

EFFICIENT AND PORTABLE SDR WAVEFORM DEVELOPMENT: THE NUCLEUS CONCEPT

Venkatesh Ramakrishnan, Ernst M. Witte,
 Torsten Kempf, David Kammler,
 Gerd Ascheid, Rainer Leupers, Heinrich Meyr
 Institute for Integrated Signal Processing Systems,
 RWTH Aachen University, Germany
 ramakrishnan@iss.rwth-aachen.de

Marc Adrat, Markus Antweiler
 and Research Establishment for Applied Science (FGAN),
 Dept. FKIE/KOM, Wachtberg, Germany
 adrat@fgan.de

ABSTRACT

Future wireless communication systems should be flexible to support different waveforms (WFs) and be cognitive to sense the environment and tune themselves. This has led to tremendous interest in software defined radios (SDRs). Constraints like throughput, latency and low energy demand high implementation efficiency. The tradeoff of going for a highly efficient implementation is the increase of porting effort to a new hardware (HW) platform. In this paper, we propose a novel concept for WF development, the Nucleus concept, that exploits the common structure in various wireless signal processing algorithms and provides a way for efficient and portable implementation. Tool assisted WF mapping and exploration is done efficiently by propagating the implementation and interface properties of Nuclei. The Nucleus concept aims at providing software flexibility with high level programmability, but at the same time limiting HW flexibility to maximize area and energy efficiency.

I. INTRODUCTION

Flexibility in modern radio devices has been proposed to serve different goals, such as efficient multi-modal or multi-standard transmission in order to support compliance with new and old WFs, promote interoperability and reduction of costs via modular and parametric design. Furthermore, system cognition results in high flexibility requirements, which need to be efficiently addressed by the transceiver leading to a solution like SDR system.

One of the key requirements for flexibility is WF portability across various HW platforms. Portability cannot be defined as a binary term but as an inverse to the porting effort [1]. Portability can offer several key advantages like implementation reuse and fast time-to-market. It is also a goal of the joint tactical radio system (JTRS) program [2]. Pure software solutions for a SDR offer maximum portability. Due to low implementation efficiency and tough throughput, latency constraints pure software

This research project was performed under contract with the *Technical Center for Information Technology and Electronics (WTD-81)*, Germany.

radios are not yet feasible. Therefore, currently some of the processing intensive components are still implemented in dedicated application-specific integrated circuits thereby limiting portability.

WF implementation in C/C++ is portable, at-least, across platforms with programmable processing elements (PEs). But the implementation is not efficient enough to meet the realtime constraints. For example, assembly can be more efficient by one order of magnitude than a C program [3]. WF implementation at such a low level needs tremendous porting effort. Furthermore, C/C++ is not able to provide adequate support needed for physical (PHY) layer signal processing e.g. fixed point types, circular buffers, etc. Therefore, one of the key challenges that needs to be addressed in SDR development is to enable WF portability and maintain implementation efficiency at the same time.

This challenge is currently addressed by raising the abstraction level for WF implementations leading to library based approaches. As depicted in Figure 1, the basis of current approaches [4–7] is a library with basic components of a few WFs. A component X is shown as CX in Figure 1. The WF is constructed as a structured assembly of components, each of the components implementing a part of the WF functionality [8].

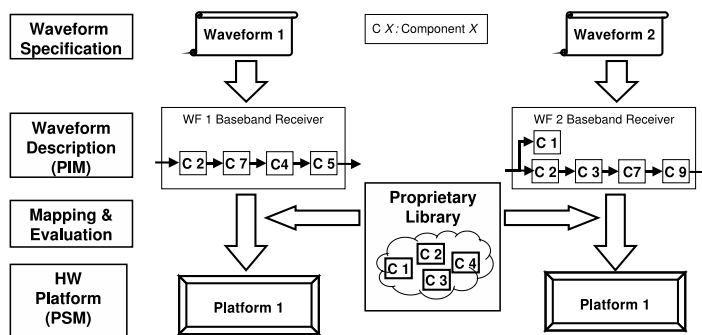


Figure 1. Traditional Library Based Waveform Development

The library provides efficient implementations of some basic components. From a platform independent model (PIM) of a WF a platform specific model (PSM) can be

arXiv:0906.3313v2 [cs.IT] 23 Jul 2009

derived using the library. The advantage of this approach is that the same PIM can be used for porting to another HW. Also, it is flexible to develop components of different models independently. Library based approaches enable efficient utilization of heterogeneous multiprocessor system-on-chips (MPSoCs) due to the presence of hardware-aware implementations.

