2026 ASEE ANNUAL CONFERENCE June 21–24, 2026 | Charlotte Convention Center, NC
Host Sponsor UNC Charlotte
ASEE AMERICAN SOCIETY FOR ENGINEERING EDUCATION
Paper ID #53565

Leveraging Digital Twins for Industrial Automation and Robotics Education

Zachary Jester, Pennsylvania State University
Katie Fitzsimons, Pennsylvania State University
Ilya Kovalenko, Pennsylvania State University

Ilya Kovalenko is currently an Assistant Professor in the Department of Mechanical Engineering and the Department of Industrial & Manufacturing Engineering at Penn State University. He received both his PhD in Mechanical Engineering (2020) and his MS degree in Mechanical Engineering from the University of Michigan (2018), and his BS degree in Mechanical Engineering from the Georgia Institute of Technology (2015). His current research interests lie in the areas of control theory, artificial intelligence, and smart manufacturing, with a focus on cooperative control, cyber-physical systems, and robotics.

Hongliang Li, Pennsylvania State University
Brian Zajac, Pennsylvania State University

Brian Zajac is currently a Professor of Practice & Director of Industry Relations for the Harold & Inge Marcus Department of Industrial & Manufacturing Engineering at The Pennsylvania State University. He teaches courses in engineering design, additive manufacturing, and industrial robotics. Before pivoting to academia, Brian spent over 15 years in industry developing robotic systems for corporate and government sponsors. At the National Robotics Engineering Center (NREC) in Pittsburgh, he developed prototypes for Shell Oil, Anglo American, the US Army, and DARPA. Brian joined Uber in 2015 when the Advanced Technologies Group (ATG) was founded to develop autonomous vehicles. At ATG, he held several technical and executive leadership positions, including Director of Hardware Engineering and Director of Systems Engineering and Testing. Brian holds a Bachelor's and Master's degree in Mechanical Engineering from Cornell University.

©American Society for Engineering Education, 2026

WIP - Leveraging Digital Twins for Industrial Automation and Robotics Education

Abstract

The workforce shortage in manufacturing poses the urgent need for educating engineers who can understand and interface with industrial automation and robotics. However, high equipment costs, limited lab access, and time constraints present major barriers to hands-on instruction in undergraduate courses. Digital Twins (DTs), purpose-driven virtual representations of physical systems, offer a promising solution. While DTs have been widely adopted in industry for virtual commissioning and predictive maintenance, their application in education remains limited due to a lack of defined use cases and proven results. This work presents the integration of DT technology into an undergraduate industrial automation and robotics course. We leverage the ProtoTwin software to create a high-fidelity DT of a modular production line composed of fischertechnik industrial education kits and a 4-axis robotic arm. Our DT supports both fully simulated and hardware-in-the-loop configurations, enabling students to design, test, and refine solutions in a virtual environment before committing to hardware. This flexibility addresses limited lab availability while allowing students to iterate more rapidly, including outside of scheduled class hours.

1 Introduction

As manufacturing increasingly incorporates advanced technologies to meet customer requirements and respond to supply chain disruptions, there is a critical need for engineers who can work with robotics and automation[1,2]. Automation engineers must be able to both conceptually and practically develop automated workflows, requiring proficiency in high-level process planning and low-level equipment control[3]. Training students on real production systems can equip students with these skills, which have been showcased in the existing literature[4,5]. However, real systems are expensive and time-consuming to purchase, utilize, and maintain, making it difficult to provide hands-on training to more than a few students at a time[6].

Educators have addressed this challenge by developing accessible virtual tools that represent real or hypothetical production systems[7]. However, these tools typically focus on either high-level process flow or low-level equipment behavior, but rarely both. Some rely on purely mathematical models that simulate system interactions by updating state variables[8], while others assemble abstracted components in a 3D environment but permit only supervisory-level control[9]. Due to their limited state representations and interactivity, such models have a restricted capacity to exhibit complex or emergent behavior—for example, objects often move only linearly along fixed conveyor paths[10]. Conversely, high-fidelity simulations that closely replicate physical systems in 3D

have been developed for individual components such as industrial robot arms[11], but their narrow scope prevents students from experiencing integrated, multi-device production systems. This separation between high-level and low-level tools prevents students from understanding how decisions at one level propagate through the manufacturing hierarchy. Scaling accurate simulation to an entire production line would enable students to design, test, and iteratively refine control programs in a virtual environment before deploying them to physical hardware. Achieving this requires a full-scale simulation, allowing all devices to operate together as an integrated system.

