DEVELOPMENT OF SOFTWARE DEFINED RADIO PLATFORM

Katsuhiko Tsunehara (Hitachi, Ltd., Kokubunji, Tokyo, Japan; tsune@crl.hitachi.co.jp)
Hirotake Ishii (Hitachi, Ltd., Kokubunji, Tokyo, Japan; ishii@crl.hitachi.co.jp)
Takashi Ishikawa (Hitachi, Ltd., Kokubunji, Tokyo, Japan; tiskw@crl.hitachi.co.jp)
Manabu Kawabe (Hitachi, Ltd., Kokubunji, Tokyo, Japan; kawabe-m@crl.hitachi.co.jp)

ABSTRACT

The platform development is important to realize a low-cost and high flexible software defined radio (SDR) equipment. In this paper, we propose a SDR platform and describe the prototype development of the SDR platform. The SDR platform consists of three components. The first is software library, that can be commonly used for signal processings of various radio communication systems, by having the distributed control function. The second is analog front-end that supports different radio parameters for various radio systems. The other is a versatile and scalable digital signal processing hardware. By developing a prototype of the SDR platform and the softwares that support to IEEE 802.11b and advanced PHS, it is confirmed that the SDR platform can adapt to various radio communication systems, and improve the efficiency of the SDR equipment development.

1. INTRODUCTION

In recent years, various radio communication systems, such as cellular, wireless LAN and Bluetooth, are used in the consumer communication field. And the standardization and practical use activities of mobile broadband systems, near range communication system and digital broadcasting are in progress, thus the number of available radio communication systems is continuously increasing. In addition, in the next generation mobile communication field, the hybrid system which consists of integrated various radio communication formals is expected [1].

On the other hand, depending on higher data rating and upgrading of radio communication standard, the functions of radio equipments, such as a base station and a terminal, are furthermore complicated. So it is necessary for radio equipment development to support short turn around time, following of standardization, bug fixing and so on.

Under the above mentioned situation, a software defined radio is well-examined [2][3]. Because of its function, it is necessary for a software defined radio equipment to have high flexibility. In general, high development cost and long development term are required to develop a hardware with high flexibility. So it is very important issue to create a platform in order to provide a low-cost software defined radio equipment with short turn around time [4][5].

In this paper, we propose a platform of software defined radio. In chapter 2, the structure and operation of the proposed platform, which consists of software library, analog front-end and digital signal processing part, are shown. In chapter 3, we represent a prototype of the platform. This prototype can support a part of physical and MAC layer signal processing of wireless LAN (IEEE 802.11b) and advanced PHS by installing the corresponding software. And in the last part, we describe a conclusion.

2. SOFTWARE DEFINED RADIO PLATFORM

Fig. 1 and Tab. 1 shows the block diagram of the proposed SDR platform and the description of the each components of the SDR platform respectively. In the following sections, the detail of each component is described.

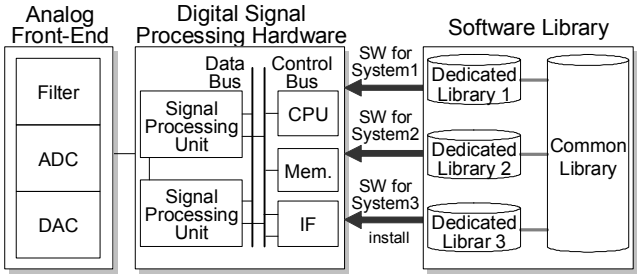

Fig. 1 Proposed SDR platform

Tab. 1 Description of the SDR platform components

Component	Description
Software Library	- Consist of common and dedicated libraries - Each library can be reused in softwares for various radio systems
Analog Front-End	- Support various radio parameters - Communize analog-digital interface
Digital Signal Processing HW	- Support various digital signal processings - Have scalability to support future systems

2.1. Software Library

2.1.1. Overview

In this section, the structure and operation of the software library are described. A software defined radio need to support various existing and new radio systems by changing the software in short turn-around time. So, it is not an efficient way to develop a software independently for each radio system. On the other hand, there are many kinds of signal processings, such as filter and error control, which are often used in various radio systems. Thus, making libraries of these signal processings makes it possible to reuse the libraries and to reduce cost and turn around time to develop software defined radio equipment.

