ระบบเฝ้าระวังแปลงผักไฮโดรโพนิกส์ด้วยเซ็นเซอร์ไร้สายผ่านระบบ GSM

โดย

นางสาวปิยะดา บำรุงเขตต์	รหัสนักศึกษา	B5112418
นายคมสัน แสงอรุณ	รหัสนักศึกษา	B5114238
นางสาวชลกานต์ มาตรมณีวงศ์	รหัสนักศึกษา	B5114627

รายงานนี้เป็นส่วนหนึ่งของการศึกษาวิชา 427499 โครงงานวิศวกรรมโทรคมนาคม และ
ประจำภาคการศึกษาที่ 1 ปีการศึกษา 2554
หลักสูตรวิศวกรรมศาสตร์บัณฑิต สาขาวิชาวิศวกรรมโทรคมนาคม หลักสูตรปรับปรุง พ.ศ. 2546
สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี

ระบบเฝ้าระวังแปลงผักโฮโดรโพนิกส์ด้วยเซ็นเซอร์ไร้สายและ GSM

คณะกรรมการสอบโครงงาน

(ผู้ช่วยศาสตราจารย์ ดร. วิภาวี หัตถกรรม)
กรรมการ/อาจารย์ที่ปรึกษาโครงงาน

(ผู้ช่วยศาสตราจารย์ ดร.พีระพงษ์ อุฑารสกุล)
กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร. ชุติมา พรหมมาก)
กรรมการ

มหาวิทยาลัยเทคโนโลยีสุรนารี อนุมัติให้นับรายงานโครงงานฉบับนี้ เป็นส่วนหนึ่งของการศึกษาระดับปริญญาตรี สาขาวิชาวิศวกรรมโทรคมนาคม รายวิชา 427499 โครงงานวิศวกรรมโทรคมนาคม ประจำปีการศึกษา 2554

โครงงาน	ระบบเฝ้าระวังแปลงผักไฮโดรโพนิกส์ด้วยเซ็นเซอร์ไร้สายผ่านระบบ GSM
จัดทำโดย	นางสาวปิยะดา บำรุงเขตต์
	นายคมสัน แสงอรุณ
	นางสาวชลกานต์ มาตรมณีวงศ์
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์ ดร.วิภาวี หัตถกรรม
สาขาวิชา	วิศวกรรมโทรคมนาคม
ภาคการศึกษาที่	1/ 2554

บทคัดย่อ

(Abstract)

ปัจจุบันเทคโนโลยีเครือข่ายเซ็นเซอร์ไร้สาย (Wireless Sensor Network หรือ WSN) ได้เข้ามามีบทบาทในการดำรงชีวิต และมีแนวโน้มนำเทคโนโลยีมาประยุกต์ใช้ทางด้านเกษตรกรรม ไม่ว่าจะเป็นการทำไร่ ทำสวนหรือเพาะเลี้ยงสัตว์น้ำ เพื่อช่วยเพิ่มประสิทธิภาพผลผลิตให้มีคุณภาพมากขึ้น ดังนั้นจึงได้นำเสนอโครงงานเฝ้าระวังแปลงผักไฮโดรโพนิกส์ด้วยเซ็นเซอร์ไร้สายผ่านระบบ GSM เพื่อช่วยตรวจสอบค่าพารามิเตอร์ต่างๆที่เกี่ยวข้องกับการเจริญเติบโตของผัก ซึ่งจะทำให้ประหยัดแรงงานและเวลาในการตรวจสอบ ซึ่งระบบเซ็นเซอร์ทำการวัดค่า ความนำไฟฟ้าในสารละลาย (eC) ค่าความเป็นกรด-ด่าง (pH) และค่าความเข้มแสง (Light) ซึ่งข้อมูลทั้งหมดถูกส่ งผ่านเซ็นเซอร์ไร้สายมายัง สถานีฐานและ โปรแกรมจะทำการ ประมวลผลเพื่อตรวจสอบค่าพารามิเตอร์ต่างๆ เมื่อพบว่ามีค่าพารามิเตอร์ที่ไม่เหมาะสมกับการเจริญเติบโตของผักในแปลงผักไฮโดรโพนิกส์แล้วโปรแกรมก็จะทำการสร้างข้อความและทำการส่ง SMS ไปยังโทรศัพท์มือถือของผู้ใช้งานเพื่อแจ้งเตือน โดยจะทำการส่งผ่านทาง GSM Module

กิตติกรรมประกาศ

(Acknowledgement)

จากการจัดทำโครงงานเรื่องระบบเฝ้าระวังแปลงผักไฮโดรโพนิกส์ด้วยเซ็นเซอร์ไร้สายผ่านระบบGSM ส่งผลให้คณะผู้จัดทำได้รับความรู้และประสบการณ์ต่างๆ มากมาย โครงงานฉบับนี้สามารถสำเร็จลุล่วงไปได้ด้วยดี เนื่องจากได้รับความกรุณาจากอาจารย์ที่ปรึกษาโครงงาน ผู้ช่วยศาสตราจารย์ ดร.วิภาวี หัตถกรรม ที่ได้ให้ความช่วยเหลือ ให้คำปรึกษาแนะนำในทุกๆ ด้าน ตลอดจนชี้แนะข้อบกพร่อง ต่างๆ รวมถึงการให้แนวคิด การดูแลเอาใจใส่ติดตามงาน ให้แก่คณะผู้จัดทำมาโดยตลอด และขอขอบพระคุณคณาจารย์ บุคลากร และพี่นักศึกษาบัณฑิตศึกษาสาขาวิชาวิศวกรรมโทรคมนาคมทุกท่านที่คอยแนะนำและให้ความช่วยเหลือ

คณะผู้จัดทำใคร่ขอขอบพระคุณทุกๆ ท่านที่ได้กล่าวไปแล้วไว้ ณ ที่นี้ ซึ่งเป็นผู้ให้โอกาสทางการศึกษาและคอยสนับสนุน สำหรับส่วนดีของโครงงานชิ้นนี้ ขออุทิศให้แก่อาจารย์ทุกท่านที่ได้ประสิทธิ์ประสาทวิชาความรู้ให้แก่คณะผู้จัดทำ

นางสาวปิยะดา บำรุงเขตต์
นายคมสัน แสงอรุณ
นางสาวชลกานต์ มาตรมณีวงศ์

สารบัญ

สารบัญ(ต่อ)

สารบัญ(ต่อ)

สารบัญรูป

รายการ **หน้า**

สารบัญรูป(ต่อ)

สารบัญตาราง

รายการ **หน้า**

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของโครงงาน

ผักไฮโดรโพนิกส์เป็นนวัตกรรมการปลูกผักแบบไร้ดินซึ่งกำลังได้รับความนิยมอย่างมาก เพราะเป็นผักที่ปลอดภัยไม่มีสารพิษ เนื่องจากการปลูกผักในระบบดังกล่าวจะถูกควบคุมโ ดยระบบน้ำ ดังนั้นผักที่ปลูกในระบบไฮโดรโพนิกส์จะไม่ได้รับเชื้อโรคและสิ่งปนเปื้อนที่มาจากดิน ซึ่งผู้ปลูกไม่จำเป็นต้องใช้สารเคมีในการป้องกันโรคและส่วนใหญ่ทุกฟาร์มจะปลูกผักอยู่ในระบบปิด ซึ่งช่วยควบคุมศัตรูพืชได้ง่าย ทำให้ไม่ต้องฉีดยาพ่นฆ่าแมลงเนื่องจากการปลูกพืชแบบไฮโดร - โพนิกส์มีการจัดปัจจัยต่างๆ เช่น น้ำ แร่ธาตุ แสงอุณหภูมิ ค่าความเป็นกรดด่าง (pH) ค่าความเข้มของสารละลาย (eC) ให้แก่พืชอย่างเหมาะสม พืชจึงเจริญเติบโตเร็วและให้ผลผลิตมากสม่ำเสมอ สะอาด มีคุณภาพดี ปลูกได้ต่อเนื่องตลอด ทั้งปี สามารถปลูกพืชได้ในพื้นที่ไม่มีดินหรือมีดิน ที่ไม่เหมาะสมต่อการปลูกพื ช ประหยัดพื้นที่ในการปลูก การใช้น้ำและปุ๋ยเป็นไปอย่างมีประสิทธิภาพ ใช้แรงงานน้อย การควบคุมโรคและแมลงศัตรูพืชทำได้ง่ายกว่า ฉะนั้นโครงงานระบบ เฝ้าระวังแปลงผักไฮโดโพนิกส์ด้วยเซ็นเซอร์ ไร้สายผ่านระบบ GSM จึงได้เห็นว่าจะใช้ระบบการส่ง SMS เข้ามาช่วยในการแก้ปัญหาเนื่องจากว่าผู้ประกอบการ ก็มีโทรศัพท์มือถือเป็นอุปกรณ์ติดต่อสื่อสารอยู่แล้ว เราจึงทำให้โครงงานเฝ้าระวังแปลงผัก ไฮโดโพนิกส์ด้วยเซ็นเซอร์ ไร้สายผ่านระบบ GSM นี้ ส่ง SMS เข้าไปยังโทรศัพท์มือถือของเกษตรกร เมื่อเซ็นเซอร์ทำการวัดค่าก็จะส่งข้อมูลมาเก็บไว้ที่สถานีฐาน แล้วทำการส่ง SMS ตามข้อกำหนด เช่น ให้ส่งเมื่อค่าพารามิเตอร์ เมื่อ มีค่าพารามิเตอร์ ผิดปกติ เพื่อให้ผู้ประกอบ การสามารถตรวจสอบข้อมูลได้สะดวก ขึ้น ซึ่งจากการแก้ปัญหาโดยใช้การส่ง SMS จะเห็น ได้ว่าผู้ประกอบการสามารถรับรู้ข้อมูลได้แม้จะอยู่สถานที่ไหนก็ตาม ซึ่งโครงงานนี้ได้ใช้ MODULE GSM CPU เป็นตัวรับคำสั่งและทำการส่ง SMS โดยการส่ง SMS นี้ได้ใช้ระบบ Dtac GSM ในการส่งเป็นหลักเนื่องจากเป็นการประหยัดต้นทุน เพราะค่าใช้จ่ายในการรับส่ง SMS ค่อนข้างถูกและระบบการส่ง SMS ก็เป็นที่นิยมแพร่หลายในปัจจุบัน

1.2 วัตถุประสงค์ของโครงงาน

1. เพื่อเป็นการศึกษาในการใช้โปรแกรม Microsoft Visual C# 2008
2. เพื่อนำเอาระบบการส่ง SMS ไปช่วยในการแจ้งค่าความเข้มแสง (Light) ค่าความเป็นกรด-ด่าง (pH) และค่าความนำไฟฟ้าในสารละลาย (eC) ในแปลงผักไฮโดรโพนิกส์แทนการจ้างแรงงาน
3. เพื่อติดตั้งและทดสอบระบบเซ็นเซอร์ ให้สามารถตรวจจับค่าความเข้มแสง (Light) ค่าความเป็นกรด-ด่าง (pH) และค่าความนำไฟฟ้าในสารละลาย (eC) ได้

1.3 ขอบเขตการดำเนินงาน

1. ศึกษาการทำงานของโปรแกรม Microsoft Visual C# 2008
2. ศึกษาการทำงานของ Module Wireless CPU รุ่น Q26 (GSM)
3. เขียนโปแกรมควบคุมการทำงานของ Module Wireless CPU รุ่น Q26 (GSM) เพื่อส่ง SMS โดยใช้ Microsoft Visual C# 2008
4. จัดเตรียมอุปกรณ์สำหรับติดตั้ง
5. ทดสอบอุปกรณ์ทั้งระบบให้ได้ตามวัตถุประสงค์
6. นำไปติดตั้ง ณ สถานที่จริง

1.4 ขั้นตอนการดำเนินงาน

1. ศึกษาโปรแกรม Microsoft Visual C# 2008
2. ศึกษาการทำงานของ Module Wireless CPU รุ่น Q26 (GSM)
3. เขียนโปรแกรมให้สามารถส่ง SMS ได้ตามเงื่อนไขที่กำหนด
4. ทดสอบการทำงานโปรแกรมส่ง SMS
5. จัดทำอุปกรณ์ทั้งหมด
6. ทดสอบการรับส่งข้อมูลระหว่างเซ็นเซอร์กับสถานีฐาน
7. นำอุปกรณ์ทั้งหมดไปติดตั้งและทดสอบเพื่อให้ได้ตามวัตถุประสงค์
8. สรุปผลการทดลอง เขียนรายงาน และนำเสนอผลงาน

บทที่ 2
ทฤษฎีพื้นฐานที่เกี่ยวข้อง

2.1 บทนำทฤษฎี

ในบทนี้จะกล่าวถึงทฤษฎีพื้นฐานที่เกี่ยวข้องกับโครงงานระบบเฝ้าระวังแปลงผักไฮโดโพ - นิกส์ด้วยเซ็นเซอร์ไร้สายผ่านระบบ GSM ซึ่งเนื้อหาจะเกี่ยวกับระบบ GSM รวมไปถึงเครือข่ายเซ็นเซอร์ไร้สาย และพื้นฐานการใช้งานเซ็นเซอร์ ค่าความนำไฟฟ้าในสารละลาย (eC) ความเป็นกรด-ด่าง (pH) ค่าความเข้มแสง (Light)

2.2 เครือข่ายเซ็นเซอร์ไร้สาย (Wireless Sensor Networks)

ระบบเครือข่ายเซ็นเซอร์แบบไร้สาย (Wireless Sensor Networks หรือ WSNs) คือการใช้อุปกรณ์เซ็นเซอร์เล็กๆ จำนวนมากเพื่อตรวจวัดคุณสมบัติต่างๆ ของสิ่งแวดล้อมที่เราสนใจและประมวลผลข้อมูลเหล่านั้นเพื่อสร้างองค์ความรู้ใหม่เกี่ยวกับสิ่งแวดล้อมรอบตัวเราหรือตอบสนองกับการเปลี่ยนแปลงของสภาพแวดล้อมได้โดยอัตโนมัติ ซึ่งอุปกรณ์พื้นฐานของ WSN คือ Mote เป็นคอมพิวเตอร์ขนาดเล็กสำหรับวัดอุณหภูมิความชื้นหรือสภาวะแวดล้อมอื่นๆ ทำงานได้โดยใช้แบตเตอรี่ธรรมดาและสื่อสารกับ Mote ที่อยู่ใกล้เคียงโดยใช้ลักษณะการส่งข้อมูลระบบในเครือข่ายไร้สาย ซึ่งข้อมูลจะถูกส่งผ่านระหว่าง Mote จนกระทั่งถึงจุดหมายปลายทางซึ่งอาจจะเป็นคอมพิวเตอร์หรืออุปกรณ์อื่นๆ สำหรับรวบรวมข้อมูลเพื่อวัดได้

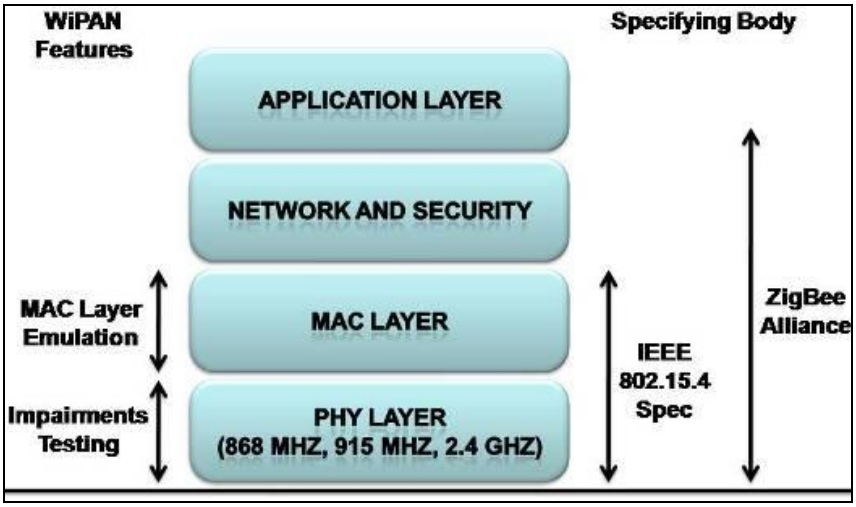
ตัวอย่างการใช้งานของเครือข่ายตรวจรู้ไร้สายนี้ ได้แก่การฝังอุปกรณ์เซ็นเซอร์ไว้ในรังนกหายากบางชนิดเพื่อตรวจจับการเปลี่ยนแปลงอุณหภูมิที่มีผลต่อการย้ายถิ่นฐานของนกเหล่านั้น การติดตั้งเซ็นเซอร์ไว้ในอุปกรณ์ผสมสารเคมีขนาดใหญ่หรือท่อส่งเคมีในโรงงานอุตสาหกรรมเพื่อตรวจจับการรั่วซึมของสารเคมี การใช้เซนเซอร์ตรวจวัดการสั่นไหวของอุปกรณ์หลายๆ อย่างที่ใช้สำหรับการสร้างคอมพิวเตอร์ชิพ เพื่อตรวจจับความผิดปกติของเครื่องมือเหล่านั้นเพื่อ ให้สามารถเข้าไปดูแลได้ก่อนที่มันจะเสียหาย การติดตั้งเซ็นเซอร์ไว้รอบ ๆ สนามบินเพื่อตรวจจับการบุกรุกในบริเวณที่ห้ามคนเข้า เป็นต้น จากตัวอย่างเหล่านี้จะเห็นได้ว่านักวิทยาศาสตร์และวิศวกรสามารถใช้ประโยชน์จากเครือข่ายตรวจรู้ไร้สายได้มากมายหลายรูปแบบ นอกจากนี้การออกแบบเซ็นเซอร์โหนดให้มีขนาดที่เล็กและใช้พลังงานน้อยทำให้สามารถติดตั้งได้ในสภาพแวดล้อมที่หลากหลาย เทคโนโลยีเครือข่ายเซ็นเซอร์จึงได้ถูกคาดการณ์ว่าจะเป็นเทคโนโลยีหลักในการขับดันสู่ยุคของคอมพิวเตอร์ทุกแห่งหน (Ubiquitous Computing, Pervasive Computing) ด้วยการสร้างสภาพแวดล้อมประดิษฐ์ในรอบๆตัวของเราทุกคน

อย่างไรก็ตาม การทำให้ WSN ทำงานได้จริงอย่างมีประสิทธิภาพนั้น มีอุปสรรคหลายอย่าง เนื่องจาก Mote มีขีดจำกัดในความเร็วของ Processor, ความจุของ Data Storage, และ Bandwidth ในการสื่อสาร ดังนั้นอายุการใช้งานของมันจึงขึ้นอยู่กับการบริหารการใช้พลังงานของมันเองซึ่งมีอยู่จำกัดด้วยเหตุนี้ผู้พัฒนา Mote จึงต้องออกแบบระบบ Hardware และ Software รวมถึงระบบการสื่อสารของ Mote ให้ทำงานโดยใช้พลังงานน้อยที่สุด นอกจากนี้ในแง่ของการใช้งานผู้พัฒนา WSN ต้องสร้างเครื่องมือที่ทำผู้ใช้ซึ่งไม่จำเป็นจะต้องมีความรู้ขั้นสูงทางด้าน Computer Engineering สามารถใช้งานและสร้าง WSN Applications โดยง่ายด้วย

2.2.1 ZigBee

ZigBee เป็นการสื่อสารที่ออกแบบขึ้นสำหรับการสื่อสารในเครือข่ายเซ็นเซอร์แบบไร้สาย (Wireless Sensor Network) โดยเริ่มจากการกำหนดมาตรฐานการรับ-ส่งข้อมูลแบบ IEEE 802.15.4 ที่เน้นการสื่อสารแบบประหยัดพลังงาน ความเร็วการรับส่งข้อมูลต่ำและมีราคาถูก การสื่อสารลักษณะนี้ได้ถูกนำมาใช้สำหรับการสื่อสาร ระหว่างเครื่องตรวจวัดหรือเซ็นเซอร์ ที่ต้องการสื่อสารแบบไร้สาย เพื่อลดความยุ่งยากซับซ้อนสำหรับการติดตั้ง เช่น บริเวณโรงงานหนึ่งๆอาจจะต้องใช้จำนวนเซ็นเซอร์ปริมาณมากๆ และเครื่องรับส่งที่มีราคาถูก และประหยัดพลังงาน

มาตรฐาน IEEE 802.15.4 กำหนดขึ้นสำหรับการรับส่งข้อมูลเบื้องต้นในวงจรเครื่องรับส่งวิทยุ (Physical Layer) และการควบคุมการรับส่ง (Link Layer) ดังต่อไปนี้ การสื่อสารใช้คลื่นวิทยุความถี่ 2.5 กิกะเฮิรตซ์ (GHz) แบ่งออกเป็น 16 ช่องสัญญาณๆ ละ 5 เมกะเฮิรตซ์ (MHz) สำหรับความถี่ 900 เมกะเฮิรตซ์ (MHz) แบ่งออกเป็น 10 ช่องสัญญาณๆ ละ 2 เมกะเฮิรตซ์ (MHz) ใช้การผสมสัญญาณ (Modulation) แบบ Offset Quadrature Phase Shift Keying (Offset-QPSK) และใช้การแก้ปัญหาสัญญาณรบกวนแบบ Direct Sequence Spread Spectrum (DSSS) ที่มีอัตราการสเปรดดิ้ง (Spreading) 2 ล้าน chip/sec ควบคุมการรับส่งข้อมูลโดยใช้โปรโตคอลแบบ Carrier Sense Multiple Access / Collision Avoidance (CSMA/CA) และเพื่อให้การสื่อสารเครือข่ายเซ็นเซอร์ไร้สายเป็นมาตรฐานเดียวกัน จึงกำหนดมาตรฐานเพิ่มสำหรับการเชื่อมต่อเป็นเครือข่าย (Network Layer) และการนำไปใช้งาน (Application Layer) ร่วมกับมาตรฐาน IEEE 802.15.4 เป็นมาตรฐานใหม่ที่กำหนดโดยองค์กร ZigBee Alliance



รูปที่ 2.1 แสดง Protocol Zigbee

Zigbee ได้แบ่งตามลักษณะการทำงาน 3 แบบ คือ

1. Coordinator มีหน้าที่สร้างการสื่อสารเชื่อมโยงเครือข่ายระหว่าง End-device กับ Router หรือ Coordinator กับ Coordinator ด้วยกัน หรือ Coordinator กับ Router กำหนด Address ให้กับ Device ที่อยู่ในวงเครือข่าย ไม่ให้ซ้ำกัน ดูแลจัดการเรื่องการ Routing เส้นทาง ซึ่งเทียบได้กับ FFD

2. End-device เป็นอุปกรณ์ปลายทางสุด ซึ่งจะใช้รับสัญญาณจาก Sensor ที่ปลายทาง โดยที่ใช้พลังงานต่ำในการทำงานเทียบได้กับ RFD หรือ FFD บางกรณี ขึ้นอยู่กับ Sensor ที่ใช้

3. Router มีหน้าที่รับส่งข้อมูลในเส้นทางต่าง ๆของเครือข่าย ซึ่งเทียบได้กับ FFD

2.2.2 ขั้นตอนการทำงานของโพรโทคอล ZigBee

1.) ขั้นตอนการทำงานของ ZigBee Coordinator

ZigBee Coordinator จะเริ่มต้นเครือข่าย โดยการตรวจสอบการใช้ช่องสัญญาณวิทยุภายในบริเวณรอบๆ ถ้ามีช่องสัญญาณที่ไม่ถูกใช้โดย Coordinator ตัวอื่น ก็สามารถเริ่มต้นเครือข่ายได้ หลังจากนั้น Coordinator ก็จะทำหน้าที่เป็นศูนย์กลางของเครือข่าย รองรับการเข้าร่วมเครือข่ายของ ZigBee End-device และรองรับการร้องขออื่นๆ ตามมาตรฐานด้วยเช่นกัน

2.) ขั้นตอนการทำงานของ ZigBee End-device

ZigBee End-device จะเริ่มต้นการทำงานโดยการร้องขอการเข้าร่วมเครือข่ายไปยัง Coordinator ประจำเครือข่ายนั้นๆ โดยการตรวจสอบผ่านช่องสัญญาณต่างๆ ว่า Coordinator ใช้ช่องสัญญาณใดอยู่เมื่อเข้าร่วมเครือข่ายแล้ว End-device จึงสามารถทำการร้องขอคำสั่งอื่นๆ ผ่านทาง Coordinator ได้ เช่น การส่งข้อความทั่วไป (Message), การร้องขอการ Binding (Binding Request), การขอออกจากเครือข่าย

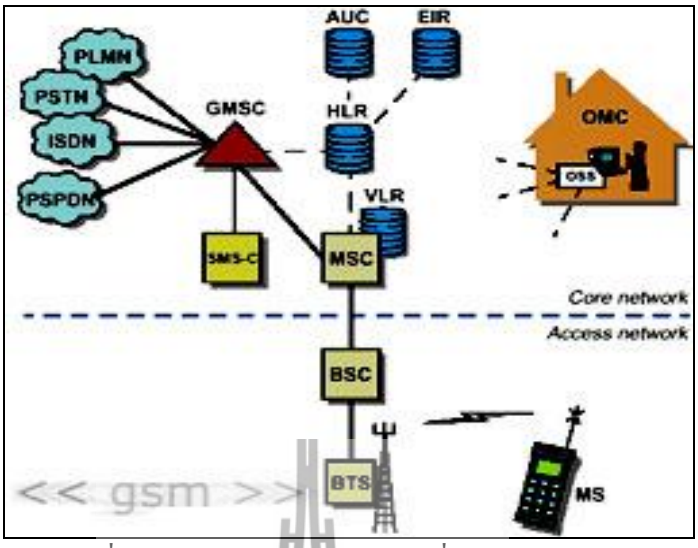
2.3 ระบบโทรศัพท์เคลื่อนที่ GSM และบริการข่าวสารสั้น

2.3.1 ระบบเคลื่อนที่ GSM

ในปัจจุบันนี้ระบบโทรศัพท์เคลื่อนที่ที่กำลังได้รับความนิยมเป็นอย่างมากระบบหนึ่งก็คือระบบ GSM (Global System for Mobile Communication) ซึ่งเป็นระบบโทรศัพท์แบบดิจิทัล ที่มีประสิทธิภาพสูงกว่าระบบแอนาลอกที่ใช้อยู่เดิมหลายด้านได้แก่ ประสิทธิภาพในการใช้สเปกตรัม โดยสามารถรองรับจำนวนผู้ใช้ได้มากกว่า สามารถทนต่อสัญญาณรบกวนได้ดีกว่า มีความปลอดภัยสูง และยังใช้กำลังในการส่งสัญญาณน้อยกว่าอีกด้วย ระบบโทรศัพท์เคลื่อนที่ GSM ประกอบด้วยระบบย่อยๆ 4 ระบบดังนี้

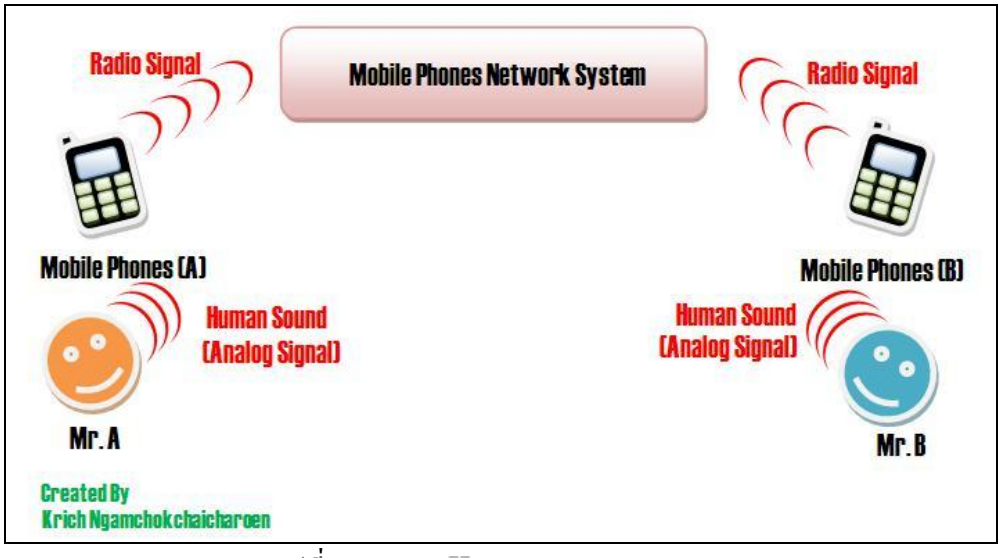
1. ระบบย่อยสถานีฐาน (Base Station Sub System : BSS) ประกอบด้วย
   - สถานีฐานรับ-ส่งสัญญาณ (Base Transceiver Station: BTS)
   - ตัวควบคุมสถานีฐาน (Base Station Controller: BSC)
2. ระบบย่อยสวิตชิง (Switching Sub System: SSS) ประกอบด้วย
   - ชุมสายโทรศัพท์เคลื่อนที่ (Mobile Switching Center: MSC)
   - ฐานข้อมูลทะเบียนผู้ใช้ (Home Location Register: HLR)
   - ฐานข้อมูลผู้มาเยือน (Visitor Location Register: VLR)
   - ฐานข้อมูลตรวจสอบความถูกต้องของผู้ใช้ (Authentication Center: AC)
   - ฐานข้อมูลเครื่องโทรศัพท์มือถือ (Equipment Identity Register: EIR)
   - ศูนย์บริการเสริม เช่น ศูนย์บริการส่งข่าวสารสั้น (Short Message Service Center: SMS-C) เป็นต้น
3. ศูนย์ปฏิบัติการและบำรุงรักษา (Operation and Maintenance Center: OMC)
   - ศูนย์ปฏิบัติการและบำรุงรักษาสำหรับระบบย่อยสถานีฐาน (Operation and Maintenance Center for Base Station Sub System: OMC-B)
4. สถานีเคลื่อนที่หรือโทรศัพท์เคลื่อนที่ (Mobile Station: MS)

2.3.2 การใช้โทรศัพท์ในการติดต่อสื่อสาร (Mobile Phone for Human Communications)



รูปที่ 2.2 แสดงรูปแบบการติดต่อสื่อสารแบบ GSM

การติดต่อสื่อสารของโทรศัพท์มือถือนั้น สามารถติดต่อกันได้ทั้งระหว่าง PLMN (Public Land Mobile Network) กับ PSTN (Public Switched Telephone Network) โดยมี Gateway MSC (GMSC) เป็นอุปกรณ์ในการติดต่อประสานหรือติดต่อภายในระบบเครือข่าย PLMN ด้วยกันเอง ดังนั้น ถ้านาย A ทำการโทรเข้าโทรศัพท์มือถือของนาย B โดยใช้โทรศัพท์ที่บ้านของเค้าเอง เมื่อนาย A เริ่มการโทรออก ข้อมูลจากโทรศัพท์ต้นทางจะถูกนำไปใช้ในการหาที่อยู่ ณ ขณะนั้นของโทรศัพท์ปลายทาง โดยส่งไปยัง PSTN เพื่อค้นหาเส้นทางในการเชื่อมต่อเครือข่ายปลายทาง (GMSC) และไปที่ HLR (Home Location Register) ส่วนที่เก็บข้อมูล รายละเอียด และเครือข่าย SIM ของเครื่องเอาไว้ และเช็คต่อไปยัง VLR (Visitors Location Register) ในแต่ละเครือข่ายว่าอยู่ใน MSC ไหน (VLR จะมีกระบวนการ Location Update เพื่อบันทึกเครื่องที่เข้ามาภายใน Location Area ของชุมสายนั้นๆ) โดยใช้ IMSI (International Mobile Subscriber Identity) รหัสเฉพาะเครื่องที่ได้มาจาก HLR เปรียบเทียบ หากพบว่าอยู่ที่ MSC ไหน VLR จะส่ง MSRN (Mobile Station Roaming Number) เลขหมายชั่วคราวที่กำหนดขึ้นมาเป็นเส้นทางให้กับ GMSC ใช้ในการ Route ไปยัง MSC ปลายทาง เมื่อ Route เส้นทางได้ก็จะทำการ Paging แจ้งไปยังโทรศัพท์เครื่องนั้นๆ สัญญาณ Paging จะถูกส่ง ออกจาก BS (Base Station) ที่ควบคุมบริเวณ Location Area นั้นอยู่ และเมื่อโทรศัพท์ปลายทางรับสายตอบรับกลับมาการเชื่อมต่อก็จะเสร็จสมบูรณ์ ทำให้นาย A สามารถติดต่อสนทนากับนาย B ได้



รูปที่ 2.3 แสดงการสนทนาของ A และ B

เนื่องด้วยเสียงของมนุษย์ถือเป็นสัญญาณแบบอนาล็อค (Analog Signal) เมื่อเราใช้เสียงของเราพูดผ่านโทรศัพท์มือถือ เสียงของเราจะถูกแปลงผสมสัญญาณเสียงกับคลื่นพาหะ (Modulate) เพื่อลดการถูกรบกวนจากสัญญาณอื่น และขยายสัญญาณด้วยเครื่องส่ง เพื่อส่งออกอากาศด้วยคลื่นวิทยุไปให้สถานีฐาน (BS) จากนั้นสัญญาณวิทยุจะถูกส่งจากเครือข่ายไปสู่บุคคลปลายทางที่เราได้ทำการติดต่อ คลื่นวิทยุจะถูกแปลงกลับมาเป็นสัญญาณเสียงอีกครั้งหนึ่งด้วยเครื่องรับภายในโทรศัพท์ปลายสาย เพื่อให้อีกบุคคลหนึ่งสามารถรับฟังและเกิดความเข้าใจได้

ขอบเขตในการติดต่อสื่อสาร

การติดต่อของโทรศัพท์มือถือกับสถานีฐานต้องอยู่ในขอบเขตหรือพื้นที่สัญญาณที่สถานีฐานส่งไปถึง ซึ่งขอบเขตดังกล่าวมีปัจจัยร่วมด้วยกัน 2 ปัจจัย คือ

1. กำลังส่งของโทรศัพท์มือถือ โทรศัพท์มือถือที่มีกำลังส่งต่างกันจะมีผลทำให้ขอบเขตของการติดต่อกับสถานีฐาน (BS) ต่างกันด้วย โดยที่กำลังส่งของโทรศัพท์มือถือจะแปรผันไปตามระยะห่างจากสถานีฐาน ถ้าสถานีฐานวัดระดับสัญญาณที่ได้รับจากโทรศัพท์มือถือแล้วไม่อยู่ในระดับที่กำหนด สถานีฐานจะส่งสัญญาณไปยังส่วนควบคุมของโทรศัพท์มือถือ เพื่อให้เพิ่มหรือลดกำลังของโทรศัพท์มือถือ

2. ตำแหน่งของโทรศัพท์มือถือ ตำแหน่งที่เราคุยโทรศัพท์มือถือจะมีผลต่อขอบเขตในการติดต่อกับสถานีฐาน (BS) เช่น ถ้าเรายืนคุยโทรศัพท์อยู่ที่ชั้นดาดฟ้าของตึก CB4 จะมีขอบเขตในการติดต่อไปยังสถานีฐานไกลกว่ากับการที่เรายืนคุยโทรศัพท์อยู่ที่ 7-Eleven หน้าตึก CB4 แต่การคุยโทรศัพท์ที่ 7-Eleven จะถูกสัญญาณรบกวนน้อยกว่าการไปคุยโทรศัพท์ที่ชั้นดาดฟ้า เพราะตำแหน่งของโทรศัพท์มือถือนั้นมีผลต่อคุณภาพของสัญญาณด้วย

2.3.3 หลักการรับส่ง SMS ของโทรศัพท์มือถือ

SMS ย่อมาจาก Short Message Service เป็นบริการส่งข้อความสั้นๆจา กโทรศัพท์มือถือต้นทางผ่านชุมสายไปยังโทรศัพท์มือถือปลายทาง โดยสามารถส่งได้สูงสุด 160 ตัวอักษรต่อครั้ง ตามข้อกำหนดมาตรฐานขององค์การ ETSI (European Telecommunications Standards Institute) นอกจากนี้ยังสามารถส่งข้อความไปที่เครื่อง Fax, PC หรือ Internet Address อีกด้วย

ระบบSMSในระบบเครือข่ายโทรศัพท์มือถือรองรับโดยระบบ GSM (Global System for Mobile Communication), TDMA (Time Division Multiple Access) และ CDMA (Code Division Multiple Access)

เมื่อ SMS ถูกส่งจากโทรศัพท์มือถือเครื่องหนึ่ง ข้อความนั้นจะถูกส่งไปที่ Short Message Service Center (SMSC) จากนั้นจึงจะส่งไปยังโทรศัพท์มือถือเครื่องรับอีกทอดหนึ่ง โดยมีกระบวนการดังนี้

1. SMSC จะส่ง SMS Request ไปยัง Home Location Register (HLR) เพื่อหาตำแหน่งของผู้รับ

2. เมื่อ HLR ได้รับสัญญาณ Request ก็จะส่งสถานะของผู้รับ (Subscriber's Status) กลับมายัง SMSC คือ สถานะของเครื่องรับ Inactive หรือ Active, ตำแหน่งของเครื่องรับถ้าสถานะของเครื่องรับเป็น Inactive แล้ว SMSC จะเก็บข้อความไว้ช่วงเวลาหนึ่ง และเมื่อใดที่เครื่องรับมีสถานะ Active แล้ว HLR จะส่ง SMS Notification ไปยัง SMSC และ SMSC ก็จะตอบรับข้อความนั้นไว้ จากนั้น SMSC จะส่งผ่านข้อความในรูปแบบ Short Message Delivery Point-to-Point ไปยังระบบบริการ โดยระบบจะทำการเรียกไปยังเครื่องรับและถ้าเครื่องรับมีการตอบรับกลับมา ข้อความก็จะถูกส่งตามไปและ SMSC จะได้รับการตอบยืนยันว่า ข้อความได้ถูกรับโดยปลายทางเรียบร้อยแล้ว หลังจากนั้นข้อความจะมีสถานะเป็น SENT และจะไม่ถูกส่งอีก

โหมดของการรับส่งข้อมูล SMS

แบ่งออกเป็น 2 โหมด คือ Text Mode และ PDU Mode (Protocol Description Unit Mode)

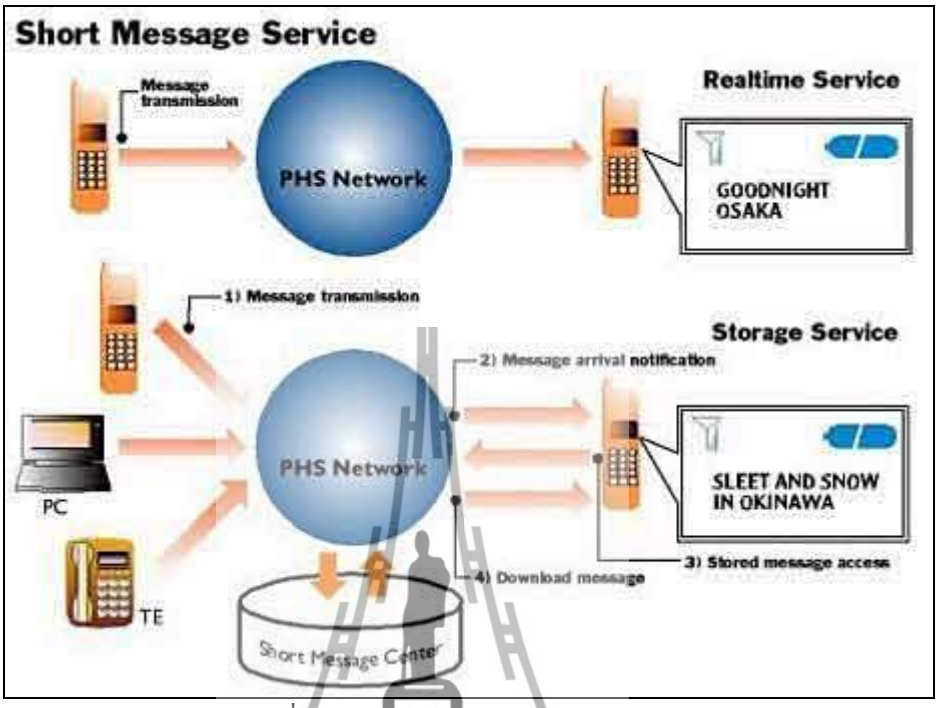
การส่งข้อความใน Text Mode นั้นจะเป็นการนำข้อความที่ต้องการส่งมาเข้ารหัสก่อน (โดยตัวเครื่องเอง) แล้วจึงส่งข้อมูลในรูป PDU Mode อีกครั้งหนึ่ง แต่ในบางเครื่องก็ไม่สนับสนุนการส่งแบบ Text Mode ผ่านทาง AT Command แต่หากเป็น PDU Mode จะสามารถส่งได้ นอกจากเครื่องจะไม่ต้องทำอาศัยการเปลี่ยนแปลงข้อมูลอีกชั้น

รูปแบบในการส่งข้อมูลในรูป SMS ผ่านทาง AT Command

มี 2 รูปแบบ คือ Text Mode และ PDU Mode

- Text Mode เป็นการส่งข้อมูลในรูปของตัวอักษรได้โดยตรง ซึ่งตัวเครื่องส่วนใหญ่ไม่รองรับการส่งข้อมูลรูปแบบนี้ผ่านทาง AT Command จึงไม่สามารถใช้งานได้สมบูรณ์

- PDU Mode PDU ย่อมาจาก PACKET DATA UNIT เป็นรูปแบบการส่งข้อความ SMS อีกรูปแบบหนึ่งที่ต้องมีการเข้ารหัสข้อมูลที่สลับซับซ้อนแต่ตัวเครื่องจะสามารถรับรู้ได้ทุกเครื่องที่รับคำสั่ง AT Command ได้



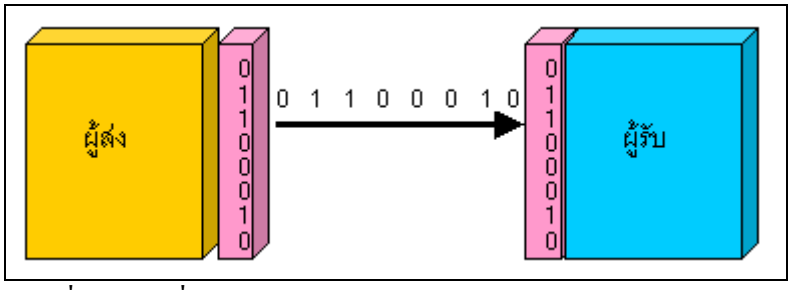
รูปที่ 2.4 แสดงขอบเขตของการส่ง SMS

2.4 การสื่อสารข้อมูล

ระบบคอมพิวเตอร์สามารถส่งข้อมูลได้สองรูปแบบคือ แบบขนาน (Parallel) ซึ่งจะส่งข้อมูลทุกบิตออกไปพร้อมกัน และแบบอนุกรม (Serial) ซึ่งจะส่งข้อมูลออกไปครั้งละบิต การส่งข้อมูลแบบขนานนี้จะต้องใช้สายในการส่งข้อมูลจำนวนมากซึ่งไม่เหมาะกับการส่งระยะไกล โดยมากแล้วจะใช้ในการส่งระยะใกล้ เช่น ระหว่างไม โครโพรเซสเซอร์กับฮาร์ดดิสก์ เครื่องพิมพ์ เป็นต้น ส่วนการส่งระยะไกลๆควรใช้การส่งข้อมูลแบบอนุกรมโดยจะส่งข้อมูลออกไปครั้งละบิต การส่งข้อมูลแบบอนุกรมนั้นจะส่งได้ช้ากว่าการส่งแบบขนานแต่ค่าใช้จ่ายจะต่ำกว่า ในไมโครโพรเซสเซอร์ 8051 จะมีพอร์ตอนุกรมอยู่ภายในซึ่งจะรับส่งข้อมูลได้สองทิศทางในพอร์ตเดียวกัน

