



XAPP879 (v1.0) May 13, 2010

PLL Dynamic Reconfiguration

Author: Karl Kurbjun and Carl Ribbing

Summary

This application note provides a method to dynamically change the clock output frequency, phase shift, and duty cycle of the Spartan®-6 FPGA Phase Locked Loop (PLL) through its Dynamic Reconfiguration Port (DRP). An explanation of the behavior of the internal DRP control registers is accompanied by a reference design. The reference design uses a state machine to drive the DRP to ensure that the registers are controlled in the correct sequence.

Caution! If post-configuration cyclic redundancy check (CRC) functionality is needed in the design, the PLL DRP port cannot be used to dynamically reconfigure the PLL. Use of the PLL DRP port breaks the functionality of post-configuration CRC.

Although the reference design performs the operations for the user, familiarity with the functional operation of the PLL is recommended. The PLL used in conjunction with the DRP interface is recommended for advanced users when the basic PLL functionality is not sufficient. The DCM_CLKGEN primitive can be a useful alternative to using the PLL with the DRP interface. For more information on PLL functionality, see [UG382](#), *Spartan-6 FPGA Clocking Resources User Guide*.

The reference design supports two reconfiguration state addresses and can be extended to support additional states. Each state does a full reconfiguration of the PLL so that most parameters can be changed.

Introduction

The clock management tiles (CMT) in the Spartan-6 devices contain two DCMs and one PLL. One of the most powerful features of the PLL is its ability to dynamically reconfigure the phase, duty cycle, and divide values of the clock outputs. This application note describes the information necessary to reconfigure the PLL, and provides a reference design that implements all of the algorithms covered. The PLL used in this reference design is intended to be used with CLKFBOUT as the feedback path. The reference design does not support the use of CLKOUT for the feedback path.

Reconfiguration is performed through the PLL DRP. The DRP provides access to the configuration bits that would normally only be initialized in the bitstream. This allows the user to dynamically change the PLL clock outputs while the design is running. Frequency, phase, and duty cycle can all be changed dynamically. To properly reconfigure the PLL, it must be initially set up with integer divider values.

The [PLL Configuration Bit Groups](#) and [PLL DRP Registers](#) sections present the configuration bits as four bit groups, provide an overview of their usage, and detail the configuration bit locations as registers. This information is not necessary to use the DRP reference design; it is intended to give an overview of the internal PLL attributes that must be changed along with their register locations. Specific information on how the attributes are calculated is provided through the reference design. The reference design functionality and use are explained in the [Reference Design](#) and [Using the Reference Design](#) sections.

PLL Configuration Bit Groups

The PLL has four user-accessible configuration bit groups that allow reconfiguration of individual clock outputs. The four groups are the divider group, the phase group, the lock group, and the filter group. These configuration bit groups are internal to the PLL primitive and clarify the operation of the PLL_DRP module. The user modifiable parameters for the PLL_DRP module are described in the [Reconfiguration Module Ports and Attributes](#) section.

Divider Group

Every clock output has a divider group associated with it. The divider group is composed of the following parameters:

- High Time
- Low Time
- No Count
- Edge

The first two parameters associated with the divider group are the High and Low Time counters. These counters set the number of voltage-controlled oscillator (VCO) clock cycles through which the output clock should stay High or Low. For example, if you set both High and Low Time to 2, the effective divide value is 4 and the duty cycle is 50%.

The No Count parameter disables the High and Low Time counters. This in turn makes the divider output a clock with an effective divide value of 1.

The Edge parameter controls the High to Low transition. It forces the High Time counter to transition on a falling edge at the end of its count. This has the effect of increasing the High Time while decreasing the Low Time. Another way to think of the edge bit is that it adds half a VCO clock cycle to the High Time and subtracts half a clock cycle from the Low Time.

As an example, if a 50/50 duty cycle is desired with a divide value of 3, the Edge bit would be set. The High Time counter would be set to one and the Low Time counter would be set to 2. With the edge bit set, the net count for the High and Low times would be 1.5 clock cycles each.

Phase Group

Each clock output except the DIVCLK has a phase group associated with it. This group is composed of the following set of parameters:

- Phase MUX
- Delay Time

The Phase MUX selects a coarse phase from the VCO for a clock output with a resolution of 45° (360°/8) relative to the VCO clock period.

