SMART ANTENNA API FOR SDR NETWORK

Namkyu Ryu(HY-SDR Research Center, HIT406,School of Electrical and Computer Engineering, Hanyang University, Seoul, Korea, ryu7424@dsplab.hanyang.ac.kr)
Taeyoul Oh(HY-SDR Research Center, HIT406,School of Electrical and Computer Engineering, Hanyang University, Seoul, Korea, tyoh@dsplab.hanyang.ac.kr)
Seungwon Choi (HY-SDR Research Center, HIT406,School of Electrical and Computer Engineering, Hanyang University, Seoul, Korea, choi@dsplab.hanyang.ac.kr)
Soonjoon Park(Standardization & System Research Group, Mobile Communication Technology Research Lab., CTO, LG Electronics, Anyang-shi, Kyonggki-do, Korea, skypark@lge.com)

ABSTRACT

The objective of this paper is to provide the APIs of a Smart Antenna Base Station(SABS) operating in SDR network in such a way that the APIs(Application Program Interface) are suitable for the entire system to maintain the openness, object-oriented design, and software controllability. The software and hardware of SABS is first modularized and partitioned into small modules, respectively. Then, the interface among the modules are specified for determining the SA API properly for the SDR network. The suitability of the proposed APIs is verified through a design example of SABS implemented in accordance with the proposed APIs. The performance of the proposed system is shown in practical signal environments of CDMA2000 1X with commercial handsets operating in various data rate ranging from 9.6 kbps to 153.6 kbps in terms of FER (Frame Error Rate) and SINR (Signal to Interference plus Noise Ratio) which is drastically enhanced through the nicely shaped beam pattern.

1. INTRODUCTION

The objective of developing the SDR technology is to realize plural system standards on a single hardware platform that is implemented mainly with the high-speed programmable digital signal processing devices[1]. A desired system standard can be selected by choosing a proper software module.

This paper addresses the problem of designing the hardware and software architecture of SABS that operates in an SDR network. A design example of SABS architecture that satisfies the requirements of SDR functionalities is also provided in this paper. We propose a hardware platform employing the open architecture of SABS, with which one can realize the multimode SDR system by selecting the modularized software. Note that the hardware platform itself remains unchanged while selecting a desired system standard among plural different standards[2].

The SDR technology includes the design of both hardware and software modules. The hardware module is reconfigured by the software module, which means that a given hardware platform is converted into a specific system standard or special-purpose communication system depending on the change of the software module. It is the most important feature of the SDR technology that a system update or an addition/deletion/modification of services can be performed extremely easily without changing the existing hardware[3].

In this paper, we present an open architecture of SABS that is suitable to the SDR network in such a way that one can fully exploit the merits of both smart antenna and SDR technologies[4]. The proposed architecture has been applied to implement a system of SABS, which includes the modulation and demodulation parts of the SABS together with the interfaces with the SDR network as well as that among the modules within the SABS. The suitability of the proposed open architecture is demonstrated through a quantitative analysis obtained through the various experimental measurements provided from the design example of SABS.

2. SABS API

The objective of this section is to summarize the requirements and logical functionalities of the SA API, which itself defines the interfaces among system software inside the SABS.

2.1. REQUIREMENTS

As our consideration in this paper is focused on an SDR-based SABS for next generation mobile communications, the hardware and software of SABS should cope with the modularization to provide object-oriented design and openness to various communication standards, together with the capability for distributed processing [4].

After all, the key parts in designing the SABS OA are to partition the entire SABS into several small modules, to define the interface among the modules, and finally to establish an efficient SA API for defining interfaces between the system software inside the SABS. With the efficient SA API, SABS can be smoothly interlocked with the SDR network. The required features of the SA API which defines the interfaces among system programs inside the SABS are as follows.

- The various SA algorithms must be applicable to SDR-based wireless communication systems so that SA API does not confine the evolution of communication standards and system hardware.
- Interface between SABS and the SDR network must operate independently of hardware so that the given hardware can be reconfigured freely in accordance with the desired system standard.
- SABS should be partitioned into small modules and each of the modules should interface independently of various algorithms and communication standards.
- The functions and capabilities of each module must be known to the network controller so that the SA function inside a given SABS is manageable through the SDR network.
- The SDR network interface should be independent of the system standard.

The logical functionality of SA API discussed in this paper is based on the API framework defined in [2] [5]. The logical functionalities are defined by SA API primitives consisting of “command,” “variable,” “response,” and “signal” explained as follows.

Commands: Asynchronous protocols to SABS primitives for performing immediate and non-persistent actions.
Variables: Persistent SABS state or long-term measurement primitives.
Response: Synchronous response to command or variable operation.
Signals: Asynchronous protocols to SDR Network primitives for reporting recent and non-persistent events.

Fig. 1 illustrates the interface between SDR network and SABS in terms of SA API primitives.

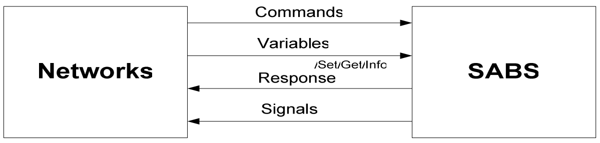


Figure 1: Interface between SDR Network and SABS through SA API primitives

2.2. API PRIMITIVES

In this section, SA Primitives for the SDR-based network are defined. As discussed in Section II, the functions of the SA Primitives required in SA API are “commands,” “signals,” and “variables,” which are shown below in Tables 1, 2, and, 3, respectively. The SA API primitives shown in this section have been designed in such a way that the SABS can be used in an SDR-based network without completely invalidating the cell planning for the existing conventional base stations. Various beamforming algorithms shown in Table 1, together with the call processing operations, antenna topologies, path calibrations, etc., have been taken into consideration in the design of the SA Primitives shown in Tables 1, 2, and 3.

Table 1: Commands

Commands	Requirements	Qualifiers	Description	Response
CmdBeamformerReset	Mandatory		Beamformer Soft Reset	Beamformer Soft Reset OK Beamformer Soft Reset Failure
CmdBeamformerExec	Mandatory		Beamformer Execution on/off	Beamformer Execution OK Beamformer Execution Failure
CmdCalibrationExec	Mandatory		Calibration Execution on/off	Calibration Execution OK Calibration Execution Failure
CmdBeamformerDMExec	Optional		Beamformer Diagnostic monitoring on/off	Beamformer Diagnostic monitoring OK Beamformer Diagnostic monitoring Failure

Table 2: Signals

Signals	Requirements	Qualifiers	Description
SignBeamformer	Mandatory		Beamformer Module loaded
SignBeamformerError	Mandatory	Interrupt	Indicating the Beamformer Error

