XAPP594 (v1.0) August 22, 2012

Parallel LVDS High-Speed DAC Interface

Author: Marc Defossez

Summary

This application note describes how dedicated SelectIO™ interface serializer (OSERDESE2) components can be used in Xilinx 7 series FPGAs to interface with digital-to-analog converters (DACs) using serial low-voltage differential signaling (LVDS) inputs. The associated reference design illustrates the basics of the LVDS interface connection and uses a Kintex-7 FPGA as a vehicle to connect to a DAC with high-speed parallel LVDS inputs.

Introduction

Common DACs have a resolution of 12, 14, or 16 bits with possible multiple converters in a single package where each converter uses separate inputs. Each set of inputs can have one or multiple data channels, called an *interleaved data supply*. This application note explains the versatility and flexibility of the OSERDESE2.

Most of the converters use a serial peripheral interface (SPI) to set the mode of operation.

The FPGA SelectIO interfaces are configured as OSERDESE2. Each OSERDESE2 can be fed with up to 8 bits from the FPGA logic and supplies serial streams of parallel data in single data rate (SDR) or double data rate (DDR) mode to the connected DAC.

FPGA Resources

7 series FPGAs have high range (HR) and high performance (HP) I/O banks. Details about these banks are in *7 Series FPGAs SelectIO Resources User Guide* [Ref 1]. The main consideration for the DAC interface is that OSERDESE2 and ODELAYE2 components are available in HP I/O banks only. OSERDESE2 without ODELAYE2 components are available in HR I/O banks.

For convenience, the symbols with attributes of both OSERDESE2 (Figure 1) and ODELAYE2 (Figure 2) are provided in this application note. Refer to UG471, *7 Series FPGAs SelectIO Resources User Guide* for details about these components [Ref 1].

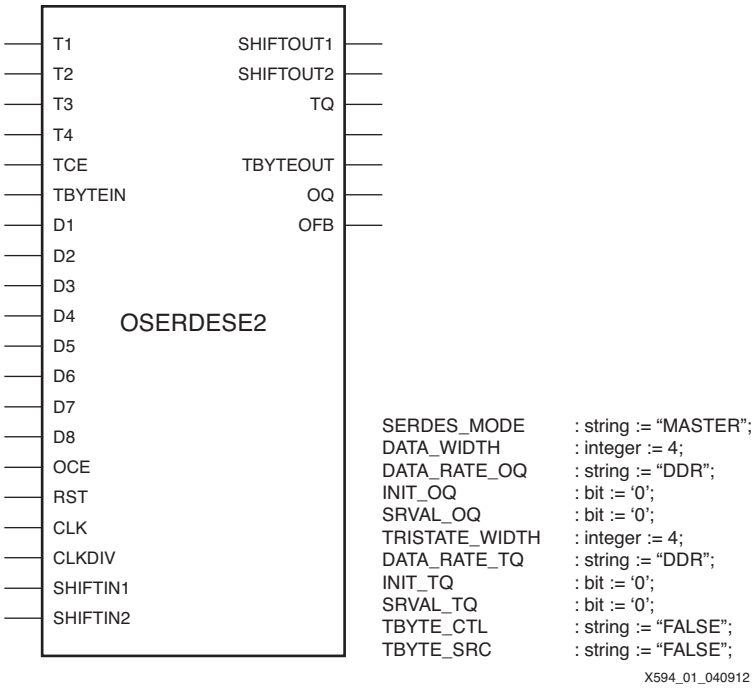

Figure 1: **OSERDESE2**

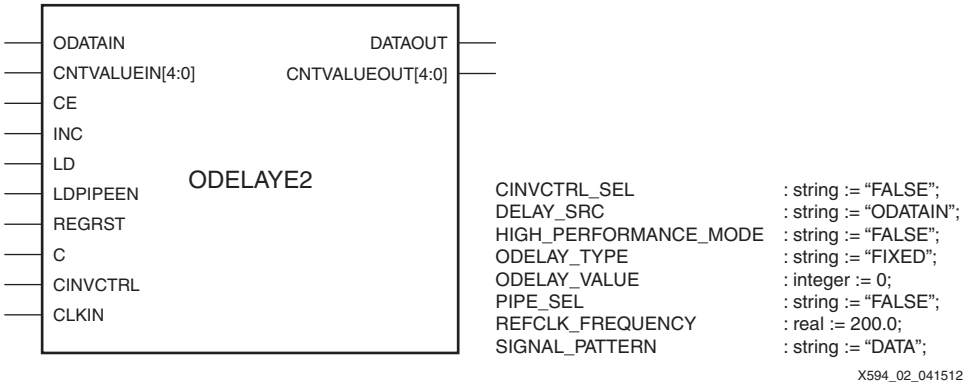

Figure 2: **ODELAYE2**

DAC LVDS Interface

Commonly, a high-speed DAC outputs a clock that is used by the interfacing component. The interfacing component, i.e., FPGA, is then required to provide data and a clock at the rate of the received clock. The data and clock from the FPGA to the DAC can be phase-aligned, or the clock can be shifted 90 degrees to the data.

Most high-speed DACs require data in interleaved format. At least two I/O banks are thus necessary. The clock from the DAC is provided through a clock-capable I/O (_CC_IO) to a mixed-mode clock manager (MMCM) in the FPGA.

This provides several advantages:

- The MMCM reduces jitter, when present, from the incoming DAC clock.
- The MMCM can provide all clocks needed for the DAC interface.
- When needed, the MMCM, through an external feedback loop, can phase-align or phase-shift (90° or other) the data provided to the DAC at the input pins of the DAC on the PCB.

Figure 3 shows a basic DAC interface setup. When the DAC resolution is more than 10 bit, this interface requires two I/O banks. An I/O bank can have up to 24 differential I/O. When the DAC resolution is 14-bit and interleaved data is required, two I/O banks are necessary.

The MMCM is placed close to the FPGA logic behind the I/O bank and spans the needed clock areas (an RLOC or LOC attribute might be needed).

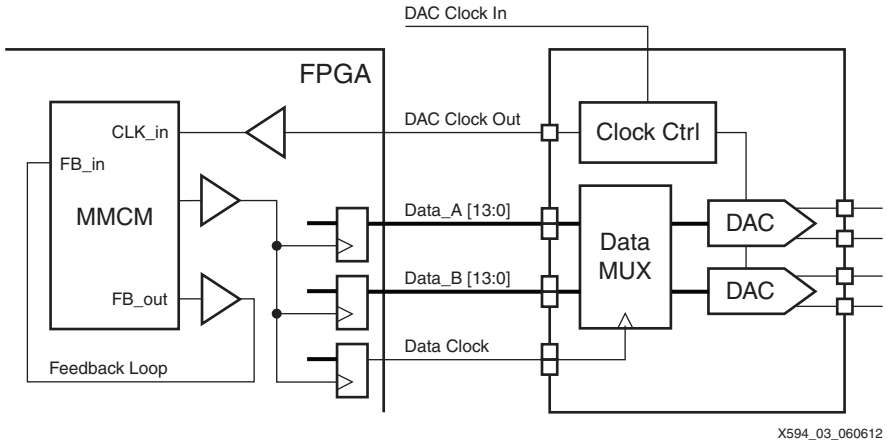

Figure 3: **Basic DAC Interface Setup**

For most DAC interfaces, an external feedback loop is not necessary. The DAC expects to get data with a phase-aligned or 90-degree shifted bit clock from the FPGA. This feedback loop can be designed using OSERDESE2 components and is described in section Bit Clock to the DAC, page 6.

