COPYRIGHT

This documentation and the product it is supplied with are Copyright HUNT ENGINEERING 1999. All rights reserved. HUNT ENGINEERING maintains a policy of continual product development and hence reserves the right to change product specification without prior warning.

WARRANTIES LIABILITY and INDEMNITIES

HUNT ENGINEERING warrants the hardware to be free from defects in the material and workmanship for 12 months from the date of purchase. Product returned under the terms of the warranty must be returned carriage paid to the main offices of HUNT ENGINEERING situated at BRENT KNOLL Somerset UK, the product will be repaired or replaced at the discretion of HUNT ENGINEERING.

Exclusions - If HUNT ENGINEERING decides that there is any evidence of electrical or mechanical abuse to the hardware, then the customer shall have no recourse to HUNT ENGINEERING or its agents. In such circumstances HUNT ENGINEERING may at its discretion offer to repair the hardware and charge for that repair.

Limitations of Liability - HUNT ENGINEERING makes no warranty as to the fitness of the product for any particular purpose. In no event shall HUNT ENGINEERING'S liability related to the product exceed the purchase fee actually paid by you for the product. Neither HUNT ENGINEERING nor its suppliers shall in any event be liable for any incidental, consequential or financial damages caused by the delivery, use or performance of this product.

Because some states do not allow the exclusion or limitation of incidental or consequential damages or limitation on how long an implied warranty lasts, the above limitations may not apply to you.

TECHNICAL SUPPORT

Technical support for HUNT ENGINEERING products should first be obtained from the comprehensive Support section on the HUNT ENGINEERING web site. This includes FAQs, latest product, software and documentation updates etc. Or contact your local supplier - if you are unsure of details please refer to www.hunteng.co.uk for the list of current re-sellers.

HUNT ENGINEERING technical support can be contacted by emailing support@hunteng.demon.co.uk, calling the direct support telephone number +44 (0)1278 760775, or by calling the general number +44 (0)1278 760188 and choosing the technical support option.

Document revision	Changes
Preliminary	First written – R. Williams
01/02/06	Added new interface introduced on Virtex-4 Modules – K.Chircop

TABLE OF CONTENTS

INTRODUCTION 5
PURPOSE 5
OVERVIEW 5
SIGNAL FUNCTIONS 6
Prior to Virtex 4 6
Virtex 4 7

USER WRITE AND USER READ OPERATION 8
PRIOR TO VIRTEX 4 8
Address Phase 8
Data Phase (Write) 8
Data Phase (Read) 9
VIRTEX 4 10
Address Phase 10
Data Phase (Write) 10
Data Phase (Read) 11

APPLICATION FPGA MESSAGE SENDING 12
PRIOR TO VIRTEX 4 12
Address Byte 12
Data Byte 13
Initiating Message Transfer 14
VIRTEX 4 15
Message Transfer 15