Currently, vendors maintain their own custom-built libraries for their platforms. Most of them are inaccessible to the public. There are no criteria for selecting components for the library which is crucial for a common library. We believe that the standardization of the library and its (components) interfaces is necessary and will lead to the development of a waveform description language (WDL). This will enable vendors not only to describe the WFs using this library but also to provide efficient implementations. Some important aspects which have to be considered when building such a library are highlighted in the next section.

This paper proposes a new concept to enable WDL based WF development offering the following contributions

- An analysis is made on existing SDR WF development approaches to highlight portability and efficiency issues.
- A new concept is proposed for tool assisted, portable and efficient WF development. Details on processes in a WDE are also presented.
- Challenges in envisioning the concept are highlighted.

II. RELATED WORK AND MOTIVATION

Various solutions have been existing for developing SDRs. Solutions differ in portability, efficiency, usage of WDL, component granularity and reusability. This section provides an overview of some of the solutions and motivates the need for our Nucleus concept.

Many library approaches that exist for WF development are component based and/or model driven [4–7]. As illustrated in Figure 1, some of the approaches [5, 6] have developed libraries for one particular HW platform (HW-specific). Others like [7, 9] have developed special libraries for one WF (WF-specific). Some library approaches are even both HW and WF specific. For example, the library in [7] has been designed to support one type of WF, WiMax (with several modes), for one fixed HW using a TMS320TCI6482 DSP. The specifics of the above approaches limits porting of WFs. Since the components themselves and their interface are not standardized, implementations from different vendors are not compatible in most scenarios and therefore lead to high integration effort. Furthermore, the components of the library are functionality based (e.g. modulator/rake receiver). This results in limited commonality among WFs leading to a reduced reuse of

components and a huge implementation effort for a new WF. For example, when a WF uses a different modulation scheme whose component is not present in the library, the component has to be implemented. Such a functionality based library cannot be used for specifying a WF and will not lead to a WDL. A library of algorithmic kernels, using which the functionalities can be built, will improve usage among WFs.

The term WDL has been first introduced by E.D. Willink in [10]. The need for a WDL arises from the specification-to-implementation problem. Specifying WFs using textual or mathematical/formal methods is not efficient enough for automated implementation. A WDL aims at capturing the WF specification by a behavioral model and derive the implementation automatically. Though a WDL library is discussed, details about the library and portability issues are not provided.

In [11], Gudaitis presents a WDL concept based on the unified modeling language (UML), Matlab and the extensible markup language (XML). The FM3TR WF has been used as an example to demonstrate the concept. Though language components of a WDL are presented, information about the WDL library, reusability, realization and portability is not given.

The radio description language developed by Vanu Inc, targets WF portability [12]. However, they have used C++ as signal processing implementation language. It is also mentioned that few components (referred to as modules) were reused unchanged from previously built WFs. But, information about the compute-intense (demanding) and the reused components is lacking. This is necessary because highly demanding components need to be optimized to meet the WF requirements. The portability effort is highly influenced by demanding components. From our investigations ([1, 3]), implementation efficiency is low when a high level general-purpose language is used even with good compilers.

A methodology for selecting the components (referred to as common operators) for SDRs has been proposed in [13]. Similar to other existing library based approaches, the authors concentrate on identifying the common parts that exist in the implementation of functionalities and not on the algorithms. Moreover, portability and mapping issues are not considered.

Few other important aspects have to be considered when targeting a library based approach enabling WDL. The data rate of near-future mobile systems will be in the order of few hundred Mbits/sec requiring a processing power of hundreds of Giga operations per second. For such compute-intense systems, energy efficiency will be the key factor for the system design. HW platforms made up only of PEs with general purpose architectures cannot provide the required

efficiency. Heterogenous MPSoC platforms with multiple processors having special architectures are better candidates with limited HW flexibility in order to gain efficiency.

In general, the selection of an algorithm has a huge impact on the performance. One algorithm that is good for one operational scenario might not be good for another. Hence, it is necessary for the WF developer to have the flexibility of exploiting different algorithms even with a fixed HW platform.