Delay Time is a counter that counts the number of VCO clock cycles to delay the output. This means that there is a direct correlation between the possible phase shift for the clock output and the divide value for that particular output. As the divide value increases, finer phase shift steps are available. The Delay Time counter allows for a phase offset of up to 64 VCO clock cycles.

Lock Group

This group cannot be calculated with an algorithm and is based on lookup tables created from device characterization. The appropriate lock bit settings are dependent on the feedback divider setting. This divider is set with the CLKFBOUT_MULT attribute when instantiating the PLL_DRP module. The lock group has an effect on the PLL's ability to detect that it is locked. The lookup table is located in the reference design within `pll_drp_func.h`.

Filter Group

This group cannot be calculated, and is based on lookup tables created from device characterization. There are effectively two tables, one for each bandwidth setting. The feedback divider setting (CLKFBOUT_MULT) acts as the index to the chosen table. There are three bandwidth settings allowable in the tools (High, Low, and Optimized), but in effect, there are only two. High and Optimized use the same table, while the Low bandwidth setting uses a separate table. The filter group has an effect on the phase skew and the jitter filtering capability of the PLL. The lookup table is located in the reference design within `pll_drp_func.h`.

PLL DRP Registers

Each clock output is associated with a number of configuration bits. The CLKOUT and CLKFBOUT outputs are associated with phase and divider group bits. The locked and filter bits are associated with the CLKFBOUT configuration. The DIVCLK output does not have any associated phase configuration bits. [Figure 1](#) shows the six clock outputs, the feedback clock output, and the DIVCLK (indicated by D in [Figure 1](#)).

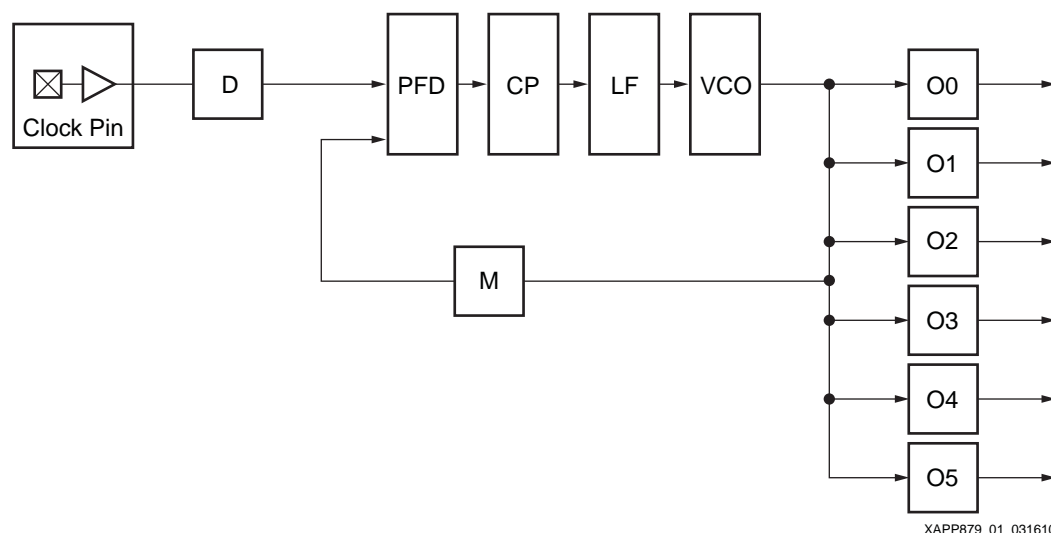


Figure 1: PLL Block Diagram

The PLL DRP address map, which stores the configuration bits, is provided in the [Appendix, page 11](#). [Table 1](#) describes the functionality associated with each register type used in the DRP address map.

Table 1: PLL Register Type Description

Register Type	Width	Description
PHASE MUX	3	This register type chooses an initial phase offset for the clock output. The resolution of the offset is equal to 1/8 of the VCO period.
HIGH TIME	6	This register type sets the amount of time, in VCO cycles, that the clock output remains High.
LOW TIME	6	This register type sets the amount of time, in VCO cycles, that the clock output remains Low.
DELAY TIME	6	This register type sets the phase offset with a resolution equal to the VCO period.
NO COUNT	1	This register type bypasses the High and Low time counters.
EDGE	1	This register type chooses the edge on which the High time counter transitions.

