10. Self-diagnostic function

10.1. Self-diagnostic function

10.1.1. Self-diagnostic function summary

The BT-2000 is equipped with a self-diagnostic function to check the status of internal devices. You can check the following devices and content using the self-diagnostic function.

Table 10-1 Confirmation items using the self-diagnostic function

Device type	Display name	Diagnostic method
CPU	CPU	Error check
Power control IC	PMIC	Check power supply status
Main memory (RAM)	DDR	Check RAM data
Data memory (Flash)	eMMC	Reading and writing to data region
Wireless LAN module	WLAN	Check communication with device
Bluetooth 3.0	Bluetooth	Check communication with device
Bluetooth Low Energy	BLE	Check communication with device
Audio IC	Audio IC	Reading device information characteristics
Audio commands IC	Voice Command	Reading device information characteristics
Sensor	Sensor MCU	Communication check
IMU sensor	IMU	Acquire sensor data
Sensor Illumination	Sensor Illumination	Acquire sensor data
Charger IC A	Charger IC A	Check status
Charger IC B	Charger IC B	Check status
Battery A	Battery A	Check status
Battery B	Battery B	Check status

The self-diagnostic function is a basic diagnostic tool. Be aware that it may not be able to locate faults in individual functions in the device. And some functions are not available.

10.1.2. Using the self-diagnostic function

The self-diagnostic function is provided as an app. Start "SelfDiagnosis" from the app list.

