XAPP636 (v1.1) November 4, 2002

Optimal Pipelining of the I/O Ports of Virtex-II Multipliers

Author: Markus Adhiwiyogo

Summary

This application note and reference design describes a high-speed, optimized implementation of a Virtex™-II pipelined multiplier primitive (MULT18X18 and MULT18X18S) implemented in VHDL and Verilog.

Introduction

The multiplier block in Virtex-II devices is an 18-bit by 18-bit two's complement signed multiplier optimized for high-speed operations. Additionally, the power consumption is lower compared to a slice implementation of an 18-bit by 18-bit multiplier. However, when pipelining the input and the output of the multiplier block, speed can be optimized by using appropriate location and timing constraints.

Full 16-bit by 16-bit Multiplier Overview

Using Virtex-II multipliers without constraints leads to non-optimized performance when creating an I/O pipelined 16-bit by 16-bit multiplier. The reference design file contains VHDL, Verilog, and UCF files to illustrate a speed-optimized pipelined design implementation. Table 1 and Table 2 list the performance improvement available by following the guidelines in this application note.

Table 1: **Non-Pipelined 16 x 16 Bit Multiplier Performance Improvement (Unconstrained vs. Constrained)[5]**

Device	Speed Grade	Unconstrained	Constrained	Units
Standard Design	-4	50	85	MHz
	-5	59	103	MHz
	-6	81	162	MHz
Enhanced Design	-4	64	127	MHz
	-5	73	146	MHz
	-6	81	162	MHz

Notes:

1. Measurements taken using speed files from ISE 4.2i Service Pack 3. The standard design with -6 speed grade devices uses the speed file of stepping 1.
2. The XC2V40, XC2V1000, XC2V3000, XC2V4000, and XC2V6000 devices have both standard and enhanced multipliers. Enhanced multipliers require speed files stepping 1.
3. The XC2V80, XC2V250, XC2V1500, XC2V2000, and XC2V8000 devices have only enhanced multipliers. Enhanced multipliers require speed files stepping 1.
4. 16 x 16 bit non-pipelined multiplier is instantiated using MULT18X18 with the two input MSBs tied to 0.
5. The unconstrained design uses timing constraints but no location constraints. The constrained design uses timing constraints and location constraints.

Table 2: **Pipelined 16 x 16 Bit Multiplier Performance Improvement (Unconstrained vs. Constrained)[5]**

Device	Speed Grade	Unconstrained	Constrained	Units
Standard Design	-4	87	118	MHz
	-5	101	139	MHz
	-6	133	246	MHz
Enhanced Design	-4	103	190	MHz
	-5	117	215	MHz
	-6	133	246	MHz

Notes:

1. Measurements taken using speed files from ISE 4.2i Service Pack 3. The standard design with
   -6 speed grade devices uses the speed file of stepping 1.
2. The XC2V40, XC2V1000, XC2V3000, XC2V4000, and XC2V6000 devices have both standard and enhanced multipliers. Enhanced multipliers require speed files stepping 1.
3. The XC2V80, XC2V250, XC2V1500, XC2V2000, and XC2V8000 devices have only enhanced multipliers. Enhanced multipliers require speed files stepping 1.
4. 16 x 16 bit pipelined multiplier is instantiated using MULT18X18S with the two input MSBs tied to 0.
5. The unconstrained design uses timing constraints but no location constraints. The constrained design uses timing constraints and location constraints.

MULT16X16S_PLUS

The MULT16X16S_PLUS module contains an instantiated and pipelined I/O MULT18X18S design using both the input and output of a basic Xilinx register primitive (FDR). Each register uses RLOC and BEL attributes. To make a 16 x 16 multiplier, the two input MSBs are tied to zero.

The input pattern has all of the input to port A of the multiplier on a slice column to the left of the multiplier, while the input to port B of the multiplier is on a slice column to the right of the multiplier. The design minimizes the distance of the register to the input ports to ensure high-speed results. In addition, timing constraints are used in the UCF file on the nets of the design to guarantee the fastest register to multiplier net delay.

For the best results, target the net delays across the multiplier input port to be as short and as fast as possible. If some input nets are longer than others, it is best to place the longer nets on the MSBs of the input port. The net delays across the inputs should be relatively equal.

