An Optimisation Algorithm for Minimising Energy Dissipation in NoC-based Hard Real-Time Embedded Systems

M. Norazizi Sham Mohd Sayuti
Real-Time Systems Group
Department of Computer Science
University of York
azizi@cs.york.ac.uk

Leandro Soares Indrusiak
Real-Time Systems Group
Department of Computer Science
University of York
lsi@cs.york.ac.uk

Alberto Garcia-Ortiz
Institute of Electrodynamics and Microelectronics
University of Bremen
agarcia@item.uni-bremen.de

ABSTRACT

This paper introduces an evolutionary multi-objective optimisation algorithm to facilitate fast and efficient task allocation of hard real-time embedded systems with Networks-On-Chip (NoC) as the interconnection at the early design stage, where evaluating as many as possible solutions is crucial. Our approach uses analytical fitness functions to provide fast evaluation of large number of solutions; a contrast to simulation-based optimisation technique, whereby it tends to be not only impractical when the design space is very large but also unfeasible as far as hard real-time systems are concerned. The proposed algorithm guarantees the predictability in timing behaviour of the systems whilst minimising energy dissipation whenever tasks are reallocated and their packets are rerouted, which differs from the state-of-the-art approaches. In addition, not only it can explore the allocation of tasks but also the encoding of the data packets. The evidence gathered from case studies shows that the proposed algorithm is able to find schedulable allocation of tasks, preserving it whilst further minimising energy dissipation.

1. INTRODUCTION

This paper addresses the early design space exploration problem in NoC-based hard real-time embedded systems design for finding the allocation of tasks which can meet the hard real-time constraints and low energy requirement. Our approach to this problem is an evolutionary multi-objective optimisation that uses analytical fitness functions to calculate the energy dissipation of NoC and to validate the schedulability of any task and communication message in the systems. The integration of such functions into the optimisation algorithm is an essential step to enable fast and efficient coverage of the design space.

The design of NoC-based hard real-time embedded systems must take into account the schedulability of the systems besides other objectives such as meeting the energy requirement. Such systems differ with non-real-time systems in such a way that just having average performance is not sufficient but each of the application task and message must be ensured to be schedulable; if not the systems cannot be guaranteed to be predictable. This can be achieved by ensuring that each of them must be able to meet their respective deadline. If a deadline is missed, their expected timing behaviour will be interrupted and no guarantee that the systems will behave accordingly.

However, the state-of-the-art energy-aware task allocation approaches for NoC based embedded systems lack the ability to validate the strictness of the timing constraints for this type of systems, limiting their application to only average case. Every time each task is reallocated to another core and its message is rerouted, significant changes in response time of the task and network latency of its message occur. In a worst-case scenario, maximum interference is likely to happen and the expected behaviour of the systems might be disrupted, thus the worst-case response time of the task and worst-case latency of its message must be analysed in advance. Allocating tasks near to each other in order to reduce the number of network hops is not sufficient, though might be effective in reducing the energy consumption in average case.

The remainder of this paper is organised as follows. Section 2 discusses the background of this research and section 3 explains the system model that is used by the proposed algorithm. In section 4, we propose an evolutionary multi-objective optimisation algorithm to address the problem of finding energy-aware task allocation for NoC based hard real-time embedded systems. Section 5 and 6 introduce the fitness functions for evaluating every allocation of tasks. Then followed by case studies in section 7 to illustrate the feasibility of our approach and lastly we conclude the contribution in section 8.

RTNS 2013 , October 16 - 18 2013, Sophia Antipolis, France

http://dx.doi.org/10.1145/2516821.2516844.

2. BACKGROUND RESEARCH

Many researchers had used simulation in multi-objective optimisation to evaluate performance and energy for NoC-based systems. Ascia et al. [1] used simulation to evaluate allocation of tasks in their GA-based optimisation with the targeted objectives are to minimise latency and aver-

age power consumption. However, simulation runtime poses severe limitation to early exploration of design when every solution in very large design space needs to be evaluated under highly detailed execution model. Unfavourable impact of the high simulation time hinders many alternative solutions from being considered, leading to inefficient design space exploration. A common technique such as pruning the design space [10] is normally used to reduce the number of simulation runs. However, even with the reduced number of simulation runs, a simulation still depends on the appropriate stimulus to evaluate worst-case scenario. It is a challenging task to construct a stimulus that could really perform this scenario (if any can be found); hence it becomes a hurdle to validate the hard real-time constraints with this technique.

Other works [18, 6, 9] applied analytical methods to evaluate task allocation but did not address the optimisation problem of hard real-time embedded systems. In [18], the authors proposed a GA-based optimisation approach to minimise latency and average energy consumption between switches. Their assumption is that both metrics can be reduced by minimising the number of average network hops required by packets to travel over the NoC. However, route minimisation can increase packet contention and such negative impact on latency is not taken into account in their evaluation model.

Jena [6] proposed a two-stage optimisation technique, targeting minimisation of average power consumption and cost, however minimisation of latency is not considered in their approach. Similar to two-stage optimisation in [6], Nedjah et al. [9] proposed an optimisation technique to minimise power, area and execution time. The contention in the transmission channel is taken into account by applying time penalty, which is the product of total flits and the time taken when travelling through channel. However, this does not take into account how much interference and latency each flow will receive when contention occurs in the channel.

The only approaches that can statically allocate tasks to multiple cores in NoC-based hard real-time embedded systems were based on single-objective optimisation [7, 14]. However, these approaches lack insight on the impact of task allocation to the energy dissipation of NoC. In our previous work [15], we have done the multi-objective optimisation, but in this paper we consider multi-objective optimisation with more sophisticated energy macromodels and optimisation based on encoding.

3. DESIGN METHODOLOGY & SYSTEM MODEL

The proposed optimisation algorithm is aimed to find static allocation of tasks at system level design. At this level of abstraction, the application and platform models resulted from separation of concerns can be combined by mapping tasks to the cores. Remapping of both models and the analysis of performance and energy can be iteratively performed until it stops when a termination condition is reached. Figure 1 shows how the algorithm performs the mapping process.

