BROADCOM®

Application Note

AFBR-S6 Pyroelectric Detectors

MCU Implementable Flame Detection Algorithm

Introduction

Optical-based flame detection systems that use infrared radiation work by comparing the energy levels within certain ranges of wavelengths of light. For example, the ezPyro® TO flame detection kit (part number AFBR-S6DPYEFL02) uses three different wavelength ranges to determine whether there is a fire. This process involves the following sensors:

- A sensor that is used to provide good signal strength for human infrared emissions.
- A sensor that is used to detect sunlight effects (and other industrial sources of IR).
- A sensor that is used to detect the emissions associated with hot CO_2 created during a fire.

Broadcom flame detectors come in three main types:

- Analog TO-39
- Digital ezPyro TO
- Digital ezPyro SMD

All three types of flame detector can be supplied with a variety of interference filters. To see the full list of flame filters, go to https://www.broadcom.com/products/optical-sensors/pyroelectric.

Broadcom detectors are the best choice for infrared flame detection. Achievable parameters of detection are as follows:

- Long detection range (85m)
- Wide detection angle (110^o)
- Instantaneous detection
- Low current consumption (3 µA to 23 µA)

These features allow for the development of an effective flame detection module.

The Broadcom flame detection kit can provide a fast and simple aid in such development. In order to detect a flame, an algorithm is required to analyze signals coming from each detector. This document covers a possible flame detection algorithm that has been implemented with ezPyro flame detectors.

Table of Contents

Chapter 1: Flame Detection Method 3
Chapter 2: Frequency Consideration 4
Chapter 3: Flame Detection Algorithm Processing 5
Chapter 4: Initial Sensor Data Processing 6
4.1 Low-Pass Filter 6
4.2 High-Pass Filter 7
4.3 Standard Deviation 8
4.3.1 MCU Limitations 9
4.4 Signal Multiplier 10
4.5 Summary of Data Processing 10
Chapter 5: Ratio-Based Algorithm 12
Chapter 6: Flow Chart 13
Chapter 7: Potential Enhancements to the Algorithm 14
Chapter 8: Conclusions 15
Revision History 16
AFBR-S6-AN100; August 11, 2022 16

Chapter 1: Flame Detection Method

Flame detection in the IR spectrum is based on monitoring the radiation coming from a flame itself. A typical flame is coated with gases such as CO_2 and CO (hydrocarbons), or water vapor in case of H_2 flames. These gases flicker along with the flame in a frequency range of 1 Hz to 5 Hz (typ.), since flame is not a static phenomenon. By monitoring the spectral range of the absorption of these gases at the mentioned frequency range, you can detect the presence of a flame very precisely. The following figures show exemplar flame signals that were created with a lighter and detected at 4.48-um central wavelength (CWL) and 620-nm half-power bandwidth (HPB) in a single flame ignition instance (Figure 1) and in a continuously flickering flame (Figure 2).

Figure 1: The Spike Occurs when the Light is First Lit

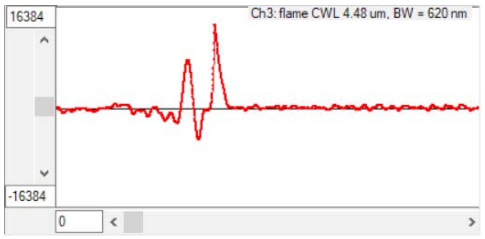
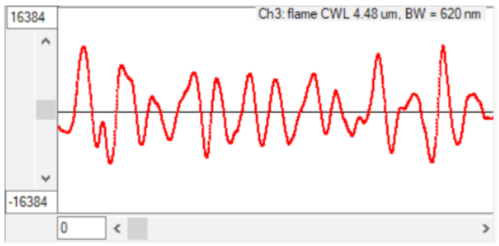


Figure 2: Moving the Lighter around Causes Variation in the Emissions, Which Produces a Continuous Signal from the Pyroelectric Sensor Element



To see these effects live, watch the video at https://www.youtube.com/watch?v=wPcjGsfYd1U. At 1 minute 51 seconds, you can see the response to a static lighter flame and notice that the rest of the time the flame is being moved to induce variation.

