Actionable Mitigation Options for J2497 Attacks

Introduction

This document’s purpose is to capture all known J2497 attack protection techniques known to-date and to reason about the solutions that could reasonably combine them so that a plan for development of fleet-actionable mitigations to the J2497 (PLC4TRUCKS) RF Induced Remote Write can be executed. Recall there are (at least) the following types of J2497 attack:

- RF induced (see the letter *Disclosure of Confirmed Remote Write,* NMFTA, January 2022)
- Malware-initiated, bitbanged (see the bitbanging transmitter proof of concept introduced in *Power Line Truck Hacking: 2TOOLS4PLC4TRUCKS*, DEF CON 28 Car Hacking Village, August 2020, slides: http://www.nmfta.org/documents/ctsrp/Power_Line_Truck_Hacking_2TOOLS4PLC4TRUCKS.pdf?v=1)
- Malware-initiated, well-formed (also see *Power Line Truck Hacking: 2TOOLS4PLC4TRUCKS)*

And the attacks are applicable to both trailer and tractor ABS controllers (and anything else that receives J2497 – but those are by far the most common pieces of equipment fielded today).

This mitigations survey document covers protections in the first section in some detail. The final section is on combined solutions which we believe are promising mitigation solutions for fleets. While IDS/IPS solutions are possible they are not covered in this document.

Protection Techniques

Consider the following protection techniques against the above types of attack. Descriptions of each technique follow the table below.

	Pros	Cons
PROT1 inline variable attenuators	• Passive components, relatively cheap and easy to install.	• Attenuates both received and transmitted signals. • Would only protect against RF induced attack and some bitbanged attacks • Requires tuning attenuator per equipment configuration. e.g. needs to be re-tuned when switching to double or triple configuration
PROT2 loading with priority override frames	• Simple blind-transmit defense (could bitbang it) • Possible against all types of J2497 attacks (but not 100% see cons)	• Interframe gap (required) is enough for malicious frames. • Attacker controlled transmitters don't have to respect frame priority • RFI noise
PROT3 trailer equipment sends priority override frames	• Mitigation against DoS of LAMP ON	• Only applicable to new equipment designs • Attackers can flood with priority override frames
PROT4 trailer wiring shielding	• Passive components, relatively cheap and easy to install.	• Unproven • Metal-decked dry-van result suggests wrapping trailer wiring in metal might not mitigate at all • But might work if left as a floating shield • Would only protect against RF induced attack
PROT5 RF chokes between chassis ground and wiring ground	• Passive components, relatively cheap and easy to install.	• Unproven, but should work based on our understanding of RF Induced attacks • Would only protect against RF induced attack

PROT6 chirp filter inline	• Stops all J2947 traffic, including malicious frames	• Stops all J2497 traffic, including LAMP ON messages needed to satisfy FMCSA regulations
PROT7 continuous dynamic address claimer	• Simple blind-transmit defense (could bitbang it) • Possible against all <u>types</u> of J2497 attacks (but not 100% see cons)	• Unproven • RFI noise • Will not protect old J249 equipment not supporting dynamic addresses • May allow intermittent unicast attacks • Does not protect tractor controllers • Might not prevent as-yet unknown exploit payloads and abuse commands that don't require unicast J1708
PROT8 loading with LAMP keyhole signal	• Simple blind-transmit defense (could bitbang it) • Should prevent exploit payloads and abuse commands • Possible against all <u>types</u> of J2497 attacks (but not 100% see cons) • Asymmetrically impacts high data rate signals more than low-rate LAMP	• Unproven, but initially confirmed on lab bench • Will not prevent LAMP ON attacks • RFI noise
PROT9 flooding with jamming signal	• Simple blind-transmit defense (could bitbang it) • Stops all J2497 traffic, including malicious frames	• Unproven, but initially confirmed on lab bench • Stops all J2497 traffic, including LAMP ON messages needed to satisfy FMCSA regulations • RFI noise

PROT1 inline variable attenuators

In RF-induced and most bitbanged attacks the signal amplitude of the attacker's J2497 signal is lower than that of the normal traffic on the powerlines. J2497 receivers have a minimum signal amplitude for reception of 5mVP-P according to the J2497 specification and also observed as practically 10mVP-P in testing. This small minimum signal amplitude enables small-signal bitbanging and RF-induced attacks. It is also necessary to have a small minimum signal amplitude because the technology needs to function on triple-trailers where the signals could be greatly attenuated between the last trailer and tractor brake controller needing to receive a trailer ABS fault message.

A defense against these small signal amplitude attacks is to attenuate (reduce) the signal amplitude of inbound powerline signals to the brake controller. This might not work in triple-trailer configurations but is possible in others.

PROT2 loading with priority override frames

In the trailer PLC research performed in collaboration with AIS and ultimately presented at DEF CON 28 CHV it was observed that it is possible to create J2497 frames with a MID that does not match the MID of their body J1708 content. This was also observed as default behavior for WABCO TCS II trailer ABS units in testing and development for this document. Since the J2497 MID should be used for arbitration, it is hence possible to create J2497 frames of an arbitrarily high priority irrespective of the J1708 MID priority.

A defense can be mounted using these by sending long frames with highest priority override (00).

PROT3 trailer equipment sends priority override frames

To avoid a DoS attack using priority override frames and/or to work in conjunction with PROT2, the trailer equipment could use priority override frames itself for LAMP frames.

PROT4 trailer wiring shielding

Perhaps the most obvious possible defense against induced RF: use shielded trailer wiring. It is also possible to try to shielded tractor-trailer ‘pigtail’ / ‘umbilical’ cables. The concept is worth discussing; however, due to the wavelengths of the frequencies involved and the triple-trailer results we have no reason to think pigtail/umbilical shielding would function as a mitigation. This protection, PROT4, pertains to shielded trailer wiring, not shielded umbilical cables.

The fact that dry-vans are less susceptible than tankers certainly suggests that having the trailer wiring run somewhere that isn’t ‘out in the open’ is better; however, the metal-decking dry-van result indicates that wrapping the trailer wiring in ‘too much’ metal makes susceptibility worse. We suspect that in the case of the metal-decked dry-van the chassis ground was joined to the wiring ground which ‘added’ susceptible metal to make a better antenna; hence the recommendation here for any wiring shielding is to try shielded trailer wiring where the chassis ground is left floating from the shield ground. This is captured below in Figure 1.

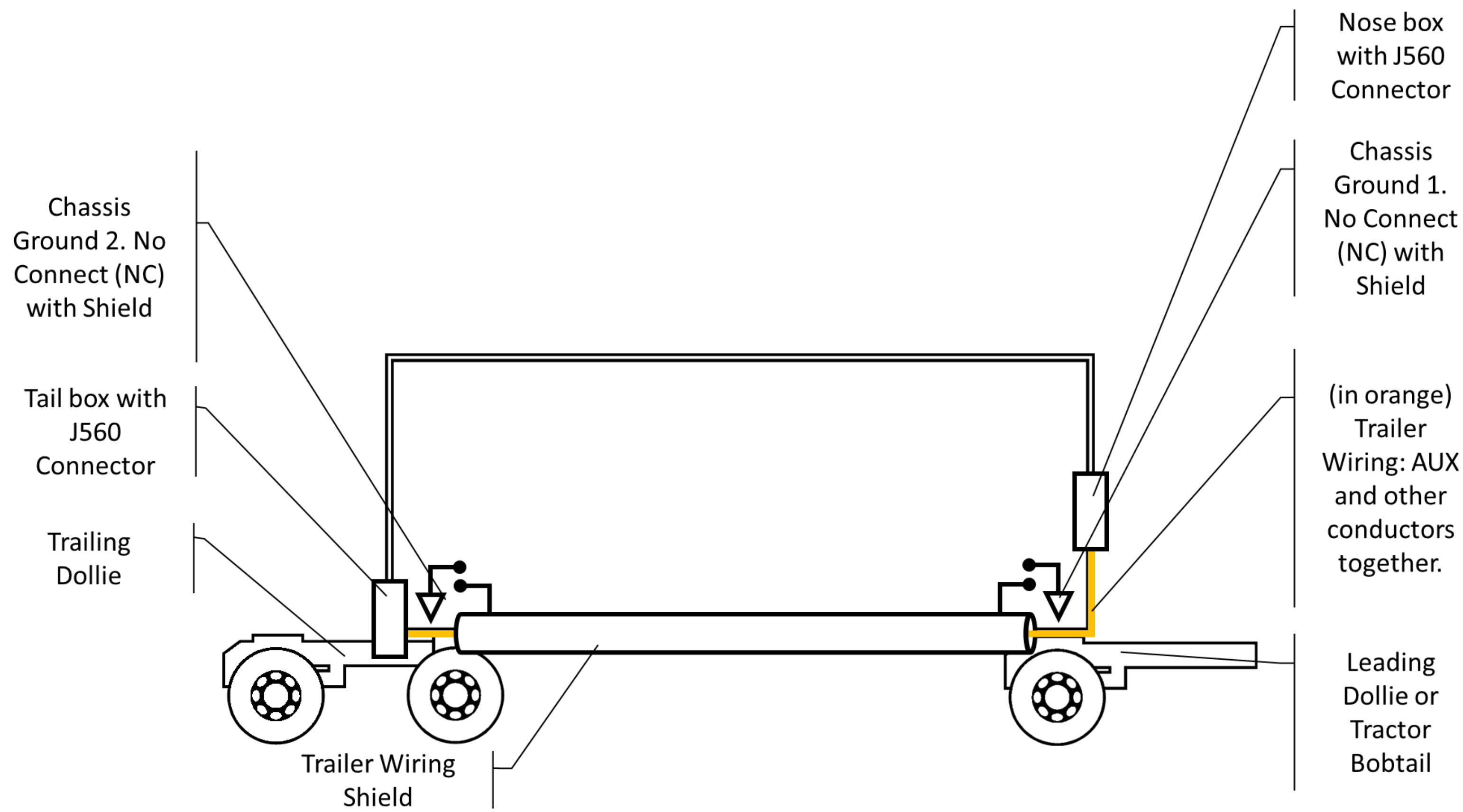

Figure 1 PROT4 trailer wiring shielding

PROT5 RF chokes between chassis ground and wiring ground

The fact that dry-vans with metal deck are more susceptible than those without suggests that the metal chassis has something to do with the susceptibility and we suspect that there is one or more galvanic connections from chassis to ground wire in the trailer wiring.

Therefore, reducing the galvanic connections to a minimum (ideally 1) and replacing each connection with an RF choke capable of suppressing the chirp band should reduce susceptibility. This is shown below in Figure 2. The performance of the RF choke needs to be able to attenuate any J2497 below the minimum receiver sensitivity. This is shown below in Figure 3.

