Credit-based Communication in NoCs

Rostislav (Reuven) Dobkin

049017 - Introduction to Networks on Chips: VLSI aspects

Abstract

Credit-based communication is widely employed in networks of different types for performance enhancement, packet loss prevention and deadlock prevention. In this work we survey different crediting techniques in context of network-on-chip (NoC) and propose a classification of the different crediting approaches.

Keywords: credits, flow control, piggybacking, backpressure.

1. Introduction

Networks-on-chip (NoCs) are designed to provide high bandwidth and parallel communication, while minimizing the number of employed interconnect resources. However, NoC performance can significantly degrade in absence of an effective flow control mechanism. The flow control is responsible for preventing resource starvation and congestion inside the network. This goal is achieved by managing the flow of the packets that compete for resources, such as links and buffers.

The flow control can be either *centralized* or *distributed*. In most of the NoCs architectures the distributed approach is preferred, when the flow control decisions are taken inside the NoC routers (Figure 1). The routers employ bi-directional interfaces, constructed of input and output ports as in Figure 2. Usually a packet entering through an IP does not loop back, and thus each IP is connected to four OPs (Figure 2) and only two bits are required to control switching. The OP emits flow control unit (flits) according to their arrival order, their priority and *creditability*.

The main concept of credit-based communication is to enable data forwarding only when there is a free data space in the destination buffer. Once there is no data space at the destination, the data should not be moved, enabling other data to utilize the communication resources.

An example of crediting is shown in Figure 3: The output port provides a single stage of buffering for each input port connection. The flits in these buffers arbitrate for the link to the input controller on the next router. Credits for buffer allocation are piggybacked on flits traveling in the reverse direction [4]. Note that in this example the credits flow *inside* the router, smoothing the operation between the input and output ports.

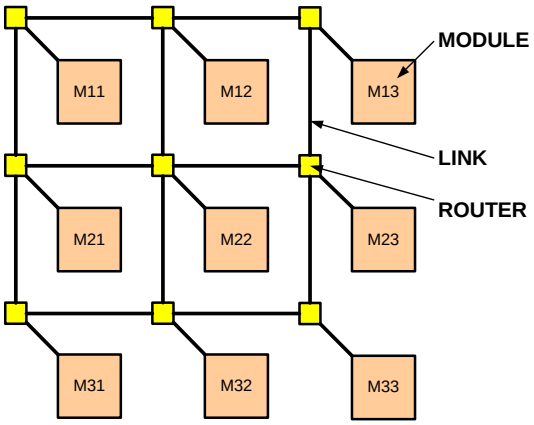

Figure 1: QNoC 2D Mesh Architecture

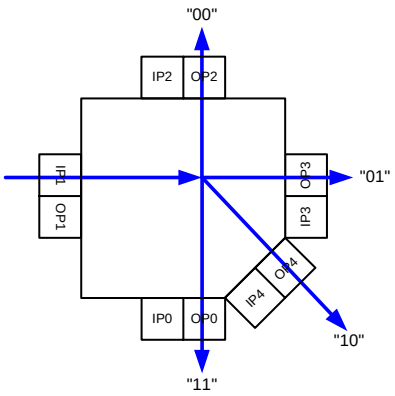

Figure 2: Routing Address from Source to Sink [5]

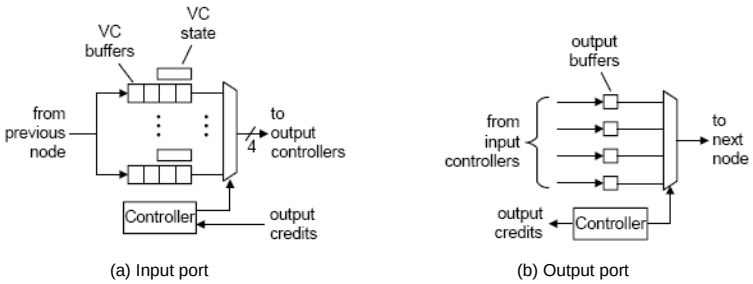

Figure 3: Input and output port controllers [4]

2. Classification

In the NoC domain, the term flow control has been used in the context of *intra-switch* [4], *switch-to-switch* [13] [1][5] or *end-to-end* [3][6] transport protocols (Figure 4). These protocols provide a smooth traffic flow by preventing buffer overflow and packet drops.

Intra-switch credits are employed inside routers, allowing correct communication in between the router input and output ports (Figure 3). This is an inherent part of the router structure and is usually implemented as a pipeline backpressure protocol.

Switch-to-switch credits are employed in between adjacent routers and between network interface (NI) and routers. The credits control the flow in between the routers indicating for the output ports the buffer space availability inside the adjacent input ports. Once an output port runs out of credits, it will cease sending flits, allowing link utilization by other (e.g. lower priority) packets.

End-to-end credits relate to global communication, when the source and destination tiles (modules) agree on a data transfer (Figure 5). The credits can either relate to an entire packet transfer or to a partial (chunk) data transfer. Note that this type of credits directly controls the amount of data injected into the NoC and thus directly affects the NoC load.

