



การพัฒนาระบบให้น้ำแบบอัจฉริยะสำหรับแปลงปลูกข้าวโพดหวาน

เอกนรินทร์ ทาไสย

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของความสมบูรณ์ของการศึกษาตามหลักสูตร

ปริญญาวิทยาศาสตรมหาบัณฑิต

สาขาวิชาสหวิทยาการเกษตร

บัณฑิตวิทยาลัย มหาวิทยาลัยแม่โจ้

พ.ศ. 2560

ลิขสิทธิ์ของมหาวิทยาลัยแม่โจ้

การพัฒนาระบบให้น้ำแบบอัจฉริยะสำหรับแปลงปลูกข้าวโพดหวาน

เอกนรินทร์ ทาไสย

วิทยานิพนธ์นี้ได้รับการพิจารณาอนุมัติให้เป็นส่วนหนึ่งของความสมบูรณ์ของการศึกษา

ตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต

สาขาวิชาสหวิทยาการเกษตร

พิจารณาเห็นชอบโดย

อาจารย์ที่ปรึกษา

อาจารย์ที่ปรึกษาหลัก

(ผู้ช่วยศาสตราจารย์ ดร.โชติพงศ์ กาญจนประโชติ)

วันที่ 30 เดือน ตุลาคม พ.ศ. 2560

อาจารย์ที่ปรึกษาร่วม

(ศาสตราจารย์ ดร.สิริวัฒน์ วงษ์สิริ)

วันที่ 30 เดือน ตุลาคม พ.ศ. 2560

อาจารย์ที่ปรึกษาร่วม

(ผู้ช่วยศาสตราจารย์ ดร.ไพศาล กาญจนวงศ์)

วันที่ 30 เดือน ตุลาคม พ.ศ. 2560

ประธานอาจารย์ประจำหลักสูตร

(ศาสตราจารย์ ดร.สิริวัฒน์ วงษ์สิริ)

วันที่ 30 เดือน ตุลาคม พ.ศ. 2560

บัณฑิตวิทยาลัยรับรองแล้ว

(รองศาสตราจารย์ ดร.เกรียงศักดิ์ เม่งอำพัน)

คณบดีบัณฑิตวิทยาลัย

วันที่ 30 เดือน ตุลาคม พ.ศ. 2560

ชื่อเรื่อง	การพัฒนาระบบให้น้ำแบบอัจฉริยะสำหรับแปลงปลูกข้าวโพดหวาน
ชื่อผู้เขียน	นายเอกนรินทร์ ทาไสย
ชื่อปริญญา	วิทยาศาสตรมหาบัณฑิต สาขาวิชาสหวิทยาการเกษตร
อาจารย์ที่ปรึกษาหลัก	ผู้ช่วยศาสตราจารย์ ดร.โชติพงศ์ กาญจนประโชติ

บทคัดย่อ

ระบบการจัดการน้ำเพื่อการผลิตข้าวโพดหวานเป็นปัจจัยหนึ่งที่มีความสำคัญต่อคุณภาพของผลผลิตข้าวโพดหวานในอุตสาหกรรมแปรรูป งานวิจัยนี้จึงทำการศึกษาเกี่ยวกับความเหมาะสมของระบบการให้น้ำในการปลูกข้าวโพดหวาน โดยทำการศึกษาเปรียบเทียบระบบให้น้ำที่แตกต่างกัน 4 ระบบ ใน 3 ระบบแรกมีการใช้อุปกรณ์ช่วยตรวจวัดค่าความชื้นในดิน ซึ่งเป็นระบบให้น้ำแบบแม่นยำ ได้แก่ ระบบให้น้ำแบบน้ำหยด (PDs) ระบบให้น้ำแบบผิวดิน (PSs) ระบบให้น้ำแบบร่องคู (PFs) และระบบสุดท้ายเป็นระบบให้น้ำด้วยวิธีการแบบดั้งเดิม (TFs)

การศึกษาในครั้งนี้มีวัตถุประสงค์เพื่อออกแบบและพัฒนาระบบการให้น้ำแบบอัจฉริยะสำหรับแปลงปลูกข้าวโพดหวาน หลังจากนั้นเปรียบเทียบต้นทุนการเพาะปลูก ปริมาณและคุณภาพของผลผลิตซึ่งเจริญเติบโตภายใต้ระบบให้น้ำที่แตกต่างกัน โดยทำการทดลองกับข้าวโพดหวาน 3 สายพันธุ์ ได้แก่ 1) Hi-Brix53 2) Sugar star และ 3) SCCT-384 จากศึกษาพบว่าปริมาณของผลผลิตสามารถแบ่งออกได้เป็น 2 กลุ่มซึ่งน้ำหนักของผลผลิตไม่มีความแตกต่างกันทางสถิติ ได้แก่ กลุ่มที่ให้น้ำโดยระบบ PFs กับระบบ TFs ($P > 0.05$) และกลุ่มที่ให้น้ำด้วยระบบ PDs กับ PSs ($P > 0.05$) ทั้งนี้ระบบ TFs เป็นระบบที่มีการใช้น้ำมากกว่าระบบอื่น ๆ ส่วนปริมาณการใช้พลังงานไฟฟ้านั้นระบบ PFs เป็นระบบที่ใช้พลังงานไฟฟ้าน้อยที่สุด

Title	THE DEVELOPMENT OF SMART IRRIGATION SYSTEM FOR PRECISION SWEET CORN FARMING
Author	Mr. Eknarin Tasai
Degree	Master of Science in Agricultural Interdisciplinary
Advisor Committee Chairperson	Assistant Professor Dr. Choatpong Kanjanaphachoat

ABSTRACT

The irrigation system for Sweet corn production is an important factor for sweet corn quality and effective in the sweet corn processing industry. This article studies the optimization of the irrigation system for sweet corn cultivation, by using four different irrigation systems, the first, three systems are the precision irrigation systems by sensors technology consisting of the precise drip system (PDs), the precise soil surface system (PSs), the precise furrow system (PFs) the last system is the traditional furrow system (TFs). The aim of this study is to design and develop the irrigation system for sweet corn cultivation, and comparing the cost of sweet corn cultivation as well as yield and quality when grown under different irrigation systems. Three sweet corn cultivars, Hi-Brix 53, Sugar star and SCCT-384, were grown in each irrigation system experiment. The results show that the yield can be classified into 2 groups. Using PFs and TFs produced no difference in the weight of sweet corn ($P>0.05$). In addition, the weight of sweet corns using PDs and PSs produced no difference as well ($P>0.05$). TFs used more water consumption than other irrigation systems. PFs provided a better quality of crop than other irrigation. The power consumption of PFs used less electrical power than other types of irrigation systems.

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปด้วยดีโดยได้รับการอนุเคราะห์จากผู้ช่วยศาสตราจารย์ ดร.โชติพงศ์ กาญจนประโชติ ศาสตราจารย์ ดร.สิริวัฒน์ วงษ์สิริ และผู้ช่วยศาสตราจารย์ ดร.ไพศาล กาญจนวงศ์ ที่ได้ช่วยเหลือให้คำชี้แนะแนวทางในการทำวิจัย ตลอดจนความรู้ในด้านต่าง ๆ

ขอขอบพระคุณคณาจารย์ประจำหลักสูตรสหวิทยาการเกษตรที่ประสิทธิภาพวิชาความรู้ อันหลากหลายศาสตร์หลายแขนงให้ผู้วิจัยได้มีแนวทางในการทำวิจัยในครั้งนี้ ขอขอบคุณเพื่อน นักศึกษาหลักสูตรสหวิทยาการเกษตรทุกท่านที่ได้ช่วยเหลือ ให้คำแนะนำและเก็บข้อมูลให้งานวิจัยนี้ สำเร็จได้ด้วยดี ขอขอบพระคุณอุทยานวิทยาศาสตร์ภาคเหนือและบริษัทชันสวิต จำกัด ผู้ให้การ สนับสนุนงบวิจัยและสถานที่ทำการศึกษาตลอดจนถึงคำแนะนำและความช่วยเหลือในการปฏิบัติงาน ด้วยดีเสมอมา

ขอกราบขอบพระคุณพ่อแม่และผู้ให้ความรัก ความห่วงใย กำลังใจ กำลังทรัพย์และทุกสิ่งทุกอย่างแก่ผู้วิจัยตลอดมา

เอกรินทร์ ทาไสย

ตุลาคม 2560

สารบัญ

	หน้า
บทคัดย่อ	(3)
ABSTRACT	(4)
กิตติกรรมประกาศ	(5)
สารบัญ	(6)
สารบัญตาราง	(8)
สารบัญภาพ	(9)
บทที่ 1 บทนำ	1
ความสำคัญของปัญหา	1
วัตถุประสงค์	2
ขอบเขตการศึกษา	3
ประโยชน์ที่คาดว่าจะได้รับ	3
นิยามศัพท์	3
บทที่ 2 ทฤษฎีและการตรวจเอกสาร	4
ทฤษฎีที่เกี่ยวข้อง	4
การตรวจเอกสาร	19
กรอบความคิดสมมติฐาน	20
บทที่ 3 วิธีการวิจัย	21
การออกแบบฮาร์ดแวร์	21
การออกแบบซอฟต์แวร์	24
สถานที่ทำการวิจัย	25
ประชากรและกลุ่มตัวอย่าง	25
วิธีการดำเนินการ	25
การเก็บข้อมูลและวิเคราะห์ผล	28
บทที่ 4 ผลการวิจัยและวิจารณ์	30
บทที่ 5 สรุปและข้อเสนอแนะ	39
สรุปผลการวิจัย	39
ข้อเสนอแนะ	40

บรรณานุกรม

41

ภาคผนวก

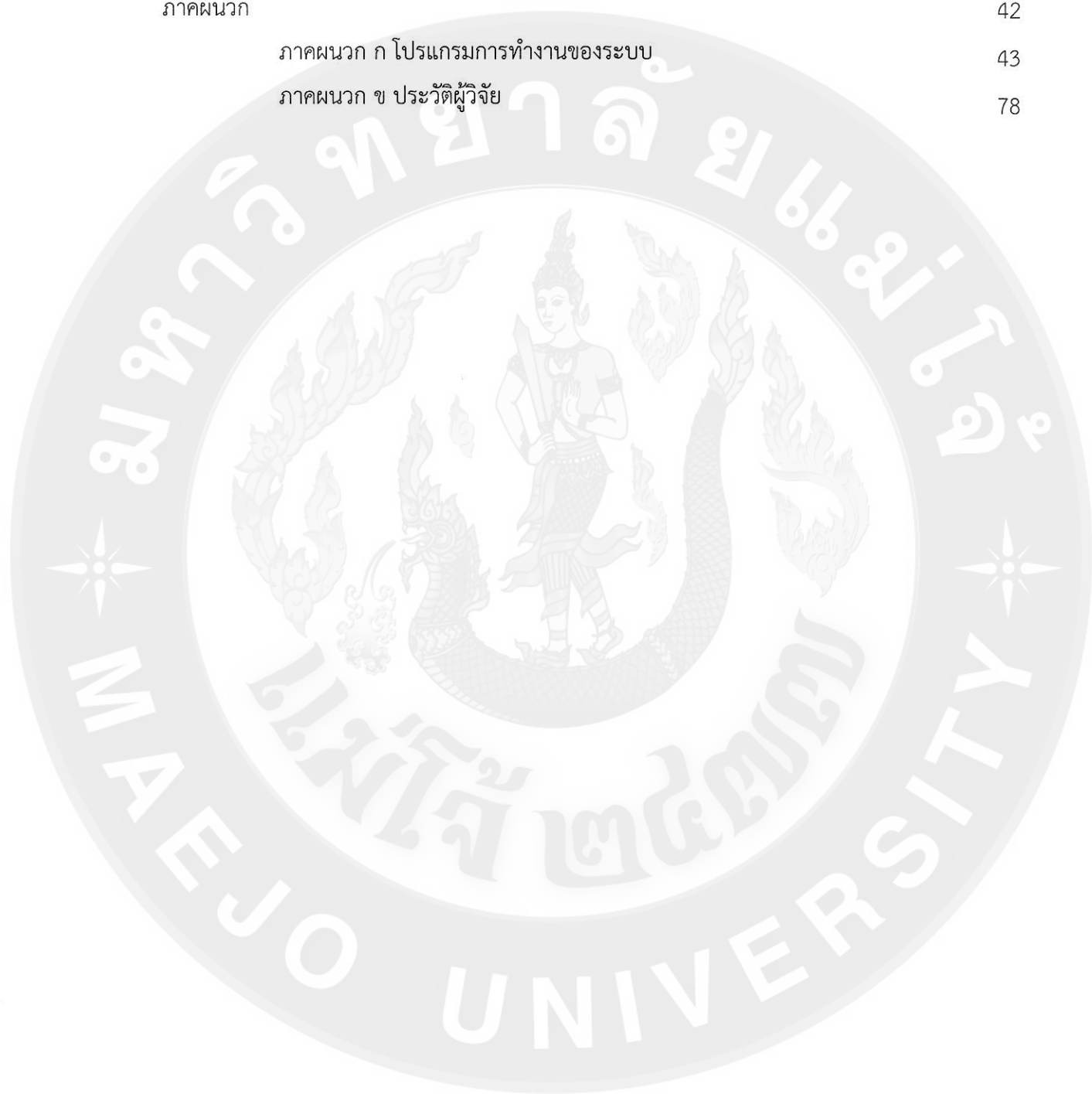
42

ภาคผนวก ก โปรแกรมการทำงานของระบบ

43

ภาคผนวก ข ประวัติผู้วิจัย

78



สารบัญตาราง

ตารางที่		หน้า
1	ข้อมูลผลผลิตจากการเพาะปลูกภายใต้วิธีการให้น้ำที่แตกต่างกัน	32
2	การเปรียบเทียบความแตกต่างของผลผลิตจากการให้น้ำแต่ละวิธีการ	33
3	การเปรียบเทียบคุณภาพของผลผลิตจากการให้น้ำแต่ละวิธีการ	33
4	ร้อยละของคุณภาพผลผลิตจากการให้น้ำแต่ละวิธีการ	34
5	ข้อมูลการใช้พลังงานไฟฟ้าตลอดช่วงการเพาะปลูก	35
6	ข้อมูลการใช้น้ำด้วยวิธีให้น้ำแบบน้ำหยดตลอดช่วงการเพาะปลูก	35
7	ข้อมูลการใช้น้ำด้วยวิธีให้น้ำแบบพ่นฝอยตลอดช่วงการเพาะปลูก	36
8	ข้อมูลการใช้น้ำด้วยวิธีให้น้ำแบบร่องคูตลอดช่วงการเพาะปลูก	36
9	ข้อมูลการใช้น้ำด้วยวิธีให้น้ำแบบตั้งเดิมตลอดช่วงการเพาะปลูก	36
10	ต้นทุนของการปลูกข้าวโพดหวานด้วยวิธีการให้น้ำแบบต่าง ๆ	38

สารบัญภาพ

ภาพที่		หน้า
1	ระดับน้ำในดินและความชื้นของดินที่จุดต่าง ๆ	5
2	ความสัมพันธ์ระหว่างความเป็นประโยชน์ต่อพืชของความชื้นในดินกับระดับความชื้นของดิน	8
3	ความสัมพันธ์ระหว่างอัตราการเจริญเติบโตของพืชกับระดับความชื้นของดิน	8
4	ความสัมพันธ์ระหว่างความชื้นในดินกับการกำหนดการให้น้ำแก่พืช	9
5	เครื่องวัดความชื้นในดินแบบวัดด้วยความต้านทานไฟฟ้า	12
6	เครื่องวัดความชื้นในดินแบบวัดแรงดึงความชื้นโดยเทนซิโอมิเตอร์	13
7	เครื่องวัดความชื้นในดินแบบวัดด้วยนิวตรอน	14
8	ความสัมพันธ์ระหว่างความชื้นในดินกับความต้านทานของสารละลายในแท่งปูนพลาสติกเตอร์	14
9	การจัดวางขาใช้งานของไมโครคอนโทรลเลอร์บอร์ด ATMEGA328	17
10	วงจรขั้วรีเลย์ทำงานด้วยทรานซิสเตอร์	19
11	กรอบความคิดสมมติฐาน	20
12	วงจรตรวจวัดความชื้นในดิน	21
13	แผงวงจรตรวจวัดความชื้นในดิน	21
14	วงจรควบคุมวาล์วน้ำและเครื่องสูบน้ำ	22
15	แผงวงจรควบคุมวาล์วน้ำและเครื่องสูบน้ำ	22
16	วงจรที่เพิ่มเติมเข้ากับบอร์ดไมโครคอนโทรลเลอร์	23
17	แผงวงจรวงจรที่เพิ่มเติมเข้ากับบอร์ดไมโครคอนโทรลเลอร์	23
18	แผนผังการทำงานของระบบควบคุม	24
19	เซนเซอร์วัดความชื้นในดิน	27
20	กราฟมาตรฐานสำหรับหาค่าความชื้นในดินด้วยการเปรียบเทียบแรงดันไฟฟ้า	30
21	ตัวอย่างผลผลิตข้าวโพดหวานพันธุ์ Hi-Brix 53	30
22	ตัวอย่างผลผลิตข้าวโพดหวานพันธุ์ Sugar star	31
23	ตัวอย่างผลผลิตข้าวโพดหวานพันธุ์ SCCT-384	31

บทที่ 1

บทนำ

ความสำคัญของปัญหา

ข้าวโพดหวานเป็นพืชเศรษฐกิจที่สามารถปลูกได้ตลอดทั้งปีและมีการส่งออกผลผลิตไปยังต่างประเทศถึงร้อยละ 75 ของผลผลิตทั้งหมด ซึ่งส่งออกขายในลักษณะแปรรูปบรรจุกระป๋องและแบบฝักสดหรือแช่แข็ง โดยตลาดต่างประเทศส่วนใหญ่ ได้แก่ อังกฤษ เยอรมนี รัสเซีย เกาหลีใต้ ญี่ปุ่น ไต้หวัน เป็นต้น จากข้อมูลการปลูกข้าวโพดหวานโดยสำนักงานเศรษฐกิจการเกษตร ปี 2557 – 2558 พบว่าในภาคเหนือมีพื้นที่การปลูกข้าวโพดหวานเพิ่มมากขึ้นจากเดิมในปี 2557 จำนวน 88,857 ไร่ เป็น 94,844 ไร่ ในปี 2558 แต่ปริมาณผลผลิตกลับลดลงคิดเป็นร้อยละ 1.7 ซึ่งแสดงให้เห็นว่าการปลูกข้าวโพดหวานประสบปัญหาในการเพาะปลูก โดยน้ำเป็นปัจจัยสำคัญอย่างหนึ่งที่มีผลต่อปริมาณผลผลิตและคุณภาพของข้าวโพดหวาน สภาวะภัยแล้งยังส่งผลกระทบต่อต้นทุนการเพาะปลูกที่เพิ่มสูงขึ้น อาทิเช่น ในด้านพลังงานที่ใช้สูบน้ำและจ่ายน้ำและค่าใช้จ่ายในด้านสารเคมีกำจัดแมลงศัตรูพืช การรับมือกับปัญหาภัยแล้งและการควบคุมต้นทุนราคาปัจจัยการผลิตให้คงที่หรือลดน้อยลง จึงเป็นปัญหาสำคัญที่ควรได้รับการแก้ไข

ระบบการทำการเกษตรแบบแม่นยำ (Precision Agricultural System) และระบบเกษตรอัจฉริยะ (Smart Farm) จึงถูกนำมาใช้เพื่อแก้ปัญหาดังกล่าว โดยระบบเกษตรแบบแม่นยำเป็นแนวทางการทำการเกษตรสมัยใหม่ที่เป็นการทำงานเกษตรในลักษณะที่ใช้ปัจจัยการผลิตด้านต่างๆ อย่างพอเพียงในปริมาณที่เหมาะสมสำหรับการเจริญเติบโตของพืชเท่านั้น (Low-External Input Agricultural; LEIA) หรือการใช้ปัจจัยการผลิตให้เกิดประสิทธิภาพสูงสุด โดยทำการตรวจวัดข้อมูลด้านปัจจัยต่างๆ ในแปลงเพาะปลูกและส่งถ่ายข้อมูลด้วยความรวดเร็วสู่ระบบประมวลผลและตัดสินใจเพื่อควบคุมอุปกรณ์ต่าง ๆ ในแปลงเพาะปลูก ดังนั้นอุปกรณ์ตรวจวัดแบบต่างๆ รวมไปถึงโปรแกรมคอมพิวเตอร์สำหรับการส่งถ่ายข้อมูล แสดงผล และควบคุมค่าปัจจัยที่จำเป็นสำหรับการเพาะปลูกในแปลงเพาะปลูกจะช่วยให้การตรวจวัดค่าปัจจัยในการเพาะปลูกและควบคุมและจัดการสภาวะการเพาะปลูกที่เหมาะสม เพื่อลดความเสี่ยงที่จะทำให้ผลผลิตเสียหายและความไม่สม่ำเสมอของผลผลิต ส่งผลให้ต้นทุนการผลิตลดลงเนื่องจากสามารถใช้ปัจจัยการผลิตที่ลดลง แต่ให้ผลผลิตเพิ่มมากขึ้น รักษาสภาพแวดล้อม และสามารถทำการเฝ้าติดตามการเจริญเติบโตของพืช

สิ่งสำคัญอันดับแรกสำหรับการเพาะปลูกภายใต้สภาพปัญหาภัยแล้งคือระบบการชลประทาน (Irrigation system) ที่มีความแม่นยำและมีประสิทธิภาพ โดยเฉพาะพื้นที่แปลงเพาะปลูกพืชขนาดใหญ่

ใหญ่ ซึ่งปริมาณของผลผลิตมีความสัมพันธ์กับระดับปริมาณความชื้นในดินโดยตรง ดังนั้นการควบคุมระดับปริมาณความชื้นในดินให้อยู่ในช่วงความชื้นที่เป็นประโยชน์ต่อพืช (available moisture capacity; AMCA) จึงเป็นงานที่ยุ่ยากสำหรับเกษตรกร ระบบควบคุมการให้น้ำแบบอัตโนมัติ ที่มีความเที่ยงตรงและแม่นยำจึงมีความจำเป็นสำหรับการเพาะปลูกพืชในพื้นที่แปลงขนาดใหญ่และปริมาณแหล่งน้ำที่มีจำกัด กล่าวคือหากสามารถบริหารจัดการการให้น้ำได้อย่างถูกต้อง ปริมาณการให้น้ำต่อครั้งและความแม่นยำในพื้นที่ที่ต้องการให้น้ำก็จะสามารถรักษาค่าปริมาณความชื้นในดินได้เพียงพอและเหมาะสมกับการเจริญเติบโตของพืช ทำให้ได้ผลผลิตที่มีปริมาณและคุณภาพที่สูงขึ้น

