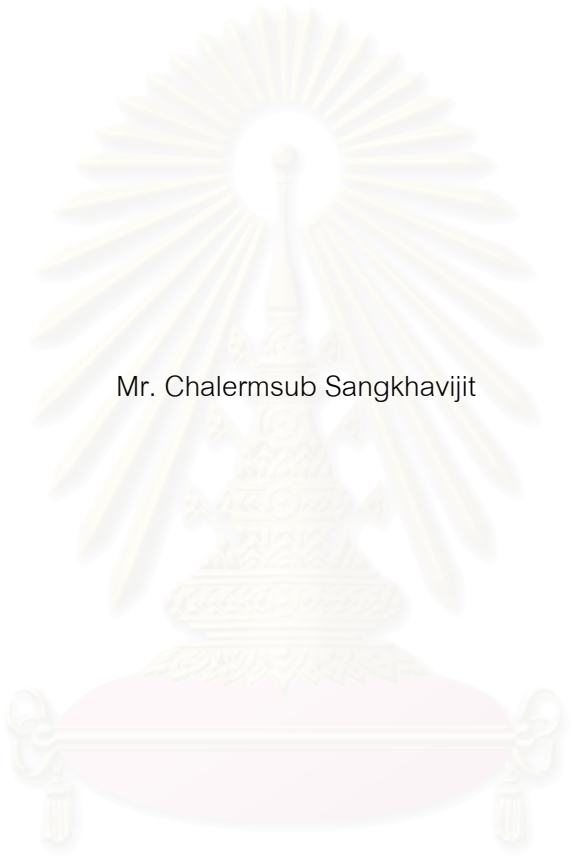
วิธีการป้องกันและลดข้อผิดพลาดในการเขียนโปรแกรมควบคุมมอเตอร์

นายเฉลิมทรัพย์ สังขวิจิตร

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2546
ISBN 974-17-3609-6

PREVENTION AND REDUCTION OF ERRORS IN MOTOR CONTROL PROGRAMMING

Mr. Chalermsub Sangkhavijit



A Thesis Submitted in Partial Fulfillment of the Requirements

for the Degree of Master of Sciencein Computer Science

Department of Computer Engineering

Faculty of Engineering

Chulalongkorn University

Academic Year 2003

ISBN 974-17-3609-6

หัวข้อวิทยานิพนธ์	วิธีการป้องกันและลดข้อผิดพลาดในการเขียนโปรแกรมควบคุมมอเตอร์
โดย	นายเฉลิมทรัพย์ สังขวิจิตร
สาขาวิชา	วิทยาศาสตร์คอมพิวเตอร์
อาจารย์ที่ปรึกษา	รองศาสตราจารย์ ดร.ประภาส จงสถิตย์วัฒนา
อาจารย์ที่ปรึกษาร่วม	อาจารย์ ดร.สมบูรณ์ แสงวงค์วาณิชย์

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

…….. คณบดีคณะวิศวกรรมศาสตร์
(ศาสตราจารย์ ดร.สมศักดิ์ ปัญญาแก้ว)

คณะกรรมการสอบวิทยานิพนธ์

…… ประธานกรรมการ
(รองศาสตราจารย์ ดร.สาธิต วงศ์ประทีป)

…….. อาจารย์ที่ปรึกษา
(รองศาสตราจารย์ ดร.ประภาส จงสถิตย์วัฒนา)

…… อาจารย์ที่ปรึกษาร่วม
(อาจารย์ ดร.สมบูรณ์ แสงวงค์วาณิชย์)

…… กรรมการ
(อาจารย์ ดร.อาทิตย์ ทองทักษ์)

เฉลิมทรัพย์ สังขวิจิตร:วิธีการป้องกันและลดข้อผิดพลาดในการเขียนโปรแกรมควบคุมมอเตอร์. (Prevention and Reduction of Errors in Motor Control Programming) อ.ที่ปรึกษา :รศ.ดร.ประภาส จงสถิตย์วัฒนา, อ.ที่ปรึกษาร่วม : อ.ดร.สมบูรณ์ แสงวงค์วาณิชย์ 70 หน้า. ISBN 974-17-3609-6.

การเขียนโปรแกรมควบคุมมอเตอร์มีการนำเข้าข้อมูลจากตัวรับรู้ เพื่อนำมาคำนวณ และส่งออกข้อมูลไปควบคุมมอเตอร์ตามที่ต้องการ โปรแกรมการคำนวณตามสมการควบคุม โดยทั่วไปนิยมใช้การคำนวณเลขแบบจุดตรึงเนื่องจากมีข้อดีที่ใช้เวลาในการคำนวณน้อย ใช้ขนาดพื้นที่หน่วยความจำน้อย และสามารถกำหนดช่วงค่าที่ต้องการได้ แต่ก็มีข้อด้อยในเรื่องการเก็บค่าที่จำกัดขึ้นอยู่กับจำนวนบิตที่ใช้ในการเก็บค่าข้อมูล ทำให้มักเกิดปัญหาในการคำนวณ โดยปัญหาที่พบบ่อยได้แก่ ปัญหาการล้น ปัญหาน้อยเกินเก็บ ปัญหาการตัดปลาย ปัญหาการปัดเศษ และปัญหามาตราส่วน ซึ่งล้วนแล้วแต่ทำให้เกิดความคลาดเคลื่อน และความผิดพลาดในการคำนวณทั้งสิ้น

งานวิจัยนี้ได้นำเสนอวิธีการวิเคราะห์ค่าผิดพลาด ของโปรแกรมในงานควบคุมมอเตอร์ เพื่อป้องกันและลดข้อผิดพลาดในส่วนการคำนวณ โดยใช้วิธีการคำนวณเลขคณิตแบบช่วงกับค่าในรูปแบบเลขทศนิยม และแบบจุดตรึง เพื่อใช้ในการรับประกันผลลัพธ์ที่ได้ว่าจะอยู่ภายในขอบเขตของผลการคำนวณ และเป็นไปตามเงื่อนไขของการคำนวณ แต่ถ้าไม่เป็นไปตามเงื่อนไข หรือเกิดความผิดพลาดขึ้นจะมีการรายงานให้ทราบเพื่อนำไปใช้ปรับปรุงแก้ไขต่อไป งานวิจัยนี้จะเลือกใช้หน่วยประมวลผลของบริษัท Hitachi รุ่น SH1 เป็นหลักในการพัฒนา เนื่องจากมีการใช้งานจริงในอุตสาหกรรมภายในประเทศ

ภาควิชาวิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต.....
สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ลายมือชื่ออาจารย์ที่ปรึกษา.....
ปีการศึกษา 2546 ลายมือชื่ออาจารย์ที่ปรึกษาร่วม.....

4470262021 : MAJOR COMPUTER SCIENCE

KEY WORD: Error Analysis/ Interval Arithmetic / Fixed-point/ Q-format/ Embedded Programming

CHALERMSUB SANGKHAVIJIT: PREVENTION AND REDUCTION OF ERRORS IN MOTOR CONTROL PROGRAMMING. THESIS ADVISOR : ASSOC. PROF. PRABHAS CHONGSTITVATANA, THESIS COADVISOR : SOMBOON SANGWONGWANICH, 70 pp. ISBN 974-17-3609-6.

Motor control programs require taking inputs from sensing devices to perform calculation and output data to control motors. In general, the program which performs the calculation according to control equations employs the fixed-point calculation because of its advantage in terms of speed, its minimal requirement on memory, and its ability to specify the value range. The disadvantage of fixed-point calculation is the limited number of bits to store values, and this causes problems such as overflow, underflow, truncation, rounding-off, and scaling,resulting in tolerance and error in calculation.

This research proposes an error analysis of programs in motor control applications to prevent and reduce errors in calculation. The proposed analysis method applies the interval arithmetic calculation to decimal and fixed-point numbers to assure that the results are in the required range and conform to calculation constraints. If the constraints are violated or errors occur, they will be reported to the users to help them make the necessary corrections. The application of the proposed analysis is confined to the processor from Hitachi model SH1 because it is widely used in the domestic industry.

Department Computer Engineering Student's signature.....
Field of study Computer Science Advisor's signature.....
Academic year 2003 Co-advisor's signature.....

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จได้ด้วยความช่วยเหลืออย่างดียิ่งจาก รศ.ดร.ประภาส จงสถิตย์วัฒนา อาจารย์ที่ปรึกษาวิทยานิพนธ์ และ ดร.สมบูรณ์ แสงวงค์วาณิชย์ อาจารย์ที่ปรึกษาร่วม ซึ่งท่านได้เสียสละเวลาในการให้คำแนะนำ และแนวทางการวิจัย ตลอดจนข้อคิดเห็นที่มีประโยชน์มากมาย เพื่อประกอบการวิจัยของข้าพเจ้ามาโดยตลอด

ขอขอบคุณอาจารย์ รศ.ดร.สาธิต วงศ์ประทีป และอาจารย์ ดร.อาทิตย์ ทองทักษ์ กรรมการวิทยานิพนธ์ ที่ท่านได้กรุณาให้คำแนะนำและข้อชี้แนะในการตรวจและแก้ไขวิทยานิพนธ์ฉบับนี้

ขอขอบคุณคุณคิดชอบ ไวยสุศรี ดร.สุรพงศ์ สุวรรณกวิน และเพื่อน ๆ พี่ ๆ น้อง ๆ ห้องวิจัยปฏิบัติการอิเล็กทรอนิกส์กำลัง และห้องปฏิบัติการวิจัย SPACE รวมถึงอาจารย์ เพื่อนๆพี่ ๆน้องๆ ในภาควิศวกรรมคอมพิวเตอร์ ที่ให้คำแนะนำ ข้อคิดเห็น และกำลังใจแก่ข้าพเจ้าตลอดเวลาที่ศึกษาในภาควิชาวิศวกรรมคอมพิวเตอร์แห่งนี้

สุดท้ายนี้ข้าพเจ้าขอกราบขอบพระคุณบิดามารดา ที่ให้โอกาสทางการศึกษาและเป็นกำลังใจด้วยดีเสมอมา

สารบัญ

บทที่ หน้า

บทคัดย่อภาษาไทย ..... ง

บทคัดย่อภาษาอังกฤษ ..... จ

กิตติกรรมประกาศ ..... ฉ

สารบัญ ..... ช

สารบัญตาราง ..... ฌ

สารบัญรูปภาพ ..... ฏ

บทที่

1 บทนำ 1

1.1 ความเป็นมาและความสำคัญของปัญหา 1

1.2 วัตถุประสงค์ของการวิจัย 2

1.3 ขอบเขตของการวิจัย 3

1.4 ขั้นตอนและวิธีดำเนินการวิจัย 3

1.5 ประโยชน์ที่คาดว่าจะได้รับ 4

1.6 ผลงานที่ตีพิมพ์จากงานวิจัย 4

1.7 โครงสร้างวิทยานิพนธ์ 4

2 ทฤษฎีและงานวิจัยที่เกี่ยวข้อง 5

2.1 ทฤษฎีที่เกี่ยวข้อง 5

2.1.1 การประมวลผลเลขฐานสอง 5

2.1.1.1 ระบบเลขฐานสอง 5

2.1.1.2 การจัดรูปแบบเลขฐานสอง 7

2.1.1.3 เลขคิดเครื่องหมาย และไม่คิดเครื่องหมาย 8

2.1.1.4 การหาค่าแบบจุดตรึง 11

2.1.1.5 การคำนวณเลขเศษ 11

2.1.1.6 การหาค่าความเที่ยง 12

2.1.1.7 การหาค่าความทน 12

2.1.2 การจัดรูปแบบคิว 13

2.1.2.1 การหาค่าพิสัยของเลขแบบจุดตรึง 17

2.1.2.2 การดำเนินการทางคณิตศาสตร์ 20

2.1.2.2.1 การบวกเลขฐานสอง 21

สารบัญ (ต่อ)

บทที่ หน้า

2.1.2.2.2 การบวกแบบมอดุโล 22

2.1.2.2.3 การลบเลขฐานสอง 22

2.1.2.2.4 การคูณเลขฐานสอง 23

2.1.2.2.5 การหารเลขฐานสอง 26

2.1.2.3 การวิเคราะห์ความผิดพลาดของการดำเนินการแบบจุดตรึง 28

2.1.3 เลขคณิตแบบช่วง 32

2.1.4 รูปแบบการอิงดรรชนีแบบ IEEE 37

2.2 งานวิจัยที่เกี่ยวข้อง 38

2.2.1 การประยุกต์ตัวควบคุมราคาถูกรุ่น SH1 ของบริษัท Hitachi ในงานควบคุม พีไอดี สำหรับระบบมอเตอร์กระแสไฟตรงสามเฟสแบบไร้แปรง 38

2.2.2 การจัดกระทำความผิดปรกติในการคำนวณเชิงวิทยาศาสตร์ 39

3. ขั้นตอนวิธีทดสอบและวิเคราะห์ค่าผิดพลาดของโปรแกรมในส่วนการคำนวณ 40

3.1 การจำลองหาการค่าผลการคำนวณ 40

4. การทดลองและผลการทดลอง 44

4.1 เครื่องมือในการทดลอง 44

4.2 วิธีการทดลองและผลการทดลอง 44

5. สรุปผลการวิจัย และข้อเสนอแนะ 66

5.1 สรุปผลการวิจัย 66

5.2 ข้อเสนอแนะ 66

รายการอ้างอิง 68

ประวัติผู้เขียนวิทยานิพนธ์ 70

สารบัญตาราง

ตาราง หน้า

ตารางที่ 2.1 การแทนค่าที่เป็นไปได้ทั้งหมดสำหรับเลขฐานสองจำนวน 4 บิต 7
ตารางที่ 2.2 ค่าเลขแบบคิดเครื่องหมาย และเลขแบบไม่คิดเครื่องหมาย 9
ตารางที่ 2.3 การแปลงค่าจากเลขฐานสิบเป็นเลขฐานสองโดยใช้การจัดรูปแบบคิว 14
ตารางที่ 2.4 การแปลงจากเลขฐานสองโดยใช้การจัดรูปแบบคิวเป็นเลขฐานสิบ 14
ตารางที่ 2.5 ตัวอย่างการเก็บค่าฟังก์ชั่นโคไซน์ด้วยการจัดรูปแบบ Q6 โดยค่าที่ใช้มีการปัดเศษขึ้นเมื่อเลขหลังหลักสุดท้ายที่ต้องการมีค่า ≥ 5 ถ้าน้อยกว่าจะถูกปัดลง 17
ตารางที่ 2.6 ตารางค่าพิสัยสูงสุดที่เป็นไปได้ตามจำนวนบิตทั้งเลขจำนวนเต็ม และเลขเศษ 18
ตารางที่ 2.7 ค่าพิสัยของการเก็บข้อมูล 8 บิต 20
ตารางที่ 2.8 ผลการบวกเลขฐานสอง โดยช่องที่มีสีทึบจะหมายถึงมีการทดเลข 21
ตารางที่ 2.9 ตัวอย่างการบวกเลขฐานสอง 21
ตารางที่ 2.10 กรณีที่แย่ที่สุด ตัวอย่าง 3 กรณี 21
ตารางที่ 2.11 ผลการลบเลขฐานสอง โดยช่องที่มีสีทึบจะหมายถึงมีการยืมเลข 22
ตารางที่ 2.12 ตัวอย่างการลบเลขฐานสองวิธีที่ 1 (ลบโดยตรง) 23
ตารางที่ 2.13 ตัวอย่างการลบเลขฐานสองวิธีที่ 2 (บวกกับค่าลบที่ได้จากการทำส่วนเติมเต็มของสอง) 23
ตารางที่ 2.14 ผลการคูณเลขฐานสอง 24
ตารางที่ 2.15 การดำเนินการเลขคณิต โดยสัญลักษณ์ $\perp$ แทนการดำเนินการที่ไม่นิยาม R แทนจำนวนจริง PR แทนจำนวนจริงบวก และ NR แทนจำนวนจริงลบ โดยแนวนอนแทนค่า y แนวตั้ง 35
ตารางที่ 2.16 การจัดจำแนกช่วงไม่ว่างเปล่าโดยเครื่องหมาย และ $a \le b$ 35
ตารางที่ 2.17 กรณีวิเคราะห์สำหรับการคูณจำนวนจริงแบบช่วง $[a,b] \times [c,d]$ 36
ตารางที่ 2.18 กรณีวิเคราะห์สำหรับการหารจำนวนจริงแบบช่วง $[a,b] / [c,d]$ เมื่อ $a \le b$, $c \le d$ และต้องไม่เป็น [0,0] ทั้งคู่ หลักสุดท้ายอ้างถึงการพิสูจน์ โดย D = Directly Proof, “S1”, “S2” เป็นการอ้างถึงการใช้หลักสมมาตรเพื่อลดรูปให้อยู่ในกรณีที่กล่าวถึงก่อนแล้ว 36
ตารางที่ 2.19 ค่าตัวเลขซึ่งเป็นข้อยกเว้นของ IEEE-754 38
ตารางที่ 4.1 ค่าเริ่มต้นและเงื่อนไขการคำนวณสมการอันดับหนึ่ง โดยผลลัพธ์ของการคำนวณในแต่ละขั้นจะอยู่ในรูปแบบ Q14 และสัญลักษณ์ U = Unsigned 47
ตารางที่ 4.2 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการอันดับที่หนึ่ง โดยใช้

สารบัญตาราง (ต่อ)

ตาราง หน้า

ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.1 47
ตารางที่ 4.3 ค่าเริ่มต้นและเงื่อนไขการคำนวณสมการอันดับหนึ่ง โดยผลลัพธ์ของการคำนวณในแต่ละขั้นจะอยู่ในรูปแบบ Q15 และสัญลักษณ์ U = Unsigned 48
ตารางที่ 4.4 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการอันดับที่หนึ่ง โดยใช้ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.3 48
ตารางที่ 4.5 ค่าเริ่มต้นและเงื่อนไขการคำนวณสมการควบคุม PI โดยผลลัพธ์ของการคำนวณในแต่ละขั้นจะอยู่ในรูปแบบ Q8 และสัญลักษณ์ U = Unsigned 51
ตารางที่ 4.6 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการควบคุม PI โดยใช้ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.5 51
ตารางที่ 4.7 ค่าเริ่มต้นและเงื่อนไขการคำนวณสมการควบคุม PI โดยผลลัพธ์ของการคำนวณในแต่ละขั้นจะอยู่ในรูปแบบ Q12 และสัญลักษณ์ U = Unsigned 52
ตารางที่ 4.8 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการควบคุม PI โดยใช้ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.7 52
ตารางที่ 4.9 ค่าเริ่มต้นและเงื่อนไขการคำนวณสมการควบคุม PI โดยผลลัพธ์ของการคำนวณในแต่ละขั้นจะอยู่ในรูปแบบ Q16 และสัญลักษณ์ U = Unsigned 53
ตารางที่ 4.10 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการควบคุม PI โดยใช้ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.9 53
ตารางที่ 4.11 ค่าเริ่มต้นและเงื่อนไขการคำนวณสมการควบคุม PI โดยผลลัพธ์ของการคำนวณในแต่ละขั้นจะอยู่ในรูปแบบ Q4 และสัญลักษณ์ U = Unsigned 55
ตารางที่ 4.12 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการควบคุม PI โดยใช้ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.11 55
ตารางที่ 4.13 ค่าเริ่มต้นและเงื่อนไขการคำนวณสมการควบคุม PI โดยผลลัพธ์ของการคำนวณในแต่ละขั้นจะอยู่ในรูปแบบ Q8 และสัญลักษณ์ U = Unsigned 56
ตารางที่ 4.14 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการควบคุม PI โดยใช้ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.13 56
ตารางที่ 4.15 ค่าเริ่มต้นและเงื่อนไขการคำนวณสมการควบคุม PI โดยผลลัพธ์ของการคำนวณในแต่ละขั้นจะอยู่ในรูปแบบ Q12 และสัญลักษณ์ U = Unsigned 57
ตารางที่ 4.16 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการควบคุม PI โดยใช้

สารบัญตาราง (ต่อ)

ตาราง หน้า

ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.15 57
ตารางที่ 4.17 ค่าเริ่มต้นและเงื่อนไขการคำนวณสมการการแปลงแกน โดยผลลัพธ์ของการคำนวณในแต่ละขั้นจะอยู่ในรูปแบบ Q15, สัญลักษณ์ U = Unsigned และสัญลักษณ์ π (Pi) มีค่าเท่ากับ 3.14159265358 60
ตารางที่ 4.18 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการการแปลงแกน โดยใช้ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.17 ยกเว้นชนิดตัวแปรที่จะใช้เป็น Float ทั้งหมด และใช้คลังโปรแกรมฟังก์ชันตรีโกณมิติของคอมไพเลอร์ Gcc 61
ตารางที่ 4.19 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการการแปลงแกน โดยใช้ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.17 และใช้คลังโปรแกรมฟังก์ชันตรีโกณมิติของคอมไพเลอร์ Gcc 62
ตารางที่ 4.20 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการการแปลงแกน โดยใช้ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.17 ยกเว้นชนิดตัวแปรที่จะใช้เป็น Float ทั้งหมด และใช้ตารางค้นหาสำหรับฟังก์ชันตรีโกณมิติ กำหนดค่าในตารางค้นหาให้อยู่ในรูปแบบคิว 15 63
ตารางที่ 4.21 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการการแปลงแกน โดยใช้ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.17 และใช้ตารางค้นหาสำหรับฟังก์ชันตรีโกณมิติ กำหนดค่าในตารางค้นหาให้อยู่ในรูปแบบคิว 15 64

สารบัญภาพ

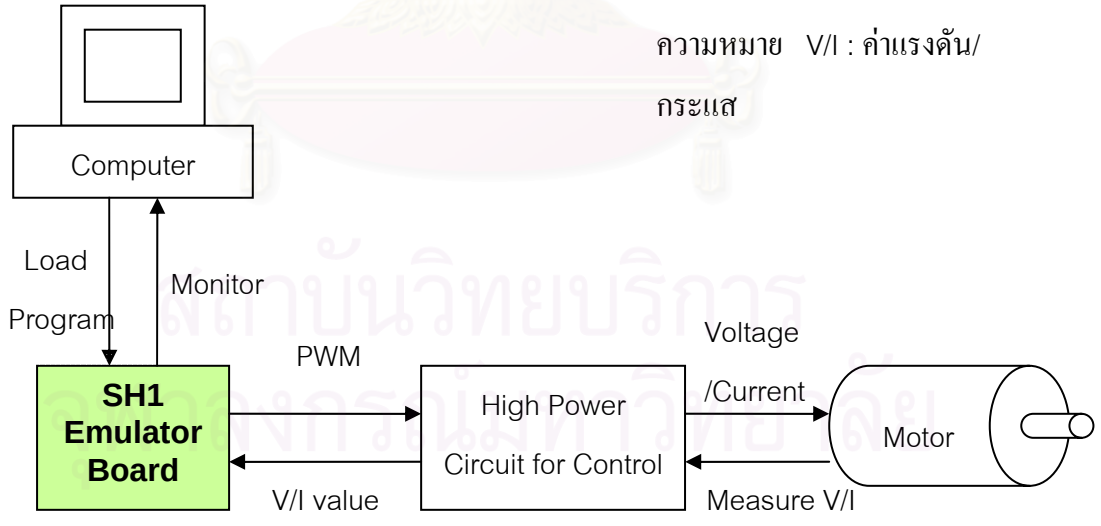
ภาพประกอบ หน้า

รูปที่ 1.1 ภาพโดยรวมของระบบควบคุมมอเตอร์ 1
รูปที่ 2.1 การแปลงค่าจากเลขฐานสองไปเป็นเลขฐานสิบ 6
รูปที่ 2.2 การเก็บค่าตัวเลขในรูปแบบของเลขฐานสอง 8
รูปที่ 2.3 การขยายจำนวนบิต รูป (ก) เป็นเลขแบบไม่คิดเครื่องหมาย และเลขแบบคิดเครื่องหมายที่มีค่าเป็นบวก รูป (ข) เป็นเลขแบบคิดเครื่องหมายที่มีค่าเป็นลบ 10
รูปที่ 2.4 การแปลงค่าจากเลขฐานสองไปเป็นเลขฐานสิบ โดยบิตข้อมูลอยู่ในรูปแบบ (IIII . FFFF) 11
รูปที่ 2.5 รูปแบบการเก็บค่าแบบจุดตรึง 19
รูปที่ 2.6 ผลต่างช่วงค่าของเลยฐานสอง 6 บิต รูปแบบ Q2 22
รูปที่ 2.7 การคูณกันของเลขฐานสอง รูป (ก) เป็นการคูณเต็มรูปแบบ และรูป (ข) เป็นการคูณเฉพาะบิตที่มีค่าเป็น 1 โดยอาศัยการเลื่อนบิตเข้ามาช่วย ทำให้ขั้นตอนการคูณลดลง 25
รูปที่ 2.8 การคูณ และการหารแบบวิธีทำซ้ำ 26
รูปที่ 2.9 การหาร (ก) การหารเลขฐานสิบ (ข) การหารเลขฐานสอง 26
รูปที่ 2.10 การหารโดยวิธีการลบแบบเลื่อนบิต 27
รูปที่ 3.1 ขั้นตอนการจำลองการคำนวณ 40
รูปที่ 4.1 ตัวอย่างโครงสร้างของระบบควบคุมเวกเตอร์แบบแรงดันโดยอาศัยการควบคุมแยกการเชื่อมร่วมที่มีวงรอบคบคุมกระแส 45
รูปที่ 4.2 แผนภาพบล็อกสมการอันดับที่หนึ่ง 46
รูปที่ 4.3 แผนภาพบล็อคสมการควบคุม PI 47
รูปที่ 4.4 แผนภาพบล็อคส่วนสมการการแปลงแกน 58
รูปที่ 4.5 การแปลงแกน 58