To meet this need, we designed an undergraduate course titled "Introduction to Automation and Industrial Robotics," cross-listed in the Mechanical Engineering and Industrial Manufacturing Engineering departments at Pennsylvania State University. The course was a 400-level class intended for fourth-year students but was open to any students in either department. It was first offered in the Fall 2025 semester, and attracted 20 Mechanical Engineering and 5 Industrial & Manufacturing Engineering students. As prerequisites to register for this course, students were required to have completed an introductory programming course (C++ or MATLAB) to ensure they were familiar with programming concepts. All students were also required to have taken a junior-level design and manufacturing course, either "Product Design and Manufacturing Processes" or "Product Design, Specification, and Measurement." As part of the course, we developed a high-fidelity Digital Twin (DT) of a modular production system. Following existing literature[12], we define the DT as a virtual, executable representation of a manufacturing system linked to physical equipment through real or emulated sensor and control data. The DT simulates physical interactions and control behaviors and mirrors Input/Output (I/O) interfaces, enabling seamless interchangeability with the physical production line. This DT was created to significantly decrease student reliance on physical automation systems throughout the course, allowing it to be held with far fewer purchased systems. The DT integrates the physical system with its virtual counterpart to support instructional use with fewer copies of the physical hardware. By using the compatible systems we have selected and created, students can work independently of each other's schedules and hardware availability, allowing us to run all PLC labs and projects with only two or three physical production lines set up. The major contributions of this work are: (i) identifying design principles for constructing a DT for manufacturing education; (ii) establishing guidelines for integrating the DT into course activities; and (iii) validating its impact through deployment in an automation course.

2 Hardware and Software Specification

The course covers PLC programming, industrial robot control, and system integration, with students programming a complete multi-device production line from low-level actuation to higher-level system coordination. A central objective is to ensure that students design, implement, and validate control logic for the full system rather than only interacting with isolated subsystems. To enable scalable course delivery, we developed a high-fidelity DT that allows development and testing in a virtual environment, eliminating the need for dedicated equipment per student.

Hardware and software selection were driven by tight integration requirements between the physical system and its DT. The physical production line is controlled by an industrial PLC, and the chosen platform supports execution of identical control code on both the physical PLC and a high-fidelity virtual counterpart, enabling switching between real and simulated operation. The simulation software was selected to balance modeling fidelity with accessibility – it should accurately

represent physical behavior and component interactions while remaining straightforward to install on student-owned devices.

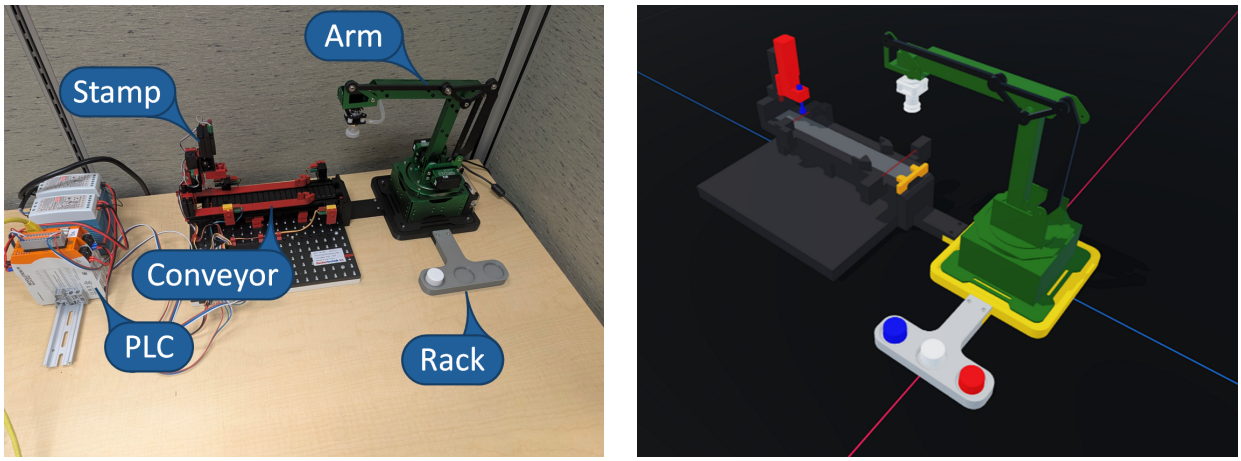
The virtual environment mirrors the I/O configuration of the physical system, allowing control code to transfer without modification. A single-source control architecture was adopted for both simulated and physical robotic arms to reduce development complexity. Communication between system components uses the Open Platform Communications Unified Architecture (OPC-UA) protocol, which exposes selected variables through a local server, allowing external programs and devices to read and write shared variables in a standardized, industry-relevant manner.

