mips

Open RISC Technology

Guidelines for MIPS R10000 Microprocessor I/O Timing Calculations

Shabbir Latif

MIPS Technologies, Inc.
www.mips.com

Published date: July 25, 1996

For more info on Mips and it's semiconductor partners visit "***http://www.mips.com***"

Introduction

1

MIPS RISC *R10000* is a high performance microprocessor. The *R10000* inputs and outputs (I/Os) are designed so that the processor can be used in variety of systems. The processor can interface with HSTL technology in a high performance system; or it can interface with CMOS/TTL technology in a low end system. The outputs of the system interface can also be configured as open drain in a system with cluster bus configuration. Maximum performance and flexibility built into the interface design necessitates certain special system design requirements.

The purpose of this document is to describe a sample methodology to show how the parameters provided can be used for calculation of *R10000* I/O timing that can be used for the system design. Especially, how the adjustment of secondary cache clock (*SCClk*), using *SCClkTap*(Reference 1), fits into the timing calculation. It should be clear that due to the transmission line, loading affects, etc. on the output buffers driving them, the best (and sometimes only) way to design is via circuit modelling and simulating the behavior. The reason for piecewise delay equations shown in this document is to allow "back of the envelope calculations" and to give an idea of how the parameters specified fits into the calculation. This document is just a guideline and is not intended to provide a knowledge in high performance system design. The material should be used carefully by an experience system designer.

Timing Calculations:

2

The external interface for the *R10000* is designed for register-to-register operation and the signals are driven and latched in by a specific interface clock. Even the secondary cache is required to be Synchronous Static RAMs (SSRAM). In case of the system interface the *SysClk* is used as a clock and in case of the secondary cache interface the *SCClk* is used as a clock. Each of these can be independently programmed, during boot time, to operate at various frequencies (Reference 1).

Due to the register-to-register operation, the total time available for the signals to travel, between the processor and an external component is, generally, one clock period and the signal must be held at the input of the register for the required hold time. This determines the minimum time to travel.

There are some differences between the *system interface* clocking and the clocking of the *secondary cache (SCache) interface*. The System Interface is *fully synchronous*; that means, the *system interface* clock (*SysClk*) is generated outside the processor and is used to trigger both processor I/O and the external system logic. The processor also uses the *SysClk* input to generate *PClk* for clocking the internal core logic and the *SCClk* which is driven outside to control the timing to interface with the Synchronous Static RAM (SSRAM). The frequency of the *PClk* and the *SCClk* depends on the *SysClkDiv* and *SCClkDiv* mode bits, (see Reference 1 for more detail on Clocks). This means secondary cache interface is *source synchronous* since the R10000 provides the *SCClk*, with the Address/Data, for the SSRAM timing. Moreover, on the secondary cache interface, the *R10000* provides a little more flexibility by allowing to adjust the clocks driven externally (*SCClk*'s) with respect to the clock edge that drives or samples the signals internally to the processor. Considering this feature the timing for the *SCache interface* is less restricted; such that, two clock periods are available for the round trip of the signals. That means the time slack available in one direction can be utilized in the return direction. Although, as shown later, the amount of *cycle stealing* is limited by the hold time requirements of both the SRAM and the *R10000*.

Although there is an internal PLL to align the necessary clock edges, there is always some uncertainty due to the minimum resolution, VCO duty cycle and the jitter of the PLL and the skews.

Note also that a PLL internal to the SRAM is unnecessary and not recommended. As it is shown later, to determine the minimum delay (which is limited by the *hold time)* one doesn't need the processor PLL jitter term; however, if there is a PLL on the SRAM, that PLL jitter would need to be included, which would worsen the *hold time*.

The next section lists different components of the *clock uncertainty* and their contribution to various timing paths. Later section shows how to calculate the *system interface timings* and the last one describes the *secondary cache timings*.

In this document the term *max path* is used for the timing path that determines the maximum delay which determines the maximum interface clock frequency; and the term *min path* is used for the timing path that determines the available *hold time*.

Clock Uncertainty

One of the key factors in the timing equations is the clock uncertainty. The *table 2-1* lists the components that can contribute to the clock uncertainty on *the system interface* and the *table 2-2* shows the components that affect *the secondary cache (SCache) interface*. One of the important components in determining the maximum and minimum delays is the difference between the clock that is seen externally verses the clock that actually triggers the data going out and latches data coming into the chip. As described in section for *SCache Interface Timing*, this factor is also dependent on the setting of the *SCClkTap* mode bits and so is the clock uncertainty.