The output port pattern simply places all the outputs two slice columns to the left and right of the multiplier. This is similar to the input ports. A UCF file is used to guarantee the fastest net delay between register to multiplier.

The net delays across the multiplier output port should be short and as fast as possible. If some output nets are longer than others, place the longer nets on the LSBs of the output port.

SAMPLE_WRAPPER_MULT16X16S_PLUS

The `sample_wrapper_mult16X16S_plus` design file illustrates how to call the MULT16X16S_PLUS in a module.

SAMPLE_WRAPPER_MULT16X16S_PLUS.UCF

The example .ucf file, `sample_wrapper_mult16X16S_plus.ucf`, is used to control the placement of the multiplier block and registers for MULT16X16S_PLUS. It shows the placement of the RLOC_ORIGIN of the registers and the multiplier block. For other desired multiplier locations, this file must be manually changed to accommodate the new locations. The first step

in changing the file involves finding the specific multiplier block to place to, followed by placing the RLOC_ORIGIN on the lowest slice position on the column to the left of the multiplier. To view the layouts of the multiplier and registers, open a new FPGA editor design and choose the appropriate part and package. Figure 1 shows a sample of the MULT16X16S_PLUS.

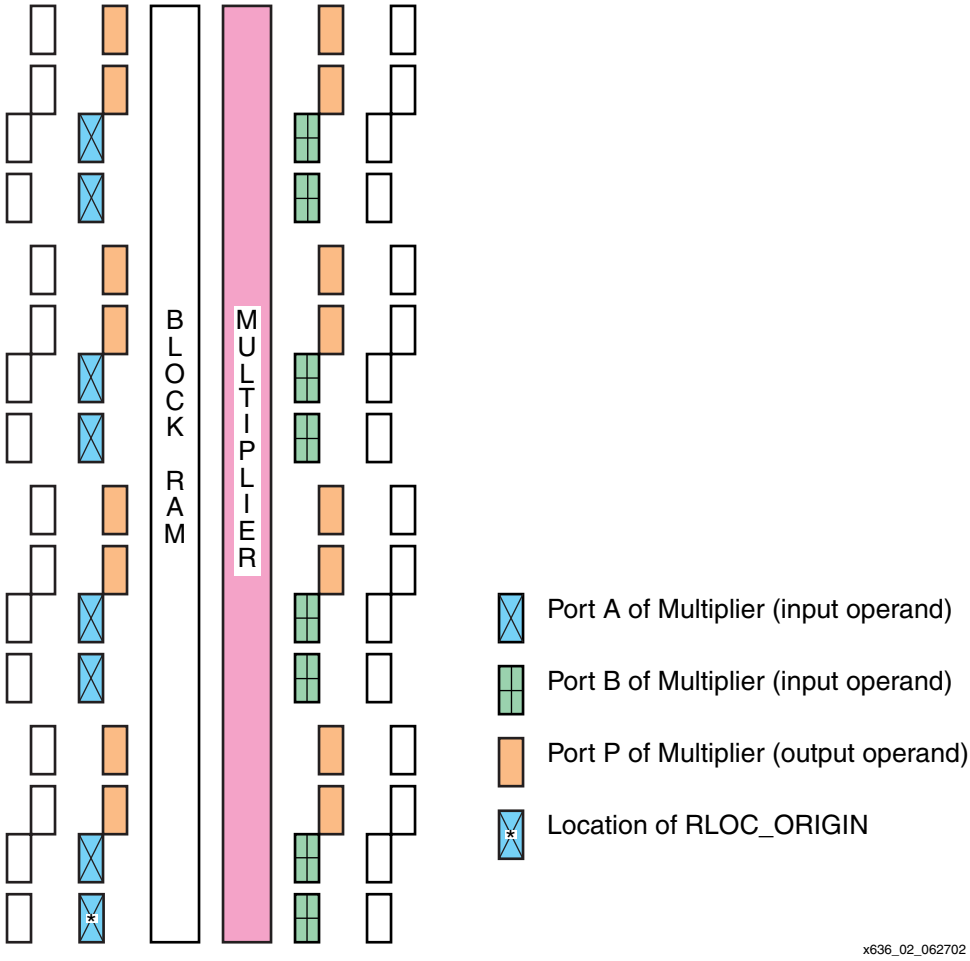

Figure 1: **16 X 16 Multiplier Floorplan**

Using the MAXDELAY attribute, each net between register to multiplier of the design is constrained to be 500 ps or faster. The value of the delay is independent of the device speed grade, and not all the nets will meet this constraint. However, this constraint allows the PAR tool to select the fastest route between the registers and the multiplier.