3 Hardware and Software Selection

Before the course started, we evaluated several options for the control hardware and software we would use. To ensure students received industry-relevant experience, the selection process was anchored to a strict requirement to utilize an industrial-style PLC. Our decisions for the PLC and Integrated Development Environment (IDE) were closely linked, primarily due to closed software families. Many of the major PLC manufacturers, such as Siemens and Beckhoff, created their own IDEs and PLC simulators, closing their ecosystem to external products. The fully and partially open-source PLC manufacturers we tested were more open with their software selection, allowing us to use more readily available software. Ultimately, we determined that the Kunbus RevPi platform was the best option despite its higher cost compared to other hardware alternatives, and the virtual environment was developed in the ProtoTwin Connect simulation software. A detailed description of the PLC and IDE selection criteria can be found in Appendix 1.

Our goal was to train students on a realistically-controlled physical production line, so the fischertechnik (FT) training family of products[13] was selected as the primary physical production system students would automate. Each module simplified wiring by condensing all inputs and outputs to a single terminal block. Almost all controls ran on 24v digital logic and could be directly hooked into single ports on one digital I/O module on the PLC; one component had a color sensor which output an analog voltage and thus required an analog input module to be purchased for the PLC. We used a multi-axis robotic arm (HIWONDER MaxArm servo-actuated platform) with 3D printed racks/chutes for part input/output.

CAD models and custom scripts defining the geometry and functionality of FT stamping line components and robotic arm were created and imported into ProtoTwin, which supported simulation of collisions, limit switches, and sensors (e.g., beam sensors). To enable students to program in the same environment as the digital twin a custom control scheme for the digital and physical robot arms was developed that enabled students to prescribe analog position in 4 axes (X, Y, Z, and rotation) as well as the state of the suction cup gripper by manipulating template variables. The rationale for the decision to use ProtoTwin and a comparison with other options can be found in Appendix II. Details of our virtual model implementation and custom control schemes created to enable identical control functionality between the physical and virtual systems can be found in Appendix III.



(a) Physical system (b) Digital twin

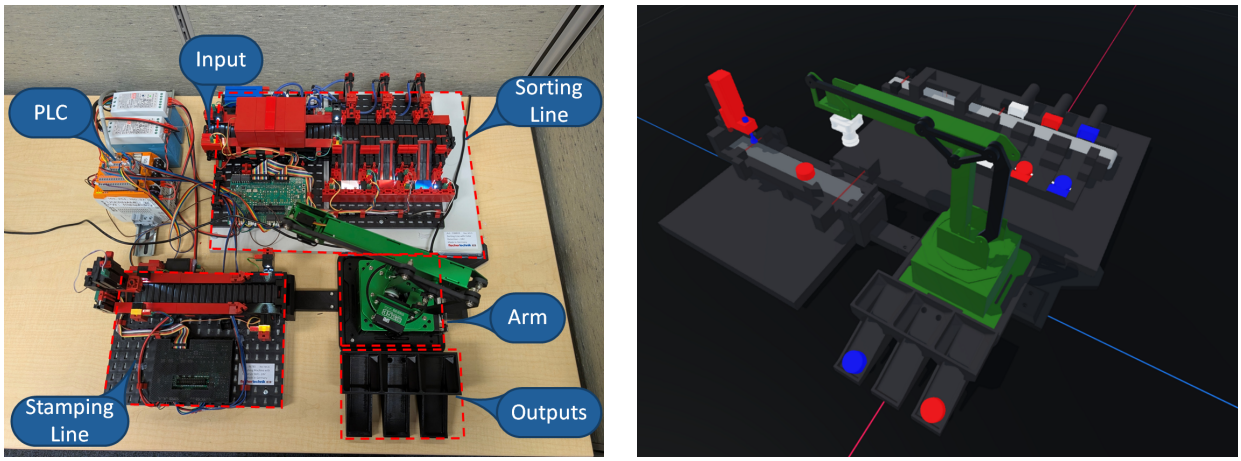
Figure 1: Real and DT stamping lines for the final lab incorporating the stamping line and robotic arm.

4 Course Integration

The key learning objectives of the course were to enable students to design, implement, and validate control logic for the full system. Theoretical and background materials were presented in the lecture and applied in a weekly 1 hour and 15 minute laboratory session, with additional TA office hours available for hands-on lab work. This section presents the representative course labs that illustrate the DT's role in the curriculum: a Hardware Optional Lab (PLC Lab 1), a Hardware Required Lab (PLC Lab 2), and a Final Project.

PLC Lab 1 (Hardware Optional): In this lab, students automated the stamping line module using the DT as a simulation-only environment, without connecting to the physical PLC or hardware. The task required students to program the system to accept round pucks placed at the input end of the conveyor, transport them to the stamping station, actuate the stamp and return it to its neutral position, and convey the puck back to the end of the belt. After manual removal of the puck, the system must reset to its initial state. The simulated station's logic and capabilities were designed to match the physical line exactly. To earn full credit, students submitted their control code along with recorded simulation results demonstrating correct operation. For bonus credit, students could deploy their code to the physical system, controlling a single conveyor module via the PLC.

Lab 2 (Hardware Required): As shown in Fig. 1, this lab expanded the system scope to include a robotic arm. The DT was updated to accurately recreate both components, with custom scripting allowing DT and physical arm to use an identical control interface. The stamping module received an updated model reflecting recent hardware modifications, though its control logic remained unchanged from PLC Lab 1. Students were required to integrate the two modules, coordinating the arm to pick pucks from an input location, place them on the stamping station conveyor, and return them after processing. Students first validated their control code in the DT simulation, then demonstrated successful execution on the physical system to complete the lab. This progression from virtual validation to hardware deployment reinforced the DT's role as a development and debugging tool while ensuring students gained hands-on experience with real equipment.



(a) Physical system (b) Digital twin

Figure 2: Real and DT production lines for the final project incorporating all hardware from the final lab asa well as a color sorting line.

Final Project: The final project required students to integrate all three modules, the stamping station, robotic arm, and a new automated sorting line. Students worked in groups of five to collaboratively develop their control solutions and technical reports. The sorting line introduced analog sensing and multi-path material routing. An analog color sensor measured reflected red light from each puck's surface, with more reflective pucks producing lower voltage readings. A conveyor transported pucks past the color sensor and a beam sensor for indexing, while three pneumatic pistons selectively diverted pucks into corresponding output troughs (white, red, and blue) based on classification. Students developed control logic to coordinate the robotic arm loading pucks onto the sorting line, classify them by color, and route them to the appropriate output. As shown in Fig. 2, a dimensionally and functionally accurate DT model of the sorting line was developed for the project. Custom scripting approximated the behavior of the analog color sensor: the simulated sensor maintained a baseline reading of approximately 7.5 volts when idle and decreased proportionally as a puck passed underneath, returning intermediate values as the puck approached and receded from the sensor mirroring the behavior of the real counterpart. Noise was added to the sensor output to further mimic the real sensor's performance. This fidelity allowed students to develop and tune their classification thresholds in simulation before deploying to the physical system; while the variability and noise were not perfectly recreated (and varied between instances of the physical sensor), students already had the logic in place to deal with them and just needed to calibrate for the specific production line they were using.

4.1 Results and Discussion

During PLC Lab 1, of the 25 students enrolled, 9 attempted the bonus credit by running their code on the physical stamping line. Because the DT accurately replicated the conditions of the real system, all students who attempted the bonus succeeded on their first hardware run without requiring assistance beyond resolving simple connection issues. This outcome validated a key benefit of the DT approach: students could confidently complete and test their programs before accessing hardware, enabling all bonus attempts to be demonstrated using a single shared production

line. Without this pre-validation capability, iterative debugging on the physical system would have created scheduling bottlenecks incompatible with a class of this size.

During PLC Lab 2, when students deployed their code to the physical system, they encountered discrepancies between simulated and real robot behavior. The physical servo-driven robot arm was precise but not accurate, showing high repeatibility but a significant amount of baseline error induced by poor calibration. As this varied between copies of the arm and would frequently drift on a day-to-day basis, this could not be easily corrected for in the DT model. Consequently, every student needed to adjust at least 2 of their programmed positions to correct for these real-world inaccuracies. Despite this additional calibration step, prior iteration in the DT meant that students' programs were largely complete upon upload. Students typically completed their hardware demonstrations within 5–10 minutes of connecting to the system, enabling the class of 25 students to complete these hybrid assignments using a maximum of 2 shared physical systems.

Final project integration went without any major problems, as students transitioned from the simulation environment to the physical production line, performing final system integration, verification, and calibration prior to demonstrating their work to the course instructors. Students were split into groups of 5 and were tasked with developing the automation logic for the simulation and the physical system. The simulated production line enabled teams to automate individual production components and explore multi-station interactions and workflow coordination without requiring continuous access to the physical system. All teams completed their course projects on schedule and successfully conducted final demonstrations using only 2 copies of the physical production system over the course of an afternoon.

5 Conclusion

There is a critical need for manufacturing automation educational curriculum and tools to equip the current and future workforce with the skills required to operate, maintain, and innovate digitally integrated production systems. Creating a detailed, realistic Digital Twin model of the course's lab equipment allowed students to develop their own solutions to realistic challenges without requiring them to use expensive hardware. Due to the accurate simulation and compatible I/O mapping, students were then able to transfer their code to the physical production lines to quickly verify their solutions. As the majority of their work was done virtually, we are able to run the class without purchasing a hardware set per student or requiring them to coordinate shared access to the system while developing their code. The 25 students in this course were able to complete the PLC-based labs and projects in our course without requiring the lab to be open more than 6-8 hours a week by using the DT-based model to perform the bulk of their work.

For future work, DT technology can be further developed to unlock new capabilities, including real-time logging and production metric visualization, and teaching physical design concepts through simulator-based prototyping. Beyond technical extensions, future course offerings will incorporate a formal evaluation framework to assess whether students are meeting defined learning objectives and outcomes. Students will be surveyed before and after the course on their understanding of key automation concepts and technical skills with the hardware and software they will be using, and what benefits and drawbacks they saw with the hybrid use DT compared to a hypothetical hardware-only or simulation-only curriculum.

Acknowledgements

This work was supported in part by a Penn State Leonhard Center Seed Grant and in part by the National Science Foundation under CAREER Award No. 2442362.

References

[1] Baotong Chen, Jiafu Wan, Lei Shu, Peng Li, Mithun Mukherjee, and Boxing Yin. Smart factory of industry 4.0: Key technologies, application case, and challenges. *IEEE Access*, 6:6505–6519, 2017.

[2] Andrea Benešová and Jiří Tupa. Requirements for education and qualification of people in industry 4.0. *Procedia manufacturing*, 11:2195–2202, 2017.

[3] Jesper Puggaard de Oliveira Hansen, Svend Hansson, and Matthias Koenig. Teaching programmable logic controllers: a hands-on and skill-building approach. In *2024 IEEE 7th International Conference on Industrial Cyber-Physical Systems (ICPS)*, pages 1–6. IEEE, 2024.

[4] Shouling He, Hossein Rahemi, and Khalid Mouaouya. Teaching plc programming and industrial automation in mechatronics engineering. In *2015 ASEE Annual Conference & Exposition*, pages 26–1483, 2015.

[5] Steven F Barrett, Amos L Purdy, and Cameron HG Wright. Hands on programmable logic controller (plc) laboratory for an industrial controls course. In *2011 ASEE Annual Conference & Exposition*, pages 22–765, 2011.

[6] Hoton Henriques de Almeida Bastos, Gilberto João Pavani, and Matheus Perin. Low-cost digital twin approach for industrial automation education in the context of industry 4.0. *Advances in Mechanical and Materials Engineering*, 42(1), 2025.

[7] MA Hazrat, NMS Hassan, Ashfaque Ahmed Chowdhury, MG Rasul, and Benjamin A Taylor. Developing a skilled workforce for future industry demand: The potential of digital twin-based teaching and learning practices in engineering education. *Sustainability*, 15(23):16433, 2023.

[8] M Aria, J Utama, F Fauzia, M Rizal, M Fahmi, and M Yudha. Virtual simulation system with various examples and analysis tools for programmable logic controller training. In *IOP Conference Series: Materials Science and Engineering*, volume 879, page 012108. IOP Publishing, 2020.

[9] Kristina Eriksson, Abdlkarim Alsaleh, Shervin Behzad Far, and David Stjern. Applying digital twin technology in higher education: An automation line case study. In *SPS2022*, pages 461–472. IOS Press, 2022.

[10] Michal Balla, Oto Haffner, Erik Kučera, and Ján Cigánek. Educational case studies: creating a digital twin of the production line in TIA Portal, Unity, and Game4Automation framework. *Sensors*, 23(10):4977, 2023.

[11] Carlos A Jara, Francisco A Candelas, Santiago T Puente, and Fernando Torres. Hands-on experiences of undergraduate students in automatics and robotics using a virtual and remote laboratory. *Computers & Education*, 57 (4):2451–2461, 2011.

[12] Yassine Qamsane, James Moyne, Maxwell Toothman, Ilya Kovalenko, Efe C Balta, John Faris, Dawn M Tilbury, and Kira Barton. A methodology to develop and implement digital twin solutions for manufacturing systems. *Ieee Access*, 9:44247–44265, 2021.

[13] PLC-Programming – fischertechnik. `https://www.fischertechnik.de/en/industry-and-universities/plc-programming`. (Accessed: 21 Jan 2026).

Appendix I: PLC Selection Criteria

Hardware selection was driven largely by the availability and usefulness of a simulated copy of the chosen PLC. An ideal candidate would allow us to use code on either the physical or virtual controller, controlling the real system or the DT. We tested several options and selected the Kunbus RevPi platform as our PLC. It was more expensive than our other options, but readily available and configurable to accept multiple IDE/runtime options. Siemens S7-series PLCs were also considered and would have been a more affordable option, but licensing issues would have prevented us from distributing copies of the simulated PLC to students.

Table 1: Score Definitions for PLC Hardware Evaluation

Criterion	Score Meaning (1–3)
Cost	3 = lowest cost; 2 = moderate cost; 1 = highest cost
Availability	3 = readily available online semi-directly; 2 = Not used; 1 = available only through authorized distributor
IDE Compatibility	3 = supports multiple IDEs for programming; 2 = IDE restricted but not vendor-locked; 1 = vendor-specific IDE only
Digital Twin Compatibility	3 = supports both OPC UA and Modbus communication; 2 = supports Modbus only; 1 = no supported DT communication interface
Soft PLC Simulation	3 = built-in or easy-to-use soft PLC simulation; 2 = simulation available but complex or limited; 1 = no simulation support
Hardware Flexibility	3 = wide range of digital and analog I/O options; 2 = limited I/O (e.g., source or relay output only); 1 = minimal or fixed I/O capability
Open Source Hardware	3 = fully open-source hardware ecosystem; 2 = partially open-source; 1 = closed-source hardware
Support	3 = strong vendor support and learning resources; 2 = limited or partial support; 1 = minimal or no support
Robot Interface	3 = easily allows control of a linked robotic arm; 2 = interface available but requires additional effort to communicate; 1 = robot control difficult or impossible
Industry Relevance	3 = PLC is or is similar to a commonly-used industrial controller; 2 = moderate industry relevance; 1 = limited or no industrial relevance

Table 2: PLC Hardware Evaluation

Criteria	Weight	P1AM	Sequent	RevPi	S7-1215C	CX7000
Cost	2	3	3	1	2	2
Availability	2	3	3	3	1	1
IDE Compatibility	2	2	2	3	1	1
Digital Twin Compat.	3	2	2	3	3	3
Soft PLC Simulation	3	2	2	3	3	3
HW Flexibility	2	3	3	2	3	3
Open Source HW	1	2	3	2	1	2
Support	1	2	1	2	2	3
Robot Interface	2	3	3	3	2	2
Industry Relevance	2	2	1	2	3	3
Weighted Score		**48**	**46**	**50**	**45**	**47**

Table 3: Score Definitions for PLC IDE Evaluation

Criterion	Score Meaning (1–3)
Cost	3 = low cost or free; 2 = moderate cost; 1 = high cost
Code Editing	3 = comprehensive and user-friendly editing tools; 2 = adequate editing support; 1 = limited editing functionality
Code Debugging	3 = comprehensive live debugging of simulated and real PLC code; 2 = basic debugging support; 1 = limited or no debugging capability
PLC HW Compatibility	3 = compatible with a wide variety of PLCs; 2 = compatible with a limited number of PLC families; 1 = compatible with a single family and / or vendor-locked
Digital Twin Compatibility	3 = easily supports a simulated PLC for DT use; 2 = supports a DT simulated PLC but requires additional work to use; 1 = no DT PLC support
Setup / License Complexity	3 = simple installation and licensing; 2 = additional setup required when installing, may require license; 1 = server-managed or device-locked licensing
Open Source	3 = fully open-source; 2 = partially open-source; 1 = closed-source
Training / Certifications	3 = widely available training and certifications; 2 = limited training resources; 1 = no formal training or certification
Industry Relevance	3 = widely adopted in industry; 2 = moderately adopted; 1 = limited industry use

Table 4: PLC IDE Evaluation

Criteria	Weight	CODESYS	OpenPLC
Cost	2	3	3
Code Editing	3	2	3
Code Debugging	2	3	3
PLC HW Compatibility	2	3	2
Digital Twin Compat.	3	3	2
Setup / License Complexity	2	2	3
Open Source SW	1	1	3
Training / Certifications	1	2	1
Industry Relevant	2	3	1
Weighted Score		**46**	**43**

Appendix II: DT Software Selection Criteria

When selecting the software environment to build our DT in, we evaluated several software options to find one that could accurately model our full production system with realistic physics and interactions between components while still being simple enough for students to install and use on their own devices. It must also be able to closely match the I/O configuration of the real system, allowing code to be interchangeable between the real system with a physical PLC and the all-virtual Digital Twin. Visual Components scored highest on our table using our original criteria, but due to greater ease of use, significantly lower price (free for education), and compatibility issues encountered when simulating small-scale hardware in Visual Components we selected ProtoTwin as our platform for this fall's course.

Table 5: Score Definitions for Digital Twin Software Evaluation

Criterion	Score Meaning (1–3)
Cost	3 = low cost or free; 2 = moderate cost; 1 = high cost
Setup / License Complexity	3 = simple installation and licensing; 2 = moderate configuration effort; 1 = complex setup or restrictive licensing
Ease of Use	3 = intuitive workflows for configuration and use; 2 = moderate learning curve; 1 = difficult to use
Custom CAD Models	3 = simple workflow to import and configure custom CAD models; 2 = limited custom model capability; 1 = no custom CAD support
Default Equipment Library	3 = comprehensive built-in equipment library; 2 = moderate library coverage; 1 = minimal or no default equipment library
Features	3 = rich feature set simulating highly capable systems; 2 = adequate feature set; 1 = limited functionality
Physics Simulation	3 = high-fidelity physics-based simulation; 2 = simplified or approximate physics; 1 = no physics-based simulation
Vendor Support	3 = strong vendor support and documentation; 2 = moderate support resources; 1 = limited or no support
Training / Certifications	3 = formal training programs and certifications available; 2 = informal or limited training resources; 1 = no training or certification offerings
Industry Relevance	3 = widely adopted in industrial applications; 2 = moderate industry adoption; 1 = limited industry use

Table 6: Digital Twin Software Evaluation

Criteria	Weight	ProtoTwin	Visual Comp.	Factory IO	Siemens PS
Cost	2	3	1	2	2
Setup / License Complexity	2	3	2	3	1
Ease of Use	2	2	2	3	1
Custom CAD Models	3	3	3	1	2
Default Equip. Library	1	1	3	2	2
Features	2	2	3	2	3
Physics Simulation	1	3	2	3	1
Vendor Support	2	2	3	2	2
Training / Certifications	1	1	3	1	3
Industry Relevant	2	1	3	1	3
Weighted Score		**40**	**45**	**35**	**36**

