# Equalization: The Correction and Analysis of Degraded Signals

White Paper

Agilent Technologies

Ransom Stephens
V1.0
15-August-2005

Equalization is a signal processing technique used to extend the use of standard flame retardant type-4 (FR4) printed circuit boards (PCBs) to higher data rates. The nonlinear frequency response and loss characteristics of FR4 can destroy signal quality at data rates as low as 3 Gb/s over trace lengths of a few inches. By applying simple corrections the signal quality can be recovered. When the corrections are applied at the transmitter it is called de-emphasis (or pre-emphasis) and when they are applied at the receiver it is called equalization.

This is an introductory article written to help engineers understand the concept of equalization and terms used in the development of emerging technologies that use standard materials (e.g., FR4) for buses and backplanes at ever higher data rates.

Equalization has many forms but is fundamentally a signal correction scheme. I start with a simple brute-force description of how a problematic bit in a specific signal is equalized. With the specific example in mind, I take you in the opposite direction and show how equalization emerges by inverting the impulse response of a system. With both the esoteric and transparent descriptions in hand it's easy to tie the two together and extend from a simple linear equalizer to the popular nonlinear Decision Feedback Equalizer (DFE). Then, to provide you the tools you need to get started developing equalizers I'll give you some practical advice on how to tune their parameters. We finish with a discussion of how to analyze and evaluate equalizers for different systems.

# 1. Introduction

Equalization enables the use of standard FR-4 PCB materials at data rates where its frequency and attenuation characteristics seriously degrade the signal. The problem is illustrated in Figure 1 where the eye-diagram of a 5 Gb/s signal is shown both as it leaves the transmitter and after traversing 40 inches of PCB trace.

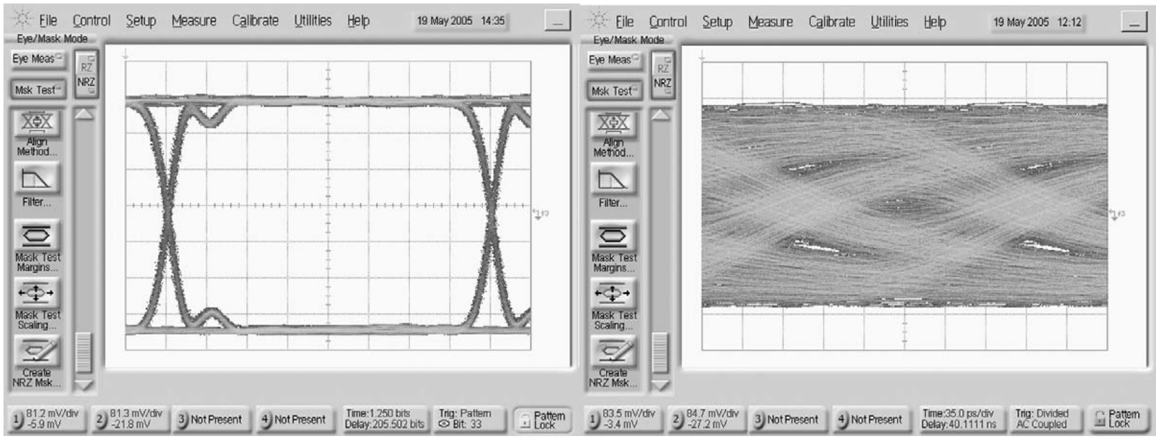


**Figure 1: (a) A clean eye observed from the transmitter, (b) the same signal observed at the end of a substantial trace on FR4.**

Equalization is a set of techniques for compensating the degrading effects of transmission paths. The focus of this discussion is on the effects of PCBs but equalization can also correct the effects of multi-path interference in wireless signals and both chromatic and polarization mode dispersion in optical signals.

There are many types of equalizers. We'll start with the Linear Feedforward Equalizer (LFE, also known in the literature as FFE) and build up to the Decision Feedback Equalizer (DFE). There are many other types of equalizers out there that we will not discuss: Non-causal DFE, selective time reversal DFE, and bidirectional DFE to name a few [1].

As data rates increase the time between logic voltage swings decreases. The fundamental frequency corresponds to a clock frequency of half the data rate. Data rates of 1-10 Gb/s are well into the radio, toward the microwave, realm of the electromagnetic spectrum. At these frequencies the PCB behaves like a dielectric waveguide. Logic signals are transmitted as electromagnetic waves that flow through the FR4 dielectric medium guided by the conducting trace. A PCB is a very complicated waveguide – not at all like a nice uniform geometry pipe – and there is no closed-form analytic solution to Maxwell's Equations [2]. But the system is cheap and, with equalization, can be made to work.

Inter-symbol interference (ISI) is caused by the non-uniform frequency response of the system which modifies the pulse-shape of different bits in a signal traversing the waveguide. For example, the dominant frequency of a segment like '0111000' is 1/3 the data rate and a segment like '010' is the same as the data rate. The dispersive effects of the channel – that is, the dielectric response dependence on frequency – changes the pulse shape of the two segments in different ways. Combine this non-uniform frequency response with the loss characteristics of a transmission path and add the possibility of interference of signals from multiple reflections and multiple paths from input to output and you have a big messy problem.

To attain a qualitative grasp of what ISI does to a signal, consider a repeating pattern transmitted through a channel, a pseudo-random binary sequence (PRBS) for example. The shape, or trajectory, of each logic transition in the pattern depends on the number of consecutive identical bits preceding the transition – by virtue of the non-uniform frequency response – and has both voltage and timing components. The eye diagram in Figure 2 was made with pattern lock and averaging to remove random noise so that the trajectories are easily distinguished. An important, and frequently overlooked aspect of ISI is that it affects both the voltage and timing components of the signal.

