IPSJ Transactions on System LSI Design Methodology Vol.9 61–71 (Aug. 2016)

[DOI: 10.2197/ipsjtsldm.9.61]

Regular Paper

Unified HW/SW Co-Verification Methodology for High Throughput Wireless Communication System

Nana Sutisna[1,a)] Reina Hongyo[1] Leonardo Lanante Jr.[2] Yuhei Nagao[2]
Masayuki Kurosaki[2] Hiroshi Ochi[2]

Received: December 4, 2015, Revised: March 15, 2016,
Accepted: May 9, 2016

Abstract: As complexity of system LSI design is increased significantly, efficient verification methodology is mandatory to achieve reliable system and to speed up development time. HW/SW co-verification, nowadays, is interesting and practical as a tool for system verification because it allows covering large number of verification scenarios in acceptable time. In this paper, we present an efficient and unified framework of HW/SW co-verification methodology for large scale system, particularly high throughput wireless communication system. The proposed methodology combine system level simulation (e.g., MATLAB or C/C++) and physical level verification (e.g., FPGA). It allows performing fast HW/SW verification, as well as fast turn-around design exploration. The proposed methodology has been successfully employed to our case study which is 4x4 MIMO wireless communication system. Experimental results show that our case study is able to run in near real-time processing, resulting in an improvement of simulation time orders of magnitude faster than software based simulation. Moreover, the proposed verification platform can be used for complete characterization of communication performance of a MIMO wireless system employing MLD MIMO decoder for various operation modes and channel models.

Keywords: Unified HW/SW co-verification, Hardware-in-the Loop, FPGA prototyping, MLD MIMO decoder

1. Introduction

Recently, with tight demand of time-to-market for product deployment, fast verification time has become a main hurdle to guarantee a reliable product, especially in developing a complex system that employs various operational modes and system parameters [1], [2]. In designing an LSI system, the verification process takes almost 80% of overall development time. The system development in wireless communication system field is one of good example of such system. For the last two decades, wireless communication technology has evolved in fast and continuous progression. Wireless system standard always changes, in order to meet high throughput and high reliability requirements. Unfortunately, every introduction of new standards, the complex and advance signal processing are adopted, as well as to support various system features. Consequently, the system complexity will increase significantly. For example, in the latest wireless LAN standard 802.11ac [3], very high throughput wireless communication system including Downlink Multi User MIMO and higher order modulation scheme up to 256-QAM is adopted.

Verifying the system functionality in all possible condition need huge amount of relevant test cases, in order to achieve a confidence level of acceptance test criteria. Furthermore, in development process we also must validate that algorithm transformation from floating point to fixed-point as well as data path bit-width optimization do not degrade the overall system performance within tolerable margin. Hence, verification time is relevant issue in a complex system. To overcome these problems, recently, hardware-based verification has attracted attention since it allows one to cover a large number of test scenarios in a shorter time compared to software based verification.

References [4] and [5] have described verification systems intended for wireless communication system featuring hardware-in-the loop. However, these verification platforms do not cover all requirements of a verification platform. In Ref. [4], Liang et al. have proposed a hardware in-the-loop verification methodology for complex functions. While it is combined with system level simulation and has the benefit of a hardware-in-the loop system, the verification system is not flexible. It is applicable only in MATLAB/simulink environment. Hence, for another design that does not support this environment, it needs more effort to realize hardware-in-the loop verification system. Moreover, to be used as a verification platform, this paper did not clearly describe its verification time improvement. On the other hand, while Ref. [5] emphasize the issue of verification time improvement, however, it is not tightly coupled with system level simulation and also do not describe performance evaluation regarding system functionality. Hence, those platforms cannot be easily used as a comprehensive evaluation of wireless communication system.

In this paper, a novel unified framework of hardware (HW) /software (SW) co-verification methodology for large scale sys-

[1] Graduate School of Computer Science and Electronics, Kyushu Institute of Technology, Fukuoka 820–8052, Japan
[2] Department of Computer Science and Electronics, Kyushu Institute of Technology, Fukuoka 820–8052, Japan
[a)] nana@dsp.cse.kyutech.ac.jp

tem, specifically wireless communication system is addressed. The proposed methodology is applied in our case study using HAPS system [13] co-operated with system level simulation such as MATLAB and C/C++. The proposed methodology and our applied platform have several advantages, which are:

(1) The verification system has flexible and scalable interface for data transfer between software part (host PC) and hardware part (FPGA). This feature enables design extension and could be applied to different system with only minor modifications.
(2) Proposed verification methodology is co-operated with system level simulation such as MATLAB, C/C++, etc and physical level verification. It allows for unified evaluation of various level of system design.
(3) The unified framework allows for fast design exploration and verification which is specially useful in verification of large scale system.

The rest of this paper is organized as follows. Section 2 describes an overview of related work of hardware-based verification platform and its methodology. In Section 3, we describe our proposed methodology for building HW/SW co-verification platform. In Section 4, an application example of proposed HW/SW co-verification system in MLD MIMO decoder is presented. In Section 5, we discuss implementation results and example performance evaluation of the MIMO Decoder. Section 6 describes the effectiveness of our proposed methodology by providing comparison as well as achievable verification speed-up of proposed methodology. Finally, in Section 7 some conclusions are drawn.

2. Related Works

There are several considerable works that have shown capability of verification environment for MIMO wireless communication system. In Refs. [6] and [7], the authors presented a complete wireless LAN system and performed related performance evaluation. For block component, the authors in Refs. [8], [9] proposed FPGA prototyping for MIMO decoder. The prototyping of complete system is very valuable to demonstrate technology capability under realistic condition such as analog-impairments and effect of various channel condition. However, its main purpose is as the final outcome rather than integral part of product development. Additionally, in these papers, the issue of verification time in large coverage test scenarios is not presented. Therefore, it cannot be used in early stage of development such as during block component design.

Other works are the FPGA prototyping for accelerating verification, as discussed in Refs. [5], [11], [12]. In these papers, the issue of verification time is mainly discussed. The verification technique to improve simulation time is presented. For example in Ref. [11], to address bottleneck on data communication between the host PC and the hardware target, the TCP/IP based communication is presented. In Ref. [5], PCI based connection is employed with network infrastructure for building an FPGA-Accelerated testbed. Furthermore, the prototyping using another hardware target such as GPU is presented in Ref. [12]. However, while those papers are dedicated for communication system, evaluation of full system performance metric such as Bit Error Rate (BER) performance was not covered.

In summary, there are some missing metrics that not discussed as a comprehensive verification platform for wireless communication system. These includes flexibility to be employed with various function block, efficiency of verification (verification time improvement), and integrability with system level simulation. As a result, those type of verification/prototyping platform cannot be used as integral part of system development process, especially in the early stage of development.