Some DACs expect to get only data. That data must be monitored by the interface design up to the pins of the DAC. In that case, an MMCM feedback loop on the PCB is required (Figure 4). This can be done through an LVDS-configured I/O that is best placed in the middle of both I/O banks (at the bottom I/O of the top-oriented bank or at the top I/O of the bottom-oriented bank). The feedback path on the PCB must have the same length or twice the length, depending on the DAC used, as the data connections from the FPGA output pins to the DAC input pins. The feedback signal is taken back into the FPGA through an LVDS-configured, clock-capable I/O.

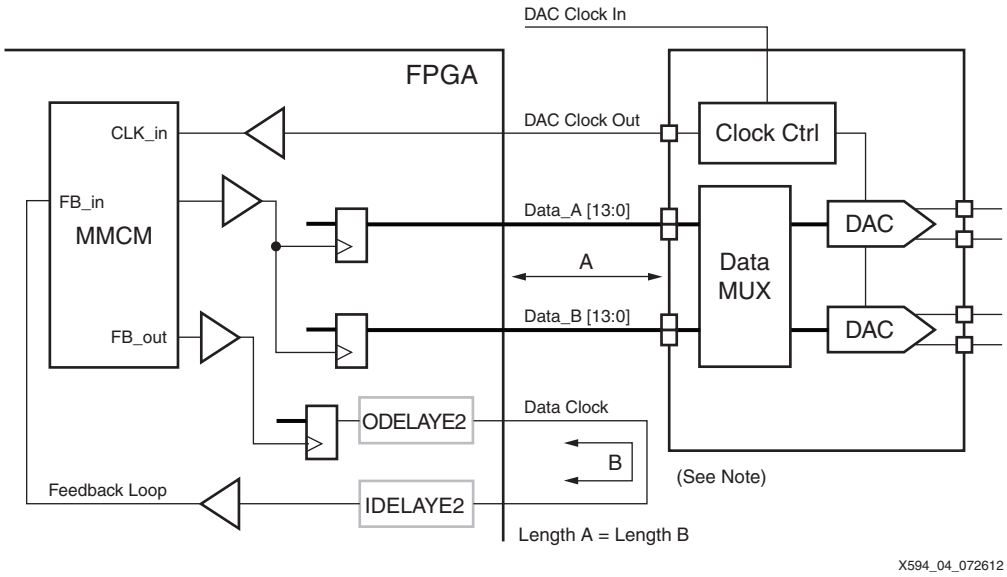

Figure 4: **Basic DAC Interface with MMCM Using External Feedback Loop**

Note: Some DAC devices have extra clock tuning features to adjust the arrival of the data with respect to the clock at the DAC input pins. An adjustable delay buffer in the DAC with clock input and output is tunable via the SPI port. That buffer can be used in the clock feedback loop instead of the ODELAYE2 and/or IDELAYE2 components.

Features such as dynamic delay adjustment can also be added in the FPGA. Figure 5 shows an ODELAYE2 in the output path of the clock feedback and an IDELAYE2 in the input path of the feedback. Either both components or one of the two components can be used. Both delay lines can be controlled by a PicoBlaze™ processor (or other processor) or by a state machine. When calibration of the MMCM feedback loop is necessary, it must be done at the initialization stage of the connection between the FPGA and the DAC.

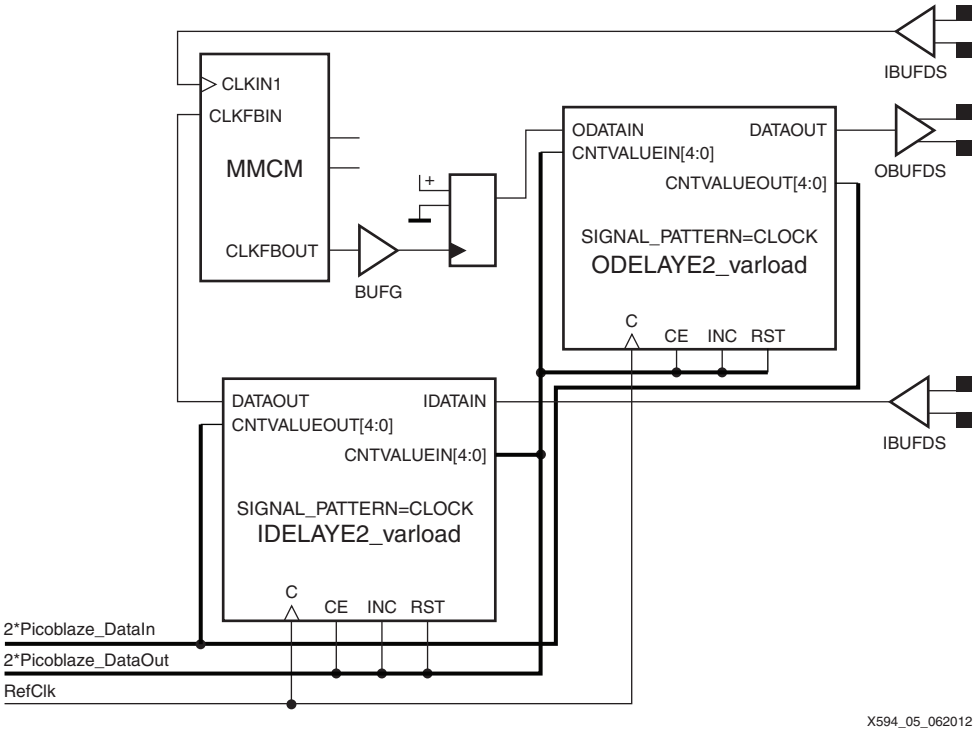

Figure 5: **External Feedback Delay Control**

Notes:

- The MMCM has an external feedback loop requirement of 3 ns or one CLKIN cycle (MMCM_$T_{FBDELAY}$).
- ODELAYE2s are only available in HR I/O banks.
- IDELAYE2 and ODELAYE2 tap values depend on the applied reference frequency, environmental conditions (voltage and temperature), and position in the delay chain.

Bit Clock from the DAC

The DAC delivers a high-speed bit rate digital clock to the FPGA. The clock is called *bit rate clock* because it is referenced to the serial output of the OSERDESE2. Many DAC devices require interleaved data from the connecting interface FPGA. This requires that the FPGA interface provides two buses of DAC resolution width. A DAC of 16-bit resolution requiring interleaved data needs two 16-bit differential buses from the FPGA, resulting in the use of two I/O banks. One bank can contain the clock input from the DAC and a 16-bit data bus and the other bank can contain the clock for the DAC and the second 16-bit data bus.