Since ISI is caused by a combination of:

1. The geometry of the circuit, i.e., the design of the trace
2. The medium from which it is composed, i.e., the conductor and dielectric
3. The voltage swing of the signal

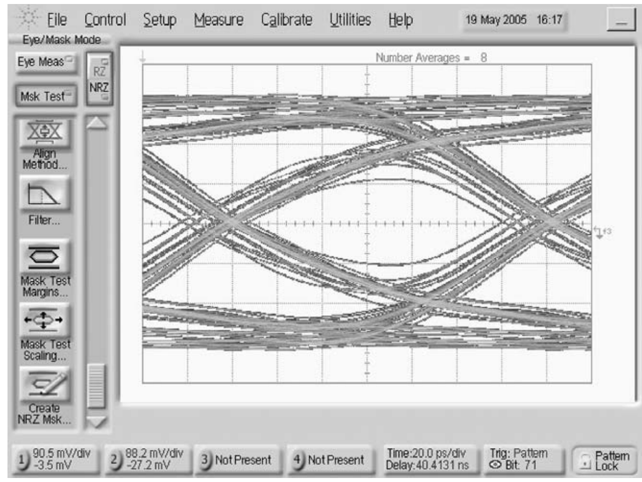


**Figure 2: A signal with ample ISI illustrating the different trajectories of the bit transitions in a repeating pattern.**

and since these are determined prior to signal transmission and subject only to small random fluctuations, ISI can be corrected. That is, since ISI is deterministic, the information of the original signal is still in the received signal whether or not the eye is closed.

Equalization techniques provide a way to discern the original signal given the received signal.

Random noise, on the other hand, is not deterministic and, in general, cannot be corrected by equalization techniques (I want to add a caveat to this, however, because you are clever and it is possible to correct low frequency random fluctuations).

The role of equalization is to invert the problems caused by the transmission channel. Consider Figure 3, starting with an open eye at the transmitter, the channel introduces ISI and there will also be some random noise (from thermal effects) and maybe some crosstalk or electromagnetic interference - indicated by the triangle. The eye emitted by the channel is closed. The equalizer can correct much of the degradation caused by the channel, but it cannot correct the noise. The eye is opened, but rarely to a quality as nice as the signal that emerges from the transmitter.

In the following we'll use the notation introduced in Figure 3:

*s(t)* = the transmitted signal (in volts)
*r(t)* = the signal that emerges from the transmission path or "channel" (in volts)
*e(t)* = the equalized signal (in volts)

Since equalization is performed by the receiver, the opened eye cannot be observed unless *r(t)* is applied to test equipment that can perform the equalization.

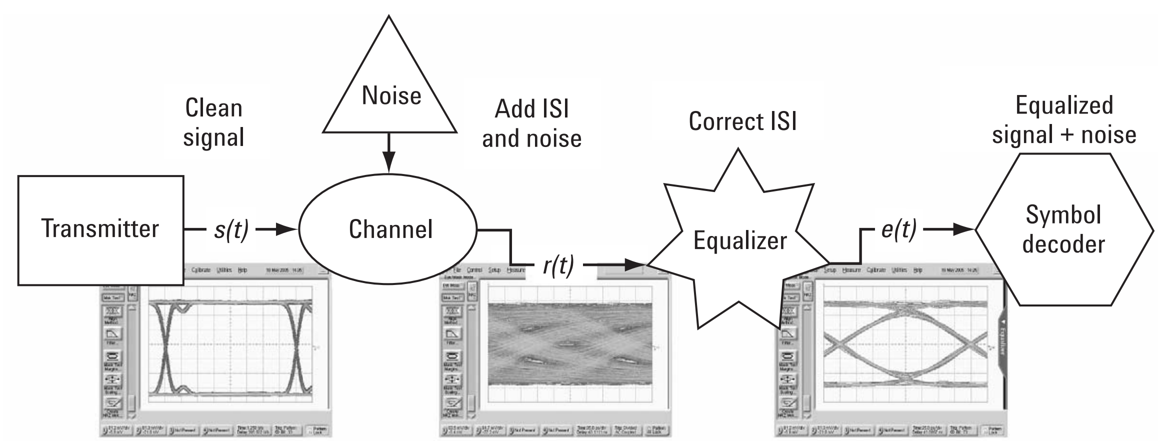


**Figure 3: A simple model of a transmission system.**

Equalization can also be implemented at the transmitter: the transmitted waveform is modified in such a way that the effects of the transmission path result in a clean signal at the receiver. The simplest technique applied at the transmitter is called “de-emphasis.” The transmitter generates a voltage swing that is higher for a bit following a transition than for a bit that doesn’t follow a transition. In other words a logic ‘1’ following a logic ‘1’ has a lower voltage - has been de-emphasized - than a ‘1’ following a ‘0’, as shown in Figure 4. In the jargon of equalization, de-emphasis is a “two-tap filter.”

In this paper we focus on equalization techniques applied at the receiver.

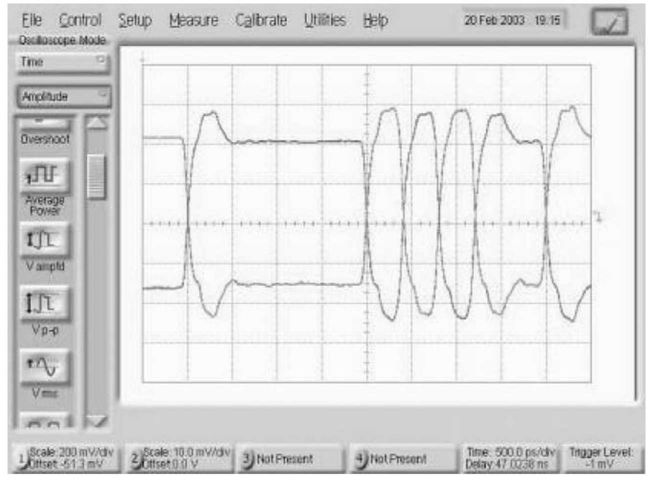


**Figure 4: A de-emphasized waveform.**

## 2. Correcting a problematic bit

The idea is to see what a simple equalization technique does to correct a given bit in a data signal. This way you can see what equalization actually does before looking at the theory and seeing why it works so well.

