G-Port HEQ Gimbal Communication Protocol

1、 Frame Structure Definition

Frame Header	Version	Length	Command	Frame Header Checksum	Data	Data Checksum

Frame structure field explanation

Field	Size (byte)	Data Type	Remarks
Frame Header	1	uint8_t	the starting value of 1 frame data is fixed to 0XAE
Version	1	uint8_t	protocol version, currently 0x01
Length	1	uint8_t	the length of the data segment content
Command	1	uint8_t	different commands correspond to different functions
Frame Header Checksum	1	uint8_t	Version, length, and command **checksum**
Data	N		Different meanings depending on different messages, see the **Protocol Details** section
Data Checksum	4	uint32_t	CRC32 value of data segment content, no data field means no CRC32 content. See **crc32.c** for CRC32 calculation code

Unless otherwise specified, the fields are all little-endian bytes. From the Assistant software/flight control/car to the gimbal is downlink, and vice versa is uplink.

2、 Protocol Details

2.1、Gimbal Function Reading

Command：0x13

Data field: None

Data flow: Downlink

Send example：AE 01 00 13 14

Example description: There is no data field, so there is no need to add CRC32 check. The same applies to the subsequent steps.

2.2、Gimbal Function Reading Return

command：0x14

Data field：Length: 15 bytes

Index	Type	Function Description	Remarks
0-10		Reserved	No practical use
11	uint8_t	Dead zone range	0-255
12	uint8_t	Following speed	When it is 0, it does not follow the aircraft head, other values follow the aircraft head
13	int8_t	Gimbal inversion	-1 means inverted, 1 means upright
14		Reserved	No practical use

Data flow：uplink

Return example：AE 01 0F 14 24 00 00 00 00 00 00 00 00 00 00 00 32 0A FF 00 26 37 1B BA

Example description: Dead zone range 32=50; follow speed 0A=10; FF=-1 inverted

2.3、Gimbal Function Setting

command：0x15

Data field：Length: 15 bytes

Index	Type	Function Description	Remarks

0-10		Reserved	No practical use
11	uint8_t	Dead zone range	0-255
12	uint8_t	Following speed	When it is 0, it does not follow the aircraft head, other values follow the aircraft head
13	int8_t	Gimbal inversion	-1 means inverted, 1 means upright
14		Reserved	No practical use

Data flow：downlink

Return example：AE 01 0F 15 25 00 00 00 00 00 00 00 00 00 00 00 32 0A FF 00 26 37 1B BA
Example description: Dead zone range 32=50; follow speed 0A=10; FF=-1 inverted

2.4、Gimbal Function Setting Return

Command：0x16

Data field：None

Data flow：uplink

Return example：AE 01 00 16 17

2.5、Gimbal Control Command

Command：0x85

Data field：Length: 13 bytes

Index	Type	Function Description	Remarks
0	int8_t	Control mode	1. Speed control 2. Angle control 3. gimbal return to center 4. Lock mode
1-2	int16 t	Roll angle control parameters	The angle unit is 0.01°
3-4	int16 t	Pitch angle control parameters	The angle unit is 0.01°
5-6	int16 t	Yaw angle control parameters	The angle unit is 0.01°

7-8	int16_t	Roll speed control parameters	The speed unit is 0.01°/s
9-10	int16_t	Pitch speed control parameters	The speed unit is 0.01°/s
11-12	int16_t	Yaw speed control parameters	The speed unit is 0.01°/s

Data flow：downlink

Send example：

Speed control Yaw 30°/sec: AE 01 0D 85 93 01 00 00 00 00 00 00 00 00 00 00 B8 0B CC F5 E1 63

Angle control: Rotate yaw to 30°: AE 01 0D 85 93 02 00 00 00 00 B8 0B 00 00 00 00 00 00 76 AB AF 70

Gimbal return to center: AE 01 0D 85 93 03 00 00 00 00 00 00 00 00 00 00 00 00 44 06 BE 68

Lock mode: AE 01 0D 85 93 04 00 00 00 00 00 00 00 00 00 00 00 00 9B 5B 72 2F

Note: Speed control is similar to joystick control. If only one piece of data is sent, the gimbal will move at that speed for 1 second and then stop. To achieve continuous control, it is recommended to send speed control to the gimbal at a frequency of 10HZ. If you want to stop immediately, you need to send a speed control command with a speed of 0. Otherwise, the gimbal will move at the speed sent by the last speed control command for 1 second and then stop.

