การออกแบบตัวควบคุมกระบวนการระดับน้ำโดยใช้ Lego NXT Mindstrom

ธีรวิชญ์ ศุภศิริพงษ์

วรุฒ์ นวรธนสาร

สิทธานต์ ไชยพงศ์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมการวัดคุม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา ๒๕๕๑

การออกแบบตัวควบคุมกระบวนการระดับน้ำโดยใช้ Lego NXT Mindstrom

ธีรวิชญ์ ศุภศิริพงษ์
วรุตม์ บวรธนสาร
สิทธานต์ ไชยพงศ์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมการวัดคุม
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2555

THE DESIGN OF WATER LEVEL CONTROLLER USING LEGO NXT MINDSTORM

TEERAWICH SUPASIRIPONG
WARUT BOWONTHANASAN
SITTHAN CHAIPONG

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENT FOR THE DEGREE OF
BACHELOR OF ENGINEERING IN INSTRUMENTATION ENGINEERRING
FACULTY OF ENGINEERING
KING MONGKUT'S INSTITUE OF TECHNOLOGY LARDKRABANG
ACADEMIC YEAR 2012

ปริญญานิพนธ์ปีการศึกษา 2555
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ใบรับรองปริญญานิพนธ์

.....

หัวข้อปริญญานิพนธ์	การออกแบบตัวควบคุมกระบวนการระดับน้ำโดยใช้ Lego NXT Mindstrom			
นักศึกษาผู้จัดทำ	นายธีรวิชญ์	ศุภศิริพงษ์	**รหัสนักศึกษา**	52010531
	นายวุรตม์	บวรธนสาร	**รหัสนักศึกษา**	52011073
	นายสิทธานต์	ไชยพงศ์	**รหัสนักศึกษา**	52011271
ปริญญา	วิศวกรรมศาสตรบัณฑิต			
สาขาวิชา	วิศวกรรมการวัดคุม			
ปีการศึกษา	2555			

อาจารย์ผู้ควบคุมปริญญานิพนธ์	**ลายมือชื่อ**
รศ.อาจินต์ น่วมสำราญ	
รศ.ดร.วิทยา ทิพย์สุวรรณพร	

หัวข้อปริญญานิพนธ์	การออกแบบตัวควบคุมกระบวนการระดับน้ำโดยใช้ Lego NXT Mindstrom		
นักศึกษาผู้จัดทำ	นายธีรวิชญ์ ศุภศิริพงษ์	**รหัสนักศึกษา**	52010531
	นายวุรตม์ บวรธนสาร	**รหัสนักศึกษา**	52011073
	นายสิทธานต์ ไชยพงศ์	**รหัสนักศึกษา**	52011271
อาจารย์ที่ปรึกษา	รศ.อาจินต์ น่วมสำราญ		
	รศ.ดร.วิทยา ทิพย์สุวรรณพร		
ปีการศึกษา	2555		

บทคัดย่อ

โครงงานนี้ นำเสนอการออกแบบและจำลองระบบควบคุมกระบวนการระดับน้ำในถัง เพื่อศึกษาการวิเคราะห์และออกแบบกระบวนการควบคุมของน้ำโดยทำการออกแบบกระบวนการของถังน้ำและออกแบบระบบควบคุมแบบดิจิตอลและเขียนโปรแกรมควบคุมโดยใช้โปรแกรม LabVIEW ซึ่งใช้ภาษากราฟฟิกเป็นภาษาของโปรแกรม เพื่อทดลองควบคุมและจำลองการควบคุมกระบวนการระดับน้ำในถังได้อย่างมีประสิทธิภาพ

Thesis Title	THE DESIGN OF WATER LEVEL CONTROLLER USING LEGO NXT MINDSTORM
Authors	Mr. Teerawich Supasiripong
	Mr. Warut Bowonthanasan
	Mr. Sitthan Chaipong
Thesis Advisor	Assoc.Prof. Arjin Numsomran
	Assoc.Prof.Ph.D. Vittaya Tipsuwanporn
Year	2012

ABSTRACT

This thesis presents the design and simulation, process control, water level in the tank.To learn analysis and design process for control water level. To designed the process of water tank. We design digital control system and write program for controlling by using Labview which is a graphic language program. To test control and simulate the process of water level.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงได้ด้วยดีด้วยคำแนะนำคำปรึกษาและอนุเคราะห์อุปกรณ์เครื่องมือในการทำวิจัยจาก รองศาสตราจารย์อาจินต์ น่วมสำราญ และรองศาสตราจารย์ ดร.วิทยา ทิพย์สุวรรณพร ซึ่งเป็นอาจารย์ผู้ควบคุมปริญญานิพนธ์ และเป็นผู้ที่ปรับทัศนคติรวมถึงกระบวนการวิธีคิดอย่างเป็นระบบและการดำเนินชีวิตให้กับผู้วิจัยด้วยดีเสมอมา ผู้วิจัยรู้สึกซาบซึ้งในความอนุเคราะห์ และขอขอบพระคุณเป็นอย่างสูง

ขอขอบพระคุณคณาจารย์สาขาวิศวกรรมการวัดคุม คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ทุกๆท่านที่ได้ประสิทธิ์ประสาทวิชาให้กับผู้วิจัย

ขอขอบคุณเพื่อนๆนักศึกษาสาขาวิศวกรรมการวัดคุมทุกคนที่ให้คำปรึกษาอันเป็นประโยชน์ต่อการศึกษาของผู้วิจัย

ขอขอบคุณ พี่สราวุธ พลเสน สำหรับคำแนะนำและการช่วยเหลือทั้งแรงกายและแรงใจ ซึ่งทำให้ปริญญานิพนธ์มีความสมบูรณ์มากยิ่งขึ้น

สุดท้ายนี้ ผู้วิจัยขอขอบพระคุณ บิดา มารดาและครอบครัวของผู้วิจัยที่ได้ให้กำลังใจและการสนับสนุน รวมถึงแนวทางการดำเนินชีวิตในสังคม ปริญญานิพนธ์ฉบับนี้ จึงสำเร็จลุล่วงด้วยดี คุณค่าและประโยชน์อันพึงมีจากปริญญานิพนธ์ฉบับนี้ ผู้วิจัยขอมอบแด่ผู้มีพระคุณทุกท่าน

คณะผู้จัดทำ

สารบัญ

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญ (ต่อ)

บทที่ 1
บทนำ

1.1 ความสำคัญของปริญญานิพนธ์

ในปัจจุบันนั้นเทคโนโลยีในการควบคุมการไหลของของเหลวในอุตสาหกรรมนั้นได้พัฒนาไปมากมายแต่ที่ยังนิยมใช้และเป็น พื้นฐานในการควบคุมคือการใช้โปรแกรม LabVIEW ในการควบคุม ซึ่งในกระบวนการการควบคุมการไหลของของเหลวนั้นได้ถูกนามาประยุกต์ ใช้ในหลายๆอุตสาหกรรม ทั้งในอุตสาหกรรมน้ำตาล อุตสาหกรรมปิโตรเคมี โครงงานนี้ จึงเป็นโครงงานเกี่ยวกับการควบคุมกระบวนการไหลของของเหลวซึ่งจะพบว่าการควบคุมกระบวนการการไหลของของเหลวนั้นมีปัญหาในด้านการจะรับค่าของระดับน้ำที่ส่งมาจากกระบวนการเราจึงต้องใช้โปรแกรม LabVIEW มาช่วยในการควบคุมการไหลของของเหลว ให้เป็นไปในลำดับขั้นตอนที่เราต้องการ

ในกระบวนการการไหลของของเหลวนั้นจำเป็นที่จะต้องมีการควบคุมอย่างเป็นระบบ เพื่อช่วยในการประหยัดเวลาในการผลิตของผู้ผลิต และช่วยให้เกิดความผิดพลาดของระบบน้อยที่สุดซึ่งจะส่งผลให้สามารถเพิ่มต้นทุนการผลิตอย่างเต็มที่ และมีประสิทธิภาพสูงสุด ซึ่งถ้าหากเกิดการผิดพลาดในการควบคุมการไหลของของเหลวแล้วจะส่งผลกระทบต่อระบบเป็นอย่างมาก ในบางกรณีทำให้เครื่องจักรเสียหายได้ ทั้งยังเสียค่าใช้จ่ายในการผลิตทำให้งานล่าช้าไม่เป็นไปตามที่ผู้ผลิตคาดหวัง ซึ่งทางผู้ศึกษาได้เล็งเห็นความสำคัญของการศึกษากระบวนการการควบคุมการไหลของของเหลวแบบสองถัง โดยจะควบคุมโดยผ่านโปรแกรม LabVIEW โดยจะทำการออกแบบลำดับการควบคุม และตัวควบคุมเพื่อควบคุมระดับน้ำภายในถังให้เป็นไปตามเงื่อนไขที่กำหนด แล้วนำผลการทดลองที่ได้มาทำการเปรียบเทียบว่าการไหลของของเหลวนั้นเป็นไปตามลำดับที่กำหนด และให้ผลตามที่ต้องการอย่างมีประสิทธิภาพต่อระบบกระบวนการการควบคุมการไหลของของเหลว ซึ่งจะก่อเกิดแนวคิดต่อผู้ที่ต้องการศึกษาและผู้ที่สนใจการควบคุมการไหลของของเหลวแบบสองถังต่อไป

1.2 วัตถุประสงค์ของปริญญานิพนธ์

1. เพื่อให้นักศึกษามีความรู้ความเข้าใจกระบวนการ Level water control tank
2. เพื่อให้นักศึกษาสามารถออกแบบและจำลองระบบควบคุมกระบวนการระดับน้ำในถังน้ำสองถัง
3. เพื่อให้นักศึกษาเข้าใจและสามารถออกแบบการควบคุมโดยใช้ LabVIEW ในการควบคุม

1.3 ขอบเขตของปริญญานิพนธ์

1. ศึกษาและออกแบบกระบวนการ Level water control tank
2. ศึกษาการออกแบบโดยใช้โปรแกรม LabVIEW ในชุดจำลอง
3. ศึกษาการใช้โปรแกรม LabVIEW ในการควบคุมระดับน้ำ
4. สามารถอ่านค่าและควบคุมระบบการไหลของถังน้ำสองถังได้ตามต้องการ

1.4 ขั้นตอนการศึกษา

1. วางแผนการทำโครงงาน โดยกำหนดระยะเวลาในการดำเนินงานที่เหมาะสม
2. ดำเนินการทำโครงงานตามแผนการที่วางไว้ โดยทำการกำหนดหน้าที่ของแต่ละคนที่วางไว้
3. ศึกษาหาความรู้และทฤษฎีที่ต้องใช้ในการทำโครงงานและปรึกษาอาจารย์ที่ปรึกษา
4. จัดทำชุดทดลองและชุดทดลองจริง เพื่อทำการทดลองและบันทึกค่า

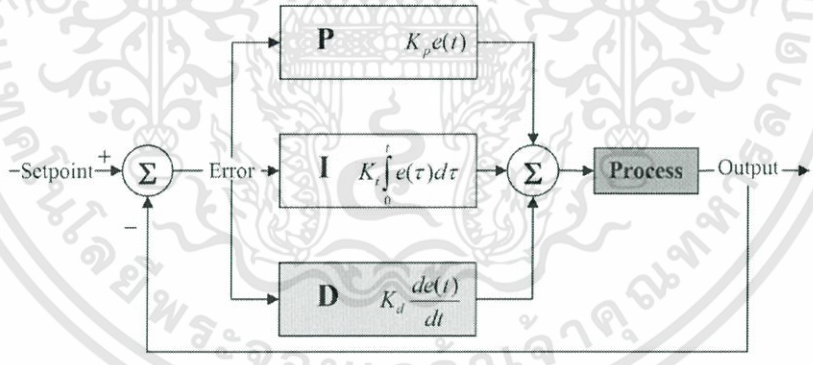
1.5 ประโยชน์ที่คาดว่าจะได้รับ

1. สามารถควบคุมกระบวนการให้เป็นไปตามที่ต้องการได้
2. สามารถออกแบบและจำลองกระบวนการควบคุมได้
3. สามารถใช้โปรแกรม LabVIEW ในการควบคุมกระบวนการได้

บทที่ 2
ทฤษฎีและหลักการที่เกี่ยวข้อง

2.1 ทฤษฎีตัวควบคุมพีไอดี

ระบบควบคุมแบบพีไอดี (PID controller) เป็นระบบควบคุมแบบป้อนกลับที่ใช้กันอย่างกว้างขวาง ซึ่งค่าที่นำไปใช้ในการคำนวณเป็นค่าความผิดพลาดที่หามาจากความแตกต่างของตัวแปรในกระบวนการและค่าที่ต้องการ ตัวควบคุมจะพยายามลดค่าผิดพลาดให้เหลือน้อยที่สุดด้วยการปรับค่าสัญญาณขาเข้าของกระบวนการ ค่าตัวแปรของ PID ที่ใช้จะปรับเปลี่ยนตามธรรมชาติของระบบ วิธีคำนวณของ PID ขึ้นอยู่กับสามตัวแปรคือค่าสัดส่วน (Proportional), ปริพันธ์ (Integral) และ อนุพันธ์ (Differential) ค่าสัดส่วนกำหนดจากผลของความผิดพลาดในปัจจุบัน, ค่าปริพันธ์กำหนดจากผลบนพื้นฐานของผลรวมความผิดพลาดที่ซึ่งพึ่งผ่านพ้นไป และค่าอนุพันธ์กำหนดจากผลบนพื้นฐานของอัตราการเปลี่ยนแปลงของค่าความผิดพลาด น้ำหนักที่เกิดจากการรวมกันของทั้งสามนี้จะใช้ในการปรับกระบวนการ โดยการปรับค่าคงที่ใน PID ตัวควบคุมสามารถปรับรูปแบบการควบคุมให้เหมาะกับที่กระบวนการต้องการได้ การตอบสนองของตัวควบคุมจะอยู่ในรูปของการไหวตัวของตัวควบคุมจนถึงค่าความผิดพลาด ค่าพุ่งเกิน (overshoots) และ ค่าแกว่งของระบบ (oscillation) วิธี PID ไม่รับประกันได้ว่าจะเป็นระบบควบคุมที่เหมาะสมที่สุดหรือสามารถทำให้กระบวนการมีความเสถียรแน่นอน การประยุกต์ใช้งานบางครั้งอาจใช้เพียงหนึ่งถึงสองรูปแบบขึ้นอยู่กับกระบวนการเป็นสำคัญ พีไอดีบางครั้งจะถูกเรียกว่าการควบคุมแบบ PI, PD, P หรือ I ขึ้นอยู่กับว่าใช้รูปแบบใด



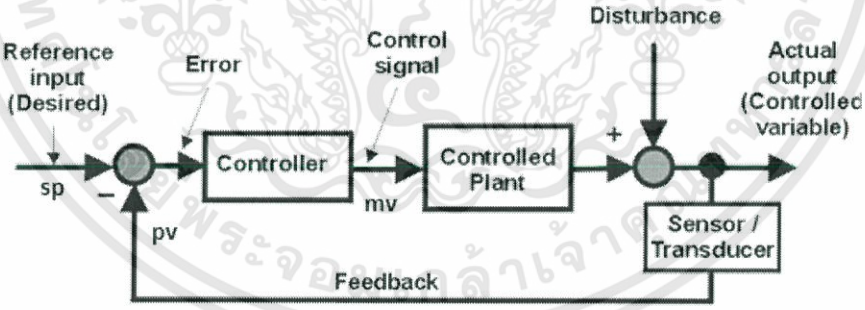
รูปที่ 2.1 แผนภาพแสดงบล็อกของการควบคุมแบบพีไอดี

2.1.1 การควบคุมแบบป้อนกลับด้วยตัวควบคุมพีไอดี

ระบบควบคุมแบบ Closed-Loop เป็นระบบควบคุมแบบหนึ่งซึ่งสัญญาณเอาต์พุตจะมีผลโดยตรงต่อการควบคุมดังนั้นระบบควบคุมแบบ Closed-Loop ก็คือระบบควบคุมป้อนกลับนั่นเอง สัญญาณค่าความคลาดเคลื่อนซึ่งเป็นสัญญาณแตกต่างระหว่างสัญญาณอินพุตกับสัญญาณป้อนกลับจะถูกป้อนให้ตัวควบคุม เพื่อที่จะได้ลดความคลาดเคลื่อนให้น้อยลงและทำให้เอาต์พุตของระบบมีค่าตามที่ต้องการ สัญญาณป้อนกลับนี้อาจเป็นสัญญาณเอาต์พุตโดยตรงหรือเป็นสัญญาณที่เป็นฟังก์ชันของสัญญาณเอาต์พุต

ระบบควบคุมป้อนกลับโดยทั่วไปประกอบด้วยอุปกรณ์ ดังนี้

1. ตัวควบคุม (Controller) คือ ตัววัดสัญญาณเพื่อนำมาเปรียบเทียบเป็นค่าเป้าหมายแล้วคำนวณหาค่าที่เหมาะสมเพื่อส่งเป็นสัญญาณควบคุมออกไปควบคุมกระบวนการ ซึ่งเราสามารถตั้งเป้าหมายให้กับตัวควบคุมนี้ได้ แบบการควบคุมนี้ได้ แบบการควบคุมมีหลายแบบ เช่น ON-OFF Control, P Control, PI Control , PID Control เป็นต้น
2. อุปกรณ์วัด (Measuring Instrument) หมายถึง อุปกรณ์ได้แก่ Sensor, Transducer หรืออุปกรณ์แปลงสัญญาณ (Converter) หรือวัดสัญญาณอื่นๆ ที่มีเอาต์พุตตามสัญญาณมาตรฐานเอาต์พุต
3. อุปกรณ์ปรับกระบวนการ (Final Control Element) เป็นอุปกรณ์ที่ทำหน้าที่ปรับสภาวะของกระบวนการด้วยสภาวะของกระบวนการการเปลี่ยนแปลงตามค่าสัญญาณควบคุม (Manipulated Variable) ของกฎการควบคุมอุปกรณ์พวกนี้ได้แก่ วาล์วควบคุม (Control Valve) , Inverter, Actuator ต่างๆเป็นต้น
4. กระบวนการ (Plant or Process) คือ กระบวนที่ถูกควบคุมหรือกระบวนการทางฟิสิกส์ที่เราต้องการควบคุมให้มีสภาวะตามต้องการ ขณะที่สภาวะการทำงานหรือสภาพแวดล้อมอาจเปลี่ยนแปลงอยู่ตลอดเวลา กระบวนการได้แก่ อุณหภูมิ ความดัน อัตราการไหล ระดับความเป็นกรดด่าง เป็นต้น
5. สัญญาณมาตรฐาน (Standard Signal) ในการเชื่อมต่ออุปกรณ์ในระบบควบคุมอัตโนมัติให้ทำงานได้ตามต้องการนั้นจำเป็นต้องมีมาตรฐานรองรับซึ่งวิวัฒนาการตั้งแต่เริ่มมีระบบควบคุมอัตโนมัติมานั้นก็มีการเปลี่ยนแปลงระบบ ตัวอุปกรณ์เครื่องมือวัดมาตั้งแต่ยุคลม (Pneumatic) , ไฟฟ้า (Electrical) และในปัจจุบันได้มีการใช้สัญญาณดิจิตอลมาใช้ในการสื่อสาร สัญญาณลมที่ใช้คือ 3-15 psi สัญญาณทางไฟฟ้า 1-5 Vdc หรือ 4-20 mA



รูปที่ 2.2 แสดงโครงสร้างของระบบควบคุมแบบป้อนกลับโดยทั่วไป

การควบคุมกระบวนการทางอุตสาหกรรม โดยทั่วไปนิยมใช้ตัวควบคุมแบบพีไอดี (PID Controller) เพราะรูปแบบของตัวควบคุมเป็นตัวควบคุมที่สามารถควบคุมกระบวนการต่างๆได้อย่างกว้างขวาง เนื่องจากมีโครงสร้างการทำงานที่ไม่ซับซ้อน สามารถเข้าใจได้ง่าย การใช้งานตัวควบคุมพีไอดีนี้ขึ้นอยู่กับการปรับค่าพารามิเตอร์ของตัวควบคุม PID ให้เหมาะสม เพื่อให้ได้ผลตอบสนองของกระบวนการตามต้องการ

ตัวควบคุม PID ประกอบไปด้วยตัวควบคุมแบบ Proportional (P) ตัวควบคุมแบบ Integral (I) และตัวควบคุมแบบ Derivative (D) ซึ่งมีฟังก์ชันถ่ายโอน (Transfer Function) ดังนี้

$$m(t) = K_p\left(e(t) + \frac{1}{T_i}\int_0^t e(t)dt + T_d\frac{de(t)}{dt}\right) \quad (2.1)$$

โดย K_p คือ ค่าอัตราขยายของตัวควบคุมแบบ P (Proportional Gain)

T_i คือ ค่าเวลา (Integral Time)

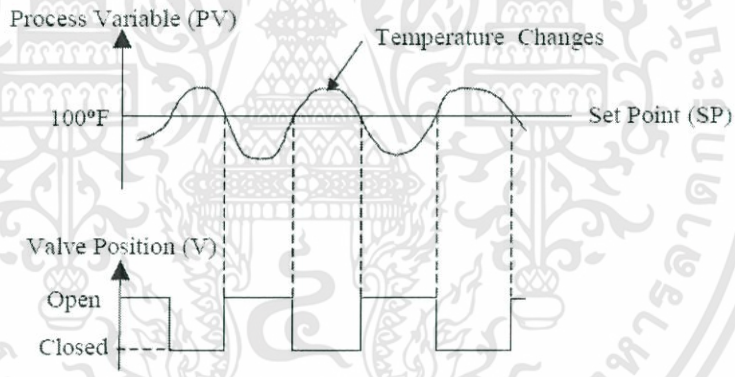
T_d คือ ค่าเวลา Derivative (Derivative Time)

จากรูปที่ 2.1 จะเห็นได้ว่าสัญญาณควบคุม หรือตัวแปรปรับกระบวนการ (mv) ที่ได้จากตัวควบคุม PID จะถูกกำหนดด้วยความสัมพันธ์ด้วยความสัมพันธ์ระหว่างสัญญาณขาเข้า ตัวควบคุมกับตัวแปรกระบวนการ (pv) กับสัญญาณอ้างอิงหรือค่าเป้าหมาย (sp) โดยที่ความสัมพันธ์ดังกล่าวจะขึ้นอยู่กับกฎเกณฑ์การควบคุมที่ผู้ควบคุมปรับแต่งไว้ล่วงหน้า ค่าพารามิเตอร์ของตัวควบคุมจะเป็นไปตามกริยาควบคุมแบบต่างๆดังกล่าวต่อไปนี้

2.1.2 กริยาการควบคุมแบบป้อนกลับ

2.1.2.1 กริยาการควบคุมแบบ ON-OFF

การควบคุมแบบ ON-OFF เป็นการควบคุมที่ง่ายที่สุดและนิยมใช้ในการควบคุมกระบวนการที่ไม่ต้องการความเที่ยงตรงสูง โดยการควบคุมจะทำงานเพียง 2 สถานะ คือ เปิด (100%) และปิด (0%) กริยาควบคุมแบบ ON-OFF ดังแสดง ภาพที่ 2.3



รูปที่ 2.3 แสดงกริยาการควบคุม ON-OFF

จากรูปที่ 2.3 จะเห็นว่าถ้าค่าคลาดเคลื่อนมากกว่าค่าวิกฤต ($+\varepsilon$) ค่าเอาต์พุตของตัวควบคุมคุมจะเปลี่ยนจาก 0% เป็น 100% เมื่อค่าความคลาดเคลื่อนน้อยกว่าค่าวิกฤต($-\varepsilon$) ค่าเอาต์พุตของตัวควบคุมจะเปลี่ยนจาก 100% เป็น 0 % ค่าเอาต์พุตที่อยู่ในช่วงเดทแบนด์ (Dead Band) จะไม่มีการเปลี่ยนแปลงแต่อย่างใดซึ่งอาจจะเป็นผลมาจากความเสียดทานที่ไม่ได้คาดไว้ก่อน หรือบางครั้งก็จำเป็นต้องทำให้เกิดช่วงเดทแบนด์ขึ้น เพื่อป้องกัน ON-OFF บ่อยเกินไป อันจะทำให้อุปกรณ์ควบคุมหรือกระบวนการได้รับความเสียหาย แต่ช่วงเดทแบนด์นี้ต้องไม่กว้างนัก เพราะจะทำให้ค่าความเที่ยงตรงของการควบคุมลดลง กริยาการควบคุมแบบ ON-OFF สามารถเขียนเป็นสมการ ได้ดังนี้

$$m(t) = \begin{cases} 0\% & , e(t) < -\varepsilon \\ 100\% & , e(t) > +\varepsilon \end{cases} \quad (2.2)$$

เมื่อ m(t) คือ สัญญาณควบคุมหรือเอาต์พุตของตัวควบคุม
e(t) คือ ค่าความคลาดเคลื่อน
ε คือ ½ ของค่าเดทแบนด์

2.1.2.2 กริยาการควบคุมแบบ Proportional (P)

กริยาการควบคุม P นั้น ค่าเอาต์พุตของตัวควบคุมจะแปรผันตรงกับค่าความคลาดเคลื่อน กล่าวคือถ้าค่าความคลาดเคลื่อนมีค่ามากขึ้น ค่าเอาต์พุตของตัวควบคุมก็จะมีค่ามากขึ้นตาม และถ้าค่าความคลาดเคลื่อนมีค่าน้อยลง ค่าเอาต์พุตของตัวควบคุมก็จะมีค่าน้อยลงตามกริยาการควบคุมแบบ P สามารถเขียนสมการได้ดังนี้

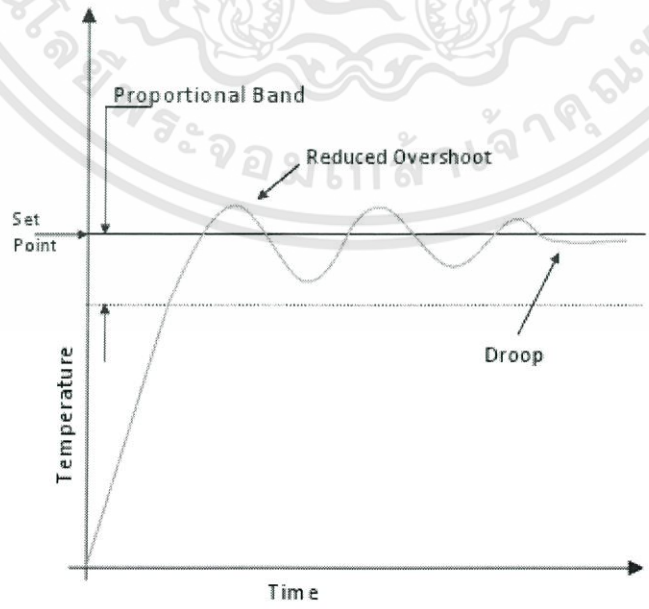
$$m_p(t) = K_p e(t) + \bar{m} \tag{2.3}$$

เมื่อ $m_p(t)$ คือ ค่าเอาต์พุตของตัวบคุมแบบ Proportional
K_p คือ อัตราขยายของตัวควบคุม แบบ Proportional
$\bar{m}$ คือ ค่าเอาต์พุตของตัวควบคุมที่ค่าความคลาดเคลื่อนเท่ากับศูนย์

ตัวควบคุมแบบ P บางตัวอาจจะใช้ค่า Proportional Band (PB) แทนการใช้ค่า K_p ซึ่ง PB คือช่วงของความคลาดเคลื่อนระหว่างที่เอาต์พุตของตัวควบคุมมีค่า 0 – 100% ดังสมการ

$$PB = \frac{100\%}{K_p} \tag{2.4}$$

ข้อเสียของกริยาควบคุมแบบ Proportional คือไม่สามารถกำจัดค่าออฟเซตได้



รูปที่ 2.4 แสดงคุณสมบัติของกริยาควบคุมแบบ Proportional

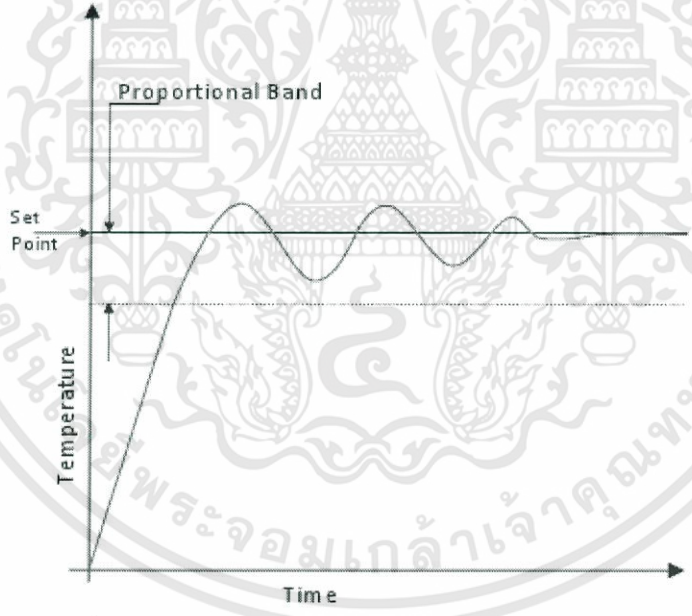
2.1.2.3 กริยาควบคุมแบบ Integral (I)

กริยาควบคุมแบบ I หรือเรียกอีกอย่างหนึ่งว่า การควบคุมแบบรีเซต (Reset Control) โดยค่าเอาต์พุตของตัวควบคุมหาได้จากพื้นที่ทั้งหมดภายใต้ของค่าความคลาดเคลื่อนต่อเวลาคูณกับค่าคงที่ที่เรียกว่า อันตราขยายของตัวควบคุมแบบ I (Integral Gain) ดังสมการต่อไปนี้

$$m_I(t) = K_I \int_0^t e(t)dt + \overline{m_I}(0) \qquad (2.5)$$

เมื่อ $m_I(t)$ คือ ค่าเอาต์พุตของตัวควบคุมแบบ Integral
K_I คือ อัตราขยายของตัวควบคุมแบบ Integral
$\int_0^t e(t)dt$ คือ พื้นที่ทั้งหมดของค่าคลาดเคลื่อน
$\overline{m_I}(0)$ คือ ค่าเอาต์พุตของตัวควบคุมที่เวลา t เท่ากับศูนย์

ผลของกริยาควบคุมแบบ I นี้จะทำให้ไม่เกิดออฟเซ็ตขึ้นในระบบ และลดค่าพุ่งเกิน (Overshoot) ของระบบลงได้ แต่ถ้ากริยาการควบคุมมีค่าสูงเกินไป จะทำให้ผลตอบสนองของกระบวนการช้าลง ผลตอบสองของกริยาควบคุมแบบ I ดังแสดงในรูปที่ 2.5



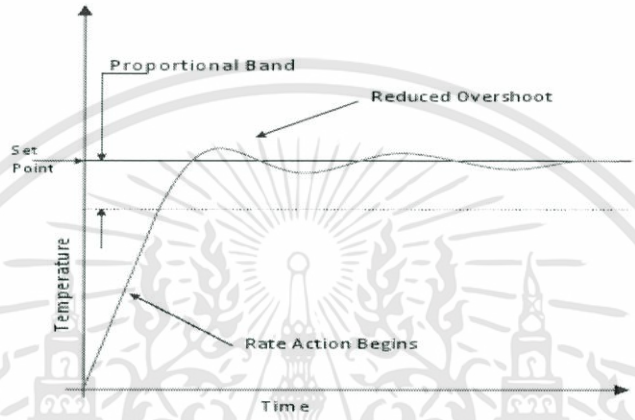
รูปที่ 2.5 แสดงผลตอบสนองของกริยาการควบคุมแบบ Integral

2.1.2.4 กริยาการควบคุมแบบ Derivative (D)

กริยาการควบคุมแบบ D ยกอีกชื่อหนึ่งว่า การควบคุมแบบอัตราส่วน (Rate Action) โดยสัญญาณเอาต์พุตของตัวควบคุมจะขึ้นอยู่กับอัตราการเปลี่ยนแปลงของค่าความคลาดเคลื่อนต่อเวลา จะเห็นว่าค่าความคลาดเคลื่อนนี้มีโอกาสเป็นศูนย์ได้ และค่าเอาต์พุตก็สามารถเปลี่ยนแปลงให้มีค่าสูงขึ้น เมื่อความคลาดเคลื่อนเปลี่ยนแปลง ซึ่งเรียกการกระทำดังกล่าวว่า อัตราการกระทำ (Rate Action) ดังสมการ

$$m_D(t) = K_D \frac{de(t)}{dt} \qquad (2.6)$$

เมื่อ $m_D(t)$ คือ ค่าเอาต์พุตของตัวควบคุมแบบ Derivative
K_D คือ อัตราขยายของตัวควบคุมแบบ Derivative
$\frac{de(t)}{dt}$ คือ ค่าความคลาดเคลื่อนที่เวลา



รูปที่ 2.6 แสดงตัวอย่างผลตอบสนองของกริยาควบคุมแบบ Derivative

2.1.2.5 กริยาควบคุมแบบ Proportional – Integral (PI)

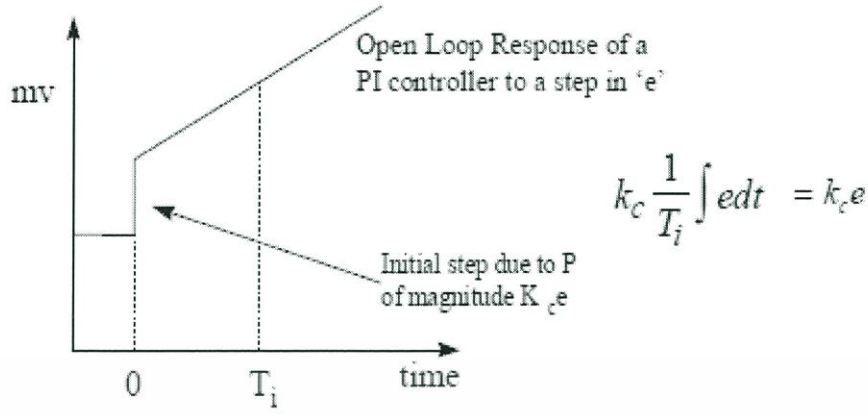
ตามที่กล่าวมาแล้วว่ากริยาควบคุมแบบ P นั้น จะมีค่าออฟเซ็ทเกิดขึ้น ซึ่งการกำจัดค่าออฟเซ็ทนี้สามารถทำได้โดยการเพิ่มกริยาควบคุมแบบ I ดังสมการต่อไปนี้

$$m_{PI}(t) = \bar{m} + K_p e(t) + K_p K_I \int_0^t e(t)dt \qquad (2.7)$$

หรือ

$$m_{PI}(t) = \bar{m} + K_p e(t) + \frac{K_c}{T_i} \int_o^t e(t)dt \qquad (2.8)$$

เมื่อ $K_c = K_p$ และ $K_I = \frac{1}{T_i}$
T_i คือ ค่าเวลา Integral



รูปที่ 2.7 แสดงตัวอย่างผลตอบสนองของกริยาควบคุมแบบ PI (Direct action)

2.1.2.6 กริยาการควบคุมแบบ Proportional-Derivative (PD)

การประยุกต์ใช้กริยาการควบคุมแบบ P ร่วมกับกริยาการควบคุมแบบ D เพื่อทำให้ผลตอบสนองของระบบรวดเร็วขึ้น แต่จะไม่มีผลโดยตรงต่อผลตอบสองของระบบที่สภาวะคงที่ ซึ่งสมการเอาต์พุตของกริยาการควบคุมแบบ PD แสดงดังสมการ

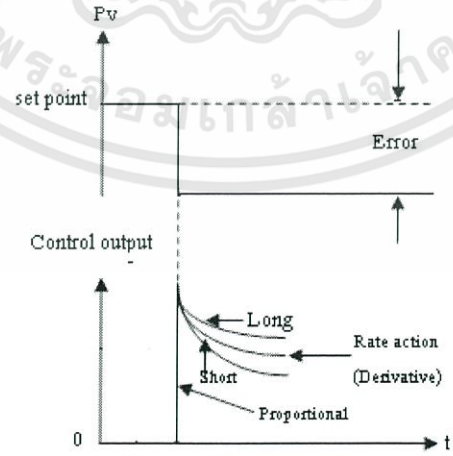
$$m_{PD}(t) = \overline{m} + K_P e(t) + K_P K_D \frac{de(t)}{dt} \tag{2.9}$$

หรือ

$$m_{PD}(t) = \overline{m} + K_P e(t) + K_c T_d \frac{de(t)}{dt} \tag{2.10}$$

เมื่อ $K_D = T_d$

T_d คือ ค่าเวลา Derivative



รูปที่ 2.8 แสดงตัวอย่างผลตอบสนองของกริยาการควบคุมแบบ PD

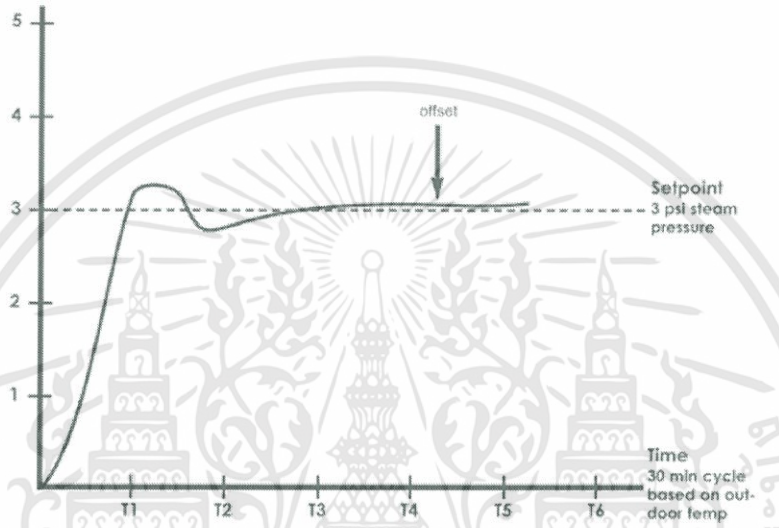
2.1.2.7 กริยาการควบคุมแบบ Proportional – Integral – Derivative (PID)

เพื่อให้ผลตอบสองของระบบควบคุมมีสมรรถนะเป็นไปตามต้องการ จึงใช้กริยาการควบคุมทั้งสามแบบร่วมกัน ซึ่งจะทำให้ได้กริยาการควบคุมแบบ PID ดังสมการ

$$m_{PID}(t) = \bar{m} + K_p e(t) + K_p K_I \int_0^t e(t)dt + K_p K_D \frac{de(t)}{dt} \qquad (2.11)$$

หรือ

$$m_{PID}(t) = \bar{m} + K_c e(t) + \frac{K_c}{T_i} \int_0^t e(t)dt \quad + K_c T_d \frac{de(t)}{dt} \qquad (2.11)$$



รูปที่ 2.9 แสดงตัวอย่างผลตอบสนองของกริยาการควบคุมแบบ PID

2.2 ระบบควบคุมแบบปรับตัวเองได้ (Adaptive Control System)

ตัวควบคุมแบบ Adaptive หมายถึงตัวควบคุมที่สามารถปรับตัวเองได้ การตอบสนองต่อการเปลี่ยนแปลงที่พลศาสตร์ของกระบวนการและคุณสมบัติของสิ่งรบกวน

วิธีการของ Ziegler Nichols คือการกำหนดปฏิกิริยาค่าคงที่ให้กับพารามิเตอร์ตัวควบคุมแบบพีไอดี คือ K_c, T_i และ T_d แต่บางระบบไม่สามารถบอกได้จึงทำให้ค่าพารามิเตอร์ของตัวควบคุมแบบพีไอดีเหล่านี้ไม่สามารถทำได้อย่างมีประสิทธิภาพ แต่ตัวควบคุมแบบปรับตัวเองได้ (Adaptive Controller) สามารถช่วยให้ระบบมีเสถียรภาพและการตอบสนองของระบบมีผลที่ดี การเปลี่ยนแปลงสัมประสิทธิ์และอัลกอริทึมของการควบคุมแบบ Real time เพื่อชดเชยการเปลี่ยนแปลงในระบบด้วยตัวของมันเอง โดยทั่วไปตัวควบคุมจะตรวจสอบการทำงานของระบบเป็นระยะๆจากทรานส์เฟอร์ฟังก์ชันของระบบและปรับเปลี่ยนอัลกอริทึมการควบคุม โดยมันสามารถเรียนรู้เกี่ยวกับพฤติกรรมของกระบวนการในขณะที่กำลังทำการควบคุมไปพร้อมๆกันได้

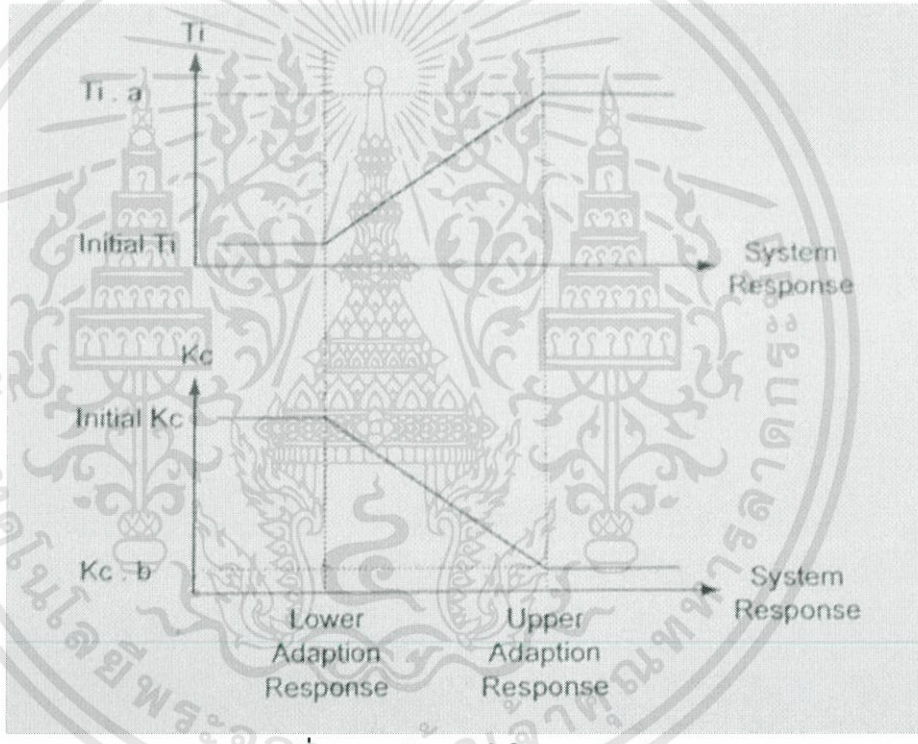
2.2.1 อัลกอริทึมการปรับตัวเองสำหรับพารามิเตอร์ K_c และ T_i ของตัวควบคุมพีไอดี

เกณฑ์ K_c และ T_i ของตัวควบคุมแบบพีไอดีมีผลโดยตรงต่อผลตอบสนองของระบบ เทอมของ มีผลต่อความสัมพันธ์กับความผิดพลาดของระบบที่แตกต่างกันระหว่างระบบอ้างอิงกับ

เอาต์พุตของระบบและเกณฑ์ K_c มีผลโดยตรงต่อค่าคงที่ทางเวลา (Time Constant) และค่าพุ่งเกิน (Overshoot) ของระบบ บางครั้งก็ไม่ได้กล่าวถึง T_d เนื่องจากตัวควบคุม PI ก็เพียงพอสำหรับระบบในอุตสาหกรรม ซึ่งตรงกันข้ามเกณฑ์ K_c และเทอม T_i ที่ใช้ในการปรับตัวของพีไอดี เนื่องจากว่าตัวควบคุมแบบพีไอดีจะกล่าวว่าเป็นตัวควบคุมแบบ PI-Adaptive Self Tune Based และ D-Self Tune Based (PI-D) อัลกอริทึมของเทอมพีไอกำหนดไว้ดังภาพที่ 3.11 และสมการของพีไอกำหนดดังสมการ

$$T_{i_{ADAPT}} = \frac{T_{i_{(S.TUNE)}}(a-1)}{U_{ADAPT} - L_{ADAPT}}(R_{RESPONSE} - L_{ADAPT}) + T_{i_{(S.TUNE)}} \quad (2.12)$$

$$T_{i_{ADAPT}} = \frac{K_{C_{(S.TUNE)}}(a-1)}{U_{ADAPT} - L_{ADAPT}}(R_{RESPONSE} - L_{ADAPT}) + K_{C_{(S.TUNE)}} \quad (2.13)$$



รูปที่ 2.10 อัลกอริทึมในการปรับตัวเอง

จากรูปที่ 2.10 ค่าเริ่มต้นของ และ เป็นค่าสัมประสิทธิ์ที่พบในวิธีการหาปฏิกิริยาของกระบวนการของ Ziegler-Nichols “a” คือสัมประสิทธิ์ของเทอม เหนือเส้นกราฟ และ “b” คือสัมประสิทธิของเทอม ใต้เส้นกราฟซึ่งสามารถกล่าวได้ว่า a มีค่ามากว่า 1 และ b มีค่าระหว่าง 0 ถึง 1 ผลตอบสนองของระบบแทนด้วยสัญลักษณ์ทางเอาต์พุต (Votage, Degree, Speed, Moment และอื่นๆ) ตัวอย่างเช่น ผลตอบสนองเอาต์พุตของความเร็วมอเตอร์ , Lower Adaption Response คือค่าที่ต่ำกว่าความเร็วมอเตอร์(rpm) และ Upper Adaption Response คือค่าที่สูงกว่าความเร็วมอเตอร์ (rpm) ซึ่งอัลกอริทึมของ Adaptive จะทำงานอยู่ในช่วงนี้

เส้น Lower Adaption อัลกอริทึมจะเริ่มทำงานโดยเพิ่มค่าและลดค่า ซึ่งเหตุผลในการเพิ่มค่า เพื่อทำให้ค่าผิดพลาดมีค่าน้อยที่สุดก่อนเข้าสภาวะคงตัวและเหตุผลในการลดค่า เพื่อไม่ทำให้มีค่าพุ่งเกินหรือไม่ให้ผลตอบสองของระบบมีค่าพุ่งเกินที่สูงเกินไป

2.2.2 วิธีการ Trial and Error (วิธีการลองผิดลองถูก)

เป็นวิธีการที่เหมาะสำหรับผู้ที่มีประสบการณ์ในด้านการปรับพารามิเตอร์ PID แล้ว โดยการควบคุมจะต้องเป็นระบบปิดและตำแหน่งการควบคุมจะต้องอยู่ในตำแหน่ง Automatic Control ส่วนวิธีการต่อไปนี้ก็คือ

2.2.2.1 วิธีการปรับแบบ PI

1.ให้ตัวควบคุมทำงานในรูปแบบ Proportional Control เพียงอย่างเดียว

2.ต่อไปทำการปรับค่า K_c จนกระทั่งค่าPV เข้าใกล้ค่าเป้าหมายซึ่งโดยทั่วไปก็จะต่ำกว่าค่าเป้าหมายเพียงเล็กน้อย

3.ต่อไปค่อยๆปรับค่า T_i เพิ่มขึ้นเพื่อชดเชยค่า Offset ที่เกิดขึ้นแล้วดูผลว่าค่า PV เข้าสู่เป้าหมายตามที่ต้องการหรือยัง ถ้ายังก็ค่อยๆปรับขึ้นอีกจนกระทั่งได้ตามต้องการโดยช่วงนี้ก็พยายามลด K_c เพื่อลดการแกว่ง

2.2.2.2 วิธีการปรับแบบ PID

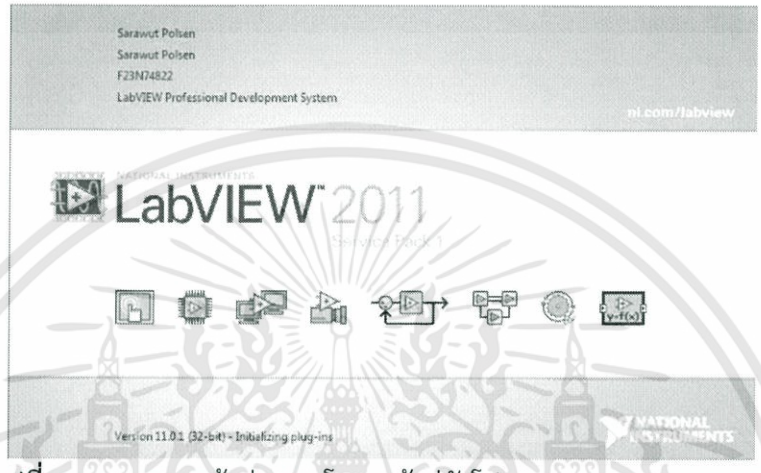
1.ทำตามวิธีการแบบ PI ทั้ง 3 ข้อ

2.ถ้าต้องการผลตอบสนองที่รวดเร็วขึ้นมักใช้ค่าอัตราส่วนระหว่างค่า Integral Time กับ Derivative Time เป็น 4/1 เป็นต้นไปพร้อมกับการเพิ่มลดของค่า K_c ไปด้วย เพื่อ ป้องกันการเกิด Overshoot และการเกิด Hunting