Consider the bit indicated by the arrow in Figure 5. The signal degradation caused by traversing a long stretch of FR4 backplane puts the logic level of the bit very close to the decision threshold, $V = 0$. What’s happened is that the frequency response of the backplane combined with the frequency content of the waveform surrounding the bit is too slow for the voltage level of the 0 in the 1111**0**1 segment to get low enough to be reliably decoded. In other words, our bit would be misidentified as a 1 in most systems.

The idea behind equalization is to use the voltage levels of the other bits to correct the voltage level of a given bit. If we measure the voltage of every bit in the observed signal, then we can assemble a simple sum to correct a given bit. Here’s some jargon:

**Cursor** – the voltage at the center of a bit. A “**pre-cursor**” is the voltage at the center of a bit prior to the one of interest, and a “**post-cursor**” is the voltage at the center of the bit after.

**Taps** – the correction factors applied to the voltage levels of other bits (i.e., the cursors).

The simplest equalizer is given by the sum of the voltage levels of the bits received prior to the bit of interest multiplied by correction factors; In equalization jargon we’d say, “the sum of the product of the taps and cursors.” Lets index the bits in the signal as shown in Figure 5, so that the bit we’re correcting is indexed one and the previous bits (pre-cursors) are indexed sequentially. Then the corrected voltage level for the bit of interest, is given by

$$e(1) = c_1 r(1) + c_2 r(2) + c_3 r(3) + c_4 r(4) + c_5 r(5) + c_6 r(6) + c_7 r(7) + c_8 r(8) \qquad (1)$$

where $c_i$ are the *taps*, *r(i)* are the voltage levels of the received bits and *e(1)* is the corrected, or *equalized*, voltage of our bit. Notice that the tap values are dimensionless; they’re just correction constants. You can think of them as the ratio of the voltage that the receiver should have seen to the voltage the receiver did see so that their units cancel.

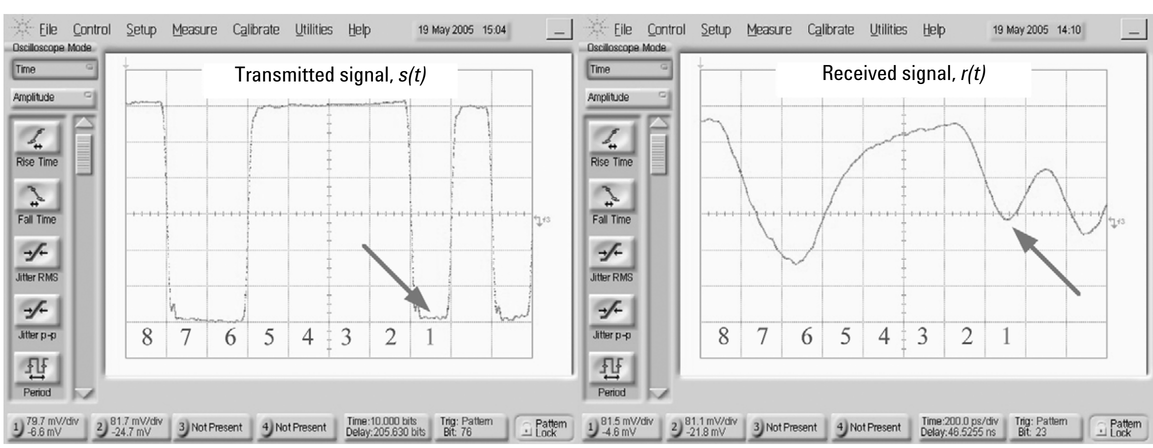


**Figure 5: The same segment of a data signal before and after traversing a backplane. The red arrows indicate the bit studied in the text.**