The bit clock is a digitized version of the clock the DAC uses to generate its analog output. Therefore, the clock has very low jitter characteristics.

The easiest method for PCB design is to bring the bit clock to a clock-capable I/O of the FPGA. In this way, all connections between the DAC and the FPGA are nearly straight connections and it is easier to adjust the lengths of all traces.

The bit clock can be used in the FPGA in different ways:

- Without any clock management, through BUFMR, BUFIO, and BUFR clock buffers.
- Using the received DAC clock as input to an MMCM to generate the necessary clocks for the OSERDESE2 components and/or the application in the FPGA.

Without Clock Management

In this case, the incoming clock from the DAC is routed from the clock-capable I/O input to a BUFMR clock buffer. From there, BUFIO and BUFR in each used I/O bank are controlled.

When multiple I/O banks must be controlled from a single regional clock input clock-capable I/O, a BUFMR (multi-region clock buffer) is required. The output of the BUFMR feeds the BUFIO and BUFR buffers in the I/O bank in which the BUFMR is located and in one or both (above or below) of the adjacent I/O banks (see Figure 6).

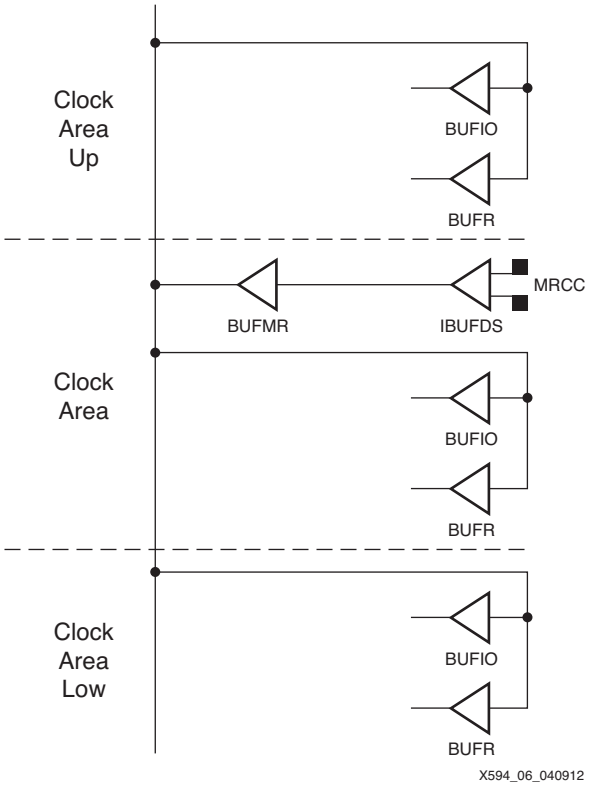

Figure 6: **Use of BUFMR Clocking**

The reference design files include an example of using a BUFMR feeding OSERDESE2 and ISERDESE2 in three I/O banks as shown in Figure 6.

Notes:

- When using the BUFMR, BUFR, and BUFIO set, make sure to LOC all components in the FPGA.
- Properly LOC all outputs (clock and data) between the FPGA and the DAC.
- Apply all guidelines for the use of OSERDESE2 (explained in OSERDESE2, page 11).
- Provide a small elastic buffer, FIFO, or data buffer between the OSERDESE2 clocked from BUFMR, BUFR, or BUFIO and the application. This data buffer allows easy clock domain

crossing between the application clock and the clocks of the OSERDESE2. Even if the OSERDESE2.CLKDIV and the application clock have the same frequency, there might be phase differences that otherwise cannot handled easily (see Figure 6).

Using the Clock with MMCM

When the clock from the DAC is connected to a clock-capable I/O and an MMCM is going to be used (Figure 7), it must pass through a BUFR to reach an MMCM. In this case, the clock from the DAC is most likely too fast for the clock network after the BUFR (see 7 series data sheet DS181, *Artix-7 FPGAs Data Sheet: DC and Switching Characteristics* [Ref 2], DS182, *Kintex-7 FPGAs Data Sheet: DC and Switching Characteristics* [Ref 3], *or* DS183, *Virtex-7 FPGAs Data Sheet: DC and Switching Characteristics* [Ref 4]).

The BUFR thus divides the incoming high-speed clock by two, and then the MMCM can feed this back to the required bit OSERDESE2.CLK and word clock OSERDESE2.CLKDIV.

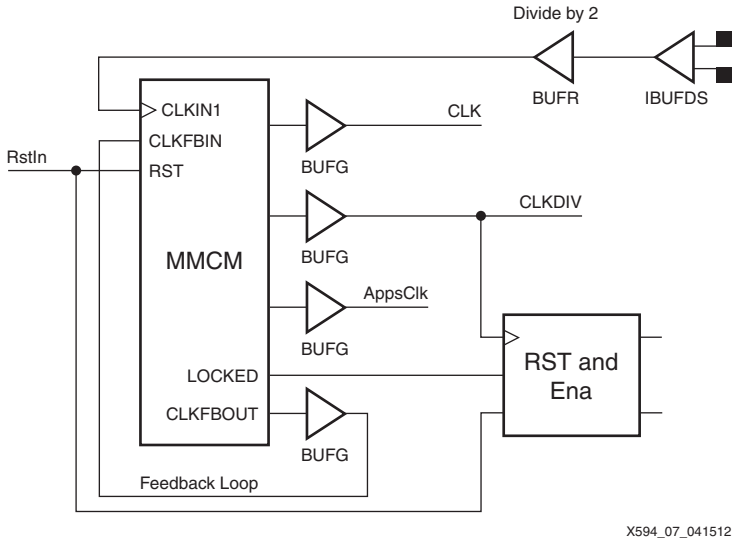

Figure 7: **Possible MMCM Setup**

Notes:

- Use two outputs of the MMCM as dedicated bit and word clocks for the OSERDESE2 (CLK and CLKDIV).
- The CLKDIV word clock from the MMCM must clock the registers in front of the OSERDESE2 or the read side of the data buffer feeding the OSERDESE2.
- Route the clocks for the OSERDESE2 through BUFG clock buffers because the DAC interface is likely to span multiple I/O banks.
- Use an MMCM in the I/O bank that captures the clock from the DAC through a clock-capable I/O. This probably requires a LOC constraint.

Bit Clock to the DAC

The DAC requires a clock running at bit rate from the connecting interface, where bit rate refers to the rate at which the OSERDESE2 generates data for the DAC. The most economical way to generate the clock for the DAC is to use an OSERDESE2 as a clock generator. This way, it is assured that clock and data are generated synchronously by the FPGA.

Each OSERDESE2 connects to the FPGA output pad in exactly the same way and with the same timing. This is also true between the different I/O banks in an FPGA, because each of the I/O banks is constructed from the same OSERDESE2 and I/O components. OSERDESE2 in

different FPGA components of one family have the same (timing) characteristics as long as the same speed grade and operating parameters are used.

An OSERDESE2 is a device with an input register running on the rising edge of the CLKDIV clock and a loadable parallel-to-serial register running on the rising edges of the CLK clock. An internal state machine with the DATA_WIDTH parameter as set point makes sure the data from the parallel input register is transferred at the right moment into the parallel-to-serial register.

