ระบบเก็บข้อมูลการใช้งานและติดตามรถยนต์บรรทุก

THE TRUCK USING AND TRACKING STORAGE SYSTEM

โดย

นายธีร์ธวัช จิระเสวี

นายพนธฤต เชี่ยวพานิชย์

นายพัฒนะ คัยนันท์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2556

ระบบเก็บข้อมูลการใช้งานและติดตามรถยนต์บรรทุก
THE TRUCK USING AND TRACKING STORAGE SYSTEM

โดย

นายธีร์ธวัช	จิระเสวี
นายพนธฤต	เชี่ยวพานิชย์
นายพัฒนะ	คัยนันท์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมโทรคมนาคม
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2556

ระบบเก็บข้อมูลการใช้งานและติดตามรถยนต์บรรทุก
THE TRUCK USING AND TRACKING STORAGE SYSTEM

โดย

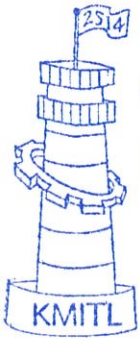
นายธีร์ธวัช	จิระเสวี	53010747
นายพนธฤต	เชี่ยวพานิชย์	53011047
นายพัฒนะ	คัยนันท์	53011100

อาจารย์ที่ปรึกษา

รศ.ดร. ปราโมทย์ วาดเขียน

รศ.ดร. จีรสุดา โกษียาภรณ์

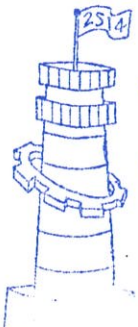
ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมโทรคมนาคม
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2556



ผ่านการตรวจรูปเล่มแล้ว

(.....)
อาจารย์ที่ปรึกษา
13/3/57

วิศวกรรมโทรคมนาคม
Telecommunications Engineering



ผ่านการตรวจชิ้นงานแล้ว

(.....)
กรรมการผู้ตรวจชิ้นงาน
14/มี.ค.57

ปริญญานิพนธ์ปีการศึกษา 2556
สาขาวิชาวิศวกรรมโทรคมนาคม
คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
เรื่อง ระบบเก็บข้อมูลการใช้งานและติดตามรถยนต์บรรทุก
THE TRUCK USING AND TRACKING STORAGE SYSTEM

ผู้จัดทำ

1.	นายธีร์ธวัช	จิระเสวี	53010747
2.	นายพนธฤต	เชี่ยวพานิชย์	53011047
3.	นายพัฒนะ	คัยนันท์	53011100

..... อาจารย์ที่ปรึกษา
รศ.ดร. ปราโมทย์ วาดเขียน

..... อาจารย์ที่ปรึกษา
รศ.ดร. จีรสุดา โกษียาภรณ์

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้ด้วยดี เนื่องจากได้รับความอนุเคราะห์อย่างดียิ่งจากท่านอาจารย์ที่ปรึกษา คือ รศ.ดร. ปราโมทย์ วาดเขียน และ รศ.ดร. จีรสุดา โกษียาภรณ์ ที่ให้คำแนะนำ คำสั่งสอน ให้ความรู้ ความเข้าใจตลอดระยะเวลาในการทำปริญญานิพนธ์ฉบับนี้ ขอขอบพระคุณท่านในความห่วงใยและความหวังดีที่ให้แก่ผู้จัดทำเป็นอย่างยิ่ง

ขอขอบคุณคณาจารย์ ประจำสาขาวิชาวิศวกรรมโทรคมนาคม
คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังทุกท่าน ที่ได้อบรมสั่งสอน และประสิทธิ์ประสาทวิชาความรู้ให้แก่ผู้จัดทำ

ขอขอบคุณบริษัท วีอาร์พี อินเตอร์ โลจิสติกส์ จำกัด ที่ให้การสนับสนุนอุปกรณ์ในการทดลองรวมทั้งให้คำแนะนำเกี่ยวกับการติดตั้ง

ขอขอบคุณเพื่อนๆ และพี่ๆ นักศึกษาสาขาวิชาวิศวกรรมโทรคมนาคมและผู้ที่มีส่วนเกี่ยวข้องทุกท่านที่คอยช่วยเหลือและให้กำลังใจแก่พวกเราเสมอมา จนกระทั่งปริญญานิพนธ์ฉบับนี้ลุล่วงไปได้ด้วยดี ความดีอันเกิดจากการทำปริญญานิพนธ์นี้ ผู้จัดทำขอมอบแด่บิดา มารดา ครูอาจารย์ เพื่อนนักศึกษาและผู้มีพระคุณทุกท่าน

นายธีร์ธวัช จิระเสวี
นายพนธฤต เชี่ยวพานิชย์
นายพัฒนะ คัยนันท์

ผู้จัดทำ

ระบบเก็บข้อมูลการใช้งานและติดตามรถยนต์บรรทุก
THE TRUCK USING AND TRACKING STORAGE SYSTEM

โดย นายธีร์ธวัช จิระเสวี 53010747
นายพนธฤต เชี่ยวพานิชย์ 53011047
นายพัฒนะ คัยนันท์ 53011100

อาจารย์ที่ปรึกษา รศ.ดร. ปราโมทย์ วาดเขียน
อาจารย์ที่ปรึกษา รศ.ดร. จีรสุดา โกษียาภรณ์

บทคัดย่อ

ปริญญานิพนธ์ฉบับนี้เป็นการออกแบบและสร้างระบบเก็บข้อมูลการใช้งานรถยนต์บรรทุก ซึ่งประกอบด้วย ชื่อผู้ขับ ความเร็ว พิกัด การใช้สัญญาณไฟ และการเบรก เพื่อเป็นการช่วยติดตามและตรวจสอบพฤติกรรมของผู้ขับขี่และยังสามารถนำข้อมูลดังกล่าวมาวิเคราะห์หาสาเหตุในกรณีที่รถเกิดอุบัติเหตุ โดยระบบจะประกอบไปด้วย 1.การตรวจสอบข้อมูลผู้ขับก่อนทำการสตาร์ทเครื่องยนต์โดยใช้ระบบอาร์เอฟไอดีหรือแป้นกด 2.การวัดการเบรกและการใช้สัญญาณไฟ (ไฟหน้า ไฟเลี้ยวซ้าย-ขวา และไฟเบรก) ด้วยเซนเซอร์ โดยระบบจะใช้ไมโครคอนโทรลเลอร์ในการส่งต่อข้อมูลที่ได้ไปยังโทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์ผ่านทางบลูธูท โดยโทรศัพท์มือถือจะมีแอพพลิเคชั่นการอ่านค่าตำแหน่งของรถด้วยจีพีเอส ที่จะทำการส่งข้อมูลทั้งหมดไปเก็บยังฐานข้อมูลกลางเพื่อแสดงผลตำแหน่งของรถผ่านทางหน้าเว็บเพจด้วยกูเกิ้ลแมพ

Abstract

This thesis is to design and build the system for tracking and storing data of truck usage. The data which are driver's name, velocity, car's position, light signal and brake will be used for tracking and checking driver's behavior and also for analyzing in the case of the accident. The system is composed of 1. Driver identification part (RFID or keypad) and 2. Sensors part (light signal and brake). The sensor data will be sent to the android mobile phone via Bluetooth by the microcontroller. The android application on the mobile phone will send the sensors data to the central database. The car's position later will be displayed on the web page using the Google Maps.

สารบัญ

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญรูป

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

สารบัญตาราง

บทที่ 1

บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

เนื่องจากการคมนาคมขนส่งในปัจจุบันเป็นสิ่งที่สำคัญจึงต้องเน้นในเรื่องความสะดวกรวดเร็วและความปลอดภัย โดยการคมนาคมขนส่งส่วนใหญ่ในภาคธุรกิจก็ยังใช้รถยนต์บรรทุกเป็นพาหนะในการขนส่ง ซึ่งสามารถเกิดปัญหามากมายจากการใช้รถยนต์บรรทุกเช่น อุบัติเหตุ การโจรกรรม และการขนส่งล่าช้า เป็นต้น โดยปัญหาเหล่านั้นอาจเกิดจากความประมาทของผู้ขับขี่เองหรือเกิดจากผู้อื่น โดยเจ้าของกิจการหรือพนักงานของรัฐซึ่งเป็นผู้ที่ไม่ได้อยู่ในเหตุการณ์ไม่สามารถรับรู้ได้

ปริญญานิพนธ์ฉบับนี้จึงเน้นเรื่องความปลอดภัยในการคมนาคมขนส่งโดยเป็นการหาสาเหตุของการเกิดอุบัติเหตุที่เกิดขึ้น คณะผู้จัดทำได้ออกแบบระบบซึ่งสามารถเก็บข้อมูลการใช้งานและติดตามรถยนต์บรรทุกโดยเป็นการออกแบบระบบที่ใช้เก็บข้อมูลต่างๆ ของรถยนต์ซึ่งประกอบด้วยชุดอุปกรณ์อาร์เอฟไอดีและแป้นกดในการตรวจสอบข้อมูลผู้ขับก่อนการติดเครื่องรถยนต์ ชุดอุปกรณ์ตรวจจับการทำงานของอุปกรณ์ภายในรถยนต์ เช่น การทำงานของไฟหน้า ไฟท้าย ไฟเลี้ยว ไฟถอยและการเบรก โดยเก็บข้อมูลผ่านไมโครคอนโทรลเลอร์ และใช้แอพพลิเคชั่นในโทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์ในการเก็บค่าพิกัดและหาค่าความเร็วด้วยระบบจีพีเอส โดยสามารถส่งข้อมูลดังกล่าวไปเก็บไว้ในฐานข้อมูลเพื่อที่จะสามารถใช้แสดงผลผ่านทางหน้าเวปเพจได้

1.2 วัตถุประสงค์

1.2.1 เพื่อใช้ในการเก็บข้อมูลการใช้งานรถยนต์บรรทุกของผู้ขับ

1.2.2 เพื่อใช้ติดตามพิกัดของรถยนต์บรรทุก

1.2.3 เพื่อเก็บข้อมูลที่จำเป็นในการใช้หาสาเหตุการเกิดอุบัติเหตุ

1.3 ขอบเขตของปริญญานิพนธ์

1.3.1 สามารถใช้ระบบอาร์เอฟไอดีและแป้นกด ในการตรวจสอบข้อมูลผู้ขับจากฐานข้อมูลและรับส่งข้อมูลไปเก็บไว้ในฐานข้อมูล

1.3.2 สามารถสร้างเซนเซอร์ที่ใช้ในการเก็บค่าข้อมูลคือ การทำงานของไฟหน้า ไฟท้าย ไฟเลี้ยว ไฟถอยและการเบรก

1.3.3 สามารถพัฒนาแอพพลิเคชั่นในการวัดความเร็วและทิศทางด้วยระบบจีพีเอส รวมทั้งเก็บข้อมูลที่ได้จากเซนเซอร์และส่งไปยังฐานข้อมูล

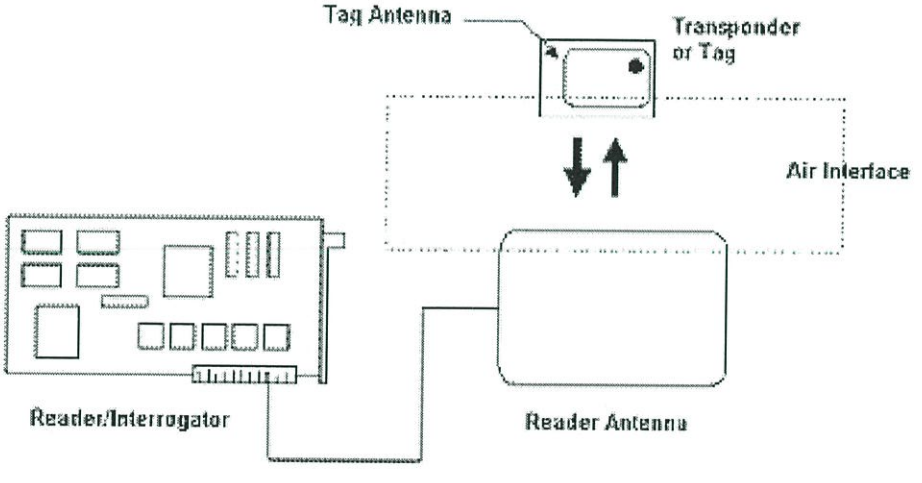
1.3.4 สามารถแสดงผลค่าข้อมูลบนหน้าเวปเพจได้

บทที่ 2

ทฤษฎีและหลักการที่เกี่ยวข้อง

2.1 อาร์เอฟไอดี (RFID)

องค์ประกอบในระบบอาร์เอฟไอดี จะมีหลักๆ อยู่ 2 ส่วนด้วยกัน คือ ส่วนแรกคือฉลากหรือป้ายขนาดเล็กที่จะถูกผนึกอยู่กับวัตถุที่เราสนใจ โดยฉลากนี้จะทำการบันทึกข้อมูลเกี่ยวกับวัตถุชิ้นนั้นๆ เอาไว้ ฉลากดังกล่าวมีชื่อเรียกว่า ทรานสพอนเดอร์ (Transponder, Transmitter & Responder) หรือที่เรียกกันโดยทั่ว ไปว่า "แท็กส์" (Tag) ส่วนที่สองก็คืออุปกรณ์สำหรับอ่านหรือเขียนข้อมูลภายในแท็กส์ มีชื่อเรียกว่า ทรานสซิฟเวอร์ (Transceiver, Transmitter & Receiver) หรือที่เรียกกันโดยทั่ว ๆ ไปว่า "เครื่องอ่าน" (Reader) ทั้งสองส่วนจะสื่อสารกันโดยอาศัยช่องความถี่วิทยุ สัญญาณนี้ผ่านได้ทั้งโลหะ และอโลหะแต่ไม่สามารถติดต่อกับเครื่องอ่านให้อ่านได้โดยตรง เมื่อเครื่องอ่านส่งข้อมูลผ่านความถี่วิทยุ แสดงถึงความต้องการข้อมูลที่ถูกระบุไว้จากป้าย ป้ายจะตอบข้อมูลกลับและเครื่องอ่านจะส่งข้อมูลต่อไปยังส่วนประมวลผลหลักของคอมพิวเตอร์ โดยเครื่องอ่านจะติดต่อสื่อสารกับคอมพิวเตอร์โดยผ่านเครือข่าย LAN (Local Area Network) หรือส่งผ่านทางความถี่วิทยุจากทั้งอุปกรณ์มีสาย และอุปกรณ์ไร้สาย โดยจะมีลักษณะดังรูปที่ 2.1

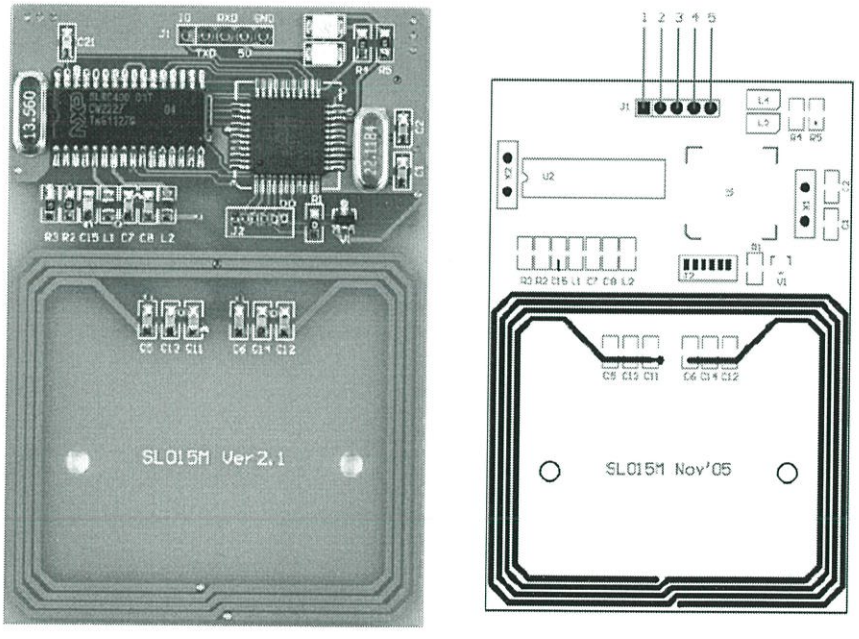


รูปที่ 2.1 องค์ประกอบของระบบอาร์เอฟไอดี

2.1.1 อาร์เอฟไอดีโมดูลรุ่น SL015M

ตารางที่ 2.1 ชนิดและลักษณะของขาเชื่อมต่อของอาร์เอฟไอดีโมดูลรุ่น SL015M

PIN	SYMBOL	TYPE	DESCRIPTION
1	TagSta	Output	Tag detect signal Low level indicating tag in detection range High level indicating tag out
2	TXD	Output	Serial output port
3	RXD	Input	Serial input port
4	VCC	PWR	Power Supply
5	GND	PWR	Ground



รูปที่ 2.2 อาร์เอฟไอดีโมดูลรุ่น SL015M

โดยในการสื่อสารกับอาร์เอฟไอดีโมดูลนั้นสามารถกำหนดค่าอัตราบอด(Baud Rate) ได้จากการใส่หรือไม่ใส่ตัวต้านทาน R6 และ R7 ซึ่งมีค่า 0 โอห์มดังตารางที่ 2.2

ตารางที่ 2.2 การกำหนดค่าอัตราบอดของอาร์เอฟไอดีโมดูล

	R6	R7	Baud rate (bps)
Assembled	No	No	9,600
	Yes	No	19,200
	No	Yes	57,600
	Yes	Yes	115,200

2.1.2 โปรโตคอลที่ใช้ในการสื่อสาร

1) การตั้งค่าการสื่อสาร

โปรโตคอลที่ใช้ในการสื่อสารเป็นโปรโตคอลจัดการแบบนับจำนวนไบต์ (byte oriented) ซึ่งทั้งการส่งและรับข้อมูลจะอยู่ในรูปแบบของเลขฐาน 16 โดยมีพารามิเตอร์ที่สำคัญดังนี้

- ค่าอัตราบอดตั้งแต่ 9600 – 115200 บิตต่อวินาที
- จำนวนข้อมูล 8 บิต
- บิตหยุด 1 บิต

2) แพกเก็จคำสั่ง และข้อมูล

ในการสื่อสารระหว่างตัวอ่านและแท็กส์นั้น จะต้องทำการป้อนคำสั่งขอค่าแท็กส์ไปยังตัวอ่านก่อนซึ่งจะมีลักษณะข้อมูลเป็นแพกเก็จ ดังรูปที่ 2.3 และค่าข้อมูลจากแท็กส์ที่ตอบกลับมานั้นจะมีลักษณะเป็นแพกเก็จเช่นกันดังรูปที่ 2.4

Get tag information

0xBA	Len	0x31	Checksum

รูปที่ 2.3 แพกเก็จคำสั่งขอค่าแท็กส์

Response:

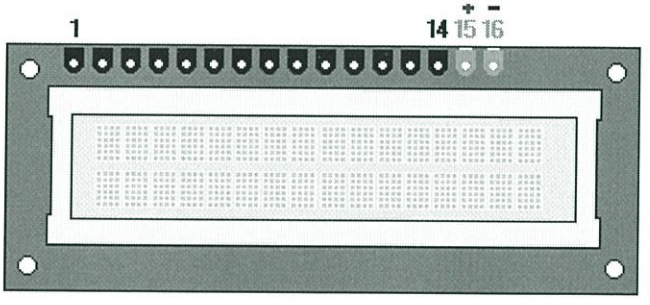
0xBD	Len	0x31	Status	UID	DSFID	AFI	Type	Checksum

Status: 0x00: Operation succeed
0x01: No tag
0x04: Read fail
0xF0: Checksum error
UID: The Unique Identifier of card, 8 bytes
AFI: The Application Family Identifier, 1byte
DSFID: The Data Storage Format Identifier, 1byte
Type: 0x31: Tag_it HF I
0x32: I.CODE SLI

รูปที่ 2.4 แพกเก็จข้อมูลของแท็กส์

2.2 แอลซีดี (LCD: Liquid Crystal Display)

ใช้ทำจอภาพ มีลักษณะเป็นของเหลวใสเหมือนแก้วเจียรนัย อัดอยู่ระหว่างเนื้อแก้วสองชั้น เมื่อถูกกระตุ้นด้วยแสงจากภายนอก จะสะท้อนแสงออกมาเป็นมุมต่าง ๆ ใช้เป็นตัวแสดงอักขระและภาพ คอมพิวเตอร์ขนาดวางตักนิยมใช้จอภาพชนิดนี้ จอภาพนี้แม้จะใช้กำลังไฟฟ้าน้อยมาก แต่จะชัดเจนดี นิยมใช้กับเครื่องคำนวณที่มีขนาดเล็กมาก ๆ ด้วย โดยจะมีลักษณะดังรูปที่ 2.5 และมีการทำงานของขาต่างๆ ดังตารางที่ 2.3



LCD 16x2 Line

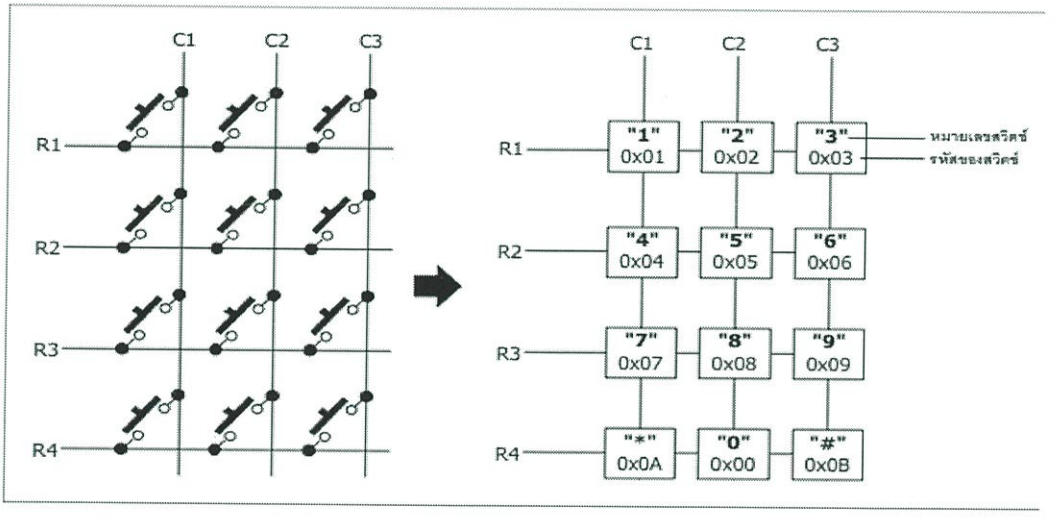
รูปที่ 2.5 ลักษณะของจอแอลซีดี

ตารางที่ 2.3 การทำงานของขาต่างๆ ของ LCD 16x2 Line

Pin No.	Symbol	Description	Level	Function	
1	VSS	Ground	-	0V	Ground
2	VDD	Power Supply	-	+5V	ต่อกับแรงดันไฟเลี้ยง +5V
3	V0	LCD Contr	-	-	ต่อกับแรงดันเพื่อปรับความเข้มของการแสดงผล
4	RS	Register Select	H/L	RS = 0 หมายถึงต้องการติดต่อกับรีจิสเตอร์คำสั่ง (Instruction Register) RS = 1 หมายถึงต้องการติดต่อกับรีจิสเตอร์ข้อมูล (Data Register)	
5	R/W	Read/Write	H/L	R/W = 0 หมายถึงต้องการเขียนข้อมูลไปยัง LCD โมดูล R/W = 1 หมายถึงต้องการอ่านข้อมูลจาก LCD โมดูล	
6	E	Enable	H, H->L	Enable Signal	
7 - 14	DB0-DB7	Data Bus	H/L	Data Bus Line	
15	A	Back Light A	-	Back Light +5V (สำหรับรุ่นที่มี Back Light)	
16	K	Back Light K	-	Back Light 0V (สำหรับรุ่นที่มี Back Light)	

2.3 แป้นกด (Keypad)

สวิตช์เมตริกซ์ (matrix switch) หรือแป้นกด (keypad) เป็นอุปกรณ์ที่มีไว้ป้อนอินพุตข้อมูลให้กับงานทางด้านไมโครคอนโทรลเลอร์ นอกเหนือจากสวิตช์กดติดปล่อยดับแบบธรรมดา (push button switch) โดยเฉพาะกับงานที่ต้องมีการป้อนข้อมูลทั้งตัวอักษร และตัวเลข และมีสวิตช์จำนวนมากแล้วสวิตช์เมตริกซ์จะเป็นตัวที่ถูกเลือกใช้งานเสมอสวิตช์ในรูปแบบเมตริกซ์ที่เห็นได้ในชีวิตประจำวัน เช่น แป้นกดตัวเลขของระบบโทรศัพท์ เป็นต้น โดยการต่อใช้งานแป้นกดเป็นการนำเอาสวิตช์ธรรมดามาต่อแบบเมตริกซ์ คือขาด้านหนึ่งจะต่อในแนวหลัก (column) และขาอีกด้านหนึ่งต่ออยู่ในแนวแถว (row) แสดงดังรูปที่ 2.6 และมีลักษณะของสวิตช์เมตริกซ์ หรือแป้นกดดังรูปที่ 2.7



รูปที่ 2.6 แสดงการกำหนดค่ารหัสของสวิตช์เมตริกซ์หรือ Keypad ขนาด 4x3

รูปที่ 2.7 สวิตช์เมตริกซ์หรือแป้นกด

โดยการแสดงค่ารหัสสวิตช์เมตริกซ์หรือแป้นกดสามารถแสดงได้ดังรูปที่ 2.8

หลัก	ค่าประจำแถว
col 1	0
col 2	1
col 3	2

แถว	ค่าประจำแถว
R1 (row 1)	0x00
R2 (row 2)	0x03
R3 (row 2)	0x06
R4 (row 4)	0x09

รูปที่ 2.8 ค่ารหัสสวิตช์เมตริกซ์

2.3.1 กำหนดรหัสประจำตำแหน่งของสวิตช์เมตริกซ์ หรือแป้นกด

หลักในการอ่านค่าจากแป้นกดนี้คือจะต้องกำหนดรหัสประจำตำแหน่งของสวิตช์แต่ละตัวไว้ ไม่ให้ซ้ำกัน ดังนั้นเมื่อสวิตช์ตัวใดถูกกดก็จะได้ค่ารหัสของสวิตช์ตัวดังกล่าวออกมาโดยจะกำหนดค่าคงที่ให้กับสวิตช์แต่ละตัวไว้ ดังนี้

- สวิตช์ตำแหน่ง R1 (หมายถึงแถวที่ 1), C1 (หมายถึงหลักที่ 1) มีค่า 0x01
- สวิตช์ตำแหน่ง R1 (หมายถึงแถวที่ 1), C2 (หมายถึงหลักที่ 2) มีค่า 0x02
- สวิตช์ตำแหน่ง R1 (หมายถึงแถวที่ 1), C3 (หมายถึงหลักที่ 2) มีค่า 0x03
- สวิตช์ตำแหน่ง R2 (หมายถึงแถวที่ 2), C1 (หมายถึงหลักที่ 1) มีค่า 0x04
- สวิตช์ตำแหน่ง R2 (หมายถึงแถวที่ 2), C2 (หมายถึงหลักที่ 2) มีค่า 0x05
- สวิตช์ตำแหน่ง R2 (หมายถึงแถวที่ 2), C3 (หมายถึงหลักที่ 2) มีค่า 0x06
- สวิตช์ตำแหน่ง R3 (หมายถึงแถวที่ 3), C1 (หมายถึงหลักที่ 1) มีค่า 0x07
- สวิตช์ตำแหน่ง R3 (หมายถึงแถวที่ 3), C2 (หมายถึงหลักที่ 2) มีค่า 0x08
- สวิตช์ตำแหน่ง R3 (หมายถึงแถวที่ 3), C3 (หมายถึงหลักที่ 2) มีค่า 0x09
- สวิตช์ตำแหน่ง R4 (หมายถึงแถวที่ 4), C1 (หมายถึงหลักที่ 1) มีค่า 0x10
- สวิตช์ตำแหน่ง R4 (หมายถึงแถวที่ 4), C2 (หมายถึงหลักที่ 2) มีค่า 0x00
- สวิตช์ตำแหน่ง R4 (หมายถึงแถวที่ 4), C3 (หมายถึงหลักที่ 2) มีค่า 0x11

จากค่าตัวเลขแถวและหลักสามารถนำเอาเขียนเป็นรหัสของสวิตช์แต่ละตัวในรูปที่ 2.6 ได้ เช่นถ้ากดในแถว R3 ตัดกับหลัก C2 ก็จะได้ผลลัพธ์เป็นเลข ‘8’

2.4 ไมโครคอนโทรลเลอร์ตระกูลเอวีอาร์ (AVR Microcontroller)

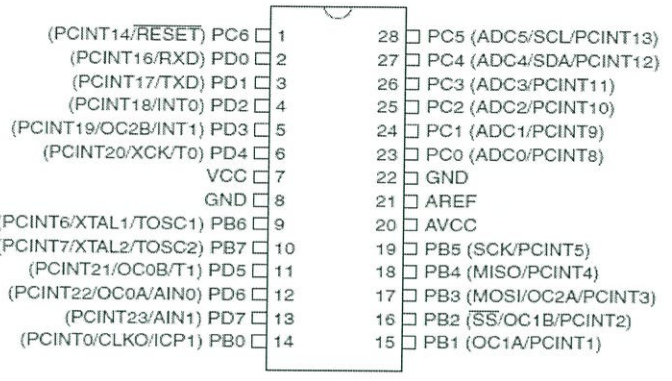
โครงสร้างโดยทั่วไปของไมโครคอนโทรลเลอร์ (Microcontroller) จะมีส่วนประกอบที่สำคัญตามตารางที่ 2.4

ตารางที่ 2.4 ส่วนประกอบต่างๆ ที่สำคัญของไมโครคอนโทรลเลอร์

ส่วนประกอบ	ความหมาย
Central Processing Unit (CPU)	ซีพียู เป็นหน่วยประมวลผลกลาง ประกอบไปด้วย 2 ส่วนใหญ่ๆ คือ หน่วยควบคุม และหน่วยคำนวณ
Memory	หน่วยความจำ เป็นที่เก็บข้อมูล และโปรแกรมสั่งงาน หน่วยความจำแบ่งเป็น 2 ประเภท คือ แรม และรอม
Port	พอร์ต เป็นส่วนที่ใช้ติดต่อกับภายนอก มี 2 ประเภท คือ อินพุตพอร์ต (Input Port) และ เอาท์พุตพอร์ต (Output Port)
Bus	บัส เป็นส่วนที่ใช้ในการแลกเปลี่ยนข้อมูลระหว่าง ซีพียู, หน่วยความจำ และ พอร์ต แบ่งเป็น 3 ประเภท คือ บัสที่อยู่ (Address Bus), บัสข้อมูล (Data Bus) และบัสควบคุม (Control Bus) บัสที่อยู่: เป็นบัสทิศทางเดียวที่ส่งค่าแอดเดรสไปยังหน่วยความจำ บัสข้อมูล: เป็นบัส 2 ทิศทางที่เป็นทางผ่านของข้อมูล บัสควบคุม: เป็นบัสที่ส่งหรือรับสัญญาณควบคุมระหว่าง ซีพียู กับ อุปกรณ์ภายนอก

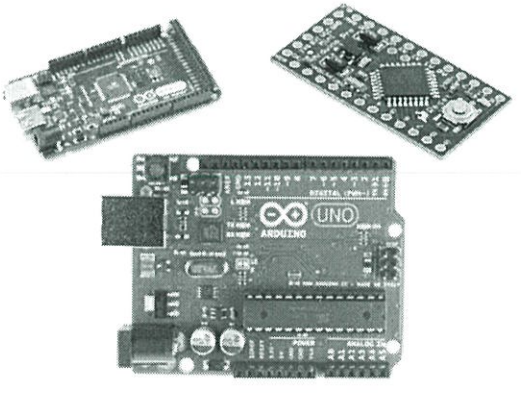
สถาปัตยกรรมของเอวีอาร์ประกอบด้วยหลากหลายขนาดแต่ในปริญญานิพนธ์นี้จะใช้ไมโครคอนโทรลเลอร์เอทีเมกา 168 (ATMega168) ซึ่งเป็นสถาปัตยกรรมแบบ 8 บิต (8 Bits Microcontroller) ที่มีการจัดเรียงขาต่างๆ ดังรูปที่ 2.9 และเอทีเมกา 168 นั้นจะมีข้อมูลจำเพาะต่างๆ ดังนี้

- หน่วยความจำโปรแกรมแบบแฟลช (Flash Memory) ขนาด 16 กิโลไบต์
- หน่วยความจำข้อมูลแบบเอสแรม (SRAM) ขนาด 2 กิโลไบต์
- หน่วยความจำข้อมูลแบบอีอีพรอม (EEPROM) ขนาด 1 กิโลไบต์
- สนับสนุนการรับส่งข้อมูลแบบไอแสควร์ซี (I^2C: Inter Integrated Bus)
- อินพุตพอร์ต (Input Port) และเอาท์พุตพอร์ต (Output Port) จำนวน 20 พอร์ต
- ตัวแปลงสัญญาณอนาล็อกเป็นดิจิตอล (Analog to Digital Converter) ขนาด 10 บิต จำนวน 6 ช่อง
- ความถี่ใช้งานสูงสุด 20 เมกกะเฮิรตซ์
- ทำงานตั้งแต่แรงดัน 1.8-5.5 โวลต์



รูปที่ 2.9 การจัดวางขาของเอทีเมกา 168

โดยที่เอวีอาร์จะสามารถใช้เอวีอาร์สตูดิโอ (AVRStudio) หรืออาร์ดูอิโน่ (Arduino) ในการพัฒนาตัวไมโครคอนโทรลเลอร์ได้ โดยในปริญญานิพนธ์นี้จะเลือกใช้ซอฟต์แวร์ของอาร์ดูอิโนในการพัฒนาตัวโปรแกรมของไมโครคอนโทรลเลอร์ เนื่องจากเป็นซอฟต์แวร์ที่ใช้งานง่ายและมีหมวดหมู่ของโปรแกรม (Program Library) เป็นจำนวนมาก อาร์ดูอิโน่เป็นโครงการพัฒนาไมโครคอนโทรลเลอร์ของเอวีอาร์ซึ่งเป็นซอฟต์แวร์แบบที่สามารถให้ผู้ใช้พัฒนาได้ (Open Source) ที่ประกอบไปด้วยส่วนของฮาร์ดแวร์ที่ใช้ไมโครคอนโทรลเลอร์เอทีเมกา 168 (Development Board) ตัวอย่างเช่นในรูปที่ 2.10 และส่วนของการพัฒนาโปรแกรมที่ใช้สำหรับติดต่อกับไมโครคอนโทรลเลอร์โดยจะทำการติดต่อผ่านทางอาร์เอส 232 (RS232)



รูปที่ 2.10 ตัวอย่างฮาร์ดแวร์ของเอวีอาร์

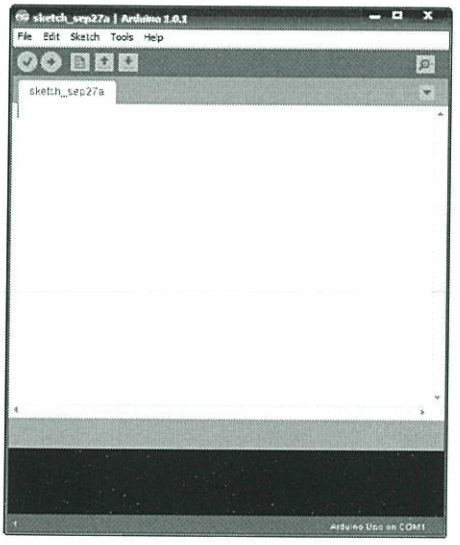
2.5 อาร์ดูอิโน่

การที่จะให้ไมโครคอนโทรลเลอร์ทำงานนั้นผู้ใช้จำเป็นจะต้องเขียนโปรแกรมคำสั่งเข้าไปสั่งการให้ไมโครคอนโทรลเลอร์ทำงานต่างๆ เช่น อ่านค่าจากตัวเซนเซอร์ แสดงผลค่าผ่านจอแสดงผล เป็นต้น ในปริญญานิพนธ์นี้จะใช้ตัวโปรแกรมที่มีชื่อว่าอาร์ดูอิโน่ในการพัฒนาซึ่งตัวโปรแกรมนี้จะมีลักษณะคล้ายๆ กับภาษาซี (C Language) โดยในการเขียนคำสั่งนั้นจะต้องมีฟังก์ชันอย่างน้อย 2 ฟังก์ชัน คือ ฟังก์ชันสำหรับการตั้งค่า (Set up Function) และฟังก์ชันสำหรับการลูปของโปรแกรม (Loop Function) โดยรูปแบบการเขียนฟังก์ชันทั้ง 2 เป็นไปดังนี้

- void setup () : ฟังก์ชันนี้เป็นฟังก์ชันการตั้งค่า ซึ่งมีไว้สำหรับกำหนดการทำงานของระบบหรือคุณสมบัติของระบบซึ่งคำสั่งที่อยู่ภายในฟังก์ชันนี้นั้นจะเปรียบเสมือนโปรแกรมย่อย
- void loop () : ฟังก์ชันนี้เป็นฟังก์ชันสำหรับคำสั่งที่ต้องการให้มีการวนลูป (ทำซ้ำ) ซึ่งคำสั่งต่างๆ ที่อยู่ภายในจะเปรียบเสมือนโปรแกรมหลัก

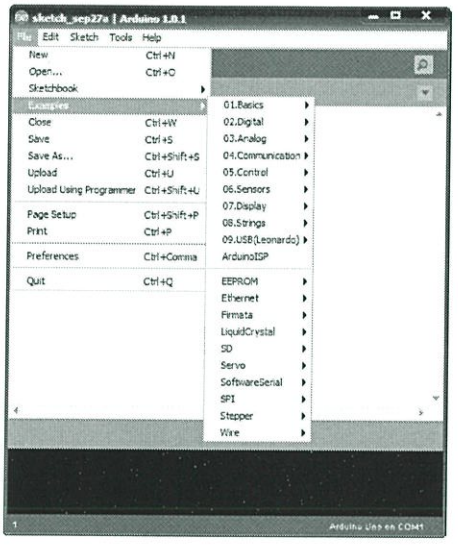
การอธิบายวิธีใช้งานรวมไปถึงวิธีการลงโปรแกรมอาร์ดูอิโน่เบื้องต้น

1) ทำการโหลดตัวโปรแกรมอาร์ดูอิโน่จากเวปไซต์ http://www.arduino.cc/en/Main/Software
2) แตกไฟล์ arduino-1.0.1-windows.zip
3) เปิดโปรแกรม arduino.exe จะขึ้นหน้าต่างโปรแกรมดังรูปที่ 2.11



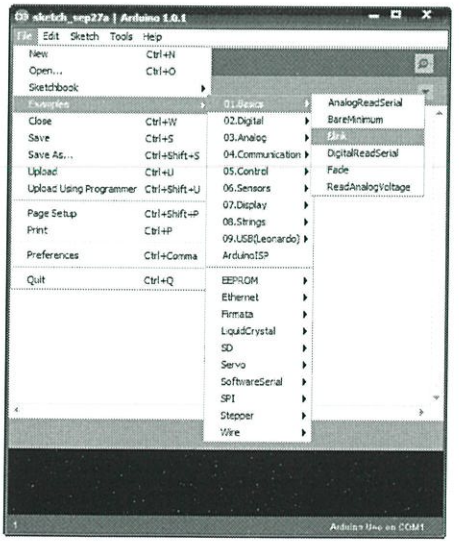
รูปที่ 2.11 หน้าต่างโปรแกรมอาร์ดูอิโน่

4) ในโปรแกรมอาร์ดูอิโน่นั้นจะมีหมวดหมู่ของโปรแกรมต่างๆ และในหมวดหมู่ของโปรแกรมนั้นจะมีตัวอย่างของโปรแกรม (Example) โดยมีลำดับขั้นตอนคือ กดที่ File > Example จะได้หน้าต่างของโปรแกรมขึ้นตามรูปที่ 2.12



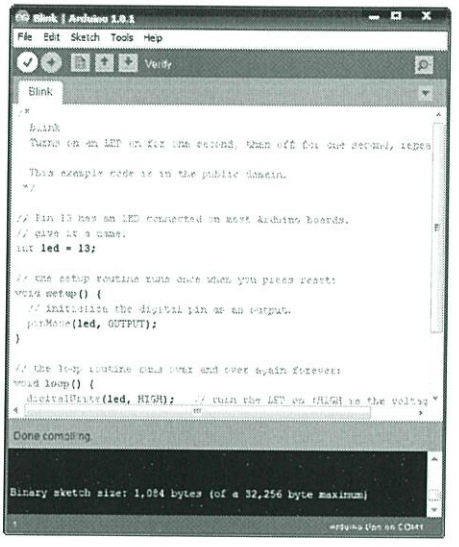
รูปที่ 2.12 ขั้นตอนการเข้าสู่การเปิดตัวอย่างของโปรแกรม

5) เลือกโปรแกรมตัวอย่างในที่นี้จะเลือกโปรแกรม Blink ซึ่งอยู่ในหมวดหมู่โปรแกรมที่ชื่อว่า Basics ดังรูปที่ 2.13



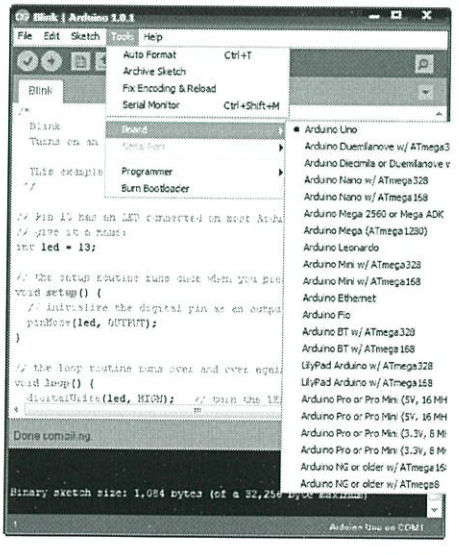
รูปที่ 2.13 การเปิดโปรแกรมตัวอย่าง Blink

6) ทำการตรวจสอบโปรแกรมที่เขียน ในที่นี้เป็นโปรแกรมตัวอย่าง Blink โดยการกดปุ่ม Verify ตามรูปที่ 2.14



