A Dynamic Link Allocation Router

Wei Song and Doug Edwards
School of Computer Science, the University of Manchester
Oxford Road, Manchester M13 9PL, UK
{songw, doug}@cs.man.ac.uk

***Abstract*—The connection oriented flow control method is subject to a high retry rate with the heavy network load. An asynchronous link induces an extra latency when its bandwidth is directly enlarged by increasing the wire count. In this paper, a new dynamic link allocation flow control method is proposed. By dynamically allocating the sub-links on a link to multiple frames, it decreases the retry rate of the connection oriented flow control method and increases the link bandwidth without the extra latency. Based on this flow control method, a new router architecture is introduced.**

I. Introduction

A novel network-on-a-configurable-chip (NoRC) platform has been proposed in [1] to dynamically load multiple real-time applications onto the on-chip network at run-time. All applications running on this NoRC search for and reconfigure network nodes automatically by the connection oriented stochastic routing algorithm [2]. The whole system releases the calculation burden of exploiting the optimal task mapping and shows an excellent fault-tolerance to permanent and intermittent errors.

This platform will be implemented on a globally asynchronous and locally synchronous network-on-chip (NoC) where processing elements are implemented by synchronous circuits but the routers are fully asynchronous. The synchronous circuits talk with the asynchronous communication network through the synchronisation interfaces in the network interfaces. Routers and network interfaces are connected by the delay insensitive CHAIN links [3].

However, the connection oriented stochastic routing algorithm shows a low network capacity and a high retry rate when the network load is heavy. In this paper, a new dynamic link allocation flow control method is proposed to dynamically allocate the sub-links on a link to multiple frames. This new method is expected to increase the saturation throughput and reduce the retry rate. A new dynamic link allocation router (DyLAR) is introduced using this flow control method.

II. Motivations

The real-time multimedia applications which run on the NoRC platform would require a 1Gbit/s bandwidth [4]. Since the single 1-of-4 CHAIN link cannot meet this requirement, the routers in NoRC need to increase the bandwidth of asynchronous links.

The low network efficiency of the connection oriented flow control methods is another problem. They are prone to saturate the network at a much lower practical capacity than the virtual channel (VC) flow control method [5], which incurs unnecessary power consumption and wastes bandwidth.

Moreover, the area consumption of current asynchronous routers, such as the routers in MANGO [6] and QNoC [7], is large. It will be meaningful to exploit some methods to solve the head-of-line (HOL) problem with a small area overhead.

III. The Dynamic Link Allocation Flow Control Method

Unlike the synchronous bus where the bandwidth can be directly enlarged by increasing the number of wires, asynchronous links run more efficiently with the lower bandwidth. Shown in Fig. 1, an N-bit channel is composed of $N/2$ dual-rail channels. The minimal latency of one data transaction can be calculated as:

$$L_{channel} = 2[L_{link} + 2L_c + L_{or} + (\log_2 N){\cdot}L_c] \quad (1)$$

Described by this equation, increasing the bandwidth of an asynchronous channel directly by adding wires incurs an extra delay of $(\log_2 N){\cdot}L_c$. In addition, the ack line is only set when all or gates are high, which is a worst case delay. Therefore, comparing with directly increasing the bandwidth of an asynchronous channel, adding more separated and self-govern sub-links and running them in parallel would be more efficient.

