

การใช้งานแพลตฟอร์มพัฒนาหุ่นยนต์ Kria KR260 ร่วมกับไมโครคอนโทรลเลอร์และอุปกรณ์รอบข้าง Implementation of Kria KR260 Robotics Development Platform Integrated with Microcontroller and Peripheral Devices

เศรษฐกาล โปร่งนุช^{1*}, วิวรรณรัตน์ สารมัตย์¹, อภิรักษ์ ธิตินามิต¹

Sethakarn Prongnuch^{1*}, Wiwanrat Saramat¹, Aphirak Thitinaruemit¹

¹สาขาวิชาวิศวกรรมหุ่นยนต์ คณะวิศวกรรมศาสตร์และเทคโนโลยีอุตสาหกรรม
มหาวิทยาลัยราชภัฏสวนสุนันทา

¹ Department of Robotics Engineering, Faculty of Engineering and Industrial Technology,
Suan Sunandha Rajabhat University

*Corresponding author email: sethakarn.pr@ssru.ac.th

Received 05 Oct. 2025 Revised 04 Mar. 2026 Accepted 23 Apr. 2026

บทคัดย่อ

การควบคุมหุ่นยนต์ในภาคอุตสาหกรรมหรือภาคบริการจำเป็นต้องพัฒนาแอปพลิเคชันควบคุมเฉพาะบนบอร์ดหรือแพลตฟอร์มที่มีรูปแบบแตกต่างกัน ซึ่งต้องอาศัยวิศวกรที่มีความเชี่ยวชาญเฉพาะด้าน ส่งผลให้ต้นทุนการพัฒนาและบำรุงรักษาระบบสูงขึ้น งานวิจัยนี้เสนอการใช้งานแพลตฟอร์มพัฒนาหุ่นยนต์ Kria KR260 ซึ่งเป็นแพลตฟอร์มที่รองรับการพัฒนาซอฟต์แวร์แบบโอเพนซอร์ส โดยเชื่อมต่อและทำงานร่วมกับไมโครคอนโทรลเลอร์ Arduino และอุปกรณ์รอบข้าง มีวัตถุประสงค์เพื่อศึกษาการออกแบบการพัฒนา และการประเมินประสิทธิภาพของแพลตฟอร์มแบบบูรณาการร่วมกับไมโครคอนโทรลเลอร์และอุปกรณ์รอบข้าง การพัฒนาใช้ซอฟต์แวร์ Vivado ML Edition สำหรับการออกแบบฮาร์ดแวร์ และใช้ภาษาระดับสูงในการพัฒนาแอปพลิเคชันควบคุม ดำเนินการทดสอบการสื่อสารข้อมูล และการควบคุมฮาร์ดแวร์ระหว่างไมโครคอนโทรลเลอร์กับอุปกรณ์รอบข้างผ่านพอร์ต Pmod ผลการทดสอบพบว่าประสิทธิภาพของการสื่อสารข้อมูลแบบ I²C มีเวลาตอบสนองที่รวดเร็ว

กว่าแบบ GPIO โดยค่าเวลารับข้อมูลของ GPIO อยู่ระหว่าง 17–61 มิลลิวินาที ขณะที่ I²C อยู่ที่ 1.13–1.36 มิลลิวินาที ส่วนเวลาส่งข้อมูลของ GPIO จะคงที่ที่ 4 มิลลิวินาที ในขณะที่ I²C อยู่ระหว่าง 0.41–0.54 มิลลิวินาที ผลการทดสอบแสดงให้เห็นว่า I²C มีประสิทธิภาพสูงกว่าในด้านความเร็ว และมีความสม่ำเสมอของเวลาตอบสนองมากกว่า โดยพิจารณาจากช่วงค่าต่ำที่สุดถึงมากที่สุด รวมถึงมีอัตราความสำเร็จของการรับส่งข้อมูลครบถ้วนภายใต้เงื่อนไขการทดลอง ทั้งนี้ผลการประเมินยืนยันว่าแพลตฟอร์ม Kria KR260 สามารถทำงานร่วมกับไมโครคอนโทรลเลอร์และอุปกรณ์รอบข้างได้อย่างมีประสิทธิภาพ และทำงานได้ถูกต้องภายใต้เงื่อนไขการทดลอง เหมาะสำหรับการประยุกต์ใช้ในการพัฒนาหุ่นยนต์ในภาคอุตสาหกรรมหรือภาคบริการ

คำสำคัญ: แพลตฟอร์มพัฒนาหุ่นยนต์ Kria KR260, ไมโครคอนโทรลเลอร์, Arduino, อุปกรณ์รอบข้าง, หุ่นยนต์

Abstract

Controlling robots in industrial or service sectors requires the development of specialized control applications on boards or platforms with varying design features. This, in turn, demands skilled engineers, leading to higher system development and maintenance costs. This research proposes the use of the Kria KR260 robot development platform, which supports open-source software development and can interface with an Arduino microcontroller and peripheral devices. The objective is to study the design, development, and performance evaluation of an integrated platform incorporating microcontrollers and peripheral devices. The development process employs Vivado ML Edition software for hardware design and a high-level programming language for developing control applications. Data communication and hardware control tests are performed between the microcontroller and peripheral devices via the Pmod port. The test results show that I²C communication has faster response times than GPIO. The GPIO receive times range from 17 to 61 milliseconds, while the I²C receive times range from 1.13 to 1.36 milliseconds. The GPIO transmit time remains constant at 4 milliseconds, whereas the I²C transmit time ranges from 0.41 to 0.54 milliseconds. The experimental results indicate that I²C delivers superior performance in terms of communication speed and exhibits more consistent response times, as evidenced by the observed minimum to maximum range. In addition, I²C achieved a complete data transmission success rate under the specified experimental conditions. Overall, the evaluation confirms that the Kria KR260 platform can interface efficiently with the microcontroller and peripheral devices and operated correctly throughout the experiments. These findings suggest that the platform is well suited for applications in the development of robots for industrial and service domains.

Keywords: Kria KR260 Robotics Development Platform, Microcontroller, Arduino, Peripheral Devices, Robot

1. บทนำ

ในยุคปัจจุบัน เทคโนโลยีหุ่นยนต์ได้เข้ามามีบทบาทสำคัญอย่างยิ่งในชีวิตประจำวันและภาคอุตสาหกรรมต่าง ๆ โดยมีการประยุกต์ใช้หุ่นยนต์ในหลากหลายด้าน ตั้งแต่หุ่นยนต์อุตสาหกรรมที่ใช้ในกระบวนการผลิตอัตโนมัติ ไปจนถึงหุ่นยนต์บริการที่ช่วยอำนวยความสะดวกภายในบ้าน ซึ่งการควบคุมหุ่นยนต์นั้น วิศวกรต้องเขียนแอปพลิเคชันสำหรับควบคุมการทำงานบนบอร์ดควบคุมหรือแพลตฟอร์มควบคุมหุ่นยนต์ที่มีรูปแบบที่แตกต่างกัน ซึ่งต้องอาศัยการจ้างงานวิศวกรที่มีความรู้และความเชี่ยวชาญเฉพาะด้าน ดังนั้นการเลือกใช้บอร์ดควบคุมหรือแพลตฟอร์มควบคุมหุ่นยนต์ที่เหมาะสมจึงเป็นแนวทางเริ่มต้นในการแก้ไขปัญหาดังกล่าว

แพลตฟอร์มควบคุมหุ่นยนต์สมัยใหม่มีลักษณะเป็นบอร์ดคอมพิวเตอร์แบบเดี่ยว (Single Board Computer) (Prongnuch & Sitjongsataporn, 2021) ซึ่งมีความสะดวกในการติดตั้งภายในตัวหุ่นยนต์เพื่อเชื่อมต่อกับไมโครคอนโทรลเลอร์ (Microcontroller) หรืออุปกรณ์รอบข้างของหุ่นยนต์ ยกตัวอย่างเช่น แพลตฟอร์มพัฒนาหุ่นยนต์ Kria KR260 เป็นแพลตฟอร์มควบคุมหุ่นยนต์ที่มีประสิทธิภาพสูงเนื่องจากมีระบบบนชิป (System-on-Chip) อยู่ภายในที่ประกอบด้วยตัวประมวลผล Arm และ FPGA (Field-Programmable Gate Array) มีความยืดหยุ่นด้วยรูปแบบวงจรระดับบอร์ดที่รวมฟังก์ชันระบบไว้ในโมดูลเดียว (System-on-Module) รองรับการทำงานร่วมกับอุปกรณ์รอบข้างได้อย่างหลากหลาย (อภิรักษ์ อิตินฤมิต, เศรษฐกุล โปร่งนุช และ ปฏิญญาณ์ แสงอรุณ, 2565) แพลตฟอร์มนี้นิยมนำไปใช้ในการประมวลผลข้อมูลจำนวนมากและซับซ้อนด้วยอัลกอริธึมที่หาค่าเหมาะสมที่สุด (Optimization Algorithm) สามารถตอบสนองต่อสภาพแวดล้อมแบบเวลาจริง (Real-time) (Prongnuch & Sitjongsataporn, 2024; Thitinaruemit, A., Sitjongsataporn & Prongnuch, 2023) เมื่อนำแพลตฟอร์มพัฒนาหุ่นยนต์ Kria KR260 ไปเชื่อมต่อกับอุปกรณ์รอบข้างที่นิยมใช้งานภายในตัวหุ่นยนต์ อาทิเช่น ไมโครคอนโทรลเลอร์ Arduino ซึ่งเป็นไมโครคอนโทรลเลอร์ที่ใช้งานในระบบอัจฉริยะ (วรสุภรางค์ ศิริพราหมณกุล และคณะ, 2564) จะช่วยให้หุ่นยนต์มีประสิทธิภาพและมีความเสถียรเพิ่มมากขึ้น