Table 1: PLL Register Type Description (Cont'd)

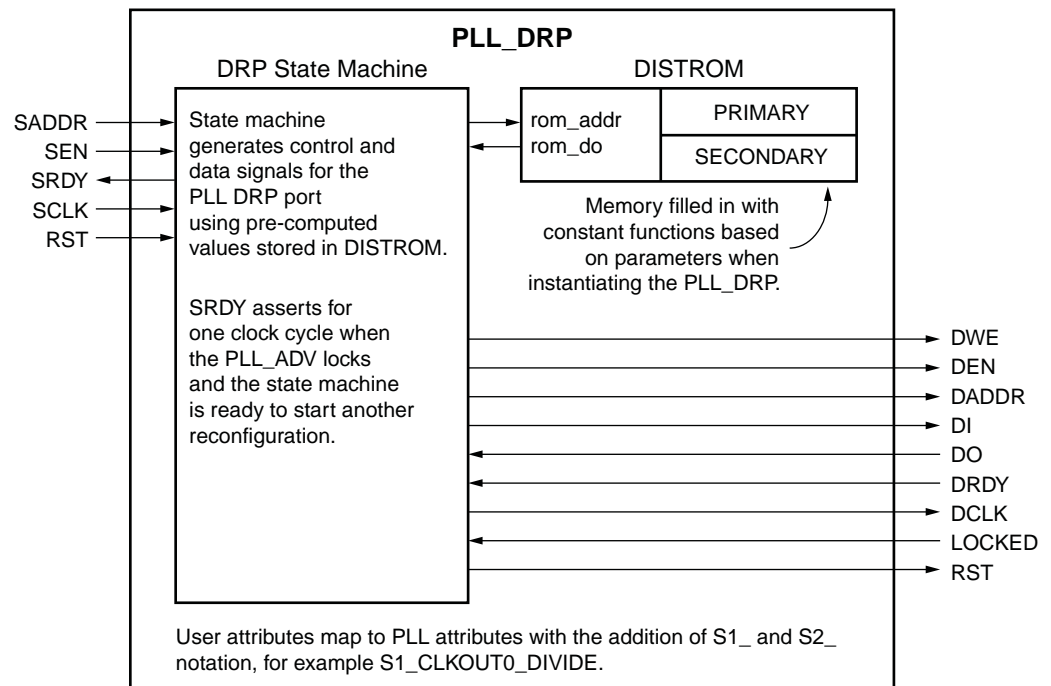
Register Type	Width	Description
LOCK TABLE	40	These bits are pulled from the lock lookup table provided in the reference design.
FILTER TABLE	10	These bits are pulled from the filter lookup table provided in the reference design.
RESERVED	X	This register type retains the previous value stored here.

Reference Design

The reference design files include a Verilog PLL reconfiguration module. This module uses only 25 total slices, comprising the reconfiguration logic and state memory.

The reference design drives the DRP port with a state machine that addresses the PLL, reads the previous value, masks the bits that need to be changed, sets the new value, and finally writes the value to the PLL DRP port. The addresses, masks, and new values are stored in a pre-initialized ROM that is filled during elaboration in synthesis. The ROM initialization is done with constant functions provided with the reference design.

Figure 2 is a block diagram of the reconfiguration module.



XAPP879_02_031610

Figure 2: PLL_DRP Internal Block Diagram

The `p11_drp.v` module contains the state machine and ROM, and calls the constant functions which are provided in `p11_drp_func.h`.

Figure 3 shows the block diagram of the system with the `p11_adv` and the `p11_drp` modules attached.

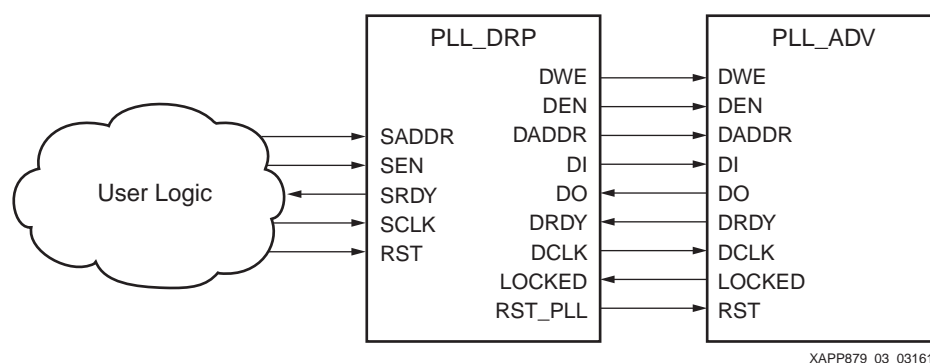


Figure 3: Reference Design Block Diagram

DRP State Machine

The DRP state machine is composed of the nine states shown in Table 2. It controls all of the signals from the `pll_drp` module.