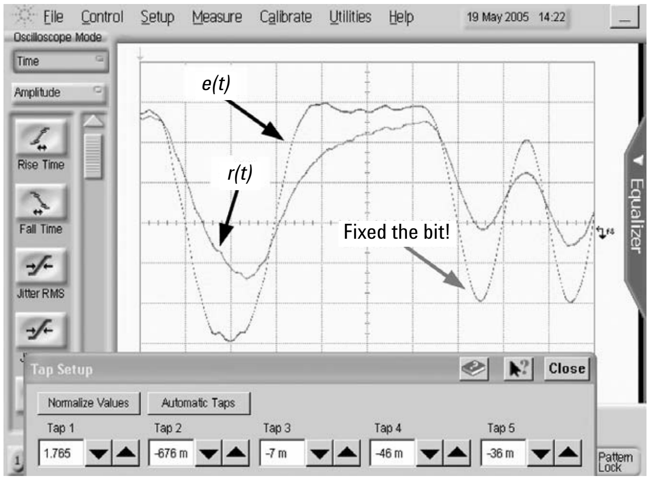


**Figure 6: The received and equalized signals using the built in LFE on the Agilent DCA-J.**

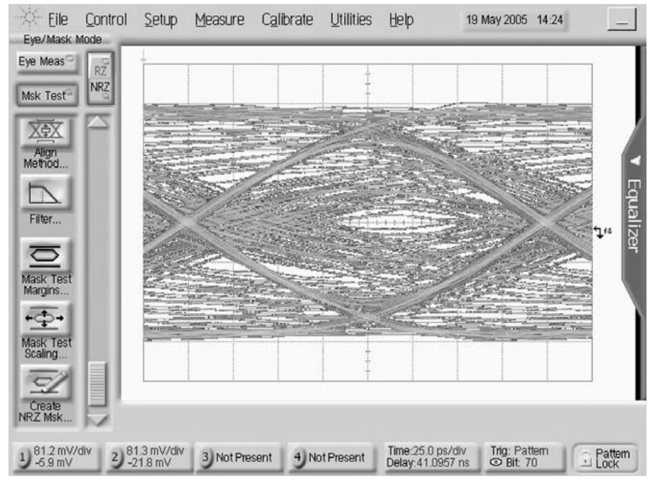


**Figure 7: The received and equalized eye diagrams.**

The equalization technique given by Eq. (1) is called a linear feedforward equalizer (LFE) because: (1) it is linear and (2) it only uses information from previously received bits – that is, it feeds forward information from earlier bits to later bits.

The Agilent 86100C DCA-J with advanced waveform analysis (option 201) provides a nice tool for studying equalization – a built in LFE. Figure 6 shows the received signal, *r(t)*, and the equalized signal, *e(t)*, along with the first five tap values (e.g., "$Tap_1$" = *c1* and the "m" stands for "milli" so that $Tap_2$ = –676m = -0.676).

That only the first two taps in this example are large indicates that a two-tap equalizer might be sufficient. A small number of relevant taps means that we used a pretty high quality transmission channel. A two-tap equalizer can be implemented by the transmitter through signal de-emphasis.

The overlaid eye diagrams for the received and equalized signals are given in Figure 7.

Consider Figure 8. On the left, *r(n)* is the voltage level of the center of the bit we care about. *r(n-1)* is the voltage level of the previous bit, *r(n-2)* is the voltage level of the bit before the previous bit, and so on. The voltage level of each bit is multiplied by a "tap" and each bit gets its own tap. The sum of the products of the voltages and taps result in the "equalized" voltage level *e(n)*.

In our example a logic '0' is proceeded by four logic '1's. Looking at Figure 5 we need to push the value of the '0' lower. We could choose to set $Tap_1$ = 1, and $Tap_2$ = –0.5, $Tap_3$ = –0.25, and $Tap_4$ = –0.1 so that when we add them up, the voltage of the logic '0' bit will be lower and easier to interpret as a logic '0'.

This is the idea of equalization: the taps are optimized so that the bits can be accurately interpreted. Physically, the taps correct for the impulse response of the transmission channel. Developers frequently use measurements of the S-parameters to determine the tap values.

An LFE can also be viewed as the sum,

$$
\begin{aligned}
e(n) &= \sum_{k=0}^{N-1} Tap_k\, r(n-k) \\
&= \sum_{k=0}^{N-1} f(k)\, r(n-k)
\end{aligned}
\tag{2}
$$

where I introduced the notation *f(k)* for the tap values because it will be useful later. The notation refers to digital signal processing (DSP) filters where *f(k)* indicates the coefficients of the *f*eedforward filter.

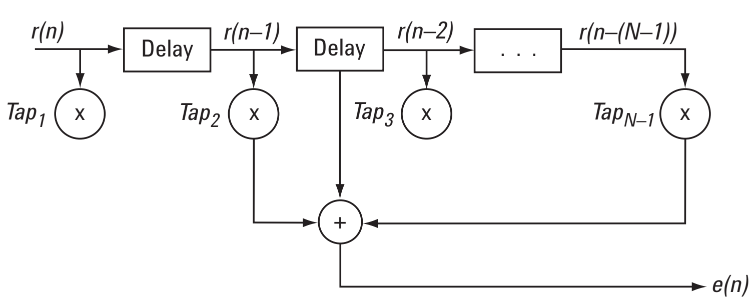


**Figure 8: A shift register description of a linear feedforward equalizer.**

## 3. Basic theory of equalization

The impulse response of a circuit contains all information about that circuit. The idea is to transmit a signal whose frequency spectrum is flat over an infinite bandwidth so that every frequency component is equally represented. In the time domain, this means transmitting an infinitely narrow, infinitely high amplitude "impulse" through the circuit [3]; that is, a Dirac-delta function, $\delta(t)$. The output of a circuit stimulated by $\delta(t)$ is precisely the impulse response, $h(t)$. The concept of impulse response is key. Think of a circuit (or for that matter *any* system) being driven by some arbitrary force. If you understand how that circuit responds to an impulse, then you can integrate the impulse response across the driving force to derive the behavior of the circuit under that force. If you have a strong mathematical background, think of $h(t)$ as the kernel of the circuit, or the solution to the Green's function of that circuit – if you've never heard of kernels or Green's functions, don't worry about it.

The idea of equalization is to process the received signal in such a way that the impulse response of the circuit is inverted. In the ideal case an input $s(t) = \delta(t)$ results in a received signal $r(t) = h(t)$ which is processed by the equalizer to reproduce the original impulse, $e(t) = \delta(t)$. What all this means is that if we can measure the impulse response then we can derive the equalizer.

### Derivation of an LFE from first principles

The formal derivation of the LFE is at once illuminating and elegant.

The transfer function, $G(s)$, is the Laplace transform of the impulse response,

$$\begin{aligned} G(s) &= \mathrm{L}[h(t)] \\ &= \int_0^\infty e^{-st}\, h(t)dt \end{aligned} \qquad (3)$$

Where $s = j\omega + \alpha$ is the Laplace parameter (not to be confused with the signal, $s(t)$ – the context of the two will distinguish them). To keep things simple, we'll ignore random noise at first. Since it contains the same information as the impulse response, the transfer function describes how a signal is affected by a circuit.