An application model describes the design-time characteristics of computation and communication load imposed by an application. It consists of a task set and a set of messages; tasks that are assigned at different cores send messages to

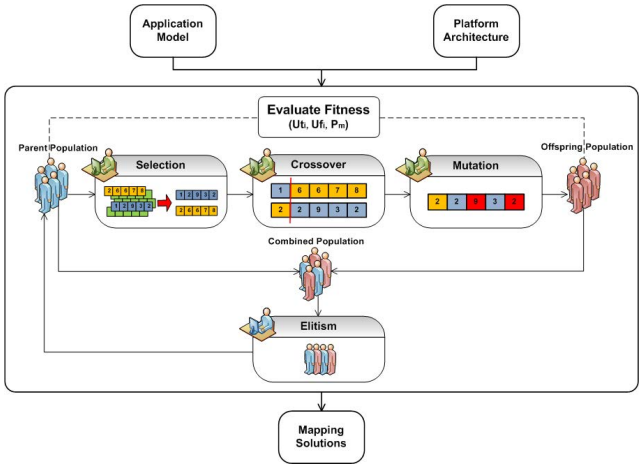


Figure 1: The multi-objective optimisation flow

communicate between each other. Packetisation of a message before its transmission produces one or several packets. A traffic flow represents a series of packets sent over the NoC from a source task to a destination task.

Following a common practise in real-time analysis (RTA) based system design, tasks and their respective messages can be described as the following tuples. The application model is characterised with timing properties (e.g. period T and deadline D) because RTA normally requires these properties to calculate the upper bounds of task response time and flow network latency.

$Task = \{C, T, V, D\}$

- C the task computation time
- T periodic interval of the task release
- V priority level of the task
- D deadline of the task's end-to-end response time

$Msg = \{So, De, T, V, F\}$

- So sender of the message
- De receiver of the message
- T periodic interval of successive packet release
- V priority level of the message
- F length of the message

We assume that the end-to-end response time upper bound of a task that communicate with other task in different core includes the amount of time since its release until the last packet that it sends arrive at the destination router in worst-case scenario. In our assumption, we also consider that a task transmits a message after it has finished its execution and the overhead of the network interface is negligible. In order to meet the hard real-time constraints, a task's end-to-end response time must not exceed its deadline (D).

The platform is modelled to represent the architecture of an on-chip multicore system that uses NoC as the interconnection. As shown in Figure 2, we implemented the platform as 2D Mesh with a NoC architecture that consists of many

routers with deterministic routing and wormhole switching. The arbitration unit supports flit-level pre-emption to control the access of packets to shared links. A processing core can run more than one tasks and a link can be shared by multiple traffic flows. The scheduling policy of tasks and flows is fixed-priority pre-emptive policy.

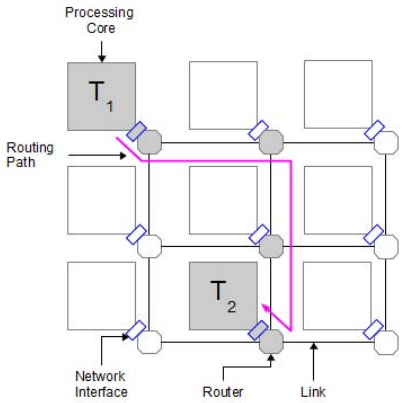


Figure 2: NoC-based system model

4. OPTIMISATION ALGORITHM

The proposed multi-objective optimisation algorithm (MO GA) is based on the work published in [3], which uses the concept of evolution in a population to produce better individuals. The improvement of the individuals is a continuous process until a termination condition is reached. During optimisation, the population undergo three transformations: the parent population, the offspring population and the combined population. The overview of the process flow of the algorithm is shown in Figure 1.

In this algorithm, an individual represents a configuration that contains instructions on how to allocate tasks to cores and how to configure the encoding of traffic flows. Each individual is a possible solution to the problem and can be defined by a uniform representation called chromosome; a chromosome is built upon a set of small units called genes. Figure 3b shows the chromosome format of the proposed algorithm. The chromosome is partitioned into two main parts, representing a set of tasks (green) and a set of traffic flows (orange). In the first part, each gene contains a location of the core (an integer index) on which a task is assigned to in the system. For the second part, each gene contains a binary decision (0 or 1) that decides if the traffic flow should be encoded or not. The total number of tasks and traffic flows of the application determines the total length of the chromosome.

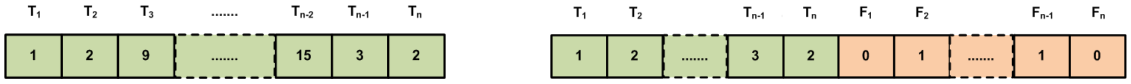
Every time a new individual (or configuration) is produced, it will be evaluated by the fitness functions (Ut_i, Uf_i, P_m) to determine its quality. The first fitness function (Ut_i) is used to calculate the number of unschedulable tasks. The second fitness function (Uf_i) is used to calculate the number of unschedulable traffic flows. The third fitness function (P_m) is used to calculate the total energy dissipated by NoC. Before the evaluations can be performed, tasks are assigned to the cores as given by the configuration and then their traffic flows are routed. Then, the worst-case response time of each task and the worst-case latency of each flow is calculated and compared with their deadline. A traffic flow will be encoded and its energy dissipation is calculated if it is true (or left unencoded if false). The total energy dissipated by NoC is given by sum of the energies dissipated by all flows.

In the early phase of evolution, the parent population contains a set of randomly generated individuals. Every individual is assigned with two quality attributes [3] known as non-domination level and crowding distance. These attributes are determined based on values calculated by the fitness functions. Once a parent population is created, its individuals will be manipulated by four type of operators that perform specific functions relevant to evolution process. These operators are known as selection operator, crossover operator, mutation operator and elitism operator.

The selection operator selects two individuals to become parents based on binary tournament selection procedure. This procedure randomly selects two individuals and determines a winner through comparison of the quality attributes. An individual who has lower non-domination level than its opponent becomes the winner. However, when two individuals have the same non-domination level, one with larger crowding distance is considered as the winner. The winner becomes a parent and the same procedure is repeated to choose its mate.

In a mating process or crossover, genes from both parents are exchanged to produce new offspring. The amount of genes that are exchanged depends on a crossover point drawn across both chromosomes of the parents; hence its name is single-point crossover. This point is randomly determined and all genes at the right side of the point are exchanged whereas genes at the left side remain unchanged. However, not all parents are being manipulated by the crossover operator; some are passed to the next generation without crossover depending on a probability called crossover rate. After crossover, a part of genes in offspring chromosome closely resembles genes of their parents. Hence, population may be filled with individuals that are roughly the same kind as their parents. In order to promote diversity, mutation is performed on individuals by altering certain genes of theirs depending on a probability called mutation rate. A gene that has a random number less than a mutation rate will be mutated by replacing its value with a new value.

