การพัฒนาโปรแกรมควบคุมการทำงานของโพลาริมิเตอร์ด้วยภาษา C#

DEVELOPMENT OF POLARIMETRY MEASUREMENT PROGRAM USING C#

พีรกิตติ์ นาแพง
ภัทรพล เอี่ยมยัง

โครงงานพิเศษนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
ปริญญาวิทยาศาสตรบัณฑิต (ฟิสิกส์ประยุกต์)
ภาควิชาฟิสิกส์ คณะวิทยาศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2561

DEVELOPMENT OF POLARIMETRY MEASUREMENT PROGRAM USING C#

PEERAKIT NAPAENG
PATTARAPON IAMYANG

A SPECIAL PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENT FOR
THE DEGREE OF BACHELOR OF SCIENCE (APPLIED PHYSICS)
DEPARTMENT OF PHYSICS FACULTY OF SCIENCE
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
ACADEMIC YEAR 2018

หัวข้อโครงงานพิเศษ	การพัฒนาโปรแกรมควบคุมการทำงานของโพลาริมิเตอร์ด้วยภาษา C#
	DEVELOPMENT OF POLARIMETRY MEASUREMENT PROGRAM USING C#
ชื่อนักศึกษา	นายพีรกิตติ์ นาแพง รหัสนักศึกษา 58051110
	นายภัทรพล เอี่ยมยัง รหัสนักศึกษา 58051116
ปริญญา	วิทยาศาสตรบัณฑิต (ฟิสิกส์ประยุกต์)
ภาควิชา	ฟิสิกส์
ปีการศึกษา	2561
อาจารย์ที่ปรึกษา	รศ.ดร.วราวุฒิ เถาลัดดา
อาจารย์ที่ปรึกษาร่วม	ผศ.ดร.ศ.ทิพวรรณ คล้ายบุญมี

คณะวิทยาศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง (สจล.) อนุมัติให้โครงงานพิเศษนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรบัณฑิต (ฟิสิกส์ประยุกต์) ประจำปีการศึกษา 2561

คณะกรรมการสอบ	ลายมือชื่อ
ดร.เมตยา กิติวรรณ ประธานกรรมการ	
ดร.ภาณุพล โชลนกระโทก กรรมการ	
รศ.ดร.วราวุฒิ เถาลัดดา กรรมการและอาจารย์ที่ปรึกษา	
ผศ.ดร.ศ.ทิพวรรณ คล้ายบุญมี กรรมการและอาจารย์ที่ปรึกษาร่วม	

หัวข้อโครงงานพิเศษ การพัฒนาโปรแกรมควบคุมการทำงานของโพลาริมิเตอร์ด้วยภาษา C#
ชื่อนักศึกษา นายพีรกิตติ์ นาแพง รหัสนักศึกษา 58051110
นายภัทรพล เอี่ยมยัง รหัสนักศึกษา 58051116
ปริญญา วิทยาศาสตร์บัณฑิต (ฟิสิกส์ประยุกต์)
ภาควิชา ฟิสิกส์
คณะ วิทยาศาสตร์
มหาวิทยาลัย สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง (สจล.)
ปีการศึกษา 2561
อาจารย์ที่ปรึกษา รศ.ดร.วราวุฒิ เถาลัดดา
อาจารย์ที่ปรึกษาร่วม ผศ.ดร.ศ.ทิพวรรณ คล้ายบุญมี

บทคัดย่อ

โครงงานพิเศษนี้มีวัตุประสงค์เพื่อพัฒนาโปรแกรมคอมพิวเตอร์โดยใช้ Visual Studio 2017 C# เป็นโปรแกรมที่พัฒนาใช้ในการควบคุมเครื่องมือ 2 ชนิดคือ ดิจิตอลมัลติมิเตอร์ Agilent รุ่น 34410A และ MMC-2 CHUO SEIKI ไดรฟเวอร์สเตปปิงมอเตอร์ 5 เฟส ซึ่งเป็นส่วนประกอบของโพลาริมิเตอร์ โพลาริมิเตอร์คือเครื่องมือทางวิทยาศาสตร์ใช้ในการกำหนดทิศทางการโพลาไรเซชั่นของแสงที่ผ่านสารที่มีคุณสมบัติทางแสง โพลาริมิเตอร์จะมีแผ่นวิเคราะห์ทิศทางโพลาไรเซชั่นเรียกว่า แผ่นอนาไรซ์ ที่ติดอยู่กับแท่นหมุนโดยใช้สเตปปิ้งมอเตอร์ ความเข้มแสงที่ผ่านจากแผ่นอนาไรซ์จะวัดโดยโฟโต้ดิเทคเตอร์ เอาต์พุตของโฟโต้ดิเทคเตอร์จะมาในรูปแบบสัญญาณแรงดันไฟฟ้าที่แปรผันตรงกับความเข้มแสงจะเชื่อมต่อกับดิจิตอลมัลติมิเตอร์ ความสัมพันธ์ระหว่างมุมที่หมุนไปของแผ่นอนาไรซ์และความเข้มแสงจากโฟโต้ดิเทคเตอร์จะถูกบันทึกโดยโปรแกรมควบคุมที่พัฒนาขึ้น นอกจากนั้นหน้าต่างโปรแกรมจะถูกออกแบบใหม่ให้มีความทันสมัยและใช้งานได้สะดวกแก่ผู้ใช้งาน โปรแกรมยังประกอบไปด้วยกราฟที่แสดงผลออกมาแบบเรียลไทม์ในรูปแบบ Polar plot และ Sinusoidal plot.

คำสำคัญ : โพลาริมิเตอร์, แผ่นอนาไรซ์, โพลาไรเซชั่น

Title	DEVELOPMENT OF POLARIMETRY MEASUREMENT PROGRAM USING C#
Students	Mr. Peerakit Napaeng Student ID 58051110
	Mr. Pattarapon Iamyang Student ID 58051116
Degree	Bachelor of science in (Applied Physics)
Department	Physics
Faculty	Science
Academic Year	2018
Advisor	Assoc.Prof.Dr. Warawoot Thowladda
Co-Advisor	Asst.Prof.Dr. S.Tipawan Khalayboonme

Abstract

The purpose of this special project is to develop a computer program based on the freeware visual studio C#. The developed program was used for controlling and interfacing two instruments; 34410A Model Agilent digital multimeter and MMC-2 CHUO SEIKI 5-Phase stepping motor driver, that are included in a setup of polarimeter. The polarimeter is a scientific instrument for determining the polarization direction of the light that passing through an optically active substance. This polarimeter setup has a polarization analyzer attached on a rotating state driven by the stepping motor. Light intensity passing from the analyzer was measured by a photodetector. The photodetector output in a voltage signal that directly relates to light intensity was connected to the digital multimeter. The relation between the rotation angle of the analyzer and the light intensity from the photodetector was also recorded by the develop computer program via a computer. In addition, the user interface window was designed to comfortable interact with user. The window was also composed of the real-time graphical result in form of polar plot and sinusoidal plot.

Keywords : Polarimeter, Analyzer, Polarization

กิตติกรรมประกาศ

โครงงานพิเศษนี้สามารถสำเร็จลุล่วงไปด้วยดีเนื่องจากได้รับการสนับสนุน ช่วยเหลือและความอนุเคราะห์จากบุคคลหลายท่าน ซึ่งต้องกราบขอบพระคุณไว้ ณ โอกาสนี้ด้วย รศ. ดร. วราวุฒิ เถาลัดดา ผู้ซึ่งถ่ายทอดวิชาความรู้และให้คำปรึกษาในการสร้างเครื่องมือและเขียนโปรแกรม ซึ่งเป็นประโยชน์อย่างมากในการทำโครงงานพิเศษนี้และยังเป็นประโยชน์ในการนำความรู้และเทคนิคต่าง ๆ ไปใช้ในหน้าที่การงานในอนาคตได้อีกด้วย อีกทั้งยังอดทนให้คนไม่ขยันได้ทำงานสำเร็จลุล่วงไปด้วยดีจนเกิดความรู้ความสามารถในด้านใหม่ๆ จึงกราบขอบพระคุณไว้ ณ ที่นี้ด้วย

ขอขอบพระคุณ ผศ. ดร. ศ. ทิพวรรณ คล้ายบุญมี ผู้ซึ่งคอยแนะนำและสนับสนุนการทำโครงงานพิเศษนี้ในทุก ๆ ด้าน ทั้งด้านทักษะความรู้ความสามารถในการทำงานและสนับสนุนเครื่องมือต่าง ๆ การใช้เครื่องมือต่าง ๆ เป็นอย่างดี จึงกราบขอบพระคุณไว้ ณ ที่นี้ด้วย

ขอขอบคุณ คุณณพสิษฐ์ ศรีสุรัตน์ ผู้ให้คำแนะนำและสนับสนับการทำโครงงานพิเศษนี้ตลอดจนช่วยแนะนำกระบวนการต่าง ๆ ในการเขียนโปรแกรมที่ใช้ในการควบคุมการทำงานของโพลาริมิเตอร์ จนทำให้โครงงานพิเศษนี้จนสำเร็จลุล่วงด้วยดี

สุดท้ายนี้ ขอกราบขอบพระคุณบุคคลที่มีความสำคัญที่สุดในชีวิต ได้แก่ บิดา มารดา ปู่ ย่า ตา และยาย ผู้คอยอบรมเลี้ยงดูและสนับสนุนทุกอย่างในทางที่ดี เนื่องจากกำลังใจเหล่านี้จึงทำให้โครงงานพิเศษนี้สามารถสำเร็จลุล่วงไปด้วยดี จึงกราบขอบพระคุณท่านไว้ ณ ที่นี้ด้วย

พีรกิตติ์ นาแพง
ภัทรพล เอี่ยมยัง

สารบัญ

	หน้า
บทคัดย่อภาษาไทย	ก
บทคัดย่อภาษาอังกฤษ	ข
กิตติกรรมประกาศ	ค
สารบัญ	ง
สารบัญตาราง	ฉ
สารบัญรูป	ช
คำย่อและสัญลักษณ์	ฌ
บทที่ 1 บทนำ	1
1.1 ความเป็นมาของงานวิจัย	1
1.2 วัตถุประสงค์ของงานวิจัย	1
1.3 ขอบเขตของงานวิจัย	1
1.4 ขั้นตอนการวิจัยและวิธีดำเนินงาน	2
1.5 ประโยชน์ที่คาดว่าจะได้รับ	2
บทที่ 2 ทฤษฎีที่เกี่ยวข้อง	3
2.1 Polarimetry	3
2.1.1 การโพลาไรซ์ของแสง	3
2.1.2 การทำให้เป็นแสงโพลาไรซ์โดยการดูดกลืน	4
2.1.3 Half Waveplate	7
2.1.4 Optical Activity	8
2.1.5 โพลาริมิเตอร์ (Polarimeter)	9
2.1.6 Specific Rotation	10
2.2 Visual Studio 2017 (C#)	12
บทที่ 3 วิธีการดำเนินงานวิจัย	14
3.1 Hardware System	14
3.1.1 ขั้นตอนการสร้างระบบโพลาริมิเตอร์	14
3.1.2 ระบบขับเคลื่อน	15

สารบัญ (ต่อ)

หน้า

3.2 Software System 17
3.2.1 แผนผังการทำงานของโปรแกรม 21
บทที่ 4 ผลการวิจัยและการอภิปายผล 22
4.1 ผลการทำงานของโปรแกรม Polarimeter 22
4.2 ตัวอย่างผลการทดลอง 25
บทที่ 5 สรุปผลการวิจัยและการข้อเสนอแนะ 29
5.1 สรุปผลการทดลอง 29
5.1.1 ผลของขั้นตอนการเชื่อมต่อกับ Digital Multimeter 29
5.1.2 ผลของขั้นตอนการเก็บค่าการทดลองเพื่อใช้ในการสร้างกราฟ 29
5.1.3 ผลการทดลองเมื่อปรับแผ่น Analyzer โดยไม่ใช้โปรแกรม 30
5.1.4 ผลการทดลองเมื่อปรับแผ่น Analyzer โดยใช้โปรแกรม 30
5.2 ข้อเสนอแนะ 30
เอกสารอ้างอิง 31
ภาคผนวก 32
ภาคผนวก ก 33
ภาคผนวก ข 35

สารบัญตาราง

ตารางที่	หน้า
1.1 ขั้นตอนการวิจัยและการดำเนินงาน	2
2.1 Specific rotation ของสารละลาย	11
2.2 ผลของความยาวคลื่นที่มีต่อค่า Specific rotation ของ sucrose	12
4.1 บันทึกค่ามุมที่แสงมีความเข้มต่ำสุดตามมุมต่าง ๆ ของ Half Waveplate	26
4.2 บันทึกค่ามุมที่แสงมีความเข้มต่ำสุดตามมุมต่าง ๆ ของ Half Waveplate (โดยใช้เครื่องคอมพิวเตอร์)	27

สารบัญรูป

รูปที่	หน้า
2.1 ทิศทางการแกว่งของสนามไฟฟ้าและสนามแม่เหล็กของคลื่นแม่เหล็กไฟฟ้า	3
2.2 แสงโพลาไรซ์เชิงเส้นหรือแสงโพลาไรซ์เชิงระนาบ	4
2.3 แสงโพลาไรซ์เชิงเส้นในแนวเฉียงที่เกิดจากสนามไฟฟ้าในแนวดิ่งและแนวนอนที่มีเฟสตรงกัน	4
2.4 คลื่นในเส้นเชือกจะเดินทางผ่านช่องเปิดแคบ ๆ ได้เมื่อคลื่นในเส้นเชือกแกว่งในแนวเดียวกับช่องเปิด	5
2.5 ในความจริงแสงโพลาไรซ์ในแนวตั้งฉากกับแนวโมเลกุลจะไม่ถูกดูดกลืนและสามารถผ่านแผ่นโพลาไรซ์ได้	5
2.6 แนวโพลาไรซ์ของแสงที่เกิดจากแผ่นโพลาไรซ์	6
2.7 แนวโพลาไรซ์ของแสงทำมุม θ กับแกนโพลาไรซ์	6
2.8 แนวโพลาไรซ์ของแสงทำมุม θ กับแกนโพลาไรซ์ของโพลาไรเซอร์ (แผ่นโพลาไรซ์ที่อยู่ใกล้แหล่งกำเนิดแสง) องค์ประกอบของสนามไฟฟ้าที่ผ่านแผ่นแอนาไลเซอร์ (แผ่นโพลาไรซ์ที่อยู่ ใกล้ผู้สังเกตุ) มีค่า $E_{\circ}\cos\theta$	7
2.9 เมื่อแสงตกกระทบมีแนวโพลาไรซ์ทำมุม θ กับแกนแสงของ half waveplate จะได้แสงโพลาไรซ์เชิงเส้นที่มีแนวโพลาไรซ์ทำมุม 2θ กับแนวโพลาไรซ์ของแสงตกกระทบ	8
2.10 โมเลกุลมือซ้ายและมือขวาจะมีสมบัติ optical activity	9
2.11 โครงสร้างของโพลาริมิเตอร์	10
2.12 แสดงตัวอย่างการรันโปรแกรม Visual Studio 2017 (C#)	12
2.13 แสดงตัวอย่างการเขียนคำสั่งโปรแกรม Visual Studio C#	13
3.1 การจัดวางอุปกรณ์ของ Polarimeter	14
3.2 MMC-2 5 Phase stepping motor driver	15
3.3 5 Phase stepping motor	16
3.4 USB/GPIB Interface	16
3.5 หม้อแปลงไฟฟ้ากระแสสลับ 220 V เป็น 110 V	17
3.6 USB/GPIB สามารถเชื่อมต่อกับคอมพิวเตอร์ได้	18
3.7 แสดงตัวอย่างการเชื่อมต่อกับ Digital Multimeter	18
3.8 ด้านซ้ายมือคือ Toolbox หรือกล่องเครื่องมือที่ใช้สร้าง User Interface(UI)	19

สารบัญรูป (ต่อ)

รูปที่	หน้า
3.9 โปรแกรม Polarimeter2019 ด้านซ้ายคือส่วนควบคุม และด้านขวาคือกราฟ	19
3.10 Code คำสั่งของโปรแกรม Polarimeter2019	20
3.11 Code คำสั่งของโปรแกรม Polarimeter2019	20
3.12 แผนผังการทำงานของโปรแกรม Polarimeter	21
4.1 แผนผังการใช้งานของโปรแกรม Polarimeter	22
4.2 ขั้นตอนการใส่ชื่อการทดลอง ความละเอียดในการวัด และจำนวน Sample	23
4.3 แสดง Reference และ Sample ในการทดลอง	23
4.4 ขั้นตอนการเชื่อมต่อของ Motor Driven และ Digital Multimeter	24
4.5 ขั้นตอนการกำหนดจุด Start Stop และ Resolution	24
4.6 แสดงตำแหน่งจุดเริ่มต้นของการทดลอง	24
4.7 ขั้นตอนการเลือก Sample ที่ทำการวัด	25
4.8 มุมของแผ่น Half Waveplate	26
4.9 การทดลองโดยใช้โปรแกรม Polarimeter2019 (แบบ Polar chart)	27
4.10 การทดลองโดยใช้โปรแกรม Polarimeter2019 (แบบ Sine wave)	28

คำย่อและสัญลักษณ์

คำย่อ/สัญลักษณ์	คำอธิบาย
θ	มุม (องศา)
E_0	ค่าสนามแม่เหล็กไฟฟ้า
α	มุมที่แนวโพลาไรซ์ปิดไป
T	อุณหภูมิ (องศาเซลเซียส)
λ	ความยาวคลื่น
l	ความยาวของเส้นทางเดินแสงในสารละลาย
c	ความเข้มข้นของสารละลาย
$HeNe$	แสงเลเซอร์

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในปัจจุบันนั้นมีเทคโนโลยีที่ช่วยในการตรวจสอบมุมของการหมุนของแสง (Optical Rotation) ของระนาบโพลาไรซ์ของแสงที่วิ่งผ่านตัวกลางชิ้นหนึ่งซึ่งหลักการที่ใช้ในการทดลองก็คือเราจะทำการวัดแนวของแสงโพลาไรซ์ที่บิดไปเมื่อผ่านสารละลาย ซึ่งเราได้นำประโยชน์ของทฤษฎีดังกล่าวมาทำการประยุกต์ใช้ในการวัดความเข้มข้นของสารละลาย

เนื่องจากโปรแกรมที่ทำการควบคุมโพลาริมิเตอร์นั้นอาศัยการทำงานบนระบบปฏิบัติการ Windows 7 แต่ตัวโปรแกรมนั้นไม่สามารถทำงานบนระบบปฏิบัติการ Windows 10 ได้ ดังนั้นพวกเราจึงทำการเขียนโปรแกรมขึ้นมาใหม่โดยอ้างอิงจากโปรแกรมเดิมเพื่อให้โปรแกรมสามารถทำงานบน Windows 10 ได้อย่างสมบูรณ์พร้อมทั้งปรับรูปแบบของโปรแกรมขึ้นมาใหม่และเพิ่มฟังก์ชันบางอย่างให้โปรแกรมมีความน่าใช้งานมากยิ่งขึ้น ทั้งนี้ประสิทธิภาพในการทำงานของโปรแกรมต้องมีความสมบูรณ์แบบไม่มีความผิดพลาดเกิดขึ้นเมื่อทำการทดลอง

1.2 วัตถุประสงค์ของงานวิจัย

1) เพื่อให้โปรแกรมสามารถทำงานบนระบบปฏิบัติการ Windows 10 ได้อย่างสมบูรณ์
2) เรียนรู้การเขียนโปรแกรม Visual Studio C# 2017
3) เรียนรู้หลักการทำงานของโพลาริมิเตอร์

1.3 ขอบเขตของงานวิจัย

1) ศึกษาถึงทฤษฎีที่เกี่ยวข้องกับโพลาริมิเตอร์
2) ศึกษาการเขียนโปรแกรมด้วยโปรแกรม Visual Studio 2017 โดยการใช้ภาษา C# ในการควบคุมเครื่องมือรวมทั้งออกแบบหน้าตาการใช้งานโปรแกรมแบบใหม่
3) ศึกษาการทำงานของโพลาริมิเตอร์ร่วมกับโปรแกรม Visual Studio 2017 โดยมี Motor driver MMC-2 และ Digital multimeter เป็นตัวเชื่อมต่อ

1.4 ขั้นตอนการวิจัย

ตารางที่ 1.1 ขั้นตอนการวิจัยและการดำเนินงาน

กิจกรรม	ระยะเวลา					
	ธันวาคม	มกราคม	กุมภาพันธ์	มีนาคม	เมษายน	พฤษภาคม
1) ศึกษาการเขียนโปรแกรมและการใช้เครื่องมือโพลาริมิเตอร์						
2) เริ่มปฏิบัติการเขียนโปรแกรม Visual Studio 2017 C#						
3) แก้ไขและปรับแต่งโปรแกรมให้ดียิ่งขึ้น						
4) สรุปและวิเคราะห์ผล						

1.5 ประโยชน์ที่คาดว่าจะได้รับ

สามารถเขียนโปรแกรม Visual Studio 2017 C# ได้อย่างมีประสิทธิภาพและสามารถทำงานบนระบบปฏิบัติการ Windows 10 ได้อย่างสมบูรณ์รวมทั้งรู้ถึงหลักการทำงานของโพลาริมิเตอร์และสามารถใช้โปรแกรมควบคุมเครื่องมือได้อย่างสมบูรณ์

