Fox-Bus (FxB) Protocol Timing (Version 4) 9/1/2011

Legend:

- The term *valid* or *reliable* means that the state has been longer than 2us in duration
- Heavy (thick) lines are periods when bus is driven by low-Z source or sink, light lines are when the bus is pulled-up via 1K ohms
- Red lines represent slave responses driving the bus low
- One time slot as generated by the master is always 70us long (max 14,286 baud)
- Timing in the slave begins when the initial pulse ends (a *valid* rising edge)
- Yellow shaded area is master window period
- Green shaded area shows when the ***slave*** window is open to detect lows on the bus
- Red shaded areas are when slave is ready to detect next falling edge of new time slot (note that slave always waits for the bus to return to a *valid* high before detecting a new time slot)
- Slave time base may vary from 0.66X to 1.5X nominal
- Round-trip delay to any device can be no more than 10us, more if the slave device has a more controlled time base.
- Dashed heavy blue lines are where the master *could* turn on the strong high drive once it has *reliably* detected that the bus has gone high
- Orange box indicates traces showing nominal slave timing
- OW slave timing is 10, 15 and 60 with 10us minimum recovery time, and gets 4-6 us of strong pull-up during recovery time
- Protocol radius limit is 1275 meters (2090 feet) at 85% cable speed (10 us round trip delay)
- Any number of low pulses can/may occur during window period, and any valid low during the window defines a zero bit
- Bus reset is signaled any low period exceeding 20us in duration and not in a time slot window
- FxB does not use a “presence pulse” but instead the master performs a single bit read after the reset to get device presence. This read is timed so that the sample of a 1-Wire bus through a translator occurs during the valid presence pulse detect window.
- FxB is driven high 65 out of 70 us during Write One bits
- FxB is driven high 50 out of 70 us during Write Zero bits
- FxB is driven high 30-40 out of 70 us during Read bits
- After 65us point, if no new time slot is to be generated, the bus may idle at 1K pull-up or may be driven to high level with protection against bus shorts in hardware or firmware. The 1-Wire bus always idles in a 1K pulled-up state so that arriving iButton devices may present presence pulses, although brief periods of strong pull-up are used to speed rise times.
- In the slowest slave, the execution time between bits is 5us, which is only 12 instruction cycles at 10Mhz. The master may allow more time between bits or bytes so that slaves can respond to commands or change pointers. If a 9600 baud serial port bit rate is used as a basis, then there will be 34 us between bits and 260 us between bytes. It has been suggested that FxB masters should always allow a minimum of one bit time (70 us) between *bytes* to allow for slave command processing time.

Chart Explanation

All waveforms are drawn as observed from the master end.

The timing chart shows the bus in a write mode (sending bits to slaves) and then in a read mode (reading bits from slaves). The nominal slave trace (in the orange box) shows the results of reading a zero bit from a slave which is near the master and has a nominal time base. The trace above shows the return of a zero bit by a nearby slave that has time base at the limit faster than nominal, and the trace below shows the return of a zero bit by a nearby slave that has a time base at the limit slower than nominal. The next trace shows the return of a zero bit by a maximally slow slave device that is also at the maximum distance from the master.

The wide variations produced by the loose tolerance of the slave time base produces other limits and effects. The area shaded in green show the times when the slave will be opening its window to look for valid pulses that indicate zero bits from the master. The area shaded in red indicates when the slave will be ready to accept a new falling edge to begin the next new time slot.