Table 3: Variables

Variables	Requirements	Qualifiers	Description
VarMode	Mandatory	Info	Return set of available system modes
		Set	Set system mode (Array, Diversity, Single Ant etc)
		Get	Get current system mode
VarAlgorithm	Mandatory	Info	Returns set of available algorithms (Response)
		Set	Set algorithm type (MMSE, Eigen-based, RLS,etc)
		Get	Get current algorithm type
VarBeamDirection	Optional	Info	Returns set of available beam directions (For tracking of particular users)
		Set	Set direction of beam (Tx, Rx)
		Get	Get direction of beam (Tx, Rx)
VarCalibrationMode	Optional	Info	Returns set of available calibration Modes
		Set	Set Calibration Mode
		Get	Get current Calibration Mode
VarCalibration		Get	Get current Calibration value (Tx, Rx)
VarLevelofCalibration	Mandatory	Info	Returns level of calibration required for each application
		Set	Set level of calibration
VarAntType	Mandatory	Info	Returns the various antenna topologies available for eg. Circular array, planar array for the SA to be configured
		Set	Set antenna type
		Get	Get current antenna type
VarRadiationPattern	Mandatory	Info	Returns maxRadiationPattern, minRadiationPattern
		Set	Set radiation pattern for antenna array as well as for single antenna
		Get	Get radiation pattern
VarNoofElements	Optional	Info	Returns the number of antenna elements
		Set	Set number of radiating antenna elements
		Get	Get number of radiating elements
VarAuxPilotWalsh	Mandatory	Info	Returns Ability of Auxiliary Pilot Decoding
		Set	Set Auxiliary Pilot Walsh (Walsh Num, Walsh length, QOF)

		Get	Get current Auxiliary Pilot Walsh (Walsh Num, Walsh length, QOF)
VarAuxPilotRelative Gain	Mandatory	Info	Returns Ability of Auxiliary Pilot Decoding
		Set	Set Auxiliary Pilot Relative Gain
		Get	Get current Auxiliary Pilot Relative Gain
VarSmart AntennaHPAPower	Mandatory	Info	Returns Available Output Power Range of HPA Linear Operation (dBm, Watts)
		Set	Set HPA output Power
		Get	Get Current HPA output Power
VarRevPowerConStepGain	Mandatory	Info	Returns Enables or Disable of Reverse Power Control
		Set	Set Reverse Power Control Step Gain (dB)
		Get	Set Current Reverse Power Control Step Gain (dB)
VarFowPowerConStepGain	Mandatory	Info	Returns Enable or Disable of Forward Power Control
		Set	Set Forward Power Control Step Gain (dB)
		Get	Get Current Forward Power Control Step Gain (dB)

2.3. API USE CASE

In this section we present a use case of each of the API primitives in the preceding section.

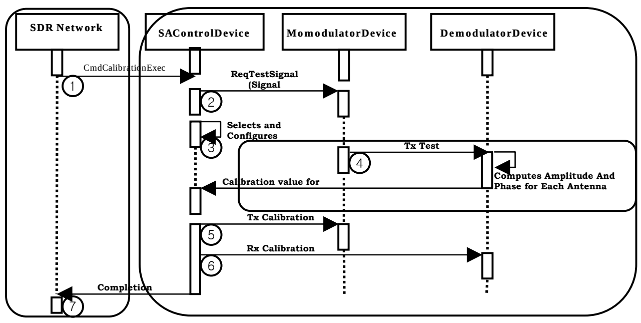


Figure 2: Use case (Calibration Execution Command)

Figure 2. shows a use case of calibration execution command. The message flow between modules is described as follows

①:SAController Module receives a request of calibration command from the SDR network
②:SAController Module passes a message for modulator to generate test signals for Rx or Tx calibration
③:SAController Module selects and configures required RF chain for Rx or Tx calibration
④:Demodulator Module Computes Rx and Tx calibration Value
⑤:SAController Module sends Tx compensation values to modulator
⑥:SAController Module sends Rx compensation values to demodulator
⑦:SAController Module sends a completion message to the SDR network

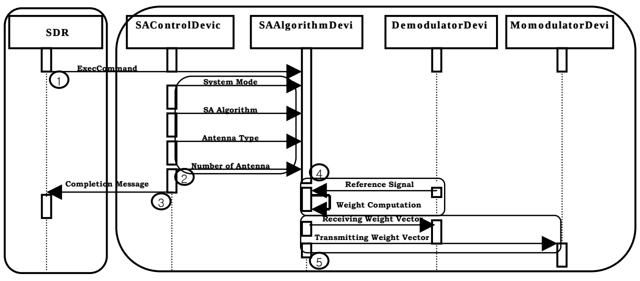


Figure 3: Use case (Beamformer Execution Command)

Figure 3. shows a use case of Beamformer execution commanand. The message flow between modules is described as follows

①:Beamforming Module receives the beamformer execution command
②:Beamforming Module fetches system mode
Beamforming Module fetches SA algorithm
Beamforming Module fetches antenna type
Beamforming Module fetches number of antenna
③:SAController Module passes a completion message to the SDR network
④:Beamforming Module fetches reference signal, $\mathrm{Ref}_{\mathrm{Pre}}$ and/or, $\mathrm{Ref}_{\mathrm{Post}}$ from demodulator
Beamforming Module computes receiving and transmitting weight vector based on algorithm selected
⑤:SAController Module sends weight values for demodulator and modulator, respectively

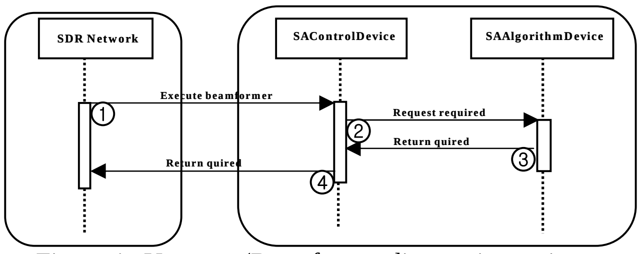


Figure 4: Use case (Beamformer diagnostic monitor Execution Command)

Figure 4. shows a use case of Beamformer diagnostic monitor command. The message flow between modules is described as follows

①:The SDR Network send the beamformer diagnostic monitor execution command to the SA system controller module
②:The SA Controller module passes the data to the SDR network

③:The SA system controller module requests required data reference signal ($\mathrm{Ref}_{\mathrm{Pre}}$ and/or $\mathrm{Ref}_{\mathrm{Post}}$), weight vector from the Beamformer module

④:The beamformer module returns the required data to the SA controller module to the SA system controller module

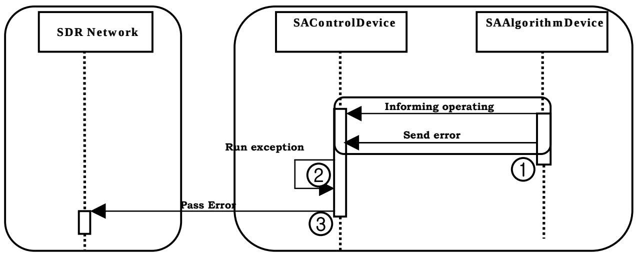


Figure 5: Use case (Beamformer Error Signal)

