The Effect of Fixed I/O Pin Positioning on The Routability and Speed of FPGAs

Mohammed A. S. Khalid and Jonathan Rose
Department of Electrical and Computer Engineering
University of Toronto, Toronto, Ontario
M5S 1A4, Canada
email: khalid or jayar@eecg.toronto.edu

Abstract

We are motivated to determine the effect of pre-assigning signals to I/O pins in FPGAs because this situation often occurs in everyday board-level use of FPGAs and in multi-FPGA emulator systems and FPGA-based compute engines. This paper presents an experimental study in which benchmark circuits are placed and routed with and without a variety of pin constraints. Experimental results for the Xilinx XC4000 architecture using the XACT 5.1.0 tool set and the Altera FLEX 8000 architecture using the MAX+PLUS II 5.0 tool set show that fixing the assignment of signals to pins in a random fashion can cause an increase of up to 19% in worst-case delay and significantly impact the routing resources needed to complete the routing. For the Xilinx XC4000 architecture random pin constraints caused an increase of up to 20% more single length interconnect segments, 11% more double length interconnect segments, and 49% more long segment interconnect, although no routing failures occurred. For the Altera FLEX 8000 architecture random pin constraints caused an increase of up to 138% more column fast track interconnect, up to 36% more row fast track interconnect, and caused routing failure in three of the fourteen benchmark circuits used. As a general conclusion, however, we found that the effect of fixed pin constraints on delay and routability to be far less detrimental than anecdotal evidence had suggested.

1 Introduction

An important issue in the use of FPGAs in board-level systems is whether or not the user should feel free to pre-assign the signals assigned to the I/O pins. The alternative, allowing the automatic placement and routing software the freedom to choose whichever pins it deems best for each signal, may result in better delay and routability. This question arises in two important situations:

1. When systems designers have already committed to the board-level layout which dictates the pin-signal assignment, and then wish to change the functionality of the FPGA. Although the original pin assignment may have been chosen by the software, the subsequent assignment must remain the same. If major delay increases result from fixing the pin locations in the second iteration, or if routability disappears, then designers will need to account for these likelihoods in the original design.

2. In a multi-FPGA Field-Programmable System (FPS) [Karc94] [Hauc94] [Arno93] [Bert93] [FCCM] [Quic94], which often consists of many FPGAs hard-wired together on a printed circuit board. Once some of the FPGAs are routed, causing signals to be assigned to pins, this implies that the signals on subsequent FPGAs to be routed must be fixed by virtue of the pre-fabricated board-level interconnect architecture [Hauc94]. In determining the architecture of this kind of system (which we are currently doing at the University of Toronto), it is important to know the effect of the fixed pin assignment on the system's speed and routability.

Anecdotal evidence [Chan93b] [Hoel94] suggests that pre-assigning FPGA package pins before placement and routing can adversely affect the speed and routability of several manufacturer's FPGAs. The speed and routability of an FPGA under pin constraints is a function of both the routing architecture of the device (whether or not there are sufficient paths from the pads to all parts of the logic), and the quality of the placement and routing tools (how cleverly it organizes the placement to overcome a difficult pin placement). In this paper we are concerned with the combined effect of routing architecture and automatic layout tools on specific commercial architectures. We present experimental results on the effect of several amounts of fixed-pin assignment on FPGA delay and routability. To our knowledge, no such formal study has yet been done. These results are for the Xilinx XC4000 and the Altera FLEX 8000 families of FPGAs. We intend to use the results of this study to aid in the architecture of the Transmogrifier-2, a Field-Programmable System under development at the University of Toronto.

The paper is organized as follows: In Section 2 we present the methodology used in this work. Research results and their analysis are presented in Section 3. Although the focus is on the effects of fixed pins on delay and routability, a number of interesting observations on other FPGA design issues can also be deduced from the results. We conclude in Section 4 with a few remarks on the significance of the results obtained and plans for future work.

2 Benchmark Circuits and Experimental Procedure

To determine the effect of fixed pin constraints we performed placement and routing on a set of benchmark circuits with and without constraints. The benchmark circuits were obtained from both the MCNC Logic Synthesis 1991 [Yang91] suite, and from several FPGA designs done at the University of Toronto.