Table 2: DRP States

State	Description	Next State	Transition Condition
RESTART	This state is entered whenever the SRST pin is asserted or if the current_state goes into an undefined state.	WAIT_LOCK	SRST = 0
WAIT_LOCK	This state waits for the lock signal from the PLL to assert. When lock is asserted, SRDY = 1.	WAIT_SEN	LOCKED = 1
WAIT_SEN	This state waits for SEN to be asserted and sets the appropriate ROM address according to SADDR.	ADDRESS	SEN = 1
ADDRESS	This state is entered from either WAIT_SEN or from WAIT_DRDY. The state sets DADDR according to the current value stored in the ROM and asserts DEN.	WAIT_A_DRDY	<always>
WAIT_A_DRDY	This state is always entered from ADDRESS. It waits for the PLL to assert the DRDY signal.	BITMASK	DRDY = 1
BITMASK	This state is always entered from WAIT_A_DRDY. It performs a bitwise AND on DO from the PLL with the mask value stored in the ROM.	BITSET	<always>
BITSET	This state is always entered from BITMASK. It performs a bitwise OR with the bitset stored in the ROM and the output from the BITMASK operation.	WRITE	<always>

Table 2: DRP States (Cont'd)

State	Description	Next State	Transition Condition
WRITE	This state asserts DEN, DWE, and RST_PLL. It updates the state counter which is used to keep track of the number of register writes needed to perform one full reconfiguration.	WAIT_DRDY	<always>
WAIT_DRDY	This state waits for DRDY to assert from the PLL.	ADDRESS (state_count > 0) WAIT_LOCK (state_count ≤ 0)	DRDY = 1

In short, the operations that must be implemented to reconfigure one value in the PLL are:

- Assert RST to the PLL (do not de-assert)
- Set DADDR on the PLL and assert DEN for one clock cycle
- Wait for the DRDY signal to assert from the PLL
- Perform a bitwise AND between the DO port and the MASK (DI = DO and MASK)
- Perform a bitwise OR between the DI signal and the BITSET (DI = DI or BITSET)
- Assert DEN and DWE on the PLL for one clock cycle
- Wait for the DRDY signal to assert from the PLL
- De-assert RST to the PLL
- Wait for PLL to lock

Reconfiguration Module Ports and Attributes

The reconfiguration module is composed of the ports shown in [Table 3](#).

Table 3: PLL Reconfiguration Ports

Port	Direction	Description
SADDR	Input	This port chooses the state to reconfigure the PLL to. A value of 0 corresponds to state 1; a value of 1 corresponds to state 2.
SEN	Input	This port enables the reconfiguration state machine. Reconfiguration is triggered if this port is asserted at a rising SCLK edge.
SCLK	Input	This is the clock for the reconfiguration module. It is passed through to the DCLK output.
RST	Input	This resets the state machine and the downstream PLL.
SRDY	Output	This port asserts for one clock cycle at the end of a reconfiguration sequence. It is to be used to notify the user that a new reconfiguration can be started.
DO[15:0]	Input	This port should be directly attached to the PLL DO port. It is used to read register values from the PLL.
DRDY	Input	This port should be directly attached to the PLL DRDY port. It is used to notify the reference design when the PLL is ready to read or write a new value.
LOCKED	Input	This port should be directly attached to the PLL LOCKED port. It is used to notify the reference design that the PLL is locked and to then transition from the WAIT_LOCK state.
DWE	Output	This port should be directly attached to the PLL DWE port. It is used to enable a register write.
DEN	Output	This port should be directly attached to the PLL DEN port. It is used to initiate a register read or write.

Table 3: PLL Reconfiguration Ports (Cont'd)

Port	Direction	Description
DADDR[6:0]	Output	This port should be directly attached to the PLL DADDR port. It is used to address a register location for reads or writes.
DI[15:0]	Output	This port should be directly attached to the PLL DI port. It is used to output a new register value for writes.
DCLK	Output	This port should be directly attached to the PLL DCLK port. It is used to clock the reconfiguration port on the PLL. It is the SCLK forwarded out of the PLL reconfiguration module.
RST_PLL	Output	This port should be directly attached to the PLL RST port. It is used to reset the PLL during a reconfiguration or when the RST port is asserted.

