PSDK API 概览

说明：

Gdu 为支持开发者开发出可挂载在 GDU 无人机上的负载设备，提供了开发工具包 Payload SDK（即 PSDK）以及开发配件，方便开发者利用 GDU 无人机上如电源、通讯链路及状态信息等**资源**，开发出可挂载在 GDU 无人机上的负载设备。开发者能够根据行业的应用需求，基于 PSDK 提供的功能接口，结合具体的结构设计、硬件设计、软件逻辑实现和算法优化，开发出如**自动巡检系统**、**红外相机**、**测绘相机**、**多光谱相机**、**喊话器**、**探照灯**等满足不同细分领域的负载设备。

主要特性

- 使用 C 语言开发
- 支持主流的嵌入式系统，如 Linux 和 RTOS
- 模块化的设计思路，易于跨平台移植

Gdu Core

PSDK Core 相关功能的头文件为 GDU_core.h ，本文档描述了 GDU_core.h 文件中结

构体和函数原型的关键信息和使用方法。

目录

函数原型

function GduCore_Init

功能：初始化 Payload SDK 内核。	product:all

以阻塞模式初始化 Payload SDK 内核。

说明：

- 这个接口的调用位置需要特别注意，需要在 console/OSAL handler 函数/HAL handler 函数注册完成后调用。 同时，必须在调用其他功能模块接口开始时进行初始化。 您需要正确填写开发者信息，以确保初始化成功。 如需更多说明，请参阅教程

“PSDK 初始化”。

- 该函数在获得正确的飞行器类型和 PSDK 适配器类型之前不会返回。 该逻辑保证飞行器和 PSDK 适配器在 PSDK 功能模块和用户程序运行前已经正常启动。 该函数的一般执行时间为 2~4 秒。

T_GduReturnCode GduCore_Init(const T_GduUserInfo *userInfo);

参数

userInfo： 指向 PSDK 应用程序信息的指针。

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

function GduCore_SetAlias

功能：为 GDU 应用或产品设置一个满足条件的别名。	product:all

别名将在 GDU Pilot 中显示（如果存在）。

说明：

- 仍然需要将从 GDU SDK 开发者网站获取的正确的 GDU APP 名称传递给 GduCore_Init() 接口。 GDU APP 名称将用

于绑定或验证。

- 别名会在一段时间后生效，最大值为 1s。

T_GduReturnCode GduCore_SetAlias(const char *productAlias);

参数

productAlias： 指向产品别名字符串的指针，别名以 '\0' 结尾。 字符串的最大长度为 31。如果别名字符串的长度大于 31，别名字符串将被截断并传入。

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode#

function GduCore_ApplicationStart

功能：通知 Payload SDK 核心应用程序启动。	product:all

说明： 该接口的调用位置需要特别注意，需要在所有模块初始化和注册接口后完成调用。

T_GduReturnCode GduCore_ApplicationStart(void);

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

GDU 版本信息

PSDK 获取版本信息相关功能的头文件为 GDU_version.h，本文档描述了 GDU_version.h 文件中的宏定义。

宏定义

- GDU SDK 主版本号

GDU SDK 主版本号，当有不兼容的 API 更改时。 范围从 0 到 99。

```
#define GDU_VERSION_MAJOR        3
```

- GDU SDK 次版本号

GDU SDK 次版本号，当以向后兼容的方式添加功能时会发生变化。 范围从 0 到 99。

```
#define GDU_VERSION_MINOR        1
```

- GDU SDK 修改版本号

GDU SDK 修改版本号，当有向后兼容的错误修复更改时。 范围从 0 到 99。

```
#define GDU_VERSION_MODIFY       0
```

- GDU SDK beta 版本信息

GDU SDK beta 版本信息，发布版本为 0，当 beta 版本发布变化时。 范围从 0 到

255。

#define GDU_VERSION_BETA 0

- GDU SDK 构建版本号，范围为 0～65535

GDU SDK 版本构建信息，当 jenkins 触发构建更改时。 范围从 0 到 65535。

#define GDU_VERSION_BUILD 1503

GDU 错误码

错误码的头文件为

GDU_error.h

，本文档描述了错误码相关的宏定义和结构体的关键信息以及使用方法。

宏定义

- 错误模块索引值偏移量

#define GDU_ERROR_MODULE_INDEX_OFFSET 32u

- 错误模块索引值掩码

#define GDU_ERROR_MODULE_INDEX_MASK 0x000000FF00000000u

- 原始错误码偏移量

```
#define GDU_ERROR_RAW_CODE_OFFSET           0u
```

- 原始错误码掩码

```
#define GDU_ERROR_RAW_CODE_MASK             0x00000000FFFFFFFFu
```

- 错误码生成宏

```
#define GDU_ERROR_CODE(moduleIndex, rawErrCode) \
((uint64_t)0 | \
((((uint64_t)(moduleIndex)) << (GDU_ERROR_MODULE_INDEX_OFFSET)) & (GDU_ERROR_MODULE_INDEX_MASK)) | \
((((uint64_t)(rawErrCode)) << (GDU_ERROR_RAW_CODE_OFFSET)) & (GDU_ERROR_RAW_CODE_MASK)))
```

- 错误对象

```
#define GDU_ERROR_OBJECTS \
/* 系统模块错误信息 */ \
{GDU_ERROR_SYSTEM_MODULE_CODE_SUCCESS, "Execution successfully.", NULL}, \
{GDU_ERROR_SYSTEM_MODULE_CODE_INVALID_REQUEST_PARAMETER, "Request parameters are invalid.", "Please double-check requested parameters."}, \
{GDU_ERROR_SYSTEM_MODULE_CODE_EXECUTING_HIGHER_PRIORITY_TASK, "A higher priority task is being executed.", "Please stop the higher priority task or try again later."}, \
{GDU_ERROR_SYSTEM_MODULE_CODE_NONSUPPORT, "Operation is not supported.", "Please check input parameters or contact GDU for help."}, \
{GDU_ERROR_SYSTEM_MODULE_CODE_TIMEOUT, "Execution timeout.", "Please contact GDU for help."}, \
{GDU_ERROR_SYSTEM_MODULE_CODE_MEMORY_ALLOC_FAILED, "Memory allocation failed.", "Please check system configure."}, \
{GDU_ERROR_SYSTEM_MODULE_CODE_INVALID_PARAMETER, "Input parameters are invalid.", "Please double-check requested parameters."}, \
{GDU_ERROR_SYSTEM_MODULE_CODE_NONSUPPORT_IN_CURRENT_STATE, "Operation is not supported in current state.", "Please try again later."}, \
{GDU_ERROR_SYSTEM_MODULE_CODE_SYSTEM_ERROR, "System error.", "Please contact GDU for help."}, \
{GDU_ERROR_SYSTEM_MODULE_CODE_HARDWARE_ERR, "Hardware error.", "Please contact GDU for help."}, \
{GDU_ERROR_SYSTEM_MODULE_CODE_INSUFFICIENT_ELECTRICITY, "Low battery.", "Please replacement battery for machine and try again."}, \
```

```
{GDU_ERROR_SYSTEM_MODULE_CODE_UNKNOWN, "Unknown error.", NULL}, \
{GDU_ERROR_SYSTEM_MODULE_CODE_NOT_FOUND, "Parameters are not found.", NULL}, \
{GDU_ERROR_SYSTEM_MODULE_CODE_OUT_OF_RANGE, "Out of range.", "Please check parameters."}, \
{GDU_ERROR_SYSTEM_MODULE_CODE_BUSY, "System is busy.", "Please try again later."}, \
{GDU_ERROR_SYSTEM_MODULE_CODE_DUPLICATE, "Have existed the same object.", "Please input valid parameters."}, \
{GDU_ERROR_SYSTEM_MODULE_CODE_ADAPTER_NOT_MATCH, "PSDK adapter do not meet requirements.", "Please try again after replacing PSDK adapter."}, \
 \
/* 云台模块错误信息 */ \
{GDU_ERROR_GIMBAL_MODULE_CODE_PITCH_REACH_POSITIVE_LIMIT, "Pitch axis gimbal reach positive limit.", "Please do not rotate towards positive direction."}, \
{GDU_ERROR_GIMBAL_MODULE_CODE_PITCH_REACH_NEGATIVE_LIMIT, "Pitch axis gimbal reach negative limit.", "Please do not rotate towards negative direction."}, \
{GDU_ERROR_GIMBAL_MODULE_CODE_ROLL_REACH_POSITIVE_LIMIT, "Roll axis gimbal reach positive limit.", "Please do not rotate towards positive direction."}, \
{GDU_ERROR_GIMBAL_MODULE_CODE_ROLL_REACH_NEGATIVE_LIMIT, "Roll axis gimbal reach negative limit.", "Please do not rotate towards negative direction."}, \
{GDU_ERROR_GIMBAL_MODULE_CODE_YAW_REACH_POSITIVE_LIMIT, "Yaw axis gimbal reach positive limit.", "Please do not rotate towards positive direction."}, \
{GDU_ERROR_GIMBAL_MODULE_CODE_YAW_REACH_NEGATIVE_LIMIT, "Yaw axis gimbal reach negative limit.", "Please do not rotate towards negative direction."}, \
{GDU_ERROR_GIMBAL_MODULE_CODE_NON_CONTROL_AUTHORITY, "Current device do not have control authority of the gimbal.", "Please do not control gimbal with other devices that have high control priority simultaneously."}, \
 \
/* 负载协同模块错误提示 */ \
{GDU_ERROR_PAYLOAD_COLLABORATION_MODULE_CODE_POSITION_NOT_MATCH, "Payload mount position do not meet requirements.", "Please read API and user documentation, ensuring input parameters satisfy requirements."}, \
\
/* 身份模块的激活错误消息 */ \
{GDU_ERROR_IDENTITY_MODULE_CODE_ACTIVATE_PARAMETER_ERROR, "Activation parameter error.", "Please check the validity of the activation parameters."}, \
{GDU_ERROR_IDENTITY_MODULE_CODE_ACTIVATE_ENCODE_ERROR, "Activation encode error", "Please check the validity of the activation parameters"}, \
{GDU_ERROR_IDENTITY_MODULE_CODE_ACTIVATE_NEW_DEVICE_ERROR, "Activate a new device error", \
"\r\n* Double-check your app_id and app_key. Does it match with your GDU developer account?\r\n"\
"* If this is a new device or has been previously activated with another app_id and app_key, you need to activate it through the GDU Assistant 2 with Internet.\r\n"\
"* Please make sure you download the correct version of GDU Assistant 2 for your drone.\r\n"\
}, \
{GDU_ERROR_IDENTITY_MODULE_CODE_ACTIVATE_SOFTWARE_NOT_CONNECTED, "Not connection to GDU Assistant 2.", "Please connect your drone to GDU Assistant 2 while activating the new device at first
```

```
time."}, \
{GDU_ERROR_IDENTITY_MODULE_CODE_ACTIVATE_NETWORK_ERROR, "GDU Assistant 2 not connected to the internet.", "Please activate through the GDU Assistant 2 with Internet"}, \
{GDU_ERROR_IDENTITY_MODULE_CODE_ACTIVATE_SERVER_ACCESS_REFUSED, "GDU server refuse this activation request.", "Please check the validity of the activation parameters."}, \
{GDU_ERROR_IDENTITY_MODULE_CODE_ACTIVATE_ACCESS_LEVEL_ERROR, "Activate level error.", "Please check your app_id level."}, \
{GDU_ERROR_IDENTITY_MODULE_CODE_ACTIVATE_OSDK_VERSION_ERROR, "Activate a wrong osdk version", "Please check your OSDK version whether match the drone or not."}, \
\
/* 数据订阅模块错误信息 */ \
{GDU_ERROR_SUBSCRIPTION_MODULE_CODE_INVALID_TOPIC_FREQ, "Frequency of topic is invalid.", "Please read API and user documentation, ensuring input parameters satisfy requirements."}, \
{GDU_ERROR_SUBSCRIPTION_MODULE_CODE_TOPIC_DUPLICATE, "Topic is subscribed repeatedly.", "Please do not subscribe a topic repeatedly."}, \
{GDU_ERROR_SUBSCRIPTION_MODULE_CODE_TOPIC_NOT_SUBSCRIBED, "Requested topic have not been subscribed.", "Please try to get value after subscribing topic."}, \
{GDU_ERROR_SUBSCRIPTION_MODULE_CODE_TIMESTAMP_NOT_ENABLE, "Requested topic do not have timestamp data. Did not enable timestamp when subscribe topic.", "Please subscribe topic enabled timestamp."},\
{GDU_ERROR_SUBSCRIPTION_MODULE_CODE_TOPIC_NOT_SUPPORTED, "Requested topic is not supported.", "Please check the topic is supported or not on current aircraft version."}, \
\
/* SDK 互联互通模块错误信息 */ \
{GDU_ERROR_MOP_CHANNEL_MODULE_CODE_CONNECTION_CLOSE, "Connection of channel is closed. The peer channel do not work or abnormally be closed.", "Please confirm state of the peer channel and reaccept the connection request of MSDK/OSDK"}, \
\
/* 飞行控制模块错误信息 */ \
{GDU_ERROR_FC_MODULE_CODE_RC_MODE_ERROR, "RC_MODE_ERROR", "Please check the RC mode"}, \
{GDU_ERROR_FC_MODULE_CODE_RELEASE_CONTROL_SUCCESS, "RELEASE_CONTROL_SUCCESS", NULL} ,\
{GDU_ERROR_FC_MODULE_CODE_OBTAIN_CONTROL_SUCCESS, "OBTAIN_CONTROL_SUCCESS", NULL} ,\
{GDU_ERROR_FC_MODULE_CODE_OBTAIN_CONTROL_IN_PROGRESS, "OBTAIN_CONTROL_IN_PROGRESS", NULL} ,\
{GDU_ERROR_FC_MODULE_CODE_RELEASE_CONTROL_IN_PROGRESS, "RELEASE_CONTROL_IN_PROGRESS", NULL} ,\
{GDU_ERROR_FC_MODULE_CODE_RC_NEED_MODE_P, "RC_NEED_MODE_P", NULL} ,\
{GDU_ERROR_FC_MODULE_CODE_RC_NEED_MODE_F, "RC_NEED_MODE_F", NULL} ,\
{GDU_ERROR_FC_MODULE_CODE_PARAM_READ_WRITE_INVALID_PARAM, "read or write invalid param", "Please check hash value and param value"} ,\
{GDU_ERROR_FC_MODULE_CODE_IOC_OBTAIN_CONTROL_ERROR, "IOC_OBTAIN_CONTROL_ERROR", NULL} ,\
{GDU_ERROR_FC_MODULE_CODE_KEY_ERROR, "Activate key error", NULL} ,\
```

```
{GDU_ERROR_FC_MODULE_CODE_NO_AUTHORIZATION_ERROR, "No authorization", "Please finish activation firstly"} ,\
{GDU_ERROR_FC_MODULE_CODE_NO_RIGHTS_ERROR, "No rights error", NULL} ,\
{GDU_ERROR_FC_MODULE_CODE_SYSTEM_ERROR, "Unknown system error", NULL} ,\
\
/* 飞控动作模块错误信息 */\
{GDU_ERROR_FC_JOYSTICK_MODULE_OBTAIN_RELEASE_JOYSTICK_AUTH_SUCCESS, "Obtain/Release joystick authority success", NULL} ,\
{GDU_ERROR_FC_JOYSTICK_MODULE_DEVICE_NOT_ALLOW, "The requesting device is not allowed to obtain/release joystick control authority, only support OSDK/MSDK", "Please use right devices(OSDK/MSDK)"} ,\
{GDU_ERROR_FC_JOYSTICK_MODULE_TAKING_OFF, "Not allowed to obtain/release joystick control authority when drone is taking off ", "Please do it before or after taking off"} ,\
{GDU_ERROR_FC_JOYSTICK_MODULE_LANDING, "Not allowed to obtain/release joystick control authority when drone is landing", "Please do it before or after landing"} ,\
{GDU_ERROR_FC_JOYSTICK_MODULE_CMD_INVALID, "Invalid input command", "Please check your input command which only support 0/1"} ,\
{GDU_ERROR_FC_JOYSTICK_MODULE_RC_NOT_P_MODE, "Not allowed to obtain/release joystick control authority when rc is not in P_MODE", "Please switch RC to P_MODE"} ,\
{GDU_ERROR_FC_JOYSTICK_MODULE_CMD_LENGTH_ERROR, "Invalid length of input command", "Please input valid length command"} ,\
{GDU_ERROR_FC_JOYSTICK_MODULE_HAS_NO_JOYSTICK_AUTHORITY, "Not allowed to release joystick control authority when drone has no joystick authority", "Please obtain joystick authority first"} ,\
{GDU_ERROR_FC_JOYSTICK_MODULE_IN_RC_LOST_ACTION, "Not allowed to obtain/release joystick control authority when drone is executing rc lost action", "Please change battery"} ,\
\
/* 飞控动作模块错误信息 */\
{GDU_ERROR_FC_ACTION_MODULE_TRIGGER_ERROR_MOTOR_ON, "now motor is on", "Please check motor status"} ,\
{GDU_ERROR_FC_ACTION_MODULE_TRIGGER_ERROR_MOTOR_OFF, "now motor is off", "Please check motor status"} ,\
{GDU_ERROR_FC_ACTION_MODULE_TRIGGER_ERROR_IN_AIR, "aircraft is in air", "Please check aircraft flight status"} ,\
{GDU_ERROR_FC_ACTION_MODULE_TRIGGER_ERROR_NOT_IN_AIR, "aircraft is not in air", "Please check aircraft flight status"} ,\
{GDU_ERROR_FC_ACTION_MODULE_TRIGGER_ERROR_HOME_POINT_NOT_SET, "home point not set", "Please set home point"} ,\
{GDU_ERROR_FC_ACTION_MODULE_TRIGGER_ERROR_BAD_GPS, "bad GPS", "Please fly where the GPS signal is good"} ,\
{GDU_ERROR_FC_ACTION_MODULE_TRIGGER_ERROR_IN_SIMULATION, "in simulation", "Please exit simulation first"} ,\
{GDU_ERROR_FC_ACTION_MODULE_TRIGGER_ERROR_CANNOT_START_MOTOR, "can not start motor", "Please check motor status"} ,\
{GDU_ERROR_FC_ACTION_MODULE_TRIGGER_ERROR_LOW_VOLTAGE, "low voltage", "Please change battery"} ,\
```

{GDU_ERROR_FC_ACTION_MODULE_TRIGGER_ERROR_SPEED_TOO_LARGE, "speed too large", "Please slow down"} ,\
\
/* 飞控限飞模块错误信息*/\
{GDU_ERROR_FC_ARREST_FLYING_MODULE_REGISTER_LOGOUT_SET_SUCCESS, "register/logout/execute arrest-flying success", NULL} ,\
{GDU_ERROR_FC_ARREST_FLYING_MODULE_REGISTER_LOGOUT_NULL_POINTER, "null pointer when register/logout arrest-flying", NULL} ,\
{GDU_ERROR_FC_ARREST_FLYING_MODULE_HMS_CODE_HAS_REGISTED, "duplicate hms code", NULL} ,\
{GDU_ERROR_FC_ARREST_FLYING_MODULE_HMS_CODE_NOT_FIND, "invalid hms code", NULL} ,\
{GDU_ERROR_FC_ARREST_FLYING_MODULE_ADD_ITEM_NO_DECRIPTION, "no description when register arrest-flying", NULL} ,\
{GDU_ERROR_FC_ARREST_FLYING_MODULE_REGISTER_ID_INVALID, "invalid request id", NULL} ,\
{GDU_ERROR_FC_ARREST_FLYING_MODULE_STOP_IN_AIR_HMSCODE_NOT_IN_WHITE_TABLE, "hms code is not in white list for allowing stop motor in the air", NULL} ,\
{GDU_ERROR_FC_ARREST_FLYING_MODULE_INVALID_FORMAT_HMSCODE, "invalid hms code format", NULL} ,\
{GDU_ERROR_FC_ARREST_FLYING_MODULE_HMSCODE_NOT_IN_WHITE_TABLE, "hms code is not in arrest-flying white list",NULL} ,\
\
/* 飞控返航位置模块错误信息 */\
{GDU_ERROR_FC_HOME_LOCATION_MODULE_UNKNOWN_FAILED_REASON, "set home location fail, unknown reason", NULL} ,\
{GDU_ERROR_FC_HOME_LOCATION_MODULE_INVALID_GPS_COORDINATE, "invalid GPS coordinate when set APP or RC to be home location", NULL} ,\
{GDU_ERROR_FC_HOME_LOCATION_MODULE_NOT_BE_RECORD, "home location is not inited", "Please waiting for aircraft recording home location"} ,\
{GDU_ERROR_FC_HOME_LOCATION_MODULE_GPS_NOT_READY, "GPS level < 4", NULL} ,\
{GDU_ERROR_FC_HOME_LOCATION_MODULE_DIS_TOO_FAR, "new home location is more than 20km away from current home locaton(APP/RC)", NULL} ,\
\
/* 飞控急停电机模块错误信息 */\
{GDU_ERROR_FC_EMERGENCY_STOP_MOTOR_MODULE_VERSION_NOT_MATCH, "emergency stop motor version not match", NULL} ,\
{GDU_ERROR_FC_EMERGENCY_STOP_MOTOR_MODULE_CMD_INVALID, "emergency stop motor cmd invalid", NULL} ,\
/* 相机管理模块错误信息 */\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_UNSUPPORTED_COMMAND, "Command not supported", "Check the firmware or command validity"} ,\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_TIMEOUT, "Camera's execution of this action has timed out", "Try again or check the firmware or command"} ,\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_RAM_ALLOCATION_FAILED, "Camera's execution of this action is out of memory", "Please contact <dev@GDU.com> for help."} ,\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_INVALID_COMMAND_PARAMETER, "Camera received invalid parameters", "Check the validity of the parameter"} ,\

{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_UNSUPPORTED_COMMAND_IN_CUR_STATE, "Camera is busy or the command is not supported in the Camera's current state", "Check current camera state is if appropriate fot the CMD"} ,\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_CAMERA_TIME_NOT_SYNCHRONIZED, "The time stamp of the camera is not sync", "Please contact <dev@GDU.com> for help."} ,\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_PARAMETER_SET_FAILED, "Camera failed to set the parameters it received", "Please check the parameter to set is if supported in your devices."} ,\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_PARAMETER_GET_FAILED, "Camera param get failed", "Please check the parameter to get is if supported in your devices."} ,\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_SD_CARD_MISSING, "Camera has no SD Card", "Please install SD card."} ,\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_SD_CARD_FULL, "The Camera's SD Card is full", "Please make sure the SD card has enough space."} ,\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_SD_CARD_ERROR, "Error accessing the SD Card", "Please check the validity of the SD card."} ,\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_SENSOR_ERROR, "Camera sensor error", "Please contact <dev@GDU.com> for help."} ,\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_SYSTEM_ERROR, "Camera system error", "Please recheck all the running conditions or contact <dev@GDU.com> for help."} ,\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_PARAMETER_TOTAL_TOO_LONG, "Camera param get failed", "Please check the validity of the parameter length"} ,\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_MODULE_INACTIVATED, "Camera module is not activated", "Please activate the module first."} ,\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_FIRMWARE_DATA_NUM_DISCONTINUOUS, "The seq number of Firmware data is invalid", "Please check the validity of the camera firmware."} ,\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_FIRMWARE_VERIFICATION_ERROR, "Firmware check error", "Please check the validity of the camera firmware."} ,\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_FLASH_WRITE_ERROR, "Camera flash write error", "Please contact <dev@GDU.com> for help."} ,\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_FIRMWARE_TYPE_MISMATCH, "Firmware type is invalid", "Please check the validity of the camera firmware."} ,\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_REMOTE_CONTROL_UNCONNECTED, "Remote Control is disconnected now", "Please check the connection with remote control is if OK."} ,\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_HARDWARE_ERROR, "Camera hardware error", "Please contact <dev@GDU.com> for help."} ,\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_AIRCRAFT_UNCONNECTED, "Disconnect with aircraft", "Please check the connection with aircraft is if OK."} ,\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_CANNOT_UPGRADE_IN_CUR_STATE, "Camera cannot not upgrade in current status", "Please contact <dev@GDU.com> for help."} ,\
{GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_UNDEFINE_ERROR, "Undefined error", "Please contact <dev@GDU.com> for help."} ,\
\
/* 云台管理模块错误信息 */\
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_UNSUPPORTED_COMMAND, "Command not supported", "Check the firmware or command validity"} ,\

