XILINX®

Design Migration from XC4000 to XC5200

XAPP 060 October 15, 1996 (Version 2.0)

Application Note by Chris Lockard and Marc Baker

Summary

The XC5200 delivers the most cost-effective solution for high-density, reprogrammable logic designs not requiring the dedicated XC4000 Select-RAM^{TM}, system features, or very high performance levels. This Application Note reviews the differences between the XC5200 and XC4000 families, recommends approaches for converting XC4000 designs to the XC5200 architecture, and provides a methodology to migrate designs easily in multiple CAE environments.

Xilinx Family	**Demonstrates**
XC5200	Techniques for migrating XC4000 designs to the XC5200 architecture

Introduction

The new fourth-generation SRAM-based XC5200 FPGA family from Xilinx is engineered to deliver the lowest cost-per-gate of any FPGA family. The Xilinx XC4000 family of FPGAs, on the other hand, offers a feature-rich architecture ideal for PCI, DSP, telecommunications, and other higher-performance applications. High-density XC4000 designs not using the XC4000 dedicated features, such as on-chip Select-RAM^{TM} and dedicated wide edge decoders, can easily achieve significant cost reductions by migrating to the XC5200, which is 100% footprint compatible with the XC4000.

Note that some variations of the XC4000 family are no longer recommended for new designs. These include the XC4000 standard family, XC4000A, XC4000H, and XC4000D. These have been superseded by the newer XC4000 Series families, which include the XC4000E, XC4000L, XC4000EX, and XC4000XL families. While designs in the older versions of the XC4000 can be easily converted to the new XC4000 Series, the XC5200 family should also be considered for its low cost. See the related application note “XAPP062 - Design Migration from XC4000 to XC4000E”.

The XC4000 to XC5200 conversion will probably require schematic changes. In addition to the lack of RAM, the XC5200 family does not have storage elements in the I/O, although it adds latches in the CLBs. The XC5200 does not have wide edge decoders, although it offers wide logic gating in the CLBs.

Following the guidelines presented in this Application Note will greatly improve the chance of a successful migration, but does not represent a guaranteed solution.

Overview

- Design Performance

  The XC5200 family is optimized for low cost, and the fastest XC5200 will be approximately one speed grade slower than the fastest XC4000. Significant architectural differences make timing changes difficult to predict between the XC4000 and XC5200.

  Recommendation: When moving from an XC4000 to the an XC5200, use a speed grade one level faster than the original speed grade for the first pass. For example, move an XC4000D-5 to an XC5200-4. Use XACT-Performance^{TM} to achieve the highest migration success. Use XDelay and simulation to verify performance prior to production. If performance is not sufficient in the fastest XC5200 device, consider migrating to the XC4000 Series instead.

- On-Chip RAM

  The XC5200 family, like the XC4000D family, does not contain dedicated on-chip RAM.

  Recommendation: Convert to the XC4000 Series if you are using RAM. Very small RAM blocks can be converted to flip-flops in the XC5200 family.

- Carry Logic

  The XC5200 family has a different type of carry logic. It requires more CLBs than the XC4000, but is still more efficient than using look-up tables for carry. XC4000 schematics with RPMs (Relationally-Placed Macros), which use the carry logic, do not need to be modified when converted to the XC5200. Performance, however, will vary according to the implementation.

Recommendation: Use XACT-Performance to maintain required design performance.

- Cascade Logic

  The XC5200 carry logic may also be used to implement wide logic functions. This feature can replace the XC4000 dedicated wide edge decoders.

  Recommendation: Replace edge decoders and wired functions with wide logic functions.

- Three-State Buffers

  The XC5200 3-state buffers cannot be used to implement wired ANDs. No pullups are available to force a mux to one when all buffers are disabled.

  Recommendation: Replace Wired ANDs with wide logic functions. Remove pull-ups from 3-state buffer outputs, or considering using logic gates instead. See Figure 2 on page 6.

- Input/Output Blocks (IOBs)

  The XC5200 IOBs are capable of sourcing 8mA, less than the XC4000 devices. The XC5200 IOBs, like the XC4000H, do not contain flip-flops or latches. As a result, the XC5200 provides the lowest cost per gate of any FPGA. Also, the XC5200 library contains macros using CLB storage elements that are identical in function to the XC4000 primitives for IOB storage elements. Thus, no schematic changes are necessary. The XC5200 inputs allow the NODELAY parameter, similar to the XC4000 and XC4000A families, to reduce setup time.

  Recommendation: Do not use XC5200 I/Os to source more than 8 mA. Convert to the XC4000 Series for 12 mA drive. If possible, consider splitting the load across two outputs.

