Mavlink communication

In this document, we will explain how mavlink communication works in an UDP program.

Initialization

Firts of all you need to know how to initialize UDP connection for listen and send packets.

-Initialize listening This first step is the more simple, you only have to initialize a listening socket on the port 14550 (port where the controller send informations) and bind it.

```
struct sockaddr_in locAddr;
int sock = socket(PF_INET, SOCK_DGRAM, IPPROTO_UDP);

locAddr.sin_family = AF_INET;
locAddr.sin_addr.s_addr = INADDR_ANY;
locAddr.sin_port = htons(14550);
memset (&locAddr.sin_zero, 0, sizeof(locAddr.sin_zero));

if (-1 == bind(sock,(struct sockaddr *)&locAddr, sizeof(struct sockaddr))){
        perror("error bind failed");
        close(sock);
        return -1;
}
```

-Initialize sending This step is more complicated because the target port always changes. The solution that we have found is to listen, with the listening socket initialized previously, all incoming messages and stop when we receive one from the target IP : 10.1.1.1 (adress of the controller). Thus we get the target port and we can initialize the sending socket.

```
struct sockaddr_in targetAddr;
targetAddr.sin_family = AF_INET;
targetAddr.sin_addr.s_addr = inet_addr("10.1.1.1");
targetAddr.sin_port = possibleTarget.sin_port;
memset (&targetAddr.sin_zero, 0, sizeof(targetAddr.sin_zero));
```

Receive mavlink by UDP

-Buffer First of all you need intialize a buffer that you fill block of memory at each use with "memset" :

```
#define BUFFER_LENGTH 2041
...
uint8_t buf[BUFFER_LENGTH];
...
memset(buf, 0, BUFFER_LENGTH);
```

-Receiving After that, you receive bytes with de function "recvfrom".

```
recsize = recvfrom(sock, (void *)buf, BUFFER_LENGTH, 0, (struct sockaddr *)&targetAddr, &
    fromlen);
```

-Parsing Then you parse it in order to convert it in a mavlink message, thus you can interpret this message.

```
if (mavlink_parse_char(chan, buf[i], &msg, &status)){
        printf("\nReceived packet: SYS: %d, COMP: %d, LEN: %d, MSG ID: %d\n\n", msg.
            sysid, msg.compid, msg.len, msg.msgid);
}
```

Send mavlink by UDP

-Buffer First of all you need intialize a buffer that you fill block of memory at each use with "memset" like before :

```
#define BUFFER_LENGTH 2041
...
uint8_t buf[BUFFER_LENGTH];
...
memset(buf, 0, BUFFER_LENGTH);
```

-Packing Second step is to "pack" the mavlink message with the "pack function". There are numerous type of "pack function" and for various uses, we will dwell on these different types later, for the moment we will use the basic message "heartbeat" which is a frequent message to give "sign of life" of a device.

```
mavlink_msg_heartbeat_pack(255,0,&msg,MAV_TYPE_GCS,MAV_AUTOPILOT_ARDUPILOTMEGA,0xc0,0x0,
    MAV_STATE_ACTIVE);
```

-Converting After being pack, we put the mavlink message in the buffer with the function "mavlink_msg_to_send_buffer".

```
len = mavlink_msg_to_send_buffer(buf, &msg);
```

-Sending To finish, send the buffer with the function "sendto".

```
bytes_sent = sendto(sock, buf, len, 0, (struct sockaddr*)&targetAddr, sizeof(struct
    sockaddr_in));
```

Sources

- Mavlink library version 1 in c : https ://github.com/mavlink/c_library_v1
- Small example using the library in c : https ://mavlink.io/en/examples/c_udp.html
- IP adress communication : https ://unix.stackexchange.com/questions/320640/3dr-solo-drone-wifi-communication
- Port communication : https ://github.com/PX4/Devguide/issues/357