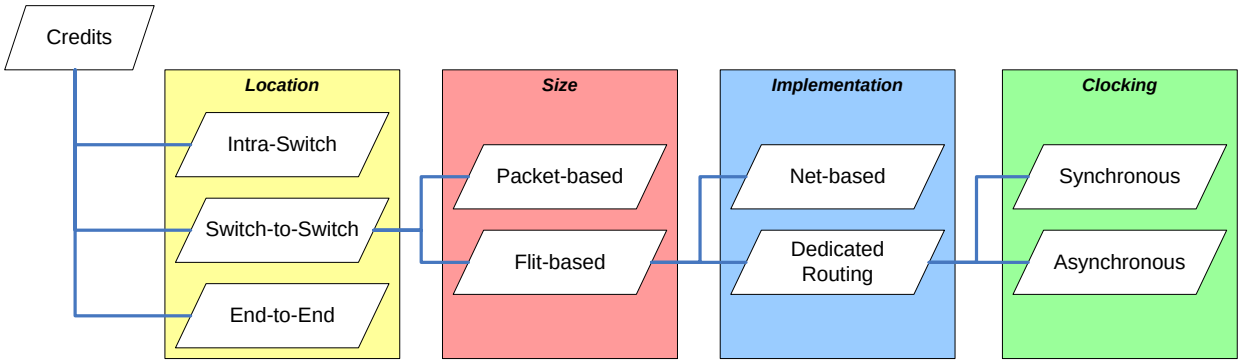

Figure 4: Classification

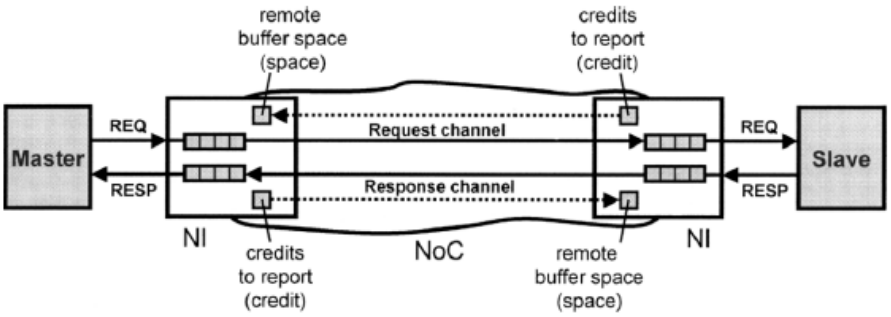

Figure 5: End-to-end credits (the credits are transferred in between network interfaces) [3]

The three protocols relate to three hierarchy levels and can be employed concurrently in the same NoC. In this work we focus on the *switch-to-switch* credit mechanism, while we note that the reviewed implementation principles can be also applicable for the other hierarchical levels.

At the next step we distinguish between different amounts of data that can be credited. In other words, single credit can relate either to a single flit or to a multiple-flit (chunk) transfer. Moreover, a single credit can be defined for different abstraction levels: flits and packets (Figure 4). In NoCs, due to the limited buffer space, the packets are not moved entirely between the routers, and therefore flit-based crediting is the common one. However, single credit can still indicate space availability for more than one flit (chunk).

There are also a few options for choosing the media for communicating the credits. The credits can be sent through dedicated-routing resources or through existent routing resources (Figure 4). The dedicated routing resources incur additional HW penalties, however, may provide better performance, while sending the credits through the existing routing resources (through the NoC itself, along with regular flits), may load the network and degraded the overall performance.

At the realization stage the credit mechanisms can be implemented either *synchronously* or *asynchronously (*Figure 4). In the synchronous implementation the crediting mechanism *must* be explicitly defined in between the routers in order to prevent loosing flits. In the asynchronous implementation, on the other hand, the asynchronous protocol employed in between the routers provides initial crediting, preventing any data loss. The asynchronous protocol enables full backpressure inherently, while in the synchronous implementation an additional effort is required for the backpressure implementation.

Unfortunately, the backpressure prevents only the flit / packet dropping, while the router / network performance can still degrade due to unwise arbitration inside the router itself. In Figure 6 we show an example, where the arbiter pushes a red flit into the output (merge) stage of the OP, when there is no place to push it forward (the corresponding buffer inside the input port is full). Thus, the connection is staled until the buffer inside the IP is emptied. If the arbiter could know that the green buffer at the IP side is free, it would have arbitrated out a green flit instead of the red one, thus enhancing the utilization of the NoC link.

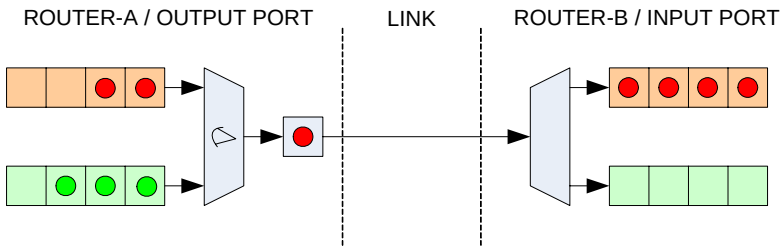

Figure 6: Example: Stalls w/o crediting. Backpressure is not enough.

In the following sections we survey different credit techniques for passing the buffer space availability information between adjacent routers. Credit based communication can be seen as an extension of a handshake protocol, wherein the transmitting circuit, after sending a message, waits for a return signal from the receiving circuit that free space for a next message has become available. Compared to such a handshake protocol, credit based communication supports a higher data rate, because the transmitting circuit can decide to transmit a next message (as well as about the size of the message) on the basis of local information, generally without waiting for a next message about the release of buffer space.

3. Crediting protocols and implementations survey

3.1. General approach

In this section we review the approaches for crediting inside the NoC routers. A general crediting scheme is shown in Figure 7, where at the transmitter side there is a counter (space) tracking the empty buffer space of the next router IP queue. This counter is initialized with the IP buffer size. When data is sent from the transmitter queue, the counter is decremented. When data is consumed from the IP buffer, credits are produced in a counter to indicate that more empty space is available. These credits are sent to the transmitting routers (red line in Figure 7) and are added to its space counter.

