

RadioHead Packet Radio library for embedded microprocessors

This is the RadioHead Packet Radio library for embedded microprocessors. It provides a complete object-oriented library for sending and receiving packetized messages via a variety of common data radios and other transports on a range of embedded microprocessors.

The version of the package that this documentation refers to can be downloaded from

<http://www.airspayce.com/mikem/arduino/RadioHead/RadioHead-1.101.zip> You can find the latest version of the documentation at <http://www.airspayce.com/mikem/arduino/RadioHead>

You can also find online help and discussion at <http://groups.google.com/group/radiohead-arduino> Please use that group for all questions and discussions on this topic. Do not contact the author directly, unless it is to discuss commercial licensing. Before asking a question or reporting a bug, please read

- http://en.wikipedia.org/wiki/Wikipedia:Reference_desk/How_to_ask_a_software_question
- <http://www.catb.org/esr/faqs/smart-questions.html>
- <http://www.chiark.greenend.org.uk/~shgtatham/bugs.html>

Caution: Developing this type of software and using data radios successfully is challenging and requires a substantial knowledge base in software and radio and data transmission technologies and theory. It may not be an appropriate project for beginners. If you are a beginner, you will need to spend some time gaining knowledge in these areas first.

Overview

RadioHead consists of 2 main sets of classes: Drivers and Managers.

- Drivers provide low level access to a range of different packet radios and other packetized message transports.
- Managers provide high level message sending and receiving facilities for a range of different requirements.

Every RadioHead program will have an instance of a Driver to provide access to the data radio or transport, and usually a Manager that uses that driver to send and receive messages for the application. The programmer is required to instantiate a Driver and a Manager, and to initialise the Manager. Thereafter the facilities of the Manager can be used to send and receive messages.

It is also possible to use a Driver on its own, without a Manager, although this only allows unaddressed, unreliable transport via the Driver's facilities.

In some specialised use cases, it is possible to instantiate more than one Driver and more than one Manager.

A range of different common embedded microprocessor platforms are supported, allowing your project to run on your choice of processor.

Example programs are included to show the main modes of use.

Drivers

The following Drivers are provided:

- **RH_RF22** Works with Hope-RF RF22B and RF23B based transceivers, and compatible chips and modules, including the RFM22B transceiver module such as this bare module: <http://www.sparkfun.com/products/10153> and this shield: <http://www.sparkfun.com/products/11018> and this board: <http://www.anarduino.com/miniwireless> and RF23BP modules such as: <http://www.anarduino.com/details.jsp?pid=130> Supports GFSK, FSK and OOK. Access to other chip features such as on-chip temperature measurement, analog-digital converter, transmitter power control etc is also provided.
- **RH_RF24** Works with Silicon Labs Si4460/4461/4463/4464 family of transceivers chip, and the equivalent HopeRF RF24/26/27 family of chips and the HopeRF RFM24W/26W/27W modules. Supports GFSK, FSK and OOK. Access to other chip features such as on-chip temperature measurement, analog-digital converter, transmitter power control etc is also provided.
- **RH_RF69** Works with Hope-RF RF69B based radio modules, such as the RFM69 module, (as used on the excellent Moteino and Moteino-USB boards from LowPowerLab <http://lowpowerlab.com/moteino/>) and compatible chips and modules such as RFM69W, RFM69HW, RFM69CW, RFM69HCW (Semtech SX1231, SX1231H). Also works with Anarduino MiniWireless -CW and -HW boards <http://www.anarduino.com/miniwireless/> including the marvellous high powered MinWireless-HW (with 20dBm output for excellent range). Supports GFSK, FSK.
- **RH_NRF24** Works with Nordic nRF24 based 2.4GHz radio modules, such as nRF24L01 and others. Also works with Hope-RF RFM73 and compatible devices (such as BK2423). nRF24L01 and RFM73 can interoperate with each other.
- **RH_NRF905** Works with Nordic nRF905 based 433/868/915 MHz radio modules.
- **RH_NRF51** Works with Nordic nRF51 compatible 2.4 GHz SoC/devices such as the nRF51822. Also works with Sparkfun nRF52832 breakout board, with Arduino 1.8.9 and Sparkfun nRF52 boards manager 0.2.3.
- **RH_RF95** Works with Semtech SX1276/77/78/79, Modtronix inAir4 and inAir9, and HopeRF RFM95/96/97/98 and other similar LoRa capable radios. Supports Long Range (LoRa) with spread spectrum frequency hopping, large payloads etc. FSK/GFSK/OOK modes are not (yet) supported.
- **RH_MRF89** Works with Microchip MRF89XA and compatible transceivers. and modules such as MRF89XAM9A.
- **RH_CC110** Works with Texas Instruments CC110L transceivers and compatible modules such as Anaren AIR BoosterPack 430BOOST-CC110L
- **RH_E32** Works with EBYTE E32-TTL-1W serial radio transceivers (and possibly other transceivers in the same family)
- **RH_ASK** Works with a range of inexpensive ASK (amplitude shift keying) RF transceivers such as RX-B1 (also known as ST-RX04-ASK) receiver; TX-C1 transmitter and DR3100 transceiver; FS1000A/XY-MK-5V transceiver; HopeRF RFM83C / RFM85. Supports ASK (OOK).
- **RH_Serial** Works with RS232, RS422, RS485, RS488 and other point-to-point and multidropped serial connections, or with TTL serial UARTs such as those on Arduino and many other processors, or with data radios with a serial port interface. **RH_Serial** provides packetization and error detection over any hardware or virtual serial connection. Also builds and runs on Linux and OSX.
- **RH_TCP** For use with simulated sketches compiled and running on Linux. Works with tools/etherSimulator.pl to pass messages between simulated sketches, allowing testing of Manager classes on Linux and without need for real radios or other transport hardware.
- **RHEncryptedDriver** Adds encryption and decryption to any RadioHead transport driver, using any encryption cipher supported by ArduinoLibs Cryptographic Library <http://rweather.github.io/arduinoilibs/crypto.html>

Drivers can be used on their own to provide unaddressed, unreliable datagrams. All drivers have the same identical API. Or you can use any Driver with any of the Managers described below.

We welcome contributions of well tested and well documented code to support other transports.

If your radio or transceiver is not on the list above, there is a good chance it wont work without modifying RadioHead to suit it. If you wish for support for another radio or transceiver, and you send 2 of them to AirSpayce Pty Ltd, we will consider adding support for it.

Managers

The drivers above all provide for unaddressed, unreliable, variable length messages, but if you need more than that, the following Managers are provided:

- **RHDatagram** Addressed, unreliable variable length messages, with optional broadcast facilities.
- **RHReliableDatagram** Addressed, reliable, retransmitted, acknowledged variable length messages.
- **RHRouter** Multi-hop delivery of RHReliableDatagrams from source node to destination node via 0 or more intermediate nodes, with manual, pre-programmed routing.
- **RHMesh** Multi-hop delivery of RHReliableDatagrams with automatic route discovery and rediscovery.

Any Manager may be used with any Driver.