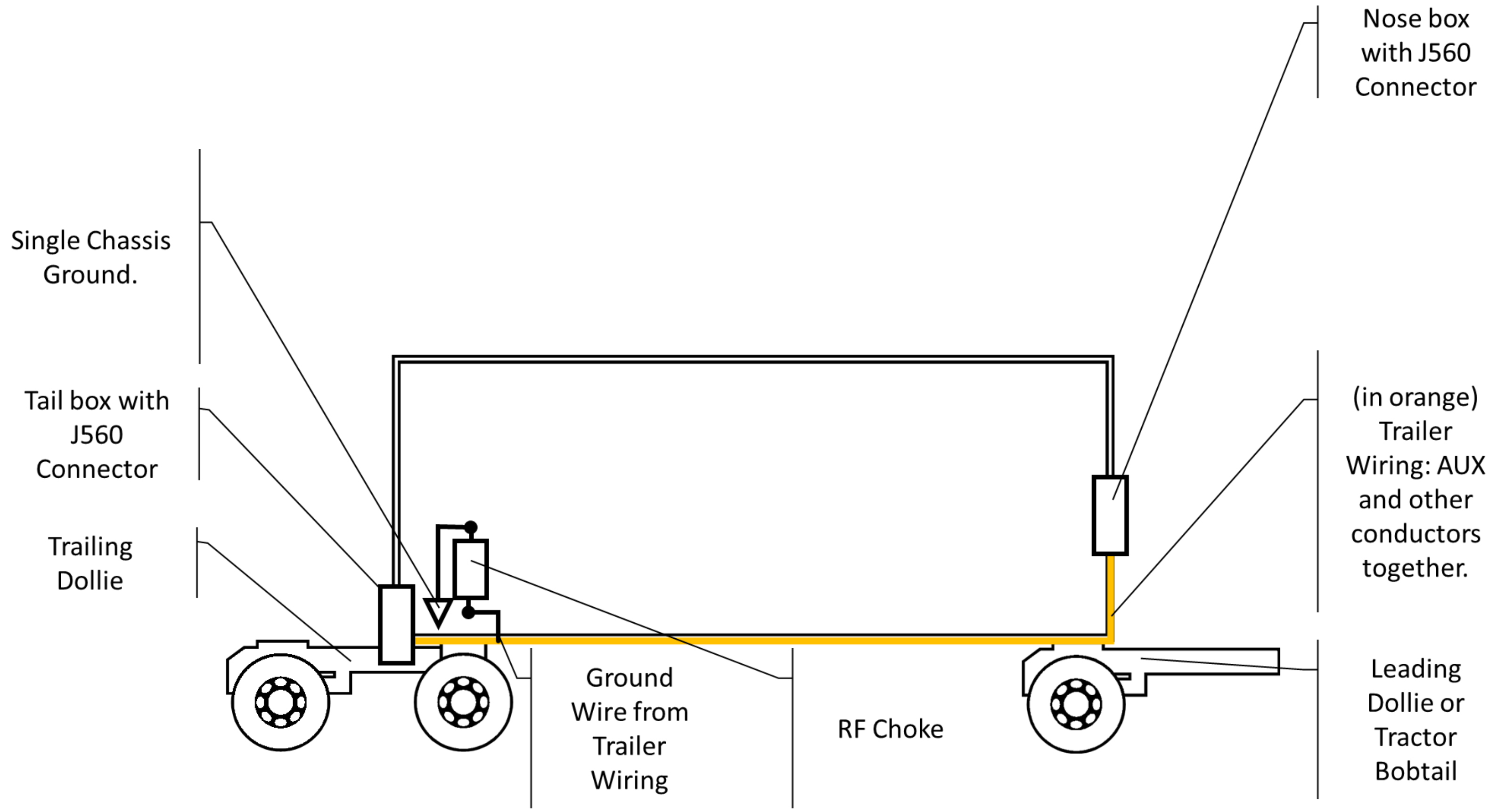

Figure 2 PROT5 RF chokes between chassis ground and wiring ground

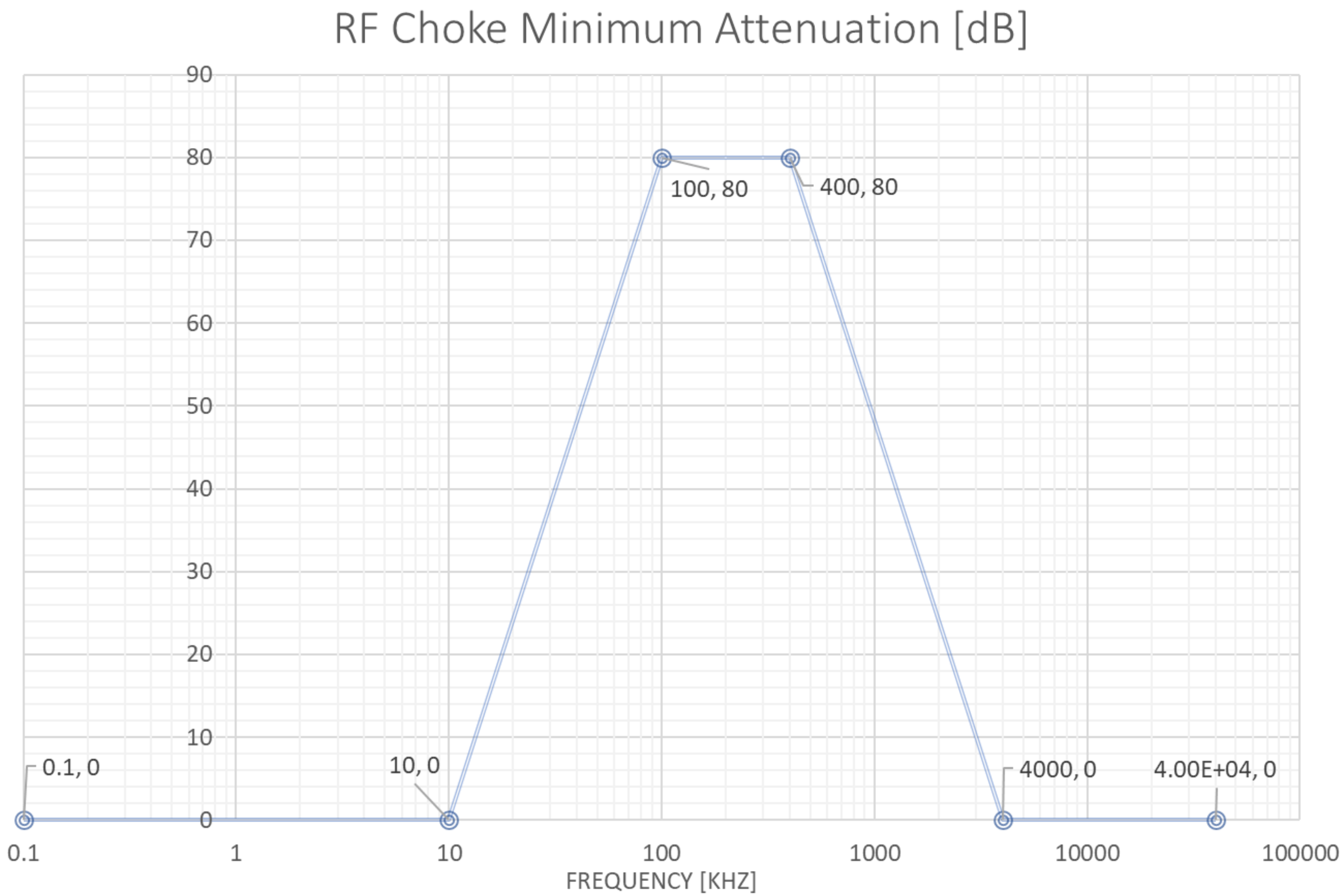

Figure 3 PROT5 RF chokes between chassis ground and wiring ground Minimum Attenuation

PROT6 chirp filter inline

Reception of any and all traffic can be inhibited by installing a filter in-line with the receiving equipment. This is shown below in Figure 4. The filter needs to attenuate signals in the chirp frequency range by at least 80dB for differential mode (typical J2497) signals and by at least 33dB for common mode signals. This is shown below in Figure 5 and Figure 6, respectively. This filter can be of a ‘lowpass’ or a ‘bandstop’ design. There are J2497 filters installed in tractors by some OEMs. These filters separate/remove/segment powerlines in the cab from the powerlines to the trailer by over-attenuating J2497 chirps that pass through them. Since the trailer ABS also controls the trailer ABS fault lamp with a relay-output the lamp control line also needs to be filtered and an RF choke as discussed in PROT5 should suffice.

The same technology could be packaged into an inline connector and installed on the trailer or tractor equipment. In the case of trailer equipment the connector is a standard Delphi/weatherpack 5 pin connector. For the tractor’s controllers it varies per supplier and model and the connectors are dense, complex and expensive thus an aftermarket inline solution is unlikely.