The outcome of the operators is an offspring population. At this stage, two types of population exist: the parent population and the offspring population. Both populations might contain individuals with good criteria and exclusion on either side may prohibit the best individuals from proceeding to the next generation. The elitism operator is responsible for retaining the best individuals from both populations. During this process, the parent population and the offspring population are merged together to create a transition called the combined population. Comparison is made between all individuals in this population to filter only the best for the next parent population. All the individuals in the lowest non-domination level to the best n-level are selected and grouped into a new population until its maximum size is reached. If the last best level has more potential individuals, then the crowding distance is used as the selection criteria. The new parent population replaces the old parent population and next cycle of evolution continues until a termination condition is reached.



(a) Task allocation (b) Task allocation & data packet encoding

Figure 3: Chromosome structures

5. SCHEDULABILITY ANALYSES

Schedulability of tasks and traffic flows need be determined when tasks are allocated to cores in hard real-time systems. This requires systematic analysis of schedulability and abundant of RTAs proposed since 1960s can be applied for this purpose. In this paper, The RTAs proposed in [2] and [17] are extended and then integrated into the optimisation algorithm to enable calculation of task end-to-end response time upper bounds.

5.1 Analysing the Schedulability of Tasks and Traffic Flows

In a hard real-time system, each task can be assigned with a fixed priority level. This is to enable pre-emption of low priority tasks by their high priority counterparts to give the latter guaranteed access to the shared computing resource. Interference caused by the pre-emptive actions can delay the response time of tasks especially those with low priority level. The classical schedulability analysis of single processor [2] can be applied to analyse the worst-case response time of tasks. Application of this analysis is based on the assumption that each processor has its own priority-ordered queue due to the partitioning scheduling in multiprocessors [4]. Equation (1) calculates the worst-case response time (r_i) of a task. If for each task i of a task set, $r_i \leq D_i$, then the task set is deemed schedulable on the processor, where r_i and D_i is the response time and deadline of task i respectively. The first term of this equation refers to the computation time (c_i) of task i when no interference occurs. The second term refers to the total interference experienced by task i caused by high priority tasks in the interference set ($hp(i)$).

$$r_i^{n+1} = c_i + \sum_{\forall Task_j \in hp(i)} \left\lceil \frac{r_i^n}{t_j} \right\rceil c_j \tag{1}$$

Similar to tasks, traffic flows on shared NoC links are assigned with fixed priority levels and packets that belong to a traffic flow inherit the same priority. Routers guarantee accesses for high priority packets by pre-emptively delay the low priority packets from using the shared NoC links. This arbitration policy causes not only direct interference but also indirect interference to packets with low priority level. The definition of direct interference is rather straightforward; it is the amount of interference caused by high priority traffic flows that share the same path as the observed traffic flow. Indirect interference is caused by the traffic flows with higher priority level than the packets that cause direct interference, sharing the same path as them but not directly with the observed traffic flow. This concept is generalised in equation (2) to calculate the worst-case latency (R_i) of a traffic flow [17]. For an observed flow i, the first term of equation (2) represents its basic latency (C_i) whilst the second term refers to the total latency caused by the direct and indirect interference. The maximum deviation from release period (J_j^R) of flow j and the maximum interference it receives from direct and indirect high priority flows (J_j^I) are known as release jitter and interference jitter respectively. Given a set of traffic flows, this set is deemed schedulable if for each flow i, $R_i \leq D_i$, where D_i is the deadline of flow i.

$$R_i^{n+1} = C_i + \sum_{\forall Flow_j \in S_i^D} \left\lceil \frac{R_i^n + J_j^R + J_j^I}{T_j} \right\rceil C_j \tag{2}$$

We extended the real-time analyses [2, 17] to enable the calculation of task end-to-end response time upper bound. The end-to-end response time of a task is defined from the moment it is released until its last packet reaches the destination router. A task is schedulable if its upper bound does not exceed its deadline (D), which we assume as equivalent to period (T) of the task i.e. $D = T$. A task set allocated to a core is schedulable if every task in this set can meet its deadline. However, packets sent from a task that communicate with other task located at a different processor may not be necessarily schedulable if they fail to meet the deadline of their sending task (e.g. due to the delay of task execution). Since we assume the release of a traffic flow is only after the task that generates it finishes its execution and overhead to write payload to NoC is negligible, the release jitter of a traffic flow is influenced by only the worst-case response time of task that releases it. Then, it is immediate that the release jitter of flow i is $J_i^R = r_i$. Therefore, from equation (2):

$$R_i^{n+1} = C_i + \sum_{\forall Flow_j \in S_i^D} \left\lceil \frac{R_i^n + r_j + J_j^I}{T_j} \right\rceil C_j \tag{3}$$

A task is schedulable if $r_i \leq D_i$, whereas a traffic flow is schedulable only if $J_i^R + R_i \leq D_i$ or after substitution $r_i + R_i \leq D_i$.

5.2 First Objective

Unschedulable task set can cause unpredictable behaviour in the systems due to deadline miss. Therefore, the first objective of the optimisation algorithm is to minimise the number of deadline miss. From equation (1) and (3), worst-case response time of a task and worst-case latency of a flow can be calculated respectively. The following comparisons determine if a task set and their respective packet flows are schedulable in the system.

$$Task_i : if\ r_i > D_i \Rightarrow Ut_i = 1$$
$$Flow_i : if\ r_i + R_i > D_i \Rightarrow Uf_i = 1$$

Given that Ut_i and Uf_i is the count for unschedulable task and flow respectively, then the metric representing the schedulability of the system is the total number of unschedulable tasks and flows. As shown in equation (4), minimisa-

tion of this metric is the first objective for the optimisation.

$$Obj_1 = \min(\sum_{i=1}^{k} Ut_i + \sum_{i=1}^{l} Uf_i) \quad (4)$$

6. NOC ENERGY MACROMODEL

Because of the large variety of NoC architectures, it is not possible to construct a universal energy macromodel to describe the energy consumption in NoCs. Our focus in this work is to develop a simple yet expressive energy macromodel for NoCs capturing the main effects observed in the energy consumption of NoCs in typical scenarios. This macromodel will be used to analyse the effect of task allocation and encoding in different (broad) classes of NoC architectures. The macromodel uses physical parameters to abstract the hardware architecture and application parameters to model the use of the NoC.

