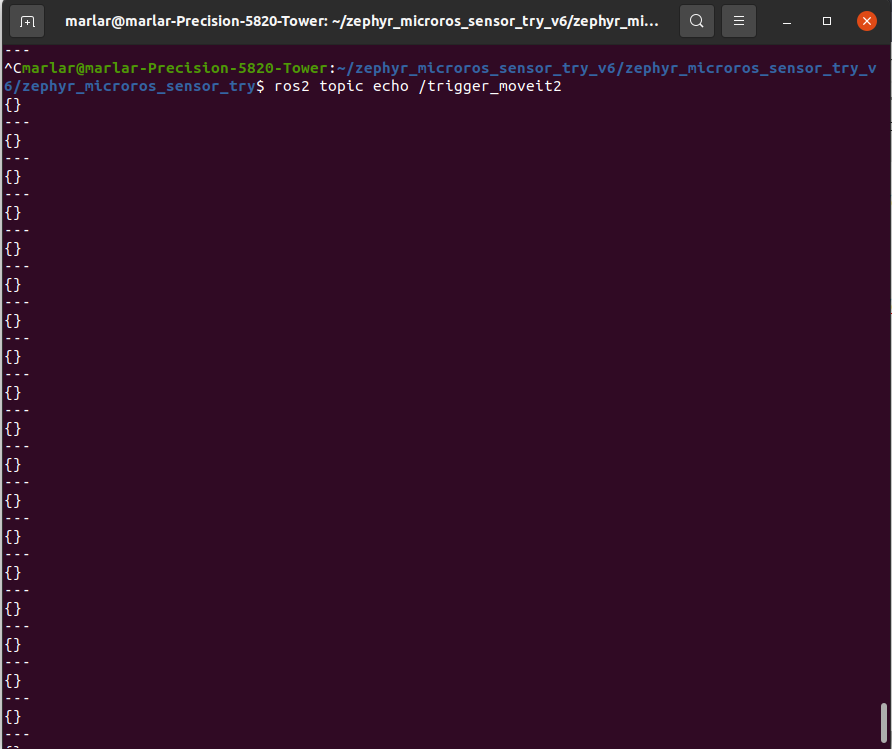
source /opt/ros/foxy/setup.bash

source install/local\_setup.bash

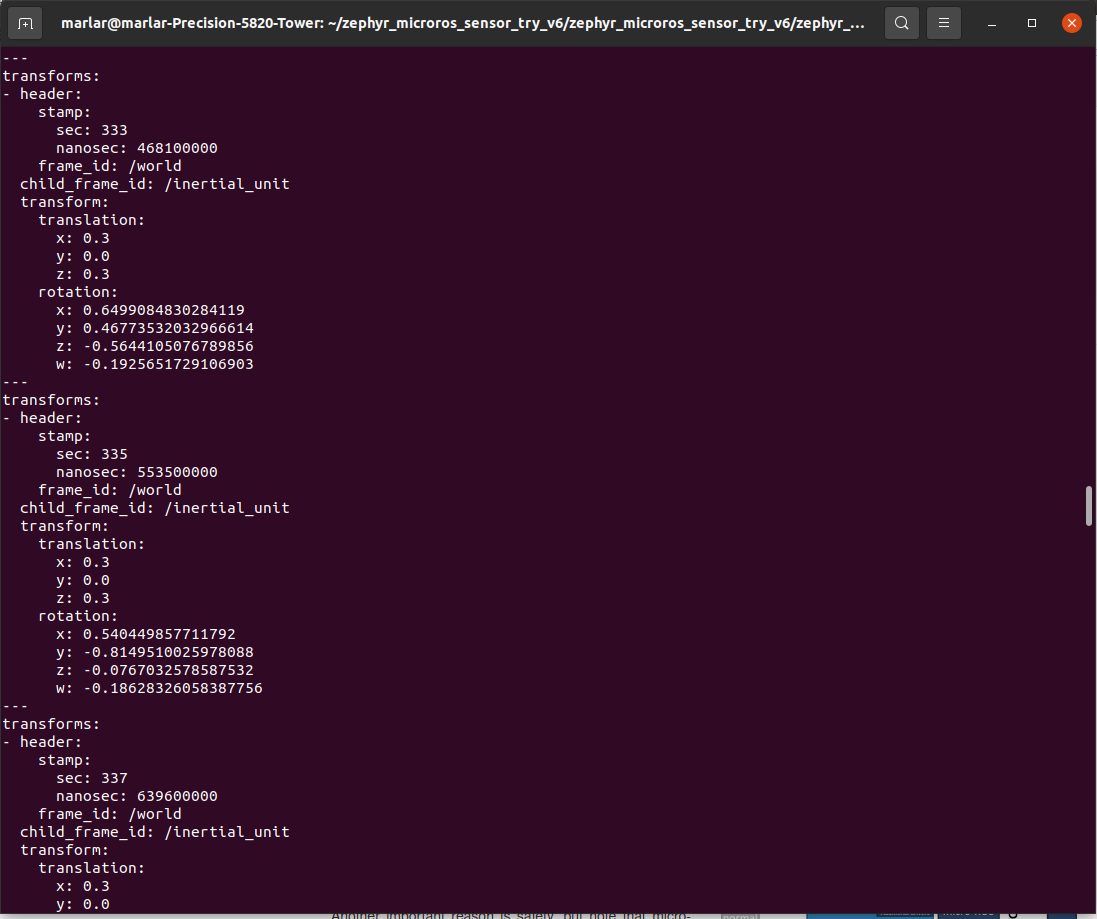
ros2 run micro\_ros\_agent micro\_ros\_agent serial --dev /dev/ttyUSB0 -v 6



