Altera XML Point To Point Delay File
Detailed Design

Version 1.1

May 27, 2005

By

Altera Corporation

Table of Contents

1. Overview

The Altera® XML Point To Point Delay File is a communications standard for describing intra-cell delay values to EDA tool vendors in a concise, accurate and digestible format. This document is intended for external use. It describes in detail the format of the Altera XML Point To Point Delay document and fills in any gaps in the format the XML Schema does not explicitly cover. It is intended as a guide for both the future maintenance of the document specification and for the third party programmer attempting to use the information contained in an XML file that meets the Altera XML Point To Point Delay File specification.

The XML format is a flexible, platform-independent method for sharing information. The Altera XML Point To Point Delay File further constrains this format to provide a flexible and understandable method for sharing Altera architecture delay information. The standard is set forth in an XML Schema and files validated against this schema are said to be valid Altera XML Point To Point Delay Files. A myriad of XML parsers are available for most languages to allow quick and easy digestion of the information in the Altera XML Point To Point Delay File.

2. Executive Summary

This document is to be used in conjunction with the document *Altera_XML_Architecture_Description_Detailed_Design.doc*. This document describes the XML description of the chip itself.

This document has the following purposes:

- Describe the file formats used to describe point-to-point delays in Altera devices
- Explain each part of the file
- Give various XML links

This file does not:

- Give detailed architecture information like block information and layout
- Give interconnect estimator file information
- Give an overview of why this information is relevant

After reading this file and referenced material you should be able to do the following:

- Understand the raw format of an Altera XML Point To Point Delay File

Any input output pairs that are not in the XML file are not valid paths through the block.

3. Glossary

XML Document – A file based on the W3C's eXtensible Markup Language format. All XML documents described in this detailed design document conform to the W3C XML 1.0 standard. For more information on XML see: http://www.xml.com/axml/testaxml.htm

XML Schema – Templates for XML documents; they allow you to specify valid arrangements of elements and attributes within an XML document and impose restrictions on their contents. An XML document that is valid according to an XML schema does not break any of the rules laid down in the XML schema.

Element – An XML container formed with a start tag (<START_TAG>) and an end tag (</END_TAG>). The start tag can contain some number of attributes (<START_TAG att1=val1 att2=val2>) according to the schema for the document. Data that falls between the start and end tags, be it additional elements or text, is said to be contained by the element. This allows for a parent-child tree structure in XML data to be formed. If an element contains no children the start/end tag notation can be abbreviated into a form known as an empty tag (<START_TAG att1=val1 att2=val2 />).

Block – The elements upon which a chip is built. They are hierarchical representations of basic pieces of the architecture. For example: a logic element block, LE, or a LAB block that contains a number of LE blocks.

Architecture – A top-level container for devices. The architecture describes common blocks that are then instantiated in devices of various sizes and configurations.

Device – A group of blocks instantiated in a particular configuration.

4. The Altera XML Point-To-Point Delay Schema

The Altera XML Point-To-Point Delay schema describes the format for any XML-based architecture description. A tree-view of this schema is seen in Figure 4-1. For more information on each specific element in the diagram please see Section 5.