{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_TIMEOUT, "Camera's execution of this action has timed out", "Try again or check the firmware or command"} ,\
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_RAM_ALLOCATION_FAILED, "Camera's execution of this action is out of memory", "Please contact <dev@GDU.com> for help."} ,\
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_INVALID_COMMAND_PARAMETER, "Camera received invalid parameters", "Check the validity of the parameter"} ,\
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_UNSUPPORTED_COMMAND_IN_CUR_STATE,
"Camera is busy or the command is not supported in the Camera's current state", "Check current camera state is if appropriate fot the CMD"} ,\
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_CAMERA_TIME_NOT_SYNCHRONIZED, "The time stamp of the camera is not sync", "Please contact <dev@GDU.com> for help."} ,\
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_PARAMETER_SET_FAILED, "Camera failed to set the parameters it received", "Please check the parameter to set is if supported in your devices."} ,\
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_PARAMETER_GET_FAILED, "Camera param get failed", "Please check the parameter to get is if supported in your devices."} ,\
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_SD_CARD_MISSING, "Camera has no SD Card", "Please install SD card."} ,\
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_SD_CARD_FULL, "The Camera's SD Card is full", "Please make sure the SD card has enough space."} ,\
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_SD_CARD_ERROR, "Error accessing the SD Card", "Please check the validity of the SD card."} ,\
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_SENSOR_ERROR, "Camera sensor error", "Please contact <dev@GDU.com> for help."} ,\
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_SYSTEM_ERROR, "Camera system error", "Please recheck all the running conditions or contact <dev@GDU.com> for help."} ,\
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_PARAMETER_TOTAL_TOO_LONG, "Camera param get failed", "Please check the validity of the parameter length"} ,\
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_MODULE_INACTIVATED, "Camera module is not activated", "Please activate the module first."} ,\
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_FIRMWARE_DATA_NUM_DISCONTINUOUS, "The seq number of Firmware data is invalid", "Please check the validity of the camera firmware."} ,\
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_FIRMWARE_VERIFICATION_ERROR, "Firmware check error", "Please check the validity of the camera firmware."} ,\
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_FLASH_WRITE_ERROR, "Camera flash write error", "Please contact <dev@GDU.com> for help."} ,\
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_FIRMWARE_TYPE_MISMATCH, "Firmware type is invalid", "Please check the validity of the camera firmware."} ,\
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_REMOTE_CONTROL_UNCONNECTED, "Remote Control is disconnected now", "Please check the connection with remote control is if OK."} ,\
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_HARDWARE_ERROR, "Camera hardware error", "Please contact <dev@GDU.com> for help."} ,\
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_AIRCRAFT_UNCONNECTED, "Disconnect with aircraft", "Please check the connection with aircraft is if OK."} ,\
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_CANNOT_UPGRADE_IN_CUR_STATE, "Camera cannot not upgrade in current status", "Please contact <dev@GDU.com> for help."} ,\

```
{GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_UNDEFINE_ERROR, "Undefined error", "Please contact <dev@GDU.com> for help."} ,\
\
/* 航线 v2 模块错误信息 */    \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_COMMON_SUCCESS, "Execute waypoint v2 cmd successfully", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_COMMON_INVALID_DATA_LENGTH, "The length of the data is illegal based on the protocol ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_COMMON_INVALD_FLOAT_NUM, "Invalid float number (NAN or INF) ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_WP_VERSION_NO_MATCH, "Waypoint mission version can't match with firmware ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_COMMON_UNKNOWN, "Fatal error  Unexpected result ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_RESV, "Reserved", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INIT_WP_NUM_TOO_MANY,
"Min_initial_waypoint_num is large than permitted_max_waypoint_num ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INIT_WP_NUM_TOO_FEW,
"Min_initial_waypoint_num is less than permitted_min_waypoint_num ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INIT_INVALID_END_INDEX, "Waypoint_end_index is equal or large than total_waypoint_num ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOAD_START_ID_GT_END_ID, "The start index is greater than end index of upload wps ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOAD_END_ID_GT_TOTAL_NUM, "The end index of uplod wps is greater than inited total numbers ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_DOWNLOAD_WPS_NOT_IN_STORED_RAGNE, "The index of first and end waypoint expected to download is not in range of stored in FC ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_CUR_POS_IS_FAR_AWAY_FROM_FIRST_WP, "Current position is far away from the first waypoint. ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_ADJ_WPS_TOO_CLOSE, "It is too close from two adjacent waypoints",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_ADJ_WPS_TOO_FAR, "The distance betwween two adjacent waypoints is not in[0.5m, 5000m]",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOAD_MAX_VEL_GT_GLOBAL, "The max vel of uplod wp is greater than global max vel ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOAD_LOCAL_CRUISE_VEL_GT_LOCAL_MAX,
"The local cruise vel of upload wp is greater than local max vel ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOAD_LOCAL_CRUISE_VEL_GT_GLOBAL_MAX,
"The local cruise vel of upload wp is greater than global max vel ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INIT_INVALID_GLOBAL_MAX_VEL, "Global_max_vel is greater than permitted_max_vel or less than permitted_min_vel ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_GLOBAL_CRUISE_VEL_GT_MAX_VEL,
"Global_cruise_vel is greater than global_max_vel ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INIT_INVALID_GOTO_FIRST_FLAG,
"Goto_first_point_mode is out of range of waypoint_goto_first_flag_t_enum ",NULL}, \
```

{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INIT_INVALID_FINISHED_ACTION, "Finished_action is out of range of wp_plan_finish_action_t_enum ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INIT_INVALID_RC_LOST_ACTION, "Rc_lost_action is out of range of wp_plan_rc_lost_action_t_enum ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOAD_YAW_MODE_INVALID, "The yaw mode of upload wp is invalid. reference to waypoint2_yaw_mode_t defined in math_waypoint_planner.h ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOAD_YAW_CMD_NOT_IN_RANGE, "The yaw command of upload wp is not in range. the range for MR:[-180 180]",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOAD_YAW_TURN_DIRECTION_INVALID, "The yaw turn direction of upload wp is invalid. it should be 0:clockwise or 1:anti-clockwise ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOAD_WP_TYPE_INVALID, "The wp type of upload wp is invalid. reference to waypoint_type_t defined in math_waypoint_planner.h ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_GO_STOP_CMD_INVALID, "Go/stop command is invalid. ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INVALID_PAUSE_RECOVERY_CMD, "The command of pause/recovery is not equal to any of the command enum ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INVALID_BREAK_RESTORE_CMD, "The command of break/restore is not equal to any of the command enum ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INIT_INVALID_REF_POINT, "Initial reference point position coordinate exceed set range ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_DAMPING_DIS_GE_DIS_OF_ADJ_POINTS, "The damping dis is greater than or equal the distance of adjacent point ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOAD_CANNT_SET_WP_LINE_EXIT_TYPE, "Cann't set wp_line_exit type to wp ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INIT_INFO_NOT_UPLOADED, "The init info of Ground Station is not uploaded yet ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_WP_HAS_NOT_UPLOADED, "The wp has not uploaded yet ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOADED_WP_NOT_ENOUGH,
"Min_initial_waypoint_num is not uploaded. ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_GS_HAS_STARTED, "Waypoint plan has started when received go command. ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_GS_NOT_RUNNING, "Waypoint plan not running when received stop command. ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_GS_NOT_RUNNING_FOR_PAUSE_RECOVERY,
"Ground station(GS) is not started(used by pause/recovery) ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_GS_NOT_RUNNING_FOR_BREAK_RESTORE, "Ground station(GS) is not started(used by break/restore) ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_NOT_IN_WP_MIS, "Not in the waypoint mission(MIS)(cannot pause/recovery or break/restore) ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_MIS_HAS_BEEN_PAUSED, "The current status is paused",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_MIS_NOT_PAUSED, "Not in paused status",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_MIS_HAS_BEEN_BROKEN, "The current status is broken",NULL}, \

{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_MIS_NOT_BROKEN, "Not in break status",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_PAUSE_RECOVERY_NOT_SUPPORTED, "The configuration forbid using pause/recovery API ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_BREAK_RESTORE_NOT_SUPPORTED, "The configuration forbid using break/restore API ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_NO_BREAK_POINT, "No break point is recorded for restore ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_NO_CUR_TRAJ_PROJECT, "No current trajectory project point is recorded for restore ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_NO_NXT_TRAJ_PROJECT, "No next trajectory project point is recorded for restore ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_NO_NNT_TRAJ_PROJECT, "No next the next trajectory project point is recorded for restore ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOAD_WP_ID_NOT_CONTINUE, "The index of upload wp is not continue after the store wp ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_WP_LINE_ENTER_NOT_SET_TO_START_WP, "The WP_LINE_ENTER wp_type set to a wp which is not the init start waypoint ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INIT_WP_WHEN_PLAN_HAS_STARTED, "The waypoint plan has started when initializing waypoint ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_DAMPING_DIS_EXCEED_RANGE, "Waypoint damping distance exceed set range ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_WAYPOINT_COOR_EXCEED_RANGE, "Waypoint position coordinate exceed rational range ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_FIRST_WP_TYPE_IS_WP_TURN_NO, "First waypoint type error",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_WP_EXCEED_RADIUS_LIMIT, "Waypoint position exceed radius limit ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_WP_EXCEED_HEIGHT_LIMIT, "Waypoint position exceed height limit ",NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_STATUS_RESV, "Reserved error code", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_STATUS_WP_MIS_CHECK_FAIL, "Head_node is null or atti_not_healthy or gyro_not_healthy or horiz_vel_not healthy or horiz_abs_pos_not_healthy. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_STATUS_HOME_NOT_RECORDED, "The home point is no recorded yet which will be executed at the first time of GPS level > 3(MR FW). ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_STATUS_LOW_LOCATION_ACCURACY, "Current location accuracy is low for bad GPS signal. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_STATUS_RTK_CONDITION_IS_NOT_READY, "Use rtk_data but rtk is not connected or rtk_data is invalid ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_SECURE_RESV, "Reserved error code", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_SECURE_CROSS_NFZ, "The trajectory cross the NFZ ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_SECURE_BAT_LOW, "Current capacity of smart battery or voltage of non-smart battery is lower than level 1 or level 2 threshold ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTION_COMMON_RESV, "Reserved error code", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTION_COMMON_ACTION_ID_DUPLICATED, "The ID of

Action is duplicated. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTION_COMMON_ACTION_ITEMS_SPACE_NOT_ENOUGH, "There is no enough memory space for new Action Item ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTION_COMMON_ACTION_SIZE_GT_BUF_SIZE, "The size of buffer used to get the info of Action is less than the size of Action. Normally users can not get this. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTION_COMMON_ACTION_ID_NOT_FOUND, "The ID of Action is not found. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTION_COMMON_DOWNLOAD_ACTION_ID_RANGE_ERROR, "The download action start id is bigger than the action end id ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTION_COMMON_NO_ACTION_ITEMS_STORED, "Can not download or get min-max action ID for no action items stored in action kernel ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRIGGER_RESV, "Reserved error code", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRIGGER_TYPE_INVALID, "The type ID of Trigger is invalid. It might not defined or the information is empty. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRIGGER_REACH_WP_END_INDEX_LT_START_INDEX, "Wp_end_index is less than wp_start_index in reach_waypoint_trigger. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRIGGER_REACH_WP_INVALID_INTERVAL_WP_NUM, "Interval_wp_num is large than the difference of wp_start_index and wp_end_index in reach_waypoint_trigger. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRIGGER_REACH_WP_INVALID_AUTO_TERMINATE_WP_NUM , "Auto_terminate_wp_num is large than interval_wp_num in reach_waypoint_trigger. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRIGGER_ASSOCIATE_INVALID_TYPE, "The associate_type is greater than the maximum value. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRIGGER_SIMPLE_INTERVAL_INVALID_TYPE, "The interval type is greater than the maximum value. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_COMMON_RESV, "Reserved error code", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_COMMON_ACTUATOR_EXEC_NON_SUPPORTED, "The execution of Actuator is not supported e.g. try to stop camera shooting. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_COMMON_ACTUATOR_TYPE_INVALID, "The type ID of Actuator is invalid. It might not defined or the information is empty. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_COMMON_ACTUATOR_FUNC_INVALID, "The Function ID of Actuator is invalid. It might not defined or the information is empty. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_CAMERA_RESV, "Reserved error code", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_CAMERA_SEND_SINGLE_SHOT_CMD_TO_CAMERA_FAIL, "Fail to send shot cmd to camera for no camera or camera is busy. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_CAMERA_SEND_VIDEO_START_CMD_TO_CAMERA_FAIL, "Fail to send video start cmd to camera for no camera or camera is busy. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_CAMERA_SEND_VIDEO_STOP_CMD_TO_CAMERA_FAIL, "Fail to send video stop cmd to camera for no camera or camera is not busy. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_CAMERA_FOCUS_PARAM_XY_INVALID, "Camera focus param xy exceed valid range (0 1). ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_CAMERA_SEND_FOCUS_CMD_TO_CAMERA_F

AIL, "Fail to send focus cmd to camera for no camera or camera is busy. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_CAMERA_SEND_FOCALIZE_CMD_TO_CAMERA_FAIL, "Fail to send focalize cmd to camera for no camera or camera is busy. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_CAMERA_FOCAL_DISTANCE_INVALID, "Focal distance of camera focalize function exceed valid range. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_CAMERA_EXEC_FAIL, "This err code indicate camera fail to exec coressponding cmd and the low 8 bit", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_GIMBAL_RESV, "Reserved error code", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_GIMBAL_INVALID_RPY_ANGLE_CTRL_CMD, "Gimbal roll pitch yaw angle ctrl cmd param invalid ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_GIMBAL_INVALID_DURATION_CMD, "Gimbal duration param invalid unable to exec. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_GIMBAL_FAIL_TO_ARRIVE_TGT_ANGLE, "Gimbal fail to arrive target angle . ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_GIMBAL_FAIL_TO_SEND_CMD_TO_GIMBAL, "Fail to send cmd to gimbal for gimbal is busy or no gimbal. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_GIMBAL_THIS_INDEX_OF_GIMBAL_NOT_DOING_UNIFORM_CTRL , "Fail to stop gimbal uniform ctrl because index error. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_FLIGHT_RESV, "Reserved error code", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_FLIGHT_YAW_INVALID_YAW_ANGLE, "Yaw angle is lager max yaw angle. ", NULL}, \
{GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_FLIGHT_YAW_TO_TGT_ANGLE_TIMEOUT, "Failed to target yaw angle because of time

操作返回值

- 操作成功

#define GDU_RETURN_CODE_OK GDU_ERROR_SYSTEM_MODULE_CODE_SUCCESS

- 空间分配错误

#define GDU_RETURN_CODE_ERR_ALLOC GDU_ERROR_SYSTEM_MODULE_CODE_MEMORY_ALLOC_FAILED

- 操作超时

#define GDU_RETURN_CODE_ERR_TIMEOUT GDU_ERROR_SYSTEM_MODULE_CODE_TIMEOUT

- 对象不存在

#define GDU_RETURN_CODE_ERR_NOT_FOUND GDU_ERROR_SYSTEM_MODULE_CODE_NOT_FOUND

- 超出指定的范围

#define GDU_RETURN_CODE_ERR_OUT_OF_RANGE GDU_ERROR_SYSTEM_MODULE_CODE_OUT_OF_RANGE

- 参数错误

#define GDU_RETURN_CODE_ERR_PARAM GDU_ERROR_SYSTEM_MODULE_CODE_INVALID_PARAMETER

- 系统错误

#define GDU_RETURN_CODE_ERR_SYSTEM GDU_ERROR_SYSTEM_MODULE_CODE_SYSTEM_ERROR

- 系统繁忙

#define GDU_RETURN_CODE_ERR_BUSY GDU_ERROR_SYSTEM_MODULE_CODE_BUSY

- 暂不支持

#define GDU_RETURN_CODE_ERR_UNSUPPORT GDU_ERROR_SYSTEM_MODULE_CODE_NONSUPPORT

- 未知错误

#define GDU_RETURN_CODE_ERR_UNKNOWN GDU_ERROR_SYSTEM_MODULE_CODE_UNKNOWN

枚举

- 错误模块

```
typedef enum {
    GDU_ERROR_MODULE_SYSTEM = 0,
    GDU_ERROR_MODULE_PLATFORM,
    GDU_ERROR_MODULE_LOGGER,
    GDU_ERROR_MODULE_TIME_SYNC,
    GDU_ERROR_MODULE_COMMAND,
    GDU_ERROR_MODULE_CAMERA,
    GDU_ERROR_MODULE_GIMBAL,
    GDU_ERROR_MODULE_XPORT,
    GDU_ERROR_MODULE_PAYLOAD_COLLABORATION,
    GDU_ERROR_MODULE_WIDGET,
    GDU_ERROR_MODULE_CORE,
    GDU_ERROR_MODULE_IDENTITY,
    GDU_ERROR_MODULE_TRANSMISSION,
    GDU_ERROR_MODULE_DATA_CHANNEL,
    GDU_ERROR_MODULE_SUBSCRIPTION,
    GDU_ERROR_MODULE_MOP_CHANNEL,
    GDU_ERROR_MODULE_POSITIONING,
    GDU_ERROR_MODULE_POWER_MANAGEMENT,
    GDU_ERROR_MODULE_AIRCRAFTINFO,
    GDU_ERROR_MODULE_PRODUCTINFO,
    GDU_ERROR_MODULE_FLOWCONTROLLER,
    GDU_ERROR_MODULE_DOWNLOADER,
    GDU_ERROR_MODULE_PARAMETER,
    GDU_ERROR_MODULE_UTIL,
    GDU_ERROR_MODULE_USER,
    GDU_ERROR_MODULE_NEGOTIATE,
    GDU_ERROR_MODULE_UPGRADE,
    GDU_ERROR_MODULE_FC_BASIC,
    GDU_ERROR_MODULE_FC_JOYSTICK,
    GDU_ERROR_MODULE_FC_ACTION,
    GDU_ERROR_MODULE_FC_ARREST_FLYING,
    GDU_ERROR_MODULE_FC_HOME_LOCATION,
    GDU_ERROR_MODULE_FC_EMERGENCY_STOP_MOTOR,
    GDU_ERROR_MODULE_CAMERA_MANAGER,
    GDU_ERROR_MODULE_GIMBAL_MANAGER,
```

```
    GDU_ERROR_MODULE_WAYPOINT_V2,
    GDU_ERROR_MODULE_ERROR,
} E_GduErrorModule;
```

- 系统通用模块原始错误码

```
typedef enum {
    GDU_ERROR_SYSTEM_MODULE_RAW_CODE_SUCCESS = 0x000,
    GDU_ERROR_SYSTEM_MODULE_RAW_CODE_INVALID_REQUEST_PARAMETER = 0x0D4,
    GDU_ERROR_SYSTEM_MODULE_RAW_CODE_EXECUTING_HIGHER_PRIORITY_TASK = 0x0D7,
    GDU_ERROR_SYSTEM_MODULE_RAW_CODE_NONSUPPORT = 0x0E0,
    GDU_ERROR_SYSTEM_MODULE_RAW_CODE_TIMEOUT = 0x0E1,
    GDU_ERROR_SYSTEM_MODULE_RAW_CODE_MEMORY_ALLOC_FAILED = 0x0E2,
    GDU_ERROR_SYSTEM_MODULE_RAW_CODE_INVALID_PARAMETER = 0x0E3,
    GDU_ERROR_SYSTEM_MODULE_RAW_CODE_NONSUPPORT_IN_CURRENT_STATE = 0x0E4,
    GDU_ERROR_SYSTEM_MODULE_RAW_CODE_SYSTEM_ERROR = 0x0EC,
    GDU_ERROR_SYSTEM_MODULE_RAW_CODE_HARDWARE_ERR = 0x0FA,
    GDU_ERROR_SYSTEM_MODULE_RAW_CODE_INSUFFICIENT_ELECTRICITY = 0x0FB,
    GDU_ERROR_SYSTEM_MODULE_RAW_CODE_UNKNOWN = 0x0FF,
    GDU_ERROR_SYSTEM_MODULE_RAW_CODE_NOT_FOUND = 0x100,
    GDU_ERROR_SYSTEM_MODULE_RAW_CODE_OUT_OF_RANGE = 0x101,
    GDU_ERROR_SYSTEM_MODULE_RAW_CODE_BUSY = 0x102,
    GDU_ERROR_SYSTEM_MODULE_RAW_CODE_DUPLICATE = 0x103,
    GDU_ERROR_SYSTEM_MODULE_RAW_CODE_ADAPTER_NOT_MATCH = 0x104,
} E_GduErrorSystemModuleRawCode;
```

- 云台模块原始错误码

```
typedef enum {
    GDU_ERROR_GIMBAL_MODULE_RAW_CODE_PITCH_REACH_POSITIVE_LIMIT = 0x000,
    GDU_ERROR_GIMBAL_MODULE_RAW_CODE_PITCH_REACH_NEGATIVE_LIMIT = 0x001,
    GDU_ERROR_GIMBAL_MODULE_RAW_CODE_ROLL_REACH_POSITIVE_LIMIT = 0x002,
    GDU_ERROR_GIMBAL_MODULE_RAW_CODE_ROLL_REACH_NEGATIVE_LIMIT = 0x003,
    GDU_ERROR_GIMBAL_MODULE_RAW_CODE_YAW_REACH_POSITIVE_LIMIT = 0x004,
    GDU_ERROR_GIMBAL_MODULE_RAW_CODE_YAW_REACH_NEGATIVE_LIMIT = 0x005,
    GDU_ERROR_GIMBAL_MODULE_RAW_CODE_NON_CONTROL_AUTHORITY = 0x006,
} E_GduErrorGimbalModuleRawCode;
```

- 负载协同模块原始错误码

```
typedef enum {
    GDU_ERROR_PAYLOAD_COLLABORATION_MODULE_RAW_CODE_POSITION_NOT_MATCH = 0x000,
} E_GduErrorPayloadCollaborationModuleRawCode;
```

- 数据订阅模块原始错误码

```
typedef enum {
    GDU_ERROR_SUBSCRIPTION_MODULE_RAW_CODE_INVALID_TOPIC_FREQ = 0x000,
    GDU_ERROR_SUBSCRIPTION_MODULE_RAW_CODE_TOPIC_DUPLICATE = 0x001,
    GDU_ERROR_SUBSCRIPTION_MODULE_RAW_CODE_TOPIC_NOT_SUBSCRIBED = 0x002,
    GDU_ERROR_SUBSCRIPTION_MODULE_RAW_CODE_TIMESTAMP_NOT_ENABLE = 0x003,
    GDU_ERROR_SUBSCRIPTION_MODULE_RAW_CODE_TOPIC_NOT_SUPPORTED = 0x004,
} E_GduErrorSubscriptionModuleRawCode;
```

- 相机管理模块原始错误码

```
typedef enum {
    GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_UNSUPPORTED_COMMAND = 0xE0, /*!< 不支持本命令 */
    GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_TIMEOUT = 0xE1, /*!< 执行 */
    GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_RAM_ALLOCATION_FAILED = 0xE2, /*!< Memory alloc failed */
    GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_INVALID_COMMAND_PARAMETER = 0xE3, /*!< Invalid parameter for the command */

GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_UNSUPPORTED_COMMAND_IN_CUR_STATE = 0xE4, /*!< Do not support this command in the current state */
    GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_CAMERA_TIME_NOT_SYNCHRONIZED = 0xE5, /*!< Timestamp of camera is not synchronized */
    GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_PARAMETER_SET_FAILED = 0xE6, /*!< Setting parameter failed */
    GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_PARAMETER_GET_FAILED = 0xE7, /*!< Getting parameter failed */
    GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_SD_CARD_MISSING = 0xE8, /*!< SD card is not installed */
    GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_SD_CARD_FULL = 0xE9, /*!< SD card is full */
```

```
    GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_SD_CARD_ERROR = 0xEA, /*!< Error accessing the SD Card */
    GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_SENSOR_ERROR = 0xEB, /*!< Sensor go wrong */
    GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_SYSTEM_ERROR = 0xEC, /*!< System error */
    GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_PARAMETER_TOTAL_TOO_LONG = 0xED, /*!< Length of the parameter is too long */
    GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_MODULE_INACTIVATED = 0xEE, /*!< Module is too not activated yet */

GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_FIRMWARE_DATA_NUM_DISCONTINUOUS = 0xF0, /*!< Fireware data number is a discontinuous number */
    GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_FIRMWARE_VERIFICATION_ERROR = 0xF2, /*!< Error verifying fireware */
    GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_FLASH_WRITE_ERROR = 0xF4, /*!< Error writing flash */
    GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_FIRMWARE_TYPE_MISMATCH = 0xF6, /*!< Firmware type don't match */
    GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_REMOTE_CONTROL_UNCONNECTED = 0xF8, /*!< Not connect remote control yet */
    GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_HARDWARE_ERROR = 0xFA, /*!< Hardware fault */
    GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_AIRCRAFT_UNCONNECTED = 0xFC, /*!< Aircraft is not connected yet */
    GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_CANNOT_UPGRADE_IN_CUR_STATE = 0xFE, /*!< Cannot upgrade in current status (Please reboot or contact with GDU support */
    GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_UNDEFINE_ERROR = 0xFF, /*!< Undefined error */
} E_GduErrorCameraManagerModuleRawCode;
```