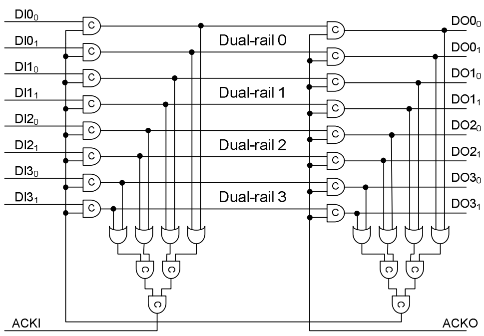
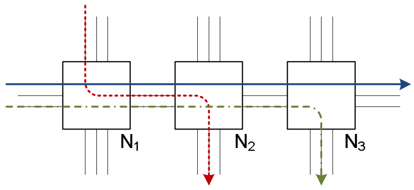


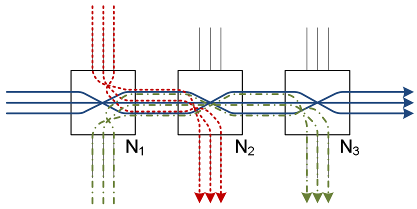
Fig. 1. A dual-rail pipelined channel

A major disadvantage of the connection oriented flow control method is its high retry rate when the network load is heavy. To reduce the retry rate, a direct and fundamental solution is to use the virtual channels. However, it is more efficient to divide a link into multiple small asynchronous sub-links but not connecting a wide channel with multiple input buffers. The spatial division multiplex (SDM) is a more convenient choice for the asynchronous routers than VC.

However, SDM unnecessarily wastes link bandwidth. This problem can be observed clearly from Fig. 2(a) where every link is divided into three sub-links. Under this traffic pattern, the network bottleneck occurs on the link $L_{N_1-N_2}$ and one sub-link on $L_{N_2-N_3}$ is wasted. Furthermore, due to the connection oriented flow control method, sub-links are busy only during the data transmission stage. As a result, no further frames can pass $L_{N_1-N_2}$ even the resources are not fully in use.



(a) An example of the SDM flow control method



(b) An example of the DyLAR flow control method

Fig. 2. The difference between SDM and DyLAR

The DyLAR flow control method intents to utilise this spare bandwidth and further reduces the average frame latency at a low network load. Shown in Fig. 2(b), all sub-links are dynamically allocated on a flit to flit basis, therefore, when other communications are idle, the active communication can fully utilise all sub-links. When the network load is heavy, all communications fairly share the bandwidth by an equal probability as far as the allocation algorithm is unbiased.

As a byproduct of the DyLAR flow control method, the HOL problem would be more serious than SDM. Fig. 3 illustrates an example of the HOL problem possible in network. There are three communications exist in network: $N_{(3,1)}$–$N_{(1,3)}$, $N_{(2,0)}$–$N_{(1,2)}$, and $N_{(1,1)}$–$N_{(1,3)}$. Suppose all three communications are active at the same time, the link $L_{(1,1)-(1,2)}$ is the bottleneck. All communications share the bandwidth on $L_{(1,1)-(1,2)}$. However, the communication $N_{(3,1)}$–$N_{(1,3)}$ exclusively occupies the bandwidth on $L_{(3,1)-(2,1)}$ and $L_{(2,1)-(1,1)}$ and the communication $N_{(2,0)}$–$N_{(1,2)}$ exclusively occupies the bandwidth on $L_{(2,0)-(1,0)}$ and $L_{(1,0)-(1,1)}$. $N_{(3,1)}$ and $N_{(2,0)}$ may send out flits by a speed faster than the actual speed on link $L_{(1,1)-(1,2)}$ at first. Since sub-links are exclusively occupied by flits, the surplus flits that cannot pass $L_{(1,1)-(1,2)}$ immediately would reserve the sub-links on $L_{(3,1)-(2,1)}$, $L_{(2,1)-(1,1)}$, $L_{(2,0)-(1,0)}$ and $L_{(1,0)-(1,1)}$, and block other communications.