The experimental procedure used in our investigation is described below for Xilinx and Altera FPGAs. For the Xilinx FPGAs, each benchmark circuit available in the Xilinx netlist format (XNF), was technology mapped (called “partitioning” by Xilinx), placed, and routed using 5.1.0 version of the Xilinx place and route tool PPR [Xili94b]. For the Altera FPGAs, each benchmark circuit available in the Xilinx netlist format (XNF) was mapped into FLEX 8000 FPGAs by using the version 5.0 of MAX+PLUS II compiler. The compiler accepts circuits described using many standard netlist formats, including XNF, and performs technology independent logic optimization, technology mapping, placement, and routing [Alte94a]. To determine the effect of pin assignment, each circuit was processed under four types of pin constraints, for both Xilinx and Altera FPGAs.

1. No pin constraints (referred to as **npc** in the sequel): Technology Mapping, placement and routing was performed without pre-assigning (fixing) any signals to pins.

2. Same pin constraints (**spc**): The pin-signal assignment was fixed before the placement and routing; the pin assignments were the same as those generated by unconstrained placement and routing run (i.e. **npc**).

3. Bad pin constraints (**bpc**): The pin-signal assignment was fixed before the placement and routing and the pin assignment was intentionally bad. Signals that were assigned to adjacent pins by unconstrained placement and routing run were assigned to pins at opposite ends of the FPGA chip.

4. Random pin constraints (**rpc**): The pin-signal assignment was fixed before the placement and routing and signals were assigned to randomly generated pin numbers.

The output files after place and route were analyzed for worst-case delay and utilization of routing resources. The worst-case delay was determined using the static timing analysis tools available in Xilinx and Altera tool sets. For the Xilinx FPGAs, routing utilization was automatically extracted from the output LCA file using a C program specifically developed for this purpose. The latter measures the number of single-length segments, double-length segments and long lines used by the Xilinx placement and routing tool PPR. For the Altera FPGAs routing utilization statistics are available from the report file that is generated after each compilation run.

For the Xilinx FPGAs, for each of the above four pin constraint cases, five PPR runs were performed and the average delay and the average routing utilization were used. This was done to determine the consistency of the results. The annealing option in PPR was used to obtain different placement and routing results, and hence different delay and routing statistics, for each PPR run. For the Altera FPGAs, a single compilation run for each pin constraint case was sufficient. This is because for a given circuit and pin constraint case, the compiler gives the same placement and routing results for multiple compilation runs, presumably because it uses deterministic algorithms for placement and routing and has no non-deterministic option.

3. Experimental Results and Analysis

In this section we present the result of the experiments. Delay and routability results are given for 16 benchmark circuits for the Xilinx FPGAs and for 14 benchmark circuits for the Altera FPGAs. The circuits are the same except for two circuits that utilized on-chip RAM that is available in XC4000 FPGAs but not in FLEX 8000 FPGAs.

3.1 Results for the Xilinx XC4000 FPGAs

Table 1 presents the effect of fixed-pin assignment on the delay of the Xilinx FPGAs for the benchmark circuits. The circuit name and function is given in column 1. Columns 2 and 3 give the number of I/O pins and the FPGA device used. For all the circuits, the smallest FPGA that would fit the circuit was used. Column 4 gives the percentage of the available pins and configurable logic blocks (CLBs) used by the circuit after placement and routing, and the number of "packed" CLBS or PCLBs. PCLBs is a term used by Xilinx to indicate the minimum number of CLBs the circuit could be packed into if that were the tool's goal. The Xilinx place and route tool will use more than the minimum number if they are available during the placement and routing phase to ease the routing congestion. Column 5 gives the average critical path delay obtained for the circuit with no pin constraints during placement and routing (i.e. the **npc** case). Columns 6, 7, and 8 give the average critical path delay obtained for pin-constrained placement and routing runs for the pin constraints **spc**, **bpc**, and **rpc** respectively. The percentage increase in delay compared to the unconstrained case is given in brackets. The standard deviation in delay was not more than 5% about the average for each type of constraint, for all circuits.