ดังนั้น บทความนี้นำเสนอ การใช้งานแพลตฟอร์มพัฒนาหุ่นยนต์ Kria KR260 ร่วมกับไมโครคอนโทรลเลอร์ Arduino และอุปกรณ์รอบข้างผ่านพอร์ต Pmod (Peripheral Module) โดยอาศัยสัญญาณควบคุมแบบพื้นฐานแบบ GPIO (General Purpose Input/Output) และโพรโทคอล (Protocol) การสื่อสารแบบอนุกรมแบบ I²C (Inter-

Integrated Circuit) ประกอบด้วย ตัวรับรู้ (Sensor) แบบอินฟราเรดหรือไออาร์ (Infrared: IR) เซอร์โวมอเตอร์ (Servo Motor) และมอเตอร์ไฟฟ้ากระแสตรง (DC Motor) มีเป้าหมายเพื่อศึกษาการใช้งาน ออกแบบและทดสอบด้วยซอฟต์แวร์ Vivado ML Edition และการประเมินประสิทธิภาพของแพลตฟอร์มดังกล่าวจากแอปพลิเคชันควบคุมที่พัฒนาขึ้นด้วยภาษาระดับสูง ได้แก่ Shell Script ภาษา C/C++ และภาษา Python เพื่อประเมินความสามารถในการทำงานร่วมกันเป็นระบบอย่างมีประสิทธิภาพ

2. วัตถุประสงค์

2.1 ศึกษาการใช้งานแพลตฟอร์มพัฒนาหุ่นยนต์ Kria KR260

2.2 ออกแบบและทดสอบแพลตฟอร์มพัฒนาหุ่นยนต์ Kria KR260 ร่วมกับไมโครคอนโทรลเลอร์และอุปกรณ์รอบข้าง

2.3 ประเมินประสิทธิภาพของแพลตฟอร์มพัฒนาหุ่นยนต์ Kria KR260 ร่วมกับไมโครคอนโทรลเลอร์และอุปกรณ์รอบข้างจากแอปพลิเคชันที่ได้พัฒนาขึ้น

2.1 ขอบเขตของการวิจัย

งานวิจัยนี้ มุ่งเน้นการบูรณาการแพลตฟอร์ม Kria KR260 กับไมโครคอนโทรลเลอร์ Arduino และอุปกรณ์รอบข้างผ่านพอร์ต Pmod

2.1.1 พิจารณา 2 รูปแบบการสื่อสาร ได้แก่ แบบ GPIO ตรวจสอบความถูกต้องด้วยระบบไฟล์ Sysfs และแบบ I²C จากการพัฒนามอดูล AXI IIC สำหรับควบคุมและจัดการการ

2.1.2 อุปกรณ์รอบข้างที่ใช้ทดสอบประกอบด้วย แอลอีดีและปุ่มกดแบบ GPIO ตัวรับรู้แบบอินฟราเรด เซอร์โวมอเตอร์ และมอเตอร์ไฟฟ้ากระแสตรงแบบ I²C ผ่านไมโครคอนโทรลเลอร์ Arduino

2.1.3 มีการประเมินผล ครอบคลุมความถูกต้องในการทำงาน อัตราความสำเร็จของการรับส่งข้อมูล และเวลาตอบสนอง ภายใต้เงื่อนไขการทดลองที่กำหนด

งานวิจัยเลือกใช้การเชื่อมต่อผ่าน Pmod จึงไม่ครอบคลุมการทดสอบภายใต้สภาพแวดล้อมอุตสาหกรรมจริงหรือการสื่อสารระยะไกล

3. ทฤษฎีที่เกี่ยวข้อง

การศึกษาทฤษฎีที่เกี่ยวข้องแบ่งออกเป็น 3 เรื่อง ได้แก่ 1) แพลตฟอรม์พัฒนาหุ่นยนต์ Kria KR260 2) ไมโครคอนโทรลเลอร์ Arduino และ 3) อุปกรณ์รอบข้าง ดังนี้

3.1 แพลตฟอรม์พัฒนาหุ่นยนต์ Kria KR260

แพลตฟอร์มพัฒนาหุ่นยนต์ Kria KR260 (Kria KR260 Robotics Starter Kit) เป็นระบบพัฒนาแบบสมบูรณ์สำหรับงานหุ่นยนต์ โดยใช้ชิป Zynq UltraScale+ MPSoC ซึ่งผสานรวมหน่วยประมวลผลและลอจิกที่สามารถโปรแกรมได้ (Programmable Logic: PL) ภายในชิปเดียว ทำให้สามารถประมวลผลทั้งงานทั่วไปและงานแบบเวลาจริงได้อย่างมีประสิทธิภาพ (Advanced Micro Devices, Inc., 2025) หัวข้อนี้นำเสนอสถาปัตยกรรมของแพลตฟอร์มพัฒนาหุ่นยนต์ Kria KR260 ได้แก่ ชิป Zynq UltraScale+ MPSoC พอร์ต (Port) การเชื่อมต่อ และซอฟต์แวร์ Vivado ML Edition ดังนี้

3.1.1 ชิป Zynq UltraScale+ MPSoC

ชิป Zynq UltraScale+ MPSoC ประกอบด้วย 2 ส่วน ได้แก่ ส่วนที่ 1 คือ ระบบประมวลผล (Processing System: PS) ประกอบด้วย Arm Cortex-A53 สำหรับงานทั่วไป และระบบปฏิบัติการ Linux และ Arm Cortex-R5 สำหรับงานแบบเวลาจริง เช่น การควบคุมมอเตอร์และการอ่านค่าจากตัวรับรู้ และส่วนที่ 2 คือ ลอจิกที่สามารถโปรแกรมได้ใช้เทคโนโลยี FPGA เพื่อสร้างวงจรเฉพาะสำหรับเร่งการประมวลผล เช่น การประมวลผลภาพ การสื่อสารความเร็วสูง และการควบคุมอุปกรณ์เฉพาะทาง (อภิรักษ์ ธิติณมิต และคณะ, 2565) ดังนั้นการรวมสถาปัตยกรรม PS และ PL เข้าด้วยกันทำให้แพลตฟอร์มพัฒนาหุ่นยนต์ Kria KR260 มีทั้งความยืดหยุ่นของซอฟต์แวร์และความเร็วของฮาร์ดแวร์ในระบบเดียว เรียกว่าสถาปัตยกรรมที่มีตัวประมวลผลหลายชนิด (Prongnuch et al., 2020)

3.1.2 พอร์ตการเชื่อมต่อ

แพลตฟอร์มพัฒนาหุ่นยนต์ Kria KR260 มีพอร์ตการเชื่อมต่อมาตรฐานหลากหลาย เช่น พอร์ต Gigabit Ethernet พอร์ต USB พอร์ต Display และพอร์ต Pmod เพื่อรองรับการเชื่อมต่อกับอุปกรณ์รอบข้างต่าง ๆ ได้อย่างสะดวก (Aubonnoi, & Prongnuch, 2025) โดยเฉพาะพอร์ต Pmod ที่เหมาะสำหรับการสื่อสารระหว่างไมโครคอนโทรลเลอร์ และอุปกรณ์ตรวจจับในงานวิจัยหุ่นยนต์

3.1.3 ซอฟต์แวร์ Vivado ML Edition

เป็นซอฟต์แวร์ที่ใช้สำหรับการออกแบบฮาร์ดแวร์บนแพลตฟอร์มพัฒนาหุ่นยนต์ Kria KR260 จากบริษัท Advanced Micro Devices, Inc. ใช้สำหรับการพัฒนาระบบดิจิทัลบนชิป FPGA ผู้พัฒนาสามารถสร้างบล็อกการทำงานเฉพาะทาง ปรับแต่งสัญญาณ และจำลองการทำงานได้อย่างมีประสิทธิภาพ (Advanced Micro Devices, Inc., 2025) ทั้งนี้ซอฟต์แวร์ Vivado ML Edition เมื่อใช้งานกับแพลตฟอร์มพัฒนาหุ่นยนต์ Kria KR260 จะรองรับการพัฒนาแอปพลิเคชันด้วยภาษา C/C++ และ Python สำหรับงานควบคุมหรือการประมวลผลระดับสูง

3.2 ไมโครคอนโทรลเลอร์ Arduino

ไมโครคอนโทรลเลอร์ เป็นวงจรรวม (Integrated Circuit) ที่รวมหน่วยประมวลผลกลาง (Central Processing Unit) หน่วยความจำ (Memory) และพอร์ตอินพุต/เอาต์พุต (I/O Ports) ไว้ภายในชิปเดียว เพื่อใช้ในการควบคุมอุปกรณ์อิเล็กทรอนิกส์หรือระบบอัตโนมัติขนาดเล็ก ไมโครคอนโทรลเลอร์มีคุณสมบัติเด่นคือ ขนาดเล็ก ต้นทุนต่ำ ใช้พลังงานน้อย และสามารถโปรแกรมได้อย่างยืดหยุ่น จึงถูกนำมาใช้อย่างแพร่หลายในงานควบคุมเครื่องจักร หุ่นยนต์ ระบบฝังตัว (Embedded System) และระบบอินเทอร์เน็ตประสาทรพลสิ่ง (Internet of Things) ไมโครคอนโทรลเลอร์ที่นิยมนำมาใช้งาน เช่น ไมโครคอนโทรลเลอร์ Arduino ในงานวิจัยการออกแบบเครื่องทำความสะอาดอัจฉริยะ (วรสุภรางค์ ศิริพราหมณกุล และคณะ, 2564) และงานวิจัยการศึกษาเซ็นเซอร์ตรวจวัดอุณหภูมิร่างกายแบบไม่สัมผัสสำหรับคัดกรองผู้ป่วยเบื้องต้น (นทวิร์ ไชยจำ และคณะ, 2567)