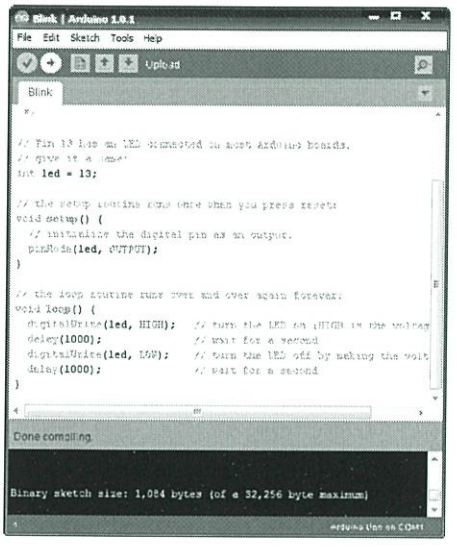
รูปที่ 2.14 การตรวจสอบโปรแกรมที่เขียน

7) ก่อนจะทำการเขียนข้อมูลคำสั่งเข้าสู่ไมโครคอนโทรลเลอร์ด้วยโปรแกรมอาร์ดูอิโน่ นั้นจะต้องเลือกคอมพอร์ต (Com port) และเลือกชนิดของบอร์ด (Board) โดยกดที่ Tools > Board สำหรับเลือกชนิดของบอร์ดและ Tools > Serial Port สำหรับเลือกคอมพอร์ต ดังรูปที่ 2.15 เป็นการแสดงการเลือกชนิดของบอร์ด



รูปที่ 2.15 การเลือกชนิดของบอร์ด

8) สุดท้ายเป็นการเขียนข้อมูลคำสั่งเข้าสู่ไมโครคอนโทรลเลอร์สามารถทำได้โดยการกดปุ่ม Upload ดังรูปที่ 2.16



รูปที่ 2.16 ทำการเขียนข้อมูลคำสั่งเข้าสู่ไมโครคอนโทรลเลอร์

โดยที่การเขียนคำสั่งต่างๆ นั้นจะคล้ายคลึงกับการเขียนภาษาซี และสามารถหาหมวดหมู่ของโปรแกรมต่างๆ เพิ่มเติมได้

2.6 การสื่อสารข้อมูล

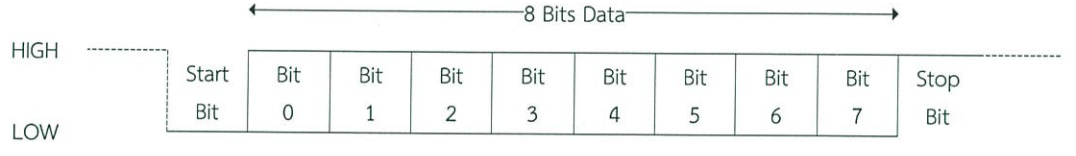
การสื่อสารข้อมูลคือการส่งข้อมูลที่ทำการเข้ารหัสแล้วระหว่างอุปกรณ์ 2 ชนิด โดยมีองค์ประกอบหลักสำคัญอยู่ 3 ส่วน คือ อุปกรณ์ฝั่งส่ง อุปกรณ์ฝั่งรับ และตัวกลาง ซึ่งตัวกลางจะสามารถแบ่งได้เป็น 2 ประเภทได้แก่ แบบมีสาย และแบบไร้สาย โดยจะสามารถแบ่งแยกรูปแบบในการสื่อสารข้อมูลได้เป็น 3 แบบคือ การสื่อสารทางเดียว การสื่อสารกึ่งสองทาง และการสื่อสารสองทาง

2.6.1 การสื่อสารแบบอนุกรม

การสื่อสารแบบอนุกรมเป็นการสื่อสารที่มีการรับส่งข้อมูลทีละบิตเรียงต่อกันไปจนสิ้นสุดซึ่งการสื่อสารแบบนี้จะแตกต่างจากการสื่อสารแบบขนานเนื่องจากการสื่อสารแบบขนานนั้นถ้าจำนวนข้อมูลมากขึ้นก็ต้องเพิ่มจำนวนสายสัญญาณจึงไม่มีความเหมาะสมในการใช้งานที่มีระยะทางในการรับส่งข้อมูลไกลๆ ซึ่งเป็นการสิ้นเปลืองค่าใช้จ่ายเป็นอย่างมาก แต่การสื่อสารแบบอนุกรมนั้นใช้จำนวนสายสัญญาณเพียง 2 ถึง 3 เส้นเท่านั้น จึงทำให้ใช้เวลาในการรับส่งข้อมูลมากกว่าการสื่อสารแบบขนานแต่ค่าใช้จ่ายจะลดลง โดยการสื่อสารแบบอนุกรมนั้นจะสามารถแบ่งแยกลักษณะของการสื่อสารได้เป็น 2 ประเภท คือ การสื่อสารแบบซิงโครนัส และการสื่อสารแบบอะซิงโครนัส

การสื่อสารแบบซิงโครนัส (Synchronous) การสื่อสารแบบซิงโครนัสนั้นจำเป็นจะต้องอาศัยสัญญาณนาฬิกามาช่วยกำหนดจังหวะในการสื่อสาร

การสื่อสารแบบอะซิงโครนัส (Asynchronous) การสื่อสารแบบอะซิงโครนัสนั้นจะใช้รูปแบบในการส่งข้อมูล (Bit Pattern) เป็นตัวกำหนดว่าข้อมูลส่วนไหนเป็นข้อมูลอะไรซึ่งรูปแบบโดยทั่วไปจะเป็นดังนี้คือ เริ่มต้นด้วยบิตเริ่มต้น (Start Bit) ตามด้วยข้อมูล (Data) ตามด้วยบิตพาริตี้ (Parity Bit) สิ้นสุดด้วยบิตสิ้นสุด (Stop Bit) โดยการส่งข้อมูลแบบอะซิงโครนัสนั้นจะมีอุปกรณ์ที่เรียกว่า ยูอาร์ท (UART : Universal Asynchronous Receiver/Transmitter) เป็นตัวควบคุมในการรับส่งสามารถแสดงรูปแบบในการส่งข้อมูลได้ดังรูปที่ 2.17



รูปที่ 2.17 รูปแบบในการส่งข้อมูลแบบอะซิงโครนัส

ในการสื่อสารแบบอนุกรมข้อมูลที่ใช้ในการสื่อสารจะอยู่ในลักษณะของกลุ่มบิตข้อมูล (Bit Stream) จำเป็นจะต้องสนใจในเรื่องอัตราเร็วในการรับส่งข้อมูลโดยจะใช้การกำหนดอัตราบอร์ด (Buad Rate) ในการกำหนดซึ่งจะมีค่ามาตรฐานได้แก่ 110 150 300 1200 2400 4800 9600 19200 เป็นต้น

2.6.2 มาตรฐานอาร์เอส 232 (RS-232)

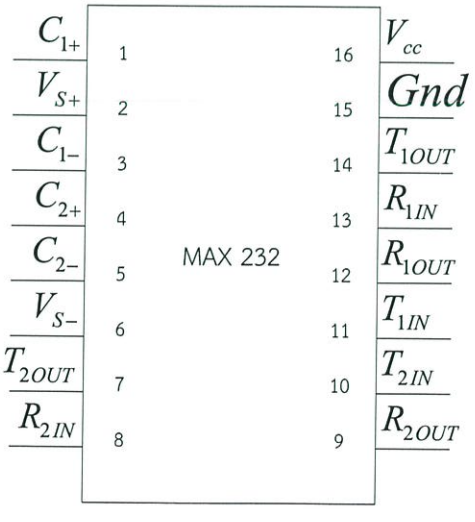
ในปัจจุบันมีอุปกรณ์ต่างๆ มากมายจากผู้ผลิตหลายๆ ค่ายจึงจำเป็นต้องมีตัวกำหนดมาตรฐานเพื่อให้อุปกรณ์ต่างๆ สามารถสื่อสารกันได้ โดยมาตรฐานที่นิยมใช้กันอย่างแพร่หลายนั้นก็คือมาตรฐานอาร์เอส 232 (RS 232 : Recommended Standard 232) ซึ่งจะมีระดับแรงดันอยู่ในช่วง -15 ถึง 15 โวลต์ โดยลอจิก “0” จะมีแรงดันอยู่ในช่วง 3 ถึง 15 โวลต์ และลอจิก “1” จะมีแรงดันในช่วง -15 ถึง -3 โวลต์ ซึ่งอาร์เอส 232 นั้นมีสายสัญญาณ 3 เส้นก็คือ สายส่งข้อมูล (Tx) สายรับข้อมูล (Rx) และกราวด์ (Gnd)

2.6.3 ทีทีแอล

ทีทีแอล (TTL : Transistor – Transistor Logic) เป็นการกำหนดระดับแรงดันที่เกิดขึ้นในยุคแรกๆ ใช้ในการสื่อสารระหว่างทรานซิสเตอร์โดยมีช่วงของระดับแรงดันอยู่ที่ 0 ถึง 5 โวลต์ และในปัจจุบันมีการพัฒนาระดับแรงดันทีทีแอลแบบทีทีแอลแรงดันต่ำ (LVTTL : Low-Voltage TTL) ซึ่งมีระดับแรงดันอยู่ที่ 0 ถึง 3.3 โวลต์

2.6.4 การแปลงระดับแรงดัน

การที่จะทำการสื่อสารระหว่างคอมพิวเตอร์กับไมโครคอนโทรลเลอร์นั้นจำเป็นจะต้องทำการแปลงระดับแรงดัน เนื่องจากข้อมูลที่ออกมาจากพอร์ตอนุกรมของคอมพิวเตอร์มีระดับแรงดันตามมาตรฐานอาร์เอส 232 แต่ข้อมูลที่สามารถสื่อสารกับไมโครคอนโทรลเลอร์ได้นั้นต้องมีระดับแรงดันแบบทีทีแอล จะเห็นว่ามีระดับแรงดันที่ต่างกันจึงจำเป็นจะต้องทำการแปลงระดับแรงดันโดยที่การแปลงระดับแรงดันนั้นจะอาศัยไอซีเบอร์แม็กซ์ 232 (IC MAX232) ในการแปลงระดับแรงดันซึ่งแม็กซ์ 232 นั้นมีการจัดเรียงขาแสดงดังรูปที่ 2.18



รูปที่ 2.18 การเรียงขาของไอซีแม็กซ์ 232

จากรูปที่ 2.18 จะเห็นได้ว่าขาของไอซีแม็กซ์ 232 ที่เกี่ยวข้องกับการสื่อสารข้อมูล จะได้แก่ขา 7 8 9 10 11 12 13 และ 14 โดยที่รายละเอียดต่างๆ ของขาดังกล่าวแสดงได้ตามตารางที่ 2.5

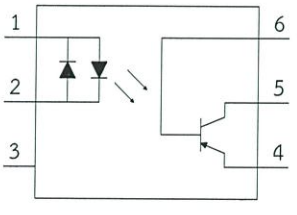
จากตารางที่ 2.5 จะเห็นว่าไอซีแม็กซ์ 232 สามารถแปลงระดับแรงดันได้ 2 ชันแนล และการที่จะแปลงสัญญาณนั้นจำเป็นจะต้องเลือกใช้ขาของไอซีแม็กซ์ 232 ให้ตรงกัน คือ ถ้าเลือกใช้ขา 8 ในการรับเข้าสัญญาณอาร์เอส 232 สัญญาณทีทีแอลที่จะออกจะออกที่ขา 9 เป็นต้น

ตารางที่ 2.5 รายละเอียดของขาต่างๆ ของไอซีแม็กซ์ 232 ที่เกี่ยวข้องกับการสื่อสารข้อมูล

หมายเลขของขา	ชื่อขา	รายละเอียดของขา	ระดับแรงดัน (โวลต์)
7	T_{2OUT}	ส่งออกสัญญาณอาร์เอส 232	-15 ถึง 15
8	R_{2IN}	รับเข้าสัญญาณอาร์เอส 232	-15 ถึง 15
9	R_{2OUT}	ส่งออกสัญญาณทีทีแอล	0 ถึง 3.3 หรือ 0 ถึง 5
10	T_{2IN}	รับเข้าสัญญาณทีทีแอล	0 ถึง 3.3 หรือ 0 ถึง 5
11	T_{1IN}	รับเข้าสัญญาณทีทีแอล	0 ถึง 3.3 หรือ 0 ถึง 5
12	R_{1OUT}	ส่งออกสัญญาณทีทีแอล	0 ถึง 3.3 หรือ 0 ถึง 5
13	R_{1IN}	รับเข้าสัญญาณอาร์เอส 232	-15 ถึง 15
14	T_{1OUT}	ส่งออกสัญญาณอาร์เอส 232	-15 ถึง 15

2.7 ออปโตไอโซเลเตอร์ (Opto Isolator)

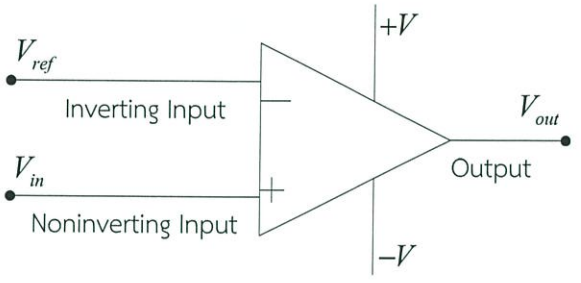
ออปโตไอโซเลเตอร์ หรือสามารถเรียกว่าออปโตคัปเปลอร์ (Opto-Coupler) เป็นอุปกรณ์อิเล็กทรอนิกส์ที่ใช้ในการเชื่อมต่อทางแสงโดยอาศัยหลักการเปลี่ยนสัญญาณไฟฟ้าเป็นสัญญาณแสง และเปลี่ยนสัญญาณแสงกลับมาเป็นสัญญาณไฟฟ้าตามเดิม สามารถจำแนกได้เป็นหลายๆ ชนิดซึ่งแต่ละชนิดจะประกอบด้วยไดโอดเปล่งแสง (LED) เป็นแหล่งกำเนิดแสง และตัวรับแสงจะเป็นโฟโตทรานซิสเตอร์ (Photo Transistor) หรือ โฟโตไดโอด (Photo Diode) โดยที่ทั้งสองส่วนจะถูกผลิตไว้ในตัวถังเดียวกันโดยตัวอย่างของภายในตัวถังของออปโตไอโซเลเตอร์เบอร์ เฮช 11 เอเอ 1 (H11AA1) นั้นสามารถแสดงได้ดังรูปที่ 2.19



รูปที่ 2.19 โครงสร้างภายในของ H11AA1

2.8 วงจรเปรียบเทียบสัญญาณ (Comparator)

วงจรเปรียบเทียบสัญญาณเป็นวงจรที่ไว้สำหรับเปรียบเทียบสัญญาณใดๆ กับสัญญาณอ้างอิงโดยวงจรเปรียบเทียบสัญญาณสามารถแสดงได้ดังรูปที่ 2.20



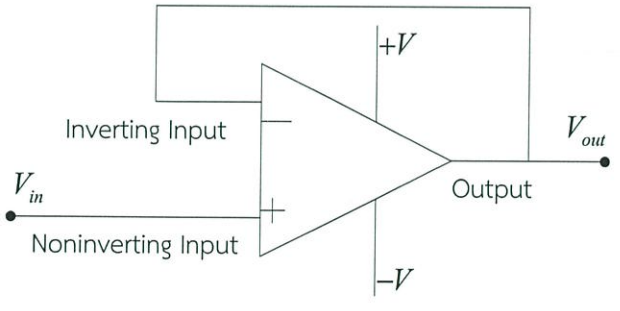
รูปที่ 2.20 ลักษณะของวงจรเปรียบเทียบสัญญาณ

จากรูปที่ 2.20 เอาต์พุตของวงจรเปรียบเทียบสัญญาณนั้นจะเป็นไปได้ 2 กรณี คือ

1. $V_{out} = +V$ เมื่อ $V_{in} \geq V_{ref}$
2. $V_{out} = -V$ เมื่อ $V_{in} \leq V_{ref}$

2.9 วงจรบัฟเฟอร์ (Buffer)

วงจรบัฟเฟอร์เป็นวงจรที่ทำหน้าที่ส่งผ่านสัญญาณโดยจะไม่มีการขยายสัญญาณ หรือเป็นวงจรที่มีอัตราขยายเท่ากับ 1 โดยวงจรบัฟเฟอร์ สามารถแสดงได้ดังรูปที่ 2.21

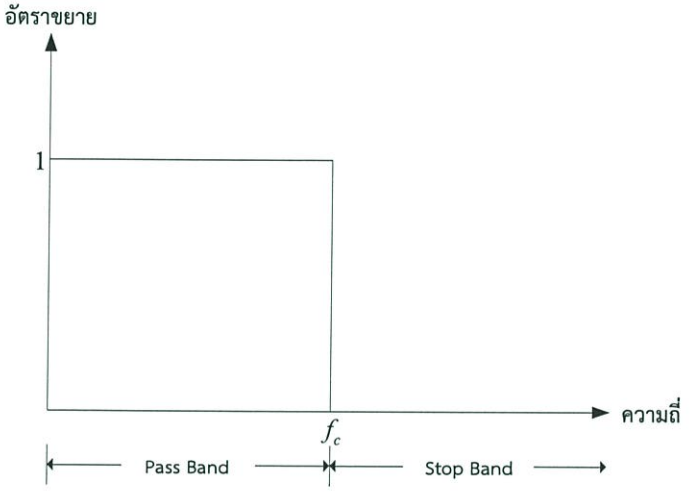


รูปที่ 2.21 วงจรบัฟเฟอร์

จากรูปที่ 2.21 เอาต์พุตของวงจรบัฟเฟอร์ $V_{in} = V_{out}$ ทำให้อัตราขยาย $A_v = 1$

2.10 วงจรกรองความถี่ (Filter)

สำหรับปริญญานิพนธ์นี้จะกล่าวถึงวงจรกรองความถี่ต่ำผ่าน (Low Pass Filter) เท่านั้น วงจรกรองความถี่ต่ำผ่านจะเป็นวงจรที่ยอมให้สัญญาณที่มีความถี่ต่ำผ่านไปได้เพียงอย่างเดียว และจะกั้นความถี่สูงเอาไว้ โดยผลตอบสนองทางความถี่ของวงจรกรองความถี่ต่ำผ่านในอุดมคติสามารถแสดงได้ดังรูปที่ 2.22



รูปที่ 2.22 ผลตอบสนองทางความถี่ของวงจรกรองความถี่ต่ำผ่านในอุดมคติ

2.11 ลอจิกเกท (Logic Gate)

ในทางตรรกศาสตร์สามารถแบ่งแยกตรรกะได้ 2 สถานะคือ จริง (TRUE) กับ ไม่จริง (FALSE) มาใช้ประโยชน์ในทางอิเล็กทรอนิกส์ได้ คือให้สภาวะการทำงาน (NO) แทนจริง และให้สภาวะไม่ทำงาน (OFF) แทนไม่จริง โดยที่ทางอิเล็กทรอนิกส์ตรรกวิทยาค่าอินพุต และค่าเอาต์พุตจะอยู่ในรูปของระดับแรงดัน 2 ระดับคือ ระดับสูง (HIGH level: 1) และระดับต่ำ (LOW level: 0) ซึ่งจะนำใช้แทนค่าสภาวะการทำงานคือ สภาวะทำงานแทนด้วยระดับสูง และสภาวะไม่ทำงานแทนด้วยระดับต่ำ โดยที่ระดับแรงดันระดับสูง และต่ำนั้นขึ้นอยู่กับวงจรที่ใช้งาน เช่น วงจรหนึ่งระดับแรงดันสูงเป็น 5 โวลต์ ระดับแรงดันต่ำเป็น 0 โวลต์ แต่ในอีกวงจรหนึ่งระดับแรงดันสูงเป็น +5 โวลต์ ระดับแรงดันต่ำเป็น -5 โวลต์วงจรพื้นฐานในทางตรรกศาสตร์นั้นประกอบไปด้วย

2.11.1 แอนเกท (AND gate)

แอนเกทเป็นวงจรหลายอินพุตที่ให้ค่าเอาต์พุตของวงจรเป็น 1 เมื่อทุกๆ อินพุตเป็น 1 โดยตารางค่าความจริง และสัญลักษณ์ของแอนเกทสามารถแสดงได้ดังรูปที่ 2.23 ตัวอย่างแอนเกทเช่น SN7408 CD404081

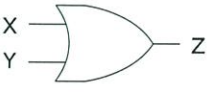
X	Y	Z
0	0	0
0	1	0
1	0	0
1	1	1

รูปที่ 2.23 ตารางค่าความจริง (บน) และสัญลักษณ์ (ล่าง) ของแอนเกท

2.11.2 ออเกท (OR gate)

ออเกทเป็นวงจรหลายอินพุตที่จะให้ค่าเอาต์พุตเป็น 1 เมื่อมีอินพุตอย่างน้อย 1 ตัวอยู่ในอินพุตโดยตารางค่าความจริง และสัญลักษณ์ของวงจรหรือสามารถแสดงได้ดังรูปที่ 2.24 ตัวอย่างออเกท เช่น SN7432 CD4001

X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	1

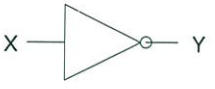


รูปที่ 2.24 ตารางค่าความจริง (บน) และสัญลักษณ์ (ล่าง) ของออเกท

2.11.3 อินเวิร์ทเตอร์ (Inverter)

อินเวิร์ทเตอร์เป็นวงจรอินพุตเดียวจะให้ค่าเอาต์พุตตรงข้ามกับค่าอินพุตโดยตารางค่าความจริง และสัญลักษณ์ของอินเวิร์ทเตอร์สามารถแสดงได้ดังรูปที่ 2.25 ตัวอย่างไอซี อินเวิร์ทเตอร์ เช่น SN7404 CD4069

X	Y
0	1
1	0



รูปที่ 2.25 ตารางค่าความจริง (บน) และสัญลักษณ์ (ล่าง) ของอินเวิร์ทเตอร์

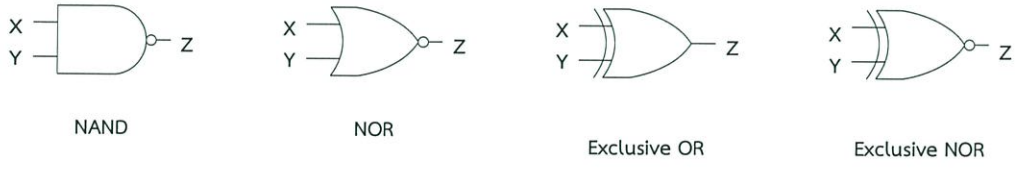
จากวงจรพื้นฐานสามารถสามารถนำมาสร้างวงจรทางตรรกศาสตร์ได้อีกเช่น แนนเกท (NAND gate) คือแอนเกทตามด้วยอินเวิร์ทเตอร์ นอร์เกท (NOR gate) คือออเกทตามด้วยอินเวิร์ทเตอร์ วงจรเอ็กคลูซีฟออ (Exclusive-OR) คือวงจรที่ให้เอาต์พุตเป็น 1 เมื่ออินพุตมีค่าไม่เท่ากัน และเอ็กคลูซีฟนอร์ (Exclusive-NOR) คือวงจรที่ให้เอาต์พุตเป็น 1 เมื่ออินพุตเท่ากัน โดยตารางค่าความจริง และสัญลักษณ์ของวงจรลอจิกเกทชนิดต่างๆ สามารถแสดงได้ดังรูปที่ 2.26

X	Y	Z
0	0	1
0	1	1
1	0	1
1	1	0

X	Y	Z
0	0	1
0	1	0
1	0	0
1	1	0

X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	0

X	Y	Z
0	0	1
0	1	0
1	0	0
1	1	1



รูปที่ 2.26 ตารางค่าความจริง และสัญลักษณ์ของวงจรลอจิกเกทชนิดต่างๆ

2.12 บลูธูท (Bluetooth)

บลูธูทเป็นระบบการสื่อสารของอุปกรณ์อิเล็กทรอนิคส์แบบสองทางด้วยคลื่นวิทยุระยะสั้น (Short-Range Radio Link) เพื่อใช้ในการส่งข้อมูล ภาพ และเสียงผ่านช่องสัญญาณความถี่ 2.4 GHz โดยจะแบ่งช่องความถี่ออกเป็นขนาดเล็กจำนวนหลายช่องเพื่อใช้ในการสลับช่องสัญญาณไปมาในขณะสื่อสารเพื่อป้องกันสัญญาณรบกวนและการดักสัญญาณ สามารถทำงานได้ในระยะ 5-10 เมตร บลูธูทสามารถจัดการให้อุปกรณ์หลายชนิดติดต่อสื่อสารได้พร้อมกัน โดยจะมีอุปกรณ์ตัวหนึ่งทำหน้าที่เป็น master และอุปกรณ์อื่นๆ ทำหน้าที่เป็น slave โดยในการรวมกลุ่มของอุปกรณ์เป็นเครือข่ายของบลูธูทนั้นเรียกว่า พิโคเน็ต (piconet) โดยในแต่ละพิโคเน็ต สามารถมีอุปกรณ์ที่ติดต่อสื่อสารกันได้ทั้งหมด 8 ชิ้นซึ่งการติดต่อสื่อสารจะเป็นแบบจุดต่อหลายจุด ช่องสัญญาณและแบนด์วิดธ์จะถูกแบ่งระหว่างอุปกรณ์ในพิโคเน็ต

2.13 ระบบปฏิบัติการแอนดรอยด์ (Android Operating System)

เป็นระบบปฏิบัติการสำหรับอุปกรณ์พกพาเช่น โทรศัพท์มือถือ และคอมพิวเตอร์แท็บเล็ต ทำงานบนลินุกซ์ เคอร์เนล เริ่มพัฒนาโดยบริษัทแอนดรอยด์ จากนั้นถูกซื้อโดยบริษัทกูเกิลและได้ร่วมมือกับกลุ่มบริษัททางด้านฮาร์ดแวร์ ซอฟต์แวร์ และการสื่อสาร เพื่อจัดตั้งองค์กรความร่วมมือที่มีชื่อว่า Open Handset Alliance โดนมีจุดประสงค์ในการสร้างแพลตฟอร์มสำหรับอุปกรณ์พกพาที่มีพื้นฐานอยู่บนมาตรฐานเปิด (Open Standard)

2.13.1 สถาปัตยกรรมแอนดรอยด์ (Android Architecture)

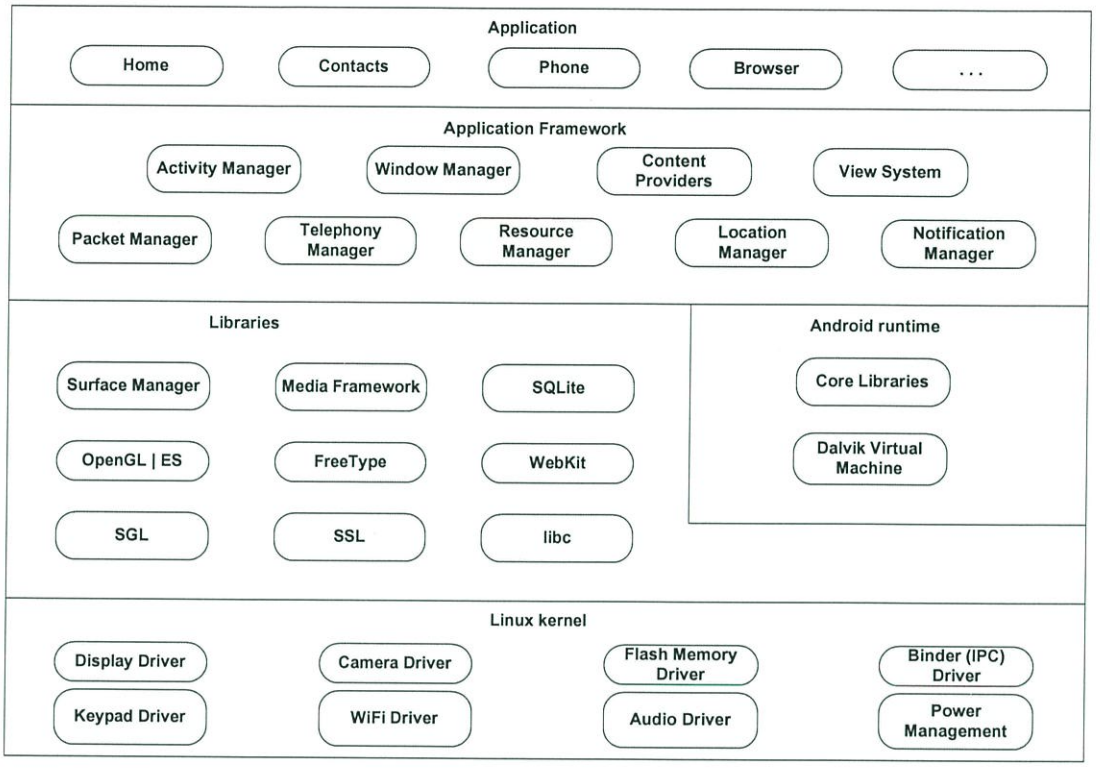
สถาปัตยกรรมของระบบแอนดรอยด์สามารถแสดงได้ ดังรูปที่ 2.27

1) แอพพลิเคชัน (Application)

แอพพลิเคชันต่างๆ ทั้งที่ติดตั้งมากับเครื่องอยู่แล้ว (Core Application) เช่น Phone dialer, Email, Contacts, Web browser และ Google Play เป็นต้น รวมถึงแอพพลิเคชันที่เราสร้างขึ้น ซึ่งแอพพลิเคชันทั้งหมดในส่วนนี้จะเขียนด้วยภาษาจาวา

2) Application Framework

เป็นส่วนของเฟรมเวิร์คที่ใช้พัฒนาแอพพลิเคชั่น ซึ่งประกอบด้วยคอมโพเนนต์พื้นฐานต่างๆ ที่ใช้ในการสร้างแอพพลิเคชั่นของผู้พัฒนา โดยคอมโพเนนต์เหล่านี้จะติดตั้งมากับแอนดรอยด์อยู่แล้วและผู้พัฒนาสามารถแทนที่ด้วยคอมโพเนนต์ที่เราสร้างขึ้นเองได้ ส่วนสำคัญใน Application Framework เช่น



รูปที่ 2.27 สถาปัตยกรรมแอนดรอยด์

- Activity Manager คือคอมโพเนนต์ที่ควบคุม Lifecycle ของแอพพลิเคชั่น
- Content Providers คือคอมโพเนนต์ที่ทำให้แอพพลิเคชันต่างๆ สามารถแชร์ข้อมูลกันได้
- View System ประกอบด้วยคอมโพเนนต์ที่ใช้สร้างส่วนติดต่อผู้ใช้ เช่น ปุ่ม เท็กซ์บ็อกซ์ ลิสต์ กริด เป็นต้น
- Location Manager คือคอมโพเนนต์ที่ใช้ในการระบุตำแหน่งโดยอ้างอิงจาก GPS เสาสัญญาณมือถือ หรือ WIFI เป็นต้น

3) Native Libraries

เป็นส่วนที่เป็นไลบรารีของแอนดรอยด์ ซึ่งทั้งหมดเขียนด้วยภาษา C หรือ C++ และถูกคอมไพล์มาสำหรับฮาร์ดแวร์ของอุปกรณ์แต่ละรุ่น ไลบรารีที่สำคัญต่างๆ เช่น

- Surface Manager คือไลบรารีจัดการส่วนแสดงผลที่มีความสามารถในการผสมกราฟิกทั้ง 2 มิติและ 3 มิติจากแอพพลิเคชันต่างๆ เข้าด้วยกัน ทำให้สามารถสร้างเอฟเฟ็คเช่น วินโดวส์ ที่มองทะลุไปข้างหลังได้และ Transition ในรูปแบบต่างๆ

- Media Libraries คือไลบรารีที่จัดเตรียมบริการในการเล่น และบันทึกเสียง วีดีโอ และรูปภาพในฟอร์แมตต่างๆ เช่น MPEG4, H.264, MP3, AAC, AMR, JPG และ PNG

- SQLite คือ Database Engine ที่มีประสิทธิภาพและมีขนาดเล็ก เพื่อให้เราจัดเก็บข้อมูลของแอพพลิเคชันไว้ในรูปแบบของฐานข้อมูลเชิงสัมพันธ์ (Relation Database)

- WebKit คือไลบรารีที่ใช้แสดงเนื้อหาเว็บเพจ ซึ่งเป็นตัวเดียวกับที่ใช้ใน Google Chrome และ Apple Safari รวมถึงเว็บบราวเซอร์ในมือถือ iPhone และมือถือตระกูล S60 ของโนเกียด้วย

2.13.2 วงจรการทำงานของแอคทิวิตี

ในแอนดรอยด์นั้นแอคทิวิตีต่างๆ จะถูกจัดการในลักษณะของสแตค (Stack) กล่าวคือ แอคทิวิตีใหม่ที่เริ่มต้นขึ้นมาจะถูกวางไว้ที่ด้านบนสุดของสแตค และกลายเป็นแอคทิวิตีที่กำลังทำงาน (Running Activity) ส่วนแอคทิวิตีที่ทำงานมาก่อนจะอยู่ถัดลงไป และจะกลับมาอยู่ด้านบนตามเดิมเมื่อแอคทิวิตีใหม่จบการทำงานโดยมีลักษณะดังรูปที่ 2.28

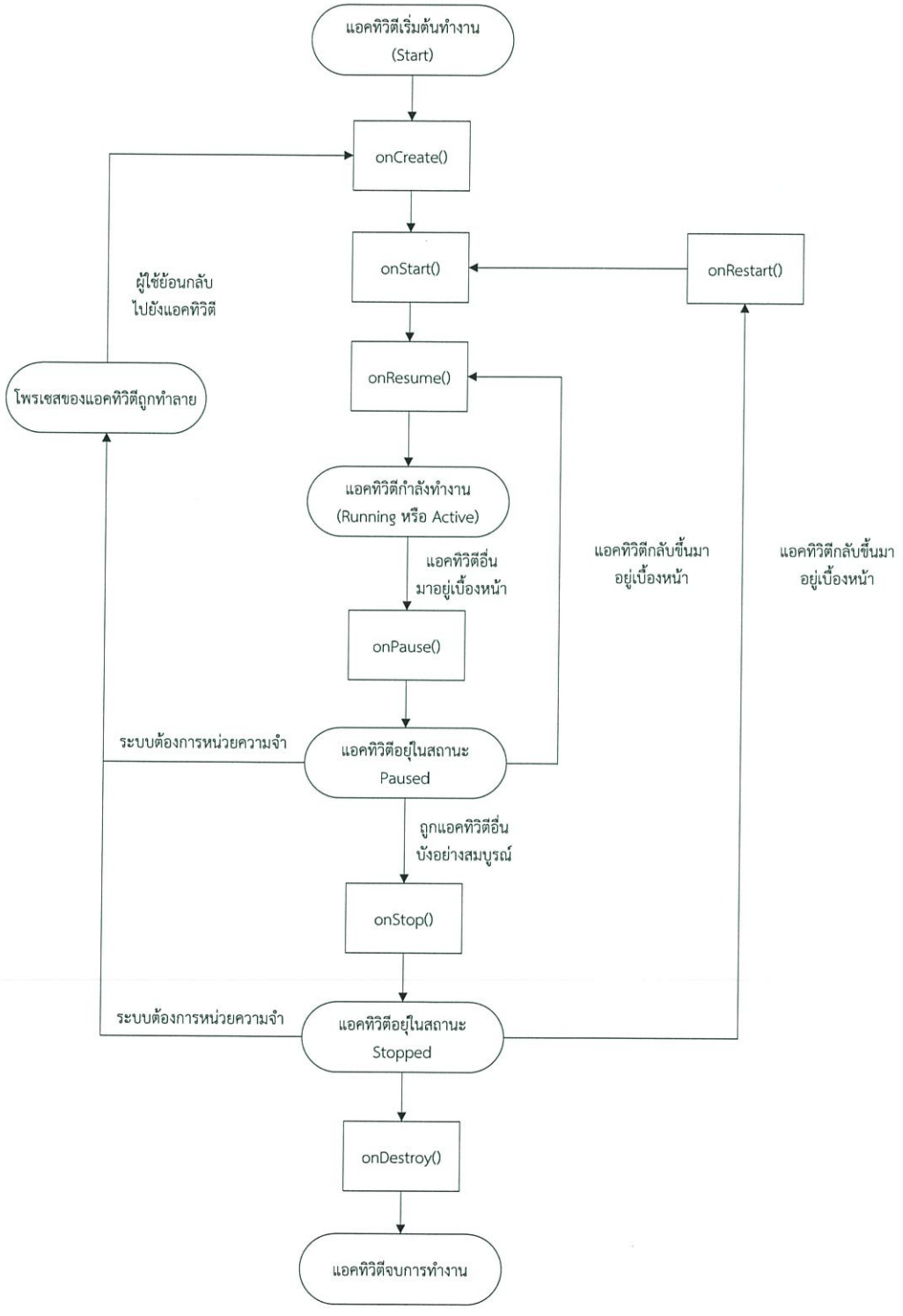
ซึ่งผู้พัฒนาไม่สามารถควบคุมสถานะของแอคทิวิตีได้เนื่องจากแอนดรอยด์จะจัดการให้ แต่แอนดรอยด์จะแจ้งให้ทราบเมื่อสถานะของแอคทิวิตีเปลี่ยนไปโดยผ่านทางเมธอดโดยรายละเอียดของแต่ละเมธอดมีดังนี้

- **onCreate(Bundle)** จะถูกเรียกเมื่อแอคทิวิตีเริ่มต้นทำงาน จึงเหมาะสำหรับการทำ Innitialization ต่างๆ อย่างเช่นการสร้างวิวหรือผูกข้อมูลไว้กับลิสต์ เป็นต้น เมธอดนี้มีพารามิเตอร์ 1 ตัวเป็นชนิด Bundle ซึ่งจะมีค่า null ในกรณีของแอคทิวิตีใหม่ที่เพิ่งเริ่มต้นแต่หากเป็นแอคทิวิตีที่โพรเซสของมันถูกทำลายไปเพราะระบบต้องการใช้หน่วยความจำพารามิเตอร์ตัวนี้จะเก็บข้อมูลในสถานะเดิมของแอคทิวิตีไว้

- **onStart()** จะถูกเรียกเมื่อแอคทิวิตีกำลังจะแสดงผลออกไปให้ผู้ใช้เห็น

- **onResume()** จะถูกเรียกเมื่อแอคทิวิตีเริ่มโต้ตอบกับผู้ใช้ โดยแอคทิวิตีจะอยู่บนสุดของสแตค และได้รับอินพุตต่างๆ จากผู้ใช้

- **onPause()** จะถูกเรียกเมื่อแอคทิวิตีกำลังจะลงไปอยู่เบื้องหลัง โดยทั่วไปเนื่องมาจากมีแอคทิวิตีอื่นกำลังจะขึ้นมาอยู่เบื้องหน้าแทน เมธอดนี้เป็นจุดที่ผู้ใช้สามารถ commit การเปลี่ยนแปลงต่างๆ ลงฐานข้อมูล, หยุดการเล่นแอนิเมชั่น, และงานใดๆที่จะเป็นภาระต่อ CPU



รูปที่ 2.28 วงจรการทำงานของแอคทิวิตี

- **onStop()** จะถูกเรียกเมื่อแอคทิวิตีไม่ได้แสดงผลให้ผู้ใช้เห็นแล้ว เนื่องจากถูกแอคทิวิตีอื่นบังอย่างสมบูรณ์ ซึ่งอาจเป็นแอคทิวิตีใหม่หรือแอคทิวิตีเก่าที่ถูกดึงกลับขึ้นมาเบื้องหน้า แต่กรณีที่ระบบเหลือความจำน้อยมากไม่สามารถเรียกมายังเมธอด onStop ได้จึงทำลายโพรเซสของแอคทิวิตีไปเลย ดังนั้นบางครั้งเมธอด onStop จึงเป็นเมธอดสุดท้ายที่ถูกเรียกใช้

- **onRestart()** จะถูกเรียกเมื่อแอคทิวิตีกำลังจะกลับมาแสดงผลให้ผู้ใช้เห็นอีกครั้ง หลังจากถูกแอคทิวิตีอื่นบังอย่างสมบูรณ์

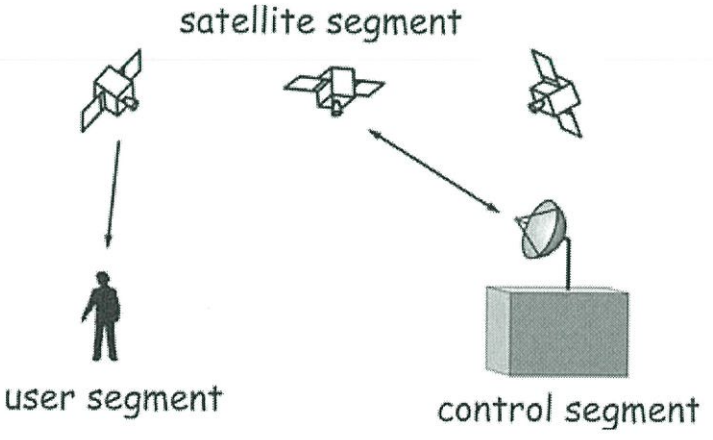
- **onDestroy()** จะถูกเรียกเป็นเมธอดสุดท้ายก่อนที่แอคทิวิตีจะถูกทำลายไป ซึ่งอาจเป็นเพราะแอคทิวิตีกำลังจะจบการทำงานหรือระบบขอทำลายแอคทิวิตีเพื่อนำหน่วยความจำไปใช้ แต่หากระบบเหลือหน่วยความจำน้อยมากก็จะทำลายโพรเซสของแอคทิวิตีไปเลยโดยไม่เรียกมายังเมธอด onDestroy

2.14 จีพีเอส (GPS: Global Positioning System)

จีพีเอสถูกพัฒนาขึ้นมาโดยหน่วยงาน The United States Department of Defence (DOD) ของสหรัฐอเมริกา ใช้ในการหาพิกัดตำแหน่งในที่ต่าง ๆ บนโลกด้วยดาวเทียม โดยการใช้ส่งสัญญาณคลื่นวิทยุลงมาบนโลก เมื่อเครื่องรับจีพีเอสรับสัญญาณได้ จึงเอาข้อมูลต่างๆ ที่มากับสัญญาณจีพีเอสไปคำนวณหาตำแหน่ง เพื่อประโยชน์ทางการทหาร

2.14.1 องค์ประกอบของระบบตำแหน่งบนโลก

ลักษณะทั่วไปของระบบ GPS ประกอบด้วยส่วนประกอบที่สำคัญ 3 ส่วน ดังรูปที่ 2.29



รูปที่ 2.29 องค์ประกอบระบบกำหนดตำแหน่งบนโลก

- **ส่วนอวกาศ (Space Segment)** ในระบบดาวเทียม GPS จะประกอบด้วยดาวเทียมทั้งหมด 24 ดวง โดยดาวเทียมจำนวน 21 ดวง จะใช้ในการบอกค่าพิกัด ส่วนที่เหลือ 3 ดวง จะสำรองเอาไว้ ดาวเทียมทั้ง 24 ดวงนี้จะมีวงโคจรอยู่ 6 วงโคจรด้วยกัน โดยแบ่งจำนวนดาวเทียมวงโคจรละ 4 ดวง และมีรัศมีวงโคจรสูงจากพื้นโลกประมาณ 20,200 กิโลเมตร (12,600 ไมล์) วงโคจรทั้ง 6 จะเอียงทำมุมกับเส้นศูนย์สูตร (Equator) เป็นมุม 55 องศา ในลักษณะสานกันคล้ายลูกตะกร้อ โดยมีระยะเวลาในการโคจร 1 รอบเท่ากับ 12 ชม.