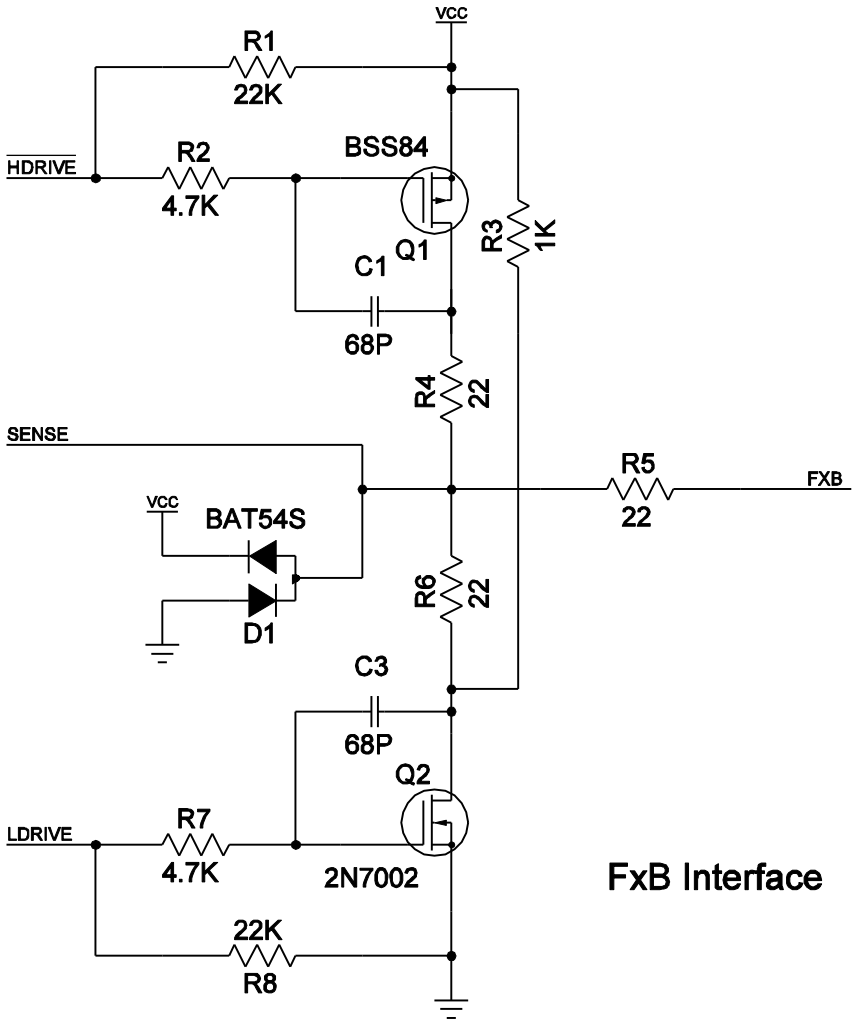
The master drives the bus with low-impedance high or low drive (app 50 ohms) during periods indicated by the thicker traces, and with a 1000 ohm impedance pull up where the traces are drawn with thin lines. Slave can only pull the bus hard low (as shown by thicker red lines).

Slaves operating on parasite power (i.e. power derived from the data line) can use a diode-and-capacitor arrangement or they can use a switch device controlled by the slave logic to rob power from the line only when the line is known to be driven high by the master (purple area).