3.3 อุปกรณ์รอบข้าง

อุปกรณ์รอบข้างที่นิยมใช้งานสำหรับหุ่นยนต์ แบ่งออกเป็น 2 ประเภท ได้แก่

3.3.1 ตัวรับรู้

ตัวรับรู้ คือ อุปกรณ์ทางอิเล็กทรอนิกส์ที่ทำหน้าที่ตรวจจับและวัดค่าทางกายภาพต่าง ๆ แล้วแปลงสัญญาณทางกายภาพให้อยู่ในรูปแบบของสัญญาณไฟฟ้าที่สามารถส่งต่อให้ไมโครคอนโทรลเลอร์นำไปใช้ได้ ตัวรับรู้จึงเป็นส่วนสำคัญของระบบหุ่นยนต์ เพราะทำหน้าที่เชื่อมโยงระหว่างโลกจริงกับระบบดิจิทัล เพื่อให้หุ่นยนต์สามารถรับรู้สภาพแวดล้อมและตอบสนองได้อย่างเหมาะสม (Li et al., 2023) ตัวรับรู้ แบ่งได้เป็น 2 ประเภท ได้แก่ ตัวรับรู้แบบแอคทีฟ (Active Sensor) ซึ่งต้องใช้แหล่งพลังงานภายนอกในการทำงาน อาทิเช่น ตัวรับรู้แบบอินฟราเรดที่ส่งสัญญาณออกไปเพื่อตรวจจับวัตถุ และตัวรับรู้แบบพาสซีฟ (Passive Sensor) ซึ่งตรวจจับสัญญาณจากสิ่งแวดล้อมโดยไม่ต้องใช้พลังงานภายนอก

3.3.2 แอคชูเอเตอร์

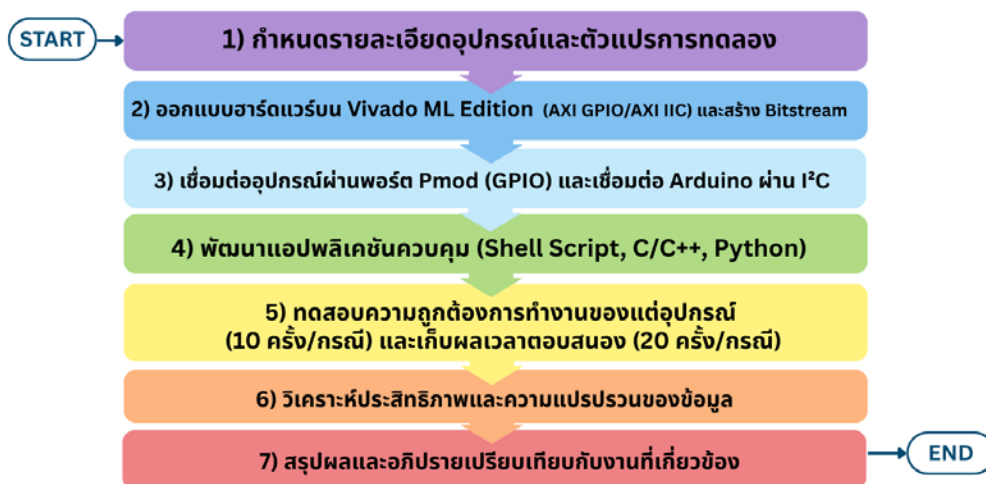
แอคชูเอเตอร์ (Actuator) ได้แก่ แอลอีดี (LED) ปุ่มกด (Push Button) เซอร์โวมอเตอร์ และมอเตอร์ไฟฟ้ากระแสตรง ซึ่งเป็นอุปกรณ์ที่พบได้ในงานพัฒนาหุ่นยนต์

4. วิธีดำเนินการวิจัย

วิธีดำเนินการวิจัยเรื่อง การใช้งานแพลตฟอร์มพัฒนาหุ่นยนต์ Kria KR260 ร่วมกับ ไมโครคอนโทรลเลอร์และอุปกรณ์รอบข้างแบ่งออกเป็น 2 หัวข้อ ประกอบด้วย การออกแบบ และวิธีการทดสอบ และการวิเคราะห์ข้อมูลและเกณฑ์การประเมิน ดังนี้

4.1 การออกแบบและการทดสอบ

ขั้นตอนการออกแบบระบบ และการทดสอบการทำงานของ Kria KR260 ร่วมกับ ไมโครคอนโทรลเลอร์ Arduino หรือบอร์ด Arduino และอุปกรณ์รอบข้าง รวมถึงตัวแปรการทดลอง ดังภาพที่ 1 ดำเนินการออกแบบและสร้างฮาร์ดแวร์ด้วยซอฟต์แวร์ Vivado ML Edition 2023.1 โดยมีอุปกรณ์รอบข้าง ได้แก่ ตัวรับและแอคชูเอเตอร์ที่เชื่อมต่อผ่านพอร์ต Pmod โดยแบบ GPIO และแบบ I²C รวมถึงพัฒนาแอปพลิเคชันสำหรับการควบคุม โดยเฉพาะ มีเกณฑ์การประเมิน และวิธีการวิเคราะห์ข้อมูล

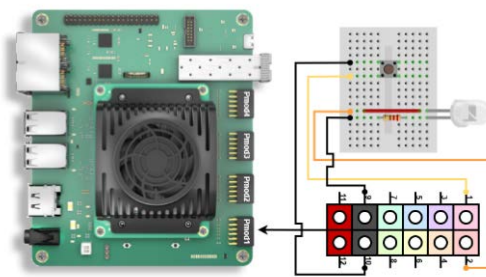


ภาพที่ 1 ขั้นตอนการทดลองและการประเมินผล

การประเมินจากการทดสอบเหล่านี้มีบทบาทสำคัญในการตรวจสอบความสามารถของระบบ ในการทำงานภายใต้สภาวะแวดล้อมจริง โดยประเมินทั้งด้านประสิทธิภาพและการ

4.1.2 การเชื่อมต่ออุปกรณ์ผ่านพอร์ต Pmod

การออกแบบการเชื่อมต่ออุปกรณ์รอบข้างผ่านพอร์ต Pmod ในบอร์ด Kria KR260 ออกแบบบน Vivado ML Edition โดยพัฒนามอดูล AXI GPIO เพิ่มสำหรับควบคุม และตรวจสอบสถานะของพอร์ต GPIO เพื่อเชื่อมต่อกับอุปกรณ์รอบข้าง โดยกำหนดค่าให้รองรับข้อมูลขนาด 8 บิต มีความถี่ 100 MHz อุปกรณ์รอบข้างที่เชื่อมต่อผ่านพอร์ต Pmod เลือกใช้ แอคชูเอเตอร์ ได้แก่ แอลอีดี และปุ่มกด เพื่อทดสอบการทำงานของพอร์ต GPIO โดยเชื่อมต่อ ขาแอโนด (Anode) ของแอลอีดีเข้ากับพอร์ต Pmod1 ช่องที่ 2 และขาแคโทด (Cathode) ผ่านตัวต้านทาน (Resistor) ไปยังพอร์ต Pmod1 ช่องที่ 10 ซึ่งเป็นกราวด์ (Ground) ส่วน ปุ่มกดเชื่อมต่อขาที่ 1 กับพอร์ต Pmod1 ช่องที่ 1 และขาที่ 2 กับ Pmod1 ช่องที่ 9 ซึ่งเป็นกราวด์เช่นกัน ดังภาพที่ 3



ภาพที่ 3 การเชื่อมต่อพอร์ต Pmod ด้วยแอลอีดีและปุ่มกด

วิธีการทดสอบความถูกต้องในการรับส่งข้อมูลผ่านพอร์ต Pmod ใช้ระบบไฟล์ Sysfs เพื่อตรวจสอบความถูกต้องของการเชื่อมต่อและการทำงานของพอร์ต GPIO โดยมีการกำหนดค่าให้รองรับทั้งการอ่านและส่งสัญญาณจากอุปกรณ์รอบข้างแสดงดังภาพที่ 4

```

root@kr260 $ echo 318 > /sys/class/gpio/export
root@kr260 $ echo in > /sys/class/gpio/gpio318/direction
root@kr260 $ echo 1 > /sys/class/gpio/gpio318/direction # เปิด
root@kr260 $ echo 0 > /sys/class/gpio/gpio318/direction # ปิด

```

```

root@kr260 $ echo 319 > /sys/class/gpio/export
root@kr260 $ echo out > /sys/class/gpio/gpio319/direction
root@kr260 $ cat /sys/class/gpio/gpio319/direction
1 # กดปุ่มครั้งแรก
root@kr260 $ cat /sys/class/gpio/gpio319/direction
0 # กดปุ่มอีกครั้ง

