การวิเคราะห์ค่าผิดพลาดของเลขจุดตรึงแบบช่วงเพื่อการโปรแกรมระบบฝังตัว

Error Analysis for Fixed-point Interval Arithmetic in Embedded System Programming

เฉลิมทรัพย์ สังขวิจิตร สมบูรณ์ แสงวงค์วาณิชย์* ประภาส จงสถิตย์วัฒนา
ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
* ภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ถ.พญาไท ปทุมวัน กรุงเทพมหานคร 10330
E-mail: penockio@chula.com, prabhas@chula.ac.th

บทคัดย่อ

ระบบแบบฝังตัวใช้ในการเฝ้าสังเกต (Monitor) การตอบสนอง (Response) หรือควบคุมการทำงาน (Control) ในอุปกรณ์หลายชนิด ระบบแบบฝังตัวมีข้อจำกัดในเรื่องการทำงานให้ทันเวลาที่กำหนด (Real-Time) ขนาดของหน่วยความจำ และขนาดของเรจิสเตอร์ที่ใช้ในการประมวลผล รวมถึงข้อจำกัดด้านอื่น ๆ ทำให้การเขียนโปรแกรมควบคุมในส่วนของการคำนวณมีข้อจำกัดเกิดขึ้นอย่างมาก การโปรแกรมนิยมใช้เลขแบบจุดตรึง (Fixed-Point) ในส่วนของการคำนวณ เนื่องจากช่วยลดเวลาในการคำนวณ ใช้ขนาดพื้นที่หน่วยความจำน้อย และสามารถกำหนดช่วงค่าที่ต้องการได้ ในบทความนี้เสนอวิธีการตรวจสอบความถูกต้องและความผิดพลาดของการคำนวณเลขแบบจุดตรึงในรูปแบบการคำนวณค่าแบบช่วง (Interval Arithmetic)

คำสำคัญ : การคำนวณค่าแบบช่วง, เลขจุดตรึง, รูปแบบคิว

Abstract

Embedded system are small computer systems that monitor, respond or control external environment. They have been found in various equipments. The system has many limitation such as real-time, memory size, register size and other constrains that cause of problems in calculation programming. Fixed-Point calculation is one popular method use to solve the problems because it can reduce the size of instruction, lessen memory usage and set the value range. In the real world, the result of calculation may be out of expectation or can't accept. In this paper, we propose an evaluated simulation model/method/tool to prevent and reduce the problem of calculation in embedded programming.

Keywords: Interval Arithmetic, Fixed-Point, Q-format,

1. บทนำ

ระบบฝังตัวเป็นระบบคอมพิวเตอร์ขนาดเล็ก มีความสามารถจำกัด และมีโครงสร้างไม่ซับซ้อน เมื่อเทียบกับคอมพิวเตอร์ส่วนบุคคล โดยปกติระบบแบบฝังตัวจะพบได้ในระบบที่มีการเฝ้าสังเกต การตอบสนอง หรือการควบคุม ให้เป็นไปตามที่ต้องการ โดยข้อมูลที่รับเข้ามาอาจได้จากตัวรับรู้ (Sensor) หรือส่วนต่อประสานเพื่อรับเข้าส่งออก (Input/Output Interface) ระบบฝังตัวมีข้อจำกัดด้านความเร็วในการประมวลผล ขนาดพื้นที่หน่วยความจำ ขนาดเรจิสเตอร์ที่ใช้ในการประมวลผล และข้อจำกัดด้านอื่น ๆ ทำให้การโปรแกรมควบคุมการทำงานเกิดปัญหาในหลายด้าน

ในงานที่เกี่ยวข้องกับเรื่องการคำนวณ หรือมีการคำนวณร่วมด้วย มักเกิดปัญหาผลลัพธ์จากการคำนวณไม่เป็นไปตามที่กำหนด สาเหตุอาจเกิดจากวิธีการคำนวณไม่ถูกต้อง หรือการโปรแกรมผิดพลาด ในสองกรณีนี้เป็นปัญหาที่สามารถตรวจสอบได้ไม่ยากนัก เนื่องจากเป็นความผิดพลาดในระดับที่นักเขียนโปรแกรมสามารถสังเกตเห็น และตรวจสอบได้ แต่ปัญหาที่ตรวจสอบได้ยากเป็นปัญหาการทำงานในระดับส่วนเครื่อง (Hardware) ปัญหานี้สามารถตรวจสอบได้จากการแก้จุดบกพร่องแบบทันที (Real-Time Debugging) แต่วิธีนี้สามารถทำได้ลำบากในระบบที่มีความซับซ้อน และนักเขียนโปรแกรมต้องมีทักษะความชำนาญในการตรวจสอบด้วย วิธีที่สามารถทำได้ง่ายกว่าในการตรวจสอบความผิดพลาดคือการจำลอง (Simulation) รูปแบบการคำนวณ เพราะจะเป็น

