HIGH PERFORMANCE
HIGH RELIABILITY
HIGH SECURITY

Fiber SenSys®

Setting the Tuning Parameters

Application Note

FSI
2925 NW Aloclek Dr.
Suite 120
Hillsboro, OR 97124
USA

Tel: 1-503-692-4430
Fax: 1-503-692-4410
info@fibersensys.com
www.fibersensys.com

Contents

Introduction

Whenever calibrating the Alarm Processing Unit (APU), there are two crucial objectives; to detect potential intruders, and to avoid nuisance alarms that, over time, might lower confidence in the security system. To achieve these two objectives, **Fiber SenSys** has developed sophisticated digital signal processing (DSP) algorithms that make it easy for you to "tune" the APU for almost any situation.

This application note provides detailed information about the tuning parameters available for calibrating **Fiber SenSys** APUs, as well as suggestions for how to adjust those parameters in different situations. Please note, however, that some APUs may not support all the tuning parameters described in this application note. See your user's manual for a description of the tuning parameters supported by your instrument.

Calibration of the APU should come after installation of the sensing hardware, and before the system is commissioned. Some APUs have just one zone, most have two zones, and others have eight or more zones. Whichever APU you are using, you should tune and test each zone separately because the environment surrounding each zone is unique. Take, for example, two zones on a chain link fence. One zone might be on a section of fence that is aligned broadside to the wind, while the other zone might be in a "wind shadow." In this example, you may want to adjust the wind compensation algorithm higher on the first zone than on the second. Similar scenarios exist for applications in data security where one zone might pass near an elevator, where it is subject to building vibrations, while the other zone might be located in the basement.

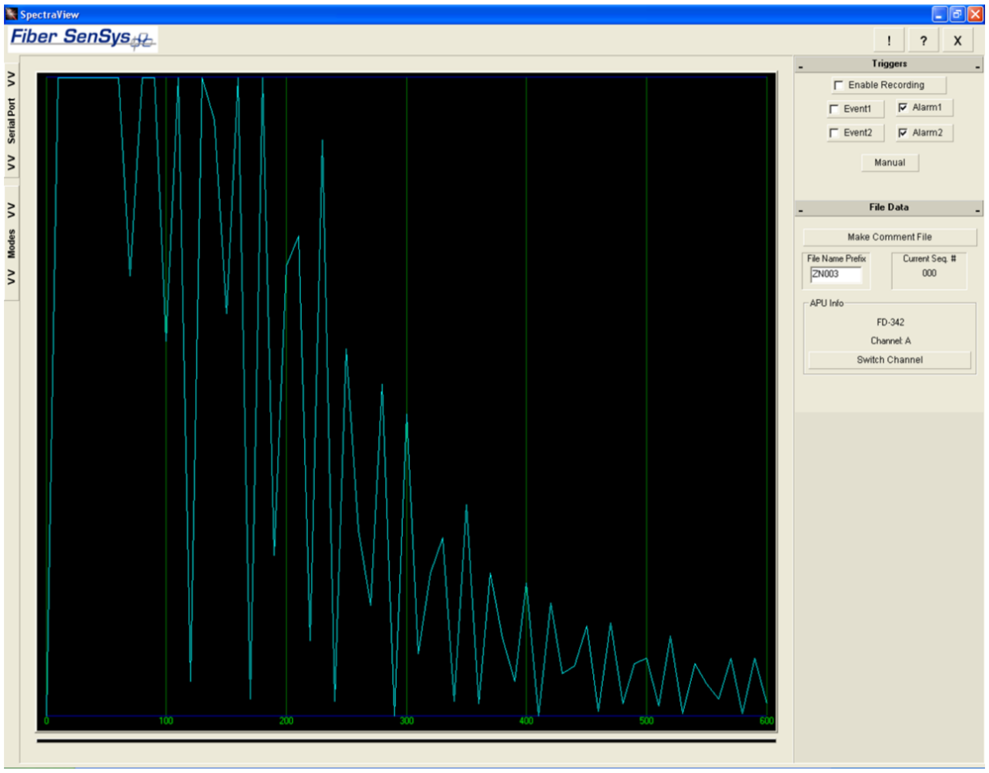
There are many options for tools that can be used to calibrate or tune the APUs; almost any computer can be used, as well as Head Ends. In addition, **Fiber SenSys** offers ruggedized hand-held computers for outdoor work with some APUs.

General comments about tuning

There are two principal tools that **Fiber SenSys** has developed, for tuning an APU to detect intruders and avoid nuisance alarms. The first of these tools is a real-time spectral display of the acoustic energy being detected by the sensor (see figure 1)[1], and the second tool is a parameter editor that allows you to adjust the various properties that have been built into **Fiber SenSys**' sophisticated intruder-detecting algorithms (see figure 2)[2].

[1] The screen shot in figure 1 is from SpectraView. Depending on your APU, the view that you see may have slight differences from figure 1.

[2] Figure 2 shows how the parameters appear in a fully functional version of SpectraView. Some APUs (the 500 series, for example) use a different parameter editor, which appears different, and has some different tuning parameters. For the most part, however, the tuning parameters are all the same, although there are some variations, between APU models, with regard to the terminology that's used.