Table 2-1 Clock uncertainty for the System Interface†

Uncertainty Sources	**Cycle-Cycle Jitter**	**Peak-Peak Jitter**	**Skew**	**Static Phase Offset**	**Units**
Jitter due to external components.‡	t_{sycc}	t_{sypp}	t_{sysk}	t_{syspo}	psecs
PLL R10K*	500	1000	400	200	psecs

‡ Since the numbers depends on the external components, they are represented here by the variables.

* These values are measured under noise injection and are assumed worst case system values; as opposed to the standard numbers (150ps) taken with clean lab conditions.

Table 2-2 Clock uncertainty for the SCache Interface

Uncertainty Sources	**Cycle-Cycle Jitter**	**Peak-Peak Jitter**	**Skew (ISCClk-RSCCLk)**	**Static Phase Offset**	**Units**
PLL R10K	500		400	200	psecs
R10k SCLK tap delay half-quantization error			280		psecs

† The number given in both of these tables are preliminary. The finalized numbers will be provided in the data sheets from the chip suppliers.

Uncertainty Sources	Cycle-Cycle Jitter	Peak-Peak Jitter	Skew (ISCClk-RSCCLk)	Static Phase Offset	Units
R10K SCLK tap delay variation			200		psecs
R10K SCClk buffer clock to out delay variation with process			200		psecs
PWB & Package			50		psecs
For SCache Maxpath	500	NA	1130	NA	psecs
For SCache Minpath	NA	NA	1130	NA	psecs

Cycle-Cycle Jitters - describes the random variance in time between a clock edge and the next adjacent clock edge one period away.

Peak-Peak Jitter - describes the random variation in time between a clock edge and another clock edge more than one period away.

Clock Skew - describes the time of arrival difference between two clocks due to factors other than the PLL.

Static Phase Offset - describes the delay difference associated between the reference clock and the feedback clock of a phase detector (normally associated with either the PLL or a delay lock loop).

Tap Delay Quantization Error - is due to the resolution provided by the circuit which determines how close it can place the *SCCLK* with respect to the *ISCClk* compare to what is defined by the mode bits. For example, the *PClk* period divided by 12 for 150MHz (6.67nS cycle) is 560ps. Worst case uncertainty is half of 560ps which gives 280ps.

Tap Delay Variation - Any variation of the actual tap points due to variation in supply, process, etc.

PWB & Package - The uncertainty is system dependent and it is assumed to be 50psec in this example.

The clock uncertainty for each path is lumped into a single term, t_{ckjs} in the timing equation shown in later sections. For the system interface, the uncertainty is attributed a lot by external PLL, oscillators, etc. Thus, it is difficult to show it as a number. The clock uncertainty for both maximum and minimum path can be derived as shown in equation below:

t_{ckjs} for System Interface = t_{sypp} + 1/2(*R10KPLL* $_{p\text{-}p}$ jitter) + t_{sysk} + $R10KPLL_{skew}$+ t_{syspo} + $R10KPLL_{spo}$.

To provide higher noise margin SGI has developed a tighter HSTL specifications. Several vendors, such as IBM and LSI Logic, have agreed to meet this specifications. The document describing the modified SGI HSTL specifications is available on the *R10000* page at the MIPS web site (*http://www.mips.com*).

Considering that the SSRAMs does not include a PLL, all the uncertainty components, except the PWB and package, used for the SCache interface timings, are R10000 dependent. Thus, assuming the PWB & Packaging uncertainty to be 50pS, the uncertainty numbers for both *max path* and *min path* calculations would be 1630pS and 1130pS, respectively, as shown in the *table 2-2*. Note again that the jitter term doesn't affect the *min path* (hold time) calculations.

Often using the algebraic sum for the clock uncertainty on minpath results in an unworkable numbers. In such a situation one can use a statistical method of summing clock uncertainties that take into account the likelihood of all uncertainty numbers being at their worst case values simultaneously. In the above case the algebraic sum for minimum path is 1130pS; while Statistical sum (RMS) would be 566pS and the (RMS+Algebriac Sum)/2 will be 848pS.

System Interface Timing

Figure 2-1 shows a representative path on the system interface and the timing waveform. The equations for the maximum and the minimum time are given in *equation 1 and 2*†. The timing reference point for all the timing terms is the clock at the output buffer, as shown by T=0 in *figure 2-1*.

