Journal of KIIT. Vol. 18, No. 10, pp. 133-138, Oct. 31, 2020. pISSN 1598-8619, eISSN 2093-7571
http://dx.doi.org/10.14801/jkiit.2020.18.10.133

FPGA 검증을 위해 SDK의 Memory Map을 이용한 데이터 자동화 시스템 구현

박상욱*, 강봉순**

Implementation of Data Automation System using SDK's Memory Map for FPGA Verification

Sangwook Park*, Bongsoon Kang**

This paper was supported by research funds from Dong-A University.

요 약

영상처리는 가시적으로 많은 정보를 얻을 수 있으므로 자율주행, 얼굴인식 등 다양한 연구가 진행되고 있다. 따라서 영상처리는 회로 수정이 쉽고 빠른 검증이 가능한 비메모리 반도체인 FPGA(Field Programmable Gate Array)를 많이 사용한다. 본 논문에서는 Xilinx사의 Zynq-7000 ZC706 보드를 사용하며 PC와 통신을 위해 SDK를 사용한다. SDK는 보드에 사용될 알고리즘이 바뀔 경우, 새로운 알고리즘의 파라미터를 위해 SDK의 환경을 수정해야 한다. 따라서 빠른 검증을 위해 알고리즘이 바뀌더라도 사용될 파라미터를 자동으로 인식하고 원활한 통신을 위한 SDK의 자동화 시스템을 제안한다. 또한, 알고리즘이 최대 8개까지 사용 가능하도록 자동화를 위한 규칙화를 제안한다.

Abstract

Since image processing can obtain a lot of information visually, various studies such as autonomous driving and face recognition are being conducted. Therefore, image processing uses a lot of FPGA (Field Programmable Gate Array), which is a non-memory semiconductor that allows easy circuit modification and quick verification. In this paper, we use Xilinx's Zynq-7000 ZC706 board and SDK for communication with PC. When the algorithm to be used on the board is changed, the SDK environment must be modified for the parameters of the new algorithm. Therefore, for quick verification, even if the algorithm is changed, the parameter to be used is automatically recognized and an automated system of SDK is proposed for smooth communication. In addition, we propose regularization for automation so that up to 8 algorithms can be used.

Keywords
FPGA, memory map, SDK, data automation

* 동아대학교 전자공학과
- ORCID: https://orcid.org/0000-0002-0226-6799
** 동아대학교 전자공학과 교수(교신저자)
- ORCID: http://orcid.org/0000-0001-6716-5799

· Received: Aug. 19, 2020, Revised: Oct. 07, 2020, Accepted: Oct. 10, 2020
· Corresponding Author: Bongsoon Kang
Dept. of Electronic Engineering, Dong-A University, 37 Nakdong-Daero 550 beon-gil, Saha-gu, Busan, Korea.
Tel.: +82-51-200-7703, Email: bongsoon@dau.ac.kr

Ⅰ. 서 론

최근 영상처리 기술의 발달로 이미지에서 정보를 얻을 수 있는 자율주행, 얼굴인식, 인공지능 등 다양한 연구가 진행되고 있다[1]-[5]. 영상처리 기술을 하드웨어에 적용해 빠르게 검증하기 위해 FPGA (Field Programmable Gate Array)를 택하는 시스템이 증가하고 있다[6]-[10]. FPGA를 이용한 영상처리는 회로 수정이 쉽고 빠른 검증이 가능하다. 하지만 새로운 알고리즘 및 기존 알고리즘의 파라미터가 추가될 경우, SDK의 환경을 새로 구축해야 한다. 환경을 구축하기 위해선 입력의 RGB 채널의 정보와 알고리즘에 사용되는 파라미터의 개수를 파악하여 메모리 주소를 사용자가 직접 할당해야 한다. 만약 파라미터 및 RGB 채널의 주소를 SDK에서 일치시켜주지 않으면 출력이 나오지 않거나 색상이 왜곡되는 오류를 발생시킨다. 또한, 사용자마다 주소를 다르게 할당하면 메모리 주소의 중복 사용으로 인한 오류 및 비효율적 주소 할당이 발생하게 된다.

