Prepared	No	Rev
Magnus Weiner	13-001-01-1	1.0
Approved	Date	Project
Bengt Erlandsson	2015-03-25	Gen 3 open radio protocol

Gen 3 open radio communication protocol

Abstract
This document is a specification for the open communication protocol to be used within Tunstall third generation radio communication system.

Application
Tunstall radio products

History

Rev	Change	By
1.0	First release, 2015-03-25	BE

Prepared Magnus Weiner	No 13-001-01-1	Rev 1.0
Approved Bengt Erlandsson	Date 2015-03-25	Project Gen 3 open radio protocol

Table of Contents

Prepared	No	Rev
Magnus Weiner	13-001-01-1	1.0
Approved	Date	Project
Bengt Erlandsson	2015-03-25	Gen 3 open radio protocol

1. General

1.1. Background

This document is a specification for Tunstall Healthcare's third generation open radio communication protocol.

1.2. License agreement

This protocol is released under GNU General Public License version 3.

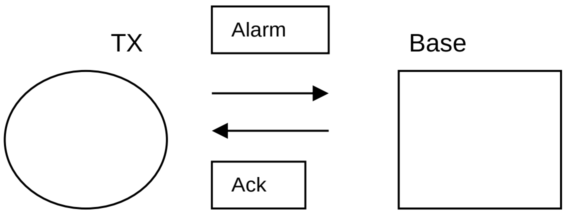
1.3. Recognition

Tunstall would like to thank Richard M Stallman (RMS) and the Free Software Foundation (FSF) for their contribution in making the world a more interoperable place.

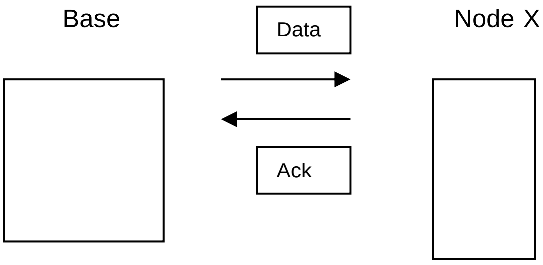
2. Typical protocol applications

The information below shows some typical communication examples the protocol is able to handle.

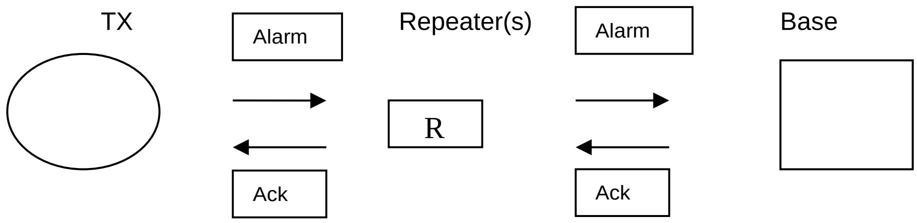
2.1. Transmitter sends an alarm to base station.



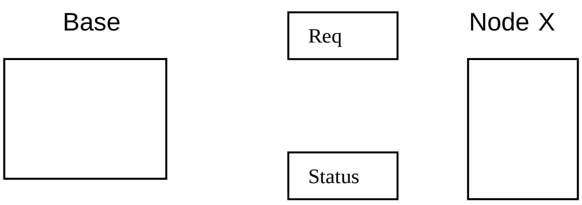
2.2. Base station updates parameter in a node.



2.3. Transmitter sends an alarm to base station. Communication relayed via repeater(s)



2.4. Base station requests status from a node.



3. General requirements

3.1. Reliability

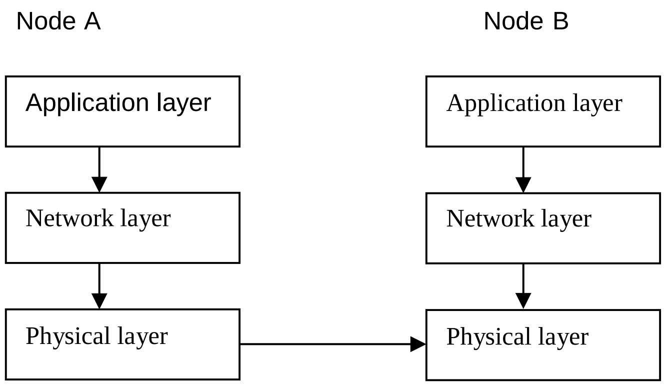
The protocol is designed to be able to handle situations with difficult transmission conditions. This is solved including error correction and detection algorithms in the protocol.
The protocol also includes packet acknowledgement and packet sequencing to ensure data is valid and properly delivered.

