การพัฒนาระบบควบคุมอาร์เอฟแมกนีตรอนสปัตเตอริง ผ่านหน้าจอสัมผัส

DEVELOPMENT OFRF MAGNETRON SPUTTERING CONTROL SYSTEM VIA TOUCH SCREEN

นายศุภวัฒน์ ยิ่งแจ่มศิริ

นางสาวศิรญา สมลือแสน

โครงงานพิเศษนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
ปริญญาวิทยาศาสตรบัณฑิต สาขาวิชาฟิสิกส์ประยุกต์
ภาควิชาฟิสิกส์ คณะวิทยาศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2557

การพัฒนาระบบควบคุมอาร์เอฟแมกนีตรอนสปัตเตอริง ผ่านหน้าจอสัมผัส

DEVELOPMENT OF RF MAGNETRON SPUTTERING CONTROL SYSTEM VIA TOUCH SCREEN

นายศุภวัฒน์ ยิ่งแจ่มศิริ
นางสาวศิรญา สมลือแสน

โครงงานพิเศษนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
ปริญญาวิทยาศาสตร์บัณฑิต สาขาวิชาฟิสิกส์ประยุกต์
ภาควิชาฟิสิกส์ คณะวิทยาศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2557

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

DEVELOPMENT OF RF MAGNETRON SPUTTERING CONTROL SYSTEM VIA TOUCH SCREEN

Mr. Suphawt Yingjamsiri

Miss. Siraya Somluesaen

A SPECIAL PROJECT SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF BACHELOR OF SCIENCE
IN APPILED PHYSICS DEPARTMENT OF PHYSICS
FACULTY OF SCIENCE
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
ACADEMIC YEAR 2010

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หัวข้อโครงงานพิเศษ	การพัฒนาระบบควบคุมอาร์เอฟแมกนีตรอนสปัตเตอริงผ่านหน้าจอสัมผัส DEVELOPMENT OF RF MAGNETRON SPUTTERING CONTROL SYSTEM VIA TOUCH SCREEN
ชื่อนักศึกษา	นายศุภวัฒน์ ยิ่งแจ่มศิริ รหัสนักศึกษา 54050603 นางสาวศิรญา สมลือแสน รหัสนักศึกษา 54050601
ปริญญา	วิทยาศาสตรบัณฑิต ฟิสิกส์ประยุกต์
ภาควิชา	ฟิสิกส์
ปีการศึกษา	2557
อาจารย์ที่ปรึกษา	รศ.ดร.วราวุฒิ เถาลัดดา
อาจารย์ที่ปรึกษาร่วม	ดร.ศ.ทิพวรรณ คล้ายบุญมี

คณะวิทยาศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง อนุมัติให้โครงงานพิเศษนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร วิทยาศาสตรบัณฑิต ฟิสิกส์ประยุกต์ ประจำปีการศึกษา 2557

คณะกรรมการสอบ	ลายมือชื่อ
อ.สุรชาติ กมลดิลก	
รศ.สาหร่าย เล็กชะอุ่ม	
อ.ธนภรณ์ ลีลาวัฒนานนท์	
รศ.ดร.วราวุฒิ เถาลัดดา	
ดร.ศ.ทิพวรรณ คล้ายบุญมี	

ลิขสิทธิ์ของคณะวิทยาศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หัวข้อโครงงานพิเศษ	การพัฒนาระบบควบคุมอาร์เอฟแมกนีตรอนสปัตเตอริงผ่านหน้าจอสัมผัส DEVELOPMENT OF RF MAGNETRON SPUTTERING CONTROL SYSTEM VIA TOUCH SCREEN
ชื่อนักศึกษา	นายศุภวัฒน์ ยิ่งแจ่มศิริ รหัสนักศึกษา 54050603 นางสาวศิรญา สมลือแสน รหัสนักศึกษา 54050601
ปริญญา	วิทยาศาสตรบัณฑิต ฟิสิกส์ประยุกต์
ภาควิชา	ฟิสิกส์
ปีการศึกษา	2557
อาจารย์ที่ปรึกษา	รศ.ดร.วราวุฒิ เถาลัดดา
อาจารย์ที่ปรึกษาร่วม	ดร.ศ.ทิพวรรณ คล้ายบุญมี

บทคัดย่อ

โครงงานพิเศษนี้เป็นการสร้างชุดควบคุมระบบอาร์เอฟแมกนีตรอนสปัตเตอริงผ่านหน้าจอสัมผัสโดยการดำเนินงานจะมีทั้งสิ้น 3 ส่วนใหญ่ๆ คือ 1. การออกแบบลายวงจรที่จะนำมาใช้รับส่งข้อมูลระหว่าง Microcontroller กับ IC ต่างๆเพื่อนำไปสั่งการหรือรับสัญญาณจากอุปกรณ์ต่างๆ 2. การเขียนโปรแกรมให้กับ Microcontroller เพื่อสงและรับข้อมูลกับ Computer และสั่งการไปยัง IC ต่างๆบนวงจร 3. การสร้าง Windows User Interface เขียนโปรแกรมโดยใช้ Visual C++ สร้างเป็น Window Form ให้ผู้ใช้งานสามารถปรับเปลี่ยนและรับรู้ค่าพารามิเตอร์ต่างๆของระบบ และแสดงผลบนหน้าจอมอนิเตอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Title	DEVELOPMENT OF RF MAGNETRON SPUTTERING CONTROL SYSTEM VIA TOUCH SCREEN
Students	Mr. Suphawt Yingjamsiri Student ID 54050603
	Miss. Siraya Somluesaen Student ID 54050601
Degree	Bachelor of Science in Applied Physics
Department	Physics
Academic Year	2014
Advisor	Assoc. Prof. Dr. Warawoot Thowlada
Co-advisor	Dr. S.Tippawan Klayboonme

Abstract

The special project focus on the control system of the RF Magnetron Sputtering System via the touch screen monitor. The operation has 3 section; 1. Design the circuit board for Transmit and receive data between Microcontroller and IC to control or receive the signal from the device, 2. Programming for the Microcontroller to transmit and receive the data between computer and dictate any IC on the circuit Board 3, Creation of the Windows User Interface, use the Visual C++ to build the window form for the user can adjust and acknowledge all of parameter of the system and show these on the monitor. The experimental results showed that the developed system has the good capability to control the RF Magnetron Sputtering via touch screen monitor.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

โครงงานพิเศษเล่มนี้สำเร็จลุล่วงไปได้ด้วยดีนั้น เนื่องจากได้รับการสนับสนุน และ ความอนุเคราะห์เป็นอย่างดีจาก รศ.ดร.วราวุฒิ เถาลัดดา อาจารย์ภาควิชาฟิสิกส์ คณะวิทยาศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง อาจารย์ที่ปรึกษา ซึ่งคอยให้คำแนะนำอย่างใกล้ชิดพร้อมทั้งเสนอแนวทางและวิธีปัญหาต่างๆ

ขอขอบคุณ ดร.ศ.ทิพวรรณ คล้ายบุญมี อาจารย์ภาควิชาฟิสิกส์ คณะวิทยาศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง อ.ที่ปรึกษาร่วม ที่มอบความรู้เกี่ยวกับระบบบอาร์เอฟแมกนีตรอนสปัตเตอริง การทำงาน และการแก้ปัญหา

สุดท้ายนี้ ขอขอบคุณคณะอาจารย์ที่เคารพทุกท่าน ที่ถ่ายทอดความรู้ความสามารถต่างๆ

นายศุภวัฒน์ ยิ่งแจ่มศิริ

นางสาวศิรญา สมลือแสน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

	หน้า
บทคัดย่อภาษาไทย	ก
บทคัดย่อภาษาอังกฤษ	ข
กิตติกรรมประกาศ	ค
สารบัญ	ง
สารบัญรูปภาพ	ฉ
บทที่1 บทนำ	1
1.1 ความเป็นมาและความสำคัญ	1
1.2 วัตถุประสงค์ของงานวิจัย	1
1.3 ขอบเขตของงานวิจัย	1
1.4 ขั้นตอนดำเนินงาน	2
1.5 ประโยชน์ที่คาดว่าจะได้รับ	2
บทที่ 2 ทฤษฎีที่เกี่ยวข้อง	3
2.1 ไมโครคอนโทรเลอร์	3
2.2 Arduino	5
2.3 รูปแบบของการส่งผ่านข้อมูล	8
2.4 Serialport	14
2.5 SPI Serial Peripheral Interface	15
บทที่ 3 ขั้นตอนการดำเนินงาน	18
3.1 อุปกรณ์ที่เราจะทำการควบคุม รับค่าและแสดงผล	18
3.1.1 Massflow Controller Argon and Nitrogen	18
3.1.2 Rotary Pump and Heater(Diffusion Pump)	20
3.1.3 RF Generator	21
3.1.4 Water cooler	23
3.2 โปรแกรม Microcontroller	24
3.3 โปรแกรม Visual c++	26

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ(ต่อ)

	หน้า
บทที่ 4 ผลการทดลองและการอภิปรายผล	30
บทที่ 5 สรุปผลและข้อเสนอแนะ	32
5.1 สรุปผลการวิจัย	32
5.2 ข้อเสนอแนะ	32
บรรณานุกรม	33
ภาคผนวก	34
ภาคผนวก ก	35
ภาคผนวก ข	46
ภาคผนวก ค	68

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูปภาพ

รูปที่	หน้า
2.1 องค์ประกอบของ Microcontroller	4
2.2 Arduino UNO Board	5
2.3 การเชื่อมต่อสื่อสารระห่างคอมพิวเตอร์กับ Microcontroller	6
2.4 การเลือกชนิดของ Board ใน Arduino IDE	6
2.5 เลือกหมายเลข Comport ของบอร์ด	6
2.6 การ Compiled และ การ Upload	7
2.7 Layout & Pin out Arduino Board (Model: Arduino UNO R3)	7
2.8 การส่งข้อมูลแบบขนาน	9
2.9 การส่งข้อมูลแบบอนุกรม	9
2.10 การสื่อสารแบบอะซิงโครนัสที่ไม่ได้ใช้พาริตี้บิต	9
2.11 การสื่อสารแบบอะซิงโครนัสที่ใช้พาริตี้บิต	11
2.12 การสื่อสารแบบซิงโครนัส	11
2.13 ตัวอย่างการส่งข้อมูล 8 Bit	11
2.15 แสดงการตัดแถวของบิตออกเป็นกลุ่มๆ ละ 8 บิต	12
2.16 การส่งผ่านข้อมูลแบบซิงโครนัส	12
2.17 การส่งผ่านข้อมูลแบบอะซิงโครนัส	12
2.18 ทิศทางการส่งผ่านข้อมูล	14
2.19 การติดต่อสื่อสารด้วย SPI: Serial Peripheral Interface	15
2.20 การติดต่อสื่อสารด้วย SPI แบบมีหลาย Slave	17
3.1 Mass Flow Controller Interface Port	18
3.2 MCP4822	19
3.3 วงจรขยายแบบไม่กลับเพส	20
3.4 MCP3208	20
3.5 Relay coil 24V (ก) และ Relay coil 5V(ข)	21
3.6 LM324(ก) 2n2222(ข)	21
3.7 Shift-Register 74589 logic diagram	23
3.8 YF121 Water Flow Sensor	24
3.9 การเขียนโปรแกรมบน Arduino IDE	25
3.10 Window Form บน Visual C++	26
3.11 (ก) การเลือก COMPOT	27

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูปภาพ(ต่อ)

รูปที่	หน้า
3.11 (ข) การเลือก BuadRate	27
3.11 (ค) รูปเมื่อมีการเชื่อต่อกับ Microcontroller	27
3.12 Gas Flow Panel	27
3.13 Pump and Cooler Panel	28
3.14 (ก) ถ้าสถานะเมื่อ RF พร้อมใช้งาน	28
3.14 (ข) การตั้งค่า Setpoint แต่ยังไม่สั่งให้ปรับ Power Output	28
3.15 (ก) ถ้าสถานะเมื่อ RF ปรับ Power Output	29
3.15 (ข) สถานะเมื่อ Overheat	29
4.1 ภายในระบบ ขณะเคลือบ Aluminium	30
4.2 ภายในระบบ ขณะเคลือบ Aluminium Nitride	30
4.3 ฟิล์ม Aliminium ที่ทดลองเคลือบใน Condition RF power ที่ต่างกัน	31
4.4 ฟิล์ม Aliminium Nitrie ที่ทดลองเคลือบใน Condition RF power ที่ต่างกัน	31

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญ

เนื่องจากระบบ Magnetron Sputtering เป็นระบบที่มีส่วนประกอบในการทำงานหลายส่วน และชิ้นงานที่ทำผ่านระบบนี้จะออกมาถูกต้อง มีคุณภาพดีหรือไม่ ก็ขึ้นอยู่กับการเซ็ตระบบของผู้ใช้งานซึ่งต้องควบคุมให้มีค่า parameter ของอุปกรณ์ต่างๆตรงตามที่คำนวณไว้ ซึ่งเป็นเรื่องยุ่งยากที่ผู้ใช้งานจะต้องคอยตรวจสอบอุปกรณ์ทุกส่วนว่ายังทำงานตรงตามที่ตั้งค่าหรือไม่ หรือมีอุปกรณ์ชิ้นใดหยุดทำงานไป

โครงงานนี้เป็นการพัฒนาการควบคุมระบบ Magnetron Sputtering โดยจะให้แสดงค่า parameter ทั้งหมดของระบบที่ผู้ใช้จำเป็นต้องตรวจเช็คระหว่างการใช้งาน อยู่บนหน้าจอสัมผัส ผู้ใช้สามารถมองเห็น กำหนด และสามารถเปลี่ยนแปลงค่า parameter ทุกส่วนของระบบได้โดยผ่านหน้าจอสัมผัสนี้

1.2 วัตถุประสงค์ของงานวิจัย

1. เพื่อสร้างชุดควบคุมระบบ RF Magnetron Sputtering
2. เพื่อพัฒนาโปรแกรมให้ผู้ใช้งาน ใช้งานระบบได้ง่ายขึ้นกว่าเดิม
3. เพื่อนำไมโครคอนโทรเลอร์มาใช้ให้เกิดประโยชน์

1.3 ขอบเขตของงานวิจัย

1. ศึกษาการเขียนโปรแกรมโดยใช้ Arduino IDE
2. ศึกษาการเขียนโปรแกรมโดยใช้ Visual C++
3. ศึกษาวิธีการควบคุมและหลักการทำงานของอุปกรณ์ที่ใช้ในระบบ RF Magnetron Sputtering

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.4 ขั้นตอนดำเนินงาน

สิงหาคม 2557	ศึกษาส่วนประกอบต่างๆของระบบ Magnetron Sputtering วางแผนงานที่ต้องทำ สิ่งที่ต้องใช้
กันยายน – ธันวาคม 2557	ซื้ออุปกรณ์ที่จำเป็น ศึกษา-เขียนโปรแกรมและสร้างชุดควบคุม
มกราคม – มีนาคม 2557	นำโปรแกรม และชุดควบคุมที่สร้างไปลองเชื่อมต่อเข้ากับระบบจริง ตรวจพบปัญหาและแก้ไขงานให้ถูกต้อง ใช้งานได้จริง
เมษายน – พฤษภาคม 2557	พัฒนาโปรแกรมให้ใช้งานได้ง่ายและตรงความต้องการมากขึ้น

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1. สามารถนำความรู้ในการเขียนโปรแกรมควบคุม Microcontroller และสร้าง Windows Forms ในคอมพิวเตอร์ ไปสร้างประโยชน์และใช้งานจริงได้
2. เรียนรู้ส่วนประกอบและลักษณะการทำงานของอุปกรณ์ต่างๆของระบบ Magnetron Sputtering

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

2.1 ไมโครคอนโทรเลอร์

ไมโครคอนโทรลเลอร์ (อังกฤษ: Microcontroller) คือ อุปกรณ์ควบคุมขนาดเล็ก ซึ่งบรรจุความสามารถที่คล้ายคลึงกับระบบคอมพิวเตอร์โดยในไมโครคอนโทรลเลอร์ได้รวมเอาซีพียู ,หน่วยความจำ และพอร์ต ซึ่งเป็นส่วนประกอบหลักสำคัญของระบบคอมพิวเตอร์เข้าไว้ด้วยกัน โดยทำการบรรจุเข้าไว้ในตัวถังเดียวกัน

โครงสร้างโดยทั่วไป ของไมโครคอนโทรลเลอร์นั้น สามารถแบ่งออกมาได้เป็น 5 ส่วนใหญ่ๆ ดังต่อไปนี้

1. หน่วยประมวลผลกลางหรือซีพียู (CPU : Central Processing Unit) ทำหน้าที่เป็นศูนย์กลางควบคุมการทำงานของระบบคอมพิวเตอร์ทั้งหมด โดยนำข้อมูลจากอุปกรณ์รับข้อมูลมาทำงานประมวลผลข้อมูลตามคำสั่งของโปรแกรม และส่งผลลัพธ์ออกไปหน่วยแสดงผล

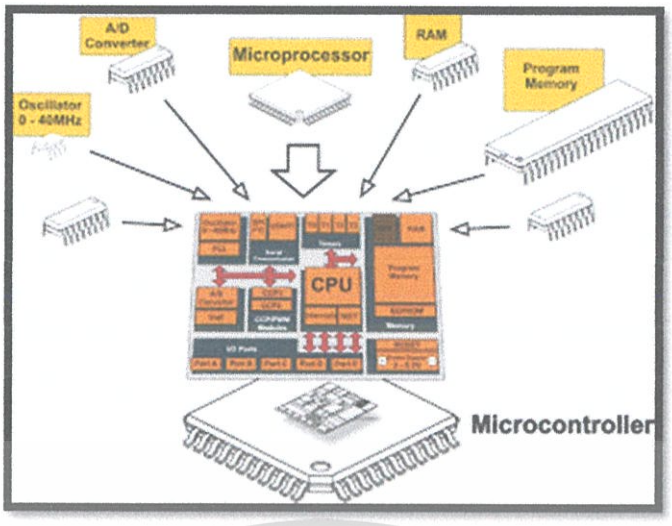
2. หน่วยความจำ (Memory) สามารถแบ่งออกเป็น ส่วน คือ หน่วยความจำที่มีไว้สำหรับเก็บโปรแกรมหลัก (Program Memory) เปรียบเสมือนฮาร์ดดิสก์ของ เครื่องคอมพิวเตอร์ตั้งโต๊ะ คือ ข้อมูลใดๆ ที่ถูกเก็บไว้ในนี้จะไม่สูญหายไปแม้ไม่มีไฟเลี้ยง อีกส่วนหนึ่งคือหน่วยความจำข้อมูล (Data Memory) ใช้เป็นเหมือนกกระดาษทดในการคำนวณของซีพียู และเป็นที่พักข้อมูลชั่วคราวขณะทำงาน แต่หากไม่มีไฟเลี้ยง ข้อมูลก็จะหายไปคล้ายกับหน่วยความแรม (RAM) ในเครื่องคอมพิวเตอร์ทั่วๆ ไป แต่สำหรับไมโครคอนโทรลเลอร์สมัยใหม่ หน่วยความจำข้อมูลจะมีทั้งที่เป็นหน่วยความจำแรม ซึ่งข้อมูลจะหายไปเมื่อไม่มีไฟเลี้ยง และเป็นอีอีพรอม (EEPROM : Erasable Electrically Read-Only Memory) ซึ่งสามารถเก็บข้อมูลได้แม้ไม่มีไฟเลี้ยง

3. ส่วนติดต่อกับอุปกรณ์ภายนอก หรือพอร์ต (Port) มี 2 ลักษณะคือ พอร์ตอินพุต (Input Port) และพอร์ตส่งสัญญาณหรือพอร์ตเอาต์พุต (Output Port) ส่วนนี้จะใช้ในการเชื่อมต่อกับอุปกรณ์ภายนอก ถือว่าเป็นส่วนที่สำคัญมาก ใช้ร่วมกันระหว่างพอร์ตอินพุต เพื่อรับสัญญาณ อาจจะด้วยการกดสวิตช์ เพื่อนำไปประมวลผลและส่งไปพอร์ตเอาต์พุต เพื่อแสดงผลเช่น การติดสว่างของหลอดไฟเป็นต้น

4. ช่องทางเดินของสัญญาณ หรือบัส (BUS) คือเส้นทางการแลกเปลี่ยนสัญญาณข้อมูลระหว่าง ซีพียู หน่วยความจำและพอร์ต เป็นลักษณะของสายสัญญาณ จำนวนมากอยู่ภายในตัวไมโครคอนโทรลเลอร์ โดยแบ่งเป็นบัสข้อมูล (Data Bus) , บัสแอดเดรส (Address Bus) และบัสควบคุม (Control Bus)