그림 1은 FPGA를 구동시키기 위한 과정을 순서도로 나타냈다. 첫 번째로, 알고리즘을 위한 Verilog 언어의 User Logic을 생성한다. User Logic은 입력 RGB, 알고리즘 파라미터를 받아 영상처리 후 출력 RGB를 전달하는 부분이다. 두 번째로 Vivado를 통해 IP(Information Provider)를 생성한다. 세 번째로, 생성된 IP는 SDK(Software Development Kit)를 통해 SoC 보드에 업로드되며 SDK에서는 Platform과 SoC 보드 간의 통신을 담당한다. 마지막으로 C++ MFC인 Platform을 이용해 입력 RGB와 알고리즘 파라미터는 CMD(Command) Address로 데이터가 보드로 전달된다.

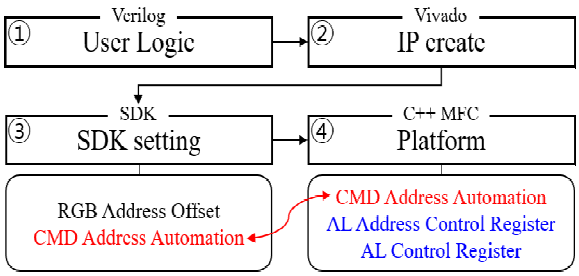

그림 1. SoC System 구동 과정
Fig. 1. SoC system drive processing

그림 2는 Platform과 FPGA 보드 간의 통신 과정을 나타냈다. 영상처리를 위한 입력 RGB 정보와 알고리즘에 사용될 파라미터 값들은 외부 PC에서 입력된다. 입력된 정보들은 USB통신으로 SDK의 메모리 주소에 데이터를 저장하여 값을 공유한다.

본 논문에서는 FPGA 검증 소요 시간을 단축하기 위해 새로운 알고리즘 및 기존 알고리즘의 파라미터가 추가되더라도 자동으로 데이터를 처리하는 시스템을 제안한다. 또한, 사용자마다 메모리 주소를 다르게 할당하는 방식이 아닌 고정된 주소를 사용하여 오류 발생 빈도를 줄여 FPGA 검증 소요 시간을 단축한다. 제안하는 시스템은 규칙화를 통해 메모리에서 사용되는 알고리즘의 파라미터는 최대 32개의 주소를 할당할 수 있다. 이런 알고리즘은 최대 8개가 사용할 수 있도록 256개의 주소를 할당했다. 입력 RGB와 알고리즘 파라미터 정보는 Microsoft 사에서 제공하는 C++ MFC(Platform)를 사용하며, 사용된 보드는 Xilinx사의 Zynq-7000 ZC706을 사용했다. 이때 Zynq 보드로 전달하기 위해 사용되는 tool은 Vivado SDK를 사용한다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 Platform과 SDK 간의 메모리 주소에 관한 내용을 소개하고 메모리 주소번지를 자동화하여 공유하는 방법을 설명한다. Ⅲ장에서는 실제 Xilinx사의 Zynq-7000 ZC706 보드와 Microsoft사의 C++ MFC 간의 데이터 송, 수신을 UART 통신으로 데이터 및 자동화된 주소번지의 할당된 파라미터 값을 제시한다. 또한, 실제 알고리즘이 적용된 Platform의 입출력을 제시한다. 마지막으로 Ⅳ장에서는 결론을 맺는다.