Platforms

A range of processors and platforms are supported:

- Arduino and the Arduino IDE (version 1.0 to 1.8.1 and later) Including Diecimila, Uno, Mega, Leonardo, Yun, Due, Zero etc. <http://arduino.cc/>, Also similar boards such as
 - Moteino <http://lowpowerlab.com/moteino/>
 - Anarduino Mini <http://www.anarduino.com/mini/>
 - RedBearLab Blend V1.0 <http://redbearlab.com/blend/> (with Arduino 1.0.5 and RedBearLab Blend Add-On version 20140701)
 - MoteinoMEGA <https://lowpowerlab.com/shop/moteinomega> (with Arduino 1.0.5 and the MoteinoMEGA Arduino Core <https://github.com/LowPowerLab/Moteino/tree/master/MEGA/Core>)
 - ESP8266 on Arduino IDE and Boards Manager per <https://github.com/esp8266/Arduino> Tested using Arduino 1.6.8 with esp8266 by ESP8266 Community version 2.1.0 Also Arduino 1.8.1 with esp8266 by SparkFun Electronics 2.5.2 Examples serial_reliable_datagram_* and ask_* are shown to work. CAUTION: The GHz radio included in the ESP8266 is not yet supported. CAUTION: tests here show that when powered by an FTDI USB-Serial converter, the ESP8266 can draw so much power when transmitting on its GHz WiFi that VCC will sag causing random crashes. We strongly recommend a large cap, say 1000uF 10V on VCC if you are also using the WiFi.
 - Various Talk2 Whisper boards eg <https://wisen.com.au/store/products/whisper-node-lora>. Use Arduino Board Manager to install the Talk2 code support.
 - etc.
- STM32 F4 Discover board, using Arduino 1.8.2 or later and Roger Clarkes Arduino_STM from https://github.com/rogerclarkmelbourne/Arduino_STM32 Caution: with this library and board, sending text to Serial causes the board to hang in mysterious ways. Serial2 emits to PA2. The default SPI pins are SCK: PB3, MOSI PB5, MISO PB4. We tested with PB0 as slave select and PB1 as interrupt pin for

- various radios. **RH_ASK** and **RH_Serial** also work. Also works with stm32duino 1.8.0 from https://github.com/stm32duino/Arduino_Core_STM32, which can be installed on Arduino with BoardManager. Select board: STM32 Discovery F407.
- ChipKIT Core with Arduino IDE on any ChipKIT Core supported Digilent processor (tested on Uno32) http://chipkit.net/wiki/index.php?title=ChipKIT_core
 - Maple and Flymaple boards with libmaple and the Maple-IDE development environment <http://leaflabs.com/devices/maple/> and <http://www.open-drone.org/flymaple>
 - Teensy including Teensy 3.1 and earlier built using Arduino IDE 1.0.5 to 1.6.4 and later with teensyduino addon 1.18 to 1.23 and later. <http://www.pjrc.com/teensy>
 - Particle Photon <https://store.particle.io/collections/photon> and ARM3 based CPU with built-in Wi-Fi transceiver and extensive IoT software support. RadioHead does not support the built-in transceiver but can be used to control other SPI based radios, Serial ports etc. See below for details on how to build RadioHead for Photon
 - ATtiny built using Arduino IDE 1.8 and the ATtiny core from https://raw.githubusercontent.com/damellis/attiny/ide-1.6.x-boards-manager/package_damellis_attiny_index.json using the instructions at <https://medium.com/jungletronics/attiny85-easy-flashing-through-arduino-b5f896c48189> (Caution: these are very small processors and not all RadioHead features may be available, depending on memory requirements) (Caution: we have not had good success building **RH_ASK** sketches for ATtiny 85 with SpenceKonde ATtinyCore)
 - ATtiny Mega (tinyAVR 1-series) chips supported by Spencer Konde's megaTinyCore (<https://github.com/SpenceKonde/megaTinyCore>) (on Arduino 1.8.9 or later) such as AtTiny 3216, ATtiny 1616 etc. These chips can be easily programmed through their UPDI pin, using an ordinary Arduino board programmed as a jtag2updi programmer as described in <https://github.com/SpenceKonde/megaTinyCore/blob/master/MakeUPDIProgrammer.md>. Make sure you set the programmer type to jtag2updi in the Arduino Tools->Programmer menu. See <https://github.com/SpenceKonde/megaTinyCore/blob/master/megaavr/extras/ImportantInfo.md> for links to pinouts and pin numbering information for all the supported chips.
 - nRF51 compatible Arm chips such as nRF51822 with Arduino 1.6.4 and later using the procedures in <http://redbearlab.com/getting-started-nrf51822/>
 - nRF52 compatible Arm chips such as as Adafruit BLE Feather board <https://www.adafruit.com/product/3406>
 - Adafruit Feather. These are excellent boards that are available with a variety of radios. We tested with the Feather 32u4 with RFM69HCW radio, with Arduino IDE 1.6.8 and the Adafruit AVR Boards board manager version 1.6.10. <https://www.adafruit.com/products/3076>
 - Adafruit Feather M0 boards with Arduino 1.8.1 and later, using the Arduino and Adafruit SAMD board support. <https://learn.adafruit.com/adafruit-feather-m0-basic-protocol/using-with-arduino-ide>
 - ESP32 built using Arduino IDE 1.8.9 or later using the ESP32 toolchain installed per <https://github.com/espressif/arduino-esp32> The internal 2.4GHz radio is not yet supported. Tested with RFM22 using SPI interface
 - Raspberry Pi Uses BCM2835 library for GPIO <http://www.airspayce.com/mikem/bcm2835/> Currently works only with **RH_NRF24** driver or other drivers that do not require interrupt support. Contributed by Mike Poulton.
 - Linux and OSX Using the RHutil/HardwareSerial class, the **RH_Serial** driver and any manager will build and run on Linux and OSX. These can be used to build programs that talk securely and reliably to Arduino and other processors or to other Linux or OSX hosts on a reliable, error detected (and possibly encrypted) datagram protocol over various types of serial line.

- Mongoose OS, courtesy Paul Austen. Mongoose OS is an Internet of Things Firmware Development Framework available under Apache License Version 2.0. It supports low power, connected microcontrollers such as: ESP32, ESP8266, TI CC3200, TI CC3220, STM32. <https://mongoose-os.com/>

Other platforms are partially supported, such as Generic AVR 8 bit processors, MSP430. We welcome contributions that will expand the range of supported platforms.

If your processor is not on the list above, there is a good chance it won't work without modifying RadioHead to suit it. If you wish for support for another processor, and you send 2 of them to AirSpayce Pty Ltd, we will consider adding support for it.

RadioHead is available (through the efforts of others) for PlatformIO. PlatformIO is a cross-platform code builder and the missing library manager. <http://platformio.org/#!/lib/show/124/RadioHead>

History

RadioHead was created in April 2014, substantially based on code from some of our other earlier Radio libraries:

- **RHMesh**, **RHRouter**, **RHReliableDatagram** and **RHDatagram** are derived from the RF22 library version 1.39.
- **RH_RF22** is derived from the RF22 library version 1.39.
- **RH_RF69** is derived from the RF69 library version 1.2.
- **RH_ASK** is based on the VirtualWire library version 1.26, after significant conversion to C++.
- **RH_Serial** was new.
- **RH_NRF24** is based on the NRF24 library version 1.12, with some significant changes.