ตัวแสดงให้เห็นว่าการคำนวณที่กำหนดขึ้นนั้นมีความคลาดเคลื่อนเท่าใด รวมถึงบอกได้ว่าจะเกิดขึ้นในส่วนใด ขั้นตอนใด และมีลักษณะเป็นอย่างไร

ปัญหาที่มักเกิดขึ้นในส่วนของการคำนวณได้แก่ ปัญหาการล้น (Overflow) ปัญหาน้อยเกินเก็บ (Underflow) ปัญหาการตัดปลาย (Truncation) ปัญหาการปัดเศษ (Rounding Off) และปัญหามาตราส่วน (Scaling) ซึ่งล้วนแล้วแต่ทำให้เกิดความคลาดเคลื่อน และความผิดพลาดในการคำนวณทั้งสิ้น การแก้ปัญหานิยมใช้การคำนวณในรูปแบบจุดตรึง (Fixed-Point) เพราะสามารถกำหนดช่วงค่าที่ต้องการได้แน่นอน และสามารถคำนวณค่าความคลาดเคลื่อนได้อย่างแม่นยำ โดยที่ไม่ขึ้นกับตัวแปลโปรแกรม (Compiler) และสามารถใช้ได้กับทุกสถาปัตยกรรมของหน่วยประมวลผล รวมถึงช่วยลดจำนวนรอบคำสั่งเครื่อง (Instruction Cycle) ในการคำนวณ

รูปแบบการคำนวณที่มักเกิดปัญหา เป็นรูปแบบการคำนวณที่มีความซับซ้อน มีค่าตัวแปร และหรือมีการเรียกใช้ฟังก์ชันทางคณิตศาสตร์ร่วมด้วย ทำให้ต้องใช้วิธีการคำนวณค่าแบบช่วงสำหรับค่าตัวแปร เพื่อเป็นการรับประกันว่าค่าผลลัพธ์ที่ได้จะอยู่ภายในขอบเขตของผลการคำนวณ และเป็นไปตามเงื่อนไขของการคำนวณ แต่วิธีการคำนวณค่าแบบช่วงจะมีกรณียกเว้นเกิดขึ้นบางกรณี ขึ้นกับตัวดำเนินการ หรือฟังก์ชัน เช่น การหารในส่วนช่วงค่าที่เป็นตัวหาร ต้องไม่มีสมาชิกเป็นค่าศูนย์ เป็นต้น

บทความนี้เสนอวิธีการตรวจสอบความถูกต้อง และความ ผิดพลาดของการคำนวณเลขแบบจุดตรึง ในรูปแบบการคำนวณค่าแบบช่วง เพื่อใช้ในการรับประกันผลลัพธ์ที่ได้ว่าจะอยู่ภายในขอบเขตของผลที่ได้จากการคำนวณ โดยในส่วนที่ 2 จะกล่าวถึงทฤษฎีพื้นฐานที่ใช้ในการคำนวณ และการหาค่าความผิดพลาด ในส่วนที่ 3 จะกล่าวถึงขั้นตอน และวิธีการจำลองรูปแบบการคำนวณ ในส่วนที่ 4 จะกล่าวถึงผลลัพธ์ที่ได้จากการทดลอง และในส่วนที่ 5 เป็นการสรุปผลและข้อเสนอแนะ

2. รูปแบบการคำนวณ

ในส่วนนี้จะอธิบายรูปแบบการคำนวณ ซึ่งอยู่ในรูปแบบเลขฐานสอง เนื่องจากมองในรูปแบบการคำนวณของหน่วยประมวลผล ในส่วนการคำนวณแบบจุดตรึงจะใช้การจัดรูปแบบคิว (Q-format)

2.1 รูปแบบเลขจุดตรึง

รูปแบบของเลขจุดตรึงจะแบ่งออกเป็น 3 ส่วน ได้แก่ ส่วนที่แทนเครื่องหมาย บวก หรือลบ ส่วนที่เป็นเลขจำนวนเต็ม และส่วนที่เป็นเลขเศษ ลองพิจารณาตัวอย่างต่อไปนี้ รูปแบบข้อมูล 16 บิต แบ่งเป็นบิตเครื่องหมาย 1 บิต จำนวนเต็ม 7 บิต และเศษ 8 บิต

S I I I I I I I FFFF FFFF

โดย S แทนบิตเครื่องหมาย (บวก/ลบ)