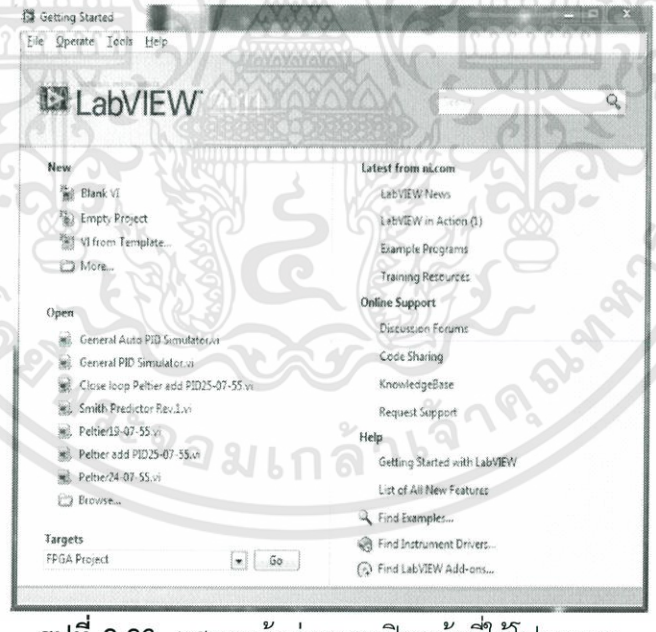
3.เนื่องด้วยวิธีการนี้เป็นความพึงพอใจเฉพาะบุคคลจึงยากที่จะกำหนดเป็นกฎเกณฑ์ตายตัว แต่วิธีการเริ่มต้นในการปรับนั้นมักเหมือนกัน

2.3 การใช้งานโปรแกรม LabVIEW

เมื่อผู้ใช้ดับเบิ้ลคลิ๊กที่ไอคอนโปรแกรม LabVIEW 2011 จะปรากฏหน้าต่างดังรูปที่ 2.22 หน้าต่างนี้จะแสดงการโหลดเข้าสู่ตัวโปรแกรม LabVIEW 2011 เมื่อเข้าสู่ตัวโปรแกรมแล้วจะปรากฏหน้าต่างดังรูปที่ 2.23 ทำการเลือกคลิ๊กที่ Blank VI เพื่อเข้าสู่หน้าต่างการเขียนโปรแกรม LabVIEW 2011

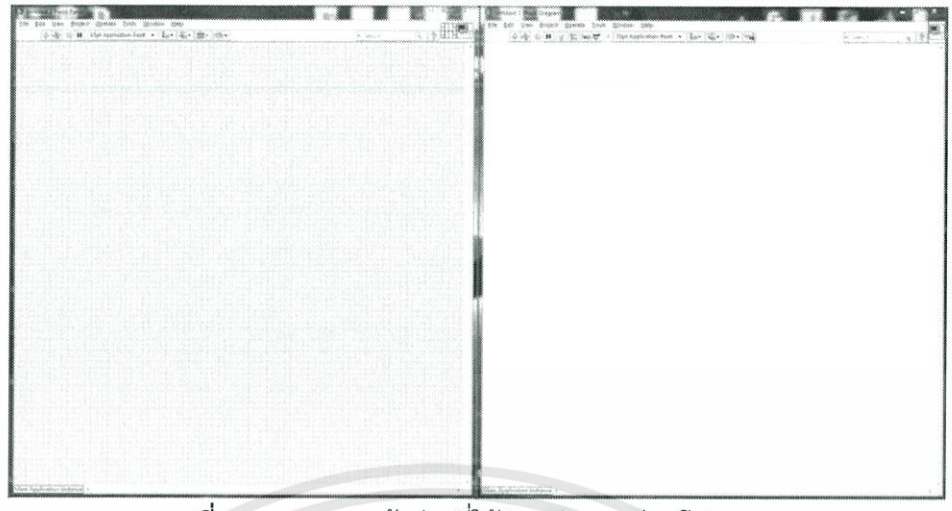


รูปที่ 2.22 แสดงหน้าต่างการโหลดเข้าสู่ตัวโปรแกรม LabVIEW 2011



รูปที่ 2.23 แสดงหน้าต่างการเปิดหน้าที่ใช้โปรแกรม

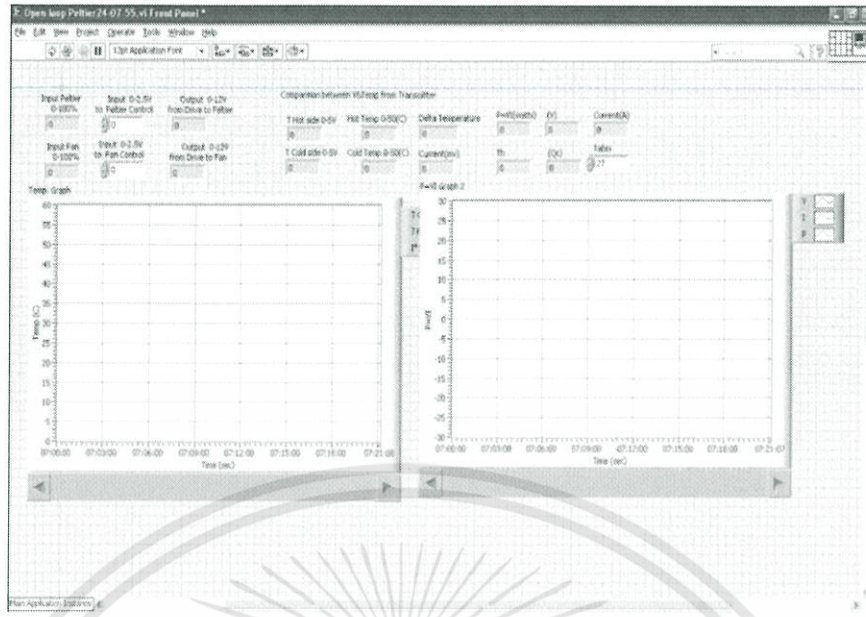
เมื่อคลิ๊กเลือก Blank VI แล้วจะปรากฏหน้าต่างที่ใช้สำหรับเขียนโปรแกรม LabVIEW 2011 ขึ้นมา 2 หน้าต่างดังรูปที่ 2.24 โดยด้านซ้ายเรียกว่า พาเนล และด้านขวาเรียกว่า บล็อกไดอะแกรม ตามลำดับ



รูปที่ 2.24 แสดงหน้าต่างที่ใช้สำหรับการเขียนโปรแกรม

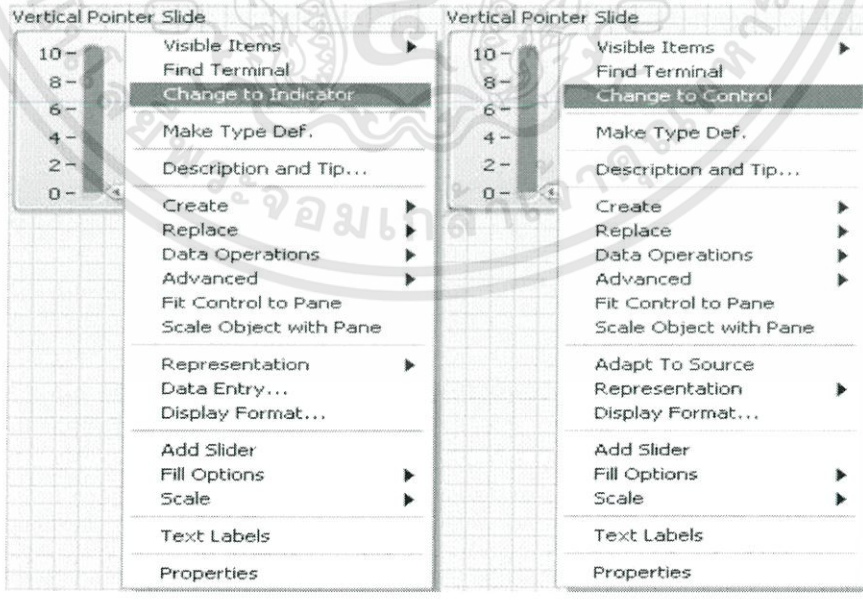
2.3.1 พาเนล

พาเนลเป็นส่วนที่ใช้ในการแสดงผลค่าต่างๆ ที่ผู้ใช้ต้องการทราบ ไม่ว่าจะการแสดงผลในเชิงตัวเลข ตัวอักษร กราฟ และอื่นๆ และใช้ในการควบคุมโดยการป้อนค่าที่ผู้ใช้ต้องการ ไม่ว่าจะเป็นตัวเลขหรือเป็นอักษรก็ตาม พูดง่ายๆก็คือพาเนลเป็นได้ทั้ง indicator และ control หรือเป็น Output และ Input นั่นเอง ซึ่งผู้ใช้สามารถสร้างรูปแบบขึ้นเองได้ตามความพอใจของผู้ใช้ ส่วนประกอบที่ของพาเนลมีอยู่ 3 ส่วน คือ พื้นที่สำหรับเขียนโปแกรม เมนูบาร์ และทูลบาร์ การเขียนโปรแกรมลงบนพาเนลนั้นไม่ใช้เรื่องยาก เพียงแต่ให้ผู้ใช้เข้าใจเสียก่อนว่าผู้ใช้ต้องการเขียนอะไรลงบนพาเนล คือ ต้องการให้ตัวใดเป็นตัวแสดงผลหรือต้องการให้ตัวใดเป็นตัวควบคุม เพราะ Tool หนึ่งตัวสามารถทำหน้าที่ได้เพียงหน้าที่เดียว คือ ถ้าไม่ตัวควบคุมก็ต้องเป็นตัวแสดงผลในส่วนของพาเนล ผู้ใช้สามารถคลิ๊กเมาส์ทางด้านขวาบนหน้าต่างพาเนล จะปรากฏบล็อกทูลพาเลทขึ้นมาซึ่งภายในทูลพาเลทจะประกอบด้วยตัวควบคุมและตัวแสดงผลแบบต่างๆ การวางอุปกรณ์ทำได้โดยการใช้เมาส์คลิ๊กค้างไว้ที่อุปกรณ์ที่ต้องการจากนั้นลากไปวางในหน้าต่างพาเนลเป็นอันเสร็จเรียบร้อย หน้าต่างในส่วนของพาเนลแสดงดังรูปที่ 2.25



รูปที่ 2.25 แสดงหน้าต่างพาเนล

ซึ่งเมื่อผู้ใช้ทำการลากตัวอุปกรณ์ลงไปวางที่พาเนลแล้ว ผู้ใช้สามารถสั่งให้ตัวอุปกรณ์ตัวนั้นเป็นตัวควบคุมหรือตัวแสดงผลก็ได้ แต่ต้องเป็นอย่างใดอย่างหนึ่งเท่านั้น วิธีการคือ คลิ๊กขวาที่ตัวอุปกรณ์ที่ผู้ใช้ต้องการจะเปลี่ยน จากนั้นจะเห็นคำสั่ง Change to Indicator หรือ Change to Control ดังรูปที่ 2.26 ซึ่งแล้วแต่ว่า ณ ตอนนั้นอุปกรณ์ที่ผู้ใช้ลากไปวางเป็นอะไร ถ้าเป็นประเภท Control เมื่อคลิ๊กขวาที่ตัวอุปกรณ์ก็จะเห็นเป็น Change to Indicator แต่ถ้าอุปกรณ์ ณ ตอนนั้นเป็น Indicator เมื่อคลิ๊กขวาที่ตัวอุปกรณ์ก็จะเห็นเป็น Change to Control ที่ได้กล่าวไปนั้นเป็นส่วนของพาเนลเท่านั้น

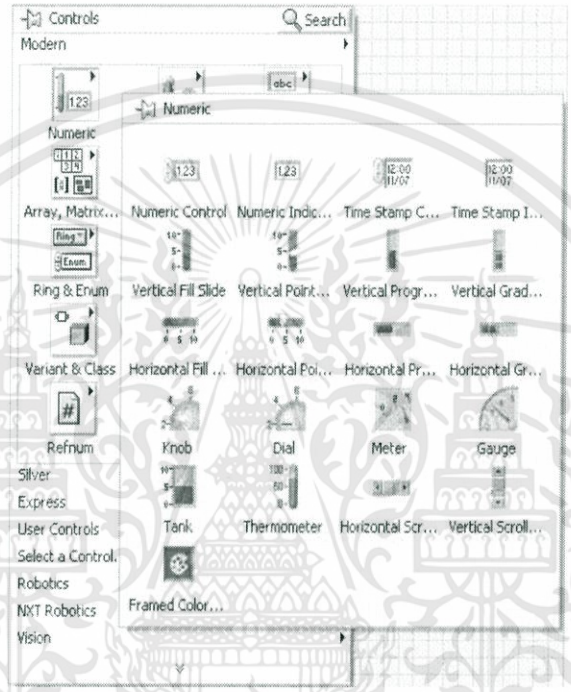


รูปที่ 2.26 แสดงหน้าต่างการเปลี่ยน Control และ Indicator ที่หน้าต่างพาเนล

ตัวอย่างเครื่องมือที่ผู้วิจัยใช้บ่อยในหน้าต่างพาเนล ได้แก่ Numeric และ Graph เป็นต้น

2.3.2 Numeric Controls and Indicator

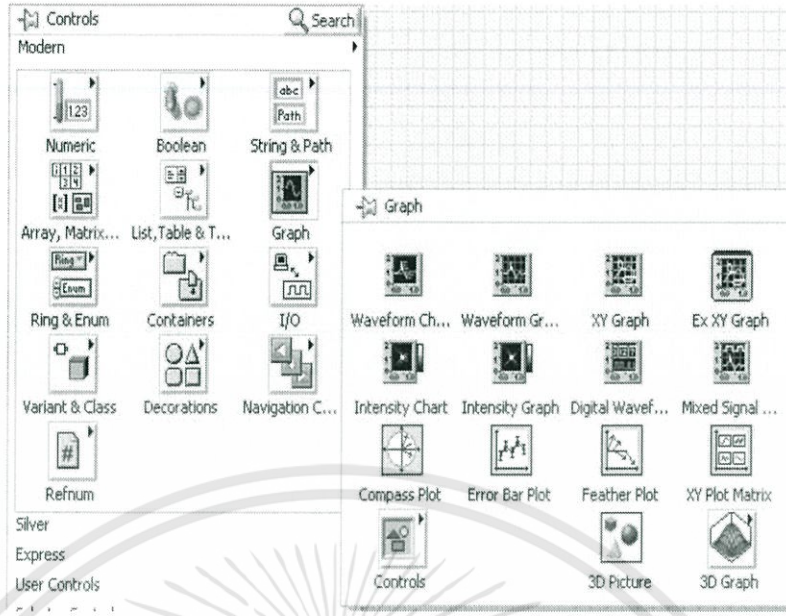
ใช้สำหรับเป็นตัวควบคุมและตัวแสดงผลที่มีลักษณะเป็นตัวเลขและกราฟฟิกเคลื่อนไหว ซึ่งสามารถเรียกใช้ในส่วนของหน้าต่าง Numeric โดยการคลิ๊กขวาที่พาเนลจะปรากฏ Control Palette คลิ๊กที่ Modern แล้ว Numeric จะปรากฏหน้าต่างของ Numeric ดังรูปที่ 2.27 ภายในก็จะมี Numeric Control, Numeric Indicator, Knob, Meter เป็นต้น ซึ่งค่าที่ควบคุมหรือค่าที่ใช้ในการแสดงผลล้วนแล้วแต่มีความสำคัญทั้งสิ้น เพราะค่าที่ผู้ใช้ควบคุมกระบวนการต้องมีการนำมาคำนวณหาค่าอื่นๆอีกในขั้นตอนต่อไป และค่าที่ใช้แสดงผลก็เช่นเดียวกัน



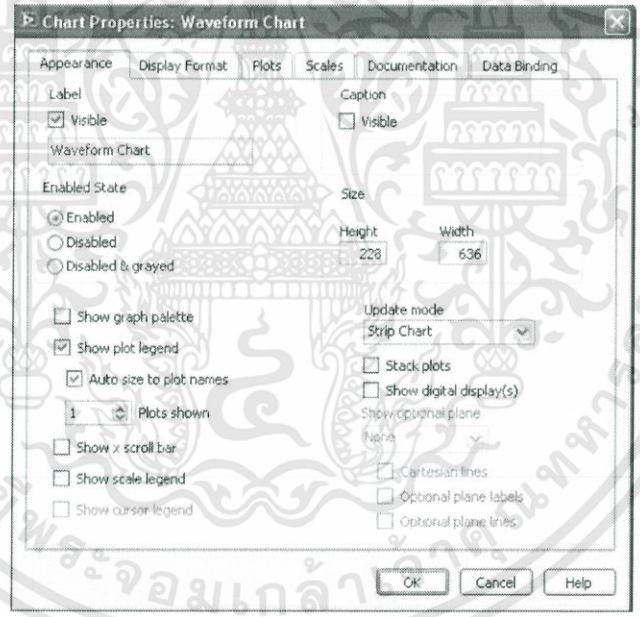
รูปที่ 2.27 แสดงหน้าต่างในส่วน Numeric Controls and Indicator

2.3.3 Graph

ใช้สำหรับเป็นตัวแสดงผลมีลักษณะแบบเส้นกราฟและสามารถแสดงเป็นตัวเลขได้อีกด้วยถ้าผู้ใช้ต้องการทราบ ซึ่งเส้นกราฟนั้นเป็นการพล๊อตแบบเรียลไทม์ ผู้ใช้สามารถปรับแต่งหรือตั้งค่าที่กราฟได้ตามการใช้งานของผู้ใช้ ซึ่งการปรับแต่งหรือตั้งค่ากราฟสามารถทำได้โดยการคลิ๊กขวาที่กราฟแล้วเลือก Properties ปรากฏหน้าต่างดังรูปที่ 2.29 ซึ่งผู้ใช้สามารถเรียกใช้ในส่วนของหน้าต่าง Graph โดยการคลิ๊กขวาที่พาเนลจะปรากฏ Control Palette คลิ๊กที่ Modern แล้ว Graph จะปรากฏหน้าต่างของ Graph ดังรูปที่ 2.28 ภายในก็จะมี Waveform Chart, Waveform Graph, XY Graph และอื่นๆ เป็นต้น



รูปที่ 2.28 แสดงหน้าต่างในส่วนของกราฟ

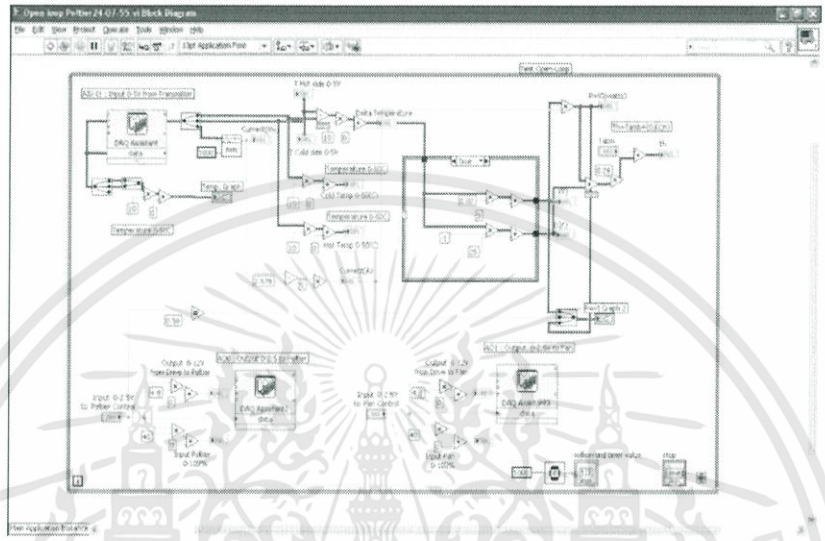


รูปที่ 2.29 แสดงหน้าต่างในส่วนของการปรับแต่งและตั้งค่ากราฟ

2.3.4 บล็อกไดอะแกรม

บล็อกไดอะแกรมเป็นส่วนที่ใช้สำหรับการเขียนโปรแกรม และแสดงค่าต่างๆของการทำงาน เช่น ถ้าผู้ใช้ตองการทราบค่าที่ออกมาจากบล็อกเครื่องมือต่างที่อยู่ในบล็อกไดอะแกรม ก็ให้ใช้คำสั่ง Highligh Execution เป็นต้น นอกจากจะแสดงค่าต่างๆแล้ว ยังแสดงบล็อกของเครื่องมือต่างที่ใช้ในการเขียนโปรแกรม ว่ามีการวาง การเชื่อมโยงบล็อกโดยใช้สายไฟต่อระหว่างบล็อกกันอย่างไร ในเบื้องต้นนี้ผู้ใช้อาจมองบล็อกไดอะแกรมว่าเป็น Data Flow Chart และตัวโปรแกรมหรือ code ของ LabVIEW ก็ได้ การเขียนบล็อกไดอะแกรม คือ การเขียน code ในภาษา G นั่นเอง หรือ เรียกว่า

การเขียนโปรแกรมด้วยภาษากราฟิก (G-Languages) ซึ่งก็เหมือนกับการเขียน code ในภาษา C นั่นเอง ความแตกต่างที่สำคัญระหว่าง code ในภาษา C กับ บล็อกไดอะแกรมใน LabVIEW ก็คือ บล็อกไดอะแกรมนั้นพร้อมที่จะทำการประมวลผลตลอดเวลา นั่นคือในระหว่างที่ผู้ใช้สร้างบล็อกไดอะแกรมอยู่ LabVIEW จะทำการตรวจสอบการทำงานของ VI อยู่ตลอดเวลา ซึ่งหน้าต่างในส่วนของบล็อกไดอะแกรมแสดงดังรูปที่ 2.30



รูปที่ 2.30 แสดงหน้าต่างบล็อกไดอะแกรม

ถ้าหากผู้ใช้ได้พิจารณาองค์ประกอบในบล็อกไดอะแกรม ผู้ใช้จะพบว่าในส่วนบล็อกไดอะแกรม จะมีส่วนประกอบที่สำคัญ 3 ส่วน คือ

- Terminal
- Node
- Wire

ทั้งสามส่วนจะมีหน้าที่หลัก คือ ทำหน้าที่ควบคุมการส่งผ่านข้อมูลหรือผู้ใช้อาจอีกอย่างเรียกว่าการไหลของข้อมูล (Data Flow) นอกจากนั้นยังทำหน้าที่กำหนดวิธีการประมวลผลของข้อมูลอีกด้วย ซึ่งจะขออธิบายทั้ง 3 ส่วน ดังต่อไปนี้

2.3.4.1 Terminal

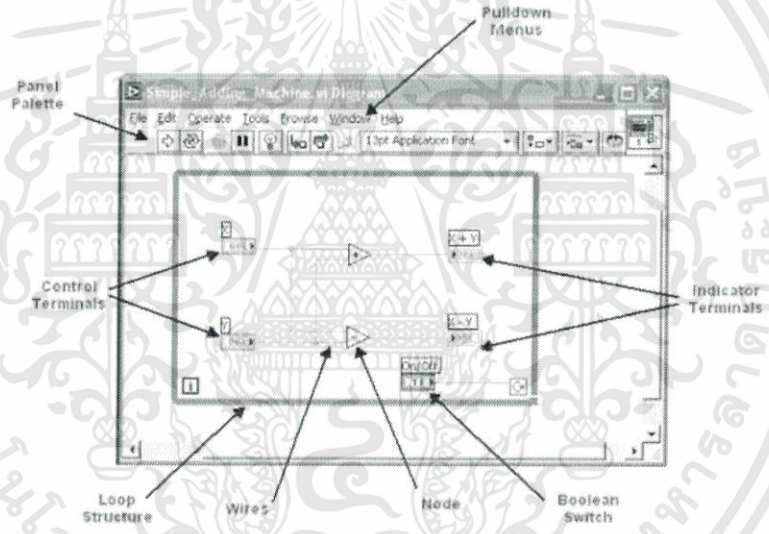
ทุกครั้งที่ผู้ใช้สร้างตัวควบคุมหรือตัวที่ใช้แสดงผลบนพาเนลในหน้าต่างของบล็อกไดอะแกรมจะปรากฏ Terminal ขึ้น Terminal ก็คล้ายกับสถานีของข้อมูล คือจะเป็นทั้งสถานีต้นทางของข้อมูล ถ้า Terminal นั้นเป็น Terminal ของตัวควบคุมและขณะเดียวกันจะเป็นทั้งสถานีปลายทางของข้อมูลถ้า Terminal นั้นเป็น Terminal ของตัวแสดงผล

สิ่งที่ควรเข้าใจอย่างหนึ่งก็คือ ถ้าหากผู้ใช้จะลบตัวควบคุมหรือตัวแสดงผลนั้นออกไปจากพาเนลแล้ว Terminal เหล่านี้ก็จะหายไปจากบล็อกไดอะแกรมเช่นกัน หรือถ้าหากผู้ใช้

ลบ Terminal ของตัวควบคุมหรือตัวแสดงผลออกจากบล็อกไดอะแกรม ตัวควบคุมหรือตัวแสดงผลเหล่านั้นก็จากหายไปจากพาเนลด้วยเช่นกัน

2.3.4.2 Node

Node เป็นคำที่ใช้เรียก object ที่ทำกรรมวิธีใดๆ เพื่อประมวลข้อมูลในบล็อกไดอะแกรมเช่นเดียวกับที่ผู้ใช้เขียน Flow Chart แล้วใช้สัญลักษณ์ต่างๆแทนวิธีการวิเคราะห์ข้อมูล เมื่อมีข้อมูลเข้าสู่ node สิ่งที่เกิดขึ้นภายใน node ก็จะขึ้นอยู่กับว่าจะกำหนดให้ข้อมูลที่ส่งเข้าไปนั้น จะมีการประมวลผลอย่างไร ซึ่งอาจจะเป็นการบวก ลบ คูณ หาร หารากยกกำลัง หรือเป็นประเภทการเปรียบเทียบข้อมูล ว่ามากหรือน้อยกว่า หรืออื่นๆ ซึ่งจะเป็นการประมวลผลทางคณิตศาสตร์ทั่วไป นอกเหนือจากนั้น node นี้จะมีส่วนที่เรียกว่า Function แบบต่างๆ ซึ่งจะเหมือนกับ Function สำเร็จรูปเช่น sine, cosine, log เป็นต้น ซึ่งก็จะเหมือนกับในภาษาที่เป็นตัวอักษรทั่วๆไป รูปที่ 2.31 แสดงถึงลักษณะของ Node และ Terminal และส่วนอื่นๆ ที่บรรจุอยู่ภายในบล็อกไดอะแกรมของ LabVIEW



รูปที่ 2.31 แสดงส่วนประกอบต่างๆของบล็อกไดอะแกรม

นอกเหนือจากการประมวลผลทางคณิตศาสตร์แล้ว ผู้ใช้ยังมี node ประเภท Structure หรือ Control Flow อีกด้วย (ในภาษาตัวอักษร Structure Command จะเป็น คำสั่งจำพวก IF...THEN, FOR..., WHILE... เป็นต้น)

2.3.4.3 Wires

ขณะที่ผู้ใช้มีที่มาของข้อมูล ส่วนประมวลหรือปรับแต่งข้อมูล และส่วนแสดงผลข้อมูลเรียบร้อยแล้ว ขั้นต่อไปคือผู้ใช้จะต้องสามารถควบคุมการส่งผ่านข้อมูลให้เป็นไปตามที่ผู้ใช้ต้องการ อุปกรณ์ที่ใช้ใน LabVIEW ก็คือ การต่อสาย หรือ Wires ซึ่งจะเป็นการเชื่อมการส่งข้อมูลระหว่าง terminal หรือ node ต่างๆที่มีในบล็อกไดอะแกรมนี้เข้าด้วยกัน โดย wires นี้จะเป็นการกำหนดเส้นทางของข้อมูลว่าเมื่อออกจาก terminal หนึ่งแล้ว จะกำหนดการไหลไปที่ node ใดบ้าง

มีลำดับเป็นอย่างไร และสุดท้ายจะให้แสดงผลที่ terminal ใดนั่นเอง ซึ่งการเชื่อมต่อสายนี้จะทำให้ผู้ใช้เข้าใจถึงหลักการของ Data Flow Programming ได้ดียิ่งขึ้น

เนื่องจากข้อมูลนั้นมีหลายแบบไม่ว่าจะเป็นเลขทศนิยม, เลขจำนวนจริง, ตัวอักษร หรือค่าจริง-เท็จ (Boolean) ดังนั้นเพื่อแสดงถึงความแตกต่างของข้อมูลแต่ละแบบ LabVIEW จึงได้กำหนดให้ลักษณะของ wires สำหรับข้อมูลแต่ละแบบมีลักษณะของเส้นและสีที่แตกต่างกัน โดยการใช้ Wiring Tool (รูปหลอดด้าย) ใน Tool Palette โดยที่

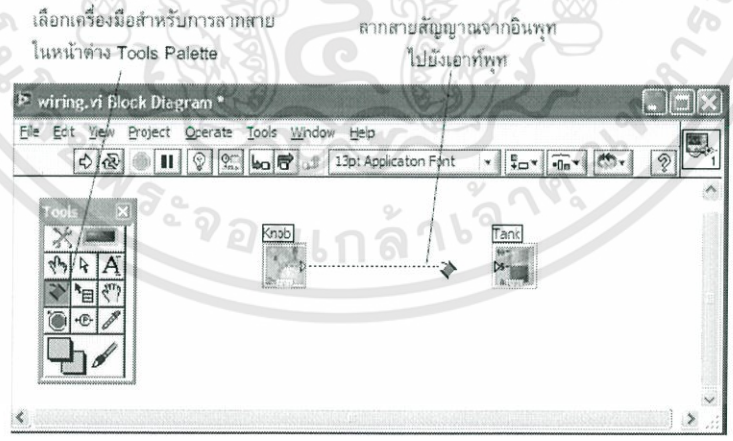
สีของ Control หรือ Indicator

1. สีน้ำเงิน แทนตัวเลขที่เป็นจำนวนเต็ม
2. สีส้ม แทนตัวเลขที่มีค่าทศนิยม
3. สีเขียว แทน Logic หรือ Boolean
4. สีชมพู แทนค่าของ String

รูปแบบของWiring

1. เส้นบางแทนค่าจำนวนเดียว
2. เส้นหนาแทนค่าของอาร์เรย์ ขนาด 1 มิติ
3. เส้นคู่แทนค่าของอาร์เรย์ ขนาด 2 มิติ

การ Wiring ทำโดยใช้ Wiring Tool เลื่อนไปให้ตรง Terminals แรกที่ต้องการเชื่อมต่อ สังเกตให้ Terminals กระพริบแล้วคลิ๊กเมาส์ จากนั้นให้ลากไปยัง Terminals ที่ต้องการเชื่อมต่อโดยจะ เกิดเส้นประสีดาลากตาม cursor ให้คลิ๊กเมาส์บน Terminals ที่สองในขณะที่ Terminals นั้นกำลังกระพริบ ข้อแตกต่างระหว่าง Control กับ Indicator บนบล็อกไดอะแกรม คือ Control จะมีขอบเป็นเส้นหนา ส่วน Indicator จะมีขอบเป็นเส้นบาง แสดงดังรูปที่ 2.32



รูปที่ 2.32 แสดงการเชื่อมต่อสัญญาณ

นอกจากนั้นข้อมูลแต่ละแบบดังกล่าวยังอาจมีลักษณะเป็น scalars, 1-D array, 2-D array ได้ซึ่งลักษณะของเส้นของข้อมูลแต่ละแบบก็จะแตกต่างออกไปอีก

ผู้ใช้จะเข้าใจหลักการของ Data Flow Programming ได้ในขณะที่ผู้ใช้ต่อสายระหว่าง terminal และ Node ต่างๆ เข้าด้วยกัน ซึ่งหลักการทำงานของ Data Flow Programming

จะต่างจากการเขียนโปรแกรมโดยใช้ตัวอักษร เพราะในโปรแกรมตัวอักษรคำสั่งจะถูกส่งเข้าสู่ส่วนประมวลผลทีละบรรทัด เพื่อการคำนวณตามลำดับบรรทัด แต่ใน Data Flow นั้น โปรแกรมจะคำนวณเมื่อมีข้อมูลส่งเข้ามาถึง Input ของ Node นั้นๆ ครบ และเมื่อข้อมูลส่งครบเข้าถึง Node นั้นแล้ว จะมีการประมวลผลและส่งค่าที่ได้ออกไปตามการต่อเชื่อมสาย การคำนวณจะเสร็จสิ้นในแต่ละรอบเมื่อข้อมูลส่งข้ามไปถึง Terminal สุดท้าย การประมวลผลไม่ได้เป็นไปตามลำดับการจัดวางคือไม่ได้ทำจากซ้ายไปขวาหรือบนลงล่าง แต่เป็นไปตามขั้นตอนการเดินทางของข้อมูลซึ่งในการเขียนโปรแกรมในเบื้องต้นผู้ที่คุ้นเคยกับภาษาตัวหนังสือโดยทั่วไปอาจจะต้องใช้เวลาสักระยะเพื่อที่จะทำความเข้าใจการทำงานของ LabVIEW

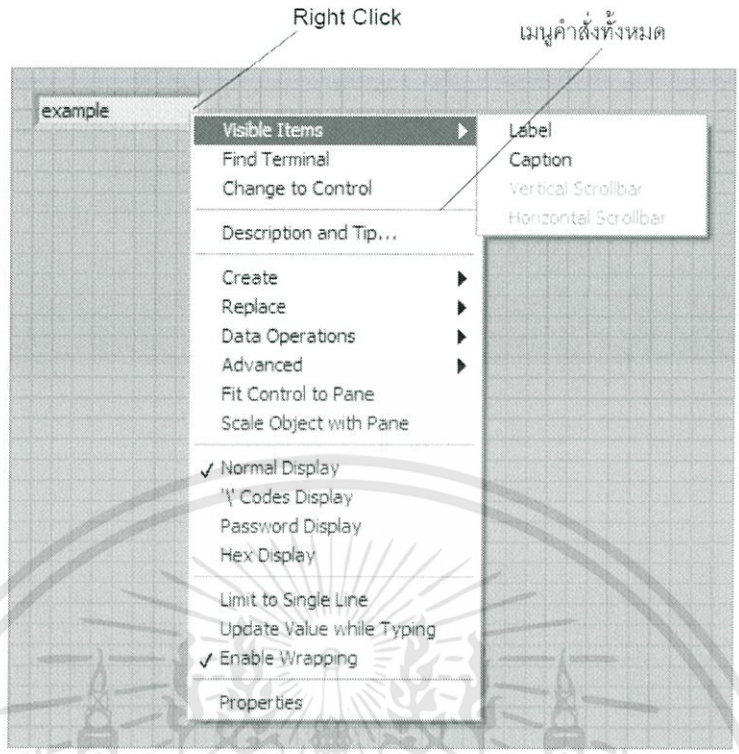
2.3.5 ไอคอนและจุดเชื่อมต่อ

ไอคอน (Icon) และจุดเชื่อมต่อ (Connector) ในโปรแกรม LabVIEW แล้วก็คืออุปกรณ์ตัว เดียวกันเพียงแต่อุปกรณ์นั้นสามารถปรับเปลี่ยนคุณสมบัติให้เหมาะสมกับการใช้งานซึ่งโปรแกรมที่เราเห็น สัญลักษณ์ของอุปกรณ์โดยทั่วไปเราจะเรียกสัญลักษณ์เหล่านั้นว่า ไอคอน (Icon) เมื่อเราต้องการเชื่อมต่อ สายสัญญาณของไอคอนเราจำเป็นต้องรู้ว่าจุดเชื่อมต่อ (Connector) ของไอคอนนั้นอยู่ที่ตำแหน่งใดของไอคอนและมีคุณสมบัติอย่างไรเราจึงจำเป็นต้องเปลี่ยนคุณสมบัติของไอคอนให้เป็นคอนเน็คเตอร์ โดยคลิ๊กเมาส์ทางขวาที่รูปไอคอนแล้วเลือก Visible Item» Terminal ไอคอนก็จะเปลี่ยนไปเป็นเทอร์มินอลที่แสดงจุดเชื่อมต่ออย่างชัดเจน

2.3.6 เมนูคำสั่ง

2.3.6.1 เมนูคำสั่งแบบ Shortcut

การเรียกใช้งานชุดคำสั่งในโปรแกรม LabVIEW มีอยู่สองวิธีคือ เรียกในเมนูบาร์ (Menu bar) และเรียกจากเมนูคำสั่งแบบย่อ โดยวิธีการเรียกใช้งานชุดคำสั่งแบบย่อทำได้โดยการลากเมาส์ไปวาง บริเวณอ๊อปเจ็คบนหน้าต่างฟร้อนท์พาเนลหรือหน้าต่างบล็อกไดอะแกรมแล้วคลิ๊กเมาส์ทางด้านขวาซึ่งจะ ปรากฏชุดคำสั่งย่อขึ้นมาภายในคำสั่งก็จะมีชุดคำสั่งย่อย การเรียกใช้งานนั้นทำได้โดยการลากเมาส์ไปชี้ ตรงคำสั่งที่ต้องการซึ่งจะเกิดแถบสีน้ำเงินทึบบริเวณที่เมาส์ชี้อยู่ถ้าต้องการใช้คำสั่งให้คลิ๊กเมาส์บริเวณ คำสั่งที่ต้องการ ดังรูปที่ 3.33



รูปที่ 2.33 การใช้โครงสร้างของคำสั่งแบบ Shortcut

2.3.6.2 เมนูคำสั่งแบบ Pull Down

เมนูคำสั่งแบบ Pull Down ในหน้าต่างพาเนลและไดอะแกรมจะมีชุดคำสั่งที่เมนูบาร์อยู่ 8 คำสั่ง คือ File, Edit, View, Project, Operate, Tools, Window และ Help

File Edit View Project Operate Tools Window Help

รูปที่ 2.34 แสดงหน้าต่างคำสั่งแบบ Pull Down

คำสั่ง File ในคำสั่ง File จะประกอบไปด้วยคำสั่งย่อยสำหรับการจัดการเกี่ยวกับการการสร้าง VI การเปิดไฟล์ VI การปิด VI สำหรับหน้าต่างที่ผู้ใช้เปิดอยู่ การปิด VI ทุกหน้าต่างที่กำลังเปิดอยู่ การบันทึก การสร้าง Project การเปิดไฟล์ Project การตั้งค่าหน้ากระดาษ การปริ้นหน้าพาเนลหรือไดอะแกรม ดูคุณสมบัติของ VI ไฟล์ที่เปิดล่าสุดและสุดท้ายออกจากโปรแกรม อีกทั้งยังมีคำสั่งลัดที่สั่งเปิดจากคีย์บอร์ด เช่น Ctrl+N สำหรับสร้างหน้าต่าง VI ขึ้นมาใหม่ Ctrl+O เพื่อเปิดไฟล์ VI Ctrl+S เพื่อบันทึก VI Ctrl+P เพื่อปริ้นหน้าพาเนลหรือไดอะแกรมของ VI Ctrl+Q เพื่อปิด VI เป็นต้น

คำสั่ง Edit ในคำสั่ง Edit จะประกอบไปด้วยคำสั่งย่อยสำหรับการจัดการเกี่ยวกับการยกเลิก(Ctrl+Z) การทำซ้ำ(Ctrl+Shift+Z) การตัด(Ctrl+X) การคัดลอก(Ctrl+C)การวาง(Ctrl+V) การเลือกวัตถุทั้งหมด(Ctrl+A) การทำให้ค่าที่ผู้ใช้ต้องการนั้นเป็นค่าเริ่มต้น สร้าง SubVI การแทรกรูปภาพ การเอาสายที่ Broken ออก(Ctrl+B) คำสั่งย่อยในคำสั่ง Edit ที่ถูกใช้บ่อยครั้งที่สุดคือ คำสั่ง Undo Data Chang และ Redo ซึ่งคำสั่งนี้จะช่วยผู้ใช้ในเรื่องของการแก้ไข

โปรแกรมหากการแก้ไขผิดพลาดหรือไม่ต้องการ ผู้ใช้สามารถกลับไปยังจุดที่ผู้ใช้เริ่มต้นแก้ไขได้โดยใช้คำสั่งนี้

คำสั่ง View ในคำสั่ง View จะประกอบไปด้วยชุดแสดงผลของโปรแกรม เช่น Tools Palette, Controls palette และ Function Palette และยังมีส่วนแสดงผลของค่าความผิดพลาด

คำสั่ง Project ในคำสั่ง Project ประกอบไปด้วยชุดคำสั่งพื้นฐานของโปรแกรม ใช้สำหรับจัดการกับโปรแกรมที่เปิดพร้อมกันหลายๆ โปรแกรม ให้เปิด ปิด บันทึก และสร้างโปรแกรม

คำสั่ง Operate ใช้สำหรับสั่งให้โปรแกรมทำงาน(Ctrl+R) หรือหยุดทำงาน (Ctrl+.) เปลี่ยนหรือเซ็ทค่ามาตรฐานของ VI และสามารถเปลี่ยนโหมดการทำงานจากโหมดรันไปเป็นโหมดแก้ไขได้(Ctrl+M)

คำสั่ง Tools สำหรับเมนูนี้ใช้เกี่ยวกับการติดต่อระหว่างโปรแกรมกับอุปกรณ์ภายนอก เปรียบเทียบ VI เช่น Instrument Real-Time Module FPGA Module DSC Module เป็นต้น การสร้างแอบพลิเคชั่นต่างๆ และติดต่อกับเว็ปไซด์ของ NI ด้วย

คำสั่ง Window ใช้สำหรับเลือกการเปิดหน้าต่างพาเนลและไดอะแกรมสลับกันหรือเปิดทั้ง 2 หน้าจอพร้อมกันก็ได้ เช่น Show Block Diagram(Ctrl+E) แสดงหน้าต่างไดอะแกรม Tile Left and Right(Ctrl+T) แสดงหน้าต่างทั้งพาเนลและไดอะแกรมซ้ายและขวา เป็นต้น

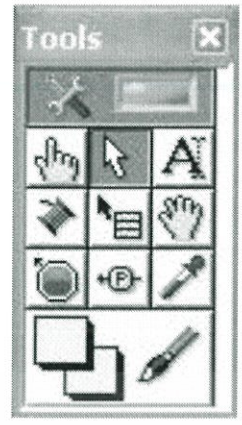
คำสั่ง Help ใช้ในการแสดงรายละเอียดของ Tool แต่ละตัวและใช้สำหรับสืบค้นข้อมูลต่างๆ ภายในโปรแกรม LabVIEW เช่น ตัวอย่างโปรแกรม เป็นต้น

2.3.7 พาเลท

พาเลทเป็นหน้าต่างย่อยที่มีส่วนประกอบของอ๊อบเจ็คที่ใช้สำหรับการเขียน VI การนำไปใช้งานนั้นทำได้ง่ายเพียงแค่คลิ๊กเมาส์ทางด้านขวาบริเวณที่ว่างของหน้าต่างพาเนลหรือบล็อกไดอะแกรม ซึ่งจะปรากฏพาเลทขึ้นมาจากนั้นใช้เมาส์คลิ๊กที่อ๊อปเจ็คที่ต้องการแล้วลากไปวางบนหน้าต่างพาเนลหรือหน้าต่างไดอะแกรม พาเลทมีองค์ประกอบหลักๆ อยู่ 3 ส่วนคือ Tools, Control และ Function

2.3.7.1 Tools Palette

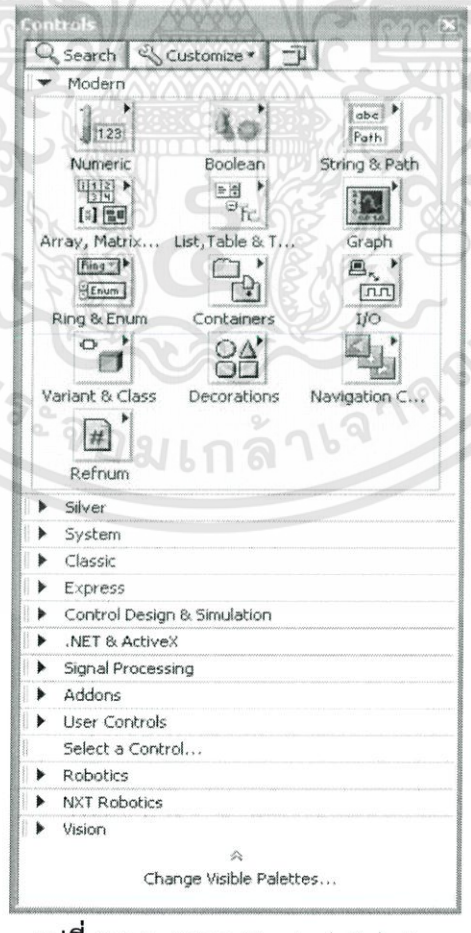
ทูลพาเลทจะเป็นตัวบอกหรือกำหนดสถานะของเมาส์ว่าทำงานอยู่ในโหมดใด การเรียกใช้พาเลทนี้ทำได้โดยการเลือกที่เมนู View แล้วเลือกคำสั่งย่อย Tools Palette แล้วจะปรากฏ Tools Palette ขึ้นมาดังรูปที่ 2.35 การใช้งานนั้นเพียงแค่เราใช้เมาส์ไปคลิ๊กที่สัญลักษณ์ใน Tools Palette เคอร์เซอร์ของเมาส์ก็จะเปลี่ยนไปตามลักษณ์ที่เราเลือก



รูปที่ 2.35 แสดง Tools Palette

2.3.7.2 Control Palette

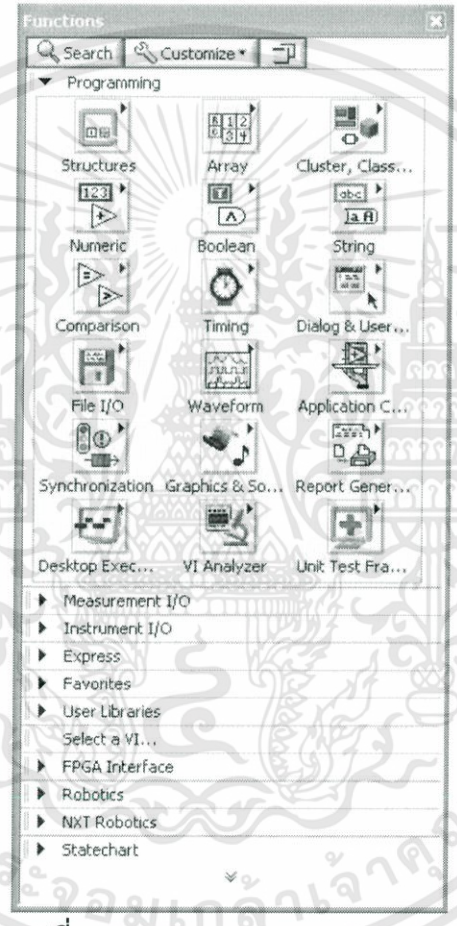
มีส่วนประกอบของอุปกรณ์ที่เป็นตัวควบคุมและตัวแสดงผลที่ใช้สำหรับสร้าง VI โดยภายใน Control Palette ก็จะมีพาเลทย่อยๆ อีกเพราะว่าอุปกรณ์ใน Control Palette ก็จะมีอุปกรณ์ย่อยอีก แต่อย่างไรก็ตามเราสามารถเรียกดูอุปกรณ์ทั้งหมดที่อยู่ใน Control Palette ได้โดยคลิ๊กเมาส์ขวา ซึ่งจะรวมเอาอุปกรณ์ทั้งหมดไว้ในนี้ นอกจากนี้แล้วเรายังสามารถเปิดใช้งาน Control Palette ได้อีกทางหนึ่งคือเลือกที่เมนู View แล้วเลือก Control Palette ก็สามารถเลือกใช้ Control Palette ได้เช่นกัน ซึ่ง Control Palette แสดงดังรูปที่ 2.36



รูปที่ 2.36 แสดง Control Palette

2.3.7.3 Function Palette

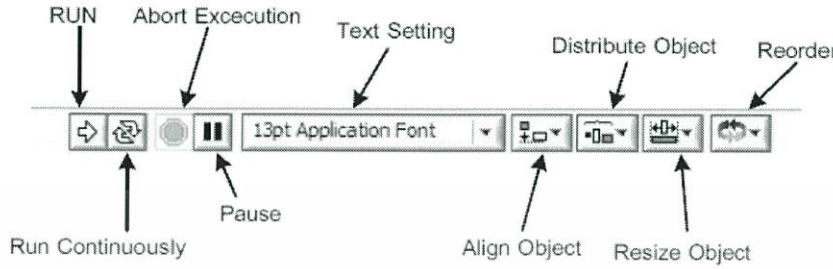
การใช้งานโดยทั่วไปจะมีความคล้ายกันกับ Control Palette ซึ่งภายในประกอบไปด้วยอุปกรณ์ในการสร้าง VI ตัวอย่างอุปกณ์หลักๆ ใน Function Palette เช่น For Loops, While Loops และ Formula Nodes สามารถเรียกใช้งานได้ 2 วิธีเช่นเดียวกันกับการเรียกใช้งาน Control Palette คือ เลือกที่เมนู View ในหน้าต่างของบล็อกไดอะแกรม แล้วเลือก Function Palette หรือคลิ๊กเมาส์ทางด้านขวาบริเวณพื้นที่ว่างในหน้าต่างไดอะแกรม ก็จะปรากฏ Function Palette เช่นกัน ซึ่ง Function Palette แสดงดังรูปที่ 2.37



