Minimizing Power Consumption of Spatial Division based Networks-on-Chip Using Multi-Path and Frequency Reduction

Sheng Hao Wang[1], Anup Das[2], Akash Kumar[2], Henk Corporaal[1]

[1]Department of Electrical Engineering, Eindhoven University of Technology, Netherlands
[2]Department of Electrical & Computer Engineering, National University of Singapore, Singapore
s.h.wang@student.tue.nl, akdas@nus.edu.sg, akash@nus.edu.sg, h.corporaal@tue.nl

***Abstract*—With an increasing number of processing elements being integrated on a single die, networks-on-chip (NoCs) are emerging as a significant contributor to overall chip power consumption. While some solutions have been proposed to reduce this power consumption, none of them can be applied to spatial division multiplexing (SDM)-based NoCs. In this paper, we introduce a method to minimize the power consumption of an SDM-based NoC by frequency minimization, while still satisfying the bandwidth requirements. The problem is integrated with the connection-routing problem which is modeled as a mixed-integer quadratic constrained problem (MIQCP). However, solving this MIQCP formulation directly using existing solvers is infeasible for large use-cases. We propose a two-step approach by first computing the minimum feasible frequency for the entire network taking bandwidth of all connections into consideration. This first step reduces the frequency-minimization-routing MIQCP problem into a routing-only mixed-integer linear programming (MILP) problem. In the second step, this MILP problem is solved using a standard ILP solver. Two other techniques are proposed to solve the routing and frequency minimization problem. Experiments are performed with synthetic examples and a case-study with JPEG decoder to evaluate the performance and results of the three methods. MILP-based approach achieves up to 55% power reduction as compared to the other methods albeit at the cost of higher execution time.**

***Index Terms*—Network-on-Chip; Spatial division multiplexing; power minimization; routing; mixed integer linear programming**

I. Introduction

The number of transistors on an integrated circuit doubles every 18 months according to Moore's law. Multiprocessor Systems-on-Chip are therefore being used to extract higher performance from modern day platforms. With many processors on a chip, scalable architectures for communication are necessary. Traditional buses are therefore being replaced with Networks-on-Chip (NoCs). NoCs provide an efficient on-chip communication architecture in terms of scalability, re-usability and high performance. Although NoCs have these advantages, there is still the issue of a high power consumption due to routing requirements. In portable devices, in particular, low power consumption is imperative.

A host of research has been done to reduce the power consumption of a NoC [1] [2]. While most of the networks are based on time-division-multiplexing (TDM), one of the efforts to reduce power is to use a Spatial-Division Multiplexed (SDM) NoC [3]. SDM networks do not require any switching at the routers once the connections are configured; it is essentially circuit switched once the data leaves the network interface. When the routing is done properly, SDM NoCs can provide guarantees on throughput and end-to-end latency since the connections do not need to compete for resources. Guaranteed throughput together with low area and power are key to the success of SDM-based NoCs for streaming applications. However, to the best of our knowledge, no automated techniques have been reported to route the required connections optimally.

As more and more processing elements are connected together using SDM-based NoCs, the power consumed in the network can become quite significant. In this paper, we introduce a method to further lower the power consumption of a Spatial-Division Multiplexed (SDM) NoC. Once the network is designed and the system is operational, power cannot be saved by reducing the physical resources. The toggle activity on the wires is application dependent and is beyond user control. So the only factors that can be tuned to reduce power are supply voltage and frequency.

Contributions: In this paper, we aim to operate the network at the lowest possible frequency given the bandwidth constraints of the application. Operating the network at a lower frequency implies that the supply voltage can be lowered resulting in reduced network power. Following are the major contributions of this paper:

- A mechanism to reduce network operating frequency while still meeting bandwidth requirements. This reduces dynamic power considerably by allowing the network to operate at a lower supply voltage.
- A MIQCP formulation of the integrated frequency-minimization and routing problem for SDM-based NoC.
- Reduction of the above MIQCP problem to MILP using optimal operating frequency computation.
- A non-LP based heuristic to solve the integrated problem using Dijkstra's shortest path algorithm.

The minimization of frequency is integrated together with the routing problem for an SDM-based NoC. The integrated problem is modeled as a Mixed Integer Quadratic Constrained Programming (MIQCP) problem. However, due to non-linearity in the model, standard commercial solvers like CPLEX [4] and AIMMS [5] are unable to solve the problem directly for large use-cases. We reduce the MIQCP problem to mixed-integer linear programming (MILP) problem by first calculating a set of *optimal* operating frequencies given the incoming and outgoing bandwidth of each network interface. These frequencies are used to determine the number of wires needed for all connections. The MILP problem, although being NP-hard, can be solved very efficiently using the aforementioned solvers. We also developed a non-LP based heuristic to solve the problem using the same optimal frequency computation function and Dijkstra's shortest path algorithm. We name the two approaches as follows:

- **PMAM:** Power Minimization Algo using MILP formulation.
- **PMAD:** Power Minimization Algo using Dijkstra's approach.

Both PMAM and PMAD are compared with brute force method where we limit our solutions to only shortest paths or paths with 1 detour. Further, there is a choice to backtrack only until path level or until wire level. This gives in total 4 versions for the brute force approach. In the event that a connection may require multiple wires to meet the bandwidth requirements, the formulation does not constrain the connection to use the same path for all wires. This gives more freedom to the algorithm and may potentially lead to multiple paths for the same connection. While this may pose significant problems for TDM networks, the design of network interfaces in SDM-based NoC ensures that data ordering is maintained at all times. Experiments are performed using synthetic and JPEG application graphs with networks ranging from 2×2 to 5×5 tiles. We observe that brute force method is not scalable and fails to provide results for networks above 3×3. PMAM gives the best results in terms of power savings providing up to 55% lower power as compared to PMAD. However, PMAD is more efficient in terms of execution time running up to 30x faster than PMAM.