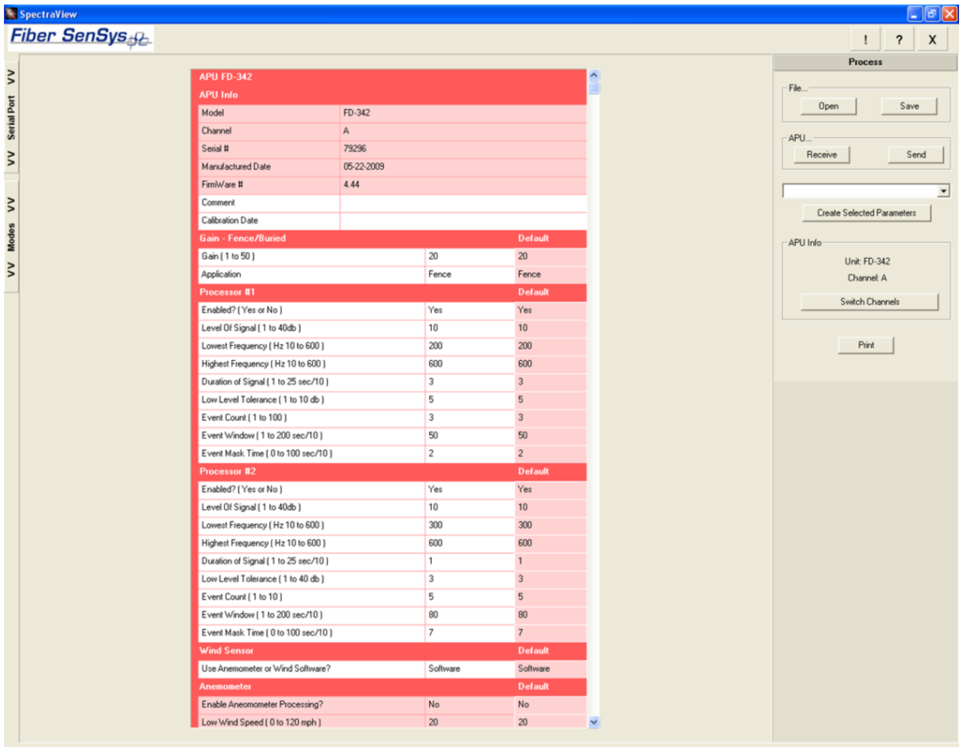


☐ Enable Recording
☐ Event1 ☑ Alarm1
☐ Event2 ☑ Alarm2

File Name Prefix: ZN003
Current Seq. #: 000

APU Info
FD-342
Channel: A

Figure 1. Real-time view of the acoustic energy being sensed by an intruder. Frequency is plotted along the horizontal axis, from 0 to 600 Hz. The vertical axis corresponds to linear amplitude.



APU Info
Unit: FD-342
Channel: A

APU FD-342		
APU Info		
Model	FD-342	
Channel	A	
Serial #	79296	
Manufactured Date	05-22-2009	
FirmWare #	4.44	
Comment		
Calibration Date		
Gain - Fence/Buried		**Default**
Gain (1 to 50)	20	20
Application	Fence	Fence
Processor #1		**Default**
Enabled? (Yes or No)	Yes	Yes
Level Of Signal (1 to 40db)	10	10
Lowest Frequency (Hz 10 to 600)	200	200
Highest Frequency (Hz 10 to 600)	600	600
Duration of Signal (1 to 25 sec/10)	3	3
Low Level Tolerance (1 to 10 db)	5	5
Event Count (1 to 100)	3	3
Event Window (1 to 200 sec/10)	50	50
Event Mask Time (0 to 100 sec/10)	2	2
Processor #2		**Default**
Enabled? (Yes or No)	Yes	Yes
Level Of Signal (1 to 40db)	10	10
Lowest Frequency (Hz 10 to 600)	300	300
Highest Frequency (Hz 10 to 600)	600	600
Duration of Signal (1 to 25 sec/10)	1	1
Low Level Tolerance (1 to 40 db)	3	3
Event Count (1 to 10)	5	5
Event Window (1 to 200 sec/10)	80	80
Event Mask Time (0 to 100 sec/10)	7	7
Wind Sensor		**Default**
Use Anemometer or Wind Software?	Software	Software
Anemometer		**Default**
Enable Aneomometer Processing?	No	No
Low Wind Speed (0 to 120 mph)	20	20

Figure 2. Parameter editor in SpectraView.

The tuning parameters (which we'll discuss in greater detail later in this application note) allow you to calibrate the APU so that it has maximum probability of detecting real intrusion attempts, and minimum probability of generating nuisance alarms. Not all these parameters are available with all APUs – check with your user's manual to find out which parameters your APU supports. Mostly, the names for these parameters are common across **Fiber SenSys**' 300- 500-series APUs, though their location in the user's interface may vary between different brands of APUs (where differences in terminology exist, this application note will point them out).

Within the detection algorithms are two virtual processors. In APUs sold for perimeter security, processor 1 has default parameters for detecting intruders that are climbing a fence, while processor 2 is designed to catch intruders that are cutting the fence. In APUs designed for data security, processor 1 has default parameters for catching intruders that are cutting the conduit, and processor 2 has default parameters for detecting intruders that are physically handling the fiber. Either processor can be deactivated if you want to tune the APU to detect a single type of intrusion.

The tuning process involves two people, one person to monitor the spectral display in **RealTime** (see figure 1), and the other to simulate intrusions. The type of simulated intrusion depends on the application. For fence perimeters the simulations include climbing and cutting the fence. For data protection applications, the simulations include cutting the conduit, and physically handling the fibers. The following example illustrates how the parameters would be tuned for a fenced perimeter, using the parameter editor in SpectraView.

1. Open the **RealTime** tab and have your assistant simulate an intruder who is cutting the fence. The assistant can do this by tapping on the fence with a medium-sized screwdriver. Observe the signal amplitude and adjust the **Sensitivity** parameter (found in the **APU Parameters** tab) until the signals from the simulated fence cuts reach about 80% of the way to the top of the display.[3]

2. Next, while observing the sensor signal in **RealTime**, have an assistant simulate an intruder who is climbing the fence. As before, adjust **Sensitivity** to ensure that signal is at about 80% during the simulated intrusion.[4]

Once **Sensitivity** has been adjusted, the tuning process continues with the selection of the low- and high-frequency cutoff settings. The low- and high-frequency cutoff parameters define which acoustical frequencies the APU processes when looking for intruders; all frequencies below the low-frequency cutoff, and all frequencies above the high-frequency cutoff are ignored.

[3] With too little **Sensitivity** it will be difficult to see changes, and with too much there is a risk of saturating the processor

[4] It may not always be possible to have both cut and climb signals reach optimum amplitude, as one may be too low while the other is too high. In this situation it's only necessary to ensure that neither signal exceeds the viewable area of the **RealTime** screen during a simulated intrusion.

The basic idea behind the low- and high-frequency cutoffs is that things which cause nuisance alarms generally have different acoustic signatures than real intruders. Thus it's possible to adjust the frequency cutoffs so that the acoustic frequencies generated by nuisances are ignored, but not the acoustic frequencies from real intrusions.

Figure 3 illustrates how this process might work in a simple situation. In the figure, you can see the acoustic signature caused by a stealthy intruder. This acoustic signature consists mostly of low-frequency energy, below 350 Hz, but with some acoustic energy all the way up to 600 Hz.