Figure 5. shows a use case of Beamformer error signal. The message flow between modules is described as follows

①:The Beamformer module detects an error while the beamformer is operating and informs the SA system controller module
The Beamformer module sends the signal to the SA controller module

②:The SA Controller module runs predefined exceptional handler

③:The SA controller module passes the error message to the SDR Network

3. DESIGN EXAMPLE

In this section, we present a design example of SDR-based SABS considering the requirements and interfaces discussed in Section 2. As mentioned in the previous section, the hardware part of SABS to be designed should be abstracted as much as possible from the Application layer For achieving this, the entire system is partitioned into small modules in accordance with the function of each of the modules. Consequently, both hardware and software part are properly partitioned, layered, and modularized. In addition, the middleware of the SABS presented in this section provides the object-oriented operation to each of the hardware resources. Fig. 6. is the photograph of SABS implemented in accordance with the proposed APIs[6][7].

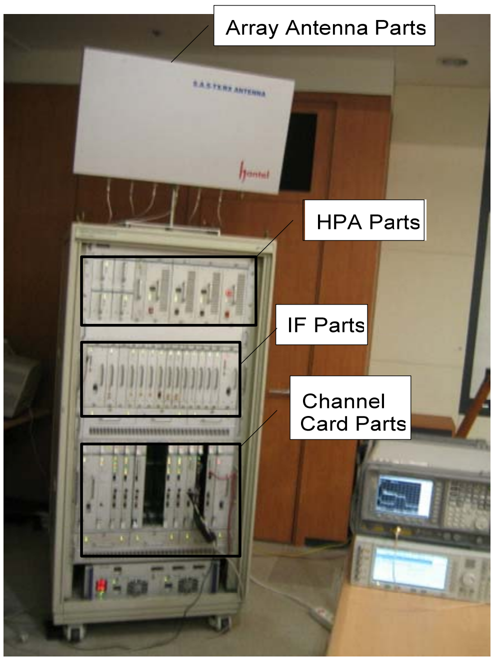


Figure 6: Photograph of SDR-based SABS

4. NUMERICAL ANALYSIS

In Section 3, we presented a design example of a SABS system based on the proposed APIs shown in Section 2. This section provides a performance analysis of the implemented SABS system obtained from the measurements in a practical signal environment of CDMA2000 1X. Through the experimental results, together with the various computer simulations, we claim that the APIs shown in Section 2 and 3 is suitable for designing the SABS system operating in the SDR-based network. The performance of the proposed SABS system is compared to that of the conventional base station system consisting of 2 diversity antennas to demonstrate the superiority of the smart antenna system. The amount of performance enhancement is also analyzed to verify the feasibility of the SABS system in terms of both economic and technical factors. The adaptive algorithm employed in the implemented SABS system is the generalized Lagrange's [8][9]. Any one of the adaptive algorithms could be selected arbitrarily according to the given signal environment. The SABS system implemented in our experiments adopts 6 antennas for uplink and 4 antennas for downlink communication. There is another antenna in the array element for calibration functionality and other experimental purposes. The carrier frequency used in our experiments is what has been assigned as the PCS band in Korea, i.e. 1.7725 GHz and 1.8625 GHz for the uplink and downlink, respectively.

4.1. VOICE AND DATA CHANNEL PERFORMANCE ANALYSIS

Fig. 7. illustrates the FER performance for the voice data (9.6 kbps) provided by the proposed SABS system when the target handset is located at the center of the array antenna. Table 4. shows a comparison between the SABS system and a conventional 2-antenna diversity system when the required FER is set to 1%. Compared to the conventional base station system, SABS provides an enhancement of about 5.8dB, which means a 3-4x increase in capacity can be obtained with the SABS system consisting of 6 receiving antenna elements.

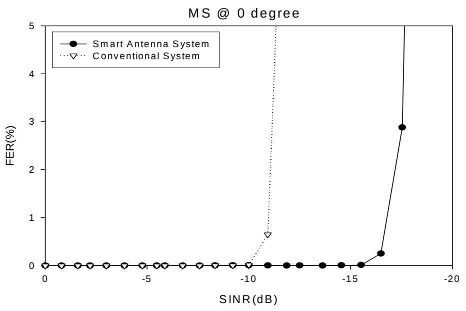


Figure 7: Uplink Voice Channel Performance (when the target handset is located at 0^o)

Table 4: Performance enhancement for FER = 1% (9.6 kbps)

	Conventional System	Smart Antenna System	Enhancement
SINR	**-10.6dB**	**-16.4dB**	**5.8dB**

Note that the location of the target handset is coincident with that of one of the interferers. When the target handset is located differently from the interferers, we have observed that the performance enhancement is much greater, approximately 10-12 dB, due to the deep nulls provided along the direction of each of 3 interferers.

Fig. 8. and Table 5. show the FER performance of the proposed SABS system when the target handset with a data rate of 153.6 kbps is located at the center of the array antenna element. From Table 5., the SABS system exhibits better performance than the conventional one by about 5.5 dB when the required FER is 1%.

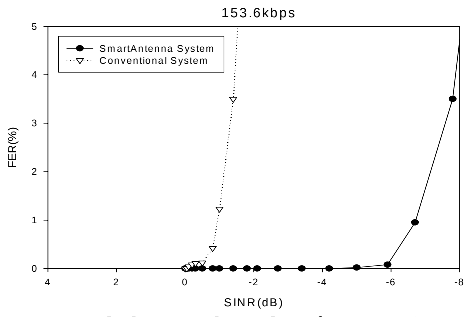


Figure 8: Uplink Data Channel Performance (153.6kbps)

Table 5: Performance enhancement for FER = 1% (153.6kbp)

	Conventional System	Smart Antenna System	Enhancement
SINR	**-1.1dB**	**-6.6dB**	**5.5dB**

Observe that the required SINR for a given FER increases by about 3 dB as the data rate of the target handset 153.6 kbps.

In our experimental analysis for the uplink performances, it has been observed that the proposed SABS system consisting of 6 antenna elements increases the communication capacity, or, equivalently, the total throughput, by about 3-4 times compared to the conventional 2-antenna diversity base station system.

4. 2. DOWNLINK PERFORMANCE ANALYSIS

For the downlink performance analysis, we measure the directivity provided by the downlink beam pattern. The beam pattern has been measured from a receiving antenna while the SABS system transmits a constant power signal with a weight vector in which the steering angle sweeps by 2^0 at a time in the range of -60^0 to $+60^0$. Fig. 9. illustrates the beam pattern of the SABS system consisting of 4 antenna elements when the target receiver is located at 0^o . When the receiving antenna is located at the center, i.e. 0^o , the directivity was about 8.31 dB. The directivity is the average amount of interference reduction provided by the SABS system. In our experiments, it has been observed that the directivity gain of a given SABS system increases when the target handset is located at the center area of the array antenna element.