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ห้องปฏิบัติการวิจัยอิเล็กทรอนิกส์กำลัง (Power Electronics Research Laboratory) ภาควิชาวิศวกรรมไฟฟ้า มีโครงงานเกี่ยวกับการควบคุมความเร็วมอเตอร์ที่มีการใช้บอร์ดอิเล็กทรอนิกส์เป็นตัวควบคุม ซึ่งชิปหน่วยประมวลผลที่ใช้มีหลายบริษัท เช่น Hitachi หรือ Texas Instrument เป็นต้น ในที่นี้เราจะสนใจเฉพะชิปของบริษัท Hitachiในรุ่น SH1 [1] เท่านั้น ซึ่งเป็นรุ่นที่มีการใช้งานจริงในอุตสาหกรรมภายในประเทศ เพราะปัจจัยเรื่องของราคา และความสามารถในการสนับสนุนการทำงาน ในการควบคุมความเร็วมอเตอร์นั้นจำเป็นจะต้องเขียนโปรแกรมเพื่อควบคุมการทำงานโดยมีข้อมูลเข้า (Input) ของระบบเป็นค่าที่ได้จากการตรวจจับกระแส และแรงดันของมอเตอร์แล้วแปลงค่าเป็นข้อมูลดิจิตอล หลังจากนั้นหน่วยประมวลผลก็จะทำหน้าที่ในการคำนวณค่าที่ได้โดยอาศัยสมการการควบคุมที่ได้ออกแบบไว้ แล้วจึงส่งค่าข้อมูลออก (Output) เป็นค่าสัญญาณ PWM (Pulse Width Modulation) [2] เพื่อไปใช้ควบคุมความเร็วรอบในการหมุนของมอเตอร์



รูปที่ 1.1 ภาพโดยรวมของระบบควบคุมมอเตอร์

การเขียนโปรแกรมควบคุมมอเตอร์นิยมใช้เลขแบบจุดตรึง (Fixed-point) [3] ในส่วนการคำนวณ เพราะสามารถกำหนดช่วงค่าที่ต้องการได้แน่นอน และสามารถคำนวณค่า

ความคลาดเคลื่อนได้อย่างแม่นยำ โดยที่ไม่ขึ้นกับตัวแปลโปรแกรม (Compiler) และสามารถใช้ได้กับทุกสถาปัตยกรรมของหน่วยประมวลผล รวมถึงช่วยลดจำนวนรอบคำสั่งเครื่อง (Instruction Cycle) ในการคำนวณ ซึ่งถือเป็นข้อดีด้านอื่นที่ไม่เกี่ยวข้องกับการคำนวณโดยตรง แต่เลขแบบจุดตรึงมีข้อจำกัดในเรื่องการเก็บค่าข้อมูล เพราะสามารถเก็บช่วงค่าได้จำกัดขึ้นอยู่กับจำนวนบิตที่ใช้เก็บข้อมูล และในการคำนวณตามสมการบางพจน์ในสมการอาจต้องทำการปรับแต่งค่าเพื่อให้ได้ผลตามที่ต้องการ ทำให้ต้องเสียเวลาปรับแต่งค่าหลายครั้ง และมักจะพบปัญหาว่าผลลัพธ์ที่ได้ไม่เป็นไปตามที่ได้ทำการออกแบบไว้ ทำให้การพัฒนาโปรแกรมมีความล่าช้าเป็นอย่างมาก โดยปัญหาที่พบนั้น สามารถแบ่งออกได้เป็น

1. ปัญหาเนื่องจากแบบจำลองควบคุม (Control Model) [2] ที่ได้ออกแบบไว้นั้นผิด ถือเป็นข้อผิดพลาดในส่วนการออกแบบ
2. โปรแกรมที่เขียนสั่งงานมีข้อผิดพลาด ถือเป็นข้อผิดพลาดในส่วนการเขียนโปรแกรม รวมถึงเครื่องมือที่ใช้ด้วย
3. ปัญหาเนื่องมาจากอุปกรณ์ฮาร์ดแวร์ในระบบทำงานผิดพลาด

ในที่นี้เราจะสนใจเฉพาะปัญหาในข้อที่สอง ที่เกิดจากการเขียนโปรแกรมสั่งงานมีข้อผิดพลาด โดยจากที่ได้มีการสำรวจปัญหากับผู้เขียนโปรแกรม เราได้ข้อสรุปว่าปัญหาที่พบบ่อยได้แก่

1. ปัญหาการล้น (Overflow) และน้อยเกินเก็บ(Underflow) [4] [5] [6]
2. ปัญหาการปัดเศษ (Rounding Off) [4] [5] [6]
3. ปัญหามาตราส่วน (Scaling) เป็นปัญหาในเรื่องของการแปลงหน่วย [6]
4. ปัญหาการเขียนโปรแกรมให้เหมาะกับสถาปัตยกรรมของหน่วยประมวลผล
5. ปัญหาเรื่องของเวลา เพราะว่างานที่ทำเป็นงานประเภททำงานแบบเวลาจริง (Real-time)[7]

ปัญหาที่เกิดขึ้นมักเกิดขึ้นจากส่วนการคำนวณค่าเป็นหลัก ดังนั้นจึงได้มีแนวคิดที่จะนำเสนอวิธีการทดสอบการคำนวณ และการวิเคราะห์ค่าผิดพลาด เพื่อป้องกันและลดการเกิดปัญหาดังกล่าว

1.2 วัตถุประสงค์ของการวิจัย

เสนอวิธีการทดสอบการคำนวณและวิเคราะห์ค่าผิดพลาด ของการโปรแกรมในงานควบคุมมอเตอร์ เพื่อป้องกันและลดข้อผิดพลาดในส่วนการคำนวณ

1.3 ขอบเขตของการวิจัย

1. งานวิจัยนี้จะศึกษาเฉพาะปัญหาดังต่อไปนี้
   - ปัญหาการล้น (Overflow) และน้อยเกินเก็บ (Underflow) ของค่าในเรจิสเตอร์ (Register) เท่านั้นไม่รวมถึงปัญหาการล้นในส่วนของการเรียงทับซ้อน (Stack) หรือหน่วยความจำ (Memory)
   - ปัญหาการปัดเศษ (Rounding off)
   - ปัญหามาตราส่วน (Scaling)
2. ใช้โปรแกรม Borland Delphi 6 กับภาษาซี และภาษาแอสเซมบลี ใช้คอมไพเลอร์ภาษาซีของบริษัท KPIT Infosystems Limited ซึ่งเป็นฟรีแวร์แบบ GNU [8]
3. ใช้โปรแกรม “gcc-sh-run” ของบริษัท KPIT Infosystems Limited เป็นตัวจำลอง (Simulator) การดำเนินการคำนวณ โดยผลที่ได้จากตัวจำลองจะถือเป็นค่าเสมือนที่ได้จากการทำงานของหน่วยประมวลผลจริง โดยในงานวิจัยนี้มุ่งเน้นผลจากหน่วยประมวลผลของบริษัท Hitachi รุ่น SH1 เป็นหลัก
4. ปัญหาที่ใช้เป็นกรณีศึกษาคือปัญหาการโปรแกรมรหัสต้นฉบับส่วนควบคุมแบบพีไอดี ส่วนคำนวณสมการอนุพันธ์อันดับหนึ่ง และส่วนการคำนวณค่าบนแกนอ้างอิงหมุน ของระบบควบคุมมอเตอร์กระแสสลับสามเฟส ที่มีใช้จริงในห้องปฏิบัติการวิจัยอิเล็กทรอนิกส์กำลัง ภาควิศวกรรมไฟฟ้า

1.4 ขั้นตอนและวิธีดำเนินการวิจัย

1. ศึกษาระบบงานเดิมของการเขียนโปรแกรมควบคุมมอเตอร์
2. สำรวจหาปัญหาที่เกิดขึ้นบ่อยในการเขียนโปรแกรมควบคุมมอเตอร์
3. ศึกษาการเขียนโปรแกรมภาษาแอสเซมบลี และภาษาซี เพื่อการควบคุมอุปกรณ์ฮาร์ดแวร์
4. ศึกษาสถาปัตยกรรมของหน่วยประมวลผลของบริษัทฮิตาชิ รุ่น SH1
5. ศึกษาปัญหาที่เกิดในรหัสต้นฉบับ (Source Code) ของโปรแกรมควบคุมมอเตอร์ที่ใช้อยู่จริง
6. ศึกษาวิธีการทดสอบการคำนวณ และการวิเคราะห์ค่าผิดพลาด
7. ออกแบบ และพัฒนาโปรแกรมประยุกต์เพื่อใช้ทดสอบกับปัญหาจริง
8. ทดสอบ ปรับปรุง และแก้ไขวิธีการป้องกันปัญหาที่เกิดในรหัสต้นฉบับ
9. สรุปผลงานวิจัย รวมถึงข้อเสนอแนะ และจัดทำรายงานวิทยานิพนธ์

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1. ได้วิธีการทดสอบและการวิเคราะห์ค่าผิดพลาดเพื่อป้องกัน และลดปัญหาในการโปรแกรมงานควบคุมมอเตอร์
2. วิธีการที่ได้สามารถประยุกต์ใช้กับปัญหาในการคำนวณเชิงวิทยาศาสตร์ได้
3. ลดปัญหาในระบบงานควบคุมมอเตอร์ ทำให้สามารถพัฒนางานได้เร็วขึ้น

1.6 ผลงานที่ตีพิมพ์จากงานวิจัย

งานวิจัยในวิทยานิพนธ์ฉบับนี้ได้รับการตีพิมพ์และนำเสนอในงานประชุมวิชาการทางวิศวกรรมไฟฟ้า ครั้งที่ 26 (EECON-26) วันที่ 6-7 พฤศจิกายน พ.ศ.2546 ในบทความเรื่อง “การวิเคราะห์ค่าผิดพลาดของเลขคณิตแบบช่วงเพื่อการโปรแกรมระบบฝังตัว” โดย เฉลิมทรัพย์ สังขวิจิตร สมบูรณ์ แสงวงค์วาณิชย์ และ ประภาส จงสถิตย์วัฒนา

1.7 โครงสร้างวิทยานิพนธ์

วิทยานิพนธ์นี้จะแบ่งออกเป็น 5 บท บทที่ 1 เป็นบทนำ (บทปัจจุบัน) บทที่ 2 จะกล่าวถึงทฤษฎีและงานวิจัยที่เกี่ยวข้อง ส่วนบทที่ 3 จะกล่าวถึงขั้นตอนวิธีการทดสอบความถูกต้องและการวิเคราะห์ค่าผิดพลาด ส่วนบทที่ 4 จะกล่าวถึงรายละเอียดขั้นตอน และเครื่องมือต่าง ๆ ในการทดลอง พร้อมทั้งผลการทดลองการวิเคราะห์ค่าผิดพลาดในกรณีศึกษาตัวอย่าง และบทสุดท้าย บทที่ 5 เป็นการสรุปผลที่ได้จากการทดลอง และให้ข้อเสนอแนะเพื่อปรับปรุงและพัฒนาต่อไป

บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 ทฤษฎีที่เกี่ยวข้อง

2.1.1 การประมวลผลเลขฐานสอง (Process Binary Number) [9]

ในระบบดิจิทัล (Digital System) จะมีการประมวลผล และการเก็บข้อมูลเป็นบิต (Bit) ซึ่งเป็นเลขฐานสอง (Binary Numbers) ดังนั้นเราจำเป็นที่จะต้องเรียนรู้ และทำความเข้าใจกับระบบเลขฐานสอง เพื่อให้สามารถเข้าใจระบบการทำงานของระบบดิจิทัลได้ดีขึ้น

2.1.1.1 ระบบเลขฐานสอง (Binary Number System)

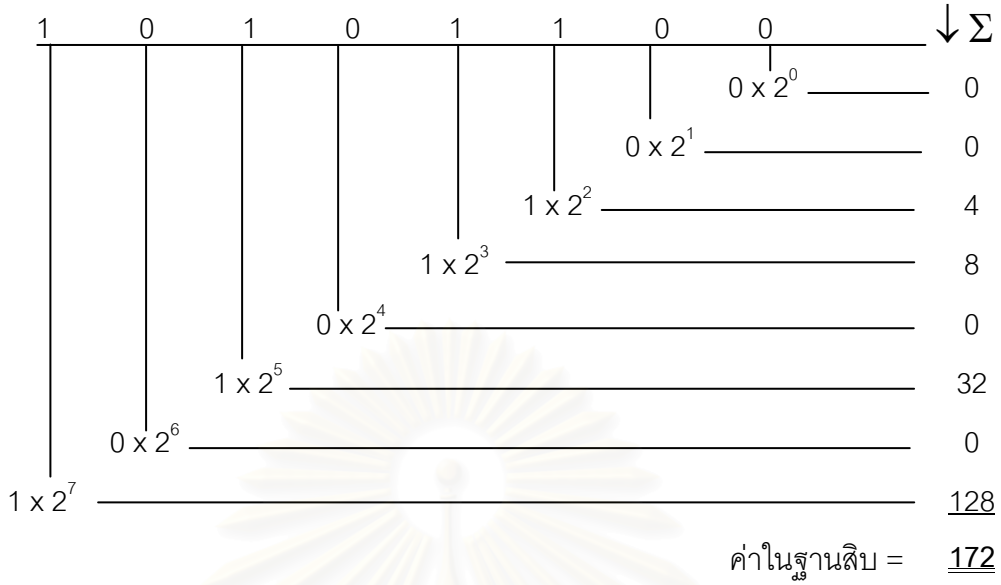
โดยส่วนใหญ่ในชีวิตประจำวันเราจะเคยชินกับการใช้เลขฐานสิบ (0 ถึง 9) แต่ในระบบดิจิทัลนั้นเราจะใช้เลขฐานสองเป็นหลัก องค์ประกอบของเลขฐานสองมีเพียงตัวเลข 0 กับ 1 เท่านั้น โดยตัวเลข 1 ตัว จะแทนข้อมูลขนาด 1 บิต ดังตัวอย่างต่อไปนี้

$$10101100_2$$

การอ่านค่าจะอ่านจากขวาไปซ้ายจากตัวอย่างค่าหลักแรกจะเป็นเลข 0 (ขวาสุด) เป็นบิตนัยสำคัญน้อยที่สุด (LSB : Least Significant Bit) และค่าหลักมากที่สุดจะเป็นเลข 1 (ซ้ายสุด) เป็นบิตที่มีนัยสำคัญมากที่สุด (MSB : Most Significant Bit) โดยเราสามารถแปลงค่ากลับไปเป็นเลขฐานสิบเพื่อง่ายต่อการเข้าใจ โดยหลักแรก (ขวาสุด) จะเป็นหลักที่ 0 และเพิ่มค่าขึ้นเรื่อยๆ ทีละหนึ่ง ไปทางซ้ายมือ การแทนค่าของตัวเลขแต่ละหลักเป็นดังสมการดังต่อไปนี้

$$\text{Column Value} = \text{bit value (0 or 1)} \times 2^{\text{column no}}$$

จากสมการเราจะได้ค่าของตัวเลขแต่ละหลักอยู่ในรูปเลขฐานสิบ การหาค่ารวมของเลขทั้งหมดจะต้องนำค่าของเลขแต่ละหลักที่แปลงเป็นฐานสิบแล้วมาบวกเข้าด้วยกัน แสดงดังรูปที่ 2.1



รูปที่ 2.1 การแปลงค่าจากเลขฐานสองไปเป็นเลขฐานสิบ

จำนวนบิต (จำนวนตัวเลข) ในเลขฐานสองนั้นบอกถึงความสามารถในการเก็บ หรือแสดงค่าที่แตกต่างกันได้ ดังสมการต่อไปนี้

$$\text{Number of Difference Values} = 2^{\text{number of bits in the number}}$$

ตัวอย่างเช่น มีจำนวนบิตทั้งหมด 4 บิต สามารถมีค่าที่แตกต่างกันได้ถึง 2^4 หรือ 16 ค่า สามารถอธิบายได้โดยลองแทนค่าที่เป็นไปได้ทั้งหมดให้กับเลขทั้งสี่หลัก ดังตารางที่ 2.1

นอกจากนี้จำนวนบิตยังบอกถึงช่วงค่าเทียบเท่าที่เป็นไปได้ในเลขฐานสิบ โดยเริ่มจากค่า 0 เสมอ ดังสมการต่อไปนี้

$$\text{Possible Values Range in Decimal} = 0 \text{ to } 2^{\text{number of bits in the number}} - 1$$

ตัวอย่าง จากตารางที่ 2.1 เราจะได้ว่า เลขฐานสองจำนวน 4 บิตสามารถแทนค่าเลขฐานสิบได้ในช่วงค่าตั้งแต่ 0 ถึง 15 เป็นต้น

ตารางที่ 2.1 การแทนค่าที่เป็นไปได้ทั้งหมดสำหรับเลขฐานสองจำนวน 4 บิต

เลขฐานสอง 4 บิต		เลขฐานสิบ		เลขฐานสิบหก
0000	⇔	0	⇔	0
0001	⇔	1	⇔	1
0010	⇔	2	⇔	2
0011	⇔	3	⇔	3
0100	⇔	4	⇔	4
0101	⇔	5	⇔	5
0110	⇔	6	⇔	6
0111	⇔	7	⇔	7
1000	⇔	8	⇔	8
1001	⇔	9	⇔	9
1010	⇔	10	⇔	A
1011	⇔	11	⇔	B
1100	⇔	12	⇔	C
1101	⇔	13	⇔	D
1110	⇔	14	⇔	E
1111	⇔	15	⇔	F

2.1.1.2 การจัดรูปแบบเลขฐานสอง (Formatting Binary Numbers)

การคำนวณในระบบเลขฐานสองมีข้อจำกัดในเรื่องของจำนวนบิตที่ใช้เก็บข้อมูล ทำให้การคำนวณในบางรูปแบบอาจไม่สามารถเก็บข้อมูลที่ถูกต้องตามที่ต้องการได้ จึงมีการคิดวิธีในการเก็บข้อมูลค่าตัวเลขให้เพียงพอต่อการคำนวณ โดยเราสามารถแบ่งออกได้เป็น 3 รูปแบบดังนี้

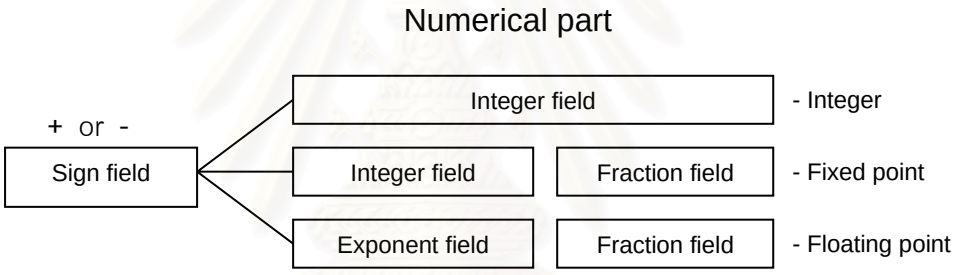
1. เลขจำนวนเต็ม (Integer Numbers) จะใช้จำนวนบิตทั้งหมดเก็บค่าเลขจำนวนเต็ม ตัวอย่างเช่น ค่า 5, 135, 428 เป็นต้น
2. เลขจุดตรึง (Fixed-Point Numbers) จะแบ่งเป็น 2 ส่วน ได้แก่ ส่วนเลขจำนวนเต็ม และส่วนเลขเศษ (Fraction Numbers) ตัวอย่างเช่น 3.9375, 43.5 เป็นต้น

3. เลขอิงดรรชนี (Floating-point Numbers) จะแบ่งออกเป็น 2 ส่วน ได้แก่ ส่วนที่เป็นเลขเศษ (Fractional Field) และส่วนที่เป็นเลขชี้กำลัง (Exponential Field) ตัวอย่างเช่น 0.25×10^{3}, 0.8679×10^{12} เป็นต้น

นอกจากนี้รูปแบบการคำนวณยังแบ่งตามเครื่องหมาย โดยแบ่งออกเป็น 2 รูปแบบ ได้แก่

1. เลขไม่คิดเครื่องหมาย (Unsigned Numbers) ตัวเลขทั้งหมดจะเป็นค่าบวกเพียงอย่างเดียวเท่านั้น และจำนวนบิตทั้งหมดจะถูกใช้เก็บค่าตัวเลข
2. เลขคิดเครื่องหมาย (Signed Numbers) ตัวเลขจะสามารถเป็นได้ทั้งค่าบวก และลบ แต่จะต้องมีการใช้บางบิตเป็นตัวสัญลักษณ์บอกว่ามีค่าเป็นบวก หรือลบ ส่วนจำนวนบิตที่เหลือจะใช้เก็บค่าตัวเลขตามปกติ

เพื่อให้ง่ายต่อการมองภาพรวมของรูปแบบการเก็บค่าตัวเลขจะแสดงดังรูป ต่อไปนี้



รูปที่ 2.2 การเก็บค่าตัวเลขในรูปแบบของเลขฐานสอง

ในการคำนวณนั้นควรเลือกรูปแบบการเก็บค่าตัวเลขให้เหมาะสม เพื่อลดปัญหาที่อาจจะเกิดขึ้นจากการคำนวณ ซึ่งแต่ละรูปแบบนั้นจะมีข้อได้เปรียบและเสียเปรียบแตกต่างกัน ซึ่งจะมีการอธิบายถึงในส่วนถัดไป

2.1.1.3 <u>เลขคิดเครื่องหมาย และไม่คิดเครื่องหมาย</u>

จากตารางที่ 2.1 เป็นเลขฐานสอง 4 บิต แบบไม่คิดเครื่องหมาย เมื่อเราต้องการทำให้เป็นเลขแบบคิดเครื่องหมาย เราจะต้องยอมเสียจำนวนบิตอย่างน้อย 1 บิต เพื่อกำหนดให้เป็นบิตเครื่องหมาย (Signed Bit) โดยเราจะกำหนดให้บิตนัยสำคัญมากสุดเป็นบิตเครื่องหมายโดยที่เมื่อไหร่บิตนี้มีค่าเป็น 1 หมายความว่ามีค่าเป็นลบ ตัวอย่างของเลขคิดเครื่องหมายแสดงดังตารางที่ 2.2

ตารางที่ 2.2 ค่าเลขแบบคิดเครื่องหมาย และเลขแบบไม่คิดเครื่องหมาย

เลขฐานสอง 4 บิต		เลขฐานสิบแบบไม่คิดเครื่องหมาย (Unsigned Values)		เลขฐานสิบแบบคิดเครื่องหมาย (Signed Values)
0000	⇔	0	⇔	0
0001	⇔	1	⇔	1
0010	⇔	2	⇔	2
0011	⇔	3	⇔	3
0100	⇔	4	⇔	4
0101	⇔	5	⇔	5
0110	⇔	6	⇔	6
0111	⇔	7	⇔	7
1000	⇔	8	⇔	-8
1001	⇔	9	⇔	-7
1010	⇔	10	⇔	-6
1011	⇔	11	⇔	-5
1100	⇔	12	⇔	-4
1101	⇔	13	⇔	-3
1110	⇔	14	⇔	-2
1111	⇔	15	⇔	-1

จากตารางจะเห็นได้ว่าการหาค่าเลขแบบคิดเครื่องหมายจะมีวิธีการหาต่างจากเลขแบบไม่คิดเครื่องหมาย การแปลงจากค่าบวกเป็นค่าลบทำได้โดยใช้วิธีส่วนเติมเต็มของสอง (2's Complement) ตามขั้นตอนต่อไปนี้

1. เปลี่ยนค่าของบิตแต่ละหลักให้เป็นค่าตรงข้าม เป็นการทำส่วนเติมเต็มของหนึ่ง (1's Complement)
2. นำค่าที่ได้จากข้อ 1. มาบวกด้วยค่า 1

ตัวอย่างของการทำวิธีส่วนเติมเต็มของสอง

บิตตัวอย่างจำนวน 4 บิต	0110	= +6
1. เปลี่ยนค่าแต่ละบิตได้เป็น	1001	= $\bar{6}$
2. บวกด้วยค่า 1	+ 1	
	1010	= -6

จะเห็นว่าจากค่า +6 เมื่อผ่านการทำส่วนเติมเต็มของสองแล้วจะได้ค่าเป็น -6

การขยายจำนวนบิตจะต่างกันระหว่างเลขคิดเครื่องหมาย กับเลขไม่คิดเครื่องหมาย โดยที่เลขไม่คิดเครื่องหมายจะใช้การเติม 0 ซ้ายมือ จนกว่าจะครบตามจำนวนบิตที่กำหนด ส่วนเลขแบบคิดเครื่องหมายนั้นจะใช้ การเติมค่า 0 หรือ 1 จนครบตามจำนวนบิตที่กำหนด โดยเลือกจากค่าบิตนัยสำคัญมากที่สุด ลองดูรูปตัวอย่างต่อไปนี้เพื่อให้เข้าใจมากขึ้น

