Deterministic scheduling in Networks-on-Chip using the Trajectory approach

Ermis Papastefanakis, Xiaoting Li, Laurent George

Deterministic scheduling in Networks-on-Chip using the Trajectory approach

Ermis Papastefanakis[†‡], Xiaoting Li[*], Laurent George[‡]
[*]ECE Paris, 75015 Paris, France
[†]Thales Communications and Security, 92622 Gennevilliers, France
[‡]Université Paris-Est, LIGM / ESIEE, Champs sur Marne, France
Email: ermis.papastefanakis@thalesgroup.com, xiaoting.li@ece.fr, laurent.george@univ-mlv.fr

***Abstract*—In this paper, we consider the problem of guaranteeing real-time end-to-end transmission time for flows sent on a Network-on-Chip (NoC) with First-in First-out (FIFO) scheduling on each node. We show how to adapt the Trajectory approach, used in the context of Avionics Full DupleX switched Ethernet (AFDX) networks to characterize end-to-end transmission delays, to the context of NoC-based Systems-on-Chip (SoCs). We characterize the benefit of the Trajectory approach on an example.**

***Keywords*—Determinism, Network-on-chip, Trajectory approach, real-time.**

I. Introduction

As the number of elements in a Multi-Processor System-on-Chip (MPSoC) increases, so does raw processing power. This introduces augmented complexity that makes certain features such as determinism or Quality of Service (QoS) more and more difficult to maintain. As a result, the gap between performance and predictability (worst-case execution time (WCET)) is quite large, suggesting underused resources. NoCs are a new paradigm for on-chip interconnection that is being adopted by the majority of new SoCs. The concept behind NoCs is to adapt the principles of networks and implement them inside the chip, achieving to transfer packets instead of electric signals [1]. NoCs possess a modular architecture that offers improved spatial and temporal separation. All this creates a natural interest to evaluate the potential to exploit resources in an efficient way, preserving at the same time the system's determinism [2].

Similar work on real-time scheduling has been realized on [3] with the difference that fixed-priority and virtual channels are considered in the NoC platform. While those characteristics are often adapted, it is not always the case. We chose the Trajectory approach because it has demonstrated low pessimism and will allow to achieve interesting results on a more generic NoC platform.

Our contribution: We show how to adapt the Trajectory approach (successfully applied to off-chip networks such as Switched Ethernet) to NoCs with FIFO scheduling. It is important to note that the constraints in NoCs are very different from those in AFDX Switched Ethernet networks which makes the adaptation to NoC-based systems necessary. In this paper we analyze the worst-case traversal time (WCTT) of sporadic flows to be able to guarantee real-time response on a chip level without requiring implementation of Virtual Channels (VCs). We achieve that by calculating the WCTT between two Intellectual Property (IP) elements using the Trajectory approach.

This paper is organized as follows. In section II, we introduce the NoC platform and the corresponding network model. We then recall existing research on NoCs in section III. The Trajectory approach adapted to the NoC ecosystem is presented in section IV. In section V we examine a use case to illustrate the method along with the results. Finally we conclude our work in section VI.

II. Platform and network model

A. Platform

Each router R_{xy} consists of five links, four located at the edges **N**orth, **E**ast, **W**est, **S**outh (NEWS), used to connect with neighbor routers and the fifth is used to connect with the **L**ocal (L) IP$_{xy}$. An illustration of a router R_{xy} is given in Figure 1. For example, R_{xy_W} signifies the West link of router R_{xy}. Here x and y are the coordinates of the router inside the 2D mesh and they range from 0 to 3 for a 4x4 NoC.

In order to traverse a router, there are two levels that a flit has to pass, each taking one clock cycle. In the first one, buffering and routing take place while the second deals with arbitration and output. From a time standpoint, during the first cycle a header flit enters a router and is stored in a small size buffer that can hold up to four flits. At the same cycle it passes through a routing mechanism to determine which output link it wants to reserve. During the second cycle the arbiter (one in each output) will decide which of the potentially competing inputs will take over the output link. At the same time the output register (no output buffers) holds the flit that will traverse the link. These two levels are pipelined and initially two cycles are needed to forward the header flit but each of the payload flits will only require one cycle to follow through the path. In this work, we study the FIFO arbitration scheme in which the output controller reserves the path for the packet whose header flit arrives first. When the path is freed, the arbiter reserves the path for the packet whose header flit arrives secondly and so on.

This platform is implemented in Verilog and is able to synthesize on a Field Programmable Gate Array (FPGA) (Xilinx Virtex-7). Measurements can be taken through a cycle accurate simulator or through traces of the FPGA output stream.

