MOTOROLA
Semiconductor Products Sector

Order Number: AN2164
Rev. 0, 7/2001

PowerPC™

*APPLICATION NOTE*

# MPC8245 Memory Clock Design Guidelines

*Esther C Epiphane*
*RISC Application*
*risc10@email.sps.mot.com*

This document explains the mathematical analysis of DLL locking range loop delay charts for the MPC8245 hardware specification document, as well as setup time requirements for $T_{os}$ (SDRAM_SYNC_IN to PCI_SYNC_IN). Understanding and implementing the requirements for DLL locking and $T_{os}$ are important the MPC8245's proper functioning. This document covers the following topics:



This document contains information on a new product under development by Motorola.
Motorola reserves the right to change or discontinue this product without notice.

© Motorola, Inc., 2001. All rights reserved.

Digital DNA From Motorola

# 1.0 Clocking

The MPC8245 allows for multiple clock options to accommodate various system configurations. Internally, the MPC8245 uses one phase-locked loop (PLL) circuit to generate master clocks to the system logic and a second PLL to generate the processor clock. The system logic PLL is synchronized to the PCI_SYNC_IN input signal.

Figure 1 shows a block diagram of the clocking signals in the MPC8245.

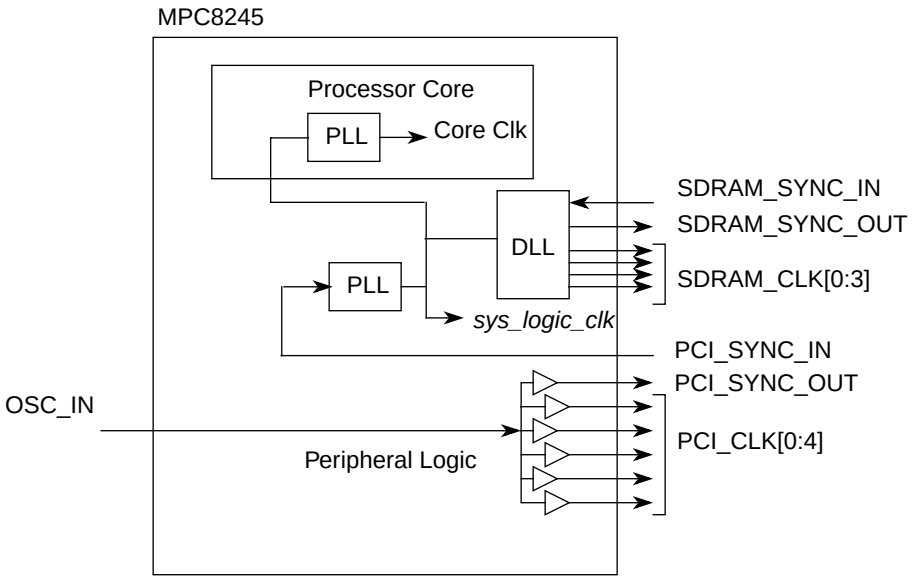



**Figure 1. Clock Subsystem Block Diagram**

The *sys_logic_clk* signal may be set to a multiple of the PCI bus frequency as defined in the *MPC8245Hardware Specification*. To help reduce the amount of discrete logic required in a system, the MPC8245 provides PCI clock fanout buffers. The MPC8245 also provides the memory clock (SDRAM_CLK*n*) signals through a delay locked loop (DLL) that is running at the same frequency as the internal system logic (*sys_logic_clk*).

The PCI and SDRAM clock signals are supplied for synchronization of external components on the system board. For minimum skew between SDRAM_SYNC_IN and the SDRAM clocks, the loop length on SDRAM_SYNC_IN should be designed to be the same as the loop lengths on the SDRAM_CLK*n* signals to their driven components. For example, for minimum skew, if an SDRAM device has a 5-inch trace, the loop trace should be 5 inches in length. Note that a system designer may deliberately vary the loop lengths in order to introduce a distinct amount of skew between SDRAM_SYNC_OUT and the SDRAM_CLK*n* signals.

# 1.1 DLL Locking Overview

The MPC8245 provides an on-chip delay-locked loop (DLL) that supplies the external memory bus clock signals to SDRAM banks. The DLL is made up of a delay line and a phase comparator and is responsible for generating the SDRAM_CLK(0:3), SDRAM Clock Synchronize Out (SDRAM_SYNC_OUT), and CPU_CLK(0:2) signals. SDRAM_SYNC_OUT should be fed back through a delay loop into the SDRAM_SYNC_IN input, which becomes the reference clock for the DLL. The clock, which goes through the DLL, is shifted up or down one of the 128 tap points in the DLL delay line by adjusting the length of the external delay loop. The phase comparator in the DLL compares the input and reference clock of the DLL every five clock cycles to determine whether a jump to an adjacent tap point in the delay line is necessary. This is done until the internal input clock to the DLL matches the reference clock, SDRAM_SYNC_IN. The DLL is locked when the input clock and reference clock to the DLL are matched. It then becomes possible to remove the effects of trace delay to the system memory by equating the delay through the external loop to the delay to the system memory. DLL locking range is required for proper operation of the MPC8245 and certain requirements must be met to ensure the DLL locks successfully. These requirements include using the design recommendations for SDRAM_SYNC_OUT timing and the propagation delay time for the DLL to lock. An increased lock range may cause slightly more jitter in the output clock of the DLL; hence, the limitations per different DLL_EXTEND modes and tap settings must be considered. When the DLL is locked, SDRAM_SYNC_IN will be in phase with *sys_logic_cl*k.

