AUTO-CROSSOVER WHITE PAPER

By Matthew Hersh

Abstract

The Auto-Crossover function was created to eliminate the need for a crossover cable between devices of similar functionality. Prior to Auto-Crossover, when connecting two devices of the same type, such as two Auto-Negotiating Network Interface Cards, or two Auto-Negotiating switches, the devices would not establish a link. Both devices would be transmitting Fast Link Pulses, on the same pair, and thus would not properly receive the signaling from the other device. When a crossover cable was inserted (switching pair 1,2 with pair 3,6), the devices' transmitters would then be on opposite pairs and a link could be established. The Auto-Crossover function eliminates the need for this cable by switching channels while transmitting Fast Link Pulses. Therefore, the two devices will switch channels until Fast Link Pulses are being transmitted on both pairs, complete the Auto-Negotiation function, and establish a link. Auto-Crossover is defined in the IEEE 802.3 2005 Standard within clause 40.4.4 – Automatic MDI/MDI-X Configuration.

Introduction

In [1], a state diagram (Figure 40-17 – Auto-Crossover state diagram) is used to illustrate how a device should implement the Auto-Crossover function. There are two states within this state diagram, MDI mode, and MDI-X mode. MDI mode indicates that FLPs will be transmitted as if a straight or standard pin out cable is used. MDI-X mode indicates that FLPs will be transmitted as if a crossover cable is used.

There are four major functions used in Auto-Crossover: sample_timer, A_timer, the Linear Feedback Shift Register and the Link_Det variable. The first timer, sample_timer, is defined as the minimum length of time in which a device can transmit on one channel. After sample_timer expires, certain checks are performed to determine which channel to transmit on next. The second timer, A_timer, is used to randomly reset the Auto-Crossover function. A_timer is asynchronous to the state diagram and other timers to eliminate any "lock-up" conditions that may occur from two similarly designed devices. The Linear Feedback Shift Register decides whether the device should commence transmissions on the opposite channel, or remain on the current one. The Link_Det variable is only set when the device receives signaling, and is used to tell the device that it should not switch channels. A device needs to implement these four parts in order for Auto-Crossover to function properly.

Linear Feedback Shift Register

The Linear Feedback Shift Register (LFSR) is a register that is used within the Auto-Crossover function. The purpose of the LFSR is to tell a device on which channel it should be transmitting. The values within the LFSR should be pseudo-random (based on the initial value) which should allow the device to transmit on either channel for varying amounts of time. This is important in order to decrease the possibility of two devices getting in a “lockup" situation where they are both transmitting on the same channel and switching channels at the same time.

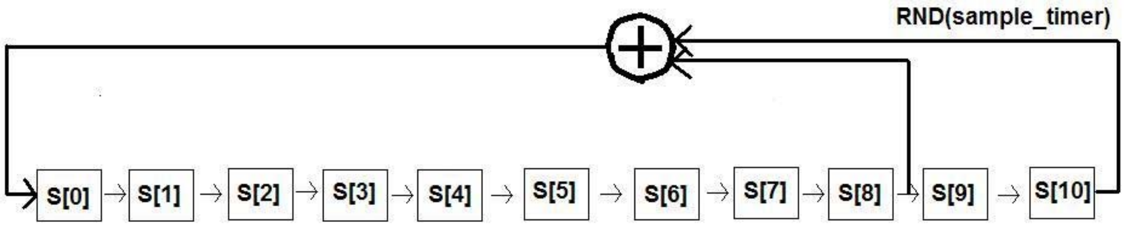

Figure 1: Linear Feedback Shift Register defined in IEEE 802.3 2005 Standard

X	Y	Result
0	0	0
0	1	1
1	0	1
1	1	0

Table 1: XOR Truth Table

The LFSR, as displayed above in Figure 1, is used within the Auto-Crossover function at the start of every sample_timer. As noted in the standard, sample_timer is defined as 62 ± 2 ms. The value of S[10], RND(sample_timer), determines the next channel in which the device should transmit Fast Link Pulses, FLPs. If the value of this bit is a zero, the device will transmit in MDI mode; if it is a one, the device will transmit in MDI-X mode. The device should remain on this channel until sample_timer has once again expired, at which time the value of S[10] will be read again. When the value of S[10] is read, all other bits within the LFSR are shifted. S[8] and S[10] are XORed, and placed into the value of S[0]. The previous value of S[0] is stored in S[1], and so on. If two consecutive values read from the LFSR are the same, the device will remain on that channel and transmit for another duration of sample_timer. This means that the device has a maximum possible opportunity to transmit on one channel for 11 sample_timers, or 64 ms (maximum sample_timer value) times 11, or 704 ms. Since XOR is defined by Table 1 above, the values within the LFSR should be pseudo-random, as devices need to load the LFSR with an initial value, which makes all future values known. It should be noted that it is only possible to transmit on the MDI channel for 10 sample_timers, as the LFSR should never contain all zeros, as this would force the device into MDI mode permanently. However, if A_timer expires after the device has transmitted for 10 multiples of sample_timer, but before switching to the MDI-X channel, the device will transmit for one extra sample_timer on the MDI channel, leaving the maximum_channel_transmit_time on either channel at 704 ms.

Link_Det with Auto-Negotiation Enabled

When a device receives signaling on one channel, it needs a mechanism to assert that valid signaling has been received. The variable Link_Det indicates to the device whether or not it should switch channels. If Link_Det=TRUE, the device must remain on the same channel in which it is currently transmitting. If Link_Det=FALSE, the device has not received signaling and it should switch channels at the appropriate time.

