รายการอ้างอิง

1. G. C. Goodwin, P. J. Ramadge, and P. E. Caines. "Discrete-Time Multivariable Adaptive Control," *IEEE Trans. Aut. Control* 25, 3 (1980): 449-456.

2. K. S. Narendra and Y. Lin. "Stable Discrete Adaptive Control," *IEEE Trans. Aut. Control* 25, 3 (1980): 456-461.

3. K. Aström and B. Wittenmark, *Adaptive Control.* MA: Addison–Wesley, 1989.

4. K. J. Aström and P. Persson. "Dominant Pole Design a Unified View of PID Controller Tuning," *Int. Federation of Automatic Control* (1993): 377-382.

5. P. M. J. Hof and R. J. P. Schrama. "An Indirect Method for Transfer Function Estimation from Closed Loop Data," *Automatica* 29, 6 (1993): 1523-1527.

6. P. M. J. Hof and R. J. P. Schrama. "Iterative Identification and Control Design: A Three Step Procedure with Robustness Analysis," *Proc. of the European Cont. Conf.* (1993): 237-241.

7. H. Hjalmarsson, M. Gevers, and S. Gunnarsson. "A Convergent Iterative Restricted Complexity Control Design Scheme," *Proc. IEEE Conf. on Decision and Control* (1994): 1735-1740.

8. H. Hjalmarsson, M. Gevers, and S. Gunnarsson. "Optimality and Sub-Optimality of Iterative Identification and Control Design Schemes," *Proc. American Control Conf.* (1995): 2559-2563.

9. P. M. J. Hof and R. J. P. Schrama. "Identification and Control: Closed Loop Issues," *Automatica* 31, 12 (1995): 1751-1770.

10. Z. Zang, R. Bitmead, and M. Gevers. "Iterative Weighted Least-Squares Identification and Weighted LQG Control Design," *Automatica* 31 (1995): 1577-1594.

11. H. Hjalmarsson, M. Gevers, and F. D. Bruyne. "For Model Based Control Design Criteria Closed Loop Identification gives better Performance," *Automatica* 32 (1996): 1659-1673.

12. H. Hjalmarsson and M. Gevers. "Frequency Domain Expressions of the Accuracy of a Model-Free Control Design Scheme," *Int. Federation of Automatic Control* 1 (1997): 135-140.

13. U. Forssell and L. Ljung. "Closed-Loop Identification Revised," *Automatica* 35 (1999): 1215-1241.

14. X. Bombois, M. Gevers, and G. Scorletti. "A Measure of Robust Stability for an Identified Set of Parametrized Transfer Functions," *IEEE Trans. Aut. Control* 45, 11 (2000): 2141-2145.

15. F. D. Bruyne and L. C. Kammer. "Iterative Feedback Tuning with Guaranteed Stability," *Proc. American Control Conf.* (1999): 3317-3321.

16. U. Forssell and L. Ljung. "A Projection Method for Closed-Loop Identification," *IEEE Trans. Aut. Control* 45, 11 (2000): 2101-2106.

17. K. Hamamoto, T. Fukuda, and T. Sugie. "Iterative Feedback Tuning of Controllers for a Two-Mass Spring System with Friction," *Control and Dynamic Systems* (2000): 2438–2443.

18. S. E. Jo and S. W. Kim. "Normalized Iterative Feedback Tuning with Modified Feedback Experiment," *Proc. American Control Conf.* (2001): 612-617.

19. H. Hjalmarsson. "Model-Free Tuning of Controllers: Experience with Time-Varying Linear Systems," *Proc. of the European Cont. Conf.* (1995): 2869-2874.

20. H. Hjalmarsson, M. Gevers, and S. Gunnarsson. "Model-Free Tuning of a Robust Regulator for a Flexible Transmission System," *Proc. of the European Cont. Conf.* 1 (1995): 148-156.

21. D. Molenaar. "Model-Free Data-Driven Optimal Tuning of Controller Parameters: A Practical Guide with Application Examples," tech. rep., Linkoping University, Sweden, 1995.

22. B. Ceysens and B. Godrons. Synthesis Iterative the Controller sans Identification. Master's thesis, University Catholique de Louvain, 1995.

23. F. D. Bruyne and P. Carrette. "Synthetic Generation of the Gradient for an Iterative Controller Optimization Method," *Proc. of the European Cont. Conf.* (1997)

24. H. Hjalmarsson. "Control of Nonlinear Systems Using Itrative Feedback Tuning," *Proc. American Control Conf.* (1998): 2083-2087.

25. H. Hjalmarsson, M. Gevers, S. Gunnarsson, and O. Lequin. "Iterative Feedback Tuning: Theory and Applications," *IEEE Control Syst. Mag.* 18, 4 (1998): 26-41.

26. H. Hjalmarsson and T. Birkeland. "Iterative Feedback Tuning of Linear Time-Invariant MIMO Systems," *Proc. IEEE Conf. on Decision and Control* (1998): 3893-3898.

27. H. Robbins and S. Monro. "A Stochastic Approximation Method," *Ann. Math. Stat* 22, 6 (1951): 400-407.

28. H. Hjalmarsson. "Iterative Feedback Tuning: an overview," *Int. J. Adapt. Control Signal Process* 16 (2002): 373-395.

29. G. Nash and A. Sofer, *Linear Nonlinear Programming*. Singapore: Mc Graw-Hill International Editions, 1996.

30. S. Boyd and L. Vandenberghe, *Convex Optimization*. UK: Cambridge University, 2004.

31. J. Songsiri and D. Banjerdpongchai. "Dynamic Models of Servo-Driven Conveyor System," *Proc. of the 2004 IEEE Region 10 Conf. on Analog and Digital Techniques in Electrical Engineering* D (2004): 538-541.

32. C. Canudas de Wit and H. Olsson and K. J. Åström and P. Lischinsky. "A New Model for Control of Systems with Friction," *IEEE Trans. Aut. Control* 40 (Mar 1995).

33. R. Kelly, J. Llamas, and R. Campa. "A Measurement Procedure for Viscous and Coulomb Friction," 49 (Aug 2000).

34. A. Wahyudie and D. Banjerdpongchai. Robust PID Controller Design for Belt Conveyor System. Master's thesis, Chulalongkorn University, 2004.

ภาคผนวก

ภาคผนวก ก

การพัฒนาโปรแกรมสำหรับระบบสายพานลำเลียง

ก.1 โปรแกรม LabVIEW

LabVIEW คือ เครื่องมือที่ใช้ในการพัฒนาโปรแกรมประยุกต์ (Application program) ชนิดหนึ่งเช่นเดียวกับ Visual Basic และ Visual C++ แต่จะเป็นการเขียนโปรแกรมโดยใช้รูปในการพัฒนา (Graphical based programming) ซึ่งจะแตกต่างจากแนวคิดแบบการเขียนโปรแกรมโดยใช้ข้อความ (Text based programming) เช่น การเขียนโปรแกรมโดยใช้ข้อความจะทำงานบนลงล่าง แต่ LabVIEW จะทำงานแบบกระแสข้อมูล (Dataflow)

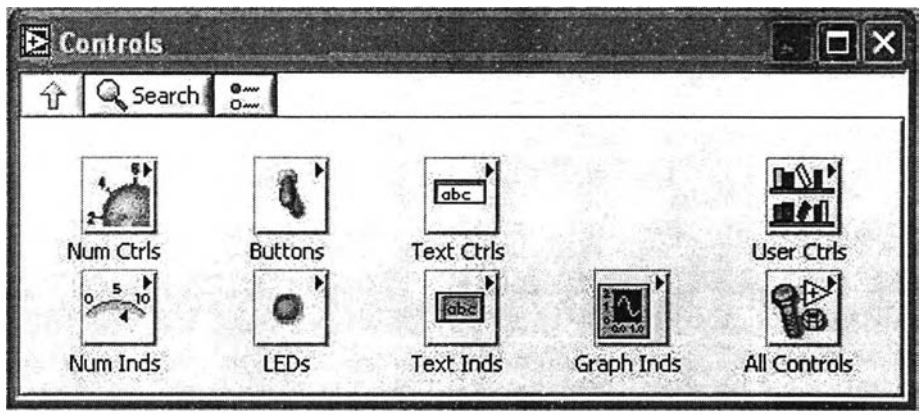
LabVIEW สามารถเขียนเป็นสมการต่างๆ ที่ซับซ้อนได้เหมือนการเขียนโปรแกรมโดยใช้ข้อความ เช่น ภาษาซี (C Language) โดยใช้จุดต่อสูตร (Formula node) และ LabVIEW การทำงานจะเน้นทางการติดต่อกับเครื่องมือ (Tools) หรือตัวแปรสัญญาณ (Transducer) ที่ใช้วัดสัญญาณทางกายภาพต่างๆ และนอกจากนั้นสามารถนำประโยชน์จากเทคโนโลยีคอมพิวเตอร์ (Computer technology) และเทคโนโลยีโครงข่าย (Network technology) มาประยุกต์ใช้งาน เช่น เก็บข้อมูลลงในฐานข้อมูล (Database) หรือส่งผ่านข้อมูลทางอินเทอร์เน็ต (Internet) ทำให้หลายอุตสาหกรรมได้นำเอา LabVIEW มาประยุกต์ใช้งานกับงานวัดควบคุมและงานอัตโนมัติ (Automation)

ที่มาของ LabVIEW

LabVIEW ย่อมาจาก Laboratory Virtual Instrument Engineering Workbench โปรแกรมที่พัฒนาขึ้นโดยใช้ LabVIEW จะเรียกว่าเครื่องมือวัดเสมือน (Virtual instrument) หรือเรียกย่อๆ ว่า VI

LabVIEW จะมีแผงด้านหน้า (Front panel) ซึ่งเปรียบเสมือนได้กับสิ่งที่ผู้ใช้จะเห็นและควบคุมการทำงาน ผู้ใช้สามารถสร้างรูปแบบขึ้นเองได้อย่างรวดเร็วเพราะ LabVIEW มีส่วนประกอบต่างๆ ที่ใช้สำหรับออกแบบหน้าจอมากมาย เช่น จอแสดงผลแบบออสซิลโลสโคป (Oscilloscope) ปุ่มหมุน (Dial) และสวิตช์ (Switch) เป็นต้น โดย LabVIEW จะแสดงผลและควบคุมการทำงานผ่านทางคอมพิวเตอร์ พื้นที่ส่วนที่เขียนโปรแกรมจะเรียกว่าแผนภาพบล็อก (Block diagram) เปรียบเสมือนกับส่วนอุปกรณ์ (Hardware) ภายในเครื่องมือวัด โดย LabVIEW จะเขียนโปรแกรมโดยอาศัยรูปภาพ LabVIEW อาศัยหลักการทำงานของเครื่องมือวัดหรือการวัดคุมทำให้ผู้ใช้สามารถออกแบบรูปแบบโปรแกรมตามที่ผู้ใช้ต้องการ หลักการดังกล่าวแบ่งออกเป็น 3 ส่วนหลักๆ คือ

1. การรับ (Acquisition) ซึ่งเป็นส่วนรับข้อมูลจากสิ่งแวดล้อมภายนอกเข้าสู่ระบบ ในที่นี้คือ คอมพิวเตอร์ โดยข้อมูลที่เข้าสู่ระบบนี้อาจจะมาจากการ์ดการรับข้อมูล (Data acquisition) สำหรับสัญญาณทางไฟฟ้า การ์ดการรับรูปภาพ (Image acquisition) เป็นต้น



รูปที่ ก.1: เครื่องมือแผงด้านหน้า

2. การวิเคราะห์ (Analysis) หลังจากที่ได้รับข้อมูลแล้วอาจจะผ่านฟังก์ชันในการวิเคราะห์ ซึ่งจะแสดงผลในรูปที่สื่อความหมายในสิ่งที่ผู้ใช้งานสามารถนำไปแสดงแทนสื่อที่วัดได้และใช้งานได้

3. การแสดงผล (Presentation) คือการแสดงผลในรูปแบบที่เป็นประโยชน์ต่อผู้ใช้งาน โดยอาจแสดงผลบนหน้าจอคอมพิวเตอร์เช่น เชิงเลขหลายมาตรา (Digital multimeter) แสดงผลเฉพาะสัญญาณที่วัดได้โดยไม่จำเป็นต้องรู้ความสัมพันธ์กับเวลา ออสซิลโลสโคปแสดงผลของข้อมูลที่สัมพันธ์กับเวลา ตัววิเคราะห์สเปกตรัม (Spectrum analyzer) จะแสดงผลสัญญาณในรูปความถี่หรือการพิมพ์ออกมาเป็นรายงานหรือเก็บข้อมูลในจานแข็ง (Hard disk)

ส่วนประกอบต่างๆ ใน LabVIEW

1. แผงด้านหน้า คือส่วนที่จะติดต่อกับผู้ใช้งาน (User interface) ซึ่งส่วนนี้จะประกอบด้วย ส่วนที่รับข้อมูลจากผู้ใช้ (Control input) และส่วนแสดงผลให้ผู้ใช้เห็น (Output indicator)

2. ส่วนควบคุม คือสิ่งที่ผู้ใช้งานจะป้อนค่าหรือเปลี่ยนแปลงค่าได้ ซึ่งก็คือส่วนที่รับข้อมูลจากผู้ใช้เข้าสู่ระบบในรูปตัวรับข้อมูลแบบตัวเลข (Numeric control)

3. ตัวชี้บอก เป็นสิ่งที่โปรแกรมแสดงผลออกมาให้ผู้ใช้งานเห็น ซึ่งก็คือส่วนที่แสดงออกมาจากระบบหรือการประยุกต์ที่เราพัฒนา

4. แผนภาพบล็อก คือส่วนที่ผู้พัฒนาใช้ในการเขียนโปรแกรม หรือส่วนของรหัสต้นฉบับ (Source code)

5. เครื่องปลายทาง (Terminal) จะมีสองรูปแบบคือ แหล่งต้นทาง (Source) และแหล่งปลายทาง (Sink)

6. สัญรูป (Icons) คือส่วนที่มีการทำงานอย่างใดอย่างหนึ่งเมื่อโปรแกรมเริ่มทำงาน เช่นฟังก์ชันซึ่งเป็นส่วนประกอบพื้นฐานที่มีอยู่แล้วใน LabVIEW ได้แก่ บวก (Add) ลบ (Subtract) เครื่องมือวัดเสมือนย่อย (SubVIs) ซึ่งหมายถึงเครื่องมือวัดเสมือนที่ถูกเรียกอ่านจากอีกเครื่องมือวัดเสมือนหนึ่ง

7. สาย (Wires) คือเส้นทางของข้อมูลที่ส่งผ่านจุดกำเนิดข้อมูลไปยังจุดรับข้อมูลหรือจุดสิ้นสุด

รูปที่ ก.2: ส่วนเครื่องมือที่ใช้สำหรับการออกแบบแผงด้านหน้า

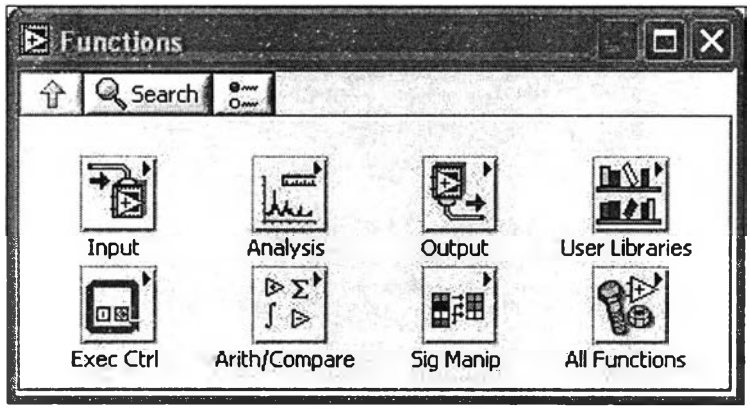
8. โครงสร้าง คือส่วนที่ควบคุมขั้นตอนการทำงานของโปรแกรม เช่น วงวน (Loop) เป็นต้น
9. จุดต่อ คือจุดเชื่อมต่อระหว่างข้อมูลกับเครื่องมือวัดเสมือน ฟังก์ชัน โครงสร้าง เป็นต้น

ก.2 เครื่องมือในการออกแบบแผงด้านหน้า

เครื่องมือที่ใช้ในการออกแบบแผงด้านหน้าจะใช้ส่วนควบคุม (Controls palette) และส่วนเครื่องมือ (Tools palette) LabVIEW มีส่วนควบคุมที่ใช้ในการออกแบบแผงด้านหน้าดังรูปที่ ก.1 ซึ่งเป็นส่วนที่ติดต่อกับผู้ใช้งาน โดยจะจัดเป็นกลุ่มๆ เช่น กลุ่มของตัวเลข ซึ่งภายในกลุ่มจะมีส่วนควบคุมและส่วนตัวชี้บอกต่างๆ ที่เกี่ยวกับตัวเลข เป็นต้น

ส่วนเครื่องมือสำหรับการออกแบบแผงด้านหน้า คือเครื่องมือที่ใช้ในการพัฒนาโปรแกรม ซึ่งจะใช้ทั้งการออกแบบแผงด้านหน้าและแผนภาพบล็อก ดังแสดงในรูปที่ ก.2 ส่วนเครื่องมือที่ใช้บ่อยๆ ในการออกแบบแผงด้านหน้า