Table 3 shows the improvement of net delays to and from the multiplier.

Table 3: **Typical Performance Improvement in Net Delays of a 16 x 16 Bit Pipeline Multiplier Using an XC2V1000 (Unconstrained vs. Constrained)**

Port	-4 Speed Grade		-5 Speed Grade		-6 Speed Grade		Units
	Unconstrained	Constrained	Unconstrained	Constrained	Unconstrained	Constrained	
Input	4.932	0.761	4.355	0.662	3.824	0.602	ns
Output	3.869	0.689	3.277	0.599	2.980	0.544	ns

Notes:
1. All values are fastest net delays of unconstrained vs. slowest net delays of constrained in a 16 x 16 multiplier.
2. Net values are independent of stepping and multiplier type.

Full 18-bit by 18-bit Multiplier Overview

Using the full 18-bit by 18-bit multiplier without constraints leads to a non-optimized performance as illustrated on Table 4 and Table 5.

Table 4: **Non-Pipelined 18 x 18 Bit Multiplier Performance Improvement (Unconstrained vs. Constrained)[5]**

Device	Speed Grade	Unconstrained	Constrained	Units
Standard Design	-4	51	76	MHz
	-5	60	92	MHz
	-6	83	147	MHz
Enhanced Design	-4	65	116	MHz
	-5	74	133	MHz
	-6	83	147	MHz

Notes:
1. Measurements taken using speed files from ISE 4.2i Service Pack 3. The standard design with -6 speed grade devices uses the speed file of stepping 1.
2. The XC2V40, XC2V1000, XC2V3000, XC2V4000, and XC2V6000 devices have both standard and enhanced multipliers. Enhanced multipliers require speed files stepping 1.
3. The XC2V80, XC2V250, XC2V1500, XC2V2000, and XC2V8000 devices have only enhanced multipliers. Enhanced multipliers require speed files stepping 1.
4. 18 x 18 Bit non-pipelined multiplier is instantiated using MULT18X18.
5. The unconstrained design uses timing constraints but no location constraints. The constrained design uses timing constraints and location constraints.

Table 5: **Pipelined 18 x 18 Bit Multiplier Performance Improvement (Unconstrained vs. Constrained)[5]**

Device	Speed Grade	Unconstrained	Constrained	Units
Standard Design	-4	83	109	MHz
	-5	97	127	MHz
	-6	133	232	MHz
Enhanced Design	-4	103	180	MHz
	-5	116	205	MHz
	-6	133	232	MHz

Notes:

1. Measurements taken using speed files from ISE 4.2i Service Pack 3. The standard design with
   -6 speed grade devices uses the speed file of stepping 1.
2. The XC2V40, XC2V1000, XC2V3000, XC2V4000, and XC2V6000 devices have both standard and enhanced multipliers. Enhanced multipliers require speed files stepping 1.
3. The XC2V80, XC2V250, XC2V1500, XC2V2000, and XC2V8000 devices have only enhanced multipliers. Enhanced multipliers require speed files stepping 1.
4. 18 x 18 bit pipelined multiplier is instantiated using MULT18X18S.
5. The unconstrained design uses timing constraints but no location constraints. The constrained design uses timing constraints and location constraints.

MULT18X18S_PLUS

The MULT18X18S_PLUS module contains an instantiated and pipelined I/O MULT18X18S design using both the input and output of a basic Xilinx register primitive (FDR). Each register uses RLOC and BEL attributes.

The input pattern has all of the input to port A of the multiplier on a slice column to the left of the multiplier, while the input to port B of the multiplier is on a slice column to the right of the multiplier. The design minimizes the distance of the register to the input ports to ensure high-speed results. In addition, timing constraints are used in the UCF file on the nets of the design to guarantee the fastest register to multiplier net delay.