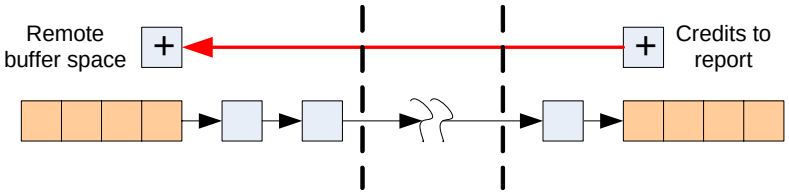

Figure 7: A general crediting scheme

The crediting mechanism always works hand in hand with arbitration, providing additional information to the arbiter that then could come up with a more correct decision. An example is shown in Figure 8, where in the Æthereal router [3] the data availability is first validated with creditability of the destination and the *request generator* produces request only for the channels that both have available data to send and free space at the destination side.

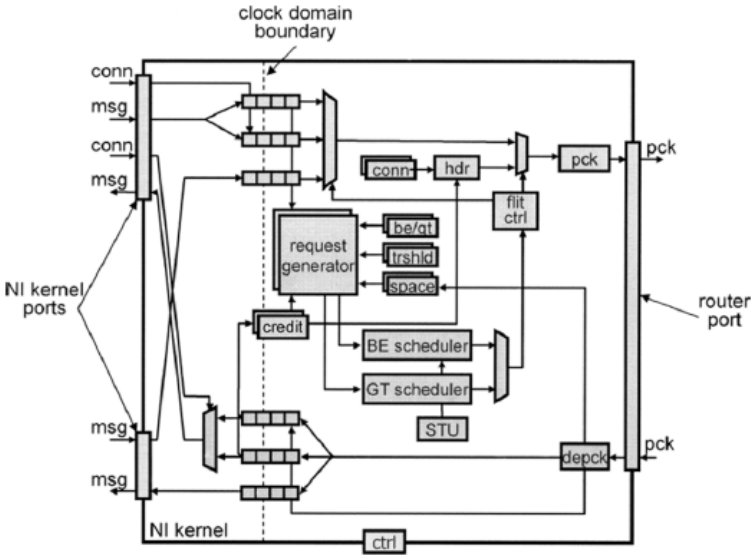

Figure 8: Æthereal router [3]. Request generator result depends both on availability of data from the input channel and on credit availability from the destination

3.2. CHAIN

We start the survey of the published crediting solutions with CHAIN [7][8] architecture (Figure 9). The packet formant of CHAIN packet is shown in Figure 10. This is an example of NoC without credits. The arbitration in CHAIN is performed only during routing setup and the further data transfer relies on the asynchronous handshake backpressure. During the setup, ROUTE part of the packet is used to correctly steer all the switches along the requested packet path. The EOP notification releases the switching allowing their allocation for a new arbitrated packet. Thus, there is no sharing of the link among multiple packets – once a link is allocated it will serve only one packet. CHAIN is also an example of a *circuit-switched* network [9]. From this example we can conclude that circuit switch network does not require credit-based mechanisms over the basic back-pressure protocol, since this type of network does not share the links during a single packet transfer (there is no packet preemption).

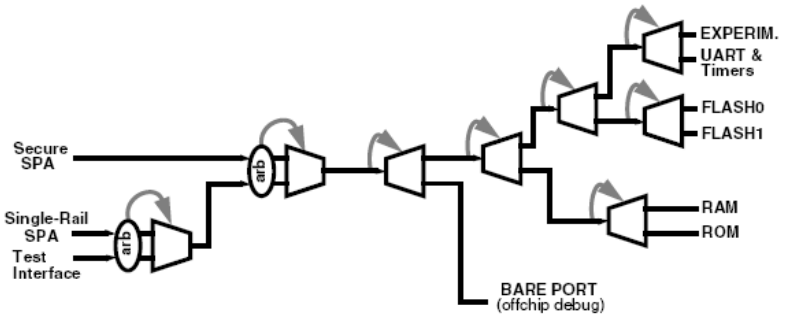

Figure 9: Example of CHAIN network [8]. Arbitration is performed at the route setup stage. No per-flit arbitrations once the route is set

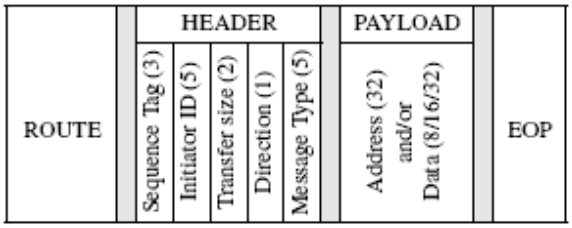

Figure 10: CHAIN Packet Format [8]

3.3. FAUST

The FAUST asynchronous router [11][12] employs two service levels (one called "real-rime" and the other BE). Similarly to CHAIN, FAUST relies on the backpressure mechanism at the low-level (Figure 11). Thus, although, a packet from the lower priority can be preempted, high priority flits can be stuck by a stuck low priority flit. In addition, to avoid network congestions, FAUST uses credit control for *end-to-end* IP unit connections [19]-[21]. The credit is determined according to FIFO sizes. Using pipelined credit transactions, the credit latency is hidden.

