Implementation of a High-speed Template Matching System for Wafer-vision Alignment Using FPGA

Jae-Hyuk So[1] and Minjoon Kim[2*]
[1] Data Convergence Platform Research Center, Korea Electronics Technology Institute
Seongnam, South Korea
[e-mail: sojh0124@keti.re.kr]
[2] Division of Semiconductor and Electronics Engineering, Hankuk University of Foreign Studies
Yongin, South Korea
[e-mail: mjoon@hufs.ac.kr]
[*]Corresponding author: Minjoon Kim

Received March 28, 2024; accepted June 3, 2024;
published August 31, 2024

Abstract

In this study, a high-speed template matching system is proposed for wafer-vision alignment. The proposed system is designed to rapidly locate markers in semiconductor equipment used for wafer-vision alignment. We optimized and implemented a template-matching algorithm for the high-speed processing of high-resolution wafer images. Owing to the simplicity of wafer markers, we removed unnecessary components in the algorithm and designed the system using a field-programmable gate array (FPGA) to implement high-speed processing. The hardware blocks were designed using the Xilinx ZCU104 board, and the pyramid and matching blocks were designed using programmable logic for accelerated operations. To validate the proposed system, we established a verification environment using stage equipment commonly used in industrial settings and reference-software-based validation frameworks. The output results from the FPGA were transmitted to the wafer-alignment controller for system verification. The proposed system reduced the data-processing time by approximately 30% and achieved a level of accuracy in detecting wafer markers that was comparable to that achieved by reference software, with minimal deviation. This system can be used to increase precision and productivity during semiconductor manufacturing processes.

Keywords: Wafer-alignment system, Normalized cross-correlation, Template matching, Vision inspection, Hardware accelerator.

A preliminary version of this paper was presented at ICONI 2023, and was selected as an outstanding paper. This work was supported by the Technology Innovation Program (20015975, Development of an apparatus for probe head based on High Precision 3 Axis Loader) funded By the Ministry of Trade, Industry & Energy(MOTIE, Korea). This work was also supported by Hankuk University of Foreign Studies Research Fund of 2024.

1. Introduction

Semiconductor inspection equipment and components are used to verify the physical, chemical, and electrical properties of semiconductor components fabricated on wafers during the manufacturing process. As technology advances, high-precision technologies capable of providing high integration density and fine patterning are required; consequently, the demand for inspection equipment that can respond to these advancements is increasing.

In semiconductor manufacturing, wafer-alignment systems are crucial subsystems for wafer probing. Optical image processing is used to detect wafer alignment keys and automatically correct the wafer position with high precision [1,2]. Recently, with semiconductor processes becoming increasingly precise, the demand for fast and highly accurate automatic vision inspection methods has increased considerably [3]. Vision inspection methods reduce the necessary steps and time required for alignment and the unforeseen issues caused by workers during manufacturing.

Template matching, a fundamental method, is particularly useful for vision inspection of wafers when aligning the alignment keys within images to probe positions. After pre-alignment, accurately locating alignment keys on the wafer is the next step, followed by aligning the center and direction of the wafer. Typically, similarity calculations, considering rotation and translation, and the rotated versions of the template are required.

This method is used to detect the current position of the target indication, transmitting error information to the stage controller in the system and subsequently moving and rotating the wafer to the inspection position. During this process, the accuracy of the template-matching algorithm and the data-processing speed of the software are critical. However, the computational cost significantly increases as multiple similarity calculations based on the target-angle step using the rotated template versions are required for each position in the detection area [4,5].

In this study, we propose a dedicated hardware architecture for implementing hardware-accelerated template-matching algorithms on a field-programmable gate array (FPGA) to address the aforementioned issue. To simplify the calculation process, an optimization strategy was applied in which the basic search area was confirmed within the overall search area and the basic search area was progressively expanded in steps. We constructed a wafer-alignment system and obtained implementation results from testing the FPGA boards.

The contributions of this study can be summarized as follows:

• Firstly, we optimize the commonly used template-matching algorithm based on the marker features used on wafers in the industry. We maintain accuracy during this process based on existing software as a reference while improving speed. We present the optimized results obtained from the simulator by limiting the search area, restricting the angle, and removing unnecessary blocks.

• Secondly, we propose a high-speed template-matching hardware architecture capable of processing high-resolution images faster than the existing system. Designed for rapidly processing images captured by high-precision cameras, this architecture leverages both the processing system and programmable logic while maintaining accuracy.

• Thirdly, we implement the proposed algorithm and hardware on the Xilinx ZCU104 board and built wafer-vision-inspection equipment used in the industry. Finally, we present the experimental results in comparison with the results obtained from the reference software based on the actual equipment used in the industry.

The rest of this paper is structured as follows. Chapter 2 describes the template-matching algorithm. Chapter 3 presents a hardware simulator for template matching optimized for wafer alignment markers based on the existing template-matching algorithm. Chapter 4 presents the hardware structure and design results. Chapter 5 discusses the validation of the platform proposed in this study. Finally, we conclude this study in Chapter 6.

2. Template-matching Algorithm based on Normalized Cross-correlation

This section describes the detailed conversion of template-matching algorithms used based on the normalized cross-correlation (NCC) method. Template matching is one of the most commonly used machine vision techniques. It uses a template to scan across an image and a similarity calculation method to detect the required targets in which an object within an image should be aligned with the position of the real object. The template image (T) is the given image, which is compared to inspection images, and a best-matched template image is expected to exist. Additionally, T can be described as a two-dimensional array, $T_{m\times n}$, where m and n are the pixel numbers in the vertical and horizontal directions of the image, respectively.