To constrain the number of sub-links allocated by the

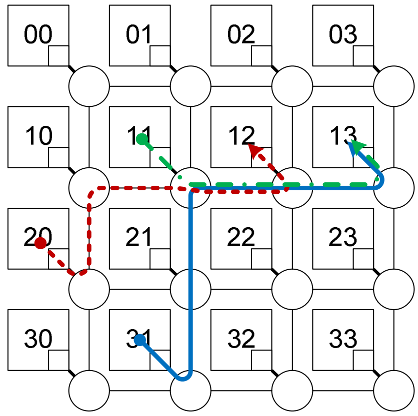


Fig. 3. An example of the HOL problem in when using DyLAR

routers before the bottleneck link, the DyLAR router can sense congestions through the backpressure technology [8]. When a new flit arrives a router, this router will set the backpressure bit to high until an idle output sub-link is allocated for this flit. Sent back to the last stage router, this backpressure bit blocks all incoming flits of this frame until it is reset. Consequently, when a blockage or bandwidth unbalance occurs, one frame only occupies one sub-link on each link in the worst case. The DyLAR flow control method has the same worst case bandwidth waste with SDM.

In the DyLAR router, sub-links are allocated by a flit to flit basis, so that path cannot be reserved by reserving one of the input buffers, which is used in SDM and VC. An extra request network is built for the DyLAR flow control method. For every frame, the path is reserved by a request line in the request network but not an input buffer. Since the area overhead of a request line in the request network is only several standard cells, this area overhead is endurable, compared against the improvement to throughput and latency.

IV. *DyLAR*, THE DYNAMIC LINK ALLOCATION ROUTER

A. *The Overall Structure*

Fig. 4 demonstrates the overall structure of a DyLAR router with three sub-links on one port.

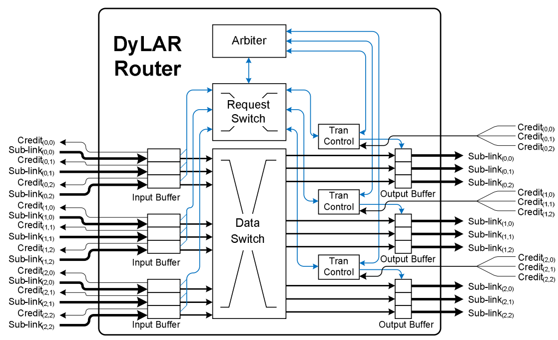


Fig. 4. The DyLAR router

Each input sub-link is connected with a flit size input buffer. According to the information in the flit header and the flit type fields, input bffers send early requests to the request switch when necessary data are collected. Initially all request lines from input buffers are connected with the central arbiter. After the arbiter allocates an transmission control (tran control in Fig. 4), this transmission control unit reconfigures the request switch and the request line is re-connected with the allocated transmission control.

The central arbiter is responsible for routing decisions and the allocation of the request switch. It gathers all necessary information from the request switch and transmission control units and makes routing decisions by requesting the selected transmission control unit to reconfigure the request switch.

The transmission control unit controls the flit transmission procedure, gathers information from the next stage router and dynamically reconfigures the data switch to link an output buffer with an input buffer under request. Thanks to the backpressure technology, the transmission control unit only allocate a new sub-link for a request line when the previous flit has been allocated with an idle output sub-link by the next stage router.

Output buffers are simple asynchronous pipelines. For every flit, the transmission control unit reconfigures the flit header to identify the target request line in the next router and reconfigures the flit type field if it is required to be changed by the arbiter. The size of the output buffer is only enough for the flit header and the flit type fields, much smaller than the flit size.

B. The Flit Definitions

All flits in network falls into one of the four categories: the *request* flit, the *data* flit, the *ok-ack* flit which denotes a path is reserved and the *false-ack* flit which denotes a release of the path. The field definitions for all these flits are shown in Fig. 5.

data	request content	flit type	flit header

(a) A *request* flit

data	flit type	flit header

(b) The field definition of other flits

Fig. 5. The flit field definitions

The flit header field is added and rewritten by routers and network interfaces to denote the target request line. The flit type field defines the type of the flit. For a *request* flit, the request destination is identified by the request content field which is defined by a specific routing algorithm and is interpreted by the central arbiter. All flits may or may not have a data field of a variable length. Therefore, the flit length in this NoC is not fixed.