I แทนบิตเลขจำนวนเต็ม

F แทนบิตเลขเศษ

การอ่านค่าในส่วนบิตเลขจำนวนเต็ม จะอ่านจากขวาไปซ้ายจากตัวอย่างค่าหลักแรกจะเป็นเลข 0 (ขวาสุด) โดยเราสามารถแปลงค่ากลับไปเป็นเลขฐานสิบเพื่อง่ายต่อการเข้าใจ โดยหลักแรก (ขวาสุด) จะเป็นหลักที่ 0 และเพิ่มค่าขึ้นเรื่อยๆ ทีละหนึ่ง ไปทางซ้ายมือ การแทนค่าของตัวเลขแต่ละหลักเป็นดังสมการดังต่อไปนี้

$$\text{Column Value} = \text{bit value (0 or 1)} \times 2^{n} \quad (1)$$

โดย n แทนเลขประจำจำหลัก

การอ่านค่าในส่วนบิตเลขเศษนั้น เราอ่านค่ากลับกันกับการอ่านค่าเลขจำนวนเต็มโดยอ่านจากซ้ายมือไปขวามือ เริ่มจากบิตที่เป็นเศษตัวแรกจากซ้ายมือถือเป็นหลักที่ 1 และบิตถัดมาจะมีค่าหลักเพิ่มขึ้นทีละ 1 การแทนค่าของตัวเลขเศษแต่ละหลักเป็นดังสมการดังต่อไปนี้

$$\text{Column Value} = \text{bit value (0 or 1)} \times 2^{-n} \quad (2)$$

โดย n แทนเลขประจำจำหลัก

จากสมการเราจะได้ค่าของตัวเลขแต่ละหลักอยู่ในรูปเลขฐานสิบ การหาค่ารวมของเลขทั้งหมดจะต้องนำค่าของเลขแต่ละหลักที่แปลงเป็นฐานสิบแล้วมาบวกเข้าด้วยกัน

การหาค่าความเที่ยง (Precision) จำนวนบิตของค่าเศษเป็นตัวบ่งชี้ถึงค่าความเที่ยงสำหรับการแทนค่า โดยมีสูตรการคำนวณดังนี้

$$\text{Precision} = \frac{1}{2^n} \qquad (3)$$

โดย n แทนจำนวนบิตที่แทนเลขเศษ

ตัวอย่าง การหาค่าความเที่ยงของตัวเลข 5 บิต = $\frac{1}{2^5} = \frac{1}{32}$

การหาค่าความทน (Tolerance) จำนวนบิตของค่าเศษนอก จากบอกถึงค่าความเที่ยงแล้วยังบอกถึงค่าความทนได้ โดยค่าความผิดพลาดมากที่สุดที่เป็นไปได้จะต้องไม่เกินค่าความทน ดังสมการต่อไปนี้

$$\text{Tolerance} = \pm\frac{1}{2^{n+1}} \qquad (4)$$

โดย n แทนจำนวนบิตที่แทนเลขเศษ

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

2.2 การจัดรูปแบบคิว

การจัดรูปแบบคิวเป็นการบอกจำนวนการปรับมาตราส่วน (Scaling) ที่ประยุกต์ใช้กับเลขแบบจุดตรึงก่อนที่จะเก็บค่าในหน่วย ความจำ โดยมีรูปแบบ Qn

โดย Q หมายถึง เป็นการบอกว่าเป็นการจัดรูปแบบคิว

n หมายถึง จำนวนบิตที่แทนจำนวนเศษ

ตัวอย่าง ข้อมูลแบบคิดเครื่องหมาย 8 บิต รูปแบบQ5 จะได้เป็น
S I I F FFFF

ตัวอย่างในการจัดรูปแบบคิว ใช้ค่าเริ่มต้นเป็น 0000 1001

รูปแบบ	เลขฐานสอง	เลขฐานสิบ
Q0	0000 1001	9
Q1	0000 100. 1	9/2 = 4(1/2) = 4.5
Q2	0000 10. 01	9/4 = 2(1/4) = 2.25
Q3	0000 1. 001	9/8 = 1(1/8) = 1.125
Q4	0000 . 1001	9/16 = 0.5625

ตารางที่ 1 ตัวอย่างการจัดรูปแบบคิว

จากตัวอย่างจะเห็นความสัมพันธ์ของการจัดรูปแบบคิวเป็นไปตามสมการ

$$Qn = 2 \times Q(n+1) \qquad (5)$$

โดย Q หมายถึง ใช้การจัดรูปแบบคิว

n หมายถึง จำนวนบิตที่แทนจำนวนเศษ

วิธีการแปลงค่าจากเลขฐานสิบไปเป็นเลขฐานสองโดยใช้การจัดรูปแบบคิว เป็นดังสมการ

$$B = D \times 2^n \qquad (6)$$

โดย B หมายถึง ค่าในรูปแบบเลขฐานสอง

D หมายถึง เลขฐานสิบ

n หมายถึง จำนวนบิตที่แทนจำนวนเศษในการจัดรูปแบบคิว

ตัวอย่างของการแปลงจากเลขฐานสิบเป็นเลขฐานสองโดยใช้การจัดรูปแบบคิว ดังตารางที่ 2

เลขฐานสิบ	รูปแบบคิว	วิธีการคำนวณและผลลัพธ์
2.390625	Q12 : 4096	2.390625 x 4096
		= 0010 0110 0100 0000_2
6.78125	Q10 : 1024	6.78125 x 1024
		= 0001 0110 0010 0000_2
6.78125	Q8 : 256	6.78125 x 256