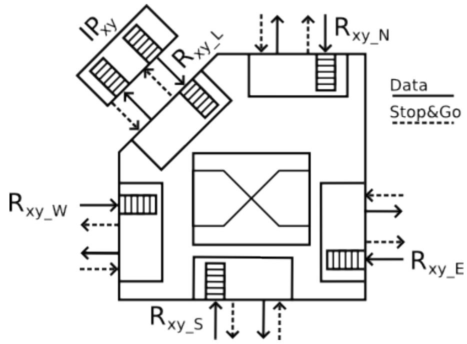

Fig. 1. Architecture of a NoC router R_{xy}

B. Network model

We consider n sporadic flows transmitted in the NoC. A sporadic flow τ_i sends packets respecting two parameters: 1) the period T_i which is the minimum temporal interval between the arrival of two consecutive packets, and 2) the maximum transmission time C_i which is the maximum time to transmit all the flits of a packet on a router. We denote D_i a bound on the WCTT of any packet of flow τ_i.

The transmission of one flit on a link takes one clock cycle T_c, and the period T_i as well as the transmission time C_i, are multiples of clock cycle T_c. In this work, we consider for each packet of a message a constant transmission cycle $C_i = 4 \times T_c$ including the header flit and three payload flits. For a packet f_i of flow τ_i, we denote the header flit f_{ih} and the payload flits $f_{i,1}$, $f_{i,2}$ and $f_{i,3}$. Due to the dimension-order X-Y routing, each packet of flow τ_i follows a static path denoted $\mathcal{P}_i$ which is composed of the source and destination IPs as well as the input ports of routers along this path. The first buffer of the source IP is denoted $first_i$, while the last buffer of the destination IP is denoted $last_i$. Then the path of flow τ_i is represented by $\mathcal{P}_i = \{first_i, ..., last_i\}$.

We consider one diffusion path in the network which means that when packets of different flows join one path, they do not leave this path until they are transmitted to the same destination. A real use case that illustrates this concept can be found in memory hierarchies where the last level, a common bottleneck in MPSoCs, is the Random Access Memmory (RAM). In such a case a number of IPs will try to access the RAM memory and combined with XY routing, the generated traffic will join a single path leading to the memory controller. An illustrative example of one diffusion path of NoC is shown in Figure 2. Flow τ_1 follows path $\mathcal{P}_1 = \{\text{IP}_{32}, R_{32\_L}, R_{22\_S}, R_{12\_S}, \text{IP}_{12}\}$. Three flows τ_2, τ_3 and τ_4 join this path till IP_{12}.

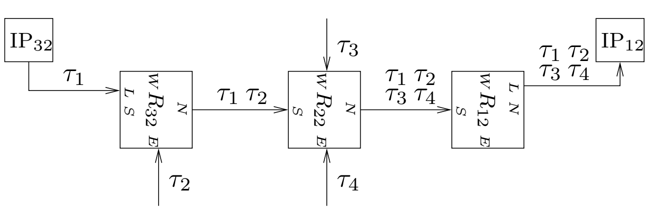

Fig. 2. An example of one diffusion path of NoC

III. Current work on NoCs

With the advance of performance requirements in System-on-Chip (SoC), with an increase in the number of IPs to be connected, the interconnect becomes a bottleneck and suffers from scalability issues. Network on Chip (NoC) are seen as a solution to this scalability issue [4] by providing configurable network paradigms at small size (computation and storage functions are impleted at silicon level). Providing real-time communications bounds in NoCs is therefore a challenge that needs to be addressed with specific approaches that take into account stringent requirements imposed by the hardware (small buffer size, flit level granularity).

Wormhole routing is a popular solution to take into accound small buffer size constraints in router [5], [4]. Wormhole routing can lead to contention problems in communication where a packet can delay all packets trying to access the output of the same router. The delay on one router can result in a domino effect eventually delaying directly all packets on the same path as well as indirectly packets on other routers. This contention problem must be taken into account in a worst-case end-to-end delay analysis [6], [7], [3]. In [3], authors prove that the general problem of exact schedulability of real-time traffic-flows sent in NoCs is NP-hard. We must therefore focus on sufficient schedulability analysis. One approach consists of computing bounds on the worst-case end-to-end response times for any flow. In [6], [7], [3], the authors have considered this approach with a priority based transmission preemption method. For their analysis, they assume a virtual channel (VC) technique [8]. With VCs, each physical link has specific buffers along its path. Hence a transmitting packet can bypass a blocked one with this method. In the case of periodic constrained deadline flows, this helps adapting classical worst case response time analysis initially proposed in the context of uniprocessor systems to wormhole routing [3]. The holistic approach is then used to compute the worst case end-to-end response time of a flow by taking into account the worst case interference jitter of competing flows along its path. This work has been recently extended [9] to support flows having two criticality modes.

In the following section, we show that the trajectory approach, used in the context of switched Ethernet networks can be adaptated to the context of NoCs.

IV. Trajectory approach adapted for NoCs

The Trajectory approach was introduced for FIFO scheduling in [10] and then applied to real-time full-duplex switched Ethernet networks [11], [12]. This approach considers the worst-case scenario that a packet can face along its path to compute a bound on its WCTT.