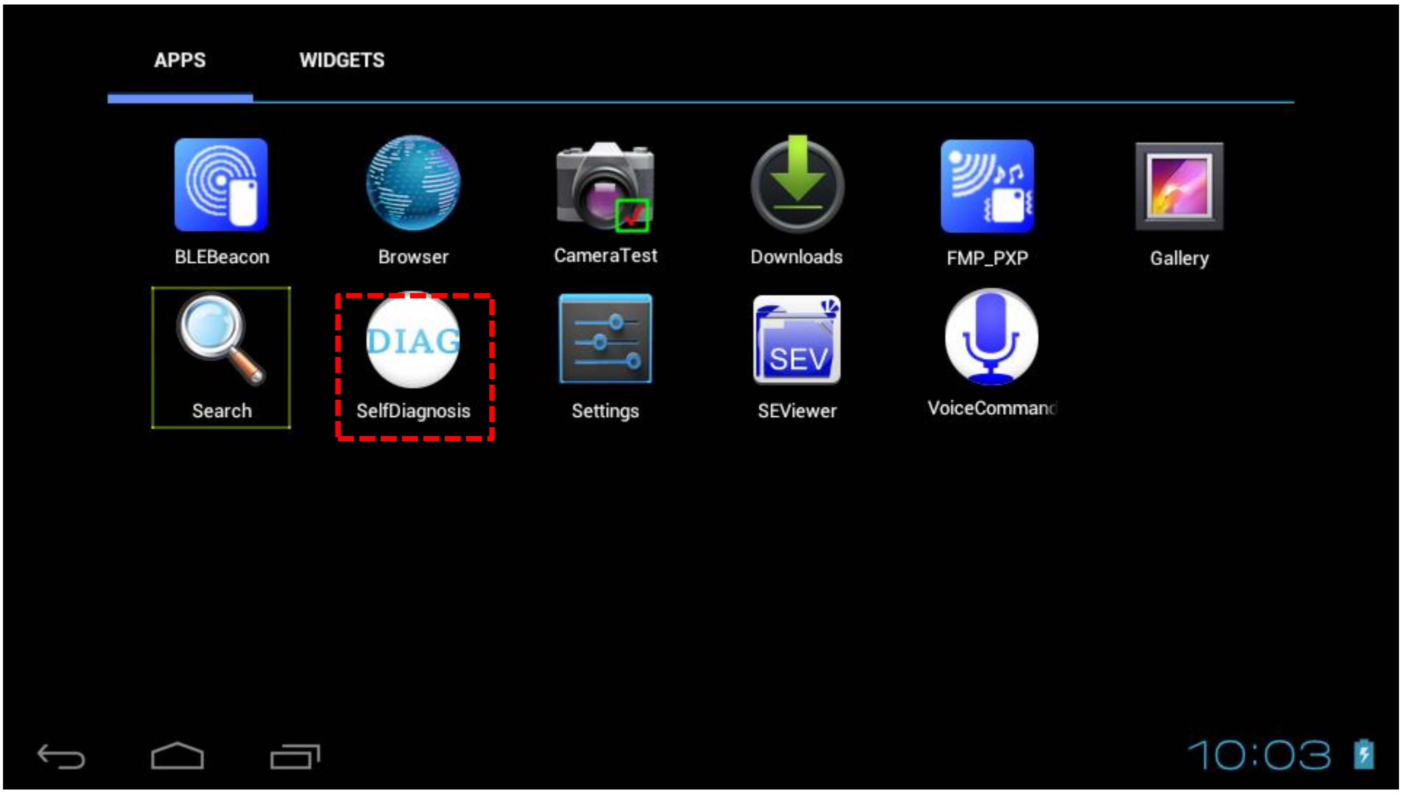