2.6、Gimbal Angle Push(V1.0.0)

Command：0x87

Data field：Length: 12 bytes

Index	Type	Function Description	Remarks
0-1	int16 t	IMU_ROLL	IMU roll angle*100
2-3	int16 t	IMU_PICTH	IMU pitch angle*100
4-5	int16 t	IMU_YAW	IMU yaw angle*100
6-7	int16 t	Hall Angle_ROLL	Hall roll angle*100
8-9	int16 t	Hall Angle_PITCH	Hall pitch angle*100
10-11	int16 t	Hall Angle_YAW	Hall yaw angle*100

Data flow：uplink

Return example：AE 01 0C 87 94 00 00 00 00 A7 F0 7F F8 F0 FF D2 01 44 0D AD 53

Example description：

IMU roll angle 00 00 =0； IMU pitch angle 00 00 =0； IMU yaw angle A7 F0 =-3929=-39.29*100；

Hall roll angle 7F F8 =-1921=-19.21；Hall pitch angle F0 FF =-16=-0.16*100；Hall yaw angle D2 01 =466=4.66*100；

2.7、Gimbal Angle Push(V2.0.0)

command：0x87

Data field：Length: 24 bytes

Index	Type	Function Description	Remarks
0-1	int16_t	IMU_ROLL	IMU roll angle*100
2-3	int16_t	IMU_PICTH	IMU pitch angle*100
4-5	int16_t	IMU_YAW	IMU yaw angle*100
6-7	int16_t	Hall Angle_ROLL	Hall roll angle*100
8-9	int16_t	Hall Angle_PITCH	Hall pitch angle*100
10-11	int16_t	Hall Angle_YAW	Hall yaw angle*100
12-13	int16_t	Hall Angle ROLL_angular velocity	Hall roll angular velocity*100
14-15	int16_t	Hall Angle PICTH_angular velocity	Hall pitch angular velocity*100
16-17	int16_t	Hall Angle YAW_angular velocity	Hall yaw angular velocity*100
18-19	int16_t	IMU_X_angular velocity	IMU_X angular velocity*100
20-21	int16_t	IMU_Y_angular velocity	IMU_Y angular velocity*100
22-23	int16_t	IMU_Z_angular velocity	IMU_Z angular velocity*100

Data flow：uplink

Return example：AE 01 18 87 A0 00 00 00 00 AF 00 91 FF 03 00 00 00 F8 FF F1 FF 01 00 F8 FF F1 FF 01 00 E0 32 7E 13

Example description：

IMU roll angle 00 00 =0； IMU pitch angle 00 00 =0； IMU yaw angle AF 00 =175=1.75*100；

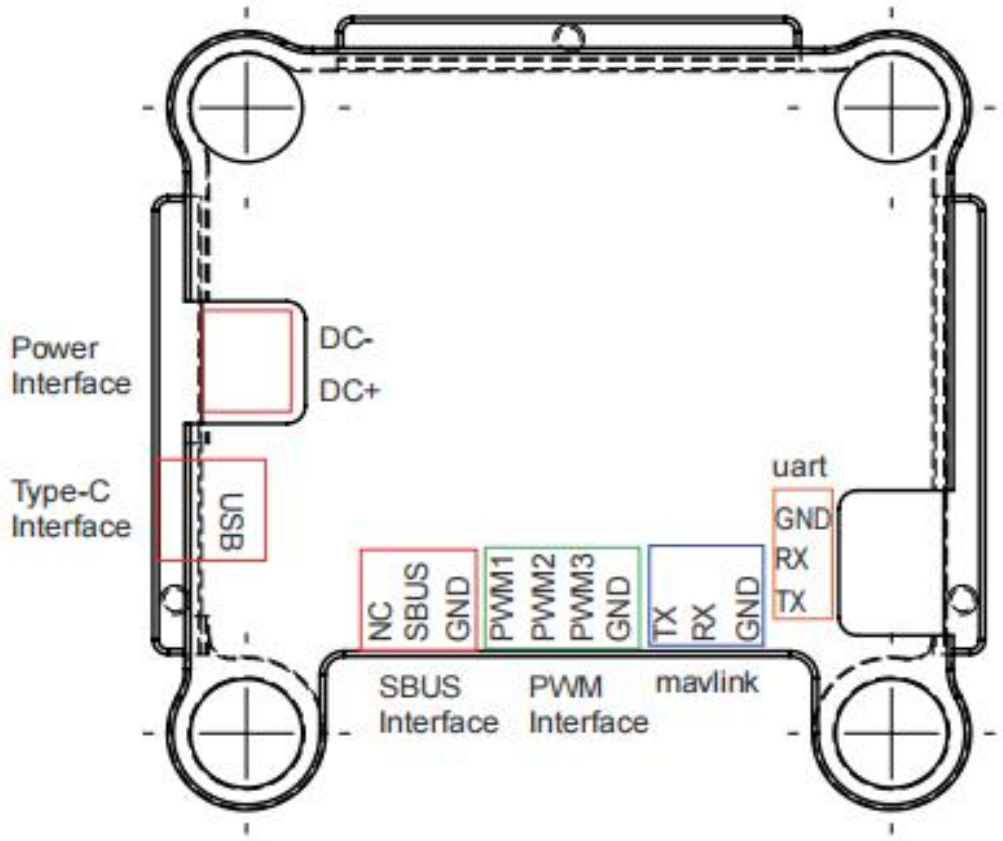
Hall roll angle 91 FF =-110=-1.10*100； Hall pitch angle 03 00 =3=0.03*100； Hall yaw angle 00 00 =0；