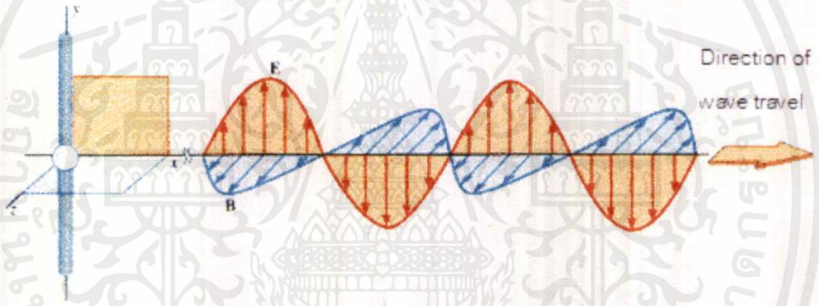
บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 Polarimetry

Polarimetry เป็นวิธีการวัดความเข้มข้นของสารละลายโดยการวัดแนวแสงโพลาไรซ์ที่บิดไปเมื่อผ่านสารละลาย เครื่องที่ทำการวัดด้วยวิธีนี้เรียกว่า Polarimeter

2.1.1 การโพลาไรซ์ของแสง (Polarization of Light)

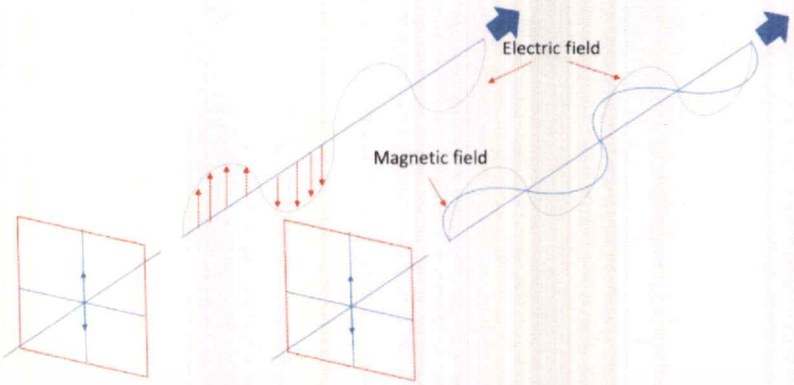
การโพลาไรซ์เป็นสมบัติของคลื่นแม่เหล็กไฟฟ้า เช่น แสง ที่อธิบายทิศทางการแกว่งของสนามไฟฟ้าตามขวางของคลื่นแม่เหล็กไฟฟ้าซึ่งตั้งฉากกับทิศทางการเคลื่อนที่ของคลื่น (คลื่นตามยาว เช่น คลื่นเสียง ไม่มีสมบัติการโพลาไรซ์ เนื่องจากการแกว่งมีทิศทางเดียวกับกาเคลื่อนที่ของคลื่น)



รูปที่ 2.1 ทิศทางการแกว่งของสนามไฟฟ้าและสนามแม่เหล็กของคลื่นแม่เหล็กไฟฟ้า

โพลาไรซ์เชิงเส้นหรือโพลาไรซ์เชิงระนาบ (Linear or plane polarization)

ถ้าสนามไฟฟ้าของคลื่นแสงมีการแกว่งในระนาบใดระนาบหนึ่ง จะเรียกคลื่นแสงนี้เป็น แสงโพลาไรซ์เชิงเส้นหรือแสงโพลาไรซ์เชิงระนาบ (Linear or plane polarized light) ดังรูปที่ 2.2 - 2.3 ถ้าสนามไฟฟ้าของคลื่นแสงมีการแกว่งในทิศทางต่างๆกันและมีเฟสต่างกันอย่างสุ่ม (randomly in time) เรียกว่าเป็นแสงไม่โพลาไรซ์หรือแสงโพลาไรซ์แบบสุ่ม (unpolarized light or random polarized light)

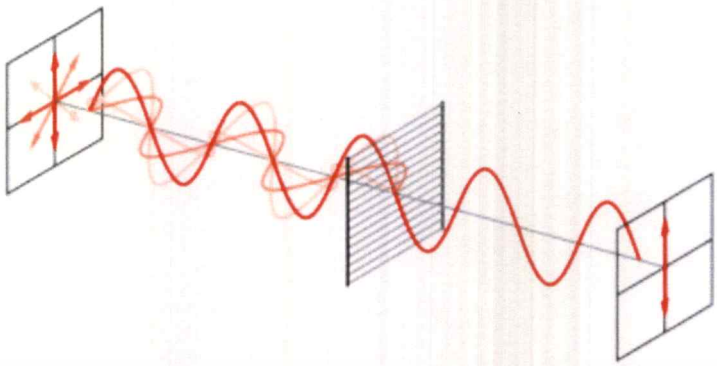


รูปที่ 2.2 แสงโพลาไรซ์เชิงเส้นหรือแสงโพลาไรซ์เชิงระนาบ

รูปที่ 2.3 แสงโพลาไรซ์เชิงเส้นในแนวเฉียงที่เกิดจากสนามไฟฟ้าในแนวดิ่งและแนวนอนที่มีเฟสตรงกัน

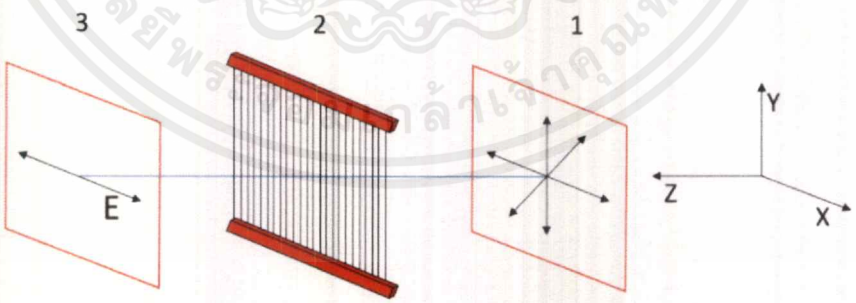
2.1.2 การทำให้เป็นแสงโพลาไรซ์โดยการดูดกลืน (absorption or dichroism)

Dichroism เป็นสมบัติที่วัสดุเลือกดูดกลืนแสงเฉพาะแนวใดแนวหนึ่งของแสงโพลาไรซ์โดยให้อีก แนวหนึ่งสามารถทะลุผ่านไปได้ คล้ายกับคลื่นในเส้นเชือกที่สามารถเดินทางผ่านช่องเปิดแคบ ๆ ได้เมื่อ คลื่นในเส้นเชือกแกว่งในแนวเดียวกับช่องเปิดนั้นและจะผ่านไม่ได้ถ้าแกว่งในแนวขวางกับช่องเปิด ดังรูปที่ 2.4

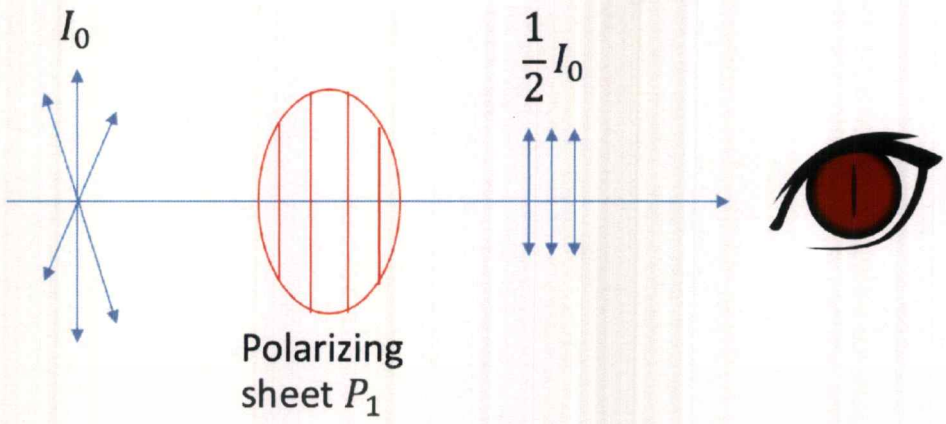


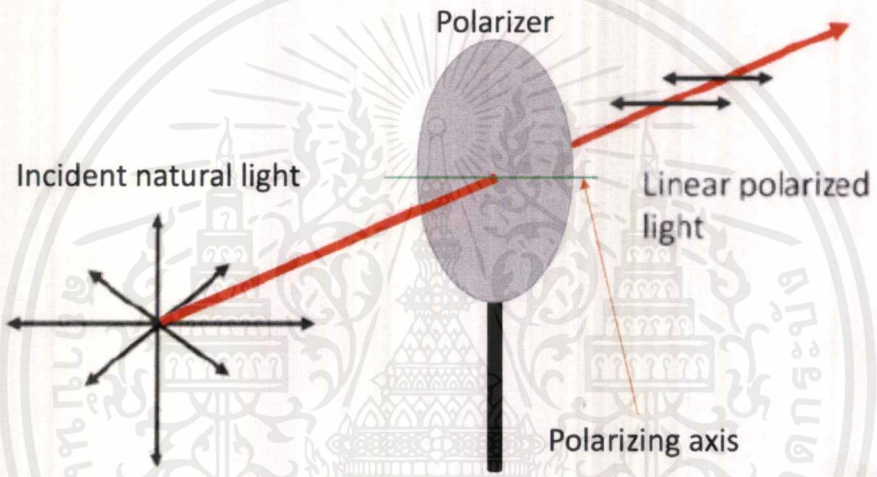
รูปที่ 2.4 คลื่นในเส้นเชือกจะเดินทางผ่านช่องเปิดแคบ ๆ ได้เมื่อคลื่นในเส้นเชือกแกว่งในแนวเดียวกับช่องเปิด

แผ่นโพลาไรซ์หรือโพลาไรเซอร์ (polarized plate or polarizer) (โพลารอยด์ เป็นชื่อทางการค้าของ แผ่นโพลาไรซ์) เป็นตัวอย่างหนึ่งของวัสดุ dichroic (dichroic material) ทำจากสารอินทรีย์ (organic material) ที่มีโมเลกุลเป็นสายยาวคล้ายโครงสร้างดังรูปที่ 2.5 แสงที่มีแนวโพลาไรซ์ในทิศขนานกับแนวยาว ของโมเลกุลจะถูกดูดกลืน ส่วนแนวโพลาไรซ์ในทิศตั้งฉากกับแนวยาวของโมเลกุลจะสามารถผ่านไปได้โดย ไม่ถูกดูดกลืน ดังนั้นแสงที่ผ่านแผ่นโพลาไรซ์จึงมีแนวโพลาไรซ์ตั้งฉากกับแนวโมเลกุลของแผ่นโพลาไรซ์ จะ เรียกแกนที่ตั้งฉากกับแนวโมเลกุลของแผ่นโพลาไรซ์หรือแกนที่ให้แสงโพลาไรซ์ผ่านได้ว่า แกนโพลาไรซ์ (polarizing axis หรือ transmission axis) ดังนั้นแสงที่ผ่านแผ่นโพลาไรซ์จะมีแนวโพลาไรซ์ขนานกับแกน โพลาไรซ์ ดังรูปที่ 2.6

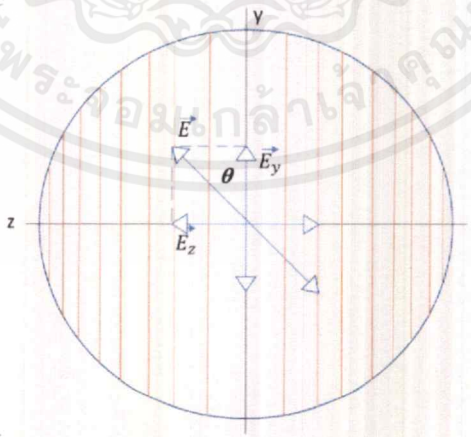


รูปที่ 2.5 ในความจริงแสงโพลาไรซ์ในแนวตั้งฉากกับแนวโมเลกุลจะไม่ถูกดูดกลืนและสามารถผ่านแผ่นโพลาไรซ์ได้





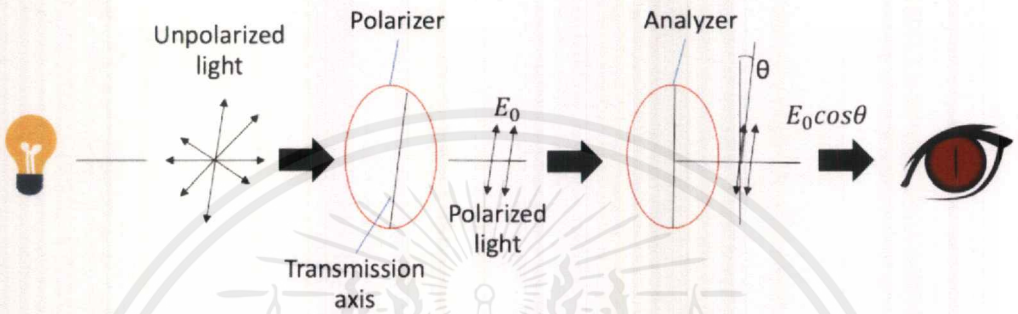
รูปที่ 2.6 แนวโพลาไรซ์ของแสงที่เกิดจากแผ่นโพลาไรซ์



รูปที่ 2.7 แนวโพลาไรซ์ของแสงทำมุม θ กับแกนโพลาไรซ์

ถ้าแสงโพลาไรซ์ตกทำมุม θ กับแกนแสงโพลาไรซ์ ดังรูปที่ 2.7 เวคเตอร์สนามไฟฟ้าของแสงที่ถูกแตกมาขนาดกับแนวแกนโพลาไรซ์ คือ $E_{\circ}\cos\theta$ ดังรูปที่ 2.8 ดังนั้น ความเข้มแสงที่ผ่านแผ่นโพลาไรซ์ จะมีค่า ดังนี้

$$I = E_{\circ}^{2}\cos^{2}\theta = I_{\circ}\cos^{2}\theta \qquad (2.1)$$

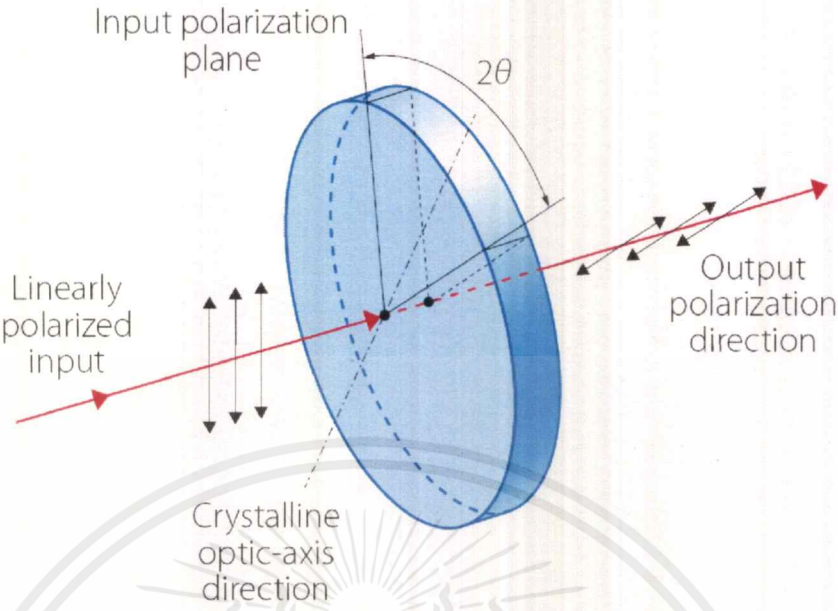


รูปที่ 2.8 แนวโพลาไรซ์ของแสงทำมุม θ กับแกนโพลาไรซ์ของโพลาไรเซอร์ (แผ่นโพลาไรซ์ที่อยู่ใกล้แหล่งกำเนิดแสง) องค์ประกอบของสนามไฟฟ้าที่ผ่านแผ่นแอนาไลเซอร์ (แผ่นโพลาไรซ์ที่อยู่ใกล้ผู้สังเกตุ) มีค่า $E_{\circ}\cos\theta$

ถ้าวางแกนโพลาไรซ์ของแผ่นโพลาไรซ์สองแผ่นทำมุมกัน ความเข้มของแสงจะลดลงคำนวณได้โดย กฎของมาลูส์ เมื่อแกนโพลาไรซ์ของแผ่นโพลาไรซ์สองแผ่นทำมุมกัน 90 องศาแสงจะผ่านแผ่นโพลาไรซ์นี้ไม่ได้

2.1.3 Half Waveplate

Half Waveplate เป็นอุปกรณ์ทางแสงที่ทำหน้าที่บิดแนวโพลาไรซ์ของแสง สร้างจากวัสดุที่มีดัชนี หักเหสองค่า (birefringence) เช่น ผลึกควอตซ์ ในผลึกจะแนวอ้างอิงแนวหนึ่งเรียกว่า แกนแสง (optical axis) ซึ่งเป็นแกนที่เมื่อแสงที่มีแนวโพลาไรซ์ต่างกันเดินทางไปตามทิศทางของแกนแสงแล้วจะมีความเร็ว เท่ากัน (โดยทั่วไปแสงที่มีแนวโพลาไรซ์ต่างกันเมื่อเดินทางผ่านผลึก birefringence จะมีความเร็วต่างกัน) ถ้าแนวโพลาไรซ์ของแสงตกกระทบทำมุม θ กับแนวแกนแสงแล้ว แสงที่ผ่าน half waveplate ออกมาจะ เป็นแสงโพลาไรซ์เชิงเส้นที่มีแนวโพลาไรซ์ทำมุม 2θ กับแนวโพลาไรซ์ของแสงตกกระทบ ดังรูปที่ 2.10 หรือ กล่าวว่าแนวโพลาไรซ์ของแสงบิดไปทำมุม θ กับแนวแกนแสงในทิศตรงข้ามกับแนวโพลาไรซ์ของแสงตกกระทบ



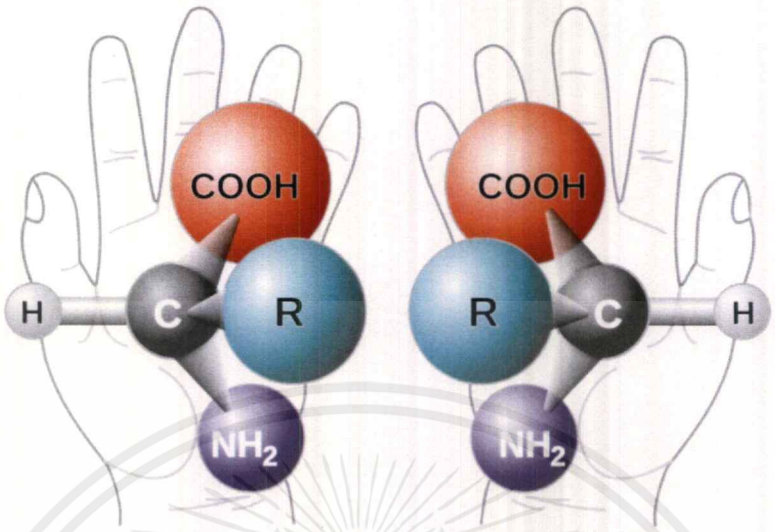
รูปที่ 2.9 เมื่อแสงตกกระทบมีแนวโพลาไรซ์ทำมุม θ กับแกนแสงของ half waveplate จะได้แสงโพลาไรซ์เชิงเส้นที่มีแนวโพลาไรซ์ทำมุม 2θ กับแนวโพลาไรซ์ของแสงตกกระทบ

2.1.4 Optical Activity

ผลึกบางชนิดรวมทั้งของเหลวบางชนิด มีสมบัติในการบิดแนวโพลาไรซ์ของแสงได้ สมบัติทางกายภาพนี้เรียกว่า optical activity หรือ optical rotation ตัวอย่างเช่น แสงโพลาไรซ์เชิงเส้น เมื่อตกตั้งฉาก กับแผ่นผลึก quartz ที่ถูกตัดในระนาบที่ตั้งฉากกับแกนแสง แสงที่ผ่านผลึกออกมายังคงเป็นแสงโพลาไรซ์ เชิงเส้นแต่จะมีแนวของสนามไฟฟ้าหรือแนวโพลาไรซ์บิดไป

ระนาบของแสงโพลาไรซ์จะบิดตามเข็มนาฬิกา (สังเกตโดยมองไปยังทิศของแหล่งกำเนิดแสง) ถ้า ผลึกนั้นเป็นผลึกชนิดมือขวา (right-handed หรือ dextrorotatory crystal) และจะบิดทวนเข็มนาฬิกาถ้า ผลึกนั้นเป็นผลึกชนิดมือซ้าย (left-handed หรือ laevorotatory crystal)

วัสดุที่มีสมบัติ optical activity จะมีโครงสร้างผลึกที่เป็นเกลียว เช่น ผลึก quartz หรือมีโมเลกุลที่ ประกอบด้วยคาร์บอนซึ่งจับอยู่กับอะตอมหรือกลุ่มของอะตอมที่แตกต่างกันสี่ชนิด ซึ่งจะมีโครงสร้างได้สอง แบบคือแบบมือซ้ายและแบบมือขวา ดังรูปที่ 2.11 โครงสร้างโมเลกุลทั้งสองนี้เป็นเสมือนภาพสะท้อนกระจก ซึ่งกันและกัน โมเลกุลทั้งสองนี้จะไม่สามารถซ้อนทับได้ ไม่ว่าจะหมุนหรือเลื่อนตำแหน่งอย่างไร เรียกว่า โมเลกุลลักษณะนี้ว่าโมเลกุลไคร้ล (chiral molecule, chiral หมายถึงมือ) (เช่น มือขวาจะเหมือนกับภาพ ในกระจกของมือซ้าย ซึ่งมือทั้งสองข้างไม่สามารถซ้อนทับกันได้ โดยให้นิ้วโป้งและนิ้วก้อยทับซึ่งกันและกัน เมื่อหันฝ่ามือไปทางเดียวกัน) วัสดุที่มีโครงสร้างโมเลกุลแบบนี้จะมีสมบัติ optical activity