To address limitations of previous works, the proposed verification platform covers verification process for all verification stage, from block component up to full system level simulation, including algorithm verification, RTL verification, and real-time HW/SW co-verification. Thus, the proposed methodology closes the loop of design, exploration, optimization, and testing. This approach can avoid time consuming, prone error, and multiple iterative design spin-off process from one group in a big research and development team. Hence, reliable design and fast time-to-market of large scale system can be achieved.

3. Unified Framework of HW/SW Co-Verification Methodology

3.1 Scope of the Framework

An efficient hardware-software co-verification platform should not only be capable of performing a fast simulation, but also at the same time it must have: 1) flexibility to support quick turn around design modification and design extension; 2) cover all verification stage, from algorithm development to hardware implementation; 3) tightly integrated with system level simulation to maintain reliable performance. In order to realize such efficient co-verification platform, in this paper we propose an effective approach to obtain reliable and efficient development of large scale systems.

In general, VLSI design and verification flow consist of three design layers, as depicted in **Fig. 1**. In the first layer, a complete system modeling is developed using system level language such as MATLAB, C/C++, etc. For example, in wireless communication system, it includes transmitter, channel model, and receiver. Because blocks in wireless communication system, especially in receiver, are very sensitive to employed algorithm and hardware optimization, before translating into hardware design, the designed algorithm should be verified in order to get realistic hardware complexity and predictable performance. After algorithm have been validated, RTL design for hardware implementation can be generated depending on hardware target. The process of algorithm transformation, recently, not become a difficult task since the availability of advance High Level Synthesis CAD tools that support model based-RTL design [10]. It allows fast design exploration and lead to reduce development time significantly. In RTL design stage, the verification is also performed to verify that hardware design in HDL code is still have same functionality as defined in system level simulation. However, verification time for bit true model in this CAD environment is very slow. The final stage of design is physical implementation. Once the RTL is obtained, hardware implementation can be carried out and once again the verification is carried out to ensure that final hardware implementation satisfy required performance.

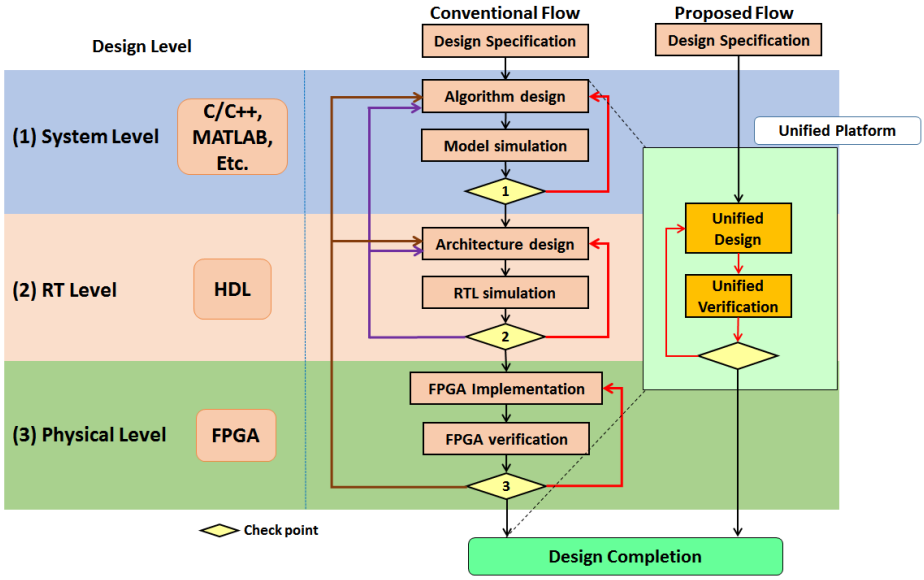


Fig. 1 VLSI Design and verification flow.

In the conventional verification system, the verification process of each stage is carried out independently and is also not integrated to system level simulation. Additionally, to verify and to evaluate overall system performance it needs to implement all blocks into hardware emulation in order to obtain fast verification results. With this approach, all hardware design of overall system should be completed before performing verification. Moreover, another potential problem will be faced when we directly implement a full system, such as lack of FPGA resource or timing problem. Therefore, there are several problems from conventional verification methodology. The first, conventional verification process may contain many iteration loop, either within same design layer or different design layer. This process take longer verification time and slow feedback for design modification. As a result, development process takes longer time. The second one, because the verification process is independent between design layer, the verification environment in each layer cannot guarantee the consistency of performance in the point of view of system level simulation. Hence, the expected performance cannot be maintained from system level design into final system implementation.

In our proposed verification platform, the verification of complex system can be carried out efficiently from block component up to system level, employing unified HW/SW co-verification platform. To realize such system, tight integration of hardware platform into system level simulation is a key element. The proposed verification platform can be used by hardware designers to design, implement, and verify related block concurrently. The verification of each block can be performed in the point of view of system level simulation. Hence, the final performance requirements of full system can be maintained and predictable. Moreover, the verification time can be significantly reduced.

3.2 Task Partition Methodology

Typically, the design process of a complex signal processing system starting from system level algorithm description, such as MATLAB or C/C++. There are many various possible algorithm implementation to fulfill system requirements, but they give different trade-off between area complexity, efficiency, flexibility, and design effort. Hence, design exploration is mandatory and should be performed quickly at the initial development. Once the algorithm is selected, a submodule can be transferred to hardware development and further verified in the point of view of system level simulation.

The first step to build an efficient HW/SW co-verification system is performing task partitioning of all system process. The task partitioning can be carried out under consideration of area complexity, timing processing, requirement of quick algorithm evaluation, or any design metrics that are determined by system requirements.

For example, as depicted in **Fig. 2**, signal processing block of wireless communication system consist of several consecutive processes: input data and parameter (referred as Test Vector), Transmitter, Channel Model, and Receiver. Assumed that we perform task partitioning to the system by selecting a complex process (e.g., Receiver process) to be simulated in hardware platform, and the rest of processes are simulated in software platform. Furthermore, the Transmitter, Channel model, and output results checking could be implemented into different software abstraction language, such as transmitter and channel model process is implemented in MATLAB, while output checking is implemented using C/C++. Both software platform, which are MATLAB part and C/C++ part, communicate each other through our developed custom transparent layer communication (API). On the other hand, receiver process, as considered the high complexity system, is implemented in hardware emulated platform. At the initial stage design, it is possible to implement receiver partially in hardware and keeping other receiver processes in software. As the development stage is growth, more tasks in software processing could be added up to the hardware target, achieving a complete full hardware emulation.