In order to achieve the above object, it is important to define an interface and operation of a library. So, we propose a software library which has uniform interface with distributed control function. Fig. 2 and Tab. 2 show the interface of the proposed software library. Here, a signal processing block is a block which carries out a signal processing, such as filter, FFT, CRC and so on.

The each of IN and OUT consists of DAT, CTRL and RCTRL. The DAT is input and output data of the library. The CTRL is forward control signals associated with the DAT. The CTRL includes a valid control signal and timing identifier signals. The valid control signal denotes whether the associated DAT is valid or not. The timing identifier signal points properties of the associated DAT, such as the beginning or the end of the radio communication frame. The details of the valid control and timing identifier signals are described in the section 2.1.2 and 2.1.4 respectively. The RCTRL is a reverse control signal associated with the DAT. The RCTRL includes flow control signal. The flow control signal controls the input / output data flow of the library. The detail of the flow control signal is described in the section 2.1.3.

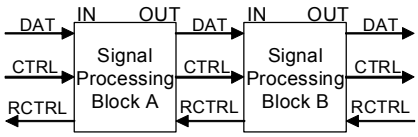

Fig. 2 Interface of the software library

Tab. 2 Function of the software library interface

Name	Description
DAT	Input / Output data
CTRL	Forward control associated with DAT
RCTRL	Reverse control associated with DAT

2.1.2. Valid Control Signal

The valid control signal shows whether the associated DAT is valid or not. Fig. 3 shows an example of the basic operation of the valid signal. When the valid datas, X_1, X_2 and X_3, are input, the signal processing block carries out the signal processing and output the valid datas, $F(X_1)$, $F(X_2)$ and $F(X_3)$, respectively, where F is the signal processing function of this block. When a invalid data is input, the block do no operation and output a invalid data. In this case, the input data rate is equal to output one. The signal processings, such as filter and gain control, belong to this type of block.

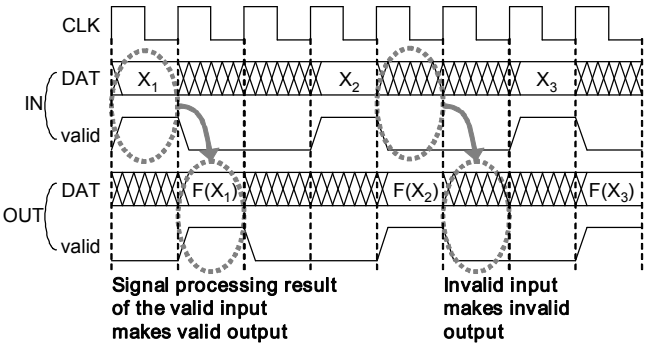

Fig. 3 Basic operation of the valid control signal

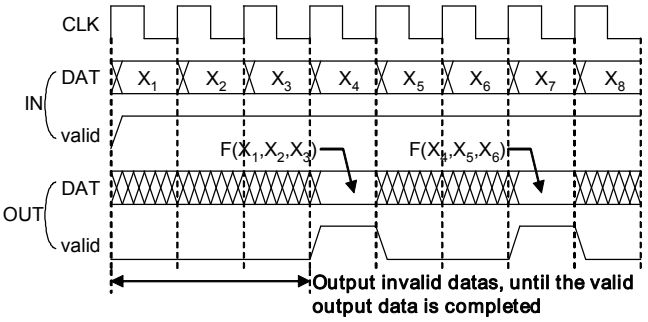

Fig. 4 Valid control signal ("Input Rate > Output Rate" block)

Fig. 4 shows an example of a valid control signal of the signal processing block whose input data rate is greater than the output one. This example assumes that the block makes one output data, such as $F(X_1, X_2, X_3)$ and $F(X_4, X_5, X_6)$, by using three input datas, such as (X_1, X_2, X_3) and (X_4, X_5, X_6). In this case, while X_1, X_2 and X_3 are input, the block can't output valid data, so the block outputs invalid data. After the three datas are input and the signal processing is completed, the block output one valid data, $F(X_1, X_2, X_3)$. The signal processings, such as I/Q mapping and accumulator, belong to this type of block.

