SSPN Simplex Black Channel with Repeaters

SSPN Simplex Black Channel with Repeaters

History

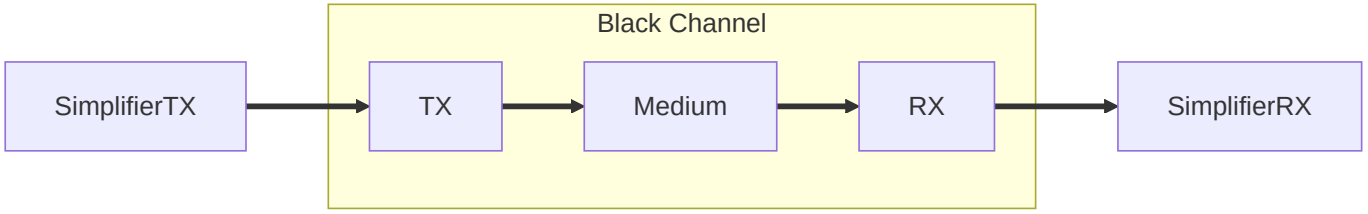
date	version	name	description
2025-01-12	v0.1	Jesper Ribbe	Initial draft
2025-01-21	v0.2	Jesper Ribbe	slight cleanup
2025-04-05	v1.0	Jesper Ribbe	details regarding delay

Background

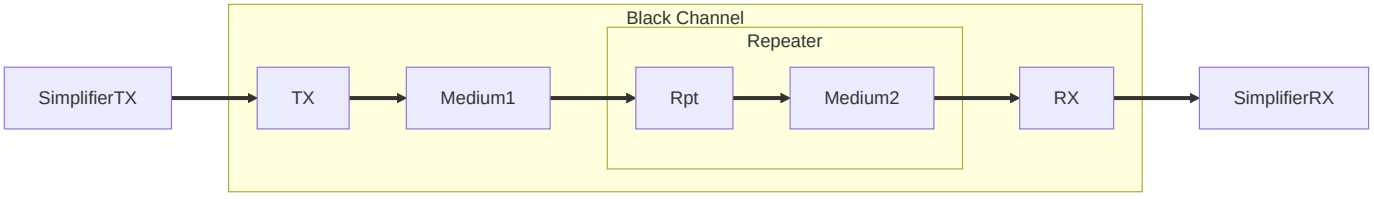
General design and definitions

The general design for simplex communication with optional repeaters is assumed to be a setup where the optional repeaters are in the middle of the black channel. Below are reference examples which are easily extended to arbitrary number of repeaters.