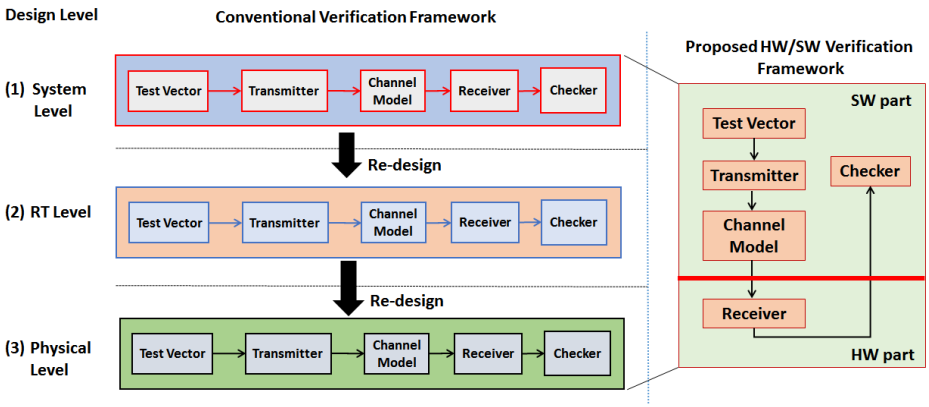


Fig. 2 Example of verification framework.

3.3 Unified HW/SW Design

To implement a complete design of HW/SW co-verification, first we have to provide a generic architecture for HW/SW implementation, as shown in **Fig. 3**. The HW/SW design should have support for flexibility and reconfigurability purpose. Hence, it could be reusable for other design targets or applications. The hardware design mainly consist of three building blocks, which are: (1) bus interface module for receiving data from end-point physical connection link (e.g., PCIe cable), (2) memory banks for storing input and output stream, and (3) hardware target that is being verified (Design Under Test). On the other hand, the software design consist of 3 main blocks, which are: (1) system level design that performs system level simulation, (2) communication API that handles data communication between software layer, and (3) bus driver that connects data communication of software part and hardware part.

The employed HW/SW platform uses HAPS board from Synopsys [13] which basic purpose is for design prototyping platform. On the other hand, the main objective of our proposed methodology is for unified verification covering all design abstraction layers. Our proposed methodology seems similar with Synopsys Hybrid Prototyping Platform [15] that can also perform system level simulation by utilizing virtual prototyping. However, our hardware platform does not include virtual prototyping packages. Therefore, system level simulation could not be carried out in employed HAPS board. Furthermore, the TLM verification flow in virtual prototyping is being a commercial package which is not an open access package. Additionally, the TLM based verification concept primarily suitable for System on Chip (SoC) prototyping case, where the data communication through on-chip bus among various modules are very important.

In order to address the limitation of basic HAPS platform, we propose flexible and scalable interface both in hardware and software part to realize unified HW/SW co-verification. In hardware side, the flexible interface is employed to handle various transfer modes, for example stream based mode or pass-through mode. Additionally, to support various CAPIMs in different applications, we use configurable architecture for I/O management. In software side, to adapt with high level system simulator we employ customized API, as communication interface to manage data that are provided by high level simulator or hardware circuit. The

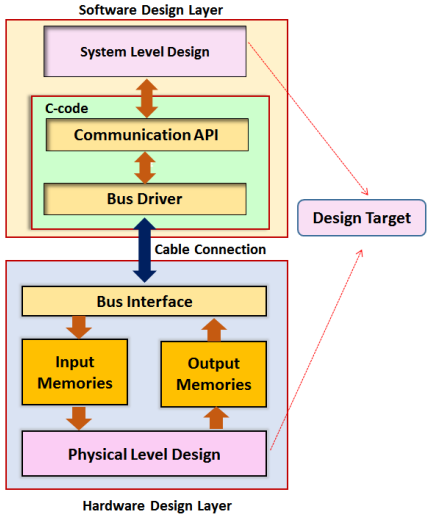


Fig. 3 HW/SW Architecture design.

API specifically has to allocate the data that will be transferred to hardware from host PC, and also receive and collect the data that are received from hardware for further processing in system level simulation.

Moreover, we also provide a key feature in proposed verification method which is called as data-driven simulation that allows the data processing could be performed in vector based (burst data). Vector based processing is considered since the data from system level simulation (e.g., MATLAB) inherently in form of matrix or array data. To realize this feature, we provide software interface that support block data transfer (burst mode) and also hardware interface that able to handle block data transfer within block RAMs or FIFOs. By utilizing this approach, the interaction of hardware software only occur in the beginning and the end of verification run-time. Therefore, the overhead of HW/SW interaction can be reduced significantly, which results significant improvement of HW/SW co-verification run-time.

4. Application Example: MLD MIMO Decoder in 4x4 MIMO Communication System

MIMO decoder play an important role in MIMO wireless communication system because the BER performance is highly

dependent on employed MIMO decoder algorithm. Many researchers have investigated several techniques that are feasible for practical implementation. The Maximum Likelihood Detection (MLD) technique is considered as the optimal technique for MIMO Decoder. However, due to increasing constellation point of the modulation and the number of spatial stream, the computation complexity in MLD MIMO decoder become extremely high and consequently increasing hardware complexity and thus the verification effort. Hence, some approaches to reduce complexity of MLD MIMO decoder computation are studied and also efficient verification approaches are developed.

In this paper, to demonstrate applicability of proposed verification platform, we consider MLD MIMO Decoder for IEEE 802.11ac WLAN system, as a case study. The designed MLD MIMO decoder should support up to 256-QAM modulation scheme. To the best our knowledge, until now there is no paper that describes VLSI implementation of full MLD MIMO decoder design for high order MIMO system up to 256-QAM. In [8] the authors propose FPGA Implementation of real time MLD MIMO decoder that support for QPSK modulation in 4x4 MIMO system. In higher modulation order, e.g., 64-QAM MIMO system, Ref. [9] described FPGA prototyping of quasi MLD MIMO Decoder.

Because of its huge complexity, design a real-time implementation and make efficient verification is still a big challenge. For example, with a carefully calculation of timing processing of highest complexity MLD parameter in 4x4 MIMO (256-QAM), verification of one packet data consisting 2 OFDM symbol, software based verification using MATLAB tools take around 70 days, while verification using assisted hardware just take 4 minutes. Obviously, the conventional approach takes much longer time and gives a slow feedback for designer. Consequently, resulting high cost development. Hence, design validation for all parameter cases using conventional approaches are not longer efficient. As an alternative, to accelerate the verification process hardware-assisted platform is proposed.

4.1 HW/SW Co-verification Platform Description

The hardware platform in this work uses HAPS (High-performance ASIC Prototyping Systems) provided from Synopsys [13], as depicted in **Fig. 4**. The HAPS system can occupy up to 7.5M gates for each FPGA chip. The HAPS system is connected to a host computer through UMRBus (Universal Multi Resource Bus) interface using PCI-e cable link [14].