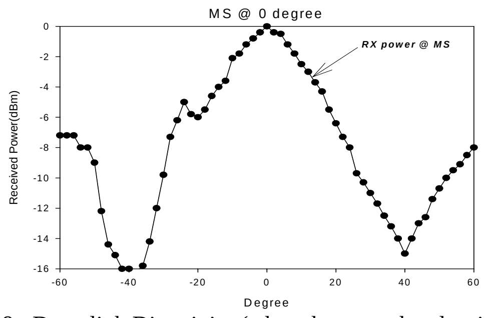


Figure 9: Downlink Directivity (when the target handset is located at the position of 0^o)

5. CONCLUSIONS

In this paper, we presented the APIs of a SABS operating in an SDR network. The SABS APIs in this paper are based on a philosophy that the software architecture should be layered for supporting the extension of application programs while the hardware architecture should be partitioned for supporting the modularization of hardware resources. The layered and partitioned structure is the key aspect for the APIs to be able to accommodate various communication standards utilizing a single hardware platform in real-time. Interface between hardware modules, software modules, and hardware and software modules should also be clearly defined for implementing a desired application successfully.

Based on the principles of the SABS APIs shown in this paper, a practical design example of SABS has been presented with all the hardware resources being partitioned into proper modules and the interconnections among the modules being specified in terms of clock/control signals and common data buses exchanged among modules.

The proposed SABS APIs has been implemented and the performance of the implemented SABS system has been measured in a practical signal environment using commercial handsets operating in CDMA2000 1X circumstances. Through the experimental tests, the suitability of the proposed SABS APIs has been verified. It is also noteworthy that the SABS system implemented in accordance with the proposed open architecture exhibits a conspicuous superiority in its performance compared to the conventional 2-antenna diversity base station system.

Acknowledgement: This paper was supported in part by MIC & IITA through IT Leading R&D Support Project

References

[1] W. Tuttlebee, *Software Defined Radio Baseband Technology for 3G Handsets and Basestations*, John Wiley & Sons, 2003

[2] J. H. Reed, *Software Radio:A modern approach to radio engineering*, Prentice Hall Communications Engineering and Emerging technology series 2002.

[3] J. Mitola, "*The software radio architecture*," IEEE Commun. Mag., vol. 33, no. 5, pp. 26-38, 1995.

[4] OMG, *Architecture and Specification CORBA 2.4.2, Standard document*, OMG, Feb. 2001(available at www.omg.org).

[5] F. Templin, *An Encoding of Radio API Primitives for the ISI APT Radio via the SLIP Protocol*, Apr. 1998.

[6] W. Tuttlebee, "*Software Defined Radio— Origins, Drivers and International Perspectives*," John Wiley & Sons, 2002.

[7] Kurt Keutzer, "*Overview of configurable architectures*", EECS,

[8] H. Im and S. Choi, "*Performance Analysis of Smart Antenna Test-Bed Operating in a Wide-Band CDMA Channel*", IEEE Trans. On Microwave Theory and Techniques, vol. 49, no. 11, pp. 2142-2146, Nov. 2001.

[9] S. Choi and D. Shim, "*A Novel Adaptation Beamforming Algorithm for a Smart Antenna System in a CDMA Mobile Communication Environment*", IEEE Transaction on Vehicular Technology, vol. 49, no. 5, pp. 1795-1799, Sep. 2000.

Smart Antenna API for SDR Network

Taeyoul Oh
HY-SDR Research Center, Hanyang University, Seoul, Korea
tyoh@dsplab.hanyang.ac.kr

Contents

1. Introduction

- The objective of developing the SDR technology is to realize plural system standards on a single hardware platform.
- The APIs(Application Program Interface) of a Smart Antenna Base Station(SABS) operating in SDR network are suitable for the entire system to maintain the openness, object-oriented design, and software controllability.
- We present an open architecture of SABS that is suitable to the SDR network in such a way that one can fully exploit the merits of both smart antenna and SDR technologies.
- The proposed architecture has been applied to implement a system of SABS, which includes the modulation and demodulation parts of the SABS together with the interfaces with the SDR network as well as that among the modules within the SABS.
- The suitability of the proposed open architecture is demonstrated through a quantitative analysis obtained through the various experimental measurements provided from the design example of SABS.

2.1 Requirements

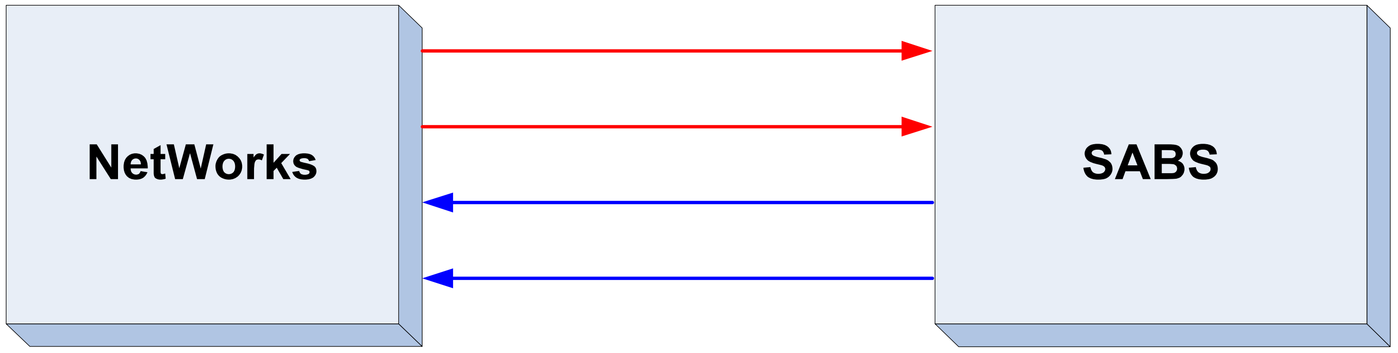
❖ Smart antenna API for SA OA

- The various SA algorithms must be applicable to SDR-based wireless communication systems such that SA API does not confine to the evolution of communication standards and system hardware.
- Interface between SABS and SDR network must operate independently of hardware such that the given hardware can be reconfigured freely in accordance with the desired system standard.
- SABS should be partitioned into small modules and each of modules should interface independently of various algorithms and communication standards.
- Functions and capability of each module must be known to the network controller such that the SA function inside a given SABS should be manageable through SDR network.
- SDR network interface should be independent of the system standard.

2.1 Requirements

❖ Smart antenna API Logical Functionality

- Commands: Asynchronous protocols to SABS primitives for performing immediate and non-persistent actions.
- Variables: Persistent SABS state or long-term measurement primitives.
- Response: Synchronous response to command or variable operation.
- Signals: Asynchronous protocols to SDR Network primitives for reporting recent and non-persistent events.