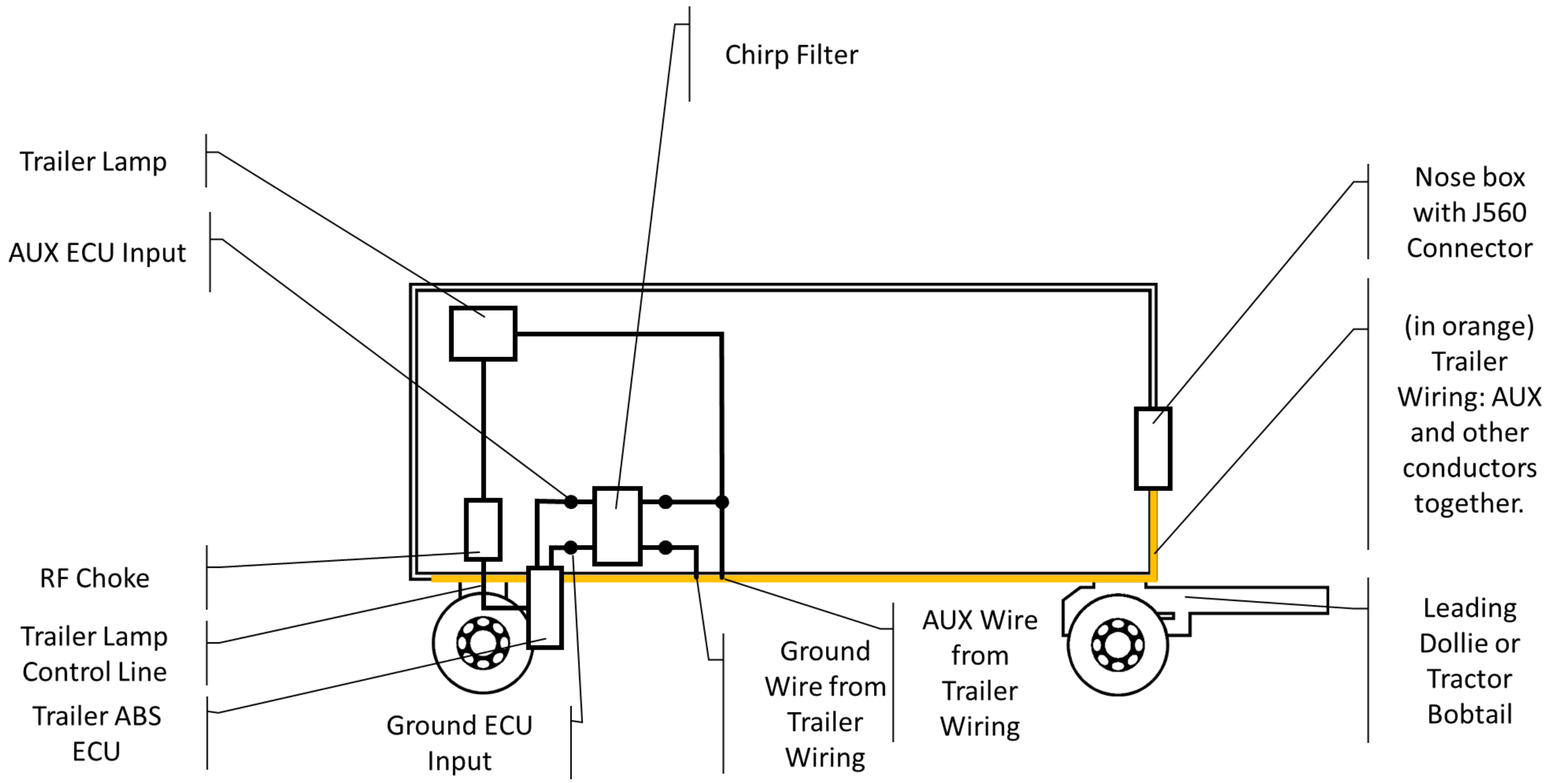

Figure 4 PROT6 chirp filter inline

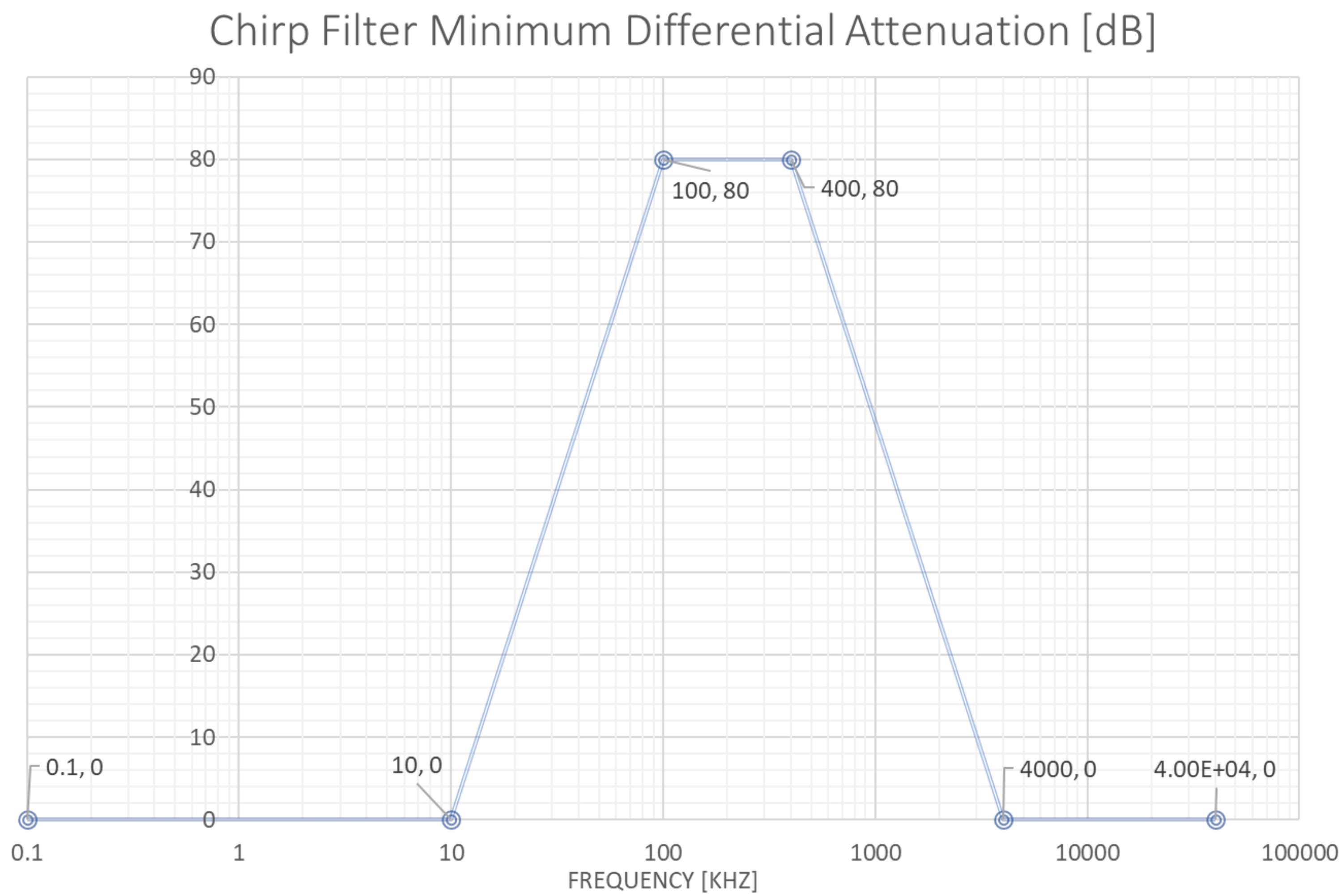

Figure 5 PROT6 chirp filter inline Minimum Differential Mode Attenuation

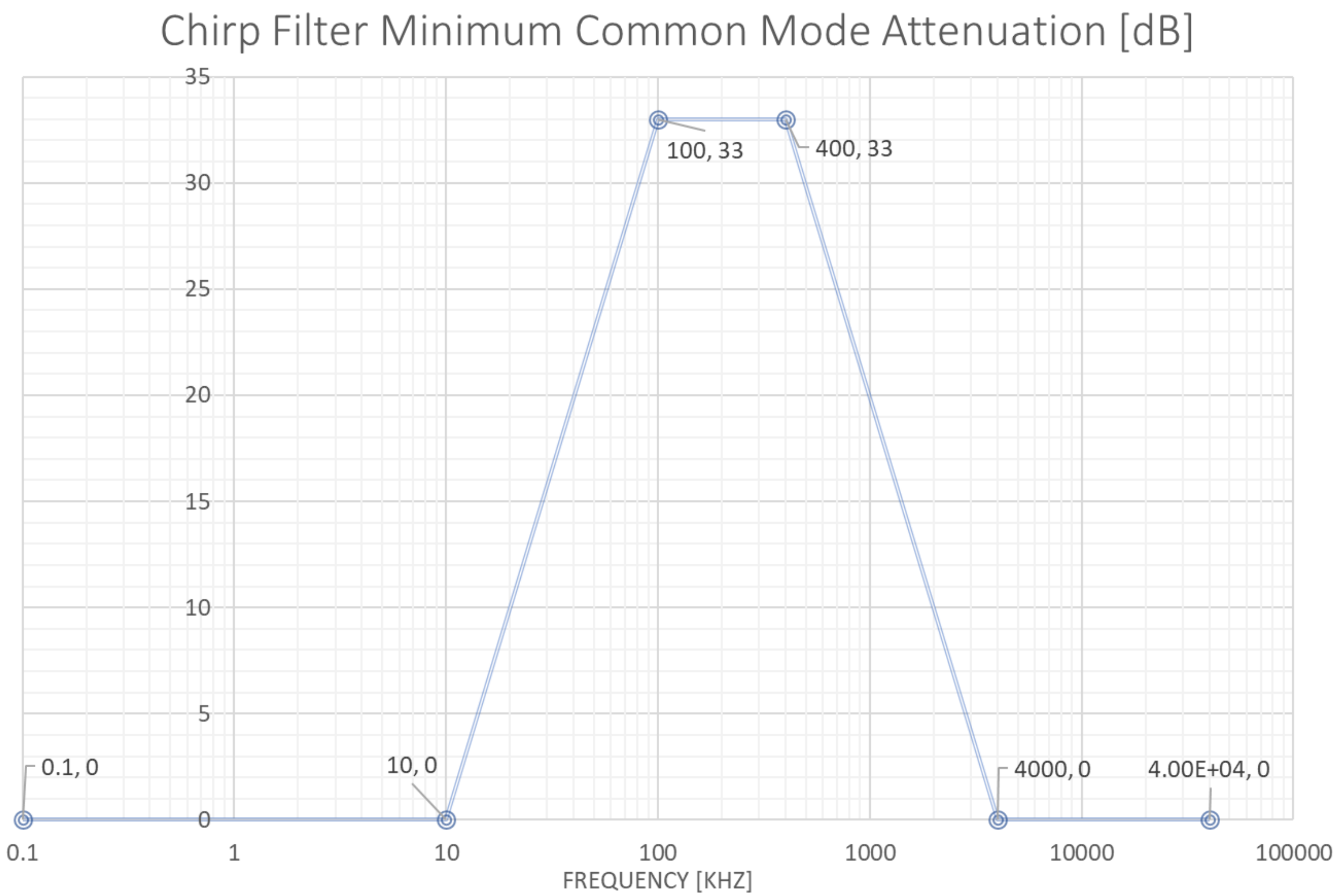

Figure 6 PROT6 chirp filter inline Minimum Common Mode Attenuation

PROT7 continuous dynamic address claimer

J2497 includes a dynamic addressing feature where all trailer equipment can change its J2497/J1708 MID (address) dynamically in response to detecting a transmitter on its current address. This is relevant as a defense option because all the dangerous J2497 frames encountered so far involve Data Link Escape commands (DLE) – PID 254 (0xfe). This proprietary space of commands is unicast, i.e. it requires a destination address and the J2497 equipment changes its address as mentioned above.

Assuming that all dangerous commands are also ultimately DLEs then attackers can be denied their malicious goal by denying them a destination address for the DLE. By repeatedly performing a dynamic address claim denial attack on the bus (insight and tests by Dan Salloum @ AIS) the receiving equipment can be forced to drop all incoming DLEs and/or change their unicast address often enough to make multi-frame DLE impossible and single-frame DLE sporadic.

Note that this address changing behavior is the practical behavior observed on trailer equipment and not the J2497 specification of dynamic addressing. The latter appears to not be implemented on trailer equipment. The available trailer Brake MIDs are 137, 138, 139, 246 and 247. The J2497 specification states that dynamic addressing should use MIDs in the range 88-110; however, in practice this range has not been implemented; only dynamic use of MIDs 137, 138, 139, 246 and 247 have been observed. The behavior of dynamic addresses appears to be the same as in the J2497 specification otherwise. Any trailer ECU that receives a (valid) message with an MID that conflicts with its own will 'move over' to a different MID. The message payload can be anything valid, in Dan's original concept a PID 4 'Dynamic MID claim' was used. To minimize potential impact on any J2497 networks and ECUs this defense should choose a payload that is short and has no effect on a J2497 network but is still valid. A Data Link Escape (DLE) message to a MID that can't be present should work: any receiver on the J2497 network will drop the message without further processing. We chose Engine #8 (MID 7). i.e. this defense is achieved by sending the following 3 byte payload messages in a loop, with a minimum time between send of 6ms:

- 137, 254, 7, 0
- 138, 254, 7, 0
- 139, 254, 7, 0
- 246, 254, 7, 0
- 247, 254, 7, 0

PROT8 loading with LAMP keyhole signal

The J2497 medium is multiple access and additive with all transmitters so two transmitters of the same power transmitting at the same time will likely corrupt each other's data for all receivers. The theory of this defense is to corrupt all J2497 messages for all receivers <u>except</u> a small list of allowed signals: just the required LAMP ON message for simplicity. According to the spec, transmitting continuous J2497 and then terminating it would work: All well-behaved transmitters will gallop together for that frame period but the priority of LAMP messages will win out. There are (at least) two problems. First, attackers do not need to respect the wait times of the spec and they can create priority override frames. Second, trailer equipment doesn't respect wait times either.

This first problem can be addressed by sending an almost-complete LAMP message (everything except the last couple bits of CRC) immediately after the corrupting signal. Only a LAMP ON message that aligns perfectly with the almost-complete ‘keyhole’ messages would validly complete the transmission. Achieving perfect alignment is tricky to accomplish in practice because transmitter phase is arbitrary and intercharacter delay is variable across suppliers. But all other messages on the wire, aligned or not, have their reception corrupted including attacker messages; only LAMP ON (when aligned to the keyhole) is permitted.

To address variability in transmitter phases, the solution is simple: try both possible phases (positive-going first, then negative-going first) in turn. For intercharacter delay, which can also be thought of as extra stop bits, our testing showed variability across suppliers. The WABCO TCS II has more inter-character delay than the Haldex TABS or Bendix TABS6. The WABCO unit stretches stop bits to the maximum two bit time length. The Bendix unit emits a variable intercharacter delay, always decreasing throughout the message, usually starting at 1 extra bit time, then 1, then 0 but 1-0-0 is also common. The Haldex unit follows the same behavior as Bendix. The solution is to try each of these sequences of extra stop bits for each of the phases also.

The other part of aligning the keyhole brings us to the second problem: trailer equipment transmitters do not follow the wait times in the specification. Each of the three pieces of equipment used in testing and development had a different unpredictable delay after bus-idle. Fortunately, it was found that the *delay before the target frame is sent* could be ‘groomed’ because it depended on the length and checksum of the frame sent prior. The relationship between prior frame length and the delay was such that if the prior frame was too short the interframe delay was unpredictable; this is surmised to be due to queuing in the transmitters. The result is that, for a sufficiently large prior frame size, the expected interframe delay was more predictable (for 2 of the three supplier’s units tested) but still different across suppliers. WABCO and Bendix controllers’ interframe delay is comparable but the Haldex unit was found to never queue regardless of prior frame size. The sufficient (minimum) size found for grooming the WABCO and Bendix controllers was 16 bytes payload; which is fortunately less than the specification’s maximum 21 bytes.

In Figure 7 below we show a selection of screenshots showing the interframe delay both across and within the three supplier’s devices discussed above for a prior frame length at the 21 byte J1708 maximum with a valid CRC.

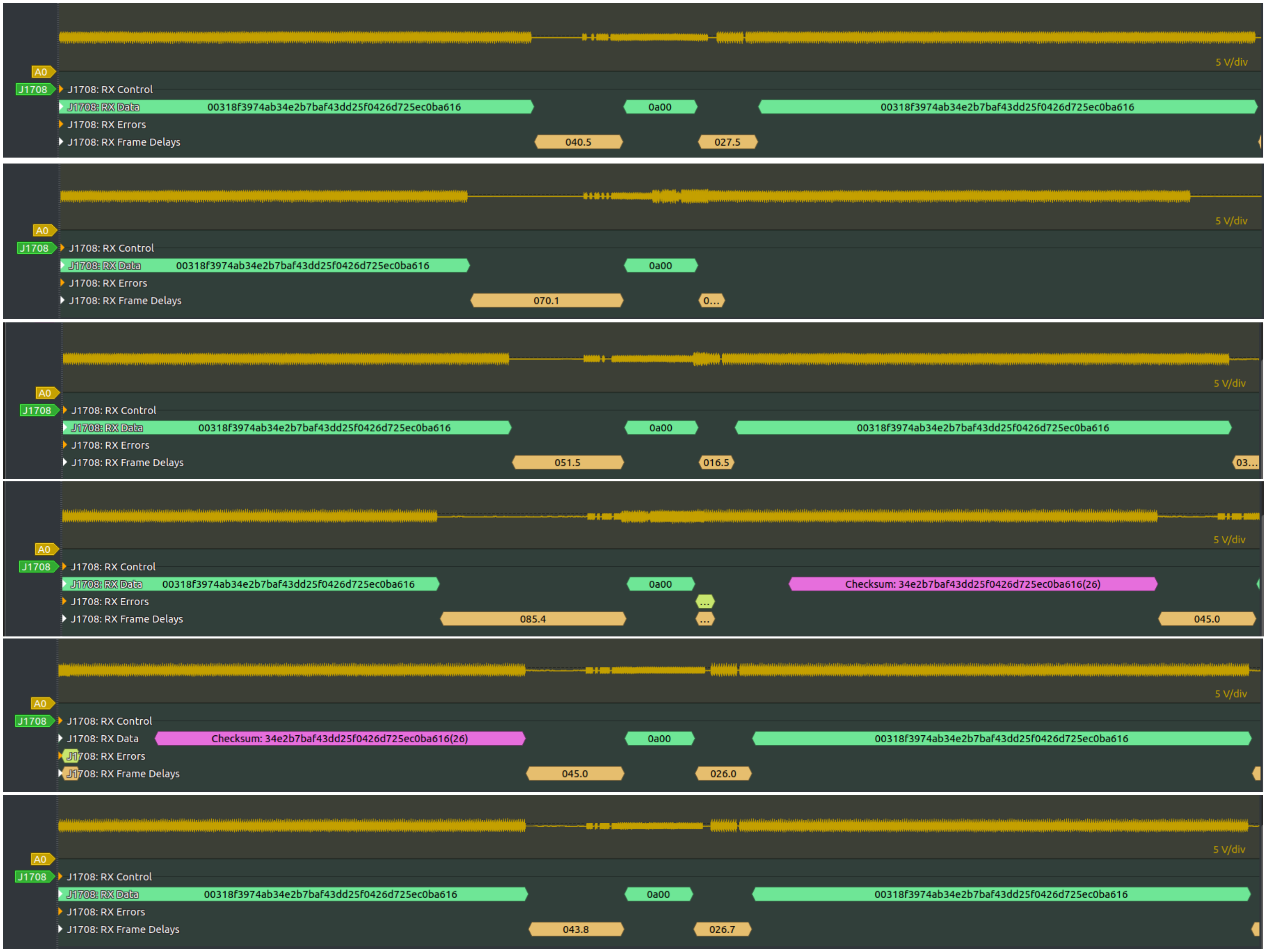

Figure 7 Examples of variable interframe delay, 0a00 LAMP ON message only

Figure 7 captures the variability of the (groomed) interframe delay after a 21byte payload, correct-CRC message of random bytes. The preceding frame used to groom the interframe delay, the ‘door’ signal, ideally causes as little impact as possible on CPU resources of connected devices and also has no ‘side effects’ on the devices either (e.g. crashes, chuffs, etc.). For reasons similar to PROT7 continuous address claimer, selecting a DLE to engine #8 will be unlikely to affect anything; furthermore, using the minimum (grooming) length of 16 bytes is best to minimize CPU resources. Finally, sending a CRC-corrupt door signal was chosen because the result is that the door signal will not show up in logs and hopefully will get dropped at the earliest processing steps in receivers.

The expected interframe delays were calibrated by measuring the UART times (as depicted in Figure 7) after the CRC-corrupted door signal described above. For the WABCO TCS II the most common delays were 45.0 and 40.6 bit-times (UART 9600bps 104.17 us). For Bendix TABS6 the most common delays were 39.5, 40.6, and 46.1 bit-times. For the HALDEX TABS – as mentioned above – the delays could not be groomed. The Haldex unit transmits its frames periodically, regardless of the state of the bus. To accommodate this the only thing to do is create keyholes as quickly as possible and to ensure that the period between keyholes doesn’t align with the Haldex transmit period of 500ms.

Creating keyholes as quick as possible would also help ensure that trailer ABS fault telltales are displayed to drivers rapidly enough to satisfy the regulations. All the possible combinations of delays and phases totals 10. The total set is transmitted every ~320ms; all three units: Bendix TABS6, WABCO TCS II and Haldex TABS sends LAMP ON every 0.5 s. Empirically the average wait before a LAMP ON is matched is 1.5s for Bendix, 7s for WABCO and 12.5s for Haldex. The performance of the keyhole signal as described thus far on the three units is presented in Figure 8 below. Even in the worst case, the delay in between LAMP ON messages is less than 35s.