The HW/SW co-verification system includes a host PC, software library, an FPGA system, template synthesizable HDL code of verification infrastructure and a dedicated hardware as design under test. The host PC is used for executing system level simulation such as MATLAB and C/C++ and it is connected to the hardware platform through UMRBus. The software API library provide communication interface between MATLAB, C/C++ program, and bus interface in FPGA system.

4.2 MIMO System Model and MLD Algorithm

We consider a MIMO wireless communication system with N Transmit Antenna and N Receive Antenna, as shown in **Fig. 5**.

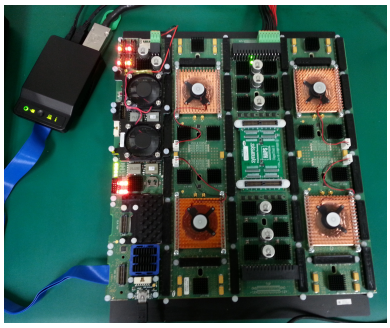


Fig. 4 HAPS hardware platform.

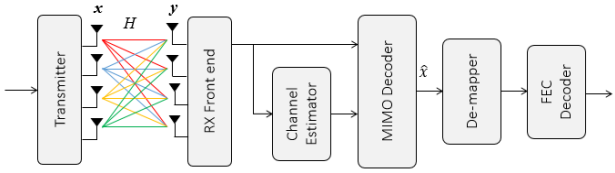


Fig. 5 MIMO communication system model.

Table 1 System parameter.

Parameter	Value
Wireless System Standard	IEEE 802.11ac
Number of Antenna (N)	4
Modulation Type (M)	QPSK (1), QAM16 (2), QAM64 (3) and QAM256 (4)
System Bandwidth	80 MHz

We also assume that the transmit symbol is taken from a quadrature amplitude modulation (QAM) which has 2^M constellation points where M is modulation order. The transmission of each vector **x** over flat-fading MIMO channels can be written as

$$y = Hx + n \tag{1}$$

where $x = [x_1, x_2, ..., x_N]^T$ is the transmitted signal vector, $y = [y_1, y_2, ..., y_N]^T$ is the received signal vector, **H** is the N x N channel matrix, and $n = [n_1, n_2, ..., n_N]^T$ is independent identically distributed Gaussian white noise vector.

To reduce computation complexity, we employ QR decomposition into channel matrix, **H**, that are provided by Channel Estimator block. Firstly, we decompose the matrix **H** into two matrices **Q** and **R**, where **Q** is the unitary matrix and **R** is the upper triangular matrix. With **H = QR**, Eq. (1) can be written as

$$z = Rx + n' \tag{2}$$

where $\boldsymbol{z} = \boldsymbol{Q^H} y$ and $\boldsymbol{n'} = \boldsymbol{Q^H} n$.

Then, the output MLD, $\hat{\mathbf{x}}$, can be calculated by searching among all candidate such that resulting minimum magnitude of error signals, as provided by following equation.

$$\hat{x_E} = \arg\min \sum_{i=1}^{\Omega} |z - Rx|^2 \tag{3}$$

where Ω is number of all candidate which is 2^{2MN} and x_E referred as Euclidean distance calculation. In this paper, we consider MIMO system with parameters as provided in **Table 1**.

4.3 Architecture Design

From Eq. (3), we can derive the MLD signal processing architecture as shown in **Fig. 6**. The MLD MIMO Decoder block

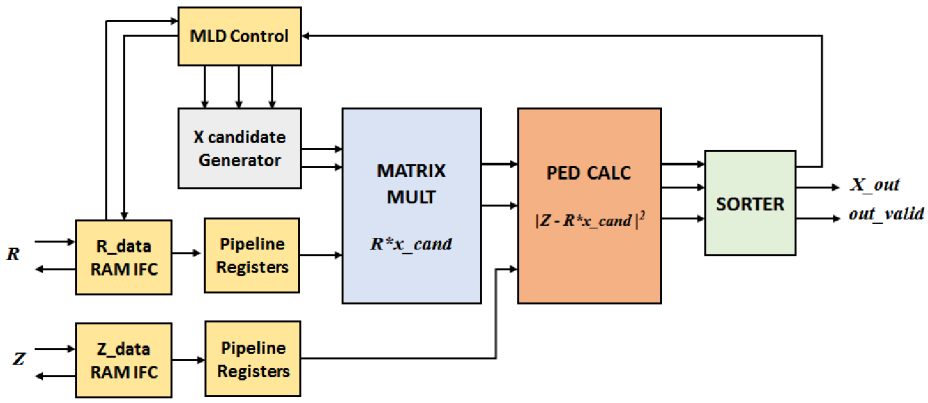


Fig. 6 MLD MIMO decoder architecture.

mainly consist of **Candidate Generator**, that provides all possible transmitted signals, **MATRIX MULT** block, that performs matrix multiplication between received signals and candidate of transmitted signal, **PED CALC**, that performs distance calculation, recursively along all possible number of candidate, **SORTER** block that determines the minimum error distance and selects the estimated transmitted signal. Other blocks are **MLD Control**, for providing signal control and parameter in appropriate timing and some **Pipeline Registers**.

The implemented structure of MLD processing in Fig. 6 refers to Euclidean distance. However, to reduce of multiplier, in practical approach we may employ simplified distance calculation by using a Manhattan distance, as given by:

$$\hat{x_M} = \arg\min \sum_{i=1}^{\Omega} |Re[z - Rx]| + |Im[z - Rx]| \qquad (4)$$

Since blocks in wireless communication system, especially in receiver, are sensitive to system performance, selected algorithm should be quickly analyzed in term of system level performance and hardware cost. The proposed verification platform allows fast design exploration and co-simulation for functional verification. Once an algorithm have been validated, RTL design for hardware implementation could be generated to be integrated with whole FPGA architecture for complete HW/SW co-verification.

4.4 Process Mapping and Template Architecture

Before implementing the design target in the hardware and software, we have to perform task system partition and mapping each block of complete wireless communication system into software part and hardware part. We use the same approach as described in Section 3.

Figure 7 shows partitioning and mapping each module of complete wireless communication system in generic architecture of software and hardware. The software part implement the transmitter process, channel model, a part of receiver process, and performance evaluation. On the other hand, the hardware part only implement MIMO Decoder block. Beside the main blocks, we also introduce a software communication interface, which are API software and bus driver software as well as the bus interface module in hardware.