ตัวอย่าง ค่าเริ่มต้น	+6	เลขฐานสอง 4 บิต →	0110
	+6	เลขฐานสอง 8 บิต →	0000 0110

(ก)

ตัวอย่าง ค่าเริ่มต้น	-6	เลขฐานสอง 4 บิต →	1010 (ส่วนเติมเต็มของ 2)
	-6	เลขฐานสอง 8 บิต →	1111 1010

(ข)

รูปที่ 2.3 การขยายจำนวนบิต รูป (ก) เป็นเลขแบบไม่คิดเครื่องหมาย และเลขแบบคิดเครื่องหมายที่มีค่าเป็นบวก รูป (ข) เป็นเลขแบบคิดเครื่องหมายที่มีค่าเป็นลบ

จากตัวอย่างจะเห็นว่าจะมีปัญหาเฉพาะกรณีที่เป็นเลขแบบคิดเครื่องหมายที่มีค่าเป็นลบเท่านั้น และรูปแบบของการแทนค่าก็จะเปลี่ยนไปดังนี้

จากเดิม 4 บิต รูปแบบ I I I I ค่า -6 เป็น 1010 (ส่วนเติมเต็มของ 2)

หลังการขยายเป็น 8 บิต รูปแบบจะเปลี่ยนเป็น SSSS I I I I ค่า -6 เป็น 1111 1010

จะเห็นว่าเมื่อมีการนำค่า 1111 1010 แปลงกลับไปเป็นเลขฐานสิบทำได้ดังนี้

1. นำไปทำส่วนเติมเต็มของสอง $-(\overline{11111010} + 1) = -(00000101 + 1) = -(110)$

2. แปลงค่าเป็นฐานสิบจาก -(110) $\xrightarrow{\text{เลขฐานสิบ}}$ -(4+2) = -6

โดยสัญลักษณ์ I แทนบิตตัวเลข และสัญลักษณ์ S แทนบิตเครื่องหมาย

2.1.1.4 <u>การหาค่าแบบจุดตรึง (Evaluating Fixed-Point Numbers)</u>

รูปแบบของเลขจุดตรึงจะแบ่งออกเป็น 2 ส่วน ได้แก่ ส่วนที่เป็นเลขจำนวนเต็ม และส่วนที่เป็นเลขเศษ ลองพิจารณาตัวอย่างต่อไปนี้

รูปแบบข้อมูล 16 บิต จำนวนเต็ม 8 บิต เศษ 8 บิต IIII IIII FFFF FFFF

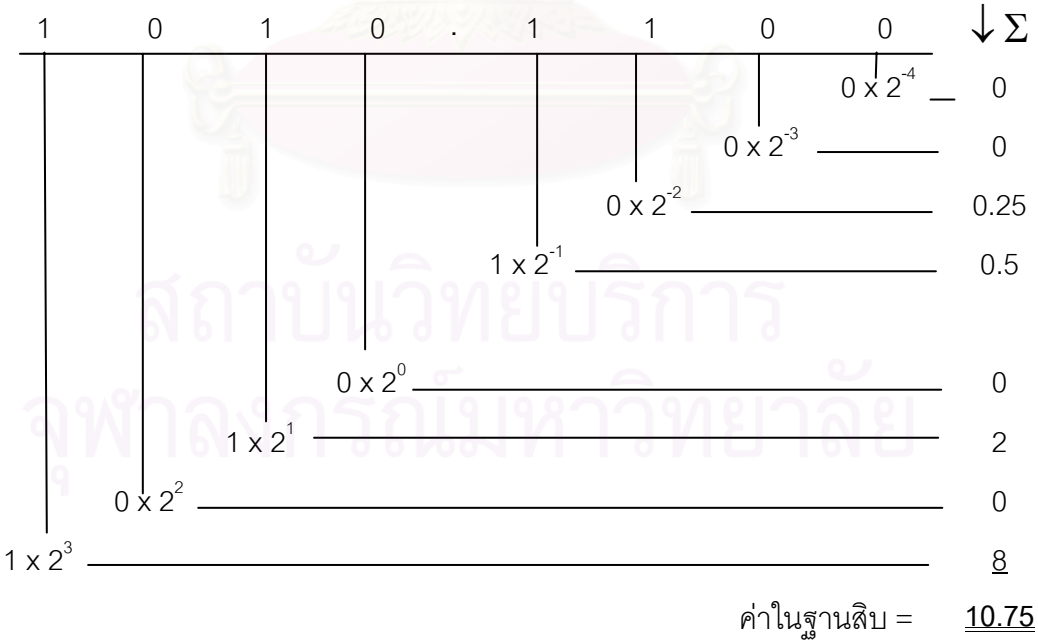
โดยสัญลักษณ์ I แทนบิตเลขจำนวนเต็ม และสัญลักษณ์ F แทนบิตเลขเศษ

2.1.1.5 <u>การคำนวณเลขเศษ</u>

การอ่านค่าเลขเศษนั้นเราอ่านค่ากลับกันกับการอ่านค่าเลขจำนวนเต็มโดยอ่านจากซ้ายมือไปขวามือ เริ่มจากบิตที่เป็นเศษตัวแรกจากซ้ายมือถือเป็นหลักที่ 1 และบิตถัดมาจะมีค่าหลักเพิ่มขึ้นทีละ 1 การแทนค่าของตัวเลขเศษแต่ละหลักเป็นดังสมการดังต่อไปนี้

$$\text{Column Value} = \text{bit value (0 or 1)} \times 2^{-\text{column no}}$$

จากสมการเราจะได้ค่าของตัวเลขแต่ละหลักอยู่ในรูปเลขฐานสิบ การหาค่ารวมของเลขทั้งหมดจะต้องนำค่าของเลขแต่ละหลักที่แปลงเป็นฐานสิบแล้วมาบวกเข้าด้วยกัน แสดงดังรูปที่ 2.4



รูปที่ 2.4 การแปลงค่าจากเลขฐานสองไปเป็นเลขฐานสิบ โดยบิตข้อมูลอยู่ในรูปแบบ (IIII . FFFF)

ตัวอย่างค่าสูงสุดที่เป็นไปได้ สำหรับเลขฐานสอง 8 บิต

ค่าสูงสุดประจำหลักเลขฐานสอง 8 บิต I I I I . F F F F

ค่าสูงสุดในเลขฐานสิบ 8 4 2 1 $\frac{1}{2}$ $\frac{1}{4}$ $\frac{1}{8}$ $\frac{1}{16}$

โดยสัญลักษณ์ I แทนบิตเลขจำนวนเต็ม และสัญลักษณ์ F แทนบิตเลขเศษ

2.1.1.6 <u>การหาค่าความเที่ยง (Precision)</u>

จำนวนบิตของค่าเศษเป็นตัวบ่งชี้ถึงค่าความเที่ยงสำหรับการแทนค่า โดยมีสูตรการคำนวณดังนี้

$$\text{Precision of a Fixed-Point number} = \frac{1}{2^{Quantity\ \ of\ \ bits\ in\ the\ \ fraction\ \ field}}$$

ตัวอย่าง การหาค่าความเที่ยงของตัวเลข 5 บิต

$$\frac{1}{2^5} = \frac{1}{32}$$

2.1.1.7 <u>การหาค่าความทน (Tolerance)</u>

จำนวนบิตของค่าเศษนอกจากบอกถึงค่าความเที่ยงแล้วยังบอกถึงค่าความทนได้ โดยค่าความผิดพลาดมากที่สุดที่เป็นไปได้จะต้องไม่เกินค่าความทน ดังสมการต่อไปนี้

$$\text{Tolerance of a Fixed-Point number} = \pm\frac{1}{2^{Quantity\ of\ bits\ in\ the\ \ fraction\ \ field\ +1}}$$

ตัวอย่าง การหาค่าความทนของตัวเลข 5 บิต

$$\pm\frac{1}{2^{5+1}} = \pm\frac{1}{64}$$

เราลองมาพิจารณาตัวอย่างต่อไป สมมติตัวเลข 8 บิต มีค่า $0000\ 0101_2 = 5_{10}$ เมื่อกำหนดให้อยู่ในรูปแบบ I I I I . FFFF จะได้

$$0000\ .\ 0101 \quad \Rightarrow \quad 0+0+0+0+\frac{0}{2}+\frac{1}{4}+\frac{0}{8}+\frac{1}{16} = \frac{5}{16}$$

เมื่อกำหนดให้อยู่ในรูปแบบ I I I I I . FFF จะได้

$$00000\ .\ 101 \quad \Rightarrow \quad 0+0+0+0+0+\frac{1}{2}+\frac{0}{4}+\frac{1}{8} = \frac{5}{8}$$

จากทั้งสองตัวอย่างที่ยกมาให้พิจารณานี้จะเห็นว่าค่าตัวเลขขนาด 8 บิต เมื่อมีการเลือกการจัดรูปแบบที่แตกต่างกันออกไปจะทำให้ค่าผลลัพธ์ที่ได้แตกต่างกันด้วย จากการสังเกตเราจะเห็นว่าไม่ว่าจะเลือกการจัดรูปแบบเป็นแบบใดก็ตามค่าเศษก็ยังคงเดิม เปลี่ยนแต่เพียงค่าส่วนตัวหารเท่านั้น โดยส่วนตัวหารนี้จะมีค่า $2^{\text{Quantity of bits in the fractional field}}$ เราจะเรียกการจัดรูปแบบแบบนี้ว่าการจัดรูปแบบแบบจุดตรึง หรือการจัดรูปแบบควอนไทเซชัน (Q-format : Quantization Format) ซึ่งกำหนดโดยจำนวนเลขเศษเป็นหลัก

2.1.2 <u>การจัดรูปแบบคิว (Q-format) [10]</u>

การจัดรูปแบบคิวเป็นการบอกจำนวนการปรับมาตรา (Scaling) ที่ประยุกต์ใช้กับเลขแบบจุดตรึงก่อนที่จะเก็บค่าในหน่วยความจำ โดยมีรูปแบบ

Qm.n โดย Q หมายถึง เป็นการบอกว่าเป็นการจัดรูปแบบคิว
m หมายถึง จำนวนบิตที่แทนจำนวนเต็ม
n หมายถึง จำนวนบิตที่แทนจำนวนเศษ

ส่วนใหญ่นิยมเขียนในรูปแบบของ Qn ตัวอย่างในการจัดรูปแบบคิว ใช้ค่าเริ่มต้นเป็น 0000 1001

รูปแบบ Q0 0000 1001 $\xrightarrow{\text{เลขฐานสิบ}}$ 9

รูปแบบ Q1 0000 100. 1 $\xrightarrow{\text{เลขฐานสิบ}}$ $\frac{9}{2} = 4\frac{1}{2} = 4.5$

รูปแบบ Q2 0000 10. 01 $\xrightarrow{\text{เลขฐานสิบ}}$ $\frac{9}{4} = 2\frac{1}{4} = 2.25$

รูปแบบ Q3 0000 1. 001 $\xrightarrow{\text{เลขฐานสิบ}}$ $\frac{9}{8} = 1\frac{1}{8} = 1.125$

รูปแบบ Q4 0000 . 1001 $\xrightarrow{\text{เลขฐานสิบ}}$ $\frac{9}{16} = .5625$

จากตัวอย่างจะเห็นความสัมพันธ์ของการจัดรูปแบบคิว เป็นไปตามสมการต่อไปนี้

$$Qn = 2 \times Q(n+1)$$

จากตัวอย่างก่อนหน้า เช่น Q0 = 2 x Q1, Q3 = 2 x Q4 เป็นต้น

วิธีการแปลงค่าจากเลขฐานสิบไปเป็นเลขฐานสองโดยใช้การจัดรูปแบบคิว เป็นดังสมการต่อไปนี้

$$\text{Binary Number} = \text{Decimal Number} \times 2^{\text{number of Q}}$$

ตัวอย่างของการแปลงจากเลขฐานสิบเป็นเลขฐานสองโดยใช้การจัดรูปแบบคิว ดังตารางที่ 2.3

ตารางที่ 2.3 การแปลงค่าจากเลขฐานสิบเป็นเลขฐานสองโดยใช้การจัดรูปแบบคิว

เลขฐานสิบ	รูปแบบคิว	วิธีการคำนวณ			เลขจุดตรึง 16 บิต
2.390625	Q12 : 4096	2.390625 x 4096	= 9792	→	0010 0110 0100 0000
6.78125	Q10 : 1024	6.78125 x 1024	= 6944	→	0001 0110 0010 0000
6.78125	Q8 : 256	6.78125 x 256	= 1736	→	0000 0110 1100 1000

ตัวอย่างของการแปลงจากเลขฐานสองโดยใช้การจัดรูปแบบคิวเป็นเลขฐานสิบ ดังตารางที่ 2.4

ตารางที่ 2.4 การแปลงจากเลขฐานสองโดยใช้การจัดรูปแบบคิวเป็นเลขฐานสิบ

รูปแบบคิว			วิธีการคำนวณ	เลขฐานสิบ
0010 0110 0100 0000	→	9792	Q12 : $\frac{9792}{4096}$	= 2.390625
0001 0110 0010 0000	→	6944	Q10 : $\frac{6944}{1024}$	= 6.78125
0000 0110 1100 1000	→	1736	Q8 : $\frac{1736}{256}$	= 6.78125

จากตัวอย่างตารางที่ 2.3 จะเห็นว่าตัวเลขคูณแล้วจะได้จำนวนเต็ม และจากตัวอย่างตารางที่ 2.4 ตัวเลขที่หารแล้วก็จะได้ผลลัพธ์เป็นเลขจำนวนเต็ม ซึ่งเป็นตัวอย่างเลขกรณีพิเศษซึ่งเมื่อดำเนินการคูณหรือหารแล้วได้ผลลัพธ์เป็นเลขจำนวนเต็มเพื่อให้ง่ายต่อการเข้าใจ แต่ในกรณีทั่วไปผลลัพธ์มีโอกาสเป็นไปได้ทั้งเลขจำนวนเต็ม และเลขทศนิยม เราลองมาพิจารณาตัวอย่างต่อไปนี้

$$1.3 \times 2^5 = 1.3 \times 32 = 41.6$$

จากตัวอย่างจะพบว่าเกิดปัญหาในเรื่องของการเก็บค่าเพราะว่าเราไม่สามารถเก็บค่าเลข 41.6 ได้ จึงต้องเก็บในรูปแบบเลขจำนวนเต็มทำให้เกิดปัญหาการปัดเศษ โดยถ้าปัดเศษลงเราจะได้ค่า 41 หรือปัดเศษขึ้นจะได้ค่า 42 ซึ่งทำให้ค่าที่ได้นั้นไม่ตรงกับค่าจริง

กรณีปัดเศษลงจาก 41.6 เป็น 41 จะได้

$$41.6 - 0.6 = 41 \longrightarrow \textbf{0010 1001}$$

เมื่อแปลงค่ากลับให้อยู่ในรูปฐานสิบจะได้

$$\frac{41}{32} = 1.28125 \text{ (จากค่าเริ่มต้นคือ 1.3)}$$

กรณีปัดเศษขึ้นจาก 41.6 เป็น 42 จะได้

$$41.6 + 0.6 = 42 \longrightarrow \textbf{0010 1010}$$

เมื่อแปลงค่ากลับให้อยู่ในรูปฐานสิบจะได้

$$\frac{42}{32} = 1.3125 \text{ (จากค่าเริ่มต้นคือ 1.3)}$$

จะเห็นว่ามีค่าความผิดพลาดเกิดขึ้น เราจะเรียกค่าผิดพลาดที่เกิดขึ้นจากการปัดเศษว่า ค่าผิดพลาดควอนไทเซชัน (Quantization Error) โดยเราสามารถหาค่าในรูปแบบของเปอร์เซ็นต์ความผิดพลาดควอนไทเซชัน ได้จากสมการต่อไปนี้

$$\text{Percentage Quantization Error} = \frac{Rounding\ Error}{Rounded\ Value} \times 100$$

ตัวอย่างการคำนวณค่าเปอร์เซ็นต์ความผิดพลาดควอนไทเซชัน จากตัวอย่างก่อน

กรณีปัดเศษลงจาก 41.6 เป็น 41 จะได้

$$\text{เปอร์เซ็นต์ความผิดพลาดควอนไทเซชัน} = \frac{-0.6}{41.6} \times 100 \approx -1.4\%$$

ค่าความผิดพลาดเป็นลบเพราะว่าเราตัดค่า 0.6 ทิ้งไป

กรณีปัดเศษขึ้นจาก 41.6 เป็น 42 จะได้

$$\text{เปอร์เซ็นต์ความผิดพลาดควอนไทเซชัน} = \frac{0.4}{41.6} \times 100 \approx +0.96\%$$

นอกจากค่าเปอร์เซ็นต์ความผิดพลาดควอนไทเซชันแล้ว เรายังสามารถหาค่าความแม่น (Percentage Accuracy) ใช้กรณีที่มีการปัดเศษลง (ตัวตั้งมีค่าน้อยกว่าตัวหาร) โดยมีรูปแบบสมการดังนี้

$$\text{Percentage Accuracy} = \frac{Rounded\ value}{Value\ we\ wanted\ to\ rounding} \times 100$$

ตัวอย่างการคำนวณหาค่าเปอร์เซ็นต์ความแม่น จากตัวอย่างก่อน

กรณีปัดเศษลงจาก 41.6 เป็น 41 จะได้

$$\text{เปอร์เซ็นความแม่น} = \frac{41}{41.6} \times 100 \approx 98.6\%$$

นอกจากการคำนวณค่าตัวเลขที่ได้กล่าวมาแล้ว ยังมีเรื่องของการคำนวณหาค่าฟังก์ชันตรีโกณมิติ เช่น ฟังก์ชันค่าโคไซน์ (Cos : Cosine) เป็นต้น ซึ่งพบได้บ่อยในการคำนวณเชิงวิทยาศาสตร์ (Scientific Calculation) การหาค่าฟังก์ชันตรีโกณมิติมี 2 วิธีที่นิยมใช้กัน ได้แก่ วิธีแรกใช้การเก็บข้อมูลของค่าฟังก์ชันตรีโกณในรูปแบบตารางโดยเก็บค่าเป็นช่วง เมื่อต้องการค่าระหว่างช่วงที่เก็บก็ต้องทำการประมาณค่าในช่วง (Interpolation) ซึ่งความถูกต้องของค่าที่ได้จะขึ้นอยู่กับความถี่ของช่วงที่เก็บข้อมูลกับจำนวนทศนิยมที่ใช้เก็บค่าข้อมูล ส่วนวิธีที่สองเป็นการประมาณค่าจากสมการอนุกรม ซึ่งความถูกต้องจะขึ้นกับจำนวนพจน์ที่ใช้ในการคำนวณกับจำนวนทศนิยมที่ใช้เก็บค่าข้อมูล แต่โดยส่วนใหญ่นิยมใช้การเก็บข้อมูลวิธีแรกคือ แบบใช้ตารางเก็บค่าเป็นช่วง ซึ่งมีข้อเสียคือเปลืองเนื้อที่หน่วยความจำ แต่มีข้อดีที่สำคัญมากคือ ทำให้คำนวณหาค่าที่ต้องการได้รวดเร็ว ส่วนเรื่องความผิดพลาด หรือความแม่นนั้นขึ้นอยู่กับจำนวนบิตที่ใช้เก็บข้อมูลซึ่งมีจำนวนจำกัด

ตัวอย่างการเก็บค่าฟังก์ชันโคไซน์ในรูปแบบตาราง โดยแบ่งค่าที่เก็บออกเป็น 64 ช่วง

$$\frac{360^{o}}{64} = 5.625^{o}$$

ค่าฟังก์ชันโคไซน์มีค่าระหว่าง -1 ถึง +1 เราจึงต้องเก็บโดยใช้การจัดรูปแบบคิวแบบคิดเครื่องหมาย โดยเก็บข้อมูลแบบ 8 บิต ในรูปแบบ Q6

S I F F F F F F

โดยใช้สัญลักษณ์ S แทนบิตเครื่องหมาย

จากที่ได้กล่าวมาแล้วเราสามารถหาค่าความเที่ยงจากรูปแบบ Q6 ได้เป็น

$$\frac{1}{2^{6}} = \frac{1}{64} = 0.015625$$

และสามารถหาค่าความทนได้เป็น

$$\frac{1}{2^{7}} = \frac{1}{128} = \pm 0.0078125$$

ตารางที่ 2.5 ตัวอย่างการเก็บค่าฟังก์ชันโคไซน์ด้วยการจัดรูปแบบ Q6 โดยค่าที่ใช้มีการปัดเศษขึ้นเมื่อเลขหลังหลักสุดท้ายที่ต้องการมีค่า ≥ 5 ถ้าน้อยกว่าจะถูกปัดลง

ค่าอุดมคติ	ค่าประมาณ	รูปแบบ Q6 SIFF FFFF	ค่าที่เก็บได้	ค่าผิดพลาด	เปอร์เซ็นต์ค่าผิดพลาด
cos(73.125°)	= +0.2903	0001 0011	+0.2969	+0.0066	+2.3
cos(78.75°)	= +0.1951	0000 1100	+0.1875	-0.0076	-3.9
cos(84.375°)	= +0.0980	0000 0110	+0.0938	-0.0042	-4.3
cos(90°)	= +0.0000	0000 0000	+0.0000	0.0000	0.00
cos(95.625°)	= -0.0980	1111 1010	-0.0938	+0.0042	+4.3
cos(101.25°)	= -0.1951	1111 0100	-0.1875	+0.0076	+3.9
cos(106.875°)	= -0.2903	1110 1101	-0.2969	-0.0066	-2.3
cos(112.5°)	= -0.3827	1110 1000	-0.3750	+0.0077	+2.0

2.1.2.1 การหาค่าพิสัยของเลขแบบจุดตรึง (Value Range of Fixed-Point Numbers)

จำนวนบิตที่ใช้ในระบบดิจิทัลนั้นส่วนใหญ่มีค่าตั้งแต่ 8 บิต จนถึง 64 บิต เราสามารถหาค่าพิสัยได้จากจำนวนบิตที่ใช้เก็บข้อมูล แสดงตัวอย่างดังตารางที่ 2.6 โดยจะเห็นได้ว่าค่าพิสัยของเลขแบบจุดตรึงขึ้นอยู่กับจำนวนบิตที่เป็นค่าจำนวนเต็ม และค่าความเที่ยงขึ้นอยู่กับจำนวนบิตที่เป็นค่าเศษ

ตัวอย่างการหาค่าพิสัยของเลขแบบ 24 บิต Q12 แบบคิดเครื่องหมาย โดยกำหนดรูปแบบ

S IIIIIIIIIII . FFFFFFFFFFFF

ตารางที่ 2.6 ตารางค่าพิสัยสูงสุดที่เป็นไปได้ตามจำนวนบิตทั้งเลขจำนวนเต็ม และเลขเศษ

จำนวนบิต	ค่าจำนวนเต็มมากสุด	ค่าเศษมากสุดของบิตขวาสุด
1	1	0.5
2	3	0.25
3	7	0.125
4	15	0.0625
5	31	0.03125
6	63	0.015625
7	127	0.0078125
8	255	0.0039625
9	511	0.001953125
10	1,023	0.0009765625
11	2,047	0.00048828125
12	4,095	0.000244140625
13	8,191	0.0001220703125
14	16,383	0.00006103515625
15	32,767	0.000030517578125
16	65,535	0.0000152587890625
17	131,071	0.00000762939453125
18	262,143	0.000003814697265625
19	524,287	0.0000019073486328125
20	1,048,577	0.00000095367431640625
21	2,097,151	0.000000476837158203125
22	4,194,303	0.000000238418579101562 5
23	8,388,607	0.000000119209289550781 25
24	16,777,215	0.000000059604644775390625
25	33,554,431	0.000000029802322387695312 5
26	67,108,863	0.000000014901161193847656 25
27	134,217,727	0.000000007450580596923828125
28	268,435,455	0.000000003725290298461914062 5
29	536,870,911	0.000000001862645149230957031 25
30	1,073,741,823	0.000000000931322574615478515625
31	2,147,483,647	0.000000000465661287307739257812 5
32	4,294,967,295	0.000000000232830643653869628906 25

จะได้ค่าพิสัยเป็น

$$-[2^{11}] \quad to \quad +\left[(2^{11}-1)+\frac{2^{12}-1}{2^{12}}\right] = -2048 \quad to \quad +2047\frac{4095}{4096}$$