รูปที่ 2.37 แสดง Function Palette

2.3.8 เครื่องมือ (Tool)

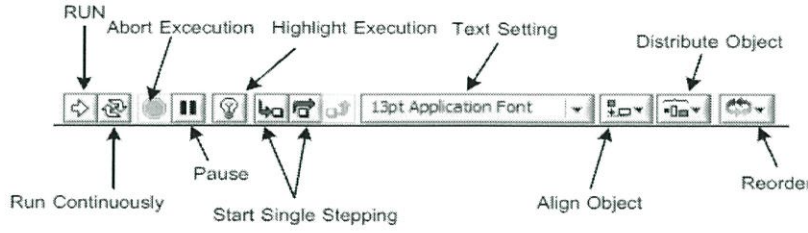
2.3.8.1 เครื่องมือของหน้าต่างพาเนล



รูปที่ 2.38 แสดงทูลบาร์ในหน้าต่างพาเนล

RUN	ใช้สำหรับรันโปรแกรม
RUN Continuously	ใช้สำหรับรันโปรแกรมอย่างต่อเนื่อง
Abort Execution	หรือเรียกอีกอย่างหนึ่งว่าปุ่ม Stop ใช้สำหรับหยุดการรันโปรแกรม
Pause	ใช้สำหรับหยุดการรันโปรแกรมชั่วคราวเพื่อตรวจสอบการทำงานของโปรแกรม เมื่อต้องการรันโปรแกรมต่อให้กดปุ่มนี้ซ้ำอีกครั้งโปรแกรมก็จะรันต่อไป
Text setting	ใช้สำหรับจัดการกับตัวอักษรทั้งหมดไม่ว่าจะเป็น ฟร้อนท์ ขนาด รูปแบบตัวอักษรรวมไปถึงการกำหนดสีให้กับตัวอักษรด้วย
Align Object	ใช้จัดObject ต่างๆ ที่เราใช้เขียนโปรแกรมให้อยู่ในแนวเดียวกันทั้งแนวตั้งและ แนวนอนเพื่อความเป็นระเบียบ
Distribute Object	ใช้สำหรับการกำหนดระยะห่างระหว่าง Objects อัตโนมัติ
Resize Object	ใช้สำหรับปรับขนาดของ Object
Reorder	ใช้สำหรับกำหนดลำดับก่อนหลังสำหรับการวางซ้อนทับกันของ Object
Context Help	ใช้แสดงคุณสมบัติหรือรายละเอียดของ Object แต่ละตัว

2.3.8.2 เครื่องมือของหน้าต่างบล็อกไดอะแกรม

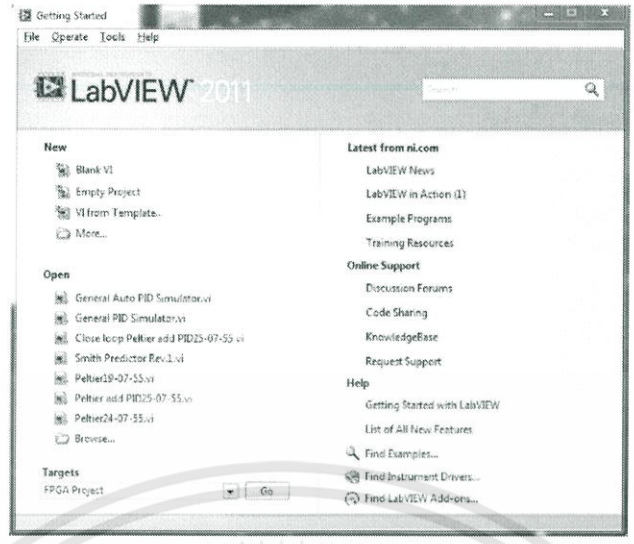


รูปที่ 2.39 แสดงทูลบาร์ในหน้าต่างบล็อกไดอะแกรม

RUN	ใช้สำหรับรันโปรแกรม
RUN Continuously	ใช้สำหรับรันโปรแกรมอย่างต่อเนื่อง
Abort Execution	หรือเรียกอีกอย่างหนึ่งว่าปุ่ม Stop ใช้สำหรับหยุดการรันโปรแกรม
Pause	ใช้สำหรับหยุดการรันโปรแกรมชั่วคราวเพื่อตรวจสอบการทำงานของโปรแกรม เมื่อต้องการรันโปรแกรมต่อให้กดปุ่มนี้ซ้ำอีกครั้งโปรแกรมก็จะรันต่อไป
Highlight Execution	แสดงลำดับขั้นการทำงานของโปรแกรม โดยแสดงให้เห็นทิศทางการไหลของสัญญาณ
Start Single Step	ใช้สำหรับสั่งให้โปรแกรมทำงานทีละคำสั่ง
Text Setting	ใช้สำหรับจัดการกับตัวอักษรทั้งหมดไม่ว่าจะเป็น ฟร้อนท์ ขนาด รูปแบบตัวอักษร รวมไปถึงการกำหนดสีให้กับตัวอักษรด้วย
Align Object	ใช้จัด Object ต่าง ๆ ที่เราใช้เขียนโปรแกรมให้อยู่ในแนวเดียวกันทั้งแนวตั้งและแนวนอนเพื่อความเป็นระเบียบ
Distribute Object	ใช้สำหรับการกำหนดระยะห่างระหว่าง Objects อัตโนมัติ
Reorder	ใช้สำหรับกำหนดลำดับก่อนหลังสำหรับการวางซ้อนทับกันของ Object
Context Help	ใช้แสดงคุณสมบัติหรือรายละเอียดของ Object แต่ละตัว

2.3.9 การเปิดและการบันทึกไฟล์

การเปิดใช้งานและการบันทึกโปรแกรม LabVIEW มีด้วยกันหลายวิธีด้วยกัน ซึ่งวิธีการเรียกใช้งานและการบันทึกแต่ละแบบก็มีรูปแบบที่แตกต่างกันออกไป ซึ่งผู้ใช้งานสามารถเรียกใช้งานได้ตามความเหมาะสม เพราะไม่ว่าจะเปิดด้วยวิธีใดก็สามารถใช้งานได้เช่นเดียวกัน



รูปที่ 2.40 แสดงการเริ่มต้นใช้งานโปรแกรม

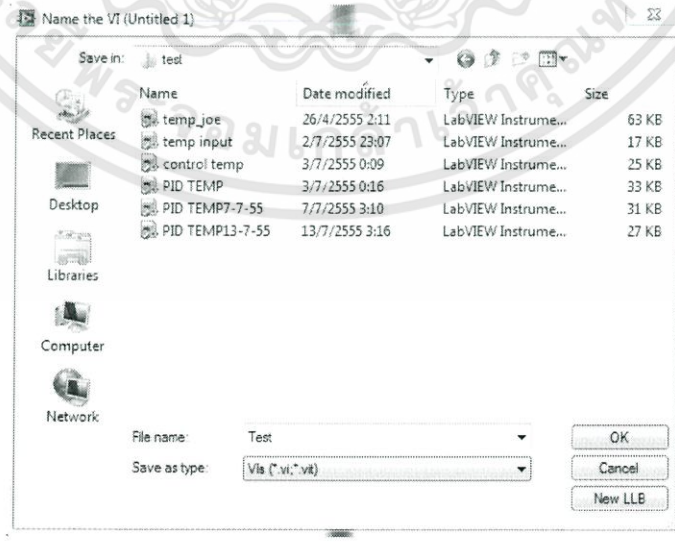
2.3.9.1 การเปิดไฟล์

การเปิดไฟล์ใหม่ จากรูปที่ 2.40 คลิ๊กเลือกไปที่ Blank VI ก็จะได้หน้าต่างพาเนลและหน้าต่างไดอะแกรมใหม่ขึ้นมา

การเรียกโปรแกรมที่สร้างไว้แล้วมาใช้งานทำได้โดยเปิดโปรแกรม LabVIEW แล้วเลือกที่เมนู File »Open จะปรากฏไดอะล็อกบล็อกที่บันทึกโปรแกรมไว้แล้วทำการเลือกไฟล์ที่ต้องการ ก็จะได้โปรแกรมที่บันทึกไว้มาใช้งานได้

2.3.9.2 การบันทึกไฟล์

การบันทึกไฟล์ VI สามารถทำได้โดยเลือก Save, Save As, Save All, หรือ Save with Option จากเมนูไฟล์ หลังจากนั้นจะปรากฏหน้าต่างสำหรับบันทึกขึ้นมา แล้วทำการเลือก โฟลเดอร์ ที่ต้องการบันทึกไฟล์ ตั้งชื่อไฟล์แล้วทำการบันทึก ไฟล์ที่บันทึกไว้จะมีนามสกุล *.vi

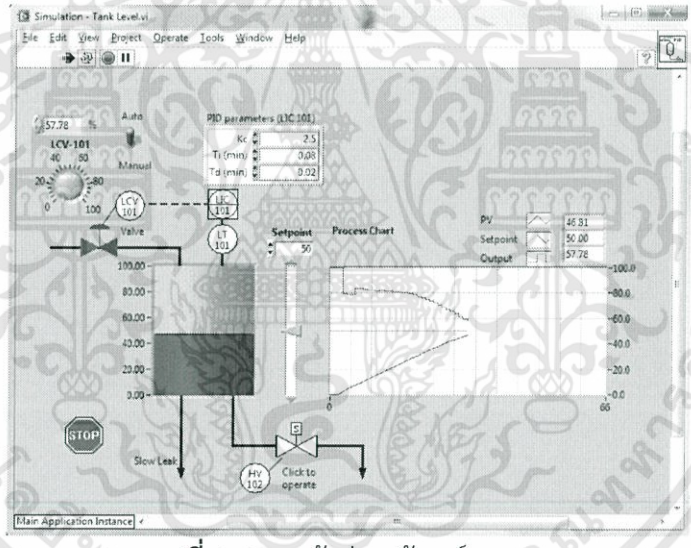


รูปที่ 2.41 แสดงไดอะล็อกบล็อกสำหรับเปิดใช้งานโปรแกรม LabVIEW

2.4 พื้นฐานการเขียนโปรแกรมภาษากราฟฟิก

2.4.1 ฟร้อนท์พาเนล

หน้าต่างฟร้อนท์พาเนลสำหรับเขียนโปรแกรม LabVIEW นั้นจะเป็นหน้าต่างที่ใช้ติดต่อกับผู้ใช้งาน ซึ่งในหน้าต่างพาเนลจะประกอบไปด้วยอุปกรณ์อยู่ 2 ส่วน คือ อุปกรณ์สำหรับการควบคุม (Control) ยกตัวอย่างเช่น knobs, switches และอุปกรณ์อีกอย่างหนึ่งก็คืออุปกรณ์สำหรับแสดงผล (Indicator) เช่น LED, Text Setting, Graph และ Chart เป็นต้น ส่วนอุปกรณ์อื่นๆ ที่ไม่ใช่อุปกรณ์ควบคุมและแสดงผลจะแสดงไอคอนอยู่ในหน้าต่างบล็อกไดอะแกรม สำหรับอุปกรณ์ควบคุมในภาษากราฟิกจะหมายถึง “อินพุท” และอุปกรณ์แสดงผลจะหมายถึง “เอาท์พุท” การวางอุปกรณ์ควบคุมและอุปกรณ์แสดงผลลงไปบนหน้าต่างฟร้อนท์พาเนลทำได้โดยการเลือกอุปกรณ์ที่ต้องการบนหน้าต่าง Control Palette แล้วใช้เมาส์ลากไปวางบนหน้าต่างพาเนล ซึ่งการเรียกหน้าต่าง Control Palette ทำได้โดยการคลิ๊กเมาส์ทางด้านขวาบริเวณพื้นที่ว่างของหน้าต่างพาเนลก็จะปรากฏหน้าต่าง Control Palette ที่ประกอบไปด้วยอุปกรณ์ควบคุมและอุปกรณ์แสดงผลขึ้นมา การเรียกใช้งานหน้าต่าง Control Palette อีกวิธีหนึ่งก็คือ เลือกที่เมนู Window ของแท็บเมนูบาร์ แล้วเลือก Show Control Palette จะ ปรากฏหน้าต่างของ Controls Palette ขึ้นมาให้เลือกใช้งาน เช่นเดียวกัน



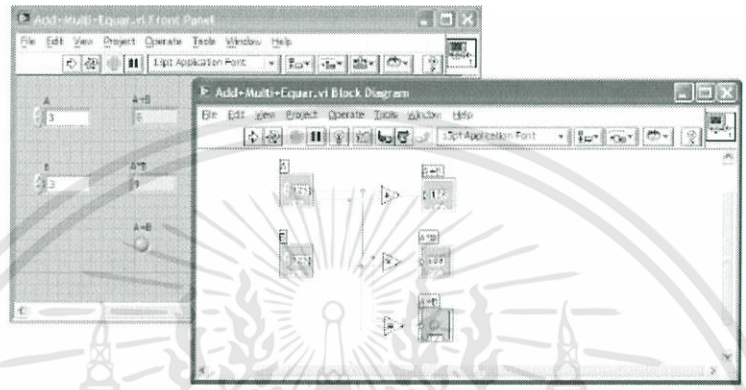
รูปที่ 2.42 หน้าต่างฟร้อนท์พาเนล

2.4.2 บล็อกไดอะแกรม

ในหน้าต่างบล็อกไดอะแกรมจะประกอบไปด้วยอุปกรณ์ที่ใช้ในการสร้างโปรแกรม ซึ่งถ้า เปรียบเทียบการภาษาอื่นๆ แล้วอุปกรณ์ที่วางลงไปบนหน้าต่างบล็อกไดอะแกรมจะถือว่าเป็น Source Code การทำให้อุปกรณ์ต่างๆ ที่อยู่ในหน้าต่างของบล็อกไดอะแกรมทำงานสัมพันธ์กันและเป็นไปตามที่ ผู้เขียนต้องการนั้นสามารถทำได้ด้วยการลากสายสัญญาณจากจุดเชื่อมต่อของอุปกรณ์แต่ละตัวให้ถึงกัน และกำหนดค่าหรือคุณสมบัติของอุปกรณ์แต่ละตัวก็จะทำให้โปรแกรมทำงานตามที่ผู้เขียนต้องการ คุณสมบัติของอุปกรณ์ที่อยู่บนหน้าต่างบล็อกไดอะแกรมมีอยู่ 3 คุณสมบัติ คือ Node, Terminal และ Wires

2.4.2.1 Nodes

เป็นตัวกระทำให้โปรแกรมทำงานตามที่ผู้เขียนต้องการถ้าเปรียบเทียบกับโปรแกรมทั่วไป Nodes ก็เปรียบเสมือน statement, functions, และ subroutines โหนดในโปรแกรม LabVIEW มี 3 ชนิด คือ Function, Sub VI โหนด และ Structure โหนดแบบ Function จะสร้างขึ้นมาเพื่อให้สามารถทำงานขั้นพื้นฐานได้ เช่น การบวกเลข การคูณเลข เป็นต้น ดังการบวกและการคูณในรูปที่ 2.43 ก็เป็นฟังก์ชั่นโหนดแบบหนึ่งเช่นกัน



รูปที่ 2.43 หน้าต่างบล็อกไดอะแกรม

2.4.2.2 Terminal

เป็นทางผ่านของข้อมูลระหว่างบล็อกไดอะแกรมกับฟร้อนท์พาเนล และระหว่างโหนดแต่ละโหนดในบล็อกไดอะแกรม เทอร์มินอลเปรียบได้กับค่าพารามิเตอร์และค่าคงที่ในโปรแกรมภาษาอื่น เทอร์มินอลแต่ละชนิดก็มีความแตกต่างกันซึ่งในโปรแกรม LabVIEW มีเทอร์มินอลอยู่ 4 ชนิดคือ Control and Indicator Terminals, node Terminal, Constants, และเทอร์มินอลพิเศษ ซึ่งเทอร์มินอลแต่ละตัวสามารถลากสายสัญญาณเพื่อเป็นทางผ่านของข้อมูล สำหรับตัวอย่างในกรณีของ Control and Indicator Terminals เมื่อมีข้อมูลแบบตัวเลขผ่านเข้ายังบล็อกไดอะแกรมโดยผ่านช่องทาง Control Terminals แล้วทำการประมวลผลในหน้าต่างบล็อกไดอะแกรมเมื่อโปรแกรมทำการประมวลผลเสร็จเรียบร้อยก็จะส่งข้อมูลจากหน้าต่างบล็อกไดอะแกรมไปยังหน้าต่าง ฟร้อนท์พาเนล โดยผ่านช่องทาง Indicator Terminal

2.4.2.3 Wires

เป็นทางผ่านของข้อมูลระหว่าง Terminal โดยการลากสายสัญญาณทำได้โดยการเลือกเครื่องมือที่ชื่อว่า connect wire ที่หน้าต่าง Tool Palette ที่มีลักษณะคล้ายกับหลอดด้าย เมาส์จะเปลี่ยนเป็นรูปหลอดด้ายจากนั้นลากไปวางตรงจุดต่อของเทอร์มินอลแรกที่ต้องการลากคลิ๊กเมาส์หนึ่งครั้งแล้วลากไปยังจุดต่อของเทอร์มินอลปลายทางที่ต้องการคลิ๊กเมาส์อีกหนึ่งครั้งก็เสร็จการเชื่อมต่อสัญญาณ

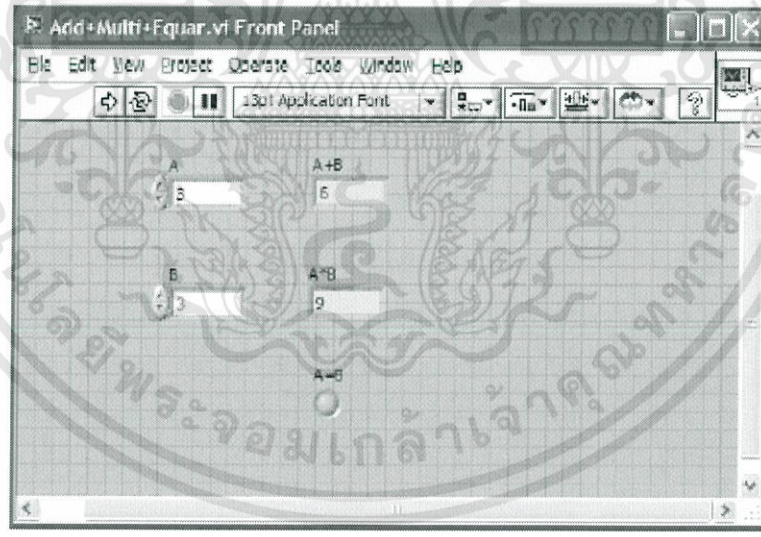
2.4.2.4 การสร้าง VI

ในส่วนของหัวข้อนี้จะกล่าวถึงขั้นตอนและวิธีการสร้าง VI โดยมีวัตถุประสงค์เพื่อให้สามารถบวกเลขและแสดงผลลัพธ์ได้, ให้สามารถคูณเลขและแสดงผลลัพธ์ได้ และเปรียบเทียบกันระหว่างอินพุทสองอินพุทถ้ามีค่าเท่ากันให้หลอด LED ติด

เริ่มต้นพิจารณาที่หน้าต่างฟร้อนท์พาเนลในรูปที่ 2.43 ในหน้าต่างฟร้อนท์พาเนลจะมีอินพุทที่เป็นดิจิตอล 2 อินพุทกำหนดให้เป็น A และ B และมีตัวแสดงผลแบบดิจิตอล 2 ตัว กำหนดให้เป็น $A+B$ และ $A\times B$ และมีตัวแสดงผลแบบ LED ซึ่งจะติดเมื่ออินพุท A กับ B มีค่าเท่ากันโดย ขั้นตอน และวิธีการสร้าง VI มีขั้นตอนดังต่อไปนี้

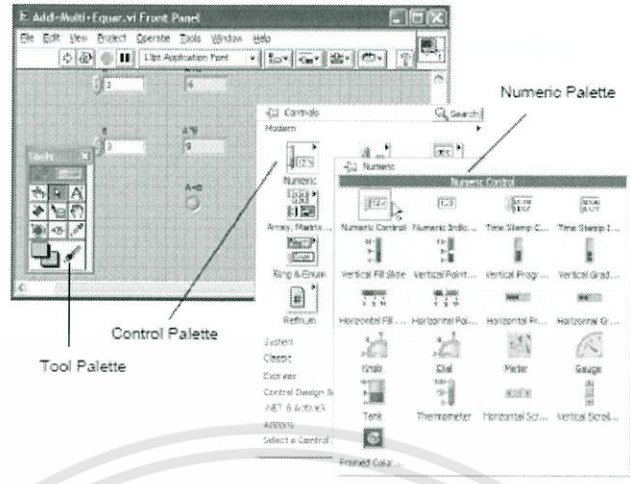
1.สร้างหน้าต่างฟร้อนท์พาเนลขึ้นมาใหม่โดยเลือกที่ file เลือก New

2.สร้าง Numerical Control และ Indicator สำหรับป้อนอินพุทและแสดงผลอย่างละ 2 ตัวและสร้างเครื่องมือในการบวก การคูณ และเปรียบเทียบค่าเลือก Numerical Control จาก Numerical Control Palette ซึ่งเป็นพา-เลทย่อย ของ Controls Palette ใช้เมาส์คลิ๊กที่ Numeric Control แล้วลากไปวางหน้าต่างฟร้อนท์พาเนล ชื่อของอุปกรณ์จะมีชื่อว่า Numeric ซึ่งเราสามารถที่จะเปลี่ยนได้โดยดับเบิ้ลคลิ๊กเมาส์ บริเวณตัวอักษรให้มีแถบสีดำทึบปรากฏขึ้น จากนั้นทำการเปลี่ยนชื่อได้ตามที่เราต้องการ ถ้า Control หรือ Indicator ไม่ปรากฏแถบลาเบล ให้คลิ๊กเมาส์ทางด้านขวาบริเวณอุปกรณ์เลือกที่ Visible Item แล้ว เลือกที่ Label แถบลาเบลก็จะปรากฏขึ้นมา



รูปที่ 2.44 แสดงการสร้าง VI

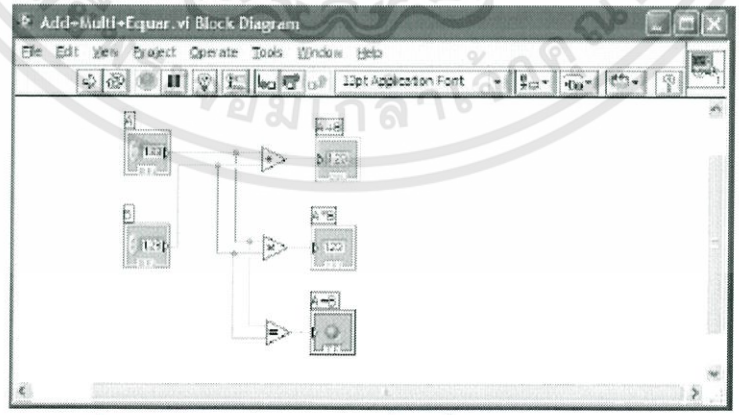
3.สร้างหลอด LED ซึ่งเราต้องการให้หลอด LED ติดเมื่อค่าของอินพุททั้งสองมีค่าเท่ากันและดับเมื่ออินพุททั้งสองต่างกันเลือก Round LED จาก LED Palette ซึ่งเป็นพาเลทย่อยของ Controls Palette คลิ๊กเมาส์ที่ Round LED แล้วลากไปวางบริเวณที่ว่างบนหน้าต่างฟร้อนท์พาเนล การตั้งชื่อสามารถทำได้เช่นเดียวกันกับการตั้งชื่อของ Numeric Control



รูปที่ 2.45 แสดงการเลือกอุปกรณ์ไปวางที่ฟร้อนท์พาเนล

ในการสร้างโปรแกรม VI นั้นเมื่อเราลากอุปกรณ์ควบคุมและแสดงผลมาวางบนหน้าต่าง ฟร้อนท์พาเนล ในเวลาเดียวกันนั้นก็จะปรากฏเทอร์มินอลของอุปกรณ์นั้นๆ ขึ้นที่หน้าต่าง บล็อกไดอะแกรมซึ่งเทอร์มินอลที่เกิดขึ้นนี้เองจะเป็นตัวที่ถูกนำมาลากสายสัญญาณเชื่อมต่อกันให้โปรแกรมทำงานเป็นไปตามเงื่อนไขที่ผู้เขียนโปรแกรมต้องการ

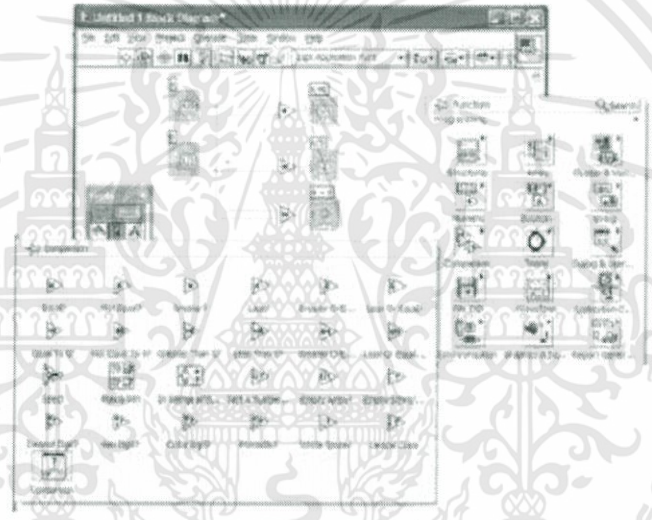
หลังจากที่เราวางอุปกรณ์ควบคุมและแสดงผลบนหน้าต่างฟร้อนท์พาเนลเรียบร้อยแล้ว ต่อไปเราจะมาพิจารณาในหน้าต่างบล็อกไดอะแกรม ดังที่กล่าวข้างต้นว่าอุปกรณ์ควบคุมและแสดงผลที่ เราวางลงไปบนหน้าต่างฟร้อนท์พาเนลนั้น ทำให้เกิดเทอร์มินอลของอุปกรณ์เหล่านั้นที่หน้าต่าง บล็อกไดอะแกรมด้วย แต่การสร้างโปรแกรมไม่ได้มีเพียงอุปกรณ์ควบคุมและแสดงผลเท่านั้น แต่ยังมีอุปกรณ์อื่นๆ ร่วมด้วย เช่น อุปกรณ์การกระทำทางคณิตศาสตร์ (บวก ลบ คูณ หาร ฯลฯ) ซึ่งอุปกรณ์ เหล่านี้จะไม่สามารถสร้างและไม่ปรากฏในหน้าต่างฟร้อนท์พาเนลจะสร้างและปรากฏในหน้าต่าง บล็อกไดอะแกรมเท่านั้น



รูปที่ 2.46 แสดงเทอร์มินอลที่เกิดขึ้นในบล็อกไดอะแกรม

1.การเปิดหน้าต่างบล็อกไดอะแกรมทำได้โดยเลือกที่เมนู Window แล้วเลือก Show Bolck Diagram ก็จะปรากฏหน้าต่างไดอะแกรมขึ้นมาดังแสดงดังรูปที่ 2.46 ซึ่งเป็นหน้าต่างไดอะแกรมที่เขียนโปรแกรมเสร็จเรียบร้อยแล้ว

2.ในหน้าต่างบล็อกไดอะแกรมในครั้งแรกที่เราเปิดมาจะมีเพียงเทอร์มินอลของอุปกรณ์ควบคุมและแสดงผลเท่านั้น เราต้องการที่จะสร้างฟังก์ชั่นสำหรับการบวก การคูณและเปรียบเท่ากัน ซึ่งทำได้โดยเลือกฟังก์ชั่นการบวกและการคูณจาก Function Palette» Numeric» Add, Multiply แล้วก็ลากฟังก์ชั่นไปวางในหน้าต่างไดอะแกรม เราจะสังเกตเห็นได้ว่าเมื่อวางฟังก์ชั่นลงไปบนหน้าต่างบล็อกไดอะแกรมแล้วจะไม่ปรากฏลาเบลของฟังก์ชั่นนั้นๆ ขึ้นมาซึ่งเราสามารถเรียกลาเบลให้ปรากฏได้โดยคลิ๊กขวาที่ฟังก์ชั่นที่ต้องการจะปรากฏ pop-up ขึ้นมาเลือก Visible Item แล้วก็เลือกที่ Label ก็จะปรากฏลาเบลของฟังก์ชั่นดังกล่าวขึ้นมาเราก็สามารถที่จะตั้งชื่อฟังก์ชั่นได้แสดงให้เห็นดังรูปที่ 2.47



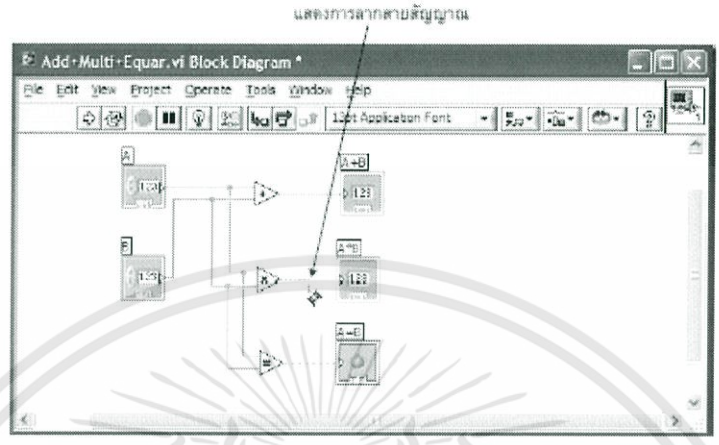
รูปที่ 2.47 แสดงฟังก์ชั่นในบล็อกไดอะแกรม

3.เลือกฟังก์ชั่นการเปรียบเทียบ (Equal) จาก FunctionPalette» Comparison» Equal ซึ่งเป็นตัวเปรียบเทียบระหว่างอินพุททั้งสองโดยถ้าอินพุททั้งสองมีค่าเท่ากันให้หลอด LED ติดและถ้าไม่เท่ากันให้หลอด LED จะดับ

4.หลังจากที่วางอุปกรณ์และฟังก์ชั่นต่างๆ ครบแล้วขั้นตอนต่อไปเป็นขั้นตอนในการลากสายสัญญาณเพื่อให้โปรแกรมทำงานตามเงื่อนไขที่กำหนดข้างต้นโดยเลือกเครื่องมือ Connect wire จาก Tool Palette หรือจะเลือกที่ Automatic Tool Select จาก Tool Palette เช่นกัน หลังจากนั้นก็เริ่มการลากสายสัญญาณเริ่มจากนำเมาส์ไปวางที่คอนเน็คเตอร์ของ Control Terminal แล้วลากไปต่อกับอินพุทคอนเน็คเตอร์ของฟังก์ชั่นจากนั้นลากสายสัญญาณจากเอาท์พุทคอนเน็คเตอร์ของฟังก์ชั่นแล้วลากไปต่อกับอินพุทคอนเน็คเตอร์ของ Indicator Terminal ทำการลากสายสัญญาณทั้งหมดให้ครบตามเงื่อนไขซึ่งแสดงดังรูปที่ 2.47

5.หลังจากนั้นเรียกหน้าต่างฟร้อนท์พาเนลขึ้นมาเพื่อทำการบันทึกไฟล์ซึ่งสำหรับตัวอย่างนี้จะขอบันทึกชื่อไฟล์ Create VI.vi โดยเลือกจาก File» Save

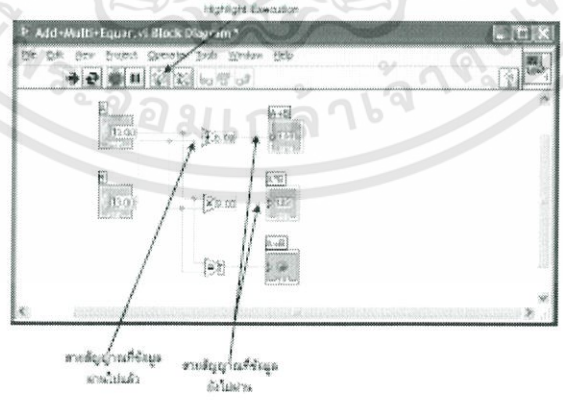
7.ทำการทดสอบโปรแกรมโดยสั่ง RUN โปรแกรมแบบต่อเนื่องแล้วเปลี่ยนแปลงอินพุท ทั้งสองไปแล้วสังเกตว่าเอาท์พุทของการบวกและการคูณถูกต้องหรือไม่และสังเกตถ้าอินพุททั้งสองมีค่าเท่ากันหลอด LED ติดหรือไม่ ถ้าตรวจเงื่อนไขต่างๆ ถูกต้องแล้ว ก็เสร็จสิ้นการสร้าง VI



รูปที่ 2.48 แสดงวิธีการลากสายสัญญาณ

2.4.2.5 การไหลของข้อมูลใน VI

ในโปรแกรม LabVIEW นั้นเราสามารถที่จะดูทิศทางการไหลของสัญญาณการทำงานของโปรแกรมที่เราสร้างขึ้นมาได้โดยเครื่องมือที่ใช้ คือ Highlight Execution การใช้งานทำได้โดยการคลิกที่สัญลักษณ์ของเครื่องมือ Highlight Execution (ลักษณะคล้ายหลอดไฟที่แถบ Tool bar) ในหน้าต่างบล็อกไดอะแกรมจากนั้นทำการกดปุ่มรันโปรแกรมจะสังเกตเห็นจุดกลมๆ เล็กๆ วิ่งจากอินพุทผ่านฟังก์ชั่นต่างๆ ตามลำดับการทำงานก่อนหลังยกตัวอย่างดังรูปที่ 2.49 มีฟังก์ชั่นการเปรียบเทียบการคูณและการบวก จะสังเกตเห็นว่าโปรแกรมจะทำการเปรียบเทียบอินพุทก่อนแล้วจึงทำการคูณและการบวกตามลำดับ



รูปที่ 2.49 แสดงการรันโดยใช้ Highlight Execution

2.5 การสร้าง VI ด้วย VI แบบเร็ว (Express VI)

ในหัวข้อนี้จะกล่าวถึงการสร้างโปรแกรม VI ด้วย VI แบบเร็ว (Express VI) ซึ่งเราต้องการที่จะสร้าง VI ที่สามารถสร้างสัญญาณคลื่นซายน์ที่เราสามารถกำหนดขนาดและความถี่ได้และให้แสดงกราฟสัญญาณที่หน้าต่างฟร้อนท์พาเนลซึ่งในโปรแกรม LabVIEW มีรูปแบบที่สร้างไว้สำหรับช่วยให้ผู้เขียนโปรแกรมสะดวกในการใช้งาน

เลือกที่ New»VI from Template» Simulation» Generate and Display เพื่อที่จะสร้าง VI สำหรับสร้างสัญญาณคลื่นซายน์

1.คลิ๊ก OK เพื่อเปิดรูปแบบในการสร้างสัญญาณคลื่นซายน์

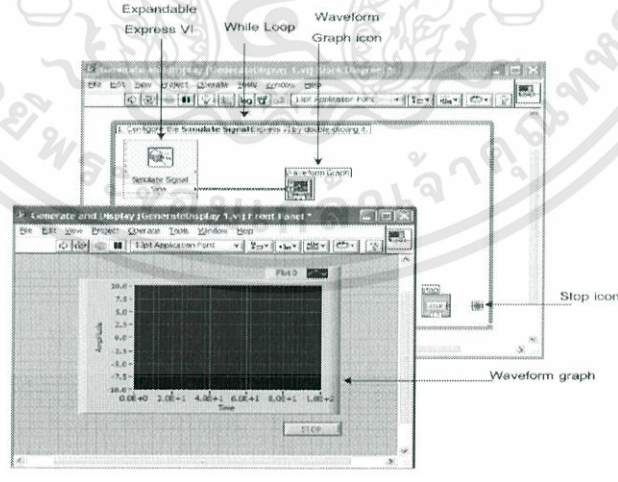
2.ในรูปที่ 2.50 สังเกตที่หน้าต่างฟร้อนท์พาเนลซึ่งจะสังเกตเห็นได้ว่าจะมีอุปกรณ์แสดงผลแบบกราฟและปุ่มหยุดการทำงานปรากฏอยู่และที่ Title bar จะแสดงชื่อของโปรแกรมชื่อ Generate and Display [Untitled] VI.

3.ที่หน้าต่างบล็อกไดอะแกรมในรูปที่ 2.50 จะประกอบไปด้วยไอคอนของ Simulate Signal Express VI, ไอคอน Waveform Graph, ไอคอน Stop Button และ While Loop ที่ Title bar จะมี ชื่อของ VI คือ Generate and Display [Untitled] VI. เช่นเดียวกับหน้าต่างฟร้อนท์พาเนล

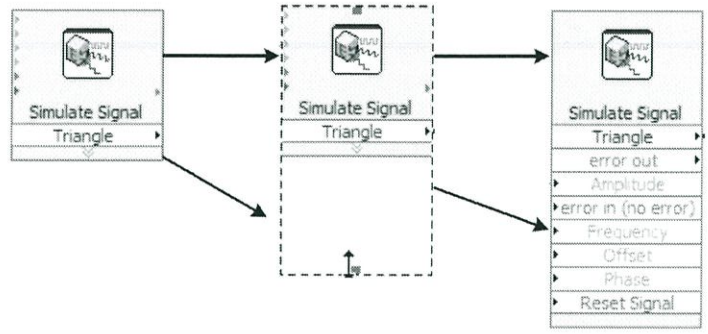
4.ที่หน้าต่างฟร้อนท์พาเนลกดปุ่มรันโปรแกรมที่แถบ Tool bar ที่กราฟแสดงผลจะปรากฏเส้นสัญญาณคลื่นซายน์

5.ถ้าต้องการหยุดการทำงานของโปรแกรมสามารถกดปุ่ม Stop ที่อยู่ทางด้านล่างขวาบนหน้าต่างฟร้อนท์พาเนล

6.เราสามารถเพิ่มอุปกรณ์อื่นเข้าไปในโปรแกรมเพื่อให้สามารถทำงานตามที่เราต้องการได้มากขึ้น และสะดวกมากขึ้นในโปรแกรมนี้จะเพิ่มอุปกรณ์ควบคุมเข้าไปเพื่อปรับขนาดของสัญญาณตามที่เรา ต้องการ



รูปที่ 2.50 แสดงการสร้าง Express VI



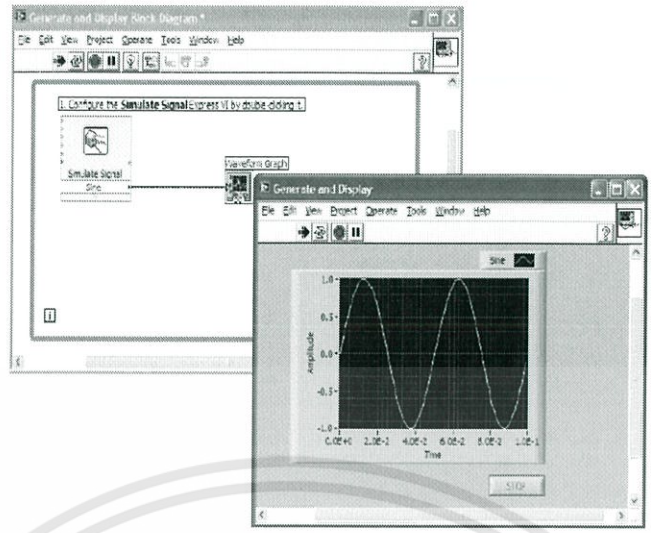
รูปที่ 2.51 การเพิ่มคุณสมบัติไอคอนของ Express VI

7.เลือกอุปกรณ์ knob จาก Numeric Controls Palette ซึ่งเป็นพาเลทย่อยของ Controls Palette ลากไปวางที่หน้าต่างฟร้อนท์พาเนลเปลี่ยนชื่อให้เป็น Amplitude

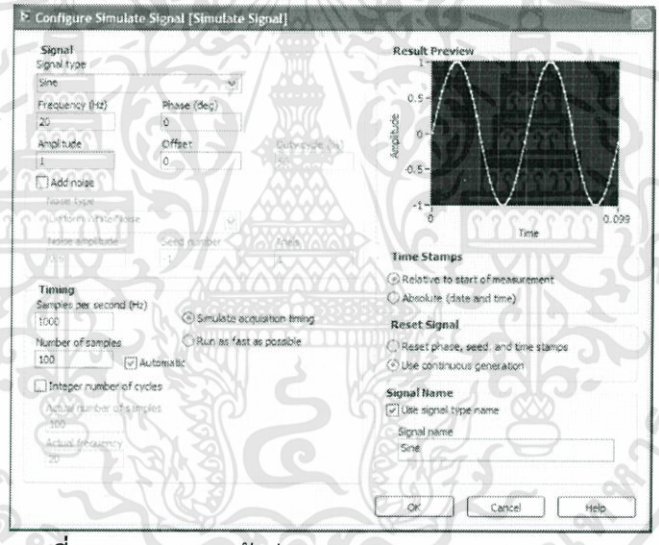
8.ที่ไอคอนของ Simulate signal Express VI ดังแสดงในรูปที่ 2.51 สามารถขยายไอคอนที่เพิ่ม คุณสมบัติในการใช้งานให้ได้มากยิ่งขึ้นซึ่งในไอคอนดังกล่าวจะมีฟังก์ชั่นการทำงานที่ซ่อนอยู่

9.ในหน้าต่างบล็อกไดอะแกรมลากสายสัญญาณจาก knob ไปต่อกับอินพุท Amplitude ของ Simulate Signal Express VI จากนั้นกลับไปยังหน้าฟร้อนท์พาเนลทำการรันโปรแกรมแล้วทำการปรับ ค่าของแอมปลิจูดที่ปุ่ม knob ไปที่ค่าต่างๆ สังเกตผลทางด้านเอาท์พุทที่แสดงในกราฟสัญญาณค่า แอมปลิจูดจะเปลี่ยนค่าไปตามที่ปรับปุ่ม knob

10.ในการกำหนดคุณสมบัติของ Simulate signal Express VI ทำได้โดยคลิ๊กเมาส์ทางด้านขวาที่ ไอคอน Simulate signal Express แล้วเลือกที่เมนู Properties จะปรากฏหน้าต่างที่มีชื่อว่า Configure Simulate Signal [Simulate Signal] ขึ้นมาดังแสดงในรูปที่ 2.51 ซึ่งในหน้าต่างนี้ประกอบไปด้วย Signal Type ใช้สำหรับให้เลือกรูปแบบของสัญญาณซึ่งประกอบไปด้วย Sine, Square, Triangle, Sawtooth และ DC และมี Result Preview อยู่ทางด้านขวาซึ่งจะแสดงลักษณะของเส้นสัญญาณตามที่ เราเลือก Signal Type นอกจากนี้ยังมียังมี Frequency (Hz) สำหรับปรับความถี่และ Amplitude สำหรับปรับขนาดของ Amplitude



รูปที่ 2.52 การต่ออุปกรณ์ใช้งานร่วมกับ Express VI



รูปที่ 2.53 แสดงหน้าต่าง Configure Simulate Signal

11.การใช้งาน Configure Simulate Signal [Simulate Signal] คือเลือกชนิดของสัญญาณ กำหนดขนาดแอมปลิจูด และกำหนดความถี่ที่เราต้องการโดยสามารถดูลักษณะของกราฟที่เรากำหนดไว้ได้ที่ Result Preview ถ้าลักษณะของกราฟเป็นไปตามที่เราต้องการแล้วกดปุ่ม OK ซึ่งขณะนี้ที่ไอคอน Simulate Signal Express VI จะมีคุณสมบัติตามที่เรากำหนด

2.6 การแก้ไขและการดีบัคโปรแกรม

ในการเขียนโปรแกรมคอมพิวเตอร์ไม่ว่าจะเขียนด้วยโปรแกรมภาษาใดก็ตามย่อมเกิดการผิดพลาด ในการเขียนโปรแกรมได้ตลอดเวลาและต้องการการแก้ไขให้ถูกต้องซึ่งการแก้ไขและการตรวจสอบความผิดพลาดของแต่ละโปรแกรมก็จะแตกต่างกันไป สำหรับโปรแกรม LabVIEW ก็มีวิธีและเครื่องมือที่ช่วยใน การแก้ไขและตรวจสอบความผิดพลาดของโปรแกรม

ในหัวข้อนี้จะพูดถึงวิธีการสร้าง การเลือก การลบ การเคลื่อนย้าย และการจัดหมวดหมู่ของ อุปกรณ์บนหน้าต่างฟร้อนท์พาเนลและหน้าต่างบล็อกไดอะแกรม ความสำคัญในการเลือกและลบ สายสัญญาณ การตรวจสอบโปรแกรมด้วย Highlight Execution และ Single-Step Code

2.6.1 เทคนิคการแก้ไขโปรแกรม

2.6.1.1 การสร้าง Controls และ Indicator บนหน้าต่างบล็อกไดอะแกรม

โดยปกติแล้วการสร้างตัวควบคุมและตัวแสดงผลเราจะทำการสร้างจากหน้าต่าง ฟร้อนท์พาเนลซึ่งได้กล่าวถึงวิธีการสร้างมาแล้วในหัวข้อการสร้าง VI สำหรับหัวข้อนี้จะกล่าวถึงวิธีการ สร้างตัวควบคุมและตัวแสดงผลในหน้าต่างบล็อกไดอะแกรมโดยจะสร้างโปรแกรมสำหรับหารากของ ตัวเลขดิจิตอลโดยมีวิธีการสร้างดังนี้

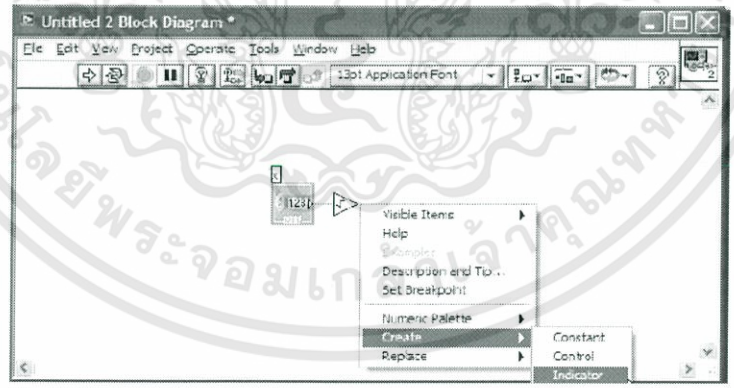
1.เปิดโปรแกรมสำหรับสร้าง VI แล้วเปิดหน้าต่างบล็อกไดอะแกรมขึ้นมา

2.คลิ๊กเมาส์ทางด้านขวาเลือก Function Palette» Numeric» Square Root แล้ว ลากมาวางที่หน้าต่างบล็อกไดอะแกรม

3.ตอนนี้เราต้องการที่จะสร้าง Controls Terminal และ Indicator Terminal ให้กับฟังก์ชั่น Square Root (แสดงดังรูปที่ 2.54) เริ่มต้นจากนำเมาส์ไปวางที่จุดเชื่อมต่อทางซ้ายมือของฟังก์ชั่น Square Root แล้วคลิ๊กขวา เลือก Create แล้วเลือก control ก็จะได้ Controls Terminal ที่มีการต่อสายสัญญาณกับจุดต่อสัญญาณของฟังก์ชั่น Square Root

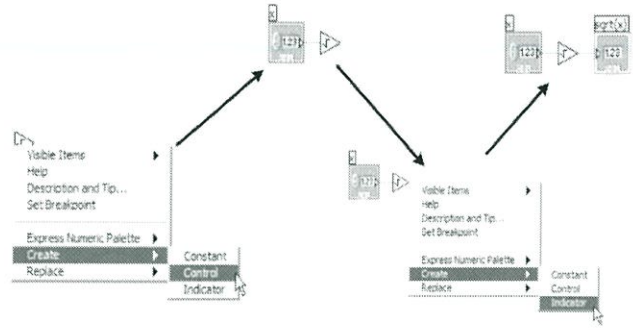
4.จากนั้นทำการสร้าง Indicator Terminal โดยนำเมาส์ไปวางที่จุดเชื่อมต่อทาง ขวามือของฟังก์ชั่น Square Root แล้วคลิ๊กขวา เลือก Create แล้วเลือก Indicator ก็จะได้ Indicator Terminal ที่มีการต่อสายสัญญาณกับจุดต่อสัญญาณของฟังก์ชั่น Square Root

5.หลังจากทำขั้นตอนในข้อ 4 และข้อ 5 เสร็จแล้วจะได้โปรแกรมดังรูปที่ 2.54 ซึ่ง สังเกตว่าที่หน้าต่างฟร้อนท์พาเนลจะปรากฏอุปกรณ์ควบคุมและแสดงผลขึ้นเช่นกัน