Thus, when the OSERDESE2 is always loaded with the same data, the serial output is a repetitive stream of data exactly like a clock.

The format of the input data dictates the format of the serial output. When the parallel input is `01010101`[7:0] for an 8-bit input OSERDESE2, the serial output (in DDR mode) is a 50/50 clock signal.

The rate of the clock signal is directly related to the rate of the CLK that feeds the OSERDESE2. For example, when CLK is 625 MHz and the OSERDESE2 runs in DDR mode, the output is 625 MHz clock.

Figure 8 shows a fixed-rate OSERDESE2 clock generator, and Figure 9 shows a programmable rate OSERDESE2 clock generator.

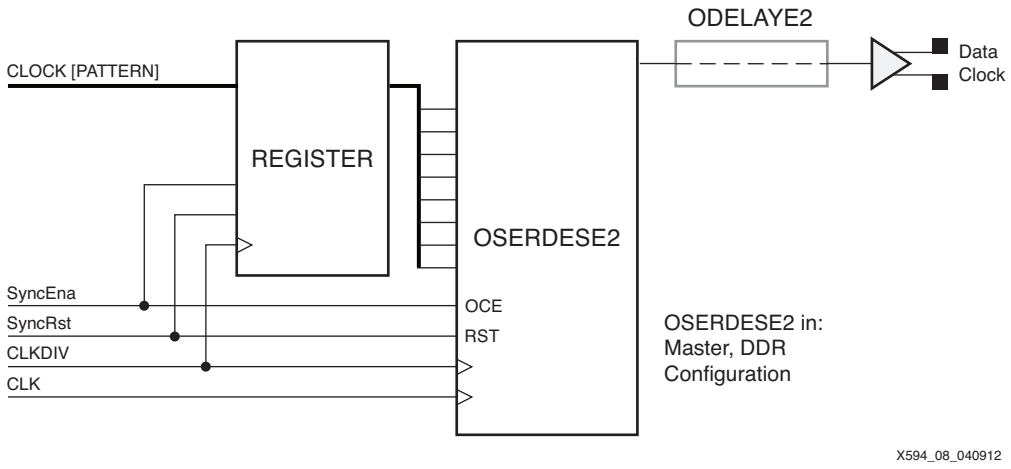

Figure 8: **Fixed Rate Clock Generator**

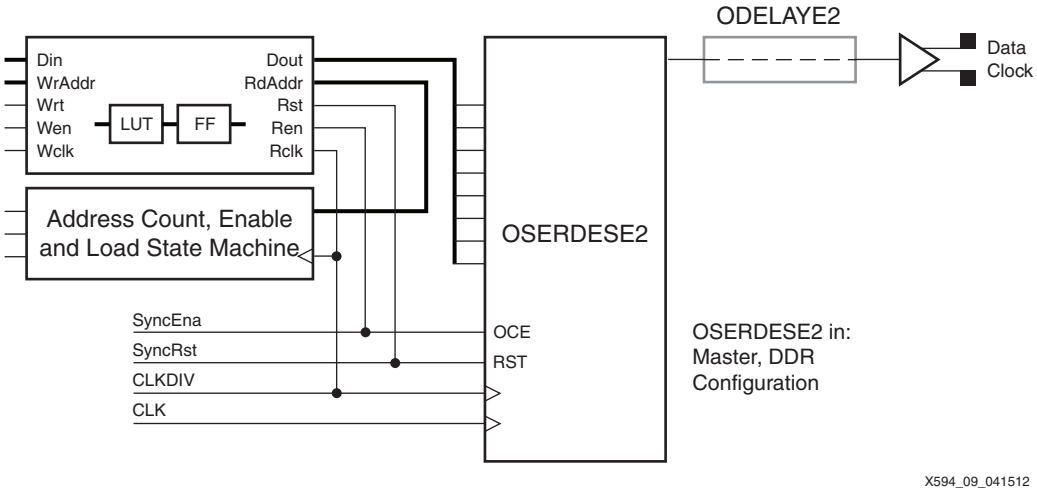

Figure 9: **Programmable Clock Generator**

The reference design files contain a project using an OSERDESE2 as a clock generator.

Whenever the DAC needs a clock that has a phase shift towards the supplied data, this phase shift can be realized using an ODELAYE2 in fixed or variable mode. Figure 8 and Figure 9 show this setup in gray.

Notes:

- Apply all guidelines for the use of OSERDESE2 as explained in OSERDESE2, page 11.
- OSERDESE2 are only available in HP I/O banks.

Data

A DAC requires parallel data. Therefore, at least one data bus must be constructed with the same width as the DAC resolution. Each bit of the bus is represented by one OSERDESE2 component. The input of each OSERDESE2 is a nibble (four bits) or a byte (eight bits). Therefore, each OSERDESE2 must load a nibble or byte that represent a number of bits in the parallel bus that connects to the DAC. Thus, the nibble or byte loaded in the OSERDESE2 represents a set of bits for one bit of the bus to the DAC, as shown in Figure 10.

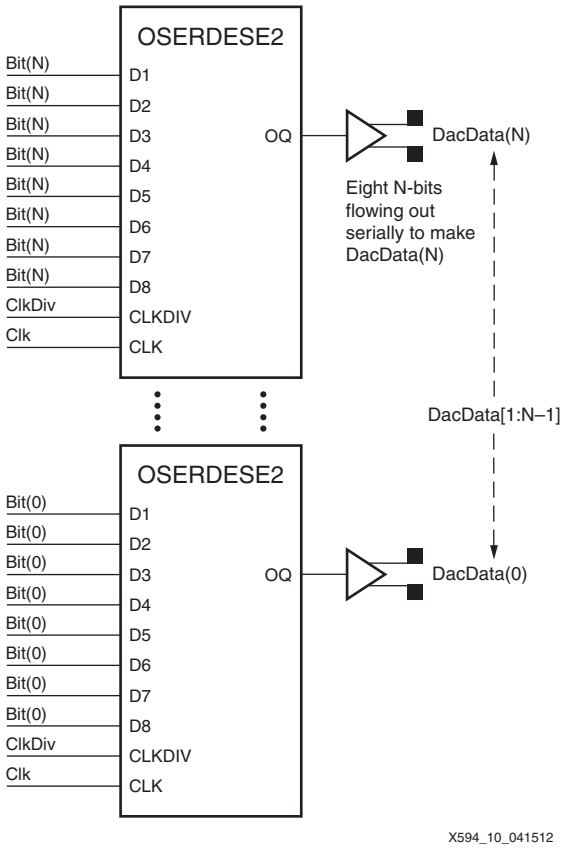

Figure 10: **OSERDESE2 Bit Arrangement for a DAC Input Bus**

If a 1.2 Gigasamples per second (GSPS) 14-bit DAC provides a 600 MHz clock to the FPGA, it can be assumed that:

- The OSERDESE2 must be used in DDR mode.
- An MMCM can be used to generate a 600 MHz CLK and a 150 MHz CLKDIV when the OSERDESE2 is used in 8-bit DDR mode.
- The application in the FPGA needs to provide 14 × 8 bits = 112 bits at a 150 MHz rate.
- This requirement is not difficult to meet when using clock domain crossing data buffers in distributed memory as temporary storage.