3.2. System resource usage

The protocol is designed to be easy to implement in low end microcontrollers. I.e. the program and data memory requirements must be kept low.
The protocol also supports low power modes to reduce current consumption.

4. Layer definition

For simplification of understanding, the communication structure is divided into three different layers. These layers are the physical layer, the network layer and the application layer. Each layer adds its own header information to the payload. The protocol supports transportation over both radio links and RS232 wired links, in which cases the physical layers will differ.



4.1. Application layer header

Basically, the application layer will consist of a command byte as header and an optional data field.

Application header (1 command byte)	Data field (0-16 bytes)

4.2. Network layer header

The network layer will typically add its header to the application layer payload as shown below.

Network header (14 bytes)	Application layer payload (1-17 bytes)

The network layer header consists of the following fields:

Hinfo (1) **|** **Gflags** (1) **|** **Tflags** (1) **|** **Rept**(1) **|** **Rseq/Fseq** (1) **|** **Skey** (1) **|** **Source addr** (4) **|** **Dest addr** (4)

4.2.1. Hinfo

Hinfo carries information and flags regarding the network layer header as described below.

Bit	Meaning
7	Reserved
6	Reserved
5	Reserved
4-0	Ver

Bits 5-7 are reserved network header flags.

4.2.2. Ver

Ver is a 5-bit field which indicates the version of the network layer header. This enables for future non-compatible network layer protocol versions to be implemented in the system.
Currently supported version is #1.

4.2.3. Gflags

Gflags, general flags, is a 8-bit flag field containing various frame control flags, as described below.

Bit	Meaning
7	Reply frame
6-0	Reserved for future use (set to 0)

Reply frame, Bit 7

This flag is set if the frame is a reply from a slave. If the frame is originated by a master, the Reply bit is cleared.

4.2.4. Tflags

Tflags, Transport Flags, is a 8-bit flag field containing various frame transport control flags, as described below.

Bit	Meaning
7	Reserved for future use (set to 0)
6	Frequency (primary or secondary
5-2	Reserved for future use (set to 0)
1-0	Forward Error Correction mode

Frequency, bit 6

This bit indicates if the frame is sent via the primary frequency or the secondary frequency. The flag is cleared for primary and set for secondary.

Forward error correction mode, bit 1-0

This is a 2 bit field indicating what kind of forward error correction should be used when the frame is transmitted over the air.

Value	Associated forward error correction/detection method
0	None
1	12,8 Hamming encoding
2	Convolutional Viterbi encoding
3	Reserved

4.2.5. Rept

Rept carries various information for control of frame repetition within repeaters, as described below.

Bit	Meaning
7	Frame repeated
6	Record route
5	Buffer full
4-0	TTL

Frame repeated, Bit 7

This flag is set by any routers or repeaters repeating the frame. This flag is used if the receiver needs to adjust reply timing depending on whether the frame goes directly or via repeaters.

Record route, Bit 6

When the record route bit is set, any repeaters or routers forwarding the frame are instructed to add their local ID byte at the end of the frame. If there is not enough space left to do this, the router or repeater will ignore this bit and set the Buffer Full bit.
Any repeaters adding their local ID to the frame will use the last byte of the frame as a counter for number of ID:s added. I.e. the first repeater receiving the frame from the originator will typically see that the Frame Repeated bit is cleared and add both its ID and a counter with a value of 1 to the buffer. The repeater then sets the Frame Repeated bit. The second repeater catching the frame will then just add it's ID and increase the counter.

Buffer full, Bit 5

This bit is set if a repeater finds the buffer space to be insufficient when it is in the process of adding it's ID. The receiving unit will then know that the recorded route is inconsistent.

TTL, bit 4-0

TTL is a 5-bit frame life time counter, Time To Live.
Every time a router or repeater receives a valid frame, the TTL field is decreased by one before the frame is forwarded. If the TTL field reaches zero, the frame is discarded.
This mechanism is used to avoid situations where frames can get “ping-ponged” between repeaters infinitely.
This field can also be used for skipping the forwarding mechanism totally. I.e. if there is a need to reduce the radio traffic when short distance is desired (E.g. future radio based Report unit), the TTL field can be set to 0 by the originator.

