Implementation of pick-and-place algorithms for the purposes of wafer handling robotics simulations

Nikolay Bratovanov
Faculty of Automation
Technical University of Sofia
Sofia, Bulgaria
nbratovanov@tu-sofia.bg

***Abstract*—An approach for efficient pick-and-place robotics simulations based on 3D CAD software has been proposed in the paper. Oriented specifically towards wafer handling robots, the development requires the implementation of several algorithms, whose main tasks are associated with determining the proximity between the robot's end-effector and the silicon wafers, as well as checking the end-effector's vacuum state. As a result, realistic object manipulations, closely matching the real-world substrate handling process can be performed, analyzed and evaluated in a completely virtual environment. The proposed approach has been successfully adapted to an already existing robot simulator based on SolidWorks API and Visual Basic .NET programming.**

Keywords—pick-and-place, wafer handling robotics, object manipulation, robot motion simulation, offline programming, SolidWorks API

I. Introduction

Wafer handling robots have been aiding electronic device manufacturing already for decades [1], [2]. Their primary role in this highly automated, advanced and demanding industry, which is becoming increasingly complex and sophisticated, is transporting the substrates (generally silicon wafers) between the different processing machines in a swift, smooth and safe manner, while maintaining superior accuracy and reliability requirements [3], [4]. Considering the importance of this main functionality, as well as the successful development of various robot simulators, it becomes clear that the execution of wafer handling simulations would be extremely beneficial both for robot manufacturing companies and customers. Implementing such a feature to an already available robot simulator based on the popular 3D CAD software SolidWorks and its application-programming interface (API), which has been created by the same author [5], is regarded as the main objective of this work. The task is associated with the development of several pick-and-place algorithms, which must be adapted to the mentioned simulator via VB.NET programming language. As a result, its overall functionality has been significantly enhanced, turning it into a powerful tool for realistic motion simulations, offline programming, complete customer layout analysis, evaluation, marketing and demonstration purposes.

II. Formal Description of the Wafer Handling Process

Ignoring the great variety of wafer handling robots, types of end-effectors and substrates for manipulation, as well as all the specific requirements imposed by the automated task, the wafer handling process itself can be considered as a standard pick-and-place operation inherent to the majority of industrial robots. In this regard, two main components necessary for the formal description of the process are taken into account – the object for manipulation/transportation, and the device that is handling the object. As already mentioned, the most common objects for manipulation in the field of semiconductor device manufacturing are the silicon substrates known as wafers [6]. These are thin disks with different diameter (i.e. 100, 150, 200, 300 & 450mm), which are used as a basis for the fabrication of integrated circuits, photovoltaics, etc. On the other hand, in addition to the most obvious handling device – the robot end-effector – the semiconductor industry automation is also being facilitated by another specific device, known as a prealigner, which also interacts with the silicon wafers in the course of operation of the automated tools (Fig. 1). It is important to be noted that the working principle of both devices (end-effector and prealigner) is most commonly based on vacuum [7].

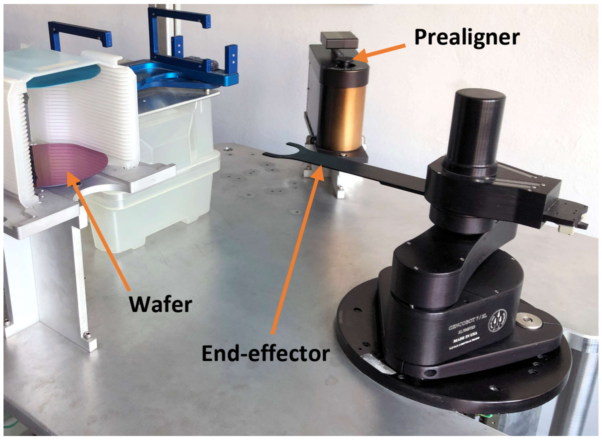


Fig. 1. Wafer handling-related components – wafer, robot end-effector and prealigner.