Link_Det is designed to check two variables, linkpulse and link_status. It checks linkpulse to detect that FLPs are being sent on a specific channel, and link_status, which is set when the appropriate link signaling is seen. Link_Det is initially set to FALSE, and once Link_Det is set to TRUE, it will remain set to true until sample_timer expires. If the device sees linkpulse=true, or link_status=READY or OK, Link_Det is set to true. When sample_timer expires, Link_Det is again set to FALSE, and the values of linkpulse and link_status are read. Since linkpulse is only set to TRUE while receiving a link pulse, the device will not see this variable set unless a pulse is being received. However, link_status maintains the value of READY or OK, therefore, Link_Det will immediately be set to TRUE when sample_timer commences. This means that once a device has a link, Auto-Crossover should never interfere with maintaining that link.

Range of sample_timer

The timer, sample_timer, defines the minimum time a device should transmit on one channel. When sample_timer expires, RND(sample_timer) is checked, and if the value is a zero the device commences transmissions in MDI mode; if the value is a one the device commences transmissions in MDI-X mode. Once RND(sample_timer) is checked and the device knows on which channel to transmit, it will start sample_timer and commence transmissions.

As defined in section 40.4.5.2 of [1], sample_timer should fall within the range of 62 ± 2 ms (60 – 64 ms inclusive). Furthermore, section 28.3.2 transmit_link_burst_timer (the spacing between transmitted FLPs) should fall within the range of 14 ± 8.3 ms (5.7 – 22.3 ms inclusive). Since sample_timer can expire at any time, transmit_link_burst_timer creates ambiguous space, which could be either before or after sample_timer has actually expired. This means that when the device has finished transmitting an FLP, and the next FLP transmitted is on the opposite pair, sample_timer could have expired anytime between the two FLPs.

Therefore, when measuring sample_timer, it is possible to observe gaps on a channel that are up to sample_timer plus 2 * transmit_link_burst_timer plus 4 ms due to the range of sample_timer, minus the difference between a minimum sample_timer and the maximum spacing between FLPs where sample_timer has not been reached. This difference can be calculated by using the following expression: n*transmit_link_burst_timer + (n+1)*burst width, where n corresponds to 4 unless the expression returns a value that is greater than 60 ms, in which case n is decreased until the expression returns a value that is less than 60 ms.

Range of A_timer

Section 40.4.5.2 of [1] requires that A_timer should fall within the range of 1300 ms ± 25%, where 25% = 325 ms, giving a range of 975 ms to 1625 ms. The purpose of this timer is to prevent the situation where two devices have the same seed values at the same point in time, thus always transmitting on the same channel, even when Auto-Crossing. 'A' within A_timer stands for asynchronous. A_timer runs asynchronously from the Auto-Crossover state diagram and other timers. When A_timer expires, and Link_Det=FALSE at that time, the device is required to restart sample_timer and begin transmissions in MDI mode. Since this timer is asynchronous, it is very difficult to observe. However, there are several cases, which make it possible to determine when A_timer does expire.

If a device transmits for less than sample_timer when transmitting in MDI-X mode, it can be determined that A_timer has expired forcing the device to MDI mode. In this case, the device commenced transmissions in MDI-X mode, then A_timer expired before sample_timer had completed, forcing the device back to MDI mode. However, this method to determine A_timer is still partially ambiguous as it is possible that the device has an invalid range for sample_timer and the value obtained was a low value for sample_timer. (Due to this effect, an exact value for the lower end of sample_timer in MDI-X mode can never be determined.)

If a device transmits for longer than the maximum allowed range as described in the sample_timer section, but less than 2 * sample_timer in MDI-X mode, it can be determined that A_timer has expired on the device. In this case, the device has transmitted for a full sample_timer, and has commenced transmissions of a second duration of sample_timer when A_timer expired, forcing the device back to MDI mode.

If a device transmits for longer than the maximum allowed range as described in the sample_timer section, but less than 2 * sample_timer in MDI mode, it can be determined that A_timer has expired on the device. In this case, the device commenced transmission of sample_timer in MDI mode, but A_timer expired before sample_timer had completed. Since A_timer forces sample_timer to restart, the device then transmits in MDI mode for another full duration of sample_timer.

Channel Hopping Due to A_timer

If a device is receiving in MDI-X mode, it is possible for the device to switch pairs, even though it is constantly receiving FLPs. If sample_timer expires, Link_Det is set to FALSE. Once signaling is received, Link_Det is again set to TRUE until the expiration of sample_timer. However, there is a period after sample_timer expires in which Link_Det will remain FALSE as no signaling has been received since sample_timer has commenced. A_timer checks the value of Link_Det, and since Link_Det=FALSE, A_timer forces the device to commence transmissions in MDI mode again. In this case, the device could be receiving constant FLPs, and appear to be acting improperly by switching channels, however, this is appropriate behavior since Link_Det remains FALSE for a brief period of time. Conversely, if a device does this while transmitting in MDI mode and switches to MDI-X, it is improper behavior since A_timer may only force a device to MDI, and not to MDI-X.

Summary

The Auto-Crossover process is a useful mechanism to eliminate the need for a crossover cable. There are several functions that aid in the implementation of this process. The timer sample_timer defines the minimum amount of time in which a device can transmit on one channel, the Linear-Feedback Shift Register is used to determine on which channel the device should be transmitting when sample_timer commences, A_timer is used to prevent the scenario in which both devices simultaneously have the same value within the LFSR, and Link_Det prevents switching once signaling is detected on one channel. These functions make Auto-Crossover an effective method to simplify the process of establishing a link.

References

[1] IEEE Std 802.3 2005 Edition – subclauses 40.4.4, 40.4.5, 40.4.6, Figure 40-14 – Automatic MDI/MDI-X linear-feedback shift register, Figure 40-17 – Auto-Crossover state diagram