Model-based Analysis and Control of Software-Defined Radio Platforms

Résumé du projet de recherche (Langue 1)

Baseband processing has become an important feature of today’s mobile platforms because the latter have to cope with a high number of air services (e.g., LTE, 3G, WIFI, GPS, . . . ). These services have in common several basic algorithms, therefore advocating for an architecture implementing these basic computations in a generic way, and thus being able to execute all air services in a flexible way. However, this flexible architecture should also be conceived with low power consumption and usability in mind. A few years ago, the laboratory on Systems-on-Chip (LabSoC), an entity of Telecom ParisTech, has introduced - with other academic and industrial partners - , a hardware and software architecture for efficiently executing all air services taking into account flexibility, low cost, and low power consumption constraints [7]. This architecture is based on a set of DSP units, each of them embedding a hardware accelerator, a micro-controller, a DMA controller and some internal memory. The overemball set of DSPs is controlled from a main processor, the internal memory of each DSPs being mapped in the one of the main processor. Data to process and processing results are moved around between memories (DSPs’ local memories and CPU’s main memory) thanks to the DMA engines. Executing concurrent baseband applications within this architecture is non trivial because of the sharing of memory and computation resources between complex and timely constrained baseband processing. Currently, an efficient sharing of these resources must be manually programmed by an expert of the platform, thus limiting the usability of the platform. A solution would be to offer a simplified capture of those applications - including memory and computations need - , and to automatically derive a memory sharing scheme and computation resources schedule. Moreover, this automatic generation should guarantee that all applications respect their timing constraints (e.g., deadlines). The LabSoC has defined a UML profile - called DIPLODOCUS [5] [1] - that targets the design space exploration of complex embedded systems, that is, assist system architects to decide which functions shall be implemented in hardware or in software. DIPLODOCUS follows the Y-methodology [4], with three main stages: (i) application modeling, (ii) architecture modeling and (iii) mapping. The DIPLODOCUS profile is implemented by an open-source toolkit named TTool [6]. The overall approach (DIPLODOCUS, TTool) has been supported by several industrial companies (Texas Instruments, Freescale) and projects (e.g., EVITA). Two Ph.D. thesis have already contributed to the programmming of EMBB-based applications. The first one has investigated the possibility to describe EMBB applications in DIPLODOCUS, and to automatically generate the control code for these applications [3]. Unfortunately, this generated code does not take into account memory sharing nor computation scheduling. The second one is currently proposing a nice way to model complex communication schemes (e.g., tricky memory transfers ) for the design space exploration of EMBB applications [2]. This is a nice step from which code generation might very probably be improved thus enhancing memory placement and data transfers scheduling.

Résumé du projet de recherche (Langue 2)
To achieve the previously described application programming issues, the thesis should focus on the following stages: 1. Contribute to the hardware (design of new DSP units or enhancement of existing ones, design of C++/SystemC simulation models) and software (DSP units' drivers, ...) development of the Embb platform. 2. Learn how to program applications on the EMBB platform. In particular, program two concurrent applications to better understand memory and computation sharing issues. 3. From the first work, study what is really specific to the implemented applications, and what is more generic, i.e. what could have been done more or less automatically. This abstraction work should be done with regards to services offered by the Operating System. The GNURadio abstraction layer might also be considered. 4. A deep bibliographical study must then be done on memory and real-time computation scheduling techniques. This includes work on real-time operating systems, on the programming of complex base-band architectures, and on code generation techniques. 5. Propose a first way to automatically generate code from DIPLODOCUS models for EMBB platforms, starting from our recent work on communication patterns [2]. This code generator shall definitely address memory and computation sharing. Implement that proposition in TTool, and evaluate it for different kinds of baseband processing applications. 6. Propose a way to enhance the previous contribution with a real-time dimension: the generated code shall take advantage of real-time capabilities of the platform (at Operating System level, or at GNURadio level) to generate a code fulfilling real-time constraints of applications. These real-time constraints are really at stake in base-band processing, with usually a time accuracy close to a few microseconds. Implement that second proposition in TTool, and evaluate it. 7. Currently, there is a semantical gap between simulations that can be performed in DIPLODOCUS/TTool (simulations with abstracted data) with regards to more precise simulations currently made for the EMBB platform (simulations with all application data). An intermediate level of simulation would be to use some real data in DIPLODOCUS models, but not all data, so as to better test some race conditions or performance metrics. Again, propositions will be made, and implemented in TTool. This last contribution is optional, i.e. work expectations on memory and computations resources must be first carefully studied before this last contribution is addressed.

Informations complémentaires (Langue 1)

Thèse réalisé dans le cadre d'un projet FUI dans lequel des entreprises internationales sont présentes.