- PSDK 错误码 PSDK 错误码全集。用户可以搜索到所有错误码信息。请参考

  错误对象获取错误描述、错误原因以及错误恢复建议。

```
enum GduErrorCode {
    /* system module error code, including some common error code */
    GDU_ERROR_SYSTEM_MODULE_CODE_SUCCESS = GDU_ERROR_CODE(GDU_ERROR_MODULE_SYSTEM, GDU_ERROR_SYSTEM_MODULE_RAW_CODE_SUCCESS),
    GDU_ERROR_SYSTEM_MODULE_CODE_INVALID_REQUEST_PARAMETER =
```

```
GDU_ERROR_CODE(GDU_ERROR_MODULE_SYSTEM,
GDU_ERROR_SYSTEM_MODULE_RAW_CODE_INVALID_REQUEST_PARAMETER),
    GDU_ERROR_SYSTEM_MODULE_CODE_EXECUTING_HIGHER_PRIORITY_TASK =
GDU_ERROR_CODE(GDU_ERROR_MODULE_SYSTEM,
GDU_ERROR_SYSTEM_MODULE_RAW_CODE_EXECUTING_HIGHER_PRIORITY_TASK),
    GDU_ERROR_SYSTEM_MODULE_CODE_NONSUPPORT =
GDU_ERROR_CODE(GDU_ERROR_MODULE_SYSTEM,
GDU_ERROR_SYSTEM_MODULE_RAW_CODE_NONSUPPORT),
    GDU_ERROR_SYSTEM_MODULE_CODE_TIMEOUT =
GDU_ERROR_CODE(GDU_ERROR_MODULE_SYSTEM,
GDU_ERROR_SYSTEM_MODULE_RAW_CODE_TIMEOUT),
    GDU_ERROR_SYSTEM_MODULE_CODE_MEMORY_ALLOC_FAILED =
GDU_ERROR_CODE(GDU_ERROR_MODULE_SYSTEM,
GDU_ERROR_SYSTEM_MODULE_RAW_CODE_MEMORY_ALLOC_FAILED),
    GDU_ERROR_SYSTEM_MODULE_CODE_INVALID_PARAMETER =
GDU_ERROR_CODE(GDU_ERROR_MODULE_SYSTEM,
GDU_ERROR_SYSTEM_MODULE_RAW_CODE_INVALID_PARAMETER),
    GDU_ERROR_SYSTEM_MODULE_CODE_NONSUPPORT_IN_CURRENT_STATE =
GDU_ERROR_CODE(GDU_ERROR_MODULE_SYSTEM,
GDU_ERROR_SYSTEM_MODULE_RAW_CODE_NONSUPPORT_IN_CURRENT_STATE),
    GDU_ERROR_SYSTEM_MODULE_CODE_SYSTEM_ERROR =
GDU_ERROR_CODE(GDU_ERROR_MODULE_SYSTEM,
GDU_ERROR_SYSTEM_MODULE_RAW_CODE_SYSTEM_ERROR),
    GDU_ERROR_SYSTEM_MODULE_CODE_HARDWARE_ERR =
GDU_ERROR_CODE(GDU_ERROR_MODULE_SYSTEM,
GDU_ERROR_SYSTEM_MODULE_RAW_CODE_HARDWARE_ERR),
    GDU_ERROR_SYSTEM_MODULE_CODE_INSUFFICIENT_ELECTRICITY =
GDU_ERROR_CODE(GDU_ERROR_MODULE_SYSTEM,
GDU_ERROR_SYSTEM_MODULE_RAW_CODE_INSUFFICIENT_ELECTRICITY),
    GDU_ERROR_SYSTEM_MODULE_CODE_UNKNOWN =
GDU_ERROR_CODE(GDU_ERROR_MODULE_SYSTEM,
GDU_ERROR_SYSTEM_MODULE_RAW_CODE_UNKNOWN),
    GDU_ERROR_SYSTEM_MODULE_CODE_NOT_FOUND =
GDU_ERROR_CODE(GDU_ERROR_MODULE_SYSTEM,
GDU_ERROR_SYSTEM_MODULE_RAW_CODE_NOT_FOUND),
    GDU_ERROR_SYSTEM_MODULE_CODE_OUT_OF_RANGE =
GDU_ERROR_CODE(GDU_ERROR_MODULE_SYSTEM,
GDU_ERROR_SYSTEM_MODULE_RAW_CODE_OUT_OF_RANGE),
    GDU_ERROR_SYSTEM_MODULE_CODE_BUSY = GDU_ERROR_CODE(GDU_ERROR_MODULE_SYSTEM,
GDU_ERROR_SYSTEM_MODULE_RAW_CODE_BUSY),
    GDU_ERROR_SYSTEM_MODULE_CODE_DUPLICATE =
GDU_ERROR_CODE(GDU_ERROR_MODULE_SYSTEM,
GDU_ERROR_SYSTEM_MODULE_RAW_CODE_DUPLICATE),
    GDU_ERROR_SYSTEM_MODULE_CODE_ADAPTER_NOT_MATCH =
GDU_ERROR_CODE(GDU_ERROR_MODULE_SYSTEM,
```

```
GDU_ERROR_SYSTEM_MODULE_RAW_CODE_ADAPTER_NOT_MATCH),

    /* gimbal module error code */
    GDU_ERROR_GIMBAL_MODULE_CODE_PITCH_REACH_POSITIVE_LIMIT =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL,
GDU_ERROR_GIMBAL_MODULE_RAW_CODE_PITCH_REACH_POSITIVE_LIMIT),
    GDU_ERROR_GIMBAL_MODULE_CODE_PITCH_REACH_NEGATIVE_LIMIT =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL,
GDU_ERROR_GIMBAL_MODULE_RAW_CODE_PITCH_REACH_NEGATIVE_LIMIT),
    GDU_ERROR_GIMBAL_MODULE_CODE_ROLL_REACH_POSITIVE_LIMIT =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL,
GDU_ERROR_GIMBAL_MODULE_RAW_CODE_ROLL_REACH_POSITIVE_LIMIT),
    GDU_ERROR_GIMBAL_MODULE_CODE_ROLL_REACH_NEGATIVE_LIMIT =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL,
GDU_ERROR_GIMBAL_MODULE_RAW_CODE_ROLL_REACH_NEGATIVE_LIMIT),
    GDU_ERROR_GIMBAL_MODULE_CODE_YAW_REACH_POSITIVE_LIMIT =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL,
GDU_ERROR_GIMBAL_MODULE_RAW_CODE_YAW_REACH_POSITIVE_LIMIT),
    GDU_ERROR_GIMBAL_MODULE_CODE_YAW_REACH_NEGATIVE_LIMIT =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL,
GDU_ERROR_GIMBAL_MODULE_RAW_CODE_YAW_REACH_NEGATIVE_LIMIT),
    GDU_ERROR_GIMBAL_MODULE_CODE_NON_CONTROL_AUTHORITY =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL,
GDU_ERROR_GIMBAL_MODULE_RAW_CODE_NON_CONTROL_AUTHORITY),

    /* payload collaboration module error code */
    GDU_ERROR_PAYLOAD_COLLABORATION_MODULE_CODE_POSITION_NOT_MATCH =
GDU_ERROR_CODE(GDU_ERROR_MODULE_PAYLOAD_COLLABORATION,
GDU_ERROR_PAYLOAD_COLLABORATION_MODULE_RAW_CODE_POSITION_NOT_MATCH),

    /* activation error code of identity module */
    GDU_ERROR_IDENTITY_MODULE_CODE_ACTIVATE_PARAMETER_ERROR =
GDU_ERROR_CODE(GDU_ERROR_MODULE_IDENTITY,
GDU_ERROR_IDENTITY_MODULE_RAW_CODE_ACTIVATE_PARAMETER_ERROR),
    GDU_ERROR_IDENTITY_MODULE_CODE_ACTIVATE_ENCODE_ERROR =
GDU_ERROR_CODE(GDU_ERROR_MODULE_IDENTITY,
GDU_ERROR_IDENTITY_MODULE_RAW_CODE_ACTIVATE_ENCODE_ERROR),
    GDU_ERROR_IDENTITY_MODULE_CODE_ACTIVATE_NEW_DEVICE_ERROR =
GDU_ERROR_CODE(GDU_ERROR_MODULE_IDENTITY,
GDU_ERROR_IDENTITY_MODULE_RAW_CODE_ACTIVATE_NEW_DEVICE_ERROR),
    GDU_ERROR_IDENTITY_MODULE_CODE_ACTIVATE_SOFTWARE_NOT_CONNECTED =
GDU_ERROR_CODE(GDU_ERROR_MODULE_IDENTITY,
GDU_ERROR_IDENTITY_MODULE_RAW_CODE_ACTIVATE_SOFTWARE_NOT_CONNECTED),
    GDU_ERROR_IDENTITY_MODULE_CODE_ACTIVATE_NETWORK_ERROR =
GDU_ERROR_CODE(GDU_ERROR_MODULE_IDENTITY,
```

```
GDU_ERROR_IDENTITY_MODULE_RAW_CODE_ACTIVATE_NETWORK_ERROR),
    GDU_ERROR_IDENTITY_MODULE_CODE_ACTIVATE_SERVER_ACCESS_REFUSED =
GDU_ERROR_CODE(GDU_ERROR_MODULE_IDENTITY,
GDU_ERROR_IDENTITY_MODULE_RAW_CODE_ACTIVATE_SERVER_ACCESS_REFUSED),
    GDU_ERROR_IDENTITY_MODULE_CODE_ACTIVATE_ACCESS_LEVEL_ERROR =
GDU_ERROR_CODE(GDU_ERROR_MODULE_IDENTITY,
GDU_ERROR_IDENTITY_MODULE_RAW_CODE_ACTIVATE_ACCESS_LEVEL_ERROR),
    GDU_ERROR_IDENTITY_MODULE_CODE_ACTIVATE_OSDK_VERSION_ERROR =
GDU_ERROR_CODE(GDU_ERROR_MODULE_IDENTITY,
GDU_ERROR_IDENTITY_MODULE_RAW_CODE_ACTIVATE_OSDK_VERSION_ERROR),

    /* subscription module error code */
    GDU_ERROR_SUBSCRIPTION_MODULE_CODE_INVALID_TOPIC_FREQ =
GDU_ERROR_CODE(GDU_ERROR_MODULE_SUBSCRIPTION,
GDU_ERROR_SUBSCRIPTION_MODULE_RAW_CODE_INVALID_TOPIC_FREQ),
    GDU_ERROR_SUBSCRIPTION_MODULE_CODE_TOPIC_DUPLICATE =
GDU_ERROR_CODE(GDU_ERROR_MODULE_SUBSCRIPTION,
GDU_ERROR_SUBSCRIPTION_MODULE_RAW_CODE_TOPIC_DUPLICATE),
    GDU_ERROR_SUBSCRIPTION_MODULE_CODE_TOPIC_NOT_SUBSCRIBED =
GDU_ERROR_CODE(GDU_ERROR_MODULE_SUBSCRIPTION,
GDU_ERROR_SUBSCRIPTION_MODULE_RAW_CODE_TOPIC_NOT_SUBSCRIBED),
    GDU_ERROR_SUBSCRIPTION_MODULE_CODE_TIMESTAMP_NOT_ENABLE =
GDU_ERROR_CODE(GDU_ERROR_MODULE_SUBSCRIPTION,
GDU_ERROR_SUBSCRIPTION_MODULE_RAW_CODE_TIMESTAMP_NOT_ENABLE),
    GDU_ERROR_SUBSCRIPTION_MODULE_CODE_TOPIC_NOT_SUPPORTED =
GDU_ERROR_CODE(GDU_ERROR_MODULE_SUBSCRIPTION,
GDU_ERROR_SUBSCRIPTION_MODULE_RAW_CODE_TOPIC_NOT_SUPPORTED),

    /* mop channel module error code */
    GDU_ERROR_MOP_CHANNEL_MODULE_CODE_CONNECTION_CLOSE =
GDU_ERROR_CODE(GDU_ERROR_MODULE_MOP_CHANNEL,
GDU_ERROR_MOP_CHANNEL_MODULE_RAW_CODE_CONNECTION_CLOSE),

    /* Flight controller basic errors */
    GDU_ERROR_FC_MODULE_CODE_RC_MODE_ERROR =
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_BASIC,
GDU_ERROR_FC_MODULE_RAW_CODE_RC_MODE_ERROR),
    GDU_ERROR_FC_MODULE_CODE_RELEASE_CONTROL_SUCCESS=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_BASIC,
GDU_ERROR_FC_MODULE_RAW_CODE_RELEASE_CONTROL_SUCCESS),
    GDU_ERROR_FC_MODULE_CODE_OBTAIN_CONTROL_SUCCESS=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_BASIC,
GDU_ERROR_FC_MODULE_RAW_CODE_OBTAIN_CONTROL_SUCCESS),
    GDU_ERROR_FC_MODULE_CODE_OBTAIN_CONTROL_IN_PROGRESS=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_BASIC,
```

```
GDU_ERROR_FC_MODULE_RAW_CODE_OBTAIN_CONTROL_IN_PROGRESS),
    GDU_ERROR_FC_MODULE_CODE_RELEASE_CONTROL_IN_PROGRESS=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_BASIC,
GDU_ERROR_FC_MODULE_RAW_CODE_RELEASE_CONTROL_IN_PROGRESS),
    GDU_ERROR_FC_MODULE_CODE_RC_NEED_MODE_P=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_BASIC,
GDU_ERROR_FC_MODULE_RAW_CODE_RC_NEED_MODE_P),
    GDU_ERROR_FC_MODULE_CODE_RC_NEED_MODE_F=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_BASIC,
GDU_ERROR_FC_MODULE_RAW_CODE_RC_NEED_MODE_F),
    GDU_ERROR_FC_MODULE_CODE_PARAM_READ_WRITE_INVALID_PARAM=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_BASIC,
GDU_ERROR_FC_MODULE_RAW_CODE_PARAM_READ_WRITE_INVALID_PARAM),
    GDU_ERROR_FC_MODULE_CODE_IOC_OBTAIN_CONTROL_ERROR=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_BASIC,
GDU_ERROR_FC_MODULE_RAW_CODE_IOC_OBTAIN_CONTROL_ERROR),
    GDU_ERROR_FC_MODULE_CODE_KEY_ERROR=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_BASIC,
GDU_ERROR_FC_MODULE_RAW_CODE_KEY_ERROR),
    GDU_ERROR_FC_MODULE_CODE_NO_AUTHORIZATION_ERROR=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_BASIC,
GDU_ERROR_FC_MODULE_RAW_CODE_NO_AUTHORIZATION_ERROR),
    GDU_ERROR_FC_MODULE_CODE_NO_RIGHTS_ERROR=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_BASIC,
GDU_ERROR_FC_MODULE_RAW_CODE_NO_RIGHTS_ERROR),
    GDU_ERROR_FC_MODULE_CODE_SYSTEM_ERROR=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_BASIC,
GDU_ERROR_FC_MODULE_RAW_CODE_SYSTEM_ERROR),

    /* Flight controller joystick errors */
    GDU_ERROR_FC_JOYSTICK_MODULE_OBTAIN_RELEASE_JOYSTICK_AUTH_SUCCESS                =
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_JOYSTICK,
GDU_ERROR_FC_JOYSTICK_MODULE_RAW_CODE_SUCCESS),
    GDU_ERROR_FC_JOYSTICK_MODULE_DEVICE_NOT_ALLOW=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_JOYSTICK,
GDU_ERROR_FC_JOYSTICK_MODULE_RAW_CODE_DEVICE_NOT_ALLOW),
    GDU_ERROR_FC_JOYSTICK_MODULE_TAKING_OFF=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_JOYSTICK,
GDU_ERROR_FC_JOYSTICK_MODULE_RAW_CODE_TAKING_OFF),
    GDU_ERROR_FC_JOYSTICK_MODULE_LANDING=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_JOYSTICK,
GDU_ERROR_FC_JOYSTICK_MODULE_RAW_CODE_LANDING),
    GDU_ERROR_FC_JOYSTICK_MODULE_CMD_INVALID                                         =
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_JOYSTICK,
GDU_ERROR_FC_JOYSTICK_MODULE_RAW_CODE_CMD_INVALID),
```

```
    GDU_ERROR_FC_JOYSTICK_MODULE_RC_NOT_P_MODE                                    =
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_JOYSTICK,
GDU_ERROR_FC_JOYSTICK_MODULE_RAW_CODE_RC_NOT_P_MODE),
    GDU_ERROR_FC_JOYSTICK_MODULE_CMD_LENGTH_ERROR=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_JOYSTICK,
GDU_ERROR_FC_JOYSTICK_MODULE_RAW_CODE_CMD_LENGTH_ERROR),
    GDU_ERROR_FC_JOYSTICK_MODULE_HAS_NO_JOYSTICK_AUTHORITY=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_JOYSTICK,
GDU_ERROR_FC_JOYSTICK_MODULE_RAW_CODE_HAS_NO_JOYSTICK_AUTHORITY),
    GDU_ERROR_FC_JOYSTICK_MODULE_IN_RC_LOST_ACTION=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_JOYSTICK,
GDU_ERROR_FC_JOYSTICK_MODULE_RAW_CODE_IN_RC_LOST_ACTION),

    /* Flight controller action errors */
    GDU_ERROR_FC_ACTION_MODULE_TRIGGER_ERROR_MOTOR_ON=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_ACTION,
GDU_ERROR_FC_ACTION_MODULE_RAW_CODE_TRIGGER_ERROR_MOTOR_ON),
    GDU_ERROR_FC_ACTION_MODULE_TRIGGER_ERROR_MOTOR_OFF=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_ACTION,
GDU_ERROR_FC_ACTION_MODULE_RAW_CODE_TRIGGER_ERROR_MOTOR_OFF),
    GDU_ERROR_FC_ACTION_MODULE_TRIGGER_ERROR_IN_AIR=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_ACTION,
GDU_ERROR_FC_ACTION_MODULE_RAW_CODE_TRIGGER_ERROR_IN_AIR),
    GDU_ERROR_FC_ACTION_MODULE_TRIGGER_ERROR_NOT_IN_AIR=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_ACTION,
GDU_ERROR_FC_ACTION_MODULE_RAW_CODE_TRIGGER_ERROR_NOT_IN_AIR),
    GDU_ERROR_FC_ACTION_MODULE_TRIGGER_ERROR_HOME_POINT_NOT_SET=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_ACTION,
GDU_ERROR_FC_ACTION_MODULE_RAW_CODE_TRIGGER_ERROR_HOME_POINT_NOT_SET),
    GDU_ERROR_FC_ACTION_MODULE_TRIGGER_ERROR_BAD_GPS=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_ACTION,
GDU_ERROR_FC_ACTION_MODULE_RAW_CODE_TRIGGER_ERROR_BAD_GPS),
    GDU_ERROR_FC_ACTION_MODULE_TRIGGER_ERROR_IN_SIMULATION=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_ACTION,
GDU_ERROR_FC_ACTION_MODULE_RAW_CODE_TRIGGER_ERROR_IN_SIMULATION),
    GDU_ERROR_FC_ACTION_MODULE_TRIGGER_ERROR_CANNOT_START_MOTOR=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_ACTION,
GDU_ERROR_FC_ACTION_MODULE_RAW_CODE_TRIGGER_ERROR_CANNOT_START_MOTOR),
    GDU_ERROR_FC_ACTION_MODULE_TRIGGER_ERROR_LOW_VOLTAGE=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_ACTION,
GDU_ERROR_FC_ACTION_MODULE_RAW_CODE_TRIGGER_ERROR_LOW_VOLTAGE),
    GDU_ERROR_FC_ACTION_MODULE_TRIGGER_ERROR_SPEED_TOO_LARGE=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_ACTION,
GDU_ERROR_FC_ACTION_MODULE_RAW_CODE_TRIGGER_ERROR_SPEED_TOO_LARGE),
```

```
    /* Flight controller arrest flying errors */
    GDU_ERROR_FC_ARREST_FLYING_MODULE_REGISTER_LOGOUT_SET_SUCCESS =
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_ARREST_FLYING,
GDU_ERROR_FC_ARREST_FLYING_MODULE_RAW_CODE_REGISTER_LOGOUT_SET_SUCCESS),
    GDU_ERROR_FC_ARREST_FLYING_MODULE_REGISTER_LOGOUT_NULL_POINTER=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_ARREST_FLYING,
GDU_ERROR_FC_ARREST_FLYING_MODULE_RAW_CODE_REGISTER_LOGOUT_NULL_POINTER),
    GDU_ERROR_FC_ARREST_FLYING_MODULE_HMS_CODE_HAS_REGISTED=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_ARREST_FLYING,
GDU_ERROR_FC_ARREST_FLYING_MODULE_RAW_CODE_HMS_CODE_HAS_REGISTED),
    GDU_ERROR_FC_ARREST_FLYING_MODULE_HMS_CODE_NOT_FIND =
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_ARREST_FLYING,
GDU_ERROR_FC_ARREST_FLYING_MODULE_RAW_CODE_HMS_CODE_NOT_FIND),
    GDU_ERROR_FC_ARREST_FLYING_MODULE_ADD_ITEM_NO_DECRIPTION =
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_ARREST_FLYING,
GDU_ERROR_FC_ARREST_FLYING_MODULE_RAW_CODE_ADD_ITEM_NO_DECRIPTION),
    GDU_ERROR_FC_ARREST_FLYING_MODULE_REGISTER_ID_INVALID=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_ARREST_FLYING,
GDU_ERROR_FC_ARREST_FLYING_MODULE_RAW_CODE_REGISTER_ID_INVALID),
    GDU_ERROR_FC_ARREST_FLYING_MODULE_STOP_IN_AIR_HMSCODE_NOT_IN_WHITE_TABLE=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_ARREST_FLYING,
GDU_ERROR_FC_ARREST_FLYING_MODULE_RAW_CODE_STOP_IN_AIR_HMSCODE_NOT_IN_WHITE_TABL
E),
    GDU_ERROR_FC_ARREST_FLYING_MODULE_INVALID_FORMAT_HMSCODE=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_ARREST_FLYING,
GDU_ERROR_FC_ARREST_FLYING_MODULE_RAW_CODE_INVALID_FORMAT_HMSCODE),
    GDU_ERROR_FC_ARREST_FLYING_MODULE_HMSCODE_NOT_IN_WHITE_TABLE=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_ARREST_FLYING,
GDU_ERROR_FC_ARREST_FLYING_MODULE_RAW_CODE_HMSCODE_NOT_IN_WHITE_TABLE),

    /* Flight controller home location errors */
    GDU_ERROR_FC_HOME_LOCATION_MODULE_UNKNOWN_FAILED_REASON=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_HOME_LOCATION,
GDU_ERROR_FC_HOME_LOCATION_MODULE_RAW_CODE_UNKNOWN_FAILED_REASON),
    GDU_ERROR_FC_HOME_LOCATION_MODULE_INVALID_GPS_COORDINATE=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_HOME_LOCATION,
GDU_ERROR_FC_HOME_LOCATION_MODULE_RAW_CODE_INVALID_GPS_COORDINATE),
    GDU_ERROR_FC_HOME_LOCATION_MODULE_NOT_BE_RECORD =
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_HOME_LOCATION,
GDU_ERROR_FC_HOME_LOCATION_MODULE_RAW_CODE_NOT_BE_RECORD),
    GDU_ERROR_FC_HOME_LOCATION_MODULE_GPS_NOT_READY =
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_HOME_LOCATION,
GDU_ERROR_FC_HOME_LOCATION_MODULE_RAW_CODE_GPS_NOT_READY),
    GDU_ERROR_FC_HOME_LOCATION_MODULE_DIS_TOO_FAR =
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_HOME_LOCATION,
```

```
GDU_ERROR_FC_HOME_LOCATION_MODULE_RAW_CODE_DIS_TOO_FAR),

    /* Flight controller emergency stop motor errors */
    GDU_ERROR_FC_EMERGENCY_STOP_MOTOR_MODULE_VERSION_NOT_MATCH=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_EMERGENCY_STOP_MOTOR,
GDU_ERROR_FC_EMERGENCY_STOP_MOTOR_MODULE_RAW_CODE_VERSION_NOT_MATCH),
    GDU_ERROR_FC_EMERGENCY_STOP_MOTOR_MODULE_CMD_INVALID=
GDU_ERROR_CODE(GDU_ERROR_MODULE_FC_EMERGENCY_STOP_MOTOR,
GDU_ERROR_FC_EMERGENCY_STOP_MOTOR_MODULE_RAW_CODE_CMD_INVALID),

    /* Camera manager errors */
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_UNSUPPORTED_COMMAND                =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_UNSUPPORTED_COMMAND),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_TIMEOUT                            =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_TIMEOUT ),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_RAM_ALLOCATION_FAILED              =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_RAM_ALLOCATION_FAILED),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_INVALID_COMMAND_PARAMETER          =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_INVALID_COMMAND_PARAMETER),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_UNSUPPORTED_COMMAND_IN_CUR_STATE   =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_UNSUPPORTED_COMMAND_IN_CUR_STATE),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_CAMERA_TIME_NOT_SYNCHRONIZED       =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_CAMERA_TIME_NOT_SYNCHRONIZED),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_PARAMETER_SET_FAILED               =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_PARAMETER_SET_FAILED),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_PARAMETER_GET_FAILED               =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_PARAMETER_GET_FAILED),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_SD_CARD_MISSING                    =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_SD_CARD_MISSING),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_SD_CARD_FULL                       =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_SD_CARD_FULL),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_SD_CARD_ERROR                      =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_SD_CARD_ERROR),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_SENSOR_ERROR                       =
```

```
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_SENSOR_ERROR),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_SYSTEM_ERROR                    =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_SYSTEM_ERROR),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_PARAMETER_TOTAL_TOO_LONG        =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_PARAMETER_TOTAL_TOO_LONG),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_MODULE_INACTIVATED              =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_MODULE_INACTIVATED),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_FIRMWARE_DATA_NUM_DISCONTINUOUS =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_FIRMWARE_DATA_NUM_DISCONTINUOUS),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_FIRMWARE_VERIFICATION_ERROR     =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_FIRMWARE_VERIFICATION_ERROR),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_FLASH_WRITE_ERROR               =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_FLASH_WRITE_ERROR),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_FIRMWARE_TYPE_MISMATCH          =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_FIRMWARE_TYPE_MISMATCH),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_REMOTE_CONTROL_UNCONNECTED      =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_REMOTE_CONTROL_UNCONNECTED),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_HARDWARE_ERROR                  =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_HARDWARE_ERROR),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_AIRCRAFT_UNCONNECTED            =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_AIRCRAFT_UNCONNECTED),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_CANNOT_UPGRADE_IN_CUR_STATE     =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_CANNOT_UPGRADE_IN_CUR_STATE),
    GDU_ERROR_CAMERA_MANAGER_MODULE_CODE_UNDEFINE_ERROR                  =
GDU_ERROR_CODE(GDU_ERROR_MODULE_CAMERA_MANAGER,
GDU_ERROR_CAMERA_MANAGER_MODULE_RAW_CODE_UNDEFINE_ERROR),

    /* Gimbal manager errors */
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_UNSUPPORTED_COMMAND             =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_UNSUPPORTED_COMMAND),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_TIMEOUT                         =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
```

```
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_TIMEOUT ),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_RAM_ALLOCATION_FAILED =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_RAM_ALLOCATION_FAILED),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_INVALID_COMMAND_PARAMETER =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_INVALID_COMMAND_PARAMETER),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_UNSUPPORTED_COMMAND_IN_CUR_STATE =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_UNSUPPORTED_COMMAND_IN_CUR_STATE),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_CAMERA_TIME_NOT_SYNCHRONIZED =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_CAMERA_TIME_NOT_SYNCHRONIZED),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_PARAMETER_SET_FAILED =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_PARAMETER_SET_FAILED),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_PARAMETER_GET_FAILED =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_PARAMETER_GET_FAILED),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_SD_CARD_MISSING =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_SD_CARD_MISSING),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_SD_CARD_FULL =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_SD_CARD_FULL),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_SD_CARD_ERROR =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_SD_CARD_ERROR),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_SENSOR_ERROR =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_SENSOR_ERROR),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_SYSTEM_ERROR =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_SYSTEM_ERROR),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_PARAMETER_TOTAL_TOO_LONG =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_PARAMETER_TOTAL_TOO_LONG),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_MODULE_INACTIVATED =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_MODULE_INACTIVATED),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_FIRMWARE_DATA_NUM_DISCONTINUOUS =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_FIRMWARE_DATA_NUM_DISCONTINUOUS),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_FIRMWARE_VERIFICATION_ERROR =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
```

```
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_FIRMWARE_VERIFICATION_ERROR),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_FLASH_WRITE_ERROR =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_FLASH_WRITE_ERROR),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_FIRMWARE_TYPE_MISMATCH =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_FIRMWARE_TYPE_MISMATCH),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_REMOTE_CONTROL_UNCONNECTED =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_REMOTE_CONTROL_UNCONNECTED),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_HARDWARE_ERROR =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_HARDWARE_ERROR),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_AIRCRAFT_UNCONNECTED =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_AIRCRAFT_UNCONNECTED),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_CANNOT_UPGRADE_IN_CUR_STATE =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_CANNOT_UPGRADE_IN_CUR_STATE),
    GDU_ERROR_GIMBAL_MANAGER_MODULE_CODE_UNDEFINE_ERROR =
GDU_ERROR_CODE(GDU_ERROR_MODULE_GIMBAL_MANAGER,
GDU_ERROR_GIMBAL_MANAGER_MODULE_RAW_CODE_UNDEFINE_ERROR),

    /* Waypoint v2 total errors */
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_COMMON_SUCCESS =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_COMMON_SUCCESS),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_COMMON_INVALID_DATA_LENGTH =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_COMMON_INVALID_DATA_LENGTH),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_COMMON_INVALD_FLOAT_NUM =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_COMMON_INVALD_FLOAT_NUM),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_WP_VERSION_NO_MATCH =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_WP_VERSION_NO_MATCH),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_COMMON_UNKNOWN =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_COMMON_UNKNOWN),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_RESV =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_RESV),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INIT_WP_NUM_TOO_MANY =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_INIT_WP_NUM_TOO_MANY),
```

```
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INIT_WP_NUM_TOO_FEW                =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_INIT_WP_NUM_TOO_FEW    ),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INIT_INVALID_END_INDEX             =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_INIT_INVALID_END_INDEX ),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOAD_START_ID_GT_END_ID          =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_UPLOAD_START_ID_GT_END_ID),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOAD_END_ID_GT_TOTAL_NUM         =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_UPLOAD_END_ID_GT_TOTAL_NUM),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_DOWNLOAD_WPS_NOT_IN_STORED_RAGNE   =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_DOWNLOAD_WPS_NOT_IN_STORED_RAGNE),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_CUR_POS_IS_FAR_AWAY_FROM_FIRST_WP  =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_CUR_POS_IS_FAR_AWAY_FROM_FIRST_WP),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_ADJ_WPS_TOO_CLOSE                  =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_ADJ_WPS_TOO_CLOSE ),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_ADJ_WPS_TOO_FAR                    =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_ADJ_WPS_TOO_FAR    ),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOAD_MAX_VEL_GT_GLOBAL           =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_UPLOAD_MAX_VEL_GT_GLOBAL ),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOAD_LOCAL_CRUISE_VEL_GT_LOCAL_MAX
=                                          GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_UPLOAD_LOCAL_CRUISE_VEL_GT_LOCAL_MAX
),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOAD_LOCAL_CRUISE_VEL_GT_GLOBAL_MAX
=                                          GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_UPLOAD_LOCAL_CRUISE_VEL_GT_GLOBAL_MA
X),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INIT_INVALID_GLOBAL_MAX_VEL        =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_INIT_INVALID_GLOBAL_MAX_VEL    ),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_GLOBAL_CRUISE_VEL_GT_MAX_VEL       =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_GLOBAL_CRUISE_VEL_GT_MAX_VEL ),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INIT_INVALID_GOTO_FIRST_FLAG       =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_INIT_INVALID_GOTO_FIRST_FLAG ),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INIT_INVALID_FINISHED_ACTION       =
```

```
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_INIT_INVALID_FINISHED_ACTION ),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INIT_INVALID_RC_LOST_ACTION                =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_INIT_INVALID_RC_LOST_ACTION),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOAD_YAW_MODE_INVALID                    =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_UPLOAD_YAW_MODE_INVALID),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOAD_YAW_CMD_NOT_IN_RANGE                =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_UPLOAD_YAW_CMD_NOT_IN_RANGE),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOAD_YAW_TURN_DIRECTION_INVALID      =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_UPLOAD_YAW_TURN_DIRECTION_INVALID),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOAD_WP_TYPE_INVALID                     =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_UPLOAD_WP_TYPE_INVALID),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_GO_STOP_CMD_INVALID                        =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_GO_STOP_CMD_INVALID),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INVALID_PAUSE_RECOVERY_CMD                 =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_INVALID_PAUSE_RECOVERY_CMD),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INVALID_BREAK_RESTORE_CMD                  =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_INVALID_BREAK_RESTORE_CMD),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INIT_INVALID_REF_POINT                     =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_INIT_INVALID_REF_POINT),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_DAMPING_DIS_GE_DIS_OF_ADJ_POINTS       =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_DAMPING_DIS_GE_DIS_OF_ADJ_POINTS),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOAD_CANNT_SET_WP_LINE_EXIT_TYPE    =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_UPLOAD_CANNT_SET_WP_LINE_EXIT_TYPE),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INIT_INFO_NOT_UPLOADED                     =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_INIT_INFO_NOT_UPLOADED),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_WP_HAS_NOT_UPLOADED                        =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_WP_HAS_NOT_UPLOADED),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOADED_WP_NOT_ENOUGH                     =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_UPLOADED_WP_NOT_ENOUGH),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_GS_HAS_STARTED                             =
```

```
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_GS_HAS_STARTED),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_GS_NOT_RUNNING                    =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_GS_NOT_RUNNING),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_GS_NOT_RUNNING_FOR_PAUSE_RECOVERY    =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_GS_NOT_RUNNING_FOR_PAUSE_RECOVERY),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_GS_NOT_RUNNING_FOR_BREAK_RESTORE     =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_GS_NOT_RUNNING_FOR_BREAK_RESTORE),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_NOT_IN_WP_MIS                     =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_NOT_IN_WP_MIS ),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_MIS_HAS_BEEN_PAUSED               =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_MIS_HAS_BEEN_PAUSED),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_MIS_NOT_PAUSED                    =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_MIS_NOT_PAUSED),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_MIS_HAS_BEEN_BROKEN               =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_MIS_HAS_BEEN_BROKEN),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_MIS_NOT_BROKEN                    =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_MIS_NOT_BROKEN),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_PAUSE_RECOVERY_NOT_SUPPORTED      =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_PAUSE_RECOVERY_NOT_SUPPORTED),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_BREAK_RESTORE_NOT_SUPPORTED       =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_BREAK_RESTORE_NOT_SUPPORTED),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_NO_BREAK_POINT                    =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_NO_BREAK_POINT ),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_NO_CUR_TRAJ_PROJECT               =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_NO_CUR_TRAJ_PROJECT),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_NO_NXT_TRAJ_PROJECT               =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_NO_NXT_TRAJ_PROJECT),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_NO_NNT_TRAJ_PROJECT               =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_NO_NNT_TRAJ_PROJECT),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_UPLOAD_WP_ID_NOT_CONTINUE         =
```

```
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_UPLOAD_WP_ID_NOT_CONTINUE),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_WP_LINE_ENTER_NOT_SET_TO_START_WP     =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_WP_LINE_ENTER_NOT_SET_TO_START_WP ),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_INIT_WP_WHEN_PLAN_HAS_STARTED         =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_INIT_WP_WHEN_PLAN_HAS_STARTED ),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_DAMPING_DIS_EXCEED_RANGE              =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_DAMPING_DIS_EXCEED_RANGE),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_WAYPOINT_COOR_EXCEED_RANGE            =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_WAYPOINT_COOR_EXCEED_RANGE),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_FIRST_WP_TYPE_IS_WP_TURN_NO           =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_FIRST_WP_TYPE_IS_WP_TURN_NO),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_WP_EXCEED_RADIUS_LIMIT                =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_WP_EXCEED_RADIUS_LIMIT ),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRAJ_WP_EXCEED_HEIGHT_LIMIT                =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRAJ_WP_EXCEED_HEIGHT_LIMIT ),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_STATUS_RESV                                =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_STATUS_RESV),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_STATUS_WP_MIS_CHECK_FAIL                   =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_STATUS_WP_MIS_CHECK_FAIL),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_STATUS_HOME_NOT_RECORDED                   =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_STATUS_HOME_NOT_RECORDED),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_STATUS_LOW_LOCATION_ACCURACY               =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_STATUS_LOW_LOCATION_ACCURACY),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_STATUS_RTK_CONDITION_IS_NOT_READY          =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_STATUS_RTK_CONDITION_IS_NOT_READY),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_SECURE_RESV                                =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_SECURE_RESV),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_SECURE_CROSS_NFZ                           =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_SECURE_CROSS_NFZ),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_SECURE_BAT_LOW                             =
```

```
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_SECURE_BAT_LOW),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTION_COMMON_RESV                    =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTION_COMMON_RESV),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTION_COMMON_ACTION_ID_DUPLICATED    =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTION_COMMON_ACTION_ID_DUPLICATED),

GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTION_COMMON_ACTION_ITEMS_SPACE_NOT_ENOUGH
=                                       GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTION_COMMON_ACTION_ITEMS_SPACE_NOT_EN
OUGH),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTION_COMMON_ACTION_SIZE_GT_BUF_SIZE  =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTION_COMMON_ACTION_SIZE_GT_BUF_SIZE),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTION_COMMON_ACTION_ID_NOT_FOUND     =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTION_COMMON_ACTION_ID_NOT_FOUND),

GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTION_COMMON_DOWNLOAD_ACTION_ID_RANGE_ERR
OR                  =                   GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTION_COMMON_DOWNLOAD_ACTION_ID_RANG
E_ERROR),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTION_COMMON_NO_ACTION_ITEMS_STORED  =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTION_COMMON_NO_ACTION_ITEMS_STORED),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRIGGER_RESV                          =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRIGGER_RESV),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRIGGER_TYPE_INVALID                  =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRIGGER_TYPE_INVALID),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRIGGER_REACH_WP_END_INDEX_LT_START_INDEX =
GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRIGGER_REACH_WP_END_INDEX_LT_START_INDEX),
    GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRIGGER_REACH_WP_INVALID_INTERVAL_WP_NUM
=                                       GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRIGGER_REACH_WP_INVALID_INTERVAL_WP_NUM
),

GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRIGGER_REACH_WP_INVALID_AUTO_TERMINATE_WP_N
UM                  =                   GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2,
GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRIGGER_REACH_WP_INVALID_AUTO_TERMINATE_
WP_NUM),
```

GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRIGGER_ASSOCIATE_INVALID_TYPE = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRIGGER_ASSOCIATE_INVALID_TYPE),
GDU_ERROR_WAYPOINT_V2_MODULE_CODE_TRIGGER_SIMPLE_INTERVAL_INVALID_TYPE = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_TRIGGER_SIMPLE_INTERVAL_INVALID_TYPE),
GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_COMMON_RESV = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_COMMON_RESV),

GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_COMMON_ACTUATOR_EXEC_NON_SUPPORTED = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_COMMON_ACTUATOR_EXEC_NON_SUPPORTED),
GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_COMMON_ACTUATOR_TYPE_INVALID = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_COMMON_ACTUATOR_TYPE_INVALID),
GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_COMMON_ACTUATOR_FUNC_INVALID = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_COMMON_ACTUATOR_FUNC_INVALID),
GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_CAMERA_RESV = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_CAMERA_RESV),

GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_CAMERA_SEND_SINGLE_SHOT_CMD_TO_CAMERA_FAIL = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_CAMERA_SEND_SINGLE_SHOT_CMD_TO_CAMERA_FAIL),

GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_CAMERA_SEND_VIDEO_START_CMD_TO_CAMERA_FAIL = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_CAMERA_SEND_VIDEO_START_CMD_TO_CAMERA_FAIL),

GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_CAMERA_SEND_VIDEO_STOP_CMD_TO_CAMERA_FAIL = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_CAMERA_SEND_VIDEO_STOP_CMD_TO_CAMERA_FAIL),
GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_CAMERA_FOCUS_PARAM_XY_INVALID = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_CAMERA_FOCUS_PARAM_XY_INVALID),

GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_CAMERA_SEND_FOCUS_CMD_TO_CAMERA_FAIL = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_CAMERA_SEND_FOCUS_CMD_TO_CAME

RA_FAIL),

GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_CAMERA_SEND_FOCALIZE_CMD_TO_CAMERA_FAIL = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_CAMERA_SEND_FOCALIZE_CMD_TO_CAMERA_FAIL),
GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_CAMERA_FOCAL_DISTANCE_INVALID = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_CAMERA_FOCAL_DISTANCE_INVALID),
GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_CAMERA_EXEC_FAIL = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_CAMERA_EXEC_FAIL),
GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_GIMBAL_RESV = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_GIMBAL_RESV),

GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_GIMBAL_INVALID_RPY_ANGLE_CTRL_CMD = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_GIMBAL_INVALID_RPY_ANGLE_CTRL_CMD),
GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_GIMBAL_INVALID_DURATION_CMD = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_GIMBAL_INVALID_DURATION_CMD),
GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_GIMBAL_FAIL_TO_ARRIVE_TGT_ANGLE = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_GIMBAL_FAIL_TO_ARRIVE_TGT_ANGLE),

GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_GIMBAL_FAIL_TO_SEND_CMD_TO_GIMBAL = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_GIMBAL_FAIL_TO_SEND_CMD_TO_GIMBAL),

GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_GIMBAL_THIS_INDEX_OF_GIMBAL_NOT_DOING_UNIFORM_CTRL = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_GIMBAL_THIS_INDEX_OF_GIMBAL_NOT_DOING_UNIFORM_CTRL),
GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_FLIGHT_RESV = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_FLIGHT_RESV),
GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_FLIGHT_YAW_INVALID_YAW_ANGLE = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_FLIGHT_YAW_INVALID_YAW_ANGLE),
GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_FLIGHT_YAW_TO_TGT_ANGLE_TIMEOUT = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_FLIGHT_YAW_TO_TGT_ANGLE_TIMEOUT

),
GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_FLIGHT_ACTION_YAW_OCCUPIED = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_FLIGHT_ACTION_YAW_OCCUPIED),

GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_FLIGHT_CUR_AND_TGT_VEL_CLE_STATUE_EQUAL = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_FLIGHT_CUR_AND_TGT_VEL_CLE_STATUE_EQUAL),
GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_PAYLOAD_RESV = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_PAYLOAD_RESV),

GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_PAYLOAD_FAIL_TO_SEND_CMD_TO_PAYLOAD = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_PAYLOAD_FAIL_TO_SEND_CMD_TO_PAYLOAD),
GDU_ERROR_WAYPOINT_V2_MODULE_CODE_ACTUATOR_PAYLOAD_EXEC_FAILED = GDU_ERROR_CODE(GDU_ERROR_MODULE_WAYPOINT_V2, GDU_ERROR_WAYPOINT_V2_MODULE_RAW_CODE_ACTUATOR_PAYLOAD_EXEC_FAILED),
};

GDU Typedef

Typedef 的头文件为

gdu_typedef.h

本文档描述了

gdu_typedef.h

文件中宏定义和结构体的关键信息和使用方法。

宏定义

- 圆周率

```
#define GDU_PI                              (3.141592653589793238f)
```

- 文件名大小

```
#define GDU_FILE_NAME_SIZE_MAX          256
```

- 路径长度

```
#define GDU_FILE_PATH_SIZE_MAX          (GDU_FILE_NAME_SIZE_MAX + 256)
```

- IP 地址长度

```
#define GDU_IP_ADDR_STR_SIZE_MAX      15
```

- MD5 缓冲区长度

```
#define GDU_MD5_BUFFER_LEN                16
```

- 订阅模块索引偏移量

```
#define GDU_SUBSCRIPTION_MODULE_INDEX_OFFSET        24u
```

- 订阅模块索引掩码

```
#define GDU_SUBSCRIPTION_MODULE_INDEX_MASK           0xFF000000u

#define GDU_SUBSCRIPTION_TOPIC_CODE_OFFSET           0u

#define GDU_SUBSCRIPTION_TOPIC_CODE_MASK             0x00FFFFFFu

#define GDU_DATA_SUBSCRIPTION_TOPIC(subscriptionModule, topicCode) \
(uint32_t)\
(((((uint32_t)(subscriptionModule))        <<        (GDU_SUBSCRIPTION_MODULE_INDEX_OFFSET))        &
```

(GDU_SUBSCRIPTION_MODULE_INDEX_MASK)) | \
((((uint32_t)(topicCode)) << (GDU_SUBSCRIPTION_TOPIC_CODE_OFFSET)) & (GDU_SUBSCRIPTION_TOPIC_CODE_MASK)))

类型

- 双精度浮点类型

typedef double gdu_f64_t;

- 单精度浮点类型

typedef float gdu_f32_t;

- 返回值类型定义（uint64）可以取详情请参见：GduErrorCode 中的任意值。

typedef uint64_t T_GduReturnCode;

枚举值

typedef enum E_GduCameraMediaFileType

相机媒体文件类型

```
typedef enum {
    GDU_CAMERA_FILE_TYPE_JPEG = 0, /*!< JPEG */
    GDU_CAMERA_FILE_TYPE_DNG = 1, /*!< DNG */
    GDU_CAMERA_FILE_TYPE_MOV = 2, /*!< MOV */
    GDU_CAMERA_FILE_TYPE_MP4 = 3, /*!< MP4 */
    GDU_CAMERA_FILE_TYPE_UNKNOWN = 255, /*!< 未知类型。 */
} E_GduCameraMediaFileType;
```

typedef enum E_GduGimbalMode

云台模式

```
typedef enum {
    GDU_GIMBAL_MODE_FREE = 0, /*!< 自由模式，在地面坐标系固定云台姿态，忽略飞行器的运动。 */
    GDU_GIMBAL_MODE_FPV = 1, /*!< FPV（第一人称视角）模式，仅控制云台在地面坐标中的横滚和偏航角跟随飞行器。 */
    GDU_GIMBAL_MODE_YAW_FOLLOW = 2, /*!< 偏航跟随模式，只控制云台在地面坐标的偏航角跟随飞行器。*/
} E_GduGimbalMode;
```

typedef enum E_GduGimbalRotationMode

云台旋转模式

```
typedef enum {
    GDU_GIMBAL_ROTATION_MODE_RELATIVE_ANGLE = 0, /*!< 相对角度旋转模式，表示根据当前角度旋转云台指定角度。 */
    GDU_GIMBAL_ROTATION_MODE_ABSOLUTE_ANGLE = 1, /*!< 绝对角度旋转模式，表示将云台旋转到地面坐标中的指定角度。 */
    GDU_GIMBAL_ROTATION_MODE_SPEED = 2, /*!< 速度旋转模式，指定云台在地面坐标下的旋转速度。 */
} E_GduGimbalRotationMode;
```

typedef enum E_GduMobileAppScreenType

订阅频率类型

```
typedef enum {
    GDU_DATA_SUBSCRIPTION_TOPIC_1_HZ = 1,
```

```
    GDU_DATA_SUBSCRIPTION_TOPIC_5_HZ = 5,
    GDU_DATA_SUBSCRIPTION_TOPIC_10_HZ = 10,
    GDU_DATA_SUBSCRIPTION_TOPIC_50_HZ = 50,
    GDU_DATA_SUBSCRIPTION_TOPIC_100_HZ = 100,
    GDU_DATA_SUBSCRIPTION_TOPIC_200_HZ = 200,
    GDU_DATA_SUBSCRIPTION_TOPIC_400_HZ = 400,
} E_GduDataSubscriptionTopicFreq;
```

typedef enum E_GduDataSubscriptionModule

GDU 模块枚举，用于定义数据订阅模块

```
typedef enum {
    GDU_DATA_SUBSCRIPTION_MODULE_FC = 0,
    GDU_DATA_SUBSCRIPTION_MODULE_CAMERA,
    GDU_DATA_SUBSCRIPTION_MODULE_ERROR,
} E_GduDataSubscriptionModule;
```

结构体

typedef struct T_GduAttitude3d

```
typedef struct {
    int32_t pitch; /*!< 指定俯仰姿态的 int32 值 */
    int32_t roll; /*!< 指定横滚姿态的 int32 值 */
    int32_t yaw; /*!< 指定偏航姿态的 int32 值 */
} T_GduAttitude3d;
```

typedef struct T_GduAttitude3f

```
typedef struct {
```

```
    gdu_f32_t pitch; /*!< 指定俯仰姿态的浮点值*/
    gdu_f32_t roll; /*!< 指定横滚姿态的浮点值*/
    gdu_f32_t yaw; /*!< 指定偏航姿态的浮点值*/
} T_GduAttitude3f;
```

typedef struct T_GduQuaternion4f

四元数数据结构

```
typedef struct {
    gdu_f32_t q0; /*!< w, 当转换为旋转矩阵或欧拉角时。 */
    gdu_f32_t q1; /*!< x, 当转换为旋转矩阵或欧拉角时。 */
    gdu_f32_t q2; /*!< y, 当转换为旋转矩阵或欧拉角时。 */
    gdu_f32_t q3; /*!< z, 当转换为旋转矩阵或欧拉角时。 */
} T_GduQuaternion4f;
```

typedef struct T_GduDataTimestamp

时间戳数据结构

```
typedef struct {
    uint32_t millisecond; /*!< 毫秒 */
    uint32_t microsecond; /*!< 微秒 */
} T_GduDataTimestamp;
```

typedef function GduReceiveDataOfTopicCallback

用于接收 topic 数据的回调函数原型。

说明： 用户不能在回调函数中执行阻塞式操作或函数，因为这会阻塞 PSDK 根线程，导致系统响应慢、payload 断开或死循环等问题。

```
typedef T_GduReturnCode (*GduReceiveDataOfTopicCallback)(const uint8_t *data, uint16_t dataSize,
                                                         const T_GduDataTimestamp
*timestamp);
```

参数

data： 指向 topic 数据的指针，用户需要将此指针的类型传递给对应的数据结构指针，以便于获取 topic 的每一项。

dataSize： data 参数指向的内存空间大小，等于 topic 对应的数据结构大小。

timestamp： 指向与此数据对应的时间戳的指针。

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

基础功能

日志管理

日志管理相关功能的头文件为 gdu_logger.h，本文档描述了 gdu_logger.h 文件中结构体和函数原型的关键信息和使用方法。

目录

GduLogger_AddConsole

GduLogger_UserLogOutput

宏定义、枚举与结构体

typedef function ConsoleFunc

功能：需要被注册的控制台方法	product：all

注册日志输出方式

说明：

在注册日志输出方式前，请先测试需要注册的方法，以确保他们能被正常使用。

typedef T_GduReturnCode (*ConsoleFunc)(const uint8_t *data, uint16_t dataLen);

typedef enum E_GduLoggerConsoleLogLevel

日志级别。

typedef enum {

GDU_LOGGER_CONSOLE_LOG_LEVEL_ERROR = 0, 打印系统错误类型（Error）的日志。控制台的方法与等级是彼此相关联的。如果注册的控制台方法的级别低于当前等级，等级接口不会被成功打印。

GDU_LOGGER_CONSOLE_LOG_LEVEL_WARN = 1, 打印警告信息类型（Warning）的日志。控制台的方法与等级是彼此相关联的。如果注册的控制台方法的级别低于当前等级，等级接口不会被成功打印。

GDU_LOGGER_CONSOLE_LOG_LEVEL_INFO = 2, 打印关键信息类型（Info）的日志。控制台

的方法与等级是彼此相关联的。如果注册的控制台方法的级别低于当前等级，等级接口不会被成功打印。

GDU_LOGGER_CONSOLE_LOG_LEVEL_DEBUG = 3, 打印调试信息类型（Debug）的日志。控制台的方法与等级是彼此相关联的。如果注册的控制台方法的级别低于当前等级，等级接口不会被成功打印。

} E_GduLoggerConsoleLogLevel;

typedef struct T_GduLoggerConsole

日志等级

typedef struct {

uint8_t consoleLevel; 指定所需打印的日志等级，日志的等级从高到低为 Debug、Info、Warn 和 Error，日志管理功能模块可打印不高于指定等级的所有日志

ConsoleFunc func; 指定日志输出方式，在注册日志输出方式前，请先测试该日志输出方式能够正常打印用户所需的日志

bool isSupportColor;

} T_GduLoggerConsole;

函数原型

function GduLogger_AddConsole

功能：为 OSDK 添加控制台功能与等级。	product：all

说明：

在注册控制台前，用户需要提供控制台方法以及与该方法相关的等级。日志等级从高到低为 Debug, Info, Warn 和 Error。日志功能模块可以打印所有不高于特定级别的日志。最大支持同时注册 8 种不同的控制台方法。在注册控制台方法前，你需要测试注

册模块以确保所有的方法是正常的。如果用户同时注册多个模块，所有的方法都会被打印。

T_GduReturnCode GduLogger_AddConsole(T_GduLoggerConsole *console);

参数

console：指向控制台功能的方法。

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

function GduLogger_UserLogOutput

功能：通过注册方法打印输出特定格式的选定等级的日志。	product：all

说明：

注册的方法是按照对应级别打印的。如果控制台的级别比需要被打印的 log 的级别低，将不会打印成功。

void PsdkLogger_UserLogInfo(const char *fmt, ...);