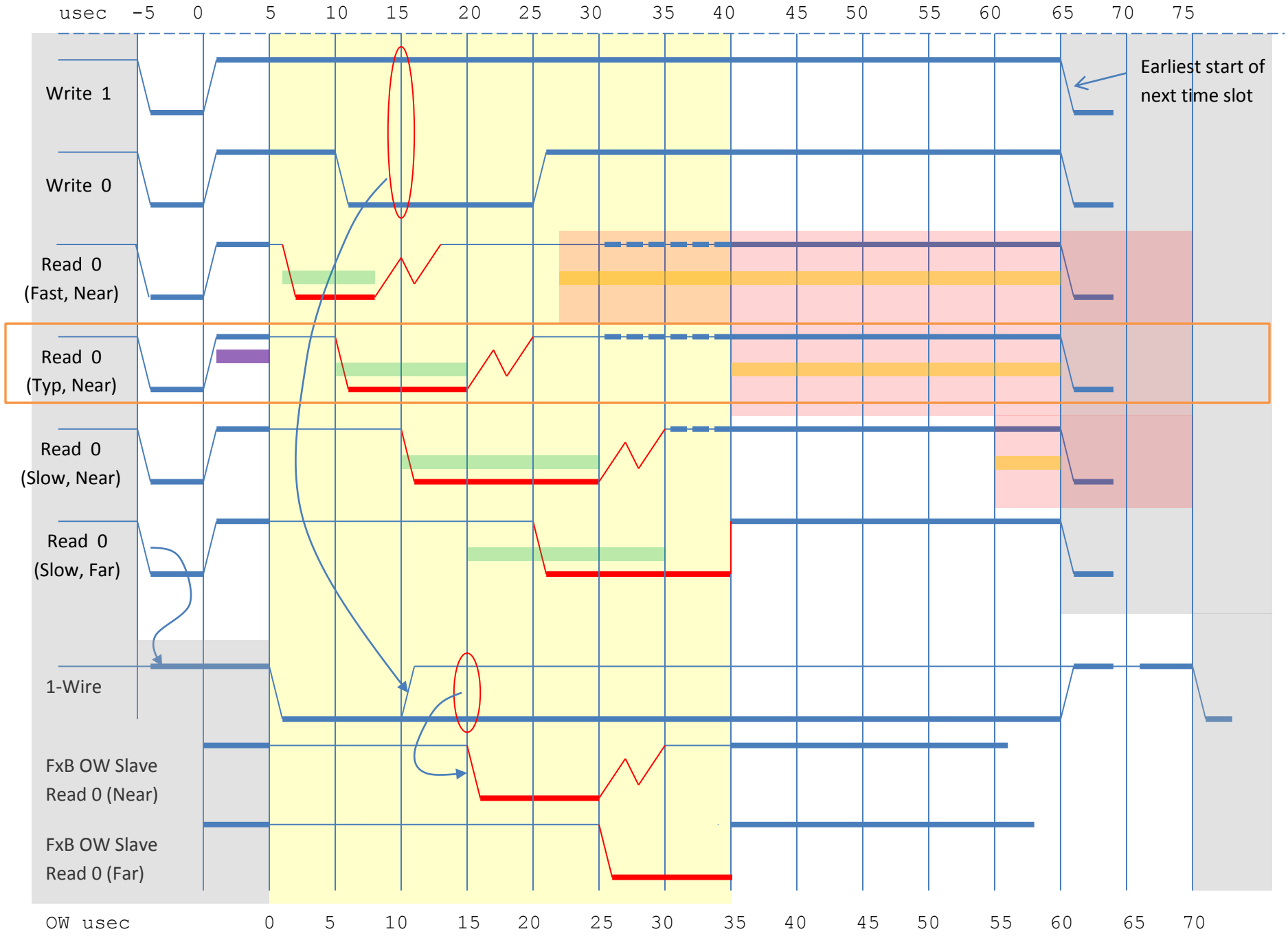
The lower traces show the waveforms produced by a translator device that supports a 1-Wire™ device, adapting it to the FxB protocol. Note that correct timing is maintained for the foreign slave device.

The bus reset sequence show how a single FxB bit is read from the bus following a bus reset pulse, and how it is timed by the master to occur at the precise time so that the sample of the 1-Wire bus will occur in the period during which the presence pulses are valid.