การตรวจวัดและควบคุมปัจจัยทางด้านความชื้นที่ต้องการให้แก่ดินให้พอเหมาะกับความต้องการของต้นไม้ สามารถทำได้โดยการประยุกต์ใช้อุปกรณ์อิเล็กทรอนิกส์ที่สามารถหาซื้อได้ในประเทศนำมาพัฒนาเป็นชุดอุปกรณ์ตรวจวัดค่าความชื้นในดิน โดยค่าความชื้นที่ตรวจวัดได้ในแปลงเพาะปลูกจะถูกประมวลผลโดยไมโครคอนโทรลเลอร์และทำการส่งการควบคุมวาล์วน้ำให้เปิดหรือปิดการจ่ายน้ำในพื้นที่นั้นๆอย่างแม่นยำ โดยระบบนี้มีส่วนประกอบหลักๆคือ ชุดอุปกรณ์ตรวจวัดความชื้นในดิน (soil moisture sensors) ไมโครคอนโทรลเลอร์ (microcontroller) และระบบควบคุมการจ่ายน้ำอัตโนมัติ โดยการทำงานของอุปกรณ์ต่างๆจะทำงานประสานกันอย่างเป็นระบบและต่อเนื่อง โดยระบบสามารถแสดงผลค่าปริมาณความชื้นในแปลงเพาะปลูกและแสดงสถานะของการทำงานของวาล์วได้แบบ real-time บนหน้าจอคอมพิวเตอร์ และค่าทั้งหมดจะถูกบันทึกเก็บไว้ในหน่วยบันทึกผล ซึ่งข้อมูลปริมาณความชื้นทั้งหมดในช่วงฤดูการเพาะปลูกสามารถนำมาวิเคราะห์ผลร่วมกับค่าปัจจัยการเพาะปลูกอื่น ๆ ที่อุปกรณ์ตรวจวัดสามารถตรวจวัดได้ เพื่อใช้สำหรับการแจ้งเตือนการระบาดของโรคและแมลง แนะนำแนวการปฏิบัติในกิจกรรมต่างๆในการเพาะปลูก ตลอดจนการคาดการณ์ปริมาณผลผลิตล่วงหน้า โดยข้อมูลในทุกๆฤดูการเพาะปลูกจะถูกเก็บเป็นฐานข้อมูลสำหรับการวางแผนการเพาะปลูกในครั้งต่อไป

ในการวิจัยนี้จึงมุ่งศึกษาวิธีการให้น้ำแก่แปลงปลูกข้าวโพดหวานด้วยวิธีการให้น้ำแบบแม่นยำเพื่อให้สามารถใช้น้ำทางการเกษตรได้อย่างมีประสิทธิภาพ

วัตถุประสงค์

1. พัฒนาอุปกรณ์และระบบตรวจวัดค่าความชื้นในดิน และระบบควบคุมการให้น้ำอัตโนมัติที่เหมาะสมกับการปลูกข้าวโพดหวานอุตสาหกรรม
2. เพื่อหาวิธีการให้น้ำที่เหมาะสมสำหรับการปลูกข้าวโพดหวานอุตสาหกรรมในด้านการเพิ่มประสิทธิภาพการผลิตและความคุ้มค่าทางเศรษฐศาสตร์

ขอบเขตการศึกษา

ขอบเขตของการวิจัยนี้ต้องการศึกษาวิธีการให้น้ำแก่ข้าวโพดหวาน 3 สายพันธุ์ ประกอบด้วย 1) Hi-Brix 53 2) Sugar star และ 3) SCCT-384 โดยวิธีการให้น้ำ 3 วิธี โดยแบ่งเป็นระบบการให้น้ำแบบอัตโนมัติและการให้น้ำแบบดั้งเดิม ในพื้นที่แปลงทดลองปลูกข้าวโพดหวาน บริษัทชั้นสวีท จำกัด อำเภอแม่วาง จังหวัดเชียงใหม่ โดยทำการศึกษา 1 ฤดูกาลเพาะปลูกเป็นระยะเวลา 90 วัน ระหว่างเดือนตุลาคม พ.ศ.2559 ถึงเดือนมกราคม พ.ศ.2560

ประโยชน์ที่คาดว่าจะได้รับ

1. ได้ระบบตรวจวัดค่าความชื้นในดินและระบบควบคุมการให้น้ำอัตโนมัติมีราคาถูก เพื่อส่งเสริมให้เกษตรกรสามารถนำไปใช้งานได้
2. ทราบถึงวิธีการให้น้ำที่เหมาะสมสำหรับการปลูกข้าวโพดหวานอุตสาหกรรมและเผยแพร่แก่เกษตรกรผู้ปลูกข้าวโพดหวานและสามารถนำไปประยุกต์ใช้กับพืชชนิดอื่นได้

นิยามศัพท์

ระบบให้น้ำแบบดั้งเดิม หมายถึง การให้น้ำด้วยวิธีการแบบร่องคูโดยให้น้ำด้วยวิธีการตดสិនใจจากผู้ดูแลและการเพาะปลูก

บทที่ 2

ทฤษฎีและการตรวจเอกสาร

ทฤษฎีที่เกี่ยวข้อง

1. น้ำในดิน

น้ำเป็นส่วนประกอบสำคัญของพืช น้ำในดินนอกจากจะเป็นแหล่งที่รากพืชดูดขึ้นมาหล่อเลี้ยงต้นพืชแล้ว ยังช่วยในการละลายธาตุอาหารในดินให้อยู่ในสภาพที่พืชสามารถนำไปใช้ได้ หากน้ำในดินเข้าไปแทนที่อากาศจนเต็มทุกช่องว่างของดิน เราถือว่าดินนั้นเป็นดินที่อิ่มตัวด้วยน้ำหรือ “อิ่มน้ำ” และน้ำที่อยู่ในช่องว่างนั้นทั้งหมดจะเป็นปริมาณน้ำสูงสุดที่ดินจะเก็บกักเอาไว้ได้ถ้าไม่มีแรงจากภายนอกมากระทำ เราสามารถแบ่งชนิดของน้ำตามความสามารถของดินที่ยึดน้ำไว้ได้ 3 ชนิดตามระดับของน้ำที่ถูกดินดูดยึดไว้ตั้งแต่ชั้นนอกเข้าไปถึงชั้นในที่ติดกับเม็ดดินคือ (ดิเรก, 2545)

1.1 น้ำอิสระ (Gravitational water หรือ Free water)

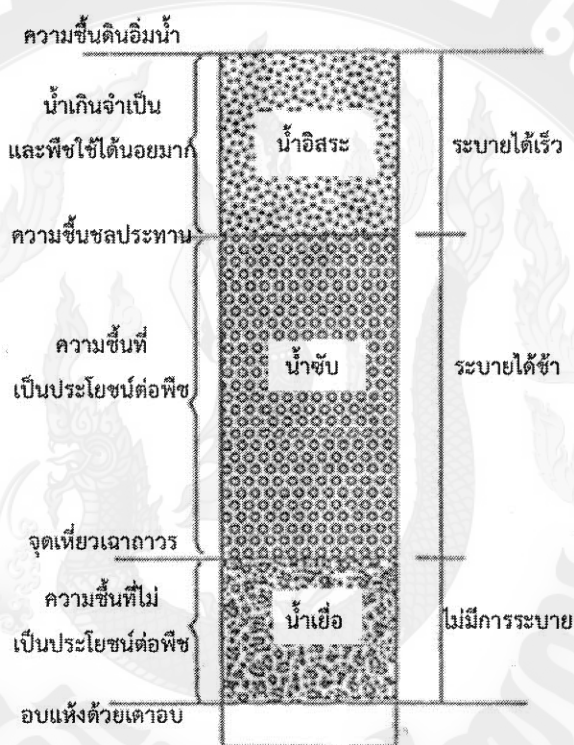
เนื่องจากว่าสสารทุกอย่างที่อยู่บนผิวโลกจะถูกแรงดึงดูดของโลกกระทำอยู่ตลอดเวลา รวมทั้งน้ำที่ขังอยู่ในช่องว่างระหว่างเม็ดดินด้วย ในช่องว่างที่มีขนาดใหญ่แรงยึดเหนี่ยวระหว่างน้ำที่ขังอยู่ตรงกลางช่องว่างกับเม็ดดินจะน้อยกว่าในช่องว่างที่มีขนาดเล็ก ถ้าผลรวมของแรงยึดเหนี่ยวระหว่างน้ำต่อน้ำและน้ำตอดินน้อยกว่าแรงดึงดูดของโลก (หรือน้ำที่ได้รับแรงเหวี่ยงจากอนุภาคดินน้อยมาก) น้ำก็จะไหลสู่ที่ต่ำกว่า น้ำที่ไหลด้วยสาเหตุดังกล่าวนี้เรียกว่า “น้ำอิสระ” น้ำชนิดนี้ถ้าหากอยู่ในดินนานนอกจากจะเป็นอันตรายต่อพืชกล่าวคือจะทำให้พืชขาดอากาศสำหรับหายใจจึงต้องหาทางระบายออกแล้วยังจะเป็นตัวการชะล้างแร่ธาตุอาหารให้สูญเสียไปจากดิน

1.2 น้ำซับ (Capillary water)

เป็นน้ำที่เกิดขึ้นในสภาพที่เมื่อฝนหยุดตกหรือหยุดให้น้ำแก่พืช น้ำถูกระบายสู่ส่วนล่างซึ่งใช้เวลาประมาณ 24-48 ชั่วโมง ในลักษณะนี้ความหนาแน่นของน้ำที่เกาะยึดอนุภาคด้วยดินเต็มเพียงช่องว่างขนาดเล็กด้วยแรงดูดซับที่สูงมากพอที่จะต่อต้านแรงฉุดของแรงดึงดูดของโลกกระทำได้พอดี ซึ่งความชื้นของน้ำซับนี้อนุภาคของดินมีแรงดึงดูดน้ำประมาณ $1/3$ บาร์ และเรียกความชื้นในขณะนี้ว่า ความชื้นชลประทานหรือความจุความชื้นในสนาม (Field capacity) ส่วนน้ำที่อยู่ในช่องว่างขนาดใหญ่จะถูกแรงฉุดของแรงดึงดูดของโลกกระทำให้ไหลซึมออกไป และจะมีอากาศเข้าทดแทนเพื่อพืชสามารถใช้ประโยชน์ในการหายใจ

1.3 น้ำเยื่อ (Hygroscopic water)

เป็นน้ำที่เกาะติดหรือชิดกับอนุภาคของผิวดินและปรากฏในชั้นที่บางมากที่พืชไม่สามารถนำมาใช้ได้ แรงดูดยึดอนุภาคของดินมีค่าประมาณ 31 บาร์ ในจำนวนน้ำในดินทั้ง 3 ชนิดที่กล่าวมานี้ น้ำที่เกี่ยวข้องกับการออกแบบให้น้ำแก่พืชมากที่สุด คือ น้ำซึบตามด้วยน้ำอิสระ ส่วนน้ำเยื่อนั้นพืชไม่สามารถดูดนำไปใช้ได้จึงไม่ค่อยมีความสำคัญนัก อย่างไรก็ตามระดับของน้ำในดินที่ชั้นต่าง ๆ นั้นเขียนแทนได้ในภาพ



ภาพที่ 1 ระดับน้ำในดินและความชื้นของดินที่จุดต่าง ๆ

ที่มา: ดิเรก และคณะ (2545)

2. ระดับความชื้นที่สำคัญของดิน

จากชนิดของน้ำในดินมีหลายชนิด แต่ละชนิดมีคุณสมบัติแตกต่างกัน ดังนั้น การนำน้ำในดินไปใช้ประโยชน์จึงต้องพิจารณาจากระดับความชื้นในดินซึ่งแบ่งความชื้นในดินออกเป็น 5 ชนิดดังนี้

2.1 จุดความชื้นอิมน้ำหรือจุดความชื้นเมื่อดินอิมน้ำ (Water saturation)

เป็นชั้นของน้ำในดินที่เกิดขึ้นเมื่อปริมาณช่องว่างระหว่างเม็ดดินทั้งหมดถูกแทนที่ด้วยน้ำ อาจจะมีอากาศอยู่บ้างในช่องว่างขนาดเล็ก ๆ แต่ก็ปริมาณน้อยมาก ถ้าดินมีความสามารถ

ระบายน้ำได้ดีแล้วปริมาณน้ำที่อยู่ในช่องว่างขนาดใหญ่ก็จะเคลื่อนที่ลงไปยังล่างเนื่องจากแรงดึงดูดของโลกภายในเวลาไม่นานนัก

2.2 ความชื้นชลประทานหรือความจุความชื้นในสนาม (Field capacity)

หมายถึง ความชื้นของดินที่เหลือในดินหลังจากที่น้ำเคลื่อนไหวอิสระได้ถูกระบายออกไปจากช่องว่างขนาดใหญ่หมดแล้วหรือเป็นปริมาณน้ำสูงสุดที่ดินสามารถอุ้มความต้านทางแรงดึงดูดของโลก ในสภาพเช่นนี้ช่องว่างขนาดเล็กจะมีน้ำอยู่เต็มแต่มีอากาศอยู่เต็มช่องว่างขนาดใหญ่ เช่น ปริมาณความชื้นหลังจากที่ฝนตกหนักหรือหยุดให้น้ำแล้ว 2-3 วันเป็นความชื้นชลประทานหรือที่ความจุความชื้นในสนาม โดยทั่วไปแล้วแรงดึงดูดความชื้นที่จุดนี้จะมีค่า $1/3$ บรรยากาศ แต่ค่านี้เปลี่ยนแปลงไปตามลักษณะเนื้อดิน เช่น ดินเหนียวจะมีแรงดึงดูดความชื้นประมาณ $1/10$ บรรยากาศ และดินเหนียวหรือดินค่อนข้างเหนียวมีค่าประมาณ 0.6 บรรยากาศ ระดับความชื้นชลประทานนี้ถือว่าเป็นระดับสูงสุดหรือพิสัยบน (Upper limit) ของความชื้นในดินที่เป็นประโยชน์ต่อพืช (Available moisture content) กล่าวคือพืชสามารถดูดกินความชื้นในระดับนี้ได้ และความชื้นระดับนี้ก็อยู่ในดินได้นานพอให้พืชดูดกิน เพราะระดับความชื้นที่สูงกว่านี้ เช่น น้ำอิสระ รากพืชมีโอกาสดูดน้ำไปใช้ประโยชน์ได้น้อยมากจนไม่ถือว่าเป็นประโยชน์ต่อพืชเนื่องจากระบายไปจากดินในบริเวณที่มีรากพืชได้เร็ว

2.3 จุดเหี่ยวเฉาถาวร (Permanent wilting point)

คือความชื้นในดินเมื่อพืชไม่สามารถดูดน้ำมาใช้ให้เพียงพอสำหรับการคายน้ำและพืชเริ่มมีอาการเหี่ยวเฉาอย่างถาวรเรียกว่าความชื้นที่ “จุดเหี่ยวเฉาถาวร” ระดับความชื้นที่จุดเหี่ยวเฉาถาวรนี้ถือว่าเป็นพิสัยล่างสุด (Lower limit) ของความชื้นในดินที่เป็นประโยชน์ต่อพืช โดยทั่วไปแล้วค่าที่นิยมใช้เป็นค่าประมาณของจุดเหี่ยวเฉาถาวรที่แรงดึงดูดความชื้นเท่ากับ 15 บรรยากาศ อาการเหี่ยวเฉาของพืชอาจเกิดขึ้นได้หลายครั้งเรียกว่าเหี่ยวเฉาชั่วคราวก่อนที่จะถึงจุดที่พืชเหี่ยวเฉาอย่างถาวร เช่น การสูญเสียน้ำโดยการคายออกมาและเมื่ออัตราการที่พืชดูดน้ำจากดินน้อยกว่าที่คายออกทางใบ การที่จะทราบว่ามีการเหี่ยวเฉาอย่างถาวรเมื่อนำพืชที่เหี่ยวเฉานั้นไปไว้ในห้องที่มีอากาศเย็นหรือบรรยากาศรอบ ๆ ต้นพืชอืดตัวด้วยไอน้ำคือมีความชื้นสัมพัทธ์ของอากาศประมาณ 100 เปอร์เซ็นต์เป็นเวลาอย่างน้อย 15 ชั่วโมงแล้วพืชนั้นยังไม่สดขึ้น

2.4 ความชื้นเมื่ออบแห้ง

หมายถึง ปริมาณความชื้นในดินหลังจากถูกอบไว้ที่อุณหภูมิ 105-110 องศาเซลเซียสเป็นเวลาประมาณ 24 ชั่วโมง จนกระทั่งไม่มีน้ำระเหยออกมาจากดิน ซึ่งถือว่าดินในสภาพนี้มีค่าแรง

ถึงความชื้นไม่น้อยกว่า 10,000 บรรยากาศขึ้นไปและนิยมใช้น้ำหนักดินอบแห้งเป็นหลักสำหรับคำนวณหาค่าต่าง ๆ

2.5 ระดับความชื้นในดินที่เป็นประโยชน์ต่อพืช (Available moisture content; AMCO)

คือผลต่างระหว่างความชื้นของดินในขณะหนึ่งกับความชื้นที่จุดเหี่ยวเฉาถาวรของดินนั้น เป็นสิ่งที่แสดงถึงปริมาณความชื้นในดินที่เป็นประโยชน์ต่อพืชของดินในขณะนั้น (สมบุญ, ม.ป.ป.)

$$AMCO = MC - PWP \quad (1)$$

เมื่อ MC คือ ความชื้นของดินในขณะนั้น (Moisture content) (เปอร์เซ็นต์โดยปริมาตร)

PWP คือ ความชื้นที่จุดเหี่ยวถาวร (เปอร์เซ็นต์โดยน้ำหนัก)

ดังนั้น เมื่อความชื้นในดินมากขึ้น ความชื้นในดินที่เป็นประโยชน์ต่อพืชจะมากขึ้นด้วย แต่ในดินชนิดหนึ่งๆ จะดูดยึดความชื้นไว้ได้ไม่เกินความจุความชื้นสนามของดินนั้น เมื่อความชื้นในดินมากขึ้นจนถึงความจุความชื้นในสนามแล้ว ความชื้นในดินที่เป็นประโยชน์ต่อพืชจะมีค่ามากที่สุด เพราะเมื่อความชื้นในดินเกินความจุความชื้นในสนามแล้ว น้ำส่วนเกินนี้จะถูกแรงดึงดูดของโลกดึงดูดลงสู่ดินชั้นล่างเกินระยะรากพืชจะดูดมาใช้ประโยชน์ได้ จึงเรียกผลต่างระหว่างความจุความชื้นสนามกับความชื้นที่จุดเหี่ยวเฉาถาวรของดินชนิดหนึ่งว่า ความจุความชื้นที่เป็นประโยชน์ (Available moisture capacity; AMCA) (สมบุญ, ม.ป.ป.) ค่านี้แสดงถึงปริมาณความชื้นสูงสุดที่เป็นประโยชน์ต่อพืชที่ดินนั้นจะมีได้

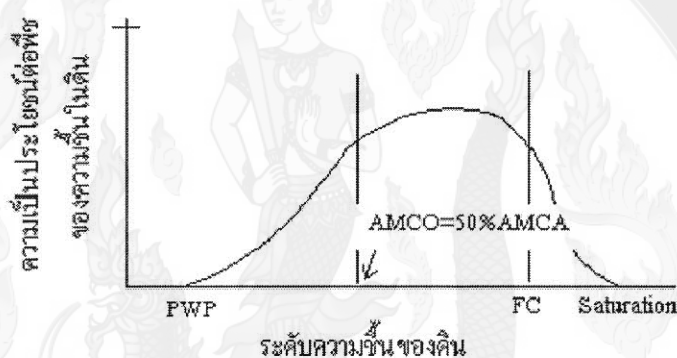
$$AMCO = FC - PWP \quad (2)$$

เมื่อ FC คือ ความจุความชื้นสนาม (เปอร์เซ็นต์โดยน้ำหนัก)

PWP คือ ความชื้นที่จุดเหี่ยวถาวร (เปอร์เซ็นต์โดยน้ำหนัก)

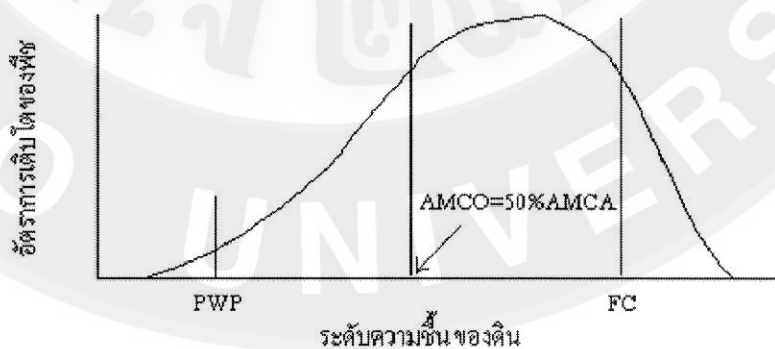
ความชื้นที่เป็นประโยชน์ต่อพืชของดินชนิดหนึ่ง ในขณะเวลาหนึ่งเป็นสิ่งที่แสดงถึงปริมาณที่แท้จริงของความชื้นที่เป็นประโยชน์ต่อพืชที่มีอยู่ในดินในขณะนั้น และมักจะไม่เท่ากับความจุความชื้นที่เป็นประโยชน์ของดินนั้น เพราะในขณะหนึ่ง ๆ ดินอาจดูดยึดความชื้นไว้ได้เต็มที่หรือไม่เต็มที่ก็ได้ และบางครั้งอาจมีความชื้นที่เกินอำนาจดูดยึดของดินอยู่ในดินด้วยความชื้นที่ดินดูดยึดไว้

ในช่วงตั้งแต่จุดเหี่ยวถาวรจนถึงความจุความชื้นสนามของดินนั้นพืชจะดูดไปใช้ประโยชน์ได้ไม่เท่ากัน โดยความชื้นตั้งแต่ขณะที่ดินมีความชื้นเท่ากับความจุความชื้นสนามพืชดูดไปใช้ได้ง่ายมาก แต่เมื่อความชื้นในดินลดลงตามลำดับจากความจุความชื้นในสนาม พืชจะดูดความชื้นที่เหลือไปใช้ประโยชน์ได้ยากมากขึ้นโดยลำดับเพราะพลังงานที่ดินดูดยึดความชื้นไว้ในแต่ละหน่วยความชื้นนั้นจะสูงมากขึ้น เมื่อระดับความชื้นของดินลดลง ด้วยเหตุนี้จึงเรียกความชื้นที่อยู่ในช่วง 75 เปอร์เซ็นต์แรกของความจุความชื้นที่เป็นประโยชน์ต่อพืชของดินตั้งแต่ความจุความชื้นในสนามลงไปว่า “ความชื้นที่พร้อมจะเป็นประโยชน์ต่อพืช” (Readily available moisture) ดังนั้นในการควบคุมความชื้นในดินที่ใช้ปลูกพืชมักจะปล่อยให้ระดับความชื้นที่เป็นประโยชน์ต่อพืชในดินต่ำกว่า 50 เปอร์เซ็นต์ของความจุความชื้นที่เป็นประโยชน์ต่อพืชของดินนั้นดังแสดงในภาพที่ 2 และ 3



ภาพที่ 2 ความสัมพันธ์ระหว่างความชื้นที่เป็นประโยชน์ต่อพืชของความชื้นในดิน
กับระดับความชื้นของดิน

ที่มา: สมบูรณ์ (ม.ป.ป.)



