Architecture Issues and Solutions for a High-Capacity FPGA

Steve Trimberger, Khue Duong, Bob Conn
Xilinx, Inc.
2100 Logic Drive
San Jose, CA 95124
steve.trimberger@xilinx.com

1 Abstract

High-capacity FPGAs pose device architects with a variety of problems. The most obvious of these problems is interconnect capacity. Others include interconnect performance, clock distribution and IO capacity. This paper describes these problems and the solutions to these problems chosen in the Xilinx XC4000EX family architecture.

2 Overview

XC4000EX family of devices extends the architecture of the XC4000 [Hsieh 1990][Trimberger 1994] to larger gate counts. Devices have been announced with over 2300 CLBs and nearly 7000 LUTs. The XC4000EX CLB is compatible with the XC4000E, leveraging eight years of applications and software development. The XC4000EX includes additions and extensions for high-capacity devices [Xilinx 1996].

The basic tiled structure of the XC4000 devices is shown in figure 1. We increased the logic capacity of the family by building larger array sizes, without changing the CLB structure or removing interconnect or switches. By limiting changes to interconnect additions, the XC4000EX arrays are backward-compatible with existing XC4000 and XC4000E designs (though the devices are not bitstream-compatible). Changes in the I/O block were also made as backward-compatible improvements. Therefore existing XC4000 family logic block cores can be easily ported to the XC4000EX.

This paper addresses three major architectural issues we resolved when making large devices:

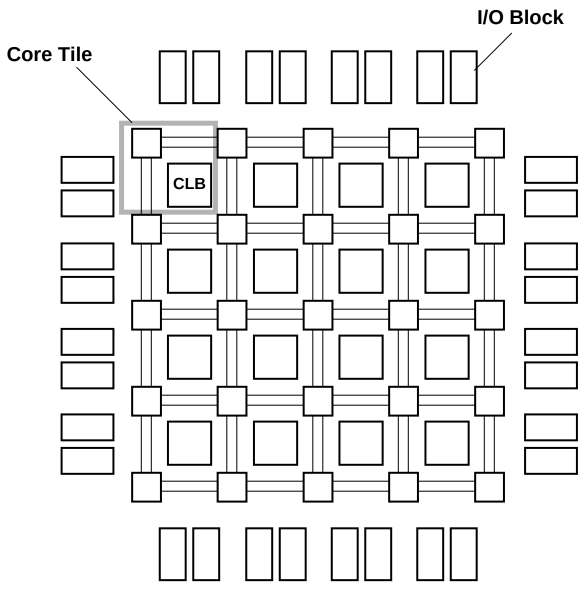


Figure 1. The XC4000 Architecture Tiled Structure

Interconnect. Larger devices require proportionally more interconnect. If the device does not have enough interconnect to support the logic, then some logic will go unused, limiting the capacity of the device, defeating the purpose of building large-capacity devices. When scaling an FPGA architecture to larger capacity, additional interconnect must be added to allow those devices to be routed efficiently.

Clocking. FPGAs typically provide low-skew global clock signals. Low-skew is achieved fundamentally by slowing down fast clock paths to be as slow as the slowest path. On a very large device, the de-skewing circuitry may incur a severe performance penalty for a clock that is only required at the periphery, but is delayed to avoid skew over the whole chip. This delay shows as long clock-to-out times and long input setup times or (worse) non-zero hold times on inputs.

I/O. Integrated circuit process feature size is shrinking much faster than minimum pad spacing. Further, the number of gates that can be implemented on a chip increases quadratically with the periphery, while I/Os are typically limited to the periphery of the chip. Large capacity devices are often pin limited. If there is insufficient IO bandwidth, designs will be unable to take advantage of the full logic capacity of the device.

The remainder of this paper discusses each of these issues in more detail and describes the solution implemented in the XC4000EX device.

3 Interconnect

Over the past twenty years, there have been many attempts to quantify the interconnect channel capacity requirements for logic designs. Theoretical efforts were pioneered by Heller [1978], Donath [1979] and El Gamal [1981], and used recently by Britton [1994]. Predictions from theoretical models are very sensitive to few exponential parameters in those models (i.e. the Rent exponent). Heller [1984] observed large variations in those parameters among designs. Therefore, theoretical models must be verified against empirical data.

