Vector divergence considerations for latency optimized high speed CDC

Patrick Fleming*, Rashid Iqbal, Niall McDonnell***, Pawel Ostropolski******

Shannon Design Centre
Intel Shannon co. Clare,
Ireland

**email: patrick.fleming@intel.com*
***email: rashid.iqbal@intel.com*
****email: niall.mcdonnell@intel.com*
*****email: pawel.ostropolski@intel.com*

As process geometries shrink and the demand for bandwidth increases the internal on-chip fabrics are run at several hundreds of MHz. These fabrics often interface to units which are operating in separate, asynchronous clock domains. With higher operational frequencies the intra-domain clock skew and inter-domain data path skew become significant factors, even if using historically safe gray encoding in CDC (Clock Domain Crossing), and need to be treated with more care. This requires proper modeling in functional verification as well as appropriate constraints for STA (Static Timing Analysis) and APR (Automatic Place & Route) tools. This paper discusses the challenges and proposed solutions for modelling and constraining the asynchronous CDC.

Keywords **– CDC, latency, divergence, asynchronous**

I INTRODUCTION

As process geometries shrink and the demand for bandwidth increases the internal on-chip fabric frequencies are run at several hundreds of MHz. The internal fabrics often interface to units which are operating in a separate, asynchronous clock domain. Queues and FIFOs are a commonly utilized method of messaging and data transfer between such clock domains where the read and write sides are timed in separate, unrelated clock domains. To ensure proper FIFO full/empty signalling the queue pointer from the read side needs to be synchronized to the write side and then decoded to produce a full signal and vice-versa from write side to read side to produce the empty signal. This paper discusses the challenges encountered in modelling and constraining the asynchronous CDC required for the FIFO pointer synchronization, but the related work and results are applicable to any multi-bit CDC.

CDC (Clock Domain Crossing) is being introduced in a synchronous (clocked) design, when a signal or group of signals is launched by a register(s) clocked in source clock domain and captured by other register(s) clocked in a destination clock domain. The source and destination clocks can be unrelated to each other in terms of frequency and/or phase or in other words – are asynchronous.
While single bit CDC synchronisation methods mainly aim to improve the immunity to metastability effects, multi-bit CDC must also ensure the correctness of the vector synchronisation. Several methods have been developed for vector synchronisation. These methods can be categorised as either closed or open loop methods.

Closed loop approaches e.g. the one presented in [1], are safe and very portable from design to design, but requiring control signals to roundtrip the CDC, and thus characterized by higher latency.

Open loop approaches on the other hand can be more latency efficient, but extra care needs to be taken in design and layout to minimize vector divergence. For the purpose of this publication the vector divergence is defined as a source_clock_skew + destination_clock_skew + data_path_delay_skew. An example representation of physical layout of CDC with large vector divergence is presented on Figure 1, where the solid lines represent wire lengths/delay.

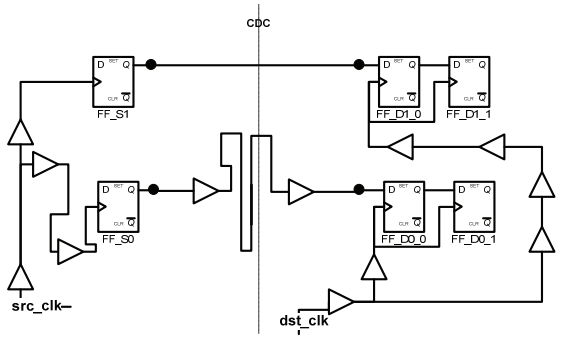

Figure 1: An example CDC with large vector divergence

The micro-architecture may require that every synchronous transition is transferred to the other side of the CDC, or just requires that the invalid vector states are not produced, and some of the intermittent transitions e.g. on fast to slow CDC can be skipped. The latter one is the property of the FIFO address pointers, which are discussed in more detail in the next section.

With higher operational frequencies the intra-domain clock skew and inter-domain data path skew become significant factors, even if using historically safe gray encoding in CDC, and need to be treated with more care. This requires proper modeling in functional verification and layout constraints, which are being discussed in the follow-on sections.