1. เครื่องมือทำงานกับค่า (Value tool) ใช้เปลี่ยนแปลงค่าของส่วนควบคุมหรือส่วนตัวชี้บอก โดยการเลื่อนไปที่ค่าของส่วนควบคุมหรือส่วนชี้บอก ที่เราต้องการเปลี่ยนแปลงค่าแล้วทำการเลือกเพื่อเปลี่ยนค่า
2. เครื่องมือทำงานกับตำแหน่ง/ขนาด/เลือก (Position/Size/Select tool) ใช้สำหรับเลือกหรือจัดวางตำแหน่งใหม่ หรือปรับขนาดของส่วนควบคุมหรือส่วนตัวชี้บอก
3. เครื่องมือตรวจแก้ข้อความ (Edit text tool) ใช้ในการแก้ไขข้อความที่เป็นตัวอักษรหรือเพิ่มข้อความลงบนแผงด้านหน้า
4. เครื่องมือตั้งค่าสี (Set color tool) ใช้ในการเปลี่ยนแปลงสีที่เราต้องการเปลี่ยนสี ซึ่งสามารถเปลี่ยนสีทั้งสีด้านบน (Foreground) ของสิ่งใดๆ และสีพื้น (Background) ถ้าสิ่งนั้นมีสีพื้น
5. เครื่องมือเมนูลัด (Shortcut menu tool) ใช้สำหรับแสดงและเลือกเมนูที่เกี่ยวข้องกับสิ่งต่างๆ บนแผงด้านหน้าและแผนภาพบล็อก



รูปที่ ก.3: เครื่องมือที่ใช้ในการเขียนโปรแกรมบนแผนภาพบล็อก

6. เครื่องมือแถบเลื่อนหน้าต่าง (Scroll window tool) ใช้สำหรับการเลื่อนหน้าต่างทั้งแผงด้านหน้าและแผนภาพบล็อก
7. เครื่องมือจัดแจงสี (Get color tool) เป็นเครื่องมือที่ใช้ในการคัดลอก (Copy) สีที่เราเลือกจากวัตถุหนึ่งเพื่อที่จะนำไปใช้ในการเปลี่ยนสีของอีกวัตถุหนึ่งให้มีสีเหมือนกับวัตถุที่เราคัดลอกสีมา โดยใช้เครื่องมือเกี่ยวกับสี

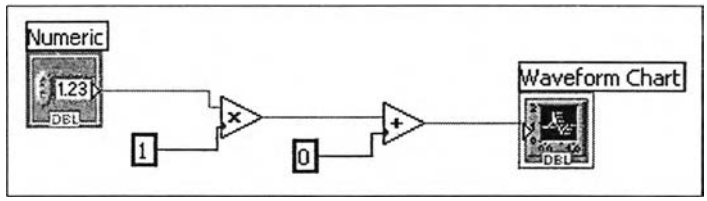
เครื่องมือที่ใช้ในการเขียนโปรแกรมบนแผนภาพบล็อกใน LabVIEW ใช้ส่วนฟังก์ชัน (Function palette) ดังแสดงในรูปที่ ก.3 ซึ่งจะมีฟังก์ชันและเครื่องมือวัดเสมือนย่อยต่างๆ ที่มีอยู่แล้วให้ผู้ใช้เลือกใช้ โดยฟังก์ชันและเครื่องมือวัดเสมือนย่อยจัดเป็นกลุ่มๆ เช่น ฟังก์ชันแบบตัวเลข (Numeric function) จะมีฟังก์ชันต่างๆ เกี่ยวกับตัวเลข เช่น บวก ลบ คูณ และหาร เป็นต้น

ส่วนเครื่องมือสำหรับสำหรับการเขียนโปรแกรมบนแผนภาพบล็อก คือเครื่องมือที่ใช้ในการเขียนโปรแกรม ส่วนเครื่องมือที่ใช้ในการเขียนโปรแกรมบนแผนภาพบล็อก

1. เครื่องมือทำงาน (Operating tool) ใช้ในการเปลี่ยนแปลงค่าหรือเลือกค่าคงที่ในแผนภาพบล็อก
2. เครื่องมือตำแหน่ง/ขนาด/เลือก ใช้ในการเลือก เคลื่อนย้าย และจัดขนาดของสิ่งที่สร้างขึ้นบนแผนภาพบล็อก
3. เครื่องมือตรวจแก้ข้อความ ใช้ในการแก้ไขข้อความที่เป็นตัวอักษรหรือเพิ่มข้อความลงบนแผนภาพบล็อก
4. เครื่องมือต่อสาย ใช้ในการโยงสายระหว่างเครื่องปลายทางหรือจุดต่อ ซึ่งสายที่ต่อนี้จะเป็นเส้นทางเดินของข้อมูล

ก.3 การพัฒนาโปรแกรม

ก่อนจะเริ่มพัฒนาโปรแกรมจะต้องรู้หลักการการทำงานของโปรแกรมในรูปทำงานแบบกระแสข้อมูล และรูปแบบข้อมูล (Data type) ซึ่งเป็นสิ่งสำคัญในการพัฒนาโปรแกรม



รูปที่ ก.4: การเขียนโปรแกรมแผนภาพบล็อกแบบกระแสข้อมูล

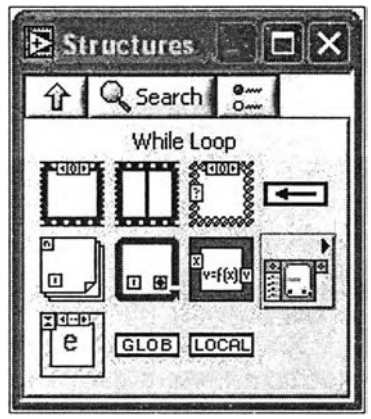
หลักการทำงานของการเขียนโปรแกรมแบบกระแสข้อมูล โปรแกรมที่เขียนขึ้นด้วย LabVIEW จะทำงานโดยอาศัยหลักการของกระแสข้อมูล ซึ่งมีหลักการดังต่อไปนี้

1. ฟังก์ชันหรือเครื่องมือวัดเสมือนย่อย จะทำงานเมื่อมีข้อมูลเข้า (Data input)
2. ฟังก์ชันหรือเครื่องมือวัดเสมือนย่อยทำงานเสร็จ จะให้ข้อมูลออก (Data output) ไปยังฟังก์ชันหรือเครื่องมือวัดเสมือนย่อยอื่นๆ ที่ต้องการข้อมูล
3. ข้อมูลจะถูกส่งผ่านโดยสาย

จากรูปที่ ก.4 ขั้นตอนการทำงานของโปรแกรมเริ่มจากฟังก์ชันคูณจะทำงานก่อนฟังก์ชันบวกเพราะฟังก์ชันคูณมีข้อมูลเข้าพร้อม แต่ฟังก์ชันบวกต้องรอข้อมูลออกจากฟังก์ชันคูณ หลังจากฟังก์ชันคูณทำงานเสร็จจะส่งผ่านข้อมูลไปยังฟังก์ชันบวก ฟังก์ชันบวกจะทำงานเพราะมีข้อมูลพร้อม หลังจากฟังก์ชันบวกทำงานจะส่งผลลัพธ์ไปให้เครื่องปลายทาง

รูปแบบข้อมูล เช่นเดียวกับภาษาโปรแกรมอื่นๆ เช่น ภาษาซีที่มีรูปแบบของข้อมูล รูปแบบของข้อมูลใน LabVIEW ที่สำคัญจะประกอบด้วย ตัวเลข บูลีน (Boolean) สายอักขระ (String) และรูปคลื่น (Waveform) ซึ่งแต่ละรูปแบบของข้อมูลจะมีการแยกรหัสสี (Color code) ให้แตกต่างกัน ซึ่งแต่ละรูปแบบข้อมูลจะมีฟังก์ชันที่แยกตามประเภทกัน รวมทั้งสีของสายต่อจะขึ้นอยู่กับรูปแบบข้อมูล รูปแบบของข้อมูลมีความสำคัญในการเขียนโปรแกรม เพราะฟังก์ชันหรือเครื่องมือวัดเสมือนย่อยรับรูปแบบข้อมูลเข้าและข้อมูลออกแตกต่างกันไป เช่น ฟังก์ชันคูณรับข้อมูลเข้าประเภทตัวเลขหรือแถวลำดับ (Array) ของตัวเลข ซึ่งถ้านำเอาข้อมูลประเภทสายอักขระมาป้อนด้านข้อมูลเข้าแล้ว LabVIEW จะแสดงเส้นเป็นเส้นประ ซึ่งเราสามารถดูคำอธิบายได้โดยการเลื่อนไปเหนือเส้นประนั้น

รูปแบบข้อมูลแบบตัวเลข (Numeric datatypes) LabVIEW มีรูปแบบการรับข้อมูลเข้าและแสดงผลชนิดแบบตัวเลขหลายรูปแบบ ซึ่งแต่ละตัวถูกกำหนดเป็นรูปแบบการรับข้อมูลหรือรูปแบบแสดงผลไว้แล้ว แต่เราสามารถเปลี่ยนจากรูปแบบการรับข้อมูลเป็นรูปแบบแสดงผล หรือรูปแบบแสดงผลเป็นรูปแบบการรับข้อมูลได้ นอกจากนั้นยังมีคุณสมบัติอื่นๆ อีกที่สามารถเปลี่ยนแปลงได้ โดยการเลือกที่วัตถุ (Object) นั้นแล้วเลือกจากคุณสมบัติของวัตถุนั้นๆ ฟังก์ชันหรือเครื่องมือวัดเสมือนย่อยที่เกี่ยวกับตัวเลขสามารถเลือกได้หลายๆ ฟังก์ชัน โดยแต่ละฟังก์ชันใช้รูปแบบข้อมูลแตกต่างกันไป ในกรณีที่สมการคำนวณซับซ้อนมากขึ้น การใช้ฟังก์ชันตัวเลขหลายๆ ตัวมาต่อกันอาจจะทำให้เกิดความเข้าใจการทำงานของสมการต่างๆ ยากขึ้น หรือเขียนโปรแกรมเร็วขึ้นถ้าเขียนเป็นสมการแบบการเขียนโปรแกรมโดยใช้ข้อความ



รูปที่ ก.5: โครงสร้างการเขียนโปรแกรมวงวน

รูปแบบข้อมูลแบบบูลีน (Boolean datatypes) รูปแบบข้อมูลแบบบูลีนเป็นรูปแบบข้อมูลเข้าและข้อมูลออกซึ่งมีสองสถานะคือ เปิดกับปิด (On off) หรือ จริงกับเท็จ (True false) นอกเหนือจากค่าเปิดกับปิดหรือจริงกับเท็จแล้ว LabVIEW สามารถกำหนดการทำงานควบคุมสวิตช์ (Switch control) ให้มีการทำงานทางกลศาสตร์คล้ายกับสวิตช์จริง

การสร้างและการเลือกใช้เครื่องวัดเสมือนย่อย เครื่องวัดเสมือนย่อย คือเครื่องวัดเสมือนที่เราสร้างขึ้นเพื่อที่จะสามารถนำมากลับใช้ใหม่ได้ โดยสามารถเลือกใช้จากเครื่องมือวัดเสมือนอื่นๆ เครื่องมือวัดเสมือนย่อยคล้ายกับรูทีนย่อย (Subroutine) หรือฟังก์ชันในแบบการเขียนโปรแกรมโดยใช้ข้อความ โดยการสร้างเครื่องมือวัดเสมือนย่อยจะมี 2 วิธีคือ การสร้างจากเครื่องมือวัดเสมือนให้เป็นเครื่องมือวัดเสมือนย่อย และการสร้างเครื่องมือวัดเสมือนย่อยจากเครื่องมือวัดเสมือนที่กำลังพัฒนาอยู่

การเรียกใช้เครื่องวัดเสมือนย่อย สามารถเรียกใช้เครื่องวัดเสมือนย่อยมาใช้ได้ 2 วิธีโดยการเปิดเครื่องวัดเสมือนย่อยที่เราต้องการเรียกใช้ตามปกติ เลือกที่สัญกรแล้วลากมาวางบนแผนภาพบล็อกของโปรแกรมที่จะเรียกใช้เครื่องวัดเสมือนย่อยนั้น หรือเลือกเครื่องวัดเสมือนย่อยจากฟังก์ชัน

การแก้จุดบกพร่อง (Debug) เครื่องวัดเสมือน การแก้จุดบกพร่องเครื่องวัดเสมือนคือการตรวจสอบว่าการทำงานของเครื่องวัดเสมือนเป็นไปตามที่เราได้ออกแบบไว้ หรือมีความผิดพลาดของการทำงานที่ใดบ้าง

เครื่องมือที่ใช้สำหรับการแก้จุดบกพร่องเครื่องมือวัดเสมือน เครื่องมือจุดหยุด (Break point tool) คือ จุดที่เรากำหนดว่าเมื่อโปรแกรมวิ่ง (Run) มาถึงจุดนี้ให้หยุดการทำงานเพื่อที่เราจะทำการตรวจสอบขั้นตอนการทำงานของโปรแกรม เครื่องมือหัวตรวจ (Probe tool) คือ ข้อมูลที่เราต้องการดู ณ จุดต่างๆ กัน

การเขียนโปรแกรมวงวน (While loop) โปรแกรมวงวน คือการกำหนดให้รหัสหรือโปรแกรมในวงวนทำงานวนซ้ำจนกระทั่งเงื่อนไข (Condition) ของวงวนเป็นไปตามที่กำหนด ดังแสดงในรูปที่ ก.5

การใช้ฟังก์ชันเวลาในการเขียนโปรแกรมวงวน ฟังก์ชันที่ใช้บ่อยในวงวนเป็นฟังก์ชันที่กำหนดเวลาให้วนซ้ำทุกๆ มิลลิวินาที (Milisecond) คือรอ (Wait ms) และ รอจนกระทั่ง (Wait until next ms mutiple) ความแตกต่างระหว่างฟังก์ชันรอกับรอจนกระทั่งคือ ฟังก์ชันรอจะรอตามจำนวนมิลลิวินาทีที่ป้อน

เข้าไป หลังจากทำงานในส่วนของรหัสใช้เวลา 10 มิลลิวินาที แล้วจะรออีก 500 มิลลิวินาทีแล้วจึงจะวนใหม่ ทำให้แต่ละวงวนจะใช้เวลา 10+500 มิลลิวินาที ส่วนฟังก์ชันรอจนกระทั่ง สมมุติว่ารหัสในวงวนใช้เวลา 10 มิลลิวินาที เช่นเดียวกันกับฟังก์ชันรอ แต่การรอจริงๆ จะใช้เวลา 490 มิลลิวินาทีก่อนที่จะวนใหม่ ทำให้ในแต่ละวงวนใช้เวลา 500 มิลลิวินาที หรือ 10 มิลลิวินาที ในรหัสและ 490 มิลลิวินาที สำหรับฟังก์ชันรอจนกระทั่ง

เรจิสเตอร์แบบเลื่อน (Shift register) เรจิสเตอร์แบบเลื่อน หรือ จุดต่อป้อนกลับ (Feedback node) ใช้ร่วมกับการเขียนโปรแกรมวงวน สำหรับการเก็บค่าไว้เพื่อใช้ขึ้นการวนซ้ำครั้งต่อไป เรจิสเตอร์แบบเลื่อนสามารถสร้างได้โดยเลือกที่ขอบของการเขียนโปรแกรมวงวน

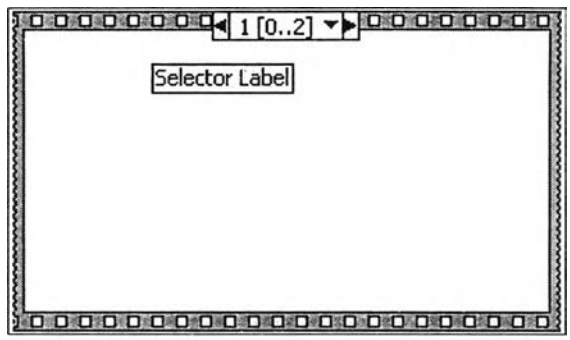
จุดต่อป้อนกลับ จุดต่อป้อนกลับจะคล้ายกับเรจิสเตอร์แบบเลื่อน แต่ใช้แทนกันในกรณีที่จะลดการลากสายที่ยาวและซับซ้อนและไม่มีการเรียงทับซ้อน (Stack) ของเรจิสเตอร์แบบเลื่อน สามารถเปลี่ยนการป้อนกลับเป็นเรจิสเตอร์แบบเลื่อนได้โดยการเลือกที่จุดต่อป้อนกลับแล้วเลือกแทนที่ (Replace)

ลำดับการเรียงทับซ้อน (Stacked sequence) และโครงสร้างลำดับเรียบ (Flat sequence structure) ลำดับเป็นการควบคุมการไหลของโปรแกรมแบบตามลำดับเฟรม (Frame) คล้ายกับฟิล์ม (Film) รูปซึ่งโปรแกรมจะเริ่มทำงานตั้งแต่เฟรมลำดับที่ 0 ถัดไปเป็นเฟรมลำดับที่ 1 ถัดไปเป็นเฟรมลำดับที่ 2 ซึ่งการไหลของโปรแกรมไม่สามารถย้อนกลับกลับไปยังเฟรมก่อนหน้าได้ เช่น ถ้าเฟรมลำดับที่ 2 ทำงานเสร็จ ก็จะไม่สามารถย้อนกลับไปทำงานลำดับที่ 1 ต่อไปได้ ลำดับการเรียงทับซ้อนกับลำดับเรียบมีการทำงานเหมือนกันเพียงแต่ว่า ลำดับเรียบนั้นสามารถมองเห็นรหัสในทุกๆ ลำดับได้ แต่ลำดับการเรียงทับซ้อนจะช่วยเพิ่มพื้นที่ในการเขียนโปรแกรมได้มากขึ้น การเพิ่มลำดับสามารถทำได้โดยการเลือกที่ขอบของลำดับแล้วเลือกเพิ่มเฟรมก่อนหรือเพิ่มเฟรมหลัง การเพิ่มเฟรมหลังจะทำการเพิ่มเฟรมลำดับถัดไป เช่น ถ้าอยู่ที่เฟรมลำดับที่ 2 แล้วเลือกเพิ่มเฟรมหลัง LabVIEW จะสร้างเฟรมลำดับที่ 3 ให้ การเพิ่มเฟรมก่อนจะทำการเพิ่มเฟรมลำดับก่อนหน้าลำดับที่เราอยู่ปัจจุบัน ซึ่งหลังจากเพิ่มเฟรมแล้วลำดับที่เราอยู่ปัจจุบันจะถูกจัดไปเป็นลำดับถัดไป เช่น ถ้าเราอยู่ที่ลำดับเฟรมที่ลำดับที่ 2 แล้วเลือกที่เพิ่มเฟรมก่อน LabVIEW จะทำการเพิ่มเฟรมลำดับที่ 2 และย้ายลำดับที่ที่เราอยู่ปัจจุบันเป็นลำดับที่ 3 เราสามารถดูลำดับต่างๆ ของลำดับทั้งหมด โดยการเลือกที่ตัวเลือกเฟรม (Frame selector) แล้วเลือกแสดงเฟรม (Show frame) ดังแสดงในรูปที่ ก.6