สูตรคำนวณค่าพิสัย

$$Delta = 2^{-fwl}$$

$$Range_{unsigned} = [0 , 2^{iwl} - Delta]$$

$$Range_{signed} = [-2^{iwl-1} , 2^{iwl-1} - Delta]$$

โดย

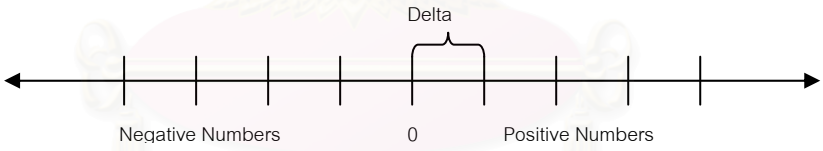
Delta = ผลต่างของช่วงค่า

fwl = จำนวนบิตเลขเศษ

iwl = จำนวนบิตเลขจำนวนเต็ม

$Range_{unsigned}$ = ค่าพิสัยแบบไม่คิดเครื่องหมาย

$Range_{signed}$ = ค่าพิสัยแบบคิดเครื่องหมาย



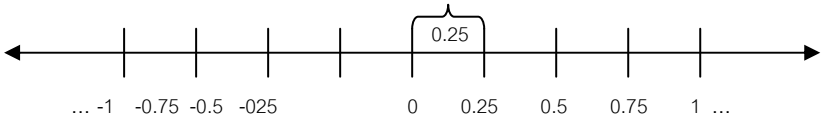
รูปที่ 2.5 รูปแบบการเก็บค่าแบบจุดตรึง

ตัวอย่างค่าพิสัย ของเลขฐานสอง 6 บิต รูปแบบ Q2 แบบคิดเครื่องหมาย

S I I I . F F

ผลต่างช่วงค่า = 2^{-2} = 0.25

ค่าพิสัย = $[-2^{4-1} , 2^{4-1} - 0.25]$ = [-8, 7.75]



รูปที่ 2.6 ผลต่างช่วงค่าของเลขฐานสอง 6 บิต รูปแบบ Q2

จากตัวอย่างจะเห็นว่าเราไม่สามารถเก็บค่าระหว่างช่วงได้ เช่น ไม่สามารถเก็บค่า 0.13 ได้เป็นต้น จึงเป็นที่มาของปัญหาการปัดเศษ และน้อยเกินเก็บ

ตัวอย่างค่าพิสัยของข้อมูล 8 บิตในรูปแบบคิว แสดงดังตารางที่ 2.7

ตารางที่ 2.7 ค่าพิสัยของการเก็บข้อมูล 8 บิต

Scaling	Precision	Range of Signed Values (low, high)	Range of Unsigned Values (low, high)
2^{1}	2.0	-256, 254	0, 510
2^{0}	1.0	-128, 127	0, 255
2^{-1}	0.5	-64, 63.5	0, 127.5
2^{-2}	0.25	-32, 31.75	0, 63.75
2^{-3}	0.125	-16, 15.875	0, 31.875
2^{-4}	0.0625	-8, 7.9375	0, 15.9375
2^{-5}	0.03125	-4, 3.96875	0, 7.96875
2^{-6}	0.015625	-2, 1.984375	0, 3.984375
2^{-7}	0.0078125	-1, 0.9921875	0, 1.9921875
2^{-8}	0.00390625	-0.5, 0.49609375	0, 0.99609375

2.1.2.2 <u>การดำเนินการทางคณิตศาสตร์ (Mathematical Operations)</u>

การดำเนินการทางคณิตศาสตร์พื้นฐานสำหรับสมการคณิตศาสตร์แบบไม่ต่อเนื่อง (Discrete Math Equation) ได้แก่ การบวก การลบ การคูณ และการหาร สิ่งที่ต้องระวังในการดำเนินการคือ ปัญหาการล้น ปัญหาน้อยเกินเก็บ ปัญหาการปัดเศษ และปัญหาการตัดปลาย (Truncation) ซึ่งปัญหาดังกล่าวเราสามารถที่จะป้องกันได้โดยการคำนวณค่า กรณีที่แย่ที่สุด (Worst Case) เพื่อที่จะทำให้มั่นใจได้ว่ารูปแบบการเก็บข้อมูล สามารถเก็บค่าในช่วงที่ต้องการได้

2.1.2.2.1 การบวกเลขฐานสอง (Binary Addition)

หลักการบวกเลขฐานสอง เป็นดังตารางที่ 2.8 ถึง 2.10

ตารางที่ 2.8 ผลการบวกเลขฐานสอง โดยช่องที่มีสีทึบจะหมายถึงมีการทดเลข

+		ตัวบวก	
		0	1
ตัวตั้งบวก	0	0	1
	1	1	0

ตารางที่ 2.9 ตัวอย่างการบวกเลขฐานสอง

บวกเลขฐานสิบ	พจน์	การบวกเลขฐานสองแบบไม่คิดเครื่องหมาย	
11	ตัวทด (Carry)	1111	1110
099	ตัวบวก (Addend)	0110	0011
095	ตัวตั้งบวก (Augend)	0101	1111
194	ผลลัพธ์	1100	0010

ตารางที่ 2.10 กรณีที่แย่ที่สุด ตัวอย่าง 3 กรณี

กรณีที่ 1	กรณีที่ 2	กรณีที่ 3
แบบไม่คิดเครื่องหมาย	แบบคิดเครื่องหมายกรณีบวก	แบบคิดเครื่องหมายกรณีลบ
1111 (15)	0111 (+7)	1000 (-8)
+ 1111 (15)	+ 0111 (+7)	+ 1000 (-8)
11110 (30)	1110 (+14)	10000 (-16)
เกิดการล้น	เกิดการเปลี่ยนเครื่องหมาย กลายเป็นค่า (-6) แทน	เกิดการล้น

จากกรณีผิดพลาดตัวอย่างทั้งสามกรณี เราจำเป็นที่จะต้องใช้จำนวนบิตเพิ่มอีก 1 บิตเพื่อทำให้ผลลัพธ์ที่ได้คงเดิม

ในการบวกเลขแบบมีเศษทศนิยมนั้นเราจำเป็นที่จะต้องเลื่อนหลักทศนิยมให้ตรงกันก่อน จึงสามารถบวกกันได้ โดยการใช้การเลื่อนเชิงคำนวณ (Arithmetic Shift) ซึ่งแตกต่างจากการเลื่อนเชิงตรรกะ (Logical Shift) วิธีการเลื่อนเชิงคำนวณแสดงดังรูปที่ 2.6

Arithmetic shift left Arithmetic shift right

รูปที่ 2.6 การเลื่อนเชิงคำนวณ

2.1.2.2.2 การบวกแบบมอดุโล (Modulo Addition)

เมื่อเรารู้ค่าจำกัดของผลลัพธ์ เราสามารถที่จะเลือกรูปแบบที่สามารถรองรับค่าจำกัดได้เพียงพอ โดยที่ไม่ต้องสนใจว่าระหว่างดำเนินการจะมีการล้นหรือไม่ ดังตัวอย่างต่อไปนี้ ใช้เลขฐานสองแบบคิดเครื่องหมาย 8 บิต ในรูปแบบ Q4 ซึ่งมีช่วงค่าตั้งแต่ -8 จนถึง +7.935

0011 . 0100 (+3.250)		Intermediate results	
0111 . 0110 (+7.375)	→	(+3.250) + (+7.373)	= +10.625 *Overflow*
1011 . 0100 (-4.750)	→	(+10.625) + (-4.750)	= +5.875
0011 . 1000 (+3.500)	→	(+5.875) + (+3.500)	= +9.375 *Overflow*
1000 . 1000 (-7.500)	→	(+9.375) + (-7.500)	= +1.875
0001 . 1110 (+1.875)	→	ได้ค่าผลลัพธ์ที่ถูกต้อง	

จากตัวอย่างจะเห็นว่าเกิดการล้นถึงสองครั้งระหว่างที่ทำการบวกแต่ผลลัพธ์ที่ได้ก็ยังถูกต้องเมื่อเรารู้ขอบเขตของผลลัพธ์สุดท้าย

สิ่งสำคัญสำหรับการบวกแบบมอดุโลคือต้องไม่ได้อยู่ในภาวะตัวสะสมอิ่มตัว (Accumulator Saturation Mode)

2.1.2.2.3 การลบเลขฐานสอง (Binary Subtraction)

หลักการลบเลขฐานสองเป็นดังตารางที่ 2.11

ตารางที่ 2.11 ผลการลบเลขฐานสอง โดยช่องที่มีสีทึบจะหมายถึงมีการยืมเลข

—		ตัวลบ	
		0	1
ตัวตั้ง	0	0	1
	1	1	0

ในการลบเลขนั้นทำได้สองวิธี วิธีแรกเป็นการลบโดยตรงเหมือนลบเลขฐานสิบ และวิธีที่สองเป็นการบวกด้วยค่าลบที่ได้จากการทำการเติมเต็มของสอง โดยทั่วไปนิยมนำนิยมวิธีที่สอง คือ นำไปบวกด้วยค่าที่ได้จากการทำการเติมเต็มของสองก่อน

ตารางที่ 2.12 ตัวอย่างการลบเลขฐานสองวิธีที่ 1 (ลบโดยตรง)

ลบเลขฐานสิบ	พจน์	การลบเลขฐานสองแบบคิดเครื่องหมาย	
1	ตัวยืม (Borrow)	110	0000
109	ตัวลบ (Minutend)	0110	1101
049	ตัวตั้งลบ (Subtrahend)	0011	0001
060	ผลลัพธ์	0011	1100

ตารางที่ 2.13 ตัวอย่างการลบเลขฐานสองวิธีที่ 2 (บวกกับค่าลบที่ได้จากการทำส่วนเติมเต็มของสอง)

บวกเลขฐานสิบ	พจน์	การบวกเลขฐานสองแบบคิดเครื่องหมาย	
1	ตัวทด (Carry)	11101	1110
109	ตัวบวก (Addend)	0110	1101
-049	ตัวตั้งบวก (Augend)	1100	1111
060	ผลลัพธ์	0011	1100

จะเห็นว่าผลลัพธ์ที่ได้เกิดการล้น แต่ตามหลักการบวกแบบมอดุโลเรารู้ว่าค่าสุดท้ายที่ได้ไม่เกินช่วงค่าที่สามารถเก็บได้ดังนั้นเราจึงพิจารณาแค่ 8 บิตเท่าเดิมจะได้ 0011 1100 = +60 ซึ่งค่าที่ได้มีความถูกต้องจริง จะสังเกตเห็นว่าหลังจากเราทำส่วนเติมเต็มของสองค่าที่ได้เมื่อคิดแบบเลขคิดเครื่องหมายแล้วค่าอาจจะเกิดการล้น เกินช่วงที่สามารถเก็บค่าได้ ดังนั้นเราจึงต้องแน่ใจว่าผลลัพธ์สุดท้ายที่ได้จะต้องไม่เกินช่วงค่าที่สามารถรองรับได้

2.1.2.2.4 การคูณเลขฐานสอง (Binary Multiplication)

หลักการคูณเลขฐานสองเป็นดังตารางที่ 2.14

ตารางที่ 2.14 ผลการคูณเลขฐานสอง

×		ตัวคูณ	
		0	1
ตัวตั้ง	0	0	0
	1	0	1

ผลลัพธ์ที่ได้จากการคูณต้องใช้จำนวนบิตข้อมูลเพิ่มขึ้นอย่างน้อยเท่ากับจำนวนบิตของสองจำนวนที่คูณกันนำมาบวกกันจึงจะสามารถเก็บค่าผลคูณได้ครบ ตัวอย่างเช่น การคูณเลขฐานสองแบบ 4 บิต

รูปแบบทั่วไป

BBBB (4 bits)
X BBBB (4 bits)
BBBBBBBB (8 bits)

เลขจุดตรึงแบบไม่คิดเครื่องหมาย

III.F (3 I และ 1 F)
XII.FF (2 I และ 2 F)
IIIII.FFF (5 I และ 3 F)

เลขจุดตรึงแบบคิดเครื่องหมาย

SII.F (1 S, 2 I และ 1 F)
XSI.FF (1 S, 2 I และ 2 F)
SSIII.FFF (2 S, 3 I และ 3 F)

จากตัวอย่างจะเห็นว่า จำนวนบิตที่ใช้แทนเครื่องหมาย จำนวนเต็ม และเลขเศษ ในผลลัพธ์จะมีค่าเท่ากับจำนวนบิตที่ใช้แทนเครื่องหมาย จำนวนเต็ม และเลขเศษ ของสองจำนวนที่คูณกัน บวกกัน ด้วยความสัมพันธ์ดังกล่าวทำให้เราจำเป็นที่จะต้องใช้จำนวนบิตเพิ่มขึ้นในการคูณเพื่อให้คำตอบที่ถูกต้องเพียงพอ ตัวอย่างเช่น เรามีเรจิสเตอร์ขนาด 32 บิต เมื่อนำมาคูณกันเราจะสามารถเก็บค่าข้อมูลที่ใช้คูณได้เพียงแค่ 16 บิตเท่านั้น เพราะว่าผลลัพธ์ที่เราต้องการถูกจำกัดอยู่

แค่ 32 บิต (16 X 16 = 32 bits) ด้วยขนาดของเรจิสเตอร์ที่ใช้เก็บข้อมูล ทำให้การคูณมีข้อจำกัดค่อนข้างมากในเรื่องของขนาดค่าที่จะนำมาคูณ และเรื่องของความถูกต้องที่ได้จากการคูณ

การคูณกันคือการบวกกันของเลขจำนวนเดิมเป็นจำนวนครั้งเท่ากับตัวคูณ ตัวอย่างเช่น

$$7 \times 5 = 35 \quad \rightarrow \quad 7 + 7 + 7 + 7 + 7 = 35$$

เมื่ออยู่ในรูปของเลขฐานสอง ก็ใช้หลักการเดียวกันแต่ใช้วิธีการเลื่อนบิตเข้ามาช่วย ดังรูปที่ 2.7

การคูณเลขฐานสอง	พจน์ (Terms)
00010001	(+17) ตัวตั้งคูณ (Multiplicand)
X 00001100	(+12) ตัวคูณ (Multiplier)
00000000	ผลคูณลำดับที่ 1
00000000	ผลคูณลำดับที่ 2
00010001	ผลคูณลำดับที่ 3
00010001	ผลคูณลำดับที่ 4
00000000	ผลคูณลำดับที่ 5
00000000	ผลคูณลำดับที่ 6
00000000	ผลคูณลำดับที่ 7
00000000	ผลคูณลำดับที่ 8
0000000000000000	ตัวทด
0000000011001100	ผลลัพธ์ (+204)

(ก)

การดำเนินการเมื่อลดขั้นตอนแล้ว	
00010001	
X 00001100	
10001	เลื่อนบิตไปทางซ้าย 2 ครั้ง
+ 10001	เลื่อนบิตไปทางซ้าย 3 ครั้ง
11001100	บวกผลลัพธ์ที่ได้จากการเลื่อนบิต

(ข)

รูปที่ 2.7 การคูณกันของเลขฐานสอง รูป (ก) เป็นการคูณเต็มรูปแบบ และรูป (ข) เป็นการคูณเฉพาะบิตที่มีค่าเป็น 1 โดยอาศัยการเลื่อนบิตเข้ามาช่วย ทำให้ขั้นตอนการคูณลดลง

จากตัวอย่างจะเห็นว่าเราไม่จำเป็นต้องคูณทีละขั้นตอน เราสามารถเลือกคูณเฉพาะบิตที่เป็น 1 แทน โดยอาศัยการเลื่อนบิตแล้วนำค่าที่ได้มาบวกกันก็จะได้ผลลัพธ์เท่ากัน

2.1.2.2.5 <u>การหารเลขฐานสอง (Binary Division)</u>

วิธีการหารนั้นจะตรงกันข้ามกับวิธีการคูณ เพราะเป็นการนำตัวหาร (Divisor) ไปลบออกจากตัวตั้งหาร (Dividend) จนกว่าเหลือเศษ 0 หรือจนกว่าจะได้ผลลัพธ์ที่ต้องการ (กรณีของการหารไม่ลงตัว เศษไม่เท่ากับ 0) ดังรูปที่ 2.6

การคูณ	**การหาร**
7 X 5 = 35	35 ÷ 7 = 5
0	35
+ 7 1	- 7 1
7	28
+ 7 2	- 7 2
14	21
+ 7 3	- 7 3
21	14
+ 7 4	- 7 4
28	7
+ 7 5	- 7 5
35	0

รูปที่ 2.8 การคูณ และการหารแบบวิธีทำซ้ำ

ตัวอย่างการหารเลขฐานสิบและเลขฐานสองดังรูปที่ 2.9

$$\begin{array}{r} 17 \\ 12\overline{)204} \\ \underline{12} \\ 84 \\ \underline{84} \\ 0 \end{array}$$

(ก)

$$\begin{array}{r} 10001 \\ 1100\overline{)11001100} \\ \underline{1100} \\ 01 \\ \underline{\ 0} \\ 011 \\ \underline{\ 0} \\ 110 \\ \underline{\ 0} \\ 1100 \\ \underline{1100} \\ 0 \end{array}$$

(ข)

รูปที่ 2.9 การหาร (ก) การหารเลขฐานสิบ (ข) การหารเลขฐานสอง

การหารนั้นมีกระบวนการตรงข้ามกับการคูณ ดังนั้นเราสามารถใช้วิธีการเลื่อนบิตมาช่วยเรื่องของการหารได้ โดยนำตัวตั้งหารไปลบออกด้วยตัวหารจนกว่าจะได้เศษเป็น 0 หรือได้ค่าที่ต้องการ ดังรูปที่ 2.10

35 ÷ 5 = 7

เลขฐานสอง	ผลหาร	กระบวนการ
0 1 0 0 0 1 1		(+35) ตัวตั้งหาร
+ 1 0 1 1 0 0 0		บวกด้วย (-5) [1011]
1 1 1 1 0 1 1	0	ผลลัพธ์ลำดับที่ 1 (เป็นลบ)
1 1 1 1 0 1 1		ผลลบลำดับที่ 1 (เป็นลบ)
+ 0 1 0 1 0 0 0		บวกกลับด้วย (+5)
0 1 0 0 0 1 1		ได้เป็นค่าก่อนทำการลบ
1 0 0 0 1 1		เลื่อนบิตไปทางซ้าย 1 ครั้ง
+ 1 0 1 1 0 0		บวกด้วย (-5)
0 0 0 1 1 1 1	1	ผลลัพธ์ลำดับที่ 2 (เป็นบวก)
0 1 1 1 1		ผลลัพธ์ลำดับ 2 เลื่อนบิต 1 ครั้ง
+ 1 0 1 1 0		บวกด้วย (-5)
0 0 1 0 1	1	ผลลัพธ์ลำดับที่ 3 (เป็นบวก)
0 1 0 1		ผลลัพธ์ลำดับที่ 3 (เป็นบวก)
+ 1 0 1 1		บวกด้วย (-5)
0 0 0 0	1	ผลลัพธ์ลำดับที่ 4 (เป็นศูนย์)

รูปที่ 2.10 การหารโดยวิธีการลบแบบเลื่อนบิต (Subtract-and-shift-left)

การหารโดยวิธีการลบแบบเลื่อนบิตทำได้ดังขั้นตอนต่อไปนี้

1. ทำให้ค่าตัวหารเป็นลบโดยนำไปทำวิธีส่วนเติมเต็มของสอง แล้วเพิ่มบิตศูนย์ทางด้านบิตนัยสำคัญน้อย (ด้านขวามือ) จนกว่าจะมีจำนวนหลักเท่ากับตัวตั้งหาร เช่น จากตัวอย่าง รูปที่ 2.8 ตัวหารมีค่า +5 (0101) เมื่อนำไปทำส่วนเติมเต็มของสองได้เป็น -5 (1011) และเมื่อเติมบิตศูนย์ทางขวามือให้มีจำนวนเท่ากับตัวตั้งหาร จากตัวอย่างมีจำนวน 7 บิต จะได้เป็น 1011000

2. นำค่าตัวหารที่ได้จากข้อที่ 1 ไปบวกเข้ากับตัวตั้งหาร
3. ตรวจดูผลจากข้อ 2 ว่าได้ค่าบิตนัยสำคัญมากที่สุดมีค่าเป็น 1 หรือ 0 ถ้ามีค่าเป็น 1 ให้บวกค่าตัวตั้งหารด้วยค่าตัวหารที่มีค่าเป็นบวก (ก่อนทำส่วนเติมเต็มของสอง) แล้วใส่ค่าผลการหารเป็น 0 แล้วเลื่อนบิตตัวตั้งหารไปทางซ้าย 1 บิต ถ้ามีค่าเป็น 0 ให้ใส่ค่าผลการหารเป็น 1 แล้วเลื่อนบิตตัวตั้งหารไปทางซ้าย 1 บิต
4. ทำซ้ำในข้อ 2 - 3 จนกว่าจะเหลือผลลัพธ์เป็น 0 หรือได้ผลลัพธ์ตามที่ต้องการ (กรณีหารไม่ลงตัว มีเศษทศนิยม)

2.1.2.3 <u>การวิเคราะห์ความผิดพลาดของการดำเนินการแบบจุดตรึง (Error Analysis of Fixed-point Operations)</u>

จากทฤษฎีเบื้องต้นที่ได้กล่าวไปแล้ว ในส่วนนี้เราจะมาลงรายละเอียดกันในเรื่องของความผิดพลาดที่เกิดเนื่องจากข้อจำกัดของการใช้เลขแบบจุดตรึงในรูปแบบคิว ซึ่งเราไม่สามารถหลีกเลี่ยงได้ อย่างแรกต้องเข้าใจก่อนว่าเมื่อเราเก็บข้อมูลในรูปแบบเลขดิจิทัลซึ่งมีจำนวนบิตจำกัด ดังนั้นบางกรณีเราอาจไม่สามารถที่จะเก็บข้อมูลให้ตรงความต้องการได้ ในระบบดิจิทัลการเก็บค่าจำนวนเต็มนั้นไม่ค่อยเกิดปัญหา แต่จะเกิดกับการเก็บค่าเลขเศษซึ่งเราไม่สามารถเก็บค่าบางค่าได้ถูกต้องครบถ้วน

การวิเคราะห์หาความผิดพลาดในงานวิจัยนี้อ้างอิงจากค่าผิดพลาด 2 แบบ ดังนี้

1. ค่าผิดพลาดสัมบูรณ์ (Absolute Error) คือ ค่าความผิดพลาดมากที่สุดที่เป็นไปได้

ค่าผิดพลาดสัมบูรณ์ = |ค่าประมาณ – ค่าจริง|

โดยเขียนค่าผิดพลาดในรูปแบบ ค่าจริง $\pm$ ค่าผิดพลาด

ตัวอย่างเช่น

5.43 ± 0.02 มีค่าพิสัยเท่ากับ $[5.43 - 0.02, 5.43 + 0.02] = [5.41, 5.45]$

2. ค่าผิดพลาดสัมพัทธ์ (Relative Error) คือ ค่าความผิดพลาดที่แปรผันตามค่าจริง

ค่าผิดพลาดสัมพัทธ์ = ค่าผิดพลาดสัมบูรณ์ / |ค่าจริง|

ค่าผิดพลาดสัมพัทธ์ = (ค่าประมาณ / ค่าจริง) – 1

เราจะบอกในรูปของค่าคลาดเคลื่อนต่างหาก ไม่ได้บอกในรูปแบบการบวกลบ ตัวอย่างเช่น ค่า 5.43 มีค่าผิดพลาดสัมพัทธ์เท่ากับ 0.02 เราสามารถคำนวณหาค่าพิสัยได้โดยนำค่าผิดพลาดสัมพัทธ์ไปคูณกับค่าหลัก ได้เป็น

มีค่าผิดพลาด $5.43 \times 0.02 = \pm 0.1086$

หรือบอกเป็นค่าพิสัยได้เป็น $[5.43 - 0.1086, 5.43 + 0.1086] = [5.3214, 5.5386]$

ค่าผิดพลาดสัมบูรณ์ และค่าผิดพลาดสัมพัทธ์มีวิธีการคำนวณที่ขึ้นกับการดำเนินการทางคณิตศาสตร์ โดยเราสามารถแบ่งการดำเนินการออกเป็น 3 ประเภทดังต่อไปนี้

สมมติ ตัวแปร a มีค่าความผิดพลาดเท่ากับ Δa และตัวแปร b มีค่าความผิดพลาดเท่ากับ Δb เราจะนิยามในรูปของ

$$A = a + \Delta a \qquad \text{และ} \qquad B = b + \Delta b$$

โดย สัญลักษณ์ Δ แทนค่าผิดพลาดสัมบูรณ์ซึ่งเป็นได้ทั้งค่าบวก และลบ

$X + \Delta X$ เป็นผลลัพธ์ที่ได้จากการดำเนินการระหว่าง A กับ B

สัญลักษณ์ δ แทนค่าผิดพลาดสัมพัทธ์

1. การบวก หรือการลบ

$$X + \Delta X = A + B = (a + \Delta a) + (b + \Delta b) = (a + b) + (\Delta a + \Delta b)$$

หรือ $$X + \Delta X = A - B = (a + \Delta a) - (b + \Delta b) = (a - b) + (\Delta a + \Delta b)$$

ค่าผิดพลาดสัมบูรณ์เท่ากับ

$$\Delta X = |\Delta a + \Delta b|$$

ค่าผิดพลาดสัมพัทธ์เท่ากับ

$$\delta X = \frac{\Delta X}{|X|} = \frac{\Delta a + \Delta b}{|X|}$$

ตัวอย่างการคำนวณ

$A = 30.6 \pm 0.2$ และ $B = 18.6 \pm 0.1$

$X = A + B = 30.6 + 18.6 = 49.2$

ค่าผิดพลาดสัมบูรณ์เท่ากับ $\Delta X = |\Delta a + \Delta b| = |0.2 + 0.1| = \pm 0.3$