The structure of the paper is as follows. First we give a short overview of the architecture used and the related works in section II. We explain the power model used in section III. In section IV we give an overview of the problem followed by the two approaches to solve this problem in section V and VI respectively. In section VII we show the results for synthetic experiments with network size up to 5x5 and for a case study with the JPEG decoder. The paper concludes with section VIII.

II. Related Work

There have been quite some efforts to reduce the power consumption in NoCs. Murali *et al.* have proposed to reduce the power consumption in the TDM-based NoC [6]. Their idea is to spread the traffic over the network using multi-path to reduce congestions and therefore a lower frequency can be used. However, their multi-path routing comes with an area overhead. We reduce the frequency of the network to lower the power consumption and by using unallocated wires of the network we satisfy the bandwidth requirements of connections. Therefore, there is no area overhead. Further, with our design we do not have to take into account for out-of-order arrival as long as the difference in hops between paths is less than 32 cycles due to the design of the network interface [7]. For networks smaller than 6x6, this is sufficient. Furthermore, they are using a packet-based network whereas we use a SDM-based network which provides guaranteed throughput.

Other works in the past tried to achieve a lower power by making changes in the hardware. These works were primarily targeted for NoCs based on time division multiplexing as well. Fattah *et al.* propose a high throughput low power FIFO buffer for a globally asynchronous locally synchronous (GALS) NoC [8]. Concatto *et al.* optimize the power using a dynamically reconfigurable router, but this also comes at the cost of more area [9].

SDM-based NoCs have been proposed as a low power and low area alternative to TDM-based NoCs [3]. In SDM-based NoCs, a subset of available wires is dedicated between a pair of PEs to form a connection. The number of wires allocated between two IP blocks depends on the bandwidth requirement. Each connection has exclusive usage of the wires assigned to it. Data from PEs is serialized at transmitter NI and sent over the wire. At the receiving end, data is de-serialized at the receiver NI.

An example of the SDM-based NoC is shown in Fig. 1. Three connections (A, B and C) are allocated some wires from router 00 to router 01. Connections A and B each get 1 wire and connection C gets 2 wires. 4 wires are still left unused.

The design uses a mesh topology for the routers and each router has an address to identify the router. This address consists of an x-coordinate and a y-coordinate based on their location in the mesh.

Each router has 5 ports. The north, south, west and east port are connected to a router in the corresponding direction. Further, there is a local port which is connected to the NI. Each of these ports has a fixed number of wires determined at design time. Further, each wire of a port has an index to identify the wire, which we call wire-index. The router is also a 1-way router, which means that each incoming wire can only switch to 4 other wires, all with the same index [7]. An example is shown in Fig. 2. The wires to the local network interface are not shown in the figure.

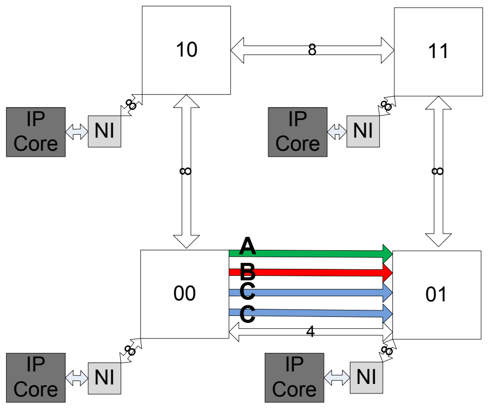

Fig. 1. SDM-based Network-on-Chip

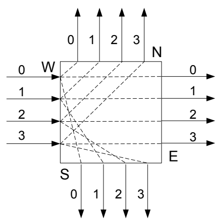

Fig. 2. 1-way Router

SDM-NoCs are power and area efficient as compared to the TDM-based NoCs. The authors in [3] observed 8% reduction in power and 31% reduction in area for an 8x8 NoC. These savings have the potential to drive future MPSoCs employing hundreds of cores.

Further, earlier works which involve SDM-based Networks-on-chip do not mention how the routing is done [10], [11], [12], [13]. The SDM-based Network-on-Chip from Leroy *et al.* [3] use the routing algorithm from Shickova *et al.* [14]. However, their routing algorithm does not guarantee an optimal routing. Their method is to check if a path is available from a router to another router and if it is not, it backtracks to the previous router and tries the other way. By doing this, some paths of other connections might not be possible anymore, which may lead to longer paths for those connections. With our approach that uses the MIQCP solver, every possibility is considered and an optimal routing is found if it exists.

III. Power Model of NoC Links

As technology is shrinking beyond 65nm, NoCs are becoming one of the major contributors to SoC power. In this section we develop a model for the dynamic power of the NoC links, switches and NI using equations formulated in [15], [16]. We also establish a relationship of power with the frequency of operation and supply voltage.

Our model is based on few key observations :

- The total power of NoC is the sum of the static and the dynamic power. i.e.
  $P_{NoC}^{total} = P_{NoC}^{static} + P_{NoC}^{dynamic}$
- Frequency is proportional to $(V_{dd} - V_t)^{\alpha}$ (using Sakuri α-power law model), where V_t is the threshold voltage.
- Voltage scaling is performed to reduce power even at deep submicron technologies.

Supply voltage scaling is an effective way of reducing both the static and dynamic power. Typically, scaling is done from nominal V_{dd} to $2/3V_{dd}$.

