Accurate and Bandwidth Efficient Architecture for CNN-based Full-HD Super-Resolution

Chung-Yan Chih, Sih-Sian Wu, Jan P. Klopp, Liang-Gee Chen, *Fellow, IEEE*
DSPIC Lab, Department of Electrical Engineering
National Taiwan University, Taiwan
andychih23@gmail.com, benwu@video.ee.ntu.edu.tw, kloppjp@gmail.com, lgchen@ntu.edu.tw

***Abstract*—CNN-based super-resolution methods achieve great performance but come with high bandwidth requirements. This paper proposes a bandwidth efficient CNN-based architecture for super resolution supporting up to Full HD images at 60 fps. The bandwidth efficiency is achieved by layer fusion with data reuse scheduling and dynamic quantization. To maintain a high utilization of processing elements, the deconvolution operation is decomposed into a several convolution operations. The proposed architecture provides a 94.8 % bandwidth reduction. Compared to the software implementation, less than 1 % performance drop is induced in this VLSI implementation. The enables the proposed architecture to obtain state-of-the-art accuracy compared to existing super-resolution hardware implementations.**

***Index Terms*—Super-resolution, FSRCNN, deconvolution, VLSI architecture, bandwidth efficiency**

I. Introduction

Super-resolution (SR) is a technique generating high-resolution (HR) images from low-resolution (LR) ones by inferring the missing high frequency information. SR is adopted in many computer vision applications but its ill-posed nature makes it a challenging problem. Although existing CNN-based SR methods [3] [7] [6] [8] [4] provide superior performance over conventional dictionary-based methods, they induce vast bandwidth demands, rendering them impractical for hardware implementations. The additive deconvolution operation in the last layer leads to an irregular computation pattern which is not hardware friendly.

The contribution of this work is threefold:

1) The first CNN-based super resolution VLSI architecture supporting Full-HD at 60FPS is presented.
2) A bandwidth-efficient CNN-based architecture is proposed with support for upscaling factors $\times 2$, $\times 3$ and $\times 4$.
3) The VLSI implementation achieves state-of-the-art performance among existing hardware SR implementations

The remainder of this paper is organized as follows: Section II summarizes related works. Our proposed method is presented in Section III. The experimental results and discussion are found in Section IV. Finally, the paper is concluded in Section V.

II. Related Works

Many different approaches for the super resolution problems have been proposed in the literature. Among them, the self-exemplars [5] method can obtain great performance by finding similar patches in different scales and applying a suitable affine transform. Learning based methods have dominated the SR domain since the A+ algorithm [14] was proposed, it transfers a target region into a sparse representation from which the super resolved patch is recovered. After the initial work, SRCNN [3], CNN-based methods [7] [6] [8] [4] have provided a significant improvement over other methods. Among them, FSRCNN [4] can significantly reduce the computation of the neural network while maintaining a good performance. SR hardware implementations can be categorized into non-learning methods and learning-based ones. Non-learning methods are [10] [2] [13] and [11]. Bowen et.al. [2] proposed a hardware implementation based on the iterative weighted mean algorithm. An SR hardware architecture based on the image registration technique has been proposed by Redlich et.al. in [10]. Another SR hardware implementation based on the iterative back projection SR algorithm is presented in [11]. They proposed a highly parallel and pipelined implementation for iterative back projection super-resolution algorithm. All above works are based on non-learning algorithms and outperformed by learning-based SR works.

Yang et.al. [15] propose a learning based super-resolution architecture using sparse dictionary representation without frame-buffer based on the A+ algorithm [14]. A hardware architecture based on SRCNN [3] is proposed by Manabe et.al. [9] which is the first CNN-based SR hardware implementation. They use horizontal and vertical flips to network input images to replace pre-enlargement techniques preventing information loss and enables the network to utilize the input image size. None of these works can provide Full-HD resolution output at 60 fps. Our work can provide more than one scaling factor and achieve the best accuracy among those hardware implementations.