2.1.3. Flow Control Signal

The flow control signal of a library (Block B in Fig. 2) controls the output of the previous library (Block A in Fig. 2) whose output is connected to the library (Block B in Fig. 2). Fig. 5 shows a basic operation of the flow control signal.

Receiving flow=1 from OUT side at the cycle T_1, this signal processing block outputs invalid (valid=0) data to OUT side and flow=1 to IN side at the next cycle T_2. As the result of the same operation of the previous block, the input data from the previous block turn to invalid at T_3. Then, when flow from OUT side return to 0 (T_4), the block outputs valid (valid=1) data to OUT side and flow=0 to IN side at the next cycle T_5. As is mentioned above, a block is controlled its output data by receiving flow control signal output by the next block, and can control the output data of the previous block.

Fig. 6 shows an example of a flow control signal of the block whose input data rate is less than the output one. This example assumes that the block makes three output data, such as $F_a(X_1)$, $F_b(X_1)$ and $F_c(X_1)$, by using one input data X_1. In order to avoid buffer overflow caused by the difference between input and output data rate, this block have to control the output data of the previous block by outputting two cycle length flow=1 (T_1 and T_2) after one valid input data (T_0). The signal processings, such as convolution encoder and spectrum spreading, belong to this type of block.

2.1.4. Timing Identifier Signal

There are many timing signals, such as start of a frame, end of a slot and start of CRC calculation, are used in the signal processings. In order to realize the distributed control signal processing, the timing signals have to be sent associated with the data signals. In this paper, we propose a timing identifier signal to express timings. Fig. 7 shows an example of the timings and timing identifier signals(8bit hex value) for IEEE 802.11b physical layer packet.

The timing identifier signal is included in the forward control signal (CTRL). Fig. 8 shows timing identifier processing of a library block. A library block decodes the input timing identifier signal to make its internal control signals, such as start or end of operation. If a library block makes a new timing identifier signal, the library block encodes its own internal control signals to make and output a timing identifier signal. For example, the library block which calculates PLCP_CRC in Fig. 7 operates as below. This block starts and stops the PLCP_CRC calculation when it receives the timing identifier signal "signal_start (0x05)" and "length_end (0x0A)" respectively. Then, this block outputs PLCP_CRC data in OUT DAT signal, and "crc_start (0x0B)" and "crc_end (0x0C) in OUT CTRL signal at the start and end of PLCP_CRC data respectively.

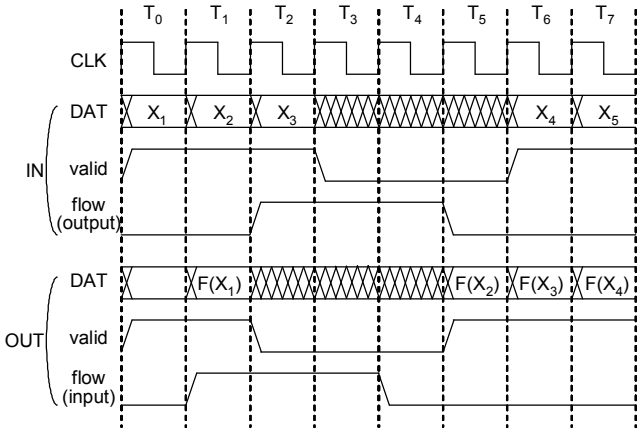

Fig. 5 The flow control signal operation (at Block B in Fig. 2)