Two MSBs of both A and B are placed slightly to the left of the top of the left column because the two MSBs will not fit in the same column without interfering with the next adjacent multiplier block. However, the two MSBs are also placed very close to the MSB input ports of the multiplier switch matrix.

For the best results, target the net delays across the multiplier input port to be as short and as fast as possible. If some input nets are longer than others, it is best to place the longer nets on the MSBs of the input port. The net delays across the inputs should be relatively equal.

The output port pattern simply places all the outputs two slice columns to the left and right of the multiplier. This is similar to the input ports. A UCF file is used to guarantee the fastest net delay between register to multiplier.

Output bits 0, 8,16, and 24 are placed to the left of the leftmost column. This was also done because fitting all 36 registers in the same columns without interfering with the adjacent multiplier block is impossible. These bits are placed close to the appropriate switch matrices that are closest to the corresponding bits.

The net delays across the multiplier output port should be short and as fast as possible. If some output nets are longer than others, place the longer nets on the LSBs of the output port.

SAMPLE_WRAPPER_MULT18X18S_PLUS

The `sample_wrapper_mult18X18S_plus` design file illustrates how to call the MULT18 X 18S_PLUS in a module.

SAMPLE_WRAPPER_MULT18X18S_PLUS.UCF

The example .ucf file, `sample_wrapper_mult18X18S_plus.ucf`, is used to control the placement of the multiplier block and registers for MULT18X18S_PLUS. It shows the placement of the RLOC_ORIGIN of the registers and the multiplier block. For other desired multiplier locations, this file must be manually changed to accommodate the new locations. The first step in changing the file involves finding the specific multiplier block to place to, followed by placing the RLOC_ORIGIN on the lowest slice position on the column to the left of the multiplier. To view the layouts of the multiplier and registers, open a new FPGA editor design and choose the appropriate part and package. Figure 2 shows a sample of the MULT18X18S_PLUS.