III. Proposed Architecture

SRCNN [3] is the first work adopting CNN to solve the SR problem directly. Input images are up-scaled to the target resolution before they enter the network. In contrast to SRCNN, FSRCNN [4] runs the network on the low resolution image and only upscales in the last layer, at a negligible drop in quality. The upscaling is realized by a deconvolution operation in the last layer to obtain the desired spatial extend. Following notation and definition in [4], the relationship between high-resolution (HR) network and low-resolution (LR) networks

978-1-5386-4881-0/18/$31.00 ©2018 IEEE

is shown in Fig. 1. Without the pre-enlargement, feature maps in each following layers preserve the original size and simplify the computation. Although FSRCNN can reduce the computational complexity, it is still impractical since its vast bandwidth requirements. The deconvolution operation in the last layer yields an irregular, hardware-unfriendly computation pattern.

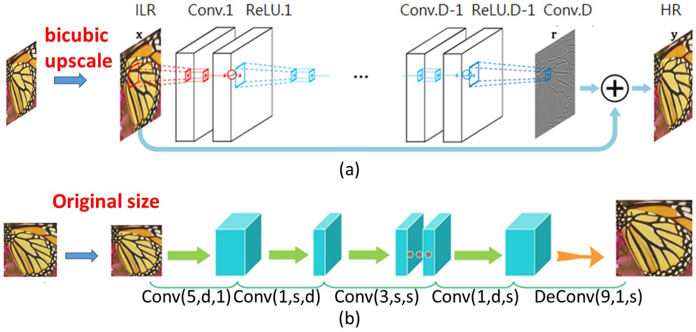


Fig. 1: CNN-based super resolution network architecture for (a) VDSR and (b) FSRCNN.

Since CNN-based methods introduce a vast bandwidth requirement, we employ two techniques to reduce the required bandwidth to a level that is practical for a hardware implementation. To further improve the hardware efficiency, the fractional convolution operation (deconvolution) is transformed into multiple convolutions.

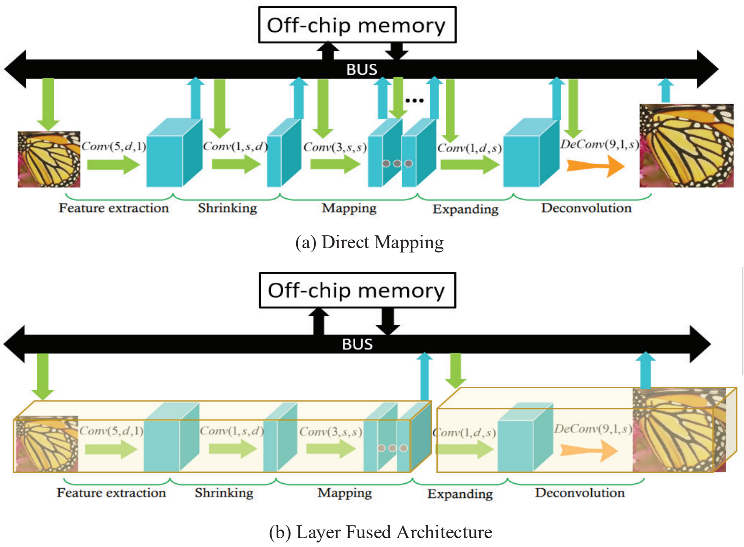
A. Bandwidth Reduction design

The total bandwidth requirement is presented in Eq. (1)

$$\text{Bandwidth} = \sum_{l \in L} \left(D^l \times \text{length}_D^l + W^l \times \text{length}_W^l \times \gamma \right). \quad (1)$$

Here, L represents the number of layers, length_D^l and length_W^l represent the bit length of data and weights in layer l, respectively. D^l and W^l denote the number of data and weight elements of layer l. γ shows the utilization of the data. If it is equals to 1, no data is reloaded, all data is reused in an optimal fashion. In this paper, two techniques are adopted to reduce the bandwidth. First, layer fusion is adopted to maximize data reuse by reducing the I/O between two layers l and $l+1$. Second, dynamic quantization is employed to reduce the bit-length of activation and weight data which are length_D^l and length_W^l.