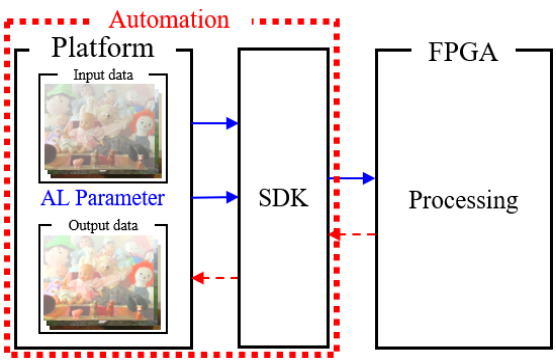

그림 2. Platform과 Zynq 보드의 전송 과정
Fig. 2. Transfer process between platform and Zynq board

II. Platform과 SDK간의 주소 자동화

2.1 SDK Memory Map 규칙화

Platform과 SDK 간의 입력 데이터, 출력 데이터, 알고리즘 파라미터 그리고 processing에 필요한 Flag 신호 등을 메모리 주소에 저장하여 데이터를 전달한다. 그림 3은 SDK Memory Map을 나타냈다. 한 주소에 4byte가 사용되어 4byte 단위로 주소 이름이 변경된다. 예를 들어, 0번지는 0~3byte를 의미하고 1번지는 4~7byte를 의미한다. CMD는 processing에 필요한 고정된 Flag 신호, 알고리즘 파라미터, input RGB address, output RGB address 등이 사용된다. Platform에서 1 frame이 전달됐음을 알려주는 end of frame 신호가 들어간다.

그리고 R_in, G_in, B_in은 실제 입력된 이미지의 data들이고, R_out, G_out, B_out은 실제 출력되는 이미지의 data들이다. FRAME_NUM_OFFSET은 입력 이미지의 width와 height 정보를 받아 이미지 데이터 크기를 할당하는 수식이다. FRAME_OUT_NUM_OFFSET은 출력 이미지의 크기를 할당하는 수식이다.

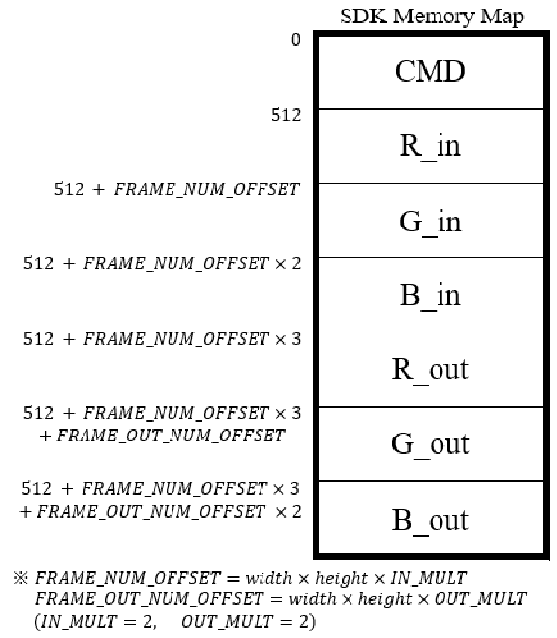

그림 3. SDK Memory Map 구조
Fig. 3. SDK memory map structure

그림 4는 SDK memory map의 CMD를 나타냈다. 0과 2번지는 고정된 값이 Platform에서 SDK로 전달된다. 1번지는 입력 이미지의 width와 height 값을 전달한다. 4, 5번지는 알고리즘 파라미터가 자동으로 주소를 찾을 수 있도록 전달되는 값이다. 입력 R, G, B 주소는 각각 8, 16, 24번지를 사용한다. 입력 이미지를 최대 8개를 사용할 수 있도록 규칙화 하였다. 예를 들어, 입력 이미지가 2개를 사용할 때 입력 R의 주소는 8과 9번지를 사용하고 G는 16, 17번지를 사용하며 B는 24, 25번지를 사용한다. 출력 R, G, B의 주소는 각각 32, 40, 48번지를 사용하며 출력 이미지도 최대 8개를 사용할 수 있도록 하였다. 알고리즘 파라미터는 64번지부터 32개의 주소를 할당하여 최대 8개의 알고리즘을 사용할 수 있도록 하였다. 따라서 알고리즘 파라미터가 사용되는 주소는 64번지부터 319번지까지이다. 마지막으로 1 frame이 모두 전달됐음을 알려주는 end_of_frame 신호가 511번지에 들어간다.