- Global Clock Buffers

  There are only four global buffers (BUFGs) in the XC5200 family. They connect to the same dedicated I/O pins as the four XC4000 primary buffers (BUFGPs), not the four XC4000 secondary buffers (BUFGSs).

  Recommendation: Move clock inputs driving an XC4000 BUFGS to the dedicated pin for the XC5200 BUFG, or accept longer clock delays in the XC5200. Use the XC4000 Series if more than four global buffers are needed.

- Oscillator

  The oscillator in the XC5200 architecture is different from that of the XC4000. The two lowest frequencies available are 1 KHz and 256 Hz.

  Recommendation: The 490 Hz and 15 Hz oscillator outputs in the XC4000 can be generated from higher frequencies in the XC5200 using counters as clock divider circuits.

Design Guidelines & Considerations

Because the XC5200 is a different architecture (see Table 1), there are a number of issues to consider before migrating an XC4000 design to the XC5200.

Table 1: Comparison of Resources Between XC4010, XC5206, and XC5210

Resource	**XC4010**	**XC5206**	**XC5210**
Max Logic Gates	10,000	10,000	16,000
Maximum flip-flops	1,120	784	1,296
Maximum I/O pins	160	148	192
Configuration bits	178,136	106,288	165,488
4-input function generators	800	784	1,296
3-input function generators	400	0	
Flip-flops per CLB	2	4	
Global buffers	8	4	
Decoder capability	Dedicated	Expanded Cascade	
Carry logic	Yes	Simplified	
Internal 3-state drivers per horizontal line	22 w/pullups	16 no pullups	20
Output drive	12 mA	8 mA	
Output slew rate control	Yes	Yes	
Boundary-scan	Yes	Yes, expanded	
Internal oscillator	Yes	Yes, more flexible	
Packages	PC84, PQ160, TQ176, PG191, PQ208, HQ208, BG225	PC84, PQ100, VQ100, PQ160, TQ144, TQ176, PG191, PQ208	PC84, PQ160, TQ144, TQ176, PG223, PQ208, PQ240, BG225

Configuration

The XC5200 uses the same configuration process as the XC4000 and supports all XC4000 configuration modes. The XC5200 also requires less configuration memory (i.e., it has smaller PROM requirements) than the comparable XC4000 device.

The XC5200 adds an Express mode that shortens configuration time. The Express mode is similar to the slave mode but instead of transferring a single bit of data per cycle, it transfers a byte of data per cycle. Taking advantage of Express mode reduces configuration time by a factor of eight.

Footprint

XC4000 and XC5200 devices are footprint compatible for every common package. All control pins, configuration pins, and power pins are in the same locations. No board re-layout is necessary when replacing an XC4000 device with an XC5200. In addition, the VersaRing^{TM} I/O routing of the XC5200 helps maintain pinouts after design changes.

Density

The XC4000 families achieve their high density through the inclusion of on-chip RAM. If you are using RAM, you should consider migrating to the newer XC4000 Series families instead of the XC5200. However, if you are not using RAM in your current XC4000 design, you may be able to migrate to the next smaller part (as indicated by the part number) in the XC5200 family. See Table 2 below.

Table 2: Density Conversion Table (For Devices Not Requiring RAM)

XC4000 Device	Max Logic Gates	XC5200 Device	Max Logic Gates
XC4002	2,000	XC5202	3,000
XC4003	3,000	XC5202	3,000
XC4004	4,000	XC5204	6,000
XC4005	5,000	XC5204	6,000
XC4006	6,000	XC5204	6,000
XC4008	8,000	XC5206	10,000
XC4010	10,000	XC5206	10,000
XC4013	13,000	XC5210	16,000
XC4020	20,000	XC5210	16,000
XC4025	25,000	XC5215	23,000

Design Performance

Because the XC5200 was engineered primarily for low cost, the fastest XC5200 device is one to two speed grades slower than the fastest XC4000. A good approach is to use a speed grade one level faster than the original speed grade for the first pass (e.g., move an XC4000D-5 to an XC5200-4). However, the slowest XC4000 (XC4000-6) will often migrate to the slowest XC5200 (XC5200-6). Use XDelay and simulation to verify performance prior to production.

XDelay can be used to report performance at various speed grades without changing the LCA file. To show the delays of the most critical paths, create a short XDelay report using the following command:

```
xdelay -u <speed> -o critical.rpt <design_name>
```

For example:

```
xdelay -u -3 -o critical.rpt new.lca
```

This command produces a text file called critical.rpt that contains the minimum worst-case pad-to-setup, clock-to-setup, and clock-to-pad values allowable for each clock in the design. Effectively, this report provides all the information necessary to evaluate the performance of the new speed grade.