รูปที่ 2.54 การสร้าง Control และ Indicator บนหน้าต่างบล็อกไดอะแกรม

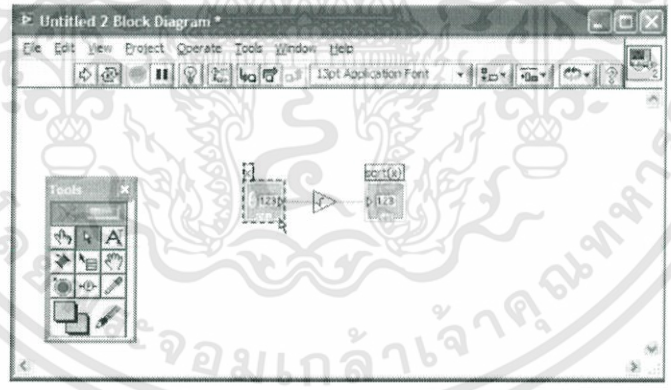
6.ทดสอบการทำงานของโปรแกรมโดยกดปุ่มรันแบบต่อเนื่อง (Run Continuously) ที่หน้าต่างฟร้อนท์พาเนล แล้วทดลองป้อนตัวเลขที่ตัวควบคุมซึ่งผลลัพธ์ที่แสดงที่ตัวแสดงผลจะเป็นรากของตัวเลขที่ป้อนเข้าไป



รูปที่ 2.55 ขั้นตอนการสร้าง Control และ Indicator บนหน้าต่างบล็อกไดอะแกรม

2.6.1.2 การเลือกอุปกรณ์

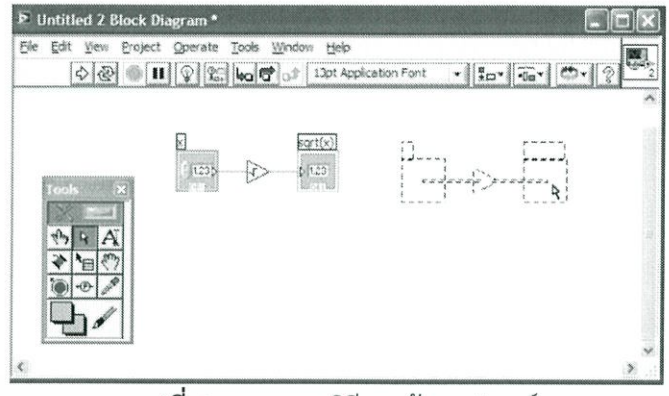
เครื่องมือที่ใช้สำหรับการเลือกอุปกรณ์คือ Positioning Tool ซึ่งอยู่ใน Tool Palette มีลักษณะดังรูปด้านซ้ายมือวัตถุประสงค์ในการเลือกอุปกรณ์คือ เพื่อย้ายตำแหน่ง ลด/ขยาย ขนาด และลบ เมื่อเราเลือก Positioning Tool แล้วเมาส์ของเราจะเปลี่ยนเป็นรูปลูกศรการเลือกอุปกรณ์ ทำได้โดยนำเมาส์ไปวางที่อุปกรณ์ที่เราต้องการจะเลือกแล้วคลิ๊กเมาส์ทางด้านซ้ายจะเกิดเส้นประล้อมรอบ อุปกรณ์นั้นแสดงว่าอุปกรณ์นั้นถูกเลือกแล้ว ถ้าต้องการเลือกอุปกรณ์มากกว่าหนึ่งตัวแต่ไม่ต้องการเลือก ทั้งหมดให้กดปุ่ม Shift ที่คีย์บอร์ด แล้วคลิ๊กเมาส์ที่อุปกรณ์ที่ต้องการเลือก หรือถ้าต้องการเลือกอุปกรณ์ ทั้งหมดทุกตัวให้คลิ๊กเมาส์บริเวณที่ว่าแล้วลากให้กรอบเส้นประสี่เหลี่ยมครอบอุปกรณ์ทุกตัวแล้วปล่อย เมาส์อุปกรณ์ก็จะถูกเลือกทั้งหมด



รูปที่ 2.56 แสดงวิธีเลือกอุปกรณ์

2.6.1.3 การเคลื่อนย้ายอุปกรณ์

การเคลื่อนย้ายอุปกรณ์ไปยังตำแหน่งต่างๆ ที่เราต้องการทำได้โดยเลือกอุปกรณ์ที่ต้องการเคลื่อนย้ายซึ่งวิธีการเลือกได้กล่าวในหัวข้อที่ผ่านมาเมื่อเลือกอุปกรณ์แล้วการเคลื่อนย้ายอุปกรณ์ทำได้โดยกดปุ่มลูกศรเลื่อนขึ้น/ลง และเลื่อนซ้าย/ขวา ที่คีย์บอร์ด หรืออีกวิธีหนึ่งคือใช้เมาส์ชี้ที่อุปกรณ์ที่เลือกไว้คลิ๊กค้างแล้วลากไปวางยังตำแหน่งที่ต้องการ



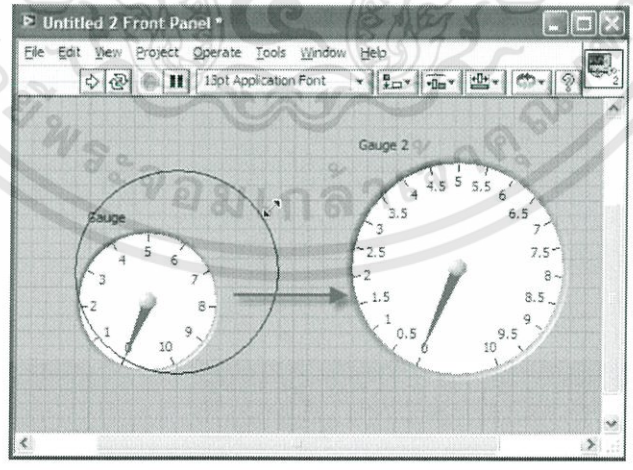
รูปที่ 2.57 แสดงวิธีการย้ายอุปกรณ์

2.6.1.4 การลบและการคัดลอกอุปกรณ์

การลบอุปกรณ์ทำได้หลายวิธีด้วยกันเริ่มต้นจากการเลือกอุปกรณ์ที่ต้องการลบ ที่คีย์บอร์ดสามารถเลือกได้ 2 ปุ่มคือ Backspace และ Delete อุปกรณ์ก็จะถูกลบและที่หน้าต่าง ฟร้อนท์พาเนลหรือบล็อกไดอะแกรมเลือกที่เมนู Edit แล้วเลือก Cut หรือ Clear ส่วนวิธีการคัดลอกอุปกรณ์นั้นทำได้โดยเลือกอุปกรณ์แล้วเลือกเมนู Edit เลือก Copy หรือจะใช้ Ctri+C ที่คีย์บอร์ดก็ได้ การวางอุปกรณ์ทำได้โดยเลือกเมนู Edit เลือก Past หรือจะใช้ Ctri+V ที่คีย์บอร์ดก็ได้เช่นกัน

2.6.1.5 การย่อและขยายขนาดของอุปกรณ์

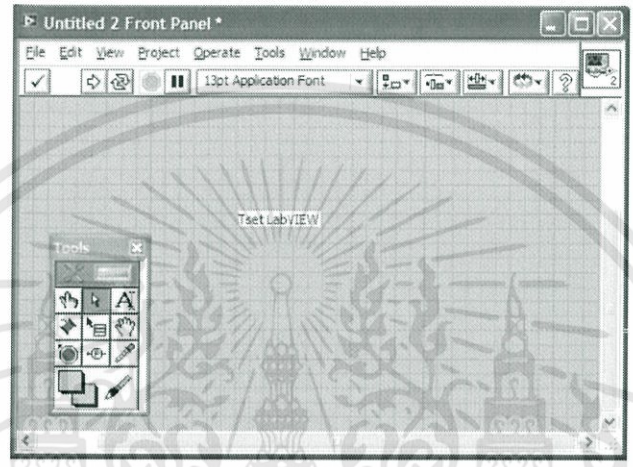
การย่อและขยายขนาดของอุปกรณ์เราจะทำที่หน้าต่างฟร้อนท์พาเนล ซึ่งจะเป็นการทำให้อุปกรณ์บนหน้าต่างฟร้อนท์พาเนลให้มีความเหมาะสมและดูสวยงาม การย่อและขยายทำได้โดยการเลือกอุปกรณ์นั้นสังเกตจะเห็นจุดสีน้ำเงินเล็กๆ อยู่บนอุปกรณ์ซึ่งโดยทั่วไปแล้วจะมี 2 จุดและ 4 จุด ใช้เมาส์คลิ๊กที่ จุดดังกล่าวจุดใดก็ได้ แล้วลากเมาส์ย่อหรือขยายให้ได้ขนาดตามที่ต้องการแล้วปล่อยเมาส์ก็จะได้ขนาดของอุปกรณ์ตามที่เราต้องการ



รูปที่ 2.58 แสดงวิธีการย่อขยายขนาดของอุปกรณ์

2.6.1.6 การสร้าง Label

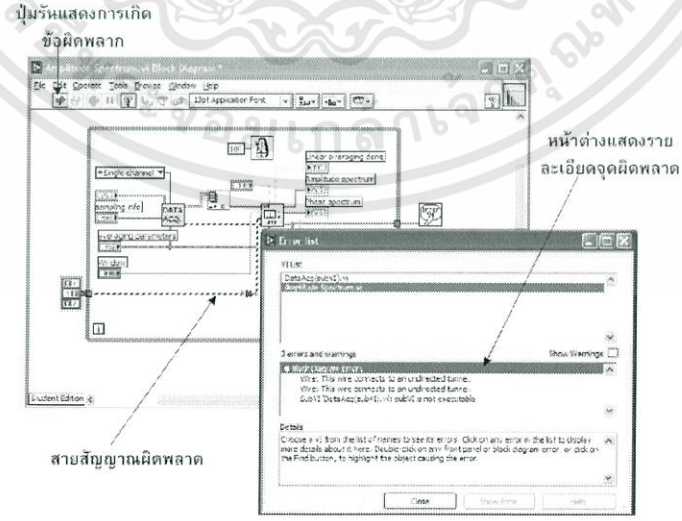
โดยปกติแล้ว Label ของอุปกรณ์จะมีติดมากับอุปกรณ์ทุกตัวอยู่แล้วแต่ก็มีอุปกรณ์บางตัวที่แถบ Label ไม่ปรากฏก็สามารถทำให้ปรากฏได้โดยคลิ๊กเมาส์ทางด้านขวาที่อุปกรณ์นั้นเลือก Visible Item แล้วเลือก Label ก็จะปรากฏแถบ Label ขึ้นมา Label อีกแบบหนึ่งคือ Label ที่เราสามารถสร้างขึ้นมาเอง โดยเลือกเครื่องมือ Edit Text จาก Tools Palette แล้วคลิ๊กเมาส์บริเวณที่เราต้องการสร้าง Label จะได้ Label ตามรูปที่ 2.58 ซึ่งเราสามารถตั้งชื่อได้ตามที่ต้องการ



รูปที่ 2.59 แสดงวิธีการสร้าง Label

2.6.2 เทคนิคการดีบัคโปรแกรม

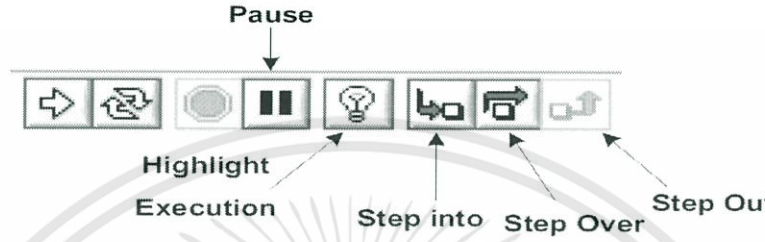
ในหัวข้อนี้เราจะกล่าวถึงพื้นฐานในการตรวจสอบโปรแกรมของ LabVIEW ซึ่งจะมีเครื่องมือที่ช่วยในการตรวจสอบหาข้อผิดพลาดของโปรแกรม เช่น Highlight Execution, Single Stepping, breakpoints เป็นต้นซึ่งเราจะได้กล่าวถึงในหัวข้อนี้



รูปที่ 2.60 แสดงการดีบัคโปรแกรม

2.6.2.1 การตรวจหาข้อผิดพลาด

เมื่อเราไม่สามารถรันโปรแกรมได้สังเกตจากปุ่มรันจะมีลักษณะแตกออก จากกันนั่น แสดงว่าเกิดความผิดพลาดในการเขียนโปรแกรมขึ้นความผิดพลาดส่วนใหญ่แล้วจะเกิดจากการลาก สายสัญญาณผิด เราจะต้องทำการแก้ไขโปรแกรมจนถูกต้องจึงจะสามารถทำการรันโปรแกรมได้ การ ค้นหาจุดผิดพลาดทำได้โดยคลิ๊กเมาส์ที่ปุ่มรันจะปรากฏไดอะล็อกบล็อก ขึ้นมาซึ่งจะแสดงรายละเอียดจุด ผิดพลาดทั้งหมดดังแสดงในรูปที่ 2.61



รูปที่ 2.61 เครื่องมือที่ใช้ตรวจสอบโปรแกรม

2.6.2.2 Highlight Execution

เราสามารถที่จะดูทิศทางการไหลของสัญญาณ ลำดับขั้นการทำงานของโปรแกรมได้โดย กดปุ่ม Highlight Execution ที่ Tools bar ในหน้าต่างบล็อกไดอะแกรมแสดงดังรูปที่ 4.96 วัตถุประสงค์ในการตรวจสอบโปรแกรมแบบนี้เพื่อดูลำดับขั้นการทำงานและดูทิศทางการไหลของ สัญญาณในหน้าต่างบล็อกไดอะแกรมแสดงดังรูปที่ 2.62 เมื่อเราทำการคลิ๊กที่ปุ่ม Highlight Execution แล้วกดปุ่มรันโปรแกรม เราจะเห็นจุดกลมๆ เล็กวิ่งจากจุดเริ่มต้นของโปรแกรมผ่านฟังก์ชั่นการทำงานต่างๆ ไปจนถึงเอาท์พุทและสิ้นสุดการทำงานจุดเล็กนี้จะแทนการไหลของข้อมูลนั่นเอง

Highlight Execution ทำให้เราสามารถเห็นถึงขั้นตอนการทำงานของโปรแกรมและทำ ให้เราเข้าใจมากยิ่งขึ้นซึ่งมีความสำคัญและเป็นประโยชน์การสร้างโปรแกรมเป็นอย่างมาก

รูปที่ 2.62 แสดงทิศทางการไหลของสัญญาณ

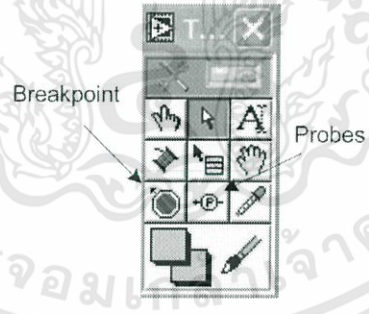
2.6.2.3 การตรวจสอบโปรแกรมแบบ Single Stepping

สำหรับการตรวจหาข้อผิดพลาดของโปรแกรมแบบโหนดต่อโหนด เรียกวิธีการนี้ว่า Single Stepping สัญลักษณ์ของ Single Stepping แสดงดังรูปที่ 2.61 ซึ่งการตรวจสอบการทำงานของโปรแกรมเราจะใช้เมาส์คลิ๊กที่ปุ่ม Single Stepping ซึ่งประกอบไปด้วยปุ่ม Step into, Step over และ Step out แทนการกดปุ่มรัน โดยการสั่งให้โปรแกรมทำงานนั้นเราจะใช้ 2 ปุ่ม คือ Step into และ Step over โดยการกดเมาส์ที่ปุ่มดังกล่าวโปรแกรมก็จะทำงานทีละโหนด จนกว่าจะจบการทำงาน ถ้าต้องการออกจากการรันโปแกรมแบบ Single Step ให้กดที่ปุ่ม Step out หรือที่ปุ่มหยุดการทำงานของ โปรแกรม ถ้าต้องการดูทิศทางการไหลของข้อมูลแต่ละโหนด ก็ทำได้โดยการเลือก Highlight Execution แล้วกดปุ่มรันแบบ Single Stepping การทำแบบนี้จะทำให้โปรแกรมรันทีละโหนดและเห็น ทิศทางการไหลของข้อมูลแต่ละโหนดด้วย

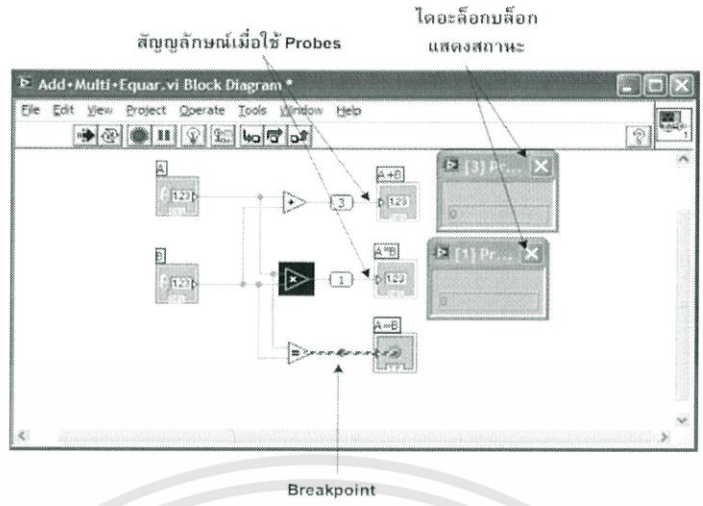
2.6.2.4 Breakpoint and Probes

Breakpoint จะเป็นตัวที่ใช้ในการสั่งให้โปรแกรมหยุดทำงาน ณ. ตำแหน่งที่เราต้องการ ซึ่งลักษณะของ Breakpoint แสดงดังรูปที่ 2.63 การใช้งานทำได้โดยใช้เมาส์คลิ๊กที่ตัว Breakpoint เมาส์ก็จะเปลี่ยนเป็นรูปสัญลักษณ์ของ Breakpoint จากนั้นนำเมาส์ไปคลิ๊กบริเวณสายสัญญาณที่ต้องการให้โปรแกรมหยุดทำงาน

Probes แสดงดังรูปที่ 2.63 เมื่อเราต้องการทราบค่าหรือสถานะของโปรแกรม ณ. จุดใดของโปรแกรมก็ใช้เครื่องมือ Probes ไปคลิ๊กบริเวณนั้นซึ่งจะปรากฏไดอะล็อกบล็อกขึ้นมาแสดงค่า ข้อมูลหรือสถานะของโปรแกรม ณ. จุดนั้นให้เราทราบ การใช้งาน Breakpoint และ Probes แสดงดังรูปที่ 2.64



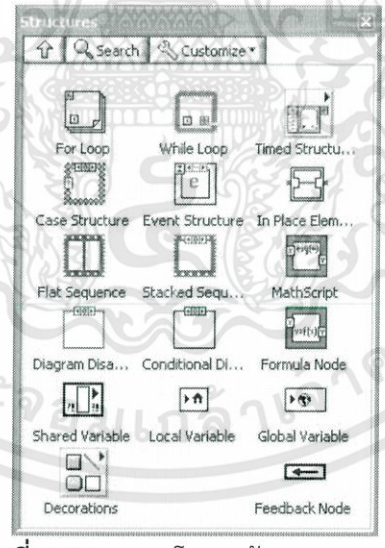
รูปที่ 2.63 แสดง Tools Palette



รูปที่ 2.64 แสดงการใช้ Breakpoint และ Probes

2.7 รูปแบบโครงสร้างการควบคุมการทำงานของโปรแกรม

ในหัวข้อนี้จะกล่าวถึงโครงสร้างการควบคุมการทำงานของโปรแกรม VIs ซึ่งโครงสร้างในโปรแกรม LabVIEW ที่จะกล่าวถึงคือ For Loop, While Loop, Shift Register and Feedback Node, Case Structure, Flate and Strack Sequency, Formula Node, MathScrip



รูปที่ 2.65 แสดงโครงสร้างการควบคุม

2.7.1 For Loop

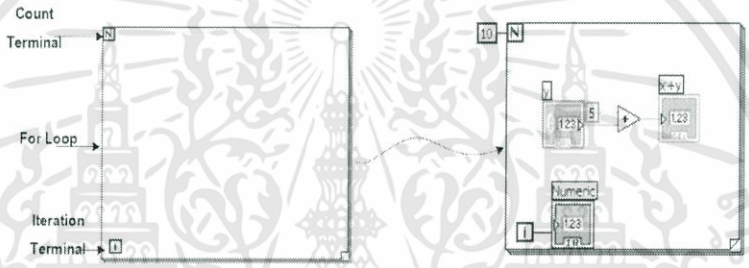
For Loop เป็นฟังก์ชั่นคำสั่งให้โปรแกรมทำงานซ้ำจนกว่าการกระทำซ้ำนั้นจะเท่ากับจำนวน N เมื่อ N คือจำนวนครั้งที่ต้องการให้ทำซ้ำ

ใน For Loop นั้นจะมีเทอร์มินอลอยู่ 2 จุดคือ count terminal และ iteration terminal สำหรับ count terminal นั้นมีหน้าที่เป็นตัวกำหนดว่าจะให้โปรแกรมทำงานซ้ำกี่รอบซึ่งสามารถจะกำหนดให้ปรับเปลี่ยนจำนวนรอบหรือเป็นค่าคงที่ก็ได้ส่วน iteration terminal จะทำ

หน้าที่แสดงจำนวนรอบการทำงานว่าโปรแกรมทำงานในวงรอบไปกี่ครั้งแล้วตัวเลขที่แสดงของ iteration terminal จะมีค่าเท่ากับ N-1 การใช้งาน For Loop แสดงดังรูปที่ 2.66

การสร้าง For Loop ทำได้โดยการเลือกไปที่หน้าต่างบล็อกไดอะแกรมแล้วเลือก Function» Structure แล้วใช้เมาส์เลือกที่ For Loop นำเมาส์ไปคลิ๊กที่หน้าต่างบล็อกไดอะแกรมแล้ว ลากให้มีขนาดที่ต้องการแล้วปล่อยเมาส์ก็จะได้โครงสร้างการทำงานของ For Loop นอกจากนี้แล้วยัง สามารถทำการย่อขยายขนาดของ For Loop ได้โดยใช้เมาส์ไปชี้ตรงบริเวณเส้นขอบของ For Loop จะปรากฏจุดสี่เหลี่ยมเล็กๆ บริเวณเส้นขอบของ For Loop ใช้เมาส์ไปวางตรงบริเวณจุดดังกล่าวแล้วเมาส์จะเปลี่ยนเป็นรูปลูกศรจากนั้นคลิ๊กเมาส์ค้างแล้วลากย่อหรือขยายได้ตามต้องการ

การทำงานของ For Loop นอกจากจะทำงานเพียงวงรอบเดี่ยวๆ แล้วยังสามารถทำงาน ซ้อนวงรอบได้ โดยการทำงานนั้นจะทำงานวงรอบด้านในก่อนจนครบเงื่อนไขแล้วจึงมาทำวงรอบนอกหนึ่งรอบแล้วก็กลับไปทำงานในวงรอบในอีกครั้ง การทำงานจะทำจนกว่าวงรอบนอกครบตามเงื่อนไข โปรแกรมจึงจะหยุดทำงาน



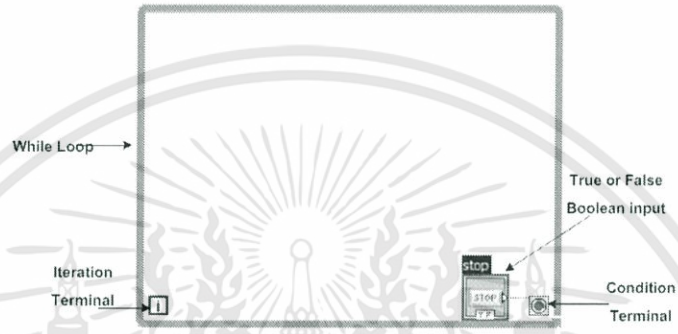
รูปที่ 4.66 แสดงโครงสร้างของ For Loop

2.7.2 While Loop

While Loop เมื่อเปรียบเทียบกับการเขียนโปรแกรมด้วยภาษาอื่นๆ ก็จะเป็นได้กับ โครงสร้างของคำสั่ง Repeats Until นั่นเองซึ่งลักษณ์ของ While Loop แสดงดังรูปที่ 2.67 หลักการทำงานของ While Loop คือจะทำงานภายในวงรอบไปจนกว่าเงื่อนไขที่ Condition Terminal จะเป็น จริงจึงจะหยุดทำงาน

สำหรับ While Loop นั้นจะมีเทอร์มินอลอยู่ 2 เทอร์มินอลเช่นเดียวกันกับ For Loop คือ Iteration Terminal และ Condition Terminal การทำงานของ Iteration Terminal จะทำงาน เหมือนกันกับ Iteration Terminal ของ For Loop คือทำหน้าที่แสดงจำนวนรอบที่โปรแกรมกระทำในวงรอบซึ่งแสดงผลออกมาในลักษณะของตัวเลขส่วน Condition Terminal เป็นการกำหนดเงื่อนไขให้โปรแกรมหยุดทำงานซึ่งเงื่อนไขในการหยุดทำงานคือจะหยุดทำงานเมื่อเงื่อนไขเป็นจริง (True) หรือจะให้หยุดทำงานเมื่อเงื่อนไขเป็นเท็จ (False) ก็สามารถที่จะกำหนดได้ นอกจากนี้แล้วเรายังสามารถกำหนด เงื่อนไขในการหยุดทำงานได้อีกหนึ่งวิธีคือการนำเอาค่าเอาท์พุทของ Iteration Terminal มาเปรียบเทียบกับค่าที่เราต้องการไม่ว่าจะเป็นค่าคงที่หรือปรับเปลี่ยนได้ โปรแกรมจะทำงานวนในวงรอบและนำค่าเอาท์พุทของ Iteration Terminal มาเปรียบเทียบกับค่าที่เรากำหนดไว้แล้วจะส่งเอาท์พุทที่ได้ไปยัง Condition Terminal เมื่อจำนวนรอบการทำงานมีค่าเท่ากับกับค่าที่เรากำหนดโปรแกรมจะหยุดทำงาน ในวงรอบนี้ทันทีการกำหนดเงื่อนไขแบบนี้เปรียบได้กับการกำหนดค่า N ใน For Loop นั่นเอง

การสร้าง While Loop ทำได้โดยการเลือกไปที่หน้าต่างบล็อกไดอะแกรมแล้วเลือก Function >> Structure แล้วใช้เมาส์เลือกที่ While Loop นำเมาส์ไปคลิ๊กที่หน้าต่างบล็อกไดอะแกรม แล้วลากให้มีขนาดที่ต้องการแล้วปล่อยเมาส์ก็จะได้โครงสร้างการทำงานของ While Loop นอกจากนี้ แล้วยังสามารถทำการย่อขยายขนาดของ While Loop ได้โดยใช้เมาส์ไปชี้ตรงบริเวณเส้นขอบของ While Loop จะปรากฏจุดสี่เหลี่ยมเล็กๆ บริเวณเส้นขอบของ While Loop ใช้เมาส์ไปวางตรงบริเวณจุดดังกล่าวแล้วเมาส์จะเปลี่ยนเป็นรูปลูกศรจากนั้นคลิ๊กเมาส์ค้างแล้วลากย่อหรือขยายได้ตามต้องการ



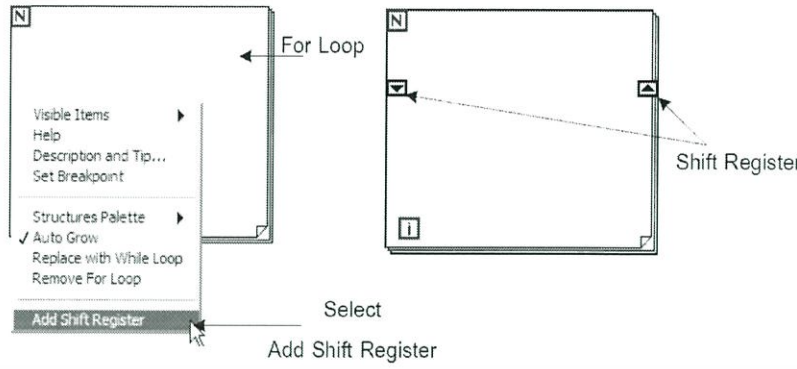
รูปที่ 2.67 แสดงโครงสร้างของ While Loop

2.7.3 Shift Register and Feedback Node

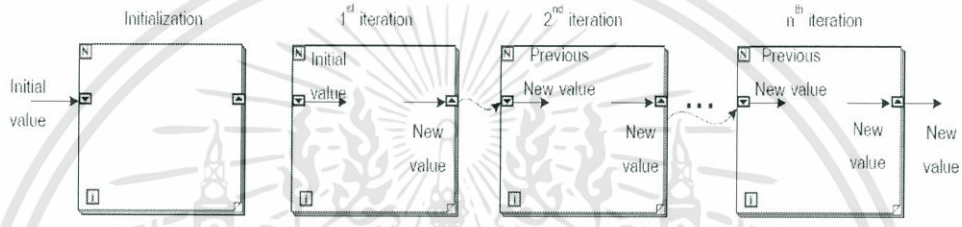
2.7.3.1 Shift Register

Shift Register ประกอบไปด้วยเทอร์มินอลสองตัวคือเทอร์มินอลทางด้านอินพุทและ เทอร์มินอลทางด้านเอาท์พุท Shift Register สามารถใช้ทำงานร่วมกับ For Loop และ While Loop เท่านั้นการสร้าง Shift Register ทำได้โดยการใช้เมาส์คลิ๊กขวาบริเวณเส้นขอบของ For Loop หรือ While Loop แล้วเลือก Add Shift Register ก็จะได้ Shift Register ที่อยู่บนเส้นขอบทางซ้ายและขวา ของ For Loop และ While Loop ดังแสดงให้เห็นในรูปที่ 2.68

การทำงานของ Shift Register คือจะส่งค่าข้อมูลจากเทอร์มินอลทางซ้ายไปยัง เทอร์มินอลทางขวาค่าเอาท์พุทของวงรอบที่ผ่านมาจะมาเป็นค่าเริ่มต้นของวงรอบปัจจุบันดังแสดงในรูปที่ 2.68 ค่าเริ่มต้นของ Shift Register สามารถกำหนดได้แต่ถ้าเราไม่กำหนดค่าเริ่มต้นก็จะมีค่ามาตรฐานที่มาจากโปรแกรม Shift Register สามารถใช้ได้กับข้อมูลทุกชนิดไม่ว่าจะเป็นตัวเลข ตัวหนังสือลอจิกทั้งที่เป็นค่าเดี่ยวหรืออาร์เรย์



รูปที่ 2.68 แสดงโครงสร้างของ Shift Register

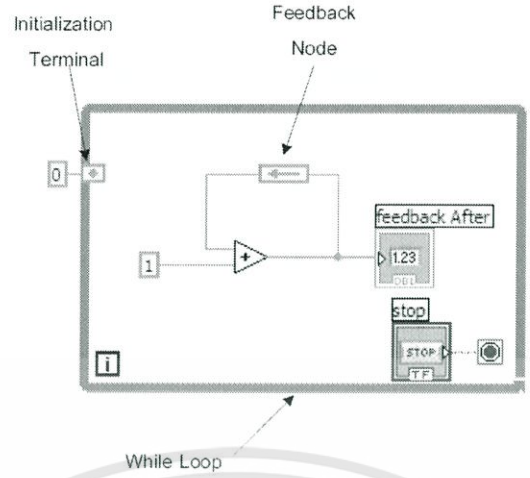


รูปที่ 2.69 แสดงการทำงานของ Shift Register

2.7.3.2 Feedback Node

Feedback Node สามารถทำงานร่วมกับ For Loop และ While Loop ได้ เช่นเดียวกันกับ Shift Register การทำงานของ Feedback Node คือจะเก็บข้อมูลไว้ที่ Iteration เมื่อ โปรแกรมทำงานภายในวงรอบเสร็จแล้วจะส่งข้อมูลดังกล่าวไปที่ Iteration ของวงรอบต่อไป ทั้งนี้ Feedback Node ยังสามารถกำหนดค่าเริ่มต้นได้ด้วยและสามารถใช้ได้กับข้อมูลทุกชนิด ลักษณะการต่อ ใช้งาน Feedback Node แสดงดังรูปที่ 2.70

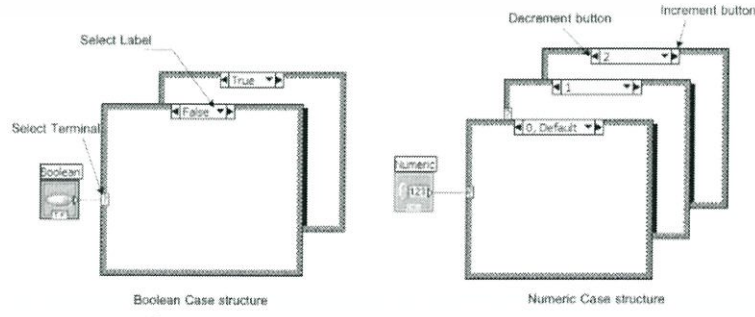
การสร้าง Feedback Node ทำได้โดยการเลือกไปที่หน้าต่างบล็อกไดอะแกรมแล้วเลือก Function» Structure แล้วใช้เมาส์เลือกที่ Feedback Node (แสดงดังรูปที่ 2.70) นำเมาส์ไปคลิ๊ก ที่หน้าต่างบล็อกไดอะแกรม (ที่มี For Loop หรือ While Loop อยู่แล้ว) แล้วลากเมาส์ไปวางภายใน กรอบของ For Loop หรือ While Loop เท่านั้นเพราะถ้าวางเมาส์ภายนอกกรอบของ For Loop หรือ While Loop จะไม่สามารถวาง Feedback Node



รูปที่ 2.70 แสดงการใช้งาน Feedback Node

2.7.4 Case Structure

Case Structure เป็นโครงสร้างการเขียนโปรแกรมตามเงื่อนไขของตัวหนังสือ (Conditional text) ซึ่งถ้าเปรียบเทียบกับการเขียนโปรแกรมภาษาอื่นๆ (text-based programming languages) สามารถเปรียบเทียบได้กับโครงสร้างของคำสั่ง *If...Then...Else* การเรียกใช้งานฟังก์ชั่น Case Structure ลงบนหน้าต่างบล็อกไดอะแกรมทำได้โดยเลือก Function» Structure แล้วใช้เมาส์เลือกที่ Case Structure (แสดงดังรูปที่ 2.65) นำเมาส์ไปคลิ๊กที่หน้าต่างบล็อกไดอะแกรมแล้วลากให้มี ขนาดที่ต้องการแล้วปล่อยเมาส์ก็จะได้โครงสร้างการทำงานของ Case Structure นอกจากนี้แล้วยัง สามารถทำการย่อขยายขนาดของ Case Structure ได้โดยใช้เมาส์ไปชี้ตรงบริเวณเส้นขอบของ Case Structure จะปรากฏจุดสี่เหลี่ยมเล็กๆ บริเวณเส้นขอบของ While Loop ใช้เมาส์ไปวางตรงบริเวณจุดดังกล่าวแล้วเมาส์จะเปลี่ยนเป็นรูปลูกศรจากนั้นคลิ๊กเมาส์ค้างแล้วลากย่อหรือขยายได้ตามต้องการ สามารถแสดงลักษณะของ Case Structure ได้ดังรูปที่ 2.71 ซึ่งมีส่วนประกอบที่สำคัญอยู่ 2 ส่วนคือ Select Terminal และ Select Label โดยที่ Select Terminal จะทำหน้าที่รับอินพุทเข้ามาแล้วทำการตรวจสอบหรือเปรียบเทียบว่าข้อมูลที่รับเข้ามาเป็นข้อมูลชนิดใดและตรงกับเงื่อนไขใดใน Case Structure เพื่อทำให้โปรแกรมแกรมทำงานตามเงื่อนไขที่ต้องการ ส่วน Select Label จะทำหน้าที่สร้างหน้าต่างย่อยของ Case Structure ซึ่งสามารถเพิ่มและลดเงื่อนไขได้ถึง 2^{31}-1 เงื่อนไขการส่งข้อมูลออกเอาท์พุทนั้นทุกๆ กรณีจะส่งข้อมูลออกที่เอาท์พุทเดียวกับซึ่งทางออกของสัญญาณเอาท์พุทเรียกว่า Output Tunnel



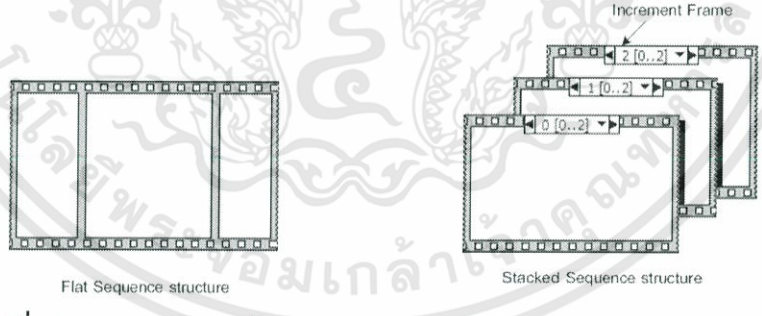
รูปที่ 2.71 แสดงโครงสร้างของ Case Structure

2.7.5 Flat and Stack Sequence Structures

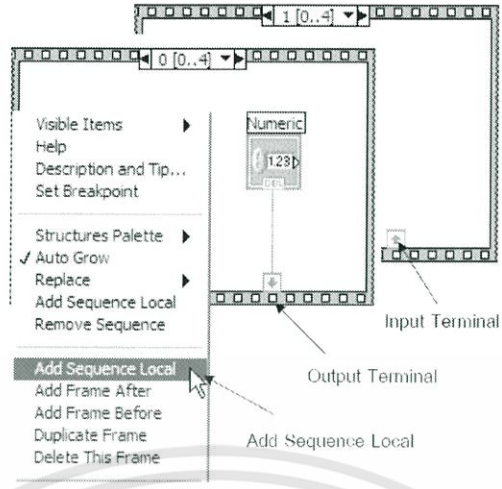
Sequence Structures เป็นการทำงานแบบลำดับขั้นรูปแบบของ Sequence Structures มีลักษณะเหมือนแผ่นฟิล์ม สำหรับ Sequence Structures มีทั้งหมด 2 แบบคือ Flat Sequence และ Stack Sequence ลักษณะของ Flat Sequence และ Stack Sequence แสดงได้ดังรูปที่ 2.72 ข้อแตกต่างของทั้ง 2 แบบ คือ Flat Sequence มีมีลักษณะของเฟรมต่อกันไปไม่ว่าจะ เพิ่มหรือลดจำนวนเฟรมเราก็จะทั้งหมดว่ามีกี่เฟรมส่วน Stack Sequence จะมีลักษณะคล้ายกับ Case Structure กล่าวคือจะมี Select Label สำหรับเลือกเฟรมที่ซ่อนอยู่ภายในโดยจะตัวเลขบอกว่ากำลังเปิด เฟรมใดอยู่

การสร้าง Flat และ Stack Sequence Structures ทำได้โดยการเลือกไปที่หน้าต่าง บล็อกไดอะแกรมแล้วเลือก Function» Structure แล้วใช้เมาส์เลือกที่ Flat หรือ Stack Sequence Structures ตามที่ต้องการ (แสดงดังรูปที่ 2.65) นำเมาส์ไปคลิ๊กที่หน้าต่างบล็อกไดอะแกรมแล้วลากให้มีขนาดที่ต้องการแล้วปล่อยเมาส์ก็จะได้โครงสร้างการทำงานของ Flat และ Stack Sequence Structures

การที่จะนำข้อมูลของเฟรมหนึ่งไปยังอีกเฟรมหนึ่งสามารถทำได้โดยคลิ๊กเมาส์ทางขวาที่ บริเวณกรอบ Sequence Structures แล้วเลือก Add Sequence Local จะปรากฏเทอร์มินอลขึ้นที่ ขอบของ Sequence Structures จะนั้นลากสายสัญญาณที่เราต้องการเอาออกเอาท์พุทไปต่อกับ เทอร์มินอลซึ่งเทอร์มินอลดังกล่าวก็จะเปลี่ยนเป็นลักษณะที่มีลูกศรพุ่งออกจากเฟรม ในขณะเดียวกันทุกเฟรมที่เหลือก็จะปรากฏเทอร์มินอลขึ้นเช่นกันแต่จะมีลักษณะลูกศรชี้เข้าไปข้างในเฟรมแสดงดัง รูปที่ 2.73



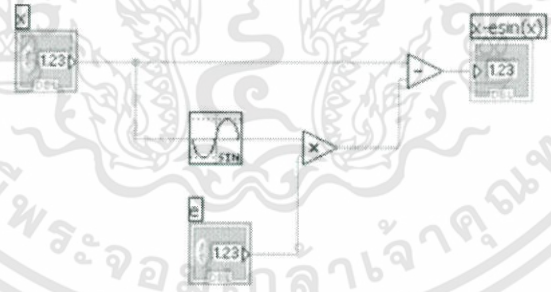
รูปที่ 2.72 แสดงโครงสร้างของ Flat และ Stack Seqence Structures



รูปที่ 2.73 การ Add Sequence Local ของ Stack Sequence Structures

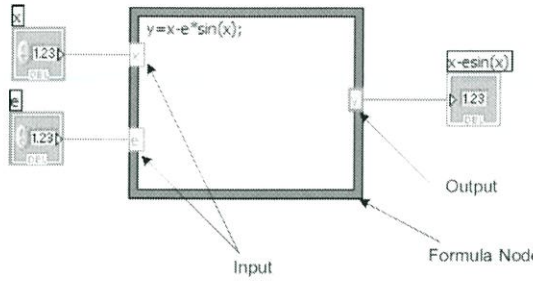
2.7.6 Formula Node

Formula Node เป็นโครงสร้างของคำสั่งที่มีความใกล้เคียงกับการเขียนโปรแกรมด้วย ภาษาอื่นๆ ทั่วไปซึ่ง Formula Node จะถูกนำมาใช้เมื่อต้องการให้โปรแกรมทำงานตามฟังก์ชั่นทาง คณิตศาสตร์ที่มีความยุ่งยาก การใช้อุปกรณ์ในการสร้างให้โปรแกรมทำงานตามฟังก์ชั่นทางคณิตศาสตร์ หนึ่งฟังก์ชั่นอาจต้องใช้อุปกรณ์หลายตัวซึ่งอาจทำให้การตรวจสอบโปรแกรมเป็นไปด้วยความยุ่งยากและ โปรแกรมมีขนาดใหญ่เกินไป ยกตัวอย่างการสร้างโปรแกรมให้ทำงานตามสมการ $y = x - e\sin(x)$ ซึ่งถ้าเขียนโปรแกรมด้วยการใช้อุปกรณ์ทั่วไปจะได้ลักษณะของโปรแกรมดังรูปที่ 2.74 แต่หากใช้ Formula Node จะได้ลักษณะของโปรแกรมดังรูปที่ 2.75



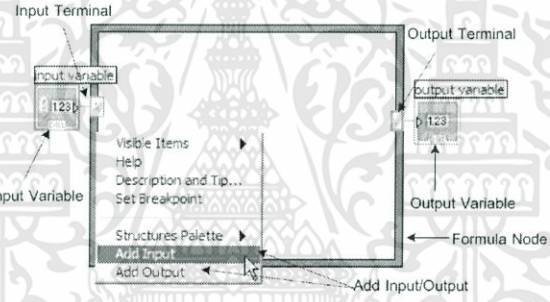
รูปที่ 2.74 การสร้างโปรแกรมจากอุปกรณ์ทั่วไป

การทำงานของ Formula Node จะเป็นลักษณะของการกำหนดฟังก์ชั่นทางคณิตศาสตร์ ลงไปใน Formula Node (แสดงดังรูปที่ 2.74) แล้วจึงกำหนดเทอร์มินอลเพื่อเชื่อมต่ออินพุทและ เอาท์พุท การสร้าง Formula Node ทำได้โดยการเลือกไปที่หน้าต่างบล็อกไดอะแกรม แล้วเลือก Function» Structure แล้วใช้เมาส์เลือกที่ Formula Node (แสดงดังรูปที่ 2.74) นำเมาส์ไปคลิ๊กที่ หน้าต่างบล็อกไดอะแกรมแล้วลากให้มีขนาดที่ต้องการแล้วปล่อยเมาส์ก็จะได้โครงสร้างการทำงานของ Formula Node



รูปที่ 2.75 โครงสร้างและการใช้งาน Formula Node

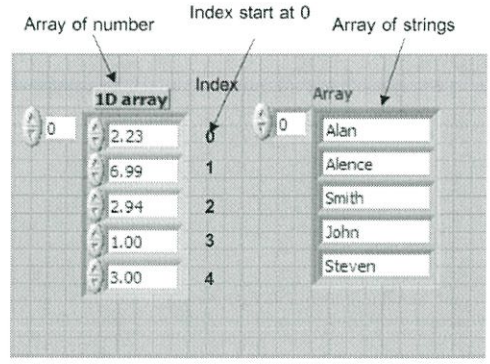
การกำหนดเทอร์มินอลอินพุทและเอาท์พุทสามารถทำได้โดยการคลิ๊กเมาส์ขวาที่ขอบของ Formula Node แล้วเลือก Add Input หรือ Add Output (แสดงดังรูปที่ 2.76) ที่เทอร์มินอลสามารถที่จะกำหนดชื่อได้และค่าอินพุทและเอาท์พุทจะเป็นข้อมูลแบบตัวเลข (Numeric) การกำหนดตัวแปรอินพุทสามารถกำหนดให้เป็นค่าคงที่หรือค่าที่ปรับเปลี่ยนได้ส่วนเอาท์พุทจะกำหนดให้เป็นตัวแสดงผล ชื่อที่เทอร์มินอลก็คือชื่อของตัวแปรของฟังก์ชั่นทางคณิตศาสตร์ที่สร้างขึ้นใน Formula Node



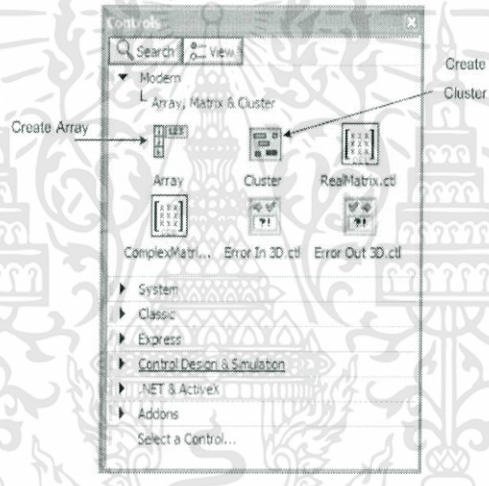
รูปที่ 2.76 แสดงการเพิ่มอินพุทและเอาท์พุทของ Formula Node

2.8 พื้นฐานการใช้งานอาร์เรย์

อาร์เรย์เป็นการเก็บตัวแปรหรือข้อมูลชนิดเดียวกันหลาย ๆ ข้อมูลไว้ด้วยกันส่วนคลัสเตอร์จะเก็บข้อมูลที่ชนิดต่างกันไว้ด้วยกันได้ ปัญหาที่พบบ่อยคือเวลาโปรแกรมทำงานจะมีข้อมูลเป็นจำนวนมาก ฉะนั้นอาร์เรย์หรือคลัสเตอร์จะช่วยในการเก็บข้อมูลอาร์เรย์สามารถสร้างได้หลายมิติเริ่มตั้งแต่ 1 มิติ ซึ่ง ในอาร์เรย์ 1 มิติสามารถเก็บข้อมูลได้สูงถึง 2^{31} -1 ข้อมูลค่าสูงสุดที่จะเก็บได้ก็ขึ้นอยู่กับขนาดของหน่วยความจำอาร์เรย์ในโปรแกรม LabVIEW สามารถเก็บข้อมูลได้หลายชนิดยกเว้น Chart และ Graph วิธีการอ้างถึงข้อมูลในอาร์เรย์เรียกว่าดัชนี (Index) ซึ่งมีค่าเริ่มต้นตั้งแต่ 0 ไปจนถึง N-1 เมื่อ N คือ จำนวนข้อมูลในอาร์เรย์ ในรูปที่ 2.77 เป็นอาร์เรย์ 1 มิติ โดยมีค่าดัชนีเริ่มต้นที่ 0



รูปที่ 2.77 แสดง Array One Dimension (1-D)



รูปที่ 2.78 การเรียกใช้งาน Array

2.8.1 การสร้างอาร์เรย์ด้วย Control และ Indicator

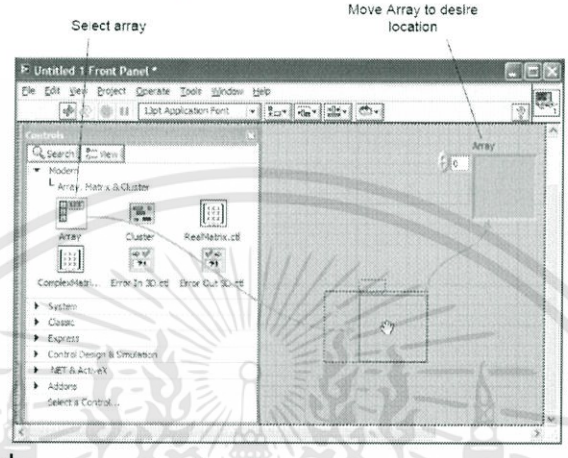
เราสามารถสร้างอาร์เรย์ด้วย Control และ Indicator ของ Numeric, Boolean, Strings ซึ่งการสร้างอาร์เรย์จะประกอบไปด้วย Array Shell และ Matrix & Cluster อยู่ในพาเลทย่อย ของ Controls» Modern ดูได้จากรูปที่ 2.79 ขั้นตอนในการสร้างอาร์เรย์ด้วย Control และ Indicator มีดังนี้