5. วงจรกำเนิดสัญญาณนาฬิกา นับ เป็นส่วนประกอบที่สำคัญมากอีกส่วนหนึ่ง เนื่องจากการทำงานที่เกิดขึ้นในตัวไมโครคอนโทรลเลอร์ จะขึ้นอยู่กับการกำหนดจังหวะ หากสัญญาณนาฬิกามีความถี่สูงจังหวะการทำงานก็จะสามารถทำได้ถี่ขึ้นส่งผลให้ไมโครคอนโทรลเลอร์ตัวนั้น มีความเร็วในการประมวลผลสูงตามไปด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.1 องค์ประกอบของ Microcontroller

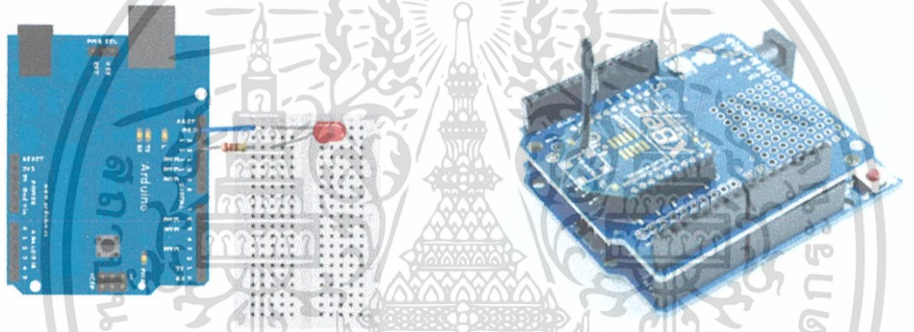
นอกจากนี้ยังมีส่วนพิเศษอื่นๆ จะขึ้นอยู่กับกระบวนการผลิตของแต่ละบริษัทที่จะผลิตขึ้นมาใส่คุณสมบัติพิเศษลงไปเช่น

- **ADC** (Analog to Digital) ส่วนภาครับสัญญาณอนาล็อกแปลงไปเป็นสัญญาณดิจิตอล
- **DAC** (Digital to Analog) ส่วนภาคส่งสัญญาณดิจิตอลแปลงไปเป็นสัญญาณอนาล็อก
- **I2C** (Inter Integrate Circuit Bus) เป็นการสื่อสารอนุกรม แบบซิงโครนัส (Synchronous) เพื่อใช้ติดต่อสื่อสาร ระหว่าง ไมโครคอนโทรลเลอร์ (MCU) กับอุปกรณ์ภายนอก ซึ่งถูกพัฒนาขึ้นโดยบริษัท Philips Semiconductors โดยใช้สายสัญญาณเพียง 2 เส้นเท่านั้น คือ serial data (SDA) และสาย serial clock (SCL) ซึ่งสามารถ เชื่อมต่ออุปกรณ์ จำนวนหลายๆ ตัว เข้าด้วยกันได้ ทำให้ MCU ใช้พอร์ตเพียง 2 พอร์ตเท่านั้น
- **SPI** (Serial Peripheral Interface) เป็นการเชื่อมต่อกับอุปกรณ์เพื่อรับส่งข้อมูลแบบซิงโครนัส (Synchronize) มีสัญญาณนาฬิกาเข้ามาเกี่ยวข้องระหว่างไมโครคอนโทรลเลอร์ (Microcontroller) หรือจะเป็นอุปกรณ์ภายนอกที่มีการรับส่งข้อมูลแบบ SPI อุปกรณ์ที่ทำหน้าที่เป็นมาสเตอร์ (Master) โดยปกติแล้วจะเป็นไมโครคอนโทรลเลอร์ หรืออาจกล่าวได้ว่าอุปกรณ์ Master จะต้องควบคุมอุปกรณ์ Slave ได้ โดยปกติตัว Slave มักจะเป็นไอซี (IC) หน้าที่พิเศษต่างๆ เช่น ไอซีอุณหภูมิ, ไอซีฐานเวลานาฬิกาจริง (Real-Time Clock) หรืออาจเป็นไมโครคอนโทรลเลอร์ที่ทำหน้าที่ในโหมด Slave ก็ได้เช่นกัน
- **PWM** (Pulse Width Modulation) การสร้างสัญญาณพัลส์แบบสแควร์เวฟ ที่สามารถปรับเปลี่ยนความถี่และ Duty Cycle ได้เพื่อนำไปควบอุปกรณ์ต่างๆเช่น มอเตอร์
- **UART** (Universal Asynchronous Receiver Transmitter) ทำหน้าที่รับส่งข้อมูลแบบอะซิงโครนัสสำหรับมาตรฐานการรับส่งข้อมูลแบบ RS-232

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2 Arduino

Arduino **อ่านว่า (อา-ดู-อิ-โน่ หรือ อาดุยโน่) เป็นบอร์ดไมโครคอนโทรเลอร์ตระกูล AVR ที่มีการพัฒนาแบบ Open Source คือมีการเปิดเผยข้อมูลทั้งด้าน Hardware และ Software ตัว บอร์ด Arduino ถูกออก แบบมาให้ใช้งานได้ง่าย ดังนั้นจึงเหมาะสำหรับผู้เริ่มต้นศึกษา ทั้งนี้ ผู้ใช้งานยังสามารถดัดแปลง เพิ่มเติม พัฒนาต่อยอดทั้งตัวบอร์ด หรือโปรแกรมต่อได้อีกด้วย**ความง่ายของบอร์ด Arduino ในการต่ออุปกรณ์เสริมต่างๆ คือผู้ใช้งานสามารถต่อวงจรอิเล็กทรอนิคส์จากภายนอกแล้วเชื่อมต่อเข้ามาที่ขา I/O ของบอร์ด (ดูตัวอย่างรูปที่ 1) หรือเพื่อความสะดวกสามารถเลือกต่อกับบอร์ดเสริม (Arduino Shield) ประเภทต่างๆ (ดูตัวอย่างรูปที่ 2) เช่น Arduino XBee Shield, Arduino Music Shield, Arduino Relay Shield, Arduino Wireless Shield, Arduino GPRS Shield เป็นต้น มาเสียบกับบอร์ดบนบอร์ด Arduino แล้วเขียนโปรแกรมพัฒนาต่อได้เลย



รูปที่1 บอร์ด Arduino ต่อกับ LED รูปที่2 บอร์ด Arduino ต่อกับบอร์ด XBee Shield

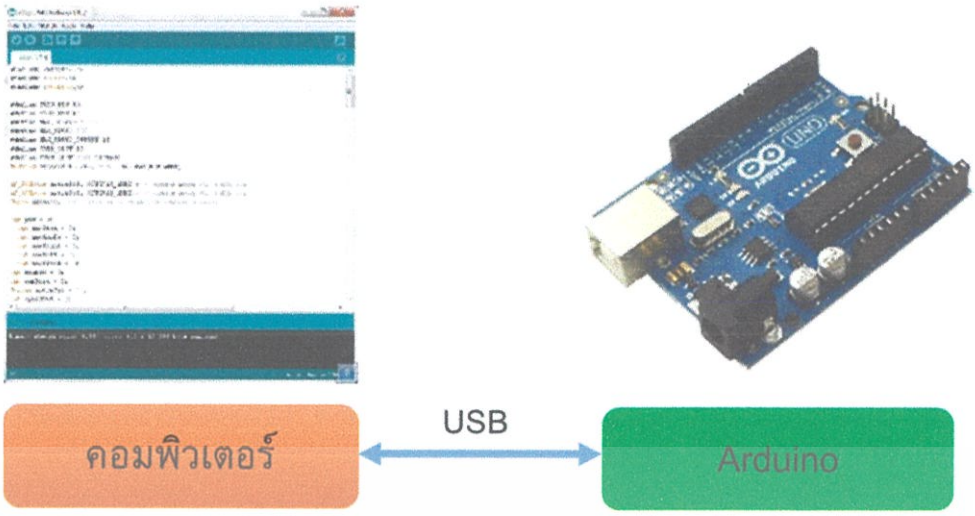
รูปที่ 2.2 Arduino UNO Board

จุดเด่นที่ทำให้บอร์ด Arduino เป็นที่นิยม

- ง่ายต่อการพัฒนา มีรูปแบบคำสั่งพื้นฐาน ไม่ซับซ้อนเหมาะสำหรับผู้เริ่มต้น
- มี Arduino Community กลุ่มคนที่ร่วมกันพัฒนาที่แข็งแรง
- Open Hardware ทำให้ผู้ใช้สามารถนำบอร์ดไปต่อยอดใช้งานได้หลายด้าน
- ราคาไม่แพง
- Cross Platform สามารถพัฒนาโปรแกรมบน OS ใดก็ได้

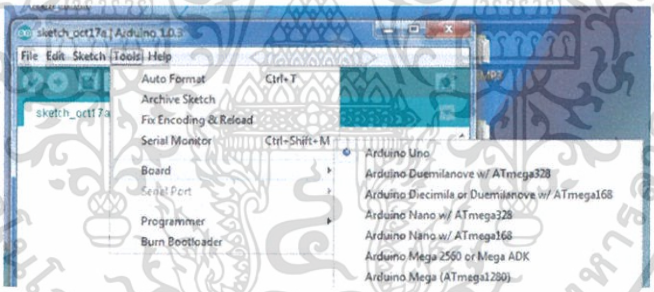
รูปแบบการเขียนโปรแกรมบน Arduino

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

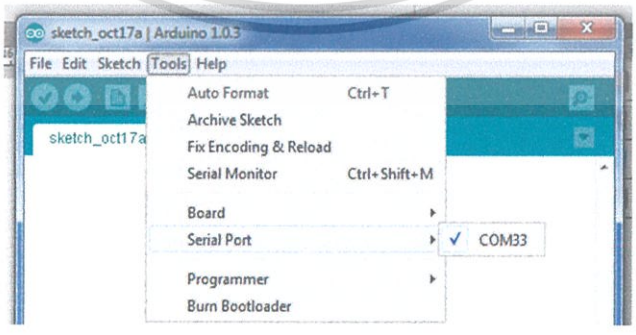


รูปที่ 2.3 การเชื่อมต่อสื่อสารระห่างคอมพิวเตอร์กับ Microcontroller

1. เขียนโปรแกรมบนคอมพิวเตอร์ ผ่านทางโปรแกรม ArduinoIDE ซึ่งสามารถดาวน์โหลดได้จาก Arduino.cc/en/main/software
2. หลังจากที่เขียนโค้ดโปรแกรมเรียบร้อยแล้ว ให้ผู้ใช้งานเลือกรุ่นบอร์ด Arduino ที่ใช้และหมายเลข Com port



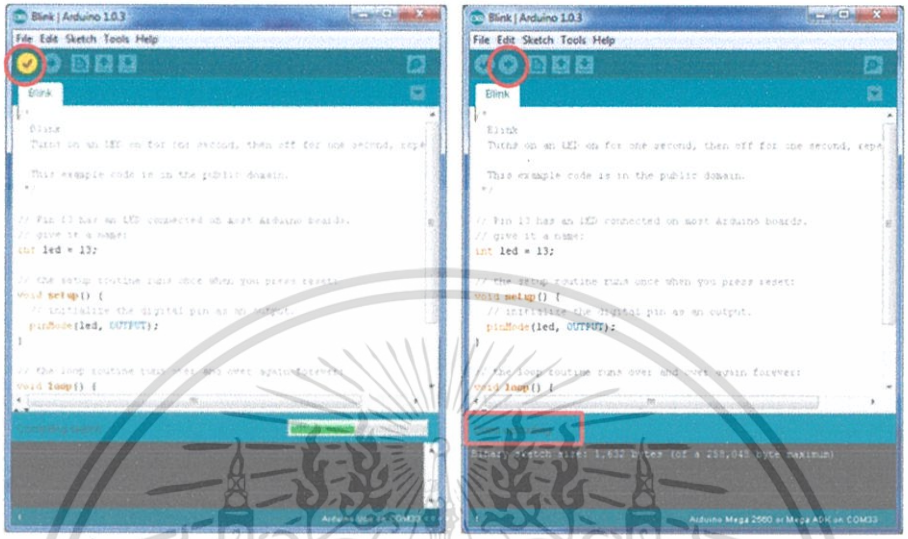
รูปที่ 2.4 การเลือกชนิดของ Board ใน Arduino IDE



รูปที่ 2.5 เลือกหมายเลข Comport ของบอร์ด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

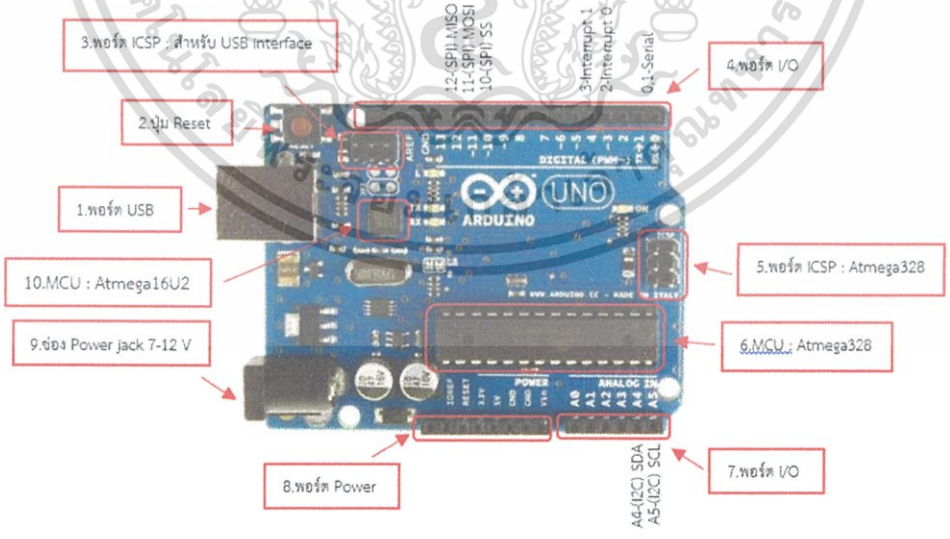
3. กดปุ่ม Verify เพื่อตรวจสอบความถูกต้องและ Compile โค้ดโปรแกรม จากนั้นกดปุ่ม Upload โค้ด โปรแกรมไปยังบอร์ด Arduino ผ่านทางสาย USB เมื่ออัปโหลดเรียบร้อยแล้ว จะแสดงข้อความแถบข้างล่าง “Done uploading” และบอร์ดจะเริ่มทำงานตามที่เขียนโปรแกรมไว้ได้ทันที



รูปที่5 กดปุ่ม Verify เพื่อตรวจสอบความถูกต้องและ Compile โค้ดโปรแกรม

รูปที่6 Upload โค้ดโปรแกรม

รูปที่ 2.6 การ Compiled และ การ Upload



รูปที่ 2.7 Layout & Pin out Arduino Board (Model: Arduino UNO R3)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.USBPort: ใช้สำหรับต่อกับ Computer เพื่ออับโหลดโปรแกรมเข้า MCU และจ่ายไฟให้กับบอร์ด
2.Reset Button: เป็นปุ่ม Reset ใช้กดเมื่อต้องการให้ MCU เริ่มการทำงานใหม่
3.ICSP Port ของ Atmega16U2 เป็นพอร์ตที่ใช้โปรแกรม Visual Com port บน Atmega16U2
4. I/OPort: Digital I/O ตั้งแต่ขา D0 ถึง D13 นอกจากนี้ บาง Pin จะทำหน้าที่อื่นๆ เพิ่มเติมด้วย เช่น Pin0,1 เป็นขา Tx,Rx Serial, Pin3,5,6,9,10 และ 11 เป็นขา PWM
5.ICSP Port: Atmega328 เป็นพอร์ตที่ใช้โปรแกรม Bootloader
6. MCU: Atmega328 เป็น MCU ที่ใช้บนบอร์ด Arduino
7. I/OPort: นอกจากจะเป็น Digital I/O แล้ว ยังเปลี่ยนเป็น ช่องรับสัญญาณอนาล็อก ตั้งแต่ขา A0-A5
8.Power Port: ไฟเลี้ยงของบอร์ดเมื่อต้องการจ่ายไฟให้กับวงจรภายนอก ประกอบด้วยขาไฟเลี้ยง +3.3 V, +5V, GND, V_{in}
9. Power Jack: รับไฟจาก Adapter โดยที่แรงดันอยู่ระหว่าง 7-12 V
10. MCU ของ Atmega16U2 เป็น MCU ที่ทำหน้าที่เป็น USB to Serial โดย Atmega328 จะติดต่อกับ Computer ผ่าน Atmega16U2

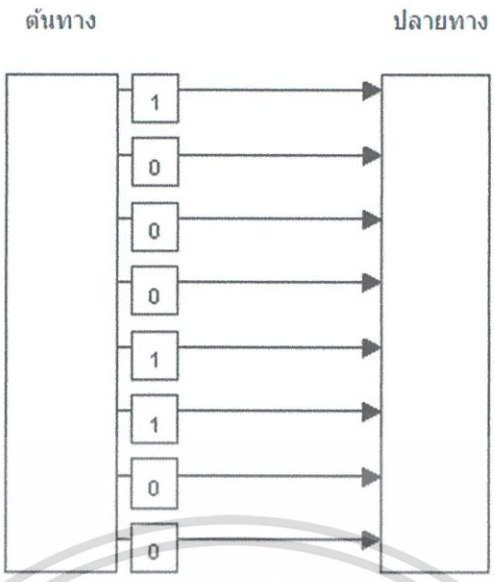
2.3 รูปแบบของการส่งผ่านข้อมูล

รูปแบบของสายส่งสัญญาณสื่อสารอาจประกอบด้วยสายส่งตั้งแต่หนึ่งสายขึ้นไป ซึ่งทำให้เกิดช่องทางการส่ง ข้อมูลได้มากกว่าหนึ่งช่องทาง รูปแบบของการส่งผ่านข้อมูลสามารถแบ่งได้เป็น 2 รูปแบบ คือ

1. การส่งผ่านข้อมูลแบบขนาน (Parallel Transmission)

ในรูปแบบการส่งผ่านข้อมูลในลักษณะนี้ทุกบิตที่แทนข้อมูลหนึ่งตัวอักษร จะถูกส่งผ่านไปตามสายส่งหลายๆ สายขนานกันไป ดังนั้นทุกบิตจะเดินทางถึงผู้รับพร้อมๆ กัน และจำนวนสายส่งเพื่อให้เกิดช่องทางการส่งจะต้องมีจำนวนเท่ากับจำนวนบิตที่เข้ารหัสแทนตัวอักษรแต่ละระบบ เช่น การส่งผ่านข้อมูลที่เข้ารหัสแบบ ASCII ก็จำเป็นต้องใช้ช่องทางการส่ง 8 ช่องทาง จึงจะทำให้ทุกบิตวิ่งผ่านสายส่งขนานกันไปได้ดังรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

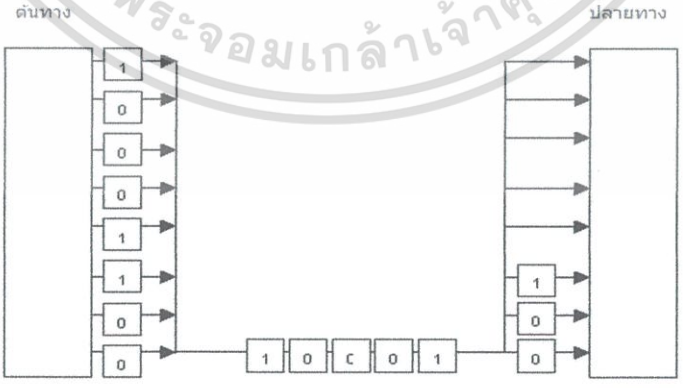


รูปที่ 2.8 การส่งข้อมูลแบบขนาน

จากรูปแสดงให้เห็นว่าทุกบิตที่เข้ารหัสตัวอักษรตัวหนึ่งๆ จะเดินทางขนานกันไป โดยเริ่มจากต้นทางผ่านสายส่งสัญญาณที่มีอยู่ด้วยกัน 8 เส้น ไปยังปลายทาง ดังนั้น ปลายทางจะได้รับทุกบิตของตัวอักษรพร้อมๆ กัน การส่งผ่านข้อมูลแบบขนานนี้ส่วนมากจะใช้ในการส่งผ่านข้อมูลในระยะใกล้ อันได้แก่การเคลื่อนย้ายข้อมูลระหว่างเครื่องคอมพิวเตอร์กับอุปกรณ์รอบข้างของมันเช่น ระหว่างเครื่องคอมพิวเตอร์กับเครื่องพิมพ์ เพราะสามารถทำได้ด้วยความเร็วสูง และถ้านำไปใช้ในระยะไกลๆ จะต้องเสียค่าใช้จ่ายสูง เนื่องจากต้องใช้สายส่งเท่ากับจำนวนบิตที่ใช้เข้ารหัสแทนข้อมูลตัวอักษร