An alternative approach using empirical methods, was employed in FPGA architecture investigation by Brown and Rose [1992]. A common criticism of empirical methods is that they are dependent on the tools used to generate those results, so the empirical results may be more a test of a tool set than of the architecture. Fundamentally, though, the tool set is also used to derive the wiring parameters used in theoretical models (the number of used pins per block and average wire length), so in this sense, theoretical methods share the same disadvantage. Sensitivity to a tool set can also be seen as an advantage. Although dependence on a tool set clouds absolute measurements of the routability of an architecture, tool performance provides exactly the right information to use when building a commercial device, because the empirical results match the results seen by users of the device. The architecture can then be tailored to avoid drawbacks in the tool capabilities and to take advantage of special capabilities of the tools.

Channel capacity is only one part of the interconnect demand. FPGAs typically provide a distribution of interconnect segment lengths. Longer segments generally increase performance since signals may traverse a longer distance between programmable switches. However, long segments result in wasted parts of interconnect, since a signal may not use the entire length of the segment to reach its destination. If the architecture has poor connectivity to the long segments, additional interconnect may be required merely to connect to the longer segments. This drives up the number of tracks required to satisfy a given wiring demand.

For evaluation of the XC4000EX interconnect, we used theoretical methods of El Gamal[1981] and Britton[1994] to get a rough estimate the interconnect required, followed by the empirical method to choose the channel segmentation and to fine-tune the number of interconnect lines. The designs we used were chosen from ten ASIC designs from our industrial partners. These results did not drive the architecture alone. Some capabilities of the device were not exercised in the test suite. We built manual designs of a few special structures, such as deep dual-port memories, to ensure that these structures can be routed without creating routing blockages. Macros and tools for automatically-generating these blocks were developed concurrently with the chip design but were unavailable during architecture definition.

Finally, layout and simulation were used to evaluate architectural choices, including channel width, segmentation and interconnect buffering. These results were used to make the final decisions on the number of interconnect lines and their length within the guidelines from the empirical methods.

3.1 Interconnect Location

A separate issue from the quantity of interconnect is the location of the interconnect. This critical issue is easily ignored, particularly when using theoretical models of routability. One option would be to collect additional interconnect into new large wiring channel in the center of the device [Britton 1994]. Intuitively, this would impose an additional burden on the existing interconnect to route long-distance signals to those large wiring channels, then from the channel to their destinations. Further, those signals would be delayed by that intervening interconnect. We decided to distribute the additional interconnect throughout the array, facilitating its usability and minimizing delay. All interconnect is added on a per-tile basis.

3.2 Interconnect Segmentation

Pass-transistor switches add series resistance to FPGA routing paths, resulting in long delays for long paths. Longer segmentation of interconnect lines has been used to address this issue. The XC4000 includes “single” length lines, shown between switch boxes in figure 1. It also include “double” length lines, which span two CLB pitches, and “long” lines, which span the width and height of the FPGA. To support high-performance routability in the XC4000EX devices, we modified long lines somewhat and added another segmentation length: quad lines, which span four CLB pitches.

At very large device sizes, long lines are heavily-loaded, and the wire resistance slows down signals on the line. In the XC4000EX, long lines are broken into four buffered segments, resulting in improved performance and marginally improved routability. These lines are often used to distribute enable signals, and the buffers skew the signals on those lines. Simulation showed that the skew would be small, and we gained a significant performance improvement for signals that traverse the whole device, as well as for those that are distributed only within a quadrant of the device.

To limit the amount of wasted interconnect and to improve performance of routed designs, quad lines are driven directly from CLB outputs and can drive CLB inputs. Quad line switch boxes include an optional buffer. This buffer is used by the delay-driven router to re-buffer signals in the interconnect and to isolate low-performance branches from high-performance loads on the same net. Quad line buffering allows the router to bound worst-case routing delays to about 0.25ns/CLB pitch, significantly improving worst-case routing. The result makes router-assisted optionally-buffered quad line interconnect the preferred routing medium on the device.