- เลือกบล็อกของอาร์เรย์จาก Array, Matrix & Cluster ในพาเลทย่อยของ Control» Modern พาเลท แล้วลากไปวางในหน้าต่างฟร้อนท์พาเนล ดังแสดงในรูปที่ 2.79
- เลือกข้อมูลที่ต้องการ เช่น Numeric, Boolean และ String แล้วลากไปวางไว้ในบล็อก อาร์เรย์ ดังแสดงในรูปที่ 2.79 โดยที่อาร์เรย์จะปรับขนาดให้เหมาะสมกับชนิดของข้อมูลที่นำมาวาง

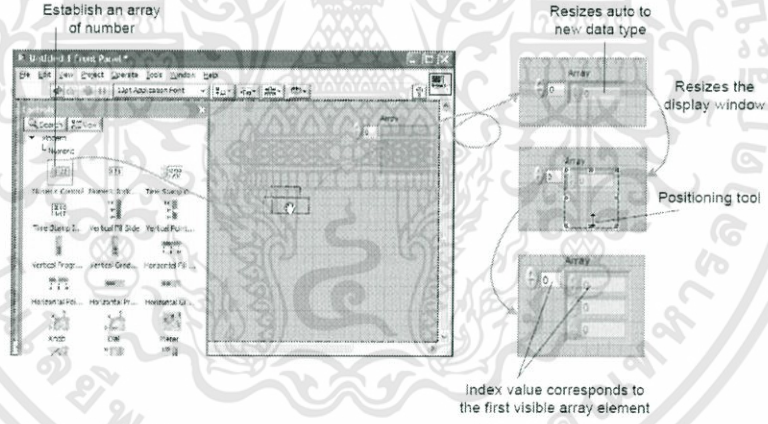
เราสามารถที่จะย่อหรือขยายอาร์เรย์เพื่อดูค่าของสมาชิกที่อยู่ภายในอาร์เรย์ได้ โดยใช้เมาส์ไปวางบริเวณอาร์เรย์ แล้วจะปรากฏจุดสี่เหลี่ยมเล็กๆ ขึ้นรอบๆ บล็อกของอาร์เรย์

จากนั้นใช้เมาส์ไป วางบริเวณจุดเหล่านั้นจุดใดก็ได้ตามต้องการ แล้วเมาส์จะเปลี่ยนไอคอนไปเป็นลูกศรแล้วทำการคลิ๊กเมาส์ค้างแล้วลากย่อขยายตามที่ต้องการ

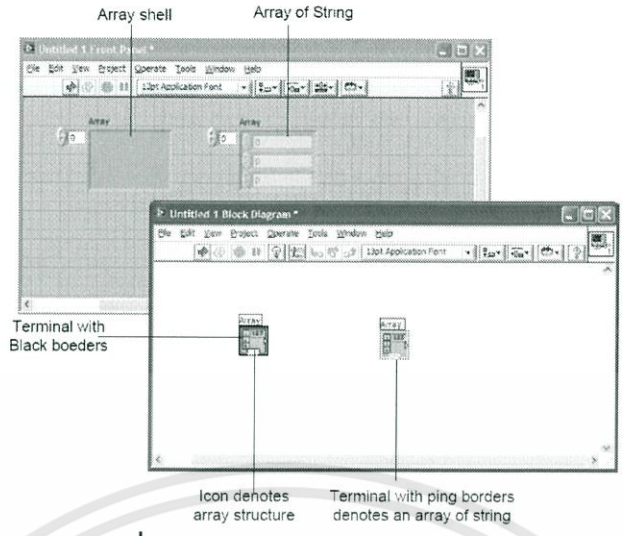
ครั้งแรกที่เราลากบล็อกของอาร์เรย์ไปวางในหน้าต่างฟร้อนท์พาเนลจะปรากฏบล็อกของ array shell ซึ่งจะเป็นแสดงชนิดของข้อมูลที่ไม่ทราบค่า ดังรูปที่ 2.80 เมื่อเราลาก string, Boolean หรือ numeric ไปวางในบล็อกดังกล่าวในหน้าต่างบล็อกไดอะแกรมจะสังเกตเห็นได้ว่าเทอร์มินอลจะ เปลี่ยนเป็นสีส้ม แสดงดังรูปที่ 2.81



รูปที่ 2.79 แสดงอาร์เรสร้างจาก Control และ Indicator



รูปที่ 2.80 ขั้นตอนการสร้างอาร์เรย์จาก Control และ Indicator

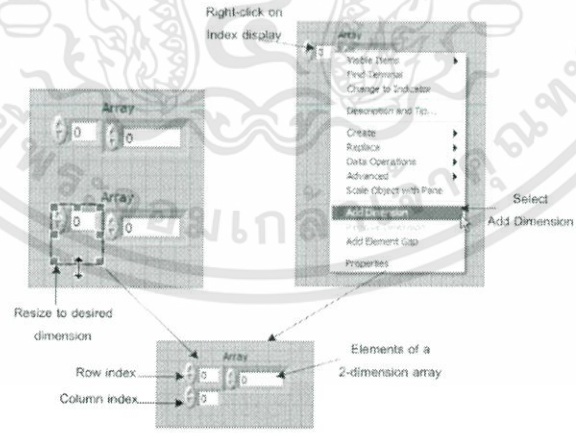


รูปที่ 2.81 เทอร์มินอลของ array shell

2.8.2 อาร์เรย์หลายมิติ

อาร์เรย์ 2 มิติ (2D) จะมีตัวบอกคุณลักษณะอยู่ 2 ตัว คือ บอกแถวและบอกหลัก อาร์เรย์ 3 มิติก็จะมีตัวบอกคุณลักษณะ 3 ตัว ดังนั้นอาร์เรย์ n มิติ ก็จะมีตัวบอกคุณลักษณะ n ตัว การเพิ่มมิติของอาร์เรย์สามารถทำได้ 2 วิธีคือ ใช้ Positioning Tool เพื่อเพิ่มมิติของอาร์เรย์และคลิ๊กเมาส์ขวาที่อาร์เรย์แล้วเลือกคำสั่ง Add Dimension จากเมนูย่อย ซึ่งวิธีการทั้งสองที่ใช้ในการเพิ่มมิติของอาร์เรย์ สามารถแสดงได้ดังรูปที่ 2.82

การลดมิติของอาร์เรย์ก็สามารถทำได้สองวิธีเช่นเดียวกันกับการเพิ่มคือ ใช้ Positioning Tool เพื่อลดขนาดมิติของอาร์เรย์และคลิ๊กเมาส์ขวาที่อาร์เรย์แล้วเลือกคำสั่ง Remove Dimension จากเมนูย่อย

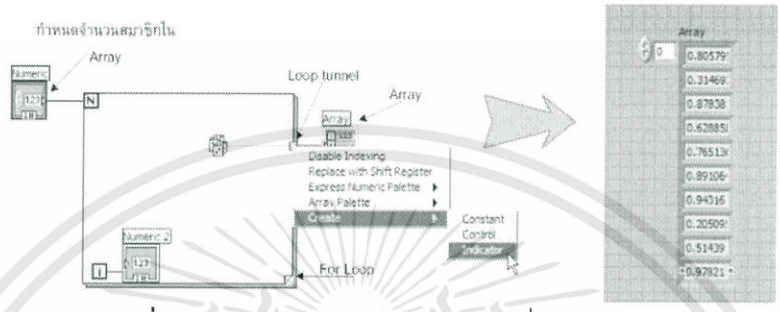


รูปที่ 2.82 แสดงวิธีการเพิ่มมิติของอาร์เรย์

2.8.3 การสร้างอาร์เรย์ด้วยลูป

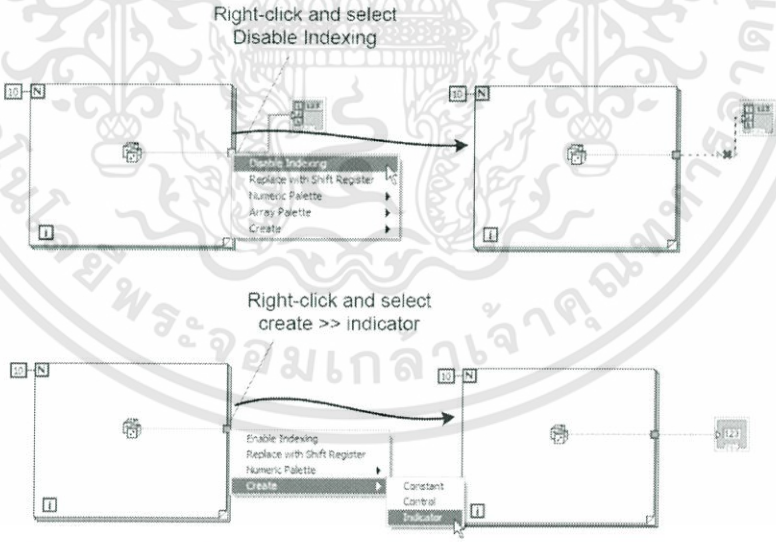
ในการสร้าง Array ด้วยลูปสามารถสร้างได้จาก For Loop และ While Loop ซึ่ง For Loop หรือ While Loop จะส่งข้อมูลของทุกลูปออกมาให้อัตโนมัติเรียกว่า Auto-indexing

ใน For Loop สามารถสร้างอาร์เรย์ได้โดยจำนวนสมาชิกของอาร์เรย์จะเท่ากับจำนวนลูปที่กำหนดไว้ การสร้างอาร์เรย์ 2 มิติโดยใช้ For Loop ทำได้โดยการใช้ For Loop ซ้อนกัน 2 ตัว For Loop วงนอกจะแทนจำนวนสมาชิกของแถวและ For Loop วงในจะแทนสมาชิกของหลัก วิธีการสร้างอาร์เรย์ด้วย For Loop แสดงดัง รูปที่ 2.83 ซึ่งเป็นวิธีการสร้างอาร์เรย์ด้วย For Loop เป็นอาร์เรย์ 1 มิติ ที่มีจำนวนสมาชิกในอาร์เรย์เท่ากับ 10



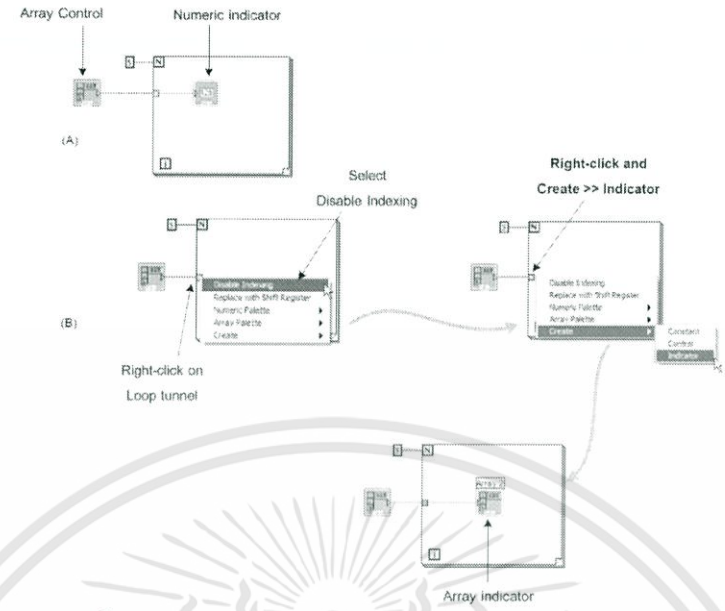
รูปที่ 2.83 การสร้างอาร์เรย์ด้วยฟังก์ชั่น For Loop

ถ้าเราไม่ต้องการที่จะสร้างอาร์เรย์สามารถทำได้โดยการคลิ๊กเมาส์ขวาที่ tunnel (สี่เหลี่ยมเล็กๆ อยู่บนเส้นขอบของฟังก์ชั่น For Loop เป็นทางผ่านของข้อมูล) แล้วเลือกคำสั่ง Disable» Indexing จากเมนูย่อย จากนั้นคลิ๊กขวาที่ tunnel เดิมแล้วเลือก Create» indicator จะปรากฏตัวแสดงผลที่เป็นแบบตัวเลขธรรมดา



รูปที่ 2.84 การเปลี่ยนแปลงการแสดงผลจากอาร์เรย์เป็นแบบ Numeric

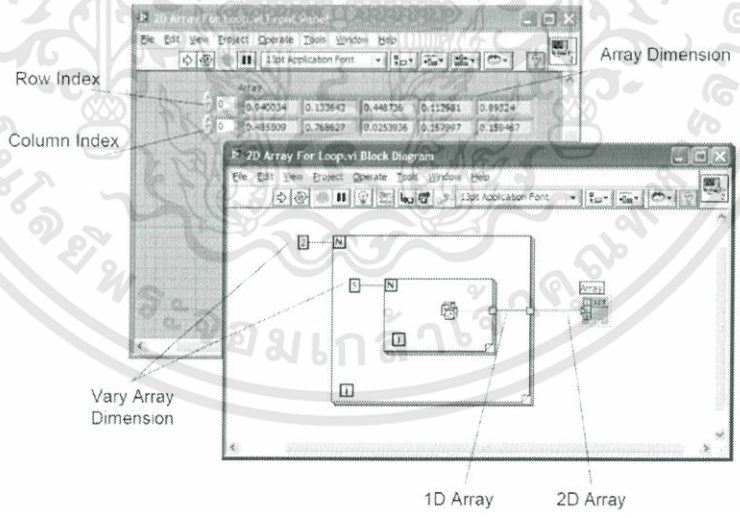
ถ้าเรามีตัวแสดงผลแบบ Numeric อยู่ภายในฟังก์ชั่น For Loop (ดังแสดงในรูปที่ 2.84) แล้ว ต้องการเปลี่ยนตัวแสดงผลให้เป็นอาร์เรย์สามารถทำได้โดยคลิ๊กเมาส์ขวาที่ tunnel แล้วเลือกคำสั่ง Disable» Indexing จากเมนูย่อย จากนั้นคลิ๊กขวาที่ tunnel เดิมแล้วเลือก Create» indicator ตัวแสดงผลจะแสดงผลในรูปแบบของอาร์เรย์ แสดงดังรูปที่ 2.85



รูปที่ 2.85 การต่อสายสัญญาณเมื่ออาร์เรย์อยู่ภายในลูป

2.8.3.1 การสร้างของอาร์เรย์ 2 มิติ

การสร้างอาร์เรย์ 2 มิติ สามารถสร้างจาก for Loop ได้โดยใช้ For Loop สองตัว โดยที่ For Loop ลูปนอกเป็นตัวสร้างแถวของอาร์เรย์และ For Loop ลูปในใช้สำหรับสร้างหลังของอาร์เรย์ อาร์เรย์ 2 มิติ ที่สร้างจาก For Loop แสดงได้ดังรูปที่ 2.86



รูปที่ 2.86 การสร้างอาร์เรย์ 2 มิติ ด้วย For Loop

2.8.4 ฟังก์ชันของอาร์เรย์

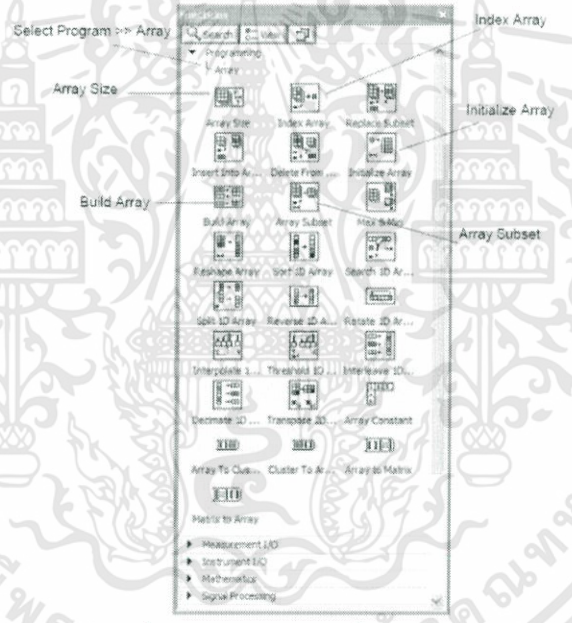
2.8.4.1 Array Size

ฟังก์ชั่น Array Size จะเป็นตัวบอกขนาดของอาร์เรย์ว่ามีขนาดใด เช่น ถ้าอาร์เรย์อินพุทเป็น อาร์เรย์ 1 มิติ (4 แถว 1 หลัก) อาร์เรย์เอาท์พุทจะเป็นอาร์เรย์ 1 มิติ มีจำนวน

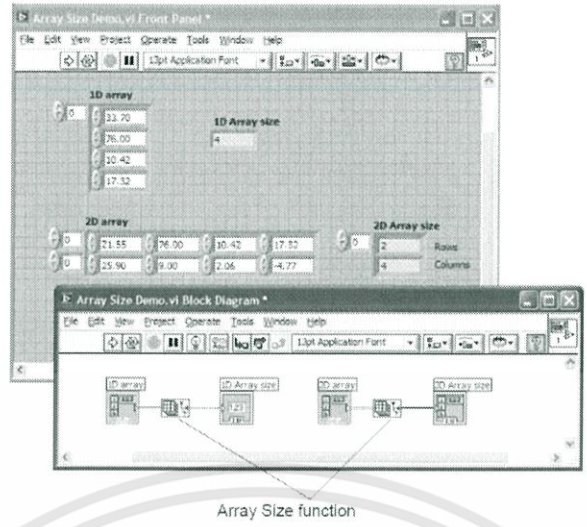
สมาชิกทั้งหมด 4 ตัว หรือถ้าอาร์เรย์อินพุทเป็นอาร์เรย์ 2 มิติ (2 แถว 4 หลัก) อาร์เรย์เอาท์พุทจะแสดงเป็นอาร์เรย์ 2 มิติ (แสดงจำนวนของแถวและหลัก) แสดงดังรูปที่ 2.87

2.8.4.2 Initialize Array

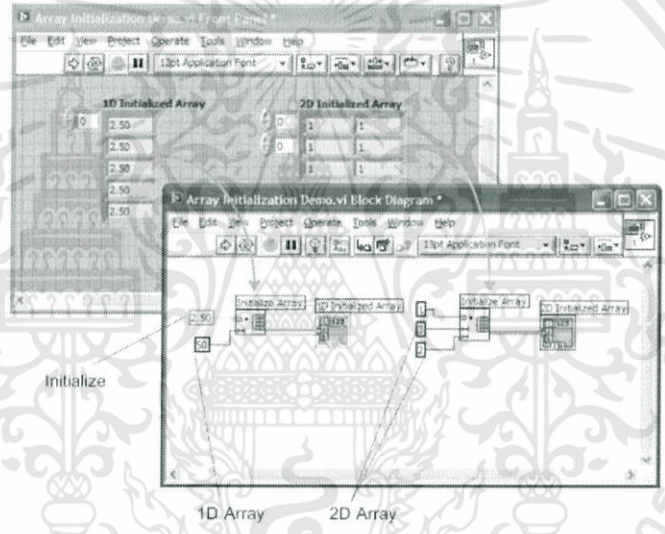
เป็นการสร้างอาร์เรย์ตามขนาดที่กำหนดพร้อมทั้งสามารถกำหนดค่าเริ่มต้นให้กับอาร์เรย์ที่สร้าง ได้อีกด้วย เช่น ต้องการสร้างอาร์เรย์ 1 มิติ มีจำนวนสมาชิกทั้งหมด 50 ตัว และกำหนดให้ค่าเริ่มต้นของอาร์เรย์เป็น 2.5 สามารถสร้างได้โดยคลิ๊กเมาส์ขวาที่เทอร์มินอล element ซึ่งอยู่ทางด้านบนซ้ายของ บล็อก Initialize Array แล้วเลือก create» constant และกำหนดค่าเริ่มต้นเป็น 2.5 ที่เทอร์มินอล dimension size ของบล็อก Initialize Array ที่อยู่ด้านล่างเทอร์มินอล element คลิ๊กเมาส์ขวาแล้ว เลือก create» constant และกำหนดให้เป็น 50 (จำนวนสมาชิก) ถ้าต้องการเพิ่มมิติของอาร์เรย์ก็สามารถทำได้โดยคลิ๊กเมาส์ขวาที่บล็อกของ Initialize Array แล้วเลือก Add dimension ซึ่งจะเพิ่มเทอร์มินอลเพื่อกำหนดขนาดของแถวและหลักแสดงดังรูปที่ 2.88



รูปที่ 2.86 แสดงฟังก์ชั่นอาร์เรย์



รูปที่ 2.87 แสดงการใช้งาน Array Size Function



รูปที่ 2.88 แสดงการใช้งาน Initialize Array Function

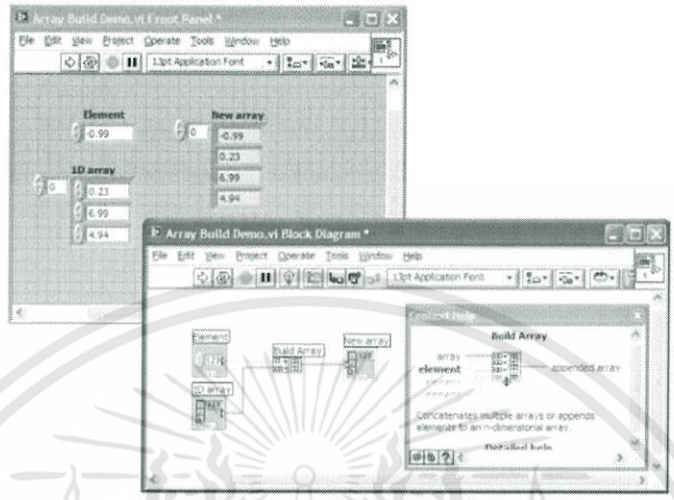
2.8.4.3 Build Array

เป็นการนำเอาอาร์เรย์หรือสมาชิกในอาร์เรย์หลาย ๆ ตัวมารวมกันแล้วเกิดเป็นอาร์เรย์ใหม่ ซึ่งอินพุทสามารถที่จะเป็นทั้งอาร์เรย์และเป็นจำนวนสเกลล่า โดยปกติแล้ว Build Array จะมีอินพุท 2 อินพุท ถ้าต้องการเพิ่มจำนวนอินพุทสามารถทำได้โดยการคลิ๊กเมาส์ขวาที่บล็อก Build Array แล้วเลือก Add Input หรือจะใช้ Positioning Tool คลิ๊กค้างแล้วลากขยายเพื่อเพิ่มอินพุทและถ้าต้องการลดจำนวนอินพุทก็สามารถทำได้โดยคลิ๊กเมาส์ขวา Build Array แล้วเลือก Remove Input ตัวอย่างการใช้งาน Build Array แสดงดังรูปที่ 2.89

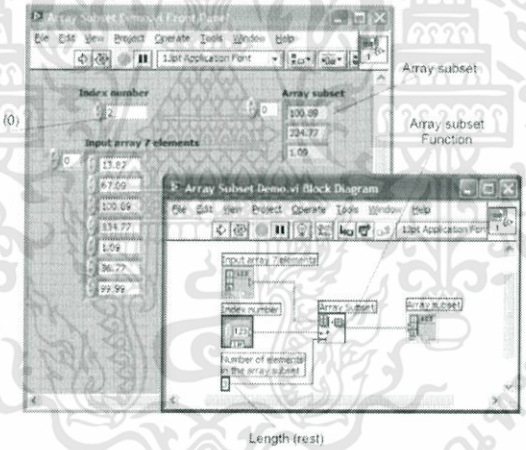
2.8.4.4 Array Subset

เป็นวิธีการเลือกค่าของข้อมูลที่อยู่ในอาร์เรย์ไปใช้งานโดยมีสิ่งที่ต้องกำหนดอยู่ 2 อย่างคือ Index (0) ซึ่งเป็นการกำหนดตำแหน่งเริ่มต้นที่เราต้องการ (ตำแหน่งแรกนับเป็นตำแหน่งที่ 0) และ Length (rest) เพื่อกำหนดจำนวนข้อมูลที่ต้องการ ดังตัวอย่างในรูปที่

2.90 อาร์เรย์ 1 มิติ มีจำนวนข้อมูล 7 ข้อมูล ต้องการใช้ข้อมูลตั้งแต่ตำแหน่งที่ 3 ถึง 5 ทำได้โดยกำหนด Index (0) ให้มีค่าเท่ากับ 2 และ Length (rest) ให้มีค่าเท่ากับ 3 แสดงดังรูปที่ 2.90



รูปที่ 2.89 แสดงการใช้งาน Build Array Function

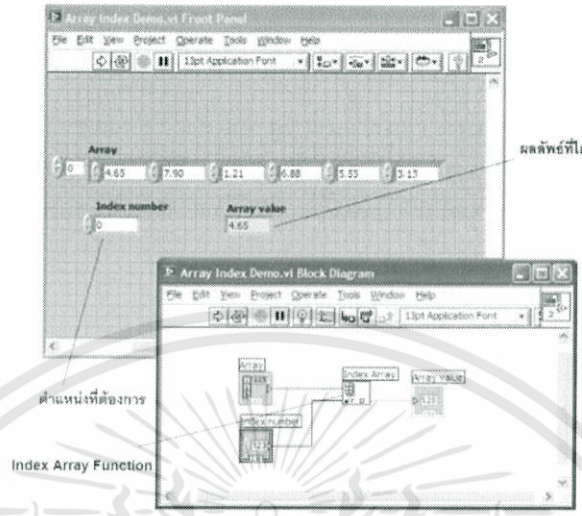


รูปที่ 2.90 แสดงการใช้งาน Subset Array Function

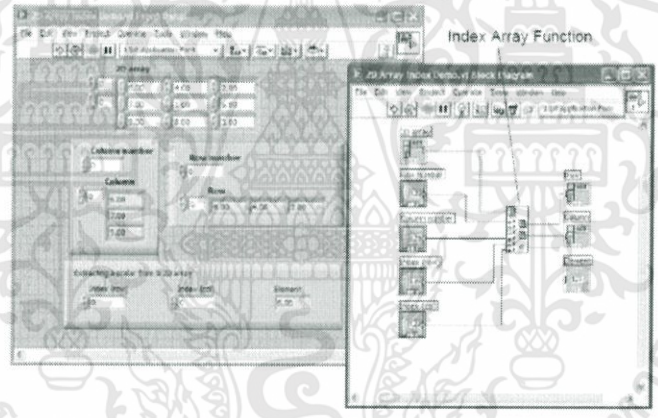
2.8.4.5 Index Array

ฟังก์ชั่น Index Array สามารถที่จะเลือกข้อมูลในอาร์เรย์ในตำแหน่งที่ต้องการมาใช้งานได้โดยการเลือกตำแหน่งของข้อมูลได้เลย ในกรณีที่เป็นอาร์เรย์ขนาด 2 มิติ ก็สามารถใช้ Index Array เลือกว่า จะเลือกข้อมูลออกเอาท์พุทที่ตำแหน่งแถวและหลัก แสดงดังรูปที่ 2.91 มีอาร์เรย์ 1 มิติ มีจำนวนข้อมูล ทั้งหมด 6 ข้อมูล ต้องการเลือกข้อมูลที่ 1 ไปใช้งานสามารถทำได้โดยกำหนดให้ Index Number มีค่า เท่ากับ 0 (ตำแหน่งแรกนับเป็นตำแหน่งที่ 0) และในรูปที่ 2.92 เป็นการใช้ฟังก์ชั่น Index Array ในการ เลือกข้อมูลจากอาร์เรย์ 2 มิติ ขนาด 3 x 3 โดย Index Array จะต้องมี 2 ตัวเพื่อเลือกตำแหน่งแถวและตำแหน่งหลังที่ต้องการ ซึ่งการเพิ่มอินพุทของ Index

Array สามารถทำได้โดยการใช้ Position Tool คลิ๊กที่ Index Array แล้วลากขยายเพื่อเพิ่มอินพุท ให้กับ Index Array



รูปที่ 2.91 แสดงการใช้งาน Index Array Function



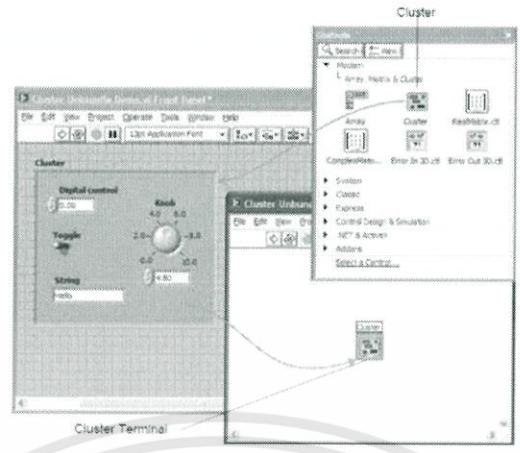
รูปที่ 2.92 แสดงการใช้งาน Index Array Function กับอาร์เรย์ 2 มิติ

2.9 พื้นฐานการใช้งานคลัสเตอร์

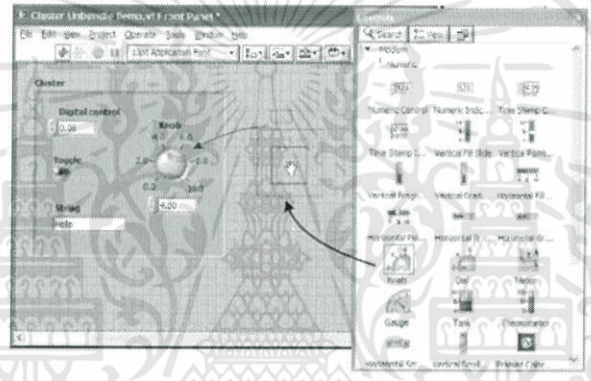
คลัสเตอร์เป็นเครื่องมือที่ทำหน้าจัดเก็บข้อมูลคล้ายกับอาร์เรย์เพียงแต่ว่าคลัสเตอร์สามารถเก็บข้อมูลที่ต่างชนิดกันได้ เช่นเก็บข้อมูลที่เป็น Numeric, Boolean และ String ไว้ด้วยกันได้ วิธีการเรียกใช้งานคลัสเตอร์ทำได้โดยเลือกที่ Controls Palette» Modern» Array & Cluster» Cluster

2.9.1 การสร้างคลัสเตอร์ด้วย Control และ Indicator

การสร้างคลัสเตอร์ด้วย Control และ Indicator นั้นสามารถที่จะกระทำได้เช่นเดียวกันกับการสร้างอาร์เรย์ แต่สำหรับการสร้างคลัสเตอร์จะแตกต่างจากการสร้างอาร์เรย์ตรงที่ข้อมูลในคลัสเตอร์ สามารถเป็นได้ทั้ง Numeric, String และ Boolean แต่ข้อมูลในอาร์เรย์จะเป็น Numeric เท่านั้น การใช้งานนั้นจะต้องมีฟังก์ชั่นที่ช่วยแยกชนิดของข้อมูลออกจากกันซึ่งจะได้กล่าวถึงในหัวข้อต่อไป



รูปที่ 2.93 แสดงการใช้งาน Cluster



รูปที่ 2.94 แสดงการสร้างคลัสเตอร์ด้วย Control และ Indicator

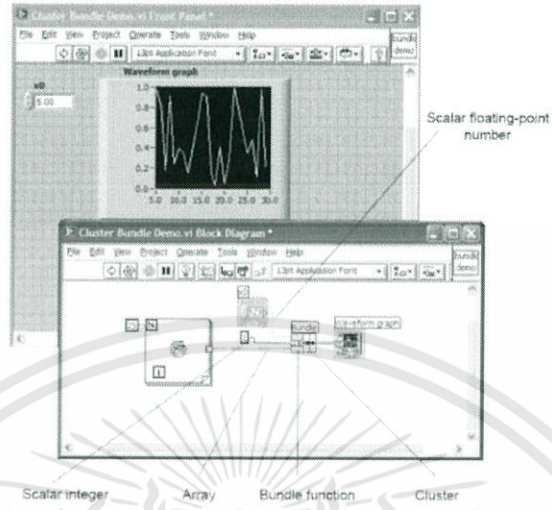
2.9.2 ฟังก์ชั่นของคลัสเตอร์

ฟังก์ชั่นที่ช่วยในการทำงานของคลัสเตอร์มีอยู่หลายแบบด้วยกันในที่นี้จะของกล่าวถึงฟังก์ชั่นที่ ถูกนำมาใช้งานบ่อยๆ 2 ฟังก์ชั่นคือ Bundle และ Unbundle ซึ่งวิที่เรียกใช้ฟังก์ชั่นทำโดยเลือกที่ Function Palette» All Function» Cluster แล้วเลือกฟังก์ชั่น Bundle หรือ Unbundle ตามต้องการ

2.9.2.1 Bundle Function

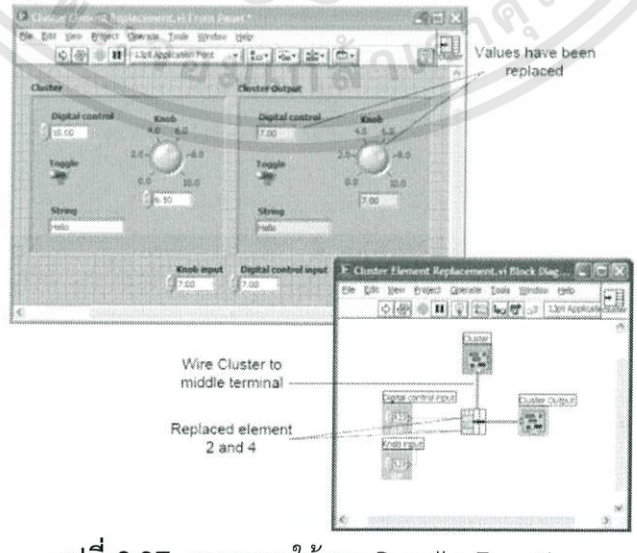
Bundle Function เป็นฟังก์ชั่นที่ใช้รวมเอาอินพุทหลายๆ ตัวมาสร้างเป็นคลัสเตอร์ใหม่ขึ้นมา โดยอินพุทนั้นสามารถเป็นได้ทั้ง ตัวเลขจำนวนจริง อาร์เรย์ หรือแม้กระทั่งคลัสเตอร์ เมื่อวางบล็อกของฟังก์ชั่น Bundle ลงไปในหน้าต่างบล็อกไดอะแกรมฟังก์ชั่น Bundle จะมีอินพุทเทอร์มินอล 2 เทอร์มินอลอยู่ทางซ้ายและเทอร์มินอลเอาท์พุท 1 เทอร์มินอลอยู่ทางขวาและอินพุทที่เป็นคลัสเตอร์จะ อยู่ด้านบนของบล็อก เราสามารถที่จะเพิ่มหรือลดจำนวนเทอร์มินอลอินพุทให้มีจำนวนเท่าที่เรา จำเป็นต้องใช้งานโดยใช้เมาส์คลิ๊กที่บล็อก แล้วเลื่อนขึ้นลงในแนวตั้งหรือใช้เมาส์ไปวางตรงเทอร์มินอล อินพุทแล้วคลิ๊กเมาส์ขวา แล้วเลือก Add Input ถ้าต้องการเพิ่มเทอร์มินอล และเลือก Remove Input ถ้าต้องการลดเทอร์มินอล

ในรูปที่ 2.94 จะเป็นตัวอย่างการสร้างคลัสเตอร์ด้วย Bundle โดยมีอินพุท 3 ชนิดคือ floating-point number, Scalar integer และ array ที่สร้างจาก For Loop และมีเอาท์พุทที่แสดงผลด้วยกราฟ



รูปที่ 2.95 แสดงการใช้งาน Bundle Function

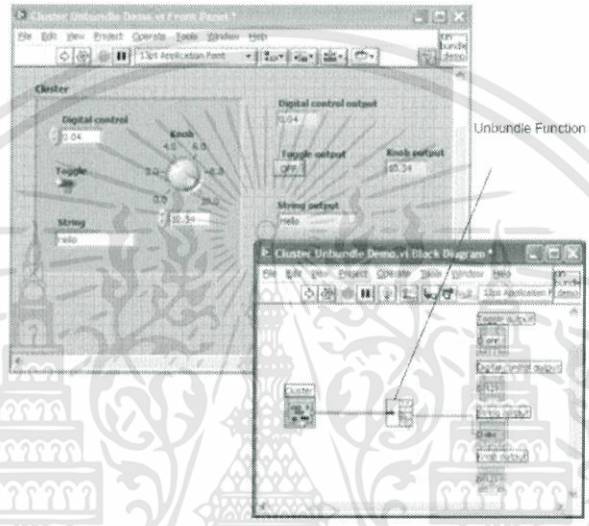
การเพิ่มเทอร์มินอลอินพุทของฟังก์ชั่น Bundle ทางซ้ายในขณะที่มีอินพุทที่เป็นคลัสเตอร์อยู่ ด้วยโดยไม่มีผลกระทบต่อโปรแกรมเดิม ดังตัวอย่างในรูปที่ 2.95 ซึ่งมีคลัสเตอร์ 2 ตัว หนึ่งตัวเป็นอินพุทและอีกหนึ่งตัวเป็นเอาท์พุท การทำงานของโปรแกรมคือเอาท์พุทจะทำงานแปรตามอินพุท ถ้าเราต้องการควบคุมเอาท์พุทโดยที่ไม่ต้องไปแก้ไขอินพุทเราสามารถที่จะเพิ่มอินพุทให้กับ Bundle และสร้างอินพุทให้กับ Bundle ให้ตรงกับเอาท์พุทที่เราต้องการควบคุมจากรูปที่ 2.96 การทำงานของโปรแกรมในขณะที่ มีอินพุทที่เป็นคลัสเตอร์เพียงอย่างเดียว เอาท์พุทจะแปรตามอินพุทแต่เมื่อเราเพิ่มอินพุทแบบ knop และ อินพุท numeric control เข้าไปที่เทอร์มินอลอินพุทของ Bundle เมื่อเราทดสอบโปรแกรมจะเห็นได้ว่า เมื่อเราเปลี่ยนค่าอินพุทแบบ knop และอินพุท numeric control ในคลัสเตอร์อินพุท ค่าภายในคลัสเตอร์เอาท์พุทจะไม่เปลี่ยนแปลงแต่เมื่อเราเปลี่ยนค่าอินพุทที่เราเพิ่มเข้าไปค่าเอาท์พุทก็จะเปลี่ยนตาม



รูปที่ 2.97 แสดงการใช้งาน Bundle Function

2.9.2.2 Unbundle Function

ในคลัสเตอร์จะประกอบไปด้วยชนิดของข้อมูลที่แตกต่างกัน Unbundle Function จะช่วยในการแยกข้อมูลอินพุทที่แตกต่างกันออกจากกันเพื่อจะนำเอาท์พุทนั้นไปใช้งานต่อไป ฟังก์ชั่นของ Unbundle ที่วางลงไปบนหน้าต่างบล็อกไดอะแกรมครั้งแรกจะมีอินพุท 1 อินพุท และมีเอาท์พุท 2 อินพุท เอาท์พุท Unbundle จะเปลี่ยนไปตามอินพุทที่เป็นคลัสเตอร์ ตัวอย่างเช่น มีคลัสเตอร์อินพุทที่มีชนิดของข้อมูลแตกต่างกัน 4 ชนิด เมื่อต่อสายของคลัสเตอร์เข้ากับอินพุทของ Unbundle เอาท์พุทของ Unbundle จะเปลี่ยนเป็น 4 เอาท์พุทโดยอัตโนมัติ



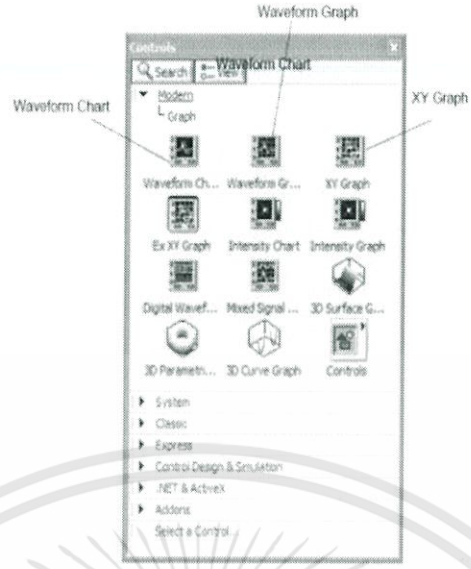
รูปที่ 2.98 แสดงการใช้งาน Unbundle Function

2.10 พื้นฐานการใช้งาน Charts และ Graph

2.10.1 Waveform Charts

การแสดงข้อมูลในลักษณะของ Chart เป็นการแสดงการพล็อตรูปคลื่นสัญญาณข้อมูลแบบหนึ่งที่สามารถพล็อตข้อมูลที่เป็นสเกลาร์และเป็นอาร์เรย์ได้ การเรียกใช้งาน คือเลือกที่ Controls Palette » Graph » Waveform Chart

Waveform Chart ในโปรแกรม LabVIEW มีเพียงชนิดเดียวแต่มีคุณสมบัติในการแสดงข้อมูลที่ แตกต่างกัน 3 อย่างคือ Strip Chart, Scope Chart และ Sweep Chart ซึ่งข้อแตกต่างของ Chart ทั้ง 3 แบบ อธิบายได้ดังรูปที่ 2.99

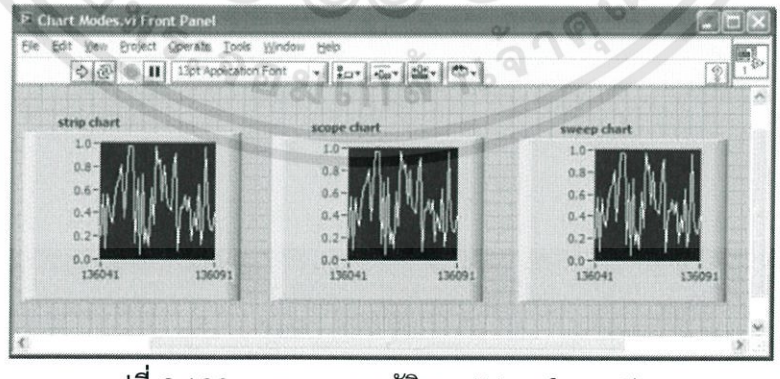


รูปที่ 2.99 แสดงวิธีการเรียกใช้งาน Waveform Chart

Strip Chart มีคุณสมบัติในการพล็อตข้อมูลไปจนกว่าจะหยุดพล็อตซึ่งจะสังเกตเห็นว่าเส้นของ สัญญาณข้อมูลที่พล็อตนั้นจะเลื่อนจากขวาไปซ้ายต่อเนื่องกันไป

Scope Chart คุณสมบัติของการพล็อตแบบนี้คือจะแบ่งเป็นช่วงของข้อมูลโดยการกำหนดที่ แกนเวลาเมื่อโปรแกรมทำการพล็อตข้อมูลไปถึงค่าสุดท้ายของช่วงแรกแล้วการพล็อตข้อมูลของช่วงต่อไป โปรแกรมจะทำการลบกราฟในช่วงแรกไปแล้วเริ่มการพล็อตใหม่โดยเวลาเริ่มต้นจะนับต่อจากค่าสุดท้าย ของช่วงแรก

Sweep Chart จะทำงานคล้ายกับ Scope Chart แต่เมื่อเริ่มทำการพล็อตกราฟในช่วงต่อไป เส้นกราฟในช่วงแรกจะไม่ถูกลบจะทำงานในลักษณะพล็อตทับเส้นกราฟเดินและจะมีเส้นสีแดงในแนว เลื่อนจากซ้ายไปขวาเพื่อแสดงถึงจุดการพล็อตกราฟครั้งล่าสุด คุณสมบัติทั้ง 3 ของ Waveform Chart แสดงดังรูปที่ 2.100 เป็นการแสดงค่าของข้อมูลที่เก็บอยู่ในอาร์เรย์เทียบกับเวลาที่รับเข้ามา

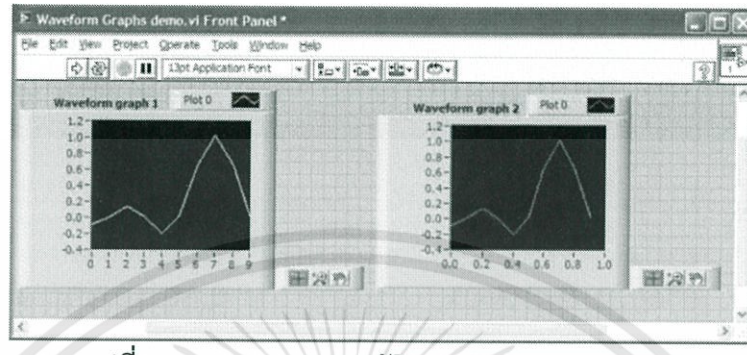


รูปที่ 2.100 แสดงคุณสมบัติของ Waveform Chart

2.10.2 Waveform Graphs

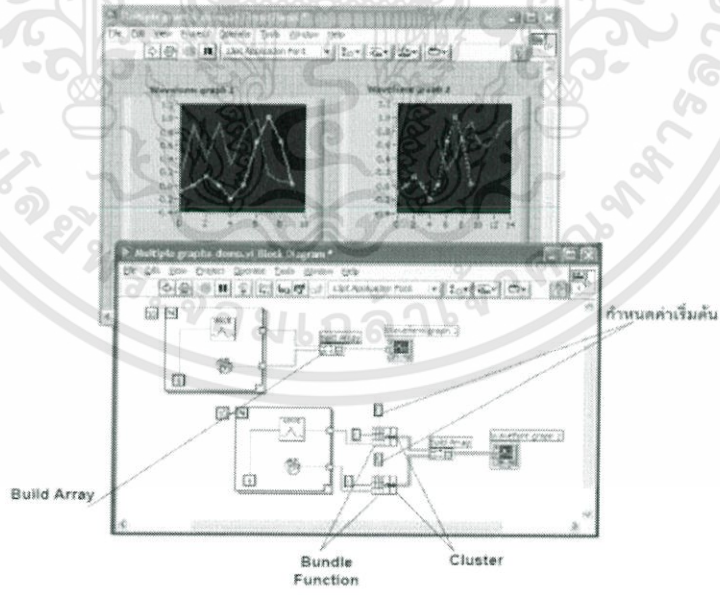
แตกต่างจาก Waveform Chart คือการพล็อตข้อมูลด้วย Waveform Graphs จะมีการกำหนด จำนวนของช่วงข้อมูลอย่างชัดเจนจำนวนข้อมูลทั้งหมดที่อยู่ในช่วงของกราฟจะถูกแสดง

ออกมาทั้งหมด การพล็อตกราฟหลายเส้นทำได้โดยการนำอาร์เรย์แต่ละตัวมาต่อรวมเพื่อสร้างอาร์เรย์ใหม่จำนวน เส้นกราฟที่แสดงจะเท่ากับมิติของอาร์เรย์ การเรียกใช้งาน Waveform Graphs ทำได้คือ เลือกที่ Controls Palette» Modern» Graph» Waveform Graphs ลักษณะของการพล็อตกราฟลงบน waveform graph แสดงได้ดังรูปที่ 2.101



รูปที่ 2.101 แสดงคุณสมบัติของ Waveform Graph

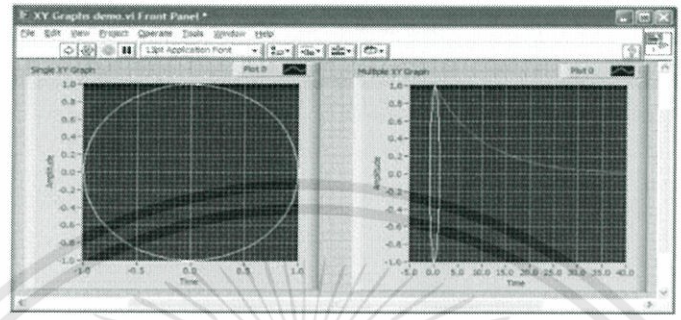
นอกจากนี้แล้วเรายังสามารถใช้ Bundle และ Cluster มาช่วยในการกำหนดค่าเริ่มต้นและ ความถี่ในการแสดงข้อมูลแสดงดังรูปที่ 2.102 กราฟแรกจะแสดงการสร้างสัญญาณ 2 เส้นให้อยู่บนกราฟ เดียวกันโดยใช้ฟังก์ชั่น Build Array เข้ามาช่วย กราฟที่สองจะเป็นการสร้างสัญญาณ 2 เส้นให้อยู่บน กราฟเดียวกันและให้สามารถกำหนดค่าเริ่มต้นของกราฟแต่ละเส้นได้ ซึ่งเราจะนำเอา Bundle และ Cluster โดยกราฟเส้นแรกให้เริ่มต้นที่ค่า 0 แสดงค่าทั้งหมด 10 ค่า และอีกเส้นให้เริ่มต้นที่ 5 และค่าที่ แสดงทั้งหมด 10 ค่า เช่นกัน ดังแสดงในรูปที่ 2.102