Once the network is designed, we have limited control on the static power. The reduction in static power comes from supply voltage scaling. In this paper, we propose to reduce NoC power at run-time and therefore the techniques developed here focuses on dynamic power and are orthogonal to any static power reduction technique. For the dynamic power, the activity factor is application dependent and is beyond user control. The only parameters that can be tuned for dynamic power reduction are supply voltage and frequency.

NoCs consists of three components:

- Links to transport traffic
- Switches to route traffic
- Network Interface to communicate with IPs

NoC Link Power

Links typically consist of one or more repeaters. The dynamic power of a link can therefore be split into two part as shown in equation 1.

$$P_{link}^{dynamic} = P_{wire}^{dynamic} + P_{repeaters}^{dynamic} \tag{1}$$

A. Repeater Dynamic Power

The dynamic power of a repeater is given by the equation 2.

$$P_{repeater}^{dynamic} = \alpha * [K * (C_g + C_d)] * f * V_{dd}^2 \tag{2}$$

Here, α is the activity factor, K is the size of the repeater, C_g and C_d are respectively the gate and the drain capacitance of the inverter, f is the frequency of operation and V_{dd} is the supply voltage. K can be assumed to be the average width of the nmos and pmos device.

B. Wire Dynamic Power

The dynamic power of the wire segment is given by equation 3.

$$P_{wire}^{dynamic} = \alpha * [C_w * L + \lambda * C_c] * f * V_{dd}^2 \tag{3}$$

C_w is the capacitance of the wire per unit length, C_c is the coupling capacitance with neighboring wire and λ is the coupling coefficients. For all practical purposes, λ can be assumed to be 1.51.

Using equation 2 and 3 we can rewrite equation 1 considering the total power of all the links as

$$P_{link}^{dynamic} = n * \alpha * K_{tech} * f * V_{dd}^2 \tag{4}$$

where K_{tech} is the technology constant and n is the number of links used.

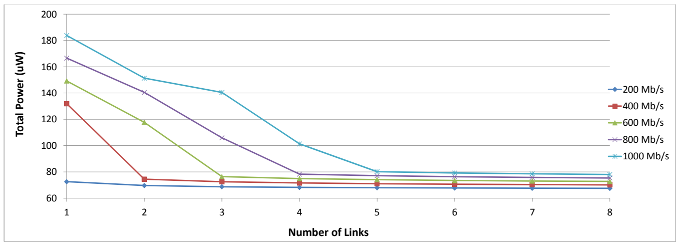

Fig. 3. Dynamic power-frequency dependency with varying number of links

NoC Switch Power

A switch in an SDM-based NoC is composed of a crossbar. The power of a switch is effectively, the power of crossbar. The dynamic power of the crossbar is given by equation 5.

$$P_{switch}^{dynamic} = n * n_{dir} * [\alpha * f * [C_w * L + \lambda * C_c] * V_{dd}^2] \tag{5}$$

where, n_{dir} is the number of directions of the switch (5 for a 1-way SDM-NoC router).

NoC Network Interface Power

In an SDM-based NoC, each link is driven by a serializer inside the NI. The major contributor to NI power are the serializers. The serializer power can be approximated by equation 6.

$$P_{serializer}^{dynamic} \propto n * [\alpha * f * [C_{ser}] * V_{dd}^2] \tag{6}$$

where, C_{ser} is the effective capacitance of the serializer. The exact derivation is omitted here.

Combining equations 4, 5 and 6, we can write equation 7 for the total dynamic power.

$$P_{NoC}^{dynamic} \propto \alpha * n * f * V_{dd}^2 \tag{7}$$

$n * f$ gives the bandwidth of the connection (denoted as B) and f is proportional to $(V_{dd} - V_t)$. Substituting these in equation 7, we get

$$P_{link}^{dynamic} \propto (B * \alpha * f^2) \tag{8}$$

Thus substantial power savings can be obtained by scaling down the frequency of operation of NoC. Figure 3 plots the power-frequency dependency using 65nm Predictive Technology Model. From the figure we see that, as we increase the number of links but keeping the bandwidth constant, power savings of 50% can be obtained. This directly correlates with the power model obtained in equation 8.

IV. Problem Formulation

In this work, we aim to minimize the frequency of the network, while still satisfying the bandwidth requirements of connections and constraints by the architecture. It is assumed that once the network is designed, the applications are also mapped onto the network by the designer. This mapping is often done at design-time since it is computationally intensive. Run-time analysis may not be able to provide throughput guarantees for the application. The mapping of the application tasks to various tiles determines the bandwidth requirements of connections between the various network interfaces. Each connection needs to be assigned a sufficient number of dedicated wires to meet these bandwidth requirements for guaranteed performance. Clearly, when the network is operated at a lower frequency, more wires may need to be assigned since available bandwidth decreases with a decrease in operating frequency. Arbitrarily reducing frequency may result in a non-routable situation while selecting a high frequency may lead to a situation where not all resources are used. Our algorithm finds the minimal frequency at which the network may be operated to satisfy bandwidth requirements of all connections without wasting resources. It should be noted that while modifying the application mapping may result in further lowering the operating frequency, it is orthogonal and beyond the scope of this paper.

In this section, we formulate the power optimization problem we want to solve. As mentioned earlier, the entire optimization is performed at design-time with sufficient compute power. We have shown that minimizing frequency is directly linked to minimizing power. The power consumed in NoC links is directly proportional to the number of links used, the supply voltage and the operating frequency. Lowering frequency alone does not have impact on the dynamic power as frequency reduction will entail more links to satisfy a given bandwidth requirement. However, an advantage of minimizing frequency is that the network itself can be operated at a much lower supply voltage. This can significantly reduce the static as well as the dynamic power. It should be noted that while for many cases, lowering frequency often results in longer execution time thereby mitigating the energy gains, in our case since data is communicated in parallel over more wires, the time taken for communicating the same amount of data does not increase. This implies that the power savings are equivalent to energy savings.

