* [**http://tmrh20.blogspot.com/2014/03/high-speed-data-transfers-and-wireless.html**](http://tmrh20.blogspot.com/2014/03/high-speed-data-transfers-and-wireless.html)
* [**http://tmrh20.github.io/RF24Network/index.html**](http://tmrh20.github.io/RF24Network/index.html)
* Base address should be 00
* To access the next available header/payload without advancing the next available message use **peek.**

while ( network.available() )

{

[RF24NetworkHeader](https://tmrh20.github.io/RF24Network/structRF24NetworkHeader.html) header;

uint32\_t time;

network.peek(header);

if(header.[type](https://tmrh20.github.io/RF24Network/structRF24NetworkHeader.html#ac99ab14e561670f4445d1fea7775e2db) == 'T')

{

network.read(header,&time,sizeof(time));

Serial.print("Got time: ");

Serial.println(time);

}

}

* **Write**

uint32\_t time = millis();

uint16\_t to = 00; // Send to master

[RF24NetworkHeader](https://tmrh20.github.io/RF24Network/structRF24NetworkHeader.html) header(to, 'T'); **// Send header type 'T'**

network.write(header,&time,sizeof(time));

* Send a **multicast** message to multiple nodes at once Allows messages to be rapidly broadcast through the network.

Multicasting is arranged in levels, with all nodes on the same level listening to the same address Levels are assigned by network level ie: nodes 01-05: Level 1, nodes 011-055: Level 2

|  |  |  |  |
| --- | --- | --- | --- |
| multicast | ( | [RF24NetworkHeader](https://tmrh20.github.io/RF24Network/structRF24NetworkHeader.html) & | *header*, |
|  |  | const void \* | *message*, |
|  |  | uint16\_t | *len*, |
|  |  | uint8\_t | *level* |
|  | ) |  |  |

* By default, multicast addresses are divided into levels.

Nodes 1-5 share a multicast address, nodes n1-n5 share a multicast address, and nodes n11-n55 share a multicast address.

This option is used to override the defaults, and create custom multicast groups that all share a single address.   
The level should be specified in decimal format 1-6

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| multicastLevel | ( | uint8\_t | *level* | ) |  |

* Enabling this will allow this node to automatically forward received multicast frames to the next highest multicast level.

Duplicate frames are filtered out, so multiple forwarding nodes at the same level should not interfere. Forwarded payloads will also be received.

**multicastRelay**

* **Writing to specific node**

This allows routing or sending messages outside of the usual routing paths. The same as write, but a physical address is specified as the last option. The payload will be written to the physical address, and routed as necessary by the recipient

|  |  |  |  |
| --- | --- | --- | --- |
| write | ( | [RF24NetworkHeader](https://tmrh20.github.io/RF24Network/structRF24NetworkHeader.html) & | *header*, |
|  |  | const void \* | *message*, |
|  |  | uint16\_t | *len*, |
|  |  | uint16\_t | *writeDirect* |
|  | ) |  |  |

* Use **txTimeout** to designate different timeout period for the transmitters so that the data do no collide (only use when the node is a leaf node, and it only transmits and doesnot receive)……(for transceiver node see **setRetries** function also). <http://tmrh20.github.io/RF24Network/Tuning.html>