ภาพที่ 3 ความสัมพันธ์ระหว่างอัตราการเจริญเติบโตของพืชกับระดับความชื้นของดิน

ที่มา: สมบูรณ์ (ม.ป.ป.)

ดังนั้นจึงต้องทำการให้น้ำเมื่อความชื้นในดินลดลงใกล้ถึงจุดเหี่ยวเฉาถาวร ซึ่งเรื่องนี้เป็นเรื่องที่สำคัญต่อการตัดสินใจในการกำหนดการให้น้ำแก่พืชมาก ซึ่งโดยทั่ว ๆ ไปจะยอมให้ความชื้นในดินลดลงประมาณ 75 เปอร์เซ็นต์ของความชื้นที่พืชดูดเอาไปใช้ได้ ซึ่งความชื้นในดินที่ยอมรับให้ลดลงก่อนทำการให้น้ำครั้งต่อไปเรียกว่า “ความชื้นที่ยอมให้พืชดูดไปใช้ได้” ส่วนความชื้นที่เหลือในดินหลังจากที่พืชดูดเอาความชื้นที่ยอมให้ดูดไปใช้ได้หมดแล้วคือ “ความชื้นที่จุดวิกฤต” ซึ่งคำอธิบายดังกล่าวนี้สามารถแสดงให้เห็นได้ง่ายดังภาพที่ 4 และจากหลักการที่ยอมให้ความชื้นในดินลดลงประมาณ 75 เปอร์เซ็นต์ของความชื้นที่พืชดูดเอาไปใช้ได้



ภาพที่ 4 ความสัมพันธ์ระหว่างความชื้นในดินกับการกำหนดการให้น้ำแก่พืช

ที่มา: ดิเรก และคณะ (2545)

3. การใช้น้ำของพืช

หมายถึงปริมาณน้ำทั้งหมดที่สูญเสียไปจากพื้นที่เพาะปลูก ซึ่งประกอบด้วยน้ำที่พืชดูดไปจากดินเพื่อนำไปใช้ในการสร้างเซลล์และเนื้อเยื่อแล้วคายสู่บรรยากาศ (Transpiration) และปริมาณน้ำที่ระเหยจากผิวดินรอบ ๆ ต้นพืช (Evaporation) รวมเรียกว่า การคายน้ำ (Evapotranspiration) (วิบูลย์, 2526) ดิเรก และคณะ (2545) รายงานว่ามากกว่า 70 เปอร์เซ็นต์ของน้ำถูกดูดที่ระดับความลึกไม่เกินครึ่งหนึ่งของรากพืช การประเมินการใช้น้ำของพืชสามารถวัดได้จากการปลูกพืชในถังวัดอัตราการใช้น้ำของพืช (Lysimeters) ซึ่งทำจากภาชนะที่ด้านล่างของถังมีท่อสำหรับระบายน้ำที่ไหลผ่านเกินชั้นรากและพืชไม่สามารถดูดไปใช้ได้ ซึ่งปริมาณน้ำที่พืชสามารถประเมินได้จากสมดุลน้ำดังนี้

$$\text{Water consumption} = \text{Evaporation} + \text{Dehydration} \quad (3)$$

ในกรณีที่ป็นต้นไม้ใหญ่ไม่สามารถปลูกในถังวัดการใช้น้ำได้ ก็สามารถทำการสมมูลน้ำจากสภาพแวดล้อมจริงได้เช่นกัน ความลึกของน้ำในดินสามารถหาได้จากการเจาะตัวอย่างดินในถังวัดอัตราการใช้น้ำของพืชที่ระดับความลึกต่าง ๆ แล้วนำไปอบเพื่อหาค่าความชื้นของดินโดยน้ำหนัก ความถ่วงจำเพาะของดิน ความชื้นของดินเชิงปริมาตร และความลึกของน้ำที่แยกออกจากชั้นดิน ดังแสดงเป็นสมการได้ดังนี้

$$P_W = \frac{100 \times W_W}{W_S} \quad (4)$$

เมื่อ P_W คือ เปอร์เซ็นต์ความชื้นโดยเทียบกับน้ำหนักแห้งของดิน

W_W คือ น้ำหนักของน้ำในดิน (กรัม)

W_S คือ น้ำหนักของดินที่อบแห้งด้วยเตาอบ (กรัม)

$$A_S = \frac{W_S}{\rho \times V} \quad (5)$$

เมื่อ A_S คือ ความถ่วงจำเพาะปรากฏของดิน

V คือ ปริมาตรของก้อนดิน ซึ่งเท่ากับปริมาตรของเมล็ดดินรวมกับปริมาตรของน้ำและอากาศ (ลูกบาศก์เซนติเมตร)

ρ คือ น้ำหนักจำเพาะของน้ำ (กรัมต่อลูกบาศก์เซนติเมตร) และสามารถหาความชื้นในดินเป็นเปอร์เซ็นต์โดยปริมาตรจาก

$$P_V = A_S \times P_W \quad (6)$$

เมื่อ P_V คือ ความชื้นของดินเป็นเปอร์เซ็นต์โดยปริมาตรความลึกของน้ำที่แยกออกจากดิน โดยพิจารณาแยกเป็นช่วงความลึก แล้วนำมารวมกันเป็นความลึกของน้ำทั้งหมดในดิน

$$d = \frac{P_V \times D}{100} \quad (7)$$

เมื่อ d คือ ความลึกของน้ำที่อยู่ในดิน (เซนติเมตร)

D คือ ความลึกของดิน (เซนติเมตร)

4. การตรวจวัดความชื้นในดิน

ระดับความชื้นภายในดินมีความสำคัญต่อการจัดการเรื่องน้ำทั้งการกำหนดเวลาและปริมาณน้ำที่จะต้องให้แก่พืช รวมทั้งการให้น้ำแก่พืชเมื่อระดับความชื้นในดินไม่เหมาะสมจะทำให้ผลผลิตต่ำและประสิทธิภาพการใช้น้ำของพืชต่ำไปด้วย ดังนั้นการตรวจวัดความชื้นในดินจึงเป็นสิ่งที่สำคัญที่สุด การให้น้ำแก่พืชจะต้องเริ่มทำเมื่อความชื้นของดินลดลงเหลือน้อยกว่าครึ่งหนึ่งของความจุความชื้นที่เป็นประโยชน์ต่อพืชของดินนั้นแต่ต้องไม่ต่ำกว่าความชื้นที่จุดวิกฤตหรือต่ำกว่า 25 เปอร์เซ็นต์ของความจุความชื้นที่เป็นประโยชน์ต่อพืชของดินนั้น ในการที่จะรู้ว่าการที่ความชื้นในดินลดลงใกล้ถึงจุดวิกฤตหรือยังนั้น จะต้องมีการตรวจวัดความชื้นของดินในเขตรากพืชก่อน เพื่อประเมินว่ามีความชื้นอยู่เท่าใด วิธีการหาความชื้นในดินทำได้ 2 วิธีคือ การวัดความชื้นในดินโดยตรงจากการชั่งและอบแห้ง และการวัดความชื้นในดินโดยทางอ้อมโดยใช้เครื่องมือทางวิทยาศาสตร์ โดยมีรายละเอียดดังต่อไปนี้

4.1 การวัดความชื้นในดินโดยตรงจากการชั่งและอบแห้ง

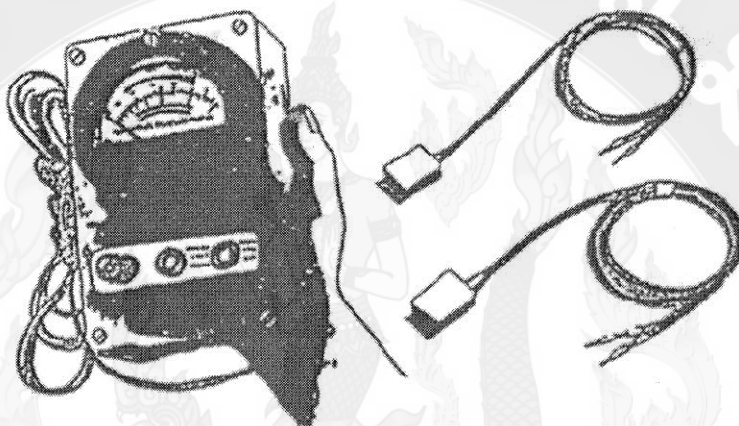
โดยการวิเคราะห์หาความชื้นโดยตรงจากดินตัวอย่างที่เก็บมาจากไร่ที่เราต้องการจะทราบความชื้น วิธีนี้เป็นวิธีที่ดีที่สุดซึ่งทำได้โดยการเก็บตัวอย่างดินบรรจุลงไปในกล่องโลหะแล้วปิดฝาให้มิดชิดแล้วชั่ง ต่อจากนั้นนำดินนั้นเข้าเตาอบที่ 105-110 องศาเซลเซียส เป็นเวลา 24 ชั่วโมง เพื่อให้ไอน้ำออก เสร็จแล้วชั่งอีกครั้งหนึ่งเพื่อหาปริมาณความชื้นที่หายไป ปริมาณความชื้นในดินอาจจะคำนวณออกมาเป็นเปอร์เซ็นต์โดยน้ำหนัก (PW:Percent by weight) ดังได้กล่าวไว้ในสมการที่ 2.4 การบอกเปอร์เซ็นต์ความชื้นในดินนิยมบอกเป็นเปอร์เซ็นต์ของดินที่เตาอบมากกว่าดินที่ขึ้น ทั้งนี้เพราะว่าน้ำหนักของดินที่เตาอบจะคงที่ ส่วนน้ำหนักของดินที่ขึ้นจะเปลี่ยนแปลงไปกับความชื้น

โดยการวิเคราะห์ปริมาณความชื้นของวัสดุพรุน (Porous media) ที่ฝังอยู่ในดินที่ต้องการทราบปริมาณความชื้นวัสดุพรุนนี้โดยปกติใช้พวกยิปซัมบล็อก (Gypsum block) ซึ่งสามารถดูดความชื้นได้ ฝังยิปซัมบล็อกนี้ลงไปในดิน แล้วปล่อยให้ความชื้นในดินแทรกซึมเข้าไปในบล็อกนี้จนกระทั่งความชื้นในบล็อกอยู่ในสภาพสมดุลกับความชื้นในดินแล้วนำยิปซัมบล็อกขึ้นมาชั่ง คำนวณหาน้ำหนักของน้ำที่อยู่ในยิปซัมบล็อกนั้น ซึ่งจะเป็นปริมาณของน้ำที่อยู่ในดินที่ต้องการทราบความชื้นนั่นเอง

4.2 การวัดความชื้นในดินโดยทางอ้อมโดยใช้เครื่องมือทางวิทยาศาสตร์

แบบวัดคุณสมบัติทางไฟฟ้าของวัสดุพรุนหรือแท่งวัดความชื้น (Moisture block) ซึ่งมีความชื้นอยู่ในภาวะสมดุลกับดินบริเวณรอบ ๆ จุดที่ฝังอยู่ คุณสมบัติทางไฟฟ้าที่วัดส่วนมากเป็นความ

ต้านทาน ดังนั้นจึงต้องมีเครื่องมือวัดความต้านทานประกอบด้วยอุปกรณ์ทั้งหมดรวมเรียกว่า เครื่องวัดความชื้นด้วยไฟฟ้า ส่วนตัววัสดุพูนเรียกว่าก่อนความต้านทาน ซึ่งประกอบขึ้นด้วยขั้วไฟฟ้า 2 ขั้วแล้วหุ้มด้วยปูนปลาสเตอร์ โนลอนหรือไฟเบอร์กลาส ดังภาพที่ 5 การใช้เครื่องมือวัดแบบนี้จะมีความเหมาะสมกับการใช้กับดินที่มีความชื้นค่อนข้างต่ำ แต่ไม่ต่ำเกินไปจนความชื้นในดินไม่พอที่จะเกิดความต่อเนื่องทางไฟฟ้าในแท่งวัสดุพูน อย่างไรก็ตามถ้าความชื้นสูงเกินไปหรือสูงใกล้เคียง -0.8 บาร์ ค่าการนำไฟฟ้าจะคงที่ จึงไม่แสดงความแตกต่างให้เห็นจากการอ่าน แม้ว่าความชื้นในดินจะผันแปรแล้วก็ตาม



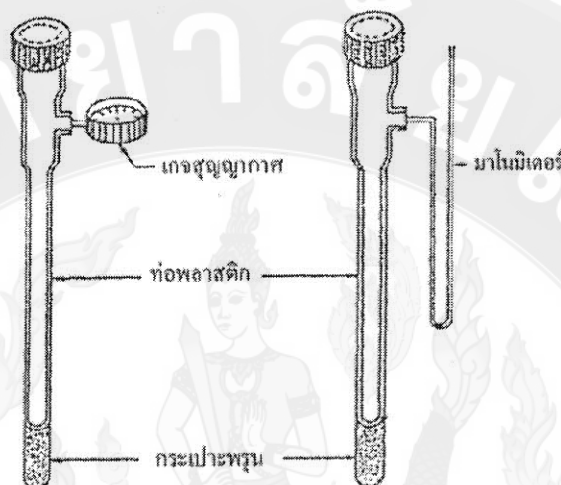
ภาพที่ 5 เครื่องวัดความชื้นในดินแบบวัดด้วยความต้านทานไฟฟ้า

ที่มา: ดิเรก และคณะ (2545)

แบบวัดค่าการนำความร้อน (Heat conductivity) เป็นการวัดความชื้นในดินโดยอาศัยคุณสมบัติบางประการของดินหรือวัสดุพูนที่เกี่ยวข้องอยู่กับความร้อนนั้นก็ได้โดยหลักการที่ว่าถ้าดินที่มีความชื้นน้อยจะนำความร้อนได้ยากกว่าดินที่มีความชื้นมาก ดังนั้นถ้าฝังแผ่นความร้อน (Heating element) ลงไปในดินแล้วผ่านกระแสไฟฟ้าลงไป ถ้าดินมีความชื้นมากความร้อนที่เกิดขึ้นก็จะแพร่กระจายออกไปจากแผ่นความร้อนอย่างรวดเร็ว ดังนั้นความร้อนที่เกิดขึ้นจะไม่สะสมอยู่บริเวณนั้นมากนักและจะไม่ทำให้ค่าของความต้านทานของแผ่นความร้อนสูงขึ้น แต่ถ้าดินนั้นมีความชื้นน้อยหรือเป็นดินที่ค่อนข้างแห้งความร้อนที่เกิดขึ้นก็จะสะสมอยู่เป็นจำนวนมาก ดังนั้นความต้านทานของแผ่นความร้อนจะสูงขึ้น ทั้งนี้เพราะความต้านทานของแผ่นความร้อนจะเป็นปฏิภาคโดยตรงกันข้ามกับความร้อน จากความสัมพันธ์ระหว่างความชื้นในดินกับความสามารถในการนำความร้อนของดินทำให้เราสามารถจะวัดระดับความชื้นในดินได้

แบบวัดแรงดึงความชื้นของดิน (Tensiometer) โดยให้แรงดึงความชื้นของดินนี้อยู่ในสถานะสมดุลกับน้ำที่บรรจุอยู่ในกระเปาะพูน ซึ่งเมื่อรู้ความสัมพันธ์ระหว่างแรงดึงความชื้นของดิน

และจำนวนความชื้นในดินตรงบริเวณจุดที่ติดตั้งเครื่องมือก็จะทราบค่าจำนวนความชื้นในดิน ณ จุดนั้นได้ เครื่องมือชนิดนี้เรียกว่า เครื่องมือแรงดึงดูดความชื้นของดินหรือเทนซิโอมิเตอร์ ดังภาพที่ 6 เทนซิโอมิเตอร์ใช้ในการวัดความชื้นได้กับดินที่มีความชื้นค่อนข้างสูง คือ -0.8 บาร์ขึ้นไป เพราะถ้าหากใช้กับดินที่แห้งกว่านี้ อากาศในดินจะซึมเข้าไปในอุปกรณ์ทางกระเปาะพรุณทำให้อ่านค่าไม่ได้

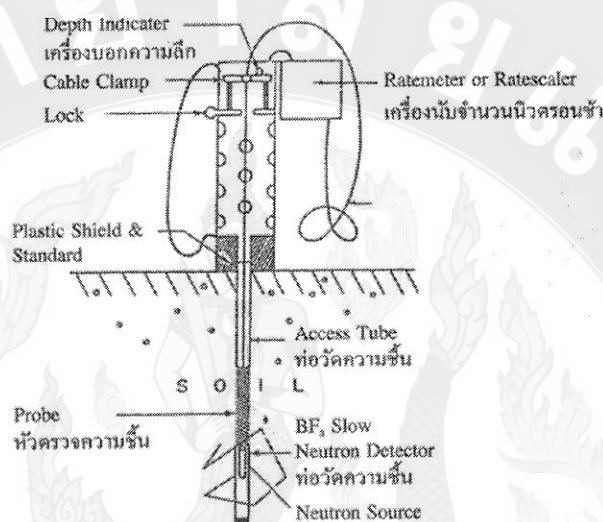


ภาพที่ 6 เครื่องวัดความชื้นในดินแบบวัดแรงดึงดูดความชื้นโดยเทนซิโอมิเตอร์

ที่มา: ดิเรก และคณะ (2545)

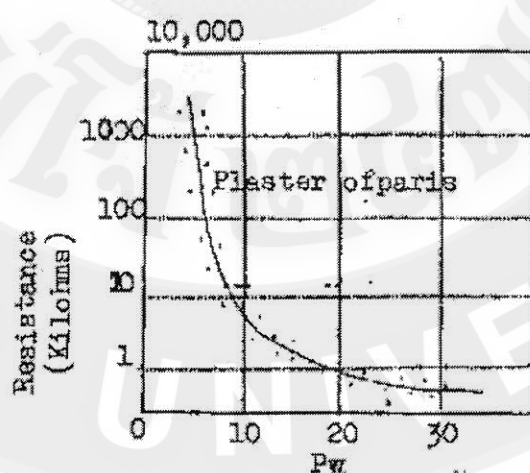
แบบการวัดการกระจายของนิวตรอน (Neutron moisture meter) หลักการทำงานของเครื่องมืออาศัยหลักการที่ว่าเมื่อส่งนิวตรอนเร็ว (Fast neutron) ออกไปจากเครื่อง เมื่อนิวตรอนออกไปกระทบกับนิวเคลียสของไฮโดรเจนอะตอมน้ำ ซึ่งอยู่ในรูปของความชื้นในดินจะเกิดการสูญเสียพลังงานและกลายเป็นนิวตรอนช้า (Slow neutron) จะทำให้ความเร็วของนิวตรอนช้าที่สะท้อนกลับมาช้าลง จำนวนนิวตรอนนิวตรอนช้าที่สะท้อนกลับมานี้สามารถวัดและเทียบเป็นความชื้นในดินได้ ถ้าในดินมีน้ำมากปริมาณของนิวตรอนก็จะสะท้อนกลับมามาก ถ้าในดินมีน้ำน้อยนิวตรอนที่สะท้อนกลับก็จะน้อย เครื่องมือแบบนี้เรียกว่าเครื่องวัดความชื้นด้วยนิวตรอน ดังภาพที่ 7 วิธีการวัดความชื้นในดินแบบนี้มีความสะดวกและแม่นยำ แต่มีราคาแพงเนื่องจากเครื่องวัดความชื้นชนิดนี้ใช้สารกัมมันตภาพรังสี อาจเป็นอันตรายต่อสุขภาพของผู้ใช้ได้หากไม่ระมัดระวัง นอกจากนี้ยังต้องคอยดูแลรักษาอย่างดีด้วย เครื่องมือชนิดนี้จึงไม่เป็นที่นิยมใช้กันทั่วไปเหมือนกับเครื่องวัดแรงดึงดูดความชื้นหรือเครื่องวัดความชื้นด้วยไฟฟ้าที่มีอยู่ส่วนมาก แต่ข้อดีของเครื่องวัดความชื้นชนิดนี้คือ สามารถวัดความชื้นได้รวดเร็ว การวัดแต่ละจุดใช้เวลาประมาณ 1 นาทีเท่านั้น การวัดซ้ำกันหลาย ๆ ครั้งค่าที่ได้ก็ไม่แตกต่างจากวิธีอื่น โครงสร้างของดินในบริเวณรอบ ๆ ก็ไม่เปลี่ยนแปลงหรือถูกทำลายไป จำนวนครั้งของการวัดความชื้นตลอดจนความลึกของดินทำได้สะดวกกว่าวิธีอื่น ซึ่งการวัดความชื้นของดิน

โดยวิธีทางอ้อมนั้น จำเป็นอย่างยิ่งที่จะต้องมีการพามาตรฐาน (Calibration curve) คือเป็นกราฟที่บอกความสัมพันธ์ระหว่างปริมาณความชื้นจริง ๆ ที่มีอยู่ในดินและค่าความชื้นที่อ่านได้จากเครื่องมือที่ใช้วัด อาทิเช่น ถ้าวัดความชื้นโดยวิธีวัดความต้านทานของยิปซัมบล็อกที่ฝังไว้ในดิน เมื่ออ่านค่าความต้านทานของบล็อกนี้ไว้แล้วยังไม่รู้ปริมาณความชื้นจะต้องนำค่าที่อ่านได้มาเทียบกับกราฟความสัมพันธ์เสียก่อน ดังตัวอย่างกราฟมาตรฐานภาพที่ 8



ภาพที่ 7 เครื่องวัดความชื้นในดินแบบวัดด้วยนิวตรอน

ที่มา: ดิเรก และคณะ (2545)