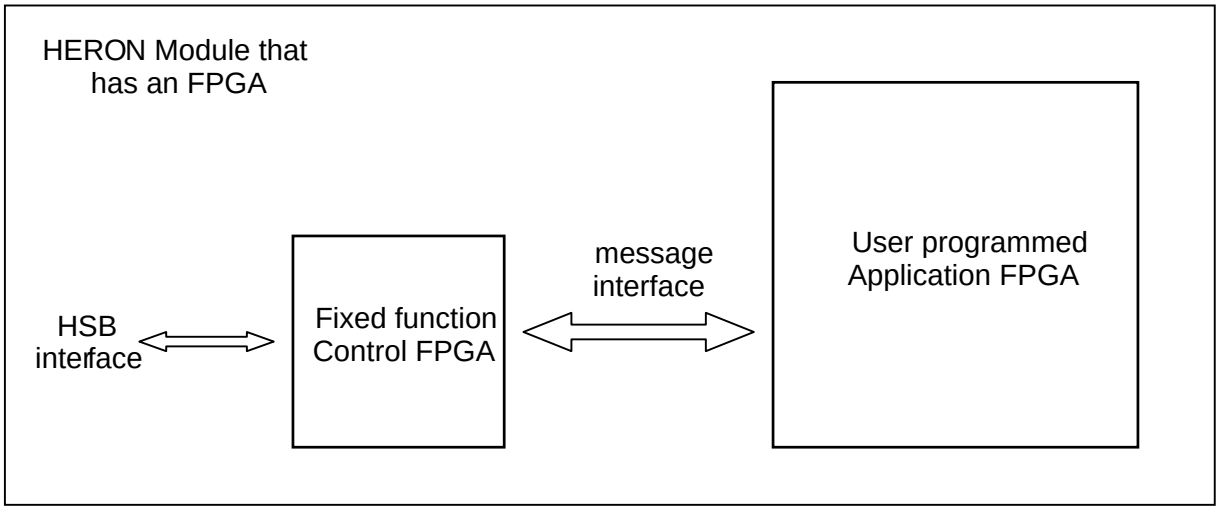
Purpose

This document describes the message interface that exists between the Control FPGA and the Application FPGA of a HERON module that has an FPGA.

The message interface performs two functions. The first function is to allow configuration of the Application FPGA via the Control FPGA from a bit-stream received over the HSB interface of the module. The second function is to allow data to be sent between the Control FPGA and Application FPGA.

The purpose of this document is to detail the operation of this second function, from the perspective of the Application FPGA. It shows how a data transfer interface should be created in the Application FPGA at the level of which signal performs which function, and the timing of each operation.

Overview



On HERON modules that have an FPGA that can be programmed by the user, there is a second fixed function FPGA (the Control FPGA) which has a direct connection to the HSB serial bus.

The Control FPGA interfaces the HSB serial bus to the user programmed Application FPGA, so that HSB messages can be used to send and receive data between the Application FPGA on the module and other devices in the system.

As shown by the diagram above, the Control FPGA is connected to the HSB interface. Messages received on this interface are interpreted by the Control FPGA, and for certain message types data is then sent to, or received from the Application FPGA using the

message interface described in this document.

The interface between the Control FPGA and Application FPGA allows data to be transferred in three ways.

a. User Write: The Control FPGA as interface master writes data to the Application FPGA.

b. User Read: The Control FPGA as interface master reads data from the Application FPGA.

c. Application Message Send: The Application FPGA sends message data to the Control FPGA that will form it into a HSB message.

The User Write and User Read operations are performed as part of the HSB message type 8 and 9 respectively. Data contained in a message of type 8 is forwarded to the Application FPGA as part of a User Write. For message type 9, data is read from the Application FPGA as part of a User Read and transmitted out over the HSB interface.

The third data transfer method is supplied to enable the application FPGA to send a simple message over HSB to another device in the system.

Signal Functions

Prior to Virtex 4

The message interface consists of 4 control signals, and 8 data signals. Each one of these signals are used to control the configuration process of the Application FPGA. Once configuration has been completed, the signals become I/Os on the Application FPGA.

The four control signals used for the message interface are the Xilinx configuration signals INIT, CS, WRITE and BUSY. The INIT signal is used as an address strobe, which we will now refer to as ASTRB. The CS signal is used as a data strobe, which we will now refer to as DSTRB. The WRITE signal is used as a read/write signal. When low, the write signal is taken to indicate a write operation, and when high is taken to indicate a read operation. The BUSY signal is used to acknowledge data transfer. It is asserted low to acknowledge transfer of one data byte.

The eight data signals used are the Xilinx configuration signals D0 to D7. These signals map directly to the eight data bits used for message data transfer.

All four control signals work in an ‘open collector’ fashion, as they may be driven both by the Control FPGA and Application FPGA at various stages of data transfer. To assert a control signal, it must be driven low. To assert a signal, it must be un-driven. If both FPGAs have removed their drive on any one signal (that is, neither FPGA is asserted that signal), then the signal will be pulled high, to a logic 1, by a resistor that is external to the Application FPGA. Therefore, interface logic in the Application FPGA must use a bidirectional I/O component for each control signal, with an output enable/disable control and no internal pullup resistor.

The data lines must default to undriven by both FPGAs. They must only be driven at the appropriate points during operation to avoid contention.

