Understanding FLEX 8000 Timing

January 1998, ver. 2 Application Note 76

Introduction

Altera® devices provide predictable performance that is consistent from simulation to application. Before configuring a device, you can determine the worst-case timing delays for any design. You can calculate propagation delays either with the MAX+PLUS® II Timing Analyzer or with the timing models given in this application note and the timing parameters listed in the *FLEX 8000 Programmable Logic Device Family Data Sheet* in this data book.

For the most precise timing results, you should use the MAX+PLUS II Timing Analyzer, which accounts for the effects of secondary factors such as placement and fan-out.

This application note defines device internal and external timing parameters, and illustrates the timing model for the FLEX® 8000 device family.

Familiarity with FLEX 8000 architecture and characteristics is assumed. Refer to the *FLEX 8000 Programmable Logic Device Family Data Sheet* for a complete description of the FLEX 8000 architecture and for specific timing parameter values.

Internal Timing Parameters

The timing delays contributed by individual FLEX 8000 architectural elements are called internal timing parameters, which cannot be measured explicitly. All internal timing parameters are shown in italic type. The following list defines the internal timing parameters for the FLEX 8000 device family.

I/O Element Timing Parameters

t_{IN} — I/O input pad and buffer delay. The time required for a signal on an I/O pin, used as an input, to reach a row or column channel of the FastTrack™ Interconnect.

t_{IOD} — Output data delay. The delay incurred by a signal routed from the FastTrack Interconnect to an I/O element (IOE).

t_{IOC} — IOE control delay. The delay for a signal used to control the I/O register's clock or clear inputs.

t_{IOE}	IOE output enable delay. The delay for a signal used to control the output enable of the IOE's tri-state buffer.
t_{IOCO}	I/O register clock-to-output delay. The delay from the rising edge of the I/O register's clock to the time the data appears at the register output.
t_{IOCOMB}	I/O register bypass delay. The delay for a combinatorial signal to bypass the I/O register.
t_{IOSU}	I/O register setup time. The time required for a signal to be stable at the I/O register's data input before the register clock's rising edge to ensure that the register correctly stores the input data.
t_{IOH}	I/O register hold time. The time required for a signal to be stable at the I/O register's data input after the register clock's rising edge to ensure that the register correctly stores the input data.
t_{IOCLR}	I/O register clear delay. The delay from the time the I/O register's asynchronous clear input is asserted to the time the register output stabilizes at a logic low.
t_{OD1}	Output buffer and pad delay with the slow slew rate logic option turned off and V_{CCIO} = 5.0 V.
t_{OD2}	Output buffer and pad delay with the slow slew rate logic option turned off and V_{CCIO} = 3.3 V.
t_{OD3}	Output buffer and pad delay with the slow slew rate logic option turned on and V_{CCIO} = 5.0 V or 3.3 V.
t_{XZ}	Output buffer disable delay. The delay required for high impedance to appear at the output pin after the tri-state buffer's enable control is disabled.
t_{ZX1}	Output buffer enable delay with the slow slew rate logic option turned off and V_{CCIO} = 5.0 V. The delay required for the output signal to appear at the output pin after the tri-state buffer's enable control is enabled.
t_{ZX2}	Output buffer enable delay with the slow slew rate logic option turned off and V_{CCIO} = 3.3 V. The delay required for the output signal to appear at the output pin after the tri-state buffer's enable control is enabled.

t_{ZX3} Output buffer enable delay with the slow slew rate logic option turned on and V_{CCIO} = 5.0 V or 3.3 V. The delay required for the output signal to appear at the output pin after the tri-state buffer's enable control is enabled.

Interconnect Timing Parameters

t_{DIN_D} Dedicated input data delay. The time required for a signal, used as a data input, to reach a logic element (LE) from a dedicated input pin. The t_{DIN_D} delay is a function of fan-out and the distance between the source pin and destination LEs. The value shown in the *FLEX 8000 Programmable Logic Device Family Data Sheet* is the longest delay possible for a pin with a fan-out of four LEs. However, the value generated by the MAX+PLUS II Timing Analyzer is more accurate because it includes considerations of the fan-out and the relative locations of the source pin and destination LEs of the design.

t_{DIN_C} Dedicated input control delay. The delay of a signal coming from a dedicated input pin that is used as an LE register control. These signals include the clock, clear, and preset inputs to the LE register.

t_{DIN_IO} Dedicated input I/O control delay. The delay of a signal from a dedicated input pin that is used as an IOE register control. These signals include the clock and clear inputs to the IOE register and the output enable control of the IOE's tri-state buffer.

t_{COL} FastTrack Interconnect column delay. The delay incurred by a signal that requires routing through a column channel in the FastTrack Interconnect.

t_{ROW} FastTrack Interconnect row delay. The delay incurred by a signal that requires routing through a row channel in the FastTrack Interconnect. The t_{ROW} delay is a function of fan-out and the distance between the source and destination LEs. The value shown in the *FLEX 8000 Programmable Logic Device Family Data Sheet* is the longest delay possible for an LE with a fan-out of four LEs. However, the value generated by the MAX+PLUS II Timing Analyzer is more accurate because it includes considerations of the fan-out and the relative locations of the source and destination LEs of the design.

t_{LOCAL} Local interconnect delay. The delay incurred by a signal entering a logic array block (LAB) or routed between LEs in the same LAB.

$t_{LABCARRY}$ Carry chain delay to the next LAB. The delay incurred by a carry-out signal that carries into the next LAB in the row.

