Adaptive OFDM System Design For Cognitive Radio

Qiwei Zhang, Andre B.J. Kokkeler and Gerard J.M. Smit
Department of Electrical Engineering, Mathematics and Computer Science
University of Twente
P.O. Box 217, 7500 AE Enschede, The Netherlands
Q.Zhang@utwente.nl

***Abstract*— Recently, Cognitive Radio has been proposed as a promising technology to improve spectrum utilization. A highly flexible OFDM system is considered to be a good candidate for the Cognitive Radio baseband processing where individual carriers can be switched off for frequencies occupied by a licensed user. In order to support such an adaptive OFDM system, we propose a Multiprocessor System-on-Chip (MPSoC) architecture which can be dynamically reconfigured. However, the complexity and flexibility of the baseband processing makes the MPSoC design a difficult task. This paper presents a design technology for mapping flexible OFDM baseband for Cognitive Radio on a multiprocessor System-on-Chip (MPSoC).**

***Index Terms*— Cognitive Radio, OFDM, Multiprocessor System-on-Chip (MPSoC), Design technology, Task Transaction Level (TTL) Interface**

I. Introduction

COGNITIVE Radio [1] is a promising technology to improve today's spectrum utilization. It can discover under-utilized spectrum by spectrum sensing and can adapt its transmission settings accordingly without causing interference to licensed users. A highly flexible OFDM system is considered for the Cognitive Radio baseband system where individual carriers can be switched off for frequencies which are occupied by a licensed user [2]. In such an adaptive OFDM system, not all processing characteristics are known at design time since processing tasks are not fixed. In order to support such an OFDM system, we propose a heterogeneous Multiprocessor System-on-Chip (MPSoC) platform which can be dynamically reconfigured. MPSoC technology has been commonly used for wireless systems (e.g. mobile phone) since the evolution of semiconductor technology enables to integrate more and more transistors on a single chip on which large systems are built. Applications are implemented on an MPSoC as software/hardware solutions. With the increasing complexity of applications, MPSoCs have to integrate more and more software/hardware components which makes MPSoC design a difficult task. In particular the communication and synchronization between different components is becoming a bottleneck from a performance and an energy point of view. In the traditional design approach, designers have to deal with communication and synchronization through low-level interfaces to integrate various software and hardware modules. Further, support for reuse is poor and a method for exploring tradeoffs is often missing. A task transaction level interface (TTL) approach was proposed in [3] to raise the abstraction level. We propose to use the TTL approach both for developing the adaptive OFDM application specification and as a platform interface for implementing the application on MPSoC architectures.

The paper is organized as follows. In section II, we will discuss the adaptive OFDM baseband for Cognitive Radio. A multiprocessor SoC architecture is proposed to support adaptive OFDM baseband processing in section III. In section IV, we present the TTL approach to adaptive OFDM system design for Cognitive Radio. The last section concludes the paper.

II. Adaptive OFDM Baseband For Cognitive Radio

Theoretically, an OFDM based Cognitive Radio system can optimally approach the Shannon capacity in the segmented spectrum by adaptive resource allocation on each subcarrier, which includes adaptive bit loading and adaptive power loading. For Cognitive Radio, OFDM also offers other benefits such as high data rates and robustness to multipath delay spread and is easy to integrate with spectrum sensing because of the hardware reuse of FFT cores.

Therefore, OFDM is a good candidate for the Cognitive Radio baseband system.

Since Cognitive Radio can operate in different frequency bands, provide multimedia services with various QoS and cope with different channel conditions, it has to adapt to different OFDM systems. OFDM standards vary a lot in the number of carriers and the transmission bandwidth. Within one standard, multiple modes can be used to cope with various transmission channels by adopting different parameter sets. Although there are a lot of differences in various OFDM systems, the basic baseband processing is rather similar. This makes OFDM based Cognitive Radio feasible. An OFDM baseband receiver generally consists of the following basic tasks (Figure 1):