Virtex 4

The message interface consists of 12 signals. The User Write and User Read operations make use of eight of these signals. Four of these are used as control signals, the rest being 4-bit data bus. The signals are ADSTRB which is an address/data strobe, an address/data select signal which we will refer to as AD, a read/write select signal RDWR and DONE which is used to acknowledge data transfer. The 4-bit data bus is called AD_BUS.

Signals ADSTRB, AD and RDWR are input to the Application FPGA, whereas, the DONE signal is output. The data bus works in an 'open collector' fashion, as it may be driven both by the Control FPGA and Application FPGA at various stages of data transfer. Therefore, interface logic in the Application FPGA must use a bi-directional I/O component for each data line, with an output enable/disable control. The data lines must default to undriven by both FPGAs. They must only be driven at the appropriate points during operation to avoid contention.

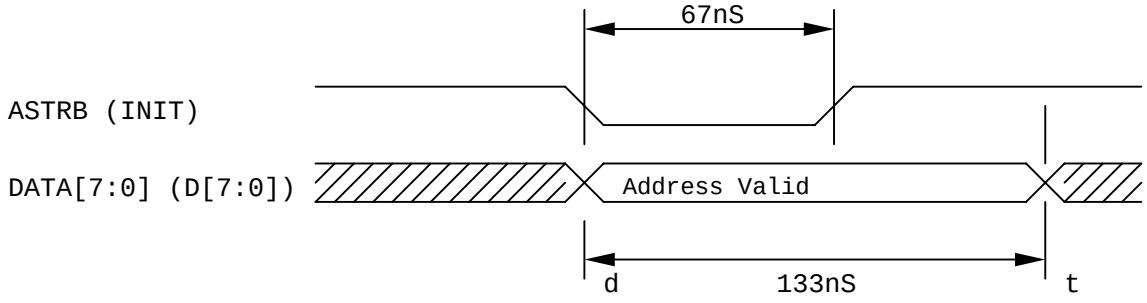
The Application Message Send operation makes use of the 4 dedicated user I/Os on the Application FPGA. One of these signals is a clock, two are used as control signals and the remaining signal is used as a data line. The clock is referred to as MCLK and the data line is referred to as MDAT. The control signals are the full flag MFF and the acknowledgement signal MACK. Signals MCLK and MDAT are output from the Application FPGA, whereas, MFF and MACK are inputs to the Application FPGA.

Prior to Virtex 4

Each message comprises of one address phase and one or more data phases. For each message, the address byte must be registered by the application FPGA. If the message contains more than one data byte, the address must be incremented internally by the application FPGA between each byte sent or received.

Address Phase

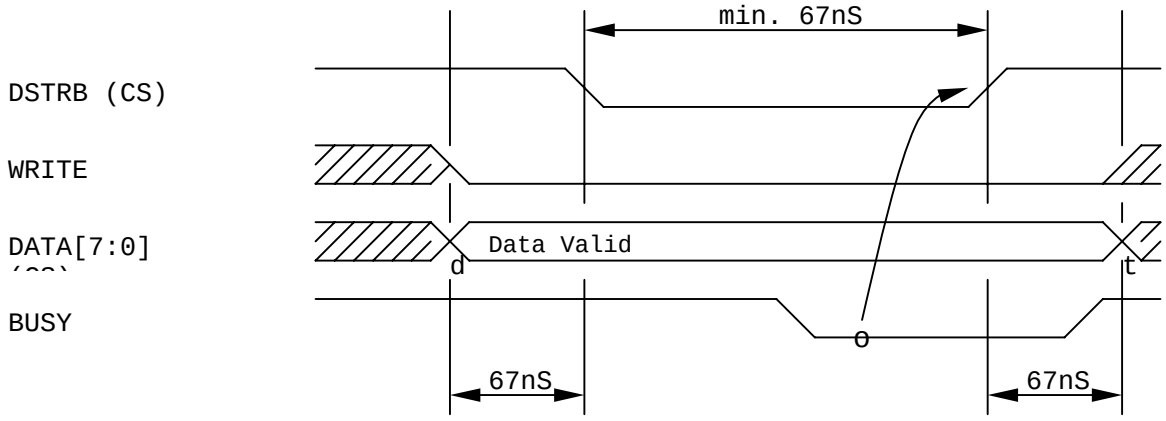
The address strobe ASTRB (Xilinx configuration signal INIT) is asserted low to indicate that a new message is being received by the control FPGA. When the address strobe is asserted, the data bus will contain the address for that access.



d = data driven by Control Fpga
t = data tri-stated by Control Fpga

Data Phase (Write)

The data strobe DSTRB (Xilinx configuration signal CS) is asserted low to indicate a data byte is to be transferred. If the data strobe is asserted while the read/write signal WRITE is low, the access is a write to the application FPGA.