- การสื่อสารข้อมูลแบบอนุกรม (Serial Data Transmission)

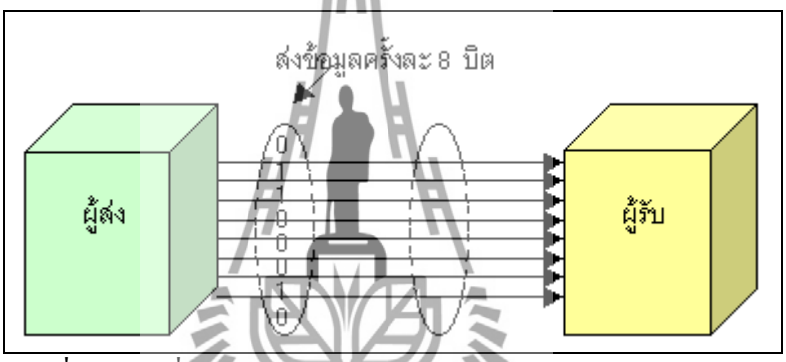
เป็นการส่งข้อมูลครั้งละ 1 บิต ไปบนสัญญาณจนครบจำนวนข้อมูลที่มีอยู่ สามารถนำไปใช้กับสื่อนำข้อมูลที่มีเพียง 1 ช่องสัญญาณได้ สื่อนำข้อมูลที่มี 1 ช่องสัญญาณนี้จะมีราคาถูกกว่าสื่อนำข้อมูลที่มีหลายช่องสัญญาณ และเนื่องจากการสื่อสารแบบอนุกรมมีการส่งข้อมูลได้ครั้งละ 1 บิตเท่านั้น การส่งข้อมูลประเภทนี้จึงช้ากว่าการส่งข้อมูลครั้งละหลายบิต



รูปที่ 2.5 การสื่อสารข้อมูลแบบอนุกรม (Serial Data Transmission)

- การสื่อสารข้อมูลแบบขนาน (Parallel Data Transmission)

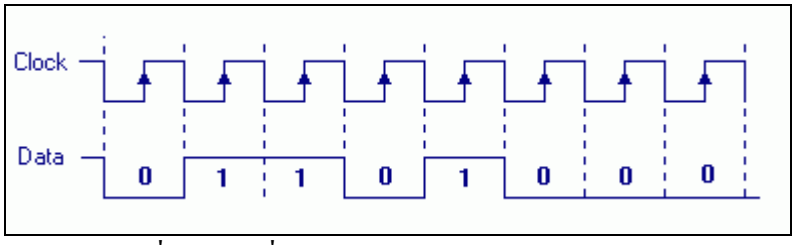
เป็นการส่งข้อมูลครั้งละหลายบิตขนานกันไปบนสื่อนำข้อมูลที่มีหลายช่องสัญญาณ วิธีนี้จะเป็นวิธีการส่งข้อมูลที่เร็วกว่าการส่งข้อมูลแบบอนุกรม จากรูปที่ 6.10 เป็นการแสดงการสื่อสารข้อมูลระหว่างอุปกรณ์ 2 ตัว ที่มีการส่งข้อมูลแบบขนาน โดยส่งข้อมูลครั้งละ 8 บิตพร้อมกัน



รูปที่ 2.6 การสื่อสารข้อมูลแบบขนาน (Parallel Data Transmission)

1.) รูปแบบการสื่อสารแบบอนุกรม มีด้วยกันอยู่ 2 แบบ คือแบบซิงโครนัส (Synchronous) และแบบเอซิงโครนัส (Asynchronous)

- การสื่อสารแบบซิงโครนัส (Synchronous) การรับส่งข้อมูล จะมีสัญญาณนาฬิกา ซึ่งเป็นตัวกำหนด จังหวะเวลา การส่งข้อมูล ร่วมอยู่ด้วยอีกเส้นหนึ่ง ใช้คู่กับสัญญาณข้อมูล ตัวอย่างเช่น การส่งสัญญาณจากคีย์บอร์ด



รูปที่ 2.7 การสื่อสารแบบซิงโครนัส (Synchronous)

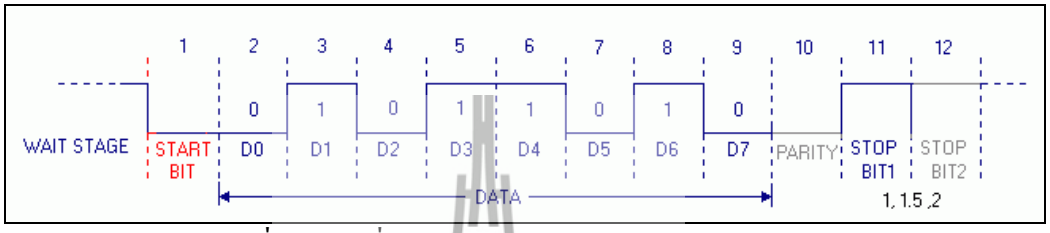
- การสื่อสารแบบ เอซิงโครนัส (Asynchronous) การรับส่งข้อมูล โดยที่ไม่จำเป็นต้องมีสัญญาณนาฬิกา ร่วมด้วย แต่จะใช้ให้ตัวส่ง และตัวรับ มีอัตราส่งข้อมูลที่เท่ากัน รูปแบบข้อมูลแบบเอซิงโครนัส ประกอบด้วย 4 ส่วนคือ

1.บิตเริ่มต้น (Start Bit) มีขนาด 1 บิต

2.บิตข้อมูล (Data) มีขนาด 5, 6, 7 หรือ 8 บิต

3.บิตตรวจสอบพาริตี้ (Parity Bit) มีขนาด 1 บิตหรือไม่มี

4.บิตหยุด (Stop Bit) มีขนาด 1, 1.5, 2 บิต



รูปที่ 2.8 การสื่อสารแบบเอซิงโครนัส (Asynchronous)

- เมื่อไม่มีการส่งข้อมูล ขา Data จะมีสถานะเป็นโลจิก "1" หรือ สถานะหยุดรอ (Waiting Stage)

- เมื่อเริ่มต้นส่งข้อมูลจะให้ขา Data เป็นโลจิก "0" เป็นจำนวน 1 บิต เรียกว่าบิตเริ่มต้น (Start Bit)

- จากนั้นก็จะเริ่มต้นส่งข้อมูล โดยส่งบิตต่ำไปก่อน (LSB)

- แล้วตามด้วยพาริตี้บิต (จะมีหรือไม่ก็ได้ ขึ้นอยู่กับการติดตั้งค่า ของทั้งสองฝ่าย)

- สุดท้ายตามด้วยโลจิก "1" อย่างน้อย 1 บิต (มีขนาด 1, 1.5, หรือ 2 บิต) เพื่อแสดงว่าสิ้นสุดข้อมูล

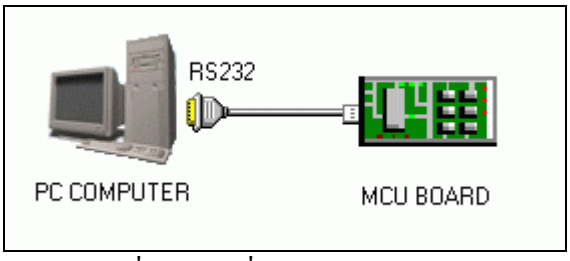
2.) การรับและส่งข้อมูลแบบอนุกรมยังแบ่งออกเป็นลักษณะการใช้งานได้ 3 แบบคือ

- แบบซิมเพลกซ์ (Simplex) เป็นการส่ง หรือรับข้อมูลแบบทิศทางเดียวเท่านั้น

- แบบฮาล์ฟดูเพลกซ์ (Half Duplex) เป็นการส่งและรับข้อมูลแบบสลับกันคือ เมื่อด้านหนึ่งส่ง อีกด้านหนึ่งเป็นฝ่ายรับ สลับกันไม่สามารถรับ-ส่งในเวลาเดียวกันได้

- แบบฟลูดูเพลกซ์ (Full Duplex) สามารถรับ-ส่งข้อมูลในเวลาเดียวกัน

3.) การใช้งานพอร์ต RS-232



รูปที่ 2.9 การสื่อสารข้อมูล RS-232

การสื่อสารแบบอนุกรม นับว่ามีความสำคัญ ต่อการใช้งาน ไมโครคอนโทรลเลอร์มาก เพราะสามารถใช้แป้นพิมพ์ และจอภาพของ PC เป็น อินพุต และ เอาต์พุต ในการติดต่อ หรือควบคุม ไมโครคอนโทรลเลอร์ ด้วยสัญญาณอย่างน้อย เพียง 3 เส้นเท่านั้น คือ

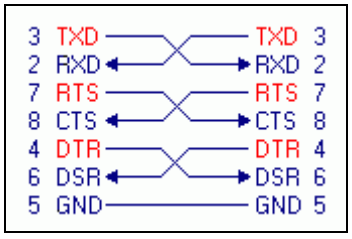
- สายส่งสัญญาณ TX
- สายรับสัญญาณ RX
- และสาย GND

โดยปกติพอร์ตอนุกรม RS-232C จะสามารถต่อสายได้ยาว 50 ฟุตโดยประมาณ ขึ้นอยู่กับชนิดของสายสัญญาณ, ระยะทาง, และปริมาณสัญญาณรบกวน

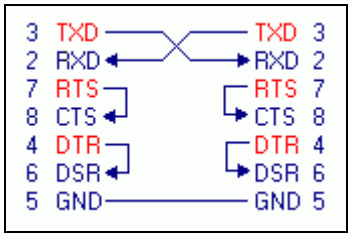
รูปที่ 2.10 พอร์ตอนุกรมของ PC DB9 ตัวผู้ (Male)

รูปที่ 2.11 พอร์ตอนุกรมของอุปกรณ์ภายนอก DB9 ตัวเมีย (Female)

- พอร์ตอนุกรมของ PC จะเป็นคอนเน็คเตอร์แบบ DB9 ตัวผู้ (Male)
- พอร์ตอนุกรม ของอุปกรณ์ภายนอก จะเป็นคอนเน็คเตอร์แบบ DB9 ตัวเมีย (Female)



รูปที่ 2.12 การเชื่อมต่ออุปกรณ์ภายนอกผ่าน DB9 แบบ Null Modem



รูปที่ 2.13 การต่ออุปกรณ์ภายนอกผ่าน DB9 แบบ 3 เส้น

4.) การทำงานของขาสัญญาณ DB9

TXD เป็นขาที่ใช้ส่งข้อมูล

RXD เป็นขาที่ใช้รับข้อมูล

DTR แสดงสภาวะพอร์ตว่าเปิดใช้งาน, DSR ตรวจสอบว่าพอร์ต ที่ติดต่อด้วย เปิดอยู่หรือไม่

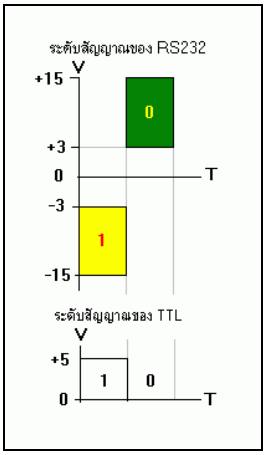
- เมื่อเปิดพอร์ตอนุกรม ขา DTR จะ ON เพื่อให้อุปกรณ์ได้รับทราบว่าต้องการติดต่อด้วย
- ในขณะเดียวกันก็จะตรวจสอบขา DSR ว่าอุปกรณ์พร้อมหรือไม่

RTS แสดงสภาวะพอร์ตว่าต้องการส่งข้อมูล, CTS ตรวจสอบว่าพอร์ตที่ติดต่ออยู่ ต้องการส่งข้อมูลหรือไม่

- เมื่อต้องการส่งข้อมูลขา RTS จะ ON และจะส่งข้อมูลออกที่ขา TXD เมื่อส่งเสร็จก็จะOFF
- ในขณะเดียวกันก็จะตรวจสอบขา CTS ว่าอุปกรณ์ต้องการที่จะส่งข้อมูลหรือไม่

GND ขา ground

5.) ระดับสัญญาณของ RS232



รูปที่ 2.14 แสดงระดับสัญญาณของ RS232C และระดับสัญญาณของ TTL

- สัญญาณรบกวนที่เกิดขึ้นในสายนำสัญญาณ มักจะมีแรงดันเป็นบวก เมื่อเทียบกับกราวด์
- เพื่อป้องกันสัญญาณรบกวนนี้ จึงออกแบบแร งดันของโลจิก "1" เป็นลบ คือ อยู่ในช่วง 3V ถึง -15V ส่วนแรงดันของโลจิก " 0" อยู่ในช่วง +3V ถึง +15V และเหตุที่ระดับสัญญาณของ RS232 อยู่ในช่วง +15V ถึง -15V ก็เพื่อให้ต่อสายสัญญาณไปได้ไกลขึ้น
- ดังนั้นจึงจำเป็นจะต้องมีวงจรเปลี่ยนระดับแรงดันของ RS232 มาเป็นระดับแรงดันของ TTL

6.) อัตราการส่งข้อมูล (Baud Rate)

คือความเร็วของการรับ-ส่งข้อมูล เป็นจำนวนบิ ตต่อวินาทีเช่น 300, 1,200, 2,400, 4,800, 9,600, 14,400, 19,200, 38,400, 56,000 เป็นต้น การเลือกอัตราการส่งข้อมูลขึ้นอยู่กับ ชนิดของสายสัญญาณ, ระยะทาง, และปริมาณสัญญาณรบกวน

2.5 เซ็นเซอร์ (Sensor)

เซ็นเซอร์เป็นตัวที่ใช้ตรวจจับสภาวะใดๆ เช่น อุณหภูมิ แสง เสียง หรือวัตถุ ต่างๆ โดยอาศัยหลักการที่แตกต่างกันไปแต่ละตัวเพื่อเปลี่ยนจากคุณสมบัติทางฟิสิกส์มาเป็นคุณสมบัติทางไฟฟ้า โดยทั่วไปเทคโนโลยีของเซ็นเซอร์ได้ถูกนำไปใช้เป็นองค์ประกอบหลักที่สำคัญในลักษณะงานสองประเภทคือ ใช้ตรวจวัดปริมาณทางฟิสิกส์เพื่อนำไปแสดงผลการตรวจวัดหรือจัดเก็บบันทึกข้อมูลในระบบการวัด เช่น การวัดอุณหภูมิ ความชื้น แสง เป็นต้น และอย่างที่สองคือใช้ตรวจสอบสภาพกระบวนการในระบบการควบคุม เซ็นเซอร์สำหรับการตรวจวัดข้อมูลที่เป็นตัวแปรทางฟิสิกส์ โดยมากจะถูกนำไปใช้เป็นข้อมูลเพื่อแสดงสถานะ สภาพของระบบในขณะนั้น

2.5.1 pH Sensor

pH เซ็นเซอร์ คืออุปกรณ์ไฟฟ้าเคมี (Electrochemical) ที่สร้างแรงเคลื่อนไฟฟ้าซึ่งเป็นสัดส่วนกันค่า pH ของสารละลายที่นำไปวัด

ส่วน pH มิเตอร์ก็ใช้หลักการที่กล่าวมา คือใช้หลักการวัดค่าความเป็นกรดหรือด่างที่มีน้ำเป็นตัวทำละลาย ซึ่งก็ใช้หลักการทางเคมีไฟฟ้าเหมือนกัน หลังจากนั้นก็นำค่าความต่างศักย์ที่เกิดขึ้นมาระหว่างอิเล็กโทรดอ้างอิง (Reference Electrode) กับอิเล็กโทรดวัด (Sensing Electrode) ไปขยาย ปรับสภาพ และแสดงผลต่อไป

หลักการวัดค่า pH

เมื่อสารละลาย 2 ตัวที่แตกต่างกันถูกใส่สงในด้านทั้งสองของเมมเบรนแก้ว จะทำให้เกิดแรงเคลื่อนไฟฟ้าที่เป็นสัดส่วนกับความแตกต่างของ pH ระหว่างสารละลายทั้งสอง และตกคร่อมทั้งสองด้านของเมมเบรน

ข้อดีของการวัดด้วยวิธีนี้ได้แก่

1. ย่านการวัดกว้าง (ไม่มีปัญหาช่วง pH 0 – 14)
2. ให้ผลตอบสนองที่รวดเร็ว (วัดในช่วงเวลาสั้น ๆ)
3. การปฏิบัติงานง่าย การวัดต่อเนื่องทำได้ง่าย
4. การผลิตซ้ำดี พร้อมกับมีความผิดพลาดโดยผู้ปฏิบัติงานน้อย
5. ความผิดพลาดเนื่องจากผลของเกลือและโปรตีนต่ำกว่าวิธีอื่นๆ (วิธีการวัดแบบอื่นๆ ไม่สามารถวัดสารตัวอย่างที่ประกอบด้วยสสารแบบออกซิไดซิงและรีดิวซิงได้)

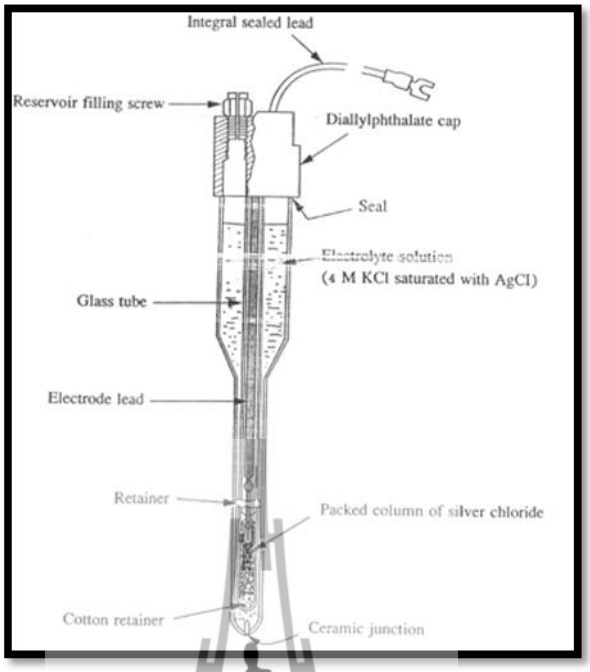
ข้อเสียของการวัดด้วยวิธีนี้ได้แก่

1. เมมเบรนเปราะ แตกได้ง่าย
2. ความต้านทานภายในของอิเล็กโทรดสูงทำให้ต้องใช้มิลลิโวลต์มิเตอร์ที่มีความต้านทานอินพุตสูง

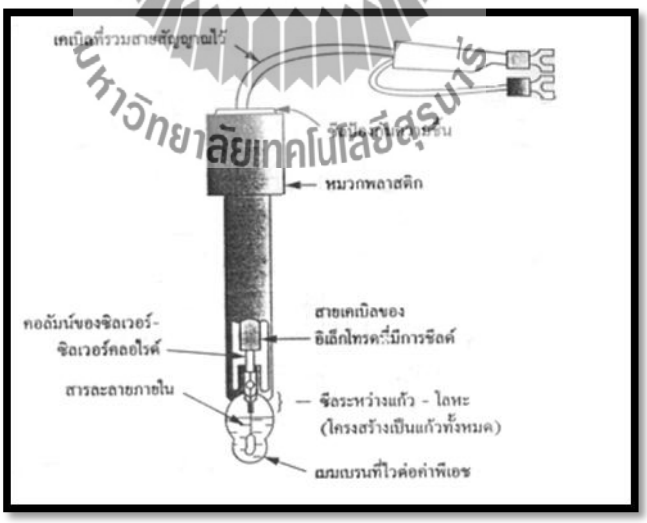
ส่วนประกอบและการทำงานของอิเล็กโทรดวัดและอิเล็กโทรดอ้างอิง

1. ขั้ววัด (Sensors Electrode) โดยปกติอิเล็กโทรดวัดจะทำจากแก้ว ขั้ววัดจะมีกระเปาะแก้วบางมาก ๆเพื่อให้ไวต่อออออน H+ ภายในอิเล็กโทรดจะบรรจุสารละลายกันชนที่เรียกว่า บัฟเฟอร์ (Buffer) ซึ่งมีค่า pH คงที่ (ประมาณ pH 7) สารละลายนี้ได้แก่ KCl อิ่มตัว ในสารละลายดังกล่าวจะมีขั้วไฟฟ้าซึ่งทำด้วยเงิน (Silver) ฉาบด้วยซิลเวอร์คลอไรด์จุ่มอยู่ ดังแสดงในรูปที่ 2.16

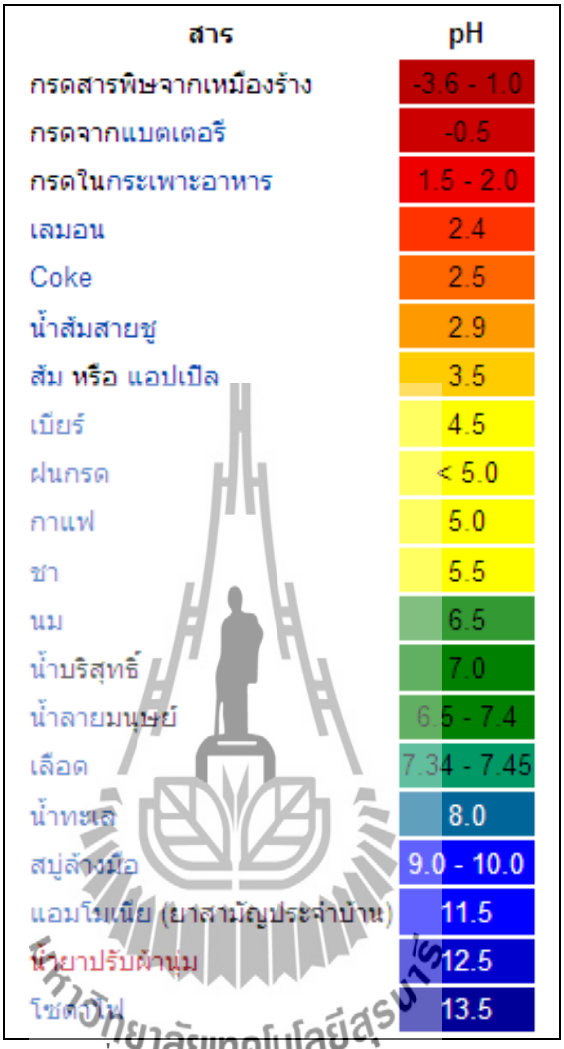
2. ขั้วอ้างอิง (Reference Electrode) หรือเรียกว่าขั้วเปรียบเทียบ มีไว้เพื่อให้ศักย์ไฟฟ้าที่ขั้ววัดครบวงจร ขั้วอ้างอิงนี้จะมีลวดนำ (Lead Wire) ต่อกับซิลเวอร์คลอไรด์และจุ่มซิลเวอร์คลอไรด์ลงใน KCl ชนิดอิ่มตัว KCl นี้จะซึมผ่านรูเล็กๆ ที่ปลายของหลอดแก้วเพื่อสัมผัสกับสารละลายที่จะวัด การเชื่อมต่อระหว่างสาร ละลายทั้งสองนี้เรียกว่า “Salt Bridge” รูที่เชื่อมต่อนี้มักจะมีสารเอสเบสตอสหรือเซรามิกวางคั่นเอาไว้ เพื่อทำให้การไหลของ KCl คงที่ตลอดเวลาขณะที่ทำการวัด ดังรูปที่ 2.15



รูปที่ 2.15 แสดงตัวอย่างของอิเล็กโทรดวัดค่า pH แบบแก้ว



รูปที่ 2.16 ตัวอย่างอิเล็กโทรดอ้างอิงที่วัดค่า pH (อิเล็กโทรดแบบซิลเวอร์/ซิลเวอร์-คลอไรด์)



รูปที่ 2.17 ตัวอย่างค่า pH ของสารต่าง ๆ

2.5.2 eC Sensor

eC (Electric Conductivity) ค่า eC ของสารละลายเป็นค่า ที่วัด เพื่อแสดงถึงความเข้มข้นของเกลือทั้งหมดที่ละลายอยู่ในน้ำ ซึ่งเป็นค่า ที่วัดโดยรวมไม่สามารถแยกบอกความเข้มข้นของเกลือแต่ละตัวได้ หน่วยการวัดค่า eC มีหลายหน่วยแล้วแต่เครื่องมือที่ใช้วัด

ความหมายของค่า eC

น้ำบริสุทธิ์จะมีค่าความนำไฟฟ้าเป็นศูนย์ แต่เมื่อน้ำมีเกลือละลายอยู่ เกลือเหล่านี้จะแตกตัวเป็นประจุบวก (Cation) และประจุลบ (Anion) ซึ่งประจุบวกและลบที่เกิดขึ้น จะเป็นตัวนำไฟฟ้าทำให้สารละลายที่มีเกลือที่แตกตัวได้มีค่าความนำไฟฟ้า (Electric Conductivity) ซึ่งค่านำไฟฟ้าที่เกิดขึ้นจะเป็นสัดส่วนโดยตรงกับปริมาณเกลือที่ละลายอยู่ในน้ำ ดังนั้น จึงสามารถใช้ค่า ความนำไฟฟ้าของสารละลายเป็นตัวบอกปริมาณเกลือที่ละลายในสารละลาย เช่น เมื่อเกลือแกงละลายในน้ำจะแตกตัวได้ NaCl -- ----------> Na^+ + Cl^- น้ำตาล และยูเรีย สามารถละลายน้ำได้เหมือนกัน แต่เมื่อละลายแล้วจะไม่แตกตัว ดังนั้นก็จะไม่เพิ่มค่าความนำไฟฟ้าของสารละลาย จึงไม่สามารถวัดความเข้มข้นด้วยค่าความนำไฟฟ้าได้ แต่เนื่องจากปุ๋ยชนิดต่าง ๆที่ใช้ ส่วนใหญ่เป็นสารที่สามารถแตกตัวได้ สารที่มีประจุบวกและประจุลบทุกตัวจึงสามารถวัดความเข้มข้นโดยการวัดค่าความนำไฟฟ้าได้ ค่าความนำไฟฟ้า และค่าความเข้มข้นของสารละลายจะมีความสัมพันธ์แบบเป็นเส้นตรง กล่าวคือ ถ้าความเข้มข้นสารละลายเพิ่มขึ้นหนึ่งเท่าตัว ค่า ความนำไฟฟ้าก็จะเพิ่มหนึ่งเท่าด้วย ดังที่กล่าวมาแล้ว ค่า ความนำไฟฟ้าบอกให้ทราบถึงปริมาณเกลือที่ละลายโดยรวมอยู่ในสารละลายไม่สามารถแยกชนิดของเกลือได้ ตัวอย่างเช่นในน้ำมีเกลือ NaCl และปุ๋ย KNO_3 ละลายรวมกันอยู่ และวัดค่า eC ได้ 2.5 mS/cm เราไม่สามารถทราบได้ว่ามี NaCl อยู่เท่าใด และ มี KNO_3 อยู่เท่าใดทราบเพียงแต่ว่ามีอยู่รวมกัน มีค่า 2.5mS/cm การเปลี่ยนค่าที่อ่านจากเครื่องวัด eC ที่มีหน่วยเป็น 1.2 mS/cm ให้เป็นความเข้มข้นสามารถทำได้โดยคูณค่า mS/cm ที่อ่านได้ด้วยค่าคงที่ซึ่งขึ้นอยู่กับชนิดของเกลือที่ละลายอยู่ ซึ่งค่าโดยทั่วไปที่ใช้เกียวกับ Fertigation คือ 850 เช่น อ่านค่า eC ของน้ำได้เท่ากับ 2.5 mS/cm ถ้าต้องการทราบความเข้มข้นเป็น ppm จะได้ว่า 2.5x850 = 2125 ppm โดยประมาณ จากหลักการนี้เราสามารถใช้เครื่องวัด eC ตรวจสอบความถูกต้องของการละลายปุ๋ยลงในระบบน้ำได้ด้วย

เซ็นเซอร์วัดค่าความนำไฟฟ้าในสารละลาย (eC Sensor) เป็นเครื่องมือที่ใช้วัดค่าการนำไฟฟ้า คือ eC Sensor จะวัดค่า Electric Conductivity ระหว่างแท่งโลหะสองแท่งจุ่มอยู่ในสารละลายที่ต้องการวัดค่า ส่วนใหญ่ปัจจุบันจะวัดเป็นค่าตัวเลขออกมาเลย (Digital) เนื่องจากอุณหภูมิของสารละลายจะมีผลต่อค่า ความนำไฟฟ้าของสารละลายเป็นอย่างมาก ดังนั้นจะต้องมีการแก้ไขค่าที่อ่านได้ให้เป็นค่าที่อุณหภูมิมาตรฐานเดียวกัน

ความสัมพันธ์ของค่า Conductivity กับค่า Hardness

ค่าความกระด้าง (Hardness) หมายถึง ปริมาณรวมหรือความเข้มข้นรวมเฉพาะของแคลเซียมและแมกนีเซียม ที่แสดงในหน่วย mg/L ของ $CaCO_3$ ส่วนค่า Conductivity หมายถึงค่าการนำไฟฟ้าของสารละลาย ซึ่งจะเกิดจาก ไอออนต่างๆ ทั้งหมดที่ละลายอยู่ในสารละลายซึ่งทั้งค่ามีความสัมพันธ์ไปในทางเดียวกันคือ เมื่อค่า ความกระด้างมีค่าสูง ค่า Conductivity ก็จะมีค่าสูงไปด้วย แต่ทั้งสองค่าไม่สามารถแปลงค่าระหว่างทั้งสองค่าได้ เนื่องจากค่าความกระด้างจะเป็นค่าที่คิดจากความเข้มข้นทั้งหมดของไอออนแคลเซียมและแมกนีเซียม ส่วนค่า Conductivity จะเป็นค่าที่คิดจากไอออนโดยรวมของสารละลาย ถึงอย่างไรก็ตามเราอาจจะเปรียบเทียบเพื่อประมาณค่าได้ ดังตารางต่อไปนี้

Conductivity(uS/cm)	Hardness
0-140	Very Soft
140-300	Soft
300-500	Slightly Hard
500-640	Moderately Hard
640-840	Hard
Above 840	Very Hard

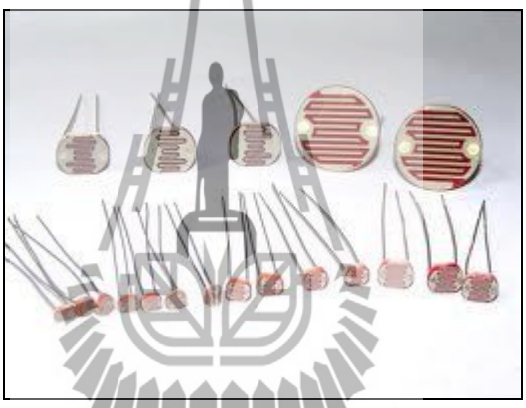
รูปที่ 2.18 แสดงตัวอย่างตารางเปรียบเทียบค่า Conductivity กับค่า Hardness

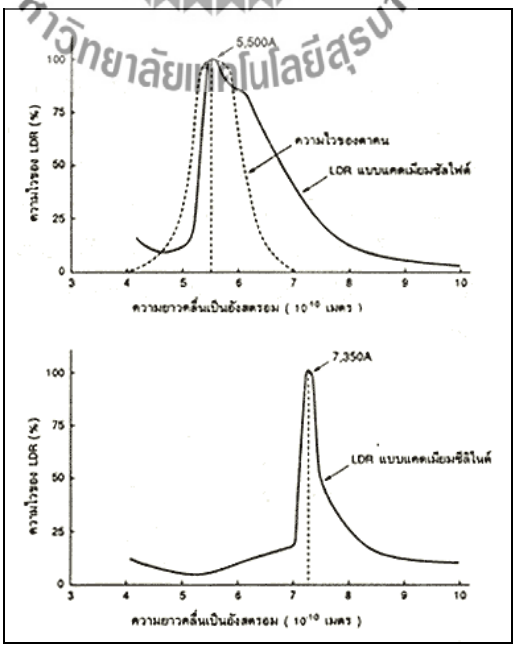
2.5.3 Light Sensor

Light Sensor คือ อุปกรณ์ที่อาศัยหลักการของแสง เช่น การนำ , การพา, และการแผ่รังสี มาเปลี่ยนเป็น สัญญาณไฟฟ้า โดยเซนเซอร์ทางแสงแบ่งเป็น 2 แบบคือ แบบไม่มีแหล่งจ่าย และแบบมีแหล่งจ่าย เช่น โฟโต้รีซิสเตอร์ (Photo Resistor), โฟโต้เซลล์, โฟโต้ไดโอดและโฟโต้ทรานซิสเตอร์

Light dependent resistor (LDR)

- Light Dependent Resistor (LDR) สร้างจากสารประกอบแคดเมียมซัลไฟด์ ซึ่งจะมีความต้านทานต่ำลงเมื่อได้รับแสง
- ความต้านทานของ LDR ไม่ได้เป็นเชิงเส้นตรงกับความเข้มแสง แต่จะเป็นเชิงลอการิทึม
- ความไวแสงขึ้นอยู่กับความยาวคลื่นแสง
- ความเร็วในการตอบสนองต่อการเปลี่ยนความเข้มแสงจะช้ากว่า Photo Diode / Photo Transistor

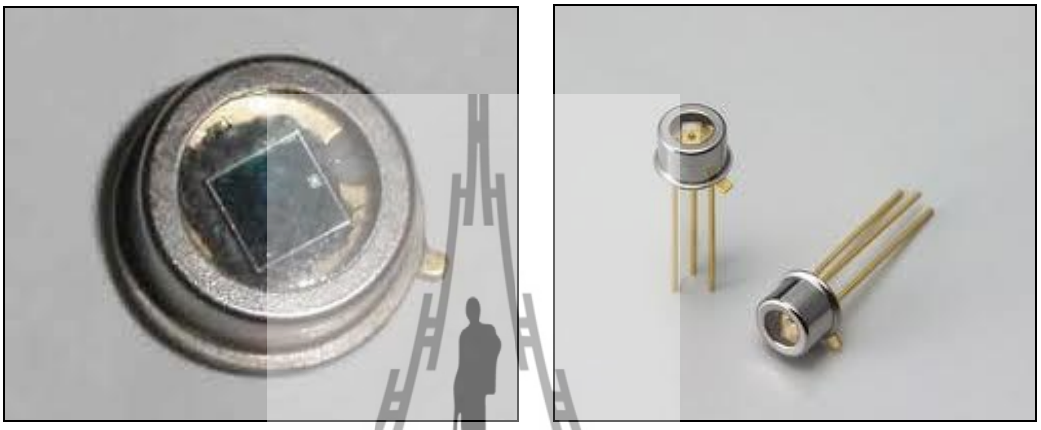


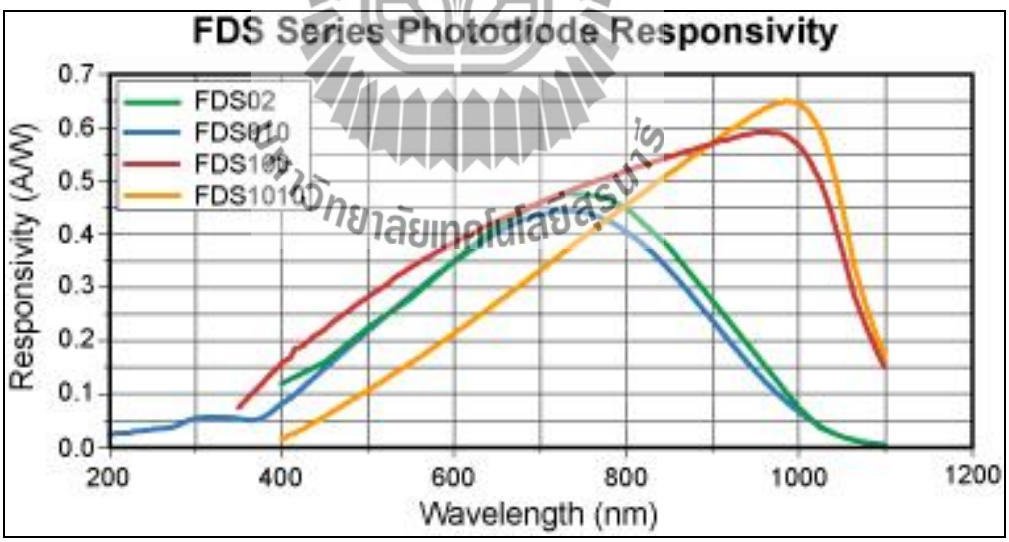


รูปที่ 2.19 แสดงตัวอย่างของ Light Dependent Resistor

Photo Diode

- Photo Diode จะเปลี่ยนแสงไฟเป็นกระแสไฟฟ้า โดยค่ากระแสไฟฟ้าจะแปรผันโดยตรงกับความเข้มแสง
- กระแสที่ได้ที่ความยาวคลื่นของแสงแตกต่างกันจะไม่เท่ากัน นั่นคือ Sensitivity ของไดโอดต่อแสงที่ความยาวคลื่นต่างกันจะไม่เท่ากันดังรูปตัวอย่างเช่น Photo Diode เช่น เบอร์ BPX65

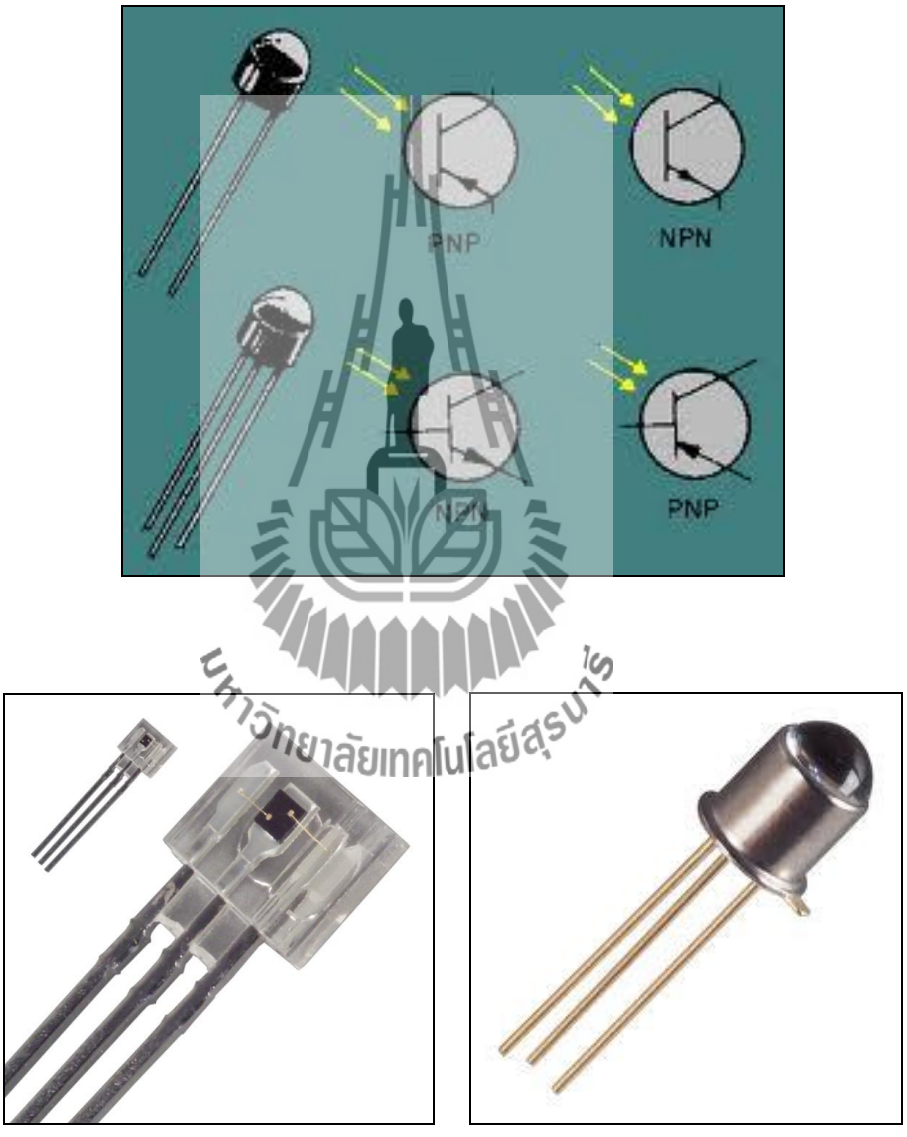




รูปที่ 2.20 แสดงตัวอย่างของ Photo Diode

Photo Transistor

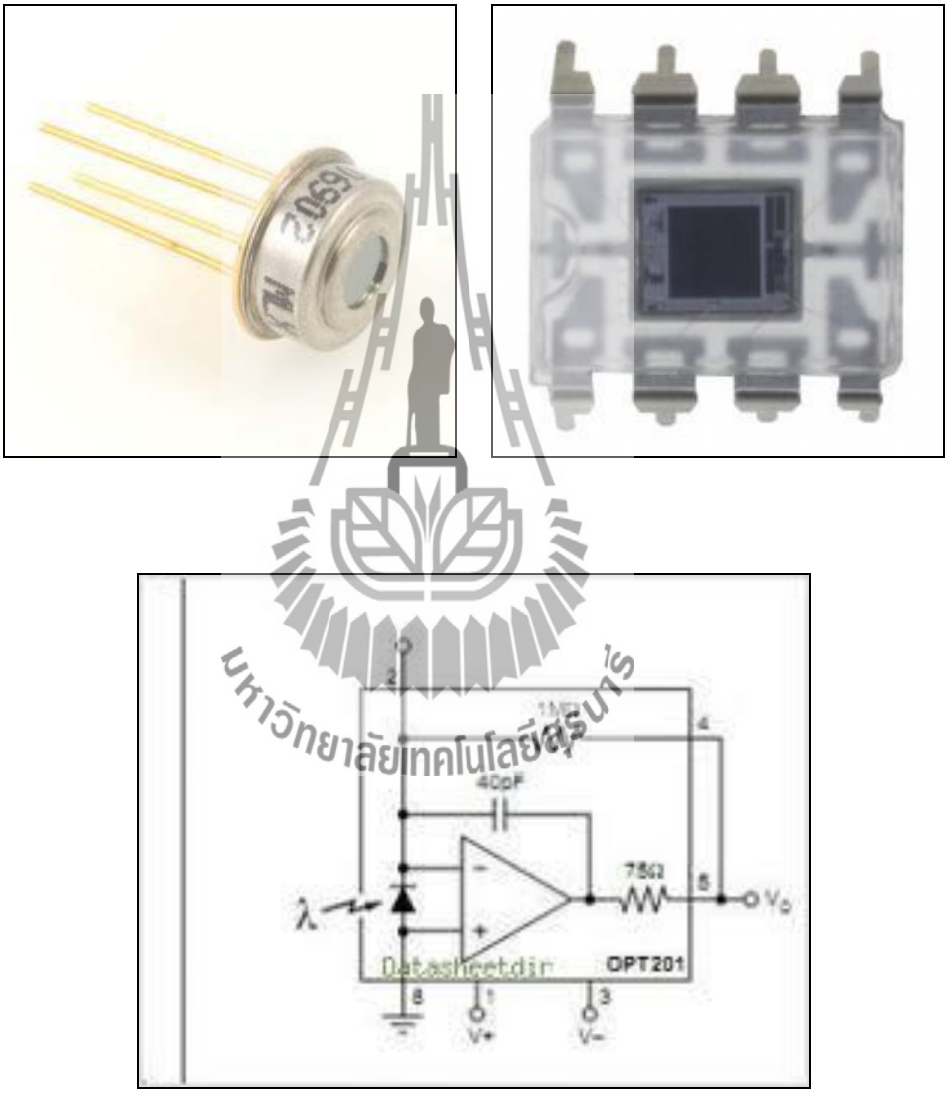
- Photo Transistor จะเปลี่ยนแสงไฟเป็นกระแสไฟฟ้า และมีการขยายสัญญาณด้วย โดยค่ากระแสไฟฟ้าจะแปรผันโดยตรงกับความเข้มแสง
- เช่น TR เบอร์
- Photo Transistor จะตอบสนองต่อแสงที่บางความยาวคลื่นได้มากเป็นพิเศษ เช่น นำมาใช้เป็นตัวรับแสงอินฟาเรดในรีโมทคอนโทรลของเครื่องใช้ไฟฟ้าต่างๆ