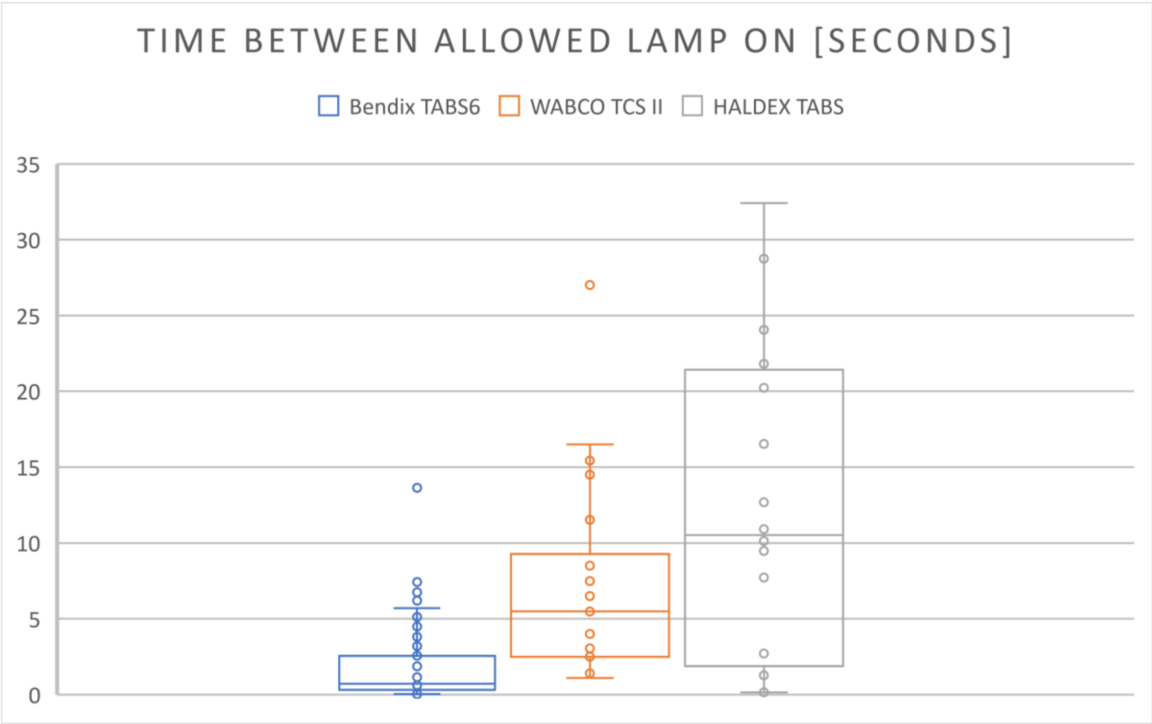

Figure 8 Test Results for time between allowed LAMP ON messages

The FMCSA regulation governing trailer ABS fault display to the driver has no language requiring time limits on transmission of a fault. The J2497 standard requires that systems displaying the trailer ABS fault turn on the indicator for 2.5 seconds in response to reception of the command; it also requires that systems turn off the indicator if no messages are received for 10s. Therefore, the result of taking 30s to get around to the correct parameters to match the LAMP signal is a blinking indicator with either 2.5s or 10s on and a period of 30s. This doesn't appear to violate the regulations and would still communicate the failure to the driver.

There are some other details of creating the sequence of door and keyhole signals to be replayed as a defense. The full details are captured in the source code Listing 1 below. There are two more parts worth some explanation here. The first, since it can also be used as a protection all by itself: the 'jamming signal.' The second, since it is an important optimization for all J2497 transmitters: the J2497 preamble is not needed at all.

The spread spectrum chirps employed in J2497 make the receivers quite robust in the presence of noise during the body phase of a signal – even to the presence of other J2497 chirps that are out of phase. For example, in Figure 9 the SSC P485 is seen here happily locking on to and receiving a comparable power signal even though the door signal arrives right in the middle of it (in the body phase / PSK).

Figure 9 J2497 receiver locking on to a slightly stronger signal during reception of another signal

In the preamble phase the receivers are more susceptible to corruption/interruption. For example, in Figure 10: reception is stopped when that collision occurs in the preamble (ASK) phase:

Figure 10 J2497 Receivers are more sensitive to corruption during the preamble phase

The fact that the receivers are more sensitive to disruption in the pre-amble phase means that a keyhole approach where the allowed signal preamble and body fragment are emitted first works better at stopping unwanted signals than properly sized segments of deadair; however, this also means that even with the door signal successfully blocking signals a clever attacker that know this could ‘race’ the keyhole. Because as a rule of thumb for signals of equal power, whichever signal gets their preamble (most importantly the SYNC train) out first will have their body received correctly.

There is little promise of stopping signals after the preamble for signals of comparable amplitude. It is tenuous to assume that the door signal would be higher power than an attacker’s signal, even in the RF induced case where attackers are inducing signals of very low signal amplitude. Anecdotally, during testing of triple trailers the RF induced signals were found to be received more reliably by the last trailer than signals produced by a PLC TestCON diagnostic adapter connected to the tractor-trailer J560 connector. It seems that the signal amplitude in triples could be itself marginal and also could vary along the trailer combination as a result of transmission line effects. This is not even to mention that the types of attacks other than RF-induced can produce signal amplitudes comparable to all signals emitted by the tractor-trailer equipment. Thus, all things considered, the defense should not be designed to assume that the defending technology will always have a larger amplitude than the attacker.

The longest interframe delay before emitting a keyhole is ~5ms. This could admit an attacker-controlled 3 byte payload frame. This isn’t big enough to allow any of the known chuff or roll call commands but it’s entirely possible that there are high impact 3 byte payload J2497 frames. The solution to prevent this is sending a signal which corrupts reception by other receivers but does pre-empt the transmissions, so that our keyhole still has a chance of matching an allowed message. A signal that can corrupt a frame without resetting idle detection (as in the solution to the Haldex extra delay) could also be used here to block any attackers racing with the small gap. This important jamming signal is alluded to in the J2497 specification where it says “The PLC for trucks technology is more sensitive to constant carrier interference than to broadband interference […]” and testing revealed that a constant carrier in the 300 – 400 KHz range works well for this purpose: it corrupts reception but does not trigger idle end detection in the J2497 receivers. In testing we found that 376.369KHz is the best constant carrier for this purpose.

There is another important fact that enables the functioning of the keyhole signal: the J2497 preamble phase is not needed at all to cause receivers to emit bits in the body phase. If a J2497 body-phase signal is transmitted (from the 5 sync symbols onwards) then a J2497 receiver can successfully decode it into UART signals. During development, when using a preamble on the keyhole signal the most common result was the transmitter that was carefully aligned-to would stop transmitting because the preamble collision was detected. Exploiting the fact that the preamble is unnecessary lets the keyhole align to a transmitter without that occurring.

It turns out that the susceptibility to corruption of J2497 in the preamble phase is a double-edged sword. This property works against the desired outcome of the door signal by both causing queuing of frames in transmitters by blocking those transmissions and triggering end of idle. A period of the constant carrier interfering signal can be added after the keyhole to ensure that messages which are late or longer and do not match are blocked. But whatever the length of the later jamming signal was extended to during development there were always cases where signals would get through at the point between the late jamming signal and the next door signal. Since the preamble is not needed at all this is also an excellent candidate for dropping it too.

The keyhole signal mitigation can be constructed (depicted in some detail in Figure 12 below) and can function at blocking most non-allowed signals at comparable power. There are some unpredictable aspects to the J2497 receivers and 100% guarantees cannot be made; however, testing at comparable power of the keyhole signal and an attacker signal results in less than 1 in 1000 3byte payloads getting through and zero payloads of 4 or more bytes getting through for tests of 5mins. During these tests LAMP ON messages were also allowed through successfully as well. In Figure 11 below is an example of a keyhole mitigation signal lining up with a trailer ABS ECUs LAMP ON message and being successfully decoded by the J2497 receiver under test and a descriptive diagram of the phases of the keyhole signal. In Figure 13 a detailed capture of another example of a successful alignment of a keyhole is presented, showing the phases of the signal which are also presented in Figure 12.

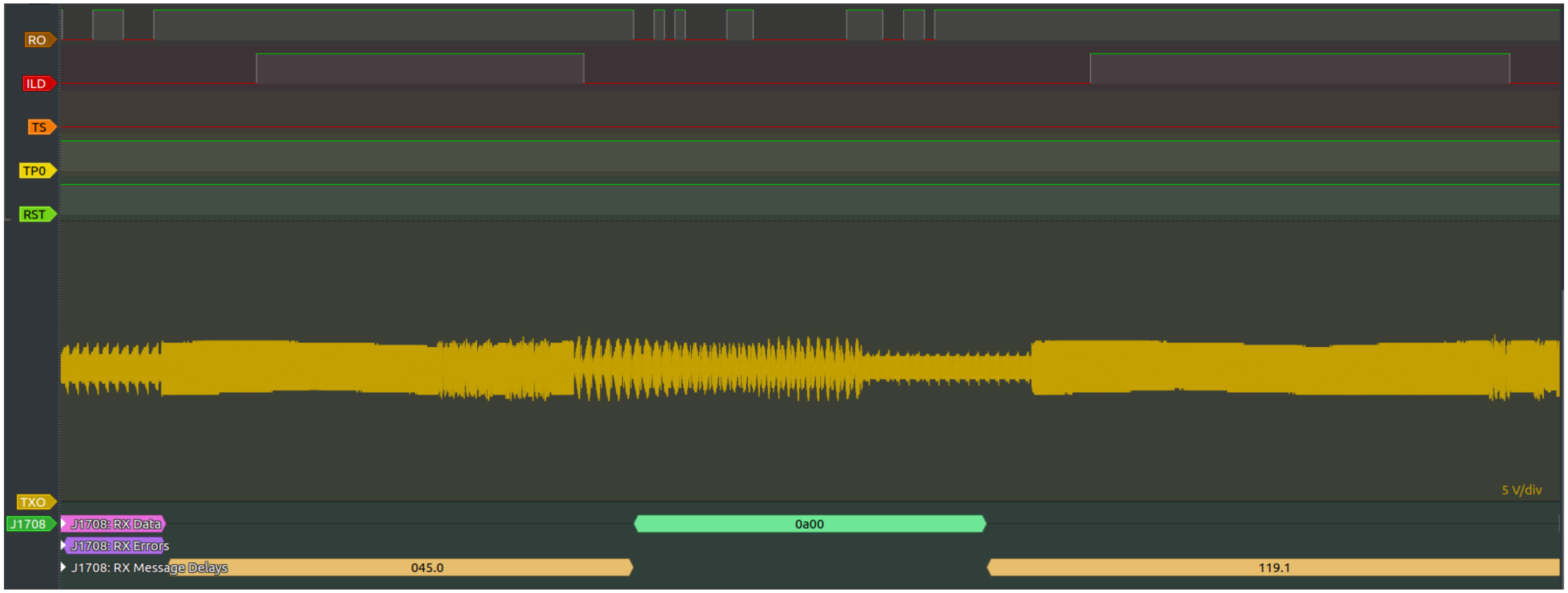

Figure 11 Example of successful keyhole signal example: matching Trailer ABS ECU's emitted 0a00 LAMP ON message