따라서 Platform과 Zynq 보드 간의 processing 과정은 다음과 같다. 첫 번째로, Platform은 입력 RGB와 파라미터를 SDK Memory Map 주소에 값을 입력한다.

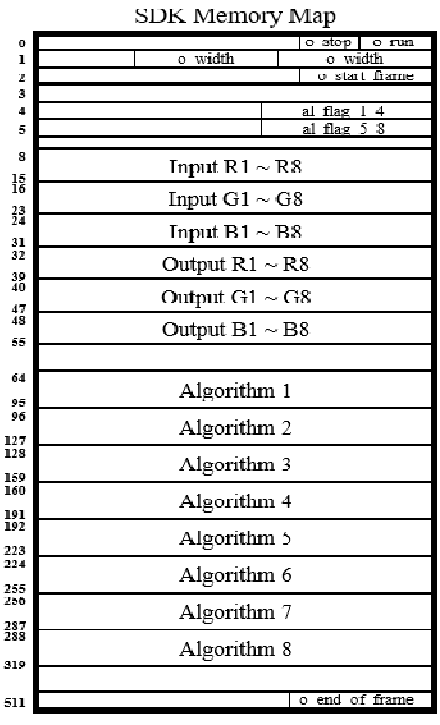

그림 4. Memory Map의 CMD 영역
Fig. 4. CMD area of memory map

두 번째로, SDK에서 각 해당하는 주소의 값을 Zynq 보드로 전달된다. 마지막으로, 데이터를 전달받은 보드에서 processing이 끝나면 출력 RGB의 주소에 값을 할당하고 Platform에서는 출력 데이터를 받아 Display 한다.

2.2 Platform과 SDK간의 주소 자동화

2.1에서는 SDK Memory Map을 효율적으로 사용하기 위해 규칙화하였다. 규칙화를 통해 메모리를 효율적으로 사용할 수 있도록 제안하였다. 하지만 Platform으로부터 파라미터를 SDK로 전달하기 위해 사용자는 직접 주소를 할당하고 값을 전달해야 하므로 개발시간이 지연된다. 2개 이상의 알고리즘이 사용될 경우 개별적으로 주소를 할당해야 하므로 복잡성이 증가한다.

따라서 그림 5와 같이 자동화를 제안한다. 자동화는 주소를 먼저 할당하여 matrix를 생성한 뒤 matrix에 알고리즘의 파라미터만 입력하도록 하였다.