ค่าผิดพลาดสัมพัทธ์เท่ากับ $\delta X = \frac{\Delta X}{|X|} = \frac{\Delta a + \Delta b}{|X|} = \frac{\pm 0.3}{|49.2|} \approx 0.0060975$

2. การคูณ หรือการหาร

$X + \Delta X = A \times B = (a + \Delta a)(b + \Delta b) = (ab) + (a\Delta b) + (b\Delta a) +$ พจน์อันดับสูงกว่า (Higher Order Term)

หรือ $X + \Delta X = A / B = (a + \Delta a) / (b + \Delta b)$

ค่าผิดพลาดสัมบูรณ์เท่ากับ

$$\Delta X = |b\Delta a + a\Delta b| = \left| ab\left(\frac{\Delta a}{a} + \frac{\Delta b}{b}\right)\right|$$

ค่าผิดพลาดสัมพัทธ์เท่ากับ

$$\delta X = \frac{\Delta X}{|X|} = \left(\frac{\Delta a}{a} + \frac{\Delta b}{b}\right)$$

ตัวอย่างการคำนวณ

$A = 30.6 \pm 0.2$ และ $B = 18.6 \pm 0.1$

$X = A \times B = 30.6 \times 18.6 = 569.16$

ค่าผิดพลาดสัมบูรณ์เท่ากับ

$$\Delta X = \left| ab\left(\frac{\Delta a}{a} + \frac{\Delta b}{b}\right)\right| = \left|(30.6)(18.6)\left(\frac{0.2}{30.6} + \frac{0.1}{18.6}\right)\right| = \pm 6.78$$

ค่าผิดพลาดสัมพัทธ์เท่ากับ

$$\delta X = \frac{\Delta X}{|X|} = \left(\frac{\Delta a}{a} + \frac{\Delta b}{b}\right) = \left(\frac{0.2}{30.6} + \frac{0.1}{18.6}\right) \approx 0.01191229$$

ในกรณีรูปแบบที่ 2 นี้จะถูกจำกัดจำนวนบิตที่ใช้ดำเนินการโดยคำสั่งเครื่อง (Machine Instruction) ในงานวิจัยนี้ใช้หน่วยประมวลผลของบริษัทฮิตาชิ รุ่น SH1 มีข้อจำกัดในการคูณ สามารถคูณได้ 2 รูปแบบ ได้แก่

แบบแรกใช้จำนวนบิต 16 บิต X 16 บิต $\rightarrow$ 32 บิต

แบบที่สองเป็นการคูณแบบมีการสะสม ใช้จำนวนบิต 16 บิต X บิต + 42 บิต $\rightarrow$ 42 บิต

3. การยกกำลัง (Power) หรือการหาค่าราก (Roots)

$X + \Delta X = A^{P} = (a + \Delta a)^{P} = a^{P} + pa^{P-1}\Delta a +$ พจน์อันดับสูงกว่า (Higher Order Term)

โดย สัญลักษณ์ P แทนเลขเศษที่ไม่ใช่จำนวนเต็ม

ค่าผิดพลาดสัมบูรณ์เท่ากับ

$$\Delta X = \left| pa^{P-1}\Delta a \right| = \left| \frac{pa^{p}\Delta a}{a} \right|$$

ค่าผิดพลาดสัมพัทธ์เท่ากับ

$$\delta X = \frac{\Delta X}{|X|} = \frac{p\Delta a}{a}$$

ตัวอย่างการคำนวณ

$A = 30.6 \pm 0.2$ และ $P = 2.2$

$X = A^{2.2} = 30.6^{2.2} \approx 1856.039173821$

ค่าผิดพลาดสัมบูรณ์เท่ากับ

$$\Delta X = \left| \frac{pa^{p}\Delta a}{a} \right| = \left| \frac{(2.2)(30.6^{2.2})(0.2)}{30.6} \right| \approx \pm 26.688144983$$

ค่าผิดพลาดสัมพัทธ์เท่ากับ

$$\delta X = \frac{\Delta X}{|X|} = \frac{p\Delta a}{a} = \frac{(2.2)(0.2)}{30.6} \approx 0.014379084$$

ในกรณีรูปแบบที่ 3 นี้มีข้อจำกัดในการเขียนโปรแกรมเนื่องจาก กรณีที่เป็นการยกกำลังเท่ากับเป็นการคูณเลขตัวเดิมซ้ำเป็นจำนวนครั้งเท่ากับเลขกำลัง ทำให้เกิดค่าผิดพลาดสะสมมาก และกรณีที่เป็นการหาค่ารากในระบบคอมพิวเตอร์จะต้องใช้วิธีการวิเคราะห์เชิงตัวเลข (Numerical Analysis) หาค่าคำตอบทำให้มีปัจจัยการคำนวณขึ้นกับวิธีการที่เลือก และถ้าไม่ได้โปรแกรมเองแต่ใช้คลังโปรแกรม (Program Library) ที่มีมาให้กับตัวแปลโปรแกรม (Compiler) ในกรณีที่ตัวแปลโปรแกรมไม่มีรหัสต้นฉบับมาให้ด้วย เราจะไม่สามารถรู้ได้ว่าฟังก์ชันที่เรียกใช้มีวิธีการโปรแกรมอย่างไร ทำให้เราไม่สามารถวิเคราะห์หาค่าความคลาดเคลื่อนที่เกิดขึ้นจริงได้

2.1.3 เลขคณิตแบบช่วง (Interval Arithmetic) [11] [12]

การคำนวณสมการส่วนใหญ่จะมีพจน์ของตัวแปรซึ่งไม่รู้ค่าดังนั้นเพื่อให้ได้ผลการคำนวณจะต้องมีการแทนค่าตัวแปรในสมการ โดยปกติเราจะแทนค่าตัวแปรแต่ละตัวด้วยค่าคงที่ แต่การคำนวณสมการแบบพลวัต (Dynamic) ค่าตัวแปรจะถูกเปลี่ยนไปเรื่อย ๆ ทำให้ต้องคำนวณสมการหลายครั้งตามค่าที่เปลี่ยนไป ในการคำนวณสมการแบบพลวัตเราจำเป็นที่จะต้องรู้ขอบเขตค่าผลลัพธ์ที่ได้จากการคำนวณ เพราะจะทำให้สามารถทำการวิเคราะห์กรณีเลวร้ายที่สุด (Worst case Analysis) ที่อาจเกิดขึ้นได้ โดยทั่วไปวิธีหาค่าขอบเขตของผลลัพธ์ทำได้โดยการพิจารณาความสัมพันธ์ของค่าต่าง ๆ ในสมการก่อนแทนค่าตัวแปรซึ่งทำได้ช้า และยากลำบาก ขึ้นกับความซับซ้อนของสมการ วิธีที่ทำได้ง่ายกว่าคือวิธีการคำนวณเลขคณิตแบบช่วง เพราะสามารถหาขอบเขตผลลัพธ์ของการคำนวณได้ในการคำนวณเพียงครั้งเดียว

ในบทนี้จะอธิบายถึงหลักการและทฤษฎีของการคำนวณเลขคณิตแบบช่วงพอสังเขปเพื่อให้เข้าใจหลักพื้นฐานการคำนวณ ทฤษฎีที่ยกมากล่าวอ้างอิงจาก [11] [12]

บทนิยาม 1

สำหรับทุกคู่ของค่าจำนวนจริงคงที่ a, b ซึ่ง $a \le b$ และเซตของจำนวนจริง x ซึ่ง $a \le x \le b$ เรียกว่า ช่วงปิด (Closed Interval) [a,b]

บทนิยาม 2

ช่วงปิดสองช่วงจะเท่ากันได้ [a,b] = [c,d] ก็ต่อเมื่อ a = c และ b = d

ทฤษฎีบท 1

กำหนดให้ [a,b] และ [c,d] เป็นช่วงปิด ดังนั้น

$$[a,b] + [c,d] = [a + c, b + d] \text{ และ}$$

$$[a,b] - [c,d] = [a - d, b - c]$$

ทฤษฎีบท 2

กำหนดให้ [a,b] และ [c,d] เป็นช่วงปิด ดังนั้น

$$[a,b] \times [c,d] = [\min(S), \max(S)]$$

โดยที่ $S = \{a \times c, a \times d, b \times c, b \times d\}$

บทนิยาม 3

ความสัมพันธ์ของตัวดำเนินการหาร $\oslash$ นิยามโดย

$$[a,b] \oslash [c,d] = \{ z \in \mathfrak{R} \mid \exists x, y.\ a \le x \le b,\ c \le y \le d,\ x = y \times z\}$$

ทฤษฎีบท 3

กำหนดให้ [a,b] และ [c,d] เป็นช่วงปิด ดังนั้น

$$[a,b] \oslash [c,d] = \begin{cases} [a,b] \times [1/d, 1/c] & \text{if } 0 \notin [c,d] \\ [-\infty,\infty] & \text{if } 0 \in [a,b] \wedge 0 \in [c,d] \\ [b/c,\infty] & \text{if } b < 0 \wedge c < d = 0 \\ [-\infty, b/d] \cup [b/c, \infty] & \text{if } b < 0 \wedge c < 0 < d \\ [-\infty,b/d] & \text{if } b < 0 \wedge 0 = c < d \\ [-\infty,a/c] & \text{if } 0 < a \wedge c < d = 0 \\ [-\infty,a/c] \cup [a/d, \infty] & \text{if } o < a \wedge c < 0 < d \\ [a/d, \infty] & \text{if } 0 < a \wedge 0 = c < d \\ \text{Empty Set} & \text{if } 0 \notin [a,b] \wedge c = d = 0 \end{cases}$$

บทตั้ง 1

กำหนดให้ x และ Y เป็นช่วงปิด เมื่อ $x / Y \subset x \oslash Y$ และ

$$x \oslash Y = \begin{cases} x / Y & \text{if } 0 \notin x \cap Y \\ \mathfrak{R} & \text{Otherwise.} \end{cases}$$

ทฤษฎีบท 4

กำหนดให้ [a,b] และ [c,d] เป็นช่วงปิดไม่ว่าง ดังนั้น

$$[a,b] / [0,0] = \text{Empty Set}, \qquad [0,0] / c,d] = \begin{cases} [0,0] & \text{if } [c,d] \neq [0,0] \\ \text{Empty Set} & \text{if } [c,d] = [0,0] \end{cases}$$

บทตั้ง 2

กำหนดให้ [a,b] เป็นช่วงปิด และ n เป็นจำนวนเต็ม โดย $n \geq 0$ ดังนั้น

$$[a,b]^n = \begin{cases} [a,b] \times [a,b] \ldots \times [a,b] & \text{if } n \in \{\text{Odd Number}\} \\ [a^n, b^n] & \text{if } n \in \{\text{Even Number}\} \end{cases}$$

ยกเว้นกรณีของ 0^0 จะได้คำตอบเป็นเซตว่าง

ทฤษฎีบท 5

จากบทตั้ง 2 เมื่อกำหนด [a,b] และ [c,d] เป็นช่วงปิด และ n เป็นจำนวนเต็ม โดย $n \geq 0$ ดังนั้น

$$[a,b]^{[c,d]} = [\min([a,b]^c), \max([a,b]^d)] \qquad \text{โดยที่ } [c,d] \in \{\text{Integer}\}$$

ทฤษฎีบท 6

กำหนดให้ [a,b] เป็นช่วงปิด และ n เป็นจำนวนจริง ดังนั้น

$$n^{[a,b]} = \begin{cases} \text{Empty Set} & \text{if } n = 0 \wedge a = 0 \cup b = 0 \\ [\min(n^a), \max(n^b)] & \text{Otherwise} \end{cases}$$

ทฤษฎีบท 7

กำหนดให้ [a,b] เป็นช่วงปิด และ $a \geq 0$ ดังนั้น

$$\sqrt{[a,b]} = [\sqrt{a}, \sqrt{b}]$$

ส่วนต่อไปเป็นการสรุปการดำเนินการพื้นฐานให้อยู่ในรูปของตารางเพื่อทำความเข้าใจได้ง่ายขึ้น ดังตารางที่ 2.15

ตารางที่ 2.15 การดำเนินการเลขคณิต โดยสัญลักษณ์ $\perp$ แทนการดำเนินการที่ไม่นิยาม R แทนจำนวนจริง PR แทนจำนวนจริงบวก และ NR แทนจำนวนจริงลบ โดยแนวนอนแทนค่า y แนวตั้ง

x+y	- ∞	NR	0	PR	+ ∞
- ∞	- ∞	- ∞	- ∞	- ∞	$\perp$
NR		NR	NR	R	+ ∞
0			0	PR	+ ∞
PR				PR	+ ∞
+ ∞					+ ∞

(ก) การบวก

x – y	- ∞	NR	0	PR	+ ∞
- ∞	$\perp$	+ ∞	+ ∞	+ ∞	+ ∞
NR	- ∞	R	PR	PR	+ ∞
0	- ∞	NR	0	PR	+ ∞
PR	- ∞	NR	NR	R	+ ∞
+ ∞	- ∞	- ∞	- ∞	- ∞	$\perp$

(ข) การลบ

x × y	- ∞	NR	0	PR	+ ∞
- ∞	+ ∞	+ ∞	$\perp$	- ∞	- ∞
NR		PR	0	NR	- ∞
0			0	0	$\perp$
PR				PR	+ ∞
+ ∞					+ ∞

(ค) การคูณ

x / y	- ∞	NR	0	PR	+ ∞
- ∞	$\perp$	0	0	0	$\perp$
NR	+ ∞	PR	0	NR	- ∞
0	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$
PR	- ∞	NR	0	PR	+ ∞
+ ∞	$\perp$	0	0	0	$\perp$

(ง) การหาร

จากรูปแบบการคำนวณที่แสดงดังตารางที่ 2.15 เป็นเงื่อนไขพื้นฐาน ส่วนต่อไปจะเป็นรายละเอียดสำหรับการคูณ และการหาร โดยมีการกำหนดสัญลักษณ์เงื่อนไขดังตารางที่ 2.16

ตารางที่ 2.16 การจัดจำแนกช่วงไม่ว่างเปล่าโดยเครื่องหมาย และ $a \leq b$

Class of [a,b]	at least one negative	at least one positive	Signs of endpoints
M	Yes	Yes	$a < 0 \wedge b > 0$
Z	No	No	$a = 0 \wedge b = 0$
P	No	Yes	$a \geq 0 \wedge b > 0$
P0	No	Yes	$a = 0 \wedge b > 0$
P1	No	Yes	$a > 0 \wedge b > 0$
N	Yes	No	$a < 0 \wedge b \leq 0$
N0	Yes	No	$a < 0 \wedge b = 0$
N1	Yes	No	$a < 0 \wedge b < 0$

ตารางที่ 2.17 กรณีวิเคราะห์สำหรับการคูณจำนวนจริงแบบช่วง [a,b] × [c,d]

Class of [a,b]	Class of [c,d]	Left Endpoint of	Right Endpoint of	Symmetry
P	P	a × c	b × d	Proved directly
P	M	b × c	b × d	Proved directly
P	N	b × c	a × d	x × y = -(x × -y)
M	P	a × d	b × d	x × y = y × x
M	M	Min(a×d, b×c)	Max(a×c, b×d)	Proved directly
M	N	b × c	a × c	x × y = -(x × -y)
N	P	a × d	b × c	x × y = -(-x × y)
N	M	a × d	a × c	x × y = -(-x × y)
N	N	b × d	a × c	x × y = -(x × -y)
Z	P, M, N, Z	0	0	Proved directly
P, M, N	Z	0	0	Proved directly

ตารางที่ 2.18 กรณีวิเคราะห์สำหรับการหารจำนวนจริงแบบช่วง [a,b] / [c,d] เมื่อ $a \le b$, $c \le d$ และต้องไม่เป็น [0,0] ทั้งคู่ หลักสุดท้ายอ้างถึงการพิสูจน์ โดย D = Directly Proof, “S1”, “S2” เป็นการอ้างถึงการใช้หลักสมมาตรเพื่อลดรูปให้อยู่ในกรณีที่กล่าวถึงก่อนแล้ว [12]

Class of	Class of	[a,b] / [c,d]	Unless	[a,d] / [c,d]	
P1	P	[a/d, b/c] \ {0}	c = 0	[a/d, ∞] \ {0}	D
P0	P	[0, b/c]	c = 0	[0, ∞]	D
M	P	[a/c, b/c]	c = 0	[-∞,∞]	D
N0	P	[a/c, 0]	c = 0	[-∞, 0]	S2
N1	P	[a/c, b/d] \ {0}	c = 0	[-∞, b/d] \ {0}	S2
P1	M	([-∞, a/c] ∪ [a/d, ∞]) \ {0}			D
P0	M	[-∞, +∞]			D
M	M	[-∞, +∞]			D
N0	M	[-∞, +∞]			S2
N1	M	([-∞, b/d] ∪ [b/c, ∞]) \ {0}			S2
P1	N	[b/d, a/c] \ {0}	d = 0	[-∞, a/c] \ {0}	S1
P0	N	[b/d, 0]	d = 0	[-∞, 0]	S1
M	N	[b/d, a/d]	d = 0	[-∞,∞]	S1
N0	N	[0, a/d]	d = 0	[0, ∞]	S2
N1	N	[b/c, a/d] \ {0}	d = 0	[b/c, ∞] \ {0}	S2

2.1.4 <u>รูปแบบการอิงดรรชนีแบบ IEEE (IEEE Floating-Point Format) [15]</u>

เป็นรูปแบบการอิงดรรชนีซึ่งเป็นไปตามมาตรฐาน IEEE-754 ซึ่งเป็นรูปแบบที่ใช้ในการกำหนดค่าเลขทศนิยมเพื่อใช้ในการคำนวณข้อมูลแบบ 32 บิต โดยมีรูปแบบดังนี้

$$N = (-1)^{S} \times 1.F \times 2^{(E-127)}$$

โดย S หมายถึง ค่าบิตเครื่องหมาย ถ้ามีค่า 1 เป็นค่าลบ ถ้ามีค่า 0 เป็นค่าบวก

N หมายถึง ค่าจำนวนเลขทศนิยม (Floating-point Number)

F หมายถึง ส่วนที่เป็นค่าเศษในเลขฐานสอง (Fractional Part in Binary Notation)

E หมายถึง เลขชี้กำลังในการการตั้งจุดทศนิยม (Exponent in Bias Representation)

มีการแบ่งค่าแต่ละบิตดังนี้

	บิตเครื่องหมาย (Sign)	บิตเลขชี้กำลัง (Exponent)	บิตค่าเศษ (Fraction)
	0	00000000	00000000000000000000000
บิตที่	31	[30 –23]	[22 -- 0]

โดยค่า บิตที่ 31 เป็นบิตบอกเครื่องหมาย ถ้ามีค่าเป็น 0 หมายถึง มีค่าเป็นเลขบวก
1 หมายถึง มีค่าเป็นเลขลบ

บิตที่ 23 - 30 เป็นบิตเลขชี้กำลัง

บิตที่ 0 - 22 เป็นบิตค่าเศษซึ่งแทนค่าส่วนแมนทิสซา

ค่าเลขชี้กำลังที่สงวนห้ามใช้ได้แก่ค่า 00000000 และ 11111111 ให้ใช้ค่าในช่วงตั้งแต่ -126 ถึง 127

ตัวอย่างการแปลงจากเลขทศนิยมฐานสิบไปเป็นรูปแบบฐานสองของเอฟอีอีอี

ค่าตัวอย่าง = 9.97 ฐานสิบ

1) แปลงให้อยู่ในรูปเลขฐานสอง :

1001.1100001

2) เลื่อนบิตให้อยู่ในรูปแบบ $1.FFFFFFFFF \times 2^{E}$:

$1.0011100001 \times 2^{3}$

3) บวกค่า 127 กับค่าเลขชี้กำลัง แล้วแปลงเป็นเลขฐานสอง :

3+127 = 130 = 10000010

4) กำหนดค่าบิตเครื่องหมายบวกหรือลบ ถ้าลบจะมีค่าเป็น 1 แต่ถ้าบวกจะมีค่าเป็น 0

5) จัดรูปแบบให้อยู่ในรูปของบิตจำนวน 32 บิต โดยเรียงตามรูปแบบที่กำหนด จะได้

0 10000010 00111000010000000000000

หรือจัดในรูปเลขฐานสิบหกจะได้

0100 0001 0001 1100 0010 0000 0000 0000 = 411C2000

การจัดรูปแบบ IEEE-754 มีเลขซึ่งเป็นข้อยกเว้นพิเศษดังตารางที่ 2.19

ตารางที่ 2.19 ค่าตัวเลขซึ่งเป็นข้อยกเว้นของ IEEE-754

Special Value	Sign (S)	Exponent (E)	Fraction (F)	Hex Value	Decimal Value
+0	0	0	0	0x0000 0000	0.0
-0	1	0	0	0x8000 0000	-0.0
1	0	127	0	0x3F80 0000	1.0
2	0	128	0	0x4000 0000	2.0
+Int	0	255	0	0x7F80 0000	+∞
-Int	1	255	0	0xFF80 0000	-∞
NaN	N/A	255	Nonzero	0x7FFF FFFF	Not A Number
LFPN	0	254	All 1's	0x7F7F FFFF	3.40282347 e+38
SFPN	0	1	All 0's	0x0080 0000	1.17549435 e-38

2.2 งานวิจัยที่เกี่ยวข้อง

2.2.1 การประยุกต์ตัวควบคุมราคาถูกรุ่น SH1 ของบริษัท Hitachi ในงานควบคุม พีไอดีสำหรับระบบมอเตอร์กระแสไฟตรงสามเฟสแบบไร้แปรง (An Application of the Low Cost Hitachi SH-1 RISC Controller for PID Control of a Three-Phase Brushless DC Motor System.) [3] โดย Ken Schultz

งานวิจัยนี้กล่าวถึงลักษณะของมอเตอร์กระแสไฟตรงสามเฟสแบบไร้แปรง และคุณสมบัติ รวมถึงฟังก์ชันการทำงานของตัวควบคุมของบริษัท Hitachi รุ่น SH1 และวิธีการเขียนโปรแกรมควบคุมแบบพีไอดี (PID Control) [12] สำหรับตัวควบคุมฮิตาชิ SH1 เพื่อควบคุมความเร็วของมอเตอร์กระแสไฟตรงสามเฟสแบบไร้แปรง โดยการนำเสนอจะเป็นการกล่าวถึงลักษณะทั่วไป ลักษณะเฉพาะ และลักษณะการทำงานของมอเตอร์กระแสไฟตรงสามเฟสแบบไร้แปรง จะมีการอธิบายถึงรายละเอียด และวิธีการควบคุมทางไฟฟ้า โดยวิธีที่นำมาใช้เป็นวิธีควบคุมแบบพีไอดีซึ่งใช้ในระบบควบคุม (Control System) ทั่วไป

งานวิจัยนี้จะกล่าวถึงทั้งทางทฤษฎี และการนำไปประยุกต์ใช้จริงในการเขียนโปรแกรมสั่งงาน โดยมีตัวอย่างการเขียนโปรแกรมพร้อมรหัสต้นฉบับ (Source Code) ที่ใช้สำหรับงานควบคุมแบบพีไอดีในรูปแบบของภาษาซี และภาษาแอสเซมบลี เพื่อใช้สำหรับควบคุมความเร็วการหมุนของมอเตอร์ โดยฟังก์ชันการทำงานจะเป็นคำสั่งเฉพาะของตัวควบคุม Hitachi รุ่น SH1 เท่านั้น

2.2.2 การจัดกระทำความผิดปรกติในการคำนวณเชิงวิทยาศาสตร์ (Exception Handling In Scientific Computing.) โดย T. E. Hull, M. S. Cohen, J. T. M. Sawchuk และ D. B. Wortman [14]

งานวิจัยนี้เกี่ยวกับการเสนอวิธีจัดการข้อผิดพลาดที่เกิดขึ้นจากการคำนวณเชิงวิทยาศาสตร์ โดยรูปแบบการนำเสนอเป็นการกล่าวถึงปัญหาที่พบบ่อย มีการยกตัวอย่างปัญหา วิธีการแก้ไขทั้งทางทฤษฎี และทางปฏิบัติในการเขียนโปรแกรม โดยอิงความ สามารถพื้นฐานของภาษาโปรแกรม และวิธีการพิจารณาปัญหาจะมองจากตัวดำเนินการ (Operator) เป็นหลัก ส่วนวิธีการที่นำมาใช้กำหนดปัญหานั้นแบ่งออกเป็นสองส่วน โดยส่วนแรกเป็นการกำหนดปัญหาไว้ก่อน (Predefine) ส่วนที่สองเป็นการให้ผู้ใช้เป็นคนกำหนดปัญหา และนักเขียนโปรแกรมจะเป็นคนแก้ปัญหาดังกล่าว ตัวอย่างของปัญหาที่ใช้ในงานวิจัยนี้ เช่น ปัญหาการล้น, ปัญหาน้อยเกินเก็บ, ปัญหาข้อผิดพลาดโดเมน (Domain error) เป็นต้น

