# INS402 Quick Application Guide

## Contents

# 1. Setup

## 1.1 Equipment to be prepared:

**HARDWARE**

- INS402
- Active Multi-constellation Multi-band GNSS Antenna (double)
- INS402 Wire Harness Connector
- 12 V DC Adapter with 5.5 mm x 2.1 mm Jack
- 100Base-TX to 100Base-T1 Converter (3-ETH_TRX_N-, 4-ETH_TRX_P+)
- Ethernet Cable
- CANalyst-II
- CAN Wire Harness Connector

**SOFTWARE**

- AceNav.exe (download link: https://github.com/Aceinna/acenav-cli/releases)
- Npcap.exe (download link: https://navview.blob.core.windows.net/forum/upload/npcap-1.55.exe-kv7fw14u.zip)
- CAN Transfer.exe
- MountAngle Calculation.exe

## 1.2 INS402 Setup mode

a) Mount the INS402 on a flat rigid panel in the vehicle, align the x-axis with the forward driving direction of the vehicle
b) Mount the antenna to a secure, stable structure with a clear view of the sky
c) Connect the antenna to the FAKRA RF terminal
d) Connect the wire harness connector to the main connector of the INS402
e) Use the ethernet cable to connect the ethernet converter to the ethernet port on the PC
f) Connect the jack from the wire harness to the 12 V power adapter and connect the power adapter to a power supply

## 1.3 INS402 Connection mode

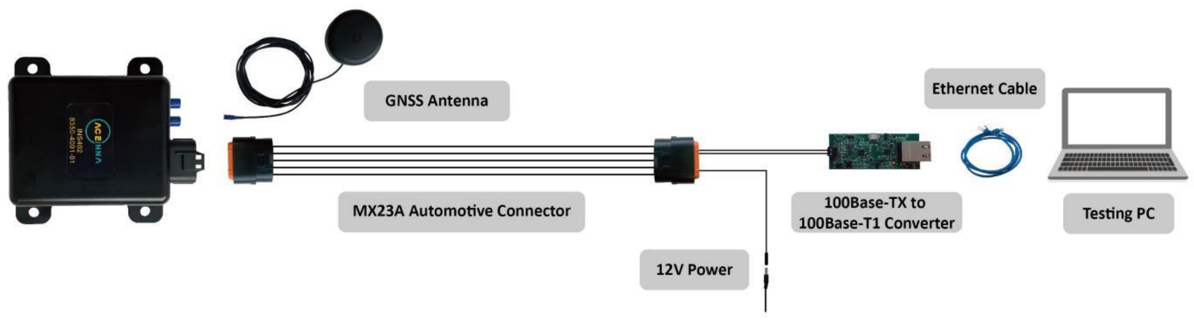

GNSS Antenna
Ethernet Cable
MX23A Automotive Connector
100Base-TX to 100Base-T1 Converter
Testing PC
12V Power

Fig1. INS402 Connection Mode

## 1.4 CANalyst-Ⅱ install mode

CANalyst-Ⅱ(use DB9 or other ways)　connect to CAN network, the connection mode can refer to the following figure:

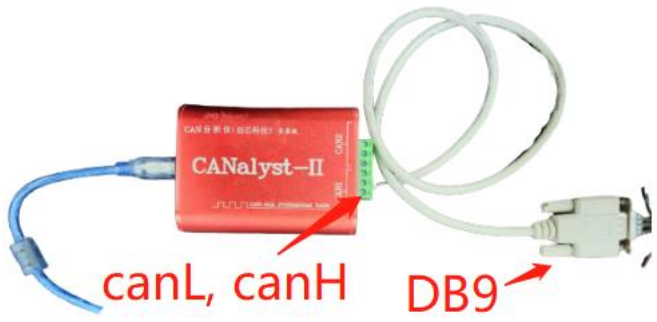
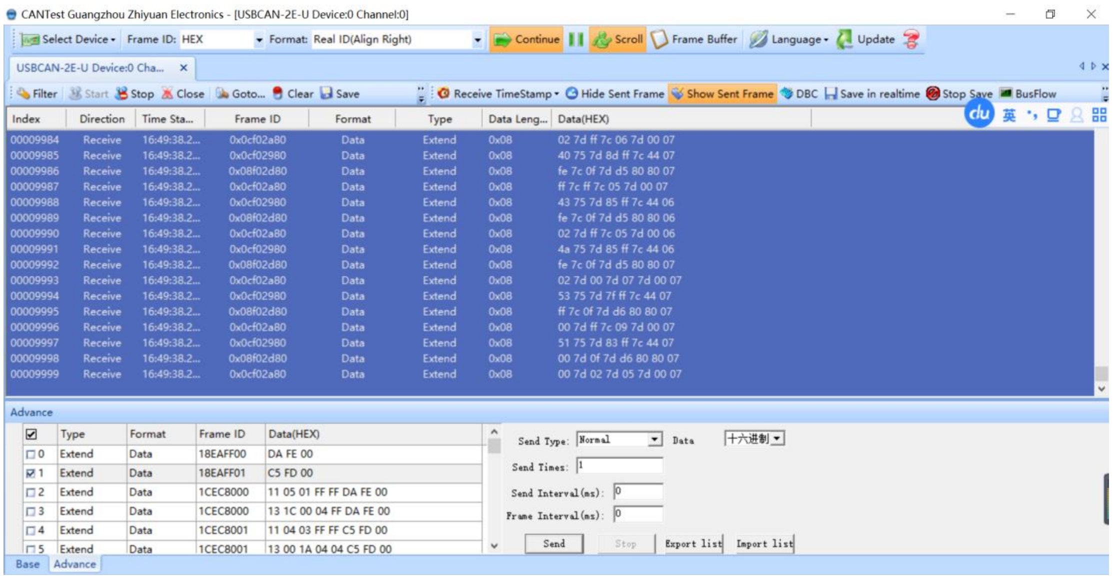