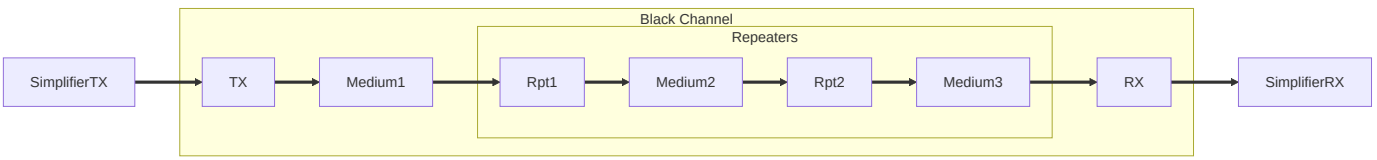
No repeater



One repeater



Two repeaters



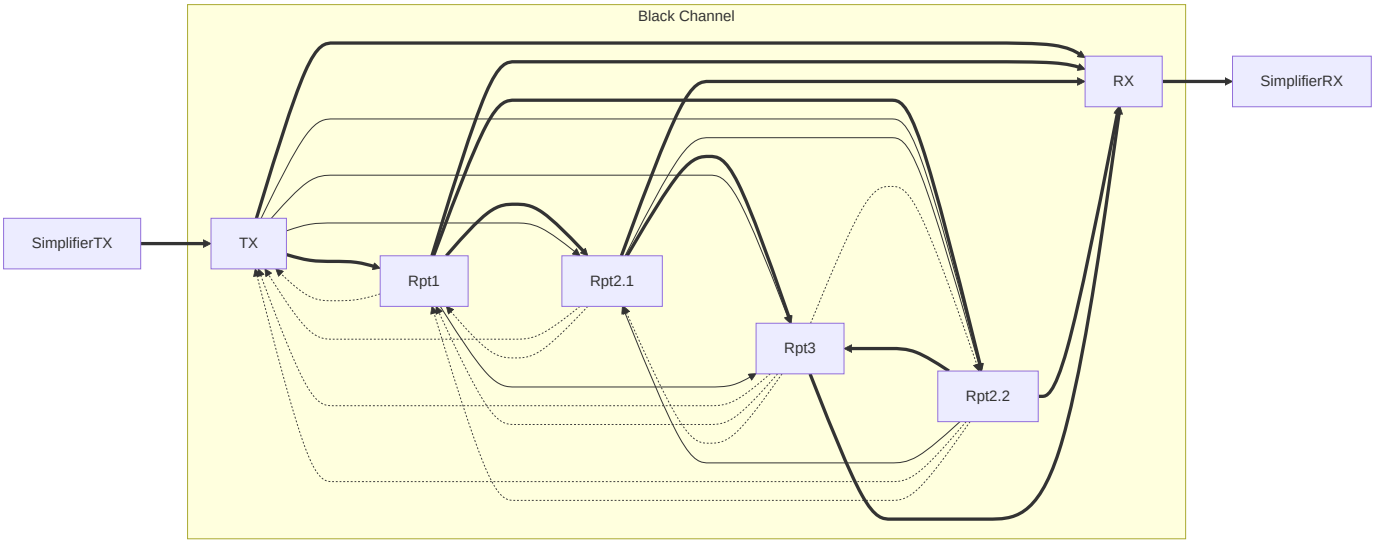
Paths

Depending on the black channel implementation, packets can travel through different paths. All variants are handled with standard black channel variants, so this is not critical for the specification. However, for understanding, one common setup is shown here.

Layered access

With this black channel setup, all units are connected in a layered/strict tree topology.

- layer0: TX
- layer1: Rpt1
- layer2: Rpt2.1 and Rpt2.2
- layer3: Rpt3
- layer4: RX
- thin lines: skipping a layer or to the same layer
- dotted connections: wrong direction



FMEDA

Repeaters in the black channel introduces several possible failures. However, as shown below in *XXX*, except for the simplex channel limitation of full control of the time delay, all failures are completely handled by standard mitigations. Obvious possible cases to handle are loops between the repeaters, same packet arriving several time, packets arriving out-of-order etc.

Unacceptable delay

To analyze the possible failures from unacceptable delay, the following diagram depicts the different delays that add up. Note that a delay can never be negative. Also, we neglect medium transfer delays as it is shown previously that within the applicable ranges, it will not contribute in any significant way.

Timing diagram

Here, every rectangle represents a time delay. BC indicates if the time is inside the black channel.

name	BC	description	typical time
SimplifierTX	no	the time it takes for the SimplifierTX to process and send out information received from the input. Note that this is outside the black channel, and need not be addressed by the black channel checks.	1-3 ms + input filter
UART_TX	yes	the actual UART transmission time to send out the packet.	bits / baudrate
TX_HANDLE	yes	time spent by the first radio transmitter to process/prepare and start send out the packet. Note that this can include CCA/LBT and possible retransmission time	1-10 ms?
RADIO1	yes	actual time for the radio transmission. Normally, the radio packet is larger than the actual safety information	<*total* bits> / baudrate
RPTn_HANDLE	yes	process the received radio packet and resend. Note that this can also include CCA/LBT and possible retransmission time	1-10 ms?
RADIOm	yes	same as RADIO1	<*total* bits> / baudrate
RX_HANDLE	yes	time in final radio module to start to send out the safety information on the UART	1-3 ms?
UART_RX	yes	the actual UART transmission time to send out the packet.	bits / baudrate
SimplifierRX	no	the time it takes for the SimplifierRX to process the received black channel packet. Note that this is outside the black channel, and need not be addressed by the black channel checks.	1-3 ms + input filter

Failure mitigation, most cases

The main mitigation for unacceptable delay is an extension to *IEC61784-3:2021 §5.4.4 "Time expectation"*. This, in combination with every packet including the SimplifierTX time stamp when it is created, results in all but one time related failure to be handled. The remaining failure mode is a constant delay error between actual and reported delay.

Definitions:

- `TRefTX`: SimplifierTX Time reference when create the packet. This is based on a 32 bit ms counter reset at start of the Simplifier.
- `TRefRX`: SimplifierRX Time reference when receive the packet. This is based on a 32 bit ms counter reset at start of the Simplifier.

- `TRefDiff`: The difference between the time reference in TX and RX at the time of packet creation. Can be both positive or negative.
- `ActDly`: Actual Delay. Can only be positive.
- `EstDly`: Estimated Delay. Can only be positive.
- `MarginDly`: The margin used to be on the safe side when using `EstDly` instead of `ActDly`. This is specified and verified by the black channel implementation.
- `DlyRange`: contract from the black channel to always have packets with delays in this range. This is verified with the extended release process.
- `drift`: The maximum drift between SimplifierTX and SimplifierRX, guaranteed to be max 100ppm+100pm = 200ppm according to the FSA of Simplifier.