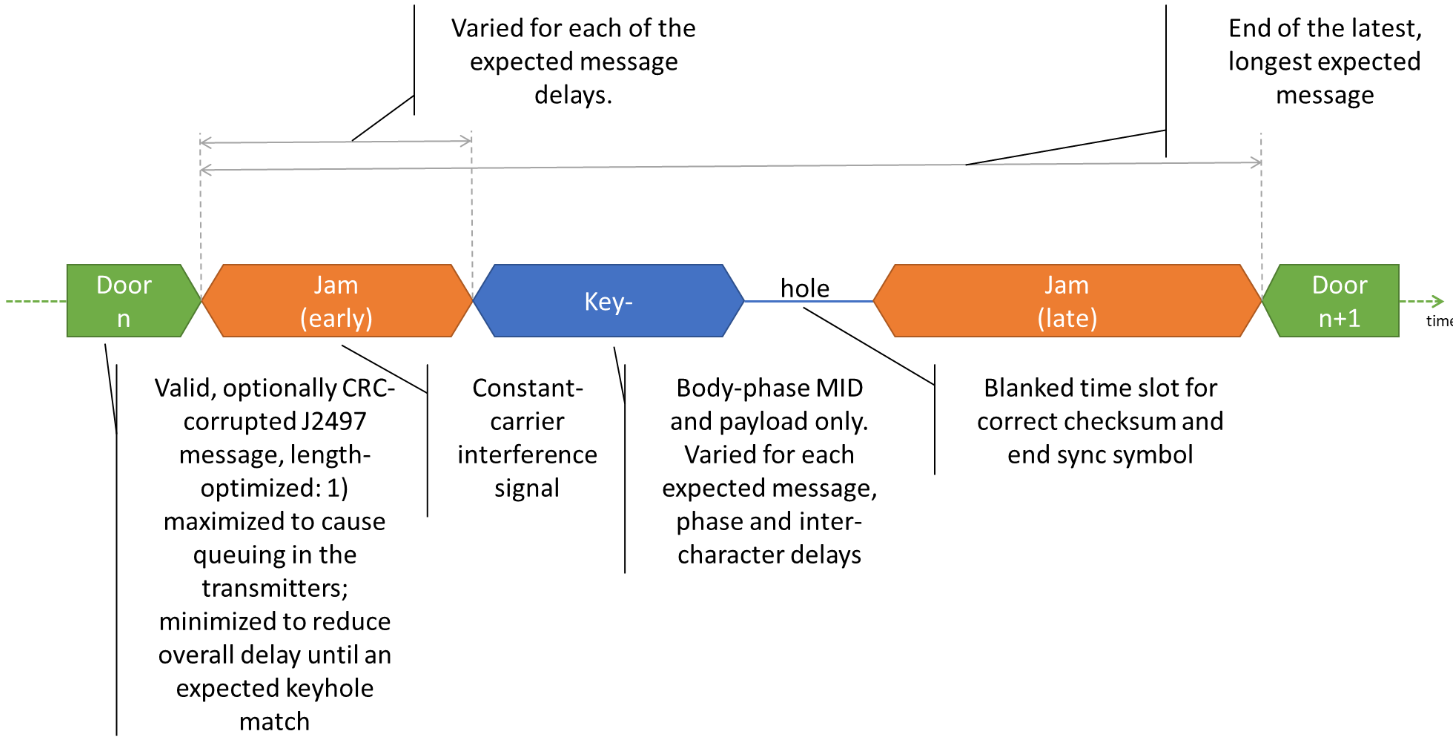

Figure 12 Explanation of phases of keyhole mitigation signal

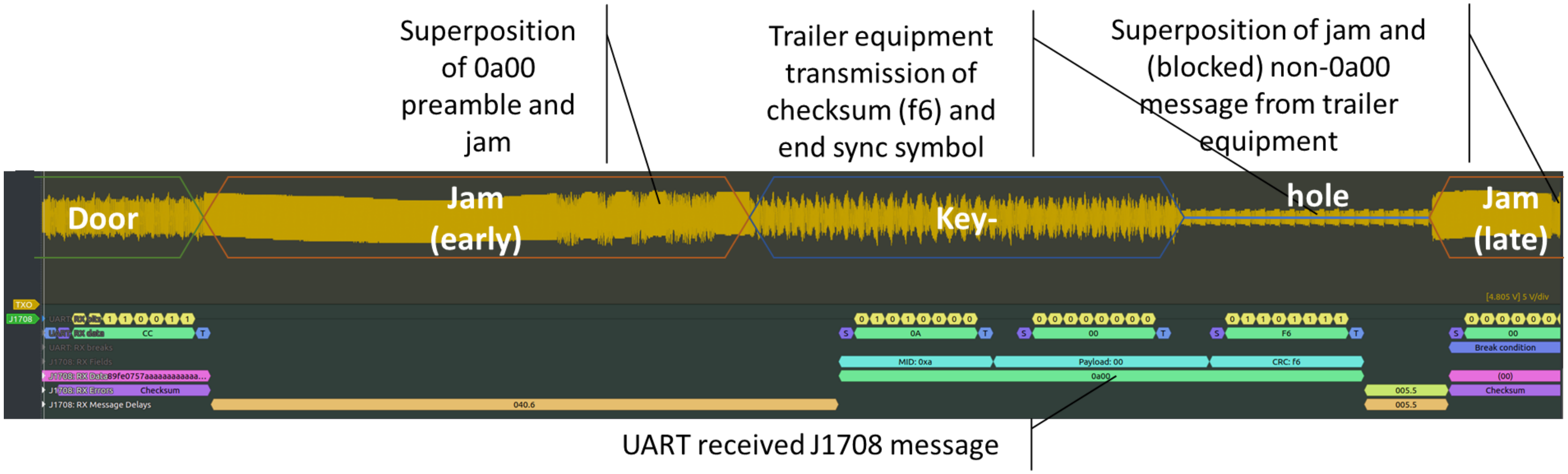

Figure 13 Description of a successful keyhole signal example

The full details of how to construct a keyhole signal train for a given set of allowed messages is captured in the MIT-licensed python3 source captured below in Listing 1. This is a python module whose public function `generate()` will yield an array of signals to be played back in a loop on a DAC or bit-banged on a GPIO. Please see the docstring of `generate()` for more details.

```
# Copyright (c) 2022 National Motor Freight Traffic Association Inc.
#
# Permission is hereby granted, free of charge, to any person obtaining a copy
# of this software and associated documentation files (the "Software"), to deal
# in the Software without restriction, including without limitation the rights
# to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
# copies of the Software, and to permit persons to whom the Software is
# furnished to do so, subject to the following conditions:
#
# The above copyright notice and this permission notice shall be included in all
# copies or substantial portions of the Software.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
# FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
# AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
# LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
# OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
# SOFTWARE.

import binascii
```

```python
import bitstring
import itertools
import numpy as np
from scipy.signal import chirp

DEFAULT_ALLOWED_MESSAGES = [b'\x0a\x00', ]  # LAMP ON only by default

DEFAULT_SUPPLIER_PARAMETERS = [
    {  # WABCO 0a00 measured @ after crc-corrupted 16byte payload door signal
        'label': 'wabco tcs ii 2s1m basic msh 400 500 101 0',
        'expected_delays': [45.0, 41.7, ],
        'extra_stop_bits': [2, 2],  # tends to do 2 extra stop bits followed by 2 extra stop bits (but can vary)
        'expected_phases': [-1, 1],  # tends to use one phase over the other but just use equal probability
    },
    {  # Bendix TABS6 0a00 measured @ after crc-corrupt 16B payload door signal
        'label': 'bendix tabs6 5014016 ES1301 K003236',
        'expected_delays': [47.2, 41.7, 40.6, ],
        'extra_stop_bits': [1, 0],  # tends to do 1 extra stop bits followed by 0 extra stop bit (but can vary)
        'expected_phases': [-1, 1],
    },
    # {  # Haldex TABS 0a00 measured @ after crc-corrupt 16B payload door signal
    #    'label': 'haldex tabs H16 0676',
    #    'expected_delays': [46.1, ],
    #    'extra_stop_bits': [1, 0],  # tends to do 1 extra stop bits followed by 0 extra stop bits (doesn't vary)
    #    'expected_phases': [-1, 1],  # only one phase observed in testing
    # },
    # because Haldex TABS doesn't queue messages to send, picking any expected delay is fine and because both of the
    # other parameters match the Bendix unit, it is sufficient to omit these supplier parameters
]

# There is a minimum period for the keyhole signals which was discovered during testing. Bendix TABS6 transmitters
# verify their sends and will retry if their transmission is corrupted, which is great! Except that they also have a
# priority inversion bug so if they can't successfully transmit a lower priority message e.g. 89c20302b502 any
# higher-priority messages e.g. 0a00 (LAMP ON) will queue. If the door+keyhole signals are transmitted too rapidly
# then the TABS6 will trigger this priority inversion bug and there will be _no_ required LAMP messages. We also send
# all-jam periods sometimes to reduce the chance of forever-retries due to keyholes corrupting the signals too
MIN_PERIOD_US = 32000
DEFAULT_PERIOD_US = MIN_PERIOD_US


def generate(sample_rate, allowed_messages=None, keyhole_supplier_parameters=None, period_us=None,
             calibration_mode=False):
    """
    Use this function to get a complete set of keyhole mitigation signals. Play them on a loop to prevent all but the
    allowed messages from being received by any J2497 receiver on the powerline segment.

    The signals will probably need to be prepared for playback on your DAC. They will work even when played on a
    bit-banged DAC (1-bit / PWM etc.). Here's an example of preparing the signals for playback @ 1Msps on a signed
    8bit output:

    dac_ready = [(x * 127).astype('int8').tobytes() for x in j2497_keyhole.generate(1E6)]
```

```python

    And these can be played back-to-back on a loop e.g.

    while True:
        for s in dac_ready:
            dac_device_driver_api.write(s)

    If you want to bit-bang the output, we have found that even the simplest PWM rule will work:

    bangs = [(x >= 0) for x in j2497_keyhole.generate(10E6)]

    Note that there is no dynamic generation required. The signals can be pre-computed and played back from non-volatile
    storage as well.

    The interframe delays from most J2497 transmitters depend on the length of the allowed messages,
    the crc-corrupted state of the door signal and the period of the signals. Changes to any of these should be
    followed by re-calibrating the measured delays and updating the supplier parameters.

    :param sample_rate: sample rate of resulting signal, must be at least 800KHz, 1MHz is good
    :param allowed_messages: messages to allow via keyholes
    :param keyhole_supplier_parameters: supplier keyhole parameter list
    :param period_us: period of the door+keyhole signals generated
    :param calibration_mode: if true, generate modified waveforms used to calibrate the supplier parameters
    :return: an iterator of np array signals of float32 values in [-1.0, 1.0]
    """
    if sample_rate < 800E3:
        raise ValueError("sample rate must be >= 800 KHz")
    if keyhole_supplier_parameters is None:
        keyhole_supplier_parameters = DEFAULT_SUPPLIER_PARAMETERS
    if allowed_messages is None:
        allowed_messages = DEFAULT_ALLOWED_MESSAGES
    if period_us is None:
        period_us = DEFAULT_PERIOD_US
    jam_amplitude = 1
    if calibration_mode:
        # to calibrate supplier parameters the keyholes must be suppressed to measure expected delays
        # and jams must be suppressed to receive J1708 and hence measure expected delays
        jam_amplitude = 0

    assert period_us >= MIN_PERIOD_US
    # it is important for transmitters that don't queue LAMP ON (e.g. Haldex) that multiples of the period of the
    # door+keyhole do not align with the 0.5s period of the LAMP messages sent. We take anything within a sync
    # symbol width as 'alignment'.
    period_samples = int(period_us * sample_rate / US_PER_SEC)
    remainder = (0.5 * sample_rate) % period_samples
    alignment_limit = len(SYNC_BITS) * BODY_BIT_TIME_US * sample_rate / US_PER_SEC
    assert remainder > alignment_limit
    assert period_samples - remainder > alignment_limit

    doors = _door_signals(sample_rate)
    # combining doors and keyholes with `next(cycle(doors))` in the for loop below is fine if there are more keyholes
```

```python
    # than there are doors. This is a generator, so confirm that after the loop -- see below
    doors_len = len(list(doors))
    doors = itertools.cycle(_door_signals(sample_rate))

    keyhole_count = 0
    for keyhole in _keyhole_signals(sample_rate, allowed_messages, keyhole_supplier_parameters, calibration_mode):
        keyhole_count = keyhole_count + 1
        door_n_keyhole = np.append(next(doors), keyhole)
        assert len(door_n_keyhole) < period_samples
        late_jam = jam_amplitude * _get_jam(sample_rate, period_samples - len(door_n_keyhole))
        door_n_keyhole = np.append(door_n_keyhole, late_jam)

        yield door_n_keyhole
    # confirm that there were, in fact, at least as many keyholes than doors
    assert keyhole_count >= doors_len

    # We need to send all-jam periods sometimes to reduce the chance of triggering a priority inversion bug. See the
    # MIN_PERIOD_US comments for more details.
    all_jam = next(doors)
    assert len(all_jam) < period_samples
    the_jam = jam_amplitude * _get_jam(sample_rate, period_samples - len(all_jam))
    all_jam = np.append(all_jam, the_jam)
    yield all_jam

    return


def _door_signals(sample_rate):
    """
    Generates 'door' signals whose purpose is to hold J2497 transmitters in wait, causing them to queue their
    messages to be sent and thus grooming the expected delays to better the chances of a keyhole aligning perfectly
    with an allowed message.

    All the values in DEFAULT_SUPPLIER_PARAMETERS are measured using the values below. Any changes to the
    payload or CRC necessitate re-calibrating DEFAULT_SUPPLIER_PARAMETERS.

    :param sample_rate:
    :return: a numpy float32 array of samples valued in [-1.0, 1.0]
    """
    # TODO: vary MID `89` through all possible trailer ABS MIDs [ 0x89, 0x8a, 0x8b, 0xf6, 0xf7 ] to _also_
    #   perform an address denial mitigation at the same time as the keyhole protection. Will need to both use correct
    #   CRC and also re-calibrate the values in DEFAULT_SUPPLIER_PARAMETERS
    mids = [b'\x89', ]
    for mid in mids:
        door_bits = _get_payload_bits(mid + binascii.unhexlify('fe0757aaaaaaaaaaaaaaaaaaaaa71c'),
                                      checksum=binascii.unhexlify('cc'))  # Correct CRC is `b4` ~( ̄▽ ̄~)
        yield _get_payload_chirps(door_bits, sample_rate)


US_PER_SEC = 1e6
UART_BIT_TIME_US = 104.17  # i.e. 9600bps
```

```
BODY_BIT_TIME_US = 100  # J2497 body bit time
SYNC_SYMBOL_TIME_US = (5  # bits in start sync symbol
                       ) * BODY_BIT_TIME_US

# Intellon ssc p485 measured J2497 -> UART latency. Needed because measured/expected delays are UART delays
FROM_J2497_OVER_TO_UART_OVER_US = 48.3
# time duration for crc and the rest of a message after the payload
TIME_AFTER_PAYLOAD_US = (1  # start bit
                         + 8  # bits in crc byte
                         + 1  # stop bit
                         + 7  # bits in end sync symbol
                         ) * BODY_BIT_TIME_US


def _keyhole_signals(sample_rate, allowed_messages, keyhole_supplier_parameters, calibration_mode):
    """
    Generates keyhole signals which will permit only J2497 messages with matching payloads in the allowed messages list.
    There are multiple possible keyholes which are generated according to the combinations of the supplier parameters
    given in keyhole_supplier_parameters.

    To calibrate your own keyhole_supplier_parameters, set calibration_mode to true and measure some J2497 waveforms!

    :param sample_rate:
    :param allowed_messages: messages to permit by matching keyholes
    :param keyhole_supplier_parameters: delays, extra stop bits and phases to match for keyholes specific to devices
    :param calibration_mode: set to true to make keyholes that can be used to calibrate keyhole_supplier_parameters vals
    :return: a numpy float32 array of samples valued in [-1.0, 1.0]
    """
    keyhole_amplitude = 1
    jam_amplitude = 1
    if calibration_mode:
        # to calibrate supplier parameters the keyholes must be suppressed to measure expected delays
        keyhole_amplitude = 0
        # and jams must be suppressed to receive J1708 and hence measure expected delays
        jam_amplitude = 0
    blank_after_payload = np.zeros(int(TIME_AFTER_PAYLOAD_US * sample_rate / US_PER_SEC), np.float32)

    for allowed_message in allowed_messages:
        for params in keyhole_supplier_parameters:
            current_expected_delays_bit_times = params['expected_delays']
            current_extra_stop_bits = params['extra_stop_bits']
            current_expected_phases = params['expected_phases']

            keyhole_bits = _get_payload_bits(allowed_message, checksum=None,
                                             extra_stop_bits=current_extra_stop_bits,
                                             truncate_at_checksum=True)

            for current_expected_delay_bit_time in current_expected_delays_bit_times:
                keyhole_signal_start_us = current_expected_delay_bit_time * UART_BIT_TIME_US \
                                          + FROM_J2497_OVER_TO_UART_OVER_US \
                                          - UART_BIT_TIME_US \
```

```python
                                          - SYNC_SYMBOL_TIME_US
                keyhole_signal_start_samples = int(keyhole_signal_start_us * sample_rate / US_PER_SEC)

                # TODO: maybe overlap the jam and door signal a little bit (1/2 a body bit time probably).
                #   For now terminate the jam as soon as the door signal starts.
                early_jam = jam_amplitude * _get_jam(sample_rate, keyhole_signal_start_samples)

                for current_phase in current_expected_phases:
                    # TODO: prepare keyhole with an arbitrary mask. In testing so far all LAMP ON receivers reject 0a00
                    #   messages with an invalid CRC; therefore it is acceptable to blank the CRC and end symbol. If
                    #   receivers (ABS tractor controllers) are found that receive 0a00 messages with invalid CRC then
                    #   blanking a subset of bits of the 0a00(f6) message will be necessary. The following append of the
                    #   valid payload with silence for the CRC and end sync symbol will need to be replaced with a more
                    #   general substitution of silence for a 'mask' (a set of bits). For 0a00 the mask will need to be
                    #   of some of the logical '1' bits in the MID 0a and some in the payload as well since the silence
                    #   gaps are decoded as '1' or '0' unpredictably but usually in the same consecutively.
                    keyhole_signal = np.append(
                        keyhole_amplitude * current_phase * _get_payload_chirps(keyhole_bits, sample_rate),
                        blank_after_payload
                    )

                    keyhole_signal = np.append(early_jam, keyhole_signal)
                    yield keyhole_signal


# Any constant carrier in the range 300E3-400E3 works; however, this frequency was optimized by testing for the best
# corrupting constant carrier at 3/4 power of the target signal.
DEFAULT_JAM_FREQ = 376.379E3


def _get_jam(sample_rate, duration_samples, freq=DEFAULT_JAM_FREQ):
    """
    this is a really dumb and degenerate use of a chirp function to make a single component sinusoid >﹏<

    :param sample_rate:
    :param duration_samples: duration of the signal in samples
    :param freq: frequency of the constant carrier interference signal
    :return: a numpy array of samples valued in [-1.0, 1.0]
    """
    constant_carrier = chirp(
        np.linspace(0, duration_samples / sample_rate, duration_samples),
        f0=freq, f1=freq, t1=duration_samples / sample_rate, phi=-90, method='linear')
    return constant_carrier


SYNC_BITS = bitstring.ConstBitArray(bin='11111')
START_BITS = bitstring.ConstBitArray(bin='0')
STOP_BITS = bitstring.ConstBitArray(bin='1')
ENDSYNC_BITS = bitstring.ConstBitArray(bin='1111111')
```

```python
def _get_payload_bits(payload, checksum=None, extra_stop_bits=None, truncate_at_checksum=False):
    if extra_stop_bits is None:
        extra_stop_bits = [0, ]
    payload_bits = bitstring.BitArray()

    payload_bits.append(SYNC_BITS)
    char_count = 0
    for b_int in bytes(payload):
        b_bytes = bytes([b_int])
        b_bits = bitstring.BitArray(bytes=b_bytes)
        b_bits.reverse()

        payload_bits.append(START_BITS)  # start bit
        payload_bits.append(b_bits)  # bit-reversed byte
        payload_bits.append(STOP_BITS)  # stop bit
        extra_stop_bit = extra_stop_bits[-1] if char_count > len(extra_stop_bits) else extra_stop_bits[char_count]
        for i in range(extra_stop_bit):
            payload_bits.append(STOP_BITS)  # stop bit

    if truncate_at_checksum:
        return payload_bits

    if checksum is None:
        checksum_bits = _get_checksum_bits(payload)
    else:
        checksum_bits = bitstring.BitArray(bytes=checksum)
    checksum_bits.reverse()

    payload_bits.append(START_BITS)
    payload_bits.append(checksum_bits)
    payload_bits.append(STOP_BITS)

    payload_bits.append(ENDSYNC_BITS)

    return payload_bits


# TODO: there are definitely more efficient ways to do the J1708 checksum
def _get_checksum_bits(payload):
    msg = str(bitstring.ConstBitArray(bytes=payload).bin)
    checksum = 0
    for n in range(0, len(msg), 8):
        checksum = checksum + int(msg[n:n+8], 2)

    # Two's Complement (10)
    binint = int("{0:b}".format(checksum))              # Convert to binary (1010)
    flipped = ~binint                                   # Flip the bits (-1011)
    flipped += 1                                        # Add one_bits (two's complement method) (-1010)
    intflipped = int(str(flipped), 2)                   # Back to int (-10)
    intflipped = ((intflipped + (1 << 8)) % (1 << 8))   # Over to binary (246) <-- .uint
    intflipped = '{0:08b}'.format(intflipped)           # Format to one_bits byte (11110110) <-- same as -10.bin
```

```python

    checksum_bits = bitstring.BitArray(bin=intflipped)
    return checksum_bits


def _get_payload_chirps(j2497_payload_bits, samp_rate, local_chirp=None):
    if local_chirp is None:
        local_chirp = _generate_single_chirp(samp_rate)
    wave = np.zeros(0, np.float32)
    for n in j2497_payload_bits:
        if n:
            wave = np.append(wave, local_chirp)
        else:
            wave = np.append(wave, local_chirp * -1)
    return wave


def _generate_single_chirp(samp_rate):
    wave = np.hstack((
        np.tile(np.hstack((
                chirp(np.linspace(0,        63E-6,  int(63E-6 * samp_rate)),
                      f0=203E3, f1=400E3, t1=63E-6, phi=-90, method='linear'),
                chirp(np.linspace(63E-6,    67E-6,  int(4E-6 * samp_rate)),
                      f0=400E3, f1=100E3, t1=67E-6, phi=-90, method='linear'),
                chirp(np.linspace(67E-6,    100E-6, int(33E-6 * samp_rate)),
                      f0=100E3, f1=200E3, t1=100E-6, phi=-90, method='linear')
                )), 1)
    ))
    target_len = int(100e-6 * samp_rate)
    wave = np.append(wave, np.zeros(np.max([0, target_len - len(wave)])))
    return wave


def _generate_single_chirp_alt(samp_rate):
    wave = np.hstack((
        np.tile(np.hstack((
                chirp(np.linspace(0,        63E-6,  int(63E-6 * samp_rate)),
                      f0=203E3, f1=394E3, t1=63E-6, phi=-90, method='linear'),
                chirp(np.linspace(63E-6,    67E-6,  int(4E-6 * samp_rate)),
                      f0=400E3, f1=100E3, t1=67E-6, phi=-90, method='linear'),
                chirp(np.linspace(67E-6,    100E-6, int(33E-6 * samp_rate)),
                      f0=1E3,   f1=216E3, t1=100E-6, phi=-30, method='linear')
                )), 1)
    ))
    target_len = int(100e-6 * samp_rate)
    wave = np.append(wave, np.zeros(np.max([0, target_len - len(wave)])))
    return wave

