TWESD'2012

1st Tunisian Workshop on Embedded Systems Design

CALL FOR Participation

May 25 – 27 2012, Monastir, Tunisia

The Tunisian Workshop on Embedded Systems Design intends to gather and illustrate experiences in tools development and utilization in the area of embedded systems design.

TWESD 2012 will include oral, poster and tutorial sessions given by experts in state-of-the-art topics. The regular technical program will run for three days. In addition, tutorial sessions will be held on the first day of the workshop. Papers are solicited in, but not limited to, the following topics:

Circuits and Systems

1. Analog and RF circuit design techniques
2. Digital signal and data processing
3. Wireless communication systems
4. Nonlinear circuits
5. System on Chip (SoCs)
6. VLSI for Signal and Image Processing
7. Software-hardware architecture co-design

Real Time Systems

1. Computer Aided Design of Smart System
2. Real Time Operating System
3. Adaptive multimedia systems
4. Artificial Intelligence and Application
5. Parallel embedded systems
6. Multiprocessor Real Time Systems
7. Robotics

Software Engineering

1. Embedded systems research and development Process
2. Requirements engineering for embedded systems
3. Virtual integration for systems development
4. Security, reliability and trustworthiness of embedded systems.
5. Formal approaches

Perspective authors are invited to submit full-length (6 pages) papers, in IEEE format, using the guidelines in the authors' info. Only electronic submissions will be accepted via the online submission system. Accepted papers will be published in the electronic Conference Proceedings (CD ROM).

Important Dates

Full Paper Submission	April 25, 2012
Notification of Acceptance	May 5, 2012
Camera Ready Submission	May 10, 2012
For Further details, contact:	M. Abid : mohamed.abid@enis.rnu.tn M. Jallouli : Mohamed.jallouli@enis.rnu.tn

Organizing Committee

General co-Chairs
Mohamed Jallouli, CES, Sfax, Tunisia
Kais Loukil, CES, Sfax, Tunisia

Honorary Chair
Mohamed Abid, CES, Sfax, Tunisia

Tech. Prog. Chair
Hafedh Trabelsi, Tunisia

Publication Chair
Mouna Baklouti, Tunisia

Web co-Chairs
Yessine Hadj Kacem, Tunisia
Mouna Ben Saïd, Tunisia

Finance Chair
Tarek Ouni, Tunisia

Local arrangement co-Chairs
Faten Ben Arfia, Tunisia
Walid Hassairi, Tunisia

Registration
Fees:
- 150DT (Academic)
- 250DT (Industrial)

*le chèque ou le virement devra être au profit de l'association Tunisienne des Techniques Numériques et de l'Automatique ; Société Tunisienne de Banque Agence: Sfax Hached, RIB : **10 700 039 2028 301788 82***

A Model-driven based Methodology for the Scheduling of Dynamically Reconfigurable Architecture

Ismail Ktata[1'2], Fakhreddine Ghaffari[1], Bertrand Granado[1] and Mohamed Abid[2]

[1] ETIS Laboratory, CNRS UMR8051, University of Cergy-Pontoise, ENSEA, France
[2]Computer & Embedded Systems Laboratory (CES), University of Sfax, ENIS, 3038 Sfax, Tunisia

Abstract— **Dynamically reconfigurable architecture, like FPGA, is a solution for implementation of complex dynamic applications, especially for these oriented data flow. One of key points consists on scheduling dynamic data flow graph. However, the problem of scheduling with uncertainty is more and more important in several modern applications. This paper presents a new approach that deal with an original task graph model and with dynamic information gathered during run-time, while meeting graph-dependencies and application constraints. In addition, it aims to better manage the reconfiguration process and minimize its latency. The experimental results prove improvements and efficiency of using our proposed technique.**

I. Introduction

AFTER the success story of FPGAs (Field Programmable Gate Arrays), dynamically reconfigurable architectures (DRAs) become popular in today's embedded systems design. To realize the reconfiguration process, hardware tasks (HW) are managed (placed and removed) dynamically at run-time. In such a context, it is necessary to realize a scheduling of the tasks. In order to make the scheduling, we need a description of the application that exhibits dependencies between the tasks and their own characteristics. The description model is usually used as an input for the scheduler who is responsible to make an ordered association between tasks and DRA resources. In case of a dynamic application where the behavior is uncertain, this association is limited to several constraints (related to application and/or architecture) and depends on a significant number of uncertain parameters. Nevertheless, previous studies [1][2] do not take into account uncertainty in the phase of implementation of the scheduler. In [3], authors studied the limits of actual models to deal with flexible applications showing relevant characteristics, and proposed a new modeling method targeted to dynamic applications implemented on DRAs. In fact, this model, as presented in figure 1, exhibits the dynamic characteristics of an application. It considers three cases of non-deterministic features in applications (number of tasks, tasks execution time, number of needed resources for tasks execution).