- **สถานีควบคุม (Control Station Segment)** ในส่วนของสถานีควบคุมจะประกอบด้วย 5 สถานีย่อย (Monitor Station) ตั้งอยู่ที่เมือง Diego Garcia, Ascension Island, Kwajalein, และ Hawaii ส่วนสถานีควบคุมหลัก (Master Control Station) 1 สถานี ซึ่งเป็นศูนย์ควบคุมการทำงานของระบบดาวเทียม GPS ตั้งอยู่ที่เมือง Colorado Springs รัฐ Colorado สหรัฐอเมริกา สถานีควบคุมต่าง ๆ เหล่านี้มีหน้าที่คอยติดต่อสื่อสาร (Tracking) กับดาวเทียมทำการคำนวณผล (Computation) เพื่อบอกตำแหน่งของดาวเทียมแต่ละดวง และส่งข้อมูลที่ได้ไปยังดาวเทียมอยู่ตลอดเวลา ทำให้ข้อมูลที่ได้เป็นข้อมูลที่ทันสมัยอยู่เสมอ

- **ส่วนผู้ใช้ (User Segment)** ผู้ใช้ประกอบด้วย 2 ส่วนใหญ่ ๆ คือ ส่วนที่เกี่ยวข้องกับพลเรือน (Civilian) และส่วนที่เกี่ยวกับทางทหาร (Military) ในส่วนของผู้ใช้จะมีหน้าที่พัฒนาเครื่องรับสัญญาณ (Receiver) ให้ทันสมัยและสะดวกแก่การใช้งาน สามารถที่จะใช้ได้ทุกแห่งในโลก และให้ค่าที่มีความถูกต้องสูง

2.15 แอปเซิร์ฟ (Appserve)

เป็นโปรแกรมที่ติดตั้งเพื่อใช้ทดสอบการใช้งานภาษาต่างๆในการพัฒนาโปรแกรม ระบบหรือเวปไซต์ โดยที่โปรแกรมแอปเซิร์ฟนี้รวบรวมเอาแหล่งโอเพนซอร์สหลายๆอย่างไว้ด้วยกัน แค่ทำการติดตั้งโปรแกรม แอปเซิร์ฟก็สามารถใช้โปรแกรมที่ติดมาทั้งหมดได้ โดยจะมีโปรแกรมต่างๆ ดังนี้

2.15.1 อาปาเช่

มีลักษณะเป็นซอฟต์แวร์ที่สามารถให้ผู้ใช้พัฒนาได้ ซึ่งตัวอาปาเช่นี้ เป็นโปรแกรมจำลองเวปเซิร์ฟเวอร์ มีหน้าที่จัดเก็บโฮมเพจและส่งโฮมเพจ ไปยังเวปบราวเซอร์ที่มีการเรียกเข้ายังเวปเซิร์ฟเวอร์ที่เก็บโฮมเพจนั้นอยู่

2.15.2 พีเอชพี

ย่อมาจาก PHP Hypertext Preprocessor หรือชื่อเดิม Personal Home Page เป็นภาษา สำหรับใช้ในการเขียนโปรแกรมบนเวปไซต์ สามารถเขียนได้หลากหลายโปรแกรม เช่นเดียวกับภาษาทั่วไป คำสั่งของพีเอชพี สามารถสร้างผ่านทางโปรแกรมแก้ไขข้อความทั่วไป เช่น โนตแพด ซึ่งทำให้การทำงานของ พีเอชพี สามารถทำงานได้ในระบบปฏิบัติการหลักเกือบทั้งหมด ซึ่งจะมีอิสระในการเลือก ระบบปฏิบัติการ และ เวปเซิร์ฟเวอร์

ภาษาพีเอชพี จะเป็นส่วนประกอบภายในเวปเพจ โดยคำสั่งจะปรากฏ ระหว่าง <?php ... ?> เช่น < ? php echo "Hello, World !" // เป็นการให้แสดงคำว่า Hello, World ! ? >

1) คำสั่ง If...else if (เงื่อนไข)
   {คำสั่งต่างๆ เมื่อเงื่อนไขเป็นจริง ;}
   else
   {คำสั่งต่างๆ เมื่อเงื่อนไขเป็นเท็จ;}
2) คำสั่ง include ( ): เป็นฟังก์ชันแทรกไฟล์จากภายนอก ซึ่งไฟล์ที่จะนำมาแทรกลง ไปได้ ต้องสามารถรวมเข้าเป็นโคดเดียวกันกับไฟล์เวปเพจที่เป็นผู้เข้ามาแทรก เช่น อาจเป็นโคดเกี่ยวกับเอชทีเอ็มแอล พีเอชพี ซีเอสเอส และจาวาสคริปต์ เป็นต้น
   รูปแบบ : include file หรือ include(file)
3) คำสั่ง require ( ): เป็นฟังก์ชันในการแทรกไฟล์เช่นเดียวกันกับ include() แต่มี ข้อแตกต่างกันคือ require( ) นั้นจะแทรกไฟล์เข้ามา โดยไม่ขึ้นกับเงื่อนไข ดังนั้นจึงนิยมใช้ require() กับไฟล์ที่ต้องการแทรกเสมอ ไม่ว่ากรณีใดๆ ก็ตาม
   รูปแบบ : require file หรือ require(file)
4) ฟังก์ชันพีเอชพี: เป็นชุดคำสั่งสำหรับการกระทำอย่างใดอย่างหนึ่ง ซึ่งโดยทั่วไป แล้วมักเป็นสิ่งที่ต้องทำซ้ำๆ จึงแยกคำสั่งบางส่วนออกมาสร้างเป็นฟังก์ชันไว้ต่างหาก
   รูปแบบ : function func_name ($param1, $param2, ..., $paramN)
   {
   ชุดคำสั่งต่างๆ ;
   return ค่าที่ส่งกลับ ;
   }

2.15.3 มายเอสคิวแอล

มายเอสคิวแอล คือ โปรแกรมระบบจัดการฐานข้อมูล มีหน้าที่เก็บข้อมูลอย่างเป็นระบบ รองรับคำสั่งเอสคิวแอล (SQL ย่อมาจาก Structured Query Language) เป็นเครื่องมือสำหรับเก็บข้อมูลที่ต้องใช้ร่วมกับเครื่องมือหรือโปรแกรมอื่นเพื่อให้ได้ระบบงานที่รองรับความต้องการของผู้ใช้ เช่น ทำงานร่วมกับ เวปเซิร์ฟเวอร์ เพื่อให้บริการแก่ภาษาสคริปต์ที่ทำงานฝั่งเครื่องบริการ เช่น ภาษาพีเอชพี ภาษาเอเอสพี หรือ ภาษาเจเอสพี เป็นต้น หรือทำงานร่วมกับโปรแกรมประยุกต์ (Application Program) เช่น ภาษาวิชวลเบสิก ภาษาจาวาสคริปต์ หรือภาษาซี เป็นต้น

มายเอสคิวแอล เป็นระบบฐานข้อมูลแบบที่สามารถให้ผู้ใช้พัฒนาได้ สำหรับจัดการระบบฐานข้อมูล (Database System) ผ่านเอสคิวแอล

1) ฟังก์ชันในการติดต่อฐานข้อมูล ฟังก์ชั่นในการติดต่อฐานข้อมูลของพีเอชพีนั้นสามารถเชื่อมต่อข้อมูลได้หลายแบบ โดยที่การเชื่อมต่อกับฐานข้อมูลแต่ละแบบก็จะใช้ฟังก์ชันที่แตกต่างกันด้วย สำหรับฟังก์ชันในการเชื่อมต่อมายเอสคิวแอล ชื่อฟังก์ชันจะขึ้นต้นด้วย คำว่า mysql เป็นส่วนใหญ่ทำให้นำไปใช้ได้ง่ายโดยมีคำสั่งต่างๆ ที่สำคัญเช่น

- คำสั่ง mysql_connect ( ) : เป็นฟังก์ชันการเชื่อมต่อไปยัง มายเอสคิวแอลซึ่งอาจถือได้ว่าเป็นฟังก์ชันแรกที่ต้องใช้เสมอในการติดต่อกับมายเอสคิวแอล และหากฟังก์ชันนี้ทำงานไม่สำเร็จก็ไม่สามารถทำงานอย่างอื่นต่อไปได้ ดังนั้นจึงควรทำการตรวจสอบผลลัพธ์ของฟังก์ชันนี้ทุกครั้งก่อนจะดำเนินการใดๆ ต่อไป หากฟังก์ชันนี้ทั้งหมดทำงานสำเร็จหรือสามารถเชื่อมต่อกับมายเอสคิวแอลได้จะคืนค่าทรู (true) ถ้าการเชื่อมต่อไม่สำเร็จจะคืนค่ากลับมาเป็นฟอลท์ (false)

รูปแบบ : mysql_connect(host, username, password) โดย host คือ ชื่อของโฮสต์ที่ติดตั้งมายเอสคิวแอลเอาไว้ อาจกำหนดเป็นชื่อเครื่องเซิร์ฟเวอร์หรือหมายเลข ไอพีแอดเดรส (IP Address) ก็ได้ หากติดตั้งมายเอสคิวแอลเอาไว้ในเครื่องที่กำลังใช้งานอยู่ สามารถกำหนดเป็น “localhost” หรือ “127.0.0.1” ได้ username คือ ชื่อผู้ใช้หรือล็อกอิน ซึ่งเป็นชื่อที่กำหนดไว้ขณะติดตั้งโปรแกรม password คือ รหัสผ่าน ซึ่งเป็นรหัสที่กำหนดไว้ขณะติดตั้งโปรแกรม

- คำสั่ง mysql_close ( ): เป็นฟังก์ชันในการปิดการเชื่อมต่อกับมายเอสคิวแอลหลังการใช้งานเสร็จ

รูปแบบ : mysql_close(connection_name) โดย connection_name คือ ตัวแปรที่เกิดจากการใช้ฟังก์ชัน mysql_connect()

2) ฟังก์ชันในการเลือกฐานข้อมูล ในการใช้ฐานข้อมูลมายเอสคิวแอลนั้นต้องกำหนดชื่อฐานข้อมูลที่จะใช้งานก่อน ซึ่งสามารถใช้ คอมมานไลน์ (Command Line) ของมายเอสคิวแอลได้โดยตรงคือการใช้ คำสั่ง USE แต่ พีเอชพี ได้มีฟังก์ชันในการเลือกฐานข้อมูลได้สะดวกขึ้น คือ

- คำสั่ง mysql_select_db ( ) : เป็นฟังก์ชันในการกำหนดชื่อฐานข้อมูลที่จะใช้งานฐานข้อมูล

รูปแบบ : mysql_select_db(db_name) โดย db_name คือ ชื่อของฐานข้อมูล

3) ฟังก์ชันในการคิวรี่ข้อมูล การคิวรีข้อมูล คือ การใช้คำสั่งเอสคิวแอลสำหรับการคิวรีข้อมูลซึ่งจะได้ผลลัพธ์เป็นอะไรนั้นขึ้นอยู่กับคำสั่งเอสคิวแอลที่ใช้ เช่น หากเป็นการอ่านข้อมูลอาจได้ผลลัพธ์เป็นข้อมูลที่อ่านได้ หรือหากเป็นการแก้ไขข้อมูลก็อาจเป็นเพียงข้อความที่บ่งชี้ว่าการทำงานสำเร็จหรือไม่ เป็นต้น ฟังก์ชันเกี่ยวกับการคิวรี ข้อมูลมีดังนี้

- คำสั่ง mysql_query ( ): เป็นฟังก์ชันที่ใช้ในการส่งคำสั่งเอสคิวแอลไปยังฐานข้อมูลมายเอสคิวแอลได้ ทั้งนี้พีเอชพีไม่ได้เป็นผู้ประมวลคำสั่งเอสคิวแอลแต่เป็นเพียงผู้ส่งคำสั่งเอสคิวแอลที่กำหนดขึ้นในรูปแบบสตริง (string) ไปยังฐานข้อมูลเท่านั้น ซึ่งผลลัพธ์เป็นอะไรก็ขึ้นกับคำสั่งเอสคิวแอลที่เขียน ดังนั้นคำสั่งเอสคิวแอลที่ระบุจะถูกหรือผิดพีเอชพีก็ไม่ อาจทราบได้ แต่อย่างไรก็ตามสามารถตรวจสอบผลลัพธ์ของคำสั่งเอสคิวแอลได้โดยพิจารณา จากค่าที่ส่งกลับคืนมาจากฟังก์ชันนี้

รูปแบบ : $result = mysql_query(sql_string) โดย sql_string คือ คำสั่งเอสคิวแอลที่เขียนในรูปแบบของสตริง คำสั่งนี้จะถูกส่งไปที่มายเอสคิวแอล สามารถตรวจสอบผลลัพธ์ที่เกิดขึ้นได้โดยหากคำสั่งเอสคิวแอลเป็นคำสั่งสำหรับการค้นหาข้อมูล (SELECT) หากการทำงานสำเร็จจะคืนค่ากลับมาเป็นข้อมูลที่ค้นมาได้ แต่หากการค้นหาข้อมูลไม่สำเร็จเช่น การเขียนคำสั่งเอสคิวแอลผิด ฟังก์ชันนี้จะคืนค่ากลับมาเป็นฟอลท์ ส่วนคำสั่งเอสคิวแอลในกรณี อื่นๆ เช่น การเปลี่ยนแปลงข้อมูล (INSERT, UPDATE, DELETE) หากการทำงานสำเร็จจะคืนค่าทรูแต่ถ้าไม่ สำเร็จจะคืนค่าฟอลท์

- คำสั่ง mysql_db_query ( ) : เป็นฟังก์ชันในการคิวรี่ข้อมูลเช่นเดียวกันกับฟังก์ชัน mysql_query() แต่ฟังก์ชันนี้จะกำหนดทั้งชื่อฐานข้อมูลและคำสั่งเอสคิวแอลเป็นอาร์กิวเมนต์ (Argument) นั่นคือฟังก์ชันนี้เป็นการรวมฟังก์ชัน mysql_select_db() และ ฟังก์ชัน mysql_query() มาไว้ด้วยกัน

รูปแบบ : mysql_db_query(batabase_name, sql_string) หากต้องการใช้ฟังก์ชันนี้เพื่อการคิวรีข้อมูลก็ไม่จำเป็นต้องใช้ฟังก์ชัน mysql_select_db() หรือต้องใช้คำสั่ง USE ก่อนเพราะสามารถใช้ฟังก์ชันนี้ได้ทันที

4) ฟังก์ชันในการอ่านและแสดงผลข้อมูล ข้อมูลในฐานข้อมูลจะมีลักษณะเป็นอาร์เรย์ ดังนั้นการอ้างอิงถึงข้อมูลเหล่านี้จึงต้อง ใช้วิธีการในรูปแบบของอาร์เรย์เป็นหลัก ฟังก์ชันที่เกี่ยวกับการอ่านข้อมูลมีหลายรูปแบบดังต่อไปนี้

- คำสั่ง mysql_result ( ) : เป็นฟังก์ชันในการดึงข้อมูลจากรีซอลท์เซต (result set) ในคอลัมน์ (field) และลำดับแถวที่ระบุ

รูปแบบ : $data = mysql_result(result_set, row, field_name) โดย result_set คือ ตัวแปร result_set ที่ได้รับจากการใช้ฟังก์ชัน mysql_query() row คือ ลำดับแถวของข้อมูลในรีซอลท์เซตที่ต้องการอ่าน โดยแถวแรกจะมีลำดับเป็น 0 field_name คือ ชื่อของฟิลด์หรือคอลัมน์ที่ต้องการอ่านข้อมูล

- คำสั่ง mysql_fetch_array ( ) : เป็นฟังก์ชันในการอ่านข้อมูลจากรีซอลท์เซตแบบการเลื่อนพ้อยเตอร์ อาร์เรย์ ผลลัพธ์ของฟังก์ชันจะอยู่ในรูปแบบคีย์ (key) หรือแวลู (value) โดยที่คีย์จะเป็นชื่อฟิลด์หรือคอลัมน์ ในขณะที่แวลูลจะเป็นข้อมูลในฟิลด์นั้น

รูปแบบ : $array = mysql_fetch_array(<result_set>)

2.15.4 พีเอชพีมายแอดมิน

การใช้ฐานข้อมูลที่เป็นมายเอสคิวแอลบางครั้งจะมีความลำบากและยุ่งยากในการใช้งาน ดังนั้นจึงมีเครื่องมือในการจัดการฐานข้อมูลมายเอสคิวแอลขึ้น เพื่อให้สามารถจัดการตัวดีบีเอ็มเอส (DBMS) ที่เป็นมายเอสคิวแอลได้ง่ายและสะดวกขึ้น

พีเอชพีมายแอดมินเป็นส่วนที่สร้างโดยภาษาพีเอชพี ซึ่งใช้จัดการฐานข้อมูลมายเอสคิวแอลผ่าน เวปบราวเซอร์ โดยสามารถทำการสร้างฐานข้อมูลใหม่ หรือทำการสร้างตารางใหม่ๆ และยังมีฟังก์ชันที่ใช้ สำหรับการทดสอบการคิวรี่ข้อมูลด้วยภาษาเอสคิวแอลพร้อมกันนั้นยังสามารถทำการแทรก ลบ อัพเดต หรือ แม้กระทั่งใช้คำสั่งต่างๆ ต่างกับการใช้ภาษาเอสคิวแอลในการสร้างตารางข้อมูล

ในส่วนของการแสดงผลหน้าแรกเมื่อเข้าสู่หน้าแสดงผลพีเอชพีมายแอดมินจะแสดงรุ่นของพี เอชพีมายแอดมินที่ใช้งานอยู่ พร้อมทั้งสามารถที่จะจัดการกับรหัสอักขระที่ใช้ในการเก็บข้อมูล ฝั่งเมนูด้านซ้ายจะแสดงข้อมูลของฐานข้อมูลปัจจุบัน (DATABASE NAME) และเมื่อทำการเลือกแล้วจะแสดง โครงสร้างของตารางข้อมูล

2.16 ระบบไฟฟ้ารถยนต์

2.16.1 ระบบไฟแสงสว่าง

รถยนต์ทุกคันจะมีระบบแสงสว่างประกอบอยู่เพื่อความปลอดภัย ซึ่งระบบไฟแสงสว่างจะประกอบด้วย

- ไฟหน้า หรือไฟใหญ่

  ไฟหน้าทำหน้าที่ส่องสว่างบนถนนหน้ารถยนต์ในเวลากลางคืนใช้แสงสีขาวอมส้ม หรือขาวอมเหลืองถึงขาวนวล

- ไฟหรี่

  ไฟหรี่มีหน้าที่บอกความกว้างของตัวรถทางด้านหน้า และจะให้แสงสว่างร่วมกับไฟหน้าใช้แสงสีขาวอมส้ม หรือสีเหลืองอำพัน

- ไฟท้าย

  ไฟท้ายทำหน้าที่บอกความกว้าง และส่องสว่างทางด้านท้ายรถทำให้ผู้ที่ขับตามมองเห็นใช้แสงสีแดง

- ไฟส่องป้ายทะเบียน

  ไฟส่องป้ายทะเบียนทำหน้าที่ส่องส่องสว่างให้ป้ายทะเบียนซึ่งจะติดตั้งอยู่ที่ป้ายทะเบียนท้ายรถใช้แสงสีขาวอมส้ม

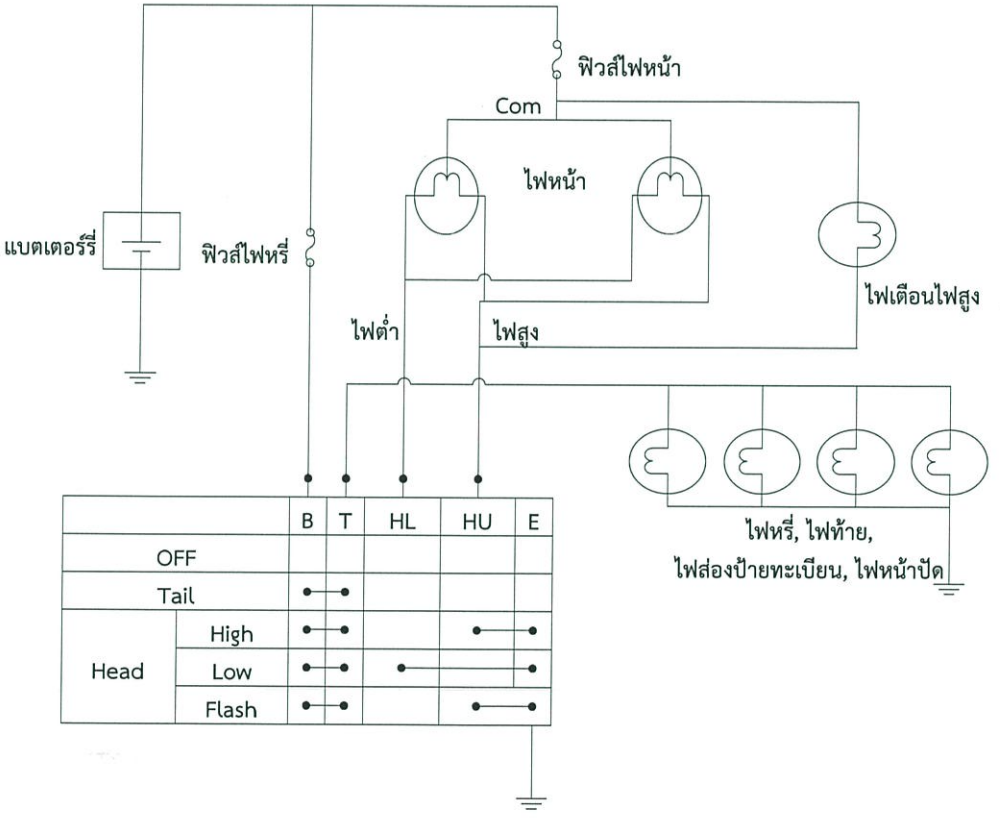
- ไฟแผงหน้าปัด

  ไฟแผงหน้าปัดทำหน้าที่ส่องสว่างให้แผงหน้าปัดเพื่อให้ผู้ขับขี่สามารถมองเห็นเกจวัดต่างๆ ได้ใช้แสงสีขาว หรืออาจจะใช้การเรืองแสง

- ไฟตัดหมอก

  ไฟตัดหมอกทำหน้าที่ส่องทางในขณะที่หมอกลงจัด ทำให้เห็นทางได้ชัดเจนใช้แสงสีเหลือง โดยปกติในประเทศไทยจะไม่มีการติดตั้งไฟตัดหมอกมาให้

วงจรระบบไฟแสงสว่างสามารถแสดงได้ดังรูปที่ 2.30



รูปที่ 2.30 วงจรระบบไฟแสงสว่าง

โดยการทำงานสามารถอธิบายได้ดังนี้

- เมื่อเปิดตำแหน่งไฟหรี่ (Tail)

  ในตำแหน่งนี้ ขั้ว B จะต่อกับขั้ว T ทำให้กระแสไฟฟ้าไหลจากแบตเตอร์รี่เข้าที่ขั้ว B แล้วออกที่ขั้ว T ไปยังหลอดไฟ ทำให้ชุดไฟหรี่ทั้งหมดติด

- เมื่อเปิดสวิตช์ตำแหน่งไฟหน้า (Head)

  ในตำแหน่งนี้ขั้ว B ยังคงต่อกับขั้ว T ซึ่งทำให้ชุดไฟหรี่ยังคงติดอยู่ สวิตช์ตำแหน่งไฟหน้าปรับได้ 3 ตำแหน่งคือ

  - ตำแหน่งไฟสูง (High)

    ตำแหน่งไฟสูงจะทำให้ขั้ว HU ต่อกับขั้ว E ทำให้กระแสไฟฟ้าจากแบตเตอร์รี่ไหลผ่านไส้ไฟสูงผ่านขัว HU ผ่านขั้ว E แล้วลงกราวด์ทำให้ครบวงจรส่งผลให้ไฟสูงติด
  - ตำแหน่งไฟต่ำ (Low)

    ตำแหน่งไฟต่ำจะทำให้ขั้ว HL ต่อกับขั้ว E ทำให้กระแสไฟฟ้าไหลจากแบตเตอร์รี่ไหลผ่านไส้ไฟต่ำผ่านขั้ว HL ผ่านขั้ว E แล้วลงกราวด์ทำให้ครบวงจรส่งผลให้ไฟต่ำติด
  - ตำแหน่งไฟขอทาง (Flash)

    ตำแหน่งไฟขอทางไม่ว่าจะเปิดไฟหรี่ หรือไม่เปิดทำให้ขั้ว HO ต่อกับขั้ว E ทำให้กระแสไฟฟ้าไหลจากแบตเตอร์รี่ไหลผ่านไส้ไฟสูงผ่านขั้ว HO ผ่านขั้ว E แล้วลงกราวด์ทำให้ครบวงจรส่งผลให้ไฟสูงติด

2.16.2 ระบบไฟสัญญาณ

ในรถยนต์จะมีระบบสัญญาณต่างๆ ไว้เพื่อความปลอดภัยในการขับขี่ของผู้ขับขี่และผู้ร่วมทาง โดยที่ระบบสัญญาณต่างๆ นั้นประกอบไปด้วย

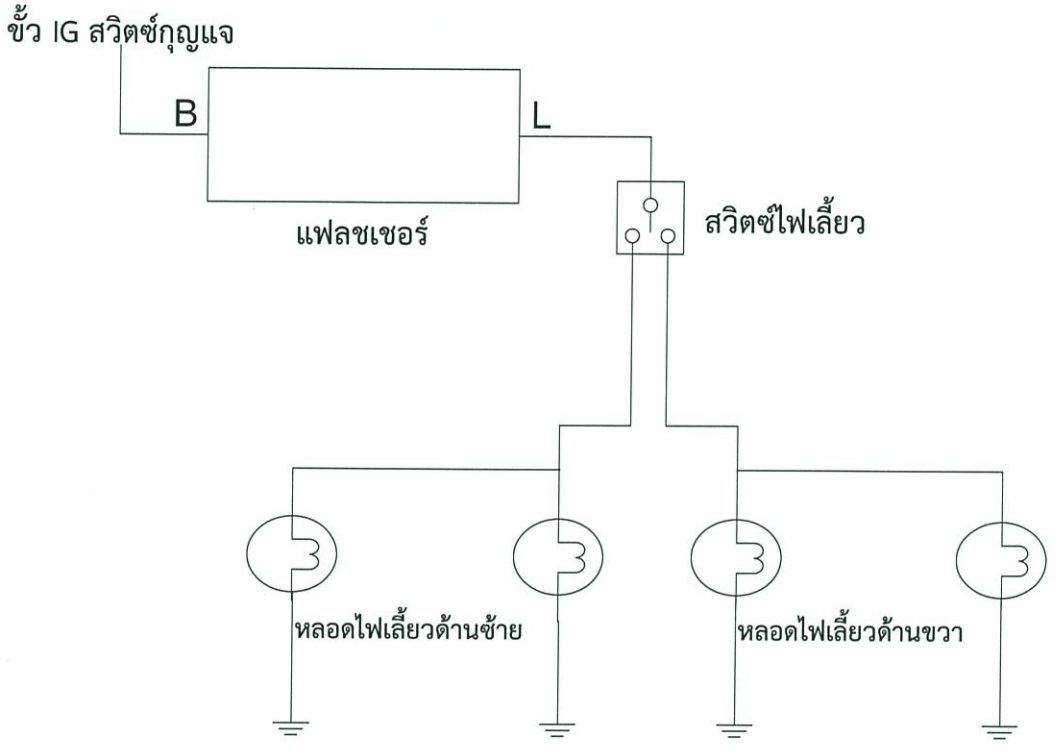
- ไฟเลี้ยว และไฟฉุกเฉิน (Turn signal and Hazard signal)

  ไฟเลี้ยวทำหน้าที่เป็นสัญญาณไฟกระพริบมีสีเหลืองเพื่อเป็นการบอกว่าจะทำการเลี้ยวซ้าย หรือเลี้ยวขวา และไฟฉุกเฉินทำหน้าที่แสดงสัญญาณไฟกระพริบทั้งซ้าย และขวาพร้อมกันโดยวงจรการทำงานของไฟเลี้ยว และไฟฉุกเฉินสามารถแสดงได้ดังรูปที่ 2.31

  ในการทำงานของไฟเลี้ยวนั้นจำเป็นจะต้องมีอุปกรณ์ที่เรียกว่าแฟลชเชอร์ (Flasher) ไว้ทำหน้าที่เป็นสวิตช์ตัดต่อทางไฟฟ้าเพื่อให้ไฟเลี้ยวกระพริบเป็นจังหวะด้วยความถี่ประมาณ 60 – 120 ครั้งต่อนาที โดยแฟลชเชอร์มีอยู่ 5 ลักษณะ คือ

  - แบบขดลวดความร้อน (Hot wire type)
  - แบบโลหะควบคู่ (Bimetal type)

- แบบคอนเดนเซอร์ และรีเลย์ (condenser and relay type)แบบกึ่งทรานซิสเตอร์ (Semi-transistor)
- แบบไอซี หรือแบบทรานซิสเตอร์ (IC type or Transistor)



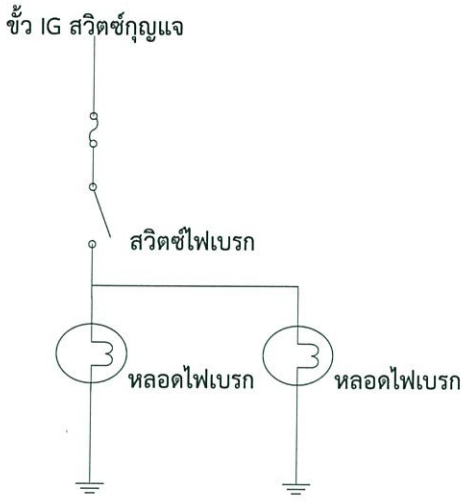
รูปที่ 2.31 วงจรการทำงานของไฟเลี้ยว

- แตร

  แตรทำหน้าที่ผลิตสัญญาณเสียงเพื่อเตือนให้ผู้ใช้ถนนได้ทราบ

- ไฟเบรก

  ไฟเบรกทำหน้าที่บอกให้รถคันที่ตามหลังมาทราบว่ารถคันที่ให้สัญญาณไฟเบรกนั้นมีการชะลอ หรือเบรกโดยสีของหลอดไฟจะมีสีแดงโดยหลอดของไฟเบรกมีทั้งใช้หลอดไฟ และโคมร่วมกับไฟท้าย หรือใช้หลอดไฟและโคมแยกกับไฟท้ายโดยวงจรการทำงานของไฟเบรกสามารถแสดงได้ดังรูปที่ 2.32

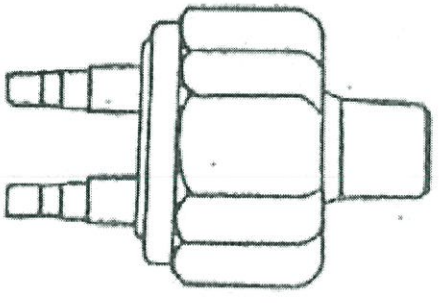


รูปที่ 2.32 วงจรการทำงานของไฟเบรก

จากรูปที่ 2.32 ไฟเบรกจะติดก็ต่อเมื่อสวิตช์ไฟเบรกทำงาน โดยที่สวิตช์ไฟเบรกมีอยู่ 2 ลักษณะคือ

- สวิตช์ไฟเบรกแบบแรงดัน

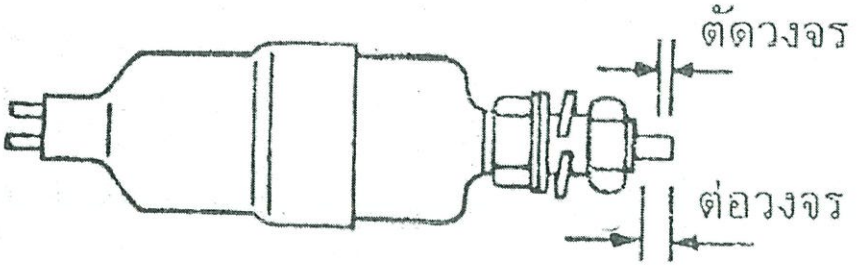
สวิตช์ไฟเบรกแบบแรงดันนั้นจะติดตั้งอยู่ที่แม่ปั๊มเบรกทำงานโดยอาศัยแรงดันน้ำมันโดยที่ลักษณะของสวิตช์ไฟเบรกแบบแรงดันสามารถแสดงได้ดังรูปที่ 2.33



รูปที่ 2.33 สวิตช์ไฟเบรกแบบแรงดัน

- สวิตช์ไฟเบรกแบบกลไก

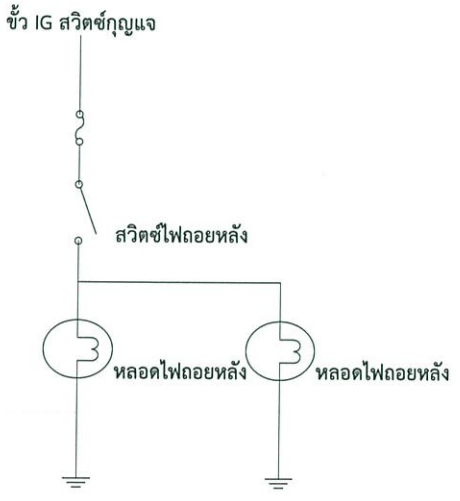
สวิตช์ไฟเบรกแบบกลไกนั้นจะติดอยู่ที่บริเวณคันเหยียบเบรก เมื่อเหยียบเบรกจะทำให้แกนสวิตช์เคลื่อนตัวออกทำให้คอนแทกภายในสวิตช์ติดต่อทางไฟฟ้าทำให้กระแสไหลผ่านสวิตช์ไฟเบรกไหลไปยังหลอดไฟเบรกได้ โดยที่ลักษณะของสวิตช์ไฟเบรกแบบกลไกนั้นสามารถแสดงได้ดังรูปที่ 2.34



รูปที่ 2.34 สวิตช์ไฟเบรกแบบกลไก

- ไฟถอยหลัง

  ไฟถอยหลัง หรือไฟเตือนถอยหลังทำหน้าที่บอกว่ารถที่ให้สัญญาณไฟถอยหลังกำลังจะถอยไปด้านหลัง และไฟเตือนจะมีความสว่างมากพอสำหรับที่รถด้านหลังจะสามารถมองเห็นได้ทั้งกลางวัน และกลางคืนโดยวงจรการทำงานของไฟถอยสามารถแสดงได้ดังรูปที่ 2.35



รูปที่ 2.35 วงจรการทำงานของไฟหลัง

บทที่ 3

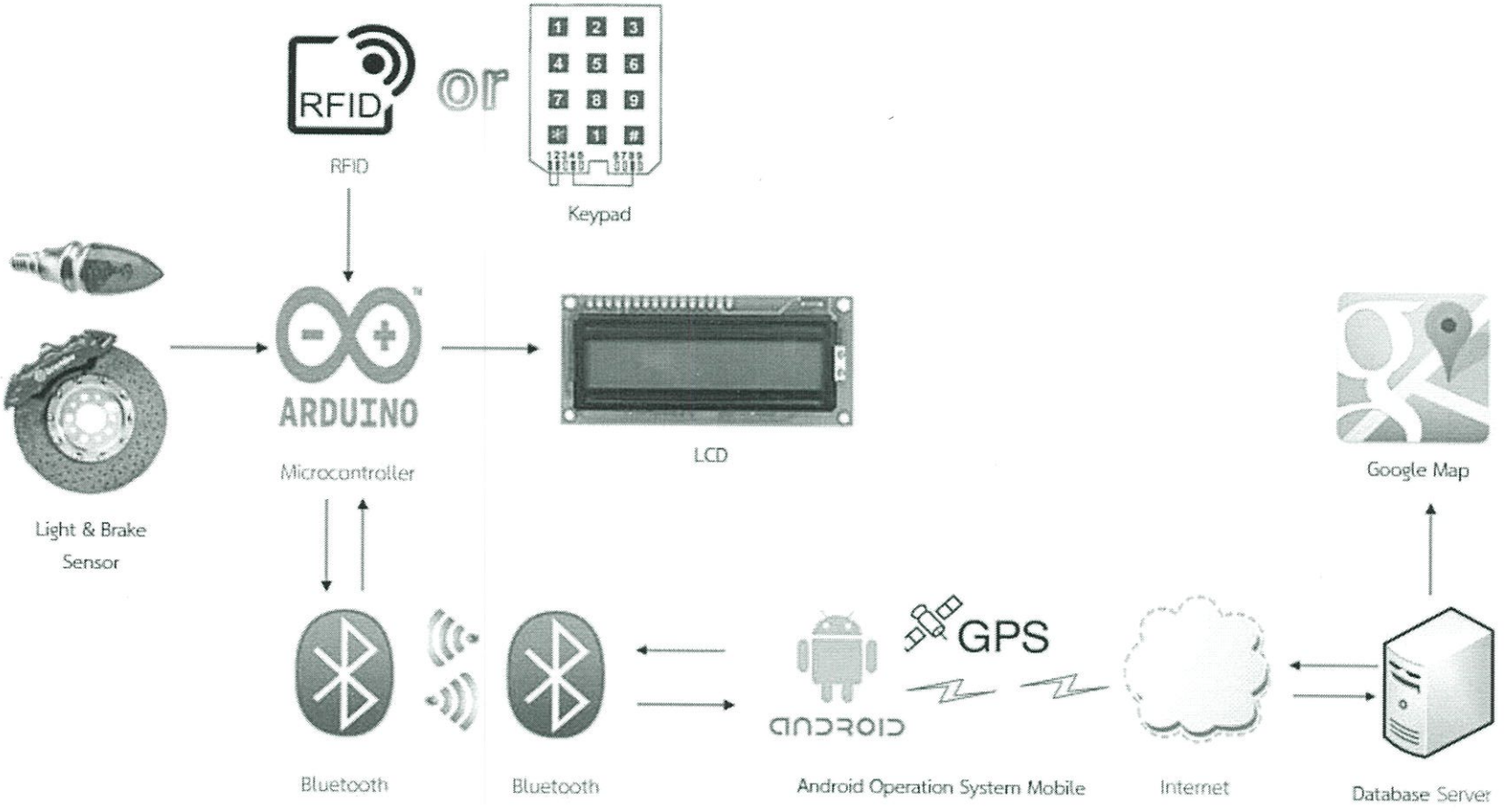
การออกแบบ และจัดทำปริญญานิพนธ์

3.1 การออกแบบ

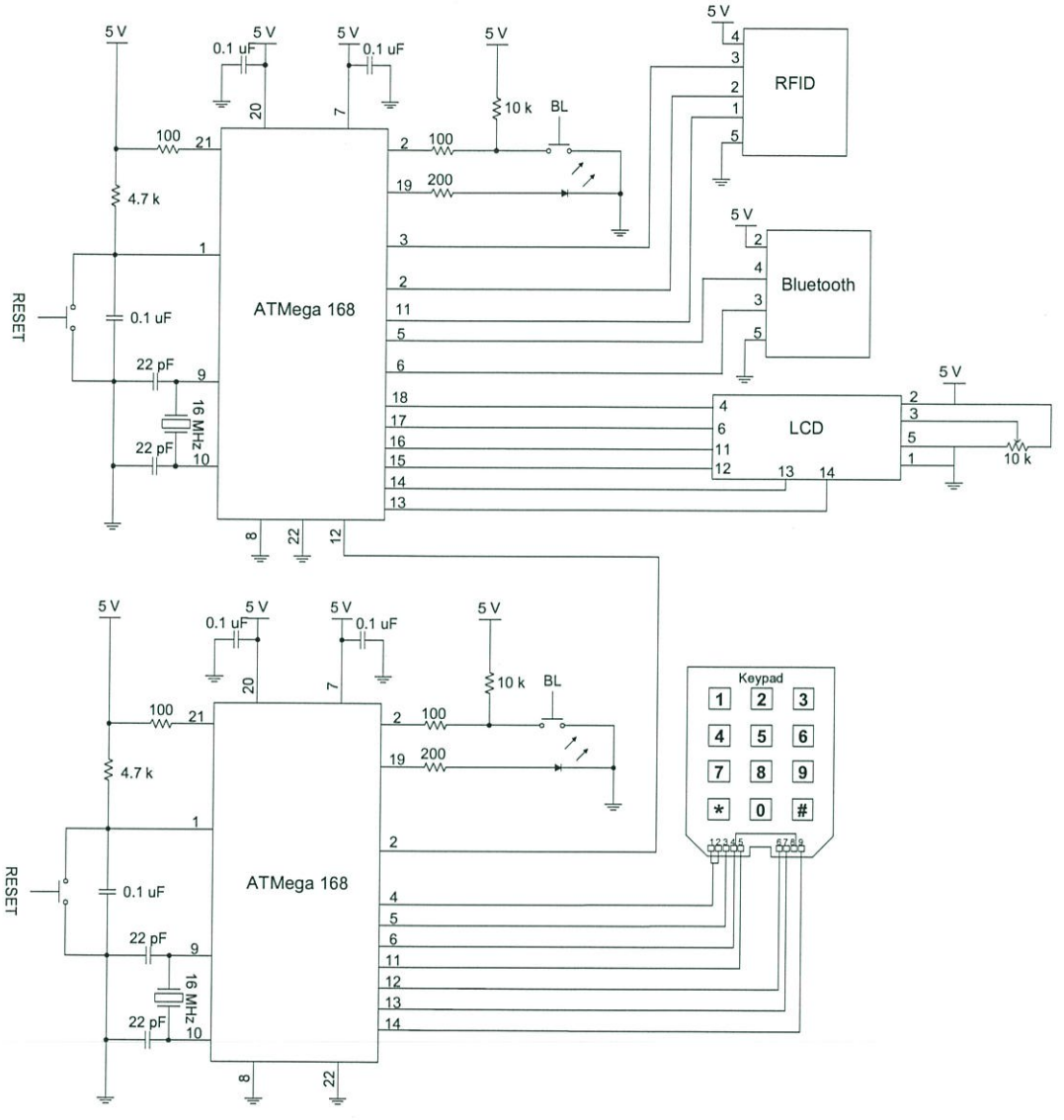
ระบบเก็บข้อมูลการใช้งาน และติดตามรถยนต์บรรทุก เริ่มต้นการทำงานด้วยการยืนยันตัวตนผ่านระบบอาร์เอฟไอดี หรือการใช้แป้นกดในการส่งข้อมูลไอดีของผู้ขับไปยังฐานข้อมูลเพื่อทำการตรวจสอบความถูกต้อง โดยจะแสดงผลการยืนยันตัวตนผ่านทางจอแอลซีดี เพื่อทำการเริ่มต้นระบบเก็บข้อมูลการใช้งาน และติดตามรถยนต์บรรทุก ซึ่งข้อมูลที่จะทำการจัดเก็บนั้นประกอบด้วย ไฟเลี้ยวซ้าย - ขวา ไฟหน้า ไฟหลัง ไฟถอยหลัง และระบบเบรก ด้วยเซนเซอร์ โดยจะใช้ไมโครคอนโทรลเลอร์ในการส่งข้อมูลผ่านบลูธูทโมดูลไปยังโทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์ และเก็บข้อมูลการติดตามด้วยระบบจีพีเอสภายในโทรศัพท์มือถือ ซึ่งข้อมูลจะถูกจัดเก็บอยู่ในฐานข้อมูล และแสดงผลผ่านทางหน้าเว็บเพจ โดยภาพรวมของระบบจะแสดงดังรูปที่ 3.1