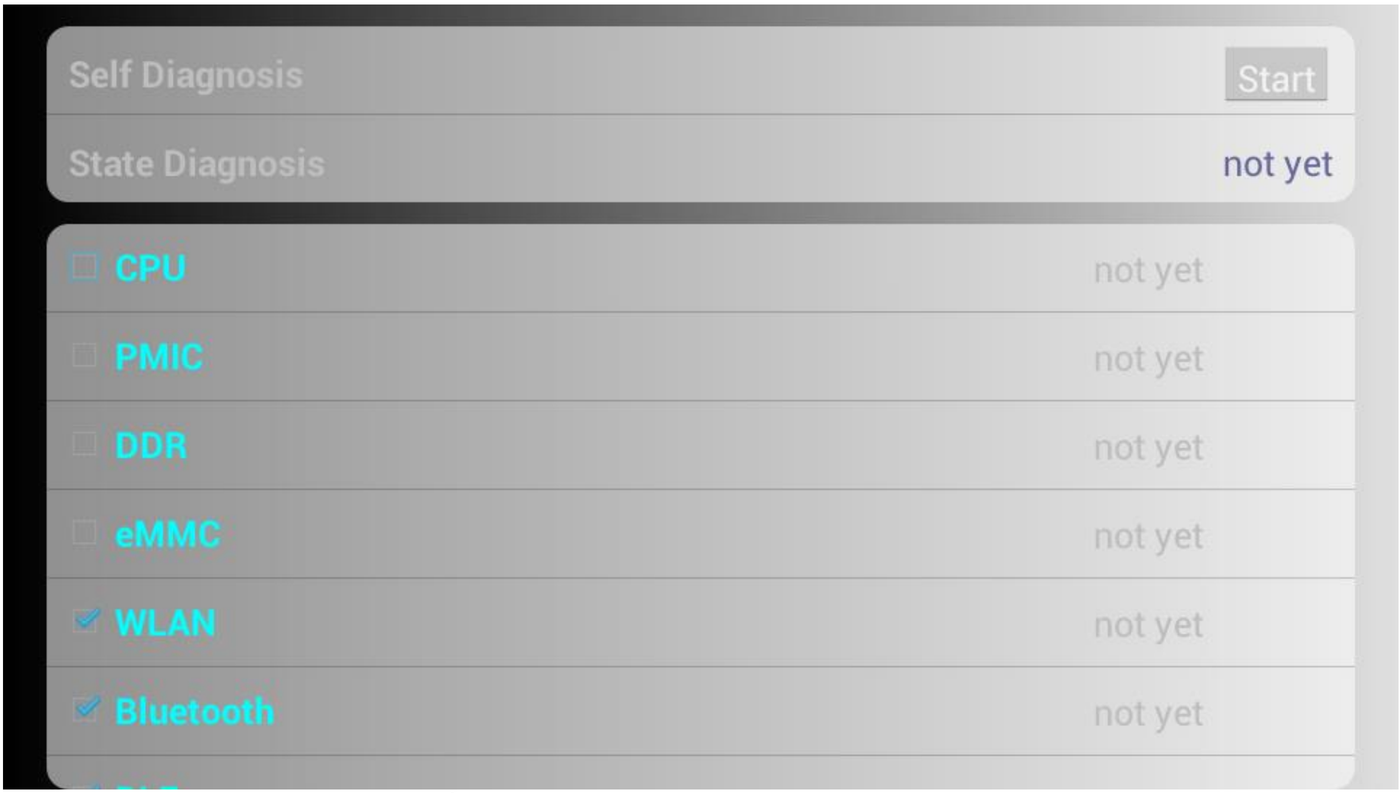