Another performance consideration was best-case performance. Many designs include small high-speed pieces, or can be pipelined heavily to gain large performance advantages. To address this high-performance requirement, we added high-speed direct connect paths across the CLB array, and into the IOBs. These direct-connect paths run from the output of a CLB or IOB to the input of an adjacent CLB or IOB. The direct connect delay is five to ten percent of the CLB logic delay. High performance logic with high fan-in can be built by cascading LUT outputs to inputs of neighboring CLBs using direct connections.

As a result of all these considerations, the XC4000EX has a collection of routing resources: long lines for high-fanout distribution; quad lines for high-speed and bounded worst-case delay; direct connect for good best-case delay (figure 2). Humans are not required to deal with this complexity; proper selection of routing resources is handled by the delay-driven router in the Xilinx tool set.

There is more vertical interconnect than horizontal interconnect in the XC4000EX. This asymmetry exists to support address distribution, particularly to dual-port memories in the XC4000EX select RAM.

3.3 Routability of XC4000EX Devices

Table 1 summarizes the results of the empirical tests of the XC4000EX interconnect. The designs in the empirical test set were gate array designs of a variety of desired sizes, chosen to fill the device sizes under test. This table shows results for designs in devices up to 68x68 array size. Many designs are near the maximum capacity of the FPGA, as indicated by the percentage utilization column. The “Router Iterations” column is an indication of router effort. A single iteration indicates that the routing task was very easy; no rip-up and retry was required to complete the route. As can be seen from Table 1, the XC4000EX interconnect is sufficient to route very large devices fairly easily. On an enlarged

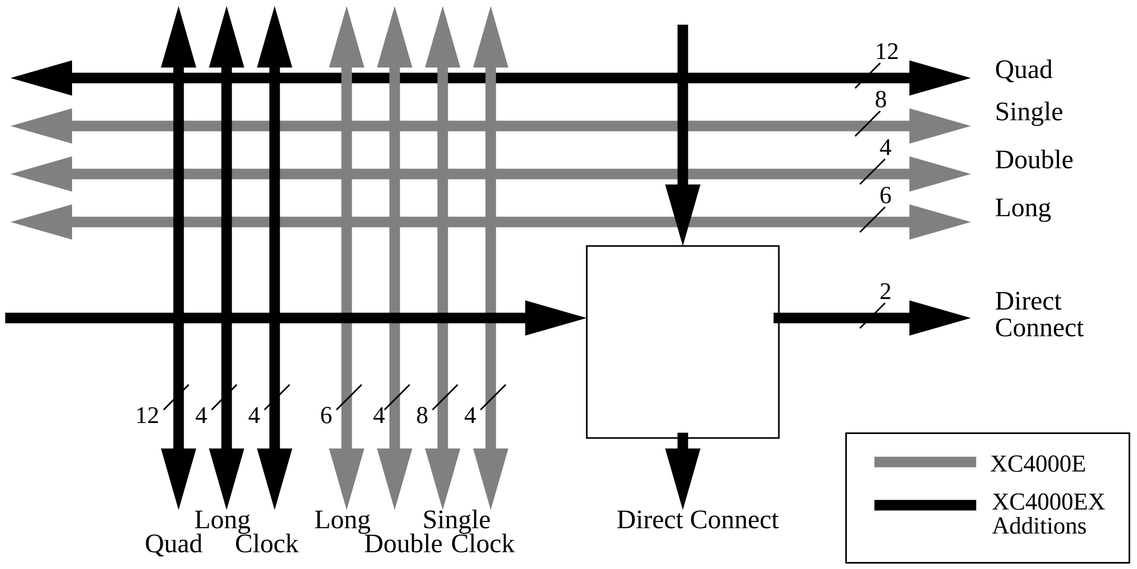


Figure 2. Comparison of routing resources in an XC4000E and XC4000EX tile.

test suite, there were no routing failures on the forty designs tested. These designs are being used to fine-tune software performance.