II LIMITATIONS OF GRAY CODED VECTOR CDC

The problem with the closed loop vector synchronization approach is the latency incurred – e.g. for an empty FIFO, from write to not-empty (NE) indication. It is possible to consider an optimized approach using gray codes and open loop approach. Gray codes have always been used to synchronize across clock domains, owning to their property of only changing a single bit in a given cycle. This eliminates the issue of transitional values being captured on the destination clock.

If the address pointers have to count in true gray codes the size of the FIFO must be a power of two. But this limitation can be somewhat overcome by using truncated gray codes for the address pointers. For an even sized FIFO of size 2m, find smallest n | $2n < 2m$, and use the 2m gray code values from $(2n - 2m)/2$ to $(2n + 2m)/2$. So for example the following constitutes a 12 value gray code sequence:

0011, 0010, 0110, 0111, 0101, 0100, 1100, 1101, 1111, 1110, 1010, 1011

This means the limit now restricts us to even size FIFOs only.

Instead of detecting a change in the source pointer and handshaking it across to the destination domain, the gray code source value can be synchronized across directly, minimizing latency. For improved MTBF (Mean Time Between Failures), a b2b (back to back) flip-flop is commonly used to synchronize signals in the destination clock domain.

Figure 2 shows an example FIFO structure and

Figure 3 presents an example scenario of generating the read side FIFO empty flag based on the write side address.

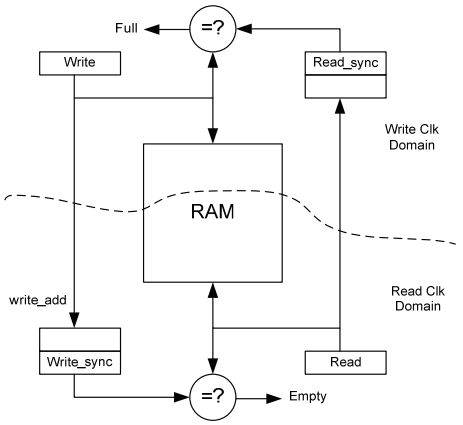

Figure 2: FIFO Based Clock Domain Crossing

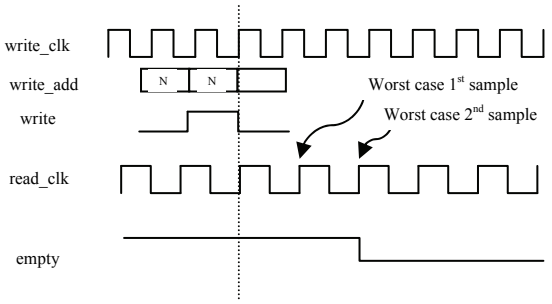

Figure 3: Fast to Slow CDC FIFO empty signal generation latency

A similar scenario exists for reads of a full FIFO propagating across to the write side to allow further writes.

With the ASIC approach where FIFOs are often a standard cell design with register file that are synthesized, placed and routed by an automated tool, the divergence needs to be controlled to ensure no invalid gray codes are present when synchronized in the destination clock domain of CDC.

This becomes important with truncated gray codes, where unexpected values can cause unpredictable design behaviour, as well as with the

full (2^n encodings) FIFO pointers where the out of order synchronized pointer value can lead in the best case to performance loss e.g. unexpected write side throttling (assertion of full signal) or more serious logical fault e.g. deasserting empty signal when there is no data in the queue. Figure 4 presents the potential effect of CDC divergence and metastability on a gray coded vector, where long wire delay causes a change of the sequence order of vector values in the destination clock domain.

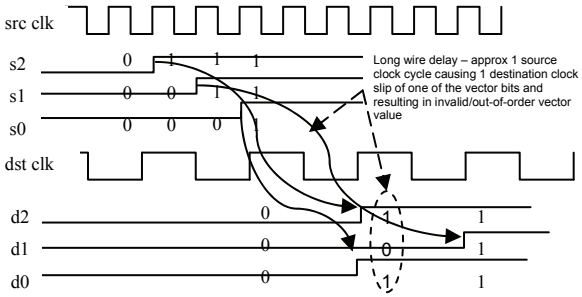

Figure 4: Divergence and metastability effect on multi-bit CDC

The next section of the paper presents an overview of the known methods of modelling vector divergence in pre-Silicon simulations, and a specific method proposed for the Gray code counters.