Fig2. CAN connection mode

And then check whether have data output from CAN port, you can use CANtest.exe to do this check:



CANTest Guangzhou Zhiyuan Electronics - [USBCAN-2E-U Device:0 Channel:0]

Frame ID: HEX　Format: Real ID(Align Right)

USBCAN-2E-U Device:0 Cha...

| Index | Direction | Time Sta... | Frame ID | Format | Type | Data Leng... | Data(HEX) |
|---|---|---|---|---|---|---|---|
| 00009984 | Receive | 16:49:38.2... | 0x0cf02a80 | Data | Extend | 0x08 | 02 7d ff 7c 06 7d 00 07 |
| 00009985 | Receive | 16:49:38.2... | 0x0cf02980 | Data | Extend | 0x08 | 40 75 7d 8d ff 7c 44 07 |
| 00009986 | Receive | 16:49:38.2... | 0x08f02d80 | Data | Extend | 0x08 | fe 7c 0f 7d d5 80 80 07 |
| 00009987 | Receive | 16:49:38.2... | 0x0cf02a80 | Data | Extend | 0x08 | ff 7c ff 7c 05 7d 00 07 |
| 00009988 | Receive | 16:49:38.2... | 0x0cf02980 | Data | Extend | 0x08 | 43 75 7d 85 ff 7c 44 06 |
| 00009989 | Receive | 16:49:38.2... | 0x08f02d80 | Data | Extend | 0x08 | fe 7c 0f 7d d5 80 80 06 |
| 00009990 | Receive | 16:49:38.2... | 0x0cf02a80 | Data | Extend | 0x08 | 02 7d ff 7c 05 7d 00 06 |
| 00009991 | Receive | 16:49:38.2... | 0x0cf02980 | Data | Extend | 0x08 | 4a 75 7d 85 ff 7c 44 06 |
| 00009992 | Receive | 16:49:38.2... | 0x08f02d80 | Data | Extend | 0x08 | fe 7c 0f 7d d5 80 80 07 |
| 00009993 | Receive | 16:49:38.2... | 0x0cf02a80 | Data | Extend | 0x08 | 02 7d 00 7d 07 7d 00 07 |
| 00009994 | Receive | 16:49:38.2... | 0x0cf02980 | Data | Extend | 0x08 | 53 75 7d 7f ff 7c 44 07 |
| 00009995 | Receive | 16:49:38.2... | 0x08f02d80 | Data | Extend | 0x08 | ff 7c 0f 7d d6 80 80 07 |
| 00009996 | Receive | 16:49:38.2... | 0x0cf02a80 | Data | Extend | 0x08 | 00 7d ff 7c 09 7d 00 07 |
| 00009997 | Receive | 16:49:38.2... | 0x0cf02980 | Data | Extend | 0x08 | 51 75 7d 83 ff 7c 44 07 |
| 00009998 | Receive | 16:49:38.2... | 0x08f02d80 | Data | Extend | 0x08 | 00 7d 0f 7d d6 80 80 07 |
| 00009999 | Receive | 16:49:38.2... | 0x0cf02a80 | Data | Extend | 0x08 | 00 7d 02 7d 05 7d 00 07 |

Advance

| ☑ | Type | Format | Frame ID | Data(HEX) |
|---|---|---|---|---|
| ☐ 0 | Extend | Data | 18EAFF00 | DA FE 00 |
| ☑ 1 | Extend | Data | 18EAFF01 | C5 FD 00 |
| ☐ 2 | Extend | Data | 1CEC8000 | 11 05 01 FF FF DA FE 00 |
| ☐ 3 | Extend | Data | 1CEC8000 | 13 1C 00 04 FF DA FE 00 |
| ☐ 4 | Extend | Data | 1CEC8001 | 11 04 03 FF FF C5 FD 00 |
| ☐ 5 | Extend | Data | 1CEC8001 | 13 00 1A 04 04 C5 FD 00 |

Send Type: Normal　Data 十六进制
Send Times: 1
Send Interval(ms): 0
Frame Interval(ms): 0

Fig3. CANTest Check

1. Choose the right device: USBCAN-2E-U
2. Choose the right bitrate: 500k
3. Don’t confuse CAN-H & CAN-L

If there is data output, the CAN connection is successful.