- The distribution of output bits from the memory to the inputs of the OSERDESE2 is done in the routing network of the FPGA as shown in Figure 11 and Figure 12. Figure 11 shows an example 16-bit resolution DAC where the application delivers data in a 16-bit bus format. In the example shown in Figure 12, the DAC has a 14-bit resolution and the back-end design delivers data in a 32-bit format.

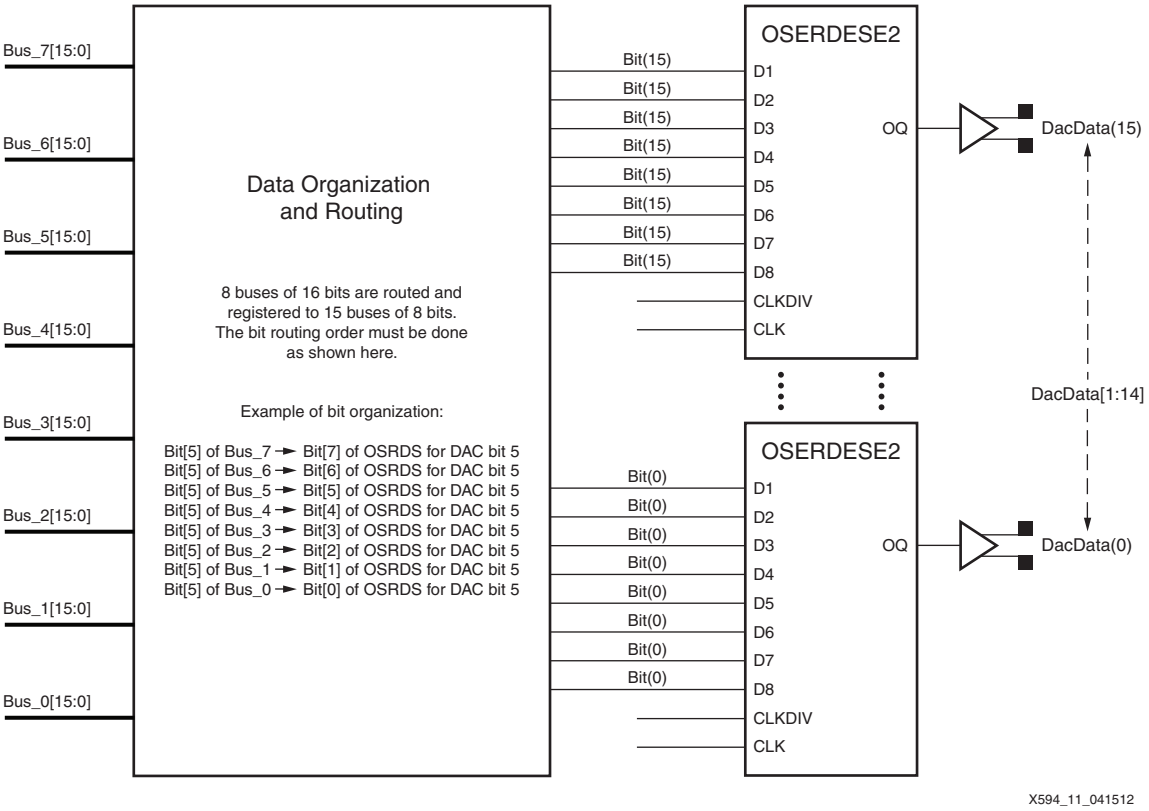

Figure 11: **Distribution of Data to the OSERDESE2 Inputs (Example 1)**

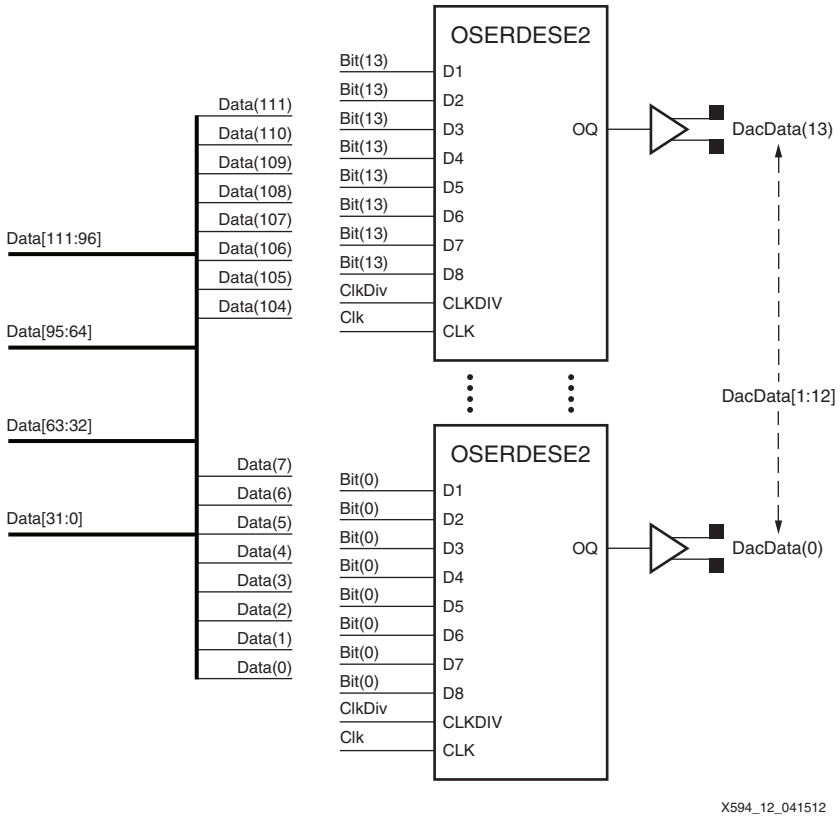

Figure 12: **Distribution of Data to the OSERDESE2 Inputs (Example 2)**

In Figure 11, eight 16-bit data buses are routed and eventually registered to sixteen 8-bit buses.