3.1.1 ส่วนยืนยันตัวตน

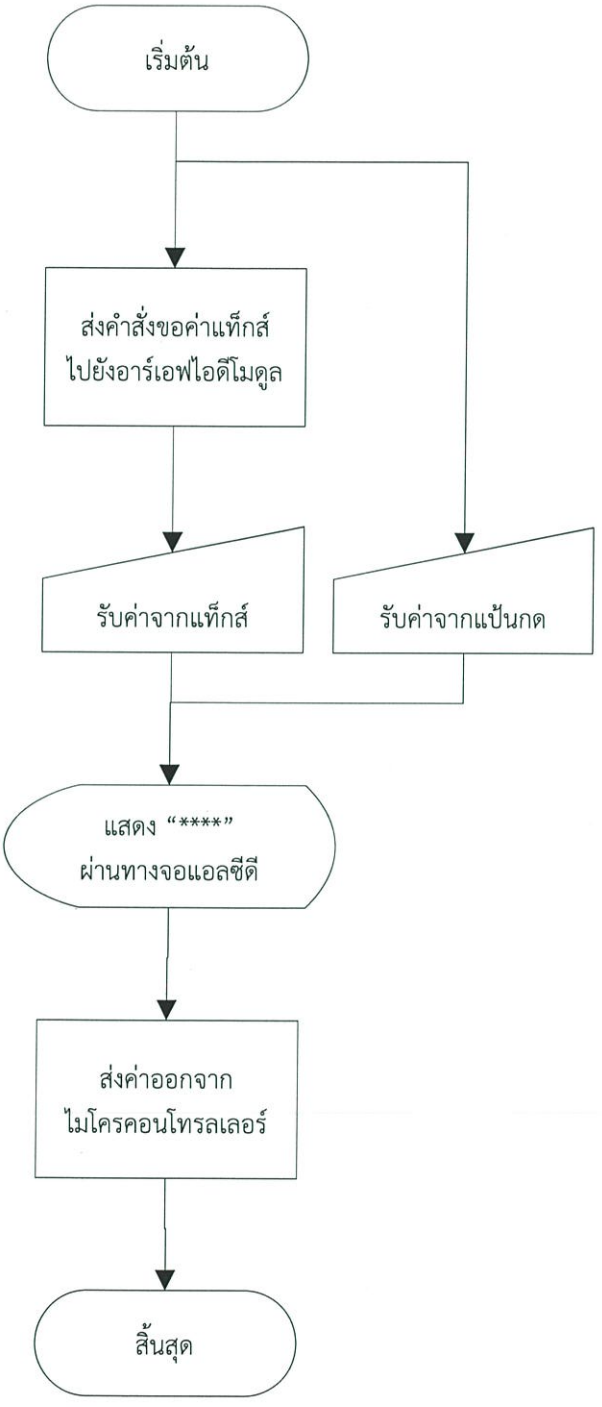
ในการออกแบบจะออกแบบให้ระบบสามารถรองรับทั้งการยืนยันตัวตนด้วยระบบอาร์เอฟไอดี และแป้นกด โดยใช้ไมโครคอนโทรลเลอร์เป็นตัวประมวลผลค่าไอดีที่รับมาจากทั้งสองระบบ จากนั้นนำไปเปรียบเทียบกับข้อมูลในฐานข้อมูลเพื่อตรวจสอบว่าไอดีของผู้ขับถูกต้องหรือไม่ โดยฐานข้อมูลจะทำการตอบกลับมายังไมโครคอนโทรลเลอร์ให้แสดงผลผ่านทางหน้าจอแอลซีดี โดยจะมีรูปแบบการต่ออุปกรณ์ของทั้งระบบ ดังรูปที่ 3.2 และมีโฟลว์ชาร์ตแสดงการทำงานของระบบดังรูปที่ 3.3 โดยเมื่อมีการแตะแท็กส์ หรือมีการกดไอดีผู้ขับที่แป้นกดจะแสดงค่า “****” ผ่านทางจอแอลซีดี จากนั้นข้อมูลไอดีจะถูกส่งไปตรวจสอบที่ฐานข้อมูล ฐานข้อมูลจะทำการตอบกลับมายังไมโครคอนโทรลเลอร์ โดยไมโครคอนโทรลเลอร์จะทำการประมวลผลข้อมูลที่ได้รับมาจากฐานข้อมูล โดย ถ้าข้อมูลไอดีผู้ขับถูกก็จะส่งค่า “Y” กลับมายังไมโครคอนโทรลเลอร์ แล้วไมโครคอนโทรลเลอร์ก็จะทำการสั่งให้แสดงผลข้อความว่า “OK” บนหน้าจอแอลซีดี แต่ถ้าข้อมูลที่ได้รับผิดฐานข้อมูลจะส่งค่า “N” กลับมายังไมโครคอนโทรลเลอร์ ไมโครคอนโทรลเลอร์ก็จะทำการสั่งให้แสดงผลข้อความว่า “Try again” บนหน้าจอแอลซีดี ซึ่งเมื่อไมโครคอนโทรลเลอร์สั่งการให้แสดงผลบนหน้าจอแอลซีดีว่า “OK” ก็จะเป็นการสิ้นสุดการทำงานของส่วนยืนยันตัวตน



รูปที่ 3.1 บล็อกไดอะแกรมของระบบเก็บข้อมูลการใช้งานและติดตามรถยนต์บรรทุก



รูปที่ 3.2 การต่ออุปกรณ์ทั้งหมดของส่วนยืนยันตัวตน

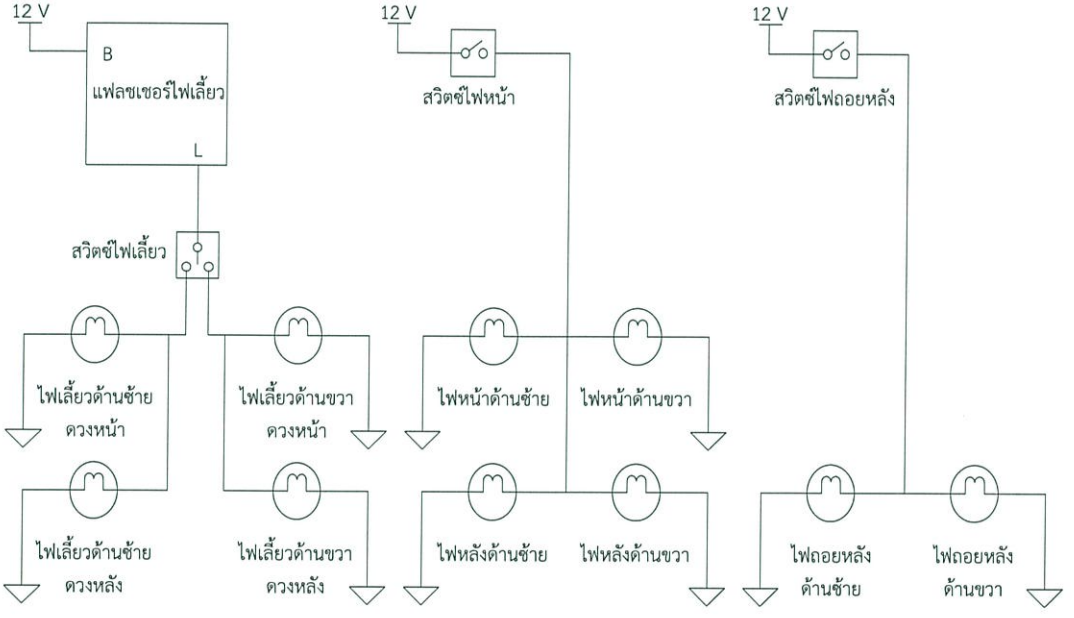


รูปที่ 3.3 โฟลว์ชาร์ตแสดงขั้นตอนการทำงานของส่วนยืนยันตัวตน

3.1.2 ชุดตรวจจับการทำงานของหลอดไฟและระบบเบรก

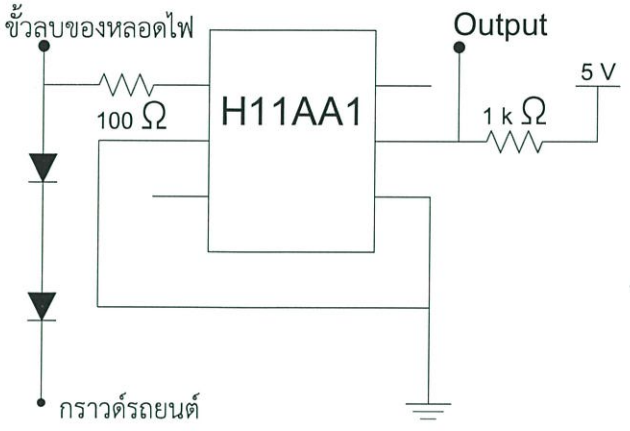
3.1.2.1 ชุดตรวจจับการทำงานของหลอดไฟ

ในการออกแบบชุดตรวจจับการทำงานของหลอดไฟนั้นผู้จัดทำได้ทำการจำลองชุดระบบสัญญาณไฟของรถยนต์ที่มีหลอดไฟจำนวน 10 หลอดประกอบไปด้วย หลอดไฟหน้า 2 หลอด หลอดไฟเลี้ยวด้านซ้าย 2 หลอด หลอดไฟเลี้ยวด้านขวา 2 หลอด หลอดไฟหลัง 2 หลอด และหลอดไฟสัญญาณขณะถอยหลัง 2 หลอด ซึ่งวงจรภายในชุดจำลองระบบสัญญาณไฟรถยนต์สามารถแสดงได้ดังรูปที่ 3.4



รูปที่ 3.4 วงจรภายในชุดจำลองระบบสัญญาณไฟรถยนต์

ในการตรวจจับการทำงานของหลอดไฟจะใช้ชุดวงจรตรวจจับการทำงานของหลอดไฟซึ่ง 1 ชุดจะประกอบด้วยวงจรตรวจจับหลอดไฟ 5 ตัวโดยวงจรตรวจจับการทำงานของหลอดไฟ 1 ตัวสามารถแสดงได้ดังรูปที่ 3.5 ซึ่งหลักการของตัวตรวจจับการทำงานของหลอดไฟแต่ละตัวนั้นจะนำเอาไดโอดไปต่ออนุกรมกับหลอดไฟแต่ละหลอด

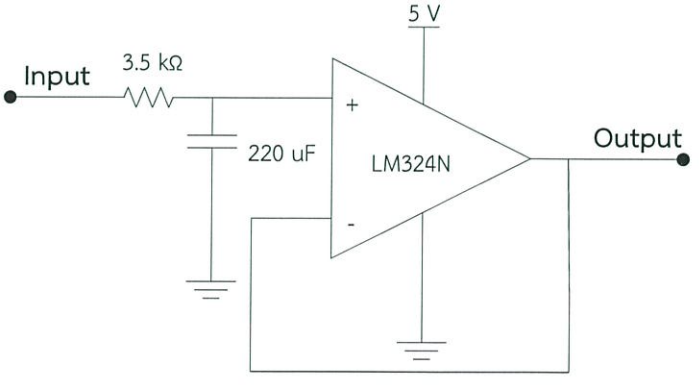


รูปที่ 3.5 วงจรตรวจจับหลอดไฟ 1 ตัว

จากวงจรในรูปที่ 3.5 ค่าเอาต์พุตที่ได้จะมีสถานะเป็น HIGH (5 V) เมื่อหลอดไฟยังไม่ทำงาน และเมื่อหลอดไฟทำงานค่าเอาต์พุตจะเปลี่ยนเป็นสถานะ LOW (0 V)

สำหรับไฟหน้า ไฟหลัง และไฟสัญญาณถอยหลังสามารถใช้วงจรในรูปที่ 3.5 ในการตรวจจับการทำงานของหลอดไฟได้

สำหรับไฟเลี้ยวทั้งไฟเลี้ยวด้านซ้าย และไฟเลี้ยวด้านขวานั้นสัญญาณไฟจะติด และดับเป็นจังหวะจึงจำเป็นต้องทำการออกแบบวงจรกรองความถี่ต่ำผ่านเพื่อให้สัญญาณที่ได้เป็นสัญญาณที่มีสถานะเดียว เนื่องจากในสถานะที่หลอดไฟยังไม่ทำงานจะให้เอาต์พุตเป็น HIGH ต้องทำการผ่านอินเวิร์ทเตอร์ให้เป็น LOW ก่อนที่จะเข้าสู่วงจรกรองความถี่ต่ำผ่านซึ่งมีความถี่ตัดที่ 0.206 เฮิรตซ์ และนำสัญญาณที่ได้จากวงจรกรองความถี่ต่ำผ่านไปเข้าวงจรเปรียบเทียบแรงดัน ซึ่งวงจรกรองความถี่ต่ำผ่านสามารถแสดงได้ดังรูปที่ 3.6



รูปที่ 3.6 วงจรกรองความถี่ต่ำผ่าน

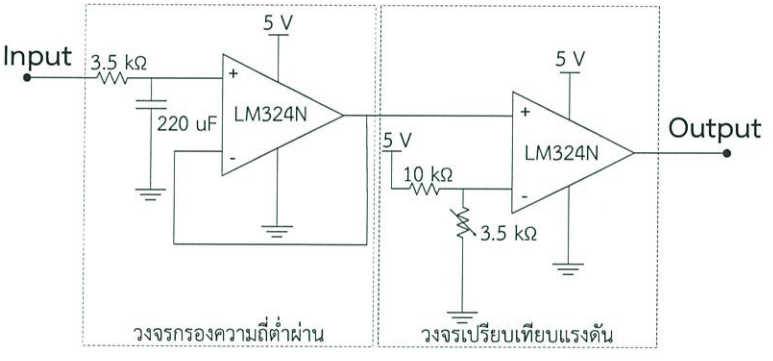
สมการในการหาค่าความถี่ตัดทางด้านต่ำสามารถแสดงได้ดังสมการที่ 3.1

$$f_C = \frac{1}{2\pi RC} \tag{3.1}$$

แทนค่า $R = 3.5k\Omega$ และ $C = 220\mu F$ จะได้

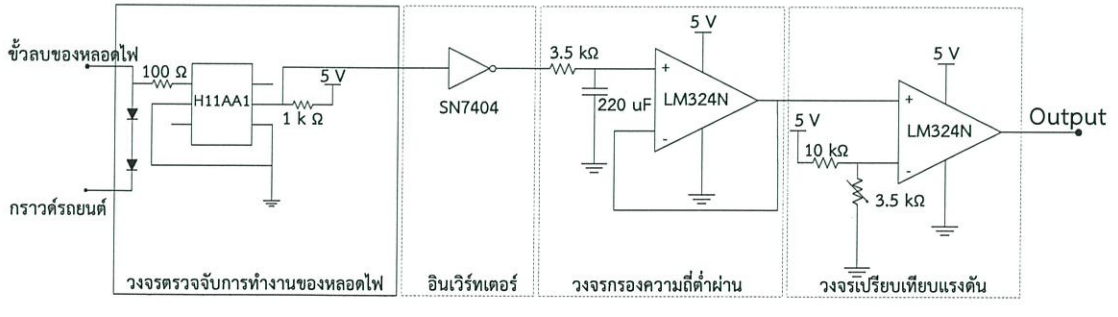
$$f_C = \frac{1}{2\pi(3500)(220\times10^{-6})} = 0.206Hz \tag{3.2}$$

โดยที่การต่อกันของวงจรกรองความถี่ต่ำผ่าน และวงจรเปรียบเทียบแรงดันสามารถแสดงได้ดังรูปที่ 3.7



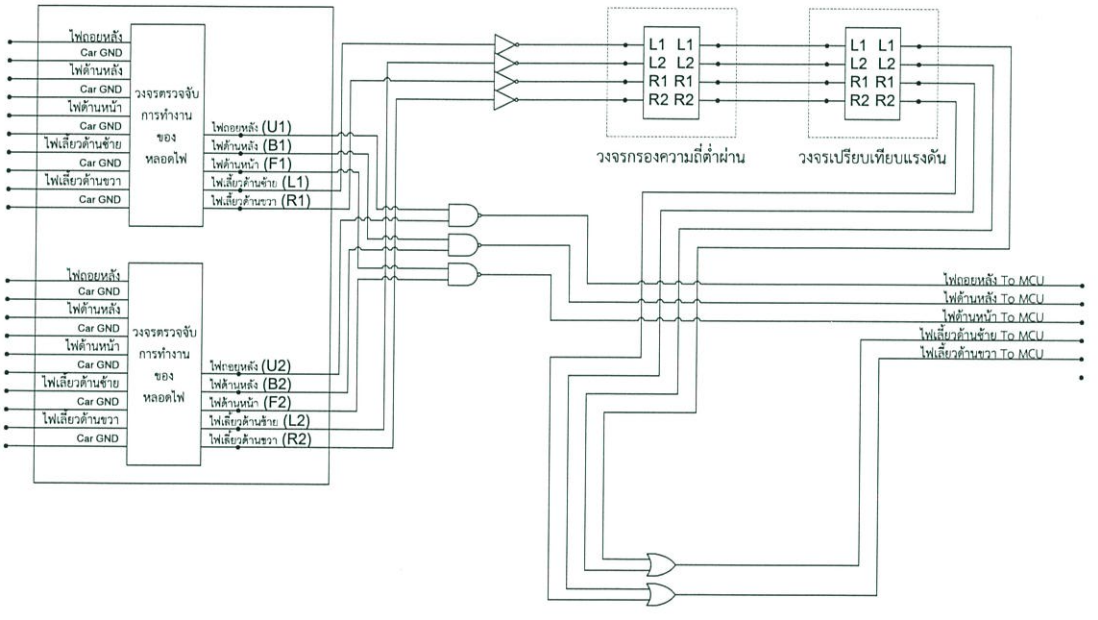
รูปที่ 3.7 การต่อกันของวงจรกรองความถี่ต่ำผ่าน และวงจรเปรียบเทียบแรงดัน

ดังนั้นชุดวงจรในการตรวจจับไฟเลี้ยวด้านซ้าย และไฟเลี้ยวด้านขวาจะประกอบไปด้วย วงจรตรวจจับการทำงานของหลอดไฟ อินเวิร์ทเตอร์ วงจรกรองความถี่ต่ำผ่าน และวงจรเปรียบเทียบแรงดัน โดยชุดการตรวจจับไฟเลี้ยวด้านซ้าย และไฟเลี้ยวด้านขวาสามารถแสดงได้ดังรูปที่ 3.8



รูปที่ 3.8 วงจรในการตรวจจับไฟเลี้ยว

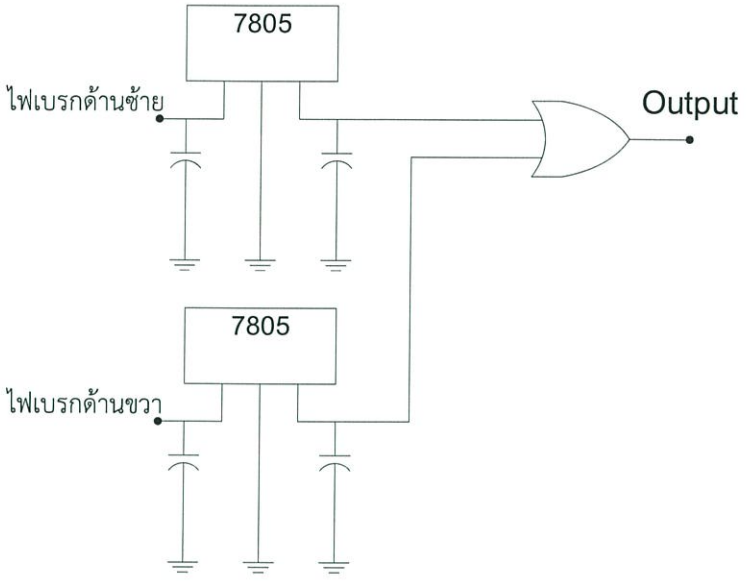
เนื่องจากหลอดไฟแต่ละชนิด (ไฟหน้า ไฟเลี้ยวด้านซ้าย ไฟเลี้ยวด้านขวา ไฟหลัง และไฟสัญญาณถอยหลัง) มีจำนวนชนิดละ 2 หลอดจึงใช้เงื่อนไขว่าถ้ามีหลอดไฟทำงานอย่างน้อย 1 หลอดจะถือว่าหลอดไฟชนิดนั้นทำงานด้วยวงจรลอจิกเกท โดยนำสัญญาณที่ออกจากวงจรตรวจจับการทำงานของหลอดไฟของหลอดไฟหน้า หลอดไฟหลัง และหลอดไฟสัญญาณถอยหลังไปเข้าสู่วงจรแนนเกทด้วยไอซีเบอร์ 7400 และสัญญาณที่ออกจากชุดวงจรในการตรวจจับไฟเลี้ยวด้านซ้าย และขวาไปเข้าสู่แอนเกทด้วยไอซีเบอร์ 7432 ซึ่งการต่อวงจรตรวจจับการทำงานของหลอดไฟเข้ากับวงจรลอจิกเกทสามารถแสดงได้ดังรูปที่ 3.9



รูปที่ 3.9 การต่อชุดตรวจจับการทำงานของหลอดไฟ

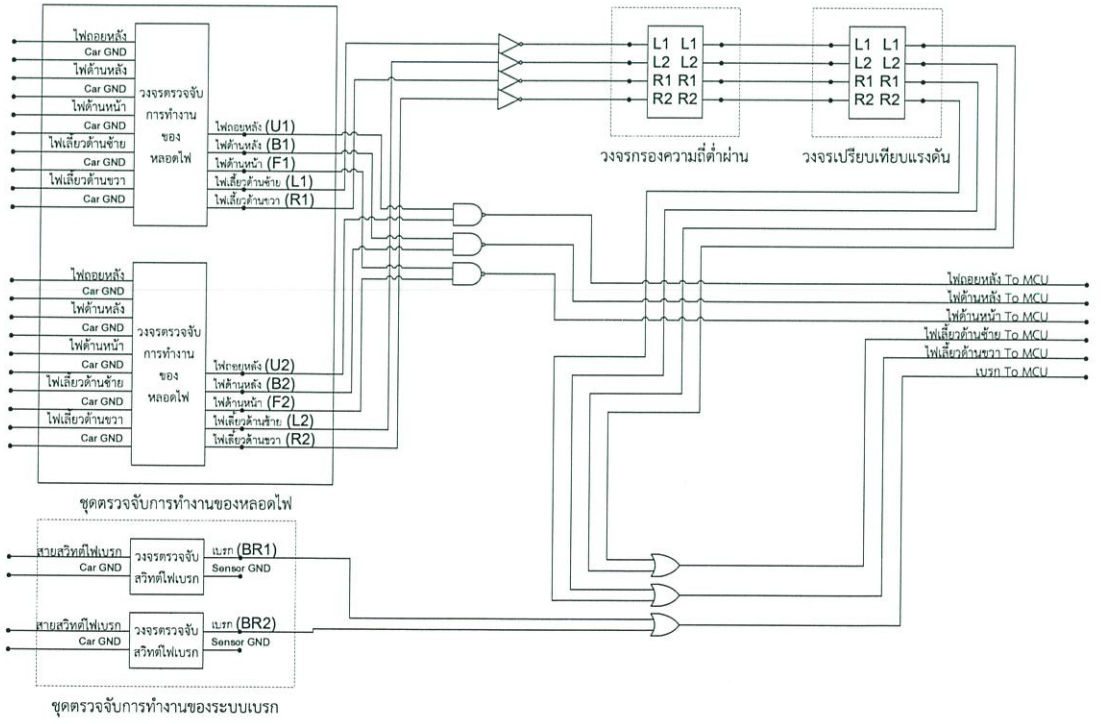
3.1.2.2 ชุดตรวจจับการทำงานของระบบเบรก

ในการตรวจจับการทำงานของเบรกนั้นจะตรวจจับจากการทำงานของสวิตซ์ไฟเบรกซึ่งการทำงานของสวิตซ์ไฟเบรกนั้นจะเป็นเมื่อเหยียบเบรกที่แป้นเบรก แป้นเบรกจะไปกดสวิตซ์ไฟเบรกทำให้ไฟเบรกติด โดยไฟเบรกนั้นจะติดเมื่อมีแรงดัน 12 โวลต์ ตกคร่อม ในการตรวจจับการทำงานคือนำแรงดันที่ตกคร่อมหลอดไฟเบรกมาลดระดับแรงดันให้เหลือ 5 โวลต์ ด้วยไอซีเรกูเลเตอร์ 7805 และนำสัญญาณไปเข้าแอนเกท ซึ่งชุดตรวจจับการทำงานของเบรกสามารถแสดงได้ ดังรูปที่ 3.10



รูปที่ 3.10 ชุดตรวจจับการทำงานของเบรก

เมื่อนำชุดตรวจจับการทำงานของหลอดไฟ และชุดตรวจจับการทำงานของเบรกมารวมกันสามารถแสดงได้ดังรูปที่ 3.11



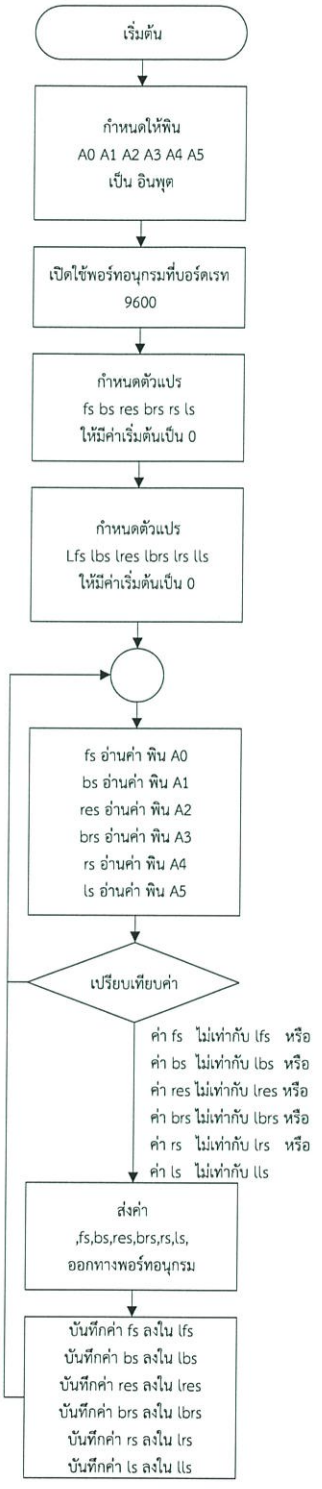
รูปที่ 3.11 ชุดตรวจจับการทำงานของหลอดไฟ และระบบเบรก

3.1.3 โปรแกรมตรวจจับการทำงานของชุดตรวจจับการทำงานของหลอดไฟและระบบเบรก

โปรแกรมจะแสดงผลข้อมูลเป็นแพ็กเกจโดยใน 1 แพ็กเกจจะมีข้อมูล 6 ชนิดคือ ไฟหน้า ไฟเลี้ยวด้านซ้าย ไฟเลี้ยวด้านขวา ไฟหลัง ไฟถอยหลัง และระบบเบรกดังรูปที่ 3.12 โดยการทำงานของเซนเซอร์จะแสดงในรูปของค่า 0 และ 1 ซึ่ง 0 คือไม่มีการใช้งานระบบไฟ หรือระบบเบรก และ 1 คือมีการใช้งานระบบไฟ หรือระบบเบรกและจะแสดงผลโดยใช้ตัวอักษร ‘,’ ขั้นระหว่างข้อมูลเช่น ,0,0,0,0,0,0, ซึ่งขั้นตอนการทำงานของโปรแกรมสามารถแสดงได้ดังรูปที่ 3.13

ไฟหน้า	ไฟเลี้ยว ด้านซ้าย	ไฟเลี้ยว ด้านขวา	ไฟหลัง	ไฟ ถอยหลัง	ระบบ เบรก

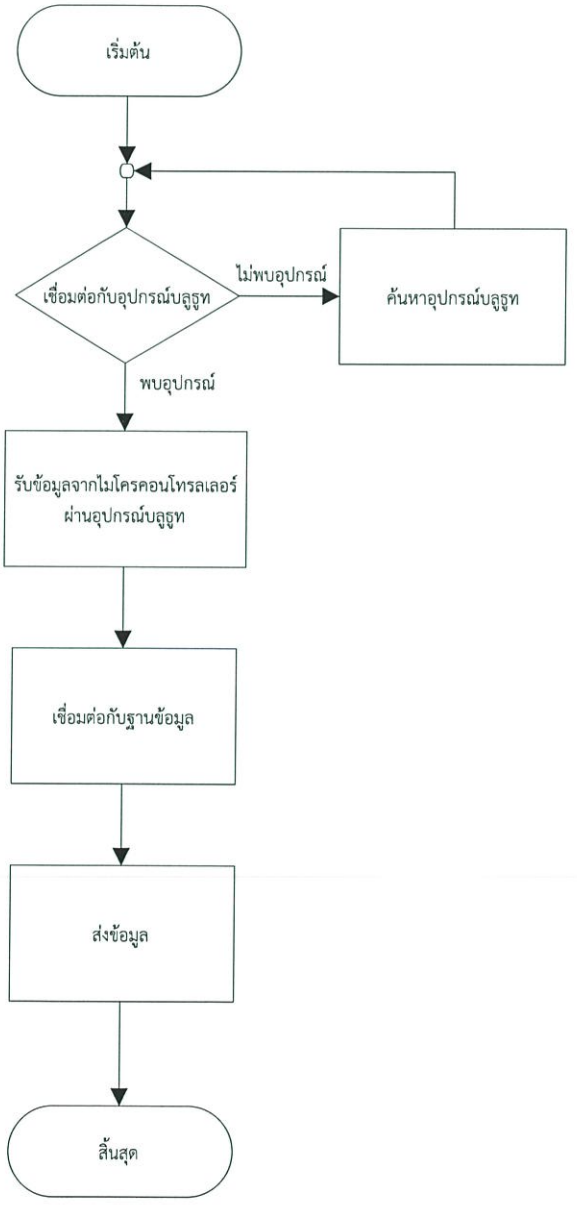
รูปที่ 3.12 การเรียงของข้อมูลในแพ็กเกจ



รูปที่ 3.13 ขั้นตอนการทำงานของโปรแกรมในการรับค่าจากชุดตรวจจับการทำงานของหลอดไฟ และระบบเบรก

3.1.4 โทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์

แอพพลิเคชันได้ถูกจัดทำขึ้นเพื่อรองรับการเชื่อมต่อบลูธูท และการส่งข้อมูลเข้าฐานข้อมูลโดยใช้โปรแกรม eclipse โดยมีขั้นตอนการทำงานของโปรแกรมเป็นไปตามโฟลว์ชาร์ตดังรูปที่ 3.14



รูปที่ 3.14 โฟลว์ชาร์ตของโปรแกรมในแอพพลิเคชันบนโทรศัพท์

จากโฟลว์ชาร์ตขั้นตอนการทำงานของโปรแกรม โปรแกรมที่ได้นั้นจึงมีส่วนประกอบดังนี้

3.1.4.1 การเชื่อมต่อกับอุปกรณ์บลูธูทโดยเรียกใช้คลาสที่ทำหน้าที่จัดการการทำงานของบลูธูทคือ Bluetooth Adapter โดยจะมีคำสั่ง getDefaultAdapter ซึ่งใช้ในการเปิดบลูธูทของโทรศัพท์มือถือ ถ้าโทรศัพท์มือถือไม่มีบลูธูทจะทำการปิดโปรแกรม

```
mBluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
    if (mBluetoothAdapter == null) {
        Toast.makeText(Main.this, "Bluetooth is not available",
Toast.LENGTH_SHORT).show();
        finish();
    }
```

3.1.4.2 การค้นหาอุปกรณ์และทำการเชื่อมต่อซึ่งจะทำภายในแอคทิวิตีรองชื่อ SelectDevice โดยใช้การ Intent ส่งค่าผลลัพธ์กลับไปที่แอคทิวิตีหลัก โดยจะเริ่มจากปุ่มกดที่ใช้ค้นหาอุปกรณ์อื่นๆดังนี้

```
btnScan = (Button) findViewById(R.id.btnScan);
            btnScan.setOnClickListener(new OnClickListener() {
        public void onClick(View v) {

            if(mBtAdapter.isDiscovering()) {
                    mBtAdapter.cancelDiscovery();
                    btnScan.setText("Scan for devices");
            } else {
                doDiscovery();
                btnScan.setText("Scanning...");
            }
        }
    });
```

โดยจะเรียกไปที่ฟังก์ชั่น doDiscovery เพื่อทำการค้นหาอุปกรณ์แต่ถ้ากดปุ่มในขณะที่บลูธูทกำลังค้นหาอยู่จะเป็นการหยุดค้นหา โดยฟังก์ชั่น doDiscovery นั้นจะทำการเคลียร์ List View แล้วดึงอุปกรณ์ที่เคยเชื่อมต่อมาแสดงผ่านฟังก์ชั่น mDevicesArrayAdapter จากนั้นทำการค้นหาอุปกรณ์เพิ่มเติมอื่นๆ ดังนี้

```
private void doDiscovery() {
        mDevicesArrayAdapter.clear();
          for (BluetoothDevice device : mBtAdapter.getBondedDevices()) {
          mDevicesArrayAdapter.add(device.getName() + "\n" + device.getAddress());
        }

        if (mBtAdapter.isDiscovering()) {
            mBtAdapter.cancelDiscovery();
        }
        mBtAdapter.startDiscovery();
    }
```

mDeviceArrayAdapter จะทำการสร้าง List View ของอุปกรณ์ที่พบใหม่แต่ถ้าไม่พบจะขึ้นว่า No devices have been paired ดังนี้

```
mDevicesArrayAdapter = new ArrayAdapter<String>(SelectDevice.this,
android.R.layout.simple_list_item_1);
        ListView pairedListView = (ListView) findViewById(R.id.lvDevice);
        pairedListView.setAdapter(mDevicesArrayAdapter);
        pairedListView.setOnItemClickListener(mClickListener);
        mBtAdapter = BluetoothAdapter.getDefaultAdapter();
        Set<BluetoothDevice> pairedDevices = mBtAdapter.getBondedDevices();
        if (pairedDevices.size() > 0) {
            for (BluetoothDevice device : pairedDevices) {
                    mDevicesArrayAdapter.add(device.getName() + "\n" +
device.getAddress());
            }
        } else {
            mDevicesArrayAdapter.add("No devices have been paired");
        }
        IntentFilter filter = new IntentFilter(BluetoothDevice.ACTION_FOUND);
        this.registerReceiver(mReceiver, filter);
        filter = new IntentFilter(BluetoothAdapter.ACTION_DISCOVERY_FINISHED);
        this.registerReceiver(mReceiver, filter);
```

3.1.4.3 การรับข้อมูลจากไมโครคอนโทรลเลอร์จะอยู่ในแอคทิวิตีหลักโดยจะอยู่ในส่วนของการ handlerMessage ที่ได้รับจากแอคทิวิตี BluetoothService โดยเมื่อมีข้อมูลเข้ามา BluetoothService จะ Intent สถานะ มายังแอคทิวิตีหลัก โดยในแอคทิวิตีหลักจะตรวจสอบสถานะต่างๆ ที่ BluetoothService ส่งมาโดยสถานะ MESSAGE_READ นั้นจะใช้อ่านข้อมูลโดยการใช้อาร์เรย์อ่านข้อมูลทีละไบต์ และจัดการข้อความเป็น String โดยใช้ฟังก์ชั่น StringBuilder ทำการโชว์ข้อความบนหน้าต่างหลักของส่วนติดต่อผู้ใช้งาน และทำการเรียกไปยังฟังก์ชั่น updateData เพื่อส่งข้อมูลเข้าสู่ฐานข้อมูลดังนี้

```
public final  Handler mHandler = new Handler() {
    public void handleMessage(Message msg) {
      switch (msg.what) {
      case MESSAGE_READ:
            txtShow = (TextView) findViewById(R.id.txtShow);
            StringBuilder sb = new StringBuilder();
            byte[] readBuf = (byte[]) msg.obj;
            String readData = new String(readBuf, 0, msg.arg1);
            sb.append(readData);
            int endOfLineIndex = sb.indexOf("\n");
            if(endOfLineIndex > 0){
                String sbprint = sb.substring(0, endOfLineIndex);
                sb.delete(0, sb.length());
                txtShow.setText(sbprint);
                Log.d(TAG, "Start BT Send");
                updateAuthen(sbprint);
                Log.d(TAG, "End BT Send");
            }
```

3.1.4.4 การอ่านค่าพิกัดจีพีเอสจะทำในฟังก์ชั่น onLocationChanged โดยจะกำหนดระยะทาง หรือระยะเวลาที่จะให้ทำการรับพิกัดจีพีเอสจากคำสั่ง requestLocationUpdates(LocationManager.GPS_PROVIDER, 60000, 100, this) โดย 60000 คือระยะเวลาเป็นวินาทีที่จะให้ทำการรับพิกัดและ 100 คือระยะทางเป็นเมตร โดยฟังก์ชั่น onLocationChanged นั้นจะทำงานเมื่อมีการเปลี่ยนค่าพิกัดโดยใช้คำสั่ง loc.getLatitude() ในการรับค่าละติจูด loc.getLongitude() ในการรับค่าลองจิจูด และ loc.getSpeed() ในการรับค่าความเร็ว จากนั้นทำการเก็บค่าดังกล่าวทั้งหมดในรูปแบบสตริง และเรียกไปยังฟังก์ชั่น updateGPS เพื่อส่งข้อมูลเข้าฐานข้อมูลดังนี้

```
public void onLocationChanged(Location loc) {
                // TODO Auto-generated method stub
                double Lat = loc.getLatitude();
                double Long = loc.getLongitude();
                double Spd = loc.getSpeed();
                String loc1 = Double.toString(Lat);
                String loc2 = Double.toString(Long);
                String loc3 = Double.toString(Spd);
                location = (String) (    loc1 + "|" + loc2 + "|" + loc3 + "|");
                lat.append("Send : " + location + "\n");
                Log.d(TAG, "Start gps Send");
                updateGPS(location);
                Log.d(TAG, "End gps Send");
}
```

3.1.4.5 การเชื่อมต่อกับฐานข้อมูลในฟังก์ชั่น updateData และ updateGPS นั้นจะทำการเรียกแอคทิวิตี ConnectServer โดยส่ง url ที่จะใช้ในการเชื่อมต่อไป และส่งค่าที่จะส่งไปยังฟังก์ชั่น addValue ที่อยู่ในแอคทิวิตี ดังนี้

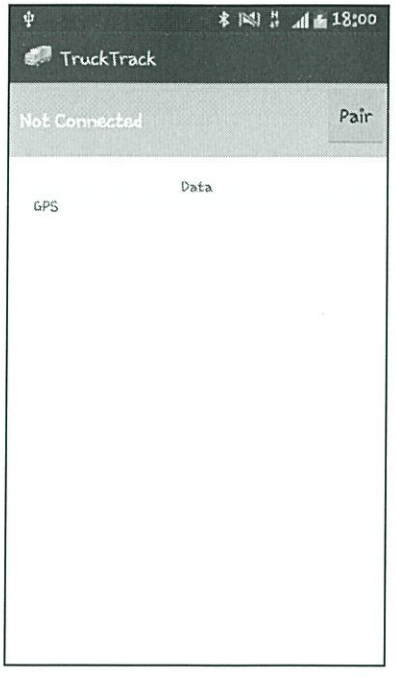
```
private void updateData(String Data){
        connectServer = new ConnectServer(this, "http://192.168.1.110/project/split.php");
        connectServer.addValue("Data", Data);
        connectServer.execute();
    }
```

3.1.4.6 การส่งข้อมูลในแอคทิวิตี ConnectServer นั้นจะมีฟังก์ชั่น addValue โดยจะนำค่าที่ส่งมาจากแอคทิวิตีหลักมาใส่ในตัวแปร nameValuePairs ซึ่งเป็นตัวแปรแบบรายการอาร์เรย์ (ArrayList) และทำการเชื่อมต่อกับเซิร์ฟเวอร์ด้วยฟังก์ชั่น doInBackground จากนั้นทำการส่งค่าในตัวแปร nameValuePairs ไปยังเซิร์ฟเวอร์ในรูปแบบ Post ดังนี้

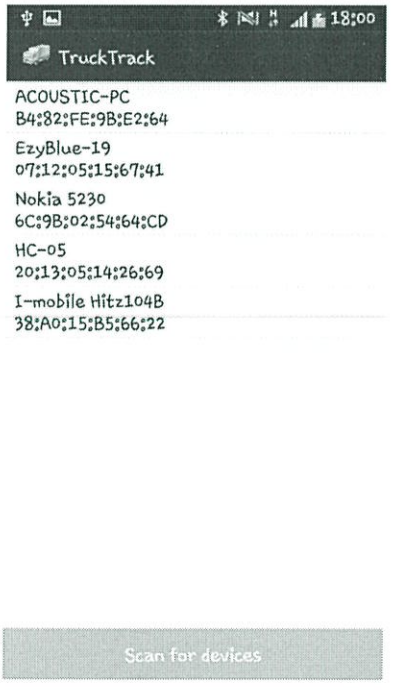
```
public void addValue(String key,String value){
      nameValuePairs.add(new BasicNameValuePair(key, value));
   }
protected String doInBackground(String... params) {
      InputStream is = null;
      String result = null;
       try {
         httppost.setEntity(new
```

```
UrlEncodedFormEntity(nameValuePairs,HTTP.UTF_8));
            HttpResponse response = httpclient.execute(httppost);
            HttpEntity entity = response.getEntity();
            is = entity.getContent();
            BufferedReader reader = new BufferedReader(new InputStreamReader(is,
"UTF-8"), 8);
            StringBuilder sb = new StringBuilder();
            String line = null;
            while ((line = reader.readLine()) != null) {
                sb.append(line + "\n");
            }
             is.close();
            result = sb.toString();
        } catch (ClientProtocolException e) {
            Log.e("ConnectServer", e.toString());
        } catch (IOException e) {
            Log.e("ConnectServer", e.toString());
        }
         return result;
    }
```