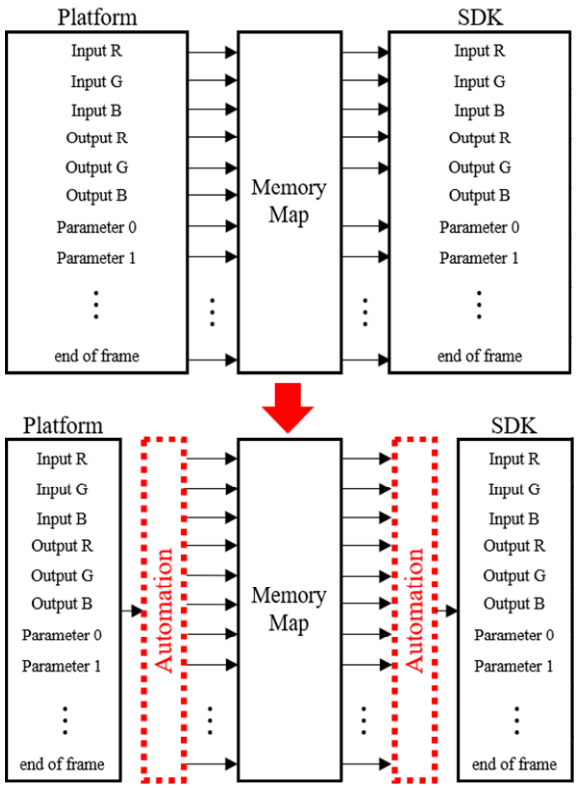

그림 5. 자동화된 SDK Memory Map
Fig. 5. Automated SDK memory map

사용자는 matrix에 입력된 파라미터는 Algorithm 1 ~ 8까지 원하는 주소를 MFC에서 선택할 수 있다. 선택한 주소의 신호는 memory map의 4, 5번지에 저장되고 SDK에서 이 신호를 이용해 주소를 자동으로 할당한다. 위 방법은 알고리즘이 2개 이상일 때, 효율적으로 사용 가능하다. 서로 다른 파라미터를 갖는 알고리즘은 정해진 matrix에 저장하면 자동화를 통해 정해진 주소에 할당되므로 FPGA를 활용하는 영상처리에서 개발시간을 단축할 수 있다.

III. 구현 결과

본 논문에서 제안한 방법을 확인하기 위해 한 개의 알고리즘을 Verilog-HDL로 설계하여 Xilinx사의 Zynq-7000 ZC706 보드에 업로드 후 실제 동작을 확인했다. 사용된 알고리즘은 CIE1931 색 좌표계를 이용한 색상보정 알고리즘을 사용했다[11].

그림 6은 SDK에 대해 기존 사용법과 자동화 시스템의 사용법을 나타냈다. 기존 사용법을 사용할 경우, RGB 정보와 알고리즘의 파라미터를 위해 SDK를 수정하게 되면 오류가 발생하고 다시 수정을 반복하게 된다. 예를 들어, 파라미터 값을 의도하지 않은 Memory Map의 주소에 할당하면 1번 오류가 발생하면 시스템이 정상적으로 작동하지 않는다. 그리고 2번 오류가 발생하면 출력 영상에 색상 왜곡이 발생할 수 있으므로 SDK 환경 구축을 다시 해야 한다. 하지만 자동화 시스템을 사용할 경우, 불필요한 수정을 줄이고 1번 또는 2번과 같은 오류가 발생하지 않으므로 빠른 FPGA 검증이 가능하다.

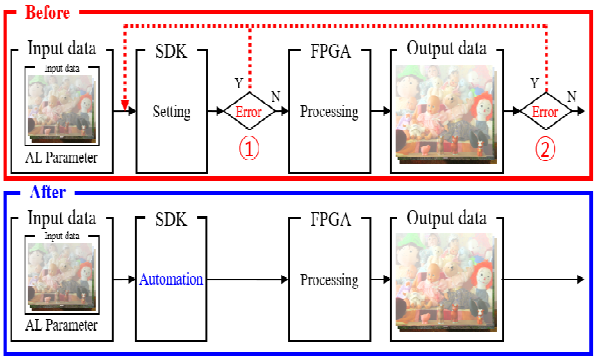

그림 6. 기존 방법과 자동화 시스템의 비교
Fig. 6. Comparison of traditional methods and automation systems

그림 7은 알고리즘을 적용한 Platform과 UART 통신 결과를 나타냈다. UART 통신에서 1부터 9까지는 파라미터의 hex 값을 나타냈고 입력, 출력 RGB의 첫 번째 픽셀값을 나타냈다. 네모박스로 표시된 영역은 알고리즘의 주소 선택 신호이다. 이 신호들은 SDK에서 알고리즘 주소 위치를 판별할 때 사용된다. 제안한 규칙화 및 자동화를 확인하기 위해 1 Frame의 RGB 데이터와 알고리즘 파라미터를 platform을 통해 SDK memory map으로 입력하였다. SDK에서는 입력받은 알고리즘 선택 신호를 통해 자동으로 주소번지를 계산하여 ZC706 보드로 전달한다. 이후 processing을 거쳐 출력 RGB가 최종적으로 Platform으로 전달되어 출력된다. 보드에서 출력된 RGB의 정보와 Matlab의 fixed point로 작성된 동일한 알고리즘의 오차가 0인 것을 확인하였다.