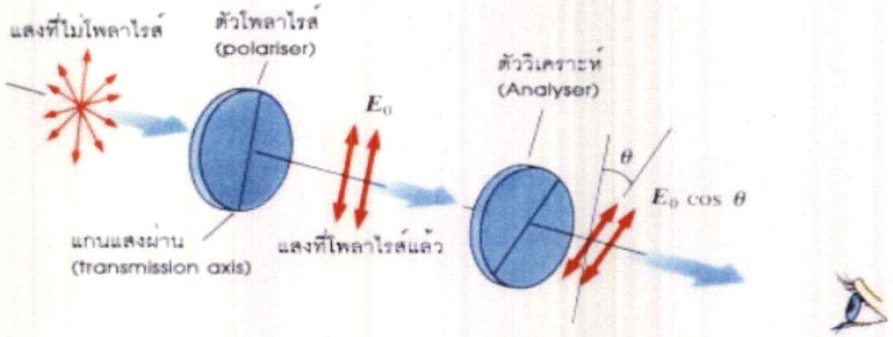


รูปที่ 2.10 โมเลกุลมือซ้ายและมือขวาจะมีสมบัติ optical activity

สมบัติ optical activity อธิบายได้โดยการพิจารณาว่า แสงโพลาไรซ์เชิงเส้นประกอบขึ้นด้วยแสง โพลาไรซ์วงกลมสองลำที่มีทิศทางการหมุนตรงกันข้าม (หมุนขวาและหมุนซ้าย) และในผลึกที่มีสมบัติ optical activity นี้ ดัชนีหักเหของแสงโพลาไรซ์วงกลมชนิดหมุนขวาและชนิดหมุนซ้ายมีค่าต่างกัน คือ n_r และ n_l ตามลำดับ ดังนั้น ความเร็วของแสงโพลาไรซ์วงกลมชนิดหมุนขวาจึงไม่เท่ากับแสงโพลาไรซ์วงกลม ชนิดหมุนซ้าย จึงเกิดความต่างเฟสเมื่อเดินทางผ่านผลึก (ขึ้นกับความหนาของผลึก) เมื่อพิจารณาแสง โพลาไรซ์เชิงเส้นที่เกิดจากแสงโพลาไรซ์วงกลมที่มีเฟสต่างกันนี้ จะได้แสงโพลาไรซ์เชิงเส้นที่มีแนวโพลาไรซ์ บิดไป โดยจะบิดไปทางขวาหรือซ้ายขึ้นกับผลึกว่ามี n_r มากหรือน้อยกว่า n_l

2.1.5 โพลาริมิเตอร์ (Polarimeter)

ของเหลวบางชนิดมีสมบัติ optical activity เช่น สารละลายน้ำตาล (เช่น น้ำเชื่อมและน้ำผลไม้ เป็นต้น) มุมที่บิดไปของแนวโพลาไรซ์ขึ้นกับความเข้มข้นของสารละลาย ถ้าสามารถวัดมุมของแนว โพลาไรซ์ที่บิดไปได้ก็สามารถหาความเข้มข้นของสารละลายได้ เครื่องมือประเภทนี้เรียกว่า โพลาริมิเตอร์ (polarimeter) ซึ่งมีโครงสร้างดังรูปที่ 2.12



รูปที่ 2.11 โครงสร้างของโพลาริมิเตอร์

2.1.6 Specific Rotation

Specific rotation [α] ของสารละลายที่มีสมบัติ optical activity นิยามว่าคือ มุมที่แนวโพลาไรซ์ ของแสงโพลาไรซ์เชิงเส้นที่บิดไปหรือหมุนไปเมื่อเดินทางผ่านสารละลายที่มีความเข้มข้น 1 g/mL บรรจุใน ภาชนะที่มีความยาว (เส้นทางเดินแสงในสารละลาย) 1 decimeter (1/10 m = 10 cm) specific rotation ของสารละลายบริสุทธิ์ (เช่น สารละลายกลูโคสในน้ำกลั่น) เป็นค่าเฉพาะตัว การบิดไปของแนวโพลาไรซ์ขึ้นกับตัวแปรต่าง ๆ ดังนี้

1) ชนิดหรือธรรมชาติของสารละลาย (เช่น สารละลายซูโคส หรือสารละลายกลูโคส)

2) ความเข้มข้นขององค์ประกอบ optical activity ในสารละลาย

3) ความยาวของเส้นทางเดินแสในสารละลาย

4) ความคลื่นของแหล่งกำเนิดแสง

5) อุณหภูมิของสารตัวอย่าง

จากตัวแปรต่าง ๆ ข้างต้นจะได้ว่า

$$[\alpha]_{\lambda}^{T} = \frac{\alpha}{l \times C} \qquad (2.2)$$

$[\alpha]_{\lambda}^{T}$ เป็นค่าที่ความยาวคลื่น λ ที่อุณหภูมิ T มีหน่วยเป็น degree $cm^3/g\ dm$

เมื่อ α คือมุมที่แนวโพลาไรซ์ปิดไป มีหน่วยเป็น (องศา)

C คือความเข้มข้นของสารละลาย มีหน่วยเป็น g/cm^3

l คือความยาวของเส้นทางเดินแสงในสารละลาย (ความยาวของเซลล์สารละลายมีหน่วยเป็น dm)

โดยทั่วไป specific rotation เป็นค่าที่วัดที่อุณหภูมิเฉพาะ T ซึ่งปกติคือ 20° ที่ความยาวคลื่นของเส้นสเปกตรัม D (D line) ของหลอดโซเดียม ซึ่งมีความยาวคลื่น 589 nm ในกรณีนี้จะเขียน specific rotation เป็น $[\alpha]_D^{20^0}$ สารบางชนิดทำให้แนวโพลาไรซ์บิดไปในทิศทางตามเข็มนาฬิกา (เมื่อสังเกตโดยมองไปยังทิศทางแหล่งกำเนิดแสง) กำหนดให้ α มีค่าเป็น + สารบางชนิดทำให้แนวโพลาไรซ์บิดไปในทิศทางทวนเข็มนาฬิกา กำหนดให้ α มีค่าเป็น - ดังแสดงในตารางที่ 2.1

ตารางที่ 2.1 specific rotation ของสารละลาย

สารละลายในน้ำ	$[\alpha]_D^{20^0}(cm^3 / g\, dm)$
Sucrose	+66.54
Glucose	+52.74
Fructose	-93.78
Maltose	+137.5
Lactose	+55.3
Dextrose	+194.8

ตัวอย่างเช่น น้ำตาลทราย (sucrose) ความเข้มข้น 0.5 g/ml บรรจุในเซลล์ใส่สารละลายที่มีความกว้าง 1 cm (เส้นทางเดินแสงในสารละลาย 1 cm) หรือ 0.1 dm แนวโพลาไรซ์จะบิดไปเท่ากับ 66.54 x 0.5 x 0.1 =

$$\alpha = (66.54\ cm^3/g\ dm) \times (0.5 g/cm^3) \times (0.1 dm) = 3.33^\circ \qquad (2.3)$$

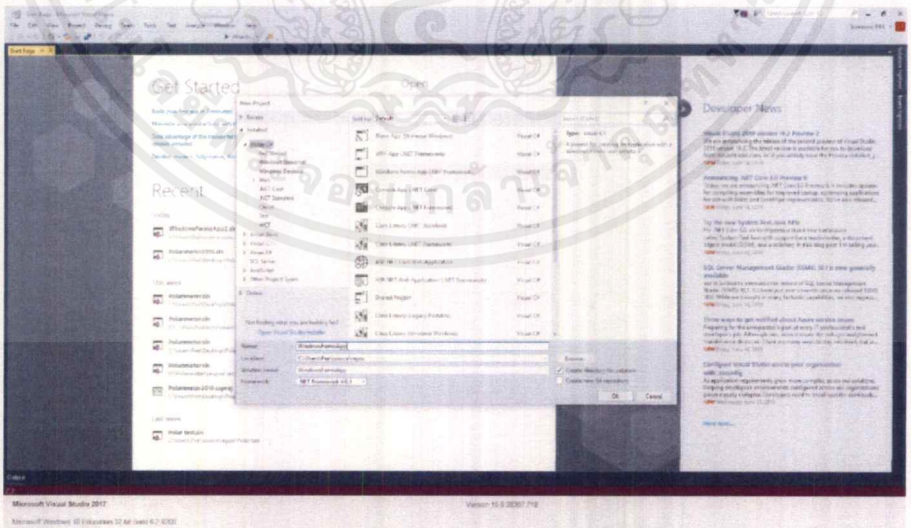
ผลของความยาวคลื่นที่มีต่อค่า specific rotation ของ sucrose แสดงดังตารางที่ 2.2

ตารางที่ 2.2 ผลของความยาวคลื่นที่มีต่อค่า specific rotation ของ sucrose

แหล่งกำเนิดแสง	ความยาวคลื่น (nm)	$[\alpha]_D^{20^0}(cm^3 / g\ dm)$
Mercury, Green	546.23	+78.4178
Sodium, yellow	589.44	+66.5885
HeNe Laser	632.80	+57.2144
Near Infrared(NIR)	882.60	+28.5462

2.2 Visual Studio 2017 (C#)

Visual C# เป็นภาษาเขียนโปรแกรมแบบ multi-paradigm ซึ่งมีรูปแบบภาษาที่ตายตัว และเป็นรูปแบบบังคับในการเขียน มีฟังก์ชัน และยังเป็นภาษาการเขียนโปรแกรมที่มีคุณสมบัติเป็นแบบ Objects ด้วย ซึ่งเราจะทำการเขียนโปรแกรมด้วย .NET framework ซึ่งเป็น Library Software ที่พัฒนาโดยบริษัท Microsoft เพื่อให้สร้าง Application ประเภทต่างๆ ภายใต้ระบบปฏิบัติการ Windows บิดาของภาษา C# คือ Anders Hejlsberg โดยมีรากฐานมาจากภาษา C++ และภาษาอื่นๆ โดยเลือกเขียนด้วย .NET Windows Form Application ที่เขียนโปรแกรมบน Windows Form Application ด้วย .NET Framework ใน Application บน Visual Studio นั้น Windows Form ถือเป็น Project พื้นฐานที่สามารถพัฒนาโปรแกรมที่ทำงานบน Windows ได้ง่ายและรวดเร็วที่สุดก็ว่าได้ เพราะเป็นการออกแบบรูปแบบ GUI การใส่ Control หรือกำหนด Event ต่าง ๆ ก็สามารถสร้างเหตุการณ์ต่าง ๆ ได้จาก Properties ของ Control

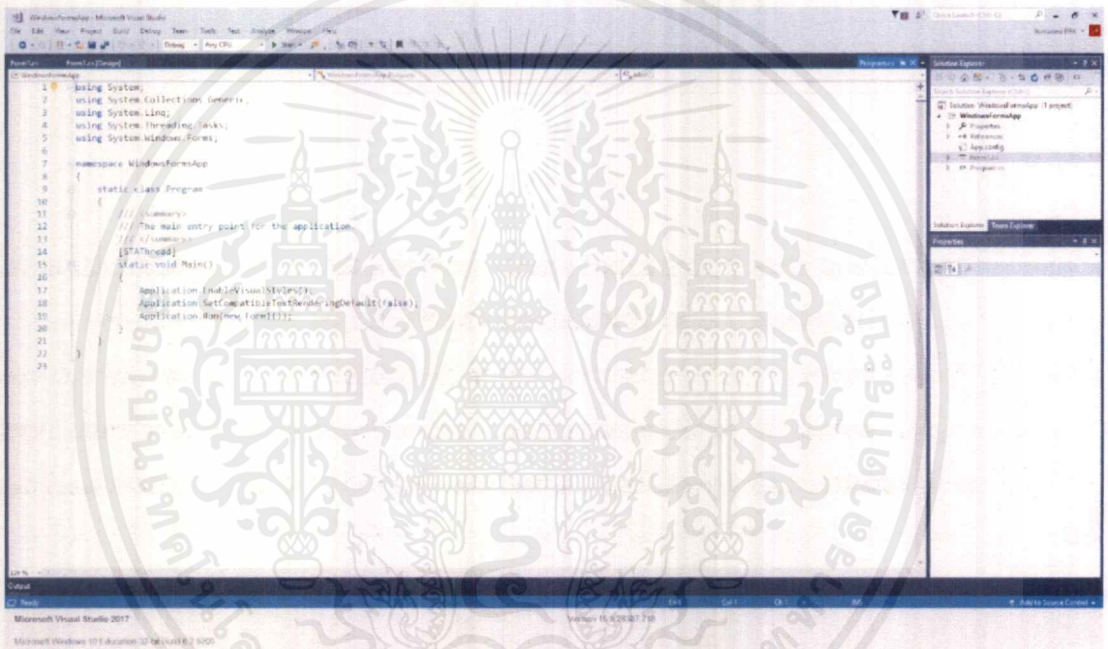


รูปที่ 2.12 แสดงตัวอย่างโปรแกรม Visual Studio 2017 บน .NET Windows Form Application

ซึ่งในปัจจุบันได้พัฒนาและปรับรูปแบบของ ภาษา C# อยู่ตลอดเวลาโดยทาง Microsoft ได้นำภาษา C# ไปอยู่ในชุดพัฒนา software อย่าง visual studio ซึ่งทำให้เป็นที่นิยมเพิ่มมากขึ้น ใน .NET Framework นี้ยังสามารถพัฒนาโปรแกรมให้มีความสามารถและการทำงานได้หลากหลาย และยังสามารถเขียนเพื่อใช้งานร่วมกับ Application อื่น ๆ ที่พัฒนาด้วย .NET Framework ได้เช่นเดียวกัน

2.2.1 จุดเด่นของ Visual Studio C#

เป็นรูปแบบของภาษาที่ทำงานเป็นลำดับ (Sequential) ตัวแปรและ Objects มีความสามารถเด่นชัดในเรื่องของการจัดการคุณสมบัติ (Properties) และการตั้งค่าเริ่มต้นที่ช่วยให้สามารถพัฒนาระบบได้อย่างสะดวกและรวดเร็ว



รูปที่ 2.13 แสดงตัวอย่างการเขียนคำสั่งโปรแกรม Visual Studio C#

บทที่ 3
วิธีการดำเนินงานวิจัย

โครงงานพิเศษนี้เริ่มจัดทำโดยการเขียนโปรแกรมเพื่อควบคุมการทำงานของโพลาริมิเตอร์ โดยการเขียนโปรแกรมผ่านโปรแกรม Visual Studio 2017 โดยการใช้ภาษา C# เพื่อควบคุมและสั่งการหมุนของแผ่นโพลาไรซ์ให้ได้ตามต้องการเพื่อวิเคราะห์แนวโพลาไรซ์ โดยระบบการทำงานนี้จะแบ่งเป็น 2 ส่วนหลักๆคือ Hardware System และ Software System

3.1 Hardware System

3.1.1 ขั้นตอนการสร้างระบบโพลาริมิเตอร์

ในขั้นตอนแรกเริ่มต้นจากการจัดระบบโพลาริมิเตอร์ ซึ่งจะเรียงลำดับตามรูปที่ 3.1 เริ่มต้นที่แสงเลเซอร์ HeNe เนื่องจากเป็นแสงโพลาไรซ์ (Polarizer) แสงจะวิ่งเข้าตัว Sample หรือสารละลายที่เราจะทดลอง แต่ในที่นี้เลือกใช้เป็นแผ่น Half Waveplate ในการทำการทดลอง จากนั้นแสงจะเข้า ND filter (Neutral Density Filter) เพื่อลดความเข้มแสง แล้วแสงจะเข้าแผ่นโพลาไรซ์ที่ติดตั้งกับแท่นหมุนที่ขับเคลื่อนด้วยสเต็ปปิงมอเตอร์ (Motor Driven Rotating State with Polarizer) ซึ่งทำหน้าที่เป็นตัววิเคราะห์แนวโพลาไรซ์ (Analyzer) โดยมีสเกลบอกมุมเป็นองศาและเศษขององศาในหน่วยลิปดา จากนั้นแสงจะเข้าสู่ Photodetector เพื่อวัดความเข้มแสงหลังผ่านแผ่นโพลาไรซ์

รูปที่ 3.1 การจัดวางอุปกรณ์ของ Polarimeter

หมายเลข 1 คือ แสงเลเซอร์ฮีเลียม-นีออน (HeNe Laser)

หมายเลข 2 คือ แผ่น Half Waveplate ซึ่งใช้แทนสารละลายที่จะทดลอง

หมายเลข 3 คือ แผ่น ND Filter (Neutral Density Filter)

หมายเลข 4 คือ แผ่นโพลาไรซ์ที่ติดตั้งกับแท่นหมุนที่ขับเคลื่อนด้วยสเต็ปปิงมอเตอร์ (Motor Driven Rotating State with Polarizer)

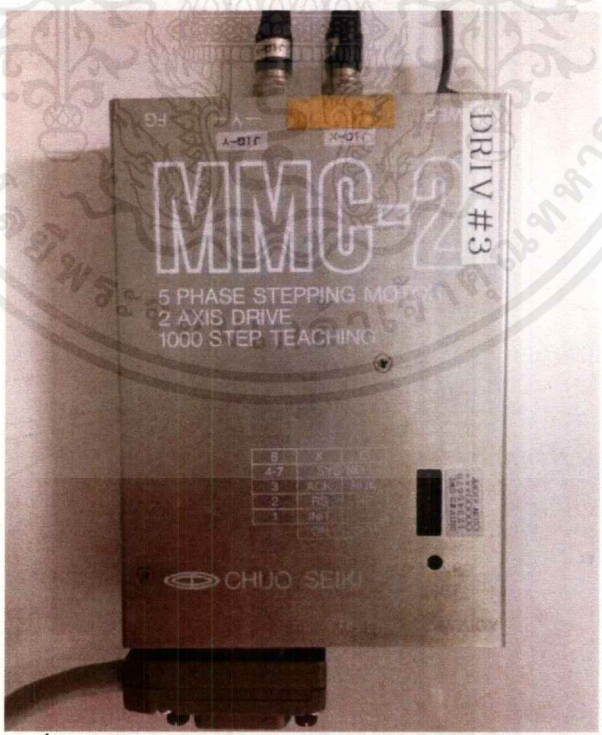
หมายเลข 5 คือ Photodetector

หมายเลข 6 คือ MMC-2 5 Phase Stepping Motor Driver

หมายเลข 7 คือ Digital Multimeter

3.1.2 ระบบขับเคลื่อน

เนื่องจากในระบบต้องการทราบแนวของโพลาไรซ์ที่บิดไปเมื่อผ่านแผ่น Half Waveplate จึงใช้การหมุนของแผ่นโพลาไรซ์เพื่อวิเคราะห์แนวของโพลาไรซ์ โดยใช้ MMC-2 5 Phase Stepping Motor Driver ดังรูปที่ 3.2 เป็นชุดขับสเต็ปปิงมอเตอร์ในการควบคุมการหมุนของสเต็ปปิงมอเตอร์ 5 เฟสดังรูปที่ 3.3 ที่ติดตั้งอยู่กับแผ่นโพลาไรซ์หมุนหาแนวโพลาไรซ์ โดยใช้คำสั่งด้วยคอมพิวเตอร์ผ่าน USB/GPIB ดังรูปที่ 3.4



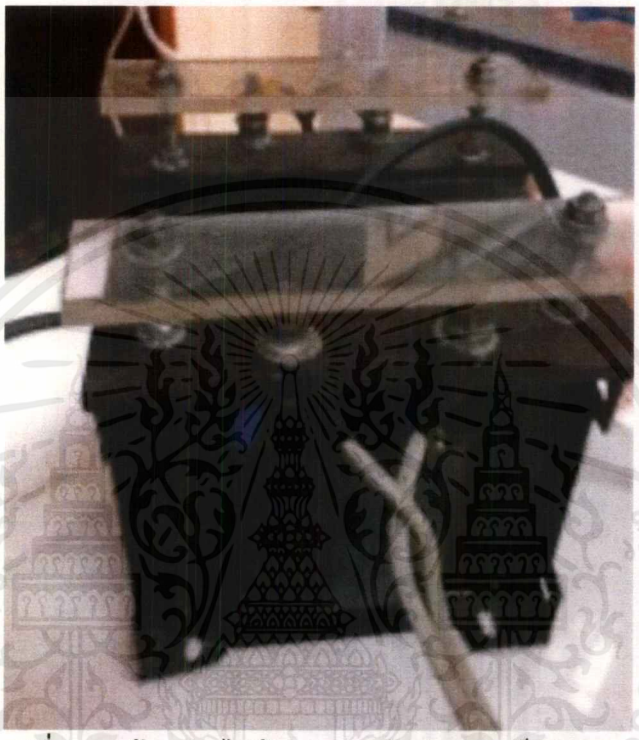
รูปที่ 3.2 MMC-2 5 Phase stepping motor driver