```

การทดสอบอินพุตของพอร์ต Pmod
ด้วยระบบไฟล์ Sysfs

การทดสอบเอาต์พุตของพอร์ต Pmod
ด้วยระบบไฟล์ Sysfs

ภาพที่ 4 การทดสอบอินพุตและเอาต์พุตของพอร์ต Pmod ด้วยระบบไฟล์ Sysfs

วิธีการทดสอบการเชื่อมต่ออุปกรณ์ผ่านพอร์ต Pmod เริ่มจากการเขียน Shell Script ซึ่งเป็นแอปพลิเคชันสำหรับควบคุมการทำงานของแอลอีดีและปุ่มกด โดยกำหนดสถานะดังนี้ สถานะเริ่มต้นแอลอีดีจะอยู่ในสถานะปิด เมื่อกดปุ่มครั้งแรกแอลอีดีจะติดหรือสถานะเปิด และจะดับลงเมื่อกดปุ่มซ้ำอีกครั้ง ตามลำดับ ดังภาพที่ 5

```
BEGIN
กำหนดหมายเลขของปุ่มกด
กำหนดหมายเลขของไฟ LED
ตั้งค่าไฟ LED ปิดอยู่

เริ่มทำงานในลูปที่ไม่มีที่สิ้นสุด
อ่านค่าจากปุ่มกด

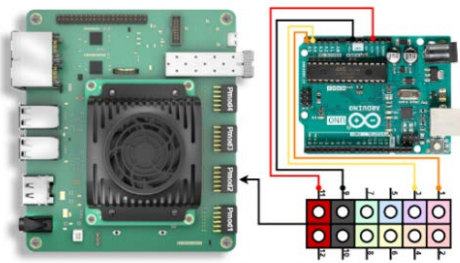
ถ้าปุ่มถูกกด (ค่าที่อ่านได้เป็น 1) THEN
    ถ้าไฟ LED ปิดอยู่ (led_state เป็น 0) THEN
        เปิดไฟ LED
        ตั้งค่า led_state เป็น 1
    ELSE
        ปิดไฟ LED
        ตั้งค่า led_state เป็น 0
    ENDIF
    รอจนกว่าปุ่มจะถูกปล่อย (ค่าที่อ่านได้เป็น 0)
ENDIF

หน่วงเวลาเล็กน้อยเพื่อป้องกันการทำงานที่เร็วเกินไป
ENDWHILE
END
```

ภาพที่ 5 Shell Script สำหรับควบคุมการทำงานของแอลอีดีและปุ่มกด

4.1.3 การเชื่อมต่อไมโครคอนโทรลเลอร์ Arduino ผ่านพอร์ต Pmod

การออกแบบการเชื่อมต่อบอร์ด Arduino กับพอร์ต Pmod ของบอร์ด Kria KR260 อาศัยโปรโตคอล I²C บนซอฟต์แวร์ Vivado ML Edition โดยพัฒนามอดูล AXI IIC เพิ่มสำหรับควบคุมและจัดการการสื่อสารระหว่างอุปกรณ์ กำหนดความถี่ 100 MHz การเชื่อมต่อบอร์ด Arduino ผ่านทางพอร์ต Pmod2 จะเชื่อมต่อขาแอนะล็อก (Analog) เอาต์พุต A5 เข้ากับ SDA ขาที่ 1 และขา A4 เข้ากับ SCL ขาที่ 3 ของบอร์ด Kria KR260 พร้อมทั้งเชื่อมต่อแหล่งจ่ายไฟฟ้า 3.3 โวลต์ (Voltage) และกราวนด์ของ Arduino เข้ากับบอร์ด Kria KR260 ตามลำดับ ดังแสดงในภาพที่ 6



ภาพที่ 6 การเชื่อมต่อไมโครคอนโทรลเลอร์ผ่านพอร์ต Pmod2

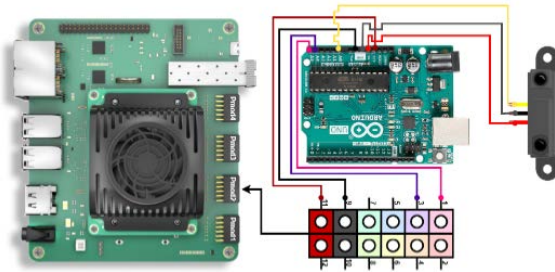
วิธีการทดสอบความถูกต้องในการรับส่งข้อมูลของการเชื่อมต่อบอร์ด Arduino ผ่านพอร์ต Pmod2 เริ่มต้นที่ Pmod2 ส่งข้อมูลผ่านระบบไฟล์เสมือน Sysfs โดยใช้สำหรับกำหนดค่าและตรวจสอบสถานะของขา SDA และ SCL หากเชื่อมต่อสำเร็จ ระบบจะแสดงตำแหน่งของอุปกรณ์รอบข้าง ดังภาพที่ 7

```
root@kr260 $ i2cdetect -y 7
    0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
00:  -- -- -- -- -- -- -- -- 08 -- -- -- -- --
10:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
20:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
30:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
40:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
```

ภาพที่ 7 การทดสอบการเชื่อมต่อไมโครคอนโทรลเลอร์ผ่านพอร์ต Pmod2

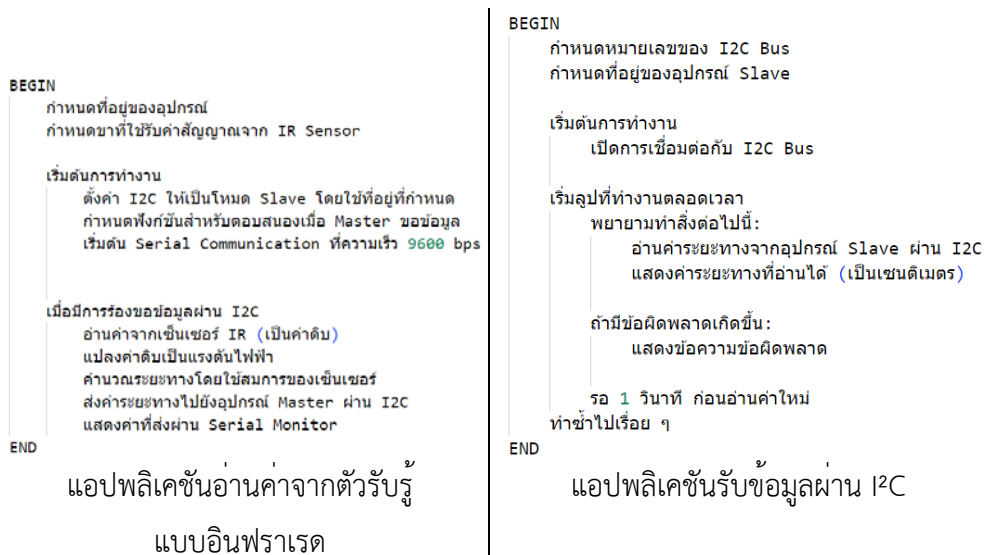
วิธีทดสอบการเชื่อมต่อกับไมโครคอนโทรลเลอร์ผ่านพอร์ต Pmod2 ด้วยอุปกรณ์รอบข้าง จำนวน 3 ชนิด ได้แก่ ตัวรับรู้แบบอินฟราเรด และแอคชูเอเตอร์ 2 ชนิด ประกอบด้วยเซอร์โวมอเตอร์ และมอเตอร์ไฟฟ้ากระแสตรง ดังนี้

1) ตัวรับรู้แบบอินฟราเรด การเชื่อมต่อตัวรับรู้แบบอินฟราเรดกับบอร์ด Arduino โดยต่อขาเอาต์พุตของตัวรับรู้แบบอินฟราเรดเข้ากับขา A0 ขา VCC เข้ากับแหล่งจ่ายไฟฟ้า 5 โวลต์ และขา GND เข้ากับกราวด์ของบอร์ด Arduino ดังภาพที่ 8



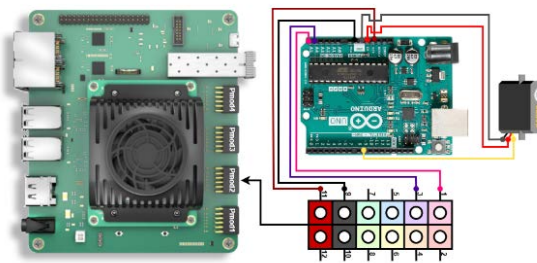
ภาพที่ 8 การเชื่อมต่อกับตัวรับรู้แบบอินฟราเรด

จากนั้นพัฒนาแอปพลิเคชันด้วยภาษาระดับสูง C/C++ บนซอฟต์แวร์ Arduino IDE เพื่ออ่านค่าจากตัวรับรู้แบบอินฟราเรด และส่งข้อมูลไปยังบอร์ด Kria KR260 ผ่านโพรโทคอล I²C ต่อมาใช้ภาษา Python เพื่อให้บอร์ด Kria KR260 รับข้อมูลจากบอร์ด Arduino ผ่านโพรโทคอล I²C หากรับข้อมูลไม่ได้จะแสดงข้อความแจ้งเตือน ดังภาพที่ 9



ภาพที่ 9 แอปพลิเคชันอ่านค่าจากตัวรับรู้แบบอินฟราเรด และรับข้อมูลผ่าน I²C

2) การเชื่อมต่อกับเซอร์โวมอเตอร์ ด้วยการเชื่อมต่อเซอร์โวมอเตอร์กับบอร์ด Arduino โดยต่อสายสัญญาณเข้าขา D9 ขา VCC เข้ากับแหล่งจ่ายไฟฟ้า 5 โวลต์ และขา GND เข้ากับกราวด์ของบอร์ด Arduino ดังภาพที่ 10