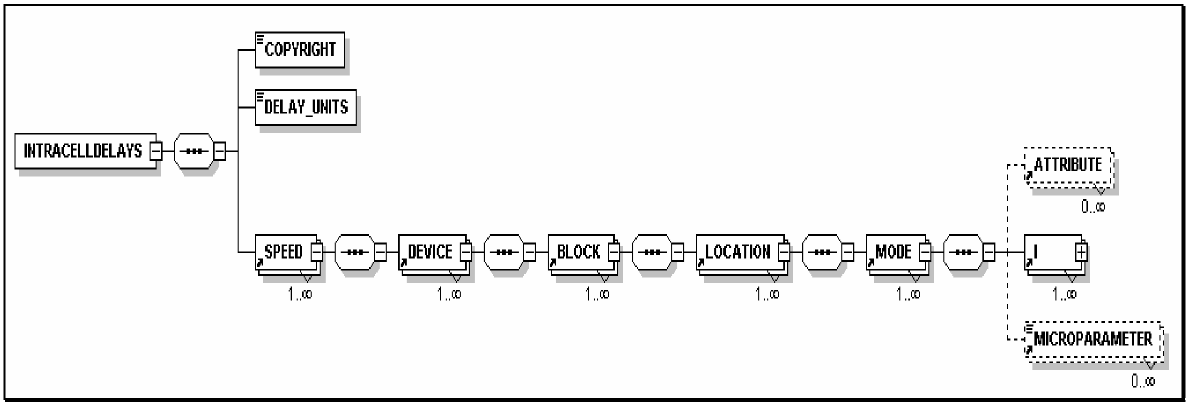


Figure 4-1: Tree View of the Altera XML Point-To-Point Delay Schema

The top-level container, or root element, is the <INTRACELLDELAYS> element which is a container element, used to hold devices built with a common set of blocks together.

The dashed line/boxes in Figure 4-1 represent optional elements. The numbers of times an element can occur are listed directly below the element boxes. Order is important in XML. For example, you can have from one to an infinite number of <ATTRIBUTE> elements inside a <MODE> element, but they must all occur before any <I> element in the <MODE> element. Once you see an <I> element, based on this schema, you can assume you will never see an <ATTRIBUTE> element again for this <MODE>.

Solid line/boxes represent elements that must occur at least once. According to Figure 4-1 an <INTRACELLDELAYS> element must have at least one <SPEED> element, and a <SPEED> element must have at least one <DEVICE> element. The <MODE> elements however can have any number of the <ATTRIBUTE> child elements shown, including zero.

5. Detailed Altera XML Point To Point Delay File Element Information

This section has detailed information for every possible element that can exist, according to the schema, in an Altera XML Point To Point Delay File. It can be used as a reference when creating as well as parsing an XML file that is validated against this schema. A list of elements and their attributes can be found in Table 1. Detailed descriptions of each element can be found in the subsections below or by clicking on the Element tag in the table.

Table 1: Element and attributes found in an Altera XML Point To Point Delay File

Element	Attribute	Type	Use	Description
<INTRACELLDELAYS>	name	string	required	The name of the architecture to which all devices following belong.
<ATTRIBUTE>	Name	string	required	The key in the (key, value) pair.
	Value	string	required	The value in the (key, value) pair.
<BLOCK>	Type	string	required	The major type of block to which this delay information corresponds.
	subtype	string	optional	The subtype of the block. The default value is DEFAULT if this attribute is omitted.
<DELAY_UNITS>	*no attributes*			
<DEVICE>	name	string	required	The name of the device to which this delay information belongs.
<I>	name	string	required	The name of the input.
<LOCATION>	X	integer	optional	The horizontal position of the coordinate.

	Y	integer	optional	The vertical position of the coordinate.
	subloc	integer	optional	The sublocation index of the coordinate.
<MODE>	name	string	optional	The name of the mode to which this delay information belongs.
<O>	name	string	required	The name of the output.
<MICROPARAMETER>	name	string	required	The name of the microparameter.
<SPEED>	grade	string	optional	A speed grade designation for the devices that follow. The default value is "COMMON" if none is supplied.

