AdcToplevel

Marc Defossez
Sr. Staff Applications Engineer

Created: June 22, 2009
Modified: December 3, 2009

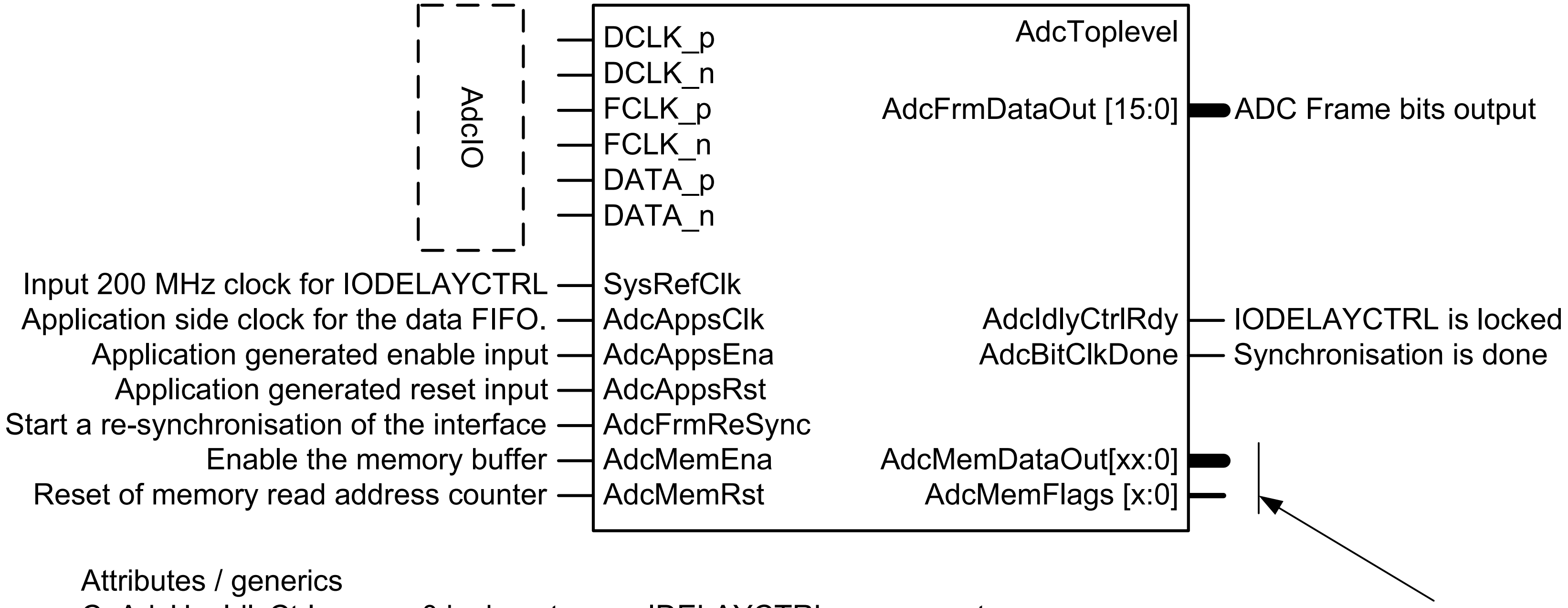
DISCLAIMER:

Contact: e-mail hotline@xilinx.com phone + 1 800 255 7778

This page is intentionally left blank.

Adc Toplevel

DATA = (C_AdcChIns*C_AdcWireInt)-1 downto 0



Attributes / generics

C_AdcUseIdlyCtrl	= 0 is do not use a IDELAYCTRL component.
C_AdcIdlyCtrlLoc	= Position of the used IDELAYCTRL.
C_AdcChIns	= Number of ADC channels in a package
C_AdcWireInt	= wire interface (1-wire or 2-wire)
C_AdcBits	= Number of bits (resolution) of the ADC

The width of these busses is calculated depending:
Number of channels, and used wire-mode.
Example:

2-wire = 2w

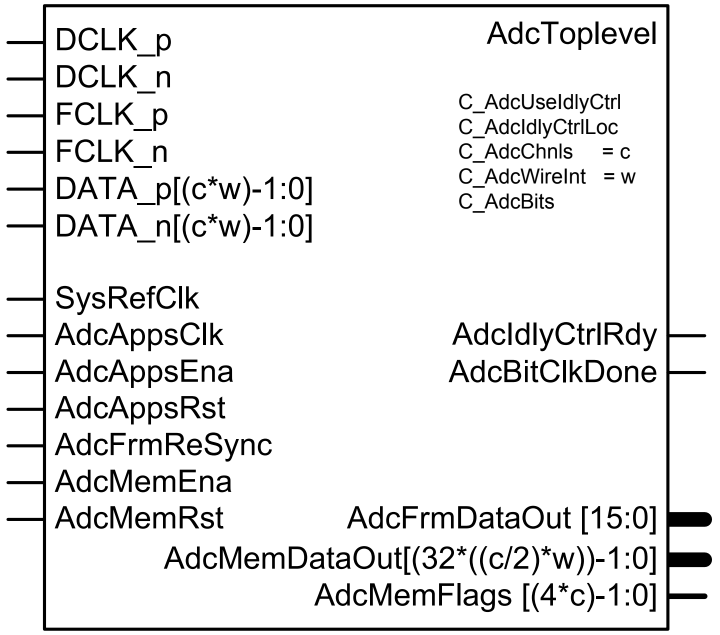
2 channels/port = 2c

Then:

AdcMemDataOut = (((32/2w)*2c)-1 : 0

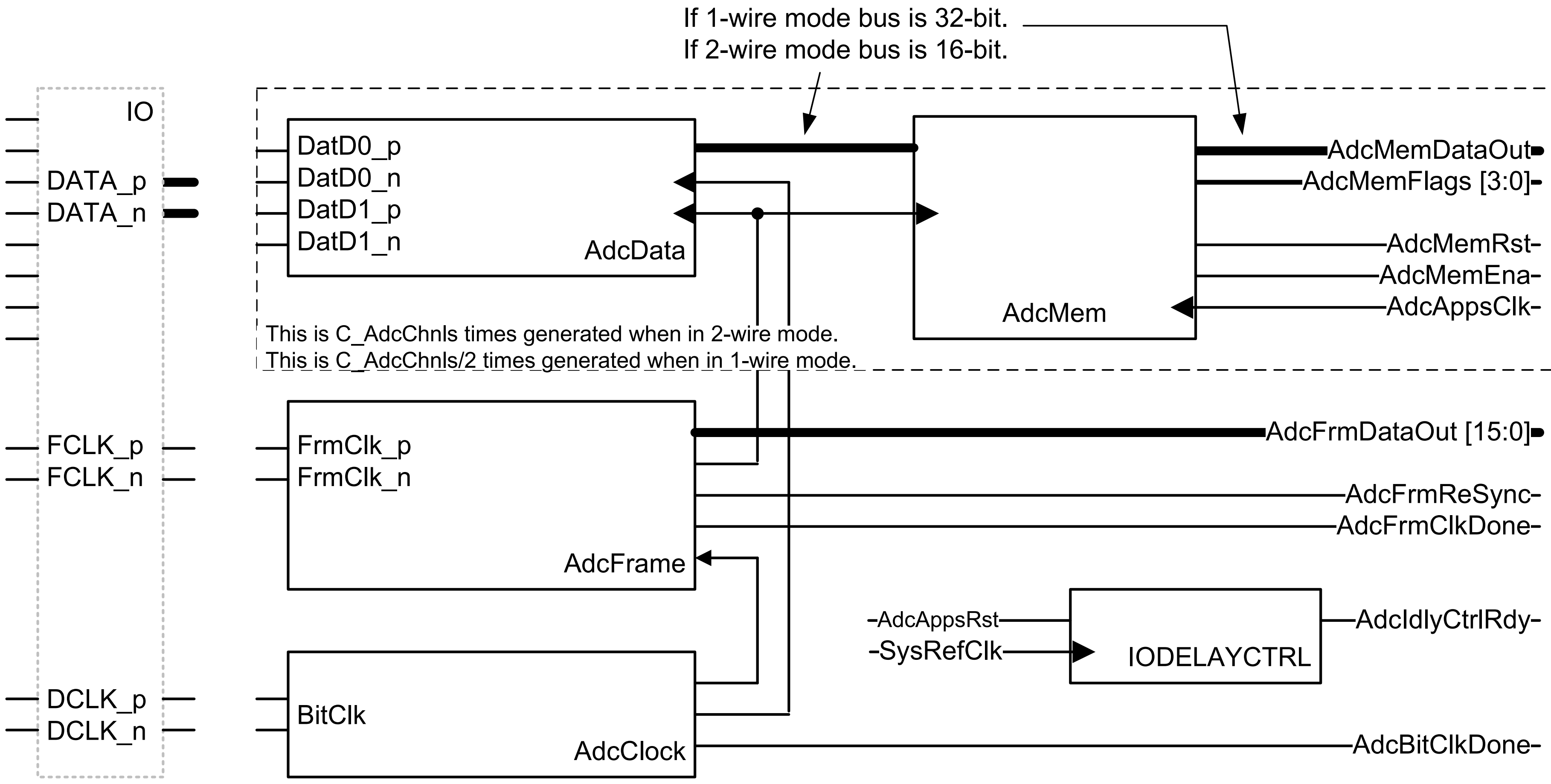
AdcmemFlags = (((4*2c)-1 : 0

Adc Toplevel Symbol



Block Level Detail

Find the schematics of these blocks in the respective sub-project directories (/Documents directory)



XILINX®

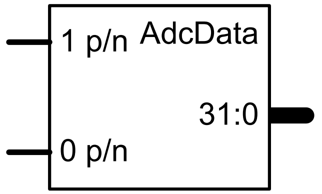
Generated Busses

It is possible to define the interface by means of generics.
That way it is possible to set:

C_AdcUseIdlyCtrl	= Use an IDELAYCTRL component or not.
C_AdcIdlyCtrlLoc	= Where the use IDELAYCTRL must be placed.
C_AdcChIns	= Number of ADC channels in a package. Normally this is the number of AD converters in a package.
C_AdcWireInt	= Wire interface (1-wire or 2-wire).
C_AdcBits	= Number of bits (resolution) of the ADC.

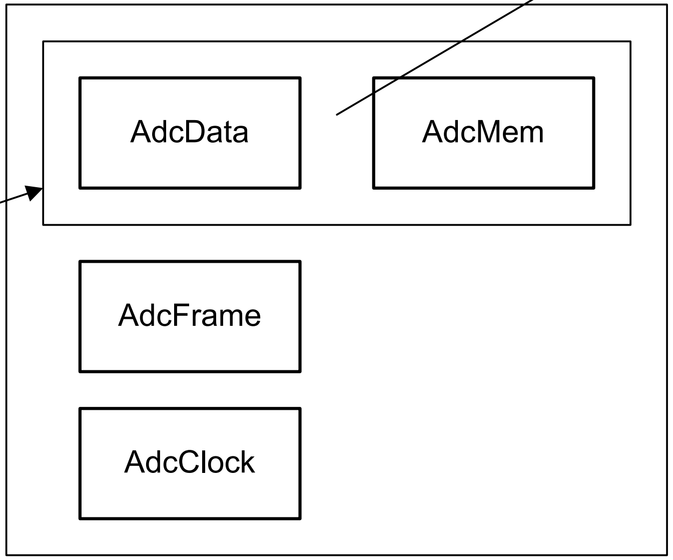
Let us make the naming somewhat shorter:

C_AdcChIns	= c
C_AdcWireInt	= w
C_AdcBits	= b



One AdcData block has two input channels.
In 1-wire mode the AdcData block represents two AD channels. The MSB 16-bit of the output bus represent channel 1 and the LSB word represent channel 0.
In 2-wire mode the AdcData block represents a single AD channel. The 32-bit output represents in both MSB and LSB words the output of the data interface.

This is “c” times generated.
The number of loops to take is calculated and represented as a value “cw”.



It is thus now the goal to automatically connect all busses in the correct order and with the correct sizes together.

View/read next pages

Generated Busses (loops)

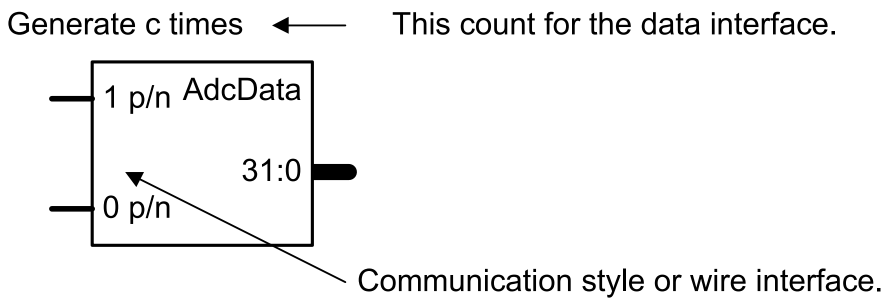
The “AdcIo” hierarchical block presents a bus of width “n”.
Where “n” is the number of n/p outputs of the needed input buffers instantiated in the “AdcIo” HDL block.
Regardless the chosen wire interface (1-wire or 2-wire) this bus will have a size “n” corresponding with the amount of _n and _p inputs.
Examples:

- Assume a 4 channel ADC used in 1-wire mode.
  - It will need 4 LVDS inputs into the FPGA.
  - The “AdcIo” block presents thus a bus of size: (3:0)
- Assume a 4 channel ADC used in 2-wire mode.
  - The data of each ADC channel is now spread over two LVDS lanes.
  - The “AdcIo” block presents thus a bus of size: (7:0)

In 1-wire mode; Bit-0 is AD channel 1 and bit-3 is AD channel 4.

In 2-wire mode; Bit-0 and bit 1 are AD channel 1 and bit-6 and bit-7 are AD channel 4.

It is thus obvious that in 2-wire mode the double of AdcData blocks are needed than for 1-wire mode.
How to represent this in a always usable and adaptable formula?


Generate c times
This count for the data interface.
1 p/n
AdcData
31:0
0 p/n
Communication style or wire interface.

The number of times the AdcData must be generated depends upon the “c” and “w” values.
The loop is represented as: “**cw = ((c/2)*w)-1 : 0**”

Examples:

c = 4, w = 2
cw = 3:0 → A 4 times loop is necessary because each block, with two inputs) represents a channel.

c = 4, w = 1
cw = 1:0 → A 2 times loop is generated because each block represents 2 channels.

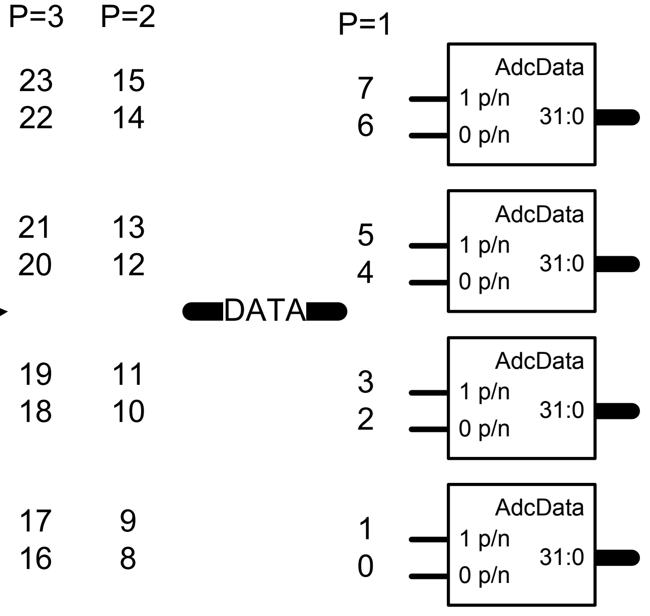
Generated Busses (inputs)

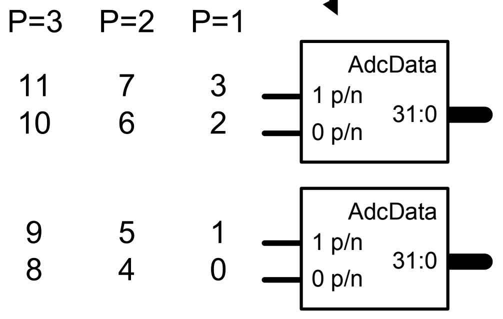
It is possible to generate "C_NmbrAdcPorts" on the AppsToplevel HDL hierarchy.
"C_NmbrAdcPorts" is represented as p.
If there is chosen to have "p" numbers of ADC devices with "c" number of channels in a "w" wire mode then the "AdcIo" interface will present a bus of "n" data width, where "n" depends from "p, c, and w".
To hook each bit correctly to the input ports of the AdcData interface, this formula is used:

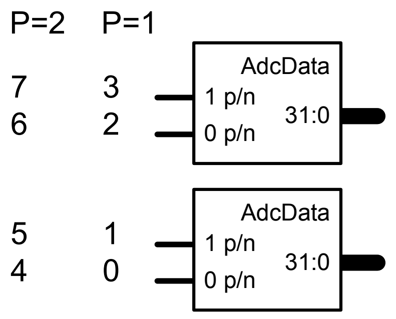
"pl" is a value in the loop range of: **p-1 : 0**
"cw" is a value in the loop range of: **(c/2*w)-1 : 0**

Op AppsToplevel this formula is used: DATA(((c*w)*p)-1: ((c*w)*p)-c*w)
In AdcToplevel this formula is used: DATA(c*w)-1:0
chnl_0 = **(cw*2)**
chnl_1 = **((cw*2) + 1**

Examples:
p = 3, c = 4, w = 2
4 AdcData blocks are generated 3 times.
p =2, c =2, w =2
2 Adcdata blocks are generated 2 times.
p = 3, c = 4, w = 1
2 AdcData blocks are generated 3 times.

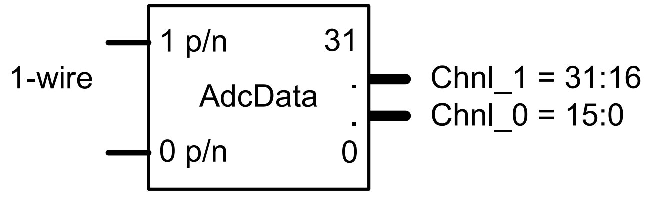


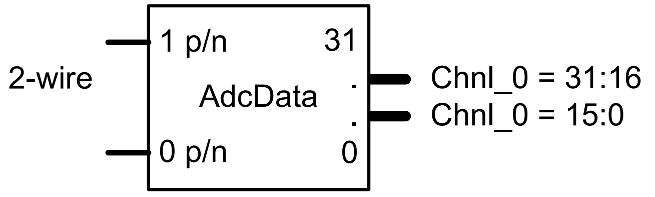




Generated Busses (data outputs)

The output of the "AdcData" block is a 32-bit bus.
In 1-wire mode the MSB word (16-bit) represents channel 1 and the LSB word represents channel 0.
In 2-wire mode the MSB and LSB word are equal.





The output bus is thus also depending on the values for "c", and "w".
The output bus must be w*c wide and depending w the whole bus must be used or only MSB or LSB must be used.
The whole bus width is represented as:

(1) (32*((c/2)*w))-1 : 0

Each section is represented for each "AdcData" block in the formula:

(2) Output data bus = ((32*((cw+1)-1 : ((32*((cw+1)-32/w)

Examples:

c = 4, w = 1

c=4,3	cw=1	**63**:32
c=2,1	cw=0	31:**0**

cw = channel loop count; (c/2*w)-1 : 0

The **bold** numbers in the examples are the calculated bus limits in formula (1).

c = 4, w = 2

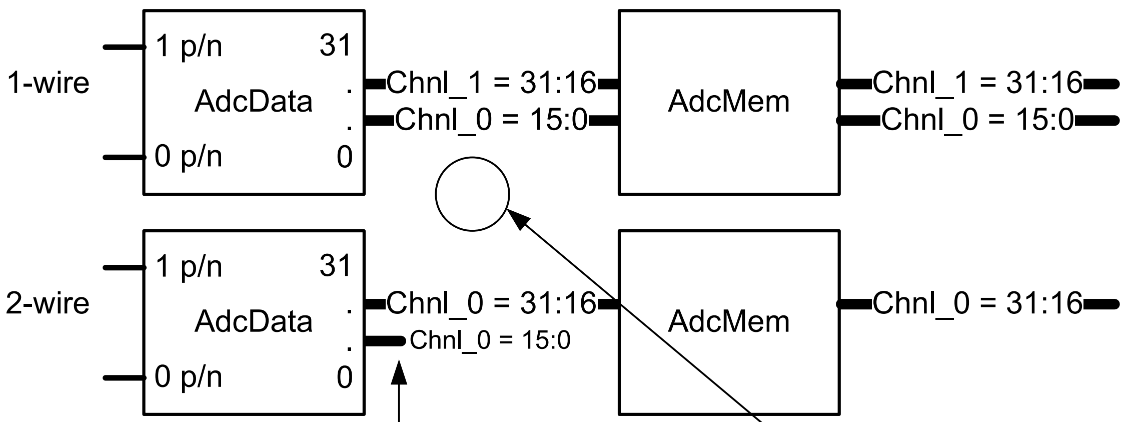
c=4	cw=3	127:112 – 111:96
c=3	cw=2	95:80 – 79:64
c=2	cw=1	63:48 – 47:32
c=1	cw=0	31:**16** – 15:0

memory

The data outputs are the same as the memory inputs because there must be as many memory block generated as there are data blocks.
The calculated busses for the "AdcData" ouputs can be used as inputs for the AdcMem blocks.