The granularity of the components is another important aspect that heavily influences re-usability and efficiency in a library based WDL. If the components are too fine-grained, like a subtractor in [10], it is inefficient to use them for constructing a WF. If they are too coarse-grained, like a complete convolutional decoder in [12], it limits reuse. Also, providing optimized HW implementations based on such components is not efficient due to limited re-use. Therefore, the granularity should be between the two extremes representing a substantial part of WFs (critical) and the same time enabling re-use.

Even with a competent library, spatial and temporal mapping of a WF on a HW platform is a key factor that determines overall system efficiency. For an efficient mapping, the WF description should encode the information about its ideal mapping. This information could be implementation properties like bit-width, type of scaling, etc. which can provide huge performance difference. The library and its components should be build such that it provides these properties using a WDL in an abstract manner. Tools in the waveform development environment (WDE) can use these properties to assist the developer for efficient mapping.

New SDR design approaches, in addition to being neither WF nor HW specific, should consider the above aspects during system design. Such approaches should enable (semi-) automatic generation of the implementation from the WF description that is not only efficient but also portable.

III. THE NUCLEUS CONCEPT

To overcome the drawbacks of traditional library based approaches, a new classification of library elements called Nuclei is proposed targeting reusability, portability and implementation efficiency. Considering the important aspects for a library based WF development, the Nucleus concept approaches system design by the following key principles:

- Limit HW flexibility to the minimum required level (for example, architecture of PEs, communications and memories)
- Maximize area and energy efficiency
- Manage/exploit flexibility by means of high level programmability

A. DEFINITIONS

- **Nucleus** : A Nucleus is defined as a critical, demanding, flexible, algorithmic kernel that captures common functionalities within and/or among WFs.
- **Genre**: A Genre is defined as a set of algorithms containing the same Nucleus. An example for a Genre is illustrated in Figure 4.
- **Flavor**: A Flavor is defined as an efficient and optimized implementation for one Nucleus. There can be several Flavors for one Nucleus. The Flavors can be based on several algorithms. Each or all of the Flavor(s) can have tunable parameters.

B. METHODOLOGY

As shown in Figure 2, the concept proposes to build a library of kernels from a class of WFs. A Nucleus kernel is shown as N in the figure. These kernels are of algorithmic nature and do not need to represent any WF or implementation specific features. The library forms the basis for constructing a WF from the specification (Figure 2).

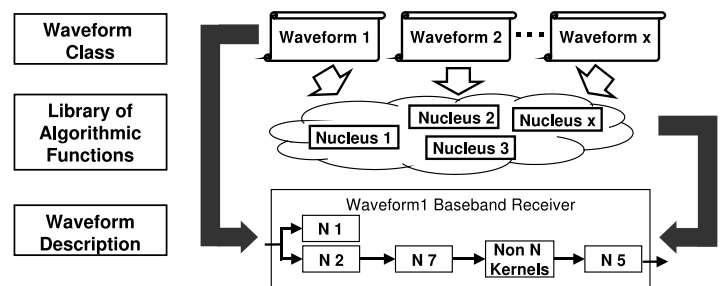


Figure 2. The Nucleus Concept

The key difference to the component-based WF development is that the Nuclei represent not only the functionality of signal processing blocks on a different level of abstraction, but also their properties required for exploration. Properties that affect the WF performance are essential for exploration. Therefore, information on the input data structure, bit-width, type of scaling, usage of truncation or rounding etc. will be abstracted and provided to the WF developer to aid exploration.

A Nucleus does not necessarily represent functionality of parts of the WF directly (e.g. equalizer or demodulator). The functionality is built using these kernels. Since the members of the Genre have the same computation and communication pattern, they can be implemented using their Flavors. But, this might need some pre or post-processing in addition (Figure 3). Since Flavors are optimized implementations, they can introduce additional constraints, e.g. requiring a certain input data format like Q15. Part of the pre-processing in this case would be the adoption

of the data format. Flavors need to be flexible enough to implement the extra processing in an efficient way.