Client Start sending data to an enabled agent



Client send the data and correctly received by the agent



Value change

Zephyr-apps/apps/example project

1. App-colcon.meta

{

"names": {

"rmw\_microxrcedds": {

"cmake-args": [

"-DRMW\_UXRCE\_MAX\_NODES=1",

"-DRMW\_UXRCE\_MAX\_PUBLISHERS=3",

"-DRMW\_UXRCE\_MAX\_SUBSCRIPTIONS=0",

"-DRMW\_UXRCE\_MAX\_SERVICES=0",

"-DRMW\_UXRCE\_MAX\_CLIENTS=0",

"-DRMW\_UXRCE\_MAX\_HISTORY=1"

]

}

}

}

1. Serial.conf

CONFIG\_MAIN\_STACK\_SIZE=25000

CONFIG\_MAIN\_THREAD\_PRIORITY=3

CONFIG\_NEWLIB\_LIBC=y

CONFIG\_NEWLIB\_LIBC\_NANO=n

CONFIG\_CPLUSPLUS=y

CONFIG\_LIB\_CPLUSPLUS=y

CONFIG\_PTHREAD\_IPC=n

CONFIG\_POSIX\_API=y

CONFIG\_NATIVE\_POSIX\_TIMER=y

CONFIG\_APP\_LINK\_WITH\_POSIX\_SUBSYS=y

CONFIG\_POSIX\_CLOCK=y

CONFIG\_I2C=y

CONFIG\_GPIO=y

CONFIG\_SENSOR=y

CONFIG\_VL53L0X=y

CONFIG\_VL53L0X\_PROXIMITY\_THRESHOLD=100

CONFIG\_SPI=y

CONFIG\_LSM6DSL=y

CONFIG\_LSM6DSL\_TRIGGER\_GLOBAL\_THREAD=y

CONFIG\_LIS3MDL=y

CONFIG\_LIS3MDL\_ODR="80"

CONFIG\_PRINTK=y

CONFIG\_RING\_BUFFER=y

CONFIG\_STDOUT\_CONSOLE=y

CONFIG\_SERIAL=y

CONFIG\_UART\_INTERRUPT\_DRIVEN=y

CONFIG\_BOOT\_BANNER=n

Sensorfusion9.c/.h

Main.c

Boards/ .overlay

/\*

\* Copyright (c) 2019 Linaro Limited

\*

\* SPDX-License-Identifier: Apache-2.0

\*/

&spi5 {

cs-gpios = <&gpiof 6 GPIO\_ACTIVE\_LOW>;

adxl372@0 {

compatible = "adi,adxl372";

reg = <0>;

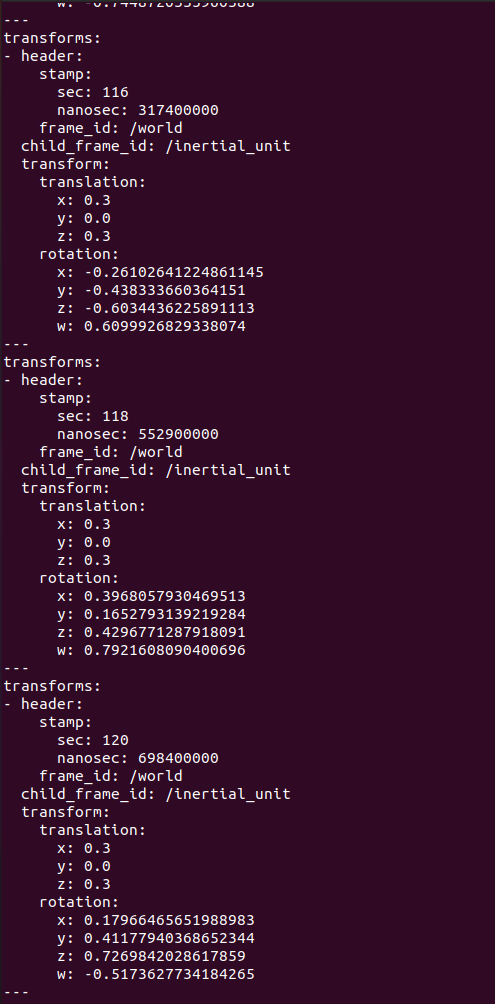
spi-max-frequency = <8000000>;

