A Cost-Optimized Heterogeneous FPGA Architecture for Non-iterative Hologram Generation

DAOMING DONG, YOUCHAO WANG[*], ANDREW KADIS AND TIMOTHY D. WILKINSON

Centre for Molecular Materials, Photonics and Electronics, Department of Engineering, University of Cambridge
[*]*yw479@cam.ac.uk*

Abstract: The generation of computer-generated holograms (CGH) require a significant amount of computational power. To accelerate the process, highly parallel field-programmable gate arrays (FPGA) are deemed to be a promising computing platform to implement non-iterative hologram generation algorithms. In this paper, we present a cost-optimized heterogeneous FPGA architecture based on one-step phase retrieval (OSPR) algorithm for CGH generation. The results indicate that our hardware implementation is 2.5× faster than the equivalent software implementation on a personal computer with a high-end multi-core CPU. Trade-offs between cost and performance have been demonstrated, and we have shown that the proposed heterogeneous architecture can be used in a compact display system that is cost- and size-optimized.

1. Introduction

Holography has been receiving growing interests in the area of display technology [1]. Holographic displays outperform conventional 2D displays by preserving the amplitude and phase information of an optical field. Moreover, holographic displays operate in a manner that diffracts the incoming light, which shall enable itself a power-efficient technology [2] as compared to the traditional display technologies that instead block the light source.

Computer-generated holography advances the traditional technique pioneered by Dannis Gabor and simulates the recording process on digital computers and reconstruct the replay field using the optical setup same as the conventional holography. Computer-generated holograms (CGHs) can be calculated by implementing algorithms based on point light source (PLS) summation, triangular mesh, Fourier transform, etc. [3].

For 2D projection applications, it is of the interest to use Fourier based algorithms as the relationship between the replay field in the Fraunhofer region and the hologram can be described by a 2D discrete Fourier transform (DFT) [1]. Moreover, noise and speckles can be suppressed when time-division multiplexing techniques are combined with Fourier transform-based algorithms [4].

Direct binary search (DBS) and Gerchberg-Saxton (GS) are two widely used iterative Fourier-based algorithms to generate CGHs [5,6]. However, both algorithms are prone to reaching local minima which would prevent the algorithms from further enhancing the quality of the replay field. Improvements were made later on by proposing variant algorithms such as simulated-annealing (SA) and Liu-Taghizadeh algorithms [7,8].

In this paper, we propose a cost-optimized heterogeneous FPGA hardware architecture that utilizes fixed-point arithmetics to generate binary CGHs using one-step phase retrieval (OSPR) algorithm. We use Verilog hardware description language (HDL) to design the circuit at the register transfer level (RTL) and then map the synthesised bitstream onto an Intel Cyclone V 5CSXFC6D6F31C6N FPGA for dynamic online analysis. Numerical and experimental reconstructions are conducted and reported. We also compare the performance of our design to a personal computer running a high-end Intel i7 processor.

2. Related work

Despite the progress being made in recent years, the computational power required for CGH calculations in real-time still remains relatively high and burdensome. One potential solution is to implement CGH algorithms in parallel hardware such as graphics processing units (GPU) or field-programmable gate arrays (FPGA). Iterative Fourier based algorithms will benefit from the parallelisation of 2D FFT while non-iterative Fourier-based algorithms, including the OSPR and single transform time-multiplexed (STTM) [9], can benefit even further thanks to the calculation process being independent of one another for different holograms. Despite the long development time consumed in the FPGA platform, it outperforms GPU in terms of power efficiency and reconfigurability; it also has the potential to transfer the design into application specified integrated circuits (ASIC) once the FPGA design has been accomplished.

Since 1992, researchers at the Chiba University have developed several application-specific hardware platforms named HOlographic ReconstructioN (HORN) computers [10]. FPGA was introduced to their design of HORN 5. HORN 8 is their latest development [10, 11]. With dedicated hardware algorithms applied, the CGHs calculated using HORN 8 are capable of being displayed by either amplitude- or phase-only spatial light modulators (SLM) [11]. For SLMs with full HD (1920×1080) resolutions, the performances for both cases are 53 frames per second (FPS) and 33.3 FPS, respectively.

In early 2019, Kim et al. proposed a hardware architecture based on Cascaded Generalized Fresnel Transform (CGFT) and Pupil Space Division Method (PSDM), where the performance of the CGH generation with the resolution of full HD for three channels is effectively 16 FPS [12]. Aside from accelerating CGH generation on FPGA, Seo et al. accomplished an ASIC design in 2017 [13]. Although the performance is not sufficient for real-time operation, their pioneer work provides fruitful guidance for future investigation.

3. Background

3.1. Heterogeneous FPGA-SoC Platform

Heterogeneous FPGA-SoC platform embeds the reconfigurable FPGA and a hardcoded microprocessor on the same chip. For Intel's heterogeneous devices, the pure FPGA fabric could be one that comes from the Stratix, Arria or Cyclone families, while the processor system is a dual-core ARM A9 microprocessor known as the hard processor system (HPS). To minimize the data transmission latency between the microprocessor and the FPGA, a highly-optimized Advanced eXtensible Interface (AXI) specified by the Advanced Microcontroller Bus Architecture 4 (AMBA4) is employed. Compared to the conventional FPGA devices, which only contains the reconfigurable fabric, the heterogeneous platform can improve the overall performance by separating tasks with different purposes and then run them in the dedicate chip area. In our proposed design, we use FPGA to perform the parallel computation and microprocessor for data retrieval.

3.2. One-step Phase Retrieval Algorithm

One-step Phase Retrieval, shown in Algorithm 1, was demonstrated by Cable et al. in 2004 [2, 14]. The algorithm relies on the time multiplexing of multiple sub-holograms during the reconstruction step. As human's perception of noise variance is considerably more sensitive than the noise means, the time-multiplexed sub-holograms altogether will result in a stacked replay field with improved projection quality [14]. Each sub-hologram is generated by a relatively straightforward process where only a single inverse 2D FFT and phase quantisation is involved. With the calculation process for different sub-frames being independent of each other, OSPR is regarded as a highly parallel algorithm which can be benefited from FPGA acceleration.

```
Algorithm 1: One step phase retrieval
Data:
I: Intensity of the target replay field
T: Target replay field
Input:
N: Number of iterations
Output: Reconstructed field
T = $\sqrt{\mathbf{I}}$;
for $k < N$ do
    Add random phase to the amplitude of the target images: $\mathbf{T}' = \mathbf{T} \cdot e^{j\theta}$
    Apply Fourier transform to get $\mathbf{H}_k = \mathcal{F}^{-1}\{\mathbf{T}'\}$
    Quantize the hologram based on the SLM modulation scheme:
        $\mathbf{H}_k^Q = \text{Quantization}(\mathbf{H}_k(x, y))$
    k = k + 1
end
```

4. Implementation

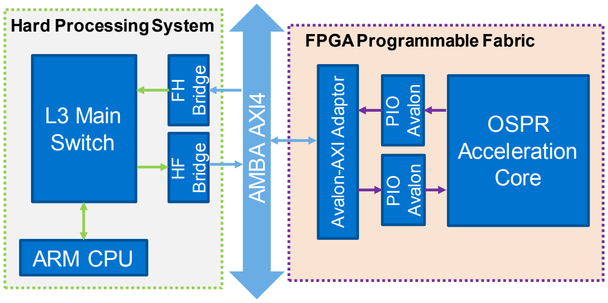


Fig. 1. HPS-FPGA system overview

Fig. 1 shows an overview of our proposed system. The OSPR acceleration core is implemented in the FPGA fabric of Cyclone V SoC. Several control signals and data path of the synthesised circuit are exposed to the AMBA AXI bus from the HPS side through parallel I/O (PIO) ports. Since PIO uses Avalon memory map (Avalon-MM) bus, a bus adapter is inserted between the PIO ports and the AXI4 bridge to interface the PIO with the FPGA-to-HPS (FH) and HPS-to-FPGA (HF) bridges [15]. HPS accesses data that are stored in M10K memory through Unix's memory mapping facility programmed in C language.

4.1. The OSPR acceleration core

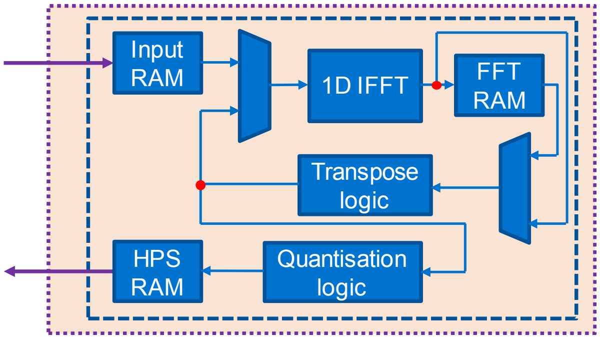


Fig. 2. Block diagram of the OSPR Acceleration core in the FPGA side

The block diagram of the OSPR acceleration core is shown in Fig. 2. The synthesised circuit performs 2D inverse FFT on a phase randomised target image followed by a binary quantisation. The single-port random access memory (RAM) is utilised to buffer the data across each module in order to carry out matrix transposition.

In total, 6 RAM blocks are instantiated in the design in order to load/store complex values. Three out of the six are for the real values while the other three are used for the imaginary parts.

We further categorise these RAM blocks into input, FFT and HPS RAMs. The input RAM stores the data of a phase randomised target image; the FFT RAM stores the result of 1D IFFT on all rows; and the HPS RAM stores the data of the quantised phase. As fixed-point arithmetics are being employed in the design, the inevitable bit width growth during the FFT operations due to additions and multiplications consequently leads to the bit width difference between each RAM block. Table 1 summaries the configuration for different RAM blocks.

Table 1. Configurations of M10K RAMs

		Real			Imaginary	
Category	INPUT	FFT	HPS	INPUT	FFT	HPS
Bit Width	8	12	20	8	12	20
RAM Depth	$1K$	512	512	$1K$	512	512

4.2. The accelerator core controller

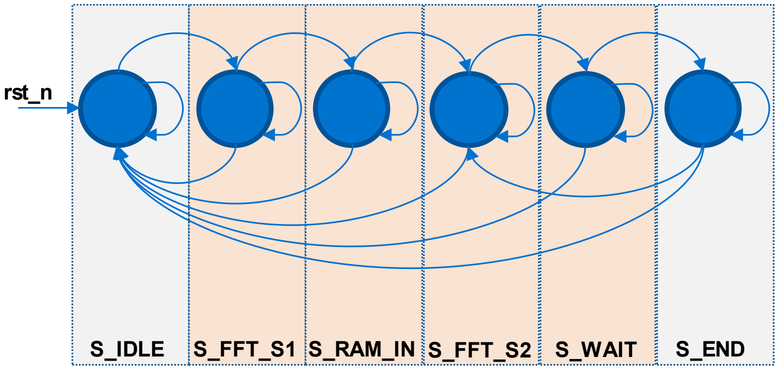


Fig. 3. Finite state machine of the system

Fig. 3 shows the finite state machine (FSM) controlling the data flow of the OSPR acceleration core. The FSM consists of the following six states:

- S_IDLE: This is the initial state, where the accelerator is in idle mode. The microprocessor can write data from the HPS into the **input RAM** and send control signals to reset or enable the accelerator cores.
- S_FFT_S1: After the HPS activates the accelerator cores, FSM will direct the system to **S_FFT_S1** state to perform 1D inverse FFT over all rows of the target image. Once the computation is finished, the system will be switched into the next available state.
- S_RAM_IN: This state is for buffering the computed FFT result from the previous state via stream input.
- S_FFT_S2: At this state, matrix transpose is performed and then followed by a 1D inverse FFT over all rows of the transposed data. Once the computation is finished, the system will be switched into the **S_WAIT** state.
- S_WAIT: This state performs binary quantisation, post transposition and *fftshift* operation before storing the data into the HPS RAM.
- S_END: A **done** signal will be send to the HPS to allow the HPS read the computed binary phase data from the HPS RAM.

4.3. Quantisation logic

After taking the 2D inverse FFT, a matrix of the complex hologram containing both the amplitude and phase information is obtained. As the holograms will be displayed on a binary phase-only SLM, quantisation is required to binarise the phase value of the complex hologram.

Conventionally, quantisation is achieved by first retrieving the phase angle, followed by an integer rounding operation that depends on the quantisation schemes used. In order to perform binary quantisation on phase angles, we implement a method inspired by the Argand diagram, which avoids the use of the *arctan2* function that takes up significant hardware resources. No extra resources other than an inverter is needed. It can be found from Fig. 4 that the sign of the phase angle follows the sign of the imaginary number. As two's complement number is used, the sign of the phase angle is then the inverted sign of the imaginary number.

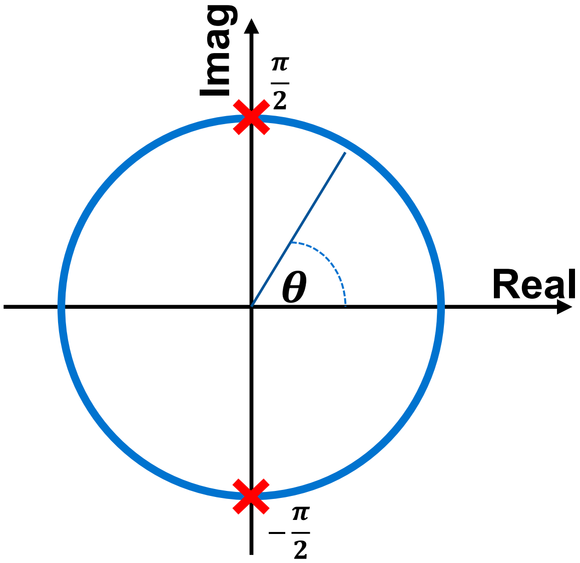


Fig. 4. Argand diagram showing binary quantisation at $-\frac{\pi}{2}$ and $\frac{\pi}{2}$

The observation is also validated by generating the truth table and performing boolean logic based on the following equations, where x and y represent the sign of the real part and imaginary part, respectively.

$$\operatorname{atan2}(y,x) = \begin{cases} 2\arctan\left(\dfrac{y}{\sqrt{x^2+y^2}+x}\right), x > 0 & \text{(1a)} \\ 2\arctan\left(\dfrac{\sqrt{x^2+y^2}-x}{y}\right), x \leq 0 & \text{(1b)} \end{cases}$$

Figure 5 shows the proposed circuit diagram for binary quantisation. The real part of the result of 2D inverse FFT is set to zero (ground) since we are only interested in the imaginary part of the output; an inverter is added between the quantised output and the most significant bit (MSB) of the imaginary part. Although only binary quantisation is presented here, it is possible to perform graylevel quantisation for multi-phase SLMs. However, this would require the implementation of *arctan* functions, which increases the design complexity.

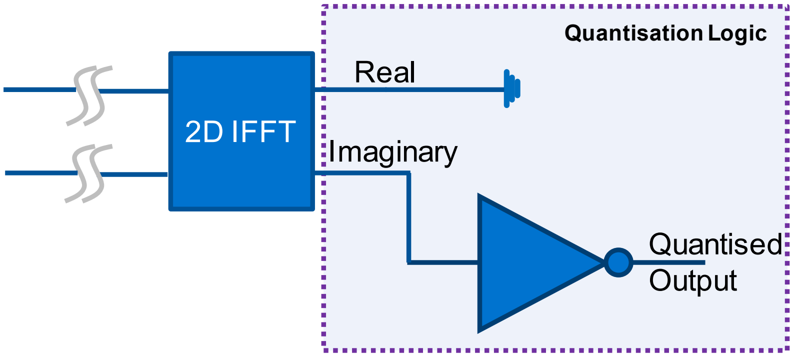


Fig. 5. Quantisation logic circuit diagram

4.4. *Optimisation for parallelism*

We further investigated the parallelism of the proposed OSPR accelerator architecture. For target images with a resolution of 128 × 128 pixels, instead of having one single OSPR core, six OSPR acceleration cores can be instantiated in the programmable logic in parallel. The reason to implement a maximum of six cores is due to the BRAM resource constraints, which is limited by the hardware. The block diagram for this parallel design is shown in Fig. 6.

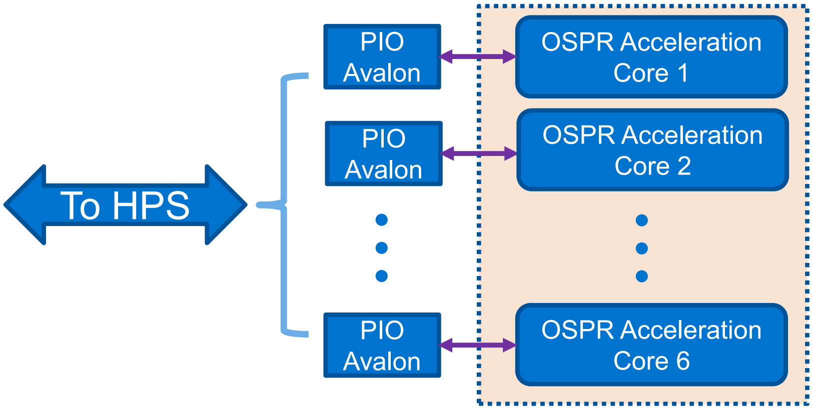


Fig. 6. Block diagram of multi-OSPR acceleration core

5. Experiment and results

5.1. *Experimental setup*

The design is implemented on an Intel DE10-Standard board [16]. It has a dual ARM Cortex-A9 CPU and a Cyclone-V 5CSXFC6D6F31C6N programmable logic [17]. The operating system running on the hard processor system is the DE10-Standard-Linux-Console provided by Terasic Inc. ModelSim (Intel FPGA Edition 10.5b) was used to perform the functional simulation. Quartus Prime 2018.1 standard software was used for synthesis and implementation of the design. After place and route, the system can operate at 50 MHz. The development board is connected to a host computer, and the on-chip HPS is accessed by UART.

For computational speed and quality comparison between a baseline personal computer, we implemented the algorithm using Matlab 2019a on a Win10 computer running an Intel Core i7-6800K CPU operating at 3.40 GHz, with a memory of 32 GB. A target image with the resolution of 128 × 128 (limited by the hardware resources of the FPGA platform) is used.

5.2. Hardware resource utilization

The hardware resources utilized for the generation of CGHs with a resolution of 128 × 128 using single acceleration core and multi-core setup are shown in Table. 2 and Table. 3, respectively. The amount of M10Ks used in the single-core and multi-core does not follow a linear fashion and such is basically due to the optimisation performed by the Quartus Prime synthesiser. Although Table. 3 shows that 28% of BRAM bits are still available, due to the specific configuration rules they cannot be further utilized [17]. It is possible to use the same design to generate holograms with a larger resolution. The main constraint for FPGA implementation of CGH calculation is the available hardware logic resources. We used a parametrized design when we implemented the system, therefore, a relatively simple modification on the design parameters will enable us to calculate high-resolution holograms as long as sufficient hardware resources are present.

Table 2. Resources usage for 1 sub-frame with the resolution of 128 × 128

	ALMs	M10Ks	DSP	BRAM Bits
Available	41,910	553	112	5,662,720
Used	4,500	107	12	683,088
Percentage	11%	19%	11%	12%

Table 3. Resources usage for 6 sub-frames with the resolution of 128 × 128

	ALMs	M10Ks	DSP	BRAM Bits
Available	41,910	553	112	5,662,720
Used	23,988	553	72	4,085,964
Percentage	57%	100%	64%	72%

5.3. Performance evaluation

5.3.1. Computational speed comparison

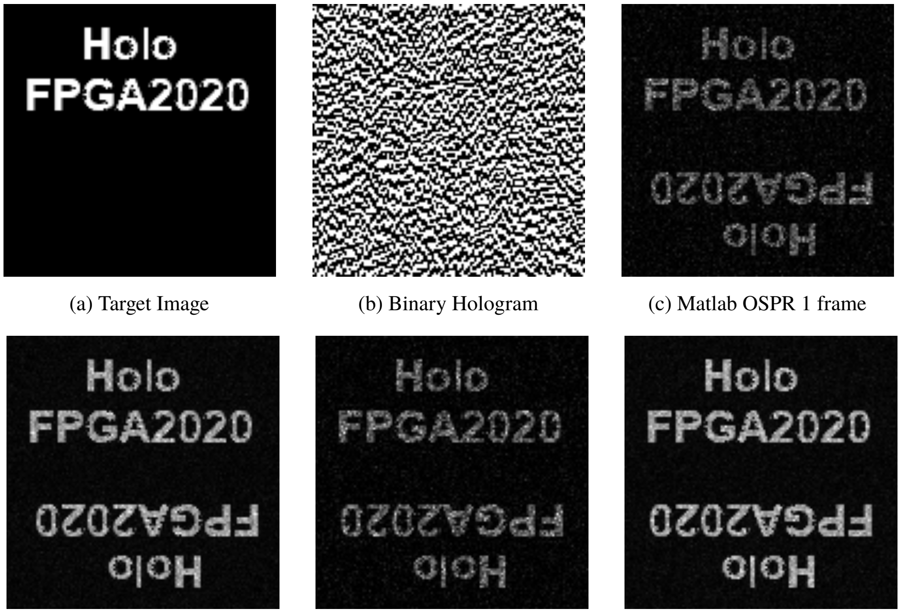
Table. 4 shows the computational speed comparison between FPGA and Matlab running on personal computer. The frequency of the system clock used in the FPGA design is 50 MHz, which is 68× slower than the CPU system clock at 3.4 GHz. The use of fixed-point logic in the FPGA design simplifies the arithmetic circuit and leads to an average calculation time of 0.674 ms. Compared with the desktop Matlab implementation, the calculation time on the FPGA platform is 1.9× slower for a single sub-hologram generation. However, if we consider the generation of six sub-holograms with multiple parallel cores, where the computation time spent in a multi-core FPGA design is the same as a single-core implementation, an overall 2.5× speed enhancement can be seen as compared to the Matlab implementation.

Table 4. Speed comparisons between FPGA and personal computer with the CGH resolution of 128 × 128

	CPU		FPGA	
	1 sub-frame	6 sub-frames	1 sub-frame	6 sub-frames
Execution time (ms)	0.3813 ± 0.1257	1.7047 ± 0.3995	≈ 0.674	≈ 0.674

5.3.2. Numerical reconstruction and quality analysis

Fixed-point arithmetics may result in the loss of accuracy, we perform a numerical reconstruction using Matlab to compare the quality of the replay field reconstructed by hologram generated by FPGA and personal computer. We visualise the reconstruction as shown in Figs. 7c to 7f to compare image quality between the two different platforms. The numerical reconstruction of the replay field for 6 sub frames is obtained by time-averaging all independent frames.



(a) Target Image (b) Binary Hologram (c) Matlab OSPR 1 frame

(d) Matlab OSPR 6 frames (e) FPGA OSPR 1 frame (f) FPGA OSPR 6 frames

Fig. 7. Visual comparison of reconstruction result between OSPR on FPGA platform and OSPR on desktop Matlab. The conjugate image is due to binary quantisation

We use peak signal and noise ratio (PSNR) and structural similarity index (SSIM) to numerically compare the quality of the replay field generated by Matlab and FPGA [18, 19]. The results were obtained by averaging the results of 10 independent measurements.

Table 5. Image quality comparisons between FPGA and personal computer

Evaluation Matrix	MATLAB		FPGA	
	1 frame	6 frames	1 frame	6 frames
PSNR	12.746 ± 0.065	13.037 ± 0.077	12.7537 ± 0.0819	13.0318 ± 0.2232
SSIM	0.141 ± 0.006	0.223 ± 0.003	0.1435 ± 0.0063	0.2232 ± 0.0033

5.3.3. Experimental reconstruction and quality analysis

In order to validate the holograms generated by the heterogeneous platform, we used an optical projection system to examine the quality of the corresponding replay field [20]. The schematic of the holographic projection system is shown in Fig. 8. A 532 nm polarised laser is used as the light source. The beam is then being focused down by an aspheric singlet to form a virtual point source for the system. Next, the beam from the virtual point source passes through a beam splitter

(BS) and then reaches a collimating lens and subsequently illuminates the SLM. Finally, the replay field is projected onto the diffusive white board, with the size defined by the lens group.

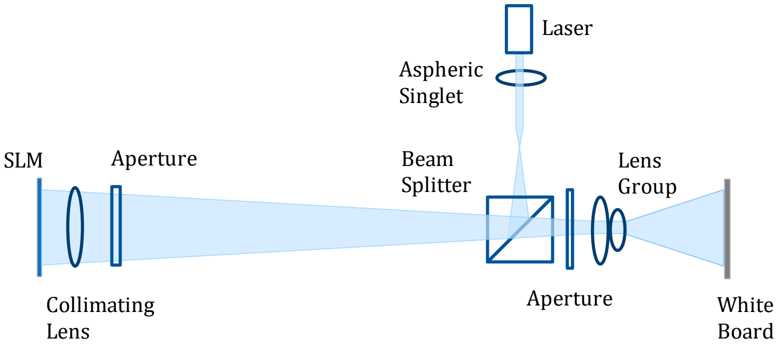


Fig. 8. The schematic of the projection system. The wavelength of the laser source is 532 nm. The reconstructed image appears on the diffusive white board after optical magnification by the lens group.

The resolution of the SLM is 1024 × 1024 which is significantly larger than that of the hologram generated by the FPGA. In order to obtain a decent replay field, we tile the FPGA generated hologram 64 times to fit the area of the SLM as shown in Fig. 9. The replayed image for 1 frame and 8 frames are shown in Fig. 10a and Fig. 10b respectively. Tiling changed the spatial frequency, and therefore the replay field of the image is stretched by a factor of 64, where each noticeable pixellated square represents one pixel in the original image.

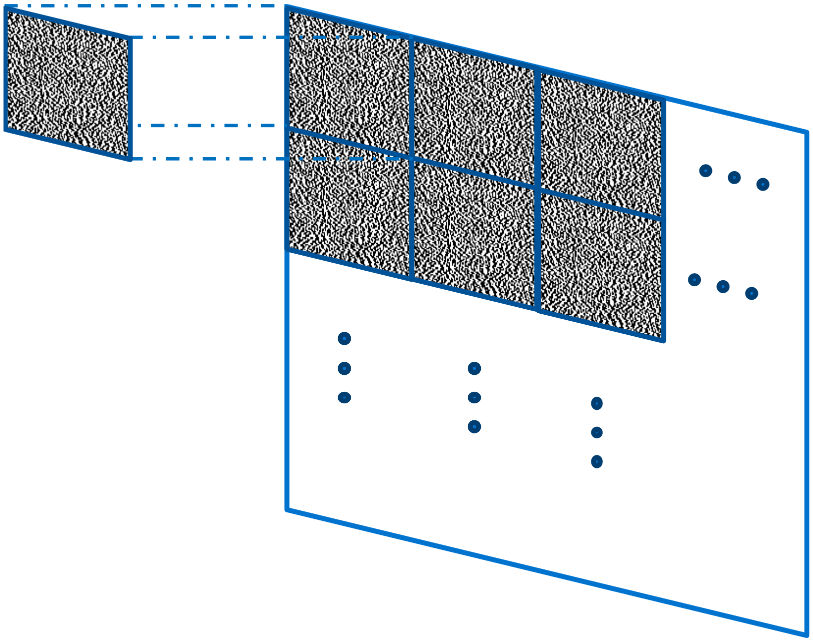
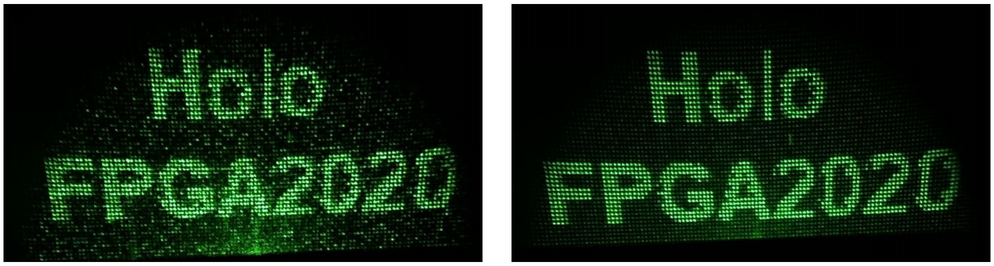


Fig. 9. Tiling SLM: a small-sized hologram is tiled into a larger one to project onto the SLM display



(a) Experimental result, 1 frame

(b) Experimental result, 8 frames

Fig. 10. Replay field of the binary phase hologram generated by the OSPR using the laboratory setup

6. Conclusion

In this paper, we present a cost-optimized heterogeneous FPGA architecture to generate CGHs based on parallelized OSPR algorithm. Two designs, one with single-core and the other with six acceleration cores, have been implemented and tested on a Intel Cyclone V FPGA SoC. Performance metrics including the computational speed and the quality of the replay field have been assessed. The results show that for the proposed design, the FPGA hardware implementation can be 2.5× faster than a software implementation on a personal computer running a multi-core i7 processor. Further modifications can be made to improve the performance by increasing the frequency of the FPGA system clock and use direct memory access to shorten the memory read/write time and minimize memory latencies.

It is of no doubt that the higher-end an FPGA platform is, the more flexibility and computational power there will be for real-time processing. Previous researches all use high-end FPGA to obtain high performance. Our work is based on a cost-effective device which is around 500 USD. With this heterogeneous FPGA, we have full control of a hardened ARM microprocessor and interface peripherals which reduces the cost and risk compared with conventional FPGA board development for holographic applications. We intend to provide a reference on the trade-off between cost and performance as we have shown the practicability of implementing holographic applications within cost-effective heterogeneous FPGA platforms. This enables a promising future to investigate into cost-optimized on-the-edge CGH calculation and its related applications.

In general, there is a growing interest in calculating CGHs using compact and low-power consumption electronics. FPGA-SoC is one of the heterogeneous systems that meets these criteria. The platform allows designers to focus on the implementation of high-performance parallel logics by reducing the protocol design needs that are often inevitable when using pure programmable logics. The processor hardcoded on the chip will handle the majority of the protocol with a relatively high performance. Compared to conventional soft processor cores, including Altera NIOS-II and Xilinx MicroBlaze, the hard processor core embedded on the FPGA-SoC systems do not occupy any of the FPGA's internal logic resources and is able to operate at the maximum performance because of the dedicated architectural design. The underlying advantages brought by the heterogeneous platform can potentially reduce the cost and size of the holography display system, enabling this display technology to become reachable to ordinary households and consumers.

Acknowledgments

The authors would like to thank Intel university program for their generous donations of the DE-10 FPGA development board and software licenses.

Disclosures

The authors declare no conflicts of interest.

References

1. J. W. Goodman, *Introduction to Fourier optics* (Roberts & Co. Publishers, 2005), 3rd ed.
2. E. Buckley, “Computer-Generated Phase-Only Holograms for Real-Time Image Display,” Adv. Hologr. - Metrol. Imaging (2011).
3. J. H. Park, “Recent progress in computer-generated holography for three-dimensional scenes,” J. Inf. Disp. **18**, 1–12 (2017).
4. A. Maimone, A. Georgiou, and J. S. Kollin, “Holographic near-eye displays for virtual and augmented reality,” ACM Transactions on Graph. **36**, 1–16 (2017).
5. M. A. Seldowitz, J. P. Allebach, and D. W. Sweeney, “Synthesis of digital holograms by direct binary search,” Appl. Opt. **26**, 2788 (1987).
6. R. W. Gerchberg and W. O. Saxton, “A practical algorithm for the determination of phase from image and diffraction plane pictures,” Optik **35**, 237–246 (1972).
7. J. Carpenter and T. D. Wilkinson, “Graphics processing unit-accelerated holography by simulated annealing,” Opt. Eng. **49**, 095801 (2010).
8. J. Liu and M. R. Taghizadeh, “Improved algorithm for the design of diffractive phase elements for beam shaping,” Conf. on Lasers Electro-Optics Eur. - Tech. Dig. **27**, 674 (2003).
9. P. J. Christopher, R. Mouthaan, V. Bheemireddy, and T. D. Wilkinson, “Improving performance of single-pass real-time holographic projection,” Opt. Commun. **457**, 124666 (2019).
10. T. Sugie, T. Akamatsu, T. Nishitsuji, R. Hirayama, N. Masuda, H. Nakayama, Y. Ichihashi, A. Shiraki, M. Oikawa, N. Takada, Y. Endo, T. Kakue, T. Shimobaba, and T. Ito, “High-performance parallel computing for next-generation holographic imaging,” Nat. Electron. **1**, 254–259 (2018).
11. T. Nishitsuji, Y. Yamamoto, T. Sugie, T. Akamatsu, R. Hirayama, H. Nakayama, T. Kakue, T. Shimobaba, and T. Ito, “Special-purpose computer HORN-8 for phase-type electro-holography,” Opt. Express **26**, 26722 (2018).
12. H. Kim, Y. Kim, H. Ji, H. Park, J. An, H. Song, Y. T. Kim, H. S. Lee, and K. Kim, “A single-chip FPGA holographic video processor,” IEEE Transactions on Ind. Electron. **66**, 2066–2073 (2019).
13. Y.-H. Seo, Y.-H. Lee, and D.-W. Kim, “ASIC chipset design to generate block-based complex holographic video,” Appl. Opt. **56**, D52 (2017).
14. A. J. Cable, E. Buckley, P. Mash, N. A. Lawrence, T. D. Wilkinson, and W. A. Crossland, “Real-time Binary Hologram Generation for High-quality Video Projection Applications,” in *SID Symposium Digest of Technical Papers,* vol. 35 (2004), pp. 1431–1433.
15. Intel Corp., “Embedded Peripherals IP User Guide,” Tech. rep. (2019).
16. Intel Corp., “DE10-Standard Computer System with ARM * Cortex * A9 DE10-Standard Computer Contents,” Tech. Rep. June (2017).
17. Intel Corp., “Cyclone V Device Handbook,” Tech. Rep. January, San Jose, CA (2018).
18. Z. Wang and A. C. Bovik, “Mean Squared Error : Love It or Leave It ?” IEEE Signal Process. Mag. **26**, 98–117 (2009).
19. W. Seo, H. Song, J. An, J. Seo, G. Sung, Y. T. Kim, C. S. Choi, S. Kim, H. Kim, Y. Kim, Y. Kim, Y. Kim, H. S. Lee, and S. Hwang, “Image quality assessment for holographic display,” IS T Int. Symp. on Electron. Imaging Sci. Technol. pp. 186–190 (2017).
20. J. P. Freeman, T. D. Wilkinson, and P. Wisely, “Visor projected HMD for fast jets using a holographic video projector,” in *Three-Dimensional Imaging, Visualization, and Display 2010 and Display Technologies and Applications for Defense, Security, and Avionics IV,* vol. 7690 (2010), p. 76901H.