		= 0000 0110 1100 1000_2

ตารางที่ 2 การแปลงค่าจากเลขฐานสิบเป็นเลขฐานสองโดยใช้การจัดรูปแบบคิว

ในการแปลงจากเลขทศนิยมในฐานสิบ เป็นเลขฐานสองจะเกิดการปัดเศษขึ้นเนื่องจากมีจำนวนบิตจำกัด เราจะเรียกค่าผิดพลาดที่เกิดขึ้นจากการปัดเศษว่าค่าผิดพลาดควอนไทเซชั่น (Quantization Error) โดยเราสามารถหาค่าได้ในรูปแบบของเปอร์เซ็นต์ความผิดพลาดควอนไทเซชั่น หาได้จากสมการต่อไปนี้

$$\% \text{ Quantization Error } = \frac{R_E}{R_V} \times 100 \qquad (7)$$

โดย R_E หมายถึง ค่าเศษที่ถูกปัด

R_V หมายถึง ค่าเริ่มต้นก่อนการปัดเศษ

ตัวอย่างการปัดเศษที่เกิดจากการเก็บค่า 1.3 แบบ 8 บิตในรูปแบบ Q5

ได้เป็น $1.3 \times 2^5 = 1.3 \times 32 = 41.6$ เมื่อจะแทนเลขในรูปแบบคิวจะใช้เลขจำนวนเต็ม จึงต้องปัดเศษ มี 2 กรณีคือ ปัดขึ้นหรือปัดลง

กรณีปัดเศษลงจาก 41.6 เป็น 41 จะได้

$$41.6 - 0.6 = 41 = 0010\ 1001_2$$

เมื่อแปลงค่ากลับให้อยู่ในรูปฐานสิบจะได้

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

กรณีปัดเศษขึ้นจาก 41.6 เป็น 42 จะได้

$$41.6 + 0.4 = 42 = 0010\ 1010_2$$

เมื่อแปลงค่ากลับให้อยู่ในรูปฐานสิบจะได้

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

การคำนวณค่า % ความผิดพลาดควอนไทเซชั่น

กรณีปัดเศษลงจาก 41.6 เป็น 41 จะได้

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

ค่าความผิดพลาดเป็นลบเพราะว่าเราตัดค่า 0.6 ทิ้งไป

กรณีปัดเศษขึ้นจาก 41.6 เป็น 42 จะได้

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

การเก็บค่าในรูปแบบคิวใด ๆ จะสามารถรองรับค่าได้ในช่วงจำกัดขึ้นอยู่จำนวนบิตที่ใช้ และค่าคิวที่เลือก โดยเฉพาะการเก็บค่าที่เป็นเลขทศนิยม เมื่อมีการแปลงให้อยู่ในรูปแบบเลขฐานสองอาจเกิดค่าความผิดพลาดขึ้น เราสามารถคำนวณหาค่าพิสัยและค่าความผิดพลาดของการจัดเก็บในรูปแบบคิวได้ดังสมการต่อไปนี้

$$\text{Delta} = 2^{-fwl} \qquad (8)$$

$$\text{Range}_{\text{unsigned}} = [0\ ,\ 2^{iwl} - \text{Delta}] \qquad (9)$$

$$\text{Range}_{\text{signed}} = [-2^{iwl-1}\ ,\ 2^{iwl-1} - \text{Delta}] \qquad (10)$$

โดย Delta = ผลต่างของช่วงค่า

fwl = จำนวนบิตเลขเศษ

iwl = จำนวนบิตเลขจำนวนเต็ม

$\text{Range}_{\text{unsigned}}$ = ค่าพิสัยแบบไม่คิดเครื่องหมาย

$\text{Range}_{\text{signed}}$ = ค่าพิสัยแบบคิดเครื่องหมาย

ตัวอย่างการหาค่าพิสัยของเลขแบบ 24 บิต Q12 แบบคิดเครื่องหมาย โดยกำหนดรูปแบบเป็น S I I I I I I I I I I I . FFFFFFFFFFFF

จะได้ผลต่างของช่วงค่า Delta = $2^{-12} = \pm\frac{1}{8192}$

และจะได้ค่าพิสัยเป็น

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

2.3 การคำนวณค่าแบบช่วง

เป็นการคำนวณหาค่าแบบพิสัย โดยแบ่งออกได้ 2 รูปแบบ

1) เป็นการดำเนินการระหว่างค่าพิสัยกับค่าพิสัย ตัวอย่างการดำเนินการพื้นฐานในการคำนวณ ได้แก่