During this combination and redevelopment, we have tried to retain all the processor dependencies and support from the libraries that were contributed by other people. However not all platforms can be tested by us, so if you find that support from some platform has not been successfully migrated, please feel free to fix it and send us a patch.

Users of **RHMesh**, **RHRouter**, **RHReliableDatagram** and **RHDatagram** in the previous RF22 library will find that their existing code will run mostly without modification. See the **RH_RF22** documentation for more details.

Installation

Install in the usual way: unzip the distribution zip file to the libraries sub-folder of your sketchbook. The example sketches will be visible in in your Arduino, mpide, maple-ide or whatever. <http://arduino.cc/en/Guide/Libraries>

Building for Particle Photon

The Photon is not supported by the Arduino IDE, so it takes a little effort to set up a build environment. Here's what we did to enable building of RadioHead example sketches on Linux, but there are other ways to skin this cat. Basic reference for getting started is: <http://particle-firmware.readthedocs.org/en/develop/build/>

- Download the ARM gcc cross compiler binaries and unpack it in a suitable place:

```
cd /tmp
```



```
wget https://launchpad.net/gcc-arm-embedded/5.0/5-2015-q4-major/+download/gcc-arm-none-eabi-5_2-2015q4-20151219-linux.tar.bz2
tar xvf gcc-arm-none-eabi-5_2-2015q4-20151219-linux.tar.bz2
```

- If dfu-util and friends not installed on your platform, download dfu-util and friends to somewhere in your path

```
cd ~/bin
wget http://dfu-util.sourceforge.net/releases/dfu-util-0.8-binaries/linux-i386/dfu-util
wget http://dfu-util.sourceforge.net/releases/dfu-util-0.8-binaries/linux-i386/dfu-suffix
wget http://dfu-util.sourceforge.net/releases/dfu-util-0.8-binaries/linux-i386/dfu-prefix
```

- Download the Particle firmware (contains headers and libraries require to compile Photon sketches) to a suitable place:

```
cd /tmp
wget https://github.com/spark/firmware/archive/develop.zip
unzip develop.zip
```

- Make a working area containing the RadioHead library source code and your RadioHead sketch. You must rename the sketch from .pde or .ino to application.cpp

```
cd /tmp
mkdir RadioHead
cd RadioHead
cp /usr/local/projects/arduino/libraries/RadioHead/*.h .
cp /usr/local/projects/arduino/libraries/RadioHead/*.cpp .
cp /usr/local/projects/arduino/libraries/RadioHead/examples/cc110/cc110_client/cc110_client.pde application.cpp
```

- Edit application.cpp and comment out any #include <SPI.h> so it looks like:

```
// #include <SPI.h>
```

- Connect your Photon by USB. Put it in DFU mode as descibed in Photon documentation. Light should be flashing yellow
- Compile the RadioHead sketch and install it as the user program (this does not update the rest of the Photon firmware, just the user part:

```
cd /tmp/firmware-develop/main
PATH=$PATH:/tmp/gcc-arm-none-eabi-5_2-2015q4/bin make APPDIR=/tmp/RadioHead all PLATFORM=photon program=dfu
```

- You should see RadioHead compile without errors and download the finished sketch into the Photon.

Compatible Hardware Suppliers

We have had good experiences with the following suppliers of RadioHead compatible hardware:

- LittleBird <http://littlebirdelectronics.com.au> in Australia for all manner of Arduinos and radios.
- LowPowerLab <http://lowpowerlab.com/moteino> in USA for the excellent Moteino and Moteino-USB boards which include Hope-RF RF69B radios on-board.
- Anarduino and HopeRF USA (<http://www.hoperfusa.com> and <http://www.anarduino.com>) who have a wide range of HopeRF radios and Arduino integrated modules.
- SparkFun <https://www.sparkfun.com/> in USA who design and sell a wide range of Arduinos and radio modules.
- Wisen <http://wisen.com.au> who design and sell a wide range of integrated radio/processor modules including the excellent Talk2 range.

Coding Style

RadioHead is designed so it can run on small processors with very limited resources and strict timing constraints. As a result, we tend only to use the simplest and least demanding (in terms of memory and CPU)

C++ facilities. In particular we avoid as much as possible dynamic memory allocation, and the use of complex objects like C++ strings, IO and buffers. We are happy with this, but we are aware that some people may think we are legaving useful tools on the table. You should not use this code as an example of how to do generalised C++ programming on well resourced processors.

Donations

This library is offered under a free GPL license for those who want to use it that way. We try hard to keep it up to date, fix bugs and to provide free support. If this library has helped you save time or money, please consider donating at <http://www.airspayce.com> or here:



Passing Sensor Data Between RadioHead nodes

Trademarks

RadioHead is a trademark of AirSpayce Pty Ltd. The RadioHead mark was first used on April 12 2014 for international trade, and is used only in relation to data communications hardware and software and related services. It is not to be confused with any other similar marks covering other goods and services.

Copyright

This software is Copyright (C) 2011-2018 Mike McCauley. Use is subject to license conditions. The main licensing options available are GPL V3 or Commercial:

Open Source Licensing GPL V3

This is the appropriate option if you want to share the source code of your application with everyone you distribute it to, and you also want to give them the right to share who uses it. If you wish to use this software under Open Source Licensing, you must contribute all your source code to the open source community in accordance with the GPL Version 3 when your application is distributed. See <https://www.gnu.org/licenses/gpl-3.0.html>

Commercial Licensing

This is the appropriate option if you are creating proprietary applications and you are not prepared to distribute and share the source code of your application. To purchase a commercial license, contact info@airspayce.com

Revision History

Version

1.1 2014-04-14

Initial public release

1.2 2014-04-23

Fixed various typos.

Added links to compatible Anarduino products.

Added [RHNRFSPIDriver](#), [RH_NRF24](#) classes to support Nordic NRF24 based radios.

1.3 2014-04-28

Various documentation fixups.

RHDatagram::setThisAddress() did not set the local copy of thisAddress. Reported by Steve Childress.

Fixed a problem on Teensy with RF22 and RF69, where the interrupt pin needs to be set for input, else pin interrupt doesn't work properly. Reported by Steve Childress and patched by Adrien van den Bossche. Thanks.

Fixed a problem that prevented RF22 honouring setPromiscuous(true). Reported by Steve Childress. Updated documentation to clarify some issues to do with maximum message lengths reported by Steve Childress.

Added support for yield() on systems that support it (currently Arduino 1.5.5 and later) so that spin-loops can support multitasking. Suggested by Steve Childress.

Added **RH_RF22::setGpioReversed()** so the reversal it can be configured at run-time after radio initialisation. It must now be called *after* init(). Suggested by Steve Childress.

1.4 2014-04-29

Fixed further problems with Teensy compatibility for **RH_RF22**. Tested on Teensy 3.1. The example/rf22_* examples now run out of the box with the wiring connections as documented for Teensy in **RH_RF22**.