参数

fmt：指向用户需打印的日志内容。

...：可变参数，与 printf 中的可变参数用法一致 。

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

信息管理

无人机信息（信息管理）相关功能的头文件为

gdu_aircraft_info.h

，本文档描述了

gdu_aircraft_info.h

文件中结构体和函数原型的关键信息和使用方法。

目录

宏定义、枚举与结构体

typedef struct T_GduMobileAppInfo

移动设备 APP 相关信息。

```
typedef struct {
    E_GduMobileAppLanguage appLanguage; /*!< 移动设备 APP 系统语言 */
    E_GduMobileAppScreenType appScreenType; /*!< 移动设备 APP 屏幕尺寸类型 */
} T_GduMobileAppInfo;
```

typedef struct T_GduAircraftInfoBaseInfo

飞行器系统的一些基础信息，主要包括系统的一些常数参数信息。

```
typedef struct {
    E_GduAircraftType aircraftType; /*!< 无人机类型 */
    E_GduSdkAdapterType GDUAdapterType; /*!< GDU 适配器类型 */
    E_GduMountPosition mountPosition; /*!< 负载安装位置 */
} T_GduAircraftInfoBaseInfo;
```

函数原型

function GduAircraftInfo_GetBaseInfo

功能：获取飞行器系统的基本信息	product：all

获取飞行器系统的基本信息，包括飞行器类型和 GDU 适配器类型。

```
T_GduReturnCode GduAircraftInfo_GetBaseInfo(T_GduAircraftInfoBaseInfo *baseInfo);
```

参数

baseInfo：指向用于存储飞机系统基本信息的内存空间的指针。

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

function GduAircraftInfo_GetMobileAppInfo

功能：获取移动设备 App 的基础信息	product：all

说明： 如果遥控器或 APP 未连接飞行器系统，移动设备 APP 语言和屏幕类型未知。

T_GduReturnCode GduAircraftInfo_GetMobileAppInfo(T_GduMobileAppInfo *mobileAppInfo);

参数

mobileAppInfo：指向用于存储移动 APP 相关信息的内存空间的指针。

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

消息订阅

消息订阅相关功能的头文件为 gdu_fc_subscription.h，本文档描述了 gdu_fc_subscription.h 文件中结构体和函数原型的关键信息和使用方法。

目录

E_GduFcSubscriptionDisplayMode

E_GduFcSubscriptionHomePointSetStatus

T_GduFcSubscriptionQuaternion

T_GduFcSubscriptionAccelerationGround

T_GduFcSubscriptionAccelerationBody

T_GduFcSubscriptionVelocity

T_GduFcSubscriptionAngularRateFusioned

T_GduFcSubscriptionAngularRateRaw

T_GduFcSubscriptionPositionFused

T_GduFcSubscriptionGpsPosition

T_GduFcSubscriptionGpsVelocity

T_GduFcSubscriptionGpsDetails

T_GduFcSubscriptionRtkPosition

T_GduFcSubscriptionRtkVelocity

T_GduFcSubscriptionCompass

T_GduFcSubscriptionRC

T_GduFcSubscriptionGimbalAngles

T_GduFcSubscriptionGimbalStatus

T_GduFcSubscriptionSingleBatteryState

T_GduFcSubscriptionWholeBatteryInfo

T_GduFcSubscriptionSingleBatteryInfo

T_GduFcSubscriptionControlDevice

SyncTimestamp

T_GduFcSubscriptionHardSync

T_GduFcSubscriptionRCWithFlagData

EscStatusIndividual

T_GduFcSubscriptionEscData

T_GduFcSubscriptionRTKConnectStatus

T_GduFcSubscriptionFlightAnomaly

T_GduFcSubscriptionPositionVO

T_GduFcSubscriptionAvoidData

T_GduFcSubscriptionHomePointInfo

GimbalSingleData

T_GduFcSubscriptionThreeGimbalData

- 函数原型

  GduFcSubscription_Init

  GduFcSubscription_DeInit

  GduFcSubscription_SubscribeTopic

  GduFcSubscription_GetLatestValueOfTopic

宏定义、枚举与结构体

- 数据结构

```
typedef gdu_f32_t T_GduFcSubscriptionAltitudeFused;
```

typedef gdu_f32_t T_GduFcSubscriptionAltitudeBarometer;

typedef gdu_f32_t T_GduFcSubscriptionAltitudeOfHomePoint;

typedef gud_f32_t T_GduFcSubscriptionHeightFusion;

typedef gdu_f32_t T_GduFcSubscriptionHeightRelative;

typedef uint32_t T_GduFcSubscriptionGpsDate;

typedef uint32_t T_GduFcSubscriptionGpsTime;

typedef uint8_t T_GduFcSubscriptionRtkPositionInfo;

typedef uint8_t T_GduFcSubscriptionRtkYawInfo;

typedef uint8_t T_GduFcSubscriptionFlightStatus;

typedef uint8_t T_GduFcSubscriptionDisplaymode;

typedef uint8_t T_GduFcSubscriptionLandinggear;

typedef uint16_t T_GduFcSubscriptionMotorStartError;

typedef uint8_t T_GduFcSubscriptionGpsControlLevel;

typedef uint8_t T_GduFcSubscriptionGimbalControlMode;

typedef uint8_t T_GduFcSubscriptionHomePointSetStatus;

typedef uint8_t T_GduFcSubscriptionGpsSignalLevel;

typedef int16_t T_GduFcSubscriptionRtkYaw;

typedef uint8_t T_GduFcSubscriptionRtkPositionInfo;

typedef uint8_t T_GduFcSubscriptionRtkYawInfo;

typedef enum E_GduFcSubscriptionTopic

typedef enum {

```
/*!
 * Quaternion of aircraft topic name. Quaternion topic provides aircraft body frame (FRD) to ground frame
 * (NED) rotation. Please refer to ::T_GduFcSubscriptionQuaternion for information about data structure.
 * @details The GDU quaternion follows Hamilton convention (q0 = w, q1 = x, q2 = y, q3 = z).
 * | Angle | Unit | Accuracy | Notes |
   |--------------|------|------------|-----------------------------------------------|
   | pitch, roll | deg | <1 | in NON-AHRS mode |
   | yaw | deg | <3 | in well-calibrated compass with fine aligned |
   | yaw with rtk | deg | around 1.2 | in RTK heading fixed mode with 1 meter baseline |
 * 数据结构： T_GduFcSubscriptionQuaternion
 */
GDU_FC_SUBSCRIPTION_TOPIC_QUATERNION = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 0),
/*!
 * @brief Provides aircraft's acceleration w.r.t a ground-fixed \b NEU frame @ up to 200Hz
 * @warning Please note that this data is not in a conventional right-handed frame of reference.
 * @details This is a fusion output from the flight control system. The output is in a right-handed NED frame, but the
 * sign of the Z-axis acceleration is flipped before publishing to this topic. So if you are looking to get acceleration
 * in an NED frame, simply flip the sign of the z-axis value. Beyond that, you can convert using rotations to
 * any right-handed frame of reference.
 * @units m/s<SUP>2</SUP>
 * 数据结构： T_GduFcSubscriptionAccelerationGround
 */
GDU_FC_SUBSCRIPTION_TOPIC_ACCELERATION_GROUND = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 1),
/*!
 * @brief Provides aircraft's acceleration w.r.t a body-fixed \b FRU frame @ up to 200Hz
 * @warning Please note that this data is not in a conventional right-handed frame of reference.
 * @details This is a fusion output from the flight control system.
 * @units m/s<SUP>2</SUP>
 * 数据结构： T_GduFcSubscriptionAccelerationBody
 */
GDU_FC_SUBSCRIPTION_TOPIC_ACCELERATION_BODY = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 2),
/*!
 * @brief Provides aircraft's acceleration in an IMU-centered, body-fixed \b FRD frame @ up to 400Hz
 * @details This is a filtered output from the IMU on board the flight control system.
 * @sensors IMU
 * @units m/s<SUP>2</SUP>
```

```
     * 数据结构： T_GduFcSubscriptionAccelerationRaw
     */
    GDU_FC_SUBSCRIPTION_TOPIC_ACCELERATION_RAW                                  =
GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 3),
    /*!
     * @brief Velocity of aircraft topic name. Velocity topic provides aircraft's velocity in a ground-fixed
NEU frame.
     * Please refer to ::T_GduFcSubscriptionVelocity for information about data structure.
     * @warning Please note that this data is not in a conventional right-handed frame of reference.
     * @details This velocity data is a fusion output from the aircraft. Original output is in a right-handed
NED frame, but the
     * sign of the Z-axis velocity is flipped before publishing to this topic. So if you are looking to get velocity
     * in an NED frame, simply flip the sign of the z-axis value. Beyond that, you can convert using rotations
to
     * any right-handed frame of reference.
     *         |         Axis                                  |         Unit         |         Accuracy
|
        |----------|------|-----------------------------------------------------------------------------------------|
        | vgx, vgy | m/s   | Around 5cm/s for GNSS navigation. Around 3cm/s with VO at 1 meter height
|
        | vgz         | m/s   | 10cm/s only with barometer in steady air. 3cm/s with VO at 1 meter height
with 8cm baseline |
     * 数据结构： T_GduFcSubscriptionVelocity
     */
    GDU_FC_SUBSCRIPTION_TOPIC_VELOCITY                                          =
GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 4),
    /*!
     * @brief Provides aircraft's angular velocity in a ground-fixed \b NED frame @ up to 200Hz
     * @details This is a fusion output from the flight control system.
     * @units rad/s
     * 数据结构： T_GduFcSubscriptionAngularRateFusioned
     */
    GDU_FC_SUBSCRIPTION_TOPIC_ANGULAR_RATE_FUSIONED                             =
GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 5),
    /*!
     * @brief Provides aircraft's angular velocity in an IMU-centered, body-fixed \b FRD frame @ up to
400Hz
     * @details This is a filtered output from the IMU on board the flight control system.
     * @sensors IMU
     * @units rad/s
     * 数据结构： T_GduFcSubscriptionAngularRateRaw
     */
    GDU_FC_SUBSCRIPTION_TOPIC_ANGULAR_RATE_RAW                                  =
GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 6),
    /*!
```

```
     * @brief Fused altitude of aircraft topic name. Fused altitude topic provides aircraft's fused altitude from sea
     * level. Please refer to ::T_GduFcSubscriptionAltitudeFused for information about data structure.
     * @units m
     * 数据结构： T_GduFcSubscriptionAltitudeFused
     */
    GDU_FC_SUBSCRIPTION_TOPIC_ALTITUDE_FUSED = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 7),
    /*!
     * @brief Provides aircraft's pressure altitude from sea level using the ICAO model @ up to 200Hz
     * @details
     * This is a filetered output from the barometer without any further fusion.
     *
     * The ICAO model gives an MSL altitude of 1013.25mBar at 15&deg; C and a temperature lapse rate of -6.5&deg; C
     * per 1000m. In your case, it may be possible that the take off altitude of the aircraft is recording a higher pressure
     * than 1013.25mBar. Let's take an example - a weather station shows that SFO (San Francisco International Airport) had
     * recently recorded a pressure of 1027.1mBar. SFO is 4m above MSL, yet, if you calculate the Pressure Altitude using
     * the ICAO model, it relates to -114m. You can use an online calculator to similarly calculate the Pressure Altitude
     * in your area.
     *
     * Another factor that may affect your altitude reading is manufacturing differences in the barometer - it is not
     * uncommon to have a variation of &plusmn;30m readings at the same physical location with two different aircraft. For a given
     * aircraft, these readings will be consistent, so you will need to calibrate the offset of your system if your code
     * relies on the accuracy of the absolute value of altitude.
     * @sensors GPS, Barometer, IMU
     * @units m
     * 数据结构： T_GduFcSubscriptionAltitudeBarometer
     */
    GDU_FC_SUBSCRIPTION_TOPIC_ALTITUDE_BAROMETER = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 8),
    /*!
     * @brief Provides the altitude from sea level when the aircraft last took off.
     * @details
     * This is a fusion output from the flight control system, and also uses the ICAO model.
     *
     * The ICAO model gives an MSL altitude of 1013.25mBar at 15&deg; C and a temperature lapse rate of -6.5&deg; C
```

```
     * per 1000m. In your case, it may be possible that the take off altitude of the aircraft is recording a higher pressure
     * than 1013.25mBar. Let's take an example - a weather station shows that SFO (San Francisco International Airport) had
     * recently recorded a pressure of 1027.1mBar. SFO is 4m above MSL, yet, if you calculate the Pressure Altitude using
     * the ICAO model, it relates to -114m. You can use an online calculator to similarly calculate the Pressure Altitude
     * in your area.
     *
     * Another factor that may affect your altitude reading is manufacturing differences in the barometer - it is not
     * uncommon to have a variation of &plusmn;30m readings at the same physical location with two different aircraft. For a given
     * aircraft, these readings will be consistent, so you will need to calibrate the offset of your system if your code
     * relies on the accuracy of the absolute value of altitude.
     *
     * @note This value is updated each time the drone takes off.
     *
     * @sensors Visual Odometry (M210 only), Barometer, IMU
     * @units m
     * 数据结构： T_GduFcSubscriptionAltitudeOfHomePoint
     */
    GDU_FC_SUBSCRIPTION_TOPIC_ALTITUDE_OF_HOMEPOINT = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 9),
    /*!
     * @brief Provides the relative height above ground at up to 100Hz.
     * @details
     * This is a fusion output from the flight control system. The height is a direct estimate of the closest large object below the aircraft's ultrasonic sensors.
     * A large object is something that covers the ultrasonic sensor for an extended duration of time.
     *
     * @warning This topic does not come with a 'valid' flag - so if the aircraft is too far from an object for the
     * ultrasonic sensors/VO to provide any meaningful data, the values will latch and there is no way for user code to
     * determine if the data is valid or not. Use with caution.
     * @sensors Visual Odometry, Ultrasonic
     * @units m
     * 数据结构： T_GduFcSubscriptionHeightFusion
     */
    GDU_FC_SUBSCRIPTION_TOPIC_HEIGHT_FUSION = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 10),
    /*!
```

```
     * @brief Relative height above ground of aircraft topic name. Please refer to
     * ::T_GduFcSubscriptionHeightRelative for information about data structure.
     * @details This data is a fusion output from aircraft. The height is a direct estimate of the closest large object
     * below the aircraft's ultrasonic sensors.
     * @warning This topic does not come with a 'valid' flag - so if the aircraft is too far from an object for the
     * ultrasonic sensors/VO to provide any meaningful data, the values will latch and there is no way for user to
     * determine if the data is valid or not. Please use with caution.
     * 数据结构： T_GduFcSubscriptionHeightRelative
     */
    GDU_FC_SUBSCRIPTION_TOPIC_HEIGHT_RELATIVE = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 11),

    /*!
     * Fused position of aircraft topic name. Please refer to ::T_GduFcSubscriptionPositionFused for information
     * about data structure.
     * @warning Please note that if GPS signal is weak (low visibleSatelliteNumber, see below), the
     * latitude/longitude values won't be updated but the altitude might still be. There is currently no way to know if
     * the lat/lon update is healthy.
     * @details The most important component of this topic is the T_GduFcSubscriptionPositionFused::visibleSatelliteNumber.
     * Use this to track your GPS satellite coverage and build some heuristics for when you might expect to lose GPS updates.
     *     | Axis | Unit | Position Sensor | Accuracy                                                |
         |------|------|-----------------|---------------------------------------------------|
         | x, y | m    | GPS             | <3m with open sky without multipath               |
         | z    | m    | GPS             | <5m with open sky without multipath               |
     * 数据结构： T_GduFcSubscriptionPositionFused
     */
    GDU_FC_SUBSCRIPTION_TOPIC_POSITION_FUSED = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 12),

    /*!
     * @brief GPS date topic name. Please refer to ::T_GduFcSubscriptionGpsDate for information about data structure.
     * 数据结构： T_GduFcSubscriptionGpsDate
     */
    GDU_FC_SUBSCRIPTION_TOPIC_GPS_DATE = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 13),

    /*!
```

```
     * @brief GPS time topic name. Please refer to ::T_GduFcSubscriptionGpsTime for information about data structure.
     * 数据结构： T_GduFcSubscriptionGpsTime
     */
    GDU_FC_SUBSCRIPTION_TOPIC_GPS_TIME                                  = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 14),

    /*!
     * @brief GPS position topic name. Please refer to ::T_GduFcSubscriptionGpsPosition for information about data structure.
     * @details
     *     | Axis | Accuracy                                                |
          |------|---------------------------------------------------|
          | x, y | <3m with open sky without multipath                |
          | z    | <5m with open sky without multipath                |
     * 数据结构： T_GduFcSubscriptionGpsPosition
     */
    GDU_FC_SUBSCRIPTION_TOPIC_GPS_POSITION                              = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 15),

    /*!
     * @brief GPS velocity topic name. Please refer to ::T_GduFcSubscriptionGpsVelocity for information about data structure.
     * 数据结构： T_GduFcSubscriptionGpsVelocity
     */
    GDU_FC_SUBSCRIPTION_TOPIC_GPS_VELOCITY                              = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 16),

    /*!
     * @brief GPS details topic name. GPS details topic provides GPS state and other detail information. Please refer
     * to ::T_GduFcSubscriptionGpsDetail for information about data structure.
     * 数据结构： T_GduFcSubscriptionGpsDetails
     */
    GDU_FC_SUBSCRIPTION_TOPIC_GPS_DETAILS                               = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 17),

    /*!
     * @brief GPS signal level topic name. This topic provides a measure of the quality of GPS signal. Please refer to
     * ::T_GduFcSubscriptionGpsSignalLevel for information about data structure.
     * 数据结构： T_GduFcSubscriptionGpsSignalLevel
     */
    GDU_FC_SUBSCRIPTION_TOPIC_GPS_SIGNAL_LEVEL                          = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 18),
```

```
    /*!
     * @brief RTK position topic name. Please refer to ::T_GduFcSubscriptionRtkPosition for information about data structure.
     * @details
     *     | Axis | Accuracy                                                |
           |------|---------------------------------------------------|
           | x, y | ~2cm with fine alignment and fix condition          |
           | z    | ~3cm with fine alignment and fix condition          |
     * 数据结构：  T_GduFcSubscriptionRtkPosition
     */
    GDU_FC_SUBSCRIPTION_TOPIC_RTK_POSITION                                  = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 19),

    /*!
     * @brief RTK velocity topic name. Please refer to ::T_GduFcSubscriptionRtkVelocity for information about data structure.
     * 数据结构：  T_GduFcSubscriptionRtkVelocity
     */
    GDU_FC_SUBSCRIPTION_TOPIC_RTK_VELOCITY                                  = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 20),

    /*!
     * @brief RTK yaw topic name. Please refer to ::T_GduFcSubscriptionRtkYaw for information about data structure.
     * @details The RTK yaw will provide the vector from ANT1 to ANT2 as configured in GDU Assistant 2. This
     * means that the value of RTK yaw will be 90deg offset from the yaw of the aircraft.
     * 数据结构：  T_GduFcSubscriptionRtkYaw
     */
    GDU_FC_SUBSCRIPTION_TOPIC_RTK_YAW                                       = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 21),

    /*!
     * @brief RTK position information topic name. RTK position information topic provides a state of RTK position
     * solution. Please refer to ::T_GduFcSubscriptionRtkPositionInfo for information about data structure.
     * 数据结构：  T_GduFcSubscriptionRtkPositionInfo
     */
    GDU_FC_SUBSCRIPTION_TOPIC_RTK_POSITION_INFO                             = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 22),

    /*!
     * @brief RTK yaw topic name. RTK yaw information topic provides a state of RTK yaw solution. Please refer to
```

```
     * ::T_GduFcSubscriptionRtkYawInfo for information about data structure.
     * 数据结构： T_GduFcSubscriptionRtkYawInfo
     */
    GDU_FC_SUBSCRIPTION_TOPIC_RTK_YAW_INFO = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 23),
    /*!
     * @brief Provides aircraft's magnetometer reading, fused with IMU and GPS @ up to 100Hz
     * @details This reading is the magnetic field recorded by the magnetometer in x,y,z axis, calibrated such that
     * 1000 < |m| < 2000, and fused with IMU and GPS for robustness
     * @sensors Magnetometer, IMU, GPS
     * @units N/A
     * 数据结构： T_GduFcSubscriptionCompass
     */
    GDU_FC_SUBSCRIPTION_TOPIC_COMPASS = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 24),
    /*!
     * @brief Provides remote controller stick inputs @ up to 100Hz
     * @details This topic will give you:
     * - Stick inputs (R,P,Y,Thr)
     * - Mode switch (P/A/F)
     * - Landing gear switch (Up/Down)
     *
     * 数据结构： T_GduFcSubscriptionRC
     * @also \ref TOPIC_RC_WITH_FLAG_DATA
     */
    GDU_FC_SUBSCRIPTION_TOPIC_RC = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 25),
    /*!
     * @brief Provides gimbal pitch, roll, yaw @ up to 50Hz
     * @details
     * The reference frame for gimbal angles is a NED frame attached to the gimbal.
     * This topic uses a data structure, Vector3f, that is too generic for the topic. The order of angles is :
     * |Data Structure Element| Meaning|
     * |----------------------|--------|
     * |Vector3f.x            |pitch   |
     * |Vector3f.y            |roll    |
     * |Vector3f.z            |yaw     |
     *
     * @perf
     * 0.1 deg accuracy in all axes
     *
     * @sensors Gimbal Encoder, IMU, Magnetometer
     * @units deg
     * 数据结构： T_GduFcSubscriptionGimbalAngles
```

```
     * @also \ref TOPIC_GIMBAL_STATUS, \ref TOPIC_GIMBAL_CONTROL_MODE
     */
    GDU_FC_SUBSCRIPTION_TOPIC_GIMBAL_ANGLES                                    =
GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 26),
    /*!
     * @brief Provides gimbal status and error codes @ up to 50Hz
     * @details Please see the \ref GimbalStatus struct for the details on what data you can receive.
     *
     * 数据结构： T_GduFcSubscriptionGimbalStatus
     * @also \ref TOPIC_GIMBAL_ANGLES, \ref TOPIC_GIMBAL_CONTROL_MODE
     */
    GDU_FC_SUBSCRIPTION_TOPIC_GIMBAL_STATUS                                    =
GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 27),
    /*!
     * @brief Flight status topic name. Please refer to ::T_GduFcSubscriptionFlightStatus for information
about data structure.
     * 数据结构： T_GduFcSubscriptionFlightStatus
     */
    GDU_FC_SUBSCRIPTION_TOPIC_STATUS_FLIGHT                                    =
GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 28),
    /*!
     * @brief Provides a granular state representation for various tasks/flight modes @ up to 50Hz
     * @details Typically, use this topic together with \ref TOPIC_STATUS_FLIGHT to get a
     * better understanding of the overall status of the aircraft.
     *
     * 数据结构： T_GduFcSubscriptionDisplaymode
     * @also \ref TOPIC_STATUS_FLIGHT
     */
    GDU_FC_SUBSCRIPTION_TOPIC_STATUS_DISPLAYMODE                               =
GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 29),
    /*!
     * @brief Provides status for the landing gear state @ up to 50Hz
     *
     * 数据结构： T_GduFcSubscriptionLandinggear
     */
    GDU_FC_SUBSCRIPTION_TOPIC_STATUS_LANDINGGEAR                               =
GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 30),
    /*!
     * @brief If motors failed to start, this topic provides reasons why. Available @ up to 50Hz
     * 数据结构： T_GduFcSubscriptionMotorStartError
     * \note These enumerations show up in the ErrorCode class because they can also be returned as
acknowledgements
     * for APIs that start the motors, such as \ref Control::takeoff "Takeoff" or \ref Control::armMotors
"Arm"
     */
```

```
    GDU_FC_SUBSCRIPTION_TOPIC_STATUS_MOTOR_START_ERROR                                =
GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC,

31),
    /*!
     * @brief Battery information topic name. Please refer to ::T_GduFcSubscriptionBatteryInfo for
information about data structure.
     * 数据结构： T_GduFcSubscriptionBatteryInfo
     */
    GDU_FC_SUBSCRIPTION_TOPIC_BATTERY_INFO                                           =
GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 32),
    /*!
     * @brief Provides states of the aircraft related to SDK/RC control
     * @details The following information is available in this topic:
     * |Data Structure Element| Meaning|
     * |----------------------|--------|
     * |controlMode           |The modes in which the aircraft is being controlled (control loops being
applied on horizontal, vertical and yaw axes of the aircraft)|
     * |deviceStatus          |Which device is controlling the motion of the aircraft: RC (Manual
control), MSDK (Missions kicked off through mobile), OSDK (Missions kicked off through onboard/ low-level
flight control)      |
     * |flightStatus          |Has the OSDK been granted control authority? Since MSDK and RC have
precedence, it is possible that deviceStatus shows RC or MSDK actually controlling the aircraft but this value
is 1.      |
     * |vrcStatus             |Deprecated|
     * 数据结构： T_GduFcSubscriptionControlDevice
     */
    GDU_FC_SUBSCRIPTION_TOPIC_CONTROL_DEVICE                                         =
GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 33),
    /*!
     * @brief Provides IMU and quaternion data time-synced with a hardware clock signal @ up to 400Hz.
     * @details This is the only data which can be synchronized with external software or hardware
systems. If you want to
     * fuse an external sensor's data with the aircraft's IMU, this data along with a hardware trigger from
the A3/N3's
     * expansion ports is how you would do it. You can see detailed documentation on how this process
works in the [Hardware
     * Sync Guide](https://developer.GDU.com/onboard-sdk/documentation/guides/component-guide-
hardware-sync.html).
     * @sensors IMU, sensor fusion output
     * @units
     * |Data Structure Element| Units|
     * |----------------------|--------|
     * |Timestamp |2.5ms, 1ns (See \ref SyncTimestamp)|
     * |Quaternion |rad (after converting to rotation matrix)|
```

```
     * |Acceleration |g|
     * |Gyroscope |rad/sec|
     * 数据结构： T_GduFcSubscriptionHardSync
     */
    GDU_FC_SUBSCRIPTION_TOPIC_HARD_SYNC = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 34),
    /*!
     * @brief Provides a measure of the quality of GPS signal, with a mechanism for guarding against unset homepoint @ up to 50Hz
     * @details The level varies from 0 to 5, with 0 being the worst and 5 the best GPS signal. The key difference between
     * this and TOPIC_GPS_SIGNAL_LEVEL is that this topic always returns 0 if the homepoint is not set. Once the home point is
     * set, the behavior is exactly the same as TOPIC_GPS_SIGNAL_LEVEL.
     * @sensors GPS
     * 数据结构： T_GduFcSubscriptionGpsControlLevel
     * @also \ref TOPIC_GPS_SIGNAL_LEVEL
     */
    GDU_FC_SUBSCRIPTION_TOPIC_GPS_CONTROL_LEVEL = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 35),
    /*!
     * @brief Provides normalized remote controller stick input data, along with connection status @ up to 50Hz
     * @note This topic was added in August 2018. Your aircraft may require a FW update to enable this feature.
     * @details This topic will give you:
     * - Stick inputs (R,P,Y,Thr)
     * - Mode switch (P/A/F)
     * - Landing gear switch (Up/Down)
     * - Connection status for air system, ground system and MSDK apps. The connection status also includes a
     * logicConnected element, which will change to false if either the air system or the ground system radios
     * are disconnected for >3s.
     * - Deadzones near the center of the stick positions are also handled in this topic.
     *
     * 数据结构： T_GduFcSubscriptionRCWithFlagData
     * @also \ref TOPIC_RC
     */
    GDU_FC_SUBSCRIPTION_TOPIC_RC_WITH_FLAG_DATA = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 36),
    /*!
     * @brief Provides raw data from the EscS @ up to 50Hz
     * @note This topic was added in August 2018. Your aircraft may require a FW update to enable this feature.
```

```
        * @details This topic supports reporting data for up to 8 EscS; note that only GDU Intelligent EscS are supported
        * for this reporting feature. Use this topic to get data on elements close to the hardware - e.g. motor speeds,
        * ESC current and voltage, error flags at the ESC level etc.
        * 数据结构： T_GduFcSubscriptionEscData
        */
    GDU_FC_SUBSCRIPTION_TOPIC_ESC_DATA = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 37),
    /*!
        * @brief Provides RTK connection status @ up to 50Hz
        * @note This topic was added in August 2018. Your aircraft may require a FW update to enable this feature.
        * @details This topic will update in real time whether the RTK GPS system is connected or not; typical uses
        * include app-level logic to switch between GPS and RTK sources of positioning based on this flag.
        * 数据结构： T_GduFcSubscriptionRTKConnectStatus
        */
    GDU_FC_SUBSCRIPTION_TOPIC_RTK_CONNECT_STATUS = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 38),
    /*!
        * @brief Provides the mode in which the gimbal will interpret control commands @ up to 50Hz
        * @note This topic was added in August 2018. Your aircraft may require a FW update to enable this feature.
        * @details This topic will report the current control mode which can be set in the
        * GDU Go app, MSDK apps, or through Onboard SDK gimbal control APIs (see \ref Gimbal::AngleData "AngleData" struct
        * for more information)
        * 数据结构： T_GduFcSubscriptionGimbalControlMode
        */
    GDU_FC_SUBSCRIPTION_TOPIC_GIMBAL_CONTROL_MODE = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 39),
    /*!
        * @brief Provides a number of flags which report different errors the aircraft may encounter in flight @ up to 50Hz
        * @note This topic was added in August 2018. Your aircraft may require a FW update to enable this feature.
        * @warning Most of the errors reported by this topic are cases where immediate action is required; you can use these
        * as a baseline for implementing safety-related error-handling routines.
        * 数据结构： T_GduFcSubscriptionFlightAnomaly
        */
    GDU_FC_SUBSCRIPTION_TOPIC_FLIGHT_ANOMALY = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 40),
    /*!
```