The average delay increase for the **spc** case over all circuits was negligible (1%). This indicates that the placement and routing tool was mostly capable of taking advantage of the good pin assignment it had chosen in the unconstrained case. The average delay increase for the bad pin assignment (**bpc**) case was 5%, and for the random case (**rpc**) was 3%. The greatest increase in delay across all circuits for **spc**, **bpc**, and **rpc** cases were 8%, 19.6%, and 15.4% respectively.

From these results we conclude that fixed pin assignment usually has a minor effect on delay. While the worst case increase was 19% most circuits had increases under 5%. Interestingly, this contradicts the anecdotal evidence cited earlier [Chan93b] [Hoel94]. There are two possible reasons for this:

1. The quality of the place and route tools has improved since the anecdotes were collected.
2. There are many long lines on the chip periphery (18 per row/column) and 6 long lines in each non-peripheral row/column. This allows the I/O pads that are far from where they "want to be" to be transported there effectively around this ring.

Table 2 gives the routing utilization obtained for the same placement and routing experiments as in Table 1. Column 1 gives the circuit name. Column 2 gives the average number of wire segments of length 1, 2, and "long", used by each circuit after unconstrained placement and routing runs. Columns 3, 4, and 5 give the average increase in the number of wire segments used by each circuit after placement and routing with the **spc**, **bpc** and **rpc** pin constraints.

For example, the un-constrained placement and routing of the **dalu** circuit results in an average utilization of 218 long lines, 592 double-length segments, and 1255 single length segments. For the **spc** case average utilization of long lines, doubles, and singles increased by 1%, 5%, and 2% respectively. Similarly the increase in average utilization of long lines, doubles, and singles is shown for the **bpc** and **rpc** cases. The standard deviation about the average for each type of constraint, for all circuits, was less than 10% overall for long lines, and less than 5% for doubles and singles.

It is interesting to note that for all circuits used, none of them becomes un-routable even under the worst pin constraints. This was true even for the circuits that were very tightly packed, in terms of percentage of available CLBs and I/O pins used. This implies that, for the Xilinx XC4000 series (parts 4003 to 4010), there are sufficient tracks per channel to achieve good routability. Also the routability of XC4000 series FPGAs seems to be better compared to that of XC3000 series FPGAs. There are several circuits with high CLB utilization that do not have routability problems. Other researchers working with XC3000 FPGAs reported routability problems in XC3000 FPGAs when the CLB utilization was greater than 80% [Kuzn93]. Compared to **npc** case, the average increase in utilization of wire segments is marginal for **spc** case and significant for **bpc** and **rpc** cases, where 9% more single length lines and 17% more long lines are used.

Overall, we conclude that fixed pin assignment does impact routability significantly, because the amount of routing resources used were increased, but the Xilinx XC4000 series architecture provided sufficient resources to handle the increased demand.

Table 1: Critical Path Delay under Different Pin Constraints for the Xilinx 4000 FPGAs