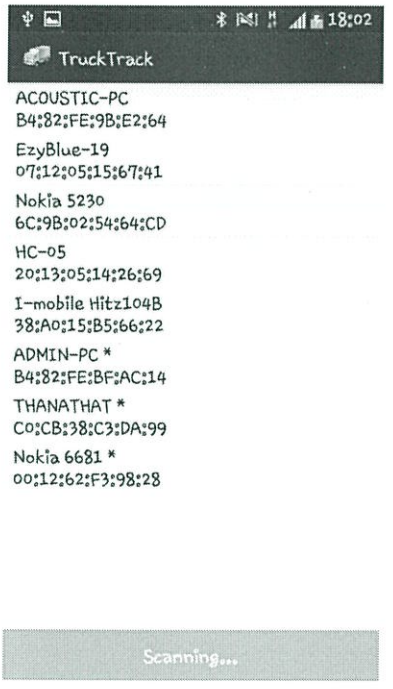
ทำการออกแบบหน้าต่างโดยจะแบ่งออกเป็น 3 หน้าคือ 1. หน้าหลักใช้สำหรับดูข้อมูลที่ได้รับจากไมโครคอนโทรลเลอร์ และมีปุ่มสำหรับเชื่อมต่ออุปกรณ์อุปกรณ์บลูธูท ดังรูปที่ 3.15 2. หน้าต่างที่ใช้แสดงอุปกรณ์บลูธูทที่จะทำการเชื่อมต่อ และมีปุ่มสำหรับค้นหาอุปกรณ์ที่ยังไม่เคยเชื่อมต่อดังรูปที่ 3.16 และ 3. หน้าต่างที่ใช้แสดงอุปกรณ์บลูธูทที่เปิดอยู่ในบริเวณนั้นดังรูปที่ 3.17 โดยในหน้าหลักจะทำการเชื่อมต่อโทรศัพท์มือถือกับบลูธูทโมดูลโดยกดที่ปุ่ม Pair ดังรูปที่ 3.15 เมื่อกดปุ่มแล้วจะไปหน้าต่างที่มีรายชื่ออุปกรณ์บลูธูทที่ได้เคยทำการเชื่อมต่อดังรูปที่ 3.16 ซึ่งถ้าไม่พบกับอุปกรณ์ที่ต้องการ ให้ทำการกดปุ่ม Scan for devices เมื่อกดปุ่มแล้วจะแสดงรายชื่ออุปกรณ์บลูธูทที่เปิดอยู่ในบริเวณนั้น ดังรูปที่ 3.17



รูปที่ 3.15 หน้าต่างหลักของส่วนติดต่อผู้ใช้งาน



รูปที่ 3.16 หน้าต่างแสดงรายชื่ออุปกรณ์บลูธูทที่เคยเชื่อมต่อ



รูปที่ 3.17 หน้าต่างแสดงรายชื่ออุปกรณ์บลูธูทที่เคยเชื่อมต่อและที่เปิดอยู่ในบริเวณนั้น

3.1.5 ฐานข้อมูล

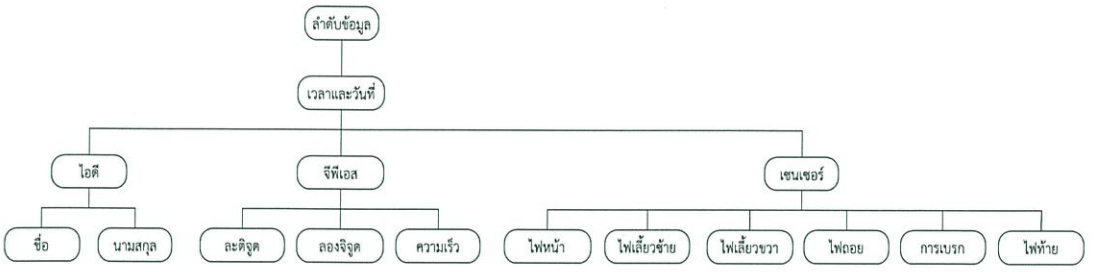
3.1.5.1 การออกแบบ

1) สิ่งที่ต้องการจัดเก็บ

ในปริญญานิพนธ์นี้ตัวแปรสำคัญต่างๆ ที่ต้องการจัดเก็บคือ 1. ลำดับที่ข้อมูล 2. ค่าไอดีจากแท็กส์ของผู้ใช้งานรถยนต์บรรทุก 3. ชื่อและนามสกุลของผู้ใช้งานของแท็กส์นั้นๆ 4. เวลาและวันที่ 5. ค่าพิกัดจีพีเอสและความเร็ว 6. ค่าเซนเซอร์คือ ไฟหน้า ไฟเลี้ยวซ้าย ไฟเลี้ยวขวา ไฟท้าย ไฟถอย และการเบรก

2) ความสัมพันธ์ของสิ่งที่ต้องการจัดเก็บ

ในการจัดเก็บข้อมูลต่างๆ นั้นจะต้องเรียงลำดับตามความสำคัญของข้อมูล คือ 1. ลำดับที่ข้อมูล 2. เวลา และวันที่ และ 3. ค่าพิกัดจีพีเอส ความเร็ว ค่าเซนเซอร์คือ ไฟหน้า ไฟเลี้ยว ไฟถอย และการเบรก ดังรูปที่ 3.18



รูปที่ 3.18 ลำดับความสำคัญของสิ่งที่ต้องการจัดเก็บ

3) ลักษณะหรือชนิดของสิ่งที่ต้องการจัดเก็บ

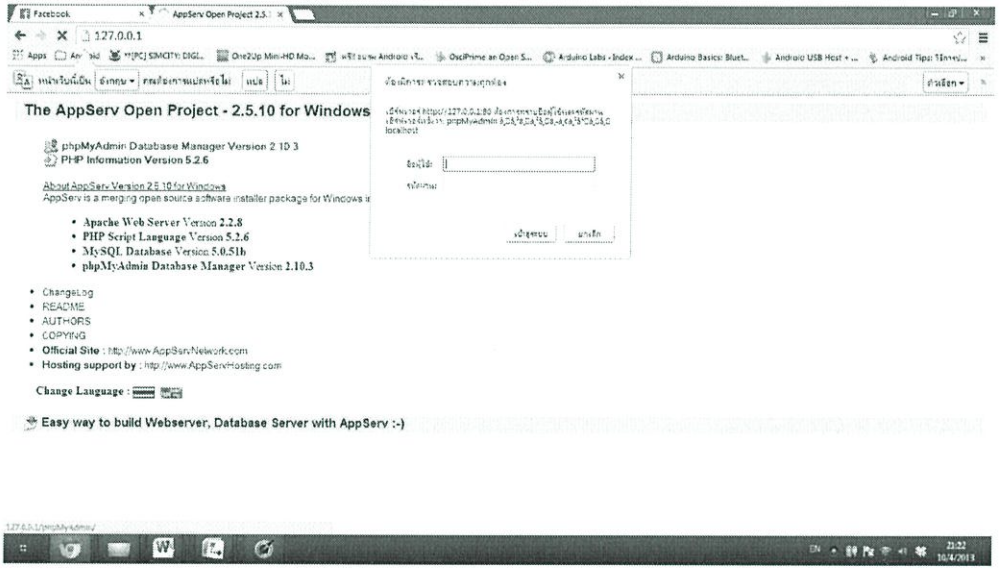
ชนิดของข้อมูลนั้นเป็นสิ่งที่สำคัญ ดังนั้นในการออกแบบจะต้องกำหนดชนิดของข้อมูลให้สอดคล้องกับข้อมูลที่ต้องการจัดเก็บ และกำหนดความของข้อมูลมีความยาวเพียงพอ โดยในที่นี้จะกำหนดดังตารางที่ 3.1

ตารางที่ 3.1 ชนิดและความยาวของข้อมูลที่ต้องการจัดเก็บ

สิ่งที่ต้องการจัดเก็บ	ชนิดของข้อมูล	ความยาว (ไบต์)
ลำดับข้อมูล	int	10
เวลาและวันที่	varchar	40
ไอดี	varchar	100
ชื่อ	text	10
นามสกุล	text	30
ละติจูด	double	10
ลองจิจูด	double	10
ความเร็ว	float	10
ไฟหน้า	varchar	10
ไฟเลี้ยวซ้าย	varchar	10
ไฟเลี้ยวขวา	varchar	10
ไฟถอย	varchar	10
ไฟท้าย	varchar	10
การเบรก	varchar	10

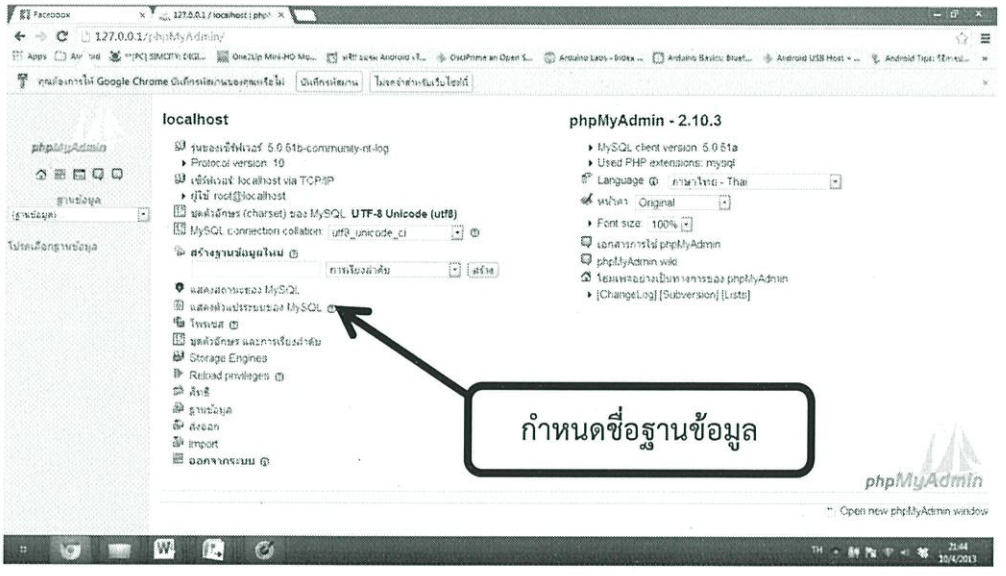
3.1.5.2 การสร้าง

1) เปิดเว็บบราวเซอร์ และเข้าไปยังแอพเซิร์ฟผ่าน ยูอาร์แอล localhost หรือไอพีแอดเดรส 127.0.0.1 จากนั้นเข้าไปสร้างฐานข้อมูลโดยไปที่ phpMyAdmin Database Manager ซึ่งต้องมีการตรวจสอบความถูกต้องโดยการใส่ชื่อผู้ใช้ และรหัสผ่านที่ได้กำหนดไว้ตอนติดตั้งโปรแกรมดังรูปที่ 3.19



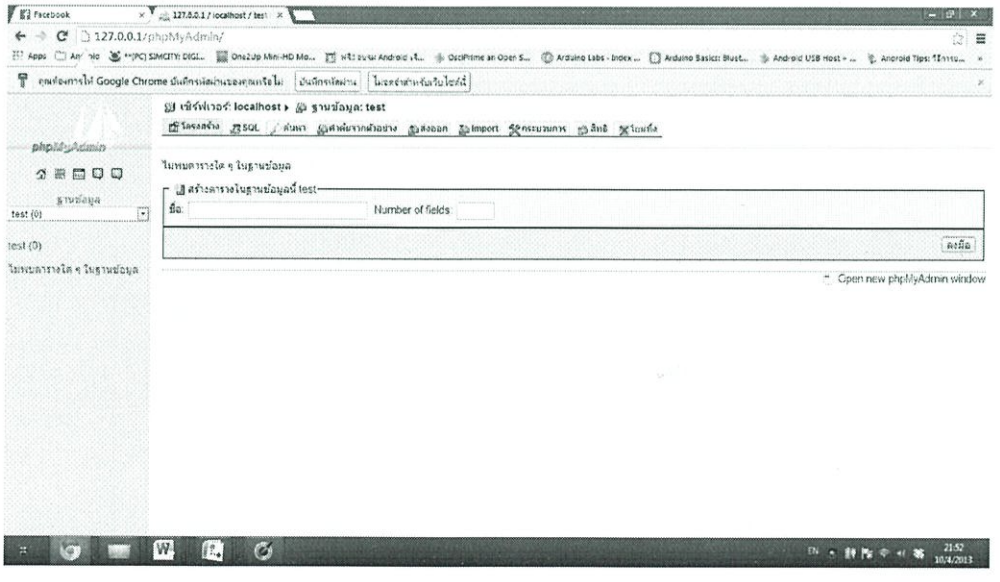
รูปที่ 3.19 หน้าต่างที่จะทำการเข้าสร้างฐานข้อมูล

2) เมื่อเข้าสู่ระบบเสร็จแล้วจะพบหน้าต่าง ให้ทำการกำหนดชื่อฐานข้อมูล และค่าต่างๆ ดังรูปที่ 3.20



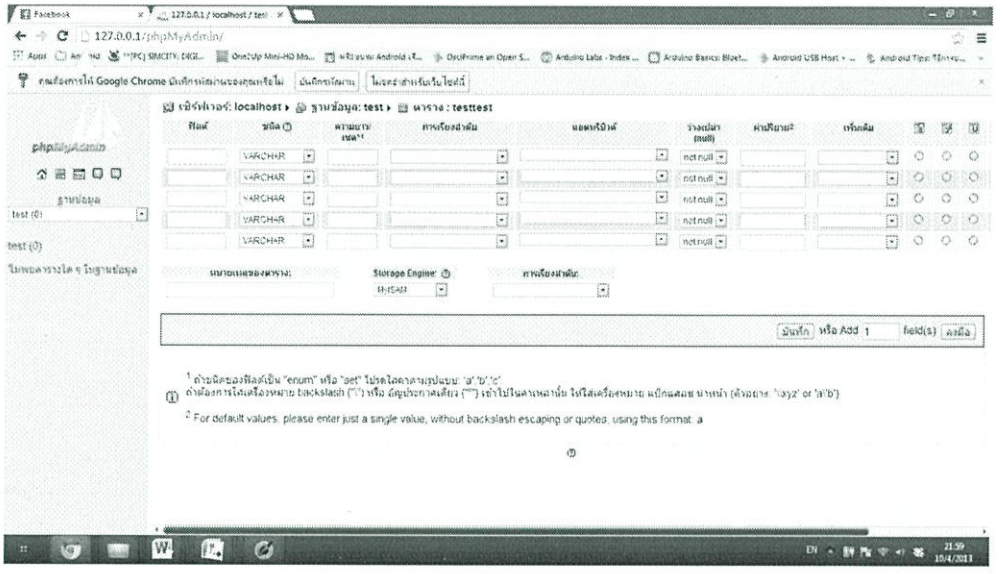
รูปที่ 3.20 หน้าต่างหลักจากกดปุ่มเข้าสู่ระบบ

3) เลือกฐานข้อมูลที่เราได้สร้างขึ้นจากรายการทางด้านซ้ายเพื่อทำการสร้างตารางฐานข้อมูลโดยการกำหนดชื่อ และจำนวนฟิลด์ที่เราต้องการ ดังรูปที่ 3.21



รูปที่ 3.21 หน้าต่างสำหรับกำหนดชื่อและจำนวนฟิลด์ของฐานข้อมูล

4) กำหนดชื่อฟิลด์ (ตัวแปร) ชนิดของข้อมูลที่จะจัดเก็บ และความยาวของข้อมูล ดังรูปที่ 3.22



รูปที่ 3.22 หน้าต่างสำหรับกำหนดชื่อฟิลด์ ชนิดของข้อมูลที่จะจัดเก็บ และความยาวของข้อมูล

ในปริญญานิพนธ์นี้กำหนดให้ฐานข้อมูลชื่อ truck และสร้างตารางฐานข้อมูลทั้งหมด 3 ตารางโดยตารางแรกชื่อ rfid ใช้เก็บค่ายูไอดีของแท็กส์เพื่อใช้ในการยืนยันตัวตน ซึ่งจะมีฟิลด์ชื่อ ID เพื่อใช้เก็บค่ายูไอดี name ใช้เก็บค่าชื่อของผู้ใช้งานรถยนต์บรรทุกและ surname ใช้เก็บค่านามสกุลของผู้ใช้งานรถยนต์บรรทุก ตารางที่สองชื่อ truckbb ใช้เก็บค่าข้อมูลเซนเซอร์ โดยจะจัดเก็บข้อมูลทั้งหมด 8 ข้อมูลดังนี้

- ตัวแปร time คือข้อมูลวันที่และเวลาที่ข้อมูลถูกส่งเข้ามายังฐานข้อมูล
- ตัวแปร ID คือค่ายูไอดีของแท็กส์ที่ถูกส่งเข้ามาเพื่อทำการยืนยันตัวตน
- ตัวแปร FL คือข้อมูลไฟหน้าที่เซนเซอร์สามารถวัดได้ และส่งเข้ามายังฐานข้อมูล
- ตัวแปร LL คือข้อมูลไฟเลี้ยวซ้ายที่เซนเซอร์สามารถวัดได้ และส่งเข้ามายังฐานข้อมูล
- ตัวแปร RL คือข้อมูลไฟเลี้ยวขวาที่เซนเซอร์สามารถวัดได้ และส่งเข้ามายังฐานข้อมูล
- ตัวแปร Rev คือข้อมูลไฟถอยที่เซนเซอร์สามารถวัดได้ และส่งเข้ามายังฐานข้อมูล

- ตัวแปร BL คือข้อมูลไฟท้ายที่เซนเซอร์สามารถวัดได้ และส่งเข้ามายังฐานข้อมูล
- ตัวแปร BR คือข้อมูลการเบรกซ้ายที่เซนเซอร์สามารถวัดได้ และส่งเข้ามายังฐานข้อมูล

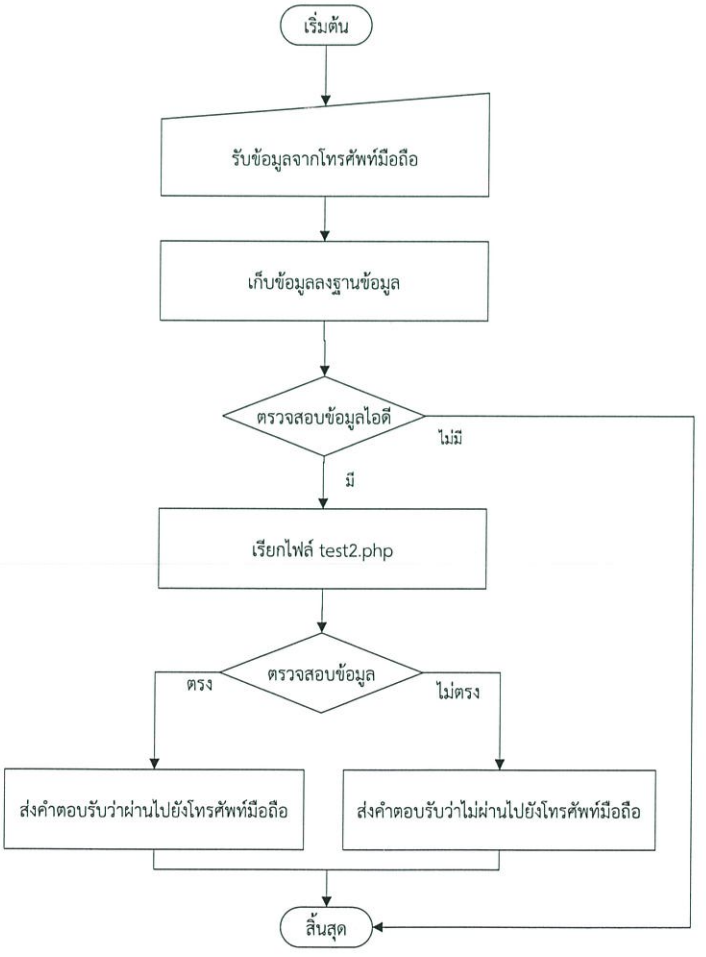
ตารางที่สามชื่อ truckgps ใช้เก็บค่าพิกัดจีพีเอสโดยจะจัดเก็บข้อมูลทั้งหมด 4 ข้อมูลดังนี้

- ตัวแปร time คือข้อมูลวันที่และเวลาที่ข้อมูลถูกส่งเข้ามายังฐานข้อมูล
- ตัวแปร Lat คือข้อมูลละติจูด (Latitude) ที่วัดได้จากจีพีเอสในโทรศัพท์มือถือ และส่งเข้ามายังฐานข้อมูล
- ตัวแปร Lon คือข้อมูลลองจิจูด (Longitude) ที่วัดได้จากจีพีเอสในโทรศัพท์มือถือ และส่งเข้ามายังฐานข้อมูล
- ตัวแปร Spd คือข้อมูลความเร็ว (Speed) ที่คำนวณจากแอพพลิเคชันในโทรศัพท์มือถือและส่งเข้ามายังฐานข้อมูล

โปรแกรมที่ใช้ในการรับข้อมูลที่ส่งมายังเซิร์ฟเวอร์นั้นแบ่งออกเป็นสองส่วนคือส่วนแรกจะรับข้อมูลการยืนยันตัวตน และเซนเซอร์ผ่านไฟล์ split.php ซึ่งจะทำการตัดชุดข้อมูลที่ถูกส่งมาโดยใช้คำสั่ง explode และทำการบันทึกลงในตาราง truckbb จากนั้นจะทำการตรวจสอบว่ามีค่าไอดีถูกส่งมาหรือไม่ ถ้ามีให้ทำการเรียกไฟล์ test2.php เพื่อตรวจสอบค่ายูไอดีที่ได้รับมากับค่าที่มีอยู่ในตารางอาร์เอฟไอดี แล้วตอบกลับไปยังโทรศัพท์มือถือโดยมีขั้นตอนการทำงานดังรูปที่ 3.23 จากโฟลว์ชาร์ตจะได้โค้ดในไฟล์ split.php ดังนี้

```
<?php
$text=$_REQUEST['Data'];
$time = date("d/m/y (H:i:s)");
$data = explode("," , $text);
$ID = $data[0];
$FL = $data[1];
$LL = $data[2];
$RL = $data[3];
$BL = $data[4];
$Rev = $data[5];
$BR = $data[6];
$con=mysql_connect('localhost','root','1234')or die(mysql_error());
mysql_select_db('truck')or die(mysql_error());
mysql_query("SET NAMES UTF8");
```

```
        date_default_timezone_set("Asia/Bangkok");
        $sql="INSERT INTO truckbb (ID, time, FL, LL, RL, BL, Rev, BR)
 VALUES ('$ID', '$time', '$FL', '$LL', '$RL', '$BL', '$Rev', '$BR')";
 $mysql_result = mysql_query($sql);
        echo mysql_error();
        mysql_close();
if(empty($ID)){}
else
                include("test2.php");
?>
```

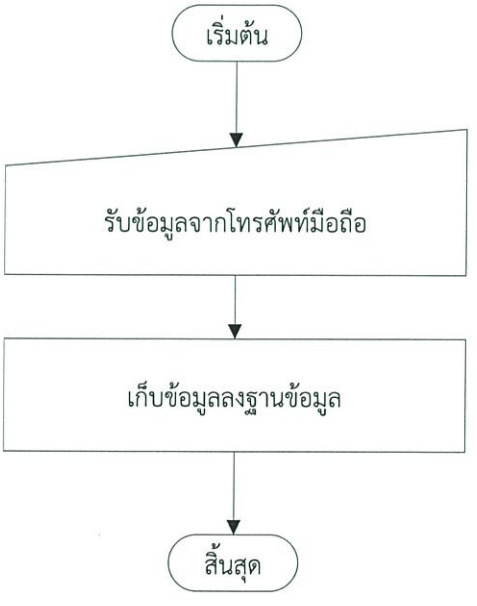


รูปที่ 3.23 โฟลว์ชาร์ตขั้นตอนการทำงานของโปรแกรมที่ใช้ยืนยันตัวตนและเก็บค่าข้อมูลเซนเซอร์

และโค้ดในไฟล์ test2.php ซึ่งใช้ในการเปรียบเทียบค่าที่ได้รับมากับค่าในฐานข้อมูลดังนี้

```
<?php
$con=mysql_connect('localhost','root','1234')or die(mysql_error());
mysql_select_db('truck')or die(mysql_error());
mysql_query("SET NAMES UTF8");
$check_log = mysql_query("SELECT * FROM rfid where ID = '$ID'");      $num =
mysql_num_rows($check_log);
        if($num <=0) {
                echo "N";
        }
        else {
                while ($data = mysql_fetch_array($check_log) ) {
                        echo "Y";
                }
        }
?>
```

ส่วนที่สองจะรับข้อมูลพิกัดจีพีเอสผ่านไฟล์ split2.php ซึ่งจะทำการตัดชุดข้อมูลที่ถูกส่งมาโดยใช้คำสั่ง explode และทำการบันทึกลงในตาราง truckGPS โดยมีขั้นตอนการทำงานดังรูปที่ 3.24



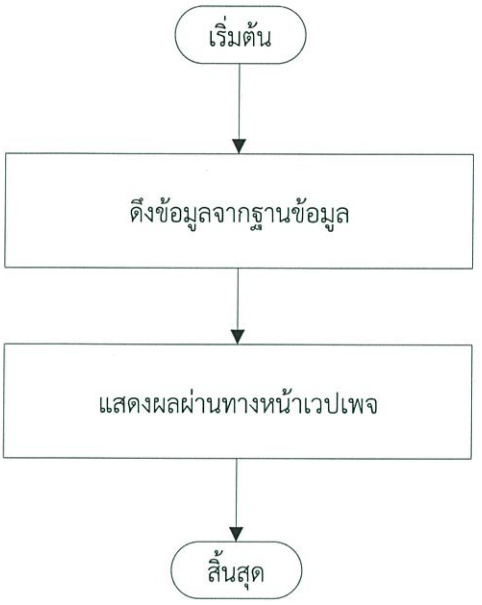
รูปที่ 3.24 โฟลว์ชาร์ตขั้นตอนการทำงานของโปรแกรมที่ใช้เก็บข้อมูลพิกัดจีพีเอส

จากโฟลว์ชาร์ตจะได้โค้ดในไฟล์ split2.php ดังนี้

```
<?php
$text=$_REQUEST['Data'];
$time = date("d/m/y (H:i:s)");
$data1 = explode("|" , $text);
$Lat = $data1[0];
$Lon = $data1[1];
$Spd = $data1[2];
$con=mysql_connect('localhost','root','1234')or die(mysql_error());
mysql_select_db('truck')or die(mysql_error());
mysql_query("SET NAMES UTF8");
                date_default_timezone_set("Asia/Bangkok");
                $sql="INSERT INTO truckgps (time, Lat, Lon, Spd)
            VALUES ('$time', '$Lat', '$Lon', '$Spd')";
            $mysql_result = mysql_query($sql);
                echo mysql_error();
                mysql_close();
?>
```

3.1.5.3 การแสดงผลผ่านทาง Google Maps API

ในการแสดงผลค่าพิกัดจะทำการแสดงผลผ่านทางแผนที่กูเกิ้ล โดยจะทำการดึงค่าเวลา ละติจูด ลองติจูด และความเร็วล่าสุดมาปักหมุดลงบนแผนที่โดยมีขั้นตอนการทำงานดังรูปที่ 3.25



รูปที่ 3.25 โฟลว์ชาร์ตขั้นตอนการแสดงผลค่าพิกัดผ่านทางแผนที่กูเกิ้ล

จากโฟลว์ชาร์ตจะได้โค้ดดังนี้

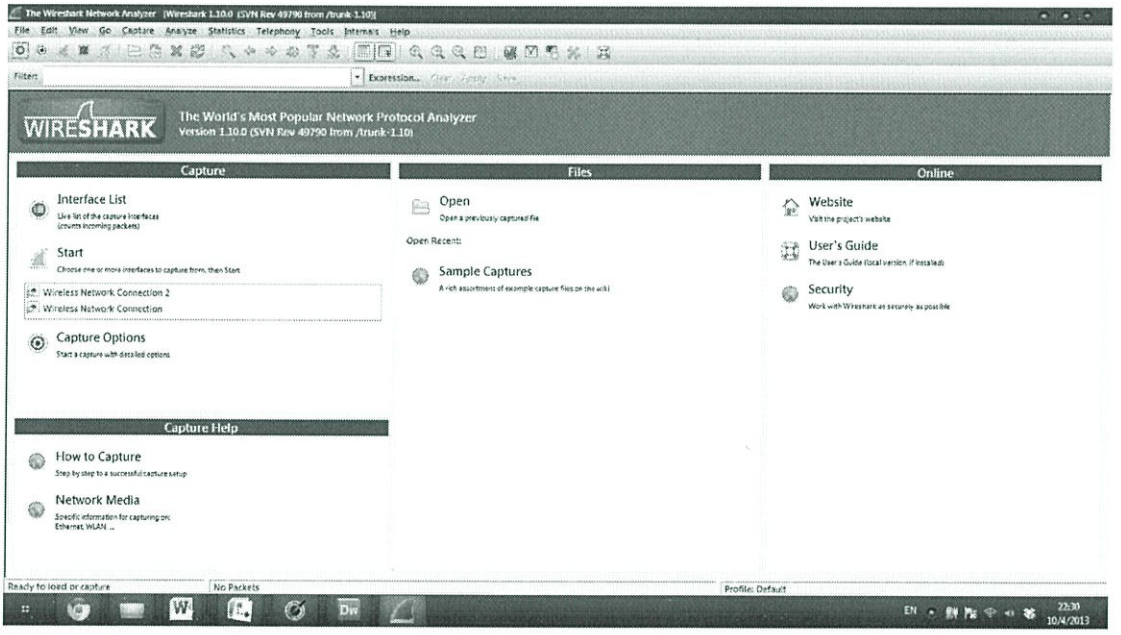
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="content-type" content="text/html; charset=utf-8"
/>
<title>Google Maps </title>
<?php
$connect=odbc_connect("mydb","root","1234")or die(mysql_error());
$sql="SELECT * FROM truckgps order by Seq ASC";
$result=odbc_exec($connect,$sql);
while(odbc_fetch_row($result))
{
  $latitude = odbc_result($result,"Lat");
  $longtitude = odbc_result($result,"Lon");
```

```
  $time = odbc_result($result,"time");
  $speed = odbc_result($result,"Spd");
}
?>
 <?php
odbc_free_result($result);
odbc_close($connect);
$comma=",";
$word=$latitude.$comma.$longtitude;
echo "Time $time , Speed $speed , Latitude $latitude , Longtitude $longtitude"; ?>
<script
src="http://maps.google.com/maps?file=api&amp;v=2&amp;key=ABQIAAAAeNsNnRX
UdrpgSw3qfvhz5hRHchrjOSPM-moa2HMmJZw-
0fE6VhTsARfVd9x1Dg8TowZALqHlOcO20g"
      type="text/javascript"></script>
<script type="text/javascript">
    function load() {
      if (GBrowserIsCompatible()) {
        var map = new GMap2(document.getElementById("map"));
        var center=new GLatLng(<? echo "$word"; ?>);
        map.setCenter(center,15);
              var point = new GLatLng(<? echo "$word"; ?>);
              var marker = new GMarker(point);
              map.addOverlay(marker);
        map.setUIToDefault();
      }
    }
    </script>
 </head>
```

