Fast three-dimensional programmable two-selector

J.-S. Ahn, D.-K. Jeong and S. Kim

A proposed three-dimensional programmable two-selector (3DP2S) selects two topmost requests from a set of requests, using little more delay time than a programmable priority encoder selecting only one. Simulation results show that a 3DP2S can be used for scheduling combined input–output queuing switches with multiple parallel crossbar planes, without requiring faster components.

Introduction: Recently, combined input–output queuing (CIOQ) switches have been widely explored [1]. By using faster versions of some or all of their components it is possible to make CIOQ switches show better characteristics, while retaining the ease of implementation of input-queuing (IQ) switches. It was theoretically shown that increasing their speed by a factor of 2 is good enough to allow CIOQ switches to emulate output-queuing (OQ) switches in a wide variety of scheduling disciplines. However, the complex scheduling algorithm makes it hard to build the scheduler that is as fast as the schedulers for IQ switches that use virtual output queues (VOQs) and programmable priority encoders (PPEs) [2]. Although there are CIOQ switches that use an IQ scheduler using VOQs and sacrifice exact emulation of an OQ switch for ease of implementation, such switches cannot guarantee the same service quality. However, running at double the speed, they are good for most situations, matching the performance of OQ on average. One of problems with such switches is the speed of memory required for their VOQs. While logic speed clearly tracks technological advance, memory speed generally lags, and, thus, it is not easy to build a memory that is fast enough for CIOQ switches.

In this Letter we propose a three-dimensional programmable two-selector (3DP2S), which eliminates the need for faster VOQ memory when used in place of a PPE in a CIOQ scheduler. When a PPE is used in a CIOQ scheduler, the same request can be used twice within a cell time for VOQ memory, requiring double the speed. However, a 3DP2S always selects two different requests simultaneously, which ensures at most one read access to a VOQ during a cell time, eliminating the need for faster memory and thus enabling the design of high-performance switches.

Structure and operation: Basically, the 3DP2S is a three-dimensional expansion of the ripple PPE (RP) described by Gupta and McKeown [2]. Since the first and second requests are closely related, a simple 3D expansion of PPEs is not possible. First, we build a planar RP, with a two-dimensional structure for higher speed. We then stack one planar RP on top of another and bind them with glue logic that passes intermediate results from the first to the second. The top plane processes the first request and the lower plane the second.

We now describe the structure and operation of the 3DP2S, beginning with the structure and operation of the RP, which is relatively simple. Starting from the unit block shown in Fig. 1*a*, we can construct the RP of Fig. 1*b* by connecting n unit blocks cyclically. The requests are propagated through the unit blocks. The topmost request to a unit block indicated by the priority pointer P_dec can override all the subsequent requests.

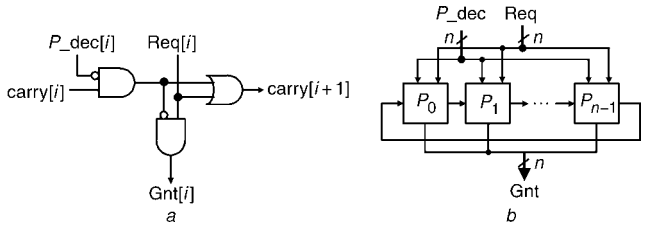

Fig. 1 *Unit block from ripple PPE and design of ripple PPE*

a Unit block
b Design

The RP has two major problems: an O(n) latency and the existence of timing loops that cannot be handled properly in CAD tools. To solve these problems, as shown in Fig. 2*a*, a planar RP is proposed which is structured as multiple stages. Carries from one stage are propagated to the adjacent stage, eliminating timing loops. Figs. 2*b* and *c* show a unit block of Fig. 2*a* and the connection scheme for the unit blocks. $P_{i,j}$ denotes the unit block that corresponds to the jth request at stage i ($1 \le j < n$, $1 \le i \le \lceil \log_2 n \rceil$). The inputs to the planar RP are the request R_j and the priority pointer P_dec_j; they are connected to $P_{1,j}$ and $P_{1,(j+1)}$. Note that, in Figs. 2*a* and *d*, the connections for P_dec are omitted for clarity. Fig. 2*d* shows that a carry, indicated as 1 or 0 in the boxes, from one stage creates two requests to the next stage, propagating them like a binary tree to override all subsequent requests after $\lceil \log_2 n \rceil$ stages. The propagated priority pointer, on the other hand, protects the higher priority requests from being overridden. After $\lceil \log_2 n \rceil$ stages, the request with the highest priority will override all other requests, and the result will be a thermometer-coded vector with the topmost request at the top. The last stage, denoted by D_i, singles out this topmost request by detecting 0 to 1 transition in the vector. The circuit design for D_i is trivial and is omitted for brevity. The proposed planar RP has O($\log_2 n$) latency and O($n \log_2 n$) complexity.