ภาพที่ 8 ความสัมพันธ์ระหว่างความชื้นในดินกับความต้านทานของสารละลายในแท่งปูนพลาสเตอร์

ที่มา: ราชศักดิ์ และ สุขใจ (2546)

5. ระบบให้น้ำแก่พืช

วิธีการในการให้น้ำแก่พืชนั้นสามารถทำได้หลายวิธี ซึ่งการจะเลือกใช้วิธีใดนั้นจะต้องพิจารณาคุณสมบัติของพืช ลักษณะของพื้นที่ วิธีการเพาะปลูก ชนิดของพืชที่ปลูกเป็นต้น อย่างไรก็ตามวิธีการให้น้ำแก่พืชโดยทั่ว ๆ ไปสามารถแบ่งออกได้เป็น 4 แบบใหญ่ ๆ คือ วิธีการให้น้ำทางผิวดิน วิธีการให้น้ำทางใต้ดิน วิธีการให้น้ำแบบฉีดฝอย และวิธีการให้น้ำแบบหยดน้ำ

5.1 วิธีการให้น้ำทางผิวดิน

วิธีการให้น้ำทางผิวดินเป็นวิธีการให้น้ำแก่พืชโดยให้น้ำข้างหรือไหลไปบนผิวดินและซึมลงไปในดินตรงบริเวณที่น้ำข้างหรือไหลผ่าน เพื่อเก็บความชื้นไว้ให้แก่พืช ในปัจจุบันมีวิธีการให้น้ำทางผิวดินหลายรูปแบบตามความเหมาะสม แต่ที่นิยมใช้กันโดยทั่วไปคือการให้น้ำแบบท่วมเป็นอ่าง (Basin irrigation) การให้น้ำแบบท่วมเป็นฝั้น (Border irrigation) และการให้น้ำแบบร่องคู (Furrow irrigation)

5.2 วิธีการให้น้ำทางใต้ดิน

วิธีการให้น้ำทางใต้ดิน เป็นการให้น้ำแก่พืชโดยการยกระดับน้ำใต้ผิวดินให้ถึงเขตรากพืช ซึ่งในระดับน้ำใต้ดินในขณะที่ยให้น้ำนั้นอยู่ระหว่าง 30-60 เซนติเมตรทั้งนี้ขึ้นอยู่กับลักษณะดินและความลึกของเขตรากพืชที่ปลูก น้ำจะไหลไปสู่จุดต่าง ๆ ในเขตรากโดยการดูดซับ การให้น้ำทางใต้ดินทำได้ 2 วิธี คือ แบบคุน้ำเปิดให้น้ำในคูไหลลงไปในดิน และแบบวางระบบท่อส่งน้ำฝังใต้ดิน

5.3 วิธีการให้น้ำแก่พืชแบบหยด

วิธีการให้น้ำแก่พืชแบบหยด เป็นการส่งน้ำลงสู่พื้นดินบริเวณที่มีรากพืชปลูกอยู่อย่างสม่ำเสมอด้วยจำนวนน้ำที่ออกมาทีละน้อย โดยรักษาความชื้นในดินที่พืชสามารถนำไปใช้ได้โดยสะดวกคือที่จุดความชื้นชลประทาน

5.4 วิธีการให้น้ำแบบฉีดฝอยหรือระบบฝนเทียม

วิธีการให้น้ำแบบฉีดฝอยหรือระบบฝนเทียม (Sprinkler irrigation) เป็นการให้น้ำแก่พืชแบบฉีดฝอย โดยน้ำที่จะให้พืชถูกสูบจากแหล่งน้ำผ่านท่อไปยังพื้นที่เพาะปลูกด้วยแรงดันสูงและให้น้ำพ่นเป็นฝอยออกทางหัวฉีดหรือตามรูที่เจาะไว้ตามท่อขึ้นไปในอากาศแล้วปล่อยให้น้ำแพร่กระจายตกลงมาบนที่เพาะปลูก โดยมีรูปทรงการกระจายของเม็ดน้ำสม่ำเสมอและอัตราของน้ำที่ตกลงบนพื้นจะต้องน้อยกว่าอัตราการซึมของน้ำเข้าไปในดิน เนื่องจากการให้น้ำแบบนี้มีลักษณะเช่นเดียวกับฝน ดังนั้นบางคนจึงเรียกการให้น้ำแบบนี้ว่า “การให้น้ำแบบฝนโปรย” วิธีการให้น้ำแก่พืชแบบนี้สามารถ

ส่งน้ำที่ต้องการได้อย่างประหยัด รวดเร็วและสม่ำเสมอ มีประสิทธิภาพสูงแต่ราคาต้นทุนครั้งแรกก็สูงเช่นกัน อย่างไรก็ตามการให้น้ำระบบนี้ก็คุ้มค่าในการลงทุน

5.5 วิธีการให้น้ำแก่พืชแบบประหยัดหรือการให้น้ำแบบน้ำน้อย

วิธีการให้น้ำแก่พืชแบบประหยัด (Micro irrigation) หรือการให้น้ำแบบน้ำน้อย เป็นการให้น้ำครั้งละน้อย ๆ แต่บ่อยครั้งด้วยอัตราการให้น้ำ (Application rate) ที่ต่ำและไม่ครอบคลุมเต็มพื้นที่เขตรากทั้งหมด โดยอาศัยคุณสมบัติของดินช่วยในการแพร่กระจายน้ำออกไปรอบข้างเพื่อให้ปริมาณดินเปียกอยู่ในวงที่จำกัด และเป็นระบบให้น้ำที่ไม่มีการซ้อนทับ (Overlap) ของวงเปียกจากหัวจ่ายน้ำแบบต่าง ๆ เช่น มินิสปริงเกลอร์ (Minisprinkler) ไมโครสปริงเกลอร์ (Microsprinkler) ไมโครเจ็ต (Micro jet) ไมโครสเปรย์ (Micro spray) มิสสเปรย์ (Mist spray) และน้ำหยด วิธีการให้น้ำแบบประหยัดแก่พืชกำลังเป็นที่นิยมทั่วไปและเป็นวิธีการให้น้ำที่กำลังมีบทบาทมากขึ้นในอนาคต เพราะสามารถนำไปประยุกต์ใช้ในการให้น้ำแก่พืชโดยเฉพาะไม้ผลได้หลายชนิดจึงช่วยให้ประหยัดน้ำได้เป็นอย่างดี เช่น ลำไย มะม่วง ทูเรียน เงาะ มะนาว ส้มโอ ลองกอง รวมทั้งพืชผักและพืชไร่บางชนิด เนื่องจากสามารถให้น้ำได้ตั้งแต่ต้นไม้ผลมีขนาดเล็ก ๆ ไปจนถึงต้นเต็มที่ โดยการขยายรัศมีวงเปียก (Wetted radius) ให้รับกับขนาดของต้นไม้ เช่น อาจจะใช้หัวจ่ายน้ำแบบไมโครเจ็ตก่อนขณะที่ต้นไม้ผลยังเล็ก เมื่อต้นโตขึ้นแล้วจึงเปลี่ยนมาใช้หัวน้ำแบบมินิสปริงเกลอร์โดยการเปลี่ยนชิ้นส่วนของหัวจ่ายน้ำมินิสปริงเกลอร์บางชิ้นเท่านั้น แต่ยังใช้เสื้อ (Body) ของสปริงเกลอร์เก่าอยู่หรือเปลี่ยนเฉพาะหัวฉีดฝอย

6. ระบบควบคุมการให้น้ำ

ในการควบคุมการให้น้ำด้วยวิธีการตรวจวัดความชื้นในดิน ประกอบด้วยความรู้พื้นฐานเกี่ยวกับไมโครคอนโทรลเลอร์ การแปลงผันสัญญาณอนาล็อกเป็นดิจิตอล และการควบคุมอุปกรณ์ภายนอก

6.1 ไมโครคอนโทรลเลอร์

ไมโครคอนโทรลเลอร์ (Microcontroller) เป็นชื่อของอุปกรณ์ทางอิเล็กทรอนิกส์ชนิดหนึ่งที่มีความสามารถในการใช้เป็นอุปกรณ์ควบคุมที่สามารถโปรแกรมการทำงานได้ (ชัยวัฒน์ และ วรพจน์, 2543) โดยภายในของไมโครคอนโทรลเลอร์จะเป็นอุปกรณ์ประเภทสารกึ่งตัวนำ (Semiconductor) ที่รวบรวมเอาฟังก์ชันการทำงานต่าง ๆ ไว้ในตัวเองโดยมีโครงสร้างใกล้เคียงกับคอมพิวเตอร์ คือ ภายในประกอบด้วยหน่วยประมวลผลกลางหรือซีพียู (Central Processing Unit, CPU) หน่วยคำนวณทางคณิตศาสตร์และลอจิก (Arithmetic Logic Unit, ALU) หน่วยความจำ (Memory unit) วงจรเชื่อมต่อสัญญาณนาฬิกา (Timer) และพอร์ตสัญญาณเข้าและออก (Input

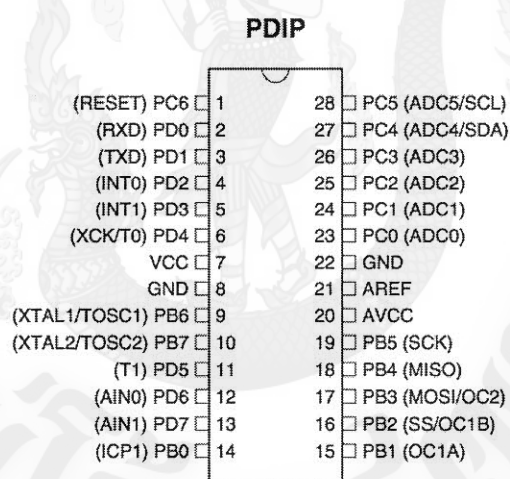
and output port) เพื่อเชื่อมต่อกับอุปกรณ์ภายนอก ซึ่งในปัจจุบันมีการใช้งานไมโครคอนโทรลเลอร์กันอย่างแพร่หลาย เนื่องจากใช้งานง่ายและสามารถประยุกต์ใช้งานได้หลากหลาย โดยทั่วไปแล้วไมโครคอนโทรลเลอร์จะประกอบด้วย

6.1.1 หน่วยประมวลผลกลาง จะเป็นส่วนที่เป็นตัวตัดสินใจต่าง ๆ ซึ่งจะทำงานตามโปรแกรมที่เขียนขึ้นและบรรจุเข้าไปในตัวไมโครคอนโทรลเลอร์

6.1.2 หน่วยความจำ (Memory) เป็นตัวที่ใช้เก็บข้อมูลต่าง ๆ ที่ต้องใช้ในไมโครคอนโทรลเลอร์ ซึ่งประกอบด้วย RAM (Random Access Memory) EPROM (Eraseable Programmable Read Only Memory)

6.1.3 พอร์ตสัญญาณเข้าและออก เป็นส่วนที่ใช้ในการติดต่อกับอุปกรณ์ภายนอกซึ่งประกอบด้วย พอร์ตอนุกรม (Serial port) พอร์ตขนาน (Parallel port) ตัวตั้งเวลา (Timer) ตัวควบคุมการอินเทอร์รัปต์ (Interrupt controller)

โดยลักษณะการจัดวางขาของไมโครคอนโทรลเลอร์ แสดงดังภาพที่ 9



ภาพที่ 9 การจัดวางขาใช้งานของไมโครคอนโทรลเลอร์บอร์ด ATMEGA328

ที่มา: www.atmel.com (2017)

6.2 การแปลงผันสัญญาณอนาลอกเป็นดิจิตอล

ปริมาณทางกายภาพต่าง ๆ เช่น อุณหภูมิ ความชื้นในอากาศและในดิน เป็นต้น จัดเป็นสัญญาณอนาลอก (Analog signal) ซึ่งเป็นสัญญาณที่มีความต่อเนื่องและมีการเปลี่ยนแปลงค่าตลอด แต่เนื่องจากในไมโครคอนโทรลเลอร์จะมีการทำงานและประมวลผลเป็นแบบรหัสหรือสัญญาณดิจิตอล (Digital signal) ดังนั้นการจะเชื่อมต่อไมโครคอนโทรลเลอร์เข้ากันได้กับสัญญาณอนาลอกเพื่อให้สามารถอ่านหรือประมวลผลสัญญาณอนาลอกได้นั้น จำเป็นต้องมีการแปลงสัญญาณ

อนาลอกให้เป็นสัญญาณดิจิทัลก่อน การแปลงผันสัญญาณอนาลอกไปเป็นสัญญาณดิจิทัลมีความสำคัญอย่างมากในระบบหรือกระบวนการวัดของเครื่องมือวัดแบบดิจิทัลที่รวมไปถึงการประมวลผลสัญญาณดิจิทัลของสัญญาณอนาลอกเอาต์พุตจากตัวผสมสัญญาณ (Signal conditioner) โดยระบบอนาลอก (Analog system) จะเชื่อมต่อกับระบบดิจิทัล (Digital system) โดยผ่านการเชื่อมต่อสัญญาณอนาลอกไปเป็นสัญญาณดิจิทัล โดยสัญญาณอนาลอกที่เปลี่ยนอยู่ในรูปของสัญญาณหรือรหัสดิจิทัลนั้น สามารถทำได้โดยวงจรอิเล็กทรอนิกส์ที่เรียกว่า ตัวแปลงผันสัญญาณอนาลอกเป็นดิจิทัล (Analog to digital converter, ADC หรือ A/D) ในการเชื่อมต่อกับพอร์ตขนานของไมโครคอนโทรลเลอร์ การคำนวณหาอัตราส่วนการเปลี่ยนแปลงของสัญญาณอนาลอกอินพุตที่ทำให้สัญญาณดิจิทัลเอาต์พุตเปลี่ยนแปลงไปหนึ่งบิตหรือที่เรียกว่า ค่าความละเอียดของวงจร ADC สามารถคำนวณได้จาก (พานิช และ โกศล, 2550)

$$R = \frac{FSV}{2^n - 1} \quad (8)$$

เมื่อ R คือ ค่าความละเอียดในการวัดของ ADC

FSV คือ ค่าแรงดันอนาลอกอินพุตเต็มสเกล (full-scale input voltage)

n คือ จำนวนบิตสูงสุดของสัญญาณดิจิทัลเอาต์พุต

ส่วนการเปลี่ยนแปลงค่าสัญญาณดิจิทัลที่อ่านได้จากเครื่องคอมพิวเตอร์เป็นค่าสัญญาณแรงดันอนาลอกอินพุต V_{in} สามารถคำนวณได้ จากสมการนี้ (พานิช และ โกศล, 2550)

$$V_{in} = D_o \times \frac{FSV}{2^n - 1} \quad (9)$$

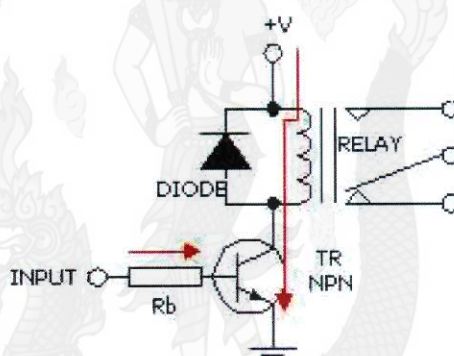
เมื่อ V_{in} คือ ค่าสัญญาณแรงดันอนาลอกอินพุต

D_o คือ ค่าสัญญาณดิจิทัลเอาต์พุต

6.3 การควบคุมอุปกรณ์ภายนอก

รีเลย์เป็นอุปกรณ์แม่เหล็กไฟฟ้าชนิดหนึ่งที่ใช้ในการควบคุมการทำงานของอุปกรณ์กำลังโดยใช้หน้าสัมผัส (Contact) เป็นตัวเปิดหรือปิดให้อุปกรณ์กำลังทำงานโดยหน้าสัมผัสของรีเลย์สามารถทนกระแสและแรงดันได้สูง ซึ่งหน้าสัมผัสของรีเลย์มีอยู่ 2 ชนิดคือ หน้าสัมผัสปกติปิด (Normally closed: NC) และหน้าสัมผัสปกติเปิด (Normally open: NO) หลักการที่จะทำให้

หน้าสัมผัสของรีเลย์เปิดหรือปิดจะใช้อำนาจแม่เหล็กเปลี่ยนแปลงหน้าสัมผัสของรีเลย์ที่เป็นโลหะให้มีการเปลี่ยนแปลงตำแหน่งตามปกติ กล่าวคือเมื่อขดลวดของรีเลย์ได้รับแรงดันจะมีกระแสไหลผ่านขดลวด ซึ่งขดลวดจะสร้างสนามแม่เหล็กขึ้นทำให้สามารถเปลี่ยนตำแหน่งของหน้าสัมผัส โดยหน้าสัมผัสปกติปิดจะเปลี่ยนตำแหน่งเป็นเปิดวงจร และหน้าสัมผัสแบบปกติเปิดจะเปลี่ยนตำแหน่งเป็นปิดวงจร การทำให้รีเลย์ทำงานแบบง่าย คือ การใช้ทรานซิสเตอร์เป็นสวิตช์ปิดหรือเปิดวงจรให้รีเลย์ทำงาน แสดงดังภาพที่ 10 เมื่อต้องการให้รีเลย์ทำงานจะต้องให้ลอจิก “1” ที่ขาเบสของทรานซิสเตอร์ ซึ่งจะทำให้ทรานซิสเตอร์นำกระแส ทำให้มีกระแสไหลผ่านขดลวดรีเลย์ทำให้เกิดสนามแม่เหล็กจึงทำให้หน้าสัมผัสเปลี่ยนตำแหน่ง สำหรับไดโอดเบี่ยงเบนเส้นที่ต่อขนานกับขดลวดเพื่อป้องกันไม่ให้แรงดันไฟฟ้าต้านกลับจากขดลวดเป็นอันตรายต่อทรานซิสเตอร์ (พานิช และ โกศล, 2550)



ภาพที่ 10 วงจรขับรีเลย์ทำงานด้วยทรานซิสเตอร์

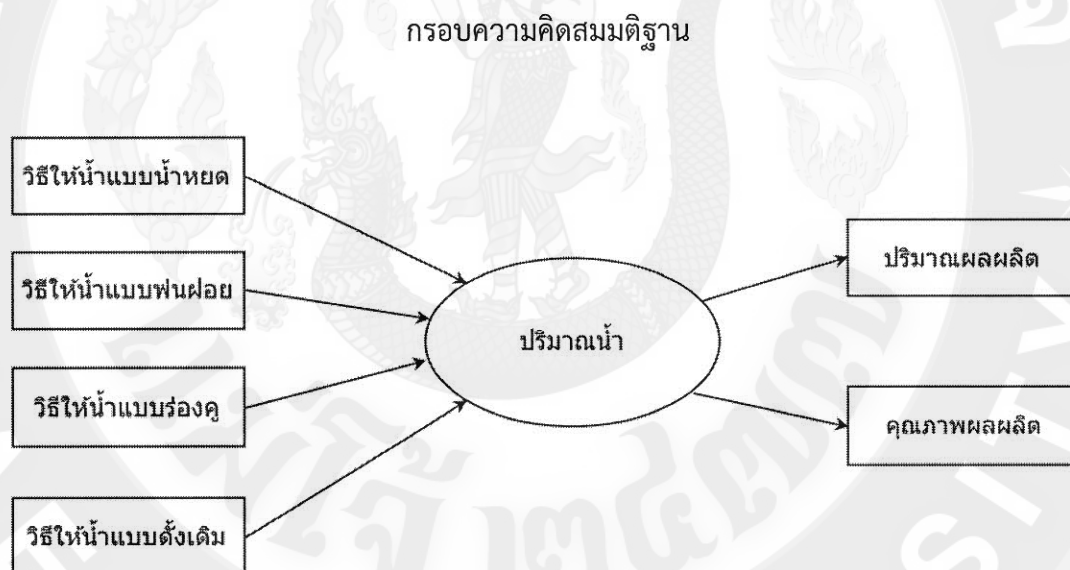
การตรวจเอกสาร

Luthra et al. (1997) ได้ออกแบบและพัฒนาระบบการให้น้ำพืชแบบอัตโนมัติ โดยการวัดค่าแรงยึดเหนี่ยวของน้ำในดินผ่านเซนเซอร์แบบมานอมิเตอร์ โดยระบบอัตโนมัติดังกล่าวจะควบคุมการให้น้ำแก่พืชตามความต้องการน้ำของพืช โดยใช้ข้อมูลจากค่าความชื้นของดินที่วัดได้จากตัวเซ็นเซอร์ (Soil moisture sensor) จากนั้นจะทำนายปริมาณการใช้น้ำและระยะเวลาการให้น้ำจากแบบจำลองปริมาณการใช้น้ำของพืช (Evapotranspiration model) และส่งสัญญาณไปควบคุมวาล์วเปิด-ปิดน้ำ

Lee and Shin (1998) ได้ทำการพัฒนาระบบการจัดการให้น้ำที่จุดเหมาะสมของโรงเรือนเพาะปลูก (Greenhouse) สำหรับมะเขือเทศ โดยใช้ข้อมูลจากสภาวะแวดล้อมของการเพาะปลูกภายในโรงเรือน โดยระบบดังกล่าวได้สร้างขึ้นโดยการผสมผสานกันระหว่างการแสดงผลขนาดลำต้น (Stem diameter) และแบบจำลองการคายน้ำ (Transpiration model) เข้าไปในระบบคอมพิวเตอร์

Chandan and Pramitee (2015) ได้ทำการสร้างระบบให้น้ำแบบอัจฉริยะราคาถูกโดยออกแบบอุปกรณ์ตรวจวัดความชื้นในดินโดยใช้หลักการความต้านทานทางไฟฟ้าของดินมาเป็นตัวกำหนดการเปิด-ปิดให้กับวงจรเปรียบเทียบแรงดันด้วยออปแอมป์ จากนั้นจึงส่งแรงดันเอาต์พุตไปประมวลผลผ่านไมโครคอนโทรเลอร์ด้วยแพลตฟอร์ม Arduino ทำการตัดสินใจในการให้น้ำและแสดงผลสถานะการทำงานของระบบด้วยข้อความผ่านข้อความสั้นสู่โทรศัพท์เคลื่อนที่และอีเมล