Added YIELDS to spin-loops in **RHRouter**, **RHMesh** and **RHReliableDatagram**, **RH_NRF24**.

Tested **RH_Serial** examples with Teensy 3.1: they now run out of the box.

Tested **RH_ASK** examples with Teensy 3.1: they now run out of the box.

Reduced default SPI speed for NRF24 from 8MHz to 1MHz on Teensy, to improve reliability when poor wiring is in use.

on some devices such as Teensy.

Tested **RH_NRF24** examples with Teensy 3.1: they now run out of the box.

1.5 2014-04-29

Added support for Nordic Semiconductor nRF905 transceiver with **RH_NRF905** driver. Also added examples for nRF905 and tested on Teensy 3.1

1.6 2014-04-30

NRF905 examples were missing

1.7 2014-05-03

Added support for Arduino Due. Tested with **RH_NRF905**, **RH_Serial**, **RH_ASK**. IMPORTANT CHANGE to interrupt pins on Arduino with **RH_RF22** and **RH_RF69** constructors: previously, you had to specify the interrupt *number* not the interrupt *pin*. Arduinos and Uno32 are now consistent with all other platforms: you must specify the interrupt pin number. Default changed to pin 2 (a common choice with RF22 shields). Removed examples/maple/maple_rf22_reliable_datagram_client and examples/maple/maple_rf22_reliable_datagram_client since the rf22 examples now work out of the box with Flymaple. Removed examples/uno32/uno32_rf22_reliable_datagram_client and examples/uno32/uno32_rf22_reliable_datagram_client since the rf22 examples now work out of the box with ChipKit Uno32.

1.8 2014-05-08

Added support for YIELD in Teensy 2 and 3, suggested by Steve Childress.

Documentation updates. Clarify use of headers and Flags

Fixed misalignment in **RH_RF69** between ModemConfigChoice definitions and the implemented choices which meant you didnt get the choice you thought and GFSK_Rb55555Fd50 hung the

transmitter.

Preliminary work on Linux simulator.

1.9 2014-05-14

Added support for using Timer 2 instead of Timer 1 on Arduino in **RH_ASK** when

RH_ASK_ARDUINO_USE_TIMER2 is defined. With the kind assistance of Luc Small. Thanks!

Updated comments in **RHReliableDatagram** concerning servers, retries, timeouts and delays. Fixed an error in **RHReliableDatagram** where `recvfrom` return value was not checked. Reported by Steve Childress.

Added Linux simulator support so simple RadioHead sketches can be compiled and run on Linux.

Added **RH_TCP** driver to permit message passing between simulated sketches on Linux.

Added example simulator sketches.

Added `tools/etherSimulator.pl`, a simulator of the 'Luminiferous Ether' that passes messages between simulated sketches and can simulate random message loss etc.

Fixed a number of typos and improved some documentation.

1.10 2014-05-15

Added support for RFM73 modules to **RH_NRF24**. These 2 radios are very similar, and can interoperate with each other. Added new **RH_NRF24::TransmitPower** enums for the RFM73, which has a different range of available powers

reduced the default SPI bus speed for **RH_NRF24** to 1MHz, since so many modules and CPU have problems with 8MHz.

1.11 2014-05-18

Testing **RH_RF22** with RFM23BP and 3.3V Teensy 3.1 and 5V Arduinos. Updated documentation with respect to GPIO and antenna control pins for RFM23. Updated documentation with respect to transmitter power control for RFM23

Fixed a problem with **RH_RF22** driver, where GPIO TX and RX pins were not configured during initialisation, causing poor transmit power and sensitivity on those RF22/RF23 devices where GPIO controls the antenna selection pins.

1.12 2014-05-20

Testing with RF69HW and the **RH_RF69** driver. Works well with the Anarduino MiniWireless -CW and -HW boards <http://www.anarduino.com/miniwireless/> including the marvellous high powered MinWireless-HW (with 20dBm output for excellent range).

Clarified documentation of **RH_RF69::setTxPower** values for different models of RF69.

Added **RHReliableDatagram::resetRetransmissions()**.

Retransmission count precision increased to `uint32_t`.

Added data about actual power measurements from RFM22 module.

1.13 2014-05-23

`setHeaderFlags(flags)` changed to `setHeaderFlags(set, clear)`, enabling any flags to be individually set and cleared by either RadioHead or application code. Requested by Steve Childress.

Fixed power output setting for boost power on RF69HW for 18, 19 and 20dBm.

Added data about actual power measurements from RFM69W and RFM69HW modules.

1.14 2014-05-26

RH_RF69::init() now always sets the PA boost back to the default settings, else can get invalid PA power modes after uploading new sketches without a power cycle. Reported by Bryan.

Added new macros **RH_VERSION_MAJOR** **RH_VERSION_MINOR**, with automatic maintenance in

Makefile.

Improvements to **RH_TCP**: constructor now honours the server argument in the form "servername:port".

Added YIELD to **RHReliableDatagram::recvfromAckTimeout**. Requested by Steve Childress.

Fixed a problem with **RH_RF22** reliable datagram acknowledgements that was introduced in version 1.13. Reported by Steve Childress.

1.15 2014-05-27

Fixed a problem with the RadioHead .zip link.

1.16 2014-05-30

Fixed **RH_RF22** so that lastRssi() returns the signal strength in dBm. Suggested by Steve Childress.

Added support for getLastPreambleTime() to **RH_RF69**. Requested by Steve Childress.

RH_NRF24::init() now checks if there is a device connected and responding, else init() will fail.

Suggested by Steve Brown.

RHSoftwareSPI now initialises default values for SPI pins MOSI = 12, MISO = 11 and SCK = 13.

Fixed some problems that prevented **RH_NRF24** working with mixed software and hardware SPI on different devices: a race condition due to slow SPI transfers and fast acknowledgement.

1.17 2014-06-02

Fixed a debug typo in **RHReliableDatagram** that was introduced in 1.16.

RH_NRF24 now sets default power, data rate and channel in init(), in case another app has previously set different values without powerdown.

Caution: there are still problems with **RH_NRF24** and Software SPI. Do not use.

1.18 2014-06-02

Improvements to performance of **RH_NRF24** statusRead, allowing **RH_NRF24** and Software SPI to operate on slow devices like Arduino Uno.

1.19 2014-06-19

Added examples ask_transmitter.pde and ask_receiver.pde.

Fixed an error in the **RH_RF22** doc for connection of Teensy to RF22.

Improved documentation of start symbol bit patterns in RH_ASK.cpp

1.20 2014-06-24

Fixed a problem with compiling on platforms such as ATtiny where SS is not defined.

Added YIELD to **RHMesh::recvfromAckTimeout()**.

1.21 2014-06-24

Fixed an issue in **RH_Serial** where characters might be lost with back-to-back frames. Suggested by Steve Childress.

Brought previous RHutil/crc16.h code into mainline RHCRC.cpp to prevent name collisions with other similarly named code in other libraries. Suggested by Steve Childress.