* @brief Provides aircraft's position in a Cartesian frame @ up to 50Hz, without the need for GPS

* @warning This topic does not follow a standard co-ordinate convention. Please read the details below for usage.

* @details This is the only topic which can provide positioning information without having a GPS fix; though this

* can be a big enabler please note the caveats of using this topic:

* - The topic will use an origin that does not have a global reference, and is not published to the SDK.

* - The topic uses a combination of VO and compass heading to identify the X-Y axes of its reference frame. This means

* that if your compass performance is not good in an environment, there is no guarantee the X-Y axes will point to

* North and East.

* - The actual directions of the X-Y axes are currently not published to the SDK.

* - If during a flight the compass performance were to change dramatically, the orientation of the X-Y axes may change

* to re-align with North-East. The aircraft's position in X and Y may exhibit discontinuities in these cases.

* - The reference frame is referred to as the Navigation Frame - Cartesian X,Y axes aligned with N,E directions on a best-effort

* basis, and Z aligned to D (down) direction.

* - A health flag for each axis provides some granularity on whether this data is valid or not.

*

* The key takeaway from these details is that this topic provides a best-effort attempt at providing position

* information in the absence of absolute references (GPS, compass etc.), without guarantees of consistency if

* environmental conditions change. So if your application is confined to a stable environment, or if you will

* have GPS and compass available at all times, this topic can still provide useful data that cannot be otherwise

* had. If using for control, make sure to have guards checking for the continuity of data.

*

* @note Since this topic relies on visual features and/or GPS, if your environment does not provide any of these

* sources of data, the quality of this topic will reduce significantly. VO data quality will reduce if you are too high

* above the ground. Make sure that the Vision Positioning System is enabled in GDU Go 4 before using this topic

* (by default it is enabled).

* @sensors IMU, VO, GPS(if available), RTK (if available), ultrasonic, magnetometer, barometer

* @units m

* 数据结构： T_GduFcSubscriptionPositionVO

*/

GDU_FC_SUBSCRIPTION_TOPIC_POSITION_VO = GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 41),

```
    /*!
     * @brief Provides obstacle info around the vehicle @ up to 100Hz
     * 数据结构： T_GduFcSubscriptionAvoidData
     */
    GDU_FC_SUBSCRIPTION_TOPIC_AVOID_DATA                                    =
GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 42),
    /*!
     * @brief Provides status of whether the home point was set or not
     * 数据结构： T_GduFcSubscriptionHomePointSetStatus
     */
    GDU_FC_SUBSCRIPTION_TOPIC_HOME_POINT_SET_STATUS                         =
GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 43),
    /*!
     * @brief Provides homepoint information, the valid of the home point infomation can ref to the
     * topic GDU_FC_SUBSCRIPTION_TOPIC_HOME_POINT_SET_STATUS
     * 数据结构： T_GduFcSubscriptionHomePointInfo
     */
    GDU_FC_SUBSCRIPTION_TOPIC_HOME_POINT_INFO                               =
GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 44),
    /*!
     * @brief Provides three gimbal information, used for S400
     * 数据结构： T_GduFcSubscriptionThreeGimbalData
     */
    GDU_FC_SUBSCRIPTION_TOPIC_THREE_GIMBAL_DATA                             =
GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC, 45),

    /*!
     * @brief Battery information topic name. Please refer to ::T_GduFcSubscriptionSingleBatteryInfo for
information about data structure.
     * 数据结构： T_GduFcSubscriptionSingleBatteryInfo
     */
    GDU_FC_SUBSCRIPTION_TOPIC_BATTERY_SINGLE_INFO_INDEX1                    =
GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC,

46),
    /*!
     * @brief Battery information topic name. Please refer to ::T_GduFcSubscriptionSingleBatteryInfo for
information about data structure.
     * 数据结构： T_GduFcSubscriptionSingleBatteryInfo
     */
    GDU_FC_SUBSCRIPTION_TOPIC_BATTERY_SINGLE_INFO_INDEX2                    =
GDU_DATA_SUBSCRIPTION_TOPIC(GDU_DATA_SUBSCRIPTION_MODULE_FC,

47),
```

```
    /*! Total number of topics that can be subscribed。  */
    GDU_FC_SUBSCRIPTION_TOPIC_TOTAL_NUMBER,
} E_GduFcSubscriptionTopic;
```

typedef

enum E_GduFcSubscriptionDataHealthFlag

订阅数据的健康状态

```
typedef enum {
    GDU_FC_SUBSCRIPTION_DATA_NOT_HEALTH = 0, 订阅数据是健康的，订阅数据可用
    GDU_FC_SUBSCRIPTION_DATA_HEALTH = 1,    订阅数据是不健康的，订阅数据不可用
} E_GduFcSubscriptionDataHealthFlag;
```

typedef

enum E_GduFcSubscriptionPositionSolutionProperty

位置解属性

```
typedef enum {
    GDU_FC_SUBSCRIPTION_POSITION_SOLUTION_PROPERTY_NOT_AVAILABLE = 0,                位置解数据不可用
    GDU_FC_SUBSCRIPTION_POSITION_SOLUTION_PROPERTY_FIX_POSITION = 1,                 位置由 FIX POSITION 命令指定
    GDU_FC_SUBSCRIPTION_POSITION_SOLUTION_PROPERTY_FIX_HEIGHT_AUTO = 2,              位置由 FIX HEIGHT / AUTO 命令指定

GDU_FC_SUBSCRIPTION_POSITION_SOLUTION_PROPERTY_INSTANTANEOUS_DOPPLER_COMPUTE_VELOCITY = 8,   速度由即时多普勒信息导出
    GDU_FC_SUBSCRIPTION_POSITION_SOLUTION_PROPERTY_SINGLE_PNT_SOLUTION = 16,        单点位置解
```

GDU_FC_SUBSCRIPTION_POSITION_SOLUTION_PROPERTY_PSEUDORANGE_DIFFERENTIAL_SOLUTION = 17, 伪距差分解

GDU_FC_SUBSCRIPTION_POSITION_SOLUTION_PROPERTY_SBAS_CORRECTION_CALCULATED = 18, SBAS 定位

GDU_FC_SUBSCRIPTION_POSITION_SOLUTION_PROPERTY_KALMAN_FILTER_WITHOUT_OBSERVATION_PROPAGATED = 19, 由一个没有新观测值的卡尔曼滤波器输出

GDU_FC_SUBSCRIPTION_POSITION_SOLUTION_PROPERTY_OMNISTAR_VBS_POSITION = 20, OmniSTAR VBS 位置 (L1 sub-metre)

GDU_FC_SUBSCRIPTION_POSITION_SOLUTION_PROPERTY_FLOAT_L1_AMBIGUITY = 32, L1 浮点解

GDU_FC_SUBSCRIPTION_POSITION_SOLUTION_PROPERTY_FLOAT_IONOSPHERIC_FREE_AMBIGUITY = 33, 消电离层浮点解

GDU_FC_SUBSCRIPTION_POSITION_SOLUTION_PROPERTY_FLOAT_SOLUTION = 34, 窄巷浮点解

GDU_FC_SUBSCRIPTION_POSITION_SOLUTION_PROPERTY_L1_AMBIGUITY_INT = 48, L1 固定解

GDU_FC_SUBSCRIPTION_POSITION_SOLUTION_PROPERTY_WIDE_LANE_AMBIGUITY_INT = 49, 宽巷固定解

GDU_FC_SUBSCRIPTION_POSITION_SOLUTION_PROPERTY_NARROW_INT = 50, 窄巷固定解

} E_GduFcSubscriptionPositionSolutionProperty;

typedef enum E_GduFcSubscriptionGpsFixState

GPS 定位状态

typedef enum {

GDU_FC_SUBSCRIPTION_GPS_FIX_STATE_NO_FIX = 0, GPS 未获取到定位信息

GDU_FC_SUBSCRIPTION_GPS_FIX_STATE_DEAD_RECKONING_ONLY = 1, GPS 推算得到 GPS 位置

GDU_FC_SUBSCRIPTION_GPS_FIX_STATE_2D_FIX = 2, 包含纬度/经度（或 X / Y）的水平位置已定位

GDU_FC_SUBSCRIPTION_GPS_FIX_STATE_3D_FIX = 3, 包含纬度/经度/海拔（或 X / Y / Z）的水平和垂直位置已定位

GDU_FC_SUBSCRIPTION_GPS_FIX_STATE_GPS_PLUS_DEAD_RECKONING = 4, GPS 推测计算位置

GDU_FC_SUBSCRIPTION_GPS_FIX_STATE_TIME_ONLY_FIX = 5, 仅时间信息有效

} E_GduFcSubscriptionGpsFixState;

typedef enum E_GduFcSubscriptionFlightStatus

无人机飞行状态

```
typedef enum {
    GDU_FC_SUBSCRIPTION_FLIGHT_STATUS_STOPED = 0,            无人机在地面上且电机静止
    GDU_FC_SUBSCRIPTION_FLIGHT_STATUS_ON_GROUND = 1,         无人机在地面上，但电机仍在转动
    GDU_FC_SUBSCRIPTION_FLIGHT_STATUS_IN_AIR = 2,            无人机在空中
} E_GduFcSubscriptionFlightStatus;
```

typedef enum E_GduFcSubscriptionDisplayMode

```
typedef enum {
    /*! 此模式需要用户手动控制飞行器在空中保持稳定。 */
    GDU_FC_SUBSCRIPTION_DISPLAY_MODE_MANUAL_CTRL = 0,
    /*! 此模式，飞行器可以保持姿态稳定并仅使用气压计定位来控制姿态。飞行器不能自动定位和稳定悬停。*/
    GDU_FC_SUBSCRIPTION_DISPLAY_MODE_ATTITUDE = 1,
    GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_2 = 2,
    GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_3 = 3,
    GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_4 = 4,
    GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_5 = 5,
    /*! 飞行器在普通 GPS 模式。在此模式，飞行器可以自动定位并稳定悬停。飞行器对于命令的反应速度中等。*/
    GDU_FC_SUBSCRIPTION_DISPLAY_MODE_P_GPS = 6,
    GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_7 = 7,
    GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_8 = 8,
    /*! 在热点模式。 */
    GDU_FC_SUBSCRIPTION_DISPLAY_MODE_HOTPOINT_MODE = 9,
    /*! 在此模式，用户可以推动油门杆以完成稳定降落。*/
    GDU_FC_SUBSCRIPTION_DISPLAY_MODE_ASSISTED_TAKEOFF = 10,
    /*! 在此模式，飞行器可以自动启动电机，爬升并最终悬停。*/
    GDU_FC_SUBSCRIPTION_DISPLAY_MODE_AUTO_TAKEOFF = 11,
    /*! 在此模式，飞行器可以自动降落。 */
    GDU_FC_SUBSCRIPTION_DISPLAY_MODE_AUTO_LANDING = 12,
    GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_13 = 13,
    GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_14 = 14,
    /*! 在此模式，飞行器可以自动返回前一个返航点。有三种模式：智能返航、低电量返航以及故
```

障返航。 */
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_NAVI_GO_HOME = 15,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_16 = 16,
/*! 在此模式，飞行器由 SDK API 控制。用户可以直接定义水平方向和垂直方向的的控制模式并向飞行棋发送数据。*/
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_NAVI_SDK_CTRL = 17,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_18 = 18,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_19 = 19,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_20 = 20,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_21 = 21,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_22 = 22,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_23 = 23,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_24 = 24,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_25 = 25,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_26 = 26,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_27 = 27,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_28 = 28,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_29 = 29,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_30 = 30,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_31 = 31,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_32 = 32,
/*! 飞行器迫降，可能是由于低电量。*/
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_FORCE_AUTO_LANDING = 33,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_34 = 34,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_35 = 35,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_36 = 36,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_37 = 37,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_38 = 38,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_39 = 39,
/*! 飞行器会搜寻当遥控器没有断连的前一个位置。*/
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_SEARCH_MODE = 40,
/*! 点击启动的模式。每当用户解锁电机，这将为第一个模式。 */
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_ENGINE_START = 41,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_42 = 42,
GDU_FC_SUBSCRIPTION_DISPLAY_MODE_RESERVED_43 = 42,
} E_GduFcSubscriptionDisplayMode;

typedef

enum E_GduFcSubscriptionHomePointSetStatus

返航点设置状态

```
typedef enum {
    GDU_FC_SUBSCRIPTION_HOME_POINT_SET_STATUS_FAILED = 0,  未设置
    GDU_FC_SUBSCRIPTION_HOME_POINT_SET_STATUS_SUCCESS = 1,  成功设置
} E_GduFcSubscriptionHomePointSetStatus;
```

typedef struct GT_GduFcSubscriptionQuaternion

四元数

```
typedef struct Quaternion {
    gdu_f32_t q0;              w,使用弧度制  (转换为旋转矩阵或欧拉角)
    gdu_f32_t q1;              x,使用弧度制  (转换为旋转矩阵或欧拉角)
    gdu_f32_t q2;              y,使用弧度制  (转换为旋转矩阵或欧拉角)
    gdu_f32_t q3;              z,使用弧度制  (转换为旋转矩阵或欧拉角)
} T_GduFcSubscriptionQuaternion;
```

typedef

struct T_GduFcSubscriptionAccelerationGround

```
typedef T_GduVector3f T_GduFcSubscriptionAccelerationGround;
```

typedef

struct T_GduFcSubscriptionAccelerationBody

```
typedef T_GduVector3f T_GduFcSubscriptionAccelerationBody;
```

typedef

struct T_GduFcSubscriptionAccelerationRaw

typedef T_GduVector3f T_GduFcSubscriptionAccelerationRaw;

typedef struct T_GduFcSubscriptionVelocity

```
typedef struct Velocity {
    T_GduVector3f data;      速度数值
    uint8_t health : 1;      健康状态，该值可为：E_GduFcSubscriptionDataHealthFlag 中的任意值
    uint8_t reserve : 7;     保留
} T_GduFcSubscriptionVelocity;
```

typedef

struct T_GduFcSubscriptionAngularRateFusioned

typedef T_GduVector3f T_GduFcSubscriptionAngularRateFusioned;

typedef

struct T_GduFcSubscriptionAngularRateRaw

typedef T_GduVector3f T_GduFcSubscriptionAngularRateRaw;

typedef struct T_GduFcSubscriptionPositionFused

```
typedef struct PositionFused {
    gdu_f64_t longitude; /*!< 经度，单位: rad。 */
    gdu_f64_t latitude; /*!< 纬度，单位: rad。 */
    gdu_f32_t altitude; /*!< 高度，WGS 84 参考椭球体，单位： m。*/
    uint16_t visibleSatelliteNumber; /*!< 可视的卫星数量。 */
} T_GduFcSubscriptionPositionFused;
```

typedef struct T_GduFcSubscriptionGpsPosition

GPS 位置

```
typedef T_GduVector3d T_GduFcSubscriptionGpsPosition;
```

typedef struct T_GduFcSubscriptionGpsVelocity

GPS 速度

```
typedef T_GduVector3f T_GduFcSubscriptionGpsVelocity;
```

typedef struct T_GduFcSubscriptionGpsDetails

GPS 详情

```
typedef struct GpsDetail {
    gdu_f32_t hdop; /*!< 水平分量精度因子, <1: 理想, 1-2: 优秀, 2-5: 良好, 5-10: 中等, 10-20: 一般, >20: 弱。 */
    gdu_f32_t pdop; /*!< 位置精度因子, <1: 理想, 1-2: 优秀, 2-5: 良好, 5-10: 中等, 10-20: 一般, >20: 弱。 */
    gdu_f32_t fixState; /*!< GPS 修复状态，可以为 E_GduFcSubscriptionGpsFixState 中的任何状态，以外的值是无效的。*/
    gdu_f32_t vacc; /*!< 垂直位置准确性 (mm)，越小越好 。 */
    gdu_f32_t hacc; /*!< 水平位置准确性 (mm)，越小越好 。 */
    gdu_f32_t sacc; /*!< 速度准确性 (cm/s)，越小越好。 */
    uint32_t gpsSatelliteNumberUsed; /*!< 用于修复定位的 GPS 卫星数量 */
    uint32_t glonassSatelliteNumberUsed; /*!< 用于修复定位的 GLONASS 卫星数量 */
    uint16_t totalSatelliteNumberUsed; /*!< 用于修复定位的卫星数量 */
    uint16_t gpsCounter; /*!< 发送 GPS 数据的累计次数 */
```

```
} T_GduFcSubscriptionGpsDetails;
```

typedef struct T_GduFcSubscriptionRtkPosition

```
typedef struct PositionData {
    gdu_f64_t longitude; /*!< 精度，单位： deg。 */
    gdu_f64_t latitude;  /*!< 纬度，单位： deg。 */
    gdu_f32_t hfsl;      /*!< 高于海平面高度，单位： m。 */
} T_GduFcSubscriptionRtkPosition;
```

typedef struct T_GduFcSubscriptionRtkVelocity

```
typedef T_GduVector3f T_GduFcSubscriptionRtkVelocity;
```

typedef struct T_GduFcSubscriptionCompass

用于数据广播的结构体，返回磁力计读数

```
typedef struct Mag {
    int16_t x;
    int16_t y;
    int16_t z;
} T_GduFcSubscriptionCompass;
```

typedef struct T_GduFcSubscriptionRC

数据广播和数据订阅的结构体，返回 RC 读数

```
typedef struct RC {
    int16_t roll;     /*!< [-10000,10000] */
    int16_t pitch;    /*!< [-10000,10000] */
    int16_t yaw;      /*!< [-10000,10000] */
    int16_t throttle; /*!< [-10000,10000] */
    int16_t mode;     /*!< [-10000,10000] */
    /*!< M100 [P: -8000, A: 0, F: 8000] */
```

```
    int16_t gear;          /*!< [-10000,10000] */
    /*!< M100 [Up: -10000, Down: -4545] */
} T_GduFcSubscriptionRC;
```

typedef struct T_GduFcSubscriptionGimbalAngles

```
typedef T_GduVector3f T_GduFcSubscriptionGimbalAngles;
```

typedef struct T_GduFcSubscriptionGimbalStatus

TOPIC_GIMBAL_STATUS 的结构体

```
typedef struct GimbalStatus {
    uint32_t mountStatus: 1; /*!< 1 - 云台安装, 0 - 云台未安装*/
    uint32_t isBusy: 1;
    uint32_t pitchLimited: 1;              /*!< 1 - 轴达极限, 0 - 未达 */
    uint32_t rollLimited: 1;               /*!< 1 - 轴达极限, 0 - 未达 */
    uint32_t yawLimited: 1;                /*!< 1 - 轴达极限, 0 - 未达 */
    uint32_t calibrating: 1;               /*!< 1 - 校准中, 0 - 未校准 */
    uint32_t prevCalibrationgResult: 1; /*!< 1 - 成功, 0 - 失败 */
    uint32_t installedDirection: 1;        /*!< 1 - 为 OSMO 反转, 0 - 普通 */
    uint32_t disabled_mvo: 1;
    uint32_t gear_show_unable: 1;
    uint32_t gyroFalut: 1;           /*!< 1 - 错误, 0 - 正常 */
    uint32_t escPitchStatus: 1;    /*!< 1 - Pitch 数据是正常的, 0 - 错误 */
    uint32_t escRollStatus: 1;     /*!< 1 - Roll 数据是正常的, 0 - 错误 */
    uint32_t escYawStatus: 1;      /*!< 1 - Yaw 数据是正常的, 0 - 错误 */
    uint32_t droneDataRecv: 1;     /*!< 1 - 正常 , 0 - 错误 */
    uint32_t initUnfinished: 1;    /*!< 1 - 初始化完成, 0 - 未完成 */
    uint32_t FWUpdating: 1;          /*!< 1 - 更新中, 0 - 未更新 */
    uint32_t reserved2: 15;
} T_GduFcSubscriptionGimbalStatus;
```

typedef

struct T_GduFcSubscriptionSingleBatteryState

```
typedef struct {
    uint32_t reserved: 12;
    uint32_t cellBreak: 5;                  /*! 0:normal;other:Undervoltage core index(0x01-0x1F)*/
    uint32_t selfCheckError: 3;             /*! enum-type: GDUSmartBatterySelfCheck*/
    uint32_t reserved1: 7;
    uint32_t batteryClosedReason: 5;        /*! enum-type: GDU_BETTERY_CLOSED_REASON*/
    uint8_t reserved2: 6;                   /*! [0]CHG state；[1]DSG state；[2]ORING state*/
    uint8_t batSOHState: 2;                 /*! enum-type: GDUSmartBatterySohState*/
    uint8_t maxCycleLimit: 6;               /*! APP:cycle_limit*10*/
    uint8_t reserved3: 2;
    uint16_t lessBattery: 1;
    uint16_t batteryCommunicationAbnormal: 1;
    uint16_t reserved4: 3;
    uint16_t hasCellBreak: 1;
    uint16_t reserved5: 4;
    uint16_t isBatteryEmbed: 1;             /*! 0:embed;1:unmebed*/
    uint16_t heatState: 2;                  /*!enum-type: GDUSmartBatteryHeatState*/
    uint16_t socState: 3;                   /*!enum-type: GDUSmartBatterySocWarning*/
} T_GduFcSubscriptionSingleBatteryState;
```

typedef

struct T_GduFcSubscriptionWholeBatteryInfo

```
typedef struct BatteryWholeInfo {
    uint32_t capacity; /*!< 电池 capacity，单位： mAh。 */
    int32_t voltage; /*!< 电池 voltage，单位： mV。 */
    int32_t current; /*!< 电池 current，单位： mA。 */
    uint8_t percentage; /*!< 电池 capacity percentage，单位： 1%。 */
} T_GduFcSubscriptionWholeBatteryInfo;
```

