High speed serial link design (SERDES) Introduction, Architectures and applications

Abdallah Ashry[#1], Mohamed Alaa[#2]
Communication & Electronics Dept., Faculty of engineering, Cairo university
abdallah.ashry@outlook.com
mohammed.alaa.2016@gmail.com

Abstract **This paper describes basics methods of transferring data through serial data buses, using serializer/deserializer (SerDes) as main device in this operation. It explains function of SerDes and different techniques for implementing it.**

Keywords **SERDES, Clock and data recovery, High speed Serdes, Parallel clock SERDES, Embedded clock SERDES, 8b/10b SERDES, Bit interleaving SERDES**

I. Introduction

The simplest method of transferring data through the inputs or outputs of a silicon chip is to directly connect the datapath from one chip to the next chip. Since data often consist of more than one bit of information , the datapath is more than one-bit wide.

There are two inherent problems of the parallel data bus. The first problem is that *n* input/output (I/O) pins are required on each chip to transfer the data. The second problem involves meeting timing requirements. Due to these two problems we thought about transforming parallel data to serial stream of data to be able to use serial communications off-chips and on-chips.

Serializers/Deserializers (SerDes) are the main devices which convert parallel data into stream of serial data and send it through a channel .

Serial interconnects form the critical backbone of modern communication systems, so the choice of SerDes can have a big impact on system cost and performance. While the maze of choices may seem confusing at first, SerDes devices fall into a few basic architectures, each tailored to specific application requirements. A basic understanding of these architectural differences enables the designer to quickly find the right SerDes for the application. In this paper we examine four distinct SerDes architectures and show how each plays a vital role in today's systems.

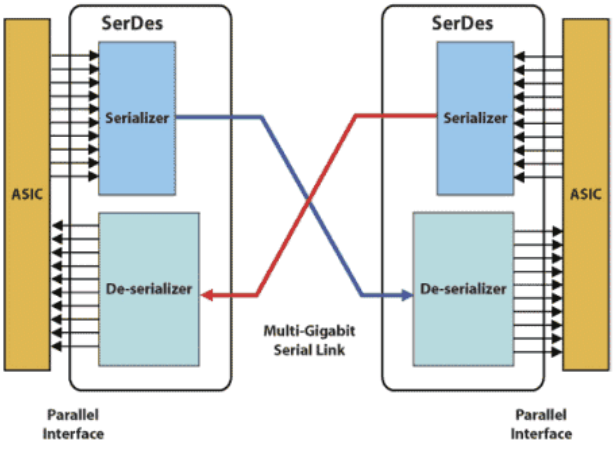

Fig.1 serdes main device

II. Architecture Of High-Speed Serdes

High-speed Serializer/Desrializer (HSS) devices are the dominant implementation of I/O interfaces at speed of 2.5 Gbps and higher. Such devices are differentiated from source-synchronous interface in that the receiver device contains a clock data recovery (CDR) circuit dynamically determines the optimal sampling point of the data signal based upon the transition edges if the signal. In other words clock information is extracted directly from the data rather than relying on a separate clock.

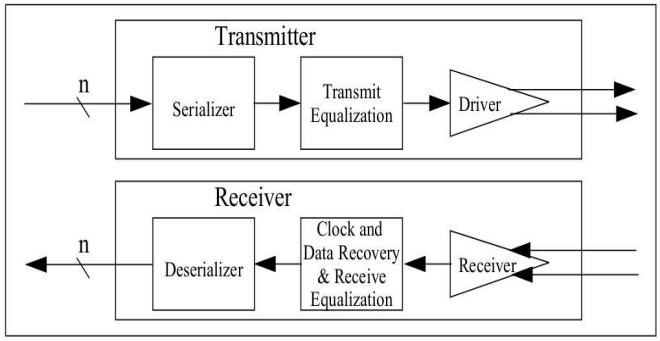

Fig.2 HSS basic block diagram

Fig.2 illustrate the basic block diagram of the transmit and receive channels of an HSS device. The transmitter serializes parallel data, equalizes it and then drives the serial data onto a differential signal pair of interconnect wires. The receiver consists of a differential receiver, a CDR circuit which may also integrate an equalizer and deserializes the data upon the sample point established by the CDR.