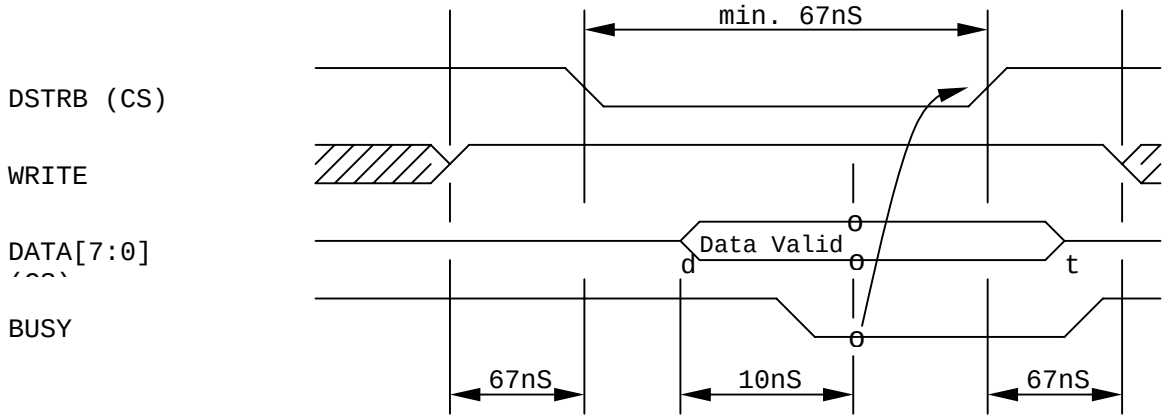


Once the data strobe has been asserted, it will remain asserted until the BUSY signal is

driven low, by the Application FPGA asserting its drive on this signal. The data should be internally registered by the Application FPGA before driving the BUSY signal low.

Data Phase (Read)

The data strobe DSTRB (Xilinx configuration signal CS) is asserted low to indicate a data byte is to be transferred. If the data strobe is asserted while the read/write signal WRITE is high, the access is a read from the application FPGA.



d = driven by Application FPGA
t = tri-stated by Application FPGA

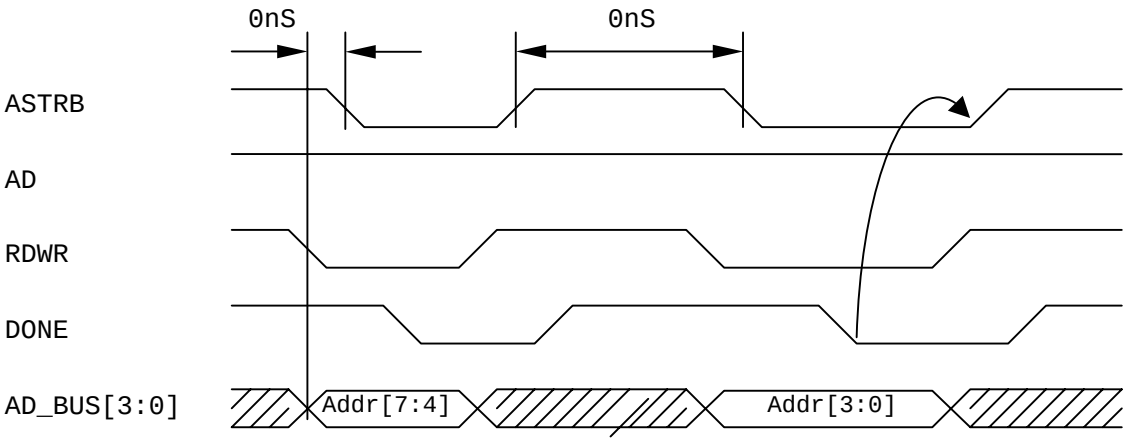
Once the data strobe has been asserted, it will remain asserted until the BUSY signal is driven low, by the Application FPGA asserting its drive on this signal. The data should be presented a minimum of 10nS before the BUSY signal is sampled low. There is 0nS hold time requirement from the point at which BUSY is sampled low by the Control FPGA.