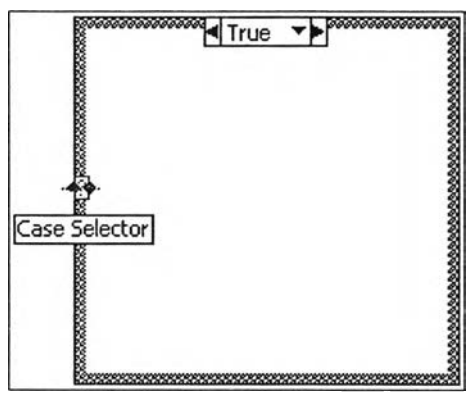
ในการส่งค่าจากเฟรมหนึ่งไปอีกเฟรมหนึ่ง หน้าถัดๆ ไปแต่ไม่ใช่เฟรมก่อนหน้าของลำดับการเรียงทับซ้อนได้โดยการเลือกที่ขอบของลำดับ แล้วเลือกเพิ่มลำดับเฉพาะที่ (Add sequence local)

โครงสร้างกรณี (Case structure) โครงสร้างกรณีใช้ในการเลือกการทำงานเฉพาะรหัสที่อยู่ในแต่ละกรณี โดยการเปรียบเทียบดูว่า ค่าที่เข้ามาในตัวเลือกดังแสดงในรูปที่ ก.7 เท่ากันกับค่ากรณี ค่าที่ใช้ในการเปรียบเทียบตัวเลือกและกรณี อาจจะเป็นค่าบูลีน ตัวเลข ตัวอักษร หรืออื่นๆ ก็ได้ ไม่จำเป็นต้องเป็นค่าบูลีนเสมอไป ซึ่งค่าที่ใช้ในการเปรียบเทียบอาจจะเป็นค่าเดียว ช่วงของค่า หรือค่าหลายๆ ค่าก็ได้

ช่วงของค่าที่สามารถใช้ ".." เป็นตัวกำหนด เช่น 1..100 หมายถึงช่วงค่าระหว่าง 1 ถึง 100 หรือ ..1 หมายถึงน้อยกว่า 1 เป็นต้น ถ้ามีหลายค่าสามารถใช้ "," ในการกำหนดค่า เช่น 1, 5, 7 หมายถึงค่า



รูปที่ ก.6: การดูลำดับต่างๆ ของลำดับทั้งหมด



รูปที่ ก.7: ค่าที่เข้ามาในตัวเลือก

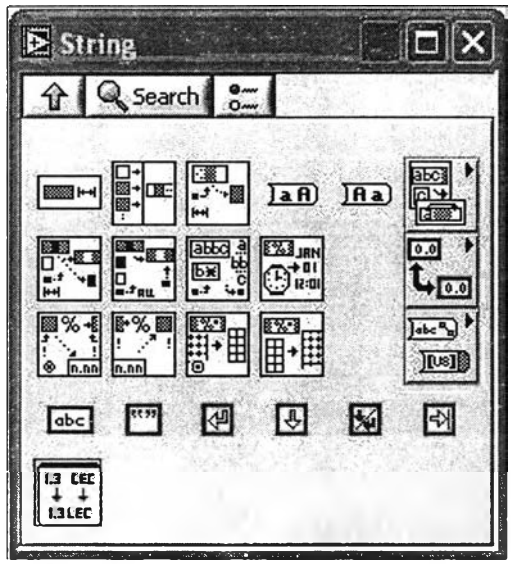
1, 5 หรือ 7 เป็นต้น สามารถเพิ่มกรณีใหม่ได้โดยการเลือกที่ขอบของโครงสร้างกรณีและเลือกที่เพิ่มกรณีก่อนหรือเพิ่มกรณีหลัง

รูปแบบข้อมูลแบบสายอักขระ (String datatype) สายอักขระหมายถึงตัวอักษรตั้งแต่หนึ่งตัวอักษรขึ้นไปมาประกอบรวมกัน การแสดงรูปแบบข้อมูลแบบสายอักขระมีได้ด้วยกันหลายรูปแบบซึ่งสามารถกำหนดได้โดย เลือกที่ส่วนควบคุมสายอักขระ (String control) หรือส่วนแสดงผลดังแสดงในรูปที่ ก.8 ซึ่งอาจจะเลือกการแสดงผลเป็นแบบรหัสผ่าน หรือเลขฐาน 16

ส่วนรูปแบบชุดแบบอักษร (Font) สามารถแก้ไขได้จากการตั้งค่าข้อความ (Text setting) บนแถบเครื่องมือ

คุณสมบัติจุดต่อ (Property node) คุณสมบัติจุดต่อใช้ในกรณีที่เราต้องการเปลี่ยนคุณสมบัติของวัตถุบนแผงด้านหน้า โดยใช้โปรแกรมในการควบคุม เช่น เราสามารถกำหนดให้ไดโอดเปล่งแสง (Light emitting diode) ทำการกระพริบเพื่อให้ผู้ใช้สังเกตเห็นได้ง่าย ในกรณีที่ต้องการเตือนให้ผู้ใช้งานรับทราบ สามารถสร้างคุณสมบัติจุดต่อได้โดยการไปที่แผนภาพบล็อก แล้วเลือกที่วัตถุที่เราต้องการเปลี่ยนแปลงคุณสมบัติแล้วเลือกที่คุณสมบัติจุดต่อ ถ้าตัวช่วยบริบท (Context help) เปิดอยู่จะแสดงคำอธิบายของแต่ละคุณสมบัติที่จะเลือก

ในกรณีที่เราต้องการเปลี่ยนคุณสมบัติ ให้เลือกที่ตำแหน่งล่างสุดของคุณสมบัติจุดต่อ คุณสมบัติจะถูกเขียนหรืออ่านจากบนลงล่าง ถ้าเราต้องการอ่านค่าคุณสมบัติให้เลือกที่คุณสมบัตินั้นๆ แล้วเลือกเปลี่ยน



รูปที่ ก.8: รูปแบบข้อมูลแบบสายอักขระ

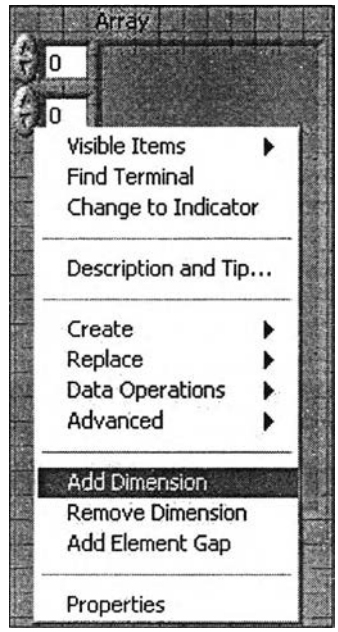
เป็นอ่านค่า (Change to read) แต่ถ้าต้องการเปลี่ยนแปลงค่าคุณสมบัติให้เลือกที่คุณสมบัติ แล้วเลือกเปลี่ยนเป็นเขียน (Change to write)

แถวลำดับคือกลุ่มของข้อมูลในรูปแบบเดียวกัน เช่น กลุ่มของตัวเลข กลุ่มของข้อความ หรือกลุ่มของบูลีน แต่ละข้อมูลในแถวลำดับเรียกว่าสมาชิก (Element) เราสามารถอ้างถึงข้อมูลในแถวลำดับในแต่ละสมาชิกได้โดยการใช้ดรรชนี (Index) สมาชิกแรกสุดในแถวลำดับจะมีดรรชนีเป็น 0 แถวลำดับที่ใช้งานบ่อยๆ จะมี 1 มิติ (1 dimensional array) และ 2 มิติ (2 dimensional array)

ส่วนควบคุมแถวลำดับ (Array control) และส่วนแสดงผลแถวลำดับ (Array indicator) การสร้างส่วนควบคุมแถวลำดับหรือส่วนแสดงผลแถวลำดับ สามารถสร้างได้โดยการเลือกส่วนควบคุม แล้วนำมาวางบนแผงด้านหน้า จากนั้นให้เลือกส่วนควบคุมหรือส่วนแสดงผลที่ต้องการ เช่น ตัวเลขจากส่วนควบคุมแล้วนำมาวางภายในแถวลำดับที่สร้างขึ้นมาก่อนหน้านี้ ถ้าต้องการเพิ่มมิติของแถวลำดับ สามารถทำได้โดยการเลือกที่แถวลำดับที่สร้างขึ้นมาแล้วเลือกที่เพิ่มมิติ (add dimension) ดังแสดงในรูปที่ ก.9

ฟังก์ชันแถวลำดับ (Array functions) ฟังก์ชันต่างๆ เกี่ยวกับแถวลำดับสามารถเลือกได้จากส่วนของฟังก์ชัน รวมถึงการเลือกใช้งานของฟังก์ชันต่างๆ ตามข้อมูลในแถวลำดับ เช่น ในกรณีที่แถวลำดับมีสมาชิกเป็นตัวเลข เราสามารถเรียกเป็นฟังก์ชันของตัวเลข (Numeric functions) ได้ เช่น สามารถเลือกใช้ฟังก์ชันบวกกับแถวลำดับที่มีข้อมูลแบบตัวเลขได้

กลุ่ม (Cluster) กลุ่มหมายถึงกลุ่มของรูปแบบข้อมูลที่แตกต่างกันมารวมอยู่ด้วยกันเป็นรูปแบบข้อมูลใหม่ เช่น เราสามารถสร้างรูปแบบข้อมูลใหม่ของเราเองขึ้นมา ซึ่งรูปแบบข้อมูลของเราประกอบด้วยส่วนควบคุมตัวเลข ส่วนควบคุมบูลีน และส่วนควบคุมสายอักขระ นอกจากการใช้กลุ่มในการสร้างรูปแบบข้อมูลแบบใหม่แล้ว กลุ่มจะใช้เพื่อสร้างเครื่องมือวัดเสมือนย่อยเข้าและเครื่องมือวัดเสมือนย่อยออก เพื่อง่ายต่อความเข้าใจและลดเครื่องปลายทางเข้าและเครื่องปลายทางออกของเครื่องมือวัดเสมือนย่อย โดยเครื่องมือวัดเสมือนย่อยมีขีดจำกัดของเครื่องปลายทางเข้าและเครื่องปลายทางออกได้มากที่สุด 28 เครื่อง



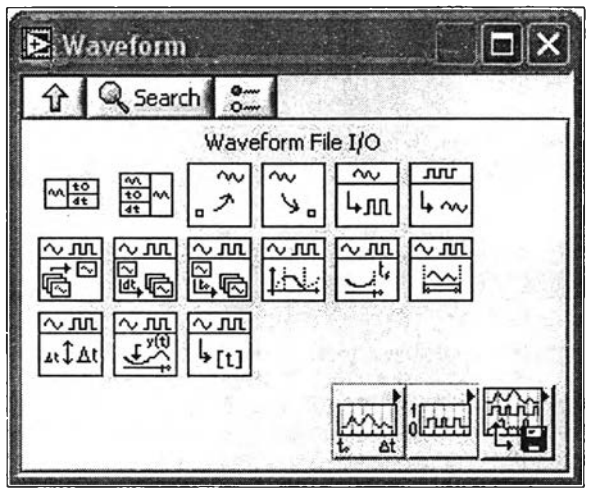
รูปที่ ก.9: การเพิ่มมิติของแถวลำดับ

ปลายทาง

พอลีเมอร์ฟิค (Polymorphic) พอลีเมอร์ฟิคของเครื่องมือวัดเสมือนหมายถึง ฟังก์ชันหรือเครื่องมือวัดเสมือนย่อยที่สามารถรับรูปแบบข้อมูลได้หลายอย่าง เช่น ฟังก์ชันคูณสามารถรับข้อมูลชนิดตัวเลขหรือแถวลำดับของตัวเลข สามารถตรวจสอบได้ว่าเครื่องมือวัดเสมือนย่อยใดเป็นพอลีเมอร์ฟิคได้จากตัวช่วยบริบทของแต่ละฟังก์ชัน แล้วเลือกดูรายละเอียดว่าข้อมูลเข้าของฟังก์ชันนั้นรับรูปแบบข้อมูลเข้าแบบใดบ้าง ฟังก์ชันคูณสามารถเป็นตัวเลขเดียวแถวลำดับของตัวเลข กลุ่มของตัวเลข หรือแถวลำดับของกลุ่มของตัวเลข

รูปแบบข้อมูลแบบรูปคลื่น (Waveform datatype) ฟังก์ชันในการวิเคราะห์ใน LabVIEW นอกจากจะใช้แถวลำดับของข้อมูล เช่น ฟังก์ชันหาค่าเฉลี่ยแล้ว ยังมีรูปแบบข้อมูลอีกชนิดหนึ่งที่ใช้ในฟังก์ชันวิเคราะห์ที่มีเวลาเข้ามาเกี่ยวข้องคือ รูปแบบข้อมูลแบบรูปคลื่น ซึ่งรูปแบบข้อมูลแบบรูปคลื่นจะคล้ายกับกลุ่มของข้อมูล ซึ่งประกอบด้วย แถวลำดับของตัวเลข เวลาเริ่มต้นของแถวลำดับของตัวเลขที่ดรรชนีเป็น 0 และระยะเวลาห่างของเวลาในแต่ละจุด ฟังก์ชันในการใช้งานเกี่ยวกับข้อมูลแบบรูปคลื่นสามารถเลือกได้จากส่วนฟังก์ชันดังแสดงในรูปที่ ก.10

รวมถึงฟังก์ชันประเภทพอลีเมอร์ฟิค เช่น ฟังก์ชันบวกสามารถทำการบวก 2 ข้อมูลแบบรูปคลื่นเข้าด้วยกัน ซึ่งคล้ายกับการบวก 2 ตัวเลขเข้าด้วยกัน นอกเหนือจากฟังก์ชันของข้อมูลแบบรูปคลื่นแล้ว ฟังก์ชันในการวิเคราะห์หลายๆ ฟังก์ชันจากส่วนของฟังก์ชัน เช่น ฟังก์ชันหาค่ารากกำลังสองเฉลี่ย (Root mean square) ของรูปคลื่นและยังมีหลายๆ เซตเครื่องมือ (Toolset) ของ LabVIEW ซึ่งจะมีการวิเคราะห์ในรูปแบบต่างๆ กัน เช่น เซตเครื่องมือสำหรับกระบวนการสัญญาณ (Signal processing toolset) ก็ใช้รูปแบบข้อมูลแบบรูปคลื่น รวมทั้งฟังก์ชันของการได้มาข้อมูล (Data acquisition) ให้ผลลัพธ์ออกมาเป็นรูปแบบข้อมูลแบบรูปคลื่นเช่นกัน



รูปที่ ก.10: ฟังก์ชันรูปแบบข้อมูลแบบรูปคลื่น

แผนภูมิรูปคลื่น (Waveform chart) แผนภูมิรูปคลื่น โดยปกติจะรับข้อมูลประเภทตัวเลขแล้วเก็บข้อมูลไว้ในตัวกัน (Buffer) ซึ่งการแสดงผลอาจจะอยู่ในรูปต่างๆ ขึ้นอยู่กับรูปแบบข้อมูลที่ป้อนเข้าไปยังแผนภูมิรูปคลื่น ดังต่อไปนี้

การแสดงผลของข้อมูลในแผนภูมิรูปคลื่น จะมีอยู่ด้วยกัน 3 รูปแบบคือ แผนภูมิรูปคลื่นแบบแถบ (Strip waveform chart) แผนภูมิรูปคลื่นแบบขอบข่าย (Scope waveform chart) และแผนภูมิรูปคลื่นแบบครอบคลุม (Sweep waveform chart)

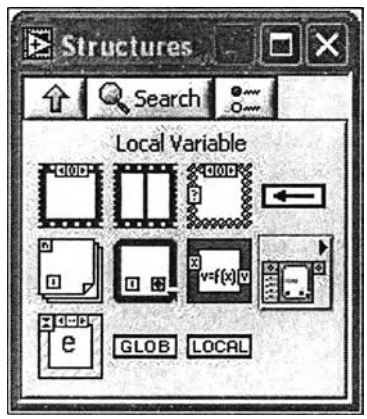
ซึ่งการแสดงผลในรูปแบบต่างๆ ของแผนภูมิรูปคลื่นจะอาศัยข้อมูลที่เก็บไว้ในตัวกันของแผนภูมิรูปคลื่น ซึ่งเราสามารถกำหนดขนาดของตัวกันได้โดยการเลือกที่แผนภูมิรูปคลื่น แล้วก็เลือกที่ความยาวของตัวกัน แต่ถ้าเราต้องการลบข้อมูลในแผนภูมิรูปคลื่นก็สามารถทำได้โดยการเลือกที่แผนภูมิรูปคลื่นแล้วก็ลบแผนภูมิรูปคลื่นทั้งหมดออก

กราฟรูปคลื่น (Waveform graph) ข้อมูลที่ป้อนให้กับกราฟรูปคลื่นจะเป็นข้อมูลประเภทแถวลำดับของตัวเลขหรือประเภทรูปคลื่น เช่น ถ้าแถวลำดับของตัวเลขเป็นชนิด 1 มิติ จะแสดงกราฟ 1 เส้น หรือถ้าเป็นแถวลำดับของตัวเลขเป็นชนิด 2 มิติ จะแสดงกราฟหลายเส้นขึ้นอยู่กับจำนวนแถว (Row) ของแถวลำดับของตัวเลข