2. การส่งผ่านข้อมูลแบบอนุกรม (Serial Transmission)

รูปแบบการส่งผ่านข้อมูลในลักษณะนี้ทุกบิตที่เข้ารหัสแทนข้อมูลหนึ่งตัวอักษรจะถูกส่งผ่านไปตามสายส่งเรียงลำดับไปทีละบิตในสายส่งเพียงเส้นเดียว ดังรูป



รูปที่ 2.9 การส่งข้อมูลแบบอนุกรม

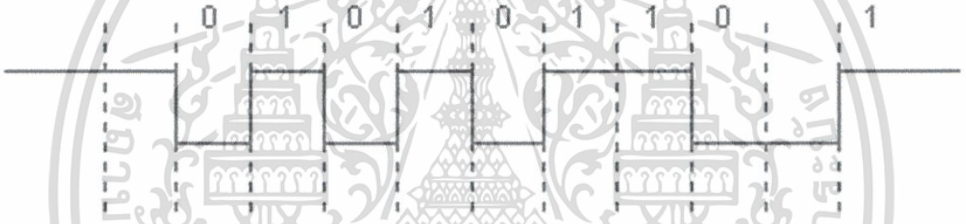
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากรูปตัวอักษรจะประกอบด้วย 8 บิต เรียงเป็นลำดับ ข้อมูลจะถูกส่งออกมาทีละบิตระหว่างต้นทางและปลายทาง และปลายทางจะรวมบิตเหล่านี้ทีละบิตจนครบ 8 บิตเป็น 1 ตัวอักษร จะเห็นว่าการส่งข้อมูลแบบนี้จะช้ากว่าแบบขนาน แต่ค่าใช้จ่ายจะถูกกว่าแบบขนาน ซึ่งเหมาะสำหรับการขนส่งระยะไกลๆ

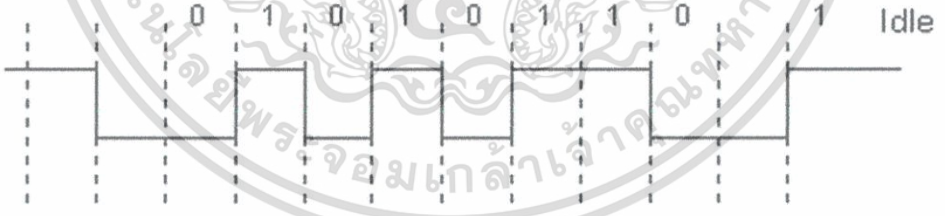
โดยทั่วไปแล้วการส่งข้อมูลนั้นจะประกอบไปด้วยกลุ่มของตัวอักษร ดังนั้นในการส่งข้อมูลแบบอนุกรมนี้จึงเกิดปัญหาขึ้นว่า แล้วต้นทางและปลายทางจะทราบได้อย่างไรว่าจะแบ่งแต่ละตัวอักษรตรงบิตใด จึงเกิดวิธีการสื่อสารข้อมูลขึ้น 2 แบบคือ การสื่อสารแบบอะซิงโครนัส (Asynchronous Transmission) และการสื่อสารแบบซิงโครนัส (Synchronous Transmission)

1. การสื่อสารแบบอะซิงโครนัส (*Asynchronous Transmission*)

การสื่อสารแบบอะซิงโครนัส หรือเรียกอีกอย่างหนึ่งว่าเป็น การสื่อสารแบบระบุจุดเริ่มต้นและจุดสิ้นสุด (Start-Stop Transmission) ลักษณะของสัญญาณที่ใช้ในการติดต่อสื่อสารกันจะประกอบไปด้วย บิตเริ่มต้น (start bit) บิตของข้อมูลที่สื่อสาร (transmission data) จำนวน 8 บิต บิตตรวจข้อผิดพลาด (parity bit) และบิตสิ้นสุด (stop bit) สำหรับบิตตรวจสอบข้อผิดพลาดจะใช้หรือไม่ใช้ก็ได้ ดังนั้นสัญญาณจึงต้องประกอบด้วยส่วนประกอบอย่างน้อย 3 ส่วนดังรูป



รูปที่ 2.10 การสื่อสารแบบอะซิงโครนัสที่ไม่ได้ใช้พาริตี้บิต



รูปที่ 2.11 การสื่อสารแบบอะซิงโครนัสที่ใช้พาริตี้บิต

จากรูปจะเห็นว่าขณะที่ไม่มีข้อมูลส่งออกมาสถานะของการส่งจะเป็นแบบว่าง (Idle) ซึ่งจะมีระดับของสัญญาณเป็น 1 ตลอดเวลา เพื่อความแน่ใจว่าปลายทาง หรือฝ่ายรับยังคงติดต่อกับต้นทางหรือฝ่ายส่งอยู่ เมื่อเริ่มจะส่งข้อมูลสัญญาณของอะซิงโครนัสจะเป็น 0 หนึ่งช่วงสัญญาณนาฬิกาซึ่งบิตนี้เราเรียกว่าบิตเริ่มต้น ตามหลังของบิตเริ่มต้นจะเป็นบิตข้อมูลสำหรับ 1 ตัวอักษร ตามหลังบิตข้อมูลก็จะเป็นบิตตรวจข้อผิดพลาดแล้วจะตามด้วยบิตสิ้นสุด ถ้าไม่ใช่บิตตรวจข้อผิดพลาด ตามหลังบิตข้อมูลก็จะเป็นบิตสิ้นสุดเลย หลังจากนั้นถ้าไม่มีข้อมูลส่งออกมาสัญญาณจะกลับไปอยู่ที่สถานะแบบว่างอีก เพื่อรอการส่งข้อมูลต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

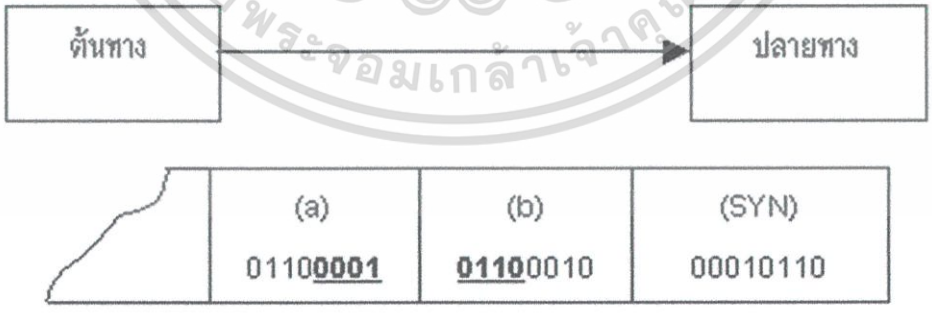
จะเห็นว่าการสื่อสารแบบอะซิงโครนัสนี้ มีลักษณะเป็นไปทีละตัวอักษร และสัญญาณที่ส่งออกมา มีบางส่วนเป็นบิตเริ่มต้น บิตสิ้นสุด และบิตตรวจข้อผิดพลาด ทำให้ความเร็วในการส่งข้อมูลต่อวินาทีน้อยลงไป เนื่องจากต้องสูญเสียช่องทางการสื่อสารให้กับบิตเริ่มต้น บิตสิ้นสุด และบิตตรวจสอบข้อผิดพลาด (ถ้ามีใช้) ตลอดเวลา การสื่อสารแบบอะซิงโครนัสนี้มักใช้ในการติดต่อระหว่างคอมพิวเตอร์กับอุปกรณ์รอบข้าง

2. การสื่อสารแบบซิงโครนัส (*Synchronous Transmission*)

การสื่อสารแบบซิงโครนัส จะทำการจัดกลุ่มของข้อมูลเป็นกลุ่มๆ และทำการส่งข้อมูลทั้งกลุ่มไปพร้อมกันในทีเดียว เราเรียกกลุ่มข้อมูลนี้ว่า บล็อกของข้อมูล (Block of Data) ซึ่งตัวอักษรตัวแรกและตัวถัดไปที่อยู่ในบล็อกเดียวกันจะไม่มีอะไรมาคั่นเหมือนอย่างแบบอะซิงโครนัส ที่ต้องใช้บิตเริ่มต้นและบิตสิ้นสุดคั่นทุกๆ ตัวอักษร แต่จะมีข้อมูลเริ่มต้นซึ่งเป็นลักษณะของบิตพิเศษที่ส่งมาเพื่อให้รู้ว่านั้นคือจุดเริ่มต้นของกลุ่มตัวอักษรที่กำลังส่งเรียงกันเข้ามา เช่น อักขระซิง (SYN character) โดยที่อักขระซิงมีรูปแบบบิต คือ 00010110 ตัวอย่างของการส่งแสดงได้ดังรูป

รูปที่ 2.12 การสื่อสารแบบซิงโครนัส

จากรูปเมื่อลายทางตรวจพบอักขระซิง หรือ 00010110 แล้วจะทราบได้ทันทีว่าบิตที่ตามมาคือบิตตัวอักษรแต่ละตัว แต่การใช้อักขระซิงเพียงตัวเดียวอาจเกิดข้อผิดพลาดได้ เช่น ถ้าเราส่งตัวอักษร b และตัวอักษร a ติดต่อกันไป ซึ่งตัวอักษร b มีรูปแบบบิตคือ 01100010 และตัวอักษร a มีรูปแบบบิตคือ 01100001 การส่งจะแสดงดังรูป

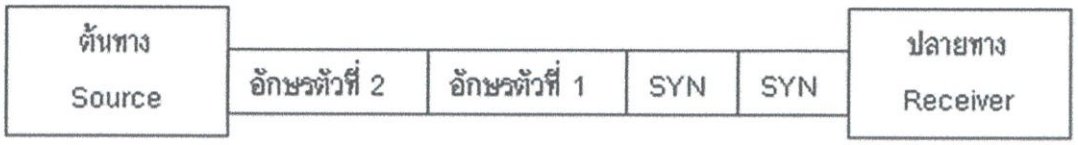


รูปที่ 2.13 ตัวอย่างการส่งข้อมูล 8 Bit

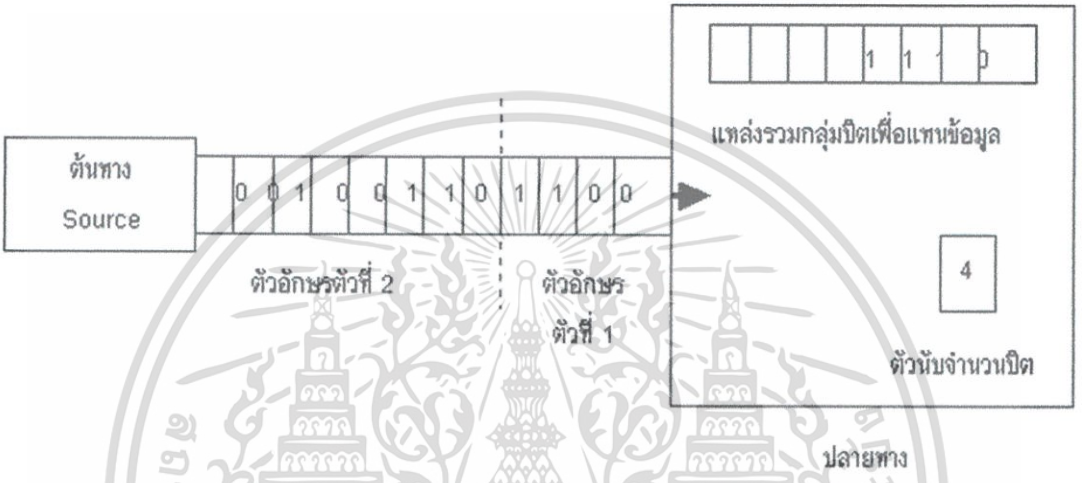
จะเห็นว่าเครื่องปลายทางจะตรวจพบอักขระซิงระหว่างบิตของตัวอักษร **b** และตัวอักษร **a** ทำให้เข้าใจว่าบิตต่อไปจะเป็นบิตของกลุ่มข้อมูลซึ่งจะทำให้การรับข้อมูลนั้นเกิดผิดพลาดขึ้นได้ ดังนั้นจึงแก้ปัญหาด้วยการใช้อักขระซิง 2 ตัวต่อกันและเป็นลักษณะของบิตพิเศษที่บอกให้ทราบว่าเป็น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จุดเริ่มต้นบิตของกลุ่มข้อมูล ตัวอย่างของการใช้อักขระซิง 2 ตัวในการสื่อสารแบบซิงโครนัสและการตัดแถวของบิตข้อมูลออกเป็นกลุ่มทีละ 8 บิตเพื่อแทนข้อมูลแสดงได้ดังรูป



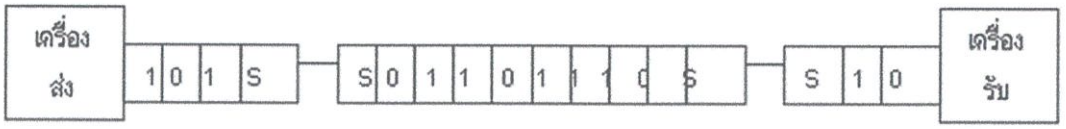
รูปที่ 2.14 ตัวอย่างของการใช้อักขระซิง 2 ตัวในการสื่อสารแบบซิงโครนัส



รูปที่ 2.15 แสดงการตัดแถวของบิตออกเป็นกลุ่มๆ ละ 8 บิต

การสื่อสารแบบซิงโครนัสนี้มักใช้ในการติดต่อระหว่างคอมพิวเตอร์
ประสิทธิภาพของการส่งผ่านข้อมูลแบบอะซิงโครนัส และแบบซิงโครนัส

รูปที่ 2.16 การส่งผ่านข้อมูลแบบซิงโครนัส



รูปที่ 2.17 การส่งผ่านข้อมูลแบบอะซิงโครนัส

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากรูปที่ 2.15 แสดงให้เห็นว่าการส่งผ่านข้อมูลแบบซิงโครนัสนั้นส่วนมากแล้ว ตลอดทางของสายส่งจะใช้ส่งผ่านข้อมูลเต็มตลอดทั้งสาย ส่วนรูปที่ 2.16 แสดงให้เห็นว่าการส่งผ่านข้อมูลแบบอะซิงโครนัสนั้นสายส่งจะขาดความต่อเนื่องของสัญญาณข้อมูลที่ส่งผ่าน หรือถ้ามีสัญญาณข้อมูลที่ส่งผ่านต่อเนื่องกันเต็มตลอดทั้งสายแล้ว ก็จะสูญเสียช่องทางในการส่งไปกับการส่งบิตเริ่มต้น และบิตสิ้นสุดของแต่ละตัวอักษร

ตัวอย่างเช่น กรณีที่ส่งผ่านข้อมูลที่อยู่ในรูปของรหัส **ASCII** ซึ่งตัวอักษรหนึ่งตัวถูกแทนด้วย 8 บิต ถ้ามีการส่งกลุ่มของข้อมูล 240 ตัวอักษร ในกรณีการส่งผ่านข้อมูลแบบซิงโครนัสมีการใช้ตัวอักขระซิง 3 ตัวและการส่งผ่าน ข้อมูลแบบอะซิงโครนัสไม่มีการใช้บิตตรวจข้อผิดพลาด ดังนั้นเราสามารถคำนวณหาอัตราส่วนระหว่างการส่งข้อมูลได้ ดังนี้

บิตทั้งหมดของตัวอักษรที่ส่งจะได้

240ตัวอักษร x 8 บิต/ตัวอักษร = 1920 บิต

แบบซิงโครนัส

-บิตของตัวอักขระซิงที่ใช้จะได้ SYN 3 ตัว เท่ากับ 3 x 8 บิต = 24 บิต

-แบบซิงโคนัส

-บิตของตัวอักขระซิงที่ใช้จะได้ SYN 3 ตัว เท่ากับ 3 x 8 บิต = 24 บิต

-ผลรวมของบิตที่ต้องส่งทั้งหมด = 1920 + 24 = 1944 บิต

-อัตราส่วนระหว่างการส่งข้อมูลที่ต้องส่งจริง กับจำนวนบิตทั้งหมดที่จำเป็นต้องส่งคือ

-1920 หารด้วย 1944 จะได้ประมาณ 99 %

แบบอะซิงโคนัส

-บิตเริ่มต้น และบิตสิ้นสุดที่ใช้จะได้ 2 x 240 = 480 บิต

-ผลรวมของบิตที่ต้องส่งทั้งหมด = 1920 + 480 = 2400 บิต

-อัตราส่วนระหว่างการส่งข้อมูลที่ต้องส่งจริง กับจำนวนบิตทั้งหมดที่จำเป็นต้องส่งคือ1920หารด้วย 2400 จะได้ประมาณ 80 %

การใช้บิตตรวจข้อผิดพลาด

บิตตรวจข้อผิดพลาด หรือพารีตี้บิต จะเป็นบิตที่ใช้เพื่อทำหน้าที่ตรวจสอบความถูกต้องของข้อมูลที่ส่ง ซึ่งมีอยู่ 2 แบบด้วยกันคือ การตรวจสอบจำนวนคี่ (odd parity) และการตรวจสอบจำนวนคู่ (even parity)

การตรวจสอบจำนวนคี่ (Odd parity) หมายถึง บิตตรวจสอบจะต้องนับบิตที่มีค่าของ 1 สำหรับกลุ่มของบิตที่จะส่งและต้องการตรวจสอบอยู่เป็นจำนวนคี่ เช่น ถ้านับบิตที่มีค่าของ 1 ในกลุ่มของบิตที่จะส่ง และต้องการ ตรวจสอบได้เป็นจำนวนคู่ บิตตรวจสอบนี้จะต้องมีค่าเป็น 1 เพื่อที่จะรวมเป็นจำนวนคี่ แต่ถ้าจำนวนนับได้เป็นจำนวนคี่ บิตตรวจสอบก็จะมีค่าเป็น 0

ตัวอย่าง

สมมุติว่าถ้าข้อมูลที่ต้องการส่งมี 7 บิต คือ 0110011 บิตตรวจสอบจำนวนคี่จะต้องมีค่าเป็น 1 เพราะนับบิตที่มีค่าของ 1 ได้เท่ากับ 4 ตัว ซึ่งเป็นเลขคู่ เมื่อรวมกับบิตตรวจสอบจำนวนคี่ที่มีค่าเป็น 1 ก็จะนับได้เป็น 5 ตัว ซึ่งเป็นเลขคี่และการส่งข้อมูลพร้อมบิตตรวจสอบไปจะได้เป็น 10110011

การตรวจสอบจำนวนคู่ (Even parity) หมายถึง บิตตรวจสอบจะต้องนับบิตที่มีค่าของ 1 สำหรับกลุ่มของบิตที่จะส่งและต้องการตรวจสอบอยู่เป็นจำนวนคู่ เช่น ถ้านับบิตที่มีค่าของ 1 ในกลุ่มของบิต

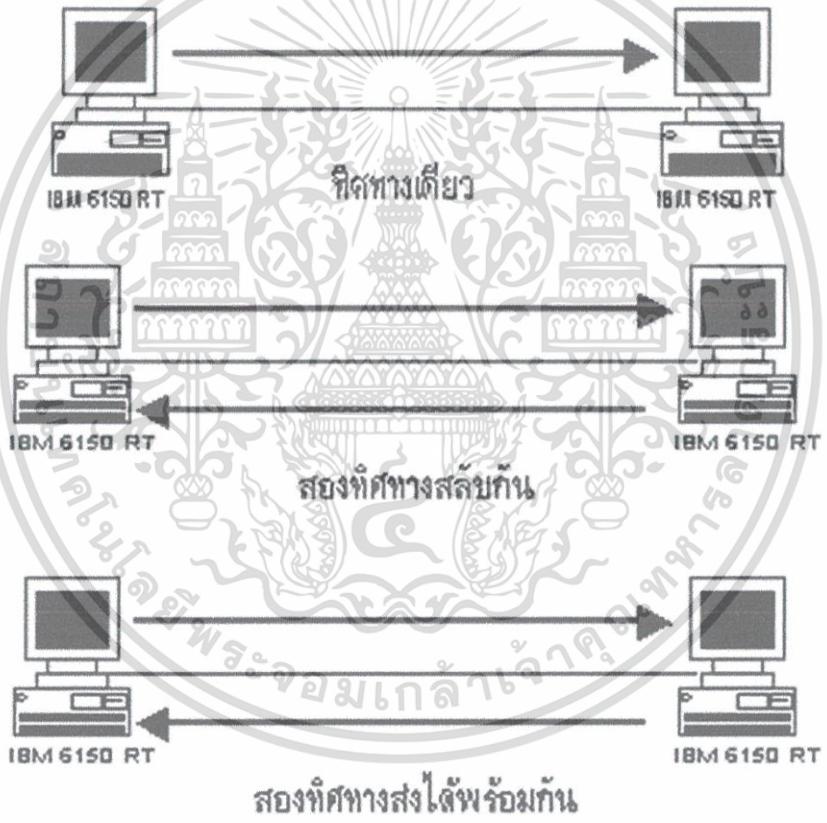
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ที่จะส่งและต้องการ ตรวจสอบได้เป็นจำนวนคู่ บิตตรวจสอบนี้จะต้องมีค่าเป็น 0 เพื่อที่จะรวมเป็นจำนวนคู่ แต่ถ้าจำนวนนับได้เป็นจำนวนคี่ บิตตรวจสอบก็จะมีค่าเป็น 1