5.1 <INTRACELLDELAYS>

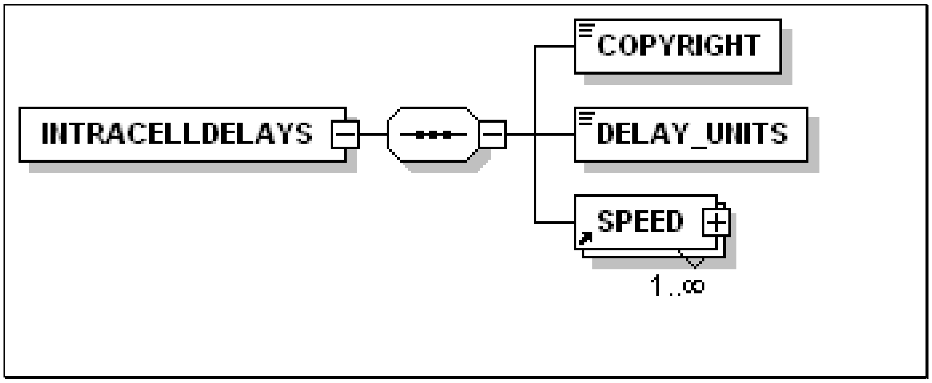


Figure 5-1 The <INTRACELLDELAYS> element

The <INTRACELLDELAYS> element is the root of all Altera XML Point-To-Point Delay files. This is the first element encountered and it is meant to contain common delay information and specific delay information for one or more <DEVICE>s in the architecture family.

Every point-to-point delay file is guaranteed to have at least one <INTRACELLDELAYS> element with at least one <DEVICE> that has as at least one <SPEED> section and one <BLOCK> at one <LOCATION> with one <MODE> that possibly contains point-to-point delay numbers. The <DELAY_UNITS> element specifies the units used for delay time through out the file.

5.2 <ATTRIBUTE>

The <ATTRIBUTE> element allows for simple (name, value) paired information to be expressed. It provides additional information about its parent or containing element that is possibly non-essential, but perhaps useful. An example of some common <ATTRIBUTE> elements for the <MODE> element is:

```
<MODE>
<ATTRIBUTE name=”EMULATE_PACKED_REGISTER” value=”TRUE” />
</MODE>
```

This <ATTRIBUTE> element tells us that the parent <MODE> element is emulating a packed register.

5.3 <BLOCK>

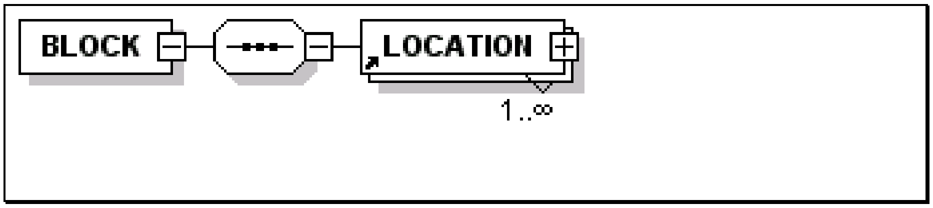


Figure 5-2: The <BLOCK> element

The <BLOCK> element contains all the point-to-point delay information for a block with a specific name and subtype.

5.4 <DELAY_UNITS>

The unit of time for all delay values in the file.

5.5 <DEVICE>

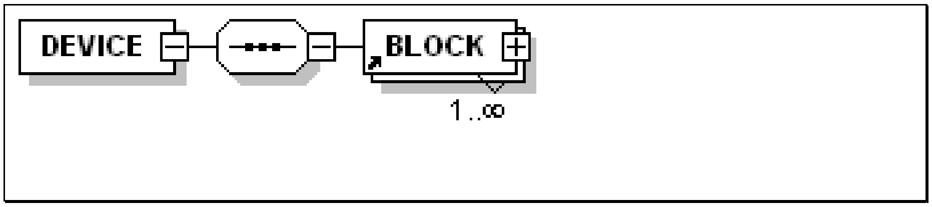


Figure 5-3: The <DEVICE> element

The <DEVICE> element contains all the point-to-point delay information for the blocks in a specific device. Device names do not include the package type, pin counts or speed grades. Delays are broken down below the <DEVICE> element into delays that are common across all devices for a specific <SPEED> grade, and delays that differ from device to device within a <SPEED> grade. Device packaging does not affect point-to-point delay numbers.

5.6 <MICROPARAMETER>

The <MICROPARAMETER> element defines a microparameter delay value in some units of time. There are two types of microparameter delays: general and register. General microparameter delays apply to a block in a particular mode, and register microparameter delay apply to a register within a block in a particular mode. The element has one attribute that tells us the name of the microparameter delay. Time units are always given at the start of the delay file using the <DELAY_UNITS> element.