The reconfiguration module also has the attributes shown in Table 4. The PLL_DRP attributes correlate with the standard PLL primitive attributes with some slight naming differences.

Table 4: PLL Reconfiguration Attributes

Attribute	Description	Valid Format Values
CLKFBOUT_MULT	This attribute modifies the input clock multiplier to change the VCO output frequency of the PLL.	1–64; Integer values only.
CLKFBOUT_PHASE	Modifies the phase of the input clock. This affects all of the PLL outputs.	Integer values multiplied by 1,000. For example, a 45° phase shift would be 45,000
BANDWIDTH	Sets the bandwidth setting of the PLL.	OPTIMIZED, HIGH, or LOW.
DIVCLK_DIVIDE	Sets the divide value for the DIVCLK output.	1–128; Integer values only.
CLKOUT0_DIVIDE	CLKOUT0 output divide value.	1–128; Integer values only.
CLKOUT0_PHASE	CLKOUT0 output phase value.	Integer values multiplied by 1,000. For example, a 45° phase shift would be 45,000.
CLKOUT0_DUTY	Changes the CLKOUT0 Duty Cycle Low Time.	Integer values multiplied by 100,000. For example, a 0.60 duty cycle would be 60,000.
CLKOUT1_DIVIDE	CLKOUT1 output divide value.	1–128; Integer values only.
CLKOUT1_PHASE	CLKOUT1 output phase value.	Integer values multiplied by 1,000. For example, a 45° phase shift would be 45,000.
CLKOUT1_DUTY	Changes the CLKOUT1 Duty Cycle Low Time.	Integer values multiplied by 100,000. For example, a 0.60 duty cycle would be 60,000.
CLKOUT2_DIVIDE	CLKOUT2 output divide value.	1–128; Integer values only.
CLKOUT2_PHASE	CLKOUT2 output phase value.	Integer values multiplied by 1,000. For example, a 45° phase shift would be 45,000.
CLKOUT2_DUTY	Changes the CLKOUT2 Duty Cycle Low Time.	Integer values multiplied by 1,000. For example, a 60/40 duty cycle would be 60,000.
CLKOUT3_DIVIDE	CLKOUT3 output divide value	1–128; Integer values only.

Table 4: PLL Reconfiguration Attributes (Cont'd)

Attribute	Description	Valid Format Values
CLKOUT3_PHASE	CLKOUT3 output phase value	Integer values multiplied by 1,000. For example, a 45° phase shift would be 45,000.
CLKOUT3_DUTY	Changes the CLKOUT3 Duty Cycle Low Time.	Integer values multiplied by 100,000. For example, a 0.60 duty cycle would be 60,000.
CLKOUT4_DIVIDE	CLKOUT4 output divide value.	1-128; Integer values only.
CLKOUT4_PHASE	CLKOUT4 output phase value.	Integer values multiplied by 1,000. For example, a 45° phase shift would be 45,000.
CLKOUT4_DUTY	Changes the CLKOUT4 Duty Cycle Low Time.	Integer values multiplied by 100,000. For example, a 0.60 duty cycle would be 60,000.
CLKOUT5_DIVIDE	CLKOUT5 output divide value.	1-128; Integer values only.
CLKOUT5_PHASE	CLKOUT5 output phase value.	Integer values multiplied by 1,000. For example, a 45° phase shift would be 45,000.
CLKOUT5_DUTY	Changes the CLKOUT5 Duty Cycle Low Time.	Integer values multiplied by 100,000. For example, a 0.60 duty cycle would be 60,000.

Using the Reference Design

Design Functionality

The `p11_drp.v` file has been written with two available reconfigurable states. They are denoted with an `S1_` or `S2_` before each of the attributes in Table 4. You can modify the parameters within each state independently. Additional states can be added or register writes removed as covered in the [Design Modification](#) section.

To change between the two states, first wait for `SRDY` to be asserted. When `SRDY` has been asserted, the state machine is ready to begin reconfiguration. The `SADDR` port specifies which state is loaded into the PLL using the `DRP` port. In an unmodified design, a 0 loads state 1 and a 1 loads state 2. Pulsing `SEN` for one clock cycle triggers the reconfiguration and loads all attributes set in the PLL `DRP` design. Once the reconfiguration is complete, the `SRDY` port is asserted and the PLL is in its newly reconfigured state.