รูปที่ 3.3 5 Phase Stepping Motor

รูปที่ 3.4 USB/GPIB

จากรูปที่ 3.3 สเต็ปปิงมอเตอร์ 5 เฟส (5 Phase Stepping Motor) เป็นมอเตอร์ที่ขับเคลื่อนด้วยสัญญาณพัลส์ (Pulse) ลักษณะการขับเคลื่อน จะหมุนรอบแกนได้ 360 องศา เนื่องจากต้องการความแม่นยำในการวัด ความสะดวก และความรวดเร็ว จึงเลือกใช้ สเต็ปปิงมอเตอร์ 5 เฟส ที่สามารถหมุนได้ตำแหน่งที่ต้องการ และละเอียดกว่ามอเตอร์ทั่วไป โดยความละเอียดสูงสุดของ สเต็ปปิงมอเตอร์ที่ใช้คือ 0.72 องศาต่อ Step

จากรูปที่ 3.2 เนื่องจาก MMC-2 5 Phase Stepping Motor Driver จะต้องใช้ไฟกระแสสลับ 110 V ความถี่ 50 Hz จึงต้องมีหม้อแปลงเพื่อแปลงไฟกระแสสลับจาก 220 V ความถี่ 50 Hz เป็นไฟกระแสสลับ 110 V ความถี่ 50 Hz ในรูปที่ 3.5 เพื่อนำไปใช้กับ MMC-2 5 Phase Stepping Motor Driver

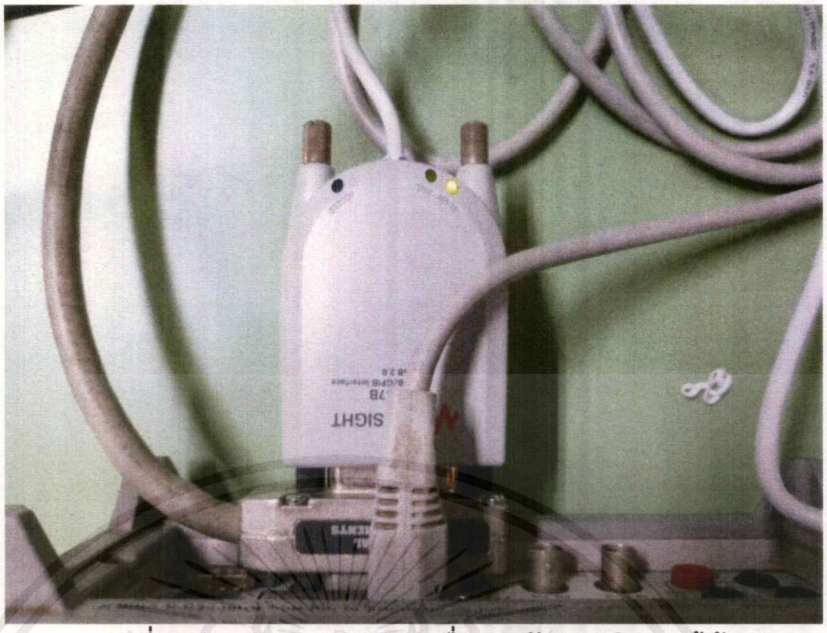


รูปที่ 3.5 หม้อแปลงไฟฟ้ากระแสสลับ 220 V เป็น 110 V

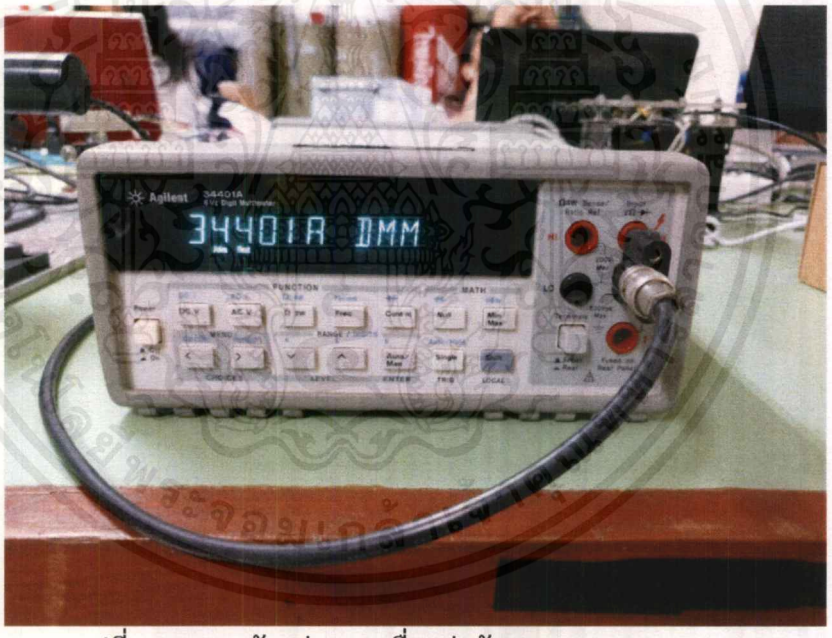
3.3 Software System

ขั้นตอนต่อไปคือการสร้างโปรแกรมควบคุมการทำงานของโพลาริมิเตอร์ในระบบปฏิบัติการ Windows 10 เริ่มต้นโดยการติดตั้งโปรแกรม Visual Studio 2017 หลังจากนั้นทำการติดตั้ง Windows Forms App ด้วย .NET Framework ใน Application บน Visual Studio C# เพื่อใช้ในการเขียนโปรแกรมหลักของ Polarimeter และใช้ Visual Studio C# ในการเขียนคำสั่งควบคุมการทำงานของ Digital Multimeter และ MMC-2 5 Phase stepping motor driver ให้ทำงานสัมพันธ์กัน

เริ่มต้นการเชื่อมต่อเครื่อง ให้ทำการติดตั้ง Keysight IO Libraries Suite และ Driver สาย GPIB ลงในเครื่องคอมพิวเตอร์ จากนั้นนำอุปกรณ์ที่ใช้มาเชื่อมต่อกับคอมพิวเตอร์ผ่านสาย GPIB เมื่อเชื่อมต่อสำเร็จจะสังเกตเห็น LED สีเขียวที่ช่อง Ready ดังรูปที่ 3.6 แสดงว่าสาย USB/GPIB สามารถเชื่อมต่อกับคอมพิวเตอร์ได้แล้วจะแสดงผลดังรูปที่ 3.7

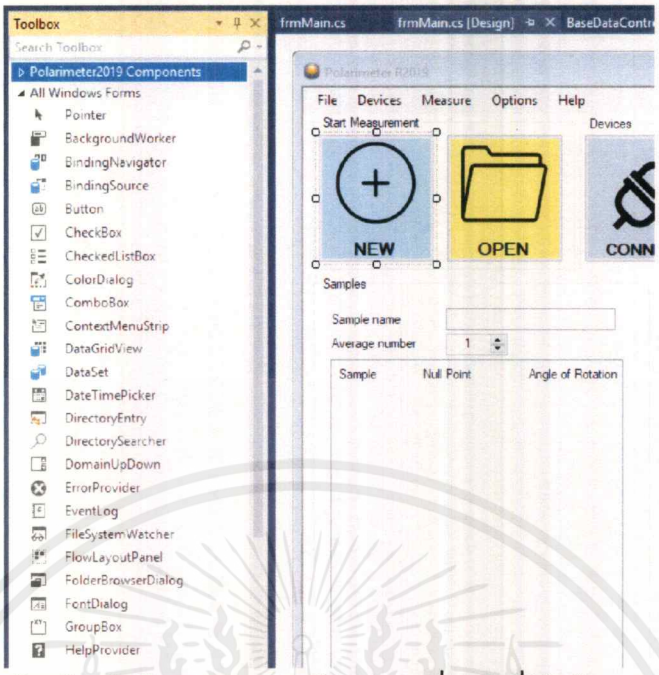


รูปที่ 3.6 USB/GPIB สามารถเชื่อมต่อกับคอมพิวเตอร์ได้



รูปที่ 3.7 แสดงตัวอย่างการเชื่อมต่อกับ Digital Multimeter

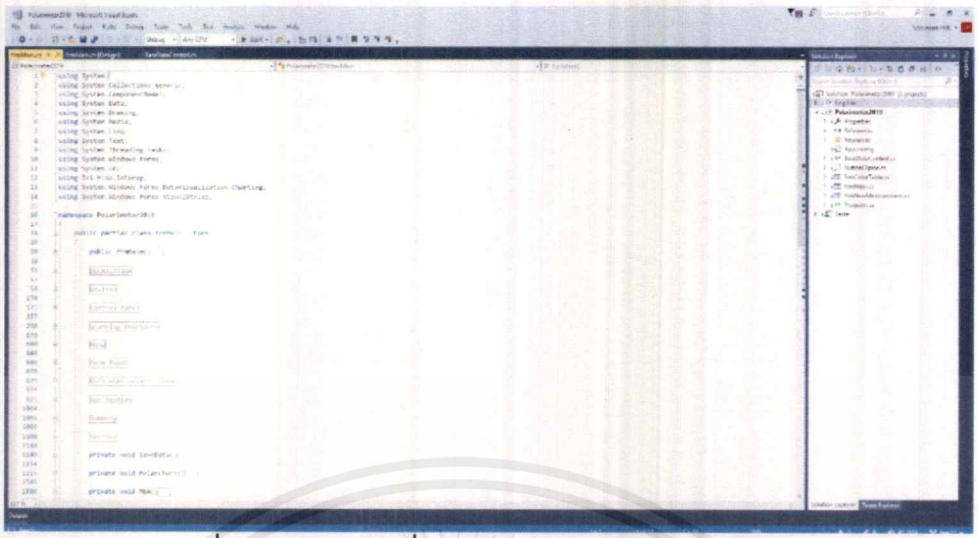
ในขั้นตอนต่อไปทำการสร้างหน้าตาของโปรแกรม (User Interface) ของโปรแกรม Polarimeter โดยใช้ Toolbox หรือกล่องเครื่องมือที่อยู่ใน Windows Forms App ในการสร้าง ซึ่งจะได้ตามรูปที่ 3.8



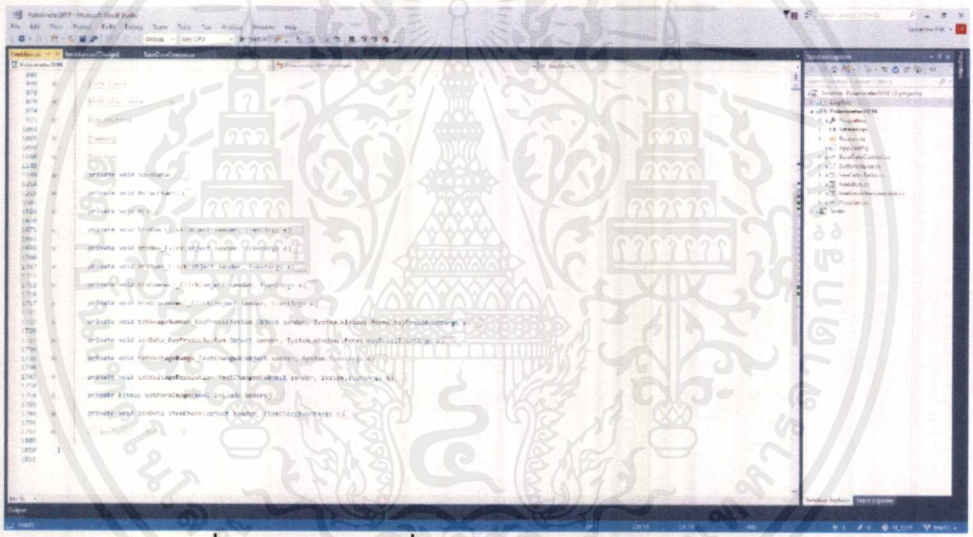
รูปที่ 3.8 ด้านซ้ายมือคือ Toolbox หรือกล่องเครื่องมือที่ใช้สร้าง User Interface(UI)

เมื่อทำการออกแบบ User Interface (UI) เสร็จ จะได้หน้าตาดังรูปที่ 3.9 และขั้นตอนต่อไปคือทำการเขียนคำสั่งเพื่อควบคุมการทำงานของ User Interface (UI) จะได้ดังรูปที่ 3.10 และรูปที่ 3.11

รูปที่ 3.9 โปรแกรม Polarimeter2019 ด้านซ้ายคือส่วนควบคุม และด้านขวาคือกราฟ

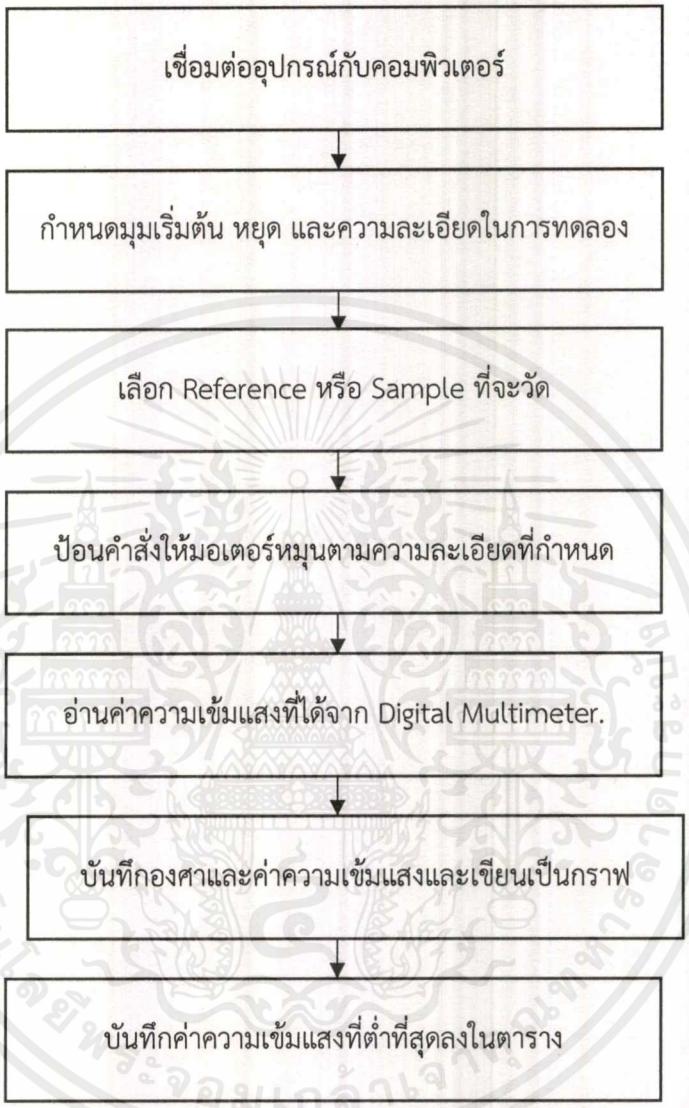


รูปที่ 3.10 Code คำสั่งของโปรแกรม Polarimeter2019



รูปที่ 3.11 Code คำสั่งของโปรแกรม Polarimeter2019

ต่อไปจะเป็นแผงผันการทำงานของโปรแกรม Polarimeter เพื่อให้เข้าใจเกี่ยวกับระบบการทำงานของโปรแกรมได้มากยิ่งขึ้น จึงแสดงลำดับการทำงานของโปรแกรม Polarimeter ดังรูปที่ 3.12

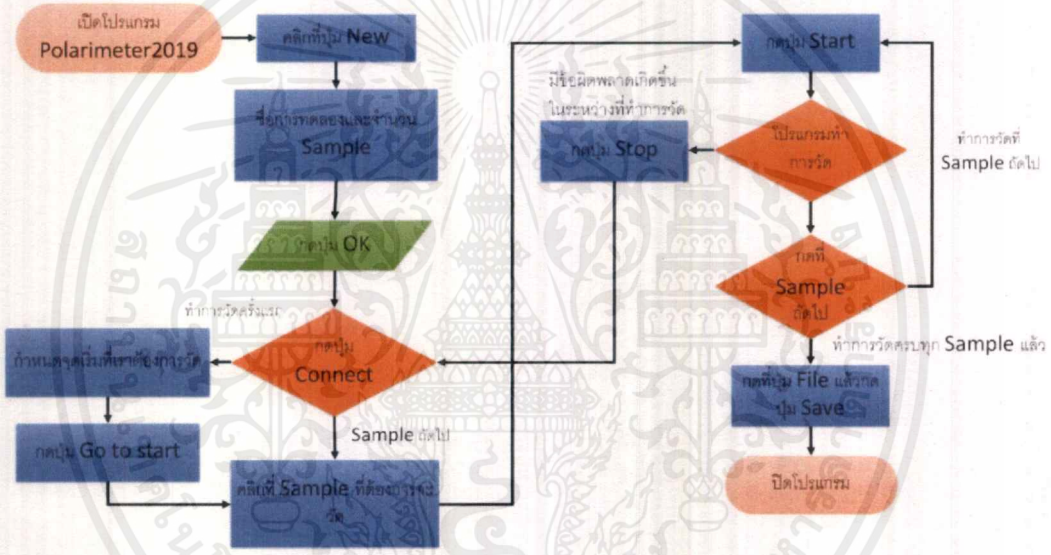


รูปที่ 3.12 แผนผังการทำงานของโปรแกรม Polarimeter

บทที่ 4
ผลการทดลองและอภิปรายผล

4.1 ผลการทำงานของโปรแกรม Polarimeter

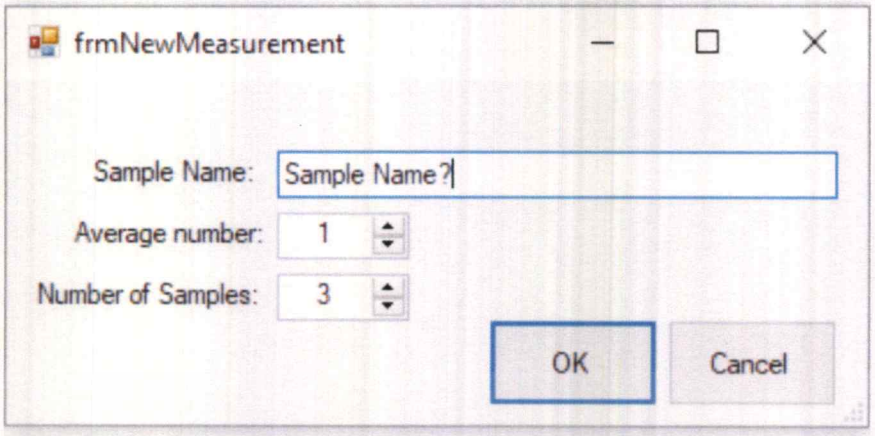
จากการสร้างโปรแกรม Polarimeter เพื่อควบคุมการทำงานของมอเตอร์ที่ติดตั้งอยู่กับแผ่นโพลาไรซ์เพื่อวิเคราะห์หาแนวโพลาไรซ์ จะได้ขั้นตอนการใช้งานของโปรแกรมดังรูปที่ 4.1 ซึ่งจากบอกวิธีการใช้งานของโปรแกรมตั้งแต่เริ่มต้นการเปิดโปรแกรม



รูปที่ 4.1 แผนผังการใช้งานของโปรแกรม Polarimeter

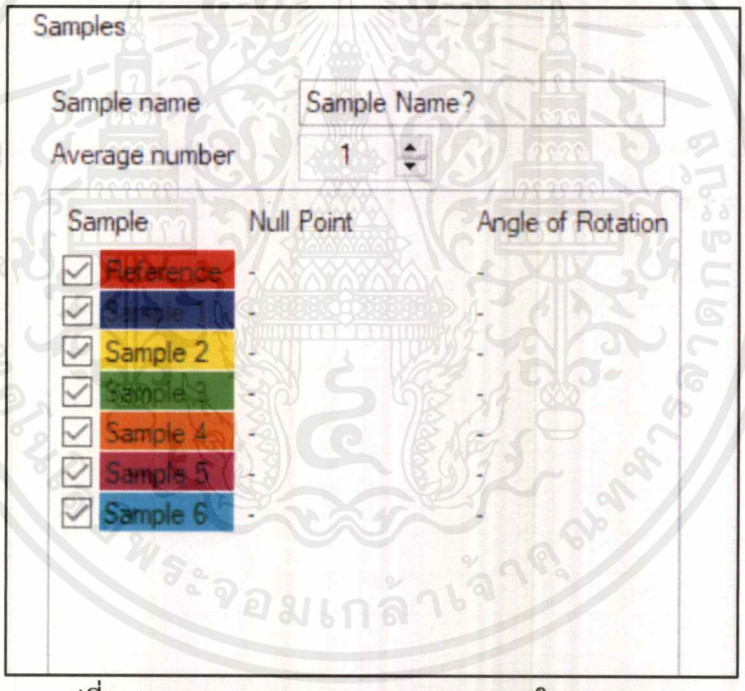
วิธีการใช้โปรแกรม

1. เปิดโปรแกรม Polarimeter
2. กด NEW เพื่อเริ่มต้นการทดลอง
3. จะขึ้นหน้าต่างการเขียนชื่อโปรแกรม ความละเอียดในการวัด และจำนวน Sample ที่จะทำการทดลองดังรูปที่ 4.2



