A 2Gb/s 256*256 CMOS Crossbar Switch Fabric Core Design using Pipelined MUX

Ting Wu, Chi-Ying Tsui, Mounir Hamdi[ξ]
Department of Electrical and Electronic Engineering
[ξ]Department of Computer Science
Hong Kong University of Science & Technology
Clear Water Bay, Hong Kong SAR, China
E-mail: eetsui@ust.hk

ABSTRACT

In this paper, we present the design of a full-custom 2Gb/s 256*256 Crossbar Switch Fabric Core circuit, using TSMC 0.25μm CMOS technology. To cope with the high data link rate, conventional approaches use duplicated multiple bit-slices of the switch core to reduce the core delay requirement. However, this increases the area and limits the size of the crossbar switch. To cater for a large number of input and output ports of the switch, we propose a novel 3-stage pipelined MUX-tree based architecture. As a result, the problem of designing a 256*256 crossbar is reduced to a 128*128 sub-crossbar. The clock cycle time of the switch core can be reduced below 1ns. To achieve a 2Gb/s link rate, only two bit-slices are needed instead of 4 or 8 in the conventional designs. By doing so, we can put a 256*256 crossbar in a single chip. The layout of the 128*128 sub-crossbar was designed and simulated. Experimental results show that 1GHz clock frequency can be achieved. Furthermore a full 2Gb/s 64*64 crossbar switch digital core circuit was designed to demonstrate the whole pipeline structure. The area of this core is only 2.4mm*1.9mm and the power consumption is about 2.5W at 1GHz.

1. INTRODUCTION

The recent years' advances in fiber optic link technology, especially using Wavelength Division Multiplexing (WDM), have made raw bandwidth in abundance in data communication networks. It is possible to transmit 64 wavelengths at OC-192 on a single fiber link these days. As a result, switches/routers are replacing the transmission link as the bottleneck of the network. This is spurring a great demand for high-speed switches and routers that can scale to a large number of I/O ports. For example, switches as large as 128*128 or 256*256, running at 10Gb/s (OC-192) or even higher speed are becoming a necessity for most IP core networks. One of the challenges in designing such scalable switches is the core of the fabric backplane.

Among all of the switch core implementations and proposals, the crossbar (or crosspoint) backplane is proven to be a very good choice for performance [11,12]. In fact, most current high-performance switches and routers are based on the crossbar fabric. The structure of a crossbar switch is illustrated in figure 1. The non-blocking characteristic and the ability to support multicast make crossbar switches very attractive for real applications. Recently, a lot of research has been done on the design of the crossbar switch [1-6]. The main difficulty in designing a crossbar switch is the scalability of the switch for high line rates. In particular, for large number of input and output ports, it is difficult to achieve a high data rate as the wiring delay becomes the critical part of the circuit.

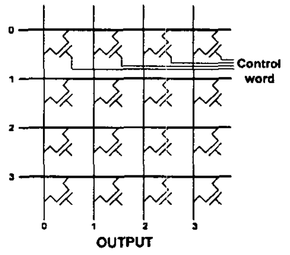

Figure 1. An X-Y based Crossbar Switch implementation

Most of the proposed designs for the crossbar switch digital core are assumed to be running at a lower speed. To cope with a high throughput requirement, multiple bit-slices of the core are then used. The input signals are first fed to a serial-to-parallel converter. The parallel data are then sent to the multiple-bit-slice core for the switching. After the switching, the outputs from the core are grouped and sent back to the high rate output link through a parallel-to-serial converter. By doing so, the switch core only needs to run at 250MHz [5] instead of 2GHz, for example. However the size of the switch is increased by a factor of 8 in this case. Even with this parallel switch cores, the core is still sometimes the bottleneck of the whole crossbar chip [5]. Also the increase in chip size hinders the scaling of the switch. For a 128*128 or 256*256 switch, the size of this multiple bit-slice approach may not be feasible.

In this paper we attempt to address the challenging issue of designing switch cores for large number of input/output ports. We propose a 3-stage pipeline architecture for a 256*256 switch so as to push the performance to the limit. By doing so, a 1GHz clock rate can be achieved even when 0.25μm technology is used. In addition, only 2 bit-slices are needed instead of 8.

The rest of the paper is organized as follows. The MUX-tree based crossbar design is introduced in Section 2. A novel 3-stage pipelined MUX circuit design is proposed and discussed in Section 3. In Section 4 we show the layout and the SPICE simulation results of the design. Finally we conclude the paper in Section 5.