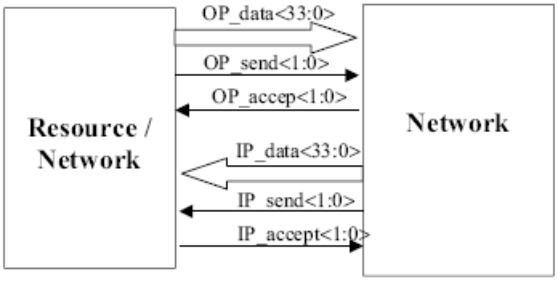

Figure 11: FAUST interface [11]. The two control bits refer to the two virtual channels

3.4. QoS Router

In the QoS router [10] a credit based flow-control mechanism (FCU) is employed to prevent data being sent to a full buffer. Each virtual channel has a separate credit based counter which is decremented when a request is forwarded to the scheduler. If the counter is zero the request is blocked until new credits are received from the receiving node. This is very similar to the general approach of Figure 7, when the credits are communicated through the dedicated wires.

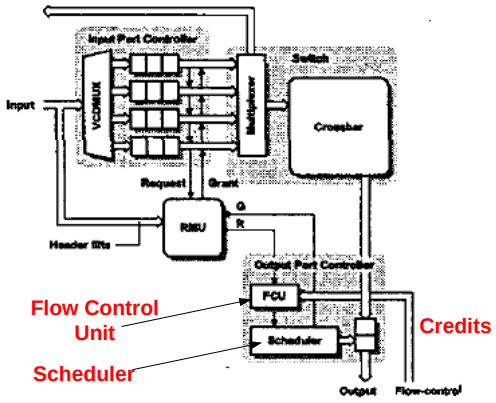

Figure 12: QoS Router Architecture. Flow Control Unit employs counters to count the credits, the credits are communicated through a dedicated wires

3.5. MANGO

The MANGO [13]—[15] router explores VC usage for hard service guarantee routing in combination with BE routing. The MANGO router comprises two sub-modules, a non-blocking switch for hard guarantee service (GS) packets and another for BE packets. Output ports are shared between the two modules using a link arbiter. The router employs a credit mechanism ("VC control"), based on two-phase signaling. The VC control block gates the output of the VC buffers, controlling the access to the ALG arbiter (Figure 13).

The connection between the source and destination is in effect a logical FIFO, distributed across the network (Figure 14). This is a key feature of MANGO, and helps simplify the use of connections, in particular with regard to *end-to-end* flow control.

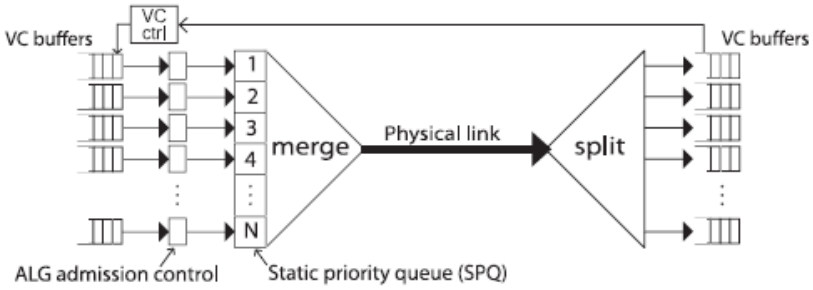

Figure 13: VC control is considered before ALG admission control [13]

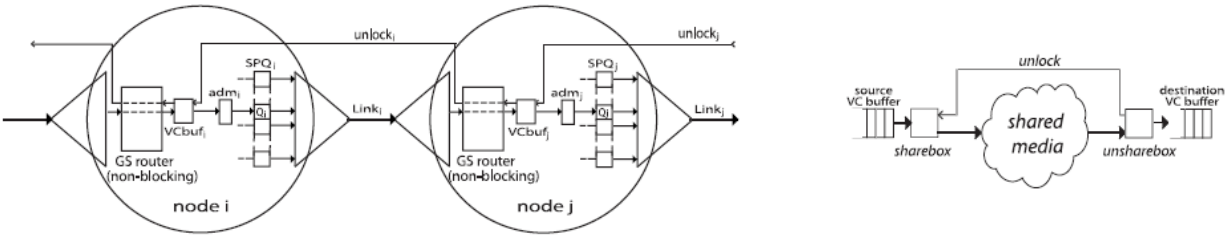

Figure 14: GS reservation through the net using credits [13]

Two crediting methods were described for MANGO [15]. Figure 15 shows a VC link, based on an access method which is called "share-unshare". The sharebox controls access to the link. After letting a flit through it locks, blocking any further flits from passing. Since the unsharebox at the link output implements a buffer, the flit will be able to leave the shared part of the link once it has crossed it. When the flit in turn leaves the unsharehox, freeing its buffer, the sharebox is unlocked by the unsharebox toggling the ***unlock*** wire.

Figure 16 shows the schematic of the share-unshare boxes [15]. Shortly put, the XOR gate generates a pulse upon the unlock signal toggling, resetting the C-elements which lock the input handshake control. The AND gate at the output of

the sharebox decouples the VC input from the shared part of the link, allowing flits to flow fast from all shareboxes, even when the input on each is slow. The unsharehox implements a latch, and a FF which toggles the unlock signal when the output handshake cycle completes, indicating that the latch is ready to receive another flit. In an actual link. optimizations can he made. e.g. if the horn implements latches on its outputs, the unshare functionality can be merged into this, reducing the area overhead further

The share-unshare method is very simple, has high performance and a very low area overhead. In terms of routing, it needs one extra global wire per VC, and this wire is toggled only once per flit, making the power overhead minimal. There are also some drawbacks. First, the BW utilization is poor, when only a single VC is active while the remaining VCs are idle. A single eager VC faces a long handshake cycle spanning the sharebox, the shared part of the link, the unsharebox, and back through the unlock wire. This causes the BW seen by a single eager VC to he only a fraction of the total BW available on the physical link. Another drawback of the share-unshare method is its poor scalability. Since one unlock signal wire is needed for each VC, it scales linearly with the number of VCS (O(N))[1].