If $S(s)$ is the Laplace transform of the signal, then the Laplace transform of the received signal is given by

$$G(s)S(s) = R(s). \quad \text{(ignoring random noise)} \qquad (4)$$

The advantage of using the Laplace transform is in the simplicity of Eq. (4). In the time domain, the received signal is given by the convolution of the impulse response and the transmitted signal.

The main point is that ISI is contained in $G(s)$ and the ideal equalizer is the inverse of the transfer function, $G(s)^{-1}$ as you can see by operating on Eq. (4) with $G(s)^{-1}$,

$$G(s)^{-1}G(s)S(s) = G(s)^{-1}R(s) = S(s). \quad \text{(ignoring random noise)} \qquad (5)$$

To get to the time domain, we invert the process,

$$L^{-1}[G(s)^{-1}R(s)] = g_{inv}(t) * r(t) = s(t) \quad \text{(ignoring random noise)} \qquad (6)$$

where

$$g_{inv}(t) * r(t) = \int g_{inv}(u)r(t-u)du \quad \text{(ignoring random noise)} \qquad (7)$$

is a convolution. (It is worth noting, though somewhat annoying, that $g_{inv}(t)$ is the inverse Laplace transform of the reciprocal of the transfer function and, while $L^{-1}[G(s)] = h(t)$, $L^{-1}[G(s)^{-1}] = g_{inv}(t) \neq h(t)$.)

The LFE emerges from Eq. (7) by converting the convolution from a continuous to a discrete form,

$$g_{inv}(n) * r(n) = \sum_{k=0}^{N-1} g_{inv}(k)r(n-k). \quad \text{(ignoring random noise)} \qquad (8)$$

Equation (8) reproduces the shift register result that we got in Section 2. Comparing Eq. (8) and Eq. (2), you can see that the best taps are given by $f(k) = g_{inv}(k)$.

### Including random noise

In going from continuous, $g_{inv}(t)$, to discrete, $g_{inv}(k)$ the number of taps went from infinite to about N ~ 5 – which makes a big difference. The **matched filter bound** (MFB) is the maximum possible signal to noise ratio when an equalizer exactly cancels the ISI. In going from Eq. (3) to Eq. (8) ignored the effects of random noise, here's the equalized signal including the noise term, $w(n)$, and the LFE taps, $f(k)$,

$$e(n) = \sum_{k=0}^{N-1}\sum_{i=0}^{\infty} f(k)h(i)s((n-k-i) + \sum_{k=0}^{N-1} f(k)w(n-k) \qquad (9)$$

Equation (9) is a little complicated, so lets go through each term. The first term is the ugliest, the product $h(i)s(n-k-i)$ in the sum represents the convolution, $h(t)*s(t)$, which gives the received signal but with random noise ignored. Applying the LFE filter completes the first term and gives the equalized signal, but again with noise ignored. The second term, $w(n)$, is the voltage level of the random noise on each cursor; thus, the second term is the result of the LFE operating on the noise.

If $|f| > 1$, it's possible for the filtered noise to be larger than the unfiltered. **Noise gain** occurs when an equalizer amplifies the noise.

## Summary of the LFE

A perfect equalizer would remove all ISI leaving just the signal and filtered noise.

Generally an LFE:

- Is discrete - usually just one tap per bit, but ISI is continuous
- Is finite - not long enough to completely correct the impulse response
- Only uses information from the current and previous bits
- May result in noise gain

A way to improve on the limitations of an LFE is to introduce a delay so that both pre and post-cursors can be used in the correction. One can also include another set of corrections that are based on the best guess of the current and previous bits to further cancel ISI. The additional correction term uses the logic decision of previous bits as feedback to improve the equalizer.

# 4. The decision feedback equalizer (DFE)

A DFE uses a feedback loop of the digital signal after it has been decoded from the output of an LFE. The DFE introduces *M* additional taps that are applied to the decoded digital signal and usually includes a delay, $\tau$, between the LFE and the feedback loop.

A diagram of a DFE is given in Figure 9. The received signal enters the LFE. The output of the LFE is added to the feedback loop resulting in the equalized signal. The equalized signal is fed back through a symbol detector and delayed. The *M*-tap feedback filter, *b(n)*, is applied to the decoded symbols. The output of the feedback filter is added to the output of the LFE to yield the equalized signal.

The idea is that the LFE can only correct the ISI that spreads over its *N* cursors, the feedback loop, based on logic, can correct the rest of the ISI. The role of the LFE as a component of a DFE is slightly different than when it stands alone: the taps, *f(k)*, are still tuned to revoke ISI over its *N* cursors, but the noise gain, $|f|^2$ is minimized. As shown in Figure 10, the DFE is a combination of two shift registers.