ส่วนประกอบของกราฟรูปคลื่น สามารถกำหนดเองได้ โดยให้แสดงหรือซ่อนไว้ได้โดยกำหนดจากคุณสมบัติหรือเลือกจากกราฟรูปคลื่นแล้วเลือกที่รายการทัศนวิสัย (Visible item) สิ่งที่เพิ่มเติมจากแผนภูมิรูปคลื่นคือ คุณลักษณะของตัวชี้ตำแหน่ง (Cursor properties) ซึ่งตัวชี้ตำแหน่งของกราฟรูปคลื่นหมายถึงสิ่งที่แสดงตำแหน่งของกราฟ ซึ่งสามารถมีมากกว่า 1 ตัวได้ รวมทั้งยังสามารถปิดกั้น (Lock) ตัวชี้ตำแหน่งกับเส้นกราฟใดๆ ก็ได้ เราสามารถสร้างตัวชี้ตำแหน่งได้โดยการเลือกรายการทัศนวิสัย

กราฟ X-Y (X-Y graph) LabVIEW จะมีกราฟ X-Y เพื่อให้ง่ายในการใช้งาน แต่หลักการของกราฟ X-Y คือ เราป้อนค่า X และ Y ในรูปของแถวลำดับ กราฟ X-Y จะทำการวาด (Plot) ค่าให้ออกมาเป็นรูป

กราฟรูปแบบอื่นๆ แผนภูมิรูปคลื่น กราฟรูปคลื่น และกราฟ X-Y เป็นรูปแบบการแสดงผลประเภท



รูปที่ ก.11: การเรียกใช้ตัวแปรเฉพาะที่

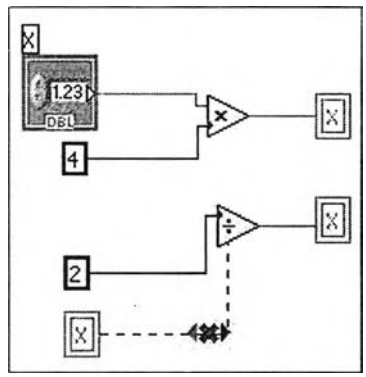
กราฟที่ใช้งานบ่อย ยังมีกราฟรูปแบบอื่นๆ อีก เช่น แผนภูมิความเข้ม (Intensity chart) กราฟความเข้ม (Intensity graph) กราฟรูปคลื่นเชิงเลข (Digital waveform graph) และกราฟ 3 มิติ (3D graph) ซึ่งตัวอย่างต่างๆ สามารถหาได้โดยการพิมพ์ชื่อรูปแบบของกราฟลงในส่วนค้นหาตัวอย่าง (Find examples)

ตัวแปรเฉพาะที่ (Local variable) ในบางครั้งเราอาจต้องการอ่าน เปลี่ยนแปลงค่าของวัตถุในส่วนควบคุมหรือส่วนแสดงผลบนแผงด้านหน้า โดยการเขียนโปรแกรมภายในแผนภาพบล็อกจากหลายๆ ตำแหน่ง ซึ่งสามารถทำได้โดยการใช้ตัวแปรเฉพาะที่ดังแสดงในรูปที่ ก.11 โดยสามารถสร้างตัวแปรเฉพาะที่ได้โดยการเลือกที่ส่วนฟังก์ชัน แล้วเลือกที่ตัวแปรเฉพาะที่ ถ้าต้องการอ่านค่าจากตัวแปรเฉพาะที่ให้เลือกที่เปลี่ยนเป็นอ่าน (Change to read) แต่ถ้าต้องอ่านเขียนค่าลงในตัวแปรเฉพาะที่เพื่อจะเปลี่ยนแปลงค่าของวัตถุบนแผงด้านหน้าให้เลือกที่เปลี่ยนเป็นเขียน (Change to write)

ตัวแปรส่วนกลาง (Global variable) ใช้ในกรณีที่ต้องการส่งผ่านข้อมูล จากเครื่องมือวัดเสมือนหนึ่งไปยังเครื่องมือวัดเสมือนหนึ่ง ที่ทำงานพร้อมๆ กัน การสร้างตัวแปรส่วนกลางใหม่ สามารถสร้างได้โดยเลือกที่ส่วนของฟังก์ชันแล้วเลือกที่ตัวแปรส่วนกลาง

หลังจากนั้นเราสามารถสร้างส่วนควบคุมหรือส่วนแสดงผลได้บนเครื่องมือวัดเสมือนเดียวกัน เช่นเดียวกับการสร้างส่วนควบคุมหรือส่วนแสดงผลบนเครื่องมือวัดเสมือนปกติ แล้วบันทึกชื่อได้เหมือนเครื่องมือวัดเสมือนปกติทุกอย่าง การเลือกตัวแปรส่วนกลางจากเครื่องมือวัดเสมือนอื่นๆ หรือเรียกใช้จากที่เคยสร้างขึ้นมาก่อน ทำได้โดยการเลือกส่วนของฟังก์ชัน แล้วเลือกที่ตัวแปรส่วนกลาง ซึ่งมีนามสกุลเป็นเครื่องมือวัดเสมือน นำมาวางบนแผนภาพบล็อก การอ่านและเขียนข้อมูลลงในตัวแปรส่วนกลางทำเช่นเดียวกับตัวแปรเฉพาะที่ คือเลือกที่ตัวแปรส่วนกลางแล้วเลือกที่ตัวแปรที่สร้างขึ้นโดยเลือกตามต้องการแล้วเลือกที่เปลี่ยนเป็นอ่านหรือว่าเปลี่ยนเป็นเขียน เพื่ออ่านหรือเขียนตามลำดับ

เงื่อนไขเรซ (Race condition) สิ่งที่ควรระวังในการใช้ตัวแปรเฉพาะที่หรือตัวแปรส่วนกลางคือเงื่อนไขเรซ ซึ่งเกิดขึ้นในกรณีที่โปรแกรมทำงานแบบขนาน (Parallel) และพยายามแก้ไขข้อมูลเดียวกัน ซึ่งจะไม่เกิดขึ้นกับการเขียนโปรแกรมโดยใช้ข้อความ เช่นการเขียนโปรแกรมโดยใช้ข้อความอาจจะเขียนโปรแกรม



รูปที่ ก.12: การเขียนโปรแกรม LabVIEW ทำให้ผลลัพธ์ของตัวแปร X เกิดเงื่อนไขเรซ

ดังนี้

$$X = X \times 4 \quad \text{(ก.1)}$$

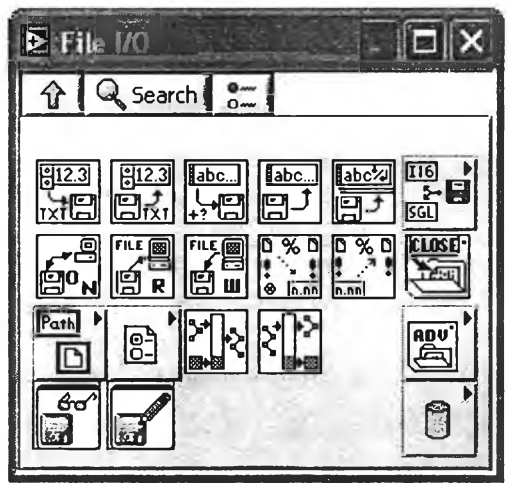
$$X = X \div 2 \quad \text{(ก.2)}$$

ถ้าตัวแปร X เท่ากับ 2 ตอนเริ่มต้นโปรแกรม ดังนั้นตัวแปร X จะเท่ากับ 8 ในสมการแรก และตัวแปร X จะเท่ากับ 4 เมื่อทำงานในสมการที่ 2 เสร็จแล้ว แต่ถ้าเขียนโปรแกรม LabVIEW ดังแสดงในรูปที่ ก.12 เนื่องจากทำงานแบบกระแสข้อมูล เราไม่สามารถรู้ได้ว่าสมการไหนจะแก้ไขตัวแปร X ก่อนหรือหลัง เช่น ถ้าตัวแปร X ก่อนถึงสมการที่เท่ากับ 2 และถ้าสมการบนทำการแก้ไขตัวแปรเฉพาะที่ของ X ก่อนอาจจะเท่ากับ 8 ชั่วคราว แล้วถูกแก้ไขโดยสมการที่ 2 คือเท่ากับ 1 ซึ่งจะทำให้ผลลัพธ์ของตัวแปร X ผิดพลาดได้ แต่ถ้าสมการที่ 2 แก้ไขตัวแปรเฉพาะที่ของ X ก่อนตัวแปร X อาจเท่ากับ 1 ชั่วคราว แล้วจะโดนแก้ไขจากสมการบน ทำให้ผลลัพธ์ของตัวแปร X จะเท่ากับ 8

เราสามารถหลีกเลี่ยงเงื่อนไขเรซ โดยการละเว้นการเขียนหรือแก้ไขตัวแปรเฉพาะที่ หรือตัวแปรส่วนกลาง หลังจากที่อ่านขึ้นมาหรืออาจใช้จุดต่อสูตรแทนก็ได้ หรือว่าจะใช้โครงสร้างลำดับกำหนดการไหลของโปรแกรม

แฟ้มเข้าและแฟ้มออก (File input and file output) ฟังก์ชันแฟ้มเข้าและแฟ้มออกใช้สำหรับการเขียนข้อมูลลงในแฟ้ม อ่านข้อมูลจากแฟ้มดังแสดงในรูปที่ ก.13 ซึ่ง LabVIEW จะแบ่งกลุ่มของฟังก์ชันแฟ้มเข้าและแฟ้มออกเป็น 3 ระดับคือ

1. ระดับสูง (High level) ซึ่งจะใช้งานง่ายแต่ไม่ยืดหยุ่น หรืออ่านแฟ้มได้ไม่รวดเร็วเมื่อเปรียบเทียบกับระดับต่ำกว่า
2. ระดับต่ำ (Low level) ซึ่งส่วนใหญ่ของการใช้งานแฟ้มเข้าและแฟ้มออกสามารถใช้ฟังก์ชันระดับต่ำได้เพราะทำงานได้เร็วและยืดหยุ่นสูง
3. เครื่องมือวัดเสมือนเร่งด่วน (Express virtual instrument) ซึ่งเป็นโครงแบบ (Configuration) การเขียนหรืออ่านแฟ้ม ใช้งานง่ายและสามารถเปลี่ยนเป็นรหัสสำหรับการเปลี่ยนแปลงแก้ไข เพื่อให้มีประสิทธิภาพดียิ่งขึ้น



รูปที่ ก.13: ฟังก์ชันแฟ้มเข้าและแฟ้มออก

หลักการของการอ่านหรือเขียนข้อมูลลงแฟ้มมี 4 ขั้นตอนใหญ่ๆ คือ การเปิดแฟ้ม แล้วทำการอ่านข้อมูลหรือเขียนข้อมูลลงแฟ้ม เมื่อเสร็จสิ้นการอ่านหรือเขียนลงแฟ้มแล้ว จึงทำการปิดแฟ้ม ถ้ามีความผิดพลาดขึ้นในขั้นตอนใดให้แสดงผลข้อผิดพลาด ซึ่งในฟังก์ชันแฟ้มระดับสูงจะรวมเอาทุกขั้นตอนตั้งแต่เปิดแฟ้มอ่านหรือเขียน ปิดแฟ้ม และแสดงข้อผิดพลาดในสัญรูปเดียว แต่ถ้าเราทำการวนซ้ำจะมีความซับซ้อนในการเปิดและปิดแฟ้ม ทำให้โปรแกรมจะทำงานได้ช้า ซึ่งไม่เร็วเท่ากับการใช้ฟังก์ชันแฟ้มระดับต่ำที่สามารถเขียนโปรแกรมให้เปิดและปิดเพียงครั้งเดียว แต่แบบฟังก์ชันแฟ้มระดับสูงช่วยให้เราสามารถทดสอบการทำงานของโปรแกรมได้เร็ว แล้วจึงค่อยไปใช้ในฟังก์ชันแฟ้มระดับต่ำในภายหลัง

รูปแบบบันทึกข้อมูลลงในแฟ้มแบ่งออกเป็น 2 รูปแบบใหญ่ๆ คือ การอ่านหรือการเขียนข้อมูลแบบแฟ้มตัวอักษร (Text file) และการอ่านหรือการเขียนข้อมูลแบบแฟ้มฐานสอง (Binary file) ซึ่งจะแตกต่างกันตรงที่การอ่านหรือการเขียนข้อมูลแบบแฟ้มตัวอักษรจะใช้งานง่ายกว่า เพราะข้อมูลที่เขียนหรืออ่านแฟ้มสามารถใช้โปรแกรมที่อ่านข้อมูลแบบแฟ้มตัวอักษร ทำการตรวจสอบความถูกต้องได้ เช่น โปรแกรมสมุดพก (Notepad) แต่แฟ้มจะใหญ่กว่าแบบแฟ้มฐานสอง และการเขียนลงในแฟ้มจะช้ากว่าแบบแฟ้มฐานสอง

โครงสร้างเหตุการณ์ (Event structure) โครงสร้างเหตุการณ์ของ LabVIEW ช่วยให้การพัฒนาโปรแกรมที่มีการติดต่อกับผู้ใช้งานที่ซับซ้อนให้สามารถทำได้ง่ายขึ้น เช่น สามารถตรวจสอบได้เมื่อมีการกดปุ่มแล้วจึงไปทำงานตามที่ปุ่มนั้นกำหนดไว้ ซึ่งโครงสร้างเหตุการณ์นี้เริ่มมีตั้งแต่ LabVIEW 6.1 โดยปกติแล้วโครงสร้างเหตุการณ์จะใช้ร่วมกับการเขียนโปรแกรมแบบวงวน

ก.4 สถาปัตยกรรมการเขียนโปรแกรม

หลักการหนึ่งในการพัฒนาโปรแกรมประยุกต์โดยใช้ LabVIEW คือโปรแกรมส่วนของรหัสต้นฉบับในแผนภาพบล็อกควรอยู่ในหนึ่งหน้าจอของคอมพิวเตอร์ ซึ่งจะทำให้โปรแกรมอ่านต่อความเข้าใจและง่ายต่อ

การปรับปรุงแก้ไข ซึ่งสามารถทำได้โดยการอาศัยการออกแบบโครงสร้างของโปรแกรมและใช้เครื่องมือวัดเสมือนย่อยเข้าช่วย ดังนั้นในบทนี้จะกล่าวถึงโครงสร้างรูปแบบต่างๆ ซึ่งสามารถนำมาใช้ได้ รูปแบบของโครงสร้างการเขียนโปรแกรมที่ใช้บ่อย มีดังต่อไปนี้คือ

1. เครื่องมือวัดเสมือนแบบง่าย (Simple virtual instrument) ใช้เมื่อเราต้องการเขียนฟังก์ชันหรือเครื่องมือวัดเสมือนย่อย เพื่อที่จะเรียกใช้ในเครื่องมือวัดเสมือนอื่นต่อไป เช่นโปรแกรม CtoF.vi ซึ่งเราพัฒนาในโครงการ (Project) แรกๆ

2. วงวนเดียว (Single loop) คือ โปรแกรมที่มีการเขียนโปรแกรมวงวนแบบอันเดียวเป็นหลัก เช่น โปรแกรมเครื่องมือวัดเสมือนการแสดงค่าเฉลี่ยอุณหภูมิ (Monitor average temperature with logging.vi) ที่ได้ทำขึ้นก่อนหน้านี้ หรือโปรแกรมเครื่องมือวัดเสมือนการจำลองระบบถัง (Tank simulation.vi)

3. วงวนขนาน (Parallel while loop) คือโปรแกรมที่มีการเขียนโปรแกรมวงวนอย่างน้อย 2 วงวนทำงานพร้อมๆ กัน เช่น โปรแกรมเครื่องมือวัดเสมือนการควบคุมกระบวนการเครื่องผสม (Control mixer process.vi)

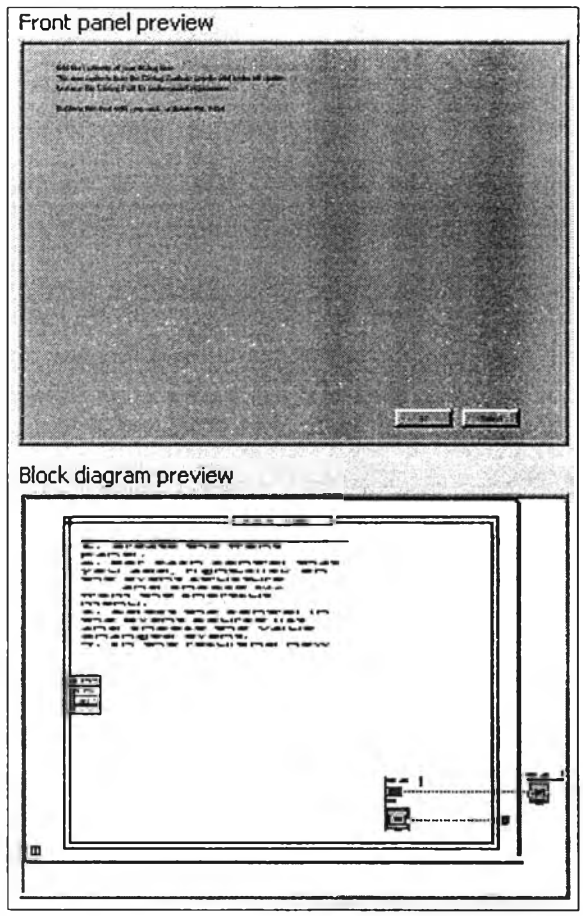
4. หลายกรณี (Multiple case) เป็นรูปแบบที่มีหลายกรณีในการเขียนโปรแกรมแบบวงวน เช่น โปรแกรมเครื่องมือวัดเสมือนออสซิลโลสโคปแบบ 2 ช่อง (Two channel oscilloscope.vi)