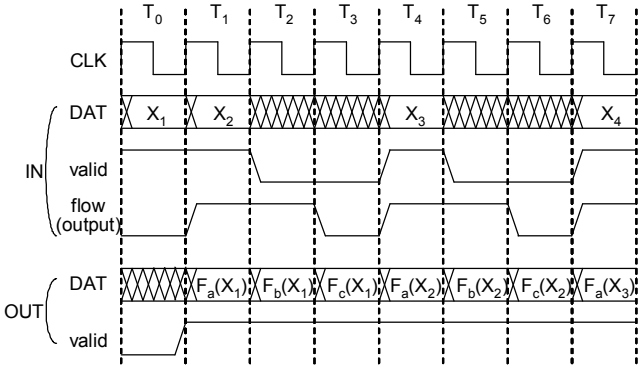

Fig. 6 Flow control signal ("Input Rate < Output Rate" block)

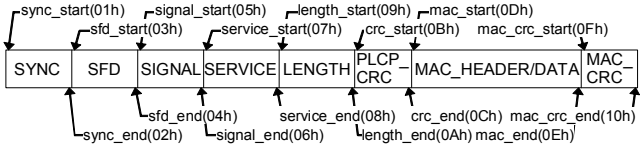

Fig. 7 Timing example of wireless LAN(802.11b)

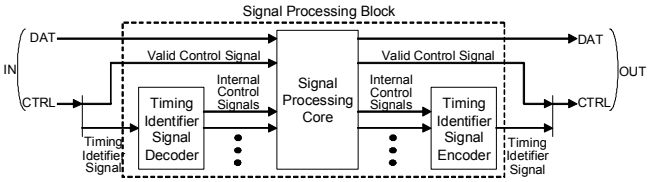

Fig. 8 Timing identifier signal processing

2.2. Analog Front-End

Considering the flexibility and versatility of a software defined radio, it is prefer to increase digital signal processings and to reduce the analog circuit as much as possible. But it is impossible to execute all signal processings in digital, because of the limit of digital circuit operating frequency, digital signal processor ability,

conversion rate of ADC/DAC, cost, power consumption and so on. Thus, it is very important where a sampling part that convert a signal between analog and digital shall be allocated. There are three alternatives for sampling, that is RF, IF and baseband. Taking account of the abilities of the current devices, such as ADC and DAC, we think that the baseband sampling is the more efficient than others. So, in this section, we propose a baseband sampling method in order to sample various signals with different band width using common antialias filter, ADC and DAC.

In the followings, the proposed sampling method is explained using three example systems having symbol rates shown in Tab. 3. In general, considering characteristic deterioration and so on, around four times oversampling is often used. Assuming four times oversampling, the sampling frequencies of system A, B and C are 15.36(f_{SA}), 44(f_{SB}) and 80(f_{SC}) [MHz] respectively (shown in Tab. 3). As shown in Fig. 9, these sampling frequencies require their specific antialias filters (I), (II) and (III) with different cut-off frequency of f_{C1}, f_{C2} and f_{C3}. So, in order to support all of three systems, the software defined radio equipment need to have these three filters, (I), (II) and (III), and select one of them, or have a filter which can change its cut-off frequency. But the former case results in increase of cost and size, and the later case is difficult to realize with devices in general use.

Tab. 3 Examples of sampling parameter

System Name	Symbol Rate[Mbps] (f_{BX})	Over-sample	Sampling Freq. [MHz] (f_{SX})	Proposed Over-sample (m_X)	Proposed Sampling Freq. [MHz] (f_{PX})
A	3.84	4	15.36	21	80.64
B	11		44	8	88
C	20		80	4	80