ภาพที่ 10 การเชื่อมต่อกับเซอร์โวมอเตอร์

การพัฒนาแอปพลิเคชันควบคุมเซอร์โวมอเตอร์ด้วยภาษา C/C++ บน Arduino IDE ตั้งค่ามุมเริ่มต้นที่ 90 องศา และรับค่ามุมจากบอร์ด Kria KR260 เพื่อควบคุมการหมุน หากค่าที่ได้รับเกิน 180 องศา เซอร์โวมอเตอร์จะหยุดนิ่ง ต่อมาใช้ภาษา Python บนบอร์ด Kria KR260 เพื่อรับค่ามุม และส่งข้อมูลผ่านโปรโตคอล I²C ไปยังบอร์ด Arduino เพื่อควบคุมเซอร์โวมอเตอร์ ดังภาพที่ 11

```
BEGIN
กำหนดที่อยู่ของอุปกรณ์ I2C เป็น 0x08
กำหนดมุมเริ่มต้นของ Servo Motor เป็น 90 องศา

เริ่มต้นการทำงาน
ตั้งค่า I2C ให้เป็นโหมด Slave โดยใช้ที่อยู่ที่กำหนด
กำหนดฟังก์ชันสำหรับรับข้อมูลจาก Master
เชื่อมต่อ Servo Motor กับขา Pin
ตั้งค่ามุมของ Servo Motor เป็น 90 องศา
เริ่มต้น Serial Communication ที่ความเร็ว 115200 bps

เมื่อมีข้อมูลส่งมาจาก Master ผ่าน I2C
ถ้ามีข้อมูลที่รออ่านอยู่:
อ่านค่ามุมที่ได้รับ
จำกัดค่ามุมให้อยู่ในช่วง 0 ถึง 180 องศา
ปรับตำแหน่งของ Servo Motor ตามค่าที่ได้รับ
แสดงค่ามุมที่ตั้งไว้ทาง Serial Monitor

END
```

แอปพลิเคชันควบคุมเซอร์โวมอเตอร์

```
BEGIN
กำหนดหมายเลขของ I2C Bus
กำหนดที่อยู่ของอุปกรณ์ Slave
เปิดการเชื่อมต่อกับ I2C Bus

ฟังก์ชันสำหรับส่งค่ามุมไปยัง Servo Motor:
รับค่ามุมที่ต้องการ
จำกัดค่ามุมให้อยู่ระหว่าง 0 ถึง 180 องศา
ส่งค่ามุมไปยังอุปกรณ์ Slave ผ่าน I2C
แสดงค่ามุมที่ตั้งไว้

เริ่มลูปที่ทำงานตลอดเวลา:
ขอให้ผู้ใช้ป้อนค่ามุมของ Servo (ระหว่าง 0-180)
ถ้าผู้ใช้ป้อน 'q' → ออกจากโปรแกรม

แปลงค่าที่ป้อนไปเป็นตัวเลข
ถ้าค่าไม่ถูกต้อง → แจ้งเตือนผู้ใช้
ถ้าค่าถูกต้อง → ส่งค่ามุมไปยัง Servo Motor
รอ 0.5 วินาที ก่อนรับค่าถัดไป
ทำซ้ำไปเรื่อย ๆ

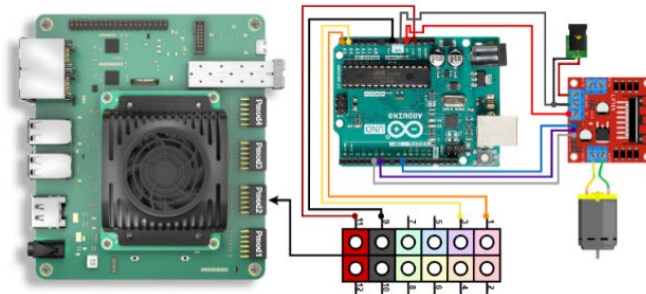
END
```

แอปพลิเคชันการรับค่ามุม

ภาพที่ 11 แอปพลิเคชันควบคุมเซอร์โวมอเตอร์ และการรับค่ามุม

3) วิธีการทดสอบการเชื่อมต่อกับมอเตอร์ไฟฟ้ากระแสตรง ด้วยการเชื่อมต่อมอเตอร์ไฟฟ้ากระแสตรงกับบอร์ด Arduino โดยใช้มอดูล L298N เป็นตัวควบคุม ซึ่งขาเอาต์พุตของมอดูลเชื่อมต่อกับมอเตอร์ไฟฟ้ากระแสตรง และขาสัญญาณควบคุมเชื่อมต่อกับบอร์ด Arduino ดังนั้น ขา IN1 กับขา D5 ขา IN2 กับขา D6 สำหรับการควบคุมทิศทาง และขา

ENA (PWM) กับ D9 สำหรับการควบคุมความเร็ว ส่วนขา 5V และ GND ของมอเตอร์เชื่อมต่อกับแหล่งจ่ายไฟฟ้า 5 โวลต์และกราวด์ของบอร์ด Arduino ตามลำดับ ดังภาพที่ 12



ภาพที่ 12 การเชื่อมต่อกับมอเตอร์ไฟฟ้ากระแสตรง

การพัฒนาแอปพลิเคชันควบคุมมอเตอร์ไฟฟ้ากระแสตรง ตั้งค่าความเร็วเริ่มต้นเป็น 0 บนซอฟต์แวร์ Arduino IDE หากบอร์ด Kria KR260 ส่งค่ามากกว่า 0 มอเตอร์ไฟฟ้ากระแสตรงจะหมุนตามค่าที่ได้รับ แต่หากค่าน้อยกว่าหรือเท่ากับ 0 มอเตอร์ไฟฟ้ากระแสตรงจะหยุด ถัดมาพัฒนาแอปพลิเคชันด้วยภาษา Python บนบอร์ด Kria KR260 เพื่อรับค่าความเร็วจากผู้ใช้งาน และส่งข้อมูลไปยังบอร์ด Arduino ผ่านโพรโทคอล I²C ดังแสดงในภาพที่ 13

```
BEGIN
กำหนดค่าสำหรับควบคุมมอเตอร์:
  MOTOR_IN1 → ขา 5
  MOTOR_IN2 → ขา 6
  MOTOR_SPEED → ขา 9
กำหนดค่าความเร็วเริ่มต้นเป็น 0

เริ่มต้นการทำงาน:
  ตั้งค่าขา MOTOR_IN1, MOTOR_IN2, และ MOTOR_SPEED เป็นโหมด OUTPUT
  กำหนดให้ Arduino เป็น I2C Slave (ที่อยู่ 0x08)
  ตั้งค่าฟังก์ชันรับข้อมูลจาก I2C Master

เริ่มลูปหลัก:
  ตั้งค่าความเร็วของมอเตอร์เป็น speedValue
  ห่วงเวลา 100 มิลลิวินาที
  ทำซ้ำไปเรื่อย ๆ

เมื่อได้รับข้อมูลจาก I2C Master:
  อ่านค่าความเร็วจาก I2C
  ถ้าค่าความเร็ว > 0:
    ตั้งค่าการหมุนของมอเตอร์ (หมุนไปข้างหน้า)
  ถ้าค่าความเร็ว = 0:
    หยุดมอเตอร์
END
```

แอปพลิเคชันควบคุมมอเตอร์ไฟฟ้ากระแสตรง

```
BEGIN
กำหนดที่อยู่ของ I2C Slave
เปิดการเชื่อมต่อ I2C Bus

ฟังก์ชันสำหรับส่งค่าความเร็ว:
  รับค่าความเร็วจากผู้ใช้
  จำกัดค่าให้อยู่ระหว่าง 0 ถึง 255
  ส่งค่าความเร็วไปยัง I2C Slave

เริ่มลูปที่ทำงานตลอดเวลา:
  ขอให้ผู้ใช้ป้อนค่าความเร็ว (0-255)
  แปลงค่าที่ป้อนไปเป็นตัวเลข
  ถ้าค่าถูกต้อง → ส่งไปยังมอเตอร์
  ถ้าค่าผิดพลาด → แจ้งเตือนผู้ใช้
  ทำซ้ำไปเรื่อย ๆ
END
```

แอปพลิเคชันรับค่าความเร็ว
จากผู้ใช้งาน

ภาพที่ 13 แอปพลิเคชันควบคุมมอเตอร์ไฟฟ้ากระแสตรง และรับค่าความเร็วจากผู้ใช้งาน

รายละเอียดอุปกรณ์ ตัวแปรการทดลอง และตัวชี้วัด แสดงดังตารางที่ 1 โดยสรุปรายละเอียดอุปกรณ์ที่ใช้ทดลอง พร้อมแรงดันไฟฟ้า วิธีเชื่อมต่อ และตัวแปรควบคุมของแต่ละอุปกรณ์ เพื่อให้การทดลองทำซ้ำได้อย่างเป็นระบบ รวมทั้งการกำหนดตัวชี้วัดหลัก เช่น ความถูกต้องของข้อมูลรับ-ส่ง สถานะหรือการตรวจจับ ความถูกต้องของมุม และการหมุนและหยุดของมอเตอร์ รวมถึงเวลาตอบสนองและอัตราความสำเร็จของระบบ เป็นต้น