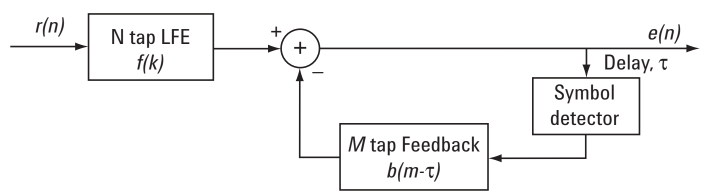


**Figure 9: Diagram of a DFE.**

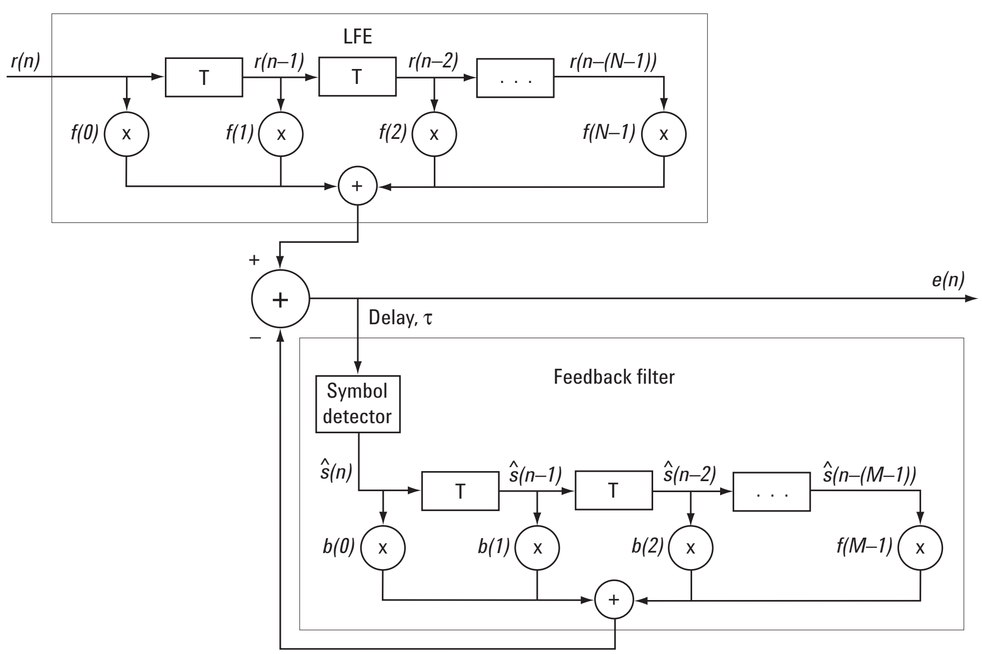


**Figure 10: The DFE is a combination of two shift registers.**

## The “ideal decision feedback” assumption

A standard assumption made in the DFE is that the symbols decoded from the equalized signal are correct. We can, in principle, write down a two term sum for the DFE.

$$e(n) = \sum_{k=0}^{N-1} f(k)r(n-k) - \sum_{M=0}^{M-1} b(m)\hat{s}(n-m-\tau). \quad (10)$$

The ideal decision feedback assumption is illustrated by the use of $\hat{s}(n)$ for the output of the symbol detector in the feedback filter delayed by $\tau$. If the ideal decision feedback assumption is valid, then $\hat{s}(n)$ is the same as the original signal, $s(n)$.

When there is an error, $\hat{s}(n) \neq s(n)$, and the error propagates back through the loop and can cause error bursts. Burst errors from violation of the ideal decision feedback assumption introduce another type of ISI. In equalization jargon, the first such error is called a primary error, and the error burst is called **error propagation**.

## How the feedback filter reduces ISI

We can write down the entire expression for a DFE. It’s kind of ugly, but staring at it helped me understand what’s going on so maybe it can help you too. To ease the pain a little bit, I’m going to assume that the delay introduced by the DFE, $\tau$, is an integer that represents a given number of bit periods. I’m also going to introduce a new quantity, $J(n)$. $J(n)$ is the impulse response of the combination of the circuit, $g(k)$, and the LFE, $f(k)$. That is,

$$J(n) = g * f = \sum_{k=0}^{L+K} g(k)f(n-k). \quad (11)$$

where $L$ is the length of the circuit’s impulse response in bit periods (in principle $L$ is infinite, in practice it’s the length of the impulse response, $g(n)$, up to the point where $g(L+1)$ is smaller than the noise level that you care about) and $K$ is the length of the impulse response of the LFE. The entire DFE can then be written, under the ideal decision feedback assumption (trust me, we don’t want to write it down in the general case!)

$$e(n) = J(k)s(n-\tau) \quad (12\text{-}1)$$

$$+ \sum_{k=0}^{\tau-1} J(k)s(n-k) \quad (12\text{-}2)$$

$$+ \sum_{k=0}^{M-1} [J(k+\tau) - b(k)]s(n-(k+\tau)) \quad (12\text{-}3)$$

$$+ \sum_{k=\tau+M}^{L+K-1} J(k)s(n-k) \quad (12\text{-}4)$$

$$+ \sum_{k=0}^{N-1} f(k)w(n-k) \quad (12\text{-}5)$$

I wrote the equation line-by-line to make it easier to discuss.

The first term (12-1) is the cursor – in the ideal case this would be the only non-zero term in the whole equation. Look at Eq. (11) and compare Eq. (8) and Eq. (2). Since the purpose of the LFE is to remove the effect of the circuit, a perfect LFE would give $f * g = 1$ for the bit of interest, $n$, and zero for all other bits.

The second term (12-2) are the pre-cursors – the contribution to the DFE output from the signal that arrived prior to the bit being corrected. Again, for a perfect LFE, all of these terms are zero.

The third term (12-3) is the modeled post-cursor – remember, $J$ includes both the circuit and LFE response. The purpose of the loopback filter is precisely to cancel whatever ISI is left over from the LFE. For a perfect filter, $b(k)$, Eq. (12-3) would be zero. This is the value of a DFE compared to an LFE!

The fourth term (12-4) is the residual post-cursor – the terms in this sum extend beyond the range that the loopback filter can correct.

The fifth term (12-5) is the filtered noise – notice that, since it operates on the digital, rather than analog, nature of the bits, the feedback filter does not affect noise. Of course were the noise to cause an error, the ideal decision feedback assumption would be violated.