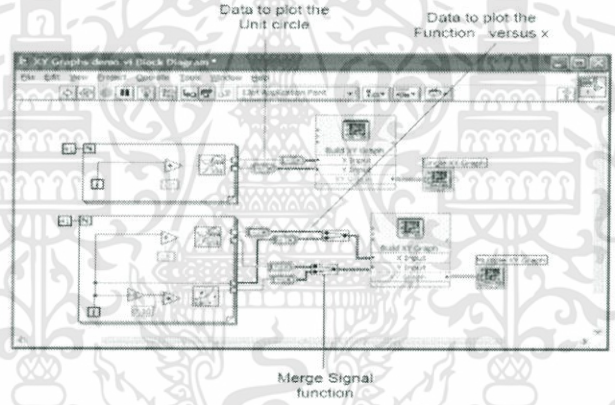
รูปที่ 2.102 แสดงคุณสมบัติของ Waveform Graph

2.10.3 XY Graphs

การพล็อตกราฟใน XY Graph เป็นการนำค่าของอาร์เรย์ 2 ชุดมาพล็อตในระนาบ XY โดย กำหนดให้อาร์เรย์ตัวที่ 1 และตัวที่ 2 เป็นค่าที่แสดงในแกน X และแกน Y ตามลำดับ โดยจะมีอุปกรณ์ที่ชื่อ Build XY Graph จะช่วยในการแปลงข้อมูลในอาร์เรย์ให้เป็นค่าใด ๆ ที่สามารถพล็อตในแกน XY ได้ แสดงวิธีการพล็อต XY Graph ดังรูปที่ รูปที่ 2.103



(a) Front Panel



(b) Block Diagram

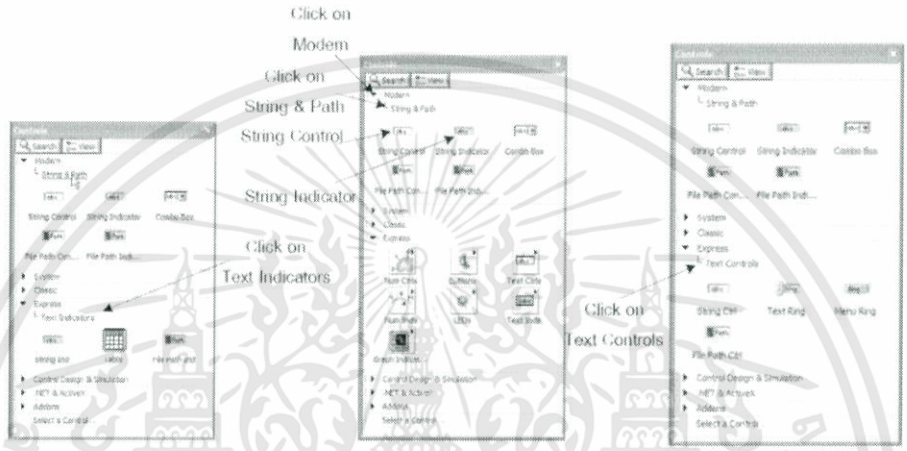
รูปที่ 2.103 แสดงคุณสมบัติของ XY Graph

2.11 พื้นฐานเกี่ยวกับ String

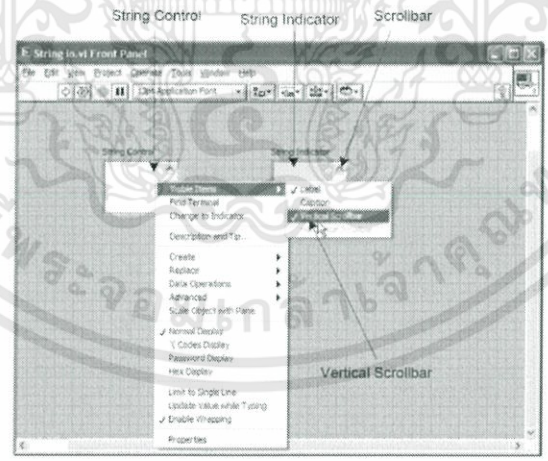
สตริงเป็นชนิดของข้อมูลที่อยู่ในรูปแบบของตัวอักษร ซึ่งสามารถที่จะกำหนดให้แสดงหรือไม่แสดงค่าก็ได้หลายครั้งที่สตริงถูกใช้ในลักษณะของรหัส ASCII ในโปรแกรม LabVIEW สตริงจะถูกใช้ใน instrument control การใช้งานข้อมูลชนิดสตริงและข้อมูลชนิดตัวเลขจะต้องมีการเปลี่ยนชนิดของข้อมูลเพื่อให้ชนิดของข้อมูลเป็นชนิดเดียวกันก่อนซึ่งก็จะมีวิธีการเปลี่ยนชนิดของข้อมูลซึ่งจะได้กล่าวในหัวข้อต่อไป ชนิดของข้อมูลก่อนที่จะบันทึกลงไฟล์บนฮาร์ดดิสก์จะต้องแปลงให้เป็นข้อมูลชนิดสตริงก่อน ในหัวข้อนี้จะพูดถึงสตริงและการใช้ไฟล์อินพุทไฟล์เอาท์พุท (File I/O)

การใช้งาน String Control และ String Indicator ที่อยู่ใน String & Path ซึ่งเป็นพาเลทย่อยของ Control» Modern ซึ่งแสดงในรูปที่ 2.103 ซึ่งเราสามารถเลือกใช้งานได้ทั้งแบบ Text Controls และ Text Indicators ซึ่งอยู่ในพาเลทย่อยของ Controls»Express การใช้งาน String Control และ String Indicator ทำได้โดยการคลิ๊กเมาส์ทางซ้ายค้างที่สัญลักษณ์ของ String Control

และ String Indicator แล้วลากไปวางที่หน้าต่างฟร้อนท์พาเนล ซึ่งเราจะเห็นลักษณะของ String Control และ String Indicator เป็นบล็อกสี่เหลี่ยมสำหรับรับข้อมูลอินพุทและแสดงผลตามลำดับ ถ้าบล็อกนั้นมีขนาดเล็กไม่เพียงพอสำหรับข้อมูลที่ป้อนเข้าไปก็สามารถขยายได้ใช้เมาส์คลิ๊กแล้วลากขยายให้ใหญ่ขึ้นหรือถ้าขนาดของฟร้อนท์พาเนลมีขนาดจำกัดซึ่งทำให้เราไม่สามารถขยายขนาดของบล็อกสตริงได้ก็สามารถเพิ่ม Scrollbar เข้าไปเพื่อให้สามารถกรอกข้อมูลชนิดสตริง ได้เป็นจำนวนมากขึ้น ซึ่งทำได้โดยคลิ๊กเมาส์ ทางขวาที่บริเวณบล็อกของ String Control หรือ String Indicator ที่ต้องการเพิ่ม Scrollbar แล้วเลือก Visible Item» Vertical Scrollbar ดังรูปที่ 2.104



รูปที่ 2.104 การเข้าถึง String controls และ String Indicators



รูปที่ 2.105 การเพิ่ม Scrollbar ให้กับบล็อก String Indicator และ String Control

สกอร์บาร์ที่เพิ่มเข้าไปใน String Control หรือ String Indicator ดังแสดงในรูปที่ 2.105 สกอร์บาร์ที่เพิ่มเข้าไปแสดงอยู่ในแนวตั้ง

การแสดงข้อมูลในบล็อกของ String Control และ String Indicator นั้นสามารถกำหนดให้แสดงได้ 4 รูปแบบ คือ

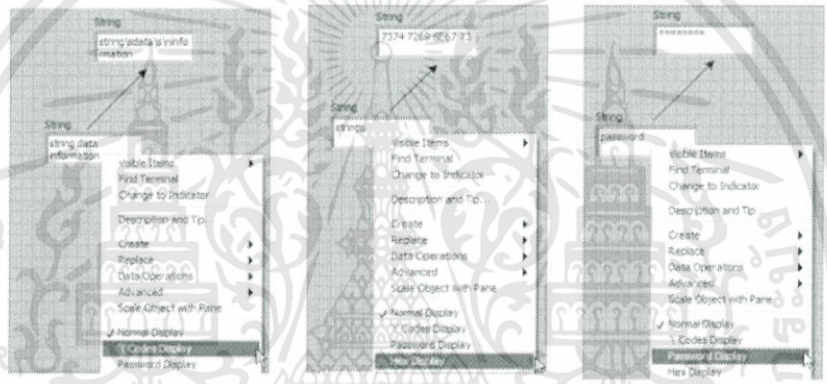
1. Normal Display จะแสดงข้อความเป็นแบบปกติตามที่เราป้อนเข้าไป

2. “\” Codes Display จะแสดงข้อความในลักษณะของ Codes โดยมีเครื่องหมาย “\” เป็น ตัวขั้นแทนการเว้นวรรค

3. Password Display จะแสดงข้อความแบบรหัสผ่านแทนด้วย “*”

4. Hex Display จะแสดงข้อความในลักษณะของเลขฐาน 16

โดยที่ลักษณะของข้อความหลังจากที่ถูกเข้ารหัสในรูปแบบต่างๆ แล้วสามารถแสดงให้เห็นได้ ดังรูปที่ 2.105 ซึ่งในรูปจะแสดงวิธีการเข้ารหัสแบบ “\” Codes Display, Hex Display และ Password Display ตามลำดับ การกำหนดรูปแบบของชนิดของข้อมูลแบบสตริงว่าจะให้แสดงอยู่ในรูปแบบใดนั้นสามารถทำได้โดยคลิ๊กเมาส์ทางขวาบริเวณบล็อกของ String Control และ String Indicator จะปรากฏเมนูย่อยขึ้นมาซึ่งในเมนูย่อยนี้จะมีชุดคำสั่งสำหรับกำหนดรูปแบบการแสดงผลอยู่ ทั้งหมด 4 รูปแบบ



รูปที่ 2.106 แสดงตัวอย่างการเข้ารหัสแบบต่างๆ

ถ้าต้องการให้แสดงผลของข้อมูลในรูปแบบใดก็ใช้เมาส์คลิ๊กที่คำสั่งนั้น รูปแบบที่แสดงผลก็จะเปลี่ยนไปตามรูปแบบที่เลือกแสดงได้ดังรูปที่ 2.106

ในตารางที่ 2.1 จะเป็นตารางแสดงสัญลักษณ์และความหมายของข้อมูลเมื่อเราทำการแสดงข้อมูลแบบโค้ด

ตารางที่ 2.1 แสดง Backslash Codes

Code	G Interpretation
\00 - \FF	Hex value of an 8-bit character; must be uppercase
\b	Backspace (ASCII BS,equivalent to \08)
\f	Form feed (ASCII FF, equivalent to \0C)
\n	Line feed (ASCII LF, equivalent to \0A)
\r	Carriage return (ASCII CR, equivalent to \0D)
\t	Tab (ASCII HT, equivalent to \09)
\s	Space (equivalent to \20)
\\	Backslash (ASCII \, equivalent to \5C)

2.11.1 การสร้างสตริง

ในหัวข้อนี้จะยกตัวอย่างการใช้งานสตริง ซึ่งจะยกตัวอย่างการใช้งานทั้งหมด 2 แบบ คือ

- Format Into String
- Scan From String

ในโปรแกรม LabVIEW สามารถที่จะสร้างโปรแกรมจำลองการทำงานของดิจิตอลมัลติมิเตอร์ (DMM) โดยเริ่มจาก เปิดหน้าต่างฟร้อนท์พาเนลของโปรแกรม LabVIEW จากนั้นวาง String control ลงไป 2 ตัว, numeric control 1 ตัว, และ String Indicator อีก 1 ตัว จากนั้นเลือกหน้าต่างบล็อกไดอะแกรมขึ้นมาแล้วเลือก Format Into String จากพาเลทย่อยของ String แสดงดังรูปที่ 2.107 สำหรับตัวอย่างนี้จะแสดงการแปลงข้อมูลที่เป็นตัวอักษรและการนำข้อมูลอินพุทมาเชื่อมต่อกันเพื่อกำหนดรูปแบบคำสั่งสำหรับดิจิตอลมัลติมิเตอร์ (DMM) โดยใช้ Positioning Tool ในการขยายให้ Format Into String เพิ่มเป็น 2 อินพุท ซึ่ง input connector จะอยู่ทางด้านล่างซ้ายมือของบล็อก Format Into String จากนั้นลากสายสัญญาณเชื่อมต่อระหว่างบล็อกอินพุทกับบล็อก Format Into String โดยที่ต่ออินพุทตัวแรกเข้ากับ numeric control และต่ออินพุทตัวที่ 2 เข้ากับ String control ชนิดของสัญญาณอินพุทสำหรับ Format Into String จะเปลี่ยนเป็นแบบ DBL หรือ abc แบบอัตโนมัติ จากนั้นต่อ String control เข้ากับอินพุท initial string ที่อยู่ทางด้านบนขวาของบล็อก Format Into String ซึ่งจะกำหนดให้เป็นชุด DMM Command และสุดท้ายให้ต่อ resulting string เข้ากับ string indicator ซึ่งเป็นชุด Command คำสั่งที่จะส่งไปยัง DMM

รูปแบบของคำสั่งสามารถกำหนดรูปแบบของคำสั่งได้โดยคลิ๊กเมาส์ทางขวาที่ connector Format string ซึ่งอยู่ด้านบนตรงกลางของบล็อก Format Into String แล้วเลือก create» constant ในเมนูย่อย ซึ่งจะแสดงบล็อกของค่าคงที่เชื่อมต่อกับ string format การกำหนดรูปแบบของ string สามารถทำได้โดย คลิ๊กเมาส์ทางขวาที่บล็อกของ Format Into String แล้วเลือกคำสั่ง Edit Format String จะปรากฏไดอะล็อกบล็อกขึ้นมาสำหรับเพิ่ม ลด และแก้ไข รูปแบบของ string ซึ่งในไดอะล็อกบล็อกนี้มีจุดสำคัญอยู่ 3 จุด คือ Current format sequence, selected operation (example) และ corresponds format string (ดังแสดงในภาพ) ซึ่งในแต่ละจุดสามารถอธิบายได้ดังนี้

1. Current format sequence : แสดงจำนวนอินพุทของ Format Into String และระบุด้วยว่าแต่ละอินพุทรูปแบบของ string เป็นแบบใด

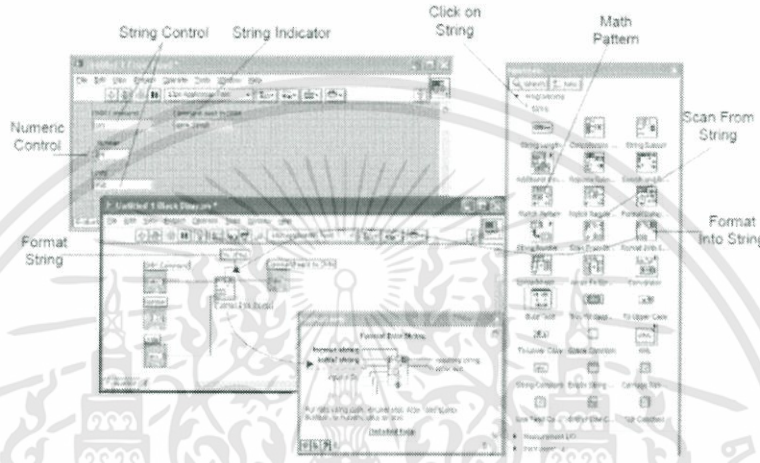
2. Selected operation (example) : จะบรรจุรูปแบบต่างๆ ของ string ซึ่งเราสามารถเลือกรูปแบบที่ต้องการได้โดยใช้เมาส์เลือกที่รูปแบบที่ต้องการ

3. Corresponds format string : จะแสดงสัญลักษณ์ของรูปแบบของ string เช่น %s %2f ซึ่งจะแสดงต่อกันตามจำนวนของอินพุท

หลังจากที่กำหนดค่าต่างๆ เสร็จเรียบร้อยแล้วให้กลับไปที่หน้าต่างฟร้อนท์พาเนล ลองป้อนข้อความลงไปในบล็อกต่างๆ ดังตัวอย่างในรูปที่ 2.107 ซึ่งป้อนคำสั่ง set ลงในช่อง DMM Command ในบล็อก Number ให้ใส่ตัวเลขใดก็ได้ลงไป (ในที่นี้เป็นเลข 4) และในบล็อก Units ให้ใส่

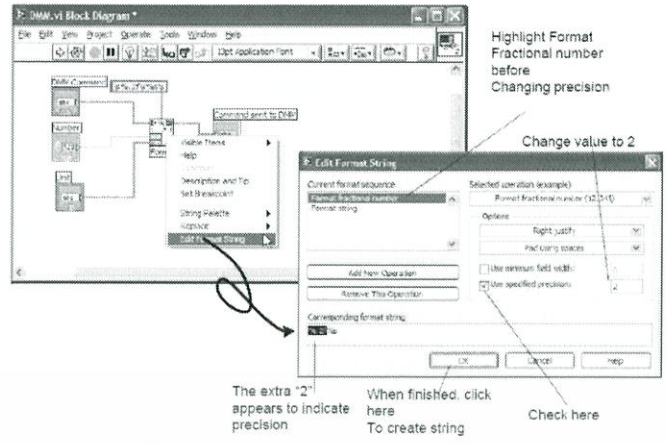
ข้อความเป็นตัวอักษร ในที่นี้จะใส่หน่วยของแรงดันซึ่งมีหน่วยเป็น โวลท์ (volt) หลังจากที่กรอกข้อมูลต่างๆ

เสร็จเรียบร้อยแล้วขั้นตอนต่อไปคือการรับโปรแกรมโดยกดที่ปุ่มรัน สังเกตที่บล็อกแสดงผลข้อมูล ซึ่งจะเห็นได้ว่าโปรแกรมจะทำการนำเอาข้อมูลอินพุททั้ง 3 ตัวมาจัดการแพ็คข้อมูลเพื่อจะส่งไปยัง DMM โดยการจัดเรียงข้อมูลจะเริ่มจากการนำข้อมูลของบล็อก DMM Command ตามด้วยข้อมูลจากบล็อก Number และปิดท้ายด้วยข้อมูลจากบล็อก Unit ดังที่แสดงในรูปที่ 2.106

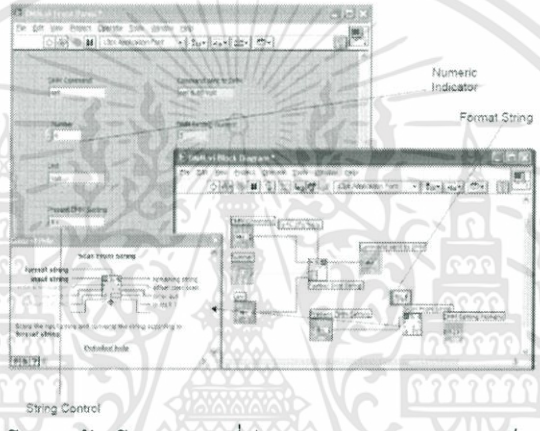


รูปที่ 2.107 โครงสร้างโปรแกรมที่มี Format into String เป็นส่วนประกอบ

ในตัวอย่างต่อไปต้องการที่เพิ่มฟังก์ชั่นการทำงานของโปรแกรมให้มากขึ้นโดยจะแสดงวิธีการ เปลี่ยนจากคุณลักษณะของสัญลักษณ์ เช่น 0 ถึง 9, +, -, e, E เป็นต้น ให้เป็นตัวเลขดังแสดงในรูปที่ 4.142 วิธีการก็คือในหน้าต่างฟร้อนท์พาเนลให้เพิ่มบล็อกของ string control และ string indicator จากนั้นกลับไปที่หน้าต่างบล็อกไดอะแกรมแล้วเลือกฟังก์ชั่นของบล็อก Scan Format String จาก Programming »String พาเลท ดังแสดงในรูปที่ 2.108 ก่อนที่จะเชื่อมต่อบล็อก Scan Format String เข้ากับ string control และ string indicator ให้ทำการดับเบิ้ลคลิ๊กที่บล็อก Scan Format String (หรือ คลิ๊กเมาส์ขวาที่บล็อกแล้วเลือก Edit Format String จากเมนูย่อย) จะปรากฏไดอะล็อกล็อกขึ้นมา โดย ในบล็อกของ current scan sequence จะแสดงรูปแบบของสตริงเป็นแบบ Scan Number และบล็อก corresponds scan string จะแสดงสัญลักษณ์ %f จากนั้นให้กดปุ่ม OK จากนั้นต่อสายสัญญาณอินพุท เข้ากับ string control และต่อเอาท์พุทเข้ากับ string indicator จากนั้นทำการทดสอบโปรแกรม โดย การลองป้อนคำสั่งที่บล็อก Present DMM Setting ยกตัวอย่างเช่น 5 V จากนั้นทำการรันโปรแกรม สังเกตที่บล็อกแสดงผลจะแสดงเฉพาะเลข 5 เท่านั้น นั่นแสดงว่าเกิดจากการที่เราเลือกรูปแบบของ string เป็นแบบ Scan Number



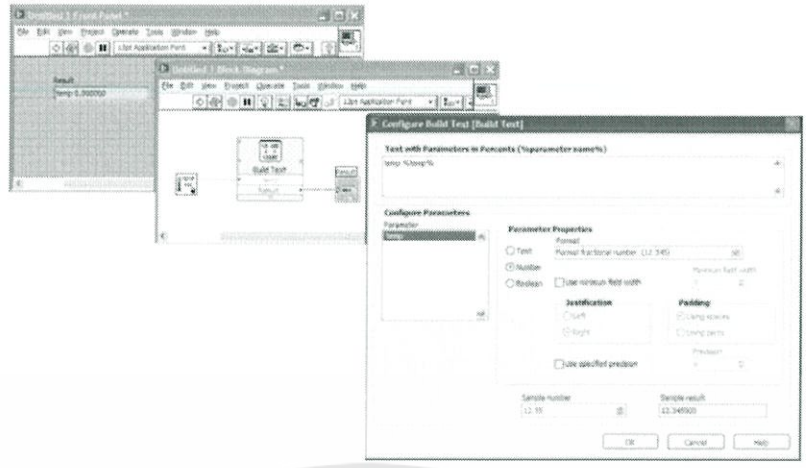
รูปที่ 2.108 การกำหนดรูปแบบของสตริง



รูปที่ 2.109 โครงสร้างโปรแกรมที่มี Scan From String เป็นส่วนประกอบ

2.11.2 การแปลงค่า Numeric เป็น String โดยใช้ Express VI

การแปลงข้อมูลแบบ Numeric ไปเป็น String นั้นมีอีกหนึ่งวิธีสำหรับการแปลงข้อมูลคือการใช้ Express VI โดยเลือกได้จากFunction» Program» String พาเลทหรือจะเลือกจาก Function» Express» Output พาเลทก็ได้เช่นกัน ดังตัวอย่างที่แสดงในรูปที่ 2.110 ถ้าข้อมูลอินพุทไม่ได้เป็นแบบ สตริง Express VI จะเป็นตัวที่ทำการแปลงข้อมูลนั้นให้เป็นแบบสตริงโดยการกำหนดค่าฟังก์ชั่นภายใน Express VI



รูปที่ 2.110 แสดงการแปลงค่า Numeric เป็น String โดยใช้ Express VI

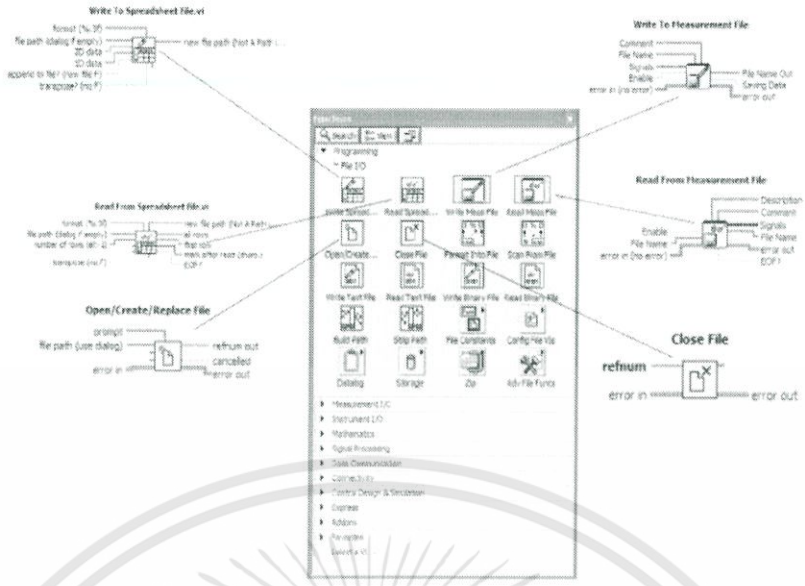
2.12 พื้นฐานเกี่ยวกับ File I/O

ไฟล์ I/O ทำหน้าที่จัดการข้อมูลไปยังไฟล์ต่างๆ โดยใช้ File I/O Vis และฟังก์ชั่นต่างๆ ที่มีอยู่ใน File I/O พาเลท ซึ่งรูปแบบของไฟล์ต่างๆ ที่อยู่ใน File I/O พาเลท มีดังนี้

- Opening and closing data files.
- Reading data from and writing data to files.
- Reading from and writing to spreadsheet-formatted files.
- Moving and renaming files and directories.
- Changing file characteristics.

ไฟล์ I/O พาเลทที่แสดงในรูปที่ 2.110 จะประกอบไปด้วย Vis และฟังก์ชั่นสำหรับการออกแบบ การทำงานของไฟล์ I/O เช่น writing to หรือ reading from ซึ่งเป็นไปตามชนิดของข้อมูลดังนี้

- Numeric values to or from spreadsheet text files
- Characters (strings) to or from text files
- Lines from text files
- Data to or from binary files



รูปที่ 2.111 ส่วนประกอบของ File I/O Palette

ชนิดของไฟล์ I/O แบ่งตามฟังก์ชั่นการทำงานได้ดังนี้

1. Create or Open a file:
2. Read from or write to the file
3. Close the file

ในรูปที่ 2.111 ฟังก์ชั่นของไฟล์ I/O ที่อยู่ในแถวบนสุดทั้ง 4 ฟังก์ชั่น สามารถ กำหนดให้ฟังก์ชั่นแต่ ละฟังก์ชั่นทำงานได้ในสามขั้นตอน ดังนี้

- Write to Spreadsheet File: เปลี่ยนอาร์เรย์ 2 มิติ หรือ 1 มิติ ของ single-precision number ให้เป็นข้อความตัวอักษร (string), จากนั้นทำการเขียน string ให้เป็นไบท์สตรีม และสุดท้ายเพิ่ม ส่วนปิดท้ายขอมูลแบบ string แล้วส่งไปยังไฟล์ที่ต้องการ
- Read Form Spreadsheet File: อ่านข้อมูลจากบรรทัดหรือจากแถวของไฟล์ข้อมูลแบบ numeric, กำหนดให้ข้อมูลที่อ่านมาเป็นตัวอักษร, แปลงข้อมูลให้เป็นอาร์เรย์ 2 มิติ แบบ single-precision
- Write To Measurement File: เขียนข้อมูลไปยัง measurement file แบบ text-base (.lvm) หรือ เขียนแบบไบนารี่ (.tdm)
- Read From Measurement File: อ่านข้อมูลจาก measurement file ที่เขียนข้อมูลแบบ text-base (.lvm) หรือ measurement file ที่เขียนข้อมูลแบบไบนารี (.tdm)

ในFile I/O พาเลทมีการเพิ่มฟังก์ชั่นการควบคุมให้ฟังก์ชั่นไฟล์ I/O แต่ละตัวทำงานแยกกัน โดยใช้ฟังก์ชั่น create file, open file, read data from, write data from และ close file ซึ่งเราสามารถใช้ฟังก์ชั่นต่างๆ ที่ได้กล่าวมาให้ทำงาน เช่น

- Create directories
- Move, copy or delete files
- List directory contents
- Change file characteristics
- Manipulate paths

ตัวอย่างการใช้งานฟังก์ชั่น open/create/replace file โดยเริ่มจากเลือกบล็อก open/create/replace file จาก File I/O พาเลท (ดังแสดงในรูปที่ 2.111) จากนั้นลากไปวางที่หน้าต่างบล็อกไดอะแกรม การกำหนดอินพุททำได้โดยการคลิ๊กเมาส์ขวาที่เทอร์มินอล file path (use dialog) ซึ่งเป็นเทอร์มินอลอินพุทอยู่มุมบนซ้ายของบล็อก ทำการเลือก create control

2.12.1 Writing Data to a File

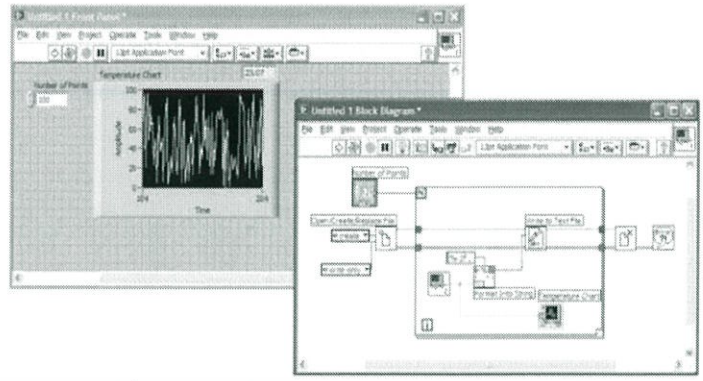
ในหัวข้อนี้จะกล่าวถึงการเขียนข้อมูลลงไฟล์ ซึ่งข้อมูลที่เขียนจะอยู่ในรูปแบบ string หรือ array string บล็อกที่จะใช้ในการสร้างโปรแกรมคือ open/create/replace file สำหรับสร้างไฟล์ใหม่ และ Close file สำหรับปิดไฟล์และข้อมูล

ในตัวอย่างนี้มีวัตถุประสงค์เพื่อที่จะสร้าง VI เพื่ออ่านค่าของอุณหภูมิซึ่งเป็นข้อมูลที่เป็นรหัส ASCII โดยใช้ฟังก์ชั่น For Loop ช่วยในการสร้างอุณหภูมิ จากนั้นให้โปรแกรมทำการแปลงข้อมูลที่เป็น Numeric ให้เป็นแบบ string

การสร้างเริ่มจากเปิดโปรแกรม LabVIEW ขึ้นมา แล้วเลือก numeric control และ wave form chart วางลงไปบนหน้าต่างฟร้อท์พาเนล สำหรับ wave form chart จะแสดงผลของอุณหภูมิให้คลิ๊กเมาส์ทางขวาบน wave form chart แล้วเลือก Visible Items» Digital Display และยกเลิก Visible Items» Plot Legend จากนั้นกำหนดย่านของอุณหภูมิในแกน Y ให้อยู่ในช่วง 0-100 องศา ที่บล็อก numeric control จะแสดงจำนวนของข้อมูลทั้งหมดที่อ่านมาว่ามีทั้งหมดกี่ชนิดข้อมูลเปลี่ยนชื่อให้เป็น Number of Points และคลิ๊กขวาที่บล็อกแล้วเลือก Representation»I32

จากนั้นเลือกไปที่หน้าต่างบล็อกไดอะแกรม เลือกฟังก์ชั่น For Loop วางลงไปบนหน้าต่างบล็อกไดอะแกรม ขยายให้มีขนาดพอที่จะวางฟังก์ชั่นอื่นลงไปได้ จากนั้นเลือกโปรแกรมย่อยของ Temperature มาวางโดยเลือกจาก Functions »Select a VI

เพิ่มฟังก์ชั่น Open/Create/Replace โดยวางไว้ด้านนอกฟังก์ชั่น For Loop คลิ๊กขวาที่บล็อกแล้วเลือก Create »Constant จากนั้นเลือก create ซึ่งหมายถึงการสร้างไฟล์ใหม่ทุกครั้งที่มีการรันโปรแกรม เพิ่มฟังก์ชั่น write to text file ไว้ภายในฟังก์ชั่น For Loop และเพิ่มฟังก์ชั่น Close a file จากนั้นต่อสายสัญญาณแต่ละบล็อกดังแสดงในรูปที่ 2.112 นำ Format into String มาช่วยในการแปลงจาก numeric ไปเป็น string



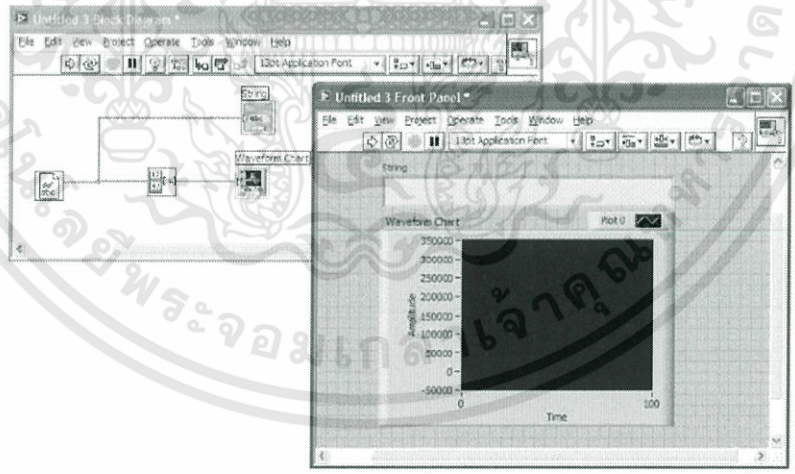
รูปที่ 2.112 ตัวอย่างโปรแกรมสำหรับเขียนไฟล์

2.12.2 Reading Data from a File

ในหัวข้อที่ผ่านมาได้กล่าวถึงวิธีการเขียนข้อมูลลงไฟล์ ไปแล้ว ในหัวข้อนี้จะกล่าวถึงวิธีการอ่านข้อมูลจากไฟล์โดยใช่ฟังก์ชั่น Read From Text File ซึ่งมีความสามารถในการอ่านไฟล์ทั้งแบบตัวเลขและแบบตัวอักษร เพื่อให้เข้าใจการทำงานมากยิ่งขึ้นให้พิจารณาตัวอย่างที่จะกล่าวต่อไปนี้ ซึ่งจะเป็นตัวอย่างวิธีการเขียนโปรแกรมเพื่ออ่านค่าของอุณหภูมิ

จุดประสงค์เพื่อให้เข้าใจโครงสร้างการเขียนโปรแกรมเพื่ออ่านค่าอุณหภูมิที่เราเขียนไว้ในตัวอย่างที่แล้วให้แสดงผลข้อมูลด้วยกราฟ ในกรณีที่ข้อมูลที่เขียนได้บันทึกเป็นรหัส ASCII เราจำเป็นที่จะต้องเขียนโปรแกรมให้อ่านข้อมูลนั้นให้อยู่ในรูปแบบของข้อมูลแบบ string

การเขียนโปรแกรมเริ่มจากเปิดหน้าต่างฟร้อนท์พาเนลขึ้นมา จากนั้นเลือก string indicator และ waveform graph วางลงไปเพื่อแสดงค่าอุณหภูมิ แสดงดัง รูปที่ 2.113



รูปที่ 2.113 ตัวอย่างโปรแกรมสำหรับการอ่านไฟล์

จากนั้นสร้างโปรแกรมในหน้าต่างบล็อกไดอะแกรม ดังรูปที่ 2.113 โดยเลือกฟังก์ชั่น Read form text file ซึ่งเป็นบล็อกสำหรับอ่านข้อมูลจากไฟล์หรือเอาท์พุทที่เป็นสตริงที่ต้องการ เมื่อทำการรันโปรแกรมบล็อกดังกล่าวจะเรียกไดอะล็อกบล็อกขึ้นมาให้เลือกไฟล์ที่ต้องการอ่าน

บล็อกต่อไปคือ Extract Numbers เป็น SubVI ซึ่งอยู่ที่ example »general »string »Extract Numbers ซึ่งจะทำหน้าที่แยกข้อมูลที่เป็นรหัส ASCII, String, Numeric ออกจากกันโดยใช้เครื่องหมายคอมม่า (,), การขึ้นบรรทัดใหม่ การเรียกใช้งานคือเลือกที่ Function »Select a VI หลังจากที่ได้บล็อกของฟังก์ชั่นต่างๆ สำหรับใช้งานแล้วให้ลากสายสัญญาณเชื่อมต่อดัง รูปที่ 2.112 จากนั้นกลับไปยังหน้าต่างฟร้อนท์พาเนลแล้วทำการทดลองรับโปรแกรม หลังจากที่กดปุ่มรันแล้วโปรแกรมจะเรียกไดอะล็อกบล็อกขึ้นมาจากนั้นให้เลือกไฟล์ที่ต้องการจะอ่านแล้วกดปุ่ม OK โปรแกรมก็จะอ่านค่าข้อมูลและแสดงผลที่กราฟ

2.12.3 Property Nodes

2.12.3.1 เกี่ยวกับ Property Node

รูปที่ 2.114 แสดงลักษณ์ของ Property Node

Property Node ใช้สำหรับปรับเปลี่ยนหรือกำหนดคุณสมบัติให้กับอุปกรณ์ต่างๆ ที่เรานำมาใช้สร้างโปรแกรม การเลือกคุณสมบัติจากโหนดทำได้โดยใช้ Operating tool โดยคลิ๊กเมาส์ที่ property terminal หรือ คลิ๊กขวาที่พื้นที่สีขาของโหนดและเลือก Property จากเมนูย่อย

Property Node ในโปรแกรม LabVIEW ถูกนำมาใช้เป็นอย่างมากซึ่งในหัวข้อนี้จะกล่าวถึงการนำ Property Node มาใช้ในการเปลี่ยนคุณสมบัติในการแสดงผลและเปลี่ยนคุณสมบัติของฟังก์ชั่นของวัตถุหรืออุปกรณ์ที่หน้าต่าง ฟร้อนท์พาเนล เราสามารถเซตคุณสมบัติได้ เช่น คุณสมบัติในการแสดงสี , ทัศนวิสัย, ตำแหน่ง, ขนาด, การกระพริบ, เมนูสตริง, สเกลกราฟและชาร์ต และอื่นๆ

2.12.3.2 การสร้าง property Node

การสร้าง Property Node ทำได้โดยคลิ๊กเมาส์ขวาที่อุปกรณ์ที่หน้าต่างฟร้อนท์พาเนลและเลือก Create » Property Node หรือคลิ๊กเมาส์ขวาที่เทอร์มินอลที่หน้าต่างบล็อกไดอะแกรมและเลือก Create » Property Node จากนั้นโปรแกรมจะสร้างโหนดใหม่ขึ้นมาที่หน้าต่างบล็อกไดอะแกรมและจะมีลาเบลแสดงชื่อเหมือนกับลาเบลของเทอร์มินอลต้นฉบับ เราสามารถที่จะเปลี่ยนชื่อของโหนดที่สร้างขึ้นมาใหม่ได้โดยที่ไม่มีผลกระทบต่อโปรแกรมและเนื่องจากว่าคุณสมบัติของ Property Node มีความแตกต่างกันการที่จะจากคุณสมบัตินั้นก็เป็นเรื่องที่ค่อนข้างยากฟังก์ชั่นที่จะช่วยอธิบายความหมายและหน้าที่การทำงานของ Property Node แต่ละแบบคือฟังก์ชั่น Context Help

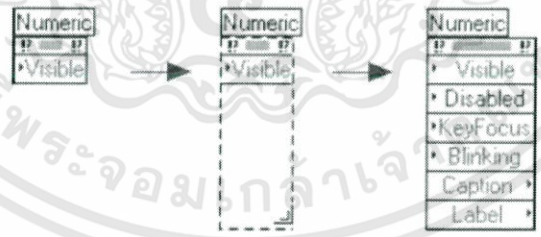
2.12.3.3 การใช้งาน property Node

ในครั้งแรกที่เราสร้าง Property Node นั้นค่าเริ่มต้นจะมีเพียงเทอร์มินอลเดียวเท่านั้นซึ่งเราสามารถที่จะทำการเพิ่มเทอร์มินอลของ Property Node ได้และสามารถเปลี่ยนสภาวะการทำงานจากอุปกรณ์อินพุทเป็นอุปกรณ์เอาท์พุทได้ตัวอย่างเช่น ถ้าเราสร้าง Property Node ของตัวควบคุมแบบดิจิตอล หลังจากที่สร้างเสร็จจะปรากฏบล็อกของ Property Node ขึ้นมาโดยมีหนึ่งเทอร์มินอลและมีรูปลูกศรเล็กๆ อยู่ทางขวาของเทอร์มินอล การกำหนดคุณสมบัติให้อุปกรณ์ดังกล่าวแสดงผลของตัวควบคุมดิจิตอลสามารถทำได้โดยเปลี่ยนคุณสมบัติของ Property Node โดยการคลิ๊กเมาส์ขวาที่ Property Node แล้วเลือก Change To Write จากเมนูย่อย แล้วต่อกับอุปกรณ์อินพุทแบบบูลีน

รูปที่ 2.115 ตัวอย่างการใช้งาน Property Node

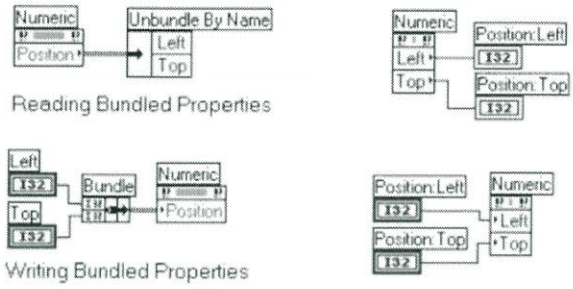
ถ้าต้องการอ่านค่าสถานะปัจจุบันของ Property node ก็สามารถทำได้โดยคลิ๊กเมาส์ขวาที่เทอร์มินอลของ Property Node และเลือก Change To Read

จากเทอร์มินอลของ Property Node เราสามารถเปลี่ยนคุณสมบัติได้โดยใช้ Operation tool คลิ๊กที่โหนดที่ต้องการแก้ไข แล้วจะปรากฏเมนูย่อยขึ้นมาซึ่งภายในประกอบไปด้วยคุณสมบัติต่างๆ ให้เลือกใช้งาน ใน Property Node นั้นสามารถที่จะกำหนดให้อ่านหรือเขียนข้อมูลได้โดยการเพิ่มจำนวน เทอร์มินอลให้กับ Property Node ซึ่งวิธีการขยายจำนวนเทอร์มินอลก็สามารถทำได้โดยใช้เมาส์คลิ๊กที่มุมล่างของ Property Node แล้วลากเมาส์เลื่อนลงเพื่อเพิ่มจำนวนเทอร์มินอลของ Property node และสามารถที่จะปรับเปลี่ยนรูปแบบของ Property ได้



รูปที่ 2.116 การเพิ่มเทอร์มินอลให้ Property Node

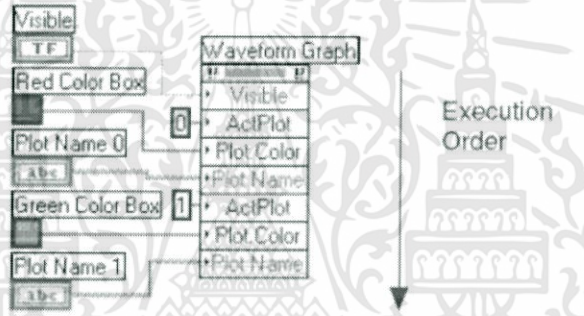
Property Node บางตัวอาจจะมีหลายคุณสมบัติอยู่ในตัวเดียว เช่น Position เป็นคุณสมบัติการบอกตำแหน่งซึ่งภายในจะมีตำแหน่งซ้ายและขวา ซึ่งเราจะใช้ฟังก์ชั่น Bundle และ Unbundle มาช่วย ในการต่อใช้งาน



รูปที่ 2.117 การใช้งาน Property Node ร่วมกับ Bundle และ Unbundle

2.12.3.4 ลำดับการทำงานของ property Node

ในกรณีที่เรากำหนดคุณสมบัติให้กับ Property Node ตั้งแต่ 2 คุณสมบัติขึ้นไปโดยที่คุณสมบัติที่เรากำหนดเพิ่มเติมจะเรียงต่อจากคุณสมบัติเดิมที่เรากำหนดลงด้านล่าง ซึ่งลำดับการทำงานโปรแกรมจะ เริ่มทำงานจากบนลงล่าง



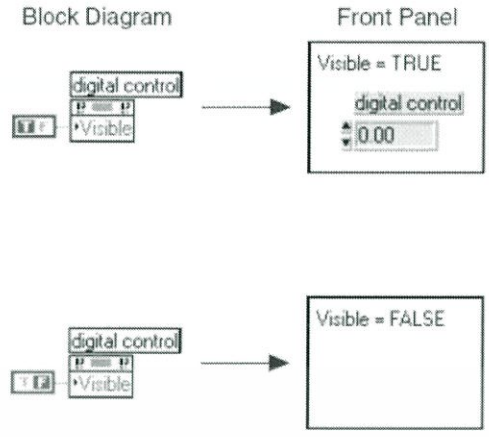
รูปที่ 2.118 แสดงลาดับการทำงานของ property Node

2.12.3.5 Common Property

คุณสมบัติของ Property Node ที่เราจะกำหนดให้กับอุปกรณ์ต่างๆ นั้นในโปรแกรม LabVIEW มีอยู่จำนวนมากพอสมควร ในหัวข้อนี้จะขอกล่าวถึงคุณสมบัติพื้นฐานและที่ถูกนำไปใช้งานเป็นประจำ ได้แก่ Visible, Disable, Key Focus, Blink, Position, Bounds และ Value

2.12.3.6 Visible Property

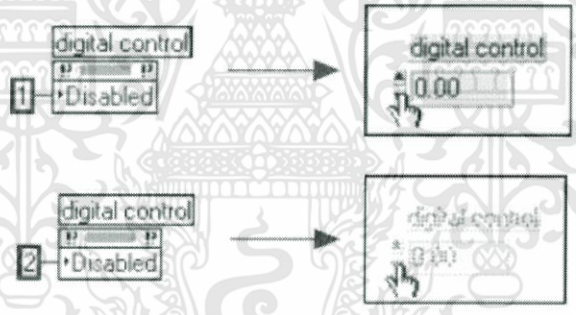
Visible Property เป็นการกำหนดทัศนวิสัยในการมองเห็นวัตถุในหน้าต่างฟร้อนท์พาเนล โดยปกติแล้วจะถูกต่อร่วมใช้งานอยู่กับอินพุทแบบบูลีน โดยการทำงานคือถ้าเงื่อนไขของอินพุทเป็นจริง (True) จะมองเห็นวัตถุนั้น แต่ถ้าเงื่อนไขของอินพุทเป็นเท็จ (False) วัตถุนั้นก็จะถูกซ่อนไว้ไม่ให้มองเห็น



รูปที่ 2.119 ตัวอย่างการใช้งาน Visible Property

2.12.3.7 Disabled Property

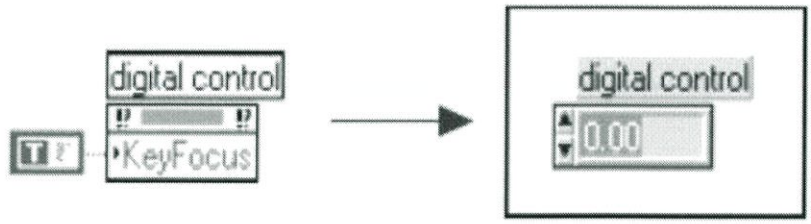
Disabled Property เป็นการกำหนดหรืออ่านค่าสถานะในการทำงานของวัตถุโดยรับอินพุทเป็นตัวเลขมี 3 ระดับคือ ถ้าค่าที่รับมาเป็น 0 วัตถุนั้นก็จะทำงานได้เป็นปกติ ถ้าค่าที่รับมาเป็น 1 วัตถุนั้นก็จะทำงานได้ถูกป้องกันการทำงานไม่ให้ปรับเปลี่ยนและถ้าค่าที่รับมาเป็น 2 วัตถุนั้นก็ถูกตัดออกจากโปรแกรม



รูปที่ 2.120 ตัวอย่างการใช้งาน Disable Property

2.12.3.8 Key Focus Property

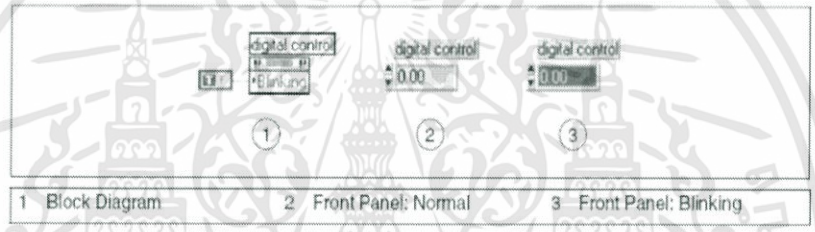
Key Focus Property การกำหนดหรืออ่านค่าของ Key Focus จากหน้าต่างฟร้อนท์ พาเนล ถ้าค่าอินพุทเป็นจริง (True) เมื่อเรากดปุ่ม Tab บนคีย์บอร์ดเคอร์เซอร์ก็จะไปโฟกัสที่วัตถุที่เรากำหนด แต่ถ้าอินพุทเป็นเท็จ (False) เมื่อเรากดปุ่ม Tab บนคีย์บอร์ดเคอร์เซอร์ก็จะเปลี่ยนไปโฟกัสวัตถุ อื่นๆ จนครบ