```
  <body onload="load()" onunload="GUnload()">
    <div id="map" style="width:1350px;height:600px"></div>
  </body>
</html>
```

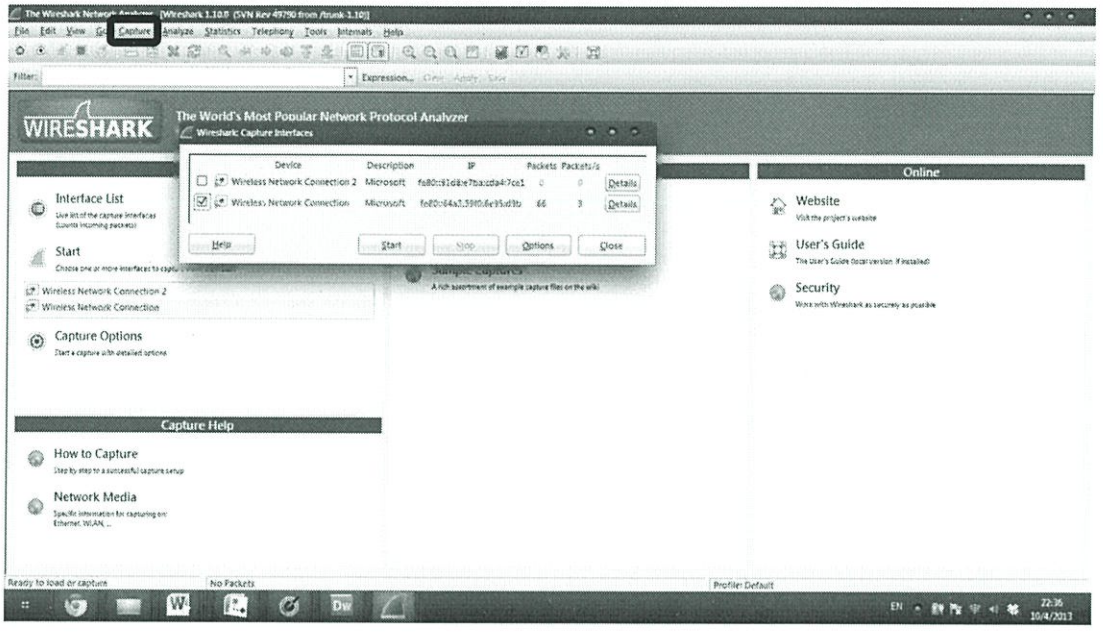
3.1.5.4 การตรวจสอบข้อมูลที่ฐานข้อมูลได้รับ

ในการตรวจสอบข้อมูลที่ฐานข้อมูลได้รับนั้นจะใช้โปรแกรมวายชาร์กในการตรวจจับแพ็กเกจของข้อมูลโดยหลังจากลงโปรแกรมแล้วให้ทำการเปิดโปรแกรมขึ้นมาจะพบหน้าต่างของโปรแกรมดังรูปที่ 3.26



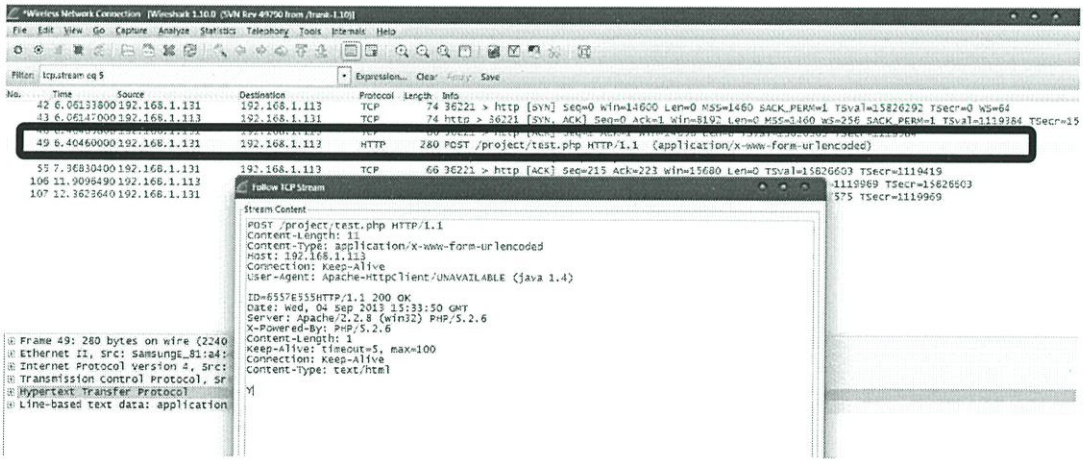
รูปที่ 3.26 หน้าต่างแรกหลังเปิดโปรแกรมของโปรแกรมวายชาร์ก

ทำการกำหนดฟิลเตอร์ที่จะใช้ในการกรองชนิดของข้อมูลที่จะตรวจจับ จากนั้นเริ่มทำการตรวจจับแพ็กเกจของข้อมูลโดยทำการเลือกแลนการ์ดที่เซิร์ฟเวอร์ใช้ในการเชื่อมต่ออินเตอร์เน็ต โดยกดที่ปุ่ม Capture > Interfaces จากนั้นกดปุ่ม Start ดังรูปที่ 3.27



รูปที่ 3.27 ขั้นตอนการตั้งค่าการตรวจจับแพ็กเกจข้อมูล

เมื่อทำการส่งข้อมูลเสร็จแล้วให้กดที่ปุ่ม Stop สีเหลี่ยมสีแดงจะปรากฏข้อมูลต่างๆ มากมายโดยในที่นี้เราจะสนใจเฉพาะไอพี Source ที่ตรงกับไอพีของอุปกรณ์ที่ใช้ส่งข้อมูลและโปรโตคอล HTTP ทำการเลือกข้อมูลที่ต้องการ และทำการอ่านค่าข้อมูลโดยคลิกขวาแล้วเลือก Follow TCP Stream จะปรากฏข้อมูลต่างๆ โดยสีแดงคือข้อมูลที่อุปกรณ์ส่งมา และสีฟ้าคือข้อมูลที่เซิร์ฟเวอร์ตอบรับดังรูปที่ 3.28



รูปที่ 3.28 แพ็กเกจข้อมูลที่อุปกรณ์ส่งเข้าฐานข้อมูล

3.2 เครื่องมือที่ใช้ในการทดลอง

3.2.2 พาวเวอร์ซัพพลาย (Power Supply)

3.2.3 วงจรตรวจจับการทำงานของหลอดไฟ

3.2.4 ชุดจำลองการทำงานระบบสัญญาณไฟรถยนต์

3.2.5 วงจรกรองความถี่ต่ำผ่าน

3.2.6 ออสซิลโลสโคป

3.2.7 ไมโครคอนโทรลเลอร์

3.2.8 โปรแกรมอาร์ดูอิโน่

3.2.9 โทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์

3.2.10 บลูธูทโมดูล

3.2.11 โปรแกรมอีคลิปส์

3.2.12 ฐานข้อมูล

3.2.13 อาร์เอฟไอดีโมดูล

3.2.14 แท็กส์อาร์เอฟไอดี

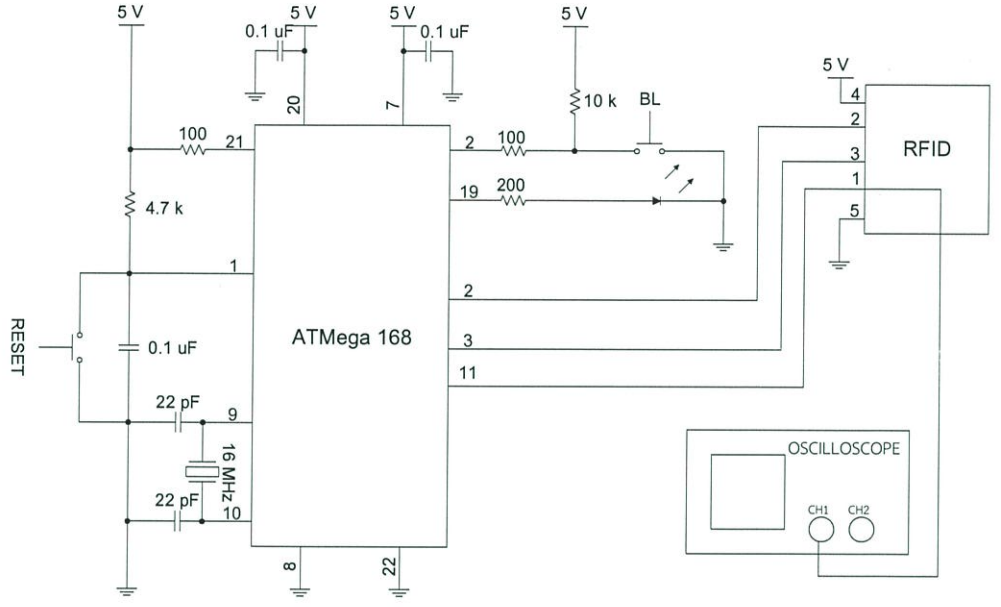
3.2.15 วงจรตัดไฟ

3.3 การจัดเก็บผลการทดลอง

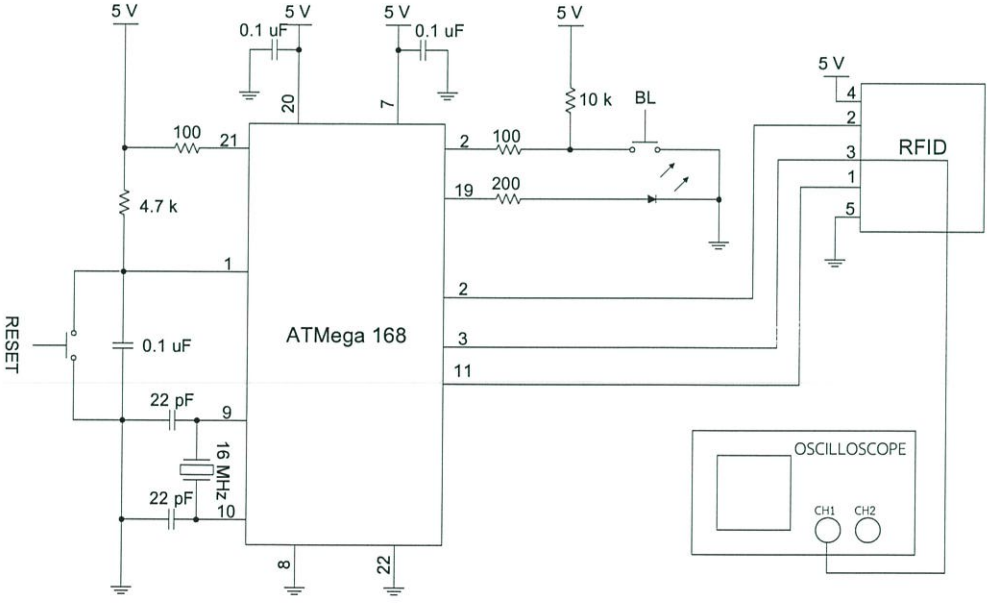
3.3.1 ส่วนตรวจสอบข้อมูลผู้ขับ

3.3.1.1 ระบบอาร์เอฟไอดี

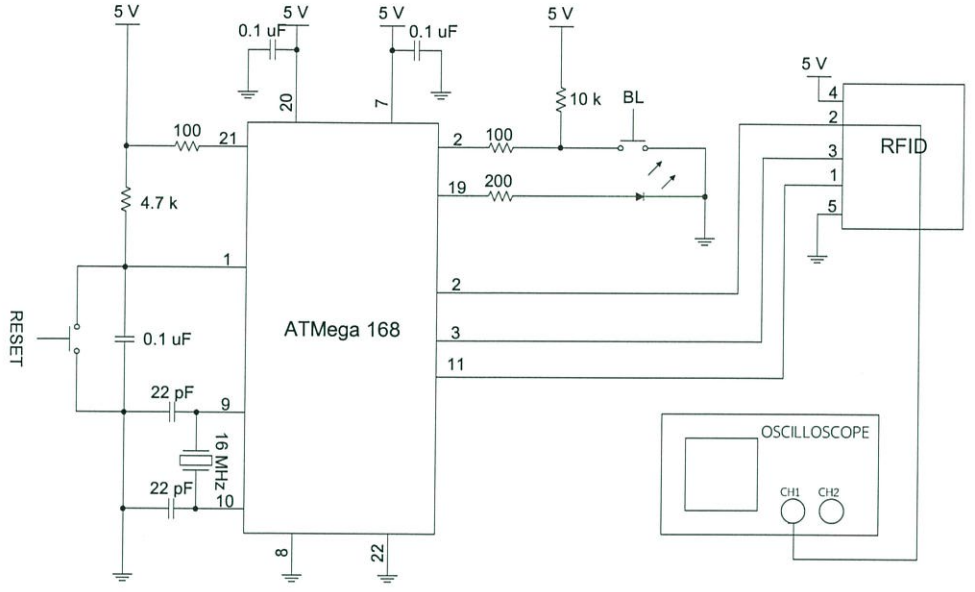
ทำการวัดสัญญาณจากตัวอ่านด้วยออสซิลโลสโคปโดยวัดสัญญาณการแตะแท็กส์ที่ขา 1 ของตัวอ่าน ดังรูปที่ 3.29 จากนั้นทำการวัดคำสั่งที่ส่งไปขอค่าแท็กส์ ที่ขา 3 ของตัวอ่าน ดังรูปที่ 3.30 และวัดค่าข้อมูลของแท็กส์ที่อ่านได้ที่ขา 2 ของตัวอ่าน ดังรูปที่ 3.31 พร้อมทั้งตรวจสอบข้อมูลที่อ่านได้จากซีเรียลมอนิเตอร์



รูปที่ 3.29 การวัดสัญญาณการแตะแท็กส์



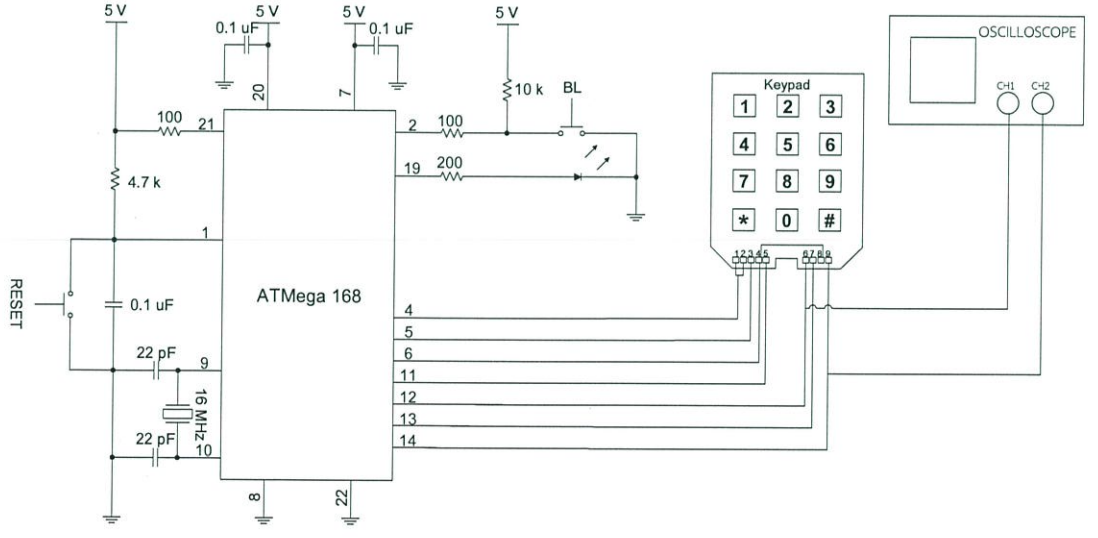
รูปที่ 3.30 การวัดสัญญาณคำสั่งขอค่าแท็กส์



รูปที่ 3.31 การวัดสัญญาณค่าข้อมูลแท็กส์

3.3.1.2 ระบบแป้นกด

ทำการตรวจสอบสัญญาณการกดแป้นกดด้วยออสซิลโลสโคป ดังรูปที่ 3.32

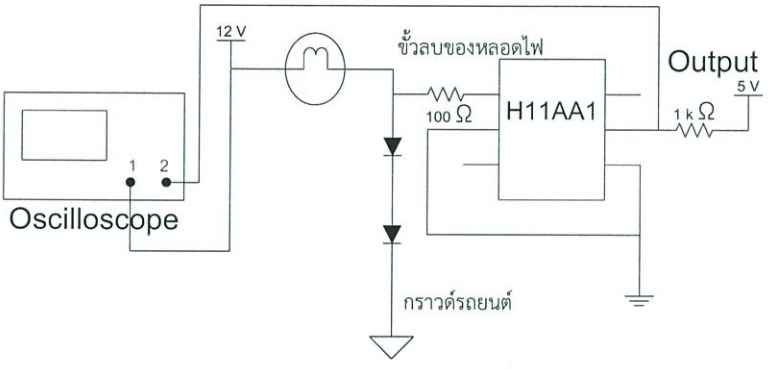


รูปที่ 3.32 การวัดสัญญาณการกดแป้นกด

3.3.2 ชุดตรวจจับการทำงานของหลอดไฟและระบบเบรก

3.3.2.1 ชุดตรวจจับการทำงานของหลอดไฟ

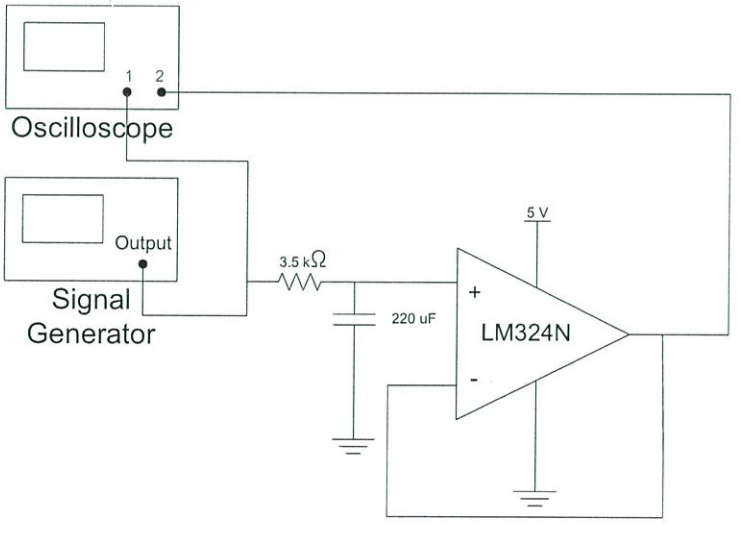
ทำการต่อหลอดไฟเข้ากับวงจรตรวจจับการทำงานของหลอดไฟ และบันทึกผลการทดลองเมื่อเปิดไฟ และปิดไฟด้วยเครื่องออสซิลโลสโคป รูปการทดลองสามารถแสดงได้ ดังรูปที่ 3.33



รูปที่ 3.33 การทดลองวงจรตรวจจับการทำงานของหลอดไฟ

3.3.2.2 วงจรกรองความถี่ต่ำผ่าน

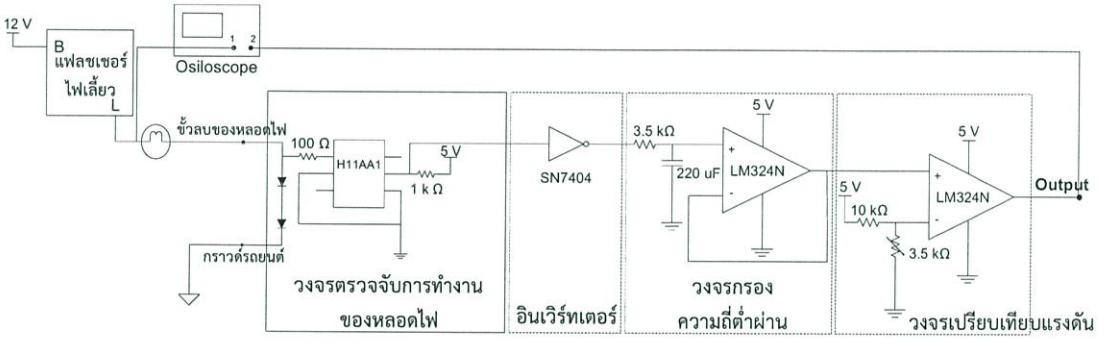
ทำการป้อนสัญญาณไซน์เข้าสู่วงจรกรองความถี่ต่ำผ่าน และบันทึกแอมปลิจูดของสัญญาณขาออกเมื่อความถี่ที่ป้อนเพิ่มขึ้น รูปการทดลองสามารถแสดงได้ ดังรูปที่ 3.34



รูปที่ 3.34 การทดลองหาความถี่ตัดทางด้านต่ำของวงจรกรองความถี่ต่ำผ่าน

3.3.2.3 ชุดตรวจจับการทำงานของไฟเลี้ยว

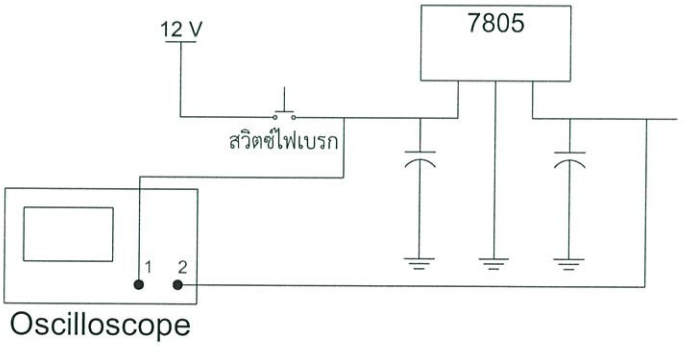
ทำการต่อหลอดไฟเลี้ยวเข้ากับชุดตรวจจับการทำงานของไฟเลี้ยว และบันทึกผลการทดลองเมื่อเปิดไฟ และปิดไฟด้วยเครื่องออสซิลโลสโคป รูปการทดลองสามารถแสดงได้ ดังรูปที่ 3.35



รูปที่ 3.35 การทดลองชุดตรวจจับการทำงานของหลอดไฟ

3.3.2.4 ชุดตรวจจับการทำงานของเบรก

ทำการต่อชุดตรวจจับการทำงานของเบรกเข้ากับสายสัญญาณสวิตซ์ไฟเบรก และบันทึกผลการทดลองเมื่อสวิตซ์ไฟเบรกทำงาน และไม่ทำงาน รูปการทดลองสามารถแสดงได้ ดังรูปที่ 3.36



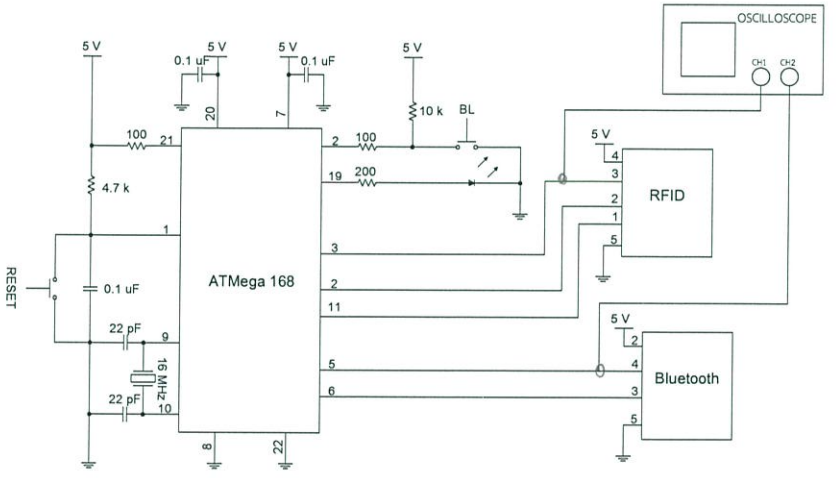
รูปที่ 3.36 การทดลองชุดตรวจจับการทำงานของเบรก

3.3.2.5 โปรแกรมการทำงานของชุดตรวจจับการทำงานของหลอดไฟและระบบเบรก

ทำการต่อชุดจำลองระบบสัญญาณไฟรถยนต์เข้ากับชุดตรวจจับการทำงานของหลอดไฟ และระบบเบรก และต่อชุดตรวจจับการทำงานของหลอดไฟ และระบบเบรกเข้ากับไมโครคอนโทรลเลอร์เข้าที่ขา 23 24 25 26 27 และ 28 ทำการทดลองเปิดไฟ – ปิดไฟแต่ละหลอด และบันทึกผลผ่านทางซีเรียลมอนิเตอร์

3.3.3 ส่วนส่งข้อมูลเข้าโทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์

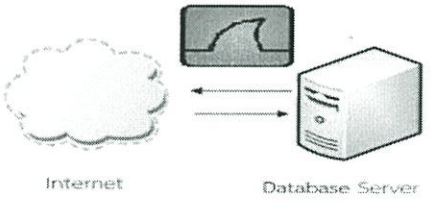
การเก็บผลการทดลองทำโดยการวัดสัญญาณที่ขา 4 ของบลูธูทโมดูลเทียบกับสัญญาณที่ขา 3 ของอาร์เอฟไอดี ดังรูปที่ 3.37



รูปที่ 3.37 การวัดสัญญาณของการส่งข้อมูลเข้าโทรศัพท์มือถือ

3.3.4 โทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์และฐานข้อมูล

การเก็บผลการทดลองในการส่งข้อมูลเข้าสู่ฐานข้อมูลทำได้โดยใช้โปรแกรมวายชาร์กดักจับข้อมูลที่ถูกส่งมาจากโทรศัพท์มือถือดังรูปที่ 3.38 แล้วทำการอ่านค่าข้อมูลต่างๆ ในโปรแกรมว่าสอดคล้องกับข้อมูลที่ส่งมา หรือไม่ส่วนในโทรศัพท์มือถือใช้การถ่ายภาพหน้าจอ



รูปที่ 3.38 การเก็บผลการทดลองในการส่งข้อมูลเข้าสู่ฐานข้อมูล

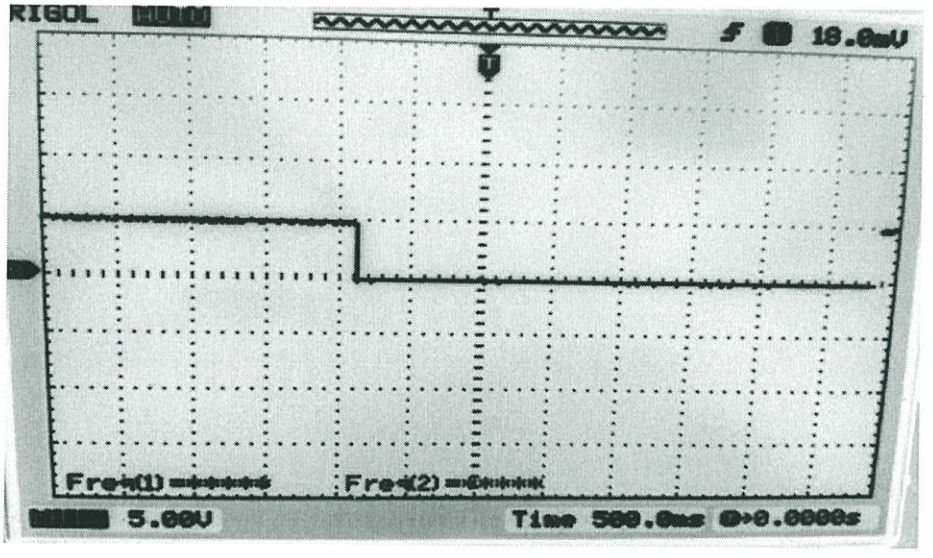
บทที่ 4
ผลการทดลอง

4.1 ส่วนการยืนยันตัวตน

4.1.1 อาร์เอฟไอดี

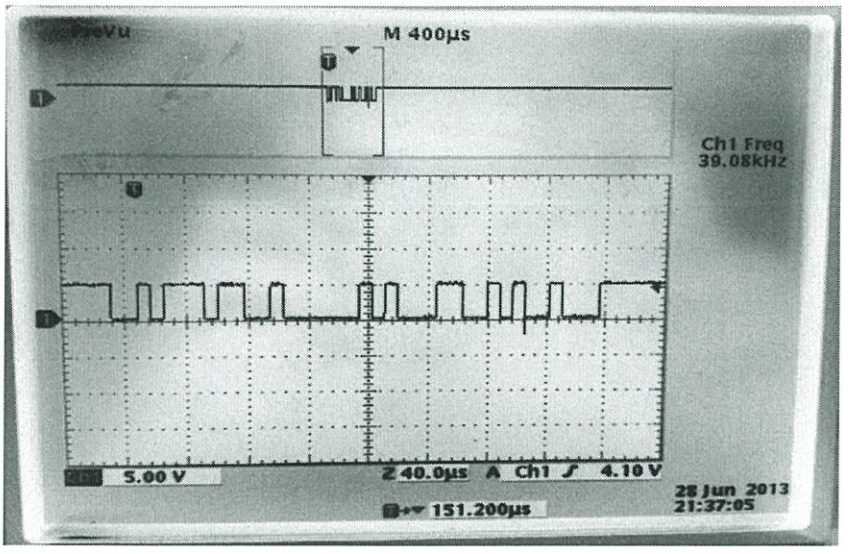
จากการออกแบบการทดลองที่ผ่านมา ทำการวัดสัญญาณจากขาของโมดูลอาร์เอฟไอดี 3 ขา ได้แก่

1) วัดแรงดันไฟฟ้าในขา 1 ดังรูปที่ 3.29 ว่ามีการเปลี่ยนแปลงอย่างไร โดยผลที่ได้จะเป็นไปดังรูปที่ 4.1 คือ ค่าแรงดันไฟฟ้าในขานี้จะลดลงเมื่อมีการนำแท็กส์เข้ามาใกล้กับตัวโมดูลอาร์เอฟไอดี โดยที่การวัดสัญญาณในช่วงแรกที่ยังไม่มีแท็กส์เข้ามาใกล้สัญญาณแรงดันไฟฟ้าจะอยู่ที่ 5 โวลต์และเมื่อมีแท็กส์เข้ามาใกล้ตัวอาร์เอฟไอดีโมดูล สัญญาณแรงดันไฟฟ้าก็จะลดลงมาที่ 0 โวลต์



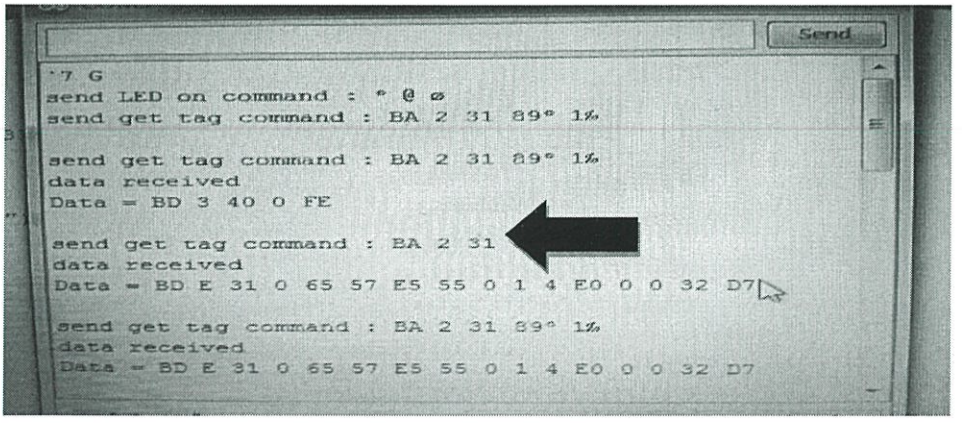
รูปที่ 4.1 สัญญาณที่วัดได้จากขาที่ 1 ของตัวโมดูลอาร์เอฟไอดี

2) วัดสัญญาณที่ขารับข้อมูลของโมดูลอาร์เอฟไอดีดังรูปที่ 3.30 เพื่อตรวจสอบว่าคำสั่งที่ส่งไปตรงกับโปรแกรมคำสั่งที่เขียนหรือไม่ โดยจะได้ผลของสัญญาณดังรูปที่ 4.2



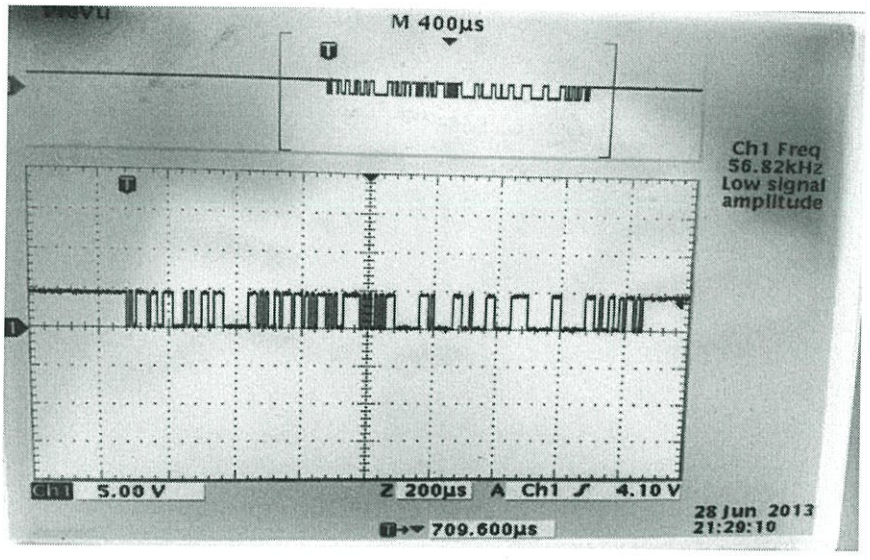
รูปที่ 4.2 สัญญาณที่วัดได้จากขารับข้อมูลของตัวโมดูลอาร์เอฟไอดี

เมื่อทำการอ่านค่าของสัญญาณจากซ้ายไปขวาออกมาจะได้ค่าดังนี้ คือ 0 0101 1101 10 0100 0000 10 1000 1100 10 1001 0001 10 จากนั้นทำการตัดบิตเริ่มต้นและบิตสิ้นสุดจะได้ค่าข้อมูลที่เหลือคือ 0101 1101 0100 0000 1000 1100 1001 0001จัดข้อมูลให้เป็นชุดละ 8 บิต และอ่านกลับจากขวาไปซ้ายจะได้ชุดข้อมูลใหม่เป็น 1011 1010 0000 0010 0011 0001 1000 1001 จากนั้นอ่านข้อมูลออกมาเป็นเลขฐานสิบหกจะได้ค่าข้อมูลเป็น BA 02 31 89 และทำการตรวจสอบข้อมูลในซีเรียลมอนิเตอร์จากรูปที่ 4.3 (ที่ลูกศรชี้)



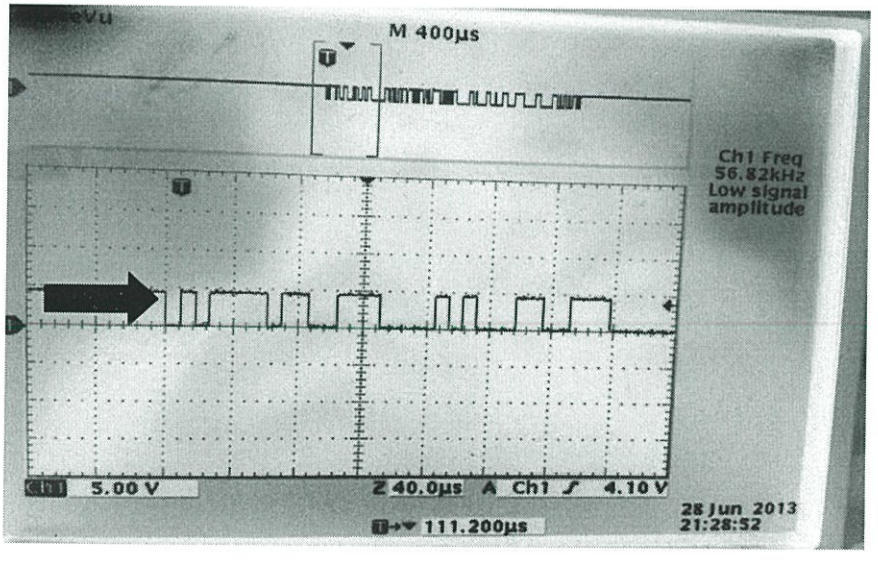
รูปที่ 4.3 ค่าข้อมูลที่อ่านได้จากซีเรียลมอนิเตอร์

3) วัดสัญญาณจากขาส่งข้อมูลของโมดูลอาร์เอฟไอดีดังรูปที่ 3.31 เพื่อทำการตรวจสอบค่าข้อมูลที่ตัวโมดูลตอบกลับมาโดยจะมีผลของสัญญาณที่วัดได้ ดังรูปที่ 4.4



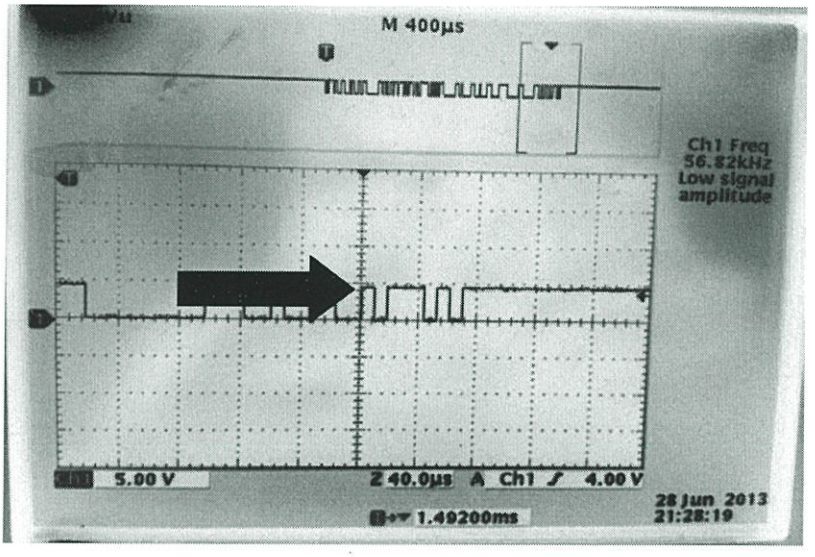
รูปที่ 4.4 สัญญาณที่วัดได้จากขา 4 ของโมดูลอาร์เอฟไอดี

ทำการตรวจสอบค่าของข้อมูลชุดแรก และข้อมูลชุดสุดท้ายเนื่องจากข้อมูลมีความยาวมาก และเป็นการประหยัดเวลาโดยข้อมูลในชุดแรกจะมีรูปแบบดังรูปที่ 4.5



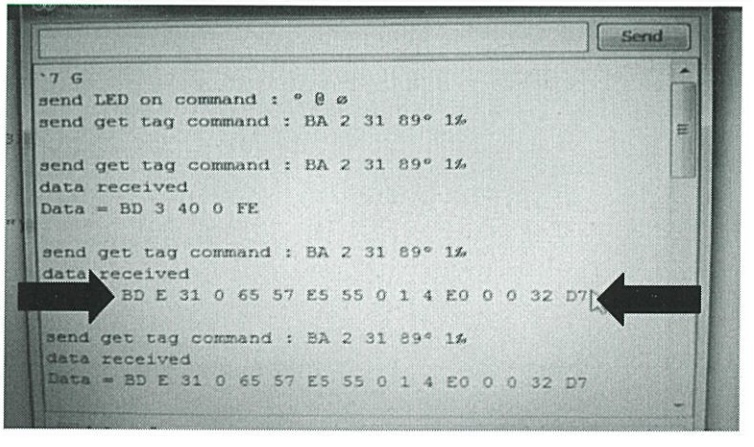
รูปที่ 4.5 สัญญาณชุดแรกที่วัดได้

ทำการอ่านข้อมูลออกมาได้บิตข้อมูลดังนี้ 0 1011 1101 10 ทำการตัดบิตเริ่มต้นและบิตสิ้นสุดจะได้เป็น 1011 1101 และทำการอ่านข้อมูลจากขวาไปซ้ายได้ 1011 1101 แปลงเป็นเลขฐานสิบหกจะได้ค่าข้อมูลได้เป็น BD จากนั้นทำการตรวจสอบค่าข้อมูลชุดสุดท้ายโดยค่าข้อมูลชุดสุดท้ายที่วัดได้จะมีรูปแบบ ดังรูปที่ 4.6



รูปที่ 4.6 สัญญาณชุดสุดท้ายที่วัดได้

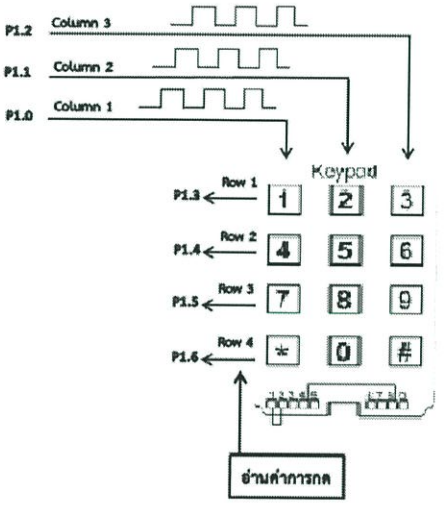
ทำการอ่านข้อมูลออกมาได้บิตข้อมูลดังนี้ 10 1110 1011 1 ทำการตัดบิตเริ่มต้นและบิตสิ้นสุดจะได้เป็น 1110 1011 และทำการอ่านข้อมูลจากขวาไปซ้ายได้ 1101 0111 แปลงเป็นเลขฐานสิบหกจะได้ค่าข้อมูลได้เป็น D7 จากนั้นนำค่าที่ได้ไปตรวจสอบกับข้อมูลในซีเรียลมอนิเตอร์จากรูปที่ 4.7



รูปที่ 4.7 ค่าข้อมูลที่อ่านได้จากซีเรียลมอนิเตอร์

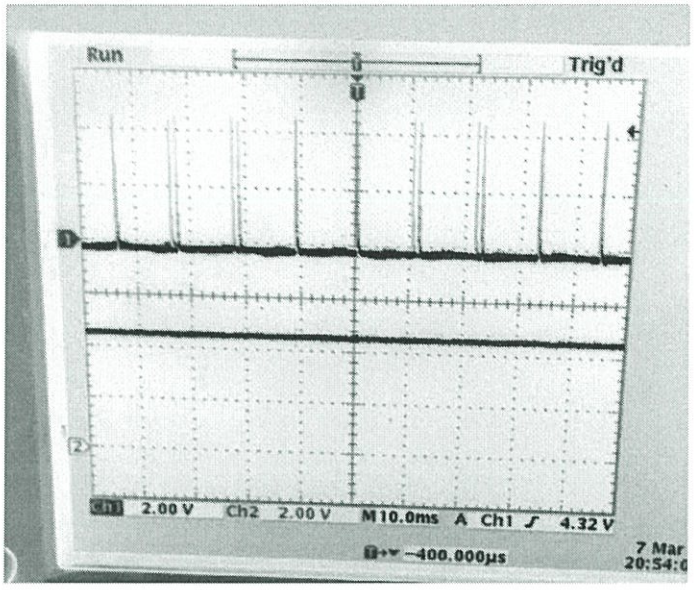
4.1.2 แป้นกด

ทำการวัดสัญญาณแป้นกดโดยการกดปุ่ม โดยสัญญาณที่ Column 1 Column 2 และ Column 3 จะเป็นสัญญาณที่ใช้ในการตรวจสอบและจะส่งออกมาโดยจะมีคาบเวลาที่แตกต่างกัน เมื่อมีการกดปุ่มบนแป้นกด สัญญาณตรวจสอบ ก็จะทำการวิ่งไปที่ขา Row ใดๆที่หมายเลขนั้นอยู่ และไมโครคอนโทรลเลอร์ก็จะประมวลผลและแสดงออกมาเป็นหมายเลยเป้าหมายตามที่ต้องการ แสดงดังรูปที่ 4.8

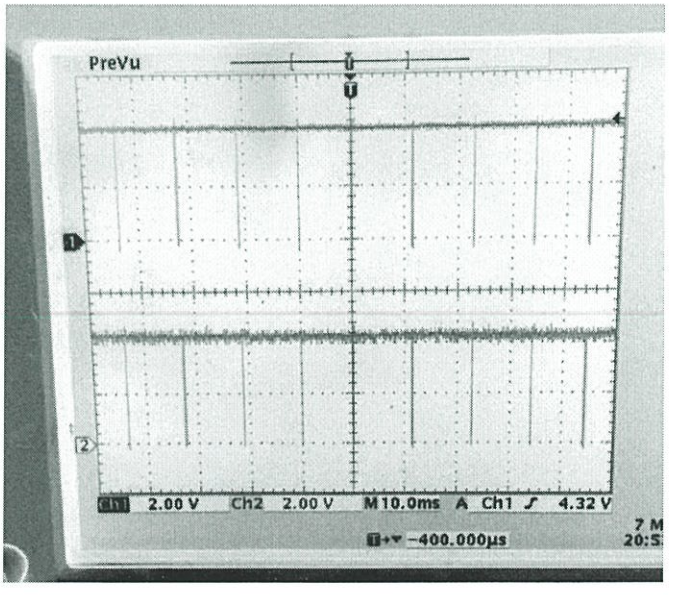


รูปที่ 4.8 รูปสัญญาณการตรวจสอบของแป้นกด

จากการทดลองแป้นกด เมื่อทำการวัดสัญญาณโดยใช้ดิจิตอลออสซิลโลสโคปวัดสัญญาณที่เกิดขึ้น สังเกตได้ว่าสัญญาณที่เกิดขึ้นนั้นเป็นสัญญาณที่มีความถี่สูงมาก โดยได้ทำการวัดแป้นกด ซึ่งยังไม่กดปุ่ม ซึ่งจะเป็นสัญญาณอินพุต ทางด้าน Column และ Row ดังรูปที่ 4.9 จากนั้นทำการวัดสัญญาณที่ออกจากแป้นกด ในขณะที่ทำการกดปุ่มกดจะได้สัญญาณที่เป็นเอาท์พุตจาก Column และ Row ดังรูปที่ 4.10



รูปที่ 4.9 รูปสัญญาณขณะยังไม่ได้กดปุ่มในแป้นกด
Ch 1 วัดสัญญาณที่ Column
Ch 2 วัดสัญญาณที่ Row

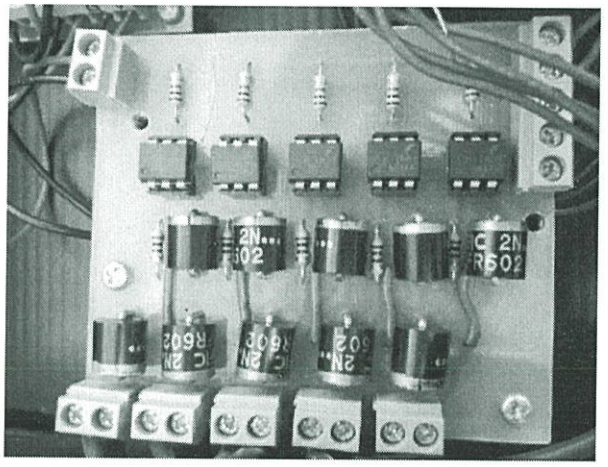


รูปที่ 4.10 รูปสัญญาณขณะกดปุ่มในแป้นกด
Ch 1 วัดสัญญาณที่ Column
Ch 2 วัดสัญญาณที่ Row

4.2 ชุดตรวจจับการทำงานของหลอดไฟ และระบบเบรก

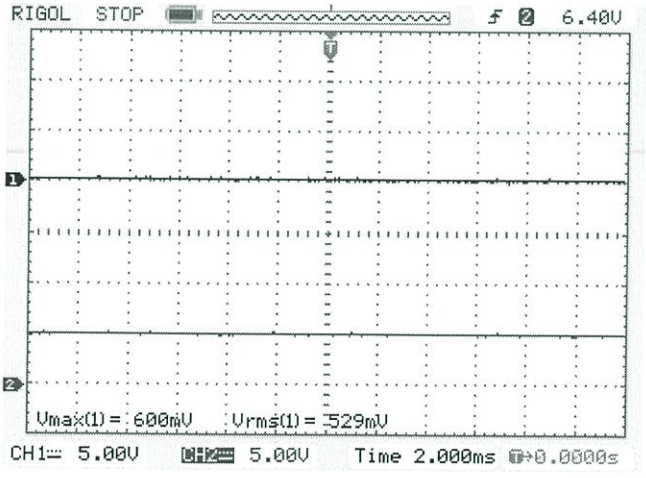
4.2.1 วงจรตรวจจับการทำงานของหลอดไฟ

ในการออกแบบวงจรตรวจจับการทำงานของหลอดไฟ 1 ชุดจะประกอบด้วยวงจรตรวจจับการทำงานของหลอดไฟจำนวน 5 ตัว โดยวงจรที่เสร็จแล้วสามารถแสดงได้ดังรูปที่ 4.11



รูปที่ 4.11 วงจรตรวจจับการทำงานของหลอดไฟ

จากการต่อวงจรดังรูปที่ 3.33 เมื่อหลอดไฟยังไม่ทำงานสัญญาณสามารถแสดงได้ดังรูปที่ 4.12

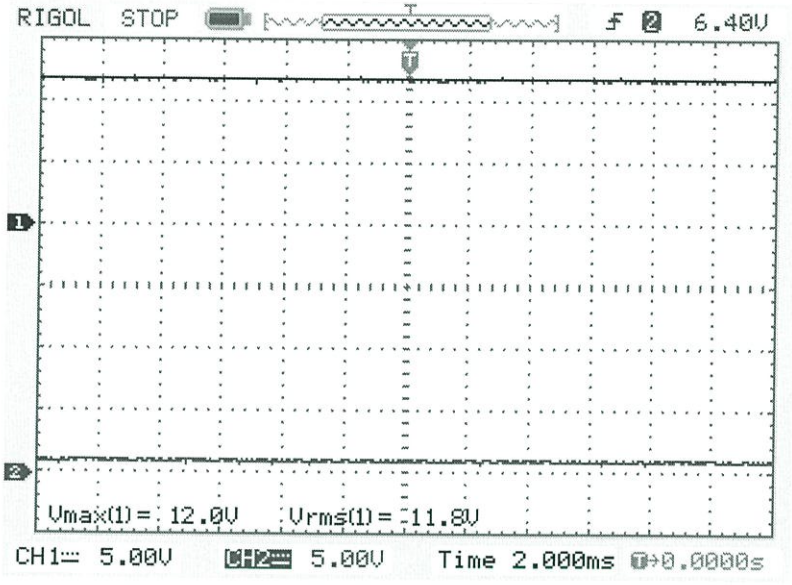


รูปที่ 4.12 สัญญาณที่ออกจากวงจรตรวจจับการทำงานของหลอดไฟเมื่อหลอดไฟไม่ทำงาน

CH1 สัญญาณที่หลอดไฟ

CH2 สัญญาณจากวงจรตรวจจับการทำงานของหลอดไฟ

เมื่อหลอดไฟทำงานสัญญาณสามารถแสดงได้ดังรูปที่ 4.13



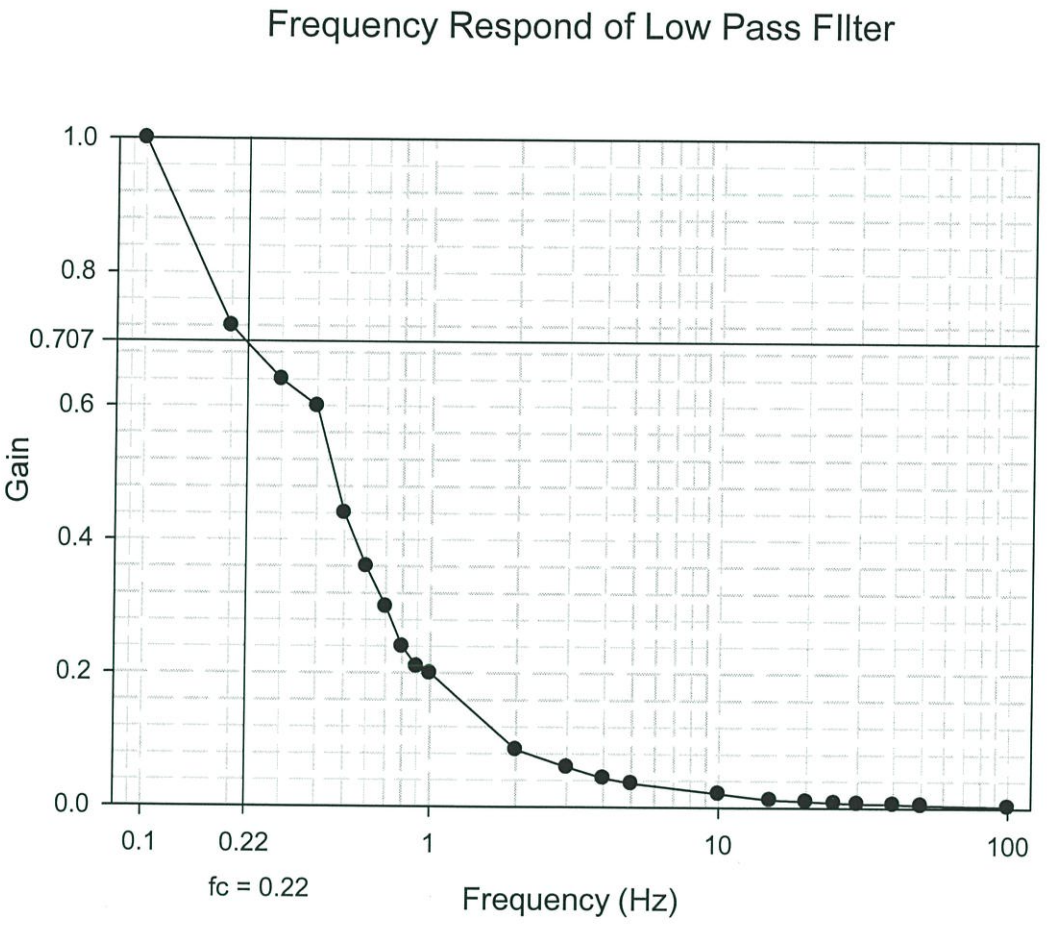
รูปที่ 4.13 สัญญาณที่ออกจากวงจรตรวจจับการทำงานของหลอดไฟเมื่อหลอดไฟทำงาน
CH1 สัญญาณที่หลอดไฟ
CH2 สัญญาณจากวงจรตรวจจับการทำงานของหลอดไฟ

จะเห็นได้ว่าวงจรตรวจจับการทำงานของหลอดไฟในขณะที่หลอดไฟยังไม่มีการทำงานนั้นจะให้สถานะ HIGH ออกมา และเมื่อหลอดไฟทำงานจะเปลี่ยนเป็นสถานะ LOW

4.2.2 วงจรกรองความถี่ต่ำผ่าน

จากการทดลองดังรูปที่ 3.34 ป้อนสัญญาณไซน์ที่ความถี่ต่างๆ เพื่อหาความถี่ตัดทางด้านต่ำจะได้กราฟผลตอบสนองเชิงความถี่สามารถแสดงได้ดังรูปที่ 4.14

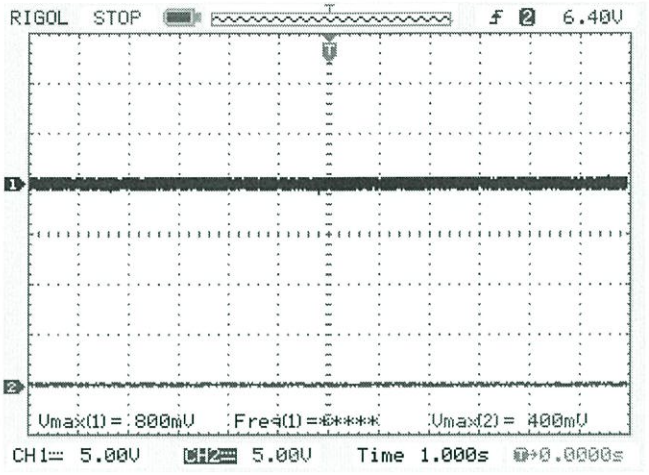
จากกราฟในรูปที่ 4.14 จะได้ความถี่ตัดทางด้านต่ำเท่ากับ 0.22 เฮิรตซ์ซึ่งมีค่าใกล้เคียงกับที่ได้ทำการออกแบบไว้



รูปที่ 4.14 ผลตอบสนองเชิงความถี่ของวงจรกรองความถี่ต่ำผ่าน

4.2.3 ชุดตรวจจับการทำงานของไฟเลี้ยว

จากการต่อวงจรดังรูปที่ 3.35 เมื่อไฟเลี้ยวยังไม่ทำงานสัญญาณสามารถแสดงได้ดัง รูปที่ 4.15

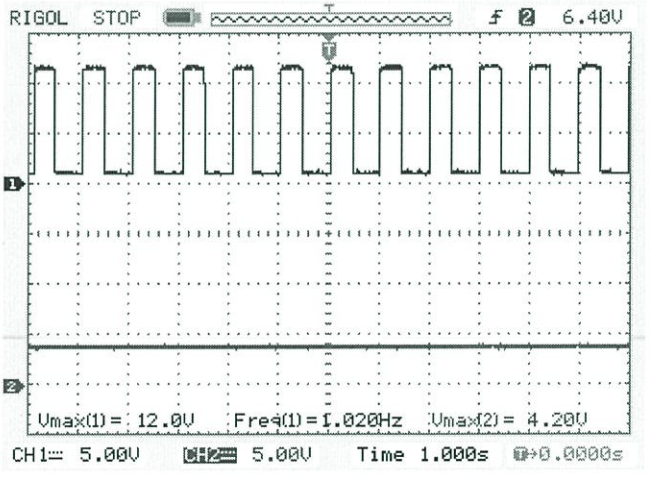


รูปที่ 4.15 สัญญาณที่ออกจากชุดตรวจจับการทำงานของไฟเลี้ยวเมื่อไฟเลี้ยวไม่ทำงาน

CH1 สัญญาณที่ไฟเลี้ยว

CH2 สัญญาณจากชุดตรวจจับการทำงานของไฟเลี้ยว

เมื่อไฟเลี้ยวทำงานสัญญาณสามารถแสดงได้ดังรูปที่ 4.16



รูปที่ 4.16 สัญญาณที่ออกจากชุดตรวจจับการทำงานของไฟเลี้ยวเมื่อไฟเลี้ยวทำงาน

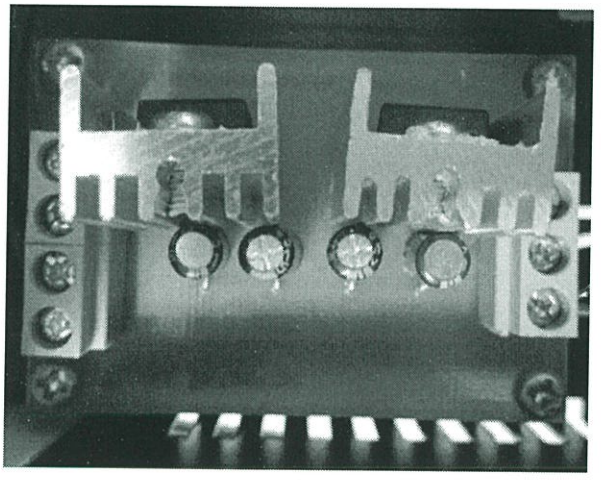
CH1 สัญญาณที่ไฟเลี้ยว

CH2 สัญญาณจากชุดตรวจจับการทำงานของไฟเลี้ยว

จะเห็นได้ว่าเมื่อไฟเลี้ยวไม่ทำงานสัญญาณที่ออกจากชุดตรวจจับการทำงานของไฟเลี้ยวจะมีสถานะ LOW และเมื่อไฟเลี้ยวทำงานสัญญาณจะมีสถานะ HIGH

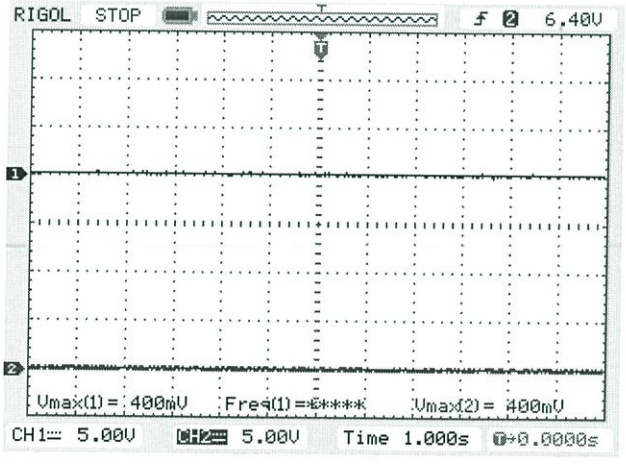
4.2.4 ชุดตรวจจับการทำงานของเบรก

ในการออกแบบชุดตรวจจับการทำงานของเบรกนั้นวงจรที่เสร็จแล้วสามารถแสดงได้ดังรูปที่ 4.17



รูปที่ 4.17 วงจรตรวจจับการทำงานของเบรก

จากการต่อวงจรดังรูป 4.17 เมื่อสวิตช์ไฟเบรกยังไม่ทำงานสัญญาณสามารถแสดงได้ดังรูปที่ 4.18

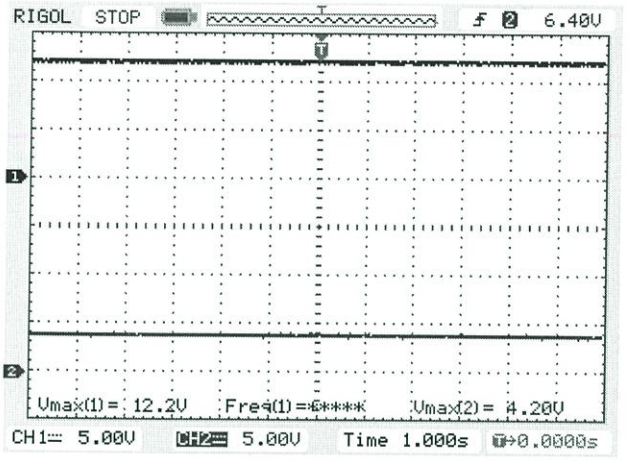


รูปที่ 4.18 สัญญาณที่ออกจากชุดตรวจจับการทำงานของเบรกเมื่อสวิตช์ไฟเบรกไม่ทำงาน

CH1 สัญญาณที่สวิตช์ไฟเบรก

CH2 สัญญาณจากชุดตรวจจับการทำงานของเบรก

เมื่อสวิตช์ไฟเบรกทำงานสัญญาสามารถแสดงได้ดังรูปที่ 4.19



รูปที่ 4.19 สัญญาณที่ออกจากชุดตรวจจับการทำงานของเบรกเมื่อสวิตซ์ไฟเบรกทำงาน

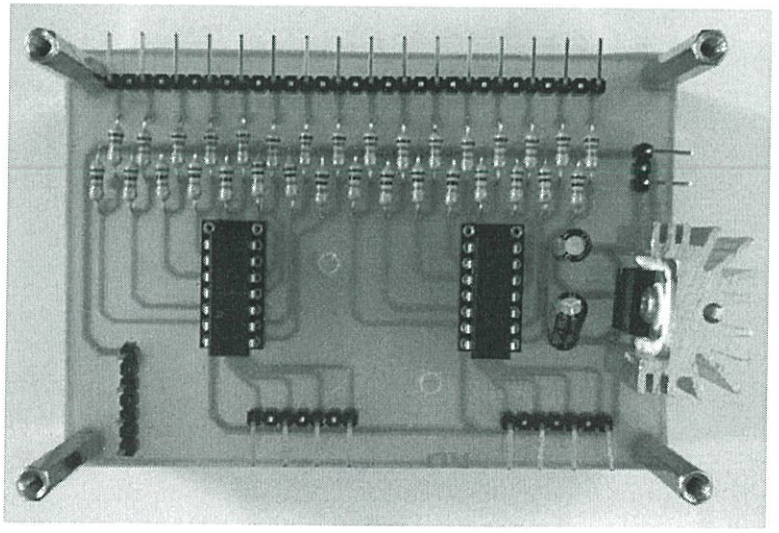
CH1 สัญญาณที่สวิตซ์ไฟเบรก

CH2 สัญญาณจากชุดตรวจจับการทำงานของเบรก

จะเห็นได้ว่าเมื่อสวิตซ์ไฟเบรกไม่ทำงานสัญญาณที่ออกจากชุดตรวจจับการทำงานของเบรกจะมีสถานะ LOW และเมื่อสวิตซ์ไฟเบรกทำงานสัญญาณจะมีสถานะ HIGH

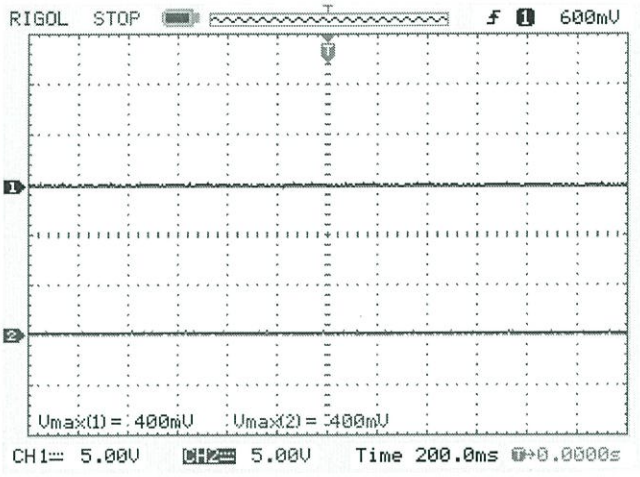
4.2.5 วงจรตรวจจับการเปิด – ปิดหลอดไฟ

จากการออกแบบวงจรได้แผ่นวงจรสามารถแสดงได้ดังรูปที่ 4.20



รูปที่ 4.20 แผ่นวงจรตรวจจับการเปิด – ปิดหลอดไฟ

จากการต่อวงจรดังรูปที่ 4.20 เมื่อหลอดไฟปิด สัญญาณสามารถแสดงได้ดังรูปที่ 4.21

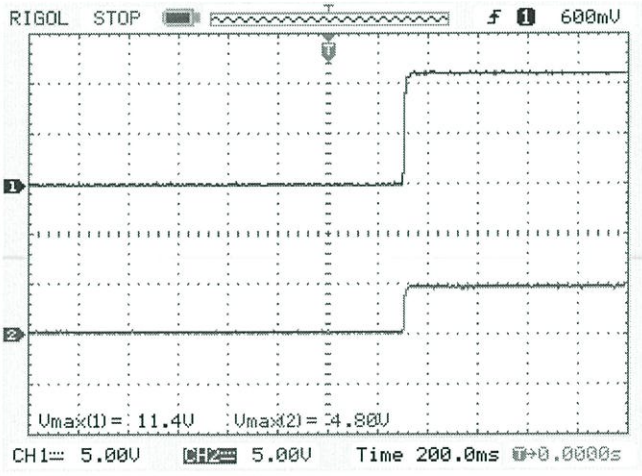


รูปที่ 4.21 สัญญาณของวงจรตรวจจับการเปิด – ปิดหลอดไฟเมื่อหลอดไฟปิด

CH 1 สัญญาณที่หลอดไฟ

CH 2 สัญญาณที่ออกจากวงจรตรวจจับ

เมื่อหลอดไฟเปิดสัญญาณสามารถแสดงได้ดังรูปที่ 4.22

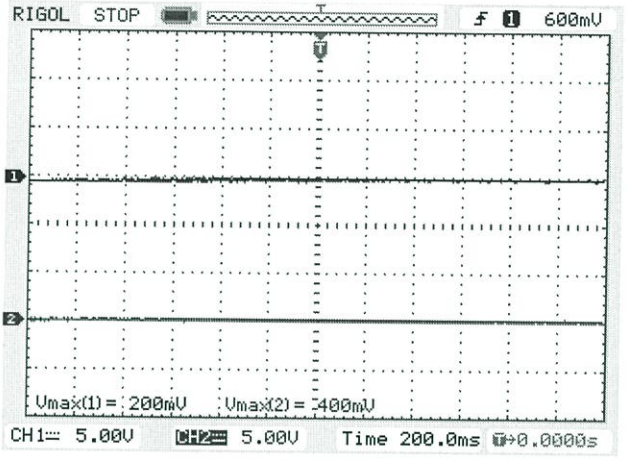


รูปที่ 4.22 สัญญาณของวงจรตรวจจับการเปิด – ปิดหลอดไฟเมื่อหลอดไฟเปิด

CH 1 สัญญาณที่หลอดไฟ

CH 2 สัญญาณที่ออกจากวงจรตรวจจับ

จากการต่อวงจรดังรูปที่ 4.20 เมื่อหลอดไฟเลี้ยวปิด สัญญาณสามารถแสดงได้ดังรูปที่ 4.23

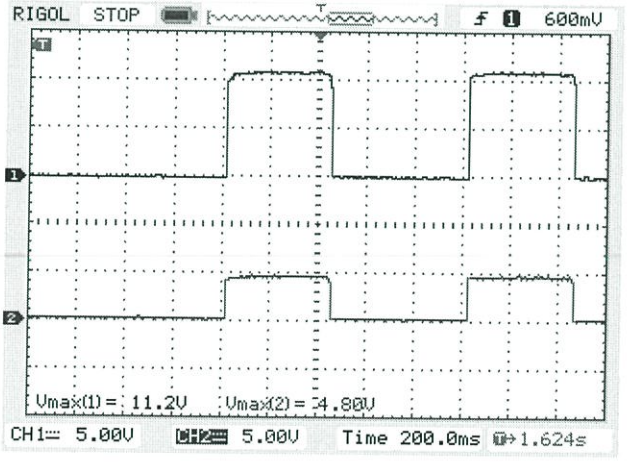


รูปที่ 4.23 สัญญาณของวงจรตรวจจับการเปิด – ปิดหลอดไฟเมื่อหลอดไฟเลี้ยวปิด

CH 1 สัญญาณที่หลอดไฟเลี้ยว

CH 2 สัญญาณที่ออกจากวงจรตรวจจับ

เมื่อหลอดไฟเปิดสัญญาณสามารถแสดงได้ดังรูปที่ 4.24



รูปที่ 4.24 สัญญาณของวงจรตรวจจับการเปิด – ปิดหลอดไฟเมื่อหลอดไฟเลี้ยวเปิด

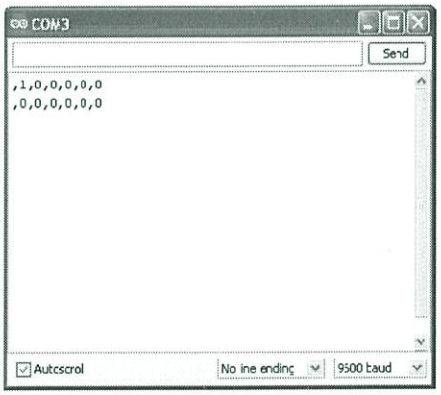
CH 1 สัญญาณที่หลอดไฟเลี้ยว

CH 2 สัญญาณที่ออกจากวงจรตรวจจับ

4.2.6 โปรแกรมการทำงานของชุดตรวจจับการทำงานของหลอดไฟ และระบบเบรก

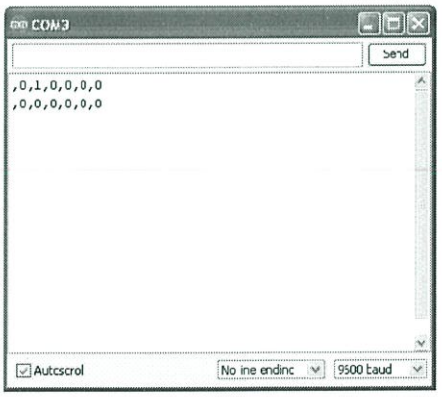
โดยในการทดลองโปรแกรมการทำงานของชุดตรวจจับการทำงานของหลอดไฟจะทำการเปิด – ปิดหลอดไฟ และสวิตซ์ไฟเบรกเพื่อดูค่าที่ออกมาโดยผลการทดลองมีดังนี้

4.2.6.1 ทดลองเปิด - ปิดไฟหน้า ค่าที่ได้สามารถแสดงได้ดังรูปที่ 4.25



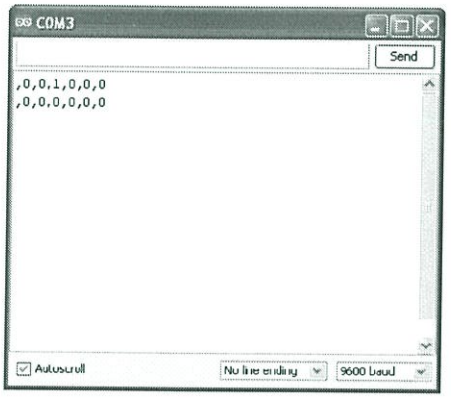
รูปที่ 4.25 ผลของโปรแกรมเมื่อเปิด – ปิดไฟหน้า
บรรทัดบน เมื่อไฟหน้าทำงาน
บรรทัดล่าง เมื่อไฟหน้าไม่ทำงาน

4.2.6.2 ทดลองเปิด - ปิดไฟเลี้ยวด้านซ้าย ค่าที่ได้สามารถแสดงได้ดังรูปที่ 4.26



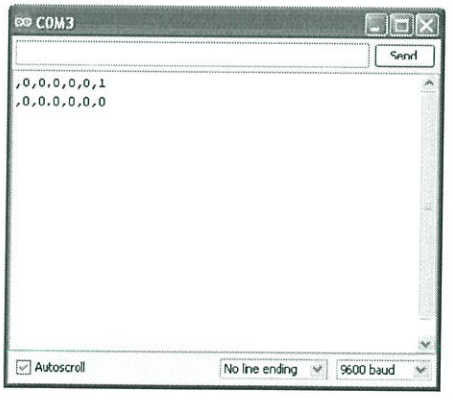
รูปที่ 4.26 ผลของโปรแกรมเมื่อเปิด – ปิดไฟเลี้ยวด้านซ้าย
บรรทัดบน เมื่อไฟเลี้ยวด้านซ้ายทำงาน
บรรทัดล่าง เมื่อไฟเลี้ยวด้านซ้ายไม่ทำงาน

4.2.6.3 ทดลองเปิด - ปิดไฟเลี้ยวด้านขวา ค่าที่ได้สามารถแสดงได้ดังรูปที่ 4.27



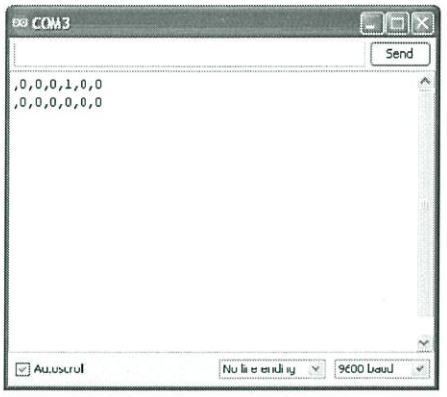
รูปที่ 4.27 ผลของโปรแกรมเมื่อเปิด – ปิดไฟเลี้ยวด้านขวา
บรรทัดบน เมื่อไฟเลี้ยวด้านขวาทำงาน
บรรทัดล่าง เมื่อไฟเลี้ยวด้านขวาไม่ทำงาน

4.2.6.4 ทดลองเปิด - ปิดสวิตซ์ไฟเบรก ค่าที่ได้สามารถแสดงได้ดังรูปที่ 4.28



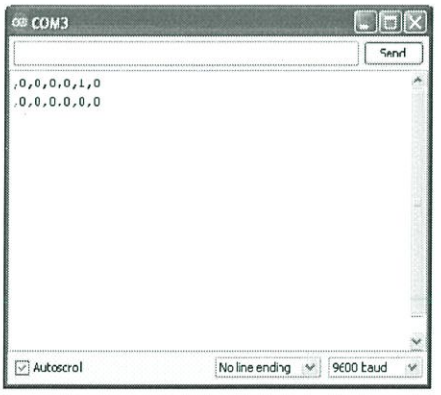
รูปที่ 4.28 ผลของโปรแกรมเมื่อเปิด – ปิดสวิตซ์ไฟเบรก
บรรทัดบน เมื่อสวิตซ์ไฟเบรกทำงาน
บรรทัดล่าง เมื่อสวิตซ์ไฟเบรกไม่ทำงาน

4.2.6.5 ทดลองเปิด – ปิดไฟด้านหลังค่าที่สามารถแสดงได้ดังรูปที่ 4.29



รูปที่ 4.29 ผลของโปรแกรมเมื่อเปิด – ปิดไฟด้านหลัง
บรรทัดบน เมื่อไฟด้านหลังทำงาน
บรรทัดล่าง เมื่อไฟด้านหลังไม่ทำงาน

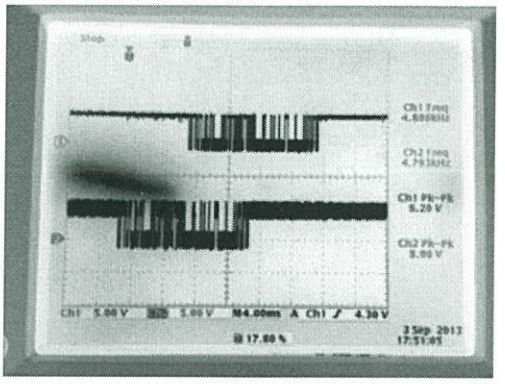
4.2.6.6 ทดลองเปิด – ปิดไฟสัญญาณถอยหลัง ค่าที่ได้สามารถแสดงได้ดังรูปที่ 4.30



รูปที่ 4.30 ผลของโปรแกรมเมื่อเปิด – ปิดไฟสัญญาณถอยหลัง
บรรทัดบน เมื่อไฟสัญญาณถอยหลังทำงาน
บรรทัดล่าง เมื่อไฟสัญญาณถอยหลังไม่ทำงาน

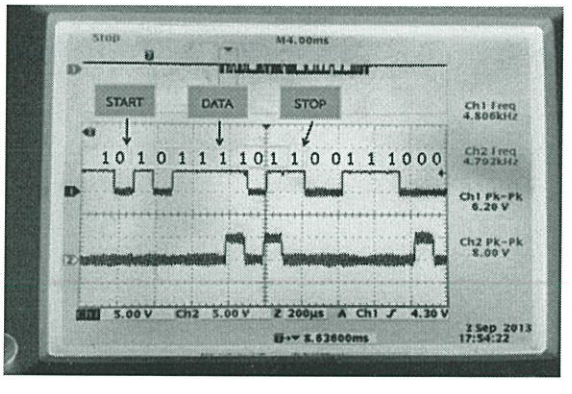
4.3 ส่วนส่งข้อมูลเข้าโทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์

จากการต่อวงจรดังรูปที่ 3.37 ทำการวัดสัญญาณที่ขา 3 ของอาร์เอฟไอดีเทียบกับสัญญาณที่ขา 4 ของบลูธูทโมดูลได้สัญญาณดังรูปที่ 4.31



รูปที่ 4.31 สัญญาณจากขา 3 ของอาร์เอฟไอดีกับขา 4 ของบลูธูทโมดูล
บน ขา 3 ของ อาร์เอฟไอดี
ล่าง ขา 4 ของ บลูธูทโมดูล

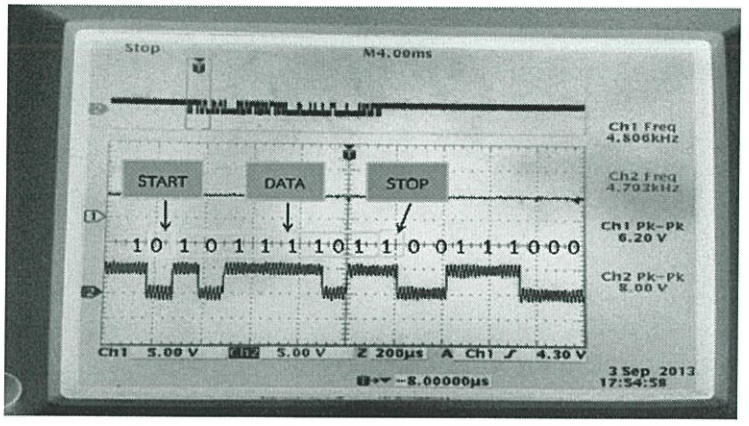
อ่านค่าสัญญาณที่ได้โดย 1 ชุดสัญญาณจะประกอบด้วยบิต 0 และ 1 จำนวน 10 บิตโดยบิต 0 ตัวแรกเป็น บิตเริ่มต้นตามด้วยข้อมูล 8 บิตและปิดด้วยบิต 1 เป็นบิตสิ้นสุด ทำการอ่านสัญญาณชุดแรกที่ขา 3 ของอาร์เอฟไอดี จะได้ข้อมูล 8 บิตคือ 10111101 ซึ่งเมื่อแปลงเป็นเลขฐาน 16 จะได้ BD ดังรูปที่ 4.32



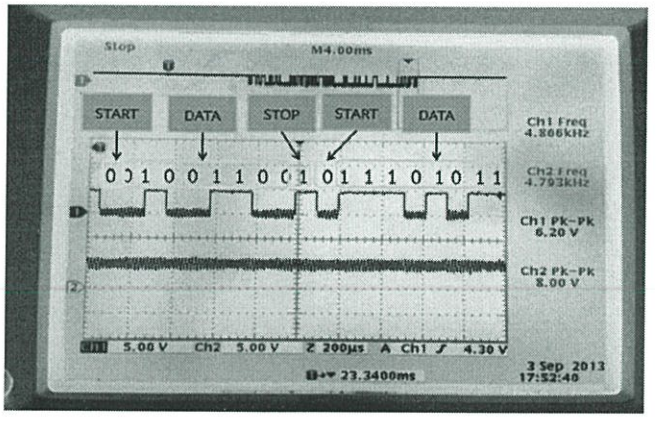
รูปที่ 4.32 สัญญาณชุดแรกจากขา 3 ของอาร์เอฟไอดี
บน สัญญาณทั้งหมดจากขา 3 ของอาร์เอฟไอดี