รูปที่ 2.21 แสดงตัวอย่างของ Photo Transistor

Light Sensor IC

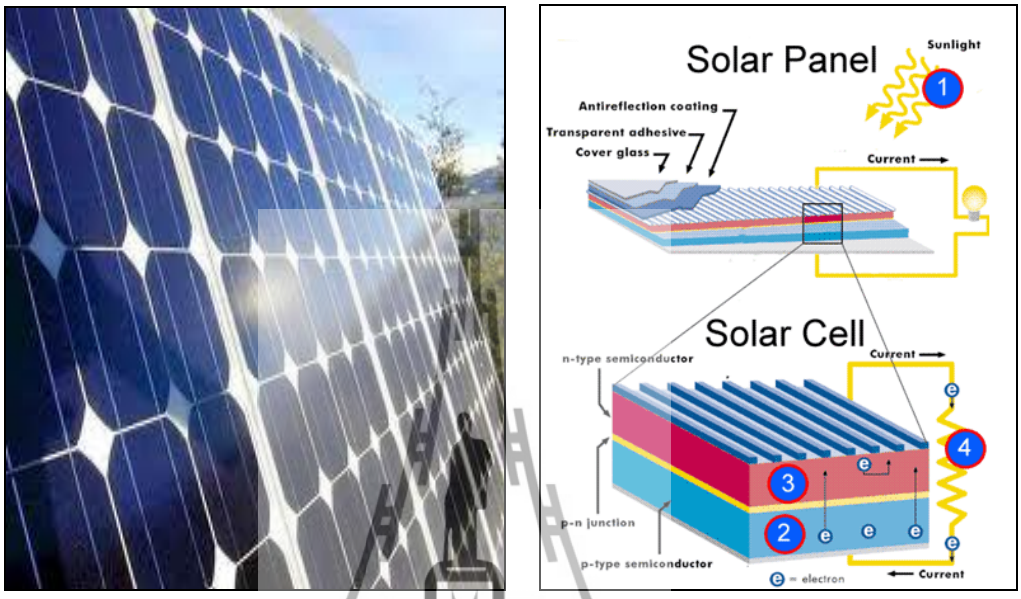
- มีการนำ Photo Diode / Transistor มาประกอบกับวงจรขยายสัญญาณ รวมเป็นไอซีซึ่งทำให้การใช้งานง่ายขึ้น และมี Sensitivity และ Linearity สูงขึ้น
- เช่นไอซี OPT101 จะให้สัญญาณ Voltage Output ช่วง 0-5 V ที่แปรผันโดยตรงกับความเข้มแสง



รูปที่ 2.22 แสดงตัวอย่างของ Light Sensor IC

Photo Cell

- Solar Cell จะเปลี่ยนแสงเป็นกระแสไฟฟ้า ได้เช่นเดียวกัน โดยค่ากระแสไฟฟ้าจะแปรผันโดยตรงกับความเข้มแสง
- เนื่องจากมีพื้นที่รับแสงขนาดใหญ่จึงสามารถใช้ผลิตไฟฟ้าได้

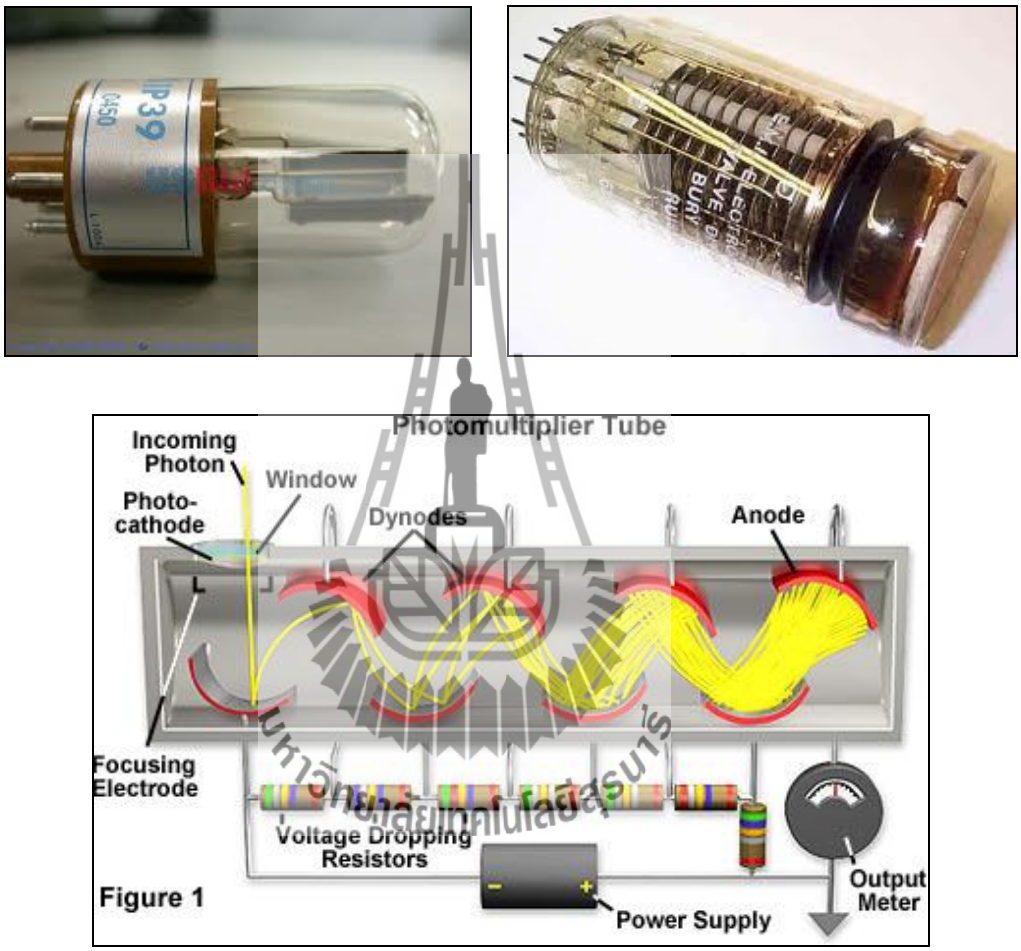


รูปที่ 2.23 แสดงตัวอย่างของ Photo Cell

Photo Tube& Photo Multiplier Tube

- Photo Tube อาศัยหลักการ Photoelectric Effect นั่นคือแสงจะชนให้อิเล็กตรอนที่ผิวของโลหะบางชนิดหลุดออกมาได้ อิเล็กตรอนดังกล่าวจะเคลื่อนที่ในสนามไฟฟ้าเกิดเป็นกระแสไฟฟ้าขึ้น กระแสไฟฟ้าที่ได้แปรผันโดยตรงกับความเข้มแสง โลหะดังกล่าวต้องอยู่ในหลอดสุญญากาศ

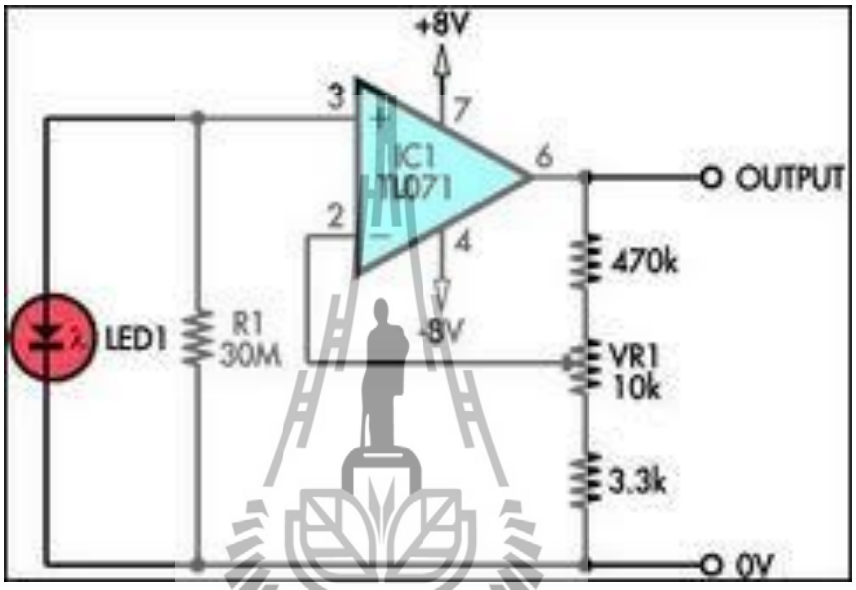
- PMT จะมีการเพิ่มจำนวนอิเล็กตรอนโดยทำการเร่งอิเล็กตรอนที่หลุดออกมาไปชนกับแผ่นโลหะชิ้นถัดๆ ไปซึ่งมีความต่างศักย์สูงขึ้นตามลำดับ ทำให้สามารถเพิ่มจำนวนอิเล็กตรอนขึ้นอย่างมหาศาล ทำให้วัดกระแสได้มากแม้จะมีแสงตกกระทบน้อยมาก (ให้ Sensitivity สูงมาก)



รูปที่ 2.24 แสดงตัวอย่างของ Photo Tube& Photo Multiplier Tube

LED as Light Sensor

- รอยต่อ PN ของ LED สามารถเปลี่ยนแสงเป็นกระแสไฟฟ้าได้ โดยมีความจำเพาะกับความยาวคลื่นแสงที่ LED ให้ออกมา กระแสไฟฟ้าจะแปรผันโดยตรงกับความเข้มแสง แต่จะได้กระแสน้อยเพราะมีพื้นที่รับแสงเล็กมาก
- มีการนำมาใช้ในเครื่องคัลเลอริมิเตอร์อย่างง่าย



รูปที่ 2.25 แสดงตัวอย่างวงจรของ LED as Light Sensor

บทที่ 3

ระบบเฝ้าระวังแปลงผักไฮโดรโพนิกส์ด้วยเซ็นเซอร์ไร้สายผ่านระบบ GSM

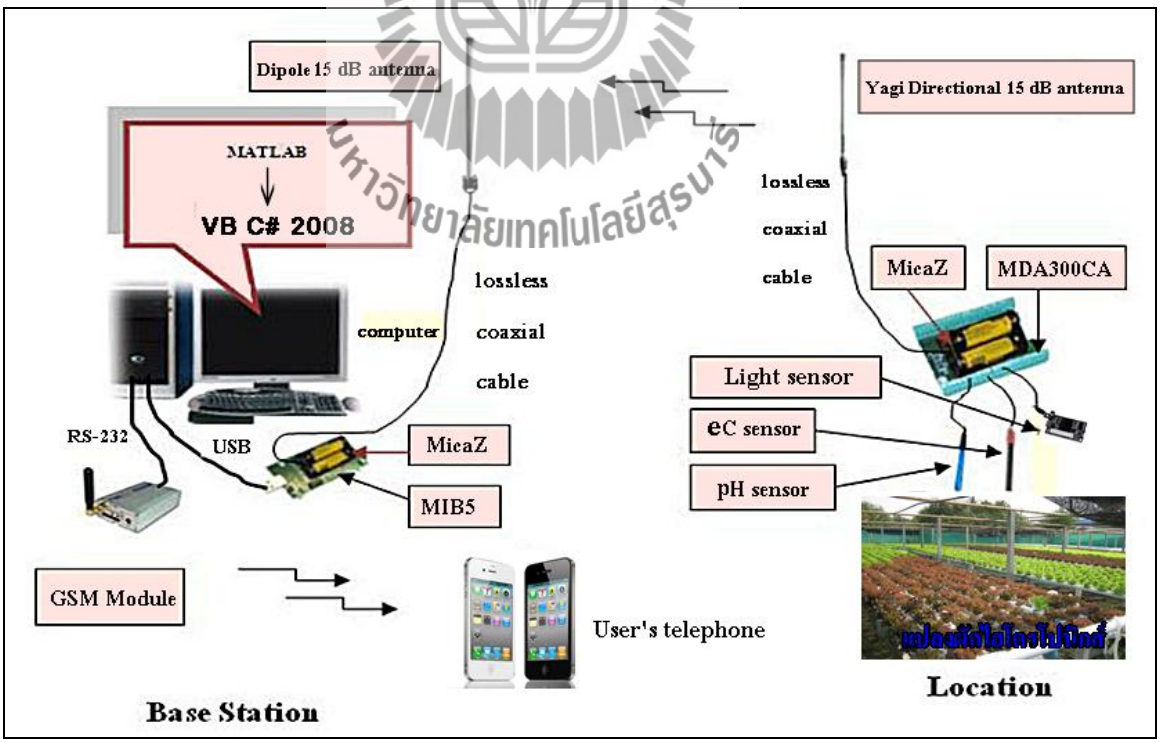
3.1 บทนำ

โครงงานเรื่อง **ระบบเฝ้าระวังแปลงผักไฮโดรโพนิกส์ด้วยเซ็นเซอร์ไร้สายผ่านระบบ GSM** นี้จัดทำขึ้นเพื่อประหยัดเวลาและแรงงานในการต้องใช้คนมาเก็บข้อมูลเพื่อที่จะควบคุมค่าต่างๆให้เหมาะสมอยู่ตลอดเวลาด้วยเซ็นเซอร์ไร้สายผ่านระบบ GSM ซึ่งเป็นการนำเทคโนโลยีไร้สายมาใช้ในด้านการเกษตร เพื่อให้เกิดการลดต้นทุนแต่เพิ่มผลผลิตทางการเกษตรให้กับเกษตรกร และเกิดการพัฒนาโปรแกรมสำหรับผู้จัดทำด้วย

3.2 รูปแบบระบบ

3.2.1 โครงสร้างระบบ

การทำงานของระบบจะ แบ่งออกเป็นสามส่วนหลัก ได้แก่ ระบบเครือข่ายเซ็นเซอร์ไร้สาย , ฐานข้อมูล และการติดต่อกับผู้ใช้งาน ซึ่งสามารถแสดงได้ดังรูปที่ 3.1



รูปที่ 3.1 โครงสร้างของระบบ

3.2.2 หลักการทำงานในแต่ละส่วนของระบบ

➢ **ระบบเครือข่ายเซ็นเซอร์ไร้สาย**

เครือข่ายเซ็นเซอร์ไร้สาย ประกอบด้วยอุปกรณ์ที่ทำหน้าที่เป็นโหนดเซ็นเซอร์ (Sensor Node) และโหนดสถานีฐาน (Base Station Node) โหนดเซ็นเซอร์ทำหน้าที่ในการส่งข้อมูลที่วัดจากเซ็นเซอร์ไปยังโหนดสถานีฐานผ่านทางคลื่น ความถี่วิทยุ ส่วนโหนดสถานีฐานจะทำหน้าที่ในการติดต่อสื่อสารระหว่างเครือข่ายเซ็นเซอร์ไร้สาย กับคอมพิวเตอร์ผ่านทางพอร์ตอนุกรม (Serial Port)

➢ **ฐานข้อมูล**

การเก็บข้อมูลจะทำผ่านโปรแกรม MATLAB R2009a โดยการรัน Code โปรแกรม อ่านค่าจาก Serial Port ที่ต่อกับโหนดสถานีฐาน ซึ่งเป็นค่าของแรงดันไฟฟ้า (ADC) จึงต้องทำการแปลงค่าจากแรงดันไฟฟ้า (ADC) ไปเป็นค่าของความเข้มแสง (Light), ค่าความเป็นกรด -ด่าง (pH) และความนำไฟฟ้าในสารละลาย (eC) ที่แท้จริง โดย Calibrate ด้วยสมการเฉพาะของข้อมูลแต่ละชนิด เมื่อได้ค่าที่แท้จริงแล้ว โปรแกรมจะทำการเก็บค่าไว้ในไฟล์ “Result.dat”

➢ **การติดต่อกับผู้ใช้งาน** มีดังนี้

1) โปรแกรมส่ง SMS จะทำการเช็คไฟล์ “Result.dat” เมื่อมีข้อมูลล่าสุดเข้ามา โปรแกรมจะทำการอ่านข้อมูลในไฟล์ แล้วถ้าเกิดข้อมูลที่ผิดพลาดจะส่ง SMS ไปให้ User ผ่านทาง GSM Module เข้าโทรศัพท์มือถือ ซึ่งรายละเอียดของการออกแบบโปรแกรมอธิบายได้ดังต่อไปนี้

3.2.3 หน้าที่ของอุปกรณ์ในแต่ละส่วนของระบบ

➢ **ระบบเครือข่ายเซ็นเซอร์ไร้สาย**

- pH Sensor, Light Sensor และ eC Sensor ทำหน้าที่ในการวัดค่าความเป็นกรด -ด่าง (pH) ,ความเข้มแสง (Light) และความนำไฟฟ้าในสารละลาย (eC) ตามลำดับค่าที่ได้อยู่ในรูปของแรงดันไฟฟ้า

- MDA300CA ทำหน้าที่เป็นตัวอินเตอร์เฟซระหว่าง pH Sensor, Light Sensor และ eC Sensor กับ MicaZ(ตัวส่ง)

- MicaZ (ตัวส่ง) คือโหมดเซ็นเซอร์ ทำหน้าที่ในการส่งค่าที่วัดได้จากเซ็นเซอร์ไปยังโหนดสถานีฐาน

- Lossless Coaxial Cable ทำหน้าที่ในการนำส่งสัญญาณไฟฟ้าระหว่าง MicaZ กับสายอากาศ

- 8dB Dipole Antenna คือ สายอากาศทางด้าน รับ ทำหน้าที่ ในการ แปลงสัญญาณไฟฟ้าไปเป็นสัญญาณคลื่นความถี่วิทยุส่งไปในอวกาศ

- 15dB Yagi Directional Antenna คือ สายอากาศทางด้านส่ง ทำหน้าที่แปลงสัญญาณคลื่นความถี่วิทยุไปเป็นสัญญาณไฟฟ้า

- MicaZ (ตัวรับ) คือโหมดเซ็นเซอร์ ทำหน้าที่ในการรับค่าที่วัดได้จากเซ็นเซอร์จากโหนดเซ็นเซอร์

- MIB5 ทำหน้าที่เป็นตัวอินเตอร์เฟซระหว่าง MicaZ (ตัวรับ) กับ คอมพิวเตอร์ผ่านพอร์ตอนุกรม

- USB ทำหน้าที่ในการเชื่อมต่อพอร์ทอนุกรมระหว่างคอมพิวเตอร์กับ MIB5

- **ฐานข้อมูล**

ด้านซอฟต์แวร์

- โปรแกรม MATLAB R2009a ทำหน้าที่ในการแปลงค่าแรงดันไฟฟ้าที่วัดได้จากเซ็นเซอร์แต่ละชนิด เป็นค่าจริงของข้อมูลชนิดนั้นๆ แล้วเก็บไว้ในไฟล์ “Result.dat”

ด้านฮาร์ดแวร์

- คอมพิวเตอร์ ทำหน้าที่เป็น Server

- **การติดต่อกับผู้ใช้งาน**

ด้านซอฟต์แวร์

- โปรแกรมส่ง SMS ทำหน้าที่ในการเข้าไปอ่านข้อมูลใหม่จากไฟล์ “Result.dat” แล้วเมื่อเกิดค่าที่ผิดพลาดจะส่ง SMS ไปให้ User

ด้านฮาร์ดแวร์

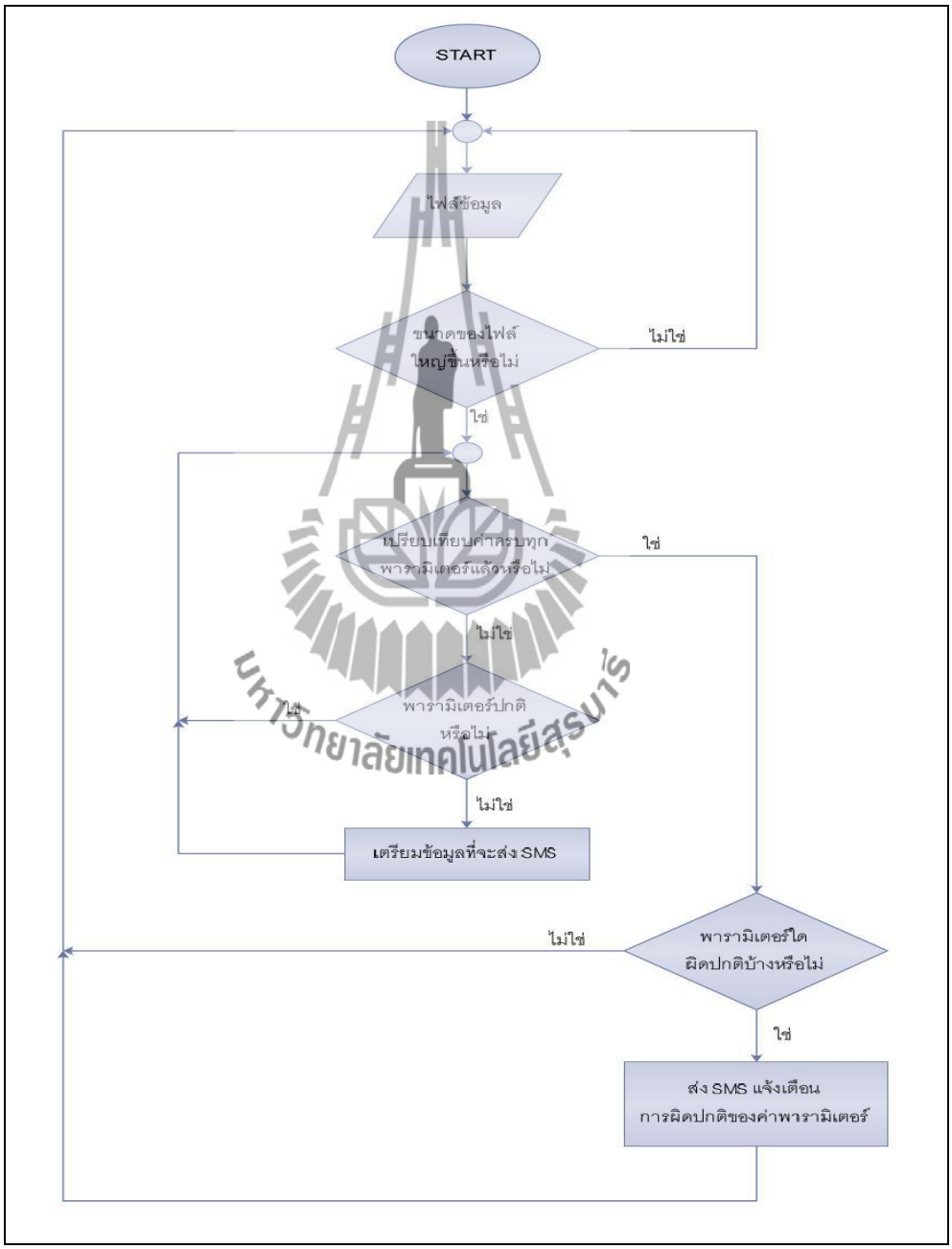
- RS-232 ทำหน้าที่ในการเชื่อมต่อระหว่าง GSM module กับ คอมพิวเตอร์

- GSM module ทำหน้าที่ในการรับคำส่งจากโปรแกรมส่ง SMS ให้ส่ง SMS ไปให้ User ผ่านเครือข่าย GSM

3.3 การสร้าง code โปรแกรมเพื่อส่ง SMS

3.3.1 การทำงานของโปรแกรม

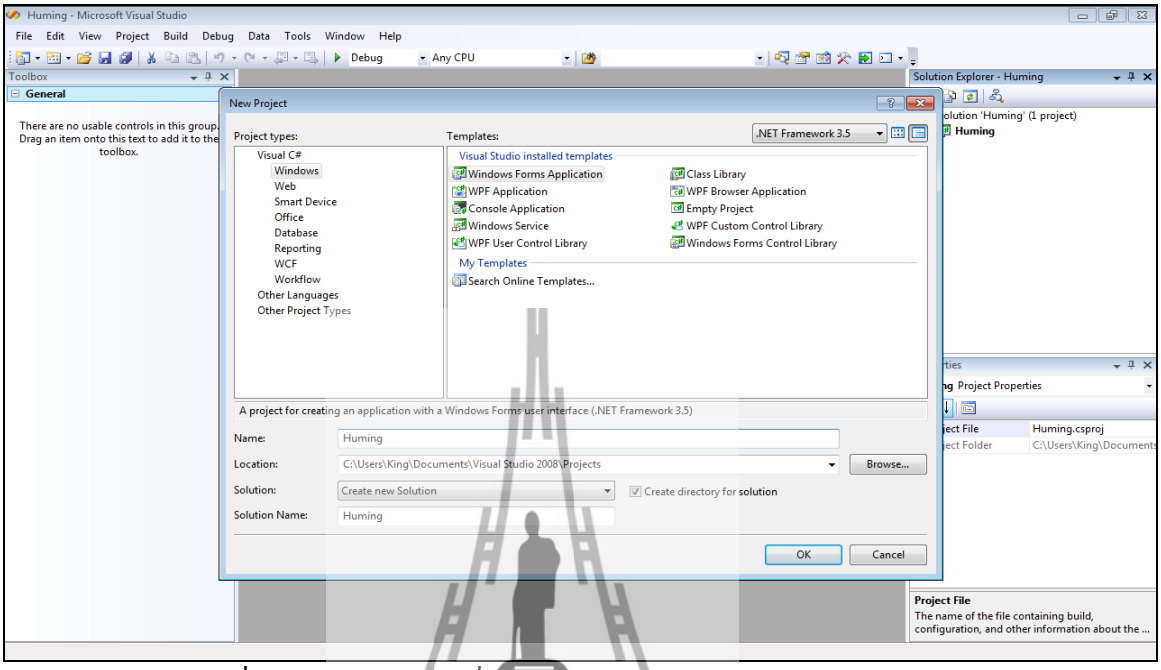
การทำงานของโปรแกรม คือ คอยตรวจเช็คข้อมูลใหม่ที่ส่งจากเซ็นเซอร์ไร้สายมายังสถานีฐาน แล้วจะทำการส่ง SMS ไปยังโทรศัพท์มือถือ เมื่อเกิดค่าที่ผิดพลาดขึ้น ซึ่งสามารถอธิบายแผนภาพได้ดังรูปที่ 3.2 ส่วนรายละเอียดของ Code โปรแกรมจะได้อธิบายในขั้นตอน ถัดไป



รูปที่ 3.2 Flowchart การทำงานของโปรแกรมแจ้งเตือนและส่ง SMS

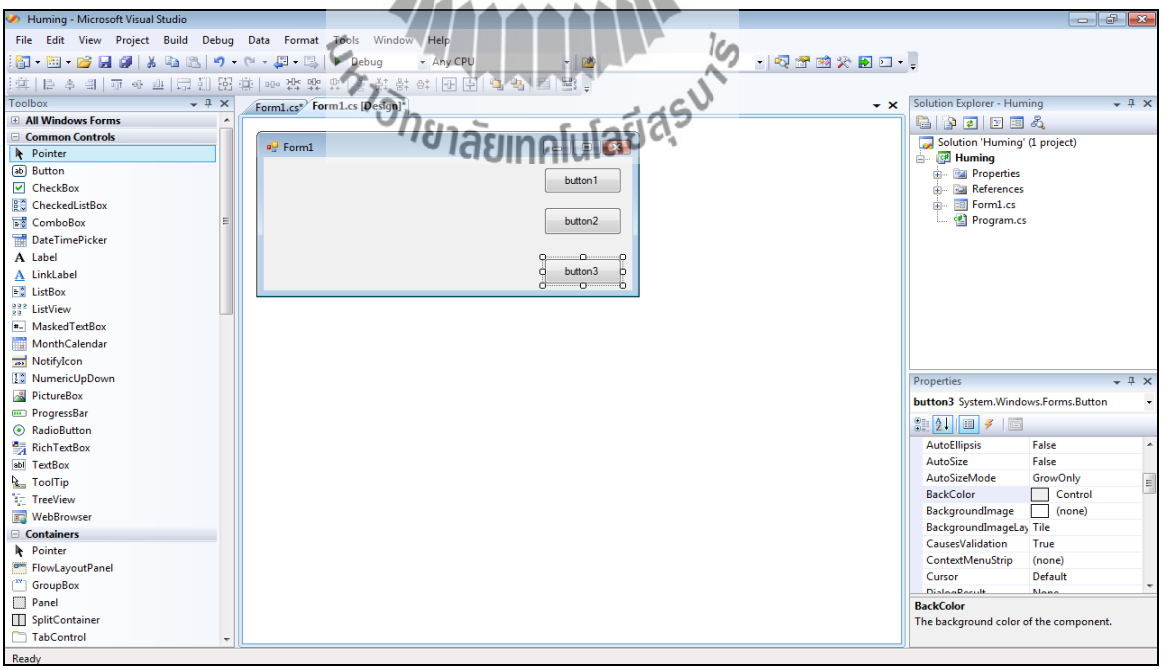
3.3.2 ขั้นตอนการออกแบบ

1) เปิดโปรแกรม Microsoft Visual C# 2008 ขึ้นมา จากนั้นไปที่ File > New > Project เลือก Windows Forms Application ตั้งชื่อโปรเจ็คใหม่ว่า “Huming” จากนั้น Click OK ด้านล่าง



รูปที่ 3.3 แสดงหน้าต่างเมื่อเปิดโปรแกรม Microsoft Visual C# 2008

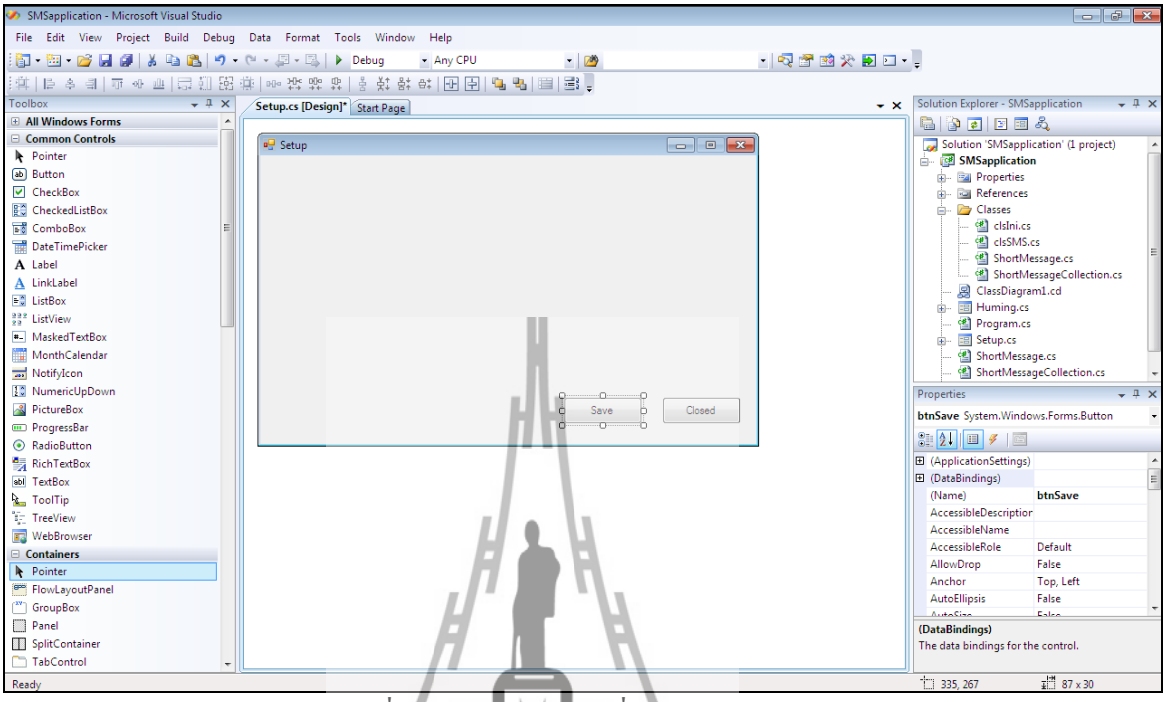
2) เลือก Botton เพื่อออกแบบกล่องข้อมูล โดยเลือกมา 3 Botton โดย Botton1 คือ Start, Botton2 คือ Setting, Botton3 คือ Closed



รูปที่ 3.4 แสดงหน้าต่างเมื่อเลือก Botton เพื่อออกแบบกล่องข้อมูล

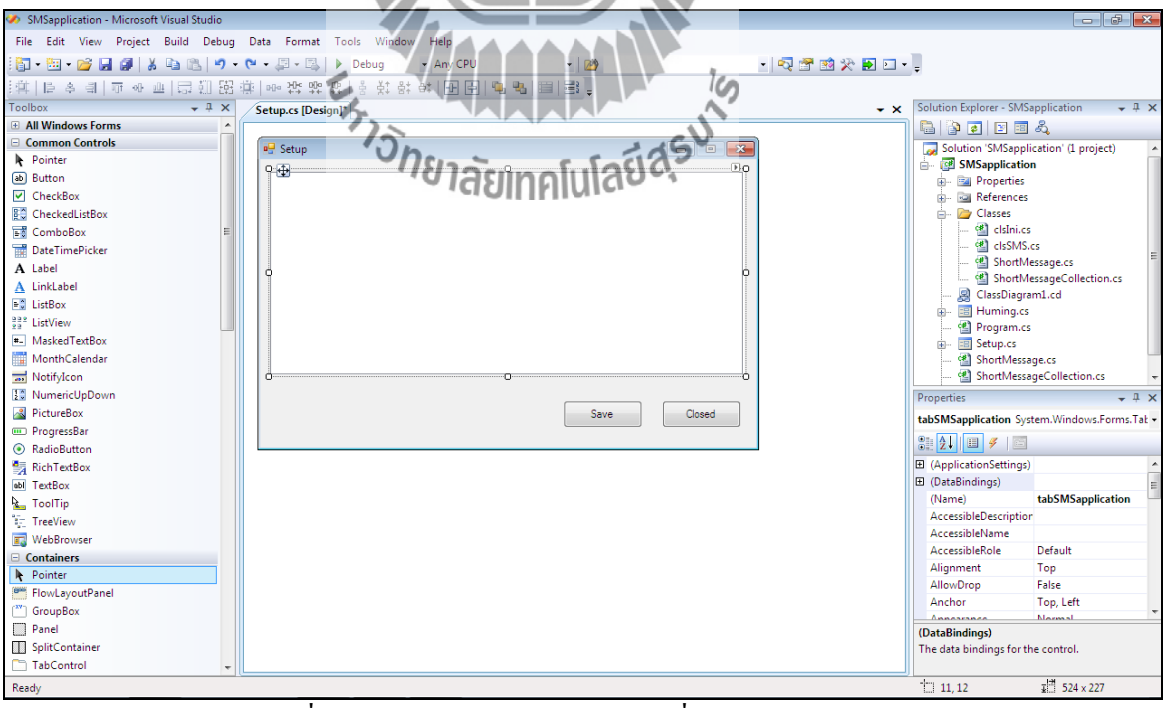
2.1) สร้างกล่องข้อมูลสำหรับกำหนดค่าพารามิเตอร์ต่างๆ เมื่อเลือกปุ่ม Setting

2.1.1) สร้างกล่องข้อมูล โดยเลือก Button มา 2 Button เพื่อสร้างปุ่ม Save และ Closed



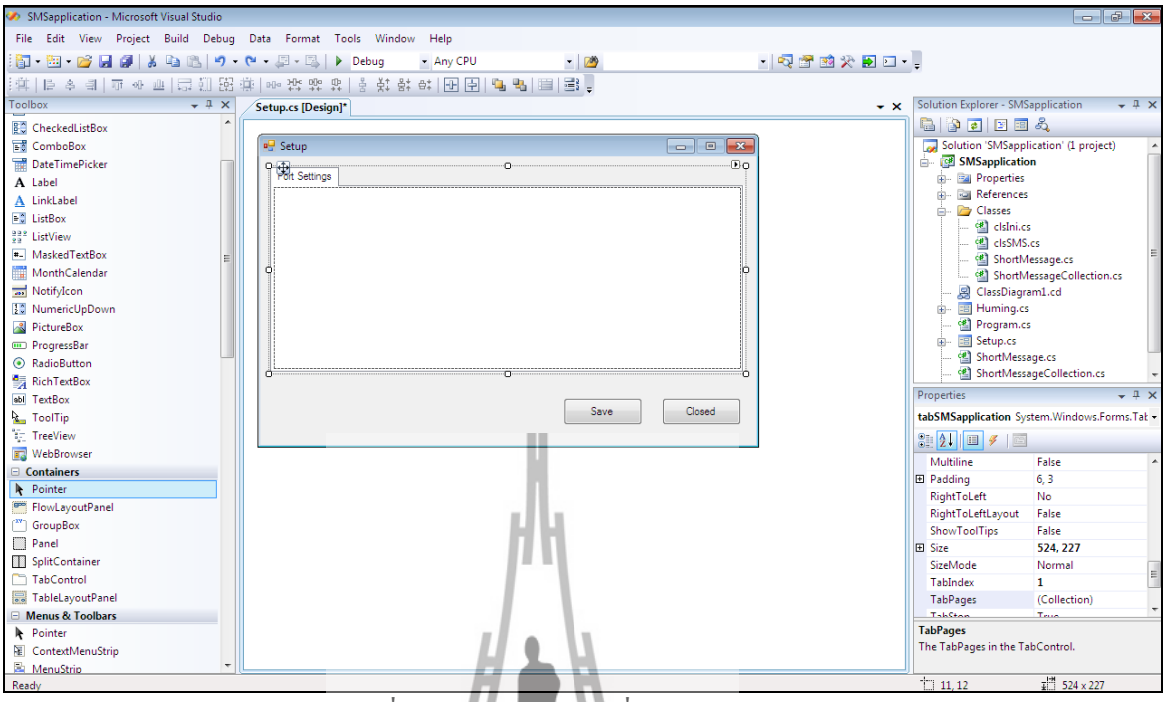
รูปที่ 3.5 แสดงหน้าต่างเมื่อเลือกใช้ Button

2.1.2) สร้างกล่องข้อมูลโดยเลือก Tabcontrol



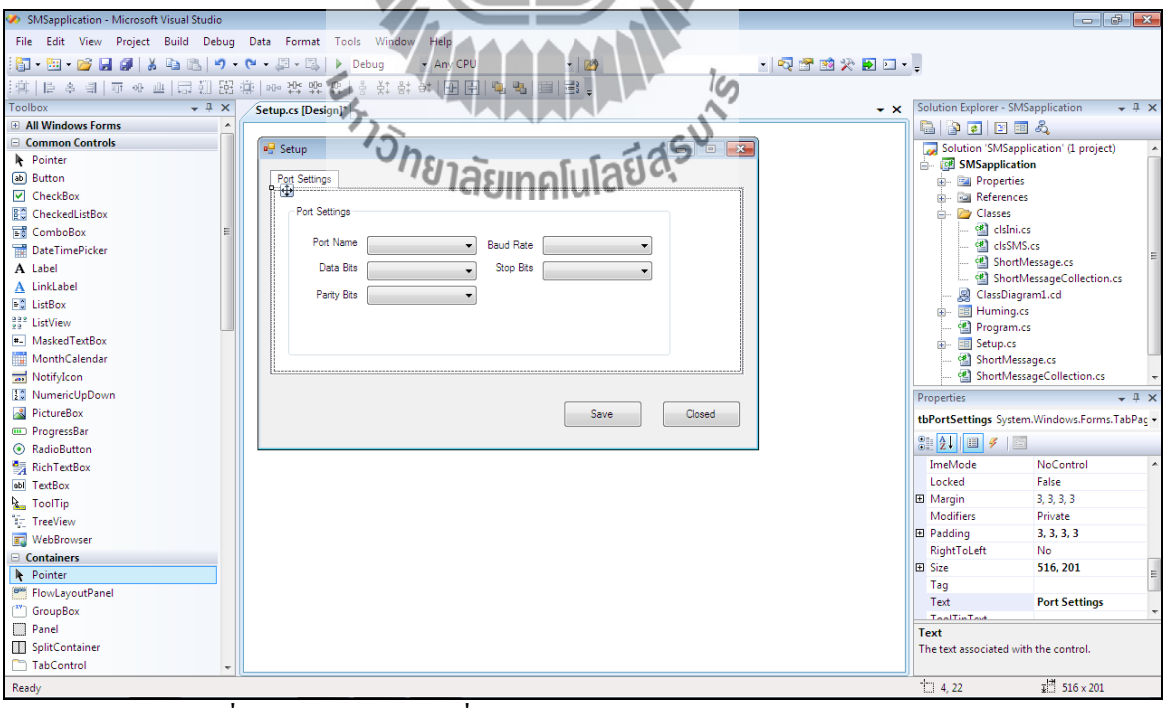
รูปที่ 3.6 แสดงหน้าต่างกล่องข้อมูลเมื่อเลือก Tabcontrol

2.1.3) เลือก Add Tab เพื่อสร้างหน้าต่างข้อมูล Port Settings



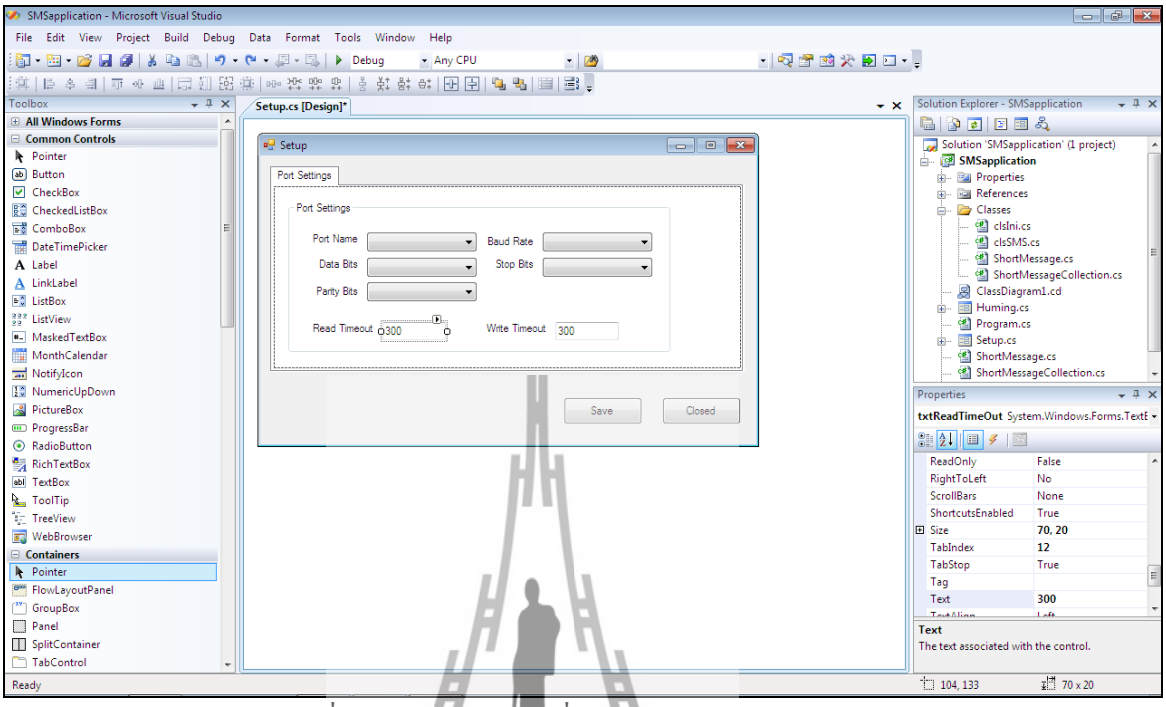
รูปที่ 3.7 แสดงหน้าต่างเมื่อเลือก Add Tab

