Cheat Sheet – RFM69HW on Arduino

Setting up:

- Create global variable: "RFM69 radio;"
- In setup initialise module:
  - "radio.initialise(FREQUENCY, ***NodeID***, ***NetworkID***);"
  - "radio.setHighPower();"
  - "radio.encrypt(***EncryptKey***);"

Example:

```
//Radio object
RFM69 radio;

void setup()
{
  Serial.begin(9600);   //Initialise serial interface for debugging
  radio.initialize(FREQUENCY,2,200);  //Initialise radio and set this node ID as 2, using channel 200
  radio.setHighPower(); //To use the high power capabilities of the RFM69HW
  radio.encrypt("sampleEncryptKey");
}
```

To send a message:

- Use: "radio.send(***ReceiverID, payload, payloadLength***);"

Example:

```
void loop()
{
  radio.send(1, "Hey", 3); //Send "Hey" to node 1. 3 characters.
  Serial.println("Send complete"); //Display debug message through USB COM port
  delay(500);
}
```

Example 1 to send a variable:

```
char payload[50];
void loop()
{
  float pi=3.1415;
  sprintf(payload,"I want to send this number: %d", (int)pi);
  Serial.print(payload);
  radio.send(1, payload, 50);
  Serial.println("Send complete");
  delay(500);
}
```

Note: "sprintf" prepares the payload string. "%d" defines a space where number after comma (variableToSend) will be inserted in string. Only integers can be inserted.

Tautvydas Mickus

Example 2 to send a variable:

```
char payload[50];
void loop()
{
  float pi=3.1415;
  String payloadstr; //create a payloadstr

  payloadstr+=String("Pi is ");  // build a message "Pi is 3.141"
  payloadstr+=String(pi,3); // 3 specifies the number of decimal places

  payloadstr.toCharArray(payload, 50); //convert this fancy String to a character array t

  Serial.println(payload); //debug message
  radio.send(1, payload, 50);
  Serial.println("Send complete");
  delay(500);
}
```

Note: Use “String” to prepare a payload. Allows floats to be incorporated

To receive message:

- Use “radio.receiveDone()” with If statement to detect reception
- Use “Serial.print((char*)radio.DATA);” to print data out

Example:

```
int packetCount=0;
void loop()
{
  if (radio.receiveDone())
  {
    Serial.print("#[");
    Serial.print(++packetCount);
    Serial.print(']');
    Serial.print('[');Serial.print(radio.SENDERID);Serial.print("] ");
    Serial.print((char*)radio.DATA);
    Serial.println();
  }
}
```

Note: it’s not necessary to use “packetCount” but it helps to identify if new messages are arriving when looking at Serial Monitor or equivalent.

Full Simple Transmitter example:

```
//Include the required libraries
#include <qbcan.h>
#include <Wire.h>
#include <SPI.h>

//Radio object
RFM69 radio;

void setup()
{
  Serial.begin(9600);   //Initialise serial interface for debugging
  radio.initialize(FREQUENCY,2,200);  //Initialise radio and set this node ID as 2, using channel 200
  radio.setHighPower(); //To use the high power capabilities of the RFM69HW
  radio.encrypt("sampleEncryptKey");
}
char payload[50];

void loop()
{
  int variableToSend;
  variableToSend=31415;
  sprintf(payload,"I want to send this number: %d", (int)variableToSend);
  Serial.print(payload);
  radio.send(1, payload, 50);
  Serial.println("Send complete");
  delay(500);
}
```

Full Simple Receiver (Ground station) example:

```
//Include the required libraries
#include <qbcan.h>
#include <Wire.h>
#include <SPI.h>

//Radio object
RFM69 radio;

void setup()
{
  Serial.begin(9600);   //Initialise serial interface for debugging
  radio.initialize(FREQUENCY,1,200);  //Initialise radio and set this node ID as 1, using channel 200
  radio.setHighPower(); //To use the high power capabilities of the RFM69HW
  radio.encrypt("sampleEncryptKey");
}

int packetCount=0;
void loop()
{
  if (radio.receiveDone())
  {
    Serial.print("#[");
    Serial.print(++packetCount);
    Serial.print(']');
    Serial.print('[');Serial.print(radio.SENDERID);Serial.print("] ");
    Serial.print((char*)radio.DATA);
    Serial.println();
  }
}
```

Notes:

- Additional functions and variables accessible via radio:
  - “radio.SENDERID” returns sender id
  - “radio.TARGETID” returns the destination of message (in case in sniffing mode)
  - “radio.RSSI” returns signal strength
  - “radio.promiscuous(true);” sets the receiver to sniffing mode (receive messages destined to other nodes)

  Example:

```
Serial.println(radio.SENDERID);
Serial.println(radio.TARGETID);
Serial.println(radio.RSSI);
```

Troubleshooting:

- Check if NetworkID on both nodes is the same
- Check if Node IDs are set correctly (different on both nodes) and that Transmitter is specifying the other nodes ID as packet destination.
- Check if Encrypt key is exactly the same on both nodes
- Try reducing the frequency of transmission (increasing delay).