Eq 1: $t_{per} \geq (t_{do} + t_{pd} * L + t_{LF} + t_{ckjs} + t_{dns} + t_{dsu})\text{Max}$

Eq 2: $t_H \leq (t_{do} + t_{pd} * L + t_{LF} - t_{ckjs})\text{Min}$

t_{per} = Clock Period

t_{do} = Output buffer delay from Clock to the output pin, unloaded & under nominal condition.

t_{pd} = Intrinsic interconnect propagation delay per unit length.

L = Net length between the driver pin and the receiver pin.

t_{ckjs} = Maximum clock uncertainty for the path (due to jitter, skew, etc.)

t_{LF} = Maximum timing delay due to loading for the given net topology (This is simply the total interconnect delay minus the transmission line propagation delay from driver pin to receiver pin).

t_{dns} = Settling time for any *dynamic noise* that falls outside the dynamic noise margin‡.

t_{dsu} = Receiver pin setup time.

t_H = Receiver pin hold time.

In *figure 2-1*, the T_{DOX} includes $t_{do} + t_{dns}$ while t_{pdx} and t_{LFx} are the propagation delay and the delay due to load factor, respectively, for the net with length L_x. The components in the equation that the system designer need as a processor specification is the *SysClk* to *Data Out delay* (T_{do}), data *setup time* (t_{dsu}), data *hold time* (t_H) and the *clock uncertainty* (T_{ckjs}). The data setup, hold time and *clock uncertainty* values are easy to specify and will be given in the data sheet by the chip suppliers. However, the output buffer drive delay depends on the load driven. Up until recently, specifying the largest and the smallest delays for given lumped load components were sufficient. However, now with the fast edge rates (*R10000* edge rates could be as fast as 300ps) the signals are driven and the small cycle time required for very high performance systems, this is not enough. In case of high performance systems for which *R10000* is designed, nothing short of circuit simulation (e.g. *SPICE*) with modelling of the output buffer, package load, socket load, the transmission line on the board, and the receiver load will give you the accurate information. MIPS and the partners will provide the output buffer model with all the necessary parameters that can be used in *SPICE* simulations. However, an expert system designer may be able to use other methods for a relatively low performance system or at least, to get good estimates, before getting to the state where simulations can be performed.

† These are sample equations. In reality, there might be other components that need to be considered depending on your system.

‡ The *dynamic noise* could be due to reflections, simultaneous switching noise, cross talk, etc. The value of t_{dns} would be determined by the overall system noise margin and noise allocations. This is very system dependent. For additional information see references [2, 3, & 4] at the end of this paper.

Although, the figure *2-1* shows the path going from *R10000* to an external receiver, the calculation of the timing of a path in the reverse direction is also very similar and thus not shown here.

