

XS1 Port I/O Timing Application Note

REV 1.1

2013/05/16
XMOS © 2013, All Rights Reserved.



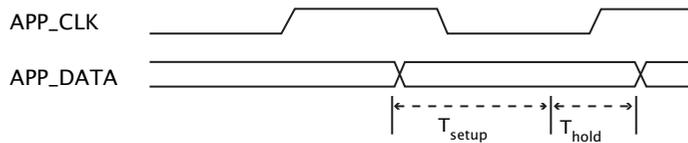
1 Introduction

XS1 devices provide the ability to delay the application clock with respect to incoming data, to provide a valid data window for sampling data on XS1 ports.

This application note explains how to calculate the valid data windows for an XS1-G4 device and the considerations that need to be taken into account when adjusting the application clock.

2 Input Timing

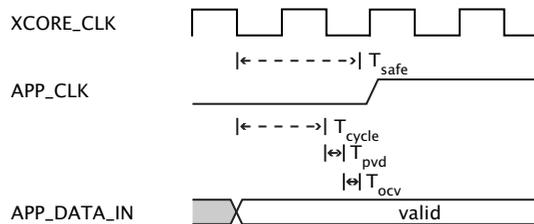
The valid data input window for an application is calculated as the sum of the input setup time and the input hold time with regard to the application clock.



APP_CLK and APP_DATA are sampled each core clock cycle, with the values proceeding down parallel pipelines. The overall delay applied to each may differ such that APP_CLK may be moved forwards and backwards with respect to APP_DATA.

The sampling process introduces some positional uncertainty of the sampled version of APP_CLK (APP_CLK'), so APP_CLK' should be positioned such that is not too near the leading edge, or the trailing edge, of the data valid window.

APP_CLK' needs to be positioned at least T_{safe} away from the leading edge of the valid window:



T_{safe} is comprised of the following components:

- ▶ T_{cycle} : The first xCORE Tile clock cycle which may just miss the APP_CLK or APP_DATA signal changing.
- ▶ T_{pdv} : The variability of pad propagation delays primarily due to the difference in rise and fall delays through the pad, and differences in input slew rate.

- ▶ T_{ocv} : The variability of the relative delay of clock and data between the pad and the first on-chip sampling registers.

Similarly APP_CLK needs to be positioned at least T_{safe} away from the trailing edge of the valid window due to the uncertainty (equal to T_{safe}) in its position.

If both constraints are met for a given valid window size and setup/hold time combination, there is no need for any relative delays to be applied to APP_CLK and APP_DATA.

If the constraints are not met, however, the clock needs to be moved by T_{safe} .

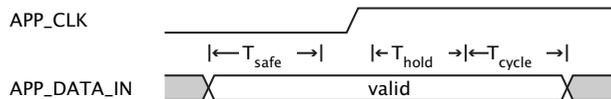
From this the overall valid window size required for a guaranteed correct operation can be derived. The worst case scenario occurs where $T_{hold} = T_{safe} - 0.1$. In this case the clock must be moved one xCORE Tile cycle towards the leading edge of the valid window, yet still remain at least T_{safe} away from the leading edge of the window. Consequently the total width of the valid window required to accommodate all these constraints is:

$$T_{safe} + T_{hold} + T_{cycle}$$

where:

T_{cycle} = xCORE Tile clock cycle plus jitter (for 400MHz operation this is 2.5ns plus 70ps jitter = 2.57 ns).

$T_{safe} = T_{cycle} + T_{pdv}$ (900ps) + T_{ocv} (250ps) = 2.57 + 1.15 = 3.72



Therefore the minimum valid window that will guarantee correct operation for any given initial positioning of APP_CLK in the valid window is (as T_{hold} tends to T_{cycle}):

$$T_{safe} + 2 \times T_{cycle} = 3.72 + 2.57 + 2.57 = 8.86 \text{ ns}$$

Note that smaller valid windows may be possible depending on the required T_{hold} and T_{setup} .

Note also that either the rising or falling edge of the incoming APP_CLK can be used to generate the rising edge of APP_CLK' internally.

3 Output Timing

Output data is launched by a sampled and arbitrarily delayed version of the incoming APP_CLK signal (APP_CLK'), which means any required hold time can be

met (subject to a granularity of one xCORE Tile clock cycle). There is an uncertainty of T_{cycle} as to the exact position of APP_CLK' due to the sampling uncertainty of the incoming APP_CLK signal.