Circuit	**# I/O pins**	**FPGA Device**	**% of FPGA Pins CLBs and PCLBs used**	**Avg. Crit. Path Delay (npc)**	**Avg. Crit. Path Delay (spc)**	**Avg. Crit. Path Delay (bpc)**	**Avg. Crit. Path Delay (rpc)**
dalu (ALU)	91	4010 DPQ160-5	70% , 100% ,83%	154.4 ns	155.8 ns (+ 1%)	158.7 ns (+2.8%)	163.3 ns (+5.8%)
c1908 (Error Correct Cct)	58	4003 PC84-5	95%, 100%, 98%	133.4 ns	135.9 ns (+2%)	142.2 ns (+7%)	134.2 ns (+1%)
mul (16-bit Mult)	64	4008 PQ160-5	49%, 100%, 99%	247.7 ns	266.4 ns (+8%)	271.2 ns (+10%)	263.8 ns (+7%)
c3540 (ALU + Control)	72	4006 PG156-5	57%, 100%, 89%	173 ns	172.8 ns (0%)	180.3 ns (+4.2%)	177.2ns (+2.5%)
c1355 (Error Correct Cct)	73	4005 PG156-5	65%, 99%, 55%	129.7 ns	128.6 ns (0%)	135.4 ns (+4.4%)	133.4 ns (+3%)
c499 (Error Correct Cct)	73	4003 PQ100-5	94%, 56%, 53%	72.7 ns	70.2 ns (-3%)	73.6 ns (+1%)	71.9 ns (-1%)
c880 (ALU + Control)	86	4005 PG156-5	76%, 48% , 30%	109.1 ns	108 ns (-1%)	108.4 ns (0%)	116.3 ns (+6.6%)
lcdm (LCD Disp Controller)	155	4010 PQ208-5	96%, 100%, 86%	57.1 ns	59.1 ns (+3.5%)	66 ns (+15.6%)	65.6ns (+15.4%)
sw_f128 (Partial Viterbi Decod)	117	4010 PQ208-5	73%, 100%, 91%	42.78 ns	42.9 ns (0%)	51.3 ns (+19.6%)	42 ns (0%)
s1196 (Logic)	30	4005 PG156-5	26%, 78%, 53%	72.6 ns	70.9 ns (-2%)	80 ns (+10%)	75.5 ns (+4%)
s1423 (Logic)	24	4003 PC84-5	39%, 100%, 90%	262.5 ns	266.3 ns (+1%)	266.8 ns (+2%)	263.3 ns (0%)
s5378 (Logic)	86	4006 PG156-5	68%, 100%, 97%	71.4 ns	74.4 ns (+4%)	72.6 ns (+2%)	73ns (+2%)
s820 (PLD)	39	4003 PC84-5	63%, 92%, 68%	53.5 ns	53.9 ns (0%)	54.2 ns (+1%)	53.5 ns (0%)
s832 (PLD)	39	4003 PC84-5	63%, 92%, 70%	53.7 ns	54.4 ns (+1%)	53.4 ns (0%)	53.6 ns (0%)
s838 (Fractional-Multiplier)	39	4003 PC84-5	63%, 100%, 74%	237.7 ns	240.3 ns (+1%)	239.8 ns (+1%)	240.7 ns (+1%)
s9234 (Logic)	43	4010 PC84-5	70% ,100%, 83%	116.7 ns	114.5 ns (-2%)	116 ns (0%)	121.7 ns (+4%)
Avg. Increase					**1%**	**5%**	**3%**

Table 2: Routing Resource Utilization under Different Pin Constraints for the Xilinx 4000 FPGAs

Circuit	**Avg. Segment Usage (npc) longs, doubles, singles**	**Increase in Avg. Segment Usage (spc) longs, doubles, singles**	**Increase in Avg. Segment Usage (bpc) longs, doubles, singles**	**Increase in Avg. Segment Usage (rpc) longs, doubles, singles**
dalu (ALU)	218, 592, 1255	+1%, +5%, +2%	+10%, +5%, +6%	+11%, +1%, +11%
c1908 (Error Correct Cct)	81, 239, 455	3+%, +1%, 0%	+23%, -1%, +7%	+27%, -9%, +9%
mul (16-bit Mult)	187, 589, 1253	+10%, +1%, +6%	+16%, +1%, +7%	+15%, +1%, +5%
c3540 (ALU + Control)	169, 483, 972	+5%, +0%, +4%	+13%, +0%, +7%	+7%, +0%, +5%
c1355 (Error Correct Cct)	67, 300, 345	-3%, +3%, +2%	+40%, +1%, +15%	+37%, -3%, +15%
c499 (Error Correct Cct)	59, 145, 194	-15%, +4%, -7%	+7%, -1%, +15%	+14%, -10%, +15%
c880 (ALU + Control)	63, 203, 266	+6%, +3%, +3%	+44%, +9%, +11%	+49%, +10%, +18%
lcdm (LCD Disp Controller)	259, 750, 2201	+4%, +2%, +5%	+10%, +6%, +10%	+10%, +6%, +12%
sw_f128 (Partial Viterbi Decod)	290, 782, 1786	+2%, +5%, +14%	+5%, +8%, +19%	+2%, +3%, +12%
s1196 (Logic)	98, 280, 516	-2%, -3%, +4%	+8%, -3%, +7%	+16%, -2%, +6%
s1423 (Logic)	60, 195, 339	+5%, +2%, +2%	+9%, +1%, +7%	+12%, -7%, +4%
s5378 (Logic)	236, 639, 1487	+3%, 3%, +6%	+9%, +11%, +20%	+8%, +9%, +15%
s820 (PLD)	63, 158, 257	+1%, +5%, 0%	+19%, +2%, +6%	+19%, -2%, +9%
s832 (PLD)	64, 156, 271	+2%, +8%, 0%	+19%, -6%, +3%	+22%, -4%, +7%
s838 (Fractional Multiplier)	59, 200, 264	-2%, -5%, +3%	+13%, -1%, +6%	+16%, -7%, +6%
s9234 (Logic)	249, 801, 1659	+1%, -4%, +1%	+5%, -1%, +4%	+9%, 0%, +2%
Average Increase		**longs: 1% doubles: 2% singles: 3%**	**longs: 16% doubles: 2% singles: 7%**	**longs: 17% doubles: 0% singles: 9%**