# 2. Configuring User Parameters

## 2.1 INS402 User Parameters Configuration

The coordinate system of INS402 is like follow:

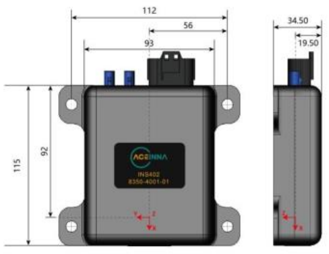


Fig4. INS402 Coordinate System

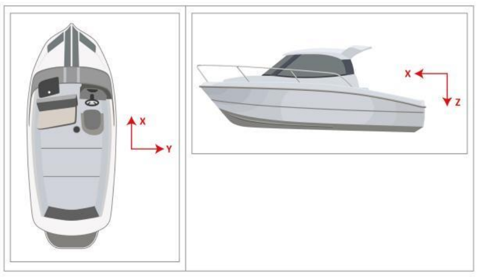


Fig5. Carrier Coordinate System

- Open ins402.json in Acenav.exe, the path is:
  "Acenav.cli\setting\RTK_INS\ins402.json", The "value" in "userParameters" (from line6 to line65) need to be configured.
  Each parameter of "lever arm" is Measured from IMU to Projection point. It is required to have the lever arm accuracy of less than 2 cm.

- "paramId": 1-3 correspond to "gnss lever arm":x, y, z
  Follow pattern show the translation offset in each direction of x, y, z. The IMU to the GNSS master antenna lever arm is [x, y, z] = [2.0, 0.0, -2.0] m.

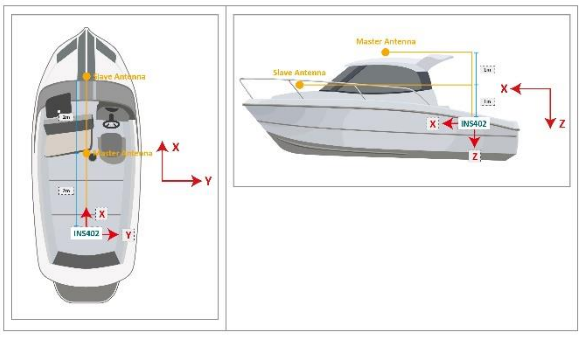
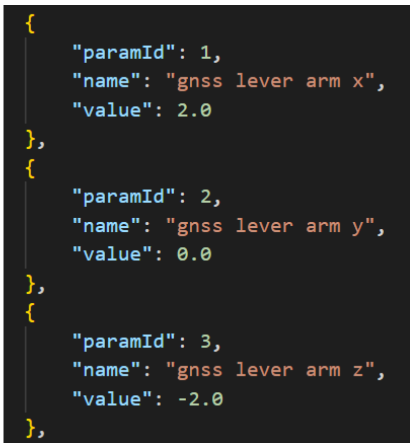


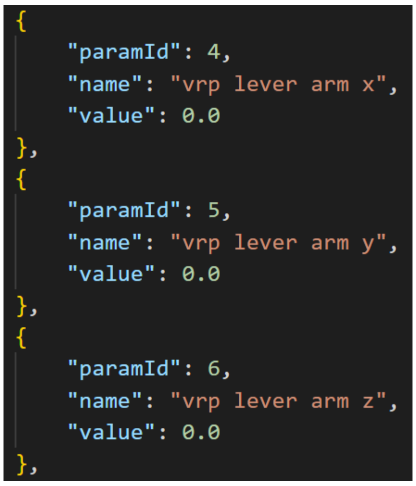
Fig6. GNSS Lever Arms Measure



```
{
    "paramId": 1,
    "name": "gnss lever arm x",
    "value": 2.0
},
{
    "paramId": 2,
    "name": "gnss lever arm y",
    "value": 0.0
},
{
    "paramId": 3,
    "name": "gnss lever arm z",
    "value": -2.0
},
```

Fig7. GNSS Lever Arm Configuration

- "paramId": 4-6 correspond to "user lever arm": x, y, z
  1. If the INS on boat, ignore these three values, Set the initial value: [x, y, z] = [0.0, 0.0, 0.0]
  2. If the INS in car, measure the IMU to rear axle center lever arm and set in the paraID4-6



```
{
    "paramId": 4,
    "name": "vrp lever arm x",
    "value": 0.0
},
{
    "paramId": 5,
    "name": "vrp lever arm y",
    "value": 0.0
},
{
    "paramId": 6,
    "name": "vrp lever arm z",
    "value": 0.0
},
```

Fig8. GNSS Lever Arm Configuration

- “paramId”: 7-9 correspond to “user lever arm”: x, y, z

  If we take the GNSS master antenna as the Projection point. follow pattern show the translation offset in each direction of x, y, z. The IMU to the Projection point lever arm is [x, y, z] = [2.0, 0.0, -2.0] m.