The layer fusion accelerator [1] is proposed to merge layers for a general CNN processor with the aim of reducing input and output bandwidth. The architecture of the proposed multi-layer computation using this technique is shown in Fig. 2. The direct implementation would create huge bandwidth demands from accessing off-chip memory as shown in Fig. 2(a). The layer-fused architecture can eliminate the traffic between layers via storing the temporal data on chip as Fig. 2(b). To further determine the optimized configuration for our scenario, we analyze which layers to fuse and how to choose tile size as shown in Fig. 3. As a result, we choose $6-2$ which means fusing the first 6 layers and last two layers separately since it leads to the smallest on-chip memory requirement at an acceptable bandwidth. Under the 128-bit I/O and 200 MHz condition the bandwidth constraint is 3.2 GB/s as presented in Fig. 3 and the proposed architecture can satisfy the constraint which renders this CNN-based super-resolution algorithm VLSI implementation practicable.



(a) Direct Mapping

(b) Layer Fused Architecture

Fig. 2: Architecture comparison for layer fusion: (a) without layer fused (b) and the architecture with layer fused.

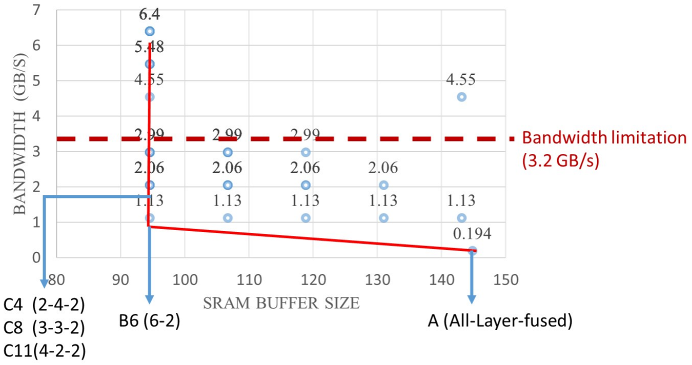


Fig. 3: Layer fused condition analysis. The $6-2$ configuration is adopted because it requires less on-chip memory.

Dynamic quantization selects appropriate quantization factors for each layer adopted in the proposed architecture. According to experiment results, 10-bit word length is chosen for both data, D^l, and weights, W^l. Further results and discussion are found in Section IV

B. De-convolution architecture design

Each layer except for the last one consists of a standard convolution operation. To further utilize processing elements,

978-1-5386-4881-0/18/$31.00 ©2018 IEEE

the deconvolution is executed using convolution hardware. The concept is to use multi-size convolution filters to replace the original de-convolution filter so that the processing elements can support the de-convolution computation as well. We can easily find that the output of de-convolution is the dot product of the adjacent pixel with corresponding weights, and it is the same computation as the convolution. The original de-convolution filter is composed into corresponding convolution kernels with different coefficients. Recently, we found that the concept is similar to the one proposed by Shi et.al. [12]. The example of a ×2 deconvolution remapped to convolution is shown in Fig. 4. It is worth to mention that we decrease the kernel size while scaling is increased. This makes ×2, ×3 and ×4 require similar amounts of computation.

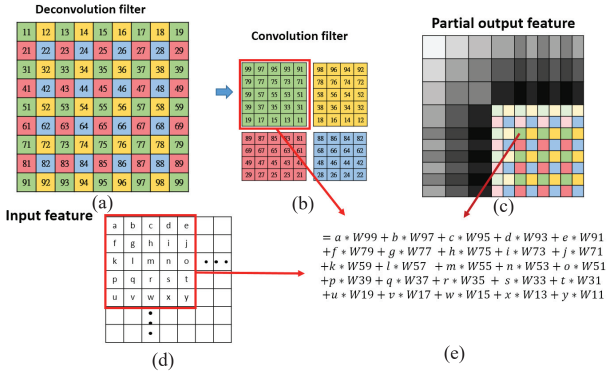

Deconvolution filter
Convolution filter
Partial output feature
Input feature
(a)
(b)
(c)
(d)
(e)
$= a*W99 + b*W97 + c*W95 + d*W93 + e*W91$
$+f*W79 + g*W77 + h*W75 + i*W73 + j*W71$
$+k*W59 + l*W57 + m*W55 + n*W53 + o*W51$
$+p*W39 + q*W37 + r*W35 + s*W33 + t*W31$
$+u*W19 + v*W17 + w*W15 + x*W13 + y*W11$

Fig. 4: An example of remapping ×2 de-convolution processing to a convolution like process. (a) The original de-convolution kernel, (b) remapped convolution-like kernel, (c) partial output feature, (d) input feature, and (e) weighted summation of the pixel.