- 16 buses because that is the resolution of the DAC in this example
- 8-bit because that is the input width of the OSERDESE2

In this case, the bit routing order must be done as follows:

- Data bus Y, where Y is one of the 16-bit data input buses
- Bit n, where n is a bit from one of the Y buses
- OSERDESE2 input bus X, where X is an 8-bit OSERDESE2 input
- Bit m, where m is a bit of one of the X buses

Bit n of bus Y must go to bit Y of OSERDESE2 input m. For example:

- Bit (5) of Bus_7 → Bit (7) of OSERDESE2 input for DAC bit (5)
- Bit (5) of Bus_6 → Bit (6) of OSERDESE2 input for DAC bit (5)
- Bit (5) of Bus_5 → Bit (5) of OSERDESE2 input for DAC bit (5)
- Bit (5) of Bus_4 → Bit (4) of OSERDESE2 input for DAC bit (5)
- Bit (5) of Bus_3 → Bit (3) of OSERDESE2 input for DAC bit (5)
- Bit (5) of Bus_2 → Bit (2) of OSERDESE2 input for DAC bit (5)
- Bit (5) of Bus_1 → Bit (1) of OSERDESE2 input for DAC bit (5)
- Bit (5) of Bus_0 → Bit (0) of OSERDESE2 input for DAC bit (5)

Figure 12 shows a second example where the application delivers data for the DAC in 32-bit bus format. The used DAC has a resolution of 14 bits, meaning the lower 16 bits from the MSB bus are used. The connection of the application buses to the inputs of the OSERDESE2 can be linear with the application bus order (DataBus_13[111:96] = OSERDESE2 13 and 12, down to

DataBus_0[31:0] = OSERDESE2 3, 2, 1, and 0 inputs), or it can be a custom order. In all cases, no logic is needed, and the FPGA routing resources ensure the right connections are realized.

Other connection schemes between an application and the OSERDESE2 inputs can be used in similar or completely different ways to those shown in Figure 12 and Figure 13. If the OSERDESE2 is used with other data widths, the connection to an application can be completely different from Example 1 (Figure 11) and Example 2 (Figure 12).

OSERDESE2

An OSERDESE2 (Figure 1) is a parallel input register followed by a loadable parallel-to-serial shift register. Data is loaded into the parallel register on the rising edge of CLKDIV and shifted out the parallel-to-serial register at the rising edges of CLK.

An internal state machine controls the connection between the two registers. The state machine bounds CLK, CLKDIV, and the DATA_WITDH attribute to make sure data is always transferred from the parallel input register into the parallel-to-serial register at the correct moment.

An OSERDESE2 can be set up as a:

- Master only or master/slave.
- 2-, 4-, 6-, 8-, 10-, or 14-bit input in DDR mode (10- and 14-bit inputs are only available in the master/slave configuration)
- 2-, 3-, 4-, 5-, 6-, 7-, 8-, 10-, or 14-bit input in SDR mode

For the DAC interface, the OSERDESE2 is used in master, 4-bit, or 8-bit DDR mode. The CLKDIV rate must be set at half (4-bit) or one-fourth (8-bit) of the CLK rate.

When using the OSERDESE2, the following points should be considered:

- The parallel input register has no enable (OCE) or reset (RST). This means that as soon as a rising CLKDIV edge is applied, any data available on the input pins of the OSERDESE2 is loaded into the register.
- The OCE pin is only connected to the serial MSB output registers of the shift register.
- To prevent the OSERDESE2 from starting to generate unknown data immediately after release of the reset, keep the enable input deasserted for a number of CLKDIV clock cycles by using a LUT as programmable shift register (SRL32). The amount of clock cycles the enable input is held deasserted after releasing the reset is now programmable via the address input of the SRL32.

After releasing the OSERDESE2 reset, a rising CLKDIV edge followed by a rising CLK edge is required before any signals change, as shown in Figure 13.

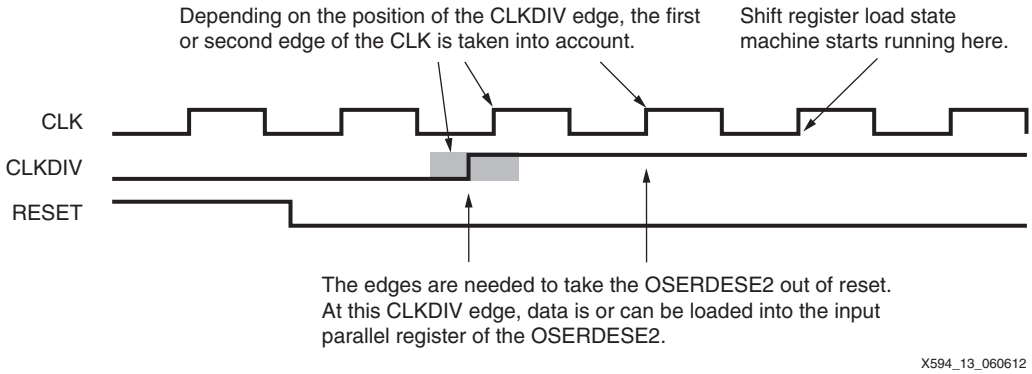

Figure 13: **Bringing the OSERDESE2 Out of Reset**

It is best to synchronize the release of the reset of the OSERDESE2 to CLKDIV and to set the enable to a few CLKDIV cycles after the release of the reset. Controlling reset and enable using timing constraints assures the designer that all OSERDESE2 of an interface become alive simultaneously and start generating data at the same moment.

Data does not immediately flow out of the ISERDESE2 after being loaded into the ISERDESE2. After release of the reset, a rising CLKDIV edge loads data in the parallel input register, and at the same time, takes the internal state machine out of reset. The rising CLK edge after a previously rising CLKDIV edge starts the internal state machine.

The internal state machine that transfers data from the parallel input register into the parallel-to-serial register depends on the DATA_WIDTH attribute. After the rising CLKDIV edge followed by the rising CLK edge, the OSERDESE2 internal state machine flushes the parallel-to-serial register. The number of bits depends on the DATA_WIDTH and DATA_RATE attributes of the OSERDESE2. Because this happens just after reset is released, the OSERDESE2 transmits all 0s until the load pulse occurs.

The internal state machine then generates a pulse, loading the contents of the parallel input register into the parallel-to-serial shift register. When not processing what is first loaded into the parallel input register, it is likely that the first serial data out of the OSERDESE2 is “garbage” data.

To prevent the OSERDES2 from outputting unknown data, it is advisable to put a register in front of the OSERDES2 input (see Figure 14) and take the following measures:

- Clock the register with CLKDIV.
- Connect the reset and enable of the register to the OSERDES2 reset and enable.
- Implement timing control constraints on reset and enable nets from the synchronization flip-flips to the registers and OSERDES2.

  The register in front of the OSERDESE2 inputs then operates as follows:

  - As long as reset is active, CLKDIV edges load 0s in the OSERDESE2 input register.
  - The first rising CLKDIV edge after release of the reset still loads 0s in the OSERDESE2.
  - That rising edge loads effective data in the OSERDESE2 front register.
  - The next rising CLKDIV edge loads meaningful data in the OSERDESE2 and it starts generating that data in serial format. The OSERDESE2 never produces unknown data at the serial output, and multiple OSERDESE2 provide synchronous data at the output pins.