Fix SPI bus speed errors on 8MHz Arduinos.

1.22 2014-07-01

Update **RH_ASK** documentation for common wiring connections.

Testing **RH_ASK** with HopeRF RFM83C/RFM85 courtesy Anarduino <http://www.anarduino.com/>

Testing **RH_NRF24** with Itead Studio IBoard Pro <http://imall.iteadstudio.com/iboard-pro.html> using both hardware SPI on the ITDB02 Parallel LCD Module Interface pins and software SPI on the nRF24L01+ Module Interface pins. Documented wiring required.

Added support for AVR 1284 and 1284p, contributed by Peter Scargill. Added support for Semtech

SX1276/77/78 and HopeRF RFM95/96/97/98 and other similar LoRa capable radios in LoRa mode only. Tested with the excellent MiniWirelessLoRa from Anarduino <http://www.anarduino.com/miniwireless>

1.23 2014-07-03

Changed the default modulation for **RH_RF69** to GFSK_Rb250Fd250, since the previous default was not very reliable.

Documented **RH_RF95** range tests.

Improvements to **RH_RF22** RSSI readings so that lastRssi correctly returns the last message in dBm.

1.24 2014-07-18 Added support for building RadioHead for STM32F4 Discovery boards, using the native STM Firmware libraries, in order to support Codec2WalkieTalkie (<http://www.airspayce.com/mikem/Codec2WalkieTalkie>) and other projects. See STM32ArduinoCompat.

Default modulation for **RH_RF95** was incorrectly set to a very slow Bw125Cr48Sf4096

1.25 2014-07-25 The available() function will longer terminate any current transmission, and force receive mode. Now, if there is no unprocessed incoming message and an outgoing message is currently being transmitted, available() will return false.

RHRouter::sendtoWait(uint8_t*, uint8_t, uint8_t, uint8_t) renamed to sendtoFromSourceWait due to conflicts with new sendtoWait() with optional flags.

RHMesh and **RHRouter** already supported end-to-end application layer flags, but

RHMesh::sendtoWait() and **RHRouter::sendtoWait** have now been extended to expose a way to send optional application layer flags.

1.26 2014-08-12 Fixed a Teensy 2.0 compile problem due yield() not available on Teensy < 3.0.

Adjusted the algorithm of **RH_RF69::temperatureRead()** to more closely reflect reality.

Added functions to **RHGenericDriver** to get driver packet statistics: rxBad(), rxGood(), txGood().

Added **RH_RF69::printRegisters()**.

RH_RF95::printRegisters() was incorrectly printing the register index instead of the address.

Reported by Phang Moh Lim.

RH_RF95, added definitions for some more registers that are usable in LoRa mode.

RH_RF95::setTxPower now uses RH_RF95_PA_DAC_ENABLE to achieve 21, 22 and 23dBm.

RH_RF95, updated power output measurements.

Testing **RH_RF69** on Teensy 3.1 with RF69 on PJRC breakout board. OK.

Improvements so RadioHead will build under Arduino where SPI is not supported, such as ATtiny.

Improvements so RadioHead will build for ATtiny using Arduino IDE and tinycore arduino-tiny-0100-0018.zip.

Testing **RH_ASK** on ATtiny85. Reduced RAM footprint. Added helpful documentation. Caution: RAM memory is *very* tight on this platform.

RH_RF22 and **RH_RF69**, added setIdleMode() function to allow the idle mode radio operating state to be controlled for lower idle power consumption at the expense of slower transitions to TX and RX.

1.27 2014-08-13 All **RH_RF69** modulation schemes now have data whitening enabled by default.

Tested and added a number of OOK modulation schemes to **RH_RF69** Modem config table.

Minor improvements to a number of the faster **RH_RF69** modulation schemes, but some slower ones are still not working correctly.

1.28 2014-08-20 Added new **RH_RF24** driver to support Si446x, RF24/26/26, RFM24/26/27 family of transceivers. Tested with the excellent Anarduino Mini and RFM24W and RFM26W with the generous assistance of the good people at Anarduino <http://www.anarduino.com>.

1.29 2014-08-21 Fixed a compile error in **RH_RF24** introduced at the last minute in the previous release.

Improvements to **RH_RF69** modulation schemes: now include the AFCBW in the ModemConfig. ModemConfig **RH_RF69::FSK_Rb2Fd5** and **RH_RF69::GFSK_Rb2Fd5** are now working.

1.30 2014-08-25 Fixed some compile problems with ATtiny84 on Arduino 1.5.5 reported by Glen Cook.

1.31 2014-08-27 Changed **RH_RF69** FSK and GFSK modulations from Rb2_4Fd2_4 to Rb2_4Fd4_8 and FSK_Rb4_8Fd4_8 to FSK_Rb4_8Fd9_6 since the previous ones were unreliable (they had modulation indexes of 1).

1.32 2014-08-28 Testing with RedBearLab Blend board <http://redbearlab.com/blend/>. OK.

Changed more **RH_RF69** FSK and GFSK slowish modulations to have modulation index of 2 instead of 1. This required changing the symbolic names.

1.33 2014-09-01 Added support for sleep mode in RHGeneric driver, with new mode RHModeSleep and new virtual function sleep().

Added support for sleep to **RH_RF69**, **RH_RF22**, **RH_NRF24**, **RH_RF24**, **RH_RF95** drivers.

1.34 2014-09-19 Fixed compile errors in example rf22_router_test.

Fixed a problem with **RH_NRF24::setNetworkAddress**, also improvements to **RH_NRF24** register printing. Patched by Yveaux.

Improvements to **RH_NRF24** initialisation for version 2.0 silicon.

Fixed problem with ambiguous print call in RH_RFM69 when compiling for Codec2.

Fixed a problem with **RH_NRF24** on RFM73 where the LNA gain was not set properly, reducing the sensitivity of the receiver.

1.35 2014-09-19 Fixed a problem with interrupt setup on **RH_RF95** with Teensy3.1. Reported by AD.

1.36 2014-09-22 Improvements to interrupt pin assignments for **AVR_ATmega1284**

and **__AVR_ATmega1284P__**, provided by Peter Scargill.

Work around a bug in Arduino 1.0.6 where digitalPinToInterrupt is defined but NOT_AN_INTERRUPT is not.

1.37 2014-10-19 Updated doc for connecting **RH_NRF24** to Arduino Mega.

Changes to **RHGenericDriver::setHeaderFlags()**, so that the default for the clear argument is now RH_FLAGS_APPLICATION_SPECIFIC, which is less surprising to users. Testing with the excellent MoteinoMEGA from LowPowerLab <https://lowpowerlab.com/shop/moteinomega> with on-board RFM69W.

1.38 2014-12-29 Fixed compile warning on some platforms where **RH_RF24::send** and **RH_RF24::writeTxFifo** did not return a value.

Fixed some more compiler warnings in **RH_RF24** on some platforms.

Refactored printRegisters for some radios. Printing to Serial is now controlled by the definition of RH_HAVE_SERIAL.

Added partial support for ARM M4 w/CMSIS with STM's Hardware Abstraction lib for Steve Childress.