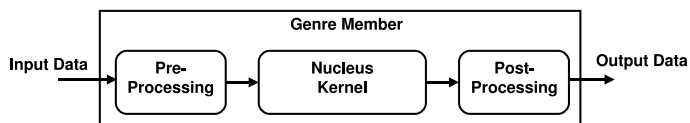


Figure 3. Nucleus Genre Relationship

One important aspect of the proposed approach is that different vendors can provide Flavors of some or all of the Nuclei in the form of a board-support-package (BSP). Flavors could be assembly codes for an ASIP or DSP and/or IP cores in HW or on a reconfigurable FPGA. Therefore, a Flavor in general can be seen as a bundle of the PE and low-level software (if applicable). Due to this bundling it will be possible to capture the Nucleus functionality by means of a HW independent high level Nucleus application programming interface (API). This enables to describe a WF independently of the targeted MPSoC platform. Still, the WF can be mapped efficiently to the platform due to the bundled Nuclei low-level implementation. Participation of different vendors is also possible due to the standardization of the functionality and the interface of Nuclei. Vendors can also provide simulation models of the Nuclei e.g. in Matlab/Simulink.

The Nucleus concept enables WF developers to program the HW platform on a very high level (Nucleus level) as the Nuclei are visible via algorithm factors (Genre, Flavors). The developer will still be able to explore among the existing Flavors for WF implementation. Since the programming is done at the Nucleus level hiding the underlying HW platform, it is efficient and simple for the developers.

Even though new WF implementations go through many rounds of development and adjustment, the core algorithmic kernels often evolve at a relatively slow rate [14]. Since the proposed concept exploits such core common structures that exist in receiver algorithms [15], various WFs can be built even after the availability of the HW platform. Due to existence of such kernels in general purpose applications research in the same direction is also done by computer science experts [16].

C. AN EXAMPLE

FFT is an example of a Nucleus. FFT has been used for several decades in diverse applications. But the core algorithm itself has not evolved in the same pace as the implementations [17]. The granularity is at an optimum level that enables reuse in various applications. We are using the existing work based on the FFT algorithm as an example for explaining the definitions in our concept. The references to the existing work are given as the explicit proof for our arguments. Since the motive is not to explain the example

but to show the relationship between the definitions only brief comments are given for each Genre member.

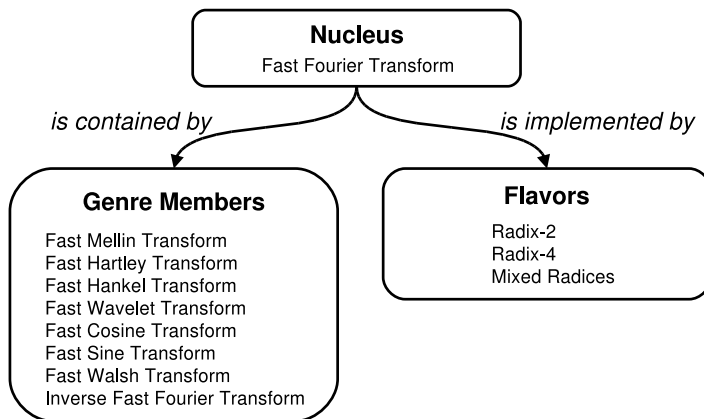


Figure 4. Nucleus Example

Figure 4 depicts the relation between the definitions that were introduced in the concept. As illustrated in the figure, for one Nucleus (e.g. FFT) there could be several members belonging to its Genre. The Genre member fast Mellin transform (FMT) can be realized using the FFT kernel [18]. But this requires pre and post-processing as shown in Figure 5. Pre-processing in this case is re-sampling the input and post-processing is the amplitude calculation. Flavors need to implement this extra processing also efficiently. Fast Hankel transform (FHT) is the (one dimensional) Fourier transform of the Abel transform [19]. Here, the pre processing portion is the Abel transformation. Also, FHT using fast cosine transform and fast sine transform can be found in [20]. FFT can be used as a basis for doing fast wavelet transforms [21]. Discrete cosine transform and Walsh transform can also be realized with pre and post-processing [22]. Efficient implementations using an FFT kernel as basis for realizing Walsh-Hadamard transform, discrete cosine transform (DCT) and discrete Hartley transform exist in [23]. In addition to that, the inverses of some of the above transforms belong to the same Nucleus, e.g. IFFT and IDCT.