2. CONVENTIONAL CROSSBAR DESIGN

Some of the previous crossbar switch implementations are summarized in Table 1 [1-5]. Here we just focus on the design using CMOS technology because of its high density, good reliability, lower power and low cost. It can be seen that most of the previous monolithic crossbar chips are limited to 64*64 with the exception of [3], in which the authors demonstrated the feasibility of designing a

Ref. Year	Crossbar size	Line rate	Technology	Circuit Technique
[1] 1988	64*17	90 Mb/s	1.0 μm CMOS	NAND-Inv Tree
[2] 1989	64*16	150 Mb/s	3.0 μm CMOS	2-to-1 MUX Tree
[3] 1992	256*256	100 Mb/s	1 μm BiCMOS	OR gate Tree
[4] 1994	32*32	250 Mb/s	1.2 μm CMOS	Reduced Swing X-Y
[5] 1999	32*32	1.6 Gb/s	0.27 μm CMOS	4-to-1 MUX Tree

Table 1. Summary of the previous crossbar switch designs.

256*256 crossbar from the area point of view. However, no actual implementation has been demonstrated. Also the line rate requirement in [3] is only at OC-3 (155.5Mb/s). If the line rate has to match with the current requirement (say OC-192, 9.953Gb/s), the performance due to the technology scaling may not help since the scaling is only improved by a factor of 16 (from 3μm to 0.18μm) while the performance requirement is increased by factor of 64. Therefore the architecture proposed in [3] may not be able to scale to the present requirements.

To reduce the delay requirement of the switch core, most of the conventional techniques use a bit-slice approach. In [5-6], 8 bit-slices are used. This requires a large die size and becomes the bottleneck of designing larger crossbar switch fabrics.

From Table 1, we can see that there are mainly two methods to implement the switch core [6], the X-Y based implementation as shown in Figure 1, and the MUX-tree based implementation as shown in Figure 2. Comparing these two design schemes, the tree-based implementation has obvious advantages over the X-Y based counterpart in terms of speed. For an $N*N$ crossbar, each input line connects to a long wire and N capacitors, in both implementations. However, the output of the crosspoint must drive N load capacitors in the X-Y based switch, while there is only one capacitor load in the tree based circuit output. Therefore the speed should be faster using the latter approach. In addition, using the X-Y based switch each crosspoint needs one control bit. As a result, the total number of control bits is N^2. For the tree based switch the control bits is $\log_2 N$ for each output column and thus the total number is equal to $N*\log_2 N$. Other advantages of the tree-based switch are discussed in detail in [6]. In addition to MUX tree approach, other variations such as the OR gate or NAND gate trees were also described in [3].

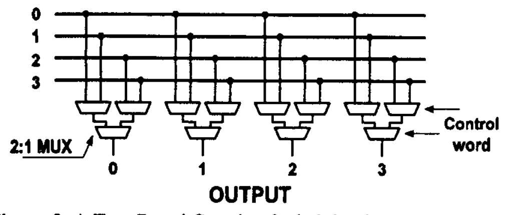

Figure 2. A Tree Based Crossbar Switch implementation.

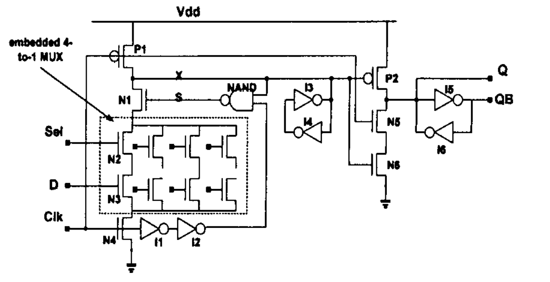

Figure 3. The schematic of the SDFF embedded with MUX [10]

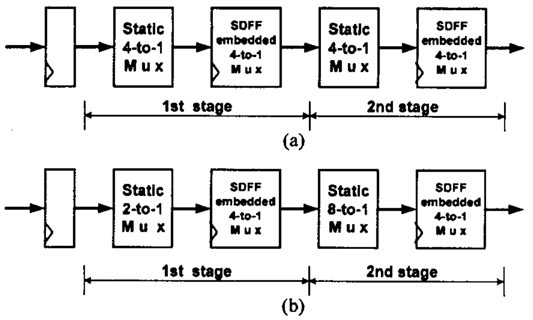

Figure 4. 2 stage Pipeline design of the 256-to-1 MUX.