4.2.6. Rseq/Fseq

The Rseq/Fseq field contains two 4 bit sequence number counters, Radio sequence number and Frame sequence number.
These sequence numbers span from 0 to 0fh and are used for ensuring data is valid and properly delivered.

Fseq is used to uniquely tag every outgoing data session from the transmitter. I.e. retransmitted frames carry the same Fseq but different Rseq tags. Fseq is used for ensuring correct packet acknowledge sequencing on the application level in the communication chain.

Rseq and Fseq are described more in detail in the sections describing packet forwarding and packet acknowledgement.

4.2.7. Skey

The SKEY field is reserved for future use.

4.2.8. Source addr

Source addr is a 4 byte field carrying the address of the originating device. A separate section describes the addressing scheme in detail.

4.2.9. Dest addr

Dest addr is a 4 byte field carrying the address the frame is destined for. A separate section describes the addressing scheme in detail.

4.3. Physical layer

The physical layer will add its header to the network layer payload as shown below. The physical layer header will be different depending on whether the frame is sent on a radio link or via wire.

4.3.1. Radio link transmission

In case of radio link transmission, the physical layer will add the following:

Physical layer header (24+n bits)	Network layer payload (15-31 bytes)	**CRC (2 bytes)**

The radio link physical layer header consists of the following fields:

Preamble (n bits) | Frame sync word (16 bits)| Frame length (8 bits)

Preamble

The preamble consists of a block of 10101010-transitions. This is mainly used for alignment of the quadrature detector on the receiver side, but also as a wake up sequence for sleeping receivers. The length of the preamble would typically be a minimum of 40 transitions. However, the length of the preamble will be significantly longer when the system is in wake up mode, i.e. a transmitter sends a long preamble in order to wake any power conserving sleeping receivers. The preamble in this case would typically last for around 1 second.

Frame sync word

The frame sync word is used to indicate for the receiver that the preamble ends and the transmission of the actual frame data starts. The frame sync word will also indicate what kind of forward error correction code (FEC) the data has been processed with.
The following frame sync words are used:

Frame sync word	Associated forward error correction/detection method
D391 (hex)	None
D392 (hex)	12,8 Hamming encoding
D393 (hex)	Convolutional Viterbi encoding
D394 (hex)	Reserved

None

No forward error correction is used in this case. Any bit errors will however be detected by means of the incorporated 16 bit CRC method.

Hamming encoding

Hamming encoding is a method to add a piece of redundancy information for implementing single bit error correction. A parity nibble is calculated and added to every outgoing 8 bit character, i.e. data is expanded from 8 to 12 bits. The parity nibble will then allow for correction of a single bit failure in a transmitted byte. If two bits are garbled, the system will also detect that and discard the data.
The system will not be able to detect more than 2 bits of failure in a byte. However the added 16 bit CRC will deal with that.

Convolutional Viterbi encoding

Convolutional Viterbi encoding is a method for adding parity information to a bitstream. The method is specifically used for correcting bit failures introduced by burst interference. This method is not implemented yet and is reserved for future use.

4.3.2. Wired link transmission

In case of wired link transmission, the physical layer will add the following header:

BOF	Network layer payload (15-31 bytes)	**CSUM**	**EOF**

BOF

BOF, hex C0, is used as a marker for the beginning of the frame. The presence of BOF is used on the receiver side to trigger the frame reception process.

CSUM

CSUM is an 8-bit field containing the checksum for the frame.
The checksum is calculated according to the following algorithm:
Every time a frame is to be sent, the checksum is reset to 0. Every byte in the message to be sent is then xor:ed with the checksum byte. After each xor operation the checksum byte is incremented by 1. As the checksum is an 8-bit field, the checksum is calculated modulo 256.

EOF

EOF, hex C1, is used as a marker for the end of the frame. The presence of EOF is used on the receiver side to switch the system to a frame reception complete state.

Character stuffing

Since the length of a frame is not fixed, the system must be able to distinguish data and control characters containing hex C0 and C1 from the encapsulation control characters BOF and EOF.
A system called Character Stuffing is used to achieve this. The system examines each byte to send in the data and control fields. If the byte is a BOF (hex C0), EOF (hex C1) or CE (Control Escape, hex 7D), bit 5 of the byte is inverted and a CE character is sent preceeding the byte.
The receiving side knows that when a CE character is received, this byte shall be discarded and bit 5 of the next received byte should be inverted. Thus the original byte is retained.