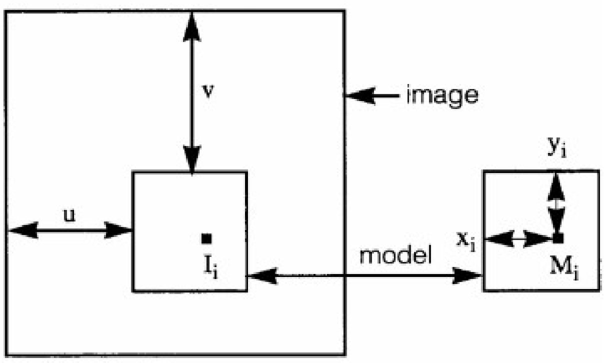

Fig. 1. Principle of template matching

The algorithm aims to locate the area within the given image that is the most similar to the template image. **Fig. 1** shows the principle of the template-matching process. The reference template image moves across the search region in the input-inspection image, with each pixel moving along the x and y axes. In each moving step, a similarity is calculated. Next, the (x, y) position at which the similarity between the two images is the maximum is located within the search region [6].

Typically, the most popular and powerful correlation-based similarity measurement is NCC, which is used to evaluate the similarity between the template and the source images. The optimal results obtained from template matching are indicated by the maximum NCC correlation, and an NCC coefficient close to 1 indicates fine matching, i.e., the two compared images are identical. The NCC coefficient can be used for aligning the template image $\mathrm{T}(i, j)$ and the inspection image $\mathrm{I}(i, j)$, as follows:

$$\delta_g(x,y) = \frac{\sum_{i=-m/2}^{m/2} \sum_{j=-n/2}^{n/2} [I(x+i,y+j)\cdot T(i,j)] - m\cdot n\cdot \mu_I \cdot \mu_T}{\sqrt{\sigma_I \cdot \sigma_T}} \tag{1}$$

$$\mu_T = \frac{1}{m \cdot n} \sum_{i=-m/2}^{m/2} \sum_{j=-n/2}^{n/2} T(i,j).T(i,j) \tag{2}$$

$$\mu_I = \frac{1}{m \cdot n} \sum_{i=-m/2}^{m/2} \sum_{j=-n/2}^{n/2} I(x+i,y+j) \tag{3}$$

where the size of the template image is defined by m·n, and uT and uI are the gray-level averages of the template and windowed search images, respectively, i.e., $\delta_g(x, y)$ is the degree of similarity between template and windowed search image at coordinates (x, y) for gray-level images [3].

However, NCC incurs a high computational cost when achieving a high-precision scale and rotation matching, and long scanning times are a major problem. The larger the image, the longer the scanning time. To calculate the NCC, a template in a scene image is obtained by sliding the template window on a pixel-by-pixel basis and the NCC coefficient is computed. Therefore, the matching time unavoidably increases. Template matching based on a full-search-correlation metric is exhaustively time-consuming; therefore, locating the target object rapidly is an important task during the matching process [7].

To reduce the computational burden, an elimination method based on the upper bounds has been applied in the full-exhaustive-search template-matching procedure to omit the unlikely target candidates [8]. We propose a coarse-to-fine scheme and two-stage techniques for the search step to check the primary search area and expand to the entire search area. To avoid unnecessary computation of the NCC metric, a multilevel winner-update scheme [9] and weak classifiers are also proposed.

To address these problems, the NCC method with a rotated angle is introduced [10,11]. In [10], the pre-computed threshold score set from rotated templates to the original template is used to recognize the target object by omitting unnecessary computation in the primary search phase. Subsequently, the rotation angle of the target object is estimated in the full search phase in which the templates are precisely compared based on the adjacent angle determined in the first phase. However, the alignment accuracy obtained depends on the rotation angle.

The template-matching process proceeds in three steps, as shown in **Fig. 2**. First the Gaussian pyramid is applied to the input source and template images, and the feature points within the pyramid image are compared. Top-layer search, which involves rotating the target image by various angles at the smallest resolution of the image pyramid, is the first phase of the search process. More specifically, this search process extracts approximate search points at a low resolution. A pyramid search is a process of closely comparing all remaining pyramid images resulting from the primary search [12].

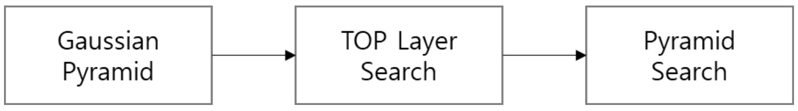

Fig. 2. Main process of template matching

2.1 Gaussian pyramid

The Gaussian pyramid generates image sets of various resolutions for the feature-point analysis of an image. Image sets are constructed by applying a Gaussian filter to the original image and repeatedly performing 2:1 down sampling to generate low-resolution images. **Fig. 3** shows the result of the repeated application of a Gaussian filter and down sampling of the original image, and Gaussian smoothing is applied in the form of an infinite impulse response between interlayers. In the image pyramid, as the image decreases in size, the high-frequency component is removed.

$$ceil(\log_2(width \times height \gg 8)) \tag{4}$$

In this study, the images with the smallest and largest resolutions are defined as depth N or TOP layer and depth 0, respectively. The depth of the Gaussian pyramid is determined by the image resolution, as expressed in in the Equation (4). The depth of the pyramid directly affects the extent of computation; therefore appropriate considerations are required. A pyramid is constructed by performing 2D convolution using 5 × 5 Gaussian filter coefficients in **Fig. 3**. To apply 5 × 5 convolution, the image boundary must be padded, and the image is symmetrically copied based on the boundary. A Gaussian pyramid is applied to the source and the target template images respectively.