- **Packet/frame detection and synchronization** is used to determine the starting point of an OFDM frame. This is usually achieved by correlating the received signal with known preambles.
- **Frequency offset estimation and correction** are done to remove the frequency offset which destroys the frequency orthogonality of subcarriers. A frequency correction coefficient determined by the frequency offset estimation is multiplied by each OFDM data symbol.
- **Channel estimation and equalization** are used to correct the frequency selective fading. Due to the robustness of OFDM to frequency selective fading, less complex frequency domain equalization techniques can be used.
- **Guard time removal** is done for every OFDM symbol.
- **FFT** is the basic component of all OFDM systems which is also one of the most computational intensive tasks.
- **Phase offset tracking and correction** correct the frequency offset residual error by using the pilots in each OFDM symbol.
- **De-map** transforms the complex numbers to a bitstream according to their modulation schemes. Different modulation types can be loaded on each subcarrier by applying adaptive bit-loading.
- **De-interleaving** is the opposite task of data interleaving which is intended to reduce the effects of burst errors on a receiver. De-interleaving is the inverse operation of interleaving.
- **Channel decoding** exists in all communication systems not only in OFDM systems. However, OFDM is almost always used in conjunction with channel coding to create coded OFDM (COFDM). A Viterbi code is commonly used, but other channel codes such as turbo code or LDPC code may also be applied. The code rate can be adaptive to provide different degrees of error protection.

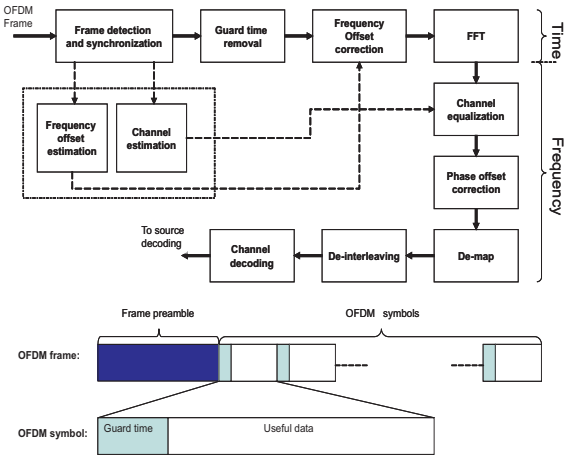

Fig. 1. Basic OFDM processing tasks

For simplicity the tasks performed by the receiver are shown. The transmitter basically does the inverse operations which are less complex. The aforementioned functional blocks can be found in all OFDM systems, but different standards may use different algorithms for each functional block. This means the communication system can select an algorithm to perform each function depending on the requirements of the system. For example, different channel encoders/decoders (codecs) can be applied to achieve different QoS requirements. For a specific algorithm, there are also opportunities for adaptivity by changing parameters of the algorithm. For example the size of FFT and the code rate of the Viterbi codec can be tuned by different standards or modes. Therefore a fully reconfigurable hardware platform is required to support the adaptivity of Cognitive Radio.

III. Multiprocessor System-on-Chip Architecture

The reconfigurable platform we propose for Cognitive Radio is a heterogeneous MPSoC architecture shown in Figure 2. This SoC is a heterogeneous tiled architecture, where tiles can be various processing elements including General Purpose Processors (GPPs), Field Programmable Gate Arrays (FPGAs), Application Specific Integrated Circuits (ASICs) and

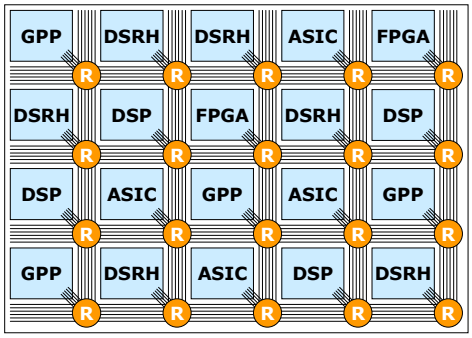

Fig. 2. An example of a heterogeneous Multiprocessor System on Chip (MPSoC). DSRH = Domain Specific Reconfigurable Hardware