การบวก $[y,z] = [a,b] + [c,d] = [a + c, b + d]$

การลบ $[y,z] = [a,b] - [c,d] = [a - c, b - d]$

การคูณ $[y,z] = [a,b] \times [c,d] =$
$[\min(ac, ad, bc, bd), \max(ac, ad, bc, bd)]$

การหาร [y,x] = [a,b] / [c,d] = [a,b] × $\left[\frac{1}{c},\frac{1}{d}\right]$

$[\min(\frac{a}{c},\frac{a}{d},\frac{b}{c},\frac{b}{d}), \max(\frac{a}{c},\frac{a}{d},\frac{b}{c},\frac{b}{d})]$

โดยที่ [c, d] ต้องไม่มีสมาชิกเป็น 0

2) เป็นการดำเนินการระหว่างค่าคงที่กับค่าพิสัย ตัวอย่างการดำเนินการพื้นฐานในการคำนวณ ได้แก่

การบวก [y,x] = [a,b] + 5.33 = [a + 5.33, b + 5.33]

การลบ [y,x] = [a,b] – 5.33 = [a – 5.33, b – 5.33]

การคูณ [y,x] = [a,b] × 5.33 = [a×5.33,b×5.33]

การหาร [y,x] = [a,b] / 5.33 = [a / 5.33, b / 5.33]

โดยทั้งสองรูปแบบมีข้อจำกัด และข้อยกเว้นในบางกรณี เช่นการหาร [a,b] / [c,d] โดยที่ [c,d] ต้องไม่มีสมาชิกเป็น 0 เป็นต้น

2.4 การคำนวณหาค่าผิดพลาด

การวิเคราะห์หาความผิดพลาดอ้างอิงจากค่าผิดพลาด 2 แบบ

1) ค่าผิดพลาดสัมบูรณ์ (Absolute Error) คือ ค่าความผิดพลาดมากที่สุดที่เป็นไปได้

ค่าผิดพลาดสัมบูรณ์ = | ค่าประมาณ ─ ค่าจริง | (11)

โดย ค่าประมาณ หมายถึง ค่าหลังการปัดเศษ

ค่าจริง หมายถึง ค่าเริ่มต้นที่ไม่ได้ถูกปัดเศษ

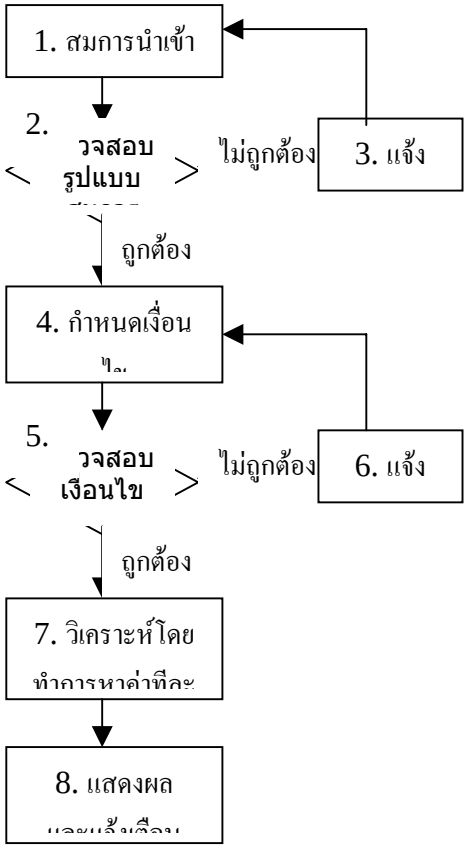
2) ค่าผิดพลาดสัมพัทธ์ (Relative Error) คือ ค่าความผิดพลาดที่แปรผันตามค่าจริง

ค่าผิดพลาดสัมพัทธ์ = ค่าผิดพลาดสัมบูรณ์ / |ค่าจริง|(12)

ค่าผิดพลาดสัมพัทธ์ = | (ค่าประมาณ / ค่าจริง) ─ 1 | (13)

3. การจำลองการหาค่าผลการคำนวณ

การวิเคราะห์ความผิดพลาดของผลการคำนวณ ใช้การจำลองแบบการคำนวณ โดยผู้ใช้ใส่อินพุตเป็นสมการ ชนิดของตัวแปร ค่าช่วง และเลือกกำหนดรูปแบบคิวของตัวแปรในสมการ โปรแกรมวิเคราะห์จะทำการคำนวณตามสมการทีละขั้น และตรวจสอบผลลัพธ์ว่าอยู่ในช่วงที่กำหนดหรือไม่ โปรแกรมวิเคราะห์มีการทำงาน 8 ขั้นตอนดังรูปที่ 1



รูปที่ 1 ขั้นตอนการจำลองการคำนวณ

ขั้นตอนที่ 1. ใส่สมการที่ต้องการวิเคราะห์ได้หนึ่งสมการ ประกอบด้วยพจน์ที่เป็นค่าคงที่ ค่าตัวแปร ตัวดำเนินการ และฟังก์ชันคณิตศาสตร์ โดยเราอนุญาตให้ใช้ตัวดำเนินการ และฟังก์ชันได้ดังต่อไปนี้