2.1.4) เลือก GroupBox, Label และ ComboBox เพื่อสร้างช่องเลือกค่าต่างๆดังรูปข้างล่าง



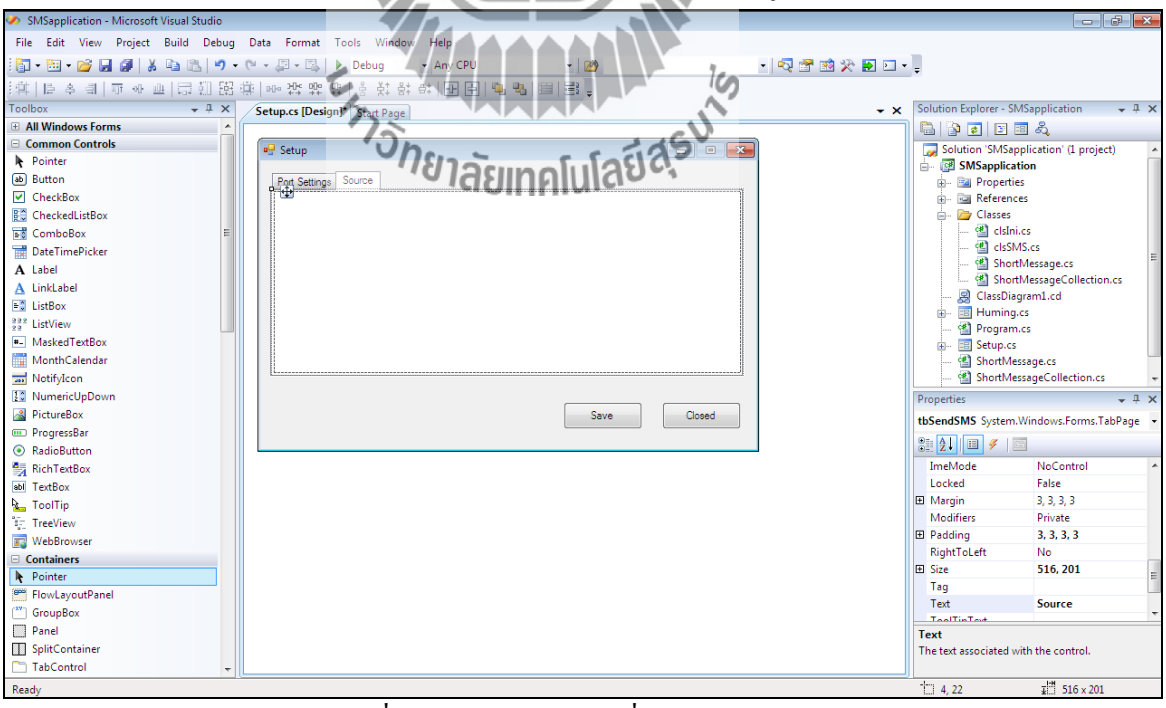
รูปที่ 3.8 แสดงหน้าต่างเมื่อเลือกใช้ GroupBox, Label และ ComboBox

2.1.5) เลือก Label และ Textbox เพื่อใช้กำหนดค่า Read Timeout และ Write Timeout



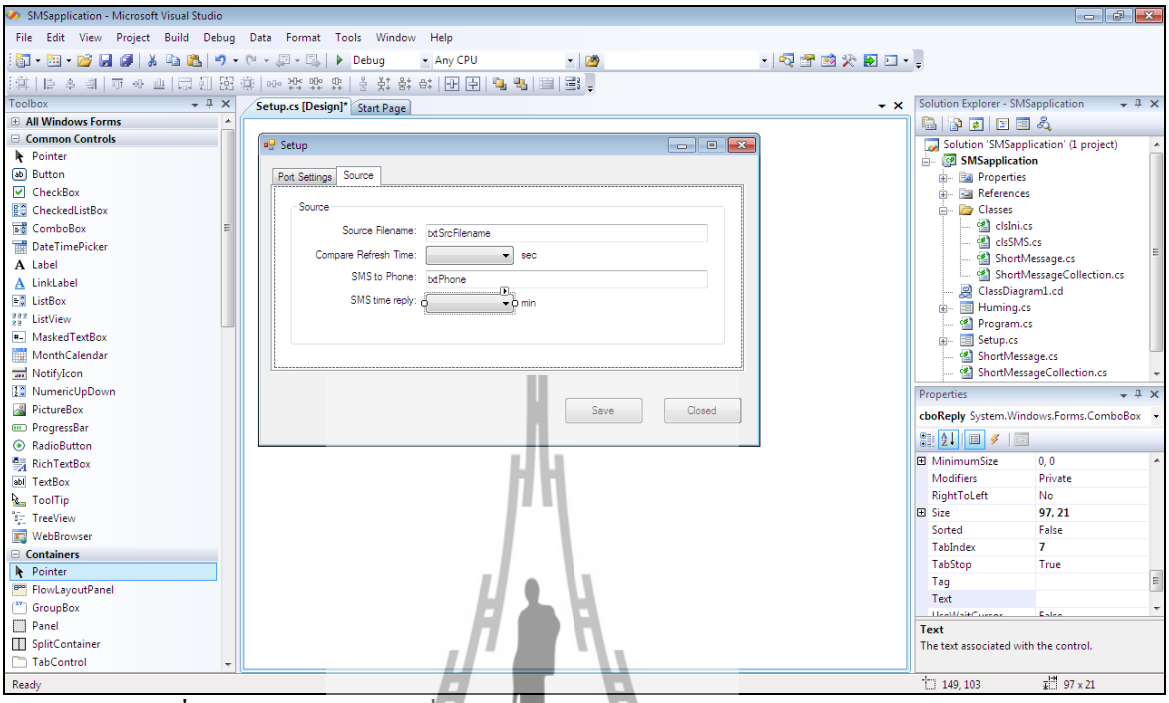
รูปที่ 3.9 แสดงหน้าต่างเมื่อเลือก Label และ Textbox

2.1.6) เลือก Add Tab เพื่อสร้างหน้าต่างข้อมูล Source



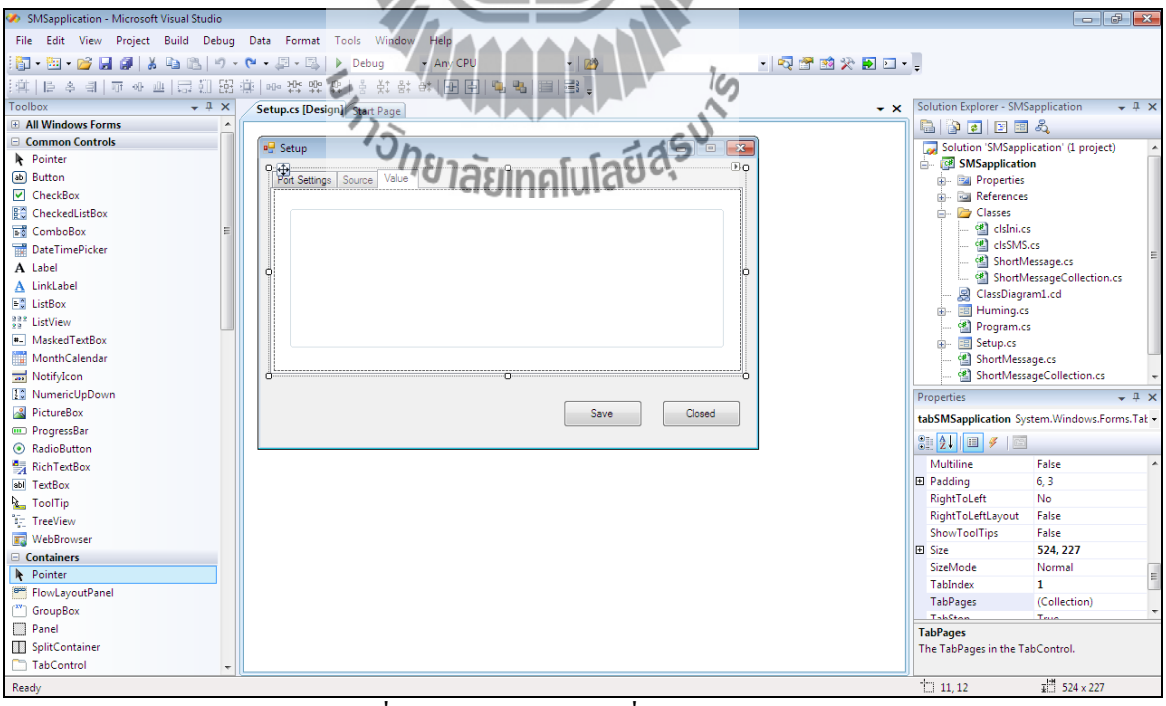
รูปที่ 3.10 แสดงหน้าต่างเมื่อเลือก Add Tab

2.1.7) เลือก GroupBox, Label, TextBox และ ComboBox เพื่อสร้างช่องเลือกค่าต่างๆ



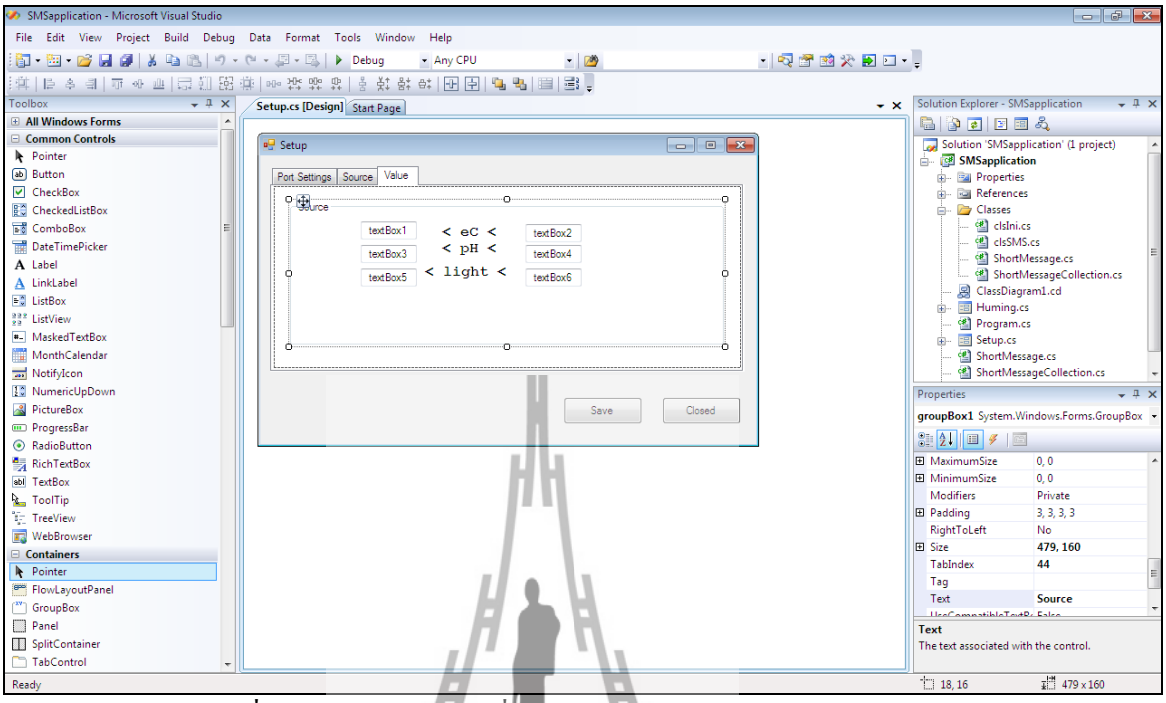
รูปที่ 3.11 แสดงหน้าต่างเมื่อเลือก GroupBox, Label, TextBox และ ComboBox

2.1.8) เลือก Add Tab เพื่อสร้างหน้าต่างข้อมูล Value



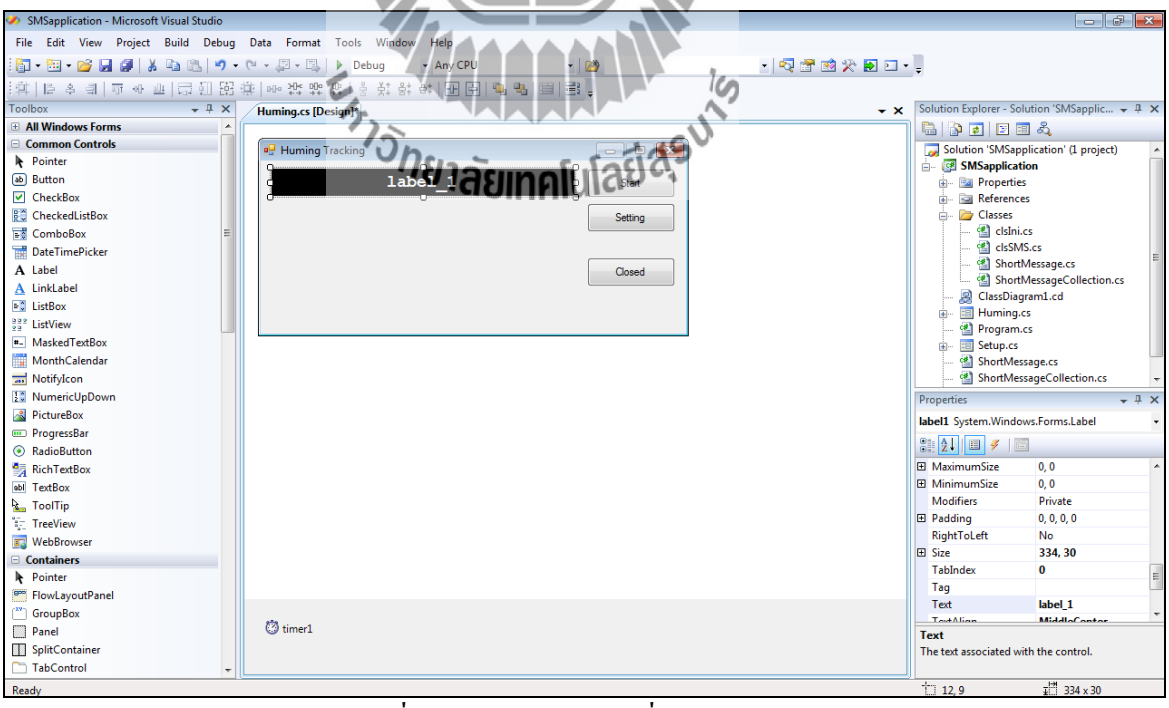
รูปที่ 3.12 แสดงหน้าต่างเมื่อเลือก Add Tab

2.1.9) เลือก GroupBox, Label และ TextBox เพื่อสร้างช่องเลือกค่าเพื่อเปรียบเทียบ



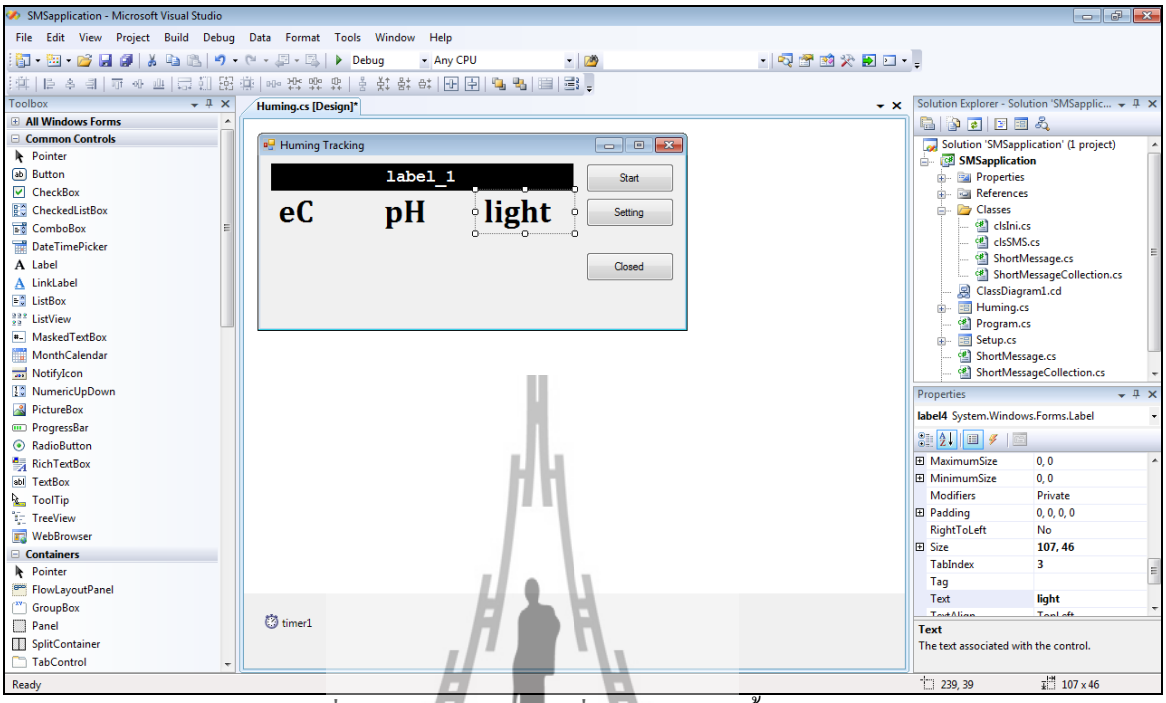
รูปที่ 3.13 แสดงหน้าต่างเมื่อเลือก GroupBox, Label และ TextBox

3) สร้าง Label _1 โดยใช้คำสั่ง Label เพื่อสร้างช่องแสดงเวลา



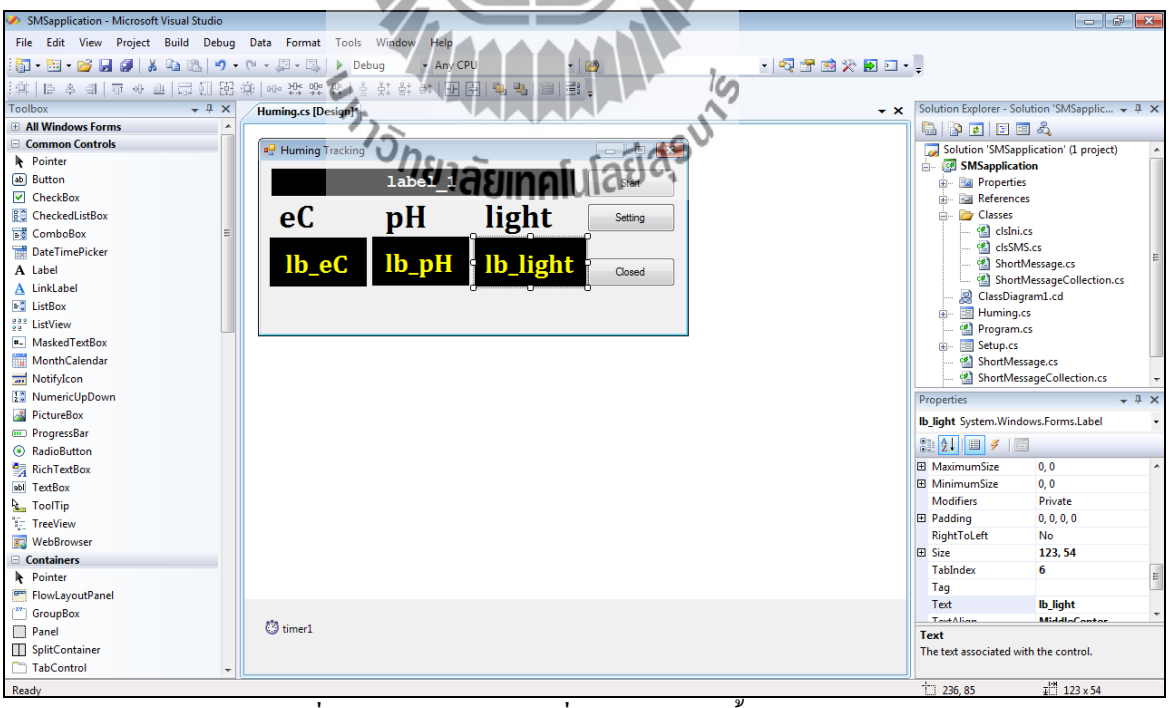
รูปที่ 3.14 แสดงหน้าต่างเมื่อเลือก Label

4) สร้าง Label ขึ้น 3 Label คือ ความนำไฟฟ้าในสารละลาย (eC) ค่าความเป็นกรด-ด่าง (pH) และความเข้มแสง (light)



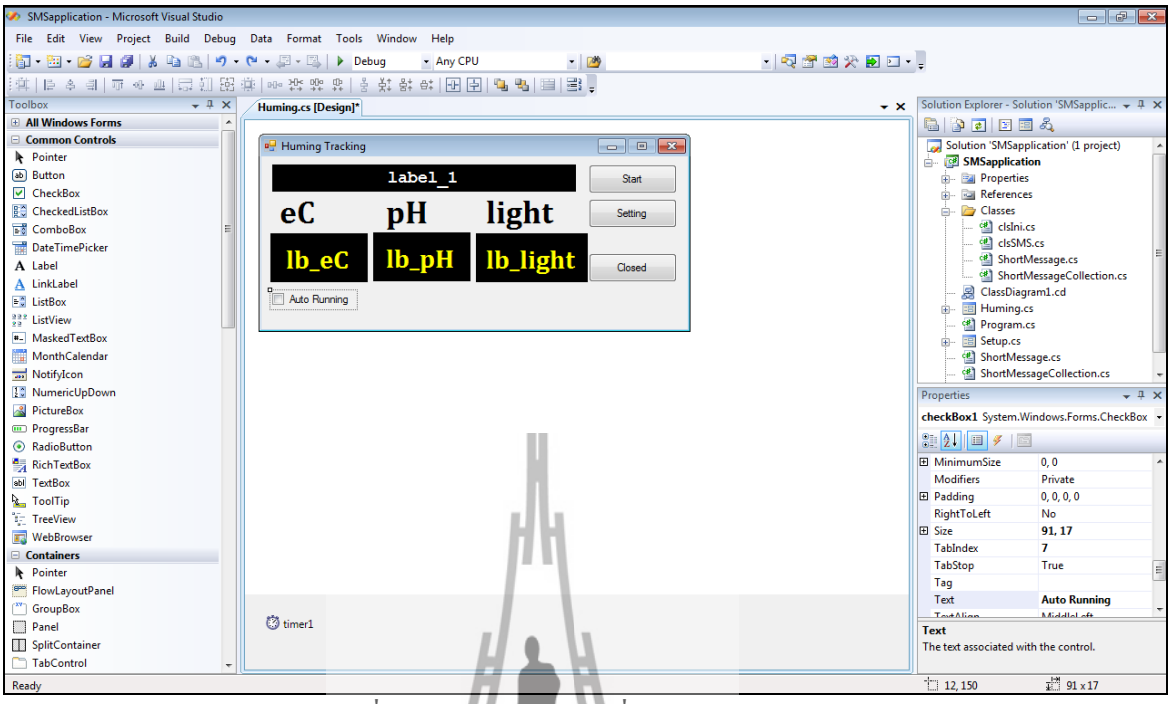
รูปที่ 3.15 แสดงหน้าต่างเมื่อเลือก Label ขึ้น 3 Label

5) สร้าง Label ขึ้นอีก 3 Label เพื่อดึงค่าความนำไฟฟ้าในสารละลาย (eC) ค่าความเป็นกรด-ด่าง (pH) และความเข้มแสง (Light) มาโชว์ในช่องดังกล่าว



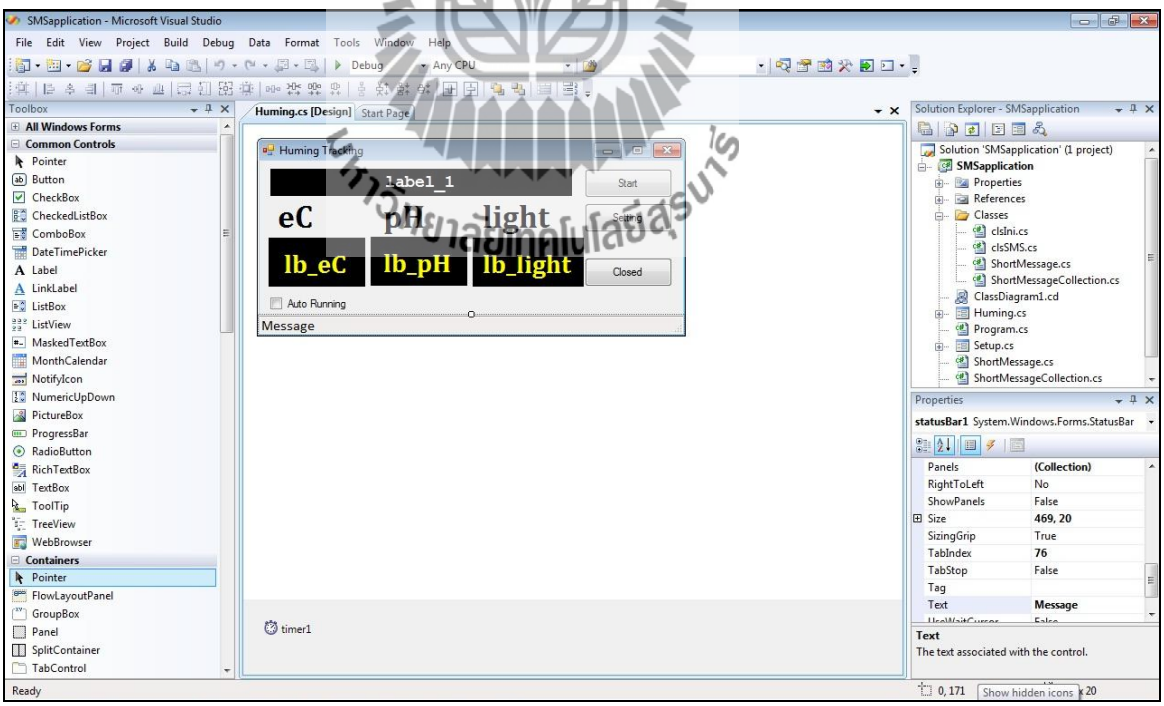
รูปที่ 3.16 แสดงหน้าต่างเมื่อเลือก Label ขึ้นอีก 3 Label

6) เลือก CheckBox เพื่อใช้เลือกว่าจะให้โปรแกรมเป็นแบบอัตโนมัติหรือไม่



รูปที่ 3.17 แสดงหน้าต่างเมื่อเลือก CheckBox

7) เลือก StatusBar เพื่อให้แสดงผลเมื่อ SMS ส่ง



รูปที่ 3.18 แสดงหน้าต่างเมื่อเลือก StatusBar

บทที่ 4

ผลการทดสอบระบบ

4.1 บทนำ

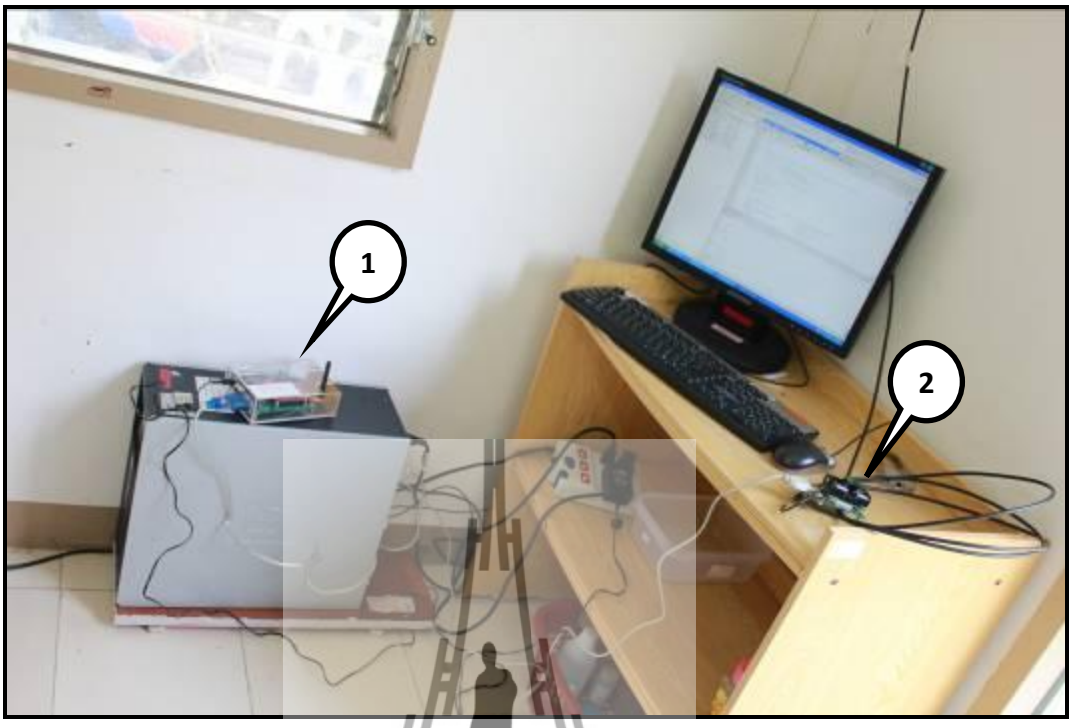
จากการศึกษาและทำความเข้าใจเกี่ยวกับทฤษฎีพื้นฐานในบทที่ 2 และ 3 นั้น ทำให้สามารถออกแบบโปรแกรมที่เสร็จสมบูรณ์พร้อมที่จะนำไปทดสอบการใช้งานจริง เพื่อให้บรรลุวัตถุประสงค์ของโครงงาน

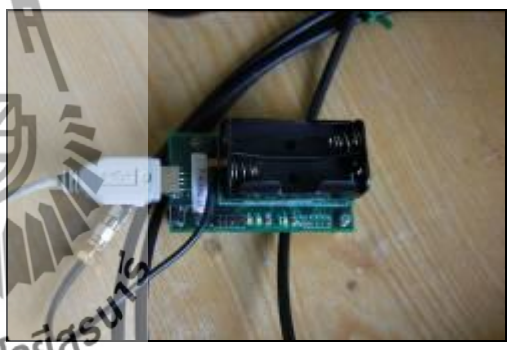
4.2 โครงสร้างระบบทั้งหมดในการติดตั้ง

Location Base Station

รูปที่ 4.1 แผนที่แสดงบริเวณที่ติดตั้ง ณ แปลงผักไฮโดรโพนิกส์ มหาวิทยาลัยเทคโนโลยีสุรนารี

4.2.1 ส่วนของภาครับ (สถานีฐาน)

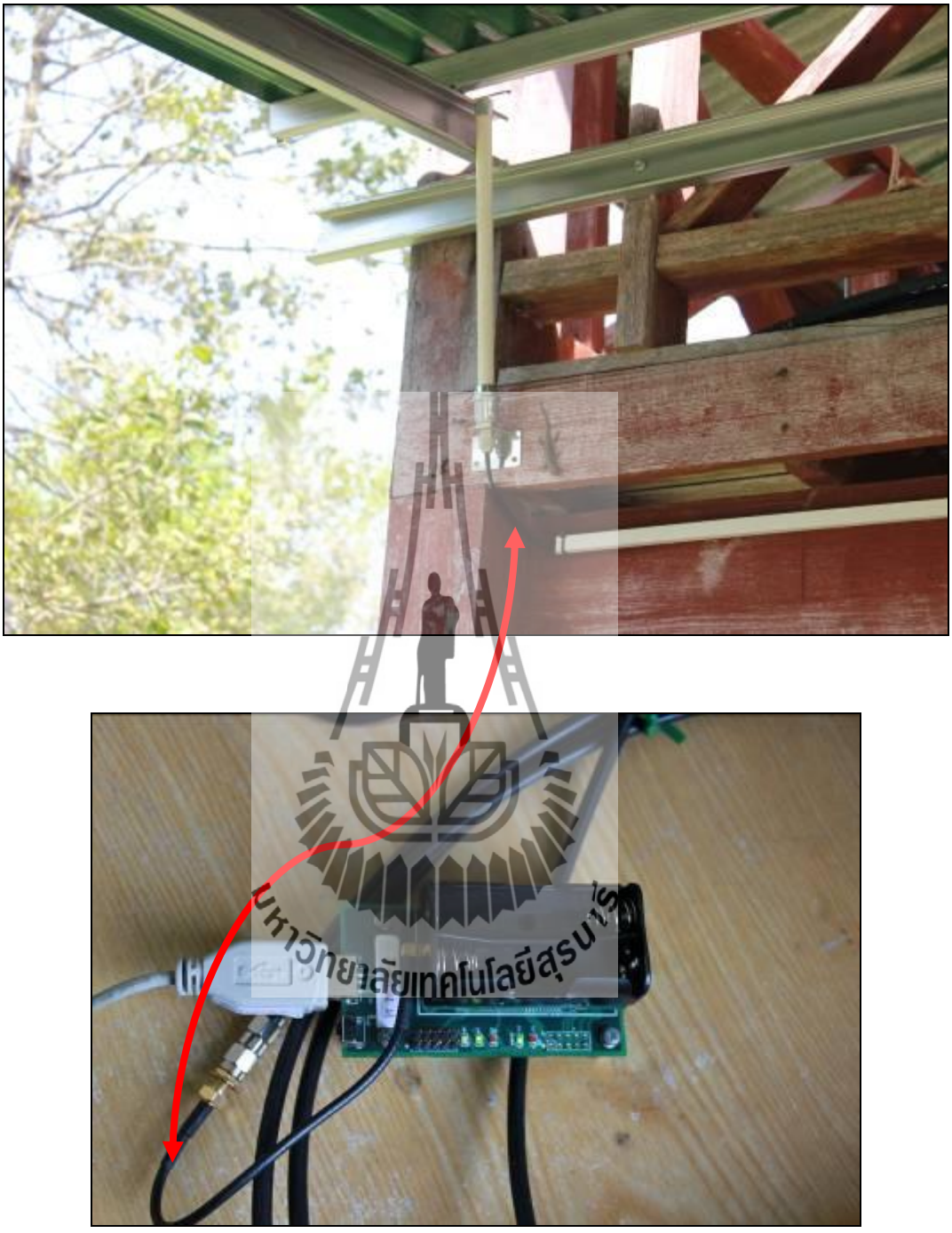




หมายเลข 1 บอร์ด GSM Module Wireless CPU

หมายเลข 2 ตัวรับสัญญาณ

รูปที่ 4.2 แสดงส่วนประกอบโดยรวมของสถานีฐาน



รูปที่ 4.3 การเชื่อมต่อระหว่างสายอากาศกับโหมดของภาครับ (สถานีฐาน)

4.2.2 ส่วนของภาคส่ง

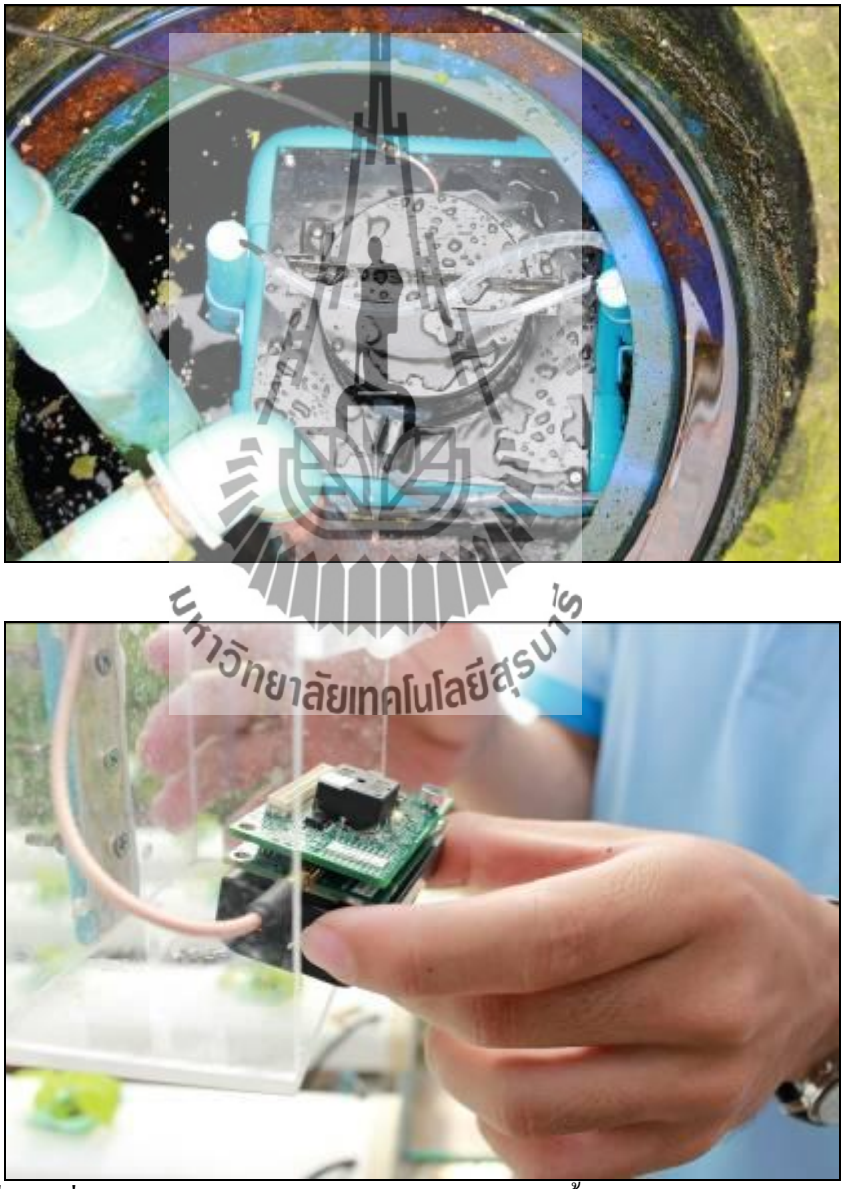
Light Sensor

หมายเลข 1 (pH Sensor)

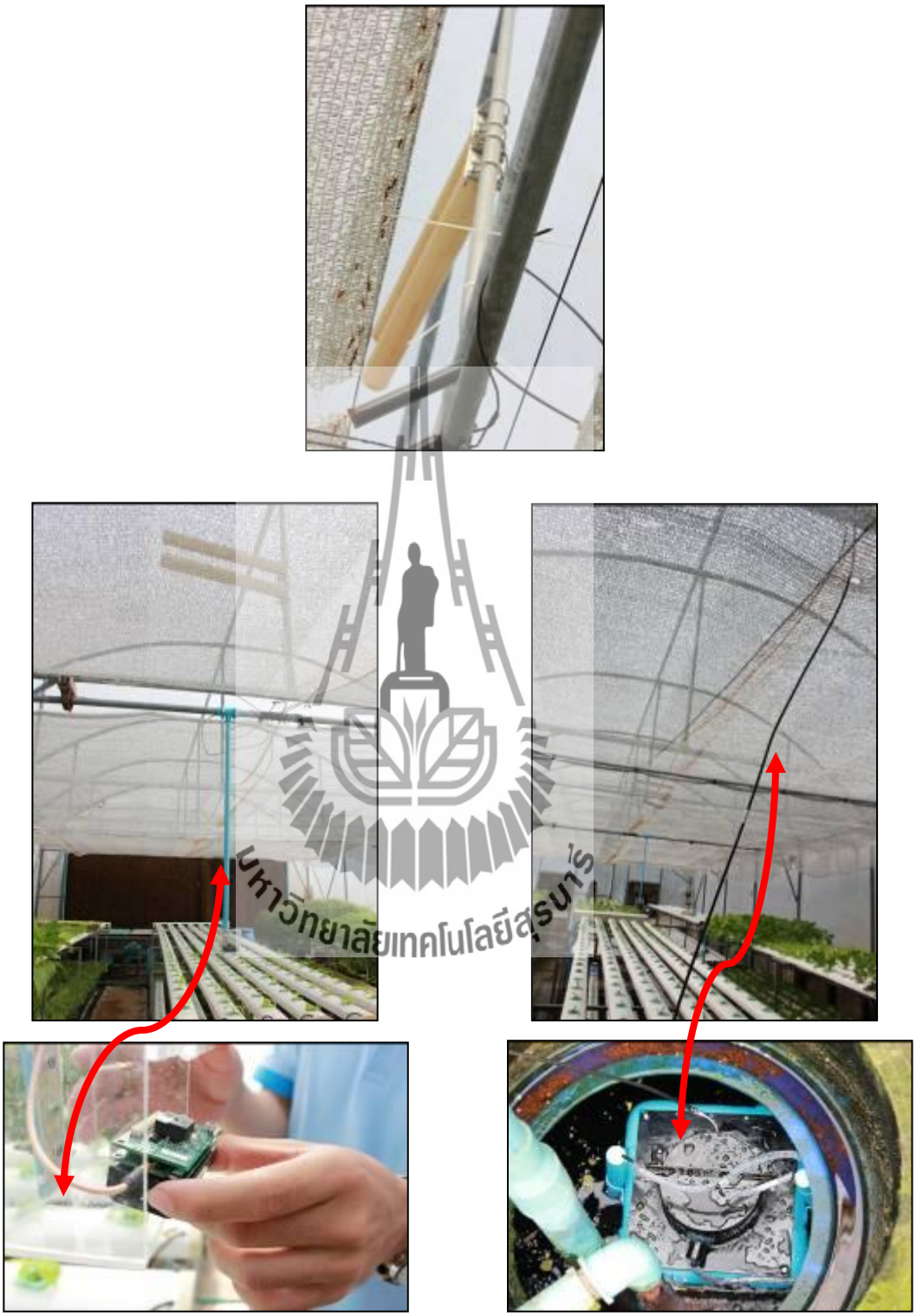
หมายเลข 2 (eC Sensor)