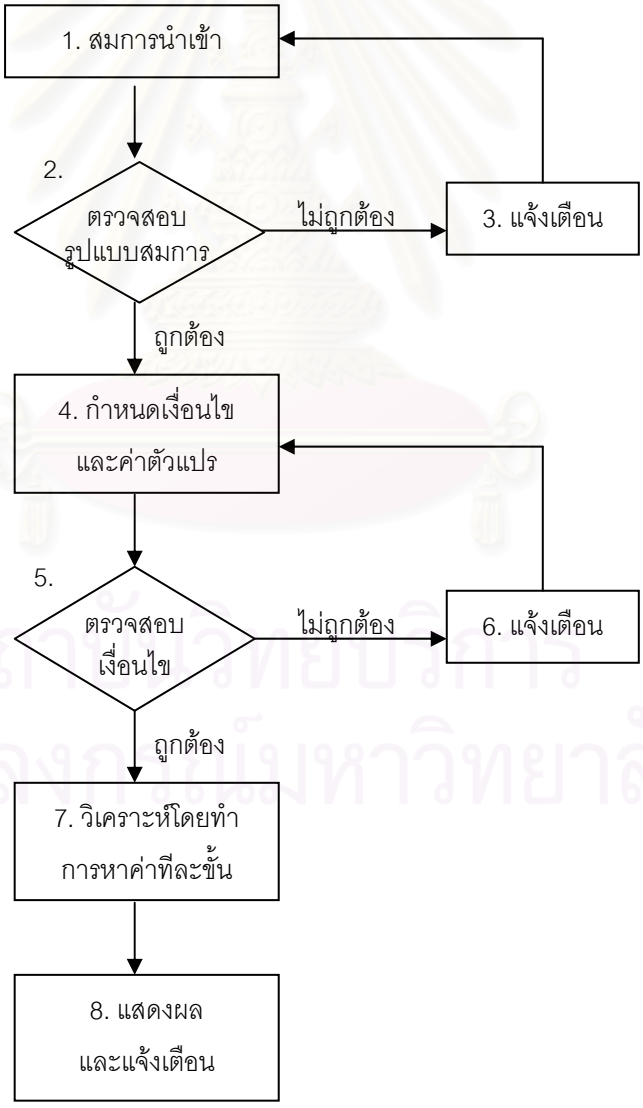
ผลที่ได้จากงานวิจัยนี้จะเป็นรูปแบบของการอธิบาย แบ่งแยกลักษณะของข้อผิดพลาดในการคำนวณเชิงวิทยาศาสตร์จากตัวดำเนินการเป็นหลัก รวมทั้งเสนอแนะวิธีการป้องกันปัญหาซึ่งสามารถนำไปประยุกต์ใช้กับปัญหาอื่นได้

บทที่ 3
ขั้นตอนวิธีทดสอบและวิเคราะห์ค่าผิดพลาดของโปรแกรมในส่วนการคำนวณ

ในบทนี้จะกล่าวถึงรายละเอียดขั้นตอนวิธีทดสอบและวิเคราะห์หาค่าผิดพลาดของโปรแกรมในส่วนการคำนวณตามรูปแบบจำลองการคำนวณดังรูปที่ 3.1 เพื่อทดสอบสมการ ค่าเงื่อนไข และค่าเริ่มต้น แล้วรายงานผลการคำนวณ และค่าผิดพลาดให้ทราบ

3.1 การจำลองการหาค่าผลการคำนวณ

ในส่วนนี้จะแสดงให้เห็นถึงขั้นตอนการดำเนินการในการจำลองหาค่าผลการคำนวณ โดยจะมีการอธิบายรายละเอียดในแต่ละขั้นตอนว่ามีวิธีการ เงื่อนไข และข้อกำหนดอย่างไรบ้างเพื่อให้เข้าใจภาพโดยรวม ในส่วนรายละเอียดของโปรแกรมที่ใช้ทดสอบจริงนั้นจะอยู่ในภาคผนวก (ก)



รูปที่ 3.1 ขั้นตอนการจำลองการคำนวณ

ขั้นตอนที่ 1 (สมการนำเข้า)

ในการจำลองสามารถใส่สมการที่ต้องการวิเคราะห์ได้ทีละหนึ่งสมการ โดยรูปแบบสมการสามารถประกอบด้วยพจน์ที่เป็นค่าคงที่ ค่าตัวแปร ตัวดำเนินการ และฟังก์ชันคณิตศาสตร์ โดยเราอนุญาตให้ใช้ตัวดำเนินการ และฟังก์ชันได้ดังต่อไปนี้

ตัวดำเนินการ ได้แก่ บวก (+) ลบ (-) คูณ (*) หาร (/) ยกกำลัง (^) และมอดุโล (%)

ฟังก์ชันคณิตศาสตร์ ได้แก่ sin, cos, tan, arcsin, arccos, arctan, exp, log, ln, sqrt, abs, ceil และ floor

ตัวอย่างสมการนำเข้า

32.14 * a + sin(40.357) ^ log(ki)

โดย ค่าคงที่ ได้แก่ 32.14 และ 40.357

ตัวแปร ได้แก่ a และ ki

ตัวดำเนินการ ได้แก่ * + และ ^ (ยกกำลัง)

ฟังก์ชันคณิตศาสตร์ ได้แก่ sin() และ log()

ขั้นตอนที่ 2 (ตรวจสอบรูปแบบสมการ)

ตรวจสอบความถูกต้องของรูปแบบสมการว่าสามารถคำนวณได้จริงหรือไม่ ตัวอย่างเช่น ใส่ชื่อฟังก์ชันคณิตศาสตร์ผิดหรือไม่ จาก sin(..) เป็น son(..) ถือว่าใส่ผิด หรือใส่สัญลักษณ์ที่ไม่ได้กำหนดไว้หรือไม่ เช่น & ซึ่งระบบไม่รู้จัก เป็นต้น

ขั้นตอนที่ 3 (แจ้งเตือน)

แจ้งเตือนข้อผิดพลาดในขั้นตอนที่ 2 ว่าผิดพลาดที่ส่วนไหน อย่างไร เพื่อทำการแก้ไขใหม่ให้ถูกต้อง

ขั้นตอนที่ 4 (กำหนดเงื่อนไขและค่าตัวแปร)

ส่วนกำหนดเงื่อนไขมี 2 ส่วน ส่วนแรกเป็นการกำหนดชนิดของตัวแปรที่ใช้เก็บค่า เช่น byte หรือ integer หรือ float เป็นต้น และส่วนที่สองเป็นการกำหนดค่ารูปแบบคิว สำหรับใช้ในส่วนการคำนวณแบบจุดตรึง

ส่วนกำหนดค่าให้กับตัวแปรสามารถกำหนดให้เป็นค่าคงที่หรือค่าพิสัย โดยค่าที่กำหนดจะต้องอยู่ภายในช่วงค่าพิสัยของชนิดตัวแปรที่เลือก

ขั้นตอนที่ 5 (ตรวจสอบเงื่อนไข)

ตรวจสอบความถูกต้องของเงื่อนไข และค่าตัวแปร ในส่วนค่าเงื่อนไขจะตรวจสอบความสอดคล้องกันระหว่างชนิดตัวแปรที่กำหนด และรูปแบบคิวที่เลือก เช่น ตัวแปรชนิด char มีขนาด 8 บิต ดังนั้นการกำหนดรูปแบบคิว จะต้องใช้จำนวนบิตไม่เกิน 8 บิต เป็นต้น และในส่วนของค่าตัวแปรจะตรวจสอบความเป็นไปได้ของค่าที่กำหนดว่ามีขนาดเกินกว่าชนิดของตัวแปรที่เลือกหรือไม่ ตัวอย่างเช่น ตัวแปร ki ถูกกำหนดให้เป็นตัวแปรชนิด unsigned char (ข้อมูลขนาด 8 บิต แบบไม่คิดเครื่องหมาย มีค่าพิสัยในช่วง [0,255]) มีการจัดรูปแบบ Q12 และมีค่าแบบพิสัยเป็น [10, 112] ซึ่งไม่เกินค่าพิสัยของชนิดตัวแปรที่ใช้เก็บ เป็นต้น

ขั้นตอนที่ 6 (แจ้งเตือน)

แจ้งเตือนข้อผิดพลาดในขั้นตอนที่ 5 ว่าผิดพลาดที่ส่วนไหน อย่างไร เพื่อทำการแก้ไขใหม่ให้ถูกต้อง

ขั้นตอนที่ 7 (วิเคราะห์โดยทำการหาค่าทีละขั้น)

การวิเคราะห์โดยทำการหาค่าทีละขั้นเริ่มจากแปลงรูปแบบสมการ จากสัญกรณ์เติมกลาง (Infix Notation) ไปเป็นรูปแบบสัญกรณ์เติมหลัง (Postfix Notation) แล้วจึงดำเนินการหาค่าทีละขั้นตามเงื่อนไข และค่าที่กำหนด ในส่วนของการหาค่าผิดพลาดจะใช้ค่าที่ได้จากการคำนวณแบบอิงดรรชนี (Floating-point) แบบความเที่ยงเชิงเดี่ยว (Single Precision) ตามมาตรฐานไอออีอีอีเจ็ดห้าสี่ (IEEE754) [15] เป็นค่าความถูกต้องอ้างอิง โดยมีรายละเอียดดังต่อไปนี้

ส่วนการคำนวณจะแบ่งออกเป็น 2 ส่วน ได้แก่

1. ส่วนคำนวณค่าความถูกต้องอ้างอิง โดยใช้โปรแกรม Borland Delphi 6

2. ส่วนที่ใช้โปรแกรม “gcc-sh-run” ของบริษัท KPIT Infosystems Limited เป็นตัวจำลองการคำนวณ โดยผลที่คำนวณได้จะถือเป็นค่าเสมือนที่ได้จากการทำงานของหน่วยประมวลผลของบริษัท Hitachi รุ่น SH1 เป็นหลัก ซึ่งในส่วนนี้ก็จะแบ่งการคำนวณออกเป็น 3 รูปแบบ ได้แก่

2.1 การคำนวณแบบปรกติ โดยใช้คำสั่ง และฟังก์ชันมาตรฐานของคอมไพเลอร์ Gcc เป็นหลัก

2.2 การคำนวณแบบจุดตรึง จะมีการแปลงชนิดตัวแปร และค่าเริ่มต้นให้อยู่ในรูปแบบเลขแบบจุดตรึง และใช้วิธีการคำนวณเฉพาะสำหรับเลขแบบจุดตรึง

2.3 การคำนวณแบบ IEEE754 แบบความเที่ยงเชิงเดี่ยว (Single Precision) โดยอาศัยความสามารถของคอมไพเลอร์ Gcc ที่สนับสนุนวิธีการคำนวณดังกล่าว

ส่วนการหาค่าผิดพลาดจะใช้ค่าผิดพลาดสัมบูรณ์ กับค่าผิดพลาดสัมพัทธ์ โดยใช้ผลที่ได้จากการคำนวณด้วยโปรแกรม gcc-sh-run เทียบกับผลถูกต้องอ้างอิงที่หาได้จากโปรแกรม Borland Delphi 6

ขั้นตอนที่ 8 (แสดงผลและแจ้งเตือน)

เป็นการแสดงผลลัพธ์ที่หาได้จากขั้นตอนที่ 7 โดยผลลัพธ์จะแสดงเป็นลำดับเรียงตามขั้นตอนการดำเนินการ มีการแสดงความผิดพลาดของผลลัพธ์ที่ได้เทียบกับค่าความถูกต้องอ้างอิง รายงานในรูปแบบค่าผิดพลาดสัมพัทธ์ และค่าผิดพลาดสัมบูรณ์ รวมถึงมีการแจ้งเตือนในกรณีที่ผลลัพธ์ไม่เป็นไปตามเงื่อนไขในขั้นตอนการดำเนินการนั้น ๆ เช่น ผลลัพธ์ที่ได้เกิดการล้น เป็นต้น และแจ้งเตือนข้อควรระวัง เช่น การหารในส่วนตัวหารต้องไม่มีสมาชิกเป็นค่า 0 เป็นต้น โดยผลลัพธ์รายงานผลในรูปแบบตาราง ส่วนการแจ้งเตือนต่าง ๆ นั้นจะแสดงในส่วนวิจารณ์ (Comment) ในโปรแกรมที่พัฒนาขึ้น

จากการดำเนินการตามขั้นตอนดังกล่าวจนได้ผลลัพธ์มีการตรวจสอบค่าเริ่มต้น และเงื่อนไข ซึ่งเป็นการรับประกันความถูกต้องเบื้องต้นได้ ส่วนผลลัพธ์และแจ้งเตือนจะทำให้ผู้ใช้ทราบได้ว่าเกิดปัญหาขึ้นจากจุดใดในการดำเนินการ รวมถึงสามารถสังเกตเห็นความเป็นไปของความสัมพันธ์ของค่าผิดพลาดในแต่ละขั้นตอนการคำนวณ ซึ่งข้อมูลที่ได้จะช่วยให้ผู้ใช้ตัดสินใจได้ว่าควรจะดำเนินการอย่างไรต่อไป

บทที่ 4
การทดลองและผลการทดลอง

ในบทนี้จะกล่าวถึงรายละเอียดขั้นตอน และเครื่องมือต่าง ๆ ในการทดลอง พร้อมทั้งผลการทดลองการวิเคราะห์ค่าผิดพลาดในกรณีศึกษาตัวอย่าง

4.1 เครื่องมือในการทดลอง

การทดลองในงานวิจัยนี้ใช้โปรแกรม Visual CEA (Visual Calculation & Error Analysis) ที่ผู้วิจัยได้พัฒนาขึ้นเอง เพื่อใช้เป็นตัวทดสอบการคำนวณและวิเคราะห์ค่าผิดพลาดของสมการนำเข้า ตามค่าเริ่มต้น และเงื่อนไขที่กำหนด โดยทดสอบการทำงานบนระบบปฏิบัติการ WindowsXp

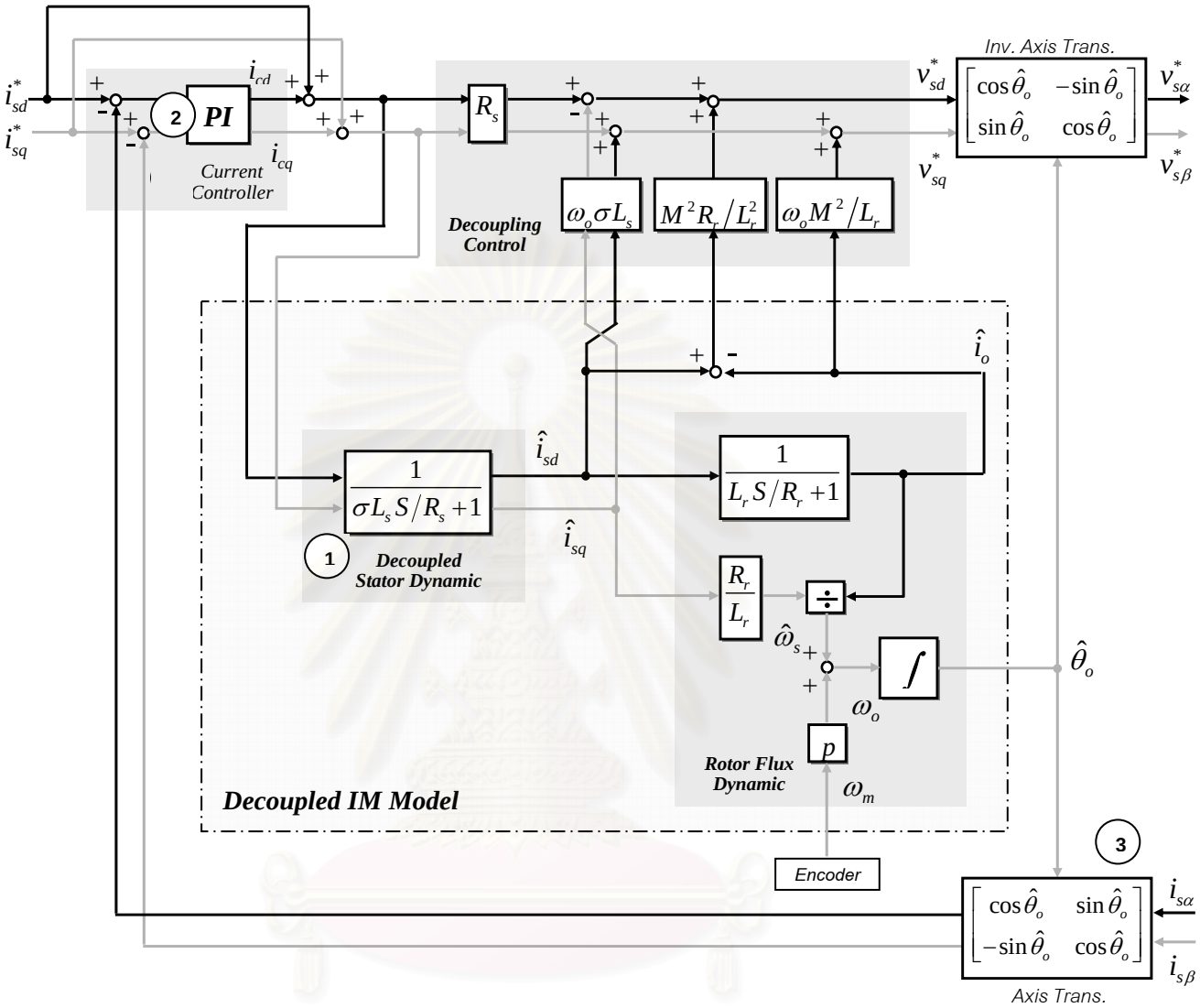
4.2 วิธีการทดลองและผลการทดลอง

ข้อมูลที่ใช้เป็นกรณีศึกษาเป็นปัญหาการโปรแกรมรหัสต้นฉบับของระบบควบคุมมอเตอร์กระแสสลับสามเฟส ที่มีใช้จริงในห้องปฏิบัติการอิเล็กทรอนิกส์กำลัง ภาควิศวกรรมไฟฟ้า จุฬาลงกรณ์มหาวิทยาลัย

ก่อนที่จะกล่าวถึงชุดข้อมูลที่ใช้เป็นกรณีศึกษาในงานวิจัยนี้ ต้องขอกล่าวถึงวิธีการโปรแกรมควบคุมมอเตอร์ ซึ่งมีรายละเอียดดังต่อไปนี้

ในส่วนการโปรแกรม นักโปรแกรมจะต้องมีความรู้เกี่ยวกับวิธีการโปรแกรมรหัสต้นฉบับของระบบควบคุมมอเตอร์ซึ่งมีขั้นตอนดังนี้

1. จะต้องมีแบบจำลองการควบคุมมอเตอร์ แสดงดังรูปที่ 4.1
2. มีการแบ่งแบบจำลองออกเป็นส่วนย่อย ๆ เพื่อทดสอบเฉพาะส่วน
3. นำส่วนย่อยที่แบ่งแล้วมาหารูปแบบสมการควบคุม
4. แปลงสมการควบคุมให้อยู่ในรูปแบบสมการคำนวณแบบไม่ต่อเนื่อง เพื่อที่จะสามารถเขียนโปรแกรมควบคุมได้
5. กำหนดค่าเริ่มต้นให้สมการ
6. เขียนโปรแกรมตามสมการคำนวณที่ได้
7. ทดสอบการทำงานของโปรแกรม
8. แก้ไขและปรับแต่งค่าในการคำนวณ จนกว่าจะได้ผลตามที่ต้องการ



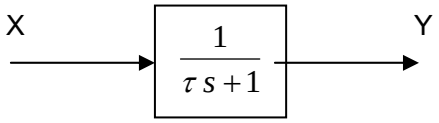
รูปที่ 4.1 ตัวอย่างโครงสร้างของระบบควบคุมเวกเตอร์แบบแรงดันโดยอาศัยการควบคุมแยกการเชื่อมร่วมที่มีวงรอบควบคุมกระแส

เราจะใช้รูปที่ 4.1 เป็นตัวอย่างกรณีศึกษาในงานวิยนี้ โดยจะแบ่งส่วนที่สนใจออกเป็นสามส่วน ในภาพจะแสดงด้วยตัวเลขในวงกลม ได้แก่

1. ส่วนแผนภาพบล็อกสมการอันดับที่หนึ่ง (First Order)
2. ส่วนแผนภาพบล็อกการควบคุมแบบ PI
3. ส่วนแผนภาพบล็อกสมการการแปลงแกน (Axis Transform)

โดยในแต่ละส่วนจะแสดงวิธีการหาสมการคำนวณแบบไม่ต่อเนื่อง กำหนดค่าเริ่มต้น และทำการทดสอบการคำนวณและวิเคราะห์ค่าผิดพลาด โดยแสดงผลในรูปแบบตาราง ดังต่อไปนี้

1. <u>ส่วนแผนภาพบล็อกสมการอันดับที่หนึ่ง</u>



รูปที่ 4.2 แผนภาพบล็อกสมการอันดับที่หนึ่ง

จากสมการควบคุม $\frac{1}{\tau s+1}$ เราสามารถแปลงให้อยู่ในรูปการคำนวณแบบไม่ต่อเนื่อง ได้ดังนี้

$$Y(k) = (X(k)Ta + Y(k-1)\tau) / (\tau + Ta)$$

แปลงให้อยู่ในรูปแบบสัญกรณ์เติมหลังได้เป็น

$$Y(k) = X(k)\ Ta \times Y(k-1)\ \tau \times + \tau\ Ta + /$$

จากนั้นจะทำการดำเนินการหาค่าที่ละขั้น ตามค่าเริ่มต้นและเงื่อนไขที่กำหนด

ส่วนถัดไปเป็นการแสดงผลลัพธ์ที่หาได้โดยจะแสดงอยู่ในรูปของตารางเรียงลำดับตามขั้นตอนการดำเนินการ มีการแสดงความผิดพลาดของผลลัพธ์ที่ได้เทียบกับค่าความถูกต้องอ้างอิงรายงานในรูปแบบค่าผิดพลาดสัมพัทธ์ และค่าผิดพลาดสัมบูรณ์

ข้อมูลค่าเริ่มต้นและเงื่อนไขแบ่งออกเป็น 2 กรณี ดังตารางที่ 4.1 และ 4.3 โดยกำหนดพจน์ที่เป็นทศนิยม ในแต่ละตารางให้มีค่ารูปแบบคิวเป็น 14 และ 15 ตามลำดับ เนื่องจากเราต้องการแสดงให้เห็นถึงความสำคัญในการเลือกค่ารูปแบบคิวที่เหมาะสม จะทำให้เกิดค่าผิดพลาดน้อยลง หรือไม่เกิดเลยก็ได้ ผลที่ได้ในแต่ละกรณีแสดงในตารางที่ 4.2 และ 4.4 ตามลำดับ

จากตารางที่ 4.1 จะเห็นว่าพจน์ลำดับที่ 2 และ 4 มีค่าผิดพลาดสัมพัทธ์มากสุดประมาณ 0.38964 หรือคิดเป็นร้อยละ 38.964 ซึ่งถือว่ามีค่ามาก ทำให้เมื่อมีการดำเนินการกับพจน์ดังกล่าวมีโอกาสที่จะทำให้เกิดค่าผิดพลาดขึ้นได้มาก ในสมการอันดับที่หนึ่งจะเห็นได้ว่าพจน์ที่มีปัญหาดังกล่าวจะมีการดำเนินการกับพจน์อื่น ๆ ทุกพจน์ โดยเฉพาะการคูณ และการหาร ที่เป็นสาเหตุให้ค่าผิดพลาดมีค่ามาก และจากตารางที่ 4.3 พจน์ลำดับที่ 2 กับ 3 มีค่าผิดพลาดสัมพัทธ์มากสุดประมาณ 0.08447 หรือคิดเป็นร้อยละ 8.447 ซึ่งมีค่าน้อยกว่าในตารางที่ 4.1 มาก ทำให้ผลลัพธ์ของการคำนวณที่ได้มีค่าผิดพลาดลดลงอย่างเห็นได้ชัด สาเหตุที่ทำให้ค่าผิดพลาดสัมพัทธ์ลดลงเป็นเพราะเราเลือกเก็บข้อมูลด้วยค่ารูปแบบคิวที่สูงขึ้น ทำให้สามารถเก็บค่าได้ละเอียดขึ้น ช่วยลดปัญหาการน้อยเกินเก็บ ทำให้สามารถเก็บค่าได้เท่ากับค่าจริง หรือใกล้เคียงค่าจริงมากขึ้น

ตารางที่ 4.1 ค่าเริ่มต้นและเงื่อนไขการคำนวณสมการอันดับหนึ่ง โดยผลลัพธ์ของการคำนวณในแต่ละขั้นจะอยู่ในรูปแบบ Q14 และสัญลักษณ์ U = Unsigned

No.	Term	Real Value	Variable Type	Q-format	Value in Q-format	Absolute Error	Relative Error
1	X(k)	[2,50]	U 8 bits	0	[2,50]	[0, 0]	[0, 0]
2	Ta	[0.0001,0.001]	U 16 bits	14	[1,16]	[0.0000234375, 0.00003896484375]	[0.0234375, 0.3896484375]
3	Y(k-1)	[2,50]	U 8 bits	0	[2,50]	[0, 0]	[0, 0]
4	τ	[0.0001,0.001]	U 16 bits	14	[1,16]	[0.0000234375, 0.00003896484375]	[0.0234375, 0.3896484375]
5	Result	-	U 32 bits	14	-	-	-

ตารางที่ 4.2 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการอันดับที่หนึ่ง โดยใช้ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.1