Table 3: Critical Path Delay under Different Pin Constraints for the Altera FLEX 8000 FPGAs

Circuit	**# I/O pins**	**FPGA Device**	**% of FPGA Pins and LEs used**	**Avg. Crit. Path Delay (npc)**	**Avg. Crit. Path Delay (spc)**	**Avg. Crit. Path Delay (bpc)**	**Avg. Crit. Path Delay (rpc)**
dalu (ALU)	91	EPF8820 GC192-3	60% pins 67% LEs	175.2 ns	181.6 ns (+4%)	180.3 ns (+3%)	193.5 ns (+10%)
c1908 (Error Correct Cct)	58	EPF8282 LC84-3	87% pins 63% LEs	137.1 ns	139.6 ns (2+%)	137.9 ns (0%)	139.9 ns (+2%)
mul (16-bit Mult)	64	EPF8820 GC192-3	41% pins 97% LEs	297.6 ns	297.1 ns (0%)	344 ns (+16%)	**75pfp:** 332.5ns (+12%)
c3540 (ALU + Control)	72	EPF8452 GC160-3	60% pins 97% LEs	176.5 ns	166.6 ns (-6%)	**70pfp:** 163 ns (-7%)	**55pfp:** 181.6ns(+3%)
c1355 (Error Correct Cct)	73	EPF8282 TC100-3	95% pins 39% LEs	95.4 ns	91.8 ns (-4%)	**85pfp:** 90.9 ns (-5%)	**90pfp:** 89.3 ns (-6%)
c499 (Error Correct Cct)	73	EPF8282 TC100-3	95% pins 39% LEs	89.5 ns	94.4 ns (+6%)	90.7 ns (+1%)	92.7 ns (+4%)
c880 (ALU + Control)	86	EPF8452 GC160-3	72% pins 31% LEs	137.5 ns	149.8 ns (+9%)	144.4 ns (+5%)	148.9 ns (+8%)
s1196 (Logic)	30	EPF8452 LC84-3	43% pins 65% LEs	72.6 ns	70.9 ns (-2%)	80 ns (+10%)	75.5 ns (+4%)
s1423 (Logic)	24	EPF8282 LC84-3	37% pins 79% LEs	207.6 ns	203.9ns (-2%)	207.9 ns (0%)	205.6 ns (-1%)
s5378 (Logic)	86	EPF8820 GC192-3	56% pins 69% LEs	66.8 ns	68.6ns (+3%)	73 ns (+9%)	71.4 ns (+6%)
s820 (PLD)	39	EPF8282 LC84-3	60% pins 59% LEs	64 ns	60.1 ns (-6%)	69.4 ns (+8%)	62.7 ns (-2%)
s832 (PLD)	39	EPF8282 LC84-3	60% pins 60% LEs	63.1 ns	65.5 ns (+4%)	67.5 ns (+7%)	65.1 ns (+4%)
s838 (frac-tional mult)	39	EPF8282 LC84-3	60% pins 60% LEs	42.5 ns	42.9 ns (0%)	41.7 ns (-2%)	40.3 ns (-5%)
s9234 (Logic)	43	EPF8820 GC192-3	23% pins 52% LEs	101.6 ns	103.7 ns (+2%)	107.1 ns (+5%)	107.8 ns (+6%)
Average Increase					**0.7%**	**3.6%**	**3%**

Table 4: Routing Resource Utilization under Different Pin Constraints for the Altera FLEX 8000 FPGAs