```

Listing 1 Keyhole Mitigation source code

PROT9 flooding with jamming signal

As was introduced in PROT8, above, there is a signal alluded to in the J2497 specification where it says “The PLC for trucks technology is more sensitive to constant carrier interference than to broadband interference […].” This interference effectively blocks reception of J2497 signals when present at comparable amplitude. By ‘comparable’ we mean near the same amplitude as the signal that is targeted to be blocked. In our testing we observed intermittent blocking at -2dB but at 0dB (equal amplitude) or greater the blocking is very reliable.

This protection involves transmitting this signal on the powerline segment at an amplitude comparable to the target signals to be blocked. Recall there are three types of attacks, at different expected power levels. e.g. transmitting the jamming signal to stop the highest amplitude, Malware-initiated, well-formed, attacks would also block the other two lower expected amplitude attacks but even transmitting with a basic 5V GPIO bitbang method would block both the other types of attack.

This protection alone is not likely to be sufficient for application to defending fleets because, like PROT6 chirp filter inline, this protection alone will block reception of all traffic including the LAMP messages required for the only industry standard way to satisfy regulations in North America; however, it can be combined with other techniques to produce what could be viable solutions for fleets.

Solutions

In this section we present combinations of some of the techniques above into solutions with merit for mitigating the risks to fleets posed by the J2497 attacks. Although there are 3 types of J2497 attack, both of the malware-initiated types are less likely since they first require remote code execution. The RF induced type is practical and of some concern to the industry; therefore, solutions addressing the RF induced J2497 attack will be considered here.

The vulnerable technology, J2497, has been fielded since 2001 and the service lifetime of trailers is 15 years in their first life and another 15 in the second-hand market; therefore, bolt-on solutions for fielded tractors and trailers should be the focus of development and testing. For new equipment, the industry should be dropping all J2497 features entirely except for backwards compatibility with LAMP ON detection <u>only</u>. For trailer equipment this means migrating all diagnostics to whatever newer trailer buses are established as the norm. For tractor equipment this means removing support for reception of any J2497 message other than LAMP messages <u>and also</u> protecting the backwards compatible trailers from attack; we propose such a solution below in SOLNF. All the other solutions are applicable to retrofit on existing equipment.

Trucking is a small-business industry where >90% of the fleets are operating less than 6 trucks and operators often don’t own or otherwise control the trailer equipment they haul (in North America) and trailers generally outnumber tractors; therefore, solutions which can be installed on the tractor should be prioritized.

	Technique Combinations	Solution Pros	Solution Cons
SOLNA LAMP ON firewall	PROT6 chirp filter inline plus: MCU with dual J2497 interfaces	• Can be configured to allow fleet-specific uses of J2497 • A variation of this can protect tractor ABS also[1]	• Must be installed on each ECU, tractor and trailer
SOLNB LAMP detect circuit LAMP ON sender	PROT6 chirp filter inline plus: MCU that sends LAMP ON when LAMP circuit is asserted	• This is likely a technology already developed for purchase to retrofit pre-J2497 equipment that had only trailer fault LAMPs	• Must be installed on each trailer ECU • Won't protect tractor ABS from attack e.g. 'roll-call'
SOLNC trailer address denier	Just PROT7 continuous dynamic address claimer	• Simple blind-transmit defense (could bitbang it) • Possible against all types of J2497 attacks (but not 100% see cons) • Can be installed on tractor	• Unproven • RFI noise • May allow intermittent unicast attacks • Doesn't protect tractor controllers • Might not prevent as-yet unknown exploit payloads and abuse commands that don't require unicast J1708
SOLND just RF chassis chokes	Just PROT5 RF chassis chokes	• Passive components, relatively cheap and easy to install.	• Unproven, but may work based on our understanding of RF Induced attacks • Would only protect against RF induced attack • Must be installed on each trailer
SOLNE LAMP keyhole	Just PROT8 LAMP keyhole signal	• Simple blind-transmit defense (could bitbang it) • Should prevent exploit payloads and abuse commands • Possible against all types of J2497 attacks (but not 100% see cons)	• Unproven, but initially confirmed on lab bench • Won't prevent LAMP ON attacks • RFI noise • Can delay trailer ABS fault telltale by tens of seconds for Haldex trailer brake controllers

		• Asymmetrically impacts high data rate signals more than low-rate LAMP • Can be installed on tractor	
SOLNF (for new equipment) jamming signal and coherent removal of it	PROT9 flooding with jamming signal plus coherent removal of that jamming signal at the tractor ECU	• Should prevent exploit payloads and abuse commands • Possible against all <u>types</u> of J2497 attacks (but not 100% see cons) • Enables new tractor brake controllers to protect backwards-compatible J2497 trailers and still receive required LAMP messages	• Won't prevent LAMP ON attacks • RFI noise • Possible for new tractor brake ECUs only • Could yield an unprotected tractor brake controller if legacy J2497 commands aren't also removed.
SOLNG (for retrofit) jamming signal and coherent removal of it	PROT9 flooding with jamming signal plus coherent removal of that jamming signal in dedicated J2497 receiver device for LAMP etc.	• Should prevent exploit payloads and abuse commands • Possible against all <u>types</u> of J2497 attacks (but not 100% see cons) • Can be installed on a tractor • Capable of custom reception of J2497 traffic	• Won't prevent LAMP ON attacks • RFI noise • Possible for trucks with accessible trailer fault telltale wiring or using the PGN 61441 for the telltale

[1]Each tractor ABS controller has its own supplier-specific cable and as such it is not practical to produce a bolt-on mitigation for attacks on tractor ABS controllers. The suppliers need to release software updates for the tractor controllers or adapt one or more of the above to be bolt-on protections for their tractor controllers (e.g. SOLNA).

SOLNA LAMP ON firewall

A conventional firewall approach to J2497 can be realized by combining the defense of chirp filters to isolate the J2497 device and using a MCU with two separate J2497 interfaces to receive and selectively forward J2497 messages bi-directionally (depicted in Figure 14 below). This would permit fleets that are using J2497 features other than the LAMP messages to continue use of those features while also denying any other messages which could contain exploit payloads or abuse commands. The usual firewall caveats of all security benefits being subject to correct configuration apply to this system. Care must be taken by fleets to not add abuse commands (such as solenoid test) to the firewall passlist; furthermore, firewall designers must take care to prevent malicious reconfiguration and bypass of the firewall.