More precisely, it maximizes each busy period[1] at each buffer along the path where competition occurs. The NoC architecture we study in this work shares common characteristics with real-time full-duplex switched Ethernet networks, like full-duplex links and static routing. We present

[1]A busy period of packet f_i with FIFO scheduling is defined by a temporal interval $[t_1, t_2)$ during which all the packets that arrive before or at the same time with f_i are transmitted and there is no idle time in (t_1, t_2).

in the following paragraphs how to adapt the Trajectory approach to the context of NoCs with FIFO policy. We explain its main differences with Switched Ethernet networks on a detailed example.

Let us consider packet f_i of a flow τ_i generated at clock cycle t_c. It is transmitted over the NoC following a path $\mathcal{P}_i = \{first_i, ..., last_i\}$ along which it crosses $|\mathcal{P}_i|-2$ routers where $|\mathcal{P}_i|$ represents the number of input buffers in the path (the first and last nodes are IPs).

Then according to the Trajectory approach, the WCTT of flow τ_i is bounded by:

$$D_i = \max_{0 \leq t_c \leq \lceil \frac{\mathcal{B}_i}{T_c} \rceil} \{W_{i,t_c}^{last_i} + C_i - t_c \cdot T_c\} \quad (1)$$

where $\mathcal{B}_i$ represents the maximum possible length of the busy period resulting from all flows following the same path as τ_i and $W_{i,t_c}^{last_i}$ is the latest starting time at the last visited buffer $last_i$ of flow τ_i computed by the following equation [10] (with $(x)^+ = max(0, x)$) :

$$\begin{aligned} W_{i,t_c}^{last_i} &= \sum_{\substack{j \in \{1,\dots,n\} \\ \mathcal{P}_i \cap \mathcal{P}_j \neq \emptyset}} \left(1 + \left\lfloor \frac{t_c \cdot T_c + A_{i,j}}{T_j} \right\rfloor\right)^+ \cdot C_j & (2) \\ & +(|\mathcal{P}_i| - 1) \cdot T_c & (3) \\ & +(|\mathcal{P}_i| - 2) \cdot T_c & (4) \\ & -C_i & (5) \end{aligned}$$

- Term 2 is the delay of packet f_i due to competition with other packets in all the output ports along its path, as well as the transmission delay generated by packet f_i itself. Term $t_c \cdot T_c + A_{i,j}$ represents the maximized interval during which packets of flow τ_j can arrive before or at the same clock cycle as f_i at the first node where flow τ_j joins the path of flow τ_i. This node is denoted $first_{i,j}$. Consequently, Term $(1 + \lfloor \frac{t_c \cdot T_c + A_{i,j}}{T_j} \rfloor)^+$ indicates the maximum number of packets generated by a flow τ_j that can delay the studied packet f_i.
- Term 3 is the sum of transmission delay from one node to the next along the path. Since wormhole switching is adopted and the transmission unit is a flit, the transmission delay from one node to the next is **one clock cycle** T_c per link. Term 3 represents this transmission delay along the path $\mathcal{P}_i$.
- Term 4 is the time for routing and arbitration along the path $\mathcal{P}_i$. Each router in the NoC takes **one clock cycle** T_c for routing and arbitration for each header flit. Since a packet f_i encounters $|\mathcal{P}_i|-2$ routers along its path, the induced delay along the path $\mathcal{P}_i$ is upper bounded by Term 4.
- Term 5 is subtracted because $W_{i,t_c}^{last_i}$ is the latest starting time of transmission of packet f_i at $last_i$. Since the transmission time C_i of f_i has been counted in Term 3, it should be subtracted from $W_{i,t_c}^{last_i}$.

More details on the classical Trajectory approach can be found in [10].

In order to better understand the adapted Trajectory approach, we illustrate it on the example of Figure 2. Each flow's temporal parameters are given in Table I. In this example, we consider the clock cycle $T_c = 1\ \mu s$.

τ_i	τ_1	τ_2	τ_3	τ_4
C_i (multiple of T_c)	4	4	4	4
T_i (multiple of T_c)	100	8	14	14

TABLE I. FLOW PARAMETERS

Consider flow τ_1 following the path $\mathcal{P}_1 = \{\text{IP}_{32}, R_{32\_L}, R_{22\_S}, R_{12\_S}, \text{IP}_{12}\}$. A packet f_1 of flow τ_1 is released by IP_{32} at clock cycle $t_c = 0$. The corresponding scenario is illustrated in Figure 3 where f_1 is marked by bold solid squares. Packet f_1 contains one header flit f_{1h} and three payload flits f_{11}, f_{12} and f_{13} which arrive at IP_{32} at clock cycles 0, 1, 2 and 3, respectively. The header flit f_{1h} advances along the path $\mathcal{P}_1$ and arrives at the input buffer of R_{32_L} at clock cycle 1. The payload flits advance in a pipeline way. A packet f_2 including four flits f_{2h}, f_{21}, f_{22} and f_{23} competes with packet f_1 for the North output port of router R_{32}. The header flit f_{2h} arrives at the same clock cycle 1 as the header flit f_{1h} and then it gets the access to the output first. After one clock cycle (clock cycle 2 marked by a cross in Figure 3 and Figure 4) dedicated for arbitration and transmission, f_{2h} arrives at the input port R_{22_S} of router R_{22} and reserves the input buffer for its following payload flits. Meanwhile, the flits of packet f_1 wait in the input buffer of R_{32_L}.