When implementing the User Read logic in the Application, the simplest way to meet the data set-up and hold times is to ensure that the data bus is driven with the valid data a minimum of 10nS before the BUSY signal is driven low. The data should then be held until the DSTRB signal is sampled high, by which time the Control FPGA will have registered internally the value driven on the data lines.

Each message comprises of one address phase and one or more data phases. For each message, the address byte must be registered by the application FPGA. If the message contains more than one data byte, the address must be incremented internally by the application FPGA between each byte sent or received.

Address Phase

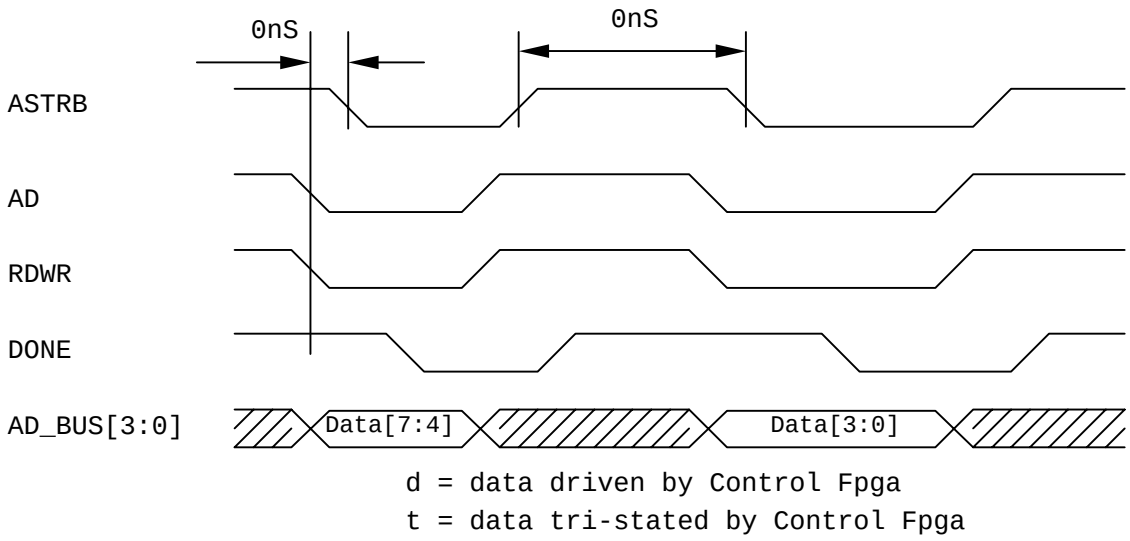
The address phase must start by driving the most significant address nibble on AD_BUS, driving RDWR low to indicate a write and AD high to indicate an address nibble is on AD_BUS. Subsequently, ADSTRB must be driven low indicating a valid address/data nibble is on AD_BUS. The nibble is qualified as an address nibble by RDWR and AD. Once the address nibble is registered, DONE must be asserted to indicate the nibble has been registered and therefore AD_BUS can be tri-stated. Following DONE being asserted, ADSTRB must be released. Once ADSTRB is driven high, DONE is released allowing the second address nibble to be transferred. The least significant address nibble is transferred in exactly the same manner as the most significant address nibble.



d = data driven by Control Fpga
t = data tri-stated by Control Fpga

Data Phase (Write)