Integration of software part and hardware part realizing the

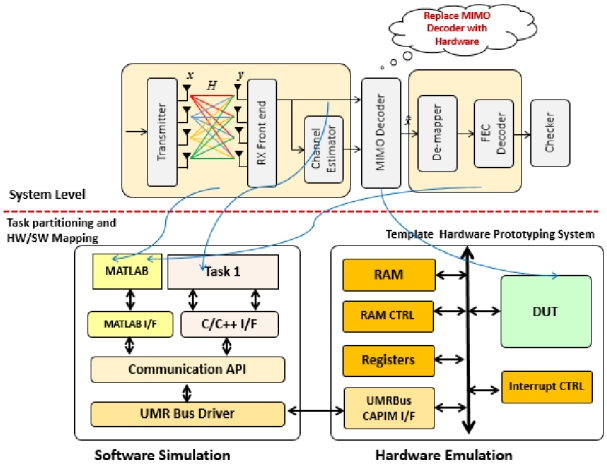


Fig. 7 HW/SW partition and mapping for MIMO Decoder verification.

simulation platform that able to perform HW/SW co-simulation of wireless communication system. The system level simulation will provide data for the target hardware. First, it delivers the data to the API module to provide appropriate data format as required by Bus driver. Then, the software test bench in the host PC send the data stream and the control data to the specified memory buffer in hardware platform. Finally, HW/SW co-verification of MLD MIMO Decoder in the point of view of system level simulation, such as BER performance, can be carried out. Furthermore, this proposed methodology and platform can accommodate for fast turn around design changing and iteration, when performance results do not satisfy the requirements.

4.5 Hardware Design

In hardware side, the data transaction will be carried out by bus interface module that connect the FPGA hardware with UMRBus end point. This module is called as CAPIM (Client Application Peripheral Interface Module) and its timing follow the UMRBus protocol. Each CAPIM unit could be connected to specific memory, register, or any instance module inside FPGPA hardware. The CAPIM units are accessible from host PC through the UMRBus driver by giving index number in every data transfer.

In order to implement MLD MIMO Decoder into template FPGA architecture, first we consider that our design example have 14 inputs, which are R data matrices and Z data matrices. Additionally, we also will reserve one CAPIM to handle data in-

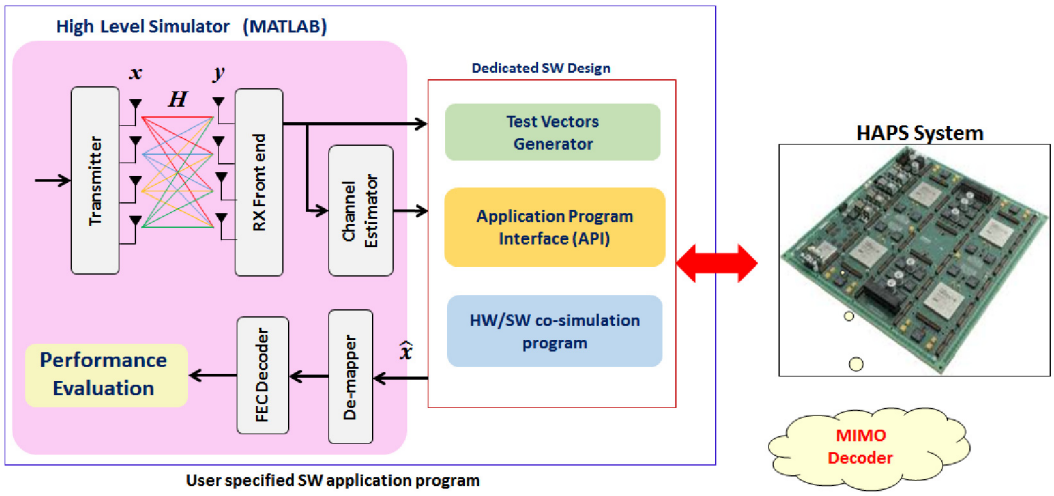


Fig. 8 Hardware-in-the loop co-verification system.

put for system parameter and control, such as modulation type, number of stream and start/enable signals. According to the requirements of MLD MIMO Decoder timing specification, the input of MLD MIMO Decoder for each decoding process should be available at the same clock cycle. For simplification, we will store each element of matrices input into different memory. Hence, we map each input data R and Z to one index CAPIM. In the same approach, we can assign each data output to one CAPIM unit. In case of MLD MIMO decoder, we need 14 CAPIM units for input buffer, 4 CAPIM units for output and 1 CAPIM unit for register control. Moreover, the way of CAPIM structure is easily modified and configured for other DUT. Hence, the effort of configuring the interface design between FPGA and host PC for other applications is relative small. To maintain a communication between host PC and FPGA system, the interrupt signal is also generated and it is sent through bus to be monitored by software application.

4.6 Software Design

The software design mainly consist of three layers which are:

(1) **System level simulation** of complete wireless communication system in MATLAB. This simulation performs all data processing of wireless communication system as well as performance evaluations, except the MIMO Decoding processing that is implemented in hardware.

(2) **API program** that handle communication data between system level simulation in MATLAB and main test bench program. The API perform a specific data processing of array data from MATLAB simulation resulted in process before MIMO decoding block. The custom API program is developed with emphasize on flexibility and reconfigurability for design extension. Hence, It could build a smooth data transaction between software layer and further make convenient verification flow.

(3) **Bus driver package** that connect the host PC and the FPGA platform. The bus driver mainly performs read and write process to specified CAPIM target, as well as monitor the interrupt signal from hardware target. The bus driver can transfer data either in burst sequence or single transfer depend on the verification requirements. It also can point out the CAPIM target by assigning the CAPIM address number.

All software package is wrapped in unified test bench simulation that performs simulation (co-verification) process for the design target. The test bench software carry out verification process including hardware controlling, data loading, and retrieving data for post processing analysis. The way of data flow in test bench software also reflect the behavior of data flow in full system level. Hence, the verification sequence process (task sequence) can be transfer to other development team for further verification process or integration to full system level.

5. Experimental Result and Performance Evaluation

5.1 FPGA Implementation Results

First, we show the implementation results of the proposed MIMO Decoder design in **Table 2**. The MIMO Decoder design occupy 18,024 look-up tables (LUTs), 7989 registers, 576 blocks of RAM, and can achieve clock frequency up 180 MHz.

However, for the simplification the running clock frequency is set to 100 MHz as the internal FPGA oscillator source. The logic utilization of hardware target is very small compared to total capacity of FPGA, because this results is for implementation of single processing element of distance calculation. Hence, it is possible to implement a number of parallel processing elements to speed up hardware processing time because our hardware platform have big capacity of logic resources. The optimum number of processing element can be determined by performing design exploration, which will be described later in section 5.3. It should be noticed that implementing parallel processing element does not change the design of bus interface and memory structure. Hence, we can utilize the rest of the LUT resource, as well as the DSP block for implementing the parallel architecture.