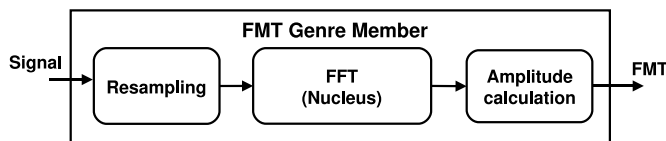


Figure 5. Nucleus Genre Example

Flavors for FFT have been existing using different algorithms and radices. For example, an FFT Flavor might use either a Cooley-Tukey or Sandey-Tukey algorithm depending on the decomposition of FFT stages. Flavors can be available by using a single radix like radix-2 or radix-4. If the number of FFT points is not a multiple of the basic radices, combination of different radices is present in one Flavor (Figure 4). The efficient implementations from the

BSP could be assembly codes for an application specific instruction processor (ASIP) or digital signal processor (DSP) (e.g. optimized FFT implementations from TI library [24]) and/or IP cores in HW or on a reconfigurable field programmable gate array (FPGA) (e.g. FFT intellectual property (IP) cores from Xilinx [25] and Altera [26]). The TI library [24] even provides support for IFFT. Simulation models for Flavors are also provided by vendors [24–26]. One Flavor might be different to another in algorithm, flexibility and performance. For example, a Flavor with radix-2 has more FFT stages compared to radix-4. Therefore, it is more flexible to scale the outputs in the intermittent stages than radix-4. Flavors can offer a wide performance range with respect to latency and throughput, e.g. [25].

From the above references, it can be inferred that the members in the Genre can be realized using the same FFT kernel. As a special case, an FFT kernel can be used for both PHY layer (e.g. OFDM) and application layer (e.g. video/audio compression) [22] signal processing.

IV. WAVEFORM DEVELOPMENT ENVIRONMENT CONCEPT

In general, a WDL forms the basis of a WDE. WDE denotes a system-level tool suite for automated WF development for SDRs. The tool-suite should cover the entire WF development process of requirements, design, implementation, integration and testing for SW and HW [27]. Figure 6 depicts these processes with respect to our proposed concept. While the WF description captures the requirements and design of the WF under consideration, implementation is provided by the BSP. Integration and testing processes are done in the mapping and evaluation stage (Figure 6). NI represents the Nucleus implementation of one Nucleus kernel. The first subscript of NI denotes the Nucleus API (NA) and second subscript denotes the PE for which the implementation is available in the BSP. Therefore, NI_{23} denotes the NI of Nucleus 2 on PE 3.

A. WAVEFORM DESCRIPTION

The WF developer uses WDL for describing the WF using the API on the basis of the Nuclei library. The WF description will then consist of Nuclei kernels and non-nuclei (NN) kernels connected by communication links (Figure 2). The description also contains information about critical paths, constraints and control flow of the WF. The NN kernels reflect the other computation-light functionalities of the WF.

B. WAVEFORM IMPLEMENTATION

The BSP plays an important role in the WF implementation. It provides efficient implementations of some or all of the Nuclei kernels. This can enable tool based WF

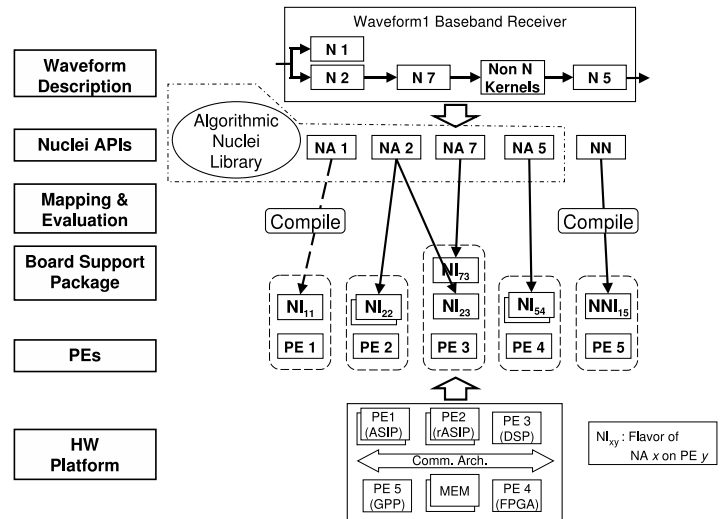


Figure 6. Waveform Development Concept

development where the Nuclei kernels are replaced by the BSP and the rest of the WF (like NN kernels) is built using a conventional approach (e.g. C/C++ code). If the implementation for a kernel that is in the WF description is not available in the BSP, it also has to be built using the conventional approach. This is indicated by the dotted line in Figure 6 for NA1. However, depending on performance requirements, Nuclei kernels for which the BSP is missing may have to be optimized for a given platform.