Alternatively, in the Windows environment use the Performance Summary in the Timing Analyzer. Both XDelay and the Timing Analyzer can also be used to examine specific path delays in more detail if desired.

Because the XC5200 has a different architecture, most of the XC4000 mapping, placement, and routing information will be irrelevant when migrating to the new XC5200. The PPR Guide option cannot be used. Consequently, the best way to ensure that design performance will be met in the XC5200 is to use XACT-Performance to define the timing requirements of the design.

Development System

The XC5200 is supported by the same XACTstep^{TM} development system and uses the basic software tools that support the XC4000. A new XC5200 library has been added to the set of Unified Libraries to support the architectural features of the XC5200. These new library elements are described in detail in the *Libraries Supplement Guide*.

All Unified Library elements for the XC4000, with the exception of those listed in Table 3, are compatible with the XC5200. Library differences result from the XC5200 family's lack of Asynchronous Preset, RAM, and internal three-state pull-ups.

Table 3: XC4000 Unified Library Symbols Not Compatible With XC5200

FDP	RAM16X1	ROM16X1
FDPE	RAM16X2	ROM32X1
IFDI	RAM16X4	WAND1
ILDI(_1)	RAM16X8	WAND4
OFDI	RAM32X1	WAND8
OFDEI	RAM32X2	WAND16
OFDTI	RAM32X4	WOR2AND
OSC4 => OSC5	RAM32X8	BUFOD

Many new XC5200 optimized elements were added to the XC5200 schematic library. These elements include:

CY_MUX, CY_INIT

XC5200 carry mux, automatically used in XC5200 arithmetic functions and wide gates.

DEC_CC4-DEC_CC16

Cascadable versions of decoders, using CY_MUX. Use these to create AND gates wider than 16 inputs. In addition, the DECODE macros use the CY_MUX, and no longer require a pull-up on the output. Be sure to remove pull-ups on DECODE symbols when converting a schematic from the XC4000 to the XC5200 libraries. The XC5200 library also adds wider DECODE macros: DECODE32 and DECODE64. See Figure 1 on page 5.

BSCAN

Boundary scan symbol. Has been expanded in the XC5200 library to add RESET, UPDATE, and SHIFT outputs. Since the other pin locations did not change, no schematic changes should be necessary after conversion. See Figure 4 on page 8.

OSC5

Internal multiple-frequency clock-signal generator. Creates only two outputs, of programmable frequency. The XC4000 equivalent, OSC4, creates five outputs of fixed frequency, of which only three can be used in a design. After converting a schematic, the OSC5 symbol will have to be added and the appropriate frequency selected. Frequency selection is specified via the DIVIDE1_BY and DIVIDE2_BY parameters. See Figure 3 on page 7.

CK_DIV

Internal clock divider. Allows division of a user-provided external clock. New in XC5200, did not exist in XC4000.

STARTUP

Defines source of Global Reset and Global Three-State. The GSR (Global Set/Reset) pin on the XC4000 version of the symbol is limited to GR (Global Reset) in the XC5200 version. Flip-flops can be reset, but not set. Flip-flops that must be initialized to a “one” need inverters added to the D and Q pins to emulate the XC4000 functionality.

LD

CLB latches, a feature not available in the XC4000 families. These should be used in place of XC4000 IOB latches, since the XC5200 family has no I/O storage elements. An XC4000 IOB latch component will automatically convert to an XC5200 CLB latch macro. However, you may want to consider taking advantage of the lookup table or control signals available to a CLB latch. See the *Libraries Supplement Guide* for a complete description of the variations on the XC5200 latch library components.

AND12, AND16 (OR12/16, NAND12/16, NOR12/16)

Wide gates using the carry mux in the XC5200 CLB. These gates are similar to the DECODE macros in the XC5200 library. Either the DECODE macros or the wide gate macros should be used in place of XC4000 DECODE or WAND macros. The XC5200 wide gates should also be considered as replacements for cascaded gates in the XC4000. Note that all Boolean functions of eight or more inputs are RPMs (Relationally Placed Macros) that use the XC5200 carry logic.

F5_MUX

2-to-1 lookup table multiplexer primitive. Allows two adjacent lookup tables in an XC5200 CLB to be combined together into one result without using another lookup table. Allows any five-input function in one CLB. The F5_MUX is not used automatically, as five-input functions can be effectively implemented in two levels of logic. The F5_MAP mapping symbol forces the use of the F5_MUX. Add the F5_MAP to any five-input functions that demand the highest possible speed.