ตัวดำเนินการ ได้แก่ บวก (+) ลบ (-) คูณ (*) หาร (/) และยกกำลัง (^)

ฟังก์ชันคณิตศาสตร์ ได้แก่ sin cos tan exp log ln sqrt abs ceil 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. ตรวจสอบความถูกต้องของรูปแบบสมการ ตัวอย่างเช่น ใส่ชื่อฟังก์ชันคณิตศาสร์ผิด หรือใส่สัญลักษณ์ที่ไม่ได้กำหนดไว้

ขั้นตอนที่ 3. แจ้งเตือนข้อผิดพลาดในขั้นตอนที่ 2 ว่าผิดพลาดที่ส่วนไหน อย่างไร เพื่อทำการแก้ไขใหม่ให้ถูกต้อง

ขั้นตอนที่ 4. ส่วนกำหนดเงื่อนไขมี 2 ส่วน ส่วนแรกเป็นการกำหนดชนิดของตัวแปรที่ใช้เก็บค่า เช่น byte integer หรือfloat เป็นต้น และส่วนที่สองเป็นการกำหนดค่าแบบจุดตรึงในรูปแบบคิว ว่าต้องการให้เป็นเลขคิดเครื่องหมายหรือไม่ มีบิตเลขจำนวนเต็มกี่บิต และมีบิตเลขเศษกี่บิตส่วน กำหนดค่าให้กับตัวแปรสามารถกำหนดให้เป็นค่าคงที่หรือค่าแบบพิสัย โดยค่าที่กำหนดจะต้องอยู่ภายในค่าพิสัยของชนิดตัวแปรที่เลือก
ตัวอย่างเช่น ตัวแปร ki ถูกกำหนดเป็นตัวแปรแบบ byte (8 บิต) มีการจัดรูปแบบ Q12 และมีค่าแบบพิสัยในช่วง [10, 112] เป็นต้น

ขั้นตอนที่ 5. ตรวจสอบความถูกต้องของเงื่อนไข และค่าตัวแปร ในส่วนค่าเงื่อนไขจะตรวจสอบความสอดคล้องกันระหว่างชนิดตัวแปรที่กำหนด และรูปแบบคิวที่เลือก เช่น ตัวแปรชนิด char มีขนาด 8 บิต ดังนั้นการกำหนดรูปแบบคิว จะต้องใช้จำนวนบิตไม่เกิน 8 บิต เป็นต้น และในส่วนของค่าตัวแปรจะตรวจสอบความเป็นไปได้ของค่าที่กำหนดว่ามีขนาดเกินกว่าชนิดของตัวแปรที่เลือกหรือไม่ เช่น ตัวแปรชนิด char สามารถรับค่าได้ในช่วง [0, 255] เป็นต้น

ขั้นตอนที่ 6. แจ้งเตือนข้อผิดพลาดในขั้นตอนที่ 5 ว่าผิดพลาดที่ส่วนไหน อย่างไร เพื่อทำการแก้ไขใหม่ให้ถูกต้อง

ขั้นตอนที่ 7. การวิเคราะห์โดยทำการหาค่าทีละขั้นเริ่มจากแปลงรูปแบบสมการจาก สัญกรณ์เติมกลาง (Infix Notation) ไปเป็นรูปแบบสัญกรณ์เติมหลัง (Postfix Notation) แล้วจึงดำเนินการหาค่าทีละขั้นตามเงื่อนไข และค่าที่กำหนด ในส่วนของการหาค่าผิดพลาด ใช้ค่าที่ได้จากการคำนวณแบบอิงดรรชนี (Floating-Point) แบบความเที่ยงสองเท่า (Double Precision) ตามมาตรฐาน IEEE754 เป็นค่าความถูกต้องอ้างอิง

ขั้นตอนที่ 8. เป็นการแสดงผลลัพธ์ที่หาได้จากข้อที่ 7 ผลลัพธ์จะแสดงเป็นลำดับเรียงตามขั้นตอนการดำเนินการ มีการแสดงความผิดพลาดของผลลัพธ์ที่ได้เทียบกับค่าความถูกต้องอ้างอิง รายงานในรูปแบบค่าผิดพลาดสัมพัทธ์ และค่าผิดพลาดสัมบูรณ์ รวมถึงมีการแจ้งเตือนในกรณีที่ผลลัพธ์ไม่เป็นไปตามเงื่อนไขในขั้นตอนการดำเนินการนั้น ๆ เช่น ผลลัพธ์ที่ได้เกิดการล้น เป็นต้น และแจ้งเตือนข้อควรระวัง เช่น การหารในส่วนตัวหารต้องไม่มีสมาชิกเป็นค่า 0 โดยทั้งหมดนี้รายงานผลในรูปแบบตาราง