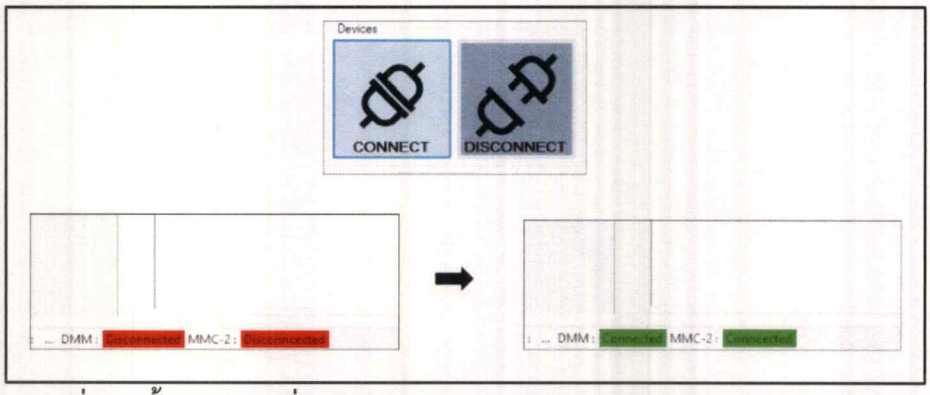
รูปที่ 4.2 ขั้นตอนการใส่ชื่อการทดลอง ความละเอียดในการวัด และจำนวน Sample

4. กด OK จะแสดง Reference และจำนวน Sample ดังรูปที่ 4.3



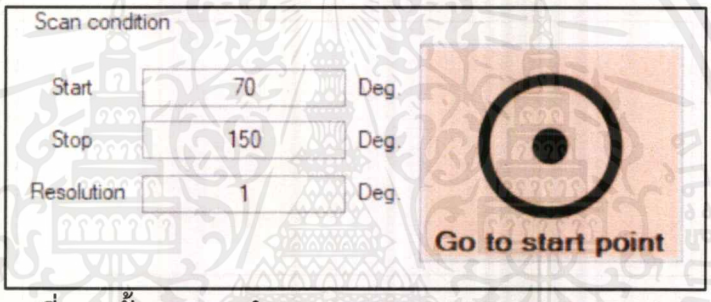
รูปที่ 4.3 แสดง Reference และ Sample ในการทดลอง

5. กดปุ่ม Connect เพื่อเชื่อมต่อกับ Motor Driven และ Digital Multimeter จากนั้นจะแสดงสถานะว่าเชื่อมต่อแล้วเป็นไฟสีเขียว ดังรูปที่ 4.4

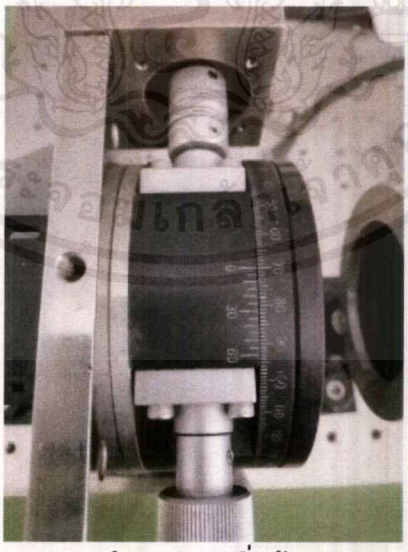


รูปที่ 4.4 ขั้นตอนการเชื่อมต่อของ Motor Driven และ Digital Multimeter

6. เลือกมุม Start Stop และ Resolution ของการทดลองดังรูปที่ 4.5 มอเตอร์จะหมุนไปยังตำแหน่งที่เริ่มต้นการทดลองดังรูปที่ 4.6

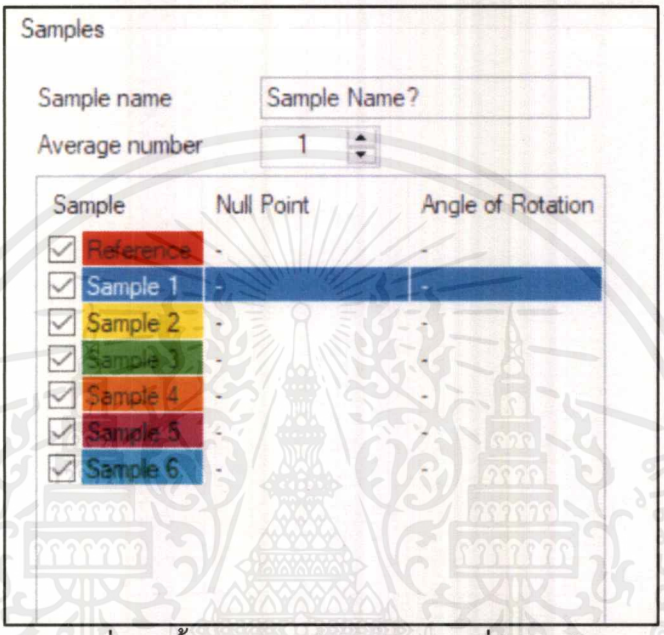


รูปที่ 4.5 ขั้นตอนการกำหนดจุด Start Stop และ Resolution



รูปที่ 4.6 แสดงตำแหน่งจุดเริ่มต้นของการทดลอง

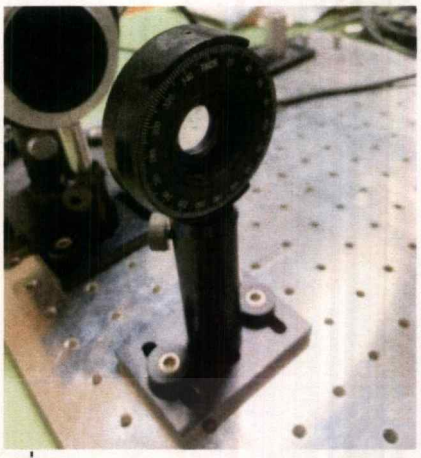
7. กดที่ Reference เพื่อว่าเดิมที่แนวโพลาไรซ์อยู่ที่ตำแหน่งใด
8. กดปุ่ม Start เพื่อทำการหมุนมอเตอร์หาค่ามุมที่ให้ความเข้มแสงที่ต่ำที่สุด
9. กด File แล้วกด Save เพื่อบันทึกค่าความเข้มแสงและมุมต่างๆ
10. ทำการเปลี่ยน Sample ที่จะทำการทดลอง จากนั้นกดที่ช่อง Sample ดังรูปที่ 4.7 แล้วทำการทดลองเหมือนข้อ 8 กับข้อ 9



รูปที่ 4.7 ขั้นตอนการเลือก Sample ที่ทำการวัด

4.2 ตัวอย่างผลการทดลอง

สำหรับการวัดความเข้มข้นของสารละลายชนิดต่าง ๆ เพื่อต้องการที่จะหาว่าสารละลายชนิดนั้นมีความเข้มข้นมากน้อยเพียงใด โดยการหมุนแผ่นโพลาไรซ์ที่ทำหน้าที่เป็นตัววิเคราะห์มุมที่บิดไปของแนวโพลาไรซ์เมื่อผ่านสารละลาย ในการทดลองนี้เลือกใช้เป็นแผ่น Half Waveplate แทนการใช้สารละลายจริงๆ เพราะไม่ทราบว่าแกนแสงของสารละลาย (Optical axis) มีมุมเท่าไหร่ จึงใช้แผ่น Half Waveplateเป็นตัวสังเกตแทนสารละลายในการกำหนดมุมดังรูปที่ 4.8



รูปที่ 4.8 มุมของแผ่น Half Waveplate

จากรูปที่ 4.8 ทำการปรับมุมของแผ่น Half Waveplate ไปที่มุม 10°, 20°, 30°, -10°, -20° และ -30° ซึ่งผลการทดลองในการวัดค่าความเข้มแสงเมื่อปรับมุมของแผ่น Half Waveplate ไปตามมุมที่ต้องการวัดจะได้ผลการทดลองดังตารางที่ 4.1

ตารางที่ 4.1 บันทึกค่ามุมที่แสงมีความเข้มต่ำสุดตามมุมต่าง ๆ ของ Half Waveplate

มุมของแผ่นโพลาไรซ์ θ_0 = 89.75°

$\theta_{\lambda/2}$	θ_{Pol}	$\theta_0 - \theta_{Pol}$	ทิศทางการหมุน
10	70.00	19.75	ตามเข็ม
20	50.16	39.59	ตามเข็ม
30	30.00	59.75	ตามเข็ม
-10 (350)	110.00	-20.25	ทวนเข็ม
-20 (340)	130.00	-40.25	ทวนเข็ม
-30 (330)	150.00	-60.25	ทวนเข็ม

เมื่อทำการทดลองในตอนแรกเสร็จแล้วในขั้นตอนต่อไปจะทำการทดลองเช่นเดิมแต่จะใช้โปรแกรม Polarimeter ในควบคุมการหมุนของแผ่นโพลาไรซ์โดยตั้งค่ามุมเริ่มต้นที่ 0° และหยุดที่ 180° ด้วยความละเอียด 2.0° จากนั้นบันทึกมุมที่แสงมีความเข้มต่ำสุดลงในตารางซึ่งได้ผลการทดลองออกมาดังตารางที่ 4.2

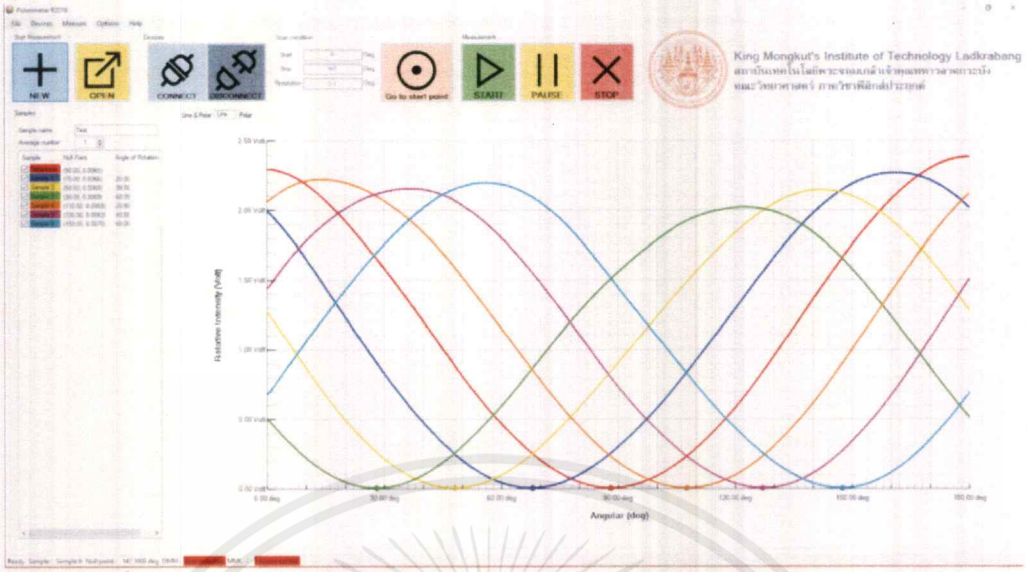
ตารางที่ 4.2 บันทึกค่ามุมที่แสงมีความเข้มต่ำสุดตามมุมต่าง ๆ ของ Half Waveplate (โดยใช้เครื่องคอมพิวเตอร์)

มุมของแผ่นโพลาไรซ์ θ_0 = 89.50°

$\theta_{\lambda/2}$	θ_{Pol}	$\theta_0 - \theta_{Pol}$	ทิศทางการหมุน
10	70.00	19.50	ตามเข็ม
20	50.00	39.50	ตามเข็ม
30	30.50	59.00	ตามเข็ม
-10 (350)	110.50	-21.00	ทวนเข็ม
-20 (340)	130.00	-40.50	ทวนเข็ม
-30 (330)	149.50	-60.00	ทวนเข็ม

ซึ่งจากการทดลองดังกล่าวจะพบว่ามุมของแผ่นโพลาไรซ์ของการปรับด้วยตนเองกับการใช้คอมพิวเตอร์ควบคุมการหมุนของแผ่นโพลาไรซ์มีความแตกต่างกันอยู่ที่ 0.27% และจากการทดลองโดยใช้คอมพิวเตอร์โปรแกรมก็สามารถสร้างกราฟจากการนำค่าความเข้มแสงมาเทียบกับมุมที่หมุนไปของแผ่นโพลาไรซ์ซึ่งจากรูปที่ 4.9 จะสามารถทำให้เราเห็นได้ชัดเจนว่าที่มุม 90° จะเป็นมุมที่มีความเข้มแสงต่ำมากหรือหมายถึงแสงสามารถผ่านออกไปได้น้อยมาก ๆ

รูปที่ 4.9 การทดลองโดยใช้โปรแกรม Polarimeter2019 (แบบ Polar chart)



รูปที่ 4.10 การทดลองโดยใช้โปรแกรม Polarimeter2019 (แบบ Sine wave)

บทที่ 5
สรุปผลและข้อเสนอแนะ

สรุปผลการพัฒนาโปรแกรมควบคุมโพลาริมิเตอร์ด้วยภาษา C# โดยได้ทำการศึกษาการเขียนโปรแกรมเบื้องต้นและศึกษาการทำงานของโปรแกรมโพลาริมิเตอร์ในรุ่นก่อนหน้านี้ เพื่อนำมาพัฒนาและปรับปรุงโปรแกรมให้สามารถทำงานกับโพลาริมิเตอร์ได้อย่างสมบูรณ์มากยิ่งขึ้น ซึ่งจากการทดลองเราจะพบว่าในโปรแกรมเก่านั้นจะถูกเขียนขึ้นด้วยภาษา VB ซึ่งเป็นคนละภาษากันกับที่ผู้พัฒนาได้เลือกใช้และเนื่องจากการเขียนโปรแกรมผ่านภาษา VB นั้นมีความละเอียดไม่มากเท่ากับภาษา C# ผู้ทำการทดลองจึงเลือกใช้ภาษา C# ในการสร้างและพัฒนาโปรแกรมขึ้นมา ซึ่งเรารู้อยู่แล้วว่าโปรแกรมเก่าที่ถูกเขียนขึ้นด้วยภาษา VB นั้นจะไม่สามารถทำงานบนระบบปฏิบัติการณ์ Windows 10 รุ่นล่าสุดได้ผู้ทำการทดลองจึงได้ทำการอัพเดตโค้ดเดิมโดยการแปลงโค้ดจากภาษา VB ให้เป็นภาษา C# ซึ่งจากการแปลงโค้ดมาผู้ทำการทดลองก็พบว่าโค้ดที่แปลงมาจากภาษา VB นั้นมีข้อผิดพลาดค่อนข้างมากจึงได้ทำการแก้ไขไปในทีละจุด

จากการทดลองสร้างกราฟในโปรแกรม Polarimeter2019 ด้วยการปรับแผ่น Half Waveplate ไปที่มุม 0° เราจะพบว่าที่มุม 90° และมุม 270° นั้นจะเป็นจุดที่มีความเข้มแสงน้อยมาก ๆ ซึ่งมุม 90° นั้นจะเป็นมุมที่มีค่าความเข้มแสงน้อยที่สุดและถ้าสังเกตจากกราฟแบบ Polar chart จะพบว่ากราฟมีลักษณะคล้ายรูปเลขแปดซึ่งหมายถึงความเข้มแสงในจุดแรกที่มุม 0° นั้นจะมีค่าสูงที่สุดและเมื่อเราหมุนแผ่น Analyzer ไปเรื่อย ๆ แสงก็จะมีความเข้มลดลงจนกระทั่งแสงไม่สามารถผ่านไปได้ในมุม 90° ซึ่งจากลักษณะกราฟแบบนี้ทำให้เราทราบว่าผลการทดลองที่ได้นั้นเป็นไปตามทฤษฎี

5.1 สรุปผลการทดลอง

5.1.1 ผลของขั้นตอนการเชื่อมต่อกับ Digital Multimeter

ในการทดลองเครื่องมือโพลาริมิเตอร์นั้นจำเป็นต้องใช้ Digital Multimeter ในการแสดงค่าความเข้มแสงออกมาในรูปของความต่างศักย์ไฟฟ้า (Volt) เพื่อแสดงความเข้มแสงในขณะที่มุมต่าง ๆ ของแผ่นโพลาไรซ์ซึ่งผลการทดลองจะพบว่าผู้ทดลองสามารถทำการเชื่อมต่อโปรแกรม Visual studio 2017 ภาษา C# กับ Digital multimeter ได้ซึ่งที่เครื่อง Digital multimeter สามารถแสดงข้อความที่ผู้ทดลองทำการสั่งงานลงไปได้อย่างสมบูรณ์ถูกต้อง

5.1.2 ผลของขั้นตอนการเก็บค่าการทดลองเพื่อใช้ในการสร้างกราฟ

ในขั้นตอนนี้ผู้ทดลองได้ทำการสร้างกราฟขึ้นมาทั้งหมด 2 แบบคือแบบ Polar chart และแบบ Sine wave ซึ่งในขั้นตอนนี้จะเป็นการนำค่าจากมัลติมิเตอร์มาเก็บไว้ใน Array เพื่อดึงไปใช้ในการสร้างกราฟขึ้นมาแบบเรียลทามส์ ซึ่งจากการทดลองก็จะพบว่าผู้ทำการทดลองสามารถสร้างกราฟขึ้นมาเพื่อแสดงผลการทดลองได้อย่างสมบูรณ์

5.1.3 ผลการทดลองการเมื่อปรับแผ่น Analyzer โดยไม่ใช้โปรแกรม

ในการทดลองนี้ผู้ทำการทดลองได้ทำการหมุนแผ่น Analyzer เพื่อหาค่าความเข้มแสงที่น้อยที่สุดและจากการทดลองจะพบว่าเราสามารถหาค่าความเข้มแสงที่น้อยที่สุดได้ที่มุม 89.75° และจะพบว่าที่ค่า $\theta_0 - \theta_{Pol}$ จะมีความคลาดเคลื่อนจากทฤษฎีเล็กน้อย

5.1.4 ผลการทดลองการเมื่อปรับแผ่น Analyzer โดยใช้โปรแกรม

ในการทดลองนี้เราได้ทำการควบคุมการปรับแผ่น Analyzer ด้วยโปรแกรม Polarimeter2019 เพื่อหาความเข้มแสงที่มีค่าน้อยที่สุดซึ่งเราก็จะพบว่าค่าความเข้มแสงที่น้อยที่สุดนั้นจะอยู่ที่มุม 88.50° และจะพบว่าค่า $\theta_0 - \theta_{Pol}$ จะมีความคลาดเคลื่อนจากค่าทฤษฎีที่ควรจะได้เล็กน้อย

5.2 ข้อเสนอแนะ

เนื่องจากการทดลองโพลาริมิเตอร์จะต้องใช้แสงที่เป็นโพลาไรซ์ ผู้ทำการทดลองจะต้องจัดแสงเลเซอร์ให้อยู่ในแนวโพลาไรซ์และตั้งสเกลของแท่นหมุนแผ่นโพลาไรซ์ที่ติดอยู่กับมอเตอร์ให้ตรงกับ 0°

ในการทดลองในการทำโครงงานพิเศษนี้เลือกใช้เป็นแผ่น Half Waveplate แทนสารละลายในการทดลอง ดังนั้นผู้ทำการทดลองควรใช้สารละลายแทนเพื่อให้สังเกตแนวโพลาไรซ์ที่เปลี่ยนไปได้ชัดเจน

เอกสารอ้างอิง

พงศธร สุทธิ และสราวุธ ดอนคำชัยมงคล. 2555. **ภาษา C#**. [Online]. Available: http://tonkung.ueuo.com/index.html. เข้าถึงเมื่อวันที่ 16 มิ.ย. 62.

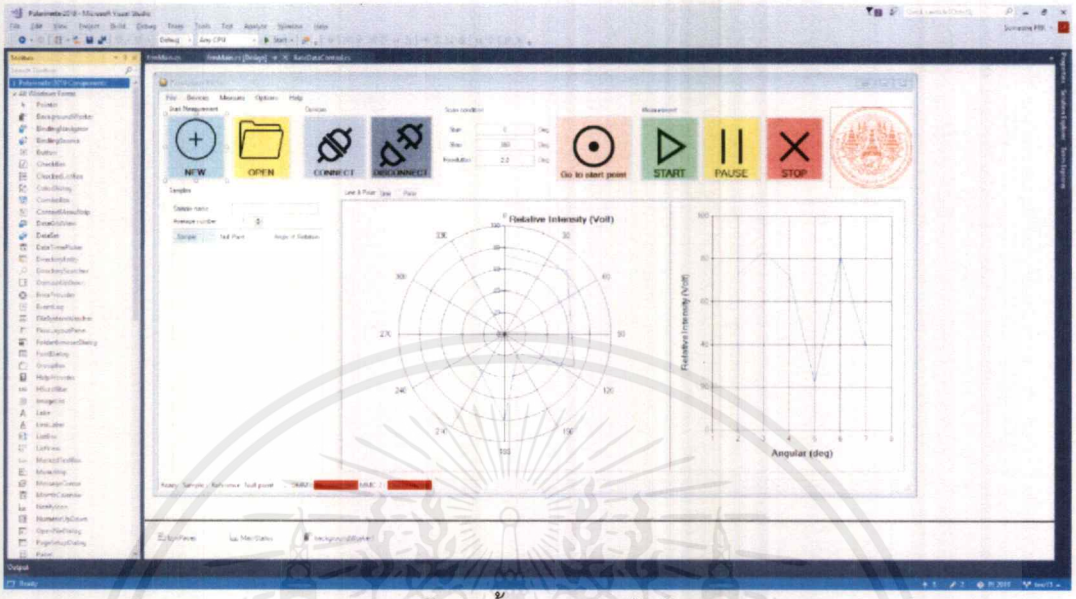
รศ.ดร. วราวุฒิ เถาลัดดา. 2560. **Measurement and Instrumentation Laboratory I**. ฉบับปีการศึกษา 2560. ภาควิชาฟิสิกส์ คณะวิทยาศาสตร์, สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง.

http://marcuscode.com/lang/csharp. เข้าถึงเมื่อวันที่ 16 มิ.ย. 62.

