Demonstrating a Dataflow-based RTOS for Heterogeneous MPSoC by means of a Stereo Matching Application

Julien Heulot, Judicaël Menant, Maxime Pelcat, Jean-François Nezan, Luce Morin, Muriel Pressigout, Slaheddine Aridhi

Demonstrating a Dataflow-based RTOS for Heterogeneous MPSoC by means of a Stereo Matching Application

Julien Heulot*, Judicaël Menant*, Maxime Pelcat*, Jean-François Nezan*,
Luce Morin*, Muriel Pressigout*, Slaheddine Aridhi**

*IETR, INSA Rennes, CNRS UMR 6164, UEB
20 Avenue des Buttes de Coësmes,
Rennes, France
email: {jheulot,jmenant,mpelcat,jnezan,lmorin,mpressig}@insa-rennes.fr

**Texas Instruments France
5 Chemin Des Presses, 4 Allée Technopolis,
Cagnes-Sur-Mer, France
email: saridhi@ti.com

***Abstract*—This demonstration paper presents a multicore Real Time Operating System (RTOS) that schedules a parameterized dataflow Model of Computation (MoC) onto a multicore Digital Signal Processor (DSP) at runtime. This RTOS called Synchronous Parameterized and Interfaced Dataflow Embedded Runtime (SPIDER) exploits the Parameterized and Interfaced Synchronous Dataflow (PiSDF) MoC and its features at runtime to identify locally static regions and to optimize their execution onto multicore platforms. The RTOS is used to dispatch a stereo matching algorithm tasks with a varying range of disparities. The platform used for this demonstration is a Texas Instruments Keystone II Multiprocessor System-on-Chip (MPSoC) device composed of 8 DSP cores, 4 ARM cores, a shared memory subsystem, Multicore Navigator and multiple dedicated accelerators.**

I. Introduction

Limitations in the processing power of each individual Processing Element (PE) caused by size and power consumption considerations is currently leading to the integration of more and more PEs into MPSoC devices such as Texas Instruments' Keystones processors[1]. Concurrently, signal processing applications are becoming increasingly dynamic in terms of hardware resource requirements. This trend is due to the growing complexity of algorithms to meet the target performance required for complex applications.

SPIDER runtime presented in this demonstration is an open source project[2]. One of the primary challenges in the design and implementation of multicore signal processing systems is to dispatch computational jobs efficiently onto the available PEs while taking into account dynamic changes in both application functionality and resource requirements. The SPIDER runtime optimizes the *multicore scheduling* composed of the assignation, the ordering and the timing of the actors on PEs.

To handle these challenges, applications managed by the SPIDER runtime are described using the PiSDF dataflow MoC [1]. A dataflow MoC decomposes algorithms into pieces of computation called *actors*. These actors exchange data called *data tokens* through *First In, First Out data queues (FIFOs)*. In embedded systems, one of the most commonly used dataflow MoC is Synchronous DataFlow (SDF) [2]. The PiSDF model is obtained by the utilization of Parameterized and Interfaced dataflow Meta-Model (PiMM) over an SDF graph. The PiMM enhances the SDF by integrating two features: a parameter acyclic graph which transmits control values between actors and an interfaced hierarchical scheme to maintain schedulability over hierarchy exploration [1]. The SPIDER runtime exploits these features to efficiently extract parallelism, and reduce the overall latency of a dataflow graph execution.

Stereo matching techniques aim to reconstruct depth maps from a pair of images. Embedded vision is becoming very important in many application fields including intelligent vehicle, cartography, civil engineering, medicine or augmented reality applications. The SPIDER runtime enables the reuse of the stereo matching algorithm in MPSoCs irrespective of the nature and the number of computing elements as the algorithm description is independent of the targeted platform.

This paper is organized as follows: Section II describes the SPIDER components and Section III details the stereo matching use case. Finally, Section IV presents the demonstrator.

II. Synchronous Parameterized and Interfaced Dataflow Embedded Runtime (SPIDER)

A. Dataflow MoC

The SPIDER runtime is based on a Dataflow MoC. The MoC chosen for this RTOS is the PiSDF due to its good properties on parametrization and hierarchy [1]. It will demonstrate how the predictability inherited from SDF allows the SPIDER runtime to take relevant and fast scheduling decisions. Parameters are variables integers that influence data production and consumption of actors in the graph. They can be static, constant for the whole execution, or they can be set dynamically by specific actors called ***configuration actors***.

To extract the maximum parallelism from PiSDF, the methodology presented transforms the PiSDF graph into an single rate Synchronous DataFlow (srSDF) graph at runtime

This work is supported by the ANR COMPA project.

[1]www.ti.com

[2]http://github.com/COMPA-Runtime/COMPA-Runtime

once parameters are resolved. An srSDF graph is an SDF graph where production and consumption rates are equal on each edge. This is achieved by replicating the actors which need to be fired more than once, thus ensuring a single execution of each actor of the srSDF graph during a graph iteration. In this way, the scheduler retains a global knowledge of actors dependencies for the entire graph iteration.