C. Input Buffers

Every sub-link is connected with an input buffer that contain a flit size buffer. The input buffer analyses the flit header and the flit type field, and send out a request to the request switch when necessary data are collected. The central arbiter may read the whole flit when making the routing decision.

Fig. 6 depicts the internal structure of an input buffer. All data stored in input buffers are in the 1-of-4 format. The whole data buffer is divided into three parts: flit header, flit type and data field. Data buffer may be implemented by latches or C-elements. The adaptive pipeline length control technology [9] would be used to reduce the pipeline latency. The buffer busy bit to the previous stage router is set when receiving a flit and is reset when all data in the data buffer are transmitted. Request lines are triggered as early as possible. Multiple request lines are selected by the flit type field through the demux gate. The mux gate is controlled by the flit header which denotes the specific request line in the request network.

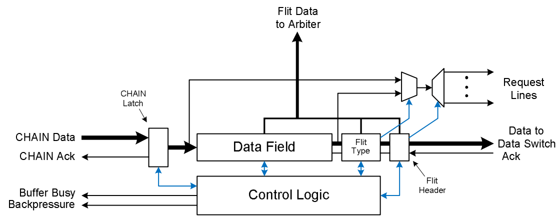


Fig. 6. An input buffer

D. The Request Switch

Suppose there are CH_NUM sub-links connected with one port and for each port there are a maximal REQ_NUM requests sharing the bandwidth. Fig. 7 shows one sub switch.

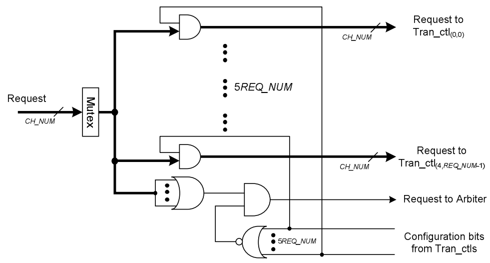


Fig. 7. An sub switch for one group of request line

The flits arrived at an input buffer may belong to anyone of the requests that share the bandwidth. To identify the specific input buffer, there are CH_NUM request lines for each request. The mutex guarantees that only one of the request lines can fire to the arbiter or the transmission control unit. All configuration bits in Fig. 7 are initialised to low, therefore, the request lines are initially connected to the arbiter.

For a *request* flit, the arbiter makes routing decisions by setting the request configuration registers (req_cfg_reg) in the transmission control units, shown in Fig. 8. This configuration register is composed of two fields: the direction field and the request selection field. Both of them are coded in the one-hot style. The and gates connected with the configuration registers translate the register into $5REQ\_NUM$ configuration bits which connect one of the $5REQ\_NUM$ request lines to this request. The configuration register is reset when the *false-ack* flit is transmitted and detected by the corresponding output buffer. Consequently the request lines are connected to the arbiter again.

For all requests in a router, a total $5REQ\_NUM$ sub requests are needed to connect all request lines to their target transmission control units. The area of the request network is proportional to the square of the request number on one link.

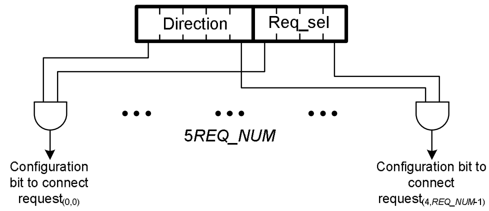


Fig. 8. The request configuration register of a request in the transmission control

E. The Data Switch

The data switch could be the most area consuming part in the DyLAR router. For any output sub-link, it may receive a flit from all input sub-links. Therefore, the data switch is a $5CH\_NUM \times 5CH\_NUM$ matrix and each line in this matrix is a 1-of-4 data bus along with the acknowledgement wire. The connection of the data switch is controlled by the data_cfg_reg registers in the transmission control units in the same way with the req_cfg_reg registers.