$t_{LABCASC}$ Cascade chain delay to the next LAB. The delay incurred by a cascade-out signal that cascades into the next LAB in the row.

Logic Element Timing Parameters

t_{LUT} Look-up table (LUT) delay. The delay incurred by generating an LUT output from an LAB local interconnect signal.

t_{RLUT} LUT using LE feedback delay. The time required for the output of an LE register to be fed back and used to generate the LUT output in the same LE. This parameter is used in one of the LE counter modes.

t_{CLUT} Carry chain LUT delay. The delay incurred by a carry chain signal that is used to generate the LUT output.

t_{CGEN} Carry-out generation delay. The delay incurred by generating a carry-out signal from an LAB local interconnect signal.

t_{CGENR} Carry-out generation using LE feedback delay. The time required for the output of an LE register to be fed back and used to generate the carry-out signal in the same LE.

t_{CICO} Carry-in to carry-out delay. The delay incurred by generating a carry-out signal that uses the carry-in signal from the previous LE.

t_{C} Register control delay. The time required for a signal to be routed to the clock, preset, or clear input of an LE register.

t_{GATE} Cascade gate delay. The time required for a signal to pass through the cascade-generating AND gate in the LE. This delay is incurred whether or not the cascade output is used.

t_{CASC}	Cascade chain delay. The time required for a cascade-out signal to be routed to the next LE in the same LAB. This delay, along with $t_{LABCASC}$, is also used to calculate the delay for a cascade-out signal to be routed to an LE in the next LAB in the row.
t_{CO}	LE clock-to-output delay. The delay from the rising edge of the LE register's clock to the time the data appears at the register output.
t_{COMB}	Combinatorial output delay. The time required for a combinatorial signal to bypass the LE register and become the output of the LE.
t_{SU}	LE register setup time. The minimum time a signal is required to be stable at the LE register's data input before the register clock's rising edge to ensure that the register correctly stores the input data.
t_{H}	LE register hold time. The minimum time a signal is required to be stable at the LE register's data input after the register clock's rising edge to ensure that the register correctly stores the input data.
t_{PRE}	LE register preset delay. The delay from the assertion of the LE register's asynchronous preset input to the time the register output stabilizes at a logic high.
t_{CLR}	LE register clear delay. The delay from the assertion of the LE register's asynchronous clear input to the time the register output stabilizes at a logic low.
t_{CH}	Minimum LE register clock-high time. The minimum time the LE register's clock input must remain at a stable logic high state before the falling edge of the clock.
t_{CL}	Minimum LE register clock-low time. The minimum time the LE register's clock input must remain at a stable logic low state before the rising edge of the clock.

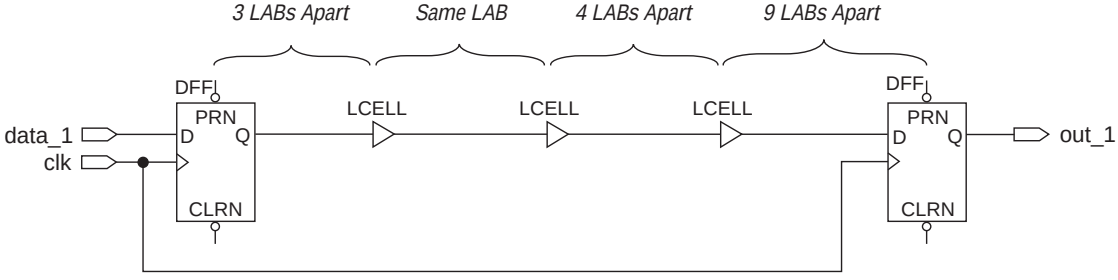
External Timing Parameters

External timing parameters represent actual pin-to-pin timing characteristics. Each external timing parameter consists of a combination of internal delay elements. They are worst-case values, derived from extensive performance measurements and are ensured by device testing or characterization. For example, $\mathbf{t_{DRR}}$ is the AC operating specification. All external timing parameters are shown in bold type. Other external timing parameters can be estimated by using the timing model in Figure 2.

t_{DRR} Register-to-register delay. The time required for the signal from one register to pass through four LEs via three row interconnects and four local interconnects to reach the D input of a second register. The test circuit used for this parameter is a register with an output that goes through three LCELL primitives in two different LABs; the last LCELL feeds another register in another LAB. Figure 1 shows this path. The test circuit file is available from Altera Applications.

t_{ODH} Output data hold time after clock. The minimum time a registered output pin will remain at its previous value after a rising edge is applied to the clock input pin. This parameter applies to global and non-global clocking, and for logic element and I/O element registers.

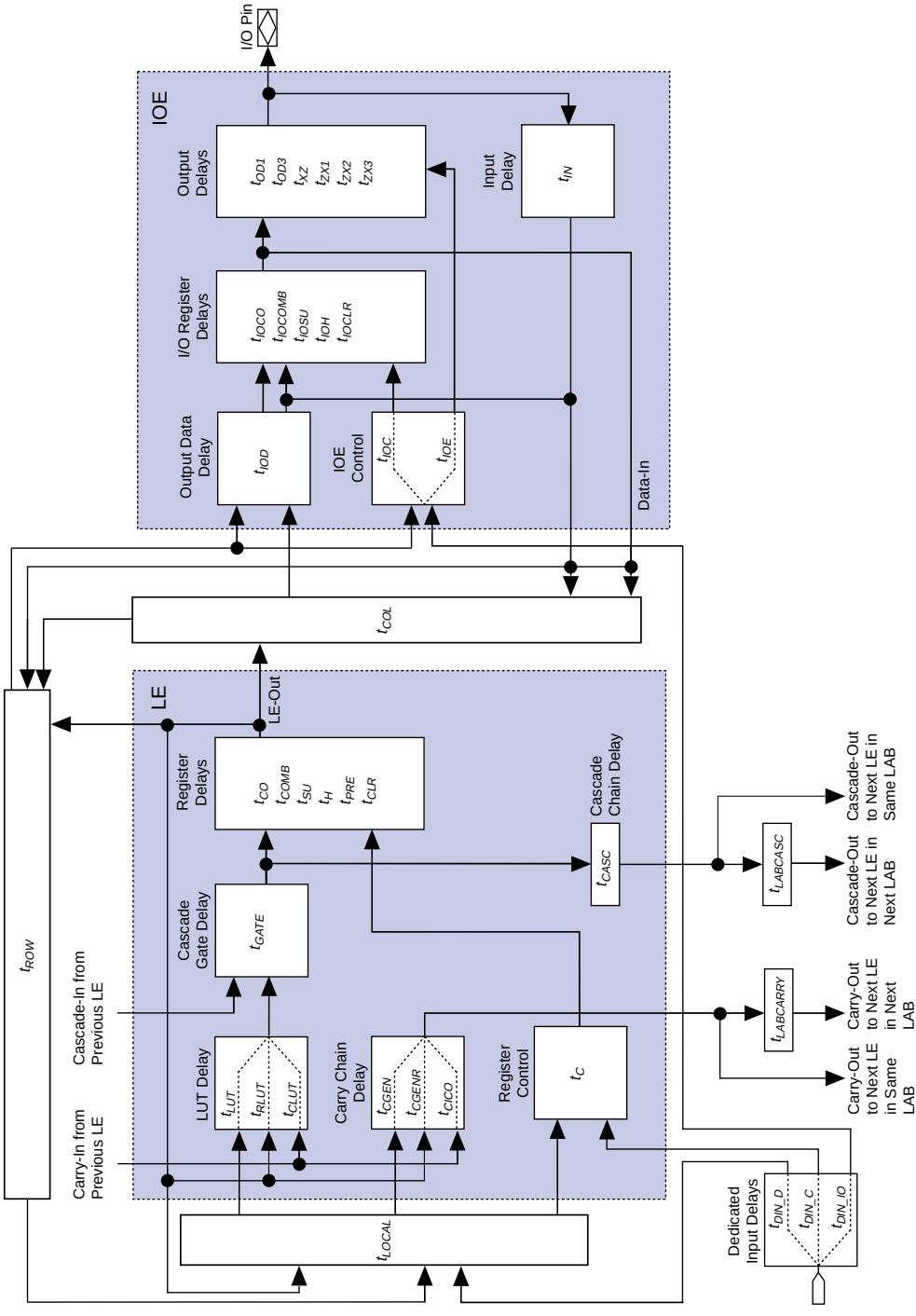
Figure 1. Path for t_{DRR} Circuit for 21-Column FLEX 8000 Devices



FLEX 8000 Timing Model

Timing models are simplified block diagrams that illustrate the propagation delays through Altera devices. Logic can be implemented on different paths. You can trace the actual paths used in your FLEX 8000 device by examining the equations listed in the MAX+PLUS II Report File (**.rpt**) for the project. You can then add up the appropriate internal timing parameters to calculate the approximate propagation delays through the FLEX 8000 device. However, the MAX+PLUS II Timing Analyzer provides the most accurate timing information. Figure 2 shows the timing model for FLEX 8000 devices.

Figure 2. FLEX 8000 Timing Model



Calculating Timing Delays

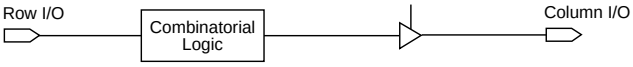
You can calculate approximate pin-to-pin timing delays for FLEX 8000 devices with the timing model shown in Figure 2 and the internal timing parameters in the *FLEX 8000 Programmable Logic Device Family Data Sheet* in this data book. Each timing delay is calculated from a combination of internal timing parameters. Figure 3 shows the FLEX 8000 device family LE timing delays. To calculate the delay for a signal that follows a different path through the FLEX 8000 device, refer to the FLEX 8000 timing model to determine which internal timing delays to add together.

Figure 3. Logic Element Timing Delays (Part 1 of 4)

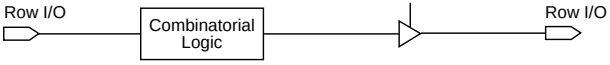
Combinatorial Delay

From Row I/O Inputs:

t_1



t_2

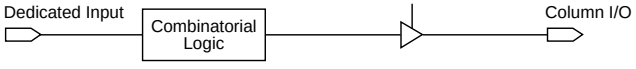


$$t_1 = t_{IN} + t_{ROW} + t_{LOCAL} + t_{LUT} + t_{GATE} + t_{COMB} + t_{COL} + t_{IOD} + t_{IOCOMB} + t_{OD1}$$

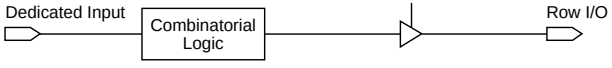
$$t_2 = t_{IN} + t_{ROW} + t_{LOCAL} + t_{LUT} + t_{GATE} + t_{COMB} + t_{ROW} + t_{IOD} + t_{IOCOMB} + t_{OD1}$$

From Dedicated Inputs:

t_1



t_2



$$t_1 = t_{DIN\_D} + t_{LOCAL} + t_{LUT} + t_{GATE} + t_{COMB} + t_{COL} + t_{IOD} + t_{IOCOMB} + t_{OD1}$$

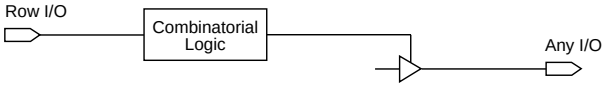
$$t_2 = t_{DIN\_D} + t_{LOCAL} + t_{LUT} + t_{GATE} + t_{COMB} + t_{ROW} + t_{IOD} + t_{IOCOMB} + t_{OD1}$$

Figure 3. Logic Element Timing Delays (Part 2 of 4)

Tri-State Enable/Disable Delay

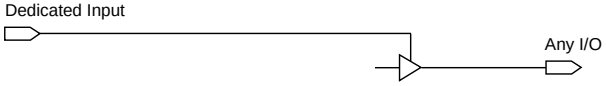
t_{XZ} or **t_{ZX}**

From Row I/O Inputs through logic:



$$t_{XZ}, t_{ZX} = t_{IN} + t_{ROW} + t_{LOCAL} + t_{LUT} + t_{GATE} + t_{COMB} + t_{ROW} + t_{IOE} + (t_{XZ} \text{ or } t_{ZX1})$$

Directly from Dedicated Inputs:



$$t_{XZ}, t_{ZX} = t_{DIN\_IO} + t_{IOE} + (t_{XZ} \text{ or } t_{ZX1})$$

Directly from Row I/O Inputs:

$$t_{XZ}, t_{ZX} = t_{IN} + t_{ROW} + t_{IOE} + (t_{XZ} \text{ or } t_{ZX1})$$

Figure 3. Logic Element Timing Delays (Part 3 of 4)

LE Register Clear & Preset Time

From Row I/O Inputs to Row or Column Outputs:

t_{CLR}

t_{PRE}

$$t_{CLR} = t_{IN} + t_{ROW} + t_{LOCAL} + t_C + t_{CLR} + (t_{ROW} \text{ or } t_{COL}) + t_{IOD} + t_{IOCOMB} + t_{OD1}$$

$$t_{PRE} = t_{IN} + t_{ROW} + t_{LOCAL} + t_C + t_{PRE} + (t_{ROW} \text{ or } t_{COL}) + t_{IOD} + t_{IOCOMB} + t_{OD1}$$

From Dedicated Inputs to Row or Column Outputs:

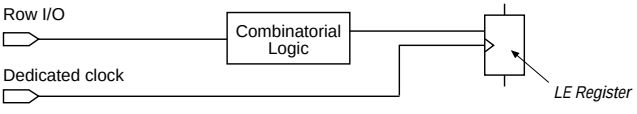
t_{CLR}

t_{PRE}

$$t_{CLR} = t_{DIN\_C} + t_C + t_{CLR} + (t_{ROW} \text{ or } t_{COL}) + t_{IOD} + t_{IOCOMB} + t_{OD1}$$

$$t_{PRE} = t_{DIN\_C} + t_C + t_{PRE} + (t_{ROW} \text{ or } t_{COL}) + t_{IOD} + t_{IOCOMB} + t_{OD1}$$

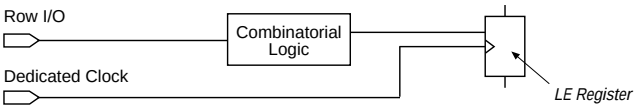
Setup Time from a Global Clock & Row I/O Data Input



$$t_{SU} = (t_{IN} + t_{ROW} + t_{LOCAL} + t_{LUT} + t_{GATE}) - (t_{DIN\_C} + t_C) + t_{SU}$$

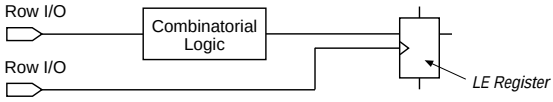
Figure 3. Logic Element Timing Delays (Part 4 of 4)

Hold Time from a Global Clock & Row I/O Data Input



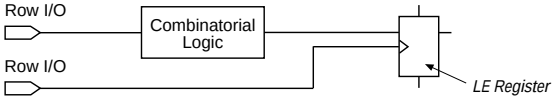
$$t_H = (t_{DIN\_C} + t_C) - (t_{IN} + t_{ROW} + t_{LOCAL} + t_{LUT} + t_{GATE}) + t_H$$

Asynchronous Setup Time from a Row I/O Clock & Row I/O Data Input



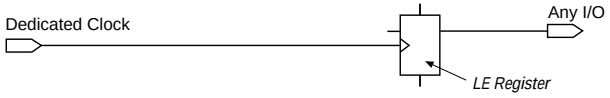
$$t_{ASU} = (t_{IN} + t_{ROW} + t_{LOCAL} + t_{LUT} + t_{GATE}) - (t_{IN} + t_{ROW} + t_{LOCAL} + t_C) + t_{SU}$$

Asynchronous Hold Time from a Row I/O Clock & Row I/O Data Input



$$t_{AH} = (t_{IN} + t_{ROW} + t_{LOCAL} + t_C) - (t_{IN} + t_{ROW} + t_{LOCAL} + t_{LUT} + t_{GATE}) + t_H$$

Clock-to-Output Delay from a Global Clock to Any Output



$$t_{CO} = t_{DIN\_C} + t_C + t_{CO} + (t_{ROW} \text{ or } t_{COL}) + t_{IOD} + t_{IOCOMB} + t_{OD1}$$

Clock-to-Output Delay from a Row I/O Clock to Any Output

$$t_{ACO} = t_{IN} + t_{ROW} + t_{LOCAL} + t_C + t_{CO} + (t_{ROW} \text{ or } t_{COL}) + t_{IOD} + t_{IOCOMB} + t_{OD1}$$

Figure 4 shows the FLEX 8000 device family I/O element timing delays. To calculate the delay for a signal that follows a different path through the FLEX 8000 device, refer to the FLEX 8000 timing model to determine which internal timing parameters to add together.

Figure 4. I/O Element Timing Delays (Part 1 of 2)

I/O Element Clear Time

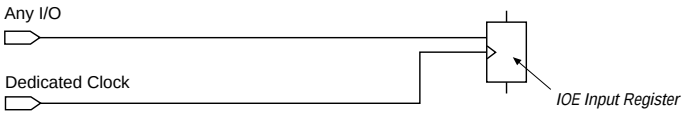
From Row I/O Inputs:

$$t_{CLR} = t_{IN} + t_{ROW} + t_{IOC} + t_{IOCLR} + t_{OD1}$$

From Dedicated Inputs:

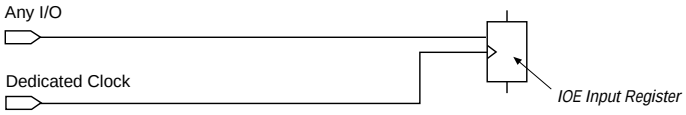
$$t_{CLR} = t_{DIN\_IO} + t_{IOC} + t_{IOCLR} + t_{OD1}$$

Setup Time from a Global Clock & Any I/O Data Input



$$t_{SU} = t_{IN} - (t_{DIN\_IO} + t_{IOC}) + t_{IOSU}$$

Hold Time from a Global Clock & Any I/O Data Input



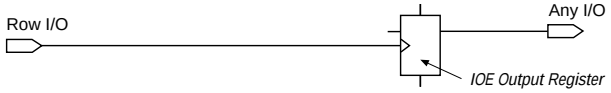
$$t_{H} = (t_{DIN\_IO} + t_{IOC}) - t_{IN} + t_{IOH}$$

Figure 4. I/O Element Timing Delays (Part 2 of 2)

Clock-to-Output Delay from a Global Clock to Any Output

$t_{CO} = t_{DIN\_IO} + t_{IOC} + t_{IOCO} + t_{OD1}$

Clock-to-Output Delay from a Row I/O Clock to Any Output



$t_{ACO} = t_{IN} + t_{ROW} + t_{IOC} + t_{IOCO} + t_{OD1}$

Timing Model vs. MAX+PLUS II Timing Analyzer

Hand calculations based on the timing model provide a good estimate of a design's performance. However, the MAX+PLUS II Timing Analyzer always provides the most accurate information on the performance of a design, because it takes into account three secondary factors that influence the t_{ROW} and t_{DIN_D} parameters:

- Fan-out for each signal in the delay path
- Positions of other loads relative to the source and destination
- Distance between signal source and destination

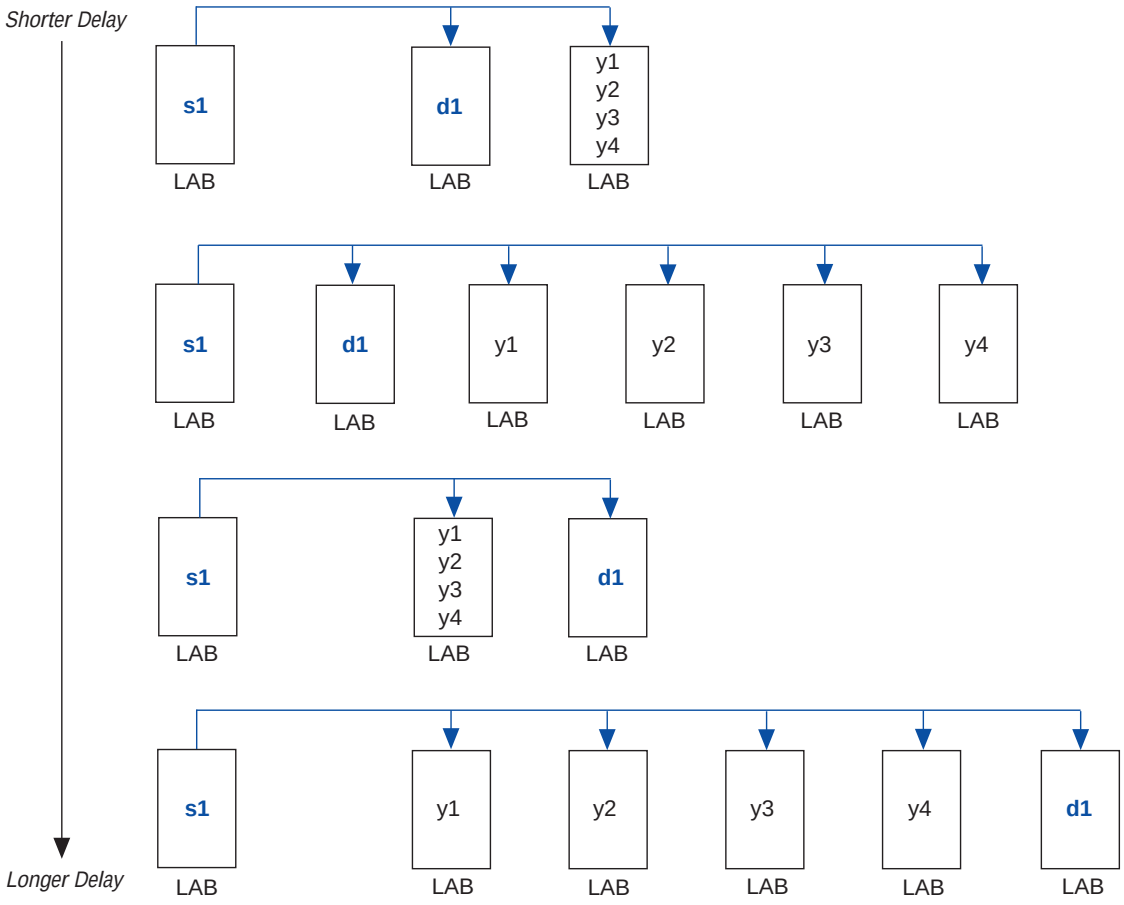
Fan-Out

The more loads a signal has to drive, the longer the delay across t_{ROW} and t_{DIN_D}. These delays are functions of the number of LABs that a signal source has to drive, as well as of the number of LEs in the LAB that use the signal. The number of LABs that a signal drives has a greater effect on the delay than the number of cells in the LAB that use the signal.

Load Distribution

The load distribution relative to the source and destination also affects the t_{ROW} and t_{DIN_D} delays. Consider a signal s1 that feeds destination d1 and logic elements y[4..1]. If y[4..1] are in different LABs, s1 has four additional loads. However, if the LEs are all in the same LAB, s1 has four shorter-delay loads. Therefore, the row interconnect delay from s1 to d1 is greater when each load y[4..1] is in a different LAB. Figure 5 illustrates the change in the t_{ROW} and t_{DIN_D} delays caused by variations in the position of d1 and the distribution of y[4..1].

Figure 5. Delay from s1 to d1 as a Function of Relative Position & Load Distribution



Distance

The distance between the source and destination LEs also affects the t_{ROW} and t_{DIN_D} parameters. For example, if `s1` drives an LE in the same row, the delay from `s1` to the LE increases as the distance from `s1` to the LE increases.

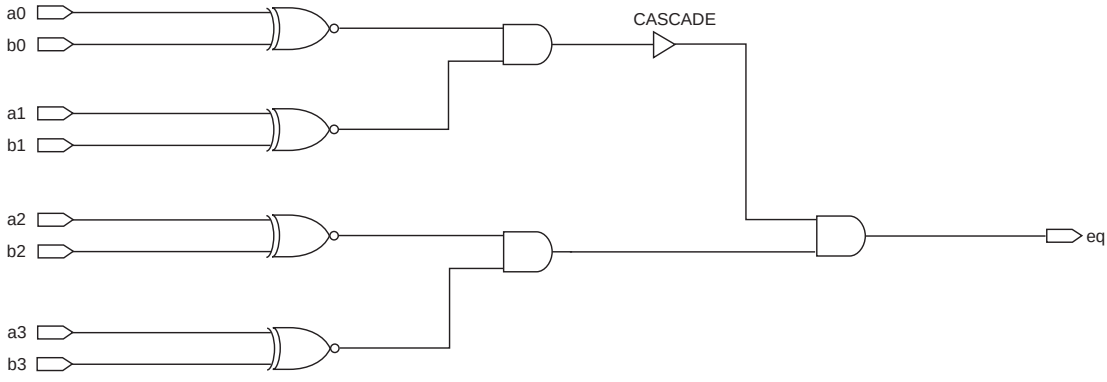
Examples

The following examples show how to use internal timing parameters to estimate the delays for real applications.

Example 1: 4-Bit Equality Comparator with a Cascade Chain

You can analyze the timing delays for circuits that have been subjected to minimization and logic synthesis. The synthesized equations are available in your project's MAX+PLUS II Report File (**.rpt**). These equations are structured so that you can quickly determine the logic implementation of any signal. For example, Figure 6 shows a 4-bit equality comparator.

Figure 6. 4-Bit Equality Comparator Circuit

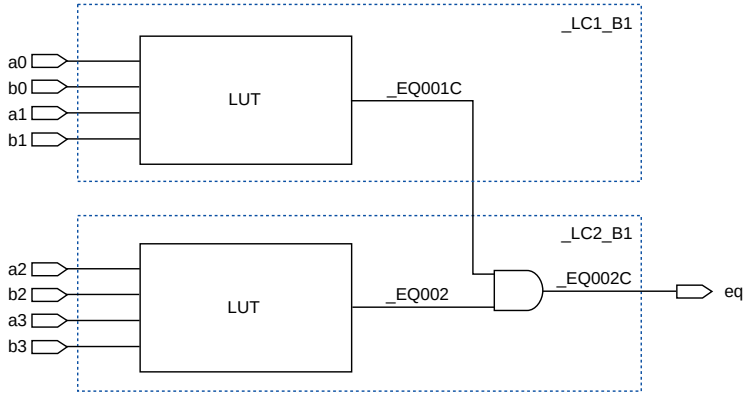


The MAX+PLUS II Report File for the circuit shown in Figure 6 gives the equations for eq, the output of the comparator:

```
eq      =  _LC2_B1;
_LC2_B1 =  LCELL( _EQ002C);
_EQ002C =  _EQ002 & CASCADE( _EQ001C);
_EQ002  =  a2 & a3 & b2 & b3
           # a2 & !a3 & b2 & !b3
           # !a2 & a3 & !b2 & b3
           # !a2 & !a3 & !b2 & !b3;
_LC1_B1 =  LCELL( _EQ001C);
_EQ001C =  _EQ001;
_EQ001  =  a0 & a1 & b0 & b1
           # a0 & !a1 & b0 & !b1
           # !a0 & a1 & !b0 & b1
           # !a0 & !a1 & !b0 & !b1;
```

Figure 7 shows a synthesized 4-bit equality comparator.

Figure 7. Synthesized 4-Bit Equality Comparator



The output pin eq is the output of the second LE of a cascade chain. The LUT of _LC1_B1 implements the comparison of the first two bits. The comparison of the second two bits is implemented in the LUT of _LC2_B1. The outputs of these two LUTs are then cascaded together to form the output of _LC2_B1.

If a2 and eq are both row I/O pins, the timing delay from a2 to eq can be estimated by adding the following parameters:

$$t_{IN} + t_{ROW} + t_{LOCAL} + t_{LUT} + t_{GATE} + t_{COMB} + t_{ROW} + t_{IOD} + t_{IOCOMB} + t_{OD1}$$

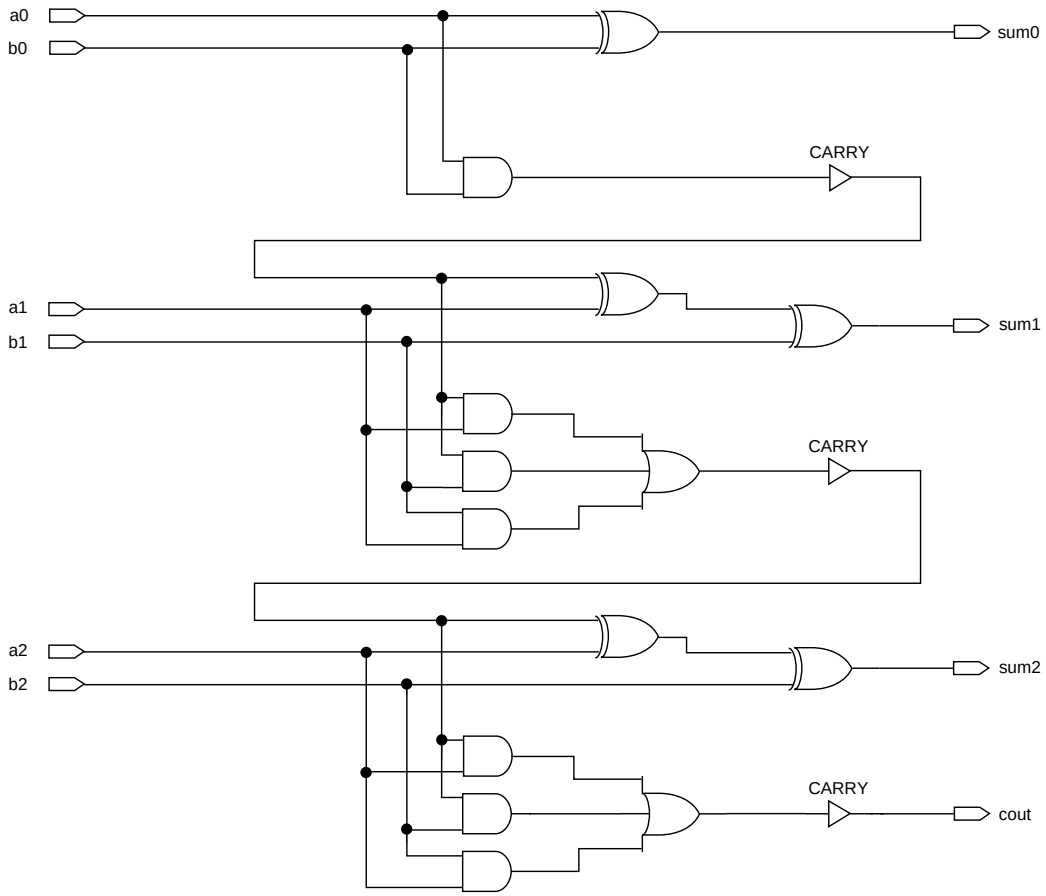
If a0 is a row I/O pin, the timing delay from a0 to eq can be estimated by adding the following parameters:

$t_{IN} + t_{ROW} + t_{LOCAL} + t_{LUT} + t_{GATE} + t_{CASC} + t_{GATE} + t_{COMB} + t_{ROW} + t_{IOD} + t_{IOCOMB} + t_{OD1}$

Example 2: 3-Bit Adder Using a Carry Chain

FLEX 8000 devices have specialized resources that implement complex arithmetic functions. For instance, adders and counters require a carry function to determine whether or not to increment the next significant bit. The FLEX 8000 architecture has a built-in carry chain that performs this function. This example explains how to estimate the delay for a 3-bit adder that uses a carry chain (See Figure 8).