Circuit	**FastTrack Interconnect Usage (npc) rows, columns**	**Increase in FastTrack Interconnect Usage (spc) rows, columns**	**Increase in FastTrack Interconnect Usage (bpc) rows, columns**	**Increase in FastTrack Interconnect Usage (rpc) rows, columns**
dalu (ALU)	310, 88	-2%, +24%	+6%, +69%	+6%, +76%
c1908 (Error Correct Cct)	138, 46	+3%, -7%	-2%, -4%	+10%, +7%
mul (16-bit Mult)	436, 117	+12%, +11%	+9%, +56%	**100pfp: failure** **75pfp: +7%, +42%**
c3540 (ALU +Control)	230, 68	0%, 0%	**100pfp: failure** **70pfp: +9%, +15%**	**100pfp: failure** **55pfp: +18%, +79%**
c1355 (Error Correct Cct)	87, 56	+2%, 0%	**100pfp: failure** **85pfp: +13%, +7%**	**100pfp: failure** **85pfp: +18%, +5%**
c499 (Error Correct Cct)	86, 55	-2%, +2%	+13%, +11%	+21%, +11%
c880 (ALU + Control)	103, 60	0%, +5%	+4%, +10%	+36%, +35%
s1196 (Logic)	141, 32	0%, +22%	+6%, +138%	+6%, +138%
s1423 (Logic)	93, 17	+8%, +35%	+1%, 0%	+9%, -6%
s5378 (Logic)	371, 108	+14%, +33%	+18%, +61%	+26%, +81%
s820 (PLD)	85, 22	+9%, +9%	+11%, +18%	+12%, +23%
s832 (PLD)	81, 24	+7%, 0%	+15%, 0%	+11%, +17%
s838 (fractional mult)	96, 7	-2%, +29%	-1%, +43%	+4%, +43%
s9234 (Logic)	260, 64	+8%, +36%	+10%, +34%	+11%, +53%
Average Increase		**row tracks:3% col tracks:14%**	**row tracks:7% col tracks:33%**	**row tracks:13% col tracks:43%**

An interesting observation is that the number of doubles and singles used is a small fraction of the total number of doubles and singles available. For example, in the unconstrained case of sw_f128 only about 32% of the available doubles and 21% of the available singles are used. On average less than 25% of the available doubles and singles were used. This demonstrates that a great deal of flexibility may have to be present, but not necessarily used, to complete the routing. Also it appears that the Xilinx placement and routing tool uses as many long lines as possible to minimize routing delay. Note that routing delay is a major contributing factor to the overall critical path delay in an FPGA. More detailed information on the percentage of the total available logic and routing resources used up by each benchmark circuit is given in [Khal95].

3.2 Results for the Altera FLEX 8000 FPGAs

Table 3 presents the effect of fixed-pin assignment on the delay of the Altera FPGAs for the benchmark circuits. This is similar to Table 1 and the purpose of each column is the same. For all the circuits the smallest FPGA that would fit the circuit was used. The average delay increase for the **spc** case over all circuits was negligible (0.7%). The average delay increase for the bad pin assignment (**bpc**) case was 3.6%, and for the random case (**rpc**) was 3%. The worst case increase in delay for **spc**, **bpc**, and **rpc** cases were 9%, 12%, and 16% respectively. We can conclude from these results that the average increase in delay over all the circuits is small and the worst case increase in delay is significant.

Two circuits (**c3540** and **c1355**) were un-routable for the **bpc** case and three circuits (**mul**, **c3540**, and **c1355**) were un-routable for the **rpc** case. To enable the tool to complete routing, some of the pins were left unassigned (the tool chose the pin assignment). For example, for the circuit **mul** under the **rpc** case, **75pfp** implies that the circuit would successfully route if 75% of the pins were fixed and 25% of the pins were left unassigned.

Table 4 gives the routing utilization obtained for the same placement and routing experiments as in Table 3. This is similar to Table 2 and the purpose of each column is the same. Altera uses a two-level hierarchical routing architecture and the routability is determined by the utilization of the row and column fast track interconnects that span the whole length and width of the chip [Alte94b]. Compared to the **npc** case, the average increase in utilization of row and column fast track interconnect is quite pronounced in all other pin constraint cases.