For example, a register named ff that has a setup time (Tsu) of 30 time units would be annotated as:

```
<I name=”ff”>
<MICROPARAMETER name=”tsu”>30</MICROPARAMETER>
</I>
```

5.7 <I>

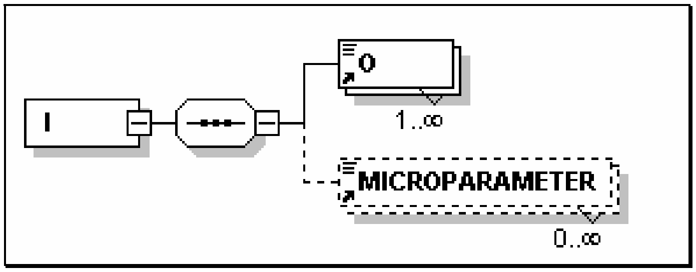


Figure 5-4: The <I> element

The <I> element defines a set of point-to-point delays from this “in” port with a specific name. When the element refers to the output port of a register, it also defines a set of register microparameter delays associated with this register. It is a child of a <MODE> element and only appears in a mode if the input is logically available for the block in the particular mode. It is important to realize that an “in” port does not necessarily mean a port with an input direction in to the block, rather it is the starting port from which you want to obtain a point-to-point delay. The <I> element can contain multiple <O> elements, the ports to which you want the point-to-point delay information from this <I> port.

5.8 <LOCATION>

A simple element, the <LOCATION> has three optional attributes: *x*, *y* and *subloc*. The *x* attribute is an integer value representing a horizontal location. The *y* attribute is an integer value representing a vertical location. The *subloc* attribute represents an indexed location in a 1 x 1 block.

For example: a LAB in a Stratix™ device occupies a 1 x 1 region but contains 10 LE sub-blocks [1]. The LAB has a coordinate with *x* and *y* attributes whereas each LE sub-block has a coordinate with a zero valued *x* and *y* attribute and an indexed *subloc* attribute value in the range [0,9]. No two LABs share the same coordinate, and no two LEs in a LAB share the same *subloc*. This is done to allow sub-block, or clustered block, access without defining a rigid sub-location scheme. The unique *subloc* value is enough to indicate that the LE has a unique, non-overlapping, location at the same coordinate as its containing LAB element.

Location information in the point-to-point delay file is provided *if and only if* the delay values for a block differ depending on its location inside it’s parent block. If the delay values for a block do not differ depending on location, one default location (0,0,0) is written in the point-to-point delay file.

5.9 <MODE>

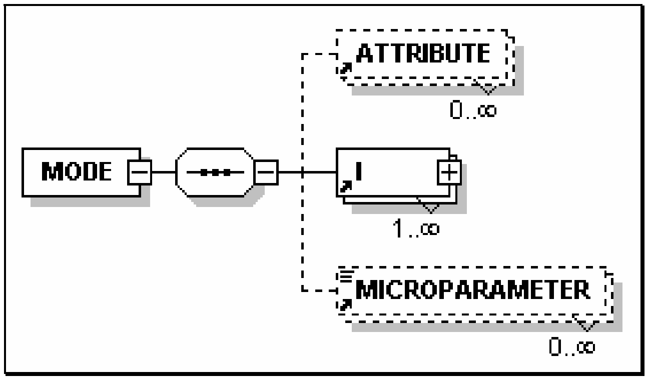


Figure 5-5: The <MODE> element

The <MODE> element contains a set of point-to-point delays for a block when specific block attributes are set. It also contains a set of general microparameter delays that apply to such block. For blocks that have no discernable attributes one <MODE> will be defined that contains no <ATTRIBUTE> elements.

5.10 <O>

The <O> element defines a delay value in some units of time from one port to the port named in this <O> element. The element contains an integer that represents a delay value. Time units are always given at the start of the delay file using the <DELAY_UNITS> element.