Figure 8. 3-Bit Adder Implemented with a Carry Chain

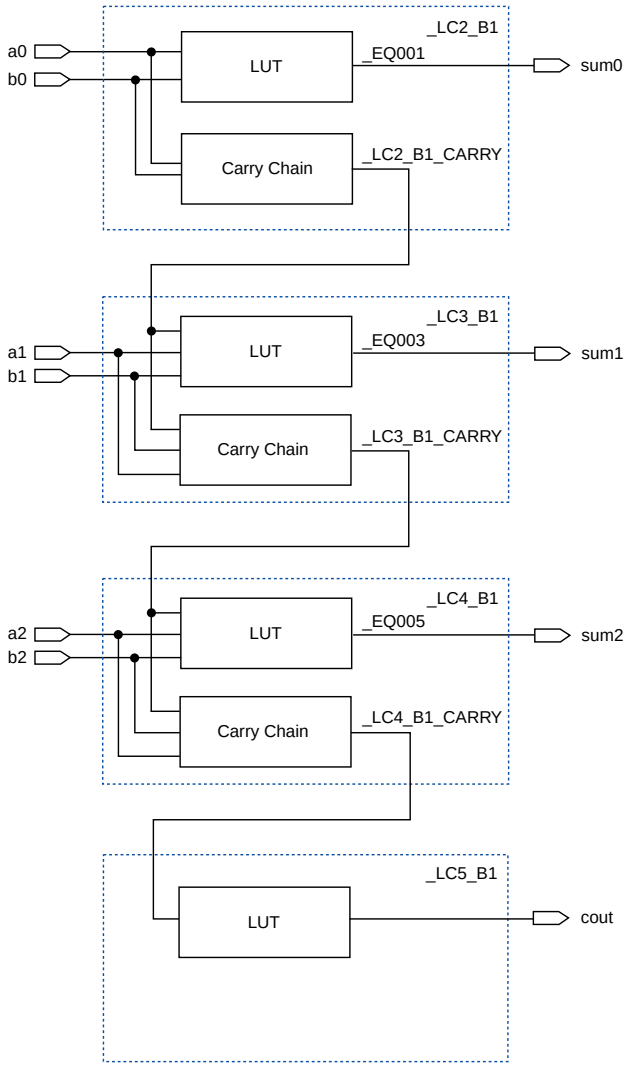


The MAX+PLUS II Report File contains the following equations for the 3-bit adder in Figure 8:

```
cout            = _LC5_B1;
sum0            = _LC2_B1;
sum1            = _LC3_B1;
sum2            = _LC4_B1;
_LC2_B1         = LCELL( _EQ001);
  _EQ001        = !a0 & b0
                  # a0 & !b0;
_LC2_B1_CARRY   = CARRY( _EQ002);
  _EQ002        = a0 & b0;
_LC3_B1         = LCELL( _EQ003);
_EQ003          = a1 & !b1 & !_LC2_B1_CARRY
                  # !a1 & !b1 & _LC2_B1_CARRY
                  # a1 & b1 & _LC2_B1_CARRY
                  # !a1 & b1 & !_LC2_B1_CARRY;
_LC3_B1_CARRY   = CARRY( _EQ004);
_EQ004          = a1 & _LC2_B1_CARRY
                  # a1 & b1
                  # b1 & _LC2_B1_CARRY;
_LC4_B1         = LCELL( _EQ005);
_EQ005          = a2 & !b2 & !_LC3_B1_CARRY
                  # !a2 & !b2 & _LC3_B1_CARRY
                  # a2 & b2 & _LC3_B1_CARRY
                  # !a2 & b2 & !_LC3_B1_CARRY;
_LC5_B1         = LCELL( _LC4_B1_CARRY);
_LC4_B1_CARRY   = CARRY( _EQ006);
_EQ006          = a2 &  _LC3_B1_CARRY
                  # a2 & b2
                  # b2 & _LC3_B1_CARRY;
```

Figure 9 shows a synthesized 3-bit adder.

Figure 9. Synthesized 3-Bit Adder



In Figure 9, LE _LC2_B1 generates sum0 and a carry-out signal (_LC2_B1_CARRY) that feeds the carry-in of _LC3_B1. _LC3_B1 generates sum1 and a carry-out signal (_LC3_B1_CARRY) that feeds the carry-in of _LC4_B1. _LC4_B1 generates sum2 and cout using a2, b2, and _LC3_B1_CARRY. The cout signal must pass through _LC5_B1 because a carry buffer cannot directly feed a pin.

If a0 and sum1 are row I/O pins, the timing delay from a0 to sum1 can be estimated by adding the following microparameters:

$t_{IN} + t_{ROW} + t_{LOCAL} + t_{CGEN} + t_{CICO} + t_{CLUT} + t_{GATE} + t_{COMB} + t_{ROW} + t_{IOD} + t_{IOCOMB} + t_{ODI}$

If a0 and cout are row I/O pins, the timing delay from a0 to cout can be estimated by adding the following microparameters:

$t_{IN} + t_{ROW} + t_{LOCAL} + t_{CGEN} + t_{CICO} + t_{CICO} + t_{CLUT} + t_{GATE} + t_{COMB} + t_{ROW} + t_{IOD} + t_{IOCOMB} + t_{ODI}$

Conclusion

The FLEX 8000 device architecture has predictable internal timing delays that can be estimated based on signal synthesis and placement. The MAX+PLUS II Timing Analyzer provides the most accurate timing information. However, you can also use the FLEX 8000 timing model shown in Figure 2, along with the timing parameters listed in the *FLEX 8000 Programmable Logic Device Family Data Sheet* in this data book, to estimate a design's performance before compilation. Both methods enable you to accurately predict your design's in-system timing performance.