Table of Contents

HSIOOpcodeDev

v4129

Introduction

How-to write an **opcode block (OCB)** for the HSIO. Opcodes are a 16b number that specify an operation in the HSIO. When software would like the HSIO to do something it sends a block of data prefixed by an opcode. In other words, the opcode defines what the data will be used for. Along with the opcode, a 16b sequence number and 16b payload size are also sent as part of the opcode packet. As each opcode defines a specific function, we have come to use **opcode** as shorthand for a block of firmware that performs that function. Consequently **opcode-id** is sometimes used to identify the 16b number identifying an opcode. Firmware for processing opcodes lives in an opcode block (OCB). An OCB receives data from the PC (identified by its *opcode-id*), does something and sends a reply/ack to back to the PC. An OCB can send a serial stream, provide a read/write register block, collect and send data to the PC etc etc. All OCBs send **something** back to the PC, even if it's just a tiny "Ack" packet that only contains 1 word of payload (`0xACAC`). This helps the software keep track of what is outstanding and whether the HSIO has stopped responding.

An OCB has a fixed/common interface for transfering data to/from the PC, and it's hoped can be treated somewhat as a *plug-in*.

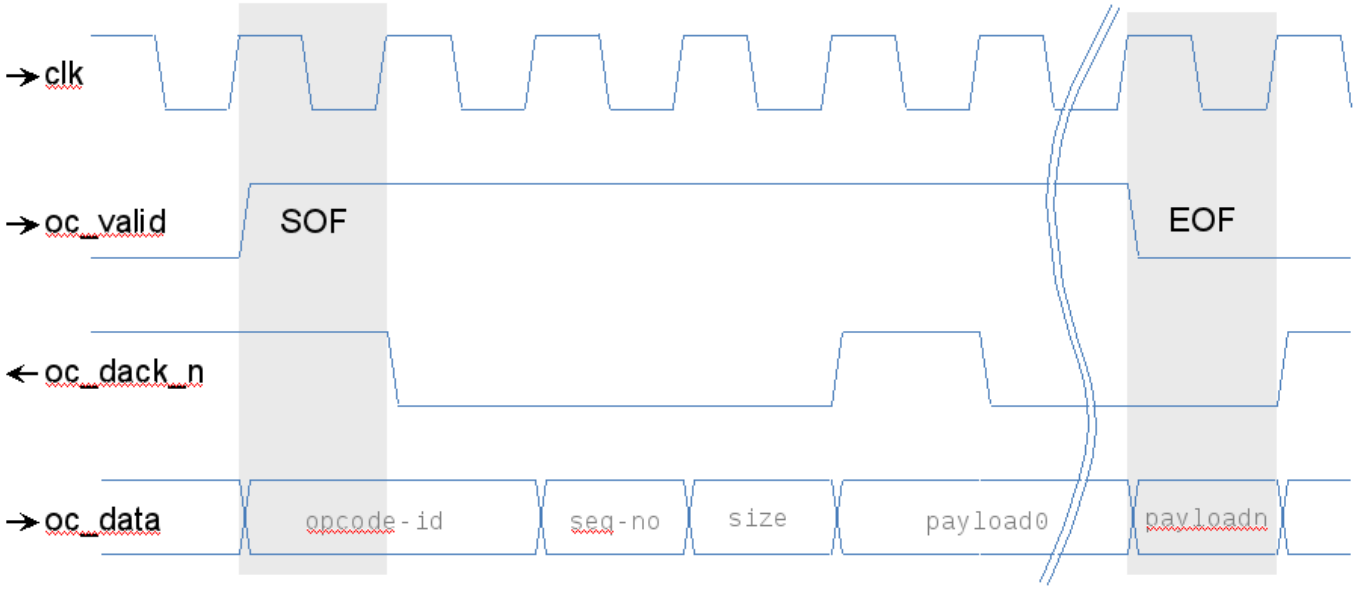
Overview of OCB operation

Data sent to the HSIO arrives in packet form. The packet is stored with transport specific headers etc. stripped. Packets start with some information about length and number of opcodes it contains. When the HSIO packet decoder is ready it requests one word at a time and fills and opcode FIFO. The opcode FIFO then feeds the "opcode bus". The opcode bus is a simplified FIFO inteface without source flow control by design. It consists of 16b `oc_data` , an `oc_valid` signal and a bussed data acknowledge signal: `oc_dack_n` .

Opcode bus operation

The packet decoder will place the received opcode number on the oc_data lines and assert oc_valid. It then waits for a `dack_n` . Each OCB has a dack_n output, these are tristates that will become ORed into a single global dack_n (a way of making a wired-OR in an FPGA). When an OCB sees the oc_valid signal transition low to high it knows the data on oc_data is an opcode, and compares it with it's own value. If there is a match it asserts dack_n. At this point all other OCBs must remain idle until the oc_valid line is deasserted. See picture below.

In the case where an opcode is un-recognised by any OCB, the echo OCB will timeout and and send the opcode back as an echo reply, but with the forst nibble of the opcode-id set ot 0xB. When an OCB is finished a task it will issue an ack packet.



All data sent to the PC needs some sort of FIFO to allow for flow-control from downstream. Data is sent from an OCB is in Xilinx LocalLink format. This is more complicated than the opcode bus as it needs to respect both source and destination flow control. To make this a little easier there are a few FIFO's written that provide for almost all needs. See `ll_ack_gen` and `ll_fifo_ack_gen` . The former has no buffering and implements a little protocol to allow variable length packets to be sent, the later is a complete 1kx16 FIFO.

Writing an OCB *(don't start from scratch!)*

Existing opcode blocks are prefixed `ocb_` in the `hsio/src/` directory. Try to select one that most closely matches your application and use it as a template. A good criteria for selecting a similar OCB is the type of data it will return to the PC - whether it will send just a tiny "ack" packet, a short fixed length reply, or a larger (or arbitary length) data block. Talk to me (Matt)!

For simulation use `rx_packet_decoder_tb`: it allows injection of packets at a usefully high-level.

Major updates:
-- MattWarren - 30-Jun-2011

Responsible: warren_40hep_2eucl_2eac_2euk
Last reviewed by: **Never reviewed**

This topic: Atlas > HSIOOpcodeDev
Topic revision: r7 - 21-Sep-2012 - warren_40hep_2eucl_2eac_2euk

Copyright &© by the contributing authors. All material on this collaboration platform is the property of the contributing authors.
Ideas, requests, problems regarding TWiki? Send feedback