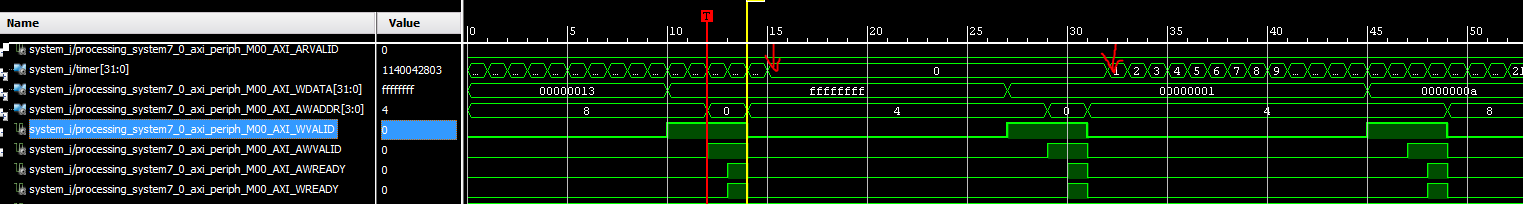
The aim of this lab is to introduce a design flow which allows you to create your own custom Intellectual Property (Custom IP) using Xilinx’s Vivado. Following this this lab introduces how to modify the generated component, by focusing on how the AXI-LITE protocol works and how it can be utilised to establish a circular data flow between the processor and the hardware component. This lab concludes on methods of maintaining and integrating this IP as part of a larger design.

The protocol





The above waveforms demonstrate the master writing 0xFFFFFFFF @ BASE\_ADDR (0x0) then a 0x00000001 @ BASE\_ADDR, finally writing 0x0000000a @ BASE\_ADDDR+4 (0x4).

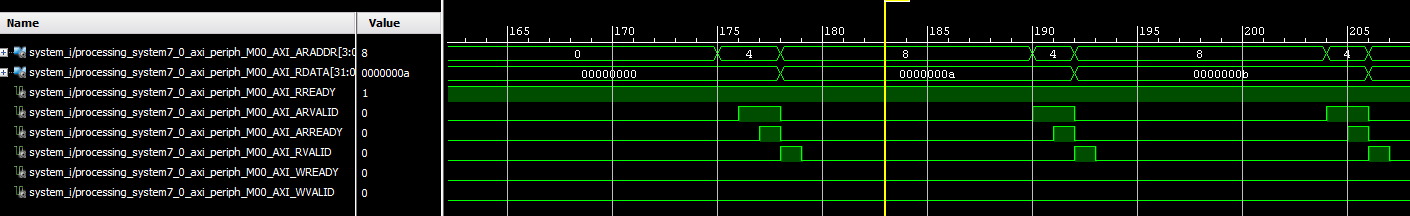
How AXI WRITES work:

* Master\* asserts WVALID (write data is valid, we can see that it is asserted the clock cycle 0xffffffff the data to be written is seen on WDATA)
* Master\* asserts AWVALID (master asserting that it has placed the valid address on the address bus, seen on AWADDR becoming 0)

\*By Master Asserts – strictly speaking it is the AXI interconnect which acts as the master for this slave AXI component, the difference between the interconnect and the processor being the interconnect multiplexes the WADDR/RADDR and only asserts the valid signals for the relevant slave (since there are often multiple slaves connected to the same interconnect)

Slave then does the following:

* Asserts AWREADY (write address can be accepted by the slave, determined by WVALID && AWVALID)
* Asserts WREADY (write data can be accepted by the slave, determined by WVALID && AWVALID)
* Also the write address is also latched (stored)
* Once WVALID & AWVALID & AWREADY & WREADY are all asserted
  + Slave write is enabled
  + Next clock cycle (14cc in diagram, yellow line) the slave register (slv\_regX) has data on WDATA bus written into it
  + Because our timer implementation reads the slave register value on the rising edge the next clock cycle (15th clock cycle) it realises the value is 0xFFFFFFFF the trigger value to reset the timer, and we can see the timer being reset here



In the above example the processor is reading from a fifo which contains the data {0x0a, 0x0b, 0x0c, …..} from the custom IP via AXI-LITE at the address of BASEADDR+4.

For AXI Reads there are 4 basic signals to worry about:

* Master pretty much always has the signal RREADY asserted, signalling it is always able to receive requested data back from the slave
* Master then places the address that it wants to read from onto the ARADDR bus and asserts ARVALID

The slave then performs the following:

* Slave the firstly asserts ARREADY (to signal that the address can be accepted by the slave, determined by ARVALID only) and then is upon this signal that the
* Slave then asserts RVALID upon which ARREADY is de-asserted and the read is completed by the master, at this point the correct read data is placed onto the bus, where it has a limited amount of time to be read by the master. The RVALID signal is then de-asserted exactly one clock cycle later.

Other signals which can be utilised but are not included in your own designs are:

* BRESP/RRESP – Write response/Read response: AXI allows for a return of value by the slave upon every write/read.
* BVALID & BREADY – used in the same nature the AXI read valid/ready signals are, with the master asserting the ready signal when it can accept a response (pretty much all the time) and the valid signal being asserted by the slave when there is valid data on the BRESP bus.

## AXI – LITE