On-Chip RAM

Neither the XC5200 family nor the XC4000D contains the on-chip RAM feature of the rest of the XC4000 families. This similarity makes the XC5200 family a natural choice for designs being converted from the XC4000D. RAM elements are not available in the library for either the XC4000D or the XC5200. It is still possible to duplicate RAM functionality using flip-flops (as X-BLOX™ will, for example). However, implementing RAM in flip-flops requires a prohibitive amount of resources for anything beyond the most insignificant RAM block. Consequently the XC5200 devices are recommended for designs that do not use on-chip RAM.

Carry Logic

Like the XC4000, the XC5200 supports a carry logic feature that enhances the performance of arithmetic functions such as adders, counters, and comparators. However, the XC5200 carry logic implementation differs significantly from that of the XC4000 family. The XC5200 carry logic is less complicated than that of the XC4000, with fewer resources per CLB devoted to carry logic. Arithmetic functions that use carry logic will require roughly twice as many resources in the XC5200 as in the XC4000. Note, however, that a loadable up/down counter requires the same number of function generators in both families.

Like the XC4000 library, the XC5200 library contains Relationally Placed Macros (RPMs), a set of arithmetic functions designed to take advantage of the dedicated carry logic. RPMs include counters, adders, subtractors, and comparators. The RPM symbols are identical in both libraries, so that no schematic modifications will be necessary when migrating from XC4000 to XC5200. The RPMs can also be used as examples when implementing customized RPMs.

Cascade Logic

The XC5200 carry logic offers the ability to implement a high-speed AND or OR cascade. Cascade logic can be used to implement a high-speed pattern decode in an XC5200 without the need for the additional resources of the dedicated edge decoders of the XC4000. The XC4000 provides a wired function (WAND) requiring a pullup on an open drain symbol; the XC5200 provides the same functionality via a wide logic function implemented primarily in Look-Up Tables (LUTs) with cascaded outputs. The symbols for these two comparable functions are, therefore, slightly different. (See Figure 1.)

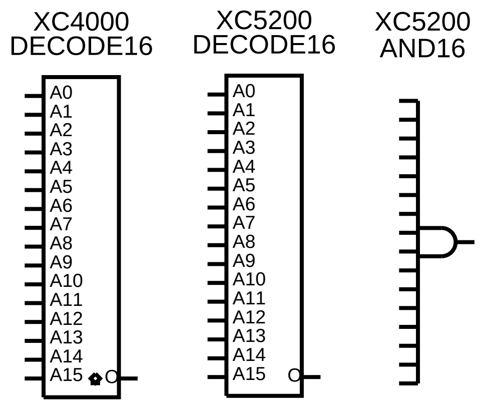


Figure 1: XC4000 Open Drain Decoder Symbol vs. XC5200 Wide Logic Function

If edge decoders are used in the XC4000 design, they will translate to a DECODE macro in the XC5200 library that is equivalent to the wide AND gate functionality. However, the pullup on the output of the XC4000 DECODE symbol will have to be removed in the XC5200 version of the schematic.

Three-State Buffers

Like the XC4000, the XC5200 has dedicated 3-state buffers (BUFTs), but the XC5200 implementation differs in several important aspects. The XC5200 has four BUFTs per logic block, each sharing a common output enable (OE) line. Each BUFT can be driven by any one of the CLB outputs, and can drive two of the eight horizontal *and vertical* longlines. The XC4000 allows each BUFT to drive only a single horizontal longline, and has more restrictive input facilities.

These differences should not be an issue when migrating designs from XC4000 to XC5200, but one additional difference could make an impact. Unlike the XC4000, the XC5200 has no pullups on the ends of the long lines sourced by BUFTs. Consequently, wired functions (i.e., WAND and WORAND) and wide multiplexing functions requiring pullups in undefined states (i.e., some 3-state bus applications) cannot be implemented directly in the XC5200 architecture. Note that in the absence of a pullup, the weak keeper circuit avoids undefined logic levels in both XC4000 and XC5200 families.

There are some simple workarounds for these restrictions. In the case of the wired functions, the same functionality can be achieved by taking advantage of the carry/cascade logic described above, implementing a wide logic function in place of the wired function. In the case of the 3-state bus applications, if the undefined states are “don’t cares,” then the XC5200 weak keeper circuit will retain the previous value on the bus lines when all drivers are 3-stated, and no action is required. If the undefined states are required to be pulled up, then the user must make sure that all states of the multiplexing function are defined. This process is as simple as adding an additional BUFT to drive the bus High during any of the previously undefined states. See Figure 2 on page 6 for an example.