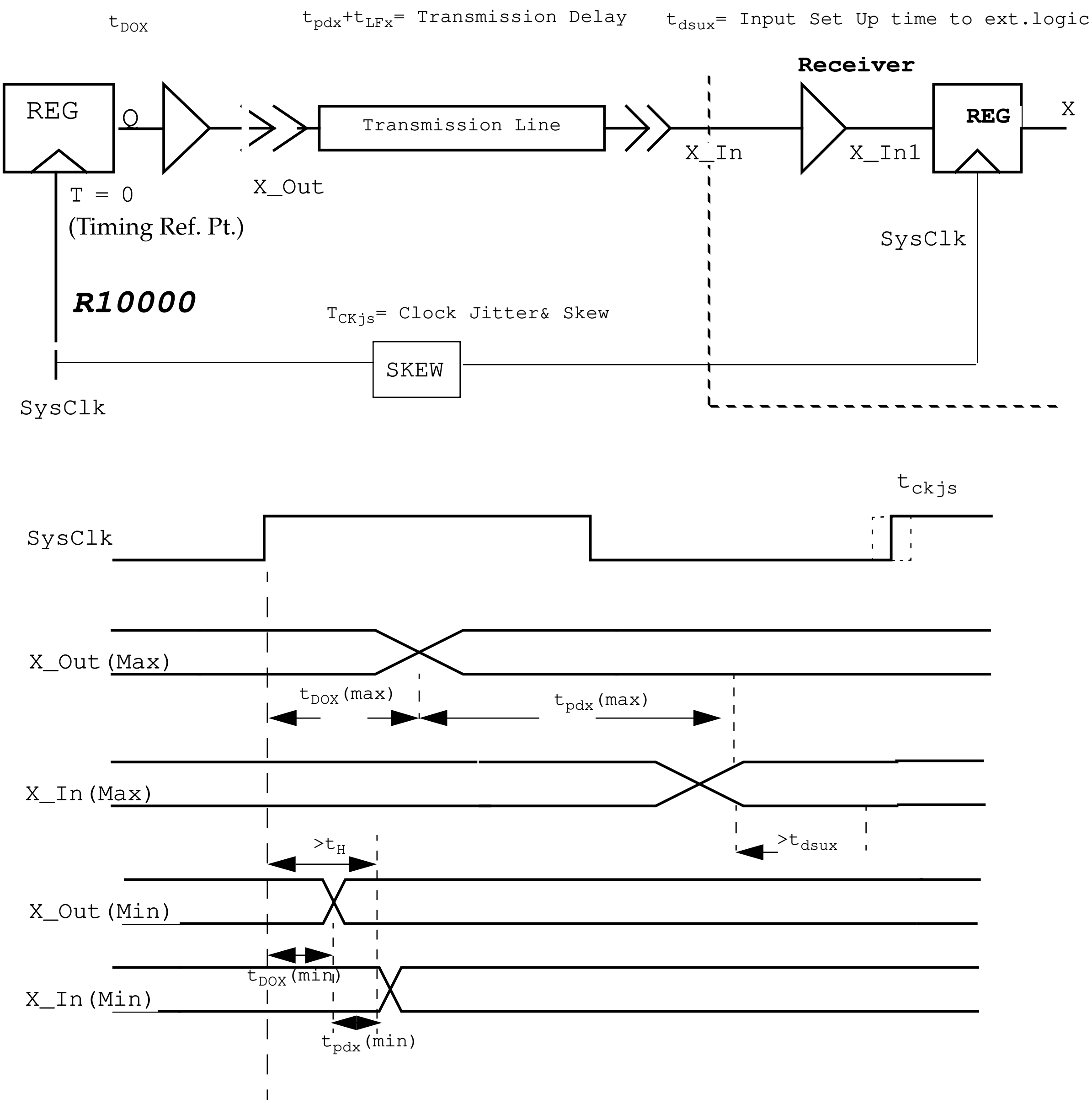

Figure 2-1 Example of a path from R10000 to an External Component on the System Interface

Secondary Cache Interface Timing

Figure 2-2 shows the representative path and the timing waveform for the secondary cache interface. Besides having its own clock, *SCClk*, the secondary cache interface provides programmable adjustment of the external *SCClk*† signals with respect to the Internal *SCClk* (*ISCClk*). This adjustment can be set, during boot time, via "*SCClkTap*" mode bits [Reference 1]. In *figure 2-2*, the "*SCClkTap*" controller is shown, conceptually, as a negative delay element. The earlier clocks are brought out from the PLL and selected by the *SCClkTap* (as shown by *ESCClk*). It is not critical to know the actual details of the generation of clocks. When the tap adjustment is set to 0 (*SCClkTap* = 0), the edges of the *SCClk* and *ISCClk* are aligned‡. The *SCClk* can be adjusted earlier with respect to ISCClk in the increments of $1/12^{th}$ of the PClk (internal core clock) period. These delay numbers (*AdjN*), for various *SCClkTap* settings and *PClk* frequencies, are provided in *Table2-3*. There are several potential benefits to adjust the *SCClk* earlier than the *ISCClk*: a) to loosen the hold time on the SRAM side; and b) to use the slack available during the time of the signal traveling from the processor to the secondary cache, for the return path. Although, this adjustment will tighten the *hold time* margin at the processor side. Thus, the timing requirements, as shown in equation 3 to 7, are similar to those shown for the system interface timing; however, a delay term, t_{tap} is included in the equations which is determined by the *SCClkTap* setting. Also, since there is only one PLL (assuming SRAM is without PLL), the jitter doesn't have to be included in the *min path* (*hold time*) calculations.

Timing *equations 3 & 4* are for the flow from the *R10000* to the SSRAMs and *equations 5 & 6* are for the reverse flow, in the direction from the SSRAMs to the processor; in general, the subscript x & y specifies the direction of the flow, respectively. Note that values of certain parameters varies depending on the direction.

Eq 3: $T_{per} + t_{tap}^{*} \geq (t_{dox} + t_{pdx} * Lx + t_{LFX} + t_{ckjs} + t_{dnsx} + t_{dsux})Max$

Eq 4: $t_{Hx} + t_{tap} \leq (t_{dox} + t_{pdx} * Lx + t_{LFX} - t_{ckjs})Min$

Eq 5: $T_{per} - t_{tap} \geq (t_{doy} + t_{pdy} * Ly + t_{LFY} + t_{ckjs} + t_{dnsy} + t_{dsuy})Max$

Eq 6: $t_{Hy} - t_{tap} \leq (t_{doy} + t_{pdy} * Ly + t_{LFY} - t_{ckjs})Min$

† SCClk* is not specifically mentioned with the assumption that edges of SCClk* and SCClk are perfectly aligned (complementary) and that any uncertainty will be lumped in the over all uncertainty number.