< Interface between Network and SABS through Network protocol >

2.2 API Primitives(1/4)

◆ Commands

Commands	Requirements	Qualifiers	Description	Response
❖CmdBeamformerReset	Mandatory		❖Beamformer Soft Reset	❖Beamformer Soft Reset OK. ❖Beamformer Soft Reset Failure.
❖CmdBeamFormerExec	Mandatory		❖BeamFormer Execution on/off	❖BeamFormer Execution OK. ❖BeamFormer Execution Failure.
❖CmdCalibrationExec	Mandatory		❖Calibration Execution on/off	❖Calibration Execution OK. ❖Calibration Execution Failure.
❖CmdBeamFormerDMExec	Optional		❖Beamformer Diagnostic monitoring on/off	❖Beamformer Diagnostic monitoring OK. ❖Beamformer Diagnostic monitoring Failure.

◆ Signals

Signals	Requirements	Qualifiers	Description
❖SignBeamformer	Mandatory		❖Beamformer Module loaded
❖SigBeamformerError	Mandatory	❖Interrupt	❖Indicating the Beamformer Error

2.2 API Primitives(2/4)

◆ Variables

Variables	Requirements	Qualifiers	Description
❖VarMode	Mandatory	Info	❖Returns set of available system modes
		Set	❖Set system mode (Array, Diversity, Single Ant. etc)
		Get	❖Get current system mode
❖VarAlgorithm	Mandatory	Info	❖Returns set of available algorithms (Response)
		Set	❖Set algorithm type (MMSE, Eigen-based, RLS, etc)
		Get	❖Get current algorithm type
❖VarBeamDirection	Optional	Info	❖Returns set of available beam directions (For tracking of particular users)
		Set	❖Set direction of beam (TX,RX)
		Get	❖Get direction of beam (TX,RX)
❖VarCalibrationMode	Optional	Info	❖Returns set of available Calibration Modes
		Set	❖Set Calibration Mode
		Get	❖Get current Calibration Mode
❖VarCalibration		Get	❖Get current calibration value (TX,RX)

2.2 API Primitives(3/4)

◆ Variables

Variables	Requirements	Qualifiers	Description
❖VarLevelofCalibration	Mandatory	Info	Returns level of calibration required for each application
		Set	Set level of calibration
❖VarAntType	Mandatory	Info	Returns the various antenna topologies available for eg. Circular array , planar array for the SA to be configured
		Set	Set antenna type
		Get	Get current antenna type
❖VarRadiationPattern	Mandatory	Info	Returns maxRadiationPattern, minRadiationPattern
		Set	Set radiation pattern for antenna array as well as for single antenna
		Get	Get radiation pattern
❖VarNoofElements	Optional Can be used in case of antenna array system	Info	Returns the number of antenna elements
		Set	Set number of radiating antenna elements
		Get	Get number of radiating elements

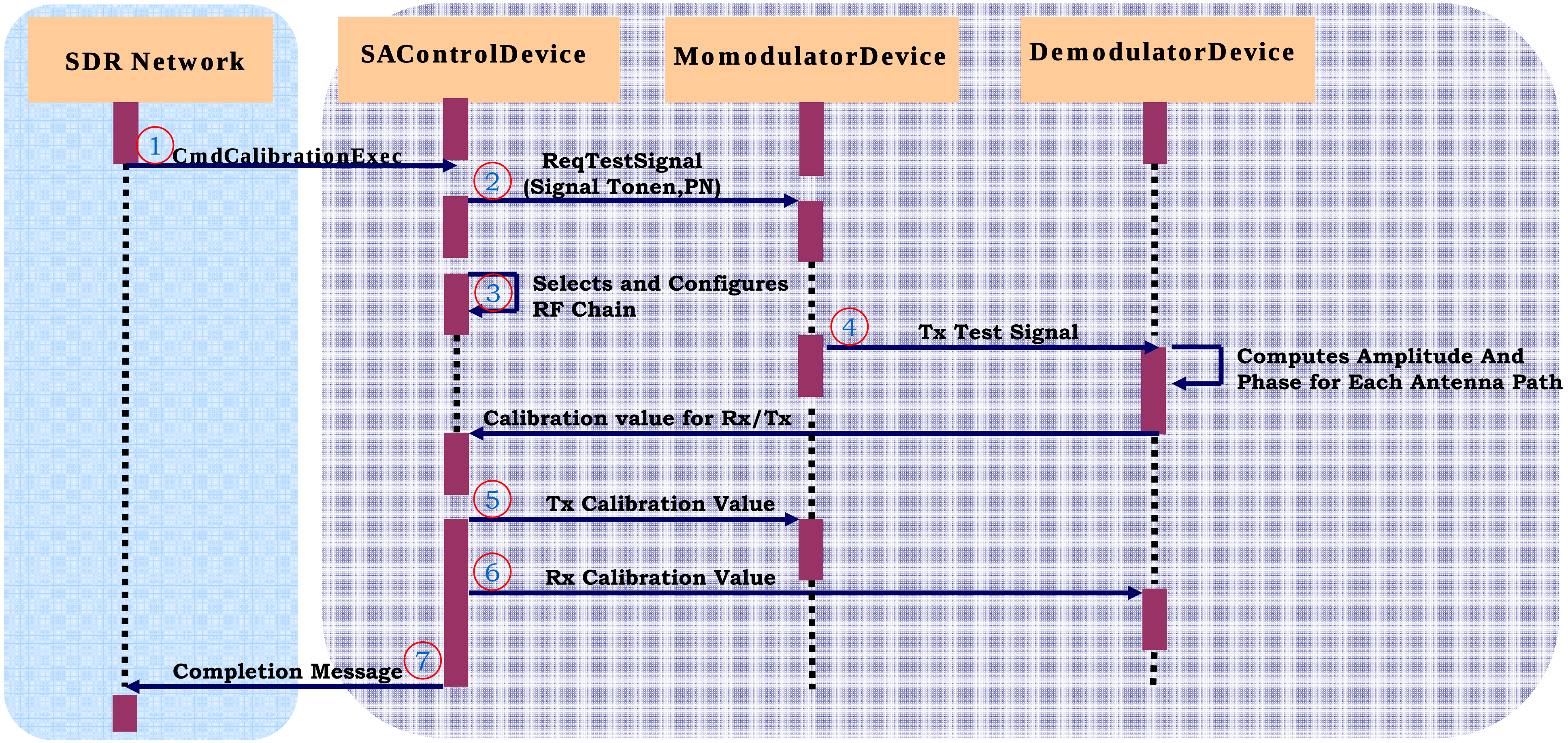
2.2 API Primitives(4/4)