In order to be successfully simulated, the nature of the real-world wafer handling process must be analyzed. Let us review the basic layout shown in Fig. 1. If the wafer that is present inside the carrier is to be picked by the end-effector, a 'GET' command has to be executed via the robot's motion control software (MCS). What happens as a result can be summarized as follows: the end-effector starts moving towards the wafer in a predefined manned, based on specially designed motion patterns/trajectories; once the end-effector has reached below the wafer (the so-called 'pickup' position), the vacuum valve responsible for generating the end-effector's grasping force opens (*eeVacValve [0]* = 0) and the end-effector starts moving towards the wafer in a vertical manner; as the two objects get sufficiently close to each other, the vacuum causes the wafer to stick to the contact surface of the end-effector's horseshoe, thus the wafer handling procedure is being executed. In order to place an already grasped wafer the corresponding motion pattern (initiated via the 'PUT' command) takes place where, similarly to the GET-procedure, the closure of the vacuum valve (*eeVacValve [0]* = 1) determines the exact moment of releasing the wafer from the end-effector.

The ambition of the current work is to develop an approach for realistic wafer handling simulations observing the above principle. Obviously, in order to do so two procedures have to be considered – ‘wafer-to-gripper’ proximity calculation and vacuum valve state monitoring. While the second one is going to rely on relatively simple I/O communication between the SolidWorks API-based simulator and the robot MCS, the first procedure imposes the development of several mathematical models and algorithms whose purpose is to determine what is the spatial proximity between the gripper (robot end-effector or prealigner) and each wafer available for handling. For this reason, let us introduce two main components, referred to as ‘wafer’ and ‘gripper’, with coordinated frames $\mathrm{O}_w\,\mathbf{e}_{w1}\mathbf{e}_{w2}\mathbf{e}_{w3}$ and $\mathrm{O}_g\,\mathbf{e}_{g1}\mathbf{e}_{g2}\mathbf{e}_{g3}$, as shown in Fig. 2.

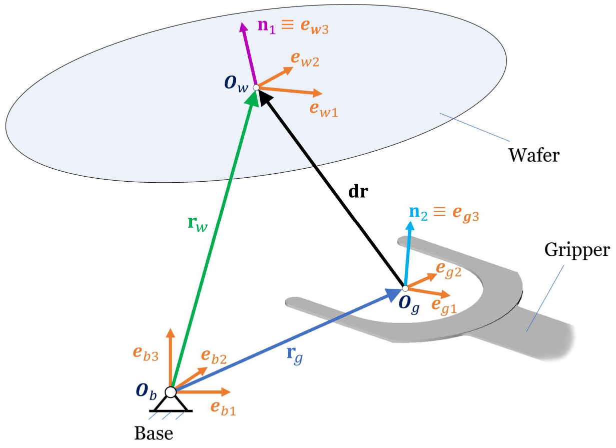


Fig. 2. Graphical representation of the two main components related to the wafer handling process.

The wafer and the gripper’s centers with respect to the base coordinate frame $\mathrm{O}_b\,\mathbf{e}_{b1}\mathbf{e}_{b2}\mathbf{e}_{b3}$ are represented by the vectors $\mathbf{r}_w$ and $\mathbf{r}_g$ respectively (known and taken directly from their transformation matrices). It is obvious from the same figure that the vector $\mathbf{dr}$ connecting the frame origins O_w and O_b can be expressed as:

$$\mathbf{dr} = \mathbf{r}_w - \mathbf{r}_g \tag{1}$$

In addition to the vector $\mathbf{dr}$, which will be later used as a basis for determining if the gripper is close enough to the specific wafer (both in vertical direction, as well as horizontally), two other vectors – $\mathbf{n}_1$ and $\mathbf{n}_2$ – must be reflected for the purpose of determining the parallelism deviations (difference in the orientation) between the two components. As it can be seen in Fig. 2, $\mathbf{n}_1$ and $\mathbf{n}_2$ are the normal vectors of the wafer and the gripper, respectively. The last can be derived from the rotation matrices ${}^b_w\mathbf{R}$ and ${}^b_g\mathbf{R}$ (taking their third columns), expressing the orientation of the wafer and the gripper with respect to the base coordinate frame O_b.