ตัวอย่าง

สมมุติว่าถ้าข้อมูลที่ต้องการส่งมี 7 บิต คือ 0110011 บิตตรวจสอบจำนวนคู่จะต้องมีค่าเป็น 0 เพราะนับบิตที่มีค่าของ 1 ได้เท่ากับ 4 ตัว ซึ่งเป็นเลขคู่ การส่งข้อมูลพร้อมบิตตรวจสอบไปจะได้เป็น 00110011

การตรวจสอบความถูกต้องทำได้โดย ระหว่างต้นทางและปลายทางจะต้องตกลงกันว่าจะใช้ตัวตรวจข้อผิดพลาดชนิดใด ถ้าใช้ตัวตรวจข้อผิดพลาดแบบจำนวนคี่แล้วเมื่อปลายทางรับข้อมูลจะตรวจสอบจำนวนบิตที่มีค่าเป็น 1 ว่าเป็นจำนวนคี่หรือไม่ ถ้าไม่เป็นจำนวนคี่แสดงว่าข้อมูลเกิดความผิดพลาดขึ้น ปลายทางจะต้องแจ้งให้ต้นทางทราบ อาจจะให้ต้นทางส่งข้อมูลมาใหม่อีกครั้ง ส่วนการใช้ตัวตรวจข้อผิดพลาดแบบจำนวนคู่ก็จะใช้หลักการคล้ายๆ กัน



รูปที่ 2.18 ทิศทางการส่งผ่านข้อมูล

2.4 Serialport Serial port และ Parallel port คืออะไร พร้อมยกตัวอย่างการนำไปใช้งาน

Serial Port คือ พอร์ตอนุกรม ในการสื่อสารข้อมูลนั้นพอร์ตอนุกรมจะมีความเร็วในการสื่อสารที่ช้ากว่าแบบ ขนาน เพราะการเคลื่อนย้ายข้อมูลแบบอนุกรมนั้นเป็นการส่งข้อมูลครั้งละ 1 บิต แต่พอร์ตขนานนั้นสามารถส่งข้อมูลที ละหลายๆ บิตพร้อมๆกันได้ แต่ข้อดีของการสื่อสารข้อมูลแบบอนุกรมคือ สามารถส่งข้อมูลได้ในระยะทางที่ไกลกว่าแบบขนาน และใช้สายสัญญาณที่น้อยกว่าการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สื่อสารข้อมูลแบบขนาน

ประเภทของการสื่อสารแบบอนุกรมแบ่งตามลักษณะสัญญาณในการส่งแบ่งได้ 2 แบบ คือ

1.การสื่อสารแบบซิงโครนัส (Synchronous) เป็นการสื่อสารข้อมูลโดยใช้สัญญาณนาฬิกาในการควบคุมจังหวะของการรับส่งสัญญาณ

2.การสื่อสารแบบอะซิงโครนัส (Asynchronous) เป็นการสื่อสารที่ใช้สายข้อมูลเพียงตัวเดียว จะใช้รูปแบบของการส่งข้อมูล(Bit Pattern) เป็นตัวกำหนดว่าส่วนไหนเป็นส่วนเริ่มต้นข้อมูล ส่วนไหนเป็นตัวข้อมูล ส่วนไหนจะเป็นตัวตรวจสอบความถูกต้องของข้อมูล และส่วนไหนเป็นส่วนปิดท้ายของข้อมูลโดยต้องกำหนดให้สัญญาณนาฬิกาเท่ากันทั้งภาคส่งและภาครับ

สำหรับการติดต่อสื่อสารลักษณะนี้จะใช้การรับส่งข้อมูลแบบ Asynchronous คือจะใช้สายข้อมูลเพียงสายเดียว

มาตรฐาน RS-232 เป็นมาตรฐานของการรับส่งข้อมูลแบบอนุกรมที่ได้รับการออกแบบมาเพื่อที่จะทำ ให้อุปกรณ์ต่อพ่วงจากผู้ผลิตต่างกันสามารถทำงานร่วมกันได้ มาตรฐานRS-232 นี้ได้รับความนิยมและใช้กันกว้างขวางมากในปัจจุบัน ซึ่งสามารถแบ่งอุปกรณ์ได้เป็น 2 ประเภทคือ

1. อุปกรณ์ DTE (Data Terminal Equipment) เป็นอุปกรณ์สำหรับส่งข้อมูล (Output) โดยทั่วไปคอนเน็กเตอร์จะเป็นตัวผู้

2. อุปกรณ์ DCE (Data Communication Equipment) เป็นอุปกรณ์สำหรับรับข้อมูล (Input)โดยทั่วไปคอนเน็กเตอร์จะเป็นตัวเมียคอนเน็กเตอร์ที่นิยมใช้จะเป็นชนิด D-Type แบบ 9 ขา และแบบ 25 ขา โดยจะติดตั้งอยู่หลังเครื่องคอมพิวเตอร์ ระดับแรงดันจะมีค่าระหว่าง -3 โวลต์ ถึง-15 โวลต์

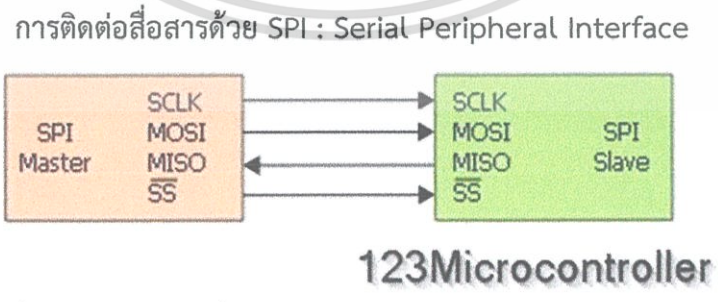
ข้อดีข้อเสียของการใช้พอร์ตอนุกรม (serial port) /ขนาน (parallel port)

-พอร์ตขนานเขียนโปรแกรมรับส่งง่ายและส่งข้อมูลได้อัตราความเร็วสูง

-พอร์ตอนุกรมมีจำนวนเส้นสัญญาณน้อยกว่า ทำให้ประหยัดค่าสายต่างๆมากกว่าแต่ข้อมูลหนึ่งชุดจะต้องเสียเวลาส่งนานขึ้น(เพราะต้องเรียงบิตส่งกันไป)

ปัจจุบันข้อเด่นข้อด้อยดังกล่าวไม่ได้เห็นชัด ทั้งนี้เพราะพอร์ตอนุกรมความเร็วสูงมีแล้ว (USB)ในขณะที่พอร์ตอนุกรมเก่าๆ เองยังต้องมีสายควบคุมมากมาย (acknowledge bus มาก)

2.5 SPI Serial Peripheral Interface



รูปที่ 2.19 การติดต่อสื่อสารด้วย SPI: Serial Peripheral Interface

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาพรวมของ SPI

SPI หรือ Serial Peripheral Interface เป็นวิธีการสื่อสารรูปแบบหนึ่ง ที่ใช้ในการติดต่อสื่อสารกับอุปกรณ์ ตัวอย่างเช่น พริ้นเตอร์ กล้องถ่ายรูป เครื่องสแกนเนอร์ และอื่นๆ อีกมามาย ถึงแม้ว่าการสื่อสารของ USB ที่มีฟังก์ชั่นการทำงานที่กว้างกว่า แต่การสื่อสารในรูปแบบ SPI ก็ยังถูกใช้งานกันอยู่ในบาง Application

SPI ทำงานในรูปแบบที่ให้อุปกรณ์ตัวหนึ่งทำหน้าที่เป็น MASTER ในขณะที่อีกตัวหนึ่งทำหน้าที่เป็น SLAVE และส่งข้อมูลในโหมด Full-duplex นั่นหมายความว่า สัญญาณสามารถส่งหากันได้ระหว่าง MASTER และ SLAVE ได้อย่างต่อเนื่อง ในการสื่อสารแบบ SPI นี้ ไม่ได้มาตรฐานกำหนดตายตัว ว่าข้อมูลที่ส่งหากันต้องอยู่ในรูปแบบหรือ format แบบไหน เป็นการคิด protocol การสื่อสารกันเอาเอง

อุปกรณ์ที่ยังคงมีการใช้การสื่อสารแบบ SPI อยู่

- การแปลงข้อมูลจาก Analog to Digital หรือจาก Digital to Analog
- การติดต่อกับหน่วยความจำ EEPROM และ FLASH
- ไอซีประเภท Real Time Clock : RTC
- เซ็นเซอร์ จำพวก Temperature sensor , Pressure sensor
- อื่น ๆ เช่น signal mixer , Potentiometer , LCD controller , USART , CAN controller , USB controller , Amplifier

พื้นฐานการทำงาน

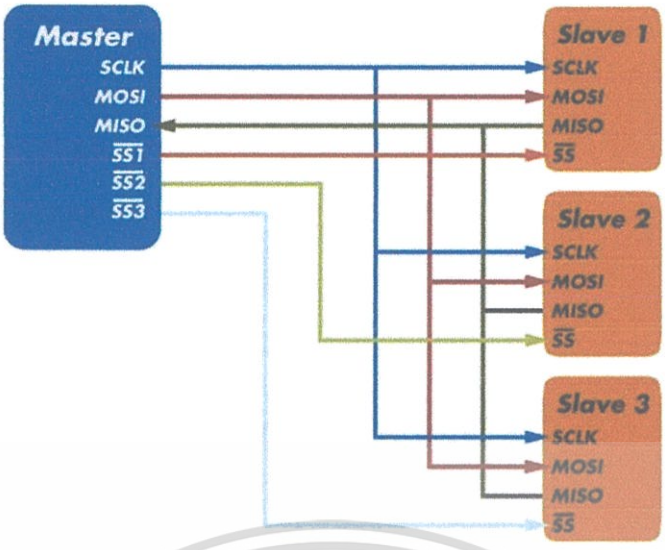
SPI ต้องการสายสัญญาณ สี่เส้น บางครั้งเราเรียกว่าบัสอนุกรม "four wire" เส้นสัญญาณทั้งสี่เส้น ได้แก่

ตารางที่ 2.1 **four wire**

Line	Name	Description
SCLK	Serial Clock	Output from master
MOSI/SIMO	Master Output, Slave Input	Output from master
MISO/SOMI	Master Input, Slave Output	Output from slave
SS	Slave Select	Output from master (active low)

ที่เราเรียกว่า Master ก็เพราะว่า ตัวที่เป็นมาสเตอร์ ทำหน้าที่เป็นตัวควบคุมการสื่อสารทั้งหมด โดยควบคุมการสื่อสารตามสัญญาณนาฬิกา ตัวมาสเตอร์จะเป็นตัวที่ตัดสินใจ รับ หรือ ส่งข้อมูล ภายในการสื่อสาร จะเป็นการสื่อสารแบบ full duplex ในการนำส่งข้อมูล เราใช้ สัญญาณเส้น SS หรือ Slave select ในกรณี ที่เรามีตัว slave มากกว่า 1 ตัว โดยการทำให้เส้น SS มีระดับสัญญาณเป็น Low เมื่อต้องการติดต่อกับ Slave ตัวใด จากรูปด้านล่างหากเราต้องการติดต่อสื่อสารกับ Slave ตัวใด ก็เพียงทำให้สัญญาณ SS ของ Slave ตัวนั้น มีระดับสัญญาณเป็น Low

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.20 การติดต่อสื่อสารด้วย SPI แบบมีหลาย Slave

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

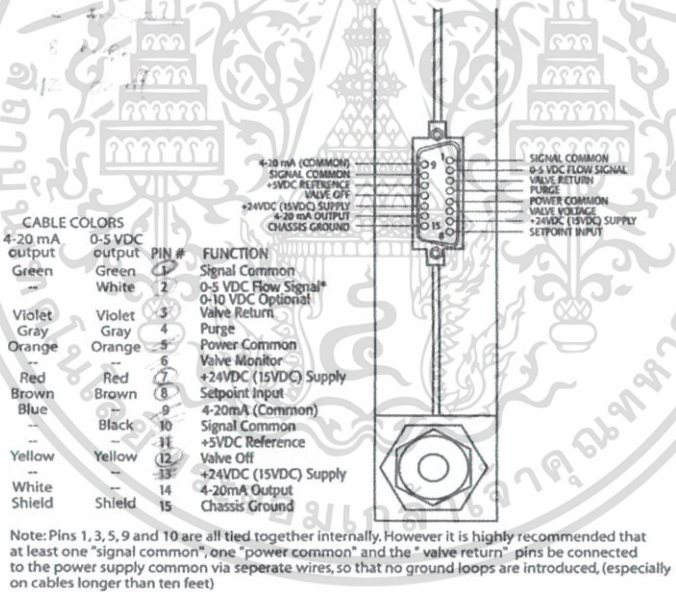
บทที่ 3
ขั้นตอนการดำเนินงาน

เริ่มต้นการออกแบบจะส่งสัญญาณควบคุม และรับค่าตอบกลับจากอุปกรณ์ต่างๆอย่างไร

3.1 อุปกรณ์ที่เราจะทำการควบคุมรับค่าและแสดงผล

- Massflow Controller Argon
- Massflow Controller Nitrogen
- Rotary Pump
- Heater (Diffusion Pump)
- RF Generator
- Water cooler flow

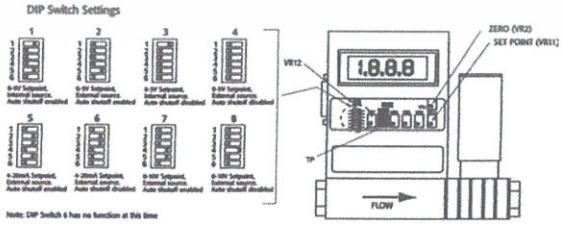
3.1.1 Massflow Controller Argon and Nitrogen



CABLE COLORS

4-20 mA output	0-5 VDC output	PIN #	FUNCTION
Green	Green	1	Signal Common
--	White	2	0-5 VDC Flow Signal* 0-10 VDC Optional
Violet	Violet	3	Valve Return
Gray	Gray	4	Purge
Orange	Orange	5	Power Common
--	--	6	Valve Monitor
Red	Red	7	+24VDC (15VDC) Supply
Brown	Brown	8	Setpoint Input
Blue	--	9	4-20mA (Common)
--	Black	10	Signal Common
--	--	11	+5VDC Reference
Yellow	Yellow	12	Valve Off
--	--	13	+24VDC (15VDC) Supply
White	--	14	4-20mA Output
Shield	Shield	15	Chassis Ground

Note: Pins 1, 3, 5, 9 and 10 are all tied together internally. However it is highly recommended that at least one "signal common", one "power common" and the "valve return" pins be connected to the power supply common via seperate wires, so that no ground loops are introduced, (especially on cables longer than ten feet)

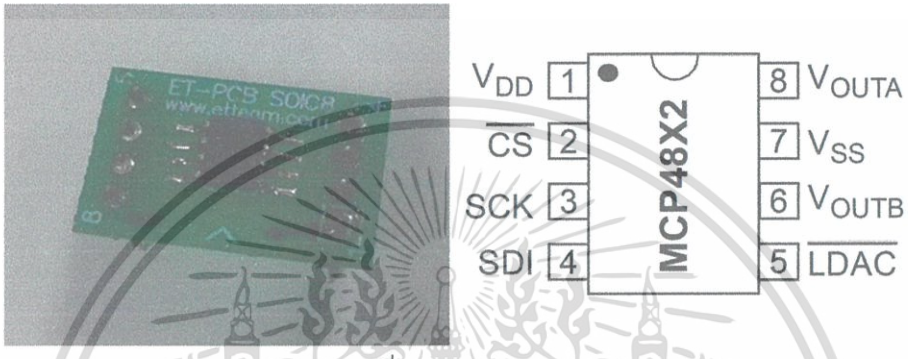


รูปที่ 3.1 Mass Flow Controller Interface Port

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

พอร์ตสำหรับการเชื่อมต่อของ massflow module มีทั้งสิ้น 15 ขา แต่ในที่นี้เราเลือกใช้ 4 ขาคือขา 1 GND, 2 flow Singnal, 7 +24 Volt Supply และ 8 Setpoint

ลักษณะของ Flow Signal และ Set Point คือเป็นสัญญาณอะนาล็อก 0-5 V DC Argon Module อัตราการไหลจะอยู่ที่ 0 -100 sccm ส่วน Nitrogen Module จะอยู่ที่ 0-50 sccm วิธีที่เราจะเลือกใช้คือการใช้ IC Digital to Analog Converter รับค่าจาก Microcontroller จ่ายแรงดันให้กับขา SetPoint และใช้Analog to Digital Converter รับค่าสัญญาณ Voltage การไหลที่แท้จริงไปให้กับ Microcontroller



รูปที่ 3.2 MCP4822

MCP4822 เป็น IC Digital to Analog Converter ความละเอียดขนาด 12 Bit ทำงานที่ไฟเลี้ยง 5 โวลต์ โดยมีการเชื่อมต่อแบบ SPI interface มี OUTPUTA และ OUTPUTB ระดับโวลต์สูงสุดที่ 4095 bit สามารถเลือกระดับโวลต์ output ได้เป็น 2.048 โวลต์ หรือ 4.096 โวลต์ ตามการเลือกใน Bit Selector การสั่งงาน 1 ครั้งจะมีการรับส่งข้อมูลทั้งหมด 16 Bit

Bit ที่ 15 0 เลือก OutputA 1 เลือก OutputB

Bit ที่ 14 เป็น Don't Care

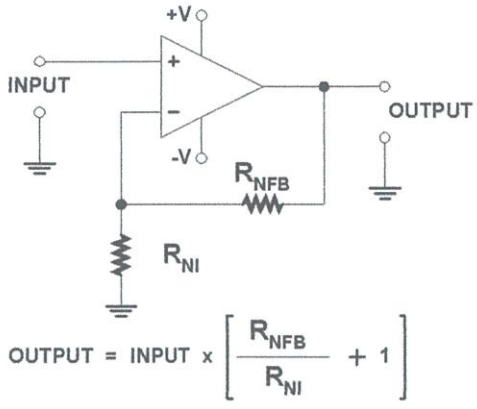
Bit ที่ 13 เลือกระดับแรงดัน Output สูงสุด 0 เลือกเป็น 2.048 V 1 เลือกเป็น 4.096 V

Bit ที่ 12 เลือกการทำงาน 0 เป็น Non-available mode 1 เป็น Active Mode

Bit ที่ 11-0 เป็นข้อมูลจำนวนจริง จำนวณ 12 Bit 0 – 4095

เนื่องจาก Massflow Controller มีระดับแรงดัน input ได้สูงสุด 5 โวลต์ แต่ DAC ตัวดังกล่าวให้แรงดัน Output สูงสุดเพียง 4.096 โวลต์เท่านั้น เพราะฉะนั้นจึงต้องเพิ่มวงจรขยายแบบ Non-inverting Amplifier ก่อนที่จะนำไปใช้ ในที่นี้จะใช้ LM324

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.3 วงจรขยายแบบไม่กลับเฟส

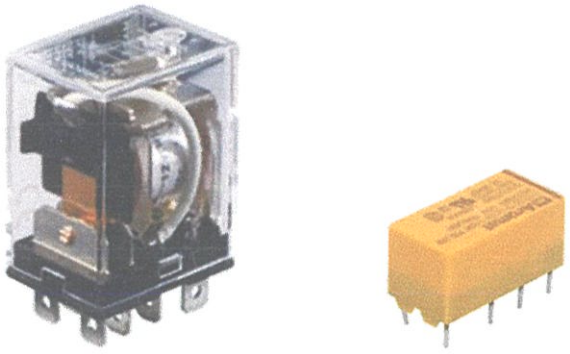
รูปที่ 3.4 MCP3208

MCP3208 เป็น IC Analog to Digital Converter ความละเอียดขนาด 12 Bit ทำงานที่ไฟเลี้ยง 5 โวลต์ โดยมีการเชื่อมต่อแบบ SPI interface มีขารับระดับสัญญาณแรงดัน input 8 ขา การสั่งงาน 1 ครั้งจะมีการรับส่งข้อมูล 4 Bit แรก จะเริ่มต้นเป็น StartBit Bit เลือก mode single หรือ Drifferential mode 3 Bit ต่อมา จะเป็นตัวเลือก Channel ที่ต้องการจะวัดแรงดัน input

3.1.2 Rotary Pump and Heater (Diffusion Pump)

อุปกรณ์ใช้ไฟขนาด 220 โวลต์ การออกแบบการควบคุม 2 สิ่งนี้โดยใช้ Relay ที่มีขนาดของคอยล์ 24 โวลต์เพื่อเปิดปิดไฟบ้าน ในที่นี้จะใช้ Relayตัวเล็กขนาดคอยล์ 5V DC เพื่อจ่ายไฟ 24 โวลต์ให้กับ Relay ตัวใหญ่อีกต่อหนึ่ง

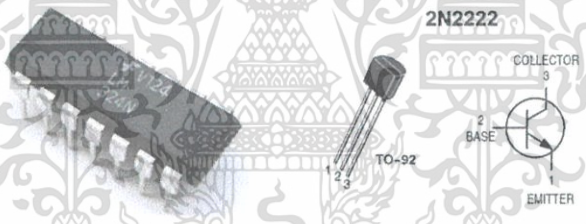
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



(ก) (ข)

รูปที่ 3.5 Relay coil 24V (ก) และ Relay coil 5V(ข)

ตอนนี้เราจะให้ Microcontroller ควบคุมการเปิดปิด Relay ตัวเล็ก แต่เนื่องจากขา I/O Output แต่ละขามีความสามารถในการจ่ายกระแสได้เพียง 40 mA คงจะไม่ส่งผลดีนักหากใช้ I/O Pin ของ Microcontroller ในการจ่ายกระแสให้กับ Relay โดยตรง เราจึงใช้ Op-amp และ Transistor ทำหน้าที่เป็นสวิตช์จ่ายกระแสแทน โดยใช้ Op-amp Lm324 และ 2N2222



รูปที่ 3.6 LM324(ก) 2n2222(ข)

3.1.3 RF Generator

รายชื่อ DB-25 Pin ด้านหลังของเครื่อง RF OEM 650A

Pin Name Type Description

1 Max. Power (E)

2 Reflected power(+)

3 Forward power (+)

4 RF ON/OFF Control (+)

5 Power set point (+)

6 +28VDC

7 RF ON (E)

8 Analog remote

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

9 Overheat (E)
10 Interlock
11 Leveling select
12 Load power (+)
13 +15VDC
14 Max. power (C)
15 Reflected power return (-)
16 Forward power return (-)
17 RF ON/OFF Control (-)
18 Power set point return (-)
19 GND Signal / Chassis ground.
20 RF ON (C)
21 GND Signal / Chassis ground.
22 Overheat (C)
23 Interlock
24 Option V User voltage for interface purposes
25 Load power return (-)

Note 1: *For all isolated transistor outputs.*
Transistor OFF (switch open) - VCEmax = 40VDC (IC < 500uA)
Transistor ON (switch closed) - ICmax = 10mA (VCE < 1V)

Note 2: *For all ground referenced logic level inputs.*
HIGH = 2VDC min to 30VDC max.
LOW = -0.2VDC min to 1VDC max.