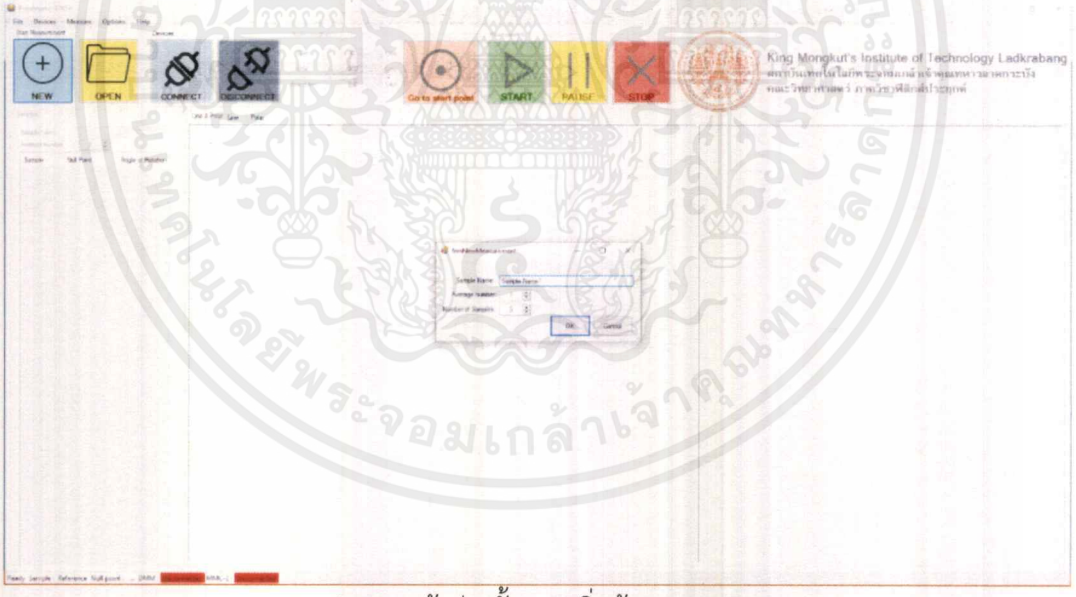
ภาคผนวก

ภาคผนวก ก

หน้าต่างการใช้งานโปรแกรม Polarimeter



รูปแสดงหน้าต่างขั้นตอนการสร้างโปรแกรม



รูปแสดงหน้าต่างขั้นตอนเริ่มต้นการทดลอง

ภาคผนวก ข

ชุดคำสั่งของโปรแกรม Polarimeter

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.IO;
using Ivi.Visa.Interop;
using System.Windows.Forms.DataVisualization.Charting;

namespace Polarimeter2019
{
    public partial class frmMain : Form
    {
        public frmMain()
        {
            InitializeComponent();
            BDC = new BaseDataControl();
            DMM = new Ivi.Visa.Interop.FormattedIO488();
            MMC = new Ivi.Visa.Interop.FormattedIO488();
        }
        #region DECRALATION
        private Ivi.Visa.Interop.FormattedIO488 ioDmm;
        public BaseDataControl BDC;
        Random rand = new Random();
        const double StepFactor = 0.013333;
        bool IsScanning = false;
```

ภาคผนวก ข (ต่อ)

```
bool IsContinuing = false;
int CurrentPointIndex = 0;
double SpecificRotation;
int NumberOfRepeatation;
int SelectedIndex;
double CurrentLightIntensity =0;
double CurrentLightIntensity2 = 0;
double CurrentTheta = 0;
double CurrentTheta2 = 0;
int StepNumber;
string MSG;
public Color ReferenceColor = Color.Red;
public Color[] ColorTable = new Color[20];
#endregion
#region Devices
private Ivi.Visa.Interop.IFormattedIO488 DMM;
private Ivi.Visa.Interop.IFormattedIO488 MMC;
void MMCDisconnectDevices()
{
    try
    {
        MMC.IO.Close();
        MMC.IO = null;
        lblMMC.Text = "Disconncected";
        lblMMC.BackColor = Color.Red;
    }
    catch (Exception ex)
    {
```

ภาคผนวก ข (ต่อ)

```
            MessageBox.Show("IO Error" + ex.Message);
        }
    }
    void DMMDisconnectDevices()
    {
        try
        {
            DMM.IO.Close();
            DMM.IO = null;
            lblDMM.Text = "Disconnected";
            lblDMM.BackColor = Color.Red;
        }
        catch (Exception ex)
        {
            MessageBox.Show("IO Error" + ex.Message);
        }
    }
    void DisconnectDevices()
    {
        try
        {
            DMMDisconnectDevices();
            MMCDisconnectDevices();
        }
        catch (Exception ex)
        {
            MessageBox.Show("IO Error" + ex.Message);
```

ภาคผนวก ข (ต่อ)

```
        }
        void DMMConnection()
        {
            try
            {
                Ivi.Visa.Interop.ResourceManager mgr1;
                string DMMAddress;
                DMMAddress = txtDMMAddress.Text;
                mgr1 = new Ivi.Visa.Interop.ResourceManager();
                DMM.IO = (IMessage)mgr1.Open(DMMAddress);
                DMM.IO.Timeout = 7000;
                DMM.WriteString("*CLS");
                System.Threading.Thread.Sleep(100);
                DMM.WriteString("CONF:VOLT:DC " + txtVoltageRange.Text + ", " +
txtVoltageResolution.Text);
                DMM.WriteString("TRIG:SOUR IMM");
                DMM.WriteString("TRIG:DEL 1.5E-3");
                DMM.WriteString("TRIG:COUNT 1");
                DMM.WriteString("SAMP:COUNT 1");
                lblDMM.Text = "Connected";
                lblDMM.BackColor = Color.Lime;
            }
            catch (Exception ex)
            {
                MessageBox.Show("InitIO Error:" + MessageBoxButtons.RetryCancel +
ex.Message);
                lblDMM.Text = "Disconnected";
                lblDMM.BackColor = Color.Red;
```

ภาคผนวก ข (ต่อ)

```
    }

}
void MMCConnection()
{
    try
    { //CONNECT MMC
        Ivi.Visa.Interop.ResourceManager mgr2;
        string MMCAddress;
        MMCAddress = txtMMCAddress.Text;
        mgr2 = new Ivi.Visa.Interop.ResourceManager();
        MMC.IO = (IMessage)mgr2.Open(MMCAddress);
        MMC.IO.Timeout = 7000;
        lblMMC.Text = "Conncected";
        lblMMC.BackColor = Color.Lime;
    }
    catch (Exception ex)
    {
        lblMMC.Text = "Disconncected";
        lblMMC.BackColor = Color.Red;
    }

}
void ConnectedDevices()
{
    try
    {
        DMMConnection();
```

ภาคผนวก ข (ต่อ)

```
            MMCConnection();
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }
    #endregion
    #region Control Panel
    private void DoStart()
    {
        PlotReferenceCurve();
        btnStart.Enabled = false;
        btnPause.Enabled = true;
        btnStop.Enabled = true;
        gbStartMea.Enabled = false;
        gbSample.Enabled = false;
        gbScanCondition.Enabled = false;
        gbDevices.Enabled = false;
        gbMeasurement.Enabled = true;
        CurrentPointIndex = 0;
        IsScanning = true;
        lblMainStatus.Text = "Measuring...";
        DoScanLightIntensity();
        lblMainStatus.Text = "Ready";
    }
    private void DoStop()
```

ภาคผนวก ข (ต่อ)

```
        StopScanning();
        btnStart.Enabled = true;
        btnPause.Enabled = false;
        gbSample.Enabled = true;
        gbDevices.Enabled = true;
        gbMeasurement.Enabled = true;
    }
    private void DoPause()
    {
        if (btnPause.Text == "PAUSE")
        {
            DoPasuseScanning();
        }
        else
        {
            DoContinueScanning();
        }
        btnStart.Enabled = false;
        btnPause.Enabled = true;
        btnStop.Enabled = true;
        if (btnPause.Text == "PAUSE")
        {
            btnPause.Text = "CONTINUE";
        }
        else
        {
            btnPause.Text = "PAUSE";
            DoScanLightIntensity();
```

ภาคผนวก ข (ต่อ)

```
        }
    }
    private void btnStart_Click(System.Object sender, System.EventArgs e)
    {
        string trt;
        if (SelectedIndex == 0)
        {
            trt = "Reference data";
            DialogResult result = MessageBox.Show("Are you sure to measure " + trt,
"Measure", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
            if (result == DialogResult.Yes)
            {
                ListViewItem lvi;
                lvi = lsvData.Items[SelectedIndex];
                if (lvi.SubItems[1].Text == "-")
                {
                    DoStart();
                }
                else
                {
                    MessageBox.Show("Again!!!", "Delete!?", MessageBoxButtons.YesNo,
MessageBoxIcon.Question);
                    if(result == DialogResult.Yes)
                    {
                        if (SelectedIndex==0)
                        {
                            Series ser1 = chart1.Series[SelectedIndex];
                            Series ser2 = chart2.Series[SelectedIndex];
```

ภาคผนวก ข (ต่อ)

```
                    Series ser3 = chart3.Series[SelectedIndex];
                    Series ser4 = chart4.Series[SelectedIndex];
                    chart1.Series.Remove(ser1);
                    chart2.Series.Remove(ser2);
                    chart3.Series.Remove(ser3);
                    chart4.Series.Remove(ser4);
                }
                PolarChart();
                DoStart();
            }
            else
            {

            }
        }
    }
    else
    {

    }
}
else
{
    trt = "Sample" + SelectedIndex;
    DialogResult result = MessageBox.Show("Are you sure to measure " + trt,
"Measure", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
    if (result == DialogResult.Yes)
    {
```

ภาคผนวก ข (ต่อ)

```
                ListViewItem lvi;
                lvi = lsvData.Items[SelectedIndex];
                if (lvi.SubItems[1].Text == "-")
                {
                    DoStart();
                }
                else
                {
                    MessageBox.Show("Again!!!", "Delete!?", MessageBoxButtons.YesNo,
MessageBoxIcon.Question);
                    if (result == DialogResult.Yes)
                    {
                        if (SelectedIndex == 0)
                        {
                            Series ser1 = chart1.Series[SelectedIndex];
                            Series ser2 = chart2.Series[SelectedIndex];
                            Series ser3 = chart3.Series[SelectedIndex];
                            Series ser4 = chart4.Series[SelectedIndex];
                            chart1.Series.Remove(ser1);
                            chart2.Series.Remove(ser2);
                            chart3.Series.Remove(ser3);
                            chart4.Series.Remove(ser4);
                        }
                        else
                        {
                            if (SelectedIndex < BDC.Data.Length)
                            {
                                Series ser1 = chart1.Series[SelectedIndex];
```

ภาคผนวก ข (ต่อ)

```
                    Series ser2 = chart2.Series[SelectedIndex];
                    Series ser3 = chart3.Series[SelectedIndex];
                    Series ser4 = chart4.Series[SelectedIndex];
                    chart1.Series.Remove(ser1);
                    chart2.Series.Remove(ser2);
                    chart3.Series.Remove(ser3);
                    chart4.Series.Remove(ser4);
                }
            }
            PolarChart();
            DoStart();
        }
        else
        {
        }
    }
    else
    {
    }
  }
}
private void btnPause_Click(System.Object sender, System.EventArgs e)
{
  DoPause();
}
```

ภาคผนวก ข (ต่อ)

```
        private void btnStop_Click(System.Object sender, System.EventArgs e)
        {
            DoStop();
        }
        #endregion
        #region Scanning Procedure
        Random Rnd = new Random();
        public void DoScanLightIntensity()
        {
            if (lsvData.SelectedItems.Count <= 0)
            {
                MessageBox.Show("Please select item in samples list view that you want
to measure!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
                btnStart.Enabled = true;
                btnStop.Enabled = false;
                btnPause.Enabled = false;
                btnPause.Text = "PAUSE";
                btnNew.Enabled = true;
                btnOpen.Enabled = true;
                gbStartMea.Enabled = true;
                gbSample.Enabled = true;
                gbScanCondition.Enabled = true;
                return;
            }
            if (!mnuOptionsDemomode.Checked)
            {
                ConnectedDevices();
            }
```

ภาคผนวก ข (ต่อ)

```
label10.Show();
label11.Show();
label12.Show();
label13.Show();
double ThetaA = System.Convert.ToDouble(txtStart.Text);
double ThetaB = System.Convert.ToDouble(txtStop.Text);
double Delta = System.Convert.ToDouble(txtResolution.Text);
try
{
    double dNumberOfPoint = Math.Abs((ThetaA - ThetaB) / Delta);
    int NumberOfPoint = (int)Math.Floor(dNumberOfPoint);
    if (0 < (dNumberOfPoint % Math.Floor(dNumberOfPoint)))
    {
        NumberOfPoint++;
    }
    NumberOfPoint++;
    if (SelectedIndex == 0)
    {
        BDC.Reference.X = new double[1 + NumberOfPoint];
        BDC.Reference.Y = new double[1 + NumberOfPoint];
    }
    else
    {
        if (SelectedIndex < BDC.Data.Length)
        {
            BDC.Data[SelectedIndex].X = new double[1 + NumberOfPoint];
            BDC.Data[SelectedIndex].Y = new double[1 + NumberOfPoint];
        }
```

ภาคผนวก ข (ต่อ)

```
}
if (!IsContinuing)
{
    if (SelectedIndex == 0)
    {
        BDC.Reference.Ym = 99999999;
    }
    else if (BDC.Data != null)
    {
        if (SelectedIndex < BDC.Data.Length)
        {
            BDC.Data[SelectedIndex].Ym = 99999999;
        }
    }
}
while (IsScanning)
{
    Application.DoEvents();
    if (ThetaA < ThetaB)
    {
        CurrentTheta = ThetaA + CurrentPointIndex * Delta;
    }
    else if (ThetaA > ThetaB)
    {
        CurrentTheta = ThetaA - CurrentPointIndex * Delta;
    }
    if (ThetaA < ThetaB)
    {
```

ภาคผนวก ข (ต่อ)

```
                    if (ThetaB <= CurrentTheta)
                    {
                        IsScanning = false;
                    }
                }
                else if (ThetaA > ThetaB)
                {
                    if (CurrentTheta <= ThetaB)
                    {
                        IsScanning = false;
                    }
                }
                if (mnuOptionsDemomode.Checked == false)
                {
                    StepNumber = -1 * System.Convert.ToInt32(CurrentTheta /
StepFactor); // step
                    MSG = "A:WP" + StepNumber.ToString() + "P" +
StepNumber.ToString();
                    MMC.WriteString(MSG);
                    int nAvg = (int)numRepeatNumber.Value;
                    CurrentLightIntensity = 0;
                    for (int tt = 0; tt <= nAvg - 1; tt++)
                    {
                        DMM.WriteString("READ?");
                        CurrentLightIntensity = CurrentLightIntensity +
DMM.ReadNumber();
                    }
                    CurrentLightIntensity = CurrentLightIntensity / nAvg;
                }
                Else
```

ภาคผนวก ข (ต่อ)

```
                {
                    CurrentLightIntensity = Rnd.NextDouble() * 0.1 +
Math.Cos((CurrentTheta - Rnd.NextDouble() * 50) * Math.PI / 180) + 2;
                }
                if (SelectedIndex == 0)
                {
                    BDC.PatchReference(CurrentPointIndex, CurrentTheta,
CurrentTheta2, CurrentLightIntensity,CurrentLightIntensity2);
                }
                else
                {
                    BDC.PatchData(SelectedIndex, CurrentPointIndex, CurrentTheta,
CurrentTheta2, CurrentLightIntensity,CurrentLightIntensity2);
                }
                #region Add Chart4
                chart4.Series[SelectedIndex].Points.AddXY(CurrentTheta,
CurrentLightIntensity);
                if (chart4.Series[SelectedIndex + lsvData.Items.Count].Points.Count <=
0)
                {
                    if (SelectedIndex == 0)
                        chart4.Series[SelectedIndex +
lsvData.Items.Count].Points.AddXY(BDC.Reference.Xmax, BDC.Reference.Ymax);
                    else
                        chart4.Series[SelectedIndex +
lsvData.Items.Count].Points.AddXY(BDC.Data[CurrentPointIndex].Xmax,
BDC.Data[CurrentPointIndex].Ymax);
                }
                Else
```

ภาคผนวก ข (ต่อ)

```
                {
                    if (SelectedIndex == 0)
                    {
                        chart4.Series[SelectedIndex +
lsvData.Items.Count].Points[0].XValue = BDC.Reference.Xmax;
                        chart4.Series[SelectedIndex +
lsvData.Items.Count].Points[0].YValues[0] = BDC.Reference.Ymax;
                    }
                    else
                    {
                        chart4.Series[SelectedIndex +
lsvData.Items.Count].Points[0].XValue = BDC.Data[SelectedIndex].Xmax;
                        chart4.Series[SelectedIndex +
lsvData.Items.Count].Points[0].YValues[0] = BDC.Data[SelectedIndex].Ymax;
                    }
                }
                chart4.Series[SelectedIndex + lsvData.Items.Count].MarkerStyle =
MarkerStyle.Circle;
                chart4.Series[SelectedIndex + lsvData.Items.Count].MarkerSize = 10;
                chart4.Series[SelectedIndex + lsvData.Items.Count].MarkerBorderWidth
= 2;
                chart4.Invalidate();
                #endregion
                #region Add Chart3
                chart3.Series[SelectedIndex].Points.AddXY(CurrentTheta,
CurrentLightIntensity);
                if (chart3.Series[SelectedIndex + lsvData.Items.Count].Points.Count <=
0)
                {
                    if (SelectedIndex == 0)
```

ภาคผนวก ข (ต่อ)

```
                    chart3.Series[SelectedIndex +
lsvData.Items.Count].Points.AddXY(BDC.Reference.Xm, BDC.Reference.Ym);
                else
                    chart3.Series[SelectedIndex +
lsvData.Items.Count].Points.AddXY(BDC.Data[CurrentPointIndex].Xm,
BDC.Data[CurrentPointIndex].Ym);
            }
            else
            {
                if (SelectedIndex == 0)
                {
                    chart3.Series[SelectedIndex +
lsvData.Items.Count].Points[0].XValue = BDC.Reference.Xm;
                    chart3.Series[SelectedIndex +
lsvData.Items.Count].Points[0].YValues[0] = BDC.Reference.Ym;
                }
                else
                {
                    chart3.Series[SelectedIndex +
lsvData.Items.Count].Points[0].XValue = BDC.Data[SelectedIndex].Xm;
                    chart3.Series[SelectedIndex +
lsvData.Items.Count].Points[0].YValues[0] = BDC.Data[SelectedIndex].Ym;
                }
            }
            chart3.Series[SelectedIndex + lsvData.Items.Count].MarkerStyle =
MarkerStyle.Circle;
            chart3.Series[SelectedIndex + lsvData.Items.Count].MarkerSize = 10;
            chart3.Series[SelectedIndex + lsvData.Items.Count].MarkerBorderWidth
= 2;
```

ภาคผนวก ข (ต่อ)

```
                chart3.Invalidate();
                #endregion
                #region Add Chart2
                chart2.Series[SelectedIndex].Points.AddXY(CurrentTheta,
CurrentLightIntensity);
                if (chart2.Series[SelectedIndex + lsvData.Items.Count].Points.Count <=
0)
                {
                    if (SelectedIndex == 0)
                        chart2.Series[SelectedIndex +
lsvData.Items.Count].Points.AddXY(BDC.Reference.Xm, BDC.Reference.Ym);
                    else
                        chart2.Series[SelectedIndex +
lsvData.Items.Count].Points.AddXY(BDC.Data[CurrentPointIndex].Xm,
BDC.Data[CurrentPointIndex].Ym);
                }
                else
                {
                    if (SelectedIndex == 0)
                    {
                        chart2.Series[SelectedIndex +
lsvData.Items.Count].Points[0].XValue = BDC.Reference.Xm;
                        chart2.Series[SelectedIndex +
lsvData.Items.Count].Points[0].YValues[0] = BDC.Reference.Ym;
                    }
                    else
                    {
                        chart2.Series[SelectedIndex +
lsvData.Items.Count].Points[0].XValue = BDC.Data[SelectedIndex].Xm;
```

ภาคผนวก ข (ต่อ)

```
                    chart2.Series[SelectedIndex +
lsvData.Items.Count].Points[0].YValues[0] = BDC.Data[SelectedIndex].Ym;
                }
            }
            chart2.Series[SelectedIndex + lsvData.Items.Count].MarkerStyle =
MarkerStyle.Circle;
            chart2.Series[SelectedIndex + lsvData.Items.Count].MarkerSize = 10;
            chart2.Series[SelectedIndex + lsvData.Items.Count].MarkerBorderWidth
= 2;
            chart2.Invalidate();
            #endregion
            #region Add Chart1
            chart1.Series[SelectedIndex].Points.AddXY(CurrentTheta,
CurrentLightIntensity);
            if (chart1.Series[SelectedIndex + lsvData.Items.Count].Points.Count <=
0)
            {
                if (SelectedIndex == 0)
                    chart1.Series[SelectedIndex +
lsvData.Items.Count].Points.AddXY(BDC.Reference.Xmax, BDC.Reference.Ymax);
                else
                    chart1.Series[SelectedIndex +
lsvData.Items.Count].Points.AddXY(BDC.Data[CurrentPointIndex].Xmax,
BDC.Data[CurrentPointIndex].Ymax);
            }
            else
            {
                if (SelectedIndex == 0)
                {
```

ภาคผนวก ข (ต่อ)