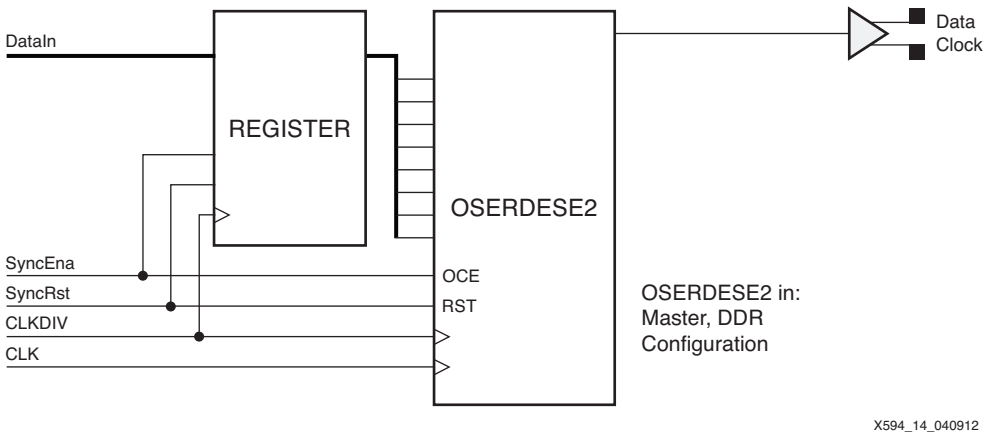

Figure 14: **OSERDESE2 with Activation Control Register**

After the first load following reset, data is loaded and shifted in regular patterns.

Figure 15 shows an OSERDESE2 in 8-bit DDR mode. After reset is released and data is loaded into the OSERDESE2 on the rising CLKDIV edge, the data appears at the output in four CLK cycles. Four CLK cycles are required because eight bits are loaded, and the controller first shifts the eight previous bits (all 0s) out at the DDR CLK rate.

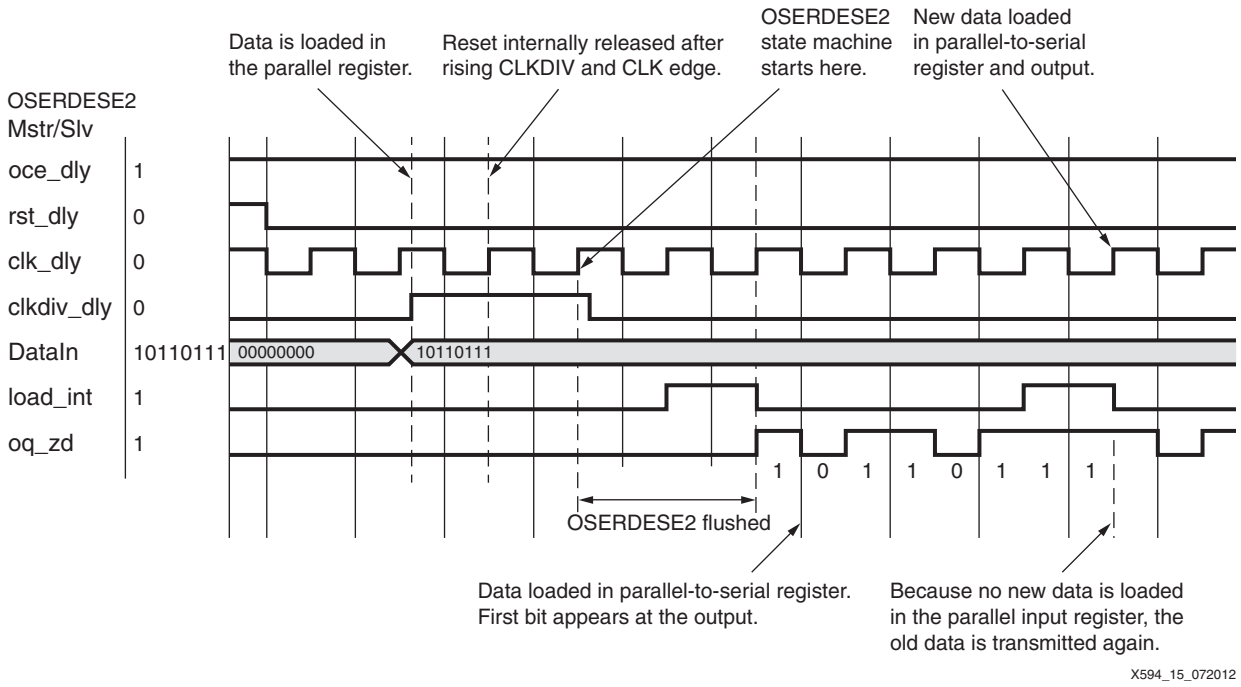

Figure 15: **First Data Out after Release of Reset**

New data can be loaded into the OSERDESE2 at any time. However, not all data loaded always appears at the output. The state machine only transfers data from the input parallel register into the output shift register after it completes the ongoing shift operation, as shown in Figure 16.

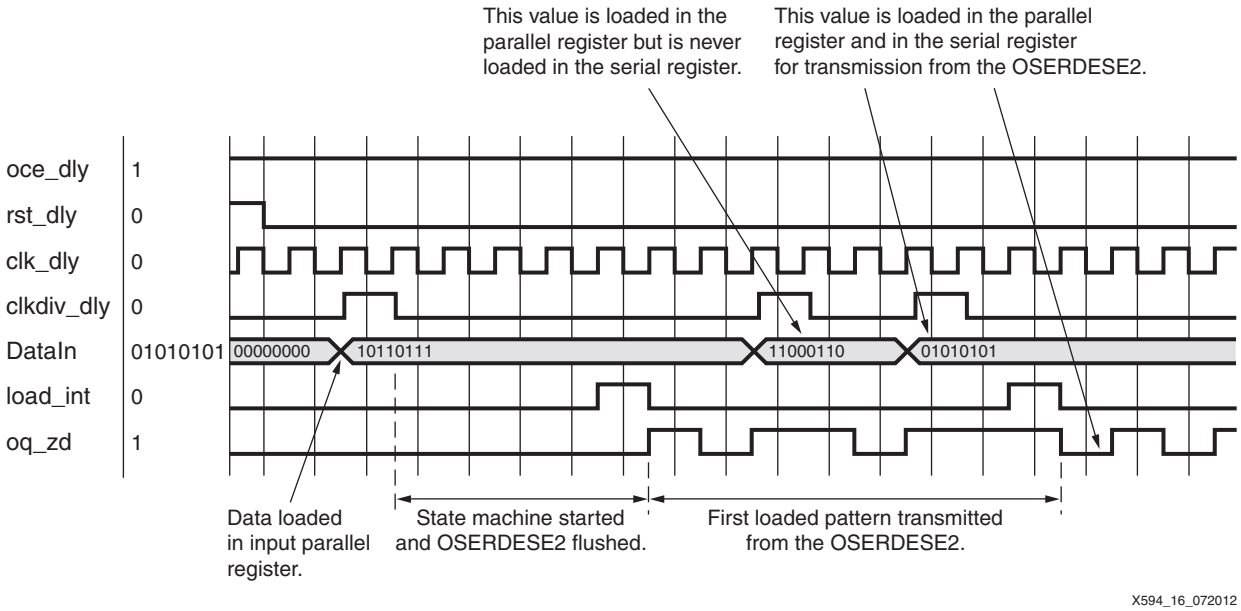

Figure 16: **Data Flow in Continue Operation**

The state machine generates shift register load pulses at the CLK rate depending on the DATA_WIDTH and DATA_RATE attributes. Data is not erased from the input parallel register after being loaded into the parallel-to-serial output register. Therefore, when no new data is loaded into the parallel register with every subsequent load pulse, the parallel-to-serial register is loaded with the same data (see Figure 16).