In this case, the clock cycle where packet f_2 can delay packet f_1 is clock cycle 1, and then $t_c \cdot T_c + A_{1,2} = 1\ \mu s$. According to Term 2, the delay of packet f_1 introduced by packets of flow τ_2 is computed by:

$$\left(1 + \left\lfloor \frac{t_c \cdot T_c + A_{1,2}}{T_2} \right\rfloor\right)^+ \cdot C_2 = \left(1 + \left\lfloor \frac{1}{8} \right\rfloor\right)^+ \cdot 4 = 4\ \mu s$$

The header flit f_{2h} continues to advance after clock cycle 4 (again dedicated for routing purposes) and arrives at the input buffer of R_{12_S} at clock cycle 5. The payload flits follow the header flit and at the same time free the input buffer of R_{22_S} which allows the flits of packet f_1 to advance. However, at clock cycle 7 where the header flit f_{1h} arrives at the input buffer of R_{22_S}, there are two more header flits f_{3h} and f_{4h} arriving at the same clock cycle and competing for the same output port in order to reach R_{12_S}. Consider the scenario when packets f_3 and f_4 are transmitted before packet f_1 since this is the worst-case scenario for packet f_1 with FIFO scheduling. In FIFO scheduling, the worst-case scenario in a buffer happens when all the frames arriving at the same time as the studied frame are transmitted before the studied frame. In that case, packet f_1 is blocked in router R_{22} till clock cycle 17 as illustrated in Figure 3.

The clock cycles where packet f_3 can delay packet f_1 are from clock cycle 3 to clock cycle 7, and then $t_c \cdot T_c + A_{1,3} = 5\ \mu s$. Similarly, we have $t_c \cdot T_c + A_{1,4} = 5\ \mu s$. Therefore, the delay of packet f_1 introduced by packets of flows τ_3 and τ_4 is computed by:

$$\begin{aligned} & \sum_{j \in \{3,4\}} \left(1 + \left\lfloor \frac{t_c \cdot T_c + A_{1,j}}{T_j} \right\rfloor\right)^+ \cdot C_j \\ = & \left(1 + \left\lfloor \frac{5}{14} \right\rfloor\right)^+ \cdot 4 + \left(1 + \left\lfloor \frac{5}{14} \right\rfloor\right)^+ \cdot 4 \\ = & \ 4 + 4 = 8\ \mu s \end{aligned}$$

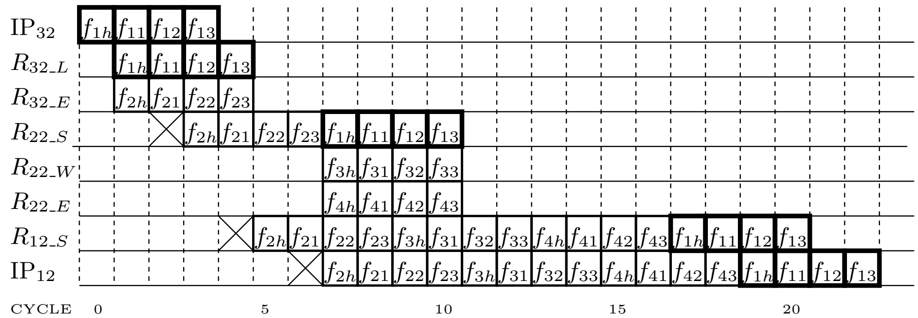

Fig. 3. Illustration on the scenario of $t_c = 0$

In addition to the delay introduced by other competing packets, there are also transmission delays and routing/arbitration delays calculated by Term 3 and Term 4:

$$(|\mathcal{P}_1| - 1) \cdot T_c + (|\mathcal{P}_1| - 2) \cdot T_c = 4 + 3 = 7\ \mu s$$

Finally, the header flit f_{1h} arrives at IP_{12} at clock cycle 19, i.e. $W_{1,t_c}^{\mathrm{IP}_{12}} = 19\ \mu s$. According to Equation 1, the delay of packet f_1, generated at clock cycle $t_c = 0$, is 23 μs.

We have illustrated how the Trajectory approach calculates the delay of a packet f_1 in the example of Figure 2 in the particular case where $t_c = 0$. For the Trajectory approach, each value of the release time t_c corresponds to a separate scenario. Contrary to the delay analysis in the context of uni-processors which only considers the synchronous scenario as the worst-case, the Trajectory approach verifies all possible scenarios in order to obtain bounds on the WCTT. This verification of all the possible scenarios is done by Equation 1. Indeed, in the example the worst-case scenario is not the one where $t_c = 0$. In the following paragraph, we present another scenario which leads to a worse WCTT for packet f_1.