งานวิจัยต่าง ๆ ที่ได้กล่าวข้างต้นเป็นการสร้างระบบให้มีความเหมาะสมกับวิธีการให้น้ำแบบใดแบบหนึ่ง ซึ่งยังไม่มีการศึกษาในเรื่องของลักษณะวิธีการให้น้ำที่แตกต่างกันด้วยระบบเดียวกันเพื่อศึกษาทางด้านผลผลิต ดังนั้นในการศึกษาทดลองในครั้งนี้จะทดลองใช้ระบบการตรวจวัดปริมาณความชื้นในดินด้วยระบบที่เหมือนกันในการทดลองให้น้ำแก่แปลงปลูกข้าวโพดหวานด้วยวิธีการที่แตกต่างกัน เพื่อหาวิธีการที่มีความเหมาะสมในการให้น้ำแก่แปลงปลูกโดยคำนึงถึงปริมาณผลผลิตคุณภาพผลผลิต และทรัพยากรที่ใช้ในการผลิต



ภาพที่ 11 กรอบความคิดสมมติฐาน

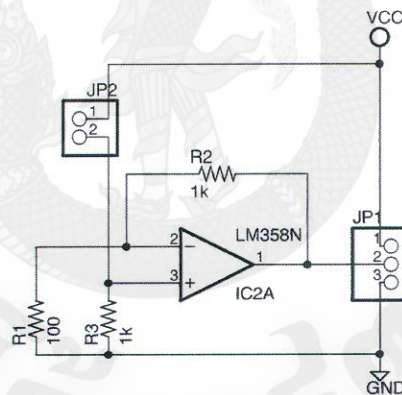
บทที่ 3 วิธีการวิจัย

การออกแบบฮาร์ดแวร์

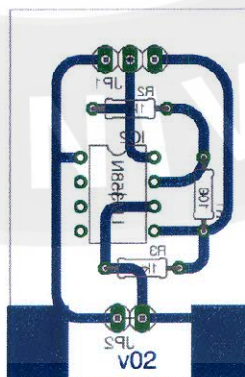
ทำการออกแบบวงจร 3 วงจรได้แก่ วงจรตรวจวัดความชื้นในดิน วงจรควบคุมวาล์วน้ำและเครื่องสูบน้ำ และวงจรควบคุมระบบ ดังนี้

1. วงจรตรวจวัดความชื้นในดิน

วงจรตรวจวัดความชื้นในดินที่ใช้ในการวิจัยนี้จะอาศัยหลักการนำไฟฟ้าที่เกิดขึ้นในดิน อินพุตของวงจรจะอาศัยแรงดันไฟฟ้าจ่ายไบแอสให้แก่วงจรขยายความแตกต่าง (Difference amplifier) ซึ่งต่อวงจรอยู่ในลักษณะขยายแบบไม่กลับเฟส (Non-inverting amplifier) และมีการป้อนกลับแบบลบ ดังแสดงในภาพที่ 12 และ 13



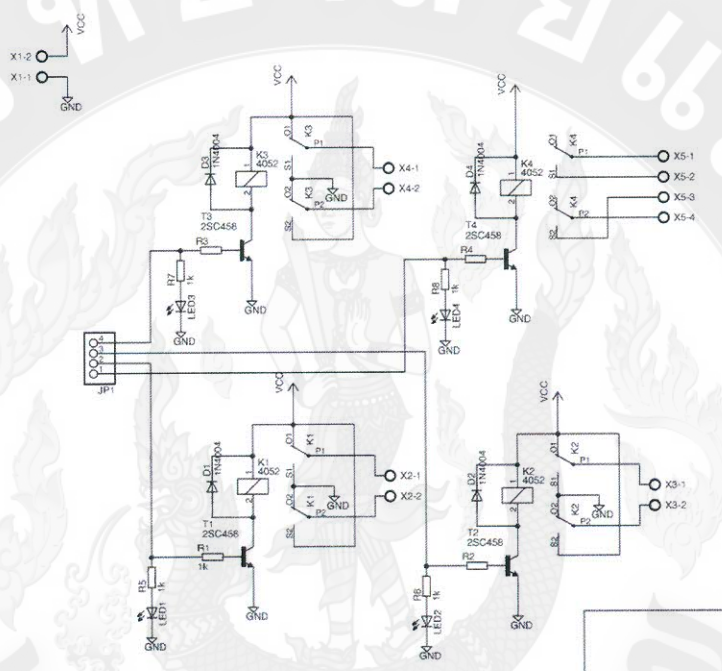
ภาพที่ 12 วงจรตรวจวัดความชื้นในดิน



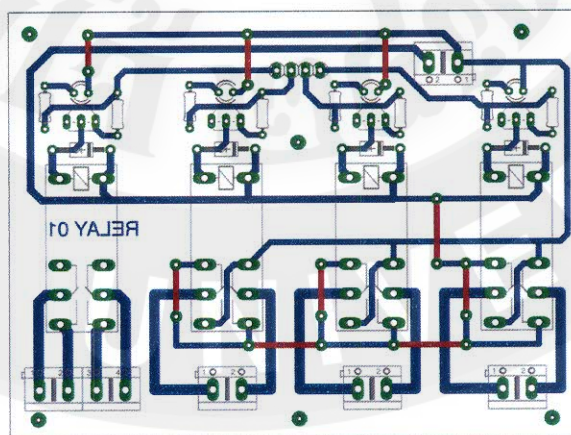
ภาพที่ 13 แผงวงจรตรวจวัดความชื้นในดิน

2. วงจรควบคุมวาล์วน้ำ

วงจรควบคุมวาล์วน้ำและเครื่องสูบน้ำทำหน้าที่ในการสั่งการเปิด/ปิดวาล์วน้ำ โดยอาศัยการสลับขั้วแรงดันไฟฟ้าเพื่อควบคุมทิศทางการหมุนของมอเตอร์โดยรีเลย์จะทำหน้าที่ในการสลับขั้วไฟฟ้าสำหรับการควบคุมเครื่องสูบน้ำ รีเลย์จะทำหน้าที่ในการขับสวิตช์แม่เหล็กอีกทอดหนึ่งเนื่องจากเครื่องสูบน้ำต้องการกระแสไฟฟ้ามากเกินไปจนจำกัดการทนกระแสของรีเลย์



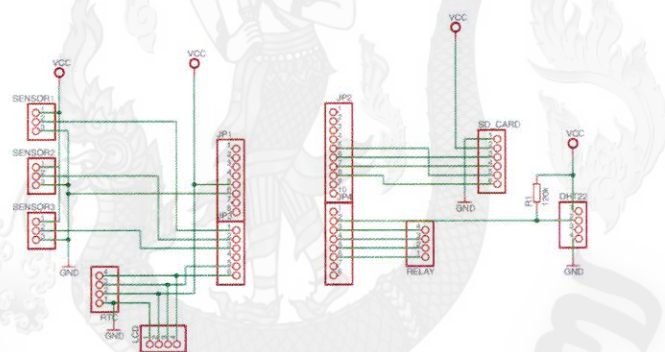
ภาพที่ 14 วงจรควบคุมวาล์วน้ำและเครื่องสูบน้ำ



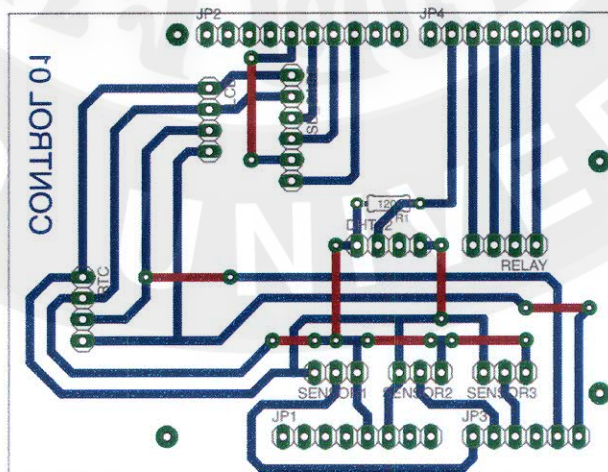
ภาพที่ 15 แผงวงจรควบคุมวาล์วน้ำและเครื่องสูบน้ำ

3. วงจรควบคุมระบบ

วงจรควบคุมระบบมีหน้าที่ในการอ่านค่าความชื้นในดินที่ตรวจวัดได้จากวงจรตรวจวัดค่าความชื้นในดินมาทำการประมวลผลเพื่อตัดสินใจในการจ่ายน้ำโดยควบคุมวาล์วน้ำและเครื่องสูบน้ำผ่านวงจรวงจรควบคุมวาล์วน้ำและเครื่องสูบน้ำ นอกจากนี้แล้ววงจรควบคุมระบบยังมีหน้าที่ในการเก็บบันทึกข้อมูลการสั่งเปิด/ปิดวาล์วน้ำและการจ่ายน้ำเข้าแปลงปลูก โดยจะเก็บข้อมูลในรูปแบบตารางด้วยหน่วยความจำแบบ SD Card แกนหลักของระบบนี้จะทำงานผ่านแพลตฟอร์ม Arduino ซึ่งเป็นแพลตฟอร์มที่ใช้กับไมโครคอนโทรเลอร์ตระกูล AVR ซึ่งในที่นี้ได้ใช้บอร์ดทดลอง Arduino รุ่น Arduino UNO R3 ซึ่งเป็นบอร์ดทดลองที่สามารถหาซื้อได้ง่ายและมีคุณสมบัติครอบคลุมตามความต้องการของงานวิจัยนี้ ในงานวิจัยนี้จะทำการออกแบบวงจรเสริมเข้ากับบอร์ดทดลอง Arduino เพื่อให้สามารถทำงานได้ตามความต้องการ โดยทำการเพิ่มส่วนแสดงผลข้อมูลผ่านจอ LCD ขนาด 20 ตัวอักษร 2 บรรทัด วงจรสัญญาณนาฬิกา และวงจรเขียน/อ่านข้อมูล SD Card ดังแสดงในภาพที่ 16



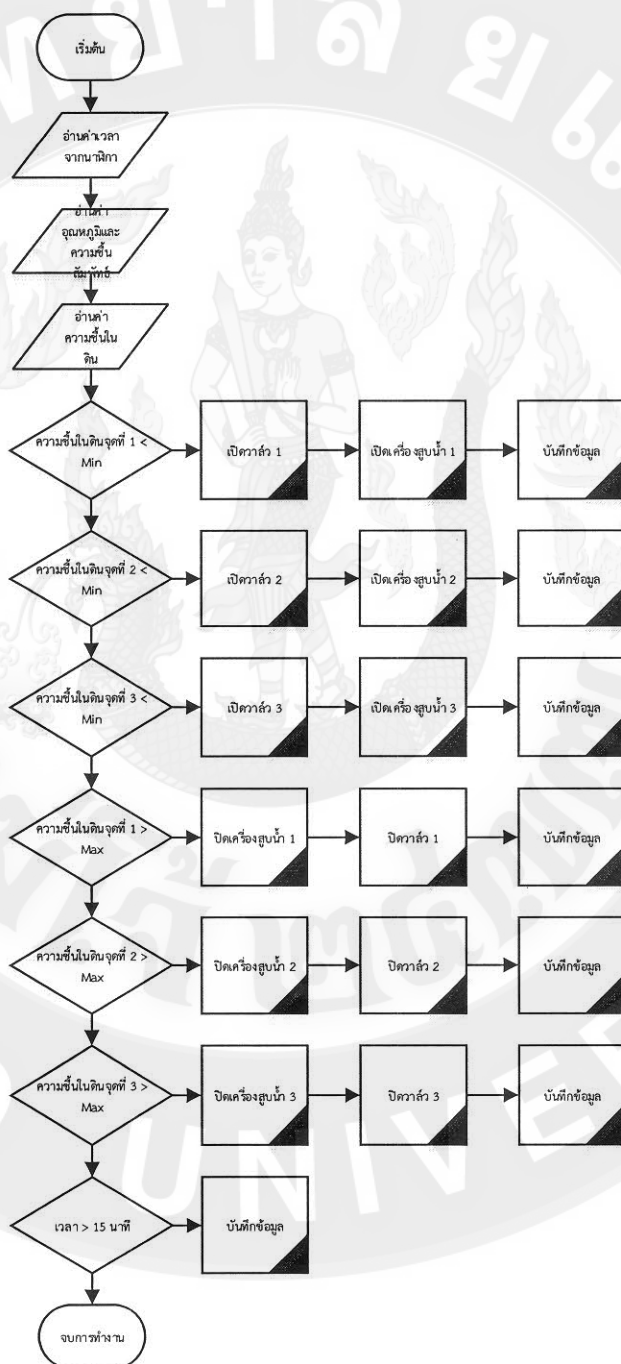
ภาพที่ 16 วงจรที่เพิ่มเติมเข้ากับบอร์ดไมโครคอนโทรเลอร์



ภาพที่ 17 แผงวงจรวงจรที่เพิ่มเติมเข้ากับบอร์ดไมโครคอนโทรเลอร์

การออกแบบซอฟต์แวร์

พัฒนาชุดคำสั่งควบคุมการทำงานของระบบด้วยภาษาซี (C Programming) ซึ่งพัฒนาชุดคำสั่งให้ทำงานบนแพลตฟอร์ม Arduino บนไมโครคอนโทรลเลอร์ตระกูล AVR เบอร์ ATmega328 โดยมีขั้นตอนการทำงานดังภาพ



ภาพที่ 18 แผนผังการทำงานของระบบควบคุม

สถานที่ทำการวิจัย

ทำการทดลองระบบให้น้ำแปลงปลูกข้าวโพดหวาน ณ แปลงสาธิตการปลูกข้าวโพด อำเภอแม่ว่าง จังหวัดเชียงใหม่ โดยทดลองปลูกข้าวโพดทั้งหมด 3 สายพันธุ์ ได้แก่ Hi-Brix 53 Sugar Star และ SCCT-384 ทำการปลูกโดยวิธีการให้น้ำ 4 ลักษณะ คือ 1) การให้น้ำด้วยวิธีน้ำหยดด้วยระบบอัตโนมัติ 2) การให้น้ำด้วยมินิสปริงเคลอร์ด้วยระบบอัตโนมัติ 3) การให้น้ำแบบร่องคูด้วยระบบอัตโนมัติ และ 4) การให้น้ำแบบดั้งเดิม ทำการเก็บข้อมูล 3 ซ้ำ รวมทั้งหมด 36 แปลง

ประชากรและกลุ่มตัวอย่าง

ทำการเก็บข้อมูลน้ำหนักผลผลิตจากการสุ่มตัวอย่างด้วยวิธีการสุ่มตัวอย่างแบบเป็นระบบ (Systematic sampling) ในแต่ละแปลงจำนวน 10 ตัวอย่าง จากนั้นคัดกรองข้อมูลโดยไม่นำข้อมูลน้ำหนักน้อยที่สุด 3 ค่าและค่าน้ำหนักมากที่สุด 2 ค่ามาใช้ในการหาค่าเฉลี่ย

วิธีการดำเนินการ

1. การสร้างกราฟมาตรฐานสำหรับการสอบเทียบค่าอุปกรณ์ตรวจวัดค่าความชื้นในดิน

ในการทดสอบการวัดความชื้นของดินโดยวิธีการใช้เซ็นเซอร์วัดความชื้นนั้น จำเป็นอย่างยิ่งที่จะต้องมีการพามาตรฐานเพื่อบอกความสัมพันธ์ระหว่างปริมาณความชื้นจริงที่มีอยู่ในดินและค่าความชื้นที่อ่านได้จากเครื่องมือที่ใช้วัด ดังนั้นจึงมีวิธีการดำเนินการดังนี้

1. ทำการเตรียมตัวอย่างดินที่มีค่าความชื้นต่างกันโดยทำการเติมน้ำลงในภาชนะบรรจุดินแต่ละอันในอัตราส่วนที่แตกต่างกันไป โดยคิดเป็นเปอร์เซ็นต์โดยน้ำหนักของดิน ดังนี้ 5, 10 และ 15 เปอร์เซ็นต์ ตามลำดับ
2. นำเซ็นเซอร์วัดความชื้นดินเสียบลงในตัวอย่างดินที่เตรียมไว้และอ่านค่าจากอุปกรณ์บันทึกข้อมูลลงในคอมพิวเตอร์
3. ทำการทดสอบซ้ำในข้อที่ 1-2 อีกจำนวน 3 ครั้ง
4. นำค่าแรงดันเอาต์พุตที่อ่านได้จากอุปกรณ์มาสร้างสมการความสัมพันธ์ระหว่างค่าแรงดันเอาต์พุตกับค่าปริมาณความชื้นในดิน สำหรับใช้โปรแกรมลงในไมโครคอนโทรลเลอร์

2. การวิเคราะห์หาค่าความชื้นที่จุดชลประทาน

เนื่องจากในทางปฏิบัติจะไม่ปล่อยให้ความชื้นในดินลดลงจนถึงจุดเหี่ยวเฉาถาวรเพราะจะมีผลกระทบต่อต้นไม้และผลผลิต ซึ่งโดยทั่ว ๆ ไปเมื่อต้นไม้ใช้น้ำในดินไปประมาณ 50 เปอร์เซ็นต์ของความชื้นที่เป็นประโยชน์ทั้งหมดก็จะเริ่มให้น้ำชลประทาน แต่จะไม่ให้มากกว่าความชื้นชลประทาน เพราะจะทำให้สูญเสียน้ำไปกับการซึมลงด้านล่างของชั้นดิน จึงมีการทดสอบหาค่าความชื้นที่จุดชลประทาน โดยมีวิธีการดังนี้

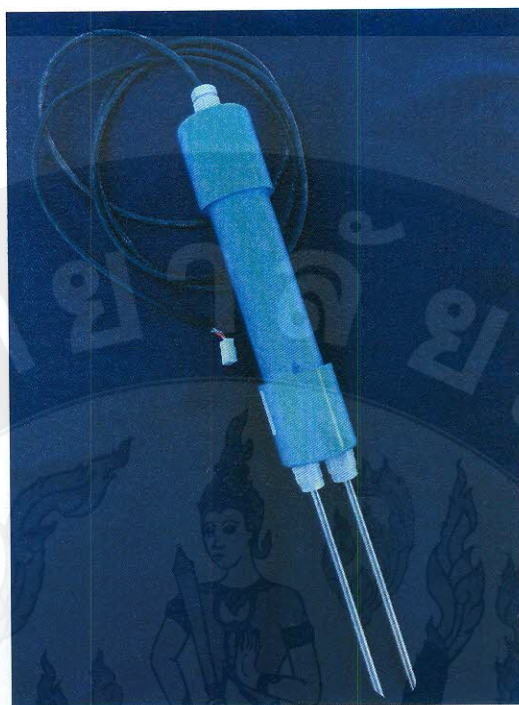
1. ชั่งน้ำหนักกระบอกสแตนเลสโดยหุ้มผ้าขาวบางและมัดด้วยลวด
2. นำดินตัวอย่างที่ใช้ทดสอบใส่กระบอกสแตนเลสจนเต็มแล้วชั่งน้ำหนัก
3. นำกระบอกสแตนเลสที่มีดินอยู่เต็มกระบอกวางลงบนภาชนะที่มีน้ำ จนกระทั่งดินอิ่มตัวด้วยน้ำ จากนั้นนำไปชั่งน้ำหนัก
4. นำไปอบด้วยเครื่องอบดินที่อุณหภูมิ 105-110 องศาเซลเซียส ประมาณ 24 ชั่วโมง หลังจากนั้นรอให้เย็น แล้วจึงนำไปชั่งน้ำหนัก
5. นำค่าที่ได้บันทึกไว้คำนวณค่าความชื้นที่จุดชลประทานของดินชุดทดสอบ
6. คำนวณค่า 50 เปอร์เซ็นต์ของความชื้นที่เป็นประโยชน์ของดินชุดทดสอบ ซึ่งเป็นจุดที่เหมาะสมในการเริ่มต้นรดน้ำของระบบ เพื่อนำไปเป็นค่าเงื่อนไขการให้น้ำในโปรแกรมควบคุม

3. การออกแบบและสร้างระบบตรวจวัดและควบคุมระบบการให้น้ำ

การวิจัยในครั้งนี้จะทำการออกแบบและสร้างระบบการให้น้ำอัตโนมัติแบบแม่นยำโดยลักษณะโครงสร้างของต้นแบบระบบการให้น้ำจากการตรวจวัดความชื้นในดิน ประกอบด้วย

1. เซ็นเซอร์วัดความชื้นดิน (Soil moisture sensor)

ทำหน้าที่ในการตรวจวัดค่าความชื้นในดินด้วยหลักการความต้านทานทางไฟฟ้า ถ้าดินมีความชื้นต่ำค่าความต้านทานจะสูง ถ้าดินมีความชื้นสูงค่าความต้านทานจะต่ำ จากนั้นวงจรตรวจวัดจะส่งข้อมูลทางไฟฟ้าให้ไมโครคอนโทรลเลอร์ประมวลผล



ภาพที่ 19 เซนเซอร์วัดความชื้นในดิน

2. ตัวแปลงผันสัญญาณอนาลอกเป็นสัญญาณดิจิทัล (Analog to digital converter)

วงจรแปลงสัญญาณอนาลอกเป็นสัญญาณดิจิทัลจะทำการเปลี่ยนสัญญาณไฟฟ้าในรูปแบบอนาลอกเป็นสัญญาณไฟฟ้าแบบดิจิทัลซึ่งมีลักษณะเป็นสัญญาณเชิงเลข ส่งให้กับหน่วยประมวลผลทำการประมวลผลข้อมูลได้

3. หน่วยประมวลผลไมโครคอนโทรลเลอร์ (Microcontroller unit)

ทำการประมวลผลค่าระดับความชื้นที่ถูกแปลงสัญญาณมาเป็นสัญญาณดิจิทัลแล้ว เพื่อทำการตัดสินใจในการจ่ายน้ำหรือหยุดจ่ายน้ำให้กับแปลงปลูกข้าวโพดหวาน

4. วงจรควบคุมรีเลย์

ทำหน้าที่ควบคุมการเปิดปิดอุปกรณ์จ่ายน้ำเข้าแปลงเพาะปลูกโดยรับสัญญาณควบคุมการปิด/เปิดวาล์วและเครื่องสูบน้ำจากไมโครคอนโทรลเลอร์

การศึกษาในครั้งนี้ใช้ไอซีไมโครคอนโทรลเลอร์บอร์ด ATmega 328 Arduino เพื่อทำการประมวลผลข้อมูล โดยการแปลงค่าสัญญาณดิจิทัลเอาท์พุตที่อ่านได้เป็นค่าแรงดันอินพุตอนาลอกซึ่งค่าแรงดันอินพุตอนาลอกที่วัดได้ จะถูกนำไปประมวลผลด้วยโปรแกรมที่บรรจุไว้ในตัว