Domain Specific Reconfigurable Hardware (DSRH) modules. The tiles in the SoC are interconnected by a Network-on- Chip (NoC). Both the SoC and NoC can be dynamically reconfigurable, which means that the programs (running on the reconfigurable processing elements) as well as the communication links between the processing elements are configured at run-time. Different processing elements are used for different purposes. The general purpose processors are fully programmable to perform different computational tasks, but they are not energy-efficient. The dedicated ASICs are optimized for power and cost. However, they can not be reconfigured to adapt to new applications. FPGAs which are reconfigurable by nature, are good at performing bit-level operations but not that efficient for word level DSP operations. The Domain Specific Reconfigurable Hardware (DSRH) is a relatively new type of processing element, where the configurable hardware is tailored towards a specific application domain. The Montium [4] tile processor developed at the University of Twente is an example of DSRH. It targets the digital signal processing (DSP) algorithm domain, which is the heart of the wireless baseband processing. In our previous work [4] [5] [6], several DSP algorithms used in wireless communication have been mapped onto the Montium architecture. The implementation results show that the Montium architecture is flexible enough to adapt to different algorithms with good energy-efficiency. Therefore the reconfigurable platform we propose not only targets flexibility but also energy-efficiency.

IV. Design Technology: A Task Transaction Level Interface Approach

The MPSoC technology has a range of advantages such as high performance, energy efficiency and reconfigurability. However, the design of an

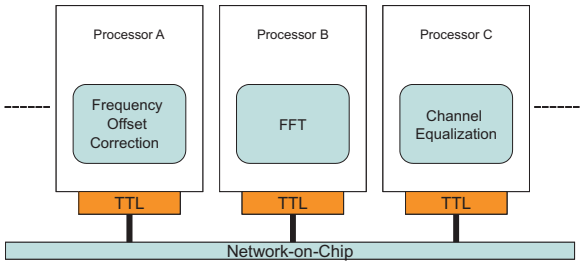

Fig. 3. The TTL interface approach for OFDM processing tasks

MPSoC is a difficult task. First, the applications to be mapped on the MPSoC become more and more complex. they consist of more and more tasks and some of tasks even change their behavior dynamically. Second, in order to map tasks to different components on an MPSoC, designers have to deal with the low level interfaces for the inter-component communication and synchronization which become a bottleneck from a performance and an energy point of view. Further, opportunities for reuse of hardware and software modules are limited and a method for exploring their trade-offs is missing. Therefore, there is a gap between the application models used for specification and the optimized implementation of the application on an MPSoC. A task transaction level interface approach [3] was proposed to help to close the gap by raising the abstraction level. We propose to use the TTL approach both for developing the adaptive OFDM application for Cognitive Radio and as a platform interface for implementing the application on the targeted MPSoC, see Figure 3. The TTL interface approach has the following advantages 1) inter-task communication is made explicit, 2) at an early design stage hardware/software trade-offs can be made, 3) it allows to experiment with new design alternatives (e.g. new algorithms, new implementations for existing algorithms), 4) it fits well to the MPSoC architecture, 5) it can generate an executable specification.

A. Introduction to The TTL Interface

The TTL approach [3] is a design method for implementing parallel media processing applications on MPSoCs. Based on their previous work on task-level interfaces, the TTL interface unifies all these interfaces as a set of interpretable interface types. Application developers can use TTL to build executable specifications. On the other hand, TTL allows efficient implementations of the platform infrastructure and the tasks integrated on top of

it. A set of different interface types in the TTL offers support for different inter-task communication styles, which satisfy the varying needs for modelling media processing applications and the variety in platform implementations. TTL also supports multi-tasking and reconfiguration.

B. A Design Case: HiperLAN/2 Receiver

As a first step, the TTL interface approach is used to model a HiperLAN/2 [7] baseband receiver to demonstrate its efficiency for modelling basic OFDM processing tasks. OFDM is partitioned into several computational tasks which communicate with each other, see Figure 1. The TTL approach enables the separation of computational tasks from the details of the communication. Communication is performed by calling the TTL interface function provided by the TTL library. The functional behaviors of computational tasks are specified in C code for the system level design. Communications are attached to TTL channels. All these channels are connected to setup a task graph. Here we give a pseudo code example of the FFT task implementation.

```
process Task_FFT {
initialization;
while (true)
{\\ declaration of local variables
 complex Y[64],Z[64];
 for(i=0;i<64; i++)
 \\ read input samples
 InRead(&(Task_FFT->inport), &Y[i]);
 \\ perform the FFT
 call function Z=FFT(Y);
 for(i=0; i<64; i++)
 \\ write output samples
 OutWrite(&(Task_FFT->outport), &Z[i]);
 }
}
```