Several methods can reduce the area of the data switch. One solution is to use the hierarchical conjunctions. By grouping the links and adding several layers in the matrix the area may drop. Another solution is to constrain the connectivity of the data matrix. If an input sub-link is connected with only two of the output sub-links on a port, it would reduce the data switch from a $5CH\_NUM \times 5CH\_NUM$ matrix to $5CH\_NUM \times 10$ matrix and still provides a constrained dynamic bandwidth allocation to all requests.

F. The Transmission Control Unit

The transmission control units in the DyLAR router allocate output sub-links by dynamically configuring the data switch. A transmission control unit is deployed to each link to allocate the sub-links belong to this link. The internal structure of a transmission control unit is demonstrated in Fig. 9.

There are three groups of configuration registers in a transmission control unit: the request configuration registers (req_cfg_reg), the data configuration registers (data_cfg_reg) and the flit_type registers. The req_cfg_reg are written by the arbiter and reset by the transmission control unit after the *false-ack* flit is received by an input buffer or sent out by an output buffer. The data_cfg_reg registers have the same implementation with the req_cfg_reg registers. They are written by the transmission control unit when a sub-link is allocated to an input buffer and reset by the output_buffer_free signal when the flit is transmitted. The flit_type registers store

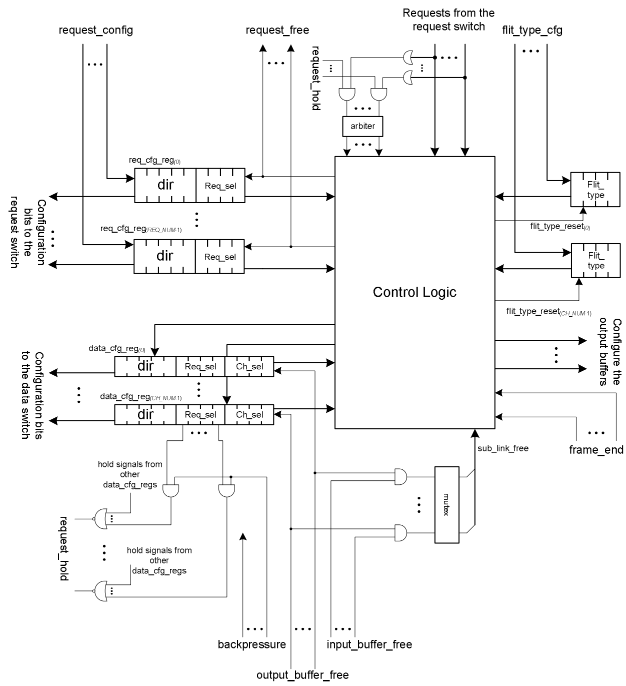


Fig. 9. The internal structure of the transmission control

the flit types for each request. They are set by the arbiter when a *request* flit or a *data* flit is changed to a *false-ack* flit.

Denoting the status of local request lines, the request_free bus is set when a frame is sent. This bus is also connected to the arbiter where it is used to make routing decisions. Similarly, output_buffer_free bus records the status of output buffers. It is set when the a flit is sent by the output sub-link. This bus and the input_buffer_free bus from the next stage router generate the sub_link_idle bus which is used to allocate an idle output buffer for an requesting input sub-link.

The backpressure signals are transformed into the request_hold signals for each request line. Due to the arbiter on the request lines, only one request, which is not holden by the request_hold signal, is served by the transmission control unit in one time slot. Note that the arbiter is a REQ_NUM arbiter but not a $REQ\_NUM \cdot 5REQ\_NUM \cdot CH\_NUM$ one because only one request line for the total $5REQ\_NUM \cdot CH\_NUM$ request lines connected to the single request can fire at one time.

G. The Arbiter

The arbiter in the DyLAR router is responsible for the routing decisions and the allocation of request lines. Since the sub-link allocation is proceeded by the transmission control units, the arbiter can work in serial that only one request is served at one interval. It is expected that the serial arbiter is smaller than the parallel arbiter in the VC router. Any routing algorithms can be implemented in this arbiter which is not the topic of this paper.