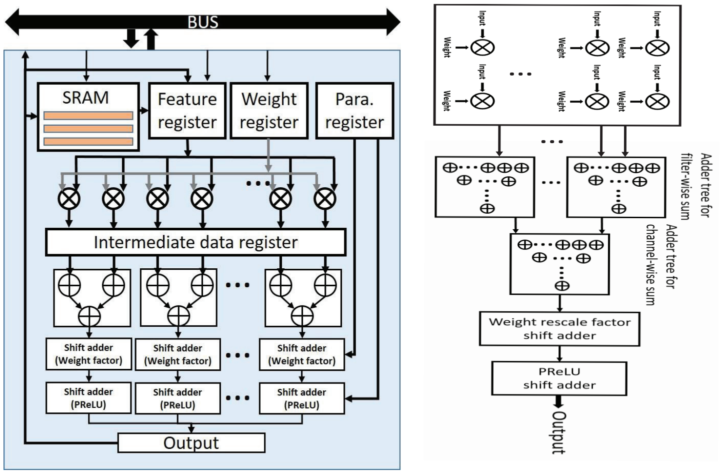
C. The proposed architecture

The proposed architecture is presented in Fig. 5. The computation unit is pipelined into two stages to further improve the throughput. The first stage is the multiplier array. The second stage is consist of the summation by adder trees and the dynamic quantization by shift adders. In the end, we realize the PReLU operation by another group of shift adders.

IV. Experimental Results

We implement our design in TSMC's 40 nm technology. To achieve a realistic implementation, the proposed architecture uses of two techniques to reduce bandwidth requirement. Results for different quantization activation and weight word length are shown in Fig. 6. As the results indicate, accuracy is saturated when a word length of 10 bit is reached, hence we use 10 bits for both activations and weights.

Effects of bandwidth and accuracy of each technique are listed in Table I. The complete proposed architecture achieves a bandwidth reduction of 94.8%. As Table I shows, the accuracy drop results only from the quantization and the accuracy degradations for both datasets are below 1%.


BUS
SRAM
Feature register
Weight register
Para. register
Intermediate data register
Shift adder (Weight factor)
Shift adder (PReLU)
Output
Input
Weight
Adder tree for filter-wise sum
Adder tree for channel-wise sum
Weight rescale factor shift adder
PReLU shift adder
Output

(a) (b)

Fig. 5: (a)The proposed architecture and (b) the process of each operation.

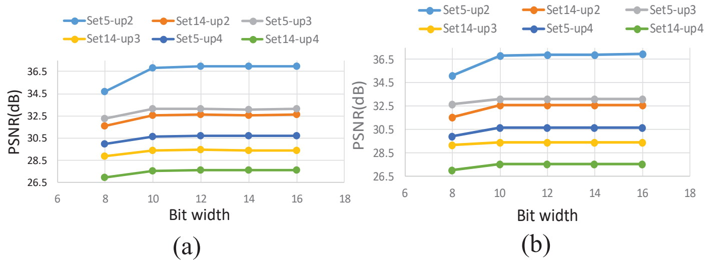


Fig. 6: Quantization bits and quality with (a) activation and (b) weight data.

The specification of the proposed architecture in comparison with others is listed in Table III. The comparison images are shown in Fig. 7. FSRCNN-RTL outperforms A+ [14] and SRCNN [3] algorithms in high frequency regions and as well as in terms of PSNR. Furthermore, the proposed FSRCNN-RTL is more accurate than the implementations of Yang [15] and Manabe [9] since both architectures suffer the accuracy degradation from their quantization steps.

V. Conclusion

An accurate and bandwidth efficient architecture for super-resolution CNN-based supporting ×2, ×3 and ×4 upscaling is proposed in this paper. Modified layer fusion and dynamic quantization are adopted to reduce bandwidth. The bandwidth

TABLE I: Bandwidth analysis for separate techniques.