ไมโครคอนโทรลเลอร์ให้เป็นเปอร์เซ็นต์ความชื้นของดิน จากนั้นข้อมูลจะถูกส่งไปยังชุดควบคุมกลาง (Central control unit) พร้อมกับเก็บบันทึกข้อมูลต่าง ๆ ไว้ในอุปกรณ์บันทึกผล จากนั้นโปรแกรม จะทำการคำนวณหาปริมาณการใช้น้ำที่จุดเหมาะสม โดยใช้การตรวจวัดความชื้นในดินโดยตรง ซึ่งถ้า ถึงจุดที่เหมาะสมในการให้น้ำชุดไมโครคอนโทรลเลอร์จะส่งสัญญาณควบคุมเอาต์พุตแบบดิจิทัลไป ควบคุมวงจรรีเลย์ให้ควบคุมการเปิด/ปิด วาล์วหรือเครื่องสูบน้ำเพื่อย้ายน้ำ

การเก็บข้อมูลและวิเคราะห์ผล

กำหนดตัวแปรสำหรับการทดลองดังนี้ ตัวแปรต้น ได้แก่ วิธีการให้น้ำแก่แปลงปลูกข้าวโพด หวาน ตัวแปรตาม ได้แก่ น้ำหนักของผลผลิต ขนาดของผลผลิต ความสมบูรณ์ของฝัก และตัวแปร ควบคุม ได้แก่ การให้ปุ๋ย การให้ยากำจัดศัตรูพืช สภาพดินที่เพาะปลูก

1. ข้อมูลผลผลิต

ทำการเก็บผลผลิตในแต่ละแปลงโดยแยกเป็นสายพันธุ์ ระบบการให้น้ำ และจำนวนซ้ำ ทำ การเก็บค่าน้ำหนักผลผลิตทั้งแปลง ทำการสุ่มตรวจวัดขนาดฝัก ความยาวฝัก ความสมบูรณ์ของฝัก จำนวนเมล็ดรอบฝัก รวมไปถึงค่า Brix ของข้าวโพดหวานแต่ละสายพันธุ์ และแต่ละวิธีการให้น้ำ เพื่อ คัดแยกคุณภาพ โดยกำหนดให้ ข้าวโพดหวานที่มีขนาดฝักยาวเกิน 18 เซนติเมตรและมีน้ำหนัก มากกว่า 400 กรัม มีเมล็ดเต็มฝักเป็นคุณภาพเกรด A ข้าวโพดหวานที่มีขนาดฝักสั้นกว่า 15 เซนติเมตรและมีน้ำหนักมากกว่า 300 กรัมแต่ไม่เกิน 400 กรัม เป็นคุณภาพเกรด B ส่วนฝักใดที่ไม่ เข้าเกณฑ์ที่กำหนดนี้ให้กำหนดคุณภาพเป็นเกรด C จากนั้นนำข้อมูลที่ได้ทั้งหมดมาทำการวิเคราะห์ ทางสถิติด้วยการวิเคราะห์ความแปรปรวน (ANOVA) และ Tukey's HSD test ด้วยโปรแกรม R 3.3.2 เพื่อหาความแตกต่างของผลผลิตเทียบกับวิธีการให้น้ำในแต่ละแบบ โดยกำหนดค่า confidential value 0.05

2. ข้อมูลผลการใช้น้ำ

เก็บข้อมูลที่บันทึกภายในการคำนวณหาความจำเป็นการใช้น้ำโดยหาได้จากระยะเวลาที่ ระบบจ่ายน้ำเข้าสู่แปลงปลูกเทียบกับอัตราการไหลของน้ำในแต่ละระบบ

3. ข้อมูลการใช้พลังงานไฟฟ้า

เก็บข้อมูลการใช้พลังงานจากมิเตอร์ซึ่งติดตั้งในแต่ละระบบการให้น้ำ จากนั้นนำข้อมูลการใช้พลังงานของทุกระบบการให้น้ำตลอดการเพาะปลูกมาเปรียบเทียบเพื่อหาอัตราการใช้พลังงานของแต่ละระบบ

4. การวิเคราะห์เศรษฐศาสตร์

การวิเคราะห์เศรษฐศาสตร์ของระบบให้น้ำแบบอัตโนมัติในวิธีการให้น้ำแต่ละแบบ โดยจะทำการวิเคราะห์ต้นทุนในการผลิตทั้งทางด้านการใช้ไฟฟ้า และต้นทุนในการผลิตทางด้านการใช้น้ำ ตั้งแต่เริ่มต้นเพาะปลูกจนถึงสิ้นสุดกระบวนการเพาะปลูก เพื่อเปรียบเทียบต้นทุนในการผลิตกับการให้น้ำแบบดั้งเดิมและหาจุดคุ้มทุนของวิธีการให้น้ำในแต่ละแบบ โดยที่การคำนวณหาจุดคุ้มทุนสามารถหาได้จาก

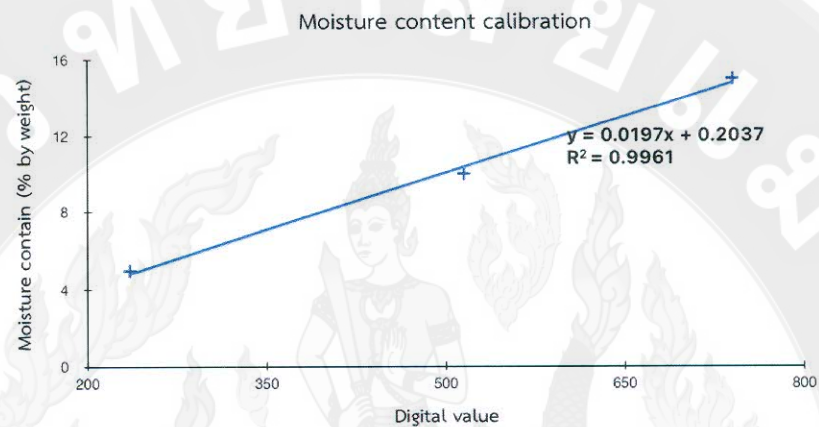
$$\text{จุดคุ้มทุน} = \frac{\text{ต้นทุนคงที่}}{\text{ราคาขาย} - \text{ต้นทุนผันแปร}} \quad (10)$$

โดยกำหนดให้ราคารับซื้อข้าวโพดโรงงานราคากิโลกรัมละ 5 บาทโดยไม่คัดแยกเกรด

บทที่ 4

ผลการวิจัยและวิจารณ์

1. ผลการสร้างกราฟมาตรฐานสำหรับการสอบเทียบค่าอุปกรณ์ตรวจวัดค่าความชื้นในดิน

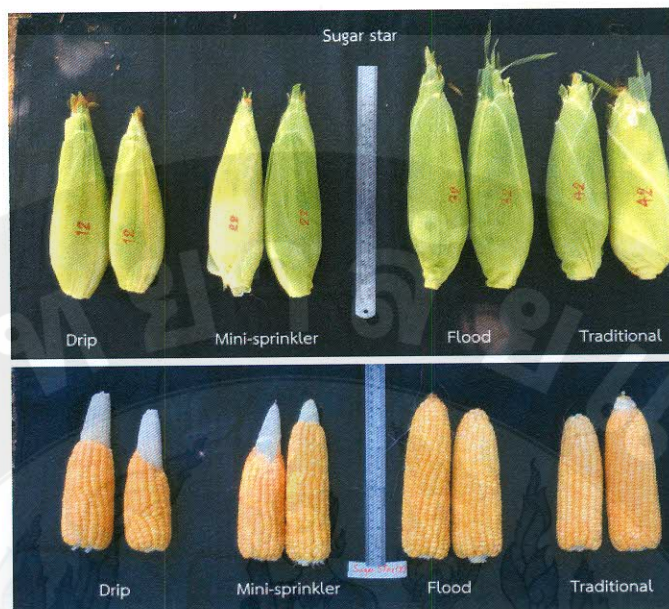


ภาพที่ 20 กราฟมาตรฐานสำหรับหาค่าความชื้นในดินด้วยการเปรียบเทียบแรงดันไฟฟ้า

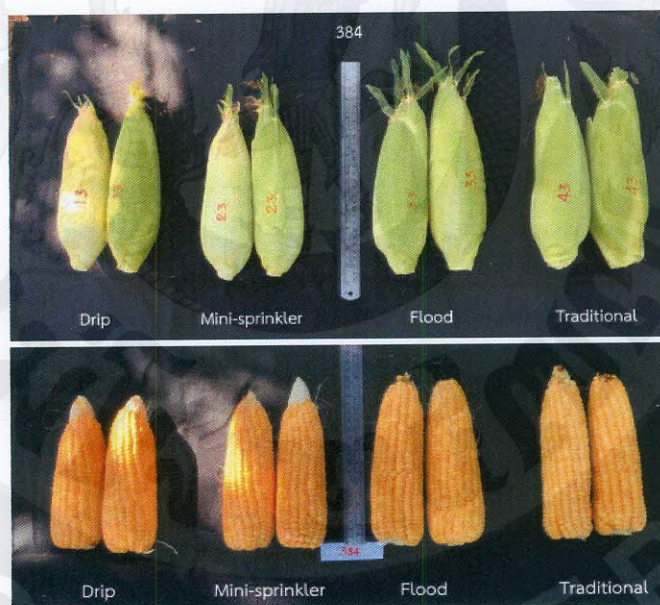
2. ข้อมูลผลผลิตข้าวโพดหวานภายใต้การระบบการให้น้ำแต่ละแบบ



ภาพที่ 21 ตัวอย่างผลผลิตข้าวโพดหวานพันธุ์ Hi-Brix 53



ภาพที่ 22 ตัวอย่างผลผลิตข้าวโพดหวานพันธุ์ Sugar star



ภาพที่ 23 ตัวอย่างผลผลิตข้าวโพดหวานพันธุ์ SCCT-384

สำหรับข้อมูลผลผลิตที่ได้จากการเพาะปลูกโดยใช้วิธีการให้น้ำที่ต่างกันดังแสดงในตาราง

ตารางที่ 1 ข้อมูลผลผลิตจากการเพาะปลูกภายใต้วิธีการให้น้ำที่แตกต่างกัน

วิธีการให้น้ำ	น้ำหนัก (กรัม)		ความยาวฝัก	เส้นรอบวงฝัก	ความสูงลำต้น	จำนวนใบ	%Brix
	รวมเปลือก	แกะเปลือก					
น้ำหยด	295.13	240.93	18.07	18.53	229.20	12	13
	385.33	283.93	20.00	18.60	273.00	12	13
	328.80	251.73	18.51	18.17	271.93	12	14
พ่นฝอย	370.70	291.80	19.30	18.50	272.80	11	14
	357.07	286.77	18.37	18.00	281.13	11	14
	401.20	319.23	20.20	18.77	288.47	12	13
ร่องคู	524.83	402.47	21.40	19.70	267.93	12	13
	529.63	393.90	20.63	19.50	270.40	12	13
	482.80	365.50	19.50	19.20	268.00	12	13
ดั้งเดิม	498.87	353.37	20.30	18.73	268.73	12	13
	510.27	378.03	20.40	19.17	268.40	12	13
	527.00	363.93	20.73	19.03	258.73	11	13

ตารางที่ 2 การเปรียบเทียบความแตกต่างของผลผลิตจากการให้น้ำแต่ละวิธีการ

การเปรียบเทียบวิธีการให้น้ำ	P-Value
ร่องคู – น้ำหยด	0.000
พ่นฝอย – น้ำหยด	0.399
ดั่งเดิม – น้ำหยด	0.000
พ่นฝอย – ดั่งเดิม	0.002
ดั่งเดิม – ร่องคู	0.999
ดั่งเดิม – พ่นฝอย	0.002

จากตารางที่ 1 แสดงให้เห็นว่าการเจริญเติบโตของลำต้นมีการเจริญเติบโตที่ใกล้เคียงกันโดยความสูงของลำต้นเฉลี่ยอยู่ในช่วง 2 - 3 เมตร มีจำนวนใบ 11 - 12 ใบ แต่น้ำหนักฝักสดมีความแตกต่างกันอย่างมีนัยสำคัญ กล่าวคือ การให้น้ำด้วยวิธีร่องคูและวิธีดั่งเดิมมีน้ำหนักที่ใกล้เคียงกันและเป็นกลุ่มที่มีน้ำหนักมากที่สุด เมื่อเปรียบเทียบความแตกต่างทางสถิติตารางที่ 3 พบว่าน้ำหนักสดของผลผลิตสามารถแบ่งออกได้เป็น 2 กลุ่มคือ กลุ่มของผลผลิตที่ได้จากวิธีการให้น้ำแบบน้ำหยดกับวิธีการให้น้ำแบบพ่นฝอย และกลุ่มของผลผลิตที่ได้จากวิธีการให้น้ำแบบร่องคูกับวิธีการให้น้ำแบบดั่งเดิม

ตารางที่ 3 การเปรียบเทียบคุณภาพของผลผลิตจากการให้น้ำแต่ละวิธีการ

วิธีการให้น้ำ	เกรด	น้ำหนักฝักเฉลี่ย (g)	จำนวนฝัก/ไร่	น้ำหนักต่อไร่(kg)
น้ำหยด	A	422.8	1,004	424.5
	B	380.1	3,661	1,391.5
	C	274.9	4,224	1,161.2
พ่นฝอย	A	469.9	1,776	834.5
	B	404	3,267	1,319.9
	C	316.8	3,845	1,218.1
ร่องคู	A	523.7	5,992	3,138.0
	B	496.7	1,559	774.4
	C	451.6	1,339	604.7
ดั่งเดิม	A	469.2	2,701	1,267.3
	B	545.2	5,154	2,810.0
	C	419.8	1,034	434.1

ตารางที่ 4 ร้อยละของคุณภาพผลผลิตจากการให้น้ำแต่ละวิธีการ

วิธีการให้น้ำ	เกรด	คิดเป็นร้อยละ
น้ำหยด	A	11.29
	B	41.19
	C	47.52
พ่นฝอย	A	19.98
	B	36.75
	C	43.26
ร่องคู	A	67.41
	B	17.45
	C	15.06
ดั้งเดิม	A	30.39
	B	57.98
	C	11.63

เมื่อเปรียบเทียบผลผลิตในเชิงของปริมาณโดยเทียบน้ำหนักผลผลิตต่อไร่พบว่าวิธีการให้น้ำแบบร่องคูเป็นวิธีการให้น้ำที่ให้น้ำหนักต่อไร่มากที่สุดรองลงมาคือวิธีการให้น้ำด้วยวิธีดั้งเดิม วิธีการให้น้ำด้วยวิธีพ่นฝอย และวิธีการให้น้ำด้วยวิธีน้ำหยดตามลำดับ

เมื่อเปรียบเทียบผลผลิตในเชิงของคุณภาพพบว่าวิธีการให้น้ำแบบร่องคูเป็นวิธีที่มีผลผลิตเกรด A มากที่สุดในทุกวิธีการให้น้ำ รองลงมาคือวิธีการให้น้ำด้วยวิธีดั้งเดิม วิธีการให้น้ำด้วยวิธีพ่นฝอย และวิธีการให้น้ำแบบน้ำหยดตามลำดับ

3. ข้อมูลการใช้น้ำและพลังงานไฟฟ้า

ตารางที่ 5 ข้อมูลการใช้พลังงานไฟฟ้าตลอดช่วงการเพาะปลูก

ระบบให้น้ำ	จำนวนต้น	พื้นที่ (ไร่)	เลข มิเตอร์ ก่อนปลูก	เลข มิเตอร์ หลังปลูก	ปริมาณการใช้ พลังงาน (หน่วย)	การใช้ พลังงาน/ ไร่
น้ำหยด	563	0.063	2.5	18	15.5	244.72
พ่นฝอย	541	0.061	14.6	23.3	8.7	142.95
ร่องคู	301	0.034	0.3	1.5	1.2	35.44
ดั้งเดิม	304	0.034	0.3	1.9	1.6	46.78

การใช้พลังงานไฟฟ้าตลอดการเพาะปลูกเมื่อเทียบเป็นการใช้พลังงานต่อพื้นที่ พบว่าวิธีการให้น้ำแบบน้ำหยดเป็นวิธีการที่ใช้พลังงานไฟฟ้ามากที่สุดและวิธีการให้น้ำด้วยวิธีการให้น้ำแบบร่องคูเป็นวิธีการที่ใช้พลังงานไฟฟ้าน้อยที่สุด

ตารางที่ 6 ข้อมูลการใช้น้ำด้วยวิธีให้น้ำแบบน้ำหยดตลอดช่วงการเพาะปลูก

อัตราการให้น้ำ (ลิตร)	เวลา (นาท)	จำนวน แปลง	ปริมาณน้ำที่ใช้ (ลิตร)	จำนวนต้น	คิดเป็นหน่วย พื้นที่ (ไร่)
5.074	325	3	4,947.15	204	0.023
4.386	324	3	4,263.19	213	0.024
4.563	286	3	3,915.05	146	0.016
	รวม	9	1,3125.40	563	0.063
รวมปริมาณการใช้น้ำทั้งฤดูเพาะปลูก			208.33	ลูกบาศก์เมตร/ไร่	

ตารางที่ 7 ข้อมูลการใช้น้ำด้วยวิธีให้น้ำแบบพ่นฝอยตลอดช่วงการเพาะปลูก

อัตราการให้น้ำ (ลิตร)	เวลา (นาทีก)	จำนวน แปลง	ปริมาณน้ำที่ใช้ (ลิตร)	จำนวน ต้น	คิดเป็นหน่วย พื้นที่ (ไร่)
7.833	115	3	2,702.39	147	0.017
7.833	133	3	3,125.37	195	0.022
7.833	129	3	3,031.37	199	0.022
	รวม	9	8,859.12	541	0.061
รวมปริมาณการใช้น้ำทั้งฤดูเพาะปลูก			145.23	ลูกบาศก์เมตร/ไร่	

ตารางที่ 8 ข้อมูลการใช้น้ำด้วยวิธีให้น้ำแบบร่องคูตลอดช่วงการเพาะปลูก

อัตราการให้น้ำ (ลิตร)	เวลา (นาทีก)	จำนวน แปลง	ปริมาณน้ำที่ใช้ (ลิตร)	จำนวน ต้น	คิดเป็นหน่วย พื้นที่ (ไร่)
52.8	60	3	9,504.00	95	0.011
52.8	63	3	9,979.20	101	0.011
52.8	67	3	10,612.80	105	0.012
	รวม	9	30,096.00	301	0.034
รวมปริมาณการใช้น้ำทั้งฤดูเพาะปลูก			885.18	ลูกบาศก์เมตร/ไร่	

ตารางที่ 9 ข้อมูลการใช้น้ำด้วยวิธีให้น้ำแบบดั้งเดิมตลอดช่วงการเพาะปลูก

อัตราการ ให้น้ำ (ลิตร)	เวลา (นาทีก)	จำนวน แปลง	ปริมาณน้ำ ที่ใช้ (ลิตร)	จำนวน ต้น	คิดเป็นหน่วย พื้นที่ (ไร่)
52.8	75	3	11,880.00	103	0.012
52.8	75	3	11,880.00	100	0.011
52.8	75	3	11,880.00	101	0.011
	รวม	9	35,640.00	304	0.034
รวมปริมาณการใช้น้ำทั้งฤดูเพาะปลูก			1,048.23	ลูกบาศก์เมตร/ไร่	

จากข้อมูลการใช้น้ำตลอดฤดูการเพาะปลูกของแต่ละวิธีการเมื่อเทียบเป็นการให้น้ำต่อพื้นที่ ดังแสดงในตารางที่ 6-9 พบว่าวิธีการให้น้ำด้วยวิธีการดั้งเดิมเป็นวิธีที่ใช้น้ำมากที่สุดเมื่อเทียบกับทั้ง 4 วิธีโดยมีการใช้น้ำตลอดทั้งฤดูการเพาะปลูกอยู่ที่ 1,048.23 ลูกบาศก์เมตรต่อไร่ และวิธีการให้น้ำด้วยวิธีการพ่นฝอยเป็นวิธีที่ใช้น้ำน้อยที่สุดคือ 145.23 ลูกบาศก์เมตรต่อไร่

4. การวิเคราะห์ต้นทุนและจุดคุ้มทุนของการให้น้ำด้วยวิธีการที่แตกต่างกัน

ในการวิจัยครั้งนี้สามารถสรุปต้นทุนปัจจัยที่ใช้ในการเพาะปลูกข้าวโพดหวานด้วยระบบการให้น้ำแบบอัจฉริยะและการให้น้ำแบบดั้งเดิมรวมไปถึงจุดคุ้มทุนของระบบให้น้ำแบบต่าง ๆ ได้ดังตารางที่ 10

ตารางที่ 10 ต้นทุนของการปลูกข้าวโพดหวานด้วยวิธีการให้น้ำแบบต่าง ๆ

ต้นทุน	ระบบน้ำหยด			ระบบพ่นฝอย			ระบบร่องคู			ระบบคังเดิม		
	จำนวน (หน่วย)	ราคาต่อ หน่วย	รวม (บาท)	จำนวน (หน่วย)	ราคาต่อ หน่วย	รวม (บาท)	จำนวน (หน่วย)	ราคาต่อ หน่วย	รวม (บาท)	จำนวน (หน่วย)	ราคาต่อ หน่วย	รวม (บาท)
ต้นทุนคงที่												
ท่อประธาน	1	3,430	3,430	1	3,430	3,430	1	3,430	3,430	1	3,430	3,430
ท่อแขนง	1	2,340	2,340	1	9,300	9,300	-	-	-	-	-	-
เครื่องสูบน้ำ	1	5,000	5,000	1	5,000	5,000	1	5,000	5,000	1	5,000	5,000
ระบบควบคุมการให้น้ำ	1	10,000	10,000	1	10,000	10,000	1	10,000	10,000	-	-	-
			20,770			27,730			18,430			8,430
ต้นทุนผันแปร												
ค่าเตรียมพื้นที่	1	1,200	1,200	1	1,200	1,200	1	1,200	1,200	1	1,200	1,200
เมล็ดพันธุ์	1	816.67	816.67	1	816.67	816.67	1	816.67	816.67	1	816.67	816.67
ปุ๋ยสูตร 13-13-21	1	750	750	1	750	750	1	750	750	1	750	750
ปุ๋ยสูตร 46-0-0	1	550	550	1	550	550	1	550	550	1	550	550
ยาคุมหญ้า	1	291.67	291.67	1	291.67	291.67	1	291.67	291.67	1	291.67	291.67
ยาป้องกันแมลง	1	570	570	1	570	570	1	570	570	1	570	570
ค่าจ้างปลูก	1	600	600	1	600	600	1	600	600	1	600	600
ค่าจ้างพ่นยา	4	500	2,000	4	500	2,000	4	500	2,000	4	500	2,000
ค่าจ้างเก็บเกี่ยว	1	1,200	1,200	1	1,200	1,200	1	1,200	1,200	1	1,200	1,200
ค่าจ้างติดตั้งอุปกรณ์	1	600	600	1	600	600	1	600	600	-	-	-
ค่าจ้างดูแลการเพาะปลูก	-	-	-	-	-	-	-	-	-	1	1,500	1,500
ค่าไฟฟ้า	244.72	3	734.16	142.95	3	428.85	35.44	3	106.32	46.78	3	140.43
			30,082.5			36,737.19			27,144.66			18,048.6
จุดคุ้มทุน (รอบการเพาะปลูก)			10			7			3			2

