ICT-EMUCO. AN INNOVATIVE SOLUTION FOR FUTURE SMART PHONES

Maria E. Gonzalez[1], Attila Bilgic[1], Adam Lackorzynski[2] Dacian Tudor[3], Emil Matus [2], Irv Badr[4]

1. Ruhr-University Bochum, 2. Technische Universität Dresden,
3. "Politehnica" University of Timisoara, 4. IBM- Telelogic

ABSTRACT

Mobile communication has become the dominant branch in the communication business over the last decade and is still rapidly growing in the market. With the recent advances in wireless networks and the exponential growth in the usage of multimedia applications, multi-core platforms point to be the solution of feature-rich phones, such as the iPhone or the BlackBerry Storm to deliver the performance comparable to today's computer system.
On the other hand, system scalability and flexibility are vital to enable fast time-to-market and allow manufacturers and service providers to be competitive. Use of virtualization techniques and software development to scalable parallel hardware architectures are inevitable outcome to face the migration to multi-core platforms on mobile devices.

Index Terms— multi-core, load balancer, virtualization, microkernel, mobile embedded systems.

1. INTRODUCTION

Mobile communication has become the dominant branch in the communication business over the last decade and is still rapidly growing in the market. Flexibility, scalability, power consumption and an optimized user-experience appear to be the main requirements of the future mobile devices. System scalability and flexibility are vital to enable fast time-to-market and allow manufacturers and service providers to be competitive. Power consumption has become an issue in feature-rich phones since the new features lead to higher usage of the phones thus decreasing the standby time and generating stronger battery constraints and higher heat dissipation. The mobile user-experience are associated to the functionalities of the applications and the Quality of service (QoS) delivered by the wireless networks, which are impacted by the processing capacity of the mobile devices.
Future mobile devices will incorporate multiple radio access technologies of standards as Universal Mobile Telecommunication System (UMTS) and the 3GPP Long Term Evolution (LTE) in order to provide broad-band mobile-data access and the best quality of service in the current environment of the user. In addition to this, it is accept as a fact that there will be an exponential growth in the usage of multimedia applications such as video streaming, video conferencing, complex graphics etc., raising the computational power demand, which can not be pursued further by accelerating the processor clock. And the coexistence of multiple software environments will be a must in order to offer all demanded services to the user and facilitate application portability.
Today's hardware architecture of user equipments usually contain multiple heterogeneous processing units, but individual units are typically dedicated to specific tasks rather than being general purpose, e.g. specifically for protocol stack handling. Other design approaches allocate special tasks to dedicated systems resulting in unnecessary hardware for situations other than high load.
Under the premises exposed above, migration into multi-core platforms and use of virtualization techniques seems to be an inevitable outcome of the trends of handheld mobile devices.
Multi-core systems for mobile devices are an emerging technology with no (currently) known wide-spread use. This paper gives an overview of a forerunner approach: the eMuCo, Multi-Core Processing for Mobile Communication Systems and how applications can benefit of this.

2. APPLICATIONS OF MOBILE HANDSETS

A mobile handset has basically two functionalities: the modem and application functionality as shown in Figure 1. The modem subsystem is considered as one of the mandatory software component which makes possible

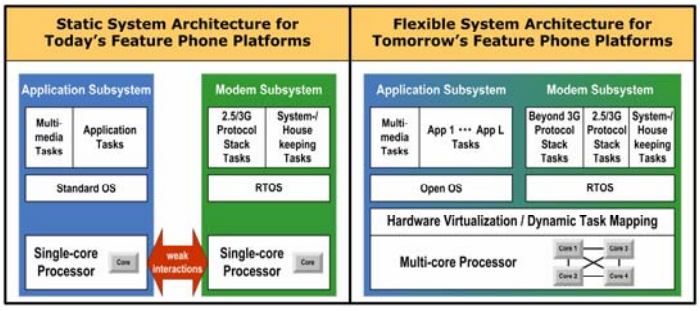

Figure 1. Today's Phone Platforms and ICT-eMuCo's Vision of Tomorrow's Phone Platforms [1]

communication over the air, which contains the so-called protocol stack. The application subsystem tends towards mobile user's applications that look much like PC applications. Protocol stack and user's applications have inherently different requirements on the computational system they use to fulfill their tasks. Whereas the applications of the modem subsystem are determined by hard real-time operation, the applications of the application subsystem require high flexibility and extensibility. Today's approach either combines the processing of applications coming from both subsystems on one microcontroller core or separates them physically on two or more cores with weak interactions (see Figure 1, left picture). This architecture is very inflexible and provides little scalability. The contradiction of exponentially increasing computational performance requirements and low power consumption in combination with high flexibility can be solved by a multi-core approach, homogeneous or heterogeneous with appropriate software stack.