กลาง สัญญาณชุดแรกที่ถูกขยายของขา 3 ของอาร์เอฟไอดี
ล่าง สัญญาณที่ถูกขยายของขา 4 ของบลูธูทโมดูล

และเมื่อเทียบกับสัญญาณชุดแรกที่ขา 4 ของบลูธูทโมดูลจะได้ข้อมูล 8 บิตนั้นคือ 10111101 ซึ่งเมื่อแปลงเป็นเลขฐาน 16 จะได้ BD เช่นกันดังรูปที่ 4.33

ทำการอ่านสัญญาณชุดสุดท้ายที่ขา 3 ของ อาร์เอฟไอดี จะได้ข้อมูล 8 บิตคือ 11010111 ซึ่งเมื่อแปลงเป็นเลขฐาน 16 จะได้ D7 ดังรูปที่ 4.34

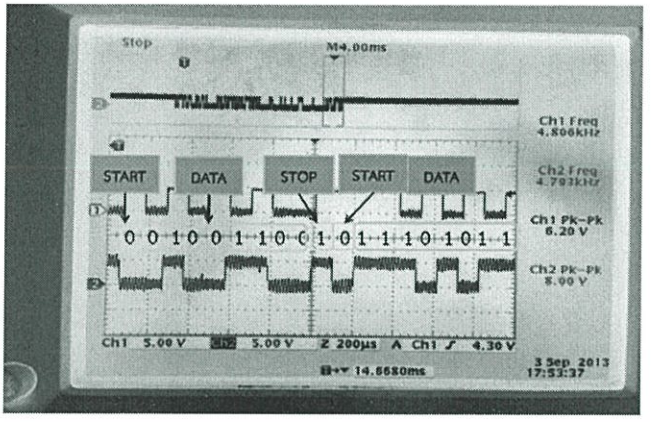


รูปที่ 4.33 สัญญาณชุดแรกจากขา 4 ของบลูธูทโมดูล
บน สัญญาณทั้งหมดจากขา 4 ของบลูธูทโมดูล
กลาง สัญญาณที่ถูกขยายของขา 3 ของอาร์เอฟไอดี
ล่าง สัญญาณชุดแรกที่ถูกขยายของขา 4 ของบลูธูทโมดูล



รูปที่ 4.34 สัญญาณชุดสุดท้ายจากขา 3 ของอาร์เอฟไอดี
บน สัญญาณทั้งหมดจากขา 3 ของ อาร์เอฟไอดี
กลาง สัญญาณชุดสุดท้ายที่ถูกขยายของขา 3 ของ อาร์เอฟไอดี
ล่าง สัญญาณที่ถูกขยายของขา 4 ของ บลูธูทโมดูล

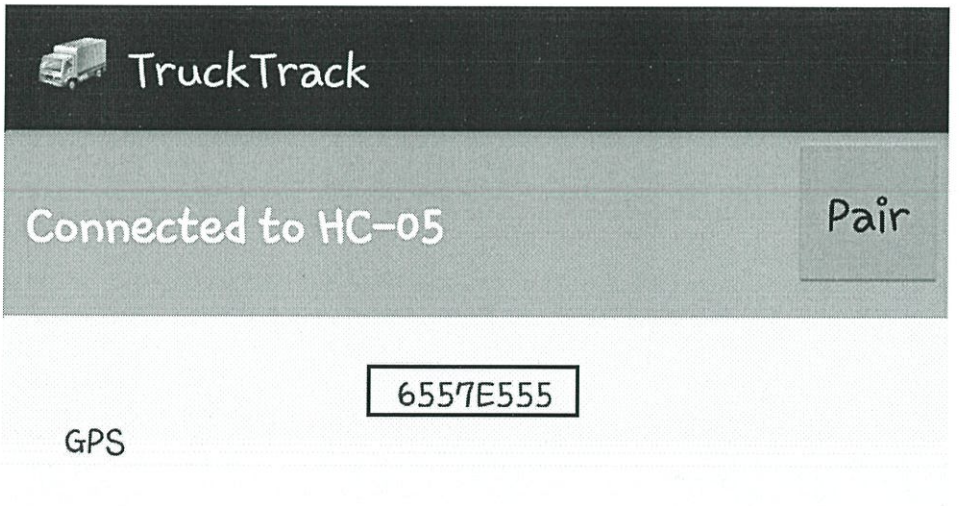
และเมื่อเทียบกับสัญญาณชุดสุดท้ายที่ขา 4 ของ บลูธูทโมดูลจะได้ข้อมูล 8 บิตคือ 11010111 ซึ่งเมื่อแปลงเป็นเลขฐาน 16 จะได้ D7 เช่นกันดังรูปที่ 4.35



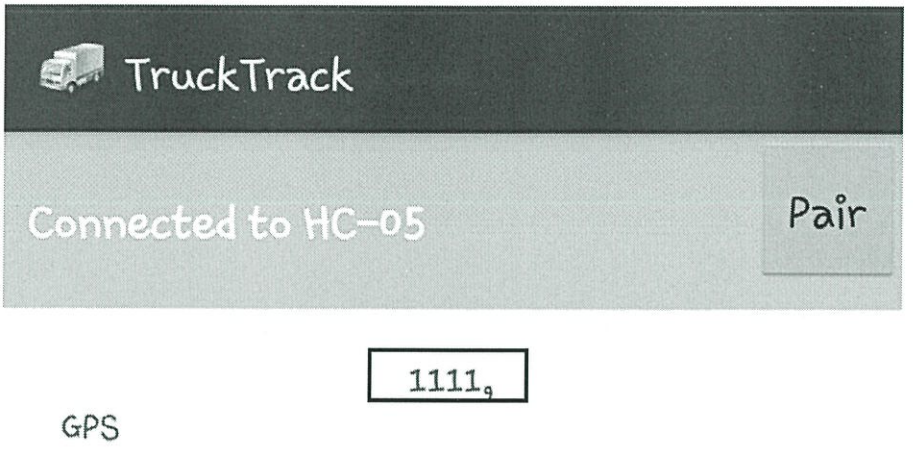
รูปที่ 4.35 สัญญาณชุดสุดท้ายจากขา 4 ของบลูธูทโมดูล
บน สัญญาณทั้งหมดจากขา 4 ของบลูธูทโมดูล
กลาง สัญญาณที่ถูกขยายของขา 3 ของอาร์เอฟไอดี
ล่าง สัญญาณชุดสุดท้ายที่ถูกขยายของขา 4 ของบลูธูทโมดูล

4.4 ชุดส่งข้อมูลเข้าฐานข้อมูล

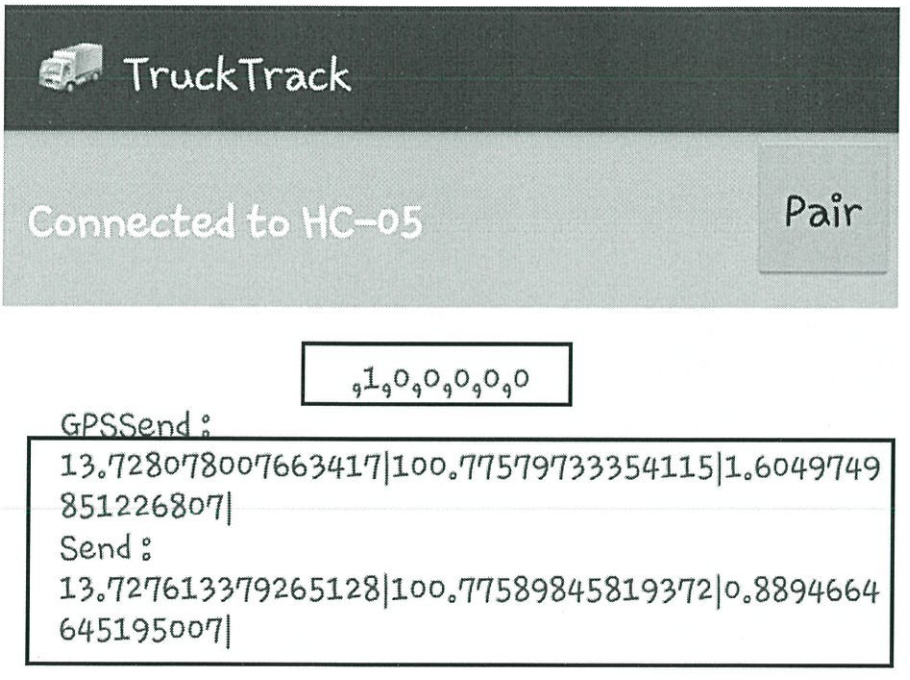
ข้อมูลไอดีจากอาร์เอฟไอดี แป้นกด และค่าเซนเซอร์ที่โทรศัพท์มือถือรับมาจากไมโครคอนโทรลเลอร์ผ่านบลูธูทและค่าพิกัดจีพีเอสที่โทรศัพท์มือถืออ่านค่าได้จะปรากฏค่าในหน้าต่างของแอพพลิเคชั่นดังรูปที่ 4.36 4.37 และ 4.38 ตามลำดับ



รูปที่ 4.36 ค่าของแท็กอาร์เอฟไอดีที่ไมโครคอนโทรลเลอร์ส่งมา

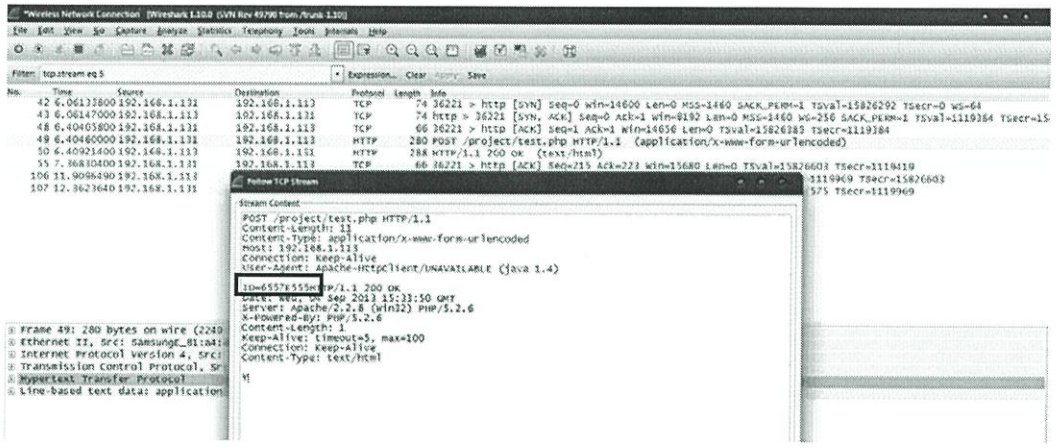


รูปที่ 4.37 ค่าของแป้นกดที่ไมโครคอนโทรลเลอร์ส่งมา

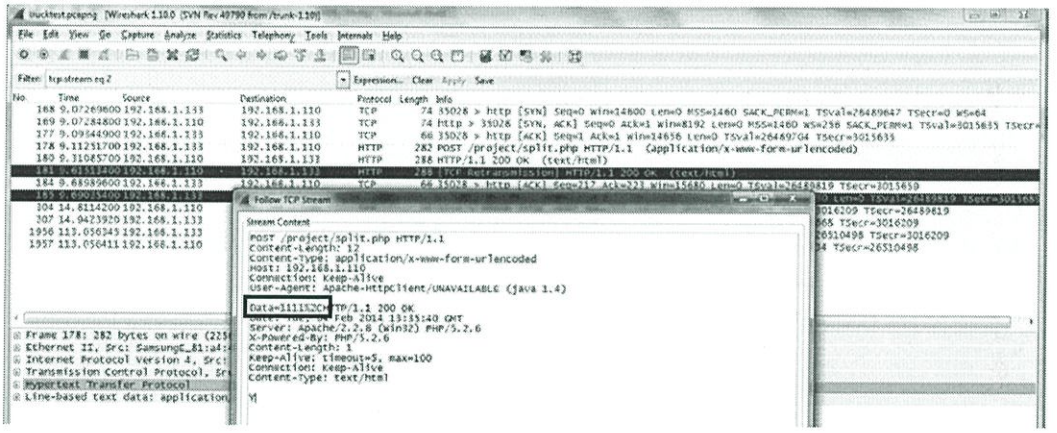


รูปที่ 4.38 ค่าที่แอพลิเคชันแสดงเมื่อรับค่าข้อมูล
บน ค่าจากเซนเซอร์
ล่าง ค่าจากจีพีเอส

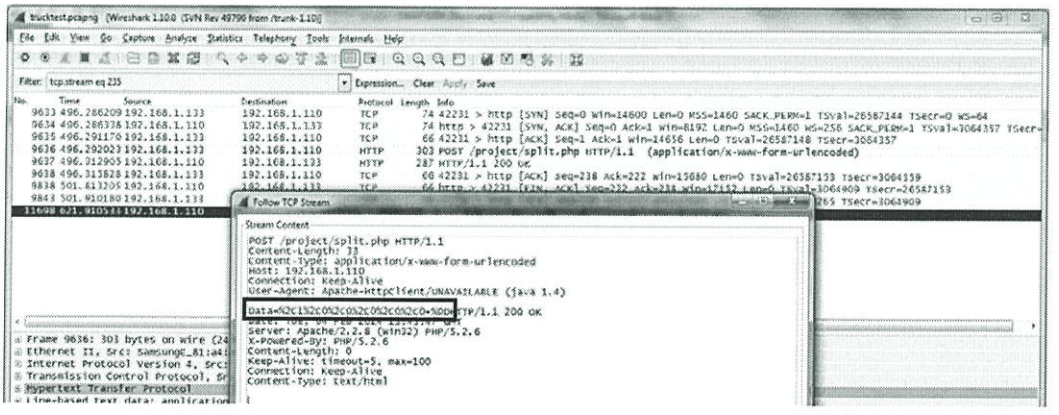
จากนั้นโทรศัพท์มือถือจะส่งค่าทั้งหมดดังกล่าวไปยังฐานข้อมูลโดยในฝั่งของฐานข้อมูลจะใช้โปรแกรมวายชาร์กในการดักจับแพ็กเกจข้อมูลดังรูปที่ 3.38 ซึ่งข้อมูลที่ได้รับนั้นจะแสดงได้ดังรูปที่ 4.39 4.40 4.41 และ 4.42 ตามลำดับ



รูปที่ 4.39 แพ็กเกจข้อมูลไอดี (แท็กส์) ที่ดักจับได้ด้วยโปรแกรมวายชาร์ก



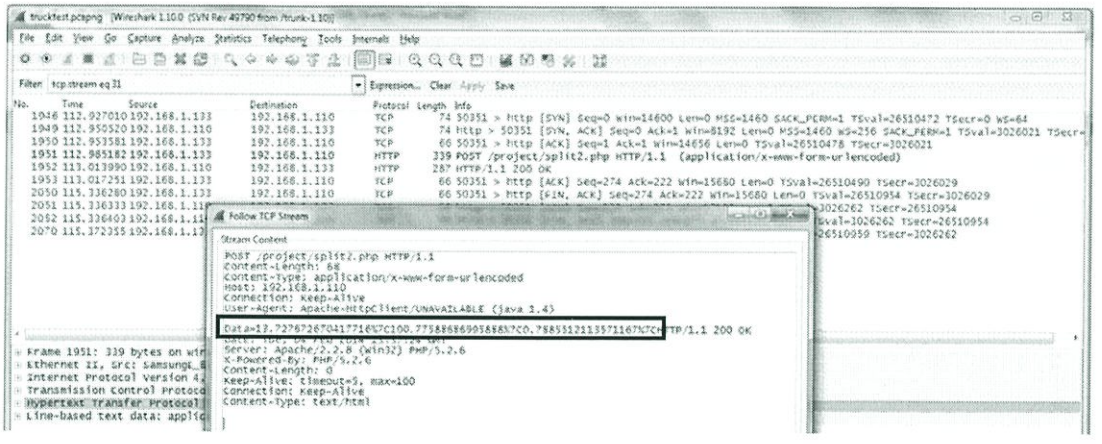
รูปที่ 4.40 แพ็กเกจข้อมูลไอดี (แป้นกด)ที่ดักจับได้ด้วยโปรแกรมวายชาร์ก



รูปที่ 4.41 แพ็กเกจข้อมูลเซนเซอร์ที่ดักจับได้ด้วยโปรแกรมวายชาร์ก

จากรูปที่ 4.41 เป็นการส่ง ,1,0,0,0,0,0 ซึ่งก็คือการเปิดการทำงานของไฟหน้า โดยแพ็กเก็จที่จับได้คือ %2C1%2C0%2C0%2C0%2C0%2C0+%0D ซึ่งจากแพ็กเก็จดังกล่าวสามารถอ่านค่าได้ดังนี้

1	คือ มีการทำงานของไฟหน้า
0	คือ ไม่มีการทำงานของไฟเลี้ยวซ้าย-ขวา ไฟหลัง ไฟถอย และระบบเบรก
%2C	เป็นเลขฐาน 16 ซึ่งเมื่อนำไปเทียบกับตาราง ASCII คือตัวอักษร ‘,’
%0D	เป็นเลขฐาน 16 ซึ่งเมื่อนำไปเทียบกับตาราง ASCII คือการขึ้นบรรทัดใหม่



รูปที่ 4.42 แพ็กเกจข้อมูลจีพีเอสที่ดักจับได้ด้วยโปรแกรมวายชาร์ก

จากรูปที่ 4.42 เป็นการส่ง 13.727672670417716|7C100.77588686995888|7C0.7885512113571167| ซึ่งก็คือพิกัดจีพีเอสที่โทรศัพท์สามารถอ่านค่าได้ โดยแพ็กเก็จที่จับได้คือ 13.727672670417716%7C100.77588686995888%7C0.7885512113571167%7C ซึ่งจากแพ็กเก็จดังกล่าวสามารถอ่านค่าได้ดังนี้

13.727672670417716	คือ ค่าละติจูด
100.77588686995888	คือ ค่าลองติจูด
0.7885512113571167	คือ ค่าความเร็ว

%0 เป็นเลขฐาน 16 ซึ่งเมื่อนำไปเทียบกับตาราง ASCII คือตัวอักษร ‘|’

ทำการตรวจสอบค่าทั้งหมดในฐานข้อมูลซึ่งพบว่าค่าข้อมูลที่ได้รับมานั้นมีค่าตรงกันดังรูปที่ 4.43 4.44 4.45 และ 4.46 ตามลำดับ

แสดง : 30 แถว เริ่มจากแถวที่ 30 > >> หมายเลขหน้า: 1

อยู่ใน แนวนอน และซ้ำหัวแถวทุกๆ 100 เซลล์

เรียงโดยคีย์: ไม่มี ลงมือ

	Seq	time	ID	Lat	Lon	Spd	TL	FL	RL	BL
	287	30/08/13 (18:04:39)	6557E555	0	0	0				
	286	30/08/13 (18:04:39)	6557E555	0	0	0				

รูปที่ 4.43 ข้อมูลไอดี (แท็กส์) ในฐานข้อมูล

	Seq	time	ID	FL	LL	RL	BL	Rev	BR
	586	04/02/14 (20:35:40)	1111						

รูปที่ 4.44 ข้อมูลไอดี (แป้นกด) ในฐานข้อมูล

	Seq	time	ID	FL	LL	RL	BL	Rev	BR
	782	04/02/14 (20:43:57)		0	0	0	0	0	0
	781	04/02/14 (20:43:50)		1	0	0	1	0	1
	780	04/02/14 (20:43:47)		1	0	0	0	0	0

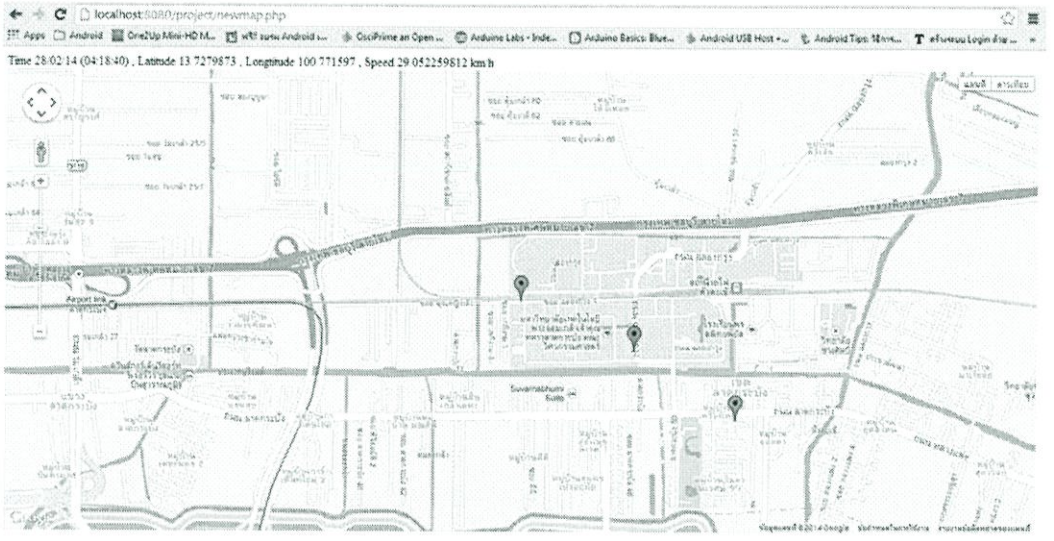
รูปที่ 4.45 ข้อมูลเซนเซอร์ในฐานข้อมูล

	Seq	time	Lat	Lon	Spd
	66	04/02/14 (20:39:47)	13.7276133	100.775898	0.88946646
	65	04/02/14 (20:39:30)	13.7280780	100.775797	1.60497498
	64	04/02/14 (20:38:15)	13.7282955	100.775824	1.73635208
	63	04/02/14 (20:37:28)	13.7279602	100.775906	2.51034975
	62	04/02/14 (20:37:24)	13.7276726	100.775886	0.78855121

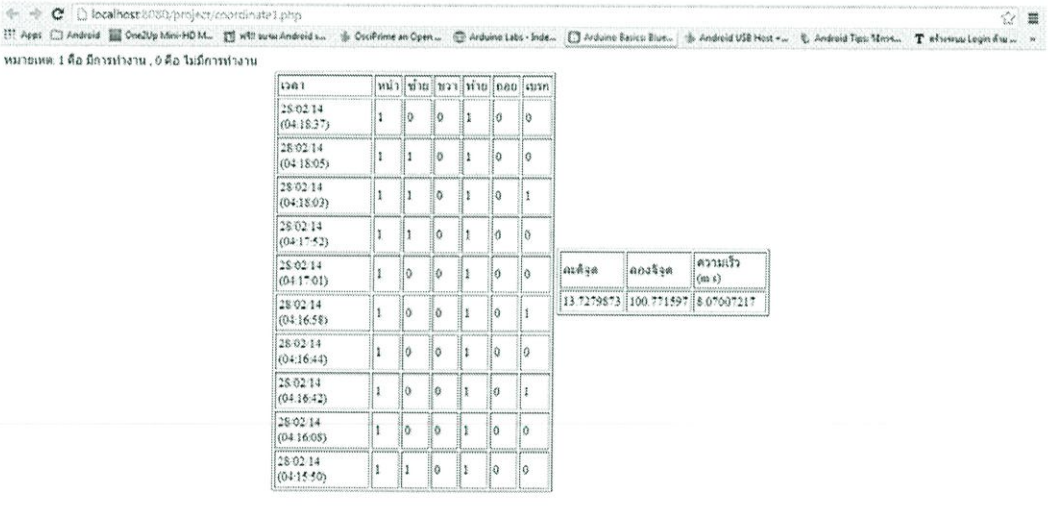
รูปที่ 4.46 ข้อมูลพิกัดจีพีเอสในฐานข้อมูล

4.5 การแสดงผลผ่านทางหน้าเวปเพจ

เวปเพจจะทำการแสดงผลค่าพิกัด 3 ค่าล่าสุดที่ฐานข้อมูลเก็บไว้ผ่านทางแผนที่กูเกิ้ลดังรูปที่ 4.47 โดยสามารถตรวจสอบข้อมูลการใช้งานสัญญาณไฟและเบรกได้โดยการกดที่พิกัด ซึ่งจะทำการแสดงผลในอีกหน้าต่างหนึ่งดังรูปที่ 4.48



รูปที่ 4.47 หน้าเว็บเพจแสดงค่าพิกัด 3 ค่าล่าสุด



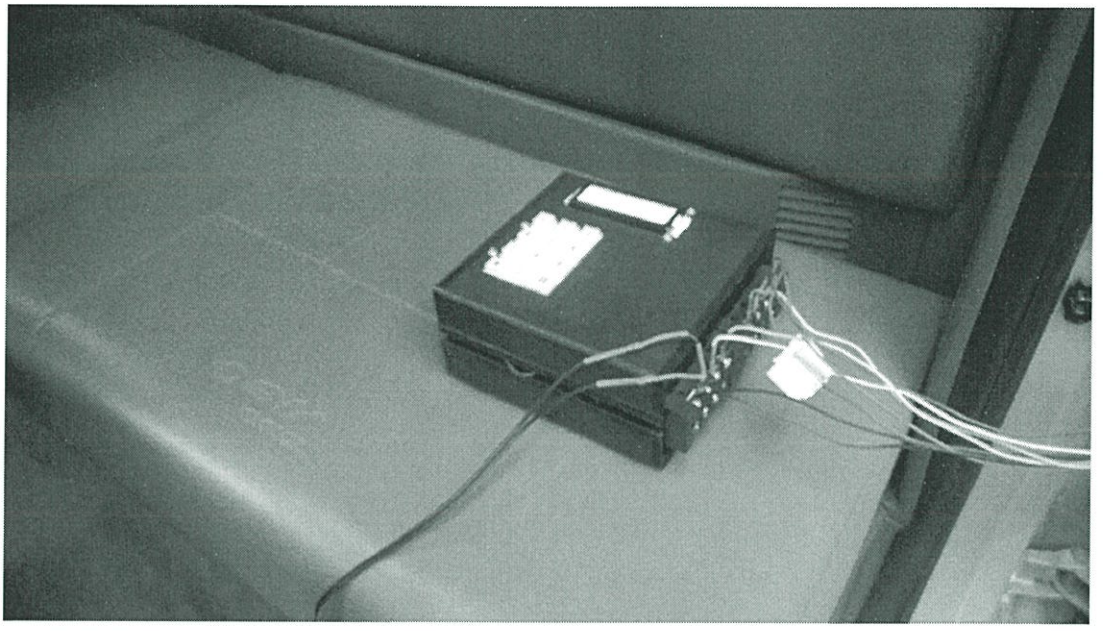
เวลา	หน้า	ซ้าย	ขวา	หัน	ถอย	เบรก
28/02/14 (04:18:37)	1	0	0	1	0	0
28/02/14 (04:18:05)	1	1	0	1	0	0
28/02/14 (04:18:03)	1	1	0	1	0	1
28/02/14 (04:17:52)	1	1	0	1	0	0
28/02/14 (04:17:01)	1	0	0	1	0	0
28/02/14 (04:16:58)	1	0	0	1	0	1
28/02/14 (04:16:44)	1	0	0	1	0	0
28/02/14 (04:16:42)	1	0	0	1	0	1
28/02/14 (04:16:08)	1	0	0	1	0	0
28/02/14 (04:15:50)	1	1	0	1	0	0

ละติจูด	ลองจิจูด	ความเร็ว (m/s)
13.7279873	100.771597	8.0700217

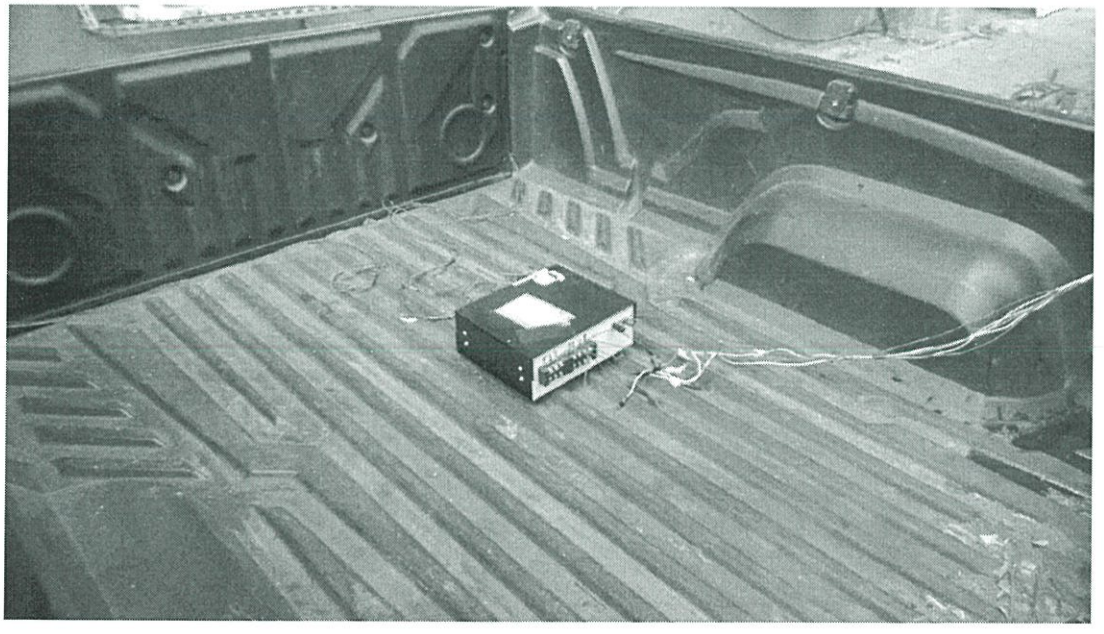
รูปที่ 4.48 หน้าเว็บเพจแสดงค่าการใช้งานสัญญาณไฟและเบรก

4.6 ผลการทดลองรวมทั้งระบบกับรถยนต์บรรทุก

ทำการติดตั้งอุปกรณ์บนรถยนต์บรรทุกดังรูปที่ 4.49 และรูปที่ 4.50 จากนั้นเริ่มต้นระบบโดยการยืนยันตัวตนด้วยกดแป้นกด จะขึ้นคำตอบรับที่หน้าจอแอลซีดี ดังรูปที่ 4.51



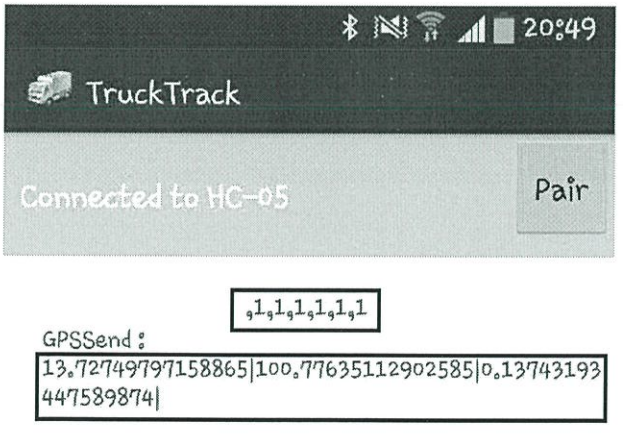
รูปที่ 4.49 กล่องอุปกรณ์ที่อยู่ในรถยนต์บรรทุก (ใช้ในการยืนยันตัวตน)



รูปที่ 4.50 กล่องอุปกรณ์ที่อยู่หลังรถยนต์บรรทุก (ใช้เก็บข้อมูลสัญญาณไฟและเบรก)

รูปที่ 4.51 ผลหน้าจอแอลซีดีจากการกดแป้นกด

หลังจากการยืนยันตัวตนถ้าข้อมูลผู้ขับถูกต้องระบบจะเริ่มทำการเก็บค่าการใช้งานสัญญาณไฟและเบรก ซึ่งจะถูกส่งจากไมโครคอนโทรลเลอร์ไปยังโทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์ผ่านบลูธูท โดยสามารถตรวจสอบค่าได้จากหน้าต่างของแอพพลิเคชันและโทรศัพท์จะอ่านค่าพิกัดจีพีเอสที่อยู่ปัจจุบันแล้วแสดงผลในหน้าต่างเดียวกัน ดังรูปที่ 4.52 จากนั้นข้อมูลทั้งหมดจะถูกส่งไปเก็บในฐานข้อมูลผ่านอินเตอร์เน็ต ซึ่งสามารถตรวจสอบได้ ดังรูปที่ 4.53 และ 4.54 (ในกรอบสี่เหลี่ยม)



รูปที่ 4.52 หน้าต่างของแอพพลิเคชันแสดงค่าข้อมูล
บน ค่าการใช้งานสัญญาณไฟและเบรก
ล่าง ค่าพิกัดจีพีเอส

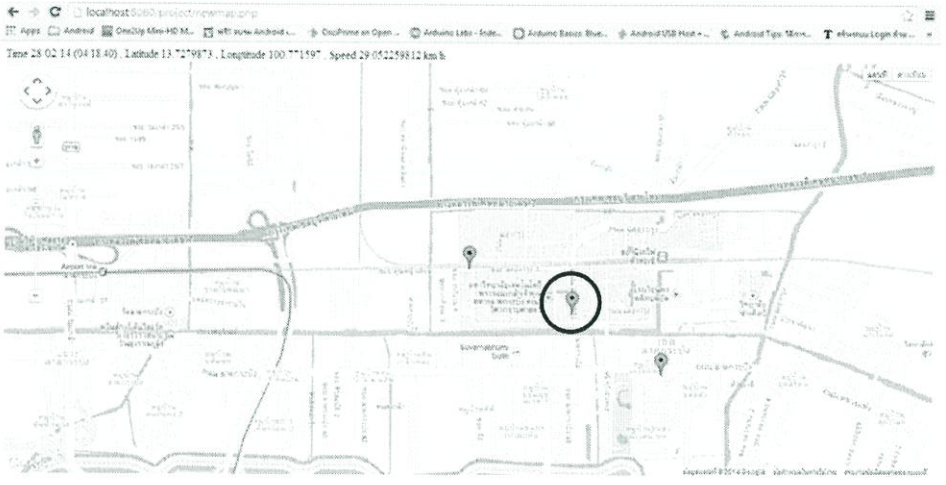
←T→	Seq	time	ID	FL	LL	RL	BL	Rev	BR
✎ ✕	19	28/02/14 (04:10:22)	1111						
✎ ✕	20	28/02/14 (04:12:30)		1	0	1	1	0	0
✎ ✕	21	28/02/14 (04:13:02)		1	0	0	1	0	0
✎ ✕	22	28/02/14 (04:13:50)		1	1	0	1	0	0
✎ ✕	23	28/02/14 (04:15:08)		1	0	0	1	0	1
✎ ✕	24	28/02/14 (04:15:11)		1	0	0	1	0	0
✎ ✕	25	28/02/14 (04:14:05)		1	1	0	1	0	1
✎ ✕	26	28/02/14 (04:14:08)		1	0	0	1	0	0
✎ ✕	27	28/02/14 (04:15:50)		1	1	0	1	0	0
✎ ✕	28	28/02/14 (04:16:08)		1	0	0	1	0	0
✎ ✕	29	28/02/14 (04:16:42)		1	0	0	1	0	1
✎ ✕	30	28/02/14 (04:16:44)		1	0	0	1	0	0
✎ ✕	31	28/02/14 (04:16:58)		1	0	0	1	0	1
✎ ✕	32	28/02/14 (04:17:01)		1	0	0	1	0	0
✎ ✕	33	28/02/14 (04:17:52)		1	1	0	1	0	0
✎ ✕	34	28/02/14 (04:18:03)		1	1	0	1	0	1
✎ ✕	35	28/02/14 (04:18:05)		1	1	0	1	0	0
✎ ✕	36	28/02/14 (04:18:37)		1	0	0	1	0	0

รูปที่ 4.53 ผลข้อมูลเซนเซอร์ที่แสดงในฐานข้อมูล

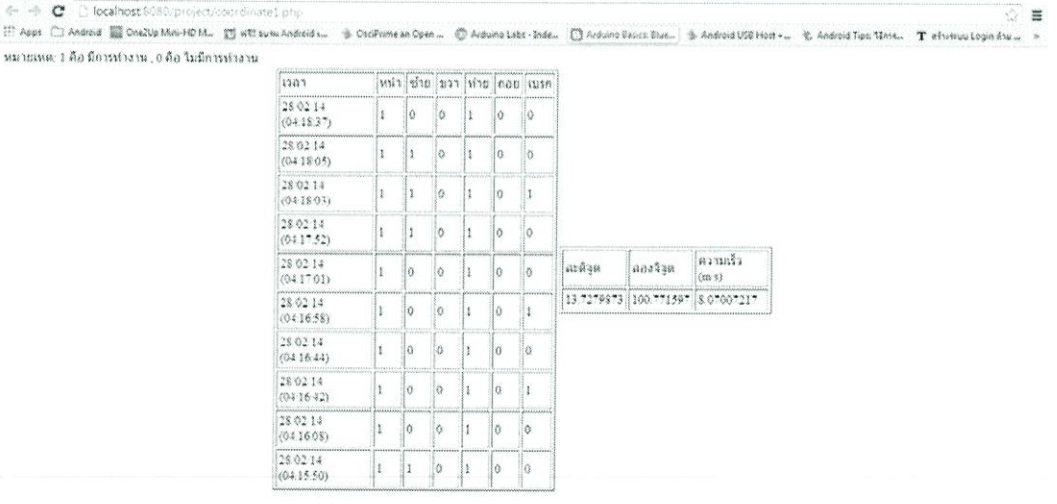
←T→	Seq	time	Lat	Lon	Spd
✎ ✕	183	28/02/14 (04:18:40)	13.7279873	100.771597	8.07007217
✎ ✕	182	28/02/14 (04:16:50)	13.7253130	100.777987	8.14598369
✎ ✕	181	28/02/14 (04:11:40)	13.7215251	100.783589	1.34001183

รูปที่ 4.54 ข้อมูลจีพีเอสที่แสดงในฐานข้อมูล

ตรวจสอบการแสดงผลข้อมูลพิกัด 3 ค่าล่าสุดผ่านหน้าเว็บเพจบนแผนที่กูเกิ้ล ดังรูปที่ 4.55 และตรวจสอบค่าการทำงานของสัญญาณไฟและเบรก ซึ่งจะแสดงในหน้าต่างใหม่ โดยการกดที่พิกัด ดังรูปที่ 4.56



รูปที่ 4.55 หน้าเว็บเพจแสดงค่าพิกัดบนแผนที่กูเกิ้ล



เวลา	หน้า	ซ้าย	ขวา	ท้าย	ถอย	เบรก
28 02 14 (04 18 37)	1	0	0	1	0	0
28 02 14 (04 18 05)	1	1	0	1	0	0
28 02 14 (04 18 03)	1	1	0	1	0	1
28 02 14 (04 17 52)	1	1	0	1	0	0
28 02 14 (04 17 01)	1	0	0	1	0	0
28 02 14 (04 16 58)	1	0	0	1	0	1
28 02 14 (04 16 44)	1	0	0	1	0	0
28 02 14 (04 16 42)	1	0	0	1	0	1
28 02 14 (04 16 08)	1	0	0	1	0	0
28 02 14 (04 15 50)	1	1	0	1	0	0

ละติจูด	ลองจิจูด	ความเร็ว (m/s)
13.72798 73	100.77 1597	8.07007217

รูปที่ 4.56 หน้าเว็บเพจแสดงค่าการใช้งานสัญญาณไฟและเบรก

บทที่ 5

สรุปผลและข้อเสนอแนะ

5.1 สรุปผล

จากการจัดทำปริญญานิพนธ์ระบบเก็บข้อมูลการใช้งานและติดตามรถยนต์บรรทุกซึ่งได้ทำการศึกษาหลักการต่างๆ เพื่อนำมาประยุกต์ใช้ในการจัดเก็บข้อมูลการใช้งานรถยนต์บรรทุก สามารถสรุปได้ดังนี้

1. ระบบสามารถทำการยืนยันตัวตนผู้ขับรถยนต์บรรทุกโดยใช้อาร์เอฟไอดีหรือแป้นกด
2. ระบบสามารถตรวจจับการทำงานของระบบเบรกและสัญญาณไฟ (ไฟหน้า ไฟท้าย ไฟถอยหลัง ไฟเลี้ยวซ้าย-ขวา) โดยข้อมูลจะถูกส่งไปยังโทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์ผ่านทางบลูธูทโมดูล
3. ระบบสามารถติดตามตำแหน่งของรถยนต์บรรทุกด้วยพิกัดจีพีเอสโดยใช้แอพพลิเคชันบนโทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์
4. ข้อมูลทั้งหมด (การเบรก สัญญาณไฟ และพิกัดจีพีเอส) จะถูกส่งไปเก็บในฐานข้อมูลผ่านระบบอินเตอร์เน็ต
5. ระบบสามารถแสดงผลการติดตามรถยนต์บรรทุกจากพิกัดจีพีเอสผ่านทางแผนที่กูเกิ้ลบนหน้าเวปเพจ และสามารถตรวจสอบข้อมูลการใช้งานระบบเบรกและสัญญาณไฟในช่วงเวลานั้นได้จากการกดที่พิกัดบนแผนที่

5.2 ปัญหาและข้อเสนอแนะ

1. ในกรณีที่เครื่องยนต์ดับ เมื่อสตาร์ทเครื่องใหม่จะต้องทำการยืนยันตัวตนใหม่อีกครั้งระบบถึงจะทำงานได้
2. ไม่สามารถรับพิกัดจีพีเอสได้ เมื่อรถยนต์บรรทุกวิ่งผ่านอุโมงค์หรือใต้สะพาน
3. สามารถเพิ่มประสิทธิภาพของระบบได้จากการเขียนโปรแกรมประมวลผลความสัมพันธ์ระหว่างข้อมูลพิกัดจีพีเอสกับข้อมูลการเบรกและสัญญาณไฟ เพื่อแจ้งเตือนบนหน้าเวปเพจ เช่น การเบรกกะทันหัน การเลี้ยวโดยไม่ใช้สัญญาณไฟ เป็นต้น

บรรณานุกรม

[1] พร้อมเลิศ หล่อวิจิตร. *คู่มือเขียนแอพ Android สำหรับผู้เริ่มต้น.* กรุงเทพฯ : โปรวิชั่น, 2555.

[2] สมศักดิ์ โชคชัยชุติกุล. *อินไซท์ PHP 5.* กรุงเทพฯ : โปรวิชั่น, 2547.

[3] เอกชัย มะการ. *เรียนรู้และเข้าใจใช้งานไมโครคอนโทรลเลอร์ตระกูล AVR ด้วย Arduino.* กรุงเทพฯ : บริษัทอีทีทีจำกัด, 2552.

ภาคผนวก

RFID MODULE

ISO15693 Reader / Writer

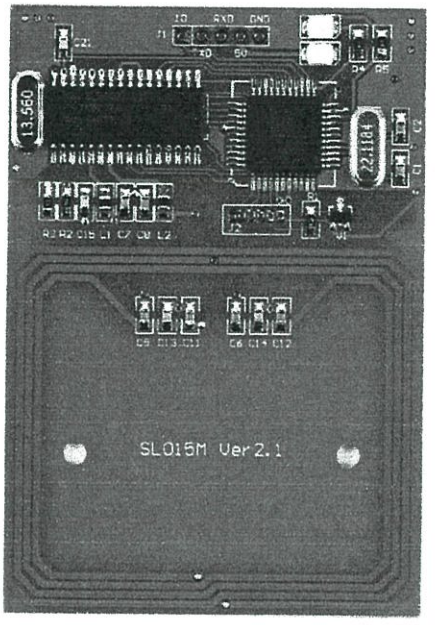
SL015M-3

User Manual

Version 2.0
Dec 2011
StrongLink

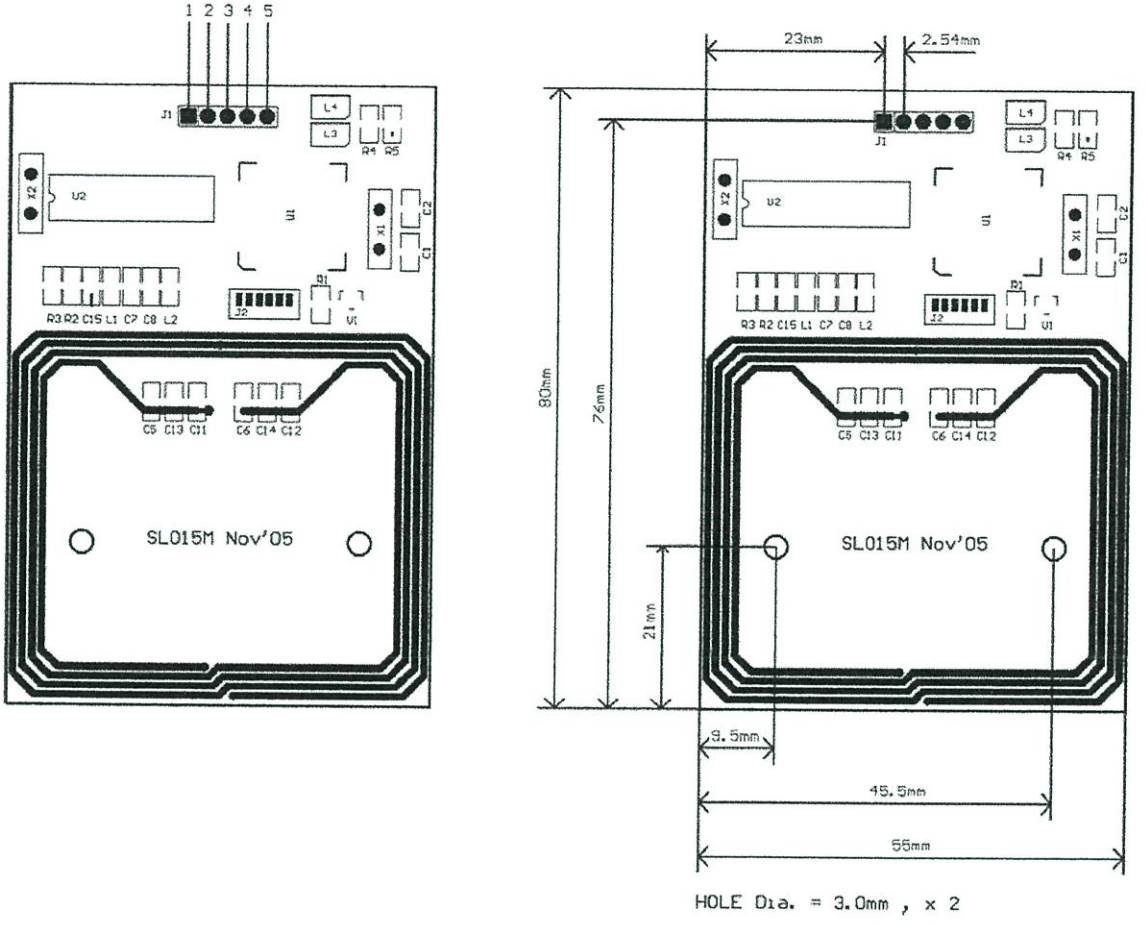
CONTENS

1. MAIN FEATURES



- Tags supported: I.CODE SLI, Tag_it HF I
- Auto detecting tag
- Integrated antenna
- UART interface, baud rate 9,600 ~ 115,200 bps
- DC4.5V to DC5.5V VDD operating
- Operating distance: Up to 80mm, depending on tag
- Storage temperature: -40 ℃ ~ +85 ℃
- Operating temperature: -20 ℃ ~ +70 ℃
- Dimension: 80 × 55 × 7 mm
- Two LEDs, green led is auto light when tag in detection range, red led is controlled by host
- The OUT pin at low level indicates tag in detective range, and high level indicating tag out

2. PINNING INFORMATION



PIN	SYMBOL	TYPE	DESCRIPTION
1	TagSta	Output	Tag detect signal low level indicating tag in detection range high level indicating tag out
2	TXD	Output	Serial output port
3	RXD	Input	Serial input port
4	VCC	PWR	Power Supply
5	GND	PWR	Ground

3. BAUD RATE SETTING

R6 & R7 are two 0 ohm resistances assembled on the bottom layer of module, are used for config baud rate as follows sheet

	R6	R7	Baud rate bps
Assembled	no	no	9,600
	yes	no	19,200
	no	yes	57,600
	yes	yes	115,200

4. Communication Protocol

4-1. Communication setting

The communication protocol is byte oriented. Both sending and receiving bytes are in hexadecimal format. The communication parameters are as follows,

Baud rate: 9,600 ~ 115,200 bps
Data: 8 bits
Stop: 1 bit
Parity: None
Flow control: None

4-2. Communication Format

Host to Reader:

Preamble	Len	Command	Data	Checksum

Preamble: 1 byte equal to 0xBA
Len: 1 byte indicating the number of bytes from Command to Checksum
Command: 1 byte Command code, see Table 3
Data: Variable length depends on the command type
Checksum: 1 byte XOR of all the bytes from Preamble to Data

Reader to Host:

Preamble	Len	Command	Status	Data	Checksum

Preamble: 1 byte equal to 0xBD
Len: 1 byte indicating the number of bytes from Command to Checksum
Command: 1 byte Command code, see Table 3
Status: 1 byte Command status, see Table 4
Data: Variable length depends on the command type.
Checksum: 1 byte XOR of all the bytes from Preamble to Data

4-4. Command List

4-4-1. Get tag information

0xBA	Len	0x31	Checksum

Response:

0xBD	Len	0x31	Status	UID	DSFID	AFI	Type	Checksum

Status: 0x00: Operation succeed
0x01: No tag
0x04: Read fail
0xF0: Checksum error
UID: The Unique Identifier of card, 8 bytes
AFI: The Application Family Identifier, 1byte
DSFID: The Data Storage Format Identifier, 1byte
Type: 0x31: Tag_it HF I
0x32: I.CODE SLI

4-4-2. Get block security status

0xBA	Len	0x32	block	number	Checksum

block: Start block number
number: Number of blocks to be read

Response:

0xBD	Len	0x32	Status	Data	Checksum

Status: 0x00: Operation succeed
0x01: No tag
0x04: Read fail
0xF0: Checksum error
Data: Security status, 1 byte to 1 block

4-4-3. Read blocks

0xBA	Len	0x33	block	number	Checksum

block: Start block number
number: Number of blocks to be read, max 15 blocks

Response:

0xBD	Len	0x33	Status	Data	Checksum

Status: 0x00: Operation succeed
0x01: No tag
0x04: Read fail
0xF0: Checksum error
Data: Blocks data returned if operation succeeds, 4 bytes to 1 block

4-4-4. Write data to a block

0xBA	Len	0x34	Block	Data	Checksum

Block: The block number to be written, 1 byte.
Data: The data to write, 4 bytes.

Response:

0xBD	Len	0x34	Status	Data	Checksum

Status: 0x00: Operation succeed
0x01: No tag
0x05: Write fail
0x06: Unable to read after write
0x07: Read after write error
0xF0: Checksum error

Data: Block data written if operation succeeds, 4 bytes.

4-4-5. Write AFI

0xBA	Len	0x35	Data	Checksum

Data: The AFI data to write, 1 bytes.

Response:

0xBD	Len	0x35	Status	Data	Checksum

Status: 0x00: Operation succeed
0x01: No tag
0x05: Write fail
0x06: Unable to read after write
0x07: Read after write error
0xF0: Checksum error

Data: AFI data written if operation succeeds, 1 bytes.

4-4-6. Write DSFID

0xBA	Len	0x36	Data	Checksum

Data: The DSFID data to write, 1 bytes.

Response:

0xBD	Len	0x36	Status	Data	Checksum

Status: 0x00: Operation succeed
0x01: No tag
0x05: Write fail
0x06: Unable to read after write
0x07: Read after write error
0xF0: Checksum error

Data: DSFID data written if operation succeeds, 1 bytes.

4-4-7. Lock block

0xBA	Len	0x37	block	Checksum

Block: The block number to be locked, 1 byte.

Response:

0xBD	Len	0x37	Status	Checksum

Status: 0x00: Operation succeed
0x01: No tag
0x11: Lock fail
0xF0: Checksum error

4-4-8. Lock AFI

0xBA	Len	0x38	Checksum

Response:

0xBD	Len	0x38	Status	Checksum

Status: 0x00: Operation succeed
0x01: No tag
0x11: Lock fail
0xF0: Checksum error

4-4-9. Lock DSFID

0xBA	Len	0x39	Checksum

Response:

0xBD	Len	0x39	Status	Checksum

Status: 0x00: Operation succeed
0x01: No tag
0x11: Lock fail
0xF0: Checksum error

4-4-10. Control Red Led

0xBA	Len	0x40	Code	Checksum

Code: 0 command red led turn off , other red led turn on, 1 byte

Response:

0xBD	Len	0x40	Status	Checksum

Status: 0x00: Operation succeed
0xF0: Checksum error

4-4-11. Reset

0xBA	Len	0xFF	Checksum

Response: None