The Altera FLEX 8000 FPGAs seem to be slightly susceptible to routing failures under random pin constraints in cases where the I/O pin or logic element utilization is close to 100%. The cause of this seems to be the architectural restriction that each I/O pin can connect to only one (unique) row or column of routing tracks (fast tracks). Some flexibility here, e.g. allowing an I/O pin to connect to a number of rows or columns, will probably lead to better routability under random pin constraints. It seems that system designers, when implementing a circuit using FLEX 8000 FPGAs, should leave about 20% of the logic elements and I/O pins free to avoid routability problems due to pin constraints. For less tightly packed circuits, the amount of routing resources used were increased markedly, but there are sufficient routing resources available to handle the increased demand. More detailed information on the percentage of the total available logic and routing resources used up by each benchmark circuit is given in [Khal95].

4 Conclusions and Future Work

In this paper we have presented experimental results on the effect of fixing FPGA pin assignment on the resulting speed and routability. We showed that the effects on delay are marginal on average and significant in particular cases. The effects on delay are more pronounced in the case of circuits that use up almost all of the available FPGA I/O pins and logic blocks. The effects on routability are significant for almost all the circuits.

The main contribution of our work is that we have presented some quantitative results on the effects of fixing FPGA pins on delay and routability. So far the evidence to this effect was anecdotal and contrary to what our results indicate. Our results will be useful to system designers using FPGAs in their hardware designs and CAD tool developers involved in the development of layout synthesis tools for multi-FPGA systems.

It appears that the extra long lines provided on the periphery of XC4000 FPGAs are effective in handling pin constraints. One research issue this raises is how many such lines should be provided for FPGAs of different sizes and different number of I/O pins. The FPGAs from the Xilinx XC3000 family are still widely used. A similar study for those FPGAs is needed to determine the effect of pin constraints on their speed and routability. Circuits that utilize carry chains and wide edge decoders [Xili94a] may limit the flexibility available to the placement and routing tool. The effect of pin constraints may be more pronounced for such circuits.

References

[Alte94a] Altera Corporation, MAX+PLUS II User Manual, 1994.

[Alte94b] Altera Corporation, FLEX 8000 Handbook, 1994.

[Hauc94] S. Hauck and G. Boriello, “Pin Assignment for Multi-FPGA systems”, Proceedings of IEEE Workshop on FPGAs for Custom Computing Machines, 1994.

[Karc94] D. Karchmer, A Field-Programmable System with Reconfigurable Memory, M.A.Sc. Thesis, Dept. of Electrical and Computer Engineering, University of Toronto, 1994.

[Bert93] P. Bertin, D. Roncin, and J. Vuillemin, “Programmable Active Memories: A Performance Assessment,” Proceedings of the 1993 Symposium: Research on Integrated Systems, MIT Press, 1993.

[Chan93a] P. K. Chan, M. D. F. Schlag, and J. Y. Zien, “On Routability Prediction for FPGAs,” Proceedings of 30th ACM/IEEE DAC, 1993.

[Chan93b] P.K. Chan, University of California, Santa Cruz, California, Private Communication.

[FCCM] Proceedings of IEEE Workshops on FPGAs for Custom Computing Machines, 1992, 1993, and 1994

[Hoel94] W. Hoelich, Aptix Corporation, San Jose, California, Private Communication.

[Khal95] M. Khalid and J. Rose, “The Effect of Fixed I/O Pin Positioning on The Routability and Speed of FPGAs”, CSRI Technical Report under preparation, University of Toronto, 1995.

[Kuzn93] R. Kuznar, F. Brglez, and K. Kozminski, “Cost Minimization of Partitions into Multiple Devices”, Proceedings of 30th ACM/IEEE DAC, 1993.

[Quic94] Quickturn Systems, Product brief: The Enterprise Emulation System and Logic Animator, 1994.

[Arno93] J. M. Arnold and D. A. Buell, “VHDL Programming on Splash 2,” Proceedings of IEEE Workshop on FPGAs for Custom Computing Machines, 1993.

[Yang91] S. Yang, Logic Synthesis and Optimization Benchmarks User Guide, Version 3.0, Microelectronics Center of North Carolina, January 1991.

[Xili94a] Xilinx, Inc., The Programmable Logic Data Book (page 9-11), 1994.

[Xili94b] Xilinx, Inc., XACT Development System User Guide, February 1994.