ตารางที่ 1 รายละเอียดอุปกรณ์ ตัวแปรการทดลอง และตัวชี้วัด

รายละเอียดอุปกรณ์	แรงดันไฟฟ้า	การเชื่อมต่อ	ตัวแปรควบคุม	ตัวแปรที่วัด/เกณฑ์
บอร์ด Kria KR260 Robotics Starter Kit (Zynq UltraScale+ MPSoC)	12V	Pmod, AXI GPIO, AXI IIC	Clock 100 MHz, Vivado 2023.1	เวลาตอบสนอง และ ความสำเร็จ
บอร์ด Arduino UNO R3 (ATmega328P)	5V	I ² C (SDA ขา A4 และ SCL ขา A5)	I ² C Address	ข้อมูลที่รับ-ส่ง ตรงกัน
แอลอีดี (LED 5mm) ปุ่มกด และตัวต้านทาน	3.3V	GPIO	R = 220 Ω , กด ปุ่ม 1-2 ครั้ง	สถานะเปิดปิด และ อัตราความสำเร็จ
ตัวรับรูแบบ อินฟราเรด (IR LM393)	5V	Arduino Analog ขา A0	ระยะทดสอบ 5-30 Inch วัตถุที่ใช้ทดสอบ แผ่นกระดาษสี ขาว	ค่าระยะ และสถานะ ตรวจจับ

รายละเอียดอุปกรณ์	แรงดันไฟฟ้า	การเชื่อมต่อ	ตัวแปรควบคุม	ตัวแปรที่วัด/เกณฑ์
เซอร์โวมอเตอร์ (Servo SG90)	5V	Arduino PWM ขา D9	มุม 0–180 องศา	ความถูกต้องมุม และการตอบสนอง
มอเตอร์ไฟฟ้า กระแสตรง และ มอดูล L298N	5V	Direction ขา D5 กับขา D6 และ PWM ขา D9	ความเร็วค่า PWM command 0–150 หรือ Duty Cycle ประมาณ 0–58.8%	การหมุน การหยุด และการตอบสนอง

4.2 การวิเคราะห์ข้อมูลและเกณฑ์การประเมิน

การวิเคราะห์ประสิทธิภาพใช้ตัวชี้วัดเวลาตอบสนอง (Response Time) แบ่งออกเป็น เวลาในการส่งข้อมูล (Send Time) และเวลาในการรับข้อมูล (Receive Time) โดยบันทึกผลการทดสอบกรณีละ 20 ครั้ง และรายงานช่วงค่าต่ำที่สุดถึงมากที่สุด (Min–Max) เพื่อแสดงการกระจายของข้อมูล นอกจากนี้ มีการคำนวณค่าความแปรปรวนสัมพัทธ์ (Normalized Range) (Bandini, 2016) ดังสมการ

$$\text{Normalized Range} = (\text{Max} - \text{Min}) / \text{Mean} \quad (1)$$

เมื่อค่า Mean คือ ค่ากึ่งกลางของช่วงค่าต่ำที่สุดถึงมากที่สุด เพื่อช่วยพิจารณาว่าข้อมูลเกาะกลุ่มหรือแกว่งมาก ส่วนความเสถียรภาพ ได้นิยามในงานวิจัยนี้เป็น อัตราความสำเร็จ (Success Rate) ของการทำงาน และการสื่อสาร ภายใต้เงื่อนไขการทดลองที่กำหนด

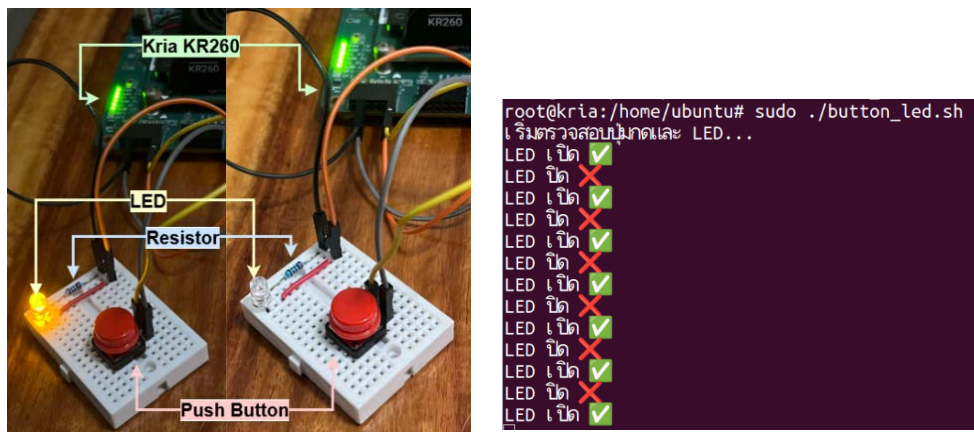
5. ผลการวิจัย

หัวข้อนี้นำเสนอผลการทดสอบ แบ่งเป็น 5 หัวข้อ ได้แก่ 1) การเชื่อมต่ออุปกรณ์ผ่านพอร์ต Pmod 2) การเชื่อมต่อกับตัวรับรู้แบบอินฟราเรด 3) การเชื่อมต่อกับเซอร์โวมอเตอร์

4) การเชื่อมต่อกับมอเตอร์ไฟฟ้ากระแสตรง และ 5) เวลาตอบสนองการทำงานผ่านพอร์ต Pmod มีรายละเอียดดังนี้

5.1 การเชื่อมต่ออุปกรณ์ผ่านพอร์ต Pmod

การทดสอบการควบคุมแอลอีดีด้วยปุ่มกดทำงานได้ตามเงื่อนไขที่กำหนด เริ่มต้นแอลอีดีอยู่ในสถานะปิด กดปุ่มครั้งแรกแอลอีดีจะติด และกดปุ่มครั้งที่สองแอลอีดีจะดับ โดยแสดงสถานะผ่าน เทอร์มินอล (Terminal) ทดสอบทั้งหมด 10 ครั้ง ผลการทดสอบพบว่า ได้ผลถูกต้องครบถ้วนทุกครั้ง ดังแสดงในภาพที่ 14 โดยผลการทดสอบนี้เป็นไปตามวัตถุประสงค์ข้อ 2.2 ในการตรวจสอบการเชื่อมต่อและการทำงานของอุปกรณ์ผ่านพอร์ต Pmod



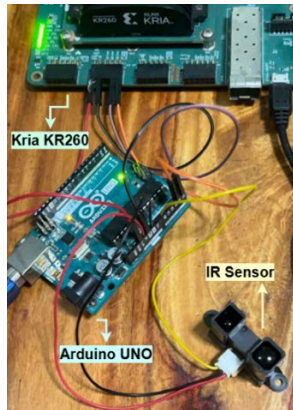
การทดสอบด้วยการกดปุ่ม
เพื่อเปิดปิดแอลอีดี

ผลการทดสอบของแอลอีดีและปุ่มกด
แสดงสถานะผ่านเทอร์มินอล

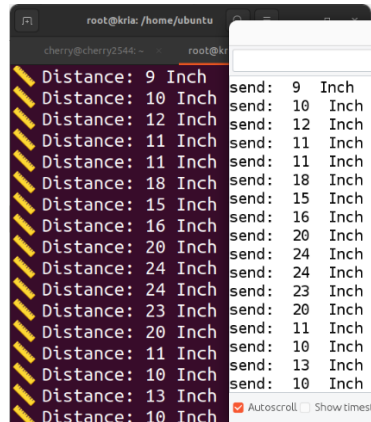
ภาพที่ 14 การทดสอบและผลการทดสอบด้วยการกดปุ่มเพื่อเปิดปิดแอลอีดี

5.2 การเชื่อมต่อกับตัวรับรู้แบบอินฟราเรด

ผลการทดสอบการรับข้อมูลจากตัวรับรู้แบบอินฟราเรดพบว่าสามารถรับและประมวลผลข้อมูลได้ถูกต้อง การสื่อสารระหว่างทั้งสองบอร์ดมีประสิทธิภาพ โดยค่าที่ส่งและรับตรงกันทุกครั้งที่จากการทดสอบ 10 ครั้ง ระยะห่างที่วัดได้ตรงกับค่าที่แสดงบนบอร์ด Arduino โดยปรับระยะห่างระหว่างหน้าตัวรับรู้ถึงแผ่นกระดาษสีขาว ในช่วง 5–30 นิ้ว และบันทึกสถานะการตรวจจับเพื่อนำไปใช้เป็นเงื่อนไขในการสั่งงานมอเตอร์ไฟฟ้ากระแสตรง ดังภาพที่ 15 โดยผลการทดสอบนี้เป็นไปตามวัตถุประสงค์ข้อ 2.2 ในการทดสอบการทำงานร่วมกับอุปกรณ์รอบข้างผ่าน Arduino และ I²C



การทดสอบด้วยตัวรับรู้แบบอินฟราเรด

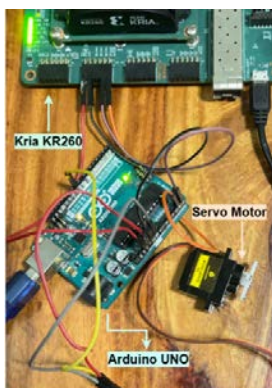


ผลการทดสอบการรับส่งข้อมูล
ของตัวรับรู้แบบอินฟราเรด

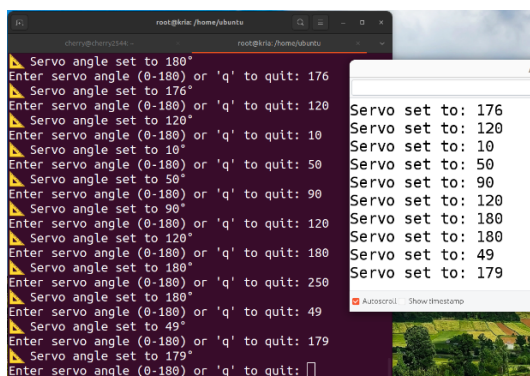
ภาพที่ 15 การทดสอบและผลการทดสอบด้วยการรับส่งข้อมูลของตัวรับรู้แบบอินฟราเรด

5.3 การเชื่อมต่อกับเซอร์โวมอเตอร์

ผลการทดสอบการเชื่อมต่อกับเซอร์โวมอเตอร์พบว่า เซอร์โวมอเตอร์ทำงานได้ถูกต้อง โดยเริ่มต้นที่มุม 90 องศา และสามารถหมุนตามค่าที่บอร์ด Kria KR260 ส่งมาได้อย่างแม่นยำ หากป้อนค่ามุมเกิน 180 องศา ระบบจะจำกัดไว้ที่ 180 องศาโดยอัตโนมัติ จากการทดสอบ 10 ครั้ง เซอร์โวมอเตอร์หมุนตรงตามคำสั่งทุกครั้ง ดังแสดงในภาพที่ 16 โดยผลการทดสอบนี้เป็นไปตามวัตถุประสงค์ข้อ 2.2



การทดสอบการเชื่อมต่อกับ
เซอร์โวมอเตอร์

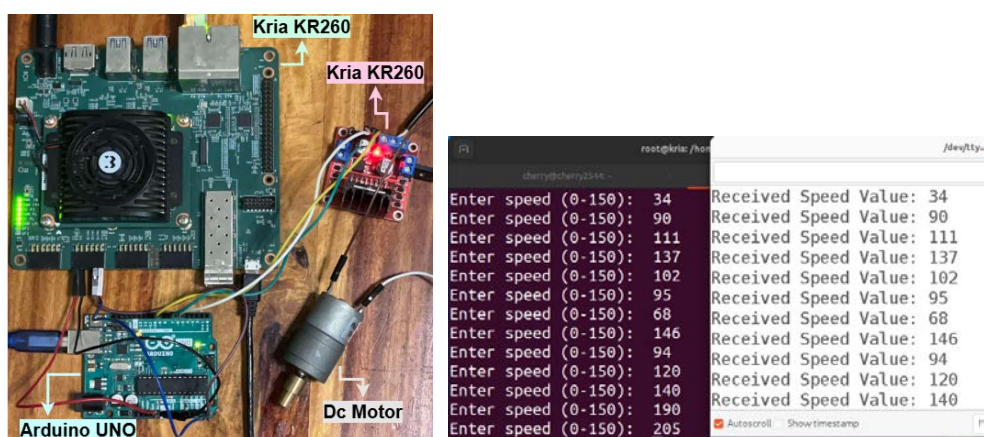


ผลการทดสอบการเชื่อมต่อกับเซอร์โวมอเตอร์

ภาพที่ 16 การทดสอบและผลการทดสอบการเชื่อมต่อกับเซอร์โวมอเตอร์

5.4 การเชื่อมต่อกับมอเตอร์ไฟฟ้ากระแสตรง

ผลการทดสอบการเชื่อมต่อกับมอเตอร์ไฟฟ้ากระแสตรงพบว่า มอเตอร์ไฟฟ้ากระแสตรงทำงานถูกต้องตามค่าความเร็วที่ได้รับจากบอร์ด Kria KR260 โดยมอเตอร์จะหมุนเมื่อค่าความเร็วมากกว่า 0 และหยุดเมื่อค่าน้อยกว่าหรือเท่ากับ 0 หากป้อนค่ามากกว่า 150 ระบบจะจำกัดไว้ที่ 150 หรือ Duty Cycle ประมาณ 0–58.8% โดยอัตโนมัติ การควบคุมผ่านคีย์บอร์ดทำงานได้แม่นยำ โดยค่าที่รับและส่งตรงกันทุกครั้ง ดังแสดงในภาพที่ 17 โดยผลการทดสอบนี้เป็นไปตามวัตถุประสงค์ข้อ 2.2



การทดสอบการเชื่อมต่อกับมอเตอร์

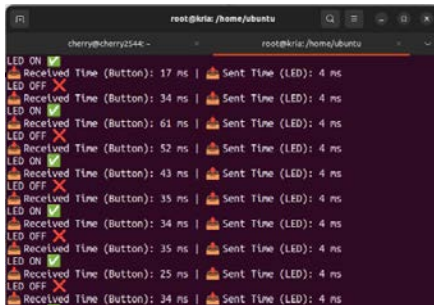
ผลการทดสอบมอเตอร์ไฟฟ้ากระแสตรง

ไฟฟ้ากระแสตรง

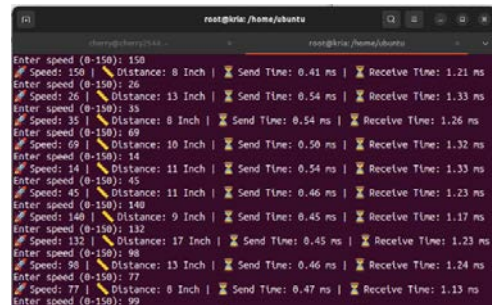
ภาพที่ 17 การทดสอบและผลการทดสอบการเชื่อมต่อกับมอเตอร์ไฟฟ้ากระแสตรง

5.5 เวลาตอบสนองการทำงานผ่านพอร์ต Pmod

จากผลการทดสอบที่กำหนดผ่าน Pmod แบบ GPIO ที่ใช้ปุ่มกดเป็นอินพุตและแอลอีดีเป็นเอาต์พุต เพื่อวัดเวลาตอบสนองจากการกดปุ่มจนแอลอีดีเปลี่ยนสถานะจำนวน 20 ครั้ง และผลการทดสอบแบบ I²C ใช้ตัวรับรู้แบบอินฟราเรดเป็นอินพุต และมอเตอร์ไฟฟ้ากระแสตรงเป็นเอาต์พุต โดยวัดเวลาตั้งแต่ตัวรับรู้แบบอินฟราเรดตรวจจับจนไฟฟ้ากระแสตรงเริ่มทำงาน จำนวน 20 ครั้งเช่นกัน ดังแสดงในภาพที่ 18 ซึ่งผลการวัดเวลาตอบสนองนี้สอดคล้องกับวัตถุประสงค์ข้อ 2.3 การประเมินประสิทธิภาพของแพลตฟอร์มแบบบูรณาการ



ผลการทดสอบเวลาตอบสนองแบบ GPIO



ผลการทดสอบเวลาตอบสนองแบบ I2C

ภาพที่ 18 ผลการทดสอบเวลาตอบสนองแบบ GPIO และ I2C

การประเมินประสิทธิภาพในการทำงานของบอร์ด Kria KR260 ด้วยเวลาตอบสนองการทำงานผ่านพอร์ต Pmod ทั้งแบบ GPIO และแบบ I2C ดังตารางที่ 2 พบว่า I2C มีเวลาตอบสนองเร็วกว่า GPIO อย่างชัดเจน โดยเวลาในการรับข้อมูล GPIO อยู่ที่ 17–61 ms ขณะที่ I2C อยู่ที่ 1.13–1.36 ms ส่วน เวลาส่งข้อมูล GPIO จะคงที่ที่ 4 ms ในขณะที่ I2C อยู่ระหว่าง 0.41–0.54 ms แสดงให้เห็นว่า I2C มีประสิทธิภาพด้านความเร็วสูงกว่า เพื่อให้เห็นการกระจายของข้อมูลเพิ่มเติม ได้สรุปการแปรปรวนสัมพัทธ์จากช่วงค่าต่ำที่สุดถึงมากที่สุด (Min–Max) ในตารางที่ 3 ซึ่งสะท้อนว่าค่าเวลาตอบสนองของ I2C เกาะกลุ่มมากกว่า GPIO

ตารางที่ 2 การประเมินประสิทธิภาพการทำงานของบอร์ด Kria KR260 ร่วมกับไมโครคอนโทรลเลอร์และอุปกรณ์รอบข้าง

เกณฑ์การประเมิน	GPIO	I2C
จำนวนการทดสอบ	20 ครั้ง	20 ครั้ง
เวลาในการส่งข้อมูล (Send Time)	4 ms	0.41 - 0.54 ms
เวลาในการรับข้อมูล (Receive Time)	17 - 61 ms	1.13 - 1.36 ms
อุปกรณ์รอบข้างแบบอินพุต	ปุ่มกด	ตัวรับรูแบบอินฟราเรด
อุปกรณ์รอบข้างแบบเอาต์พุต	แอลอีดี	มอเตอร์ไฟฟ้ากระแสตรง

ตารางที่ 3 แสดงความแปรปรวนของข้อมูลในมุ่เกาะกลุ่มหรือแกว่ง จากช่วงค่า Min–Max โดยใช้ค่าความแปรปรวนสัมพัทธ์ (Normalized Range) เพื่อดูความกว้างของการกระจายเมื่อเทียบกับค่าเฉลี่ย. ผลพบว่า I2C เกาะกลุ่มมากกว่าหรือแกว่งน้อยกว่า ทั้งเวลาในการส่งข้อมูล (Send Time) (0.41–0.54 ms; NR=0.274) และเวลาในการรับข้อมูล

(Receive Time) (1.13–1.36 ms; NR=0.185) ขณะที่ GPIO แกว่งมากกว่า โดยเฉพาะเวลาในการรับข้อมูล (17–61 ms; NR=1.128) สะท้อนการกระจายกว้าง และความสม่ำเสมอของเวลาตอบสนองต่ำกว่า I²C ภายใต้งี๋เงื่อนไขเดียวกัน

ตารางที่ 3 การวิเคราะห์ความแปรปรวนของเวลาตอบสนองจากผล Min–Max

ตัวชี้วัด	GPIO (Min–Max)	GPIO Normalized Range	I ² C (Min–Max)	I ² C Normalized Range
Send Time (ms)	4.00	0.000	0.41–0.54	0.274
Receive Time (ms)	17.00–61.00	1.128	1.13–1.36	0.185
ความแปรปรวนสัมพัทธ์ (Normalized Range)	ต่ำ–ปานกลาง	อ้างอิงช่วง	ต่ำ	อ้างอิงช่วง

6. สรุปผลการวิจัย

งานวิจัยนี้มีวัตถุประสงค์เพื่อศึกษาการใช้งานแพลตฟอร์มพัฒนาหุ่นยนต์ Kria KR260 ร่วมกับไมโครคอนโทรลเลอร์และอุปกรณ์รอบข้างแบบบูรณาการ เพื่อประเมินประสิทธิภาพในการสื่อสารและการควบคุมอุปกรณ์รอบข้าง ผลการทดสอบแสดงให้เห็นว่าแพลตฟอร์ม Kria KR260 สามารถทำงานร่วมกับไมโครคอนโทรลเลอร์ Arduino และอุปกรณ์รอบข้าง เช่น แอลอีดี ตัวรับรู้แบบอินฟราเรด เซอร์โวมอเตอร์ และมอเตอร์ไฟฟ้ากระแสตรง ได้อย่างถูกต้อง ภายใต้งี๋เงื่อนไขการทดลอง โดยมีอัตราความสำเร็จของการทำงานครบถ้วนในการทดสอบแต่ละกรณี การทดสอบการเชื่อมต่อและการสื่อสารผ่านพอร์ต Pmod ทั้งแบบ GPIO และ I²C พบว่าอุปกรณ์ทั้งหมดสามารถทำงานได้ตรงตามคำสั่งทุกครั้ง โดยเฉพาะการสื่อสารแบบ I²C มีประสิทธิภาพสูงกว่าแบบ GPIO อย่างชัดเจน โดยมีเวลาตอบสนองต่ำกว่าถึงระดับมิลลิวินาที ซึ่งสะท้อนถึงความเหมาะสมของการใช้โปรโตคอลนี้ในระบบที่ต้องการการตอบสนองแบบเวลาจริง จึงเหมาะสำหรับการประยุกต์ใช้ในการพัฒนาหุ่นยนต์ในภาคอุตสาหกรรมหรือภาคบริการ งานวิจัยนี้ได้นำเสนอความเสถียรภาพ ในมุมของอัตราความสำเร็จ และความ

สม่ำเสมอของเวลาตอบสนองจากช่วงค่าต่ำที่สุดถึงมากที่สุด งานในอนาคตควรเพิ่มการทดสอบต่อเนื่องระยะยาว และภายใต้สภาพแวดล้อมการรบกวนเพื่อยืนยันความทนทานของระบบ

7. อภิปรายผล

จากผลการทดสอบการเชื่อมต่อและการสื่อสารระหว่างบอร์ด Kria KR260 กับ ไมโครคอนโทรลเลอร์ Arduino และอุปกรณ์รอบข้าง พบว่าระบบสามารถทำงานได้ถูกต้อง และมีเสถียรภาพ ตามนิยามอัตราความสำเร็จ ซึ่งไม่พบการสื่อสารที่ผิดพลาด และคำสั่งไม่ทำงานภายใต้เงื่อนไขการทดลอง ในทุกหัวข้อการทดสอบ การเปรียบเทียบเวลาตอบสนองของพอร์ต Pmod ระหว่างแบบ GPIO และ I²C พบว่า I²C มีเวลาตอบสนองที่ดีกว่าอย่างชัดเจน แสดงให้เห็นว่า I²C มีประสิทธิภาพสูงกว่าในด้านความเร็วและความหน่วงต่ำ ซึ่งเหมาะกับระบบหุ่นยนต์ที่ต้องการการตอบสนองแบบเวลาจริง ผลการทดสอบทั้งหมดสอดคล้องกับคุณลักษณะของชิป Zynq UltraScale+ MPSoC ที่มีทั้งหน่วยประมวลผล Arm Cortex-A53 และ Arm Cortex-R5 ทำให้ระบบสามารถแบ่งภาระการประมวลผลและตอบสนองได้อย่างมีประสิทธิภาพ การเชื่อมต่อกับไมโครคอนโทรลเลอร์และอุปกรณ์ภายนอกจึงมีเสถียรภาพตามอัตราความสำเร็จ ซึ่งยืนยันถึงศักยภาพของแพลตฟอร์มนี้ในการพัฒนาระบบหุ่นยนต์และระบบอัตโนมัติในอนาคต เมื่อพิจารณาหลักการสื่อสารที่ใช้ในการทดสอบ พบว่าการอ่านหรือเขียน GPIO ผ่านระบบไฟล์ Sysfs มีขั้นตอนการเรียกใช้ระดับระบบปฏิบัติการหลายขั้นตอน ส่งผลให้เกิดความหน่วงมากกว่า ขณะที่การสื่อสารผ่าน I²C ใช้บัสอนุกรมที่มีการจัดการแบบเป็นโปรโทคอล ทำให้สามารถรับส่งข้อมูลได้รวดเร็ว และมีความสม่ำเสมอมากกว่า ผลที่ได้จึงสอดคล้องกับแนวคิดการออกแบบระบบฝังตัวที่เลือกใช้บัสสื่อสารสำหรับงานที่ต้องการเวลาตอบสนองต่ำ และสอดคล้องกับงานที่เกี่ยวข้องที่นำ Kria KR260 ไปใช้ในงานเวลาจริงหรือการควบคุมหุ่นยนต์ (Prongnuch & Sitjongsataporn, 2024; Thitinaruemit et al., 2023)

8. เอกสารอ้างอิง

- นทวีร์ ไชยจำ วีระศักดิ์ นาโศก และ นัฐดนัย วงศ์ศรี. (2567). การศึกษาเซ็นเซอร์ตรวจจับอุณหภูมิร่างกายแบบไม่สัมผัสสำหรับคัดกรองผู้ป่วยเบื้องต้น. *วารสารวิศวกรรมศาสตร์และเทคโนโลยีอุตสาหกรรมบ้านสมเด็จ*, 5(1), 47–58.
- วรสุภรางค์ ศิริพราหมณกุล จิโรจน์ สุทธิภาค พรทิพย์ เหลียวตระกูล และ รัตนา สี่รุ่งนาวา รัตน์. (2564). การออกแบบเครื่องทำความสะอาดอัจฉริยะ. *วารสารวิศวกรรมศาสตร์และเทคโนโลยีอุตสาหกรรมบ้านสมเด็จ*, 2(1), 70–88.
- อภิรักษ์ อิตินฤมิตร เศรษฐกุล โปร่งนุช และ ปฏิญญาณ์ แสงอรุณ. (2565). การศึกษาฮาร์ดแวร์ประมวลผล Kria KR260 สำหรับหุ่นยนต์. *วารสาร Engineering Transactions, มหาวิทยาลัยเทคโนโลยีมหานคร*, 25(2), 153-163.
- Advanced Micro Devices, Inc. (2025). *AMD Vivado™ Design Suite*. Retrieved from <https://www.amd.com/en/products/software/adaptive-socs-and-fpgas/vivado.html>
- Advanced Micro Devices, Inc. (2025). *Kria KR260 Robotics Starter Kit*. Retrieved from <https://www.amd.com/en/products/system-on-modules/kria/k260/kr260-robotics-starter-kit.html>
- Aubonnoi, B. & Prongnuch, S. (2025). Functional Evaluation of External Interfaces on the Kria KR260 Robotics Starter Kit for Practical Deployment. *International Electrical Engineering Transactions*, 11(2), 23-31, 2025.
- Bandini, A. (2016). *Applications in Neurophysiology*. (Doctoral dissertation). University of Bologna.
- Li, Z., Li, X., Lin, J., Pang, Y., Yang, D., Zhong, L., & Guo, J. (2023). Design and Application of Multidimensional Force/Torque Sensors in Surgical Robots: A Review. *IEEE Sensors Journal*, 23(12), 12441-12454.
- Prongnuch, S. & Sitjongsataporn, S. (2024). Path Optimization of Small Tractor for Removal Weeding System in Oil Palm Plantation. *ASEAN Engineering Journal*, 14(4), 41-48.

- Prongnuch, S. & Sitjongsataporn, S. (2021). Differential Drive Analysis of Spherical Magnetic Robot Using Multi-Single Board Computer. *International Journal of Intelligent Engineering and Systems*, 14(4), 264-275.
- Prongnuch, S., Sitjongsataporn, S., & Wiangtong, T. (2020). A Heuristic Approach for Scheduling in Heterogeneous Distributed Embedded Systems. *International Journal of Intelligent Engineering and Systems*, 13(1), 135-145.
- Thitinaruemit, A., Sitjongsataporn, S., & Prongnuch, S. (2023). Bio-inspired Algorithm for Decisioning Wireless Access Point Installation. *International Journal of Electrical and Computer Engineering*, 13(5), 4996-5005.