We restrict our attention to energy optimised NoC designs, where current state-of-the art low-power techniques are applied, as for example, clock-gating, power-gating, and crosstalk minimisation techniques. Under this restriction, it can be assumed that the energy used by the NoC can be closely approximated by the sum of the energy that is required by each of the transmitted packets. A packet is composed by header and data flits. Since the processing and statistical characteristics of those two kinds of flits are completely different, it is necessary to describe their contributions separately. In the sequel, we use the sub index H and D for header and data respectively. There are three main elements involved in the energy consumption of the NoC: the network interface (N), the router (R), and the links (L). Let us denote as E_{NH}, E_{RH}, and E_{LH} their respective energy consumption when transmitting the header flits. Similarly, E_{ND}, E_{RD}, and E_{LD} are the energy for a data flit corresponding to the network interface, router, and data link respectively. We assume a mesh topology. By a simple topological analysis, it is clear that a packet requiring h hops to move from the source to the destination will require that each flit traverse 2 network interfaces, $(h+1)$ routers and h links. Using an approach similar to [11] but employing energy instead of power, the energy consumption associated with transmitting an n-flits packet is:

$$\begin{aligned} E_{pack} = {} & (2E_{NH} + E_{RH}) + h(E_{LH} + E_{RH}) \\ & + n(2E_{ND} + E_{RD}) + nh(E_{LD} + E_{RD}) \end{aligned} \quad (5)$$

Let us observe that the previous parameters (e.g. E_{LD}) are not constant. They depend on the data being transmitted. The next section analyses this dependency.

6.1 Links

As the aspect ratio of the wires get larger, coupling capacitances dominate ground capacitances and both latency and power get dependent not only on the transition activity (toggling) of one line, but also on the activity on neighbour wires. Although coupling avoidance codes exist [11], it is more efficient to use other strategies such as wire interleaving between sender and receiver, shielding [8] or edge shifting [16]. A consequence of the use of these approaches is that the energy consumption over the link is proportional to the transition activity, tr, as in classical busses. For the energy macromodel, it is advantageous to employ the parameter dt which measures the difference in transition activity with respect to random noise, instead of tr directly. The rationale is that $dt = 0$ for uncorrelated data (default case) while $dt > 0$ when using encoding. Since random noise has a transition activity of $\frac{1}{2}$, dt is given by $dt = \frac{1}{2} - tr$. As it is well known, the energy consumption for sending a data (e.g. a flit) over a link for a bus is:

$$E_{LD} = C_{eq} V_{dd}^2 tr = E_{LO}(1 - 2dt)$$

With E_{LO} being the energy consumption required when sending uncorrelated random data. Since the header is uncorrelated with the data flits, we have $E_{LH} = E_{LO}$.

6.2 Router

Inside the router, the main contributors to the energy consumption are the buffers, the crossbar switches and the clock. Some typical power breakdowns [13] are shown in Table 1. The crossbar is combinatorial, and its dynamic energy consumption is almost proportional to the transition activity tr of the signal being transmitted. The energy in the buffers (typically implemented with flip-flops or SRAM cells) is also dependent of the transition activity; the dependency is much smaller because of the overhead of the clock. Let us denote by α_{RD} the actual dependency of the energy consumption in the router with the transition activity and as E_{RO} the energy when sensing random data. Thus:

$$E_{RD} = E_{RO}(1 - \alpha_{RD} dt)$$

For the header, there is no variation with the transition activity; however the energy value is typically greater than E_{RO} since the processing of the header requires the use of additional functionality and digital gates. Defining $k_H = \frac{E_{RH}}{E_{RO}}$ we can express E_{RH} as:

$$E_{RH} = k_H E_{RO}$$

The advantage is that k_H is more independent of the technology than E_{RH}.

6.3 Network Interface

In principle, we could repeat the previous analysis for the network interface. However, in this case, the energy required to process a flit is rather independent of the transition activity. The rationale is the larger energy overhead of the network interface. For the sake of simplicity, we assume $E_{NH} = E_{ND} = E_{NO}$ where E_{NO} is constant.

6.4 Complete Network Without Encoding

Using (5) and the previous formulas, it is straightforward that:

$$\begin{aligned} E_{pack} = {} & nh[E_{LO} + E_{RO} - dt(2E_{LO} + \alpha_{RD} E_{RO})] \\ & + h[E_{LO} + k_H E_{RO}] \\ & + n[2E_{NO} + E_{RO} - dt(\alpha_{RD} E_{RO})] \\ & + [2E_{NO} + k_H E_{RO}] \end{aligned} \quad (6)$$

For architectural comparisons, it is useful to normalise the energies in order to provide a technology independent comparison. We use the link energy, E_{LO}, as the normalisation factor. Defining the coefficients as $\beta_R = \frac{E_{RO}}{E_{LO}}$ and

Table 1: Power Breakdowns in NoC

	Buffers	Crossbar	Links
RAW	31%	30%	39%
TRIPS	35%	33%	31%
TeraFLOPS	22%	15%	17%

$\beta_N = \frac{E_{NO}}{E_{LO}}$. It is immediate that:

$$\begin{aligned} e_{pack} &= \frac{E_{pack}}{E_{LO}} \\ &= nh[1 + \beta_R - (2 + \alpha_{RD}\beta_R)dt] + h[1 + k_H\beta_R] \\ &\quad + n[2\beta_N + \beta_R - \alpha_{RD}\beta_R dt] + [2\beta_N + k_H\beta_R] \end{aligned} \tag{7}$$

This formula shows that (to a first approximation) the normalised energy consumption for sending a packet over the NoC is a polynomial function of the message size (n), the hop count (h), and the signal activity characteristics of the data (dt). These are the application parameters of our macromodel. Moreover, the coefficients of the polynomial are a function of some physical parameters. The coefficients β_R and β_N characterise the relative energy consumption of the router and network interface versus the link, while k_H measures the overhead for processing the header and α_{RD} the decrease in the router energy when the data flits are correlated.