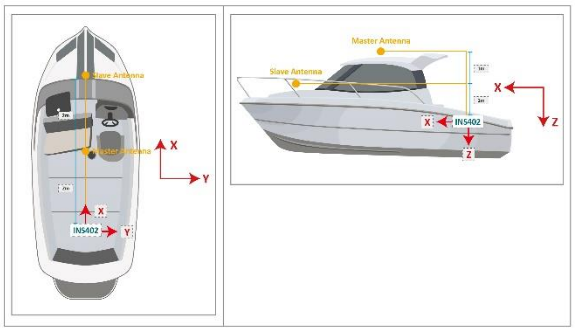
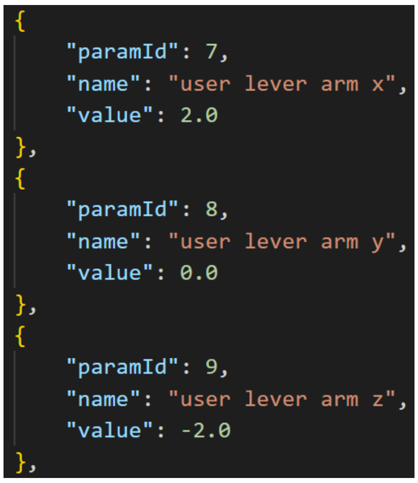


Fig9. User Lever Arms Measure



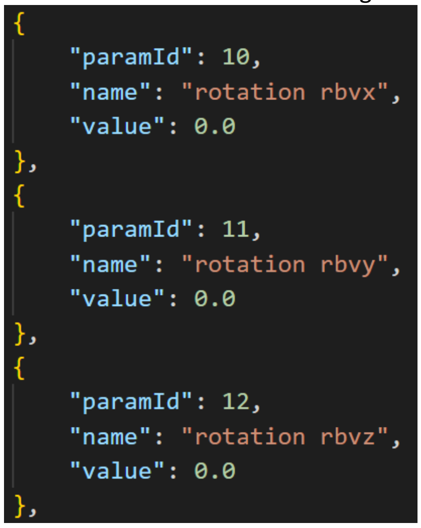
```
{
    "paramId": 7,
    "name": "user lever arm x",
    "value": 2.0
},
{
    "paramId": 8,
    "name": "user lever arm y",
    "value": 0.0
},
{
    "paramId": 9,
    "name": "user lever arm z",
    "value": -2.0
},
```

Fig10. User Lever Arm Configuration

- “paramId”: 10-12 correspond to “rotation align”: x, y, z

  If the coordinate system of INS402 installed on carrier is consistent with the carrier coordinate system, then rotation align [x, y, z] = [0.0, 0.0, 0.0] deg.

  This is the initial value, fill in the corrected value after misalignment calibration. (See2.3)



```
{
    "paramId": 10,
    "name": "rotation rbvx",
    "value": 0.0
},
{
    "paramId": 11,
    "name": "rotation rbvy",
    "value": 0.0
},
{
    "paramId": 12,
    "name": "rotation rbvz",
    "value": 0.0
},
```

Fig11. Rotation Align Configuration

- “paramId”: 13-15 correspond to “sec lever arm”: x, y, z

  Follow pattern show the translation offset in each direction of x, y, z. The IMU to the GNSS slave antenna lever arm is [x, y, z] = [4.0, 0.0, -1.0] m.

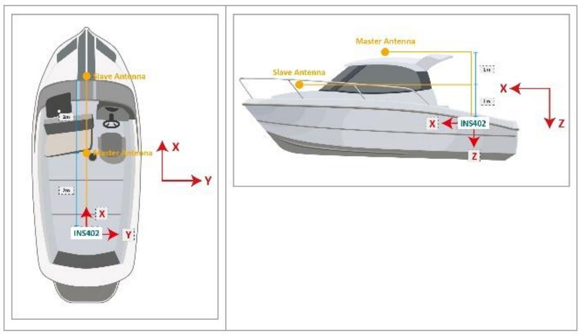
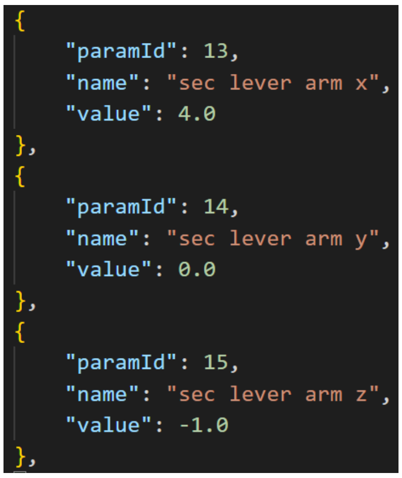