รูปที่ 4.4 อุปกรณ์ Sensor แต่ละชนิด

รูปที่ 4.5 ส่วนประกอบวงจรภายในของทุ่นลอย



รูปที่ 4.6 เมื่อนำทุ่นลอยไปวางในบ่อสารละลายและติดตั้งเซนเซอร์แสงเรียบร้อยแล้ว



รูปที่ 4.7 เชื่อมต่อระหว่างโหมดภาคส่งกับสายอากาศ

4.2.3 คอมพิวเตอร์ที่ใช้ในการติดตั้ง

1. PC: HP Compaq รุ่น dc5750 Microtower Base DT PC

- Processor: AMD Athlon(tm) 64X2 Dual Core Processor 5200+ 2.59 GHz
- RAM: 1.87 GB
- Harddisk: 80 GB 7200RPM SATA Hard Drive
- CD-ROM: DVD-RW รุ่น HL-DT-ST DVD+-RW GSA-H31L
- VGA: ATI Radeon Xpress 1150 Secondary
- Monitor: Samsung 19 นิ้ว รุ่น SyncMaster 940N
- Keyboard: HP
- Mouse: Logitech-compatible Mouse PS/2

4.2.4 โปรแกรมที่ใช้ในการติดตั้ง

1. โปรแกรม MATLAB R2009a
2. โปรแกรม Microsoft Visual C# 2008

4.2.5 ฮาร์ดแวร์ที่ใช้ในการติดตั้ง

1. บอร์ด GSM Module Wireless CPU 1 บอร์ด
2. MDA 300CA 1 บอร์ด
3. MicaZ 2 บอร์ด
4. MIB 520CB 1 บอร์ด

4.2.6 สายนำส่งสัญญาณที่ใช้ในการติดตั้ง

1. Lossless Coaxial Cable 50 Ω 2 เส้น

4.2.7 สายแปลงสัญญาณที่ใช้ในการติดตั้ง

1. USB Serial Port
2. RS-232

4.2.8 เซ็นเซอร์ที่ใช้ในการติดตั้ง

1. pH Sensor (PH-BTA)
2. eC Sensor (CON-BTA)
3. Light Sensor (MTS 300)

4.2.9 สายอากาศที่ใช้ในการติดตั้ง

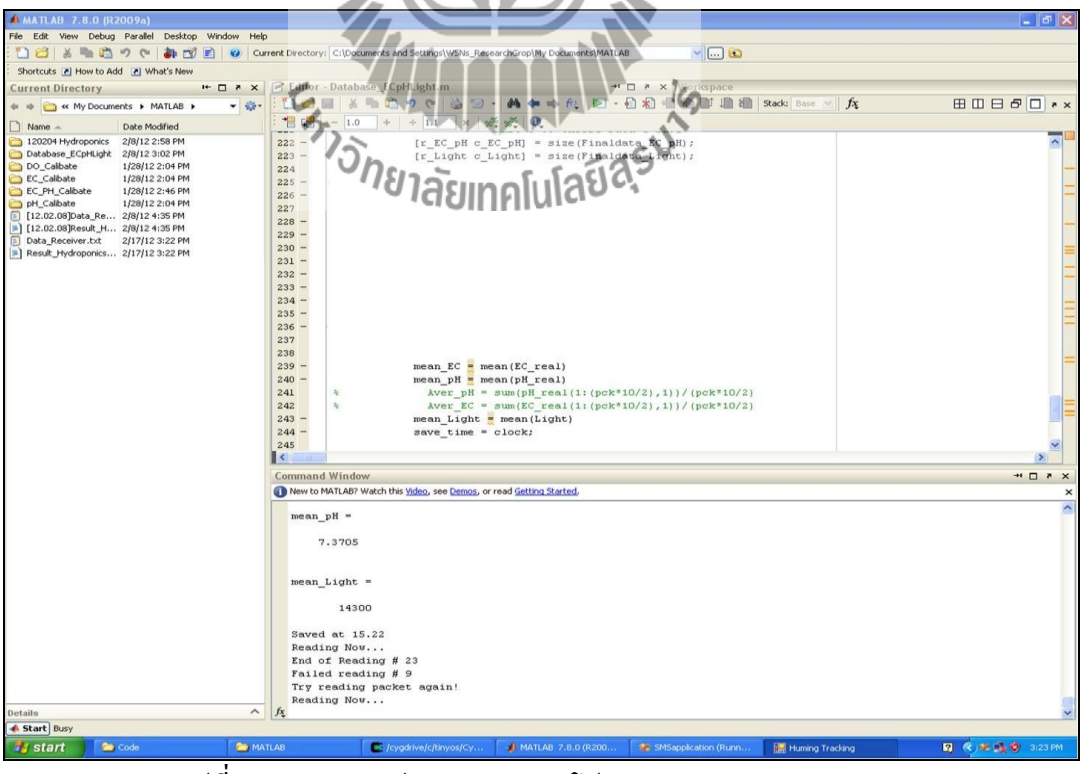
1. สายอากาศไดโพล 8 dB
2. สายอากาศไดเร็กชันเนล ยากิ 15 dB

4.3 ขั้นตอนการทดสอบโปรแกรมส่ง SMS

1) เชื่อมต่อ Module เข้ากับคอมพิวเตอร์ ผ่านทาง Serial Port

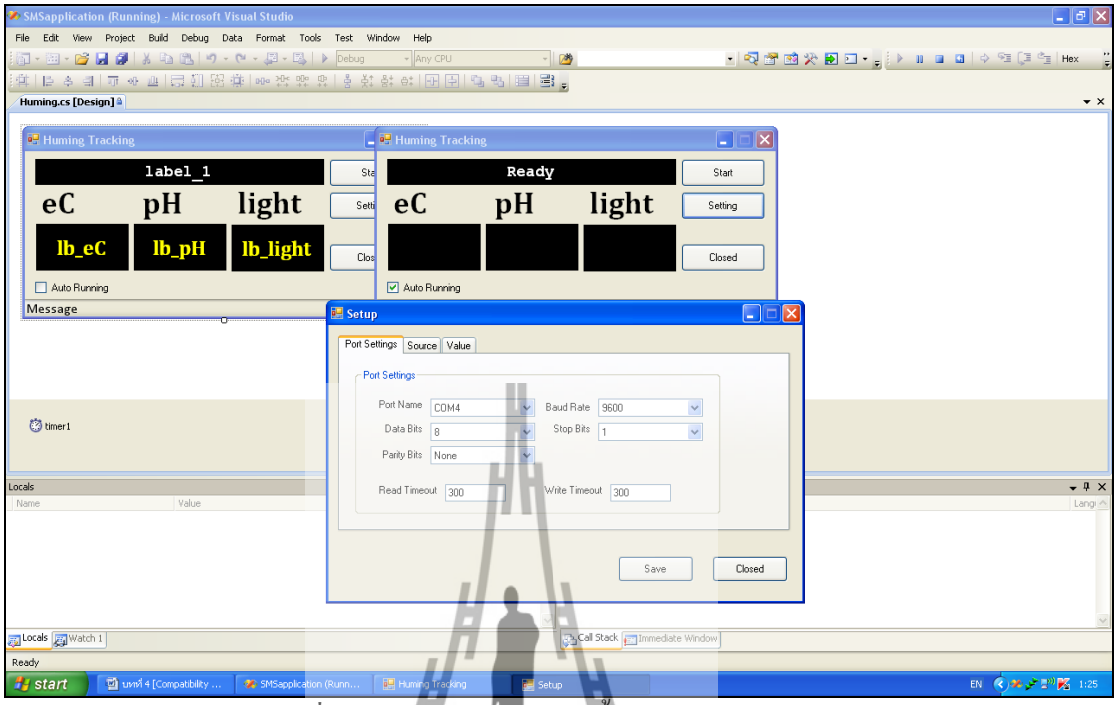
รูปที่ 4.8 แสดงการเชื่อมต่อ GSM Module Wireless CPU กับ Port RS-232

2) เปิดโปรแกรม MATLAB R2009a ขึ้นมา แล้วทำการรันโปรแกรมเพื่อเก็บข้อมูลจากเซ็นเซอร์ไร้สาย



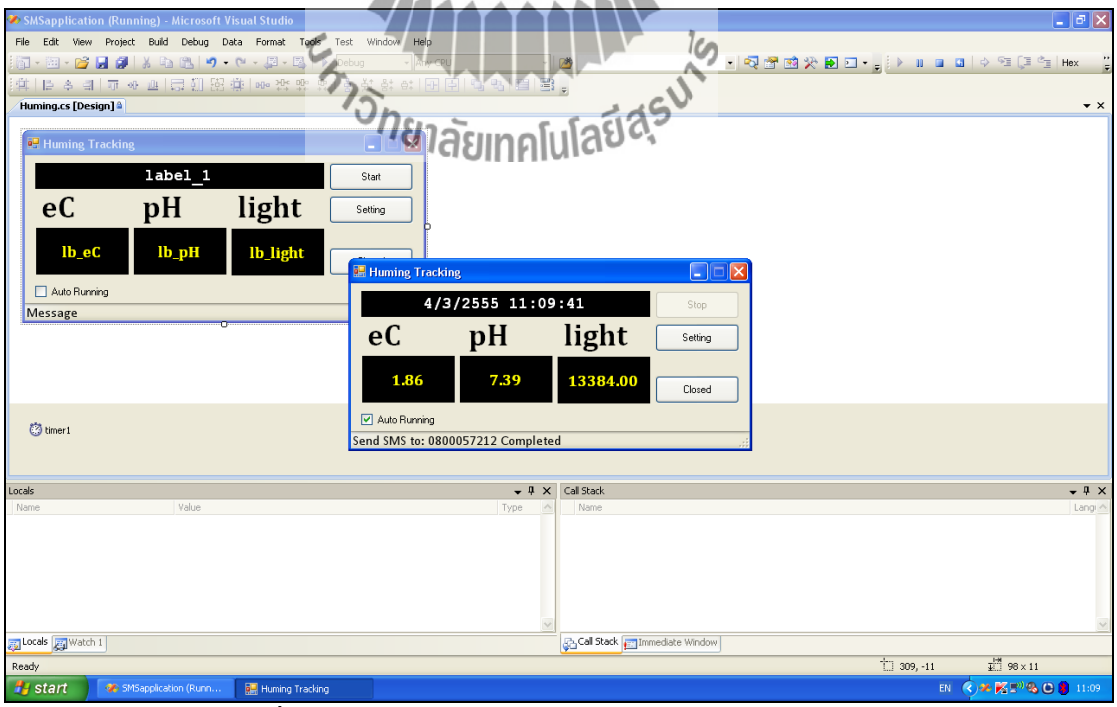
รูปที่ 4.9 แสดงการประมวลผลของโปรแกรม MATLAB R2009a

3) เปิดไฟล์โปรแกรม VB C#2008 ขึ้นมา เปิดไฟล์โปรแกรม Huming แล้วทำการตั้งค่าพารามิเตอร์ต่างๆ และป้อนหมายเลขโทรศัพท์ปลายทาง



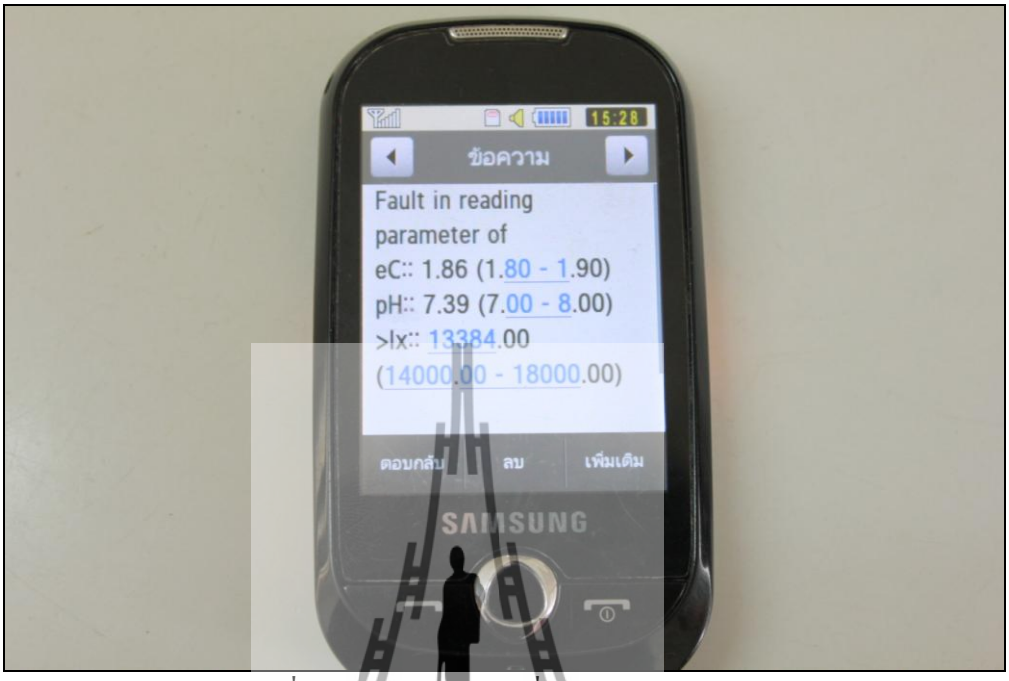
รูปที่ 4.10 แสดงหน้าต่างการตั้งค่าพารามิเตอร์ต่างๆ

4) กดปุ่ม START เพื่อเริ่มทำการประมวลผลค่าพารามิเตอร์



รูปที่ 4.11 แสดงหน้าต่างขณะโปรแกรมทำการประมวลผล

5) เมื่อมีค่าพารามิเตอร์ผิดพลาดโปรแกรมก็จะส่ง SMS ไปยังหมายเลขโทรศัพท์มือถือปลายทาง โดยมีสัญลักษณ์ > อยู่ข้างหน้าพารามิเตอร์นั้น (จากรูป ค่าความเข้มแสงผิดพลาด)



รูปที่ 4.12 แสดงข้อความที่ถูกส่งไปยังผู้ใช้งาน

4.4 สรุปผล

หลังจากที่ได้ทำการทดสอบ ระบบแล้ว ซึ่งสามารถแจ้งเตือน เมื่อค่าความเข้มแสง (Light) ค่าความเป็นกรด-ด่าง (pH) หรือค่าความนำไฟฟ้าในสารละลาย (eC) ผิดปกติไปจากค่าที่เหมาะสม สำหรับการเจริญเติบโตของผักในแปลงผักไฮโดรโพรนิกส์ ซึ่งค่าที่เหมาะสมของการเจริญเติบโตของผักในแปลงผักไฮโดรโพนิกส์ คือค่าความเข้มแสง (Light) จะอยู่ที่ 14000 – 18000 Lux, ค่าความเป็นกรด-ด่าง (pH) จะอยู่ที่ 5.60 – 6.50 และค่าความนำไฟฟ้าในสารละลาย (eC) จะอยู่ที่ 1.80 – 2.20 mS/cm โดยจะทำการส่ง SMS แจ้งเตือนไปยังผู้ใช้ ทำให้ประหยัดเวลาและแรงงานในการดูแลได้มาก

บทที่ 5
บทสรุปของโครงงาน

5.1 บทนำ

เนื้อหาในบทนี้จะกล่าวถึงบทสรุปของโครงงานระบบเฝ้าระวังแปลงผักไฮโดรโพนิกส์ด้วยเซ็นเซอร์ไร้สาย ผ่านระบบ GSM ซึ่งประกอบไปด้วยข้อสรุปของโครงงาน ปัญหาที่พบขณะดำเนินงาน วิธีแก้ปัญหา ข้อเสนอแนะและวิธีการพัฒนาโครงงานต่อไป

5.2 สรุปโครงงาน

โครงงานระบบเฝ้าระวังแปลงผักไฮโดรโพนิกส์ด้วยเซ็นเซอร์ไร้สายผ่านระบบ GSM ได้บรรลุตามวัตถุประสงค์ คือ ระบบเซ็นเซอร์ไร้สายสามารถรับ -ส่งข้อมูลได้อย่างสมบูรณ์ ทำให้มีข้อมูลมาเก็บไว้ที่สถานีฐาน โปรแกรมส่ง SMS ก็สามารถดึงข้อมูลจากสถานีฐานมาตรวจสอบค่าพารามิเตอร์ต่างๆได้ เมื่อพบว่ามีค่าพารามิเตอร์ไม่เหมาะสมกับการเจริญเติบโตของผักก็จะทำการส่ง SMS ไปยังโทรศัพท์เพื่อแจ้งเตือนต่อผู้ใช้งานได้ ซึ่งทำให้เกิดเป็นระบบเฝ้าระวังแปลงผักไฮโดรโพนิกส์ด้วยเซ็นเซอร์ไร้สายผ่านระบบ GSM ที่สามารถนำไปติดตั้งใช้งาน ณ สถานที่จริงได้

5.3 ปัญหาและแนวทางในการแก้ไข

ในการทำโครงงานระบบเฝ้าระวัง แปลงผักไฮโดรโพนิกส์ ด้วยเซ็นเซอร์ไร้สาย ผ่านระบบ GSM ปัญหาที่พบบ่อยๆ แสดงดังตารางที่ 5.1 ซึ่งประกอบด้วยตัวปัญหาที่พบ สาเหตุของปัญหาและรวมถึงวิธีการแก้ไขปัญหา

ตารางที่ 5.1 ปัญหาและสาเหตุที่พบในขณะดำเนินงานและวิธีการแก้ไข

ปัญหาที่พบขณะดำเนินงาน	สาเหตุและวิธีการแก้ไข
1. การเลือกใช้ภาษาในการเขียนโปรแกรมให้เหมาะสมกับโครงงาน	**สาเหตุ:** เนื่องจากผู้จัดทำมีความรู้ในการเขียนโปรแกรมไม่มากนัก **วิธีการแก้ไข**: ทดลองศึกษาเขียนโปรแกรมด้วยภาษาต่างๆ โดยค้นคว้าข้อมูลจากหนังสือและ Internet จนพบภาษาของโปรแกรมที่เหมาะกับโครงงาน คือ โปรแกรม Microsoft Visual C# 2008
2. เกิดข้อผิดพลาดในการเขียนโปรแกรมส่ง SMS	**สาเหตุ:** เนื่องจากทางผู้จัดทำยังไม่มีความชำนาญในการเขียนโปรแกรมด้วยภาษา C# **วิธีการแก้ไข:** ได้ทำการศึกษาหาความรู้เพิ่มเติมของการเขียนโปรแกรมด้วยภาษา C# จากหนังสือและInternet
3. การใช้งานบอร์ด GSM	**สาเหตุ:** เกิดจากทางบริษัท ไม่ได้ส่งคู่มือการใช้งานมาให้ จึงทำให้ผู้จัดทำไม่สามารถใช้งาน **วิธีแก้ไข:** ติดต่อกลับไปที่บริษัทให้ส่งคู่มือแนะนำการใช้งานมาให้และค้นคว้าจาก Internet
4. ปัญหาเกิดเมื่อนำไปติดตั้ง ณ สถานที่จริง	**สาเหตุ:** คอมพิวเตอร์ที่ Base Station ไม่รองรับ Serial Port ของสาย USB-RS232 **วิธีแก้ไข:** ค้นหา Software ของอุปกรณ์ จาก Internet แล้วทำการติดตั้ง Software เพื่อให้คอมพิวเตอร์รองรับอุปกรณ์ สามารถ Download ได้จาก (www.denontech.com/Download.html)

5.4 ข้อเสนอแนะ

5.3.1 ควรศึกษาวิธีการใช้โปรแกรมอย่างละเอียด ก่อนใช้งานเนื่องจากอาจเกิดข้อผิดพลาดระหว่างใช้งานได้

5.3.2 การใช้โปรแกรมควรตรวจสอบการตั้งค่าระบบของโปรแกรมให้เหมาะสม เช่น การตั้งค่าของ Serial Port (วิธีการติดตั้ง Serial Port แนบอยู่ที่ภาคผนวก ก หน้าที่ 118)

5.3.4 การดูแลรักษาเซ็นเซอร์ต้องเป็นไปตามข้อแนะนำที่มากับเซ็นเซอร์เนื่องจากเซ็นเซอร์มีความเปราะบาง ถ้าใช้ไม่ถูกวิ ธีอาจทำให้เกิดความเสียหายได้ ซึ่งจะทำให้ค่าที่วัดได้เกิดความผิดเพี้ยน

5.5 แนวทางในการพัฒนาต่อไป

เนื่องจากโครงงานระบบเฝ้าระวัง แปลงผักไฮโดรโพนิกส์ ด้วยเซ็นเซอร์ไร้สาย ผ่านระบบ GSM สามารถทำการพัฒนาในด้าน Software ให้มีความหลากหลายในการใช้งานได้ เช่น เมื่อผู้ใช้งานอยากทราบสถานะค่าพารามิเตอร์ต่างๆ ณ ปัจจุบัน ก็สามารถส่ง SMS เข้ามาสอบถามได้ และยังสามารถพัฒนาในด้าน Hardware ได้ เช่น ทำการเพิ่มอุปกรณ์เซ็นเซอร์ไร้สาย ในการตรวจวัดค่าพารามิเตอร์ๆได้อีกมากมาย ทั้งนี้เพื่อทำให้ประหยัดทั้งเวลาและบุคลากรที่ต้องคอยตรวจสอบอยู่เสมอ

5.6 กล่าวสรุป

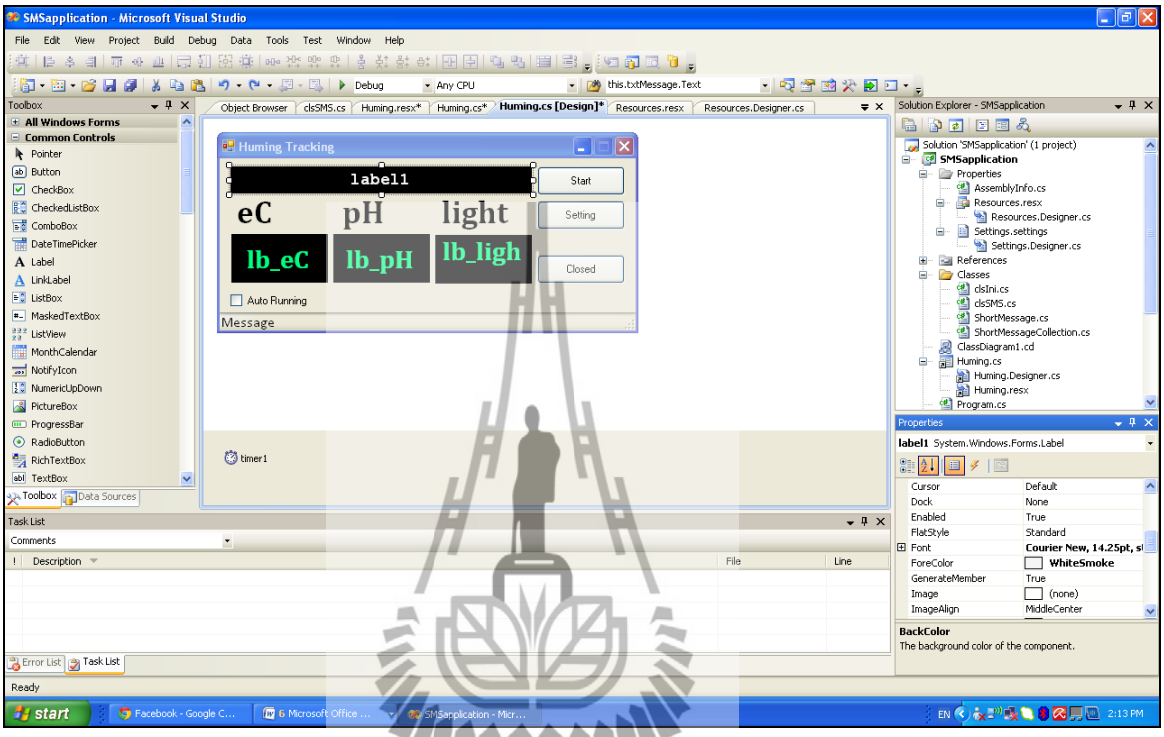
โครงงานระบบเฝ้าระวัง แปลงผักไฮโดรโพนิกส์ ด้วยเซ็นเซอร์ไร้สาย ผ่านระบบ GSM มีส่วนประกอบหลักดังนี้

1. ภาษาที่ใช้ในการพัฒนาเขียนโปรแกรม
   - ภาษา C#
2. Hardware ที่นำมาใช้
   - Computer PC
   - บอร์ด GSM Module Wireless CPU
   - Sim Card โทรศัพท์เคลื่อนที่
   - สายอากาศ
3. เซ็นเซอร์
   - pH (ค่าความเป็นกรด-ด่าง)
   - eC (ค่าความนำไฟฟ้าในสารละลาย)
   - Light (ค่าความเข้มแสง)
4. Software ที่นำมาใช้
   - Microsoft Visual C# 2008

ภาคผนวก

ภาคผนวก ก
Code โปรแกรมที่เกี่ยวข้อง

1.การออกแบบบล็อก




```
namespace SMSapplication
{
    partial class Huming
    {
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
```

```
            base.Dispose(disposing);
        }
        #region Windows Form Designer generated code
        private void InitializeComponent()
        {
            this.components = new System.ComponentModel.Container();
            this.label1 = new System.Windows.Forms.Label();
            this.label2 = new System.Windows.Forms.Label();
            this.label3 = new System.Windows.Forms.Label();
            this.label4 = new System.Windows.Forms.Label();
            this.lb_eC = new System.Windows.Forms.Label();
            this.lb_pH = new System.Windows.Forms.Label();
            this.lb_light = new System.Windows.Forms.Label();
            this.checkBox1 = new System.Windows.Forms.CheckBox();
            this.btnStart = new System.Windows.Forms.Button();
            this.btnClosed = new System.Windows.Forms.Button();
            this.btnSetting = new System.Windows.Forms.Button();
            this.statusBar1 = new System.Windows.Forms.StatusBar();
            this.timer1 = new System.Windows.Forms.Timer(this.components);
            this.SuspendLayout();
            this.label1.BackColor = System.Drawing.Color.Black;
            this.label1.Font = new System.Drawing.Font("Courier New", 14.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label1.ForeColor = System.Drawing.Color.WhiteSmoke;
            this.label1.Location = new System.Drawing.Point(12, 9);
            this.label1.Name = "label1";
            this.label1.Size = new System.Drawing.Size(334, 30);
            this.label1.TabIndex = 0;
            this.label1.Text = "label1";
```

```
            this.label1.TextAlign = System.Drawing.ContentAlignment.MiddleCenter;
            this.label2.Font = new System.Drawing.Font("Cambria", 26.25F,
System.Drawing.FontStyle.Bold);
            this.label2.Location = new System.Drawing.Point(12, 39);
            this.label2.Name = "label2";
            this.label2.Size = new System.Drawing.Size(107, 45);
            this.label2.TabIndex = 1;
            this.label2.Text = "eC";
            this.label3.Font = new System.Drawing.Font("Cambria", 26.25F,
System.Drawing.FontStyle.Bold);
            this.label3.Location = new System.Drawing.Point(129, 39);
            this.label3.Name = "label3";
            this.label3.Size = new System.Drawing.Size(104, 45);
            this.label3.TabIndex = 2;
            this.label3.Text = "pH";
            this.label4.Font = new System.Drawing.Font("Cambria", 26.25F,
System.Drawing.FontStyle.Bold);
            this.label4.Location = new System.Drawing.Point(239, 39);
            this.label4.Name = "label4";
            this.label4.Size = new System.Drawing.Size(107, 46);
            this.label4.TabIndex = 3;
            this.label4.Text = "light";
            this.lb_eC.BackColor = System.Drawing.Color.Black;
            this.lb_eC.Font = new System.Drawing.Font("Cambria", 21.75F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.lb_eC.ForeColor = System.Drawing.Color.SpringGreen;
            this.lb_eC.Location = new System.Drawing.Point(13, 84);
            this.lb_eC.Name = "lb_eC";
            this.lb_eC.Size = new System.Drawing.Size(107, 54);
```

```
            this.lb_eC.TabIndex = 4;
            this.lb_eC.Text = "lb_eC";
            this.lb_eC.TextAlign = System.Drawing.ContentAlignment.MiddleCenter;
            this.lb_pH.BackColor = System.Drawing.Color.Black;
            this.lb_pH.Font = new System.Drawing.Font("Cambria", 21.75F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.lb_pH.ForeColor = System.Drawing.Color.SpringGreen;
            this.lb_pH.Location = new System.Drawing.Point(126, 84);
            this.lb_pH.Name = "lb_pH";
            this.lb_pH.Size = new System.Drawing.Size(107, 54);
            this.lb_pH.TabIndex = 5;
            this.lb_pH.Text = "lb_pH";
            this.lb_pH.TextAlign = System.Drawing.ContentAlignment.MiddleCenter;
            this.lb_light.BackColor = System.Drawing.Color.Black;
            this.lb_light.Font = new System.Drawing.Font("Cambria", 21.75F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.lb_light.ForeColor = System.Drawing.Color.SpringGreen;
            this.lb_light.Location = new System.Drawing.Point(239, 85);
            this.lb_light.Name = "lb_light";
            this.lb_light.Size = new System.Drawing.Size(107, 54);
            this.lb_light.TabIndex = 6;
            this.lb_light.Text = "lb_light";
            this.lb_light.TextAlign = System.Drawing.ContentAlignment.MiddleCenter;
            this.checkBox1.AutoSize = true;
            this.checkBox1.Location = new System.Drawing.Point(12, 150);
            this.checkBox1.Name = "checkBox1";
            this.checkBox1.Size = new System.Drawing.Size(91, 17);
            this.checkBox1.TabIndex = 7;
            this.checkBox1.Text = "Auto Running";
```

```
            this.checkBox1.UseVisualStyleBackColor = true;
            this.checkBox1.CheckedChanged += new
System.EventHandler(this.checkBox1_CheckedChanged);
            this.btnStart.ForeColor = System.Drawing.SystemColors.ControlText;
            this.btnStart.Location = new System.Drawing.Point(352, 9);
            this.btnStart.Name = "btnStart";
            this.btnStart.Size = new System.Drawing.Size(97, 32);
            this.btnStart.TabIndex = 8;
            this.btnStart.Text = "Start";
            this.btnStart.UseVisualStyleBackColor = true;
            this.btnStart.Click += new System.EventHandler(this.btnStart_Click);
            this.btnClosed.Location = new System.Drawing.Point(352, 107);
            this.btnClosed.Name = "btnClosed";
            this.btnClosed.Size = new System.Drawing.Size(97, 32);
            this.btnClosed.TabIndex = 9;
            this.btnClosed.Text = "Closed";
            this.btnClosed.UseVisualStyleBackColor = true;
            this.btnClosed.Click += new System.EventHandler(this.btnClosed_Click);
            this.btnSetting.Location = new System.Drawing.Point(352, 47);
            this.btnSetting.Name = "btnSetting";
            this.btnSetting.Size = new System.Drawing.Size(97, 32);
            this.btnSetting.TabIndex = 10;
            this.btnSetting.Text = "Setting";
            this.btnSetting.UseVisualStyleBackColor = true;
            this.btnSetting.Click += new System.EventHandler(this.btnSetting_Click);
            this.statusBar1.Font = new System.Drawing.Font("Calibri", 12F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.statusBar1.Location = new System.Drawing.Point(0, 171);
            this.statusBar1.Name = "statusBar1";
```

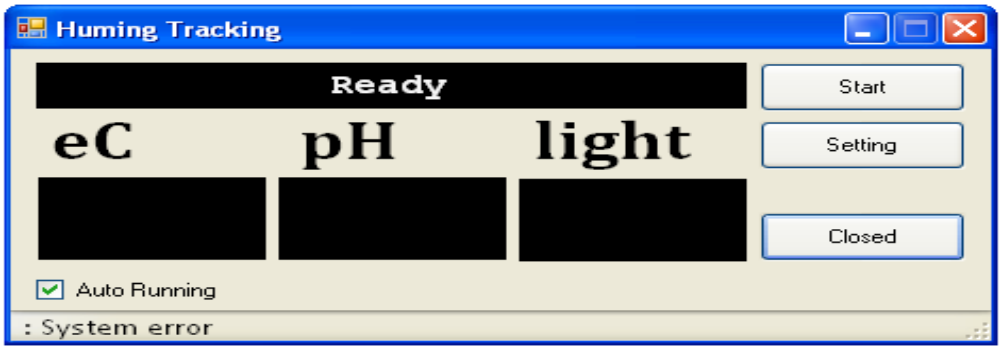
```
this.statusBar1.Size = new System.Drawing.Size(461, 20);
this.statusBar1.TabIndex = 76;
this.statusBar1.Text = "Message";
this.timer1.Interval = 1000;
this.timer1.Tick += new System.EventHandler(this.timer1_Tick);
this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
this.ClientSize = new System.Drawing.Size(461, 191);
this.Controls.Add(this.statusBar1);
this.Controls.Add(this.btnSetting);
this.Controls.Add(this.btnClosed);
this.Controls.Add(this.btnStart);
this.Controls.Add(this.checkBox1);
this.Controls.Add(this.lb_light);
this.Controls.Add(this.lb_pH);
this.Controls.Add(this.lb_eC);
this.Controls.Add(this.label4);
this.Controls.Add(this.label3);
this.Controls.Add(this.label2);
this.Controls.Add(this.label1);
this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedSingle;
this.MaximizeBox = false;
this.Name = "Huming";
this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
this.Text = "Huming Tracking";
this.Load += new System.EventHandler(this.Huming_Load);
this.Activated += new System.EventHandler(this.Huming_Activated);
this.ResumeLayout(false);
this.PerformLayout();
```

```
        }

            #endregion

        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.Label label2;
        private System.Windows.Forms.Label label3;
        private System.Windows.Forms.Label label4;
        private System.Windows.Forms.Label lb_eC;
        private System.Windows.Forms.Label lb_pH;
        private System.Windows.Forms.Label lb_light;
        private System.Windows.Forms.CheckBox checkBox1;
        private System.Windows.Forms.Button btnStart;
        private System.Windows.Forms.Button btnClosed;
        private System.Windows.Forms.Button btnSetting;
        private System.Windows.Forms.StatusBar statusBar1;
        private System.Windows.Forms.Timer timer1;
        private System.ComponentModel.IContainer components;
    }
}
```

2.โปรแกรมของการดึงข้อมูล, เปรียบเทียบข้อมูลและส่ง SMS




```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.IO;
using System.IO.Ports;
using System.Diagnostics;
namespace SMSapplication
{
   public partial class Huming : Form
   {
      #region Private Variables
      SerialPort port = new SerialPort();
      clsSMS objclsSMS = new clsSMS();
 #endregion
      clsIni myConfig = new clsIni("");
      bool sms_first_send;
```

```
bool b_Active = false;
Int32 refresh_check, sms_reply, sec_count, min_count;
public Huming()
{
   InitializeComponent();
   this.Closing += new System.ComponentModel.CancelEventHandler(this.Huming_Closing);
}
private void btnClosed_Click(object sender, EventArgs e)
{
   this.Close ();
}
private void Huming_Load(object sender, EventArgs e)
{
   timer1.Enabled = false;
   label1.Text = "Ready";
   lb_eC.Text = "";
   lb_pH.Text = "";
   lb_light.Text = "";
   statusBar1.Text = "";
   myConfig.refresh ();
   checkBox1.Checked = myConfig.ReadBool("StartUp", "AutoRun");
   if (checkBox1.Checked == true)
   {
      btnStart_Click(sender, e);
   }
}

private void Huming_Closing(object sender, System.ComponentModel.CancelEventArgs e)
{
```

```
        if (MessageBox.Show("Do you want stop process & exit program", "Closed program",
MessageBoxButtons.OKCancel, MessageBoxIcon.Exclamation) == DialogResult.OK)
        {
            if (this.port.IsOpen == true)
            {
                ClosedPort();
            }
        }
        else
        {
            e.Cancel = true;
        }
    }
    private bool ConnectPort()
    {
        try
        {
            String PortName = myConfig.ReadString("COMPORT","PortName");
            String BaudRate = myConfig.ReadString("COMPORT","BaudRate");
            String DataBit = myConfig.ReadString("COMPORT","DataBits");
            String StopBit = myConfig.ReadString("COMPORT","StopBits");
            String rdTimeOut = myConfig.ReadString("COMPORT", "ComReadTimeOut");
            String wrTimeOut = myConfig.ReadString("COMPORT", "ComWriteTimeOut");
            this.statusBar1.Text = "Open comport >" + PortName;
            this.port = objclsSMS.OpenPort(
                        PortName,
                        Convert.ToInt32(BaudRate),
                        Convert.ToInt32(DataBit),
                        Convert.ToInt32(rdTimeOut),
```

```
                Convert.ToInt32(wrTimeOut));
        if (this.port != null)
        {
            this.statusBar1.Text = " : Completed";
            return true;
        }
        else
        {
            this.statusBar1.Text = " : Fail";
            return false;
        }
    }
    catch (Exception)
    {
        this.statusBar1.Text = " : System error";
        return false;
    }
}
private void ClosedPort()
{
    timer1.Stop();
    objclsSMS.ClosePort();
}
private void Read_Huming()
{
    double eC = 0.00, pH = 0.00, lig = 0.00;
    bool trig = false;
    string sms;
    label1.Text = DateTime.Now.ToString();
```

```
if(myConfig.DataFile().Length > 0 )  {
   try
   {
      TextReader trs = new StreamReader(myConfig.DataFile());
      string[] des;
      string[] src = trs.ReadToEnd().Split(new Char[] { '\r' });
      trs.Close();
      for (int ind = src.Length - 1; ind > 0; ind--)
      {
         if ((src[ind]).Trim() != "")
         {
            des = src[ind].Split(new char[] {' ', '\t'});
            eC = Convert.ToDouble(des[5].Trim());
            pH = Convert.ToDouble(des[6].Trim());
            lig = Convert.ToDouble(des[7].Trim());
            lb_eC.Text = eC.ToString("#0.00");
            lb_pH.Text = pH.ToString("#0.00");
            lb_light.Text = lig.ToString("#0.00");
            sms = "Fault in reading parameter of\r\n";
            if (eC < myConfig.ec_min() || eC > myConfig.ec_max()) { trig = true; sms = sms +
">"; }
            sms = sms + "eC:: " + eC.ToString("#0.00") + " (" +
myConfig.ec_min().ToString("#0.00") + " - " + myConfig.ec_max().ToString("#0.00") + ")\r\n";
            if (pH < myConfig.ph_min() || pH > myConfig.ph_max()) { trig = true; sms = sms +
">"; }
            sms = sms + "pH:: " + pH.ToString("#0.00") + " (" +
myConfig.ph_min().ToString("#0.00") + " - " + myConfig.ph_max().ToString("#0.00") + ")\r\n";
            if (lig < myConfig.lx_min() || lig > myConfig.lx_max()) { trig = true; sms = sms +
">"; }
```

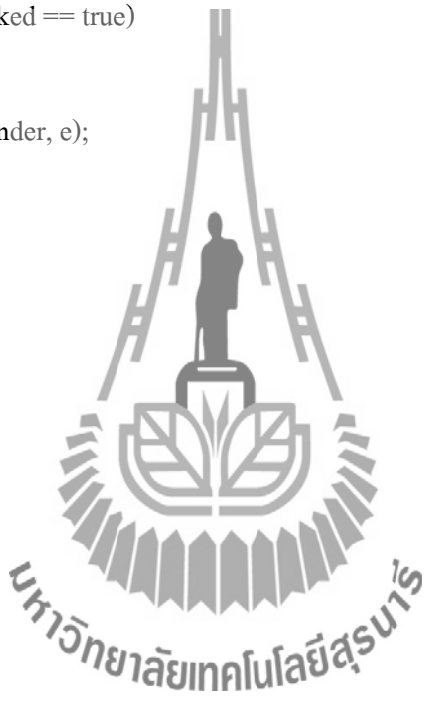
```
                sms = sms + "lx:: " + lig.ToString("#0.00") + " (" +
myConfig.lx_min().ToString("#0.00") + " - " + myConfig.lx_max().ToString("#0.00") + ")\r\n";
                if (trig == true)
                {
                  if (sms_first_send == false)
                  {
                    timer1.Stop();
                    string[] pho;
                    pho = myConfig.Phone().Split(new char[] { ';' });
                    Int32 smsLen = sms.Length;
                    foreach (string s in pho)
                    {
                      if (objclsSMS.sendMsg(s, sms) == true)
                      {
                        statusBar1.Text = "Send SMS to: " + s + " Completed";
                      }
                    }
                    sec_count = 0;
                    min_count = 0;
                    sms_first_send = true;
                    timer1.Start();
                  }
                }
                break;
              }
            }
          }
```

```
    catch (Exception)
      {
      }
   }
}
private void btnStart_Click(object sender, EventArgs e)
{
   btnStart.Enabled = false;
   myConfig.refresh();
   if (btnStart.Text  == "Start")
   {
      if (port.IsOpen == false)
      {
         if (ConnectPort() == true)
         {
            refresh_check = 0;
            sms_first_send = false;
            sms_reply = 0;
            sec_count = 0;
            min_count = 0;
            btnStart.Text = "Stop";
            timer1.Enabled = true;
         }
      }
   }
   else
   {
      if (port.IsOpen == true)
      {
```

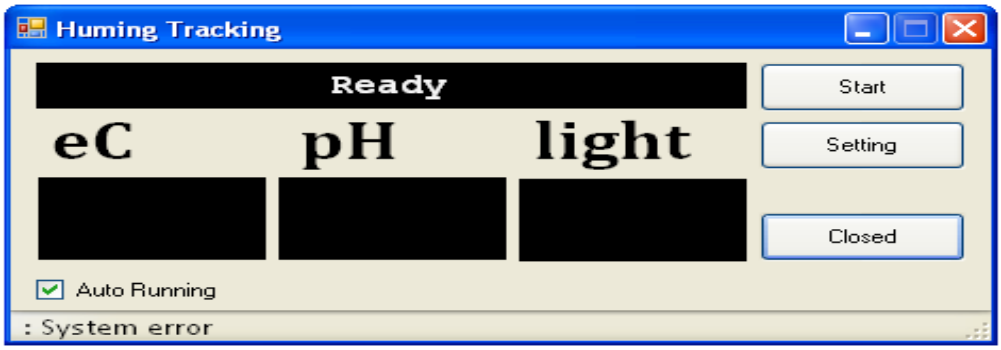
```
            ClosedPort();
        }
        btnStart.Text = "Start";
        timer1.Enabled = false;
        label1.Text = "Ready";
    }
    if (port.IsOpen == false)
    {
        ClosedPort();
        btnStart.Enabled = true ;
    }
    if (checkBox1.Checked == false) { btnStart.Enabled = true; }
}
private void timer1_Tick(object sender, EventArgs e)
{
    timer1.Stop();
    label1.Text = DateTime.Now.ToString();
    if (sec_count % myConfig.TimeRefresh() == 0)
    {
        Read_Huming();
    }
    if (++sec_count == 60)   {
        sec_count = 0;
        ++min_count;
    }
    if (min_count == myConfig.TimeReply())
    {
        min_count = 0;
```

```
            sms_first_send = false ;
        }
        timer1.Start();
    }
private void checkBox1_CheckedChanged(object sender, EventArgs e)
    {
        if (checkBox1.Checked == true)
        {
            if (this.port.IsOpen == false)
            {
                btnStart_Click(sender , e);
            }
            if (this.port.IsOpen == true)
            {
                btnStart.Enabled = false;
            }
        }
        else
        {
            if (this.port.IsOpen == true)
            {
                ClosedPort();
            }
            btnStart.Enabled = true;
        }
        myConfig.IniWriteValue("StartUp", "AutoRun",Convert.ToString(checkBox1.Checked));
    }
    private void btnSetting_Click(object sender, EventArgs e)
    {
```