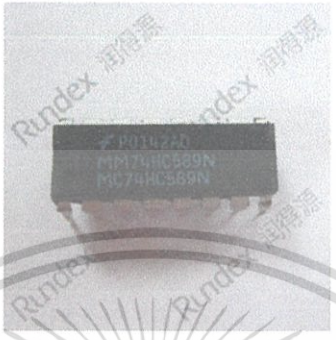
ส่วนแรก คือ การเปิดปิดเครื่อง RF ทำได้โดยการ Jump ขา 10 กับขา 23 เข้าด้วยกัน เครื่อง RF จึงจะสามารถเริ่มทำงานได้ และการสั่ง RF ON ต้องป้อนไฟ 15 โวลต์เข้าที่ ขา 4 ของ DB-25 ทั้งสองกระบวนการนี้เราใช้ Relay 5 โวลต์ ในการสั่งการเช่นเดียวกับ Rotary และ Heater Relay ตัวหนึ่งทำหน้าที่สับสวิตซ์ให้ขา 10 กับ ขา23 เชื่อมถึงกัน Relay อีกตัวหนึ่งทำหน้าที่สับสวิตซ์ให้แรงดัน +15 โวลต์กับขา 4

ส่วนที่ 2 คือ Power Setpoint, Forward Power และ Reflect Power Power Setpoint คือขาที่เราต้องจ่ายแรงดันเพื่อควบคุม Power RF โดยใช้แรงดันสูงสุดคือ 10 โวลต์ สำหรับ Power 650 วัตต์ การควบคุมในส่วนนี้ใช้วงจรเดียวกับที่ออกแบบให้กับ Massflow Controller โดยเพิ่มวงจรขยายในตอนท้ายให้ขยายแรงดันสูงสุดจาก 4.096 โวลต์ ให้กลายเป็น 10 โวลต์ Forward Power คือขาที่ให้สัญญาณโวลต์ส่งกลับมาเพื่อบอกถึง Power ที่ส่งออกไปจริง Reflect Power เช่นเดียวกันกับ Forward Power แต่บอกถึง Power ที่ถูกสะท้อนกลับมายังแหล่งกำเนิด การรับค่าสัญญาณทั้ง 2 นี้ ใช้ IC Analog to Digital Converter ตัวเดียวกับ Massflower Controller

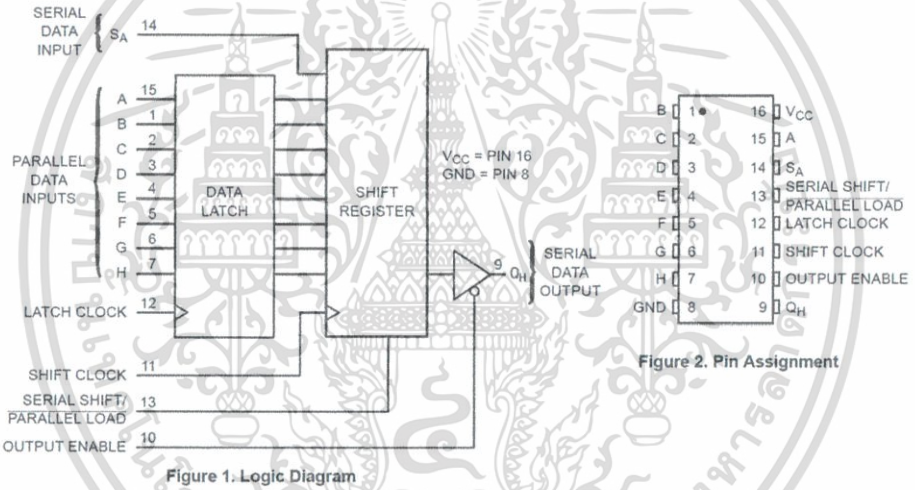
ส่วนที่ 3 คือการรับรู้สถานะต่างๆของเครื่อง มี 3 อย่าง คือ RF_ON, Maxpower และ Overheat สถานะทั้ง 3 ถูกระบุโดย Transistor ที่ติดตั้งอยู่ในเครื่อง โดยถ้า Transistor ON หมายถึง เกิดสถานะนั้นๆขึ้น เราจึงออกแบบโดยการต่อ Pull-up Resistor 5 โวลต์ ให้กับขา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

collector ของ Transistor แต่ละตัวและต่อขา Emitter ลง ground แล้วคอยวัดสัญญาณว่าแรงดันตรงขา Collector นั้นเป็น 5 โวลต์ หรือ 0 โวลต์ ถ้า เป็น 5 โวลต์แสดงว่า Transistor กำลัง Off อยู่ ก้คือยังไม่เกิดสถานะดังกล่าว แต่ถ้าเป็น 0 โวลต์ คือกำลังเกิดสถานะนั้นอยู่ ในการรับรู้ระดับแรงดันตรงส่วนนี้เป็นสัญญาณที่มีแค่ 5 กับ 0 โวลต์ เพราะฉะนั้นเราจะไม่ใช้ Analog to Digital Converter เช่นเดียวกับ Massflow Controller แต่จะใช้ไอซีเพิ่มขึ้นมาอีก 1 ตัว คือ Shift-Register



MC74HC589A

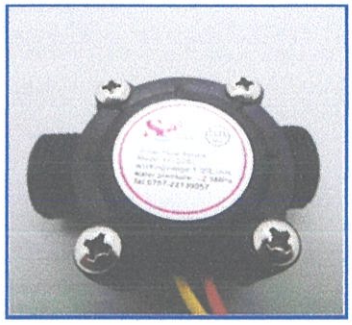


รูปที่ 3.7 Shift-Register 74589 logic diagram

74ls589 เป็น Shift-Register Pallarel in Serial out ขนาด 8 Bit เราใช้ Bit ที่ 1,2 และ 3 ในการรับรู้สถานะทั้ง 3 ของเครื่อง RF แล้วส่งไปที่ Microcontroller

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.1.4 Water cooler



รูปที่ 3.8 YF121 Water Flow Sensor

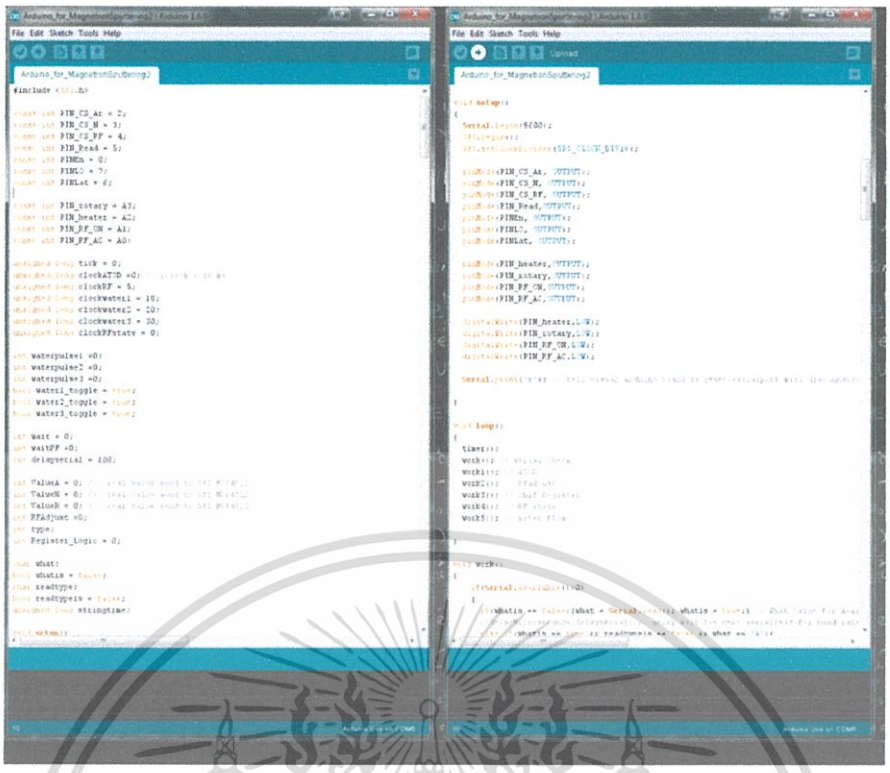
YF 121 เป็น Sensor ตรวจจับการไหลของน้ำโดยมีสายไฟ 3 เส้น คือ ไฟเลี้ยง 5 โวลต์ ground และ สัญญาณ Output ซึ่งมีลักษณะเป็น Square wave โดยความถี่ของสัญญาณ ขึ้นอยู่กับอัตราไหลของน้ำ ถ้าไหลมากความถี่ก็จะมาก ในการวัดสัญญาณจาก Sensor ตัวนี้ เราใช้ Shift-Register ตัวเดียวกับที่ใช้รับสถานะจาก RF Generator Square wave ณ เวลาที่เป็น HIGH เมื่อ Register เก็บค่าจะเจอข้อมูลที่เป็น 1 ณ เวลาที่เป็น LOW ก็จะเจอค่าที่เป็น 0

ทั้งหมดนี้คือการออกแบบวงจรเพื่อใช้ในการส่งและรับข้อมูลจาก Device และ Sensor ต่างๆซึ่งจังหวะการรับและการส่งข้อมูลนั้น จะถูกควบคุมโดย Microcontroller

3.2 โปรแกรม Microcontroller

หน้าที่ของ Microcontroller นั้น คือการรับค่าที่ต้องการ Set อุปกรณ์ต่างๆจาก Computer แปลความหมายและส่งข้อมูลไปให้วงจร หรือรับข้อมูลจากวงจรส่วนต่างๆ คำนวณแล้วส่งกลับไปให้ Computer ในการออกแบบครั้งนี้การเชื่อมต่อระหว่าง Microcontroller กับ Computer นั้นจะเชื่อมต่อผ่าน USB แล้วแปลงเป็น Serial ลักษณะของข้อมูลที่รับส่งจะเป็นข้อมูลชนิด “String” ซึ่งก็คือรหัส Ascii นั่นเอง และการสื่อสารระหว่าง Microcontroller กับ IC ต่างๆ จะกระทำผ่านกระบวนการ SPI Microcontroller ที่เราเลือกใช้ในงานนี้คือ Arduino UNO R3

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.9 การเขียนโปรแกรมบน Arduino IDE

หน้าที่การทำงานหลักๆใน Loop มีทั้งหมด 6 งาน

1. Serial check คือการตรวจข้อมูลที่ถูกส่งจาคอมพิวเตอร์ผ่านทาง Serialport ว่ามีการสั่งให้เปิด-ปิด อุปกรณ์ หรือ สั่งปรับแรงดัน Output หรือปรับเปลี่ยนค่า Setpoint หรือไม่

2. ATOD คือการรับค่า Analog ที่วัดได้จากอุปกรณ์ต่างๆ แล้วส่งไปให้ Computer

3. RFadjust การทำงานของการปรับ Power ให้กับ RF Generator นั้น ต่างกับการปรับ Massflow Controller ตรงที่การปรับ Massflow เมื่อได้รับคำสั่งมาให้ Gas ไหล ณ ปริมาณที่ต้อการค่าๆหนึ่ง แรงดันที่จ่ายให้กับ Device จะถูกจ่ายออกไปค่าเท่านั้นทันที แต่สำหรับRF นั้น การเปลี่ยนแปลง Power อย่างฉับพลันนั้นจะส่งผลให้เครื่อง Overhear ได้ ดังนั้นเมื่อได้รับค่า Setpoint power ที่ต้องการแล้วคำสั่งชุดนี้จะค่อยๆเพิ่ม หรือลด Power ทีละเล็กน้อยจนถึงจุดที่กำหนด

4. Shift Register คือการรับข้อมูลทั้ง 6 Bit ที่สนใจจาก Shift-Register คือสถานะของ RF ทั้ง 3 ตัว และสัญญาณการไหลของน้ำอีก 3 ตัว

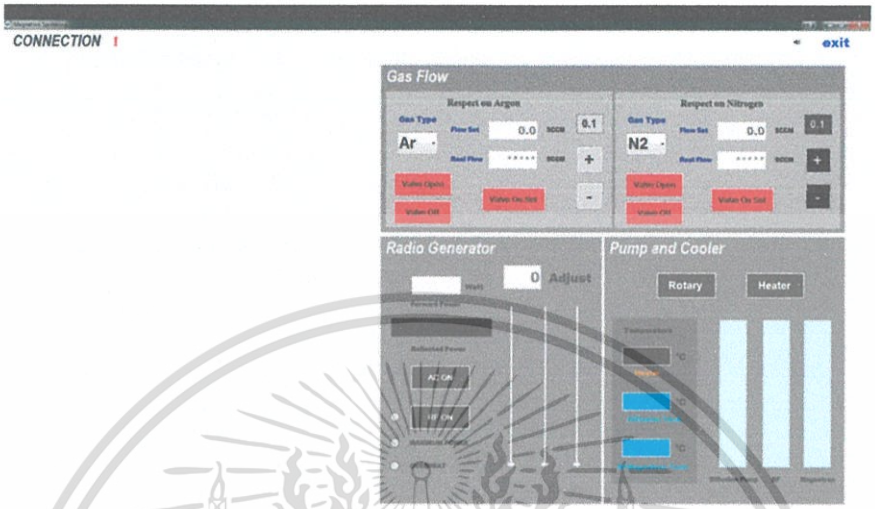
5. RF state คือการวิเคราะห์ข้อมูลที่ได้มาจาก Shift-Register ว่าสถานะ RF ทั้ง 3 กำลังเป็นอย่างไร รวมทั้งส่งผลไปให้ Computer ด้วย

6. Water flow คือการนำผลที่ได้จาก Shift-Register มาวิเคราะห์เมื่อครบ 1 วินาที เพื่อหาความถี่ของสัญญาณที่ได้มาจาก Sensor Water Flow คำนวณค่าอัตราการไหลแล้วส่งผลไปให้ Computer

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

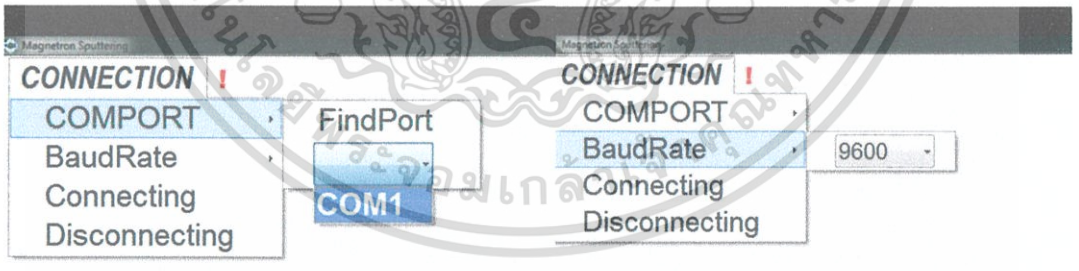
3.3 โปรแกรม Visual c++

ในส่วนของโปรแกรม Computer คือการสร้างหน้าต่าง Interface กับผู้ใช้ เพื่อให้ผู้ใช้สามารถคลิก เลือกกดปุ่มบนหน้าจอแล้วสั่งการไปยัง Microcontroller และส่วนที่รับข้อมูล มาแสดงผล สถานะต่างๆ



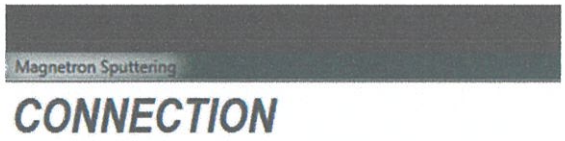
รูปที่ 3.10 Window Form บน Visual C++

หน้าต่างเริ่มต้นเมื่อเปิดโปรแกรมขึ้นมา เครื่องหมายตกใจสีแดงที่ปรากฏ บ่งบอกถึง Computer ยังไม่ได้เชื่อมต่อกับ Microcontroller เริ่มต้นการใช้งาน คือการกดที่ปุ่ม connection เลือก COM PORT Buad Rate จะถูกตั้งไว้โดยอัตโนมัติ เมื่อกด Connecting ถ้าการเชื่อมต่อเริ่มขึ้น เครื่องหมายตกใจจะหายไป



(ก) (ข)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

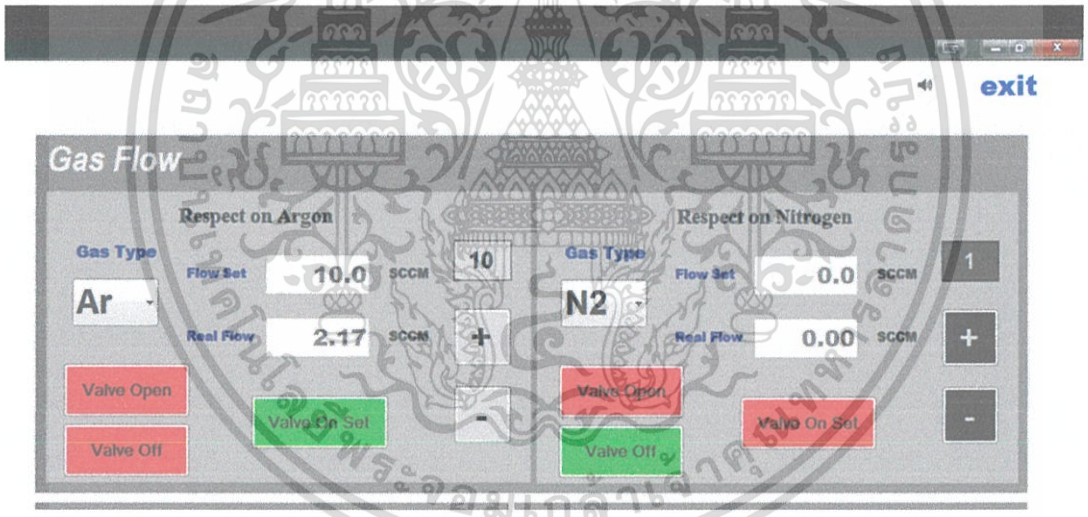


(ค)