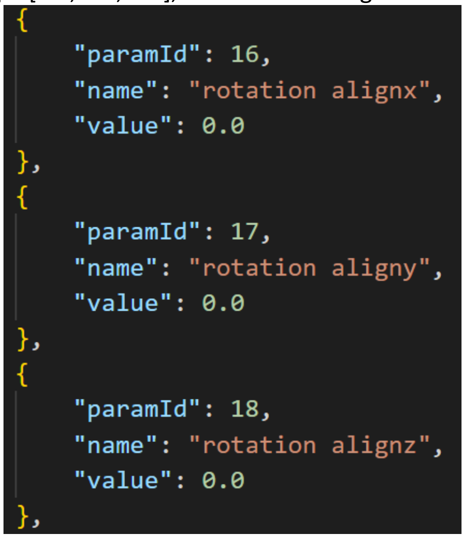
Fig12. User Lever Arms Measure



```
{
    "paramId": 13,
    "name": "sec lever arm x",
    "value": 4.0
},
{
    "paramId": 14,
    "name": "sec lever arm y",
    "value": 0.0
},
{
    "paramId": 15,
    "name": "sec lever arm z",
    "value": -1.0
},
```

Fig13. User Lever Arm Configuration

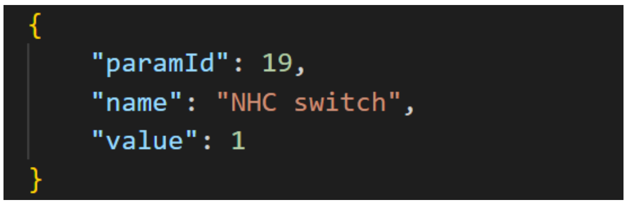
- "paramId": 16-18 correspond to "rotation align": x, y, z
  Set the initial value: [x, y, z] = [0.0, 0.0, 0.0], no need to change



```
{
    "paramId": 16,
    "name": "rotation alignx",
    "value": 0.0
},
{
    "paramId": 17,
    "name": "rotation aligny",
    "value": 0.0
},
{
    "paramId": 18,
    "name": "rotation alignz",
    "value": 0.0
},
```

Fig14. Rotation Align Configuration

- "paramId": 19 correspond to "NHC switch"
  0: Ocean (INS on boat)
  1: Land (INS in car)



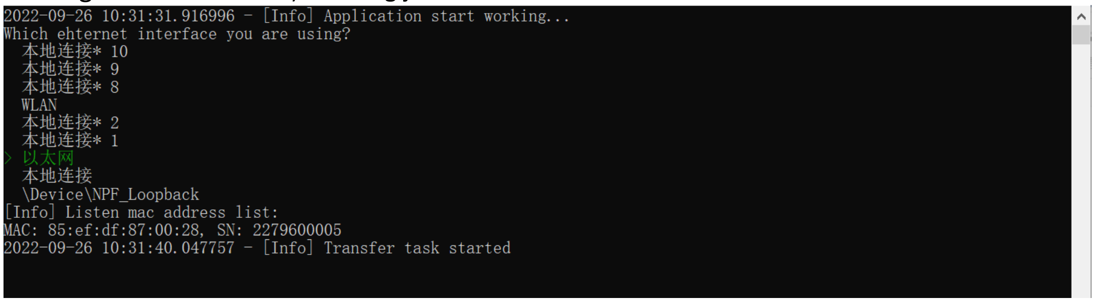
```
{
    "paramId": 19,
    "name": "NHC switch",
    "value": 1
}
```

Fig15. NHC Switch Configuration

## 2.2 CAN Transfer Configuration

CAN-Transfer is a tool to listen odometer message(speed and gear) on canbus, and wrap the message( speed × gear ), then send to ethernet in frequency about 20Hz. If every things work, there is a can_speed.log in data/session_YYYYMMDD_hhmmss folder. It logs all received speed on canbus.

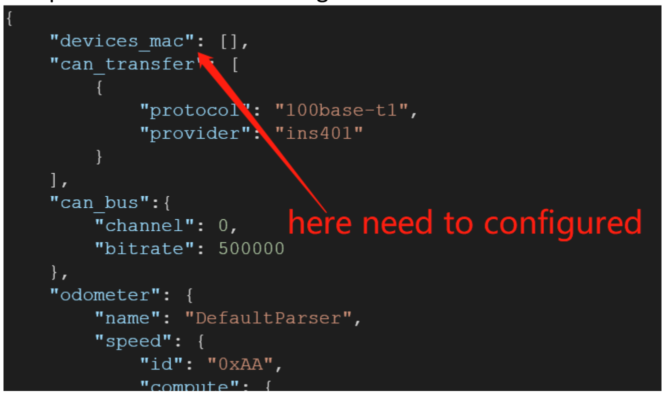
To use CAN-Transfer, you need to configure **"devices_mac"** (a list of INS402, it needs wrapped message from ethernet.) in config.json.



```
2022-09-26 10:31:31.916996 - [Info] Application start working...
Which ehternet interface you are using?
  本地连接* 10
  本地连接* 9
  本地连接* 8
  WLAN
  本地连接* 2
  本地连接* 1
> 以太网
  本地连接
  \Device\NPF_Loopback
[Info] Listen mac address list:
MAC: 85:ef:df:87:00:28, SN: 2279600005
2022-09-26 10:31:40.047757 - [Info] Transfer task started
```

