**Chapter 21**

**Design Evaluation**

The design of CCM was captured using schematic editor of Xilinx Foundation Series software, and was implemented on 17 Xilinx XC3090A FPGA chips (see Figure 18.11) using M1 software from Xilinx. Now that the CCM has been verified in its operation, a proper timing analysis must be done to evaluation of the design of the CCM.

Since our design was mapping on multiple FPGA chips, we will focus on some paths that span multiple FPGA chips and are likely to have greater delays. The following paths will be discussed in this section:

• The path begins from the outputs of registers accu and data, and goes to the input of the output FIFO. The delay of this path is the time that the CCM takes to compute a combinational cube operation once the content of registers are set properly. This path will be refereed as vertical path in the following section.

• The counter carry path includes the entire iterative network of counter blocks and the circuit that is used to evaluate pre-relation. The delay of this path is the time that the CCM takes to evaluation the signal pre\_res (see section 16.7) once the registers and control signals are set properly.

• The empty carry path is the data path used to generate signal empty. The delay of this path is the that time the CCM takes to generate the result cube, and then determine whether the resultant cube is an empty cube or not once the registers and control signals are set properly.

• The memory path connects two memory banks (MEM\_A and MEM\_b) and the registers accu and data.

The delay of the ready signal will not be discussed here since our design is already able to handle it (see section 16.3.3). Actually, the delay of ready signal is approximate to that of the empty carry path. Now let us analysis the time characteristics of these paths.

• The vertical path.

All ITs are mapped in the first column of the matrix of the DEC-PERLE-1. The output of the ITs (resultant cubes) goes to the output FIFO through MBusE, SWE, DBusNE, FSW and FifoOutData. The delay of 385 ns is the greatest delay of this path.

• The counter carry path.

This path goes to CSW through 4 matrix FPGAs, 3 segments of matrix direct connections, MBusS, SWS and RingSW. The delay of 643 ns is the greatest delay of this path.

• The empty carry path

This path goes to CSW through 4 matrix FPGAs, 3 segments of matrix direct connections and RingMat. The delay of 648 ns is the greatest delay of this path.

• The memory path.

The memory path that connects the memory bank MEM\_A and the registers goes through RamDataW, SWW and MBusW. This path has a delay of 104 ns. The other memory path that connects the memory bank MEM\_B and the registers goes through MBusE, one matrix FPGA, MBusS, SWS and RamDataS. This path has a delay of 160 ns.

As we discussed in section 18.4, the CCM evaluates pre-relation in states P2 and P5 of GCU, and this should be done in one clock period. Therefore, the clock period should be greater than 643ns.

For comparing the performance of the CCM and that of the software approach, a program can carry out disjoint sharp operation on two arrays of cubes was created using C language. Then this program and the CCM are used to solve the following problems:

• Three variables problem: 1# (all minterm with 3 binary variables).

• Four variables problem: 1# (all minterm with 4 binary variables).

• Five variables problem: 1# (all minterm with 5 binary variables).

The C program is compiled by GNU C compiler version 15.7.2, and is run on Sun Ultra5 workstation with 64MB real memory. The CCM is simulated using QuickHDL software from Mentor Graphics. We simulated the VHDL model of CCM, got the number of clocks used to solve the problem, then calculated the time used by CCM using formula: clock \times clock-period. A clock of 1.33 MHz (clock period: 750 ns) is used as the clock of the CCM. The experiential result is shown in Table 21.1.

Table 21.1: Compare CCM (1.33 MHz) with software approach



It can be seen from Table 21.1 that our CCM is about 4 times slower than the software approach. But, the clock of the CPU of Sun Ultra5 workstation is 270 MHz, which is 206 times faster than the clock of the CCM. Therefore, we still can say that the design of the CCM is very efficient for cube calculus operations.

It also can be seen from Table 21.1 that the more variables the input cubes have, the more efficient the CCM is. This is due to the software approach need to iterate through one loop for each variable that is presented in the input cubes.

However, the clock period of 750ns is too slow. From the state diagram of the GCU (shown in Figure 18.10), it can be found that the delays of empty carry path and counter carry path only occur in a few states. Thus, if we can just give more time to these states, then we can speedup the clock of the whole CCM. This is very easy to achieve: for example, the state P2 of GCU need more time for the delay of counter carry path, so add two more states in series between states P2 and P3. These two extra states do nothing but give the CCM two more clock periods to evaluate the signal prel\_res, which means that the CCM has 3 clock periods to evaluate signal prel\_res in state P2 after adding two more “delay" states. After making similar modifications to all these kind of states, the CCM can run against a clock of 4 Mhz (clock period of 250 ns). The CCM was simulated again, and the result is shown in Table 21.2.

Table 21.2: Compare CCM (4MHz) with software approach

****

It is very hard to increase the clock frequency again with this mapping because some other paths like memory path have delays greater than 150 ns.

From the above comparison result, we can conclude that a design like CCM with a complex control unit and complex data path is not good for the architecture of the DEC-PERLE-1 board. It can be seen from our CCM mapping that since a lot of signals must go through multiple FPGA chips, this leads to greater signal delays. For instance, if we can connect the memory banks and the registers directly, then the memory path has a delay of only 35 ns. But our current memory path has a delay of 160 ns. Another issue is that XC3090 FPGA is kind of “old" now (6 to 8 years old technology). The latest FPGA from Xilinx or other vendors has more powerful CLBs and more routing resource, and they are made using deep sub-micron process technology.

If we can map the entire CCM inside one FPGA chip, then we can speedup the CCM from the following aspects:

• If we map entire CCM into one FPGA chip, the signals do not need to go through multiple chips again, which means the routing delay is reduced.

• Since the new FPGA chip has more powerful CLBs and routing resource, we can map the CCM denser. This also reduces the routing delays.

• Since new FPGA chips are made using deep sub-micron technology, the delay of CLB and routing wires are both reduced. For example, the delay of the CLB of XC3090A is 4.5 ns while the delay of CLB of XC4085XL (0.35 micron technology) is only 1.2 ns. This means that it is very easy to achieve 3 times faster mapping.

XC4085XL FPGA, a new FPGA from Xilinx, has a CLB matrix of 56\times56 and up to 448 user I/O pins. The CCM should be able to map into one XC4085XL FPGA. With this new chip, it should not be difficult to run the CCM against a clock of 20 MHz (clock period: 50 ns). This means that our CCM will be about 4 times faster than the software approach while the system clock of the CCM is still 5 times slower than that of the workstation.

As said by the designers of the DEC-PERLE-1 board in paper [9]: PAM technology is currently best applied to low-level, massively repetitive task such as image or signal processing. The example applications are a long integer multiplier, RSA cryptography and Fast Hough transform [9]. All these applications have no or very simple control units, and their data paths can be easily pipelined.

The CCM has a complex control unit, and a complex data path. It is difficult to pipeline the data path of the CCM. Therefore, the DEC-PERLE-1 board is not good for the CCM.