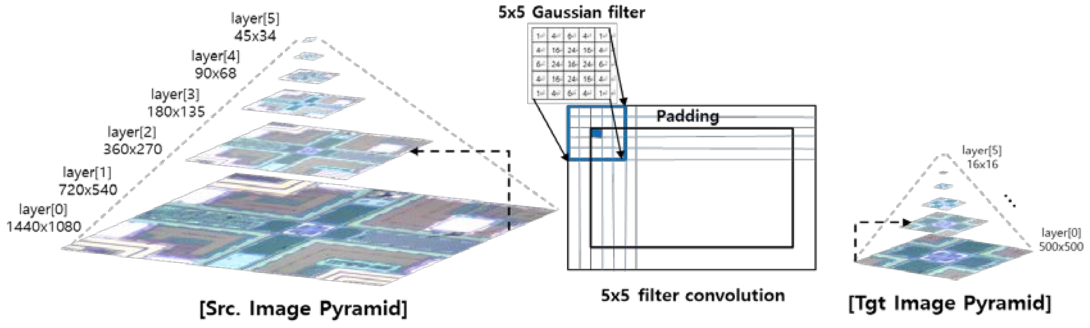

Fig. 3. Process of generating a Gaussian pyramid structure

2.2 Template-matching algorithm

The template-matching algorithm first runs a top-layer search, which consists of four steps, as shown in **Fig. 4**. The rotation angle corresponding to the image search is determined, and while rotating the top layer of the source pyramid, scores representing high similarity with the top layer of the target pyramid is found and sorted. During pyramid-layer search, an additional search process is repeatedly applied to the pyramid layers based on the parameters calculated during the previous search process.

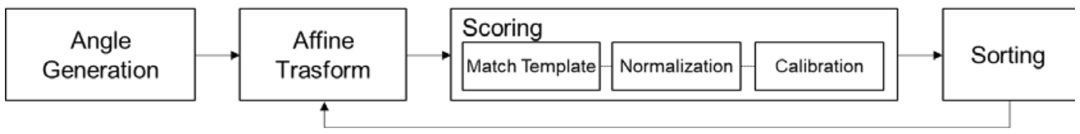

Fig. 4. Process of top-layer search

2.2.1 Angle generation

During angle generation, the calculated angle step according to the resolution of the image is used to determine the rotation angle. As the image size increases, the angle step decreases and the time required for matching increases. If the tolerance value is 180°, all angles up to 360° can be searched.

$$step = arct\left(\frac{2}{\max(dst_{w_n}, dst_{h_n})}\right) \times 180 \div PI \tag{5}$$

$$angles = [-tolerance_{angle} : tolerance_{angle} : step] \tag{6}$$

2.2.2 Affine transform

To compare the rotated source image with various angles and the target image, an affine transform is performed on the source image at every angle step determined during the previous process. As shown in Equation (7), a pixel is interpolated using a bilinear filter on the distorted transformed image using a 2 × 3 rotation matrix.

$$\mathrm{M} = \begin{bmatrix} \alpha & \beta & (1-\alpha)\cdot \mathrm{center.x} - \beta \cdot \mathrm{center.y} \\ -\beta & \alpha & \beta \cdot \mathrm{center.x} + (1-\alpha)\cdot \mathrm{center.y} \end{bmatrix} \quad \text{where} \quad \begin{cases} \alpha = \mathrm{scale}\cdot\cos(\mathrm{angle}) \\ \beta = \mathrm{scale}\cdot\sin(\mathrm{angle}) \end{cases} \tag{7}$$

2.2.3 Scoring and Sorting

During the scoring process, the optimal location is searched by calculating the similarity between the transformed source and target images. NCC is used to determine image similarity. Using the match template, the entire image is searched, and correlations of all locations are calculated the larger the correlation, the higher the probability that the two images are similar. The bit-depth corresponding to the calculated correlation value is high; therefore bit-depth is normalized to reduce the data size. Subsequently, during calibration, nine additional adjacent pixels are searched based on the optimal location. During the sorting process, the location information with the highest correlation that exceeds the threshold value and the correlation score of the data are classified and stored as parameters.

3. Hardware Simulator

Hardware refers to specialized equipment designed for specific applications. Unlike general-purpose software, hardware offers advantages by providing high-performance operations in constrained environments. Accuracy at the hardware simulation stage must be evaluated for performance verification and the early detection of issues before hardware development. In this study, the algorithm used as a reference is the template-matching algorithm designed using the OpenCV software [3]. This algorithm must be optimized by re-designing for identifying wafer-alignment marks in semiconductor equipment. This section presents the process of designing and the results obtained from a hardware simulator by removing unnecessary computational steps and converting the code into a format suitable for hardware design during the direct implementation of OpenCV functions.

During TOP-layer search, the angle step is influenced by the resolution of the target image, as expressed in Equation (2). As the image size increases, the number of steps generated decreases; therefore the number of computations required increases. As affine and template matching are compared for each search point, the angle step directly affects computational complexity.

In the reference software, a tolerance value of 180° is used to search for all angles. Depending on the size of the input target image, a typical range of 50–200 angles may be searched. The reference software applies various angle steps depending on the resolution of the input image; however, not all angles can be applied using hardware. Therefore, accuracy and speed are measured during performance evaluation after fixing the step value. As the angle step increases, the search speed increases but the success rate decreases as objects located at unsearched angles cannot be found. In addition to the angle step, the existing algorithm adds another nine search positions for each angle. This process involves searching for an additional nine positions around the optimal position based on the highest similarity obtained during the scoring step conducted on the TOP layer. If 50 optimal positions are found, an additional total of 450 angles need to be searched. Therefore, owing to the high computational load, ensuring real-time performance is difficult; however, this issue can be resolved using the hardware simulator by removing additional search block while retaining the accuracy related to finding wafer markers.