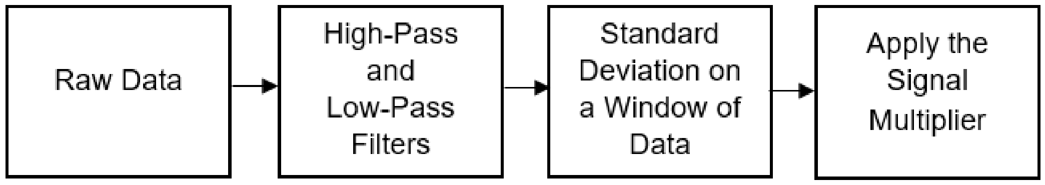
Chapter 2: Frequency Consideration

Each Broadcom detector signal has the raw data put through a high-pass filter and a low-pass filter to cut out frequencies that are not associated with the dynamics of fire. In general, a good range to capture is between 1 Hz and 30 Hz. However, this is a large generalization and every circumstance may produce different results. For example, a forest fire will produce different results than a small indoor fire. The specific frequency ranges need to be determined through testing in order to reduce the range of frequencies being analyzed for the specific application that a system is being designed to work in. The first step to choosing your bandwidth for filtering should be to collect raw data for the certification fire that is to be used and to plot the Fast Fourier Transform (FFT) for a stable period of burning of the fire. This FFT plot will indicate the dominant frequencies associated with the dynamics of the fire, and consequently the filtering that is best suited to maximize SNR.

Chapter 3: Flame Detection Algorithm Processing

The purpose of the flame detection algorithm is to turn raw data from the sensors into a measure of the magnitude of incident light variations. This involves three stages of processing on the raw data (part of this can be accomplished with the ezPyro ASIC settings). This document describes this processing based on the assumption that an analog sensor is being used without additional hardware filtering. Some of the sections in this document can be omitted when implemented on the digital ezPyro family by making use of the ASIC settings. If a section can be omitted, it will be noted at the end of the section.

The flow of data processing is shown in the following block diagram.


Raw Data
High-Pass and Low-Pass Filters
Standard Deviation on a Window of Data
Apply the Signal Multiplier

The high-pass and low-pass filters can be applied in either of the following sequences:

- Low-pass filter first then high-pass filter.
- High-pass filter first then low-pass filter.

The important point is that the output of the filtering that is performed first becomes the input to the second filter. The combination of both provides the band pass between the desired frequencies set by the frequency cutoffs of each filter.

Chapter 4: Initial Sensor Data Processing

Both low-pass filters and high-pass filters can be implemented in either hardware or software. This section describes the equation that needs to be implemented in code to achieve the filtering in processing.

4.1 Low-Pass Filter

The discrete version of the filter is derived from the following analog low-pass filter equation.

Equation 1:

$$V_{out}(t) = V_{in}(t) - RC\left(\frac{dV_{out}(t)}{dt}\right)$$

If we allow the input value at time *t* to be designated x_i and the output value to be designated y_i (y_{i-1} is the previous output value), then we can write the discrete time base equation of the analog equivalent.

Equation 2:

$$y_i = x_i - RC\left(\frac{y_i - y_{i-1}}{\Delta t}\right)$$

Where Δt is the time period of sampling of the ADC. Rearranging to make y_i only appear as the subject of the formula yields the following equation.

Equation 3:

$$y_i = x_i\left(\frac{\Delta t}{RC + \Delta t}\right) + y_{i-1}\left(\frac{RC}{RC + \Delta t}\right)$$

This is the equation that should be implemented in code to achieve the low-pass filter effect through data processing.

If we consider the bracketed terms, we can further define the following equation.

Equation 4:

$$\alpha_{LP} = \frac{\Delta t}{RC + \Delta t}$$

And rewrite the previous equation.

Equation 5:

$$y_i = x_i\alpha_{LP} + y_{i-1}(1 - \alpha_{LP})$$

The final part to implementing this approach in code is to select a cutoff frequency. This is accomplished using the frequency cutoff equation for an RC filter.

Equation 6:

$$f_c = \frac{1}{2\pi RC}$$

Equation 7:

$$RC = \frac{1}{2\pi f_c}$$

A recommended value for the low-pass filter in general flame detection applications is 30 Hz. However, every application is different, so testing should be performed to acquire the best values for a particular situation. If the frequency for cutoff is 30 Hz and the sample period is 1 ms, then α_{LP} becomes the following equation.