Reference Design

There is no device utilization summary table because the reference design files for this application note can be downloaded from:

https://secure.xilinx.com/webreg/clickthrough.do?cid=192049

The reference design checklist is shown in Table 1.

Table 1: **Reference Design Checklist**

Parameter	Description
General	
Developer name	Marc Defossez
Target device	XC7K325T-2FFG900
Source code provided	Yes
Source code format	VHDL
IP used	No
Simulation	
Functional simulation performed	Yes
Timing simulation performed	No
Test bench format	VHDL
Simulator software and version	ISE® Design Suite 13.4
SPICE/IBIS simulations	No
Implementation	
Synthesis tool/version	ISE Design Suite 13.4, XST 13.4
Implementation tool/version	ISE Design Suite 13.4
Static timing analysis performed	Yes
Hardware verification	
Hardware verified	Yes
Hardware platform used for verification	KC705 board

The reference design consists of four small designs, from which some or all can be used to build a DAC interface. The use of FPGA resources thus depends on the brand and type of DAC used.

If a DAC needing interleaved data is selected, a double amount of data OSERDESE2 components are needed. It is possible, depending on the DAC resolution, that two I/O banks are necessary.

If the user wants an estimate of area, assuming that all OSERDESE2 in the chosen I/O bank are used, count for an area of 8 by 50 slices. This is the area between the ISERDESE2/OSERDESE2 components and the first set of block RAM components (see Figure 17).

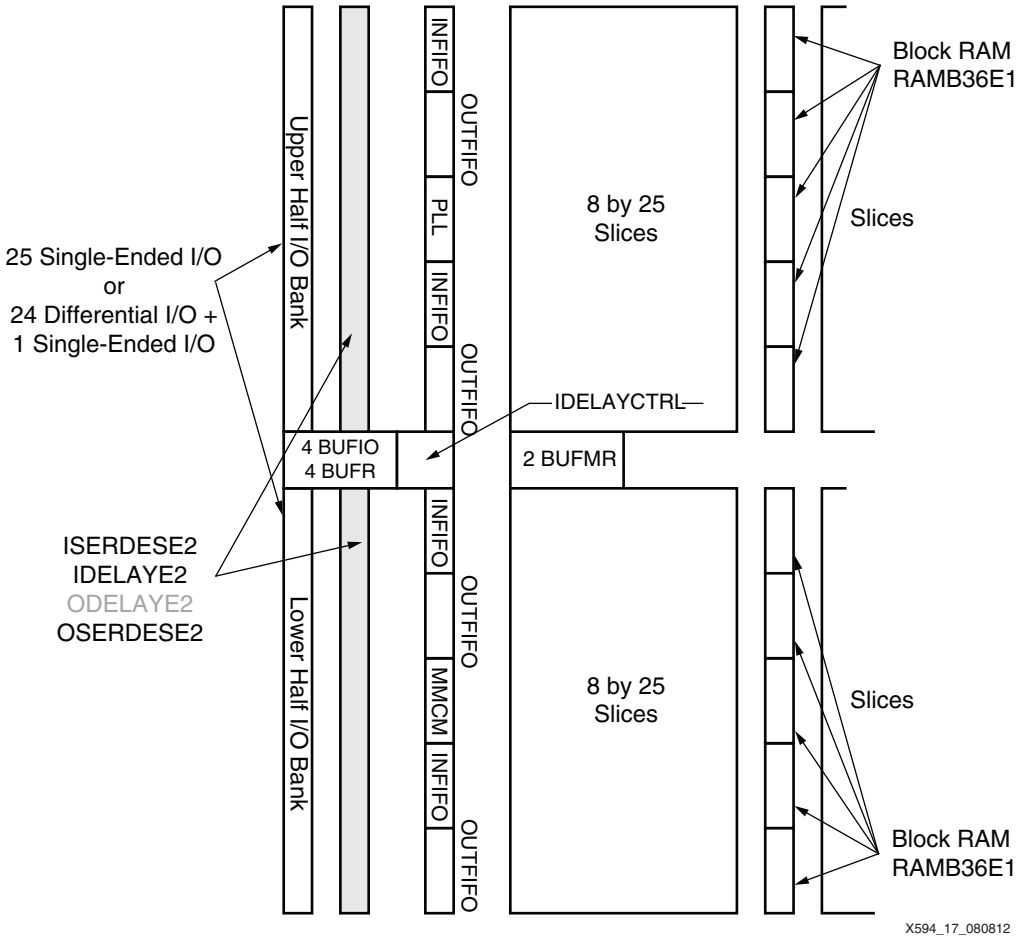

Figure 17: **Possible Device Utilization Area**

Reference Design Directory Setup

Figure 18 shows the directory structure of the reference design.

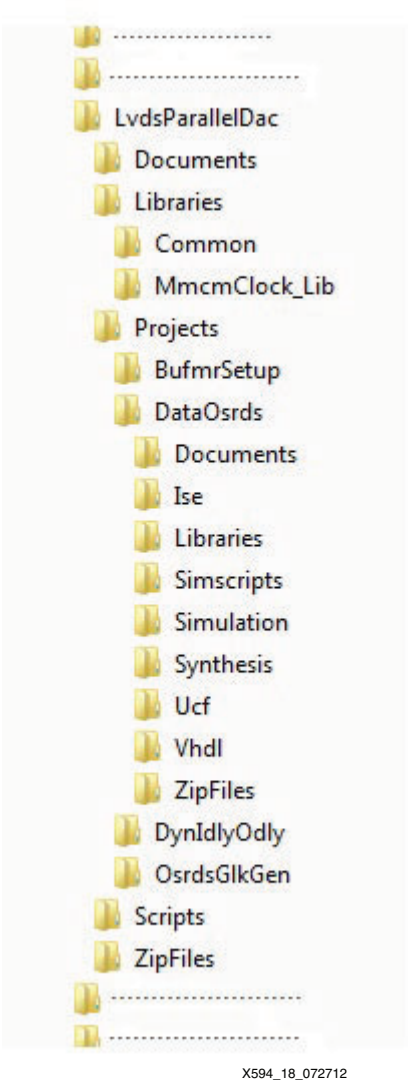

Figure 18: **Reference Design Directory Structure**

References

This document uses the following references:

1. UG471, *7 Series FPGAs SelectIO Resources User Guide*
2. DS181, *Artix-7 FPGAs Data Sheet: DC and Switching Characteristics*
3. DS182, *Kintex-7 FPGAs Data Sheet: DC and Switching Characteristics*
4. DS183, *Virtex-7 FPGAs Data Sheet: DC and Switching Characteristics*

Conclusion

The 7 series FPGA interfaces use OSERDESE2 features to provide a flexible and versatile platform for building high-speed LVDS interfaces to all the latest DAC devices on the market.

Revision History

The following table shows the revision history for this document.

Date	Version	Description of Revisions
08/22/2012	1.0	Initial Xilinx release.

Notice of Disclaimer