Working Principle For Game pad

1. It is a 2.4G RF product. It works at the frequency of ISM Band (2.4GHZ). There are up to 91 channels, and the frequency interval between each channel is 864K Hz.
2. Evenly Randomized Frequency Hopping Sequence method is implemented in the RF technology.
3. It works in "Slave" Mode. We call it "Device.

The working procedures are:

a) When power on, the Device will search all of the channels, to see whether there is a Host.
b) If there is a Host, the Device will identity by the data received, to see whether it can be connected with the Host.
c) If it can connect to the Host, then the Device will respond to the Host.
d) The Host builds up the connection after receiving the response from the Device.
e) The Host sends the command request to the Device for getting the Axes and Buttons value.
f) The Device sends Axes and Button value to the Host.
g) The Host identifies the data received and does the error detection and the error correction.
h) The Host sends Motor commands to the Device.
i) The Device will handle the Motor behaviors by the Motor data value that sent from the Host.
j) Repeat from step e) to step i).

Typical Product Characteristics:

Items	Description
Type of Modulation	GFSK
Number of Channels	91
Frequency Band	2402 MHz ~ 2479 MHz
Antenna Type	PCB Antenna
Testing Duty Cycle	100%
Test Power Source	DC 3V From battery
Temperature Range(Operating)	0 ~ 50 ℃

Basic Function:

- Fully compatible with NGC Platform
- Low power Indicator
- Power on/off selection
- Rubber grips
- Auto Player matching
- 10 Meter(30ft) 2.4G RF wireless transmitting & receiving distance

Working Principle For Host

4. It is a 2.4G RF product. It works at the frequency of ISM Band (2.4GHZ). There are up to 91 channels, and the frequency interval between each channel is 864K Hz.
5. Evenly Randomized Frequency Hopping Sequence method is implemented in the RF technology.
6. It works in "Master" Mode. We call it "Host."

The working procedures are:

k) When power on, the Host will do the frequency hopping according to a certain sequence, and then send the connection command.
l) If there is a Device response, the Host will judge whether it can be permitted to connect.
m) If it can be permitted to connect, then send the connection command to build up the connection.
n) The Host sends the request command to the Device to get the Axes and Buttons value.
o) The Device sends the Axes and Buttons value to the Host.
p) The Host will identify the data received and then do the data detection and data correction.
q) The Host will save the data received if there is no error.
r) The Host judges whether it is required to send Axes and Buttons value to NGC console.
s) If it is required, then send the saved data to NGC console.
t) The Host judges whether there is any Motor command and/or Motor value sent from NGC console.
u) If there is, then the Host will save the received Motor data.
v) The Host send Motor commands to the Device, and send the Motor value to the Device.
w) Repeat Step e) to Step l)
x)

Typical Product Characteristics:

Items	**Description**
Type of Modulation	GFSK
Number of Channels	91
Frequency Band	2402 MHz ~ 2479 MHz
Antenna Type	PCB Antenna
Testing Duty Cycle	100%
Test Power Source	DC 3.5V From NGC
Temperature Range(Operating)	0 ~ 50 ℃

Basic Function:

- A LED Indicate Link & Search Mode

Randomized Frequency Hopping Sequence Working Principle

The sequence of the jump frequency will be settle by the host. The method is the host to be connected to the device, The sequence data S(8bit) and HostID(16bit) will bring through scan the RSSI of RF Module in different channel, and the data will be transmitted to the device. Then the both side will calculate to the next number (F) by same agument(S).

The way of bring sequence is as follows:

N: the numbers of useful channel

S: number of seed (8bit)=S7-S0

P: replacement control (21bit)=P20-P0 ,Each 3 bit become one figure, as follow :

$Q_6=P_{20}P_{19}P_{18}$ $Q_5=P_{17}P_{16}P_{15}$ $Q_4=P_{14}P_{13}P_{12}$

$Q_3=P_{11}P_{10}P_9$ $Q_2=P_8P_7P_6$ $Q_1=P_5P_4P_3$

$Q_0=P_2P_1P_0$

$Q_i \neq Q_j$ $(0 \leq i<6, i<j \leq 6)$ to be request!

F: the figure of the sequence $(0 \leq F<N)$

Surpose the S to be S(k) in K cycle. It means the channel of the (K+1) cycle will be F(K+1) channel.

1. $S(k+1)=S(k)+1=S(k+1)_7 \sim S(k+1)_0$
2. S(k+1)'s Bit6~Bit0 will conversion to $R(k+1)=r(k+1)_7 \sim r(k+1)_0$

   $r(k+1)_i = S(k+1)_{Qi}$ $(0 \leq i<6)$, $r(k+1)_7 = s(k+1)_7$ to be request!
3. $F(k+1)=R(k+1) \bmod N$

So we got to know from above procedure, S data is come from the device ,then transmitter the data to the device. And the above replace control data P is calculate from 16bit HostID of host by the Host and Device both side.

Please reference to the following C program for to calculate method.

```
void FindFHSeed(unsigned char HostIDH, unsigned char HostIDL)
{
//-------------------------------------------------------------------------
//        The following code does the same job, but in C language format
  unsigned short b = 5040;
  unsigned short m;
  char i,j,k,l;
 m = (unsigned short)(HostIDH) *0x0100 + (unsigned short)(HostIDL);
```

```
  for (i=7; i>1; i--)
   {
     m %= b;
    b /= (unsigned short)i;
      q[i-1] = (unsigned char)(m / b);
   }
  q[0] = 0;
  for (i=6; i>0; i--)
   {
     for (j=i-1; j>=0; j--)
      {
       if (q[j] >= q[i]) q[j]++;
         do
         {
           l = 0;
          for (k=i; k<7; k++)
            {
              if (q[j] == q[k])
                {
                   q[j]++;
                 l = 1;
                }
            }
         }while (l!=0);
      }
   }
//---------------------------------------------------------------------
}
void NextFrequencyHop() //Calculate the frequency channel of the next hop
{                              //"seed"  and  "q[0]- q[6]" are known parameters.
   seed++;                  //Result  is  in  r18
  r11 = 0x01;
  if (seed & 0x80) r18 = 0x80;
```

```
  else r18 = 0x00;
  for (r10=0; r10<7; r10++)
   {
     if (r11 & seed)
      {
        r12 = q[r10];
        r18 |= errorPattern[r12];
      }
      r11 <<= 1;
   }
  while (r18 >= TOTAL_CHANNEL_NO) r18 -= TOTAL_CHANNEL_NO;
}
```