There are several possible delay line lengths. The shortest delay line length has a small amount of possible delay and a smaller clock jitter due to the tap point movement. Note that although an increased lock range makes it easier to guarantee that the reference clock is within the range, an increased lock range may cause slightly more jitter in the output clock of the DLL.

There are cases in which the DLL tap point may need to be explicitly altered, such as systems that do not use the DLL_MAX_DELAY bit to lengthen the DLL lock range and that are unable to meet the timing requirements, particularly with a low-speed memory bus. In this case, the DLL_EXTEND bit of PMCR2 can be written to shift the lock range of the DLL by half of an SDRAM clock cycle. Note that this bit should only be written during system initialization and should not be altered during normal operation. See *MPC8245 Hardware Specification* and user's manual for more information about the use of DLL_EXTEND and the locking ranges supplied by the MPC8245.

There is a bit (DLL_RESET) in the AMBOR register that controls the initial tap point of the DLL. Note that although this bit is cleared after a hard reset, it must be explicitly set and then cleared by software during initialization in order to guarantee correct operation of the DLL and the SDRAM_CLK[0:3] signals (if they are used).

This document describes the mathematical analysis of the four graphs in Section 1.4.3.1 of the MPC8245 hardware specification document which are specific to DLL locking. Each graph defines a scenario of DLL locking based on the settings of DLL_EXTEND (bit 7 of the Power Management Configuration Register 2 - 0x72) and the DLL_MAX_DELAY (bit 2 of the Miscellaneous I/O Control Register 1 - 0x76).

## 1.1.1 DLL Locking Graphs for the MPC8245

The general formula for calculating the DLL lock range depends on the settings of DLL_EXTEND. This is shown in Section 1.4.3.1, "Clock AC Timing Specifications," of the hardware specification document for the MPC8245. For DLL Lock Range with DLL_EXTEND disabled (Default=0), the lock range must be $0 \le (N*T_{clk} - t_{loop} - t_{fix0}) \le 7$ nanoseconds. The formula for calculating lock range with DLL_EXTEND enabled requires that the lock range must be $0 \le (N*T_{clk} - T_{clk}/2 - t_{loop} - t_{fix0}) \le 7$ nanoseconds. Where: N is a non-zero integer (1 or 2). $T_{clk}$ is the period of one SDRAM_SYNC_OUT clock cycle in nanoseconds. $t_{loop}$ is the propagation delay of the DLL synchronization feedback loop (PC board runner) from SDRAM_SYNC_OUT to SDRAM_SYNC_IN in nanoseconds; 6.25 inches of loop length (unloaded PC board runner) corresponds to approximately 1 nanoseconds of delay. $t_{fix0}$ is a fixed delay inherent in the

design when the DLL is at tap point 0 and the DLL is contributing no delay; $t_{fix0}$ equals approximately 3 nanoseconds. These equations may be used with the DLL locking graphs in the case studies that follow to verify that the delay of $T_{loop}$ is supported for specific cases. The grey areas of each graph are the regions for which DLL Locking occurs. The graphs are also present in Section 1.4.3.1, “Clock AC Timing Specifications,” in the MPC8245 hardware specification document.

The DLL_MAX_DELAY bit can lengthen the amount of time through the delay line. This is accomplished by increasing the time between each of the 128 tap points in the delay line. Although this increased time makes it easier to guarantee that the reference clock will be within the DLL lock range, it also means there may be slightly more jitter in the output clock of the DLL, should the phase comparator shift the clock between adjacent tap points. The impact of this time increase between tap points will be illustrated in the case studies to follow.

## 1.1.2 Variables and Equations

The following table defines the variables that will be used throughout the remainder of this document. These values are based on characterization data.

**Table 1. Terms and Definitions[1]**

| Term | Definition | Minimum | Maximum | Units | Notes |
|---|---|---|---|---|---|
| m | Integer | 10 | 118 | none | |
| N | An arbitrary integer | 0 | No limit | none | |
| $T_{clk}$ | Period of one SDRAM_SYNC_IN clock cycle | 7.5 | 30 | nanoseconds | 2 |
| $T_{ctq}$ | *sys_logic_clk* to SDRAM_SYNC_OUT timing at tap point 0 | 2.41 | 4.81 | nanoseconds | |
| Tap delay | Delay for going from one tap point to the next consecutive one (128 tap points) | 84 for normal<br>123 for max | 177 for normal<br>247 for max | picoseconds | 3 |
| $T_{dl}$ | Delay added by delay line in DLL | m*(Maximum Tap Delay)<br>= 10(MaximumTap Delay) | m*(MinimumTap delay)<br>=118(Minimum Tap Delay) | nanoseconds | 4 |
| $t_{loop}$ | Propagation delay of the DLL synchronization feedback loop from SDRAM_SYNC_OUT to SDRAM_SYNC_IN | Mode dependant | Mode Dependant | nanoseconds | 5 |
| $T_{os}$ | Offset period required for phase alignment of SDRAM_SYNC_IN to *sys_logic_clk* | 0.65 | 1.00 | nanoseconds | |

Notes:
1. The data in this table is based on design simulation.
2. The period of the SDRAM clocks is dependent on the frequency of the memory bus. The frequency range for the memory bus is 33-133 MHz ehrn operating at 266MHz CPU frequency and 33-100 MHz when operating at 300MHz CPU frequency. The period values represented in the table are for a 300 MHz CPU part.
3. Normal tap delay, bit 2 of offset 0x76 is cleared. Maximum Tap Delay, bit 2 of offset 0x76 is set.