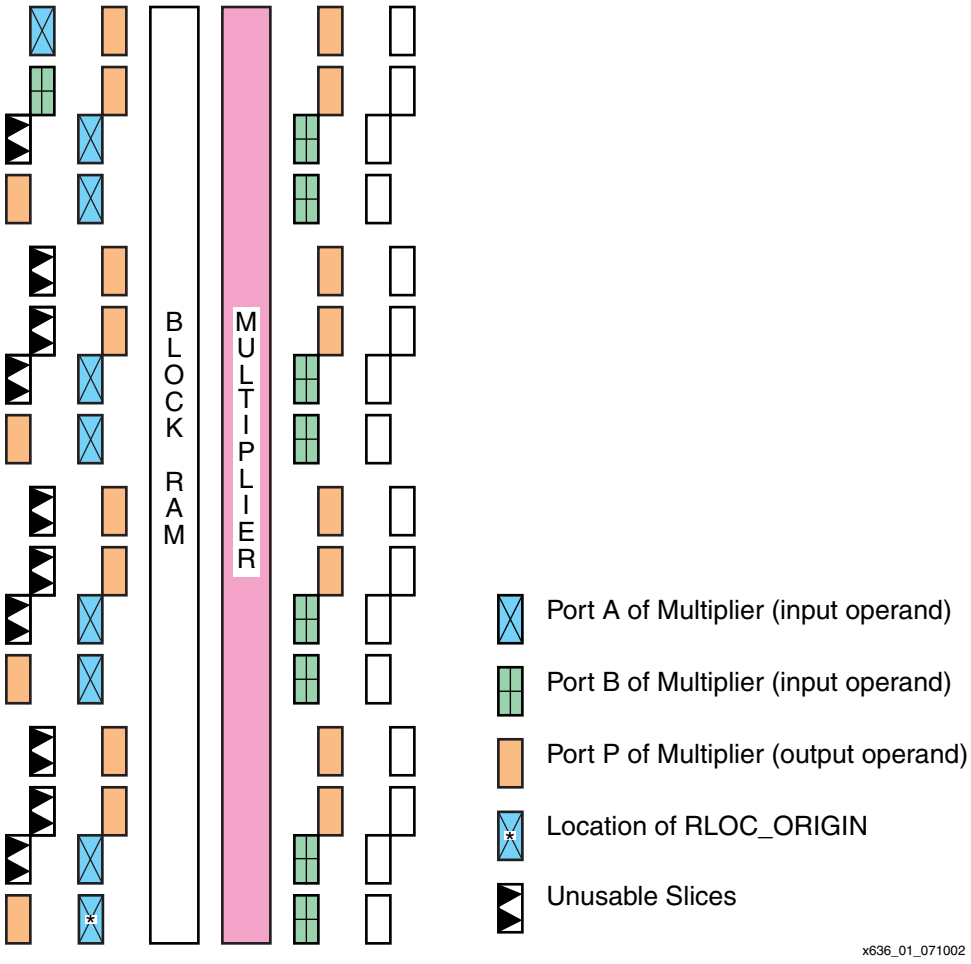

Figure 2: **18 X 18 Multiplier Floorplan**

When using RLOC_ORIGIN and RLOC on the 18 x 18 bit multiplier, about 10 slices will be unusable for every multiplier used. This is because these slices are reserved for the RPM set. The RPM set is bordered by the X and Y, minimum and maximum location of the RLOC.

Using the MAXDELAY attribute, each net between register to multiplier of the design is constrained to be 500 ps or faster. The value of the delay is independent of the device speed grade, and not all the nets will meet this constraint. However, this constraint allows the PAR tool to select the fastest route between the registers and the multiplier.

Unlike the unusable 10 slices, the BRAM is still usable. However due to the heavy usage of the shorter nets into the multiplier, there is a possibility of a non-optimal BRAM performance.

Table 6 shows the typical performance improvement in net delays to and from the multiplier.

Table 6: **Typical Performance Improvement in Net Delays of an 18 x 18 Pipeline Multiplier Using an XC2V1000 (Unconstrained vs. Constrained)**

Port	-4 Speed Grade		-5 Speed Grade		-6 Speed Grade		Units
	Unconstrained	Constrained	Unconstrained	Constrained	Unconstrained	Constrained	
Input	4.932	1.062	4.355	0.925	3.824	0.840	ns
Output	3.869	0.96	3.277	0.836	2.980	0.759	ns

Notes:
1. All values are fastest net delays of unconstrained vs. slowest net delays of constrained in a 18 x 18 multiplier.
2. Net values are independent of stepping and multiplier type.

Smaller than 16-bit by 16-bit Pipeline Multipliers

Using the current methodology of placing the I/O registers into the multiplier will not result in a performance gain as significant as changing the bit width from 18 bits to 16 bits. There are important factors to consider when designing multipliers with less than an 18-bit by 18-bit input. Since the LSB is critical for fast computation, the LSBs of the input should be placed close to the switch matrices that are close to the input port of the multipliers. Similarly, the MSBs of the output should be placed close to the output of the multipliers since the MSB of the result is the slowest to be computed.

Non-pipelined

For the best performance, use the constraints given in the 16 x 16 multiplier example utilizing the inputs and outputs. Both input and output net delays need to be as short as possible for the highest clock rate.

Pipelined

For the best performance, use the constraints given in the 16 x 16 multiplier example utilizing the inputs and outputs. In the small pipelined multipliers, the input net delays are more important than the output net delays because the setup and hold time of the multiplier will be greater than the clock to out timing. Lab simulations show multiplier speed improvement when shifting the inputs to make them centered around input bit 12. These improvements are not reflected in the timing analysis because the setup and hold of the multiplier is the same regardless of the input chosen or how many inputs are used.

Multiplier Performance Comparisons

A suitable Virtex-II device is chosen based upon targeted design performance expectations. Table 7 shows typical net delay performances available from a multiplier listed by device speed grade and multiplier width.

Table 7: **Optimal Net Delays of Pipeline Multiplier**

Multiplier Configuration and Port Location	-4 Speed Grade	-5 Speed Grade	-6 Speed Grade	Units
	Min/Max	Min/Max	Min/Max	
16 x 16 at Input Ports	456/761	397/662	361/602	ps
16 x 16 at Output Ports	639/695	556/605	505/550	ps
18 x 18 at Input Ports	456/1062	397/925	361/836	ps
18 x 18 at Output Ports	638/960	556/836	505/759	ps

Notes:

1. All values apply to all devices with margin of error of ±5 ps

The enhanced multiplier uses a slightly different architecture that allows the multiplier block to have a shorter logic delay. The data for these tables were derived using the Xilinx Integrated Software Environment (ISE) 4.2I SP2 software. Comparisons between the original and enhanced multipliers in a Virtex-II device are summarized in Table 8 and Table 9. These tables also illustrate the performance difference between non-pipelined and pipelined 18 x 18 multiplier primitives.