Equation 8:

$$\alpha_{LP} = \frac{\Delta t}{RC + \Delta t} = \frac{0.001}{\frac{1}{60\pi} + 0.001} \approx 0.1586$$

NOTE: When using the ezPyro family of sensors, the high-pass and low-pass filters are contained in the ASIC, so it is not necessary to do further filtering (unless more specific frequency cutoffs are required than are available in the ASIC).

4.2 High-Pass Filter

The discrete version of the filter is derived from the following analog high-pass filter equation.

Equation 9:

$$V_{out}(t) = RC\left(\frac{dV_{in}(t)}{dt} - \frac{dV_{out}(t)}{dt}\right)$$

If we allow the input value at time *t* to be designated x_i and the output value to be designated y_i (x_{i-1} and y_{i-1} are the previous input and output values respectively), then we can write the discrete time base equation of the analog equivalent.

Equation 10:

$$y_i = RC\left(\frac{x_i - x_{i-1}}{\Delta t} - \frac{y_i - y_{i-1}}{\Delta t}\right)$$

Where Δt is the time period of sampling of the ADC. Rearranging the formula to make y_i only appear as the subject yields the following equation.

Equation 11:

$$y_i = \left(\frac{RC}{RC + \Delta t}\right) y_{i-1} + \left(\frac{RC}{RC + \Delta t}\right)(x_i - x_{i-1})$$

This is the equation that should be implemented in code to achieve the low-pass filter effect through data processing.

If we consider the bracketed terms, then we can further define the following equation.

Equation 12:

$$\alpha_{HP} = \frac{RC}{RC + \Delta t}$$

And rewrite the previous equation as follows.

Equation 13:

$$y_i = \alpha_{HP} y_{i-1} + \alpha_{HP}(x_i - x_{i-1})$$

The final part to implementing this approach in code is to select a cutoff frequency. This is accomplished using the frequency cutoff equation for a RC filter.

Equation 14:

$$f_c = \frac{1}{2\pi RC}$$

Equation 15:

$$RC = \frac{1}{2\pi f_c}$$

A recommended value for the high-pass filter in general flame detection applications is 1 Hz. However, every application is different, so testing should be performed to acquire the best values for a particular situation. If the frequency for cutoff is 1 Hz and the sample period is 1 ms, then α_{HP} becomes the following equation.

Equation 16:

$$\alpha_{HP} = \frac{RC}{RC + \Delta t} = \frac{\frac{1}{2\pi}}{\frac{1}{2\pi} + 0.001} \approx 0.9938$$

NOTE: When using the ezPyro family of sensors, the high-pass and low-pass filters are contained in the ASIC, so it is not necessary to do further filtering (unless more specific frequency cutoffs are required than are available in the ASIC).

4.3 Standard Deviation

Once the filtering is completed to remove frequencies that are not involved in the phenomena that is being analyzed, the next stage is to create a window of data based on the output of the filtering functions. This full window of data can then be analyzed to give a value for the signal strength within the desired frequency range.

At this point, one of the choices that has to be made is the number of data points that the window of data contains. The impact of the size of the window is that a small window will provide a faster response to a change in signal strength, but will be more susceptible to short noise events producing false positives. Conversely, a large window of data will provide a slower response, but more robustness to short term causes of noise. This choice needs to be made based on the requirements of the application and the robustness of the system design to reduce undesired effects causing output signal.

The reason to use standard deviation rather than RMS is to take any drift of the signal or DC offsets into account. Although the use of filtering will reduce the low frequency drifts, it will not entirely remove it. Therefore, using a calculation that takes these effects into account is desirable.

