Speed-Signaling in the IEEE Std 1394 Serial Bus

Prepared by Jim Skidmore

Texas Instruments Incorporated
Bus Solutions
Advanced Bus Functions Design Group

This paper gives a general overview of speed-signaling protocols, discusses timing of speed-signal transmission and reception, and discusses sampling and filtering techniques for the speed-signal.

This paper references the IEEE Std 1394-1995, and the proposed P1394a Supplement Draft 1.2.

General

Speed-signaling in a IEEE Std 1394 serial bus system occurs during the self-ID phase of bus initialization and during packet transmission in the normal arbitration phase. During the self-ID phase, peer PHYs transmit speed-signals in order to indicate their speed capabilities to each other. During normal packet transmission, a speed-signal is transmitted during the data-prefix period preceding the packet data in order to indicate the packet speed to the receiving PHYs.

The speed-signal is transmitted as a common-mode current on the TPB twisted pair, and is received as a common-mode voltage drop (relative to the TPBIAS voltage) on the TPA twisted pair.

Speed-Signaling During Self-ID

During self-ID, parent and child PHYs perform a speed-signaling handshake after the child PHY has transmitted its self-ID packet.

The child PHY will be in the S4:Self-ID-Transmit state. After transmitting its own self-ID packet (or packets), the child PHY transmits the TX_IDENT_DONE signal to indicate that the identification process is complete for that branch of the network, and simultaneously transmits a speed-signal for 100-120 ns to indicate its maximum speed-capability. The child PHY continues sending the TX_IDENT_DONE signal after speed-signaling.

While transmitting the TX_IDENT_DONE signal in the S4 state, the child monitors the received speed-signal from the parent. The highest indicated speed is recorded as the speed capability of the parent. The child PHY then transitions to the A0:Idle state when it receives an RX_DATA_PREFIX signal from its parent.

The parent PHY will be in the S2:Self-ID-Receive state to receive the self-ID packet(s) from the child. When the parent PHY receives an RX_IDENT_DONE signal from the child PHY, the parent transitions to the S3:Send-Speed-Capabilities state. In the S3 state, the parent transmits a speed-signal for 100-120 ns to indicate its own speed capability, and monitors the received speed-signal from the child. The highest indicated speed is recorded as the speed capability of the child. After transmitting its own speed-signal the parent PHY transitions to the S0:Self-ID-Start state.

Although not explicitly stated in the standard, the parent PHY must also monitor the received speed-signal while in the S2 state, as well as the S3 state, whenever RX_IDENT_DONE is received from the child. Because of resynchronization delays in repeating the packet, the parent PHY may not finish its receive actions (i.e., retransmission of the packet data and data-end signal) for up to 144 ns after the start of the RX_IDENT_DONE signal from the child. Since the child sends its speed-signal for only a maximum of 120 ns at the start of the RX_IDENT_DONE signal, the parent could miss the speed-signal from the child if it waited to enter the S3 state before sampling the speed-signal.

Speed-Signaling During Packet Transmission

When transmitting or repeating a packet during normal bus operation, a PHY sends a speed-signal during the data-prefix period which precedes the packet data in order to indicate the speed of the packet to receiving peer PHYs. This speed-signal is 100-120 ns in duration, and terminates at least 40 ns before clocked data begins. When the speed-signal is sent, the transmitting PHY will be in either the TX:Transmit state (if it is the originator of the packet) or the RX:Receive state (if it is repeating a received packet), and the receiving PHYs will be in either the A0:Idle or RX:Receive state.

A PHY begins packet transmission by sending the TX_DATA_PREFIX signal along with the speed-signal, indicating the speed of the packet, for 100-120 ns. It then removes the speed-signal and continues transmitting the TX_DATA_PREFIX signal for at least 40 ns before it starts transmitting clocked data.

A PHY begins packet reception when it receives RX_DATA_PREFIX on any port. While RX_DATA_PREFIX is being received, the PHY monitors the received speed-signal. The highest indicated speed is recorded as the speed of the incoming packet.

At a receiver, there may be a long period of data-prefix before the speed-signal is received due to arbitration activity on the bus. During arbitration, as a request from some PHY is propagated to the root and a grant is propagated back, the requesting PHY and all intervening PHYs between it and the root will send data-prefix to non-requesting child ports, which then propagates throughout the network. Thus, a PHY in a non-requesting child branch of the network may receive RX_DATA_PREFIX for a relatively long period before the requesting PHY receives the grant from the root and begins transmission.