In share-unshare approach there is only *one* credit token circulating in the system per each VC, thus degrading the throughput of the single VC down to the throughput of a direct handshake. The long single handshake cycle of a each single VC affect all other VCs that are willing and have credit to send data.

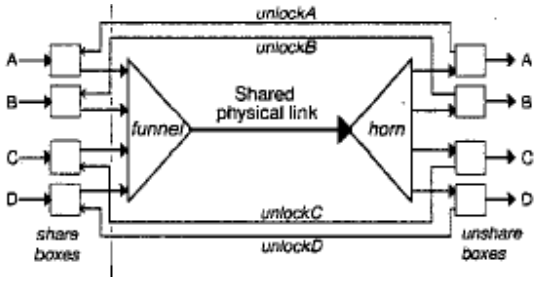

Figure 15: Virtual channel link based on share-unshare method [15]

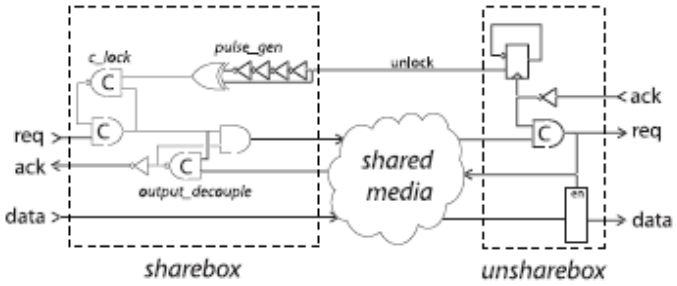

Figure 16: MANGO, Share- and Unsharedbox schematics [13]

The second method described in [15] is the credit-uncredit VC solution shown in Figure 17. It overcomes the drawbacks of the shared-unshared method. Here, the creditbox controls access to the link, by pairing the incoming flits with credits from a credit FIFO. Credits are dataless flow control items, tokens which grant access to the link. Figure 18 shows the schematic for the credit-uncredit boxes. It is made entirely of standard asynchronous handshake components: dataless credit FIFO, join module, output decoupled latch, data FlFO and a fork module. The size of the credit FIFO is equal to the size of the data FIFO, which buffers flits at the receiving end of the link (2 elements shown). At reset the credit FIFO is full of credits, while the data FIFO is empty. Thus, when there are no more credits, this indicates that the data FIFO may be full, and the receiving end can accept no more flits. The uncreditbox in turn sends credits back through the credit link as flits leave through its output. Thus a continuous flow of flits can he maintained, even by a single eager VC, while maintaining non-blocking behavior.

The credit feedback mechanism has a simplifying point to its flow-control: there will never be sent more credits back than the credit FIFO can accept. Each flit passing the creditbox consumes one credit, each flit leaving the uncreditbox generates one credit. A credit can only he generated by a flit which consumed a credit earlier. Therefore there will be

[1] N is the number of VCs.

buffer space available for any credit being generated. This means that the credit flow can use a shared link omitting any VC control, non-blocking behavior still being guaranteed.

The credit-based solution compensates both drawbacks of the share-based: a single eager channel can make full use of the link BW and the number of global link wires scales well with regard to number of VCs. The credit link will have $\log_2(N)$ wires, indicating to which VC the credit belongs, plus handshake wires. The cost is in area used by the latches of the data FIFO, and in power consumption, in that the credit link needs a full handshake per flit. Even if using a 2-phase protocol on the credit link, a minimum of 2 global wire transitions are needed per flit, twice as much as for the share-unshare method. Under most circumstances however this is acceptable, considering the amount of toggling on the global data path.

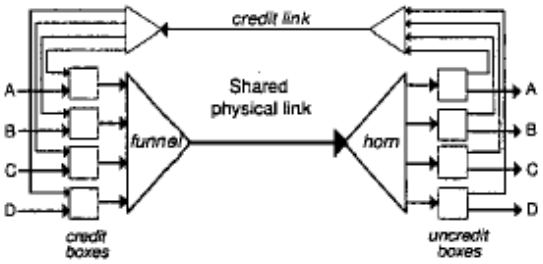

Figure 17: Virtual channel link based on credit-uncredit method [15]

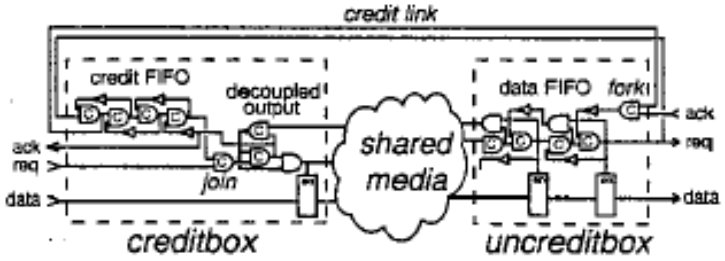

Figure 18: MANGO, Credit- and Uncredit schematics [13][15]

3.6. QNoC without VCs

In [1] the credit mechanism reminds of the shared-unshared MANGO approach since it allows only one credit token per a service level (Figure 19). However, having more pipeline stages after the 4-way SPA, more then one flit (from different service levels) can be sent over the shared media.

The operation of the crediting mechanism is as follows. The data transfer for each service level is enabled only when there is a free buffer space in the next router. A free space indication signal is required per each service level. This indication enables requests of the same service level inside the SPA (Figure 20). The buffer space indication can be made more complex on the receiver side to support higher throughputs.