This frequency optimization problem is integrated with a routing problem which has been modeled as a Mixed Integer Quadratic Constrained Programming (MIQCP) problem [17] [18]. The objective is linear, but some constraints are non-linear. This problem is known to be NP-hard. The model is as following:

$$\begin{aligned}
&\textbf{Objective: } \min f_{network} \qquad (9)\\
&\textbf{subject to: } x_{m,r,c,d,i} \in \{0,1\}\\
&\quad m \in \{1, 2.., \text{nr of connections}\}\\
&\quad r \in \{0,1,.., \text{Rows}-1\}, c \in \{0,1.., \text{Cols}-1\}\\
&\quad d \in \{L_{in}, L_{out}, N, W, S, E\}\\
&\quad i \in \{0,1,.., W_p - 1\}
\end{aligned}$$

$$f_{network} \sum_i x_{m,r_{ms},c_{ms},L_{in},i} \geq B_m \qquad \forall m \qquad (10)$$

Where:

r_{ms} := Source row of connection m

c_{ms} := Source column of connection m

$$\sum_m x_{m,r,c,d,i} \leq 1 \qquad \forall r,c,d,i \qquad (11)$$

$$\sum_m \sum_i x_{m,r,c,d,i} \leq W_p \qquad \forall r,c,d \qquad (12)$$

Where:

W_p := Number of wires per port

$$x_{m,r_{ms},c_{ms},L_{in},i} - x_{m,r_{md},c_{md},L_{out},i} = 0 \qquad \forall m,i \qquad (13)$$

Where:

r_{md} := Dest. row of connection m

c_{md} := Dest. column of connection m

$$\begin{aligned}
&x_{m,r,c,L_{in},i} - x_{m,r,c,E,i} - x_{m,r,c,W,i} -\\
&x_{m,r,c,N,i} - x_{m,r,c,S,i} + x_{m,r,c-1,E,i} + \qquad \forall m,r,c,i \qquad (14)\\
&x_{m,r-1,c,N,i} + x_{m,r+1,c,S,i} + x_{m,r,c+1,W,i} +\\
&x_{m,r,c,L_{out},i} = 0
\end{aligned}$$

$$\begin{aligned}
&x_{m,r,c,L_{in},i} + x_{m,r,c,E,i} + x_{m,r,c,W,i} +\\
&x_{m,r,c,N,i} + x_{m,r,c,S,i} \leq 1 \qquad \forall m,r,c,i \qquad (15)
\end{aligned}$$

$$\begin{aligned}
&x_{m,r,c,E,i} + x_{m,r,c+1,W,i} \leq 1\\
&x_{m,r,c,W,i} + x_{m,r,c-1,W,i} \leq 1\\
&x_{m,r,c,N,i} + x_{m,r+1,c,S,i} \leq 1 \qquad \forall m,r,c,i \qquad (16)\\
&x_{m,r,c,S,i} + x_{m,r-1,c,N,i} \leq 1\\
&x_{m,r,c,L_{out},i} + x_{m,r,c,L_{in},i} \leq 1
\end{aligned}$$

In equation 9 the boolean variable x represents if a wire i is used or not for a certain connection m, router row r, router column c and which port of the router is from (north, west, south, east or the local port which is connected to the network interface). L_{in} indicates the direction from network interface to router and L_{out} from router to network interface. Note that there is no need to define an outgoing direction for the other ports, because an incoming direction from one port of a router is an outgoing direction from another port of another router. Whereas the same cannot be said for L_{in} and L_{out}. To indicate which wire we are using from a port, we have a wire index which is represented by the variable i. From this, we notice that the number of variables can become large if you have many connections and a large network, but all connections still share the same resources.

Equation 10 is a non-linear constraint, which sets the bandwidth requirement of the connections. The number of wires used, that is entering the router from the network interface, times the frequency of the network should be larger or equal to the bandwidth requirement of the connection.

Equation 11 tells that the wire can only be used by one connection and equation 12 sets the constraint that all the wires of a port should not exceed the number of wires per port W_p. If there is a wire used that is going into the network, there should also be a wire leaving which is represented by equation 13. Equation 14 sets the constraint that for a certain router, the number of wires coming in the router must be the same as the number of wires leaving and also takes the restriction of the 1-way router into account. So there is a lot of flexibility in choosing the path from one router to another router. Equation 15 and 16 prevents the formation of cycles on the paths.

A graphical way to represent the model is as multiple directed graphs $G(V,E)$, where there is a graph for each connection, and the routers are the vertices and the wires are the edges.

V. Power Minimization using MILP Formulation

In this section we describe our first technique of power minimization using the MIQCP formulation described in section IV. We tried to use mathematical programming solvers, such as CPLEX from IBM [4] and AIMMS [5], which incorporate multiple solvers. However,