```
                    chart1.Series[SelectedIndex +
lsvData.Items.Count].Points[0].XValue = BDC.Reference.Xmax;
                    chart1.Series[SelectedIndex +
lsvData.Items.Count].Points[0].YValues[0] = BDC.Reference.Ymax;
                }
                else
                {
                    chart1.Series[SelectedIndex +
lsvData.Items.Count].Points[0].XValue = BDC.Data[SelectedIndex].Xmax;
                    chart1.Series[SelectedIndex +
lsvData.Items.Count].Points[0].YValues[0] = BDC.Data[SelectedIndex].Ymax;
                }
            }
            chart1.Series[SelectedIndex + lsvData.Items.Count].MarkerStyle =
MarkerStyle.Circle;
            chart1.Series[SelectedIndex + lsvData.Items.Count].MarkerSize = 10;
            chart1.Series[SelectedIndex + lsvData.Items.Count].MarkerBorderWidth
= 2;
            chart1.Invalidate();
            #endregion
            DefineAngleOfRotation();
            PlotReferenceCurve();
            PlotTreatmentsCurve();
            PlotSelectedTRTMarker();
            if (ThetaA < ThetaB)
            {
                if (ThetaB < CurrentTheta)
                {
                    IsScanning = false;
```

ภาคผนวก ข (ต่อ)

```
                }
            }
            else if (ThetaA > ThetaB)
            {
                if (CurrentTheta < ThetaB)
                {
                    IsScanning = false;
                }
            }
            CurrentPointIndex += 1;
        }
        if (btnPause.Text != "CONTINUE" )
        {
            if (!mnuOptionsDemomode.Checked)
            {
                MSG = "A:WP" + System.Convert.ToInt32(-1 * ThetaA /
StepFactor).ToString() + "P" + System.Convert.ToInt32(-1 * ThetaA /
StepFactor).ToString();
                MMC.WriteString(MSG);
                DisconnectDevices();
            }
            btnStart.Enabled = true;
            btnPause.Enabled = false;
            btnStop.Enabled = false;
            btnNew.Enabled = true;
            btnOpen.Enabled = true;
            gbSample.Enabled = true;
            gbScanCondition.Enabled = true;
            gbStartMea.Enabled = true;
```

ภาคผนวก ข (ต่อ)

```
            gbDevices.Enabled = true;
            gbMeasurement.Enabled = true;
        }
        else
        {
            if (!mnuOptionsDemomode.Checked)
            {
                DisconnectDevices();
            }
            btnStart.Enabled = false;
            btnPause.Enabled = true;
            btnStop.Enabled = true;
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
        StopScanning();
        btnStart.Enabled = true;
        btnPause.Enabled = false;
        btnStop.Enabled = false;
        btnNew.Enabled = true;
        btnOpen.Enabled = true;
        gbMeasurement.Enabled = true;
        gbStartMea.Enabled = true;
        gbSample.Enabled = true;
        gbDevices.Enabled = true;
        gbScanCondition.Enabled = true;
```

ภาคผนวก ข (ต่อ)

```
        }
    }
    private void StopScanning()
    {
        IsScanning = false;
        IsContinuing = false;
    }
    private void DoPasuseScanning()
    {
        IsScanning = false;
        IsContinuing = false;
    }
    private void DoContinueScanning()
    {
        IsScanning = true;
        IsContinuing = true;
    }
    #endregion
    #region Menu
    #region File
    private void newToolStripMenuItem1_Click(object sender, EventArgs e)
    {
        NewMeasurement();
    }
    private void openToolStripMenuItem_Click(object sender, EventArgs e)
    {
        BDC.OpenFile();
    }
```

ภาคผนวก ข (ต่อ)

```
        private void saveToolStripMenuItem_Click(object sender, EventArgs e)
        {
            if (lsvData.Items.Count > 0)
            {
                DialogResult result = MessageBox.Show("Data will be deleted. Do you
want to save file?", "Save file", MessageBoxButtons.OKCancel,
MessageBoxIcon.Question);
                if (result == DialogResult.OK)
                {
                    SaveData();
                }
                else if (result == DialogResult.Cancel)
                {

                }
            }
        }
        private void exitToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Close();
        }
        #endregion
        #region Measure
        private void mnuStartToolStripMenuItem_Click(object sender, EventArgs e)
        {
            DoStart();
        }
        private void mnuStopToolStripMenuItem_Click(object sender, EventArgs e)
        {
```

ภาคผนวก ข (ต่อ)

```
            DoStop();
        }
        private void mnuPauseToolStripMenuItem_Click(object sender, EventArgs e)
        {
            DoPause();
        }
        private void mnuCoutinewToolStripMenuItem_Click(object sender, EventArgs e)
        {
            DoPause();
        }
        #endregion
        #region Devices
        private void ConnectToolStripMenuItem_Click(System.Object sender,
System.EventArgs e)
        {
            ConnectedDevices();
        }
        private void DisconnectToolStripMenuItem_Click(System.Object sender,
System.EventArgs e)
        {
            DisconnectDevices();
        }
        private void mnuDevicesClearDMM_Click(System.Object sender,
System.EventArgs e)
        {
            try
            {
                ConnectedDevices();
```

ภาคผนวก ข (ต่อ)

```
            DMM.WriteString("*CLS");
            DisconnectDevices();
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }
    private void mnuDevicesResetDMM_Click(System.Object sender,
System.EventArgs e)
    {
        try
        {
            ConnectedDevices();
            DMM.WriteString("*RST");
            DisconnectDevices();
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }
    #endregion
    private void AboutToolStripMenuItem_Click(object sender, EventArgs e)
    {
        MessageBox.Show("Polarimeter2019 program. (c)2019, Physics, KMITL. Design
by S. Saejia.");
    }
```

ภาคผนวก ข (ต่อ)

```
        private void colorTableToolStripMenuItem_Click(object sender, EventArgs e)
        {
            frmColorTable f = new frmColorTable(this);
            DialogResult result = f.ShowDialog();
            if (result == DialogResult.OK)
            {
                ResetDynaplot();
                PlotReferenceCurve();
                PlotTreatmentsCurve();
                PlotSelectedTRTMarker();
            }
        }
        #endregion
        #region Form Event
        private void frmMain_Load(object sender, EventArgs e)
        {
            this.WindowState = FormWindowState.Maximized;
            LoadSetting();
            gbSample.Enabled = false;
            gbMeasurement.Enabled = false;
            gbScanCondition.Enabled = false;
            label10.Hide();
            label11.Hide();
            label12.Hide();
            label13.Hide();
        }
        private void Form1_FormClosing(object sender,
    System.Windows.Forms.FormClosingEventArgs e)
```

ภาคผนวก ข (ต่อ)

```
        {
            DialogResult result = MessageBox.Show("Do you want to quit program?",
"Quit", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
            if (result == DialogResult.Yes)
            {
                IsScanning = false;
                SaveSetting();
            }
            else if (result == DialogResult.No)
            {
                e.Cancel = true;
            }
        }
        #endregion
        #region PLOT when select curve
        private void ResetDynaplot()
        {
            try
            {
                for (int i = 0; i <= NumberOfRepeatation - 1; i++)
                {
                }
            }
            catch (Exception ex)
            {
            }
        }
        private bool PlotReferenceCurve()
```

ภาคผนวก ข (ต่อ)

```
{
    bool e = false;
    try
    {
        if (lsvData.Items[0].Checked == true)
        {
            if (BDC.Reference.X != null)
            {
                lblNullPoint.Text = BDC.Reference.Xm.ToString("0.0000") + " deg";
            }
            else
            {
            }
        }
    }
    catch (Exception ex)
    {

    }
    return e;
}
private bool PlotTreatmentsCurve()
{
    if (BDC == null)
    {
        return false;
    }
    if (BDC.Data.ToString() == null)
```

ภาคผนวก ข (ต่อ)

```
            {
                return false;
            }
            for (int i = 0; i <= NumberOfRepeatation - 1; i++)
            {
                try
                {
                    if (lsvData.Items[i + 1].Checked)
                    {

                    }
                }
                catch (Exception ex)
                {

                }
            }
            return true;
        }
        private void PlotSelectedTRTMarker()
        {
            try
            {
                if (lsvData.Items[SelectedIndex].Checked)
                {
                    lblNullPoint.Text = BDC.Data[SelectedIndex].Xm.ToString("0.0000") + "
deg";
                }
                Else
```

ภาคผนวก ข (ต่อ)

```
                {
                    lblNullPoint.Text = BDC.Data[SelectedIndex].Xm.ToString() + "deg";
                }
            }
            catch (Exception ex)
            {

            }
        }
        #endregion
        #region Sub-Routine
        private void NewMeasurement()
        {
            if (lsvData.Items.Count > 0)
            {
                DialogResult result = MessageBox.Show("Data will be deleted. Do you
want to new measurement?", "New measurement", MessageBoxButtons.OKCancel,
MessageBoxIcon.Question);
                if (result == DialogResult.OK)
                {
                    MEA();
                }
                else if (result == DialogResult.Cancel)
                {
                }
            }
            else
            {
```

ภาคผนวก ข (ต่อ)

```
            MEA();
        }
    }
    private void DefineAngleOfRotation()
    {
        ListViewItem lvi;
        lvi = lsvData.Items[SelectedIndex];
        if (SelectedIndex == 0)
        {
            lvi.SubItems[1].Text = "(" + BDC.Reference.Xm.ToString("0.00") + ", " +
BDC.Reference.Ym.ToString("0.0000") + ")";
        }
        else
        {
            lvi = lsvData.Items[SelectedIndex];
            lvi.SubItems[1].Text = "(" + BDC.Data[SelectedIndex].Xm.ToString("0.00") +
", " + BDC.Data[SelectedIndex].Ym.ToString("0.0000") + ")";
            lvi.SubItems[2].Text =
BDC.Data[SelectedIndex].AngleOfRotation.ToString("0.00");
        }
    }
    private void LoadSetting()
    {
        txtVoltageRange.Text = Properties.Settings.Default.VoltageRange.ToString();
        txtVoltageResolution.Text =
Properties.Settings.Default.VoltageResolution.ToString();
        mnuOptionsDemomode.Checked = Properties.Settings.Default.IsDemo;
        ReferenceColor = Properties.Settings.Default.ReferenceColor;
```

ภาคผนวก ข (ต่อ)

```
        ColorTable[0] = Properties.Settings.Default.color1;
        ColorTable[1] = Properties.Settings.Default.color2;
        ColorTable[2] = Properties.Settings.Default.color3;
        ColorTable[3] = Properties.Settings.Default.color4;
        ColorTable[4] = Properties.Settings.Default.color5;
        ColorTable[5] = Properties.Settings.Default.color6;
        ColorTable[6] = Properties.Settings.Default.color7;
        ColorTable[7] = Properties.Settings.Default.color8;
        ColorTable[8] = Properties.Settings.Default.color9;
        ColorTable[9] = Properties.Settings.Default.color10;
        ColorTable[10] = Properties.Settings.Default.color11;
        ColorTable[11] = Properties.Settings.Default.color12;
        ColorTable[12] = Properties.Settings.Default.color13;
        ColorTable[13] = Properties.Settings.Default.color14;
        ColorTable[14] = Properties.Settings.Default.color15;
        ColorTable[15] = Properties.Settings.Default.color16;
        ColorTable[16] = Properties.Settings.Default.color17;
        ColorTable[17] = Properties.Settings.Default.color18;
        ColorTable[18] = Properties.Settings.Default.color19;
        ColorTable[19] = Properties.Settings.Default.color20;
    }
    private void SaveSetting()
    {
        Properties.Settings.Default.IsDemo = mnuOptionsDemomode.Checked;
        Properties.Settings.Default.Save();
    }
    public void ApplyColorTableToSamples()
    {
```

ภาคผนวก ข (ต่อ)

```
            try
            {
                lsvData.Items[0].BackColor = ReferenceColor;
                for (int i = 1; i <= lsvData.Items.Count - 1; i++)
                {
                    lsvData.Items[i].BackColor = ColorTable[(i - 1) % ColorTable.Length];
                }
            }
            catch (Exception ex)
            {
            }
        }
        #endregion
        #region Summary
        private void lsvData_ItemSelectionChanged(object sender,
System.Windows.Forms.ListViewItemSelectionChangedEventArgs e)
        {
            if (lsvData.SelectedIndices == null)
            {
                return;
            }
            if (lsvData.SelectedIndices.Count <= 0)
            {
                return;
            }
            SelectedIndex = lsvData.SelectedIndices[0];
            try
            {
```

ภาคผนวก ข (ต่อ)

```
            if (SelectedIndex == 0)
            {
                lblSample.Text = "Reference";
            }
            else
            {
                lblSample.Text = "Sample " + SelectedIndex.ToString();
            }
            ResetDynaplot();
            PlotReferenceCurve();
            PlotTreatmentsCurve();
            PlotSelectedTRTMarker();
        }
        catch (Exception ex)
        {
        }
    }
    #endregion
    #region Textbox
    private void txtStart_KeyPress(object sender, KeyPressEventArgs e)
    {
        if ((e.KeyChar == '.') && ((sender as TextBox).Text.IndexOf('.') > -1))
        {
            e.Handled = true;
        }
    }
    private void txtStop_KeyPress(object sender, KeyPressEventArgs e)
    {
```

ภาคผนวก ข (ต่อ)

```
        if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar) && (e.KeyChar != '.'))
        {
            e.Handled = true;
        }
        if ((e.KeyChar == '.') && ((sender as TextBox).Text.IndexOf('.') > -1))
        {
            e.Handled = true;
        }
    }
    private void txtResoluton_KeyPress(object sender, KeyPressEventArgs e)
    {
        if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar) && (e.KeyChar != '.'))
        {
            e.Handled = true;
        }
        if ((e.KeyChar == '.') && ((sender as TextBox).Text.IndexOf('.') > -1))
        {
            e.Handled = true;
        }
    }
    #endregion
    private void SaveData()
    {
        SaveFileDialog dlg = new SaveFileDialog();
        dlg.Filter = "(csv)|*.csv";
        DialogResult redlg = dlg.ShowDialog();
        if (redlg != DialogResult.OK)
        {
```

ภาคผนวก ข (ต่อ)

```
MessageBox.Show("Bye","Save",MessageBoxButtons.OK,MessageBoxIcon.Information);
            return;
        }
        string path = dlg.FileName;
        LogFile.Log theSave = new LogFile.Log(path, "");
        frmNewMeasurement f = new frmNewMeasurement();
        LogFile.PolarimeterHeader header = new LogFile.PolarimeterHeader()
        {
            DMMPort = txtDMMAddress.Text,
            MMCPort = txtMMCAddress.Text,
            SampleName = txtSampleName.Text,
            SampleNumber = (int)f.OfRepeatation,
        };
        theSave.AppendText(header.ToString());
        foreach (ListViewItem lvi in lsvData.Items)
        {
            theSave.AppendText($"{lvi.Index +
1},{lvi.Text},{lvi.SubItems[1].Text},{lvi.SubItems[2].Text}");
        }
        theSave.AppendText("[Reference]");
        for (int i = 0; i < BDC.Reference.X.Length - 1; i++)
        {
            theSave.AppendText(BDC.Reference.X[i].ToString() + "," +
BDC.Reference.Y[i].ToString());
        }
        for (int k = 1; k <= BDC.Data.Length - 1; k++)
        {
```

ภาคผนวก ข (ต่อ)

```
                theSave.AppendText("[Sample " + k.ToString() + "]");
                for (int i = 0; i < BDC.Data[k].X.Length - 1; i++)
                {
                    theSave.AppendText(BDC.Data[k].X[i].ToString() + "," +
BDC.Data[k].Y[i].ToString().ToString());
                }
            }
        }
        private void PolarChart()
        {
            chart1.Series.Clear();
            chart2.Series.Clear();
            chart3.Series.Clear();
            chart4.Series.Clear();
            try
            {
                #region Chart1
                Series newSeries = new Series("Reference");
                newSeries.ChartType = SeriesChartType.Polar;
                newSeries.BorderWidth = 3;
                newSeries.Color = Properties.Settings.Default.ReferenceColor;
                chart1.Series.Add(newSeries);
                for (int i = 1; i <= NumberOfRepeatation; i++)
                {
                    Series sample = new Series("Sample" + i.ToString());
                    sample.ChartType = SeriesChartType.Polar;
                    sample.BorderWidth = 3;
                    sample.XValueType = ChartValueType.Auto;
```

ภาคผนวก ข (ต่อ)

```
            sample.YValueType = ChartValueType.Auto;
            chart1.Series.Add(sample);
            sample.Color = ColorTable[(i - 1) % ColorTable.Length];
        }
        Series newSeriesRefMarker = new Series("Reference_Marker");
        newSeriesRefMarker.ChartType = SeriesChartType.Polar;
        newSeriesRefMarker.Color = Properties.Settings.Default.ReferenceColor;
        chart1.Series.Add(newSeriesRefMarker);
        for (int i = 1; i <= NumberOfRepeatation; i++)
        {
            Series sampleMarker = new Series("sample_Marker" + i.ToString());
            sampleMarker.ChartType = SeriesChartType.Polar;
            sampleMarker.BorderWidth = 3;
            sampleMarker.XValueType = ChartValueType.Auto;
            sampleMarker.YValueType = ChartValueType.Auto;
            chart1.Series.Add(sampleMarker);
            sampleMarker.Color = ColorTable[(i - 1) % ColorTable.Length];
        }
        chart1.ChartAreas[0].Axes[0].LabelStyle.Format = "{0:0.00}";
        chart1.ChartAreas[0].Axes[0].MajorGrid.LineColor = Color.DarkGray;
        chart1.ChartAreas[0].Axes[0].MajorGrid.Enabled = true;
        chart1.ChartAreas[0].Axes[0].MajorGrid.Interval = 10;
        chart1.ChartAreas[0].Axes[0].MajorTickMark.Enabled = true;
        chart1.ChartAreas[0].Axes[0].MajorTickMark.Interval = 10;
        chart1.ChartAreas[0].Axes[0].MajorTickMark.LineWidth = 2;
        chart1.ChartAreas[0].Axes[0].MajorTickMark.Size = 1;
        chart1.ChartAreas[0].Axes[0].MajorTickMark.TickMarkStyle =
TickMarkStyle.InsideArea;
```

ภาคผนวก ข (ต่อ)

```
            chart1.ChartAreas[0].Axes[0].MinorGrid.LineColor = Color.LightGray;
            chart1.ChartAreas[0].Axes[0].MinorGrid.LineDashStyle =
ChartDashStyle.Dash;
            chart1.ChartAreas[0].Axes[0].MinorGrid.Enabled = true;
            chart1.ChartAreas[0].Axes[0].MinorGrid.Interval = 2;
            chart1.ChartAreas[0].Axes[0].MinorTickMark.Enabled = true;
            chart1.ChartAreas[0].Axes[0].MinorTickMark.LineColor = Color.Gray;
            chart1.ChartAreas[0].Axes[0].MinorTickMark.Interval = 2;
            chart1.ChartAreas[0].Axes[0].MinorTickMark.LineWidth = 1;
            chart1.ChartAreas[0].Axes[0].MinorTickMark.Size = 1;
            chart1.ChartAreas[0].Axes[0].MinorTickMark.TickMarkStyle =
TickMarkStyle.InsideArea;
            chart1.ChartAreas[0].Axes[1].LabelStyle.Format = "0.00";
            chart1.ChartAreas[0].Axes[1].MajorGrid.LineColor = Color.DarkGray;
            chart1.ChartAreas[0].Axes[1].MajorGrid.Enabled = true;
            chart1.ChartAreas[0].Axes[1].MajorGrid.Interval = 0.5;
            chart1.ChartAreas[0].Axes[1].MajorTickMark.Enabled = true;
            chart1.ChartAreas[0].Axes[1].MajorTickMark.Interval = 0.5;
            chart1.ChartAreas[0].Axes[1].MajorTickMark.LineWidth = 2;
            chart1.ChartAreas[0].Axes[1].MajorTickMark.Size = 1;
            chart1.ChartAreas[0].Axes[1].MajorTickMark.TickMarkStyle =
TickMarkStyle.InsideArea;
            chart1.ChartAreas[0].Axes[1].MinorGrid.LineColor = Color.LightGray;
            chart1.ChartAreas[0].Axes[1].MinorGrid.LineDashStyle =
ChartDashStyle.Dash;
            chart1.ChartAreas[0].Axes[1].MinorGrid.Enabled = true;
            chart1.ChartAreas[0].Axes[1].MinorGrid.Interval = 0.25;
            chart1.ChartAreas[0].Axes[1].MinorTickMark.Enabled = true;
            chart1.ChartAreas[0].Axes[1].MinorTickMark.LineColor = Color.Gray;
            chart1.ChartAreas[0].Axes[1].MinorTickMark.Interval = 0.1;
```

ภาคผนวก ข (ต่อ)