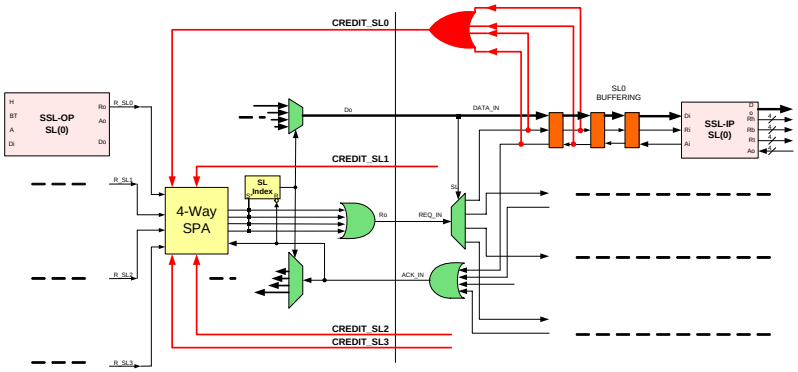

Figure 19: Credit Mechanism in Multi-Service Levels Asynchronous Router [1]

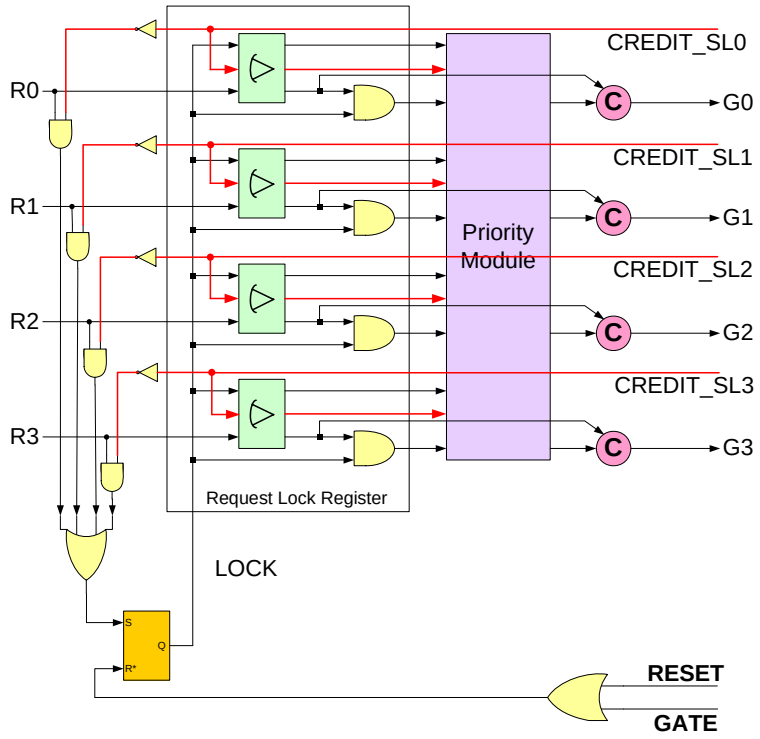

Figure 20: Four-Way Static Priority Arbiter with Credits Support

3.7. QNoC with VCs

When two-dimensional arbitration is performed [16][17] the credits are considered at the earliest arbitration stage (VC-Arbiter) as shown in Figure 21. The VC-IP of the next router generates a 'credit' token once it has room for a new flit, and the VC Arbiter emits a flit only after receiving a credit token. The VC-IP and the VC-OP may include multiple buffers, arranged in an asynchronous FIFO. The deeper pipelining of this router makes the approach of [1] unattractive due to its degraded performance. Therefore, similarly to [15] a credit FIFO is employed. We minimize the number of wires by breaking the full handshake protocol into a partial one, omitting the acknowledge line of the credit connection (Figure 18) and employing timing assumptions.

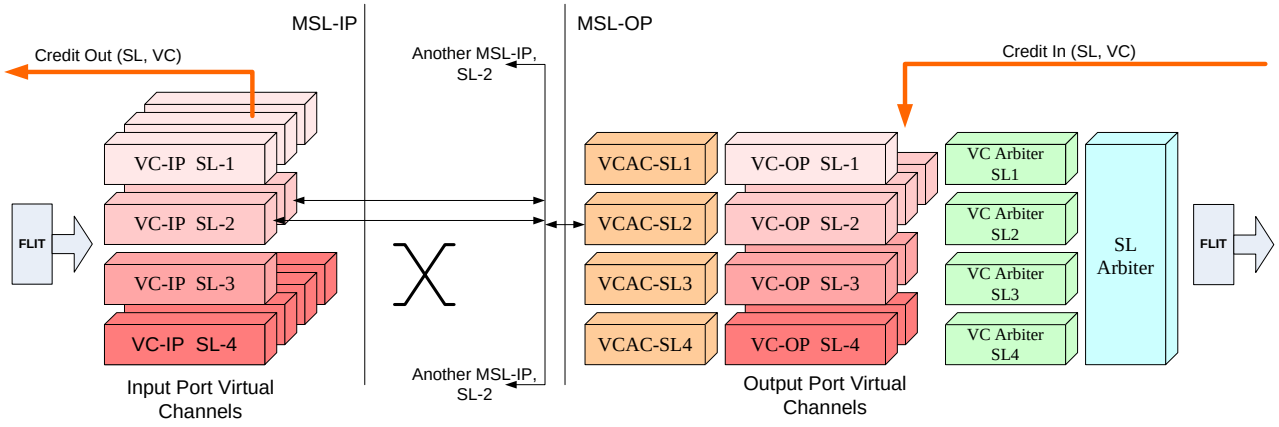

Figure 21: QNoC Router Data Flow

3.8. Net-based crediting: Over-NoC, Bi-directional and Piggybacking

The asynchronous routers described in the previous sub-sections either rely on backpressure only or employ dedicating routing (wires) for crediting communication. Another option is to send the credits over the existing interconnect (over-NoC). One option is to send the credits as *separate messages using the existing available NoC routes*. This approach is employed for end-to-end crediting in [20], Figure 22. However, it can be also considered for switch-to-switch flit-based crediting.