รูปที่ 3.11 การเลือก COMPOT(ก) BuadRate(ข) รูปเมื่อมีการเชื่อมต่อกับMicrocontrollerแล้ว(ค)

หน้าต่างเริ่มต้นเมื่อเปิดโปรแกรมขึ้นมา เครื่องหมายตกใจสีแดงที่ปรากฏ บ่งบอกถึง Computer ยังไม่ได้เชื่อมต่อกับ Microcontroller เริ่มต้นการใช้งาน คือการกดที่ปุ่ม connection เลือก COM PORT Buad Rate จะถูกตั้งไว้โดยอัตโนมัต เมื่อกด Connecting ถ้าการเชื่อมต่อเริ่มขึ้น เครื่องหมายตกใจจะหายไป

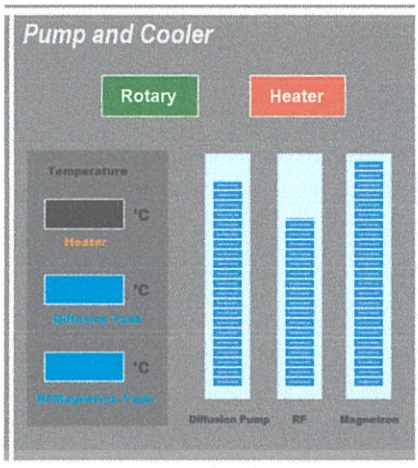
ในโปรแกรม มี Panel หลักๆอยู่ทั้งหมด 3 Panel



รูปที่ 3.12 Gas Flow Panel

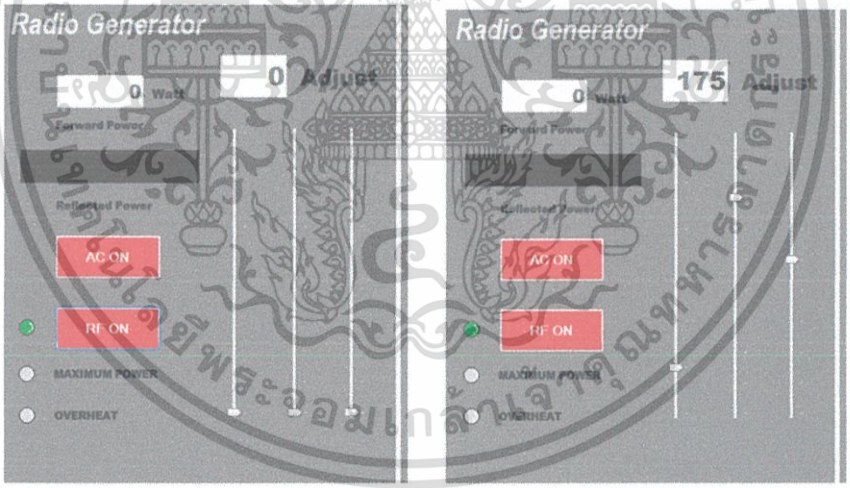
ส่วนแรกคือ Gas Flow panel ย่อยทางซ้ายจะใช้สำหรับ Argon ทางขวาจะเป็น Nitrogen ลักษณะการสั่งคือจะให้ผู้ใช้งานเริ่มเลือกสถานะ ว่าจะให้ Gas off คือปิด ,open คือไหลเต็มที่ หรือ On Set คือให้ไหลตามค่าในช่อง Flow set ส่วนค่าในช่อง Real Flow คือค่าที่ Gas ไหลจริง ระหว่างที่เลือก Valve On set ถ้ากดปุ่ม + หรือ – ด้านข้าง ค่า Flow set จะเพิ่มขึ้นตาม ปุ่มด้านบนสามารถเลือกได้ว่าจะให้การเพิ่มลดทีละเท่าไหร่ มี 3 ค่า คือ 0.1, 1 และ 10 เลือกได้โดยการคลิก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.13 Pump and Cooler Panel

ส่วนที่ 2 Pump and cooler จะมีปุ่มใหญ่ๆอยู่ 2 ปุ่มคือทางซ้ายใช้สำหรับเปิด-ปิด Rotary Pump ถ้าขึ้นสีเขียวแสดงว่าให้เปิด สีดำแสดงว่าปิด ปุ่ม Heater ก็เช่นกัน เมื่อกดแล้วปุ่มกลายเป็นสีส้ม แสดงว่า Heater On ถ้าเป็นสีดำแสดงว่า Heater off แท่งสีฟ้าทั้ง 3 บอกถึงอัตราการไหลของน้ำทั้ง 3 สาย ไปหล่อเย็น Diffusion Pump เครื่อง RF Generator และ หัว Magnetron



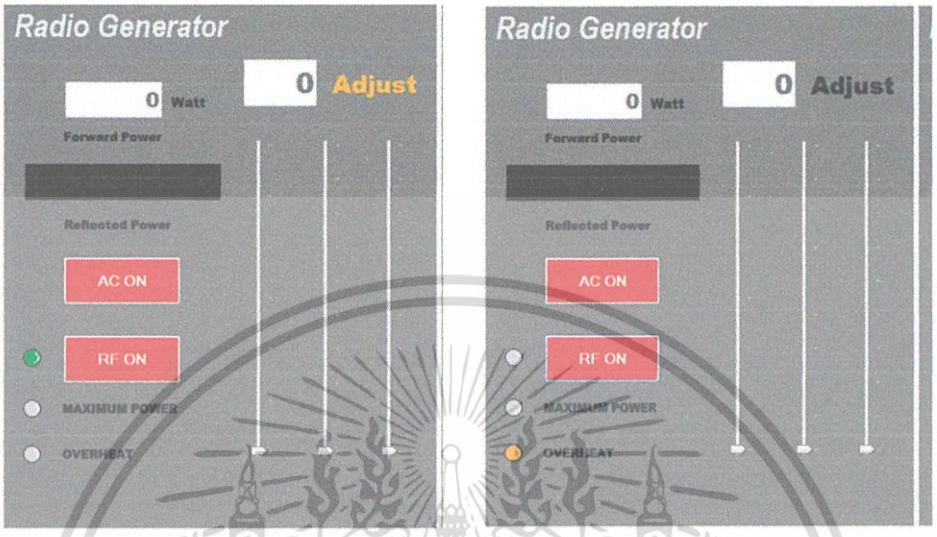
(ก) (ข)

รูปที่ 3.14 ถ้าสถานะเมื่อ RF พร้อมใช้งาน (ก) การตั้งค่า Setpoint แต่ยังไม่สั่งให้ปรับ Power Output (ข)

ส่วนที่ 3 Radio Generator ในส่วนนี้การเปิดเครื่อง RF จะเริ่มจากการกดปุ่ม AC ON ให้ปุ่มเป็นสีแดงเพื่อให้ไฟจ่ายเข้าไปที่เครื่อง หลังจากนั้นกดปุ่ม RF ON ไฟสถานะสีเขียวจะแสดงขึ้น แสดงว่า RFพร้อมที่จะใช้งานแล้ว แท่นเลื่อนทั้ง 3 ที่อยู่ด้านขวาเป็นตัวกำหนด Power ที่ผู้ใช้งานต้องการ สามารถทำการเลื่อนเพื่อให้ได้ค่าที่ต้องการ ค่า setpoint จะถูกแสดงอยู่บนช่องด้านบน แต่Power จะยังไม่ถูกสั่งให้ดำเนินการปรับจริงจนกว่าผู้ใช้งานจะกดที่ตำแหน่ง Adjust เมื่อกดปุ่ม Adjust แล้ว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปุ่มกลายเป็นสีส้ม Power RF จะค่อยถูกปรับเข้าหาจุด Setpoint อย่างช้าๆ ถ้าระหว่างที่ Power กำลังถูกปรับหรือปรับเสร็จแล้วก็ตาม ถ้าผู้ใช้งานกด Adjust อีกครั้งหนึ่ง (กลายเป็นสีดำ) power จะคงสภาวะล่าสุดไว้จนกว่าจะมีการ Adjust อีกครั้งหนึ่ง ในขณะที่มี Output ในช่อง Forward Power จะแสดงค่า Power จริงที่ถูกส่งออกไปขณะนั้น Reflect Bar จะแสดงขีดสัญญาณที่ถูกสะท้อนกลับ



(ก) (ข)

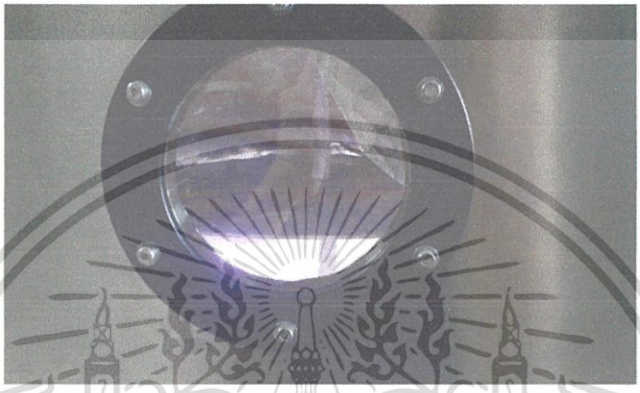
รูปที่ 3.15 ถ้าสถานะเมื่อ RF ปรับ Power Output(ก) สถานะเมื่อ Overheat (ข)

ในกรณีที่มีการผิดพลาด เกิด Overheat ขึ้นไฟสส้มจะขึ้น ไฟสีเขียวที่ RFON จะหายไป ค่า Setpoint ที่ผู้ใช้งานตั้งไว้ และค่า Power ที่กำลังจ่ายอยู่ล่าสุด จะถูกรีเซตให้กลายเป็น 0 ในขณะที่ OverHeat ผู้ใช้งานจะไม่สามารถกดปุ่ม Adjust หรือเลื่อนปรับ แท่น Setpoint ได้ จนกว่าไฟ Overheat จะหายไป และ RF ON ขึ้นไฟสีเขียวใหม่อีกครั้งหนึ่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4
ผลการทดลองและการอภิปรายผล

เราทำการทดสอบระบบโดยการ Operate ทั้งระบบพร้อมกัน และเคลือบฟิล์มขึ้นมาโดยใช้ระบบควบคุมที่สร้างขึ้น การทดลอง flow gas หลายๆครั้งในปริมาณที่เท่าเดิม ภายใต้ condition เดียวกัน แล้วสังเกตความดัน ผลคือ ได้ค่าความดันที่เท่ากันทุกครั้ง



รูปที่ 4.1 ภายในระบบ ขณะเคลือบ Aluminium

รูปที่ 4.2 ภายในระบบ ขณะเคลือบ Aluminium Nitride

ในส่วนของ RF การ Setpower มีความต่อเนื่องดี ผู้ใช้งานสามารถปรับ Power ขึ้นหรือลงได้ตามต้องการ การเพิ่มและการลดมีอัตราต่ำ ไม่เร็วเกินไป การป้องกันเมื่อเกิด OverHeat สมบูรณ์ไม่บกพร่อง ป้องกันความเสียหายของตัวอุปกรณ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 4..3 ฟิล์ม Aliminium

รูปที่ 4.4 ฟิล์ม Aliminium Nitrie

ผลของการใช้ระบบควบคุมที่สร้างขึ้นมีความสม่ำเสมอในระดับที่น่าพอใจ ระดับ Gas Flow และ Power RF มีความคลาดเคลื่อนน้อย สถานะของการ Flow water cooling ถูกต้อง สถานะของ RFON Maxpower และ Overheat ถูกต้อง การเปิดปิด Rotary และ Heater มีความถูกต้อง และสะดวกรวดเร็ว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5
สรุปผลและข้อเสนอแนะ

5.1 สรุปผลการวิจัย

การเคลือบฟิล์มโดยวิธี RF Magnetron Sputtering โดยใช้โปรแกรมและชุดควบคุมที่สร้างขึ้นมานั้น มีประสิทธิภาพ สามารถควบคุม Parameter ของอุปกรณ์ต่างๆได้อย่างแม่นยำ ฟิล์มที่เคลือบออกมามีคุณภาพดี และในขณะดำเนินงาน ผู้ใช้งานมีความสะดวกในการควบคุมปัจจัยหลายๆ อย่าง ทั้งการควบคุมการไหลของแก๊สทั้ง 2 ชนิด และ การปรับค่า Power ของ RF Generator ไปในเวลาเดียวกัน รวมทั้งสามารถรับรู้สภาวะของระบบขณะทำงาน เพื่อป้องกันความผิดพลาดและการเกิดอุบัติเหตุได้เป็นอย่างดี

5.2 ข้อเสนอแนะ

การพัฒนาต่อเพื่อให้สมบูรณ์ ความต้องการท้ายที่สุดคือให้อุปกรณ์ต่างๆของระบบถูกควบคุมกาเปิดปิดผ่านทางคอมพิวเตอร์ ทั้งการเปิดปั๊มน้ำ, Butterfly Valve , Isolate Valve ตลอดไปจนถึงสามารถทำการ Sputter แบบ Auto Pilot ได้ นั่นคือระบบสามารถทำงานเองได้ด้วยตัวเองโดยที่ผู้ใช้งานไม่ต้องควบคุม แต่เพียงแค่ใส่ชนิดและความหนาของฟิล์มที่ต้องการเท่านั้น จำเป็นจะต้องใช้บุคลากรในการพัฒนาระบบที่มากขึ้นเพื่อแบ่งงานกันทำแต่ละส่วน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บรรณานุกรม

Arduino products and language reference

[Online]. Available : http://www.arduino.cc/

Microsoft Visual C++

[Online]. Available : https://msdn.microsoft.com/

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ก

```
#include <SPI.h>

const int PIN_CS_Ar = 2;
const int PIN_CS_N = 3;
const int PIN_CS_RF = 4;
const int PIN_Read = 5;
const int PINEn = 8;
const int PINLO = 7;
const int PINLat = 6;

const int PIN_rotary = A3;
const int PIN_heater = A2;
const int PIN_RF_ON = A1;
const int PIN_RF_AC = A0;

unsigned long tick = 0;
unsigned long clockATOD =0; // 1clock = 10 ms
unsigned long clockRF = 5;
unsigned long clockwater1 = 10;
unsigned long clockwater2 = 20;
unsigned long clockwater3 = 30;
unsigned long clockRFstate = 0;

int waterpulse1 =0;
int waterpulse2 =0;
int waterpulse3 =0;
bool water1_toggle = true;
bool water2_toggle = true;
bool water3_toggle = true;

int wait = 0;
int waitRF =0;
int delayserial = 100;

int ValueA = 0; //  real value send to SPI MCP4822
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
int ValueN = 0; //  real value send to SPI MCP4822
int ValueR = 0; //  real value send to SPI MCP4822
int RFAdjust =0;
int type;
int Register_Logic = 0;

char what;
bool whatis = false;
char readtype;
bool readtypeis = false;
unsigned long stringtime;

void setup()
{
  Serial.begin(9600);
  SPI.begin();
  SPI.setClockDivider(SPI_CLOCK_DIV16);

  pinMode(PIN_CS_Ar, OUTPUT);
  pinMode(PIN_CS_N, OUTPUT);
  pinMode(PIN_CS_RF, OUTPUT);
  pinMode(PIN_Read,OUTPUT);
  pinMode(PINEn, OUTPUT);
  pinMode(PINLO, OUTPUT);
  pinMode(PINLat, OUTPUT);

  pinMode(PIN_heater,OUTPUT);
  pinMode(PIN_rotary,OUTPUT);
  pinMode(PIN_RF_ON,OUTPUT);
  pinMode(PIN_RF_AC,OUTPUT);

  digitalWrite(PIN_heater,LOW);
  digitalWrite(PIN_rotary,LOW);
  digitalWrite(PIN_RF_ON,LOW);
  digitalWrite(PIN_RF_AC,LOW);

  Serial.print('r'); // tell visual Arduino board is reset(serialport will disconnect) try to
connect again
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
}

void loop()
{
  timer();
  work(); // serial check
  work1(); // ATOD
  work2(); // RFadjust
  work3(); // Shif Register
  work4(); // RF state
  work5(); // water flow

}

void work()
{
    if(Serial.available()>0)
    {
      if(whatis == false){what = Serial.read(); whatis = true;} // What Value for A=Ar
N=N R=RF P=Pressue Gauge
      //delayMicroseconds(delayserial);// delay wait foe next serialread for buad rate ...
      else if(whatis == true && readtypeis == false && what == 'A'){
                          readtype = Serial.read();  readtypeis = true;
                          if(readtype == 'C'){setOutput('A',0); whatis = false; readtypeis =
false; }
                          else if(readtype == 'F'){setOutput('A',4095); whatis = false;
readtypeis = false; }
                          else{stringtime = millis();}
                          }
      else if(whatis == true && readtypeis == true && what == 'A' && stringtime <
(millis()-50) ){
                          String txt = Serial.readStringUntil('s');
                          float total = txt.toFloat();
                          if(readtype == 'A'){ValueA = total*40.95;}
                          else if(readtype == 'N'){ValueA = (total/1.39)*40.95;}
                          else if(readtype == 'O'){ValueA = (total*0.993/1.39)*40.95;}
                          else if(readtype == 'H'){ValueA = (total*1.01/1.39)*40.95;}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    setOutput('A',ValueA);
                    whatis = false; readtypeis = false;

                    }
    else if(whatis == true && readtypeis == false && what == 'N'){
                    readtype = Serial.read();  readtypeis = true;
                    if(readtype == 'C'){setOutput('N',0); whatis = false; readtypeis =
false;}
                    else if(readtype == 'F'){setOutput('N',4095); whatis = false;
readtypeis = false;}
                    else{stringtime = millis();}
                    }
    else if(whatis == true && readtypeis == true && what == 'N' && stringtime <
(millis()-50)){
                    String txt = Serial.readStringUntil('s');
                    float total = txt.toFloat();
                    if(readtype == 'N'){ValueN = total*81.9;}
                    else if(readtype == 'A'){ValueN = (total*1.39)*81.9;}
                    else if(readtype == 'O'){ValueN = (total*0.993)*81.9;}
                    else if(readtype == 'H'){ValueN = (total*1.01)*81.9;}
                    setOutput('N',ValueN);
                    whatis = false; readtypeis = false;
                    }


    else if(whatis == true && readtypeis == false && what == 'f'){ // f for function
                    readtype = Serial.read();
                    readtypeis = true;}
    else if(whatis == true && readtypeis == true && what == 'f'){
                    if(readtype == 'H'){
                    char command = Serial.read();
                    if(command == 'O'){digitalWrite(PIN_heater,HIGH); } // on heater
                    else if(command == 'C'){digitalWrite(PIN_heater,LOW); }} // cut
heater
                    else if(readtype == 'R'){
                    char command = Serial.read();
                    if(command == 'O'){digitalWrite(PIN_rotary,HIGH); } // on heater
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    else if(command == 'C'){digitalWrite(PIN_rotary,LOW);}} // cut
heater
                    whatis = false; readtypeis = false;}

    else if(whatis == true && readtypeis == false && what == 'R'){ // f for function
                    readtype = Serial.read();
                    readtypeis = true;
                    if(readtype =='A'){stringtime = millis();}
                    else if(readtype == 'S'){ValueR = RFAdjust; whatis = false;
readtypeis = false;} // stop adjusting
                    else if(readtype == 'a'){digitalWrite(PIN_RF_AC,HIGH); whatis =
false; readtypeis = false;}// AC ON
                    else if(readtype == 'b'){digitalWrite(PIN_RF_AC,LOW); whatis =
false; readtypeis = false;}// AC OFF
                    else if(readtype == 'O'){digitalWrite(PIN_RF_ON,HIGH); whatis =
false; readtypeis = false;}// RF ON
                    else if(readtype == 'C'){digitalWrite(PIN_RF_ON,LOW); ValueR =0;
RFAdjust =0; setOutput('R',0); whatis = false; readtypeis = false;}}// RF cut}
    else if(whatis == true && readtypeis == true && what == 'R' && stringtime <
(millis()-50)){
                    String txt = Serial.readStringUntil('s');
                    int total = txt.toInt();
                    ValueR = total*6.3;
                    whatis = false; readtypeis = false;}
  }


}
//
//
//
void work3(){//////////////////////////
  digitalWrite(PINEn,LOW);  // Enable
  digitalWrite(PINLat,HIGH); // latch
  digitalWrite(PINLO,HIGH); // load

  Register_Logic = SPI.transfer(0x00);
  digitalWrite(PINLO,LOW);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  digitalWrite(PINLat,LOW);
  digitalWrite(PINEn,HIGH);
}
//
//
//
void work4(){ ////////////////// RF State ////////////////////

    if(clockRFstate >= 10)
      {clockRFstate = 0;
    int Overheat = (Register_Logic >> 2)&1;
    if(Overheat ==1){Serial.print('2'); Serial.print('H');
                    }
    if(Overheat ==0){Serial.print('2'); Serial.print('L');
                        ValueR =0; RFAdjust =0; setOutput('R',0);
                       }

    int Maxpower = (Register_Logic >> 1)&1;
    if(Maxpower ==1){Serial.print('1'); Serial.print('H');
                    }
    else if(Maxpower ==0){Serial.print('1'); Serial.print('L');
                         }

    int RFON = Register_Logic &1;
    if(RFON ==1){Serial.print('3'); Serial.print('H');
                }
    else if(RFON ==0){Serial.print('3'); Serial.print('L');
                     }
    }
}
//
//
//
void work5(){ ////////////////////////// water flow//////////////////////
  bool water1 = (Register_Logic >> 3)&1;
  bool water2 = (Register_Logic >> 4)&1;
  bool water3 = (Register_Logic >> 5)&1;
  if(water1 == 1 && water1_toggle){waterpulse1++; water1_toggle = false;}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  if(water1 == 0){water1_toggle = true;}
  if(water2 == 1 && water2_toggle){waterpulse2++; water2_toggle = false;}
  if(water2 == 0){water2_toggle = true;}
  if(water3 == 1 && water3_toggle){waterpulse3++; water3_toggle = false;}
  if(water3 == 0){water3_toggle = true;}

  if(clockwater1 >= 100){
    String a = String(waterpulse1);
    Serial.print('W');
    Serial.print('1');
    Serial.print(a);
    Serial.print("s");
    clockwater1 = 0;
    waterpulse1 = 0;
  }
if(clockwater2 >= 100){
    String a = String(waterpulse2);
    Serial.print('W');
    Serial.print('2');
    Serial.print(a);
    Serial.print("s");
    clockwater2 = 0;
    waterpulse2 = 0;
  }
if(clockwater3 >= 100){
    String a = String(waterpulse3);
    Serial.print('W');
    Serial.print('3');
    Serial.print(a);
    Serial.print("s");
    clockwater3 = 0;
    waterpulse3 = 0;
  }

}
//
//
//
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void work2() ////////////////////////////////////// RF adjust //////////////////////
{
 if(clockRF >=10){clockRF = 0;
 RFadjust(ValueR);}
}

//
//
//
void work1()
{
   if(clockATOD >= 25){
   clockATOD =0;
   ATOD('A');
   ATOD('N');
   ATOD('F');
   ATOD('R');
   //ATOD('H');
   //ATOD('W');
   //ATOD('w');
   }

}

void timer(){
  unsigned long time = millis();
  if(time >= tick+10){tick = millis(); clockATOD++; clockRF++; clockwater1++;
clockwater2++; clockwater3++; clockRFstate++;}
}



//
//  SPI transfer the data 12 bit to the MCP4822
//
void setOutput(char ch,unsigned int val)
{
  if(val<4096)
    {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    byte lowByte = val & 0xff;
    byte highByte = ((val >> 8) & 0xff) | 0x10;

    // 10 mean DACA OUTPUT on mcp4822   ,Vmax=4.08 ,enableOutput

    drive_cs_LOW(ch);
    SPI.transfer(highByte);
    SPI.transfer(lowByte);
    drive_cs_HIGH(ch);
  }
}


//
//  code for receive data from MCP3208 A2D 12 bit  ********** Not finished
//
void ATOD(char a)
{
  digitalWrite(PIN_Read,LOW);
  unsigned int mode;
  unsigned int Ch;
  if(a == 'A'){mode = 0x06; Ch = 0x00;} // Argon
  if(a == 'N'){mode = 0x06; Ch = 0x40;} // Nitrogen
  if(a == 'F'){mode = 0x06; Ch = 0x80;} // RF foward
  if(a == 'R'){mode = 0x07; Ch = 0x00;} // RF reflect
  if(a == 'H'){mode = 0x07; Ch = 0x40;} // heater
  if(a == 'W'){mode = 0x07; Ch = 0x80;} // water1
  if(a == 'w'){mode = 0x07; Ch = 0xc0;} // water2

  unsigned int aa = SPI.transfer(mode); // start bit and choose single-ended mode
  unsigned int bb = SPI.transfer(Ch); // choose channel 00
  unsigned int cc = SPI.transfer(0);
  unsigned int dd = SPI.transfer(0);
  unsigned int ee = SPI.transfer(0);
  digitalWrite(PIN_Read,HIGH);

   unsigned int mcp = ((bb&0x0F)<<8) | cc;
   float massflowvalue;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  if(a=='A'){massflowvalue = (float)mcp/40.95;
            String massflowtext = String(massflowvalue);
            Serial.print(a);
            Serial.print(massflowtext);
            Serial.print("s");}
  else if(a=='N'){massflowvalue = (float)mcp/81.9;
                String massflowtext = String(massflowvalue);
                Serial.print(a);
                Serial.print(massflowtext);
                Serial.print("s");}
  else if(a == 'F' || a == 'R'){
    mcp/=12.6;
    if(mcp >4090){ mode = 0x06; Ch = 0xc0;
                digitalWrite(PIN_Read,LOW);
                unsigned int aa = SPI.transfer(mode); // start bit and choose single-
ended mode
                unsigned int bb = SPI.transfer(Ch); // choose channel 00
                unsigned int cc = SPI.transfer(0);
                unsigned int dd = SPI.transfer(0);
                unsigned int ee = SPI.transfer(0);
                digitalWrite(PIN_Read,HIGH);
                mcp = ((bb&0x0F)<<8) | cc;
                mcp = (mcp/12.6)+325;}

        String textRF = String(mcp);
        Serial.print(a);
        Serial.print(textRF);
        Serial.print("s");
         }
  else if(a == 'H'){mcp = mcp/8.19;
                String temp = String(mcp);
                Serial.print('T');
                Serial.print('1');
                Serial.print(temp);
                Serial.print("s");}
  else if(a == 'W' || a =='w'){ mcp = mcp*0.008547;
                        String temp = String(mcp);
                        Serial.print('T');
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    if(a=='W'){Serial.print('2');}
                    else if(a=='w'){Serial.print('3');}
                    Serial.print(temp);
                    Serial.print("s");}

}
//
//
//
void RFadjust(int a)
{
  if(a > RFAdjust){RFAdjust += 6;  setOutput('R',RFAdjust); }
  if(a < RFAdjust){RFAdjust -= 6;  setOutput('R',RFAdjust); }
}

//
//
//
void drive_cs_LOW(char device)
{
  if(device == 'A'){digitalWrite(PIN_CS_Ar,LOW);}
  if(device == 'N'){digitalWrite(PIN_CS_N,LOW);}
  if(device == 'R'){digitalWrite(PIN_CS_RF,LOW);}
}

//
//
//
void drive_cs_HIGH(char device)
{
  if(device == 'A'){digitalWrite(PIN_CS_Ar,HIGH);}
  if(device == 'N'){digitalWrite(PIN_CS_N,HIGH);}
  if(device == 'R'){digitalWrite(PIN_CS_RF,HIGH);}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ข