5.2 Performance Evaluation of MIMO Decoder

In this section, we provide an example of performance evaluation that can be carried out by proposed verification platform.

Table 2 FPGA logic utilizations.

Resource	Logic Utilization	Available
LUTs	18,024	474,240
Registers	7,989	948,480
RAM blocks	576	720
Max Clock Frequency	182 MHz	-

Table 3 Verification condition.

Parameter	Values
System Model	IEEE.80211ac Model
Number of Antennas	4
Modulation Type	QPSK, QAM-16, QAM-64, and QAM-256
Channel Model	TGac Channel Type D (Indoor) and Gaussian Random
Data Packet	1,000 Bytes
SNR	0 - 30 dB

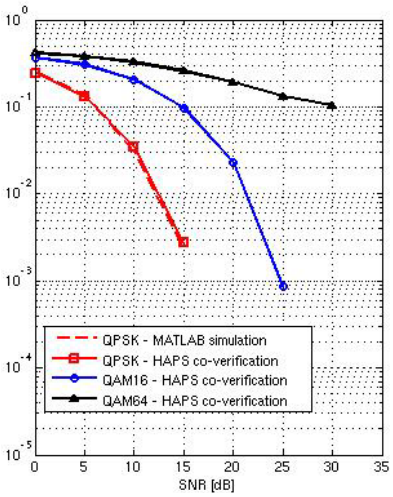


Fig. 9 Example performance evaluation: (1) BER performance on various modulation types.

The verification condition for performance evaluation is provided in **Table 3**.

The performance of proposed MIMO Decoder design has been evaluated regarding to various channel characteristics and modulation types, as shown in **Figs. 9** and **10** respectively. Figure 9 shows BER performance for various modulation types. We also provide MATLAB simulation result that performs floating point simulation of algorithm. We can see that the performance of hardware implementation of MIMO Decoder is close to MATLAB simulation. Hence, we can conclude that the algorithm transformation to hardware fixed point format is sufficient. Figure 10 shows the performance evaluation for various channel type. We have evaluated performance of MIMO Decoder in three types channel, which are : Identity channel, TGac AWGN, and TGac Channel D. From Figs. 9 and 10, we can prove the capability of our verification platform for comprehensive evaluation of wireless communication system employing MLD MIMO decoder as well as verification of system level design (MATLAB) and physical implementation (FPGA).

5.3 Evaluation of Design Exploration

As the hardware implementation of communication system is highly dependent to various constraints, design exploration is a key enabling to obtain optimum design. Some metrics such as

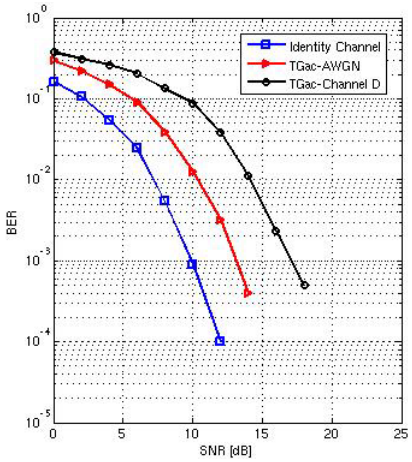


Fig. 10 Example performance evaluation: (2) BER performance on various channel types (QPSK Mode).

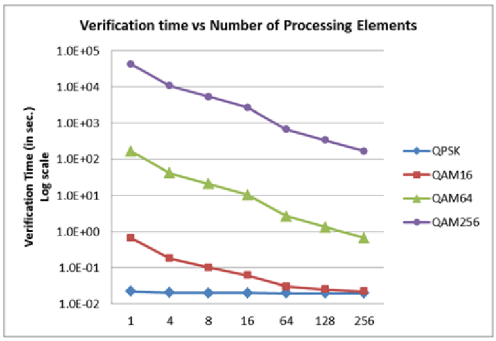


Fig. 11 Verification time for various PE numbers.

throughput, hardware complexity, as well as performance can be consider as input for performing design exploration subject to available resource area and acceptable development time (include verification time). In conventional verification approach, to evaluate the design exploration the designer should perform verification in different layer abstraction, which are algorithm evaluation in system level, RTL simulation and hardware verification independently. In unified framework, the verification can be performed simultaneously, which reduce total verification time.

In order to validate the effectiveness of proposed methodology for design exploration we provide an example of design exploration task in terms of efficiency of using parallel processing elements. In this task we will find out the optimum number of processing element that can be employed with subject to achievable verification time efficiency and the hardware cost. We consider this case since it directly affects the speed-up of verification time which is the most important objective of our proposed verification framework. For another cases of design exploration such as selecting optimum bit length subject to error performance and available hardware resource and also determining the optimum algorithm subject to error performance and logic resource practically could be carried out by using proposed framework.

To further speed up hardware processing, it is possible to change the design architecture. For example, the using of parallel architecture of MLD Processing Element can speed up hardware processing and finally resulting improvement of system throughput. In **Fig. 11** we show that increasing number processing el-

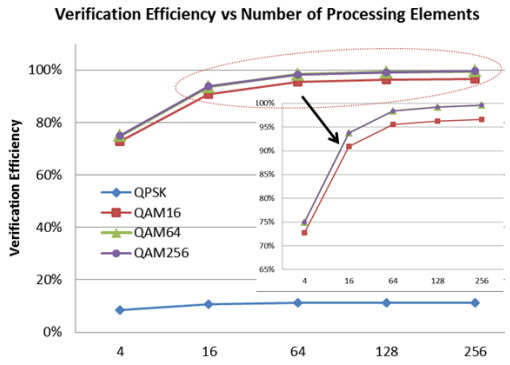


Fig. 12 Verification efficiency for various PE numbers.

ement up to 256 units, will give another increasing verification time about 100 times faster than using single processing element. This result use the same simulation parameter as provided in Table 3. The total speed up of verification time is up to 10^5 time compare to pure software verification.

However, increasing number of processing element no longer give high benefit for speed up verification time. As shown in **Fig. 12** the verification efficiency as the number of parallel processing elements area more than 64 units, the efficiency not increased significantly regarding to hardware cost. The verification efficiency, η is calculated based on Eq. (5):

$$\eta = 1 - \frac{T_{singlePE}}{T_{paraPE}} \tag{5}$$

where $T_{singlePE}$ is verification time carried out in single processing element architecture, and T_{paraPE} is verification time carried out in parallel processing element architecture.

Furthermore, from synthesis results it is found that the hardware cost for 64 PE units of MIMO MLD Decoder are 140,851 LUTs which is almost 8 times compare to single PE and 35,827 registers which is equivalent to 4.5 times of single PE usage, while the RAM usage occupies the same resource blocks which is 576 blocks. These results occupy around 30 % of available logic resource. Therefore, by trading-off the verification efficiency and hardware cost we can decide that the optimum parallel processing element is 64 units.