Once this srSDF graph has been generated, the scheduler dispatches actors onto the MPSoC. To do so, the process is separated into: *ordering* and *mapping*. The ordering task consists of sorting all non-executed actors of the srSDF graph into one list. This list is then used to map actors onto each PE of the MPSoC. Both ordering and mapping tasks may be optimized to reduce different metrics such as: latency, throughput, memory utilization or energy efficiency. The current runtime focuses on execution latency.

B. RTOS Topology

The SPIDER runtime targets heterogeneous platforms. A local decision on actor firing may lead to an inefficient global decision as there may be another PE available for this actor which could have permitted earlier completion of the job. In order to ensure efficient global decisions, a Master/Slave execution scheme is thus preferred for heterogeneous platforms.

The SPIDER runtime uses a PiSDF graph as an input algorithm graph defining parameters that influence algorithm execution. Since configuration actors can be executed on any PE of the MPSoC, it is important to send parameters back to the Master PE. The Master PE then makes scheduling decisions based on these parameter values.

The runtime execution scheme is described in figure 1. A *Local RunTime (LRT)* is a low footprint operating system that processes actors. A LRT can be implemented over multiple types of PE: general-purpose processor, DSP, accelerator (unsupported for the moment), and so on. The *Global RunTime (GRT)* is the master of the system. It thus knows the algorithm topology and takes multicore scheduling decisions but can also process actors.

The GRT sends orders to each LRT via a dedicated *job* queue. A *job* embeds all data required to execute one instance of an actor. When a *parameter* is set by a configuration actor, its value is sent to the GRT via a parameter queue. To obtain timing feedbacks on previous and current executions, LRTs sends back actor start and end times through low priority timing FIFOs.

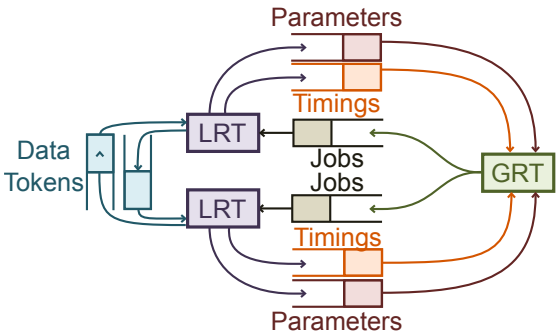

Fig. 1. Runtime execution scheme

III. Stereo Matching Use Case Algorithm

Stereo matching algorithms generate a disparity map from a pair of images by matching pixels from left and right images. This disparity map is representative to a depth map of the scene. For this demonstration, a dense local stereo matching algorithm[3] has been selected. For each pixel, an error is computed for all matching possibilities and the disparity with the lowest error is chosen. This algorithm is deterministic, but results in high computational complexity.

The matching error is computed from the correlation of the two pixels (and their neighborhood). This error is then iteratively refined with a bilateral filter. This refinement step has the highest computing cost. It may be noted that since these errors are computed independently, this stereo matching algorithm can be easily parallelized.

As the errors are computed for each matching possibility (each disparity), the total execution time is linear with respect to the disparity range; hence reducing the disparity range significantly decreases the total execution time. In order to optimize the algorithm and reduce its complexity without sacrificing performance, the disparity range can be adjusted in real time using the histogram of the disparity map.

IV. Demonstrator

The demonstration of the SPIDER runtime will be performed on a Texas Instruments c6636 Keystone II platform. This platform embeds 8 c66x DSP cores and 4 general-purpose ARM Cortex A-15 cores interconnected by a TeraNet Network-on-Chip. To synchronize the cores, the SPIDER runtime uses the Multicore Navigator. This communication system is composed of 16k hardware queues which send data tokens between LRTs and is also used for the bidirectional GRT LRTs communication for jobs, parameters and timings. To exchange data, the SPIDER runtime allocates memory regions in the 6MB shared L2 (MSMC) memory and DDR memory. One of the 20 shared timers is used for global system monitoring.

A USB stereo camera will be plugged on the board to provide a stereo video source. The resulting disparity map will be sent through the Ethernet port to an assisting computer to be displayed. DIP switches or a web interface will be used to change the disparity range at runtime. Then, the SPIDER runtime will automatically dispatch the computation onto all 12 cores; the schedule being dependent on the disparity range chosen at runtime. The disparity range is an example of PiSDF parameters. One of the 4 ARM PEs will run the GRT while the other cores will run LRTs. A Gantt chart of the current execution will also be displayed on the assisting computer.

References

[1] K. Desnos, M. Pelcat, J.-F. Nezan, S. S. Bhattacharyya, and S. Aridhi, "Pimm: Parameterized and interfaced dataflow meta-model for mpsocs runtime reconfiguration," in *Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS XIII), 2013 International Conference on*. IEEE, 2013, pp. 41–48.

[2] E. A. Lee and D. G. Messerschmitt, "Synchronous data flow," *Proceedings of the IEEE*, vol. 75, no. 9, pp. 1235–1245, 1987.

[3] X. Mei, X. Sun, and M. Zhou, "On building an accurate stereo matching system on graphics hardware," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, nov. 2011, pp. 467 –474.