No.	Operation	Q-format Value	Q-Converted Value	Reference Value	Absolute Error	Relative Error	Result Name
1	xk * ta	[2, 800]	[0.0001220703125, 0.048828125]	[0.0002, 0.05]	[7.79296875E-5, 0.001171875]	[0.0234375, 0.3896484375]	ans1
2	y * τ	[2, 800]	[0.0001220703125, 0.048828125]	[0.0002, 0.05]	[7.79296875E-5, 0.001171875]	[0.0234375, 0.3896484375]	ans2
3	ans1 + ans2	[4, 1600]	[0.000244140625, 0.09765625]	[0.0004, 0.1]	[0.000155859375, 0.00234375]	[0.0234375, 0.3896484375]	ans3
4	τ + ta	[2, 32]	[0.0001220703125, 0.001953125]	[0.0002, 0.002]	[4.6875E-5, 7.79296875E-5]	[0.03896484375, 0.234375]	ans4
5	ans3 / ans4	[2048, 13107200]	[0.125, 800]	[0.2, 500]	[0.075, 300]	[0.375, 0.6]	result

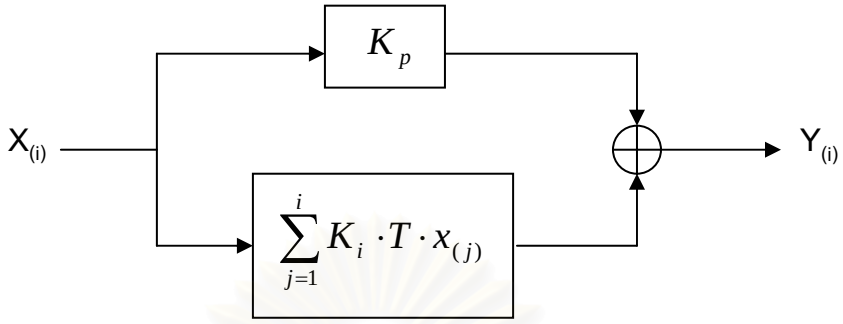
ตารางที่ 4.3 ค่าเริ่มต้นและเงื่อนไขการคำนวณสมการอันดับหนึ่ง โดยผลลัพธ์ของการคำนวณในแต่ละขั้นจะอยู่ในรูปแบบ Q15 และสัญลักษณ์ U = Unsigned

No.	Term	Real Value	Variable Type	Q-format	Value in Q-format	Absolute Error	Relative Error
1	X(k)	[2,50]	U 8 bits	0	[2,50]	[0, 0]	[0, 0]
2	Ta	[0.0001,0.001]	U 16 bits	15	[3,32]	[0.000008447265625, 0.0000234375]	[0.0234375, 0.08447265625]
3	Y(k-1)	[2,50]	U 8 bits	0	[2,50]	[0, 0]	[0, 0]
4	τ	[0.0001,0.001]	U 16 bits	15	[3,32]	[0.000008447265625, 0.0000234375]	[0.0234375, 0.08447265625]
5	Result	-	U 32 bits	15	-	-	-

ตารางที่ 4.4 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการอันดับที่หนึ่ง โดยใช้ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.3

No.	Operation	Q-format Value	Q-Converted Value	Reference Value	Absolute Error	Relative Error	Result Name
1	xk * ta	[6, 1600]	[0.00018310546875, 0.048828125]	[0.0002, 0.05]	[1.689453125E-5, 0.001171875]	[0.0234375, 0.08447265625]	ans1
2	y * τ	[6, 1600]	[0.00018310546875, 0.048828125]	[0.0002, 0.05]	[1.689453125E-5, 0.001171875]	[0.0234375, 0.08447265625]	ans2
3	ans1 + ans2	[12, 3200]	[0.0003662109375, 0.09765625]	[0.0004, 0.1]	[3.37890625E-5, 0.00234375]	[0.0234375, 0.08447265625]	ans3
4	τ + ta	[6, 64]	[0.00018310546875, 0.001953125]	[0.0002, 0.002]	[1.689453125E-5, 4.6875E-5]	[0.0234375, 0.08447265625]	ans4
5	ans3 / ans4	[6144, 17476266]	[0.1875, 533.333312988281]	[0.2, 500]	[0.0125, 33.333312988281]	[0.0625, 0.066666625976562]	result

2. <u>ส่วนแผนภาพบล็อกการควบคุมแบบ PI</u>



รูปที่ 4.3 แผนภาพบล็อกสมการควบคุม PI

จากแผนภาพบล็อกสมการ PI Control เราสามารถแปลงให้อยู่ในรูปการคำนวณแบบไม่ต่อเนื่องได้ดังนี้

$$Y_{(i)} = K_p \cdot x_{(i)} + K_I \cdot T \cdot [\, x_{(i)} + x_{(0)} \,]$$

แปลงให้อยู่ในรูปแบบสัญกรณ์เติมหลังได้เป็น

$$Y_{(i)} = K_p\ x\ *\ K_I\ T\ *\ x_{(i)}\ \ x_{(0)}\ +\ *\ +$$

จากนั้นจะทำการดำเนินการหาค่าทีละขั้น ตามค่าเริ่มต้นและเงื่อนไขที่กำหนด

ส่วนถัดไปเป็นการแสดงผลลัพธ์ที่หาได้โดยจะแสดงอยู่ในรูปของตารางเรียงลำดับตามขั้นตอนการดำเนินการ มีการแสดงความผิดพลาดของผลลัพธ์ที่ได้เทียบกับค่าความถูกต้องอ้างอิงรายงานในรูปแบบค่าผิดพลาดสัมพัทธ์ และค่าผิดพลาดสัมบูรณ์

ในส่วนนี้ใช้ชุดข้อมูลทดสอบจำนวน 2 ชุด แบ่งตามการกำหนดค่าเริ่มต้นให้แต่ละพจน์ในสมการ ชุดแรกใช้กับตารางที่ 4.5 ถึง ตารางที่ 4.10 ชุดที่สองใช้กับตารางที่ 4.11 ถึง ตารางที่ 4.15

<u>ข้อมูลชุดที่ 1</u>

ในตารางที่ 4.5, 4.7 และ 4.9 จะเห็นว่าพจน์ลำดับที่ 2, 4 และ 5 กำหนดค่ารูปแบบคิวเท่ากับ 8, 12 และ 15 ตามลำดับ ซึ่งค่าคิวที่สูงกว่าจะสามารถเก็บข้อมูลได้ละเอียดกว่า แต่สำหรับตารางที่ 4.5 และ 4.7 ค่ารูปแบบคิวที่เลือกทั้งสองแบบก็ยังไม่สามารถเก็บค่าข้อมูลของพจน์ลำดับที่ 5 (T) ไว้ได้ ทำให้เกิดปัญหาการน้อยเกินเก็บขึ้น ค่าผลลัพธ์สุดท้ายของการคำนวณจึงขึ้นกับผลลัพธ์ของการคำนวณในขั้นที่ 1 ซึ่งแสดงให้เห็นว่าใช้ค่ารูปแบบคิวสูงกว่าสามารถเก็บข้อมูลได้ละเอียดกว่าแสดงดังตารางที่ 4.6 และตารางที่ 4.8 ตามลำดับ ส่วนตารางที่ 4.9 แม้ว่าค่ารูปแบบ

คิวจะเพียงพอในการเก็บค่าข้อมูลบางส่วนของพจน์ลำดับที่ 5 แต่การคำนวณในขั้นที่ 4 ผลลัพธ์ที่ได้กลับมีค่าเท่ากับศูนย์เหมือนในตารางที่ 4.6 และตารางที่ 4.8 เนื่องจากเกิดการล้นขึ้นในขั้นตอนการคูณ เพราะชนิดตัวแปรที่เลือกใช้มีขนาดไม่เพียงพอในการเก็บข้อมูล ซึ่งเป็นสิ่งที่ควรระวังและไม่ควรมองข้ามในการกำหนดเงื่อนไขเริ่มต้นด้วย ทำให้ผลลัพธ์สุดท้ายของการคำนวณขึ้นกับการคำนวณในขั้นตอนที่ 1 เหมือนตารางที่ 4.6 และตารางที่ 4.8

รายละเอียดการล้นของการคำนวณขั้นตอนที่ 4 ในตารางที่ 4.10 เป็นดังนี้

การดำเนินการ ans2 * ans3

โดยค่า ans2 = 3 และค่า ans3 = [0, 6552] ในรูปแบบ Q15 ดังนั้นจะได้

$$3 * [0, 6552] = [3 * 0, 3 * 6552] = [0, 19656]$$

แต่การคูณผลที่ได้อยู่ในรูปแบบ Q30 ซึ่งต้องทำให้กลับมาอยู่ในรูปแบบ Q15 ตามเดิม เพราะผลการคำนวณในทุกขั้นตอนถูกกำหนดให้อยู่ในรูปแบบ Q15 จึงต้องมีการเลื่อนบิตไปทางขวา 15 บิต จะได้เป็น

$$[0, 19656] / 2^{15} = [0, 0.5998535156] = [0,0] \text{ ในรูปแบบ Q15}$$

ตารางที่ 4.5 ค่าเริ่มต้นและเงื่อนไขการคำนวณสมการควบคุม PI โดยผลลัพธ์ของการคำนวณในแต่ละขั้นจะอยู่ในรูปแบบ Q8 และสัญลักษณ์ U = Unsigned

No.	Term	Real Value	Variable Type	Q-format	Value in Q-format	Absolute Error	Relative Error
1	K_p	2	U 16 bits	0	2	0	0
2	x_i	[0, 0.1]	U 16 bits	8	[0, 25]	[0, 0.00234375]	[0, 0.0234375]
3	K_i	1	U 16 bits	0	1	0	0
4	x_0	[0, 0.1]	U 16 bits	8	[0, 25]	[0, 0.00234375]	[0, 0.0234375]
5	T	0.0001	U 16 bits	8	0	0.0001	1
6	Result	-	U 16 bits	8	-	-	-

ตารางที่ 4.6 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการควบคุม PI โดยใช้ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.5

No.	Operation	Q-format Value	Q-Converted Value	Reference Value	Absolute Error	Relative Error	Result Name
1	$K_p * x_i$	[0, 50]	[0, 0.1953125]	[0, 0.2]	[0, 0.0046875]	[0, 0.0234375]	ans1
2	$K_i * T$	0	0	0.0001	0.0001	[0, 1]	ans2
3	$x_i + x_0$	[0, 50]	[0, 0.1953125]	[0, 0.2]	[0, 0.0046875]	[0, 0.0234375]	ans3
4	ans2 * ans3	[0, 0]	[0, 0]	[0, 0.00002]	[0, 0.00002]	[0, 1]	ans4
5	ans1 + ans4	[0, 50]	[0, 0.1953125]	[0, 0.20002]	[0, 0047075]	[0, 0.023535146485351]	result

ตารางที่ 4.7 ค่าเริ่มต้นและเงื่อนไขการคำนวณสมการควบคุม PI โดยผลลัพธ์ของการคำนวณในแต่ละขั้นจะอยู่ในรูปแบบ Q12 และสัญลักษณ์ U = Unsigned

No.	Term	Real Value	Variable Type	Q-format	Value in Q-format	Absolute Error	Relative Error
1	K_p	2	U 16 bits	0	2	0	0
2	x_i	[0, 0.1]	U 16 bits	12	[0, 409]	[0, 0.000146484375]	[0, 0.00146484375]
3	K_i	1	U 16 bits	0	1	0	0
4	x_0	[0, 0.1]	U 16 bits	12	[0, 409]	[0, 0.000146484375]	[0, 0.00146484375]
5	T	0.0001	U 16 bits	12	0	0.0001	1
6	Result	-	U 16 bits	12	-	-	-

ตารางที่ 4.8 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการควบคุม PI โดยใช้ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.7

No.	Operation	Q-format Value	Q-Converted Value	Reference Value	Absolute Error	Relative Error	Result Name
1	$K_p * x_i$	[0, 818]	[0, 0.19970703125]	[0, 0.2]	[0, 0.00029296875]	[0, 0.00146484375]	ans1
2	$K_i * T$	0	0	0.0001	0.0001	[0, 1]	ans2
3	$x_i + x_0$	[0, 818]	[0, 0.19970703125]	[0, 0.2]	[0, 0.00029296875]	[0, 0.00146484375]	ans3
4	ans2 * ans3	[0, 0]	[0, 0]	[0, 0.00002]	[0, 0.00002]	[0, 1]	ans4
5	ans1 + ans4	[0, 818]	[0, 0.19970703125]	[0, 0.20002]	[0, 0.00031296875]	[0, 0.001564687281271]	result

ตารางที่ 4.9 ค่าเริ่มต้นและเงื่อนไขการคำนวณสมการควบคุม PI โดยผลลัพธ์ของการคำนวณในแต่ละขั้นจะอยู่ในรูปแบบ Q16 และสัญลักษณ์ U = Unsigned

No.	Term	Real Value	Variable Type	Q-format	Value in Q-format	Absolute Error	Relative Error
1	K_p	2	U 16 bits	0	2	0	0
2	x_i	[0, 0.1]	U 16 bits	15	[0, 3276]	[0, 0.0000244140625]	[0, 0.000244140625]
3	K_i	1	U 16 bits	0	1	0	0
4	x_0	[0, 0.1]	U 16 bits	15	[0, 3276]	[0, 0.0000244140625]	[0, 0.000244140625]
5	T	0.0001	U 16 bits	15	3	0.000008447265625	0.08447265625
6	Result	-	U 16 bits	15	-	-	-

ตารางที่ 4.10 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการควบคุม PI โดยใช้ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.9

No.	Operation	Q-format Value	Q-Converted Value	Reference Value	Absolute Error	Relative Error	Result Name
1	$K_p * x_i$	[0, 6552]	[0, 0.199951171875]	[0, 0.2]	[0, 0.48828125]	[0, 0.000244140625]	ans1
2	$K_i * T$	3	0.000091552734375	0.0001	0.000008447265625	[0, 0.08447265625]	ans2
3	$x_i + x_0$	[0, 6552]	[0, 0.199951171875]	[0, 0.2]	[0, 0.48828125]	[0, 0.000244140625]	ans3
4	ans2 * ans3	[0, 0]	[0, 0]	[0, 0.00002]	[0, 0.00002]	[0, 1]	ans4
5	ans1 + ans4	[0, 6552]	[0, 0.199951171875]	[0, 0.20002]	[0, 0.68828125]	[0, 0.000344106214378]	result

ข้อมูลชุดที่ 2

ในตารางที่ 4.11, ตารางที่ 4.13 และตารางที่ 4.15 มีการกำหนดค่าในรูปแบบคิวเท่ากับ 4, 8 และ 12 ตามลำดับ โดยในแต่ละตารางมีการกำหนดค่ารูปแบบคิวเท่ากันหมดทุกพจน์ จากตารางที่ 4.12 และตารางที่ 4.14 จะเห็นว่ามีค่าผลลัพธ์และค่าความผิดพลาดเท่ากันหมดแม้ว่าจะเลือกค่ารูปแบบคิวต่างกัน จุดสำคัญที่ทำให้ได้ค่าเท่ากันคือพจน์ลำดับที่ 5 ในตารางที่ 4.11 และตารางที่ 4.13 ไม่สามารถเก็บค่าไว้ในรูปแบบคิวที่กำหนดได้ แต่พจน์ลำดับอื่น ๆ สามารถเก็บค่าในรูปแบบคิวที่กำหนดได้โดยไม่มีค่าผิดพลาดเกิดขึ้น เป็นการชี้ให้เห็นว่าไม่จำเป็นจะต้องกำหนดค่ารูปแบบคิวสูง ๆ เพื่อให้สามารถเก็บข้อมูลได้ละเอียดขึ้น แต่ขึ้นกับค่าที่ต้องการเก็บว่าสามารถเก็บในรูปแบบคิวที่กำหนดได้โดยไม่เกิดค่าผิดพลาดหรือไม่ ในตารางที่ 4.15 ค่าพจน์ลำดับที่ 5 สามารถเก็บอยู่ในรูปแบบคิวที่กำหนดได้บางส่วน แต่การเลือกค่ารูปแบบคิวที่สูงเกินไปก็อาจทำให้เกิดปัญหาการล้นได้ เหมือนในตารางที่ 4.16 การคำนวณในขั้นตอนที่ 4 เกิดการล้นเนื่องจากไม่สามารถเก็บค่าผลลัพธ์ให้อยู่ในขอบเขตของชนิดตัวแปรที่กำหนดได้ ทำให้ค่าที่ได้ผิดไปจากค่าที่ถูกต้อง ในกรณีนี้แม้ว่าผลที่ได้จะมีค่ามากกว่าศูนย์ซึ่งต่างจากชุดข้อมูลที่ 1 ในตารางที่ 4.10 แต่กลับทำให้มีค่าผิดพลาดมากขึ้นกว่าผลการคำนวณที่ได้จากตารางที่ 4.12 และตารางที่ 4.14 และทำให้เกิดความไม่มีเสถียรภาพในการคำนวณค่าผิดพลาดเนื่องจากเราไม่สามารถคาดเดาลักษณะการล้นที่จะเกิดขึ้นจากค่าภายในช่วงได้ ในกรณีนี้จะเห็นว่าปัญหาการล้นมีความรุนแรงกว่าปัญหาน้อยเกินเก็บที่เกิดขึ้นในตารางที่ 4.12 และตารางที่ 4.14 ในการกำหนดค่ารูปแบบคิว ควรสังเกตค่าที่ถูกแปลงให้อยู่ในรูปแบบคิวที่กำหนดว่ามีขนาดใหญ่เกินไปหรือไม่ แม้ว่าผลบวกของค่ารูปแบบคิวของพจน์ที่คูณกันมีค่าไม่เกินชนิดตัวแปรที่เลือกก็ตาม

ตารางที่ 4.11 ค่าเริ่มต้นและเงื่อนไขการคำนวณสมการควบคุม PI โดยผลลัพธ์ของการคำนวณในแต่ละขั้นจะอยู่ในรูปแบบ Q4 และสัญลักษณ์ U = Unsigned

No.	Term	Real Value	Variable Type	Q-format	Value in Q-format	Absolute Error	Relative Error
1	K_p	[1, 100]	U 32 bits	4	[16, 1600]	[0, 0]	[0, 0]
2	x_i	[2, 50]	U 32 bits	4	[32, 800]	[0, 0]	[0, 0]
3	K_i	[10, 5000]	U 32 bits	4	[160, 80000]	[0, 0]	[0, 0]
4	x_0	[2, 50]	U 32 bits	4	[32, 800]	[0, 0]	[0, 0]
5	T	[0.0001, 0.001]	U 32 bits	4	[0, 0]	[0.0001, 0.001]	[1, 1]
6	Result	-	U 32 bits	4	-	-	-

ตารางที่ 4.12 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการควบคุม PI โดยใช้ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.11

No.	Operation	Q-format Value	Q-Converted Value	Reference Value	Absolute Error	Relative Error	Result
1	$K_p * x_i$	[32, 80000]	[2, 5000]	[2, 5000]	[0, 0]	[0, 0]	ans1
2	$K_i * T$	[0, 0]	[0, 0]	[0.001, 5]	[0.001, 5]	[1, 1]	ans2
3	$x_i + x_0$	[64, 1600]	[4, 100]	[4, 100]	[0, 0]	[0, 0]	ans3
4	ans2 * ans3	[0, 0]	[0, 0]	[0.004, 500]	[0.004, 500]	[1, 1]	ans4
5	ans1 + ans4	[32, 80000]	[2, 5000]	[2.004, 5500]	[0.004, 500]	[0.001996007984031, 0.090909090909090]	result

ตารางที่ 4.13 ค่าเริ่มต้นและเงื่อนไขการคำนวณสมการควบคุม PI โดยผลลัพธ์ของการคำนวณในแต่ละขั้นจะอยู่ในรูปแบบ Q8 และสัญลักษณ์ U = Unsigned

No.	Term	Real Value	Variable Type	Q-format	Value in Q-format	Absolute Error	Relative Error
1	K_p	[1, 100]	U 32 bits	8	[256, 25600]	[0, 0]	[0, 0]
2	x_i	[2, 50]	U 32 bits	8	[512, 128000]	[0, 0]	[0, 0]
3	K_i	[10, 5000]	U 32 bits	8	[2560, 1280000]	[0, 0]	[0, 0]
4	x_0	[2, 50]	U 32 bits	8	[512, 128000]	[0, 0]	[0, 0]
5	T	[0.0001, 0.001]	U 32 bits	8	[0, 0]	[0.0001, 0.001]	[1, 1]
6	Result	-	U 32 bits	8	-	-	-

ตารางที่ 4.14 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการควบคุม PI โดยใช้ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.13

No.	Operation	Q-format Value	Q-Converted Value	Reference Value	Absolute Error	Relative Error	Result
1	$K_p * x_i$	[512, 1280000]	[2, 5000]	[2, 5000]	[0, 0]	[0, 0]	ans1
2	$K_i * T$	[0, 0]	[0, 0]	[0.001, 5]	[0.001, 5]	[1, 1]	ans2
3	$x_i + x_0$	[1024, 25600]	[4, 100]	[4, 100]	[0, 0]	[0, 0]	ans3
4	ans2 * ans3	[0, 0]	[0, 0]	[0.004, 500]	[0.004, 500]	[1, 1]	ans4
5	ans1 + ans4	[512, 1280000]	[2, 5000]	[2.004, 5500]	[0.004, 500]	[0.001996007984031, 0.090909090909090]	result

ตารางที่ 4.15 ค่าเริ่มต้นและเงื่อนไขการคำนวณสมการควบคุม PI โดยผลลัพธ์ของการคำนวณในแต่ละขั้นจะอยู่ในรูปแบบ Q12 และสัญลักษณ์ U = Unsigned

No.	Term	Real Value	Variable Type	Q-format	Value in Q-format	Absolute Error	Relative Error
1	K_p	[1, 100]	U 32 bits	12	[4096, 409600]	[0, 0]	[0, 0]
2	x_i	[2, 50]	U 32 bits	12	[8192, 204800]	[0, 0]	[0, 0]
3	K_i	[10, 5000]	U 32 bits	12	[40960, 20480000]	[0, 0]	[0, 0]
4	x_0	[2, 50]	U 32 bits	12	[8192, 204800]	[0, 0]	[0, 0]
5	T	[0.0001, 0.001]	U 32 bits	12	[0, 4]	[,0.0000234375, 0.0001]	[0.0234375, 1]
6	Result	-	U 32 bits	12	-	-	-

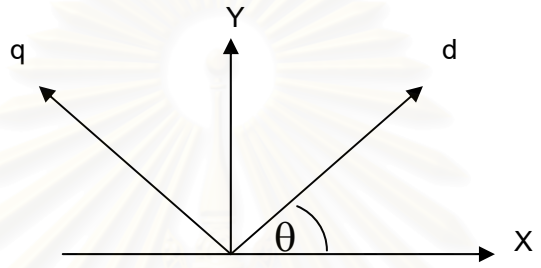
ตารางที่ 4.16 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการควบคุม PI โดยใช้ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.15

No.	Operation	Q-format Value	Q-Converted Value	Reference Value	Absolute Error	Relative Error	Result
1	$K_p * x_i$	[8192, 819200]	[2, 200]	[2, 5000]	[0, 4800]	[0, 0.96]	ans1
2	$K_i * T$	[0, 20000]	[0, 4.882815]	[0.001, 5]	[0.001, 0.1171875]	[0.0234375, 1]	ans2
3	$x_i + x_0$	[16384, 409600]	[4, 100]	[4, 100]	[0, 0]	[0, 0]	ans3
4	ans2 * ans3	[-6072, 5000]	[-23.71875, 19.53125]	[0.004, 500]	[23.72275, 480.46875]	[0.9609375, 5930.6875]	ans4
5	ans1 + ans4	[-88960, 899200]	[-21.78125, 219.53125]	[2.004, 5500]	[23.72275, 5280.46875]	[0.960085227272727, 11.8376996007984]	result

3. <u>ส่วนแผนภาพบล็อกสมการการแปลงแกน</u>

$$\begin{bmatrix} d \\ q \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

รูปที่ 4.4 แผนภาพบล็อกส่วนสมการการแปลงแกน



รูปที่ 4.5 ภาพการแปลงแกน

จากแผนภาพบล็อกสมการการแปลงแกน เมื่อทำให้อยู่ในรูปการคำนวณแบบไม่ต่อเนื่องได้เป็น

$\xi = \int \omega dt$
$\theta = \xi \bmod 2\pi$
$d = \cos\theta \cdot x + \sin\theta \cdot y$

แปลงให้อยู่ในรูปแบบ ณ. เวลาใด ๆ

$\xi_{(i)} = \omega \cdot T + \xi_{(0)}$
$\theta_{(i)} = \xi_{(i)} \bmod 2\pi$
$d_{(i)} = \cos\theta_{(i)} \cdot x_{(i)} + \sin\theta_{(i)} \cdot y_{(i)}$

เมื่อแทนค่าให้อยู่ในรูปสมการเดียวจะได้

$$d_{(i)} = \cos((\omega \cdot T + \xi_{(0)}) \bmod 2\pi) \cdot x_{(i)} + \sin((\omega \cdot T + \xi_{(0)}) \bmod 2\pi) \cdot y_{(i)}$$