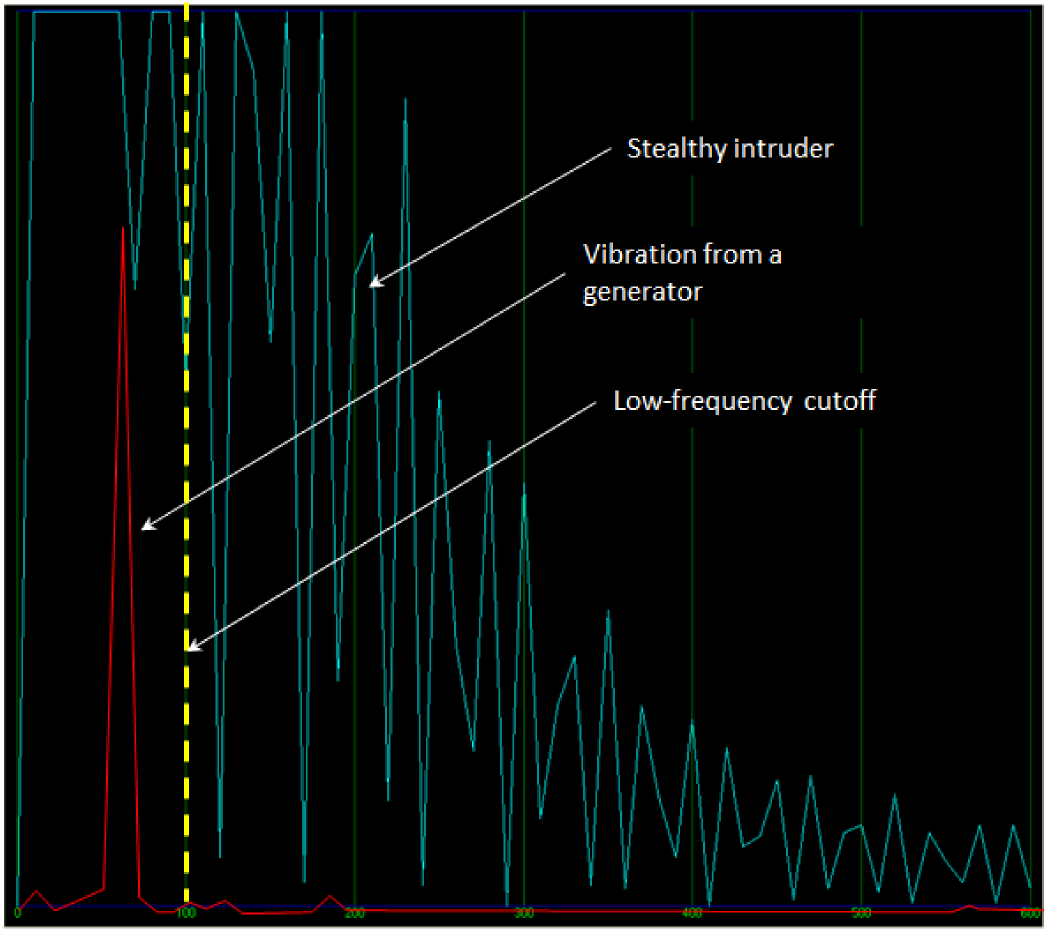


Figure 3. A simple example showing how to use the low-frequency cutoff so as to eliminate nuisance alarms from a nearby generator.

In this example, the nuisance source was a nearby vibrating generator. Notice that the noise from the generator was almost exclusively at 60 Hz, with virtually no acoustic energy above 100 Hz. Thus, in this example, the low-frequency cutoff was set to 100 Hz. By setting the low-frequency cutoff to 100 Hz, essentially all the noise from the vibrating generator is eliminated. While it's true that some of the acoustic energy from the stealthy climber is also eliminated, there is still plenty of acoustic energy from the intruder, above 100 Hz, to generate an alarm.

As the previous example showed, proper setting of the low- and high-frequency cutoff parameters requires some knowledge about the **RealTime** signal caused by a nuisance. This means that, during the tuning process, you should try to simulate expected nuisances as well as expected intrusions.

Once the **Sensitivity**, **LOW-FREQUENCY**, and **HIGH-FREQUENCY** have been set, there are several additional tuning parameters that provide further tuning flexibility. Often, the default settings for these parameters are sufficient, and no further tuning is required. But sometimes you may want to make adjustments. The following sections provide some additional information about all the tuning parameters, as a way of helping you understand what effects will result from modifying them.

Overview of tuning parameters

Depending on the APU, up to twelve tuning parameters are available, most of which are independently configurable in the two alarm processors (see figure 4). The twelve tuning parameters are:

✓ ***Signal pre-conditioning***	✓ ***Event processing***	✓ ***Alarm processing***
1. Gain 2. Sensitivity 3. Prefilter 4. Wind rejection 5. Low-frequency reject 6. High-frequency reject	7. Level (also called Signal Level or Signal, in some APUs) 8. Duration 9. Low-level tolerance	10. Event count 11. Event window 12. Event mask time

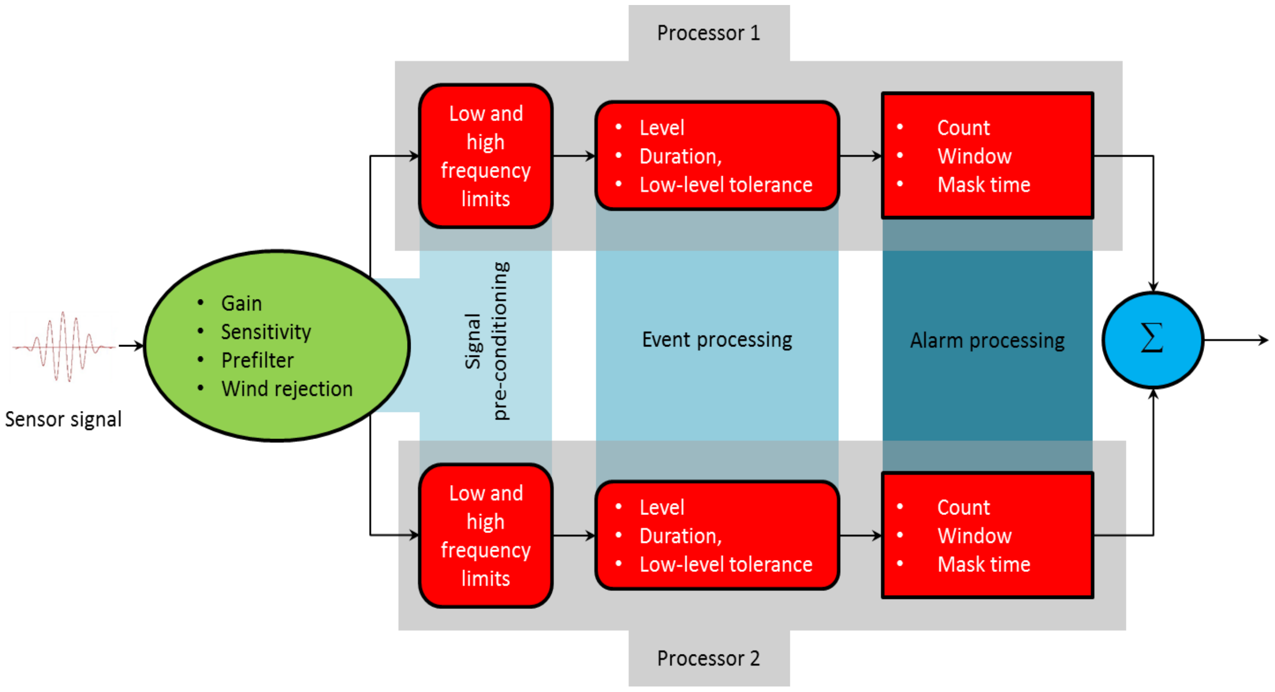


Figure 4. Signal processing algorithm, showing the processors and tuning parameters available on each zone in the FD525 (other APUs may have a subset of these tuning parameters).