Input/Output Pads

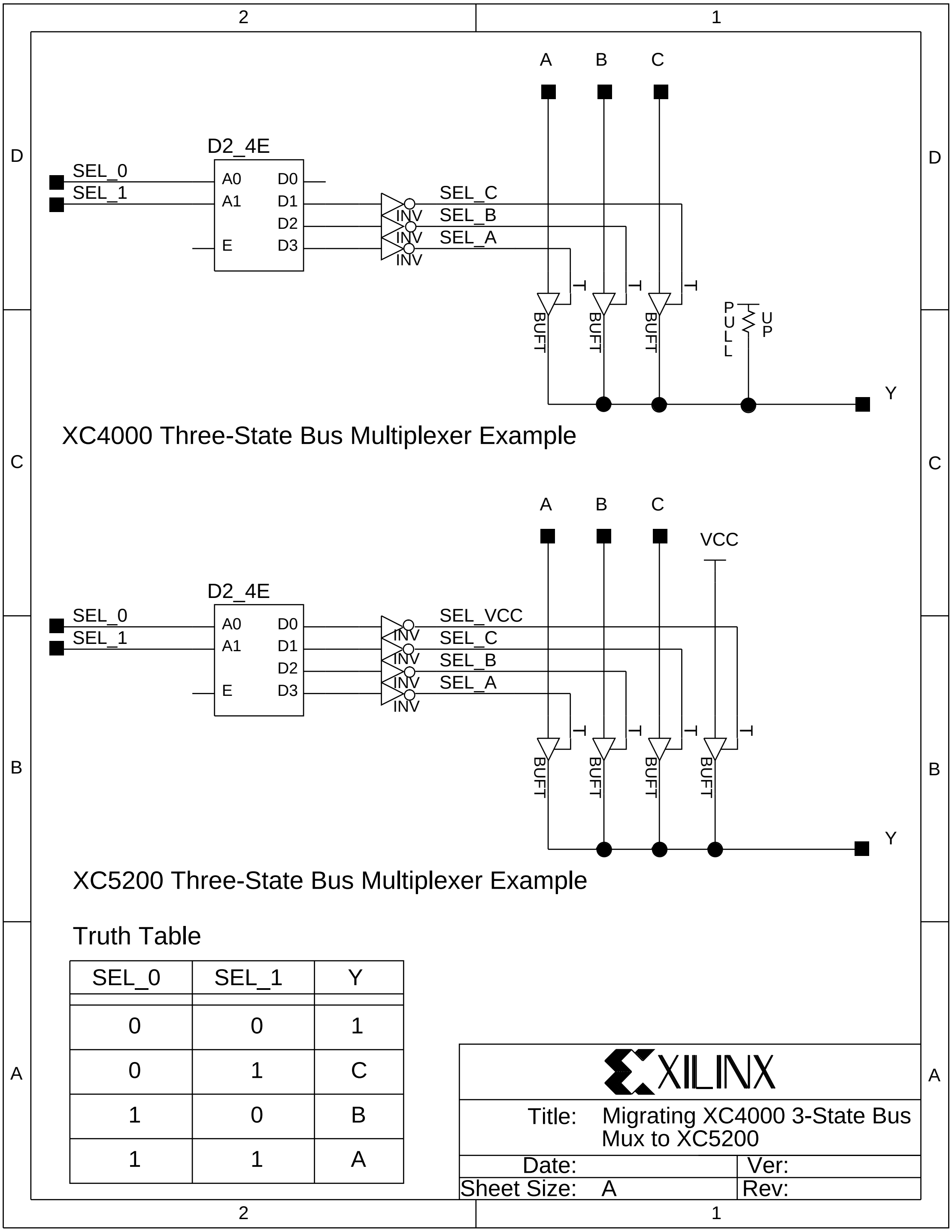
The I/O blocks in the XC5200 differ from those in the XC4000 in that they contain no flip-flops or latches. In order to achieve performance comparable to an XC4000 input flip-flop (IFD), the XC5200 input pad must drive one of the flip-flops in the adjacent CLB. Similarly, an XC4000 input latch (ILD) can be emulated in an XC5200 by driving an LD in the CLB from an adjacent pad. This important placement and routing combination can be achieved by using XACT-Performance to indicate to PPR the timing requirements on the pad-to-setup path, or by using the XACT-Floorplanner.