Fig16.Ethernet and device

- Choose your PC Ethernet and connect
- The CAN-transfer can read the device automatically, but for further using, you can write the device mac in the square brackets show in Fig17.



```
{
    "devices_mac": [],
    "can_transfer": [
        {
            "protocol": "100base-t1",
            "provider": "ins401"
        }
    ],
    "can_bus":{
        "channel": 0,
        "bitrate": 500000
    },
    "odometer": {
        "name": "DefaultParser",
        "speed": {
            "id": "0xAA",
            "compute": {
```

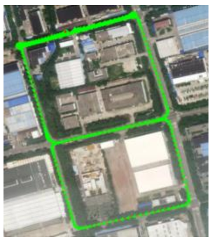
Fig17. CAN-Transfer Configuration

- This config.json shows all the information we need to configure odometer
  "device_mac"
  "can_bus"
  "odometer": the dbc file of the vehicle or boat is needed to configure odometer
- If the customer is confused with this configuration, they can send us the dbc file and we will modify the config.json for them

## 2.3 Misalignment Calibration

**Vehicle Online Calibration Procedure:**

i. Vehicle stops at level road in open sky, signal good environment.
ii. Start device and check device work normal.
iii. Write INS initial configuration parameters, configure RTK correction and start data logging.
iv. Start to run as "日" (including left and right turns and keep straight after turning) after RTK correction is received and INS convergence.



v. Confirm start point, keep run as straight line with > 5m/s speed for at least 250m. Then make a U-turn, run back to start point with > 5m/s speed. Do at least 3 iterations to make sure the desired accuracy reached.

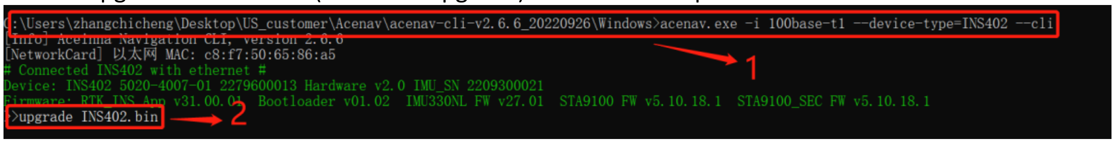
***Notes**

Log tool & tool command:

a) Log tool: Acenav.exe

b) Tool command:

- ./acenav -i 100base-t1 --device-type=INS402 –cli ——first step
  >>upgrade <firmware> (firmware upgrade)——second step

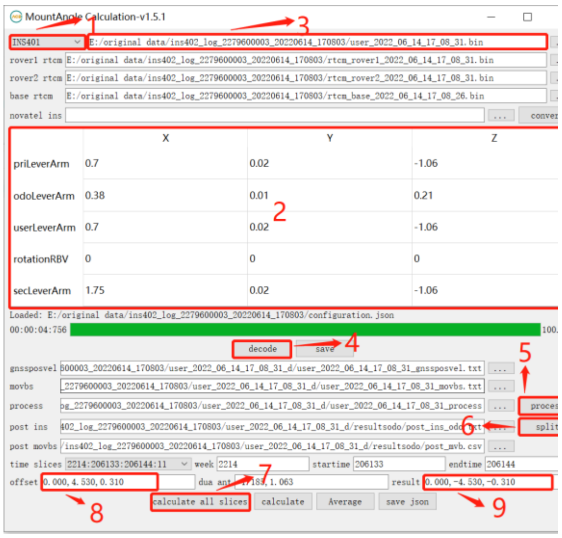


```
C:\Users\zhangchicheng\Desktop\US_customer\Acenav\acenav-cli-v2.6.6_20220926\Windows>acenav.exe -i 100base-t1 --device-type=INS402 --cli
[Info] Aceinna Navigation CLI, version 2.6.6
[NetworkCard] 以太网 MAC: c8:f7:50:65:86:a5
# Connected INS402 with ethernet #
Device: INS402 5020-4007-01 2279600013 Hardware v2.0 IMU_SN 2209300021
Firmware: RTK_INS App v31.00.01  Bootloader v01.02  IMU330NL FW v27.01  STA9100 FW v5.10.18.1  STA9100_SEC FW v5.10.18.1
>upgrade INS402.bin
```

Fig18. Firmware upgrade

- ./acenav -i 100base-t1 --device-type=INS402 (log data)
- ./acenav -i 100base-t1 --device-type=INS402 - s (save the configuration, every time you change the parameters in INS402.json, you need to use this command )
- ./acenav parse –t ins402 - p <file path> (decode data)

