Mots clés :

- **Directeur de thèse**: LUDOVIC APVRILLE
- **Co-encadrant(s)**:
- **Unité de recherche**: Laboratoire Traitement et Communication de l'Information
- **Ecole doctorale**: École Doctorale Informatique, Télécommunications, Electronique de Paris
- **Domaine scientifique principal**: Divers

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., 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. This architecture is based on a set of DSP units, each of them having one hardware accelerator, a microcontroller and an internal memory. The overall set of DSPs is controlled from a main processor, the internal memory of each DSPs being mapped in the one of the main processor. DMA transfers may be used between DSPs’ local memory and the global memory. 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 simplify 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 [3] [1] - that implements the Y-methodology [2], 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 [4]. The overall approach (DIPLODOCUS, TTool) is supported by several industrial companies (Texas Instruments, Freescale) and projects (e.g., EVITA). A Ph.D. thesis has already investigated the possibility to describe EMBB applications in DIPLODOCUS, and to automatically generate the control code for these applications. Unfortunately, this generated code does not take into account memory sharing nor computation scheduling: this is the main contribution that is expected in this Ph.D.

Résumé du projet de recherche (Langue 2)

To achieve the previously described issues, the thesis should focus on the following stages: 1. Learn how to program applications on the EMBB platform. In particular, program two concurrent applications to better understand memory and computation sharing issues. 2. 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. 3. 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 baseband architectures, and on code generation techniques. 4. Propose a first way to automatically generate code from DIPLODOCUS models for EMBB platforms. This code generator shall definitely address memory and computation sharing. Implement that proposition in TTool, and evaluate it for different kinds of baseband processing concurrency. 5. 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, in particular) to generate a code fulfilling real-time constraints of applications. These real-time constraints are really at stake in baseband processing, with usually a time accuracy close to a few microseconds. Implement that second proposition in TTool, and evaluate it. 6. 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 2)