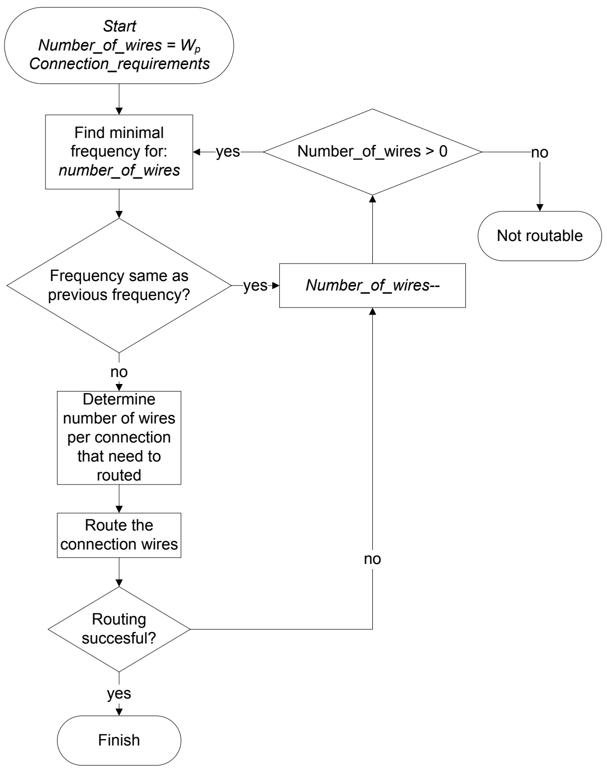
Start
Number_of_wires = W_p
Connection_requirements
Find minimal frequency for: number_of_wires
yes
Number_of_wires > 0
no
Not routable
Frequency same as previous frequency?
yes
Number_of_wires--
no
Determine number of wires per connection that need to routed
Route the connection wires
no
Routing succesful?
yes
Finish

Fig. 4. PMAM algorithm flow

these two software tools could not solve the model without guidance, because of the non-linear constraint set by the MIQCP equation.

Therefore, we limit the search space by first calculating the optimal set of frequencies for a certain number of wires available per port using Algorithm 1. The MIQCP problem can then be simplified to a Mixed Integer Linear Programming problem (MILP), but is still NP-hard. The commercial solvers solves the MILP problem with high execution-time efficiency. This MILP based power minimization technique is referred to as **PMAM**.

A. General overview

The general overview of the PMAM algorithm is shown in Fig. 4. Connection requirements specify if there should be a connection between two IP cores and how much bandwidth it requires. With the bandwidth requirement, we can determine what the minimum frequency of the network is for a given number of wires available per port W_p. Since the number of wires is an integer, only a maximum of W_p ideal frequencies are possible. Therefore, the algorithm starts with the highest possible number of wires that are available per port, calculates what the minimum frequency can be, taking all routers into account, determines how many wires it needs to route for all the connections and then tries to route all the wires one by one. If the routing is not successful, it decreases the number of wires available by 1 and calculates again the minimal frequency and so on. When the number of wires that are available reaches 1 and not all the wires can be routed then the connection requirements are considered as unsatisfiable with the current mapping and resources.

B. Finding minimum frequency

Finding the minimum frequency is achieved by calculating the minimum frequency for each router and taking the minimum value

Algorithm 1 Find_minimal_frequency(i)

```
Input: $i$, number of wires per port
Output: $freq$, minimum frequency
  for every router do
    for every incoming conn. from NI to that router do
      $Bandwidth$ += $bandwidth\_of\_conn$
    end for
    $t\_freq = \frac{Bandwidth}{number\_of\_wires}$
    repeat
      for every incoming conn. from NI to that router do
        $wires$ += $\left\lceil \frac{bandwidth\_of\_conn}{t\_freq} \right\rceil$
      end for
      if $wires > number\_of\_wires$ then
        find minimum frequency difference
        $t\_freq = \frac{conn\_bandwidth\_with\_min\_freq\_diff}{conn\_wires-1}$
      else
        $success = 1$
      end if
      $freq = max(freq, t\_freq)$
    until $success == 1$
  end for
```

over all the routers. Since we can only assign an integer number of wires to the connection i.e., a wire cannot be shared among connections, an extra step is needed to calculate the minimal frequency. The initial estimate for the minimal frequency of the SDM NoC is the total amount of incoming bandwidth divided by the number of wires available. Then the number of wires needed for every connection in the router is calculated and checked if it exceeds the number of wires available or not. If it exceeds, the minimum frequency is the minimum difference of all connections when 1 less wire is given to the connection. Algorithm 1 shows the algorithm in more detail. While only incoming bandwidth is illustrated in Algorithm 1, the outgoing bandwidth needs to be checked as well.

C. Routing

Once the optimal frequencies are obtained using Algorithm 1, the reduced MILP problem is solved using a commercial solver.

VI. Power Minimization using Dijkstra's Shortest Path

In this section we provide the details of our second approach of solving the integrated problem of power minimization and optimum routing using Dijkstra's shortest path algorithm. This is defined as **PMAD**. We start with the highest frequency first and then decrease it as shown in Algorithm 2. We begin with the frequency where only 1 wire per port is available and then increase 1-by-1. In this way, each connection gets at least 1 wire, which we try to route. The number of wires for connections are increased depending on their bandwidth requirements and the algorithm tries to route this. This continues till we reach the lowest frequency possible for the network or till the wires are not routable anymore. In the latter case, the routing goes back to the state of the previous frequency.

The biggest problems in routing the wires of this approach are choosing the paths optimally and the wire indices to use. Choosing a certain path blocks some wire indices which cannot be used anymore by other connections. In order to cope with this problem, Dijkstra's shortest path algorithm is used in this approach to find a path from one router to another one. The weights are chosen as the number of wires used in a link. In this way, the probability of finding a

Algorithm 2 Heuristic Approach for Minimizing Frequency

```
Input: Connection requirements and $W_p$
Output: Minimum frequency and routing
  $t\_noc\_routing = \phi$
  $conn\_wires = \phi$
  for $i = 1$ to $W_p$ do
    /*find min. frequency using Algorithm 1*/
    $freq = find\_minimal\_frequency(i)$
    if $freq == prev\_freq$ then
      $continue$
    else
      $prev\_freq = freq$
    end if
    /*backup routing*/
    $t\_noc\_routing = noc\_routing$
    for every connection do
      if $freq \times conn\_wires < bandwidth\_req\_conn$ then
        if route_wire() == false then
          $noc\_routing = t\_noc\_routing$
          $return$
        else
          $connection\_wires++$   /*routing succeeded*/
        end if
      end if
    end for
  end for
```

suitable path, that meets the restrictions of the 1-way router, is higher. In case the costs are the same for 2 paths, the path with the less amount of hops is chosen. It should be noted that all-pairs shortest algorithms cannot be used here since the wires cannot be shared between multiple connections.