Table 8: **Detailed Performance of Non-Pipelined Multipliers[1]**

Device and Configuration[2,3]	Speed Grade	T_{MULT} (ns)[6]	T_{CKO} + NET + T_{MULT} + NET + T_{DICK} (ns)[7]	Maximum Frequency (MHz)
16 x 16 Standard Design	-4	9.527	11.915	84
	-5	7.82	9.896	101
	-6	4.272	6.160	162
16 x 16 Enhanced Design	-4	5.412	7.842	127
	-5	4.708	6.822	146
	-6	4.272	6.193	162
18 x 18 Standard Design	-4	10.36	13.049	76
	-5	8.5	10.839	92
	-6	4.660	6.786	147
18 x 18 Enhanced Design	-4	5.904	8.599	116
	-5	5.135	7.48	133
	-6	4.66	6.791	147

Notes:

1. Measurements taken using speed files from ISE 4.2i Service Pack 3. The standard design with -6 speed grade devices uses the speeds file of stepping 1.
2. The XC2V40, XC2V1000, XC2V3000, XC2V4000, and XC2V6000 devices have both standard and enhanced multipliers. Enhanced multipliers require speed files stepping 1.
3. The XC2V80, XC2V250, XC2V1500, XC2V2000, and XC2V8000 devices have only enhanced multipliers. Enhanced multipliers require speed files stepping 1.
4. A 16 x 16 bit non-pipelined multiplier is instantiated using MULT18X18 with the two input MSBs tied to 0.
5. A 18 x 18 bit non-pipelined multiplier is instantiated using MULT18X18.
6. Delay across multiplier block.
7. Register to multiplier to register delay.

Table 9: **Detailed Performance of Pipelined Multipliers [1]**

Device and Configuration[2,3]	Speed Grade	T_{MULTCK} (ns)[6]	T_{CKO} + NET + $T_{MULIDCK}$ (ns)[7]	T_{MULTCK} +NET +T_{DICK} (ns)[8]	Maximum Frequency (MHz)
16 x 16 Standard Design	-4	7.368	5.218	8.427	118
	-5	6.27	4.604	7.191	139
	-6	2.655	4.051	3.492	246
16 x 16 Enhanced Design	-4	3.364	5.254	4.429	190
	-5	2.926	4.636	3.853	215
	-6	2.655	4.079	3.497	246
18 x 18 Standard Design	-4	8.112	5.519	9.171	109
	-5	6.91	4.867	7.831	127
	-6	3.043	4.289	3.880	232
18 x 18, Enhanced Design	-4	3.856	5.53	4.921	180
	-5	3.354	4.876	4.281	205
	-6	3.043	4.298	3.895	232

Notes:

1. Measurements taken using speed files from ISE 4.2i Service Pack 3. The standard design with -6 speed grade devices uses the speeds file of stepping 1.
2. The XC2V40, XC2V1000, XC2V3000, XC2V4000, and XC2V6000 devices have both standard and enhanced multipliers. Enhanced multipliers require speed files stepping 1.
3. The XC2V80, XC2V250, XC2V1500, XC2V2000, and XC2V8000 devices have only enhanced multipliers. Enhanced multipliers require speed files stepping 1.
4. A 16 x 16 bit pipelined multiplier is instantiated using MULT18X18S with the two input MSBs tied to 0.
5. A 18 x 18 bit pipelined multiplier is instantiated using MULT18X18S.
6. Clock-to-Out delay across multiplier block.
7. Register to multiplier to register delay.
8. Output of multiplier to register delay.

Conclusion

Optimized pipeline multiplier speed is achieved when using the guidelines and example design found on the Xilinx FTP site at **xapp636.zip**.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
07/30/02	1.0	Initial Xilinx release.
11/04/02	1.1	Updates to almost all the tables.