The table below gives a quick summary of the different tuning parameters, and the sections in the rest of this application note give more detailed information.

Parameter	Quick description
Enabled	Enables one or both of the processors used with a zone. Disabling both processors turns the zone off.
Application	Enables or disables a special hardware-implemented low-frequency filter that is used for buried applications in APUs designed for perimeter security.
Gain	Defines signal amplitude, in dB. Does not affect the signal that is visible in Real Time.
Sensitivity	Similar to **Gain**, but operates on the signal that is visible in Real Time.
Prefilter	Digital pre-filter for signal conditioning.
Enable Wind Reject	Enables the wind rejection software.
Wind Rejection	Parameter that determines how much the APU will reduce **Gain** during windy conditions, to avoid nuisance alarms.
Lowest Frequency	APU ignores all signal content lower than this frequency.
Highest Frequency	APU ignores all signal content higher than this frequency.
Level	Sets a threshold above that must be exceeded before an event is generated by the processor. Also called **Signal Level** or **Signal** in some APUs.
Duration	Time interval during which the signal must stay above **Level** in order to qualify as an event.
Low Level Tolerance	Threshold for detecting long-lasting low-level signals that are not higher than **Level**.
Event Count	Number of events required to generate an alarm.
Event Window	Time threshold for grouping events, for the purpose of defining an alarm.
Event Mask Time	Time threshold for ignoring events, for the purpose of defining an alarm.
Anemometer	Incorporates anemometer data in wind algorithm.
HyperZone	A "Macro," or a group of zones that all share a common parameter set.

Enabled

Each zone has two processors that can be independently tuned to detect a particular type of intrusion. By default, both processors are enabled, but you have the option of disabling one or both of the processors. *Please note, however, that disabling both processors will deactivate the zone.*

Application

Some of FSI's perimeter-protection APUs can be configured for either fence or buried applications. The default configuration is fence, but if the APU is to be used with buried sensors, and the APU supports buried applications, you can select the buried application with this parameter.

Signal pre-conditioning

Signal pre-conditioning involves amplification and frequency filtering that shapes and scales the sensor signal before subjecting it to the event and alarm algorithms.

Gain

Gain raises and lowers the overall signal level, in units of dB. Raising the gain increases the signal level, making it easier for the signal to exceed the **Level** threshold that defines events. Thus, increasing **Gain** increases the overall system sensitivity. Note, however, that when increasing **Gain** you will *not* see a signal increase in the **RealTime** mode. **Sensitivity** works much the same as **Gain**, but increases the signal amplitude linearly, rather than in dB. Unlike the **Gain** adjustment, when you increase **Sensitivity** you *will* see the signal increase in **RealTime**.

For 500-series APUs, the **Sensitivity** and **Gain** settings are applied to both processors of each zone within a hyperzone. For the 300-series APUs, the **Sensitivity** and **Gain** settings are the same for both processors for a given channel/zone.

Because **Gain** boosts the sensor signal, a higher **Gain** value increases the likelihood of detecting an intruder, but also increases the likelihood of nuisance alarms. After adjusting **Sensitivity**[5], set the **Gain** value to the lowest possible level sufficient to detect an intruder, but no higher, in order to minimize nuisance alarms.

To adjust the **Gain**, follow these steps:

1. Set **Sensitivity** as described in the section on **General comments about tuning.**
2. Start with the default **Gain** setting. Go to the zone that you wish to calibrate, and simulate the sort of intrusion that you want to detect (climbing or cutting[6] the fence, stepping into the security zone, etc.).
3. Check to see if the APU generates an alarm during the simulated intrusion.
4. If the APU detects the intrusion, lower the **Gain**
5. If the APU fails to detect the intrusion, raise the **Gain**
6. Repeat steps 1-4 until you have the **Gain** set about 2 dB higher than the level that is just barely adequate for detecting the intrusion.

[5] See the section on **General comments about tuning.**

[6] You can simulate fence cutting by attaching a spare section of chain link to the fence with cable ties, and then cutting them.

Sensitivity

Sensitivity, like **Gain**, increases the overall signal level. But unlike **Gain**, when you increase the **Sensitivity** you can see the resultant signal increase when viewing in **RealTime**. As with **Gain**, a higher **Sensitivity** setting increases overall sensitivity, which means it also increases the signal from nuisance sources, making it more likely that nuisance signals will generate alarms. For more detailed information on setting **Sensitivity**, refer to the section, **General comments about tuning**.

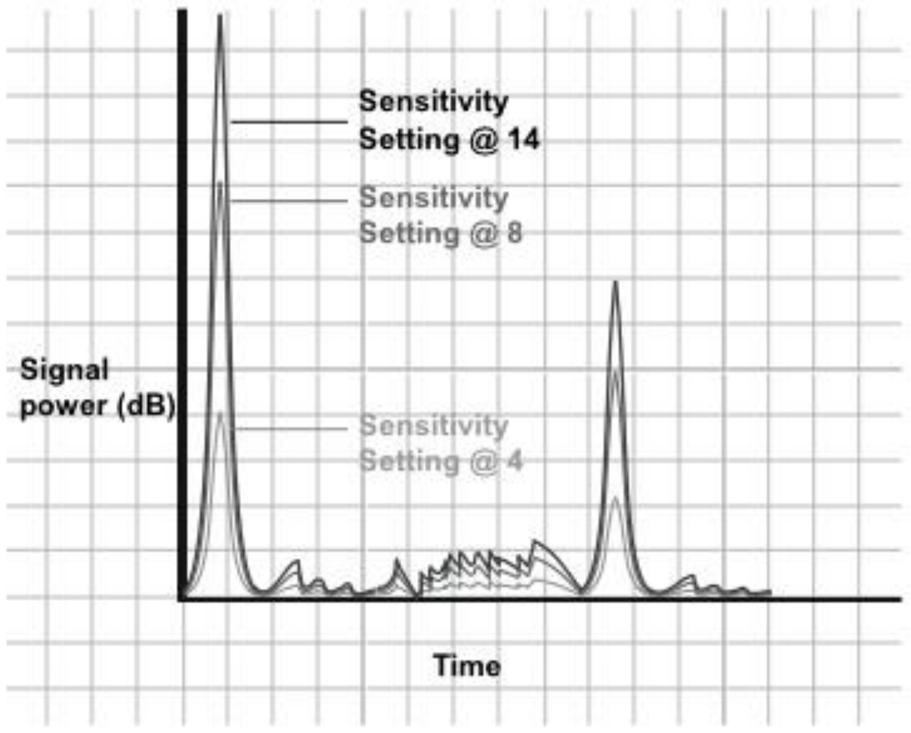


Figure 5. Unlike Gain, when you adjust the Sensitivity parameter you will see the signal increase in Real Time.