1.39 2014-12-30 Fix some compiler warnings under IAR.

RH_HAVE_SERIAL and Serial.print calls removed for ATtiny platforms.

1.40 2015-03-09 Added notice about availability on PlatformIO, thanks to Ivan Kravets.

Fixed a problem with **RH_NRF24** where short packet lengths would occasionally not be transmitted due to a race condition with RH_NRF24_TX_DS. Reported by Mark Fox.

1.41 2015-03-29 **RH_RF22**, **RH_RF24**, **RH_RF69** and **RH_RF95** improved to allow driver.init() to be

called multiple times without reallocating a new interrupt, allowing the driver to be reinitialised after sleeping or powering down.

1.42 2015-05-17 Added support for **RH_NRF24** driver on Raspberry Pi, using BCM2835 library for GPIO pin IO. Contributed by Mike Poulton.

Tested **RH_NRF24** module with NRF24L01+PA+LNA SMA Antenna Wireless Transceiver modules similar to: <http://www.electfreaks.com>

[/wiki/index.php?title=2.4G_Wireless_nRF24L01p_with_PA_and_LNA](http://wiki/index.php?title=2.4G_Wireless_nRF24L01p_with_PA_and_LNA) works with no software changes.

Measured max power output 18dBm.

1.43 2015-08-02 Added **RH_NRF51** driver to support Nordic nRF51 family processor with 2.4GHz radio such as nRF51822, to be built on Arduino 1.6.4 and later. Tested with RedBearLabs nRF51822 board and BLE Nano kit

1.44 2015-08-08 Fixed errors with compiling on some platforms without serial, such as ATTiny. Reported by Friedrich Müller.

1.45 2015-08-13 Added support for using **RH_Serial** on Linux and OSX (new class RHUtil/HardwareSerial encapsulates serial ports on those platforms). Example examples/serial upgraded to build and run on Linux and OSX using the tools/simBuild builder. **RHMesh**, **RHRouter** and **RHReliableDatagram** updated so they can use **RH_Serial** without polling loops on Linux and OSX for CPU efficiency.

1.46 2015-08-14 Amplified some doc concerning Linux and OSX **RH_Serial**. Added support for 230400 baud rate in HardwareSerial.

Added sample sketches nrf51_audio_tx and nrf51_audio_rx which show how to build an audio TX/RX pair with RedBear nRF51822 boards and a SparkFun MCP4725 DAC board. Uses the built-in ADC of the nRF51822 to sample audio at 5kHz and transmit packets to the receiver which plays them via the DAC.

1.47 2015-09-18 Removed top level Makefile from distribution: its only used by the developer and its presence confuses some people.

Fixed a problem with **RHReliableDatagram** with some versions of Raspberry Pi random() that causes problems: random(min, max) sometimes exceeds its max limit.

1.48 2015-09-30 Added support for Arduino Zero. Tested on Arduino Zero Pro.

1.49 2015-10-01 Fixed problems that prevented interrupts working correctly on Arduino Zero and Due. Builds and runs with 1.6.5 (with 'Arduino SAMD Boards' for Zero version 1.6.1) from arduino.cc. Arduino version 1.7.7 from arduino.org is not currently supported.

1.50 2015-10-25 Verified correct building and operation with Arduino 1.7.7 from arduino.org. Caution: You must burn the bootloader from 1.7.7 to the Arduino Zero before it will work with Arduino 1.7.7 from arduino.org. Conversely, you must burn the bootloader from 1.6.5 to the Arduino Zero before it will work with Arduino 1.6.5 from arduino.cc. Sigh. Fixed a problem with **RH_NRF905** that prevented the power and frequency ranges being set properly. Reported by Alan Webber.

1.51 2015-12-11 Changes to RH_RF6::setTxPower() to be compatible with SX1276/77/78/79 modules that use RFO transmitter pins instead of PA_BOOST, such as the excellent Modtronix inAir4 <http://modtronix.com/inair4.html> and inAir9 modules <http://modtronix.com/inair9.html>. With the kind assistance of David from Modtronix.

1.52 2015-12-17 Added **RH_MRF89** module to support Microchip MRF89XA and compatible transceivers. and modules.

- 1.53 2016-01-02 Added **RH_CC110** module to support Texas Instruments CC110L and compatible transceivers and modules.
- 1.54 2016-01-29 Added support for ESP8266 processor on Arduino IDE. Examples `serial_reliable_datagram_*` are shown to work. CAUTION: SPI not supported yet. Timers used by **RH_ASK** are not tested. The GHz radio included in the ESP8266 is not yet supported.
- 1.55 2016-02-12 Added macros for `htons()` and friends to **RadioHead.h**. Added example sketch `serial_gateway.pde`. Acts as a transparent gateway between **RH_RF22** and **RH_Serial**, and with minor mods acts as a universal gateway between any 2 RadioHead driver networks. Initial work on supporting STM32 F2 on Particle Photon: new platform type defined. Fixed many warnings exposed by test building for Photon. Particle Photon tested support for **RH_Serial**, **RH_ASK**, SPI, **RH_CC110** etc. Added notes on how to build RadioHead sketches for Photon.
- 1.56 2016-02-18 Implemented timers for **RH_ASK** on ESP8266, added some doc on IO pin selection.
- 1.57 2016-02-23 Fixed an issue reported by S3B, where **RH_RF22** would sometimes not clear the `rxbufvalid` flag.
- 1.58 2016-04-04 Tested **RH_RF69** with Arduino Due. OK. Updated doc. Added support for all ChipKIT Core supported boards http://chipkit.net/wiki/index.php?title=ChipKIT_core Tested on ChipKIT Uno32. Digilent Uno32 under the old MPIDE is no longer formally supported but may continue to work for some time.
- 1.59 2016-04-12 Testing with the excellent Rocket Scream Mini Ultra Pro with the RFM95W and RFM69HCW modules from <http://www.rocketstream.com/blog/product/mini-ultra-pro-with-radio/> (915MHz versions). Updated documentation with hints to suit. Caution: requires Arduino 1.6.8 and Arduino SAMD Boards 1.6.5. See also <http://www.rocketstream.com/blog/2016/03/10/radio-range-test-with-rfm69hcw/> for the vendors tests and range with the RFM69HCW version. They also have an RF95 version equipped with TCXO temperature controlled oscillator for extra frequency stability and support of very slow and long range protocols. These boards are highly recommended. They also include battery charging support.
- 1.60 2016-06-25 Tested with the excellent talk2 Whisper Node boards (<https://talk2.wisen.com.au/> and <https://bitbucket.org/talk2/>), an Arduino Nano compatible board, which include an on-board RF69 radio, external antenna, run on 2xAA batteries and support low power operations. RF69 examples work without modification. Added support for ESP8266 SPI, provided by David Skinner.
- 1.61 2016-07-07 Patch to `RH_ASK.cpp` for ESP8266, to prevent crashes in interrupt handlers. Patch from Alexander Mamchits.
- 1.62 2016-08-17 Fixed a problem in **RH_ASK** where `_rxInverted` was not properly initialised. Reported by "gno.sun.sop". Added support for `waitCAD()` and `isChannelActive()` and `setCADTimeout()` to `RHGeneric`. Implementation of **RH_RF95::isChannelActive()** allows the RF95 module to support Channel Activity Detection (CAD). Based on code contributed by Bent Guldbjerg Christensen. Implementations of `isChannelActive()` plus documentation for other radio modules will be welcomed.
- 1.63 2016-10-20 Testing with Adafruit Feather 32u4 with RFM69HCW. Updated documentation to reflect.
- 1.64 2016-12-10 **RHReliableDatagram** now initialises `_seenids`. Fix from Ben Lim. In **RH_NRF51**, added `get_temperature()`. In **RH_NRF51**, added support for AES packet encryption, which required a slight change to the on-air

message format.