Hall roll angular velocity F8 FF =-7=-0.07*100; Hall pitch angular velocity F1 FF =-14=-0.14*100; Hall yaw angular velocity 01 00 =1=0.01*100;

IMU_X angular velocity F8 FF =-7=-0.07*100; IMU_Y angular velocity F1 FF =-14=-0.14*100; IMU_Z angular velocity 01 00 =1=0.01*100

3、 G-Port Serial Communication Precautions

The communication interface is as shown below: UART port; baud rate 115200; data bit 8bt, stop bit 1; no parity check; if the wiring and configuration are correct, the serial port will always receive the gimbal attitude report starting with AE.



Appendix, CRC32 calculation (C language version)

```
uint32 Crc32Table[ 256 ] =
{
  0x00000000, 0x04C11DB7, 0x09823B6E, 0x0D4326D9, 0x130476DC, 0x17C56B6B,
  0x1A864DB2, 0x1E475005, 0x2608EDB8, 0x22C9F00F, 0x2F8AD6D6, 0x2B4BCB61,
  0x350C9B64, 0x31CD86D3, 0x3C8EA00A, 0x384FBDBD, 0x4C11DB70, 0x48D0C6C7,
  0x4593E01E, 0x4152FDA9, 0x5F15ADAC, 0x5BD4B01B, 0x569796C2, 0x52568B75,
  0x6A1936C8, 0x6ED82B7F, 0x639B0DA6, 0x675A1011, 0x791D4014, 0x7DDC5DA3,
  0x709F7B7A, 0x745E66CD, 0x9823B6E0, 0x9CE2AB57, 0x91A18D8E, 0x95609039,
  0x8B27C03C, 0x8FE6DD8B, 0x82A5FB52, 0x8664E6E5, 0xBE2B5B58, 0xBAEA46EF,
  0xB7A96036, 0xB3687D81, 0xAD2F2D84, 0xA9EE3033, 0xA4AD16EA, 0xA06C0B5D,
  0xD4326D90, 0xD0F37027, 0xDDB056FE, 0xD9714B49, 0xC7361B4C, 0xC3F706FB,
  0xCEB42022, 0xCA753D95, 0xF23A8028, 0xF6FB9D9F, 0xFBB8BB46, 0xFF79A6F1,
  0xE13EF6F4, 0xE5FFEB43, 0xE8BCCD9A, 0xEC7DD02D, 0x34867077, 0x30476DC0,
  0x3D044B19, 0x39C556AE, 0x278206AB, 0x23431B1C, 0x2E003DC5, 0x2AC12072,
  0x128E9DCF, 0x164F8078, 0x1B0CA6A1, 0x1FCDBB16, 0x018AEB13, 0x054BF6A4,
  0x0808D07D, 0x0CC9CDCA, 0x7897AB07, 0x7C56B6B0, 0x71159069, 0x75D48DDE,
  0x6B93DDDB, 0x6F52C06C, 0x6211E6B5, 0x66D0FB02, 0x5E9F46BF, 0x5A5E5B08,
  0x571D7DD1, 0x53DC6066, 0x4D9B3063, 0x495A2DD4, 0x44190B0D, 0x40D816BA,
  0xACA5C697, 0xA864DB20, 0xA527FDF9, 0xA1E6E04E, 0xBFA1B04B, 0xBB60ADFC,
  0xB6238B25, 0xB2E29692, 0x8AAD2B2F, 0x8E6C3698, 0x832F1041, 0x87EE0DF6,
  0x99A95DF3, 0x9D684044, 0x902B669D, 0x94EA7B2A, 0xE0B41DE7, 0xE4750050,
  0xE9362689, 0xEDF73B3E, 0xF3B06B3B, 0xF771768C, 0xFA325055, 0xFEF34DE2,
  0xC6BCF05F, 0xC27DEDE8, 0xCF3ECB31, 0xCBFFD686, 0xD5B88683, 0xD1799B34,