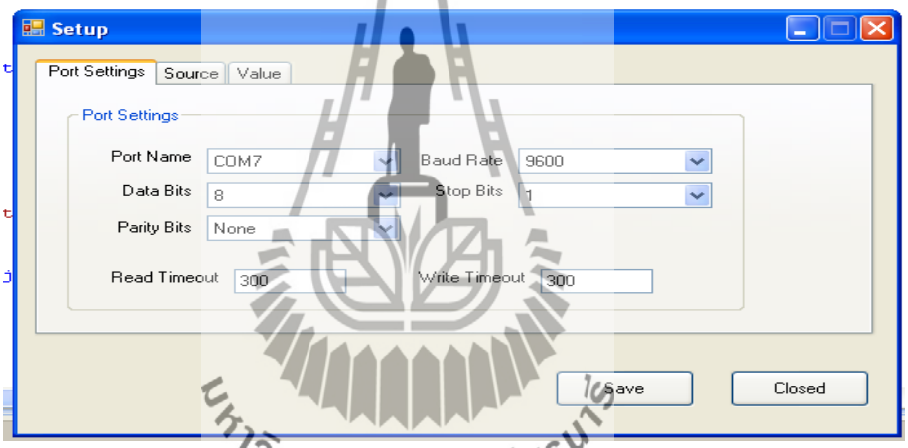
```
            Setup f = new Setup();
            f.ShowDialog();
            myConfig.refresh();
        }
        private void Huming_Activated(object sender, EventArgs e)
        {
            if (b_Active == true) return;
            checkBox1.Checked = myConfig.ReadBool("StartUp", "AutoRun");
            if (checkBox1.Checked == true)
            {
                btnStart_Click(sender, e);
            }
        }
    }
}
```

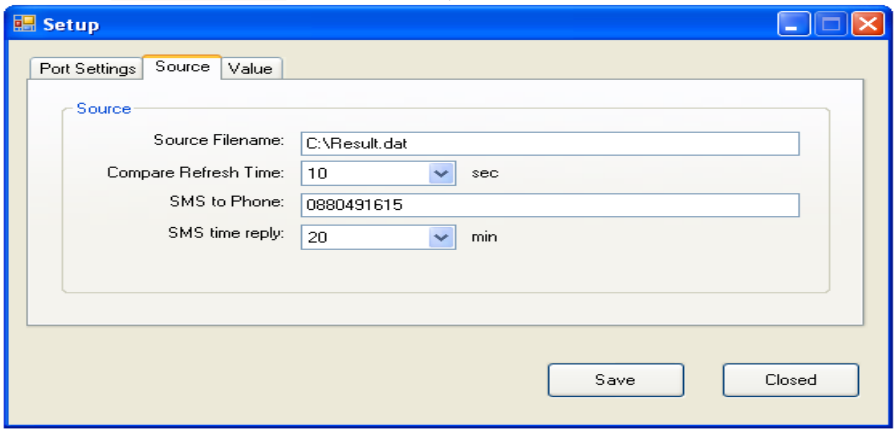


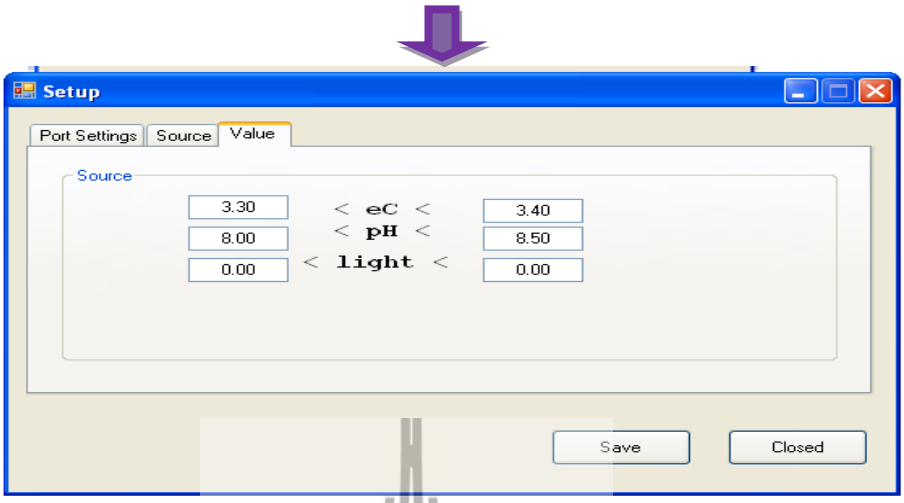
3.โปรแกรม Set ค่าต่างๆเพื่อเชื่อมต่อกับบอร์ด GSM



Click








```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
namespace SMSapplication
{
   public partial class Setup : Form
   {
      public clsIni myConfig = new clsIni("");
      public Setup()
      {
         InitializeComponent();
      }
private void Setup_Load(object sender, EventArgs e)
      {
         Int16 i;
```

```
        myConfig.refresh();
        cboPortName.Items.Clear();
        for (i = 1; i < 30; i++)
        {
           cboPortName.Items.Add("COM" + Convert.ToString(i));
        }
        cboPortName.Text= myConfig.ReadString("COMPORT", "PortName","COM1");
        cboBaudRate.Text = myConfig.ReadString("COMPORT", "BaudRate","9600");
        cboDataBits.Text = myConfig.ReadString ("COMPORT", "DataBits","8");
        cboStopBits.Text = myConfig.ReadString ("COMPORT", "StopBits","1");
        cboParityBits.Text = myConfig.ReadString ("COMPORT", "ParityBits","None");
        txtReadTimeOut.Text = myConfig.ReadString ("COMPORT", "ComReadTimeOut","300");
        txtWriteTimeOut.Text = myConfig.ReadString ("COMPORT", "ComWriteTimeOut","300");
        txtPhone.Text = myConfig.Phone();
        txtSrcFilename.Text = myConfig.DataFile();
        cboCompare.Text = myConfig.TimeRefresh ().ToString ("#0");  // 10 sec compare
        cboReply.Text = myConfig.TimeReply ().ToString("#0");  // 10 min reply sms
        textBox1.Text = myConfig.ec_min().ToString("#0.00");
        textBox2.Text = myConfig.ec_max().ToString("#0.00");
        textBox3.Text = myConfig.ph_min().ToString("#0.00");
        textBox4.Text = myConfig.ph_max().ToString("#0.00");
        textBox5.Text = myConfig.lx_min().ToString("#0.00");
        textBox6.Text = myConfig.lx_max().ToString("#0.00");
    }
private void btnClosed_Click(object sender, EventArgs e)
    {
        this.Close();
    }
```

```
private void btnSave_Click(object sender, EventArgs e)
    {
        myConfig.IniWriteValue("COMPORT", "PortName", cboPortName.Text);
        myConfig.IniWriteValue ("COMPORT", "BaudRate", cboBaudRate.Text);
        myConfig.IniWriteValue ("COMPORT", "DataBits", cboDataBits.Text);
        myConfig.IniWriteValue ("COMPORT", "StopBits", cboStopBits.Text);
        myConfig.IniWriteValue ("COMPORT", "ParityBits", cboParityBits.Text);
        myConfig.IniWriteValue ("COMPORT", "ComReadTimeOut", txtReadTimeOut.Text);
        myConfig.IniWriteValue ("COMPORT", "ComWriteTimeOut", txtWriteTimeOut.Text);
        myConfig.Phone(txtPhone.Text);
        myConfig.DataFile(txtSrcFilename.Text);
        myConfig.TimeRefresh(Convert.ToInt32(cboCompare.Text));
        myConfig.TimeReply(Convert.ToInt32(cboReply.Text));
        if (textBox1.Text.Length > 0) myConfig.ec_min(Convert.ToDouble(textBox1.Text.Trim()));
        if(textBox2.Text.Length>0) myConfig.ec_max (Convert.ToDouble(textBox2.Text.Trim()));
        if (textBox3.Text.Length > 0) myConfig.ph_min (Convert.ToDouble(textBox3.Text.Trim()));
        if (textBox4.Text.Length > 0) myConfig.ph_max (Convert.ToDouble(textBox4.Text.Trim()));
        if (textBox5.Text.Length > 0) myConfig.lx_min (Convert.ToDouble(textBox5.Text.Trim()));
        if (textBox6.Text.Length > 0) myConfig.lx_max (Convert.ToDouble(textBox6.Text.Trim()));
        this.Close();
    }
  }
}
```

4. Code ในการออกแบบบล็อก Setup ด้วย VB C# (ต่อจากข้อ 3)

```
namespace SMSapplication
{
   partial class Setup
   {
      private System.ComponentModel.IContainer components = null;
      protected override void Dispose(bool disposing)
      {
         if (disposing && (components != null))
         {
            components.Dispose();
         }
         base.Dispose(disposing);
      }
      #region Windows Form Designer generated code
      private void InitializeComponent()
      {
         this.tabSMSapplication = new System.Windows.Forms.TabControl();
         this.tbPortSettings = new System.Windows.Forms.TabPage();
         this.gboPortSettings = new System.Windows.Forms.GroupBox();
         this.txtWriteTimeOut = new System.Windows.Forms.TextBox();
         this.txtReadTimeOut = new System.Windows.Forms.TextBox();
         this.cboParityBits = new System.Windows.Forms.ComboBox();
         this.cboStopBits = new System.Windows.Forms.ComboBox();
         this.cboDataBits = new System.Windows.Forms.ComboBox();
         this.cboBaudRate = new System.Windows.Forms.ComboBox();
         this.cboPortName = new System.Windows.Forms.ComboBox();
         this.label7 = new System.Windows.Forms.Label();
```

```
this.label6 = new System.Windows.Forms.Label();
this.label5 = new System.Windows.Forms.Label();
this.label4 = new System.Windows.Forms.Label();
this.label3 = new System.Windows.Forms.Label();
this.label2 = new System.Windows.Forms.Label();
this.label1 = new System.Windows.Forms.Label();
this.tbSendSMS = new System.Windows.Forms.TabPage();
this.gboSource = new System.Windows.Forms.GroupBox();
this.cboReply = new System.Windows.Forms.ComboBox();
this.cboCompare = new System.Windows.Forms.ComboBox();
this.txtPhone = new System.Windows.Forms.TextBox();
this.txtSrcFilename = new System.Windows.Forms.TextBox();
this.label10 = new System.Windows.Forms.Label();
this.label11 = new System.Windows.Forms.Label();
this.label9 = new System.Windows.Forms.Label();
this.label8 = new System.Windows.Forms.Label();
this.tabPage1 = new System.Windows.Forms.TabPage();
this.groupBox1 = new System.Windows.Forms.GroupBox();
this.textBox6 = new System.Windows.Forms.TextBox();
this.textBox5 = new System.Windows.Forms.TextBox();
this.textBox4 = new System.Windows.Forms.TextBox();
this.textBox3 = new System.Windows.Forms.TextBox();
this.textBox2 = new System.Windows.Forms.TextBox();
this.textBox1 = new System.Windows.Forms.TextBox();
this.label13 = new System.Windows.Forms.Label();
this.label14 = new System.Windows.Forms.Label();
this.label15 = new System.Windows.Forms.Label();
this.btnSave = new System.Windows.Forms.Button();
this.btnClosed = new System.Windows.Forms.Button();
```

```
this.label12 = new System.Windows.Forms.Label();
this.label16 = new System.Windows.Forms.Label();
this.tabSMSapplication.SuspendLayout();
this.tbPortSettings.SuspendLayout();
this.gboPortSettings.SuspendLayout();
this.tbSendSMS.SuspendLayout();
this.gboSource.SuspendLayout();
this.tabPage1.SuspendLayout();
this.groupBox1.SuspendLayout();
this.SuspendLayout();
this.tabSMSapplication.Controls.Add(this.tbPortSettings);
this.tabSMSapplication.Controls.Add(this.tbSendSMS);
this.tabSMSapplication.Controls.Add(this.tabPage1);
this.tabSMSapplication.Location = new System.Drawing.Point(11, 12);
this.tabSMSapplication.Name = "tabSMSapplication";
this.tabSMSapplication.SelectedIndex = 0;
this.tabSMSapplication.Size = new System.Drawing.Size(524, 227);
this.tabSMSapplication.TabIndex = 1;
this.tbPortSettings.Controls.Add(this.gboPortSettings);
this.tbPortSettings.Location = new System.Drawing.Point(4, 22);
this.tbPortSettings.Name = "tbPortSettings";
this.tbPortSettings.Padding = new System.Windows.Forms.Padding(3);
this.tbPortSettings.Size = new System.Drawing.Size(516, 201);
this.tbPortSettings.TabIndex = 0;
this.tbPortSettings.Text = "Port Settings";
this.tbPortSettings.UseVisualStyleBackColor = true;
this.gboPortSettings.Controls.Add(this.txtWriteTimeOut);
this.gboPortSettings.Controls.Add(this.txtReadTimeOut);
this.gboPortSettings.Controls.Add(this.cboParityBits);
```

```
this.gboPortSettings.Controls.Add(this.cboStopBits);
this.gboPortSettings.Controls.Add(this.cboDataBits);
this.gboPortSettings.Controls.Add(this.cboBaudRate);
this.gboPortSettings.Controls.Add(this.cboPortName);
this.gboPortSettings.Controls.Add(this.label7);
this.gboPortSettings.Controls.Add(this.label6);
this.gboPortSettings.Controls.Add(this.label5);
this.gboPortSettings.Controls.Add(this.label4);
this.gboPortSettings.Controls.Add(this.label3);
this.gboPortSettings.Controls.Add(this.label2);
this.gboPortSettings.Controls.Add(this.label1);
this.gboPortSettings.Location = new System.Drawing.Point(16, 17);
this.gboPortSettings.Name = "gboPortSettings";
this.gboPortSettings.Size = new System.Drawing.Size(422, 167);
this.gboPortSettings.TabIndex = 0;
this.gboPortSettings.TabStop = false;
this.gboPortSettings.Text = "Port Settings";
this.txtWriteTimeOut.Location = new System.Drawing.Point(295, 133);
this.txtWriteTimeOut.MaxLength = 5;
this.txtWriteTimeOut.Name = "txtWriteTimeOut";
this.txtWriteTimeOut.Size = new System.Drawing.Size(70, 20);
this.txtWriteTimeOut.TabIndex = 13;
this.txtWriteTimeOut.Text = "300";
this.txtReadTimeOut.Location = new System.Drawing.Point(104, 133);
this.txtReadTimeOut.MaxLength = 5;
this.txtReadTimeOut.Name = "txtReadTimeOut";
this.txtReadTimeOut.Size = new System.Drawing.Size(70, 20);
this.txtReadTimeOut.TabIndex = 12;
this.txtReadTimeOut.Text = "300";
```

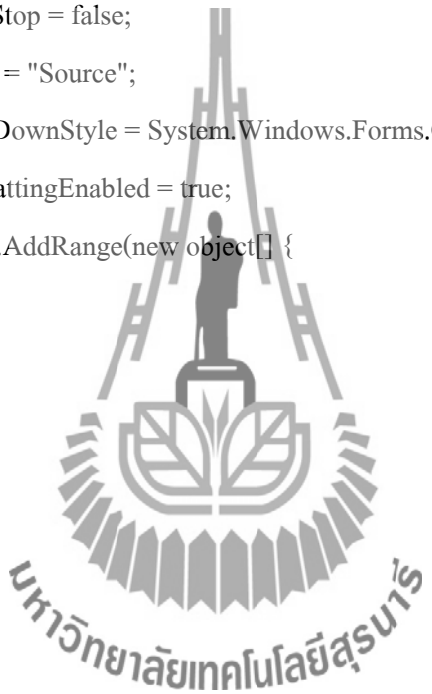
```
            this.cboParityBits.DropDownStyle =
System.Windows.Forms.ComboBoxStyle.DropDownList;
            this.cboParityBits.FormattingEnabled = true;
            this.cboParityBits.Items.AddRange(new object[] {
            "Even",
            "Odd",
            "None"});
            this.cboParityBits.Location = new System.Drawing.Point(87, 89);
            this.cboParityBits.Name = "cboParityBits";
            this.cboParityBits.Size = new System.Drawing.Size(121, 21);
            this.cboParityBits.TabIndex = 11;
            this.cboStopBits.DropDownStyle = System.Windows.Forms.ComboBoxStyle.DropDownList;
            this.cboStopBits.FormattingEnabled = true;
            this.cboStopBits.Items.AddRange(new object[] {
            "1",
            "1.5",
            "2"});
            this.cboStopBits.Location = new System.Drawing.Point(281, 62);
            this.cboStopBits.Name = "cboStopBits";
            this.cboStopBits.Size = new System.Drawing.Size(121, 21);
            this.cboStopBits.TabIndex = 10;
            this.cboDataBits.DropDownStyle = System.Windows.Forms.ComboBoxStyle.DropDownList;
            this.cboDataBits.FormattingEnabled = true;
            this.cboDataBits.Items.AddRange(new object[] {
            "5",
            "6",
            "7",
            "8"});
            this.cboDataBits.Location = new System.Drawing.Point(87, 62);
```

```
            this.cboDataBits.Name = "cboDataBits";
            this.cboDataBits.Size = new System.Drawing.Size(121, 21);
            this.cboDataBits.TabIndex = 9;
            this.cboBaudRate.DropDownStyle =
System.Windows.Forms.ComboBoxStyle.DropDownList;
            this.cboBaudRate.FormattingEnabled = true;
            this.cboBaudRate.Items.AddRange(new object[] {
            "110",
            "300",
            "1200",
            "2400",
            "4800",
            "9600",
            "19200",
            "38400",
            "57600",
            "115200",
            "230400",
            "460800",
            "921600"});
            this.cboBaudRate.Location = new System.Drawing.Point(281, 34);
            this.cboBaudRate.Name = "cboBaudRate";
            this.cboBaudRate.Size = new System.Drawing.Size(121, 21);
            this.cboBaudRate.TabIndex = 8;
            this.cboPortName.DropDownStyle =
System.Windows.Forms.ComboBoxStyle.DropDownList;
            this.cboPortName.FormattingEnabled = true;
            this.cboPortName.Location = new System.Drawing.Point(87, 34);
            this.cboPortName.Name = "cboPortName";
```

```
this.cboPortName.Size = new System.Drawing.Size(121, 21);
this.cboPortName.TabIndex = 7;
this.label7.AutoSize = true;
this.label7.Location = new System.Drawing.Point(216, 133);
this.label7.Name = "label7";
this.label7.Size = new System.Drawing.Size(73, 13);
this.label7.TabIndex = 6;
this.label7.Text = "Write Timeout";
this.label6.AutoSize = true;
this.label6.Location = new System.Drawing.Point(24, 133);
this.label6.Name = "label6";
this.label6.Size = new System.Drawing.Size(74, 13);
this.label6.TabIndex = 5;
this.label6.Text = "Read Timeout";
this.label5.AutoSize = true;
this.label5.Location = new System.Drawing.Point(28, 92);
this.label5.Name = "label5";
this.label5.Size = new System.Drawing.Size(53, 13);
this.label5.TabIndex = 4;
this.label5.Text = "Parity Bits";
this.label4.AutoSize = true;
this.label4.Location = new System.Drawing.Point(226, 62);
this.label4.Name = "label4";
this.label4.Size = new System.Drawing.Size(49, 13);
this.label4.TabIndex = 3;
this.label4.Text = "Stop Bits";
this.label3.AutoSize = true;
this.label3.Location = new System.Drawing.Point(31, 62);
this.label3.Name = "label3";
```

```
            this.label3.Size = new System.Drawing.Size(50, 13);
            this.label3.TabIndex = 2;
            this.label3.Text = "Data Bits";
            this.label2.AutoSize = true;
            this.label2.Location = new System.Drawing.Point(217, 37);
            this.label2.Name = "label2";
            this.label2.Size = new System.Drawing.Size(58, 13);
            this.label2.TabIndex = 1;
            this.label2.Text = "Baud Rate";
            this.label1.AutoSize = true;
            this.label1.Location = new System.Drawing.Point(24, 34);
            this.label1.Name = "label1";
            this.label1.Size = new System.Drawing.Size(57, 13);
            this.label1.TabIndex = 0;
            this.label1.Text = "Port Name";
            this.tbSendSMS.Controls.Add(this.gboSource);
            this.tbSendSMS.Location = new System.Drawing.Point(4, 22);
            this.tbSendSMS.Name = "tbSendSMS";
            this.tbSendSMS.Padding = new System.Windows.Forms.Padding(3);
            this.tbSendSMS.Size = new System.Drawing.Size(516, 201);
            this.tbSendSMS.TabIndex = 1;
            this.tbSendSMS.Text = "Source";
this.tbSendSMS.UseVisualStyleBackColor = true;
            this.gboSource.Controls.Add(this.label16);
            this.gboSource.Controls.Add(this.label12);
            this.gboSource.Controls.Add(this.cboReply);
            this.gboSource.Controls.Add(this.cboCompare);
            this.gboSource.Controls.Add(this.txtPhone);
            this.gboSource.Controls.Add(this.txtSrcFilename);
```

```
this.gboSource.Controls.Add(this.label10);
this.gboSource.Controls.Add(this.label11);
this.gboSource.Controls.Add(this.label9);
this.gboSource.Controls.Add(this.label8);
this.gboSource.Location = new System.Drawing.Point(18, 16);
this.gboSource.Name = "gboSource";
this.gboSource.Size = new System.Drawing.Size(479, 160);
this.gboSource.TabIndex = 43;
this.gboSource.TabStop = false;
this.gboSource.Text = "Source";
this.cboReply.DropDownStyle = System.Windows.Forms.ComboBoxStyle.DropDownList;
this.cboReply.FormattingEnabled = true;
this.cboReply.Items.AddRange(new object[] {
"1",
"2",
"3",
"4",
"5",
"6",
"7",
"8",
"9",
"10",
"11",
"12",
"13",
"14",
"15",
"16",
```



```
            "17",
            "18",
            "19",
            "20"});
            this.cboReply.Location = new System.Drawing.Point(149, 103);
            this.cboReply.Name = "cboReply";
            this.cboReply.Size = new System.Drawing.Size(97, 21);
            this.cboReply.TabIndex = 7;
            this.cboCompare.DropDownStyle =
System.Windows.Forms.ComboBoxStyle.DropDownList;
            this.cboCompare.FormattingEnabled = true;
            this.cboCompare.Items.AddRange(new object[] {
            "1",
            "2",
            "3",
            "4",
            "5",
            "6",
            "7",
            "8",
            "9",
            "10",
            "11",
            "12",
            "13",
            "14",
            "15",
            "16",
            "17",
```

"18",
"19",
"20",
"21",
"22",
"23",
"24",
"25",
"26",
"27",
"28",
"29",
"30",
"31",
"32",
"33",
"34",
"35",
"36",
"37",
"38",
"39",
"40"});
this.cboCompare.Location = new System.Drawing.Point(149, 50);
this.cboCompare.Name = "cboCompare";
this.cboCompare.Size = new System.Drawing.Size(97, 21);
this.cboCompare.TabIndex = 6;
this.txtPhone.Location = new System.Drawing.Point(149, 77);
this.txtPhone.Name = "txtPhone";

```
this.txtPhone.Size = new System.Drawing.Size(311, 20);
this.txtPhone.TabIndex = 5;
this.txtPhone.Text = "txtPhone";
this.txtSrcFilename.Location = new System.Drawing.Point(149, 26);
this.txtSrcFilename.Name = "txtSrcFilename";
this.txtSrcFilename.Size = new System.Drawing.Size(311, 20);
this.txtSrcFilename.TabIndex = 4;
this.txtSrcFilename.Text = "txtSrcFilename";
this.label10.AutoSize = true;
this.label10.Location = new System.Drawing.Point(63, 103);
this.label10.Name = "label10";
this.label10.Size = new System.Drawing.Size(80, 13);
this.label10.TabIndex = 3;
this.label10.Text = "SMS time reply:";
this.label11.AutoSize = true;
this.label11.Location = new System.Drawing.Point(64, 77);
this.label11.Name = "label11";
this.label11.Size = new System.Drawing.Size(79, 13);
this.label11.TabIndex = 2;
this.label11.Text = "SMS to Phone:";
this.label9.AutoSize = true;
this.label9.Location = new System.Drawing.Point(25, 53);
this.label9.Name = "label9";
this.label9.Size = new System.Drawing.Size(118, 13);
this.label9.TabIndex = 1;
this.label9.Text = "Compare Refresh Time:";
this.label8.AutoSize = true;
this.label8.Location = new System.Drawing.Point(54, 26);
this.label8.Name = "label8";
```

```
this.label8.Size = new System.Drawing.Size(89, 13);
this.label8.TabIndex = 0;
this.label8.Text = "Source Filename:";
this.tabPage1.Controls.Add(this.groupBox1);
this.tabPage1.Location = new System.Drawing.Point(4, 22);
this.tabPage1.Name = "tabPage1";
this.tabPage1.Size = new System.Drawing.Size(516, 201);
this.tabPage1.TabIndex = 2;
this.tabPage1.Text = "Value";
this.tabPage1.UseVisualStyleBackColor = true;
this.groupBox1.Controls.Add(this.textBox6);
this.groupBox1.Controls.Add(this.textBox5);
this.groupBox1.Controls.Add(this.textBox4);
this.groupBox1.Controls.Add(this.textBox3);
this.groupBox1.Controls.Add(this.textBox2);
this.groupBox1.Controls.Add(this.textBox1);
this.groupBox1.Controls.Add(this.label13);
this.groupBox1.Controls.Add(this.label14);
this.groupBox1.Controls.Add(this.label15);
this.groupBox1.Location = new System.Drawing.Point(18, 16);
this.groupBox1.Name = "groupBox1";
this.groupBox1.Size = new System.Drawing.Size(479, 160);
this.groupBox1.TabIndex = 44;
this.groupBox1.TabStop = false;
this.groupBox1.Text = "Source";
this.textBox6.Location = new System.Drawing.Point(260, 75);
this.textBox6.Name = "textBox6";
this.textBox6.Size = new System.Drawing.Size(62, 20);
this.textBox6.TabIndex = 10;
```

```
this.textBox6.Text = "textBox6";
this.textBox6.TextAlign = System.Windows.Forms.HorizontalAlignment.Center;
this.textBox5.Location = new System.Drawing.Point(78, 75);
this.textBox5.Name = "textBox5";
this.textBox5.Size = new System.Drawing.Size(62, 20);
this.textBox5.TabIndex = 9;
this.textBox5.Text = "textBox5";
this.textBox5.TextAlign = System.Windows.Forms.HorizontalAlignment.Center;
this.textBox4.Location = new System.Drawing.Point(260, 49);
this.textBox4.Name = "textBox4";
this.textBox4.Size = new System.Drawing.Size(62, 20);
this.textBox4.TabIndex = 8;
this.textBox4.Text = "textBox4";
this.textBox4.TextAlign = System.Windows.Forms.HorizontalAlignment.Center;
this.textBox3.Location = new System.Drawing.Point(78, 49);
this.textBox3.Name = "textBox3";
this.textBox3.Size = new System.Drawing.Size(62, 20);
this.textBox3.TabIndex = 7;
this.textBox3.Text = "textBox3";
this.textBox3.TextAlign = System.Windows.Forms.HorizontalAlignment.Center;
this.textBox2.Location = new System.Drawing.Point(260, 26);
this.textBox2.Name = "textBox2";
this.textBox2.Size = new System.Drawing.Size(62, 20);
this.textBox2.TabIndex = 6;
this.textBox2.Text = "textBox2";
this.textBox2.TextAlign = System.Windows.Forms.HorizontalAlignment.Center;
this.textBox1.Location = new System.Drawing.Point(78, 23);
this.textBox1.Name = "textBox1";
this.textBox1.Size = new System.Drawing.Size(62, 20);
```

```
this.textBox1.TabIndex = 5;
this.textBox1.Text = "textBox1";
this.textBox1.TextAlign = System.Windows.Forms.HorizontalAlignment.Center;
this.label13.AutoSize = true;
this.label13.Font = new System.Drawing.Font("Courier New", 12F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
this.label13.Location = new System.Drawing.Point(146, 70);
this.label13.Name = "label13";
this.label13.Size = new System.Drawing.Size(98, 18);
this.label13.TabIndex = 2;
this.label13.Text = "< light <";
this.label14.AutoSize = true;
this.label14.Font = new System.Drawing.Font("Courier New", 12F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
this.label14.Location = new System.Drawing.Point(165, 44);
this.label14.Name = "label14";
this.label14.Size = new System.Drawing.Size(68, 18);
this.label14.TabIndex = 1;
this.label14.Text = "< pH <";
this.label15.AutoSize = true;
this.label15.Font = new System.Drawing.Font("Courier New", 12F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
this.label15.Location = new System.Drawing.Point(165, 26);
this.label15.Name = "label15";
this.label15.Size = new System.Drawing.Size(68, 18);
this.label15.TabIndex = 0;
this.label15.Text = "< eC <";
this.btnSave.Location = new System.Drawing.Point(335, 267);
this.btnSave.Name = "btnSave";
```

```
this.btnSave.Size = new System.Drawing.Size(87, 30);
this.btnSave.TabIndex = 2;
this.btnSave.Text = "Save";
this.btnSave.UseVisualStyleBackColor = true;
this.btnSave.Click += new System.EventHandler(this.btnSave_Click);
this.btnClosed.Location = new System.Drawing.Point(444, 267);
this.btnClosed.Name = "btnClosed";
this.btnClosed.Size = new System.Drawing.Size(87, 30);
this.btnClosed.TabIndex = 3;
this.btnClosed.Text = "Closed";
this.btnClosed.UseVisualStyleBackColor = true;
this.btnClosed.Click += new System.EventHandler(this.btnClosed_Click);
this.label12.AutoSize = true;
this.label12.Location = new System.Drawing.Point(252, 53);
this.label12.Name = "label12";
this.label12.Size = new System.Drawing.Size(24, 13);
this.label12.TabIndex = 8;
this.label12.Text = "sec";
this.label16.AutoSize = true;
this.label16.Location = new System.Drawing.Point(252, 106);
this.label16.Name = "label16";
this.label16.Size = new System.Drawing.Size(23, 13);
this.label16.TabIndex = 9;
this.label16.Text = "min";
this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
this.ClientSize = new System.Drawing.Size(548, 319);
this.Controls.Add(this.btnClosed);
this.Controls.Add(this.btnSave);
```

```
        this.Controls.Add(this.tabSMSapplication);
        this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedSingle;
        this.MaximizeBox = false;
        this.Name = "Setup";
        this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
        this.Text = "Setup";
        this.Load += new System.EventHandler(this.Setup_Load);
        this.tabSMSapplication.ResumeLayout(false);
        this.tbPortSettings.ResumeLayout(false);
        this.gboPortSettings.ResumeLayout(false);
        this.gboPortSettings.PerformLayout();
        this.tbSendSMS.ResumeLayout(false);
        this.gboSource.ResumeLayout(false);
        this.gboSource.PerformLayout();
        this.tabPage1.ResumeLayout(false);
        this.groupBox1.ResumeLayout(false);
        this.groupBox1.PerformLayout();
        this.ResumeLayout(false);
    }
    #endregion
    private System.Windows.Forms.TabControl tabSMSapplication;
    private System.Windows.Forms.TabPage tbPortSettings;
    private System.Windows.Forms.GroupBox gboPortSettings;
    private System.Windows.Forms.TextBox txtWriteTimeOut;
    private System.Windows.Forms.TextBox txtReadTimeOut;
    private System.Windows.Forms.ComboBox cboParityBits;
    private System.Windows.Forms.ComboBox cboStopBits;
    private System.Windows.Forms.ComboBox cboDataBits;
    private System.Windows.Forms.ComboBox cboBaudRate;
```

```
private System.Windows.Forms.ComboBox cboPortName;
private System.Windows.Forms.Label label7;
private System.Windows.Forms.Label label6;
private System.Windows.Forms.Label label5;
private System.Windows.Forms.Label label4;
private System.Windows.Forms.Label label3;
private System.Windows.Forms.Label label2;
private System.Windows.Forms.Label label1;
private System.Windows.Forms.TabPage tbSendSMS;
private System.Windows.Forms.GroupBox gboSource;
private System.Windows.Forms.Label label11;
private System.Windows.Forms.Label label9;
private System.Windows.Forms.Label label8;
private System.Windows.Forms.Button btnSave;
private System.Windows.Forms.Button btnClosed;
private System.Windows.Forms.ComboBox cboReply;
private System.Windows.Forms.ComboBox cboCompare;
private System.Windows.Forms.TextBox txtPhone;
private System.Windows.Forms.TextBox txtSrcFilename;
private System.Windows.Forms.Label label10;
private System.Windows.Forms.TabPage tabPage1;
private System.Windows.Forms.GroupBox groupBox1;
private System.Windows.Forms.TextBox textBox1;
private System.Windows.Forms.Label label13;
private System.Windows.Forms.Label label14;
private System.Windows.Forms.Label label15;
private System.Windows.Forms.TextBox textBox2;
private System.Windows.Forms.TextBox textBox6;
private System.Windows.Forms.TextBox textBox5;
```

```
        private System.Windows.Forms.TextBox textBox4;
        private System.Windows.Forms.TextBox textBox3;
        private System.Windows.Forms.Label label16;
        private System.Windows.Forms.Label label12;
    }
}
```

5. Code โปรแกรม Matlab

```
%//////////////////////////////////////////////
%        HYDROPONIC PLANT
%
% Data sensing : pH (pH5.6 - 6.5)
%              : EC (1.8 - 2.2 mS/cm)
%              : Light (LUX)
%////////////////////////////////////////////// /
clear all;
clc;
%=================================================================================
%              Part 2 :: Reading Data from Sensor
%=================================================================================
   pck = 12;
   round = 1;  % Reading counter
   Fl = 1;     % Fail reading counter
   Cp = 0;     % Completed reading counter
while 1 %Cp <= 5    % Continuous running
   c = clock;
   min_time = c(1,5);
   sec_time = c(1,6);
   failed = 0;
  while failed ~= 1
   disp ('Reading Now...')
   s = serial('COM7','BaudRate',57600);    % Set port@COM4 and BaudRate
   set(s,'TimeOut',90);                  % Determine Timeout
   fopen(s)                            % Open port
   save1 = fread(s,37*pck);              % Read and Save
```

```
fclose(s)                       % Close port
delete(s)
clear s
disp(['End of Reading # ',num2str(round)])
round = round+1;
Pos_1 = find(save1 == 126);     PD1 = Pos_1(1,1);
Pos_2 = find(save1 == 66);      PD2 = Pos_2(1,1);
Pos_3 = find(save1 == 255);     PD3 = Pos_3(1,1);
   if (PD2-PD1 == 1) && (PD3-PD2 == 1)&&(save1(pck*37,1) == 126)
      disp(['Now saving'])
      Data_Base(:,1) = save1;
      dlmwrite('Data_ECpHLight.txt', Data_Base,'-append','delimiter','\t','newline','pc');
      Cp = Cp+1;
      failed = 1;
   else disp(['Failed reading # ',num2str(Fl)])
      disp(['Try reading packet again!'])
      Fl = Fl+1;
      failed = 0;
   end
 end
%=========================================================================
%                   Part 3 :: Extract Data
%=========================================================================
    DTB = Data_Base;
    clear Data_Base;
    PP3 = 1;    % Used in line 20
    PP1 = 1;    % Used in line 31
```

```
%---------------------
% Choose the channel %
%---------------------
CH_EC_ph = 7;
CH_Light = 1;
Pos_1 = find(DTB(:,:) == 126);
Pos_2 = find(DTB(:,:) == 66);
[rw_p1 cl_p1] = size(Pos_1);
%--------------------------
% Extract only header-126
%--------------------------
for P1 = 1:2:rw_p1
   Pos_H1(PP1,:) = Pos_1(P1,1);
   PP1 = PP1+1;
end
%=========================%
[rw_pH1 cl_pH1] = size(Pos_H1);
[rw_p2 cl_p2] = size(Pos_2);
[rw_tm cl_tm] = size(DTB);
%------------------------
% Packet classification
%------------------------
pt2 = 1;
pt_EC_ph = 1;
pt_Light = 1;
Ex_Data_EC_pH = [];
Ex_Data_Light = [];
for tm = 1:cl_tm
% tm = 3;
```

```
  for pt = 1:pck
     Ex_pData(:,pt) = DTB((Pos_H1(pt,1):Pos_H1(pt,1)+36),tm);
     if (Ex_pData(13,pt) == CH_EC_ph)
        Ex_Data_EC_pH(:,pt_EC_ph) = Ex_pData(:,pt);
        pt_EC_ph = pt_EC_ph + 1;
     elseif (Ex_pData(13,pt) == CH_Light)
        Ex_Data_Light(:,pt_Light) = Ex_pData(:,pt);
        pt_Light = pt_Light + 1;
     end
  end
  [dta1 pck_EC_pH] = size(Ex_Data_EC_pH);
  [dta2 pck_Light] = size(Ex_Data_Light);
%-----------------------
% Extract EC_pH packet
%-----------------------
  a=1;
  for b = 1:pck_EC_pH;
     for c = 15:1:34;  % Data zone @ #15 - #34
        data_EC_pH(a) = Ex_Data_EC_pH(c,b);        %Data set [1x60]
        a = a+1;
     end
  end
%-----------------------
%  Extract Light packet
%-----------------------
  d=1;
  for e = 1:pck_Light;
     for f = 15:1:34;  % Data zone @ #15 - #34
        data_Light(d) = Ex_Data_Light(f,e);        %Data set [1x60]
```

```
        d = d+1;
      end
    end
%=========================================================================
%                Part 4 :: Generate ADC Data
%=========================================================================
    [b a] = size(data_EC_pH);
    [c d] = size(data_Light);
    Gen_EC_pH = repmat(256,1,a);
    Gen_Light = repmat(256,1,d);
    MostBit_EC_pH = data_EC_pH.*Gen_EC_pH   ;          %256x3 [1x60]
    MostBit_Light = data_Light.*Gen_Light   ;
    [o1 Mbit_EC_pH] = size(MostBit_EC_pH);
    [o2 Mbit_Light] = size(MostBit_Light);
    %-------------------------------------
    % Extract even row from 256-matrix
    %-------------------------------------
    x = 1;
    for q = 2:2:Mbit_EC_pH;
      RdataH_EC_pH(x) = MostBit_EC_pH(q);
      x = x+1;
    end
    y = 1;
    for r = 2:2:Mbit_Light;
      RdataH_Light(y) = MostBit_Light(r);
      y = y+1;
    end
```

```
        %---------------------------------
        % Extract odd row from data-matrix
        %---------------------------------
        xx = 1;
        for qq = 1:2:Mbit_EC_pH;
           RdataL_EC_pH(xx) = data_EC_pH(qq);
           xx = xx + 1;
        end
        yy = 1;
        for rr = 1:2:Mbit_Light;
           RdataL_Light(yy) = data_Light(rr);
           yy = yy + 1;
        end
        Finaldata_EC_pH = RdataH_EC_pH + RdataL_EC_pH;
        Finaldata_Light = RdataH_Light + RdataL_Light;
% dlmwrite('ADCdata_EC_pH.txt', Finaldata_EC_pH,'-append','delimiter','\t','precision',
'%.4f','newline','pc');
% dlmwrite('ADCdata_Light.txt', Finaldata_Light,'-append','delimiter','\t','precision',
'%.4f','newline','pc');
%=========================================================================
%                Part 5 :: Sensor Calibration
%=========================================================================
        % For Ch A0-A6
        v_ph = 2.5 * Finaldata_EC_pH/4096;              % For Ch A5
        v_EC = 2.5 * Finaldata_EC_pH/4096;              % For Ch A2
        v_Light = Finaldata_Light;
        %--------------------
        %    pH
        %--------------------
```

```
pH = (-3.838*(v_ph))+13.820;
%--------------------
%   EC (mS/cm)
%--------------------
Range = 1;
if Range == 1;
    x = [0.016039119 0.038890199 0.087086462 0.186057247 0.213877191 0.343188847
0.522154744 0.609240016 0.719247844 0.852108802 0.918182224 1.124047134 1.173505606
1.308987553  1.393643030 1.436430317 1.58863];
    y = [0 0.1 0.2 0.3 0.4 0.6 1 1.2 1.4 1.6 1.8 2.1 2.3 2.5 2.6 2.9 3.0];
    EC = interp1(x,y,[v_EC],'spline');
else
    x = [0.195042787 0.2681413 0.330247988 0.405772888 0.472091184 0.542332887
0.609776895 0.694324126];
    y = [3 4 5 6 7 8 9 10];
    EC = interp1(x,y,[v_EC],'spline');
end
%--------------------
%   Light (LUX)
%--------------------
Range = 2;
if Range == 1;
    x = [0 7 9 19 31 42 474 529 620 751
 789 822 848 865 880 885 891 899 905 909 915
 921 925 928 930 934 937 939 940 942 944 946
 946 946 946 947 947 948 949 949 949 949 949
 949 950 951 954 956 957 964 964 964 966 966
 967 967 967 968 968 969 970 970 972 973 973
 974 974 974 974 974 975 975 975 975 975 976
```

976 976 976 976 976 977 977 977 977 977 977

977 977 977 977 977 977 977 977 978 978 978

978 978 978 978 978 979 979 979 979 979 980

980 980 980 981 981 981 982 982 982 982 982

982 982 983 983 983 983 983 983 983 983 983

983 984 984 984 985 985 985 985 985 986 986

986 986 986 986 987 987 987 987 987 987 987

987 987 988 988 988 989 989 989 989 989 989

989 989 989 989 989 989 989 989 989 990 990

990 990 990 990 990 990 990 990 990 991 991

991 991 991 991 991 991 991 991 991 991 991

991 991 991 991 991 991 992 992 992 992 992

992 992 992 992 992 992 992 992 993 993 993

993 993 993 993 993 993 993 993 993 994 994

994 994 994 994 994 994 994 994 994 994 994

994 994 995 995 995 995 995 995 995 995 995

995 995 996 996 997 997 997 998] ;

y = [0 0 0.01 0.015 0.025 0.03 0.25 5 10 15

20 25 30 35 40 45 50 55 60 65 70

75 80 85 90 95 100 105 108 112 118 141

138 135 139 144 145 147 153 150 151 152 153

153 157 164 177 185 190 245 245 245 255 250

290 280 270 300 300 320 325 325 350 360 370

378.2 378.2 378.2 378.2 378.2 404.4 404.4 404.4 404.4 404.4 464.333

464.333 464.333 464.333 464.333 464.333 517.21143 517.21143 517.21143

517.21143 517.21143 517.21143 517.21143 517.21143

517.21143 517.21143 517.21143 517.21143 517.21143

517.21143 543 543 543 543 543 543 543 543 764.8

764.8 764.8 764.8 764.8 984 984 984 984 1036 1036 1036

1160 1160 1160 1160 1160 1160 1160 1181 1181 1181 1181
1181 1181 1181 1181 1181 1181 1190 1190 1190 1200 1200
1200 1200 1200 1230 1230 1230 1230 1230 1230 1368 1368
1368 1368 1368 1368 1368 1368 1368 1374.133 1374.133
1374.133 1374.133 1374.133 1374.133 1374.133
1374.133 1374.133 1374.133 1374.133 1374.133
1374.133 1374.133 1374.133 1374.133 1374.133
1374.133 1536.818182 1536.818182 1536.818182 1536.818182
1536.818182 1536.818182 1536.818182 1536.818182 1536.818182
1536.818182 1536.818182 1581.526316 1581.526316 1581.526316
1581.526316 1581.526316 1581.526316 1581.526316 1581.526316
1581.526316 1581.526316 1581.526316 1581.526316 1581.526316
1581.526316 1581.526316 1581.526316 1581.526316 1581.526316
1581.526316 2142.769231 2142.769231 2142.769231 2142.769231
2142.769231 2142.769231 2142.769231 2142.769231 2142.769231
2142.769231 2142.769231 2142.769231 2142.769231 2214 2214 2214
2214 2214 2214 2214 2214 2214 2214 2214 2214 2275.533333
2275.533333 2275.533333 2275.533333 2275.533333 2275.533333
2275.533333 2275.533333 2275.533333 2275.533333 2275.533333
2275.533333 2275.533333 2275.533333 2275.533333 2460 2460 2460
2460 2460 2460 2460 2460 2460 2460 2480 2480 2490
2490 2490 2500] ;