III MODELLING VECTOR DIVERGENCE IN PRE-SILICON VALIDATION

One of the methods of modelling vector divergence and resulting metastability effects in back to back FF synchronizers is Cycle Based Randomness, where the input signal in destination clock domain is randomly delayed by 2 or 3 clock cycles.

The main advantage of this approach is that it is frequency agnostic, thus advantageous from a reusability perspective. It does not require a frequency difference between asynchronous clock domains to detect synchronization problems in a design and it is useful for validating CDC on interfaces that rely on a small frequency offset between two clocks e.g. non-common clock high speed serial links, where the receive clock is recovered from a bit stream emitted by a transmitter clocked with a clock source unrelated to the receiver's clock source.

The aforementioned properties of the Cycle Based Randomness method will accelerate finding design faults, as the synchronization cycle delay randomness is not dependent on the clock edge relationship. As the cycle slip modelling is done on a per bit basis, this approach is very pessimistic regarding multi-bit vectors. This is specifically true for the multi-bit vectors on CDC, where the source domain clock is faster than destination clock domain. In this case each of the bits can be delayed by an extra clock cycle in the destination timing domain and can easily produce invalid values causing logic failure. This modelling approach, although quite pessimistic, still needs a certain amount of layout constraints to be applied for CDC paths, especially with higher frequency designs. These would need to ensure the vector divergence cannot exceed the slower clock period.

Another method, involves introducing a Random Transport Delay for the input signal to the first flop in the b2b FF synchronizer. The main advantage of this method is that it is very close to the realistic scenario in silicon. A disadvantage is that a meta-stable event is modelled only when the clock edges are within a specific window. The testing is limited to small windows on the interfaces that rely on small offset between transmit and receive clocks, e.g. non-common clock high speed serial links. The method also requires definition of a metastability window that needs to be tuned to the specific frequency to get more pessimistic modelling, as desired in validation. Modelling approach also needs a specific set of layout constraints to make sure the vector divergence across CDC is less than one source clock period.

Taking the advantages of the two aforementioned methods, the Mixed method is being proposed which utilizes the "Random Transport Delay" method timing constraints, but is pessimistic and accelerates finding bugs as in "Cycle Based Randomness" method.

As discussed in more detail in following sections, the design has to be constrained so that all timing paths between FFs in source clock domain and destination clock domain synchronizer's FFs are a maximum:

source_clock_period – destination_FF_setup_time – clock_skew_within_source_domain_FFs – clock_skew_within_destination_domain_FFs

This guarantees that when in fast source clock domain two signals that were launched one source clock cycle apart will never be captured at the destination domain synchronizer FF in reverse order, as it is illustrated in Figure 4. Additionally if both signal changes are seen prior to the destination clock sampling edge, the first signal is guaranteed to not enter a meta-stable condition due to the above timing constraint.

As RTL simulations are fully synchronous, with zero sim-time propagation time, the changes on the output from the source clock domain are

expected on the source clock edges. In effect only the last change seen on the signal from source clock domain since the last destination domain clock edge needs to be taken into consideration when randomizing the meta-stable condition occurrence in the destination domain synchronizer FF. This case is being presented on Figure 5

One of the limitations of the method is that if any hardcoded delays exist in the RTL code (e.g. as found in IO models), they will cause a non-zero simulation time propagation and create false changes – not corresponding to clock cycles.

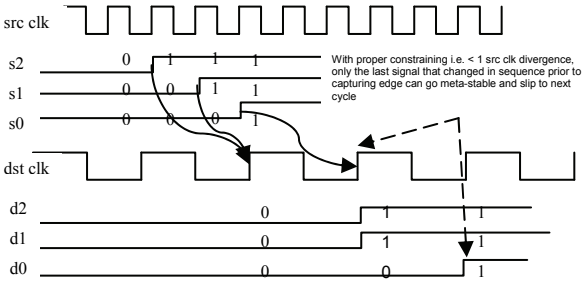

Figure 5: Multi-bit CDC modelling with constraints

The Mixed method is a moderately realistic approach, but is still quite pessimistic, while preventing invalid state transitions e.g. in gray encoded vectors. It is frequency agnostic, thus highly reusable. It does not need the frequency difference between asynchronous clock domains to test metastability immunity of the design and is useful for interfaces that rely on a small offset between transmit and receive clocks e.g. non-common clock high speed serial links.