Based on the hardware design, the resolutions of the wafer-marker images are limited. The depth of the Gaussian pyramid is determined by image resolution. As the depth of the pyramid directly affects the computational load, the resolution of the target image was set to up to 512 × 512 in the current system. The pyramid layer was designed to support up to six layers, and for each layer, a fixed angle step was used. To ensure accuracy, support was provided up to the second decimal place.

Initially, semiconductor equipment is pre-aligned to ensure that the wafer-alignment marker mechanically enters the field of view (FOV) of the camera. When the alignment marker enters the FOV (Field of View), the alignment error range is limited within a certain angle range, so there is no need to search angles up to 360°. The algorithm in the reference software includes angles below 360°; however, the hardware simulator is designed to support a tolerance angle from –10° to +10°. To increase the precision of angle searches, the design allows for angles up to 0.01° to be searched, and accordingly, a trigonometric look-up table (LUT) is constructed using 0.01° increments. In the first phase, the top layer was searched in large angle steps, whereas the bottom layer was gradually searched in small angle steps. Therefore, the LUT containing trigonometric functions supported in the top layer is configured for angles from –20 to +20°. **Fig. 5** demonstrates the search structure based on angle steps from the pyramid layer, extending up to the final layer. From the second phase, the search is conducted using a predetermined angle step as a reference for each layer, exploring angles on the left and right of this reference.

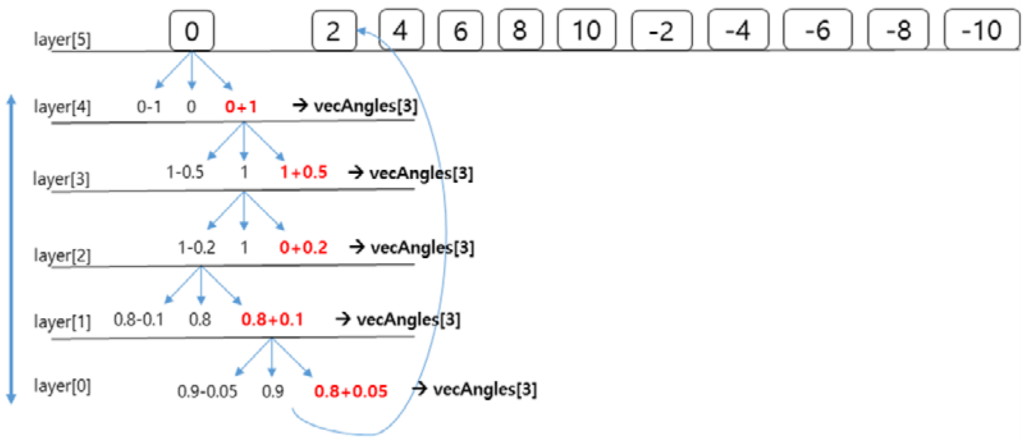
layer[5]
0
2
4
6
8
10
-2
-4
-6
-8
-10
layer[4]
0-1 0 0+1 → vecAngles[3]
layer[3]
1-0.5 1 1+0.5 → vecAngles[3]
layer[2]
1-0.2 1 0+0.2 → vecAngles[3]
layer[1]
0.8-0.1 0.8 0.8+0.1 → vecAngles[3]
layer[0]
0.9-0.05 0.9 0.8+0.05 → vecAngles[3]

Fig. 5. Search structure of the pyramid-layer angle

4. Hardware Architecture

In this section, we introduce the hardware platform related to the FPGA-based template-matching algorithm for identifying alignment markers based on the results obtained from the hardware simulator. **Fig. 6** illustrates the diagram of the top block designed using operational blocks.

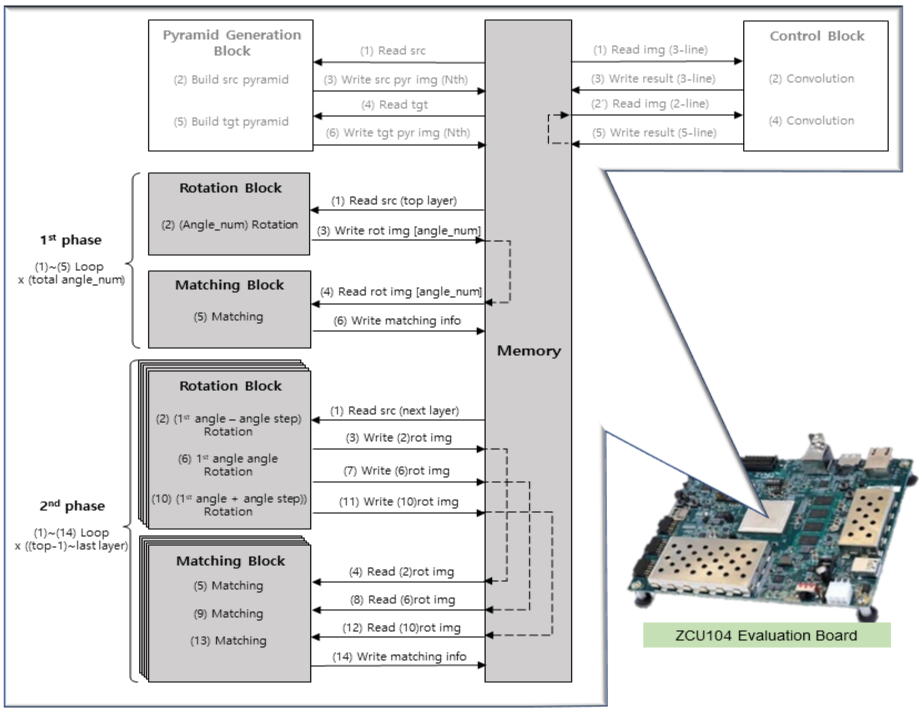

Fig. 6. Diagram of the top block depicting template matching

The operations of the pyramid-image-generation and matching blocks are designed to be accelerated using programmable logic (PL) because these blocks can rapidly access memory by sequentially processing each image pixel and can handle consecutive pixels in parallel.

However, the rotation block, which performs numerous mathematical computations and lacks continuity in coordinate values during memory access, is processed by the processing system (PS). Fig. 7 illustrates the blocks included under the PS and PL.

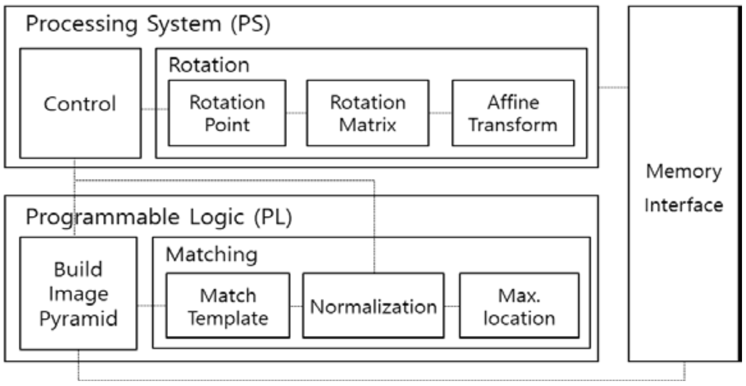

Fig. 7. Block diagram of the FPGA PS/PL

In the pyramid block, the source or target image is down sampled at a 2:1 ratio until it reaches a minimum size determined by the user within a maximum of six layers. The down-sampling results are stored in the memory. To apply 5 × 5 convolution, only the necessary data is read line-by-line from memory and stored in the internal line buffers. Seven lines of buffers are present, and the first three lines are used to initially perform the convolution. Subsequently, batches of five lines are processed at once, and an output is generated. At each clock cycle, one pixel of the output result is produced, and the pyramid image generated for the specified number of layers is stored in the memory. Fig. 8 represents the pyramid-block process.

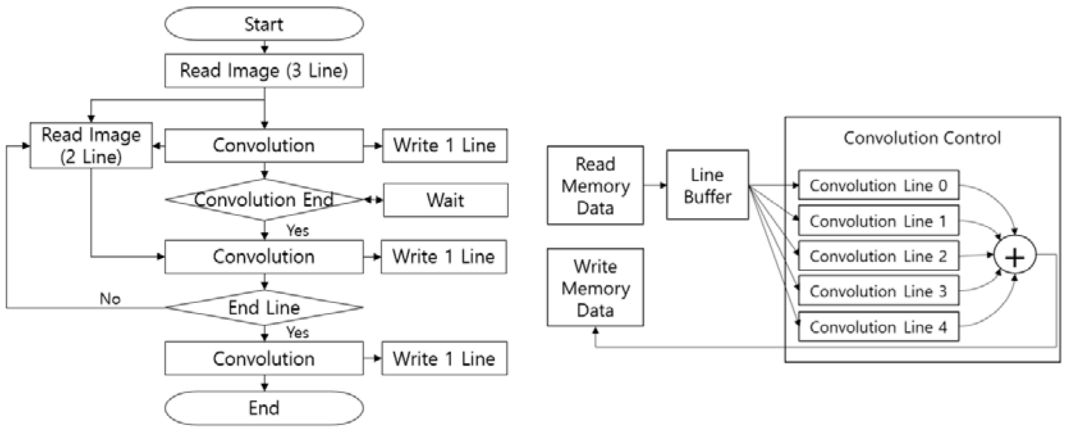

Fig. 8. Pyramid-block process

In the proposed system, the template-matching block requires the maximum data processing and memory. The reference software performs template matching based on cross-correlation. However, a normalization process is applied before directly using the cross-correlation value for template matching. Cross-correlation is calculated as the sum of the multiplications of all pixels in the template region, including the pixels in the overlapping region of the template and the original image; this yields a large value, which depends on the size of the template.

In the reference software, integral images related to the sum and squared sum of the input image are precomputed for normalization. These precomputed images are normalized at each pixel level. Using these integral images requires a high initial computational load, as they must be calculated once for all pixels in the input image. Additionally, as the integral image values accumulate differently for each pixel position and their representation ranges vary, a substantial amount of memory is required for storing these integral images, especially if stored based on the maximum values.

However, once the integral images are calculated at the frame level, regardless of the size variation of the template, the sum of the pixel values at a specific template position can be calculated using the same computational load, which is advantageous. Integral images can be beneficial for optimization because no memory constraints are present from the software perspective. However, from a hardware-implementation perspective, the quality of results may vary depending on the architecture. Moreover, an initial delay exists as the integral images need to initially be calculated for all pixels at the frame level. Therefore, the architecture of the proposed system effectively utilizes processing elements for this operation by performing block-level pipelined processing without using integral images. We directly computed the operations using this design while saving processing time at each step. **Fig. 9** illustrates the processing steps of the matching block. To conserve FPGA resources, only the portion of the source image corresponding to the height of the target image is read from memory and stored in a buffer for matching.

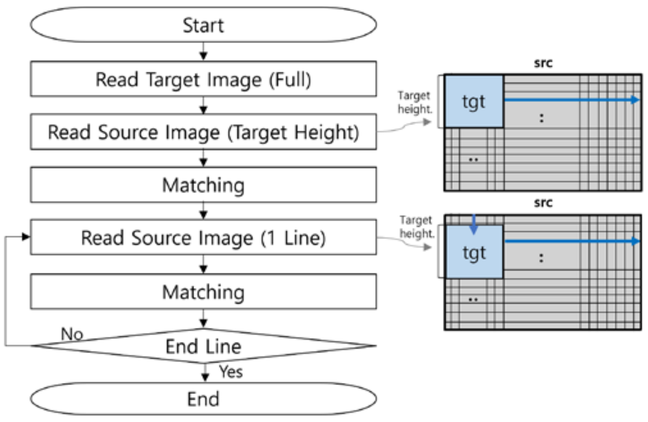

Fig. 9. Process implemented by the matching block

5. System Implementation

The architecture described in Section 4 was implemented, simulated, and synthesized using Xilinx ZCU104. **Fig. 10** and **Fig. 11** show the proposed design synthesized on an FPGA board.

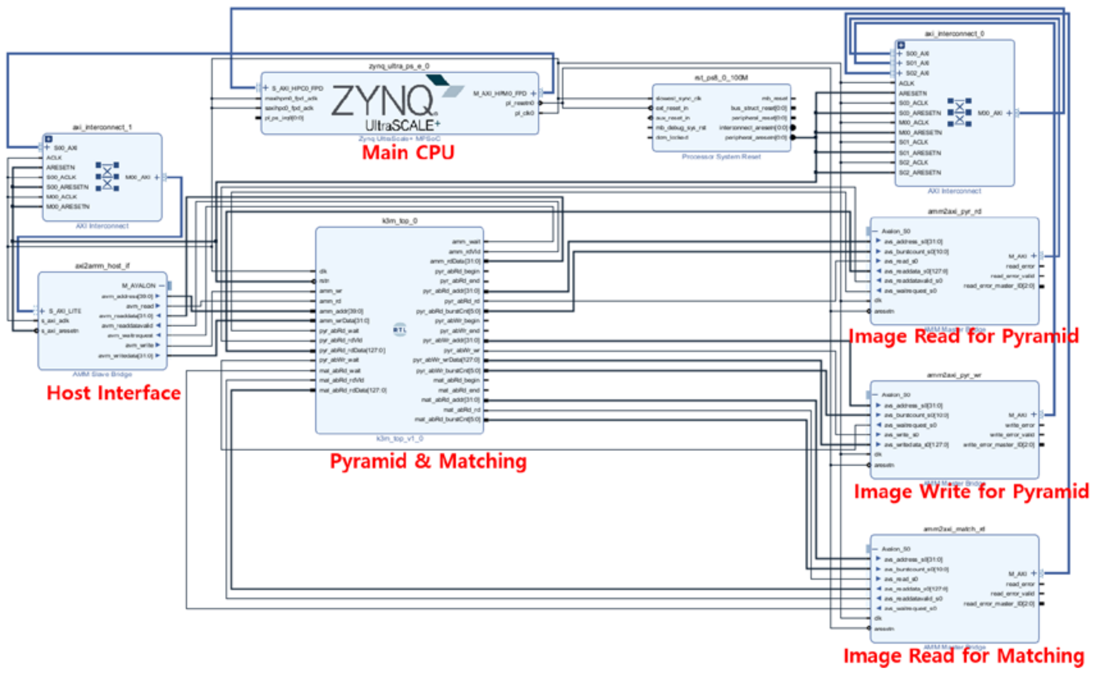

Fig. 10. Diagram of the top block on the FPGA board

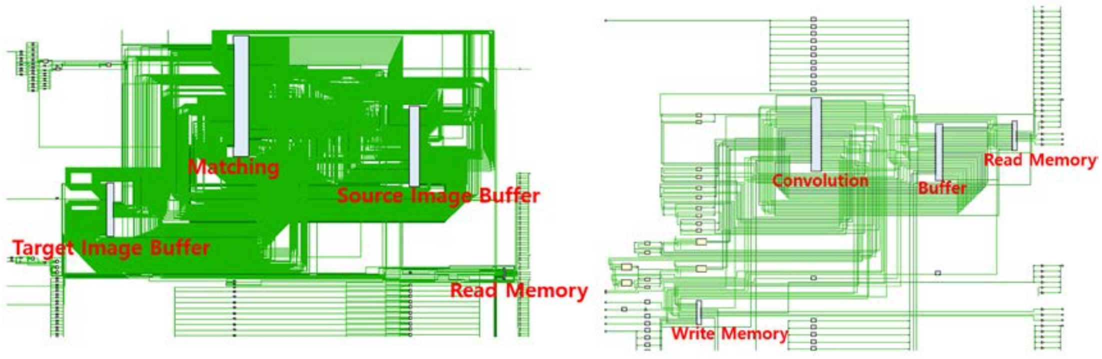

Fig. 11. Schematics of the matching and pyramid blocks

A system implementation apparatus, consisting of a vision module, three-axis motorized stage, and controller, is developed, as shown in **Fig. 12**. The vision module had four 1440 × 1080 vision sensors with 1.4–9.0x zoom lens, which were mounted on a top plate, and each alignment-mark image was considered simultaneously. NCC-based template matching was implemented and evaluated on the Xilinx ZCU104 development board. To link the developed wafer-alignment hardware system with the evaluation board, they were driven by automation processing through alignment API. The raw image data input from the camera sensor was processed in real time by the FPGA board to extract significant information regarding mark translation and rotation. A motorized stage controller controlled the three-axis (X, Y, θ) stage and automatically performs alignment tasks.