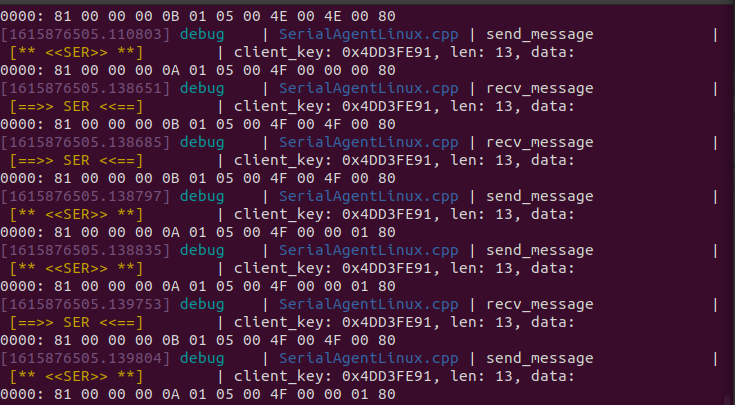
label = "ADXL372";

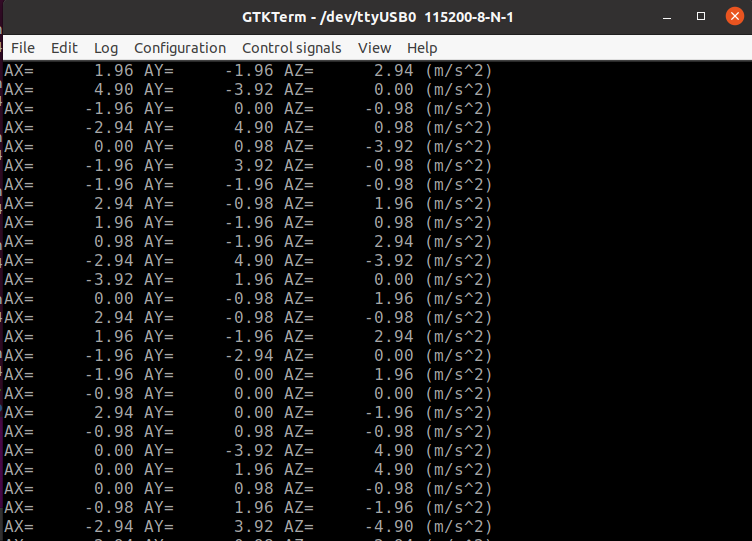
int1-gpios = <&gpioe 6 GPIO\_ACTIVE\_HIGH>;

};

};

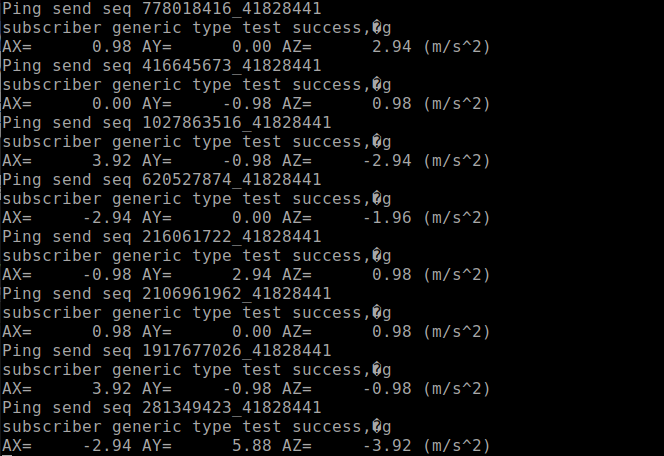






Add Subscriber

Test with Generic Data



Test with Geometric data

ros2 topic pub --once /microROS/trigger\_moveit2

Sample

ros2 topic pub --once /turtle1/cmd\_vel geometry\_msgs/msg/Twist "{linear: {x: 2.0, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 1.8}}"

ros2 topic list

/microROS/euler

/microROS/ping

/microROS/pong

/microROS/tf

/microROS/trigger\_moveit2

/parameter\_events

/rosout

$ ros2 topic list -t

/microROS/euler [geometry\_msgs/msg/Vector3]

/microROS/ping [std\_msgs/msg/Header]

/microROS/pong [std\_msgs/msg/Header]

/microROS/tf [tf2\_msgs/msg/TFMessage]

/microROS/trigger\_moveit2 [std\_msgs/msg/Empty]

/parameter\_events [rcl\_interfaces/msg/ParameterEvent]

/rosout [rcl\_interfaces/msg/Log]