Design Modification

The reference design is intended to be modified to suit the specific needs of a design in a limited fashion. The process of doing these changes is left to the user, but there is one common need that warrants some general instructions on the modification process. It should be noted that the header file `p11_drp_func.h` should *not* be changed. The file `p11_drp.v` is the primary file where design-specific modifications should be done. To perform design modifications, it is expected that the user has become intimately familiar with the functionality of the reconfiguration interface in `p11_drp.v` by reading through the provided source.

The only recommended design modification is to potentially add an additional state to the reference design. To do this, everything that contains an `S#_` (where # is a number) must be replicated to create an `S3_` or higher set of parameters, constant function calls, and ROM initializations. The `SADDR` port must be updated to be a vector allowing the additional state to be addressed, and the `WAIT_SEN` state must be updated to include the ability to set the initial ROM reconfiguration address based on `SADDR`.

Design Verification

The reference design was verified in hardware and with simulation. This ensures that the simulation models and the hardware functionality are equivalent. The verification process chose a number of corner cases for reconfiguration along with some standard configurations to verify that the calculations worked across each scenario. The functions that calculate the various bit settings have also gone through a complete analysis to ensure they match the calculations performed by the ISE® software backend tools during implementation.

Conclusion

This application note and reference design provide a complete implementation of the PLL DRP functionality. Due to its modular nature, the design can be used as a full solution for DRP or can be easily extended to support additional reconfiguration states. The design also uses minimal Spartan-6 FPGA resources, consuming only 25 slices.

Reference Design Additional Information

Files

The reference design files can be downloaded from:

<https://secure.xilinx.com/webreg/clickthrough.do?cid=143937>

Characteristics

The reference design characteristics are summarized in [Table 5](#).

Table 5: Reference Design Matrix

Parameter	Description
General:	
Developer Name	Karl Kurbjun and Carl Ribbing
Target devices	Spartan-6 Family
Source code provided	Yes
Source code format	Verilog
Design uses code/IP from existing Xilinx application note/reference designs, CORE Generator™ software, or 3rd party	No
Simulation:	
Functional simulation performed	Yes
Timing simulation performed	Yes
Testbench used for functional and timing simulations	Yes
Testbench format	Verilog
Simulator software/ version used	ModelSim 6.4b
Implementation:	
Synthesis software tools/version used	XST 12.1
Implementation software tools /versions used	ISE Design Suite 12.1
Static timing analysis performed	Yes
Hardware Verification:	
Hardware verified	Yes
Hardware platform used for verification	Spartan-6 FPGA Characterization Board

Device Utilization and Performance

The reference design device utilization and performance are summarized in [Table 6](#).

Table 6: Device Utilization and Performance for PLL_DRP

Parameters	Specification/Details	
Maximum Frequency (by speed grade)	-2	200 MHz
	-3	200 MHz
Device Utilization without Testbench	Slices	25
	GCLK Buffers	0
	Block RAM	0
HDL Language Support	Verilog	

Appendix

Table 7 identifies the address map for the Spartan-6 FPGA PLL DRP.

Table 7: Address Map for the Spartan-6 PLL DRP

Register Address	Bit	Register Name
0x05	15	CLK0 Delay Time [5]
	14	RESERVED
	13	CLK0 Delay Time [4]
	12	RESERVED
	[11:10]	CLK0 Delay Time [2:3]
	[9:8]	CLK0 Delay Time [1:0]
	[7:0]	RESERVED
0x06	[15:14]	CLK1 Low Time [4:5]
	13	CLK1 Low Time [3]
	12	CLK1 No Count
	[11:10]	CLK1 Low Time [1:2]
	9	CLK1 Delay Time [5]
	8	RESERVED
	[7:6]	CLK1 Delay Time [3:2]
	[5:4]	CLK1 Delay Time [0:1]
	3	RESERVED
	2	CLK0 Edge
	[1:0]	RESERVED
0x07	[15:13]	RESERVED
	12	CLK1 High Time [5]
	[11:10]	CLK1 High Time [3:4]
	[9:7]	CLK1 High Time [2:0]
	6	CLK1 Phase MUX[0]
	5	RESERVED
	4	CLK0 Edge
	[3:2]	RESERVED
	[1:0]	CLK1 Phase MUX[1:2]
0x08	15	CLK2 Phase MUX [2]
	14	RESERVED
	13	CLK2 Low Time [5]
	12	CLK2 Phase MUX [1]
	11	CLK2 No Count
	[10:8]	CLK2 Low Time [4:2]
	7	CLK2 Low Time [0]
	6	CLK2 Delay Time [5]
	[5:4]	CLK2 Delay Time [3:4]
	[3:2]	CLK2 Delay Time [1:2]
	1	CLK2 Delay Time [0]
	0	RESERVED
0x09	[15:14]	CLK3 Delay Time [0:1]
	[13:12]	CLK0 Phase MUX [1:2]
	[11:10]	RESERVED
	9	CLK2 High Time [4]
	8	RESERVED
	[7:6]	CLK2 High Time [3:2]
	[5:4]	CLK2 High Time [0:1]
	3	CLK2 Edge
	2	CLK2 Phase MUX [0]
	[1:0]	RESERVED