Consider that packet f_1 is released at clock cycle $t_c = 8$. The header flit f_{1h} arrives at the input buffer of R_{32_L} at clock cycle 9 where the header flit f_{2h} arrives at the input buffer of R_{32_E} and therefore delays packet f_1. It then leads to the maximized interval $t_c \cdot T_c + A_{1,2} = 9\ \mu s$ for packets of flow τ_2. Note that there is another packet $f_2^{'}$ of flow τ_2 which arrives at the input buffer of R_{32_E} at clock cycle 1 due to its short period $T_2 = 8\ \mu s$. This scenario is illustrated in Figure 4. Packet $f_2^{'}$ does not delay packet f_1 for the output port of router R_{32}. However, packet $f_2^{'}$ has an indirect influence on packet f_1 which can be observed at router R_{22}. The delay introduced by packets of flow τ_2 is then calculated by:

$$\left(1 + \left\lfloor \frac{t_c \cdot T_c + A_{1,2}}{T_2} \right\rfloor\right)^+ \cdot C_2 = \left(1 + \left\lfloor \frac{9}{8} \right\rfloor\right)^+ \cdot 4 = 8\ \mu s$$

Packet $f_2^{'}$ advances along the path and arrives at the input buffer of R_{22_S} at clock cycle 3 after a clock cycle dedicated to routing. Meanwhile, a packet $f_3^{'}$ of flow τ_3 arrives at the input buffer R_{22_W} and a packet $f_4^{'}$ of flow τ_4 arrives at the input buffer R_{22_E} at the same clock cycle. Suppose that packet $f_2^{'}$ advances first before packets $f_3^{'}$ and $f_4^{'}$ and therefore delays packets $f_3^{'}$ and $f_4^{'}$ which eventually delay the transmission of packet f_2. Contrary to the scenario of $t_c = 0$ in Figure 3 where packet f_2 advances from router R_{22} to router R_{12} without being delayed and releases the input buffer of R_{22_S} immediately, packet f_2 is delayed for 2 clock cycles in the senario of $t_c = 8$ in Figure 4 and then releases the input buffer of R_{22_S} 2 clock cycles later. It imposes that packet f_1 stays at the input buffer of R_{32_L} for 2 more clock cycles due to a limited input buffer size of 4 flits. Packet f_1 waits till the clock cycle 17 where the header flit f_{21} advances and releases input buffer of R_{22_S}. This is the indirect influence introduced by packets $f_2^{'}$, $f_3^{'}$ and $f_4^{'}$ and the corresponding scenario of $t_c = 8$ is given at the part of R_{12_S} in Figure 4.

The header flit f_{1h} arrives at the input buffer of R_{22_S} resulting in the maximized interval $t_c \cdot T_c + A_{1,3} = t_c \cdot T_c + A_{1,4} = 15\ \mu s$ for frames of flow τ_3 and τ_4. Therefore, there can be another two header flits f_{3h} and f_{4h} arrive at the same clock cycle 17 due to their short periods $T_3 = T_4 = 14\ \mu s$ and packets f_3 and f_4 are transmitted before packet f_1. Accordingly, the delay introduced by packets of flows τ_3 and τ_4 is computed by:

$$\begin{aligned} & \sum_{j \in \{3,4\}} \left(1 + \left\lfloor \frac{t_c \cdot T_c + A_{1,j}}{T_j} \right\rfloor\right)^+ \cdot C_j \\ = & \left(1 + \left\lfloor \frac{15}{14} \right\rfloor\right)^+ \cdot 4 + \left(1 + \left\lfloor \frac{15}{14} \right\rfloor\right)^+ \cdot 4 \\ = & \ 8 + 8 = 16\ \mu s \end{aligned}$$

The summation of transmission delays and routing/arbitration delays is 7 μs, the same as calculated for scenario $t_c = 0$. Consequently, the latest starting time of packet f_1 is computed by $W_{1,8}^{\mathrm{IP}_{12}} = 31\ \mu s$. According to Equation 1, the WCTT of packet f_1 is obtained by: $D_1 = W_{1,8}^{\mathrm{IP}_{12}} + 4 - 8 = 27\ \mu s$

Obviously, the delay 27 μs obtained when $t_c = 8$ is worse than the delay 23 μs obtained when $t_c = 0$. The reason of the extra introduced delay is that some packets may not delay the packet under analysis at the beginning of the path, but can eventually delay it when they advance along the path, as for packets $f_2^{'}$, $f_3^{'}$ and $f_4^{'}$. The Trajectory approach verifies all the possible scenarios in order to compute a bound on the WCTT of flow τ_1. In this example, the obtained bound of flow τ_1 is equal to 27 μs when $t_c = 8$.

For the sake of simplicity, we consider an upper bound of the sum of the transmission delay and of the routing/arbitration delay. In order to do so, we combine Term 3 and Term 4 in

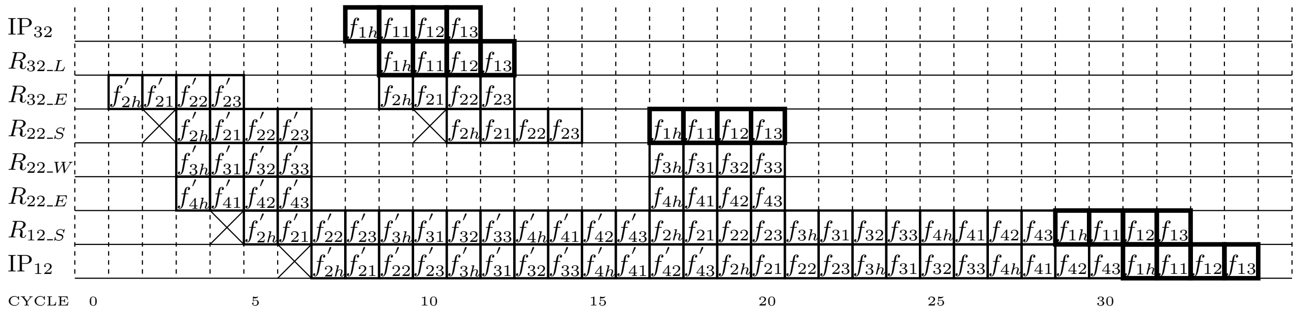

Fig. 4. Illustration on the scenario of $t_c = 8$

the following simplified computation formula of $W_{i,t_c}^{last_i}$:

$$
\begin{aligned}
W_{i,t_c}^{last_i} &= \sum_{\substack{j \in \{1,\ldots,n\} \\ \mathcal{P}_i \cap \mathcal{P}_j \neq \emptyset}} \left(1 + \left\lfloor \frac{t_c \cdot T_c + A_{i,j}}{T_j} \right\rfloor \right)^+ \cdot C_j \\
&\quad + (|\mathcal{P}_i| \cdot 2 - 3) \cdot T_c - C_i
\end{aligned} \tag{6}
$$

Discussion and Improvement: The computed delay D_1 is the exact WCTT of flow τ_1 as illustrated in Figure 4. However, it is not always the case for some flows. Let us take flow τ_3 in Figure 2 as an example.

Flow τ_3 under analysis follows a path $\mathcal{P}_3 = \{IP_{23}, R_{23\_L}, R_{22\_W}, R_{12\_S}, \mathrm{IP}_{12}\}$. The delay computation of flow τ_3 considers the delay introduced by packets f_1, f_2 and f_4 of flows τ_1, τ_2 and τ_4 at router R_{22}.

$$
\begin{aligned}
W_{3,t_c=0}^{\mathrm{IP}_{12}} &= \sum_{j \in \{1,2,3,4\}} \left(1 + \left\lfloor \frac{t_c \cdot T_c + A_{3,j}}{T_j} \right\rfloor \right)^+ \cdot C_j \\
&\quad + (|\mathcal{P}_3| \cdot 2 - 3) \cdot T_c - C_3 \\
&= 16 + 5 \times 2 - 3 - 4 = 19\ \mu s
\end{aligned}
$$

After the verification of all the possible scenarios, it is the one of $t_c = 0$ leading to the worst-case delay of flow τ_3 bounded by:

$$D_3 = W_{3,t_c=0}^{\mathrm{IP}_{12}} + C_3 - t_c \cdot T_c = 23\ \mu s$$

Indeed, packets f_1 and f_2 are both transmitted by the link $R_{32} \rightarrow R_{22}$. As they are serialized which means that one packet is transmitted after another, their header flits cannot arrive at router R_{22} at the same clock cycle. Therefore, only one packet (f_1 or f_2) can actually cause a delay to packet f_3 of flow τ_3. This scenario is illustrated in Figure 5.

The exact WCTT of flow τ_3 is then 19 μs, meaning that the computed delay $D_3 = 23\ \mu s$ is pessimistic but safe. The physical constraint is called packet serialization which has been integrated in the Trajectory approach in the context of switched Ethernet network [11] and has been revisited and corrected in [12] for an optimism problem. In order to improve the delay evaluation, it is important to integrate it in the formula in the context of NoCs.

The part of workload which cannot actually delay the packet under analysis at the router R_{xy} due to packet serialization is denoted by $\Delta_{i,t_c}^{R_{xy}}$. This serialization term is subtracted from Equation 6 and has been minimized in [11], [12] in order to guarantee the delay upper bound. As illustrated in the example of packet f_3, generated at clock cycle $t_c = 0$, of Figure 5, packet f_1 does not delay packet f_3 at router R_{22} which leads to $\Delta_{3,t_c=0}^{R_{22}} = C_1 = 4\ \mu s$. At the other nodes (routers or IPs) along the path $\mathcal{P}_3$, there is no packet serialization, i.e. $\Delta_{3,t_c=0}^{\mathrm{IP}_{23}} = \Delta_{3,t_c=0}^{R_{23}} = \Delta_{3,t_c=0}^{R_{12}} = \Delta_{3,t_c=0}^{\mathrm{IP}_{12}} = 0$. Therefore, the total effect of packet serialization is:

$$\sum_{h \in \mathcal{P}_3} \Delta_{3,t_c=0}^{h} = 4\ \mu s$$

According to the correction proposed in [12], a part of packet serialization is overlapped with the time interval from time origin 0 to the release time of packet f_i. The duration fo this time interval is $t_c \cdot T_c$. Then the corrected serialization term is:

$$\max\left(\sum_{h \in \mathcal{P}_3} \Delta_{3,t_c=0}^{h}, t_c \cdot T_c\right) = \max(4, 0) = 4\ \mu s$$