또한, 실제 2개의 알고리즘이 적용된 Platform으로 테스트하여 정상적인 출력을 확인하였다. 1개의 알고리즘 파라미터는 SDK memory map의 Algorithm 1 주소번지에 입력하고 다른 1개의 알고리즘 파라미터는 Algorithm 2 주소번지에 입력하였다. 입력된 정보들은 MFC에서 SDK memory map에 할당된다. 할당된 정보는 SDK를 통해 Zynq 보드로 전달되고 processing을 거친 출력 RGB가 정상적으로 나타나는 것을 확인하였다. 2개를 사용한 알고리즘도 Matlab의 fixed point와 오차가 0인 것을 확인하였다.

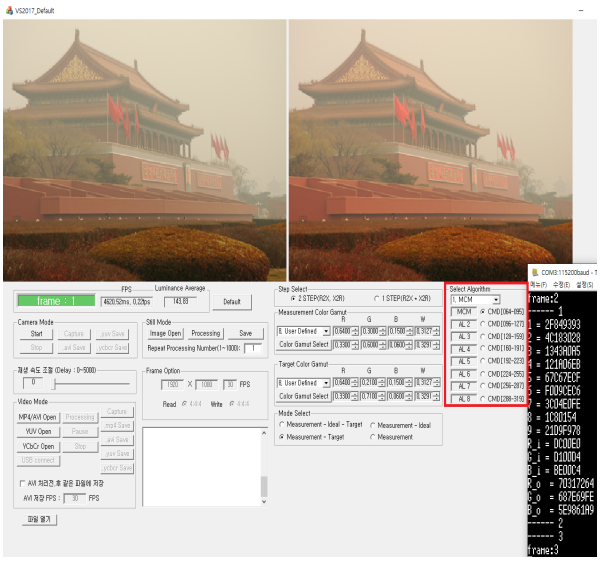

그림 7. Platform 동작과 UART 통신 결과
Fig. 7. Platform operation and UART transfer result

위 결과에서 SDK가 정상적으로 동작하지 않는다면 PC에서 SDK를 통해 정확한 정보를 Zynq 보드로 전달할 수 없다. 따라서 출력 RGB의 오차가 0인 것은 SDK의 자동화 시스템이 정상적으로 동작하는 것을 의미한다.

Ⅳ. 결 론

본 논문에서는 SDK와 Platform 간의 사용되는 memory map의 자동화 시스템을 제안하였다. 사용할 memory map에 할당될 입력 RGB, 출력 RGB 그리고 알고리즘 파라미터의 구역을 나눠 규칙화 하였다. 그리고 규칙화 된 memory map의 시작 주소번지를 이용하여 자동화를 구현하였다. 구현된 기능을 검증하기 위해 Xilinx사의 Zynq-7000 ZC706 보드와 Microsoft사의 Visual Studio의 C++ MFC를 사용하였다. MFC와 ZC706간의 통신을 위해 Vivado의 SDK를 사용하였다. MFC에서 입력 RGB와 알고리즘 파라미터를 자동화된 SDK memory map에 할당하면 Zynq 보드에서 영상처리가 이루어지고 출력 RGB가 MFC로 전달된다. 검증할 때 사용한 알고리즘은 CIE1931 색 좌표계를 이용한 색상보정을 사용했으며 출력이 정상적으로 나오는 것을 확인하였다. 또한, 보드를 통해 출력된 RGB 정보와 Matlab의 fixed point로 작성된 동일한 알고리즘의 오차가 0인 것을 확인하였다. 앞의 내용을 근거로 자동화 기능이 정상적으로 구현되었다는 것을 증명한다. 또한, 2개 이상의 알고리즘이 적용된 Platform에서 정상적인 출력을 확인하였고 Matlab의 fixed point와 오차가 0인 것을 통해 SDK의 자동화 시스템이 정상적으로 동작하는 것을 확인하였다.