## Summary of the DFE

The loopback component of the DFE cancels much of the ISI that the LFE leaves behind. The idea is for the feedback filter to cancel the post-cursor ISI by matching the taps, $b(k)$, to the combined response of the LFE and the circuit.

Using a DFE introduces more design freedom to the problem than using an LFE alone. For example, it allows one to balance the goals of canceling ISI and eliminating noise-gain by tuning the LFE taps with an optimization technique instead of trying to attain $f(k) = g_{inv}(k)$.

## 5. Tuning the taps

In Section 3 we derived the LFE from impulse response. Equation (8) indicates how tap values can be set using a measurement of circuit response. Tap values can also be derived, for either an LFE or a DFE, by using optimization techniques. The two standard methods are called least mean square error and minimum mean square error. The ability to set taps can also be dynamic; a system can perfect its own taps as conditions evolve – which is called adaptive equalization.

### Deriving taps from S-parameters

The scattering matrix consists of four complex quantities called S-parameters that quantify the frequency response and loss characteristics of a circuit [4]. The relationship between S-parameters and the impulse response is easiest to discuss in the context of their measurement on a vector network analyzer (VNA) [5]. A signal is transmitted into the circuit and the response of the transmitted and reflected signals are measured. The signal is swept in frequency over a large bandwidth (as high as 110 GHz – but a factor of two of the data rate should suffice) with constant amplitude so that the frequency response and loss characteristics of the signal are measured. In the limit of infinite bandwidth, the complex frequency response is the Laplace transform of a Dirac delta function, $\delta(x)$. Thus, the S-parameters contain the transfer function, $G(s)$, and can be inverted to get the LFE taps in deriving Eq. (8).

### Deriving taps by optimizing the eye – LMS, MMSE, and zero-forcing

Another way to calculate the best taps is to approach the problem from a perspective where all that matters is getting the best signal. In other words, all we care about is converting Figure 1b into Figure 1a. It is a simple optimization problem: we want to open the eye as wide as possible, or, equivalently, to choose taps that minimize the difference between the equalized signal, e(n), and the transmitted (or ideal) signal, s(n). That is, find the $N$ values of $f(k)$ and $M$ values of $b(m)$ so that

$$\sum_n (s(n)-e(n))^2 = \sum_n \left[ s(n) - \left( \sum_{k=0}^{N-1} f(k)r(n-k) - \sum_{m=0}^{M-1} b(m)s(n-m-\tau) \right) \right]^2 \tag{13}$$

is a minimum. This is called the least mean square (LMS) technique or, equivalently (in most of the literature) minimum mean square error (MMSE) technique.

There are many different ways to find the minimum of Eq. (13). Keep in mind that Eq. (13) is a function of $N + M$ different variables, that is, it describes an $N + M$ dimensional surface - just the sort of thing mathematicians love to confuse themselves with. Not coincidentally, function minimization is a large field in applied mathematics and there are many different approaches. The most common technique in engineering literature is the method of steepest decent. The idea is to take the gradient of Eq. (13) and go downhill until you find the bottom of a valley - and then assume it is the function's minimum. Unless it's already too late, don't write steepest descent software yourself!

The best way to solve an optimization problem like this is to use William Press' book *Numerical Recipes* [6]. In chapter ten of Press's book several different methods are described with sample software for their implementation. To save your self a lot of work, I recommend the simplex technique. It is a geometric approach that is very easy to implement in a large number of dimensions and is extremely effective at determining the global minimum of a function. The problem with steepest descent techniques is that they can stop on a local minimum or, even worse, a saddle point. The simplex technique takes more computer cycles than the method of steepest descent, but since the taps are usually static, you only have to calculate them occasionally and computer speed should never be the limiting problem.

Zero-forcing is an algebraic technique. Rather than find the $N + M$ taps from the square of the differences, one uses a signal of length $N + M$ and solves the $N + M$ linear equations in $N + M$ unknowns,

$$\begin{aligned} s(0) - \left( \sum_{k=0}^{N-1} f(k)r(0-k) - \sum_{m=0}^{M-1} b(m)s(0-m-\tau) \right) &= 0 \\ s(1) - \left( \sum_{k=0}^{N-1} f(k)r(1-k) - \sum_{m=0}^{M-1} b(m)s(1-m-\tau) \right) &= 0 \\ &\vdots \\ s(N+M) - \left( \sum_{k=0}^{N-1} f(k)r(N+M-k) - \sum_{m=0}^{M-1} b(m)s(N+M-m-\tau) \right) &= 0 \end{aligned} \tag{14}$$

In addition to being an algebraic annoyance, there are several reasons to avoid zero-forcing: First, the solution isn't unique. Different data sequences can lead to different taps; second, it's difficult to come up with a sequence of $N + M$ bits each of whose response differs enough for the equations in Eq. (14) to be linearly independent [7]; and third, it rarely yields an equalized signal with the lowest possible bit error ratio.

## Adaptive equalization

Adaptive equalizers adjust their tap values dynamically as conditions vary. In electric applications the temperature and humidity of a transmission path can change the impulse response. In optical applications conditions can dramatically alter the transmission properties of optic fibers, most notably changing the dispersive effects of polarization variations – polarization mode dispersion (PMD). As long as the changes are slow compared to the data rate, then it's not too hard for a DFE-type filter to adjust.

If a training pattern is included in the protocol, the ideal decision assumption can be used to dynamically tune equalizer taps on the known part of the signal. Even when a training pattern isn't included it is possible to dynamically tune the taps by using other types of monitors of signal quality – these are usually proprietary techniques.

The PMD example is interesting. The slightest vibration of an optic fiber changes the polarization of the light which changes the signal ISI. But the causes of PMD have timescales that are larger than about a millisecond. For a 5 Gb/s signal, five million bits are transmitted in a millisecond which should leave an opportunity to adjust the taps as conditions change.