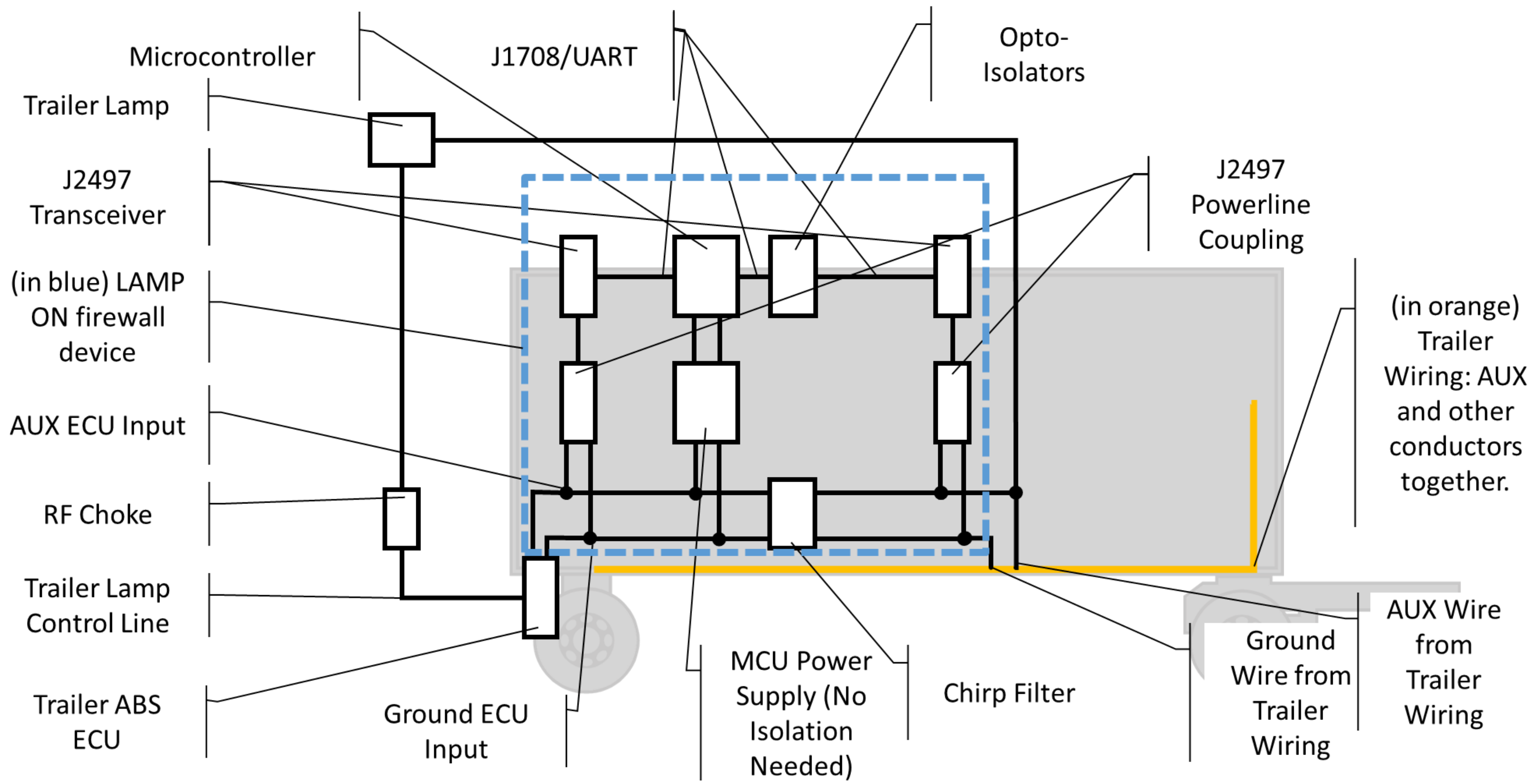

Figure 14 SOLNA LAMP ON firewall

SOLNB LAMP detect circuit LAMP ON sender

All possible messages receivable by a trailer ECU can be denied and still permit sending the required LAMP messages without the risks of incorrect or malicious reconfiguration of firewall rules as in SOLNA. All trailer ABS controllers have a fault lamp control pin which is driven on fault conditions that match the sending of LAMP messages. A device could be wired to the trailer lamp or the lamp control line which sends J2497 LAMP messages is response to lamp control line state changes. This is shown in Figure 15 below. When combined with a chirp filter than denies both messages from and to the trailer ABS controller the result is a J2497 network segment that can still communicate trailer ABS faults to the tractor but which cannot response to any potential exploit payloads or abuse commands. Note that since the device which sends LAMP messages has no receiver requirements it could be built using GPIO toggling / bit-banging to send J2497 instead of using the more expensive J2497 transceivers.

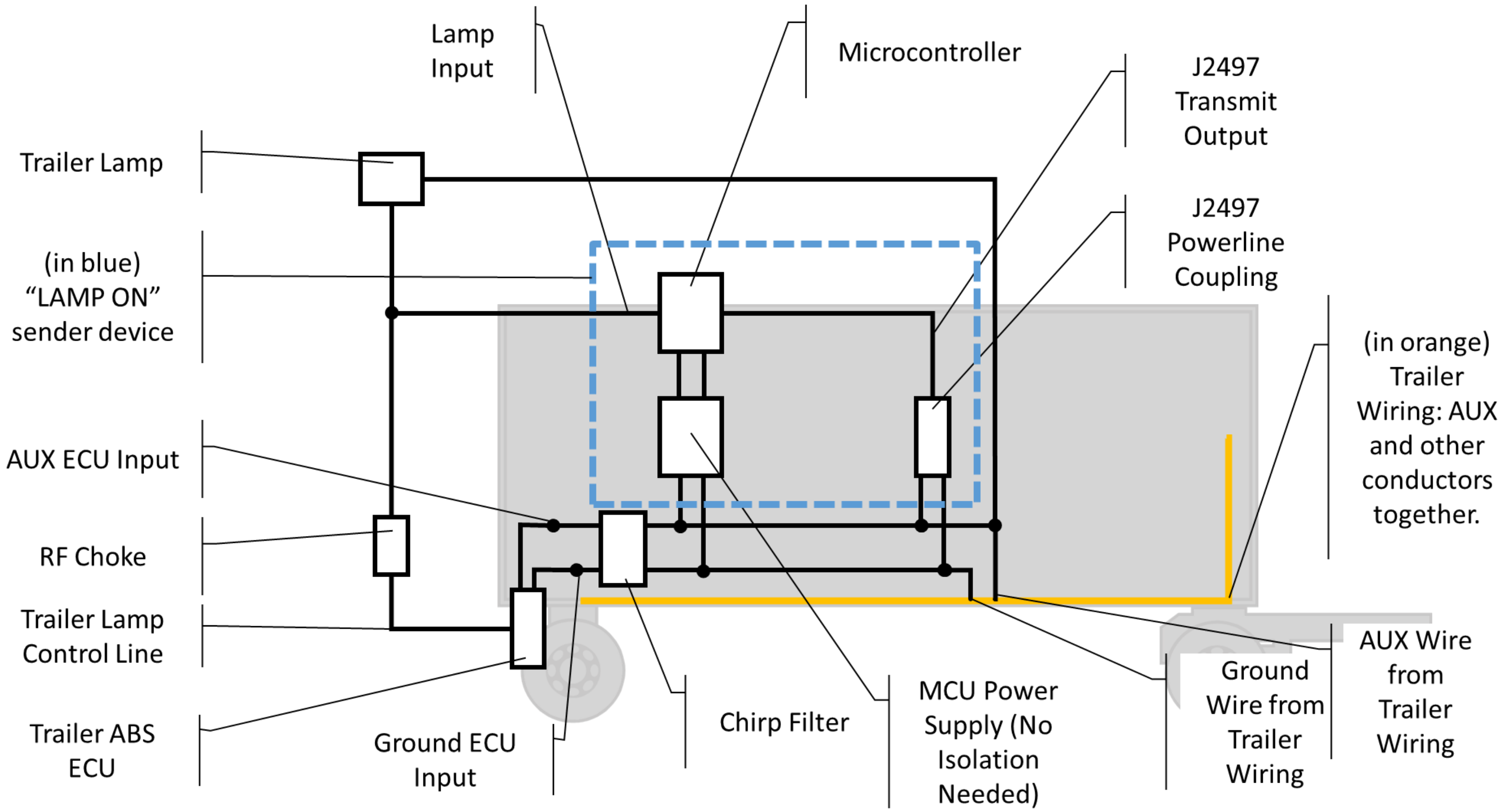

Figure 15 SOLNB LAMP detect circuit LAMP ON sender

It would furthermore be possible to package the J2497 LAMP message sender into a trailer lamp product, reducing parts count, a ‘LAMP ON’ lamp (depicted in Figure 16 below).

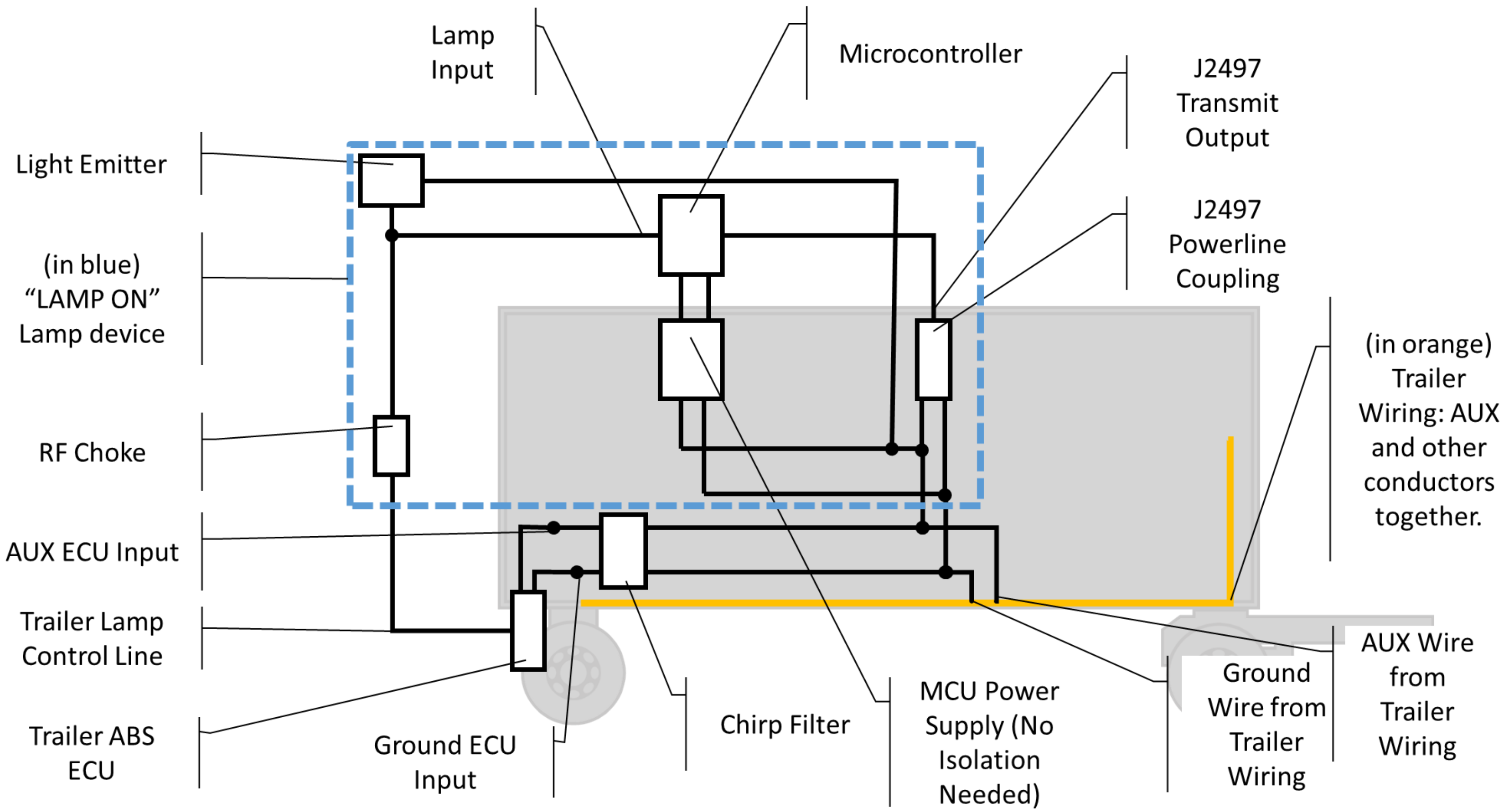

Figure 16 SOLNB LAMP detect circuit "LAMP ON" Lamp Device

SOLNC trailer address denier

This proposed solution involves only deploying the PROT7 continuous dynamic address claim technique to deny any unicast address trailer targets to attackers. The PROT7 technique has no receiver requirements it could be built using GPIO toggling / bit-banging to send J2497 instead of using the more expensive J2497 transceivers.

This defense works by relying on trailer equipment changing their ‘addresses’ (MIDs) in response to receiving a J2497 message with an MID matching their own at the time. It could only possibly defend against attacks that use ‘unicast’ messages e.g. Data Link Escape (DLE) messages. It could not defend from other as-yet unidentified attacks that do not use unicast e.g. any PID-based messages.

This defense does not require that the transmission be of equal or greater power than the attacker signal, only that it is received. In testing bitbanged J2497 signals it has been observed that those from 5V GPIOs can still be received at the last ABS controller in a triple road train; therefore, for most deployments no amplification is needed, only coupling (e.g. a capacitor) is needed to the power line segment.

This solution could be retrofitted on existing tractors (depicted in Figure 17 below) where it would only defend against attack as described above but still permit all other J2497 traffic to be received by the tractor and trailer equipment. On some tractors the power pins at the diagnostic and/or RP1226 connector are unfiltered from the those of the J560 wiring and the device described here could even be installed at those locations.