Prefilter

Prefilter controls a high-pass filter that reduces the low-frequency amplitude content of the sensor signal (see figure 6). A value of zero means that no filter is applied, while a value of 100 corresponds to maximum filtering, which removes most of the signal below 50-Hz. **Prefilter** is used to remove low-frequency signals that are often the source of nuisance alarms. For fence installations the default **Prefilter** setting is 80, while for buried installations a lower **Prefilter** value should be used.

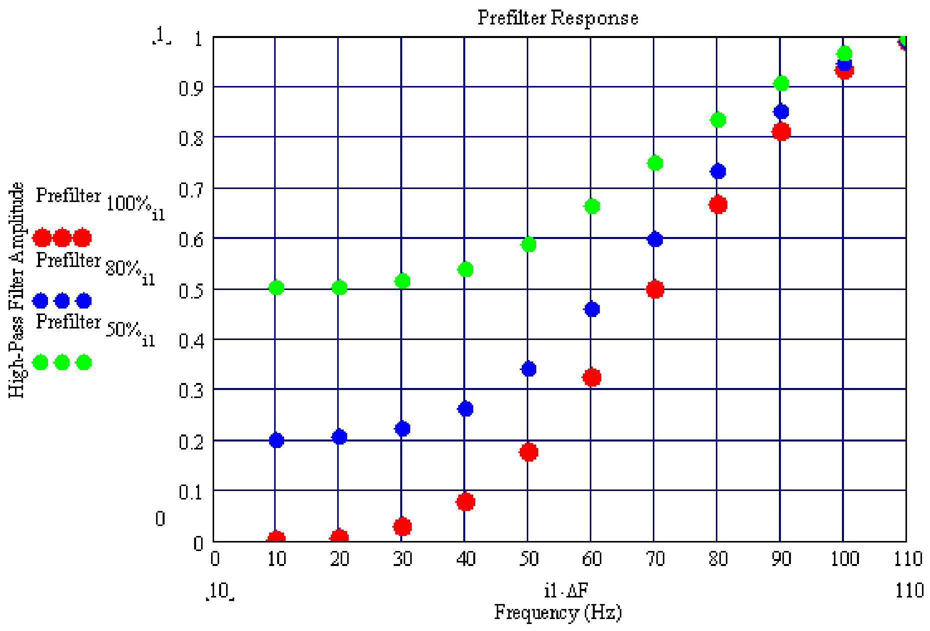


Figure 6. Shape of the pre-filter response, for various values of the Prefilter parameter.

Enable Wind Rejection Software

This parameter turns the wind rejection algorithm on or off. For buried sensor applications, leave the **Enable Wind Rejection Software** box unchecked (off).

Anemometer

This parameter enables an anemometer to be used as a means of adjusting the wind-reject software so as to reduce the nuisance alarms that might otherwise be caused by wind (available only on select APUs).

Wind Rejection

The **Wind Rejection** algorithm monitors the noise floor and automatically and dynamically adjusts the **Gain** so that wind does not cause nuisance alarms. The **Wind Rejection** parameter defines how strongly the algorithm adjusts the **Gain**, for a given amount of wind. A higher **Wind Rejection** parameter results in more reduction in **Gain**, resulting in a larger signal being required in order to create an alarm. After the wind subsides, the wind-reject algorithm returns the **Gain** parameter to its normal setting. Note: during windy conditions you will not see the **Gain** parameter change – the changes to the **Gain** parameter are carried out automatically and out of view of the user.

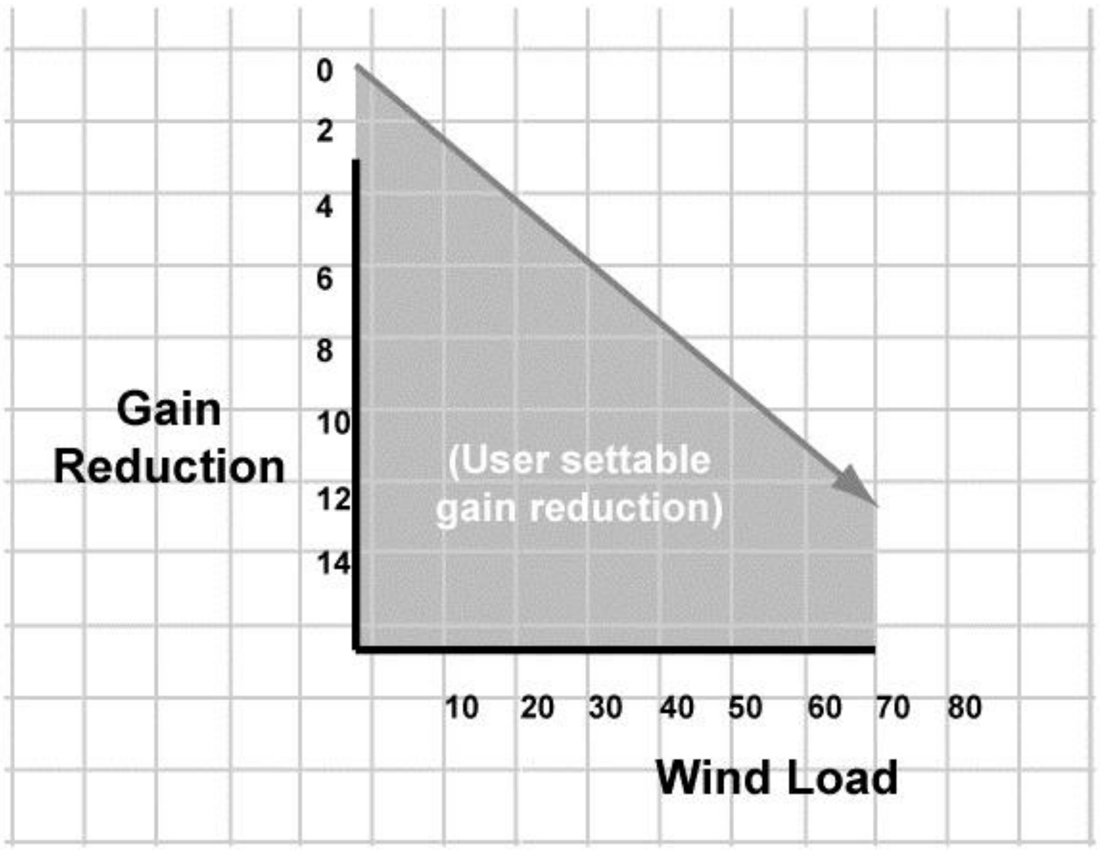


Figure 7. The wind-rejection algorithm reduces the Gain parameter in the presence of wind, in order to avoid nuisance alarms.