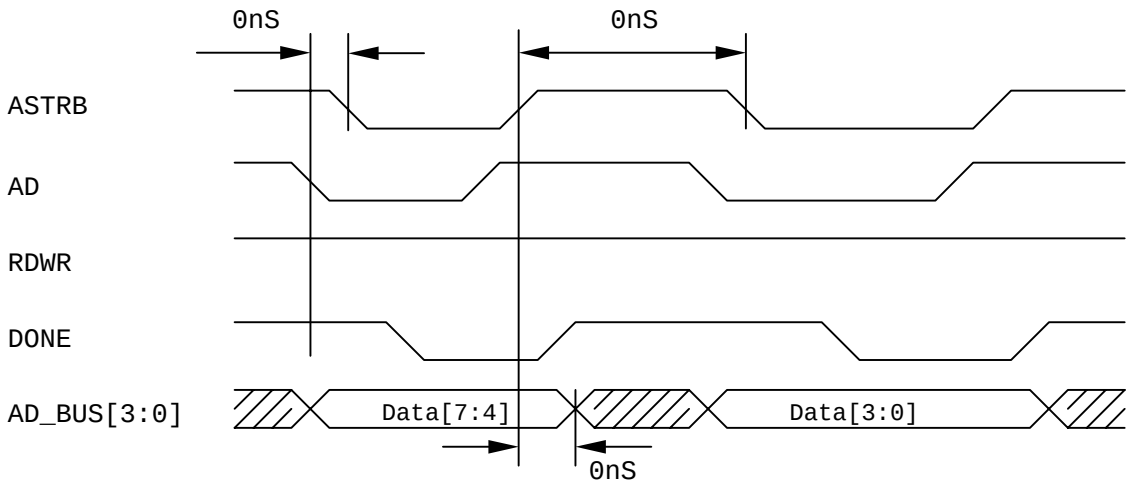
A data write to the Application FPGA must start by driving the most significant data nibble of the data byte to be transferred on AD_BUS, driving RDWR low to indicate a write and AD asserted low to indicate a data nibble is on AD_BUS. Subsequently, ADSTRB must be driven low indicating a valid address/data nibble is on AD_BUS. The nibble is qualified as a data nibble by RDWR and AD. Once the data nibble is registered, DONE must be asserted to indicate the nibble has been registered therefore AD_BUS can be tri-stated. Following DONE being asserted, ADSTRB must be released. Once ADSTRB is driven high, DONE is released allowing the second data nibble to be transferred. The least significant data nibble is transferred in exactly the same manner as the most significant data nibble.



d = data driven by Control Fpga
t = data tri-stated by Control Fpga

Data Phase (Read)

A data read from the Application FPGA must start by driving RDWR high to indicate a read and AD asserted low to indicate it is a data phase. Subsequently, ADSTRB must be driven low to request the most significant data nibble on AD_BUS. At this stage, the most significant data nibble must be driven on AD_BUS. Once the data nibble is on the bus, DONE must be asserted to indicate the nibble is ready on the bus. On DONE being asserted, the data nibble must be registered and ADSTRB must be released. Once ADSTRB is driven high, DONE is released allowing the second data nibble to be transferred. The least significant data nibble is transferred in exactly the same manner as the most significant one.



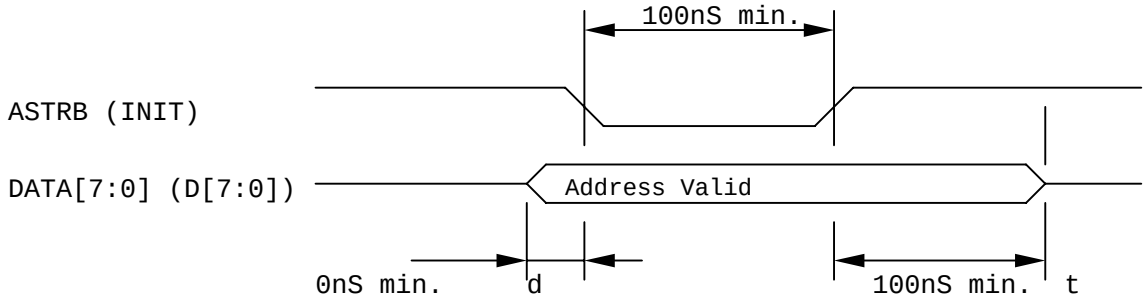
d = data driven by Control Fpga
t = data tri-stated by Control Fpga

Prior to Virtex 4

For the Application FPGA to successfully send a message over the HSB interface, it must first send an address byte that indicates the destination for the message, and then send multiple data bytes over the message interface in a particular format. In addition, the WRITE signal must be driven low to initiate message sending.