This is summarised together with other delay factors in the table below:

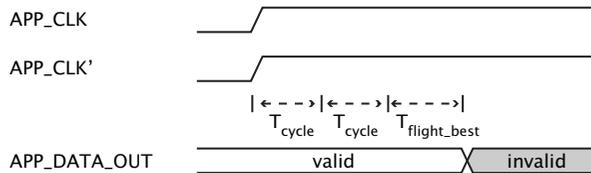
Factor	Best Case	Worst Case
APP_CLK	Sampling uncertainty 0	$1 \times T_{cycle}$
Latency between APP_CLK' and data launch	$2 \times T_{cycle}$	$2 \times T_{cycle}$
Flight time to pin	T_{flight_best}	T_{flight_worst}

The flight time to pin is summarized in the table below:

pF _{load}	1	5	10	20	
T_{flight_best}	1.588	1.707	1.857	3.147	ns
T_{flight_worst}	5.346	5.766	6.295	7.422	ns

3.1 Best case output scenario (APP_CLK delay of 0)

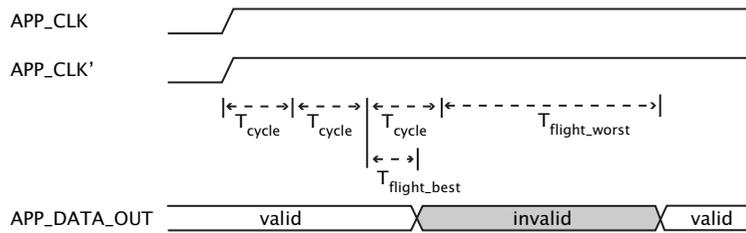
The fastest possible output case is when APP_CLK' is not delayed and the output flight time is at a minimum ($2 \times T_{cycle} + T_{flight_best}$).



If $2 \times T_{cycle} + T_{flight_best}$ exceeds the required T_{op_hold} period, no clock delay adjustments are required.

3.2 Worst case output scenario (APP_CLK delay of 0)

The worst case scenario is when APP_CLK' is not delayed with respect to APP_CLK:



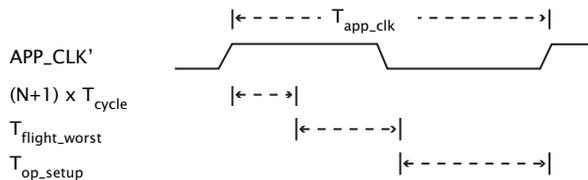
In this case the longest time output data may take to become valid after APP_CLK rises is:

$$3 \times T_{\text{cycle}} + T_{\text{flight_worst}}$$

3.3 Calculating output setup time

The setup time offered by the output data can be calculated as:

$$T_{\text{app_clk}} - ((N+1) \times T_{\text{cycle}}) + T_{\text{flight_worst}}$$



where N is the number of T_{cycle} delays added to APP_CLK' to meet the required hold time (see *Input Timing* - Section 2).

Note that $T_{\text{flight_best}}$ and $T_{\text{flight_worst}}$ incorporate all on chip and pad related variability (T_{pdv} and T_{ocv}) and represent genuine best and worst cases.

The cumulative delay of $3 \times T_{\text{cycle}} + T_{\text{flight_worst}}$ may leave insufficient time to meet application setup timing requirements. If this case APP_CLK' may be positioned so that it occurs some time before APP_CLK, such that the addition of $T_{\text{flight_best}} + 2 \times T_{\text{cycle}}$ to APP_CLK' still meets output hold time requirements with respect to APP_CLK, and setup time is also met.

The above can be accomplished by sampling the falling edge of APP_CLK and then delaying that such that APP_CLK' ends up positioned just before the *following* APP_CLK edge. This scheme is safe as long as APP_CLK is free running and will not be stopped or drop cycles.

4 Turnaround Timing

Many applications (such as USB 2.0) specify turnaround timing constraints, which are defined as a constraint on the time an event is seen at the input pins of the device, to a given response being output from the device. The value is usually specified as a fixed time, or as a number of application clock cycles. This section explains how to calculate the response of the XS1 device in such situations.

The path an input signal takes from input to output is as follows:

1	Input pad delay	1 ns*
2	Input resynchronization flip flops	$2 \times T_{\text{cycle}}$
3	Sampling uncertainty	$1 \times T_{\text{cycle}}$
4	Input port delay	$1 \times T_{\text{cycle}}$
5	Internal processing in the xCORE Tile	$N \times T_{\text{cycle}}$
6	Output port delay	$\text{TBD} \times T_{\text{cycle}}$
7	Output flight time	See Section 3 ($T_{\text{flight_worst}}$)

The total turnaround time is the sum of the delays items 1-7 in the table above.

The value of item 5 is dependent on the software running on the xCORE Tile. The XTA toolset can help to determine this number as a worst case.

5 Further Information

If you have any doubt as to whether the XS1 ports can be configured for the particular timing requirements of your application please contact XMOS support.



Copyright © 2013, All Rights Reserved.

Xmos Ltd. is the owner or licensee of this design, code, or Information (collectively, the "Information") and is providing it to you "AS IS" with no warranty of any kind, express or implied and shall have no liability in relation to its use. Xmos Ltd. makes no representation that the Information, or any particular implementation thereof, is or will be free from any claims of infringement and again, shall have no liability in relation to any such claims.

XMOS and the XMOS logo are registered trademarks of Xmos Ltd. in the United Kingdom and other countries, and may not be used without written permission. All other trademarks are property of their respective owners. Where those designations appear in this book, and XMOS was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.