By reducing the size of the I/O drivers to an output source capability of 8mA, and by removing the registered elements from the I/O pads, the XC5200 family has a significantly smaller die size and hence lower cost than the XC4000 family. However, the output drive is higher in the XC4000 families:

- XC4000/D: 12 mA
- XC4000A: 24 mA
- XC4000H: 4/24 mA (programmable)

XC4000H to XC5200

The XC4000H I/O have a default SoftEdge slew-rate control that limits drive to 4 mA. SoftEdge is intended for capacitively loaded outputs, and is selected with the CAP parameter. The alternative is the resistive (RES) mode, which increases drive to 24 mA. XC4000H designs will need the CAP or RES parameters removed when converted to the XC5200 library. Add the FAST parameter where speed is important, and use two pins tied together to double the drive to 16 mA if necessary.



Truth Table

SEL_0	SEL_1	Y
0	0	1
0	1	C
1	0	B
1	1	A

Title:	Migrating XC4000 3-State Bus Mux to XC5200
Date:	Ver:
Sheet Size: A	Rev:

Figure 2: Multiplexing in 3-state Bus Applications

XC4000H I/O allow per-pin designation of CMOS or TTL input thresholds and output levels, selected for each pin using a CMOS or TTL parameter (TTL is the default). In the XC5200, all outputs are CMOS threshold, and all input thresholds are the same value, either CMOS or TTL (selected as an implementation option). If the XC4000H design has mixed input thresholds, the recommended approach is to designate the XC5200 inputs as TTL, since an input set for TTL can resolve both TTL and CMOS levels. XC4000H designs will need the CMOS or TTL parameters removed when converted to the XC5200.

XC4000H devices have a higher I/O count than the corresponding XC4000E devices. If a high I/O count is required, a larger XC4000E device must be used to provide the same number of pins, as shown in Table 4.

Table 4: XC4000H Device Replacement Guide

XC4000H	Max. I/O	XC5200	Max. I/O
XC4003H	160	XC5206	148
XC4005H	192	XC5210	196

XC4000A to XC5200

The XC4000A I/O allow two additional slew rate options, Medium Fast (MEDFAST) and Medium Slow (MEDSLOW). These parameters, if used in an XC4000A design, will need to be changed to FAST in the XC5200 version to achieve higher-than-default speed.

Flip-Flops

Both the XC5200 and the XC4000 provide flip-flops in the CLBs. The XC5200 also provides level-sensitive latches. Both flip-flops and latches have an Asychronous Clear (Reset). To emulate an Asynchronous Preset (Set) as in the XC4000, add an inverter to the D and Q pins of the flip-flop or latch.

Global Clock Buffers

Global buffers in Xilinx FPGAs are special buffers that drive a dedicated interconnect network throughout the device. This network is specialized for the distribution of high-fan-out clocks or other control signals, such that it minimizes delay and skew while distributing the signal to many loads. The XC4000 has a total of eight global buffers; four global primary buffers (BUFGP), and four global secondary buffers (BUFGS). They share four global-routing channels per column.

The XC5200 has a total of four global buffers (BUFG). For easy conversion from the XC4000 family, the BUFGP and BUFGS components have been included in the XC5200 library, equivalents to the BUFG component.

Each BUFG has its own dedicated routing channel. Two are distributed vertically and two are distributed horizontally such that each global buffer can reach every CLB in the device. The global buffers in the XC5200 map onto the same locations as the global primary buffers (BUFGP) in the XC4000. BUFGP locations were chosen over those of the more flexible BUFGSs to avoid sharing a user clock signal with the DOUT pin, a configuration pin used extensively in daisy-chain applications. This fact is very important to remember when migrating from the XC4000, because additional delay will be added to the distribution of the clock signal if the dedicated pads for the global primary buffers were not used.

Oscillator

The oscillator in the XC5200 is very different from that of the XC4000, as indicated in Figure 3. The XC4000 oscillator (OSC4) provides five predetermined clock frequencies, of which the user may select two, divided down from a nominal 8 MHz internal clock. The XC5200 oscillator is more flexible in that it lets the user choose to divide either the internal 16 MHz clock, using the OSC5 component, or a user clock which is connected to the ‘C’ pin of the CK_DIV component. The user can use both the ‘OSC1’ and ‘OSC2’ outputs of the symbol, and has the choice of division by 4, 16, 64, or 256 on pin ‘OSC1’ and a division by 2, 8, 32, 128, 1024, 4096, 16384, or 65536 on pin ‘OSC2’. The division in each case is user-specified by adding an attribute to the symbol: DIVIDE1_BY= for ‘OSC1’ or DIVIDE2_BY= for ‘OSC2’. Note that two of the frequencies available with the XC4000 are not available with the XC5200 (namely, 490 Hz, and 15 Hz). A small counter could be used to create these frequencies from those available.