6. The Effectiveness of Proposed Methodology

In this section we present evaluation to clearly describe the effectiveness of proposed methodology with other simulation platforms. We also further provide analytical approach and some results to show the capability and the effectiveness of our HW/SW co-verification methodology, particularly in verification time improvement.

6.1 Methodology Comparison

Although high level simulation can provide all abstraction of system functionality and also easier for environment setup, however, the results are too far from real hardware performance. In order to address this limitation, RTL simulation is carried out to obtain more accurate evaluation. However, performing verification of complex circuit and extensive computation using RTL simulator is prohibited due to very long run-time simulation, particularly for exhaustive verification involving various system parameters and many different design versions. To accelerate design verification and achieve the optimum performance, the standalone FPGA prototyping is employed in Ref. [8]. Unfortunately, the standalone FPGA implementation needs huge effort since it must implement all system into hardware part. Hence, the verification task only could be performed at very late stage of design development. Furthermore, this simulation approach is also less flexible for various test scenarios as required in system level simulation.

Table 4 Comparison with other methods.

Objectives/Metrics	Synopsys Hybrid Prototyping [15]	HIL [4]	Proposed
High Level Simulation			
a) Time-driven (cycle-based)	✓	✓	✓
b) Data-driven (vector-based)	×	×	✓
RTL Simulation	×	✓	✓
FPGA simulation	✓	✓	✓
Unified HW/SW verification	✓	✓	✓
Verification speed	moderate	moderate	fast

Recently, the hardware-in-the-loop is a promising solution for fast verification is also has flexibility for support various system parameters, as proposed in our work and other HIL work [4]. However, there is significant difference between our proposed work and Ref. [4]. The HIL method in Ref. [4] employs Simulink environment that performs cycle-based simulation. Thus, the simulation is carried out as a time driven simulation and it will introduced large overhead due to HW/SW interaction in each cycle simulation. On the other hand, our proposed verification is performed in data-driven based simulation. To quantify the effectiveness of our proposed methodology, we summarize the comparison of several important features from different methodology/verification environment, as shown in **Table 4**.

6.2 Achievable Verification Time Speed-up

HIL methodology co-operated with Simulink can provide more convenience verification environment, however, the data processing is carried out in cycle-based simulation as well as time-driven based. This means the data processing is performed one-by-one data involving intensive interaction HW/SW communication. This method introduces significant overhead for HW/SW communication for each cycle and affect overall verification speed-up improvement, particularly in high complex circuits. The overhead software processing for each cycle, denoted by T_{SW_pre} and T_{SW_post}, are mainly due to its communication speed and also cycle penalty of interrupt (handshaking) handling.

On the other hand, our proposed verification use vector based verification. Since the transfer is also done in burst mode (for one vector data), this approach involves interaction between SW and HW in the beginning and ending of verification run time, denoted by $Prop\_T_{SW\_pre}$ and $Prop\_T_{SW\_post}$. Additionally, due to characteristic of data interface in our employed platform, the SW data transfer can be performed in high speed transfer, achieving up-to 800 Mbps, as compared to Ref. [4] that can only achieve several Mbps. Thus, the overhead for HW/SW interaction in the proposed method will be reduce significantly. **Figure 13** illustrates

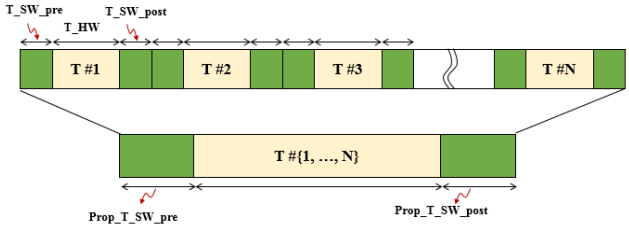


Fig. 13 HW-SW interaction in verification process: time-driven simulation (upper), data-driven simulation (lower).

Table 5 Estimation of verification time.

Mod Type	SW proc (sec.)	HW proc (sec.)	Total proc (sec.)
1 (QPSK)	0.02	0.00261	0.02261
2 (QAM16)	0.02	0.65541	0.67541
3 (QAM64)	0.02	167.77221	167.79221
4 (QAM256)	0.02	42,949.67301	42,949.69

the differences of time-driven simulation and data-driven simulation.

6.3 Estimation of HW/SW Verification Run-time

In order to obtain valid verification time, estimation of processing time is the most important factor. Therefore, we make an accurate approach by abstracting each computation tasks involving in HW/SW co-verification, as shown in **Fig. 8**.

Note that the verification time consist two main parts process, which are software processing and hardware processing. Furthermore, execution time of software processing has several task, which are initial setup and data transfer from host PC to FPGA target in beginning of simulation time, and interrupt handling, data transfer from FPGA target to Host PC, and post processing analysis for final performance evaluation following hardware process. The software execution time depend on the number of input and length of burst data. In our case, software execution time only varies with number of CAPIM and length of data that are being simulated. Variation of modulation type parameter does not affect software execution time.

On the other hand, the hardware processing mainly consist of iteration execution time of processing element (PE) which the number of iteration depend on employed modulation type. In our simulation case, to decode one subcarrier data, the PE will process iteratively as number of candidate. Each processing of subcarrier will also take additional cycle for memory access. The estimation of processing time for each modulation type is given in **Table 5**. In our design case, by performing software profiling, the software takes around 0.02 s for every run time. With the same approach, the estimation time methods for any target application can be estimated accurately.

6.4 Simulation Speed Comparison

The simulation speed of the different environment verification has been measured in order to present benefit of proposed verification platform. As shown in **Fig. 14**, the speed up of verification time is more than 3 times magnitude (1,000 times) compare to software simulation, such as Modelsim and MATLAB. It also can be seen that the simulation time is close as predicted by pre-calculation. Furthermore, from experimental simulation it is

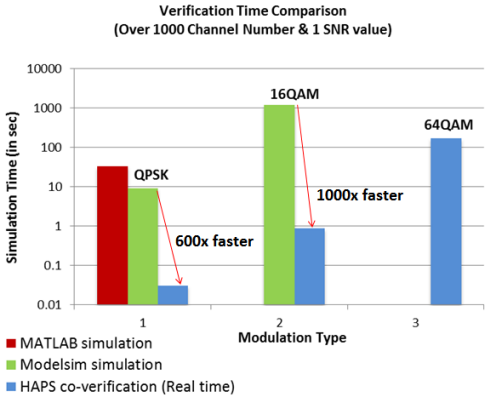


Fig. 14 Verification time comparison.

shown that the bottleneck of data transfer is no longer occurred. For instance, when evaluate transmitting large data (1,000 Byte) using high order modulation, the software processing for data transfer between host PC and HAPS system is negligible and the limitation lied in hardware processing. Because the software processing part took a constant time, while the hardware processing part depend on employed modulation type.