Serdes cores which contain only transmit or only receive channels are called *simplex* cores; Serdes cores which contain both transmit and receive channels are called *full duplex* cores. Note that the terminology "*full duplex*" does not imply that the electrical interface is bidirectional. Any given electrical interconnect channel has a fixed direction of data transmission. If a protocol application requires "*full duplex*" communication, then independent transmit and receive channels with independent interconnections are used to implement the interface.

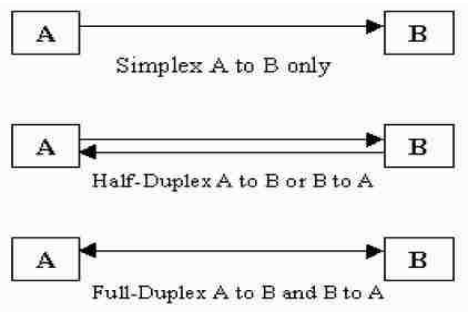

Fig.3 different types of serdes cores

A. Serializer/Deseializer Blocks:

The serializer stage latches data on the *n*-bit input at the frequency of baud rate/n. The high-speed clock in the Serdes is divided down to generate a sample clock for the parallel data. Because the phase of this clock is determined by the internal state of the serializer, the Serdes channel generally provides this clock as an output for use by logic driving data to the transmit channel.

Conceptually, the deserializer receive block performs the inverse function of the serializer block. Serial data is deserialized onto an *n*-bit databus of similar width to the serializer. A sample clock is generated by dividing down the internal high-speed clock, and this clock is supplied as an output for use by logic latching the parallel data. In a similar manner to the serializer, actual implementations may perform partial deserialization in a prior stage.

B. Equalizers:

Signal integrity concerns frequently dictate that the data signal be equalized at the transmitter and/or receiver in order to counter the effects of the channel and decode the signal properly. Many variations on filter architectures are possible, all of which accomplish this. Fig.4 illustrates the addition of an equalizer at the transmitter with a transfer function that is roughly the inverse of the channel's frequency response. This equalizer distorts the signal at the transmitter output such that the resulting signal at the receiver input is a clean waveform.

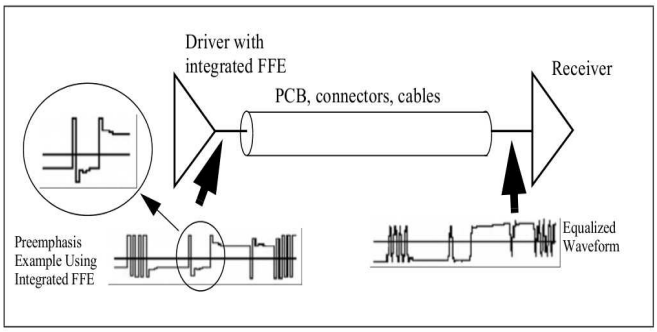

Fig.4 Equalizer proper operation

C. Clock and Data Recovery (CDR):

CDR circuits monitor transitions of the data signal and select an optimal sampling phase for the data at the mid-point between edges. Since the timing of data transitions includes a jitter component, the CDR must perform some averaging to provide stability of this sampling point from one bit to the next. Inter symbol interference (ISI) and other components of deterministic jitter (DJ) are dependent on the spectral content of the data signal, and this frequency spectrum does change based on the data content. Shifts in this frequency spectrum sustained for hundreds of bits or more cause the CDR to adjust the optimal sampling phase dynamically.