Note that the CE, BOF and EOF characters does not occupy any buffer space. They only exist on the wired link and are generated/discarded by the transmit/receive routines.

5. Addressing scheme

5.1. General

In principle, each node in the system has a unique address. The address field is a 4-byte field. This is used for unicast messages where the address is known. The most significant byte is used for unit identification. The system also uses some specially reserved addresses, such as the broadcast address for broadcast purposes.

5.2. Reserved addresses

Broadcast address	255.255.255.255
Local unit address	255.255.255.254
Null address	0.0.0.0
Base station	10.x.x.x
Transmitter	50.x.x.x
Transmitter	51.x.x.x
Wristband transmitter	55.x.x.x
Repeater	100.x.x.x
OEM equipment	140.x.x.x – 149.x.x.x
Reserved for test equipment	254.x.x.x, x.x.254.x
Reserved for future use	Other

6. Communication session

1.1 General

The communication principle can be looked upon as a variable master-slave system.
The master is defined as the node which initiates the transmission session. Any node can act as a master. When the system is in idle condition, all nodes are standing by in slave mode. If a node is configured as an active receiver, the node has the radio switched to RX mode to listen for incoming frames.

6.1. Node modes

A node can typically be in one of the following modes:

6.1.1. SLEEPING_NODE

Node in power down mode, does not listen on the radio. Typically an alarm transmitter that needs to conserve power.

6.1.2. IDLE_SLAVE

Active slave, ready for communication. Typically a base station, ready to receive data from an alarm transmitter.

6.1.3. BUSY_SLAVE

Active slave, occupied by a master. For example a base station receiving alarm info from an alarm transmitter.

6.1.4. MASTER

Active master, communicating with a slave. For example an alarm transmitter sending an alarm to a base station.

6.1.5. Communication process

When a node wants to transmit a message to another node, it enters the master mode and initiates a transmission to the receiving slave node.
A slave node can only handle communication with one master at a time, and is considered busy for other nodes than the current master node. (Exceptions to this can exist, for example a base station which can handle more than one master for receiving multiple alarms simultaneously)
A slave is considered available for other nodes a certain time period (typically 5 seconds) after the last communication session with the current master or after the master has issued a LEAVE command.

During startup, each node switches to slave mode and sets its current master address field to NULL. When a master contacts the slave (either by direct addressing or by broadcast address where the slave then discovers it has been programmed to respond to that particular master), the slave saves the address of the master in its current master address field and is considered busy occupied by that master.

6.2. Frame sequencing, Fseq

6.2.1. Initialization of Fseq

Each master and each slave has a frame sequencing number, Fseq. The purpose of Fseq is to ensure data is delivered correctly and to ensure commands are not executed twice if the corresponding slave answer gets lost and the master retransmits a frame.

The Fseq number is initialized to 0 during startup of a node. When an idle slave is called by the master, that slave will change to Busy Slave mode and accept any value of Fseq that the master sends since this is the first transmission in the data session.

6.2.2. Slave mode Fseq handling

Each time a slave receives a frame with a valid checksum and a Fseq which is same as the local Fseq, the local Fseq is increased by 1 (modulo 16)
After that, the slave performs the action that the frame command specifies, and responds with a frame corresponding to the actual frame command. The Fseq number in the responding frame is set to the local Fseq number, i.e. the frame sequence number of the frame the slave expects next.

If the slave receives a frame with an Fseq where Fseq+1 (modulo 16) = local Fseq and the frame command is the same as the previously received frame command (and the checksum also is correct), this is an indication that the master did not receive the previously sent reply from the slave. The slave then just resends the previous frame. (but with an updated Rseq number, see below)

If a slave receives a frame with an incorrect sequence number, NAK is sent. As in the case with a command reply, the NAK frame also carries the local Fseq number, i.e. the number of the frame the slave expects next.

6.2.3. Master mode Fseq handling

Every time the master receives a frame with correct checksum from the slave it is currently communicating with, the local Fseq number in the master is set to the received Fseq number. (even when NAK is received)
If the received frame is NAK, the previous frame is retransmitted. Otherwise next frame in the ongoing data session is prepared and sent, if this should be done.