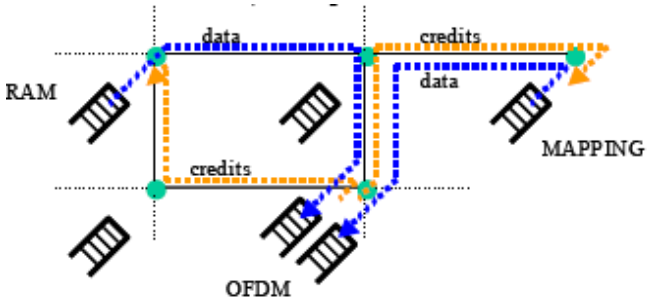

Figure 22: FAUST end-to-end crediting [20]

Another approach is to restrict the NoC links to be bi-directional, while the other direction is used both for data and as the shortest path for token communication. A similar approach is employed in SpaceWire routers [18] and requires internal communication between the input and output ports of the same planar connection (Figure 2 and Figure 23). The IP is responsible for de-muxing two types of messages: flow-control and data. When a flow control messages comes in, it is not passed over to the crossbar, but is forwarded directly as a token to the adjacent OP (RCV CREDIT), indicating to the OP that the IP on the other size has one additional buffer space. In addition, for each flit sent out over DATA_IN, IP issues SEND_CREDIT token that generates a flow control packet with single credit token, that is sent by the OP, and is dedicated to the OP of the other router. This approach should not necessarily be restricted to the bi-directional NoC link: the SEND_CREDIT can be sent over the NoC through as special packets, however, this might require additional buffering space for the credits (small) in the net.

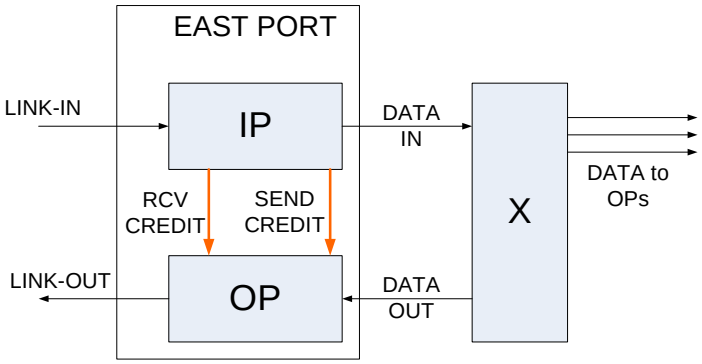

Figure 23: Net-based crediting using bi-directional connection

In order to improve the bandwidth the flow control messages can be combined with the data messages. In the Æthereal router [3], the credits are piggybacked in the header of the packets (Figure 24) for the data in the other direction to improve NoC efficiency (overhead bandwidth can be reduced with up to 20% by piggybacking). Note that this approach requires bi-directional communication for any two adjacent routers and may complicate the router design.

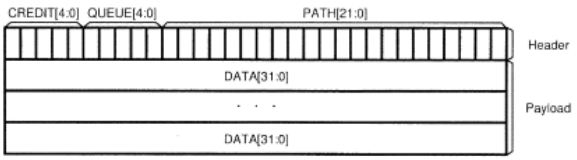

Figure 24: Piggybacked credits [3]

3.9. Additional improvements: Credit threshold

In order to improve the utilization and bandwidth of the network credit threshold may be employed. This is applicable only when a chunk-crediting is considered. In this approach the credits are send only when the amount of the free space inside the buffer is greater than a certain threshold [3]. Credit thresholds increase NoC utilization, especially for small burst sizes, because by forcing credit accumulation, less empty packets carrying only credits are generated. However, as opposed to the data thresholds, setting credit thresholds has little impact on the latency and buffer requirements [3]. The reason is that the application has a periodic bursty behavior, and the time needed to report credits back (in one or multiple packets) is lower than the time between bursts. Consequently, credits are always reported in time, thus preventing data being buffered, and not affecting the latency.

As credits are piggybacked on packets, a queue becomes eligible for scheduling (i.e., request generator in Figure 8 issues a signal for that queue to GT scheduler and BE scheduler) when either the amount of sendable data is above its data threshold, or when the amount of credits is above its credit threshold. However, once a queue is selected, a packet containing the largest possible amount of credits and data will be produced. Note the amount of credits is limited by the implementation to the given number of bits in the packet header (Figure 24), and BE packets have a maximum length to avoid links being used exclusively by a single packet/channel, which could cause NoC congestion and/or starvation.

For the incoming packets, the NI inspects the header, adds the credits to the counter space, and stores the data (without the header) in the queue specified by the queue id field in the packet header. The data is then ready for consumption by the shells at the NI-kernel ports.

4. Summary and Conclusions

The different crediting approached employed in the published asynchronous routers are listed in Table 1. In most of the approached flit-based crediting is employed. Token implementation seems to be simpler and therefore more attractive. In some approaches that also relate to fabricated chips (CHAIN, FAUST) the designers omitted the switch-to-switch crediting, relying on higher-level end-to-end crediting or/and on basic asynchronous protocol backpressure.