6.5 Complete Network With Encoding

In the case of a NoC using the low-power encoding techniques, an encoder and decoder are introduced in the network interface to decrease the transition activity (dt is thus increased). Let us denote E_{encod} and E_{dec} the energy consumption of the encoder and decoder respectively, and the reduction in activity because of the encoding, Δt is:

$$\Delta t = dt_{encod} - dt_{unencod} = t_{unencod} - t_{encod}$$

In principle, we could redo the analysis for the complete NoC again, but adding now the energies (per data flit) for the encoder E_{encod} and decoder E_{dec} and using dt_{encod} instead of dt. Clearly:

$$\begin{aligned} E_{pack\_encod} &= nh[E_{LO} + E_{RO} - dt_{encod}(2E_{LO} + \alpha_{RD}E_{RO})] \\ &\quad + h[E_{LO} + k_H E_{RO}] + n[E_{encod} + E_{dec} \\ &\quad + 2E_{NO} + E_{RO} - dt_{encod}(\alpha_{RD}E_{RO})] \\ &\quad + [2E_{NO} + k_H E_{RO}] \end{aligned} \tag{8}$$

After some algebraic manipulations:

$$\begin{aligned} E_{pack\_encod} &= E_{pack\_unencod} + n(E_{dec} + E_{encod}) \\ &\quad - n[2hE_{LO} + (h+1)\alpha_{RD}E_{RO}]\Delta t \end{aligned} \tag{9}$$

Again, it is convenient to normalise the energy with E_{LO}. Let us define the relative encoding overhead factor, β_{encod}, as:

$$\beta_{encod} = \frac{(E_{encod} + E_{dec})}{E_{LO}}$$

and the overall factor for the effect of the transition, α_{encod}, as:

$$\alpha_{encod} = 2h + (h+1)\alpha_{RD}\beta_R$$

Then:

$$e_{pack\_encod} = e_{pack\_unencod} - n(\alpha_{encod}\Delta t - \beta_{encod}) \tag{10}$$

This formula shows that a low-power encoding strategy will achieve a decrease in the energy when the decrease of the transition activity (Δt) rated with the effect that transition activity has in the particular NoC (α_{encod}) outperforms the energy overhead of the encoder and decoder (β_{encod}). Moreover, the effect of the transition activity increases with the hop count (h). In the worst-case (i.e. the energy in the router does not depend on the signal activity, $\alpha_{RD} = 0$, and the energy gains are only due to the reduction of the link energy), the α_{encod} factor is $\alpha_{encod} = 2h$. Thus, low-power encoding is beneficial if the hop count, h, is larger than:

$$h \geq \frac{\frac{1}{2}\beta_{encod}}{\Delta t} \tag{11}$$

Hereafter, it is referred to as low-energy encoding condition.

6.6 Second Objective

The energy macromodels (7) and (10) provide estimation of normalised energy dissipated by NoC components for traversing a packet in NoC implemented without or with encoding technique respectively. These macromodels can be applied to analyse the effects of changing the allocation of tasks and configuring data packets with encoding technique in terms of energy dissipation in typical NoC architectures. We define the second metric as the total energy dissipation (normalised) for traversing all packets in the network, given as:

$$P_m = \sum_{m=1}^{l} e_{pack\_encod}$$

Hence, the second objective is defined as minimisation of this metric.

$$Obj_2 = \min(P_m) \tag{12}$$

7. EXPERIMENTAL WORK

7.1 Case Studies

Two case studies were used in this experimental work. The first case study is based on an autonomous vehicle application (AVA) that consists of 33 tasks, 38 traffic flows and run on a 4x4 NoC-based multicore platform. The second case study is synthetically created to represent a communication-intensive application (SA) with 50 tasks running on a 5x5 multicore platform and transmits 50 traffic flows. Both case studies are described using the application model explained in section 3, allowing real-time analyses to perform schedulability test on each task and traffic flow. The goal of this experimental work is to investigate the impact when changing task allocation and encoding of traffic flows in terms of schedulability and energy dissipation.

7.2 Determining the Energy Macromodel Parameters

The first phase in the experimental work has been to determine some reasonable values for the different parameters involved in the energy macromodel of the NoC in 3 scenarios. As shown in Table 2, scenario 1 corresponds to a NoC where the major energy bottleneck is located in the

Table 2: Parameters of NoC Energy Macromodel

Name	Scenario 1	Scenario 2	Scenario 3
β_R	2	1	1/4
β_N	1	1	1/4
k_H	1.08	1.08	1.08
α_{RD}	0	0	0

router. Scenario 2 corresponds to the typical case of a NoC where energy dissipation is evenly distributed while scenario 3 refers to a case where the interconnects are the major energy contributor. Further on, we have implemented an encoder/decoder (Code1) in an industrial 65nm technology similar to the probability encoder of the PMD approach [8]. It uses a 16b flit and it is memory-less. The energy consumption reported by Synopsys Design Vision is E_{encod}=91.3fJ. An even simpler code approach (Code2) similar to K0 [5] would require one half of that energy. As in [11] we consider a typical value for the link energy in a 65nm bus of 592 fF/mm and a supply voltage of V_{dd}=1.2V. Thus, depending on the length of the link in the NoC a different value of encoding overhead (β_{encod}) is obtained. The typical value of encoding overhead is in between 0.25 to 4.

The reduction of transition activity achieved by the low-power encoder depends on the statistical characteristics of the signals. We analysed a set 12 typical signals ranging from html files, gzip program, images, OFDM signals, and several DSP signals. For the sake of simplicity, all the signals were 8b wide, but two of them were packed in a 16b flit. This arrangement tends to make encoding more difficult because of the lost of correlation. As a result, we observed that the signals can be clustered in three characteristic groups:

- Signals with few correlation (html, gzip). For those signals, neither Code1 nor Code2 achieved a reduction in transition activity.
- Signals with some correlation (OFDM, image, etc). Code1 achieved approximately a 30% in transition activity, while Code2 could only get 15%.
- DSP signals with high temporal correlation. In this case the reduction with Code1 spans between 35% and 40%, while Code2 achieves a reduction between 27% and 30%.

7.3 Optimisation of NoC-based Systems

Based on the case studies, we performed a comparison between the proposed MOGA, the GA with encoding mode determined by the fitness function (GA-ENF) and the GA without encoding technique (GA-UNE). One of the differences between MOGA and the two GAs (GA-ENF and GA-UNE) is their chromosome structure; both using the chromosome structure as shown in Figure 3a. In detail, GA-ENF decides encoding decision of data packet in its fitness function according to the low-energy encoding condition as shown in equation (11), which allows it to gain optimal energy dissipation for a given task allocation. Contrary to MOGA and GA-ENF, GA-UNE does not have encoding technique implementation, thus no data packets are encoded.

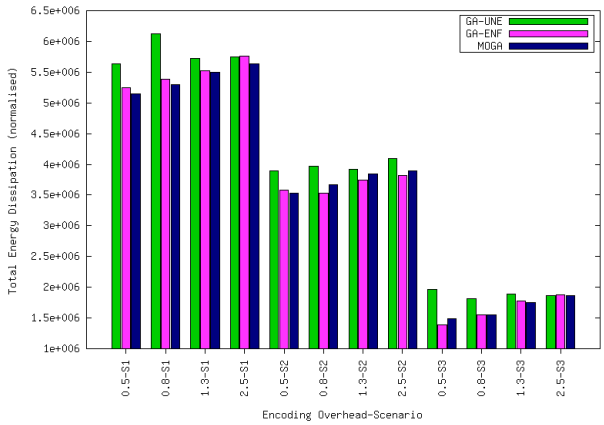
For the MOGA, with an addition of encoding mode in its chromosome structure, the design space that it can explore is larger than the other GAs by 2^l i.e. $m^n \times 2^l$ in total size of design space, where m, n, l are the total cores, total tasks and total traffic flows respectively. This gives the MOGA with an ability to determine the encoding mode of traffic flows, but it has to explore larger design space than GA-ENF and GA-UNE. For the GA-ENF, while its design space is smaller than the MOGA, the evaluation model is optimal, whereby it can determine accurately which data packet flow requires encoding, however, it suffers a slight calculation overhead as it has to check the encoding condition first before knowing which of which needs encoding. This is an extra overhead to the existing number of evaluations performed by the GA, which is approximated around 5,555,000 evaluations (= 100 populations $\times$ 500 generations $\times$ (33 tasks + (38 $\times$ 2) flows)) in every run. The GA-UNE does not have both overheads, but the energy dissipation might be higher than the MOGA and GA-ENF due to the unencoded data packets.

The experimental hypothesis is that the MOGA is as good as the last two GAs in producing schedulable allocation of tasks, and in terms of the energy dissipation it is as comparably good as the GA-ENF and always better than the GA-UNE. In addition to this, we also compared the proposed MOGA with several baselines: a nearest neighbour (NN) and a random mapper, as the second part of the hypothesis that describe it is always better in both metrics (schedulability and total energy dissipation) than the baselines. With these comparisons, we aim to show that the proposed MOGA is a better technique to find the schedulable allocation of tasks for encoded NoC based hard real-time embedded systems.

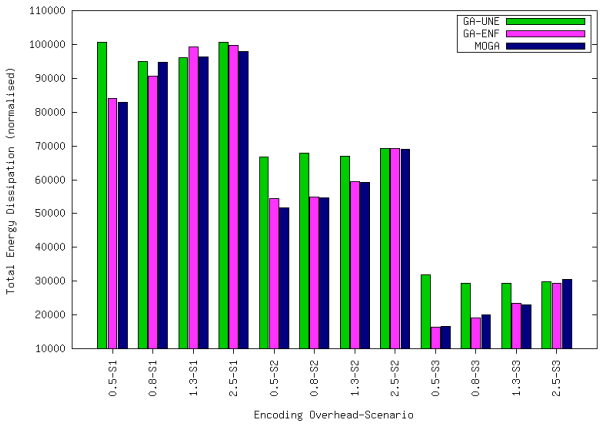
In order to provide an insight into how much impact does the task allocation and data packet encoding impose on energy dissipation of NoC, every scenario in Table 2 was simulated with different encoding overheads (0.5, 0.8, 1.3 and 2.5); representing the amount of overhead as the length of wire is gradually decreased. This creates a set of scenario-encoding overhead combinations as shown in Figure 4. With this set, the performance of the MOGA and the other GAs can be examined as in different situations.

Figure 4a and 4b show the level of energy dissipated by the NoC varies at different scenarios for AVA and SA respectively. These graphs plot only the best task allocation found at the last generation, which meet schedulability of the application and with the lowest energy dissipation. Each point in x-axis represents a combination of encoding overhead-scenario, for example, 0.5-S1 is scenario S1 with encoding overhead of 0.5. Note that an unencoded NoC does not have encoding overheads; the best solution is plotted with regard to scenarios in the x-axis (e.g. S1, S2 and S3).

It is clearly depicted in Figure 4a, that the energy dissipation of unencoded NoC (GA-UNE, green bar) is higher in almost all scenarios than encoded NoC produced either by the proposed MOGA (blue bar) or the GA-ENF (pink bar). This is also true in SA case study (Figure 4b) but with an exception in scenario 1.3-S1, which shows the energy dissipation of GA-UNE is lower than the others, probably due to the stochastic characteristic of the GAs. When the encoding overhead becomes 2.5 the length of wires between routers is very close to nil, increasing the energy dissipation of encoded NoC to the same level as the unencoded NoC, as shown at 2.5-S1 and 2.5-S3 of AVA case study, and 2.5-S2 and 2.5-S3 of SA case study. The proposed MOGA shown better energy dissipation than the GA-ENF in 7 and 8 scenarios (out



(a) Autonomous Vehicle Application 4x4



(b) Synthetic Application 5x5

Figure 4: Comparison of energy dissipation across different scenarios

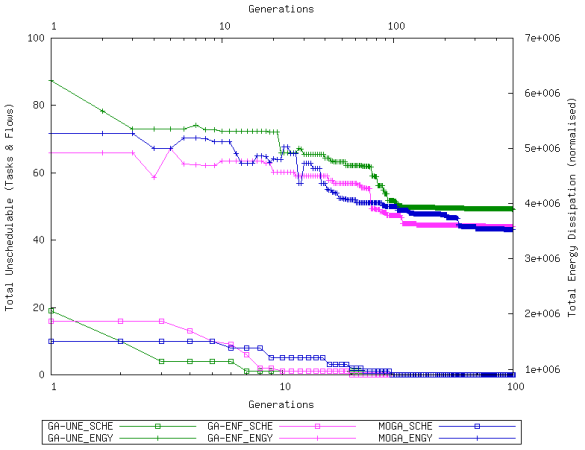
of total 12 scenarios) with AVA and SA respectively.

Both graphs of Figure 5 plot the best task allocation in terms of schedulability and energy dissipation in each generation, showing the convergence of the proposed MOGA (blue), GA-ENF (pink) and GA-UNE (green) through generations (1-500). The lower part of the graphs is where the schedulability is plotted whereas the energy dissipation is shown at the upper part of the graphs. The best task allocation in this graph is defined as the one that was found at every generation with the smallest number of unschedulable tasks and flows, which generates the lowest energy dissipation. In terms of meeting the hard real-time timing constraints, all of them able to find allocation of tasks that meet the requirement below 100 generations in both case studies (AVA and SA); and maintaining it to the last generation while improving the energy dissipation of the NoC. The proposed MOGA seems a little late in finding the schedulable allocation of tasks. One of the possible reasons is that it has a larger design space to explore as compared with the other two, thus it might require more generations to find the schedulable allocation of tasks, while at the same time finding the right encoding combination of the flows that reduces the energy dissipation. However, the proposed MOGA gradually shows better convergence in terms of energy dissipation when the generation approaching 500, as clearly shown in Figure 5a and 5b. This implies that, at least for scenario 0.5-S2, the proposed MOGA was able to find a schedulable task allocation as good as the GA-ENF and GA-UNE (within 100 generations) and then preserved it to the last generation. Then, by having it to decide the encoding mode of the traffic flows provide it with an ability to further reduce the energy dissipation of NoC, lower than the other two GAs.