Knowing the above, it is already possible to perform some basic vector operations that would provide all the information (the scalars $n_{1\mathrm{projn}_2}$, dr_{projZ}, and the vector $\mathbf{dr}_{\mathrm{projXY}}$), needed for the wafer handling simulation procedure. The last can be expressed as follows, considering the relationships visualized in Fig. 3:

$$n_{1\mathrm{projn}_2} = \mathbf{n}_1 . \mathbf{n}_2 \tag{2}$$

$$dr_{\mathrm{projZ}} = \mathbf{dr} . \mathbf{n}_1 \tag{3}$$

$$\mathbf{dr}_{\mathrm{projXY}} = {}^b_w\mathbf{R}^{\mathrm{T}} . \mathbf{dr} \tag{4}$$

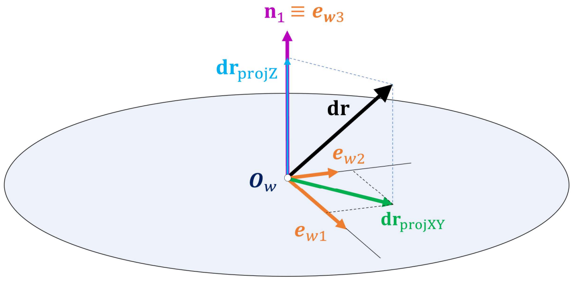

$\mathbf{n}_1 \equiv \boldsymbol{e}_{w3}$
$\mathbf{dr}_{\mathrm{projZ}}$
$\mathbf{dr}$
$\boldsymbol{e}_{w2}$
$\boldsymbol{O}_w$
$\mathbf{dr}_{\mathrm{projXY}}$
$\boldsymbol{e}_{w1}$

Fig. 3. Graphical representation of the **dr** vector projection.

The final computational step consists of deriving the norm of the $\mathbf{dr}_{\mathrm{projXY}}$ vector in the XY plane, so that it can be further utilized in determining the wafer-to-gripper horizontal offset:

$$\left\|\mathbf{dr}_{\mathrm{projXY}}\right\| = \sqrt{\mathbf{dr}_{\mathrm{projXY}_1}{}^2 + \mathbf{dr}_{\mathrm{projXY}_2}{}^2} \tag{5}$$

The formulation of (2), (3) and (5) and the knowledge of their output contributes to the implementation of the wafer handling simulation algorithm based on comparing the derived values with explicitly defined reference parameters. The adaptation of the proposed approach to an existing robot simulator based on SolidWorks API is presented in the following chapter.

III. Adaptation of the Pick-and-Place Algorithms to an Existing VB.NET-Based Robot Simulator

As already mentioned, the ambition of the current work is associated mainly with extending the functionality of a CAD-based simulator (developed by the same author), utilized for motion simulation and offline programming of wafer handling robots. Hence, the relationships describing the wafer handling process as well as the I/O communication between the MCS and SolidWorks have to be implemented and adapted into the simulator’s VB.NET-based source code. In addition, all wafer handling-related components have to be introduced into the existing SolidWorks 3D assemblies observing that all frames are attached exactly as described in Chapter II.

A. Wafer handling-compliant 3D modeling

Obviously, the simulation of the wafer handling process requires the availability of 3D models of all wafers available for handling, as well as 3D models of the grippers (robot end-effector, prealigner, or both). As already mentioned above, the most important consideration at this point is making sure that the coordinate frame of each component/3D model is located and oriented as imposed by the formal description of the wafer handling process (Fig. 4).

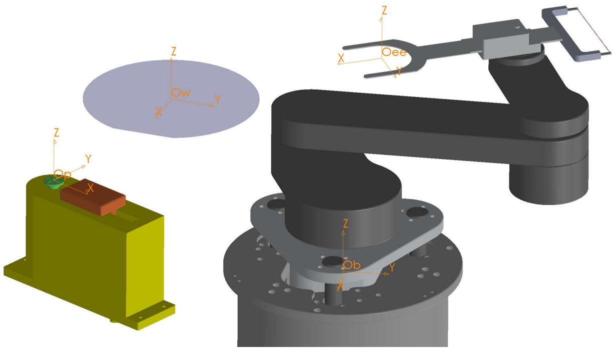


Fig. 4. 3D models of all main wafer handling-related components.

Once all components are available in the simulation assembly their geometrical features and corresponding relationships are derived into the simulator's VB.NET-based source code (via various SolidWorks API methods and interfaces), thus making sure that all the basic information necessary for the 'wafer-to-gripper' proximity check has been provided.

