MPC8245/MPC8241 Memory Clock Design Guidelines: Part 1

by *Esther C. Alexander*
RISC Applications, CPD
Freescale Semiconductor, Inc.
Austin, TX

This application note is the first of a two-part series that explains the details of determining the SDRAM clock trace lengths and design concerns for the memory clock interface of the MPC8245/MPC8241. This part describes the details of DLL locking, DLL modes, and T_{os}, which are important to understand the functioning of the MPC8245/MPC8241. The second part, *MPC8245/MPC8241 Memory Clock Design Guidelines: Part 2* (AN2746) goes into timing analysis and delay issues.

The following Freescale documents are referenced throughout this application note:

- *MPC8245 Integrated Processor User's Manual* (MPC8245UM)
- *MPC8245 Integrated Processor Hardware Specifications* (MPC8245EC)
- *MCP8241 Integrated Processor Hardware Specifications* (MPC8241EC)
- *MPC8245 Part Number Specification for the MPC8245ARZUnnnX Series* (MPC8245ARZUPNS)
- *MPC8245 Part Number Specification for the XPC8245TXXnnnx Series* (MPC8245TXXPNS)
- *MPC8245 Part Number Specification for the XPC8245TZUnnnx Series* (MPC8245TZUPNS)
- *MPC8241 Part Number Specification for the XPC8241TXXnnnx Series* (MPC8241TXXPNS)
- *MPC8245/MPC8241 Integrated Processor Chip Errata* (MPC8245CE)

Contents

1 Clocking

The MPC8245/MPC8241 allows for multiple clock options to accommodate various system configurations. Internally, the MPC8245/MPC8241 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.

The DLL uses the *sys_logic_clk* signal as the reference clock to synchronize the memory clock signals with the core clock. The speed of memory clock signals and *sys_logic_clk* may be set to a multiple of the PCI bus frequency as defined in the PLL tables of the MPC8245 and MCP8241 hardware specifications. To help reduce the amount of discrete logic required in a system, the MPC8245/MPC8241 provides PCI clock fanout buffers.

Figure 1 shows a block diagram of the clocking signals in the MPC8245/MPC8241.

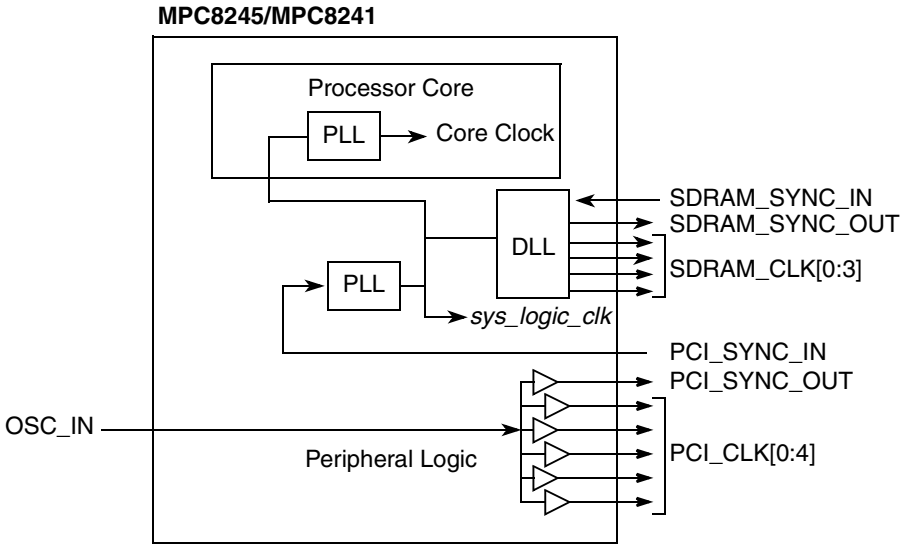

Figure 1. Clock Subsystem Block Diagram

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. This is discussed in Section 5, "T_{os} (SDRAM_SYNC_IN to sys_logic_clk)," when adjustments for the T_{os} related delay should be considered.

2 DLL Locking Overview

The MPC8245/MPC8241 provides an on-chip 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] and SDRAM clock synchronize out (SDRAM_SYNC_OUT to SDRAM_SYNC_IN) 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 memory devices. The clock, which goes through the DLL, is shifted up or down 1 of 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 (*sys_logic_clk*) 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. When the DLL is locked, SDRAM_SYNC_IN will be in phase with *sys_logic_clk*. 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 is required for proper operation of the MPC8245/MPC8241, and certain requirements must be met to ensure the DLL locks successfully. These requirements include using the design recommendations for SDRAM_SYNC_OUT to SDRAM_SYNC_IN timing and the propagation delay time for the DLL to lock. There are four possible DLL locking modes. They are determined by the values of the following bits:

- The DLL_EXTEND bit of power management configuration register 2 (PMCR2[7], at offset 0x72)
- The DLL_MAX_DELAY bit of the miscellaneous I/O control register 1 (MIOCR1[2], at offset 0x76)

PMCR2[DLL_EXTEND] can be used to shift the lock range of the DLL by one half of an SDRAM clock cycle. MIOCR1[DLL_MAX_DELAY] is used to increase the length between two consecutive tap points.

Note that although an increased time between tap points makes it easier to guarantee that the reference clock is within the DLL lock range, it may cause slightly more jitter in the output clock of the DLL, should the phase comparator shift the clock between adjacent tap points.

There are cases in which the DLL tap point may need to be explicitly altered, such as in systems that do not use MIOCR1[DLL_MAX_DELAY] 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, PMCR2[DLL_EXTEND] can be written to shift the lock range of the DLL by one half of an SDRAM clock cycle.

Note that these bits should be written only during system initialization and should not be altered during normal operation. Bit 5 (DLL_RESET) of the address map B options register (AMBOR, at offset 0xE0) controls the initial tap point of the DLL. The tap point that is the start point as a result of toggling AMBOR[DLL_RESET] is tap point 12.

NOTE

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 should be done after the DLL mode setting has been made. See the MPC8245 and MCP8241 hardware specifications and the MPC8245 user's manual for more information about the use of DLL_EXTEND and the locking ranges supplied by the MPC8245/MPC8241.

Figure 2 shows the optimal result of the DLL locking. See Section 5, "T_{os} (SDRAM_SYNC_IN to sys_logic_clk)," for details on an internal offset delay in the MPC8245/8241 and how it affects DLL locking.

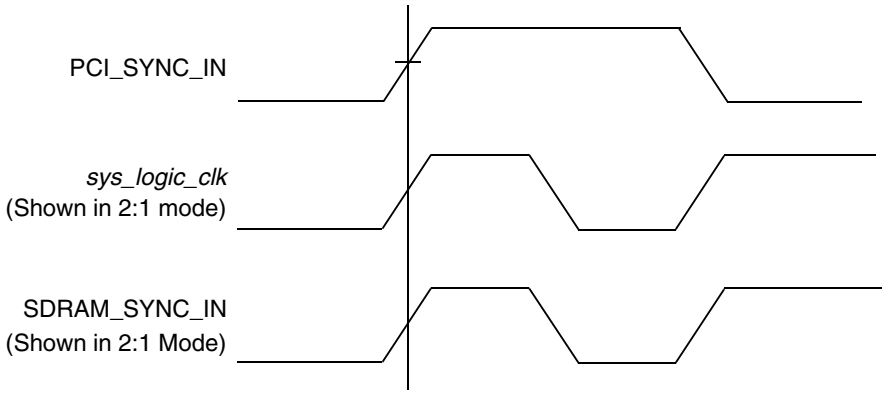

Figure 2. Optimal Result of DLL Lock (Without T_{os})

This application note describes the mathematical analysis of the four DLL locking graphs given in the MPC8245 and MCP8241 hardware specifications document. Each graph defines a scenario of DLL locking based on the settings of PMCR2[DLL_EXTEND] and MIOCR1[DLL_MAX_DELAY].

3 DLL Locking Graphs for the MPC8245/MPC8241

The DLL locking graphs consist of a vertical axis which represents T_{clk} and a horizontal axis which represents T_{loop}. T_{clk} is the period of one SDRAM clock cycle in nanoseconds (ns). T_{loop} is the propagation delay of the DLL synchronization feedback loop (PC board runner) from SDRAM_SYNC_OUT to SDRAM_SYNC_IN in ns; 6.25 inches of loop length (unloaded PC board runner) corresponds to approximately 1 ns of delay. The grey areas of each graph are the regions for which DLL locking occurs. The values are given in the case analyses as shown in Section 3.2, “Case Analyses.”

3.1 Variables and Equations

Table 1 defines the variables that will be used throughout the remainder of this application note. These values are based on characterization data. The data in this table is based on design simulation and characterization data.

Table 1. Terms and Definitions

Term	Definition	Minimum	Maximum	Unit	Notes
T_{clk}	Period of one SDRAM clock cycle	7.5	20	ns	1
T_{loop}	Propagation delay represents the actual trace length required for the DLL synchronization feedback loop from SDRAM_SYNC_OUT to SDRAM_SYNC_IN (6.25 inches: 1 ns)	Mode dependent	Mode dependent	ns	2

Notes:
1. The period of the SDRAM clocks is dependent on the frequency of the memory bus. The frequency range for the memory bus is 50–133 MHz when operating at 266-, 333-, and 400-MHz CPU frequencies and 50–100 MHz when operating at 300- and 350-MHz CPU frequencies due to the multiplier ratios in the PLL tables. See the MPC8245 and MCP8241 hardware specifications, and their respective part number specifications, for details on the multipliers available for the various processor speeds. The clock period values represented in this table are for a memory bus speed range of 50–133 MHz.
2. There are four modes for DLL locking depending on the bit combination of PMCR2[DLL_EXTEND] and MIOCR1[DLL_MAX_DELAY].

Note that the register settings, which define each DLL locking mode are shown in Table 2.

Table 2. DLL Mode Definition

DLL Mode	MIOCR1[DLL_MAX_DELAY]	PMCR2[DLL_EXTEND]
Normal tap delay, No DLL extend	0	0
Normal tap delay, DLL extend	0	1
Maximum tap delay, No DLL extend	1	0
Maximum tap delay, DLL extend	1	1

3.2 Case Analyses

This section shows how the equations for the graphs of the DLL locking regions were derived. These graphs are used in the MPC8245 and MPC8241 hardware specifications to evaluate trace lengths for the SDRAM clocks and SDRAM_SYNC_OUT to SDRAM_SYNC_IN for various mode combinations. For each case study, the initial value of T_{loop} for a grey region is calculated and compared with the T_{loop} obtained by using the DLL locking graphs.

The following applies to all four DLL graphs described in this section. The grey bands, labeled A and B in each graph, represent where the DLL will lock. The equations that mark the edge of each gray band are the limits outside which the DLL is not guaranteed to lock. For each graph, A_{max} represents the limit on the maximum side of band A and A_{min} is the limit on the minimum side. Likewise, B_{max} represents the limit on the maximum side of band B and B_{min} is the limit on the minimum side.

3.2.1 Case 1: Normal Tap Delay and Non-DLL-Extended Mode

Normal tap delay—MIOCR1[DLL_MAX_DELAY] is cleared.

No DLL extend—PMCR2[DLL_EXTEND] is cleared.

Figure 3 shows two grey bands (call them A and B) that have been identified as the regions for DLL locking when in normal tap delay and non-DLL-extended mode.

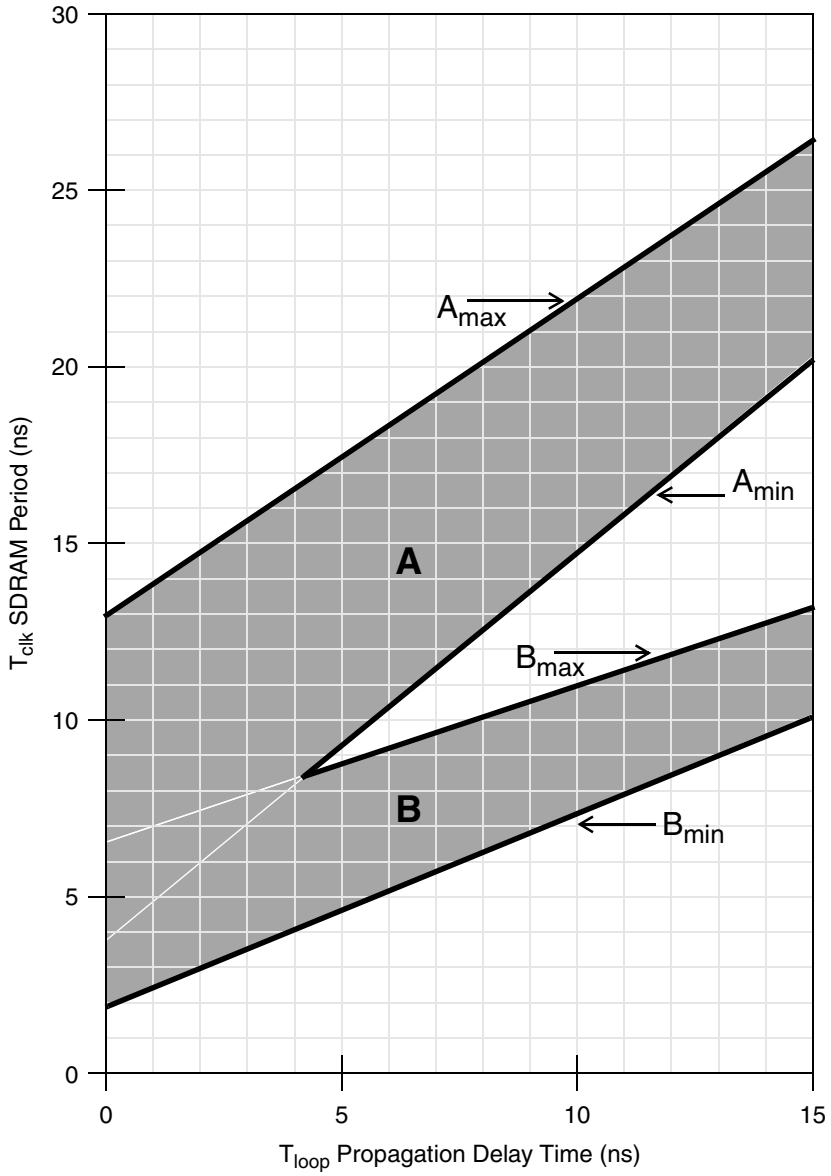

Figure 3. DLL Locking Region for Normal Tap Delay and Non-DLL-Extended Mode

Equation for A_{max}: $T_{clk} = (0.90 \times T_{loop}) + 13.08$ ns

Equation for A_{min}: $T_{clk} = (1.10 \times T_{loop}) + 3.78$ ns

Equation for B_{max}: $T_{clk} = (0.45 \times T_{loop}) + 6.54$ ns

Equation for B_{min}: $T_{clk} = (0.55 \times T_{loop}) + 1.89$ ns

The area overlapped by values below A_{max} and above A_{min} represents the region of DLL locking. Hence, the relationship between T_{loop} and T_{clk} is such that:

$$[1.10 \times (T_{clk} - 13.08)] < T_{loop} < [0.90 \times (T_{clk} - 3.78)] \text{ for region A.}$$

This relationship between T_{loop} and T_{clk} determines the range of T_{loop} guaranteed for DLL locking by mathematical analysis.

Similarly, the area overlapped by values below B_{max} and above B_{min} represents the region of DLL locking. Hence, the relationship between T_{loop} and T_{clk} is such that:

$$[2.20 \times (T_{clk} - 6.54)] < T_{loop} < [1.82 \times (T_{clk} - 1.89)] \text{ for region B.}$$

This relationship between T_{loop} and T_{clk} determines the range of T_{loop} guaranteed for DLL locking by mathematical analysis.

Note that the equations defining DLL locking regions may be used to calculate the range of T_{loop} for a particular memory bus frequency (inverse of T_{clk}), or a direct determination may be made by observing the data points of the graph. In the case where the calculated lower limit of T_{loop} is negative, zero should be considered as the lower limit. Note that for this mode there is a region of overlap for which some values above A_{min} are equal to some values below B_{max}.

3.2.2 Case 2: Normal Tap Delay and DLL-Extended Mode

Normal tap delay—MIOCR1[DLL_MAX_DELAY] is cleared.

DLL extend—PMCR2[DLL_EXTEND] is set.

Figure 4 shows two grey bands (call them A and B) that have been identified as the region for DLL locking when in normal tap delay and DLL-extended mode.

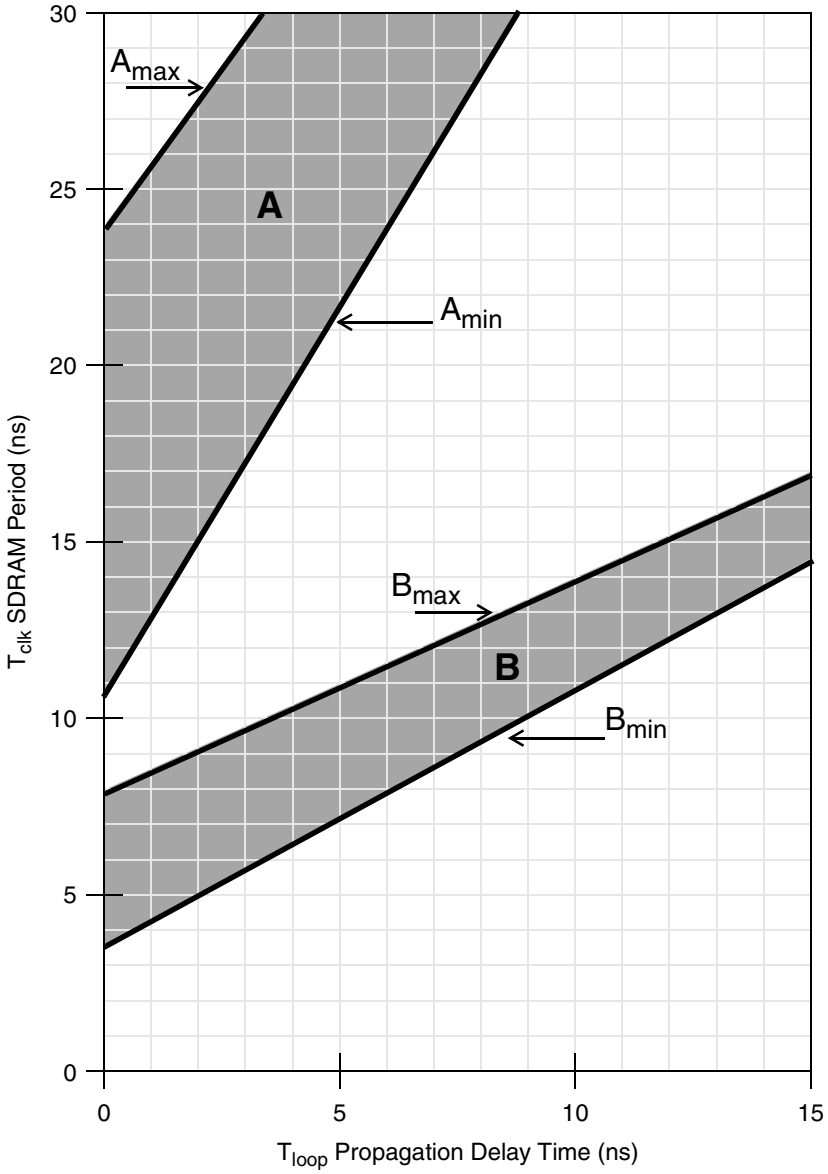

Figure 4. DLL Locking Region for Normal Tap Delay and DLL-Extended Mode

Equation for A_{max}: $T_{clk} = (1.80 \times T_{loop}) + 23.98$ ns

Equation for A_{min}: $T_{clk} = (2.20 \times T_{loop}) + 10.56$ ns

Equation for B_{max}: $T_{clk} = (0.60 \times T_{loop}) + 7.99$ ns

Equation for B_{min}: $T_{clk} = (0.73 \times T_{loop}) + 3.52$ ns

The area overlapped by values below A_{max} and above A_{min} represents the region of DLL locking. Hence, the relationship between T_{loop} and T_{clk} is such that:

$$[0.56 \times (T_{clk} - 23.98)] < T_{loop} < [0.45 \times (T_{clk} - 10.56)] \text{ for region A.}$$

This relationship between T_{loop} and T_{clk} determines the range of T_{loop} guaranteed for DLL locking by mathematical analysis.

Similarly, the area overlapped by values below B_{max} and above B_{min} represents the region of DLL locking. Hence the relationship between T_{loop} and T_{clk} is such that:

$$[1.67 \times (T_{clk} - 7.99)] < T_{loop} < [1.36 \times (T_{clk} - 3.52)] \text{ for region B.}$$

This relationship between T_{loop} and T_{clk} determines the range of T_{loop} guaranteed for DLL locking by mathematical analysis.

Note that the equations defining DLL locking regions may be used to calculate the range of T_{loop} for a particular memory bus frequency (inverse of T_{clk}), or a direct determination may be made by observing the data points of the graph. In the case where the calculated lower limit of T_{loop} is negative, zero should be considered as the lower limit.

3.2.3 Case 3: Maximum Tap Delay and Non-DLL-Extended Mode

Maximum tap delay—MIOCR1[DLL_MAX_DELAY] is set.

No DLL extend—PMCR2[DLL_EXTEND] is cleared.

Figure 5 shows two grey bands (call them A and B) that have been identified as the region for DLL locking for maximum tap delay and non-DLL-extended mode.

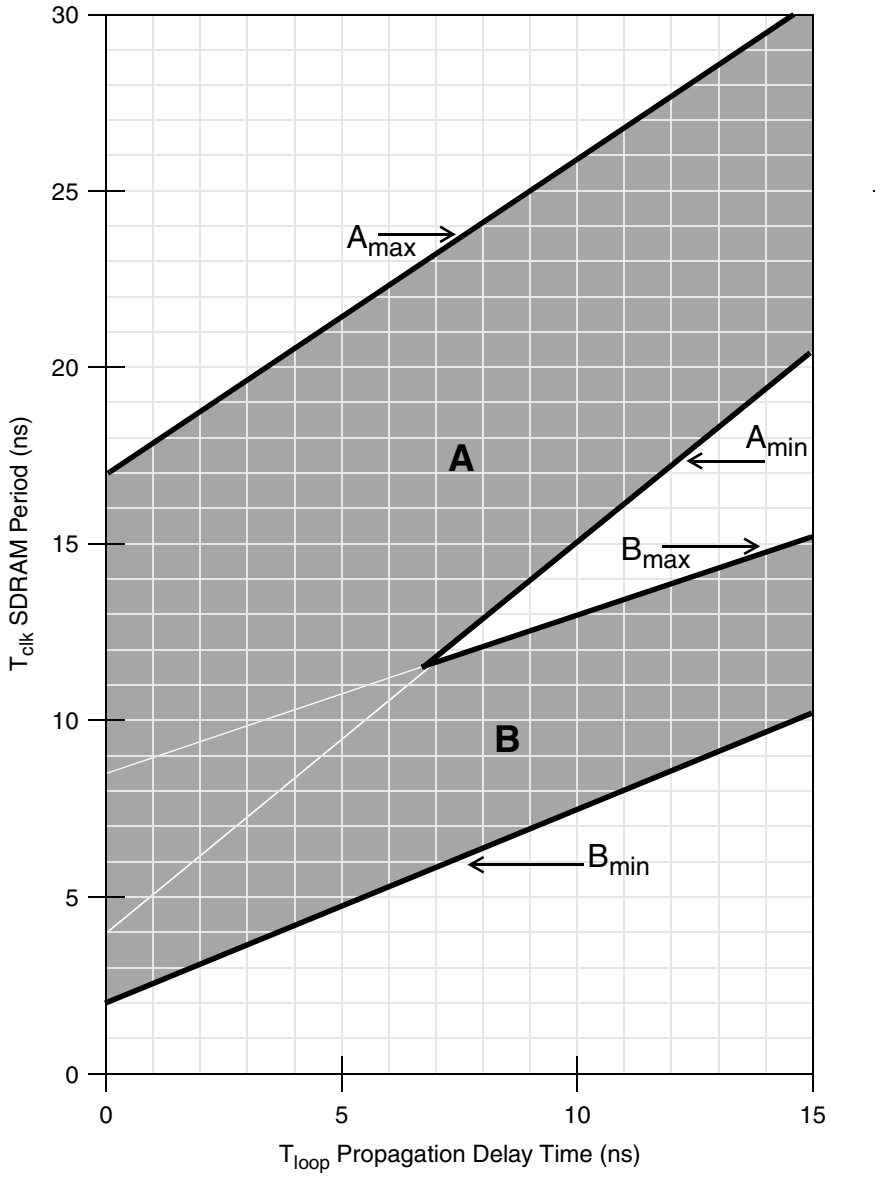

Figure 5. DLL Locking Region for Maximum Tap Delay and Non-DLL-Extended Mode

Equation for A_{max}: $T_{clk} = (0.90 \times T_{loop}) + 16.99$ ns

Equation for A_{min}: $T_{clk} = (1.10 \times T_{loop}) + 3.92$ ns

Equation for B_{max}: $T_{clk} = (0.45 \times T_{loop}) + 8.49$ ns

Equation for B_{min} : $T_{clk} = (0.55 \times T_{loop}) + 1.95$ ns

The area overlapped by values below A_{max} and above A_{min} represents the region of DLL locking. Hence, the relationship between T_{loop} and T_{clk} is such that:

$$[1.1 \times (T_{clk} - 16.99)] < T_{loop} < [0.90 \times (T_{clk} - 3.92]\text{for region A.}$$

This relationship between T_{loop} and T_{clk} determines the range of T_{loop} guaranteed for DLL locking by mathematical analysis.

Similarly, the area overlapped by values below B_{max} and above B_{min} represents the region of DLL locking. Hence, the relationship between T_{loop} and T_{clk} is such that:

$$[2.20 \times (T_{clk} - 8.49)] < T_{loop} < [1.82 \times (T_{clk} - 1.95)]\text{for region B.}$$

This relationship between T_{loop} and T_{clk} determines the range of T_{loop} guaranteed for DLL locking by mathematical analysis.

Note that the equations defining DLL locking regions may be used to calculate the range of T_{loop} for a particular memory bus frequency (inverse of T_{clk}), or a direct determination may be made by observing the data points of the graph. In the case where the calculated lower limit of T_{loop} is negative, zero should be considered as the lower limit. Note that for this mode there is a region of overlap for which some values above A_{min} are equal to some values below B_{max}.

3.2.4 Case 4: Maximum Tap Delay and DLL-Extended Mode

Maximum tap delay—MIOCR1[DLL_MAX_DELAY] is set.

DLL extend—PMCR2[DLL_EXTEND] is set.

Figure 6 shows two grey bands (call them A and B) that have been identified as the region for DLL locking based on maximum tap delay and DLL-extended mode.

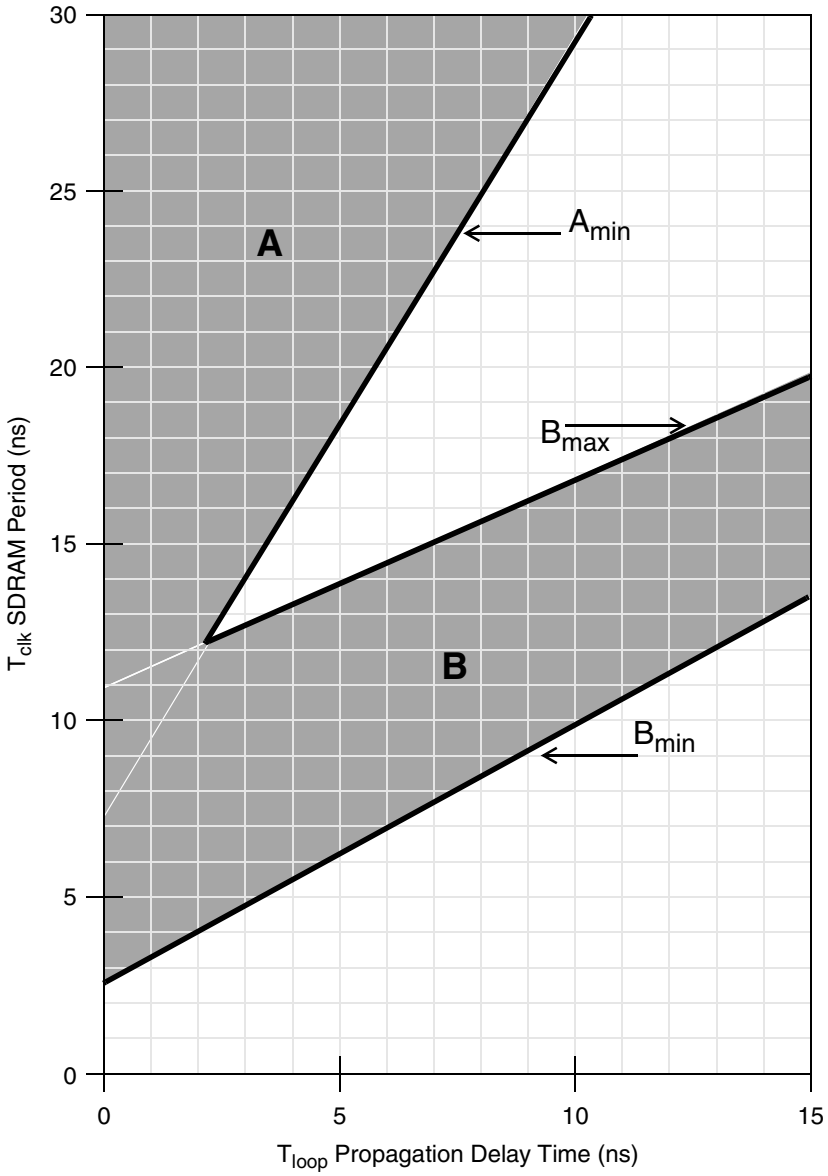

Figure 6. DLL Locking Region for Maximum Tap Delay and DLL-Extended Mode

Equation for A_{max}: $T_{clk} = (1.80 \times T_{loop}) + 32.58$ ns (not shown in graph)

Equation for A_{min}: $T_{clk} = (2.20 \times T_{loop}) + 7.39$ ns

Equation for B_{max}: $T_{clk} = (0.60 \times T_{loop}) + 10.86$ ns

Equation for B_{min}: $T_{clk} = (0.73 \times T_{loop}) + 2.46$ ns

The area overlapped by values below A_{max} and above A_{min} represents the region of DLL locking. Hence, the relationship between T_{loop} and T_{clk} is such that:

$$[0.56 \times (T_{clk} - 32.58)] < T_{loop} < [0.45 \times (T_{clk} - 7.39] \text{for region A.}$$

This relationship between T_{loop} and T_{clk} determines the range of T_{loop} guaranteed for DLL locking by mathematical analysis.

Similarly, the area overlapped by values below B_{max} and above B_{min} represents the region of DLL locking. Hence, the relationship between T_{loop} and T_{clk} is such that:

$$[1.67 \times (T_{clk} - 10.86)] < T_{loop} < [1.36 \times (T_{clk} - 2.46)] \text{for region B.}$$

This relationship between T_{loop} and T_{clk} determines the range of T_{loop} guaranteed for DLL locking by mathematical analysis.

Note that the equations defining DLL locking regions may be used to calculate the range of T_{loop} for a particular memory bus frequency (inverse of T_{clk}), or a direct determination may be made by observing the data points of the graph. In the case where the calculated lower limit of T_{loop} is negative, zero should be considered as the lower limit. Note that for this mode there is a region of overlap for which some values above A_{min} are equal to some values below B_{max}.

4 Application Examples

To obtain the trace length of SDRAM_SYNC_OUT to SDRAM_SYNC_IN calculate the expected value of T_{loop} using these DLL locking region figures and the equations for DLL lock range. The following are examples using two of the previously listed DLL modes. Assume that 1 ns of T_{loop} represents 6.25 inches of actual trace length.

4.1 Example 1

By mathematical analysis:

Assumptions:

- Memory bus speed is 100 MHz; hence, the value of T_{clk} is 10 ns.
- DLL_EXTEND = 0 (PMCR2[DLL_EXTEND] is cleared).
- Normal tap delay is used (MIOCR1[DLL_MAX_DELAY] is cleared).

Recall that there are two equations for T_{loop} based on region A and B of the graph for this mode, defined in Section 3.2.1, “Case 1: Normal Tap Delay and Non-DLL-Extended Mode.” The two equations provide the relationship between T_{loop} and T_{clk}.

$$[1.10 \times (T_{clk} - 13.08)] < T_{loop} < [0.90 \times (T_{clk} - 3.78)] \text{ for region A.}$$

$$[2.20 \times (T_{clk} - 6.54)] < T_{loop} < [1.82 \times (T_{clk} - 1.89)] \text{ for region B.}$$

To find out if there is a case for a range of T_{loop} available in either region A or B, use the equations as follows:

- For region A, $[1.10 \times (T_{clk} - 13.08)] < T_{loop} < [0.90 \times (T_{clk} - 3.78)]$.

  Hence, if T_{clk}=10 ns, this implies that:

  $[-3.39 < T_{loop} < 5.59]$.

  Based on this inequality, the range of T_{loop} for region A is between 0 and 5.59 ns (between 0 and 34.9 inches).
- For region B, $[2.20 \times (T_{clk} - 6.54)] < T_{loop} < [1.82 \times (T_{clk} - 1.89)]$.

  Hence, if T_{clk}=10 ns, this implies that:

  $[7.61 < T_{loop} < 14.76]$.

  Based on this inequality, the range of T_{loop} for region B is between 7.61 and 14.76 ns (between 47.56 and 92.25 inches).

Compare the above method of finding T_{loop} to that of the graphical analysis of Figure 7 that follows.

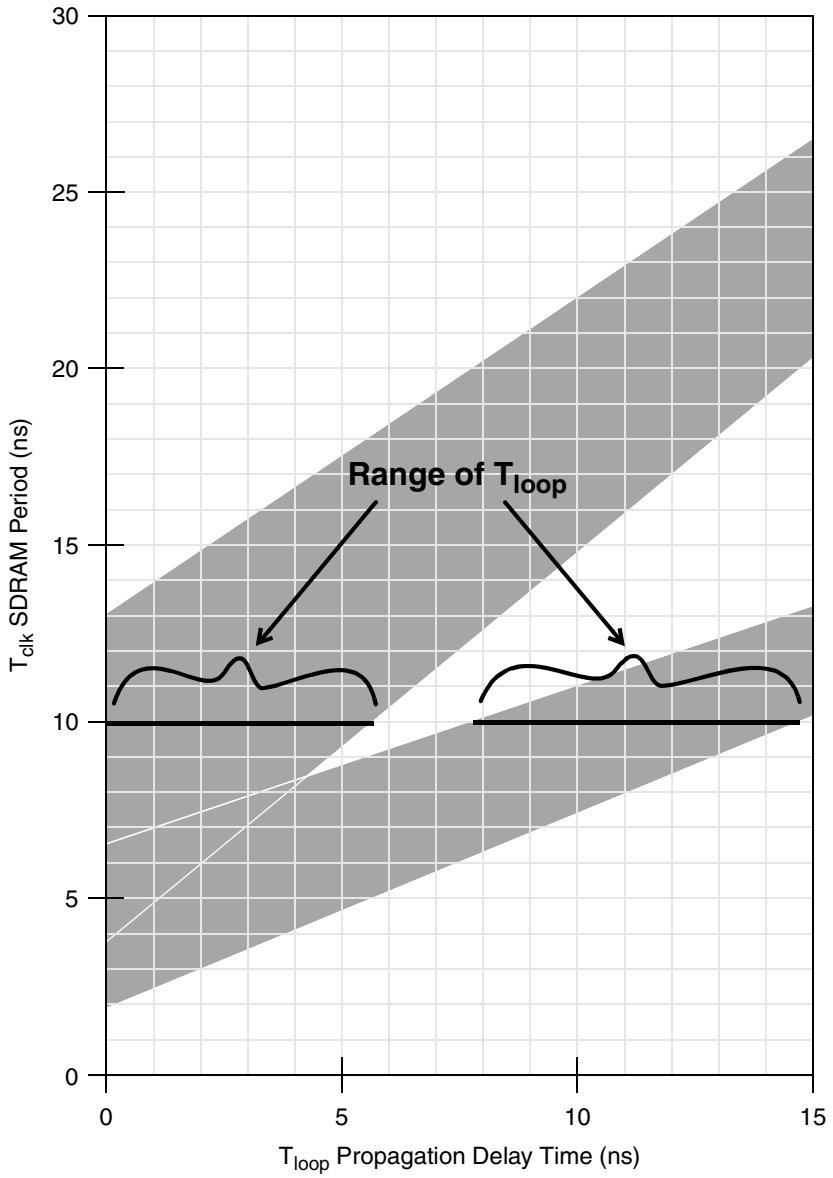

Figure 7. DLL Locking Verification for T_{loop} for Normal Tap Delay and Non-DLL-Extended Mode

Using the graph in Figure 7 (same as Figure 3).

1. Find where T_{loop} is in the grey region for T_{clk} = 10 ns.
2. Note that the solid line shows the range of T_{loop} where the DLL is expected to lock. This is between 0 and 5.6 ns and also between 7.6 and 14.8 ns, which is about what was calculated for the range of T_{loop}.

Hence, if the chosen delay for T_{loop} is 2 ns, the expected trace length is 2×6.25 inches, since 1 ns represents about 6.25 inches of trace length. Therefore, the length that SDRAM_SYNC_OUT to SDRAM_SYNC_IN and the SDRAM clocks should be 12.50 inches.

4.2 Example 2

By mathematical analysis:

Assumptions:

- Memory bus speed is 133 MHz; hence, the value of T_{clk} is 7.5 ns.
- DLL_EXTEND = 1 (PMCR2[DLL_EXTEND] is set).
- Maximum tap delay is used (MIOCR1[DLL_MAX_DELAY] is set).

Recall that there are two equations for T_{loop} based on region A and B of the graph for this mode as defined in Section 3.2.4, “Case 4: Maximum Tap Delay and DLL-Extended Mode.” The two equations provide the relationship between T_{loop} and T_{clk}.

$$[0.56 \times (T_{clk} - 32.58)] < T_{loop} < [0.45 \times (T_{clk} - 7.39] \text{ for region A.}$$

$$[1.67 \times (T_{clk} - 10.86)] < T_{loop} < [1.36 \times (T_{clk} - 2.46)] \text{ for region B.}$$

To find out if there is a case for a range of T_{loop} available in either region A or B, use the equations as follows:

- For region A, $[0.56 \times (T_{clk} - 32.58)] < T_{loop} < [0.45 \times (T_{clk} - 7.39]$.
  Hence, for T_{clk}=7.5 ns, this implies that:
  $[-14.04 < T_{loop} < 0.05]$.
  Based on this inequality, the range of T_{loop} for region A is between 0 and 0.05 ns (between 0 and 0.3 inches).
- For region B, $[1.67 \times (T_{clk} - 10.86)] < T_{loop} < [1.36 \times (T_{clk} - 2.46)]$
  Hence, for T_{clk}=7.5 ns, this implies that:
  $[-5.61 < T_{loop} < 6.85]$.
  Based on this inequality, the range of T_{loop} for region B is between 0 and 6.85 ns (between 0 and 42.81 inches). Note that since the range for region A is a subset of the range for region B, the latter range is what should be considered.

Compare the above method of finding T_{loop} to that of the graphical analysis of Figure 8 that follows.

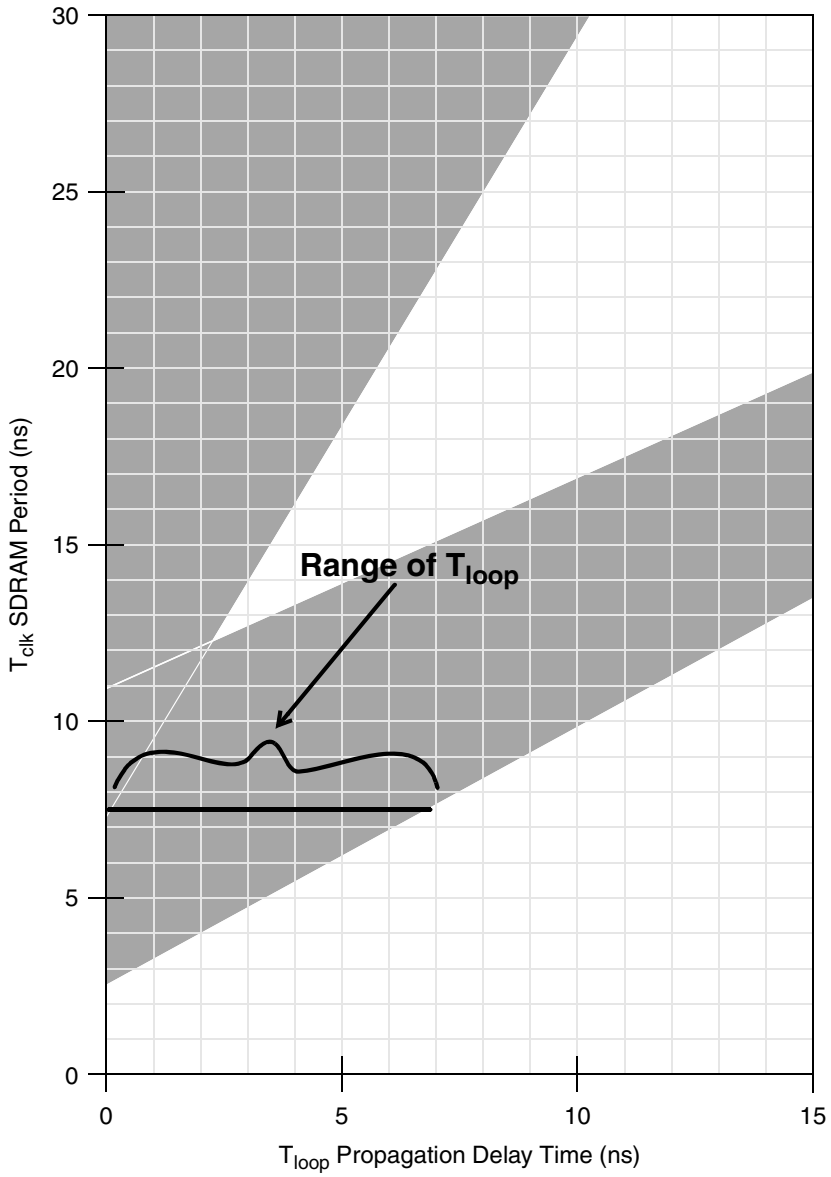

Figure 8. DLL Locking Verification for T_{loop} for Maximum Tap Delay and DLL-Extended Mode

Using the graph in Figure 8 (same as Figure 6).

1. Find where T_{loop} is in the grey region for T_{clk} = 7.5 ns.
2. Note that the solid line shows the range of T_{loop} where the DLL is expected to lock. This is between 0 and 6.8 ns which is about what was calculated for the range of T_{loop}.

Hence, if the chosen delay for T_{loop} is 1.8 ns, the expected trace length is 1.8×6.25 inches, since 1 ns represents about 6.25 inches of trace length. Therefore, the length that SDRAM_SYNC_OUT to SDRAM_SYNC_IN and the SDRAM clocks should be 11.25 inches.

5 T_{os} (SDRAM_SYNC_IN to *sys_logic_clk*)

The MPC8245/MPC8241 has an internal delay in the feedback path for SDRAM_SYNC_IN with respect to the internal *sys_logic_clk* signal. The DLL tries to compensate for this delay when it compares the clock of SDRAM_SYNC_IN to that of *sys_logic_clk*. The result is that the adjustment causes the DLL to try to lock earlier than it would without the delay. This causes SDRAM_SYNC_IN to be seen earlier than *sys_logic_clk* when the two signals are observed after the DLL locks. Based on this characterization, the accommodation for the delay ranges from 0.4 to 1.0 ns. This is referred to as T_{os} and represents the timing adjustment for SDRAM_SYNC_IN with respect to *sys_logic_clk*. Figure 9 shows the way that SDRAM_CLK at the memory is seen after DLL locking, before and after making an adjustment for T_{os}.

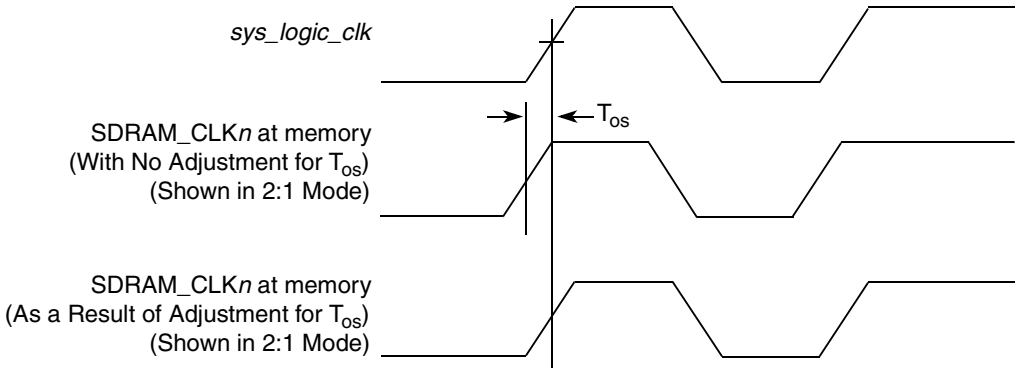

Figure 9. T_{os} Relative to *sys_logic_clk* and SDRAM_CLKs after the DLL Locks

The feedback trace length of SDRAM_SYNC_OUT to SDRAM_SYNC_IN should be shortened by 0.7 ns so that the impact of T_{os} can be reduced. This is because 0.7 ns is the midpoint of the range of T_{os} and allows the impact from the range of T_{os} to be reduced. Additional analyses that take into account trace lengths and SDRAM loading, need to be performed to optimize the timing. During the timing optimization evaluation, various input setup and hold times can be tried based on the timing for the bit settings of MIOCR2[SDRAM_DSCD]. For example, if MIOCR2 is set to 0b*nn*00*nnnn* it allows for the maximum input setup time from the programming options. Please see the hardware specifications documents for the MCP8241 and the MPC8245 for details of the times that pertain to various bit settings.The problem of the SDRAM clocks accessing memory early needs to be examined on a case-by-case basis, considering the overall system architecture, including memory bus speed and the impact of certain trace lengths used. In some cases, using the programmability of the tap points may be an alternative to consider before shortening the trace length of SDRAM_SYNC_OUT to SDRAM_SYNC_IN for memory bus speeds of 100 MHz and below.

6 DLL Tap Count Register (DTCR[6–0], Offset 0xE3)

The value of the current tap point once the DLL has locked can be determined by reading bits 6–0 (DLL_TAP_COUNT) of the DLL tap count register (DTCR, located at offset 0xE3). These bits store the value (binary 0 through 127) of the current tap point and can indicate whether the DLL advances or decrements as it maintains the DLL lock. Therefore, for evaluation purposes, DTCR can be read for all DLL modes that support the T_{loop} value that is used for the trace length of SDRAM_SYNC_OUT to SDRAM_SYNC_IN. The DLL mode that provides the smallest tap point value seen in DTCR should be used. This is because the bigger the tap point value, the more jitter that can be expected for clock signals.

7 Conclusion

When designing for the MPC8245/MPC8241, the DLL locking range must be considered to design the appropriate trace lengths for SDRAM_SYNC_OUT to SDRAM_SYNC_IN. The offset adjustment, T_{os}, should be removed from T_{loop} (SDRAM_SYNC_OUT to SDRAM_SYNC_IN) to create the final trace length or the software workaround of setting MIOCR2 to 0b*nn*00*nnnn* should be used in cases where T_{loop} is less than 0.65 ns. It is necessary to evaluate the trace lengths on a design-specific basis, with the use of the DLL locking graphs. Toggling AMBOR[DLL_RESET] must be done after setting up a desired DLL mode in the initialization process.

8 Revision History

Table 3 provides a revision history for this application note.

Table 3. Document Revision History

Revision	Date	Substantive Change(s)
0	07/2001	Initial document.
1	11/2001	Added explanation about DLL_Extend in first paragraph of Section 1.1. Updated paragraph 2 of Section 1.1 to better explain jitter with increased tap delay length. Removed Figure 2 and Table 2 and their explanation.
2	09/2002	Rearranged format of document for better flow of information. Updated entire document to be more comprehensive for T_{os} documentation. Added application examples. Nontechnical reformatting (added Tundra Tsi107 trademarking).
3	03/2004	Updated DLL graphs, equations and examples. Rearranged sections for better flow of information. Added section on DTCR[DLL_TAP_COUNT] (offset 0xE3). Nontechnical reformatting.
3.1	03/2004	In Section 2, the lock range change caused by using the DLL_EXTEND bit was corrected from one half of a PCI clock cycle to one half of an SDRAM clock cycle. In Section 5, the first paragraph was simplified.
3.2	05/13/04	Updated T_{os} range and recommendations in Section 5, "T_{os} (SDRAM_SYNC_IN to sys_logic_clk)."
4	08/31/04	Updated to Freescale template Changed name of application note to represent Part 1 of a two-part series. Added sentences in introduction regarding application note AN2746, the second part of this two-part series.
4.1	3/16/07	Changed title of Figure 9 from " T_{os} Relative to *sys_logic_clk* and SDRAM_SYNC_IN After the DLL Locks" to "T_{os} Relative to *sys_logic_clk* and SDRAM_CLks after the DLL Locks"

THIS PAGE INTENTIONALLY LEFT BLANK

THIS PAGE INTENTIONALLY LEFT BLANK

THIS PAGE INTENTIONALLY LEFT BLANK

How to Reach Us:

Home Page:
www.freescale.com

email:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
(800) 521-6274
480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Technical Information Center
3-20-1, Minami-Azabu, Minato-ku
Tokyo 106-0047 Japan
0120 191014
+81 3 3440 3569
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate,
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor
Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
(800) 441-2447
303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@
hibbertgroup.com

AN2164
Rev. 4.1
03/2007

freescale™
semiconductor