3. EMUCO SOLUTION

Adding software to a mobile phone to permit to exploit the enormous performance gain provided by multi-cores as can be seen in today's PCs creates several challenges, which are met with the eMuCo software architecture approach (Figure 2). It proposes a multi-core hardware platform, which is effectively exploited by the combination of an L4 micro-kernel, a load balancer, and virtualization techniques. [1]

The eMuCo solution offers the co-existence of several protocol stacks into the modem subsystem together with a pluggable Rich-OS based applications subsystem. The software architecture is composed by three main layers:

1.- The microkernel, which is a minimal computer operating system kernel, which only provides a minimal set of kernel functionality and allows building user-level services by providing mechanisms such as address space management, thread handling, and inter-process communication (IPC). Furthermore, the microkernel offer good isolation characteristic as it allows separating sub-systems from each other. Therefore, it is supposed to be a good basis for secure systems since it allows isolating potentially untrusted components while maintaining security properties for others.

2.- The resource layer contains all the software needed to run the applications. It is basically divided in two parts:

The basic resource management contains the needed software to manage the physical resources such as low level drivers, memory allocation, load balancing, etc.

The load balancer, which is one of the core components in the resource layer to support multi-core, provides the services such as allocation of task/thread on the multiple cores, priority management and thread monitoring.

The adaptability sublayer offers services to the applications such as high level driver services.

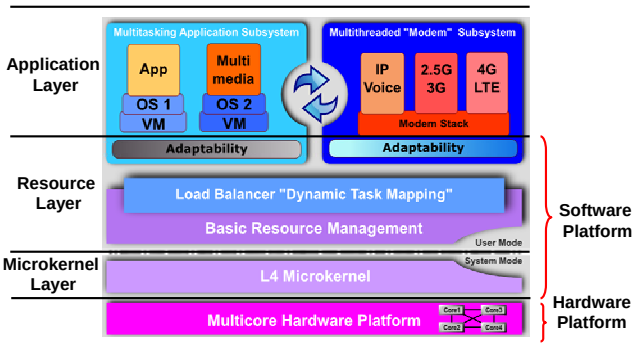

Figure 2. eMuCo software architecture [1]

3.- The application layer shown in the Figure 2 is the consumer of the services.

In the application sub-system, para-virtualization allows running a broad set of existing applications in new environments by using virtual machines to run whole operating systems along with their applications.

In the modem sub-system, the protocol stack does not have virtual cores, it is just a "normal" multi-threaded application, which will be scheduled among cores by the load balancer and on the single cores by the kernel.

Additionally the applications, both in the application sub-system and the modem sub-system, should be multi-threaded to take advantage of the multi-core hardware platform.

4. USE CASES

Since application and the modem subsystem have to coexist and share the resources in the eMuCo solution, two use cases have been chosen. As reference multimedia application the H264 video codec and as reference protocol stack the data path component of LTE were chosen.

The eMuCo project will use the video decoder as an example multi-core application that is using multiple cores in the system to distribute work of a single application, either to increase the video quality or avoid the use for special hardware acceleration. Internally the video decoder is using different threads where it decodes the slices in parallel. It is expected that the resource layer schedules the threads accordingly to multiple cores.

In the context of the eMuCo project, the data path of the protocol stack is the most interesting part from the software architecture point of view since in a well dimensioned cellular network, the granted radio capacity for data transmission through the air interface should be enough to satisfy the application requirements and full fill the user expectations. However, if the average processing time in the user's equipment is less than the time required to capture the data, the user experiments undesired system behaviour and the data waiting for processing backlog in the data pipes and in the main buffer memory leading eventually to an overflow condition.

4.1. H.264 video code use case

H.264 / Advanced Video Coding (AVC) is a popular industry standard for video compression. H.264 is the next-generation video compression technology in the MPEG-4 standard also known as MPEG-4 Part 10. H.264/AVC features high compression ratio, which is very well suited for video streaming in mobile systems. However, it also requires significant effort for decoding. Fortunately a multi-core system can be beneficial as the H.264/AVC standard includes a concept called 'slices' where a single video frame is divided into several independent parts or slices (See Figure 3). Using multiple slices slightly increases the size of the compressed video file. However, slices can be decoded independently by one processing unit. Consequently, using multi-core systems, decoding work can be distributed to multiple cores, which can e.g. increase the video quality or avoid the use for special hardware acceleration.
Internally the video decoder is using different threads where it decodes the slices in parallel. It is expected that the operating system schedules the threads accordingly to multiple cores. Besides an environment supplying threads the video decoder needs a frame buffer to display the video on which is one of the preconditions for the demonstrator.
Some requirements for the MPEG4 application design can be stated as follows:

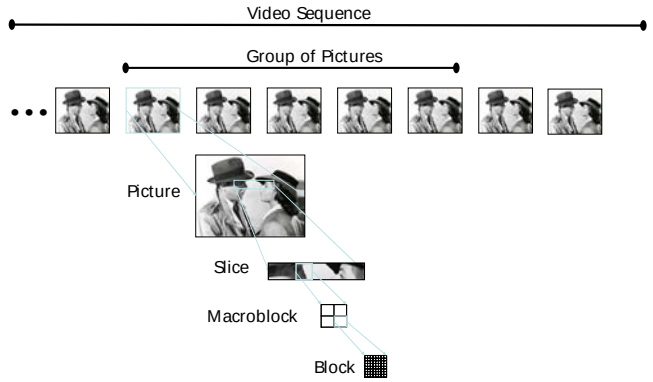

Figure 3. Video Stream Example [2]