```
#pragma endregion


////////////////////////////////////////// Connect Serial Port Process
//////////////////////////////

        private: void findPorts(void)
                        {
                        // get port names
                        this->toolStripComboBox1->Items->Clear();
                    array<Object^>^ COMPORT = SerialPort::GetPortNames();
                        // add string array to combobox
                        this->toolStripComboBox1->Items->AddRange( COMPORT );

                        }

        private: System::Void findPortToolStripMenuItem_Click(System::Object^
sender, System::EventArgs^  e) {
                            findPorts();
                        }


        private: System::Void connectingToolStripMenuItem_Click(System::Object^
sender, System::EventArgs^  e)
                        {
                            if(this->toolStripComboBox1->Text==String::Empty ||
this->toolStripComboBox2->Text==String::Empty){}
                                else{
                                        try{
                                            // make sure port isn't open
                                            if(this->serialPort1->IsOpen ==
false){
                                                this->serialPort1-
>PortName = this->toolStripComboBox1->Text;
                                                //this->textBox1-
>Text=this->comboBox1->Text;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                                                    this->serialPort1-
>BaudRate=Int32::Parse(this->toolStripComboBox2->Text);
                                                    //this->textBox1-
>Text=this->comboBox2->Text;
                                                    this->serialPort1-
>Open();
                                                    this-
>connectingToolStripMenuItem->Enabled = false;
                                                    this-
>toolStripMenuItem1->Visible = false;
                                                    this-
>connectingToolStripMenuItem->Enabled = false;
                                                }
                                            else{
                                                }
                                        }
                                    catch(UnauthorizedAccessException^){}
                                }
                            }

private: System::Void disconnectingToolStripMenuItem_Click(System::Object^  sender,
System::EventArgs^  e)
                {
                    if(this->serialPort1->IsOpen){this->serialPort1->Close();}
                    this->toolStripComboBox1->Items->Clear();
                    if(this->connectingToolStripMenuItem->Enabled == false){this-
>connectingToolStripMenuItem->Enabled = true;}
                    if(this->toolStripMenuItem1->Visible == false){this-
>toolStripMenuItem1->Visible = true;}
                }

////////////////////////////////////// Real time zone
//////////////////////////////////////////////

private: System::Void timer1_Tick(System::Object^  sender, System::EventArgs^  e)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
{

                    if(soundspeak != soundcounting && second >
speaktime+waitspeak[soundspeak]){
                                        speaktime = second;
                                        soundspeak++; if(soundspeak ==
10){soundspeak = 0;}
                                        if(soundarrow ==1){
                                        player->SoundLocation =
soundline[soundspeak];
                                        player->Load();
                                        player->Play();}
                    }

                    if(this->serialPort1->IsOpen== false){
                        connecting++;

                    if(this->toolStripMenuItem1->Visible == false){this-
>toolStripMenuItem1->Visible = true;}
                    this->textBox2->Clear();
                    this->textBox2->Text= "*****";
                    this->textBox4->Clear();
                    this->textBox4->Text= "*****";
                    this->label36->Enabled = false;

                        if(connecting ==1){
                            soundcounting++; if(soundcounting ==
10){soundcounting = 0;}
                                        soundline[soundcounting] =
"d:/project/sound/loss_connection.wav";
                                        waitspeak[soundcounting] = 150;
                                        soundcounting++; if(soundcounting ==
10){soundcounting = 0;}
                                        soundline[soundcounting] =
"d:/project/sound/connecting.wav";
                                        waitspeak[soundcounting] = 250;
                                    }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    if(second== 10){
                        soundcounting++; if(soundcounting ==
10){soundcounting = 0;}
                            soundline[soundcounting] =
"d:/project/sound/welcome.wav";
                            waitspeak[soundcounting] = 230;
                            soundcounting++; if(soundcounting ==
10){soundcounting = 0;}
                            soundline[soundcounting] =
"d:/project/sound/connecting.wav";
                            waitspeak[soundcounting] = 150;

                    }
                }

                try
                {
                if(this->serialPort1->IsOpen)
                {
                    connecting = 0;
                    this->label36->Enabled = true;
                    if(this->serialPort1->BytesToRead)
                    {
                        char Rx = this->serialPort1->ReadChar();
                        if(Rx == 'r')// controller reset
                            {
                                if(this->serialPort1->IsOpen){this-
>serialPort1->Close();}
                                this->toolStripComboBox1-
>Items->Clear();
                                if(this-
>connectingToolStripMenuItem->Enabled == false){this-
>connectingToolStripMenuItem->Enabled = true;}
                                if(this->toolStripMenuItem1-
>Visible == false){this->toolStripMenuItem1->Visible = true;}
                                this->button18->BackColor =
System::Drawing::Color::Red;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                                        this->button19->BackColor =
System::Drawing::Color::Red;
                                        this->button20->BackColor =
System::Drawing::Color::Red;
                                        this->button21->BackColor =
System::Drawing::Color::Red;
                                        this->button22->BackColor =
System::Drawing::Color::Red;
                                        this->button23->BackColor =
System::Drawing::Color::Red;
                                    }
                                else if(Rx == 'A' || Rx == 'N' )
                                    { int Ch;
                                    if(Rx == 'A'){Ch = 1;
                                    try{
                                        String^ massflow1 = this-
>serialPort1->ReadTo("s");
                                        String^ mat =
String::Format("{0:F1}", massflow1);
                                        TXT[1]->Clear();
                                        TXT[1]->Text = mat;
                                        }
                                    catch(TimeoutException^){ }
                                    }
                                    else if(Rx == 'N'){Ch = 3;
                                    try{
                                        String^ massflow1 = this-
>serialPort1->ReadTo("s");
                                        String^ mat =
String::Format("{0:N0}", massflow1);
                                        TXT[3]->Clear();
                                        TXT[3]->Text = mat;
                                        }
                                    catch(TimeoutException^){ }
                                    }
                                }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                                    else if(Rx == 'F'){String^ RF1 = this->serialPort1-
>ReadTo("s"); // foward
                                                        this->textBox9->Clear();
                                                        this->textBox9->Text =
RF1; }

                                    else if(Rx == 'R'){String^ RF1 = this->serialPort1-
>ReadTo("s");// reflect
                                                        int bar =
System::Convert::ToInt32(RF1);
                                                        this->panel15->Location
= System::Drawing::Point(bar, 5);}
                                    else if(Rx == '1'){try{ char onoff = this-
>serialPort1->ReadChar(); // maxpower

                                                        if(onoff ==
'H'){this->pictureBox3->Visible = true; this->pictureBox7->Visible = false;}
                                                        if(onoff == 'L'){this-
>pictureBox3->Visible = false; this->pictureBox7->Visible = true;}

        }catch(TimeoutException^){ }}
                                    else if(Rx == '2'){try{ char onoff = this-
>serialPort1->ReadChar(); // overheat
                                                        if(onoff ==
'H'){this->pictureBox4->Visible = true; this->pictureBox8->Visible = false;}
                                                        if(onoff == 'L'){this-
>pictureBox4->Visible = false; this->pictureBox8->Visible = true;

                RFallow =0;

                this->label36->ForeColor = System::Drawing::Color::Black;

                this->TXT[4]->Text = "0";

                this->trackBar1->Value = 0;

                this->trackBar2->Value = 0;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                this->trackBar3->Value = 0;}

        }catch(TimeoutException^){ }}
                                        else if(Rx == '3'){try{ char onoff = this-
>serialPort1->ReadChar();  // RF on
                                                                        if(onoff ==
'H'){this->pictureBox2->Visible = true; this->pictureBox6->Visible = false;}
                                                                        if(onoff == 'L'){this-
>pictureBox2->Visible = false; this->pictureBox6->Visible = true;}

        }catch(TimeoutException^){ }}

                                        else if(Rx == 'W'){ // water flow
                                                try{
                                                char waterCh = this->serialPort1-
>ReadChar();
                                                String^ waterflow1 = this->serialPort1-
>ReadTo("s");
                                                int bar1 =
System::Convert::ToInt32(waterflow1);
                                                int a = 300-(bar1*14);
                                                if(waterCh =='1'){this->panel9->Size =
System::Drawing::Size(40,a);}
                                                if(waterCh =='2'){a = 300-(bar1*16); this-
>panel11->Size = System::Drawing::Size(40,a);}
                                                if(waterCh =='3'){a = 300-(bar1*8); this-
>panel13->Size = System::Drawing::Size(40,a);}
                                                        }
                                                catch(FormatException^){ }
                                                catch(TimeoutException^){ }

                                        }

                                        /*if(Rx = 'T'){ // water and heater temperature
                                                try{
                                                char temp_channel = this->serialPort1-
>ReadChar();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                                        String^ Temp = this->serialPort1-
>ReadTo("s");
                                        int tempcolor =
System::Convert::ToInt32(Temp);
                                        if(temp_channel == '1'){this->textBox5-
>Text = Temp; // Heater

        int a = tempcolor;

        if(a>=255){a = 255;}

        else if(a<=0){a=0;}

this->textBox5->BackColor =
System::Drawing::Color::FromArgb(static_cast<System::Int32>(static_cast<System::Byte
>(a)),

        static_cast<System::Int32>(static_cast<System::Byte>(0)),

        static_cast<System::Int32>(static_cast<System::Byte>(0)));}
                                        if(temp_channel == '2'){this->textBox6-
>Text = Temp; // water1

        int a = tempcolor*7;

        if(a>=255){a = 255;}

        else if(a<=0){a=0;}

this->textBox6->BackColor =
System::Drawing::Color::FromArgb(static_cast<System::Int32>(static_cast<System::Byte
>(0)),
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
static_cast<System::Int32>(static_cast<System::Byte>(a)),



static_cast<System::Int32>(static_cast<System::Byte>(255)));}
                                        if(temp_channel == '3'){this->textBox7-
>Text = Temp; // water2

int a = tempcolor*7;

if(a>=255){a = 255;}

else if(a<=0){a=0;}

this->textBox7->BackColor =
System::Drawing::Color::FromArgb(static_cast<System::Int32>(static_cast<System::Byte
>(0)),



static_cast<System::Int32>(static_cast<System::Byte>(a)),



static_cast<System::Int32>(static_cast<System::Byte>(255)));}
                                                    }
                                        catch(FormatException^){ }
                                        catch(TimeoutException^){ }}*/
                        }
                    }
                    }
                        catch(ArgumentNullException^){ }
                        catch(ArgumentException^){ }
                        catch(TimeoutException^){ }
                        catch(InvalidOperationException^){ }
                }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
////////////////////////////////////////////////////////////////////////////////////////////////////
//

                /* choose = choose textbox 0 = textbox1(Ar control)
                                                        1 = textbox2(Ar Read  )
                                                        2 = textbox3(N  control)
                                                        3 = textbox4(N  Read  )
                                                        4 = textbox5(RF
control)
                                                        5 = textbox6(RF Read
)
                                                        6 = textbox7(pressure
gauge read)
                        allow 0 = off
                                        1 = on
                                        2 = on set
                */

//////////////////////////////////////////////// Argon panel
////////////////////////////////////////////////////////////

private: System::Void button18_Click(System::Object^  sender, System::EventArgs^  e)
{ // Ar valve on set
                        if(this->serialPort1->IsOpen == true)
                        {
                                this->button18->BackColor =
System::Drawing::Color::Lime;
                                this->button19->BackColor =
System::Drawing::Color::Red;
                                this->button20->BackColor =
System::Drawing::Color::Red;
                                Arallow =1;
                                sendAr();
                        }
                }
private: System::Void button19_Click(System::Object^  sender, System::EventArgs^  e)
{// Ar valve open
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                if(this->serialPort1->IsOpen == true)
                {
                        this->button18->BackColor =
System::Drawing::Color::Red;
                        this->button19->BackColor =
System::Drawing::Color::Lime;
                        this->button20->BackColor =
System::Drawing::Color::Red;
                        Arallow = 0;
                        this->serialPort1->Write("AF");
                }
        }
private: System::Void button20_Click(System::Object^  sender, System::EventArgs^  e)
{// Ar valve off
                if(this->serialPort1->IsOpen == true)
                {
                        this->button18->BackColor =
System::Drawing::Color::Red;
                        this->button19->BackColor =
System::Drawing::Color::Red;
                        this->button20->BackColor =
System::Drawing::Color::Lime;
                        Arallow = 0;
                        this->serialPort1->Write("AC");
                }
        }
private: System::Void button31_Click(System::Object^  sender, System::EventArgs^  e)
{
                String^ multi = this->button31->Text;
                if(multi=="0.1"){this->button31->Text = "1";}
                else if(multi=="1"){this->button31->Text = "10";}
                else if(multi=="10"){this->button31->Text = "0.1";}
        }
private: System::Void button30_Click(System::Object^  sender, System::EventArgs^  e)
{
                String^ before = this->textBox1->Text;
                String^ plus = this->button31->Text;
                double a = System::Convert::ToDouble(before);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                double b = System::Convert::ToDouble(plus);
                double c =a+b;
                if(c<100){
                        String^ com = String::Format("{0:F1}", c);
                        this->textBox1->Text = com;}
                sendAr();
        }
private: System::Void button29_Click(System::Object^  sender, System::EventArgs^  e)
{
                String^ before = this->textBox1->Text;
                String^ plus = this->button31->Text;
                double a = System::Convert::ToDouble(before);
                double b = System::Convert::ToDouble(plus);
                double c =a-b;
                if(c>=0){
                        String^ com = String::Format("{0:F1}", c);
                        this->textBox1->Text = com;}
                else if(c<0){
                        this->textBox1->Text = "0.0";}
                sendAr();
        }
private: void sendAr(){
                if(this->serialPort1->IsOpen == true & Arallow!=0)
                {String^ willsend;
                                                        if(this-
>comboBox1->Text == "Ar"){willsend = "A";}
                                                        else if(this-
>comboBox1->Text == "N2"){willsend = "N";}
                                                        else if(this-
>comboBox1->Text == "O2"){willsend = "O";}
                                                        else if(this-
>comboBox1->Text == "H2"){willsend = "H";}