While the condition is true, samples are read into the local buffer from the channel which is connected to the input port. When samples are ready, data processing is executed by calling the FFT function. Results are written to the channel connected to the output port. Both synchronization and data transfer are done by simple read and write function calls. For each task of an application, one or more task implementations may be provided. A task implementation is the implementation of a task on a particular tile, e.g. object code for an ARM or configuration data for a Montium. Computation components can be plugged-in and replaced as functions, which allows application adaptivity within the same TTL framework. So, it is advantageous to use the transaction level model to speed up simulation and allow

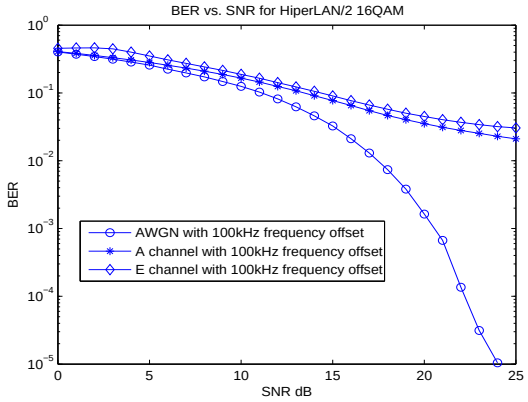

Fig. 4. A TTL design case for OFDM processing: HiperLAN/2 16QAM

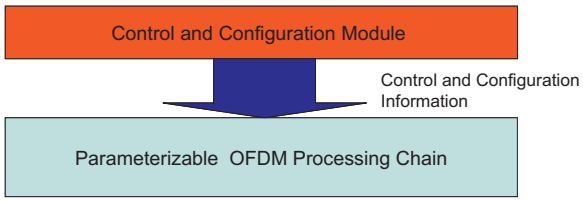

Fig. 5. The conceptual model of adaptive OFDM

exploring and validating design alternatives at a high level of abstraction. The TTL model of the application can be implemented on the targeted platform providing that the TTL shells [3] are available for each processor. We implement the TTL model of HiperLAN/2 on a software shell executing on a PC. The implementation result of a 16QAM OFDM receiver is tested with 100kHz frequency offset in three different channel models: an AWGN channel, an indoor channel ('A' channel) and an open space channel with a large delay spread ('E' channel). These multipath channel models for HiperLAN/2 have been defined for standardized system analysis [7]. In Figure 4, we show the simulated BER vs. SNR for our TTL implementation of HiperLAN/2.

C. The TTL Approach For Adaptive OFDM Baseband

Since the Cognitive Radio baseband system is an adaptive OFDM system, a proper model for a dynamic application is needed. Conceptually, adaptive OFDM can be modelled as a parameterizable OFDM processing chain controlled and configured by a control and configuration module, see Figure 5. The parameterizable OFDM part can be reconfigured to adapt to different OFDM systems. To give an indication, Table I shows the parameter set for the OFDM receiver in Figure 1. The control and config-

Processing task	Parameters
Packet/Frame Detection and Synchronization	Frame size
	Frame header size
	Type of sync.
Frequency Offset Estimation&Correction	Type of estimation
	Pilot for estimation
Channel Estimation and Equalization	Type of equalization
	Pilot for estimation
	Num. of filter tap
Guard Time Removal	Symbol size
	Guard time size
FFT	FFT size
Phase Offset Tracking&Correction	Pilot for tracking
	Tracking method
De-mapping	Type of modulation
De-interleaving	De-interleaving method
	De-interleaving size
Channel Decoding	Type of coding
	Polynomial
	Code rate
	CRC size