5. สถานะเครื่องจักร (State machine) เป็นรูปแบบที่ใช้การเขียนโปรแกรมวงวน เรจิสเตอร์แบบเลื่อน และโครงสร้างกรณี ประกอบกันเป็นรูปแบบที่นิยมกันมากรูปแบบหนึ่ง เช่นโปรแกรมเครื่องมือวัดเสมือนทดสอบลำดับสถานะเครื่องจักร (State machine test sequence.vi)

6. การจัดกระทำเหตุการณ์เพื่อติดต่อกับผู้ใช้งาน (User interface event handler) เป็นการใช้โครงสร้างเหตุการณ์ ซึ่งสามารถนำมาประยุกต์ใช้ร่วมกับรูปแบบต่างๆ ที่กล่าวมา

แผ่นแบบเครื่องมือวัดเสมือน (Virtual instrument templates) โครงสร้างต่างๆ ของ LabVIEW ซึ่งรวมถึงสถาปัตยกรรมที่กล่าวถึงไปแล้วได้จัดทำเป็นแผ่นแบบดังแสดงในรูปที่ ก.14 เพื่อให้ง่ายต่อการเริ่มต้นเขียนโปรแกรมใน LabVIEW 7 Express จะให้เปิดให้เลือกแผ่นแบบ ตั้งแต่การเริ่มสร้างเครื่องมือวัดเสมือนใหม่ใน LabVIEW เวอร์ชัน (Version) ก่อนหน้านี้รวมถึง LabVIEW 7 Express สามารถหาแผ่นแบบได้จากการเลือกที่แผ่นแบบ เราสามารถบันทึกรูปแบบโครงสร้างของเครื่องมือวัดเสมือนที่เราใช้บ่อยๆ เป็นแผ่นแบบของเราเองได้โดยการเลือกบันทึกข้อมูลเป็นแผ่นแบบซึ่งเมื่อบันทึกลงในสารบบ (Directory) โปรแกรม LabVIEW ก็จะแสดงแผ่นแบบขึ้นมาให้เราเลือกใช้

คุณลักษณะเครื่องมือวัดเสมือน (Virtual instrument properties) คุณลักษณะเครื่องมือวัดเสมือน คือ คุณลักษณะต่างๆ ของโปรแกรมที่เราพัฒนาขึ้น เราสามารถกำหนดลักษณะการทำงานหรือการแสดงผลได้ การเรียกคุณลักษณะเครื่องมือวัดเสมือนทำได้ 3 วิธี คือเลือกที่สัญรูปปุ่มบนขวาของบนแผงด้านหน้า หรือแผนภาพบล็อก หรือเลือกที่คุณลักษณะเครื่องมือวัดเสมือน (VI properties) หรือกดปุ่ม Ctl+I คุณลักษณะเครื่องมือวัดเสมือนแบ่งออกเป็นประเภท (Category) เช่น รูปแบบทั่วไปจะแสดงชื่อและตำแหน่งของเครื่อง



รูปที่ ก.14: ตัวอย่างแผ่นแบบใน LabVIEW

มือวัดเสมือนนี้และจำนวนครั้งที่ได้แก้ไขเราสามารถเลือกดู คุณลักษณะต่างๆ โดยเลือกที่ตัวเลือกประเภท (Category selector) คุณลักษณะเครื่องมือวัดเสมือนที่สำคัญคือ

1. หน้าต่างสภาพปรากฏ (Window appearance) ซึ่งทำให้เราสามารถกำหนดการแสดงหน้าต่างของแผงด้านหน้าในรูปแบบต่างๆ เช่น ให้แสดงในรูปแบบแสดงค่าคือไม่มีเมนูต่างๆ ของ LabVIEW ปรากฏ หรือเราสามารถเลือกในส่วนที่เราอยากแสดงหรือไม่แสดงได้โดยการเลือกเครื่องมือลูกค้า (Customize) เช่น ถ้าโปรแกรมของเราจะถูกเปิดโดยการเลือกปุ่มจากอีกเครื่องมือวัดเสมือน เราสามารถเลือกแสดงผลบนแผงด้านหน้าเมื่อเรียก (Show front panel when called) เพราะโดยปริยาย (Default) ของเครื่องมือวัดเสมือนจะทำงานเฉพาะในส่วนของแผนภาพบล็อก เมื่อถูกใช้โดยอีกเครื่องมือวัดเสมือนหนึ่ง วิธีนี้จะเป็นการกำหนดว่าเครื่องมือวัดเสมือนนี้ถ้าถูกเรียกใช้โดยเครื่องมือวัดเสมือนอื่นจะแสดง แผงด้านหน้าทุกครั้งเป็นโดยปริยาย ถ้าในบางครั้ง หรือเฉพาะครั้งที่ต้องการเปิดแผงด้านหน้าจากเครื่องมือวัดเสมือนอื่น เราสามารถเลือกที่เครื่องมือวัดเสมือนย่อยนี้ในเครื่องมือวัดเสมือนอื่นแล้วเลือก การจัดวางจุดต่อเครื่องมือวัดเสมือนย่อย (SubVI node setup)

2. กระทำการ (Execution) คือ คุณลักษณะเมื่อเครื่องมือวัดเสมือนทำงาน เช่น เราสามารถกำหนดว่าให้ทำงานทันทีที่ถูกเปิด ซึ่งถ้าเราเลือกเปิดจาก LabVIEW แล้วเมื่อเปิดขึ้นจะทำงานในทันที หรือ

กำหนดลำดับความสำคัญ (Priority) หรือลำดับความสำคัญของการทำงานของเครื่องมือวัดเสมือนนี้ เช่น ถ้าเลือกเป็นลำดับความสำคัญสูงสุด (Highest piority) เครื่องมือวัดเสมือนย่อยนี้จะใช้หน่วยประมวลผลกลาง (Central processing unit) โดยที่ลำดับความสำคัญที่ต่ำกว่าต้องรอหรือจะให้มีการตรวจสอบความผิดพลาดให้โดยอัตโนมัติ โดยเลือกสามารถตรวจสอบความผิดพลาดโดยอัตโนมัติ (Enable automatic error handling)

3. หน่วยความจำทางการใช้งาน (Memory usage) จะบ่งบอกถึงขนาดของหน่วยความจำต่างๆ ของโปรแกรมที่พัฒนาขึ้น

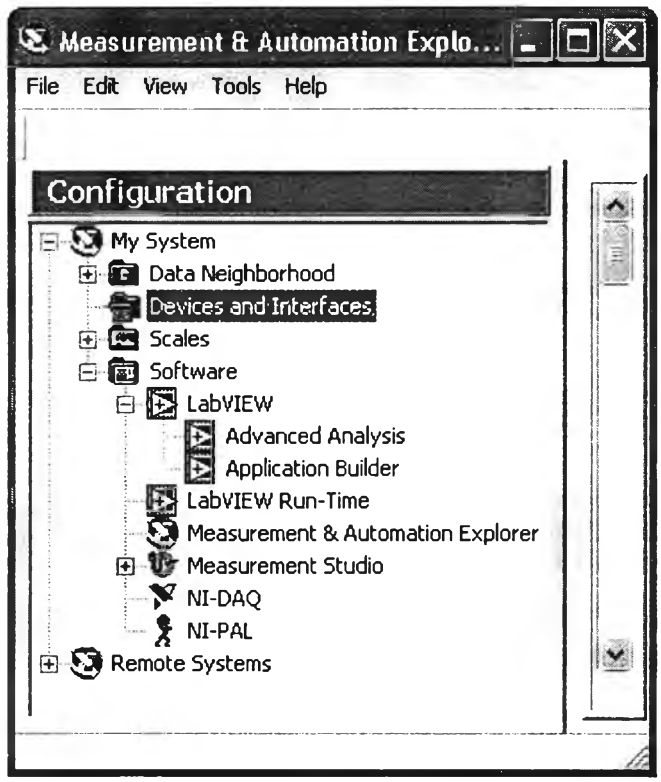
4. การจัดทำเอกสาร (Documentation) เป็นส่วนที่เราสามารถใส่คำอธิบายเกี่ยวกับเครื่องมือวัดเสมือนที่เราพัฒนาขึ้น ซึ่งจะติดไปกับเครื่องมือวัดเสมือน

5. การแก้ไขฮิสโตรแกรม (Revision histrogram) เราสามารถป้อนสิ่งที่เราได้ทำการเปลี่ยนแปลงเครื่องมือวัดเสมือน เพื่อการอ้างอิงถึงในอนาคต

รูปแบบการบันทึก (Save with option) LabVIEW มีรูปแบบการบันทึกหลายแบบ โดยปกติเราจะบันทึกแยกเป็นแฟ้มไป เมื่ออยู่ในขั้นตอนการพัฒนา LabVIEW ยังมีรูปแบบการบันทึกอีกหลายรูปแบบ โดยการเลือกรูปแบบการบันทึก เช่น ในกรณีที่เราต้องการบันทึกเครื่องมือวัดเสมือนทั้งหมดเพื่อส่งต่อให้ผู้อื่น เราอาจต้องบันทึกเครื่องมือวัดเสมือนรวมถึงเครื่องมือวัดเสมือนย่อยที่เราสร้างขึ้น เพราะผู้อื่นอาจไม่มีเครื่องมือวัดเสมือนย่อยเราอยู่ เราสามารถเลือกรูปแบบการบันทึกแบบการแจกตัวการประยุกต์ (Application distribution) ซึ่งจะบันทึกโปรแกรมเครื่องมือวัดเสมือนหลัก (Main virtual instrument) รวมทั้งเครื่องมือวัดเสมือนย่อยต่างๆ ในรูปแบบห้องสมุด (Library) หรือ .llb ทำให้ประหยัดเนื้อที่ที่จะจัดเก็บ เพราะเครื่องมือวัดเสมือนจะถูกบันทึกแบบบีบอัด แต่การแก้ไขโดยตรงจาก .llb แล้วบันทึกจะเสียเวลานานขึ้น เมื่อเรียกโปรแกรม .llb LabVIEW จะแสดงแฟ้มทั้งหมดใน .llb โดยแยกระหว่างเครื่องมือวัดเสมือนหลักกับเครื่องมือวัดเสมือนหลักย่อยออกจากกันโดยใช้เส้นประ เราสามารถย้ายเครื่องมือวัดเสมือนย่อยมาอยู่บริเวณเครื่องมือวัดเสมือนหลักได้ โดยเลือกที่เมนูเครื่องมือแล้วเลือกเครื่องมือวัดเสมือนย่อยที่เราต้องการแล้วเลือกระดับสูง (Top level)

คีย์บอร์ด (Keyboard) การใช้คีย์บอร์ดกับโปรแกรมเราสามารถกำหนดการเคลื่อนที่ระหว่างส่วนควบคุม หรือส่วนแสดงผลต่างๆ โดยใช้แท็ป (Tab) หรือชิฟต์แท็ป (Shift +Tab) เพื่อกำหนดลำดับการควบคุมโดยเลือกที่เมนูคำสั่งตั้งแท็ป (Set tabbing order) หลังจากนั้น เราสามารถป้อนตัวเลขในช่องแล้วใช้ตัวชี้ตำแหน่งควบคุม (Cursor click controls) หรือส่วนแสดงผลที่เราต้องการเปลี่ยน แล้วบันทึกหรือยกเลิกการจัดลำดับ เราสามารถกำหนดคีย์ฟังก์ชัน (Function key) บนคีย์บอร์ด เพื่อใช้แทนการคลิก (Click) ที่ส่วนควบคุมต่างๆ เช่น ปุ่มหยุดเราอยากกำหนดว่าปุ่ม F4 แล้ว ปุ่มหยุดจะถูกเลือกเพื่อหยุดการทำงาน ซึ่งสามารถกำหนดโดยการเลือกที่ปุ่มหยุดแล้วเลือกคีย์เดินทาง (Key navigation) แล้วเลือก F4 เพื่อกำหนดปุ่มบนคีย์บอร์ดที่สอดคล้องกับปุ่มหยุด

ปรับแต่งการใช้งานของ LabVIEW เราสามารถปรับรูปแบบของ LabVIEW ตามความถนัดในการใช้งานของเราได้ เช่น เราสามารถปรับรูปแบบแผงด้านหน้า หรือแผนภาพบล็อกในรูปแบบที่เราต้องการ



รูปที่ ก.15: รูปแบบการใช้งาน NI-DAQmx

ระหว่างการพัฒนาโปรแกรม โดยเลือกรูปแบบได้จากแผงด้านหน้าและแผนภาพบล็อก เช่นที่แผนภาพบล็อกเราสามารถกำหนดโยงสายอัตโนมัติ (Auto wiring) รวมทั้งระยะในการทำโยงสายอัตโนมัติได้ โดยเลือกที่สามารถโยงสายอัตโนมัติ (Enable auto wiring) หรือกำหนดจำนวนครั้งที่สามารถไม่กระทำ (Undo) ได้ หรือแผงด้านหน้าเราสามารถกำหนดความเร็วในการกระพริบของส่วนควบคุมหรือส่วนแสดงผลได้

ก.5 การรับข้อมูลกับ LabVIEW

ใน LabVIEW 7.0 มีการเริ่มใช้ NI-DAQmx ดังแสดงในรูปที่ ก.15 ซึ่งเป็นชุดขับ (Diver) ยุคใหม่ของ NI-DAQ ซึ่ง NI-DAQmx จะช่วยให้ใช้งานการรับข้อมูล (วัดสัญญาณหรือสร้างสัญญาณทางฟิสิกส์ เช่น สัญญาณไฟฟ้าโดยใช้คอมพิวเตอร์) ได้ง่ายและรวดเร็วยิ่งขึ้นและมีโปรแกรมหนึ่งเรียกว่า สำรวจเครื่องมือวัดและอัตโนมัติ (Measurement and automation explorer) หรือย่อๆ ว่า MAX สำหรับตรวจสอบ โครงแบบและทดสอบสัญญาณที่ต่อเข้ากับช่องต่างๆ ของการ์ด DAQ รวมถึงทำการเทียบมาตรฐานตัวเอง (Self calibration) หรือข้อมูลการเทียบมาตรฐานครั้งล่าสุด

ภายในข้อมูลเครื่องมือและการติดต่อการใช้งานจะแสดง DAQ การ์ดต่างๆ ที่ได้ติดตั้งลงในคอมพิวเตอร์ ในการติดตั้งครั้งแรกถ้าการ์ดไม่แสดงลงขึ้นมาให้กด F5 หรือเลือกที่เมน ถ้าทุกอย่างปกติ MAX จะแสดงการ์ดที่ติดตั้งขึ้นมาที่ข้อมูลเครื่องมือและการติดต่อการใช้งาน (Devices and interfaces) MAX จะช่วยในการทดสอบสัญญาณที่ต่อเข้ากับ DAQ การ์ด โดยเลือกที่ทดสอบแผง MAX (Test panel

MAX) เราสามารถเริ่มใช้งานการรับข้อมูลกับโปรแกรม LabVIEW ได้โดยการเลือกเครื่องมือวัดเสมือนการรับข้อมูลกับNI-DAQmx (Data acquisition with NI-DAQmx.vi) หรือเปิดดูตัวอย่างซึ่งมีอยู่มากมาย

ก.6 การควบคุมเครื่องมือด้วย LabVIEW

ใช้ในการควบคุมเครื่องมือวัด เช่น ออสซิลโลสโคปหรือหลายมาตรา โดยใช้คอมพิวเตอร์โดยส่วนใหญ่ๆ ในทางอุตสาหกรรมจะนิยมใช้การ์ด GPIB ซึ่งการ์ด GPIB หนึ่งการ์ดสามารถควบคุมเครื่องมือวัดได้ถึง 15 เครื่องมือ โดยอ้างถึงแต่ละเครื่องมือวัดโดยใช้ที่อยู่ (Address) ถ้าท่านติดตั้งการ์ด GPIB ลงในคอมพิวเตอร์ การ์ด GPIB ก็จะแสดงอยู่ในข้อมูลเครื่องมือและการติดต่อการใช้งาน ใน MAX เช่นเดียวกับการ์ด DAQ

ในการควบคุมเครื่องมือวัดสิ่งที่ต้องรู้คือ ที่อยู่ของเครื่องมือวัดเพราะเครื่องมือวัดที่ต่อร่วมกับการ์ด GPIB อันเดียวกันต้องมีที่อยู่ต่างกัน (การตั้งค่าที่อยู่ของแต่ละเครื่องมือให้ดูคู่มือวัดของแต่ละเครื่องมือวัดที่ท่านมีอยู่) การตรวจสอบว่าเครื่องมือไหนมีที่อยู่เท่าใด สามารถทำได้โดยการเลือกที่การ์ด GPIB ที่ต่อกับเครื่องมือวัดนั้นแล้วเลือกที่กราดตรวจสำหรับเครื่องมือ (Scan for instruments) ซึ่งถ้าเครื่องมือวัดต่ออยู่ก็จะขึ้นชื่อของเครื่องวัดนั้นที่รายละเอียดประกอบ (Description) พร้อมทั้งแสดงที่อยู่ที่ช่องค่า (Value) แต่ถ้าไม่มีเครื่องมือต่ออยู่หรือไม่สามารถติดต่อกับเครื่องมือวัดได้แสดงไม่พบเครื่องมือ (Instruments not found) ที่ช่องชื่อ (Name)

หลังจากนั้นท่านสามารถตรวจสอบคำสั่งต่างๆ ที่จะส่งไปควบคุมที่เครื่องมือวัดได้โดยการเลือกที่เครื่องมือวัดนั้นแล้วเลือกช่องสื่อสารกับเครื่องมือ (Communicate with instrument) เช่นเครื่องมือวัดส่วนใหญ่จะรู้จักคำสั่ง "*IDN?" ซึ่งเป็นคำสั่งให้เครื่องมือวัดเตรียมตอบว่าเครื่องมือวัดนั้นเป็นชนิดใดของบริษัทอะไรเป็นผู้ผลิต เราสามารถส่งคำสั่งได้โดยการพิมพ์ "*IDN?" ลงที่ช่องส่งสายอักขระ (Send string) แล้วเลือกปุ่มเขียน และกดปุ่มอ่านเพื่ออ่านค่าที่เครื่องมือวัดเตรียมส่งค่าให้กับคอมพิวเตอร์ ในการใช้คำสั่งอื่นๆ ก็สามารถทำได้เช่นเดียวกัน บางคำสั่งอาจใช้คำสั่งส่งอย่างเดียว เช่น คำสั่งที่จะให้เครื่องมือวัดตั้งค่าใหม่ (Reset) หรือปรับค่าต่างๆ