For example, a 100 ps delay from a port named dataa to a port named combout would be annotated as:

```
<I name=”dataa”>
<O name=”combout”>100</O>
</I>
```

5.11 <SPEED>

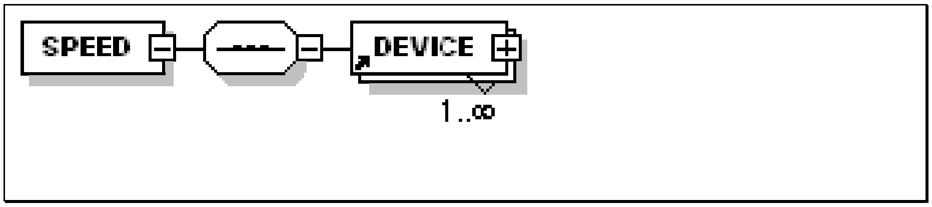


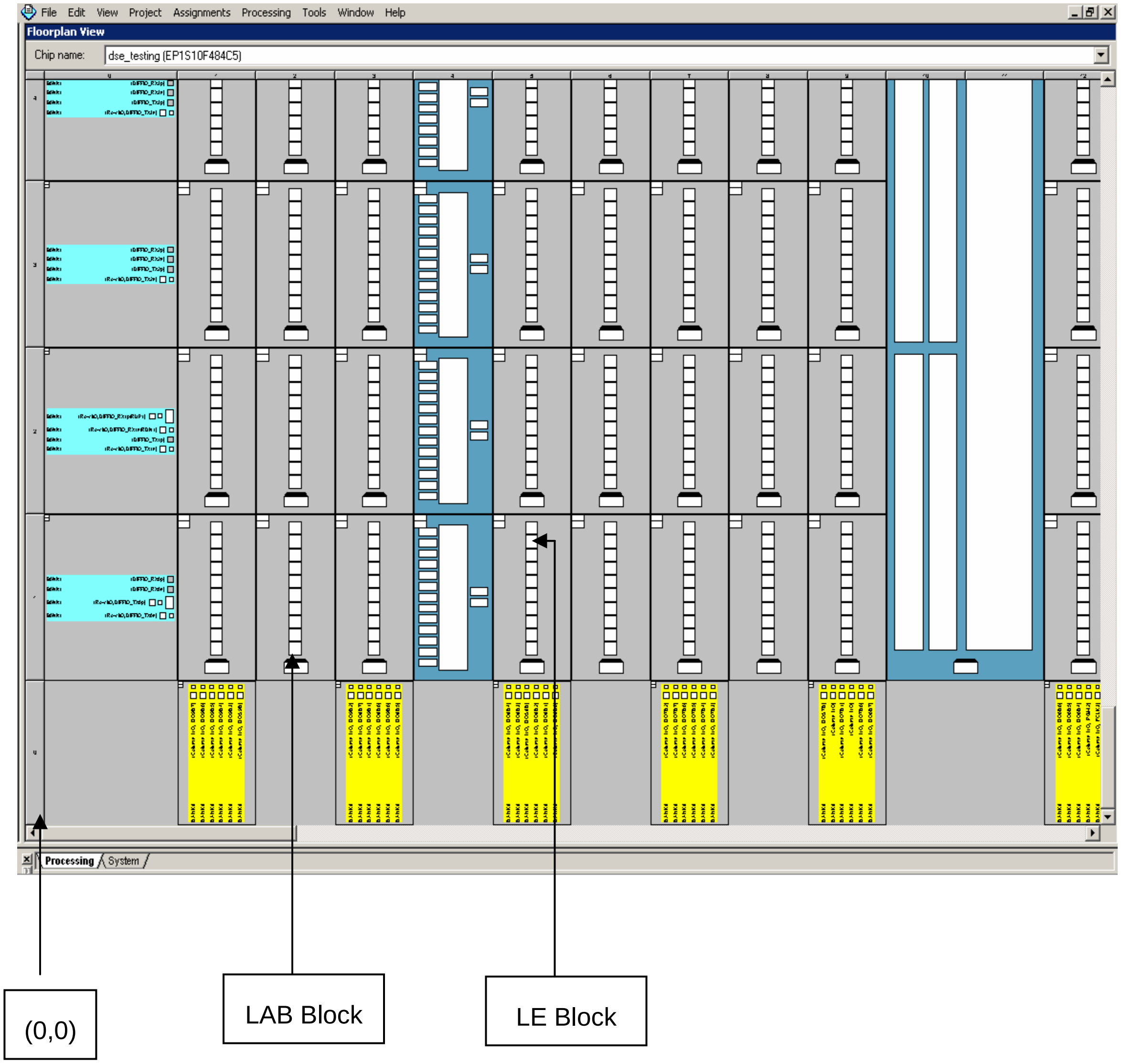
Figure 5-6: The <SPEED> element

The <SPEED> element contains one or more <DEVICE> elements. It encapsulates a set of delays that are common for a single speed grade across several <DEVICE> tags. The use of the “grade” attribute is optional. The default value is “COMMON” if no grade is specified.

6. Point To Point Delay Example

The following example shows how to specify the point-to-point delays within a device using the Altera Point To Point Delay format. In particular, we will examine the delays within a logic cell (LCELL or LC), otherwise referred to as a logic element (LE), for a Stratix EP1S10F484C5 device (for more information on the Stratix LE please see [1]). Though it may never be necessary to create your own delay files, this exercise should help familiarize the reader with the structure of the file format and provide a more tactile meaning to the information that it encapsulates.

Figure 6-1 shows a section of the Stratix EP1S10F484C5 device. Observe that the chip is composed of several blocks, including LABs, which contain ten LEs, each at a unique sublocation.


File Edit View Project Assignments Processing Tools Window Help