VII. Results

In this section, we discuss and compare the results of the two approaches – PMAM and PMAD with brute force method where we made a distinction between backtracking at path level or at path and wire level, and with or without paths with 1 detour. This gives in total 4 versions for the brute force approach. In the event that a connection may require multiple wires to meet the bandwidth requirements, the formulation does not constrain the connection to use the same path for all wires. This gives more freedom to the algorithm and may potentially lead to multiple paths for the same connection. For the PMAM approach, we first find all the possible frequencies with Algorithm 1 and then use a commercial MILP solver to find a possible routing for the frequency. The PMAD technique is based on Dijkstra's shortest path algorithm.

Experiments are conducted using synthetic and JPEG application graphs with networks ranging from 2×2 to 5×5 tiles with 8 wires per port. The model of the NoC is implemented in Matlab using 65nm technology parameters from Predictive Technology Mappings [19]. The power numbers are obtained by simulations over multiple runs.

The results depend on many choices, such as the size of the network and specific applications. Choosing a larger network than required gives more flexibility in routing wires and has a higher success rate of lowering the frequency to the minimum, but this increases the power and area. Each application requires different connections and bandwidth requirements. Minimizing the frequency is easier for applications with a few connections than applications with many connections.

TABLE I
Results for various approaches for synthetic test-cases

Network Size (Connections)	Algorithms	Execution Time (s)	Min. Freq (MHz)	Power (μW)	%Savings
2x2 (6)	Single Wire	-	2800	951.6	-
	Path Backtracking	0.031	933.33	702.2	26.2
	Wire backtracking	1.75	700	647.1	32
	Path Backtracking: 1 detour	0.109	700	695.1	26.9
	Wire backtracking: 1 detour	163.27	700	647.1	32
	PMAD	0.047	700	695.1	26.9
	PMAM	0.080	700	647.1	32
3x3 (8)	Single Wire	-	3200	4723	-
	Path Backtracking	0.125	400	2662	43.6
	Path Backtracking: 1 detour	0.125	400	2841	39.8
	PMAD	0.109	457.33	3217	31.8
	PMAM	0.67	400	2662	43.6
4x4 (16)	Single Wire	-	3200	20919	-
	PMAD	0.203	533.33	15475	26
	PMAM	3.98	400	9139	56.3
5x5 (26)	Single Wire	-	3200	54660	-
	PMAD	0.297	800	50573	7.5
	PMAM	9.55	600	22705	58.5

A. Synthetic Testcases

Synthetic test cases were generated to test and evaluate the algorithms. We generated cases for 2x2, 3x3, 4x4 and 5x5 network, where connections from source to destination were randomly generated and the bandwidth requirements of the connections were chosen randomly between 400 and 3,200 Mbit/s.

The results can be found in Table I. As can be seen in this table, the backtracking algorithms do not scale well. The execution time increases exponentially with the network size; for network of sizes larger than 3x3, the algorithms do not even finish after 1 hour and even then they were still trying to find a route for the first frequency. For the 2x2 case, the *path-level* backtracking algorithms do not achieve the minimum frequency of 700 MHz, while the other algorithms do. This is probably because the search space of the *path-level* backtracking algorithm is more limited. The PAMD approach is fast. Even for a 5x5 case it only takes 0.297 seconds to obtain a solution, but it does not achieve the minimum frequency in most cases. PMAM approach handles large problems well and gives a solution within 10 seconds for a network size of 5x5 with the lowest frequency possible. Fig. 5 shows execution times that various approaches require to achieve their respective frequencies. The execution times have been normalized with respect to PMAM execution time. Please note that *wire-level* backtracking algorithms are not shown in the figure since they take a very long time to execute.

Table I also reports the power and percentage of power savings for each of the algorithms. The power numbers are obtained using simulations with technology parameters from Predictive Technology Mappings [19]. Since no standard routing algorithms are available for SDM NoCs, we have added the power for transmission using a