Address Byte

The address strobe ASTRB (Xilinx configuration signal INIT) must be asserted low by the Application FPGA to load the address for a new HSB message. When the address strobe is asserted, the data bus must also be driven with the correctly formatted address.



d = data driven by Application Fpga
t = data tri-stated by Application Fpga

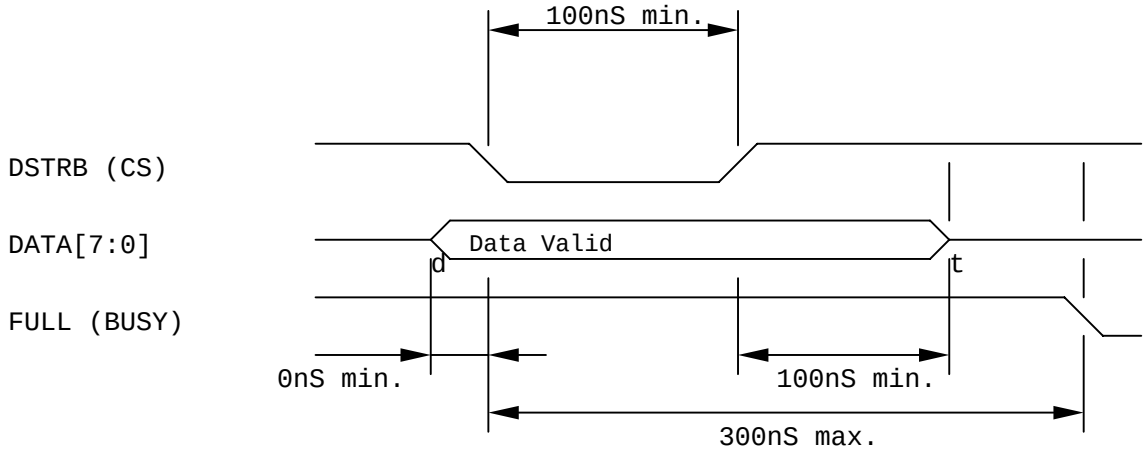
When presenting the address on the data bus, the data must be setup with respect to the falling edge of the address strobe, with a minimum setup time of 0nS. The data must be held for a minimum of 100nS from the rising edge of the address strobe.

Bit 0 of the address must always be set low, bits 3 down to 1 must contain the 3-bit slot destination, and the top half of the byte must contain the 4-bit board ID (carrier ID).

Data Byte

The data strobe DSTRB (Xilinx configuration signal CS) must be driven low by the Application FPGA to a load data byte into the message buffer of the Control FPGA. When the data strobe is asserted, the data bus must also be driven with the correctly formatted data.

The Control byte can only store one data byte at a time during message transmission. Therefore, the Application FPGA must monitor the state of the BUSY signal, which will be driven low by the Control FPGA after each byte has been loaded. The BUSY signal will stay low to indicate that the message buffer is full. No more data accesses must be made until the Full signal (BUSY signal) returns high.



d = data driven by Application Fpga
t = data tri-stated by Application Fpga

When presenting data on the data bus, the data must be setup with respect to the falling edge of the data strobe, with a minimum setup time of 0nS. The data must be held for a minimum of 100nS from the rising edge of the data strobe.

The FULL signal will be driven low by the Control FPGA a maximum of 300nS from the falling edge of the data strobe.

The first data byte that is written to the Control FPGA following the address byte must be the message type. The second byte written must be the source address. That is, the second byte must contain information indicating the board number and slot number for this module. The bottom 3-bits of the source address byte must contain the slot number, the next four bits must contain the board ID (carrier ID), and the top bit must be set to 0.

The remaining bytes will form the data section of the HSB message.

Initiating Message Transfer

The WRITE signal must be used by the Application FPGA to begin the message sending process. With the address byte and first data byte (the message type data byte) loaded in the Control FPGA message transmission can be started. This is done by driving the WRITE signal low.