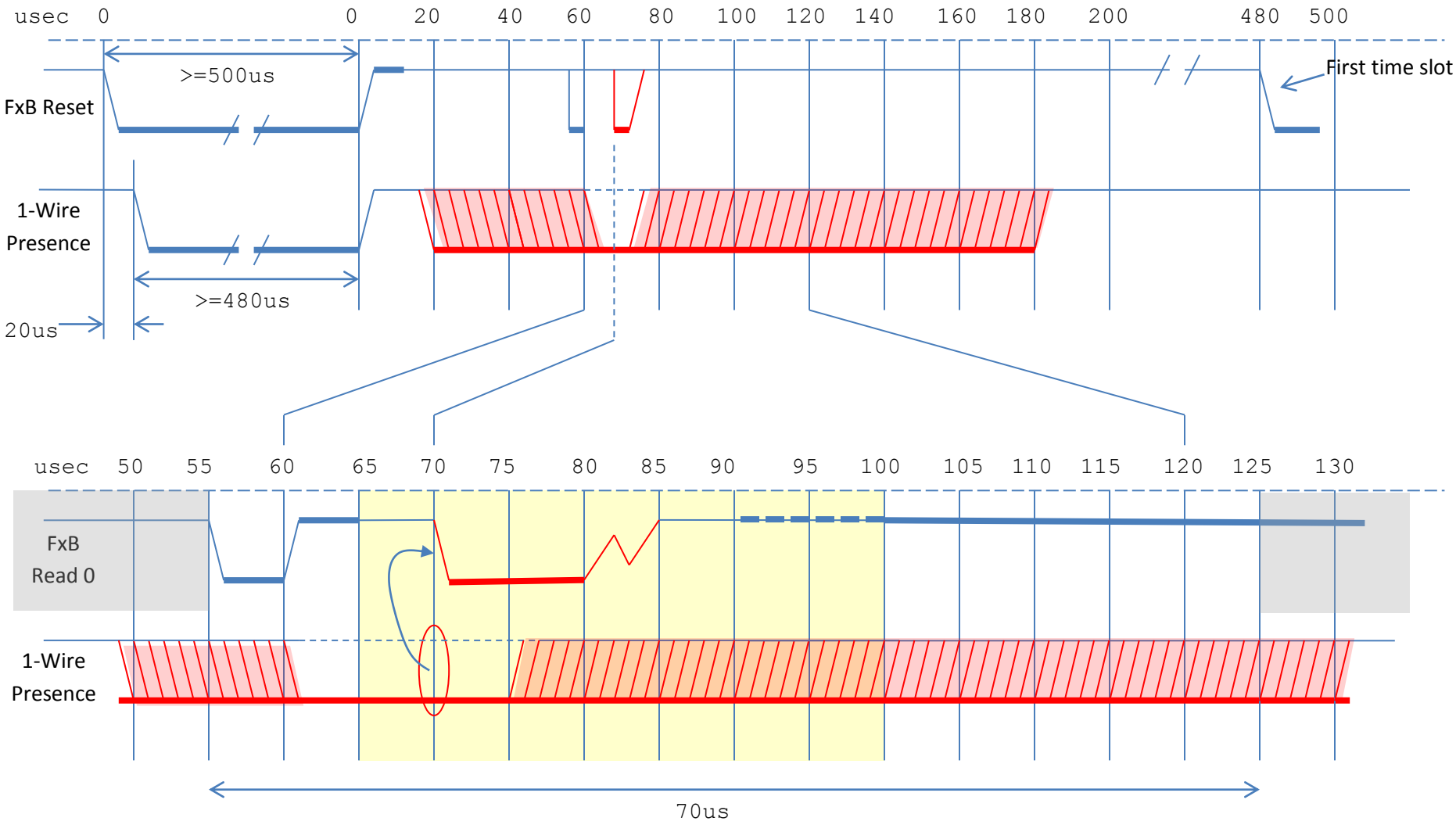
Master Physical Bus Interface



FxB Interface


usec -5 0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75
Write 1
Write 0
Read 0
(Fast, Near)
Read 0
(Typ, Near)
Read 0
(Slow, Near)
Read 0
(Slow, Far)
1-Wire
FxB OW Slave
Read 0 (Near)
FxB OW Slave
Read 0 (Far)
Earliest start of
next time slot
OW usec 0 5 10 15 20 25 30 35 40 45 50 55 60 65 70

Bus Reset Sequence



Flow Control Ideas

An idea has been discussed about ways that a slave could hold-off bus operation while it performs some required, time-consuming duty. If the low pulse from the master that begins a time slow was extended, this would drive against the master's strong pull-up and also trigger the detection of a bus reset condition.

One suggestion has been that the slave would be presented with a time slot in which it would return a zero bit, but it would extend the low pulse in the window to hold-off further bus activity. If the master waits for the bus to return to a valid high before moving on, this would be an effective way for the slave to hold-off the master.

Assuming that all slaves would similarly hold-off during this time, it could extend ad-infinitum and not cause a bus reset. Any slave that does perceive this as a bus reset is likely not involved in the conversation and so this is a non-issue.

The question is: Could this cause a legitimate bus reset to be missed by a slave device? Should a slave device perceive a legitimate bus reset as an extended bit, it would find itself out-of-sync and not properly reset when communication begins again.