รูปที่ 2.121 ตัวอย่างการใช้งาน Key Focus Property

2.12.3.9 Blinking Property

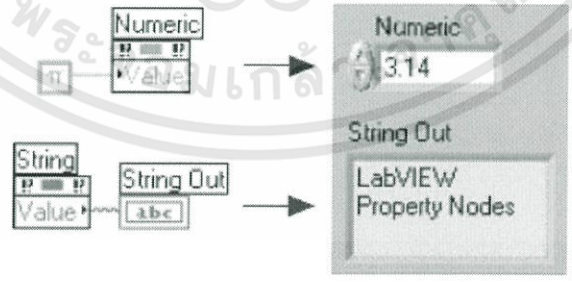
Blinking Property การกำหนดหรืออ่านค่าสถานการณ์กระพริบของวัตถุบนหน้าต่าง ฟร้อนท์พาเนล โดยวัตถุจะเริ่มกระพริบเมื่อเงื่อนไขอินพุทเป็นจริงและจะหยุดกระพริบเมื่อเงื่อนไขอินพุท เป็นเท็จ นอกจากนี้แล้วเรายังสามารถกำหนดอัตราการกระพริบและสีในการกระพริบได้โดยเลือกที่ Tools»Options ที่หน้าต่างฟร้อนท์พาเนล



รูปที่ 2.121 ตัวอย่างการใช้งาน Blink Property

2.12.3.10 Value Property

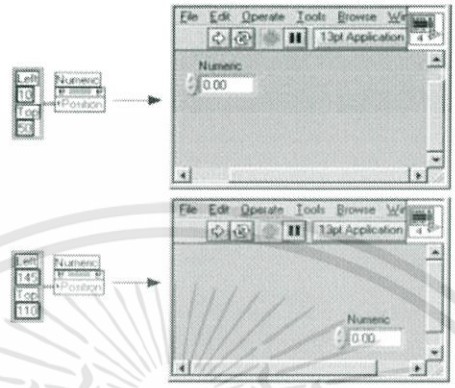
Value Property เป็นการกำหนดหรืออ่านค่าสถานะปัจจุบันของวัตถุ ซึ่งสามารถกำหนดได้ ทั้งเขียนและอ่านข้อมูล เช่นถ้าเรากำหนดให้เขียน Value Property ก็จะเขียนข้อมูลลงไปยังวัตถุนั้นและ ถ้ากำหนดให้อ่านข้อมูลมันก็จะอ่านค่าข้อมูลปัจจุบันของวัตถุนั้น



รูปที่ 2.122 ตัวอย่างการใช้งาน Value Property

2.12.3.11 Position Property

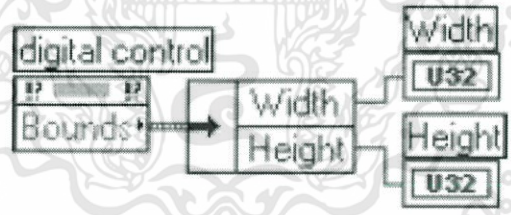
Position Property เป็นการกำหนดตำแหน่งให้กับวัตถุโดยตำแหน่งอ้างอิงคือตำแหน่งมุม บนซ้ายเพราะฉะนั้นอินพุทที่จะกำหนดตำแหน่งให้กับวัตถุจะมี 2 ค่า (เป็นคลัสเตอร์) คือค่าระยะห่างจากทางซ้ายและระยะห่างจากด้านบน มีหน่วยเป็นพิกเซล (pixel)



รูปที่ 2.123 ตัวอย่างการใช้งาน Position Property

2.12.3.12 Bound Property

Bound Property เป็นการอ่านค่าขนาดของวัตถุและแสดงค่าในหน้าต่างฟร้อนท์พาเนลซึ่ง ขนาดที่แสดงมี 2 ค่าคือความกว้างและความสูง มีหน่วยเป็นพิกเซล (pixel) Bound Property จะถูกใช้งานร่วมกับ Bundle ฟังก์ชั่นเพื่อทำการแยกข้อมูลความกว้างและความสูงออกจากกัน



รูปที่ 2.124 ตัวอย่างการใช้งาน Bound Property

2.12.3.13 Numeric Property: Format and Precision

Format and Precision Property เป็นการอ่านและกำหนดรูปแบบ (ชนิดของ เครื่องหมาย) และความแม่นยา (ตัวเลขหลังจุดทศนิยมสิบตำแหน่ง) ของตัวเลขที่แสดงบนหน้าต่าง ฟร้อนท์พาเนล โดยอินพุทจะเป็นคลัสเตอร์ที่ประกอบไปด้วยจำนวนจริงสองค่า ซึ่งค่าแรกเป็นการกำหนดรูปแบบและค่าที่สองเป็นการกำหนดความแม่นยา Format Property สามารถที่จะกำหนดให้เป็นไปตามค่าจำนวนจริงเต็มบวกดังนี้

0 Decimal Notation

1 Scientific Notation

2 Engineering Notation

3 Binary Notation

4 Octal Notation

5 Hexadecimal Notation

6 Relative Time Notation

รูปที่ 2.125 ตัวอย่างการใช้งาน Numeric Property

2.13 สรุป

องค์ประกอบต่างๆและโปรแกรมของกระบวนการควบคุมระดับน้ำที่ได้กล่าวไปนั้น ไม่ว่าจะเป็นอุปกรณ์ เครื่องมือหรือแม้แต่โปรแกรมที่นำมาใช้ในกระบวนการนี้ ล้วนแล้วแต่มีความสำคัญเป็นอย่างยิ่ง จะขาดสิ่งหนึ่งสิ่งใดไปไม่ได้เพราะอาจจะทำให้กระบวนการควบคุมระดับน้ำนี้ไม่สมบูรณ์แบบตามที่ผู้วิจัยต้องการควบคุม และผู้วิจัยเองก็ต้องมีความรู้และความเข้าใจในเครื่องมือ อุปกรณ์ โปรแกรมที่ใช้ในกระบวนการ ซึ่งได้อธิบายอย่างละเอียดไปแล้วในบทนี้ สิ่งเหล่านี้จะเป็นประโยชน์ที่ช่วยในการสร้างกระบวนการควบคุมระดับน้ำและใช้ในการหาค่าตัวควบคุมต่อไปนั่นเอง

บทที่ 3
วิธีการดำเนินงาน

3.1 กล่าวนำ

ในบทที่ 3 เป็นบทที่กล่าวถึงวิธีการดำเนินงานซึ่งจะประกอบไปด้วยองค์ประกอบของกระบวนการ การทดลอง และการจำลองการทดลอง โดยวิธีการดำเนินงานจะแบ่งออกเป็นการทดลองกับกระบวนการจริง กับการจำลองการทดลองด้วยโปรแกรม โดยจะบอกถึงรายละเอียดทางด้านฮาร์ดแวร์และซอฟต์แวร์ของทั้ง 2 ชุดการทดลอง เพื่อให้ผู้ศึกษาได้เข้าใจถึงโครงงานและนำความรู้ที่ได้ไปทำโครงงานต่อไป ในการทดลองนั้น ผู้ศึกษาได้ทำการทดลองกับชุดทดลองที่ใช้ Model ของชุดกระบวนการจริงก่อนแล้วจึงทำการทดลองกับชุดการทดลองจำลองเพื่อนำผลที่ได้มาหาประสิทธิภาพของกระบวนการจริง โดยองค์ประกอบทางด้านฮาร์ดแวร์ของชุดทดลองกระบวนการจริงได้แก่ชุดทดลอง LEGO Mindstorm NXT 2.0, คอมพิวเตอร์ ,ชุดจำลองกระบวนการระดับน้ำแบบ 2 ถัง ,อุปกรณ์วัดระดับน้ำ,อุปกรณ์แปลงสัญญาณไฟฟ้า, มอเตอร์ปั๊มน้ำและองค์ประกอบทางด้านซอฟท์แวร์ของชุดการทดลองจริงได้แก่ โปรแกรม LabVIEW 2011 หลังจากการทดลองกับชุดทดลองกระบวนการจริงแล้วผู้ศึกษาได้ทำการทดลองกับชุดการทดลองจำลอง ซึ่งมีองค์ประกอบทางด้านฮาร์ดแวร์ได้แก่ คอมพิวเตอร์, และองค์ประกอบทางด้านซอฟท์แวร์ของชุดการทดลองได้แก่ โปรแกรม LabVIEW 2011

3.2 การทำงานของกระบวนการระดับน้ำ

แบ่งออกเป็น 2 ชุดการทดลองดังนี้

3.2.1 ชุดการทดลองกระบวนการระดับน้ำจริง

กระบวนการทำงานเริ่มต้นเมื่อเราสั่งให้มอเตอร์ปั๊มน้ำเข้าไปในถังโดยสั่งผ่านโปรแกรม LabVIEW บนคอมพิวเตอร์ คอมพิวเตอร์ส่งสัญญาณผ่าน LEGO Mindstorm NXT 2.0 แล้วส่งผ่านมายัง บอร์ดขับปั๊มมอเตอร์ และส่งต่อมายังปั๊มมอเตอร์ แล้วปั๊มมอเตอร์ขับกระแสไปที่ปั๊มน้ำ เพื่อปั๊มน้ำเข้าสู่ถังน้ำเมื่อน้ำถึงระดับที่ต้องการจะมีชุดเซนเซอร์วัดความดันระดับน้ำแล้วแปลงเป็นกระแส 4-20 mA ไปที่ Current Loop โดย Current Loop จะแปลงกระแส 4-20 mA เป็นแรงดัน 1-5 V แล้วส่งสัญญาณกลับไปที่ LEGO Mindstrom NXT 2.0 เพื่อส่งไปที่คอมพิวเตอร์เพื่อทำการควบคุมกระบวนการต่อไป

3.2.2 ชุดการทดลองกระบวนการระดับน้ำจำลอง

กระบวนการนี้ทำเพื่อทดสอบหาประสิทธิภาพของกระบวนการจริง โดยทำการเขียนโปรแกรมบนคอมพิวเตอร์โดยใช้โปรแกรม LabVIEW ในการจำลองกระบวนการของระดับน้ำ

3.3 องค์ประกอบด้านฮาร์ดแวร์

3.3.1 กระบวนการของระดับน้ำแบบ 2 ถัง

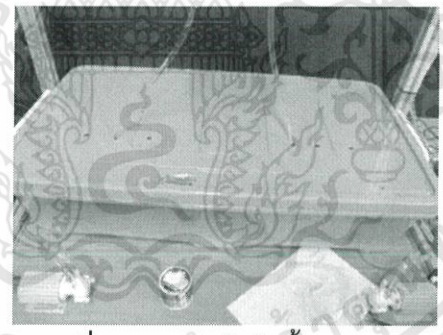
กระบวนการที่เราต้องการควบคุมระดับน้ำโดยที่เราต้องการควบคุม ระดับของน้ำในถังด้าน ล่าง ประกอบด้วย

รูปที่ 3.1 แสดงกระบวนการของระดับน้ำ

- ถังบรรจุน้ำ(Tank) เป็นถังทำมาจากอะคริลิคโปร่งใสเพื่อให้สามารถมองเห็นระดับที่เปลี่ยนแปลง โดยถังบรรจุน้ำนี้เป็นภาชนะสาหรับบรรจุน้ำที่ต้องการจะควบคุมระดับน้ำของถัง ถังบรรจุน้ำ นั้นจะมีขนาดของเส้นผ่าศูนย์กลางภายในของถัง 15 cm เส้นศูนย์กลางภายนอกของท่อน้ำทิ้ง 3 cm ความสูงของถัง 25 cm ความสูงของท่อน้ำทิ้ง 22 cm และถังบรรจุน้ำมีปริมาตรความจุที่ 20 cm เท่ากับ 3,391.2 cm^3

- สายยาง (Tube) ทำมาจากยาง ทำหน้าที่เป็นตัวส่งผ่านน้ำเข้าสู่ถังบรรจุน้ำและถังพักน้ำ มีขนาดความจุของสายยาง

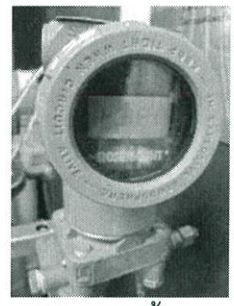
- ถังพักน้ำ (Reservoir Tank) ทำมาจากถังพลาสติกทำหน้าที่พักน้ำเพื่อจ่ายน้ำให้กับถังบรรจุน้ำทั้ง 2 ถัง ปริมาตรความจุของถังพักน้ำ 26,629 cm^3



รูปที่ 3.2 แสดงถังพักน้ำ

3.3.2 อุปกรณ์วัดระดับน้ำ

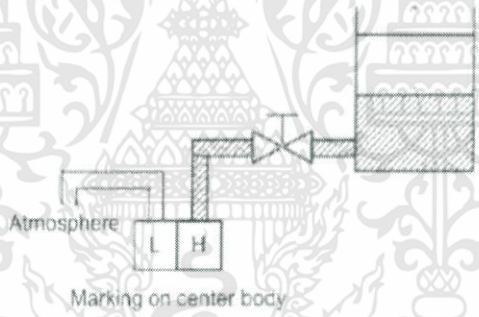
เป็นอุปกรณ์ที่ใช้ในการตรวจวัดระดับน้ำในกระบวนการระดับน้ำโดยอุปกรณ์ที่เลือกใช้คือ Differential Pressure Transmitter ของ Rosemount 3051C Differential ใช้วัดความดันด้านล่างของถังน้ำเพื่อวัดความดันซึ่งจะแปรผันตรงกับระดับความสูงของน้ำภายในถัง จากนั้นจึงส่งสัญญาณออกไป โดยสัญญาณที่ส่งออกไปจะเป็นสัญญาณมาตรฐาน 4-20 mA



รูปที่ 3.3 แสดงอุปกรณ์วัดระดับน้ำ Rosemount รุ่น 3051C

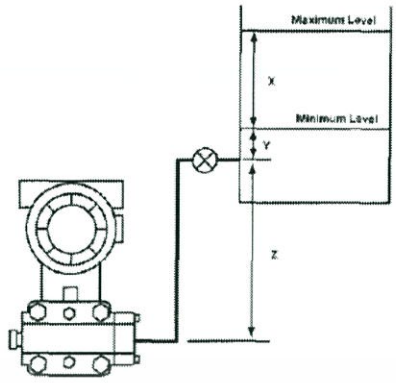
จากรูปที่ 3.3 เป็นอุปกรณ์วัดระดับน้ำของ Rosemount รุ่น 3051C โดยมีคุณสมบัติดังนี้

- Performance up to 0.075% accuracy with 100:1 turndown
- Coplanar platform enables integrated manifold, primary element and diaphragm seal solutions
- Calibrated spans/ranges from 0.1 inH2O to 4000 psi (0.25 mbar to 276 bar)
- 316L SST, Hastelloy C®, Monel®, Tantalum, Gold-plated Monel, or Gold-plated 316L SST process isolators



รูปที่ 3.4 แสดงการติดตั้ง D/P Transmitter เพื่อใช้ในการวัดระดับน้ำ

จากรูปที่ เป็นการใช้ D/P Transmitter วัดระดับของเหลวในถังโดยจะติดตั้ง D/P Transmitterไว้ในระดับของถังแล้วต่อด้าน High ของ D/P Transmitter กับก้นของถังและด้าน Low ของ D/P Transmitter จะเปิดปล่อยว่างไว้กับบรรยากาศภายนอก (Atmosphere) เป็นลักษณะการวัดแบบถังเปิด



รูปที่ 3.5 แสดงช่วงความดันแตกต่างของถังน้ำ

การคำนวณเพื่อใช้ในการติดตั้ง

Span = xG_L

Zero Suppression = $yG_L + zG_S$

G_L = ความถ่วงจำเพาะของของเหลวในถัง

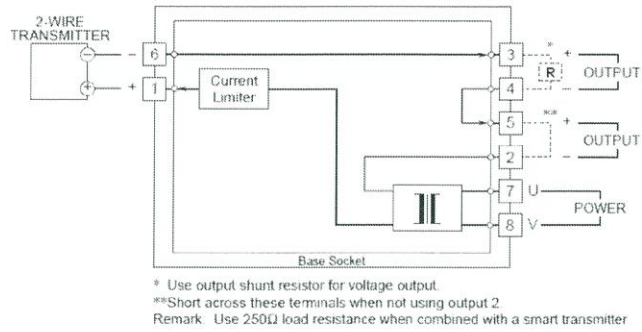
G_S = ความถ่วงจาเพาะของของเหลวในการต่อสาย

การนำ D/P Transmitter มาใช้ในการตรวจวัดระดับน้ำในกระบวนการจะต้องมีการ ปรับเทียบเพื่อความแม่นยำของการตรวจวัดระดับน้ำโดยการปรับ Zero และ Span ที่ตัว D/P Transmitter ให้ที่ระดับน้ำในถังบรรจุน้ำที่ 1 เป็น 0 cm ให้เอาท์พุตเท่ากับ 4 mA และที่ระดับน้ำ ในถังบรรจุน้ำที่ 1 เป็น 20 cm เอาท์พุตเท่ากับ 20 mA

3.3.3 อุปกรณ์แปลงสัญญาณไฟฟ้า

เป็นอุปกรณ์แปลงสัญญาณเอาท์พุตของอุปกรณ์วัดระดับน้ำซึ่งมีสัญญาณเอาท์พุตเป็นสัญญาณไฟฟ้ามาตรฐาน 4-20 mA เป็นสัญญาณไฟฟ้ามาตรฐานขนาด 0-5 Vdc เพื่อเข้าสู่กระบวนการควบคุมต่อไป

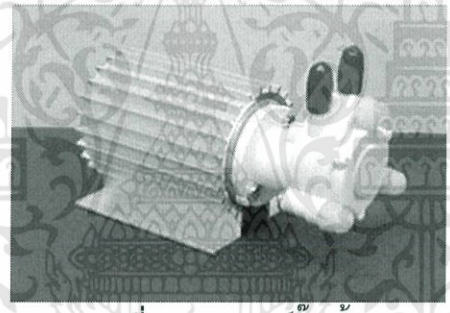
- GENERAL SPECIFICATIONS
  - Construction: Plug-in
  - Connection: M3.5 screw terminals
  - Housing material: Flame-resistant resin (black)
  - Isolation: Input or output to power
- SUPPLY OUTPUT
  - Output voltage (with no load):
  - 24 V use: 24 – 28 V DC
  - 42 V use: 39 – 43 V DC
  - Current rating: ≤ 22 mA DC
  - Short circuit Protection
  - Current limited: 40 mA max.



รูปที่ 3.6 การต่อวงจรเพื่อนำไปใช้งานของ Signal converter

3.3.4 มอเตอร์ปั๊มน้ำ

อุปกรณ์ในการปั๊มน้ำเข้าสู่กระบวนการ ประกอบด้วยสองส่วนใหญ่ๆคือ ส่วนมอเตอร์ไฟฟ้ากระแสตรง และส่วนตัวปั๊มน้ำ โดยมอเตอร์ปั๊มน้ำที่ใช้ในครั้งนี้เป็นมอเตอร์ปั๊มน้ำของบริษัท Grayler รุ่น PQ-24



รูปที่ 3.7 มอเตอร์ปั๊มน้ำ

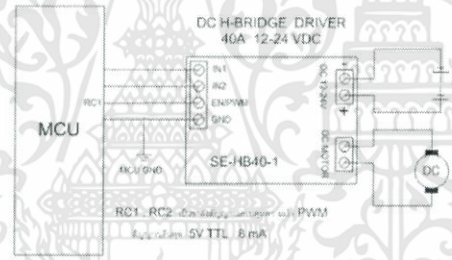
ตารางที่ 3.1 ตารางแสดงคุณสมบัติของมอเตอร์ปั๊มน้ำ

รายละเอียด	**คุณสมบัติ**
Max. Flow rate	2200 ml/min (Vdc)
Max. Suction	10 ft. wet,4” dry 2 H O
Max. Pressure	20 psig (46 ft.) 2 H O
Port Size (OD)	4.8 mm (0.18”) hose barb
Power Require (Vdc)	24 Vdc (Up to 28W)
Max. operating Temp	93 (200) c F

Ambient Fluid Temp	0 to 93 c (32 to 200 F)
Max. Viscosity	200 cps
Dimensions	88 x 81 x 92 mm
Weight	1.4 kg (3 lb)

3.3.5 บอร์ดขับปั๊มมอเตอร์ H-bridge(SE –HB40-1)

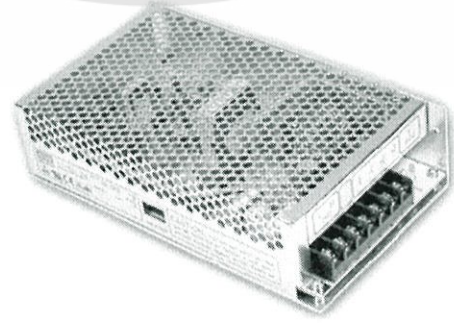
เป็นอุปกรณ์ที่ใช้ในการขับมอเตอร์ขนาด 12/24 Vdc โดยจะรับ input จาก mcu 5 vDC เราสามารถเลือกโหมดการ drive ได้ 3 mode คือ ON – OFF Control ,Direction Control ,Speed Control (PWM Drives) PWM Frequency : 400 Hz - 1000 Hz (800 Hz Recommend)



รูปที่ 3.8 การต่อวงจรเพื่อนำไปใช้งานของ **H-bridge(SE –HB40-1)**

3.3.6 Switching power supply

เป็นแหล่งจ่ายไฟฟ้ากระแสตรงที่มีกาลังส่งสูง มีน้ำหนักเบา และขนาดเล็กแหล่งจ่ายแบบเชิงเส้น ใช้ในการขับมอเตอร์ปั๊มน้ำ โดย switching power supply โดยรุ่นที่ใช้ในครั้งนี้เป็นของบริษัท Phoka รุ่น LCS 150-24



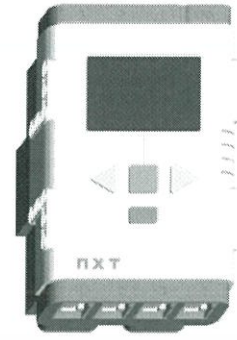
รูปที่ 3.9 switching power supply

ตารางที่ 3.2 ตารางแสดงคุณสมบัติของ Switching Power Supply

รายละเอียด	**คุณสมบัติ**
Output DC Voltage	24 V
Output Rated Current	6.5 A
Output Current Range	0–6.5 A
Output Rated Power	156 W
Output Voltage Adjust Range	21–28 V
Input Voltage Range	85-132 VAC/170-264 VAC
Input Frequency Range	50-60 Hz
Input AC Current	2.0 Amax
Input Leakage Current	3.5 mA/240V
Overload	105-150% Rate Output Power
Overvoltage	30-34.8V
Working Temp.	-10 - +60 (Refer to Output Load derating Curve)
Dimension	199 110 x 50 mm.

3.3.7 LEGO Mindstrom NXT 2.0

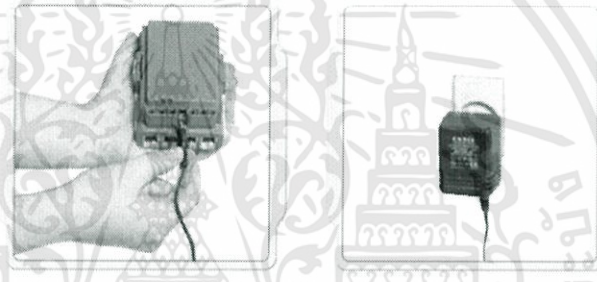
เป็นอุปกรณ์หุ่นยนต์ประเภทหนึ่งซึ่งผลิตโดยบริษัท LEGO สถาปัตยกรรมภายในประกอบด้วยไมโครคอลโทรลเลอร์ตระกูล ARM 7 32 bit สามารถสื่อสารไร้สายผ่านระบบ bluetooth 2.0 กับคอมพิวเตอร์ผ่านโปรแกรม LabVIEW มี 3 พอร์ตเอาต์พุต และ 4 พอร์ตอินพุต แบบ 6 สายในระบบดิจิทัล



รูปที่ 3.10 LEGO Mindstrom NXT 2.0

- ความรู้เบื้องต้นเกี่ยวกับ LEGO Mindstrom NXT 2.0

3.3.7.1 การชาร์จแบตเตอรี่

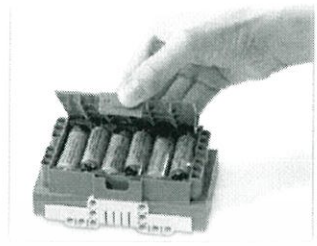


รูปที่ 3.11 การชาร์จแบตเตอรี่

การชาร์จแบตเตอรี่ ให้เสียบสายชาร์จด้านหนึ่งในที่เสียบสายชาร์จที่อยู่ใต้พอร์ต 1-4 ส่วนอีกด้านให้เสียบกับปลั๊กไฟ

- ไฟสีเขียวจะสว่างขึ้นเมื่อต่อแบตเตอรี่กับที่ชาร์จ
- ไฟสีแดงจะสว่างระหว่างชาร์จ และจะดับลงเมื่อชาร์จเสร็จแล้ว
- การชาร์จแบตเตอรี่ใช้เวลาประมาณ 4 ชั่วโมง
- สามารถใช้งาน NXT ระหว่างชาร์จได้ แต่จะทำให้ใช้เวลาในการชาร์จนานขึ้น
- แบตเตอรี่นี้สามารถชาร์จได้ 500 ครั้ง

- การใช้ถ่านไฟฉายขนาด AA



รูปที่ 3.12 การใช้ถ่ายไฟฉายขาด AA

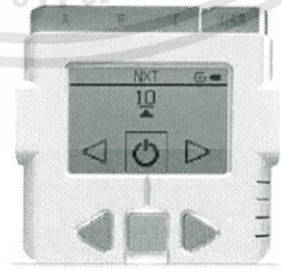
-ให้ใส่ถ่านไฟฉายขนาด AA จำนวน 6 ก้อน ลงไปในช่องใส่ถ่านแล้วปิดด้วยฝาปิด

- กรณีแบตเตอรี่อ่อน

เมื่อแบตเตอรี่อ่อน จะมีข้อความแสดงบนจอ NXT ว่า **Low Battery**

รูปที่ 3.13 แบตเตอรี่อ่อน

เราสามารถประหยัดพลังงาน NXT ได้โดยการตั้งเวลาปิดเครื่อง โดยไปที่เมนู Settings / Sleep แล้วเลือกเวลาที่ต้องการให้ NXT ปิดเครื่อง



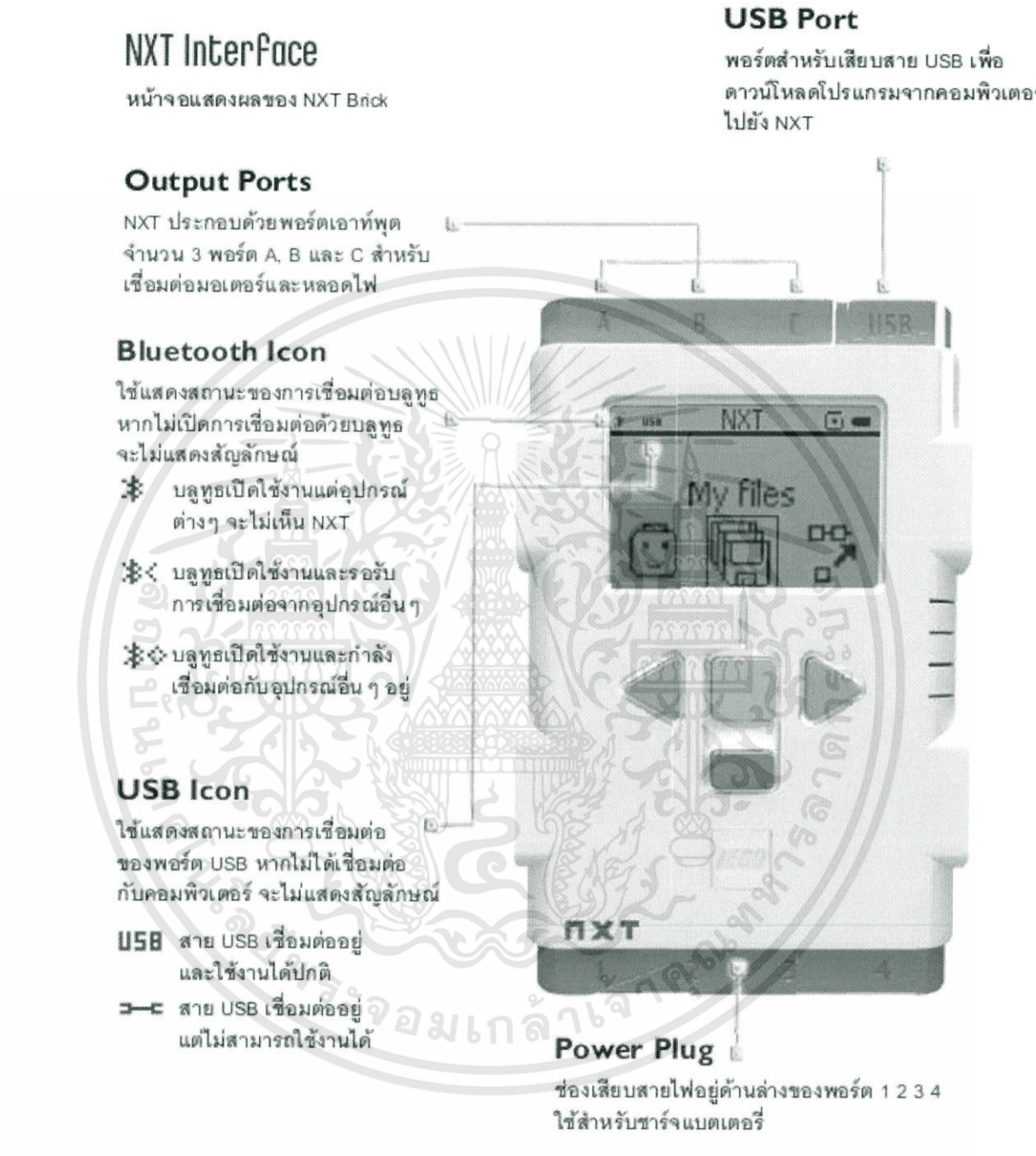
รูปที่ 3.14 โหมดประหยัดพลังงาน

ข้อควรระวังเกี่ยวกับแบตเตอรี่

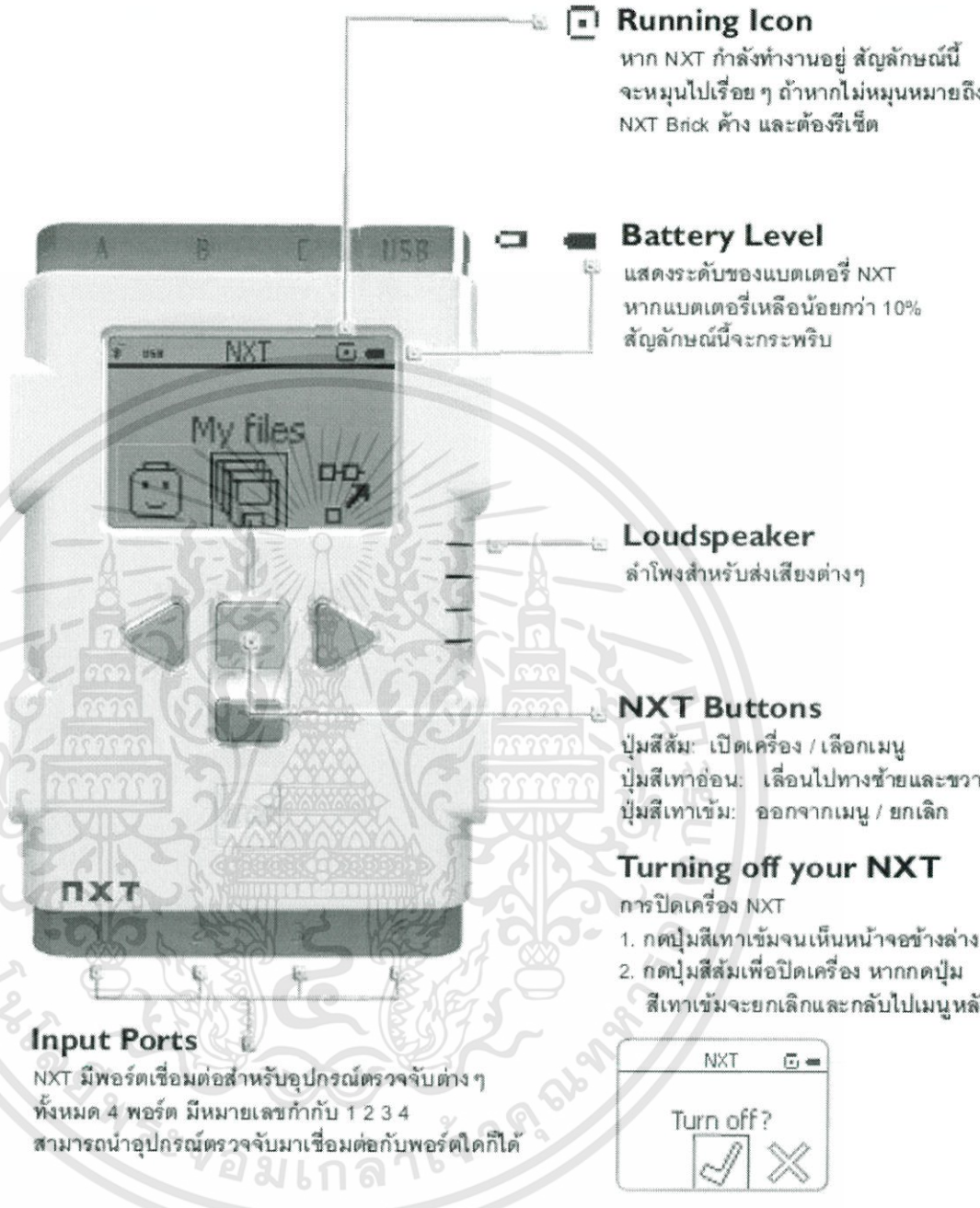
-อย่าใช้แบตเตอรี่คนละชนิดบน NXT

-หากจะไม่ได้ใช้ NXT เป็นเวลานาน ให้นำแบตเตอรี่ออกจาก NXT ก่อน
-หากแบตเตอรี่หมด ให้เอาออกจาก NXT โดยทันที
-การชาร์จแบตเตอรี่ควรทำตามคำแนะนำเกี่ยวกับการชาร์จแบตเตอรี่
-อย่าชาร์จแบตเตอรี่ที่ไม่สามารถชาร์จได้

3.3.7.2 หน้าจอแสดงผล NXT



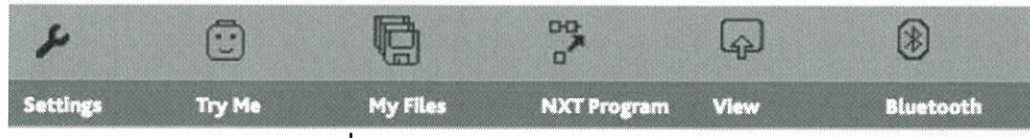
รูปที่ 3.14 หน้าจอแสดงผล NXT



รูปที่ 3.15 หน้าจอแสดงผล NXT

3.3.7.3 เมนูต่างๆ ในเมนูหลักของ NXT

เมื่อเราเปิด NXT เราจะพบกับเมนูหลัก ประกอบไปด้วยเมนูต่างๆ



รูปที่ 3.15 เมนูหลักของ NXT

-Settings: เมนูนี้ สามารถตั้งค่าต่างๆ เกี่ยวกับหุ่นยนต์ เช่น การปรับความดังของลำโพง Loudspeaker, หรือตั้งเวลาปิด NXT เพื่อประหยัดแบตเตอรี่ และมีเมนูให้ลบโปรแกรมทั้งหมดที่เราเคยดาวน์โหลดใส่หุ่นยนต์ NXT

-Try Me: เมนูนี้ จะมีโปรแกรมพื้นฐานต่างๆ ที่มีอยู่ในหุ่น NXT ไว้สำหรับให้เราทดลองเบื้องต้น

-My Files: เมนูนี้ จะเป็นที่เก็บโปรแกรมทั้งหมดที่เขียนด้วยโปรแกรมควบคุม เช่น NXT-G

-NXT Program: เมนูนี้ สามารถเขียนโปรแกรมง่ายๆ บน NXT โดยไม่จำเป็นต้องใช้คอมพิวเตอร์

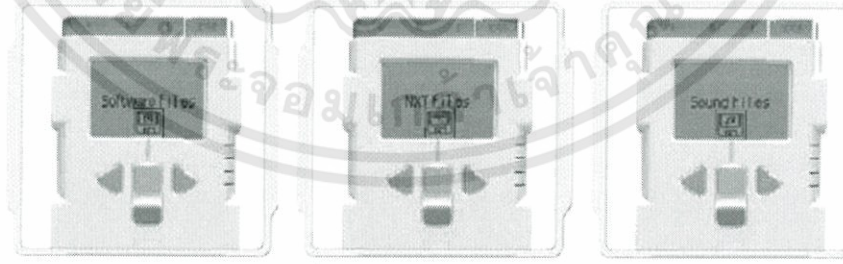
-View: เมนูนี้ สามารถตรวจสอบค่าและทดสอบเซ็นเซอร์ต่างๆ บน NXT

-Bluetooth: เมนูนี้ สามารถตั้งค่าของ Bluetooth และเชื่อมต่อกับอุปกรณ์อื่นๆ เช่นโทรศัพท์มือถือ คอมพิวเตอร์ หรือหุ่นยนต์ NXT ตัวอื่นๆ

- My Files

รูปที่ 3.16 เมนู My File

ในเมนู My Files จะเก็บโปรแกรมทั้งหมดที่เราเคยสร้างบน NXT หรือดาวน์โหลดมาจากคอมพิวเตอร์ ประกอบด้วยเมนูย่อย 3 เมนู



รูปที่ 3.17 เมนูย่อย My File

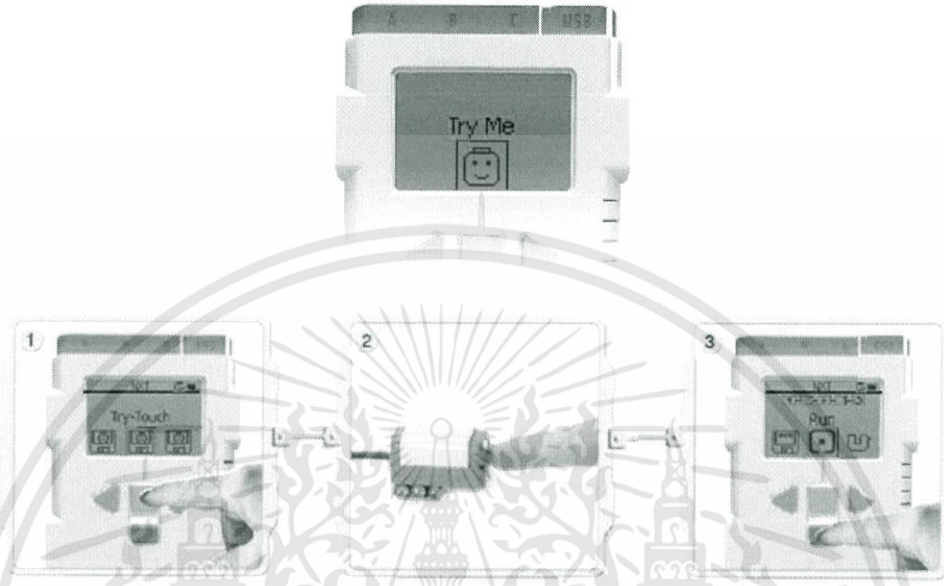
1. Software Files เก็บโปรแกรมที่เขียนจากโปรแกรมคอมพิวเตอร์แล้วดาวน์โหลดมายัง NXT

2. NXT Filesโปรแกรมที่สร้างบน NXT โดยไม่ใช้คอมพิวเตอร์

3. Sound Files ไฟล์เสียงที่เคยใช้เป็นส่วนประกอบของโปรแกรม

- Try Me

ในเมนู Try Me จะมีโปรแกรมต่างๆ ที่มากับ NXT ใช้เพื่อทดลองต่างๆ เกี่ยวกับมอเตอร์และเซ็นเซอร์

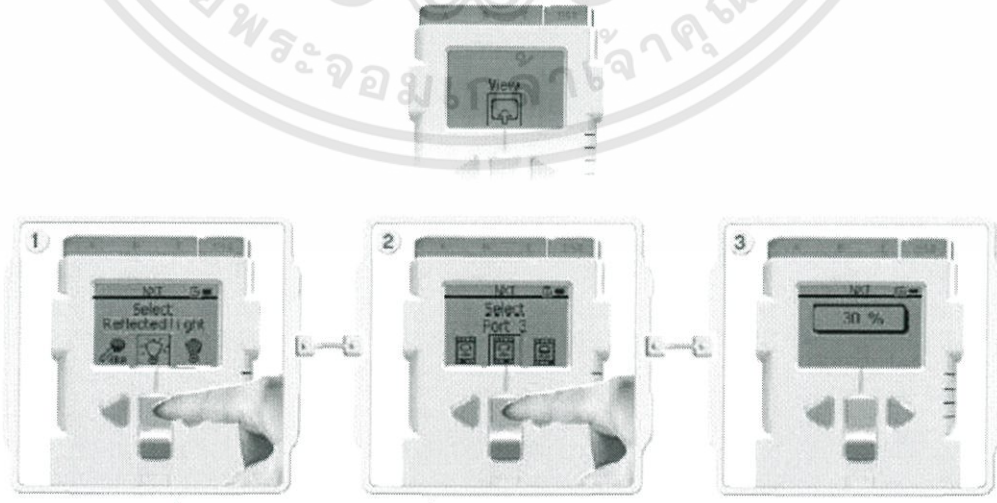


รูปที่ 3.18 เมนู Try Me

1.เข้าโปรแกรม เลือกโปรแกรม Try-Touch
2.ลองกดปุ่มที่ Touch Sensor
3.โปรแกรมจะทำงานไปเรื่อยๆ ให้กดปุ่มสีเทาเข้มเพื่อจบการทำงานของโปรแกรม

- View

เมนู **View** ใช้เพื่อดูค่าของเซ็นเซอร์และมอเตอร์บน NXT

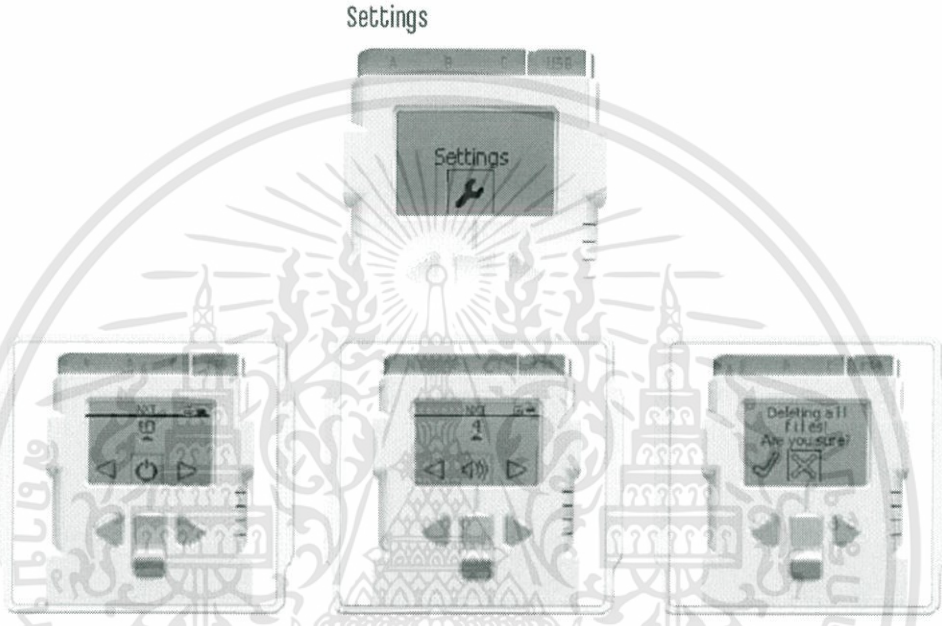


รูปที่ 3.19 เมนู **View** ใช้เพื่อดูค่าของเซ็นเซอร์และมอเตอร์บน NXT

1. เลือกชนิดของเซ็นเซอร์
2. เลือกพอร์ต
3. ดูค่าของเซ็นเซอร์

- **Settings**

ในเมนู Settings สามารถตั้งค่าต่างๆ ของ NXT เช่นความดังของลำโพง ตั้งเวลาการปิดเครื่อง หรือจะลบโปรแกรมทั้งหมดออกจาก NXT ก็ได้



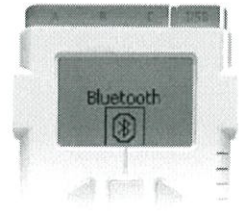
รูปที่ 3.20 เมนู **Setting**

Sleep Mode : สามารถตั้งให้ NXT ปิดเองเมื่อไม่ใช้ เพื่อประหยัดพลังงาน

Change Volume : เลือกความดังของลำโพง Loudspeaker ตั้งแต่ 0 (ปิดเสียง) จนถึง 4 (ดัง)

Delete All Program : สามารถลบโปรแกรมทั้งหมดบน NXT ได้แก่ โปรแกรมใน Software Files, NXT Files และ Try Me

- **Bluetooth**

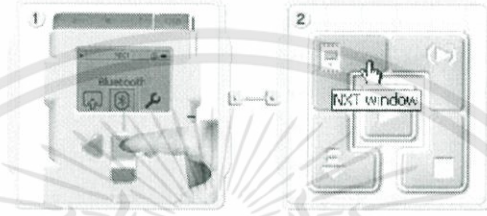


รูปที่ 3.21 เมนู **Bluetooth**

ในเมนู Bluetooth สามารถตั้งค่าการเชื่อมต่อกับอุปกรณ์ Bluetooth อื่นๆ เช่น NXT ตัวอื่นๆ โทรศัพท์มือถือ หรือคอมพิวเตอร์ นอกจากนี้ยังสามารถใช้ดาวน์โหลดโปรแกรมโดยไม่ต้องใช้สาย USB ได้อีกด้วย

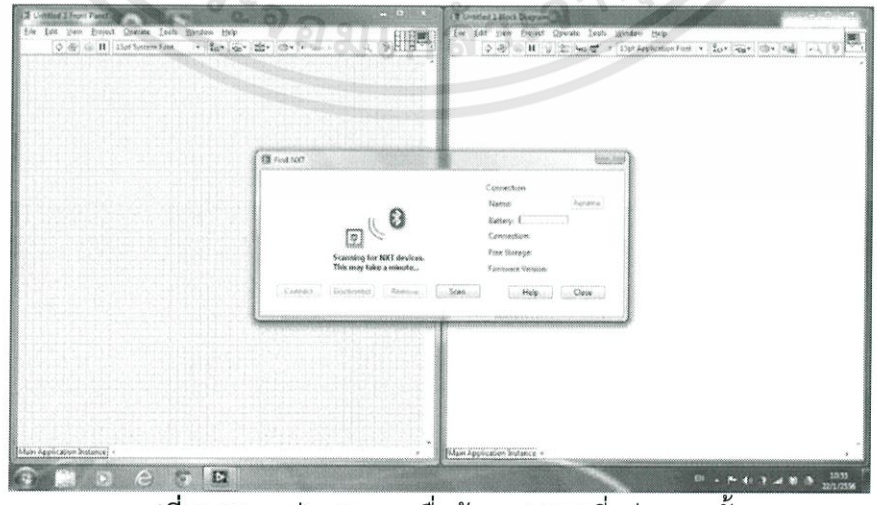
- **วิธีการตั้งค่า Bluetooth**

การเชื่อมต่อ NXT ผ่าน Bluetooth
ก่อนดำเนินการ ขอให้แน่ใจก่อนว่าเราได้ติดตั้งโปรแกรม NXT ไว้เรียบร้อยแล้ว