4. ตัวอย่างการวิเคราะห์

ในส่วนนี้จะกล่าวถึงเครื่องมีที่ใช้ในการทดลอง ตัวอย่างการคำนวณ และผลลัพธ์ที่ได้เพื่อทดสอบความถูกต้องของรูปแบบการจำลองการหาค่าผลคำนวณ

โปรแกรมที่ใช้ในการทดลองเขียนขึ้นด้วยภาษาซี โดยใช้คอมไพเลอร์จีซีซี (Gcc) รันบนระบบปฏิบัติการวินโดว์เอกซ์พี (WindowsXP) ตัวอย่างสมการ และข้อมูลที่ใช้ในการทดลอง เป็นสมการ First-order low-pass filter คำนวณหาค่ากระแสควบคุมมอเตอร์ขนาด 1 ถึง 30 แรงม้า ดังต่อไปนี้
สมการคำนวณค่ากระแส

$$Y(k) = (X(k)Ta + Y(k-1)\tau) / (\tau + Ta) \qquad (14)$$

แปลงให้อยู่ในรูปแบบสัญกรณ์เติมหลังได้เป็น

$$Y(k) = X(k)\ Ta\ *\ Y(k-1)\ \tau\ *\ +\ \tau\ Ta\ +\ / \qquad (15)$$

จากสมการนำเข้ากำหนดค่าตัวแปรได้ดังตารางที่ 1 ส่วนการดำเนินการ ผลลัพธ์ และค่าผิดพลาดเป็นดังตารางที่ 2

การเลือกรูปแบบคิวในตัวอย่างนี้เลือกค่าที่น้อยที่สุดที่ทำให้ค่าในรูปแบบคิวไม่เป็นศูนย์ ได้ค่ารูปแบบคิวเท่ากับ 14 และการปัดเศษในการแปลงจากเลขฐานสิบไปอยู่ในรูปแบบคิว เราจะใช้การปัดเศษแบบปกติ คือปัดลงเมื่อทศนิยมหลักแรกมีค่าต่ำกว่า 5 และปัดขึ้นเมื่อทศนิยมหลักแรกมีค่าสูงกว่า 5 ในกรณีทศนิยมหลักแรกมีค่าเท่ากับ 5 จะปัดขึ้นเมื่อเลขจำนวนเต็มหลักหน่วยมีค่าเป็นเลขคี่ แต่ถ้าเป็นเลขคู่จะถูกปัดลง

ตารางที่ 1 กำหนดค่าและรูปแบบให้ตัวแปรในสมการที่ 16

ลำดับ	ชื่อตัวแปร	ค่าปกติ (ก่อนแปลง)	ชนิดตัวแปร	Q-format	ค่าในรูปแบบคิว
1	X(k)	[2,50]	unsigned 8 bits	0	[2,50]
2	Ta	[0.0001,0.001]	unsigned 16 bits	14	[2,16]
3	Y(k-1)	[2,50]	unsigned 8 bits	0	[2,50]
4	τ	[0.0001,0.001]	unsigned 16 bits	14	[2,16]
5	Y(k) ผลลัพธ์	-	unsigned 32 bits	14	-

ตารางที่ 2 ขั้นตอนการดำเนินการ ผลการดำเนินการ และค่าผิดพลาดตามสมการที่ 17

ลำดับ	การดำเนินการ	ผลการคำนวณ			ค่าผิดพลาด		ชื่อผลลัพธ์
		รูปแบบคิว	รูปแบบปกติ	ผลถูกต้องอ้างอิง	สัมบูรณ์	สัมพัทธ์	
1	X(k) * Ta	[4,800]	[0.000244,0.048828]	[0.0002,0.05]	[0.000044, 0.001171]	[0.024,0.1808]	ans1
2	Y(k-1) * τ	[4,800]	[0.000244, 0.048828]	[0.0002,0.05]	[0.000044, 0.001171]	[0.024,0.1808]	ans2
3	ans1 + ans2	[8,1600]	[0.000488, 0.097656]	[0.0004,0.1]	[0.000088, 0.002343]	[0.024,0.1808]	ans3
4	τ + Ta	[4,32]	[0.000244, 0.001953]	[0.0002,0.002]	[0.000044, 0.000046]	[0.024,0.1808]	ans4
5	ans3 / ans4	[4096,6553600]	**[0.25,400]**	**[0.2,500]**	**[0.05,100]**	**[0.2,0.25]**	**result**