# 6. Analyzing closed eyes

Figure 1b demonstrates that conventional eye analysis such as mask testing is impossible after a high rate signal has traversed a long transmission channel.

Since the eye is closed by properties of the transmission path – be it a backplane, a cable, or just a stretch of PCB – there are two points where the analysis of a closed eye is important: the development of an equalizer and the analysis of the signal quality that is relevant to a receiver's symbol decision circuit.

As we've shown, equalizers can be developed from S-parameters. The efficacy of an equalizer can be estimated by simulating the propagation of a signal through a transmission channel and then simulating the effect of the equalization scheme on the simulated signal. There are a couple of problems with the use of simulations. First, the character of the transmitter is difficult to simulate – the S-parameters of the transmitter are very difficult to measure. Rather, the transmitted signal itself should be captured and averaged to remove random noise. The random noise should be measured independently. With the averaged signal and the random noise both known, the simulation of the transmission channel can be expected to give a faithful estimate of the real channel. The equalization scheme applied to the simulated signal can then be evaluated.

The transmission channel S-parameters can be measured with a vector network analyzer (e.g. Agilent N5230A) or on the same equipment that you use for eye analysis, Agilent Technologies' 86100C DCA-J with advanced time domain reflectometry (TDR) analysis (Option 202). The DCA-J when equipped with Advanced Waveform Analysis (Option 201) can also provide a long ($2^{23}$ bits) averaged waveform. The random noise can be measured on the DCA-J independent of the particular waveform using the advanced jitter analysis (Option 200). The combination of the S-parameters, the transmitted waveform, and random jitter provides the data necessary to simulate an equalization scheme.

A more concrete approach to analyzing an equalization scheme is to try it on a real signal. The DCA-J has a built in LFE so that a simple equalization scheme can be applied immediately to any combination of transmitter and transmission channel with immediate results. The ability to vary the tap values and the number of taps interactively makes it easy to try different schemes.
If you're implementing a DFE or an adaptive DFE, or almost any other equalization scheme, you can use the MATLAB interface feature of the Advanced Waveform Analysis (Option 201) to implement the equalization scheme and see its effect on a real signal.

For someone developing a transmission path, say a backplane, it is important that they evaluate the backplane performance based on how the signal will be evaluated by a receiver. Since receivers at these high data rates can be expected to incorporate an equalizer, the eye-diagram output by the backplane, Figure 1b, is not relevant. Rather, the eye-diagram that follows the equalizer, Figure 7, is the appropriate signal to use for evaluating the quality of the transmission path.
Each tap of an equalizer increases the complexity of the system which increases its cost. At high data rates, it makes more sense to evaluate the quality of a transmission path based on the simplicity of the equalization scheme that must accompany it. In the example given in Section 2, only the first three taps have appreciable magnitude indicating that the backplane we used is of high enough quality that a simple three tap LFE would be adequate.

## 7. Conclusion

This should provide you sufficient background to understand how and why equalizers work and, hopefully, encourage you to experiment with your own. It's important to keep in mind that there are many types of equalizers beyond what is discussed here, each with its own preferred application. When you face a challenging signal integrity problem it's possible that you can adapt an equalization scheme from a very different application to yours.

---

[1] Jaiganesh Balakrishnan, "Bidirectional Decision Feedback Equalization and MIMO Channel Training," Ph.D. dissertation, Cornell University, 2002.

[2] John David Jackson, Classical Electrodynamics 3rd Edition, John Wiley & Sons, Inc., 1999.

[3] Eugene Butkov, Mathematical Physics, Addison-Wesley, 1968.

[4] There's a nice description of S-parameters in Agilent Application Note AN 154, "S-Parameter Design," Agilent Literature number 5952-1087, available from http://cp.literature.agilent.com/litweb/pdf/5952-1087.pdf.; another note worth reading concerns S-parameters for differential circuits, Agilent Application Note 1382-7, "VNA-Based System Tests the Physical Layer," Agilent Literature number 5988-5075EN, available from http://cp.literature.agilent.com/litweb/pdf/5988-5075EN.pdf.

[5] For a nice description of VNAs see Agilent Application Note AN 1287-2, "Exploring the Architectures of Network Analyzers," Agilent Literature number 5965-7708E, available from http://cp.literature.agilent.com/litweb/pdf/5965-7708E.pdf

[6] William Press et al., Numerical Recipes: The Art of Scientific Computing, 2nd Edition, Cambridge University Press, 1992.

[7] Linear independence is an important concept from linear algebra, if you forgot what it means, here's my short version: "a set of linear equations is independent if none of them can be written as the linear sum or difference of any of the others." If that doesn't do it for you, see Ref. [3].




**United States:**
(tel) 800 829 4444
(fax) 800 829 4433
**Canada:**
(tel) 877 894 4414
(fax) 800 746 4866
**China:**
(tel) 800 810 0189
(fax) 800 820 2816
**Europe:**
(tel) 31 20 547 2111
**Japan:**
(tel) (81) 426 56 7832
(fax) (81) 426 56 7840
**Korea:**
(tel) (080) 769 0800
(fax) (080)769 0900
**Latin America:**
(tel) (305) 269 7500
**Taiwan**:
(tel) 0800 047 866
(fax) 0800 286 331
**Other Asia Pacific Countries:**
(tel) (65) 6375 8100
(fax) (65) 6755 0042
Email: tm_ap@agilent.com
Contacts revised: 05/27/05

**For more information on Agilent Technologies' products, applications or services, please contact your local Agilent office. The complete list is available at:**

**www.agilent.com/find/contactus**




Printed in USA, September 15, 2005
5989-3777EN