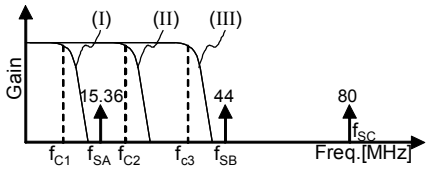

Fig. 9 Sampling frequencies and antialiasing filters

In the proposed method, the sampling frequencies and cut-off frequency of the antialias filter are calculated by the following method. Considering the characteristics, such as symbol rate and performance, the highest sampling frequency among the target systems is selected as a reference sampling frequency f_{SR}. In this section, we assume the four times oversample sampling frequency of system C (f_{SC}=80[MHz]) as the reference sampling frequency. Then, the proposed sampling frequency for system A, B and C (f_{PX}) is set by using symbol rate (f_{BX}), proposed oversample factor (m_X) and reference sampling frequency f_{SR} as below,

$$f_{PA} = f_{BA} \times m_A, \quad f_{PB} = f_{BB} \times m_B, \quad f_{PC} = f_{BC} \times m_C \qquad (1)$$

$$f_{PA} >= f_{SR}, \quad f_{PB} >= f_{SR}, \quad f_{PC} >= f_{SR} \qquad (2)$$

where m_A, m_B and m_C are positive integers and the formulas (1) and (2) are necessary conditions. In order to avoid the increase of power consumption and operation frequency of the circuit, it is preferable to add the following conditions.

m_A is the smallest value that satisfies $f_{PA} >= f_{SR}$
m_B is the smallest value that satisfies $f_{PB} >= f_{SR}$ (3)
m_C is the smallest value that satisfies $f_{PC} >= f_{SR}$

Using the formulas (1), (2) and (3), the proposed oversampling (m_X) and proposed sampling frequencies (f_{PX}) are calculated as shown in Tab. 3.

On the other hand, the cut-off frequency (f_C)of an antialias filter is calculated by using the proposed sampling frequencies (f_{PX}) as below,

$$MAX(f_{MA}, f_{MB}, f_{MC}) <= f_C <= MIN(f_{PA}, f_{PB}, f_{PC})/2 \qquad (4)$$

where f_{MX} is the highest frequency of the baseband signal band width of a system, and MAX() and MIN() is the functions which select maximum and minimum value respectively. Assuming that f_{MX} is equal to a half of the symbol rate (f_{BX}), the cut-off frequency of an antialias filter fC is calculated by using formula (4) and Tab. 3 as below.

$$10[MHz] (=f_{MC}) <= f_C <= 40[MHz] (=f_{PC}/2) \qquad (5)$$

Fig. 10 shows sampling frequencies and antialiasing filter characteristics calculated by above mentioned method. By using the proposed method, an antialiasing filter (IV) can be commonly used by all systems A, B and C, so the problems to select one of the filters or to change the characteristics of an antialiasing filter.

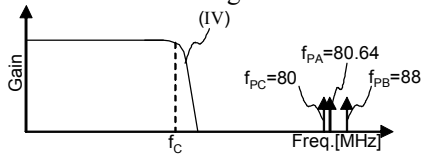

Fig. 10 Proposed sampling frequencies and an antialiasing filter

2.3. Digital Signal Processing Hardware

The digital signal processing hardware of the software defined radio need to support various wireless systems used in the present and developed in the near future. Tab. 4 shows main requirements for the digital signal processing hardware.

Fig. 11 shows a proposed architecture of the digital signal processing hardware. CPU controls the whole part of the radio equipment and carries out signal processings with soft time limitation, such as the protocol control. The memory and program memory is used for the work memory

of CPU processing and the storage space of CPU programs respectively. The external I/F is responsible for the interface between this digital signal processing hardware and the external devices, such as microphone, speaker and PC. The signal processing unit mainly consists of DSP and reconfigurable device, such as FPGA, and executes signal processings with hard time limitation. The proposed digital signal processing hardware architecture has two features. One is the separation of data bus and control bus, and the other is the unit construction of the signal processing part. These features are described below.