Techniques	BW. (GB/s)	Red. (%)	Set5		Set14	
			PSNR	(%)	PSNR	(%)
Baseline Design	20.22	0	37.0	0	32.63	0
Only Layer Fusion	1.68	91.7	37.0	0	32.63	0
Only Quantization	12.64	37.5	36.67	0.9	32.39	0.7
Proposed	1.05	94.8	36.67	0.9	32.39	0.7

978-1-5386-4881-0/18/$31.00 ©2018 IEEE

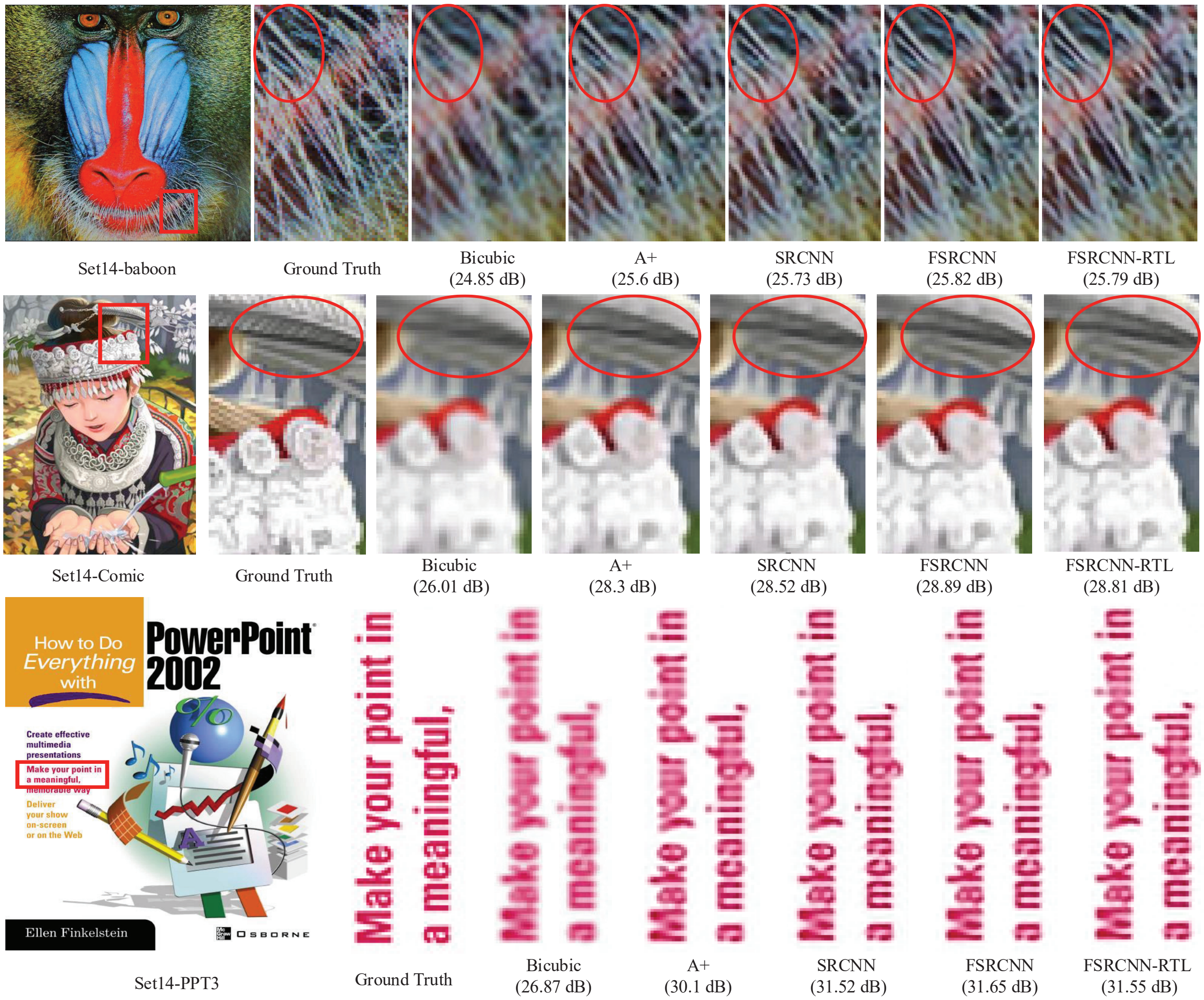


Fig. 7: Comparison figures.