typedef

struct T_GduFcSubscriptionSingleBatteryInfo

```
typedef struct BatterySingleInfo {
    uint8_t reserve;
```

```
    uint8_t batteryIndex;
    int32_t currentVoltage;                 /*! uint:mV*/
    int32_t currentElectric;                /*!uint:mA*/
    uint32_t fullCapacity;                  /*!uint:mAh*/
    uint32_t remainedCapacity;              /*!uint:mAh*/
    int16_t batteryTemperature;             /*!uint:□*/
    uint8_t cellCount;
    uint8_t batteryCapacityPercent;    /*!uint:%*/
    T_GduFcSubscriptionSingleBatteryState batteryState;
    uint8_t reserve1;
    uint8_t reserve2;
    uint8_t SOP;                                /*!相对功率百分比*/
} T_GduFcSubscriptionSingleBatteryInfo;
```

typedef struct T_GduFcSubscriptionControlDevice

```
typedef struct SDKCtrlInfo {
    uint8_t controlMode;          /*!< See CtlrMode in GDU_status.hpp*/
    uint8_t deviceStatus: 3; /*!< For S400 and M210V2(firmware version V01.00.0690 and later):0->rc
1->app    4->serial;
                                                Other: 0->rc    1->app    2->serial*/
    uint8_t flightStatus: 1; /*!< 1->opensd    0->close */
    uint8_t vrcStatus: 1;
    uint8_t reserved: 3;
} T_GduFcSubscriptionControlDevice;
```

typedef struct SyncTimestamp

```
typedef struct SyncTimestamp {
    uint32_t time2p5ms; /*!< 时钟时间为 2.5ms 的倍数。 同步定时器以 400Hz 运行，该字段以整
数步长递增 */
    uint32_t time1ns;     /*!< 2.5ms 脉冲的纳秒时间偏移 */
    uint32_t resetTime2p5ms; /*!< 自硬件同步开始以来经过的时钟时间（以 2.5 毫秒的倍数计）*/
    uint16_t index;     /*!< 这是您在使用上面的 setSyncFreq API 时填写的标签字段； 使用它来识
别具有同步数据的数据包。 这在您使用 freqInHz = 0 调用 setSyncFreq API 时很有用，因此您可以
获得一个可以用标签唯一标识的单个脉冲 - 允许您创建自己的具有唯一可标识脉冲的脉冲序列。*/
    uint8_t flag;        /*!< 当数据包对应于硬件脉冲时为真，否则为假。用户可以请求以比硬件线路
更高的频率发送软件数据包。*/
} SyncTimestamp;
```

typedef struct T_GduFcSubscriptionHardSync

```
typedef struct HardSyncData {
    SyncTimestamp ts;   /*!< 传入数据的时间戳 */
    struct Quaternion q;   /*!< 四元数 */
    T_GduVector3f a;   /*!< 加速度计读数单元: g */
    T_GduVector3f w;   /*!< 陀螺仪读数单元: rad/sec */
} T_GduFcSubscriptionHardSync;
```

typedef

struct T_GduFcSubscriptionRCWithFlagData

时间戳

```
typedef struct RCWithFlagData {
    gdu_f32_t pitch;          /*!< 上 = -0.999, 中 = 0.000, 上    =0.999 */
    gdu_f32_t roll;           /*!< 左 = -0.999, 中 = 0.000, 右    =0.999 */
    gdu_f32_t yaw;            /*!< 左 = -0.999, 中 = 0.000, 右    =0.999 */
    gdu_f32_t throttle;       /*!< 下 = -0.999, 中 = 0.000, 上    =0.999 */
    struct {
        uint8_t logicConnected: 1;   /*!< 0 如果天空或地面侧断开 3 秒   */
        uint8_t skyConnected: 1;   /*!< 天空端连接，接收器与 FC 相连 */
        uint8_t groundConnected: 1;   /*!< 地面段连接，RC 开启并与 FC 相连 */
        uint8_t appConnected: 1;   /*!< 移动设备 App 与 RC 相连 */
        uint8_t reserved: 4;
    } flag;
} T_GduFcSubscriptionRCWithFlagData;
```

typedef struct EscStatusIndividual

```
typedef struct EscStatusIndividual {
    int16_t current;                /*!< ESC 电流，单位： mA */
    int16_t speed;                  /*!< ESC 速度，单位： rpm */
    uint16_t voltage;               /*!< 从电池到 ESC 的输入电压，单位： mV */
```

```
    int16_t temperature;                /*!< ESC 温度，单位： degree C */
    uint16_t stall: 1; /*!< 电机堵转 */
    uint16_t empty: 1; /*!< 电机无负载 */
    uint16_t unbalanced: 1; /*!< 电机速度不平衡 */
    uint16_t escDisconnected: 1; /*!< ESC 断连 */
    uint16_t temperatureHigh: 1; /*!< 温度较高 */
    uint16_t reserved: 11;
} EscStatusIndividual;
```

typedef struct T_GduFcSubscriptionEscData

```
typedef struct EscData {
    EscStatusIndividual esc[8];
} T_GduFcSubscriptionEscData;
```

typedef

struct T_GduFcSubscriptionRTKConnectStatus

```
typedef struct RTKConnectStatus {
    uint16_t rtkConnected: 1;
    uint16_t reserve: 15;
} T_GduFcSubscriptionRTKConnectStatus;
```

typedef

struct T_GduFcSubscriptionFlightAnomaly

```
typedef struct FlightAnomaly {
    uint32_t impactInAir: 1;    /*!< 0: No impact,                          1: Impact happens in Air */
    uint32_t randomFly: 1;    /*!< 0: Normal,                               1: Randomly fly in GPS mode
without stick input*/
    uint32_t heightCtrlFail: 1;    /*!< 0: Height control normal,           1: Height control failed */
    uint32_t rollPitchCtrlFail: 1;    /*!< 0: Tilt control normal,          1: Tilt control failed */
    uint32_t yawCtrlFail: 1;    /*!< 0: Yaw control normal,                 1: Yaw control failed */
```

```
    uint32_t aircraftIsFalling: 1;    /*!< 0: Aircraft is not falling,              1: Aircraft is falling */
    uint32_t strongWindLevel1: 1;    /*!< 0: Wind is under big wind level 1, 1: wind is stronger than    big wind level 1*/
    uint32_t strongWindLevel2: 1;    /*!< 0: Wind is under big wind level 2, 1: wind is stronger than    big wind level 2*/
    uint32_t compassInstallationError: 1;    /*!< 0: Compass install right,              1: Compass install error */
    uint32_t imuInstallationError: 1;    /*!< 0: IMU install right,              1: IMU install error */
    uint32_t escTemperatureHigh: 1;    /*!< 0: ESC temperature is normal,          1: ESC temperature is high */
    uint32_t atLeastOneEscDisconnected: 1;    /*!< 0: No ESC disconnected,              1: At least one ESC is disconnected */
    uint32_t gpsYawError: 1;    /*!< 0: No GPS yaw error,              1: GPS yaw error */
    uint32_t reserved: 19;
} T_GduFcSubscriptionFlightAnomaly;
```

typedef struct T_GduFcSubscriptionPositionVO

```
typedef struct PositionVO {
    gdu_f32_t x;                /*!< 北 (best effort)，单位： m */
    gdu_f32_t y;                /*!< 东 (best effort)，单位： m */
    gdu_f32_t z;                /*!< 下， 单位： m */
    uint8_t xHealth: 1;
    uint8_t yHealth: 1;
    uint8_t zHealth: 1;
    uint8_t reserved: 5;
} T_GduFcSubscriptionPositionVO;
```

typedef struct T_GduFcSubscriptionAvoidData

```
typedef struct RelativePosition {
    gdu_f32_t down;              /*!< 距障碍物的距离 (m) */
    gdu_f32_t front;             /*!< 距障碍物的距离 (m) */
    gdu_f32_t right;             /*!< 距障碍物的距离 (m) */
    gdu_f32_t back;              /*!< 距障碍物的距离 (m) */
    gdu_f32_t left;              /*!< 距障碍物的距离 (m) */
    gdu_f32_t up;                /*!< 距障碍物的距离 (m) */
    uint8_t downHealth: 1;    /*!< Down sensor flag: 0 - not working, 1 - working */
    uint8_t frontHealth: 1; /*!< Front sensor flag: 0 - not working, 1 - working */
    uint8_t rightHealth: 1; /*!< Right sensor flag: 0 - not working, 1 - working */
```

```
    uint8_t backHealth: 1;    /*!< Back sensor flag: 0 - not working, 1 - working */
    uint8_t leftHealth: 1;    /*!< Left sensor flag: 0 - not working, 1 - working */
    uint8_t upHealth: 1;      /*!< Up sensor health flag: 0 - not working, 1 - working */
    uint8_t reserved: 2;      /*!< Reserved sensor health flag*/
} T_GduFcSubscriptionAvoidData;
```

typedef

struct T_GduFcSubscriptionHomePointInfo

```
typedef struct HomeLocationData {
    gdu_f64_t latitude;   /*!< 单位：  rad */
    gdu_f64_t longitude; /*!< 单位：  rad */
} T_GduFcSubscriptionHomePointInfo;
```

typedef struct GimbalSingleData

```
typedef struct GimbalSingleData {
    gdu_f32_t pitch;
    gdu_f32_t roll;
    gdu_f32_t yaw;
    uint32_t status;
    uint8_t mode;
} GimbalSingleData;
```

typedef

struct T_GduFcSubscriptionThreeGimbalData

```
typedef struct GimbalThreeData {
    GimbalSingleData gbData[3];
} T_GduFcSubscriptionThreeGimbalData;
```

函数原型

function GduFcSubscription_Init

功能：消息订阅模块初始化	product：all

消息订阅功能初始化，并开始执行消息订阅功能。

说明

- 在使用消息订阅功能订阅无人机上的信息前，请先使用本接口初始化消息订阅功能。
- 请勿在 main()函数中调用本接口，请在用户线程中调用本接口，启动调度器后，该接口将正常运行。
- 本接口执行时间可能会超过 500ms。

```
T_GduReturnCode GduFcSubscription_Init(void);
```

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

function GduFcSubscription_DeInit

功能：释放消息订阅功能	product：all

释放消息订阅功能。在无需使用消息订阅功能时，可调用本接口释放消息订阅功能。

说明： 释放消息订阅功能后，无人机中被占用的资源也将释放。

```
T_GduReturnCode GduFcSubscription_DeInit(void);
```

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

function GduFcSubscription_SubscribeTopic

功能：以阻塞模式订阅。 在从飞机订阅任何数据之前，GduFcSubscription_Init()函数必须被调用。	product：all

```
T_GduReturnCode GduFcSubscription_SubscribeTopic(E_GduFcSubscriptionTopic topic,
                                                 E_GduDataSubscriptionTopicFreq frequency,
                                                 GDUReceiveDataOfTopicCallback callback);
```

参数

topic：订阅名。

frequency：订阅的订阅频率。 订阅频率不能超过的最大频率限制，必须是枚举 E_GduFcSubscriptionTopicFreq 的值。 并且，订阅频率必须大于 0。用户可以在开发者网站（developer.GDU.com）文档的数据订阅部分找到的最大频率。

callback：用于接收订阅数据的回调函数。 如果不需要回调函数，此项可以设置为 NULL。

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

function GduFcSubscription_GetLatestValueOfTopic

功能：获取最新的数据和时间戳	product：all

获取无人机上订阅项最新的数据和时间戳，订阅失败时，该接口将返回错误码。

说明： 调用本接口时，请指定订阅项，以及相应的数据。

```
T_GduReturnCode GduFcSubscription_GetLatestValueOfTopic(E_GduFcSubscriptionTopic topic,
                                                        uint8_t *data, uint16_t dataSizeOfTopic,
                                                        T_GDUDataTimestamp *timestamp);
```

参数

topicName：订阅项的名称。

data：请正确地指向用于存储订阅项数据的存储空间，否则，本接口将返回错误码。

dataSizeOfTopic：请正确地指向用于存储订阅项数据的存储空间，正常情况下，该数值与订阅项的长度相同，否则可能会导致内存溢出等问题。

timestamp：请正确地指向用于存储时间戳的内存空间，否则，本接口将返回错误码或出现内存溢出等问题，若无需获取时间戳，则该参数可为空。

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GDUErrorCode

时间同步

时间同步相关功能的头文件为 GDU_time_sync.h，本文档描述了 GDU_time_sync.h 文件中结构体和函数原型的关键信息和使用方法。

目录

宏定义、枚举与结构体

typedef

function GduGetNewestPpsTriggerLocalTimeUsCallback

功能：获取最新的 PPS 触发的时间戳	product：all

用于获取最新 PPS 触发时间戳的回调函数原型。

说明： 用户不能在回调函数中执行阻塞式操作或函数，因为这会阻塞 GDU 根线程，导致系统响应慢、负载断开或死循环等问题。

typedef T_PsdkReturnCode (*GetNewestPpsTriggerLocalTimeUsCallback)(uint64_t *localTimeUs);

参数

localTimeUs：指向用于存储 PPS 触发时间戳的内存空间的指针。

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

typedef struct T_GduTimeSyncAircraftTime

无人机中时间系统的数据结构

```
typedef struct {
    uint16_t year;              年
    uint8_t month;              月，1～12
    uint8_t day;                日，1～31
    uint8_t hour;               小时，0～23
    uint8_t minute;             分钟，0～59
    uint8_t second;             秒，0～59
    uint32_t microsecond;       微秒，0～999999.
} T_GduTimeSyncAircraftTime;
```

函数原型

function GduTimeSync_Init

功能：初始化时间同步功能模块	product：all

在阻塞模式下初始化时间同步模块。 用户应在所有其他时间同步操作之前调用此函数，就像传输时间一样。

说明：

- 该函数必须在用户任务中调用，而不是 main() 函数，并且在调度程序启动后调用。

```
T_GduReturnCode GduTimeSync_Init(void);
```

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

function GduTimeSync_RegGetNewestPpsTriggerTimeCallback

功能：获取最新的时间戳	product：all

注册回调函数，用于在检测到 PPS 上升沿信号时获取本地时间系统中的最新时间戳。

```
T_GduReturnCode
GduTimeSync_RegGetNewestPpsTriggerTimeCallback(GDUGetNewestPpsTriggerLocalTimeUsCallback
callback);
```

参数

callback：指向回调函数的指针。

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

function GduTimeSync_TransferToAircraftTime

功能：时间转换	product：all

将负载设备的本地时间转换为无人机上的时间。

说明： 在使用本接口时，请调用 PsdkTimeSync_RegGetNewestPpsTriggerTimeCallback() 注册用于获取负载设备检测到 PPS 信号上升沿时，负载设备本地最新的时间戳的回调函数。

T_GduReturnCode GduTimeSync_TransferToAircraftTime(uint64_t localTimeUs, T_GduTimeSyncAircraftTime *aircraftTime);

参数

localTimeUs：负载设备中的本地时间，单位：微秒

aircraftTime：指向用于存储无人机系统时间的内存空间

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

移植

平台移植

跨平台相关功能的头文件为 G_platform.h，本文档描述了 GDU_platform.h 文件中结构体和函数原型的关键信息和使用方法。

目录

T_GduHalUartHandler

T_GduHalUsbBulkChannelInfo

T_GduHalUsbBulkInfo

T_GduHalUsbBulkDeviceInfo

T_GduHalUsbBulkHandler

T_GduHalNetworkHandler

T_GduOsalHandler

T_GduFileSystemHandler

T_GduSocketHandler

GduPlatform_RegHalUartHandler

- 函数原型

GduPlatform_RegHalUsbBulkHandler

GduPlatform_RegHalNetworkHandler

GduPlatform_RegOsalHandler

GduPlatform_RegFileSystemHandler

GduPlatform_RegSocketHandler

GduPlatform_GetOsalHandler

GduPlatform_GetFileSystemHandler

GduPlatform_GetSocketHandler

宏定义

- uart 操作的平台句柄。

typedef void *T_GduUartHandle;

- 使 usb 批量操作的平台句柄。

typedef void *T_GduUsbBulkHandle;

- 网络操作的平台句柄。

typedef void *T_GDUNetworkHandle;

- 线程任务操作的平台句柄。

typedef void *T_GduTaskHandle;

- 互斥操作的平台句柄。

typedef void *T_GDUMutexHandle;

- 信号量操作的平台句柄。

typedef void *T_GduSemaHandle;

- 文件操作的平台句柄。

typedef void *T_GduFileHandle;

- dir 操作的平台句柄。

typedef void *T_GDUDirHandle;

- socket 操作的平台句柄。

typedef void *T_GduSocketHandle;

#枚举

typedef enum E_GduHalUartNum

```
typedef enum {
    GDU_HAL_UART_NUM_0,
    GDU_HAL_UART_NUM_1,
} E_GduHalUartNum;
```

typedef enum E_GduSocketMode

```
typedef enum {
    GDU_SOCKET_MODE_UDP,
    GDU_SOCKET_MODE_TCP,
} E_GduSocketMode;
```

结构体

typedef struct T_GduUrtStatus

```
typedef struct {
    bool isConnect;
} T_GduUartStatus;
```

typedef struct T_GduTime

```
typedef struct {
    uint16_t year;
    uint8_t month;
    uint8_t day;
    uint8_t hour;
    uint8_t minute;
    uint8_t second;
} T_GduTime;
```

typedef struct T_GduFileInfo

```
typedef struct {
    uint32_t size;
    T_GduTime createTime;
    T_GduTime modifyTime;
    char path[GDU_FILE_PATH_SIZE_MAX];
    bool isDir;
} T_GduFileInfo;
```

typedef struct T_GduHalUartHandler

```
typedef struct {
    T_GduReturnCode (*UartInit)(E_GduHalUartNum uartNum, uint32_t baudRate, T_GduUartHandle *uartHandle);

    T_GduReturnCode (*UartDeInit)(T_GduUartHandle uartHandle);

    T_GduReturnCode (*UartWriteData)(T_GduUartHandle uartHandle, const uint8_t *buf, uint32_t len, uint32_t *realLen);

    T_GduReturnCode (*UartReadData)(T_GduUartHandle uartHandle, uint8_t *buf, uint32_t len, uint32_t *realLen);

    T_GduReturnCode (*UartGetStatus)(E_GduHalUartNum uartNum, T_GduUartStatus *status);
} T_GduHalUartHandler;
```

typedef struct T_GduHalUsbBulkChannelInfo

```
typedef struct {
    uint16_t interfaceNum;
    uint16_t endPointIn;
    uint16_t endPointOut;
} T_GduHalUsbBulkChannelInfo;
```

typedef struct T_GduHalUsbBulkInfo

```
typedef struct {
    bool isUsbHost;
    // attention: if 'isUsbHost' equals false, the following parameters is not valid.
    uint16_t pid;
    uint16_t vid;
    T_GduHalUsbBulkChannelInfo channelInfo;
} T_GduHalUsbBulkInfo;
```

typedef struct T_GduHalUsbBulkDeviceInfo

```
typedef struct {
    uint16_t pid;
    uint16_t vid;
    uint8_t bulkChannelNum;
    T_GduHalUsbBulkChannelInfo *channelInfo;
} T_GduHalUsbBulkDeviceInfo;
```

typedef struct T_GduHalUsbBulkHandler

```
typedef struct {
    T_GduReturnCode (*UsbBulkInit)(T_GduHalUsbBulkInfo usbBulkInfo, T_GduUsbBulkHandle
*usbBulkHandle);

    T_GduReturnCode (*UsbBulkDeInit)(T_GduUsbBulkHandle usbBulkHandle);

    T_GduReturnCode (*UsbBulkWriteData)(T_GduUsbBulkHandle usbBulkHandle, const uint8_t *buf,
                                        uint32_t len, uint32_t *realLen);

    T_GduReturnCode (*UsbBulkReadData)(T_GduUsbBulkHandle usbBulkHandle, uint8_t *buf,
                                       uint32_t len, uint32_t *realLen);
```

```
    T_GduReturnCode (*UsbBulkGetDeviceInfo)(T_GduHalUsbBulkDeviceInfo *deviceInfo);
} T_GduHalUsbBulkHandler;
```

typedef struct T_GduHalNetworkHandler

```
typedef struct {
    T_GduReturnCode (*NetworkInit)(const char *ipAddr, const char *netMask, T_GDUNetworkHandle
*networkHandle);

    T_GduReturnCode (*NetworkDeInit)(T_GDUNetworkHandle networkHandle);
} T_GduHalNetworkHandler;
```

typedef struct T_GDUOsalHandler

```
typedef struct {
    T_GduReturnCode (*TaskCreate)(const char *name, void *(*taskFunc)(void *),
                                  uint32_t stackSize, void *arg, T_GduTaskHandle *task);

    T_GduReturnCode (*TaskDestroy)(T_GduTaskHandle task);

    T_GduReturnCode (*TaskSleepMs)(uint32_t timeMs);

    T_GduReturnCode (*MutexCreate)(T_GDUMutexHandle *mutex);

    T_GduReturnCode (*MutexDestroy)(T_GDUMutexHandle mutex);

    T_GduReturnCode (*MutexLock)(T_GDUMutexHandle mutex);

    T_GduReturnCode (*MutexUnlock)(T_GDUMutexHandle mutex);

    T_GduReturnCode (*SemaphoreCreate)(uint32_t initValue, T_GduSemaHandle *semaphore);

    T_GduReturnCode (*SemaphoreDestroy)(T_GduSemaHandle semaphore);

    T_GduReturnCode (*SemaphoreWait)(T_GduSemaHandle semaphore);

    T_GduReturnCode (*SemaphoreTimedWait)(T_GduSemaHandle semaphore, uint32_t waitTimeMs);

    T_GduReturnCode (*SemaphorePost)(T_GduSemaHandle semaphore);
```

```
    T_GduReturnCode (*GetTimeMs)(uint32_t *ms);

    T_GduReturnCode (*GetTimeUs)(uint64_t *us);

    void *(*Malloc)(uint32_t size);

    void (*Free)(void *ptr);
} T_GDUOsalHandler;
```

typedef struct T_GduFileSystemHandler

```
typedef struct {
    T_GduReturnCode (*FileOpen)(const char *fileName, const char *fileMode, T_GduFileHandle *fileObj);

    T_GduReturnCode (*FileClose)(T_GduFileHandle fileObj);

    T_GduReturnCode (*FileWrite)(T_GduFileHandle fileObj, const uint8_t *buf, uint32_t len, uint32_t
*realLen);

    T_GduReturnCode (*FileRead)(T_GduFileHandle fileObj, uint8_t *buf, uint32_t len, uint32_t *realLen);

    T_GduReturnCode (*FileSeek)(T_GduFileHandle fileObj, uint32_t offset);

    T_GduReturnCode (*FileSync)(T_GduFileHandle fileObj);

    T_GduReturnCode (*DirOpen)(const char *filePath, T_GDUDirHandle *dirObj);

    T_GduReturnCode (*DirClose)(T_GDUDirHandle dirObj);

    T_GduReturnCode (*DirRead)(T_GDUDirHandle dirObj, T_GduFileInfo *fileInfo);

    T_GduReturnCode (*Mkdir)(const char *filePath);

    T_GduReturnCode (*Unlink)(const char *filePath);

    T_GduReturnCode (*Rename)(const char *oldFilePath, const char *newFilePath);

    T_GduReturnCode (*Stat)(const char *filePath, T_GduFileInfo *fileInfo);
} T_GduFileSystemHandler;
```

typedef struct T_GduSocketHandler

```
typedef struct {
    T_GduReturnCode (*Socket)(E_GduSocketMode mode, T_GduSocketHandle *socketHandle);

    T_GduReturnCode (*Close)(T_GduSocketHandle socketHandle);

    T_GduReturnCode (*Bind)(T_GduSocketHandle socketHandle, const char *ipAddr, uint32_t port);

    T_GduReturnCode (*UdpSendData)(T_GduSocketHandle socketHandle, const char *ipAddr, uint32_t
port,
                                   const uint8_t *buf, uint32_t len, uint32_t *realLen);

    T_GduReturnCode (*UdpRecvData)(T_GduSocketHandle socketHandle, char *ipAddr, uint32_t *port,
                                   uint8_t *buf, uint32_t len, uint32_t *realLen);

    T_GduReturnCode (*TcpListen)(T_GduSocketHandle socketHandle);

    T_GduReturnCode (*TcpAccept)(T_GduSocketHandle socketHandle, char *ipAddr, uint32_t *port,
                                 T_GduSocketHandle *outSocketHandle);

    T_GduReturnCode (*TcpConnect)(T_GduSocketHandle socketHandle, const char *ipAddr, uint32_t
port);

    T_GduReturnCode (*TcpSendData)(T_GduSocketHandle socketHandle,
                                   const uint8_t *buf, uint32_t len, uint32_t *realLen);

    T_GduReturnCode (*TcpRecvData)(T_GduSocketHandle socketHandle,
                                   uint8_t *buf, uint32_t len, uint32_t *realLen);
} T_GduSocketHandler;
```

函数原型

function GduPlatform_RegHalUartHandler

功能：注册 Hal 层接口函数	product：all

注册适配负载设备 Hal 层接口的回调函数。

说明

- 在注册 Hal 层接口的回调函数前，请测试该函数能够正常写入 Hal 层接口配置信息；
- 请在使用负载设备的功能前先调用该接口，否则负载设备将无法正常运行。

T_PsdkReturnCode GduPlatform_RegHalUartHandler(const T_GduHalUartHandler *halUartHandler);

参数

halUartHandler：指向负载设备操作系统 hal 层接口函数

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

function GduPlatform_RegHalUsbBulkHandler

功能：注册 USB 批量接口函数	product：all

通过您的平台注册 USB 批量接口的处理程序。

T_GduReturnCode GduPlatform_RegHalUsbBulkHandler(const T_GduHalUsbBulkHandler *halUsbBulkHandler);

参数

fileSystemHandler：指向您的平台的 USB 批量接口的处理程序的指针。

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

function GduPlatform_RegHalNetworkHandler

功能：注册 hal 网络接口接口函数	product：all

通过您的平台注册 hal 网络接口的处理程序。

说明：

- 在注册 Osal 层接口的回调函数前，请测试该函数能够正常写入 Osal 层接口配置信息；
- 请在使用负载设备的功能前先调用该接口，否则负载设备将无法正常运行。