D. Differential Driver:

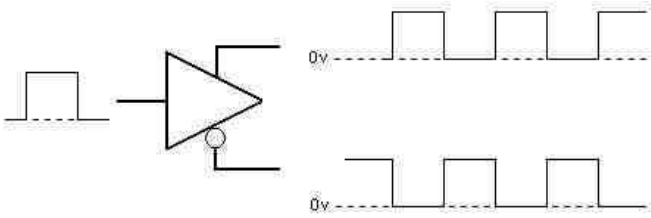

Fig.5 Differential Driver

The differential driver stage is an analog circuit which drives the true and complement legs of the differential signal. Output data must be driven such that jitter is minimized.

E. Differential Receiver:

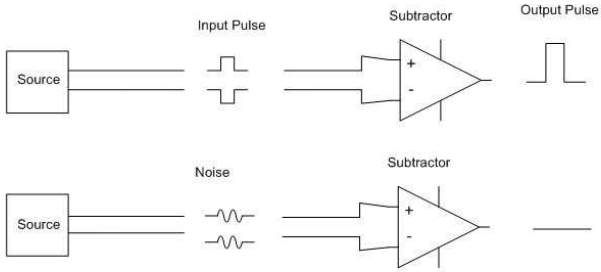

Fig.6 Differential Receiver

the differential receiver stage is an analog comparator circuit which compares the true and complement legs of the differential signal and outputs a "0" or "1" logic level based on the relative signal voltages.

III. TYPES OF SERDES

A. Parallel Clock SerDes:

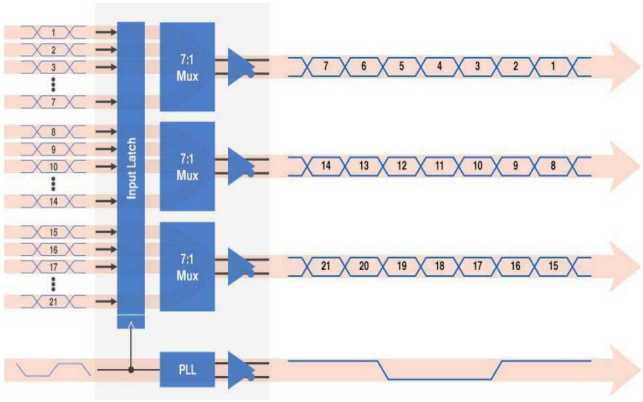

Fig.7 Parallel Clock SerDes transmitter

Parallel clock SerDes are normally used to serialize wide "data-address-control" parallel buses such as PCI, UTOPIA, processor buses, and control buses, etc. Instead of tackling the whole bus with one multiplexer, the parallel clock SerDes architecture employs a bank of n-to-1 multiplexers, each

serializing its section of the bus separately. The resulting serial data streams travel to the receiver in parallel with an additional clock signal pair that the receiver uses to latch in and recover the data. Since clock and data travel on multiple pairs, pair-to-pair skew must be minimized for proper deserialization.

Parallel clock SerDes offer excellent price/performance and are often the only practical way to transmit a traditional wide parallel bus over several meters of cable. Common parallel bus widths for these chipsets include 21-, 28-, and 48- bits.

B. Embedded Clock SerDes:

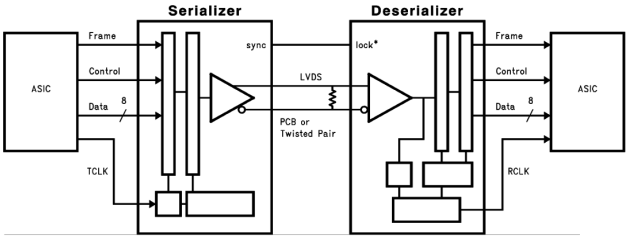

Fig.8 Embedded clock serdes

The embedded clock bits architecture transmitter serializes the data bus and the clock onto one serial signal pair. Two clock bits, one low and one high, are embedded into the serial stream every cycle, framing the start and end of each serialized word (hence the alternative name “start-stop bit” SerDes) and creating a periodic rising edge in the serial stream. Data payload word widths are not constrained to byte multiples; 10- and 18- bit widths are popular bus widths. After powering up, the receiver automatically searches for the periodic embedded clock rising edge.