Boundary Scan

Boundary scan in the XC5200 is supported in every way that it is supported in the XC4000, and also provides three additional signals — Reset, Update, and Shift. The signals represent the decoding of the corresponding state of the boundary-scan internal state machine. The library symbol for the boundary-scan function (BSCAN) in the XC5200,

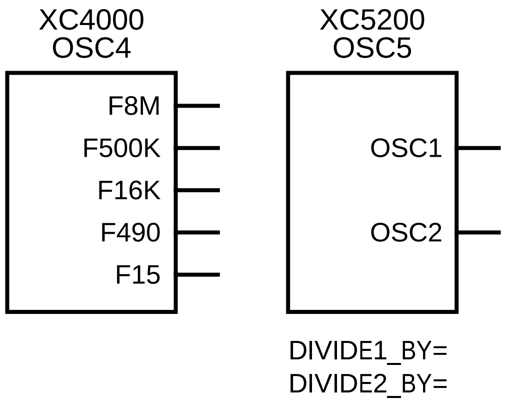


Figure 3: XC4000 OSC4 vs. XC5200 OSC5

therefore, has three more output pins than that of the XC4000. These pins have been added in such a way that user modifications of the schematic are not necessary when migrating from XC4000 to XC5200. See Figure 4 for a comparison.

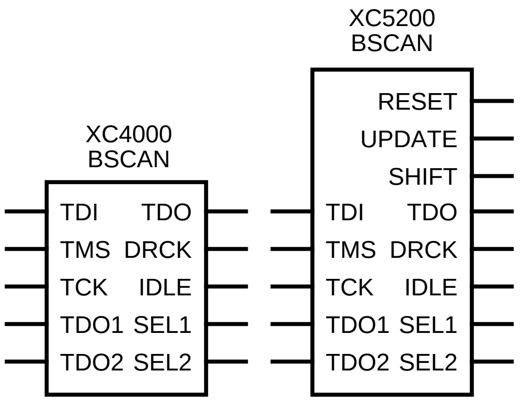


Figure 4: XC4000 vs. XC5200 Boundary-Scan

Startup

The STARTUP symbol in the two devices has the same functionality except for one minor difference (see Figure 5). The signal called Global Set-Reset (GSR) for the XC4000 is called Global-Reset (GR) in the XC5200. The GSR in the XC4000 will asynchronously preset all flip-flops with the attribute INIT=S or a Preset signal, and all IOB storage elements with a trailing “I” in the name (see Table 3 on page 3), and reset all others (INIT=R). The XC5200 does not support the INIT parameter or flip-flops with a Preset signal; activation of the GR signal asynchronously resets all flip-flops in the design.

If an asynchronous preset is desired in an XC5200 design, inverters can be added to the D and Q pins of the flip-flop to create equivalent functionality.

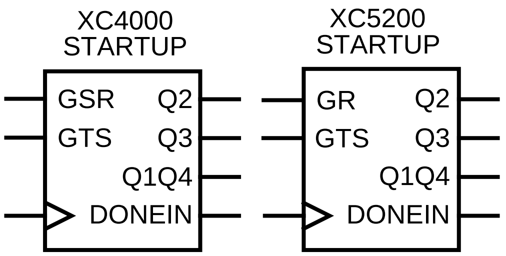


Figure 5: XC4000 STARTUP vs. XC5200 STARTUP

Migration Methodology

Using the Unified Libraries, it is possible to migrate an XC4000 design to the XC5200 architecture, and thus take advantage of significant cost reductions. The migration methodology itself is simple, and following the guidelines and considerations prescribed in this document will greatly improve the success of the migration.

This section describes how to perform the actual migration of an XC4000 design into the XC5200 architecture for three third-party CAE interfaces.

VIEWlogic

To migrate an XC4000 VIEWlogic schematic to the XC5200 architecture, perform the following steps:

1. Add the XC5200 library to the VIEWlogic library search path. Edit the **`viewdraw.ini`** file in the project directory and add the XC5200 library path so that it appears in **`viewdraw.ini`** before the path to the XC4000 library.
2. To convert the XC4000 alias to XC5200, run altran, the VIEWlogic library alias maintenance program:

   `altran -l primary xc4000=xc5200`
   where **`xc4000`** is the alias assigned to the XC4000 library; and **`xc5200`** is the alias assigned to the XC5200 library.
3. Reprocess the design by running XMake or the Flow Engine.