**The steps of misalignment calibration on software is:**



MountAngle Calculation-v1.5.1

```
INS401       E:/original data/ins402_log_2279600003_20220614_170803/user_2022_06_14_17_08_31.bin
rover1 rtcm  E:/original data/ins402_log_2279600003_20220614_170803/rtcm_rover1_2022_06_14_17_08_31.bin
rover2 rtcm  E:/original data/ins402_log_2279600003_20220614_170803/rtcm_rover2_2022_06_14_17_08_31.bin
base rtcm    E:/original data/ins402_log_2279600003_20220614_170803/rtcm_base_2022_06_14_17_08_26.bin
novatel ins
```

| | X | Y | Z |
|---|---|---|---|
| priLeverArm | 0.7 | 0.02 | -1.06 |
| odoLeverArm | 0.38 | 0.01 | 0.21 |
| userLeverArm | 0.7 | 0.02 | -1.06 |
| rotationRBV | 0 | 0 | 0 |
| secLeverArm | 1.75 | 0.02 | -1.06 |

```
Loaded: E:/original data/ins402_log_2279600003_20220614_170803/configuration.json
00:00:04:756
gnssposvel   600003_20220614_170803/user_2022_06_14_17_08_31_d/user_2022_06_14_17_08_31_gnssposvel.txt
movbs        _2279600003_20220614_170803/user_2022_06_14_17_08_31_d/user_2022_06_14_17_08_31_movbs.txt
process      og_2279600003_20220614_170803/user_2022_06_14_17_08_31_d/user_2022_06_14_17_08_31_process
post ins     402_log_2279600003_20220614_170803/user_2022_06_14_17_08_31_d/resultsodo/post_ins_odo.txt
post movbs   /ins402_log_2279600003_20220614_170803/user_2022_06_14_17_08_31_d/resultsodo/post_mvb.csv
time slices  2214:206133:206144:11   week 2214   startime 206133   endtime 206144
offset 0.000,4.530,0.310   dua ant 1.185,1.063   result 0.000,-4.530,-0.310
```

Fig19. Misalignment calibration on software

i. Select the device model for testing. ("INS401" can be used for both INS401&402)

ii. Drags the configuration file of the calibrated data into the specified location.

iii. Place the user.bin (logged by Acenav.exe) in the specified location.

iv. Decode the file.

v. Process the decode file

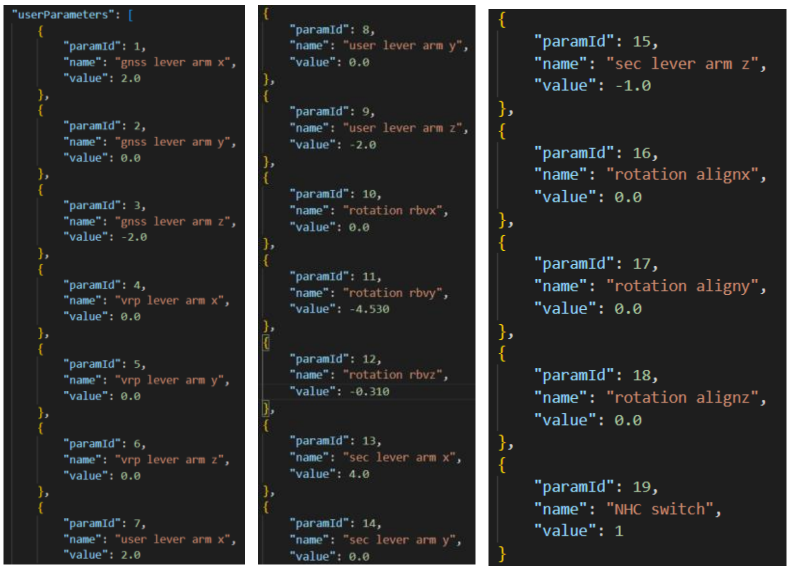
vi. Split a file into multiple pieces of data.

vii. calculate each piece of data separately or a whole piece of data.

viii. The estimated parameter is shown in '8' (Fig. 14).

ix. The parameter value '9' (Fig. 14) of the file that is written to the ins402.json – rotation RBV

**The final configuration**



```
"userParameters": [
    {
        "paramId": 1,
        "name": "gnss lever arm x",
        "value": 2.0
    },
    {
        "paramId": 2,
        "name": "gnss lever arm y",
        "value": 0.0
    },
    {
        "paramId": 3,
        "name": "gnss lever arm z",
        "value": -2.0
    },
    {
        "paramId": 4,
        "name": "vrp lever arm x",
        "value": 0.0
    },
    {
        "paramId": 5,
        "name": "vrp lever arm y",
        "value": 0.0
    },
    {
        "paramId": 6,
        "name": "vrp lever arm z",
        "value": 0.0
    },
    {
        "paramId": 7,
        "name": "user lever arm x",
        "value": 2.0
```

```
{
    "paramId": 8,
    "name": "user lever arm y",
    "value": 0.0
},
{
    "paramId": 9,
    "name": "user lever arm z",
    "value": -2.0
},
{
    "paramId": 10,
    "name": "rotation rbvx",
    "value": 0.0
},
{
    "paramId": 11,
    "name": "rotation rbvy",
    "value": -4.530
},
{
    "paramId": 12,
    "name": "rotation rbvz",
    "value": -0.310
},
{
    "paramId": 13,
    "name": "sec lever arm x",
    "value": 4.0
},
{
    "paramId": 14,
    "name": "sec lever arm y",
    "value": 0.0
```