6.3. Radio sequencing, Rseq

Each frame contains a radio sequencing number, Rseq.
Rseq is used to uniquely tag every outgoing frame from a transmitter.
Typically, Rseq will be different for each frame leaving a master or slave within a limited time frame. (Rseq is incremented modulo 16)

Rseq is necessary since radio repeaters or routers are included in the system. Rseq is used by a node to distinguish repeated frames from new valid frames. These “ghost frames” typically arrive at the node shortly after the valid frame has been received. The master or slave node will then store the source address and Fseq+Rseq information of the last valid frame received. If another frame with exactly the same source and frame information is received, that frame is discarded.

6.4. Master unit retransmission procedure

The master contains a timer which is set each time a frame is transmitted to a slave. If the master does not receive an answer from the slave before the timer times out, the frame is retransmitted. This is repeated a maximum number of times (typically 10) before the master gives up and reports an error to the application layer.
Typical timeout period for the retransmission timer is 2 seconds, depending on what operation the slave is instructed to perform.

Depending on the application, special action might be taken after a couple of failing retransmissions.
In case of an alarm transmitter failing to get in contact with an alarm receiver, the alarm transmitter might change operation mode after a number of failing transmission.
The alarm transmitter can for example decide to transmit frames with longer preamble sessions to be able to wake up any sleeping receivers in case of power failure.
The alarm transmitter might also decide to change frequency band after a couple of failing transmissions in its efforts to get through to the receiving side.

7. Application layer commands

Command from Master	**Purpose**
System commands	
0, None	Null command. Reserved and not used.
1, Ping	Frame carries a ping request
11, Leave	Frame carries a Leave request
Alarm parameters	
52, Alarm message	Frame carries alarm message information in the data field.
200, Test frame 1	Frame type used by test equipment.
Responses from slave	**Purpose**
242, Not implemented	Frame indicates requested command is not implemented in actual node
243, Gen Data	General data frame
244, Gen Error	General error report frame
253, NAK	Negative acknowledge
254, ACK	Acknowledge of received frame

Other	Reserved for future use

7.1. 1, Ping

7.1.1. Data field structure:

No data field for this frame.

7.1.2. Expected reply:

254. Ack

This reply is sent by the targeted device.

7.2. 11, Leave

This command is used for logging off from a slave, leaving it ready to accept data sessions from other nodes.

7.2.1. Data field structure:

No data field for this frame.

7.2.2. Expected reply:

254. Ack

This reply is sent if the operation was completed successfully

7.3. 52, Alarm message

This alarm frame is sent from alarm devices.

7.3.1. Data field structure:

Data[0] Alarm reason
- Bit [1..0] = 00 -> Alarm button not pressed.
- Bit [1..0] = 01 -> Alarm button pressed once
- Bit [1..0] = 10 -> Alarm button pressed twice
- Bit [1..0] = 11 -> Alarm button pressed three times
- Bit 2 = External alarm input 1 activated
- Bit 3 = External alarm input 2 activated
- Bit 4 = Low battery alarm
- Bit 5 = Low battery state (1 = low battery)

Bit 6 = Periodic test alarm
Bit 7 = Startup alarm

Data[1] Alarm type
See table in the misc section.

Data[2..15] Reserved for future use.

7.3.2. Expected reply:

243, Gen Data:

On line configuration data according to:

Data [0] Operating parameters1
Bit 7 = Active / Storage mode (1 = storage)
Bit 6 = Automatic test alarm (1 = active)
Bit 5 = Test mode (1 = active)
Bit 4 = Ignore/Delay mode (1 = active)
Bit 3 = Reserved for future use (set to 0)
Bit 2 = = Reserved for future use (set to 0)

Bit 1 = 1 -> = Reserved for future use (set to 0)
Bit 0 = 1 -> = Reserved for future use (set to 0)

Data [1] Reserved for future use. (set to 0)

7.4. General reply frames

These reply frames can be sent generally as an answer to a frame.

242, Not implemented

This reply is sent if the requested function is not implemented in the remote node.

253, NAK

This reply is sent if the Fseq sequencing number is totally out of sequence and needs to be synchronized.

8. Miscellaneous data tables

8.1. Registered alarm types

Number	Type
1	Medical alarm
10	Assault/Emergency

20	Fire
30	Passive
40	Door
41	Door bell
50	Misc
51	Medical dispenser alarm