◆ Variables

Variables	Requirements	Qualifiers	Description
❖VarAuxPilotWalsh	Mandatory	Info	❖Returns Ability of Auxiliary Pilot Decoding
		Set	❖Set Auxiliary Pilot Walsh (Walsh Num, Walsh length, QOF)
		Get	❖Get current Auxiliary Pilot Walsh (Walsh Num, Walsh length, QOF)
❖VarAuxPilotRelativeGain	Mandatory	Info	❖Returns Ability of Auxiliary Pilot Decoding
		Set	❖Set Auxiliary Pilot Relative Gain
		Get	❖Get current Auxiliary Pilot Relative Gain
❖VarSmartAntennaHPAPower	Mandatory	Info	❖Returns Available Output Power Range of HPA Linear Operation (dBm, Watts)
		Set	❖Set HPA Output Power
		Get	❖Get Current HPA Output Power
❖VarRevPowerConStepGain	Mandatory	Info	❖Returns Enable or Disable of Reverse Power Control
		Set	❖Set Reverse Power Control Step Gain (dB)
		Get	❖Set Current Reverse Power Control Step Gain (dB)
❖VarFowPowerConStepGain	Mandatory	Info	❖Returns Enable or Disable of Forward Power Control
		Set	❖Set Forward Power Control Step Gain (dB)
		Get	❖Get Current Forward Power Control Step Gain (dB)