จากผลที่ได้ในลำดับการดำเนินการที่ 5 ในตารางที่ 2 จะเห็นว่าเกิดค่าความผิดพลาดสัมพัทธ์มากสุดถึงร้อยละ 25 ซึ่งความผิดพลาดดังกล่าวเกิดจากค่าผิดพลาดควอนไทเซชัน ในขั้นตอนการแปลงค่าเลขฐานสิบไปเป็นเลขรูปแบบคิวในส่วนการกำหนดค่าและรูปแบบตัวแปรในตารางที่ 1 ทำให้เกิดค่าความคลาดเคลื่อนสะสมในการคำนวณ ถ้าเรากำหนดรูปแบบคิวในตารางที่ 1 ให้มีค่ามากกว่า 14 ขึ้นไป ความคลาดเคลื่อนที่เกิดจากการคำนวณในตารางที่ 2 ก็จะมีค่าลดลง แปรผกผันกับค่ารูปแบบคิวที่เพิ่มขึ้น ตัวอย่างเช่น เมื่อเปลี่ยนรูปแบบตัวแปรทั้งหมดจากรูปแบบคิว 14 ไปเป็นรูปแบบคิว 15 จะทำให้ค่าผิดพลาดสัมพัทธ์มากสุดลดลงเหลือเพียงประมาณร้อยละ 10 เป็นต้น

ตัวอย่างที่แสดงเป็นสมการที่พบได้ทั่วไปในงานควบคุมต่าง ๆ และค่าตัวแปรถูกกำหนดให้อยู่ในรูปแบบที่ไม่มีความละเอียดมากนัก แต่ผลการวิเคราะห์ความผิดพลาดที่ได้แสดงให้เห็นว่าในขั้นการคำนวณ ถ้าเราไม่ตรวจสอบค่าและรูปแบบตัวแปรที่เลือกให้ดี อาจเกิดปัญหาที่เราคาดไม่ถึงได้ในผลลัพธ์ของการคำนวณ ดังนั้นเราจึงควรจำลองการคำนวณก่อนที่จะนำค่าไปใช้จริงในการโปรแกรม เพื่อเป็นการป้องกันความผิดพลาดที่อาจเกิดขึ้นได้

5. สรุปและข้อเสนอแนะ

ในบทความนี้เสนอวิธีการตรวจสอบความถูกต้องและความผิดพลาด โดยใช้การจำลองวิธีการคำนวณซึ่งเป็นวิธี

หนึ่งที่ช่วยลดปัญหาในการพัฒนาโปรแกรมให้สะดวกขึ้น นักโปรแกรมสามารถมองเห็นความเป็นไปของการคำนวณว่าเกิดความผิดพลาดขึ้นในส่วนใด ขั้นตอนใดและเพราะสาเหตุใด ทำให้สามารถป้องกัน และลดปัญหาที่อาจจะเกิดขึ้นได้ ถือเป็นการแก้ปัญหาที่ต้นเหตุ เพราะในระบบแบบฝังตัวมีส่วนประกอบหลายส่วน แม้ว่าการทำการแก้จุดบกพร่องแบบทันทีจะเป็นวิธีที่ใช้ได้ผล แต่สาเหตุของปัญหาอาจเกิดจากผลกระทบจากส่วนอื่นที่ไม่เกี่ยวข้องกับการคำนวณ แต่อาจนำมาใช้ต่อในการคำนวณ หรือเกิดความผิดพลาดอื่นที่มีผลกระทบทำให้แยกปัญหาออกจากกันได้ลำบาก และไม่สามารถยืนยันปัญหาที่เกิดได้แน่ชัดว่าเกิดจากส่วนใด และสาเหตุใด จากการทดลองผลที่ได้สามารถนำไปใช้ได้จริง

แนวทางในการพัฒนาต่อไปนั้นจะเป็นส่วนของปัญหาเรื่องข้อจำกัดในการคำนวณหาค่าฟังก์ชันแบบต่าง ๆ ซึ่งอาจจะต้องมีการกำหนดให้เป็นมาตรฐาน หรือจัดทำการวิเคราะห์ความผิดพลาด (Error Analysis) ของแต่ละฟังก์ชันในคลังโปรแกรมที่ใช้ เพื่อสามารถนำไปใช้คำนวณค่าผิดพลาดในการจำลองได้แม่นยำมากขึ้น

6. กิตติกรรมประกาศ

ขอบคุณความช่วยเหลือ และความร่วมมือจากภาควิชาไฟฟ้า สาขาอิเล็กทรอนิกส์กำลัง จุฬาลงกรณ์มหาวิทยาลัย ในคำแนะนำรวมถึงข้อมูลในการโปรแกรมระบบแบบฝังตัว และ Gcc Developer Group ที่ได้ให้คำแนะนำในการใช้งานโปรแกรม Gcc เป็นอย่างดี

7. เอกสารอ้างอิง

[1] Brigitte Verdonk, Annie Cuty, and Dennis Erschaeren, *"A Precision- and Range-Independent Tool for Testing Floating-Point Arithmetic I:Conversions."*, ACM Transactions on Mathematical Software, Vol. 27, No. 1, March 2001, Pages 92–118.

[2] R.E. Boche *"An Operational Interval Arithmetic"* Illinois National Electronics Conference, October 1963