When the Control FPGA samples the WRITE signal low it will begin sending the message to the address specified in the address byte. For each byte that is transmitted the FULL flag will return high to indicate the next byte can be accepted. While there are more bytes to be sent, the WRITE signal must be held constantly low.

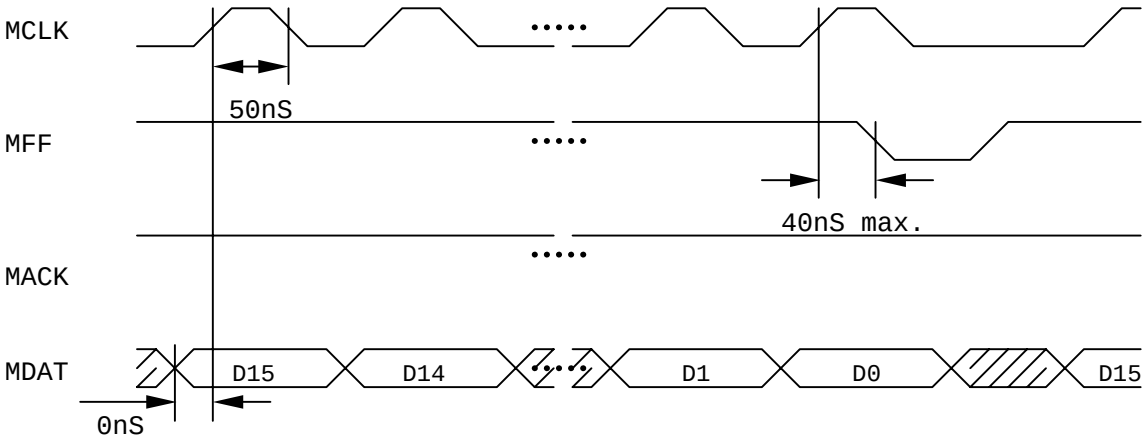
When the last byte has been sent, and the FULL flag has returned high for the last time, the WRITE signal can be released (set high).

If at any time there is a problem with sending the message then the Control FPGA will drive the INIT signal (ASTRB) low while the WRITE signal is also low. For example, if another device on the HSB bus won arbitration before the Control FPGA on this module, then the INIT signal (ASTRB) will be driven low, to indicate there was an error.

When an error condition is encountered the WRITE signal must be driven high to end the message sending process. Also, the entire message must be resent, starting by reloading the address byte.

For the Application FPGA to successfully send a message over the HSB interface, every data byte sent to the Control FPGA must be accompanied by a control byte. The 16-bit word is transmitted MSB first where bits 15 to 11 are undefined (always set to zero), bit 10 indicates the last message, bit 9 indicates the last byte in the current message and bit 8 indicates that the module's own address must be transmitted in place of the associated data byte. Bits 7 to 0 form the actual data byte to be transmitted on the Heron Serial Bus.

The first byte to be transmitted must be the address byte that indicates the destination for the message. Bit 0 of the address must always be set low, bits 3 down to 1 must contain the 3-bit slot destination, and the top half of the byte must contain the 4-bit board ID (carrier ID). The second data byte that is written to the Control FPGA is the message type. The third data byte written must be the source address. That is, the second byte must contain information indicating the board number and slot number for this module. The bottom 3-bits of the source address byte must contain the slot number, the next four bits must contain the board ID (carrier ID), and the top bit must be set to 0. The remaining bytes will form the data section of the HSB message.

Message Transfer

The message interface is idle when MCLK is driven constantly low. To initiate a message MCLK must start running and a data bit must be readily available on MDAT with every rising edge of MCLK. The data must be transmitted most significant bit first, and it must be registered with every rising edge of MCLK. After a data word of 16 bits has been transmitted, MFF must be asserted low until further data can be received. During the time in which MFF is asserted, MCLK must be driven low. Once MFF is released, the following data word is transmitted in a similar manner. This cycle continues until the end of the message. At the end of the message MACK is asserted low only if the HSB message transfer was unsuccessful. This can happen, for example, if another device on the HSB bus wins arbitration over the Control FPGA on the module at hand.