Therefore, the improved calculation of the latest starting time of packet f_3 at its destination IP IP_{12} is given by:

$$
\begin{aligned}
W_{3,t_c=0}^{\mathrm{IP}_{12}} &= \sum_{j \in \{1,2,3,4\}} \left(1 + \left\lfloor \frac{t_c \cdot T_c + A_{3,j}}{T_j} \right\rfloor \right)^+ \cdot C_j \\
&\quad + (|\mathcal{P}_3| \cdot 2 - 3) \cdot T_c - C_3 \\
&\quad - \max\left(\sum_{h \in \mathcal{P}_3} \Delta_{3,t_c=0}^{h}, t_c \cdot T_c\right) \\
&= 19 - 4 = 15\ \mu s
\end{aligned}
$$

which results in the WCTT of flow τ_3 bounded by:

$$D_3 = W_{3,t_c=0}^{\mathrm{IP}_{12}} + C_3 - t_c \cdot T_c = 19\ \mu s$$

With the integration of serialization term, the computation formula of $W_{i,t_c}^{last_i}$ for flow τ_i is improved by:

$$
\begin{aligned}
W_{i,t_c}^{last_i} &= \sum_{\substack{j \in \{1,\ldots,n\} \\ \mathcal{P}_i \cap \mathcal{P}_j \neq \emptyset}} \left(1 + \left\lfloor \frac{t_c \cdot T_c + A_{i,j}}{T_j} \right\rfloor \right)^+ \cdot C_j \\
&\quad + (|\mathcal{P}_i| \cdot 2 - 3) \cdot T_c \\
&\quad - \max\left(\sum_{h \in \mathcal{P}_i} \Delta_{i,t_c}^{h}, t_c \cdot T_c\right) - C_i
\end{aligned} \tag{7}
$$

V. CASE STUDY

In Figure 6, we consider a 4x4 NoC with 10 flows $\tau_1 \ldots \tau_{10}$ reaching three destinations and where each IP is indexed with the coordinates of its router. We focus on flow τ_1 following

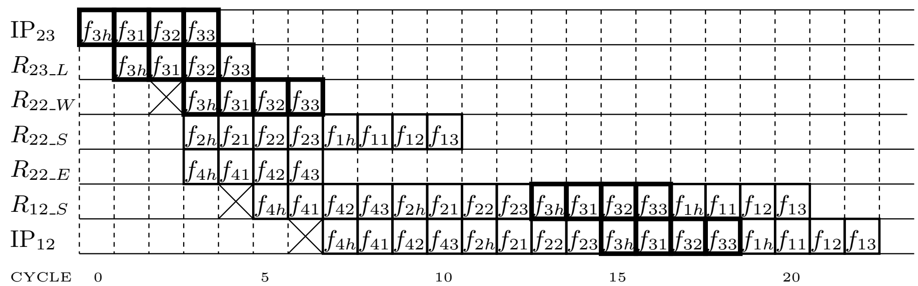

Fig. 5. Illustration on the scenario of packet f_3

the path $\mathcal{P}_1 = \{\mathrm{IP}_{32}, R_{32\_L}, R_{22\_S}, R_{12\_S}, \mathrm{IP}_{12}\}$. The paths of the other flows are:

$$\begin{aligned}
\mathcal{P}_2 &= \{\mathrm{IP}_{33}, R_{33_L}, R_{32_E}, R_{22_S}, R_{12_S}, \mathrm{IP}_{12}\} \\
\mathcal{P}_3 &= \{\mathrm{IP}_{23}, R_{23_L}, R_{22_E}, R_{12_S}, \mathrm{IP}_{12}\} \\
\mathcal{P}_4 &= \{\mathrm{IP}_{20}, R_{20_L}, R_{21_W}, R_{22_W}, R_{12_S}, \mathrm{IP}_{12}\} \\
\mathcal{P}_5 &= \{\mathrm{IP}_{10}, R_{10_L}, R_{11_W}, R_{01_S}, \mathrm{IP}_{01}\} \\
\mathcal{P}_6 &= \{\mathrm{IP}_{11}, R_{11_L}, R_{01_S}, \mathrm{IP}_{01}\} \\
\mathcal{P}_7 &= \{\mathrm{IP}_{13}, R_{13_L}, R_{12_E}, R_{11_E}, R_{01_S}, \mathrm{IP}_{01}\} \\
\mathcal{P}_8 &= \{\mathrm{IP}_{21}, R_{21_L}, R_{31_N}, \mathrm{IP}_{31}\} \\
\mathcal{P}_9 &= \{\mathrm{IP}_{22}, R_{22_L}, R_{21_E}, R_{31_N}, \mathrm{IP}_{31}\} \\
\mathcal{P}_{10} &= \{\mathrm{IP}_{00}, R_{00_L}, R_{01_W}, R_{11_N}, R_{21_N}, R_{31_N}, \mathrm{IP}_{31}\}
\end{aligned}$$