เราสามารถเริ่มต้นเขียนโปรแกรมสำหรับการควบคุมเครื่องมือวัดโดยการใช้ ผู้ช่วยเครื่องมือเข้าและออก (Instrument I/O assistant) โดยเลือกที่ผู้ช่วยเครื่องมือเข้าและออก (GPIB) แล้วเลือกอ่านและแสดงผลเพื่อเลือกแผ่นแบบ (Template) นี้ที่แผนภาพบล็อกจะเห็นข้อแนะนำพร้อมทั้งชื่อผู้ช่วยเครื่องมือเข้าและออก และแฟ้มการเขียน LabVIEW เครื่องมือวัด (Write LabVIEW measurement file) เราสามารถเลือกที่สัญรูปของผู้ช่วยเครื่องมือเข้าและออกเพื่อทำการควบคุมเครื่องมือวัด ซึ่งจะเปิดหน้าต่างให้เราป้อนคำสั่งโครงแบบให้เลือกเครื่องมือวัดที่เราจะควบคุมจากการเลือกเครื่องมือ (Select an instrument) แล้วป้อนค่าหมดเวลารอ (Timeout) เพื่อเผื่อเวลาให้พอเพียงกับการติดต่อกับเครื่องวัด ซึ่งจะขึ้นอยู่กับคำสั่ง และแต่ละเครื่องมือวัด หลังจากนั้นให้เลือกเพิ่มขั้น (Add step) เพื่อที่จะสั่งคำสั่งไปยังเครื่องมือวัด ซึ่งมีขั้นอยู่ 4 รูปแบบ คือ

1. เลือกเครื่องมือ (Select instrument) คือเลือกเครื่องมือที่เราจะควบคุม

2. สอบถามและแจงส่วน (Query and parse) คือการส่งคำสั่งไปยังเครื่องมือวัดและอ่านข้อมูลกลับจากเครื่องมือวัดแล้วแปลงข้อมูล

3. เขียน (Write) คือการส่งคำสั่งไปยังเครื่องมือวัด

4. อ่านและแจงส่วน (Read and parse) คือการอ่านข้อมูลจากเครื่องมือวัดแล้วแปลงข้อมูล

ในที่นี้จะใช้คำสั่งที่แทบทุกเครื่องมือวัดรู้จักคือ "*IDN?" เพื่อที่จะอ่านชื่อของเครื่องมือวัดที่เราเลือกไว้ขึ้นมาแสดงผลโดยการเลือกเมนูเขียนแล้วป้อนคำสั่งลงในช่องคำสั่งป้อนเข้า (Enter a command) แล้วเลือกปุ่มวิ่งขั้นนี้ (Run this step) เพื่อที่จะส่งคำสั่งไปยังเครื่องมือวัดที่เลือกไว้ หลังจากนั้นให้เลือกปุ่มเพิ่มขั้น แล้วเลือกเมนูอ่านและแจงส่วนเพื่อที่จะอ่านข้อมูลจากคำสั่ง "*IDN?" ที่เราส่งไป เลือกปุ่มวิ่งหรือวิ่งขั้นนี้ จะเห็นข้อความที่เครื่องมือวัดตอบกลับมาถ้าต้องการลบหรือแทรกคำสั่งให้เลือกที่คำสั่งนั้นแล้วเลือกคำสั่งตามต้องการ หลังจากนั้นเราสามารถเพิ่มคำสั่งต่อๆ ไปได้โดยเลือกที่เพิ่มขั้น เนื่องจากข้อมูลที่ส่งจากเครื่องมือวัดจะเป็นรูปแบบตัวอักษรมาตรฐานรหัสอะเมริกันสำหรับข้อมูลข่าวสารแลกเปลี่ยนกัน (American Standard Code for Information Interchange (ASCII)) และอาจมีหลายข้อมูลเรื่องต่อๆ กัน โดยใช้เครื่องหมายที่ผู้ผลิตเครื่องมือวัดกำหนด เราสามารถใส่เครื่องหมายเพื่อที่จะแปลงค่าจาก เป็นค่าที่เหมาะสมโดยเลือกว่าจะแปลงค่าเป็นแบบอักษรหรือตัวเลขในการส่งค่า โดยเลือกที่ชนิดข้อมูล (Data type) และจำนวนตัวอักษรที่จะออก ขึ้นอยู่กับจำนวนที่เราป้อนลงในนับอักขระ (Character count) พร้อมตั้งชื่อในช่องตั้งชื่อ (Token name) เมื่อเสร็จสิ้นการป้อนคำสั่งแล้วให้ปุ่มตกลง จะได้สัญรูปที่ให้ที่เราตั้งชื่อ โดยใช้เครื่องมือโยงสายแล้วเลือกที่เครื่องปลายทาง จะได้ ตัวแสดงผลตัวอักษร (Text indicator) แล้วทดลองวิ่งโปรแกรม

ก.7 การใช้งานอื่นๆ

LabVIEW ยังมีคุณลักษณะการใช้งาน (Feature) อีกหลายประการ ซึ่งมาถึงจุดนี้แล้วท่านสามารถศึกษาได้ด้วยตัวเองจากตัวอย่างของ LabVIEW รวมถึงสามารถหาตัวอย่างได้เพิ่มเติมจากตัวช่วยในโปแกรม ต่อไปนี้เป็นหัวข้อที่น่าสนใจซึ่งสามารถหาข้อมูลเพิ่มเติมได้คือ ฟังก์ชันอ้างอิง (Function reference) ซึ่งจะรวมฟังก์ชันต่างๆ ที่ LabVIEW แต่ละรุ่น (Package) มีอยู่ ซึ่งสามารถอ้างอิงฟังก์ชันต่างๆ ได้โดยเลือกที่เมนูตัวช่วย

รูปแบบการติดต่อการใช้งานอื่นๆ ซึ่งหาดูตัวอย่างได้อย่างใน LabVIEW

1. แสดงรายการและตาราง (List and table)

2. วงแหวนและแจงนับ (Ring and enumerate)

3. ตู้บรรจุสินค้า (Containers) ซึ่งประกอบด้วย แถบควบคุม (Control tab) แผงย่อย (Sub panel) และตู้บรรจุสินค้าแอ็กทิฟ (Active containers)

4. เครื่องตกแต่ง (Decoration) สำหรับตกแต่งแผงด้านหน้าให้สวยงามขึ้น

ตัวอย่างเกี่ยวกับเมนูหาได้จาก Menubars.llb โดยสามารถสร้างเมนูได้จากการเลือกเมนูเฉพาะใช้งาน (Runtime menu) และจากแผงด้านหน้าหรือแผนภาพบล็อก รุ่นรายงาน (Report generation) สำหรับสร้างรายงาน โดยหาตัวอย่างได้จากรายงาน (Reports) หรือสามารถใช้การค้นหาตัวอย่าง (Search example)

การควบคุมประยุกต์ (Application control) เช่น ฟังก์ชันออกจาก LabVIEW (Quit LabVIEW) ใช้ในกรณีที่เขียนโปรแกรมเสร็จแล้ว และต้องการให้ปิด LabVIEW เมื่อโปรแกรมเสร็จสิ้นการทำงาน เช่น กดปุ่มหยุด แล้วปิด LabVIEW หรือใช้ในกรณีที่จะแปลโปรแกรม (Compile) เป็น .exe ให้เพิ่มฟังก์ชันออกจาก LabVIEW เป็นคำสั่งสุดท้ายเพื่อที่จะปิดโปรแกรมที่เขียนขึ้นจาก LabVIEW อย่างสมบูรณ์

ภาคผนวก ข

โปรแกรมสำหรับตรวจจับตำแหน่งของมอเตอร์กระแสตรง

จากโปรแกรมที่ได้จัดทำขึ้นเป็นโปรแกรมในส่วนงานสำหรับตรวจจับตำแหน่งของมอเตอร์กระแสตรง แบ่งออกได้เป็น 4 ส่วนด้วยกัน โดยแต่ละส่วนมีหน้าที่การทำงาน ดังนี้

ส่วนที่ 1 : กำหนด library ที่จำเป็นในการใช้งาน

ส่วนที่ 2 : กำหนดค่าคงที่ต่างๆ ที่ใช้ในโปรแกรมนั้นๆ

ส่วนที่ 3 : ทำหน้าที่รับข้อมูลจากไมโครคอนโทรเลอร์ เพื่อนับจำนวนสัญญาณพัลส์ ที่มาจากตัวเข้ารหัสทั้งหมด

ส่วนที่ 4 : ทำหน้าที่สร้างสัญญาณความกว้างพัลส์

โปรแกรม encoder.c

```
#include <18F2431.h>
#fuses HS,NOWDT,NOPROTECT,NOLVP
#device ADC=10
#use delay(clock=20000000)
#define Vbe 0.00488758553274682306940371456500 49
#use rs232(baud=9600, parity=N, Bits=8, xmit=PIN_C6, rcv=PIN_C7,stream=COM_1)
#use fast_io(A)
#bit QEICONDIR=0xFB6.5
#include STDLIB.H>
unsigned int16 quadhigh=0;
signed int32 position=0;
int32 PWM0,PWM1;
void init_PWM(void);
void init_program(void);
#int_IC2QEI
QuadRollover()
{
// increment high bytes if + direction on interrupt, decrement if -
QEICONDIR?quadhigh++:quadhigh–;
```

ส่วนที่ 1

```
// to see when this happens,
// I find it helpful to toggle a digital out here
}
```

```
        void main()
{

        int16 value1,value2;
float PWM_UD,PWM_LR,value_UD,value_LR;
setup_adc_ports(sAN0);
setup_adc_ports(sAN1);
setup_adc(ADC_CLOCK_INTERNAL);
init_PWM();
init_program();
enable_interrupts(INT_IC2QEI); // interrupt on maxcount or underflow
enable_interrupts(GLOBAL);
```

ส่วนที่ 2

```
        while (1)
{
while (input(PIN_c3)==1){
set_adc_channel(0);
PWM0=read_ADC();
set_adc_channel(1);
PWM1=read_ADC();
PWM_UD=Vbe*(float)PWM0;
PWM_LR=Vbe*(float)PWM1;
position=(((int32)(quadhigh))<<16)+(CAP2BUFL+((int16)CAP2BUFH<<8));
printf("value_LR=100*PWM_LR;
if (value_LR>500)
value_LR=500;
if (value_LR<0)
value_LR=0;
if (input(pin_c1)==1)
output_high(pin_b0);
else
```

ส่วนที่ 3

```
output_low(pin_b0);
value1=abs(value_LR);
PDC0L = make8(value1,0);
PDC0H = make8(value1,1);
value_UD=100*PWM_UD;
if (value_UD>500)
value_UD=500;
if (value_UD<0)
value_UD=0;
if (input(pin_c2)==1)
output_high(pin_b2);
else output_low(pin_b2);
value2=abs(value_UD);
PDC1L = make8(value2,0);
PDC1H = make8(value2,1);
PWMCON1 = 0b00000001; // Output overides synched wrt PWM timebase
DTCON = 0; // no dead band timer
PTCON1 = 0b10000000; // Activate PWM Time Base*/
}
reset_cpu();
}
}
```

```
    void init_PWM(void)
{
PTCON0 = 0; // Time Base config : Free running mode
PWMCON0 = 0b01111111; // PWM 0 to 5 independant mode
// Fpwm = 10 kHz : Fosc = 10Mhz => PTPER = 0x03E8
// Fosc = 10Mhz => PTPER = 0x00fa
PTPERH = 0x00;
PTPERL = 0xf9;
OVDCOND = 0xff; //pwm output controlled by duty cycle
OVDCONS = 0;
FLTCONFIG = 0x80; //fault A config
```

```
PDC1L = 0;
PDC1H = 0;
PDC0L = 0;
PDC0H = 0;
PWMCON1 = 0b00000001; // Output overides synched wrt PWM timebase
DTCON = 0; // no dead band timer
PTCON1 = 0b10000000; // Activate PWM Time Base*/
}

      void init_program(void)
{
QEICON=24; // quad in x4 mode, resettable by maxcount
DFLTCON=49; // noise filter on QEA, QEB,, 1:2 clock
CAP3BUFL=0xFF; // set max count
CAP3BUFH=0xFF;
CAP2BUFL=0;
CAP2BUFH=0;
position=0;
}
```

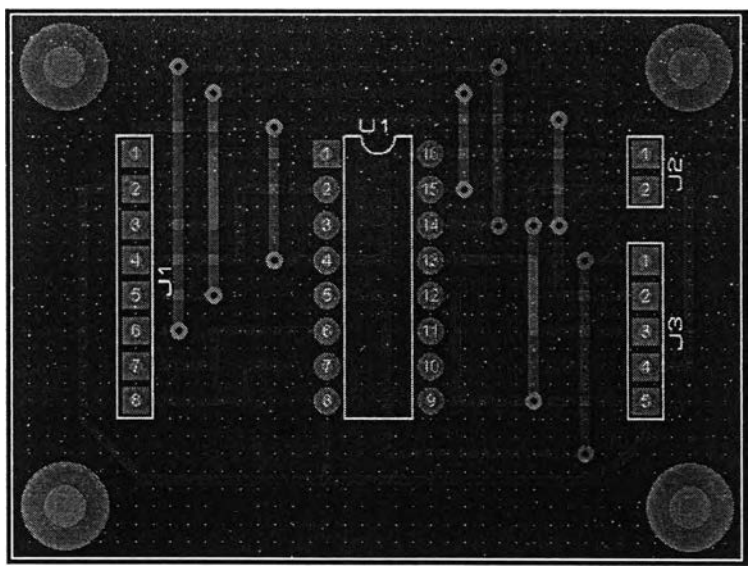
ส่วนที่ 4

ภาคผนวก ค

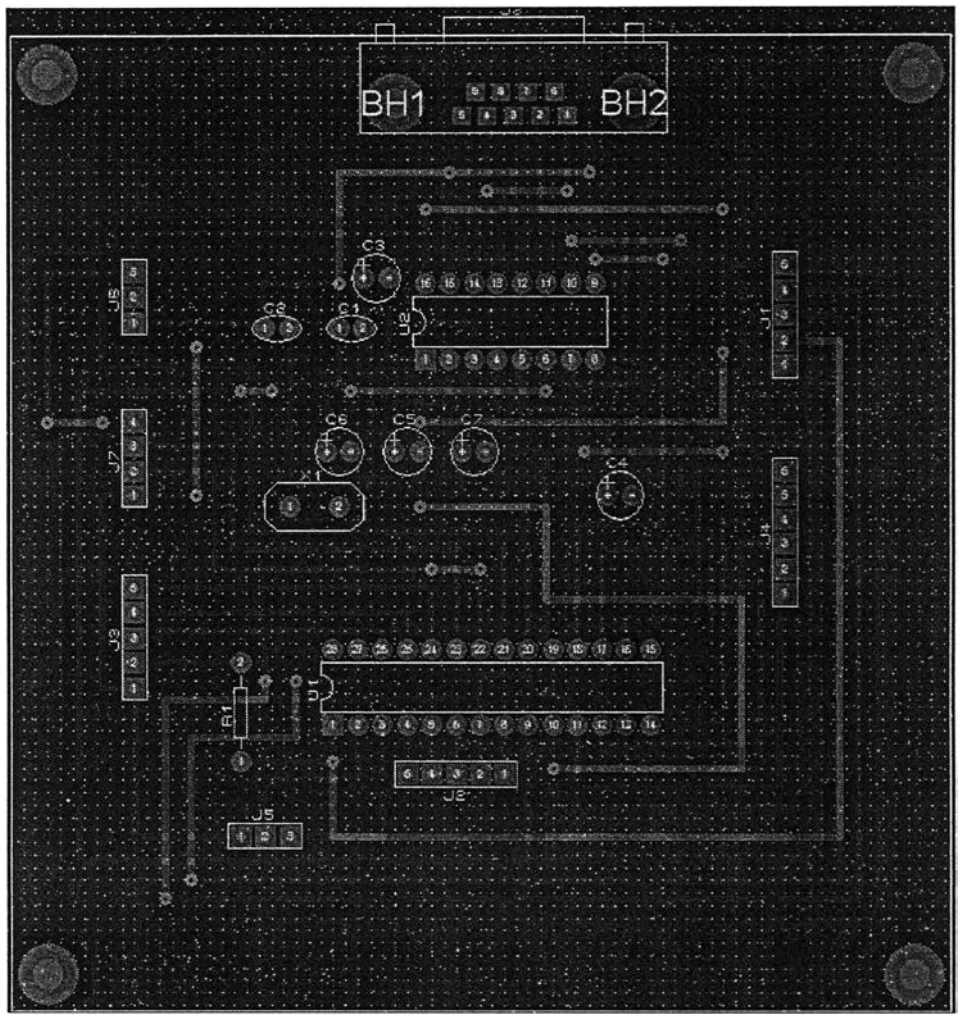
แบบแปลนและลายทองแดงของวงจรสำหรับระบบสายพานลำเลียง

เมื่อสามารถออกแบบวงจรสำหรับแปลสัญญาณจากอุปกรณ์ตรวจจับตำแหน่งของมอเตอร์กระแสตรง และอุปกรณ์ขับมอเตอร์กระแสตรงได้แล้ว ลำดับต่อไปต้องใช้โปรแกรมที่สามารถออกแบบแปลนลายวงจรได้ ซึ่งโปแกรม Proteus เป็นโปรแกรมที่สามารถนำมาใช้ในงานนี้ได้ ลายทองแดงที่สร้างขึ้นได้มาจากการออกแบบแปลนลายวงจรที่ใช้โปรแกรมนี้ สามารถแบ่งออกได้เป็น 3 ส่วนด้วยกัน ดังนี้