In this paper, we present a scheduling approach implementation on a DRA named OLLAF. It is performed to deal with such original task graph and with dynamic information gathered during run-time, while meeting graph-dependencies and application constraints. In addition, it aims to better manage the reconfiguration process and minimize its latency.

II. Related Works

Scheduling problems have been treated in many fields of engineering, operations research and computer science. In particular, in software context, multiprocessor scheduling has

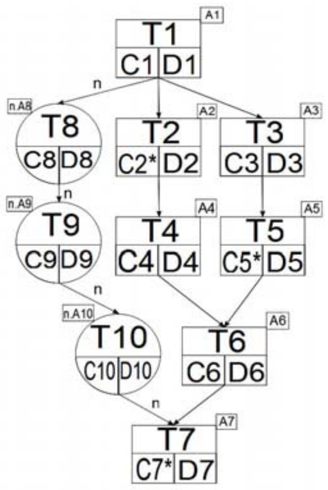

Fig. 1. New graph model of non-deterministic applications.

matured over the last years with many scheduling strategies [4]. Similarly, in HW context, task graph resolution consists in scheduling HW tasks on homogeneous reconfigurable cells partitioned on the hardware device. This problem of scheduling HW/SW tasks is known to be NP-hard. Thus, the main research efforts in this area focus on heuristic methods and few of them propose analytic resolutions.

For previous studies, the major common drawback of proposed techniques is that they do not address real-time constraints. For non-deterministic behavior, they mostly consider one task's feature (e.g. execution time). Moreover, they assume that dynamicity could be seen as different versions (graphs or scenarios) generated at design time, and run-time scheduler has to select the most suitable one further to particular events. However, such assumption remind to static problem where application behavior is either known or could be predicted offline.

In Contrary to all these works, in our proposed approach, three types of dynamicity are considered (number of tasks, execution time, and resource number). We target non-critical real time systems. For such applications, a small percentage of constraints violation (e.g. missed frame deadlines) is usually acceptable. For these reasons, the deterministic techniques are not enough generic to analyze such systems. These techniques are used to indicate whether constraints are violated or not, but not the frequency of violation. Therefore, we could aim for a prediction method introducing the risk that a frame may get miss-predicted, so we may miss the deadline.

III. Methodology Of Scheduling On Target Architecture

In our work, we target OLLAF architecture (figure 2) [5] as a Fine Grained DRA (FGDRA) which has proved its advantage to well deal with soft real-time dynamic applications. It is designed to minimize the reconfiguration overhead and support efficiency

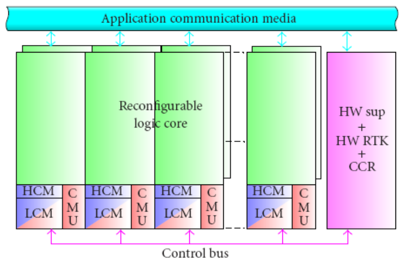

Fig. 2. Global view of OLLAF architecture

operating system (OS) which would abstract design complexity and provide dedicated services to be able to response rapidly to events. The reconfigurable logic core of OLLAF is organized in columns and uses a double memory plan (active plan for execution and the hidden one for configuration). A hardware configuration manager (HCM) and a local cache memory (LCM) are associated to each column [5]. Based on the model of the dynamic behavior of the application, we propose here a technique which combines design-time analysis and optimizations with information collected at run-time about the environment in which the system is operating, and the inputs being received.

1) At Design-Time, graph analysis is performed. It aims to distinguish applications features. Each tasks is defined by its characteristics which are [C, D, N, A]. For variable characteristics (C and N), they are expressed by their maximum estimated value. Then, tasks are classified to permanent or hazardous. Permanent tasks are always executing (they are periodic, repetitive). Hazardous task's execution is not sure in each iteration. From the set of permanent tasks, another classification tries to identify critical tasks which belong to the longest path (in terms of execution time) from the beginning to the end of the execution of the whole graph with an As-Soon-As-Possible (ASAP) schedule.