Figure 10-1 Start SelfDiagnosis app

When the app starts, click the "Start" button at the top of the screen to start the diagnostic procedure.



Self Diagnosis Start

State Diagnosis not yet

□ CPU not yet

□ PMIC not yet

□ DDR not yet

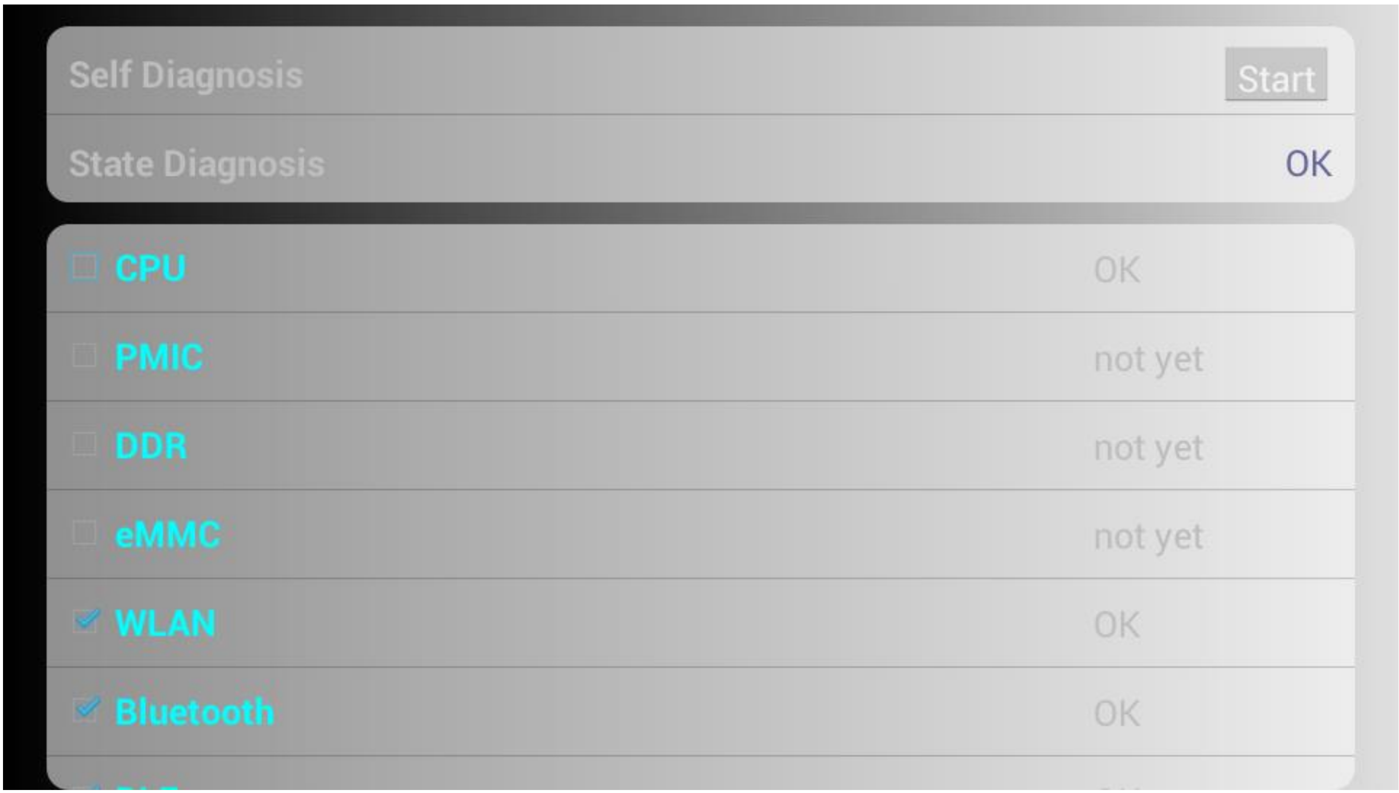
□ eMMC not yet

☑ WLAN not yet

☑ Bluetooth not yet

Figure 10-2 Start SelfDiagnosis app screen

When the diagnosis has been performed, the results for each device are displayed. See table 10-1 above for details about the target items and the checking method.



State Diagnosis OK

□	CPU	OK
□	PMIC	not yet
□	DDR	not yet
□	eMMC	not yet
☑	WLAN	OK
☑	Bluetooth	OK

Figure 10-3　SelfDiagnosis check results

11.　GPS Assist

11.1. GPS Assist function summary

BT-2000 has GPS assist function to reduce initial positioning time (Time To First Fix: TTFF) of cold start when it takes long GPS positioning time. By downloading GPS satellite orbit prediction data in advance and write in the BT-2000, it will reduce download time of satellite data and reduce TTFF as a result.

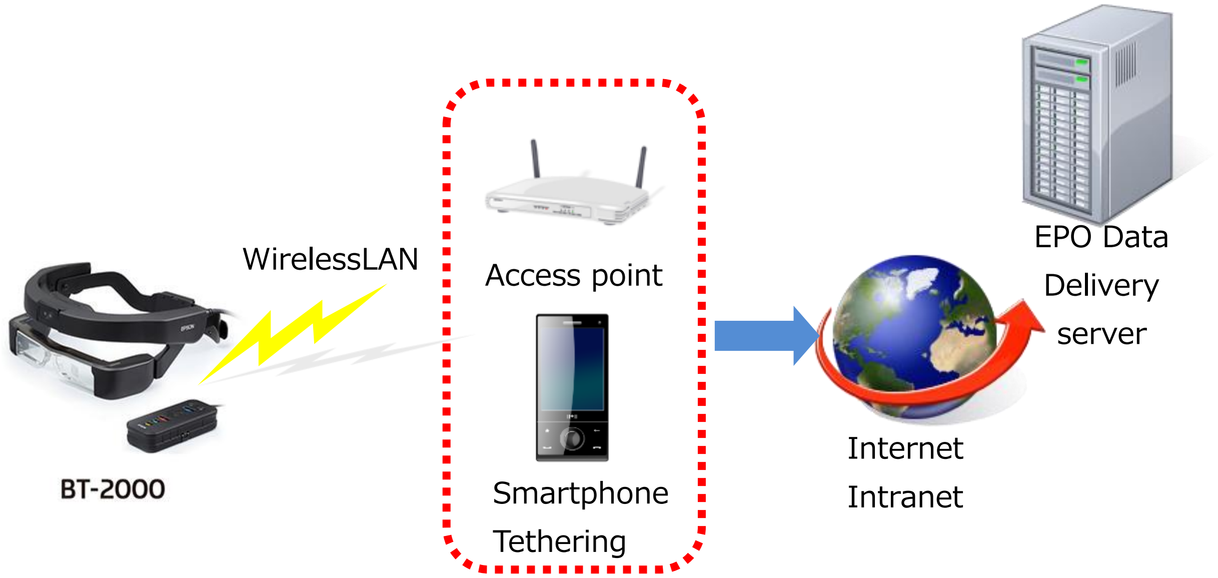


Figure 11-1　EPO Data download image

11.2. Precaution and limitation of using /developing GPS Assist function

■GPS Assist function Effect

To fully demonstrate GPS Assist function effect, it is required to capture 4 GPS satellites.