                                                        this->serialPort1-
>Write("A");
                                                        this->serialPort1-
>Write(willsend);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                                                    this->serialPort1-
>Write(textBox1->Text);
                                                    this->serialPort1-
>Write("s");}
                }


////////////////////////////////////////////////// Nitrogen panel
//////////////////////////////////////////////////////////

private: System::Void button23_Click(System::Object^  sender, System::EventArgs^  e)
{ // Nitrofen ONSET
                        if(this->serialPort1->IsOpen == true)
                        {
                                this->button23->BackColor =
System::Drawing::Color::Lime;
                                this->button22->BackColor =
System::Drawing::Color::Red;
                                this->button21->BackColor =
System::Drawing::Color::Red;
                                Nallow =1;
                                sendN();
                        }
                }
private: System::Void button22_Click(System::Object^  sender, System::EventArgs^  e)
{// Nitrofen ON
                        if(this->serialPort1->IsOpen == true)
                        {
                                this->button23->BackColor =
System::Drawing::Color::Red;
                                this->button22->BackColor =
System::Drawing::Color::Lime;
                                this->button21->BackColor =
System::Drawing::Color::Red;
                                Nallow = 0;
                                this->serialPort1->Write("N");
                                this->serialPort1->Write("F");
                        }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        }
private: System::Void button21_Click(System::Object^  sender, System::EventArgs^  e)
{// Nitrofen OFF
                if(this->serialPort1->IsOpen == true)
                {
                        this->button23->BackColor =
System::Drawing::Color::Red;
                        this->button22->BackColor =
System::Drawing::Color::Red;
                        this->button21->BackColor =
System::Drawing::Color::Lime;
                        Nallow = 0;
                        this->serialPort1->Write("N");
                        this->serialPort1->Write("C");
                }
        }
private: System::Void button32_Click(System::Object^  sender, System::EventArgs^  e)
{
                String^ multi = this->button32->Text;
                if(multi=="0.1"){this->button32->Text = "1";}
                else if(multi=="1"){this->button32->Text = "10";}
                else if(multi=="10"){this->button32->Text = "0.1";}
        }
private: System::Void button33_Click(System::Object^  sender, System::EventArgs^  e)
{
                String^ before = this->textBox3->Text;
                String^ plus = this->button32->Text;
                double a = System::Convert::ToDouble(before);
                double b = System::Convert::ToDouble(plus);
                double c =a+b;
                if(c<50){
                        String^ com = String::Format("{0:F1}", c);
                        this->textBox3->Text = com;}
                sendN();
        }
private: System::Void button34_Click(System::Object^  sender, System::EventArgs^  e)
{
                String^ before = this->textBox3->Text;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                String^ plus = this->button32->Text;
                double a = System::Convert::ToDouble(before);
                double b = System::Convert::ToDouble(plus);
                double c = (a-b);
                if(c>=0){
                        String^ com = String::Format("{0:F1}", c);
                        this->textBox3->Text = com;}
                else if(c<0){
                        this->textBox3->Text = "0.0";}
                sendN();
        }

private: void sendN(){
                if(this->serialPort1->IsOpen == true & Nallow !=0){
                                String^ willsend;
                                if(this->comboBox2->Text ==
"Ar"){willsend = "A";}
                                else if(this->comboBox2->Text
== "N2"){willsend = "N";}
                                else if(this->comboBox2->Text
== "O2"){willsend = "O";}
                                else if(this->comboBox2->Text
== "H2"){willsend = "H";}

                                this->serialPort1->Write("N");
                                this->serialPort1->Write(willsend);
                                this->serialPort1->Write(textBox3-
>Text);
                                this->serialPort1->Write("s");}
        }

private: System::Void serialPort1_DataReceived(System::Object^  sender,
System::IO::Ports::SerialDataReceivedEventArgs^  e)
        {


        }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
////////////////////////////////////////////////// RF panel
//////////////////////////////////////////////////////////////////

private: System::Void trackBar1_Scroll(System::Object^  sender, System::EventArgs^
e) {

                Radio_power();
        }

private: System::Void trackBar2_Scroll(System::Object^  sender, System::EventArgs^
e) {

                Radio_power();
        }

private: System::Void trackBar3_Scroll(System::Object^  sender, System::EventArgs^
e) {

                Radio_power();
        }

private: void Radio_power()
        {
                int radio1 = (this->trackBar3->Value);
                int radio2 = (this->trackBar2->Value)*10;
                int radio3 = (this->trackBar1->Value)*100;

                int radiopower = radio1 +radio2 +radio3;
                String^ radioset = System::Convert::ToString(radiopower);
                this->textBox8->Text = radioset;

                if( RFallow ==1){
                        if(this->serialPort1->IsOpen == true){
                                this->serialPort1->Write("R");
                                this->serialPort1->Write("A");
                                this->serialPort1->Write(textBox8->Text);
                                this->serialPort1->Write("s");}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                                                    }
                }
private: System::Void label36_Click(System::Object^  sender, System::EventArgs^  e)
{// Adjust or stop
                        if(RFallow ==1){
                                RFallow =0;
                                this->label36->ForeColor =
System::Drawing::Color::Black;
                                if(this->serialPort1->IsOpen == true){this->serialPort1-
>Write("R");this->serialPort1->Write("S");}
                                }
                        else if(RFallow ==0){
                                if(this->serialPort1->IsOpen == true){
                                RFallow =1;
                                this->label36->ForeColor =
System::Drawing::Color::Orange;
                                this->serialPort1->Write("R");
                                this->serialPort1->Write("A");
                                this->serialPort1->Write(TXT[4]->Text);
                                this->serialPort1->Write("s");}
                        }}
private: System::Void button25_Click(System::Object^  sender, System::EventArgs^  e)
{// RF AC
                        if(RF_AC  == false){
                        if(this->serialPort1->IsOpen == true){ this->serialPort1-
>Write("Ra");}
                        this->button25->BackColor = System::Drawing::Color::Red;
                        RF_AC = true;}
                        else if(RF_AC  == true){
                        if(this->serialPort1->IsOpen == true){ this->serialPort1-
>Write("Rb");}
                        this->button25->BackColor = System::Drawing::Color::Black;
                        RF_AC = false;}
                }
private: System::Void button27_Click(System::Object^  sender, System::EventArgs^  e)
{/// RF ON
                        if(RF_ON  == false){
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    if(this->serialPort1->IsOpen == true){ this->serialPort1-
>Write("RO");}
                    this->button27->BackColor = System::Drawing::Color::Red;
                    RF_ON = true;}
                    else if(RF_ON  == true){
                    if(this->serialPort1->IsOpen == true){ this->serialPort1-
>Write("RC");

                RFallow =0;

                this->label36->ForeColor = System::Drawing::Color::Black;

                this->TXT[4]->Text = "0";

                this->trackBar1->Value = 0;

                this->trackBar2->Value = 0;

                this->trackBar3->Value = 0;
            }
                    this->button27->BackColor = System::Drawing::Color::Black;
                    RF_ON = false;}
                }

//////////////////////////////////////////// Pump and Cooler Panel
////////////////////////////////////////////////////
private: System::Void button28_Click(System::Object^  sender, System::EventArgs^  e)
{ // Rotary
                    if(rotary  == false){
                    if(this->serialPort1->IsOpen == true){ this->serialPort1-
>Write("fRO");}
                    this->button28->BackColor = System::Drawing::Color::Green;
                    rotary = true;}
                    else if(rotary  == true){
                    if(this->serialPort1->IsOpen == true){ this->serialPort1-
>Write("fRC");}
                    this->button28->BackColor = System::Drawing::Color::Black;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    rotary = false;}

            }
private: System::Void button26_Click(System::Object^  sender, System::EventArgs^  e)
{ // Heater
                    if(heater  == false){
                    if(this->serialPort1->IsOpen == true){ this->serialPort1-
>Write("fHO");}
                    this->button26->BackColor =
System::Drawing::Color::OrangeRed;
                    heater = true;}
                    else if(heater  == true){
                    if(this->serialPort1->IsOpen == true){ this->serialPort1-
>Write("fHC");}
                    this->button26->BackColor = System::Drawing::Color::Black;
                    heater = false;}
            }

//////////////////////////////////// Sound
////////////////////////////////////////////////////////////////////////////////

private: System::Void toolStripMenuItem3_Click(System::Object^  sender,
System::EventArgs^  e) {

                    if(soundarrow == 1){soundarrow = 0; this-
>toolStripMenuItem3->Image = toolStripMenuItem3->BackgroundImage ;}
                    else {soundarrow = 1; this->toolStripMenuItem3->Image =
sound ;}
            }

////////////////////////////////////  Exit
////////////////////////////////////////////////////////////////////////

private: System::Void exitToolStripMenuItem_Click(System::Object^  sender,
System::EventArgs^  e) {
                        Form ^parts = gcnew CloseForm();
        parts->Show();
```

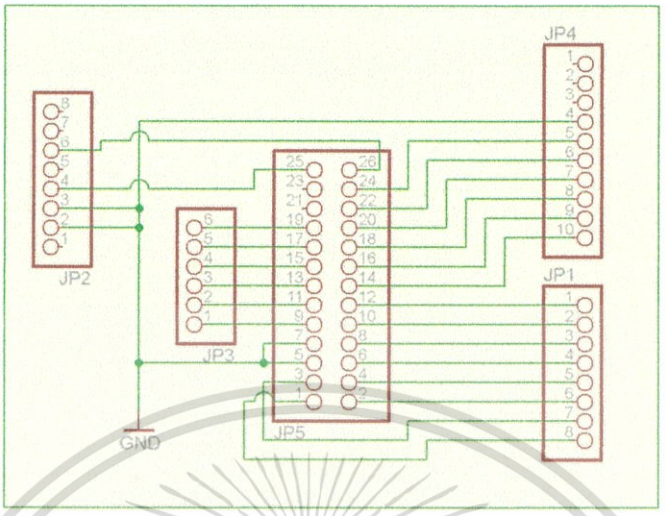
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

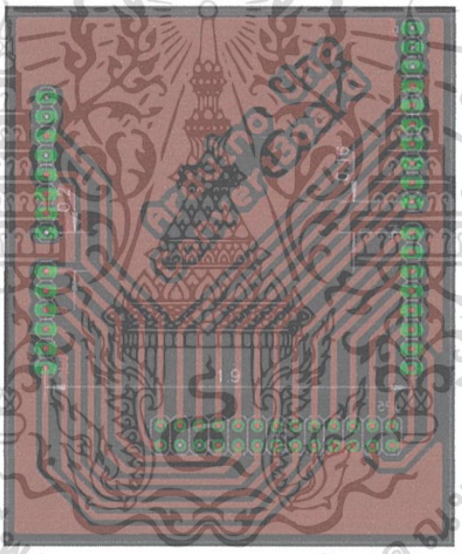
```
        }

}; //// End of public ref class Form1
} ////////////////////////////// End of namespace Magnetronsputtering
//////////////////////////////////////////
```

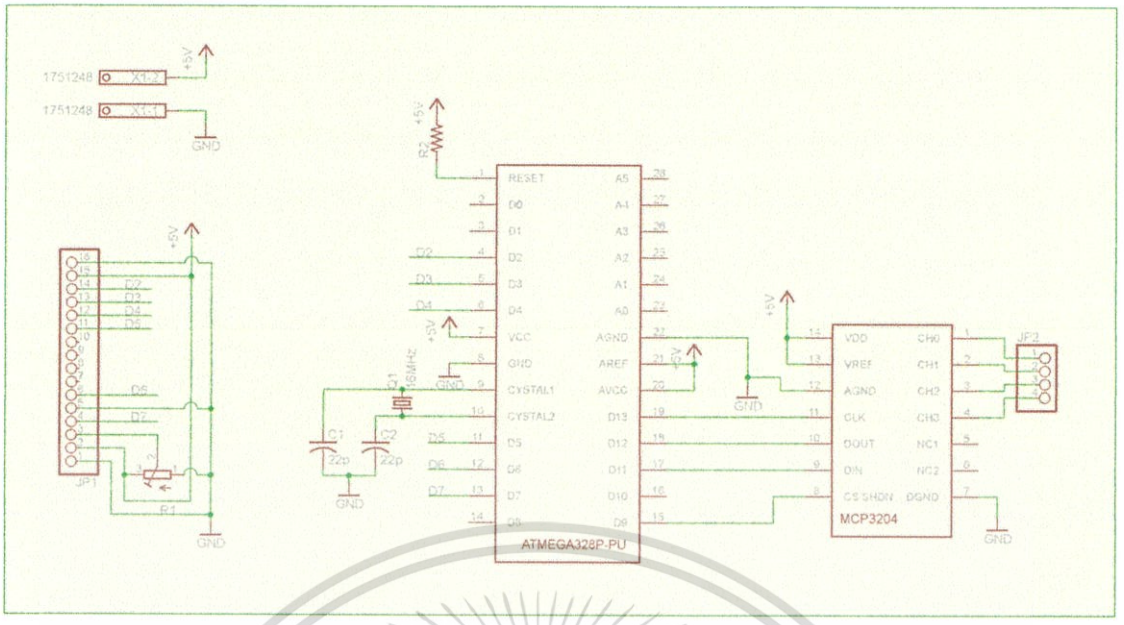

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ค

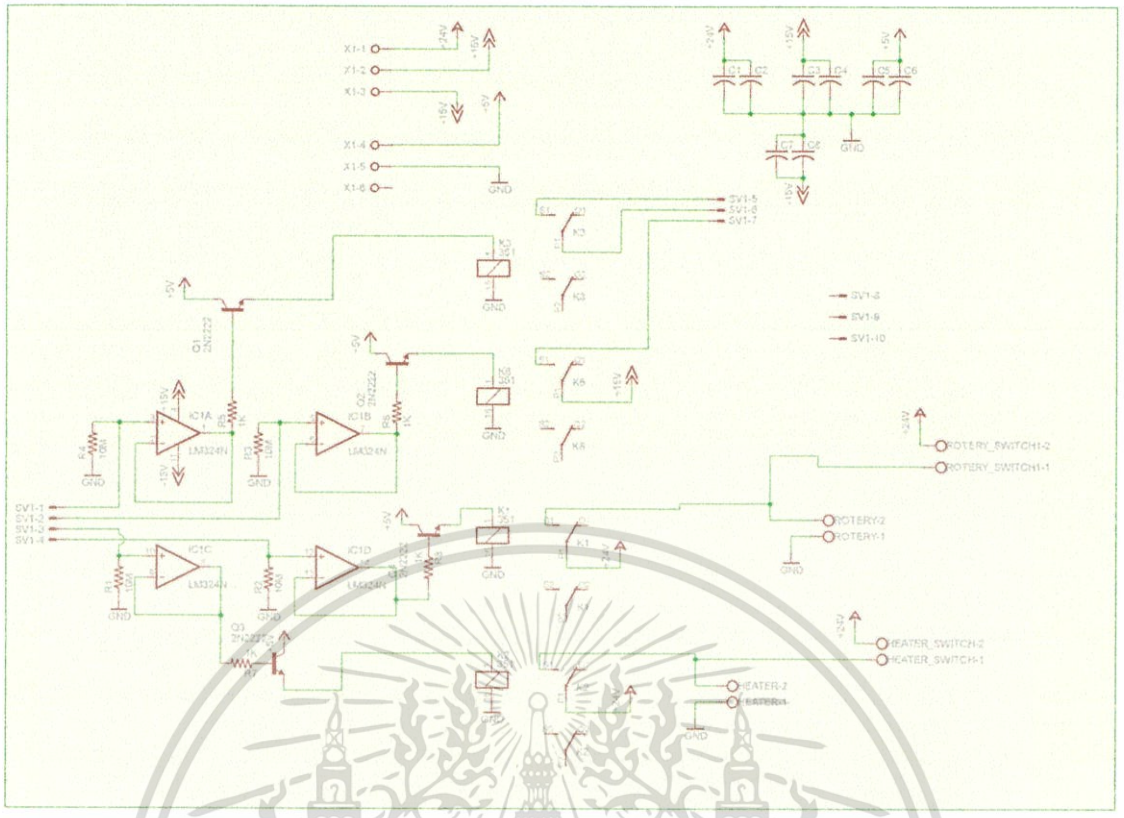


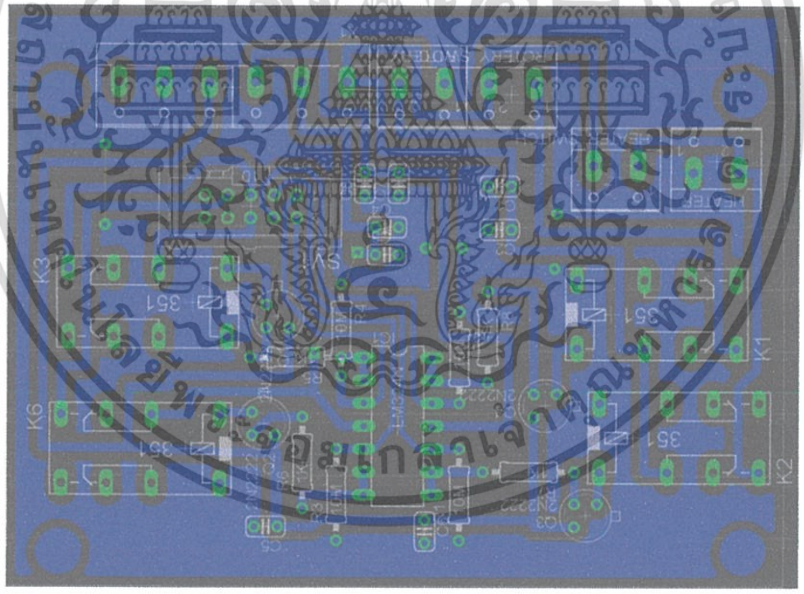


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

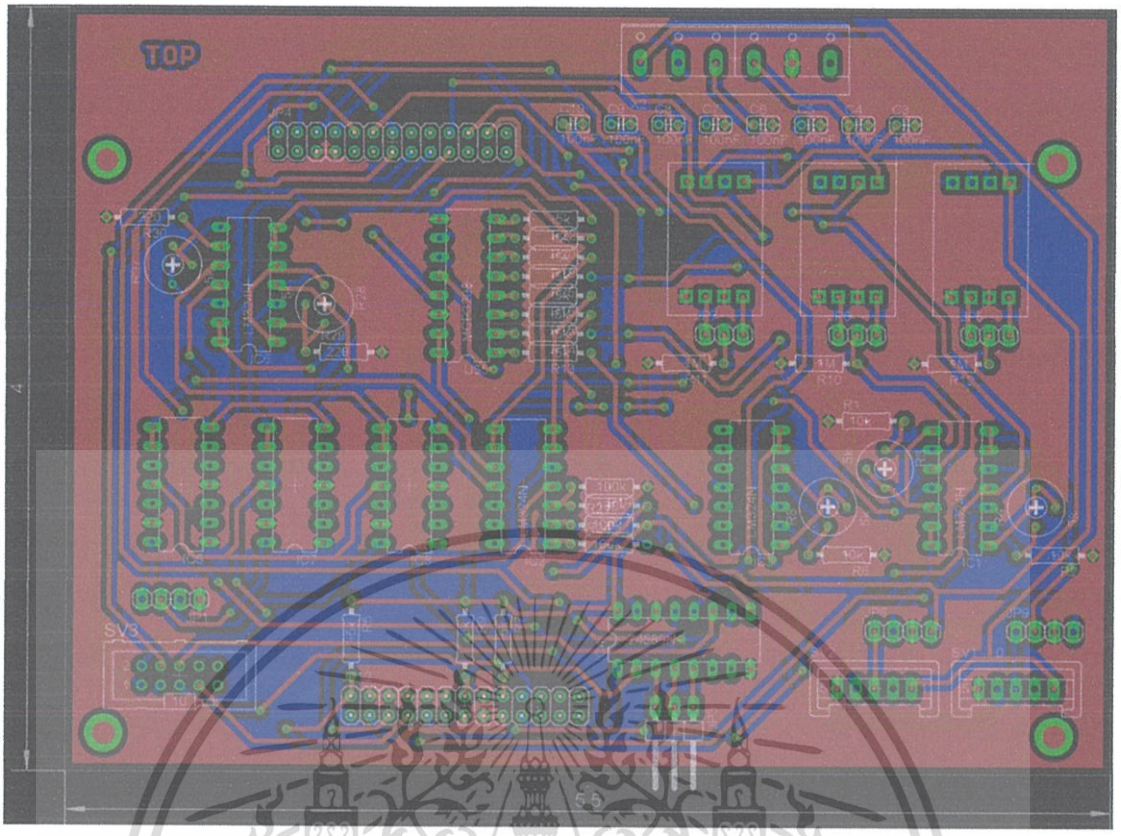


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้





เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้