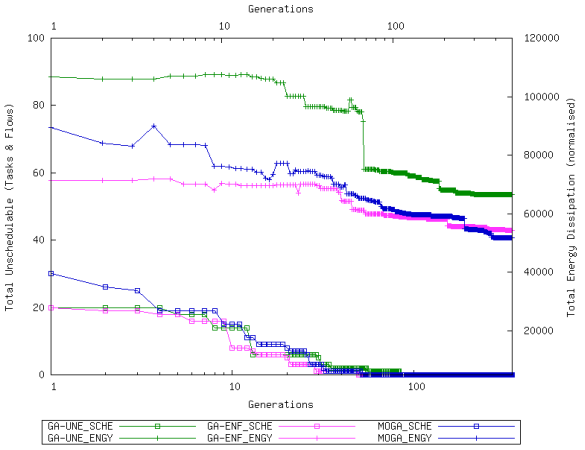
In multi-objective optimisation, having all objectives satisfactorily met like the single-objective is difficult. The solutions found by an optimisation algorithm usually contain trade-off between the objectives. An improvement is said to have been made by new solutions from the previous solutions if they can show an improvement in one objective but not worse in the others towards the Pareto-optimal region [12]. Those solutions are not dominated by the others in the population and are known as the non-dominated set. In order to assess the performance of our proposed MOGA in this regard, we plot the graphs as depicted in Figure 6 to show the improvement made from the random distribution of solutions in the first generation until the last generation (500). It is clearly depicted in graph 6a and 6b that the non-dominated set of MOGA (blue line) has better trade-off than the GA-ENF (pink line) at the last generation in both case studies. The trade-off produced by GA-UNE is no better than MOGA and GA-ENF, as shown in the graphs its non-dominated set is largely dominated by non-dominated set of MOGA, which again validating our hypothesis.

In the above paragraph of this section, we have discussed the analysis presented in Figure 4 that compares the energy dissipation of the schedulable task allocation found by MOGA, GA-ENF and GA-UNE in different combinations of scenario. In most of the scenarios of both case studies, MOGA produced better energy dissipation as compared with the others; while in some other scenarios it performed worse than them. In order to have better insight of their performance we chose the Wilcoxon rank-sum test to statistically determine which has the lowest energy dissipation between the MOGA and GA-ENF. The statistic test is chosen to test the two groups because their population from which they are drawn are not normally distributed; as the result of stochastic characteristic of the GAs. Each group contains 12 samples and the critical value is 115 for two-tailed testing at 0.95 confidence level, which we used to reject the null hypothesis if the obtained critical value is less than that. The null hypothesis of this test states that there is no significant difference in terms of the energy dissipation between the two groups of schedulable task allocation and the alternative hypothesis is either the MOGA has greater or lesser energy dissipation than GA-ENF.

From Table 3, in AVA case study, the obtained critical value is 36 and with p-value of 0.0196 we have strong evidence against the null hypothesis in favour of the alternative hypothesis. The alternative hypothesis is that the energy dissipated by the NoC with task allocation generated by MOGA is lesser than the allocation produced by GA-ENF. However, in SA case study the performance of GA-ENF is

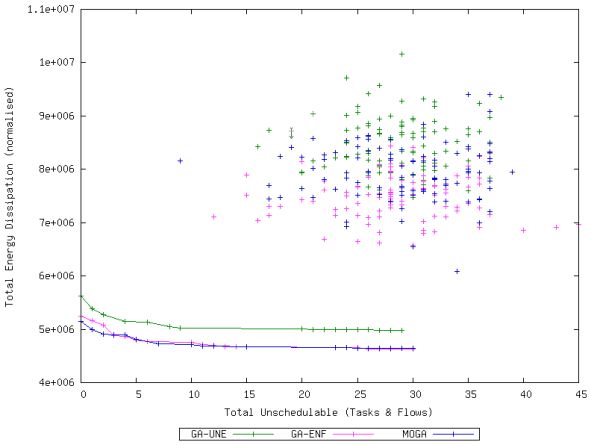


(a) Autonomous Vehicle Application 4x4 (0.5-S2)

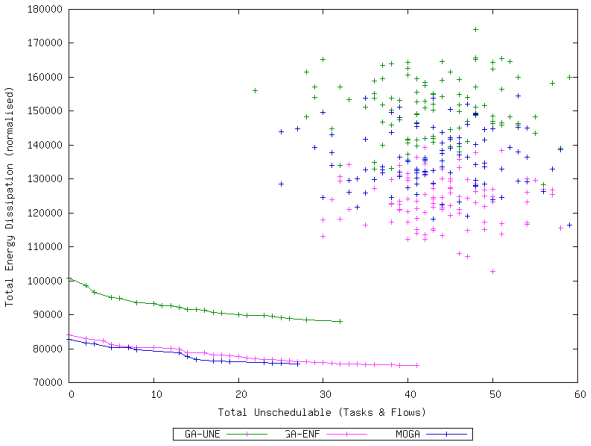


(b) Synthetic Application 5x5 (0.5-S2)

Figure 5: Comparison of convergence of the best task allocation in each generation



(a) Autonomous Vehicle Application 4x4 (0.5-S1)



(b) Synthetic Application 5x5 (0.5-S1)

Figure 6: Comparison of initial solutions and non-dominated sets at last generation

better than MOGA as the null hypothesis is rejected with critical value of 21 supported by strong evidence shown by the p-value of 0.00115. The alternative hypothesis states that the energy dissipation of the NoC with task allocation found by the GA-ENF is lower than the MOGA. Here we can conclude that both, MOGA and GA-ENF are comparably good techniques for finding the allocation of tasks, in spite of their difference in deciding the encoding mode of traffic flows. This again validates our hypothesis mentioned at the beginning of this section that MOGA is as good as the GA-ENF; one may be better for an application than the other or vice versa.