บทที่ 5

สรุปและข้อเสนอแนะ

งานวิจัยนี้ได้ทำการออกแบบและสร้างระบบควบคุมการให้น้ำแบบอัตโนมัติสำหรับศึกษาการปลูกข้าวโพดหวานเพื่อให้ได้ระบบการให้น้ำที่มีความแม่นยำ ศึกษาผลผลิตที่ได้จากการให้น้ำด้วยวิธีการที่แตกต่างกันรวมไปถึงศึกษาทางด้านเศรษฐศาสตร์เมื่อนำระบบการให้น้ำแบบแม่นยำไปช่วยในการเพาะปลูกเปรียบเทียบกับ การเพาะปลูกแบบดั้งเดิม

สรุปผลการวิจัย

การวิจัยในครั้งนี้สามารถสรุปได้ว่าปริมาณการใช้น้ำของการวิธีการให้น้ำแบบต่าง ๆ จะขึ้นอยู่กับอัตราการให้น้ำของแต่ละระบบและเวลาในการให้น้ำ การให้น้ำด้วยวิธีการแบบร่องคูทั้งแบบดั้งเดิมและแบบใช้อุปกรณ์ตรวจวัดความชื้นมีอัตราการให้น้ำมากที่สุด รองลงมาคือการให้น้ำแบบผิวดินและการให้น้ำแบบน้ำหยดตามลำดับ ในส่วนของการใช้พลังงานไฟฟ้านั้นการให้น้ำแบบหยดเป็นการใช้พลังงานมากที่สุดเนื่องจากอัตราการให้น้ำมีค่าน้อยจึงต้องทำการจ่ายน้ำเป็นระยะเวลานานจนกว่าระดับความชื้นในดินจะมีค่าจนถึงระดับความชื้นที่ระบบจะทำการหยุดจ่ายน้ำและการให้น้ำแบบร่องคูที่มีอุปกรณ์ตรวจวัดความชื้นในดินมีการใช้พลังงานไฟฟ้าที่น้อยที่สุดเนื่องจากอัตราการให้น้ำที่สูงจึงทำให้ความชื้นในดินเพิ่มขึ้นอย่างรวดเร็วรวมกับการตรวจวัดที่สามารถควบคุมการจ่ายน้ำได้แม่นยำมากกว่าการให้น้ำแบบดั้งเดิม

สำหรับคุณภาพและปริมาณของผลผลิตที่ได้นั้นพบว่า การให้น้ำแบบร่องคูทั้งแบบดั้งเดิมและแบบใช้อุปกรณ์ตรวจวัดค่าความชื้นในดินมีปริมาณน้ำหนักรากที่ไม่มีความแตกต่างกันทางสถิติแต่ในด้านคุณภาพ จำนวนของฝักข้าวโพดที่มีความสมบูรณ์ในเกรด A ของระบบให้น้ำแบบร่องคูที่ใช้ อุปกรณ์ตรวจวัดค่าความชื้นในดินมีจำนวนฝักที่มากกว่าวิธีการให้น้ำแบบดั้งเดิม

ในการวิเคราะห์ทางเศรษฐศาสตร์ของระบบให้น้ำในทุกระบบเทียบกับปริมาณและคุณภาพของผลผลิตที่ได้ พบว่าระบบการให้น้ำแบบน้ำหยดที่เป็นระบบที่มีระยะเวลาการคืนทุนมากที่สุดคือ 10 ฤดูการเพาะปลูกโดยประมาณ นอกจากนี้แล้วหากพิจารณาถึงต้นทุนการเพาะปลูกและจุดคุ้มทุนพบว่าวิธีการให้น้ำแบบดั้งเดิมมีต้นทุนและจุดคุ้มทุนที่น้อยที่สุด แต่หากพิจารณาถึงคุณภาพของผลผลิตแล้วพบว่าวิธีการให้น้ำแบบร่องคูให้ผลผลิตคุณภาพเกรด A มากกว่าวิธีการให้น้ำทุกแบบและมีต้นทุนใกล้เคียงกับวิธีการให้น้ำแบบดั้งเดิมและจุดคุ้มทุนมากกว่าวิธีการให้น้ำแบบดั้งเดิมเพียง 1 รอบการเพาะปลูกเท่านั้น

ข้อเสนอแนะ

1. การใช้วิธีการตรวจวัดค่าความชื้นในดินเพียงอย่างเดียวอาจไม่เพียงพอต่อการตัดสินใจในการควบคุมการให้น้ำแก่พืช ซึ่งการเจริญเติบโตของพืชนั้นอาศัยปัจจัยหลายปัจจัยในการเจริญเติบโตเพื่อให้ระบบมีความสมบูรณ์และมีการทำงานที่มีความแม่นยำ จำเป็นต้องใช้ปัจจัยสภาพแวดล้อมอื่นๆ เข้ามาประมวลผลด้วย เช่น การใช้ข้อมูลจากสถานีตรวจวัดอากาศในระดับจุลภาคก็จะทำให้ระบบมีความชาญฉลาดและสามารถตัดสินใจได้ดีขึ้น
2. พัฒนาอัลกอริทึมของระบบให้สามารถนำค่าปัจจัยอื่น ๆ มารวมตัดสินใจในการให้น้ำเพื่อให้เกิดความแม่นยำ และให้น้ำได้อย่างเหมาะสม เช่น นำปัจจัยทางด้านอุณหภูมิ ความชื้นสัมพัทธ์ ในอากาศและเวลามาร่วมด้วย อาทิเช่น วิธีการหาค่าแรงดึงน้ำของอากาศ ก็จะเป็นอีกปัจจัยหนึ่งที่สามารถนำมาตัดสินใจในการควบคุมอุณหภูมิและความชื้นในอากาศเพื่อให้พืชสามารถเกิดกระบวนการสังเคราะห์ด้วยแสงได้อย่างต่อเนื่อง ซึ่งจะส่งผลถึงอัตราการเจริญเติบโตของพืชที่ดีขึ้นด้วย
3. พัฒนาอุปกรณ์วัดค่าความชื้นในดินให้สามารถใช้ได้กับดินหลายชนิดได้อย่างแม่นยำโดยไม่ต้องหาค่ามาตรฐานเพื่อสอบเทียบค่าในดินแต่ละชนิด เนื่องจากขีดจำกัดของวิธีการตรวจวัดความชื้นในดินด้วยหลักการความต้านทานไฟฟ้า ทำให้ต้องมีการสร้างกราฟมาตรฐานสำหรับตรวจวัดปริมาณความชื้นในดินเป็นการเฉพาะกับดินในแต่ละแปลงเพาะปลูก ทำให้เกิดความยุ่งยากและเสียเวลาในการปฏิบัติงาน โดยเฉพาะอย่างยิ่งเมื่อมีการนำไปใช้งานจริงกับเกษตรกร

บรรณานุกรม

- ชัยวัฒน์ ลิ้มพรจิตรวิไล และ วรพจน์ กรแก้ววัฒนกุล. 2543. **เรียนรู้และปฏิบัติ ไมโครคอนโทรเลอร์ MCS-51**. กรุงเทพฯ: อินโนเวทีฟ เอ็กเพอริเมนต์.
- ดิเรก ทองอร่าม, วิทยา ตั้งก่อสกุล, นาวิ จิระชีวี และ อธิสุนทร นันทกิจ. 2545. **การออกแบบและเทคโนโลยีการให้น้ำแก่พืช**. กรุงเทพฯ: ฐานการพิมพ์.
- พานิช อินต๊ะ. 2549ก. การเขียนโปรแกรมเชื่อมต่อคอมพิวเตอร์กับอุปกรณ์ภายนอก (1). วารสาร Electrical&Control, 4(23), 61-67.
- _____. 2549ข. การเขียนโปรแกรมเชื่อมต่อคอมพิวเตอร์กับอุปกรณ์ภายนอก (2). วารสาร Electrical&Control, 5(27), 57-61.
- พานิช อินต๊ะ และ โกศล โอฬารไพโรจน์. 2550. **เรียนรู้การเชื่อมต่อคอมพิวเตอร์กับอุปกรณ์ภายนอกด้วย Visual Basic**. กรุงเทพฯ: The Knowledge center.
- ราชศักดิ์ มานิมนต์ และ สุขใจ มาไชยะ. 2546. **เครื่องมือวัดความชื้นในดินโดยวิธีวัดค่าความต้านทาน**. ขอนแก่น: ภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ มหาวิทยาลัยขอนแก่น.
- วิบูลย์ บุญจรโรกุล. 2526. **หลักการชลประทาน**. กรุงเทพฯ: เอเชีย.
- สมบุรณ์ มั่นความดี. ม.ป.ป. **ความสัมพันธ์ระหว่าง ดิน น้ำ พืช**. กรุงเทพฯ: กลุ่มงานดินด้าน วิทยาศาสตร์ ส่วนวิจัยและพัฒนาด้านวิทยาศาสตร์ สำนักวิจัยและพัฒนากรมชลประทาน.
- สำนักงานเศรษฐกิจการเกษตร. 2560. **สถิติการเกษตรของไทย**. [ระบบออนไลน์]. แหล่งที่มา <http://aginfo.oae.go.th/ewtnews/sweetcorn.html> (16 ตุลาคม 2560).
- Chandan kumar sahu and Pramitee Behera. 2015. A Low Cost Smart Irrigation Control System. pp. 1146-1152. In **Electronics and Communication Systems**. Conference Paper of the 2nd International conference on electronics and communication system. Coimbatore: Institute of Electrical and Electronics Engineers.
- Lee, B. & Shin, J. 1998. Optimal irrigation management system of greenhouse tomato based on stem diameter and transpiration monitoring. **Agricultural Information Technology in Asia and Oceania**, 1998, 87-90.
- Luthra, S. K., Kaledhonkar, M. J., Singh, N. K. & Tyagi, N. K. 1997. Desing and Development of an Auto Irrigation System. **Agricultural Information Technology**, 33, 169-181.





ภาคผนวก ก

โปรแกรมการทำงานของระบบ

Main Program

DHT

```
#include <Wire.h>
```

```
#include <LiquidCrystal_I2C.h>
```

```
#include <DHT.h>
```

```
#include <Time.h>
```

```
#include <DS1307RTC.h>
```

```
#include <SPI.h>
```

```
#include <SD.h>
```

```
// -- Define port --
```

```
#define DHTPIN 6
```

```
#define DHTTYPE DHT22
```

```
#define probe1 A0
```

```
#define probe2 A1
```

```
#define probe3 A2
```

```
#define valve1 2
```

```
#define valve2 3
```

```
#define valve3 4
```

```
#define pump 5
```

```
#define SS 10
```

```
#define delayTime 9000
```

```
// -- Initial devices --
```

```
LiquidCrystal_I2C lcd(0x3f,20,4); // setting LCD address
```

```
DHT dht(DHTPIN, DHTTYPE);
```

```
// -- Other initialization --
```

```
byte degree[8] =
```

```
{
```

```

B00111,
B00101,
B00111,
B00000,
B00000,
B00000,
B00000,
B00000
};

// -- Variables --
tmElements_t tm;
int moistureMax = 15;
int moistureMin = 10;
int moisture[3];
boolean sdcard = false;

void setup() {
  lcd.init();
  lcd.createChar(0,degree);
  lcd.backlight();

  pinMode(valve1, OUTPUT);
  pinMode(valve2, OUTPUT);
  pinMode(valve3, OUTPUT);
  pinMode(pump, OUTPUT);
  pinMode(SS, OUTPUT);

  digitalWrite(pump, LOW);
  digitalWrite(valve1, LOW);
  digitalWrite(valve2, LOW);
  digitalWrite(valve3, LOW);
  digitalWrite(SS, LOW);

  if(SD.begin()){

```



```

    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("SD card OK");
}
else{
    lcd.print("No SD card");
}
}

void loop() {
    showTime();
    showAmbient();
}

void showTime(){
    if (RTC.read(tm)) {
        lcd.setCursor(0,0);
        lcd.print(tm.Day);
        lcd.print('/');
        lcd.print(tm.Month);
        lcd.print('/');
        lcd.print(tmYearToCalendar(tm.Year));
        lcd.print(" ");
        print2digits(tm.Hour);
        lcd.print(':');
        print2digits(tm.Minute);
        Time = tm.Minute;
        Time2 = tm.Minute;
        Time3 = tm.Minute;
    } else {
        if (RTC.chipPresent()) {
            lcd.clear();
            lcd.print("DS1307 is stopped");
        } else {

```

```

        lcd.clear();
        lcd.print("DS1307 read error!");
    }
}

void showAmbient(){
    int temperature = dht.readTemperature();
    int humidity = dht.readHumidity();
    if(isnan(humidity) || isnan(temperature)){
        lcd.clear();
        while(1){
            lcd.setCursor(0,0);
            lcd.print("DHT no connect");
            temperature = dht.readTemperature();
            humidity = dht.readHumidity();
            if(!isnan(humidity) || !isnan(temperature)){
                lcd.clear();
                break;
            }
        }
        lcd.setCursor(1,1);
        lcd.print(t);
        lcd.print(char(0));
        lcd.print("C");
        lcd.setCursor(11,1);
        lcd.print(h);
        lcd.print("%RH");
    }
}

void print2digits(int number) {
    if (number >= 0 && number < 10) {
        lcd.write('0');
    }
    lcd.print(number);
}

```

Preprocessor directive

```
/* DHT library
```

```
MIT license
```

```
written by Adafruit Industries
```

```
*/
```

```
#include "DHT.h"
```

```
#define MIN_INTERVAL 2000
```

```
DHT::DHT(uint8_t pin, uint8_t type, uint8_t count) {
```

```
  _pin = pin;
```

```
  _type = type;
```

```
  #ifdef __AVR
```

```
    _bit = digitalPinToBitMask(pin);
```

```
    _port = digitalPinToPort(pin);
```

```
  #endif
```

```
  _maxcycles = microsecondsToClockCycles(1000); // 1 millisecond timeout for
                                                    // reading pulses from DHT sensor.
```

```
  // Note that count is now ignored as the DHT reading algorithm adjusts itself
```

```
  // based on the speed of the processor.
```

```
}
```

```
void DHT::begin(void) {
```

```
  // set up the pins!
```

```
  pinMode(_pin, INPUT_PULLUP);
```

```
  // Using this value makes sure that millis() - lastreadtime will be
```

```
  // >= MIN_INTERVAL right away. Note that this assignment wraps around,
```

```
  // but so will the subtraction.
```

```
  _lastreadtime = -MIN_INTERVAL;
```

```
  DEBUG_PRINT("Max clock cycles: "); DEBUG_PRINTLN(_maxcycles, DEC);
```

```
}
```

```
//boolean S == Scale. True == Fahrenheit; False == Celcius
```

```
float DHT::readTemperature(bool S, bool force) {
```

```
  float f = NAN;
```

```

if (read(force)) {
    switch (_type) {
    case DHT11:
        f = data[2];
        if(S) {
            f = convertCtoF(f);
        }
        break;
    case DHT22:
    case DHT21:
        f = data[2] & 0x7F;
        f *= 256;
        f += data[3];
        f *= 0.1;
        if (data[2] & 0x80) {
            f *= -1;
        }
        if(S) {
            f = convertCtoF(f);
        }
        break;
    }
    return f;
}

float DHT::convertCtoF(float c) {
    return c * 1.8 + 32;
}

float DHT::convertFtoC(float f) {
    return (f - 32) * 0.55555;
}

float DHT::readHumidity(bool force) {

```

```

float f = NAN;
if (read()) {
    switch (_type) {
    case DHT11:
        f = data[0];
        break;
    case DHT22:
    case DHT21:
        f = data[0];
        f *= 256;
        f += data[1];
        f *= 0.1;
        break;
    }
}
return f;
}

//boolean isFahrenheit: True == Fahrenheit; False == Celcius
float DHT::computeHeatIndex(float temperature, float percentHumidity, bool isFahrenheit) {
    // Using both Rothfusz and Steadman's equations
    // http://www.wpc.ncep.noaa.gov/html/heatindex_equation.shtml
    float hi;

    if (!isFahrenheit)
        temperature = convertCtoF(temperature);

    hi = 0.5 * (temperature + 61.0 + ((temperature - 68.0) * 1.2) + (percentHumidity * 0.094));

    if (hi > 79) {
        hi = -42.379 +
            2.04901523 * temperature +
            10.14333127 * percentHumidity +
            -0.22475541 * temperature * percentHumidity +
            -0.00683783 * pow(temperature, 2) +

```



```

-0.05481717 * pow(percentHumidity, 2) +
0.00122874 * pow(temperature, 2) * percentHumidity +
0.00085282 * temperature * pow(percentHumidity, 2) +
-0.00000199 * pow(temperature, 2) * pow(percentHumidity, 2);

if((percentHumidity < 13) && (temperature >= 80.0) && (temperature <= 112.0))
    hi -= ((13.0 - percentHumidity) * 0.25) * sqrt((17.0 - abs(temperature - 95.0)) * 0.05882);

else if((percentHumidity > 85.0) && (temperature >= 80.0) && (temperature <= 87.0))
    hi += ((percentHumidity - 85.0) * 0.1) * ((87.0 - temperature) * 0.2);
}

return isFahrenheit ? hi : convertFtoC(hi);
}

boolean DHT::read(bool force) {
    // Check if sensor was read less than two seconds ago and return early
    // to use last reading.
    uint32_t currenttime = millis();
    if (!force && ((currenttime - _lastreadtime) < 2000)) {
        return _lastresult; // return last correct measurement
    }
    _lastreadtime = currenttime;

    // Reset 40 bits of received data to zero.
    data[0] = data[1] = data[2] = data[3] = data[4] = 0;

    // Send start signal. See DHT datasheet for full signal diagram:
    //
    http://www.adafruit.com/datasheets/Digital%20humidity%20and%20temperature%20sensor%20AM2302.pdf

    // Go into high impedance state to let pull-up raise data line level and
    // start the reading process.
    digitalWrite(_pin, HIGH);

```

```

delay(250);

// First set data line low for 20 milliseconds.
pinMode(_pin, OUTPUT);
digitalWrite(_pin, LOW);
delay(20);

uint32_t cycles[80];
{
    // Turn off interrupts temporarily because the next sections are timing critical
    // and we don't want any interruptions.
    InterruptLock lock;

    // End the start signal by setting data line high for 40 microseconds.
    digitalWrite(_pin, HIGH);
    delayMicroseconds(40);

    // Now start reading the data line to get the value from the DHT sensor.
    pinMode(_pin, INPUT_PULLUP);
    delayMicroseconds(10); // Delay a bit to let sensor pull data line low.

    // First expect a low signal for ~80 microseconds followed by a high signal
    // for ~80 microseconds again.
    if (expectPulse(LOW) == 0) {
        DEBUG_PRINTLN(F("Timeout waiting for start signal low pulse."));
        _lastresult = false;
        return _lastresult;
    }
    if (expectPulse(HIGH) == 0) {
        DEBUG_PRINTLN(F("Timeout waiting for start signal high pulse."));
        _lastresult = false;
        return _lastresult;
    }
}

// Now read the 40 bits sent by the sensor. Each bit is sent as a 50

```

```

// microsecond low pulse followed by a variable length high pulse. If the
// high pulse is ~28 microseconds then it's a 0 and if it's ~70 microseconds
// then it's a 1. We measure the cycle count of the initial 50us low pulse
// and use that to compare to the cycle count of the high pulse to determine
// if the bit is a 0 (high state cycle count < low state cycle count), or a
// 1 (high state cycle count > low state cycle count). Note that for speed all
// the pulses are read into a array and then examined in a later step.
for (int i=0; i<80; i+=2) {
    cycles[i] = expectPulse(LOW);
    cycles[i+1] = expectPulse(HIGH);
}
} // Timing critical code is now complete.

// Inspect pulses and determine which ones are 0 (high state cycle count < low
// state cycle count), or 1 (high state cycle count > low state cycle count).
for (int i=0; i<40; ++i) {
    uint32_t lowCycles = cycles[2*i];
    uint32_t highCycles = cycles[2*i+1];
    if ((lowCycles == 0) || (highCycles == 0)) {
        DEBUG_PRINTLN(F("Timeout waiting for pulse."));
        _lastresult = false;
        return _lastresult;
    }
    data[i/8] <<= 1;
    // Now compare the low and high cycle times to see if the bit is a 0 or 1.
    if (highCycles > lowCycles) {
        // High cycles are greater than 50us low cycle count, must be a 1.
        data[i/8] |= 1;
    }
    // Else high cycles are less than (or equal to, a weird case) the 50us low
    // cycle count so this must be a zero. Nothing needs to be changed in the
    // stored data.
}

DEBUG_PRINTLN(F("Received:"));

```

```

DEBUG_PRINT(data[0], HEX); DEBUG_PRINT(F(" "));
DEBUG_PRINT(data[1], HEX); DEBUG_PRINT(F(" "));
DEBUG_PRINT(data[2], HEX); DEBUG_PRINT(F(" "));
DEBUG_PRINT(data[3], HEX); DEBUG_PRINT(F(" "));
DEBUG_PRINT(data[4], HEX); DEBUG_PRINT(F(" ==? "));
DEBUG_PRINTLN((data[0] + data[1] + data[2] + data[3]) & 0xFF, HEX);

// Check we read 40 bits and that the checksum matches.
if (data[4] == ((data[0] + data[1] + data[2] + data[3]) & 0xFF)) {
  _lastresult = true;
  return _lastresult;
}
else {
  DEBUG_PRINTLN(F("Checksum failure!"));
  _lastresult = false;
  return _lastresult;
}
}

// Expect the signal line to be at the specified level for a period of time and
// return a count of loop cycles spent at that level (this cycle count can be
// used to compare the relative time of two pulses). If more than a millisecond
// ellapses without the level changing then the call fails with a 0 response.
// This is adapted from Arduino's pulseInLong function (which is only available
// in the very latest IDE versions):
//
https://github.com/arduino/Arduino/blob/master/hardware/arduino/avr/cores/arduino/wiring\_pulse.c
uint32_t DHT::expectPulse(bool level) {
  uint32_t count = 0;
  // On AVR platforms use direct GPIO port access as it's much faster and better
  // for catching pulses that are 10's of microseconds in length:
  #ifdef __AVR
    uint8_t portState = level ? _bit : 0;
    while ((*portInputRegister(_port) & _bit) == portState) {

