Buffer sizing problems in networks and asynchronous circuits: similarities and differences

Rajit Manohar
Yale University
rajit.manohar@yale.edu

Robert Soulé
Yale University
robert.soule@yale.edu

ABSTRACT

Determining the optimal buffer size is a problem that arises in both networking and asynchronous digital circuits. However, to date, there has been little discussion between the two communities. We believe it would be worthwhile to see if the techniques independently developed by researchers in each of the domains could apply to the other. We hope this paper is the beginning of a fruitful conversation, and could lead to advances that benefit both communities.

1 INTRODUCTION

Asynchronous digital circuits correspond to computations that operate without an external clock signal to synchronize operations. Instead, circuits interact through local handshaking signals on wires and these signals are used for both communication and synchronization. Asynchronous circuits have a rich history, and many of the mathematical theories used to study them have their roots in the theory of concurrent systems. For example, Petri nets are a popular model for describing asynchronous control circuits [17]. Trace theory has been used for circuit verification as well as studying various classes of asynchronous circuits [6, 20]. Recently, key concepts from the theory of distributed systems related to knowledge and causality were translated into equivalent results for asynchronous circuits [12].

High-level descriptions of asynchronous circuits use conventional message-passing programming language notation [15]. Interactions between hardware compoments occur using communication channels that have first-in first-out semantics. Since circuits have a physical realization, these channels have finite buffer space (zero by default, so as to make the actual physical need for storage explicit). The amount of buffer space on a channel is referred to as the synchronization slack [14], and this slack has to be carefully adjusted for optimal performance through a process referred to as slack-matching [13]. Slack matching has been formulated as a mixed-integer linear programming problem [2, 19].

The analogy between asynchronous circuits and networks has been previously exploited to develop a fast hardware accelerated simulator for wireless networks. Asynchronous circuits were configured to implement the same functionality as the underlying wireless network being modeled, and were shown to replicate all the key network properties of wireless networks such as throughput, goodput, latency, etc. since the behavior of the underlying circuits was fundamentally the same as the network being modeled [11]. This strong correspondence between asynchronous circuits and distributed systems, and the interest in buffer sizing (slack matching) in both communities begs the question: can techniques from one domain be applied to the other? What keys insights that exist in the circuit domain can be translated into insights in the networking domain, and vice versa?

2 TECHNIQUES FROM CIRCUITS

Much of the work in the domain of asynchronous circuits has focused on timing characterization and performance analysis. Buffers tend to be small, since each buffer has to be physically realized as a circuit. The key goal is minimizing the number of buffers needed while operating at a performance point that meets the system throughput requirement.

The performance behavior is characterized by studying the number of data items in flight in a pipeline section that has a fixed peak occupancy. The behavior of the throughput as a function of the number of data items in flight can be explained intuitively in the following manner: (i) when there are no data items in flight, the throughput is zero; (ii) as data items in flight increase, the throughput increases linearly until it hits the maximum local throughput of the pipeline; (iii) at the other extreme, when the number of data items in flight equals the pipeline capacity, the throughput is also zero—because there has to be an open slot (a "hole") in the pipeline in order for a new data item to be inserted; (iv) as the number of holes increases, the throughput also increases until it hits the pipeline throughput. The overall shape of the throughput curve looks like a trapezoid, with the pipeline section operating at peak throughput for a range of occupancy values. This curve is sometimes called a "canopy graph." Constraints on token flow can be used to compose these canopy graphs together to build system canopy graphs.

In general, the entire performance optimization problem can be formulated using a collection of inequalities that capture timing constraints [4], and under a general set of conditions the throughput of the circuit can be bursty but still exhibits periodicity [10]. Both the period and the repeat interval can be computed using efficient algorithms when all the delay values for the entire circuit are known.

3 TECHNIQUES FROM NETWORKING

Much of the prior work in the domain of networking has focused on sizing buffers for internet routers. Historically, it was assumed that buffers should be big, with the sizing based on flow round-trip time measurements. The rule-of-thumb was traditionally $B = \overline{RTT} \times C$, where $\overline{RTT}$ is the average round-trip time of a flow passing across the link, and C is the data rate of the link. Appenzeller et al. [1] argued that buffers could be much smaller, proposing that a link with n flows requires no more than $B = (\overline{RTT} \times C)/\sqrt{n}$. Enachescu et al. [7] proposed even smaller sizes still, arguing that only 20-50 packets are necessary if we are willing to sacrifice a fraction of link capacities. Choudhury and Hahne [5] proposed an adaptive scheme that allocates buffers among queues sharing memory. They use a control-theoretic approach based on monitoring the total amount of unused buffer space.

Although there have been studies evaluating the different proposals [3], there is still no clear consensus on what is the optimal buffer size. Partly, this is because the buffer sizing problem in networking is complicated by a number of issues, including the congestion control scheme, the Active Queue Management discipline (e.g., RED [8], PIE [18], etc.), load-balancing, and traffic-engineering policies. Since these issues are not relevant to asynchronous circuits, there is reason to be skeptical that techniques from asynchronous circuits would directly apply.