```
            chart1.ChartAreas[0].Axes[1].MinorTickMark.LineWidth = 1;
            chart1.ChartAreas[0].Axes[1].MinorTickMark.Size = 1;
            chart1.ChartAreas[0].Axes[1].MinorTickMark.TickMarkStyle =
TickMarkStyle.InsideArea;
            #endregion
            #region Chart2
            Series newSeries2 = new Series("Reference");
            newSeries2.ChartType = SeriesChartType.Line;
            newSeries2.BorderWidth = 3;
            newSeries2.Color = Properties.Settings.Default.ReferenceColor;
            chart2.Series.Add(newSeries2);
            for (int i = 1; i <= NumberOfRepeatation; i++)
            {
                Series sample = new Series("Sample" + i.ToString());
                sample.ChartType = SeriesChartType.Line;
                sample.BorderWidth = 3;
                sample.XValueType = ChartValueType.Auto;
                sample.YValueType = ChartValueType.Auto;
                chart2.Series.Add(sample);
                sample.Color = ColorTable[(i - 1) % ColorTable.Length];
            }
            Series newSeriesRefMarker1 = new Series("Reference_Marker");
            newSeriesRefMarker1.ChartType = SeriesChartType.Line;
            newSeriesRefMarker1.Color = Properties.Settings.Default.ReferenceColor;
            chart2.Series.Add(newSeriesRefMarker1);
            for (int i = 1; i <= NumberOfRepeatation; i++)
            {
                Series sampleMarker = new Series("sample_Marker" + i.ToString());
```

ภาคผนวก ข (ต่อ)

```
                sampleMarker.ChartType = SeriesChartType.Line;
                sampleMarker.BorderWidth = 3;
                sampleMarker.XValueType = ChartValueType.Auto;
                sampleMarker.YValueType = ChartValueType.Auto;
                chart2.Series.Add(sampleMarker);
                sampleMarker.Color = ColorTable[(i - 1) % ColorTable.Length];
            }
            chart2.ChartAreas[0].Axes[0].Minimum = Convert.ToDouble(txtStart.Text);
            chart2.ChartAreas[0].Axes[0].LabelStyle.Format = "{0:0.00}";
            chart2.ChartAreas[0].Axes[0].MajorGrid.LineColor = Color.DarkGray;
            chart2.ChartAreas[0].Axes[0].MajorGrid.Enabled = true;
            chart2.ChartAreas[0].Axes[0].MajorGrid.Interval = 10;
            chart2.ChartAreas[0].Axes[0].MajorTickMark.Enabled = true;
            chart2.ChartAreas[0].Axes[0].MajorTickMark.Interval = 10;
            chart2.ChartAreas[0].Axes[0].MajorTickMark.LineWidth = 2;
            chart2.ChartAreas[0].Axes[0].MajorTickMark.Size = 1;
            chart2.ChartAreas[0].Axes[0].MajorTickMark.TickMarkStyle =
TickMarkStyle.InsideArea;
            chart2.ChartAreas[0].Axes[0].MinorGrid.LineColor = Color.LightGray;
            chart2.ChartAreas[0].Axes[0].MinorGrid.LineDashStyle =
ChartDashStyle.Dash;
            chart2.ChartAreas[0].Axes[0].MinorGrid.Enabled = true;
            chart2.ChartAreas[0].Axes[0].MinorGrid.Interval = 2;
            chart2.ChartAreas[0].Axes[0].MinorTickMark.Enabled = true;
            chart2.ChartAreas[0].Axes[0].MinorTickMark.LineColor = Color.Gray;
            chart2.ChartAreas[0].Axes[0].MinorTickMark.Interval = 2;
            chart2.ChartAreas[0].Axes[0].MinorTickMark.LineWidth = 1;
            chart2.ChartAreas[0].Axes[0].MinorTickMark.Size = 1;
```

ภาคผนวก ข (ต่อ)

```
            chart2.ChartAreas[0].Axes[0].MinorTickMark.TickMarkStyle =
TickMarkStyle.InsideArea;
            chart2.ChartAreas[0].Axes[1].LabelStyle.Format = "0.00";
            chart2.ChartAreas[0].Axes[1].MajorGrid.LineColor = Color.DarkGray;
            chart2.ChartAreas[0].Axes[1].MajorGrid.Enabled = true;
            chart2.ChartAreas[0].Axes[1].MajorGrid.Interval = 0.5;
            chart2.ChartAreas[0].Axes[1].MajorTickMark.Enabled = true;
            chart2.ChartAreas[0].Axes[1].MajorTickMark.Interval = 0.5;
            chart2.ChartAreas[0].Axes[1].MajorTickMark.LineWidth = 2;
            chart2.ChartAreas[0].Axes[1].MajorTickMark.Size = 1;
            chart2.ChartAreas[0].Axes[1].MajorTickMark.TickMarkStyle =
TickMarkStyle.InsideArea;
            chart2.ChartAreas[0].Axes[1].MinorGrid.LineColor = Color.LightGray;
            chart2.ChartAreas[0].Axes[1].MinorGrid.LineDashStyle =
ChartDashStyle.Dash;
            chart2.ChartAreas[0].Axes[1].MinorGrid.Enabled = true;
            chart2.ChartAreas[0].Axes[1].MinorGrid.Interval = 0.25;
            chart2.ChartAreas[0].Axes[1].MinorTickMark.Enabled = true;
            chart2.ChartAreas[0].Axes[1].MinorTickMark.LineColor = Color.Gray;
            chart2.ChartAreas[0].Axes[1].MinorTickMark.Interval = 0.1;
            chart2.ChartAreas[0].Axes[1].MinorTickMark.LineWidth = 1;
            chart2.ChartAreas[0].Axes[1].MinorTickMark.Size = 1;
            chart2.ChartAreas[0].Axes[1].MinorTickMark.TickMarkStyle =
TickMarkStyle.InsideArea;
            #endregion
            #region Chart3
            Series newSeries3 = new Series("Reference");
            newSeries3.ChartType = SeriesChartType.Line;
            newSeries3.BorderWidth = 3;
```

ภาคผนวก ข (ต่อ)

```
newSeries3.Color = Properties.Settings.Default.ReferenceColor;
chart3.Series.Add(newSeries3);
for (int i = 1; i <= NumberOfRepeatation; i++)
{
    Series sample = new Series("Sample" + i.ToString());
    sample.ChartType = SeriesChartType.Line;
    sample.BorderWidth = 3;
    sample.XValueType = ChartValueType.Auto;
    sample.YValueType = ChartValueType.Auto;
    chart3.Series.Add(sample);
    sample.Color = ColorTable[(i - 1) % ColorTable.Length];
}
Series newSeriesRefMarker2 = new Series("Reference_Marker");
newSeriesRefMarker2.ChartType = SeriesChartType.Line;
newSeriesRefMarker2.Color = Properties.Settings.Default.ReferenceColor;
chart3.Series.Add(newSeriesRefMarker2);
for (int i = 1; i <= NumberOfRepeatation; i++)
{
    Series sampleMarker = new Series("sample_Marker" + i.ToString());
    sampleMarker.ChartType = SeriesChartType.Line;
    sampleMarker.BorderWidth = 3;
    sampleMarker.XValueType = ChartValueType.Auto;
    sampleMarker.YValueType = ChartValueType.Auto;
    chart3.Series.Add(sampleMarker);
    sampleMarker.Color = ColorTable[(i - 1) % ColorTable.Length];
}
chart3.ChartAreas[0].Axes[0].Minimum = Convert.ToDouble(txtStart.Text);
chart3.ChartAreas[0].Axes[0].Interval = 30;
```

ภาคผนวก ข (ต่อ)

```
            chart3.ChartAreas[0].Axes[1].Interval = 0.5;
            chart3.ChartAreas[0].Axes[0].LabelStyle.Format = "{0:0.00}";
            chart3.ChartAreas[0].Axes[0].MajorGrid.LineColor = Color.DarkGray;
            chart3.ChartAreas[0].Axes[0].MajorGrid.Enabled = true;
            chart3.ChartAreas[0].Axes[0].MajorGrid.Interval = 10;
            chart3.ChartAreas[0].Axes[0].MajorTickMark.Enabled = true;
            chart3.ChartAreas[0].Axes[0].MajorTickMark.Interval = 10;
            chart3.ChartAreas[0].Axes[0].MajorTickMark.LineWidth = 2;
            chart3.ChartAreas[0].Axes[0].MajorTickMark.Size = 1;
            chart3.ChartAreas[0].Axes[0].MajorTickMark.TickMarkStyle =
TickMarkStyle.InsideArea;
            chart3.ChartAreas[0].Axes[0].MinorGrid.LineColor = Color.LightGray;
            chart3.ChartAreas[0].Axes[0].MinorGrid.LineDashStyle =
ChartDashStyle.Dash;
            chart3.ChartAreas[0].Axes[0].MinorGrid.Enabled = true;
            chart3.ChartAreas[0].Axes[0].MinorGrid.Interval = 2;
            chart3.ChartAreas[0].Axes[0].MinorTickMark.Enabled = true;
            chart3.ChartAreas[0].Axes[0].MinorTickMark.LineColor = Color.Gray;
            chart3.ChartAreas[0].Axes[0].MinorTickMark.Interval = 2;
            chart3.ChartAreas[0].Axes[0].MinorTickMark.LineWidth = 1;
            chart3.ChartAreas[0].Axes[0].MinorTickMark.Size = 1;
            chart3.ChartAreas[0].Axes[0].MinorTickMark.TickMarkStyle =
TickMarkStyle.InsideArea;
            chart3.ChartAreas[0].Axes[1].MajorGrid.LineColor = Color.DarkGray;
            chart3.ChartAreas[0].Axes[1].LabelStyle.Format = "0.00";
            chart3.ChartAreas[0].Axes[1].MajorGrid.Enabled = true;
            chart3.ChartAreas[0].Axes[1].MajorGrid.Interval = 0.5;
            chart3.ChartAreas[0].Axes[1].MajorTickMark.Enabled = true;
            chart3.ChartAreas[0].Axes[1].MajorTickMark.Interval = 0.5;
```

ภาคผนวก ข (ต่อ)

```
            chart3.ChartAreas[0].Axes[1].MajorTickMark.LineWidth = 2;
            chart3.ChartAreas[0].Axes[1].MajorTickMark.Size = 1;
            chart3.ChartAreas[0].Axes[1].MajorTickMark.TickMarkStyle =
TickMarkStyle.InsideArea;
            chart3.ChartAreas[0].Axes[1].MinorGrid.LineColor = Color.LightGray;
            chart3.ChartAreas[0].Axes[1].MinorGrid.LineDashStyle =
ChartDashStyle.Dash;
            chart3.ChartAreas[0].Axes[1].MinorGrid.Enabled = true;
            chart3.ChartAreas[0].Axes[1].MinorGrid.Interval = 0.1;
            chart3.ChartAreas[0].Axes[1].MinorTickMark.Enabled = true;
            chart3.ChartAreas[0].Axes[1].MinorTickMark.LineColor = Color.Gray;
            chart3.ChartAreas[0].Axes[1].MinorTickMark.Interval = 0.1;
            chart3.ChartAreas[0].Axes[1].MinorTickMark.LineWidth = 1;
            chart3.ChartAreas[0].Axes[1].MinorTickMark.Size = 1;
            chart3.ChartAreas[0].Axes[1].MinorTickMark.TickMarkStyle =
TickMarkStyle.InsideArea;
            #endregion
            #region Chart4
            Series newSeries4 = new Series("Reference");
            newSeries4.ChartType = SeriesChartType.Polar;
            newSeries4.BorderWidth = 3;
            newSeries4.Color = Properties.Settings.Default.ReferenceColor;
            chart4.Series.Add(newSeries4);
            for (int i = 1; i <= NumberOfRepeatation; i++)
            {
                Series sample = new Series("Sample" + i.ToString());
                sample.ChartType = SeriesChartType.Polar;
                sample.BorderWidth = 3;
                sample.XValueType = ChartValueType.Auto;
```

ภาคผนวก ข (ต่อ)

```
        sample.YValueType = ChartValueType.Auto;
        chart4.Series.Add(sample);
        sample.Color = ColorTable[(i - 1) % ColorTable.Length];
    }
    Series newSeriesRefMarker3 = new Series("Reference_Marker");
    newSeriesRefMarker3.ChartType = SeriesChartType.Polar;
    newSeriesRefMarker3.Color = Properties.Settings.Default.ReferenceColor;
    chart4.Series.Add(newSeriesRefMarker3);
    for (int i = 1; i <= NumberOfRepeatation; i++)
    {
        Series sampleMarker = new Series("sample_Marker" + i.ToString());
        sampleMarker.ChartType = SeriesChartType.Polar;
        sampleMarker.BorderWidth = 3;
        sampleMarker.XValueType = ChartValueType.Auto;
        sampleMarker.YValueType = ChartValueType.Auto;
        chart4.Series.Add(sampleMarker);
        sampleMarker.Color = ColorTable[(i - 1) % ColorTable.Length];
    }
    chart4.ChartAreas[0].Axes[0].Interval = 10;
    chart4.ChartAreas[0].Axes[1].Interval = 0.5;
    chart4.ChartAreas[0].Axes[0].LabelStyle.Format = "{0:0.00} ";
    chart4.ChartAreas[0].Axes[0].MajorGrid.LineColor = Color.DarkGray;
    chart4.ChartAreas[0].Axes[0].MajorGrid.Enabled = true;
    chart4.ChartAreas[0].Axes[0].MajorGrid.Interval = 10;
    chart4.ChartAreas[0].Axes[0].MajorTickMark.Enabled = true;
    chart4.ChartAreas[0].Axes[0].MajorTickMark.Interval = 10;
    chart4.ChartAreas[0].Axes[0].MajorTickMark.LineWidth = 2;
    chart4.ChartAreas[0].Axes[0].MajorTickMark.Size = 1;
```

ภาคผนวก ข (ต่อ)

```
            chart4.ChartAreas[0].Axes[0].MajorTickMark.TickMarkStyle =
TickMarkStyle.InsideArea;
            chart4.ChartAreas[0].Axes[0].MinorGrid.LineColor = Color.LightGray;
            chart4.ChartAreas[0].Axes[0].MinorGrid.LineDashStyle =
ChartDashStyle.Dash;
            chart4.ChartAreas[0].Axes[0].MinorGrid.Enabled = true;
            chart4.ChartAreas[0].Axes[0].MinorGrid.Interval = 2;
            chart4.ChartAreas[0].Axes[0].MinorTickMark.Enabled = true;
            chart4.ChartAreas[0].Axes[0].MinorTickMark.LineColor = Color.Gray;
            chart4.ChartAreas[0].Axes[0].MinorTickMark.Interval = 2;
            chart4.ChartAreas[0].Axes[0].MinorTickMark.LineWidth = 1;
            chart4.ChartAreas[0].Axes[0].MinorTickMark.Size = 1;
            chart4.ChartAreas[0].Axes[0].MinorTickMark.TickMarkStyle =
TickMarkStyle.InsideArea;
            chart4.ChartAreas[0].Axes[1].LabelStyle.Format = "0.00 ";
            chart4.ChartAreas[0].Axes[1].MajorGrid.LineColor = Color.DarkGray;
            chart4.ChartAreas[0].Axes[1].MajorGrid.Enabled = true;
            chart4.ChartAreas[0].Axes[1].MajorGrid.Interval = 0.5;
            chart4.ChartAreas[0].Axes[1].MajorTickMark.Enabled = true;
            chart4.ChartAreas[0].Axes[1].MajorTickMark.Interval = 0.5;
            chart4.ChartAreas[0].Axes[1].MajorTickMark.LineWidth = 2;
            chart4.ChartAreas[0].Axes[1].MajorTickMark.Size = 1;
            chart4.ChartAreas[0].Axes[1].MajorTickMark.TickMarkStyle =
TickMarkStyle.InsideArea;
            chart4.ChartAreas[0].Axes[1].MinorGrid.LineColor = Color.LightGray;
            chart4.ChartAreas[0].Axes[1].MinorGrid.LineDashStyle =
ChartDashStyle.Dash;
            chart4.ChartAreas[0].Axes[1].MinorGrid.Enabled = true;
            chart4.ChartAreas[0].Axes[1].MinorGrid.Interval = 0.1;
            chart4.ChartAreas[0].Axes[1].MinorTickMark.Enabled = true;
```

ภาคผนวก ข (ต่อ)

```
            chart4.ChartAreas[0].Axes[1].MinorTickMark.LineColor = Color.Gray;
            chart4.ChartAreas[0].Axes[1].MinorTickMark.Interval = 0.1;
            chart4.ChartAreas[0].Axes[1].MinorTickMark.LineWidth = 1;
            chart4.ChartAreas[0].Axes[1].MinorTickMark.Size = 1;
            chart4.ChartAreas[0].Axes[1].MinorTickMark.TickMarkStyle =
TickMarkStyle.InsideArea;
            #endregion
        }
        catch (Exception ex)
        {
            MessageBox.Show($"{ex.Message} {ex.StackTrace}");
        }
    }
    private void MEA()
    {
        frmNewMeasurement f = new frmNewMeasurement();
        f.StartPosition = FormStartPosition.CenterScreen;
        DialogResult result = f.ShowDialog();
        if (result == DialogResult.OK)
        {
            if (f.Verify() == true)
            {
                try
                {
                    txtSampleName.Text = f.SampleName;
                    numRepeatNumber.Value = f.RepeatNumber;
                    NumberOfRepeatation = (int)f.OfRepeatation;
                    BDC = new BaseDataControl();
```

ภาคผนวก ข (ต่อ)

```
                BDC.SampleName = txtSampleName.Text;
                BDC.Data = new
BaseDataControl.strucCurveData[NumberOfRepeatation+1];
                lsvData.Items.Clear();
                ListViewItem lvi;
                lvi = new ListViewItem();
                lvi.Text = "Reference";
                lvi.SubItems.Add("-");
                lvi.SubItems.Add("-");
                lvi.SubItems.Add("-");
                lvi.Checked = true;
                lvi.BackColor = ReferenceColor;
                lvi.UseItemStyleForSubItems = false;
                lsvData.Items.Add(lvi);
                for (int i = 1; i <= NumberOfRepeatation; i++)
                {
                    lvi = new ListViewItem();
                    lvi.Text = "Sample " + i.ToString();
                    lvi.SubItems.Add("-");
                    lvi.SubItems.Add("-");
                    lvi.SubItems.Add("-");
                    lvi.Checked = true;
                    lvi.BackColor = ColorTable[(i - 1) % ColorTable.Length];
                    lvi.UseItemStyleForSubItems = false;
                    lsvData.Items.Add(lvi);
                }
                gbMeasurement.Enabled = true;
                btnNew.Enabled = true;
```

ภาคผนวก ข (ต่อ)

```
                btnOpen.Enabled = true;
                gbSample.Enabled = true;
                gbScanCondition.Enabled = true;
                lsvData.Items[0].Selected = true;
                lsvData.CheckBoxes = true;
                lsvData.GridLines = true;
                lsvData.FullRowSelect = true;
                lsvData.AllowColumnReorder = true;
                lsvData.Focus();
                PolarChart();
                label10.Hide();
                label11.Hide();
                label12.Hide();
                label13.Hide();
            }
            catch
            {
            }
        }
    }
    else
    {
    }
}
private void btnRun_Click(object sender, EventArgs e)
{
    try
    {
```

ภาคผนวก ข (ต่อ)

```
            string MSG = "A:WP" + System.Convert.ToInt32(-1 *
Convert.ToDouble(txtStart.Text) / StepFactor).ToString() + "P" +
System.Convert.ToInt32(-1 * Convert.ToDouble(txtStart.Text) / StepFactor).ToString();
            MMC.WriteString(MSG);
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }
    private void btnNew_Click(object sender, EventArgs e)
    {
        if (lsvData.SelectedItems.Count > 0)
        {
            DialogResult result = MessageBox.Show("Do you want to save file before
new measurement?", "New file", MessageBoxButtons.YesNoCancel,
MessageBoxIcon.Question);
            {
                if (result == DialogResult.Yes)
                {
                    SaveData();
                    NewMeasurement();
                }
                else if (result == DialogResult.No)
                {
                    NewMeasurement();
                }
            }
        }
```

ภาคผนวก ข (ต่อ)

```
            else
            {
                NewMeasurement();
            }
        }
        private void btnOpen_Click(object sender, EventArgs e)
        {
            BDC.OpenFile();
        }
        private void btnConnect_Click(object sender, EventArgs e)
        {
            ConnectedDevices();
        }
        private void btnDisconnect_Click(object sender, EventArgs e)
        {
            DisconnectDevices();
        }
        private void lsvData_KeyPress(System.Object sender,
System.Windows.Forms.KeyPressEventArgs e)
        {
            if (e.KeyChar == Convert.ToChar(Keys.Enter))
            {
                SendKeys.Send("{TAB}");
            }
        }
        private void txtVoltageRange_TextChanged(object sender, System.EventArgs e)
        {
            Try
```

ภาคผนวก ข (ต่อ)

```
            {
                Properties.Settings.Default.VoltageRange =
Convert.ToDouble(txtVoltageRange.Text);
                Properties.Settings.Default.Save();
            }
            catch (Exception ex)
            {
            }
        }
        private void txtVoltageResolution_TextChanged(object sender,
System.EventArgs e)
        {
            try
            {
                Properties.Settings.Default.VoltageResolution =
Convert.ToDouble(txtVoltageResolution.Text);
                Properties.Settings.Default.Save();
            }
            catch (Exception ex)
            {
            }
        }
        private void lsvData_ItemCheck(object sender, ItemCheckEventArgs e)
        {
            ResetDynaplot();
            PlotReferenceCurve();
            PlotTreatmentsCurve();
            PlotSelectedTRTMarker();
        }
```

ภาคผนวก ข (ต่อ)

```
        private void lsvData_ItemChecked(object sender, ItemCheckedEventArgs e)
        {
        }
    }
}
```