B. *'Wafer-to-gripper' proximity check*

The availability of wafer/gripper geometrical information (such as position vectors and rotation matrices with respect to the base frame of the assembly) allows the implementation of the most important procedure responsible for determining the proximity between the gripper and the wafer for handling (also referred to as '*checkReadinessToPick*' procedure). In addition to featuring and resolving the equations displayed in Chapter II, the procedure deals with comparing the calculated values with the following explicitly defined reference parameters responsible for determining the 'wafer-gripper' relationship at each time-slice of the motion simulation:

- $\boldsymbol{\varepsilon}_{\mathbf{projZ}}$ – used for defining the vertical deviation (is the gripper too high or too low with respect to the wafer's contact surface).
- $\boldsymbol{\varepsilon}_{\mathbf{projXY}}$ – used for defining the horizontal deviation (is the gripper's center too far off the wafer's center in the horizontal plane).
- $\boldsymbol{\varepsilon}_{\mathbf{planar}}$ – used for defining the orientation deviation (the degree of parallelism between the gripper's plane and the wafer's contact surface).

Hence, in order to determine if a particular wafer is ready to be picked by the specific gripper (end-effector/prealigner) at a given moment of the simulation process, the following three conditions have to be continuously evaluated and met:

$$|dr_{\text{projZ}}| < \boldsymbol{\varepsilon}_{\mathbf{projZ}} \tag{6}$$

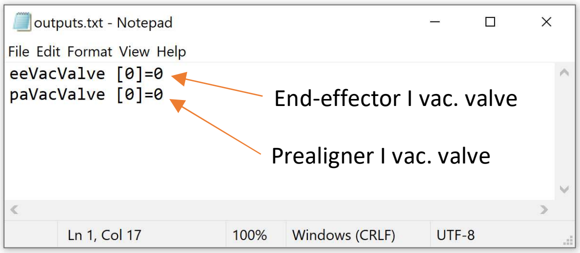
$$\|\mathbf{dr}_{\text{projXY}}\| < \boldsymbol{\varepsilon}_{\mathbf{projXY}} \tag{7}$$

$$n_{\text{projn}_2} > \boldsymbol{\varepsilon}_{\mathbf{planar}} \tag{8}$$

The actual picking of the wafer, however, is executed only if an additional fourth condition has been satisfied. It is related to the current state of the gripper's vacuum valve. This check requires the implementation of another sub-procedure, whose concept has been presented in the following subsection.

C. *Vacuum valve state monitoring*

Seeking to represent/simulate the real-life wafer handling process as closely as possible, its implementation requires the development of a specific sub-procedure, which monitors the state of the vacuum valve of each gripper available in the 3D assembly (i.e. several robot end-effectors or prealigners). The state monitoring process relies on a specific communication between the simulator (SolidWorks API) and the robot MCS, based on text file I/O. The following concept has been used: the robot MCS generates a text file ('*outputs.txt*') containing a number of parameters, equal to the number of grippers (end-effectors/prealigners) available in the particular 3D assembly. Each parameter has two states corresponding to the state of the vacuum valve – either "0" (vacuum valve is open), or "1" (vacuum valse is closed). Obviously, as the real-life process imposes, the successful wafer handling is associated not only with the proper wafer-to-gripper relative position/orientation, but also with the actual state of the gripper's vacuum valve. It is therefore essential for the successful implementation of the wafer handling simulation that all parameter values in the '*outputs.txt*' file are being continuously read and processed by the simulator. This is facilitated by a simple sub-procedure referred to as '*Outputs*' whose main task is associated with selecting, splitting and exporting the values of each parameter contained in the file (Fig. 5).



```
eeVacValve [0]=0
paVacValve [0]=0
```

End-effector I vac. valve

Prealigner I vac. valve

Fig. 5. Vacuum valve state monitoring parameters stored in '*outputs.txt'*.

Finally, the proposed wafer handling simulation approach can be summarized as follows:

- The proximity check represented by (6), (7) and (8) is generalized by a variable (type Boolean) referred to as '*isReadyToPick*XX', which becomes *True* if all the conditions are met and vice versa.
- The vacuum valve state check procedure retrieves the current values of all related parameters and stores them into the 'xx*VacValve*_x' variable (type Double).
- If '*isReadyToPick*XX' = *True* & 'xx*VacValve*_x' = 0, it is assumed that the particular wafer (i) is grasped by the corresponding gripper at that very moment of the code execution: '*getWafer*XX(i)' = *True*, where $i = 1, 2, \dots, \text{n} - 1, \text{n}$ (n – number of wafers available in the 3D assembly).
- Once '*getWafer*XX(i)' = *True*, the coordinate frame of the specific wafer becomes fixed with respect to the coordinate frame of the active gripper until the value of the corresponding vacuum valve changes to 1 (when a 'PUT' command for placing the wafer is executed in the robot MCS). This relative temporary fixture of the two coordinate frames involves a sequence of math operations (based on position vectors and rotation matrices) associated with calculating the instantaneous transformation data reflecting the gripper's arbitrary position/orientation with respect to the wafer at the moment of handling.