TABLE I

PARAMETER SET FOR OFDM RECEIVER

uration module can dynamically set the parameters for OFDM processing tasks. In addition to parameter settings, the topology of the task graph can also be changed dynamically by the control and configuration module (for example a digital filter can be added before synchronization to remove unwanted signals). Therefore, the static task graph model is no longer sufficient. In [8], the TTL reconfiguration interface is proposed to describe the dynamic behavior of applications and facilitate the implementation on an MPSoC. Apart from the processing tasks, there is a *configuration manager* (CM) which manages the configuration of tasks. The configuration service provided by CM includes: creation and deactivation of tasks, channels and ports connection and task states transition (e.g. stop, pause and running). The TTL reconfiguration service offers an interesting opportunity to explore Cognitive Radio, a highly dynamic application, on an MPSoC.

Based on the TTL model of OFDM processing tasks, we will build a control and configuration module on top of it by applying the TTL reconfiguration interface. Finally all TTL interfaces will be implemented on the targeted MPSoC. In Figure 6 this is illustrated together with a possible distribution of tasks over different tiles.

V. CONCLUSION

This paper presents a design methodology for implementing an adaptive OFDM system, which is a part of Cognitive Radio baseband, on an MPSoC. A task transaction level interface approach is proposed for both modelling the adaptive OFDM application at the abstract level and mapping the application on the targeted MPSoC. TTL helps to close the gap between application modelling and platform implementation.

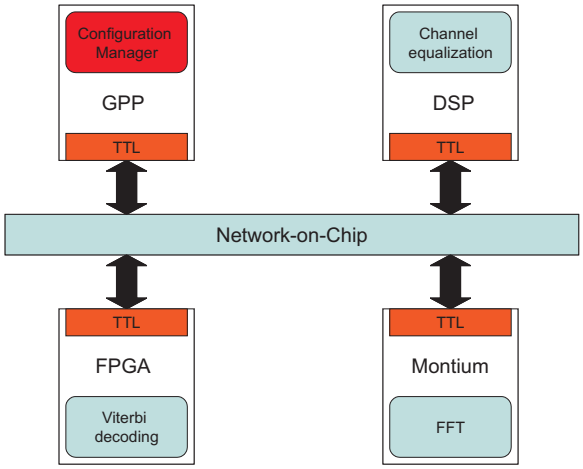

Fig. 6. TTL reconfiguration interface model for adaptive OFDM on MPSoC

ACKNOWLEDGMENTS

We acknowledge the support for TTL by Philips Research. The work is sponsored by the Dutch Ministry of Economic affairs Freeband AAF project.

REFERENCES

[1] J. Mitola III. *Cognitive Radio: An Integrated Agent Architecture for Software Defiened Radio*, PhD Thesis, Royal Institue of Technology, Sweden, May. 2000.

[2] T.A. Weiss and F.K. Jondral Spectrum Pooling: An Innovative Strategy for the Enhancement of Spectrum Efficiency, *IEEE Commun. Mag.* vol. 24, Mar. 2004

[3] Pieter van der Wolf et al. Design and Programming of Embedded Multiprocessors: An Interface-Centric Approach, In *Proceedings of ISSS+CODES*, Sept. 2004

[4] Paul M. Heysters. *Coarse-Grained Reconfigurable Processors; Flexibility meets Efficiency*, PhD. Thesis, Sept. 2004

[5] Gerard K. Rauwerda et al. Implementing An Adaptive Viterbi Algorithm In Coarse-grained Reconfigurable Hardware, In *Proceedings of ERSA*, 2005

[6] Arnaud Rivaton, Jerome Quevremont, Qiwei Zhang, Pascal T. Wolkotte, and Gerard J.M. Smit, Implementing Non Power-of-two FFTs On Coarse Grain Reconfigurable Architectures, In *Proceedings of the International Symposium on System-on-Chip (SoC 2005)*, 2005

[7] ETSI Broadband Radio Access Networks (BRAN); HIPERLAN type 2; Physical (PHY) Layer Technical Specification ETSI TS 101 475 V1.2.2 (2001-02), 2001

[8] Jeffrey Kang et al. An Interface For the Design and Implementation of Dynamic Applications on Multi-processor architectures, In *Proceedings of ESTImedia*, 2005