```
/* THE TRUCK USING AND TRACKING STROAGE SYSTEM
Telecommunication Engineering 2013 */

#include <SoftwareSerial.h>
#include <LiquidCrystal.h>

// Sensor pin

int front = A0;
int back  = A3;
int rev   = A4;
int brk   = A5;
int r     = A2;
int l     = A1;

// current state

int fs  = 0;  // LOW
int bs  = 0;  // LOW
int res = 0; // LOW
int brs = 0; // LOW
int rs  = 0;  // LOW
int ls  = 0;  // LOW

// previous state

int lfs  = 0;  // LOW
int lbs  = 0;  // LOW
int lres = 0; // LOW
int lbrs = 0; // LOW
```

```
int lrs  = 0;  // LOW
int lls  = 0;  // LOW

unsigned char data[16], key[10];
int trigval, count = 0;
unsigned char i;
char chk, k;

SoftwareSerial mySerial(3, 4);  // rx, tx
SoftwareSerial youSerial(6, 13);

LiquidCrystal lcd(12, 11, 10, 9, 8, 7);

void setup() {
  pinMode(front, INPUT); //front pin as INPUT
  pinMode(l, INPUT); //left pin as INPUT
  pinMode(r, INPUT); //right pin as INPUT
  pinMode(back, INPUT); //back pin as INPUT
  pinMode(rev, INPUT); //reverse pin as INPUT
  pinMode(brk, INPUT); //brake pin as IPNPUT

  pinMode(13, OUTPUT);    //LED
  pinMode(5, INPUT);       //  trig
  pinMode(6, INPUT);       //  recieve Key
  Serial.begin(9600);       //  Bluetooth
  mySerial.begin(9600);      //  RFID
  youSerial.begin(9600);
  lcd.begin(16, 2);
  lcd.clear();
  lcd.print("Enter Password..");
```

```
 // resetRFID();
  delay(1000);
}
void loop() {
  delay(10);
  fs = digitalRead(front);   //Read front state
  ls = digitalRead(l);     //Read left state
  rs = digitalRead(r);   //Read right state
  bs = digitalRead(back);     //Read back state
  res = digitalRead(rev);       //Read reverse state
  brs = digitalRead(brk);       //Read brake state
  trigval = digitalRead(5);
  if(trigval == LOW) {
    getTag();
    readRFID();
   // initialstate();
    digitalWrite(13,LOW);
    delay(100);
//   count=1;
 }
 else if(trigval == HIGH){
     getKey();
     checkLog();
     if(count >= 1){
       digitalWrite(13,HIGH);
       delay(100);
       //condition
       if (fs != lfs || bs != lbs || brs != lbrs || res != lres || rs != lrs || ls !=
lls)
       {
```

```
          Serial.print(",");
          Serial.print(fs);
          Serial.print(",");
          Serial.print(ls);
          Serial.print(",");
          Serial.print(rs);
          Serial.print(",");
          Serial.print(bs);
          Serial.print(",");
          Serial.print(res);
          Serial.print(",");
          Serial.print(brs);
          Serial.println(" ");
        }
      //write current to previous state
      lfs = fs;
      lbs = bs;
      lres = res;
      lbrs = brs;
      lrs = rs;
      lls = ls;
    }
    else
        digitalWrite(13,LOW);       //////////////////////////
        delay(200);
  }
}
void getTag() {
  unsigned char checksum = 0;
  unsigned char command[] = {0xBA,0x02,0x31};
```

```
  for(i=0;i<sizeof(command);i++) {
    checksum = checksum^command[i];
  }
  for(i=0;i<sizeof(command);i++) {
     mySerial.write(command[i]);
  }
  mySerial.write(checksum);
}
void resetRFID() {
  unsigned char checksum = 0;
  unsigned char command[] = {0xBA,0x02,0xFF};
  for(i=0;i<sizeof(command);i++) {
    checksum = checksum^command[i];
  }
  for(i=0;i<sizeof(command);i++) {
     mySerial.write(command[i]);
  }
  mySerial.write(checksum);
}
void readRFID(){
  if(mySerial.available()>0){
  delay(80);
    while(mySerial.available()) {
      for(i=0 ; i<16 ; i++) {
        data[i] = mySerial.read();
      }
    }
    delay(100);
      for(i=4 ; i<8 ; i++) {
        Serial.print(byte(data[i]), HEX);
```

```
      }
      Serial.print(",");
      Serial.print("\n");
    delay(1000);
  }
}
void checkLog(){
  if(Serial.available()>0) {
    chk = Serial.read();
 //   Serial.print(r);
    if (chk == 'Y'){
      lcd.setCursor(0, 1);
      lcd.print("OK              ");
      count = 1;
  //    initialstate();                  ////////////////////
  //    digitalWrite(13, HIGH);
    }
    if (chk == 'N'){
      lcd.setCursor(0, 1);
      lcd.print("Try Again       ");
  //    digitalWrite(13, LOW);
    }
  }
  delay(200);
}
void getKey(){
  if(youSerial.available()>0){
  delay(80);
    while(youSerial.available()) {
      for(i=0 ; i<10 ; i++) {
```

```
        key[i] = youSerial.read();
      }
    }
    delay(100);
      for(i=0 ; i<4 ; i++) {
        Serial.print(byte(key[i]) - '0');
      }
      lcd.setCursor(0, 1);
      lcd.print("****          ");
      Serial.print(",");
      Serial.print("\n");
    delay(200);
  }
}
/*void initialstate()
{
  int inifront = digitalRead(frontPin);
  inifront = inifront;
  int inileft = digitalRead(leftPin);
  inileft = inileft;
  int iniright = digitalRead(rightPin);
  iniright = iniright;
  int iniback = digitalRead(backPin);
  iniback = iniback;
  int inirev = digitalRead(revPin);
  inirev = inirev;
  int inibrk = digitalRead(brkPin);
  inibrk = inibrk;

  Serial.print(",");
```

```
  Serial.print(inifront);
  Serial.print(",");
  Serial.print(inileft);
  Serial.print(",");
  Serial.print(iniright);
  Serial.print(",");
  Serial.print(iniback);
  Serial.print(",");
  Serial.print(inirev);
  Serial.print(",");
  Serial.print(inibrk);
  Serial.print(",");
  Serial.print("\n");
} */
```