There are some cases that cannot fully demonstrate the effect under condition that cannot capture or hard to capture the GPS satellites.

■GPS cold start

To start GPS positioning, there are hot start using satellite data of last time, warm start and cold start when you cannot use positioning data of last time.

GPS assist function is a function to demonstrate effect in cold start. It does not have effect to reduce TTFF in hot start and warm start.

■Setting time

It uses time information to simulate GPS satellite position. So it is necessary to set the system time accurately.

■Downloading EPO data

EPO data validity period is 30 days. But as the GPS assist function effect is higher by using new data, so please re-download the EPO data by 1 week as a guide.

■EPO data delivery server

BT-2000 can download EPO data by internet from EPSON EPO data delivery server.

When you cannot use internet or using GPS assist function by local internet, copy EPO data from EPSON data delivery server and place at local internet.

To set EPO data delivery server in BT-2000, write delivery server URL at ”internal storage¥agps¥agps.conf”.

■GPS assist function API

BT-2000 GPS assist function API is using EPSON own API.

To develop an App using GPS assist function, install SDK provided by EPSON and use EPSON own API.

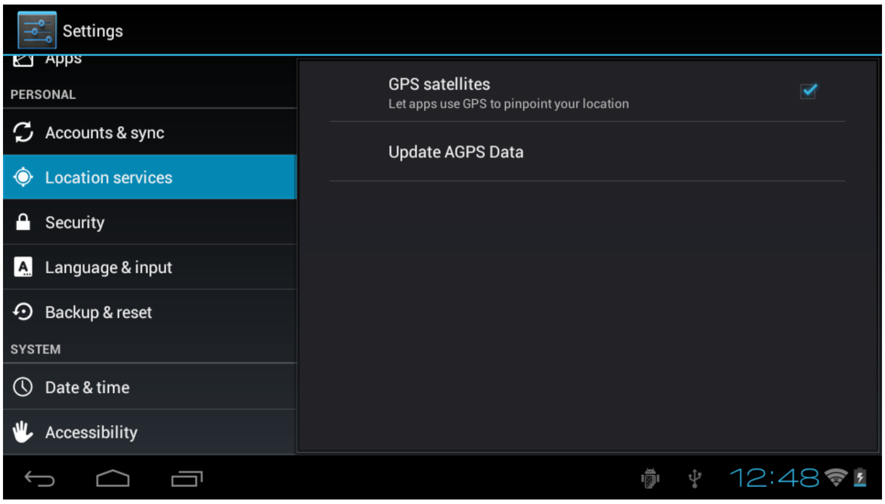
11.3. How to use

This section describes how to use GPS assist function.

11.3.1.Downloading EPO data

- Turn Wi-Fi function ON to access to EPO delivery server.
  Internet environment is required to access to EPSON EPO data delivery server.
  When placing EPO data to intranet, connection to intranet is required.

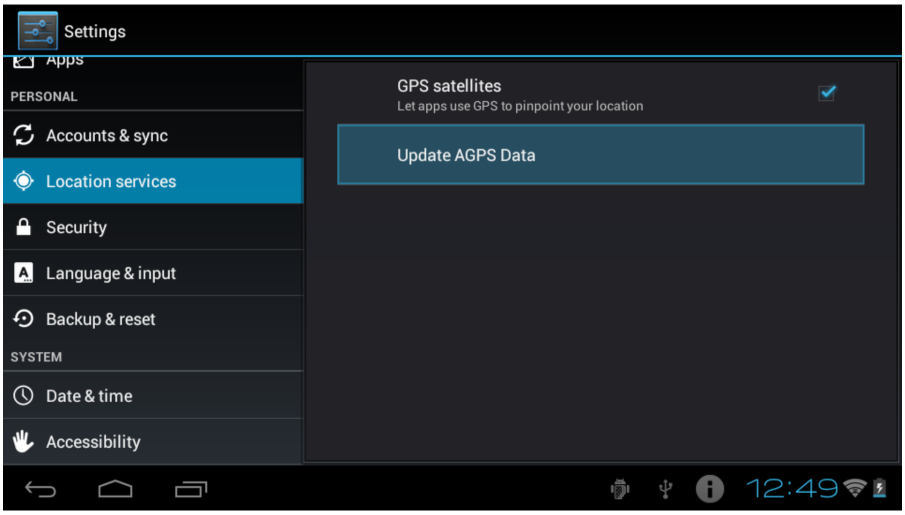
- Open [Setting] App and select [Position information service].