Since the software code of the NIs is highly optimized it achieves high performance, but it is bound to that particular PE. As the input data structure of a Flavor will be known from the BSP, WDE can generate additional logic required for gluing Flavors. This results in (semi-) automatic generation of WF implementation. Some of the features of WDE concept using the Nucleus approach are

- The Flavors in different BSPs can use different algorithms to implement the same behavior while offering various tradeoffs.
- There can be several and different Flavors for one Nucleus in one BSP. This provides the flexibility to the WF designer to choose among the existing implementations according to the need.

C. MAPPING

The WF description and the NI from the BSP form the basis of WF mapping onto the HW platform. Even though Flavors are efficient, the overall system performance is not guaranteed and it is heavily influenced by the spatial and temporal mapping. Figure 6 sketches the scenario of spatial mapping for a given HW platform. The key difference to the traditional WF spatial mapping is that the kernel from the WF description is mapped not to the HW PE, but to the available NI from the BSP. This is possible due to the bundling of NI to the PE. As shown in Figure 7, there might

be several choices for selection depending on the BSP.

- For one Nucleus, there can be only one NI (Figure 7a).
- For one Nucleus, there can be several NIs available for one PE (Figure 7b).
- For one Nucleus, there can be several NIs available for several PEs (Figure 7c).
- On one PE, there can be several NIs available for different Nuclei (Figure 7d).
- NIs can come from different vendors (Figure 7d).

The presence of several mapping choices advocate the need for tool assistance. The choice of the NI will be based not only on the performance but also on the data structure of the neighboring NI interfaces. Since the NN kernels are computationally light, they might be mapped to the general purpose core (PE5) as shown in Figure 6.

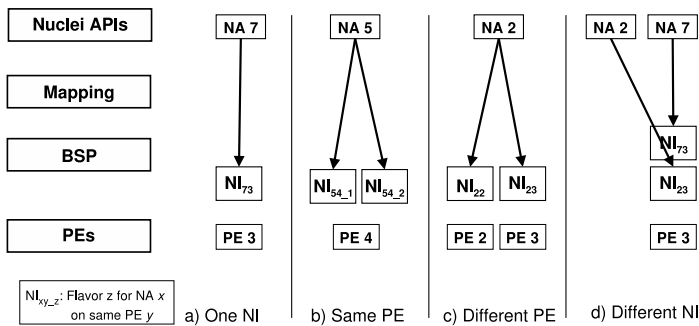


Figure 7. Flavor Selection Scenario

One or more Nuclei (and/or non-nuclei) kernels can be mapped onto a single PE. This requires temporal mapping/scheduling of kernels based on the data dependencies. Evaluation is needed to make sure that the constraints of the WF are met.

D. EVALUATION

A detailed evaluation of the mapping decisions is necessary primarily to check if the WF constraints are met. It will also aid in finding a mapping that maximizes total system performance. The WF constraints need to be verified in terms of latency, throughput, bit error rate and critical loops. To maximize total system performance, metrics are needed with which different mapping decisions can be evaluated. The following metrics can be used:

- Increasing data localization
- Minimizing communication
- Minimizing synchronization
- Maximizing PE utilization
- Maximizing energy efficiency

The mapping exploration should be done in an iterative manner and at different abstraction levels to trade-off between simulation speed and accuracy. The WDE will provide the infrastructure necessary for doing the exploration and simulation for evaluating the mapping decisions. The mapping challenge is to identify the right choice of

NIs that are not only compatible to each other and meet the requirements of the WF but also maximize the overall system efficiency. Since the mapping and evaluation will be an iterative process making it tool assisted can ease the whole process.

E. ADVANTAGES

The WDE methodology using the Nucleus concept delineates WF requirements in a manner that is neither application specific nor HW specific. This leads to the following main advantages in realizing SDRs.

- **Portability:** Since the WF description is not HW specific, the same description can be ported to multiple HW platforms.
- **Efficiency:** HW specific Flavors of the WF aids in efficient implementation.
- **Reusability:** Since the approach is not WF specific the same NAs can be used for describing several WFs and this will enable WDL based radio development.
- **Flexibility:** Since the interface and functionality of the Nuclei will be known through API, different vendors can provide different Flavors.
- **Future Use:** This approach paves way for capturing future WFs.