4. See explanation in Section 1.1.2., “Variables and Equations.”
5. Non-extended mode; bit 7 of offset 0x72 is cleared. Extended mode; bit 7 of offset 0x72 is set.

The general formulas for creating the DLL Locking graphs are:

(1) $n*T_{clk} = T_{ctq} + T_{os} + T_{dl} + T_{loop}$ (Non-Extended mode)

(2) $(n-0.5)*T_{clk} = T_{ctq} + T_{os} + T_{dl} + T_{loop}$ (Extended mode)

These equations are different from the equations in Section 1.1.1, “DLL Locking Graphs for the MPC8245,” which show the general formulas for obtaining the trace length to guarantee DLL locking. Once the delay timing desired is obtained, the graphs may be used to figure out whether the delay length $T_{loop}$ is supported in the grey areas of the graph. Here follows a more detailed analysis of how the DLL Locking range graphs for the MPC8245 were obtained.

Let $T_{dp} = T_{ctq} + T_{os} + T_{dl}$

From (1), $T_{loop} = N*T_{clk} - T_{dp}$ (Non-Extended mode)

From (2), $T_{loop} = (N-0.5)*T_{clk} - T_{dp}$ (Extended mode)

Please note that the use of “(min)” throughout the remainder of the document refers to minimum value of the variable that it is used in context with. Likewise for “(max)” throughout the remainder of the document refers to maximum value of the variable with which it is used in context.

The value $T_{loop}$ can be determined as follows:

(3) $(N*T_{clk} - T_{dp}(max)) \le T_{loop} \le (N*T_{clk} - T_{dp}(min))$; (Non-Extended mode)

(4) $((N-0.5)*T_{clk} - T_{dp}(max)) \le T_{loop} \le ((N-0.5)*T_{clk} - T_{dp}(min))$; (Extended mode)

This is because the value of $T_{loop}$ should take into consideration the worse case values for $T_{dp}$. Therefore, $T_{loop}$ has to cover the range of its smallest to largest possible values. The smallest possible value is “$N*T_{clk} - T_{dp}(max)$” for Non-Extended and “$(N-0.5)*T_{clk} - T_{dp}(max)$” for Extended mode. The largest possible value of $T_{loop}$ is “$N*T_{clk} - T_{dp}(min)$” for Non-Extended and “$(N-0.5)*T_{clk} - T_{dp}(min)$” for Extended mode.

Analysis of the values of $T_{dp}$ shows that:

$T_{dp}(min) = T_{ctq}(max) + T_{os}(max) + T_{dl}(min)$ Eq. (5)

$T_{dp}(max) = T_{ctq}(min) + T_{os}(min) + T_{dl}(max)$ Eq.(6)

Note that the maximum values of $T_{ctq}$ and $T_{os}$ are being used for the minimum value of $T_{dp}$. The reverse is the case for the maximum value for $T_{dp}$ where the minimum values of $T_{ctq}$ and $T_{os}$ are being used. This is because the worse case values are being used for the minimum and maximum values.

For example, let the characterization data times found at tap point 0 to be from 0 to 46.5 picoseconds and the times at tap point 1 to be 130.5 to 177.0 picoseconds. The largest variation of the time from tap point 0 to tap point 1 is from 0 to 177.0 picoseconds, or a difference of 177 picoseconds. By observation the smallest variation is from 46.5 to 130.5 picoseconds, which is 84 picoseconds. Figure 2 shows the minimum and maximum worse case analysis.

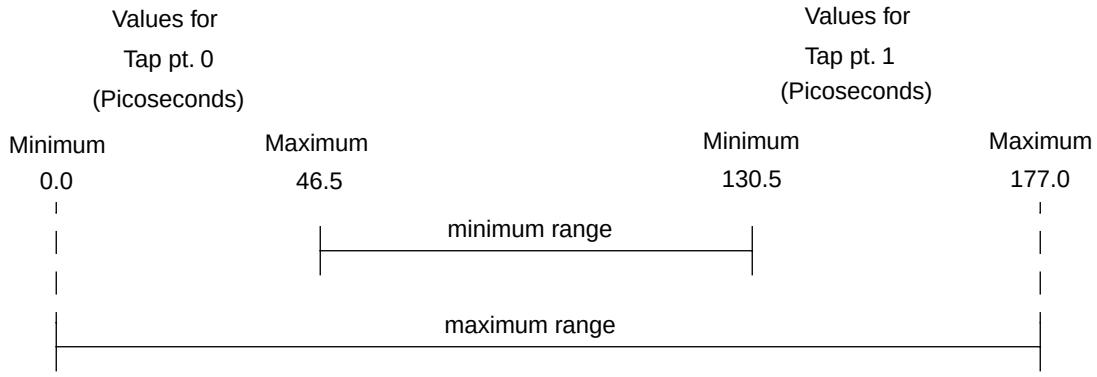



**Figure 2. Explanation of Maximum and Minimum Value Usage in Equations 5 and 6.**

**Table 2. Tap Delay Data**

| Term | Mode | Minimum | Maximum | Unit |
|---|---|---|---|---|
| Tap Delay | Normal Tap Delay | 84 | 177 | picoseconds |
| | Maximum Tap Delay | 123 | 247 | picoseconds |

Using the values in Tables 1 and 2 as well as the equations of Section 1.1.2, we can use calculations to show how the graphs were obtained for DLL Locking in the four various modes.

The graphs in Section 1.4.3.1, “Clocking AC Specifications,” of the hardware specification document were derived from the graphs in this document. (The grey areas in each graph represent where the DLL will lock.) The areas of DLL locking for various modes are as follows:

1. For Standard DLL mode (non extended): Clear bit 7 (DLL_EXTEND) at offset 0x72. (Default)

2. For Extended DLL mode: Set bit 7 (DLL_EXTEND) at offset 0x72.

3. Normal Tap Delay (Default): Clear bit 2 (DLL_MAX_DELAY) at offset 0x76.

4. Max Tap Delay: Set bit 2 (DLL_MAX_DELAY) at offset 0x76.

## 1.1.3 Case Analyses

### 1.1.3.1 Case 1: Normal Tap Delay and Non-Extended Mode:

Recall the following:

Non-Extended mode $(N*T_{clk} - T_{dp}(max)) \leq T_{loop} \leq (N*T_{clk} - T_{dp}(min))$;

$T_{dp}(min) = T_{ctq}(max) + T_{os}(max) + T_{dl}(min)$ Eq. (5)

$T_{dp}(max) = T_{ctq}(min) + T_{os}(min) + T_{dl}(max)$ Eq.(6)

Therefore using Tables 1 and 2, the following values were obtained:

$T_{ctq}(max) = 4.81$ ns, $T_{os}(max) = 1.215$ ns, $T_{dl}(min) = 10$ (0.177) ns

$T_{dp}(min) = 4.81 + 1.215 + 1.77 = 7.795$ ns

Also, $T_{ctq}$(min) = 2.407 ns, $T_{os}$(min) = 0.647 ns, $T_{dl}$(max) = 118(0.084) ns

This gives $T_{dp}$(max) = 2.407 + 0.647 + 9.912 = 12.966ns.

Recall that (N*$T_{clk}$ - $T_{dp}$(max)) ≤ $T_{loop}$ ≤ (N*$T_{clk}$ - $T_{dp}$(min))

Note that for Graph 1 in Figure 3, if the grey strip at the maximum $T_{clk}$ area is extrapolated to the x-axis we can find the range of $T_{loop}$ when N = 0, and $T_{clk}$ = 0. Based on the calculations above - 12.966 ≤ $T_{loop}$ ≤ -7.795 nanoseconds. Comparing our calculations to the extrapolated values on the graph we have estimated around the range of -13 ≤ $T_{loop}$ ≤ - 8 nanoseconds. Note that as the values of N and $T_{clk}$ are increased the range of $T_{loop}$ shifts right on the x-axis eventually creating the grey DLL Locking areas. Each band represents the DLL Locking at a particular value of N.

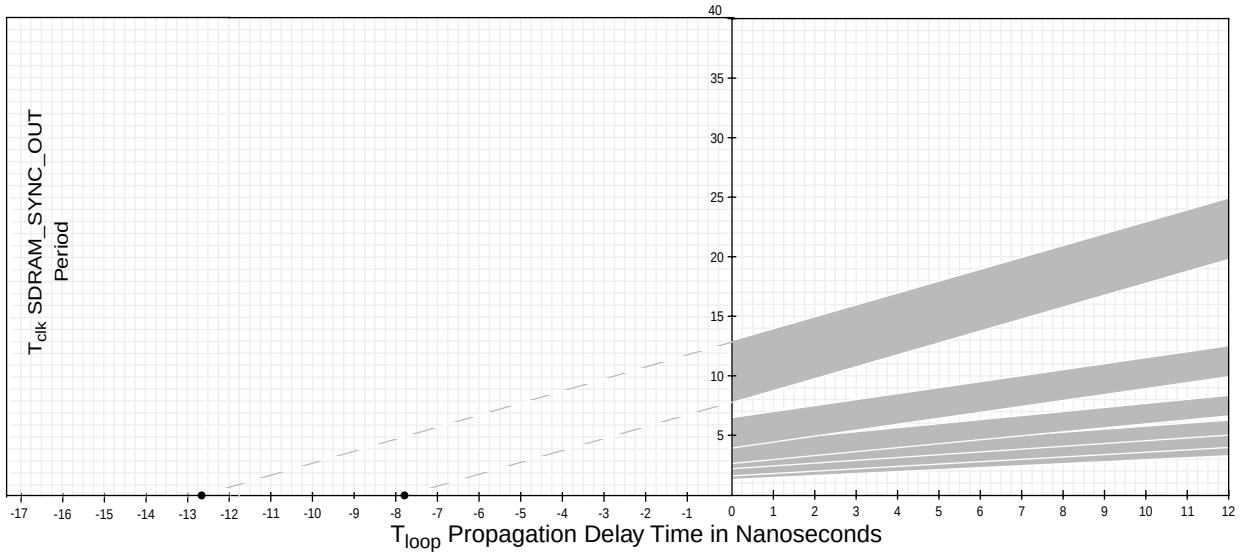



**Figure 3. DLL Locking Range Loop Delay vs. Frequency of Operation for DLL_Extend=0 and Normal Tap Delay**

## 1.1.3.2 Case 2: Maximum Tap Delay and Non-Extended Mode:

Recall the following:

Non-Extended mode (N*$T_{clk}$ - $T_{dp}$(max)) ≤ $T_{loop}$ ≤ (N*$T_{clk}$ - $T_{dp}$(min));

$T_{dp}$(min) = $T_{ctq}$(max) + $T_{os}$(max) + $T_{dl}$(min) Eq. (5)

$T_{dp}$(max) = $T_{ctq}$(min) + $T_{os}$(min) + $T_{dl}$(max) Eq.(6)

Therefore using Tables 1 and 2, the following values were obtained:

$T_{ctq}$(max) = 4.81 ns, $T_{os}$(max) = 1.215 ns, $T_{dl}$(min) = 10 (0.247) ns

$T_{dp}$(min) = 4.81 + 1.215 + 2.47 = 8.495 ns

Also, $T_{ctq}$(min) = 2.407 ns, $T_{os}$(min) = 0.647 ns, $T_{dl}$(max) = 118(0.123) ns

This gives $T_{dp}$(max) = 2.407 + 0.647 +14.514 = 17.568 ns.

Recall that (N*$T_{clk}$ - $T_{dp}$(max)) ≤ $T_{loop}$ ≤ (N*$T_{clk}$ - $T_{dp}$(min))

Note that for Graph 2 in Figure 4, if the grey strip at the maximum $T_{clk}$ area is extrapolated to the x-axis we can find the range of $T_{loop}$ when N = 0, and $T_{clk}$ = 0. Based on the calculations above - $17.568 \leq T_{loop} \leq -8.495$ nanoseconds. Comparing our calculations to the extrapolated values on the graph we have estimated around the range of $-17.6 \leq T_{loop} \leq -8.5$ nanoseconds. Note that as the values of N and $T_{clk}$ are increased the range of $T_{loop}$ shifts right on the x-axis eventually creating the grey DLL Locking areas. Each band represents the DLL Locking at a particular value of N.

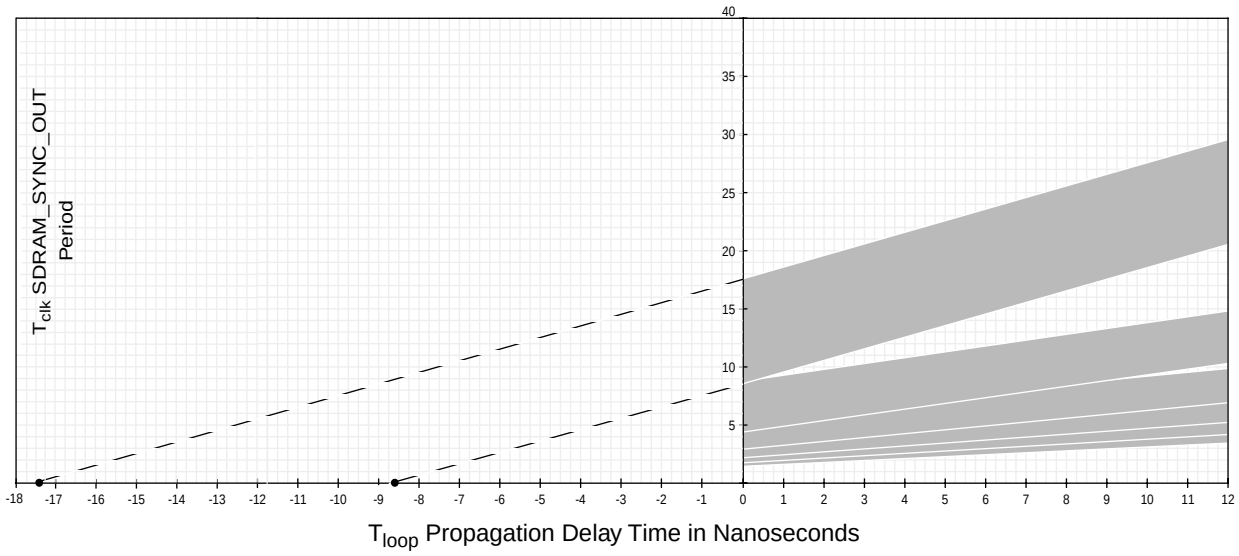



**Figure 4. DLL Locking Range Loop Delay vs. Frequency of Operation for DLL_Extend=0 and Max Tap Delay**

## 1.1.3.3 Case 3: Normal Tap Delay and Extended Mode:

Recall the following:

Extended mode $((N-0.5)*T_{clk} - T_{dp}(max)) \leq T_{loop} \leq ((N-0.5)*T_{clk} - T_{dp}(min))$;

$T_{dp}(min) = T_{ctq}(max) + T_{os}(max) + T_{dl}(min)$ Eq. (5)

$T_{dp}(max) = T_{ctq}(min) + T_{os}(min) + T_{dl}(max)$ Eq.(6)

Therefore using Tables 1 and 2, the following values were obtained:

$T_{ctq}(max) = 4.81$ ns, $T_{os}(max) = 1.215$ ns, $T_{dl}(min) = 10$ (0.177) ns

$T_{dp}(min) = 4.81 + 1.215 + 1.77 = 7.795$ ns

Also, $T_{ctq}(min) = 2.407$ ns, $T_{os}(min) = 0.647$ ns, $T_{dl}(max) = 118$(0.084) ns

This gives $T_{dp}(max) = 2.407 + 0.647 + 9.912 = 12.966$ns.

Recall that $((N-0.5)*T_{clk} - T_{dp}(max)) \leq T_{loop} \leq ((N-0.5)*T_{clk} - T_{dp}(min))$

Note that for Graph 3 in Figure 5, if the grey strip at the maximum $T_{clk}$ area is extrapolated to the x-axis, we can find the range of $T_{loop}$ when N = 0, and $T_{clk}$ = 0. Based on the calculations above: $12.966 \leq T_{loop} \leq -7.795$ nanoseconds. Comparing our calculations to the extrapolated values on the graph, we have estimated around the range of $-13 \leq T_{loop} \leq -8$ nanoseconds. Note that as the values of N and $T_{clk}$ are

increased the range of $T_{loop}$ shifts right on the x-axis eventually creating the grey DLL Locking areas. Each band represents the DLL Locking at a particular value of N.

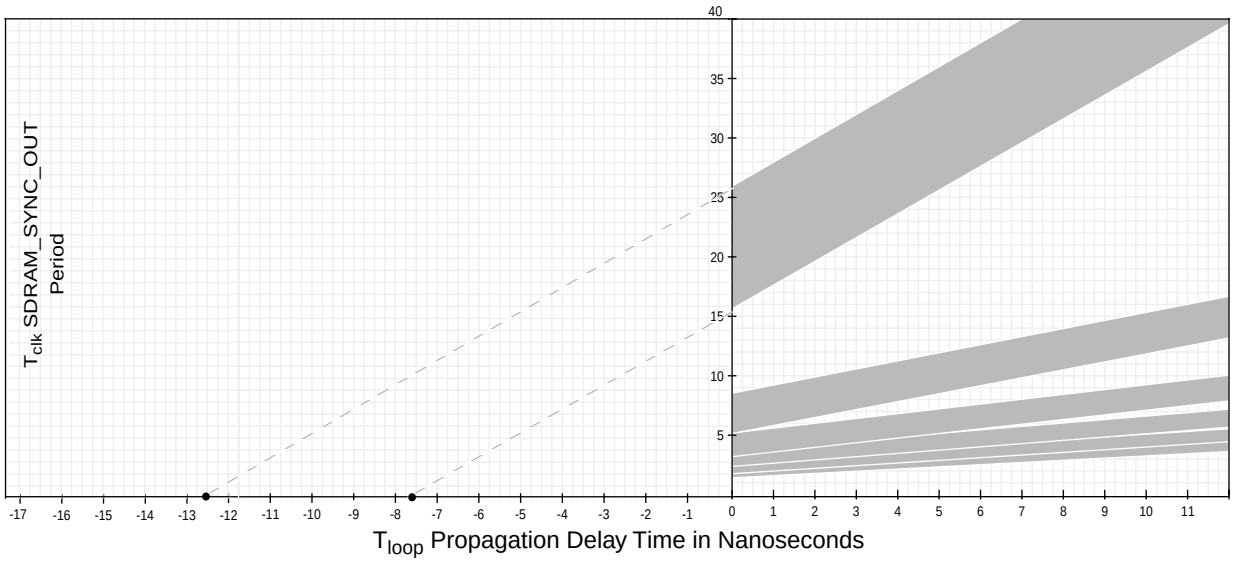



**Figure 5. DLL Locking Range Loop Delay vs. Frequency of Operation for DLL_Extend=1 and Normal Tap Delay**

## 1.1.3.4 Case 4: Maximum Tap Delay and Extended Mode:

Recall the following:

Extended mode $((N-0.5)*T_{clk} - T_{dp}(max)) \le T_{loop} \le ((N-0.5)*T_{clk} - T_{dp}(min))$;

$T_{dp}(min) = T_{ctq}(max) + T_{os}(max) + T_{dl}(min)$ Eq. (5)

$T_{dp}(max) = T_{ctq}(min) + T_{os}(min) + T_{dl}(max)$ Eq.(6)

Therefore using Tables 1 and 2, the following values were obtained:

$T_{ctq}(max) = 4.81$ ns, $T_{os}(max) = 1.215$ ns, $T_{dl}(min) = 10$ (0.247) ns

$T_{dp}(min) = 4.81 + 1.215 + 2.47 = 8.495$ ns

Also, $T_{ctq}(min) = 2.407$ ns, $T_{os}(min) = 0.647$ ns, $T_{dl}(max) = 118$(0.123) ns

This gives $T_{dp}(max) = 2.407 + 0.647 + 14.514 = 17.568$ ns.

Recall that $((N-0.5)*T_{clk} - T_{dp}(max)) \le T_{loop} \le ((N-0.5)*T_{clk} - T_{dp}(min))$

Note that for Graph 4 in Figure 6, if the grey strip at the maximum $T_{clk}$ area is extrapolated to the x-axis, we can find the range of $T_{loop}$ when N = 0, and $T_{clk} = 0$. Based on the calculations above: $17.568 \le T_{loop} \le -8.495$ nanoseconds. Comparing our calculations to the extrapolated values on the graph we have estimated around the range of $-17.6 \le T_{loop} \le -8.5$ nanoseconds. Note that as the values of N and $T_{clk}$ are increased, the range of $T_{loop}$ shifts right on the x-axis eventually creating the grey DLL locking areas. Each band represents the DLL Locking at a particular value of N.

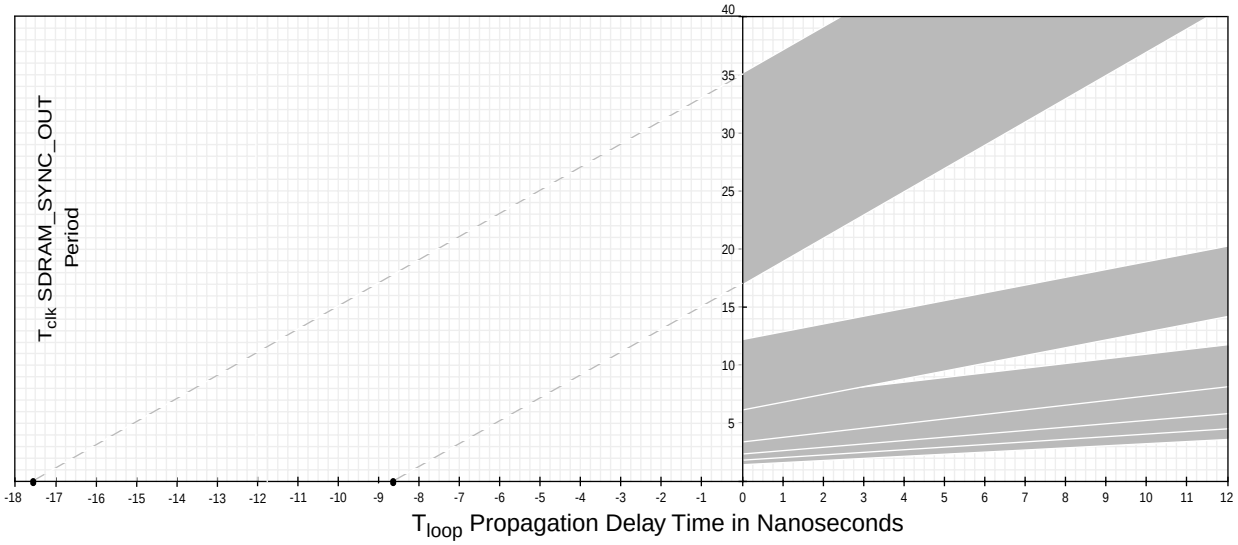


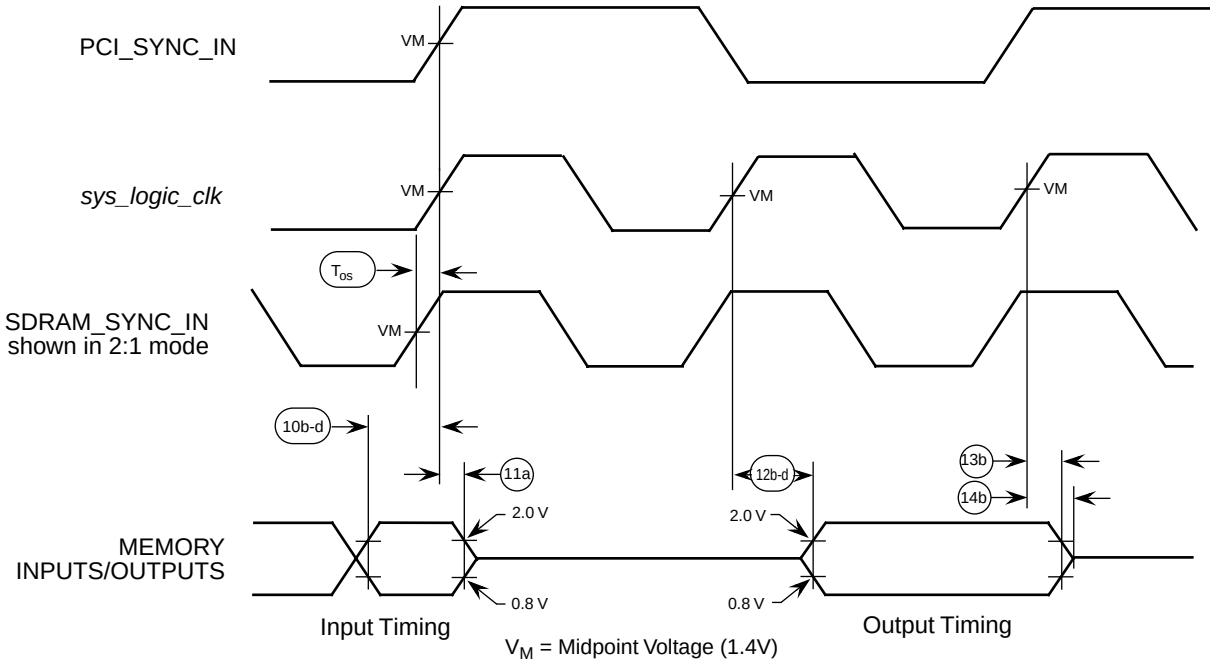

**Figure 6. DLL Locking Range Loop Delay vs. Frequency of Operation for DLL_Extend=1 and Max Tap Delay**

# 1.2 SDRAM_SYNC_IN to *sys_logic_clk* Time ($T_{os}$)

The MPC8245 has an internal delay in the feedback path for SDRAM_SYNC_IN with respect to the internal *sys_logic_clk* signal. This results in the SDRAM_CLKs not being phase-aligned with SDRAM_SYNC_IN. The internal *sys_logic_clk* signal is used by the peripheral logic to latch input data and launch output data on the memory interface. Due to the additional internal delay present in *sys_logic_clk* and the DLL, the resulting SDRAM_CLK pins will be offset by the delay amount.

$T_{os}$ represents the timing adjustment for SDRAM_SYNC_IN with respect to *sys_logic_clk*. The feedback trace length of SDRAM_SYNC_OUT to SDRAM_SYNC_IN must be shortened by this amount relative to the SDRAM clock output trace lengths so that phase alignment of the memory clocks can be maintained with respect to *sys_logic_clk*. Based on the range of values given in Table 1 for $T_{os}$, the trace length of SDRAM_SYNC_OUT to SDRAM_SYNC_IN can be shortened by 0.65 nanoseconds to about 1.0 nanoseconds in timing. This offset cannot be totally accommodated for by using the length adjustment of $T_{os}$. The problem of the SDRAM clocks accessing memory early needs to be assessed on a case-by-case basis considering the overall system architecture- including memory bus speed and the impact of certain trace lengths used.

Figure 7 shows $T_{os}$ as the phase delay between SDRAM_SYNC_IN and *sys_logic_clk*.




$V_M$ = Midpoint Voltage (1.4V)
10b-d = Input signals valid timing
11a = Input hold time of SDRAM_SYNC_IN to Memory.
12b-d = SDRAM_SYNC_IN to output valid timing