GPS satellites ☑
Let apps use GPS to pinpoint your location

Update AGPS Data

- Turn [GPS function] ON and select [AGPS data update].



GPS satellites ☑
Let apps use GPS to pinpoint your location

Update AGPS Data

11.3.2.Writing EPO data

While writing EPO data to BT-2000, the icon ⓘ will be displayed in system bar.
Writing is completed after the iconⓘ disappears.

11.3.3.Using EPO data

- Start App using GPS. GPS assist functions automatically operate at cold start.

11.4. Setting method

This section describes about setting for writing EPO data.

Setting file will be located at “internal storage¥agps¥agps.conf”.
Locate created file according to below list.

11.4.1.Setting file agps.conf

File path: /mnt/sdcard/agps/agps.conf
Default: None It is generated with the default value when [Settings]-[AGPS data update] is executed.
File format: Conform to notation of java.util.Properies
Use ”¥:” to write colon

Table 11-1 Setting items in agps.conf

No	Item	Explanation
1	EPO_URL	Set URL of EPO data delivery server. Default value is” https://tech.moverio.epson.com/epo/data.bin”
2	EPO_NUM_SEGS	Set EPO writing segment number（1～28）. 1 segment =6 hours. Default value is 28(6 hours ×28=7 days)
3	LAST_MOD	Shows download time of last time. Not need for setting.

11.5. Application interface function summary

This section describes API of EPO data download and writing to BT-2000.

1) Download EPO data from EPO data delivery server
2) Write downloaded EPO data to BT-2000 GPS module

To use each API, import below module.

com.epson.moverio.bt2000.agpsctl

Table 11-2 Interface list of AGPSControl class

NO.	Function name	Function summary	Note
1	downloadEpo	Download EPO data by setting according to agps.conf setting file.	EPSON original API
2	downloadEpo_url	Download EPO data from URL specified by argument	EPSON original API
3	writeEpo	Write EPO data to BT-2000 by setting according to agps.conf setting file.	EPSON original API
4	writeEpo_segs	Write EPO data of specified segment by argument to BT-2000	EPSON original API

11.6. Application interface detail

11.6.1.downloadEpo

- Function

Download EPO data according to setting written in Setting file (/sdcatd/agps.conf).

EPO data will be delivered from EPSON data delivery server（Update every day at 0 :10 UTC)

- Format

```
int downloadEpo(void);
```

- Parameter

None

- Return value

Always 0

- Broadcast Intent

Intent	Status
AGPS_DOWNLOAD_COMPLETED	download completed
AGPS_DOWNLOAD_ERROR	download error
AGPS_CONF_ERROR	agps.conf error（agps.conf not exist, or defect in EPO_URL）
AGPS_NETWORK_ERROR	Error　by Wi-Fi OFF
AGPS_FILE_NOT_UPDATED	No update in server file

11.6.2.downloadEpo_url

- Function

Download EPO data from the URL of parameter).

- Format

```
int downloadEpo(String arg0);
```

- Parameter

Type name	Explanation
String arg0	URL of EPO data delibery server

- Return value

Always 0

- Broadcast Intent

Same as downloadEpo

11.6.2.1. writeEpo

- Function

  Write EPO data to BT-2000 GPS module by agps.conf setting.