Design	Device	Array Size	Utilization	Router Iterations
A	XC4062EX	48x48	88%	1
B	XC4062EX	48x48	88%	2
C	XC4062EX	48x48	90%	1
D	XC4062EX	48x48	62%	1
E	XC4062EX	48x48	90%	1
F	XC4062EX	48x48	96%	1
G	XC40125EX	68x68	99%	1
H	XC40125EX	68x68	99%	1

Table 1. Routability Tests for XC4000EX

Increased routing capacity also improves software run times. Results from the design suite and beta test software show the XC4028EX completing placement and routing two to eight times faster than the identically-sized XC4025E.

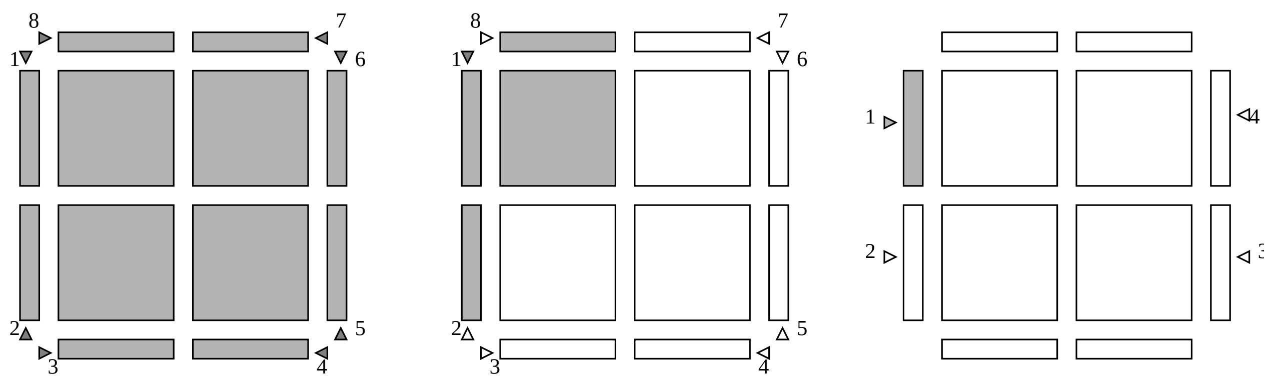
4 Clocking

Figure 3 shows the three different types of clock distribution available on the XC4000EX. The shaded areas are the extent of the clock distribution network for that clock type. The XC4000EX allows eight global clocks (BUFGLS) to be used in each column of the device (figure 3a). Each column can also have other clock sources: interconnect on the device or a special high-speed path from an IOB, simplifying placement and clock distribution for larger number of local clocks or internally-generated clocks.

Fundamentally, low skew is achieved by slowing down fast clock paths to be as slow as the slowest path. This results in long clock delays, particularly for larger devices. Two techniques were used to improve clocking. The first was obvious: improve the clock distribution on the chip with additional buffering and tuned distribution paths.

The second technique was to trade of the size of the low-skew domain for clock speed. Two restricted clocking domains were added to the device. The first domain consists of a quadrant of the chip and the IOBs along one edge of the chip. Each global low-skew clock has an associated restricted clocking domain for an early version of the clock. Clocks on these eight global “early” buffers (BUFGE) bypass the whole-chip de-skewing circuitry to provide faster clocking within the localized domain. Early clocks are about 3-4ns faster than low-skew clocks. Figure 3b shows the clocking domain associated with early clock 1. Logic can be built within a quadrant or along an edge of the device with a faster clock than is possible with the whole-chip low-skew clock.

The second new clocking domain consists of a the IOBs on half a vertical edge of the device. There are four of these smaller, faster clocking domains, each sourced by a dedicated clock buffer (Fast Clock) taking a signal from a specific designated pin directly into the clocking circuitry. Figure 3c shows the clocking domain for fast clock 1. This fast clock provides minimum delay from clock to out on the output flip flop, and minimum setup time on device inputs. It is also used with the OUTMUX (described later) to build high-speed pin-to-pin logic.



a. Eight Global Low-Skew Clocks **b. Eight Global Early Clocks** **c. Four Fast Clocks**

Figure 3. XC4000EX Clocking Domains. a) Global Low-Skew Clock. b) Global Early Clock. c) Fast Clock

5 Input/Output Block

5.1 Input

Figure 4 shows a simplified view of the XC4000EX IOB. Features that are new to the XC4000EX are shown shaded. On the input path, there are two programmable delays. The longer delay (D1) guarantees zero hold time on input signals relative to a clock signal on a low-skew clock buffer (BUFGLS). The shorter delay (D2) guarantees zero hold time on input signals relative to a clock signal on an early clock buffer (BUFGE).