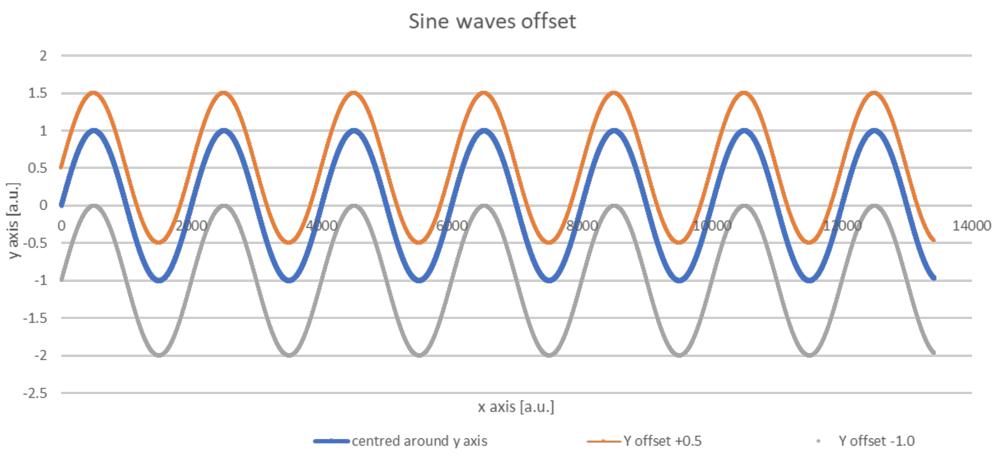
Equation 17:

$$SD = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \bar{x})^2}{N}}$$

The slight difference between the normal standard deviation and the preceding equation is that the denominator is *n* rather than (*n* – 1). This is because the entire data set in this case is known and does not require Bessel's correction as is normally standard for most uses of the equation; it is a special case of RMS with the mean subtracted from each sample to take low frequency drift and DC offsets into account and stop them affecting the results.

The following figure shows the difference in signals taken as a standard deviation described by the preceding equation and the RMS value. Y-axis offsets have been manually added to the dataset to mimic the signal drift in a real system.

Figure 3: Standard Deviation versus RMS Values of Pure Sine Waves with a Y-Axis Offset



	Sine centred around y axis	Sine with Y offset -1.0	Since with Y offset +0.5
RMS	0.7080	0.8781	1.2090
St. Deviation	0.7077	0.7077	0.7077

4.3.1 MCU Limitations

If we use the ezPyro sample rate of 1 kHz as an example and use 1 second of data to consider the signal strength, we would need a window size of 1000 samples of data to have the standard deviation calculated on. For a lot of MCUs, this is a problem in terms of both the memory available and whether the MCU can calculate a standard deviation window of this size in the 1 ms that is available before the next sample arrives. This has been dealt with in the ezPyro API example code by only using 100 data points per standard deviation window and then calculating the average of 10 of these windows. This approach produces a value for the signal strength over that 1-second period, but does so while using far less memory and computational power than a single 1000 data point standard deviation calculation.

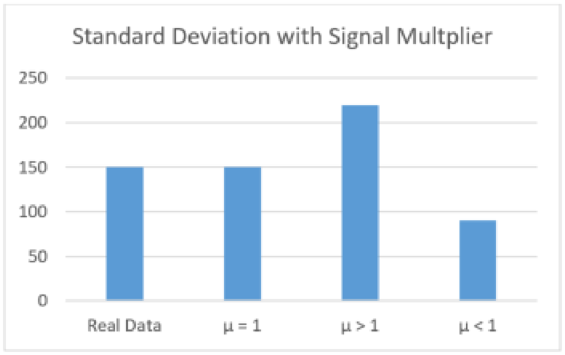
To further reduce the computation time of the standard deviation, the following standard deviation calculation method has been used: https://www.strchr.com/standard_deviation_in_one_pass.

4.4 Signal Multiplier

This next stage allows for further calibration of the system and is simply a means to reduce or increase the value that was produced by the standard deviation performed previously. This allows for the compensation of differences in responsivity and noise levels of the sensors used in the system. This is just a way to compensate for system-to-system differences and is used to meet the requirement of calibrating each system independently while keeping the ratio check value the same. Note that the calibration could involve keeping all of these values constant and changing the ratio check value instead. Which one is used for calibration is up to the system developer.

The effect of μ_x is shown in the following figure and is simply multiplication.

Figure 4: Signal Multiplier Effect

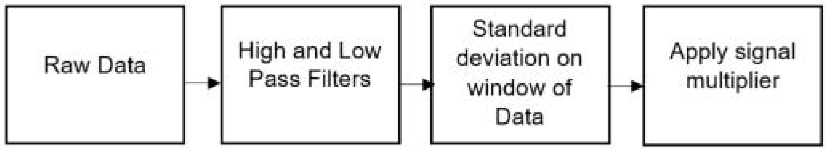


This is the end of the data processing. The next stage is to do a comparison of the values produced by each sensor's data, having gone through these stages to make a decision on whether the data is indicative of a flame in the field of view of the system.