Low- and high-frequency filters

The low- and high-frequency filters are used to reject un-wanted frequencies that might otherwise cause nuisance alarms. All frequencies below the low frequency setting, and above the high-frequency setting, are ignored (see figure 8).

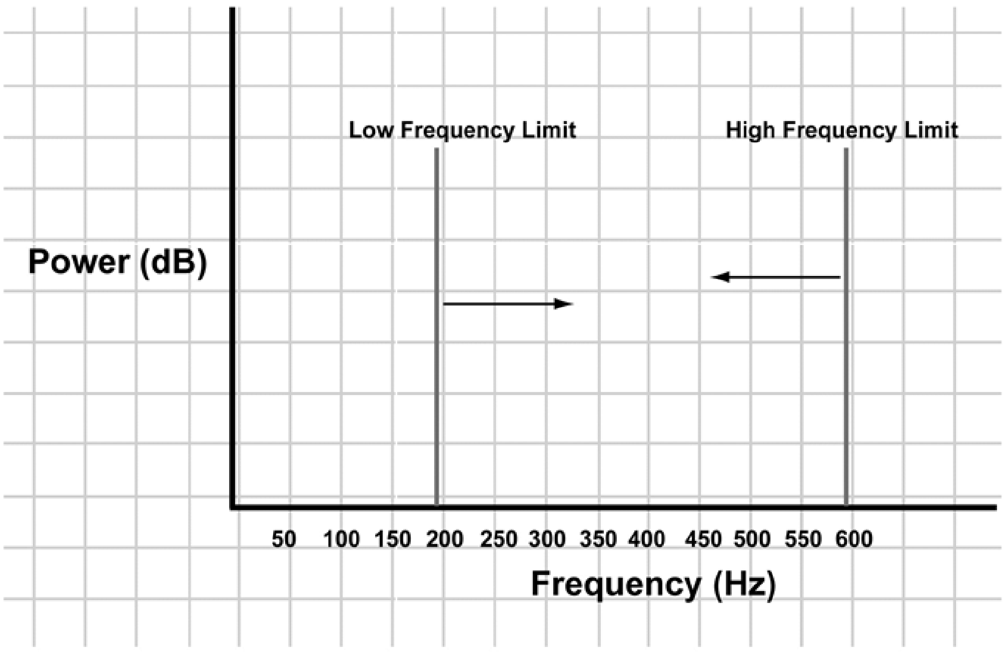


Figure 8. The low- and high-frequency filters define what spectral components of the signal will be used in the detection process. All signal energy lower than the low-frequency limit, or higher than the high-frequency limit, is ignored.

Event classification

The settings that define signals that qualify as events include **Level**, **Low Level Tolerance**, and **Duration**.

Level

Level (also, sometimes, called **SIGNAL Level** or **SIGNAL**) defines the signal threshold that must be exceeded in order to generate an event. The overall system sensitivity is equal to **Gain** minus **Level**, and so the **Gain** parameter should always be greater than **Level** parameter.

The lower the level of **Signal**, the more sensitive the system becomes. Setting **Level** to 5, for example, would require much less signal to trigger an event than setting **Level** to 15. The *effective* **Gain** for the system is the difference between level and **Gain**: effective **Gain** = **Gain** - **Level**.

Gain ➡	20	20	25	15
Level setting, Processor 1	10	15	12	8
Level setting, Processor 2	10	5	10	10
Effective **Gain** Processor 1	10	5	13	7
Effective **Gain** Processor 2	10	15	15	5

Duration:

To be classified as an event, the intrusion signal must be higher than **Level** for a minimum amount of time that is defined by the **Duration** parameter. The **Duration** parameter works in tandem with the **Level** setting. The duration of a signal can help distinguish an intrusion attempt from a nuisance signal. Some low-magnitude nuisances can cause more lengthy disturbances than an intruder, while high-magnitude nuisances, such as the popping of a metal fence as the temperature changes, have very short durations. Coupling signal **Duration** with the signal **Level** can thus help screen out a number of nuisances.

Duration defines how long the signal must last, in order for it to be classified as an event. The algorithms will log an event if the signal intensity equals or exceeds **Level** for the length of time indicated by **Duration**. A shorter disturbance won't trigger an event, no matter how intense it is. Assume, for example, that **Level** is set to 10 dB and that **Duration** is set to 3 (= 3/10ths of a second); in this condition, to be classified as an event, the signal must exceed 10 dB for 0.3 seconds (see figure 9).

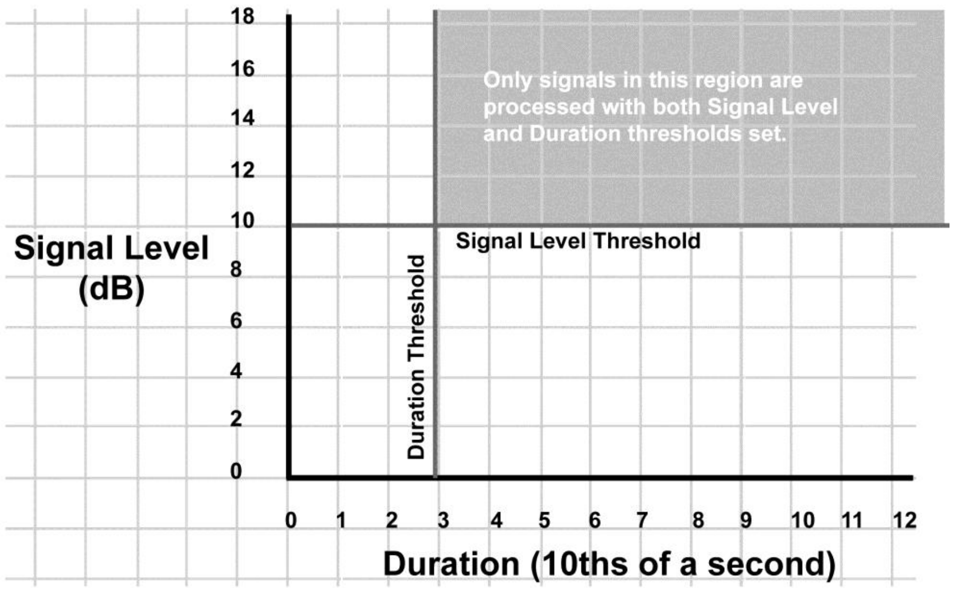


Figure 9. Signal level, coupled with signal duration, define what intrusion signals qualify as potential events.