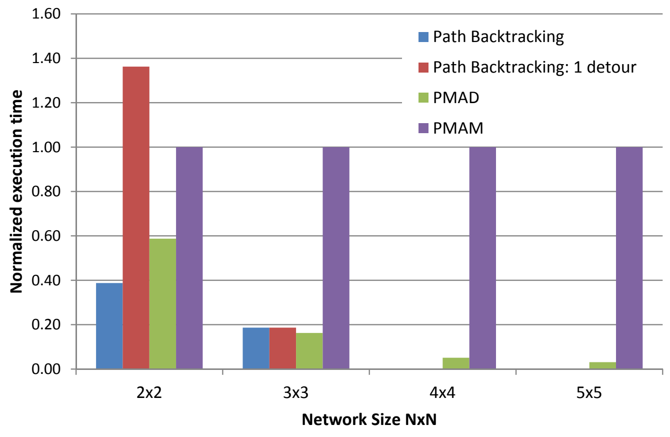

Fig. 5. Normalized execution time for various approaches

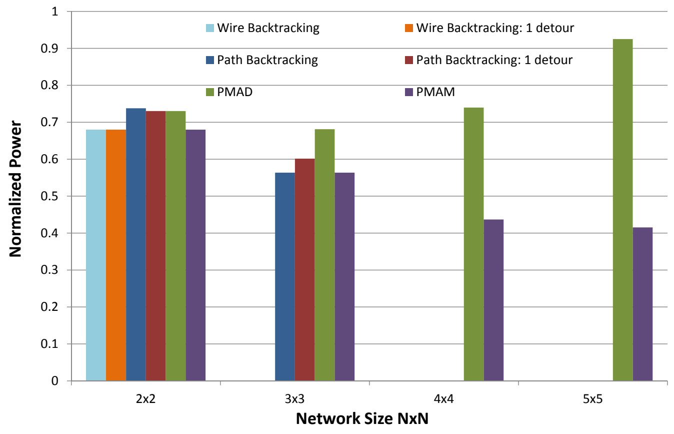

Fig. 6. Normalized power for various approaches

single wire from each NI for reference. The percentage power savings calculated is with respect to the power of using a single wire for each connection. As can be seen from this table, for each of the four configurations studied, PMAM provides the best result with power savings of up to 58%. We have not included the actual number of wires used for each of the algorithms in the table. A point to note is that, for the 4x4 and 5x5 configuration, the power reduction is not in proportion with the frequency reduction obtained. For the 5x5 configuration, the power savings using PMAD algorithm is only 7.5% as compared to PMAM where it is 58.5% with minimum frequencies as 800MHz and 600MHz respectively. This is because, the number of wires that are used in the solution using the PMAD algorithm (306) is much more than that from PMAM (188). The 4x4 network configuration has the same behaviour. For all the other configurations, the number of paths from both the algorithm are comparable and so the power savings are more in relation to the frequency. A bar graph of the normalized power is shown in Fig. 6 to give a visual idea of performance of various algorithms.

B. JPEG decoder

As a real test case we used a JPEG decoder. The bandwidth requirements are calculated using the SDF graph shown in Fig. 7 [20]. The execution times are shown in cycles and we assume the actors run on a 3 GHz processor and the size of a token is 64 bytes. The resulting bandwidth requirements and mapping can be found in Table II.

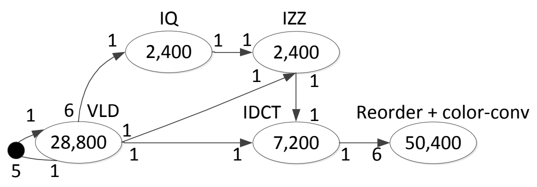

Fig. 7. SDF graph JPEG decoder

Table III shows the results of various approaches along with the power savings obtained. In this table the number of wires used from and to the network interface are not included. The minimum frequency achieved is 213.3 MHz, which is actually the minimum frequency the network can operate at without violating the bandwidth requirements. As shown in Table II, there are 3 connections leaving from router 00 with a total bandwidth requirement of 1333.8 Mbit/s, which need to share the 8 wires. This results in a frequency of $\frac{1333.8}{8} = 166.8$ MHz. However, operating at this frequency leads to total 9 wires, which is too much. The minimum frequency would be therefore $\frac{640.2}{3} = 213.3$. All algorithms except the one without *wire-level* backtracking achieve the minimum frequency possible. At this lowest frequency, it is easy to verify that the total number of wires required is 22.

We can see from Table III that the algorithm with *wire-level* backtracking takes more time. However, it does achieve the lowest frequency possible with only shortest paths. So there is no latency difference between the multi-paths. One reason that the algorithm with only *path-level* backtracking does not achieve the minimum frequency is that a specific connection can only use one index which is already occupied by another connection. However, if another connection can be moved to another index, the wires can be routed. By including paths with 1 detour, the backtracking algorithm can achieve a lower frequency at the cost of using more wires and a longer execution time.

PMAM execution time is 90 ms and achieves the lowest frequency possible. This is a good result, because PMAM considers all possible paths to achieve the minimum frequency and it takes a reasonable time to solve the model.

The PMAD algorithm with Dijkstra's shortest path is very fast and also achieves the lowest frequency possible for the network in this case. This is due to the fact that the algorithm chooses a path which uses the least number of wires, which implies that the chance of finding an available index is higher.

The power results for the JPEG test case are also shown in the table. Power saving increases as we reduce the frequency of operation. This is evident from the results of *path-level* backtracking and PMAM. The PMAM with 22 wires operating at 213MHz achieves lower power than *path-level* backtracking with 15 wires operating at 320MHz.

Thus, both the synthetic test cases and the JPEG use case concludes that around 50% power savings can be obtained by using more wires for communication while operating them at a lower frequency. A point to note is that, the design is fixed and so is the delay of the network. The algorithm only uses a lower frequency over more links to communicate the same data. Hence, the energy savings is directly proportional to the power savings obtained.

TABLE II
CONNECTION REQUIREMENTS FOR JPEG DECODER

Connection	Source row	Source column	Dest. row	Dest. column	Bandwidth (Mbit/s)
1: VLD - IQ	0	0	1	0	53.4
2: VLD - IZZ	0	0	0	1	640.2
3: VLD - IDCT	0	0	1	1	640.2
4: IQ - IZZ	1	0	0	1	640.2
5: IZZ - IDCT	0	1	1	1	640.2
6: IDCT - Reorder and Color-conv	1	1	1	0	640.2

TABLE III
RESULTS ALGORITHMS JPEG DECODER

Algorithm	Exec. Time (ms)	Min. Frequency (MHz)	Number of wires	power (μW)	%savings
Single Wire	-	640.2	8	238.3	-
Path Backtracking	47	320.1	15	208.1	12.6
Wire backtracking	3700	213.3	22	122.7	48.5
Path Backtracking: 1 detour paths	78	213.3	28	136.2	42.8
Wire backtracking: 1 detour paths	42090	213.3	22	122.7	48.5
PMAD	62	213.3	22	122.7	48.5
PMAM	90	213.3	22	122.7	48.5