1.65 2017-01-11 Fixed a race condition with **RH_NRF51** that prevented ACKs being reliably received. Removed code in **RH_NRF51** that enabled the DC-DC converter. This seems not to be a necessary condition for the radio to work and is now left to the application if that is required.

Proven interoperation between nRF51822 and nRF52832.

Modification and testing of **RH_NRF51** so it works with nRF52 family processors, such as Sparkfun nRF52832 breakout board, with Arduino 1.6.13 and Sparkfun nRF52 boards manager 0.2.3 using the procedures outlined in <https://learn.sparkfun.com/tutorials/nrf52832-breakout-board-hookup-guide>

Caution, the Sparkfun development system for Arduino is still immature. We had to rebuild the nrfutil program since the supplied one was not suitable for the Linux host we were developing on. See <https://forum.sparkfun.com/viewtopic.php?f=32&t=45071> Also, after downloading a sketch in the nRF52832, the program does not start executing cleanly: you have to reset the processor again by pressing the reset button. This appears to be a problem with nrfutil, rather than a bug in RadioHead.

1.66 2017-01-15 Fixed some errors in (unused) register definitions in **RH_RF95.h**.

Fixed a problem that caused compilation errors in **RH_NRF51** if the appropriate board support was not installed.

1.67 2017-01-24 Added **RH_RF95::frequencyError()** to return the estimated centre frequency offset in Hz of the last received message

1.68 2017-01-25 Fixed arithmetic error in **RH_RF95::frequencyError()** for some platforms.

1.69 2017-02-02 Added **RH_RF95::lastSNR()** and improved lastRssi() calculations per the manual.

1.70 2017-02-03 Added link to Binpress commercial license purchasing.

1.71 2017-02-07 Improved support for STM32. Patch from Bent Guldbjerg Christensen.

1.72 2017-03-02 In **RH_RF24**, fixed a problem where some important properties were not set by the ModemConfig. Added properties 2007, 2008, 2009. Also property 200a was not being set in the chip. Reported by Shannon Bailey and Alan Adamson. Fixed corresponding convert.pl and added it to the distribution.

1.73 2017-03-04 Significant changes to **RH_RF24** and its API. It is no longer possible to change the modulation scheme programatically: it proved impossible to cater for all the possible crystal frequencies, base frequency and modulation schemes. Instead you can use one of a small set of supplied radio configuration header files, or generate your own with Silicon Labs WDS application. Changing modulation scheme required editing RH_RF24.cpp to specify the appropriate header and recompiling. convert.pl is now redundant and removed from the distribution.

1.74 2017-03-08 Changed **RHReliableDatagram** so it would not ACK messages heard addressed to other nodes in promiscuous mode.

Added **RH_RF24::deviceType()** to return the integer value of the connected device.

Added documentation about how to connect RFM69 to an ESP8266. Tested OK.

RH_RF24 was not correctly changing state in sleep() and setModIdle().

Added example rf24_lowpower_client.pde showing how to put an arduino and radio into a low power mode between transmissions to save battery power.

Improvements to **RH_RF69::setTxPower** so it now takes an optional ishighpowermodule flag to indicate if the connected module is a high power RFM69HW, and so set the power level correctly. Based on code contributed by bob.

1.75 2017-06-22 Fixed broken compiler issues with **RH_RF95::frequencyError()** reported by Steve

Rogerson.

Testing with the very excellent Rocket Scream boards equipped with RF95 TCXO modules. The temperature controlled oscillator stabilises the chip enough to be able to use even the slowest protocol Bw125Cr48Sf4096. Caution, the TCXO model radios are not low power when in sleep (consuming about ~600 uA, reported by Phang Moh Lim).

Added support for EBYTE E32-TTL-1W and family serial radio transceivers. These RF95 LoRa based radios can deliver reliable messages at up to 7km measured.

1.76 2017-06-23 Fixed a problem with **RH_RF95** hanging on transmit under some mysterious circumstances. Reported by several people at <https://forum.pjrc.com/threads/41878-Probable-race-condition-in-Radiohead-library?p=146601#post146601>

Increased the size of rssi variables to 16 bits to permit RSSI less than -128 as reported by RF95.

1.77 2017-06-25 Fixed a compilation error with lastRssi().

1.78 2017-07-19 Fixed a number of unused variable warnings from g++.

Added new module **RHEncryptedDriver** and examples, contributed by Philippe Rochat, which adds encryption and decryption to any RadioHead transport driver, using any encryption cipher supported by ArduinoLibs Cryptographic Library <http://rweather.github.io/arduino-lib-crypto.html> Includes several examples.

1.79 2017-07-25 Added documentation about 'Passing Sensor Data Between RadioHead nodes'.

Changes to **RH_CC110** driver to calculate RSSI in dBm, based on a patch from Jurie Pieterse.

Added missing passthrough methods to **RHEncryptedDriver**, allowing it to be used with **RHDatagram**, **RHReliableDatagram** etc. Tested with **RH_Serial**. Added examples

1.80 2017-10-04 Testing with the very fine Talk2 Whisper Node LoRa boards <https://wisen.com.au/store/products/whisper-node-lora> an Arduino compatible board, which include an on-board RFM95/96 LoRa Radio (Semtech SX1276), external antenna, run on 2xAAA batteries and support low power operations. RF95 examples work without modification. Use Arduino Board Manager to install the Talk2 code support. Upload the code with an FTDI adapter set to 5V.

Added support for SPI transactions in development environments that support it with SPI_HAS_TRANSACTION. Tested on ESP32 with RFM-22 and Teensy 3.1 with RF69 Added support for ESP32, tested with RFM-22 connected by SPI.

1.81 2017-11-15 **RH_CC110**, moved setPaTable() from protected to public.

RH_RF95 modem config Bw125Cr48Sf4096 altered to enable slow data rate in register 26 as suggested by Dieter Kneffel. Added support for nRF52 compatible Arm chips such as Adafruit BLE Feather board <https://www.adafruit.com/product/3406>, with a patch from Mike Bell.

Fixed a problem where rev 1.80 broke Adafruit M0 LoRa support by declaring bitOrder variable always as a unsigned char. Reported by Guilherme Jardim.

In **RH_RF95**, all modes now have AGC enabled, as suggested by Dieter Kneffel.