V. The Expected Performance

Suppose all routers have the same theoretic bandwidth, table I illustrates the expected performance comparison results between SDM, DyLAR and VC.

TABLE I
The expected performance comparison

	Latency	Throughput	Power	Area
SDM	worst	worst	best	best
DyLAR	medium	medium	medium	medium
VC	best	best	worst	worst

When considering the average frame latency, we suppose the network load is low. Under this condition, the DyLAR router would use less time to transmit a frame than the SDM router. DyLAR can allocate all sub-links of a link to a frame as far as no other frames are competing for the bandwidth. However, the VC routers have the minimal latency because they always allocate the whole bandwidth to the flit under transmission.

The throughput represents the overall throughput under the maximal practical frame injection rate. It reflects the efficiency of a flow control method to use the bandwidth under a heavy network load. Due to the large input buffers, the VC router can temporarily store numerous flits, which lead to the best tolerance to congestions. DyLAR router should outperform the SDM router because the DyLAR router support more parallel communications on each link.

Definitely the SDM router has the lowest power and area consumption, especially when the crossbars are statically configured. Compared against the SDM router, the DyLAR router has a flit size buffer for each sub-link. However, the DyLAR router is more power and area efficient than the VC routers because every VC in the VC router is composed of multiple flit size buffers.

VI. Conclusion

In this paper, a novel flow control method — dynamic link allocation — is proposed for the NoRC platform and a DyLAR router is introduced using this flow control method.

Developed from the SDM flow control method, the DyLAR flow control method can dynamically allocate the spare bandwidth on a link to multiple frames. As a result, the DyLAR router shows the less zero-load latency and the larger full-load throughput than the SDM router and obtains a comparable latency and throughput performance with the VC router. Moreover, the DyLAR router built on the DyLAR flow control method consumes less area and power than the VC router.

References

[1] J. L. Nunez-Yanez, D. Edwards, and A. M. Coppola, “Adaptive routing strategies for fault-tolerant on-chip networks in dynamically reconfigurable systems,” *IET Computers & Digital Techniques*, vol. 2, no. 3, pp. 184–198, May 2008.

[2] M. Hosseinabady and J. Nunez-Yanez, “Fault-tolerant dynamically reconfigurable NoC-based SoC,” in *Proceedings of IEEE International Conference Application-specific Systems, Architectures and Processors*, Leuven, Belgium, June 2008.

[3] J. Bainbridge and S. Furber, “Chain: a delay-insensitive chip area interconnect,” *IEEE Micro*, vol. 22, pp. 16–23, 2002.

[4] P. Guerrier and A. Greiner, “A generic architecture for on-chip packet-switched interconnections,” in *Proceedings of Design, Automation, and Test in Europe*, 2000, pp. 250–256.

[5] D. Wiklund and D. Liu, “SoCBUS: switched network on chip for hard real time embedded systems,” in *Proceedings of InternationalParallel and Distributed Processing Symposium*, April 2003.

[6] T. Bjerregaard and J. Sparsø, “A router architecture for connection-oriented service guarantees in the MANGO clockless network-on-chip,” in *Proceedings of Design, Automation and Test in Europe*, 2005, pp. 1226–1231.

[7] R. R. Dobkin, R. Ginosar, and A. Kolodny, “QNoC asynchronous router,” *Integration, the VLSI Journal*, vol. in press, 2008.

[8] L. Tassiulas, “Adaptive back-pressure congestion control based on local-information,” *IEEE Transactions on Automatic Control*, vol. 40, no. 2, pp. 236–250, February 1995.

[9] A. Efthymiou and J. D. Garside, “Adaptive pipeline structures for speculation control,” in *Proceedings of International Symposium on Asynchronous Circuits and Systems*, May 2003, pp. 46–55.