Low Level Tolerance

There is an exception for the rule defined by **Level** and **Duration** (see figure 10), and that exception is defined by the **TOLERANCE** parameter. This parameter specifies a tolerance for the signal-level setting in the case of low-level signals. If the signal is lower than the **Level** threshold, by no more than the **TOLERANCE** setting, then it can still register as an event, provided that it lasts longer than **Duration** multiplied by an amount that is automatically set by the processor. The higher the **TOLERANCE** value, the longer the duration that is automatically determined by the processor. If the signal is lower than the **Level** threshold by more than the **TOLERANCE** setting, it cannot generate an event regardless of how long it lasts.

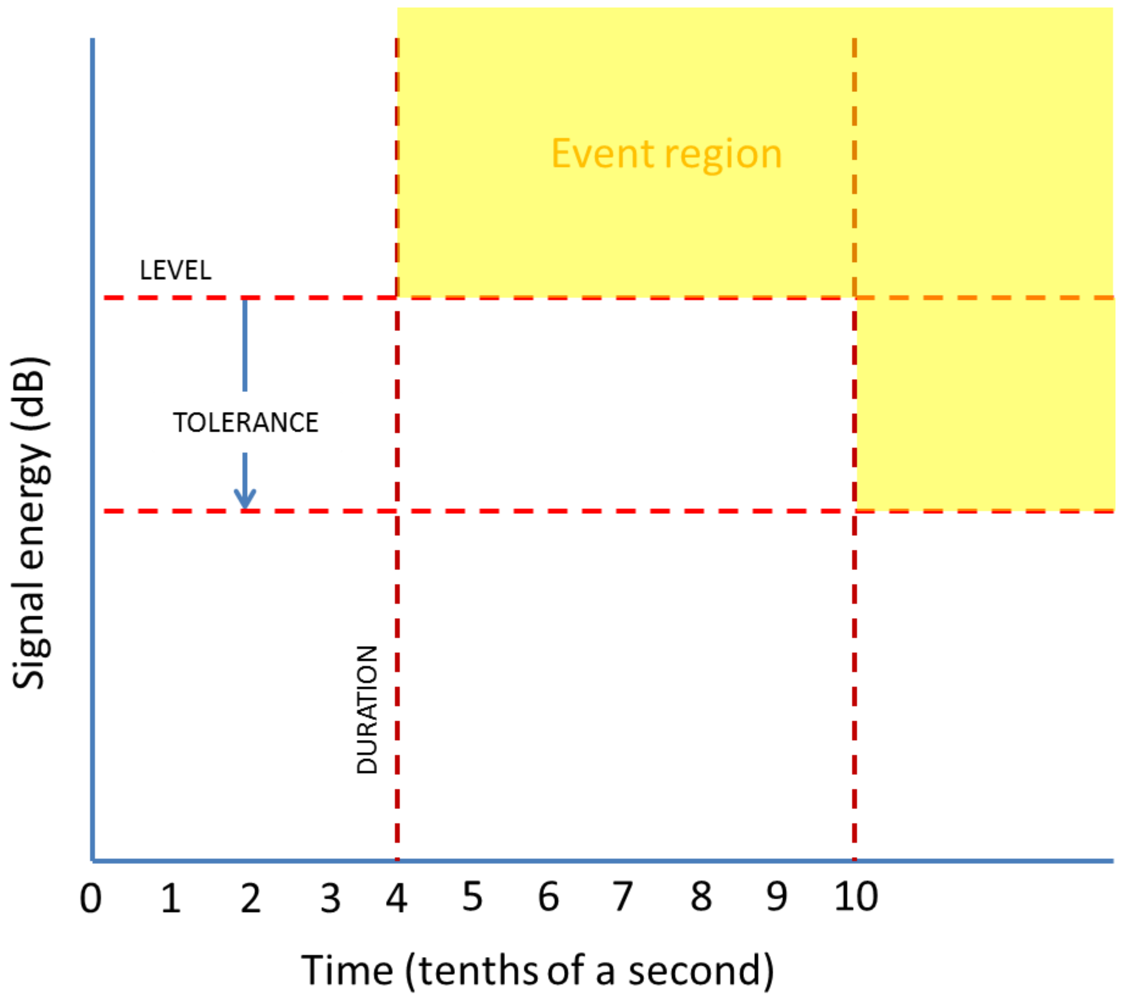


Figure 10. How Level, Duration and Tolerance work together to define what signals will be classified as events. In this example, there is one duration for signals that exceed the Level setting, and a different duration for signals that are lower than the Level threshold, but still within the TOLERANCE setting. Note: the increase in duration is set automatically by the APU.

Alarm classification

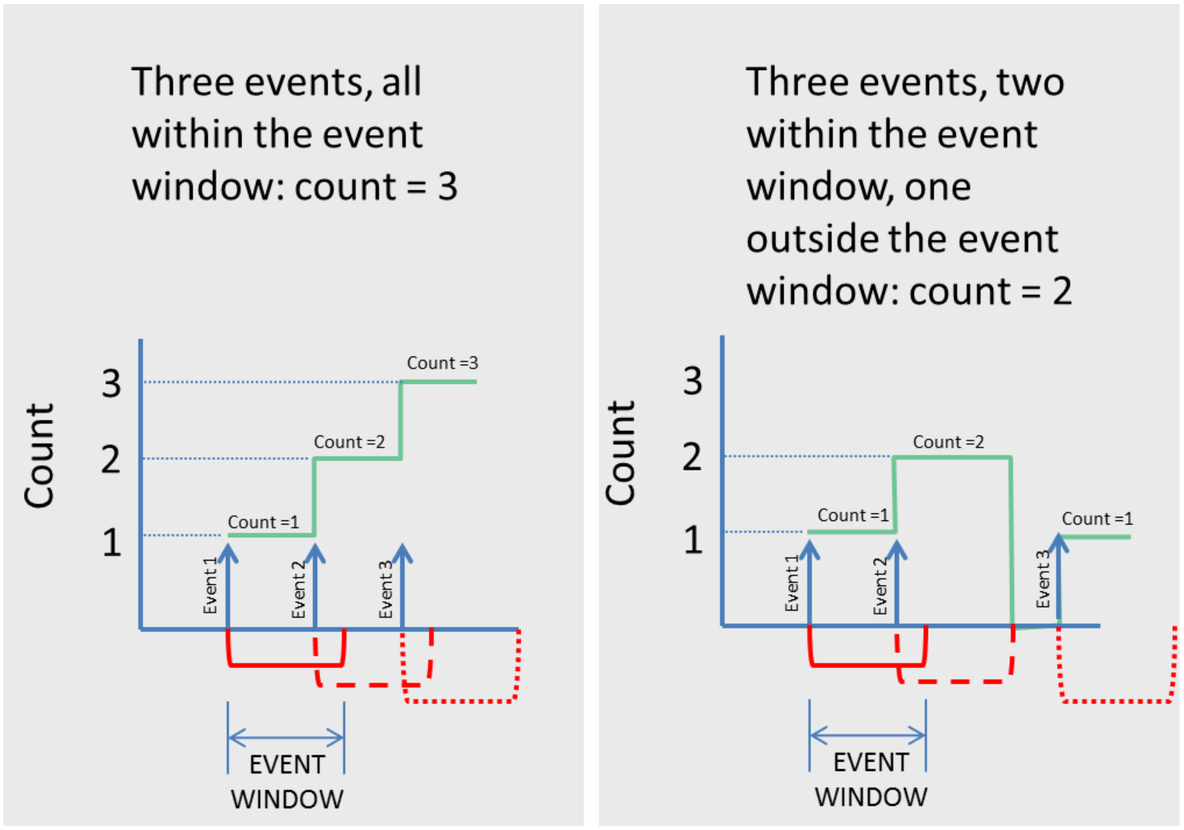
Events are not alarms, but rather the precursors to alarms. For an alarm to exist, a certain number of events must be recorded in a given sequence and time. The parameters **Event Count**, **Event Window**, and **Event Mask Time** are the three principal settings that define when a series of events will generate an alarm.