References

[1] D. Ngo, S. Lee, and B. Kang, "Robust Single-Image Haze Removal Using Optimal Transmission Map and Adaptive Atmospheric Light", Remote Sens., Vol. 12, No. 14, Jul. 2020. DOI: https://doi.org/10.3390/rs12142233

[2] J. S. Oh, K. I. Lim, and J. H. Kim, "A Research of Obstacle Detection and Path Planning for Lane

Change of Autonomous Vehicle in Urban Environment", Journal of Institute of Control, Robotics and Systems, Vol. 21, No. 2, pp. 115-120, Feb. 2015.

[3] T. Okuyama, T. Gonsalves, and J. Upadhay, "Autonomous Driving System based on Deep Q Learnig", 2018 International Conference on Intelligent Autonomous Systems (ICoIAS), pp. 201-205, Oct. 2018. DOI: 10.1109/ICoIAS.2018.8494053.

[4] Y. Akbulut, A. Şengür, Ü. Budak, and S. Ekici, "Deep learning based face liveness detection in videos", 2017 International Artificial Intelligence and Data Processing Symposium (IDAP), pp. 1-4, Nov. 2017, DOI: 10.1109/IDAP.2017.8090202.

[5] S. G. Cho and W. Xu, "A Study on Person Re-Identification System using Enhanced RNN", Journal of IIBC, Vol. 17, No. 2, pp. 15-23, Apr. 2017.

[6] D. Ngo, S. Lee, Q. H. Nguyen, T. M. Ngo, G. D. Lee, and B. Kang, "Single Image Haze Removal from Image Enhancement Perspective for Real-Time Vision-Based Systems", Sensors, Vol. 20, No. 18, Sep. 2020. DOI: https://doi.org/10.3390/s20185170

[7] H. Jeong, C. Kang and Y. Jeong, "Implementation of Lane Departure Warning system on FPGA hardware Based on Images surrounding the Vehicle", Journal of Korea Institute Of Communication Sciences, pp. 467-468, Jan. 2014.

[8] D. Ngo, G. D. Lee, and B. S. Kang, "Improved color attenuation prior for single-image haze removal", Appl. Sci., Vol. 9, No. 19, 2019. DOI: https://doi.org/10.3390/app9194011

[9] Y. Jeon and Y. Kim, "Design and Implementation of 8K UHD Signal Generation System with Synchronized Multiple Split Outputs", The Journal of Korean Institute of Communications and Information Sciences, Vol. 45, No. 7, pp. 1269-1276, 2020. DOI: 10.7840/kics.2020.45.7.1269

[10] Q. H. Nguyen and B. Kang, "FPGA-based Haze Removal Architecture Using Multiple-exposure Fusion", Journal of JKIIT, Vol. 18 No. 5, pp. 85-90, May. 2020. DOI: 10.14801/jkiit.2020.18.5.85

[11] S. Lee, S. Park, and B. Kang, "Hardware implementation of CIE1931 color coordinate system transformation for color correction", Journal of IKEEE, Vol. 24, No. 2, pp. 502-506, Jun. 2020.

저자소개

박 상 욱 (Sangwook Park)

2019년 2월 : 동아대학교 전자공학과(공학사)
2019년 3월 ~ 현재 : 동아대학교 전자공학과(공학석사)
관심분야 : FPGA, 영상처리

강 봉 순 (Bongsoon Kang)

1985년 : 연세대학교 전자공학과(공학사)
1987년 : 미국 University of Pennsylvania 전기공학과(공학석사)
1990년 : 미국 Drexel University 전기 및 컴퓨터공학과(공학박사)
1989년 12월 ~ 1999 2월 : 삼성전자 반도체 수석연구원
1999년 3월 ~ 현재 : 동아대학교 전자공학과 교수
관심분야 : 영상신호처리, SoC설계 및 무선 통신