- Format

```
int writeEPpo();
```

- Parameter

None

- Return value

  Always 0

- Broadcast Intent

Intent	Status
AGPS_WRITE_COMPLETED	write the data completed
AGPS_WRITE_ERROR	Write the data error
AGPS_CONF_ERROR	agps.conf error (No agps.conf, or the number of EPO_NUM_DAYS is not crrect)
AGPS_FILE_EXPIRED	EPO file is old. (Expired at the present time, there is no data for the specified period.)

11.6.2.2. writeEPO_segs

- Function

Writes EPO data for the number of segments specified by the argument to the GPS module of BT-2000.

One segment means the 6 hours.

- Format

```
int writeEPO_segs(int arg0);
```

- Parameter

Type name	Explanation
int arg0	Writing segment number

- R

- Broadcast Intent

  Same as writeEPO

11.7. How to use API for GPS assist

11.7.1. Download of EPO data

Create the instance of AGPSControl and call API for download.

■Implementation example for download

```
public void execDownload(View view) {
  Log.d(TAG, "exec Download");

  try {
    AGPSControl ctl = new AGPSControl();
    ctl.downloadEpo();
  } catch (Exception ex) {
    ex.printStackTrace();
  }
}
```

11.7.2. Intent reception method

As download and writing process are done asynchronously, return value cannot be referred directly.

So, receive the process result by using Broadcast Intent.

For each Broadcast Intent value issued by each API, refer to the item of Broadcast intent of each API.

This section describes about the intent reception method.

11.7.2.1. At start

Set intent filter and receiver at the start of application.

■Implementation example

```
private MyReceiver receiver = new MyReceiver();
private IntentFilter filter = new IntentFilter();

protected void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  setContentView(R.layout.activity_main);
  filter.addAction(AGPSControl.AGPS_CONF_ERROR);
  filter.addAction(AGPSControl.AGPS_DOWNLOAD_COMPLETED);
  filter.addAction(AGPSControl.AGPS_DOWNLOAD_ERROR);
  filter.addAction(AGPSControl.AGPS_FILE_EXPIRED);
  filter.addAction(AGPSControl.AGPS_FILE_NOT_UPDATED);
  filter.addAction(AGPSControl.AGPS_NETWORK_ERROR);
  filter.addAction(AGPSControl.AGPS_WRITE_COMPLETED);
  filter.addAction(AGPSControl.AGPS_WRITE_ERROR);

  getApplicationContext().registerReceiver(receiver, filter);
}
```

■Intent receiver example

```
class MyReceiver extends BroadcastReceiver {
  public MyReceiver() {
    super();
  }
  @Override
  public void onReceive(Context context, Intent intent) {
    if (intent.getAction().equals(AGPSControl.AGPS_CONF_ERROR)) {
      Toast.makeText(getApplicationContext(), "AGPS_CONF_ERROR",
Toast.LENGTH_LONG).show();
      Log.d(TAG, "Intent Received | AGPS Conf Error");
    }
    if (intent.getAction().equals(AGPSControl.AGPS_DOWNLOAD_COMPLETED)) {
      Toast.makeText(getApplicationContext(), "AGPS_DOWNLOAD_COMPLETED",
Toast.LENGTH_LONG).show();
      Log.d(TAG, "Intent Received | AGPS Download Completed");
    }
      ～～～～～(Omit other intent)～～～～～
  }
}
```

11.7.2.2. Re-setting of receiver due to application situation transition

Cancel intent receiver when application situation transit like becoming background.

■Receiver cancellation example

```
  @Override
  protected void onPause() {
    super.onPause();
      getApplicationContext().unregisterReceiver(receiver);
  }
```

When returning to application again, re-register the intent receiver.

■Re-register example

```
  @Override
  protected void onResume() {
    super.onResume();
      getApplicationContext().registerReceiver(receiver, filter);
  }
```

― Memo ―