  0xDC3ABDED, 0xD8FBA05A, 0x690CE0EE, 0x6DCDFD59, 0x608EDB80, 0x644FC637,
  0x7A089632, 0x7EC98B85, 0x738AAD5C, 0x774BB0EB, 0x4F040D56, 0x4BC510E1,
  0x46863638, 0x42472B8F, 0x5C007B8A, 0x58C1663D, 0x558240E4, 0x51435D53,
  0x251D3B9E, 0x21DC2629, 0x2C9F00F0, 0x285E1D47, 0x36194D42, 0x32D850F5,
  0x3F9B762C, 0x3B5A6B9B, 0x0315D626, 0x07D4CB91, 0x0A97ED48, 0x0E56F0FF,
  0x1011A0FA, 0x14D0BD4D, 0x19939B94, 0x1D528623, 0xF12F560E, 0xF5EE4BB9,
  0xF8AD6D60, 0xFC6C70D7, 0xE22B20D2, 0xE6EA3D65, 0xEBA91BBC, 0xEF68060B,
  0xD727BBB6, 0xD3E6A601, 0xDEA580D8, 0xDA649D6F, 0xC423CD6A, 0xC0E2D0DD,
  0xCDA1F604, 0xC960EBB3, 0xBD3E8D7E, 0xB9FF90C9, 0xB4BCB610, 0xB07DABA7,
  0xAE3AFBA2, 0xAAFBE615, 0xA7B8C0CC, 0xA379DD7B, 0x9B3660C6, 0x9FF77D71,
  0x92B45BA8, 0x9675461F, 0x8832161A, 0x8CF30BAD, 0x81B02D74, 0x857130C3,
  0x5D8A9099, 0x594B8D2E, 0x5408ABF7, 0x50C9B640, 0x4E8EE645, 0x4A4FFBF2,
  0x470CDD2B, 0x43CDC09C, 0x7B827D21, 0x7F436096, 0x7200464F, 0x76C15BF8,
  0x68860BFD, 0x6C47164A, 0x61043093, 0x65C52D24, 0x119B4BE9, 0x155A565E,
  0x18197087, 0x1CD86D30, 0x029F3D35, 0x065E2082, 0x0B1D065B, 0x0FDC1BEC,
  0x3793A651, 0x3352BBE6, 0x3E119D3F, 0x3AD08088, 0x2497D08D, 0x2056CD3A,
  0x2D15EBE3, 0x29D4F654, 0xC5A92679, 0xC1683BCE, 0xCC2B1D17, 0xC8EA00A0,
  0xD6AD50A5, 0xD26C4D12, 0xDF2F6BCB, 0xDBEE767C, 0xE3A1CBC1, 0xE760D676,
  0xEA23F0AF, 0xEEE2ED18, 0xF0A5BD1D, 0xF464A0AA, 0xF9278673, 0xFDE69BC4,
  0x89B8FD09, 0x8D79E0BE, 0x803AC667, 0x84FBDBD0, 0x9ABC8BD5, 0x9E7D9662,
  0x933EB0BB, 0x97FFAD0C, 0xAFB010B1, 0xAB710D06, 0xA6322BDF, 0xA2F33668,
  0xBCB4666D, 0xB8757BDA, 0xB5365D03, 0xB1F740B4 };
```

```
//Lookup table method
uint32 crc_32(uint8 *pData, uint16 Length)
{

  uint32nReg; //CRC Register
   uint32 nTemp = 0;
   uint16 i, n;

   nReg = 0xFFFFFFFF; //
   for ( n = 0; n < Length; n++ )
   {
     nReg ^= (uint32) pData [ n ];

     for ( i = 0; i < 4; i++ )
     {
       nTemp = Crc32Table [ ( uint8 )( ( nReg >> 24 ) & 0xff ) ]; //Take a byte and look up the table
       nReg <<= 8; //Discard the last calculated BYTE
       nReg ^= nTemp; //XOR with the calculation result of the previous BYTE
       }
   }
   return nReg;
}
```