Generated Busses (memory inputs)

The output of the "AdcData" block is fed into the input of the "AdcMem" block.
The AdcMem block is formatted according the C_AdcWireInt (w parameter).
When "w" is 1 a 32-bit wide memory is generated
When "w" is 2 a 16-bit wide memory is generated.
This translates in something as the figure:



This is only for one "AdcData" and one "AdcMem" block. With the "c" values given the input and output busses of the memory will look as:

Formula used as given on previous page
Output data bus = **((32*(cw+1))-1 : (32*(cw+1))-32/w)**

cw = channel loop count; ((c/2)*w)-1 : 0

In this example with "w" = 2 the data output is a 32-bit bus with only MSB used

Example:

c = 4, w = 2

c=4	cw=3	127:112 – 111:96
c=3	cw=2	95:80 – 79:64
c=2	cw=1	63:48 – 47:32
c=1	cw=0	31:**16** – 15:0

Memory input is 15:0 when C_AdcWireInt (w) = 2.
Memory input is 31:0 when C_AdcWireInt (w) = 1.

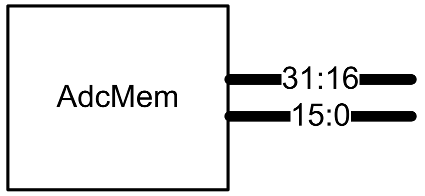
Generated Busses (memory outputs)

The memory outputs are also the ouputs of the “AdcToplevel” block.
The calculated busses between the “AdcData” and “AdcMem” block do not have a nice order.
When everything is assembled the numbering doesn’t refer as one bus.

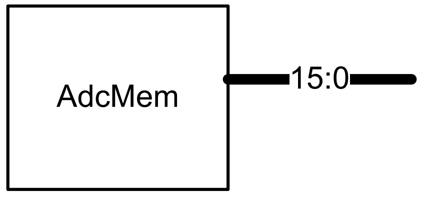
The output of the “AdcToplevel” block must have a nice looking and easy to connect bus structure.

Therefore the output bus of each generated “AdcMem” block must be nicely aligned into the output of the “AdcToplevel”

When w = 1, AdcMem output is 32-bit



When w = 2, AdcMem output is 16-bit



Output of the memories.

((32/w*(cw+1))-1 : (32/w*(cw+1))-32/w)

Examples:

c = 4, w = 1

c=4,3	cw=1	**63**:32
c=2,1	cw=0	31:**0**

cw = channel loop count; (c/2*w)-1 : 0

c = 4, w = 2

c=4	cw=3	63:48
c=3	cw=2	47:32
c=2	cw=1	31:16
c=1	cw=0	15:0

What is Used

AdcData

- Uses ISERDES components in a Master-Slave configuration for 14- and 16-bit ADC connections.
- Each “AdcData” block is one data channel from the ADC.
- The ISEDRES BITSLIP possibility is used in a sort of slave mode, the “AdcFrm” logic holds the master.

AdcFrame

- This block uses a ISERDES in master-slave configuration.
- The ADC frame signal is a slow clock signal that is phase aligned with the data. Therefore it can be used to train the receiver of the ADC data. BITSLIP is used until the correct frame data is discovered. At the same time the frame dats is searched, the data channels are shifted along.

AdcClock

- High speed incoming clock from the ADC.
- This circuit uses an ISERDES to capture the clock as data and uses a IDELAY to shift the clock up or down for adjustment of the internal clock to the external clock.
- Because the IDELAY is use the IDELAYCTRL block must also be used.

AdcMem

- Distributed RAM (LUT-RAM) is used in a small FIFO setup.
- This is done to bridge the gap between a possible phase shift of the CLKDIV clock from the BUFR (generated from the incoming high speed ADC clock) and the application clock.