Integrating the above concept into the continuous motion loop and executing it for each wafer available in the 3D assembly guarantees the efficiency and responsiveness of the proposed approach for simulation of the wafer handling process. The last is verified and demonstrated with the help of an exemplary 3D model, which has been thoroughly reviewed and analyzed in the following chapter.

IV. Simulation and Verification of the Proposed Approach

The verification of the proposed approach for simulation of the wafer handling process imposes the setup and utilization of an exemplary SolidWorks 3D assembly, comprised of the following main components, as shown in Fig. 6:

- GB8Y wafer handling robot with a single end-effector (vacuum-based).
- Remote Prealigner with vacuum gripper (chuck).
- Two 300mm wafers for transportation/handling.
- Additional components e.g. wafer carriers (FOUPs), Load Ports, table, etc.

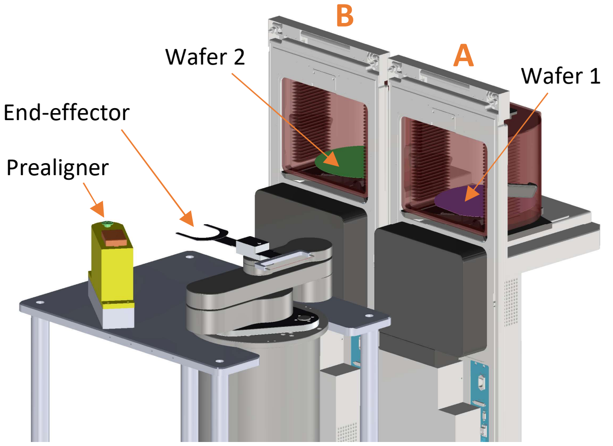


Fig. 6. SolidWorks 3D assembly used for verification and demonstration of the proposed approach for wafer handling simulation.

Two main task associated with confirming the functionality, accuracy and efficiency of the proposed VB.NET algorithms and procedures for wafer handling simulations are considered – robot wafer handling/transportation and wafer prealigning. Some of the key verification steps associated with both task seek to confirm the reliability of the communication between SolidWorks and the robot's MCS, as well as the precision of the arbitrary wafer handling concept.

A. *Simulation of robot wafer handling/transportation*

The first simulation scenario involves the performance of the following sequence:

*1) GET 'Wafer 1' from Station 'A' (slot 1): S*tation A has been intentionally tilted (1.5°) in order to simulate the ability of the end-effector to handle titled objects; the wafer has been centrally placed inside the FOUP (no horizontal offsets), as shown in Fig. 7b.

2) PUT 'Wafer 1' in Station 'B' (slot 2): Station B has been properlly oriented (no tilt).

3) GET 'Wafer 2' from Station 'B' (slot 1): the wafer has been intentionally placed with some horizontal offsets inside the FOUP ($dX = 13.794\text{mm}$; $dY = 7.2385\text{mm}$), as shown in Fig. 7a.

4) PUT 'Wafer 2' in Station 'B' (slot 10).

Assuming that all wafer handling algorithms and procedures have been properly implemented, the following results are to be expected:

- Placing 'Wafer 1' in Station B (step 2) should result in null horizontal offsets ($dX = 0\text{mm}; dY = 0\text{mm}$).
- Placing 'Wafer 2' in Station B (step 4) should result in the same horizontal offsets as given in step 3.
- It must be visually confirmed that the wafer handling process has been smoothly and accurately performed.

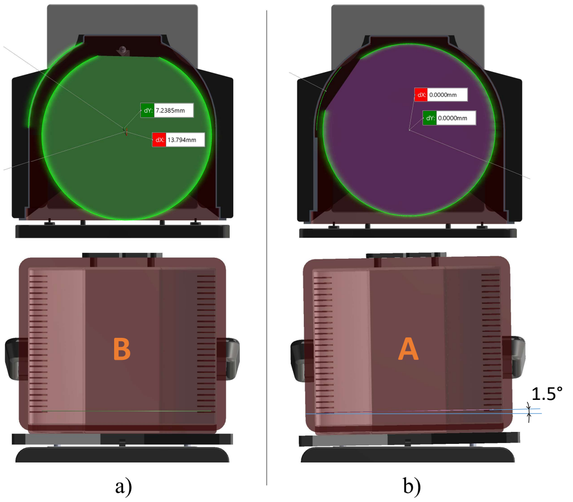


Fig. 7. Stations A & B used in the first wafer handling simulation task.