```

```
if (count++ >= _maxcycles) {  
    return 0; // Exceeded timeout, fail.  
}  
  
}  
  
// Otherwise fall back to using digitalWrite (this seems to be necessary on ESP8266  
// right now, perhaps bugs in direct port access functions?).  
#else  
while (digitalRead(_pin) == level) {  
    if (count++ >= _maxcycles) {  
        return 0; // Exceeded timeout, fail.  
    }  
}  
#endif  
  
return count;  
}
```

```
DHT
/* DHT library

MIT license
written by Adafruit Industries
*/
#ifndef DHT_H
#define DHT_H

#if ARDUINO >= 100
#include "Arduino.h"
#else
#include "WProgram.h"
#endif

// Uncomment to enable printing out nice debug messages.
// #define DHT_DEBUG

// Define where debug output will be printed.
#define DEBUG_PRINTER Serial

// Setup debug printing macros.
#ifdef DHT_DEBUG
#define DEBUG_PRINT(...) { DEBUG_PRINTER.print(__VA_ARGS__); }
#define DEBUG_PRINTLN(...) { DEBUG_PRINTER.println(__VA_ARGS__); }
#else
#define DEBUG_PRINT(...) {}
#define DEBUG_PRINTLN(...) {}
#endif

// Define types of sensors.
#define DHT11 11
#define DHT22 22
#define DHT21 21
```



```
#define AM2301 21
```

```
class DHT {
public:
    DHT(uint8_t pin, uint8_t type, uint8_t count=6);
    void begin(void);
    float readTemperature(bool S=false, bool force=false);
    float convertCtoF(float);
    float convertFtoC(float);
    float computeHeatIndex(float temperature, float percentHumidity, bool isFahrenheit=true);
    float readHumidity(bool force=false);
    boolean read(bool force=false);

private:
    uint8_t data[5];
    uint8_t _pin, _type;
    #ifdef __AVR
        // Use direct GPIO access on an 8-bit AVR so keep track of the port and bitmask
        // for the digital pin connected to the DHT. Other platforms will use digitalRead.
        uint8_t _bit, _port;
    #endif
    uint32_t _lastreadtime, _maxcycles;
    bool _lastresult;

    uint32_t expectPulse(bool level);
};
```

```
class InterruptLock {
public:
    InterruptLock() {
        noInterrupts();
    }
    ~InterruptLock() {
        interrupts();
    }
};
```

```
}
```

```
};
```

```
#endif
```



Real-Time clock

/*

* DS1307RTC.h - library for DS1307 RTC

Copyright (c) Michael Margolis 2009

This library is intended to be uses with Arduino Time.h library functions

The library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

30 Dec 2009 - Initial release

5 Sep 2011 updated for Arduino 1.0

*/

#include <Wire.h>

#include "DS1307RTC.h"

#define DS1307_CTRL_ID 0x68

DS1307RTC::DS1307RTC()

```
{
  Wire.begin();
}
```

```

// PUBLIC FUNCTIONS

time_t DS1307RTC::get() // Aquire data from buffer and convert to time_t
{
    tmElements_t tm;
    if (read(tm) == false) return 0;
    return(makeTime(tm));
}

bool DS1307RTC::set(time_t t)
{
    tmElements_t tm;
    breakTime(t, tm);
    tm.Second |= 0x80; // stop the clock
    write(tm);
    tm.Second &= 0x7f; // start the clock
    write(tm);
}

// Aquire data from the RTC chip in BCD format
bool DS1307RTC::read(tmElements_t &tm)
{
    uint8_t sec;
    Wire.beginTransmission(DS1307_CTRL_ID);
    #if ARDUINO >= 100
        Wire.write((uint8_t)0x00);
    #else
        Wire.send(0x00);
    #endif
    if (Wire.endTransmission() != 0) {
        exists = false;
        return false;
    }
    exists = true;

    // request the 7 data fields (secs, min, hr, dow, date, mth, yr)

```

```

Wire.requestFrom(DS1307_CTRL_ID, tmNbrFields);
if (Wire.available() < tmNbrFields) return false;
#ifdef ARDUINO >= 100
    sec = Wire.read();
    tm.Second = bcd2dec(sec & 0x7f);
    tm.Minute = bcd2dec(Wire.read() );
    tm.Hour = bcd2dec(Wire.read() & 0x3f); // mask assumes 24hr clock
    tm.Wday = bcd2dec(Wire.read() );
    tm.Day = bcd2dec(Wire.read() );
    tm.Month = bcd2dec(Wire.read() );
    tm.Year = y2kYearToTm((bcd2dec(Wire.read())));
#else
    sec = Wire.receive();
    tm.Second = bcd2dec(sec & 0x7f);
    tm.Minute = bcd2dec(Wire.receive() );
    tm.Hour = bcd2dec(Wire.receive() & 0x3f); // mask assumes 24hr clock
    tm.Wday = bcd2dec(Wire.receive() );
    tm.Day = bcd2dec(Wire.receive() );
    tm.Month = bcd2dec(Wire.receive() );
    tm.Year = y2kYearToTm((bcd2dec(Wire.receive())));
#endif
if (sec & 0x80) return false; // clock is halted
return true;
}

bool DS1307RTC::write(tmElements_t &tm)
{
    Wire.beginTransaction(DS1307_CTRL_ID);
#ifdef ARDUINO >= 100
    Wire.write((uint8_t)0x00); // reset register pointer
    Wire.write(dec2bcd(tm.Second));
    Wire.write(dec2bcd(tm.Minute));
    Wire.write(dec2bcd(tm.Hour)); // sets 24 hour format
    Wire.write(dec2bcd(tm.Wday));
    Wire.write(dec2bcd(tm.Day));

```

```

Wire.write(dec2bcd(tm.Month));
Wire.write(dec2bcd(tmYearToY2k(tm.Year)));
#else
Wire.send(0x00); // reset register pointer
Wire.send(dec2bcd(tm.Second)) ;
Wire.send(dec2bcd(tm.Minute));
Wire.send(dec2bcd(tm.Hour)); // sets 24 hour format
Wire.send(dec2bcd(tm.Wday));
Wire.send(dec2bcd(tm.Day));
Wire.send(dec2bcd(tm.Month));
Wire.send(dec2bcd(tmYearToY2k(tm.Year)));
#endif
if (Wire.endTransmission() != 0) {
    exists = false;
    return false;
}
exists = true;
return true;
}

// PRIVATE FUNCTIONS

// Convert Decimal to Binary Coded Decimal (BCD)
uint8_t DS1307RTC::dec2bcd(uint8_t num)
{
    return ((num/10 * 16) + (num % 10));
}

// Convert Binary Coded Decimal (BCD) to Decimal
uint8_t DS1307RTC::bcd2dec(uint8_t num)
{
    return ((num/16 * 10) + (num % 16));
}

bool DS1307RTC::exists = false;

```



```
DS1307RTC RTC = DS1307RTC(); // create an instance for the user
```

Real-Time clock

```
/*
 * DS1307RTC.h - library for DS1307 RTC
 * This library is intended to be uses with Arduino Time.h library functions
 */

#ifndef DS1307RTC_h
#define DS1307RTC_h

#include <Time.h>

// library interface description
class DS1307RTC
{
    // user-accessible "public" interface
public:
    DS1307RTC();
    static time_t get();
    static bool set(time_t t);
    static bool read(tmElements_t &tm);
    static bool write(tmElements_t &tm);
    static bool chipPresent() { return exists; }

private:
    static bool exists;
    static uint8_t dec2bcd(uint8_t num);
    static uint8_t bcd2dec(uint8_t num);
};

#ifndef RTC
#undef RTC // workaround for Arduino Due, which defines "RTC"...
#endif
```

extern DS1307RTC RTC;

#endif



Liquid Crystal Display I2C

// Based on the work by DFRobot

```
#include "LiquidCrystal_I2C.h"
#include <inttypes.h>
#if defined(ARDUINO) && ARDUINO >= 100

#include "Arduino.h"

#define printIIC(args)Wire.write(args)
inline size_t LiquidCrystal_I2C::write(uint8_t value) {
    send(value, Rs);
    return 1;
}

#else
#include "WProgram.h"

#define printIIC(args)Wire.send(args)
inline void LiquidCrystal_I2C::write(uint8_t value) {
    send(value, Rs);
}

#endif
#include "Wire.h"
```

// When the display powers up, it is configured as follows:

//

// 1. Display clear

// 2. Function set:

// DL = 1; 8-bit interface data

// N = 0; 1-line display

// F = 0; 5x8 dot character font

```

// 3. Display on/off control:
//   D = 0; Display off
//   C = 0; Cursor off
//   B = 0; Blinking off
// 4. Entry mode set:
//   I/D = 1; Increment by 1
//   S = 0; No shift
//
// Note, however, that resetting the Arduino doesn't reset the LCD, so we
// can't assume that its in that state when a sketch starts (and the
// LiquidCrystal constructor is called).

LiquidCrystal_I2C::LiquidCrystal_I2C(uint8_t lcd_Addr,uint8_t lcd_cols,uint8_t lcd_rows)
{
  _Addr = lcd_Addr;
  _cols = lcd_cols;
  _rows = lcd_rows;
  _backlightval = LCD_NOBACKLIGHT;
}

void LiquidCrystal_I2C::init(){
  init_priv();
}

void LiquidCrystal_I2C::init_priv()
{
  Wire.begin();
  _displayfunction = LCD_4BITMODE | LCD_1LINE | LCD_5x8DOTS;
  begin(_cols, _rows);
}

void LiquidCrystal_I2C::begin(uint8_t cols, uint8_t lines, uint8_t dotsize) {
  if (lines > 1) {
    _displayfunction |= LCD_2LINE;
  }
}

```

```

_numlines = lines;

// for some 1 line displays you can select a 10 pixel high font
if ((dotsize != 0) && (lines == 1)) {
    _displayfunction |= LCD_5x10DOTS;
}

// SEE PAGE 45/46 FOR INITIALIZATION SPECIFICATION!
// according to datasheet, we need at least 40ms after power rises above 2.7V
// before sending commands. Arduino can turn on way before 4.5V so we'll wait 50
delay(50);

// Now we pull both RS and R/W low to begin commands
expanderWrite(_backlightval); // reset expander and turn backlight off (Bit 8 =1)
delay(1000);

//put the LCD into 4 bit mode
// this is according to the hitachi HD44780 datasheet
// figure 24, pg 46

// we start in 8bit mode, try to set 4 bit mode
write4bits(0x03 << 4);
delayMicroseconds(4500); // wait min 4.1ms

// second try
write4bits(0x03 << 4);
delayMicroseconds(4500); // wait min 4.1ms

// third go!
write4bits(0x03 << 4);
delayMicroseconds(150);

// finally, set to 4-bit interface
write4bits(0x02 << 4);

```

```

// set # lines, font size, etc.
command(LCD_FUNCTIONSET | _displayfunction);

// turn the display on with no cursor or blinking default
_displaycontrol = LCD_DISPLAYON | LCD_CURSOROFF | LCD_BLINKOFF;
display();

// clear it off
clear();

// Initialize to default text direction (for roman languages)
_displaymode = LCD_ENTRYLEFT | LCD_ENTRYSHIFTDECREMENT;

// set the entry mode
command(LCD_ENTRYMODESET | _displaymode);

home();
}

/***** high level commands, for the user! */
void LiquidCrystal_I2C::clear(){
  command(LCD_CLEARDISPLAY); // clear display, set cursor position to zero
  delayMicroseconds(2000); // this command takes a long time!
}

void LiquidCrystal_I2C::home(){
  command(LCD_RETURNHOME); // set cursor position to zero
  delayMicroseconds(2000); // this command takes a long time!
}

void LiquidCrystal_I2C::setCursor(uint8_t col, uint8_t row){
  int row_offsets[] = { 0x00, 0x40, 0x14, 0x54 };
  if ( row > _numlines ) {
    row = _numlines-1; // we count rows starting w/0
  }
}

```



```

    }

    command(LCD_SETDRAMADDR | (col + row_offsets[row]));
}

// Turn the display on/off (quickly)
void LiquidCrystal_I2C::noDisplay() {
    _displaycontrol &= ~LCD_DISPLAYON;
    command(LCD_DISPLAYCONTROL | _displaycontrol);
}

void LiquidCrystal_I2C::display() {
    _displaycontrol |= LCD_DISPLAYON;
    command(LCD_DISPLAYCONTROL | _displaycontrol);
}

// Turns the underline cursor on/off
void LiquidCrystal_I2C::noCursor() {
    _displaycontrol &= ~LCD_CURSORON;
    command(LCD_DISPLAYCONTROL | _displaycontrol);
}

void LiquidCrystal_I2C::cursor() {
    _displaycontrol |= LCD_CURSORON;
    command(LCD_DISPLAYCONTROL | _displaycontrol);
}

// Turn on and off the blinking cursor
void LiquidCrystal_I2C::noBlink() {
    _displaycontrol &= ~LCD_BLINKON;
    command(LCD_DISPLAYCONTROL | _displaycontrol);
}

void LiquidCrystal_I2C::blink() {
    _displaycontrol |= LCD_BLINKON;
    command(LCD_DISPLAYCONTROL | _displaycontrol);
}

// These commands scroll the display without changing the RAM

```

```

void LiquidCrystal_I2C::scrollDisplayLeft(void) {
    command(LCD_CURSORSHIFT | LCD_DISPLAYMOVE | LCD_MOVELEFT);
}

void LiquidCrystal_I2C::scrollDisplayRight(void) {
    command(LCD_CURSORSHIFT | LCD_DISPLAYMOVE | LCD_MOVERIGHT);
}

// This is for text that flows Left to Right
void LiquidCrystal_I2C::leftToRight(void) {
    _displaymode |= LCD_ENTRYLEFT;
    command(LCD_ENTRYMODESET | _displaymode);
}

// This is for text that flows Right to Left
void LiquidCrystal_I2C::rightToLeft(void) {
    _displaymode &= ~LCD_ENTRYLEFT;
    command(LCD_ENTRYMODESET | _displaymode);
}

// This will 'right justify' text from the cursor
void LiquidCrystal_I2C::autoscroll(void) {
    _displaymode |= LCD_ENTRYSHIFTINCREMENT;
    command(LCD_ENTRYMODESET | _displaymode);
}

// This will 'left justify' text from the cursor
void LiquidCrystal_I2C::noAutoscroll(void) {
    _displaymode &= ~LCD_ENTRYSHIFTINCREMENT;
    command(LCD_ENTRYMODESET | _displaymode);
}

// Allows us to fill the first 8 CGRAM locations
// with custom characters
void LiquidCrystal_I2C::createChar(uint8_t location, uint8_t charmap[]) {
    location &= 0x7; // we only have 8 locations 0-7

```

```

command(LCD_SETCGRAMADDR | (location << 3));
for (int i=0; i<8; i++) {
    write(charmap[i]);
}
}

// Turn the (optional) backlight off/on
void LiquidCrystal_I2C::noBacklight(void) {
    _backlightval=LCD_NOBACKLIGHT;
    expanderWrite(0);
}

void LiquidCrystal_I2C::backlight(void) {
    _backlightval=LCD_BACKLIGHT;
    expanderWrite(0);
}

/***** mid level commands, for sending data/cmds */

inline void LiquidCrystal_I2C::command(uint8_t value) {
    send(value, 0);
}

/***** low level data pushing commands *****/

// write either command or data
void LiquidCrystal_I2C::send(uint8_t value, uint8_t mode) {
    uint8_t highnib=value&0xf0;
    uint8_t lownib=(value<<4)&0xf0;
    write4bits((highnib)|mode);
    write4bits((lownib)|mode);
}

void LiquidCrystal_I2C::write4bits(uint8_t value) {
    expanderWrite(value);
    pulseEnable(value);
}

```

```

}

void LiquidCrystal_I2C::expanderWrite(uint8_t _data){
    Wire.beginTransmission( _Addr);
    printIIC((int)(_data) | _backlightval);
    Wire.endTransmission();
}

void LiquidCrystal_I2C::pulseEnable(uint8_t _data){
    expanderWrite( _data | En); // En high
    delayMicroseconds(1); // enable pulse must be >450ns

    expanderWrite( _data & ~En); // En low
    delayMicroseconds(50); // commands need > 37us to settle
}

// Alias functions

void LiquidCrystal_I2C::cursor_on(){
    cursor();
}

void LiquidCrystal_I2C::cursor_off(){
    noCursor();
}

void LiquidCrystal_I2C::blink_on(){
    blink();
}

void LiquidCrystal_I2C::blink_off(){
    noBlink();
}

void LiquidCrystal_I2C::load_custom_character(uint8_t char_num, uint8_t *rows){
    createChar(char_num, rows);
}

```

```

}

void LiquidCrystal_I2C::setBacklight(uint8_t new_val){
    if(new_val){
        backlight(); // turn backlight on
    }else{
        noBacklight(); // turn backlight off
    }
}

void LiquidCrystal_I2C::printstr(const char c[]){
    //This function is not identical to the function used for "real" I2C displays
    //it's here so the user sketch doesn't have to be changed
    print(c);
}

// unsupported API functions
void LiquidCrystal_I2C::off(){}
void LiquidCrystal_I2C::on(){}
void LiquidCrystal_I2C::setDelay (int cmdDelay,int charDelay) {}
uint8_t LiquidCrystal_I2C::status(){return 0;}
uint8_t LiquidCrystal_I2C::keypad (){return 0;}
uint8_t LiquidCrystal_I2C::init_bargraph(uint8_t graphtype){return 0;}
void LiquidCrystal_I2C::draw_horizontal_graph(uint8_t row, uint8_t column, uint8_t len, uint8_t
pixel_col_end){}
void LiquidCrystal_I2C::draw_vertical_graph(uint8_t row, uint8_t column, uint8_t len, uint8_t
pixel_row_end){}
void LiquidCrystal_I2C::setContrast(uint8_t new_val){}

```

Liquid Crystal display

//YWROBOT

#ifndef LiquidCrystal_I2C_h

#define LiquidCrystal_I2C_h

#include <inttypes.h>

#include "Print.h"

#include <Wire.h>

// commands

#define LCD_CLEARDISPLAY 0x01

#define LCD_RETURNHOME 0x02

#define LCD_ENTRYMODESET 0x04

#define LCD_DISPLAYCONTROL 0x08

#define LCD_CURSORSHIFT 0x10

#define LCD_FUNCTIONSET 0x20

#define LCD_SETGRAMADDR 0x40

#define LCD_SETDDRAMADDR 0x80

// flags for display entry mode

#define LCD_ENTRYRIGHT 0x00

#define LCD_ENTRYLEFT 0x02

#define LCD_ENTRYSHIFTINCREMENT 0x01

#define LCD_ENTRYSHIFTDECREMENT 0x00

// flags for display on/off control

#define LCD_DISPLAYON 0x04

#define LCD_DISPLAYOFF 0x00

#define LCD_CURSORON 0x02

#define LCD_CURSOROFF 0x00

#define LCD_BLINKON 0x01

#define LCD_BLINKOFF 0x00

// flags for display/cursor shift

#define LCD_DISPLAYMOVE 0x08


```

#define LCD_CURSORMOVE 0x00
#define LCD_MOVERIGHT 0x04
#define LCD_MOVELEFT 0x00

// flags for function set
#define LCD_8BITMODE 0x10
#define LCD_4BITMODE 0x00
#define LCD_2LINE 0x08
#define LCD_1LINE 0x00
#define LCD_5x10DOTS 0x04
#define LCD_5x8DOTS 0x00

// flags for backlight control
#define LCD_BACKLIGHT 0x08
#define LCD_NOBACKLIGHT 0x00

#define En B00000100 // Enable bit
#define Rw B00000010 // Read/Write bit
#define Rs B00000001 // Register select bit

class LiquidCrystal_I2C : public Print {
public:
    LiquidCrystal_I2C(uint8_t lcd_Addr,uint8_t lcd_cols,uint8_t lcd_rows);
    void begin(uint8_t cols, uint8_t rows, uint8_t charsize = LCD_5x8DOTS );
    void clear();
    void home();
    void noDisplay();
    void display();
    void noBlink();
    void blink();
    void noCursor();
    void cursor();
    void scrollDisplayLeft();
    void scrollDisplayRight();
    void printLeft();

```

```

void printRight();
void leftToRight();
void rightToLeft();
void shiftIncrement();
void shiftDecrement();
void noBacklight();
void backlight();
void autoscroll();
void noAutoscroll();
void createChar(uint8_t, uint8_t[]);
void setCursor(uint8_t, uint8_t);
#if defined(ARDUINO) && ARDUINO >= 100
    virtual size_t write(uint8_t);
#else
    virtual void write(uint8_t);
#endif
void command(uint8_t);
void init();

////compatibility API function aliases
void blink_on();           // alias for blink()
void blink_off();          // alias for noBlink()
void cursor_on();           // alias for cursor()
void cursor_off();         // alias for noCursor()
void setBacklight(uint8_t new_val); // alias for backlight() and nobacklight()
void load_custom_character(uint8_t char_num, uint8_t *rows); // alias for createChar()
void printstr(const char[]);

////Unsupported API functions (not implemented in this library)
uint8_t status();
void setContrast(uint8_t new_val);
uint8_t keypad();
void setDelay(int,int);
void on();
void off();

```

```
uint8_t init_bargraph(uint8_t graphtype);  
void draw_horizontal_graph(uint8_t row, uint8_t column, uint8_t len, uint8_t pixel_col_end);  
void draw_vertical_graph(uint8_t row, uint8_t column, uint8_t len, uint8_t pixel_col_end);
```

```
private:
```

```
void init_priv();  
void send(uint8_t, uint8_t);  
void write4bits(uint8_t);  
void expanderWrite(uint8_t);  
void pulseEnable(uint8_t);  
uint8_t _Addr;  
uint8_t _displayfunction;  
uint8_t _displaycontrol;  
uint8_t _displaymode;  
uint8_t _numlines;  
uint8_t _cols;  
uint8_t _rows;  
uint8_t _backlightval;  
};
```

```
#endif
```



ภาคผนวก ข

ประวัติผู้วิจัย

ประวัติผู้วิจัย

ชื่อ-นามสกุล	นายเอกนรินทร์ ทาไสย
เกิดเมื่อ	29 มิถุนายน 2527
ประวัติการศึกษา	พ.ศ. 2552 ปริญญาตรีคณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ประวัติการทำงาน	พ.ศ. 2554-2557 Telecom Engineer บริษัท จูน (ประเทศไทย) จำกัด