Additionally, the data-prefix period between the end of the received speed-signal and the start of clocked data may grow longer as the packet is repeated from node to node because of mismatches between the delay in repeating the speed-signal and the delay in repeating the clocked data. (The data-prefix period between the end of the received speed-signal and the start of clocked data is always at least 40 ns, however.)

The transmitting PHY terminates packet transmission by sending either the TX_DATA_END signal or TX_DATA_PREFIX signal following clocked data. The TX_DATA_END signal is transmitted for 240-260 ns and indicates that all packet transmission is complete and the PHY is releasing the serial bus. The TX_DATA_PREFIX signal indicates that the PHY intends to transmit another concatenated packet.

The TX_DATA_PREFIX signal is sent for at least 340 ns before transmission of the clocked data of a concatenated packet. The transmitting PHY sends the speed-signal, indicating the speed of the concatenated packet, during this data-prefix period. The speed-signal is transmitted for 100-120 ns, begins at least 160 ns after the last bit of clocked data of the preceding packet, and ends at least 40 ns before the start of clocked data for the concatenated packet.

The receiving PHY monitors the received speed-signal during the data-prefix period separating packet data. The highest indicated speed is recorded as the speed of the concatenated packet. However, in order to maintain compatibility with legacy PHY designs, a P1394a compliant PHY will assume that any concatenated packet which is received without a speed-signal being received during the preceding data-prefix period is of the same speed as the preceding packet. Furthermore, a P1394a compliant PHY will repeat a speed-signal only if it receives one during the data-prefix period.

Summary of speed-signaling for packet transmission:

- A transmitting PHY (a PHY which is the originator of a packet) shall precede every packet (including concatenated packets) with the appropriate speed-signal for that packet. (For the purposes of this discussion, the S100 speed-signal is a 0V signal; i.e., there is no change in common-mode voltage.)
- For the first transmitted packet, a data-prefix signal shall be transmitted for MIN_DATA_PREFIX time (140 ns min) preceding clocked packet data. The speed-signal for the first packet shall be transmitted for SPEED_SIGNAL_LENGTH (100ns-120ns) during this time. The clocked data of the packet shall begin after a DATA_PREFIX_HOLD time (40 ns min) from the end of the speed-signal.
- For transmitted concatenated packets, a data-prefix signal shall be transmitted for MIN_PACKET_SEPARATION time (340 ns min) between clocked packet data. The speed-signal for

the concatenated packet shall be transmitted during this time. This speed-signal shall be transmitted after a CONCATENATION_PREFIX_TIME (160 ns min) from the end of the clocked data of the preceding packet and shall be transmitted for SPEED_SIGNAL_LENGTH (100 ns-120 ns). The clocked data of the concatenated packet shall begin after a DATA_PREFIX_HOLD time (40 ns min) from the end of the speed-signal.

- A transmitting node shall not concatenate an S100 packet to a higher speed packet. This restriction applies to “fly-by” concatenation as well.
- When receiving a packet (including a concatenated packet), a PHY shall record the highest indicated speed received during the data-prefix period preceding the start of clocked data as the speed of the packet. However, a PHY shall interpret a concatenated packet which is not preceded by a speed-signal to be of the same speed as the preceding packet.
- When repeating a received packet (including a concatenated packet), a PHY shall repeat a speed-signal received during the data-prefix period preceding clocked data, and shall maintain a DATA_PREFIX_HOLD time (40 ns min) between the end of the repeated speed-signal and the start of the repeated clocked data.
- When repeating a received concatenated packet, a PHY shall repeat a speed-signal received during the data-prefix period preceding clocked data, and shall maintain a CONCATENATION_PREFIX_TIME (160 ns min) between the end of the preceding clocked data and the start of the repeated speed-signal.
- A PHY shall repeat the speed-signal preceding a received concatenated packet if, and only if, a speed-signal was received preceding that concatenated packet. If a concatenated packet is received without a preceding speed-signal, then no speed-signal is repeated (preceding the repeated packet) on the other ports of the PHY.

Speed-Signal Sampling and Filtering

The speed-signal is a single-ended common-mode small amplitude signal. In an S400 capable PHY, there are generally two separate speed-signal receivers—one for the S200 speed-signal, and another for the S400 speed-signal. Each speed signal receiver consists of a differential comparator with one side tied to an internally generated reference voltage and the other side tied to the common-mode voltage of the TPA twisted pair (produced at the midpoint of a resistor divider network between the TPA+/- inputs).