V. CHALLENGES

For realizing the Nucleus concept, it is necessary to explore the Nucleus design space in the algorithmic, HW and tools domain in order to identify a basic set of Nuclei which will allow the realization of a wide range of WFs. Furthermore, it is important to investigate the flexibility required for each NI to allow the usage for multiple WFs. All these aspects always need to consider energy efficiency. The challenges that exist to realize the Nucleus concept are described in this section.

Nuclei Identification One big challenge is the identification of Nuclei kernels that exist among WFs. The goal is to modularize or reformulate the algorithm in such a way that efficient implementation and reusability across multiple WFs is enabled. The energy efficiency of the Nucleus interconnect structure will have a significant influence on the whole SDR system. The interconnection issues are tightly coupled with the definition of a proper API for a Nucleus library and therefore it will provide good hints for WDL specification.

Mapping Mapping a portable WF onto a HW platform spatially and temporally meeting the constraints is one of the key challenges. This requires exact knowledge of the HW interfaces of the NIs in order to deal with the required additional glue logic to connect the different HW blocks and manage their execution.

Waveform Development Environment Providing easy-to-use programmability for complex receiver systems is

essential to exploit efficiently the system resources. A programming model that can bridge the gaps between WF, HW platform and mapping is needed. A fast system simulation environment is key in order to validate mapping decisions.

Cross Layer Optimization For an efficient radio design it is mandatory to look at the PHY-media access control (MAC) interactions and enable cross-layer optimization. The focus should be not only on the modularization and reuse of higher layer functionalities but also on the interfaces between the layers.

SCA Compliance Supporting software communications architecture (SCA) compliant WF development is key for implementing military WFs. The proposed concept is aligned with several aspects from the JTRS SCA community. For instance, the information that is fed into the mapping process of the WDE resembles the software assembly descriptor which describes the assembly of software components and HW devices. Furthermore, additional logic need to be generated to glue NAs to the SCA compliant APIs. Though the propinquity of the Nucleus concept and JTRS SCA is visible in many ways, complete SCA compliancy must be guaranteed.

VI. CONCLUSIONS AND OUTLOOK

A novel concept for developing tool assisted, portable and efficient WFs for SDRs targeting a WDL has been proposed in this paper. By identifying the common, processing intensive, algorithmic kernels a Nuclei library is built. WF is described in a WDL using the Nucleus library. The optimal granularity of the kernels enable efficient implementations and reuse. The standardization of the library will lead to the availability of Flavors from vendors. This provides algorithmic and implementation trade-offs even in a fixed HW platform. The implementation related properties is propagated to the Nucleus-API. Since the API has information for ideal mapping, efficient and tool assisted mapping can be provided by the WDE. Key challenges that exist in envisioning the concept were also presented in this paper.

Future work will address these challenges by working closely in algorithm, HW and tools domain. The selected application scenario for approaching and validating the Nucleus concept is an iterative and flexible MIMO-OFDM transceiver focusing on PHY layer, MAC layer and system cognition. A part of the transceiver will be implemented as a proof-of-concept. Investigations will also be done to find the properties that are needed for enabling (semi) automatic tool assisted WF development.