However, we believe that some of these differences might disappear if we make some stronger assumptions about the use and behavior of network. For example, if we limit the scope to dedicated storage fabrics in a data center that use lossless transport protocols (e.g., Infiniband, iWARP, or RoCE) with relatively simple congestion control schemes [9, 16], there may be insights that can be gleaned from asynchronous circuits.

In any case, we believe that there should be further analysis and evaluation to understand the similarities and differences between techniques from both domains; when they could be productively used; and when they would not apply.

4 SUMMARY

Given the strong similarity between the performance properties of asynchronous circuits and networks, we believe it would be worthwhile to see if the techniques independently developed by researchers in each domain could apply to the other. We hope this paper is the beginning of a fruitful conversation, and could lead to advances that benefit both communities.

REFERENCES

[1] Guido Appenzeller, Isaac Keslassy, and Nick McKeown. 2004. Sizing Router Buffers. In *SIGCOMM*. 281–292. http://doi.acm.org/10.1145/1015467.1015499

[2] Peter A Beerel, Andrew Lines, Mike Davies, and Nam-Hoon Kim. 2006. Slack matching asynchronous designs. In *IEEE ASYNC*. IEEE.

[3] Neda Beheshti, Yashar Ganjali, Monia Ghobadi, Nick McKeown, and Geoff Salmon. 2008. Experimental Study of Router Buffer Sizing. In *SIGCOMM IMC*. 197–210. http://doi.acm.org/10.1145/1452520.1452545

[4] S M Burns and A J Martin. 1990. Performance Analysis and Optimization of Asynchronous Circuits. In *IEEE ARVLSI*.

[5] Abhijit K. Choudhury and Ellen L. Hahne. 1998. Dynamic Queue Length Thresholds for Shared-memory Packet Switches. *IEEE/ACM TON* 6, 2 (April 1998), 130–140. http://dx.doi.org/10.1109/90.664262

[6] David L Dill. 1989. *Trace theory for automatic hierarchical verification of speed-independent circuits*. Vol. 24.

[7] Mihaela Enachescu, Yashar Ganjali, Ashish Goel, Nick McKeown, and Tim Roughgarden. 2005. Part III: Routers with Very Small Buffers. *SIGCOMM CCR* 35, 3 (July 2005), 83–90. http://doi.acm.org/10.1145/1070873.1070886

[8] S. Floyd and V. Jacobson. 1993. Random early detection gateways for congestion avoidance. *IEEE/ACM TON* 1, 4 (Aug 1993), 397–413.

[9] Chuanxiong Guo, Haitao Wu, Zhong Deng, Gaurav Soni, Jianxi Ye, Jitu Padhye, and Marina Lipshteyn. 2016. RDMA over Commodity Ethernet at Scale. In *SIGCOMM*. 202–215. http://doi.acm.org/10.1145/2934872.2934908

[10] Wenmian Hua and Rajit Manohar. 2017. Exact timing analysis for asynchronous systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 37, 1 (2017), 203–216.

[11] Rajit Manohar and Clinton Kelly IV. 2001. Network on a chip: modeling wireless networks with asynchronous VLSI. *Communications Magazine, IEEE* 39, 11 (2001), 149–155.

[12] Rajit Manohar and Yoram Moses. 2015. Analyzing Isochronic Forks with Potential Causality. In *IEEE ASYNC*. IEEE, 69–76.

[13] A Martin, A Lines, R Manohar, M Nystrom, P Penzes, R Southworth, U Cummings, and Tak Kwan Lee;. 1997. The design of an asynchronous MIPS R3000 microprocessor. 164–181.

[14] Alain J Martin. 1981. An axiomatic definition of synchronization primitives. *Acta Informatica* 16, 2 (1981), 219–235.

[15] Alain J Martin. 1986. Compiling communicating processes into delay-insensitive VLSI circuits. *Distributed computing* 1, 4 (1986), 226–234.

[16] Radhika Mittal, Alexander Shpiner, Aurojit Panda, Eitan Zahavi, Arvind Krishnamurthy, Sylvia Ratnasamy, and Scott Shenker. 2018. Revisiting Network Support for RDMA. In *SIGCOMM*. 313–326. http://doi.acm.org/10.1145/3230543.3230557

[17] Tadao Murata. 1989. Petri nets: Properties, analysis and applications. *Proc. IEEE* 77, 4 (1989), 541–580.

[18] R. Pan, P. Natarajan, C. Piglione, M. S. Prabhu, V. Subramanian, F. Baker, and B. VerSteeg. 2013. PIE: A lightweight control scheme to address the bufferbloat problem. In *IEEE HPSR*. 148–155.

[19] Piyush Prakash and Alain J Martin. 2006. Slack matching quasi delay-insensitive circuits. In *IEEE ASYNC*. IEEE, 10–pp.

[20] Jan LA Van de Snepscheut. 1985. *Trace theory and VLSI design*. Number 200.