IV CONSTRAINTS FOR IMPLEMENTATION AND STA

Although the path is asynchronous, timing constraints are needed for implementation and STA (Static Timing Analysis) to ensure a valid gray code is captured in the destination clock domain. For the captured gray code to be invalid, the total vector divergence would need to approach a single source clock period. The purpose of the constraints therefore is:

- For physical design tools, to drive the logic synthesis, CTS (Clock Tree Synthesis) and APR stages
- For STA, to ensure the divergence does not exceed allowed amount

Various constraints methods were looked at and are outlined hereafter.

a) Max Delay Method

This method sets a clock skew target for the source and destination clocks and applies a max delay across the vectors' data path. The sum of the clock skew target and max_delay are constrained to be less than the period of the source clock domain. These constraints apply an absolute delay on the data path rather than constraining the divergence only and can therefore be pessimistic constraints. Figure 6 explains this method. The disadvantage of this approach is that the timing requirement is split into two parts (set_max_delay & clock skew target). Also, applying set_max_delay removes the normal timing path reporting in the tool, which means that proper timing verification can only be done with custom scripts after layout is done. Also set_max_delay constraints need to be changed before and after CTS stages to include or exclude CTS effects. Another drawback with this approach is that if a timing problem is seen in the post-layout timing verification (due to inaccurate constraints modelled in the implementation tools), it will require additional design iterations.

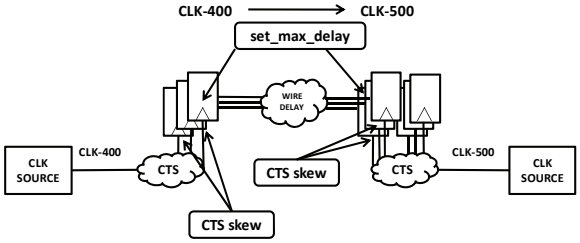

Figure 6: Constraining for max delay

b) Additional Clock Method

The explanation of this approach is given in Figure 7, with two asynchronous clocks CLK-400 and CLK-500. For paths with clk-400 as launch clock and clk-500 as capture, an additional clock CLK-400-ADD can be defined at the source pin of CLK-500 with period of CLK-400. After applying set_false_path from CLK-400 to CLK-500, the only remaining paths will be the paths from CLK-400 to CLK-400-ADD. This will ensure required time to be the period of source clock and will consider CTS skew effect. Similarly for paths from CLK-500 to CLK-400 an additional clock is defined at source pin of CLK-400 with period of CLK-500. Setting false path CLK-500 to CLK-400 will make correct timing requirement.

The following are the advantages with this approach:

- Same constraints for implementation and verification
- Same constraints commands before and after CTS. However the actual uncertainty values before CTS needs to model skew numbers which should be removed on the post-CTS database.
- No custom scripts required to identify all the vectors to be constrained with max_delay.

This method will double the number of clocks, which may be appropriate for few clocks but for a design with large number of clocks this will increase the complexity. Additional clocks need new exceptions and appropriate handling in CTS.