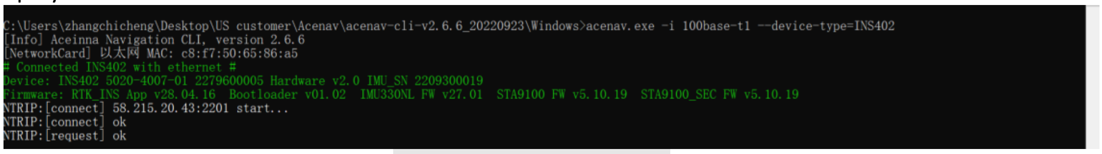
```
{
    "paramId": 15,
    "name": "sec lever arm z",
    "value": -1.0
},
{
    "paramId": 16,
    "name": "rotation alignx",
    "value": 0.0
},
{
    "paramId": 17,
    "name": "rotation aligny",
    "value": 0.0
},
{
    "paramId": 18,
    "name": "rotation alignz",
    "value": 0.0
},
{
    "paramId": 19,
    "name": "NHC switch",
    "value": 1
}
```

Fig.20-22 The Final Configuration

## 3. Data Log & Analysis

### 3.1 Data log

Run AceNav.exe to establish a connection to the INS402, the following information will be displayed on the console.



```
C:\Users\zhangchicheng\Desktop\US customer\Acenav\acenav-cli-v2.6.6_20220923\Windows>acenav.exe -i 100base-t1 --device-type=INS402
[Info] Aceinna Navigation CLI, version 2.6.6
[NetworkCard] 以太网 MAC: c8:f7:50:65:86:a5
# Connected INS402 with ethernet #
Device: INS402 5020-4007-01 2279600005 Hardware v2.0 IMU_SN 2209300019
Firmware: RTK_INS App v28.04.16  Bootloader v01.02  IMU330NL FW v27.01  STA9100 FW v5.10.19  STA9100_SEC FW v5.10.19
NTRIP:[connect] 58.215.20.43:2201 start...
NTRIP:[connect] ok
NTRIP:[request] ok
```

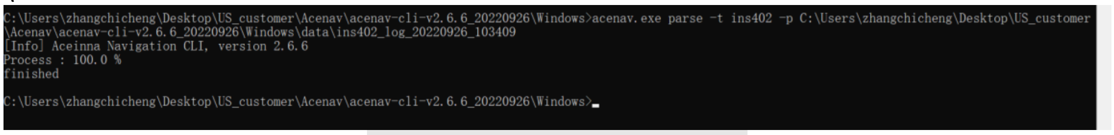
Fig23. Acenav connection

Every time AceNav.exe runs, a subfolder with the time tag will be created under “acenav/data” directory. It will contain a configuration file and four logged binary files:

- configuration.json: INS402 configuration file read from the device flash.
- rtcm_base_<time_tag>.bin: GNSS RTK correction data in RTCM format.
- rtcm_rover_<time_tag>.bin: INS402 GNSS raw data in RTCM format.
- user_<time_tag>.bin: positioning solution, raw IMU and other related variance/status information shown in #6.
- ins_save_<time_tag>.bin: save last status of INS402 when power was off.

### 3.2 Data Analysis

Run the data parsing command below to decode the user_<time_tag>.bin in ‘Acenav.cli \data’



```
C:\Users\zhangchicheng\Desktop\US_customer\Acenav\acenav-cli-v2.6.6_20220926\Windows>acenav.exe parse -t ins402 -p C:\Users\zhangchicheng\Desktop\US_customer
\Acenav\acenav-cli-v2.6.6_20220926\Windows\data\ins402_log_20220926_103409
[Info] Aceinna Navigation CLI, version 2.6.6
Process : 100.0 %
finished

C:\Users\zhangchicheng\Desktop\US_customer\Acenav\acenav-cli-v2.6.6_20220926\Windows>
```

Fig24. Acenav Data Parsing Process

The decoded data from the user_<time_tag>.bin will contain the following files:

- user_<time_tag>_dm.csv: INS402 system diagnostic messages
- user_<time_tag>_gnss.csv: GNSS solution at 1 Hz
- user_<time_tag>_imu.csv: raw IMU data at 100 Hz
- user_<time_tag>_ins.csv: INS solution at 100 Hz
- user_<time_tag>_gnss.kml: GNSS solution trajectory in kml format
- user_<time_tag>_ins.kml: INS solution trajectory in kml format
- user_<time_tag>_nmea.txt: GNSS solution in NMEA 0183 ASCII
- user_<time_tag>_odo.txt: vehicle odometer speed data received by INS402