Table 1: Classification of the published asynchronous crediting

Router	Location	Size	Implementation
CHAIN [7][8]	N/A	N/A	N/A
QoS router [10]	Switch-to-switch	Flit-based	Dedicated Routing, Counter
FAUST [11][12]	End-to-end	Chunk-based	Net-based Routing
MANGO [13][14][15]	Switch-to-switch	Flit-based	Dedicated Routing, Tokens
QNoC [5][16][17]	Switch-to-switch	Flit-based	Dedicated Routing, Tokens

We also observe that the circuit switch (GS) types of networks do not require credit-based mechanisms over the basic back-pressure protocol, since this type of network does not share the links during a single packet transfer (there is no packet preemption).

We have surveyed a few techniques for improving the performance of credit communication, however, some of them may be translated into relatively complex design or may require unsupportable system resources.

For a possible future work we note that:

a. Credit mechanisms can be described using STG with multiple-tokens.
b. Adaptive routing can be combined with crediting. For example local/global NoC congestion monitoring may be used for dynamic credit generation. While the produced credits can be distributed in different ways (through the NoC, through a centralized memory access, etc.)

References

[1] R. Dobkin, V. Vishnyakov, E. Friedman and R. Ginosar, "An Asynchronous Router for Multiple Service Levels Networks on Chip," Proc. ASYNC, pp. 44-53, 2005.

[2] S. Noh, et al, "Performance and Complexity Analysis of Credit-Based End-to-End Flow Control in Network-on-Chip", 2007.

[3] A. Radulescu, J. Dielissen, S. González Pestana, O. Prakash Gangwal, E. Rijpkema, P. Wielage and K. Goossens, "An efficient on-chip NI offering guaranteed services, shared-memory abstraction and flexible network configuration," TCAD 24(1), 2005, pp. 4-17.

[4] W. Dally, B. Towles, "Route Packets, Not Wires: On-Chip Interconnection Networks," DAC, 2001.

[5] R. Dobkin, R. Ginosar, A. Kolodny, "QNoC Asynchronous Router", VLSI Integration Journal, 2008.

[6] I. Walter, I. Cidon, R. Ginosar, A. Kolodny, "Access Regulation to Hot-Modules in Wormhole NoCs," NOCS, 2007.

[7] J.Bainbridge and S.Furber, "Chain: a Delay-Insensitive Chip Area Interconnect," IEEE Micro, 22(5):16-23, 2002.

[8] W.J. Bainbridge, L.A. Plana and S.B. Furber, "The Design and Test of a Smartcard Chip Using a CHAIN Self-timed Network-on-Chip," Proc. of DATE, Vol.3, pp. 274-279, 2004.

[9] A. Banerjee, R. Mullins, S. Moore, “A Power and Energy Exploration of Network on-Chip Architectures”, NoCs 2007.

[10] T Felicijan, S.B. Furber, "An Asynchronous On-Chip Network Router with Quality-of-Service (QoS) Support," Proc. of IEEE Int. SOC Conf., Santa Clara, CA, pp. 274-277, 2004.

[11] E. Beigne, F.Clermidy, P.Vivet, A.Clouard, M. Renaudin, "An Asynchronous NOC Architecture Providing Low Latency Service and Multi-Level Design Framework," Proc. of ASYNC, pp. 54-63, 2005.

[12] E. Beigne, P. Vivet, "Design of On-chip and Off-chip Interfaces for a GALS NoC Architecture", Proc. ASYNC, pp. 172-181, 2006.

[13] T. Bjerregaard, J. Sparso, "A Scheduling discipline for latency and Bandwidth Guarantees in Asynchronous Network-on-chip", Proc. ASYNC, pp. 34-43, 2005.

[14] T. Bjerregaard, J. Sparso, "A Router Architecture for Connection-Oriented Service Guarantees in the MANGO Clockless Network-on-Chip," Proc. DATE, Vol.2, pp. 1530-1591, 2005.

[15] T. Bjerregaard, J. Sparso, "Virtual channel designs for guaranteeing bandwidth in asynchronous network-on-chip", Norchip Conference, 269-272, 2004.

[16] R. Dobkin, R. Ginosar, A. Kolodny, "QNoC Asynchronous Router", VLSI Integration Journal, 2008.

[17] R. Dobkin, R. Ginosar, I. Cidon, "QNoC Asynchronous Router with Dynamic Virtual Channel Allocation," First ACM/IEEE Int. Symp. on Networks on Chip (NOCS), p. 218, 2007.

[18] S.M. Parkes, “SpaceWire: Links, Nodes, Routers and Networks”, European Cooperation for Space Standardization, standard number ECSS-E50-12A, January 2003.

[19] D. Lattard, E. Beigne, F. Clermidy, Y. Durand, R. Lemaire, P. Vivet, F. Berens, "A Reconfigurable Baseband Platform Based on an Asynchronous Network-on-Chip," IEEE Journal Of Solid State Circuits, Vol. 43, Issue 1, pp. 223-235, 2008.

[20] F. Clermidy, D. Varreau, D. Lattard, "A NoC-based communication framework for seamless IP integration in complex systems," Proceedings of Design and Reuse IP-SOC’2005, Grenoble, France, pp. 279-283, 2005.

[21] R. Lemaire, Y. Durand, D. Lattard, A. Jerraya, "A semi-distributed control system for application management in a NoC-based architecture," 24th IEEE Norchip Conference, 2006.