‡ The PLL aligns ESCClk and ISCClk. The difference between the two sets of register and output buffer are incorporated in the specified skew.

* t_{tap} is usually a negative number which constitutes a shortening of the time available for the path in equation. 3 and lengthening the time available for the path in equation 5.

Eq 7: $t_{tap} = t_{tap0} + AdjN$

t_{do}	=Clock to output pin delay of an unloaded buffer
t_{pd}	= Intrinsic interconnect propagation delay per unit length.
L	= Net length between the driver pin and the receiver pin.
t_{ckjs}	= Maximum clock uncertainty for the path (due to jitter, skew, etc.)
t_{LF}	= Maximum timing delay due to loading for the given net topology.
t_{dns}	= Settling time for any dynamic noise that falls outside the dynamic noise margin and must be allowed to settle to acceptable limits (worst case when all buffer switching simultaneously).
t_{dsu}	= Receiver pin setup time.
t_H	= Receiver pin hold time.
t_{tap}	= Time delta between the *RSCClk* (at SSRAM) and *ISCClk* (internal).
t_{tap0}†	= Intrinsic delay between *ISCClk* or *SCClk* and *RSCClk* when *SCClkTap* is set to 0 (*Adj0* in *Table 2-3*).
AdjN	= Delay adjustment set via *SCClkTap* mode bits (*SysAD[28:25]*). See *table 2-3*

Figure 2-3 shows the range which provides a workable *RSCClk* delay with respect to *ISCClk* (t_{tap}). Following *equations 8 to 12* can be derived for t_{tap} by lumping the delay components and rearranging *equation 3 to 6*, respectively.

† The calculation of t_{tap0} is covered in the next section of this document.