One of the expected uses of the early clock buffer is in conjunction with a the low-skew clock buffer, where the same signal is input to both clock distribution trees. In this usage, the early clock is an early phase of the low-skew clock signal (see figure 5). The early clock is used in IOBs to reduce the clock delay on the input path, and to reduce clock-to-out on the output path. The low-skew clock guarantees that the core of the FPGA can run synchronously.

In this mode, synchronization of the two clocking domains is provided in the IOB, as shown in figure 5. The early clock, connected to $\overline{\text{OK}}$, captures the input data in the SYNC latch in the input path of the IOB. That data is then moved to INFF on the low-skew clock edge, which is connected to IK. Now the entire core of the FPGA can operate synchronously with the low-skew clock. When a signal is to be output, we must re-synchronize from the core low-skew clock to the early clock (OK) on the IOB output flip flop. Since both clocks are the same frequency, this synchronization is accomplished by imposing a tighter delay constraint from the deskewed clock edge to the early clock edge (figure 5). This task is given to the router, to guarantee that the signal from the last flip flop clocked by the low skew clock to the OUTFF clocked with the early clock arrives before the advanced edge of the early clock.

Of course, other clock signals can be used to clock the synchronization latch on the IOB, and the IOB used to assist in the synchronization two unrelated clocks.

5.2 Output

On the output path, the XC4000EX includes a multiplexer that allows the IOB to multiplex two output signals onto the output pin. This feature is useful for building a multiplexed address/data bus, for RAS/CAS encoding or for simply doubling the number of effective IO of the device [Babb 1993]. The existing INFF and direct-input path in the IOB allows the IOB to de-multiplex those signals, so FPGA-to-FPGA connections can make maximum use of the multiplexed IOs.

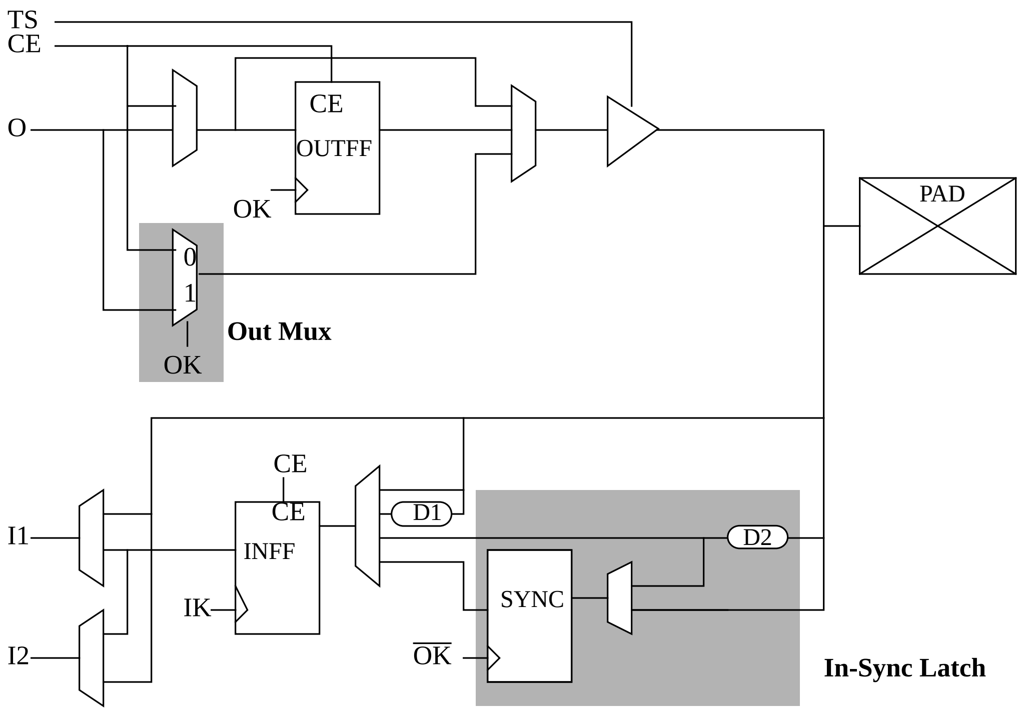


Figure 4. XC4000EX IOB.