Event Count

Event count defines how many events must be detected, within a specified amount of time, before an alarm is generated. **Event Count** is one of the best ways to differentiate an intruder from a nuisance. An intruder cutting a fence, for example, will create semi-periodic acoustic spikes as they cut through chain link. This sort of semi-periodic pinging is very uncharacteristic of more random phenomena such as an animal or tree branch hitting the fence.

Event Window

The **Event Window** is the maximum amount of time that can exist between events, for the events to be counted as an alarm. When an event is triggered, a timer is initialized. If another event occurs before the window of time has elapsed, the timer is restarted and the event count is incremented by 1. If an event does not occur in the window of time, the event counter is reset to zero. If the event counter exceeds the **Event Count** parameter, an alarm is generated (see the example in the figure below):



Three events, all within the event window: count = 3
Count
3
2
1
Count =3
Count =2
Count =1
Event 1
Event 2
Event 3
EVENT WINDOW
Three events, two within the event window, one outside the event window: count = 2
Count
3
2
1
Count =2
Count =1
Count =1
Event 1
Event 2
Event 3
EVENT WINDOW

Figure 11. How Event Window determines which events will, or won't, be included in the event count.

Since the event must be completed within the **Event Window** to be counted towards an alarm, the **Event Window** must be at least as long as the event duration. Also, if the event-mask time is not zero, then a delay as long as the event mask time occurs before the event window time begins.

Event Mask Time

Event Mask Time defines a period of time (usually short) after an event, during which signals are rejected. This helps tune out multiple events that might otherwise be caused by a single high-energy impact (see figure 9). Oscillations from nuisances usually die down within 0.5 second.

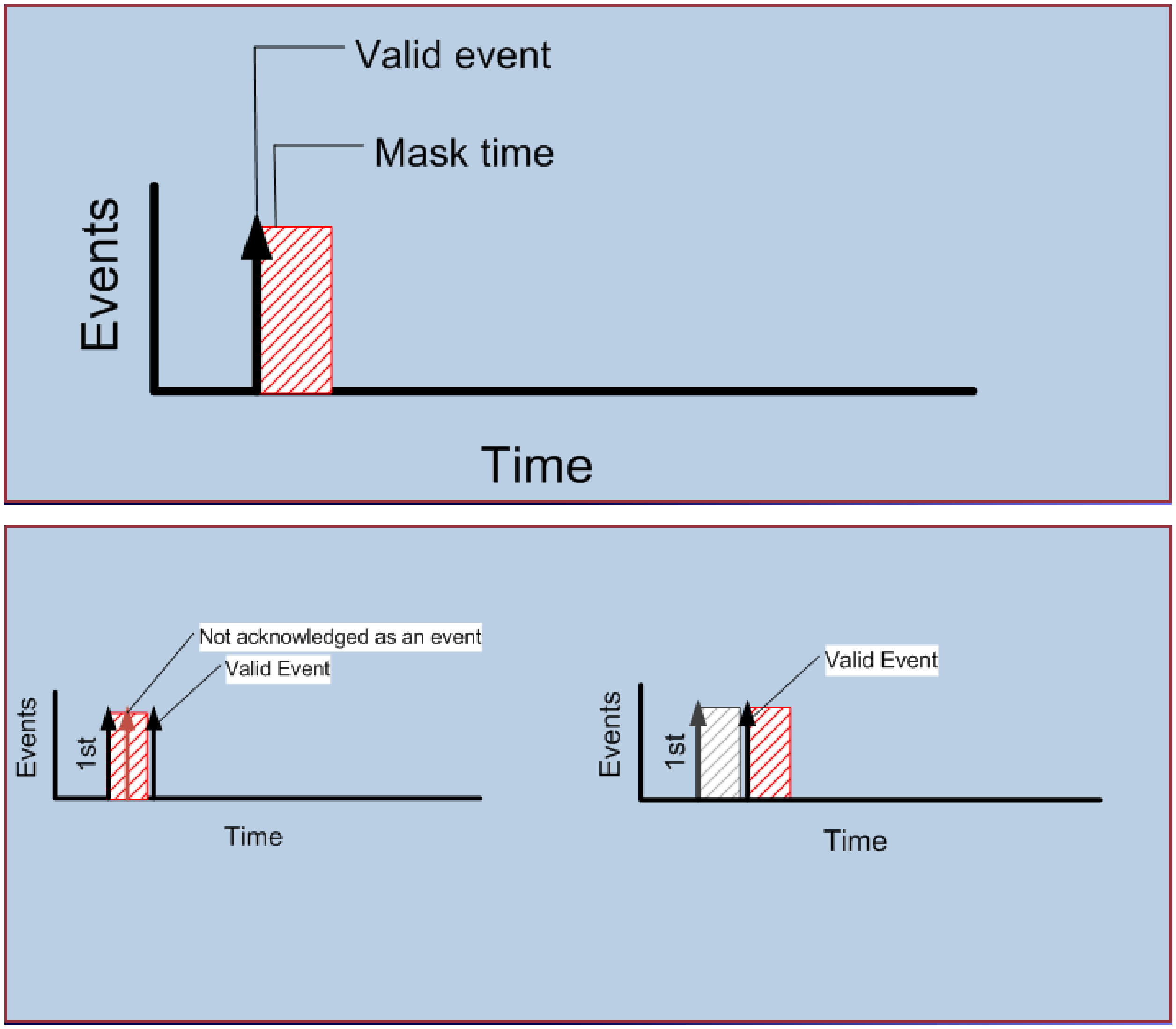


Figure 12. Event mask time used to prevent multiple events from being generated by a single large signal.