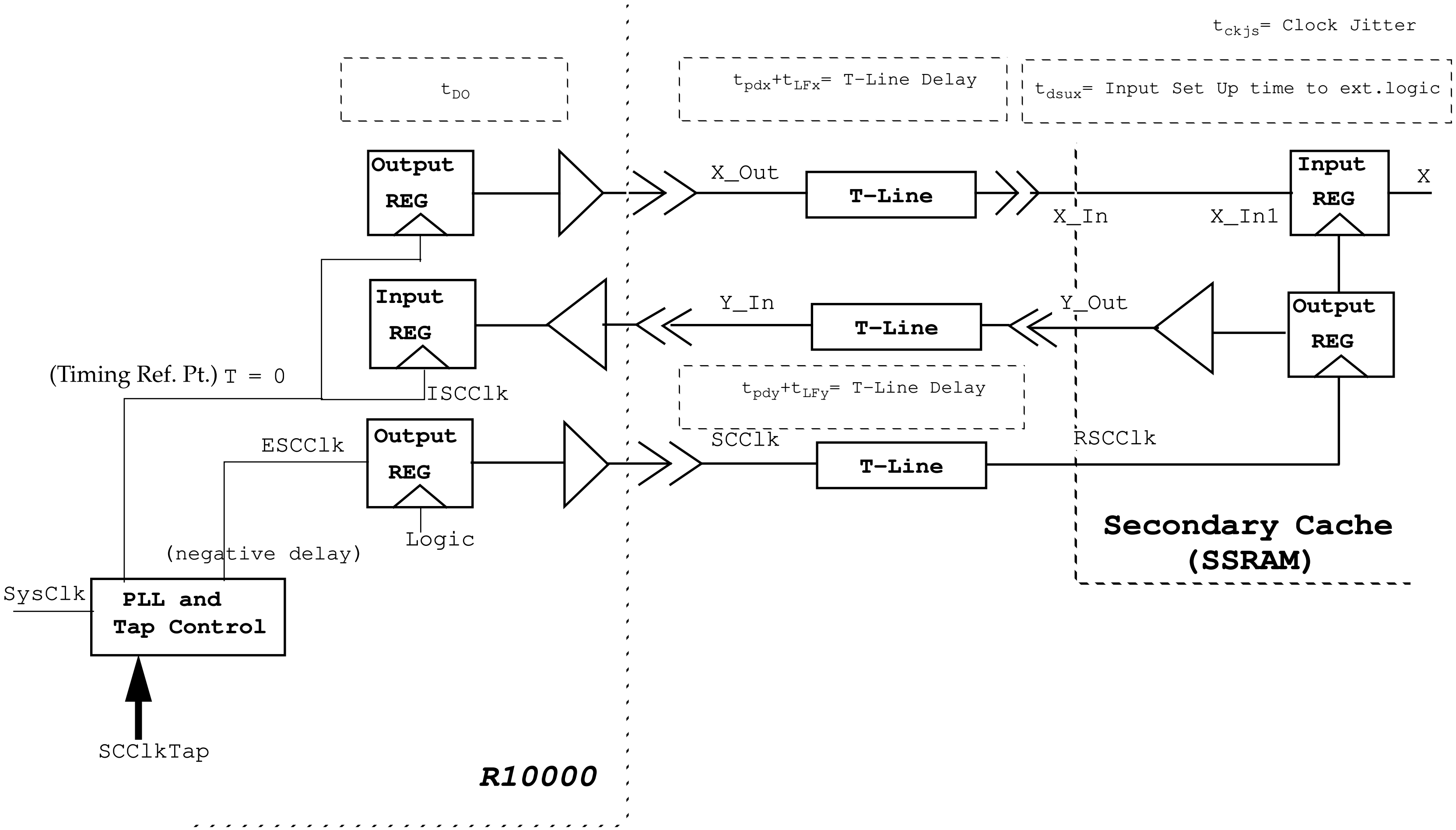

Figure 2-2 Example of paths between R10000 & Secondary Cache SSRAM

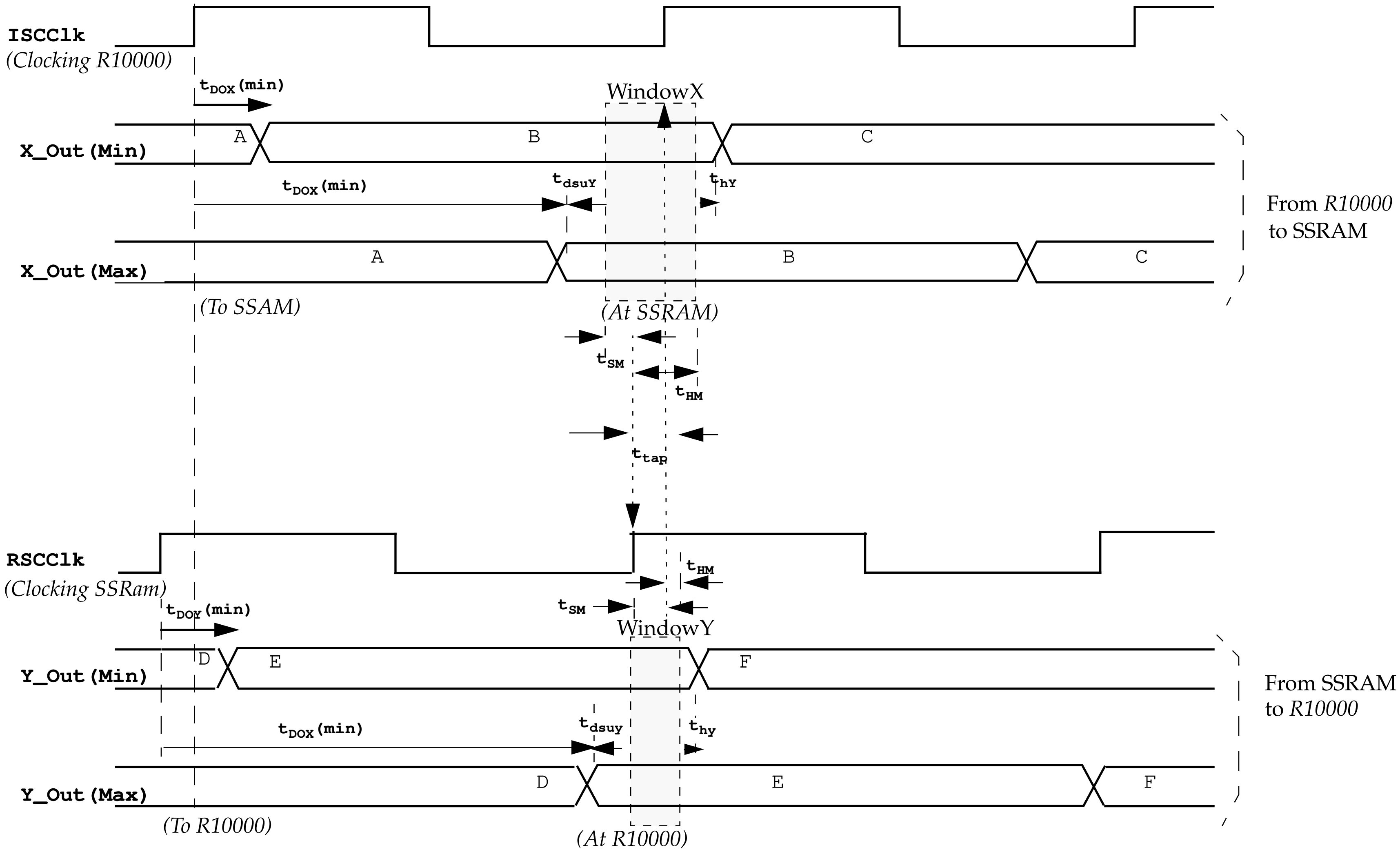

Figure 2-3 Selection of the SCClk delay with respect to ISCClk

`Eq 8:` $t_{tap} \geq -T_{per} + t_{DOX}(max) + t_{dsux}$ (This is limited by the left edge of WindowX)

`Eq 9:` $t_{tap} \leq t_{DOX}(min) - t_{hx}$ (This is limited by the right edge of WindowX)

`Eq 10:` $t_{tap} \leq T_{per} - t_{DOY}(max) - t_{dsuy}$ (This is limited by the left edge of WindowY)

`Eq 11:` $t_{tap} \geq t_{hy} - t_{DOY}(min)$ (This is limited by the right edge of WindowY)

Figure 2-3 shows how the clock period, delays, setup and hold times defines the *workable range*, indicated by the *Windows*. With each clock, *ISCClk* or *RSCClk*†, corresponding maximum and minimum output timing waveforms are shown. The different data sets are identified with letters. In case of the flow from the processor to the SSRAM (*X* direction), the data set 'B' is the one of interest for that particular cycle. The *RSCClk* rising edge triggers the data coming from the processor into the SSRAM; thus, this edge must fall within *WindowX*. Note that if the window is not large enough, then the frequency must be reduced. Similarly, the data flow 'E' is of interest in the '*Y*' direction. The *ISCClk* rising edge triggers the data coming from the SSRAM into the *R10000*; thus, this edge must fall within *WindowY*. In summary, for a set of specified *delay*, *setup* and *hold time* parameters, the width of each *workable range* (windows) can be defined by the respective *clock period* and the position of the windows relative to the triggering clock can be varied by the *SCClkTap* setting.

Using the above equations, the t_{tap} window, can be determined. Locating *AdjN* delay (*table 2-3*), for the *PClk* frequency of operation, that falls into this window, one can determine the *SCClkTap* setting. Note that *AdjN* in this table is nothing more than (N**PClk* period/12), given here just for convenience.

A couple of notes of cautions: The *clock to Q delay* of *R10000* buffers are significantly faster than that of the buffers in the SSRAM. Thus, the adjustment of *RSCCLK* is usually done to shorten the time available from *R10000* to SSAM and lengthen the time available from SSRAM to *R10000*. Care must be taken when *cycle stealing* in this way to avoid violating the *hold time* limit in the *min path* case in the path that is lengthened. Secondly, the Data lines from the processor are point to point; however, the load is spread out on the Address lines. Delay at one point of the address line could be quite different from another point. Care must be taken in *max path* (*setup time*) calculations and the design of the address lines.

† Only *ISCClk* & *RSCClk* are shown with the assumption that *ISCClk** & *RSCClk** are complements of each minus any uncertainty; and the uncertainty is lumped into the overall uncertainty number discussed somewhere else in this document.

Table 2-3 The SCClk adjustments (AdjN) with respect to the ISCClk:

***PClk*‡ Freq (MHz)**	**SysAD[28:25] Mode Value Settings (*SCClkTap*)**											
	0	**1**	**2**	**3**	**4**	**5**	**8**	**9**	**A**	**B**	**C**	**D**
	Adj0 (nS)	**Adj1 (nS)**	**Adj2 (nS)**	**Adj3 (nS)**	**Adj4 (nS)**	**Adj5 (nS)**	**Adj6 (nS)**	**Adj7 (nS)**	**Adj8 (nS)**	**Adj9 (nS)**	**Adj10 (nS)**	**Adj11 (nS)**
200	0.0	-0.42	-0.83	-1.25	-1.67	-2.08	-2.92	-3.33	-3.33	-3.75	-4.58	-5.00
190	0.0	-0.44	-0.88	-1.32	-1.75	-2.19	-2.63	-3.07	-3.51	-3.95	-4.82	-5.26
180	0.0	-0.46	-0.93	-1.39	-1.85	-2.31	-2.78	-3.24	-3.70	-4.17	-5.09	-5.56
170	0.0	-0.49	-0.98	-1.47	-1.96	-2.45	-2.94	-3.43	-3.92	-4.41	-5.39	-5.88
160	0.0	-0.52	-1.04	-1.56	-2.08	-2.6	-3.13	-3.65	-4.17	-4.69	-5.73	-6.25
150	0.0	-0.56	-1.11	-1.67	-2.22	-2.78	-3.33	-3.89	-4.44	-5.00	-6.11	-6.67
140	0.0	-0.60	-1.19	-1.79	-2.38	-2.98	-3.57	-4.17	-4.76	-5.36	-6.55	-7.14
130	0.0	-0.64	-1.28	-1.92	-2.56	-3.21	-3.85	-4.49	-5.13	-5.77	-7.05	-7.69
120	0.0	-0.69	-1.39	-2.08	-2.78	-3.47	-4.17	-4.86	-5.56	-6.25	-7.63	-8.33
110	0.0	-0.76	-1.52	-2.27	-3.03	-3.79	-4.55	-5.30	-6.06	-6.82	-8.33	-9.09
100	0.0	-0.83	-1.67	-2.50	-3.33	-4.17	-5.00	-5.83	-6.67	-7.50	9.17	-10.0

‡ ***PClk* is the clock used to run the internal core of *R10000*. Note that the tap adjustments are independent of the secondary cache clocks.**

Intrinsic Delay of the ***RSCClk*** with respect to ***ISCClk*** (t_{tap0})

The RSCClk is the clock at the SSRAM end. Even if the SCClk at the *R10000* end is aligned to the *ISCCLk*, the *RSCClk* will have some delay due to external components. This is represented by time t_{tap0}. The general equation for t_{tap0} is shown in equation 12.

Eq 12: $$t_{tap0} = t_{do} + t_{pd} * L + t_{LF}$$

t_{do}	= Clock to output pin delay of an unloaded buffer under nominal condition.
t_{pd}	= Intrinsic interconnect propagation delay per unit length.
L	= Net length between the driver pin and the receiver pin.
t_{LF}	= Timing delay due to loading for the given net topology.

In reality there are six *SCClk* pins driven by replicated drivers. This is done to provide enough drive capabilities without requiring additional external buffers. It is very likely that the paths and loading to the SSRAMs on the individual clock line might vary. Also, all the components in the above equation will have a maximum and a minimum value; due to variation in the processing parameters for chip, package, board, etc. Note that this variation buffer delay is already considered in *Table 2-2* as one of the uncertainty source; thus an average value could be used in the equations for t_{tap0}.

Summary

3

The *R10000* external interfaces are designed so the processor can be used in systems with various price performance points. It has two separate interfaces, one to the system side and the secondary cache side. For most of the systems, it will be critical to use the SPICE models for the output buffers, package, socket and devices provided by MIPS and the semiconductor suppliers, model the external components and the transmission lines on the board and simulate the timings. This is the only way to get the transmission line affects, network topology loading and noise affect accurately. A note of caution again that extreme care should be taken with the *min path* (*hold time*) calculations; especially, if no *SPICE* simulations are performed.

The components required for the calculations were considered. One of the key factors, is the *clock uncertainty*. Various factors that contribute to the *clock uncertainty* are listed. The equations for the I/O timings at both the interfaces are given. The I/O timing at secondary cache interface is complicated by the flexibility provided to adjust the external clock. The reason and trade off for adjusting the *SCClk* and the equations to determine the *SCClkTap* mode bits were discussed.

References

4

Reference 1: "MIPS *R10000* User's Manual" by Joe Heinrich (see "http://www.mips.com" for the latest copy).

Reference 2: E.E. Davidson, Electrical Design of a High Speed Computer Packaging," IBM Journal Research & Development 26, 349-361 (1982).

Reference 3: G.A. Katopis "Electrical Interconnection Design for High Performance Machine," proceedings NEPCON West 90, vol 1, pp.169-177, February 1990.

Reference 4: G.A. Katopis, "Delta-I Noise Specification for a High-Performance Computing Machine," IEEE, Vol. 73m no. 9, pp. 1409, September, 1985.