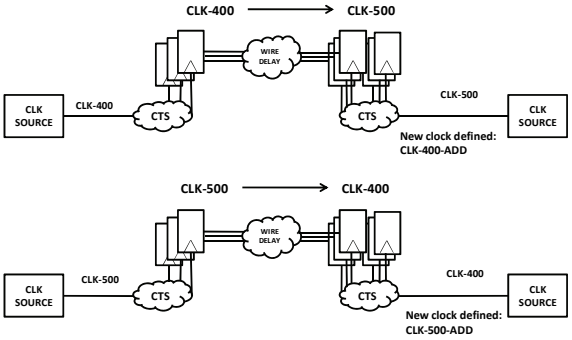

Figure 7: Constraining using additional clock

c) Constraints with negative clock uncertainty

This method uses negative clock uncertainty between two asynchronous clocks. A negative uncertainty is used to make the required time equal to the period of the launch (source) clock. Figure 8 explains this method with two clocks clk-400 and clk-500 To derive timing relationship between two different clock frequencies, the timing engine calculates the Least Common Multiple (LCM) of the two clocks called base period, expands the two clocks and finds the worst setup edge (from launch clock to capture clock) within the base period. For a timing path with source clock of clk-400 and destination clock of clk-500; the worst setup edge/timing is from 7500ps to 8000ps. This gives required interval of 500ps. Since the target is to make the required time to be the 'cycle time' of the source clock (which in this case is 2500), 2000ps more is needed. This additional time was applied as a negative clock uncertainty from clk-400 to clk-500. Similarly for a path from clk-500 to clk-400, the worst interval is again 500ps (2000 -> 2500) which require negative uncertainty of 2000ps.

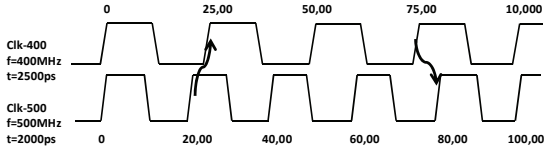

Figure 8: Two asynchronous clocks with worst case setup edges in both directions

The two example commands for applying negative clock uncertainty are given below:

set_clock_uncertainty –from clk-400 –to clk-500 –setup -2000
set_clock_uncertainty –from clk-500 –to clk-400 –setup -2000

After applying above commands, an example timing report from the tool, for clk-400 to clk-500 path is shown below:

```
Point                                          Incr       Path
-----------------------------------------------------------------
clock clk-400 (rise edge)                   7500.00    7500.00
clock network delay (ideal)                    0.00    7500.00
reg1/ck
..
..
..
..
Reg2/d    data arrival time                            7555.70

clock clk-500 (rise edge)                   8000.00    8000.00
clock network delay (ideal)                    0.00    8000.00
reg2/ck                                        0.00    8000.00 r
library setup time                           -76.15    7923.85
data required time                                     7923.85
-----------------------------------------------------------------
data required time                                     7923.85
data arrival time                                     -7555.70
-----------------------------------------------------------------
slack (MET)                                             368.15
```

This method has following advantages over the previously described approaches:

- Same constraints for implementation and verification
- Same constraints commands before and after CTS.
- No extra clocks are required.
- No custom scripts required to identify all the vectors to be constrained with max_delay.

V RESULTS

The above methods are the outcome of analysis done during the development of the multi-clock domain chip. The worst case clock crossing for FIFO synchronization was found to be from 533MHz to 125MHz. It was found necessary during development to constrain the APR of this crossing using the max_delay method. Post CTS CDC analysis was done to verify the integrity of the vector synchronization. The timing analysis indicated that:

- When all synchronization vectors were grouped together, divergence margin across the entire group was 50ps.

- When analyzing each vector individually the margin was found to be 500ps.
- The 533MHz clock skew reported across all above endpoints was 267ps, while the 125MHz clock skew was 1951ps.

As expected, the issue was far more pronounced in the slow corner. It is evident that the skew of the slower clock is the main contributor to divergence – while this skew might be more than adequate for timing within the slow clock domain, it becomes a problem with gray coded cross clocks. In the end the margin available to the chip was considerable once correct APR constraints were applied.

VI SUMMARY

Simple gray code based vector passing can be used in most FIFO based clock boundaries. But if the clock periods are short enough for clock and data vector path skew to be significant, constraining the design requires a bit more care to ensure correct operation under all conditions. A constraint must be applied to limit the maximum vector divergence to ensure only correct gray code sequences are passed from one domain to the other. Ideally such a constraint can be used to drive the APR tools and this paper has shown how this can be done using an approach based on adjustment to clock uncertainty. Once the constraint has been determined, it can be modeled in RTL simulation to verify correct operation for added reassurance. Several modeling methods are discussed and compared.

ACKNOWLEDGMENTS

The authors would like to acknowledge their colleagues from Shannon Design Centre: Paul Waters, David Keane, Shane O'Neill and Damian Finnerty and others for their contributions to the methodology development and verification.

REFERENCES/BIBLIOGRAPHY

[1] Cummings, C. E. (2008). Clock Domain Crossing (CDC) Design & Verification Techniques Using SystemVerilog. SNUG. Boston: Sunburst Design Inc.