VIII. CONCLUSIONS AND FUTURE WORK

We proposed an approach to save power in a SDM-based NoC by reducing the frequency. We also model the routing as a Mixed Integer Quadratic Constrained Programming problem. Since the problem cannot be directly solved, we have proposed a simple solution and three approaches to find the lowest frequency that the network can be operated at while still providing bandwidth guarantees. The minimum frequency that satisfies all constraints can be achieved and we provide a solution in the form of a routing. In designing a NoC, choosing the frequency of the NoC does not need to be taken into account anymore. Only the bandwidth requirements and mapping need to be supplied for our approach. We also propose a non-LP based heuristic to solve the integrated problem of frequency-minimization and optimal routing.

In terms of speed, the PMAD is fast. In the worst case it has to only route W_p wires per connection and the performance of routing a wire is the same as the worst case performance of Dijkstra's shortest path algorithm, which is $O(|E| + |V| \log |V|)$, where E is the number of wires and V the number of routers. However, there is no guarantee that it will reach the minimal frequency that can be used. PMAM on the other hand are a good compromise on the execution time and the minimum frequency acieved.

Future work. In the future, the brute force approach with backtracking can be made faster by adding more heuristics. Another possibility is to improve the algorithm which uses Dijkstra's shortest path algorithm, to get a more optimal result. For example, when the shortest path fails, the second shortest path can be tried.

Further, the case study was only done for a 2 by 2 network where the number of possible paths is very small and also the number of connections. For larger networks the problem grows exponentially as the number of possible paths grows exponentially. Larger networks also imply that the number of connections can also be more which means the depth of the backtracking can become very large.

Another possibility is to use a different objective function in CPLEX. In this case, we wanted the least amount of wires to save power, but in other scenarios least latency may be desirable.

REFERENCES

[1] K. Lee, S. Lee, and H. Yoo, "Low-power network-on-chip for high-performance soc design," *IEEE Transactions on Very Large Scale Integration Systems (TVLSI)*, 2006.

[2] Y. Hu, Y. Zhu, H. Chen, R. Graham, and C. Cheng, "Communication latency aware low power noc synthesis," in *ACM Design Automation Conference (DAC)*, 2006.

[3] A. Leroy, D. Milojevic, D. Verkest, F. Robert, and F. Catthoor, "Concepts and implementation of spatial division multiplexing for guaranteed throughput in networks-on-chip," *IEEE Transactions on Computers (TC)*, 2008.

[4] IBM ILOG CPLEX SOLVER. [Online]. Available: http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/

[5] AIMMS. [Online]. Available: http://www.aimms.com/aimms/

[6] S. Murali and et al, "A multi-path routing strategy with guaranteed in-order packet delivery and fault-tolerance for networks on chip," in *ACM Design Automation Conference (DAC)*, 2006.

[7] Z. Yang, A. Kumar, and Y. Ha, "An area-efficient dynamically reconfigurable spatial division multiplexing network-on-chip with static throughput guarantee," in *IEEE International Conference on Field-Programmable Technology (FPT) on*, 2010.

[8] M. Fattah, A. Manian, A. Rahimi, and S. Mohammadi, "A high throughput low power fifo used for gals noc buffers," in *IEEE Symposium on VLSI (ISVLSI)*, 2010.

[9] C. Concatto, D. Matos, L. Carro, F. Kastensmidt, A. Susin, and M. Kreutz, "Noc power optimization using a reconfigurable router," in *IEEE Symposium on VLSI (ISVLSI)*, 2009.

[10] J. Lim, E. Hunt Siow, Y. Ha, and P. Meher, "Providing both guaranteed and best effort services using spatial division multiplexing noc with dynamic channel allocation and runtime reconfiguration," in *IEEE Conference on Microelectronics (ICM)*, 2008.

[11] M. Modarressi, H. Sarbazi-Azad, and M. Arjomand, "A hybrid packet-circuit switched on-chip network based on sdm," in *IEEE Conference on Design, Automation and Test in Europe (DATE)*, 2009.

[12] P. Zhang, L. Fang, D. Li, and W. Jiang, "A new scheme of space division multiplexing and its analysis," in *IEEE Conference on Power Electronics and Intelligent Transportation System (PEITS)*, 2009.

[13] G. Lee and et al, "Mapping multi-domain applications onto coarse-grained reconfigurable architectures," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 2011.

[14] A. Shickova, T. Marescaux, D. Verkest, F. Catthoor, S. Vernalde, and R. Lauwereins, "Architecture exploration of interconnection networks as a communication layer for reconfigurable systems," *Annual Workshop on Circuits, Systems, and Signal Processing*, 2003.

[15] J. Balfour and W. J. Dally, "Design tradeoffs for tiled cmp on-chip networks," in *ACM Conference on Supercomputing*, 2006.

[16] M. Kim, D. Kim, and G. Sobelman, "Network-on-chip link analysis under power and performance constraints," in *IEEE Symposium on Circuits and Systems (ISCAS)*, 2006.

[17] CPLEX features. [Online]. Available: http://www.aimms.com/features/solvers/cplex

[18] E. R. R. W. Cottle, J. Pang Stone, *The linear complementarity problem.*

[19] Ptm. [Online]. Available: http://www.ptm.asu.edu

[20] A. Kumar, B. Mesman, B. Theelen, H. Corporaal, and Y. Ha, "Analyzing composability of applications on mpsoc platforms," *Journal of Systems Architecture*, 2008.