13b = Output hold time for non-PCI signals
14b = SDRAM-SYNC_IN to output high impedance timing
for non-pci signals
$T_{os}$ = Offset timing required to align sys_logic_clk with SDRAM_SYNC_IN
See hardware specifications document for details of the above timing characterization values.

**Figure 7. $T_{os}$ Relative to *sys_logic_clk*, SDRAM_SYNC_IN, and PCI_SYNC_IN Timing**

# 1.3 Conclusion

When designing for the MPC8245, the DLL locking range must be considered to design using the appropriate trace lengths for SDRAM_SYNC_OUT to SDRAM_SYNC_IN. The offset adjustment $T_{os}$ must also be considered as an additional length required to be removed from SDRAM_SYNC_OUT to SDRAM_SYNC_IN to create the final trace length.

## Table 3. Document History

| Revision Number | Changes |
|---|---|
| Rev 0 | Initial Document |
| | |

DigitalDNA is a trademark of Motorola, Inc.

The PowerPC name, the PowerPC logotype, and PowerPC 603e are trademarks of International Business Machines Corporation used by Motorola under license from International Business Machines Corporation.

Information in this document is provided solely to enable system and software implementers to use PowerPC microprocessors. There are no express or implied copyright licenses granted hereunder to design or fabricate PowerPC integrated circuits or integrated circuits based on the information in this document.

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters which may be provided in Motorola data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals” must be validated for each customer application by customer’s technical experts. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part. Motorola and Ⓜ are registered trademarks of Motorola, Inc. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

HOW TO REACH US:

USA/EUROPE/LOCATIONS NOT LISTED: Motorola Literature Distribution; P.O. Box 5405, Denver, Colorado 80217. 1-303-675-2140 or 1-800-441-2447

JAPAN: Motorola Japan Ltd.; SPS, Technical Information Center, 3-20-1, Minami-Azabu. Minato-ku, Tokyo 106-8573 Japan. 81-3-3440-3569

ASIA/PACIFIC: Motorola Semiconductors H.K. Ltd.; Silicon Harbour Centre, 2 Dai King Street, Tai Po Industrial Estate, Tai Po, N.T., Hong Kong. 852-26668334

TECHNICAL INFORMATION CENTER: 1-800-521-6274

HOME PAGE: http://www.motorola.com/semiconductors

DOCUMENT COMMENTS: FAX (512) 933-2625, Attn: RISC Applications Engineering

**WORLD WIDE WEB ADDRESSES**: http://www.motorola.com/PowerPC
http://www.motorola.com/NetComm
http://www.motorola.com/ColdFire