Practical Guidelines for Interfacing Data Converters to Field Programmable Gate Arrays

Prepared by

Craig Holmberg
Principal Engineer
Bitwise Design Labs

Presented at DATT Summit 2016 with support from

Glenn Rosenthol
CEO/President
Ulyssix Corporation

Table of Contents

1 Background 3

2 Converter Choices 4

2.1 Synchronous Serial Converters 4

2.2 Parallel Converters 5

2.3 JESD204B Converters 8

3 FPGA Considerations 10

3.1 Synchronous Serial Interface 10

3.2 Parallel Interface 12

3.3 JESD204B Interface 14

4 Board Design Considerations 15

4.1 Synchronous Serial Interface 15

4.2 Parallel Interface 15

4.3 JESD204B Interface 15

5 Summary 17

6 Appendix A: Abbreviations and Acronyms 19

List of Figures

Figure 1: Synchronous Serial DAC 5

Figure 2: Synchronous Serial ADC 5

Figure 3: Parallel DAC 6

Figure 4: Parallel ADC 6

Figure 5: Parallel ADC with source synchronous clocking 7

Figure 6: Parallel ADC with differential signaling 7

Figure 7: JESD204B Interface Between FPGA and ADC or DAC 8

Figure 8: ADC Serial Interface Using Specialized FPGA Resources 11

Figure 9: ADC Serial Interface Using Dedicated SERDES 11

Figure 10: ADC Parallel Interface Using Specialized FPGA Resources 13

Figure 11: ADC Interface Selection Flowchart 18

Practical Guidelines for Interfacing Data Converters to Field Programmable Gate Arrays

1 Background

Offering substantial digital signal processing capabilities coupled with the flexibility to interface almost any digital interface, the field programmable gate array (FPGA) is becoming commonplace in the data path of many mixed signal systems. In such systems the FPGA is a logical point to connect the converters, both analog-to-digital (ADC) and digital-to-analog (DAC). This is simple in concept – select the right converter then bolt it to the appropriate intellectual property (IP) core. Without some due diligence and attention to details, though, performance issues can be introduced including dropped bits, missed samples, and increased power dissipation. The product development cycle can also suffer through additional board spins and unnecessarily complex board routing. The intent of this document is to provide practical guidance to the engineer designing mixed-signal systems that can help avoid these issues in low, mid, or high-end converter applications.

2 Converter Choices

When bridging the continuous analog domain and discrete digital domains two obvious parameters for the conversion are the sampling rate and the sample bit width. Assuming you have selected a rate sufficient for proper sampling[1] and a bit width sufficient for the desired resolution then a third parameter becomes key – the digital interface: In both ADC and DAC applications the digital side of the converter can travel over serial, parallel, or transceiver-based (most notably JESD204B) links. Each type can be handled by FPGAs, and each has its own set of tradeoffs.

2.1 Synchronous Serial Converters

The serial interface is in general the simplest to implement. Specifically, the synchronous serial interface where a bus master (the FPGA in this context) provides a clock used by the slave (the converter) to synchronize data.

For low-speed converters (nominally below 5 MSps) the two prevalent interfaces are I^2C and SPI.[2] Higher speed synchronous serial converters typically utilize framing signals to support serialization/deserialization (SERDES).

The primary reason to choose a serial converter is simplicity, specifically the simplicity of the printed circuit board design and the logic to support the interface: Board design is simplified since the number of traces between the host FPGA and the converter is minimal (two to four typically) and the traces don't require special routing considerations beyond normal best practices. This is in contrast to the other converter types which can require trace matching, impedance control, or differential coupling. The logic design is simplified since the encoding scheme is serialized data plus overhead in the form of framing and/or addressing.

Scalability is another plus to the serial converters. Select the right serial converter and you can migrate to a higher resolution or faster converter using the same footprint and board design, or conversely cost down by switching to a reduced precision or throughput variant.

The tradeoff is bandwidth: Source synchronous clocking over single-ended signals limits the line rates. Some suppliers offer serial converters that utilize differential standards (typically LVDS) or double-data-rate (DDR) clocking to extend the rates.

The mixed-signal synchronous serial system amounts to a bus master implemented inside the FPGA, two to four traces between the FPGA and the converter[3], and the ADC or DAC device. For DAC circuits the FPGA drives out a clock, serial data, and optionally control lines such as an enable or framing:

---

[1] In simplified form, the sampling frequency must be at least 2x the highest frequency in the analog signal. This is based on foundational work by H. Nyquist and C. Shannon and is covered in great detail in many other texts.
[2] I^2C was developed by Philips, now NXP. NXP still maintains the standard. SPI was developed by Motorola and is a defacto standard.
[3] In some instances the serial bus may be shared across multiple converters. In such cases the guidance presented for the single-converter case can be extended to multiple converters.

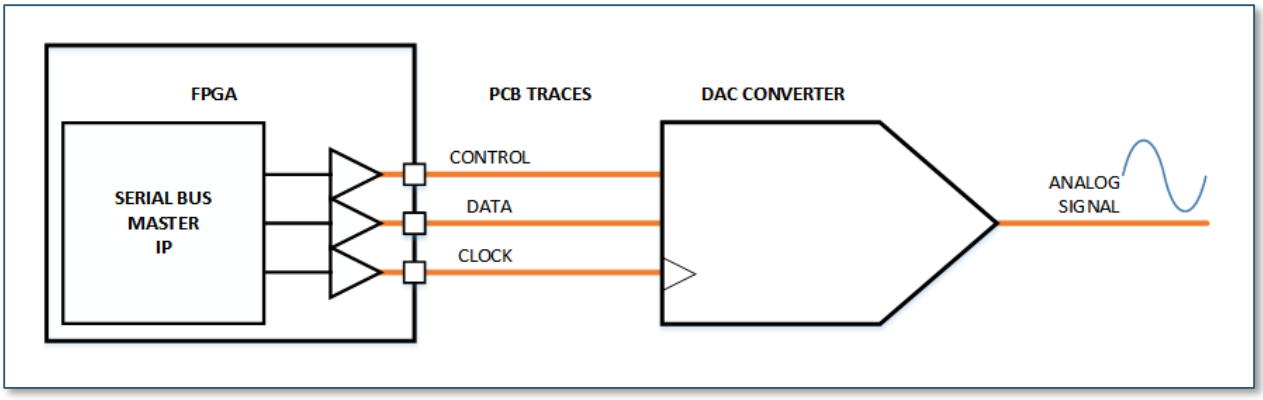


Figure 1: Synchronous Serial DAC

For ADC circuits the configuration is identical except the data direction is reversed:

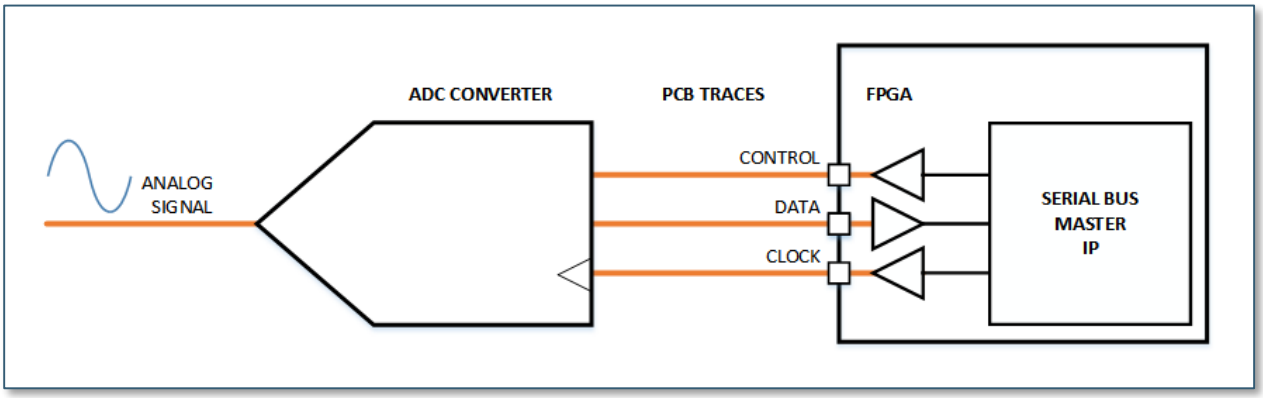


Figure 2: Synchronous Serial ADC

2.2 Parallel Converters

When synchronous serial interfaces can't move enough data the next logical step is the parallel interface. The parallel interface is pure simplicity in terms of logic – one clock and a bit lane for each bit of the digital word. At the board level a trace is required for each data bit, and increasing resolution means adding additional traces. The dedicated bit lanes also impact component packaging; in general, a parallel converter will require a larger package and more interconnects than its serial counterpart. The obvious tradeoff with parallel converters relative to serial converters is that the performance gains come at the expense of circuit board area.

For DAC circuits the FPGA drives out a clock and parallel data lanes:

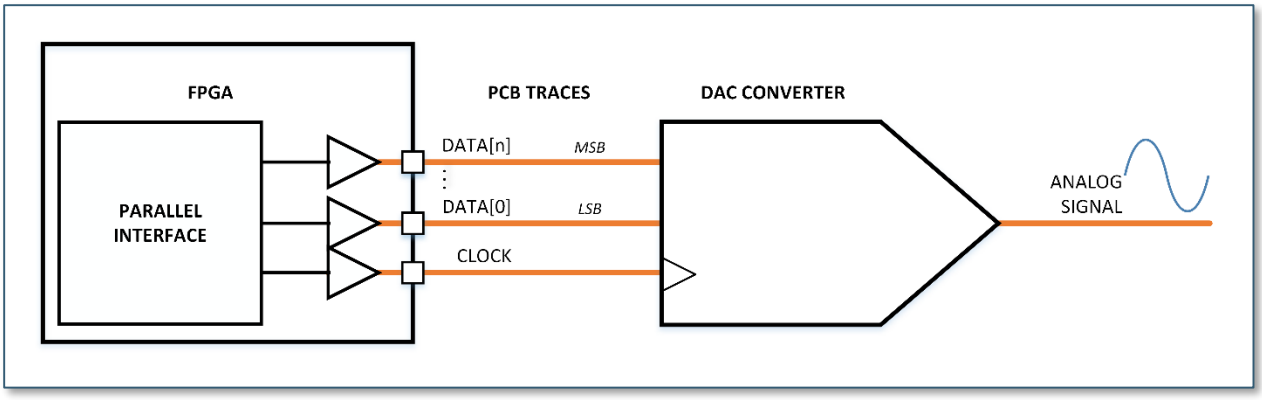


Figure 3: Parallel DAC

For parallel ADC circuits the configuration is identical except the data direction is reversed:

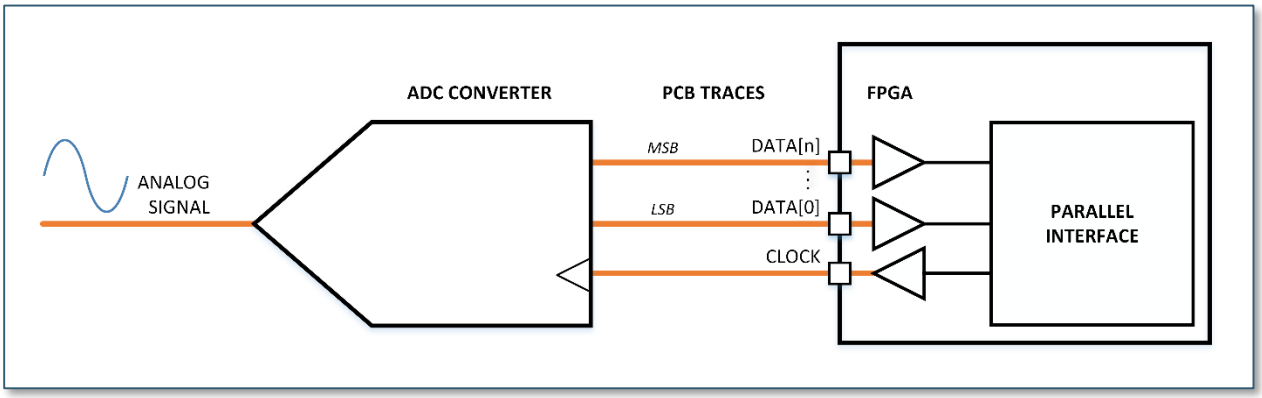


Figure 4: Parallel ADC

The maximum rate for parallel interfaces is limited by skew between the individual bits. Since all bits share a common clock, the bit lane with the most delay relative to the clock will be the boundary for the setup timing window while the bit lane with the least delay will be the boundary for the hold timing window. Minimizing skew between the clock and each bit lane is essential for high-speed parallel converters.

As rates increase the simple converter scheme of clocking parallel bits can be extended in performance (and complexity) utilizing double data-rate clocking, source synchronous clocking, and/or differential standards such as LVDS, PECL, or CML:

Double-data-rate clocking does not alter the above diagrams; the only change is that data is clocked on both the rising and falling edges of the clock. This effectively doubles the throughput over the parallel interface, but adds a requirement for timing to be constrained and maintained relative to both rising and falling clock edges.

Source synchronous clocking adds an extra clock trace. The concept is that that the reference clock is used to time the conversions, while the added clock is transmitted phase aligned to the data at the

transmission point so that if the clock and data traces are closely matched on the circuit board then clock and data will arrive at the FPGA still aligned:

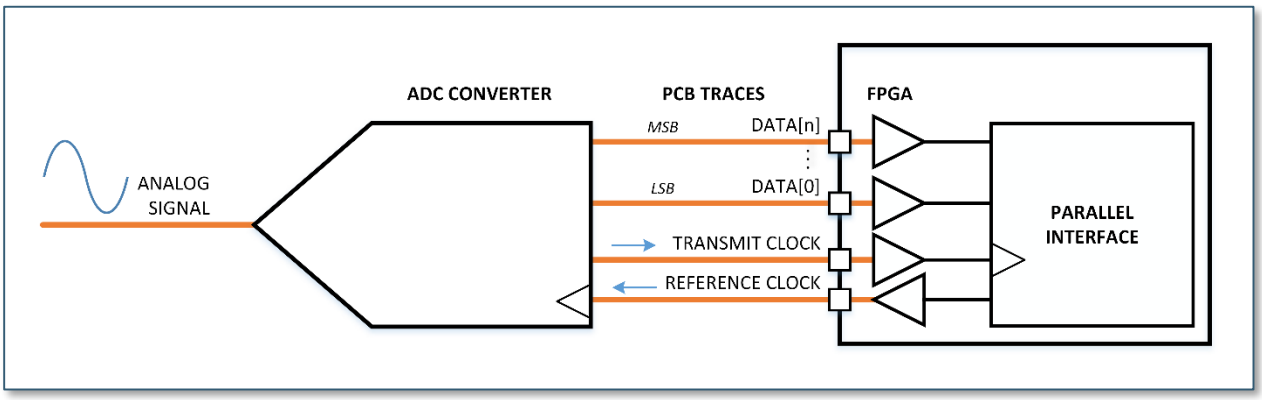


Figure 5: Parallel ADC with source synchronous clocking

Differential standards extend the rates beyond what is achievable with single-ended standards such as LVCMOS/LVTTL. The cost is that each signal now requires two physical traces, and those traces need to be properly coupled and terminated:

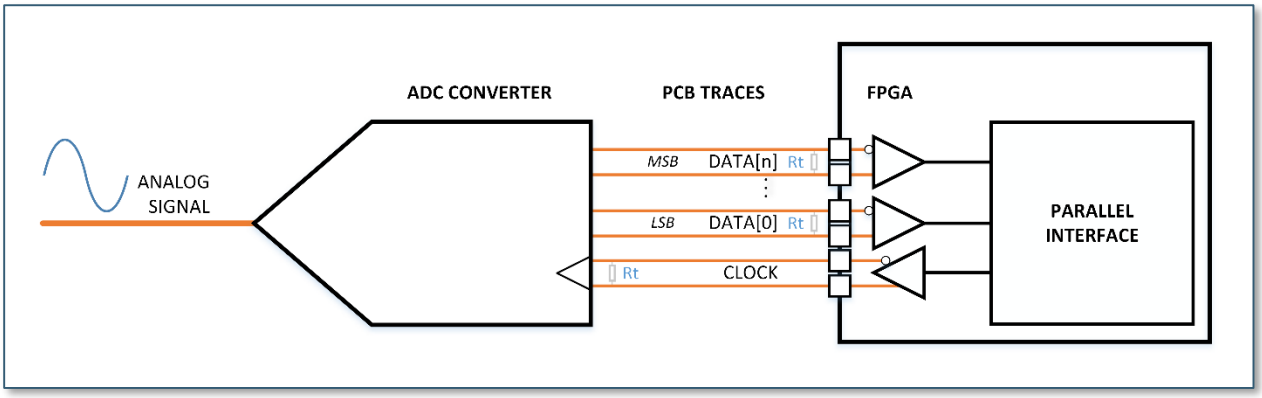


Figure 6: Parallel ADC with differential signaling

Some suppliers provide devices which can be configured to support single-ended or differential signaling.

2.3 JESD204B Converters

The third type of digital interface combines advantages of both the serial and parallel converters – requiring a minimal number of signals between the host FPGA and the converter while offering speeds that match (or exceed) parallel converters. If you plan to use high-speed data converters then you should become aware of JESD204B.

Unlike the previously discussed serial busses, JESD204B was created specifically to support interfaces between FPGAs and ADC/DAC devices. A key differentiator versus general purpose serial busses is that it offers deterministic latency. From page 1 of the specification[4], "*This specification describes a serialized interface between data converters and logic devices.*"

These benefits come at a cost: The JESD204B protocol requires high-speed serial transceivers instead of simpler capture registers or SERDES circuitry, and JESD204B links require a serial protocol instead of simple bitstreams. The good news is that multiple FPGA suppliers offer mid-range[5] and high-end FPGAs capable of supporting the interface, and the protocol can be handled using commercially available IP cores.

The block diagram for a JESD204B[6] interface between FPGA and one or more converters is as follows:

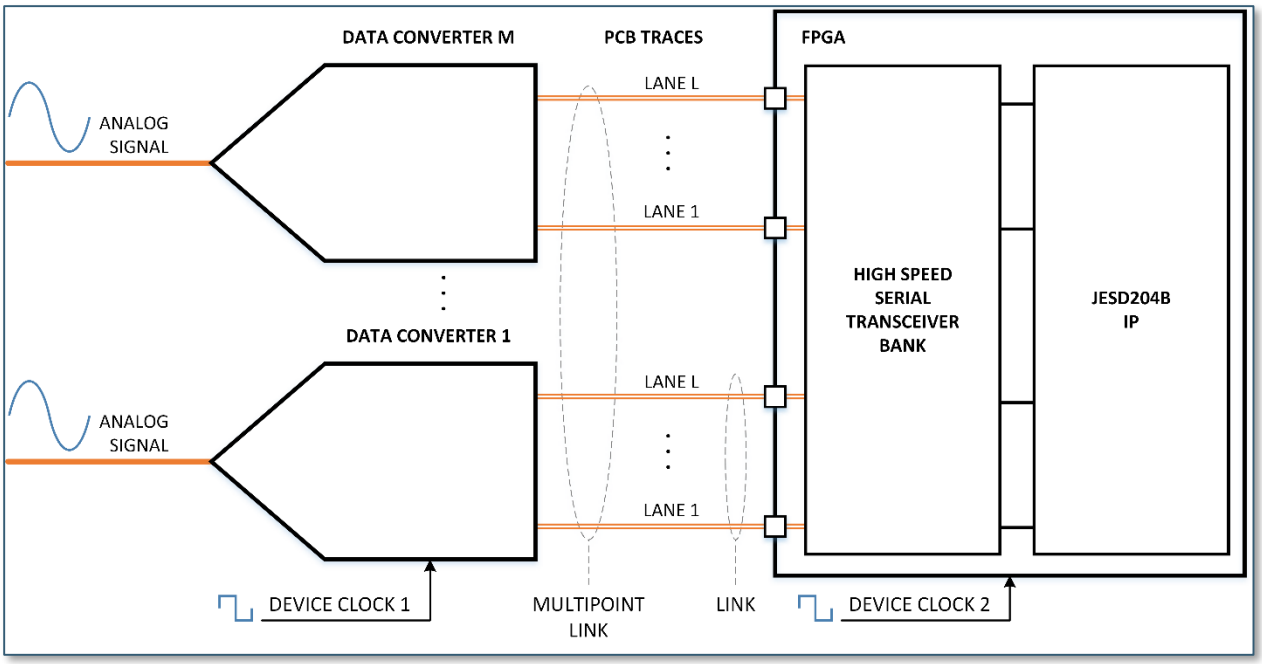


Figure 7: JESD204B Interface Between FPGA and ADC or DAC

[4] *Serial Interface for Data Converters*, JEDEC Standard JESD204B.01, January 2012

[5] Some vendors also offer cost-optimized families with transceivers capable of supporting JESD204B at lower speeds.

[6] The standard defines three subclasses – 0, 1, and 2. In the context of this document "JESD204B" will be used to refer to subclass 1 since 0 is for backward compatibility and 2 is for rates below 500 MSps.

As defined in the JEDEC specification,

a) **Lane** = one point-to-point differential CML line
b) **L** = the number of lanes per link
c) **Link** = L lanes together used to connect one converter to the FPGA
d) **M** = number of converters per device
e) **Multipoint link** = M links, each consisting of L lanes.

The interface can scale from a single lane connecting one converter to the FPGA up to multiple converters with multipoint links interfaced to a single FPGA.

Note that each end of the link requires a local reference clock, but that no separate clock travels with the data lanes. Instead, the clocking is recovered at the receiving end which is the FPGA for ADC applications and the DAC for digital-to-analog. Links utilize CML signaling to save power at higher frequencies (relative to single-ended TTL/CMOS or differential LVDS) and 8B10B encoding to guarantee run rates for clock data recovery. The standard specifies rates up to 12.5 GHz. No minimum rate is specified, however transceivers will have a minimum frequency below which they cannot recover the clock.

3 FPGA Considerations

Moving up the performance path from serial to parallel to JESD204B introduces increasingly stringent requirements for the FPGA. These requirements plus your application-specific requirements should be factored together when selecting an FPGA.

Within your selected FPGA, timing on the interface can play a critical role in circuit performance. A design that is logically functional but has timing violations can still introduce bit errors, and depending on the bit position where the error occurred the magnitude of that error can be significant. Factored into timing analysis are variations in process, voltage, and temperature (PVT); creating a design that functions properly through these PVT variations involves attention to detail in timing constraints and analysis, hence the frequent references to timing that follow for each type of interface. Specifics on constraining timing are not presented since they are specific to your selected FPGA and FPGA tool flow.

3.1 Synchronous Serial Interface

The signal rates and voltages required to interface converters that utilize serial interfaces such as I^2C or SPI can be handled by most FPGAs (and CPLD's for that matter). Supporting the serial interface to the converter is probably not going to drive the device selection. The logic to implement the bus master may require some consideration: Many FPGA tool libraries include the IP for I^2C and/or SPI masters in their libraries. Some points to keep in mind are

- These serial protocols were originally developed as simple processor-to-peripheral interfaces, and as such the library implementations may not have deterministic latency due to elasticity in the data path. This can introduce detrimental timing variations in the conversion points.

- Some data converters offer special features such as power-down or nap modes that require nonstandard bus sequences to control. A standard SPI or I^2C master may not be able to generate the proper sequences to utilize these features.

In short, you may be able to get your system up and running using standard IP cores for the bus master, but a custom bus master implemented using finite state machines and logic may allow you to improve the performance and/or utilize an extended feature set of your particular converter.

Serial converters that utilize data streams with framing (versus SPI or I^2C) can be supported using shift registers and a minimum of glue logic. As serial rates increase into the hundreds of MHz you may need to consider FPGAs with additional specialized resources such as

1) I/O Delay Elements – useful for inserting delay into a path to influence clock-data alignment

2) Phase Lock Loops (PLLs) – useful for phase shifting clocks to improve timing margins

3) Dedicated registers in the I/O elements – short fixed paths between the i/o pins and these registers yield predictable timing without the variations imposed by internal routing

4) Dedicated Double Data Rate registers – same as above, but with both rising and falling edge capture registers implemented in the IO elements

5) Dedicated SERDES circuitry – hard logic dedicated to serialization/deserialization of data streams

Note that all of these resources except the dedicated SERDES can be used in combinations; dedicated SERDES circuits generally include dedicated i/o paths and clocking and thus are not combined with other elements. These resources can be used to influence data path and/or clock timing to optimize the sampling windows. The end goal is to sample on clock edges without violating setup or hold times across process, voltage, and temperature variations.

A potential ADC interface block diagram employing all of the listed resources except SERDES would be as follows:

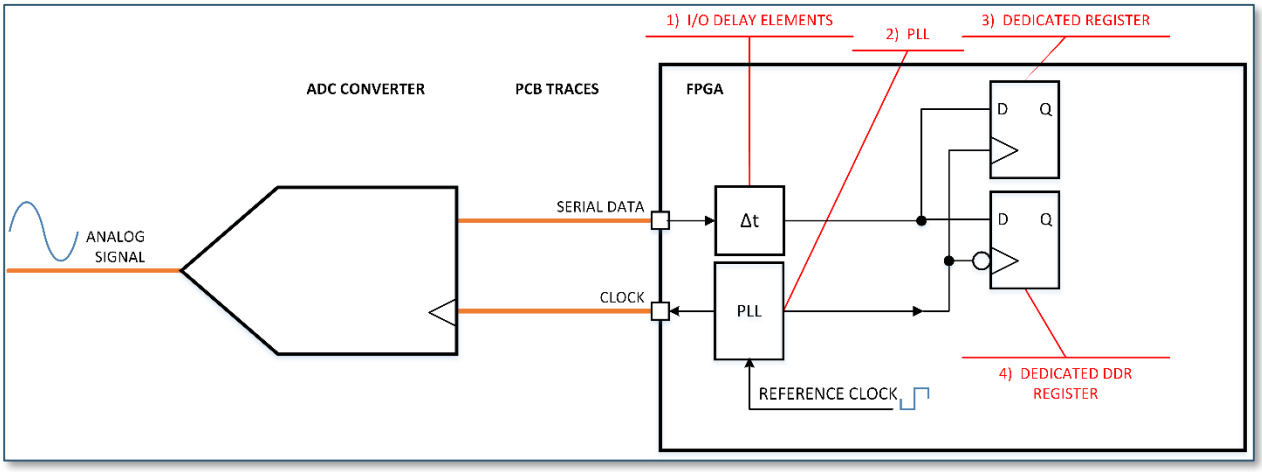


Figure 8: ADC Serial Interface Using Specialized FPGA Resources

And here is an implementation using dedicated SERDES circuitry:

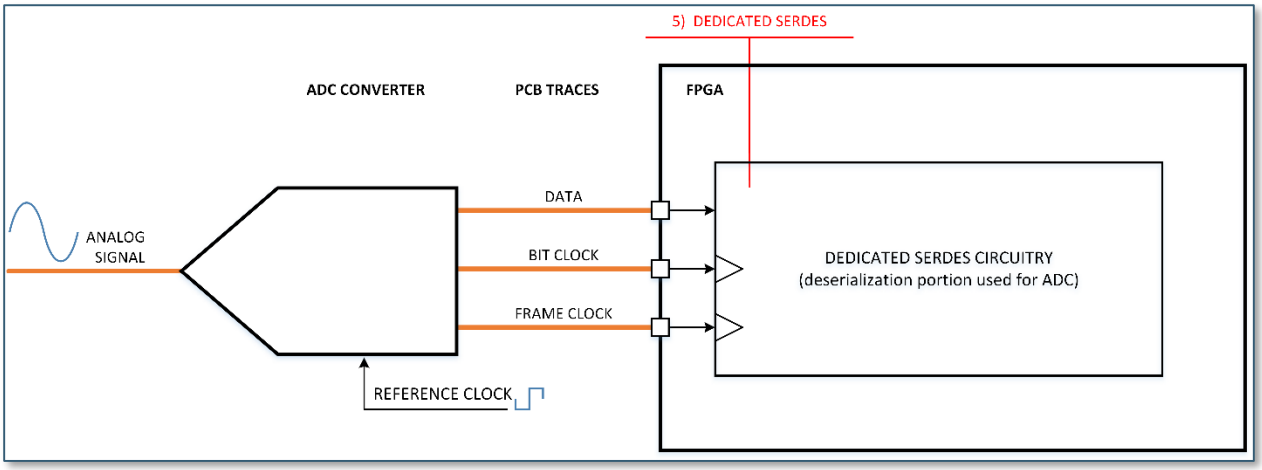


Figure 9: ADC Serial Interface Using Dedicated SERDES

Although synchronous serial interfaces are typically forgiving in nature, designing a robust link that will operate over process, voltage, and temperature extremes will require proper timing constraints. The format and approach varies greatly depending on your chosen FPGA tool flow. Regardless of your tool set a fully constrained design will require that you know timing characteristics of

- the internal clock used to drive the bus master logic
- the external clock output by the FPGA relative to the internal clock
- trace delays for both clock and data lines
- Converter's time from clock-to-output for the signals driven from the converter to the FPGA
- Setup and hold times for signals driven from the FPGA to the converter.

In reality the trace delays are generally negligible for the low speeds inherent to the synchronous serial interface, but that is a judgement call you should make based on your specific application.

3.2 Parallel Interface

The parallel interface can be implemented by registers in the FPGA and general purpose I/O. The key point is to select a device capable meeting the timing requirements for the clock and data at the target rates. As line rates increase into the hundreds of MHz you may need to consider FPGAs with additional specialized resources such as

1) I/O Delay Elements – useful for inserting delay into a path to influence clock-data alignment

2) Phase Lock Loops (PLLs) – useful for phase shifting clocks to improve timing margins

3) Dedicated registers in the I/O elements – short fixed paths between the i/o pins and these registers yield predictable timing without the variations imposed by internal routing

4) Dedicated Double Data Rate registers – same as above, but with both rising and falling edge capture registers implemented in the IO elements

5) Dynamic Phase Aligner (DPA) –circuitry used to dynamically select a phase-shifted variant of a clock independently for each incoming data lane.

Note that all of these resources except DPA can be used in combinations and can be applied to both ADC and DAC interfaces; the DPA feature supports the receive direction (ADC) only. A potential ADC interface block diagram employing all of the above except DPA would be as follows:

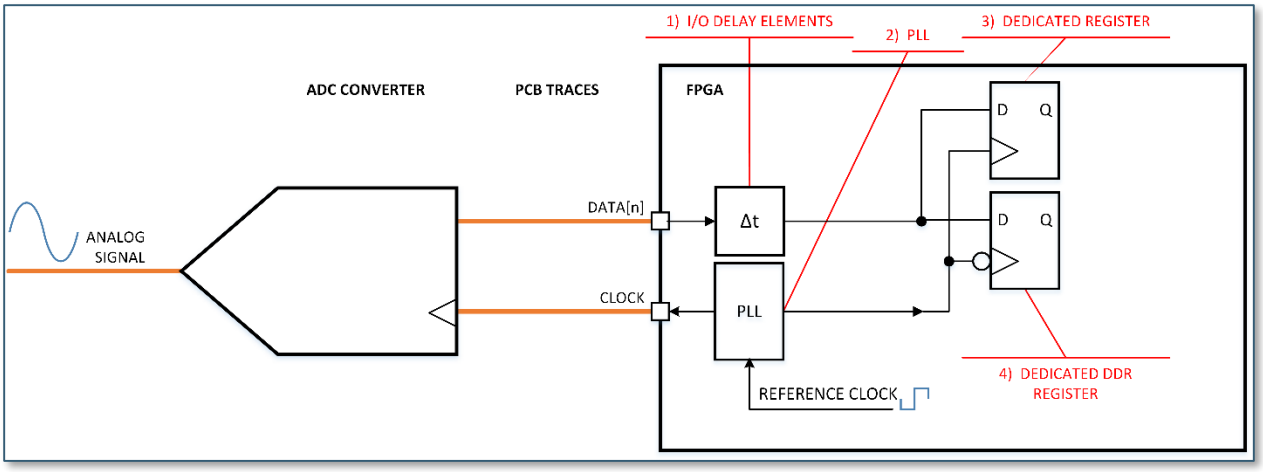


Figure 10: ADC Parallel Interface Using Specialized FPGA Resources

For clarity only one data lane is shown; the data path "DATA(n)" would be replicated for each bit.

Even with the performance gains achieved using the above approach, at some point the receiving device will no longer be able to capture all bits reliably. As a generalization this will happen between 500 MHz and 1 GHz. For applications where the FPGA is the receiving end (i.e. ADC) DPA can support even higher rates. The concept is simple: Since one clock can no longer capture all bits reliably, generate multiple phase variants of that clock then determine which shifted clock is best for each bit. The details of the implementation can be more complex. Some FPGAs[7] implement this as dedicated circuitry while others synthesize the circuitry. For more information search keyword "DPA" on your FPGA supplier's web site.

If your parallel interface involves differential standards such as LVDS then consider FPGAs with on-chip differential termination resistors. In addition to eliminating resistors this option places the termination resistor closer to the differential receiver than is possible with an external resistor, which in turn yields improved signal integrity.

Since timing budgets and sample windows drive the performance of the parallel interface it is critical that you constrain the FPGA timing. Failure to do so can lead to data integrity issues over process, voltage, and temperature extremes. The method for constraining the timing varies depending on your chosen FPGA tool flow, but in all cases a fully constrained design will require that you know timing characteristics of

- the internal clock used to drive the parallel bus interface logic
- the external clock output by the FPGA relative to the internal clock
- trace delays for both clock and data lines
- Converter's time from clock-to-output for the signals driven from the converter to the FPGA
- Converter's setup and hold times for signals driven from the FPGA to the converter.

[7] Select members of the Intel / Altera Stratix and Arria families

Also, if your FPGA timing analysis tool includes a constraint for bounding and/or reporting skew across the data bus then use it.

3.3 JESD204B Interface

The two salient factors for FPGA selection in JESD204B applications are

- ☑ FPGA transceivers capable of complying with JESD204B at the target rates
- ☑ JESD204B IP compatible with the target FPGA and converter(s).

The transceivers must be capable of meeting the requirements for the JESD204B physical layer. Some FPGA suppliers offer a characterization report documenting their part's performance specific to the JESD204B standard; that can be a good source of information.

The IP must be compatible with the selected FPGA and the supporting development tools. The IP must be compatible with selected converter as well. Most IP cores can be parameterized to work with a range of converters, however there are many potential configurations and getting the details right can prove tedious. Finding a combination of FPGA, converter, and IP that has been proven already may save you debugging time on the bench. Ideally the FPGA, converter, and IP suppliers should provide data on tested configurations.

For transceiver links the clock is embedded into the data stream. This changes the timing analysis: Where a synchronous receiving register analyzes setup and hold times, the transceiver analyzes receiver skew margin. Where a synchronous transmitting register analyzes time from clock-to-output the transceiver analyzes channel-to-channel skew. Timing constraints for the JESD204B FPGA design can be involved, so follow the guidelines provided by the IP supplier.

4 Board Design Considerations

Some aspects of board design are common to all converter applications -- attention to proper grounding, a good power distribution network, bypass and decoupling capacitors, circuit board materials and stackup all contribute to system performance. Additionally, there are requirements that vary with the type of interface used between the converter and FPGA:

4.1 Synchronous Serial Interface

As previously mentioned, the serial interface is the least demanding in terms of board design. Standard materials and design rules can be applied; a typical board specification such as FR-4 laminate, ½ to 1 ounce Copper layers, and 5 to 10 mil trace widths and spacing is generally more than adequate. Minimize trace lengths to maintain signal integrity, although longer runs can typically be tolerated. On multilayer boards, stripline or microstrip routing can be used to minimize impedance discontinuities; for noise immunity use stripline routing but account for the lower signal velocity when constraining signal delays. If signal runs are more than a couple of inches long then consider series or parallel terminations to minimize reflections, especially on clock lines. A rough rule of thumb is that if the trace length in inches is greater than the driver's rise time in nanoseconds then you should terminate[8].

4.2 Parallel Interface

In addition to the requirements mentioned for serial interfaces, parallel interfaces add the requirement of trace matching. The clock and parallel data bits interconnecting the converter and the FPGA should ideally have the same trace length. Serpentine routing is typically applied to extend shorter runs until they match longer runs. Perfect matching is not always feasible; a reasonable design goal is 200 mil (5 mm) difference between the longest and shortest traces.

In cases where the traces are differential (as in LVDS) the side-by-side arrangement is commonly used. These traces should be routed together maintaining a fixed separation, although minor excursions to circumvent vias or other components are permissible. The length of the two traces should be matched as closely as possible, and like the overall parallel interface a typical design goal is matching within 200 mils (5 mm). Regarding trace separation, a rule of thumb by Dr. Howard Johnson is to specify trace separation for the routed pair of $4 \times H$ where H is the signal height above the nearest ground plane.[9]

4.3 JESD204B Interface

The requirements for JESD204B depend on trace lengths and link rates. The links are required to use CML as the i/o standard, with controlled impedance traces routed for 100 Ω differential impedance. The standard for device clocks is left open, although LVCMOS is suggested as the general choice with

[8] Barry Olney, "Impedance Matching: Terminations", *The PCB Design Magazine*, p 38, October 2013
[9] Dr. Howard Johnson, *High-Speed Digital Design Online Newsletter*, vol 2, no. 30, 1998

differential standards as an option for higher clock rates. Trace lengths nominally target 8 inch (200 mm) runs. For trace lengths below 8 inches and rates below 6.375 Gbps FR-4 can be used; higher rates and/or longer runs will require a higher performance material.[10]

[10] *Serial Interface for Data Converters*, JEDEC Standard JESD204B.01, pp 12-27, January 2012

5 Summary

In summary, the type of digital interface used to interconnect a data converter and FPGA impacts the system scalability, circuit board design, and resources required in the FPGA. Synchronous serial, parallel, and JESD204B interfaces each offer distinct advantages and compromises:

- Synchronous serial is the simplest to implement, requiring a minimal number of traces and circuit board area. Serial converters can potentially scale in terms of bit width and sample rate without changing the circuit board. For lower data rates I^2C and SPI are the prevalent standards; at higher rates serial streams with framing are typical. Differential standards and DDR clocking may also be used to increase serial rates.

- Parallel interfaces provide higher rates than synchronous serial, but also require more circuit board area due to larger packages and an increased number of interconnects. Circuit board design is more complex than serial interfaces since trace length matching is required across the clock and data bits. The logic to interface parallel converters is simple, however specialized FPGA resources may be required to optimize timing. As rates increase the simple converter scheme of clocking parallel bits can be extended in performance (and complexity) utilizing DDR clocking, source synchronous clocking, and/or differential standards.

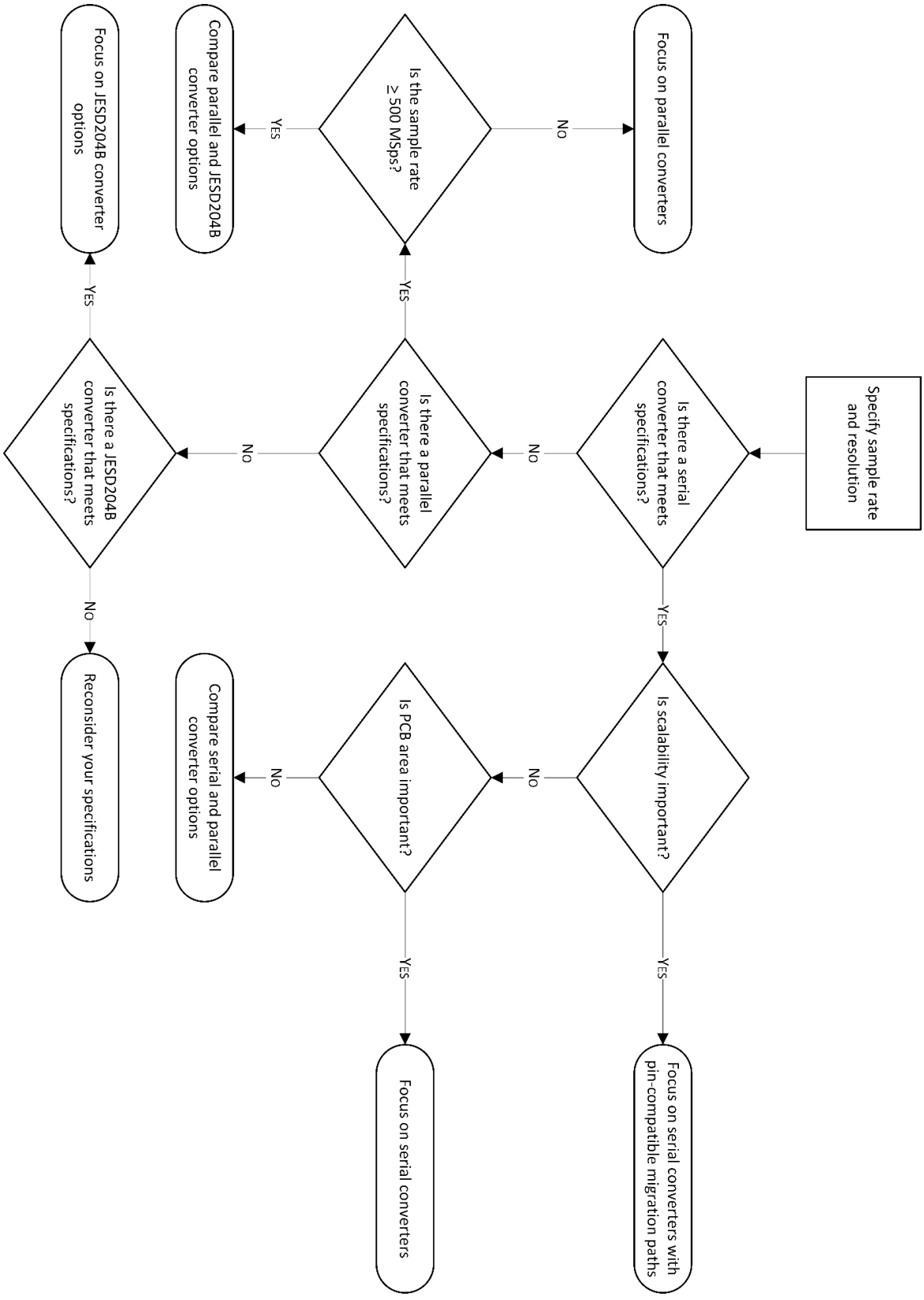
- JESD204B interfaces interface combines advantages of both the serial and parallel converters – requiring a minimal number of signals between the host FPGA and the converter while offering speeds that match (or exceed) parallel converters. The interface requires transceivers in the FPGA and IP to implement the protocol which is more complex than either serial and parallel interfaces. Circuit board requirements are more stringent too: Trace length matching, controlled impedance, and in some cases specialized materials (higher performance than FR-4) are required.

Based on these tradeoffs, the following flowchart is provided to assist in determining which interface may be the best fit for your new design.

About the author

Craig Holmberg has over 30 years of R&D experience designing electronics for telemetry, defense, digital communications, office imaging, video projection, and medical research. He is presently an engineering consultant doing business as Bitwise Design Labs. His core areas of expertise are FPGA design, PCB design, and signal processing. Leadership experience includes Director of R&D at a large company and multiple project leadership roles at other companies. He received a BS in Electrical Engineering from Virginia Tech in 1984.

Figure 11: ADC Interface Selection Flowchart



6 Appendix A: Abbreviations and Acronyms

ADC	Analog-to-digital converter
CML	Current-mode logic
CMOS	Complementary metal oxide semiconductor
DAC	Digital-to-analog converter
DDR	Double data rate
DPA	Dynamic phase aligner
FPGA	Field programmable gate array
FR-4	Flame-retardant grade 4
Gbps	10^9 bits per second
GHz	10^9 Hertz
I²C	Inter-integrated circuit
IP	Intellectual property
LVCMOS	Low-voltage complementary metal oxide semiconductor
LVDS	Low-voltage differential signaling
LVTTL	Low-voltage transistor-transistor logic
mil	1/1000 inch
mm	1/1000 meter
MSps	10^6 samples per second
PECL	Positive emitter-coupled logic
PLL	Phase-locked loop
PVT	Process, voltage, and temperature
SERDES	Serializer/Deserializer
SPI	Serial peripheral interface
TTL	Transistor-transistor logic