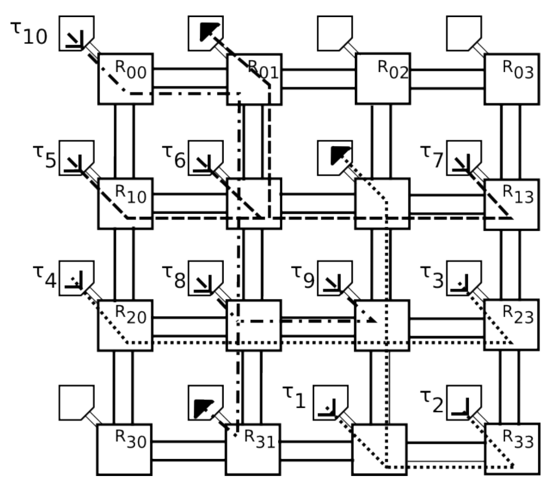

Fig. 6. NoC example of case study

All the 10 flows are with the same constant transmission time $C_i = 4 \times T_c = 4\ \mu s$. In Table II, we precise for each flow τ_i, its period T_i, its destination IP coordinates (xy) and the end-to-end delay D_i computed by formula 1. Note that flows τ_1, τ_2, τ_3 and τ_4 in the example in Figure 2 are integrated in this case study.

τ_i	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6	τ_7	τ_8	τ_9	τ_{10}
T_i (μs)	100	8	14	14	100	100	80	60	60	80
$\mathrm{IP}_{dest} : xy$	12	12	12	12	01	01	01	31	31	31
D_i (μs)	27	25	23	25	19	17	21	17	19	23

TABLE II. FLOW PERIODS AND COMPUTED DELAYS

VI. CONCLUSION

In this paper, we show how to characterize a bound on the WCTT in a NoCs with the Trajectory approach. We consider a NoC platform implementing FIFO scheduling and wormhole routing. We revisit the Trajectory approach, adapt it to the context of a NoC and show with an example the benefit it can provide. As a further work, we will characterize the pessimism brought by the Trajectory approach w.r.t. the exact WCTT obtained on a representative NoC platform.

REFERENCES

[1] W. J. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," in *Design Automation Conference, 2001. Proceedings*. IEEE, 2001, pp. 684–689.

[2] J. Flich, S. Rodrigo, J. Duato, T. Sodring, A. Solheim, T. Skeie, and O. Lysne, "On the potential of noc virtualization for multicore chips," in *Complex, Intelligent and Software Intensive Systems, 2008. CISIS 2008. International Conference on*. IEEE, 2008, pp. 801–807.

[3] Z. Shi and A. Burns, "Real-time communication analysis for on-chip networks with wormhole switching," in *Proceedings of the Second ACM/IEEE International Symposium on Networks-on-Chip*. IEEE Computer Society, 2008, pp. 161–170.

[4] N. K. Kavaldjiev and G. J. M. Smit, "A survey of efficient on-chip communications for soc," in *4th PROGRESS Symposium on Embedded Systems, Nieuwegein, The Netherlands*. Utrecht, The Netherlands: Technology Foundation STW, October 2003, pp. 129–140.

[5] L. Ni and P. McKinley, "A survey of wormhole routing techniques in direct networks," *Computer*, vol. 26, no. 2, pp. 62–76, Feb 1993.

[6] S. Hary and F. Ozguner, "Feasibility test for real-time communication using wormhole routing," *Computers and Digital Techniques, IEE Proceedings -*, vol. 144, no. 5, pp. 273–278, Sep 1997.

[7] B. Kim, J. Kim, S. Hong, and S. Lee, "A real-time communication method for wormhole switching networks," in *Parallel Processing, 1998. Proceedings. 1998 International Conference on*, Aug 1998, pp. 527–534.

[8] W. Dally, "Virtual-channel flow control," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 3, no. 2, pp. 194–205, Mar 1992.

[9] A. Burns, J. Harbin, and L. Indrusiak, "A wormhole noc protocol for mixed criticality systems," in *Real-Time Systems Symposium (RTSS), 2014 IEEE*, Dec 2014, pp. 184–195.

[10] S. Martin and P. Minet, "Schedulability analysis of flows scheduled with FIFO: application to the expedited forwarding class," in *Proc. of Int. Parallel and Distributed Processing Symposium (IPDPS)*. Rhodes Island, Greece: IEEE, Apr. 2006, pp. 8–pp.

[11] H. Bauer, J.-L. Scharbarg, and C. Fraboul, "Improving the worst-case delay analysis of an AFDX network using an optimized Trajectory approach," *IEEE Trans. Ind. Informat.*, vol. 6, no. 4, pp. 521–533, 2010.

[12] X. Li, O. Cros, and L. George, "The trajectory approach for afdx fifo networks revisited and corrected," in *Embedded and Real-Time Computing Systems and Applications (RTCSA), 2014 IEEE 20th International Conference on*, Aug 2014, pp. 1–10.