TABLE II: The accuracy comparison of each HW implemenetations.

Accuracy (dB)	×2		×3		×4	
	Set 5	Set 14	Set 5	Set 14	Set 5	Set 14
Manabe	NA	NA	NA	NA	NA	NA
Liu et.a1.	33.83	29.77	NA	NA	NA	NA
FSRCNN	37	32.63	33.16	29.43	30.71	27.59
FSRCNN-RTL	36.67	32.39	32.99	29.33	30.55	27.48
Quality drop	0.33	0.24	0.17	0.1	0.16	0.11
Drop Ratio (%)	0.9	0.7	0.5	0.3	0.5	0.4

TABLE III: Hardware implementations comparison.

	Manabe [9]	Yang [15]	Proposed
SR Algorithm	SRCNN	A+	FSRCNN
Technology	Virtex UltraScale XCVU095	TSMC 90 nm	TSMC 40 nm
Frequency (MHz)	133	148	200
Gate Count (k)	NA	**2253**	3368
Input Size	960 × 540	960 × 540	960 × 540
Frame rate (fps)	48	**60**	**75,75,62**
Scale Factor	2	2	**2,3,4**
PSNR (Set 5 ×2)	NA	33.83	**36.67**
PSNR (Set 14 ×2)	NA	29.77	**32.39**
On-chip Mem. (kB)	NA	231.74	**53.4**

is reduced to a level that allows an efficient hardware implementation using three techniques, the effect of each of them is analyzed. In total, a 94.8% bandwidth reduction is achieved. Furthermore, this work achieves superior quality compared to previous SR VLSI implementations.

References

[1] Manoj Alwani, Han Chen, Michael Ferdman, and Peter Milder. Fused-layer cnn accelerators. In *Microarchitecture (MICRO), 2016 49th Annual IEEE/ACM International Symposium on*, pages 1–12. IEEE, 2016.

[2] Oliver Bowen and Christos-Savvas Bouganis. Real-time image super resolution using an fpga. In *Field Programmable Logic and Applications, 2008. FPL 2008. International Conference on*, pages 89–94. IEEE, 2008.

[3] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In *European Conference on Computer Vision*, pages 184–199. Springer, 2014.

[4] Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the super-resolution convolutional neural network. In *European Conference on Computer Vision*, pages 391–407. Springer, 2016.

[5] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5197–5206, 2015.

[6] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1646–1654, 2016.

[7] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Deeply-recursive convolutional network for image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1637–1645, 2016.

[8] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. *arXiv preprint arXiv:1609.04802*, 2016.

[9] T. Manabe, Y. Shibata, and K. Oguri. Fpga implementation of a real-time super-resolution system using a convolutional neural network. In *2016 International Conference on Field-Programmable Technology (FPT)*, pages 249–252, Dec 2016.

[10] R. Redlich, L. Araneda, A. Saavedra, and M. Figueroa. An embedded hardware architecture for real-time super-resolution in infrared cameras. In *2016 Euromicro Conference on Digital System Design (DSD)*, pages 184–191, Aug 2016.

[11] Kerem Seyid, Sebastien Blanc, and Yusuf Leblebici. Hardware implementation of real-time multiple frame super-resolution. *IEEE/IFIP International Conference on VLSI and System-on-Chip, VLSI-SoC*, 2015-Octob:219–224, 2015.

[12] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1874–1883, 2016.

[13] T. Szydzik, G. M. Callico, and A. Nunez. Efficient fpga implementation of a high-quality super-resolution algorithm with real-time performance. *IEEE Transactions on Consumer Electronics*, 57(2):664–672, May 2011.

[14] Radu Timofte, Vincent De Smet, and Luc Van Gool. A+: Adjusted anchored neighborhood regression for fast super-resolution. In *Asian Conference on Computer Vision*, pages 111–126. Springer, 2014.

[15] Ming-Che Yang, Kuan-Ling Liu, and Shao-Yi Chien. A real-time fhd learning-based super-resolution system without a frame buffer. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 2017.

978-1-5386-4881-0/18/$31.00 ©2018 IEEE