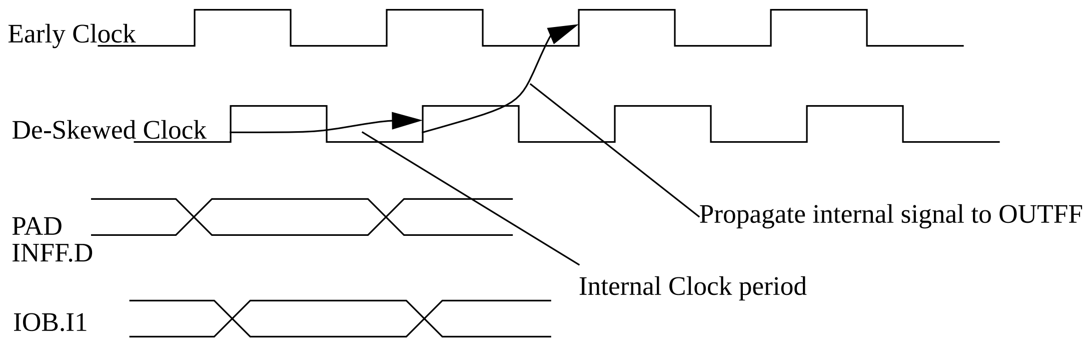


Figure 5. Timing diagram for early clock synchronization.

A typical time-multiplexed IO timing scenario is shown in figure 6. On the sending chip, the output clock (OK) in the IOB selects which of the two signals is driven on the pad. On the receiving chip, the input clock (IK), which is a delayed version of the output clock, captures the OP signal in the input flip-flop in the IOB (INFF). The other multiplexed signal (CE) is available on the other phase of the input clock on the I1 connection. The CE signal can be captured in a CLB flip flop or used directly.

A second, perhaps more interesting, use of the OUTMUX is as a gate. If we tie one leg of the multiplexer to 0 or 1, we get an AND or OR function of the other input and the MUX control. If we provide a signal and its inverse to the inputs of the OUTMUX, we can build an XOR function. The IOB inputs are invertible, so we can build any function of two inputs.

The gate formed from the OUTMUX provides a very fast pin-to-pin delay path on the chip. The MUX control connects directly to the edge clocking, which may be sourced by the FAST Clock. The fast path, then is from the Fast Clock Pad, onto the edge clock distribution circuitry, to the MUX, to the output PAD. No general interconnect is required on this path. The result is very fast pin-to-pin timing for a single gate. A typical use of this gate is as a board-level enable signal, where an address is set up in advance, then gated by a single read-strobe or write-strobe. The critical timing path requires a single gate to AND the strobe with the address decoded on-chip.

6 Summary

This paper discussed several attributes required for a high-capacity FPGA and showed how these were addressed in the XC4000EX family. XC4000EX devices have been announced in sizes with nearly 7000 LUTs and are currently available in device sizes up to 3800 LUTs. It is the author's expectation that larger devices will be available at the time of publication.

7 References

J. Babb, R. Tessier, A. Agarwal, "Virtual wires: overcoming pin limitations in FPGA-based logic emulators", Proceedings of the IEEE Workshop on FPGAs for Custom Computing Machines FCCM '93, IEEE, 1993.

B.K. Britton, et. al., "Second Generation ORCA Architecture Utilizing 0.5um Process Enhances the Speed and Usable Gate Capacity of FPGAs, *Proceedings of ASIC*, IEEE, 1994.

S.D. Brown, *Routing Algorithms and Architectures for Field-Programmable Gate Arrays*, Ph.D. Thesis, University of Toronto, 1992.