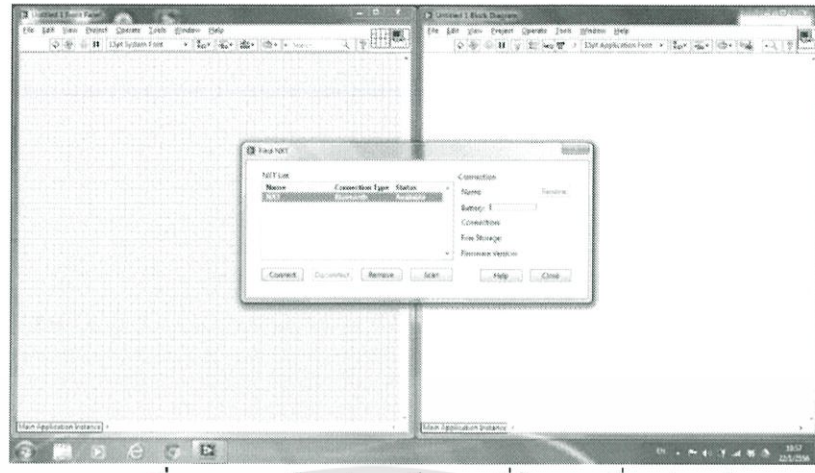


รูปที่ 3.22 เปิดระบบ Bluetooth ให้เรียบร้อย แล้วตั้ง Visibility = **On**

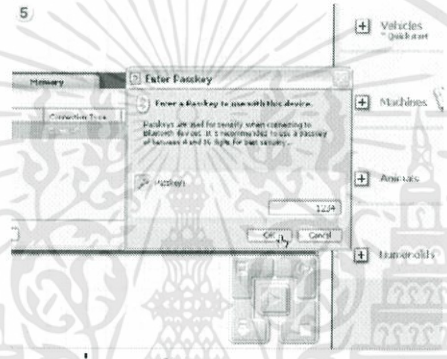
รูปที่ 3.23 คลิก Find NXT



รูปที่ 3.24 กดปุ่ม **Scan** เพื่อค้นหา NXT ที่อยู่แถวๆ นั้น

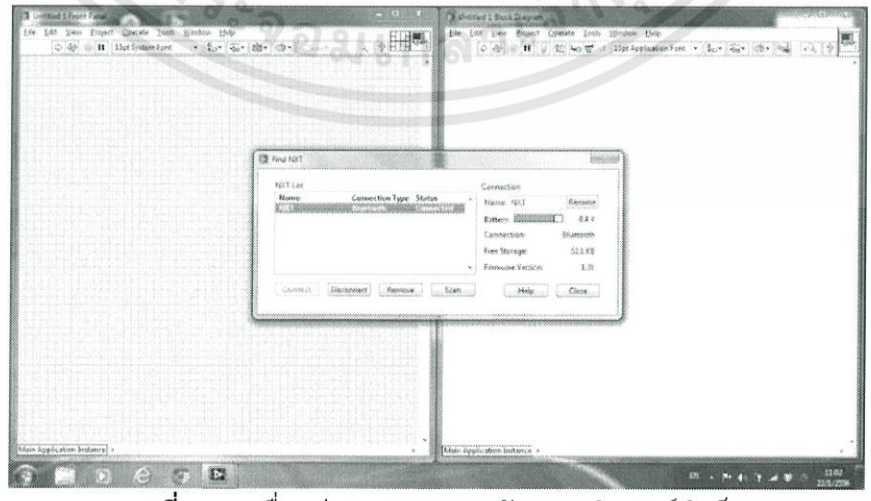


รูปที่ 3.25 กดเลือกหุ่นยนต์ NXT ที่ต้องการเชื่อมต่อ แล้วกด Connect



รูปที่ 3.26 ใส่ Passkey สำหรับหุ่นยนต์ แล้วกด OK

รูปที่ 3.27 ใส่ Passkey ในหุ่นยนต์ ให้ตรงกับที่กำหนดในขั้นที่แล้ว แล้วกด OK



รูปที่ 3.28 เชื่อมต่อ LEGO NXT กับ คอมพิวเตอร์สำเร็จ

3.4 องค์ประกอบด้านซอฟแวร์

3.4.1 โปรแกรม LabVIEW version 2011

3.5 การออกแบบตัวควบคุม

การออกแบบตัวควบคุมคุม หมายถึง การคำนวณหาค่าพารามิเตอร์ของตัวควบคุม เพื่อให้เป็นไปตามเงื่อนไข หรือข้อกำหนดที่ต้องการ ซึ่งจำเป็นจะต้องออกแบบตัวควบคุมให้เรียบร้อยก่อนที่จะทำการทดลองจริง สำหรับเนื้อหาในบทนี้จะประกอบไปด้วย วิธีการหาสมการคุณลักษณะของกระบวนการ การออกแบบตัวควบคุมพีไอ การจำลองและการทดลองกระบวนการซึ่งจะนำเอาทฤษฎีทั้งหมดที่ได้กล่าวมาแล้วรวมทั้งเรื่องฮาร์ดแวร์และซอฟท์แวร์มาช่วยในการหาจุดทำงานและหาสมการคุณลักษณะหรือโมเดลของกระบวนการโดยวิธีการออกแบบตัวควบคุมดังที่กล่าวมานั้นจะอ้างอิงทฤษฎีในบทที่ 2 เป็นหลัก ซึ่งขั้นตอนการออกแบบมีขั้นตอนดังนี้

3.5.1 การหาสมการคุณลักษณะของกระบวนการ

การกำหนดจุดทำงานเพื่อหาสมการคุณลักษณะหรือแบบจำลองของกระบวนการนั้นสำคัญมาก เนื่องจากสมการที่ได้จะถูกนำไปออกแบบหาค่าพารามิเตอร์ของตัวควบคุมแล้วจะนำค่าพารามิเตอร์ของตัวควบคุมแล้วจะนำค่าพารามิเตอร์ของตัวควบคุมที่ออกแบบได้ไปควบคุมกระบวนการจริง ถ้าค่าสมการคุณลักษณะที่หามาได้ใกล้เคียงกับกระบวนการจริงโอกาสที่พารามิเตอร์ของตัวควบคุมจะสามารถควบคุมกระบวนการได้อย่างมีประสิทธิภาพ ก็เป็นไปได้สูง แต่ในทางกลับกัน ถ้าสมการคุณลักษณะที่หามาได้ไม่ใกล้เคียงกับกระบวนการหรือไม่ถูกต้อง โอกาสที่ค่าพารามิเตอร์ของตัวควบคุมจะสามารถควบคุมกระบวนการได้อย่างมีประสิทธิภาพก็เป็นไปได้น้อยหรืออาจจะไม่สามารถควบคุมได้เลย เพราะฉะนั้นการหาสมการคุณลักษณะของกระบวนการจึงถือว่าสำคัญ ซึ่งหลักการหาจุดทำงานของกระบวนการคือ ป้อนแรงดันที่เหมาะสมค่าหนึ่งให้กับปั๊ม ปรับวาล์วให้เหมาะสม รอจนกว่าน้ำอยู่ในสภาวะคงที่ที่ค่าหนึ่งๆ จากนั้นเพิ่มแรงดันให้กับปั๊มอีกค่าหนึ่งโดยค่าแรงดันนี้จะต้องไม่ทำให้น้ำล้น และปั๊มน้ำจนกระทั่งน้ำอยู่ในสภาวะคงที่ ณ จุดนั้นจะเป็นจุดทำงานของกระบวนการ จากนั้นนำกราฟที่ได้จากการทดลองมาหาสมการคุณลักษณะ โดยใช้วิธีการหาแบบจำลองทางคณิตศาสตร์ ของกระบวนการจากผลตอบสนองแบบขั้น ดังนี้

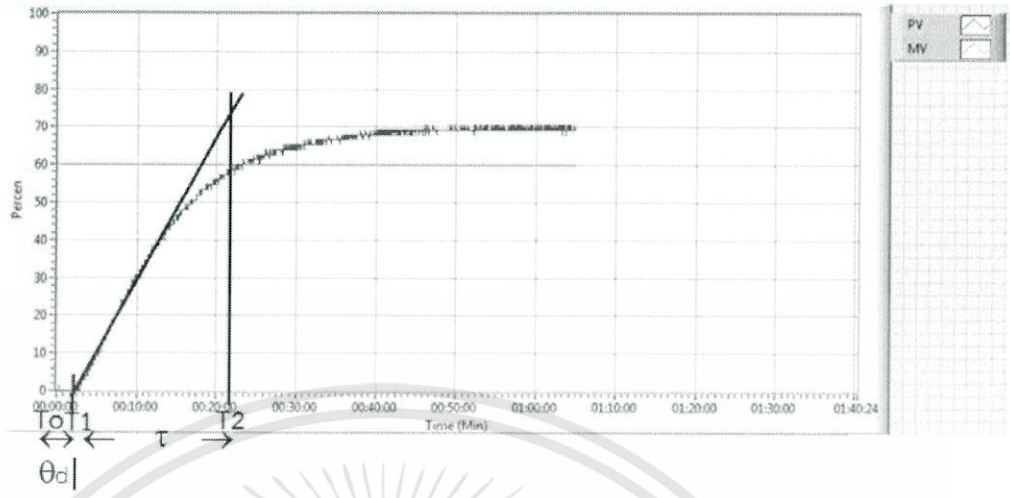
ในกระบวนการทางอุตสาหกรรมส่วนใหญ่นั้น พบว่าเป็นแบบอันดับหนึ่งที่มีเวลาหน่วงหรือที่เรียกว่า กระบวนการอันดับหนึ่งแบบมีเวลาหน่วง (First Order plus Dead Time-Process) โดยฟังก์ชันถ่ายโอนแสดงได้ดังนี้

$$G_p = \frac{K_{ss}}{\tau s+1} e^{-T_0 s} \qquad (3.1)$$

เมื่อ K_{ss} = อัตราขยายของกระบวนการในสภาวะหยุดนิ่ง

τ = ค่าเวลาคงที่ของกระบวนการ

T_0 = ค่าเวลาหน่วงของกระบวนการ หน่วยเป็น วินาที



รูปที่ 3.29 Step Response

จากผลตอบสนองที่ได้จากกระบวนการ นำมาคำนวณหาโมเดลทางคณิตศาสตร์ได้โดยหา

K_{ss}, τ , T_0 โดยคำนวณจากการประมาณค่า ดังนี้

เวลา T_1 คือ เวลาที่ MV เริ่มเปลี่ยนแปลง

เวลา T_2 คือ เวลาที่เส้นสัมผัสตัดแกนเวลา

เวลา T_3 คือ เวลาที่ระบบตอบสนองที่ 63%

ดังนั้นสามารถหาค่าอัตราขยายสภาวะนิ่ง ค่าเวลาคงที่ และเวลาหน่วงได้ดังนี้

$$K_{ss} = \frac{\Delta PV\%}{\Delta MV\%} \tag{3.2}$$

$$\tau = T_2 - T_1 \min \tag{3.3}$$

$$\theta d = T_1 - T_0 \min \tag{3.4}$$

จากรูปที่ 3.29

จะได้ $K_{ss} = \frac{70}{60} = 1.16, \tau = 25.81, \theta d = 1.66$

ดังนั้น

$$G(s) = \frac{1.16}{25.81s+1} e^{-1.66s} \tag{3.5}$$

ทำการหาค่า $e^{-1.66s}$ โดยใช้วิธี Pade จะได้ค่าของ $e^{-1.66s} = \frac{-s+1.205}{s+1.205}$

จากกสมการที่ จะได้ทรานส์เฟอร์ฟังก์ชันของโมเดลอ้างอิงดังนี้

$$\frac{H_2}{Q_i} = \frac{-1.16s + 1.3978}{25.81s^2 + 32.10s + 1.205} \tag{3.6}$$

3.5.2 การออกแบบตัวควบคุมพีไอ (PI)

จากที่ได้คำนวณหาสมการคุณลักษณะของกระบวนการของตัวควบคุมแล้ว ขั้นตอนต่อไปก็จะนำค่าพารามิเตอร์ต่างๆไปใช้ในการคำนวณหาค่าพีไอ (PI) โดยใช้วิธีของ Ziegler-Nichols แสดงได้ดังนี้

$$\alpha = \frac{\theta_d}{\tau} = \frac{1.66}{25.81} = 0.06 \qquad (3.7)$$

จากนั้นนำค่าจาก สมการ (3.2), (3.3), (3.4) และ (3.7) ไปคำนวณหาค่าพารามิเตอร์ตามรูปที่ 3.30 ได้ดังนี้

Ziegler-Nichols			
Controller Type	K_c	τ_I	τ_D
P	$\frac{1}{K\alpha}$	-	-
PI	$\frac{0.9}{K\alpha}$	$\frac{1}{3.33\theta_d}$	-
PD	$\frac{1.2}{K\alpha}$	$\frac{1}{2\theta_d}$	$0.5\theta_d$
1Recommended Range of Applicability 1.0<(a/t)<0.1			

รูปที่ 3.30 แสดงการคำนวณค่าพารามิเตอร์ในประเภทต่างๆ

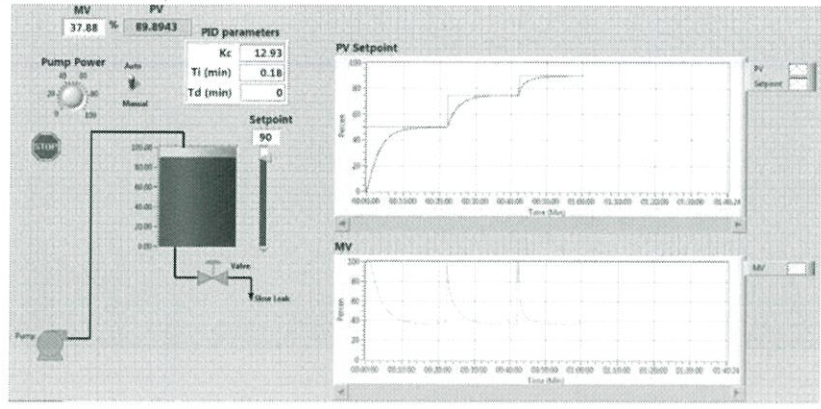
จากการคำนวณจะได้ค่าพารามิเตอร์ K_p = 12.93 และ K_i = 0.18 และนำค่าพารามิเตอร์ที่คำนวณได้ไปทดลองในกระบวนการจำลองและกระบวนการจริง

3.5.3 การจำลองและการทดลอง

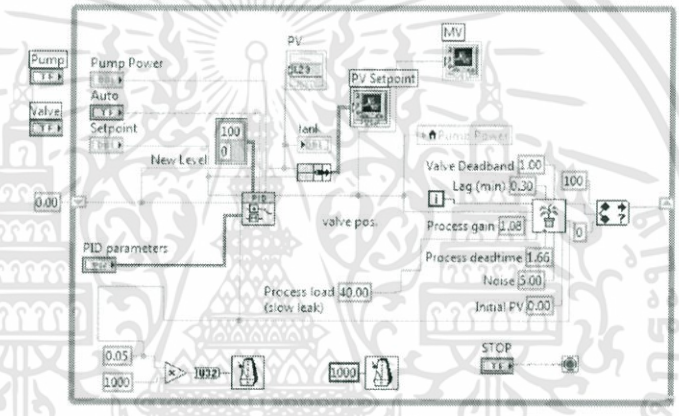
3.5.3.1 การจำลองกระบวนการ

การจำลองกระบวนการคือการนำเอาค่าพารามิเตอร์ของกระบวนการที่จากการคำนวณไปจำลองการทำงานในเครื่องคอมพิวเตอร์โดยในการทดลองนี้ใช้โปรแกรม LabVIEW เป็นโปรแกรมสำหรับจำลองการทำงานของกระบวนการเพื่อทดสอบความเป็นไปได้ก่อนนำไปทดลองจริง โดยการจำลองกระบวนการทำได้ดังนี้

การจำลองกระบวนการด้วยตัวควบคุมแบบ PI จากสมการคุณลักษณะที่คำนวณได้นำมาจำลองควบคุมด้วยตัวควบคุมพีไอได้ดังนี้



รูปที่ 3.31 แสดงการจำลองของกระบวนการ (Panel)



รูปที่ 3.32 แสดงการจำลองของกระบวนการ (บล็อกไดอะแกรม)

3.5.4 การจำลองกระบวนการผ่านชุดจำลองของถังน้ำสองถัง

3.5.4.1 การทดลองใช้โปรแกรม LabVIEW ส่งข้อมูลไปยัง Lego NXT Mindstorm ผ่าน Bluetooth เพื่อให้ปั๊มน้ำทำงาน

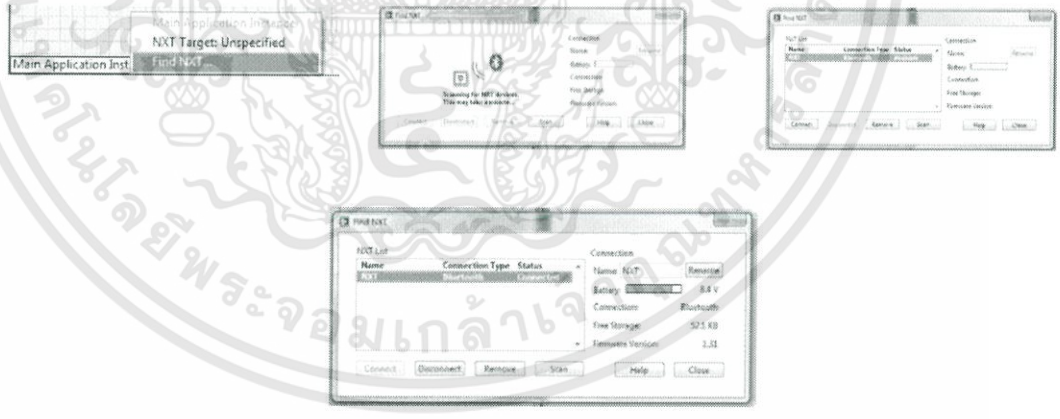
3.5.4.1.1 อุปกรณ์ที่ใช้ในการททดลอง

1. เครื่องคอมพิวเตอร์พร้อมซอฟแวร์ LabVIEW 1 ชุด
2. Bluetooth
3. Lego NXT Mindstorm
4. ชุดจำลองถังน้ำสองถัง

รูปที่ 3.33 แสดงอุปกรณ์ทั้งหมดของการทดลอง

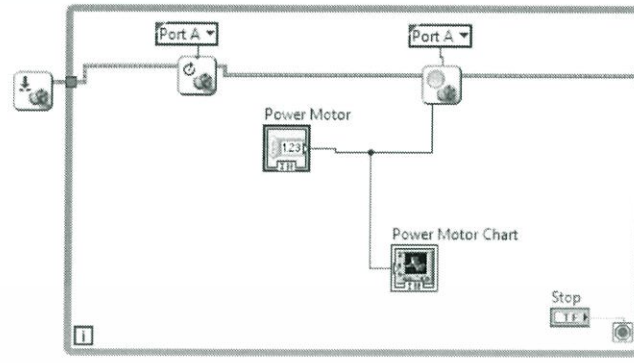
3.5.4.1.2 ลำดับขั้นการทดลอง

นำ Bluetooth เชื่อมต่อกับคอมพิวเตอร์ จากนั้นเรียกใช้โปรแกรม LabVIEW เพื่อทำการเชื่อมต่อกับ Lego NXT Mindstorm



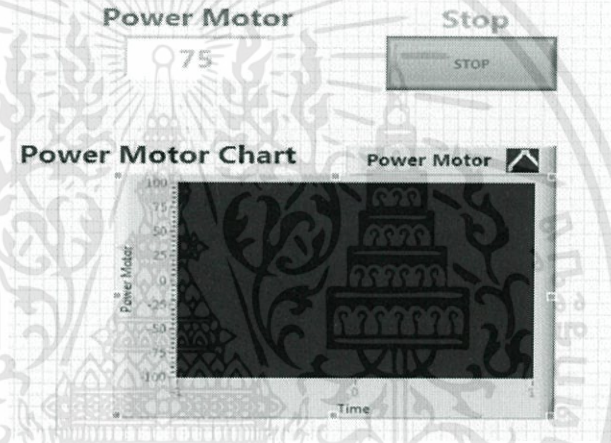
รูปที่ 3.34 แสดงการใช้ Bluetooth ค้นหา Lego NXT Mindstorm

เขียนโปรแกรมเพื่อปั๊มน้ำเข้าถังโดยใช้โปรแกรม LabVIEW



รูปที่ 3.35 แสดงโปรแกรมปั๊มน้ำเข้าถังโดยใช้โปรแกรม LabVIEW

ทำการเขียนโปรแกรมฝังใน Lego NXT Mindstorm



รูปที่ 3.36 แสดงการทำงานของโปรแกรมปั๊มน้ำ

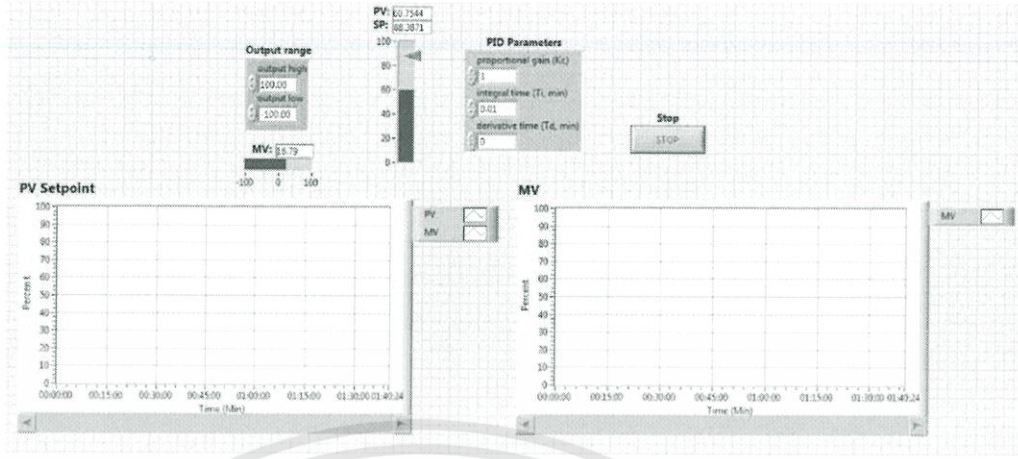
3.5.4.1.3 ผลการทดลอง

สามารถปั๊มน้ำเข้าถังได้โดยสามารถปรับแรงดันของมอเตอร์ปั๊มน้ำได้ตามต้องการ

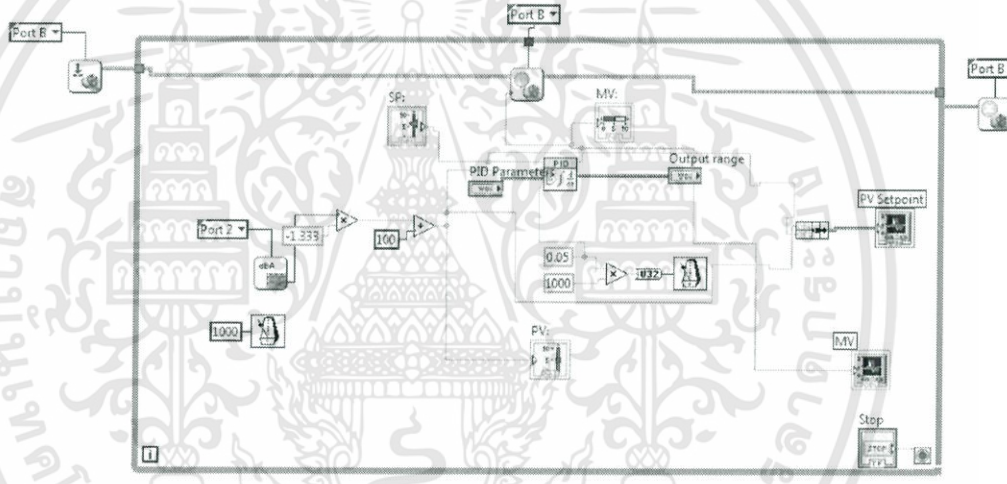
3.5.5 การควบคุมระดับน้ำของถังสองถังโดยใช้โปรแกรม LabVIEW

3.5.5.1 ลำดับขั้นการทดลอง

เขียนโปรแกรมเพื่อควบคุมระดับน้ำ โดยใช้โปรแกรม LabVIEW



รูปที่ 3.37 แสดงโปรแกรมควบคุมระดับน้ำ (Panel)



รูปที่ 3.38 แสดงโปรแกรมควบคุมระดับน้ำ (บล็อกไดอะแกรม)

3.5.5.2 ผลการทดลอง

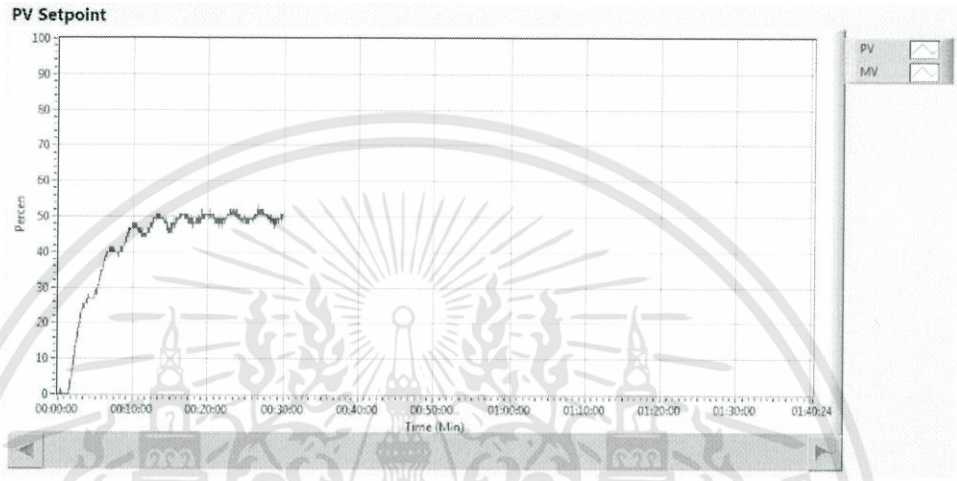
สามารถควบคุมระดับน้ำได้ตามต้องการ

บทที่ 4
ผลการทดลอง

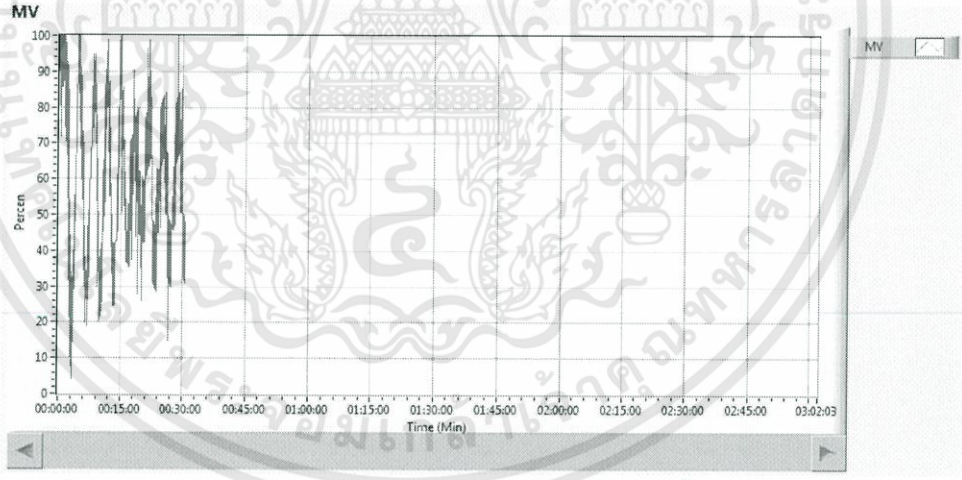
4.1 กล่าวนำ

สำหรับเนื้อหาในบทนี้จะว่าด้วยเรื่องเกี่ยวกับผลการทดลองจากกระบวนการจำลองและกระบวนการจริง ซึ่งรายละเอียดจะได้กล่าวดังนี้

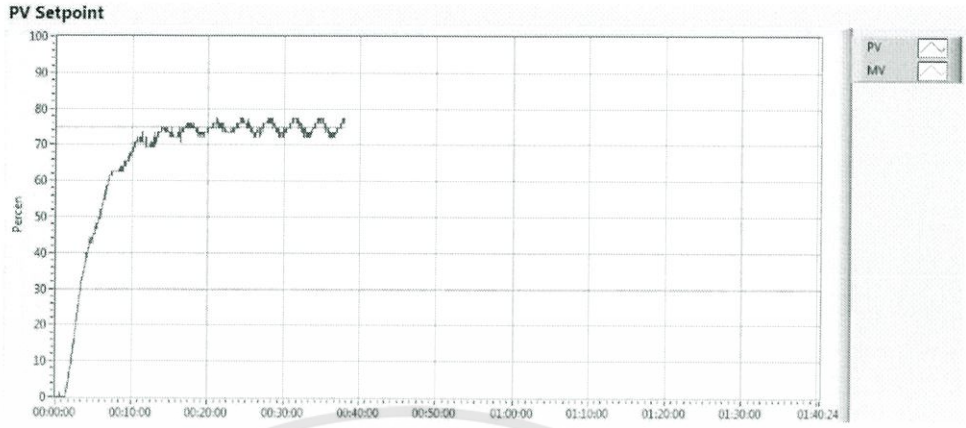
4.2 ผลการทดลองจากกระบวนการจริง



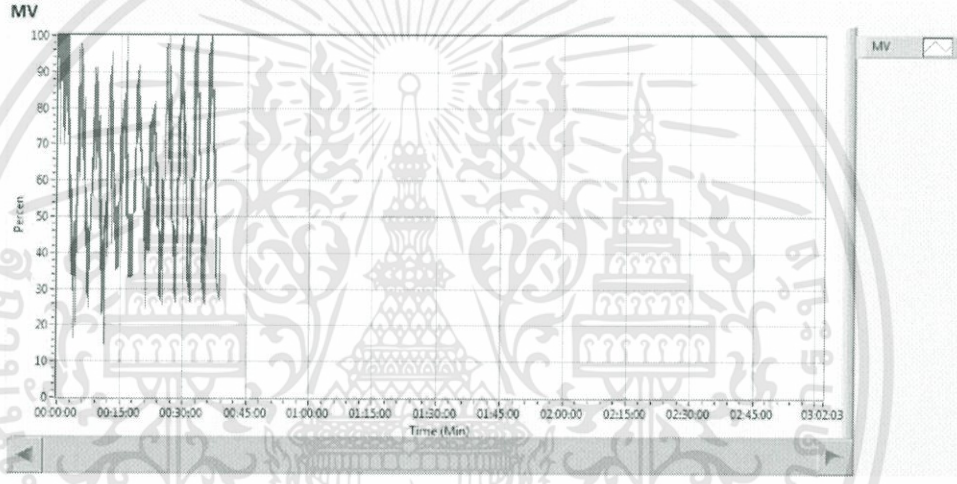
รูปที่ 4.1 แสดงผลของการควบคุมระดับน้ำของกระบวนการจริงที่ระดับน้ำ 50 เปอร์เซ็นต์



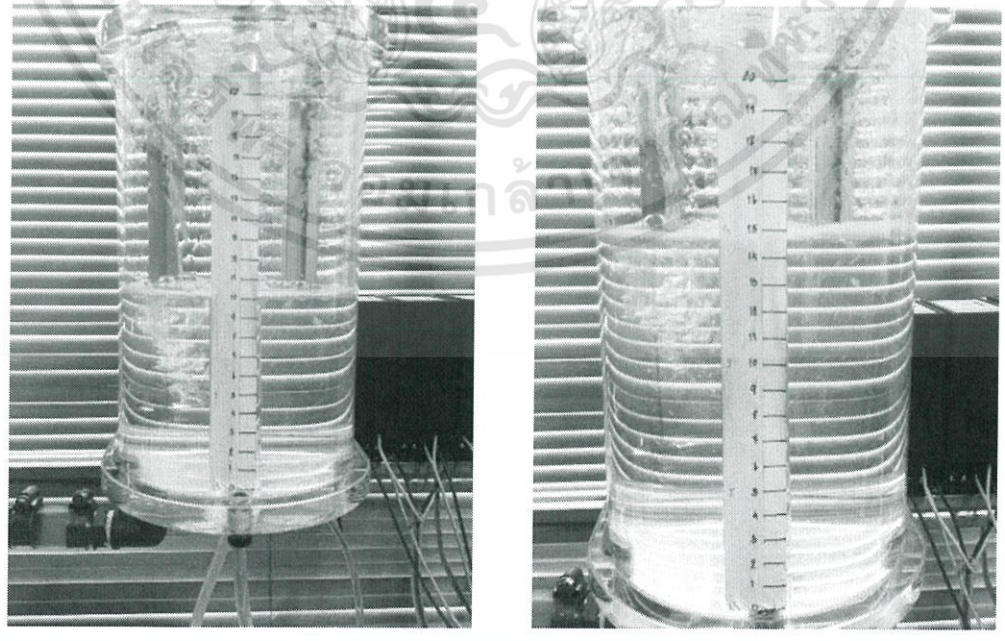
รูปที่ 4.2 แสดงผลของค่า MV ของกระบวนการจริงที่ระดับน้ำ 50 เปอร์เซ็นต์



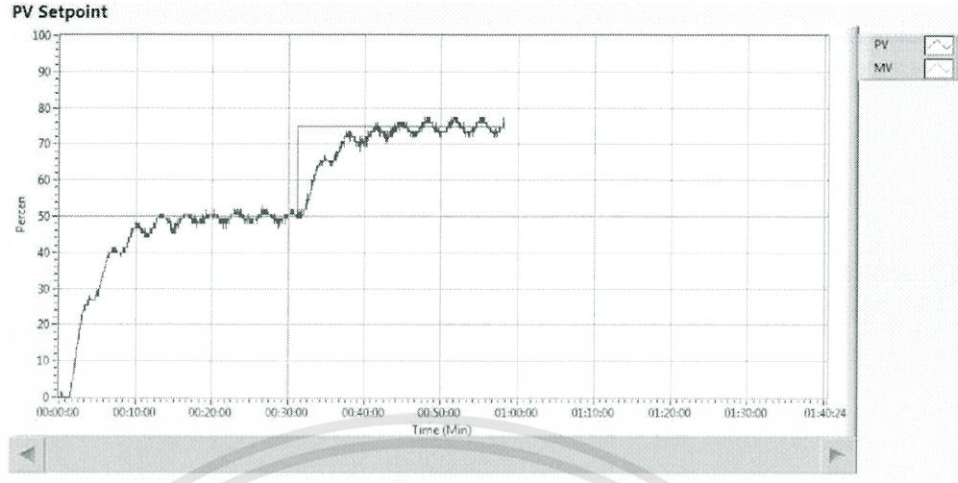
รูปที่ 4.3 แสดงผลของการควบคุมระดับน้ำของกระบวนการจริงที่ระดับน้ำ 75 เปอร์เซ็นต์



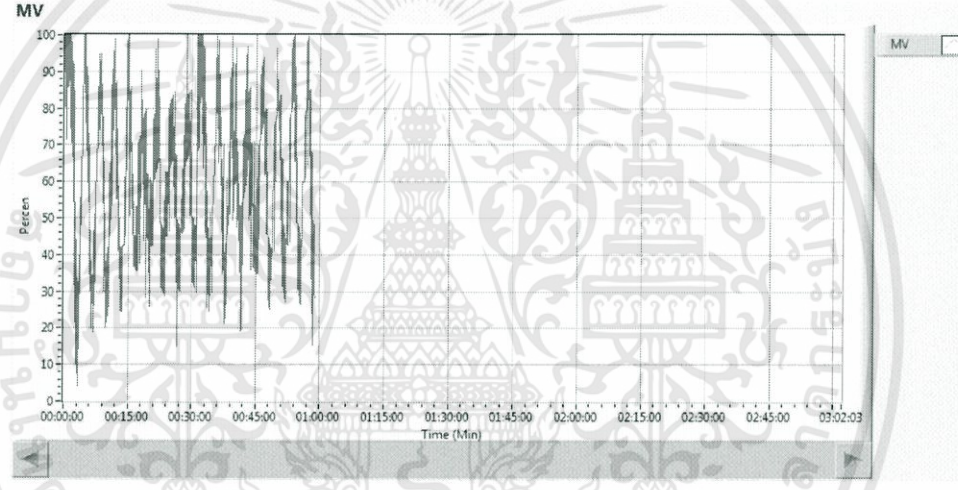
รูปที่ 4.4 แสดงผลของค่า MV ของกระบวนการจริงที่ระดับน้ำ 75 เปอร์เซ็นต์



รูปที่ 4.5 แสดงระดับน้ำในถังของกระบวนการจริงที่ 50เปอร์เซ็นต์ และที่ 75 เปอร์เซ็นต์

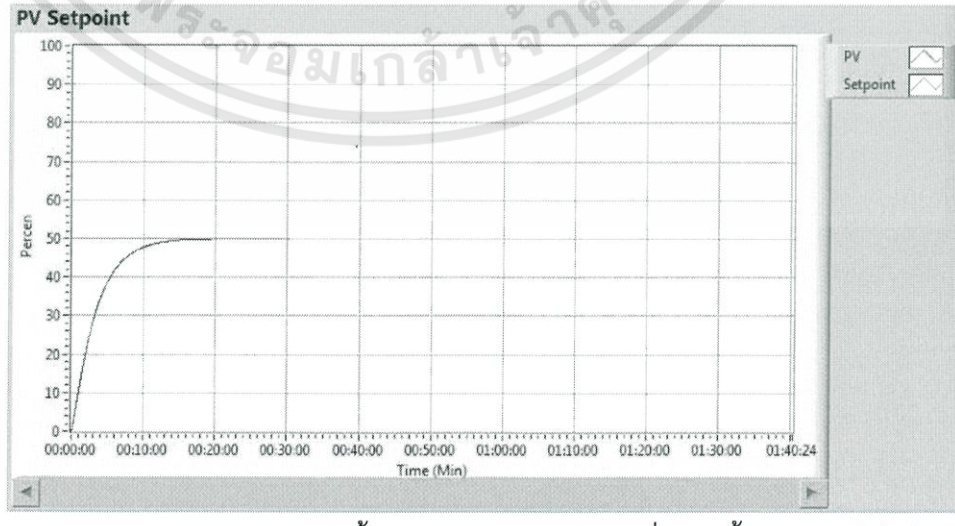


รูปที่ 4.6 แสดงผลของการควบคุมระดับน้ำของกระบวนการจริงเมื่อทำการรบกวนกระบวนการ

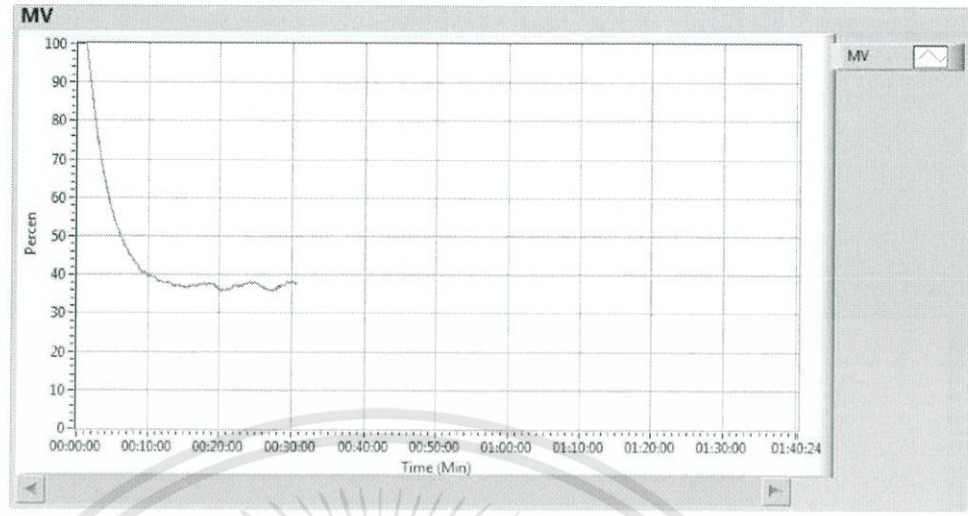


รูปที่ 4.7 แสดงผลของค่า MV ของกระบวนการจริงเมื่อทำการรบกวนกระบวนการ

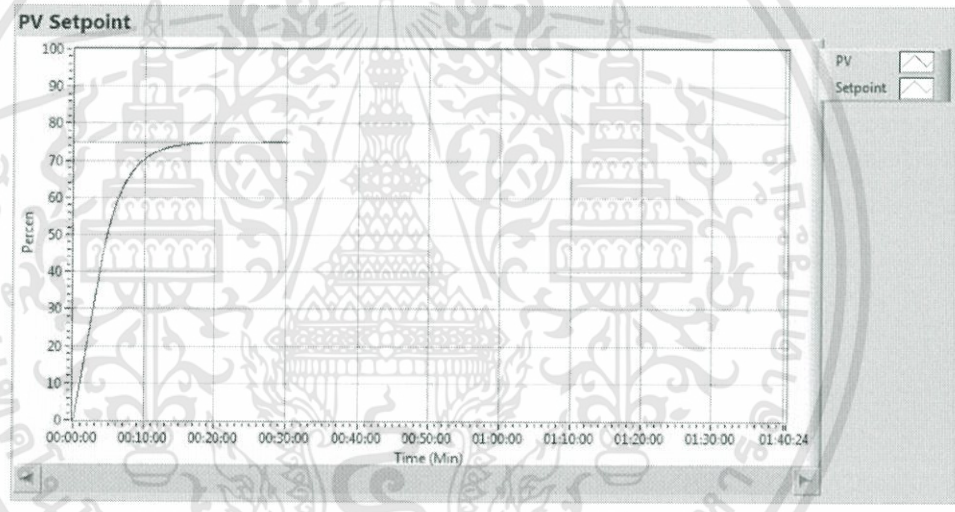
4.3 ผลการทดลองจากกระบวนการจำลอง



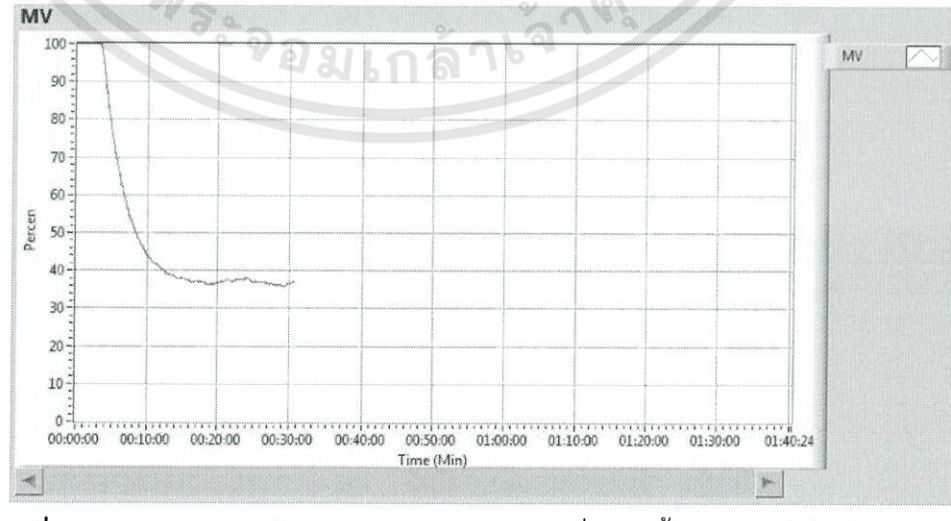
รูปที่ 4.8 แสดงผลของการควบคุมระดับน้ำของกระบวนการจำลองที่ระดับน้ำ 50 เปอร์เซ็นต์



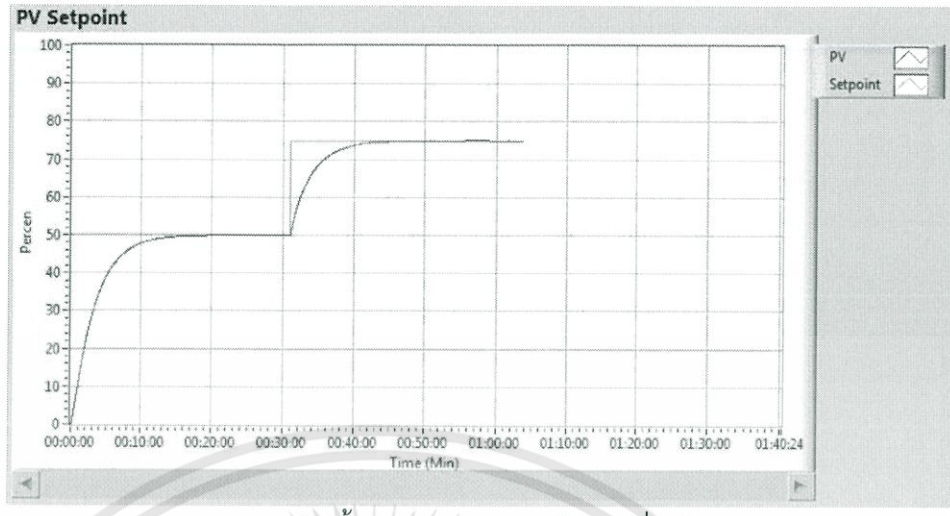
รูปที่ 4.9 แสดงผลของค่า MV กระบวนการจำลองที่ระดับน้ำ 50 เปอร์เซ็นต์



รูปที่ 4.10 แสดงผลของการควบคุมระดับน้ำกระบวนการจำลองที่ระดับน้ำ 75 เปอร์เซ็นต์



รูปที่ 4.11 แสดงผลของค่า MV กระบวนการจำลองที่ระดับน้ำ 75 เปอร์เซ็นต์



รูปที่ 4.12 แสดงผลของการควบคุมระดับน้ำของกระบวนการจำลองเมื่อทำการรบกวนกระบวนการ

บทที่ 5
สรุปผลการทดลองและข้อเสนอแนะ

5.1 สรุปผลการทดลอง

จากการศึกษาและทดลองการออกแบบตัวควบคุมกระบวนการระดับน้ำโดยใช้ LEGO Mindstrom NXT โดยใช้โปรแกรม LabVIEW เป็นตัวควบคุมผ่านคอมพิวเตอร์ไปยังกระบวนการ เทียบกับการจำลองกระบวนการโดยใช้โปรแกรม LabVIEW โดยการทดลองควบคุมโดยใช้คอมพิวเตอร์ สั่งผ่านโปรแกรม LabVIEW ส่งค่าไปที่ LEGO แล้วส่งผ่านบอร์ดขับมอเตอร์เพื่อจ่ายไฟไปขับปั้มมอเตอร์เพื่อปั้มน้ำแล้วควบคุมการไหลของน้ำเพื่อให้ได้ ระดับที่ต้องการโดยอ่านค่าผ่าน D/P transmitter เพื่อส่งค่ากลับมาที่คอมพิวเตอร์ ตามกระบวนการที่เราต้องการควบคุม สามารถสรุปผลได้ดังนี้

1. สามารถควบคุมระดับน้ำตามที่ต้องการได้
2. ในการควบคุมระดับน้ำยังมีค่าความคลาดเคลื่อนอยู่เล็กน้อย เนื่องมาจากความละเอียดของซอฟแวร์บนตัว LEGO Mindstorm NXT ที่นำมาประยุกต์ใช้น้อยเกินไปเมื่อนำมาเชื่อมต่อกับอุปการณ์วัดทางอุตสาหกรรมอย่าง D/P transmitter ทำให้ระดับน้ำไม่เป็นไปตามต้องการ
3. เมื่อนำผลการทดลองจริงมาเปรียบเทียบจากผลการทดลองจากการจำลองกระบวนการจะแสดงให้เห็นว่าประสิทธิภาพการควบคุมโดยใช้ LEGO ยังมีความคลาดเคลื่อนอยู่เล็กน้อย

5.2 ปัญหาและอุปสรรค

1. ปัญหาเกี่ยวกับความร้อนของอุปกรณ์ฮาร์ดแวร์ เช่น ชุดขับปั๊มมอเตอร์ ตัวปั๊มมอเตอร์ และตัว Switching power supply จากการทำงานที่ต่อเนื่องยาวนาน ทำให้เสถียรภาพของอุปกรณ์ลดลงไป
2. บ่อยครั้งมักจะมีเศษผลตะกอน ถูกดูดเข้าไป ทำเกิดการอุดตันในตัวปั๊มน้ำ ซึ่งเป็นสาเหตุให้อัตราการไหลของปั๊มน้ำลดลง แนวทางการแก้ไขปัญหาคือใช้อุปกรณ์กรองน้ำเพื่อกั้นไม่ให้เศษผงตะกอนไหลเข้าไปยังปั๊มได้
3. การเชื่อมต่อ LEGO Mindstrom NXT ผ่าน Bluetooth บางครั้งทำได้ยากมาก แนวทางการแก้ไขปัญหาคือ ไม่ควรเปลี่ยนคอมพิวเตอร์ที่ใช้เชื่อมต่อ
4. ซอฟแวร์บนตัว LEGO Mindstrom NXT ที่ใช้เชื่อมต่อกับ D/P transmitter ไม่สอดคล้องกัน แนวทางแก้ไขปัญหาคือ เขียนโปรแกรมบนโปรแกรม LabVIEW มาช่วย
5. การเชื่อมต่อไร้สายผ่าน Bluetooth อาจมีความผิดพลาดและล้าช้าของข้อมูล ทำให้การควบคุมไม่ได้ผลตามต้องการ แนวทางแก้ไขปัญหาคือ ส่งข้อมูลโดยสายผ่านพอร์ต USB แทน

5.3 ข้อเสนอแนะ

ในการออกแบบกระบวนการนั้นควรจะคำนึงถึงทั้งด้านฮาร์ดแวร์และซอฟต์แวร์ให้มีความเหมาะสมกับโครงงาน โดยเฉพาะในส่วนของฮาร์ดแวร์ที่ต้องออกแบบมาอย่างเหมาะสม เนื่องจากฮาร์ดแวร์จะส่งผลต่อประสิทธิภาพของระบบ