After setting the mentioned 3D layout, teaching the necessary stations and executing the corresponding commands, all steps were confirmed to be performed as expected. The verification has been achieved with the help of the dedicated 'Measure' tool for taking various dimensions, as well as by means of visual inspection of the wafer handling simulation process. The following reference parameters ($\boldsymbol{\varepsilon}$) have been determined and utilized for the first simulation scenario: $\boldsymbol{\varepsilon}_{\mathbf{projZ}} = 0.5$, $\boldsymbol{\varepsilon}_{\mathbf{projXY}} = 50.0$, $\boldsymbol{\varepsilon}_{\mathbf{planar}} = 0.99$.

B. *Simulation of prealigner wafer handling*

The ambition of the second simulation scenario is to verify the functionality of the other main wafer handling procedure associated with the prealigner. Similarly to the end-effector, the prealigner also has to provide grasping capabilities so that the silicon wafer could be rotated about the axis of its gripper (chuck) and its edge could be detected by a specially designed sensor, known as lighthouse. The purpose of this procedure is to find the wafer's center and determine its precise orientation so that it can be accurately delivered to the various processing tools in the automated cell. Even though this functionality has not been completely simulated yet, it is important to confirm that the wafer handling process (carried out by the prealigner's chuck) as well as the wafer transfer between the end-effector and the prealigner can be successfully performed. Hence, the following motion sequence and simulation layout, displayed in Fig.8 is considered:

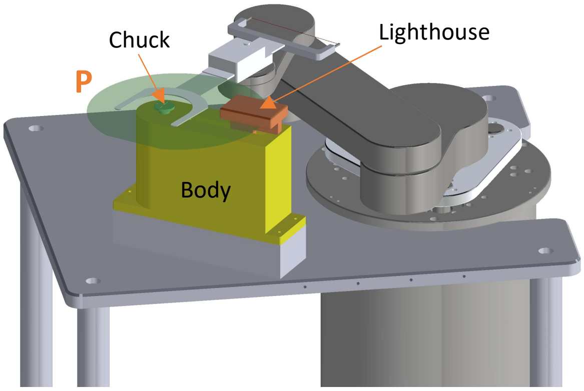


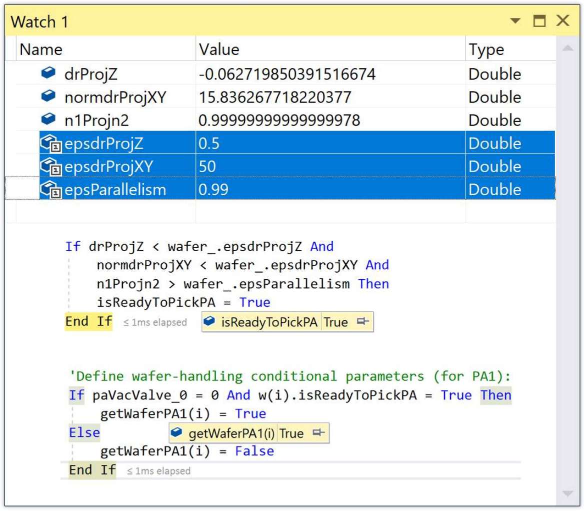
Fig. 8. Station P (prealigner) used in the second simulation task.

1) GET 'Wafer 2' from Station 'B' (slot 10).

2) PUT 'Wafer 2' in Station 'P' (slot 1): Station P has been explicitly taught as a 'prealigner' station.

3) Rotate the prealigner chuck together with the already grasped wafer: seeks to confirm the successful simulation of the prealigner's wafer handling capabilities.

As expected, the execution of the second simulation task confirms the reliability and efficiency of the prealigner wafer handling VB.NET procedure (not a surprise, considering the identical development concept for both prealigner and robot end-effector). The only significant difference is related to the utilization of a separate vacuum valve parameters associated with the current state of the prealigner's chuck (*paVacValve [0]*), as well as with the precise timing and synchronization of switching the *eeVacValve* & *paVacValve* values during the wafer transfer process. In addition, the values of all variables (introduced in Chapter II) at the moment of wafer grasping by the prealigner's chuck (step 2) are analysed and compared to the already determined in the first simulation task reference parameters (**ε**), as shown in Fig. 9.