Mentor

To migrate an XC4000 Mentor schematic to the XC5200 architecture, perform the following steps:

1. Invoke **`PLD_DA`** (it is not necessary to open the schematic).
2. On DA’s desktop background (that is, outside of any schematic or symbol windows), call up the session popup menu with the mouse button on the right and select **`Convert Design`**.

   Of the fields in the resulting dialog box these are the most relevant:
3. **`Select a group of designs from a list file?`** Whether you answer "**`yes`**" or "**`no`**" to this question affects the following field.
4. **`Enter Design name (List file = no)`**. The name of the design to retarget. **`Convert Design`** does *not* traverse the hierarchy of a schematic.
5. **`Enter list file name (List file = yes)`**. A file which lists designs, one per line, to retarget. This is useful if your design has many lower-level schematics.

TIP: A list file can easily be created by typing:

```
ls *.mgc_component.attr | sed
 s/.mgc_component.attr//g > listfile
```

The **`ls`** command lists all MGC components within a single directory. The **`sed`** command strips away the **`.mgc_component.attr`** trailer. The result is redirected to **`listfile`**.

6. **`Schematic name`**. The name of the schematic model (the default is "**`schematic`**").
7. **`Check & Save switch`**. Because all schematic sheets in **`Convert Design`** are literally re-drawn in Design Architect, you must apply **`Check & Save`** to each sheet. This switch controls whether to do this automatically. By default, this switch is set for manual checking because it allows you to spot Xilinx components that did not convert properly. Use the manual setting until you are comfortable with how **`Convert Design`** works and you are certain that all Xilinx components will convert properly.
8. **`From Technology`**. The device family from which you are converting (e.g., **`XC4000`**). This and the next field are case insensitive.
9. **`To Technology`**. The device family to which you are converting.
10. After filling out the fields in the dialog box and selecting "**`OK`**," you will see Convert Design doing its job directly in Design Architect.
11. Reprocess the design with XMake or the Flow Engine.

Foundation

To migrate an XC4000 Foundation schematic to the XC5200 architecture, perform the following steps:

1. Select the **`Project Type`** option from the **`Menu`** file. Change **`Family`**, **`Part`**, and **`Speed`** settings, as desired, and click the **`Change`** button.
2. Open and save each schematic sheet macro. To do so, run the Schematic Editor and select the **`Open`** option from the **`File`** menu. The Open Sheet window allows you to quickly open all project top-level sheets and project schematic macros. Inspect the Project Manager messages for any warnings and errors.
3. Re-synthesize and update all FSM and HDL macros. Use the hierarchy browser in the Project Manager to search the project for the macros.
4. Note: if your project contains components that are not available in the new system library, you have to modify the project so as to preserve its functionality. In the case of a top level HDL project, you will need to re-synthesize the entire project.
5. Reprocess the design with XMake or the Flow Engine.

Additional Information

If there are problems with the conversion process, please contact the Xilinx Technical Support hotline for assistance.

Email: hotline@xilinx.com (24 hours)

Voice: 1-80-255-7778 (6:30AM - 5:00PM PST)

FAX: 1-408-879-4442 (24 hours)

XILINX® The Programmable Logic Company℠

Headquarters

Xilinx, Inc.
2100 Logic Drive
San Jose, CA 95124
U.S.A.

Tel: 1 (800) 255-7778
or 1 (408) 559-7778
Fax: 1 (800) 559-7114

Net: hotline@xilinx.com
Web: http://www.xilinx.com

North America

Irvine, California
(714) 727-0780

Englewood, Colorado
(303)220-7541

Sunnyvale, California
(408) 245-9850

Schaumburg, Illinois
(847) 605-1972

Nashua, New Hampshire
(603) 891-1098

Raleigh, North Carolina
(919) 846-3922

West Chester, Pennsylvania
(610) 430-3300

Dallas, Texas
(214) 960-1043

Europe

Xilinx Sarl
Jouy en Josas, France
Tel: (33) 1-34-63-01-01
Net: frhelp@xilinx.com

Xilinx GmbH
Aschheim, Germany
Tel: (49) 89-99-1549-01
Net: dlhelp@xilinx.com

Xilinx, Ltd.
Byfleet, United Kingdom
Tel: (44) 1-932-349401
Net: ukhelp@xilinx.com

Japan

Xilinx, K.K.
Tokyo, Japan
Tel: (03) 3297-9191

Asia Pacific

Xilinx Asia Pacific
Hong Kong
Tel: (852) 2424-5200
Net: hongkong@xilinx.com