Tab. 4 Requirements for digital signal processing hardware

Item	Requirement
Versatility	- Support various systems in current use
Scalability	- Support systems developed in the near future - Can add processing capability - Downsizing and cost reduction with device progress
Multitask	- Can use plural systems at the same time
Complexity	- Simple structure and small volume

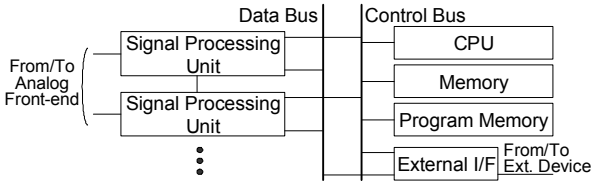

Fig. 11 Digital signal processing hardware architecture

2.3.1. Separation of Data Bus and Control Bus

The data rate of radio communication is increasing rapidly in recent years. So, the load of the data transfer in the radio equipment becomes very heavy. For example, the data transfer occupies about 50% of the bus capability, when the datas of 100[Mbps] are transferred through PCI bus which has 32bit data width and 33[MHz] clock rate. Moreover, the occupancy by data transfer increases when multitask service is realized. This means that the data transfer mainly occupies the bus capability and will cause the lack of the bus capacity.

In order to avoid the lack of bus capacity, we propose the architecture with two buses, that is a control bus and a data bus, shown in Fig. 11. The components of the digital signal processing hardware are connected to the control bus and/or the data bus.

The control bus mainly transfers control signals among the CPU and other components. In order to reduce the load of the CPU and to make it easy to develop the software of the CPU, the control bus needs to have some functions, such as DMA and interrupt. So, a general-purpose bus, such as PCI and VME, can be used as the control bus.

On the other hand, the data bus mainly transfers data signals between a signal processing unit and the external I/F. As shown in Fig. 12, the data bus logically makes one-to-one connection between a signal processing unit and the external I/F. So, the control bus can be implemented by simple structure and it is contemplated that the increase of the volume and complexity of the hardware is small due to adding the data bus.

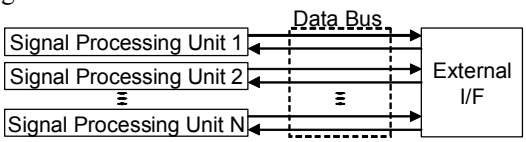

Fig. 12 Logical connection of the data bus

By separating the control bus and data bus, when the data transfer rate increases because of the speed-up of the radio communication and multitask application, the signal processing hardware can adapt to such a situation by modifying only the data bus, such as speeding up the clock frequency and/or widen the data width of the data bus. So, the hardware can deal with the change of the radio communication trend by the minimum modification.

2.3.2. Digital Signal Processing Unit Construction

As shown in Tab. 4, the digital signal processing hardware of a software defined radio equipment is required to have scalability to support new system developed in the near future and so on. Thus, the signal processing capability of the digital signal processing hardware need to be easy to add, subtract and change. So, as shown in Fig. 11, we propose the unit construction of the digital signal processing part. Fig. 13 shows an example of the digital signal processing unit.

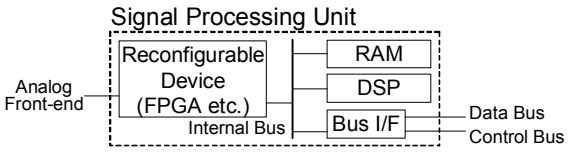

Fig. 13 Example of signal processing unit

The signal processing unit has to support three interfaces, that is analog front-end I/F, data bus I/F and control bus I/F, in order to fit the architecture shown in Fig. 11. But by unitization, the internal structure of the signal processing unit does not affect the whole hardware of the software defined radio. So, if the newly developed system requires more powerful signal processings, the software defined radio equipment can support the system without any change except for replacing the old signal processing unit with new one which comprises more powerful DSP and reconfigurable devices.

3. PROTOTYPE DEVELOPMENT

We have developed a prototype of the proposed software defined radio platform. Fig. 14 and Tab. 5 show the block diagram of the developed prototype and the hardware specifications respectively. The AD/DA Board realizes the functions of the analog front-end described in section 2.2. The sampling signals, which is used and made in the AD/DA Board, are connected to the DSP Board via LVDS Serial Bus.