Table 7: Address Map for the Spartan-6 PLL DRP (Cont'd)

Register Address	Bit	Register Name
0x0A	15	RESERVED
	14	CLK3 Edge
	[13:12]	RESERVED
	[11:10]	CLK3 Phase MUX [1:2]
	[9:8]	CLK3 Low Time [5:4]
	7	CLK3 No Count
	6	CLK3 Low Time [2]
	[5:4]	CLK3 Low Time [0:1]
	[3:2]	CLK3 Delay Time [4:5]
	1	CLK3 Delay Time [3]
	0	RESERVED
0x0B	15	CLK0 Low Time [5]
	14	CLK4 Delay Time [5]
	13	CLK4 Delay Time [0]
	12	CLK4 Delay Time [3]
	[11:10]	CLK4 Delay Time [1:2]
	9	CLK0 Low Time [4]
	8	RESERVED
	[7:5]	CLK3 High Time [5:3]
	4	RESERVED
	[3:2]	CLK3 High Time [1:2]
	1	CLK3 Phase MUX [0]
	0	CLK3 High Time [0]
0x0C	[15:14]	CLK4 High Time [1:2]
	13	CLK4 Phase MUX [0]
	12	CLK4 High Time [0]
	11	RESERVED
	10	CLK4 Edge
	[9:8]	RESERVED
	[7:6]	CLK4 Phase MUX [2:1]
	[5:4]	CLK4 Low Time [4:5]
	3	CLK4 Low Time [3]
	2	CLK4 No Count
	[1:0]	CLK4 Low Time [1:2]
0x0D	[15:14]	CLK5 Low Time [2:3]
	[13:12]	CLK5 Low Time [0:1]
	[11:10]	CLK5 Delay Time [4:5]
	[9:7]	CLK5 Delay Time [3:1]
	6	CLK0 Low Time [3]
	5	CLK0 Low Time [0]
	4	CLK0 Low Time [2]
	3	RESERVED
	2	CLK4 High Time [5]
	[1:0]	CLK4 High Time [3:4]

Table 7: Address Map for the Spartan-6 PLL DRP (Cont'd)

Register Address	Bit	Register Name
0x0E	[15:14]	CLK5 High Time [4:5]
	[13:12]	CLK5 High Time [2:3]
	[11:10]	CLK5 High Time [0:1]
	9	CLK5 Phase MUX [0]
	8	CLK5 Edge
	[7:6]	RESERVED
	5	CLK5 Phase MUX [2]
	4	RESERVED
	3	CLK5 Low Time [5]
	2	CLK5 Phase MUX [1]
	1	CLK5 No Count
	0	CLK5 Low Time [4]
0x0F	[15:14]	CLKFB Low Time [4:5]
	13	CLKFB Low Time [3]
	12	CLKFB No Count
	[11:10]	CLKFB Low Time [1:2]
	9	CLKFB Low Time [0]
	[8:6]	CLKFB Delay Time [5:3]
	[5:4]	CLKFB Delay Time [1:2]
	3	CLK0 No Count
	2	CLK0 Low Time [1]
	[1:0]	RESERVED
0x10	15	RESERVED
	14	CLK0 High Time [3]
	[13:12]	CLK0 High Time [5:4]
	[11:10]	CLKFB High Time [4:5]
	[9:6]	CLKFB High Time [3:0]
	5	CLKFB Edge
	4	CLKFB Phase MUX [0]
	[3:2]	RESERVED
	[1:0]	CLKFB Phase MUX [1:2]
0x11	[15:10]	RESERVED
	9	CLK3 Low Time [3]
	8	CLK3 Delay Time [2]
	7	CLK2 High Time [5]
	6	CLK2 Low Time [1]
	5	CLK1 Delay Time [4]
	4	CLK1 Low Time [0]
	3	CLK0 High Time [0]
	2	CLK0 Phase MUX [0]
	[1:0]	CLK0 High Time [2:1]
0x12	[15:12]	RESERVED
	11	CLK5 Delay Time [0]
	10	CLKFB Delay Time [0]
	9	CLK4 Low Time [0]
	8	CLK4 Delay Time [4]
	[7:0]	RESERVED