Name	Value	Type
drProjZ	-0.062719850391516674	Double
normdrProjXY	15.836267718220377	Double
n1Projn2	0.99999999999999978	Double
epsdrProjZ	0.5	Double
epsdrProjXY	50	Double
epsParallelism	0.99	Double

```
If drProjZ < wafer_.epsdrProjZ And
    normdrProjXY < wafer_.epsdrProjXY And
    n1Projn2 > wafer_.epsParallelism Then
    isReadyToPickPA = True
End If  ≤ 1ms elapsed   isReadyToPickPA True


'Define wafer-handling conditional parameters (for PA1):
If paVacValve_0 = 0 And w(i).isReadyToPickPA = True Then
    getWaferPA1(i) = True
Else                 getWaferPA1(i) True
    getWaferPA1(i) = False
End If  ≤ 1ms elapsed
```

Fig. 9. Values of the prealigner wafer handling-associated variables (white) and reference parameters (blue) at the moment of wafer placing in station P.

Obviously, the conditions necessary for a wafer to be picked by the chuck (shown in the middle of Fig. 9), which have been established in Chapter II, are satisfied ('*isReadyToPickPA*' = *True*). This, combined with the open state of the prealigner's vacuum valve (*paVacValve [0] = 0*) confirms the successful wafer handling simulation process ('*getWaferPA*(*i*)' = *True*).

V. Conclusions and Future Work

The successful execution of both simulation experiments confirms the effectiveness and reliability of the implemented algorithms and procedures for wafer handling simulations adapted to the existing SolidWorks API-based simulator for wafer handling robots via VB.NET-based programming. The proposed approach featuring formal description of the wafer handling process, basic mathematical operations, SolidWorks API-compliant programming, 3D modeling of wafer handling related components, proper coordinate frame attachments, I/O communication between SolidWorks and the MCS (based on text file reading/writing), etc. contributes to the significant improvement of the simulator's functionality and efficiency, allowing complex automated tools for semiconductor device manufacturing, as well as various customer scenarios to be profoundly simulated, analyzed, programmed, optimized and presented. Despite the proposed approach has been explicitly oriented towards a particular industrial robotics application, it is important to emphasize on its universal nature, allowing the implementation and efficient simulation of various pick-and-place robotic tasks, observing the same modeling principles as described in this paper.

Some of the future developments related to the current task include implementation of other robot gripper types (i.e. edge gripping/mechanical end-effectors), flipping wafer handling support, complete prealigning process simulations, as well as optimization of the wafer handling-related algorithms.

Acknowledgment

This work is supported by the Technical University of Sofia, Bulgaria under Research Project of the Ministry of Education and Science "Young scientists and postdoctoral fellows"/2021.

References

[1] K. Mathia and M. P. Aalund, "Robotics for electronics manufacturing in cleanroom environments," Next Wave Of Technology Workshop, July 2010.

[2] H. Chen, H. Cheng and B. Mooring, "Improving wafer handling performance in semiconductor manufacturing," Industrial Robot, vol. 40, No. 5, pp. 425–432, August 2013.

[3] M. Cong, Y. Du, B. Shen and L. Jin, "Robotic wafer handling systems for integrated circuit manufacturing: A review,", pp. 261–266, May 2007.

[4] M. Cong and D. Cui, "Wafer-Handling Robots and Applications," Recent Patents on Engineering, vol. 3, issue 3, pp. 170–177, 2009.

[5] N. Bratovanov, "Robot Modeling, Motion Simulation and Off-line Programming Based on SolidWorks API," Third IEEE International Conference on Robotic Computing (IRC), pp. 574–579, March 2019.

[6] G. Fisher, M. R. Seacrist and R. W. Standley, "Silicon Crystal Growth and Wafer Technologies," Proceedings of the IEEE, vol. 100, no. Special Centennial Issue, pp. 1454–1474, May 2012.

[7] Y. Liu, M. Xu and Y. Cao, "Research, design and experiment of end effector for wafer transfer robot", Industrial Robot, vol. 39, no. 1, pp. 79–91, 2012.