Floorplan View
Processing
System
(0,0)
LAB Block
LE Block

Chip name: dse_testing (EP1S10F484C5)

Figure 6-1: Stratix EP1S10F484C5 Partial Floorplan

Figure 6-2 shows a more detailed view of an individual logic element, displaying the various input and output ports.

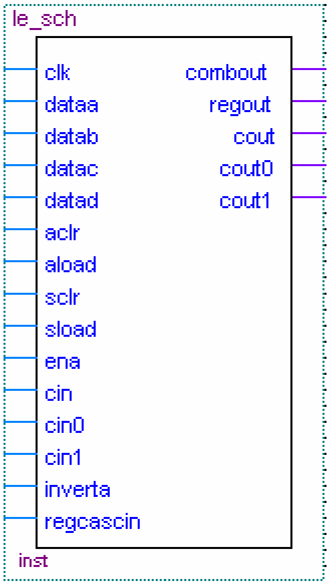


Figure 6-2: Stratix Logic Cell Ports

The point-to-point delay between two ports on an LE is a function of the sub-location occupied by the LE within the LAB and the mode of operation for the LE. If we wanted to express all such delays in an XML format adhering to the Altera Point to Point Delay File specifications, it could be done as follows (items in **bold** font are descriptions only and not actual XML statements):

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<INTRACELLDELAYS xmlns="http://www.altera.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.altera.com intracelldelays.xsd"
name="Stratix">

  <COPYRIGHT>
  Copyright statement
  </COPYRIGHT>
  <DELAY_UNITS>PS</DELAY_UNITS>
    <SPEED grade=”Common”>
      <DEVICE name="EP1S10">
         <BLOCK type="LCELL" subtype="DEFAULT">
          <LOCATION x="0" y="0" subloc="0">
           <MODE>
            <ATTRIBUTE name="FAST_CARRY" value="FALSE"/>
            <ATTRIBUTE name="CIN_AS_INVERTA" value="FALSE"/>
            <ATTRIBUTE name="EMULATING_PACK_REGISTER"
               value="FALSE"/>
```

```
14           <I name="cin">
15             <O name="cout">98</O>
16             <O name="combout">469</O>
17             <O name="ff">598</O>
18           </I>
             ...
             Other <I> / <O> pairs for this <MODE>
             ...
19         </MODE>
           ...
           Other <MODE>s for this <LOCATION>
           ...
20       </LOCATION>
         ...
         Other <LOCATION>s for this <BLOCK>
         ...
21     </BLOCK>
       ...
       Other <BLOCK>s for this <DEVICE>
       ...
      </DEVICE>
22     ...
      Other <DEVICE>s for this <SPEED>
       ...
23   </SPEED>
     ...
     Other <SPEED>s for this <INTRACELLDELAY>
     ...
24 </INTRACELLDELAY>
```

The first two lines simply define the version of XML being used, the namespace, the type of text encoding and the name of the architecture. Following the copyright statement, units for all delays given are declared as picoseconds. After this, the speed grade is specified, a device is given and the block is identified as the “DEFAULT” version of a logic cell. Next, it is noted that this information pertains to an LCELL at a zero offset from its parent at sublocation zero. Recall that the delay is also a function of the operational mode of the LE, which is given by the attribute statements on lines 11-13. These attributes control the following:

- FAST_CARRY: If true, implies that an LE uses a ripple carry chain [1] as opposed to a block carry. When this is the case the cin - cout delay is shorter.
- CIN_AS_INVERTA: This will be true for LEs that appear at the head of a carry chain. In such a case the inverta path is used for the cin signal.

- EMULATING_PACK_REGISTER: Implies that an LE is emulating a packed register mode, where SLOAD is tied to VCC. In such a case, the DATAC is directly feeding a D flip-flop internal to the LE, and so the regout port will reflect the DATAC input.

At this point it is possible to specify delays between two ports on the LE. Lines 14-18 show delays (in picoseconds) between the cin port and each of cout, combout and ff (flip-flop). After this, any other point-to-point delays related to this location and this mode are specified. Next, delays for the (0,0,0) location in other modes are specified. Finally, the delays for other locations (the 9 other sub-locations) are given and this completes the description of point-to-point delays for the LE. If the delay were not a function of the block's location, only one location would have been specified with the co-ordinates (0,0,0).

The above process is repeated for other blocks present on this device, and then for each other device and finally for additional speed grades if necessary.

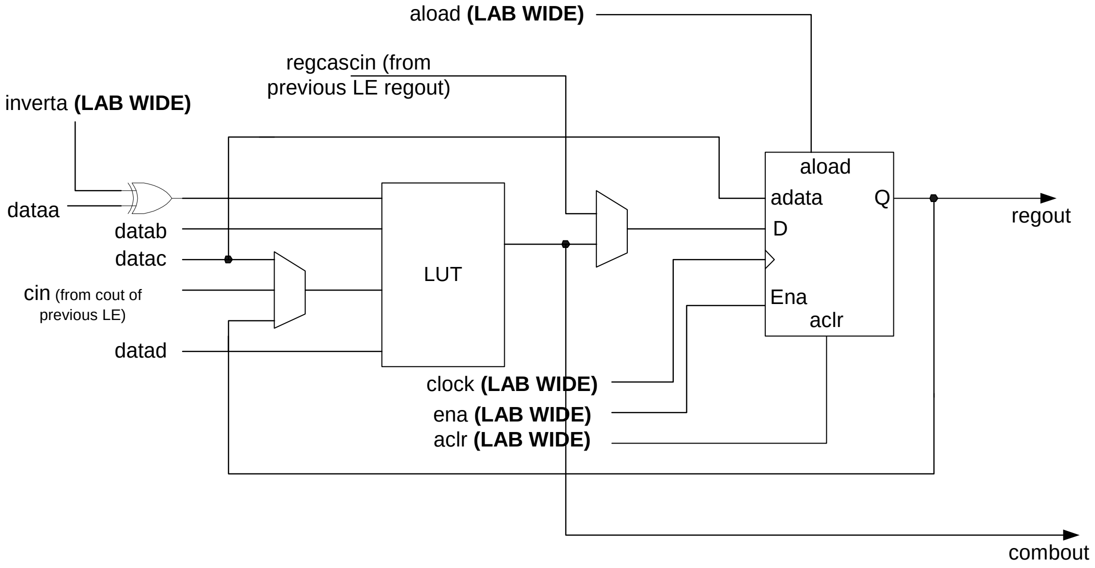
Note that not all <I> / <O> paths will be valid for every LE. The set of valid paths depends on the mode of the LE as defined in [1]. Note that the meaning of the term <MODE> as an element in an XML file is not the same as the word 'mode' in [1]. With an LE, for example, different <MODE>s in an XML file refer to variations on 3 particular parameters: FAST_CARRY, CIN_AS_INVERTA and EMULATING_PACK_REGISTER (the meaning of each variable is defined above). Each of these settings can be set to true or false and 5 different <MODE> elements correspond to 5 combinations of these values. However, modes in [1] correspond to a different set of modes that the LE can assume. These modes determine how the LE will function and they often impose restrictions on how the LE input and output ports can be used. Some of these restrictions for the Stratix LE are summarized in Table 2.