REFERENCES

- [1] E. M. Witte, T. Kempf, V. Ramakrishnan, G. Ascheid, M. Adrat and M. Antweiler, "SDR Baseband Processing Portability: A Case Study," in *5th Karlsruhe Workshop on Software Radios (WSR 2008)*, Germany, 2008.
- [2] "JTRS," May 2009. [Online]. Available: <http://sca.jpeojtrs.mil/>
- [3] T. Kempf, E. M. Witte, V. Ramakrishnan, G. Ascheid, M. Adrat and M. Antweiler, "A practical view on SDR baseband processing portability," in *2008 SDR Technical Conference and product Exposition (SDR 08)*, Washington D.C, USA, 2008.
- [4] T. Langguth and H. Schober, "SDR based Waveform Development," in *5th Karlsruhe Workshop on Software Radios (WSR 2008)*, Germany, 2008.
- [5] J. Glossner, M. Moudgill, D. Iancu, G. Nacer, S. Jintukar, S. Stanley, M. Samori, T. Raja and M. Schulte, "The Sandbridge Sandblaster Convergence Platform," May 2009. [Online]. Available: http://www.sandbridgetech.com/documents/sandbridge_white_paper_2005.pdf
- [6] H-M. Bluethgen, C. Grassmann, W. Raab, U. Ramacher and J. Hausner, "A programming baseband platform for software defined radio," in *2004 SDR Technical Conference and product Exposition (SDR 04)*, Phoenix, USA, 2004.
- [7] Wimax TI Library, "<http://www.ti.com/corp/docs/landing/wimax/index.htm>," May 2009.
- [8] L. Pucker, "Component-based development of radio systems and subsystems: are we there yet?" *IEEE Communications Magazine*, vol. 45, pp. 44-46, 2006.
- [9] Lyrtech, "Developing a GSM Modem on a DSP/FPGA Architecture," May 2009. [Online]. Available: www.lyrtech.com/publications/Article_GSM_Modem_DSP.pdf
- [10] E. D. Willink, "The waveform description language: moving from implementation to specification," in *Proc. Military Communications Conference (MILCOM 2001)*, vol. 1, 2001, pp. 208-212.
- [11] M. S. Gudaitis and R. D. Hinman, "A waveform description language for software defined radio," in *2002 SDR Technical Conference and Product Exposition (SDR 02)*, 2002.
- [12] J. Chapin, V. Lum and S. Muir, "Experiences implementing GSM in RDL (the Vanu Radio Description Language)," in *Proc. Military Communications Conference (MILCOM 2001)*, vol. 1, 2001, pp. 213-217.
- [13] C. Moy, J. Palicot, V. Rodriguez and D. Giri, "Optimal Determination of Common Operators for Multi-Standard Software-Defined Radio," in *4th Karlsruhe Workshop on Software Radios (WSR 2006)*, Germany, 2006.
- [14] K. Fan, M. Kudlury, G. Dasika, S. Mahlke, "Bridging the computation gap between programmable processors and hardwired accelerators," in *Proc. IEEE 15th International Symposium on High Performance Computer Architecture*, 2009, pp. 313-322.
- [15] H. Meyr, "Re-configurable ASIPs : Is there any need for these architectures?" in *7th International Forum on Application-Specific Multi-Processor SoC*, 2007.
- [16] "The Landscape of Parallel Computing Research: A View From Berkeley," May 2009. [Online]. Available: <http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-183.pdf>
- [17] "FFTW," May 2009. [Online]. Available: <http://www.fftw.org/>
- [18] J. Chen, B. Xu and T. Huang, "A novel robust feature of speech signal based on the Mellintransform for speaker-independent speech recognition," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 2, 1998, pp. 629-632.
- [19] E. W. Hansen, "Fast Hankel Transform Algorithm," in *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 33, 1985, pp. 666-671.
- [20] L. Knockaert, "Fast Hankel Transform by Fast Sine and Cosine Transforms: The Mellin Connection," in *IEEE Transactions on Signal Processing*, vol. 48, 2000, pp. 1695-1701.
- [21] H. Sava, M. Fleury, A.C. Downton and A.F.Clark, "Parallel pipeline implementation of wavelet transforms," in *Proc. IEEE Vision, Image and Signal Processing*, vol. 144, 1997, pp. 355-360.
- [22] E. Tell, O. Seger and D. Liu, "A converged hardware solution for FFT, DCT and Walsh transform," in *Seventh International Symposium on Signal Processing and Its Applications*, vol. 1, 2003, pp. 609-612.
- [23] K. Kloker and B. Undsley, "Efficient FFT Implementation On An Floating-Point Digital Signal Processor," in *Proc. International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, 2002, pp. 1302-1305.
- [24] Texas Instruments, "TMS320C64x DSP Library Programmers Reference (Rev. B)," October 2003.
- [25] Xilinx, "IP Release Notes Guide (XTP025 (v1.4))," September 2008.
- [26] Altera, "FFT MegaCore Function MegaCore User Guide (Version: 8.0)," May 2008.
- [27] M. S. Gudaitis and R. D. Hinman, "Practical considerations for a waveform development environment," in *Proc. Military Communications Conference (MILCOM 2001)*, vol. 1, 2001, pp. 190-194.