The algorithm to detect and handle delay errors is based on analyzing a number of packets in a row. As the TRefTX and TRefRX is known for every valid packet, this is used to check that the variance between these match the TEstDiff, with the definition that `TEstDiff` = `TRefRX`-`EstDly`-TRefTX. Ie, `TEstDiff` is the estimation of the offset between TRefTX and TRefRX. With no errors, this difference should be constant except for the very small error introduced by the drift.
The implementation will check the min and max values of all measured values in the fifo, and go into fatal error if the different `max_val` - `min_val` > `MarginDly`.
Below section with examples show this. For clearnes, the actual `ActDly` `TRefDiff` is also shown.

Initial error

If there is an initial (constant) error for a while, and then receive correct EstDly, this will be detected and handled once the correct EstDly is received. However, before that, the receiver will act on the packets. To handle this, an additional delay `AddDly` is removed from the timeout margin until a packet with a longer delay is received. This is shown in Example #2.

Examples

Example #1

- min ActDly=10ms
- max(TEstDiff) - min(TEstDiff<=10)

TRefTX	ActDly	EstDly	TRefRX	TRefDiff	TEstDiff
1000	15	15	2015	1000	1000
1050	20	20	2070	1000	1000
1090	17	19	2107	1000	998
1140	18	16	2158	1000	**1002**
1180	20	10	2200	1000	**1010**
1220	10	10	2230	1000	1000

`max_val` - `min_val` = `12`

Example #2

TRefTX	ActDly	EstDly	TRefRX	TRefDiff	TEstDiff	AddDly
1000	10	10	2010	1000	1000	90
1100	50	50	2150	1000	1000	50
1200	100	100	2300	1000	1000	0
1300	10	10	2310	1000	1000	0
1400	100	10	2500	1000	1090	0

DlyRange range 10..100
max(DlyRange) - max(EstDly) = added delay

`max_val` - `min_val` = `90`

Example #3

TRefTX	ActDly	EstDly	TRefRX	TRefDiff	TEstDiff	AddDly
1000	90	10	2090	1000	1080	90
1100	90	10	2190	1000	1080	90
1200	100	20	2300	1000	1080	80
1300	10	10	2310	1000	**1000**	80

Failure mitigation, constant delay

If there is a constant error (ie, same in *every single* packet), it is not possible in a simplex solution to detect this without other information. It is worth noting that the only dangerous situation is if the `actual delay` > `estimated delay`.
To handle this, there is an additional organizational measure requirement to verify the design of all components in the black channel.

This measure is based on verification of the black channel design before commissioning any solution.

Requirements

Safety part requirements

- To check for max/min difference, at least all packets during max configurable timeout shall be checked.
- At startup and after a timeout, the receiver shall record and check at least packets for max configurable timeout before activating any safety function.

Black channel implementation requirements

1. Report the introduced delay.
2. Rigorous test for every design update as per below.
3. Design without FIFOs to minimize logic delay and keep the design simple.
4. Guarantee hard limit on min delay and max delay in the channel.
5. Guarantee precision of EstDly
6. the precision of EstDly (defined as a `+/- (precision/2)`), shall be less than 10% of the EstDly range (defined as `max(EstDly)-min(EstDly)`).

Black channel Verification requirements

- Whenever a design update is made, all requirement shall be verified before commissioning the update. This includes any software and hardware change in the black channel part.
- The verification shall be done on all valid paths through the black channel setup.
- For every path, tests shall be made that cover both good communication (very little interference/dropped packets etc), but also all reasonable situations with non-optimal communication. These different setups shall be documented.
- For every combination (path/interference level), at least 1000 packets shall be tested.
- Every packet shall be measured that it fulfills the contract regarding `MarginDly` and `DlyRange`.

Implementation details

as TRefTX is represented as 16 bit number => 65.536 secs, theoretically, time between packets, delay etc must be less than half. To avoid rounding errors and corner cases, the requirement is 1/4. Ie, 16.384 secs. Ie, the maximum configured communication timeout is 16.384 seconds.

to cover enough packets, at least all packets during 20 seconds shall be used to check for min and max. A 2 layer tree structure of packets is used for implementation efficiency. Ie, all packets during 2.25 to 3 seconds are used to calc min and max value. Then a fifo of these buckets (8 in length) is used to calc overall min/max.

Packet content

- 2B TimeRefTX [ms]
- 2B safety data
- HASH. Based on TX ID etc.
- CRC
- ID Code

container info:

- EstDly

- CRC

Testing

Internal notes. To be moved to another document.

RPico with 2 UARTs that simulate carefully crafted delays. Can also inject errors?
LED to indicate operation?
Get packet => FIFO. Timestamp when received packet. Use this to calculate EstDly.
USB:

- set delay range (min/max, uses random to variance)
- set error range (min/max)
- xor-value of payload? To simulate errors. Both on HASH and CRC.
- enforced PER value