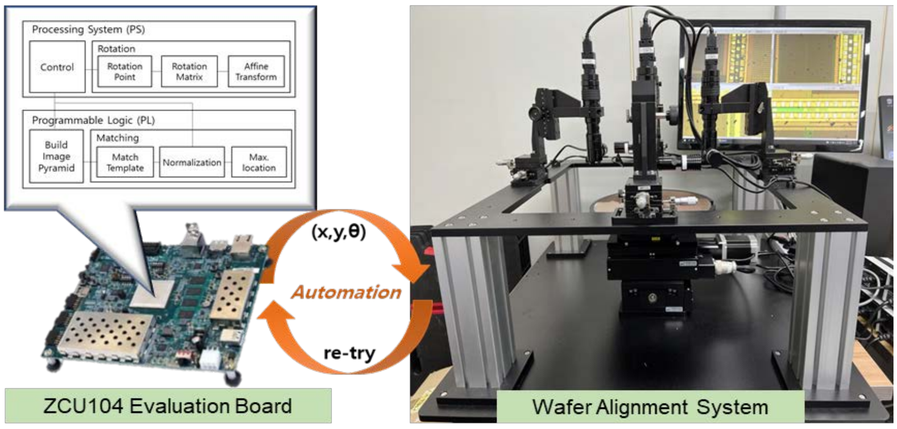

Fig. 12. Configuration of the wafer-alignment system

The system operates at a 200MHz clock frequency, and the utilization of the pyramid block and matching block designed with programmable logic for acceleration is shown in **Table 1**. It can be observed that a large number of Ultra RAMs (URAMs) are utilized in matching blocks where the use of integral images is required due to their larger size.

Table 1. FPGA resource utilization for the designed system

Resource Utilization	Pyramid Block	Matching Block	Total Utilization Ratio (@Xilinx ZCU104)
LUT	3,524	6,527	10,051 / 332,160 (3.02%)
FF	1,586	7,176	8,762 / 460,800 (1.90%)
BRAM	16	-	16 / 312 (5.12%)
URAM	-	52	52 / 96 (54.17%)

The marks on the wafer were identified and misalignment compensation was accomplished in several operation steps. First, when the wafer moved to the work plate on the stage, the vision modules captured the first snap images and detected the positions of the alignment marks. The error related to the wafer's angle of rotation was measured using the captured images. Then, the measured angle was transmitted to the stage controller. The controller performed rotation correction according to the error values. Subsequently, the vision modules captured the second snap images and detected the positions of the alignment marks. The wafer translation error was then measured, and the stage was moved for error compensation. Finally, if the deviation of the wafer position agreed with the reference data, the alignment was considered complete. If not, the alignment controller restarted the process by automatically checking rotation alignment.

Table 2. Alignment results obtained from the reference software

Rotation angle	*X*	*Y*	*θ*	Accuracy (%)	Time (s)
–0.1	719.808	539.536	-0.12	98	0.27
–0.2	717.843	538.492	-0.17	97	0.26
–0.3	715.856	536.346	-0.35	95	0.25
–0.4	713.991	535.304	-0.39	99	0.25
–0.5	713.104	534.033	-0.51	99	0.28
–0.6	711.041	533.288	-0.58	98	0.26
–0.7	710.036	532.254	-0.73	97	0.27

–0.8	707.032	530.048	-0.91	89	0.27
–0.9	706.048	529.032	-0.91	99	0.27
–1.0	704.996	528.332	-1.09	91	0.29
0.1	720.000	539.000	0.16	94	0.28
0.2	717.006	537.014	0.22	98	0.27
0.3	715.994	536.026	0.33	97	0.25
0.4	714.016	535.008	0.39	99	0.29
0.5	712.976	534.048	0.48	98	0.29
0.6	711.235	532.997	0.58	98	0.26
0.7	710.091	532.105	0.65	95	0.28
0.8	707.086	530.559	0.81	99	0.28
0.9	706.016	529.032	0.91	99	0.29
1.0	705.316	527.981	0.93	93	0.26
Average				**96.6**	**0.271**

Table 3. Alignment results obtained from the hardware simulator and FPGA

Rotation angle	***X***	***Y***	***θ***	**Accuracy (%)**	**Time (s) * Simulator**	**Time (s) *FPGA**
–0.1	719.758	539.196	-0.05	95	1.70	0.24
–0.2	718.257	538.744	-0.12	92	0.99	0.18
–0.3	716.351	536.117	-0.39	91	0.92	0.17
–0.4	713.736	535.102	-0.35	95	0.92	0.17
–0.5	712.821	534.181	-0.45	95	0.93	0.17
–0.6	711.098	533.819	-0.56	96	0.86	0.18
–0.7	710.372	532.461	-0.58	88	0.81	0.17
–0.8	706.793	530.507	-0.93	87	0.81	0.17
–0.9	705.744	529.487	-0.92	98	0.88	0.17
–1.0	705.024	528.219	-0.90	90	0.95	0.18
0.1	719.776	539.302	-0.00	90	1.01	0.19
0.2	716.902	537.493	0.22	98	0.90	0.18
0.3	716.479	536.507	0.22	92	0.88	0.17
0.4	713.706	535.684	0.45	95	0.89	0.17
0.5	713.162	533.947	0.45	95	0.95	0.19
0.6	710.841	533.266	0.65	95	0.82	0.17
0.7	709.829	532.278	0.68	98	0.73	0.16
0.8	707.423	530.442	0.76	96	0.83	0.17
0.9	706.021	529.735	0.92	98	0.88	0.17
1.0	705.113	528.667	0.90	90	0.87	0.17
Average				**93.7**	**0.9265**	**0.195**

The results obtained from the alignment system, including those obtained from the reference software and the FPGA designed in this study, as shown in Fig. 12, were summarized for four categories classified as x, y, angle, processing time, and the actual alignment markers used on the wafer were employed [13]. Table 2 lists the results obtained from the reference software, and Table 3 lists the processing times obtained using the hardware simulator and those obtained using the FPGA. The output coded using the hardware simulator resulted in a longer processing time than that obtained using the reference software as the former output was coded at the hardware level without using OpenCV functions. However, on the FPGA, acceleration blocks were applied, and the results were processed rapidly due to the pipeline structure. The experimental results show that the proposed FPGA-based hardware maintains an accuracy error of approximately 3% or less compared to the reference software while achieving a speed improvement of over 30%.

6. Conclusion

This study presents a method for applying the template-matching algorithm to real-time wafer-alignment systems using FPGA-based hardware. A hardware simulator was designed for implementing the template-matching algorithm and the efficiency of the simulator was validated. Based on this, an FPGA-based hardware architecture was developed. The developed system was realized using Xilinx ZCU104 and integrated with vision modules and motorized stages for aligning actual wafers. The experimental results showed that the proposed FPGA-based hardware, while exhibiting slightly decreased accuracy compared to the reference software, successfully implemented high-speed processing by reducing processing time. These findings underscore the utility of FPGA-based template-matching algorithms in real manufacturing environments. Moreover, the efficient optimization using the hardware simulator during the initial design stages facilitated the practical implementation of the system. Therefore, we anticipate the development of advanced wafer-alignment systems that can increase productivity and accuracy in the semiconductor industry. These results emphasize the applicability of FPGA-based hardware implementation in real-world settings.

References

[1] J. KIM, "New Wafer Alignment Process Using Multiple Vision Method for Industrial Manufacturing," *Electronics*, vol.7, no.3, 2018. Article (CrossRef Link)

[2] M. Cong, X. Kong, Y. Du, J. Liu, "Wafer Pre-Aligner System Based on Vision Information Processing," *Information Technology Journal*, vol.6, no.8, pp.1245-1251, 2007. Article (CrossRef Link)

[3] C. Chen, C. Huang, C. Yeh, W. Chang, "An accelerating CPU based correlation-based image alignment for real-time automatic optical inspection," *Computers & Electrical Engineering*, vol.49, pp.207-220, Jan. 2016. Article (CrossRef Link)

[4] B.A. Draper, J.R. Beveridge, A.P.W. Böhm, C. Ross and M. Chawathe, "Implementing image applications on FPGAs," in *Proc. of 2002 International Conference on Pattern Recognition*, vol.3, pp.265-268, 2002. Article (CrossRef Link)

[5] C. T. Johnston, K. T. Gribbon, and D. G Bailey, "Implementing Image Processing Algorithms on FPGAs," in *Proc. of the Eleventh Electronics New Zealand Conference, ENZCon'04*, pp.118-123, 2004. Article (CrossRef Link)

[6] C. Zhao, C. Cheung, and M. Liu, "Integrated polar microstructure and template-matching method for optical position measurement," *Optics Express*, vol.26, vol.4, pp.4330-4345, 2018. Article (CrossRef Link)

[7] W. Ouyang, F. Tombari, S. Mattoccia, L. Di Stefano, W.K. Cham, "Performance Evaluation of Full Search Equivalent Pattern Matching Algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.34, no.1, pp.127-143, Jan. 2012. Article (CrossRef Link)

[8] S. Mattoccia, F. Tombari, L. Di Stefano, "Efficient template matching for multi-channel images," *Pattern Recognition Letters*, vol.32, no.5, pp.694-700, Apr. 2011. Article (CrossRef Link)

[9] W. Shou-Der, L. Shang-Hong, "Fast Template Matching Based on Normalized Cross Correlation With Adaptive Multilevel Winner Update," *IEEE Transactions on Image Processing*, vol.17, no.11, pp.2227-2235, Nov. 2008. Article (CrossRef Link)

[10] S. Sassanapitak, P. Kaewtrakulpong, "An efficient translation-rotation template matching using pre-computed scores of rotated templates," in *Proc. of 2009 6th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, ECTI-CON*, pp.1040-1043, 2009. Article(CrossRefLink)

[11] X. Cui, H. Kim, E. Park, H. Choi, "Robust and accurate pattern matching in fuzzy space for fiducial mark alignment," *Machine Vision and Applications*, vol.24, pp.447-459, 2013. Article (CrossRef Link)

[12] Y. Zhang, Z. Zhang, S. Peng, D. Li, H. Xiao, C. Tang, R. Miao, L. Peng, "A rotation invariant template matching algorithm based on Sub-NCC," *Mathematical Biosciences and Engineering*, vol.19, no.9, pp.9505-9519, Jun. 2022. Article (CrossRef Link)

[13] G. Kertész, S. Szénási and Z. Vámossy, "Performance measurement of a general multi-scale template matching method," in *Proc. of 2015 IEEE 19th International Conference on Intelligent Engineering Systems (INES)*, pp.153-157, 2015. Article(CrossRefLink)

Jaehyuk So received the B.S. degree in electronic engineering from Myongji University, Yongin, Korea, in 2014 and M.S. degree in electrical and electronic engineering from Yonsei University, Seoul, Korea, in 2017. From 2017 to 2021, he was a Senior Engineer at LG Electronics, Seoul, Korea, where he involved in the design and development of video codec IP. He is currently a senior researcher in the Korea Electronics Technology Institute, Korea. His research interests include digital signal processing algorithm and SoC/VLSI implementation for image signal processing.

Minjoon Kim received the B.S. and Ph.D. degree in electrical and electronic engineering from Yonsei University, Seoul, Korea, in 2012 and 2018, respectively. From 2018 to 2024, he was a Senior Engineer at the Korea Electronics Technology Institute, Korea. He is currently an Assistant Professor at the Division of Semiconductor and Electronics Engineering, Hankuk University of Foreign Studies, Korea. His research interests include digital signal processing algorithm and its SoC/VLSI implementation.