Reliable reception and detection of speed-signal may be hampered by:

1) Common-mode noise. Because the speed-signal is a common-mode signal, it is more susceptible to noise than the differential arbitration and data signals. Spurious speed-signals may be received due to cross-talk, mismatches in differential signal transition times, common-mode ground noise, etc.
2) Receive comparator delay mismatch. The two sets of comparators for receiving the speed-signal may have different delays, and the delays through these comparators may be dependent upon the amplitude of the signal input. For example, when receiving an S400 speed-signal, the S200 comparator will generally switch before the S400 comparator at the leading edge of the speed-signal.
3) RC effects. The speed-signal rise and fall times may be degraded because of the RC filtering effects of the cable and bias network.

These factors require that the received speed-signal (outputs of the speed-signal comparators) be filtered in some way so that the speed-signal may be detected reliably. It is recommended that a speed-signal be present for at least 20 ns before it is regarded as valid. One mechanism for filtering the speed-signal is to sequentially sample the comparator outputs and to consider them as valid only if the outputs have been stable for some number of consecutive samples.

The following is an example c-code implementation for a speed-signal latch and filter. It uses the internal PHY clock to sample the speed-signal comparator outputs at a 50 MHz rate. The speed-signal is considered valid only if the outputs are stable (in the same state) for two consecutive samples, which corresponds to the outputs being stable for at least 20 ns.

NOTE: This implementation is intended as a typical example of a filtering mechanism, but does not preclude other implementations. For instance, a given implementation may require the speed-signal to be stable for three consecutive samples in order to be considered valid, or even to have different sampling requirements for S200 and S400 speed-signals. Of course, any given implementation must operate with the other analog and digital circuits of the PHY to meet all signal and timing requirements for the serial bus.

```
boolean     rx_S200[NPORT];         // outputs from S200 comparators
boolean     rx_S400[NPORT];         // outputs from S400 comparators

static speedCode ssq[NPORT, 2];     // save 2 speed-signal samples per port
static speedCode ss[NPORT];         // filtered and latched speed-code

void ss_filter (void) {             // continuously running routine
      int         i;
      boolean     valid_ss_time;    // OK to sample speed-signals

      repeat {
            wait_event (PHY_CLOCK_indication);  // wait for 50 MHz clock
            for (i = 0; i < NPORT; i++) {
                  ssq[i, 1] = ssq[i, 0];
                  ssq[i, 0] = rx_S400[i] && rx_S200[i] ? S400 :
                              !rx_S400[i] && rx_S200[i] ? S200 :
                              S100;

                  valid_ss_time =
                        (PHY_state == S4) ||

                        ((PHY_state == S2 || PHY_state == S3) &&
                              portR(i) == RX_IDENT_DONE) ||

                        ((PHY_state == A0 || PHY_state == A1 ||
                          PHY_state == A2 || PHY_state == RX) &&
                              portR(i) == RX_DATA_PREFIX &&
                              bus_initialization_complete);

                  // clear speed-code if not in valid state for
                  // speed-signal sampling
                  if (!valid_ss_time)
                        ss[i] = S100;

                  // if 2 consecutive samples of S200, then latch it
                  // (but only if S400 has not already been detected)
                  else if (ssq[i, 0] == S200 &&
                           ssq[i, 1] == S200 &&
                           ss[i] < S200)
                        ss[i] = S200;

                  // if 2 consecutive samples of S400, then latch it
                  else if (ssq[i, 0] == S400 &&
                           ssq[i, 1] == S400 &&
                           ss[i] < S400)
                        ss[i] = S400;

                  // otherwise previous value is retained
            }
      } until (FALSE);  // repeat forever
}
```

Note that this code latches the highest speed-code indicated during a given sampling period. The `portRspeed` function may now be redefined to simply return the latched and filtered speed-code variable:

```
speedCode portRspeed (int port_number) {
      return (ss(port_number));
}
```

Latching the speed-signal code somewhat simplifies and clarifies some of the state machine diagrams, particularly for the self-identify process. Since any received speed-signal is latched and retained in the S2, S3 and S4, states, the S3:S3a and S3:S3b transitions can be eliminated and the max_peer_speed variable for the child port can be set in the S3:S0 transition (max_peer_speed[receive_port] = portRspeed(receive_port)). Likewise, the S4:S4a and S4:S4b transitions can be eliminated and the max_peer_speed variable for the parent port can be set in the S4:A0 transition (max_peer_speed[parent_port] = portRspeed(parent_port)).