- The application should be implemented to achieve as high a degree of parallelism as possible e.g. Each slice could be a thread, which can be assigned to different cores depending on the load.
- Use prediction of the decoding time to get the size (number of macro-blocks) of the slices in the coded could be a plus to have comparable processing time per thread.
- Since the slices will be decoded in parallel instead serially, decoder needs a frame buffer to display the video.
- The software layer should provide enough processing power to respect the quality service constrains of such application.

As seen, the most of the requirements mentioned above are concerning to the application itself to take advantage of the multicore platform and not about the eMuCo environment since it is expected a full portability in the application sub-system, in the sense that the application software should run without any modifications in the multi-core software execution environment (that is application running on Linux should also run on the para-virtualized L4Linux).

4.2. LTE protocol stack Use Case

Currently upcoming and future mobile devices for systems like the Long Term Evolution (LTE) incorporate multiple wireless connectivity standards to enable the best quality of services for the user.
The most important components of the data path are the MAC layer, RLC layer and PDCP layer which are depicted in Figure 4.

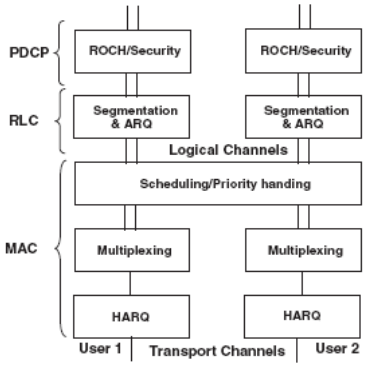

Figure 4. LTE functional distribution: MAC, RLC and PCCP layers[3]

To achieve LTE downlink data rates of 100 Mbit/s, 100 kbit have to be transferred/received from physical layer to/from MAC layer each millisecond. The target system should have enough resources and efficient communication means. As a consequence, we can expect the following requirements from the software layer where the LTE protocol stack is running:
- The software layer should expose homogeneous computing resources (e.g CPU cores)
- The software platform should support efficient data transfer paths between threads with advanced synchronization primitives (e.g. read-write locks)
- The software platform should support real-time execution models in order to cope with real-time requirements of the LTE protocol stack
- The software platform should expose standardized programming interfaces (e.g. POSIX interfaces) to make easier the porting of the protocol stack to the new eMuCo environment.
- The software platform should ensure predictable execution in terms of waiting states and task scheduling time
- The software platform should ensure that LTE threads are properly allocated to the available cores so that minimum energy is consumed and maximum processing power is delivered.

From the LTE application software design, state of the art is a multi-threaded implementation of a protocol stack, running it on a purely priority based fully pre-emptive Real Time Operating System (RTOS) and tune manually the thread priorities to achieve the desired behavior. Bringing this to multi-core creates little problems in porting but severe problems in efficient usage as awkward threads distribution to cores might result in no performance increase at all.

5. CONCLUSIONS

Our requirements analysis pointed out that execution support and portability are the most critical aspects when dealing with software applications for mobile embedded systems. An important aspect in constructing heterogeneous multi-core embedded systems is portability, in the sense that the application software should run without any modifications in the multi-core software execution environment. Additionally, multiple cores will only provide more speed when used with multi-threaded software. The software has to be designed to take advantage of the available multi-core hardware architecture.

The eMuCo mobile platform offers full portability to the user's applications by virtualization techniques and standardized interfaces (e.g. POSIX). It gives to the user's application developers the flexibility and scalability necessary to enable fast time-to-market. The limit of the exploitation capabilities of the mobile multi-core platform will be given by the application developer's imagination and the scheduling policy of the load balancer.

The development of a load balancer aware of the protocol stack interdependencies to take advantage of the multicore platform by concurrent and parallel execution, as well as, the load balancer service contracts for the user's applications to guarantee the required quality of service is an on going research topic of the eMuCo project.

6. REFERENCES

[1] eMuCo Deliverable 1.1. Requirements Analysis and Software Architecture Definition. **WWW.EMUCO.EU**

[2] Maximilian Eibl, „MPEG-1, MPEG-2, MPEG-4 Grundlagen - Vorlesung Mediencodierung". Medieninformatik, TU Chemnitz.

[3] Harri Holma, Antti Toskala. "WCDMA for UMTS: HSPA Evolution and LTE". 4th edition. John Wiley & Sons, Ltd. 2007.

ACKNOWLEDGE

eMuCo (www.emuco.eu) is a European project supported by the European Union under the Seventh Framework Programme (FP7) for research and technological development.