Table 2: LE Modes And Input / Output Port Restrictions

Mode Type	Mode Value	Restrictions
operation	Normal	- cin must not be connected unless sum_lutc_input is set to cin (see below)
operation	Arithmetic	- datad cannot be connected - cout must be connected
synch_mode	Off	- sload cannot be connected - sclr cannot be connected
synch_mode	On	- either sload or sclr must be connected - clk must be connected
register_cascade_mode	On	- regcascin must be connected - clk must be connected

register_cascade_mode	Off	- regcascin must be disconnected
sum_lutc_input	datac	- The datac input to the LUT (the 4-input LUT if in arithmetic mode) is fed by datac
sum_lutc_input	cin	- The datac input to the LUT (the 4-input LUT if in arithmetic mode) is fed by cin
sum_lutc_input	qfbk	- The datac input to the LUT (the 4-input LUT if in arithmetic mode) is fed by qfbk

The schematic for the internal operation of an LE changes for different modes (in the WYSIWYG guide sense of the word). Figure 6-3 below is the schematic for an LE whose operation mode is normal and whose synch mode is off.



operation_mode = Normal

synch_mode = off

register_cascade_mode = {off, on}

sum_lutc_input = {datac, cin, qfbk}

Figure 6-3: Stratix LE In Normal Mode With Synch Mode Off

If such an LE is not emulating a packed register and is not using any sort of carry chaining, then according to the XML code above, the point-to-point delay between cin and combout would be 469 picoseconds. However, we see from the schematic that there is no cout being used. Though the XML file may contain an entry for the point-to-point delay between cin and cout, the LE's mode (in [1]'s sense of the word) makes this path irrelevant. Please consult the full set of schematics in [1] to determine which paths make sense in which modes.

7. Additional References

http://www.w3.org/XML/ -- The World Wide Web Consortium is in charge of the XML standard. This is a good place to start for general standard information on XML, XSLT, XML Schema, DOM and other XML-related standards.

http://www.xml.com/ -- The O'Reilly Group's XML resource site. Lots of tutorials and examples are available under the 'Programming' section. There is also a great annotated version of the XML 1.0 specification, complete with in-line notes and pointers, at http://www.xml.com/axml/testaxml.com.

http://xml.apache.org/ -- Home of the Apache Group's Xerces XML parser. Their goal is "to provide commercial-quality standards-based XML solutions" within the open source framework. In addition to Xerces for C++, Java, Perl and COM you will Xalan, an XSLT stylesheet processor for transforming XML in Java and C++.

CodeNotes for XML, Edited by Gregory Brill – An excellent and invaluable reference for any developer new to XML.

The **Altera XML Architecture Description File Detailed Design document** describes in detail how blocks are assembled and built in to devices. It is a worth at least glancing at if you are trying to understand the fine details of intra-block delays described in this document.

8. Referenced Documents

1. “WYSIWYG Device Primitives User Guide For Stratix.”

This document is being provided on an "as-is" basis and as an accommodation and therefore all warranties, representations or guarantees of any kind (whether express, implied or statutory) including, without limitation, warranties of merchantability, non-infringement, or fitness for a particular purpose, are specifically disclaimed.