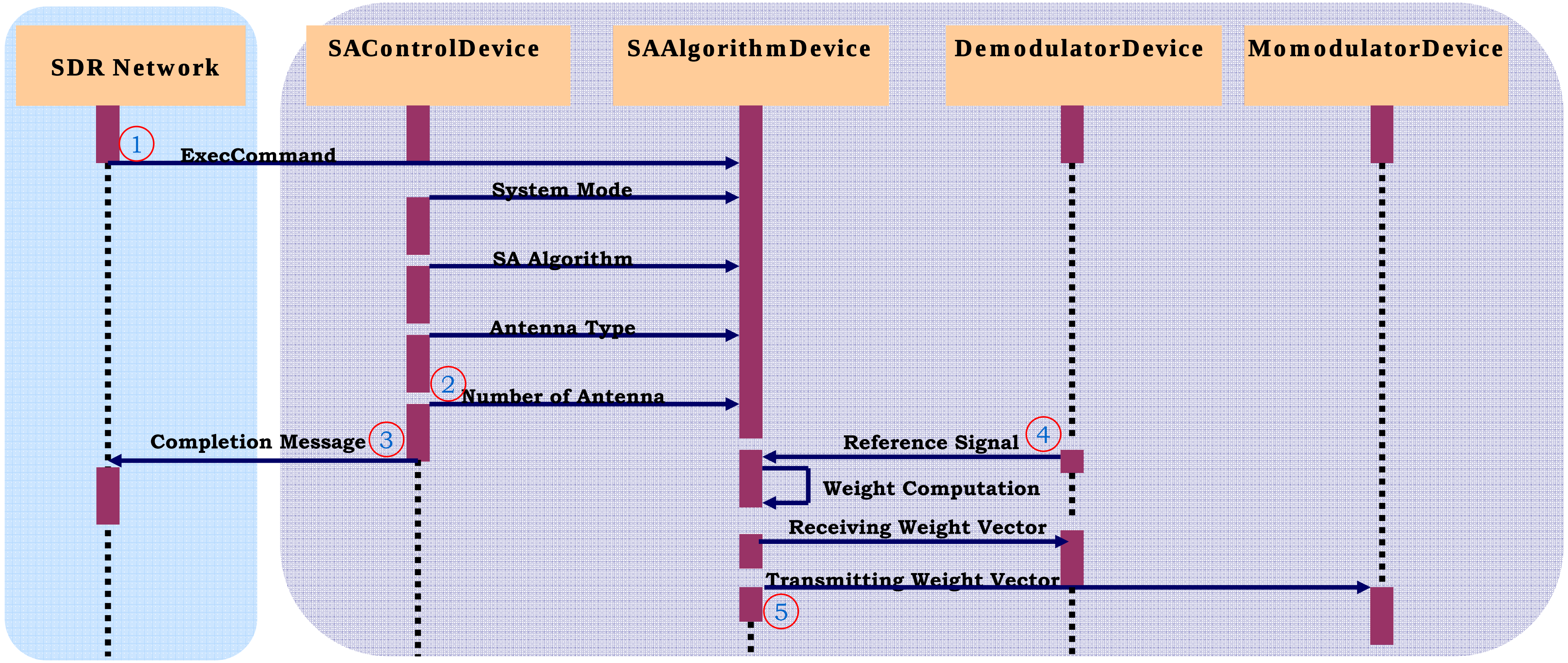
2.3 API USE Case

✓ **Calibration execution command**



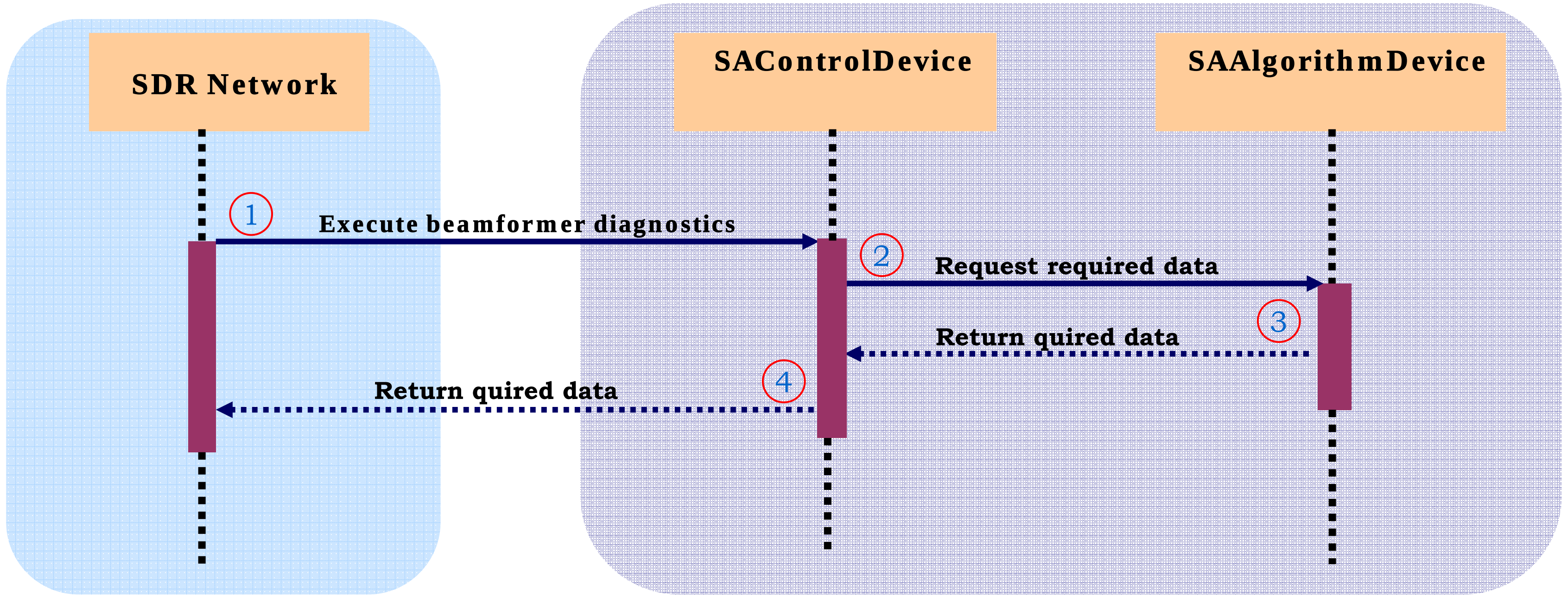
2.3 API USE Case

✓ **Beamforming execution command**



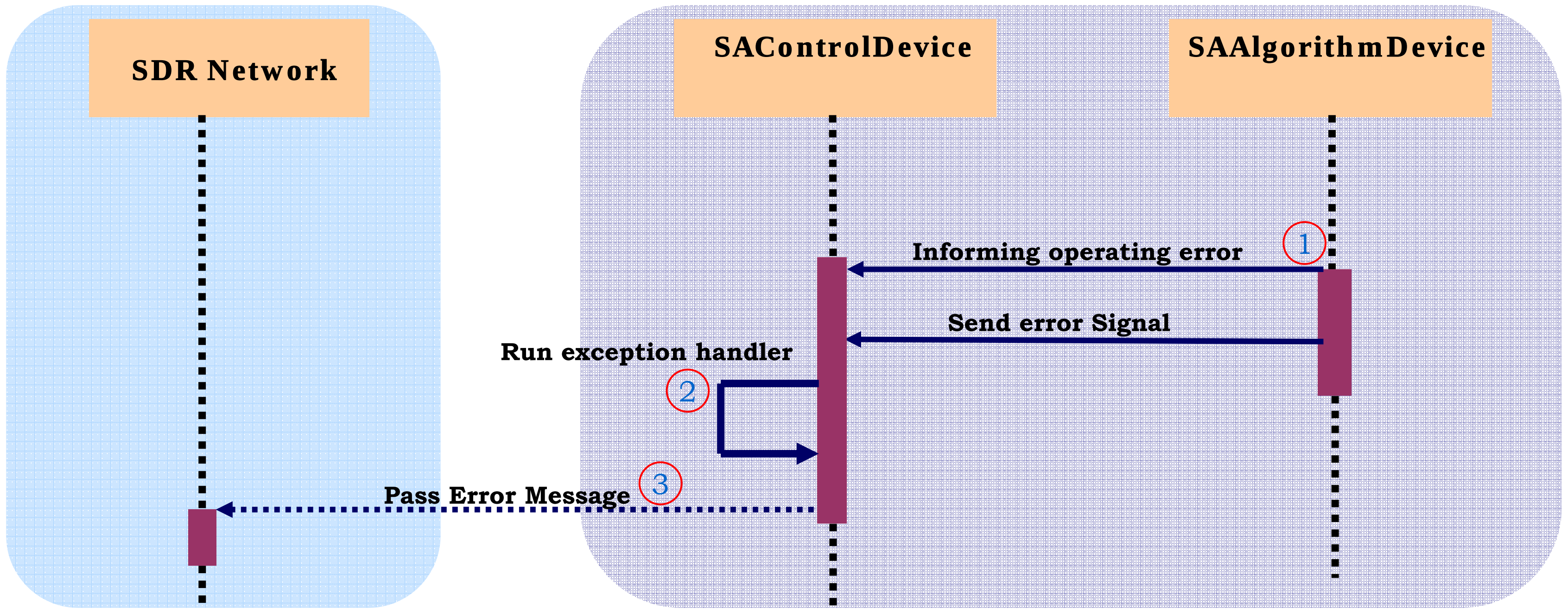
2.3 API USE Case

✓ **Beamformer diagnostic monitor execution command**



2.3 API USE Case

✓ **Beamformer module error signal**



3. Design Example

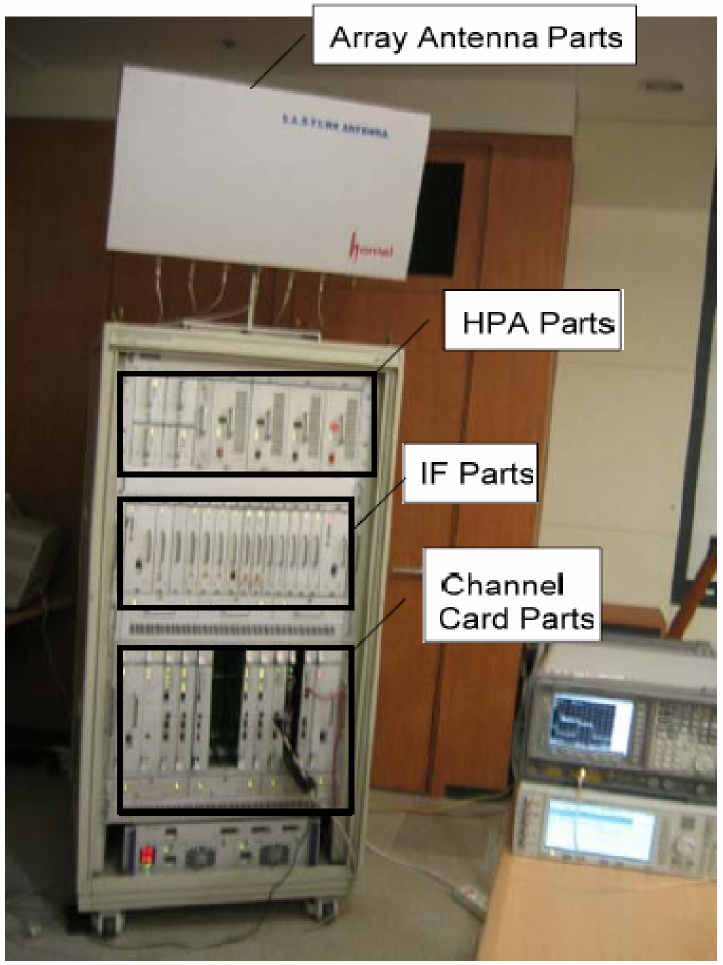


Figure 6: Photograph of SDR-based SABS

4. Numerical Analysis

- It presented a design example of a SABS system based on the proposed APIs.
- The performance of the proposed SABS system is compared to that of the conventional base station system consisting of 2 diversity antennas to demonstrate the superiority of the smart antenna system.
- The adaptive algorithm employed in the implemented SABS system is the generalized Lagrange's.
- The SABS system implemented in our experiments adopts 6 antennas for uplink and 4 antennas for downlink communication.
- There is another antenna in the array element for calibration functionality and other experimental purposes.
- The carrier frequency used in our experiments is what has been assigned as the PCS band in Korea, i.e. 1.7725 GHz and 1.8625 GHz for the uplink and downlink, respectively.