ตัดพจน์ mod 2π ทิ้งเพราะโปรแกรมคำนวณสามารถรองรับการวนรอบของค่ามุม และแปลงให้อยู่ในรูปแบบสัญกรณ์เติมหลังได้เป็น

$$d_{(i)} = ((((\omega\ T\ *)\ \xi_{(0)}\ +)\ \cos)\ x\ *)\ (((\omega\ T\ *)\ \xi_{(0)}\ +)\ \sin)\ y\ *)\ +)$$

จากนั้นจะทำการดำเนินการหาค่าทีละขั้น ตามค่าเริ่มต้นและเงื่อนไขที่กำหนด

ส่วนถัดไปเป็นการแสดงผลลัพธ์ที่หาได้โดยจะแสดงอยู่ในรูปของตารางเรียงลำดับตามขั้นตอนการดำเนินการ มีการแสดงความผิดพลาดของผลลัพธ์ที่ได้เทียบกับค่าความถูกต้องอ้างอิงรายงานในรูปแบบค่าผิดพลาดสัมพัทธ์ และค่าผิดพลาดสัมบูรณ์

ในส่วนนี้จะแสดงให้เห็นถึงความแตกต่างระหว่างการคำนวณฟังก์ชันตรีโกณมิติที่มีการใช้คลังโปรแกรมของคอมไพเลอร์ Gcc และวิธีการใช้ตารางค้นหาที่กำหนดขึ้นเอง โดยแต่ละวิธีจะแบ่งการคำนวณออกเป็นสองรูปแบบได้แก่ การคำนวณแบบปรกติ และการคำนวณแบบจุดตรึง ดังต่อไปนี้

จากตารางที่ 4.18 และตาราง 4.19 ใช้วิธีการคำนวณฟังก์ชันตรีโกณที่ใช้คลังโปรแกรมของคอมไพเลอร์ Gcc จะเห็นว่าผลลัพธ์ที่ได้จากการคำนวณแบบปรกติมีค่าผิดพลาดน้อยกว่าผลลัพธ์ที่ได้จากการคำนวณแบบจุดตรึง เนื่องจากการคำนวณแบบปรกติกำหนดชนิดตัวแปรเป็น Float ซึ่งสามารถเก็บค่าเริ่มต้นได้ทั้งหมดโดยไม่มีค่าผิดพลาด และไม่มีข้อจำกัดในการคำนวณเหมือนแบบจุดตรึง

จากตารางที่ 4.20 และตารางที่ 4.21 ใช้ตารางค้นหาที่กำหนดขึ้น โดยแต่ละฟังก์ชันจะแบ่งข้อมูลค่ามุมในตารางออกเป็น 90 ค่า ในรูปแบบองศามีค่าตั้งแต่ 0 ถึง 90 องศา โดยเก็บค่าอยู่ในรูปแบบเลขทศนิยมสิบห้าหลัก จากผลลัพธ์การคำนวณจะเห็นว่าการคำนวณแบบปรกติมีค่าผิดพลาดน้อยกว่าการคำนวณแบบจุดตรึง เนื่องจากการคำนวณแบบปรกติกำหนดชนิดตัวแปรเป็น Float ซึ่งสามารถเก็บค่าเริ่มต้นได้ทั้งหมดโดยไม่มีค่าผิดพลาด และไม่มีข้อจำกัดในการคำนวณเหมือนแบบจุดตรึง

การเปรียบเทียบกันระหว่างตารางที่ 4.18 ซึ่งใช้คลังโปรแกรมคอมไพเลอร์ Gcc กับตารางที่ 4.20 ซึ่งมีการใช้ตารางค้นหา โดยทั้งสองตารางใช้วิธีการคำนวณแบบปรกติ ผลลัพธ์ที่ได้มีค่าเท่ากัน ซึ่งแสดงให้เห็นว่าการคำนวณฟังก์ชันตรีโกณมิติแบบใช้ตารางค้นหา มีความแม่นยำเท่ากับการใช้คลังโปรแกรมคอมไพเลอร์ Gcc

การเปรียบเทียบกันระหว่างตารางที่ 4.19 ซึ่งใช้คลังโปรแกรมคอมไพเลอร์ Gcc กับตารางที่ 4.21 ซึ่งมีการใช้ตารางค้นหา โดยทั้งสองตารางใช้วิธีการคำนวณแบบจุดตรึง ผลลัพธ์ที่ได้มีค่าเท่ากัน ซึ่งแสดงให้เห็นว่าการคำนวณฟังก์ชันตรีโกณมิติแบบใช้ตารางค้นหา มีความแม่นยำเท่ากับการใช้คลังโปรแกรมคอมไพเลอร์ Gcc

ตารางที่ 4.17 ค่าเริ่มต้นและเงื่อนไขการคำนวณสมการการแปลงแกน โดยผลลัพธ์ของการคำนวณในแต่ละขั้นจะอยู่ในรูปแบบ Q15, สัญลักษณ์ U = Unsigned และสัญลักษณ์ π (Pi) มีค่าเท่ากับ 3.14159265358

No.	Term	Real Value	Variable Type	Q-format	Value in Q-format	Absolute Error	Relative Error
1	ω	[0, 100π]	U 32 bits	15	[0, 10294370]	[0, 0.000024635343777]	[0, 0.000000078416734]
2	T	0.0001	U 32 bits	15	3	[0, 0]	[0, 0]
3	$\xi(0)$	[0, 2π]	U 32 bits	15	[0, 205887]	[0, 0.000012699738125]	[0, 0.000002021226098]
4	$X_{(i)}$	[0, 1]	U 32 bits	15	[0, 32768]	[0, 0]	[0, 0]
5	$Y_{(i)}$	[0, 1]	U 32 bits	15	[0, 32768]	[0, 0]	[0, 0]
6	Result	-	U 32 bits	15	-	-	-

ตารางที่ 4.18 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการการแปลงแกน โดยใช้ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.17 ยกเว้นชนิดตัวแปรที่จะใช้เป็น Float ทั้งหมด และใช้คลังโปรแกรมฟังก์ชันตรีโกณมิติของคอมไพเลอร์ Gcc

No.	Operation	Normal Value	Reference Value	Absolute Error	Relative Error	Result Name
1	ω * T	[0, 0.031415928155184]	[0, 0.0314159265358]	[0, 1.619383E-9]	[0, 5.1546593E-8]	ans1
2	ans1 + $\xi(0)$	[0, 6.3146014213562]	[0, 6.3146012336958]	[0, 1.87660400E-7]	[0, 2.9718488E-8]	ans2
3	cos(ans2)	[0.999506533145905, 1]	[0.99950656036635, 1]	[0, 2.7220444E-8]	[0, 2.7220444E-8]	ans3
4	ans3 * $X_{(i)}$	[0, 1]	[0, 1]	[0, 0]	[0, 0]	ans4
5	ω * T	[0, 0.031415928155184]	[0, 0.0314159265358]	[0, 1.619383E-9]	[0, 5.1546593E-8]	ans5
6	ans5 + $\xi(0)$	[0, 6.3146014213562]	[0, 6.3146012336958]	[0, 1.876604E-7]	[0, 2.9718488E-8]	ans6
7	sin(ans6)	[0, 0.031410947442055]	[0, 0.031410759058453]	[0, 1.88383601E-7]	[0, 5.997422758E-6]	ans7
8	ans7 * $Y_{(i)}$	[0, 0.031410947442055]	[0, 0.031410759058453]	[0, 1.88383601E-7]	[0, 5.997422758E-6]	ans8
9	ans4 + ans8	[0, 1.03141093254089]	[0, 1.03141075905845]	[0, 1.73482440E-7]	[0, 1.68199176E-7]	result

ตารางที่ 4.19 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการการแปลงแกน โดยใช้ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.17 และใช้คลังโปรแกรมฟังก์ชันตรีโกณมิติของคอมไพเลอร์ Gcc

No.	Operation	Q-format Value	Q-Converted Value	Reference Value	Absolute Error	Relative Error	Result Name
1	ω * T	[0, 942]	[0, 0.02874755859375]	[0, 0.0314159265358]	[0, 0.002668367942050]	[0, 0.084936789593306]	ans1
2	ans1 + ξ(0)	[0, 206829]	[0, 6.31192016601563]	[0, 6.3146012336958]	[0, 0.002681067680170]	[0, 0.000424582262750]	ans2
3	cos(ans2)	[32754, 32767]	[0.99957275390625, 0.999969482421875]	[0.99950656036635, 1]	[3.0517578125E-5, 6.61935399E-5]	[3.0532644141E-5, 6.6193539900E-5]	ans3
4	ans3 * $X_{(i)}$	[0, 32767]	[0, 0.999969482421875]	[0, 1]	[0, 3.0517578125E-5]	[0, 3.0517578125E-5]	ans4
5	ω * T	[0, 942]	[0, 0.02874755859375]	[0, 0.0314159265358]	[0, 0.002668367942050]	[0, 0.084936789593306]	ans5
6	ans5 + ξ(0)	[0, 206829]	[0, 6.31192016601563]	[0, 6.3146012336958]	[0, 0.002681067680170]	[0, 0.000424582262750]	ans6
7	sin(ans6)	[0, 941]	[0, 0.028717041015625]	[0, 0.031410759058453]	[0, 0.002693718042828]	[0, 0.085757814315021]	ans7
8	ans7 * $Y_{(i)}$	[0, 941]	[0, 0.028717041015625]	[0, 0.031410759058453]	[0, 0.002693718042828]	[0, 0.085757814315021]	ans8
9	ans4 + ans8	[0, 33708]	[0, 1.0286865234375]	[0, 1.03141075905845]	[0, 0.002724235620953]	[0, 0.002641271285012]	result

ตารางที่ 4.20 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการการแปลงแกน โดยใช้ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.17 ยกเว้นชนิดตัวแปรที่จะใช้เป็น Float ทั้งหมด และใช้ตารางค้นหาสำหรับฟังก์ชันตรีโกณมิติ กำหนดค่าในตารางค้นหาให้อยู่ในรูปแบบคิว 15

No.	Operation	Normal Value	Reference Value	Absolute Error	Relative Error	Result Name
1	ω * T	[0, 0.031415928155184]	[0, 0.0314159265358]	[0, 1.619383E-9]	[0, 5.1546593E-8]	ans1
2	ans1 + $\xi(0)$	[0, 6.3146014213562]	[0, 6.3146012336958]	[0, 1.87660400E-7]	[0, 2.9718488E-8]	ans2
3	cos(ans2)	[0.999506533145905, 1]	[0.99950656036635, 1]	[0, 2.7220444E-8]	[0, 2.7220444E-8]	ans3
4	ans3 * $X_{(i)}$	[0, 1]	[0, 1]	[0, 0]	[0, 0]	ans4
5	ω * T	[0, 0.031415928155184]	[0, 0.0314159265358]	[0, 1.619383E-9]	[0, 5.1546593E-8]	ans5
6	ans5 + $\xi(0)$	[0, 6.3146014213562]	[0, 6.3146012336958]	[0, 1.876604E-7]	[0, 2.9718488E-8]	ans6
7	sin(ans6)	[0, 0.031410947442055]	[0, 0.031410759058453]	[0, 1.88383601E-7]	[0, 5.997422758E-6]	ans7
8	ans7 * $Y_{(i)}$	[0, 0.031410947442055]	[0, 0.031410759058453]	[0, 1.88383601E-7]	[0, 5.997422758E-6]	ans8
9	ans4 + ans8	[0, 1.03141093254089]	[0, 1.03141075905845]	[0, 1.73482440E-7]	[0, 1.68199176E-7]	result

ตารางที่ 4.21 ผลการคำนวณทีละขั้นและการวิเคราะห์ค่าผิดพลาดของสมการการแปลงแกน โดยใช้ค่าเริ่มต้นและเงื่อนไขตามตารางที่ 4.17 และใช้ตารางค้นหาสำหรับฟังก์ชันตรีโกณมิติ กำหนดค่าในตารางค้นหาให้อยู่ในรูปแบบคิว 15

No.	Operation	Q-format Value	Q-Converted Value	Reference Value	Absolute Error	Relative Error	Result Name
1	ω * T	[0, 942]	[0, 0.02874755859375]	[0, 0.0314159265358]	[0, 0.002668367942050]	[0, 0.084936789593306]	ans1
2	ans1 + ξ(0)	[0, 206829]	[0, 6.31192016601563]	[0, 6.3146012336958]	[0, 0.002681067680170]	[0, 0.000424582262750]	ans2
3	cos(ans2)	[32754, 32767]	[0.99957275390625, 0.999969482421875]	[0.99950656036635, 1]	[3.0517578125E-5, 6.61935399E-5]	[3.0532644141E-5, 6.6193539900E-5]	ans3
4	ans3 * X(i)	[0, 32767]	[0, 0.999969482421875]	[0, 1]	[0, 3.0517578125E-5]	[0, 3.0517578125E-5]	ans4
5	ω * T	[0, 942]	[0, 0.02874755859375]	[0, 0.0314159265358]	[0, 0.002668367942050]	[0, 0.084936789593306]	ans5
6	ans5 + ξ(0)	[0, 206829]	[0, 6.31192016601563]	[0, 6.3146012336958]	[0, 0.002681067680170]	[0, 0.000424582262750]	ans6
7	sin(ans6)	[0, 941]	[0, 0.028717041015625]	[0, 0.031410759058453]	[0, 0.002693718042828]	[0, 0.085757814315021]	ans7
8	ans7 * Y(i)	[0, 941]	[0, 0.028717041015625]	[0, 0.031410759058453]	[0, 0.002693718042828]	[0, 0.085757814315021]	ans8
9	ans4 + ans8	[0, 33708]	[0, 1.0286865234375]	[0, 1.03141075905845]	[0, 0.002724235620953]	[0, 0.002641271285012]	result

จากตัวอย่างกรณีศึกษาที่ได้กล่าวไปแล้วจะเห็นได้ว่าการคำนวณแบบจุดตรึงมีความสัมพันธ์กันระหว่างค่าเริ่มต้น เงื่อนไข และการดำเนินการ ทำให้การคำนวณมีความซับซ้อน สามารถมองเห็นปัญหาได้ยาก เพราะอาจเกิดปัญหาขึ้นได้ในหลายกรณี วิธีการเลือกรูปแบบคิวที่เหมาะสมจะต้องพิจารณาถึงขนาดข้อมูลที่ต้องการเก็บ ยิ่งข้อมูลมีค่าเล็กมากเท่าไหร่ก็ยิ่งต้องเก็บด้วยค่าคิวที่สูงขึ้นด้วย แต่ต้องพิจารณาถึงนัยสำคัญที่ต้องการในการใช้งานจริง เพราะค่าคิวที่สูงเกินไปอาจทำให้เกิดปัญหาการล้นโดยเฉพาะกรณีการคูณ และการหารซึ่งมีการเลื่อนบิตจะทำให้เกิดค่าผิดพลาดได้ง่าย การกำหนดค่ารูปแบบคิวต้องกำหนดให้สัมพันธ์กับชนิดตัวแปรที่ใช้เก็บค่าเพื่อป้องกันไม่ให้เกิดปัญหาการล้น หรือน้อยเกินเก็บได้

จากผลการทดลองแสดงให้เห็นปัญหาในการคำนวณสมการควบคุมมอเตอร์ โดยปัญหาส่วนใหญ่มักเกิดขึ้นเมื่อมีการดำเนินการคูณ หรือหาร ซึ่งอาจเกิดในขั้นตอนใดขั้นตอนหนึ่ง หรือหลายขั้นตอนต่อเนื่องกัน ในบางกรณีถ้าไม่มีการจำลองการคำนวณอาจมองไม่ออกว่าจะเกิดปัญหาอะไรขึ้นที่จุดใด ในขั้นตอนใด จึงทำให้การจำลองการคำนวณและทดสอบค่าผิดพลาดมีความสำคัญต่อการนำไปใช้งานจริง

บทที่ 5
สรุปผลการวิจัย และข้อเสนอแนะ

5.1 สรุปผลการวิจัย

วิทยานิพนธ์ฉบับนี้เสนอวิธีการทดสอบการคำนวณและวิเคราะห์ค่าผิดพลาด โดยใช้การจำลองวิธีการคำนวณซึ่งเป็นวิธีหนึ่งที่ช่วยลดปัญหาในการพัฒนาโปรแกรมให้สะดวกขึ้น นักโปรแกรมสามารถมองเห็นความเป็นไปของการคำนวณว่าเกิดความผิดพลาดขึ้นในส่วนใด ขั้นตอนใดและเพราะสาเหตุใด ทำให้สามารถป้องกัน และลดปัญหาที่อาจจะเกิดขึ้นได้ ถือเป็นการแก้ปัญหาที่ต้นเหตุ เพราะในระบบแบบฝังตัวมีส่วนประกอบหลายส่วน แม้ว่าการทำการแก้จุดบกพร่องแบบทันที จะเป็นวิธีที่ใช้ได้ผล แต่สาเหตุของปัญหาอาจเกิดจากผลกระทบจากส่วนอื่นที่ไม่เกี่ยวข้องกับการคำนวณ แต่อาจนำมาใช้ต่อในการคำนวณ หรือเกิดความผิดพลาดอื่นที่มีผลกระทบ ทำให้แยกปัญหาออกจากกันได้ลำบาก และไม่สามารถยืนยันปัญหาที่เกิดได้แน่ชัดว่าเกิดจากส่วนใด และสาเหตุใด จากการทดลองผลที่ได้สามารถนำไปประยุกต์ใช้ได้จริง

ในวิทยานิพนธ์ฉบับนี้ได้ทำการทดสอบการคำนวณและวิเคราะห์ค่าผิดพลาด กับชุดข้อมูลตัวอย่างจำนวน 3 ชุดแสดงในบทที่ 4 ผลทดสอบและวิเคราะห์ค่าผิดพลาดแสดงให้เห็นว่า ในหลายกรณีเกิดค่าผิดพลาดขึ้น เป็นการบ่งชี้ว่าปัญหาในการโปรแกรมงานควบคุมมอเตอร์ ส่วนหนึ่งเกิดจากความผิดพลาดในส่วนการคำนวณ ซึ่งเป็นจุดบกพร่องที่มองเห็นได้ยากถ้าไม่มีการทดสอบ จึงทำให้การทดสอบเป็นสิ่งที่ควรพึงกระทำก่อนการเขียนโปรแกรมจริง เพื่อช่วยป้องกันและลดข้อผิดพลาดที่อาจเกิดขึ้นได้

อย่างไรก็ดีการทดสอบในงานวิจัยนี้ยังไม่ครอบคลุมการคำนวณในทุกกรณี เนื่องจากข้อจำกัดในหลายด้าน เช่น ข้อจำกัดของคลังโปรแกรมตัวคอมไพเลอร์ เป็นต้น แต่สามารถใช้เป็นตัวชี้วัดได้ในระดับหนึ่ง ซึ่งสามารถยอมรับได้ในการใช้งานจริง

5.2 ข้อเสนอแนะ

1. ในงานวิจัยนี้ได้นำเสนอเฉพาะการวิเคราะห์ค่าผิดพลาดแบบสัมบูรณ์ และค่าผิดพลาดแบบสัมพัทธ์เท่านั้น แต่การวิเคราะห์ความสัมพันธ์ระหว่างค่าพจน์ต่าง ๆ ในสมการ หรือการวิเคราะห์ในรูปแบบอื่นอาจมีส่วนช่วยให้มองเห็นจุดบกพร่อง หรือสาเหตุของการผิดพลาดได้ชัดเจนมากยิ่งขึ้น

2. ส่วนฟังก์ชันการคำนวณไม่มีมาตรฐานแน่นอนทำให้เราไม่สามารถตรวจสอบวิธีการดำเนินการเพื่อวิเคราะห์หาค่าผิดพลาดที่เกิดขึ้นจากตัวคลังโปรแกรมฟังก์ชันได้ ทำให้ผลการคำนวณที่ได้อาจเกิดความคลาดเคลื่อนไปจากค่าที่ควรได้จริง จึงควรมีการกำหนดเป็นมาตรฐานที่แน่นอน

3. ส่วนการกำหนดค่านำเข้าของการคำนวณฟังก์ชันตรีโกณมิติในงานวิจัยนี้ได้อนุญาตให้กำหนดได้สองแบบ ได้แก่ ค่าแบบองศา และค่าแบบเรเดียน แต่คลังโปรแกรมฟังก์ชันภาษาซีที่ใช้สามารถรับค่านำเข้าได้เฉพาะแบบเรเดียนเท่านั้น ทำให้อาจเกิดความคลาดเคลื่อนในการแปลงจากค่าแบบองศาไปเป็นค่าแบบเรเดียนทำให้ผลการคำนวณที่ได้อาจเกิดค่าผิดพลาดขึ้น

4. ควรเพิ่มส่วนเลือกวิธีการปัดเศษของแต่ละพจน์ เช่น การปัดขึ้น ปัดลง เป็นต้น เพื่อให้สามารถควบคุมเงื่อนไขการคำนวณได้ละเอียดแม่นยำมากขึ้น ในงานวิจัยนี้กำหนดให้ปัดลงแบบเดียวเท่านั้น

5. ควรเพิ่มส่วนเลือกกำหนดความละเอียดของการคำนวณแต่ละพจน์ว่าต้องการให้มีนัยสำคัญหรือทศนิยมกี่หลัก ในงานวิจัยนี้ได้กำหนดขอบเขตนัยสำคัญในการคำนวณสำหรับจำนวนเต็มสิบห้าหลัก และทศนิยมสิบห้าหลัก เพื่อที่จะสามารถกำหนดขอบเขตการคำนวณได้อย่างชัดเจน

6. ปัญหาเรื่องการทำให้ทันเวลาที่กำหนด (Real-time) นับเป็นปัญหาสำคัญในการควบคุมมอเตอร์แต่ไม่ได้อยู่ในขอบเขตของงานวิจัยนี้ งานวิจัยนี้ได้ใช้ภาษาซีในส่วนจำลองการคำนวณทำให้รหัสต้นฉบับที่ได้เมื่อแปลโปรแกรมด้วยคอมไพเลอร์ Gcc เป็นภาษาแอสเซมบลีอาจทำงานไม่ทันเวลาที่กำหนด แนวทางการแก้ปัญหาอาจทำให้หลายวิธี แต่วิธีที่น่าจะสะดวกและรวดเร็ว คือการแปลรูปแบบการดำเนินการเป็นภาษาแอสเซมบลีโดยตรง ไม่ต้องผ่านภาษาซี แต่วิธีนี้มีความยุ่งยาก และอาจเกิดข้อบกพร่องได้ง่ายถ้าขาดความชำนาญ

รายการอ้างอิง

1. Hitachi Single-Chip RISC Microcomputer SH7032, SH7034 Hardware Manual. Semiconductor and IC Div. Hitachi : Technical Document Center, 1994.
2. David, M. A. and Cheng, H. T. Real-Time Software for Control Program Examples in C. (n.p.) : Prentice Hall, 1993.
3. Schultz, K. An Application of the Low Cost Hitachi SH-1 RISC Controller for PID Control of a Three-Phase Brushless DC Motor System. America : Hitachi, 1997.
4. Brigitte, V., Annie, C. and Dennis, E. A Precision and Range Independent Tool for Testing Floating-Point Arithmetic I:Conversions., ACM Transaction on Mathematical Software, 27, 1 (2001) : 9-118.
5. Brigitte, V., Annie, C. and Dennis, E. A Precision and Range Independent Tool for Testing Floating-Point Arithmetic II:Conversions. ACM Transaction on Mathematical Software, 27, 1 (2001) : 119-140.
6. David, G. What Every Computer Scientist Should Know About Floating-Point Arithmetic. ACM Computing Surveys, 23, 1 (1991).
7. Stuart, B. Real-Time Computer Control An Introduction. (Second Edition). UK : Prentice Hall International, 1994.
8. Griffith, A. The Complete Reference GCC. Singapore : Mc Graw Hill, 2002.
9. Morgan, D. Numerical Methods for DSP Systems in C. John Wiley & Sons, 1997.
10. Yu, A. S. and Mackey, G. W. Fixed-Points. Mathematical world, 2, American Mathematical Society, 1992.
11. Moore, R. Interval Analysis. Englewood Cliffs : Prentice Hall, 1996.
12. Hicky, T., Ju, Q. and Emden M. V. Interval Arithmetic: From Principle to Implementation. Journal of ACM, 8, 5 (2001).
13. John, G. and Duffie N. A. Computer Control of Machines and Processes. (n.p.) : Addison-Wesley, 1998.
14. Hull, T. E., Cohen, M. S., Sawchuk, T. M. and Wortman, D. B. Exception Handling In Scientific Computing. ACM Transactions on Mathematical Software, 14, 3 (1988).

15. IEEE. IEEE Standard for binary floating-point arithmetic. ANSI/IEEE Std 754-1985, 1985.

ประวัติผู้เขียนวิทยานิพนธ์

นายเฉลิมทรัพย์ สังขวิจิตร เกิดเมื่อวันที่ 3 กันยายน พ.ศ. 2522 สำเร็จการศึกษาหลักสูตรวิศวกรรมศาสตรบัณฑิต (วศ.บ) สาขาวิศวกรรมไฟฟ้า จากคณะวิศวกรรมศาสตร์ มหาวิทยาลัยเกษตรศาสตร์ เมื่อปีการศึกษา 2544 และเข้าศึกษาต่อในหลักสูตรวิทยาศาสตรมหาบัณฑิต ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย เมื่อปีการศึกษา 2544