Table 7: Address Map for the Spartan-6 PLL DRP (Cont'd)

Register Address	Bit	Register Name
0x13	15	DIVCLK High Time [5]
	14	RESERVED
	13	DIVCLK High Time [4]
	12	RESERVED
	[11:10]	DIVCLK High Time [1:2]
	9	DIVCLK Low Time [0]
	8	DIVCLK Delay Time [1]
	7	DIVCLK Low Time [5]
	6	DIVCLK Low Time [2]
	5	RESERVED
	4	DIVCLK Edge
	[3:0]	RESERVED
0x14	[15:14]	LKTABLE[1:2]
	13	RESERVED
	12	LKTABLE[0]
	[11:9]	RESERVED
	[8:7]	FILTER TABLE [6:7]
	[6:0]	RESERVED
0x15	15	RESERVED
	14	DIVCLK No Count
	[13:4]	RESERVED
	3	LKTABLE[38]
	2	RESERVED
	1	LKTABLE[32]
	0	LKTABLE[39]
0x16	15	LKTABLE[15]
	14	LKTABLE[13]
	13	LKTABLE[27]
	12	LKTABLE[16]
	11	RESERVED
	10	LKTABLE[10]
	9	RESERVED
	8	DIVCLK High Time [3]
	7	DIVCLK Low Time [1]
	6	RESERVED
	5	DIVCLK High Time [0]
	4	RESERVED
	3	DIVCLK Low Time [3]
	2	RESERVED
	1	DIVCLK Low Time [4]
	0	RESERVED
0x17	[15:6]	RESERVED
	5	LKTABLE[17]
	4	RESERVED
	[3:2]	LKTABLE[8:9]
	[1:0]	LKTABLE[23:22]

Table 7: Address Map for the Spartan-6 PLL DRP (Cont'd)

Register Address	Bit	Register Name
0x18	[15:14]	FILTER TABLE [6:7]
	13	FILTER TABLE [0]
	12	RESERVED
	[11:10]	FILTER TABLE [2:1]
	9	FILTER TABLE [3]
	[8:7]	FILTER TABLE [9:8]
	6	LKTABLE [26]
	[5:3]	RESERVED
	[2:1]	LKTABLE [19:18]
	0	RESERVED
0x19	[15:14]	LKTABLE [24:25]
	13	LKTABLE [21]
	12	LKTABLE [14]
	[11:10]	LKTABLE [11:12]
	9	LKTABLE [20]
	8	LKTABLE [6]
	[7:5]	LKTABLE [35:37]
	4	LKTABLE [3]
	3	LKTABLE [33]
	2	LKTABLE [31]
	1	LKTABLE [34]
	0	LKTABLE [30]
0x1A	[15:2]	RESERVED
	[1:0]	LKTABLE [28:29]
0x1D	15	LKTABLE[7]
	14	LKTABLE[4]
	13	RESERVED
	12	LKTABLE[5]
	[11:0]	RESERVED

Revision History

The following table shows the revision history for this document:

Date	Version	Description of Revisions
05/13/10	1.0	Initial Xilinx release.

Notice of Disclaimer

Xilinx is disclosing this Application Note to you "AS-IS" with no warranty of any kind. This Application Note is one possible implementation of this feature, application, or standard, and is subject to change without further notice from Xilinx. You are responsible for obtaining any rights you may require in connection with your use or implementation of this Application Note. XILINX MAKES NO REPRESENTATIONS OR WARRANTIES, WHETHER EXPRESS OR IMPLIED, STATUTORY OR OTHERWISE, INCLUDING, WITHOUT LIMITATION, IMPLIED WARRANTIES OF MERCHANTABILITY, NONINFRINGEMENT, OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT WILL XILINX BE LIABLE FOR ANY LOSS OF DATA, LOST PROFITS, OR FOR ANY SPECIAL, INCIDENTAL, CONSEQUENTIAL, OR INDIRECT DAMAGES ARISING FROM YOUR USE OF THIS APPLICATION NOTE.