Appendix III: Virtual Model and Custom Control Schemes Details

Accurate CAD models of the FT stamping line and our selected arm (the HIWONDER MaxArm servo-actuated platform) were created ahead of the course and imported into ProtoTwin. Using built-in tools, we defined colliders and materials for all objects as well as the joint configurations needed to move the press up and down and move all relevant axes of the arm. Several components in the stamp and sort modules required custom scripts to be created in ProtoTwin's built-in Typescript editor, allowing components to match the functionality of their real counterparts. By default, motorized objects in ProtoTwin operate on fully analog control, taking target speed and/or position as inputs and returning their current position. The FT modules take boolean inputs to their motion elements (one pin moves the belt forwards, another backwards, if both are true or both false it does not move), which is excellent for instructional purposes due to its simplicity. A custom I/O component was programmed and associated with each relevant object in ProtoTwin to accept these inputs from the simulated PLC and translate them into the correct analog signals for the motor. For moving axes that use limit switches on both ends of their travel, the custom component also simulates the boolean values of each switch using the known analog motor position and returns those the PLC. Photoelectric beam sensors detect plastic pucks at either end of the belt; in the DT these were simulated using cylindrical "beam" elements which detected if any pucks intersected them.

A custom control scheme was devised for both the simulated and physical robotic arms. We desired to keep the control system single-source, so students would not have to program the arm in a separate environment. The analog position in 4 axes (X, Y, Z, and rotation) as well as the state of the suction cup gripper and an overall motion enable variable were all represented by prenamed configured variables accessible by students' code. These variables were exposed to the network via the OPC-UA network protocol. This protocol hosts a local server that other programs and devices can connect to and read or write any configured variables. For the DT, a custom Typescript controller was created, which monitored these values and fed them to the simulated arm. The simulated arm was controlled using ProtoTwin's built-in inverse kinematics solver. This script checked for any changes to the pose and, if enabled, planned and executed a linear move using the arm timed to take exactly 1500 milliseconds (in sim time). For the real arm, an external control program was created to interface between the OPC-UA server and the arm's serial communications port. A custom control program was loaded onto the arm, which monitored its serial port for incoming instructions and used the arm's built-in inverse kinematics solver to move it to the specified coordinates. A custom Python script was created to read hosted OPC-UA variables and automatically command the arm to move to the matching position. It was packaged as a Windows .exe file, which can be run on any computer connected to the PLC (via Ethernet) and the arm (via USB). This software served a similar purpose to the custom DT controller, monitoring the OPC variables for any change and generating G-code on the fly to send to the arm. Several students had a non-standard serial port configuration on their devices, which required the control software to be modified, as its original port scan method would identify the wrong peripheral when these students attempted to run it.

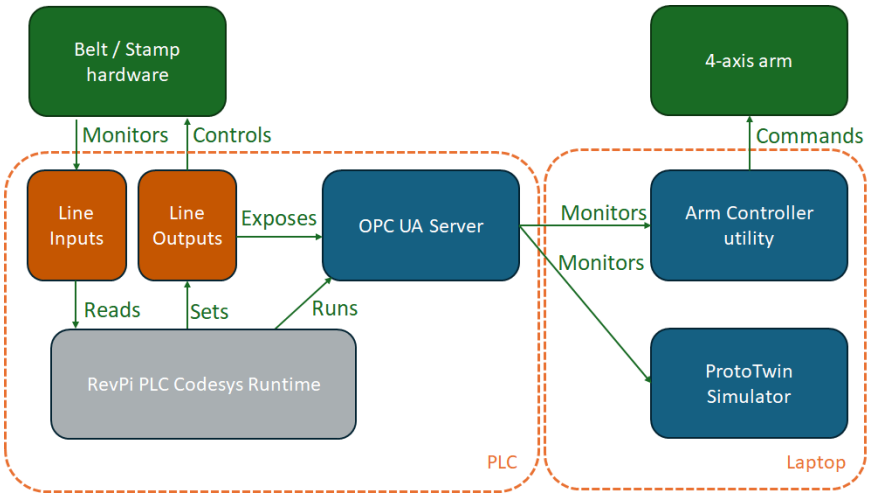


Figure 3: Manufacturing system network when configured for synchronous use.