1.82 2018-01-07 Added guard code to **RH_NRF24::waitPacketSent()** so that if the transmit never completes for some reason, the code will eventually return with FALSE. Added the low-datarate-optimization bit to config for **RH_RF95::Bw125Cr48Sf4096**. Fix from Jurie Pieterse to ensure **RH_CC110::sleep** always enters sleep mode. Update ESP32 support to include ASK timers. **RH_ASK** module is now working on ESP32.

1.83 2018-02-12 Testing adafruit M0 Feather with E32. Updated **RH_E32** documentation to show suggested connections and constructor initialisation.

Fixed a problem with **RHEncryptedDriver** that could cause a crash on some platforms when used with

RHReliableDatagram. Reported by Joachim Baumann.

Improvements to doxygen doc layout in **RadioHead.h**

1.84 2018-05-07 Compiles with Roger Clarke's Arduino_STM32 https://github.com/rogerclarkmelbourne/Arduino_STM32, to support STM32F103C etc, and STM32 F4 Discovery etc. Tested STM32 F4 Discovery board with **RH_RF22**, **RH_ASK** and **RH_Serial**.

1.85 2018-07-09 **RHGenericDriver** methods changed to virtual, to allow overriding by RHEncryptedDriver: lastRssi(), mode(), setMode(). Reported by Eyal Gal.
Fixed a problem with compiling **RH_E32** on some older IDEs, contributed by Philippe Rochat.
Improvements to **RH_RF95** to improve detection of bad packets, contributed by PiNi.
Fixed an error in **RHEncryptedDriver** that caused incorrect message lengths for messages multiples of 16 bytes when STRICT_CONTENT_LEN is defined.
Fixed a bug in **RHMesh** which causes the creation of a route to the address which is the byte behind the end of the route array. Reported by Pascal Gillès de Pélichy.

1.86 2018-08-28 Update commercial licensing, remove binpress.

1.87 2018-10-06 **RH_RF22** now resets all registers to default state before initialisation commences. Suggested by Wothke.
Added RH_ENABLE_EXPLICIT_RETRY_DEDUP which improves the handling of duplicate detection especially in the case where a transmitter periodically wakes up and start transmitting from the first sequence number. Patch courtesy Justin Newitter. Thanks.

1.88 2018-11-13 Updated to support ATTiny using instructions in <https://medium.com/jungletronics/attiny85-easy-flashing-through-arduino-b5f896c48189> Updated examples ask_transmitter and ask_receiver to compile cleanly on ATTiny. Tested using ATTiny85 and Arduino 1.8.1.

1.89 2018-11-15 Testing with ATTiny core from <https://github.com/SpenceKonde/ATTinyCore> and **RH_ASK**, using example ask_transmitter. This resulted in 'Low Memory, instability may occur', and the resulting sketch would transmit only one packet. Suggest ATTiny users do not use this core, but use the one from https://raw.githubusercontent.com/damellis/attiny/ide-1.6.x-boards-manager/package_damellis_attiny_index.json as described in <https://medium.com/jungletronics/attiny85-easy-flashing-through-arduino-b5f896c48189>

Added support for **RH_RF95::setSpreadingFactor()**, **RH_RF95::setSignalBandwidth()**, **RH_RF95::setLowDataRate()** and **RH_RF95::setPayloadCRC()**. Patch from Brian Norman. Thanks.

1.90 2019-05-21 Fixed a block size error in RhEncryptedDriver for the case when using STRICT_CONTENT_LEN and sending messages of exactly _blockcipher.blockSize() bytes in length. Reported and patched by Philippe Rochat. Patch from Samuel Archibald to prevent compile errors with RH_AAK.cpp for ATSAM51. Fixed a problem in **RH_RF69::setSyncWords** that prevented setSyncWords(NULL, 0) correctly disabling sync detection and generation. Reported by Federico Maggi. RHHardwareSPI::usingInterrupt() was a noop. Fixed to call SPI.usingInterrupt(interrupt);.

1.91 2019-06-01 Fixed a problem with new RHHardwareSPI::usingInterrupt() that prevented compilation on ESP8266 which does not have that call.

1.92 2019-07-14 Retested serial_reliable_datagram_client.pde and serial_reliable_datagram_server.pde built on Linux as described in their headers, and with USB-RS485 adapters. No changes, working correctly. Testing of nRF5232 with Sparkfun nRF52 board support 0.2.3 shows that there appears to be a problem with interrupt handlers on this board, and none of the interrupt based radio drivers can be expected to work with this chip. Ensured all interrupt routines are flagged with ICACHE_RAM_ATTR when compiled for ESP8266, to prevent crashes.

- 1.94 2019-09-02 Fixed a bug in **RHSoftwareSPI** where **RHGenericSPI::setBitOrder()** has no effect for on **RHSoftwareSPI**. Reported by Peter.
Added support in **RHRouter** for a node to optionally be leaf node, and not participate as a router in the network. See **RHRouter::setNodeTypePatch** from Alex Evans.
Fixed a problem with ESP32 causing compile errors over missing **SPI.usingInterrupt()**.
- 1.95 2019-10-14 Fixed some typos in **RH_RF05.h** macro definitions reported by Clayton Smith.
Patch from Michael Cain from **RH_ASK** on ESP32, untested by me.
Added support for RPi Zero and Zero W for the RF95, contributed by Brody Mahoney. Not tested by me.
- 1.96 2019-10-14 Added examples for RPi Zero and Zero W to **examples/raspi/rf95**, contributed by Brody Mahoney not tested by me.
- 1.97 2019-11-02 Added support for Mongoose OS, contributed by Paul Austen.
- 1.98 2020-01-06 Rationalised use of **RH_PLATFORM_ATTINY** to be consistent with other platforms.
Added support for **RH_PLATFORM_ATTINY_MEGA**, for use with Spencer Konde's **megaTinyCore** <https://github.com/SpenceKonde/megaTinyCore> on Atmel megaAVR ATtiny 1-series chips. Tested with ATtiny 3217, 3216 and 1614, using **RH_Serial**, **RH_ASK**, and **RH_RF22** drivers.
- 1.99 2020-03-07 Release under GPL V3
- 1.100 2020-03-12 Fixed a problem that prevented compilation of **RH_NRF51** on Arduino for Sparkfun nRF52832 Breakout board.
- 1.101 2020-04-10 Tested nRF52832 with RFM69W module and **RH_RF69**, using Software SPI and hardware interrupts OK.
Fixed warnings about 'deleting object of polymorphic class' if driver is dynamically allocated.
Fixed problems in **RH_ASK** and **HardwareSPI** to work with STM32F4 Discovery with latest version of **stm32duino** https://github.com/stm32duino/Arduino_Core_STM32. Testing with **stm32duino** 1.8.0 downloaded with Board Manager per https://github.com/stm32duino/Arduino_Core_STM32. Now builds and run **RH_ASK** examples with STM32F4 Discovery board. Build without error for STM32 F1 and F4 but Does not compile for Generic STM32F3.

Author

Mike McCauley. DO NOT CONTACT THE AUTHOR DIRECTLY. USE THE GOOGLE LIST GIVEN ABOVE