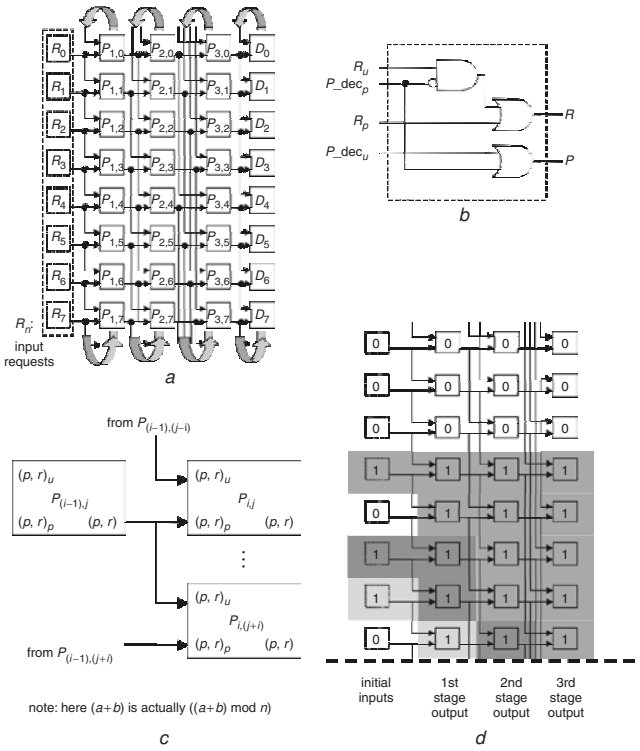

Fig. 2 *Multiple stages of ripple PPE; unit block $P_{i,j}$ from Fig. 2a; connection scheme between unit blocks; propagation of input signals*

a Multiple stages
b Unit block $P_{i,j}$ from Fig. 2*a*
c Connection scheme
d Propagation of input signals

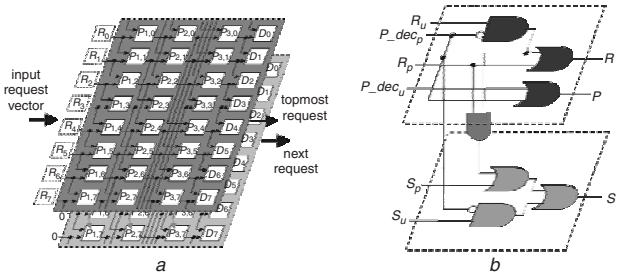

Fig. 3 *Proposed architecture of 3DP2S and unit block from Fig. 3a*

a Proposed architecture
b Unit block from Fig. 3*a*

The 3DP2S uses two of planar RPs, stacked one on top of another, as shown in Fig. 3*a*. The upper plane accepts the request vector to select the initial request, and the lower one receives the intermediate results between stages of the upper plane and propagates candidates for the next request. After the same $\lceil \log_2 n \rceil$ stages, the lower plane also yields a thermometer-coded vector, with the second request at the top.

The structure of unit blocks of the 3DP2S in the upper and lower planes is shown in Fig. 3*b*. Each half of the unit block uses basically the same circuits as Fig. 2*b*, except that the circuits for *P*_dec are shared by both planes. The planes are bound together by glue logic. The *S* signals in the lower plane have the same connection scheme as the *R* and *P*_dec signals in the upper plane. The 3DP2S also has $O(\log_2 n)$ latency and $O(n \log_2 n)$ complexity as the planar RP.

Evaluation: Table 1 shows the worst-case critical-path delays of the PPE, the planar RP, and the 3DP2S of 64 requests; all were implemented with a 0.18 μm CMOS technology, using commercial synthesis and P&R tools with the same settings. The PPE uses the architecture with thermometer-coded masks described by Gupta and McKeown [2], with the priority pointer indicated by a thermometer-coded vector, while the planar RP and 3DP2S use a decoded vector for their priority pointers.

Table 1: Worst-case latency of PPE, planar RP and 3DP2S implemented in 0.18μm CMOS technology (width = 64 bits)

Component	PPE (with thermometer encoding)	Planar RP	3DP2S
Worst-case delay (in ns/requests)	1.31 ns/1 request	1.41 ns/1 request	1.75 ns/2 requests

The results show that the 3DP2S requires only one-third more delay time to process two requests than the PPE. All effects of increased number of gates and connections are included in the results. The results would be more dramatic as we further increase the size of the switches.

Conclusions: The three-dimensional programmable two-selector proposed in this Letter selects two topmost requests from *n* inputs with priorities indicated by a pointer. With a parallel architecture, it can compute results in comparable time to a programmable priority encoder with the same input width. With a modular structure, it can be expanded easily and has $O(\log_2 n)$ latency and $O(n \log_2 n)$ complexity.

21 June 2004
Electronics Letters online no: 20045935
doi: 10.1049/el:20045935

J.-S. Ahn, D.-K. Jeong and S. Kim (*School of Electrical Engineering and Computer Science/Electrical Engineering, Seoul National University, San 56-1 Shinlim-dong, Kwanak-gu, Seoul, Korea*)

E-mail: jsahn@ee.snu.ac.kr

References

1 Chuang, S., Goel, A., McKeown, N., and Prabhakar, B.: 'Matching output queuing with a combined input/output-queued switch', *IEEE Trans. Commun.*, 1999, **47**, (8), pp. 1030–1039
2 Gupta, P., and McKeown, N.: 'Designing and implementing a fast crossbar scheduler', *IEEE Micro*, 1999, **19**, (1), pp. 20–28