2) At Run-Time, online scheduling is performed. Dynamic features of the application (hazardous tasks, variable tasks characteristics) may not be known before the lifetime execution occurs, so they may have to be estimated before each iteration based on the past lifetime data measurements. Prediction is not a trivial task: a global tradeoff between gain and cost (run-time prediction overhead) is present. Therefore, heuristic techniques are used. Prediction techniques [6] used are: (a) the mean value of the three last real execution times; (b) the mean of three last values multiplied by different weights; (c) the maximum of the last three real values; (d) the last value increased by 5%. To schedule HW tasks both configuration and context have to be loaded into corresponding column(s). Prefetching is important in that it saves configuration time. But this needs to choose the most probably column(s) where task might be mapped. Based on the static prefetching and precedence relations, every task whose all predecessors are already started their execution is able to prefetch its configuration and context. Priority is assigned firstly to tasks having least laxity (LLF), if there is equality, then to those with most important execution time, if equality again, then to those requiring more number of resources. For each variable feature, and in each execution of its related task, real values are calculated and saved. Predicting the actual run-time environment, including the input data, is obviously limited. Therefore, an adjustment is needed at run-time to minimize error accumulation from a prediction to another. From the online data measurements, information about actual values of the parameters and the quality of the resulting system are collected. Besides, a tuning mechanism is introduced in the application. This is used to adjust the values of the parameters used for prediction.

IV. Experimental Results

To illustrate the proposed strategy for scheduling of dynamic applications on OLLAF with precedence and deadline constraints, we use random generated graph as well as some soft real-life applications. One example is the 3D image rendering application [7]. Considering the 3D rendering of a scene where several objects can be decoded independently inside a frame, and that the number of objects as well as their features can vary from one frame to another. Therefore, the reconfigurable architecture needs to adapt itself quickly to these run-time variations. For instance, if some new objects must be decoded, new tasks can be instantiated and loaded on OLLAF.

We consider a scene where two types of objects (cube and cylinder) appear then disappear in successive twelve frames. For the case of our scheduling implementation, there are five under-estimated values of execution time and one wrong task's instantiation. Updating those wrong contexts needs access to the central memory with a transfer time overhead. The total time cost is about 32,28µs. The execution time prediction offers a reduction rate of 14% of the total execution time. Otherwise, and without prediction, scheduler needs to update variable tasks contexts each time they change. It makes in total 161,4µs which is five times more than the case using prediction. Moreover, in that case without prediction, there is no makespan reduction as tasks are implemented with their worst case execution time.

V. Conclusion

This paper deals with flexible applications presenting non-deterministic features. It takes advantage from both flexibility and high performance which offer new DRAs. We presented a scheduling approach which receives an enhanced application graph as inputs, and implement it in OLLAF taking into account the dynamic features explicitly defined in the graph. Experiments show that proposed approach has reduced up to 18% of the whole application execution time. In addition, prediction heuristic, which allows to prefetch tasks for pre-configuration or to reuse them, permits to reduce the reconfiguration overhead by 80%.

References

[1] Ouelhadj, D. & Petrovic, S. (2009), 'A survey of dynamic scheduling in manufacturing systems', *Journal of Scheduling* **12**, 417--431.

[2] Cowling, P. & Johansson, M. (2002), 'Using real time information for effective dynamic scheduling', *European Journal of Operational Research*, 230-244.

[3] Ktata, I.; Ghaffari, F.; Granado, B. & Abid, M. (2010), 'Novel Approach for Modeling Very Dynamic and Flexible Real Time Applications", Proceedings of the 5th International Workshop on Reconfigurable Communication-centric Systems-on-Chip'.

[4] Levner, E. (2007), *Multiprocessor scheduling: theory and applications*, I-Tech Education and Publishing.

[5] Garcia, S. & Granado, B. (2009), 'OLLAF: a fine grained dynamically reconfigurable architecture for OS support', *EURASIP J. Embedded Syst.* **2009**, 10:2--10:2.

[6] Ktata, I.; Ghaffari, F.; Granado, B. & Abid, M. (2011), 'Dynamic application model for scheduling with uncertainty on reconfigurable architectures', *Int. J. Reconfig. Comput.* Volume **2011**.

[7] Loukil, K.; Ben Amor, N. & Abid, M. (2011), 'HW/SW Partitioning Approach on Reconfigurable Multimedia System on Chip', *Int. J. of Eng.*, 568-578.