S.D. Brown, R.J. Francis, J. Rose, Z.G. Vranesic, *Field-Programmable Gate Arrays*, Kluwer Academic Publishers, 1992.

W.E. Donath, "Placement and Average Interconnection Lengths of Computer Logic", *IEEE Transactions on Circuits and Systems*, April 1979.

A. El Gamal, "Two-Dimensional Stochastic Models for Interconnection in Master-Slice Integrated Circuits", *IEEE Transaction on Circuits and Systems*, Vol. CAS-28, 1981, pp 127-138.

W.R. Heller, W.E. Mikhail, W.E. Donath, "Prediction of Wiring Space Requirements for LSI", *Journal of Design Automation and Fault Tolerant Computing*, May 1978.

W.R. Heller, C.G. Hsi, W.F. Mikhail, “Wirability -- Designing Wring Space for Chips and Chip Packages”, *IEEE Design and Test*, August, 1984.

H.C. Hsieh, et. al., “Third Generation Architecture Boosts Speed and Density of FPGAs”, *IEEE 1990 Custom Integrated Circuits Conference*, IEEE 1990.

S. Trimberger, ed., *Field Programmable Gate Array Technology*, Kluwer Academic Publishers, 1994.

Xilinx, *The Programmable Logic Data Book*, 1996.

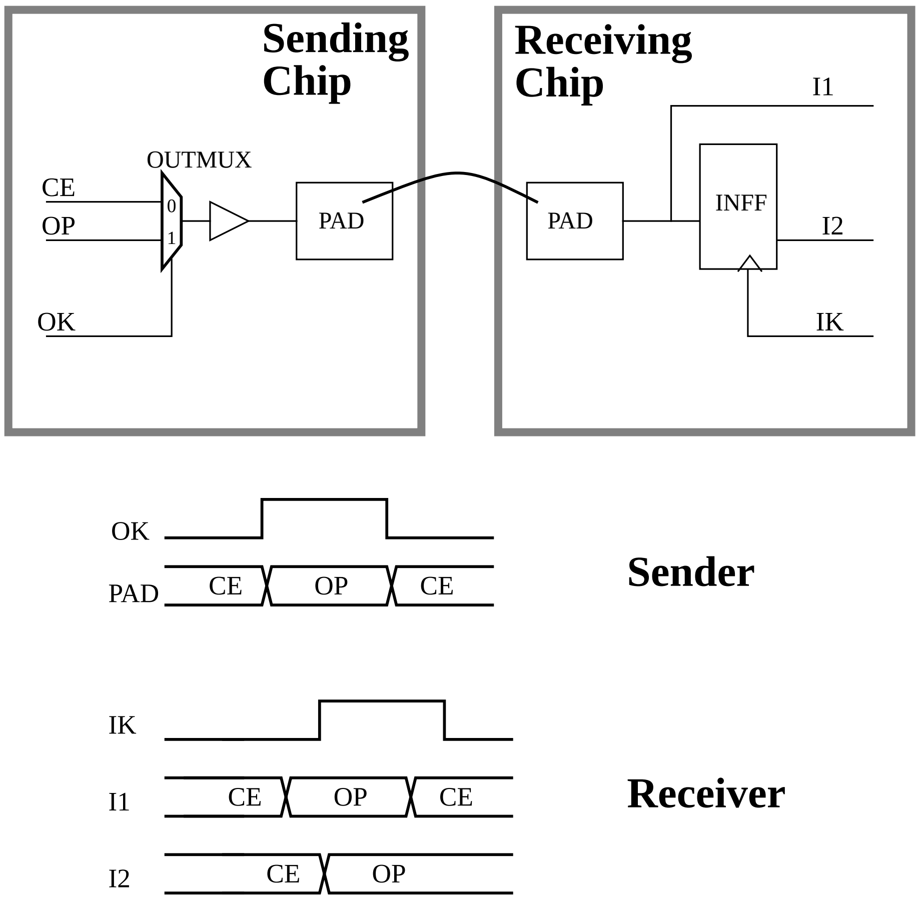


Figure 6. Timing diagram for time-multiplexed IO.