T_GduReturnCode GduPlatform_RegHalNetworkHandler(const T_GduHalNetworkHandler *halNetworkHandler);

参数

osalHandler：指向负载设备第三方开发平台 Osal 层接口函数

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

function GduPlatform_RegOsalHandler

功能：注册 osal 接口函数	product：all

T_GduReturnCode GduPlatform_RegOsalHandler(const T_GDUOsalHandler *osalHandler);

参数

osalHandler：指向平台的 osal 接口处理程序的指针。

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

function GduPlatform_RegFileSystemHandler

功能：注册文件系统 接口函数	product：all

T_GduReturnCode GduPlatform_RegFileSystemHandler(const T_GduFileSystemHandler *fileSystemHandler);

参数

fileSystemHandler：指向平台的文件系统接口处理程序的指针。

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

function GduPlatform_RegSocketHandler

功能：注册 socket 接口函数	product：all

T_GduReturnCode GduPlatform_RegSocketHandler(const T_GduSocketHandler *socketHandler);

参数

socketHandler：指向平台的 socket 接口处理程序的指针。

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

function GduPlatform_GetOsalHandler

功能：获取 osal 接口的处理程序	product：all

T_GDUOsalHandler *GduPlatform_GetOsalHandler(void);

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

function GduPlatform_GetFileSystemHandler

功能：获取文件系统接口的处理程序	product：all

T_GduFileSystemHandler *GduPlatform_GetFileSystemHandler(void);

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

function GduPlatform_GetSocketHandler

功能：获取 socket 接口的处理程序	product：all

T_GduSocketHandler *GduPlatform_GetSocketHandler(void);

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

应用实践

获取相机码流

获取相机码流相关功能的头文件为 GDU_liveview.h ，本文档描述了

GDU_liveview.h　文件中结构体和函数原型的关键信息和使用方法。

目录

宏定义、枚举与结构体

typedef enum E_GduLiveViewCameraPosition

```
typedef enum {
    GDU_LIVEVIEW_CAMERA_POSITION_NO_1 = GDU_MOUNT_POSITION_PAYLOAD_PORT_NO1,
    GDU_LIVEVIEW_CAMERA_POSITION_NO_2 = GDU_MOUNT_POSITION_PAYLOAD_PORT_NO2,
    GDU_LIVEVIEW_CAMERA_POSITION_NO_3 = GDU_MOUNT_POSITION_PAYLOAD_PORT_NO3,
    GDU_LIVEVIEW_CAMERA_POSITION_FPV = 7
} E_GduLiveViewCameraPosition;
```

typedef enum E_GduLiveViewCameraSource

角度限制的数据结构

```
typedef enum {
    GDU_LIVEVIEW_CAMERA_SOURCE_DEFAULT = 0,
    GDU_LIVEVIEW_CAMERA_SOURCE_H20_WIDE = 1,
    GDU_LIVEVIEW_CAMERA_SOURCE_H20T_WIDE = 1,
    GDU_LIVEVIEW_CAMERA_SOURCE_H20_ZOOM = 2,
    GDU_LIVEVIEW_CAMERA_SOURCE_H20T_ZOOM = 2,
    GDU_LIVEVIEW_CAMERA_SOURCE_H20T_IR = 3
} E_GduLiveViewCameraSource;
```

typedef function GduLiveview_H264Callback

```
typedef void (*GduLiveview_H264Callback)(E_GduLiveViewCameraPosition position, const uint8_t *buf, uint32_t len);
```

函数原型

function GduLiveview_Init

功能：liveview 模块初始化	product：all

接口初始化需要在 GDUCore_Init 之后。

T_GduReturnCode GduLiveview_Init(void);

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

function GduLiveview_Deinit

功能：liveview 模块反初始化	product：all

T_GduReturnCode GduLiveview_Deinit(void);

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

function GduLiveview_StartH264Stream

功能：按选定位置启动 FPV 或摄像机 H264 流。	product：all

T_GduReturnCode GduLiveview_StartH264Stream(E_GduLiveViewCameraPosition position,
E_GduLiveViewCameraSource source,
GduLiveview_H264Callback callback);

参数

position：指出哪个相机输出 H264 流

source：指出哪个子摄像头输出 H264 码流

callback：接收到新的 h264 帧时在回调线程中调用的回调函数

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

function GduLiveview_StopH264Stream

功能：按选定位置关闭 FPV 或摄像机 H264 流。	product：all

T_GduReturnCode GduLiveview_StopH264Stream(E_GduLiveViewCameraPosition position);

参数

position：指出哪个相机输出 H264 流

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

精准定位

精准定位相关功能的头文件为 GDU_positioning.h ，本文档描述了 GDU_positioning.h 文件中结构体和函数原型的关键信息和使用方法。

目录

结构体

typedef struct T_GduPositioningEventInfo

描述一个定位事件

```
typedef struct {
    uint16_t eventSetIndex;          负载设备触发定位事件的索引，该索将被写入.mark 文件，
                                     若无需为定位事件设置索引，可填入 0
    uint8_t targetPointIndex;        负载设备获取定位点的索引，该索将被写入.mark 文件，若
                                     无需为定位事件设置索引，可填入 0
    T_GduTimeSyncAircraftTime eventTime;      定位事件发生时的时间戳，用户需要通过时间同步
                                              模块中的 PsdkTimeSync_TransferToAircraftTime()
接口，该接口将该时间（负载设备上的本地时间）转换为无人机上的时间
} T_GduPositioningEventInfo;
```

typedef struct T_GduPositioningPosition

用于描述一个定位点的结构体

```
typedef struct {
    gdu_f64_t longitude;        定位点的经度，单位：度
    gdu_f64_t latitude;         定位点的维度，单位：度
    gdu_f64_t height;           定位点海平面以上的高度，单位：米
} T_GduPositioningPosition;
```

typedef

struct T_GduPositioningPositionStandardDeviatio

n

定位点标准差值

```
typedef struct {
    gdu_f32_t longitude;        定位点经度的标准差值，单位：度
    gdu_f32_t latitude;         定位点纬度的标准差值，单位：度
    gdu_f32_t height;           定位点海平面以上高度的标准差值，单位：m.
} T_GduPositioningPositionStandardDeviation;
```

typedef struct T_GduPositioningPositionInfo

获取目标点的细节信息

```
typedef struct {
    E_GduDataSubscriptionPositionSolutionProperty positionSolutionProperty;    位置解属性
    T_GDUAttitude3d uavAttitude;      获取目标点时无人机的姿态角，单位：度
    T_GduVector3d offsetBetweenMainAntennaAndTargetPoint;  在 NED 大地坐标系下，指定无人机主天线到目标点的偏移，单位：mm
    T_GduPositioningPosition targetPointPosition; 目标点在大地坐标系中的位置
    T_GduPositioningPositionStandardDeviation  targetPointPositionStandardDeviation;    指定目标点的标准差
} T_GduPositioningPositionInfo;
```

函数原型

function GduPositioning_Init

功能：初始化精准定位模块	product：all

在使用精准定位功能前，请在阻塞的模式下调用本接口初始化精准定位模块。

说明

- 请勿在主函数中调用该接口。

```
T_GduReturnCode GduPositioning_Init(void);
```

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

function GduPositioning_SetTaskIndex

功能：设置任务索引	

一次任务可能包含对一个区域做航测建图，默认任务索引为 0。

```
void GduPositioning_SetTaskIndex(uint8_t index);
```

参数

index：任务索引

function GduPositioning_GetPositionInformation Sync

功能：获取精准定位信息	product：all

调用该接口获取目标点精准的定位信息以及其他信息，详情请参见：T_GduPositioningPositionInfo。

说明

- 开发者可同时获取多个定位事件触发的定位信息，如获取无人机上多个相机类负载设备同步变焦时的定位信息；
- 当定位事件发生时候，该接口将以阻塞的模式获取精准的定位信息；
- 在获取精准定位前，开发者必须注册用于获取无人机最新的 PPS 时间戳的回调函数，详情请参见：GduPositioning_RegGetNewestPpsTriggerTimeCallback()；

- 在获取精准定位时，精准定位的目标点默认为主云台接口的中心；

- 开发者利用主云台接口的中心、主云台口与 RTK 主天线的距离、无人机姿态角度、云台结构等数据计算兴趣点精准的定位信息。

- 所有请求的时间戳必须介于比最新同步时间戳早 2 秒与比最新同步时间戳早 1 秒的时间点之间。

```
T_GduReturnCode GduPositioning_GetPositionInformationSync(uint8_t eventCount,
T_GduPositioningEventInfo *eventInfo,
                                                          T_GduPositioningPositionInfo
*positionInfo);
```

参数

eventCount：同时获取定位信息的事件数量（ < 5 ）

eventInfo：指向定位时间的信息

positionInfo：目标点的定位信息

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

标准负载

云台功能

PSDK 云台相关功能的头文件为 gdu_gimbal.h，本文档描述了 gdu_gimbal.h 文件中结构体和函数原型的关键信息和使用方法。

目录

宏定义、枚举与结构体

typedef enum E_GduGimbalAxis

云台轴

```
typedef enum {
    GDU_GIMBAL_AXIS_PITCH = 0,    俯仰轴
    GDU_GIMBAL_AXIS_ROLL = 1,    横滚轴
    GDU_GIMBAL_AXIS_YAW = 2,    偏航轴
} E_GduGimbalAxis;
```

typedef struct T_GDUGimbalSystemState

云台的状态

```
typedef struct {
    bool resettingFlag;云台重置中
    bool mountedUpward;上置云台
    bool blockingFlag;  云台卡顿
    bool pitchRangeExtensionEnabledFlag;    开启云台俯仰轴角度范围扩展功能
    E_GduGimbalMode gimbalMode;  云台控制方式
    T_GDUAttitude3d fineTuneAngle;    云台角度微调，单位：0.1 度
    T_GDUGimbalControllerSmoothFactor smoothFactor;  云台平滑度
    T_GDUGimbalControllerMaxSpeedPercentage maxSpeedPercentage;    云台最大速度百分比
} T_GDUGimbalSystemState;
```

typedef struct T_GDUGimbalAttitudeInformation

云台角度

```
typedef struct {
    T_GDUAttitude3d attitude;     云台角度 单位：0.1 度
    T_GDUGimbalReachLimitFlag reachLimitFlag;    云台关节角到达限位标志
} T_GDUGimbalAttitudeInformation;
```

typedef struct T_GduGimbalCommonHandler

云台控制，请根据本结构体中的函数构造云台控制相关功能的回调函数，其中包括：

- 获取云台当前的状态
- 获取云台当前的角度
- 获取云台转动速度
- 转动云台
- 恢复云台信息

说明： 为防止该函数阻塞 PSDK 的主线程，导致出现程序响应缓慢、相机类负载设备断连及死循环等问题，请勿以阻塞的方式在回调函数中执行该函数。

```
typedef struct {

@brief 获取云台当前状态的回调函数
@param systemState: 指向用于存储云台状态的内存空间
```

@return 执行结果
T_GduReturnCode (*GetSystemState)(T_GduGimbalSystemState *systemState);

@brief 获取云台当前角度的回调函数
@param attitudeInformation: 指向用于存储云台角度信息的内存空间
@return 执行结果
T_GduReturnCode (*GetAttitudeInformation)(T_GduGimbalAttitudeInformation *attitudeInformation);

@brief 获取云台校准状态的回调函数
@param calibrationState: 指向用于存储云台校准状态的内存空间
@return 执行结果
T_GduReturnCode (*GetCalibrationState)(T_GduGimbalCalibrationState *calibrationState);

@brief 转动云台的回调函数，当有模块控制云台时将触发该函数
@note 在某些场景中，由于该回调函数的调用频率较高（如 200Hz），因此请为该回调函数设置较短的执行时间
@param rotationMode: 云台控制方式
@param rotationProperty: 云台转动的属性
@param rotationValue: 云台转动的角度，单位：0.1 度（若云台的控制方式为 GDU_GIMBAL_ROTATION_MODE_RELATIVE_ANGLE
* 或 GDU_GIMBAL_ROTATION_MODE_ABSOLUTE_ANGLE，0.1 度/s（若云台的控制方式为 GDU_GIMBAL_ROTATION_MODE_SPEED）
@return 执行结果
T_GduReturnCode (*Rotate)(E_GduGimbalRotationMode rotationMode,
T_GduGimbalRotationProperty rotationProperty,
T_GduAttitude3d rotationValue);

@brief 控制云台开始校准
@note 云台校准的时间最大为 120s，超出该时间后，GDU Pilot 或基于 MSDK 开发的移动端 App 将认为云台校准失败
@return 执行结果
T_GduReturnCode (*StartCalibrate)(void);

@brief 设置云台的平滑度
@param smoothingFactor: 平滑度，该数值越大，云台的加速度越大，云台转动的最大
加速度= 10000 × (0.8 ^ (1 + X)) deg/s^2 (X 为平滑控制系数) 取值范围为 0～30
@param axis: 指定云台上所需设置平滑度的轴
@return 执行结果
T_GduReturnCode (*SetControllerSmoothFactor)(uint8_t smoothingFactor, E_GduGimbalAxis axis);

@brief 开启或关闭俯仰轴欧拉角扩展角的回调函数
@details 使用俯仰轴角度范围扩展功能后，可将云台俯仰轴的欧拉角角度限制设置为默认限制或扩展限制
@param enabledFlag: 开启或关闭俯仰轴欧拉角扩展角功能
@return 执行结果

T_GduReturnCode (*SetPitchRangeExtensionEnabled)(bool enabledFlag);

@brief 设置云台最大速度百分比
@param maxSpeedPercentage: 设置云台的最大速度（范围：1～100），云台实际最大的转动速度= 默认最大速度 × 最大速度百分比
@param axis: 指定云台上所需设置最大速度百分比的轴
@return 执行结果
T_GduReturnCode (*SetControllerMaxSpeedPercentage)(uint8_t maxSpeedPercentage, E_GduGimbalAxis axis);

@brief 恢复云台信息
@return 执行结果
T_GduReturnCode (*Restore FactorySettings)(void);

@brief 设置云台模式的回调函数
@param mode: 云台模式
@return 执行结果
T_GduReturnCode (*SetMode)(E_GduGimbalMode mode);

@brief 重置云台参数的回调函数
@note 重置云台参数时，将会中断云台的工作
@param mode: 云台复位模式
@return 执行结果
T_GduReturnCode (*Reset)(E_GduGimbalResetMode mode);

@brief 云台角度微调
@param fineTuneAngle: 微调值，单位：0.1 度。大于 0 代表朝向机体坐标的正方向旋转
@return 执行结果
T_GduReturnCode (*FineTuneAngle)(T_GduAttitude3d fineTuneAngle);

@brief 获取云台转动速度的回调函数
@note 调用该函数后，用户即可 20 ms 内获取云台转动的速度（50 Hz）
@param rotationSpeed: 指向用于存储云台转动速度的内存空间，使用机体坐标系，单位：0.1 度/s。
@return 执行结果
T_GduReturnCode (*GetRotationSpeed)(T_GduAttitude3d *rotationSpeed);

@brief 获取云台关节角的回调函数
@param jointAngle: 指向用于存储云台关节角的内存空间，单位：0.1 度
@return 执行结果
T_GduReturnCode (*GetJointAngle)(T_GduAttitude3d *jointAngle);
} T_GduGimbalCommonHandler;

函数原型

function GduGimbal_Init

功能：初始化云台模块	product：all

在使用云台控制功能前，请先调用本接口初始化云台模块。

```
T_GduReturnCode GduGimbal_Init(void);
```

返回值

返回云台初始化的结果，详情请参见：GduErrorCode

function GduGimbal_DeInit

功能：反初始化云台模块	product：all

```
T_GduReturnCode GduGimbal_DeInit(void);
```

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

function GduGimbal_RegCommonHandler

功能：注册云台控制功能	product：all

注册控制云台的回调函数。

T_GduReturnCode GduGimbal_RegCommonHandler(const T_GduGimbalCommonHandler *commonHandler);

参数

commonHandler：指向回调函数

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

相机功能

PSDK 相机相关功能的头文件为 GDU_payload_camera.h，本文档描述了 gdu_payload_camera.h 文件中结构体和函数原型的关键信息和使用方法。

- 宏定义、枚举与结构体

  E_GduCameraMode

  E_GduCameraShootPhotoMode

  E_GduCameraShootingState

  T_GduCameraSDCardState

  T_GduCameraSystemState

  T_GduCameraCommonHandler

- 函数原型

  GduPayloadCamera_Init

  GduPayloadCamera_RegCommonHandler

宏定义、枚举与结构体

typedef enum E_GduCameraMode

相机类负载设备的模式

```
typedef enum {
    GDU_CAMERA_MODE_SHOOT_PHOTO = 0,        拍照模式
    GDU_CAMERA_MODE_RECORD_VIDEO = 1,       录像模式
    GDU_CAMERA_MODE_PLAYBACK = 2,       媒体文件回放
} E_GduCameraMode;
```

typedef enum E_GduCameraShootPhotoMode

相机类负载设备的拍照模式

```
typedef enum {
    GDU_CAMERA_SHOOT_PHOTO_MODE_SINGLE = 1,       单张拍摄模式
    GDU_CAMERA_SHOOT_PHOTO_MODE_BURST = 4,       连拍模式
    GDU_CAMERA_SHOOT_PHOTO_MODE_INTERVAL = 6,       间隔拍摄模式
} E_GduCameraShootPhotoMode;
```

typedef enum E_GduCameraShootingState

相机类负载设备拍照时的状态

```
typedef enum {
GDU_CAMERA_SHOOTING_PHOTO_IDLE = 0,相机为空闲状态
GDU_CAMERA_SHOOTING_SINGLE_PHOTO = 1,   相机正处于单拍状态
GDU_CAMERA_SHOOTING_BURST_PHOTO = 2,    相机正处于连拍状态
GDU_CAMERA_SHOOTING_INTERVAL_PHOTO = 6,相机正处于定时拍照状态
} E_GduCameraShootingState;
```

typedef struct T_GduCameraSDCardState

相机类负载设备中 SD 卡的状态

typedef struct {
bool isInserted; 相机类负载设备中是否已插入 SD 卡
bool isVerified; 相机类负载设备中的 SD 卡是否被认证
bool isInitializing; 相机类负载设备中 SD 卡是否已初始化
bool isReadOnly; 相机类负载设备中的 SD 卡是否处于只读模式
bool isFormatting; 相机类负载设备中的 SD 卡正在格式化
bool isFull; 相机类负载设备中的 SD 卡内存已漫
bool isInvalidFormat; 相机类负载设备中的 SD 卡已格式化
bool hasError; 相机类负载设备中的 SD 卡出现错误
uint32_t totalSpaceInMB; 相机类负载设备中 SD 卡的总容量，单位：MB
uint32_t remainSpaceInMB; 相机类负载设备中 SD 卡的剩余容量，单位：MB
uint32_t availableCaptureCount;当前相机类负载设备中 SD 卡可存储的相片数量
uint32_t availableRecordingTimeInSeconds; 当前相机类负载设备中 SD 卡可存储视频的长度，单位：s
} T_GduCameraSDCardState;

typedef struct T_GduCameraSystemState

相机类负载设备的状态

typedef struct {
E_GduCameraMode cameraMode; 相机类负载设备当前的模式，详情请参见：E_GduCameraMode
E_GduCameraShootingState shootingState; 相机类负载设备当前拍照的模式，详情请参见：E_GduCameraShootingState
bool isStoring; 相机类负载设备是否正在存储媒体文件
bool isShootingIntervalStart; 相机类负载设备是否正在定时拍照
uint16_t currentPhotoShootingIntervalTimeInSeconds; 指定相机类负载设备定时拍照倒计时，当该该数值递减为 0 时将触发相机拍照，单位：s
uint16_t currentPhotoShootingIntervalCount; 指定相机类负载设备定时拍照时拍摄的照片
bool isRecording; 相机类负载设备是否正在录像
uint16_t currentVideoRecordingTimeInSeconds; 指定相机类负载设备录像时间的进度，单位：s

bool isOverheating;　　相机是否过热
bool hasError;相机是否出现错误
} T_GduCameraSystemState;

typedef struct T_GduCameraCommonHandler

控制相机类负载设备执行基础动作，请根据本结构体中的函数构造相机类负载设备执行基础功能的回调函数，其中包括：

- 获取相机类负载设备的当前状态
- 设置相机类负载设备的工作模式
- 获取相机类负载设备当前的工作模式
- 控制相机类负载设备开始录像
- 控制相机类负载设备停止录像
- 控制相机类负载设备开始拍摄照片
- 控制相机类负载设备停止拍摄照片
- 设置相机类负载设备的拍照模式
- 获取相机类负载设备当前的拍照模式
- 设置相机类负载设备在连拍模式下的连拍张数
- 获取相机类负载设备当前的连拍张数
- 设置相机类负载设备在定时拍照模式下的拍照间隔
- 获取相机类负载设备在定时拍照模式下的拍照间隔

- 获取相机类负载设备中当前 SD 卡状态
- 控制相机类负载设备格式化 SD 卡

说明

- 为防止该函数阻塞 PSDK 的主线程，导致出现程序响应缓慢、相机类负载设备断连及死循环等问题，请勿以阻塞的方式在回调函数中执行该函数。
- 使用控制相机类负载设备执行基础动作的功能时，请根据如下函数原型开发相机类负载设备的基础功能。

```
typedef struct {
  @brief  获取相机类负载设备当前状态的回调函数
  @param systemState:  指向用于存储相机类负载设备当前状态的内存空间
  @return  执行结果
  T_GduReturnCode (*GetSystemState)(T_GduCameraSystemState *systemState);

  @brief  设置相机类负载设备工作模式的回调函数
  @note  在相机类负载设备执行某项任务时无法切换工作模式
  @param mode:  相机工作的模式
  @return  执行结果
  T_GduReturnCode (*SetMode)(E_GduCameraMode mode);

  @brief  获取相机类负载设备当前工作模式的回调函数
  @param mode:  指向用于存储相机类负载设备工作模式的内存空间
  @return  执行结果
  T_GduReturnCode (*GetMode)(E_GduCameraMode *mode);

  @brief  控制相机类负载设备开始录像的回调函数
```

@return 执行结果
T_GduReturnCode (*StartRecordVideo)(void);

@brief 控制相机类负载设备停止录像的回调函数
@return 执行结果
T_GduReturnCode (*StopRecordVideo)(void);

@brief 控制相机类负载设备开始拍摄照片的回调函数
@return 执行结果
T_GduReturnCode (*StartShootPhoto)(void);

@brief 控制相机类负载设备停止拍摄照片的回调函数
@return 执行结果
T_GduReturnCode (*StopShootPhoto)(void);

@brief 设置相机类负载设备拍照模式的回调函数
@param mode: 相机拍照的模式
@return 执行结果
T_GduReturnCode (*SetShootPhotoMode)(E_GduCameraShootPhotoMode mode);

@brief 获取相机类负载设备当前拍照模式的回调函数
@param mode: 指向用于存储相机类负载设备拍照模式的内存空间
@return 执行结果
T_GduReturnCode (*GetShootPhotoMode)(E_GduCameraShootPhotoMode *mode);

@brief 设置相机类负载设备在连拍模式下，连拍张数的回调函数
@param burstCount: 相机连拍张数
@return 执行结果
T_GduReturnCode (*SetPhotoBurstCount)(E_GduCameraBurstCount burstCount);

@brief 获取相机类负载设备当前连拍张数的回调函数
@param burstCount: 指向用于存储相机类负载设备连拍张数的内存空间
@return 执行结果
T_GduReturnCode (*GetPhotoBurstCount)(E_GduCameraBurstCount *burstCount);

@brief 设置相机类负载设备在定时拍照模式下拍照间隔的回调函数
@note 在定时拍照模式下拍照的时间间隔范围为 2～255，当时间间隔设为 255 时，相机类负载设备将会持续拍摄照片
@param settings: 设置相机类负载设备在定时拍照模式下拍照的时间间隔
@return 执行结果
T_GduReturnCode (*SetPhotoTimeIntervalSettings)(T_GduCameraPhotoTimeIntervalSettings settings);

@brief 获取相机类负载设备在定时拍照模式下拍照间隔的回调函数
@param settings: 指向用于存储相机类负载设备在定时拍照模式下拍照间隔时间的内存空间
@return 执行结果

```
    T_GduReturnCode (*GetPhotoTimeIntervalSettings)(T_GduCameraPhotoTimeIntervalSettings *settings);

    @brief 获取相机类负载设备中当前 SD 卡状态的回调函数
    @param sdCardState: 指向用于存储相机类负载 SD 卡状态的内存空间
    @return 执行结果
    T_GduReturnCode (*GetSDCardState)(T_GduCameraSDCardState *sdCardState);

    @brief 控制相机类负载设备格式化 SD 卡的回调函数
    @return 执行结果
    T_GduReturnCode (*FormatSDCard)(void);

} T_GduCameraCommonHandler;
```

函数原型

function GduPayloadCamera_Init

功能：PSDK 相机模块初始化	product：all

负载设备相机模块初始化。

说明

- 使用相机类功能前，请先调用本接口初始化相机模块；
- 开发者可根据用户实际的使用需求，按照相机模块中的函数原型构造并注册回调函数，在使用相机类功能前，请先调用本接口

初始化相机模块。

T_GduReturnCode GduPayloadCamera_Init(void);

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode

function GduPayloadCamera_RegCommonHandler

功能：注册相机类负载设备的基础功能	product：all

注册相机类负载设备基础功能接口的句柄。

说明

- 在使用相机类负载设备的基础功能前，请开发者按照相机类基础功能的函数原型，构造控制。
- 相机类负载设备执行基础功能的回调函数，并将该回调函数注册到如下接口中。

```
T_GduReturnCode GduPayloadCamera_RegCommonHandler(const T_GduCameraCommonHandler *cameraCommonHandler);
```

参数

cameraCommonHandler：指向开发者实现的相机类负载设备基础功能的函数

返回值

根据程序执行的情况输出对应的返回值，详情请参见：GduErrorCode