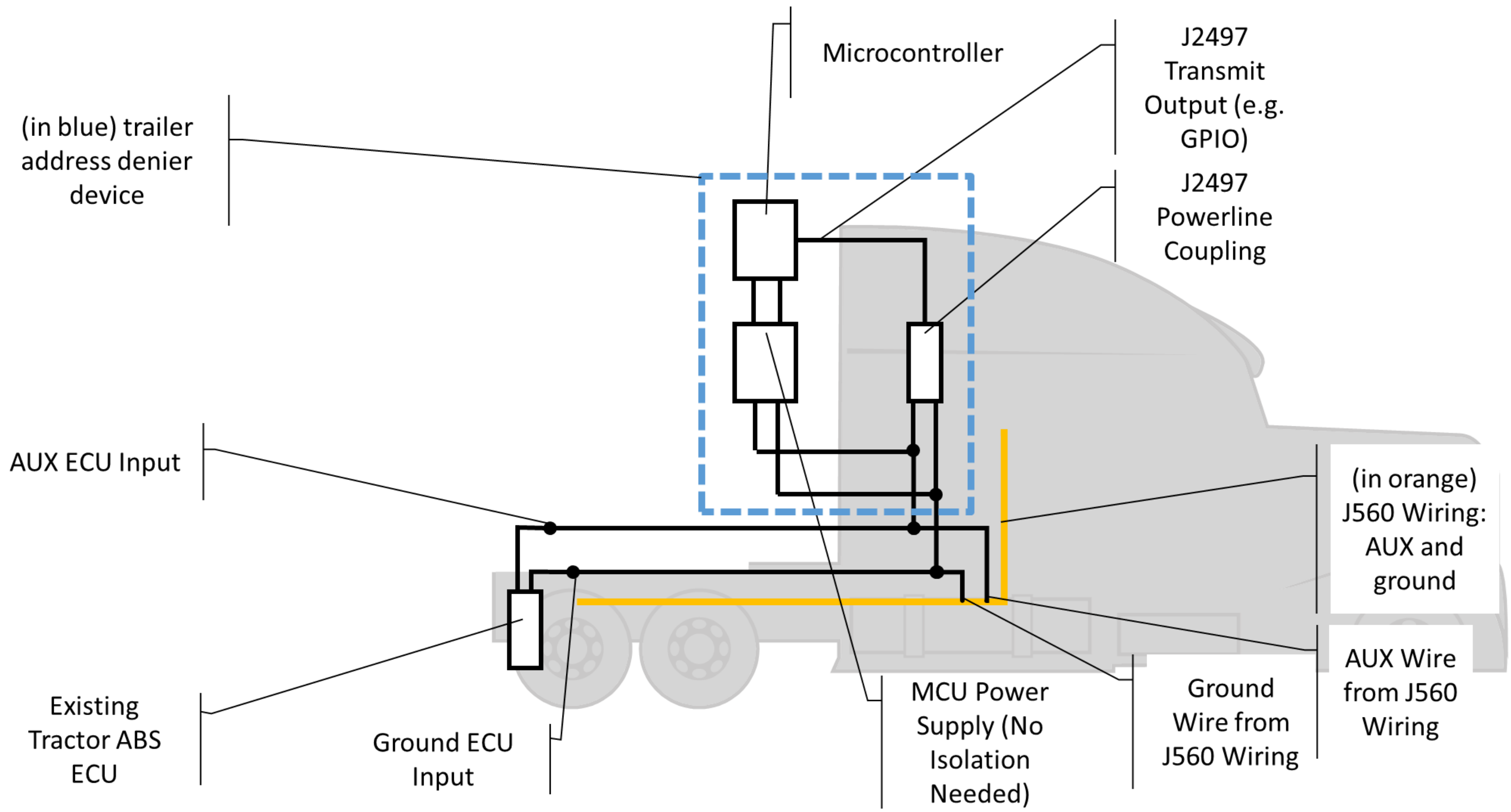

Figure 17 SOLNC trailer address denier

SOLND just RF chassis chokes

This proposed solution is to deploy only the PROT5 RF chassis chokes. No requirements for send or receive of J2497. Could be retrofitted onto some trailers if they have a single chassis ground connection to trailer wiring and/or can be modified to have a single ground connection.

SOLNE LAMP keyhole

This proposed solution involves only deploying PROT8 LAMP keyhole signal to deny all signals except LAMP on the powerline segment. The PROT8 technique has no receiver requirements it could be built using GPIO toggling / bit-banging to send J2497 instead of using the more expensive J2497 transceivers.

This solution could be retrofitted on existing tractors (depicted in Figure 18 below) where it would protect both the tractor and all connected trailer ABS ECUs from receiving attacker J2497 signals but still permit the tractor ECU to receive the necessary LAMP messages. On some tractors the power pins at the diagnostic and/or RP1226 connector are unfiltered from the those of the J560 wiring and the device described here – as was also the case with SOLNC above – could even be installed at those locations.

This solution works by deploying a transmitter of the PROT8 LAMP keyhole signal, which is a blind-transmit solution; however, as described in the PROT8 section above, to function reliably the defensive signal needs to be of an amplitude greater than or equal to that of the attacker signal. Thus, to defend against malware-initiated & well-formed attacks the transmitter needs amplification to the same magnitude as that of the strongest trailer or tractor equipment fielded. Furthermore, since it was observed in testing on triple trailers that sometimes RF-induced signals were received more reliably than those generated by diagnostic adapters, it stands to reason that to defend against even just the lowest powered attack of the three: RF-induced attacks, some amplification will be needed to successfully defend a triple trailer configuration.

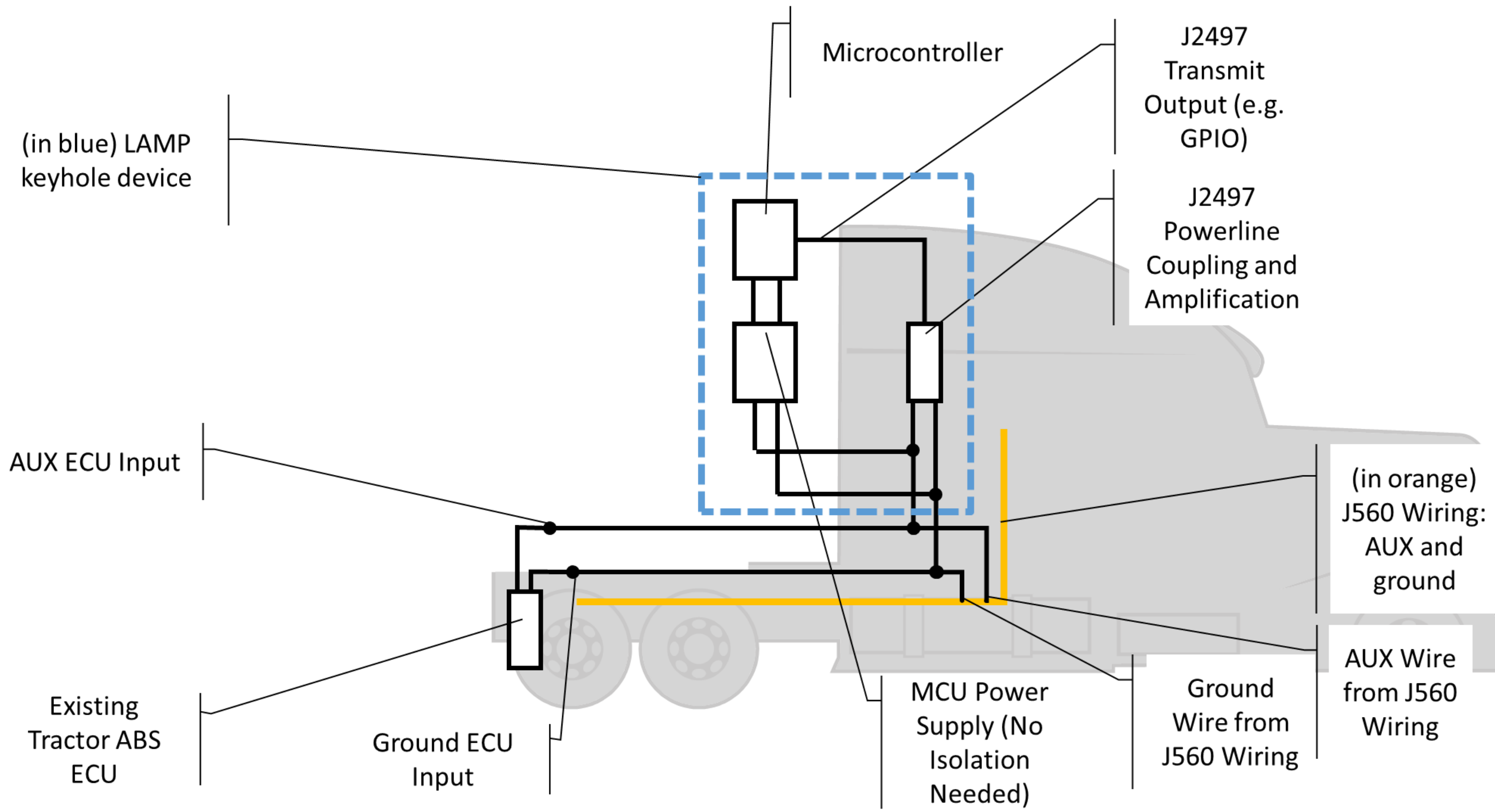

Figure 18 SOLNE LAMP keyhole

SOLNF (for new equipment) jamming signal and coherent removal of it

The constant-carrier interference signal introduced and applied in the PROT8 Lamp keyhole section, called the 'jamming signal' has useful properties which can be applied to create another possible solution for defending against J2497 attacks. Because the jamming signal corrupts reception of signals of comparable amplitude, does not trigger idle end detection in transmitters, and does not pre-empt transmission of messages it can be used to mount a defense against reception of any J2497 messages: transmit the jamming signal continuously. A continuous transmission of the jamming signal would also, however, block reception of the LAMP messages which are required. Because the powerline signals are superimposed it is possible to remove the jamming signal by signal subtraction; the device responsible for transmitting the jamming signal is well-suited to perform this subtraction since it already has the precise signal and can thus remove it in a coherent manner, leaving only the other transmitters' J2497 signals. With the jamming signal removed the resulting signal can be fed to any J2497 receiver. All the other J2497 receivers on the powerline segment will have their reception corrupted but the device coherently removing the jamming signal will be capable of correctly receiving all the messages.

This defense can be integrated into new tractor brake ECUs and would protect all the connected (legacy) trailer ABS ECUs from receiving attacker J2497 signals but still permit the new tractor brake ECU to receive necessary (for backwards compatibility) LAMP J2497 messages. To also achieve protection of the (new) tractor brake controller, these controllers will need to have all J2497 removed except for LAMP ON processing.

SOLNG (for retrofit) jamming signal and coherent removal of it

The same defense (as SOLNF) is possible in a form that can be retrofitted onto existing tractors in a separate device transmitting the jamming signal. Since it is a separate device, reception of J2497 is also corrupted at the tractor ECU. This has the benefit of also blocking attacks on the tractor ECU via J2497 but, of course, the required LAMP messages are also blocked. In the case where the tractor's trailer ABS fault instrument cluster telltale both responds to a known J1939 message (e.g. the J1939 standard PGN 61441) and the instrument cluster J1939 segment is accessible for retrofit then the limitation of corrupting reception of the required LAMP messages can be overcome. The device transmitting and coherently removing the jamming signal can response to the reception of LAMP messages with the appropriate J1939 signal. In this manner all of the ECUs of the tractor and trailer are protected from J2497 attacks but also the required LAMP telltale can still function.

For the same reasons detailed in SOLNE above some amplification will be necessary to reliably block reception of attacker signals.

This solution could be retrofitted on existing tractors (depicted in Figure 19 below) where it would protect both the tractor and all connected trailer ABS ECUs from receiving attacker J2497 signals but still permit the driver to observe the necessary trailer ABS fault telltale. The dedicated receiver could also be customized to receive and react to arbitrary J2497; however, special care should be taken since those messages could be attacker induced, even RF-induced. Reacting to only the regulatory required LAMP messages is by far the safest solution. On some tractors the power pins at the diagnostic and/or RP1226 connector are unfiltered from the those of the J560 wiring and the device described here – as was also the case with SOLNC above – could even be installed at those locations.

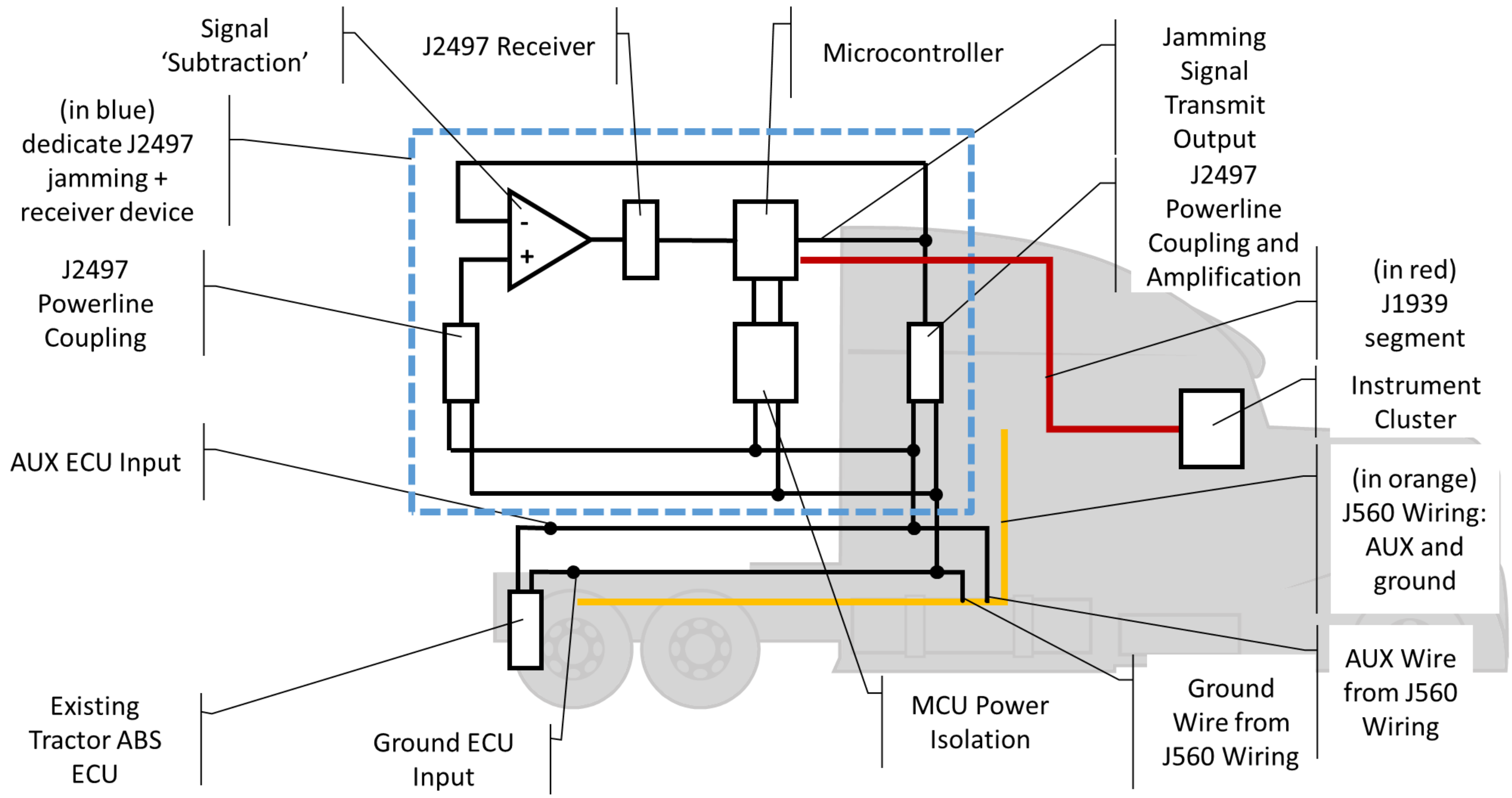

Figure 19 SOLNG (for retrofit) jamming signal and coherent removal of it