The DSP Board realizes the functions of the digital signal processing hardware described in section 2.3. By installing a signal processing software which correspond to each radio system, the DSP Board can carry out signal processings, such as modulation and coding, to the sampling signals. The software is installed on the DSP Board from the PC using Card Bus, Bridge and Compact PCI Bus. The tranceived datas, which are signal processed in the DSP Board, are connected to the PC via LVDS Serial Bus, Data Bus Board and USB 2.0. This prototype has up to two DSP Boards, and different softwares can be installed to each DSP Board. So, this prototype can support two different radio systems simultaneously. Fig. 15s are photographs of the prototype.

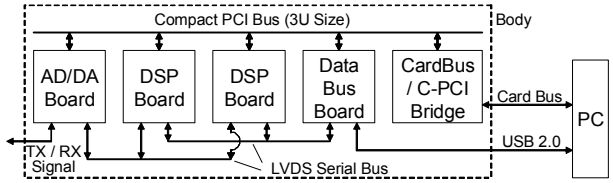

Fig. 14 Block diagram of the developed prototype

Tab. 5 Hardware specification of the prototype

Item	Specification
AD/DA Board	DAC : 14bit@88Msps 4ch ADC : 14bit@88Msps 4ch
DSP Board	FPGA : (6M Gate Class + 5.8 Mbit RAM)x2 DSP : 8000 MIPS (1GHz)
Data Bus Board	USB 2.0 Interface
LVDS Serial Bus	DSP-AD/DA Board : 528MByte/sec DSP-Data Bus Board : 50MByte/sec

(a) Prototype System (b) DSP Board

Fig. 15 Developed prototype

We have developed two signal processing softwares which support the physical layer basic functions of Wireless LAN and Advanced PHS respectively (Refer to Tab. 6). The softwares are built up based on the libraries described in section 2.1. The libraries, such as CRC and mapping, can be commonly used among the signal processing softwares for different radio systems. In this development, 31 % of the total libraries used in the softwares for Wireless LAN and Advanced PHS are commonly used among the two softwares.

By this prototype, it is confirmed that the proposed software defined radio platform can support various radio systems, and improve the efficiency of the radio equipment development.

Tab. 6 Software specification of the prototype

Item	Wireless LAN	Advanced PHS
Data Rate	11 [Mbps]	256 [kbps]
Modulation	- DBPSK + Spread - CCK	- pi/4-QPSK - 16QAM
Shared Rate	43 % of signal processing libraries are shared	
Others	- Without MAC ARQ	- Assumed Slot Format

4. CONCLUSION

In this paper, the SDR platform is proposed and developed. The SDR platform consists of three components. The first is software library, that can be commonly used for signal processings of various radio communication systems, by having the distributed control function. The second is analog front-end that supports different radio parameters for many radio systems. The other is a versatile and scalable digital signal processing hardware. By developing a prototype of the SDR platform and the softwares that are correspond to IEEE 802.11b and advanced PHS, it is confirmed that the SDR platform can adapt to various radio communication systems, and improve the efficiency of the SDR equipment development.

5. REFERENCES

[1] http://www.kddi.com/english/corporate/news_release/2005/0615/index.html

[2] J. Mitola, "The Software Radio Architecture", IEEE Communications Mag., vol. 33, issue 5, pp.26-38, May 1995.

[3] K. Araki, Software Defined Radio and Its Applications, Sipec, Japan, October 2002.

[4] H. Harada, "Software Defined Radio Prototype for W-CDMA and IEEE 802.11a/b Wireless LAN and Digital Terrestrial Broadcasting", IEICE Technical Report SR2005-33, July 2005.

[5] A. Okazaki, "A Platform for Software Defined Radio - Overview of its Platform and Implementation of Multi-channel Demodulators- ", IEICE Technical Report SR03-12, December 2003.