7. Conclusion

In this paper we have proposed a unified framework for HW/SW co-verification methodology for large scale system. The proposed HW/SW co-verification methodology have some features which include flexible and scalable platform for HW/SW co-verification, tightly integrated with system level simulation, allowing hardware-in-the loop verification, and also capable of running in near-real time performance. We have successfully demonstrated the effectiveness of proposed HW/SW co-verification methodology with the case study of MLD MIMO Decoder in 4x4 MIMO wireless communication system. The proposed HW/SW co-verification methodology can be used for complete characterization of MIMO decoder performance in the point of view of system level simulation, as well as perform fast design exploration, includes algorithm co-exploration, hardware efficiency assessment, and optimum bit length evaluation. Experimental results show that verification speed improvement can achieve up to 100,000 times faster compared with pure software based simulation. The calculated performance estimation can also predict verification efficiency of employed hardware architecture. Hence, we can choose optimum architecture of final hardware implementation.

References

[1] Huang, C.Y. et al.: SoC HW/SW Verification and Validation, *Asia and South Pacific Design Automation Conference* (*ASP-DAC*), pp.297–300 (2011).
[2] Teich, J.: Hardware/Software Codesign: The Past, The Present, and Predicting the Future, *Proc. IEEE*, Vol.100 Special Centennial Issue, pp.1411–1430 (2012).
[3] *Wireless LAN Medium Access Control* (*MAC*) *and Physical Layer* (*PHY*) *Specification*, IEEE 802.11ac/D5.0 (2013).
[4] Liang, G. et al.: A Hardware In The Loop Design Methodology for FPGA System and Its Application to Complex Functions, *International Symposium on VLSI Design, Automation, and Test* (*VLSI-DAT*), pp.1–4 (2012).
[5] Borlenghi, F. et al.: An FPGA-Accelerated Tesbed for Hardware Com-

ponent Development in MIMO Wireless Communication System, *International Conference on Embedded Computer Systems* (*SAMOS*), pp.278–285 (2012).
[6] Wenk, M. et al.: Hardware platform and implementation of a real-time multi-user MIMO-OFDM testbed, *IEEE International Symposium on Circuits and Systems* (*ISCAS*), pp.789–792 (2009).
[7] Ishihara et al.: Development and experimental validation of downlink multiuser MIMO-OFDM in gigabit wireless LAN systems, *EURASIP Journal on Advances in Signal Processing 2013*, 2013:123 (2013).
[8] Koike, T. et al.: Prototype Implementation of Real-Time ML Detector for Spatial Multiplexing Transmission, *IEICE Trans. Communication*, Vol.E89-B, No.3, pp.845–852 (Mar. 2006).
[9] Barbero, L.G. and Thompson, J.S.: Rapid Prototyping System for the Evaluation of MIMO Receive Algorithms, *The International Conference on Computer as a Tool* (*EUROCON*), pp.1779–1782 (2005).
[10] Cong, J. et al.: High-Level Synthesis for FPGAs: From Prototyping to Deployment, *IEEE Trans. Computer-Aided Design of Integrated Circuit and Systems*, Vol.30, No.4, pp.473–488 (2011).
[11] de Schryver, C. et al.: Lopy : An Open-source TCP/IP Rapid Prototyping and Validation Framework, *International Conference on Reconfigurable Computing and FPGAs* (*ReConFig*), pp.1–6 (2013).
[12] Wu, M. et al.: A GPU Implementation of Rea-Time MIMO Detector, *IEEE Workshop on Signal Processing Systems* (*SiPS*), pp.303–308 (2009).
[13] *HAPS-60 Series User Guide*, Synopsys (2013).
[14] *UMRBus Communication System Handbook v3.11*, Synopsys (2012).
[15] *Synopsys Hybrid Prototyping Solution*, available from ⟨http://www.synopsys.com/prototyping/fpgabasedprototyping/pages/hybrid-prototyping.aspx⟩ (accessed 2016-02.)

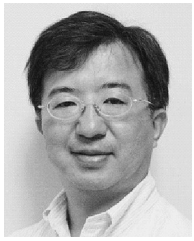
Nana Sutisna received his B.S. degree in Electrical Engineering and M.S. degree in Microelectronics, both from Bandung Institute of Technology, Indonesia, in 2005 and 2011 respectively. He is currently a Ph.D. student in Kyushu Institute of Technology, Japan. His research interests are VLSI Design, FPGA Implementation and Wireless System Design.

Reina Hongyo received her B.S. degree (2014) in Computer Science and Electronics from Kyushu Institute of Technology, Japan. She is currently a master student in Kyushu Institute of Technology, Japan. Her research interests are algorithm development and LSI design for MU-MIMO systems. She is a student member of IEICE.

Leonardo Lanante Jr. received his B.S. in Electronics and Communications Engineering degree and M.S. in Electrical Engineering both from University of the Philippines in 2005 and 2007 respectively. He received his Ph.D. degree in Information Systems in Kyushu Institute of Technology. He is currently an assistant professor in Kyushu Institute of Technology. His research interests include synchronization algorithms in wireless systems as well as signal processing in MIMO OFDM.

Yuhei Nagao received his M.E. (2006) and Ph.D. (2009) degrees from Kyushu Institute of Technology. Since 2009, he has been with Kyushu Institute of Technology as a researcher. His research interests include wireless communication. He is a member of the IEEE.

Masayuki Kurosaki received his B.E. (2000), M.E. (2002) and Ph.D. (2005) degrees from Tokyo Metropolitan University. He was with Kyushu Institute of Technology from 2005 to 2011 as an Assistant Professor. Since 2011, he has been with Kyushu Institute of Technology as an Associate Professor. His research interests include image processing and wireless communication for multimedia. He is a member of the IEEE.



Hiroshi Ochi received his B.S. and M.S. degree in electronics engineering from Nagaoka Institute of Technology, Japan in 1981 and 1984, respectively. He also received Ph.D. degree in electrical engineering from Tokyo Metropolitan University in 1991. He was with University of the Ryukyus from 1986 till 1999 as an assistant and an associate professor. He also receives MBA degree from Kyushu University in 2007. He is currently with Kyushu Institute of Technology as a professor in computer and electronics engineering department from 1999. His current research interests include signal processing for wireless communication system, VLSI chip design and MOT education. He also organizes a venture company Radrix co. ltd. as a CEO.

(Recommended by Associate Editor: *Hideho Arakida*)