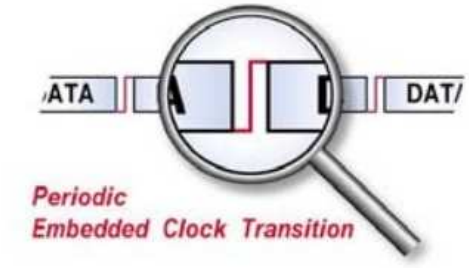

Fig.9 Periodic embedded clock transition

Since the data payload bits change value over time while the clock bits do not, the receiver is able to locate the unique clock edge and synchronize to it. Once locked, the receiver recovers data from the serial stream regardless of payload data pattern. This automatic synchronization capability is commonly called “lock to random data” and requires no external system intervention. This is an especially useful feature in systems where the receiver is in a remote module not under direct system control. Since the receiver is locked to the incoming embedded clock and not an external reference clock, jitter requirements for both transmitter and receiver input clocks are relaxed significantly.

Embedded clock bits SerDes are especially well suited to applications that transmit raw data plus other signals such as control, parity, frame, sync, status, etc.

Another feature of the embedded clock bits SerDes is automatic receiver lock to random data. This is an especially useful feature in systems where the receiver is in a remote module not under direct system control and also in systems where one transmitter broadcasts to multiple receivers. In the broadcast case, a new receiver module inserted onto the bus will lock to random data without the need to interrupt traffic to the other receivers by sending training patterns or characters.

Embedded clock bits SerDes are well suited to non-byte-oriented applications such as applications requiring transmission of unpacketized raw data plus control signals. Examples include signal-processing systems such as base stations, automotiveimaging/video and sensor systems where an analog-to-digital converter, camera or display communicates raw data with the processing unit at the other end of the link.

C. 8b/10b SerDes:

The 8-bit/10-bit (8b/10b) serializer maps each parallel data byte to a 10-bit code and serializes the 10-bit code onto a serial pair. The 10-bit transmission codes were developed by IBM Corporation in the early 1980’s and guarantee both multiple edge transitions every cycle as well as DC balance (balanced number of transmitted ones and zeros). Frequent edge transitions in the stream allow the receiver to synchronize to the incoming data stream. DC balance facilitates driving AC-coupled loads, long cables and optical modules.

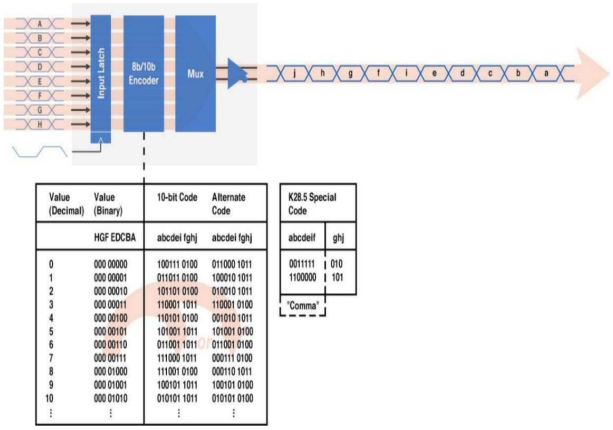

Fig.10 8b/10b serdes transmitter & encoder table of 10 bits

In order for the receiver to locate the 10-bit code word boundaries in the serial stream, the transmitter first marks one such boundary by sending a special symbol called a comma character. The unique bit sequence in this comma character never appears in normal data traffic and acts as reliable a marker for receiver code alignment. Once code alignment is accomplished, the receiver maps the 10-bit codes back to byte data, flagging an error if it detects an invalid 10b code.