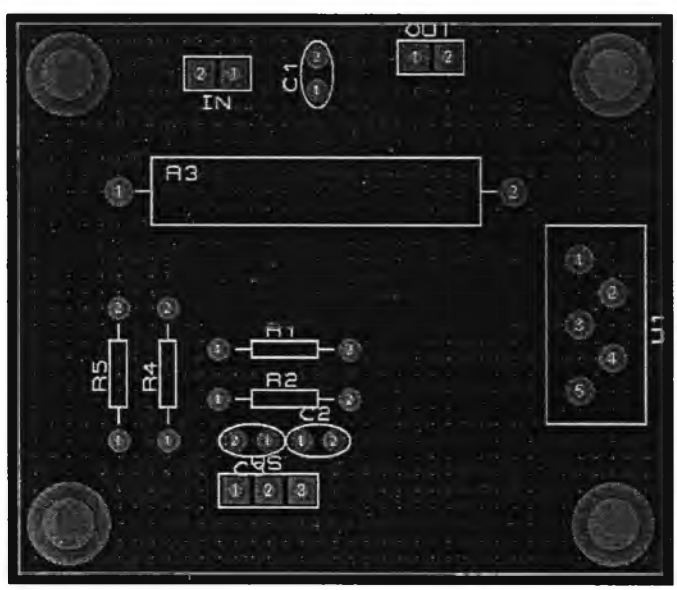
1. รูปที่ ค.1 คือส่วนที่ทำหน้าที่ขยายสัญญาณจากตัวเข้ารหัส เพื่อให้ชุดอุปกรณ์แปลสัญญาณจากอุปกรณ์ตรวจจับตำแหน่งของมอเตอร์กระแสตรงสามารถรับสัญญาณได้
2. รูปที่ ค.2 คือชุดอุปกรณ์แปลสัญญาณจากอุปกรณ์ตรวจจับตำแหน่งของมอเตอร์กระแสตรง
3. รูปที่ ค.3 คือชิ้นส่วนอุปกรณ์ขับมอเตอร์กระแสตรง



รูปที่ ค.1: ชุดอุปกรณ์ขยายสัญญาณจากตัวเข้ารหัส



รูปที่ ค.2: **ชุดอุปกรณ์แปลสัญญาณจากอุปกรณ์ตรวจจับตำแหน่งของมอเตอร์กระแสตรง**



รูปที่ ค.3: ชิ้นส่วนอุปกรณ์ขับมอเตอร์กระแสตรง

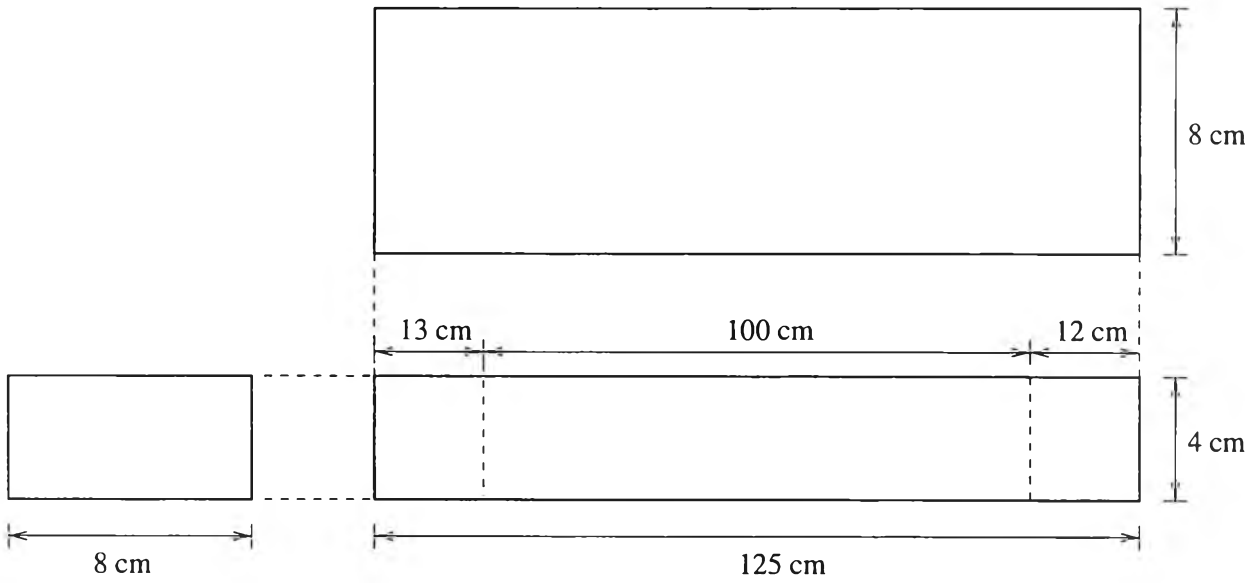
ภาคผนวก ง

โครงสร้างของชิ้นส่วนอุปกรณ์ทางกลศาสตร์ของระบบสายพานลำเลียง

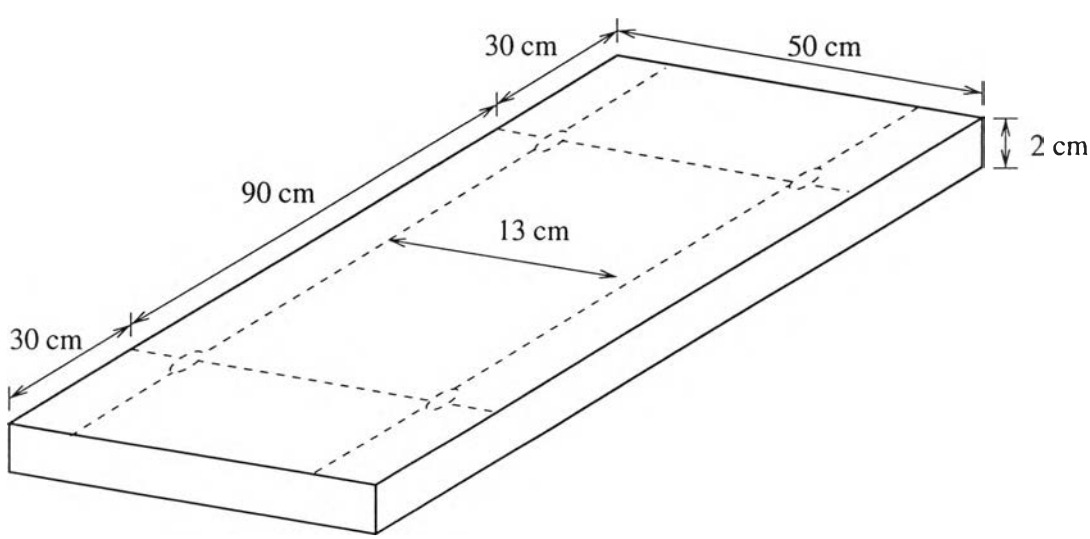
จากส่วนประกอบต่างๆ เมื่อนำมาทำการประกอบเข้าด้วยกันเป็นชิ้นงานโดย

1. นำแกนตามรูปที่ ง.1 ไปประกอบเข้ากับ ขาตั้งทั้ง 4 จากรูปที่ ง.3 ทั้ง 4 ตำแหน่ง และติดตัวปรับตั้งความตึง รูปที่ ง.4 ทางด้านปลายด้านซ้าย
2. นำแกนพร้อมขาตั้ง และตัวปรับความตึงที่ได้ติดตั้งแล้วไปติดตั้งบนฐานรอง รูปที่ ง.2
3. ติดตั้งตัวจับยึดมอเตอร์ รูปที่ ง.5 ลงบนอุปกรณ์จากข้อที่แล้ว
4. ติดตั้งมอเตอร์ลงบนตัวจับยึด
5. ประกอบอุปกรณ์อื่นๆ เช่น อุปกรณ์อิเล็กทรอนิคส์ เป็นอันเสร็จการประกอบฮาร์ดแวร์

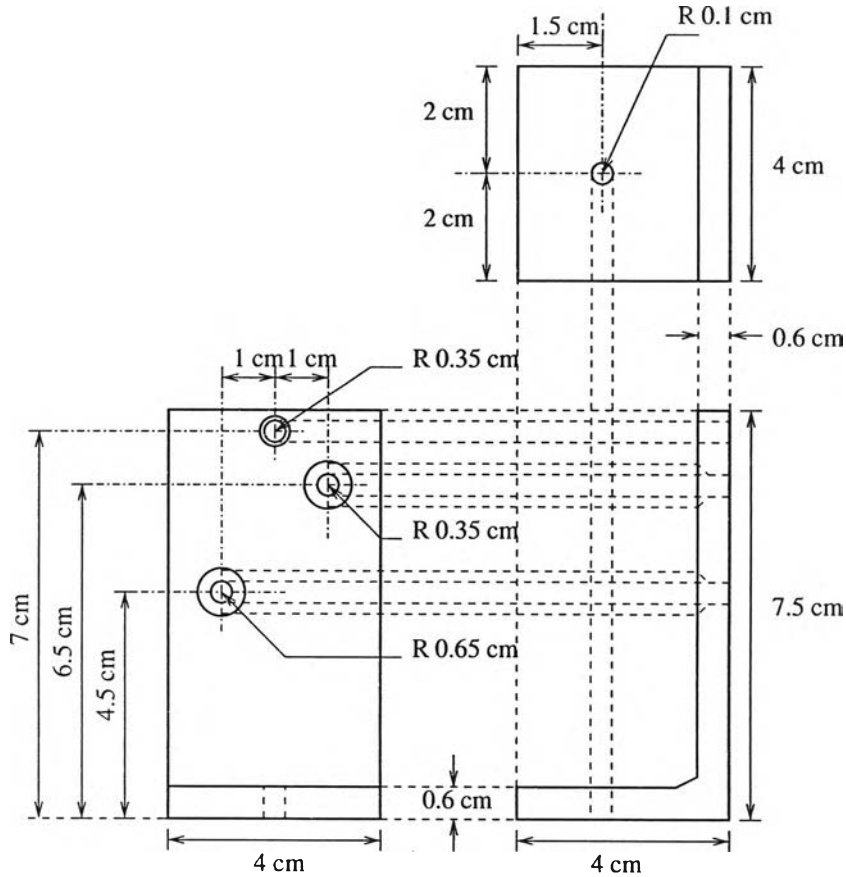
หมายเหตุ ในกรณีต้องการมีขอบกั้นวัสดุ รูปที่ ง.6 ก็สามารถติดตั้งเข้าไปได้ หรือ อาจไม่ต้องติดตั้งก็ได้ เพื่อความสะดวกในการทดลอง



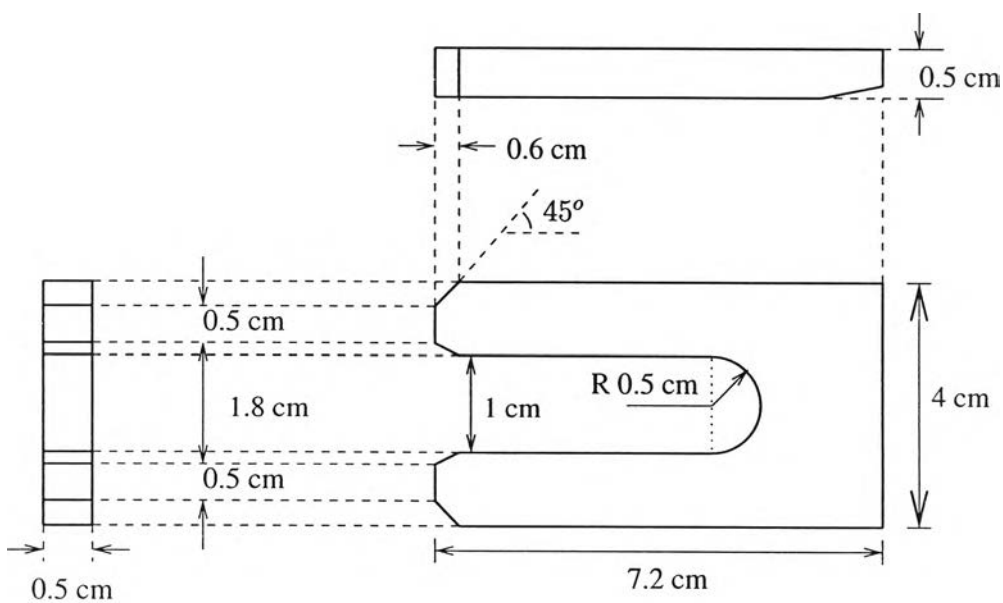
รูปที่ ง.1: แบบวาดโครงสร้างแกน



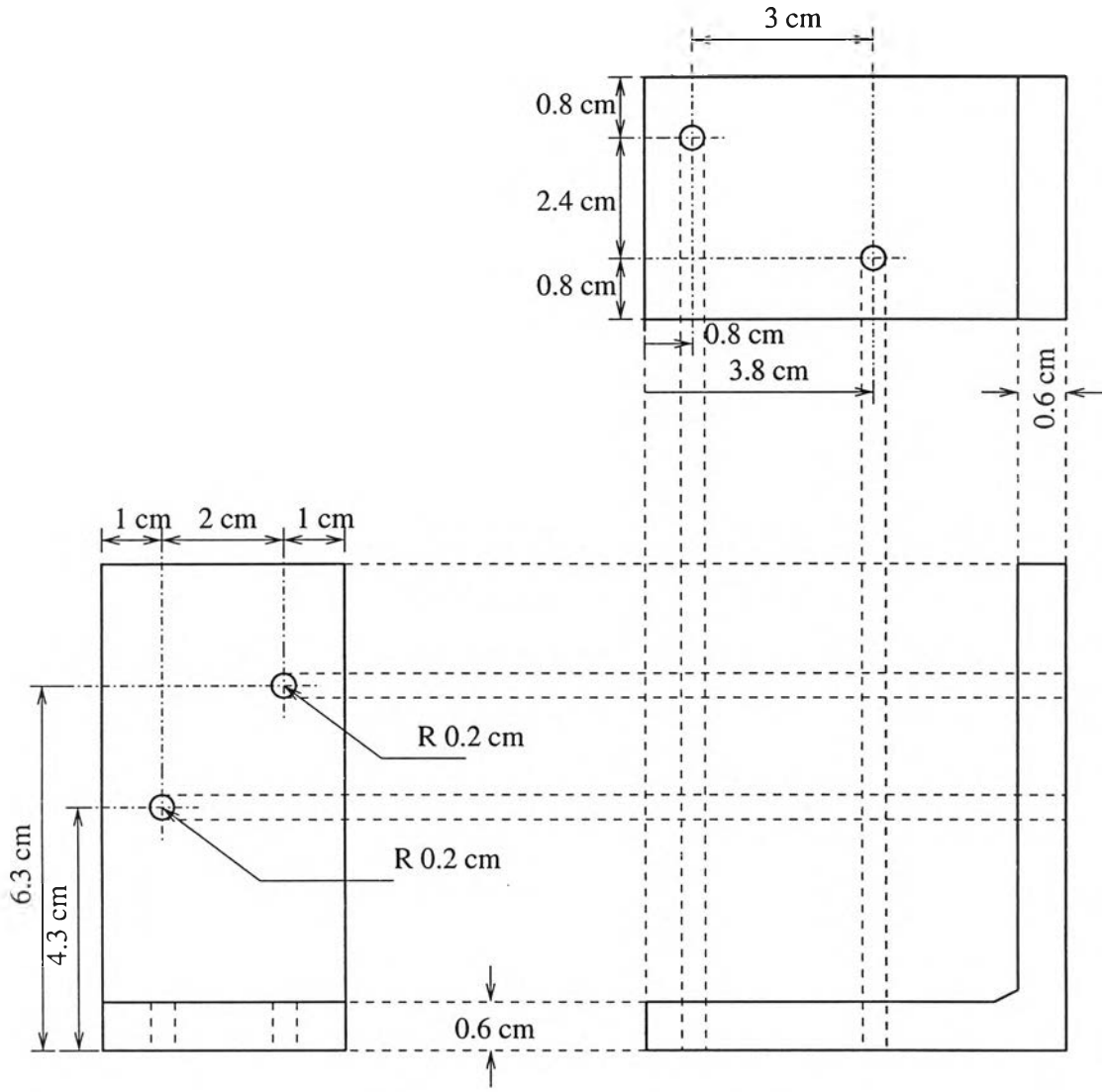
รูปที่ ง.2: แบบวาดฐานรอง



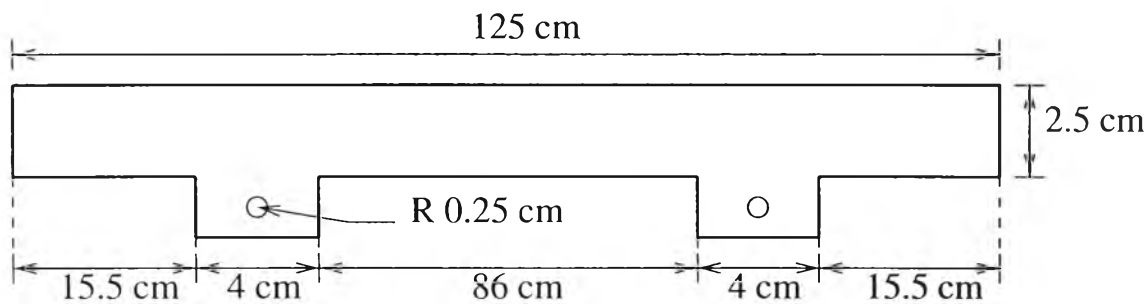
รูปที่ ง.3: แบบวาดขายึดติดสายพาน



รูปที่ ง.4: แบบวาดตัวปรับความตึงสายพาน



รูปที่ ง.5: แบบวาดตัวจับยึดมอเตอร์กับสายพาน



รูปที่ ง.6: แบบวาดขอบกั้นวัสดุบนสายพาน

ภาคผนวก จ

ค่าพารามิเตอร์จากการวนซ้ำในการทดลอง

วิธีปรับจูนการป้อนกลับวนซ้ำให้ทิศทางการค้นหา ค่าประมาณเฮสเซียนเมทริกซ์ และขนาดความยาวของการค้นหาในการทดลองตามตาราง ดังต่อไปนี้