4.5 Summary of Data Processing

The previous sections described the stages required to process the sensor's data into a value indicative of signal strength for each sensor.

1. Apply a low-pass filter algorithm and output this value into a high-pass filter algorithm to produce the desired band pass signal.
2. Use the output of the band pass filter to create a data set *N* elements long. Smaller values of *N* produce faster response times, but are more prone to short term noise effects producing false positives. Conversely, larger data sets produce more robust, but slower response detection capabilities.
3. Perform standard deviation on the data sets.
4. Apply signal multipliers to the standard deviation results to fine tune the system for responsivity differences between sensor combinations in each system.


Raw Data
High and Low Pass Filters
Standard deviation on window of Data
Apply signal multiplier

The outcome of the preceding stages results in a data point for each sensor, which for the next section will be labeled according to the ezPyro SMD flame demo kit channels. The labels specific to this use case are as follows:

1. $\varepsilon_1 = Ch1\ Human\ Motion\ Rejection\ (5.0\ um\ LP) = \mu_1(SD_1)$
2. $\varepsilon_2 = Ch2\ Sunlight\ Rejection\ (3.91\ um\ BP) = \mu_2(SD_2)$
3. $\sigma_1 = Ch3\ Flame\ Channel\ 1\ (4.48\ um\ BP) = \mu_3(SD_3)$
4. $\sigma_2 = Ch4\ Flame\ Channel\ 1\ (4.64\ um\ BP) = \mu_4(SD_4)$

ε_1 and ε_2 represent the rejection channels after the data processing listed in this section. σ_1 and σ_2 represent the flame channels output for the data processing listed. With these, it is now possible to check the ratio of the post processed flame channels against the rejection channels. This is described in the next two sections.

Chapter 5: Ratio-Based Algorithm

Since the algorithm is based on the ratio of the flame channels to rejections channels post processed values, you need to select your ratio, which if met will produce a fire detection result. This comes down to setting up the values. The values will differ from system to system, but can be adjusted to take system differences into account. In this example, the ratio used is 1.5.

$$Human\ Motion\ Rejection\ Ratio\ Threshold\ = 1.5 = \frac{\sigma_1}{\varepsilon_1}$$

or

$$Human\ Motion\ Rejection\ Ratio\ Threshold\ = 1.5 = \frac{\sigma_2}{\varepsilon_1}$$

This means that at least one of the flame channels must produce an output signal (after the processing described in section 3) that is at least 1.5 times of either of the rejection channels. It is the same for the sunlight rejection channel. In this example, the ratio used is 1.2.

$$Sunlight\ Rejection\ Ratio\ Threshold\ = 1.2 = \frac{\sigma_1}{\varepsilon_2}$$

or

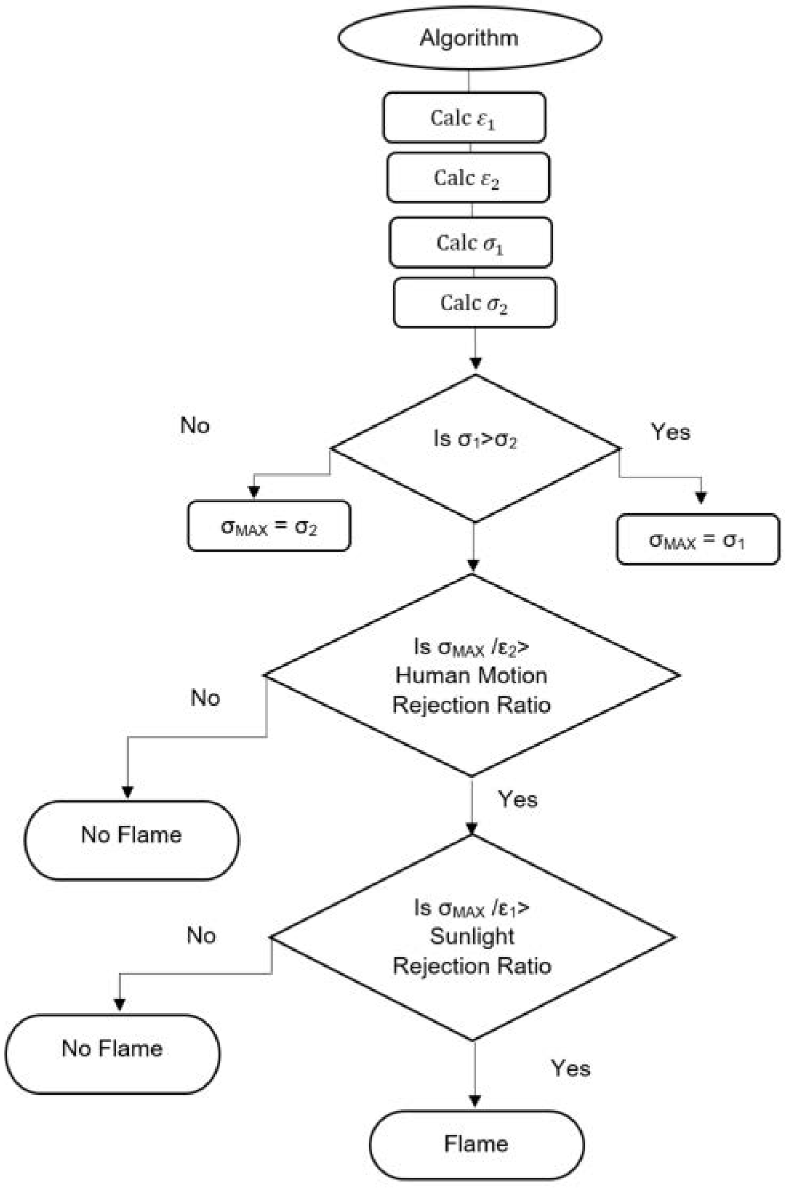
$$Sunlight\ Rejection\ Ratio\ Threshold\ = 1.2 = \frac{\sigma_2}{\varepsilon_2}$$

For a flame event to be considered real, one of the flame sensor's data must be greater than the ratios of both of the two rejection channels.

Since one of the flame channels will be larger than the other, it is only required to look at the ratios of the largest flame channel signal against the rejections.

This is shown in the flowchart in the following chapter.

Chapter 6: Flow Chart


Algorithm
Calc ε_1
Calc ε_2
Calc σ_1
Calc σ_2
Is $\sigma_1 > \sigma_2$
No
Yes
$\sigma_{MAX} = \sigma_2$
$\sigma_{MAX} = \sigma_1$
Is $\sigma_{MAX} / \varepsilon_2 >$ Human Motion Rejection Ratio
No
Yes
No Flame
Is $\sigma_{MAX} / \varepsilon_1 >$ Sunlight Rejection Ratio
No
Yes
No Flame
Flame

Chapter 7: Potential Enhancements to the Algorithm

The final part of the flow diagram can be replaced with a counter. The counter would store some number of fire or no-fire events. When the ratio of fire to no-fire events reaches some proportion of the total number of recorded events, then a fire can be said to be in the FoV of the sensors. This makes the system less responsive, but more robust against short time-duration events that could cause a false alarm.

Chapter 8: Conclusions

The key algorithm steps are as follows:

1. Perform post processing as described in Chapter 3, Flame Detection Algorithm Processing.
2. Determine the largest flame channel signal. In a 3IR system, calculate the flame channel signal strength because there is no alternative flame sensor.
3. Divide the Flame signal by rejection channel 1. If it is greater than that channel's threshold, check the other rejection channel.
4. Divide the Flame signal by rejection channel 2. If it is greater than that channel's threshold, a flame event has been registered.

The presented algorithm can be used for improvement in a flame detection technique. It has been utilized by customers worldwide and has been proven to improve detection methodology. Thanks to unique technology, Broadcom detectors are the fastest responding pyroelectric detectors available on the market. Broadcom detectors can resolve a flame signal in the frequency domain very well, allowing for novel and better techniques to analyze and detect flames.

Revision History

AFBR-S6-AN100; August 11, 2022

- Initial release.

BROADCOM®

Lead (Pb) Free
RoHS 6 fully compliant
RoHS 6 fully compliant options available; -xxxE denotes a lead-free product