Most 8b/10b deserializer architectures monitor lock by comparing the recovered clock frequency to an external reference clock. As a result, they typically require tight external clock source frequency and jitter control.

8b/10b SerDes are well suited to serializing byte-oriented data such as cell or packet traffic across backplanes, cable and fiber. Many standards such as Ethernet, Fibre Channel,

InfiniBand and others use the popular 8b/10b coding at rates of 1.0625, 1.25, 2.5, and 3.125 Gbps and many SerDes are available which span these data rates.

8b/10b coding has a maximum run length (the maximum number of consecutive ones or zeros in the serial stream) of 5 bits. This limits the spectral content of the serial stream that can ease the task of suppressing electromagnetic radiation. For example, given a 1 Gbps line rate after 8b/10b coding, the maximum and minimum 1 st harmonic frequencies are 1 GHz and (1 GHz)/5 = 200 MHz.

8b/10b coding also provides a way to check errors and send control information. As described earlier, most of the possible 10-bit code permutations are not valid 8b/10b data code words. This allows 8b/10b deserializers to flag invalid codes and provide a level of error checking similar to using a parity bit. While this scheme does not count total bit errors, it is a good way to monitor serial link performance. In addition to data code words, many standards also define control words such as packet/frame markers, fault flags, alignment characters, etc. These control code words help systems assemble and disassemble packets, making 8b/10b coding very popular in communications data processing systems.

D. Bit Interleaving SerDes:

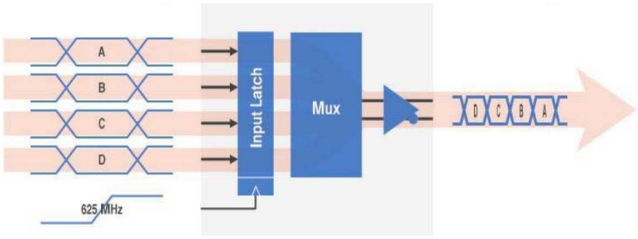

Fig.11 Bit interleaving serdes

Bit interleaving SerDes multiplex several slower SONET/SDH or 8b/10b serial streams into one faster serial stream by interleaving the bits. The receiver demultiplexes the bits back into the original slower streams. Note that a serial stream coming into transmitter input channel 1 may not come out on receiver output channel 1. This is not regarded as a problem in the applications because the serial streams contain independent cell or packet data that is processed downstream.

types of bit interleaving SerDes chips multiplex 8b/10b coded streams and are employed in switch and router equipment to get more bandwidth through existing backplanes.

IV. Conclusion

In this paper we discussed why we moved to serial bused instead of parallel buses, we made a speed overview on serialization/deserialization, we disussed main blockes in SERDES and we explained different types of SERDES.

References

[1] High Speed Serdes Devices and Applications., David R.Stauffer., Jeanne Trinko Mechler., Michael Sorna., Kent Dramstad., Clarence R.Ogilvie., Amanulla Mohammad., James Rockrohr. USA, Springer-Verlag.

[2] Shuguang Li Junyan Ren ', Lianxing Yang I Fan Ye' and Y.M. Michael Bang, “A Clock and Data Recovery Circuit for 2.5Gbps Gigabit Ethernet Transceiver” ASIC and System State Key Laboratory, Fudan University, Shanghai 200433 PR China , Vaishali LLC, Milpitas, California, 95035, USA

[3] Dave Lewis, “SerDes Architectures and Applications” National Semiconductor Corporation.

[4] Stephen Sunter, Aubin Roy, “An Automated, Complete, Structural Test Solution for SERDES” ITC INTERNATIONAL TEST CONFERENCE Ottawa, Canada.

[5] The IEEE website. [Online]. Available: http://www.ieee.org/

[6] [Online]. Available: http://en.wikipedia.org/wiki/SerDes