ตารางที่ จ.1: สภาวะที่ไม่มีมวลภาระ เมื่อ $\rho_0 = [0.01 \;\; 0.001 \;\; 0.001]^T$

การวนซ้ำครั้งที่	ทิศทางการค้นหา	ค่าประมาณเฮสเซียนเมทริกซ์	ขนาด
1	$10^5 \times \begin{bmatrix} -0.0066 & 2.0214 & 0.0010 \end{bmatrix}^T$	$10^9 \times \begin{bmatrix} 0.0000 & 0.0003 & 0.0000 \\ 0.0003 & 1.5605 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 \end{bmatrix}$	1
2	$\begin{bmatrix} -13.3733 & -263.9118 & 1.4096 \end{bmatrix}^T$	$10^9 \times \begin{bmatrix} 0.0000 & 0.0003 & 0.0000 \\ 0.0003 & 1.5698 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 \end{bmatrix}$	1
3	$\begin{bmatrix} -12.5078 & -86.8712 & 1.1852 \end{bmatrix}^T$	$10^9 \times \begin{bmatrix} 0.0000 & 0.0002 & 0.0000 \\ 0.0002 & 1.6270 & 0.0001 \\ 0.0000 & -0.0001 & 0.0000 \end{bmatrix}$	1
4	$\begin{bmatrix} -7.2605 & -296.2668 & 1.0620 \end{bmatrix}^T$	$10^9 \times \begin{bmatrix} 0.0000 & 0.0002 & 0.0000 \\ 0.0002 & 1.6275 & -0.0001 \\ 0.0000 & -0.0001 & 0.0000 \end{bmatrix}$	1
5	$\begin{bmatrix} -4.2306 & 49.2768 & 1.3282 \end{bmatrix}^T$	$10^9 \times \begin{bmatrix} 0.0000 & 0.0002 & 0.0000 \\ 0.0002 & 1.6289 & -0.0001 \\ 0.0000 & -0.0001 & 0.0000 \end{bmatrix}$	1
6	$\begin{bmatrix} -3.5986 & -74.9295 & 1.0315 \end{bmatrix}^T$	$10^9 \times \begin{bmatrix} 0.0000 & 0.0002 & 0.0000 \\ 0.0002 & 1.6321 & -0.0001 \\ 0.0000 & -0.0001 & 0.0000 \end{bmatrix}$	1
7	$\begin{bmatrix} -1.2433 & 88.2985 & 1.4542 \end{bmatrix}^T$	$10^9 \times \begin{bmatrix} 0.0000 & 0.0002 & 0.0000 \\ 0.0002 & 1.6322 & -0.0001 \\ 0.0000 & -0.0001 & 0.0000 \end{bmatrix}$	1
8	$\begin{bmatrix} -0.9143 & 176.4346 & 0.2325 \end{bmatrix}^T$	$10^9 \times \begin{bmatrix} 0.0000 & 0.0002 & 0.0000 \\ 0.0002 & 1.6310 & -0.0001 \\ 0.0000 & -0.0001 & 0.0000 \end{bmatrix}$	1

ตารางที่ จ.2: สภาวะที่ไม่มีมวลภาระ เมื่อ $\rho_0 = [0.01 \;\; 0.002 \;\; 0.002]^T$

การวนซ้ำครั้งที่	ทิศทางการค้นหา	ค่าประมาณเฮสเซียนเมทริกซ์	ขนาด
1	$10^5 \times \begin{bmatrix} -0.0034 & 1.0241 & 0.0011 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0001 & 0.0007 & 0.0000 \\ 0.0007 & 4.0380 & -0.0001 \\ 0.0000 & -0.0001 & 0.0001 \end{bmatrix}$	1
2	$10^3 \times \begin{bmatrix} -0.0085 & -1.7506 & 0.0024 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0001 & 0.0009 & 0.0000 \\ 0.0009 & 4.0038 & -0.0002 \\ 0.0000 & -0.0002 & 0.0001 \end{bmatrix}$	1
3	$10^3 \times \begin{bmatrix} -0.0082 & 1.6580 & 0.0020 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0000 & 0.0086 & 0.0000 \\ 0.0086 & 2.4468 & -0.0020 \\ 0.0000 & -0.0020 & 0.0001 \end{bmatrix}$	1
4	$\begin{bmatrix} -3.2608 & 495.5352 & 0.9481 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0000 & 0.0088 & 0.0000 \\ 0.0088 & 2.4289 & -0.0021 \\ 0.0000 & -0.0021 & 0.0001 \end{bmatrix}$	1
5	$\begin{bmatrix} -3.0405 & 313.7845 & 0.9284 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0000 & 0.0089 & 0.0000 \\ 0.0089 & 2.4193 & -0.0021 \\ 0.0000 & -0.0021 & 0.0001 \end{bmatrix}$	1
6	$\begin{bmatrix} -1.3670 & 134.6251 & 0.3347 \end{bmatrix}^T$	$10^9 \times \begin{bmatrix} 0.0000 & 0.0090 & 0.0000 \\ 0.0090 & 2.4177 & -0.0021 \\ 0.0000 & -0.0021 & 0.0001 \end{bmatrix}$	1
7	$\begin{bmatrix} -1.0105 & 114.2095 & 0.2773 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0000 & 0.0090 & 0.0000 \\ 0.0090 & 2.4164 & -0.0021 \\ 0.0000 & -0.0021 & 0.0001 \end{bmatrix}$	1

ตารางที่ จ.3: สภาวะที่มวลภาระหนัก 300 กรัม เมื่อ $\rho_0 = [0.012 \;\; 0.0017 \;\; 0.0018]^T$

การวนซ้ำครั้งที่	ทิศทางการค้นหา	ค่าประมาณเฮสเซียนเมทริกซ์	ขนาด
1	$10^5 \times \begin{bmatrix} -0.0039 & 1.3236 & 0.0010 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0001 & 0.0009 & 0.0000 \\ 0.0009 & 5.3053 & -0.0001 \\ 0.0000 & -0.0001 & 0.0000 \end{bmatrix}$	1
2	$10^3 \times \begin{bmatrix} -0.0096 & 1.9958 & 0.0010 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0001 & 0.0011 & 0.0000 \\ 0.0011 & 5.2614 & -0.0002 \\ 0.0000 & -0.0002 & 0.0000 \end{bmatrix}$	1
3	$10^3 \times \begin{bmatrix} -0.0097 & 1.9610 & 0.0012 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0001 & 0.0127 & 0.0000 \\ 0.0127 & 3.1332 & -0.0022 \\ 0.0000 & -0.0022 & 0.0000 \end{bmatrix}$	1
4	$\begin{bmatrix} -0.7194 & 107.1524 & 0.2691 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0001 & 0.0127 & 0.0000 \\ 0.0127 & 3.1328 & -0.0022 \\ 0.0000 & -0.0022 & 0.0000 \end{bmatrix}$	1
5	$\begin{bmatrix} -0.6482 & 128.4519 & 0.2461 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0001 & 0.0127 & 0.0000 \\ 0.0127 & 3.1120 & -0.0022 \\ 0.0000 & -0.0022 & 0.0000 \end{bmatrix}$	0.01

ตารางที่ จ.4: สภาวะที่มวลภาระหนัก 300 กรัม เมื่อ $\rho_0 = [0.01 \;\; 0.0009 \;\; 0.0008]^T$

การวนซ้ำครั้งที่	ทิศทางการค้นหา	ค่าประมาณเฮสเซียนเมทริกซ์	ขนาด
1	$10^4 \times \begin{bmatrix} -0.0406 & 7.9001 & 0.0171 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0001 & 0.0011 & 0.0000 \\ 0.0011 & 6.4210 & -0.0001 \\ 0.0000 & -0.0001 & 0.0002 \end{bmatrix}$	1
2	$\begin{bmatrix} -6.4229 & -701.9707 & 1.1701 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0001 & 0.0011 & 0.0000 \\ 0.0011 & 6.4689 & -0.0001 \\ 0.0000 & -0.0001 & 0.0002 \end{bmatrix}$	1
3	$\begin{bmatrix} -5.9271 & -609.8872 & 1.0334 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0000 & -0.0046 & 0.0000 \\ -0.0046 & 5.9118 & 0.0008 \\ 0.0000 & 0.0008 & 0.0002 \end{bmatrix}$	1
4	$\begin{bmatrix} -4.5255 & -420.6454 & 1.0225 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0000 & -0.0049 & 0.0000 \\ -0.0049 & 5.8807 & 0.0009 \\ 0.0000 & 0.0009 & 0.0002 \end{bmatrix}$	1
5	$\begin{bmatrix} -2.0827 & 0.6808 & 1.0827 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0000 & -0.0049 & 0.0000 \\ -0.0049 & 5.8879 & 0.0010 \\ 0.0000 & 0.0010 & 0.0002 \end{bmatrix}$	1
6	$\begin{bmatrix} -1.3033 & 137.7955 & 0.2381 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0000 & -0.0049 & 0.0000 \\ -0.0049 & 5.8990 & 0.0009 \\ 0.0000 & 0.0009 & 0.0002 \end{bmatrix}$	0.1
7	$\begin{bmatrix} -1.5569 & 60.5533 & 1.0582 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0000 & -0.0048 & 0.0000 \\ -0.0048 & 5.8834 & 0.0010 \\ 0.0000 & 0.0010 & 0.0002 \end{bmatrix}$	0.01

ตารางที่ จ.5: สภาวะที่มวลภาระหนัก 600 กรัม เมื่อ $\rho_0 = [0.015 \;\; 0.008 \;\; 0.005]^T$

การวนซ้ำครั้งที่	ทิศทางการค้นหา	ค่าประมาณเฮสเซียนเมทริกซ์	ขนาด
1	$10^3 \times \begin{bmatrix} -0.0432 & 5.1455 & 0.0176 \end{bmatrix}^T$	$10^6 \times \begin{bmatrix} 0.0002 & 0.0012 & -0.0001 \\ 0.0012 & 6.5337 & -0.0003 \\ -0.0001 & -0.0003 & 0.0002 \end{bmatrix}$	1
2	$10^3 \times \begin{bmatrix} -0.0013 & 3.7105 & 0.0392 \end{bmatrix}^T$	$10^6 \times \begin{bmatrix} 0.0003 & 0.0121 & 0.0001 \\ 0.0121 & 4.6384 & -0.0111 \\ 0.0001 & -0.0111 & 0.0002 \end{bmatrix}$	1
3	$\begin{bmatrix} 0.5892 & 138.3860 & 4.5166 \end{bmatrix}^T$	$10^6 \times \begin{bmatrix} 0.0003 & 0.0120 & 0.0001 \\ 0.0120 & 4.6430 & -0.0117 \\ 0.0001 & -0.0117 & 0.0002 \end{bmatrix}$	1
4	$\begin{bmatrix} 0.5019 & 133.5347 & 4.0882 \end{bmatrix}^T$	$10^6 \times \begin{bmatrix} 0.0003 & 0.0115 & 0.0001 \\ 0.0115 & 4.5206 & -0.0156 \\ 0.0001 & -0.0156 & 0.0001 \end{bmatrix}$	1

ตารางที่ จ.6: สภาวะที่มวลภาระหนัก 600 กรัม เมื่อ $\rho_0 = [0.019\ \ 0.003\ \ 0.002]^T$

การวนซ้ำครั้งที่	ทิศทางการค้นหา	ค่าประมาณเฮสเซียนเมทริกซ์	ขนาด
1	$10^3 \times \begin{bmatrix} -0.0753 & 1.9332 & 0.0322 \end{bmatrix}^T$	$10^7 \times \begin{bmatrix} 0.0000 & 0.0004 & 0.0000 \\ 0.0004 & 2.1711 & -0.0001 \\ 0.0000 & -0.0001 & 0.0001 \end{bmatrix}$	1
2	$\begin{bmatrix} -0.5781 & 123.1884 & 0.1627 \end{bmatrix}^T$	$10^7 \times \begin{bmatrix} 0.0000 & 0.0004 & 0.0000 \\ 0.0004 & 2.1690 & -0.0001 \\ 0.0000 & -0.0001 & 0.0001 \end{bmatrix}$	0.1
3	$\begin{bmatrix} -0.5633 & 122.5857 & 0.1613 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0000 & 0.0038 & 0.0000 \\ 0.0038 & 1.4486 & -0.0010 \\ 0.0000 & -0.0010 & 0.0001 \end{bmatrix}$	1

ตารางที่ จ.7: สภาวะที่มวลภาระหนัก 900 กรัม เมื่อ $\rho_0 = [0.01\ \ 0.002\ \ 0.002]^T$

การวนซ้ำครั้งที่	ทิศทางการค้นหา	ค่าประมาณเฮสเซียนเมทริกซ์	ขนาด
1	$10^5 \times \begin{bmatrix} -0.0036 & 1.2176 & 0.0012 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0001 & 0.0007 & 0.0000 \\ 0.0007 & 4.4504 & -0.0001 \\ 0.0000 & -0.0001 & 0.0001 \end{bmatrix}$	1
2	$10^3 \times \begin{bmatrix} -0.0064 & 1.0576 & 0.0010 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0001 & 0.0009 & 0.0000 \\ 0.0009 & 4.4360 & -0.0001 \\ 0.0000 & -0.0001 & 0.0001 \end{bmatrix}$	0.01
3	$\begin{bmatrix} -6.7709 & 852.7898 & 1.2122 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0000 & 0.0121 & 0.0000 \\ 0.0121 & 5.3494 & -0.0036 \\ 0.0000 & -0.0036 & 0.0001 \end{bmatrix}$	1
4	$\begin{bmatrix} -1.8350 & 119.5197 & 1.0618 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0000 & 0.0122 & 0.0000 \\ 0.0122 & 5.3491 & -0.0036 \\ 0.0000 & -0.0036 & 0.0001 \end{bmatrix}$	1
5	$\begin{bmatrix} -1.8393 & 47.5074 & 0.9758 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0000 & 0.0122 & 0.0000 \\ 0.0122 & 5.3631 & -0.0036 \\ 0.0000 & -0.0036 & 0.0001 \end{bmatrix}$	1

ตารางที่ จ.8: สภาวะที่มวลภาระหนัก 900 กรัม เมื่อ $\rho_0 = [0.02\ \ 0.0001\ \ 0.0001]^T$

การวนซ้ำครั้งที่	ทิศทางการค้นหา	ค่าประมาณเฮสเซียนเมทริกซ์	ขนาด
1	$10^5 \times \begin{bmatrix} -0.0016 & -2.1797 & 0.0002 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0000 & 0.0019 & 0.0000 \\ 0.0019 & 6.3313 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 \end{bmatrix}$	1
2	$\begin{bmatrix} -1.6972 & -110.8372 & 1.7113 \end{bmatrix}^T$	$10^7 \times \begin{bmatrix} 0.0000 & 0.0019 & 0.0000 \\ 0.0019 & 6.3401 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 \end{bmatrix}$	0.1
3	$\begin{bmatrix} -2.3794 & -354.7485 & 0.9687 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0000 & 0.0019 & 0.0000 \\ 0.0019 & 6.3989 & 0.0003 \\ 0.0000 & 0.0003 & 0.0000 \end{bmatrix}$	0.1
4	$\begin{bmatrix} -1.7210 & 114.4935 & -0.4895 \end{bmatrix}^T$	$10^7 \times \begin{bmatrix} 0.0000 & 0.0017 & 0.0000 \\ 0.0017 & 6.4278 & 0.0002 \\ 0.0000 & 0.0002 & 0.0000 \end{bmatrix}$	0.1
5	$\begin{bmatrix} -4.7496 & -771.2437 & 4.9289 \end{bmatrix}^T$	$10^8 \times \begin{bmatrix} 0.0000 & 0.0081 & 0.0000 \\ 0.0081 & 8.1969 & -0.0106 \\ 0.0000 & -0.0106 & 0.0001 \end{bmatrix}$	1

ประวัติผู้เขียนวิทยานิพนธ์

นายกิตติพงษ์ เธียรจันทร์วงศ์ เป็นบุตรของนายเฉลียว เธียรจันทร์วงศ์ และนางสุวณีย์ แซ่ตั้น สำเร็จการศึกษาระดับปริญญาวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า จากสถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ เมื่อปีการศึกษา 2547 และเข้าศึกษาต่อในหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต ในปีการศึกษาถัดมา ณ ภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย สังกัดห้องปฏิบัติการวิจัยระบบควบคุม โดยได้รับทุนอุดหนุนการศึกษาจากโครงการศิษย์ก้นกุฏิ ภาควิชาวิศวกรรมไฟฟ้า

ผลงานทางวิชาการที่ได้เข้าร่วมในสัมมนาวิชาการในประเทศ

การออกแบบตัวควบคุมพลวัตสำหรับระบบขับเคลื่อนสายพานลำเลียงโดยวิธีปรับจูนการป้อนกลับวนซ้ำ ผู้แต่งร่วมคือ รศ.ดร.เดวิด บรรเจิดพงศ์ชัย ณ การประชุมวิชาการทางวิศวกรรมไฟฟ้าครั้งที่ 28 ภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ มหาวิทยาลัยธรรมศาสตร์ เดือนตุลาคม พ.ศ. 2548

ผลงานที่ได้เข้าร่วมกิจกรรมทางวิชาการนอกหลักสูตรระหว่างการศึกษาในระดับปริญญามหาบัณฑิต

การควบคุมระบบขับเคลื่อนสายพานลำเลียง งานนิทรรศการทางวิศวกรรม ครั้งที่ 14 ณ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย เดือนพฤศจิกายน พ.ศ. 2548