[x1p, idx] = unique(x);

y1p = y(idx);

LUX = interp1(x1p,y1p,v_Light);

else

x = [0 7 9 19 31 42 474 529 620 751
789 822 848 865 880 885 891 899 905 909 915
921 925 928 930 934 937 939 940 942 944 946

946 946 946 947 947 948 949 949 949 949 949

949 950 951 954 956 957 964 964 964 966 966

967 967 967 968 968 969 970 970 972 973 973

974 974 974 974 974 975 975 975 975 975 976

976 976 976 976 976 977 977 977 977 977 977

977 977 977 977 977 977 977 977 978 978 978

978 978 978 978 978 979 979 979 979 979 980

980 980 980 981 981 981 982 982 982 982 982

982 982 983 983 983 983 983 983 983 983 983

983 984 984 984 985 985 985 985 985 986 986

986 986 986 986 987 987 987 987 987 987 987

987 987 988 988 988 989 989 989 989 989 989

989 989 989 989 989 989 989 989 989 990 990

990 990 990 990 990 990 990 990 990 991 991

991 991 991 991 991 991 991 991 991 991 991

991 991 991 991 991 991 992 992 992 992 992

992 992 992 992 992 992 992 992 993 993 993

993 993 993 993 993 993 993 993 993 994 994

994 994 994 994 994 994 994 994 994 994 994

994 994 995 995 995 995 995 995 995 995 995

995 995 996 996 997 997 997 998] ;

y = [0 0 0.01 0.015 0.025 0.03 0.25 5 10 15

20 25 30 35 40 45 50 55 60 65 70

75 80 85 90 95 100 105 108 112 118 124

127 129 130 132 135 135 136 138 138 141 144

147 153 164 177 185 190 205 245 245 250 255

263 275 280 284 290 296 300 305 320 350 370

372 374 380 380 382 385 400 400 405 410 420

426 430 437 450 453 454 456 458 467 475 480

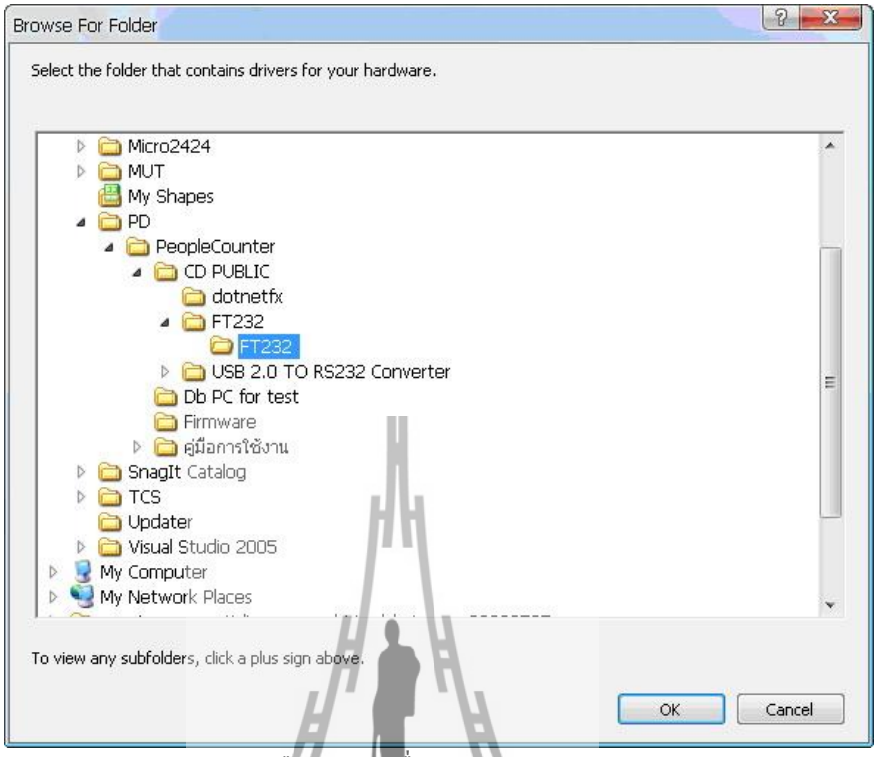
```
489 498 505 515 515 517 525 525 526 529 530
535 537 543 545 550 555 557 565 567 590 607
611 626 627 629 630 636 640 641 717 755 782
840 841 842 844 845 850 850 857 860 861 864
870 880 900 921 932 946 955 959 961 961 966
968 970 976 993 995 996 996 1004 1006 1011 1020
1020 1035 1035 1036 1060 1061 1070 1084 1090 1090 1100
1111 1116 1122 1122 1131 1132 1140 1150 1150 1157 1160
1160 1172 1172 1172 1175 1181 1190 1190 1195 1217 1217
1218 1220 1220 1220 1220 1240 1242 1262 1281 1301 1305
1314 1345 1368 1423 1427 1430 1436 1505 1515 1555 1636
1636 1639 1655 1662 1674 1674 1690 1690 1697 1700 1701
1704 1705 1716 1736 1736 1770 1815 2120 2190 2190 2240
2240 2250 2250 2260 2260 2270 2270 2300 2350 2370 2410
2410 2450 2450 2460 2480 2510 2520 2550 2560 2576 2610
2610 2610 2610 2610 2620 2620 2660 2666]                    ;
        [x1p, idx] = unique(x);
        y1p = y(idx);
        LUX = interp1(x1p,y1p,v_Light);
    end
    Light = LUX*10;
%=========================================================================
%           Part 6 :: Choose Data & Save
%=========================================================================
    [r_EC_pH c_EC_pH] = size(Finaldata_EC_pH);
    [r_Light c_Light] = size(Finaldata_Light);
    MT = 1;
    TP = 1;
```

```
        for i=1:c_EC_pH
            if mod(i,2) == 1
                EC_real(tm,MT) = EC(:,i);
                MT = MT+1;
            elseif mod(i,2) == 0
                pH_real(tm,TP) = pH(i);
                TP = TP+1;
            end
        end
        mean_EC = mean(EC_real)
        mean_pH = mean(pH_real)
%           Aver_pH = sum(pH_real(1:(pck*10/2),1))/(pck*10/2)
%           Aver_EC = sum(EC_real(1:(pck*10/2),1))/(pck*10/2)
        mean_Light = mean(Light)
        save_time = clock;
        save_data = [save_time(1:5),mean_EC,mean_pH,mean_Light];
%           dlmwrite('EC_pH_Light_data.txt',save_data,'-append','delimiter','\t','newline','pc');
        dlmwrite('Database_Hydroponics.dat',save_data,'-append','delimiter','\t','newline','pc');
%           dlmwrite('Result.dat',save_data(6:8),'-append','delimiter','\t','precision',
'%.4f','coffset',1,'newline','pc');
        c = clock;
        disp(['Saved at ',num2str(c(4)),'.',num2str(c(5))])
    end
    pause(30);
end
%==========================================================================
%                       End of Program
%==========================================================================
```

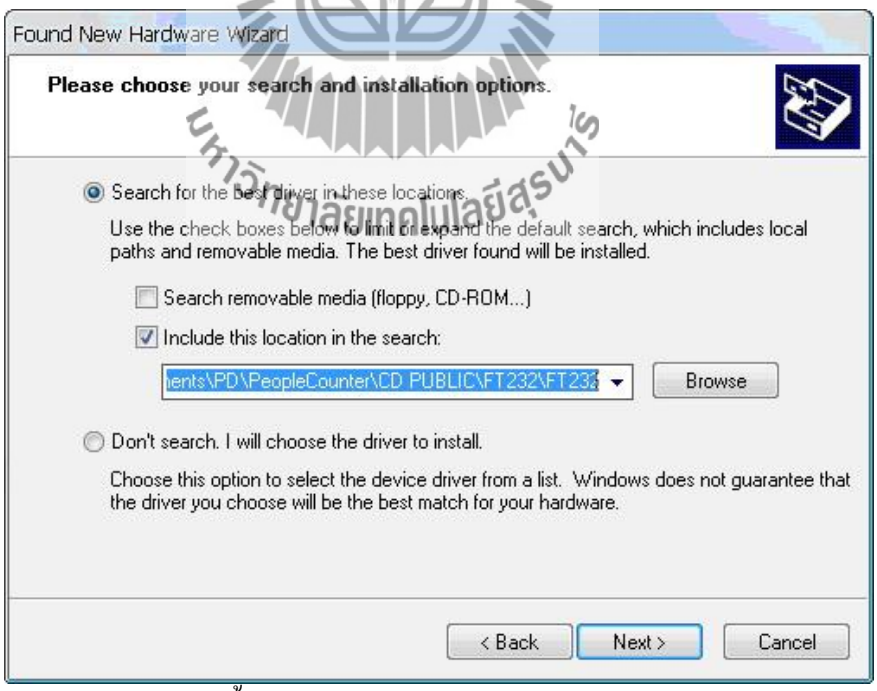
6. วิธีการติดตั้ง Driver RS232 และการดู Com Port

คลิกเลือกที่ Install from a list or specific location (Advanced) และกดปุ่ม Next

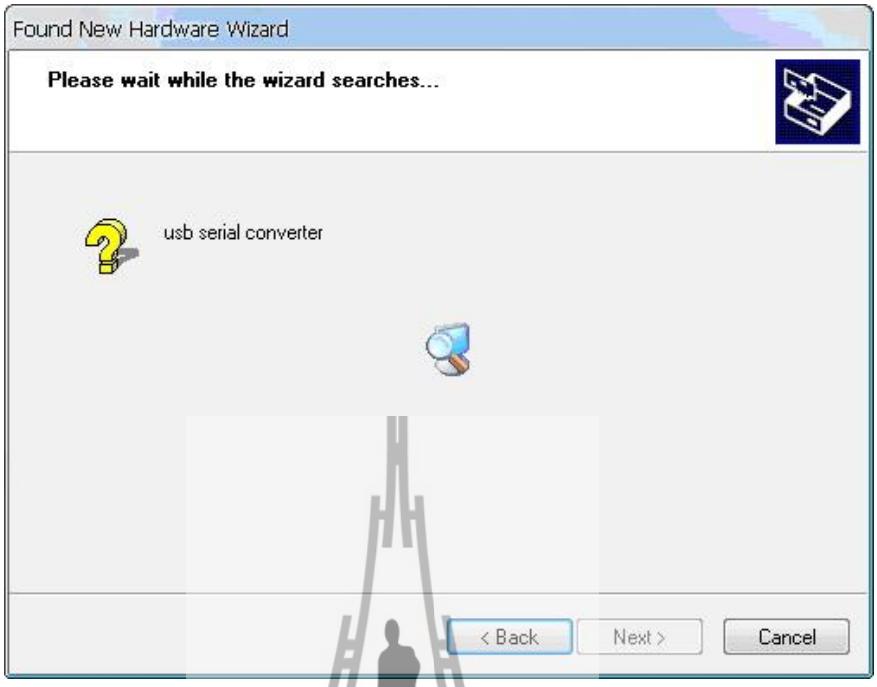
จากนั้นจะแสดงหน้าต่างดังรูป คลิกเลือก Include this location in the search และกดปุ่ม Next

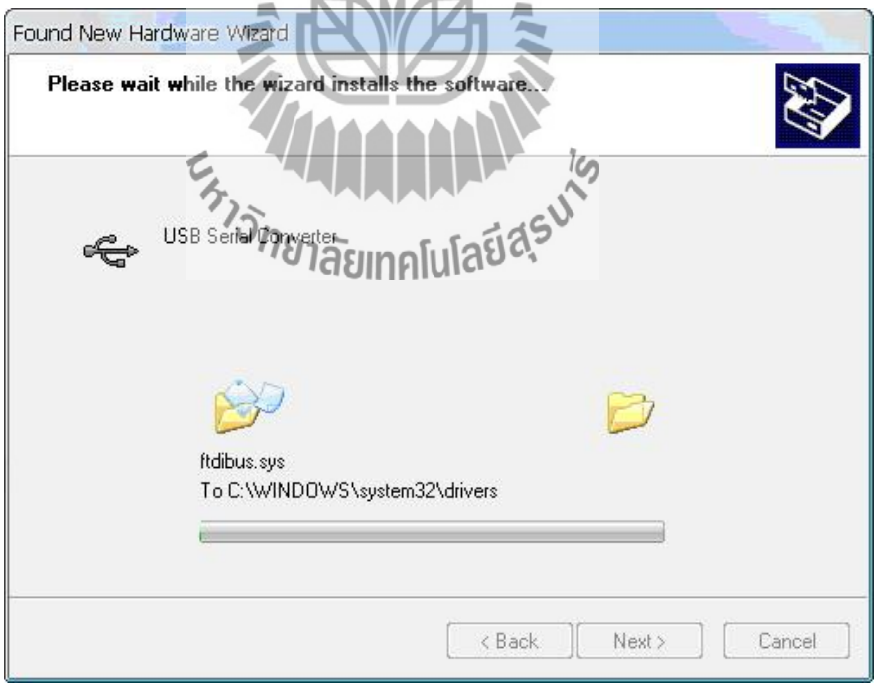


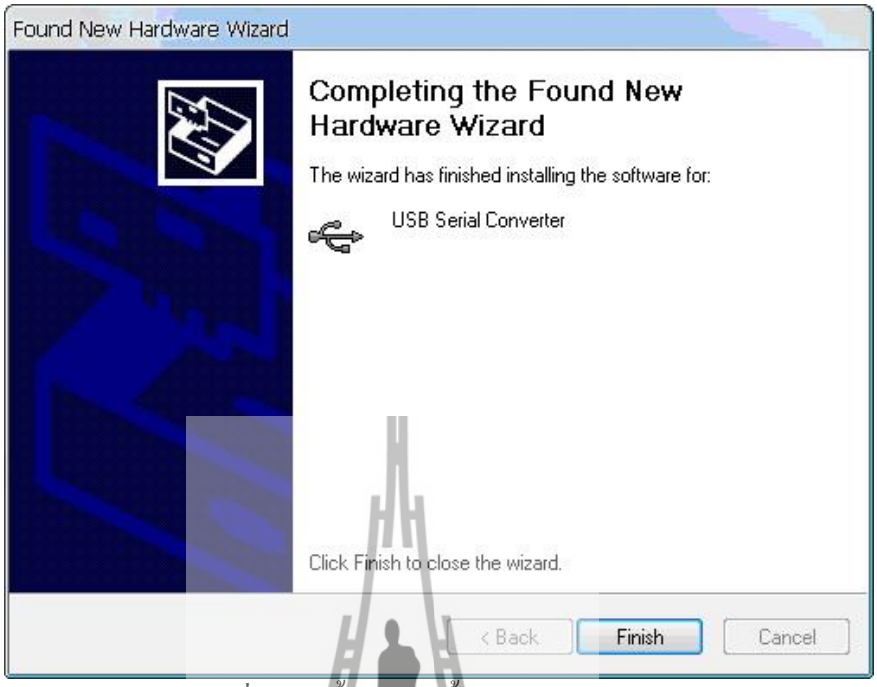
เลือก Folder ชื่อ FT232 กดปุ่ม OK



จากนั้นจะแสดงหน้าต่างดังรูป และกดปุ่ม Next



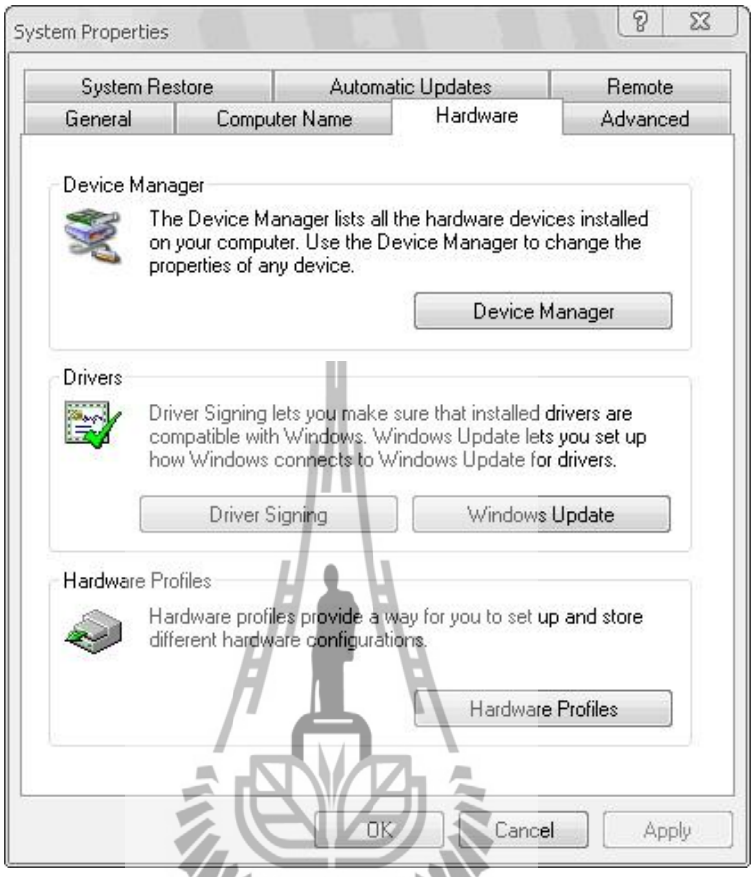




กดปุ่ม Finish เพื่อเสร็จสิ้นการติดตั้ง Driver ของ Port USB-RS232

คลิกขวาที่ My Computer

เมื่อแสดงหน้าต่างดังรูป เลือก Hardware และกดปุ่ม Device Manager



คลิกที่ Ports (COM & LPT) จะเห็น COM PORT ดังรูปตัวอย่าง

ภาคผนวก ข

คู่มือเซ็นเซอร์

pH Sensor

pH Sensor
(Order Code PH-BTA)

Our pH Sensor can be used for any lab or demonstration that can be done with a traditional pH meter. This sensor offers the additional advantages of automated data collection, graphing, and data analysis. Typical activities using our pH sensor include studies of household acids and bases, acid-base titrations, monitoring pH change during chemical reactions or in an aquarium as a result of photosynthesis, investigations of acid rain and buffering, and investigations of water quality in streams and lakes.

Collecting Data with the pH Sensor

This sensor can be used with the following interfaces to collect data:

- Vernier LabQuest® as a standalone device or with a computer
- Vernier LabQuest® Mini with a computer
- Vernier LabPro® with a computer, TI graphing calculator, or Palm® handheld
- Vernier Go!®Link
- Vernier EasyLink®
- Vernier SensorDAQ®
- CBL 2™

Here is the general procedure to follow when using the pH Sensor:

1. Connect the pH Sensor to the interface.
2. Start the data-collection software[1].
3. The software will identify the pH Sensor and load a default data-collection setup. You are now ready to collect data.

Important: Do not fully submerge the sensor. The handle is not waterproof.

Data-Collection Software

This sensor can be used with an interface and the following data-collection software.

- **Logger *Pro* 3** This computer program is used with LabQuest, LabQuest Mini, LabPro, or Go!Link.
- **Logger *Pro* 2** This computer program is used with ULI or Serial Box Interface.
- **Logger Lite** This computer program is used with LabQuest, LabQuest Mini, LabPro, or Go!Link.
- **LabQuest App** This program is used when LabQuest is used as a standalone device.
- **EasyData App** This calculator application for the TI-83 Plus and TI-84 Plus can be used with CBL 2, LabPro, and Vernier EasyLink. We recommend version 2.0 or newer, which can be downloaded from the Vernier web site, www.vernier.com/easy/easydata.html, and then transferred to the calculator.

[1] If you are using Logger *Pro* 2 with either a ULI or SBI, the sensor will not auto-ID. Open an experiment file for the pH Sensor in the Probes & Sensors folder.

See the Vernier web site, www.vernier.com/calc/software/index.html for more information on the App and Program Transfer Guidebook.

- **DataMate program** Use DataMate with LabPro or CBL 2 and TI-73, TI-83, TI-84, TI-86, TI-89, and Voyage 200 calculators. See the LabPro and CBL 2 Guidebooks for instructions on transferring DataMate to the calculator.
- **Data Pro** This program is used with LabPro and a Palm handheld.
- **LabVIEW** National Instruments LabVIEW™ software is a graphical programming language sold by National Instruments. It is used with SensorDAQ and can be used with a number of other Vernier interfaces. See www.vernier.com/labview for more information.

pH Electrode Specifications

Type:	Sealed, gel-filled, epoxy body, Ag/AgCl
Response time:	90% of final reading in 1 second
Temperature range:	5 to 80°C
12 mm OD	
Range:	pH 0–14
13-bit Resolution (SensorDAQ):	0.0025 pH units
12-bit Resolution (LabQuest, LabQuest Mini, Go!Link, LabPro, ULI, SBI):	0.005 pH units
10-bit Resolution (CBL 2):	0.02 pH units
Isopotential pH:	pH 7 (point at which temperature has no effect)
Output:	59.2 mV/pH at 25°C
Stored Calibration Values[2]:	
Intercept (k_0):	13.720
Slope (k_1):	–3.838

NOTE: This product is to be used for educational purposes only. It is not appropriate for industrial, medical, research, or commercial applications.

How the pH Sensor Works

The pH Amplifier inside the handle is a circuit which allows a standard combination pH electrode (such as the Vernier 7120B) to be monitored by a lab interface. The cable from the pH Amplifier ends in a BTA plug.

The pH Sensor will produce a voltage of 1.75 volts in a pH 7 buffer. The voltage will increase by about 0.25 volts for every pH number decrease. The voltage will decrease by about 0.25 volts/pH number as the pH increases.

The Vernier gel-filled pH Sensor is designed to make measurements in the pH range of 0 to 14. A polycarbonate body that extends below the glass sensing bulb of the

[2] These are average calibration values. Actual values may vary because sensors are individually calibrated by Vernier before shipping.

electrode makes this probe ideal for the demands of a middle school, high school, or university level science class or for making measurements in the environment. The gel-filled reference half cell is sealed—it never needs to be refilled.

This sensor is equipped with circuitry that supports auto-ID. When used with LabQuest, LabQuest Mini, LabPro, Go! Link, SensorDAQ, EasyLink, or CBL 2, the data-collection software identifies the sensor and uses pre-defined parameters to configure an experiment appropriate to the recognized sensor.

Preparing for Use

To prepare the electrode to make pH measurements, follow this procedure:

- Remove the storage bottle from the electrode by first unscrewing the lid, then removing the bottle and lid. Thoroughly rinse the lower section of the probe, especially the region of the bulb, using distilled or deionized water.
- When the probe is not being stored in the storage bottle, it can be stored for short periods of time (up to 24 hours) in pH-4 or pH-7 buffer solution. It should never be stored in distilled water.
- Connect the pH Sensor to your lab interface, load or perform a calibration (as described in the next section), and you are ready to make pH measurements. **Note:** Do not completely submerge the sensor. The handle is not waterproof.

When you are finished making measurements, rinse the tip of the electrode with distilled water. Slide the cap onto the electrode body, then screw the cap onto the storage bottle. Note: When the level of storage solution left in the bottle gets low, you can replenish it with small amounts of tap water the first few times you use the probe (but not indefinitely!). A better solution is to prepare a quantity of pH-4 buffer/KCl storage solution (see the section on Maintenance and Storage) and use it to replace lost solution.

Do I Need to Calibrate the pH Sensor?

We feel that you should not have to perform a new calibration when using the pH Sensor for most experiments in the classroom. We have set the sensor to match our stored calibration before shipping it. You can simply use the appropriate calibration file that is stored in your data-collection program from Vernier in any of these ways:

1. If you ordered the PH-BTA version of the sensor, and you are using it with a LabQuest, LabQuest Mini, LabPro or CBL 2 interface, then a calibration (in pH) is automatically loaded when the pH Sensor is connected. **Note:** Each pH Sensor (PH-BTA version) is calibrated at Vernier. This custom calibration is then stored on the sensor. This means that when you first use it, you will see pH readings that are accurate to +/- 0.10 pH units, without calibration! With time, you may see some minor loss of the initial custom calibration accuracy, but for most purposes (see below), it should not be necessary to calibrate the pH Sensor.
2. If you are using Logger *Pro* software (version 2.0 or newer) on a Macintosh or Windows computer, open an experiment file for the pH Sensor, and its stored calibration will be loaded at the same time. **Note:** If you have an earlier version of Logger *Pro*, a free upgrade is available from our web site.

3. Any version of the DataMate or EasyData program (with LabPro or CBL 2) has stored calibrations for this sensor.
4. Any version of Data Pro has stored calibrations for this sensor.

If you are performing a chemistry experiment, or doing water quality testing that requires a very accurate calibration, you can calibrate the Vernier pH Electrode following this procedure:

- Use the 2-point calibration option of the Vernier data-collection program. Rinse the tip of the electrode in distilled water. Place the electrode into one of the buffer solutions (e.g., pH 4). When the voltage reading displayed on the computer or calculator screen stabilizes, enter a pH value, "4".
- For the next calibration point, rinse the electrode and place it into a second buffer solution (e.g., pH 7). When the displayed voltage stabilizes, enter a pH value, "7".
- Rinse the electrode with distilled water and place it in the sample.

pH Buffer Solutions

In order to do a calibration of the pH Sensor, or to confirm that a saved pH calibration is accurate, you need to have a supply of pH buffer solutions that cover the range of pH values you will be measuring. We recommend buffer solutions of pH 4, 7, and 10.

- Vernier sells a pH buffer kit (order code PHB). The kit has 12 tablets: four tablets each of buffer pH 4, 7, and 10. Each tablet is added to 100 mL of distilled or deionized water to prepare respective pH buffer solutions.
- Flinn Scientific (www.flinnsci.com, Tel: 800-452-1261) sells a wide variety of buffer tablets and prepared buffer solutions.
- You can prepare your own buffer solutions using the following recipes:

pH 4.00	Add 2.0 mL of 0.1 M HCl to 1000 mL of 0.1 M potassium hydrogen phthalate.
pH 7.00	Add 582 mL of 0.1 M NaOH to 1000 mL of 0.1 M potassium dihydrogen phosphate.
pH 10.00	Add 214 mL of 0.1 M NaOH to 1000 mL of 0.05 M sodium bicarbonate.

Maintenance and Storage

Short-term storage (up to 24 hours): Place the electrode in pH-4 or pH-7 buffer solution.

Long-term storage (more than 24 hours): Store the electrode in a buffer pH-4/KCl storage solution in the storage bottle. The pH Electrode is shipped in this solution. Vernier sells 500 mL bottles of replacement pH Storage Solution (order code PH-SS), or you can prepare additional storage solution by adding 10 g of solid potassium chloride (KCl) to 100 mL of buffer pH-4 solution. Flinn Scientific (800- 452-1261) sells a Buffer Solution Preservative (order code B0175) that can be added to this storage solution. By storing the electrode in this solution, the reference portion of the electrode is kept moist. Keeping the reference junction moist adds to electrode longevity and retains electrode response time when the unit is placed back into

service. If the electrode is inadvertently stored dry (we don't recommend this!), immerse the unit in soaking solution for a minimum of eight hours prior to service.

When testing a pH Sensor, it is best to place it into a known buffer solution. This allows you to see if the sensor is reading correctly (e.g., in a buffer pH 7, is the sensor reading close to pH 7). Do not place your sensor into distilled water to check for readings—distilled water can have a pH reading anywhere between 5.5 and 7.0, due to variable amounts of carbon dioxide dissolved from the atmosphere. Furthermore, due to a lack of ions, the pH values reported with the sensor in distilled water will be erratic.

If your pH Sensor is reading slightly off of the known buffer pH (e.g., reads 6.7 in a buffer 7), you may simply need to calibrate the sensor. You can calibrate the sensor in two buffer solutions for two calibration points. If you do not remember or know how to perform a calibration, refer to the booklet that came with the pH sensor.

If your readings are off by several pH values, the pH readings do not change when moved from one buffer solution to another different buffer, or the sensor's response seems slow, the problem may be more serious. Sometimes a method called "shocking" is used to revive pH electrodes. To shock your pH Sensor, perform the following:

1. Let the pH Electrode soak for 4-8 hours in an HCl solution between 0.1 and 1.0 M.
2. Rinse off the electrode and let it sit in some buffer pH 7 for an hour or so.
3. Rinse the electrode and give it another try.

Mold growth in the buffer/KCl storage solution can be prevented by adding a commercial growth inhibitor. This mold will not harm the electrode and can easily be removed using a light detergent solution.

This sensor is designed to be used in aqueous solutions. The polycarbonate body of the sensor can be damaged by many organic solvents. In addition, do not use the sensor in solutions containing: perchlorates, silver ions, sulfide ions, biological samples with high concentrations of proteins, or Tris buffered solutions.[3] Do not use it in hydrofluoric acid or in acid or basic solutions with a concentration greater than 1.0 molar. The electrode may be used to measure the pH of sodium hydroxide solutions with a concentration near 1.0 molar, but should not be left in this concentration of sodium hydroxide for periods longer than 5 minutes. Using or storing the electrode at very high temperatures or very low temperatures (near 0°C) can damage it beyond repair.

Warranty

Vernier warrants this product to be free from defects in materials and workmanship for a period of five years from the date of shipment to the customer. This warranty does not cover damage to the product caused by abuse or improper use. Additionally, the warranty does not cover accidental breakage of the glass bulb of the pH Sensor.

eC Sensor

Conductivity Probe
(Order Code CON-BTA)

The Conductivity Probe can be used to measure either solution conductivity or total ion concentration of aqueous samples being investigated in the field or in the laboratory. Conductivity is one of the easiest environmental tests of aquatic samples. Even though it does not tell you specific ions that are present, it does quickly determine the total concentration of ions in a sample. It can be used to perform a wide variety of tests or planned experiments to determine the changes in or levels of total dissolved ions or salinity:

- Allow students to qualitatively see the difference between the ionic and molecular nature of substance in aqueous solution. This can include differences in strength of weak acids and bases, or the number of ions that an ionic substance dissociates into per formula unit.
- Use the probe to confirm the direct relationship between conductivity and ion concentration in an aqueous solution. Concentrations of unknown samples can then be determined.
- Measure changes in conductivity resulting from photosynthesis in aquatic plants, with the resulting decrease in bicarbonate-ion concentration from carbon dioxide.
- Use this sensor for an accurate on-site measurement of total dissolved solids (TDS) in a stream or lake survey.
- Monitor the rate of reaction in a chemical reaction in which dissolved ions and solution conductivity varies with time due to an ionic specie being consumed or produced.
- Perform a conductivity titration to determine when stoichiometric quantities of two substances have been combined.
- Use the Conductivity Probe to determine the rate at which an ionic species diffuses through a membrane, such as dialysis tubing.
- Monitor changes in conductivity or total dissolved solids in an aquarium containing aquatic plants *and* animals. These changes could be due to photosynthesis *or* respiration.

Collecting Data with the Conductivity Probe

This sensor can be used with the following interfaces to collect data:

- Vernier LabQuest® as a standalone device or with a computer
- Vernier LabQuest® Mini with a computer
- Vernier LabPro® with a computer, TI graphing calculator, or Palm® handheld
- Vernier Go!®Link
- Vernier EasyLink®
- Vernier SensorDAQ®
- CBL 2™

Here is the general procedure to follow when using the Conductivity Probe:

1. Connect the Conductivity Probe to the interface.
2. Start the data-collection software[1].
3. The software will identify the Conductivity Probe and load a default data-collection setup. You are now ready to collect data.

Data-Collection Software

This sensor can be used with an interface and the following data-collection software.

- **Logger *Pro* 3** This computer program is used with LabQuest, LabQuest Mini, LabPro, or Go!Link.
- **Logger *Pro* 2** This computer program is used with ULI or Serial Box Interface.
- **Logger Lite** This computer program is used with LabQuest, LabQuest Mini, LabPro, or Go!Link.
- **LabQuest App** This program is used when LabQuest is used as a stan-alone device.
- **EasyData App** This calculator application for the TI-83 Plus and TI-84 Plus can be used with CBL 2, LabPro, and Vernier EasyLink. We recommend version 2.0 or newer, which can be downloaded from the Vernier web site, www.vernier.com/easy/easydata.html, and then transferred to the calculator. See the Vernier web site, www.vernier.com/calc/software/index.html for more information on the App and Program Transfer Guidebook.
- **DataMate program** Use DataMate with LabPro or CBL 2 and TI-73, TI-83, TI-84, TI-86, TI-89, and Voyage 200 calculators. See the LabPro and CBL 2 Guidebooks for instructions on transferring DataMate to the calculator.
- **Data Pro** This program is used with LabPro and a Palm handheld.
- **LabVIEW** National Instruments LabVIEW™ software is a graphical programming language sold by National Instruments. It is used with SensorDAQ and can be used with a number of other Vernier interfaces. See www.vernier.com/labview for more information.

NOTE: This product is to be used for educational purposes only. It is not appropriate for industrial, medical, research, or commercial applications.

Taking Measurements with the Conductivity Probe

- Rinse the tip of the Conductivity Probe with distilled water. Optional: Blot the inside of the electrode cell dry only if you are concerned about water droplets diluting or contaminating the sample to be tested.
- Insert the tip of the probe into the sample to be tested. **Important:** Be sure the electrode surfaces in the elongated cell are completely submerged in the liquid.
- While gently swirling the probe, wait for the reading on your computer, calculator screen, or Palm device to stabilize. This should take no more than 5 to 10 seconds. **Note:** Do not completely submerge the sensor. The handle is not waterproof.

---

[1] If you are using Logger *Pro* 2 with either a ULI or SBI, the sensor will not auto-ID. Open an experiment file for the Conductivity Probe in the Probes & Sensors folder.

- Rinse the end of the probe with distilled water before taking another measurement.
- If you are taking readings at temperatures below 15°C or above 30°C, allow more time for the temperature compensation to adjust and provide a stable conductivity reading.
- **Important:** Do not place the electrode in viscous, organic liquids, such as heavy oils, glycerin (glycerol), or ethylene glycol. Do not place the probe in acetone or non-polar solvents, such as pentane or hexane.

Storage and Maintenance of the Conductivity Probe

- When you have finished using the Conductivity Probe, simply rinse it off with distilled water and blot it dry using a paper towel or lab wipe. The probe can then be stored dry.
- If the probe cell surface is contaminated, soak it in water with a mild detergent for 15 minutes. Then soak it in a dilute acid solution (0.1 M hydrochloric acid or 0.5 M acetic acid works well) for another 15 minutes. Then rinse it well with distilled water. **Important** Avoid scratching the inside electrode surfaces of the elongated cell.

This sensor is equipped with circuitry that supports auto-ID. When used with LabQuest, LabQuest Mini, LabPro, Go! Link, SensorDAQ, EasyLink, or CBL 2, the data-collection software identifies the sensor and uses pre-defined parameters to configure an experiment appropriate to the recognized sensor.

Specifications

Range of Conductivity Probe:

- Low Range: 0 to 200 µS/cm (0 to 100 mg/L TDS)
- Mid Range: 0 to 2000 µS/cm (0 to 1000 mg/L TDS)
- High Range: 0 to 20,000 µS/cm (0 to 10,000 mg/L TDS)

13-bit Resolution (with SensorDAQ):

- Low Range: 0.05 µS/cm (0.025 mg/L TDS)
- Mid Range: 0.5 µS/cm (0.25 mg/L TDS)
- High Range: 5 µS/cm (2.5 mg/L TDS)

12-bit Resolution (with LabQuest, LabQuest Mini, LabPro, Go!Llink, EasyLink):

- Low Range: 0.1 µS/cm (0.05 mg/L TDS)
- Mid Range: 1 µS/cm (0.5 mg/L TDS)
- High Range: 10 µS/cm (5 mg/L TDS)

10-bit Resolution (with CBL 2):

- Low Range: 0.4 µS/cm (0.2 mg/L TDS)
- Mid Range: 4 µS/cm (2.0 mg/L TDS)
- High Range: 40 µS/cm (20 mg/L TDS)

Accuracy: ±1% of full-scale reading for each range

Response time: 98% of full-scale reading in 5 seconds, 100% of full-scale in 15 seconds

Temperature compensation:	automatic from 5 to 35°C
Temperature range (can be placed in):	0 to 80°C
Cell constant:	1.0 cm^{-1}
Description:	dip type, ABS body, parallel carbon (graphite) electrodes
Dimensions:	12 mm OD and 150 mm length

How the Conductivity Probe Works

The Vernier Conductivity Probe measures the ability of a solution to conduct an electric current between two electrodes. In solution, the current flows by ion transport. Therefore, an increasing concentration of ions in the solution will result in higher conductivity values.

The Conductivity Probe is actually measuring *conductance*, defined as the reciprocal of resistance. When resistance is measured in ohms, conductance is measured using the SI unit, *siemens* (formerly known as a *mho*). Since the siemens is a very large unit, aqueous samples are commonly measured in microsiemens, or μS.

Even though the Conductivity Probe is measuring conductance, we are often interested in finding *conductivity* of a solution. Conductivity, *C*, is found using the following formula:

$$C = G \cdot k_C$$

where *G* is the conductance, and k_C is the cell constant. The cell constant is determined for a probe using the following formula:

$$k_C = d/A$$

where *d* is the distance between the two electrodes, and *A* is the area of the electrode surface.

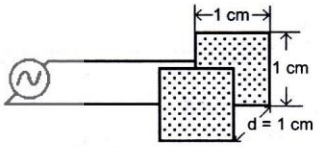


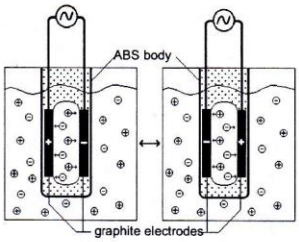
Figure 2

For example, the cell in Figure 2 has a cell constant: $k_C = d/A = 1.0\ cm/1.0\ cm^2 = 1.0\ cm^{-1}$. The conductivity value is found by multiplying conductance and the cell constant. Since the Vernier Conductivity Probe also has a cell constant of $1.0\ cm^{-1}$, its conductivity and conductance have the same numerical value. For a solution with a conductance value of 1000 μS, the conductivity, C, would be

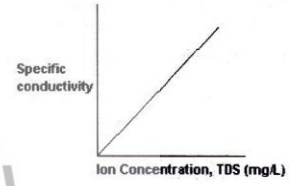
$$C = G \cdot k_C = (1000\ \mu S) \times (1.0\ cm^{-1}) = 1000\ \mu S/cm$$

A potential difference is applied to the two probe electrodes in the Conductivity Probe. The resulting current is proportional to the conductivity of the solution. This current is converted into a voltage.

Alternating current is supplied to prevent the complete ion migration to the two electrodes. As shown in the figure here, with each cycle of the alternating current, the polarity of the electrodes is reversed, which in turn reverses the direction of ion flow. This very important feature of the Conductivity Probe prevents most electrolysis and polarization from occurring at the electrodes. Thus, the solutions that are being measured for conductivity are not fouled. It also greatly reduces redox products from forming on the relatively inert graphite electrodes.



One of the most common uses of the Conductivity Probe is to find the concentration of total dissolved solids, or *TDS*, in a sample of water. This can be accomplished because there is generally a direct relationship between conductivity and the concentration of ions in a solution, as shown here. The relationship persists until very large ion concentrations are reached.



Do I Need to Calibrate the Conductivity Probe?

You should not have to perform a new calibration when using the Conductivity Probe in the classroom. We have set the sensor to match our stored calibration before shipping it. You can simply use the appropriate calibration file that is stored in your data-collection program from Vernier.

If you are using the Conductivity Probe for water quality analysis, you may choose to calibrate for more accurate readings. The Conductivity Probe can be easily calibrated at two known levels, using any of the Vernier data-collection programs. The calibration units can be µS/cm, mg/L as TDS, mg/L as NaCl, or salinity, in ppt.

- Select the conductivity range setting on the probe box: low = 0 to 200 µS, medium = 0 to 2000 µS, and high = 0 to 20,000 µS. Note: If you are not sure which setting to use, you may first want to load a stored Vernier calibration for one or more of the settings to determine an approximate value for the solution to be sampled.
- **Zero Calibration Point:** Simply perform this calibration point with the probe out of any liquid or solution (e.g., in the air). A very small voltage reading will be displayed. Call this value 0 µS or 0 mg/L.
- **Standard Solution Calibration Point:** Place the Conductivity Probe into a standard solution (solution of known concentration), such as the sodium chloride standard that is supplied with your probe. Be sure the entire elongated hole with the electrode surfaces is submerged in the solution. Wait for the displayed voltage

to stabilize. Enter the value of the standard solution (e.g., 1000 µS, 491 mg/L as NaCl, or 500 mg/L as TDS). For further information on preparing and interpreting standard solutions, see subsequent sections on calibration.

For even better results, the two-point calibration can be performed using *two* standard solutions that bracket the expected range of conductivity or concentration values you will be testing. For example, if you expect to measure conductivity in the range of 600 mg/L to 1000 mg/L (TDS), you may want to use a standard solution that is 500 mg/L for one calibration point and another standard that is 1000 mg/L for the second calibration point.

Maintaining and Replacing the Sodium Chloride Standard Calibration Solution

If you choose to calibrate the Conductivity Probe, you will want accurate standard solutions. The 1000 µS/cm Standard that shipped with the Conductivity Probe will last a long time if you take care not to contaminate it with a wet or dirty probe. This is a good concentration to calibrate your Conductivity Probe in the middle range (0 to 2000 µS/cm). Vernier sells three Conductivity Standards, one appropriate for each range of the Conductivity Probe. They come in 500 mL bottles. Order codes are:

Low Range (150 µS/cm) CON-LST
Medium Range (1413 µS/cm) CON-MST
High Range (12880 µS/cm) CON-HST

To prepare your own standard solutions using solid NaCl or KCl:

- Use a container with accurate volume markings (e.g., volumetric flask) and add the amount of solid shown in the first column of Table 1. This standard can be used to calibrate using the amount shown in mg/L as NaCl (first column), mg/L as TDS (second column), or µS/cm (third column).

Table 1

Add this amount of NaCl to make 1 liter of solution	**TDS and Conductivity values equivalent to the NaCl concentration in the first column:**	
	total dissolved solids (TDS)	**conductivity (microsiemens/cm)**
0.0474 g (47.4 mg/L)	50 mg/L as TDS	100 µS/cm
0.491 g (491 mg/L)	500 mg/L as TDS	1000 µS/cm
1.005 g (1005 mg/L)	1000 mg/L as TDS	2000 µS/cm
5.566 g (5566 mg/L)	5000 mg/L as TDS	10,000 µS/cm

- Note also that standard solutions of lower concentration can be prepared by diluting standard solutions of higher concentration. For example, if you have a solution that is 1000 mg/L, and want to dilute it to obtain a solution that is 200 mg/L, simply take 100 mL of the 1000 mg/L solution and add enough distilled water to it to yield 500 mL of solution (~400 mL of water is added). The new solution has a concentration of 1000 mg/L x (100 mL / 500 mL) = 200 mg/L.