3. PIPELINED MUX CROSSBAR CORE

To reduce the number of bit-slices required, we aim to achieve a 1GHz clock rate for the 256*256 switch core. This is nearly the maximum clock frequency practically achievable with our 0.25μm CMOS technology [7]. To satisfy this delay requirement, pipelining of the 256-to-1 MUX tree is required. Hence, high-speed circuit techniques have to be used in designing the pipelined circuit. To achieve a sub-ns cycle time, we employ a semi-dynamic Flip-Flop (SDFF) design [9,10], which is one of the fastest flip-flops due to its negative setup time characteristic. The MUX can be embedded inside the SDFF so that the delay time of the MUX itself can be absorbed by the negative setup time and the delay time of the SDFF [10]. Figure 3 shows the schematic of a SDFF embedded with a 4-to-1 MUX, which is the basic memory component used in our design.

It is proved in [8] that the optimal design of a multiplexing circuit is to use a tree of 4-to-1 MUXs. For the 256*256 core, it is natural to partition the MUX-tree to 2 pipeline stages, each with one static 4-to-1 MUX and one SDFF embedded with 4-to-1 MUX, as shown in Figure 4 (a). However, in the first stage, the inputs need to drive a large number of gate capacitances, amounting to 256 in our case. To make the matter worse, the wire that connects these capacitances needs to run across the whole chip, and the delay due to the long wire is substantial. One of the ways to tackle this problem is to change the static 4-to-1 MUX to a static 2-to-1 MUX in the first stage and push the extra 2-to-1 MUX to the later stages, as shown in figure 4 (b). To model the input line more accurately, we use a distributed RC model as illustrated in figure 5. The delay of the first stage is given by

$$T_{1st\text{-}stage} = T_{CLK0\text{-}Q0} + T_{Driver} + T_{Wire} + T_{Static2\text{-}to\text{-}1MUX}$$

Note that the delay time of the 4-to-1 MUX has been absorbed in the negative setup time of the next SDFF.

To reduce the delay due to the wire, repeaters can be inserted to divide the long wire into several shorter sections. However the repeater itself also introduces additional delay overhead and there exists an optimum number of repeaters for a wire. We did HSPICE simulations to find out the optimal number of repeaters. The simulation was done with the wire length obtained from the estimation based on the actual MUX cell design and a careful floorplan of the whole switch. Simulations were done for different wire widths, repeater sizes and repeater numbers. Simulation results for different port numbers (N=64, 128, 256) are shown in Figure 6. It can be shown that for N = 64 and 128, adding a repeater only increases the delay time. For N=256, the optimal number of repeaters is 1. Even when the repeater is added, we can see that the delay is increased from 0.908ns to 1.254ns when N is increased from 128 to 256. The 1ns delay requirement cannot be achieved for N=256.

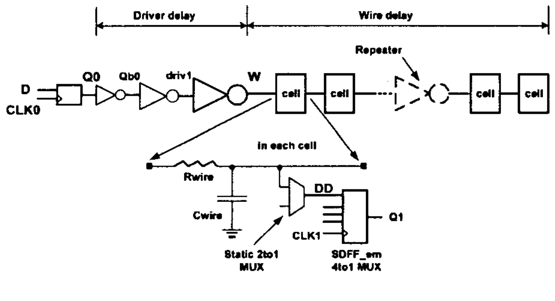

Figure 5. Model of loading at the input wire

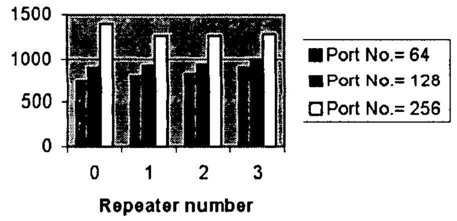

Figure 6. The optimized delay time of the first stage vs. the repeater number

To meet the target of 1 ns cycle time, we propose a novel 3-stage pipeline architecture for the 256*256 crossbar switch core. Instead of adding a repeater, a flip-flop is inserted in the middle of the long wire. By doing so, we not only decrease the wire length by a factor of 2, but also add a whole stage for driving the long wire. At each stage the input is only driving 128 inputs. In the first cycle, the input is driving the first 128 MUX inputs. At the same time, it is driving through the long wire and send to the flip-flop which is used to store the input values for the driving of the second group of 128 inputs in the next cycle. Essentially we split the 256-to-1 MUX into two 128-to-1 MUXs. A flip-flop is added at the output of the first 128-to-1 MUX to make sure the outputs of the two MUXs are synchronized. The outputs are then fed to a SDFF embedded with a 2-to-1 MUX to obtain the final 256-to-1 MUX output.

The floorplan of the whole crossbar core is shown in Figure 7. It consists of four 128*128 crossbars, which we call the sub-crossbars 0-3 in the figure. It can be seen that the problem of designing a 256*256 switch core is reduced to a 128*128 core. Each sub-crossbar is pipelined in two stages, a 8-to-1 MUX stage and a 16-to-1 MUX. From figure 6, it has been shown that the 1st stage delay time for the 128*128 sub-crossbar is less than 1ns. The second stage delay is not as critical as the first stage and we sized the MUXs such that the delays of the two stages are balanced.

The timing diagram of the proposed 3-stage pipelined crossbar switch core is shown in figure 8. It shows the timing diagram for the output port 0 as an example. In the first two cycles, the inputs [0:127] are switching in the sub-crossbar 0. In the first cycle, inputs [128:256] are driving through the wire and stored in the flip-flops. In the next two cycles, they are sent to the sub-crossbar 3 for the switching action. The outputs of the sub-crossbar 0 are delayed for 1 cycle to synchronize with the output from sub-crossbar 3. Finally the outputs of sub-crossbar0 and sub-crossbar3 are sent to a SDFF embedded with 2-to-1 MUX to complete the 256-to-1 MUX function. The operation for the other output ports is similar.

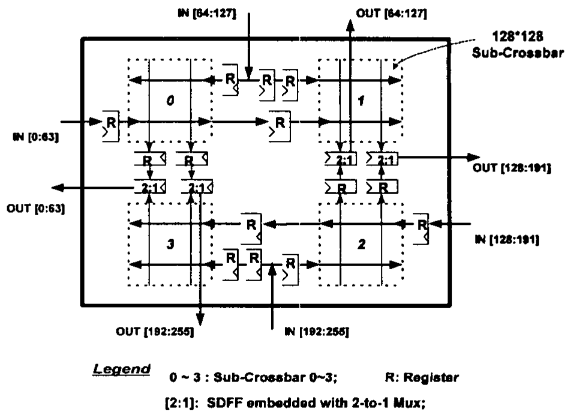

Figure 7. Floorplan of the proposed 3-Stage pipeline crossbar core (Each line represents 2 bits).

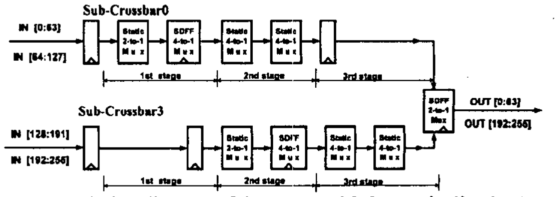

Figure 8. Timing diagram of the proposed 3-Stage pipelined MUX

4. LAYOUT AND SIMULATION RESULTS

To verify the proposed architecture, we designed the layouts for the 32*32 and 128*128 sub-crossbars using the TSMC 0.25μm SCN5M_DEEP (lambda=0.12) technology. The layout of the 128*128 sub-crossbar is shown in Figure 9. It includes two bit-slices so that a link rate of 2Gb/s is supported. The layout size and the post layout HSPICE simulation results are summarized in Table 2. The simulations are done in the typical process corner, normal condition and with a 2.5V supply voltage. The delays are the clock to the output timing. In figure 10, we show the timing waveform of the first stage in the 128*128 sub-crossbar. D, W, DD and Q1 are the waveforms of the corresponding points in Figure 5. The three different waveforms DD_1, DD_m and DD_last represent the outputs of the static 2-to-1 MUX in the first, the middle and the last cells along the wire, respectively. The waveforms of Q1_1, Q1_m, Q1_last are their corresponding outputs after the SDFF that is embedded with the 4-to-1 MUX. The delay from the clock to DD_last is about 959ps. To show the zero/negative setup time characteristic of the SDFF, we run the clock at 1.06GHz, i.e., with a 959ps cycle time. The waveforms show that even with zero setup time, the circuit is still working correctly.

Sub-crossbar		32*32	128*128
Layout size		1.0mm*0.65mm	5.9mm*3.1mm
Post-layout simulation delay time	1st stage	572ps	959ps
	2nd stage	476ps	866ps

Table 2. The 32*32 and 128*128 sub-crossbars size and delay

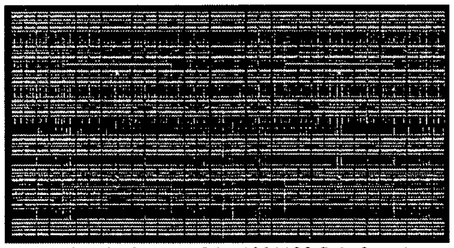
Figure 9. The layout of the 128*128 Sub-Crossbar

To demonstrate the floorplanning and the 3-stage pipeline of the crossbar switch in figure 7, we completed the full design of a 2Gb/s 64*64 crossbar switch based on 32*32 sub-crossbar. The layout is shown in Figure 11. The controllers, which generate the control bit with the appropriate drivers, are placed between the sub-crossbars. We use a reverse tag scheme similar to that used in [5]. The tags within the port-*k* carry the source of the packet that will be sent to port-*k* instead of the destination of port-*k*. Therefore 8 bits in the header are enough in the multicast case. The controller interprets the tagged information at the header and generates the corresponding control signal to configure the switch. Table 3 summarizes the specification of the 64*64 crossbar core.

5 CONCLUSIONS

In this paper, we proposed a novel 3-stage pipelined MUX design for a 2Gb/s 256*256 crossbar switch fabric. The problem of designing a 256*256 crossbar switch fabric is reduced to a 128*128 sub-crossbar. 1 GHz switching frequency is achieved for the core. To support 2Gb/s link rate, only 2 bit-slices are needed. Compared with conventional crossbar designs, the number of bit-slices required is reduced and significant area saving is achieved.

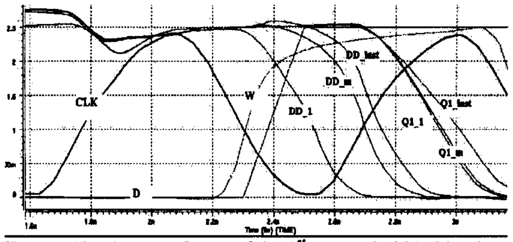

Figure 10. The waveforms of the 1st stage of 128*128 sub-crossbar

Supply voltage	2.5 V
Technology	TSMC 0.25μm SCN5M_DEEP
Layout size	2.4mm*1.9mm
Transistor count	100k
Power @ 1GHz	2.5W

Table 3. The specification of 64*64 Crossbar Core with 2bits

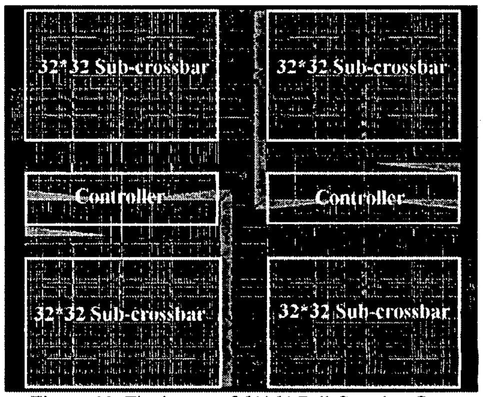

Figure 11. The layout of 64*64 Full Crossbar Core

6. REFERENCES

[1] F.Barber et.al. "A 64*17 non-blocking crosspoint switch", IEEE International Solid-State Circuits Conference, pp116-7, 1988

[2] M.Cooperman, et.al "A single chip 64*16 broadband switch", IEEE International Symposium on Circuits and Systems, pp230-3, 1989

[3] K.Choi and W.S.Adams, "VLSI implementation of a 256*256 crossbar interconnection network", Proceedings of Parallel Processing Symposium, pp289-293, 1992

[4] R.Golshan et.al"A novel reduced swing CMOS bus interface circuit for high speed low power VLSI systems" IEEE International Symposium on Circuits and Systems, pp351-4, 1994

[5] K.Chang et.al "A 50Gb/s 32*32 CMOS crossbar chip using asymmetric serial links", Symposium on VLSI circuits, pp19-22 1999

[6] M.Cooperman and P.Andrade " CMOS giabit-per-second switching" IEEE Journal of Solid-State Circuits, Vol.28, No.6, pp631-639, Jun.1993

[7] M.Horowitz et.al "High-speed electrical signaling: overview and limitations" IEEE Micro, Vol.18, No.l, pp12-24, Jan.1998

[8] I.Sutherland, et.al, Logical effort: designing fast CMOS circuits, Morgan Kaufmann Publishers, 1999

[9] V.Stojanovic and V.G.Oklobdzija, "Comparative analysis of master-slave latches and Flip-Flops for high-performance and low-power systems", IEEE Journal of Solid-State Circuits, Vol.34, No.4, pp536-548, April1999

[10] F.Klass "Semi-Dynamic and Dynamic Flip-Flops with embedded logic" Symposium on VLSI circuits, pp108-9, 1998

[11] T.Koinuma and N.Niyaho, "ATM in B-ISDN Communication systems and VLSI realization" IEEE Journal of Solid-State Circuits, Vol.30, No.4, pp.341-347, April 1998

[12] C.E.Chang et.al "High Speed Crosspoint Switches", International Journal of High speed electronics and systems, Vol.9, No.2, pp505-548, June 1998