4.1 Voice and Data channel performance analysis

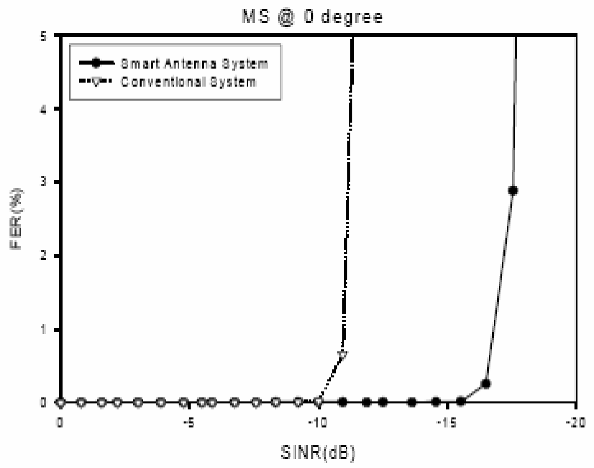


Figure 7: Uplink Voice Channel Performance (when the target handset is located at 0^o)

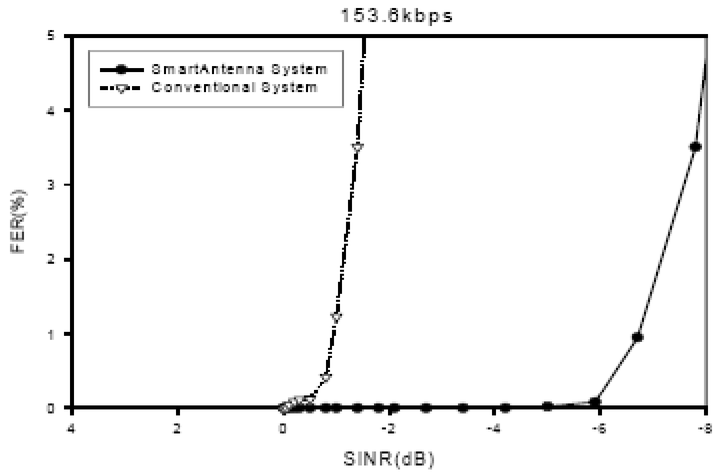


Figure 8: Uplink Data Channel Performance (153.6kbps)

Table 4: Performance enhancement for FER = 1% (9.6 kbps)

	Conventional System	Smart Antenna System	Enhancement
SINR	-10.6dB	-16.4dB	5.8dB

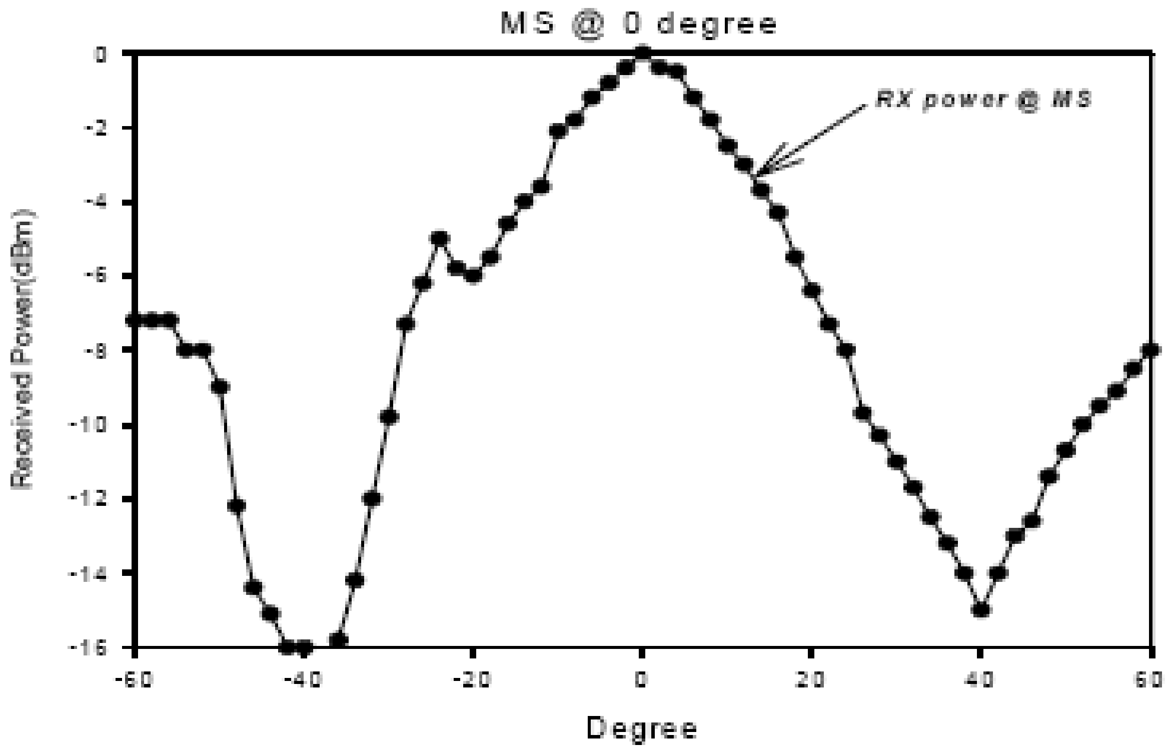
Table 5: Performance enhancement for FER = 1% (153.6kbp)

	Conventional System	Smart Antenna System	Enhancement
SINR	-1.1dB	-6.6dB	5.5dB

Hanyang University HY-SDR RESEARCH CENTER

4.2 Downlink performance analysis

❖ Downlink beam pattern



5. Conclusions

- The SABS APIs in this paper are based on a philosophy that the software architecture should be layered for supporting the extension of application programs while the hardware architecture should be partitioned for supporting the modularization of hardware resources.
- The layered and partitioned structure is the key aspect for the APIs to be able to accommodate various communication standards utilizing a single hardware platform in real-time.
- Interface between hardware modules, software modules, and hardware and software modules should also be clearly defined for implementing a desired application successfully.
- The proposed SABS APIs has been implemented and the performance of the implemented SABS system has been measured in a practical signal environment using commercial handsets operating in CDMA2000 1X circumstances.
- When the required FER is set to 1%(9.6kbps), compared to the conventional 2-antenna diversity base station system, SABS provides an enhancement of about 5.8dB, which means a 3-4x increase in capacity can be obtained with the SABS system consisting of 6 receiving antenna elements.
- Through the experimental tests, the suitability of the proposed SABS APIs has been verified.
- It is noteworthy that the SABS system implemented in accordance with the proposed open architecture exhibits a conspicuous superiority in its performance compared to the conventional 2-antenna diversity base station system.