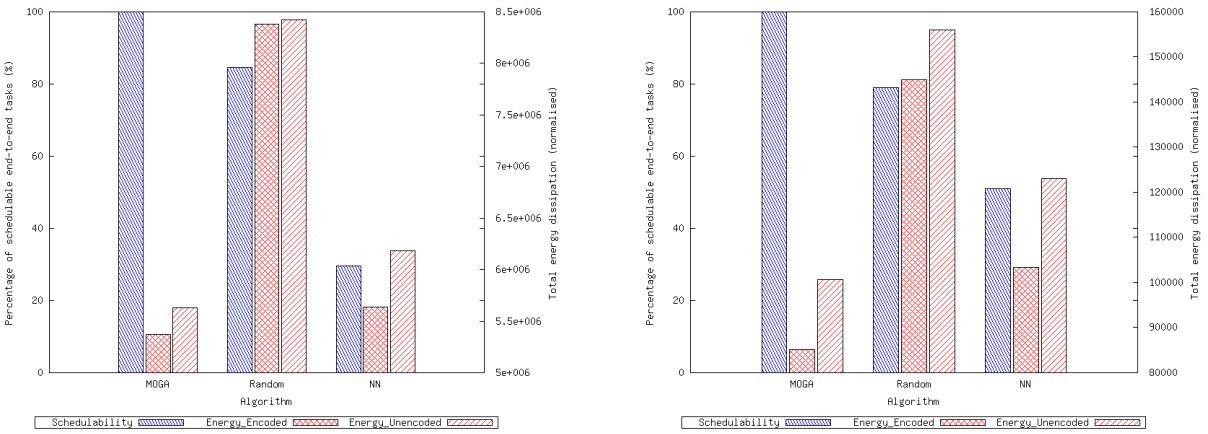
Table 3: Wilcoxon Rank-Sum Test

Application	Critical Value	p-value
AVA	36	0.01936
SA	21	0.00115

Reduction of energy dissipation as a rationale of having encoded NoC in hard real-time embedded systems is clearly shown in graphs of Figure 4, 5 and 6. The baselines depicted in Figure 7 were implemented with the same fitness function as the GA-ENF to calculate the optimal energy dissipation for a given allocation of tasks i.e. labelled as *Energy_Encoded*. Again, MOGA outperformed the baselines in terms of energy dissipation, and even worse the baselines were unable to meet the timing constraints. This validates the second part of the hypothesis that the proposed MOGA is always better in both metrics than the baselines.

8. CONCLUSION AND FUTURE WORK

The main contribution of this paper is an evolutionary multi-objective optimisation algorithm that could find schedulable task allocation whilst minimising the energy dissipation of NoC-based hard real-time embedded systems. The algorithm is efficient for the early design space exploration; able to explore large design space with its fast evaluation technique. This is achieved by integration of analytical fit-



(a) Autonomous Vehicle Application 4x4

(b) Synthetic Application 5x5

Figure 7: Comparison of schedulability and energy dissipation against baselines

ness functions into the algorithm, which consists of new energy macromodel to estimate the energy dissipated by NoC and an extended end-to-end schedulability analysis that can validate schedulability of any task or traffic flow in the systems. We illustrated its feasibility with two case studies and we have shown that the algorithm could find better trade-off between both objectives than the other GAs.

9. ACKNOWLEDGEMENTS

The authors are grateful to the EPSRC (under project 'LowPowNoC', contract EP/J003662/1), Ministry of Education Malaysia (MOE) and Universiti Sains Islam Malaysia (USIM) for providing financial support for this research work.

10. REFERENCES

[1] G. Ascia, V. Catania, and M. Palesi. Multi-objective mapping for mesh-based NoC architectures. In *CODES+ISSS*, 2004.

[2] N. Audsley, A. Burns, M. Richardson, K. Tindell, and A. Wellings. Applying new scheduling theory to static priority pre-emptive scheduling. *Software Engineering Journal*, 8(5):284 –292, 1993.

[3] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002.

[4] J. Garcia and M. Harbour. Optimized priority assignment for tasks and messages in distributed hard real-time systems. In *Third Workshop on Parallel and Distributed Real-Time Systems*, 1995.

[5] A. Garcia-Ortiz, L. Indrusiak, T. Murgan, and M. Glesner. Low-power coding for networks-on-chip with virtual channels. *Journal of Low Power Electronics*, 5(1):77–84, 2009.

[6] R. K. Jena and G. K. Sharma. A multi-objective evolutionary algorithm based optimization model for network-on-chip synthesis. In *ITNG*, 2007.

[7] P. Mesidis and L. Indrusiak. Genetic mapping of hard real-time applications onto NoC-based MPSoCs: A first approach. In *ReCoSoC*, 2011.

[8] T. Murgan, P. B. Bacinschi, S. Pandey, A. G. Ortiz, and M. Glesner. On the necessity of combining coding with spacing and shielding for improving performance and power in very deep sub-micron interconnects. In *PATMOS*, volume 4644 of *Lecture Notes in Computer Science*. Springer, 2007.

[9] N. Nedjah, M. Silva, and L. Mourelle. Customized computer-aided application mapping on NoC infrastructure using multi-objective optimization. *JSA*, 57(1):79 – 94, 2011.

[10] G. Palermo, C. Silvano, and V. Zaccaria. Respir: A response surface-based pareto iterative refinement for application-specific design space exploration. *IEEE TCAD*, 28(12):1816–1829, 2009.

[11] M. Palesi, G. Ascia, F. Fazzino, and V. Catania. Data encoding schemes in networks on chip. *IEEE TCAD*, 30(5):774 –786, 2011.

[12] V. Pareto. *Cours d'economie politique*. Librairie Droz, 1964.

[13] J. Postman, T. Krishna, C. Edmonds, L.-S. Peh, and P. Chiang. Swift: A low-power network-on-chip implementing the token flow control router architecture with swing-reduced interconnects. *IEEE TVLSI*, 21(8):1432–1446, 2013.

[14] A. Racu and L. Indrusiak. Using genetic algorithms to map hard real-time on NoC-based systems. In *ReCoSoC*, 2012.

[15] M. N. S. M. Sayuti and L. S. Indrusiak. Real-time low-power task mapping in networks-on-chip. In *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2013.

[16] J. Seo, D. Sylvester, D. Blaauw, H. Kaul, and R. Krishnamurthy. A robust edge encoding technique for energy-efficient multi-cycle interconnect. In *ISLPED*. IEEE, 2007.

[17] Z. Shi, A. Burns, and L. Indrusiak. Schedulability analysis for real time on-chip communication with wormhole switching. *IJERTCS*, 1(2):1–22, 2010.

[18] W. Zhou, Y. Zhang, and Z. Mao. Pareto based multi-objective mapping ip cores onto NoC architectures. In *APCCAS*, 2006.