Automatic Temperature Compensation

Your Vernier Conductivity Probe is automatically temperature compensated between temperatures of 5 and 35°C. Note that the temperature of a solution is being read by a thermistor that extends into the space between the graphite electrodes. Readings are automatically referenced to a conductivity value at 25°C—therefore the Conductivity Probe will give the same conductivity reading in a solution that is at 15°C as it would if the same solution were warmed to 25°C. This means you can calibrate your probe in the lab, and then use these stored calibrations to take readings in colder (or warmer) water in a lake or stream. If the probe was not temperature compensated, you would notice a change in the conductivity reading as temperature changed, even though the actual ion concentration did not change.

Using the Conductivity Probe with Other Vernier Sensors

It is very important to know that the Conductivity Probe will interact with some other Vernier sensors and probes, *if* they are placed in the same solution (in the same aquarium or beaker, for example), *and* they are connected to the same interface box (e.g., the same LabPro). This situation arises because the Conductivity Probe outputs a signal in the solution, and this signal can affect the reading of another probe. The following probes *cannot* be connected to the *same interface* as a Conductivity Probe and placed in the same solution:

- Dissolved Oxygen Probe
- pH System
- Ion Selective Electrodes

If you wish to take simultaneous readings using any of the probe combinations listed above, here are some alternative methods:

- To take simultaneous conductivity and dissolved oxygen or conductivity and pH readings, you can connect the probes to *two different* interface boxes. If the two probes in question are connected to separate interfaces, the two probes will read correctly in the same solution.
- If you are sampling a lake or stream and want to use two of the probes with a single interface, you can connect the two probes in question to the same interface and load their respective calibrations. Place one probe in the water first and take its reading. Then remove it and place the second probe in the solution to take its reading.

The Stainless Steel Temperature Probe can be used in the same container with the Conductivity Probe.

Sampling in Streams and Lakes

It is best to sample away from shore and below the water surface, if possible. In free-flowing streams, there will usually be good mixing of the water, so that samples taken near the current will be quite representative of the stream as a whole. If you are sampling an impounded stream or a lake, there will be very little mixing—therefore, it is important to sample away from shore and at different depths, if possible. We do not recommend that you drop the Vernier Conductivity Probe so that the entire electrode is submerged. The electrode is not constructed to withstand higher pressures, so seepage into electronic components of the electrode might result. Although it is better to take readings at the collection site, readings of total dissolved

solids or conductivity should not change significantly if you collect samples and take readings at a later time. However, be sure that samples are capped to prevent evaporation. If sample bottles are filled brim full, then a gas such as carbon dioxide, which is capable of forming ionic species in solution, is prevented from dissolving in the water sample.

Since the probe has built-in temperature compensation, you can do your calibration in the lab. This means that even though you will be sampling in water that has a different temperature than your calibration temperature, the probe will take correct readings at the new sampling temperature.

Sampling in Ocean Salt Water or Tidal Estuaries: SALINITY

Salinity is the total of all non-carbonate salts dissolved in water, usually expressed in parts per thousand (1 ppt = 1000 mg/L). Unlike chloride (Cl^-) concentration, you can think of salinity as a measure of the total salt concentration, comprised mostly of Na^+ and Cl^- ions. Even though there are smaller quantities of other ions in seawater (e.g., K^+, Mg^{2+}, or SO_4^{2-}), sodium and chloride ions represent about 91% of all seawater ions. Salinity is an important measurement in seawater or in estuaries where freshwater from rivers and streams mixes with salty ocean water. The salinity level in seawater is fairly constant, at about 35 ppt (35,000 mg/L), while brackish estuaries may have salinity levels between 1 and 10 ppt.

The salinity range of the Conductivity Probe is 0 to 13 ppt. Seawater has a salinity of 35 ppt, so any seawater samples will need to be diluted before making measurements with this sensor. We recommend that you dilute seawater samples (or other samples that initially give readings above 13 ppt) to 1/4 of their original concentration, then multiply their measured salinity reading by 4 to obtain a final salinity value, in ppt. Brackish water in coastal estuaries is often in the range of 0 to 10 ppt, well within the high range of the probe. Note: Vernier also sells a Salinity Sensor (order code SAL-BTA) with a range of 0 to 50 ppt.

Since there is no stored salinity calibration for a Conductivity Probe, perform a two-point calibration using 5-ppt and 10-ppt salinity standards. Write down the displayed intercept and slope calibration values after the calibration is completed. You can immediately use the calibration, save the calibration along with an experiment file if you are using a computer, or load the calibration manually at a later time if you are using a calculator.

You will need to prepare two standard solutions to calibrate for salinity:

Low Standard (5 ppt salinity)

- Add 4.60 g of NaCl to enough distilled water to prepare 1 liter of solution.

High Standard (10 ppt salinity)

- Add 9.20 g of NaCl to enough distilled water to prepare 1 liter of solution.

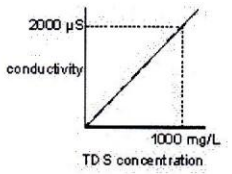
More about Conductivity

Conductivity is an easy and informative water quality test. It is sometimes used as a "watchdog" environmental test—any change in the ionic composition of a stream or lake can quickly be detected using a conductivity probe. Conductivity values will change when ions are introduced to water from salts (e.g., Na^+, Cl^-), acids (H^+), bases (OH^-), hard water (Ca^{2+}, HCO_3^-, CO_3^{2-}), or soluble gases that ionize in

solution (CO_2, NO_2, or SO_2). However, a conductivity probe will not tell you the *specific* ion responsible for the increase or decrease in conductivity. It simply gives a general indication of the level of total dissolved solids (TDS) in the stream or lake. Subsequent tests can then help to determine the specific ion or ions that contributed to the initial conductivity reading (e.g., pH for H^+, a titration for hard water as Ca^{2+}, or a colorimetric test for NO_3^-).

State and local regulations often place upper limits on the level of total dissolved solids in drinking water. These levels vary from state to state, but often must be at a level less than 1100 mg/L TDS. A conductivity probe can give a quick and accurate reading for such a determination.

Since there is a nearly linear relationship between conductivity and concentration of a specific ion or salt, the Conductivity Probe can be used to determine the concentration of an ion. A curve similar to the one shown here can be obtained if you prepare or purchase standard solutions (solutions with known concentrations). Note in this figure the 2:1 ratio between conductivity in µS/cm and TDS concentration in mg/L.



Even though total dissolved solids is often defined in terms of this 2:1 ratio, it should be understood that a TDS reading of 500 mg/L can have a different meaning in a sample that is mostly NaCl than in another sample that is composed primarily of hard water ions such as Ca^{2+} and HCO_3^-. The relationship between conductivity and sodium chloride concentration is *approximately* a 2:1 ratio and is very nearly a direct relationship. Table 1 shows some corresponding values for conductivity (µS/cm), concentration (mg/L as NaCl), and concentration (mg/L TDS).

Conductivity probes can provide students with important clues as to the ionic or molecular nature of compounds. Non-ionizing molecular compounds, such as methanol, will give readings of nearly zero conductivity. Note: Solutions that give a zero conductivity reading will be rare. Even in very pure distilled water, ions will be produced from dissociation of water into H^+ and OH^- ions or carbon dioxide dissolving and producing HCO_3^- ions. Water-soluble ionic compounds will give significant conductivity values, the size of which depends on such factors as ionic radius, charge of ions, and mobility of ions. Ionizing molecular compounds such as weak acids will yield conductivity values that can be used to relate the relative strength of these acids—an aqueous solution of a strong acid such as hydrochloric acid will give a much higher conductivity value than a weak acetic acid solution of equal concentration.

Table 2

Sodium chloride concentration (mg/L)	Total dissolved solids (TDS) (mg/L)	Conductivity (µS/cm)
1.0	1.1	2.2
5.0	5.4	10.8
10	10.7	21.4
20	21.4	42.7
50	52.5	105
100	105	210
150	158	315
200	208	415
500	510	1020
1000	995	1990
1500	1465	2930
2000	1930	3860
5000	4482	8963
10250	9000	18000

Warranty

Vernier warrants this product to be free from defects in materials and workmanship for a period of five years from the date of shipment to the customer. This warranty does not cover damage to the product caused by abuse or improper use.

Light Sensor

Crossbow

MTS

SENSOR, DATA ACQUISITION BOARDS

- Selection of 4 Standard Sensor Boards
- TinyOS Drivers Support Sensor Readings
- Supports MICA, MICA2, MICA2DOT Motes
- Individual Power Control for Each Sensor

Applications

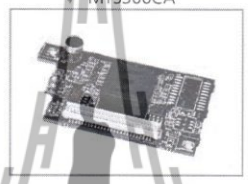
- Vibration and Magnetic Anomaly Detection
- External Sensor Connection
- Localization and Acoustic Tracking
- Robotics
- Wireless Sensor Networking

wireless sensor networks

MTS300

The MTS300CA is a flexible sensor board with a variety of sensing modalities. These modalities include Light, Temperature, Acoustic, and Sounder. The MTS300CA is for use with both MICA and MICA2 Motes.

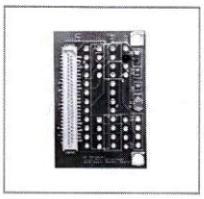
▼ MTS300CA



MTS101

The MTS101CA series sensor board has a precision thermistor, a light sensor/photocell, and general prototyping area. It is for use with the MICA and MICA2 Motes. The prototyping area supports connection to five channels of the Mote's A-D Converter, the I2C digital communications bus, and has 24 unconnected solder points that are used for breadboarding.

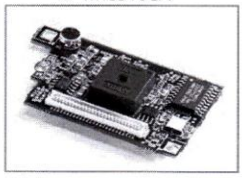
▼ MTS101CA



MTS310

The MTS310CA is a flexible sensor board with a variety of sensing modalities. These modalities include 2-Axis Accelerometer (ADXL202), 2-Axis Magnetometer, Light, Temperature, Acoustic, and Sounder. The MTS310CA is for use with the MICA and MICA2 Motes.

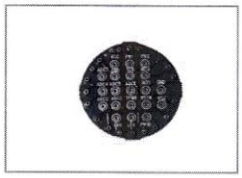
▼ MTS310CA



MDA500

The MDA500CA series prototype and data acquisition board provides a flexible user-interface for connecting external signals to the MICA2DOT Mote. All of the major I/O signals of the MICA2DOT Mote are routed to plated thru holes on the MDA500CA circuit board.

▼ MDA500CA



Ordering Information

Model	Description
MTS101CA	Light, Temperature, Prototype Area Sensor Board for MICA, MICA2 Motes
MTS300CA	Light, Temperature, Acoustic, and Sounder Sensor Board for MICA, MICA2 Motes
MTS310CA	Light, Temperature, Acoustic, Sounder, 2-Axis Accel, and 2-Axis Mag Sensor Board for MICA, MICA2 Motes
MDA500CA	General Purpose Interface for MICA2DOT Motes

Document Part Number: 6020-0047-01 Rev B

crossbow technology, inc ▼ 41 daggett drive ▼ san jose, ca 95134-2109

NOTE: If you have downloaded the PDF schematic of the Rene basic sensor board from UC Berkeley, you will see that the A/D channels appear in reverse order. This is due to a difference in wiring between the original Rene Mote and the MICA/MICA2 family of Motes.

WARNING: Never connect signals that are greater than VCC (3V typical) or less than 0 V to any of the holes that connect to the Mote Processor Radio board. It is okay to connect different voltages to the non-connected holes. However, be careful. If a voltage out of the range of 0 to VCC should reach the Mote Processor Radio Board damage **will** occur.

3 MTS300/MTS310

MTS300CA/MTS310CA and MTS300CB/MTS310CB have the same content in this chapter except for some minor changes.

The MTS300 (Figure 3-1a) and MTS310 (Figure 3-1b) are flexible sensor boards with a variety of sensing modalities. These modalities can be exploited in developing sensor networks for a variety of applications including vehicle detection, low-performance seismic sensing, movement, acoustic ranging, robotics, and other applications. The following section of the User's Manual describes the sensor circuits and general application. Please refer to the schematic diagram at end of section for exact circuit details.

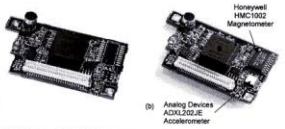


Figure 3-1. *(a) MTS300 and (b) MTS310 with the accelerometer and magnetometer highlighted*

3.1 Microphone

The microphone circuit has two principal uses: First is for acoustic ranging and second is for general acoustic recording and measurement. The basic circuit consists of a pre-amplifier (U1A-1), second-stage amplified with a digital-pot control (U1A, PT2).

This circuit amplifies the low-level microphone output. This output can be fed directly into the analog-digital converter (ADC2) by using the Microphone Output selector circuit (MX1) to connect mic_out signal to ADC2 signal. This configuration is useful for general acoustic recording and measurement. Audio files have been recorded into the logger flash memory of MICAz, MICA2, or MICA Motes for later download and entertainment (or analysis!).

The second stage output (mic_out) is routed thru an active filter (U2) and then into a tone detector (TD1). The LM567 CMOS Tone Detector IC actually turns the analog microphone signal into a digital high or low level output at INT3 when a 4 kHz tone is present. The Sounder circuit on the sensor board can generate this tone.

A novel application of the sounder and tone detector is acoustic ranging. In this application, a Mote pulses the sounder and sends an RF packet via radio at the same time. A second Mote listens for the RF packet and notes the time of arrival by resetting a timer/counter on its processor. It then increments a counter until the tone detector detects the sounder. The counter value is the time-of-flight of the sound wave between the two Motes. The time-of-flight value can be converted into an approximate distance between Motes. Using groups of Motes with Sounders and Microphones, a crude localization and positioning system can be built

NOTE: Motes are designed for power efficiency. Hence all the sensors are disconnected from power on the MTS300 and MTS310 sensor boards unless specifically turned on. See Section 3.6 for more information.

3.2 Sounder

The sounder or "buzzer" is a simple 4 kHz fixed frequency piezoelectric resonator. The drive and frequency control circuitry is built into the sounder. The only signal required to turn the sounder on and off, is Sounder_Power. Sounder_Power is controlled thru the power control switch (P1) and is set by the hardware line PW2.

3.3 Light and Temperature

NOTE: The light and temperature sensor share the same A/D converter channel (ADC1). Only turn one sensor on at a time, or the reading at ADC1 will be corrupted and meaningless.

The MTS300 and MTS310 sensor boards have a light sensor and a thermistor.

The light sensor is a simple CdSe photocell. The maximum sensitivity of the photocell is at the light wavelength of 690 nm. Typical on resistance, while exposed to light, is 2 kΩ. Typical off resistance, while in dark conditions, is 520 kΩ. In order to use the light sensor, digital control signal PW1 must be turned on. The output of the sensor is connected to the analog-digital converter channel 1 (ADC1). When there is light, the nominal circuit output is near VCC or full-scale, and when it is dark the nominal output is near GND or zero. Power is controlled to the light sensor by setting signal INT1.

The thermistor (Panasonic ERT-J1VR103J) on the MTS300 and MTS310 is a surface mount component installed at location RT2. It is configured in a simple voltage divider circuit with a nominal mid-scale reading at 25ºC. The output of the temperature sensor circuit is available at ADC1.

For MTS300CA and MTS310CA, the thermistor power is controlled by setting signal INT2.

For MTS300CB and MTS310CB, the thermistor power is controlled by setting signal PW0.

Table 3-1. *Voltage, Resistance vs. Temperature*

Temperature (ºC)	Resistance (Ohms)	ADC1 Reading (% of VCC)
-40	427,910	2.3%
-20	114,200	8.1%
0	35,670	22%
25	10,000	50%
40	4090	71%
60	2224	82%
70	1520	87%

3.3.1 Conversion to Engineering Units

The Mote's ADC output can be converted to degrees Kelvin using the following approximation over 0-50 ºC:

$1/T(K) = a + b \times \ln(R_{thr}) + c \times [\ln(R_{thr})]^3$

where:

$R_{thr} = R1(ADC\_FS-ADC)/ADC$

a = 0.0013705

b = 0.0002145[illegible]

c = 0.0000007[illegible]

R1 = 10 kΩ

ADC_FS = 1023

ADC = output value from Mote's ADC measurement.

3.4 2-Axis Accelerometer (MTS310 Only)

The accelerometer is a MEMS surface micro-machined 2-axis, ± 2 g device. It features very low current draw (< 1mA) and 10-bit resolution. The sensor can be used for tilt detection, movement, vibration, and/or seismic measurement. Power is controlled to the accelerometer by setting signal PW4, and the analog data is sampled on ADC3 and ADC4. The accelerometer at location U5 is an ADXL202JE and the full datasheet is available at http://www.analog.com. A summary of specification is provided in Table 3-2 below for reference.

Table 3-2. *Summary of ADXL202JE Specifications.*

Channels	X (ADC3), Y (ADC4)
G-range	±2 g (1 g = 9.81 m/s²)
Bandwidth	DC-50 Hz (controlled by C20, C21)
Resolution	2 mG (0.002 G) RMS
Sensitivity	167 mV/G ±17 %
Offset	2.5 V ±0.4 V

NOTE: The ADXL202 sensitivity and offset have a wide initial tolerance. A simple calibration using earth's gravitational field can greatly enhance the accuracy of the ADXL202 sensor. By rotating the sensor into a +1 G and a −1 G position, the offset and sensitivity can be calculated to within 1 %.

3.5 Two-Axis Magnetometer (MTS310 Only)

The magnetometer circuit is a silicon sensor that has a unique bridge resistor coated in a highly sensitive NiFe coating. This NiFe coating causes the bridge resistance of the circuit to change. The bridge is highly sensitive and can measure the Earth's field and other small magnetic fields. A useful application is vehicle detection. Successful test have detected disturbances from automobiles at a radius of 15 feet. The sensor is the Honeywell HMC1002 sensor. A detailed specification sheet is found at http://www.ssec.honeywell.com. The output of each axis (X, Y) is amplified by an instrumentation amplifier U6, U7. The amplified output is available at ADC5 and ADC6. Power is controlled to the magnetometers by setting signal PW5. Each instrumentation amplifier (U6, U7) can be tuned using the digital potentiometer PT1 that is controlled via the I2C bus.

WARNING: The NiFe core of the magnetic sensor is extremely sensitive. However, it is also subject to saturation. Saturation occurs when the sensor is exposed to a large magnetic field. Unfortunately the MTS310 circuit does not have an automatic saturation recovery circuit (set/reset). This limitation prevents the magnetometer from being useful in applications

requiring DC response (for example compassing). There are four pads label S/R (Set/Reset) available on the PCB for adding an external set/reset circuit.

3.6 Turning Sensors On and Off

All of the sensors have a power control circuit. The default condition for the sensor is off. This design helps minimize power draw by the sensor board.

In order to turn sensors on, control signals are issued to the power switches. Table 3-3 below lists the control settings.

Table 3-3. *Control Settings for the Sounder and Sensors*

Sensor/Actuator	Control Signal
Sounder	PW2
Microphone	PW3
Accelerometer	PW4
Magnetometer	PW5
Temperature (RT2)	INT2/PW0[1]
Photocell (R2)	INT1
Temperature(RT2)(MTS300CB/MTS310CB)	PW0

◂ **NOTE:** Only one of the INT1 and INT2/PW0 signals should be activated at a time. See Section 3.3.

[1] For MTS300CA and MTS310CA, the RT2 power is controlled by setting signal INT2. For MTS300CB and MTS310CB, the RT2 power is controlled by setting signal PW0.

3.7 Schematics of the MTS300 and MTS310

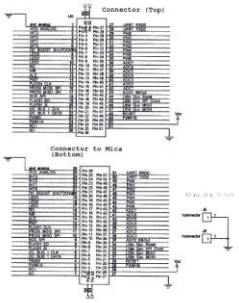


Figure 3-2. *MTS300/310 Schematic of 51-pin connector pin-outs*

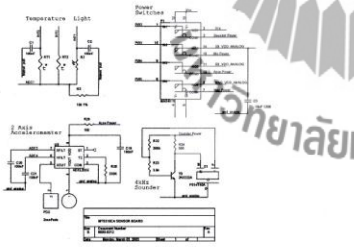


Figure 3-3(a). *MTS310CA Schematics of Accelerometer, Sounder, Temperature and Light Sensors, and Power Switches*

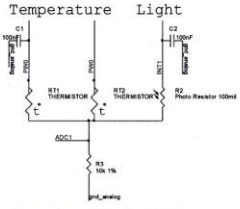


Figure 3-4(b). *Power Controlled Signal of MTS300CB/MTS310CB Temperature and Light Sensors*

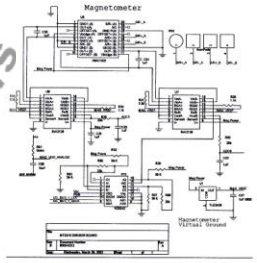


Figure 3-5. *MTS310 Schematic of Magnetometer*

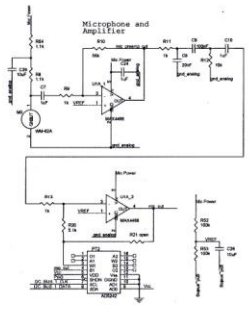


Figure 3-6. *MTS310 Schematic of Microphone and Amplifier*

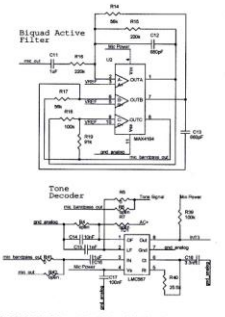


Figure 3-7. *MTS310 Schematic of Biquad Active Filter and Tone Decoder*

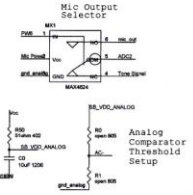


Figure 3-8. *MTS310 Schematic of Mic Output Selector and Analog Comparator Threshold Setup*

ภาคผนวก ค

คู่มือบอร์ด

บอร์ด MICAz

Crossbow

MICAz

WIRELESS MEASUREMENT SYSTEM

- 2.4 GHz IEEE 802.15.4, Tiny Wireless Measurement System
- Designed Specifically for Deeply Embedded Sensor Networks
- 250 kbps, High Data Rate Radio
- Wireless Communications with Every Node as Router Capability
- Expansion Connector for Light, Temperature, RH, Barometric Pressure, Acceleration/Seismic, Acoustic, Magnetic and other Crossbow Sensor Boards

Applications

- Indoor Building Monitoring and Security
- Acoustic, Video, Vibration and Other High Speed Sensor Data
- Large Scale Sensor Networks (1000+ Points)

ZigBee Alliance

MICAz

The MICAz is a 2.4 GHz Mote module used for enabling low-power, wireless sensor networks.

Product features include:

- IEEE 802.15.4 compliant RF transceiver
- 2.4 to 2.483 GHz, a globally compatible ISM band
- Direct sequence spread spectrum radio which is resistant to RF interference and provides inherent data security
- 250 kbps data rate
- Supported by MoteWorks™ wireless sensor network platform for reliable, ad-hoc mesh networking
- Plug and play with Crossbow's sensor boards, data acquisition boards, gateways, and software

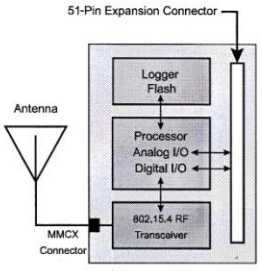
MoteWorks™ enables the development of custom sensor applications and is specifically optimized for low-power, battery-operated networks. MoteWorks is based on the open-source TinyOS operating system and provides reliable, ad-hoc mesh networking, over-the-air-programming capabilities, cross development tools, server middleware for enterprise network integration and client user interface for analysis and a configuration.

Processor & Radio Platform (MPR2400CA)

The MPR2400 is based on the Atmel ATmega128L. The ATmega128L is a low-power microcontroller which runs MoteWorks from its internal flash memory. A single processor board (MPR2400) can be configured to run your sensor application/ processing and the network/radio communications stack simultaneously. The 51-pin expansion connector supports Analog Inputs, Digital I/O, I2C, SPI and UART interfaces. These interfaces make it easy to connect to a wide variety of external peripherals. The MICAz (MPR2400) IEEE 802.15.4 radio offers both high speed (250 kbps) and hardware security (AES-128).

Sensor Boards

Crossbow offers a variety of sensor and data acquisition boards for the MICAz Mote. All of these boards connect to the MICAz via the standard 51-pin expansion connector. Custom sensor and data acquisition boards are also available. Please contact Crossbow for additional information.



MPR2400 Block Diagram

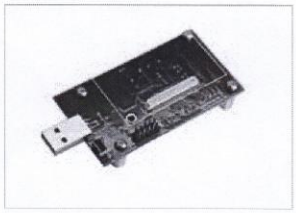
Document Part Number: 6020-0060-04 Rev A

Phone: 408.965.3300 ● Fax: 408.324.4840 ● E-mail: info@xbow.com ● Web: www.xbow.com

Crossbow

Processor/Radio Board	MPR2400CA	Remarks
Processor Performance		
Program Flash Memory	128K bytes	
Measurement (Serial) Flash	512K bytes	> 100,000 Measurements
Configuration EEPROM	4K bytes	
Serial Communications	UART	0-3V transmission levels
Analog to Digital Converter	10 bit ADC	8 channel, 0-3V input
Other Interfaces	Digital I/O,I2C,SPI	
Current Draw	8 mA	Active mode
	< 15 µA	Sleep mode
RF Transceiver		
Frequency band[1]	2400 MHz to 2483.5 MHz	ISM band, programmable in 1 MHz steps
Transmit (TX) data rate	250 kbps	
RF power	-24 dBm to 0 dBm	
Receive Sensitivity	-90 dBm (min), -94 dBm (typ)	
Adjacent channel rejection	47 dB	+ 5 MHz channel spacing
	38 dB	- 5 MHz channel spacing
Outdoor Range	75 m to 100 m	1/2 wave dipole antenna, LOS
Indoor Range	20 m to 30 m	1/2 wave dipole antenna
Current Draw	19.7 mA	Receive mode
	11 mA	TX, -10 dBm
	14 mA	TX, -5 dBm
	17.4 mA	TX, 0 dBm
	20 µA	Idle mode, voltage regulator on
	1 µA	Sleep mode, voltage regulator off
Electromechanical		
Battery	2X AA batteries	Attached pack
External Power	2.7 V - 3.3 V	Molex connector provided
User Interface	3 LEDs	Red, green and yellow
Size (in)	2.25 x 1.25 x 0.25	Excluding battery pack
(mm)	58 x 32 x 7	Excluding battery pack
Weight (oz)	0.7	Excluding batteries
(grams)	18	Excluding batteries
Expansion Connector	51-pin	All major I/O signals

Notes
[1]5 MHz steps for compliance with IEEE 802.15.4/D18-2003.
Specifications subject to change without notice.



MIB520CB Mote Interface Board

Base Stations

A base station allows the aggregation of sensor network data onto a PC or other computer platform. Any MICAz Mote can function as a base station when it is connected to a standard PC interface or gateway board. The MIB510 or MIB520 provides a serial/USB interface for both programming and data communications. Crossbow also offers a stand-alone gateway solution, the MIB600 for TCP/IP-based Ethernet networks.

Ordering Information

Model	Description
MPR2400CA	2.4 GHz MICAz Processor/Radio Board
WSN-START2400CA	2.4 GHz MICAz Starter Kit
WSN-PRO2400CA	2.4 GHz MICAz Professional Kit

Document Part Number: 6020-0060-04 Rev A

Crossbow Technology, Inc. • 4145 North First Street • San Jose, California 95134-2109

บอร์ด MIB520CB

Crossbow

MIB520

USB INTERFACE BOARD

- Base Station for Wireless Sensor Networks
- USB Port Programming for IRIS/MICAz/MICA2 Hardware Platforms
- Supports JTAG code debugging
- USB Bus Power

Applications

- USB Interface
- Testbed Deployments
- In-System Programming

MIB520CB

The MIB520CB provides USB connectivity to the IRIS and MICA family of Motes for communication and in-system programming. Any IRIS/MICAz/MICA2 node can function as a base station when mated to the MIB520CB USB interface board. In addition to data transfer, the MIB520CB also provides a USB programming interface.

The MIB520CB offers two separate ports: one dedicated to in-system Mote programming and a second for data communication over USB. The MIB520CB has an on-board processor that programs Mote Processor Radio Boards. USB Bus power eliminates the need for an external power source.

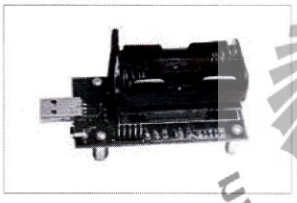
Specifications

USB Interface
- Baud Rate: 57.6 K
- Male to Female USB cable (included with unit)

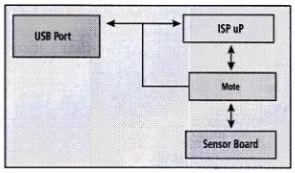
Mote Interface
- Connectors:
  - 51-pin
- Indicators:
  - Mote LED's: Red Green, Yellow

Programming Interface
- Indicators:
  - LEDs - Power Ok (Green), Programming in Progress (Red)
- Switch to reset the programming processor and Mote.

Jtag Interface
- Connector: 10-pin male header POWER
- USB Bus powered



MIB520CB with attached Mote



MIB520CB Block Diagram

Ordering Information

Model	Description
MIB520CB	USB PC Interface Board

Document Part Number: 6020-0091-03 Rev A

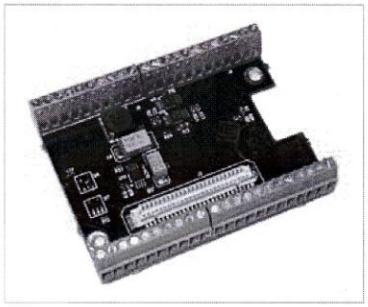
Phone: 408.965.3300 ● Fax: 408.324.4840 ● E-mail: info@xbow.com ● Web: www.xbow.com

บอร์ด MDA300CA

Crossbow

MDA300

DATA ACQUISITION BOARD



- Multi-Function Data Acquisition Board with Temp, Humidity Sensor
- Compatible with MoteView Driver Support
- Up to 11 Channels of 12-bit Analog Input
- Onboard Sensor Excitation and High-Speed Counter
- Convenient Micro-Terminal Screw Connections

Applications

- Environmental Data Collection
- Agricultural and Habitat Monitoring
- Viticulture and Nursery Management
- HVAC Instrumentation and Control
- General Data Collection and Logging

MDA300

Developed at UCLA's Center for Embedded Network Sensing (CENS), the MDA300 is an extremely versatile data acquisition board that also includes an onboard temperature/ humidity sensor. With its multi-function direct user interface, the MDA300 offers a convenient and flexible solution to those sensor modalities commonly found in areas such as environmental and habitat monitoring as well as many other custom sensing applications.

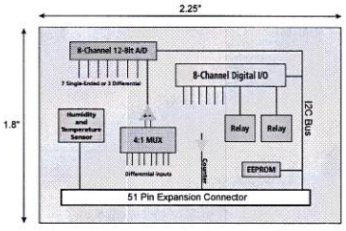
As part of a standard mesh network of Motes, the MDA300's easy access micro-terminals also make it an economical solution for a variety of applications and a key component in the next generation of low-cost wireless weather stations. Data logging and display is supported via Crossbow's MoteView user interface.

Crossbow's MoteView software is designed to be the primary interface between a user and a deployed network of wireless sensors. MoteView provides an intuitive user interface to database management along with sensor data visualization and analysis tools. Sensor data can be logged to a database residing on a host PC, or to a database running autonomously on a Stargate gateway.

Communication and Control Features Including:

- 7 single-ended or 3 differential ADC channels
- 4 precise differential ADC channels
- 6 digital I/O channels with event detection interrupt
- 2.5, 3.3, 5V sensor excitation and low-power mode
- 64K EEPROM for onboard sensor calibration data
- 2 relay channels, one normally open and one normally closed
- 200 Hz counter channel for wind speed, pulse frequencies
- External I2C interface

Drivers for the MDA300 board are included in Crossbow's MoteWorks™ software platform. MoteWorks enables the development of custom sensor applications and is specifically optimized for low-power, battery-operated networks. MoteWorks is based on the open-source TinyOS operating system and provides reliable, ad-hoc mesh networking, over-the-air-programming capabilities, cross development tools, server middleware for enterprise network integration and client user interface for analysis and configuration.



MDA300C Block Diagram

Ordering Information

Model	Description
MDA300CA	Mote Data Acquisition Board with Temperature and Humidity

Document Part Number: 6020-0052-03 Rev A

Phone: 408.965.3300 ● Fax: 408.324.4840 ● E-mail: info@xbow.com ● Web: www.xbow.com

7 MDA300CA

WARNING: The MDA300CA can be damaged by ESD. ESD damage can range from subtle performance degradation to complete device failure.

MDA300CA is designed as a general measurement platform for the MICAz and MICA2 (see Figure 7-1). Its primary applications are a) wireless low-power instrumentation, b) weather measurement systems, c) precision agriculture and irrigation control, d) habitat monitoring, e) soil analysis, and f) remote process control.

Figure 7-1. *Top view of an MDA300CA. This is the side a MICAz or MICA2 Mote would be attached.*

Analog sensors can be attached to different channels based on the expected precision and dynamic range. Digital sensors can be attached to the provided digital or counter channels. Mote samples analog, digital or counter channels and can actuate via digital outputs or relays. The combination of a MICAz (MPR2400CA) or MICA2 (MPR400CB) and a MDA300CA can be used as a low-power wireless data acquisition device or process control machine. Table 7-1 below gives the absolute maximum ratings for various electrical parameters.

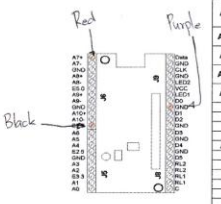
Table 7-1. *The MDA300CAs Absolute Maximum Ratings*

+VDD to GND*.....–0.3V to +5.5V
Digital Lines:
Input voltage range**.....–0.5 V to V_{CC}+ 0.5 V
Continuous output low current.....50 mA
Continuous output high current.....–4 mA
Analog Lines:
Input voltage range.....–0.2 V to V_{CC} + 0.5 V
Counter Line:
Input voltage range .....0 V to 5.5V
Relays:
Maximum Contact Voltage.....100V
Maximum Contact Current.....150mA

*Users are strongly encouraged to stay within the MICAz or MICA2 nominal input voltage of 2.7 to 3.3 VDC
**The input negative-voltage ratings may be exceeded if the input and output current ratings are observed.

7.1 Theory of Operation

This section briefly describes the operation of the pins available on the MDA300CA. A drawing of the pin-outs and their description is shown in Figure 7-2 below.



A0 or A11+	Single-ended analog channel 0 or differential analog channel 11 positive side
A1 or A11-	Single-ended analog channel 1 or differential analog channel 11 negative side
A2 or A12+	Single-ended analog channel 2 or differential analog channel 12 positive side
A3 or A12-	Single-ended analog channel 3 or differential analog channel 12 negative side
A4 or A13+	Single-ended analog channel 4 or differential analog channel 13 positive side
A5 or A13-	Single-ended analog channel 5 or differential analog channel 13 negative side
A6	Single-ended analog channel 6
A7+ A7-	Differential analog channels 7
A8+ A8-	Differential analog channels 8
A9+ A9-	Differential analog channels 9
A10+ A10-	Differential analog channels 10
DATA	I2C Data
CLK	I2C Clock
D0 - D6	Digital Lines D0 to D6
C	Counter Channel
LED1	RED LED
LED2	GREEN LED
E5.0	5.0 V excitation
E3.3	3.3 V excitation
E2.5	2.5 V excitation
Vcc	Vcc of the Mote
RL1	Relay one sides (Normally-Open)
RL2	Relay two sides (Normally-Closed)

Figure 7-2. *Pin configuration and assignments of the MDA300CA*

7.1.1 Single Ended Analog Operation (Channels A0 to A6).

NOTE: These channels are shared with differential channels A11–A13 and both of them cannot be used at the same time.

Signals with dynamic range of 0 to 2.5 V can be plugged to these channels. The least significant bit value is 0.6 mV. The result of ADC can be converted to voltage knowing that

Voltage = 2.5 × ADC_READING / 4096

Resistors need to be added (soldered) to the MDA300CA board to properly scale the voltage levels of external analog sensors so that the maximum voltage is 2.5 VDC. There are two scaling-resistors—R_A and R_B—associated with each ADC channel. These resistors form a simple two-resistor voltage divider. Therefore, choose values for R_A and R_B such that the quantity $R_B/(R_A+R_B)$ multiplied by the maximum output of the sensor is ≤ 2.5 V. The resistors corresponding to a specific ADC channel are listed in Table 7-2 and the area on the board is shown in Figure 7-3 below.

NOTE: The resistors in positions R30 to R36 are 0 Ω resistors and would need to be removed when soldering the corresponding resistor for that channel.

Table 7-2. *Analog Inputs and Resistor Locations for Voltage Scaling*

ADC Channel	R_A	R_B
0	R36	R43
1	R35	R42
2	R34	R41
3	R33	R40
4	R32	R39
5	R31	R38
6	R28	R37

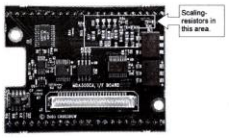


Figure 7-3. *Photo of backside of the MDA300CA.*

7.1.2 Differential Analog Signals (Channels A11 to A13)

Channels A11 to A13 can be used for differential analog signals. Dynamic range and conversion formula are the same as the single ended channels.

7.1.3 Differential Precision Analog Signals (Channels A7 to A10)

Channels A7 to A10 are precision differential channels. They have a sensor front end with gain of 100. Dynamic range of these channels is ±12.5 mV. The offset is cancelled by measurement of the constant offset and writing it to the E2PROM for software cancellation. The result of the ADC can be converted to voltage (in mV) knowing that

Voltage = 12.5 × (ADC_READING / 2048 − 1)

7.1.4 Digital Channels (Channels D0 to D5).

Channels D0–D5 are digital channels that can be used for digital input or output. They can be used for counting external phenomena, triggering based on external events or for actuating external signal.

The result of these channels can be saved to the EEPROM for totalizing sensors to avoid losing count in case of power reset. These channels can be protected against switch bouncing. When they are set as inputs they have internal pull-up resistance so that they can be plugged to switch (close-open) sensors.

7.1.5 Counter Channel

This channel is appropriate for high-speed counting or frequency measurement. It has a Schmitt triggered front-end.

7.1.6 Internal Channels

It has an internal sensor for temperature and humidity. This can be used for monitoring the health of the system. It can also be used for "cold junction compensation" in thermocouple measurement applications. The voltage of the device also can be read using the MICAz's or MICA2's internal monitor to have lifetime information.

7.1.7 Relay Channels

There are two relay channels that can be used for actuation of external phenomena. Both relays are optical solid state for maximum isolation and minimum power consumption. One relay is normally open and the other one is normally closed.

7.1.8 External Sensors Excitation

There are three excitation voltages—5.0 V, 3.3 V, and 2.5 V—available for exciting external sensors. They can be used for turning on active external sensors or they can be used in half bridge or full bridge sensors such as strain gauge, force or pressure measurement.

7.1.9 LEDs

LED signals are brought out for applications that use Motes inside enclosures and want to bring the LEDs to the case.

7.1.10 Power Supply (VCC)

It can be used for an external battery attachment.

บรรณานุกรม

[1] ผศ.ดร. พีระพงษ์ อุฑารสกุล, **427459** ระบบสื่อสารโทรศัพท์เคลื่อนที่ (Mobile Communication System). สาขาวิศวกรรมโทรคมนาคม สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี.

[2] นายณัฐพงศ์ แซ่หู, นางสาวชฎารัตน์ หาดทวายกาญจน์, นายณัฐวุฒิ ภูงามเงิน, โปรแกรมซอฟต์แวร์การกระจายข่าวสารด้วยระบบ SMS. โครงงานวิศวกรรมโทรคมนาคม สาขาวิชาวิศวกรรมโทรคมนาคม สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี, 2553.

[3] นางสาวพิมจันทร์ ชัยชนะ, นางสาวพัชรียา พวงมาลัย, ระบบเฝ้าระวังบ่อเลี้ยงสัตว์น้ำด้วยเซ็นเซอร์ไร้สายผ่านระบบ GSM. โครงงานวิศวกรรมโทรคมนาคม สาขาวิชาวิศวกรรมโทรคมนาคม สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี, 2553.

[4] สัจจะ จรัสรุ่งรวีวร . เริ่มต้น Visual C# 2008 ฉบับสมบูรณ์. กรุงเทพฯ : ไอดีซี อินโฟ ดิสทริบิวเตอร์ เซ็นเตอร์, 2552.

[5] บัญชา ปะสีละเตสัง . พัฒนาแอปพลิเคชั่นด้วย Visual C# 2008. กรุงเทพฯ : ซีเอ็ด ยูเคชั่น, 2552.

[6] วรชัย เชาว์วีระประสิทธิ์ . หัดเขียนโปรแกรมภาษา C# แบบมืออาชีพ. กรุงเทพฯ : เออาร์ อินฟอร์เมชั่น แอนด์ พับลิเคชั่น, 2547.

[7] http://www.vcharkarn.com

[8] http://www.codetoday.net/

[9] http://www.google.co.th/

[10] http://www.vernier.com/probes/

[11] http://www.mindphp.com/forums/

[12] http://thai.aspxcode.net/

[13] http://csnkc.igetweb.com/index.php?page=&mo=10&art=491315&pb=1

[14] http://www.webthaidd.com/webboard/

[15] http://www.thaicreate.com/

[16] http://www.sourcecode.in.th/index.php

[17] http://www.thaicybergames.com/main/forumdisplay.php?f=58

[18] http://www.shadowwares.com/forum/

[19] http://www.denontech.com/Download.html

ประวัติผู้เขียน

นางสาวปิยะดา บำรุงเขตต์ เกิดเมื่อวันที่ 14 ธันวาคม พ.ศ. 2532 ภูมิลำเนาอยู่ที่ ตำบลชานุมาน อำเภอชานุมาน จังหวัดอำนาจเจริญ สำเร็จการศึกษาระดับมัธยมศึกษาตอนปลายจากโรงเรียนชานุมานวิทยาคม อำเภอชานุมาน จังหวัดอำนาจเจริญ เมื่อปี พ.ศ. 2550 ปัจจุบันเป็นนักศึกษาชั้นปีที่ 4 สาขาวิชาวิศวกรรมโทรคมนาคม สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี

นายคมสัน แสงอรุณ เกิดเมื่อวันที่ 23 มิถุนายน พ.ศ. 2532 ภูมิลำเนาอยู่ที่ ตำบลในเมือง อำเภอเมือง จังหวัดหนองคาย สำเร็จการศึกษาระดับมัธยมศึกษาตอนปลายจากโรงเรียนปทุมเทพวิยาคาร อำเภอเมือง จังหวัดหนองคาย เมื่อปี พ.ศ. 2550 ปัจจุบันเป็นนักศึกษาชั้นปีที่ 4 สาขาวิชาวิศวกรรมโทรคมนาคม สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี

นางสาวชลกานต์ มาตรมณีวงศ์ เกิดเมื่อวันที่ 13 มีนาคม พ.ศ. 2533 ภูมิลำเนาอยู่ที่ ตำบลบ้านถ่อน อำเภอท่าบ่อ จังหวัดหนองคาย สำเร็จการศึกษาระดับมัธยมศึกษาตอนปลายจากโรงเรียนปทุมเทพวิทยาคาร อำเภอเมือง จังหวัดหนองคาย เมื่อปี พ.ศ. 2550 ปัจจุบันเป็นนักศึกษาชั้นปีที่ 4 สาขาวิชาวิศวกรรมโทรคมนาคม สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี