เครื่องบอกตำแหน่งด้วยระบบ GPS แบบบันทึกเสียง

(GPS Recorded-Audio Guiding System)

โดย

นายปัญญา หันตุลา รหัสประจำตัว B4903727

นายณัฐพล นฤมลต์ รหัสประจำตัว B4901822

นางสาวสุพรรณธนา ใจรักษ์ รหัสประจำตัว B4907275

รายงานนี้เป็นส่วนหนึ่งของการศึกษาวิชา 427499 โครงงานวิศวกรรมโทรคมนาคม

หลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมโทรคมนาคม หลักสูตรปรับปรุง พ.ศ. 2545

สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี

ประจำภาคการศึกษาที่ 3 ปีการศึกษา 2552

โครงงาน	เครื่องบอกตำแหน่งด้วยระบบ GPS แบบบันทึกเสียง		
ผู้ดำเนินงาน	1. นายปัญญา	หันตุลา	รหัสประจำตัว B4903727
	2. นายณัฐพล	นฤมลต์	รหัสประจำตัว B4901822
	3. นางสาวสุพรรณธนา	ใจรักษ์	รหัสประจำตัว B4907275
อาจารย์ที่ปรึกษา	ผู้ช่วยศาตราจารย์ ดร.รังสรรค์ ทองทา		
สาขาวิชา	วิศวกรรมโทรคมนาคม		
ภาคการศึกษา	3/2552		

บทคัดย่อ

(Abstract)

ในปัจจุบันเทคโนโลยี GPS ได้ถูกใช้งานอย่างแพร่หลาย และสามารถนำข้อมูลพิกัดจาก GPS มาใช้ประโยชน์ได้อย่างมากมายขึ้นอยู่กับวัตถุประสงค์ของผู้ใช้งาน ทั้งนี้ผู้จัดทำได้นำค่าพิกัดตำแหน่งจาก GPS มาแสดงผลในรูปของข้อมูลเสียง เพื่อประโยชน์ต่อการใช้งานในรูปแบบต่าง ๆ ตามที่ผู้ใช้งานจะประยุกต์ใช้งาน

โครงงานเครื่องบอกตำแหน่งด้วยระบบ GPS แบบบันทึกเสียง ประกอบด้วยส่วนการทำงานต่าง ๆ คือ 1. ส่วนที่เป็นภาครับและภาคส่งสัญญาณ ประกอบด้วยเครื่องรับสัญญาณจีพีเอสเป็นตัวรับสัญญาณจากดาวเทียมเพื่อคำนวณค่าพิกัดตำแหน่ง 2. ส่วนที่เป็นการรายงานผลในรูปแบบเสียง ประกอบด้วยบอร์ดบันทึกและเล่นเสียง 3.หน่วยประมวลผลหรือไมโครคอนโทรลเลอร์ เป็นอุปกรณ์ที่ควบคุมการทำงานของอุปกรณ์ต่าง ๆ ซึ่งเครื่องรบสัญญาณจีพีเอส จะรับสัญญาณจากดาวเทียมเพื่อคำนวณค่าพิกัดตำแหน่ง และทำการตรวจสอบว่าอยู่ในพื้นที่ของพิกัดตำแหน่งอ้างอิงที่ได้ทำการบันทึกหรือไม่ หากอยู่ในพื้นที่ของพิกัดตำแหน่งอ้างอิงจะรายงานผลในรูปแบบเสียง โดยข้อมูลเสียงที่รายงานผลนั้นสามารถบันทึกได้ตามที่ผู้ใช้งานต้องการ

กิตติกรรมประกาศ

จากการที่คณะจัดทำรายงานได้รับมอบหมายให้ทำโครงงานเรื่อง เครื่องบอกตำแหน่งด้วยระบบ GPS แบบบันทึกเสียง ส่งผลให้คณะจัดทำรายงานได้รับความรู้และประสบการณ์ต่างๆ เกี่ยวกับการเขียนโปรแกรมภาษาซี (C Language) เพื่อการควบคุมไมโครคอนโทรลเลอร์ บัดนี้โครงงานดังกล่าวพร้อมทั้งรายงานได้สำเร็จลงแล้ว ทั้งนี้ด้วยความร่วมมือและสนับสนุนจากบุคคลต่างๆ ดังนี้

- ผู้ช่วยศาตราจารย์ ดร.รังสรรค์ ทองทา อาจารย์ที่ปรึกษาโครงงานผู้เปิดโอกาสให้ผู้จัดทำได้เรียนรู้การทำโครงงานนี้ และให้คำปรึกษา ข้อเสนอแนะที่เป็นประโยชน์ และได้ดูแลผู้ทำโครงการอย่างใกล้ชิด
- นายอำนวย ทีจันทึก ผู้ที่ให้คำปรึกษาในการทำโครงงาน
- บุคลากรสาขาวิชาวิศวกรรมโทรคมนาคม สนับสนุนการดำเนินงาน

ข้าพเจ้าใคร่ขอขอบพระคุณผู้ที่มีส่วนเกี่ยวข้องทุกท่านที่มีส่วนร่วมในการให้ข้อมูลและเป็นที่ปรึกษาในการทำรายงานฉบับนี้จนเสร็จสมบูรณ์ ตลอดจนให้การดูแลและให้ความเข้าใจเกี่ยวกับพื้นฐานการใช้งานโปรแกรม ซึ่งข้าพเจ้าขอขอบพระคุณเป็นอย่างสูงไว้ ณ ที่นี้ด้วย

ปัญญา หันตุลา
ณัฐพล นฤมลต์
สุพรรณธนา ใจรักษ์

สารบัญ

เรื่อง **หน้า**

เรื่อง **หน้า**

สารบัญภาพ

เรื่อง **หน้า**

เรื่อง **หน้า**

เรื่อง **หน้า**

สารบัญตาราง

เรื่อง **หน้า**

บทที่ 1
บทนำ

1.1 ความเป็นมา

ในปัจจุบันเทคโนโลยี GPS (Global Positioning System) ได้ถูกใช้งานอย่างแพร่หลาย และสามารถนำข้อมูลพิกัดจาก GPS มาใช้ประโยชน์ได้อย่างมากมายขึ้นอยู่กับวัตถุประสงค์ของผู้ใช้งาน ทั้งนี้ผู้จัดทำได้นำค่าพิกัดตำแหน่งจาก GPS มารายงานผลในรูปแบบของข้อมูลเสียง ซึ่งสามารถที่จะเปลี่ยนแปลงข้อมูลเสียงได้ตามที่ผู้ใช้งานต้องการเพื่อประโยชน์ในใช้งานในรูปแบบต่าง ๆ อาทิ เช่น การใช้งานกับรถประจำทางในการบอกตำแหน่งของจุดต่าง ๆ ในขณะเดินทาง และนอกจากนี้ยังสามารถจะประยุกต์ใช้งานในรูปแบบต่าง ๆ ตามที่ผู้ใช้ต้องการ

1.2 วัตถุประสงค์

1.2.1. เพื่อศึกษาการทำงานของระบบ GPS

1.2.2. เพื่อศึกษาการใช้งานและการประยุกต์ใช้งานชุดวงจร GPS และ MP3 Module

1.2.3. เพื่อศึกษาโปรแกรมควบคุมและการประยุกต์ใช้งานไมโครคอนโทรลเลอร์ (Microcontroller)

1.2.4. เพื่อศึกษาวิธีการสื่อสารข้อมูลและการประมวลผลข้อมูล

1.2.5. เพื่อนำความรู้ที่ได้จากการศึกษาภาคทฤษฎีของวิชาต่างๆที่ได้ศึกษามาปฏิบัติและประยุกต์ใช้ เพื่อสร้างชิ้นงานขึ้นมาและสามารถนำไปใช้งานได้จริง

1.3 ขอบเขตการทำงาน

1.3.1. อุปกรณ์สามารถค้นหาพิกัดตำแหน่งและส่งพิกัดเพื่อให้ไมโครคอนโทรลเลอร์ (Microcontroller) ประมวลผลได้

1.3.2. สามารถบันทึกข้อมูลเสียงเพื่อใช้ในการรายงานผลได้

1.3.3. สามารถรายงานข้อมูลเสียงเมื่อถึงพิกัดตำแหน่งที่กำหนดได้

1.4 การเลือกใช้ซอฟต์แวร์

ซอฟต์แวร์ที่ใช้ในโครงงานนี้ ผู้จัดทำได้เลือกใช้ภาษาซีในการควบคุมสั่งงานไมโครคอนโทรลเลอร์ เนื่องจากภาษาซีเป็นภาษาโครงสร้าง ง่ายต่อการทำความเข้าใจ ปรับปรุงพัฒนาต่อ นอกจากนั้นภาษาซียังเป็นภาษามาตรฐานไม่ขึ้นกับฮาร์ดแวร์ (ไมโครคอนโทรลเลอร์) มีความยืดหยุ่นในการโยกย้ายไปใช้งานกับไมโครคอนโทรลเลอร์ตระกูลอื่นได้ง่าย

1.5 ขั้นตอนการดำเนินการ

1.5.1 ปรึกษาอาจารย์ที่ปรึกษาโครงงานเกี่ยวกับขอบเขตของโครงงานที่จะทำ

1.5.2 ศึกษาข้อมูลเกี่ยวกับอุปกรณ์แต่ละตัวที่ต้องใช้โครงงาน ได้แก่ GPS Module , Void Record Module และ Microcontroller

1.5.3 สั่งซื้ออุปกรณ์ที่เกี่ยวข้องในการสร้าง

1.5.4 ฝึกการใช้โปรแกรมไมโครคอนโทรลเลอร์

1.5.5 ประกอบวงจรอิเล็กทรอนิกส์

1.5.6 เขียนโปรแกรมไมโครคอนโทรลเลอร์เพื่อสั่งการ Microcontroller ให้ทำงานตามวัตถุประสงค์

1.5.7 ทดลองใช้งานและแก้ไขสิ่งที่ผิดพลาด

1.5.8 จัดทำรูปเล่มรายงานของโครงงานเพื่อเสนออาจารย์ประจำสาขาวิชา

1.6 ประโยชน์ที่คาดว่าจะได้รับ

1.6.1. ได้เรียนรู้หลักการทำงานของอุปกรณ์ตรวจจับตำแหน่งระบบ GPS

1.6.2. ได้เรียนรู้การเขียนโปรแกรมควบคุมและการประยุกต์ใช้งานไมโครคอลโทรลเลอร์

1.6.3. ได้เรียนรู้การสร้างอุปกรณ์อิเล็กทรอนิกส์ให้สามารถทำงานตามที่ต้องการได้

1.6.4. สามารถนำความรู้ที่ได้ทางทฤษฎีมาประยุกต์ใช้ในทางปฏิบัติ

1.6.5. ได้เรียนรู้วิธีหาความรู้ด้วยตัวเองเพื่อนำมาปฏิบัติและประยุกต์ใช้งานจริง

บทที่ 2

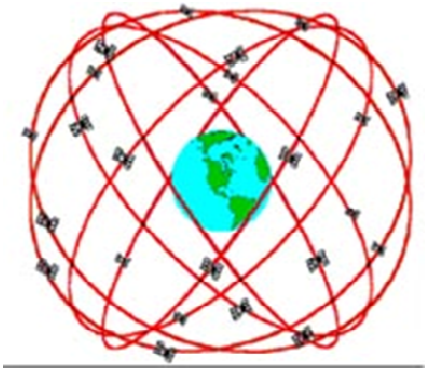
ทฤษฎีที่เกี่ยวข้อง

2.1 ระบบหาพิกัดบนพื้นโลก (Global Positioning System)

2.1.1 ความหมายของ GPS

ระบบระบุตำแหน่งบนพื้นโลก ย่อมาจากคำว่า Global Positioning System ซึ่งระบบ GPS ประกอบไปด้วย 3 ส่วนหลัก คือ

1. ส่วนอวกาศ ประกอบด้วยเครือข่ายดาวเทียม 3 ค่าย คือ
   - อเมริกา รัสเซีย ยุโรป ของอเมริกา ชื่อ NAVSTAR (Navigation Satellite Timing and Ranging GPS) มีดาวเทียม 28 ดวง ใช้งานจริง 24 ดวง อีก 4 ดวงเป็นตัวสำรอง บริหารงานโดย Department of Defenses มีรัศมีวงโคจรจากพื้นโลก 20,162.81 กม.หรือ 12,600 ไมล์ ดาวเทียมแต่ละดวงใช้เวลาในการโคจรรอบโลก 12 ชั่วโมง
   - ยุโรป ชื่อ Galileo มี 27 ดวง บริหารงานโดย ESA หรือ European Satellite Agency จะพร้อมใช้งานในปี 2008
   - รัสเซีย ชื่อ GLONASS หรือ Global Navigation Satellite บริหารโดย Russia VKS (Russia Military Space Force)



รูปที่ 2.1 วงโคจรของดาวเทียม

รูปที่ 2.2 ส่วนประกอบของระบบดาวเทียม GPS

ในขณะนี้ภาคประชาชนทั่วโลกสามารถใช้ข้อมูลจากดาวเทียมของทางอเมริกา (NAVSTAR) ได้ฟรี เนื่องจากนโยบายสิทธิการเข้าถึงข้อมูลและข่าวสารสำหรับประชาชนของรัฐบาลสหรัฐ จึงเปิดให้ประชาชนทั่วไปสามารถใช้ข้อมูลดังกล่าวในระดับความแม่นยำที่ไม่เป็นภัยต่อความมั่นคงของรัฐ กล่าวคือมีความแม่นยำในระดับบวก / ลบ 10 เมตร

2. ส่วนควบคุม ประกอบด้วยสถานีภาคพื้นดิน สถานีใหญ่อยู่ที่ Falcon Air Force Base ประเทศ อเมริกา และศูนย์ควบคุมย่อยอีก 5 จุด กระจายไปยังภูมิภาคต่าง ๆ ทั่วโลก

3. ส่วนผู้ใช้งาน ผู้ใช้งานต้องมีเครื่องรับสัญญาณที่สามารถรับสัญญาณดาวเทียมและแปรรหัสจากดาวเทียมเพื่อนำมาประมวลผลให้เหมาะสมกับการใช้งานในรูปแบบต่าง ๆ

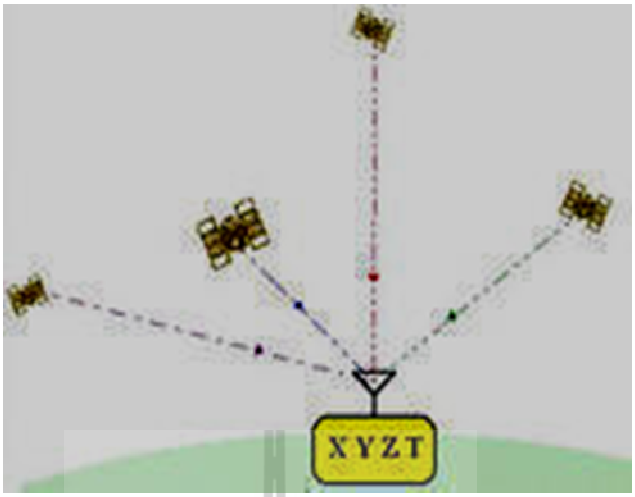
ระบบดาวเทียมทำงานโดย ดาวเทียม GPS (Navstar) ประกอบด้วยดาวเทียม 24 ดวง โดยแบ่งเป็น 6 รอบวงโคจร การโคจรจะเอียงทำมุมเอียง 55 องศากับเส้นศูนย์สูตร (Equator)ในลักษณะสานกันคล้าย ลูกตะกร้อแต่ละวงโคจรมีดาวเทียม 4 ดวง รัศมีวงโคจรจากพื้นโลก 20,162.81 กม. หรือ 12,600 ไมล์ ดาวเทียมแต่ละดวงใช้ เวลาในการโคจรรอบโลก 12 ชั่วโมง

2.1.2 หลักการทำงานของ GPS

GPS ทำงานโดยการรับสัญญาณจากดาวเทียมแต่ละดวง โดยสัญญาณดาวเทียมนี้ประกอบไปด้วยข้อมูลที่ระบุตำแหน่งและเวลาขณะส่งสัญญาณ ตัวเครื่องรับสัญญาณ GPS จะต้องประมวลผลความแตกต่างของเวลาในการรับสัญญาณเทียบกับเวลาจริง ณ ปัจจุบันเพื่อแปรเป็นระยะทางระหว่างเครื่องรับสัญญาณกับดาวเทียมแต่ละดวง ซึ่งได้ระบุมีตำแหน่งของมันมากับสัญญาณดังกล่าวข้างต้น เพื่อให้เกิดความแม่นยำในการค้นหาตำแหน่งด้วยดาวเทียม ต้องมีดาวเทียมอย่างน้อย 4 ดวง เพื่อบอกตำแหน่งบนผิวโลก ซึ่งระยะห่างจากดาวเทียมทั้ง 3 กับเครื่อง GPS (ที่จุดสีแดง) จะสามารถระบุตำแหน่งบนผิวโลกได้หากพื้นโลกอยู่ในแนวระนาบแต่ในความเป็นจริงพื้นโลกมีความโค้งเนื่องจากสัณฐานของโลกมีลักษณะกลมดังนั้นดาวเทียมดวงที่ 4 จะทำให้สามารถคำนวณเรื่องความสูงเพื่อทำให้ได้ตำแหน่งที่ถูกต้องมากขึ้น

การวัดระยะห่างระหว่างดาวเทียมกับเครื่องรับทำได้โดยใช้สูตรคำนวณ ระยะทาง(S) จะมีค่าเท่ากับ ความเร็ว (V) คูณกับ ระยะเวลา (T) วัดระยะเวลาที่คลื่นวิทยุส่งจากดาวเทียมมายังเครื่องรับ GPS คูณด้วยความเร็วของคลื่นวิทยุจะเท่ากับระยะทางที่เครื่องรับ อยู่ห่างจากดาวเทียม โดยเวลาที่วัดได้มาจากนาฬิกาของดาวเทียมที่มีความแม่นยำสูงมีความละเอียดถึงนาโนวินาที และมีการสอบทวนเสมอๆกับสถานีภาคพื้นดิน

องค์ประกอบสุดท้ายก็คือตำแหน่งของดาวเทียมแต่ละดวงในขณะที่ส่งสัญญาณมาว่าอยู่ที่ใด(Almanac) มายังเครื่องรับ GPS โดยวงโคจรของดาวเทียมได้ถูกกำหนดไว้ล่วงหน้าแล้วเมื่อถูกส่งขึ้นสู่อวกาศ สถานีควบคุมจะคอยตรวจสอบการโคจรของดาวเทียมอยู่ตลอดเวลาเพื่อทวนสอบความถูกต้อง



รูปที่ 2.3 การส่งสัญญาณจากดาวเทียม

ความแม่นยำของการระบุตำแหน่งนั้นขึ้นอยู่กับตำแหน่งของดาวเทียมแต่ละดวง กล่าวคือถ้าระยะห่างระหว่างดาวเทียมที่ใช้งานอยู่ห่างกันย่อมให้ค่าที่แม่นยำกว่าที่อยู่ใกล้กัน และยิ่งมีจำนวนดาวเทียมที่รับสัญญาณได้มากก็ยิ่งให้ความแม่นยำมากขึ้น ความแปรปรวนของชั้นบรรยากาศชั้นบรรยากาศประกอบด้วย ประจุไฟฟ้า ความชื้น อุณหภูมิ และความหนาแน่นที่แปรปรวนตลอดเวลา คลื่นเมื่อตกกระทบ กับวัตถุต่างๆ จะเกิดการหักเหทำให้สัญญาณที่ได้อ่อนลง และสิ่งแวดล้อมในบริเวณรับสัญญาณเช่นมีการบดบังจากกระจก ละอองน้ำ ใบไม้ จะมีผลต่อค่าความถูกต้องของความแม่นยำ เนื่องจากถ้าสัญญาณจากดาวเทียมมีการหักเหก็จะทำให้ค่าที่คำนวณได้จากเครื่องรับสัญญาณเพี้ยนไป และสุดท้ายก็คือประสิทธิภาพของเครื่องรับสัญญาณว่ามีความไวในการรับสัญญาณแค่ไหนและความเร็วในการประมวลผลด้วย

2.1.3 ปัจจัยที่ทำให้เกิดความคลาดเคลื่อนของพิกัด

เนื่องจากระบบจีพีเอสใช้เวลาเป็นส่วนสำคัญในการคำนวณหาพิกัดตำแหน่ง และคลื่นวิทยุเดินทางด้วยความเร็ว 3x10^8 เมตร/วินาที หรือ 300 เมตร ต่อ 1 ไมโครวินาที แต่ดาวเทียมอยู่จากพื้นโลกเป็นระยะทาง 11,000 ไมล์ หรือคลื่นวิทยุใช้เวลาเดินทางจากดาวเทียมถึงพื้นโลกประมาณ 0.06 วินาที หรือ ประมาณ 60 มิลลิวินาที ดังนั้นปัจจัยที่ทำให้เกิดความคลาดเคลื่อนมีดังนี้

1. การหักเหของคลื่นในชั้นบรรยากาศ (ทำให้มีผลต่อเวลาในการเดินทางของคลื่นวิทยุ)
2. การสะท้อนของคลื่นเมื่อเดินทางมาถึงภาคพื้นดินทำให้เกิด multipath เช่นการสะท้อนกับภูเขา,ตึก , อาคาร ปัจจัยนี้แก้ไขได้ยากเนื่องจากเป็นเรื่องของพื้นที่ของการใช้งาน
3. ความคลาดเคลื่อนหรือเสถียรภาพของฐานเวลาในภาครับ ภาครับราคาถูกต้นทุนต่ำอาจมีเสถียรภาพของฐานเวลาต่ำ
4. มุมและความห่างของดาวเทียมที่นำมาคำนวณหาพิกัด เนื่องจากมีดาวเทียมให้ภาครับทำการประมวลผล จำนวนน้อย ซึ่งอาจได้ดาวเทียมดวงที่ทำมุมแคบ สาเหตุที่ทำให้เหลือดาวเทียมน้อยดวงในการประมวลผล คือ ปัจจัยตั้งแต่ข้อ 1-3

2.1.4 GPS Module

เป็นลักษณะของชุดวงจร GPS ที่มีเฉพาะในตัวโมดูลเท่านั้น ซึ่งเป็นส่วนที่ประมวลค่าของข้อมูลต่างๆที่รับมาจากดาวเทียม GPS ซึ่งในการที่จะนำข้อมูลที่ได้รับการประมวลผลมาใช้นั้นจำเป็นที่จะต้องติดตั้งส่วนประมวลผลเพิ่มเติมเข้าไป ซึ่งอาจเป็นคอมพิวเตอร์ PC หรือ Notebook ที่ค่าที่ได้จากชุดวงจรจะสามารถนำมาแสดงบนแผนที่ได้ซึ่งเป็นชุดวงจร GPS ที่นำมาใช้งานในโครงงานนี้

รูปที่ 2.4 ลักษณะชุดวงจร GPS

การเชื่อมต่อฮาร์ดแวร์ของชุดอุปกรณ์ GPS จะเป็นการเชื่อมต่อแบบอนุกรม โดยใช้ RS-232 เชื่อมต่อกับคอมพิวเตอร์ทั่วไป เราต้องการสายนำสัญญาณเพียง 2 เส้นคือ เส้นที่ส่งข้อมูลออกจาก GPS และ Ground มีเพียงบางกรณีเท่านั้นที่จะใช้สายเส้นที่ 3 ในการรับข้อมูลจากอุปกรณ์อื่นๆเข้าตัว GPS Module ความเร็วในการส่งข้อมูลจะมีการปรับได้ตามมาตรฐานโดยส่วนใหญ่ที่พบเห็นกันทั่วไปคือแบบ 0183 [4800 baud rate, 8 bits, no parity และ1 stop bit] ซึ่งจะทวนสัญญาณทุกๆ 1 วินาที เราสามารถใช้มาตรฐานอื่นๆก็ได้หากเราต้องการอัตราการส่งข้อมูลที่สูงหรือต่ำกว่านี้ เช่น แบบ 0180 และ 0182[1200 baud rate, 8 bits, no parity และ1 stop bit]

2.1.5 การอ่านค่าข้อมูลจาก GPS โมดูล

การอ่านค่าข้อมูลจากเครื่องรับสัญญาณ GPS (GPS Module Receiver) ผ่านพอร์ตอนุกรม (Serial Port) เราจะใช้มาตรฐานของ NMEA (The National Marine Electronics Association) เป็นมาตรฐานในการอ่านข้อมูล ซึ่ง NMEA เป็นมาตรฐานที่ยอมรับในการส่งขอมูล Marine

Electronics ไปยังเครื่องคอมพิวเตอร์หรืออุปกรณ์อื่นๆ โดยข้อมูลที่เครื่องรับสัญญาณ GPS ส่งมาจะประกอบด้วย PVT (Position, Velocity, Time) ซึ่งข้อมูลที่ส่งมาจะมีลักษณะเป็นไลน์ เรียกว่า Sentence มาตรฐานของแต่ละ Sentence จะขึ้นอยู่กับอุปกรณ์แต่ละรุ่นหรือแต่ละบริษัทจะมีลักษณะที่เป็นมาตรฐานของ NMEA และทุกๆประโยค NMEA จะต้องมีอักษรขึ้นต้น (Prefix) เป็นการกำหนดชนิดของประโยค NMEA สำหรับเครื่องรับ GPS จะมีอักษรขึ้นต้นด้วย GP อื่นๆ ข้อกำหนดของประโยค NMEA โดยทั่วไปมีดังนี้

- ในแต่ละประโยค NMEA จะต้องขึ้นต้นด้วยเครื่องหมาย $ ก่อน Prefix
- แต่ละประโยค NMEA จะต้องมีความยาวไม่เกิน 80 อักขระ
- รายการของข้อมูลจะถูกแยกด้วยเครื่องหมายคอมมา (,)
- ข้อมูลจะมีลักษณะเป็นรหัส ASCII
- ข้อมูลจะเปลี่ยนแปลงตามความเที่ยงตรงที่บรรจุอยู่ในข้อความ
- มีการ Checksum ที่ท้าย Sentence ซึ่งอาจจะเช็คหรือไม่เช็คโดยหน่วยการอ่านข้อมูล
- การ Checksum ประกอบด้วยเครื่องหมาย * และอีก 2 ตัวเลขฐาน16 (HEX) แสดงการ Exclusive OR ของอักขระทั้งหมด

รูปแบบต่าง ๆ ของ NMEA Sentence

$GPGGA

ข้อมูลที่เป็นประโยค

$GPGGA,123519,4807.038,N,01131.000,E,1,08,0.9,545.4,M,,*47

จะมีความหมายดังนี้

GGA	Global Positioning System Fix Data (เจาะจงข้อมูลที่สำคัญ)
123519	Fix taken at 12:35:19 UTC
4807.038,N	ละติจูด(Latitude) 48 องศาเหนือ 07.038 ลิปดา
01131.000,E	ลองจิจูด (Longitude) 11 องศาตะวันออก 31.000 ลิปดา
1 = กำหนดคุณภาพ	0 = ผิดพลาด
	1 = GPS fix (SPS)
	2 = DGPS fix
	3 = PPS fix
	4 = เวลาจริงของ Kinematics
	5 = ทศนิยม RTK
	6 = ประมาณการ (คำนวณการสิ้นสุด)
	7 = ควบคุม input
	8 = Simulation
08	จำนวนของดาวเทียมที่มีการติดตาม
0.9	ความเที่ยงตรงของตำแน่งในแนวดิ่ง
545.4,M	ความสูงเหนือระดับน้ำทะเล
46.9,M	ความสูงเหนือระดับน้ำทะเล ทรงกลมของโลกแบบ WGS584 (เมตร)
*47	ตรวจผลรวมของข้อมูล (checksum data), ขึ้นต้นด้วย * เสมอ

$GPGSA

ข้อมูลที่เป็นประโยค

$GPGSA,A,A,3,04,05,,09,12,,,24,,,,,2.5,1.3,2.1*39

จะมีความหมายดังนี้

GSA	ข้อมูลดาวเทียมทั้งหมด
A	เลือกโดยอัตโนมัติ 2D หรือ 3D fix (M = ควบคุมเอง)
3	3D fix – ค่าประกอบด้วย : 1 = no fix
	2 = 2 มิติ (2D fix)
	3 = 3 มิติ (3D fix)
04,05...	รหัส PRNS ของดาวเทียมถูกใช้เพื่อการกำหนด (fix) (ในอวกาศ ใช้ 12)
2.5	PDOP (ความเที่ยงตรง)
1.3	ความเที่ยงตรงในแนวราบ (HDOP)
2.1	ความเที่ยงตรงในแนวดิ่ง (VAOP)
*39	ตรวจผลรวมของข้อมูล (checksum data), ขึ้นต้นด้วย * เสมอ

$GPRMC

ข้อมูลที่เป็นประโยค

$GPRMC,123519,A,4807.038,N,01131.000,E,022.4,084.4,23394,003.1,W*6A

จะมีความหมายดังนี้

RMC	บอกข้อมูลที่เล็กที่สุดของ GPS
123519	กำหนดการกระทำที่เวลา 12:35:19 UTC
A	สถานะ A = ทำงาน หรือ V = เฉย
4807.038,N	ละติจูด(Latitude) 48 องศาเหนือ 07.038 ลิปดา
01131.000มE	ลองจิจูด (Longitude) 11 องศาตะวันออก 31.000 ลิปดา
22.4	ความเร็วบนพื้นโลก
84.4	มุมของการติดตามดาวเทียมในหน่วยองศา
22394	วันที่ 23 เดือน 3 (มีนาคม) ปี ค.ศ. 1990
003.1,W	การเปลี่ยนแปลงของสนามแม่เหล็ก
*6A	ตรวจผลรวมของข้อมูล (checksum data), ขึ้นต้นด้วย * เสมอ

$GPGSV

ข้อมูลที่เป็นประโยค

$GPGSV,2,1,08,01,40,083,46,02,17,308,41,12,07,344.39,14,22,228,45*75

จะมีความหมายดังนี้

GSV	ข้อมูลดาวเทียมซึ่งมีรายละเอียดมาก
2	จำนวนของประโยคสำหรับข้อมูลทั้งหมด
1	ประโยคที่ 1 ของ 2
08	จำนวนของดาวเทียมที่รับได้
01	จำนวนดาวเทียม PRN
40	มุมเงย (evaluation), องศา
083	มุมกวาด (azimuth), องศา
46	ค่า SRN – ยิ่งสูงยิ่งดี
	สำหรับ 4 ดาวเทียมขึ้นไปต่อ 1 ประโยค
*75	ตรวจผลรวมของข้อมูล (checksum data), ขึ้นต้นด้วย * เสมอ

$GPGLL

ข้อมูลที่เป็นประโยค $GPGLL,4916.45,N,12311.12,W,225444,A,*31

จะมีความหมายดังนี้

GLL	ตำแหน่งทางภูมิศาสตร์ , ละติจูดและลองจิจูด
4916.45,N	ละติจูด (Longitude) 49 องศาเหนือ 16.45 ลิปดา
12311.12,W	ลองจิจูด (Longitude) 123 องศาตะวันตก 11.12 ลิปดา
225444	กำหนดค่าที่เวลา 22:54:44 UTC
A	สถานะ A = ทำงาน หรือ V = เฉย
*31	ตรวจผลรวมของข้อมูล (checksum data), ขึ้นต้นด้วย * เสมอ

$GPVTG

ข้อมูลที่เป็นประโยค

$GPVTG,054.7,T,034.4,M,005.5,N,010.2,K

จะมีความหมายดังนี้

VTG	การติดตามวงโคจรดาวเทียมและความเร็วบนพื้นโลก
054.7,T	ผลการติดตามวงโคจรดาวเทียม
034.4,M	ผลการติดตามวงโคจรดาวเทียมแบบแม่เหล็ก
005.5,N	ความเร็วบนพื้นโลก หน่วยนอต (knots)
010.2,K	ความเร็วบนพื้นโลก หน่วยกิโลเมตรต่อชั่วโมง

2.1.6 ระบบพิกัดตำแหน่งของ GPS

การแสดงพิกัดบนเครื่อง GPS (Global Positioning System) ที่ใช้อยู่โดยทั่วไปในประเทศจะนิยมใช้แค่สองระบบเท่านั้น คือ พิกัดภูมิศาสตร์ และพิกัดกริด UTM (Universal Transverse Mercator) การอ่านค่าในระบบพิกัด UTM นั้นจะไม่มีความยุ่งยากเท่าไหร่ เพราะอ่านตัวเลขตามค่า East (ค่า X) และค่า North (ค่า Y) และหน่วยของ UTM เป็นเมตรอยู่แล้ว แต่การอ่านค่าระบบพิกัดภูมิศาสตร์นั้นค่อนข้างยุ่งยากเล็กน้อย เพราะเครื่อง GPS บางรุ่น บางยี่ห้อแสดงค่าพิกัดภูมิศาสตร์ในหน่วยแบบที่เรียกว่า องศา ลิปดา ฟิลิปดา (DMS : Degree Minute Second) หรือแสดงเป็นหน่วยในระบบพิกัดแบบค่าตัวเลขทศนิยม (DD : Decimal Degree) เพื่อนำไปใช้ในคอมพิวเตอร์ ฉะนั้นเมื่อเราต้องการใช้งานแบบใดแบบหนึ่ง จึงต้องมีการแปลง (Convert) ค่าหน่วย DMS เป็น DD หรือ DD เป็น DMS

1. ระบบพิกัดภูมิศาสตร์ หมายถึงระบบที่เรียกกันว่า องศา ลิปดา ฟิลิปดา เป็นหน่วยแบบ DMS (Degree Minute Second) เหมือนกับหน่วยของเวลา บอกเวลาเป็น ชั่วโมง นาที และวินาที โดยมีค่าต่าง ๆ คือ

- ค่าองศา (Degree) 1 องศา มี 60 ลิปดา
- ค่าลิปดา (Minute) 1 ลิปดา มี 60 ฟิลิปดา
- ฟิลิปดา (Second) 1 ฟิลิปดา มีค่าระยะทางประมาณ 30.48 ม. หรือ 100 ฟุตบริเวณเส้นศูนย์สูตร

ตัวอย่างเช่น อาคารวิชาการ ตั้งอยู่ที่ค่าพิกัดภูมิศาสตร์ ละติจูด 100 องศา 27 ลิปดา 15 ฟิลิปดา เหนือ, ลองจิจูด 7 องศา 2 ลิปดา 25 ฟิลิปดา ตะวันออก

2. ระบบพิกัดแบบค่าตัวเลขทศนิยม หมายถึงระบบที่มีค่าตัวเลขทศนิยม ที่เป็นเลขฐานสิบในหน่วยแบบ DD (Decimal Degree)

ตัวอย่างเช่น อาคารวิชาการ ตั้งอยู่ที่ค่าพิกัดภูมิศาสตร์ ละติจูด 14.878837 เหนือ, ลองจิจูด 102.018764 ตะวันออก

2.1.7 เทคนิควิธีการแปลงค่าพิกัด

แบ่งออกได้เป็น 2 กรณี ดังนี้

1. การแปลงค่าพิกัดในหน่วย DMS ให้เป็น DD เช่น อาคารวิชาการ ตั้งอยู่ที่ ละติจูด 14 องศา 52 ลิปดา 4 พิลิปดา เหนือ ลองจิจูด 102 องศา 1 ลิปดา 7 พิลิปดา ตะวันออก เราสามารถหาได้โดย

จากสมการ DD = Degree+(Minute*60+Second)/3600

➔ค่าละติจูด = 14+(52*60+4)/3600 = 0.8716 จะได้พิกัดคือ14.8677 N

➔ลองจิจูด = 102+(1*60+7)/3600 = 0.0469 จะได้พิกัดคือ 102.0186 E

หรือ จากสมการ DD = (Seconds/3600) + (Minutes/60)+ Degrees

➔ละติจูด = (4/3600)+(52/60)+14 = 14.8677 N

➔ลองจิจูด = (7/3600)+(1/60)+102 = 102.0186 E

2. การแปลงหน่วยในระบบพิกัดแบบ DD เป็นแบบ DMS สามารถทำได้โดย นำค่าตัวเลขพิกัดในรูปแบบ DD ตัวอย่างเช่น 102.0186

2.1 ตัวเลขก่อนหน้าจุดทศนิยม จะเป็นค่าของหน่วยองศา ในที่นี่คือ 102 องศา

2.2 ให้นำตัวเลขหลังทศนิยมคูณด้วย 60 เช่น .0186 x 60 = 1.116 จากค่าที่คำนวณได้ 1.116 ตัวเลขก่อนหน้าจุดทศนิยม จะเป็นค่าของหน่วยลิปดา ในที่นี้ คือ 1 ลิปดา

2.3 ให้นำตัวเลขหลังทศนิยมจากผลคูณ มาคูณด้วย 60 อีกครั้งเช่น 0.116 x 60 = 6.96 จากค่าที่คำนวณได้ 6.96 ตัวเลขก่อนหน้าจุดทศนิยม จะเป็นค่าของหน่วยฟิลิปดา ในที่นี้ปัดทศนิยมเป็น 7 ฟิลิปดา

2.4 เมื่อนำตัวเลขมาอ่านรวมกันจะได้ 102 องศา 1 ลิปดา 7 ฟิลิปดา

2.1.8 การคำนวณหาระยะระหว่างจุด 2 จุด

การคำนวณระยะทางระหว่างพิกัด 2 ตำแหน่งสามารถทำได้จากสูตรการคำนวณ

ระยะทาง = SQRT[(Latt2 – Latt1) ^{2} + (Long2 – Long1)^2]

โดยค่าพิกัดที่นำไปคำนวณจะต้องเป็นหน่วยองศา Radians ซึ่งสามารถแปลงได้โดยนำค่าพิกัดไปคูณด้วย π/180 และนอกจากนี้การคำนวณระยะทางนั้นต้องนำค่าความแตกต่างของมุมคูณกับรัศมีโดยประมาณของโลก สำหรับค่า 6367 คือ รัศมีโดยประมาณของโลกหน่วยเป็นกิโลเมตร ซึ่งเราสามารถเขียนเป็นสมการใหม่ได้ คือ

ระยะทาง = SQRT{ [(Latt2 – Latt1) * πR/180] ^{2} + [(Long2 – Long1) * πR/180] ^{2}]

หรือ ค่ามุม$\Delta\theta$ = SQRT[(Latt2 – Latt1)^2 + (Long2 – Long1)^2]

ระยะทาง = $\Delta\theta$ * πR/180 (กิโลเมตร)

โดย

Latt1 คือ ละติจูดพิกัดตำแหน่งปัจจุบัน

Latt2 คือ ละติจูดพิกัดตำแหน่งที่บันทึก

Long1 คือ ลองจิจูดพิกัดตำแหน่งปัจจุบัน

Long2 คือ ลองจิจูดพิกัดตำแหน่งบันทึก

R คือ รัศมีโดยประมาณของโลกหน่วยเป็นกิโลเมตร

2.2 ไมโครคอนโทรลเลอร์ ARM7

2.2.1 ไมโครคอนโทรลเลอร์ ARM7 Philips LCP2148

ความสามารถของไมโครคอนโทรลเลอร์ Philips LCP2148 มีดังนี้

-ไมโครคอนโทรลเลอร์ขนาด 16/32 บิต ARM7TDMI-S มี LQFP 64 ขา

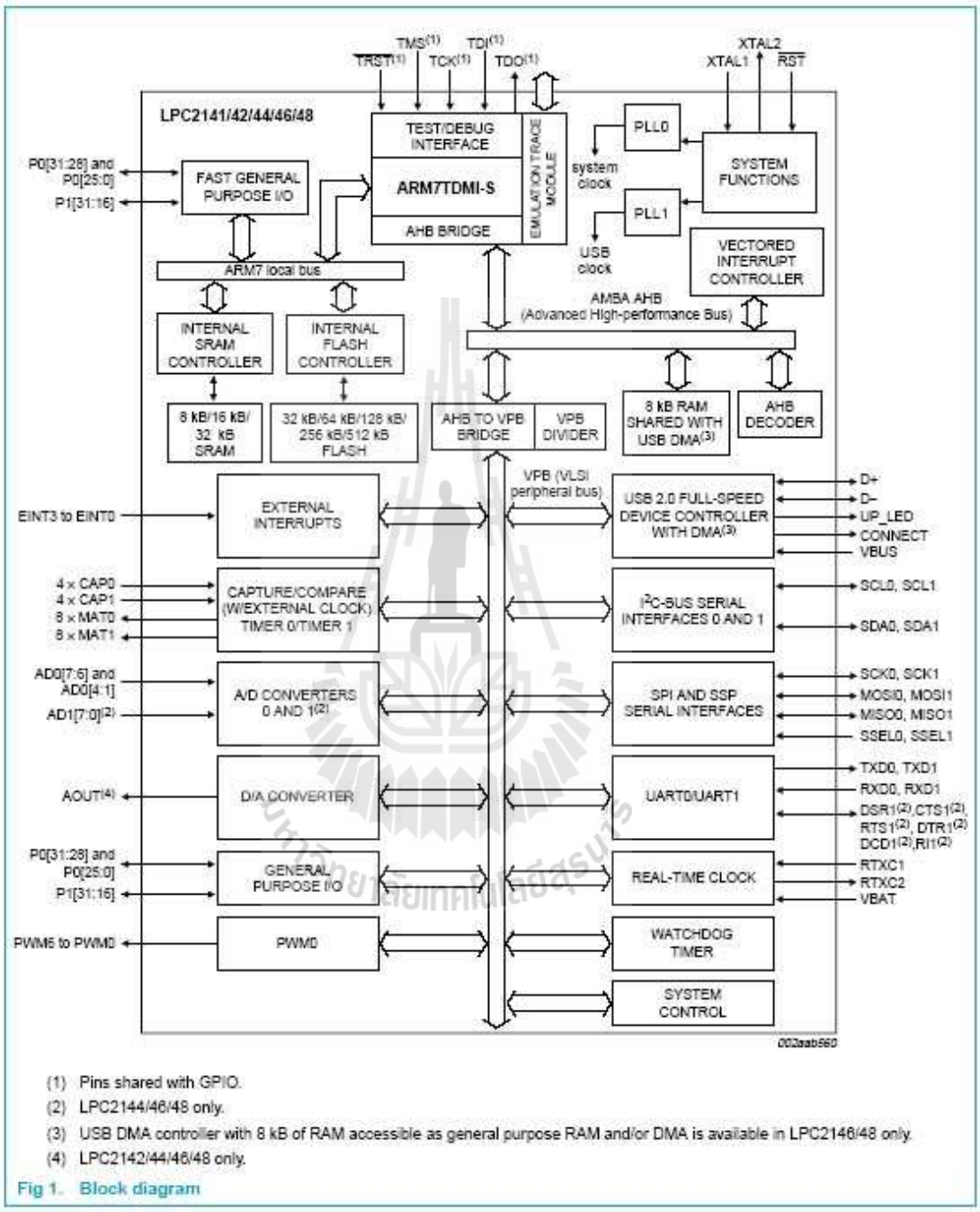
-หน่วยความจำ Static RAM มีขนาด 40kB

-หน่วยความจำ Flash Program Memory มีขนาด 512kB อยู่ภายในชิปที่สามารถ ลบ/เขียนซ้ำได้ถึง 10000 ครั้ง โปรแกรมชิปสามารถได้ทันทีผ่าน In-System Programming (ISP) และ In-Application Programming (IAP) โดยใช้ซอฟต์แวร์ boot-loader ที่อยู่ภายในชิปตัวควบคุมอุปกรณ์ USB 2.0 Full-speed โดยมี ARM สำหรับ endpoint ขนาด 8 kB สำหรับการติดต่อแบบ DMAวงจรแปลงแอนะลอกเป็นดิจิตอลความละเอียด 10 บิตจำนวน 2 ชุด ที่รับอินพุตได้ถึง 14 อินพุต โดยมีเวลาในการแปลงค่าต่ำถึง 2.44 μsวงจรแปลงดิจิตอลเป็นแอนะลอกความละเอียด 10 บิต 1 ตัววงจรไทเมอร์ขนาด 32 บิต 2 ชุด(มี 4 capture และ 4 compare channel)PWM (Pulse width modulation) 6 เอาต์พุทโมดูลนาฬิกาเวลาจริง (Real Time Clock) ที่สามารถต่อกับคริสตอลความถี่ 32 kHz และแบตเตอรี่ภายนอกได้ และวอชดอกซ์(Watchdog)

-วงจรสื่อสารอนุกรม UART(16C550) จำนวน 2 ชุดวงจรสื่อสารอนุกรม I^2C ความเร็วสูง (400Kbits/s)วงจรสื่อสารอนุกรม SPI และ SSP มีวงจร Phase lock loop ภายในเพื่อคูณค่าให้สัญญาณนาฬิกาภายในทำงานที่ความถี่สูงสุด 60 MHz Vectored Interrupt Controller ที่สามารถกำหนดลำดับความสำคัญ และกำหนดแอดเดรสของเวกเตอร์ได้ใช้กับแหล่งจ่ายไฟชุดเดียวขนาด 3.0 V และ 3.6V (3.3±10%)

-มี I/O pin อเนกประสงค์ที่สามารถใช้กับระดับแรงดัน 5 V ได้สูงสุด 45 ขาโดยสามารถจัดเป็นขาอินเตอร์รัปต์จากภายนอกได้สูงสุด 21 ขามีโหมดประหยัดพลังงาน 2 โหมดได้แก่ Idle และ Power-down

Block Diagram LPC2148



(1) Pins shared with GPIO.
(2) LPC2144/46/48 only.
(3) USB DMA controller with 8 kB of RAM accessible as general purpose RAM and/or DMA is available in LPC2146/48 only.
(4) LPC2142/44/46/48 only.

Fig 1. Block diagram

รูปที่ 2.5 Block Diagram LPC2148

จากรูป2.5 เป็นไมโครโปรเซสเซอร์ ARM7TDMI-S ซึ่งเป็นปัจจัยหลัก ด้านซ้ายมือเป็นส่วนของ AMR7 Local Bus ที่ใช้ในการติดต่อกับหน่วยความจำแบบ Flash ที่ใช้เก็บโปรแกรม และหน่วยความจำ SRAM ที่ใช้เก็บข้อมูล ส่วนที่ใช้ในการติดต่อกับหน่วยความจำภายนอกมีการติดต่อผ่านบัส AMBA AHB (Advanced High-performance Bus) ซึ่งใน LPC2148 ไม่สามารถต่อกับหน่วยความจำภายนอกได้

ในการติดต่อกับอุปกรณ์ที่เกี่ยวข้อง เช่น GPIO, I^2C, SPI, UART ฯลฯ จะติดต่อผ่านบัส VPB (VLSI peripheral BUS) ซึ่ง VPB บัสต่อกับ AHB บัส ผ่าน AHB to VPB Bridge โดยสามารถปรับลดค่าความถี่ของ VPB บัสให้ทำงานช้ากว่าความถี่ของซีพียูได้เพื่อให้ทำงานร่วมกับอุปกรณ์เสริมต่างๆ ที่มีความเร็วต่ำกว่าได้

การจัดหน่วยความจำของ LPC2148

เนื่องจาก ARM7 เป็นซีพียูขนาด 32 บิต ที่มีขาแอดเดรสต่อกับหน่วยความจำ จำนวน 32 ขา ทำให้สามารถอ้างถึงหน่วยความจำได้ 4 GB (2^{32} = 4GB) อุปกรณ์หลักของ ARM7TDMI จะมีสถาปัตยกรรมแบบ Von Neumann ที่ใช้บัสขนาด 32 บิต ชุดเดียวกันสำหรับตัวคำสั่งของโปรแกรม และข้อมูล โดยมีแค่คำสั่ง load, store, swap เท่านั้น ที่ใช้การเรียกข้อมูลที่เก็บในหน่วยความจำ การติดต่อกับคอมพิวเตอร์อินพุต หรือ เอาต์พุตก็จะใช้คำสั่งเดียวกันกับการใช้คำสั่งจัดการเกี่ยวกับหน่วยความจำในไมโครคอนโทรลเลอร์ LPC2148 ได้จัดสรรหน่วยความจำดังรูป 2.6 เมื่อซีพียูถูกรีเซตจะเริ่มต้นทำงานที่หน่วยความจำ 0x000 0000 จากหน่วยความจำทั้งหมด 4GB ได้แบ่งออกเป็น 4 ส่วน ดังนี้

- 1 GB แรก (แอดเดรส 0x0000 0000 – 0x3FFF FFFF) จัดเป็นส่วนของหน่วยความจำสำหรับเก็บโปรแกรม คือ หน่วยความจำ Flash memory ขนาด 512kB ซึ่งมีแอดเดรส 0x0000 0000 – 0x0007 FFFF
- หน่วยความจำในช่วง 1 GB - 2 GB (แอดเดรส 0x4000 0000 – 0x7FFF FFFF) จัดเป็นส่วนของหน่วยความจำ ARM จะเป็น SRAM ขนาด 40kB โดยแบ่งออกเป็น 2 ส่วน
    - ส่วนที่ 1 คือ 32kB แรกอยู่ที่แอดเดรส 0x4000 0000 – 0x4000 7FFFF
    - ส่วนที่ 2 คือ 8kB เป็นหน่วยความจำใช้สำหรับ USB โดยมีแอดเดรส 0x7FD0 0000- 0x7FD0 1FFF หน่วยความจำที่ใกล้ 2 GB จะเป็นส่วนของ Boot Block ขนาด 12 kB ซึ่งเป็นโปรแกรมที่ทำงานเพื่อเขียนโปรแกรมลงในหน่วยความจำ Flash

- หน่วยความจำ 2GB - 3 GB (แอดเดรส 0x8000 0000 – 0xDFFF FFFF) สงวนไว้สำหรับต่อกับหน่วยความจำภายนอก ซึ่งไม่ได้ใช้งาน
- หน่วยความจำ 2GB - 3GB จะเป็นพื้นที่สำหรับติดต่อกับอุปกรณ์อื่นๆ ที่อยู่ในชิป โดยแบ่งเป็น อุปกรณ์ที่ต่อกับ VPB บัส จะติดต่อกับหน่วยความจำช่วง 0xE000 00000 – 0xEFFF FFFF ถ้าเป็นอุปกรณ์ที่ติดต่อกับผ่านทาง AHB จะมีช่วงแอดเดรส 0xF000 0000 – 0xFFFF FFFF ไม่สามารถติดต่อกับหน่วยความจำภายนอก จึงไม่ได้ใช้หน่วยความจำส่วนนี้

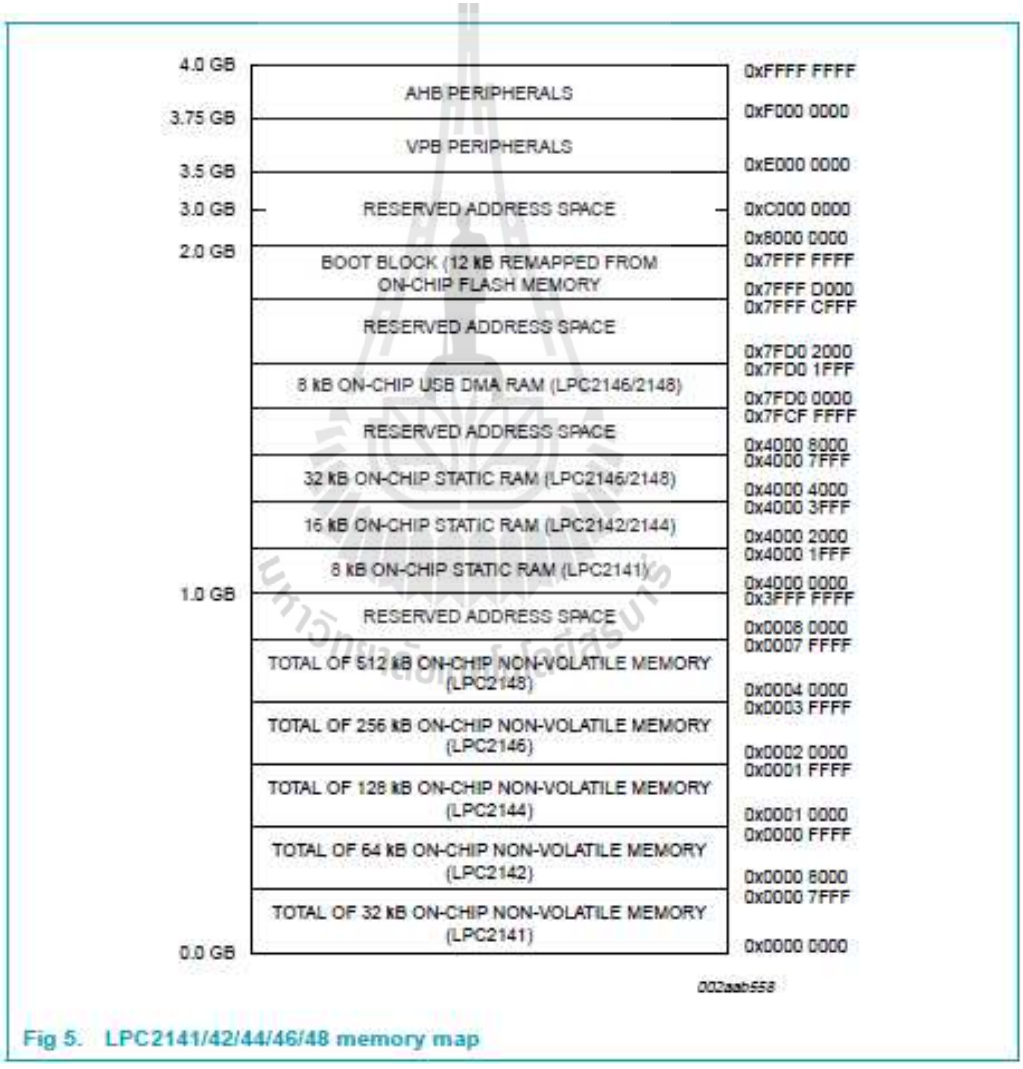


Fig 5. LPC2141/42/44/46/48 memory map

รูปที่ 2.6 memory map LPC 2148

การจัดขาของ LPC2148

มีจำนวนขาทั้งหมด 64 ขา โดยมีการจัดขาแสดงดังรูปที่ 2.7 ดังนี้

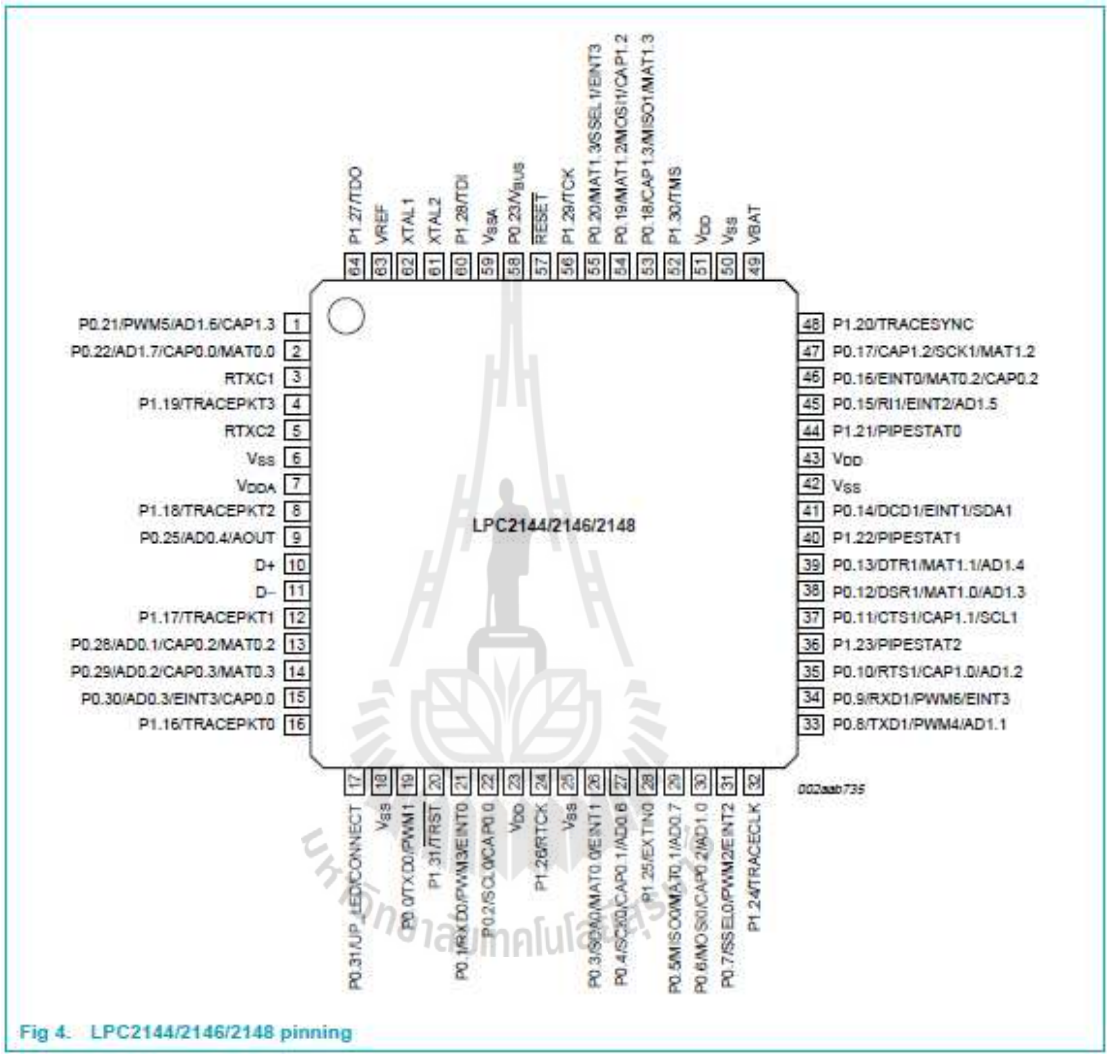


Fig 4. LPC2144/2146/2148 pinning

รูปที่ 2.7 LPC2148 pinning

หลังจากการรีเซตขาพอร์ตทั้งหมดจะถูกกำหนดให้ทำหน้าที่เป็นอินพุตขาแต่ละขา จะมีการทำงาน เช่นขาที่ 14 สามารถทำหน้าที่ได้ 4 หน้าที่ คือ

- เป็น P0.29/AD0.2/CAP0.3/MAT0.3
- ถ้าเป็นอินพุตเอาต์พุตจะเรียกว่า General Purpose Input Output : GPIO ก็คือขา P0.23
- ถ้าใช้วงจรแปลงแอนะลอกเป็นดิจิตอลขานี้คือ AD0.2 เป็นขาอินพุตสำหรับสัญญาณแอนะลอก ADC0 อินพุตสอง
- ถ้าใช้งาน Time0 ขานี้จะเป็น CAP0.3 คือ Capture input for timer 0, Channel3 หรือ MAT0.3 ซึ่งเป็น Match output for timer0, Channel3

2.2.2 ฮาร์ดแวร์ และซอฟต์แวร์ที่ใช้ในการทดลอง

ฮาร์ดแวร์ และซอฟต์แวร์ที่ใช้ในการเขียนไมโครคอนโทรลเลอร์ Philips LPC 2148 เป็นไมโครคอนโทรลเลอร์ตระกูล ARM7TDMI-S ฮาร์ดแวร์ที่ใช้ประกอบด้วย แผงวงจร CP-JR ARM7 USB-LPC2148 EXP บริษัท อีทีที จำกัด ส่วนของซอฟต์แวร์คอมไพเลอร์เพื่อแปลภาษาซีเป็นภาษาเครื่องของ LPC2148 จะใช้ชุดพัฒนา RealView Microcontroller Development Kit Version 3.02a โดยใช้คอมไพเลอร์ CARMของบริษัท Keill เป็นรุ่น Evaluation ที่อนุญาตให้ดาวน์โหลดมาใช้ทดสอบโปรแกรมฟรี โดยมีข้อจำกัดว่าโปรแกรมที่แปลเป็นภาษาเครื่องแล้ว จะมีขนาดไม่เกิน 16kB หลังจากคอมไพล์เป็นภาษาเครื่องจะทำการลงโปรแกรมชิปไมโครคอนโทรลเลอร์ด้วยโปรแกรม LPC2000 Flash Utility ของบริษัท Philips ซึ่งอนุญาตให้ใช้งานได้ฟรี

คุณสมบัติของบอร์ดไมโครคอนโทรลเลอร์ CP-JR ARM7 USB-LPC2148

1. ใช้ MCU ตระกูล ARM7TDMI-S เบอร์ LPC2148 ของ Philips ซึ่งเป็น MCU ขนาด 16/32-Bit

2. ใช้ Crystal 12.00MHz โดย MCU สามารถประมวลผลด้วยความเร็วสูงสุดที่ 60MHz เมื่อใช้งานร่วมกับ Phase-Locked Loop (PLL) ภายในตัว MCU เอง

3.รองรับการโปรแกรมแบบ In-System Programming (ISP) และ In-Application Programming(IAP) ผ่านทาง On-Chip Boot-Loader Software ทางพอร์ต UART0 (RS232)

4. Power Supply ใช้แรงดันไฟฟ้า +5VDC โดยใช้ขั้วต่อแบบ CPA-2PIN จากภายนอก หรือใช้พลังงานจาก USB Port ได้ (ในกรณีใช้กระแสไม่เกิน 500mA)

5. ภายใน MCU มีหน่วยความจำโปรแกรมแบบ Flash ขนาด 512KB, Static RAM ขนาด 40KB

6. มีวงจร USB มาตรฐาน 2.0 แบบ Full Speed ภายในตัว (USB Function มี 32 End Point)

7. จำนวน GPIO สูงสุดถึง 47 I/O Pins สามารถเชื่อมต่อกับระบบ I/O ที่เป็นสัญญาณ 5V ได้ ซึ่งขาสัญญาณ GPIO จะมีการใช้งานร่วมกันของ Function อื่นๆอีกดังนี้วงจรสื่อสารอนุกรมแบบ SPI จำนวน 2 ช่อง และ วงจรสื่อสารอนุกรมแบบ I2C จำนวน 2 ช่องวงจร ADC ขนาด 10 Bit จำนวน 14 ชุด และ วงจร DAC ขนาด 10 Bit จำนวน 1 ชุดวงจร UART แบบ Full-Duplex จำนวน 2 ช่อง คือ UART-0 มาตรฐาน 4 Pin ETT เป็นสัญญาณระดับ RS232 Level และ UART-1 เป็นสัญญาณระดับ TTL LevelTimer 32-bit จำนวน 2 ช่อง (4 Input Capture / 4 Output Compare), 6-Channel PWMOutput, Watchdog Timer และ Real Time Clock

8. มีวงจรเชื่อมต่อกับ Character LCD โดยใช้วงจรการเชื่อมต่อแบบ 4 บิต จาก GPIO1[25..31] พร้อมวงจรปรับความสว่างหน้าจอ

9. มีวงจรเชื่อมต่อกับ JTAG ARM ขนาด 20 Pin มาตรฐาน เพื่อทำการ Debug แบบ Real Time ได้

10. มีวงจรทดลองขั้นพื้นฐานสำหรับสนับสนุนการใช้งานและทดลองเรียนรู้ ขั้นพื้นฐานอย่างครบถ้วน จัดเตรียมไว้ภายในบอร์ด (ติดตั้งไว้เฉพาะรุ่น CP-JR ARM7 USB-LPC2148 EXP) ซึ่ง ได้แก่

-LED Output แบบ Sink Current สำหรับแสดงสถานะของ Output จำนวน 4 ชุด

-Push Button Switch แบบ Active Logic “0” สำหรับทดสอบ Input Logic จำนวน 4 ชุด

- Volume ปรับค่าแรงดัน 0-3.3V สำหรับทดสอบการทำงานของ ADC จำนวน 4 ชุด

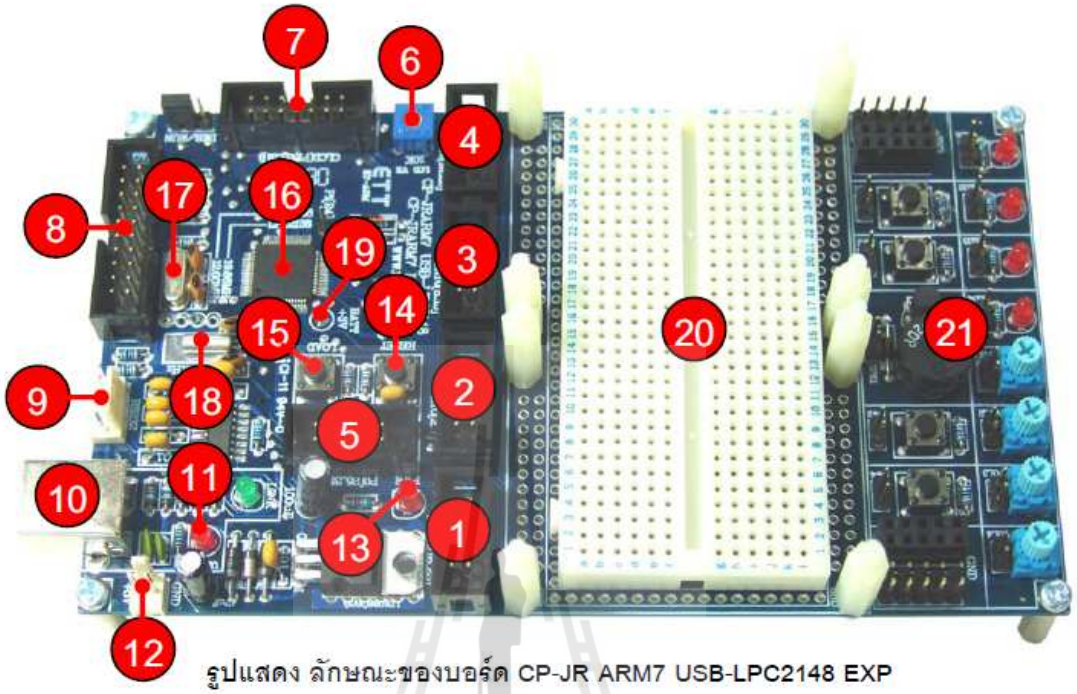
-ชุดกำเนิดสัญญาณเสียง Mini Speaker สำหรับทดสอบการเสียงแบบต่างๆ จำนวน 1 ชุด

-แผงต่อวงจร Project Board รุ่น AD-100 ขนาด 360 จุด สำหรับเป็นพื้นที่ต่อทดลองวงจรขนาดเล็กๆ เพื่อใช้งานร่วมกับ CPU ได้อย่างอิสระ

-จุดต่อแหล่งจ่ายไฟ +3.3V และ GND สำหรับเชื่อมต่อไปยังวงจรภายนอกอื่นๆ

11. ทนอุณหภูมิใช้งานระหว่าง -40 to +85°C

โครงสร้างบอร์ดไมโครคอนโทรลเลอร์ CP-JR ARM7 USB-LPC2148



รูปที่ 2.8 ลักษณะโครงสร้างของบอร์ด CP-JR ARM7 USB-LPC2148 / EXP

หมายเลข 1 คือ ขั้วต่อ Port1[16..23] จำนวน 8 บิต

หมายเลข 2 คือ ขั้วต่อ Port0[2..7] จำนวน 6 บิต

หมายเลข 3 คือ ขั้วต่อ Port0[8..15] จำนวน 8 บิต

หมายเลข 4 คือ ขั้วต่อ Port0[16..23] จำนวน 8 บิต

หมายเลข 5 คือ ขั้วต่อ Port0[25..31] จำนวน 7 บิต

หมายเลข 6 คือ ตัวต้านทานสำหรับปรับค่าความสว่าง (Contrast) ของหน้าจอ LCD

หมายเลข 7 คือ ขั้วต่อ Character LCD โดยใช้สัญญาณ Port1[25..31] ในการเชื่อมต่อ

หมายเลข 8 คือ ขั้วต่อ JTAG โดยใช้สัญญาณ Port1[26..31] และ Reset ของ CPU

หมายเลข 9 คือ ขั้วต่อ RS232 สำหรับใช้งาน และ Download Hex File ให้ CPU

หมายเลข 10 คือ ขั้วต่อ USB สำหรับเชื่อมต่อกับ USB Hub รุ่น 2.0

หมายเลข 11 คือ LED แสดงสถานะ Power จาก USB และ สถานของการเชื่อมต่อกับ USB

หมายเลข 12 คือ ขั้วต่อ Power ขนาด +5VDC และ GND เพื่อจ่ายให้กับบอร์ด

หมายเลข 13 คือ LED แสดงสถานะของแหล่งจ่ายไฟ Power ของบอร์ด

หมายเลข 14 คือ Switch RESET สำหรับสั่ง Reset การทำงานของ CPU

หมายเลข 15 คือ Switch LOAD ใช้ร่วมกับ Switch RESET เพื่อ Download Hex ให้ CPU

หมายเลข 16 คือ CPU เบอร์ LPC2148 ของ Philips ซึ่งเป็น CPU ประจำบอร์ด

หมายเลข 17 คือ Crystal 12.00 MHz สำหรับป้อนให้เป็นสัญญาณนาฬิกาของ LPC2148

หมายเลข 18 คือ Crystal 32.768 KHz สำหรับ Real Time Clock (RTC) ในตัวของ LPC2148

หมายเลข 19 คือ จุดเชื่อมต่อ ลังถ่าน Battery ขนาด +3V (อยู่ด้านใต้บอร์ด) สำหรับต่อให้กับRTC เพื่อเก็บรักษาค่าเวลาของ RTC ในขณะที่ไม่ได้จ่ายไฟเลี้ยงให้กับบอร์ด

หมายเลข 20 คือ แผง Project Board รุ่น AD-100 ขนาด 360 จุด สำหรับต่อวงจร (มีติดตั้งไว้เฉพาะในรุ่น CP-JR ARM7 USB-LPC2148 EXP)

หมายเลข 21 คือ ส่วนของวงจร I/O พื้นฐาน สำหรับใช้ทดสอบการทำงานของ Function ต่างๆของ CPU (มีติดตั้งไว้เฉพาะในรุ่น CP-JR ARM7 USB-LPC2148 EXP) โดยมีรายละเอียดดังนี้คือ

LED สำหรับแสดงผลการทำงานของ Output แบบ Sink Current มีทั้งหมด 4 ชุด

Push Button Switch สำหรับกำเนิด Logic เพื่อทดสอบการทำงาน Input มีทั้งหมด 4 ชุด

Volume สำหรับปรับค่าแรงดัน 0-3.3V เพื่อใช้ทดสอบการทำงานของ A/D มีทั้งหมด 4 ชุด

Mini Speaker สำหรับใช้กำเนิดเสียง เช่น Beep จำนวน 1 ชุด

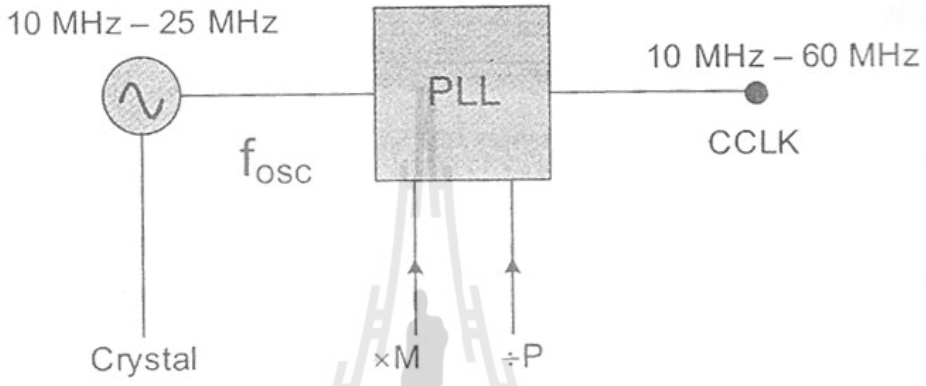
จุดต่อแหล่งจ่ายไฟ +3.3V และ GND

2.2.3 การกำหนดค่าเริ่มต้นให้กับไมโครคอนโทรลเลอร์ ARM7

การกำหนดค่าเริ่มต้นให้ก่อนโดยค่าที่ต้องการกำหนด คือ ส่วนของหน่วยความจำ RAM ที่ใช้เป็น stack pointer การกำหนดค่าของ stack pointer ต้องเขียนคำสั่งเป็นโปรแกรมภาษาแอสแซมบลี ภายในของไมโครคอนโทรลเลอร์ ARM มีวงจร Phase Lock Loop : PLL สำหรับคูณค่าความถี่ของสัญญาณนาฬิกาจากภายนอก เพื่อให้ตัวไมโครคอนโทรลเลอร์สามารถทำงานที่ความถี่สูงๆ ได้โดยการกำหนดค่าตัวคูณความถี่ สำหรับ LPC2148 ภายในจะมีวงจร PLL อยู่สองวงจรโดยตั้งชื่อเป็น PLL0 และ PLL1 โดย PLL0 ใช้คูณค่าความถี่ของสัญญาณนาฬิกาภายนอกให้กับซีพียู ARM 7 และ PLL1 ที่ทำหน้าที่คูณวามถี่ให้กับวงจรส่วนของ USB โดยต้องคูณค่าให้ได้ความถี่ 48 MHz หลังจากที่ไมโครคอนโทรลเลอร์ทำงานที่ความถี่สูงแล้วอุปกรณ์ประกอบ (Peripheral)ต่างๆ ที่อยู่ภายนอกไมโครคอนโทรลเลอร์ เช่น UART , I^2C ฯลฯ จะทำงานไม่ทัน ดังนั้นภายในไมโครคอนโทรลเลอร์ ARM7 จะมีวงจรหารความถี่เพื่อลดความถี่ที่ป้อนให้อุปกรณ์ต่างๆหน่วยความจำแบบ Flash ภายในของ LPC2148 มีการทำงานช้ากว่าตัวซีพียู ดังนั้นภายในของ LPC2148 จะมีวงจร Memory Accelerator Module (MAM) เพื่อทำหน้าที่เร่งความเร็วของการติดต่อกับหน่วยความจำแฟลช(Flash)ภายในชิป

วงจร Phase Lock Loop : PLL

วงจรคริสตัลออสซิลเลเตอร์ภายในของไมโครคอนโทรลเลอร์ตระกูล LPC2000 ใช้ได้กับคริสตัล ค่าความถี่ 1 MHz – 30MHz เอาต์พุตของวงจรเรียกว่า f_{osc} ส่วนความถี่สัญญาณนาฬิกาของไมโครคอนโทรลเลอร์จะมีชื่อเรียกว่า CCLK โดยปกติถ้าไม่ได้เปิดใช้วงจร PLL ไมโครคอนโทรลเลอร์ ARM7 จะนำความถี่ f_{osc} ไปใช้กับไมโครคอนโทรลเลอร์ ARM7 เช่นเดียวกัน (กรณีนี้ค่าของ f_{osc} และ CCLK จะมีค่าเท่ากัน)



รูปที่ 2.9 ผังวงจร PLL ภายในไมโครคอนโทรลเลอร์ ARM 7

วงจร PLL จะรับค่าความถี่สัญญาณนาฬิกาจากคริสตัลออสซิเลเตอร์ที่ถูกควบคุมค่าโดยคริสตัลภายนอก แล้วนำมาคูณด้วยค่าคงที่ M ให้เป็นความถี่ 10 MHz – 60 MHz โดยใช้วงจร Current Controlled Oscillator (CCO) ทำหน้าที่คูณความถี่ ค่าตัวคูณ M จะมีค่าตั้งแต่ 1 ถึง 32 วงจร CCO ทำงานในช่วงความถี่ 156 MHz - 320MHz ดังนั้นภายในลูปของ CCO จะต้องมีวงจรหารค่าอีกหนึ่งตัวเพื่อให้เอาต์พุตของ PLL สร้างความถี่ให้ได้ค่าตามที่ต้องการ ค่าตัวหาร คือ P กำหนดค่าได้เป็น 2,4,8 หรือ 16 เมื่อไมโครคอนโทรลเลอร์ถูกรีเซต วงจร PLL จะถูกปิดการทำงาน ซึ่งต้องให้ซอฟต์แวร์สั่งเปิดการทำงานของโปรแกรม ตัวโปรแกรมจะต้องกำหนดค่าตัวคูณ M ตัวหาร P และกระตุ้นการทำงานของ PLL และรอให้ PLL ล็อกความถี่ได้ก่อน จึงจะสั่งต่อให้ PLL เป็นสัญญาณนาฬิกาของไมโครคอนโทรลเลอร์ ARM7 ค่าเวลาในการเซต PLL เป็น 10μs เอาต์พุตของ PLL คำนวณได้ดังนี้

$CCLK = M * f_{osc}$ หรือ $CCLK = fcco/(2 * P)$

ค่าความถี่ CCO คำนวณได้ดังนี้

$fcco = CCLK * 2 * P$ หรือ $fcco = f_{osc} * 2 * M * P$

ค่า P เมื่อนำไปคูณแล้ว fcco จะมีเงื่อนไขดังนี้

$156MHz < fcco < 320MHz$

แผงวงจร CP-JR ARM7 USB-LPC2148 EXPใช้คริสตัลความถี่ 12.00MHz ดังนั้น f_{osc} = 12MHz ต้องการคูณให้ได้ความถี่สูงสุดไม่เกิน 60MHz จะได้ตัวคูณ M = 5 ดังนั้น CCLK = 5*12.00 = 60.00MHz

ในการกำหนดค่า PLL1 เพื่อกำเนิดสัญญาณนาฬิกาสำหรับวงจร USB จะต้องคำนวณ ค่า M ที่นำไปคูณกับความถี่ของสัญญาณนาฬิกาให้ได้ความถี่ 48 MHz จาก แผงวงจร CP-JR ARM7 USB-LPC2148 EXP จะคำนวณได้ M = 48MHz / f_{osc} = 48MHz / 12MHz =4

การหาค่าของ P ให้ใช้ตามค่าที่ได้จากกรณีของ PLL0 คือให้ค่า P = 2 ได้ fcco =48MHz * 2 *2 = 192MHz ซึ่งตรงตามเงื่อนไขในการกำหนดค่าการทำงานของวงจร PLL จะสั่งผ่านทางรีจิสเตอร์ ที่เกี่ยวข้องกับ PLL ดังแสดงในตารางที่ 2.2.1 (ภาคผนวก)

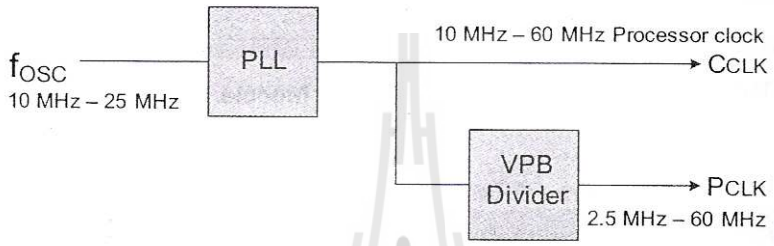
การกำหนดค่าของ P , M กำหนดที่รีจิสเตอร์ PLLCFG ซึ่งแสดงรายละเอียดแต่ละบิตของรีจิสเตอร์ PLLCFG ได้ดังตาราง 2.2.2(ภาคผนวก)

เมื่อกำหนดค่าตัวหาร P และตัวคูณ M แล้วคำสั่งให้วงจร PLL ทำงานและต่อ PLL เป็นสัญญาณนาฬิกาหลักได้ที่รีจิสเตอร์ PLLCON ซึ่งมีค่าดังตารางที่ 2.2.3(ภาคผนวก) ค่าแต่ละบิตของรีจิสเตอร์ PLLSTAT เพื่อดูสถานะของ PLL แสดงได้ในตารางที่ 2.2.4(ภาคผนวก)

เราสามารถกำหนดโหมดการทำงานของวงจร PLL ได้จากบิต PLLC และ PLLE ของรีจิสเตอร์ PLLCON ได้ดังตารางที่ 2.2.5(ภาคผนวก)

การกำหนดค่าความถี่ให้กับ VLSI Peripheral Bus

เมื่อกำหนดให้กับวงจร PLL ภายในไมโครคอนโทรลเลอร์ ARM7 ทำงานสร้างความถี่สูงและต่อให้เป็นสัญญาณนาฬิกาของโปรเซสเซอร์แล้ว อุปกรณ์ประกอบต่างๆ เช่น UART, GPIO, I^2C ที่ต่อกับบัส VPB จะทำงานไม่ทัน ดังนั้นภายในไมโครคอนโทรลเลอร์ ARM7 จะมีวงจรหารลดความถี่ของ CCLK ให้เป็นความถี่ของ VPB Bus ที่เรียกว่า PCLK โดยบล็อกไดอะแกรมภายในแสดงดังรูป 2.30



รูปที่ 2.10 Block Diagram วงจรกำเนิดสัญญาณภายใน ARM7

รีจิสเตอร์ที่เกี่ยวข้องคือ VPBDIV ที่อยู่แอดเดรส 0xE10FC100 โดยสามารถอ่านและเขียนได้ ค่าของรีจิสเตอร์ VPBDIV จะมีค่า 2 บิตสุดท้าย โดยกำหนดได้ดังตารางที่ 2.2.7(ภาคผนวก)

การกำหนดค่าให้กับ Memory Accelerator Module (MAM)

การกำหนดค่าการทำงาน MAM มีรีจิสเตอร์ที่เกี่ยวข้องสองตัว คือ MAMCR, MAMTIM ดังแสดงในตารางที่ 2.2.8 (ภาคผนวก)

การกำหนดค่าโหมดการทำงานของ MAM ผ่านทาง MAMCR ได้ดังตาราง 2.2.9 (ภาคผนวก)

การกำหนดค่ารีจิสเตอร์ MAMTIM กำหนดได้ดังตารางที่ 2.2.10(ภาคผนวก)

ในการกำหนดค่าเริ่มต้นการทำงานของ MAM เริ่มต้นด้วยการหยุดทำงานของ MAM ด้วยการโหลดค่า 0 ให้กับ MAMCR แล้วจึงกำหนดค่าของ MAMTIM ตามที่ต้องการแล้วจึงเปิดการทำงานของ MAM ด้วยการเขียนค่า 1 หรือ 2 ให้กับ MAMCR

ถ้าซีพียูมีความถี่ CCLK น้อยกว่า 20MHz ให้ใช้ MAMTIM = 1*CCLK ถ้า CCLKมีค่าระหว่าง 20 –40 MHz ให้กำหนด MAMTIM = 2*CCLK ถ้าความถี่มากกว่า 40MHz ให้กำหนดค่า MAMTIM = 3*CCLK

2.2.4. การต่อ GPIO เป็นเอาต์พุต

ภายในไมโครคอนโทรลเลอร์ มีพอร์ตอเนกประสงค์ ขนาด 32 บิต ให้ใช้งาน 2 พอร์ต คือ Port 0 , Port1 โดยให้ Port0 มีขาต่อใช้งานได้ 29 ขา คือ P0.0 –P0.31 โดยไม่มีขา P0.24, P0.26, P0.27 ให้ใช้งาน และขา P0.31 ทำหน้าที่เป็นเอาต์พุตได้อย่างเดียว ส่วน Port1 จะมีต่อให้ใช้งานแค่ 16 ขา คือ P1.16-P1.31 โดยแต่ละขาของพอร์ต P0 และ P1 มีหน้าที่การทำงานได้หลายประเภท ซึ่งต้องกำหนดรีจิสเตอร์ PINSEL

การกำหนดหน้าที่การทำงานของพอร์ต

หลังจากเกิดการรีเซตไมโครคอนโทรลเลอร์ ARM7 วงจรภายในสั่งรีเซตค่ารีจิสเตอร์ PINSEL ทุกตัวกำหนดค่าให้ Port0 ,Port1 ทุกขาเป็น GPIO และให้ทุกขาเป็นอินพุต

ขาแต่ละขาของ Port0,Port1 มีหน้าที่การทำงานได้หลายหน้าที่ โดยกำหนดการทำงานของ Port ที่รีจิสเตอร์ PINSEL0, PINSEL1 และกำหนดหน้าที่การทำงานของ Port1 ที่รีจิสเตอร์ PINSEL2 โดยรีจิสเตอร์แต่ละตัวที่มีแอดเดรสแสดงในตารางที่ 2.2.11(ภาคผนวก)

ตัวอย่างการกำหนดค่าขาให้ ขา P0.16-P0.31 มีการทำงานเป็น GPIO ทำโดยการเขียนค่า 0 ทุกบิตให้กับ PINSEL 1 ดังนี้

```
PINSEL1 = 0x00000000;  //set P0.16-P0.31  to  GPIO  Function
```

ค่าแต่ละบิตของริจิสเตอร์ PINSEL2 ที่ใช้กำหนดหน้าที่การทำงาน Port P1 แสดงได้ในตารางที่ 2.2.14(ภาคผนวก) ตัวอย่างการกำหนดค่าให้ขา P1.16-P1.31 มีการทำงานเป็น GPIO ทำการเขียนค่า 0 ทุกบิตให้กับ PINSEL1

```
PINSEL2 = 0x000000000;   //set P1.16-P1.31  to GPIO Function
```

การกำหนดค่าควบคุมการทำงานพอร์ตอเนกประสงค์ (GPIO)

หลังจากที่กำหนดค่าให้พอร์ตแต่ละตัวมีการทำงานเป็น GPIO แล้ว ควบคุมการทำงานของ GPIO จะสั่งงานผ่านทางรีจีสเตอร์ 4 ตัวได้แก่ IOPN , IOSET, IOCKR, IODIRโดยที่รีจิสเตอร์แต่ละตัวจะมีแอดเดรสแสดงในตารางที่ 2.2.15(ภาคผนวก)

ในไมโครคอนโทรลเลอร์ LPC2148 มีพอร์ตอเนกประสงค์ที่ให้ทำงานได้เร็วขึ้นจะเรียกว่าเป็นFast GPIO โดยให้กำหนดรีจิสเตอร์เพิ่มเติมขึ้นอีก แต่การทำงานที่รวดเร็วขึ้นจะต้องเขียนโปรแกรมเป็นภาษาแอสแซมบลีเท่านั้น

การสั่งงานของ GPIOเริ่มต้นด้วยการกำหนดทิศทางของพอร์ตก่อนว่าจะให้เป็นอินพุตหรือเอาท์พุต โดยกำหนดค่ารีสเตอร์ IODIR โดยค่าบิตใดเป็น 0 คือให้ขาของบิตนั้นเป็นอินพุต ถ้าให้ค่าเป็น 1 ขาของบิตนั้นจะเป็นเอาต์พุต

ตัวอย่างเช่น ต้องการให้ขา P0.22, P0.20, P0.19, และ P0.16 เป็นเอาท์พุต โดยขาของ P0 ที่เหลือเป็นอินพุต จะต้องกำหนดค่าให้กับรีจิสเตอร์ IODIR0 แต่ละบิตดังนี้

บิตที่	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ค่า	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	1
	0				0				5				9			

บิตที่	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ค่า	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0				0				0				0			

เขียนเป็นโปรแกรมภาษาซีได้ดังนี้

IODIR 0 = 0X00590000;

หลังจากการกำหนดให้พอร์ตเป็นเอาต์พุตแล้ว ถ้าต้องการสั่งให้พอร์ตที่เป็นเอาต์พุตนี้มีค่าเป็น 1 ต้องสั่งที่รีสเตอร์ IOSET จากตัวอย่างต้องสั่งให้ขา P0.16 และ P.19 เป็น 1 จะต้องกำหนดให้ค่า รีจิสเตอร์ IOSET0 ดังนี้

บิตที่	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ค่า	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
	0				0				0				9			

บิตที่	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ค่า	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0				0				0				0			

เขียนเป็นโปรแกรมภาษาซีได้ดังนี้

IOSET 0 = 0X00090000;

การเขียนค่าเป็น0 ให้รีจิสเตอร์ IOSET จะไม่มีผลต่อขาของพอร์ตที่ตรงกับบิตนั้น ขาของพอร์ตจะมีค่าคงเดิม ถ้าต้องการส่งให้พอร์ตที่เป็นเอาต์พุตนี้มีค่าเป็น 0 ต้องสั่งที่รีจิสเตอร์ IOCLR ด้วยจากตัวอย่าง ต้องการสั่งให้ขา P0.22 และ P0.20 เป็น 0 จะต้องกำหนดค่าให้กับ รีจิสเตอร์ IOCLR0 ดังนี้

บิตที่	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ค่า	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0
	0				0				5				0			

บิตที่	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ค่า	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0				0				0				0			

เขียนโดยใช้โปรแกรมภาษาซีได้ดังนี้

IOCLR = 0X00500000;

ข้อควรระวัง ในการสั่งให้ขาพอร์ตมีค่าเป็น 0 นี้ไม่ได้สั่งให้เขียนค่า 0 ให้กับรีจิสเตอร์ IOCLR ต้องเขียนค่าเป็น 1 เท่านั้น

ในการเขียนค่าให้พอร์ต จะต้องแยกข้อมูล ถ้าบิตใดเป็น 0 จะต้องเขียนสั่งที่รีจิสเตอร์ IOCLR ถ้าบิตใดเป็น 1 จะต้องเขียนสั่งที่รีจิสเตอร์ IOSET

2.2.5 อินเตอร์รัปต์ไมโครคอนโทรลเลอร์ ARM7

ในไมโครคอนโทรลเลอร์ ARM7 มีโมดูล Vectored Interrupt Controller (VIC) เป็นตัวควบคุมกลไกลของการอินเตอร์รัปต์ โดยสามารถรับอินเตอร์รัปต์จากอุปกรณ์ทั้งภายในและภายนอกไมโครคอนโทรลเลอร์ ARM7 ได้ทั้งหมด 32 อินพุตตามที่แสดงในตารางที่ 2.2.16 (ภาคผนวก)

VIC จะนำอินพุตจากการอินเตอร์รัปต์ทั้ง 32 แหล่งมาจัดแบ่งตามประเภทได้เป็น 3 ประเภท ทำให้สามารถปรับลำดับความสำคัญของอินเตอร์รัปต์จากอุปกรณ์ประกอบต่างๆได้ตามต้องการ คือ

- Fast Interrupt request(FIQ) จะลำดับความสำคัญสูงสุดโดยรีบตอบสนองเร็วสุด
- Vectored IRQs จะมีการลำดับความสำคัญอยู่กึ่งกลาง โดยสามารถนำอินเตอร์รัปต์แค่ 16 บิต หรือ 32 แหล่งมาจัดให้เป็น Vectored IRQs ได้ 16 ตัว โดย Slot 0 จะมีความสำคัญสูงสุด Slot 15 มีความสำคัญต่ำสุด
- Non-vectored IRQ มีความสำคัญต่ำสุด

รีจิสเตอร์ที่เกี่ยวข้องกับการอินเตอร์รัปต์มีทั้งหมด 43 ตัว ดังแสดงในตารางที่ 2.2.17 (ภาคผนวก) โดยในหัวข้อนี้จะเป็นการเลือกเฉพาะรีจิสเตอร์ที่เกี่ยวกับการอินเตอร์รัปต์ในแบบ Vector IRQ คือรีจิสเตอร์ VICVectAddr0-15 , VICVectCntl0-15 และ VICIntEanble

รีจิสเตอร์ VICVectAddr0-15 เป็นรีจิสเตอร์ที่เก็บค่าแอดเดรสของโปรแกรมที่ตอบสนองต่ออินเตอร์รัปต์โดย VICVectAddr0 จะเป็นอินเตอร์รัปต์ Slot 0 ที่มีความสำคัญสูงสุด ตามลำดับไปจนถึง VICVectAddr15 จะเป็นอินเตอร์รัปต์ Slot 15 ที่มีความสำคัญต่ำสุด

VICVectCntl0- 15 ใช้ในการควบคุม คือ Slot หมายเลขที่ตรงกับ VICVectCntl นี้จะรีบอินเตอร์รัปต์จากแหล่งใด จากทั้งหมด 32 แหล่ง ตามตารางที่ 2.2.17(ภาคผนวก) โดยกำหนดค่าในบิตที่ 4:0 โดยบิตที่ 5 ถ้าเป็น 1 หมายถึง อนุญาตให้อินเตอร์รัปต์จากอุปกรณ์ที่กำหนดใน Slot นี้

VICInEnable ใช้เปิดอินเตอร์รัปต์จากแหล่งต่างๆ ทั้ง 32 แหล่งตามตารางที่ 2.2.17 (ภาคผนวก) โดยบิตใดมีค่าเป็น 1 หมายถึงอนุญาตให้อินเตอร์รัปต์ได้ ถ้าบิตใดเป็น 0 ไม่อนุญาตให้อุปกรณ์นั้นๆ อินเตอร์รัปต์

การอินเตอร์รัปต์จากอินพุตภายนอกของไมโครคอนโทรลเลอร์ ARM7

ในไมโครคอนโทรลเลอร์ LPC2148 จะรับอินเตอร์รัปต์จากภายนอกได้สูงสุด 4 แหล่ง คือ EINT0, ENIT1, ENIT2 และ ENIT3 โดยคิดเป็นอินพุตได้ทั้งหมด 9 ตัว

- EINT0 อยู่ที่ขา P0.1 และ P0.16
- EINT1 อยู่ที่ขา P0.3 และ P0.14
- EINT2 อยู่ที่ขา P0.7 และ P0.15
- EINT3 อยู่ที่ขา P0.9, P0.20 และ P0.30

โดยสามารถนำอินพุตจากการอินเตอร์รัปต์ทั้งสี่ตัวนี้ มาใช้ในการกระตุ้นให้ไมโครคอนโทรลเลอร์ออกจากการทำงานในโหมด Power Down ได้

รีจิสเตอร์ที่เกี่ยวข้องกับอินเตอร์รัปต์จากภายนอก

รีจิสเตอร์ที่เกี่ยวข้องกับการอินเตอร์รัปต์จากภายนอกมีทั้งหมด 4 ตัว คือ EXTINT, EXTWAKE, EXXTMODE และ EXTPOLAR ดังแสดงในตารางที่ 2.2.18(ภาคผนวก)

เมื่อเกิดการอินเตอร์รัปต์จากภายนอกเกิดขึ้น จะทำให้ค่าบิตบางบิตของรีจิสเตอร์ EXTINT มีค่าเป็น 1 และส่งต่อการอินเตอร์รัปต์ไปยัง VIC เพื่อสร้างอินเตอร์รัปต์ขัดการทำงานของไมโครคอนโทรลเลอร์ เมื่อโปรแกรมตอบสนองต่อการอินเตอร์รัปต์ ทำงานเสร็จแล้วจะต้องเขียนค่า 1 ให้กับบิตเหล่านี้เพื่อหยุดการอินเตอร์รัปต์ ค่าแต่ละบิตของรีจิสเตอร์ EXTINT แสดงในตารางที่ 2.2.19
(ภาคผนวก)

ในการกำหนดว่าสัญญาณที่อินเตอร์รัปต์นี้ทำงานที่ระดับสัญญาณหรือที่ขาขอบของสัญญาณกำหนดได้ที่รีจิสเตอร์ EXTMODE ซึ่งมีค่าดังตารางที่ 2.2.20(ภาคผนวก)

หลังจากที่กำหนดโหมดการทำงานของสัญญาณที่ใช้อินเตอร์รัปต์ว่าเป็นระดับของสัญญาณหรือขอบสัญญาณแล้วที่รีจิสเตอร์ EXTMODE แล้วต้องกำหนดว่าสัญญาณที่อินเตอร์รัปต์นี้ทำงานที่ระดับลอจิก 1 หรือ 0 หรือทำงานที่ขอบขาขึ้นหรือขาลงของสัญญาณ โดยกำหนดที่รีจิสเตอร์ EXTPOLAR ซึ่งแสดงรายละเอียดดังตารางที่ 2.2.21(ภาคผนวก)

ถ้าต้องการให้สัญญาณที่ขา EINT0-EINT3 สามารถกระตุ้นให้ไมโครคอนโทรลเลอร์ ARM7 ออกจากการทำงานในโหมด Power Down จะต้องสั่งที่รีจิสเตอร์ EXTWAKE ซึ่งมีค่บิตของรีจิสเตอร์แสดงในตารางที่ 2.2.22(ภาคผนวก)

2.2.6 การกำหนดค่าเริ่มต้นสำหรับ UART0

UART0 มีขาสัญญาณสองขา คือ

1. ขาสัญญาณ TxD0 ไว้สำหรับส่งข้อมูลอยู่ที่ขา
2. ขาสัญญาณ RxD0 ไว้สำหรับรับข้อมูลอยู่ที่ขา P0.1

การใช้งาน UART0 เริ่มต้นด้วยการเขียนค่ายังรีจิสเตอร์ PINSEL0 เพื่อกำหนดให้ขา P0.0 และ P0.1 มีการทำงานเป็น UART0 โดยต้องกำหนดบิตที่ 3-0 ของ PINSEL0 ให้มีค่าเป็น 1010 ซึ่งเขียนเป็นคำสั่งได้ดังนี้

PINSEL0 = 0x00000005;

รีจิสเตอร์ที่เกี่ยวข้องกับ UART0 มีทั้งหมด10 ตัวโดยแต่ละตัวมีขนาด 8 บิต ดังแสดงในตารางที่ 2.2.23(ภาคผนวก)

หลังจากที่กำหนดให้ขา P0.0 และ P0.1 ทำงานเป็น UART0 แล้วถัดมาเป็นการกำหนดรูปแบบการติดต่อเช่น ติดต่อแบบ 8 ใช้การตรวจสอบบิตพลาดแบบใด เช่น even parity จำนวนของ Stop bit โดยการกำหนดค่าผ่านทางรีจิสเตอร์ UART Line Control Register : LCR : ซึ่งมีรายละเอียดของรีจิสเตอร์ดังแสดงในตารางที่ 2.2.24 (ภาคผนวก)

ในรีจิสเตอร์ LCR มีบิตที่เรียกว่า DLAB (Divisor Latch Access bit) ถ้าต้องการปรับค่าของวงจรกำเนิด Broad Rate ต้องเซตบิตนี้ให้เป็น 1ค่าของ Baud rate generator เป็นค่าของตัวหารขนาด 16 บิต เพื่อนำไปใช้หารค่าของ PCLK เพื่อให้ได้ความถี่สูงกว่าความเร็ว Broad Rate 16 เท่า ทำให้สมการค่าตัวหารเป็นดังนี้

Divisor = PCLK/(16*Baud)

แผงวงจร CP-JR ARM7 USB-LPC2148 EXP ใช้คริสตัลความถี่ 12.000MHz สั่ง PLL คูณ 5 จะได้ CCLK = 60.0 MHz และกำหนดให้ VPBDIV = 2 จะได้ PCLK = 30.0 MHz ด้วย ถ้าต้องการอัตราบอดที่ 9600 bps ตัวหารจะมีค่าเท่ากับ

Divisor = 30,000,000/(60*9600) = 195.3125 ปัดเศษลง

Divisor = 195 หรือ 0xC3 นำค่าที่ได้ไปคำนวณหาอัตราบอดจะได้

Baud = PCLK/(16*Divisor) = 30,000,000/(16*195) = 9615 bps

ซึ่งผิดพลาดไป 0.156% สามารถให้งานได้ เนื่องมาตรฐานของการสื่อสารแบบอนุกรมสามารถรับอัตราบอดที่ผิดพลาดได้ถึง 5% นำค่าหารที่ได้ไปเก็บลงในรีจิสเตอร์ขนาด 8 บิตสองตัว คือ Divisor Latch MAB(DLM) และ Divisor Latch LSB (DLL) ในขณะที่เขียนค่าเก็บในรีจิสเตอร์

DLM และ DLL ค่าบิต DLAB ต้องมีค่าเป็น 1 เมื่อเขียนเสร็จแล้วต้องรีเซตค่าบิตนี้ให้กลับเป็น 0 ซึ่งเขียนเป็นคำสั่งภาษาซีได้ดังนี้

```
U0DLM = 0x00;
U0DLL = 0xC3;
U0LCR = 0x7F;
```

ในการเรียกใช้ฟังก์ชัน uart0_init() จะต้องส่งค่าอัตราบอดที่ต้องการให้ฟังก์ชัน ตัวอย่างเช่นต้องการอัตราบอดที่ 9600 bps จะต้องเรียกใช้ฟังก์ชันดังนี้

```
uart0_init(9600);
```

เมื่อกำหนดการทำงานให้กับ UART แล้ว จะสามารถรับส่งค่าผ่านพอร์ตอนุกรมได้ในการส่งข้อมูลต้องเขียนข้อมูลไปยังรีจิสเตอร์ Transmit Holding Register (THR) ถ้าต้องการอ่านข้อมูลที่รับพอร์ตอนุกรมต้องอ่านค่าจากรีจิสเตอร์ Receiver Buffer Register (RBR) จากตารางที่ 2.2.24 (ภาคผนวก) จะพบว่าค่าแอดเดรสของรีจิสเตอร์ทั้งสอง มีค่าอยู่ที่ตำแหน่งเดียวกัน แสดงว่าการเขียนค่าให้กับ THR เป็นการเขียนค่าลงในบัฟเฟอร์แบบ FIFO ของ UART0 การอ่านค่าจากรีจิสเตอร์ RBR เป็นการอ่านค่าจากบัฟเฟอร์แบบ FIFO

ก่อนที่จะอ่านหรือเขียนค่าลงในรีจิสเตอร์ THR หรือ RBR จะต้องอ่านค่าสถานะของ UART ก่อนว่ามีการผิดพลาดหรือไม่ โดยอ่านค่าสถานะที่รีจิสเตอร์ Line Status Register (LSB) ก่อน โดยค่าประจำบิตของรีจิสเตอร์แสดงได้ในตารางที่ 2.2.25(ภาคผนวก)

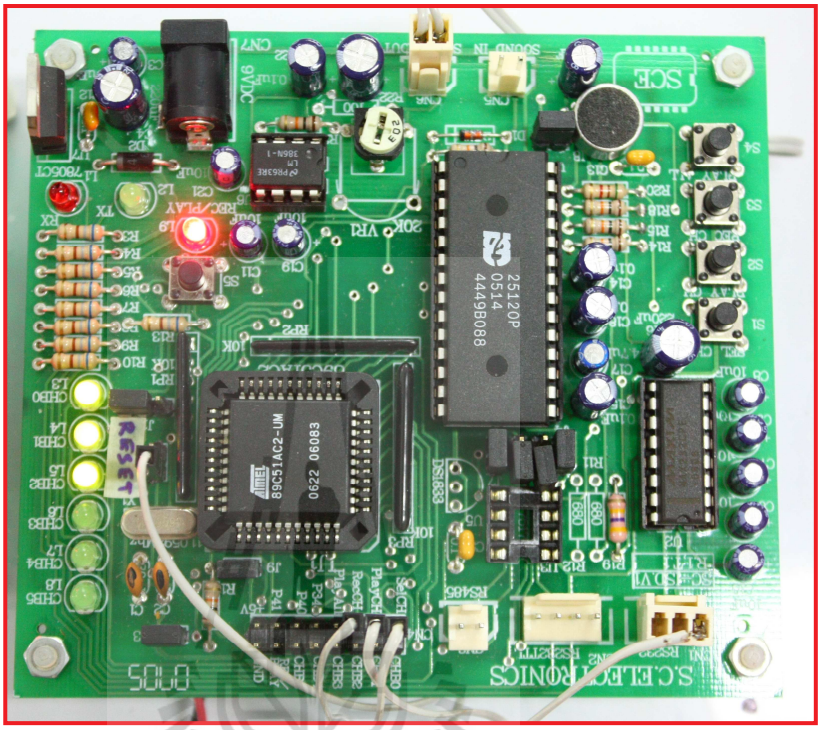
ก่อนที่จะเขียนข้อมูลให้ UART ต้องตรวจสอบดูที่บิต Transmitter Empty (TEMT) ของรีจิสเตอร์ LSR ก่อนว่าบัฟเฟอร์สำหรับส่งว่างหรือไม่ ถ้าว่างจะได้ค่าเป็น 1 จึงส่งข้อมูลได้ก่อนที่จะอ่านข้อมูลจาก UART ต้องตรวจสอบบิต Receiver Data Ready (RDR) ก่อน ถ้ามีข้อมูลพร้อมแล้วบิตนี้จะมีค่าเป็น 1 จึงอ่านค่าจาก UART ได้

สามารถเขียนเป็นฟังก์ชัน putchar(); สำหรับเขียนข้อมูลจำนวน 1 ไบต์ให้กับ UART และเขียนเป็นฟังก์ชัน getchar() สำหรับอ่านค่าจาก UART ได้

สำหรับการเขียนข้อมูลออกพอร์ตอนุกรม ในโปรแกรม Keil uVision3 ได้จัดเตรียมฟังก์ชัน printf(); ไว้ให้แล้ว และถ้าต้องการรับข้อมูลจากพอร์ตอนุกรม สามารถใช้ฟังก์ชัน scanf(); โดยต้องสั่ง #include<stdio.h>

2.3 บอร์ดบันทึกและเล่นเสียง (Voice Record Module)

บอร์ดบันทึกและเล่นเสียง (Voice Record Module) มีลักษณะต่าง ๆ ดังรูปที่ 2.11



รูปที่ 2.11ลักษณะของบอร์ดบันทึกและเล่นเสียง

2.3.1 คุณสมบัติบอร์ดของบันทึกและเล่นเสียง

- บันทึกเสียงได้ 60 วินาที สำหรับไอซีเบอร์ ISD2560 และ 120 วินาทีสำหรับไอซีเบอร์ ISD25120
- กำหนดช่องเสียงในการบันทึกได้ 1-64 ช่องเสียง
- ต่อเชื่อมการควบคุมด้วย RS232 หรือ RS485 ได้
- ควบคุมการบันทึกเสียงด้วย TTL บิทได้
- มีสวิทช์บนบอร์ดควบคุมการบันทึก
- มีไมค์ในตัวบันทึกเสียงได้เลย
- มีภาคขยายเสียงขนาด 0.25W
- สามารถนำสัญญาณเสียงจากภายนอกมาบันทึกได้

2.3.2 การนำไปใช้งานของบอร์ดบันทึกและเล่นเสียง

การประยุกต์เข้ากับโครงงานด้านระบบเสียงต่างๆ เช่น ระบบเตือนภัย นาฬิกาบอกเวลาด้วยคำพูด บอกอุณหภูมิและอื่นๆ โดยนำค่าในแต่ละช่องเสียงมาเชื่อมกันเป็นประโยคคำพูด

2.3.3 การตั้งค่าเริ่มต้นของบอร์ดบันทึกและเล่นเสียง

ก่อนเริ่มใช้งานต้องกำหนดจัมเปอร์(Jumper)ให้เหมาะสมกับการทำงานก่อนซึ่งสามารถดูตำแหน่งของจัมเปอร์(Jumper) ได้จากวงจรตามรูปที่2.11

1. J1 ➔ เลือกระบบ POWER ON RESET ว่าเราต้องการจะใช้ RC Reset หรือ IC Reset ในบอร์ดนี้ให้ใช้ RC Reset (ดูวงจรประกอบ)
2. J2 ➔ ต้อง OFF เสมอ
3. J3 ➔ ON คือ เลือกไอซีเบอร์ ISD25120
   OFF คือเลือกไอซีเบอร์ ISD2560
   J7 ➔ ON คือ RECORD ENABLE ทำให้สามารถบันทึกเสียงได้
   OFF คือ RECORD DISABLE ทำให้ไม่สามารถบันทึกเสียงได้เพื่อป้องกันการบันทึกซ้ำโดยไม่ตั้งใจ
4. J8 ➔ ON คือเลือกสัญญาณที่จะบันทึกจากไมล์บนบอร์ด
   OFF คือเลือกสัญญาณที่จะบันทึกจากภายนอกต่อเข้ามาที่ CN5(SOUND IN)
5. J9 ➔ ON คือให้ไฟเลี้ยงไปปรากฏที่ CN4
   OFF คือตัดไฟเลี้ยงที่ CN4
6. J4, J5, J6 เพื่อกำหนดจำนวนช่องเสียงสำหรับการบันทึกดังนี้

J4	J5	J6	จำนวนช่องเสียง
OFF	OFF	NO	1
OFF	NO	OFF	2
OFF	NO	NO	4
NO	OFF	OFF	8
NO	NO	NO	16
NO	NO	OFF	32
NO	NO	NO	64

*** **หมายเหตุ** เมื่อทำการกำหนดจัมเปอร์(Jumper)เสร็จให้กดปุ่ม Reset (S5) หรือทำการปิด และเปิดแหล่งจ่ายใหม่

2.3.4 การคำนวณค่าเวลาของช่องบันทึกเสียงและเล่นเสียง

สูตรการคำนวณระยะเวลาของแต่ละช่องเสียง

ระยะเวลาช่องเสียง = ระยะเวลาทั้งหมด / จำนวนช่องเสียง

กรณีใช้ไอซีเบอร์ ISD2560

ระยะเวลาการบันทึกเสียงทั้งหมดคือ 60 วินาที ถ้ากำหนดจำนวนช่องเสียงไว้ที่ 8 ช่องเสียง ดังนั้นระยะเวลาของการบันทึกในแต่ละช่องเสียงมีค่าเท่ากับ 60/8 = 7.5 วินาที ถ้ากำหนด จำนวนช่องเสียงไว้ที่ 32 ช่องเสียง ระยะเวลาของการบันทึกในแต่ละช่องเสียงเท่ากับ 60/32 = 1.87 วินาที

กรณีใช้ไอซีเบอร์ ISD25120

ระยะเวลาการบันทึกเสียงทั้งหมดคือ 120 วินาที ถ้ากำหนดจำนวนช่องเสียงไว้ที่ 8 ช่องเสียง ดังนั้นระยะเวลาของการบันทึกในแต่ละช่องเสียงมีค่าเท่ากับ120 / 8 = 15 วินาที ถ้ากำหนดจำนวนช่องเสียงไว้ที่ 32 ช่องเสียง ระยะเวลาของการบันทึกในแต่ละช่องเสียงเท่ากับ 120 / 32 = 3.75 วินาที

2.3.5 การใช้งานบอร์ดบันทึกเสียงและเล่นเสียง

S1 ➔ ใช้กำหนดช่องสำหรับการบันทึกหรือเล่นเสียงโดยแสดงหมายเลขช่องด้วย LED ในรูปแบบ ของเลขฐาน 2 (binary)

S2 ➔ ใช้เล่นเสียงในช่องเสียงที่กำหนด

S3 ➔ ใช้บันทึกเสียงและหยุดการบันทึกเสียงในช่องเสียงที่กำหนด

S4 ➔ ใช้เล่นเสียงทุกช่องเสียง

2.3.6 การควบคุมผ่าน RS232

การควบคุมผ่าน RS232 หรือ RS485 ทำได้โดยการกำหนดการสื่อสารดังนี้ ความเร็ว 9600 bps , ข้อมูล 8 บิท, ไม่มีพาริตี้บิท, มี 1 STOP BIT คำสั่งที่ใช้ในการเล่นเสียงดังนี้

STPLY:XX:ED

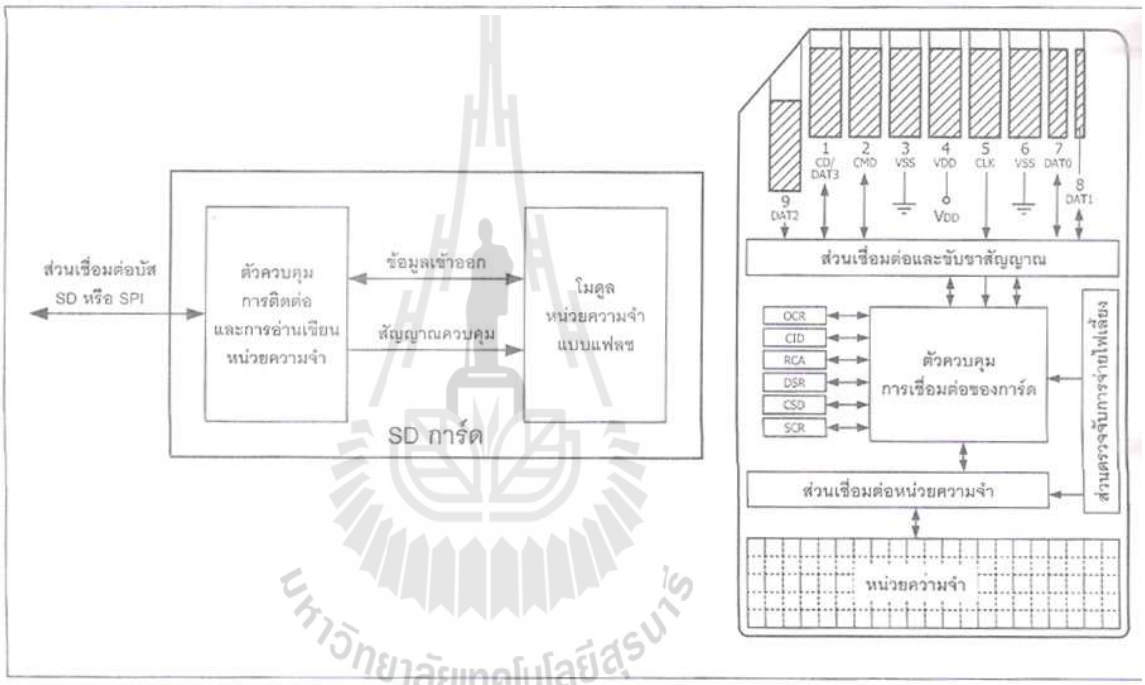
เมื่อ XX มีค่า 00 – 63 ซึ่งก็คือหมายเลขช่องที่ต้องการเล่นเสียง

*** **หมายเหตุ** สามารถส่งคำสั่งควบคุมทีละหลายๆคำสั่งได้เพื่อให้เกิดการเล่นเสียงอย่างต่อเนื่อง โดยการทำงานของบอร์ดจะมีบัฟเฟอร์(Buffer) จัดการกับคำสั่งที่เข้ามาอย่างเป็นลำดับ

2.4 ชุดเชื่อมต่อหน่วยความจำ SD/MMC CARD

2.4.1 ความรู้เบื้องต้นเกี่ยวกับ SD การ์ด

SD การ์ดเป็นหน่วยความจำแบบเขียนและลบใหม่ได้แบบหนึ่งที่ใช้เทคโนโลยีหน่วยความจำแบบแฟลช(Secure Digital Card) มีลักษณะการทำงานและการติดต่อคล้ายกับการ์ดหน่วยความจำแบบ MMC (Multi Media Card) หากแต่ใน SD การ์ดได้บรรจุส่วนการรักษาข้อมูลเข้าไปเพิ่มเติม ในรูปที่2.12 แสดงไดอะแกรมการทำงานของ SD การ์ด จะเห็นว่า มีส่วนประกอบ 2 ส่วนคือโมดูลหน่วยความจำแบบแฟลชและตัวควบคุม การติดต่อกับ SD หรือบัส SPI



รูปที่ 2.12ไดอะแกรมการทำงานเบื้องต้นของ SD การ์ด

2.4.2 คุณสมบัติเด่นของ SD การ์ด

SD การ์ดเกิดขึ้นจากความร่วมมือของ 3 บริษัทคือ Matsushita Electric Industrial (MEI),SanDisk Corporation (SanDisk) และ Toshiba Corporation (Toshiba) มีการกำหนดคุณสมบัติต่างๆรวมถึงมาตรฐานการติดต่อที่ชัดเจนภายใต้การกำกับดูแลโดย SD Card Association (www.sdcard.org)

ในปัจจุบัน SD การ์ดได้รับความนิยมสูงมากโดยเฉพาะในอุปกรณ์สารสนเทศสมัยใหม่ ไม่ว่าจะเป็นกล้องดิจิตอล โทรศัพท์เคลื่อนที่ เครื่องเล่น MP3 เป็นต้น ทั้งนี้เนื่องจาก SD การ์ดได้รับการออกแบบให้มีความโดดเด่นในทุกด้านที่หน่วยความจำชั้นดีพึงมี 5 ประการ ดังนี้

คุณสมบัติทางเทคนิคที่สำคัญของ SD การ์ด

1. สามารถเก็บข้อมูลได้ถึง 8GB (ในขณะจัดทำเอกสารนี้)
2. รองรับการติดต่อแบบหนึ่งสายสัญญาณ และแบบ 4 สายสัญญาณ รวมทั้งแบบบัส SPI
3. สามารถป้องกันการคัดลอกข้อมูลลิขสิทธิ์ได้
4. สามารถลบ-เขียนใหม่ในแต่ละเซกเตอร์ได้ 100,000 ครั้ง
5. สามารถเก็บรักษาข้อมูลได้นานมากกว่า 10 ปี

2.4.3 ระบบบัสที่ใช้ติดต่อกับ SD การ์ด

การติดต่อกับ SD การ์ดสามารถกระทำได้ 2 วิธีคือ

1.ผ่านทางบัส SD

2.บัส SPI

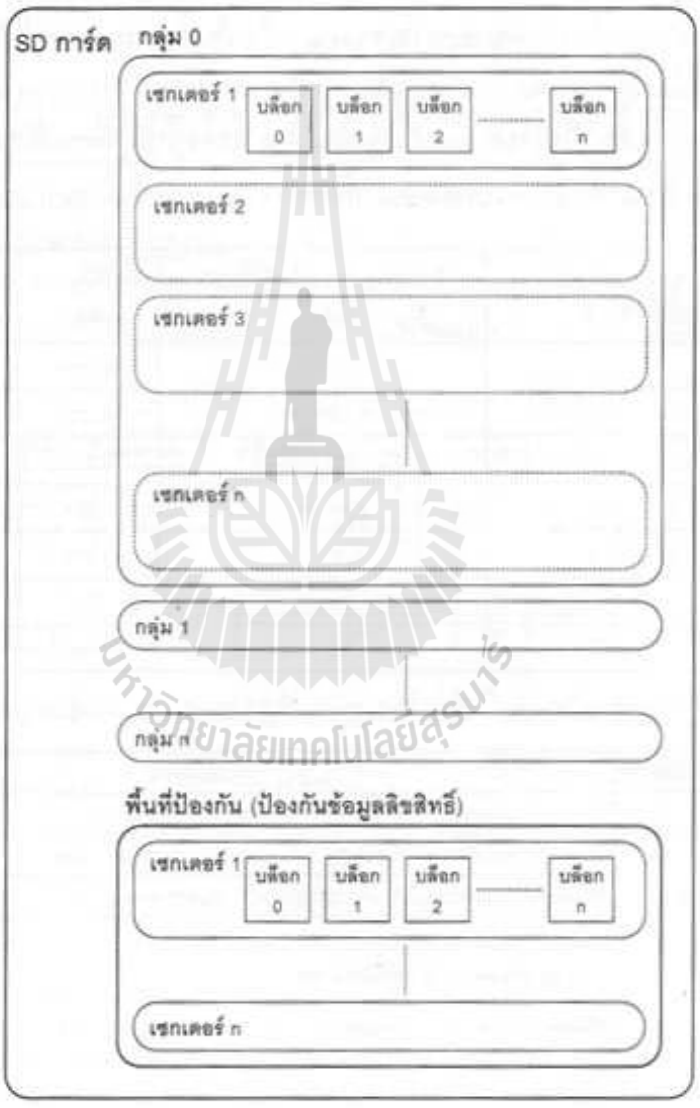
โดยสามารถสรุปได้ดังตารางที่ 2.4.1(ภาคผนวก)

ขาสัญญาณของ SD การ์ด

ขาสัญญาณมาตรฐานของ SD การ์ดมีทั้งสิ้น 9 ขา โดยมีลักษณะเป็นหน้าสัมผัสโลหะ ดังแสดงในรูปที่ 2.12 ส่วนการกำหนดชื่อและหน้าที่ของขาสัญญาณจะสัญญาณจะแตกต่างกันตามรูปแบบของการติดต่อดังสรุปได้ในตารางที่ 2.4.1(ภาคผนวก) และ 2.4.2(ภาคผนวก) โดยในตารางที่ 2.4.2(ภาคผนวก) เป็นการจัดขาเมื่อติดต่อ SD การ์ดด้วยบัส SD ส่วนตารางที่ 2.4.3(ภาคผนวก) เป็นการจัดขาเมื่อทำงานผ่านบัส SPI

2.4.4 การจัดแบ่งพื้นที่ของ SD การ์ด

หน่วยที่เล็กสุดของการถ่ายทอดข้อมูลใน SD การ์ดคือ 1 ไบต์(byte) ส่วนการถ่ายทอดข้อมูลจริงนั้น ควรกระทำในลักษณะบล็อกข้อมูล โดยสามารถกำหนดขนาดของบล็อกได้ โดยในแต่ละบล็อกสามารถบรรจุข้อมูลได้หลายๆ ไบต์ แต่โดยปกติแล้วมักจะเลือกใช้ที่บล็อกละ 512 ไบต์ ทั้งนี้เพื่อให้สอดคล้องกับระบบ FAT (File Allocation Table) หรือตารางสำหรับจัดวางแฟ้มข้อมูลซึ่งใช้ในระบบคอมพิวเตอร์



รูปที่ 2.13 การจัดแบ่งพื้นที่ของ SD การ์ด

จากรูปที่2.13 มีการจัดสรรเป็น 3 ส่วนหลักคือ บล็อกข้อมูล เป็นกลุ่มของข้อมูลที่ได้รับการกำหนดขนาดจากผู้ใช้งาน และนำไปใช้ในคำสั่ง และเขียนบล็อกข้อมูล สำหรับการกำหนดและตรวจสอบขนาดของบล็อกข้อมูลสามารถกระทำได้ที่รีจิสเตอร์

เซกเตอร์ เป็นหน่วยของพื้นที่ข้อมูลใน SD การ์ดที่สัมพันธ์กับคำสั่งลบ ใน 1 เซกเตอร์มีหลายบล็อกข้อมูล โดยได้รับการกำหนดมาตายตัวจากผู้ผลิต ผู้ใช้งานสามารถตรวจสอบขนาดของเซกเตอร์ได้จากรีจิสเตอร์

กลุ่มป้องกันการเขียน (WP Group) เป็นพื้นที่ของหน่วยความจำที่ได้รับการจัดแบ่งให้น้อยที่สุดใช้เพื่อบรรจุลิขสิทธิ์ที่ไม่ต้องการให้เกิดการเขียนทับ ขนาดของพื้นที่จะ ได้รับการกำหนดมาตายตัวเช่นกันผู้ใช้งานสามารถตรวจสอบขนาดของพื้นที่ได้จากรีจิสเตอร์ CSD

2.4.5 รีจีสเตอร์ของ SD การ์ด

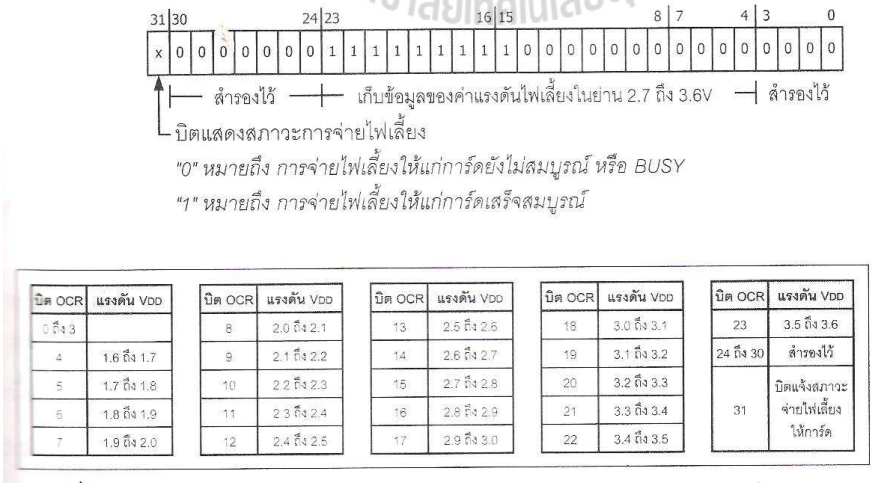
มีทั้งหมด 6 ตัว โดยเป็นรีจิสเตอร์หลักที่ใช้ 4 ตัว,รีจิสเตอร์พิเศษ 1 ตัว และรีจิสเตอร์เสริมอีก 1 ตัว ดังแสดงในตารางที่ 2.4.4 (ภาคผนวก)

➔รีจิสเตอร์ OCR (Operating Condition Register)

เป็นรีจิสเตอร์เก็บข้อมูลของค่าแรงดันไฟเลี้ยงของ SD การ์ด สำหรับตรวจสอบแรงดันของ SD การ์ด ปกติแรงดันไฟเลี้ยงของ SD การ์ดอยู่ในช่วง 2.7 ถึง 3.6 V ดังนั้นค่าของรีจิสเตอร์ OCR ควรเท่ากับ

➔รีจิสเตอร์ CID (Card Identification Register)

เป็นรีจิสเตอร์ที่มีความยาว 16 ไบต์ ใช้ในการเก็บข้อมูลเฉพาะของ SD การ์ด ซึ่งกำหนดมาจากผู้ผลิต ผู้ใช้งานไม่สามารถทำการเปลี่ยนแปลงได้ โดยค่าและความหมายของข้อมูลในรีจิสเตอร์ CID ใน SD การ์ดจะแตกต่างจาก MMC



บิต OCR	แรงดัน VDD
0 ถึง 3	
4	1.6 ถึง 1.7
5	1.7 ถึง 1.8
6	1.8 ถึง 1.9
7	1.9 ถึง 2.0

บิต OCR	แรงดัน VDD
8	2.0 ถึง 2.1
9	2.1 ถึง 2.2
10	2.2 ถึง 2.3
11	2.3 ถึง 2.4
12	2.4 ถึง 2.5

บิต OCR	แรงดัน VDD
13	2.5 ถึง 2.6
14	2.6 ถึง 2.7
15	2.7 ถึง 2.8
16	2.8 ถึง 2.9
17	2.9 ถึง 3.0

บิต OCR	แรงดัน VDD
18	3.0 ถึง 3.1
19	3.1 ถึง 3.2
20	3.2 ถึง 3.3
21	3.3 ถึง 3.4
22	3.4 ถึง 3.5

บิต OCR	แรงดัน VDD
23	3.5 ถึง 3.6
24 ถึง 30	สำรองไว้
31	บิตแสดงสภาวะจ่ายไฟเลี้ยงให้การ์ด

รูปที่ 2.14 ความสัมพันธ์ของบิตข้อมูลในรีจิสเตอร์ OCR กับแรงดันของ SD การ์ด

➔รีจิสเตอร์ CSD (Card Specific Data)

เป็นรีจิสเตอร์ขนาด 16 ไบต์ (128 บิต) ที่ใช้เก็บข้อมูลคุณสมบัติเฉพาะของ SD การ์ด ซึ่งมีรายละเอียดค่อนข้างมากเพราะรีจิสเตอร์นี้บรรจุข้อมูลเกี่ยวกับความจุ,อัตราเร็วในการถ่ายทอดข้อมูล , แรงดันและกระแสไฟฟ้า ในขณะอ่านและเขียนข้อมูล,รูปแบบของไฟล์,การป้องกันข้อมูล, การลบและข้อมูลเกี่ยวกับการเขียนข้อมูลลงใน SD การ์ด สำหรับในการทดลองนี้เลือกใช้ 2 ข้อมูล คือ C_SIZE (บิต 73: 62) และ C_SIZE_MUL(บิต 49:47) เพื่อนำมาคำนวณหาความจุของ SD การ์ดที่ติดต่อด้วย

ส่วนข้อมูลอื่นๆ เพิ่มเติมของรีจิสเตอร์ตัวนี้สามารถอ่านได้จากไฟล์ดาต้าชีตของ SD การ์ดในซีดีรอมที่จัดมาพร้อมกับบอร์ด JX-2148

➔รีจิสเตอร์ RCA (Relative Card Address)

เป็นรีจิสเตอร์ขนาด 16 บิตใช้เก็บค่าแอดเดรสของหน่วยความจำแบบสัมพันธ์ ซึ่งทางโฮสต์ (หมายถึง คอมพิวเตอร์หรือไมโครคอนโทรลเลอร์) สามารถเลือกกำหนดได้ อย่างไรก็ตาม หากเลือกการติดต่อ SD การ์ดแบบ SPI จะไม่สามารถติดต่อกับรีจิสเตอร์ตัวนี้ได้

➔รีจิสเตอร์ SCR (SD Configuration Register)

เป็นรีจิสเตอร์ขนาด 64 บิต ที่ใช้เก็บค่าคุณสมบัติพิเศษของ SD การ์ด ที่เพิ่มเติมนอกเหนือไปจากที่เก็บในรีจิสเตอร์ CSD ซึ่งข้อมูลทั้งหมดนี้ได้รับการกำหนดมาจากผู้ผลิตเช่นกัน มีทั้งสิ้น 5 ข้อมูล คือ ข้อมูลเวอร์ชันของ SCR (บิต 63:60 รวม 4 บิต),ข้อมูลเวอร์ชันของคุณสมบัติทางกายภาพของ SD การ์ด (บิต 59:56 รวม 4 บิต ใช้จริงบิตเดียว), ข้อมูลสถานะของข้อมูลหลังจากการลบ (1 บิตคือ บิต 55), ข้อมูลกำหนดระดับการป้องกัน(บิต 54 : 52 รวม 3 บิต),ข้อมูลแจ้งการรับรองขนาดของข้อมูลที่ทำการถ่ายทอดได้ของ SD การ์ด(บิต 47:32) และสำรองสำหรับใช้เฉพาะผู้ผลิตอีก 32 บิต (บิต 31: 0)

➔รีจิสเตอร์ DSR (Drive Stage Register)

เป็นรีจิสเตอร์ขนาด 16 บิต สำหรับเก็บค่าคุณสมบัติของไดรเวอร์ทางเอาต์พุตของ SDIO การ์ดจะมีความแตกต่างกันไปในอุปกรณ์เอาต์พุตแต่ละตัว

ดังนั้น รีจิสเตอร์หลักๆที่ใช้จะมี 3 ถึง 4ตัวคือ OCR,CID,CSD และ RCA สำหรับการทดสอบเบื้องต้นจะใช้เพียง 2 ตัวคือ CID และ CSD

รีจิสเตอร์แสดงสถานะการทำงานของ SD การ์ดมี 2 ตัว คือ Card Status และ SD_Status โดย Card Status มีขนาด 32 บิตใช้แสดงสถานะการทำงานปกติ มีการทำงานเหมือนกับของ MMC การ์ด SD_Status มีขนาด 512 บิต สามารถแสดงสถานะการทำงานพิเศษที่เพิ่มเติมไปจาก Card Status โดยข้อมูลสถานะจะถูกส่งส่งลงไปบนสายนำสัญญาณ DAT พร้อมกับรหัสตรวจสอบ CRC 16 บิต

รีจิสเตอร์ทั้งสองตัวนี้มีการจำแนกชนิดของสถานะการทำงานออกเป็น 4 แบบ และสามารถเคลียร์บิตได้ด้วยเงื่อนไขที่แตกต่างกันอีก 3 เงื่อนไข สามารถสรุปได้ดังนี้

• ชนิดของของสถานะการทำงาน

E - บิตแจ้งความผิดพลาด

S - บิตแจ้งสถานะ

R - บิตแจ้งการตรวจจับและเซตเมื่อได้รับการตอบสนองคำสั่ง

X - บิตแจ้งการตรวจจับและเซตในขณะที่กำลังกระทำคำสั่ง หากต้องการอ่านบิตนี้ ซีพียูจะต้องส่งคำสั่งอ่านสถานะมายัง SD การ์ดก่อน

• เงื่อนไขในการเคลียร์บิตแจ้งสถานะ

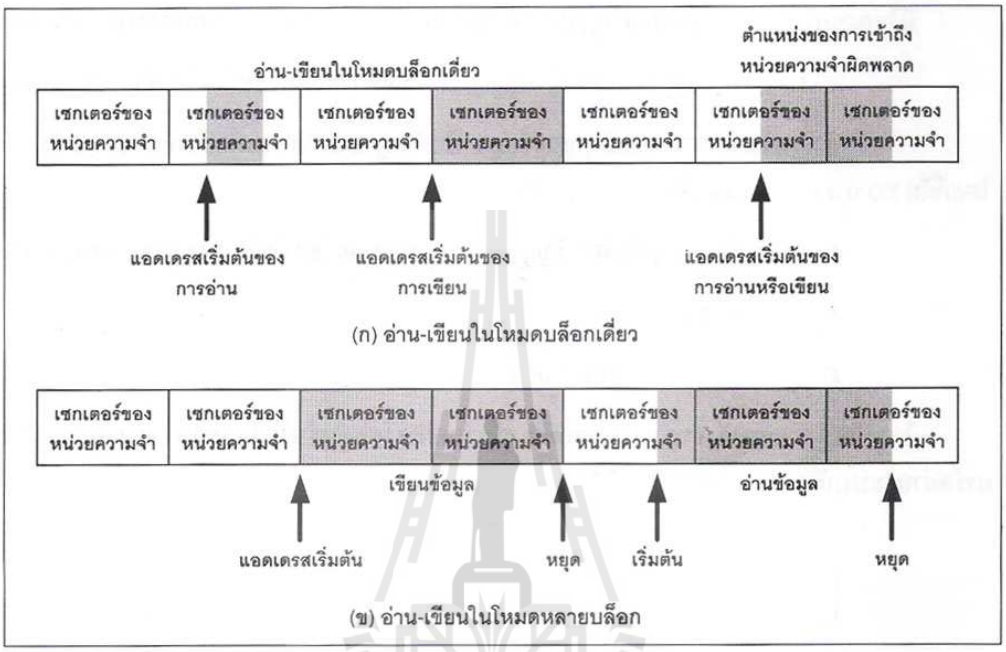
A - เคลียร์ด้วยกระบวนการทำงานตามปกติ

B - เคลียร์เนื่องจากผลของคำสั่งก่อนหน้า ดังนั้นบิตสถานะจะเคลียร์หลังจากทำงานผ่านไป 1 คำสั่ง หรือเป็นการสั่งเคลียร์บิตสถานะโดยตรง

C - เคลียร์ด้วยการอ่าน

2.4.6 กระบวนการอ่าน-เขียน SD การ์ด

SD การ์ดมีกระบวนการอ่าน-เขียนข้อมูล 2 โหมด ดังแสดงในรูปที่ 2.15 โดยมีอัตราการถ่ายทอดข้อมูล 25 เมกะบิตต่อวินาทีในกรณีใช้สายเดียว (ติดต่อแบบบัส SPI) และ 100 เมกะบิตต่อวินาทีในกรณีใช้สายข้อมูล 4 เส้น (ติดต่อแบบบัส SD)



รูปที่ 2.15 กระบวนการอ่าน-เขียนข้อมูลของ SD การ์ด

2.4.7 การติดต่อกับ SD การ์ด

โฮสต์หรือคอมพิวเตอร์หรือไมโครคอนโทรลเลอร์สามารถติดต่อกับ SD การ์ดได้ 2 วิธี คือ ผ่านบัส SD และบัส SPI โดยใช้สายสัญญาณที่แตกต่างกันดังแสดงรายละเอียดในตารางที่ 2.4.5 (ภาคผนวก)

การติดต่อ SD การ์ดผ่านบัส SD

มีไดอะแกรมแสดงการติดต่อตามรูปที่ 2.16 ใช้สายสัญญาณ 6 เส้น และสายพลังงาน 3 เส้น

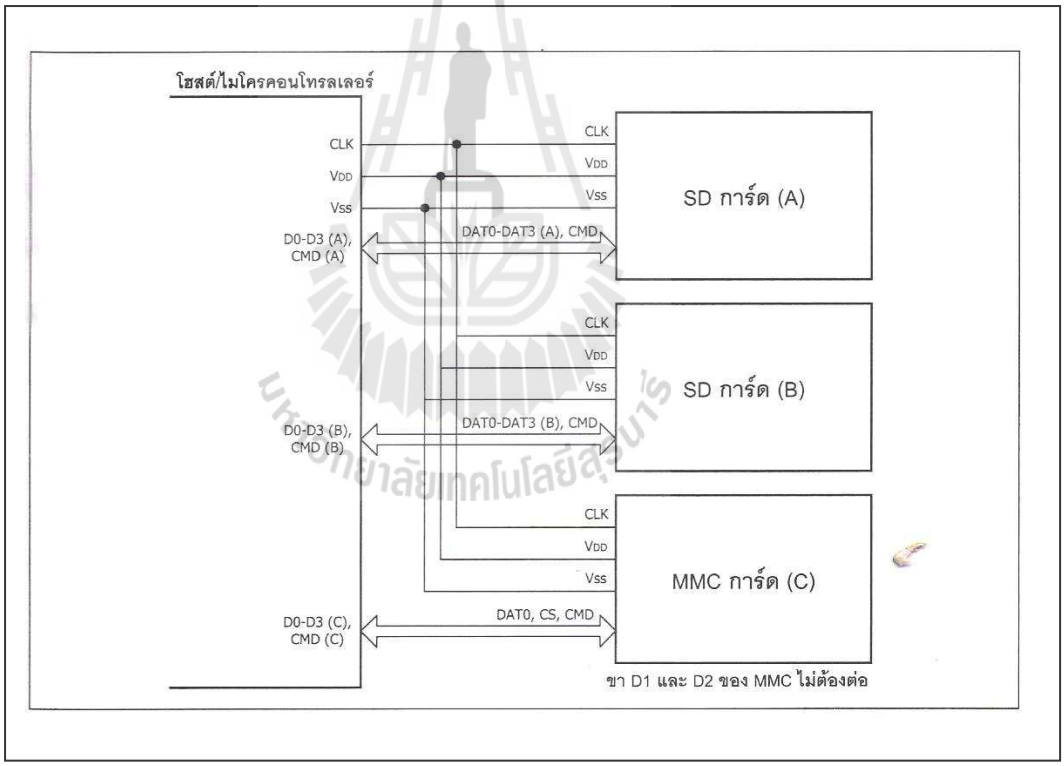
•CMD : สายสัญญาณคำสั่ง เป็นสัญญาณ 2 ทิศทางติดต่อระหว่างโฮสต์กับ SD การ์ด

•DAT0 ถึง DAT3 : สายสัญญาณข้อมูลเป็นสัญญาณ 2 ทิศทางเพื่อถ่ายทอดข้อมูลระหว่างโฮสต์กับ SD การ์ดมีทั้งสิ้น 4 เส้น

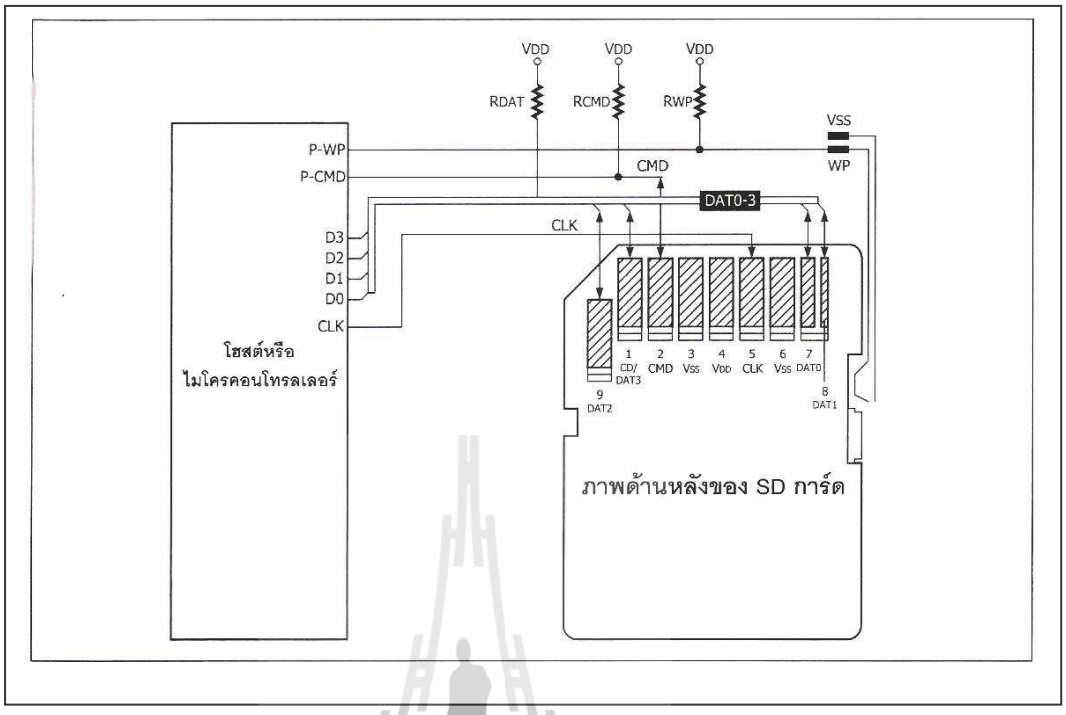
•CLK : สายสัญญาณนาฬิกา สัญญาณนี้จะส่งออกจากโฮสต์เพื่อกำหนดจังหวะการทำงาน

•VDD : สายไฟเลี้ยง

•GND : สายกราวด์ (ปกติมี 2 เส้น)



รูปที่ 2.16 ไดอะแกรมการติดต่อกับSD การ์ดผ่านบัสSD



รูปที่ 2.17 วงจรการเชื่อมต่อเบื้องต้นระหว่างโฮสต์
หรือไมโครคอนโทรลเลอร์กับ SD การ์ดผ่านระบบบัส SD

รูปแบบการติดต่อกับ SD การ์ดผ่านระบบบัส SPI

กลุ่มข้อมูลที่ใช้ในการติดต่อบนระบบบัส SPI หรือเรียกว่า SPI message ประกอบด้วย คำสั่ง (command), การตอบสนอง (response) และบล็อกข้อมูล (data-block) การสื่อสารระหว่าง โฮสต์หรือ ไมโครคอนโทรลเลอร์กับ SD การ์ดจะได้รับการกำหนดจังหวะจากโฮสต์ โดยโฮสต์จะ เริ่มต้นการติดต่อด้วยการทำให้สายสัญญาณ CS เป็นลอจิก “0”

การตอบสนองของ SD การ์ดในการติดต่อผ่านบัส SPI มีหลักเกณฑ์ดังนี้

(1) SD การ์ดที่ถูกเลือกให้ติดต่อจะต้องตอบสนองต่อทุกคำสั่งเสมอ

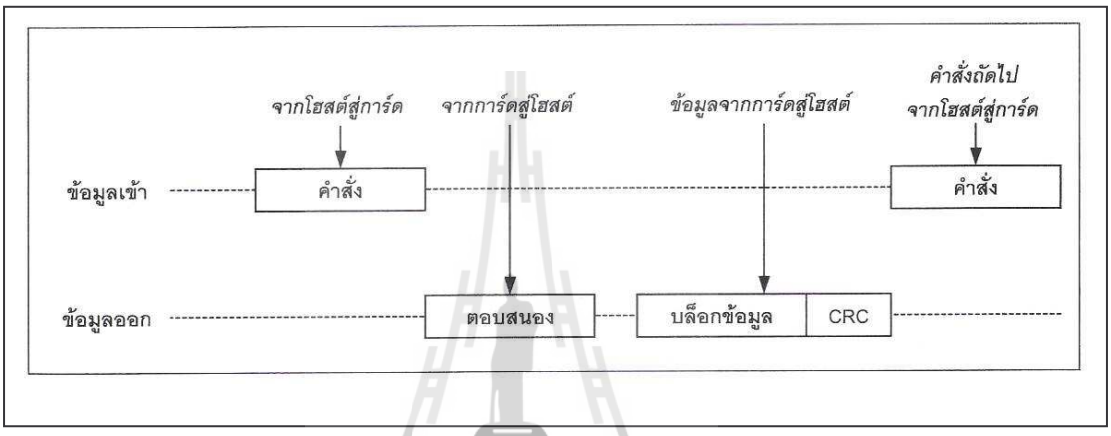
(2) ข้อมูลตอบสนองจะใช้ขนาด 8 หรือ 16 บิต

(3) เมื่อ SD การ์ดประสบปัญหาในการกู้คืนข้อมูล SD การ์ดจะแจ้งกลับด้วยข้อมูล ตอบสนองผิดพลาด (error response) แทนที่จะเป็นบล็อกข้อมูล โดยมีค่าเวลาไทม์เอาต์ที่มากกว่า การติดต่อผ่านบัส SD

มีการตอบสนองคำสั่งเมื่อทุกๆบล็อกข้อมูลถูกส่งไปยัง SD การ์ดในระหว่างการเขียน จะมี การตอบสนองด้วยสัญลักษณ์พิเศษ (special data response token) บล็อกของข้อมูลอาจมีขนาดใหญ่ เท่ากับ 1 บล็อกข้อมูลปกติ หรือเล็กเพียง 1 ไบต์ได้

การอ่านข้อมูลในโหมด SPI

การอ่านข้อมูลในโหมด SD การ์ดในโหมดการติดต่อแบบ SPI นี้ สามารถอ่านได้ทั้งแบบบล็อกเดี่ยวและหลายบล็อก คำสั่งที่ใช้คือ CMD17 สำหรับบล็อกเดี่ยว และ CMD18 สำหรับหลายบล็อก เมื่อ SD การ์ดได้รับคำสั่งร้องขอเพื่ออ่านข้อมูลแล้ว มันจะส่งรหัสตอบสนองต่อด้วยบล็อกข้อมูลที่มีความยาวตามที่กำหนดจากคำสั่ง CMD16 (SET_BLOCK_LENGTH) ปิดท้ายด้วยรหัส CRC ดังแสดงใน



รูปที่ 2.18 กระบวนการอ่านข้อมูลแบบบล็อกเดี่ยวจาก SD การ์ด

สำหรับรหัส CRC 16 บิตนั้นจะถูกกำหนดด้วยสมการโพลิโนเมียลตามมาตรฐาน CCITT ดังนี้

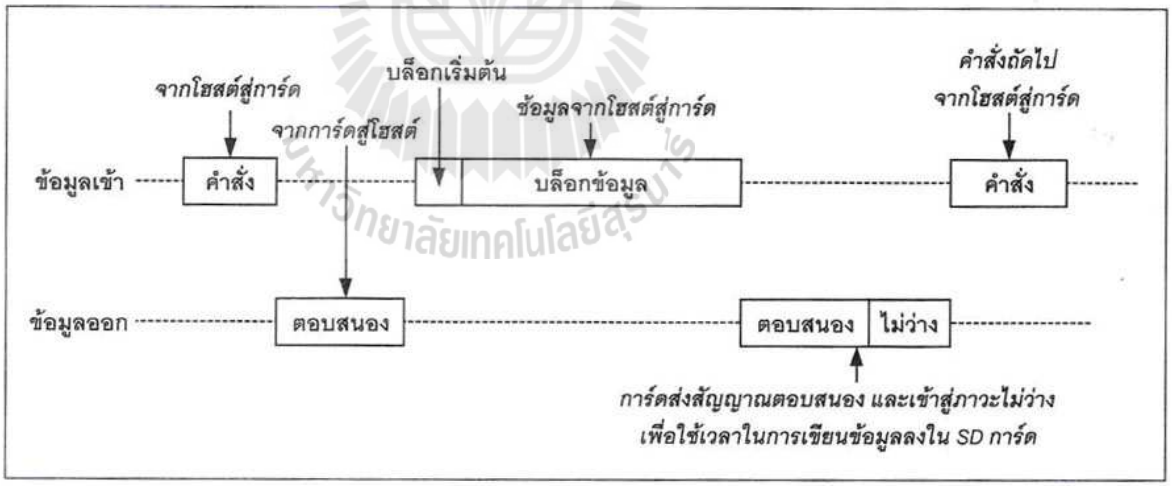
$$X^{16}+X^{12}+X^{5}+1$$

ความยาวของบล็อกข้อมูลสูงสุดคือ 512 ไบต์ กำหนดโดย REAL_BL_LEN หนึ่งพารามิเตอร์ของรีจิสเตอร์ CSD มีขอบเขตในการกำหนดค่าได้ตั้งแต่ 1 จนถึงค่าของ REAL_BL_LEN แต่โดยส่วนใหญ่แล้ว มักเลือกที่จะอ่านบล็อกข้อมูลความยาว 512 ไบต์ เพื่อความเร็วและต่อเนื่องในการทำงาน แอดเดรสเริ่มต้นของการอ่านสามารถกำหนดที่แอดเดรสใดๆ ก็ได้ภายในขอบเขตของ SD การ์ดใบนั้นๆที่ทำการอ่าน ในกรณีที่เกิดความผิดพลาดในกระบวนการอ่านข้อมูลขึ้น SD การ์ดจะไม่ส่งข้อมูลใดๆออกมาแต่จะส่งรหัสแจ้งความผิดพลาดกลับมายังโฮสต์แทน และยกเลิกกระบวนการติดต่อเพื่ออ่านข้อมูลในทันที ในกรณีที่มีการอ่านข้อมูล ในแต่ละบล็อกจะมีรหัส CRC 16 บิตปิดท้ายเสมอ เพื่อช่วยแยกข้อมูลให้ชัดเจน รวมทั้งช่วยในการตรวจสอบว่าข้อมูลที่ถูกอ่านออกไปถูกต้องสมบูรณ์ดังแสดงในรูปที่ **2.**18 และเมื่อต้องการหยุดอ่านข้อมูลต้องมีการส่งรหัสคำสั่งแจ้งแก่ SD การ์ดด้วย นั่นคือรหัสคำสั่ง CMD12 (Stop transmission command) หรือคำสั่งหยุดการส่งข้อมูลของ SD การ์ด

การเขียนข้อมูลในโหมด SPI

การเขียนข้อมูลไปยัง SD การ์ดในโหมดการติดต่อแบบ SPI นี้ สามารถเขียนได้ทั้งแบบบล็อกเดี่ยวและหลายบล็อก คำสั่งที่ใช้คือ CMD24 สำหรับบล็อกเดี่ยว และ CMD25 สำหรับบล็อก เมื่อ SD การ์ด ได้รับคำสั่งร้องขอเพื่อเขียนข้อมูลแล้ว มันจะส่งรหัสตอบสนอง จากนั้นจะรอบบล็อกข้อมูลจากโฮสต์ ความยาวของบล็อกข้อมูลในกรณีเขียนนี้ต้องใช้ 512 ไบต์ เพื่อช่วยลดความผิดพลาดในการเขียนข้อมูลในครั้งถัดไป ในรูปที่ 2.19 แสดงจังหวะการเขียนข้อมูลลงใน SD การ์ดแบบบล็อกเดี่ยว

ในทุกๆบล็อกข้อมูลที่จะนำมาเขียนลงใน SD การ์ดต้องเริ่มต้นด้วยบล็อกเริ่มต้น (start block) มีความยาว 1 ไบต์เมื่อข้อมูลถูกส่งออกมายัง SD การ์ดจะส่งสัญญาณตอบสนองตามด้วยสถานะไม่ว่าง (busy) เพื่อใช้เวลาไปตรวจสอบว่า ในขณะนั้นการ์ดยังมีพื้นที่เหลือพอสำหรับเขียนข้อมูลใหม่ลงไปหรือไม่ ถ้ามีพอก็จะเขียนข้อมูล และปรับปรุงความจุของการ์ดที่เหลือหลังจากเขียนข้อมูลใหม่แล้ว ในจังหวะที่เกิดสถานะไม่ว่างนั้น ที่สายสัญญาณข้อมูลออก (หรือข้อมูลเอาต์พุต) จะได้รับการกำหนดให้เป็นลอจิกต่ำจนกว่าจะเสร็จสิ้นกระบวนการสถานะของสายสัญญาณจะกลับมาเป็นลอจิกสูง



รูปที่ 2.19 จังหวะการเขียนข้อมูลลงใน SD การ์ดแบบบล็อกเดี่ยว

หลังจากที่การเขียนข้อมูลเสร็จสิ้นลง โฮสต์ควรตรวจสอบผลการทำงานด้วย โดยส่งรหัสคำสั่ง CMD13 (SEND_STATUS) ไปยัง SD การ์ดโดยจะเน้นการตรวจสอบรหัส CRC และบิตแจ้งเตือนความผิดพลาดจากการเขียนข้อมูลลงไปในหน่วยความจำ

2.4.8 บอร์ด ET-MINI SD/MMC

ET-MINI SD/MMC คือ ชุดอุปกรณ์สำหรับใช้ในการเชื่อมต่อกับอุปกรณ์หน่วยความ SD และMMC CARD เช่น การเชื่อมต่อระหว่างไมโครคอนโทรลเลอร์กับหน่วยความจำ Memory Card(SD/MMC) เป็นต้น ภายในชุดประกอบด้วย ช่อง (Socket) สำหรับใส่ CARD ประเภท SD และ MMC โดยจะมีการจัดขาสัญญาณออกมาที่Connector Pin เพื่อให้สามารถนำไปต่อใช้งานได้สะดวก นอกจากนั้นยังมีวงจรต่างๆ เช่น วงจรตรวจสอบสถานะการเสียบการ์ด (CARD DETECT) และวงจรพูลอัพ (Pull-Up) สัญญาณต่างๆ

รูปที่ 2.20 แสดงลักษณะของบอร์ด ET-MINI SD/MMC

คุณสมบัติบอร์ด ET-MINI SD/MMC

- รองรับการ์ดประเภท SD และ MMC
- สามารถเลือก ใช้ (Enable) หรือ ไม่ใช้ (Disable) วงจรพูลอัพ(Pull-Up)สัญญาณต่างๆได้
- สามารถแสดงสถานะการเสียบการ์ด (CARD DETECT) แสดงผลโดย LED และ ให้สัญญาณเอาพุตออกที่ขาสัญญาณ CD โดยมีคุณสมบัติดังนี้

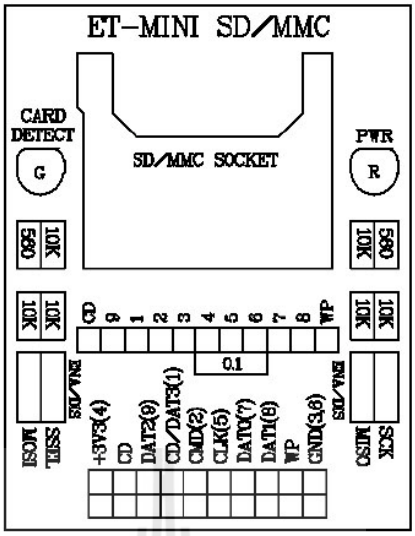
  CD = 1 คือ ไม่มี Card

  CD = 0 คือ มีการ์ด

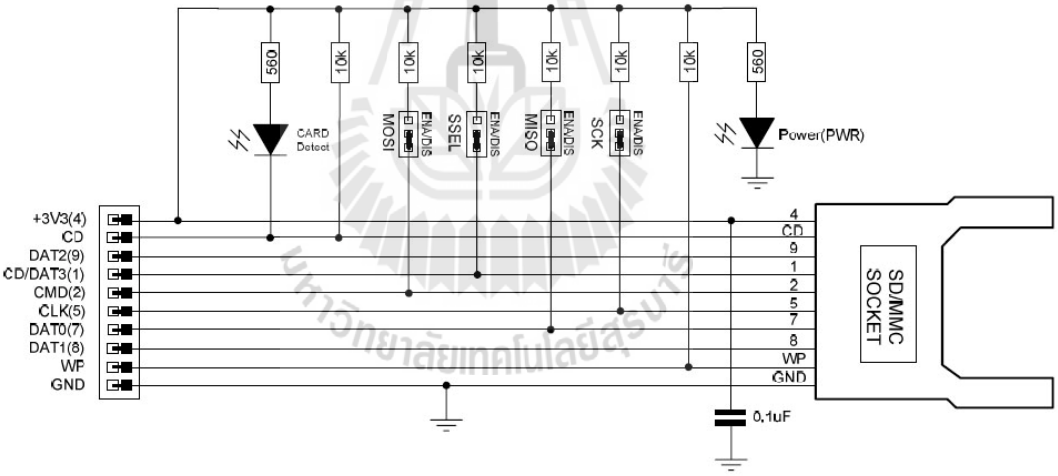
- สามารถแสดงสถานะของสวิตช์ Write Protection บน SD/MMC CARD ได้ โดยจะมีสัญญาณเอาต์พุตออกที่ขาสัญญาณ WP ดังนี้

  WP = 1 คือ ตำแหน่งของสวิตช์ Write Protection อยู่ที่ตำแหน่ง OFF

  WR = 0 คือ ตำแหน่งของสวิตช์ Write Protection อยู่ที่ตำแหน่ง ON



รูปที่ 2.21 ลักษณะของบอร์ด ET-MINI SD/MMC



รูปที่ 2.22 ลักษณะการเชื่อมต่อขาสัญญาณของ SD Card เข้ากับบอร์ด ET-MINI SD/MMC

รายละเอียดขาสัญญาณของ SD CARD

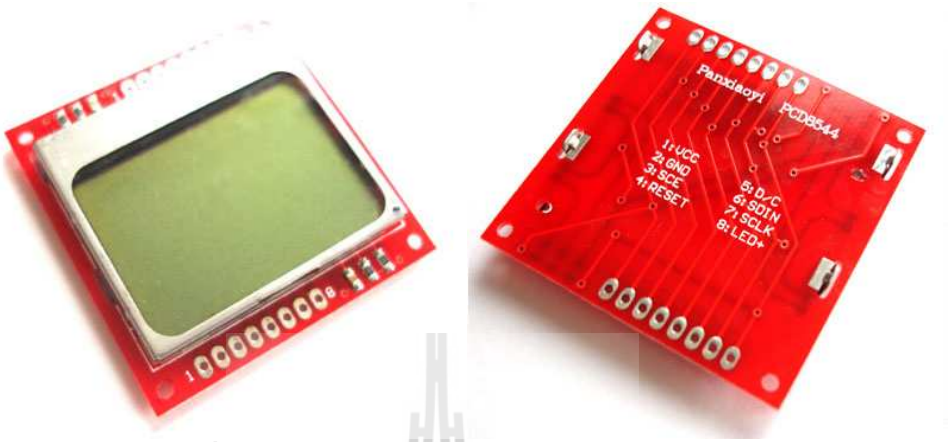
การเชื่อมต่อกับ SD CARD สามารถทำได้สองแบบ ก็คือ การเชื่อมต่อแบบ SD MODE และการเชื่อมต่อแบบ SPI MODE ดังรายละเอียดตามตารางต่อไปนี้(ภาคผนวก)

-ตารางที่ 2.4.5 แสดงสายสัญญาณของการติดต่อกับSD การ์ดทั้งแบบผ่านบัส SD และ SPI

-ตารางที่ 2.4.6 แสดงรายละเอียดขาสัญญาณต่างๆ เมื่อใช้การเชื่อมต่อในโหมด SD MODE

-ตารางที่ 2.4.7 แสดงรายละเอียดขาสัญญาณต่างๆ เมื่อใช้การเชื่อมต่อในโหมด SPI MODE

2.5 อุปกรณ์แสดงผลกราฟิก (LCD Nokia 5110G)



รูปที่ 2.23 ลักษณะของโมดูลกราฟิก LCD Nokia 5110G

2.5.1 การติดต่อกับโมดูลกราฟิก LCD

โมดูล LCD เป็นอุปกรณ์แสดงผลที่ได้รับความนิยมสูงในการนำมาประยุกต์ใช้งานร่วมกับไมโครคอนโทรลเลอร์เพื่อแสดงค่าตัวเลข ข้อความ รวมทั้งสัญลักษณ์และเครื่องหมายต่างๆจากแบบอักขระ (character) ก็ได้มีการพัฒนาเป็นแบบกราฟิก (graphic) เพื่อให้ผู้ใช้งานสามารถสร้างสรรงานแสดงผลได้อย่างกว้างขวางมากขึ้น เนื่องจากโมดูลกราฟิก LCD สามารถแสดงผลได้อิสระลงถึงระดับจุดหรือที่เรียกว่า พิกเซล (pixel) ผู้ใช้งานจึงสามารถนำภาพลงไปแสดงผลบนจอของกราฟิก LCD นี้ได้ โดยคุณภาพจะขึ้นกับความละเอียดของจอแสดงผล

โมดูลกราฟิก LCD ที่นำมาใช้ในการทดลองนี้เป็นโมดูลกราฟิก LCD ที่พัฒนาขึ้นมาเพื่อใช้ในเครื่องโทรศัพท์เคลื่อนที่ Nokia ในรุ่น 5110 ซึ่งในปัจจุบันสามารถจัดหาเพื่อนำมาใช้ในการแสดงผลของระบบไมโครคอนโทรลเลอร์ได้ง่ายขึ้น ทั้งยังมีราคาไม่สูงเมื่อเทียบกับประโยชน์ที่ได้ โมดูลกราฟิก LCD ที่นำมาใช้นี้มีชื่อว่า GLCD5110 ได้รับการประกอบให้พร้อมใช้งานจึงสะดวกอย่างยิ่งในการนำไปพัฒนาต่อนอกเหนือการทดลองเพื่อเรียนรู้

2.5.2 คุณสมบัติทางเทคนิค

- เป็นโมดูลแสดงผล LCD แบบกราฟิก เบอร์ LPH7366 ความละเอียด 84 x 84 จุด
- ใช้ตัวควบคุมเบอร์ PCD8544 ติดต่อแบบ SPI
- มี LED ส่องหลังเพื่อเพิ่มความชัดเจนในการแสดงผล สามารถควบคุมการติด-ดับได้
- ใช้ไฟเลี้ยงในย่าน +3 ถึง 3.3 v กระแสไฟฟ้าสูงสุด 10mA (เมื่อใช้ LED ส่องหลัง)
- สามารถแสดงผลทั้งตัวอักษร สัญลักษณ์ และรูปภาพโทนสีเดียว (สีดำ)
- ขนาด 4.5 x 4.5 เซนติดมตรติดตั้งใช้งานง่าย

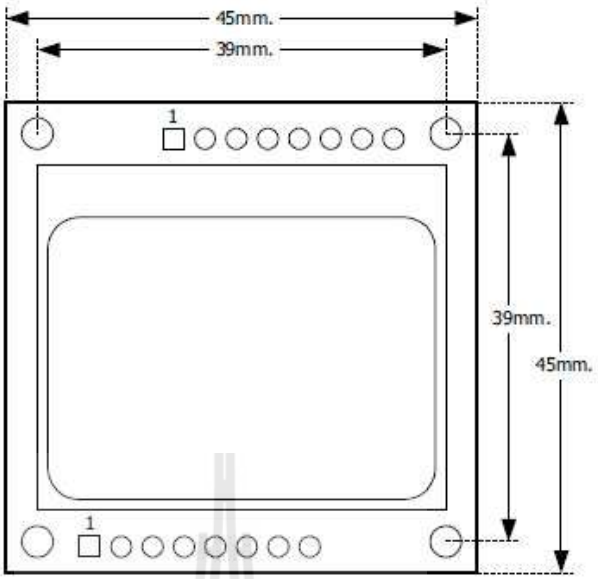
สามารถเชื่อมต่อกับไมโครคอนโทรลเลอร์สมัยใหม่ได้ทุกตระกูล

- หากใช้งานกับไมโครคอนโทรลเลอร์ที่ใช้แรงดันของระดับลอจิก 3v หรือ 3.3v สามารถเชื่อมต่อกันได้โดยตรง

- หากใช้งานกับไมโครคอนโทรลเลอร์ที่ใช้แรงดันของระดับลอจิกทีทีแอลหรือ 5v หรือสูงกว่าจะต้องต่วงจรเพื่อลดหรือปรับระดับแรงดันที่ขาเชื่อมต่อให้มีค่าไม่เกิน 3 ถึง 3.3v หรือไม่เกินไฟเลี้ยงของ GLCD5110

2.5.3 คำแนะนำในการเขียนโปรแกรมเพื่อติดต่อกับโมดูล GLCD5110

ในการใช้งานโมดูล GLCD5110 จะต้องเขียนโปรแกรมเพื่อติดต่อกับตัวควบคุมภายในโมดูลเบอร์ PCD8544 ซึ่งต้องศึกษาจาก Data Sheet สามารถดาวน์โหลด www.index.co.th โดยการติดต่อกับ PCD8544 เป็นแบบ SPI ซึ่งจะใช้สายสัญญาณข้อมูลอนุกรม 1 เส้น และสัญญาณนาฬิกา ส่วนขาควบคุมอื่นๆที่เหลือจะเป็นการส่งสัญญาณลอจิกไปควบคุมตรงๆ การเขียนโปรแกรมสามารถใช้กระทำได้ทั้งภาษาแอสเซมบลี ภาษาเบสิก (ขึ้นกับความสามารถของคอมไพเลอร์และไมโครคอนโทรลเลอร์) หรือภาษา C



ขาที่	ชื่อขา	หน้าที่
1	Vcc	ขาต่อไฟเลี้ยง +2.7 ถึง +3.3V
2	GND	ขาต่อกราวด์
3	SCE	ขาเอ็นเอเบิลการทำงานแอกติฟลอจิก "0"
4	RESET	ขารีเซต แอกติฟลอจิก "0"
5	D/$\overline{C}$	ขาอินพุตเลือกข้อมูล/คำสั่ง "0" - เลือกเขียนข้อมูลแสดงผล (data display) "1" - เลือกเขียนคำสั่ง (command)
6	SDIN	ขาอินพุตรับข้อมูลอนุกรม
7	SCLK	ขาอินพุตรับสัญญาณนาฬิกา
8	LED	ขาควบคุม LED ส่องหลังแอกติฟลอจิก "1" (มีตัวต้านทานจำกัดกระแส 33Ω ภายใน)

รูปที่ 2.24 ขนาดและข้อมูลขาต่อใช้งานของ GLCD5110 ในโมดุลกราฟิก LCD

2.5.4 การเชื่อมต่อทางฮาร์ดแวร์

SDIN ของ GLCD510 ต่อกับขา P0.06 ของไมโครคอนโทรลเลอร์ LPC2148

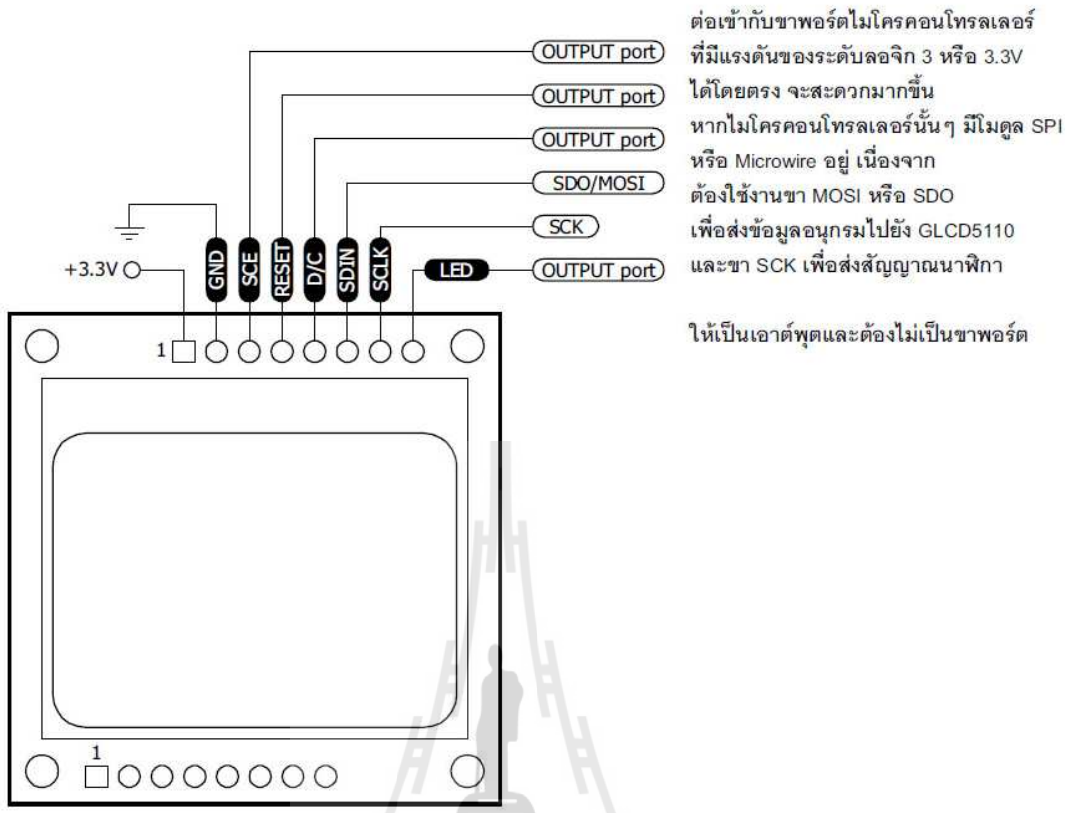
SCLK ของ GLCD510 ต่อกับขา P0.07 ของไมโครคอนโทรลเลอร์ LPC2148

SCE ของ GLCD510 ต่อกับขา P1.20 ของไมโครคอนโทรลเลอร์ LPC2148

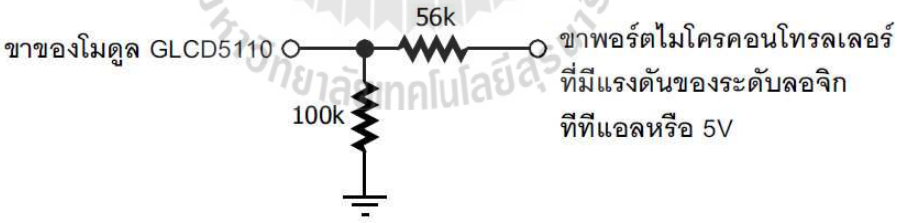
RESET ของ GLCD510 ต่อกับขา P0.04 ของไมโครคอนโทรลเลอร์ LPC2148

D/C ของ GLCD510 ต่อกับขา P0.05 ของไมโครคอนโทรลเลอร์ LPC2148

LED ของ GLCD510 ต่อกับขา P1.21 ของไมโครคอนโทรลเลอร์ LPC2148



รูปที่ 2.25 การเชื่อมต่อเพื่อใช้งาน GLCD5110 กับไมโครคอนโทรลเลอร์



รูปที่ 2.26 วงจรลดแรงดันเมื่อเชื่อมต่อกับ
ไมโครคอนโทรลเลอร์ที่ใช้แรงดันTTLหรือมีระดับ 5V

บทที่ 3

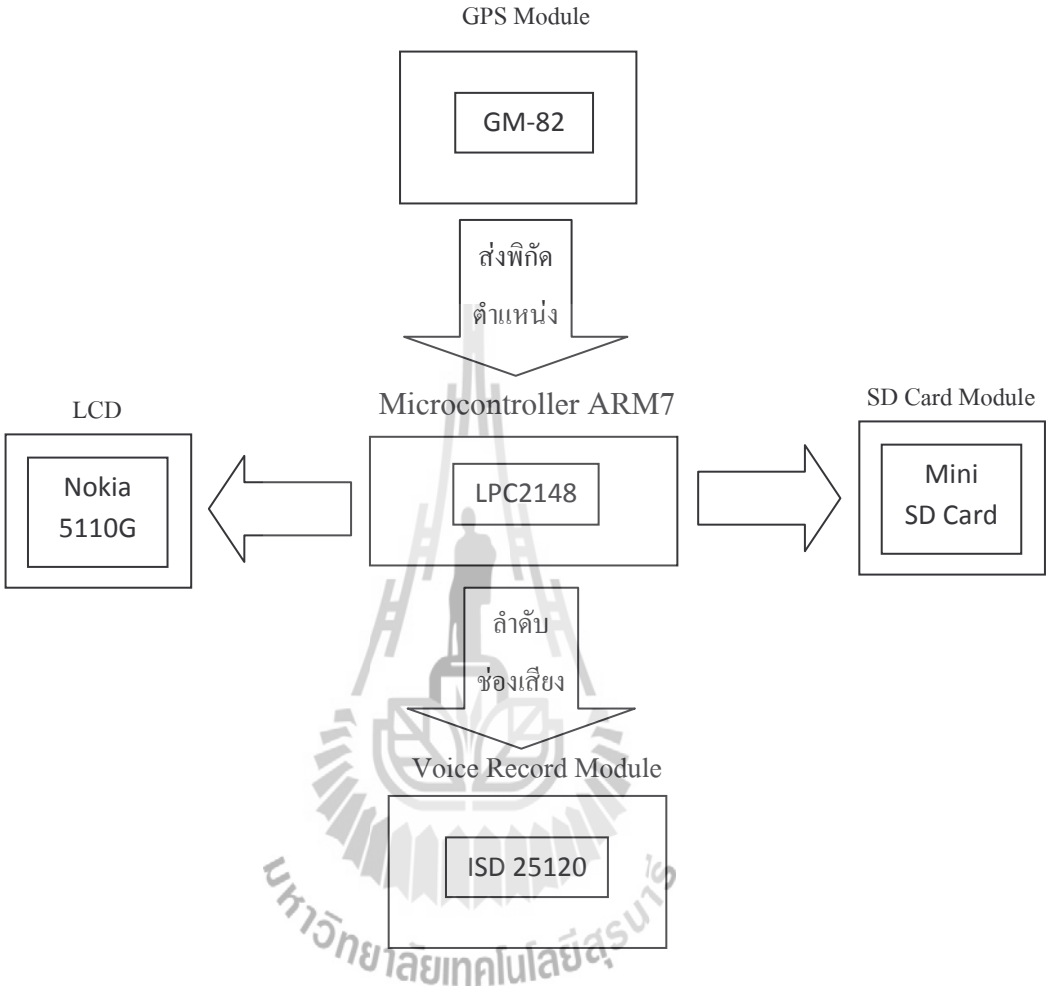
การออกแบบโครงงาน

โครงงานเครื่องบอกตำแหน่ง GPS แบบบันทึกเสียงได้มีการออกแบบเพื่อการบอกพิกัด-ตำแหน่งในรูปแบบเสียงซึ่งโครงงานประกอบด้วยด้วยเครื่องรับสัญญาณจีพีเอส(GPS Module) บอร์ดบันทึกและเล่น-เสียง ไมโครคอนโทรลเลอร์ ARM7 โดยไมโครคอนโทรลเลอร์จะนำค่าพิกัดตำแหน่งที่ได้จากเครื่องรับสัญญาณจีพีเอส (GPS Module) มาประมวลผลเพื่อหาระยะทางบนพื้นโลก ซึ่งคำนวณจากพิกัดตำแหน่งปัจจุบันเปรียบเทียบกับพิกัดตำแหน่งอ้างอิงที่บันทึก หลังจากนั้นจะนำข้อมูลเสียงจากบอร์ดบันทึกและเล่น-เสียงไปรายงานผล และนอกจากนี้เครื่องบอกพิกัดตำแหน่งสามารถบันทึกพิกัดตำแหน่งและบันทึกเสียง ณ ตำแหน่งใด ๆ เพื่อใช้เป็นตำแหน่งอ้างอิงและข้อมูลเสียงเพื่อแสดงผล โดยโครงงานนี้ประกอบด้วยด้วยส่วนของการทำงานต่าง ๆ ดังนี้

1. การรับพิกัดตำแหน่ง
2. การบันทึกและเล่นเสียง
3. การบันทึกพิกัดตำแหน่ง
4. การแสดงผล

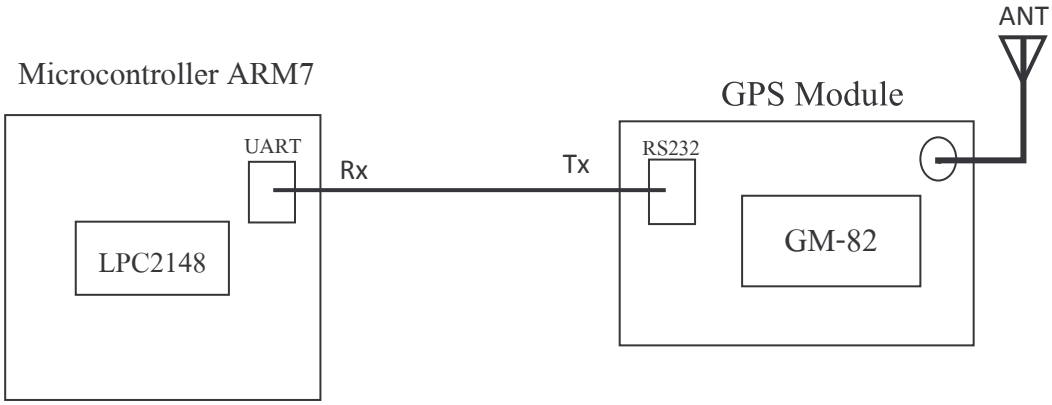
ซึ่งในแต่ละส่วนจะมีไมโครคอนโทรลเลอร์เป็นตัวควบคุมการทำงานหลักเพื่อให้สามารถทำงานได้ตามต้องการ ดังนั้นเพื่อให้การทำงานในแต่ละส่วนเป็นไปอย่างสมบูรณ์จึงได้ทำการออกแบบส่วนการทำงานในแต่ละส่วนดังนี้

3.1 การออกแบบฮาร์ดแวร์



รูปที่3.1 องค์ประกอบของโครงงาน

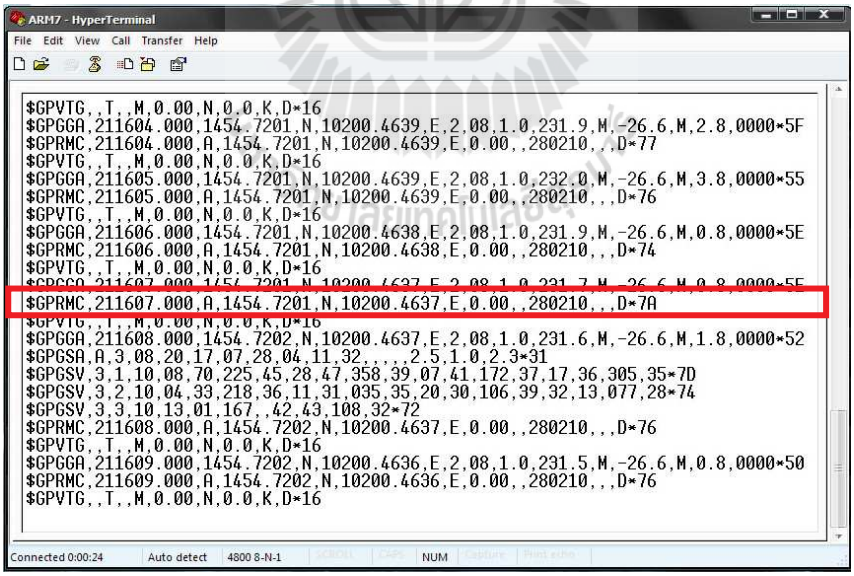
ส่วนการรับพิกัดตำแหน่ง



รูปที่ 3.2 ส่วนของการรับค่า GPS

ส่วนของการรับค่าพิกัดตำแหน่งนั้น เราสามารถรับค่าพิกัดตำแหน่งจากเครื่องรับสัญญาณจีพีเอส (GPS Module) ซึ่งจะต้องทำการเชื่อมต่อสายอากาศเข้ากับเครื่องรับสัญญาณจีพีเอส เพื่อที่จะสามารถค้นหาสัญญาณดาวเทียมได้

การรับค่าพิกัดตำแหน่งนั้นไมโครคอนโทรลเลอร์จะเป็นตัวรับค่าพิกัดตำแหน่งจากเครื่องรับ-สัญญาณจีพีเอส (GPS Module) โดยการเชื่อมต่อพอร์ตสื่อสารอนุกรม (RS232) ของ GPS Module (ขา Tx) เข้ากับพอร์ต UART ของบอร์ดไมโครคอนโทรลเลอร์ (ขา Rx) ดังรูปที่ 3.2



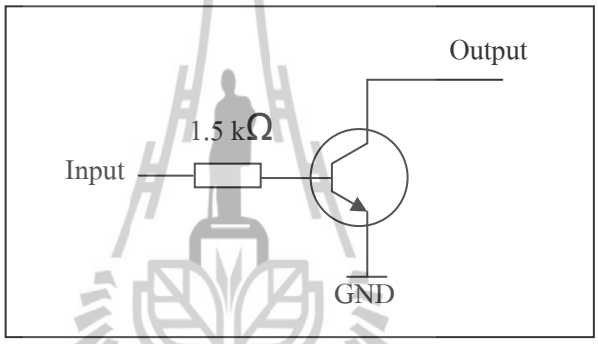
รูปที่ 3.3 รูปแบบประโยคที่ส่งออกมาจาก GPS Module

การรับค่าพิกัดตำแหน่งจาก GPS Module นั้นในโครงงานนี้จะรับในรูปแบบ $GPRMC ซึ่งจะทำการเก็บค่าละติจูด(Latitude) และลองจิจูด(Longitude) เพื่อนำมาให้กับไมโครคอนโทรลเลอร์ประมวลผล

ส่วนการบันทึกและเล่นเสียง

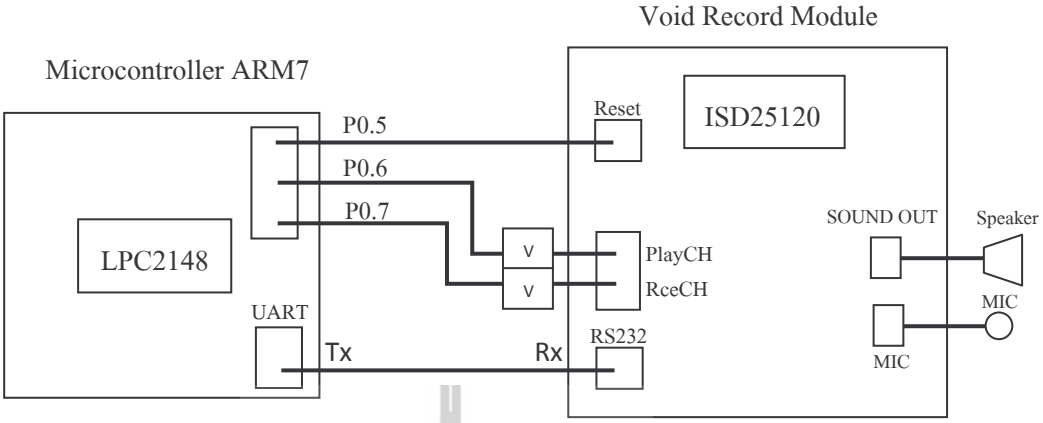
ส่วนของการบันทึกและเล่นเสียงนั้น จะเป็นส่วนของการบันทึกเสียงเก็บไว้ในหน่วยความจำของ IC เบอร์ ISD25120 และยังสามารถเล่นเสียงที่ทำการบันทึกได้

เนื่องจากบอร์ดบันทึกและเล่นเสียงนั้นใช้ระดับแรงดันไฟฟ้าไม่เท่ากับบอร์ดไมโครคอนโทรลเลอร์ ARM7 ซึ่งต้องทำการจ่ายพลังงานไฟฟ้าให้กับบอร์ดบันทึกและเล่นเสียงโดยใช้แรงดัน 9 V เพื่อให้สามารถทำงานได้ และภายในบอร์ดนั้นจะใช้แรงดันของลอจิกระดับ TTL หรือ 5 V นั้นคือ Logic เป็น 1 จะมีระดับแรงดัน 5 V และ Logic เป็น 0 จะมีระดับแรงดัน 0 V ดังนั้นการเชื่อมต่อระหว่างไมโครคอนโทรลเลอร์กับบอร์ดบันทึกเสียงและเล่นเสียงนั้น จะต้องใช้วงจรแปลงระดับแรงดันในการเชื่อมต่อ



รูปที่ 3.4 วงจรแปลงระดับแรงดัน

เนื่องจากไมโครคอนโทรลเลอร์เป็นตัวควบคุมบอร์ดบันทึกและเล่นเสียงให้สามารถทำงานตามคำสั่งได้ และนอกจากนี้บอร์ดบันทึกและเล่นเสียงมีการตอบสนองที่แรงดันต่ำ (Active Low) ดังนั้นเมื่อ ไมโครคอนโทรลเลอร์สั่งงานจึงต้องเชื่อมต่อผ่านทรานซิสเตอร์ ดังรูปที่ 3.4 ซึ่งจะเห็นว่าเมื่อพอร์ตของไมโครคอนโทรลเลอร์เชื่อมต่อกับ Input ของวงจรมีสัญญาณ Logic 1 จะทำให้ทรานซิสเตอร์มีการทำงานจึงทำให้พอร์ตของบอร์ดบันทึกและเล่นเสียงที่เชื่อมต่อกับ Output ของวงจรมีระดับสัญญาณเป็น GND หรือ Logic 0 และเมื่อพอร์ตของไมโครคอนโทรลเลอร์เชื่อมต่อกับ Input ของวงจรมีสัญญาณ Logic 0 จะทำให้ทรานซิสเตอร์ไม่ทำงาน ซึ่งจะทำให้ Output ของวงจรไม่ได้เชื่อมสัญญาณลงกราวด์ ดังนั้นพอร์ตของบอร์ดบันทึกและเล่นเสียงที่เชื่อมต่อกับ Output ของวงจรมีระดับสัญญาณเป็น 5 V หรือเป็น Logic 1 เช่นเดิม



รูปที่ 3.5 ส่วนการบันทึกและเล่นเสียง

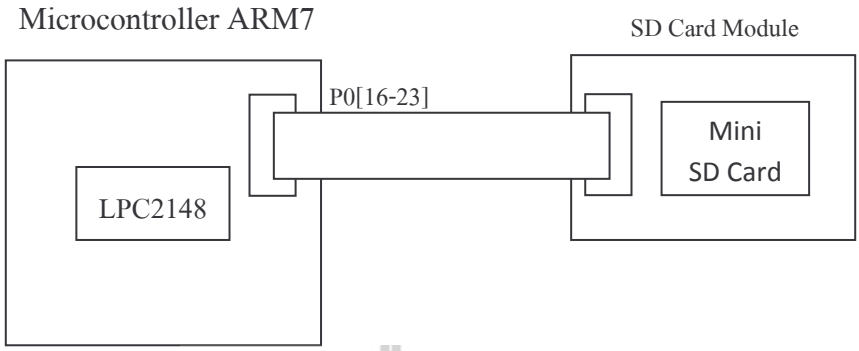
***** หมายเหตุ** สัญลักษณ์ [V] คือ การเชื่อมต่อผ่านวงจรแปลงแรงดัน

จากรูปที่ 3.5 จะเห็นว่าสามารถเชื่อมต่อไมโครคอนโทรลเลอร์เข้ากับบอร์ดบันทึกและเล่นเสียงได้ดังนี้

ไมโครคอนโทรลเลอร์ ARM7		:	บอร์ดบันทึกและเล่นเสียง
P0.5	เชื่อมต่อกับ		Reset
P0.6	เชื่อมต่อกับ(ผ่านวงจรแปลงระดับ)		PlayCH
P0.7	เชื่อมต่อกับ(ผ่านวงจรแปลงระดับ)		RceCH
Tx	เชื่อมต่อกับ		Rx

จะเห็นได้ว่า การเชื่อมต่อ P0.5 กับขา Reset นั้น ไม่ต้องผ่านวงจรแปลงระดับแรงดัน เพราะขา Reset สถานะปกติจะเป็น GND ดังนั้นเมื่อต้องการทำการรีเซตบอร์ดบันทึกและเล่นเสียงจะต้องให้ขา Reset มีระดับแรงดับเกิน 2.5 V โดยการสั่ง Logic 1 จากไมโครคอนโทรลเลอร์โดยตรง ดังนั้นการเชื่อมต่อจึงไม่ต้องผ่านวงจรแปลงแรงดัน

ส่วนการบันทึกพิกัดตำแหน่ง

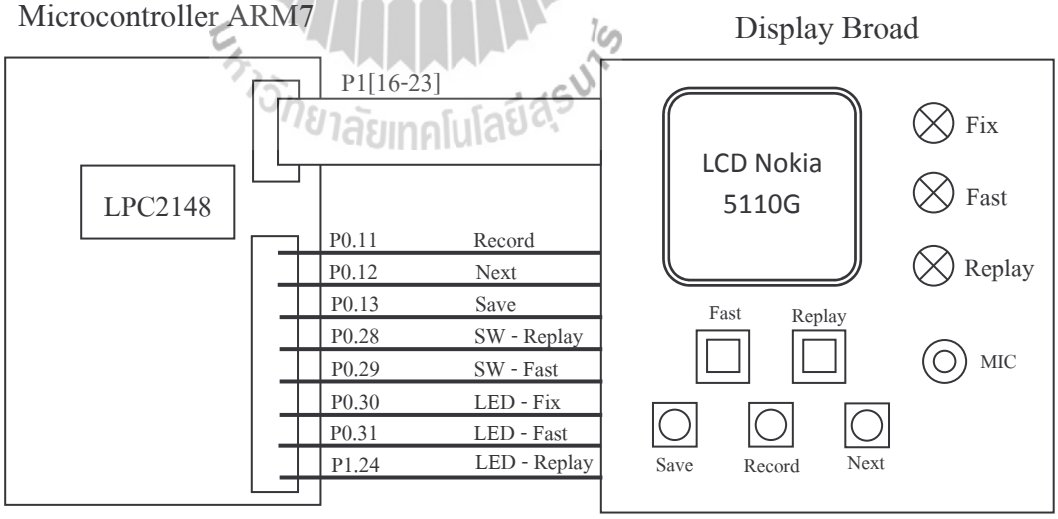


รูปที่ 3.6 ส่วนการบันทึกพิกัดตำแหน่ง

การเชื่อมต่อบอร์ด SD Card Module กับ ไมโครคอนโทรลเลอร์ นั้นสามารถทำได้โดย

ไมโครคอนโทรลเลอร์ ARM7	:	บอร์ด SD Card Module
P0.20	เชื่อมต่อกับ	CD/SATA
P0.19	เชื่อมต่อกับ	DI(CDM)
P0.18	เชื่อมต่อกับ	DO(DATA)
P0.17	เชื่อมต่อกับ	CLK

ส่วนการแสดงผล



รูปที่ 3.7 ส่วนการแสดงผล

Display Broad เป็นบอร์ดที่ประกอบด้วย สวิตซ์ , LED และจอแสดงผล LCD เพื่อใช้ในการควบคุมการทำงานและแสดงผลการทำงานของเครื่องบอกพิกัดตำแหน่ง ซึ่งจำเป็นต้องการจ่าย

พลังงานไฟฟ้าให้กับ Display Broad โดยใช้แรงดัน 3.3 V และการเชื่อมต่อ Display Broad กับไมโครคอนโทรลเลอร์ นั้นสามารถทำได้ดังรูปที่ 3.7 ดังนี้

ไมโครคอนโทรลเลอร์ ARM7 : บอร์ดบันทึกและเล่นเสียง

P0.11	เชื่อมต่อกับ	สวิตซ์	Record
P0.12	เชื่อมต่อกับ	สวิตซ์	Next
P0.13	เชื่อมต่อกับ	สวิตซ์	Save
P0.28	เชื่อมต่อกับ	สวิตซ์	Replay
P0.29	เชื่อมต่อกับ	สวิตซ์	Fast
P0.30	เชื่อมต่อกับ	LED	Fix
P0.31	เชื่อมต่อกับ	LED	Fast
P1.24	เชื่อมต่อกับ	LED	Replay

การเชื่อมต่อ LCD Nokia 5110 G กับ ไมโครคอนโทรลเลอร์ สามารถทำได้ดังนี้

P1.18	เชื่อมต่อกับ	SCE
P1.19	เชื่อมต่อกับ	RESET
P1.20	เชื่อมต่อกับ	D/C
P1.21	เชื่อมต่อกับ	SDIN
P1.22	เชื่อมต่อกับ	SCK
P1.23	เชื่อมต่อกับ	LED

3.2 การออกแบบซอฟแวร์

ไมโครคอนโทรลเลอร์ ARM7 จะเป็นตัวควบคุมการทำงานหลัก โดยจะต้องมีซอฟต์แวร์ที่ใช้ในการเขียนคำสั่งควบคุม ดังนั้นในโครงงานนี้ ผู้จัดทำเลือกใช้ภาษาซีโดยได้ใช้โปรแกรม Keil uVision3 ในการเขียนคำสั่งควบคุม เนื่องจากภาษาซีเป็นภาษาโครงสร้างง่ายต่อการทำความเข้าใจและสามารถปรับปรุงพัฒนาต่อได้ง่าย นอกจากนั้นภาษาซียังเป็นภาษามาตรฐานไม่ขึ้นกับฮาร์ดแวร์ (ไมโครคอนโทรลเลอร์) มีความยืดหยุ่นในการใช้งานกับไมโครคอนโทรลเลอร์ตระกูลอื่นได้ง่าย

3.3 การทำงานของโปรแกรม

เนื่องจากการคำนวณหาพิกัดตำแหน่งบนพื้นโลกโดยเครื่องรับสัญญาณจีพีเอส (GPS Module) ใช้หลักการของการคำนวณระยะทางจากเครื่องรับสัญญาณจีพีเอสกับดาวเทียม ซึ่งในการเชื่อมต่อสัญญาณนั้นจะทำการเชื่อมต่อกับดาวเทียมอย่างน้อย 3 ดวง เพื่อที่จะสามารถระบุพิกัดตำแหน่งบนพื้นโลกได้ การคำนวณระยะทางจะคำนวณโดยใช้ระยะเวลาที่ใช้ในการส่งสัญญาณเทียบกับเวลาจริง และจะใช้เวลาที่ได้คำนวณหาระยะทางจากเครื่องรับสัญญาณจีพีเอสกับดาวเทียม การส่งสัญญาณระหว่างเครื่องรับสัญญาณ-จีพีเอสและดาวเทียมในแต่ละครั้งนั้นอาจจะใช้เวลาที่ไม่เท่ากัน ดังนั้นในการคำนวณพิกัดตำแหน่งบนพื้นโลกนั้นจึงมีค่าคลาดเคลื่อนไป

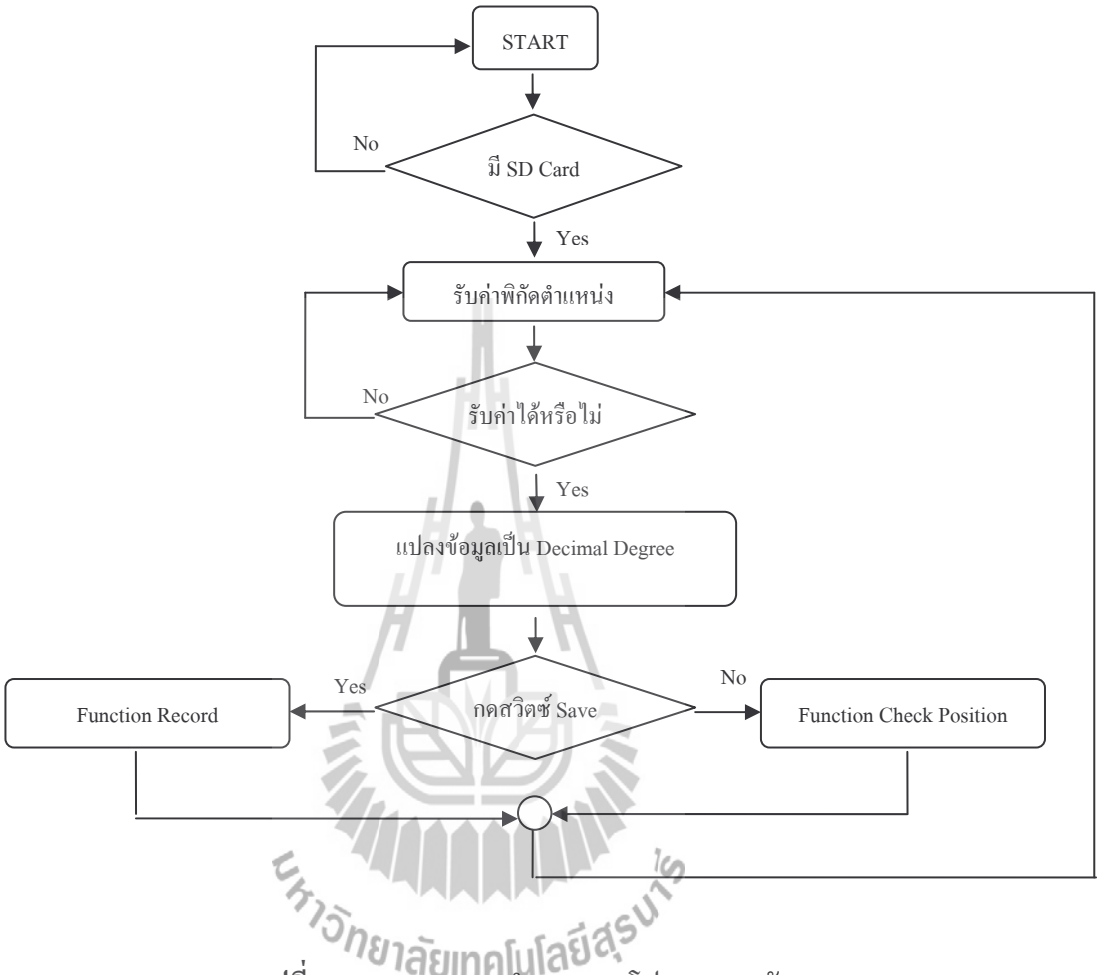
ในการออกแบบโครงงานนี้เพื่อลดความคลาดเคลื่อนที่เกิดขึ้น จึงได้มีการเฉลี่ยค่าพิกัดตำแหน่งที่ได้จากเครื่องรับสัญญาณจีพีเอสในแต่ละครั้ง และเมื่อนำค่าพิกัดตำแหน่งที่ได้มาทำการเฉลี่ยหลาย ๆ ครั้งจะได้ค่าพิกัดตำแหน่งที่ถูกต้องมากขึ้น แต่ในการรับค่าจากเครื่องรับสัญญาณจีพีเอสหลาย ๆ ครั้งนั้นจะทำให้ใช้เวลาในการรับค่ามากขึ้น ดังนั้นจึงต้องทำการเฉลี่ยด้วยค่าที่เหมาะสมตามลักษณะของการใช้งาน ดังนั้นในโครงงานนี้จึงได้ทำการเฉลี่ยค่าพิกัดตำแหน่งโดยแบ่งการทำงานเป็น 2 โหมดการทำงานดังนี้

1. โหมด **Fast** คือ การรับค่าพิกัดตำแหน่ง 3 ครั้งแล้วทำการเฉลี่ย และตรวจสอบพื้นที่ ที่มีรัศมี 50 เมตร จากการเฉลี่ยค่าพิกัดตำแหน่ง 3 ครั้งนั้น สามารถลดค่าความคลาดเคลื่อนที่เกิดได้เพียงเล็กน้อยเท่านั้น ดังนั้นในการตรวจสอบพื้นที่ของพิกัดตำแหน่งจึงต้องตรวจสอบเป็นพื้นที่ ที่มีรัศมีกว้าง ในโครงงานนี้ตรวจสอบที่รัศมี 50 เมตรและเหมาะสมกับการใช้งานที่ต้องใช้ความเร็วในการเคลื่อนที่โดยประมาณไม่เกิน 60 กิโลเมตร/ชั่วโมง

2. โหมด **Slow** คือ การรับค่าพิกัดตำแหน่ง 5 ครั้งแล้วทำการเฉลี่ย และตรวจสอบพื้นที่ ที่มีรัศมี 20 เมตร จากการเฉลี่ยค่าพิกัดตำแหน่ง 5 ครั้งจะได้พิกัดตำแหน่งที่ถูกต้องมากขึ้น จึงสามารถตรวจสอบพื้นที่ ที่มีรัศมีแคบได้ แต่มีข้อเสียคือระยะเวลาที่ใช้ในการประมวลผลมากขึ้นจึงเหมาะสมกับการใช้งานกับความเร็วต่ำ ๆ เพื่อไม่ให้เกิดความผิดพลาดของการคำนวณระยะทางเมื่อมีการเคลื่อนที่มากเกินไป

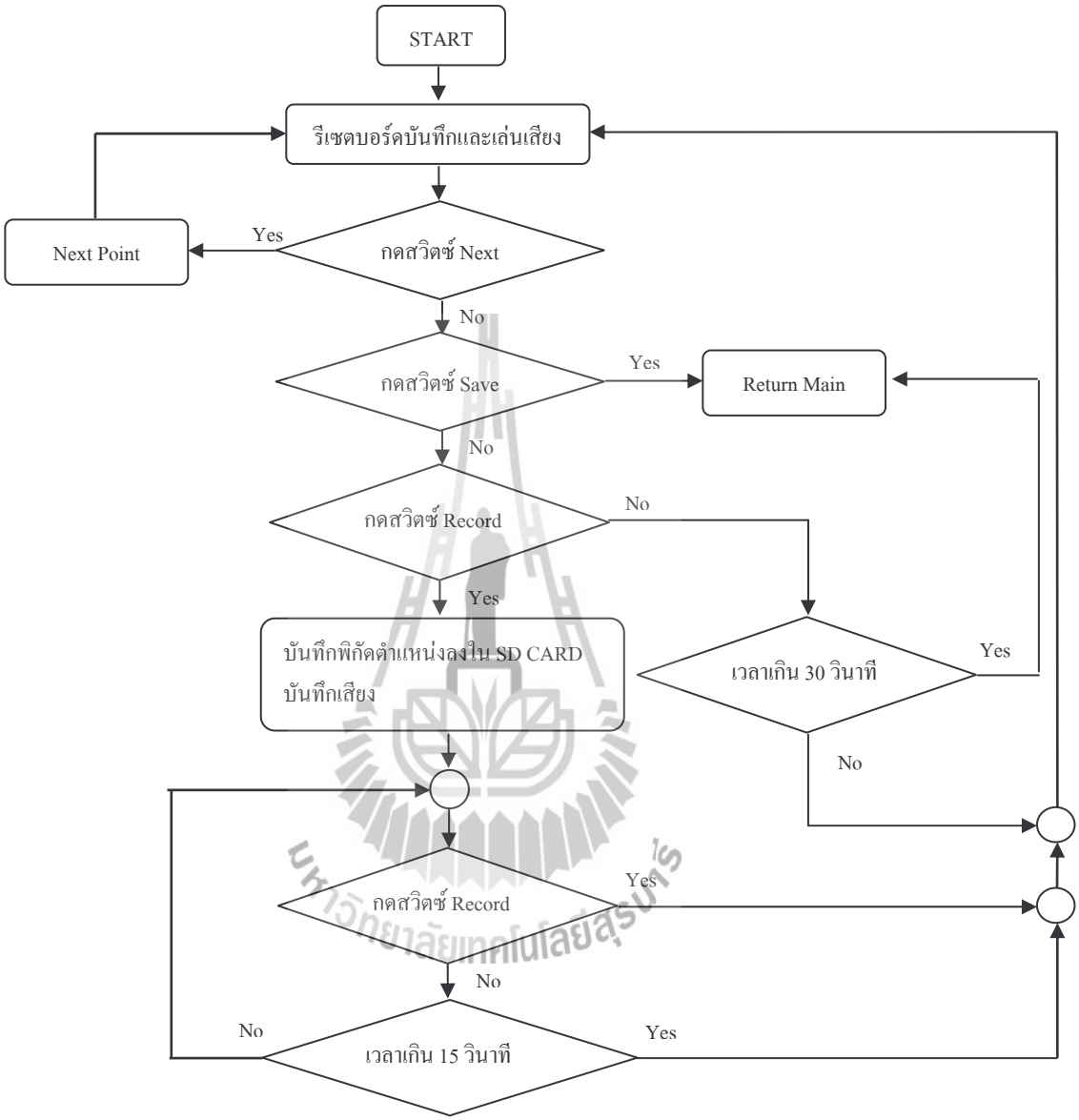
นอกจากโหมดการทำงาน 2 โหมดดังที่กล่าวแล้วนั้น โครงงานนี้ยังสามารถเลือกการเล่นเสียงเป็นแบบเล่นซ้ำตลอดเวลาเมื่ออยู่ในพื้นที่พิกัดตำแหน่งอ้างอิง (Replay) โดยการออกระบบของโครงงานจะประกอบด้วย 3 ส่วนหลัก คือ 1. ส่วนของการประมวลผลหลัก จะทำหน้าที่ในการรับค่าพิกัดตำแหน่ง และเป็นตัวควบคุมส่วนการทำงานอื่น ๆ 2. ส่วนของการบันทึกเสียง จะทำหน้าที่ในการบันทึกพิกัดตำแหน่งอ้างอิงและทำการบันทึกเสียง 3. ส่วนของการตรวจสอบพิกัดตำแหน่งและเล่นสียง ซึ่งจะรับค่าพิกัดตำแหน่งจากส่วนการประมวลผลหลักเพื่อคำนวณหาระยะทางของพิกัดตำแหน่งปัจจุบันและพิกัดตำแหน่งอ้างอิง แล้วทำเล่นเสียงเมื่อการตรวจสอบพบว่าอยู่ในพื้นที่ของพิกัดตำแหน่งอ้างอิง โดยจะมีแผนผังการทำงานของส่วนต่าง ๆ ดังนี้

การทำงานของโปรแกรมหลัก (Main)



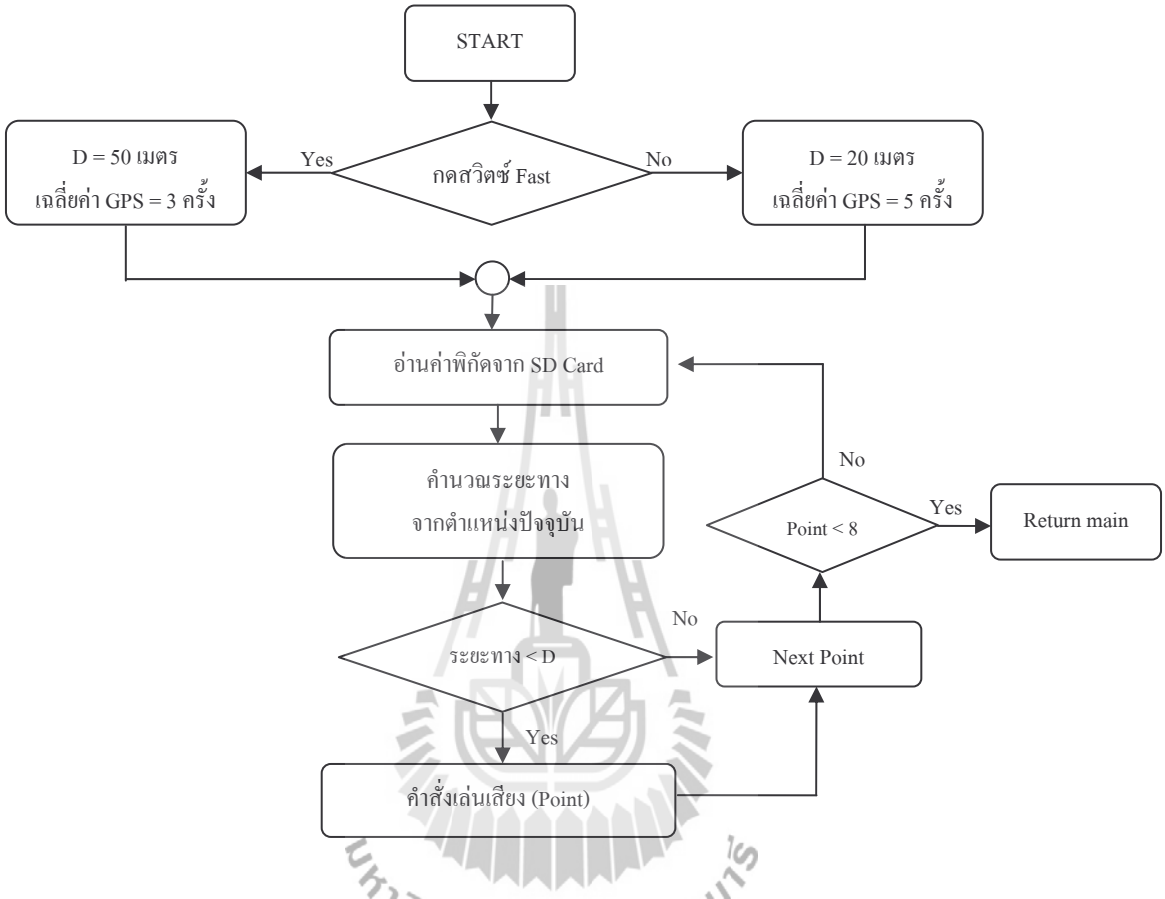
รูปที่ 3.8 แผนภาพการทำงานของโปรแกรมหลัก

การทำงานของส่วนการบันทึกเสียง (Function Record)



รูปที่ 3.9 แผนภาพการทำงานของส่วนบันทึกเสียง

การทำงานของส่วนตรวจสอบพิกัดตำแหน่งและเล่นเสียง (Function Check Position)



รูปที่ 3.10 แผนภาพการทำงานของส่วนตรวจสอบพิกัดตำแหน่งและเล่นเสียง

3.4 อธิบายการทำงานของโครงงาน

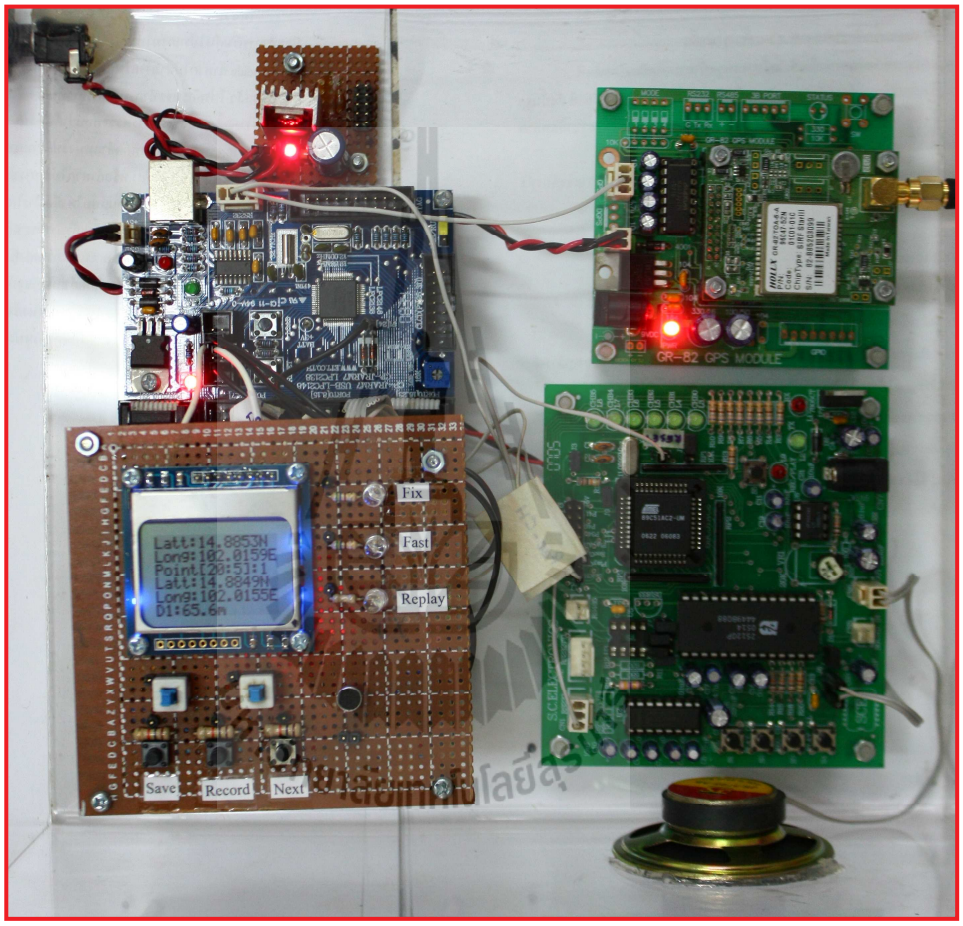
ในส่วนประมวลผลหลักจะทำการตรวจสอบว่ามีการ์ดหน่วยความจำ(SD-Card) หรือไม่ ถ้าหากไม่มีจะทำการเริ่มต้นโปรแกรมใหม่จนกว่าจะมีการ์ดหน่วยความจำ และจากนั้นจะรับข้อมูลพิกัดตำแหน่งจาก GPS Module ถ้าหากไม่สามารถรับสัญญาณได้จะรอจนกว่าจะมีสัญญาณ GPS ส่งมา เมื่อได้ข้อมูลพิกัด-ตำแหน่งแล้ว โปรแกรมจะทำการแปลงพิกัดตำแหน่งเป็นหน่วยในระบบพิกัดแบบค่าตัวเลขทศนิยม (DD : Decimal Degree) แล้วจะทำการตรวจสอบว่าต้องการบันทึกพิกัดตำแหน่งหรือไม่ ถ้าหากต้องการบันทึกส่วนประมวลผลหลักจะส่งค่าพิกัดตำแหน่งไปยังส่วนการการบันทึกเสียงเพื่อทำการบันทึกพิกัดตำแหน่งอ้างอิงและบันทึกเสียง หรือถ้าหากไม่ต้องการบันทึกพิกัดตำแหน่งจะทำการตรวจสอบว่าอยู่ในพื้นที่ของพิกัดตำแหน่งอ้างหรือไม่ โดยจะทำงานในส่วนการตรวจสอบพิกัดตำแหน่งและเล่นเสียง ซึ่งในส่วนนี้จะทำการตรวจสอบว่าอยู่ในโหมดการทำงานแบบใด เมื่ออยู่ในโหมด Fast จะระบุระยะทางในการตรวจสอบ 50 เมตรและการรับค่าพิกัดตำแหน่ง GPS 3 ครั้งแล้วมาทำการเฉลี่ยค่าพิกัดตำแหน่ง แต่ถ้าหากอยู่ในโหมด Slow จะระบุระยะทางในการตรวจสอบ 20 เมตรและการรับค่าพิกัดตำแหน่ง GPS 5 ครั้งแล้วมาทำการเฉลี่ยค่าพิกัดตำแหน่ง จากนั้นจะทำการคำนวณหาระยะทางของพิกัดตำแหน่งปัจจุบันและพิกัดตำแหน่งอ้างอิง โดยจะอ่านข้อมูลที่บันทึกจากหน่วยความจำ SD Card เมื่อคำนวณระยะทางพบว่าอยู่ในพื้นที่พิกัดตำแหน่งอ้างอิง จะทำการแสดงผลในรูปแบบเสียงตามที่ได้บันทึกเอาไว้

บทที่ 4

การใช้งานโครงงาน

4.1 การใช้งานเครื่องบอกพิกัดตำแหน่ง GPS แบบบันทึกเสียง

โครงงานเครื่องบอกพิกัดตำแหน่ง GPS แบบบันทึกเสียงมีลักษณะต่าง ๆ ดังรูปที่ 4.1

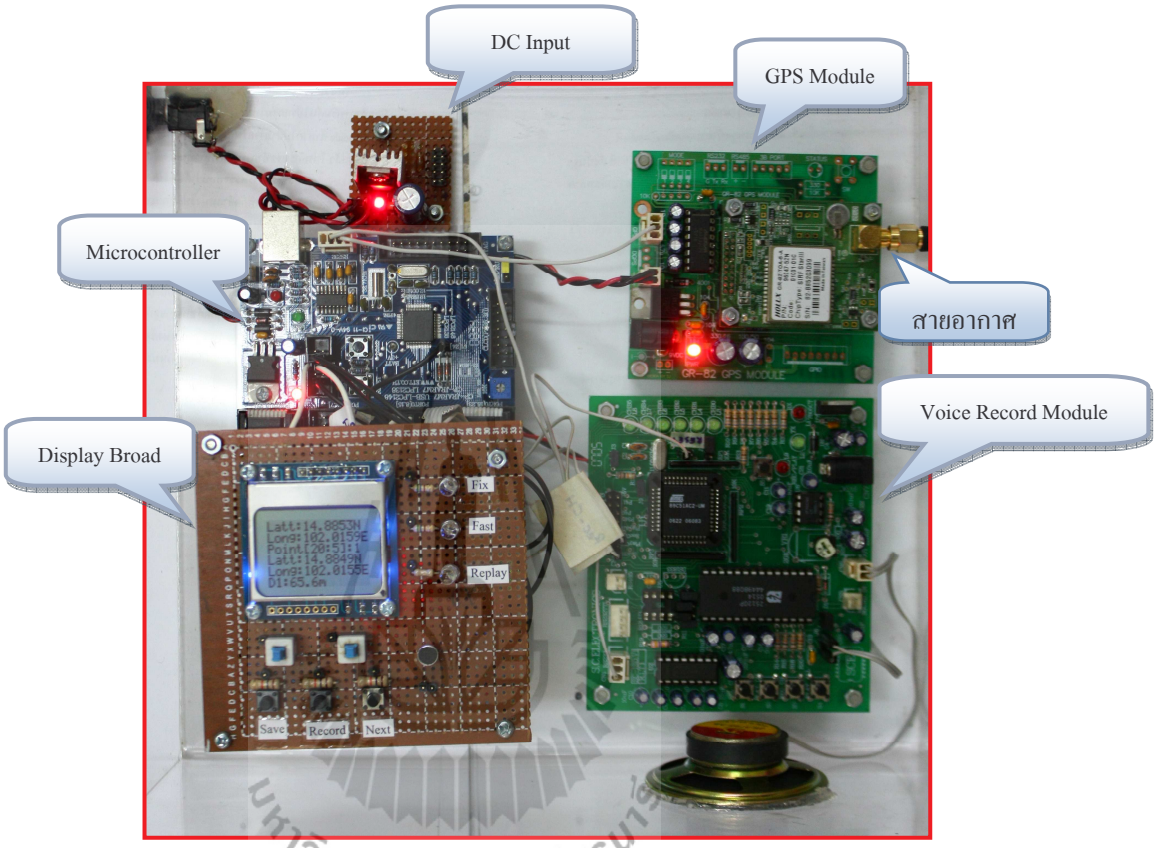


รูปที่ 4.1 ลักษณะของเครื่องบอกพิกัดตำแหน่ง GPS แบบบันทึกเสียง

เครื่องบอกพิกัด GPS แบบบันทึกเสียงนั้นมีไมโครคอนโทรลเลอร์ ARM7 เป็นตัวควบคุมการทำงาน โดยรับค่าพิกัดตำแหน่งจากเครื่องรับสัญญาณจีพีเอส (GPS Module) เพื่อนำมาประมวลผลหาระยะทางของพิกัดตำแหน่งปัจจุบันกับพิกัดตำแหน่งอ้างอิงที่ได้ทำการบันทึก เมื่อไมโครคอนโทรลเลอร์ประมวลผลหาระยะทางแล้วพบว่าอยู่ในพื้นที่ของพิกัดตำแหน่งตำแหน่งอ้างอิง จะมีการรายงานผลในรูปแบบของเสียงตามที่ได้มีการบันทึกเอาไว้

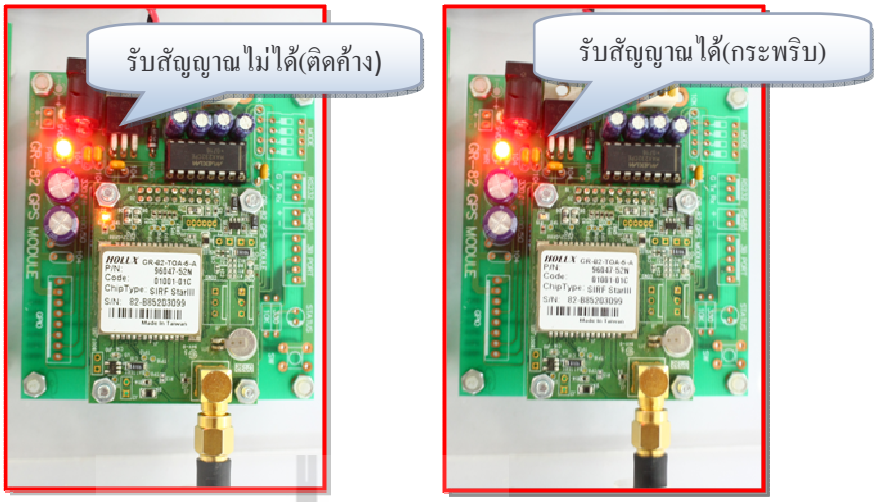
4.1.1 การเริ่มต้นใช้งานเครื่องบอกพิกัดตำแหน่ง GPS แบบบันทึกเสียง

1. เชื่อมต่อสายอากาศของเครื่องรับสัญญาณจีพีเอส (GPS Module) เพื่อให้สามารถค้นหาสัญญาณดาวเทียมได้
2. เมื่อเริ่มเปิดเครื่องจะมีไฟบอกสถานะของ Module ส่วนต่าง ๆ ดังรูปที่ 4.2



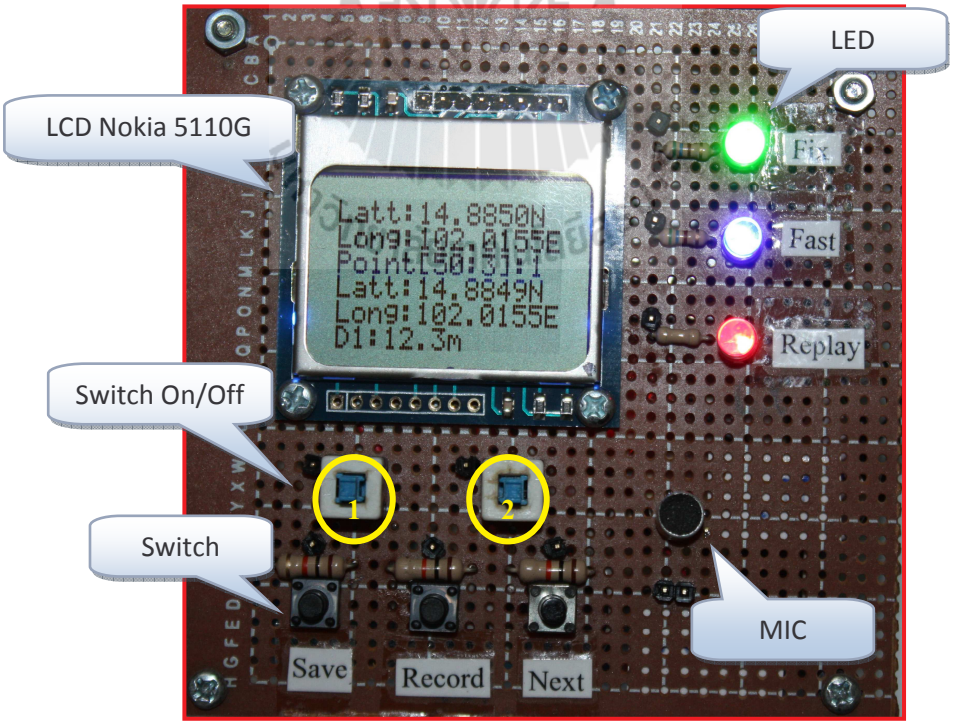
รูปที่ 4.2 ลักษณะของสถานะของ Module ส่วนต่าง ๆ

3. จากนั้นเครื่องบอกพิกัดตำแหน่งจะทำการตรวจสอบการรับสัญญาณของ GPS Module หากยังไม่สามารถรับสัญญาณดาวเทียมได้จะแสดงข้อความ “ Wait Connect GPS!” ที่จอ LCD และสามารถตรวจสอบไฟสถานะของ GPS Module จะยังคงติดค้างตลอดเวลา แต่หากรับสัญญาณดาวเทียมได้จะมีข้อความ “Connect GPS OK!” ที่จอ LCD และไฟสถานะของ GPS Module จะเปลี่ยนมาเป็นกระพริบ ดังรูปที่ 4.3



รูปที่ 4.3 ลักษณะของสถานะของการรับสัญญาณ GPS

4. เมื่อสามารถรับสัญญาณ GPS ได้แล้ว Display Broad จะแสดงสถานะต่าง ๆ ของเครื่องบอกพิกัดตำแหน่งดังรูปที่ 4.4



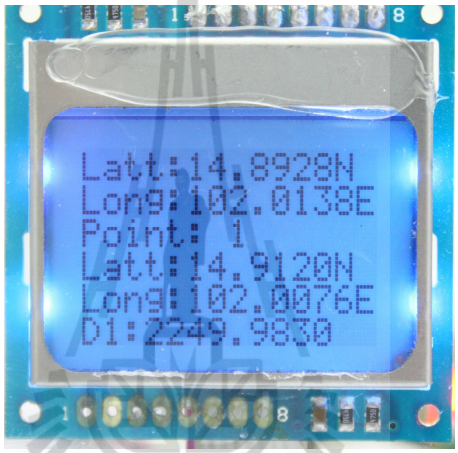
รูปที่ 4.4 ลักษณะของ Display Broad

4.1 จอ LCD Nokia 5110G จะบอกสถานะต่าง ๆ ดังรูปที่ 4.5 คือ

- Latitude และ Longitude ➔ บอกพิกัดตำแหน่งปัจจุบัน
- Point [X:Y] ➔ บอกลำดับของการบันทึกข้อมูล

  X คือ ระยะทางที่ต้องการตรวจสอบ

  Y คือ การรับพิกัดตำแหน่งเป็นจำนวน N ครั้งแล้วเฉลี่ย
- Latitude และ Longitude ➔ บอกพิกัดตำแหน่งของลำดับที่บันทึก
- D ➔ บอกระยะทางระหว่างพิกัดตำแหน่งปัจจุบันกับพิกัดตำแหน่งที่บันทึก



รูปที่ 4.5 การบอกสถานะของ LCD Nokia 5110G

4.2 LED จะบอกสถานะของการทำงาน 3 อย่างคือ

Fix ➔ จะแสดงเมื่ออยู่ในพิกัดตำแหน่งที่ได้บันทึก

Fast ➔ จะแสดงเมื่อใช้โหมดการทำงานแบบ Fast แต่ถ้าหากไม่แสดงหมายถึงใช้โหมดการทำงานแบบ Slow

Replay ➔ จะแสดงเมื่อทำงานแบบเล่นเสียงซ้ำตลอดเวลาที่อยู่ในพิกัดตำแหน่งที่บันทึกเอาไว้

4.3 Switch On/Off จะเป็นสวิตซ์เพื่อเลือกโหมดการทำงาน คือ

Switch 1 ➔ สำหรับเลือกโหมด Fast /Slow

Switch 2 ➔ สำหรับเลือกให้มีการเล่นเสียงซ้ำตลอดเวลาที่อยู่ในพิกัดตำแหน่งที่บันทึกเอาไว้

4.4 Switch จะเป็นสวิตซ์เพื่อสั่งงานเครื่องบอกพิกัดตำแหน่งให้ทำงานในรูปแบบต่าง ๆ คือ

Switch Save ➔ เป็นสวิตซ์เพื่อเข้าสู้การบันทึกเสียงและพิกัดตำแหน่ง

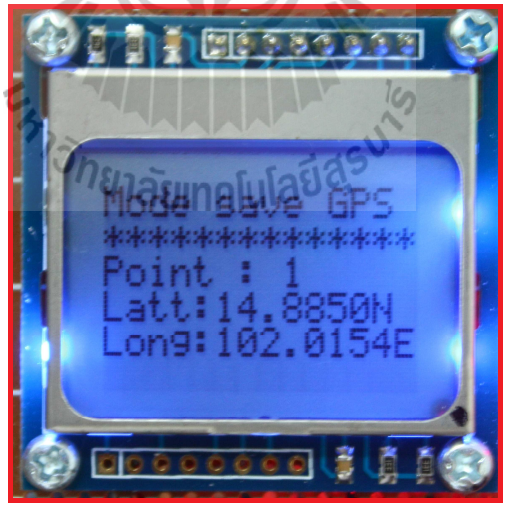
Switch Record ➔ เป็นสวิตซ์เพื่อการเริ่มและหยุดการบันทึกเสียง

Switch Next ➔ เป็นสวิตซ์เพื่อเลือกลำดับของการบันทึก

4.1.2 การบันทึกพิกัดตำแหน่งและการบันทึกเสียง

การบันทึกพิกัดตำแหน่งนั้นจะต้องกำหนดพิกัดตำแหน่งอ้างอิงที่ต้องการจะให้รายงานผล ซึ่งเป็นตำแหน่งใดก็ได้ เพราะเมื่อทำการบันทึกพิกัดตำแหน่งและบันทึกเสียงเสร็จแล้วนั้น หากไมโครคอนโทรลเลอร์ประมวลผลแล้วพบว่าอยู่ในบริเวณที่ได้มีการบันทึก จะมีการรายงานผลในรูปแบบเสียงตามที่ได้บันทึกเอาไว้ ซึ่งสามารถบันทึกพิกัดตำแหน่งได้ดังนี้

1. เลือกพิกัดตำแหน่งที่ต้องการที่จะบันทึก จากนั้นกดปุ่ม Save ค้างไว้จนกว่าจะได้ยินเสียงเตือน 1 ครั้ง เมื่อได้ยินเสียงเตือนแล้วจะเข้าสู่การบันทึกพิกัดตำแหน่งและบันทึกเสียงโดยที่หน้าจอ LCD จะแสดงข้อมูลต่าง ๆ ตามรูปที่ 4.6 และหากต้องการที่ออกจากส่วนของการบันทึกเสียงกลับไปยังโปรแกรมหลัก สามารถทำได้โดยการกดปุ่ม Save อีกครั้ง หรือถ้าหากไม่ทำการกดปุ่มใด ๆ ภายในเวลา 30 วินาที จะกลับไปสู่โปรแกรมหลักโดยอัตโนมัติ



รูปที่ 4.6 ลักษณะของจอ LCD ในการบันทึกตำแหน่ง

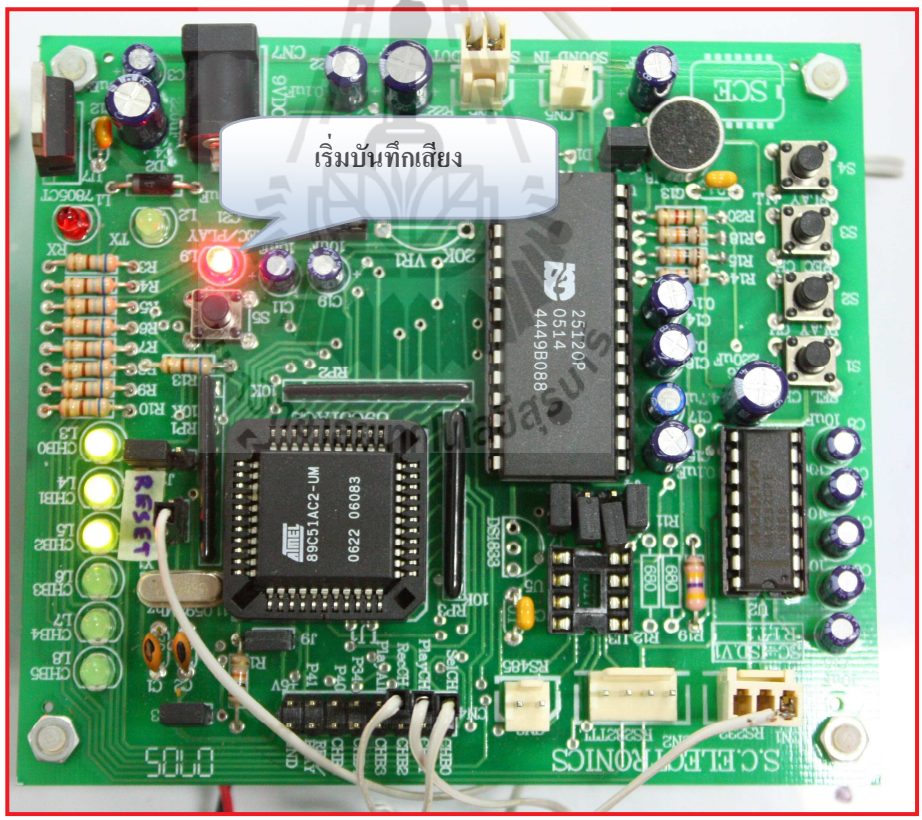
การแสดงสถานะของส่วนการบันทึกเสียงดังรูปที่ 4.6 มีดังนี้

- Point ➔ แสดงลำดับที่ต้องการบันทึก ซึ่งสามารถบันทึกได้ 8 ลำดับ
- Latitude และ Longitude ➔ บอกพิกัดตำแหน่งที่ต้องการบันทึก

2. กดปุ่ม Next เพื่อเลือกลำดับที่ต้องการบันทึก

3. กดปุ่ม Record เพื่อทำการบันทึกเสียง และรอจนกว่าจะมีไฟสถานะสีแดงที่บอร์ดบันทึกและเล่นเสียงติดค้างดังรูปที่ 4.7 จึงจะสามารถเริ่มบันทึกเสียงได้ และเมื่อต้องการสิ้นสุดการบันทึกเสียงให้กดปุ่ม Record อีกครั้งไฟสถานะสีแดงที่บอร์ดบันทึกและเล่นเสียงจะดับลง หมายถึงการบันทึกเสียงสิ้นสุด และการบันทึกเสียงในแต่ละครั้งนั้นจะมีการบันทึกพิกัดตำแหน่งด้วย ซึ่งสามารถตรวจสอบตำแหน่งได้จากจอแสดงผล ดังรูปที่ 4.6

*** **หมายเหตุ** เนื่องจากความสามารถของบอร์ดบันทึกและเล่นเสียง สามารถบันทึกเสียงได้เพียง 2 นาที ซึ่งเมื่อทำการแบ่งออกเป็น 8 ลำดับแล้ว จะสามารถบันทึกได้ลำดับละ 15 วินาที ดังนั้นเมื่อมีการบันทึกเสียงเกิน 15 วินาที เครื่องจะทำการหยุดการบันทึกเสียงโดยอัตโนมัติ ซึ่งจะบันทึกข้อมูลได้แค่ 15 วินาทีแรกเท่านั้น ส่วนที่เกิน 15 วินาทีจะไม่ถูกบันทึก



รูปที่ 4.7 ลักษณะของการเริ่มบันทึกเสียง

4. หลังจากที่มีการกดสวิตซ์ Record เพื่อทำการหยุดการบันทึกเสียง จะมีข้อความ " Record Comple " แสดงบนที่จอแสดงผล LCD และจะกลับมาส่วนของการบันทึกเสียงอีกครั้ง ดังรูปที่ 4.6 ถ้าหากต้องการที่จะบันทึกพิกัดตำแหน่งลำดับต่อไป สามารถกดปุ่ม Next เพื่อเลือกลำดับที่ต้องการบันทึกลงในเครื่องบอกพิกัดตำแหน่ง และทำการบันทึกเหมือนข้อ 3 แต่หากไม่ต้องการบันทึกและกลับไปยังหน้าโปรแกรมหลักให้กดปุ่ม Save เพื่อออกจากส่วนการบันทึกเสียง

4.2 การทดลองใช้งานโครงงาน

การทดลองใช้งานโครงงานเครื่องบอกตำแหน่ง GPS แบบบันทึกเสียง ผู้จัดทำได้นำโครงงานไปทดสอบโดยการบันทึกพิกัดตำแหน่ง ภายในมหาวิทยาลัยเทคโนโลยีสุรนารี และนำค่าพิกัดตำแหน่งที่ได้ทำการบันทึก มาแสดงในแผนที่ (Google Earth) เพื่อเป็นการเปรียบเทียบความผิดพลาดของโครงงาน โดยข้อมูลพิกัดตำแหน่งที่ได้ถูกบันทึกทั้ง 7 ลำดับมีดังนี้

➔A. ฟาร์มมหาวิทยาลัยเทคโนโลยีสุรนารี บริเวณอาคารจำหน่ายสินค้า
บันทึกลงในลำดับที่ 1
พิกัดตำแหน่ง Latitude = 14.8902 N , Longitude = 102.0053 E

➔B. อาคารเรียนรวม 2
บันทึกลงในลำดับที่ 2
พิกัดตำแหน่ง Latitude = 14.8809 N , Longitude = 102.0146 E

➔C. อาคารบรรณสาร
บันทึกลงในลำดับที่ 3
พิกัดตำแหน่ง Latitude = 14.8792 N , Longitude = 102.0157 E

➔D. อาคารเครื่องมือ 3 (F3)
บันทึกลงในลำดับที่ 4
พิกัดตำแหน่ง Latitude = 14.8757 N , Longitude = 102.0182 E

➔E. หอสุรนภา
บันทึกลงในลำดับที่ 5
พิกัดตำแหน่ง Latitude = 14.8724 N , Longitude = 102.0240 E

➔F. อาคารสุรพัฒน์ 1

บันทึกลงในลำดับที่ 6

พิกัดพิกัดตำแหน่ง Latitude = 14.8769 N , Longitude = 102.0230 E

➔G. เสาธง

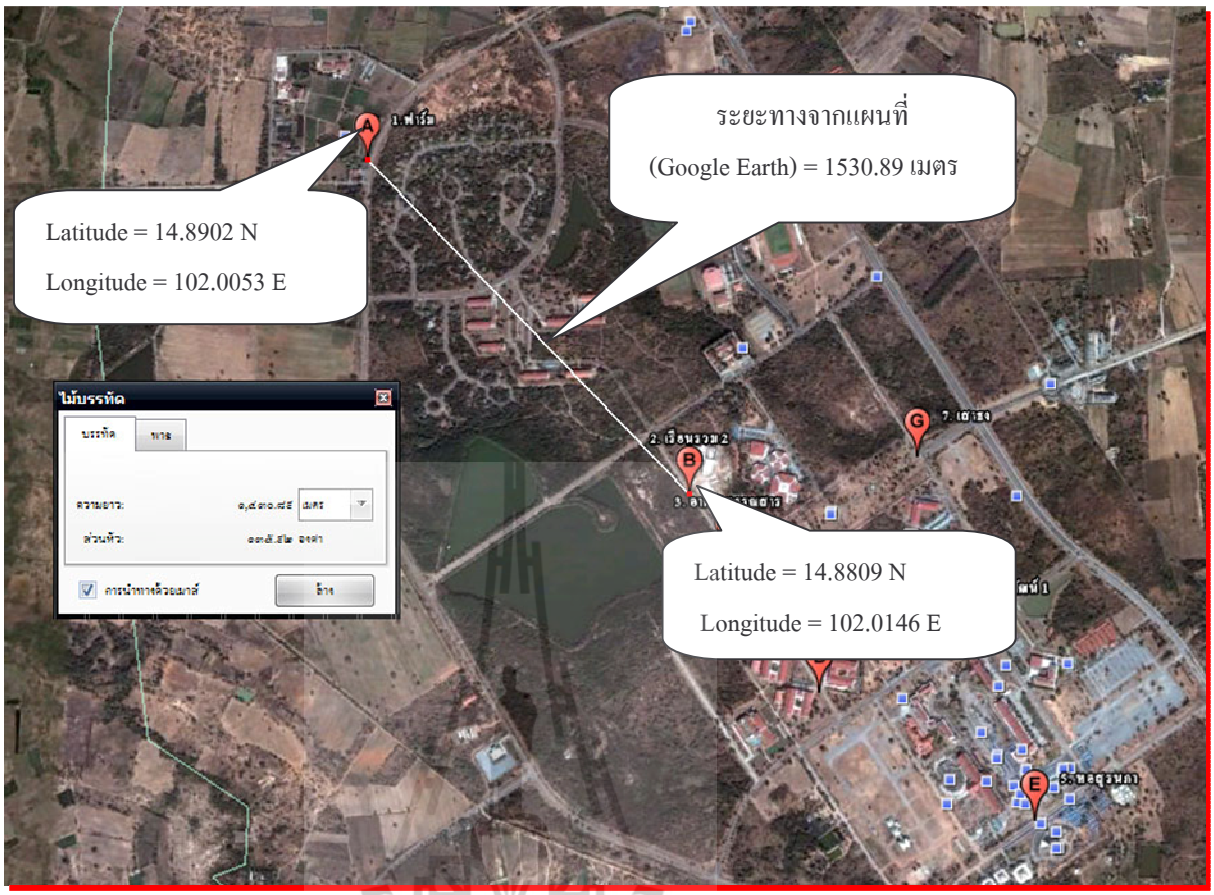
บันทึกลงในลำดับที่ 7

พิกัดตำแหน่ง Latitude = 14.8819 N , Longitude = 102.0209 E

จากพิกัดตำแหน่งที่ได้จากโครงงานเมื่อนำมาเปรียบเทียบกับแผนที่ (Google Earth) จะได้ตำแหน่งต่าง ๆ ดังรูปที่ 4.8

รูปที่ 4.8 พิกัดตำแหน่งของการทดสอบ

จากการทดสอบนำค่าพิกัดตำแหน่งที่ได้จากการโครงงานมาแสดงในแผนที่(Google Earth) จะเห็นได้ว่าเมื่อพิจารณาจากพิกัดตำแหน่งที่ปรากฏในแผนที่นั้นใกล้เคียงกับพิกัดตำแหน่งที่ได้มีการบันทึกจากเครื่องบอกพิกัดตำแหน่ง



รูปที่ 4.9 ระยะทางระหว่างพิกัดตำแหน่ง A และ B

นอกจากการนำพิกัดตำแหน่งมาแสดงแล้วนั้น การหาระยะทางระหว่างพิกัดตำแหน่ง A และ B เมื่อวัดระยะทางในแนวเส้นตรง จากการวัดในแผนที่ (Google Earth) ได้ค่าระยะทาง 1530.89 เมตร ดังรูปที่ 4.9 ซึ่งเครื่องบอกพิกัดตำแหน่งสามารถคำนวณระยะทางได้ 1467.54 เมตร ซึ่งจะเห็นว่าค่าที่ได้จากแผนที่ (Google Earth) และจากเครื่องบอกพิกัดตำแหน่งมีค่าคลาดเคลื่อนประมาณ 63.3 เมตร คิดเป็น 4.14 % ทั้งนี้เนื่องจากเครื่องบอกพิกัดตำแหน่งได้คำนวณระยะทางโดยประมาณเท่านั้น และยังขึ้นอยู่กับค่าพิกัดตำแหน่งที่ได้จากเครื่องรับสัญญาณจีพีเอสในแต่ละครั้งว่ามีความคลาดเคลื่อนเพียงใด

4.3 ผลการทดสอบโครงงาน

เมื่อนำโครงงานมาทดสอบการใช้งาน โดยเคลื่อนที่ด้วยความเร็วเฉลี่ย 40 กม./ชม. ผลการทดสอบโครงงานได้ผลการทดสอบดังนี้

1. การบันทึกพิกัดตำแหน่ง สามารถบันทึกพิกัดตำแหน่งและบันทึกเสียง เพื่อเป็นตำแหน่งอ้างอิงและข้อมูลเสียงที่จะรายงานผลได้

2. การใช้โหมด Fast คือ การรับค่าพิกัดตำแหน่ง 3 ครั้งแล้วทำการเฉลี่ยหาค่าพิกัดตำแหน่ง และตรวจสอบระยะทางที่ 50 เมตร พบว่า เมื่อเคลื่อนที่เข้าหาพิกัดตำแหน่งอ้างอิง ค่าระยะทางที่คำนวณจากเครื่องบอกพิกัดตำแหน่งจะมีค่าลดลง เมื่อค่าระยะทางน้อยกว่า 50 เมตร จะมีการรายงานผลในรูปแบบเสียงตามที่ได้มีการบันทึกเอาไว้ และเมื่อเคลื่อนที่ออกจากพิกัดตำแหน่งอ้างอิงค่าระยะทางจะเพิ่มขึ้น เมื่อค่าพิกัดตำแหน่งมีค่าเกิน 50 เมตร จะหยุดการการแสดงผลทันที นอกจากนี้เมื่อเคลื่อนที่เข้าหาพิกัดตำแหน่งอ้างอิงเดิมจนได้ระยะทางน้อยกว่า 50 เมตร จึงจะแสดงผลอีกครั้ง

3. การใช้โหมด Slow คือ การรับค่าพิกัดตำแหน่ง 5 ครั้งแล้วทำการเฉลี่ยหาค่าพิกัดตำแหน่ง และตรวจสอบระยะทางที่ 20 เมตร พบว่า เมื่อเคลื่อนที่เข้าหาพิกัดตำแหน่งอ้างอิง ค่าระยะทางที่คำนวณจากเครื่องบอกพิกัดตำแหน่งจะมีค่าลดลง เมื่อค่าระยะทางน้อยกว่า 20 เมตร จะมีการรายงานผลในรูปแบบเสียงตามที่ได้มีการบันทึกเอาไว้ และเมื่อเคลื่อนที่ออกจากพิกัดตำแหน่งอ้างอิงค่าระยะทางจะเพิ่มขึ้น เมื่อค่าพิกัดมีค่าเกิน 20 เมตร จะหยุดการการแสดงผลทันที

4. การเล่นเสียงแบบ Replay เมื่ออยู่ในพื้นที่พิกัดตำแหน่งอ้างอิง จะมีการรายงานผลในรูปแบบเสียงตามที่บันทึกเอาไว้ตลอดเวลาจนกว่าจะมีการเคลื่อนที่ออกจากพิกัดตำแหน่งอ้างอิง

ผลการทดสอบโครงงานในแต่ละตำแหน่ง

1. ฟาร์มมหาวิทยาลัยเทคโนโลยีสุรนารี บริเวณอาคารจำหน่ายสินค้า

ทดสอบครั้งที่	เล่นเสียง	ไม่เล่นเสียง
1	✓	
2	✓	
3	✓	
4	✓	
5	✓	
6	✓	
7	✓	
8	✓	
9	✓	
10	✓	

2. อาคารเรียนรวม 2

ทดสอบครั้งที่	เล่นเสียง	ไม่เล่นเสียง
1	✓	
2	✓	
3	✓	
4	✓	
5	✓	
6	✓	
7	✓	
8	✓	
9	✓	
10	✓	

3. อาคารบรรณสาร

ทดสอบครั้งที่	เล่นเสียง	ไม่เล่นเสียง
1	✓	
2	✓	
3	✓	
4	✓	
5	✓	
6	✓	
7	✓	
8	✓	
9	✓	
10	✓	

4. อาคารเครื่องมือ 3 (F3)

ทดสอบครั้งที่	เล่นเสียง	ไม่เล่นเสียง
1	✓	
2	✓	
3	✓	
4	✓	
5	✓	
6	✓	
7	✓	
8	✓	
9	✓	
10	✓	

5. หอสุรนภา

ทดสอบครั้งที่	เล่นเสียง	ไม่เล่นเสียง
1	✓	
2	✓	
3	✓	
4	✓	
5	✓	
6	✓	
7	✓	
8	✓	
9	✓	
10	✓	

6. อาคารสุรพัฒน์ 1

ทดสอบครั้งที่	เล่นเสียง	ไม่เล่นเสียง
1	✓	
2	✓	
3	✓	
4	✓	
5	✓	
6	✓	
7	✓	
8	✓	
9	✓	
10	✓	

7. เสาธง

ทดสอบครั้งที่	เล่นเสียง	ไม่เล่นเสียง
1	✓	
2	✓	
3	✓	
4	✓	
5	✓	
6	✓	
7	✓	
8	✓	
9	✓	
10	✓	

สรุปผลการทดลอง

จากการทดสอบจะเห็นว่า เครื่องบอกพิกัดตำแหน่งสามารถเล่นเสียงเมื่ออยู่ในพื้นที่ที่ได้ทำการบันทึกเอาไว้ ซึ่งสามารถเล่นได้ทุกตำแหน่งที่ได้ทำการบันทึก

บทที่ 5

สรุปผลการทดลอบและข้อเสนอแนะ

โครงงานเครื่องบอกตำแหน่ง GPS แบบบันทึกเสียง เป็นโครงงานที่ใช้งานสำหรับการบอกพิกัดตำแหน่งที่รายงานผลในรูปแบบเสียง ผู้ใช้งานสามารถบันทึกพิกัดตำแหน่งและบันทึกเสียงได้ตามต้องการ เมื่อต้องการจะรายงานผลในพิกัดตำแหน่งใด ให้ทำการบันทึกเสียงที่พิกัดตำแหน่งนั้น และเมื่อเคลื่อนที่ไปยังพิกัดตำแหน่งที่ได้ทำการบันทึก เครื่องรายงานผลในรูปแบบเสียงตามที่ผู้ใช้งานได้บันทึกเอาไว้ และจากการทดสอบการใช้งานของโครงงานเครื่องบอกพิกัดตำแหน่ง GPS แบบบันทึกเสียงสามารถสรุปได้ดังนี้ คือ

1. เมื่อถึงพิกัดตำแหน่งอ้างอิง จะเล่นเสียงตามที่บันทึกเอาไว้
2. สามารถบันทึกพิกัดตำแหน่งตำแหน่งและบันทึกเสียงเพิ่มได้
3. สามารถบันทึกตำแหน่งได้ 8 ลำดับ (ความยาวของข้อความเสียงลำดับละ 15 วินาที)
4. สามารถเลือกเล่นเสียงแบบเล่นซ้ำตลอดเวลาเมื่ออยู่ในพิกัดตำแหน่งอ้างอิง
5. สามารถเลือกโหมดการตรวจสอบพิกัดตำแหน่งได้ 2 โหมดการทำงานคือ
   - Fast (ตรวจสอบระยะทาง 50 เมตร, รับค่าพิกัดตำแหน่ง GPS 3 ครั้งแล้วทำการเฉลี่ย)
   - Slow (ตรวจสอบระยะทาง 20 เมตร, รับค่าพิกัดตำแหน่ง GPS 5 ครั้งแล้วทำการเฉลี่ย)

ปัญหาและอุปสรรค

1. โครงงานนี้สามารถบันทึกเสียงได้เพียงข้อความสั้น ๆ เท่านั้น
2. โครงงานนี้จะมีการเปลี่ยนแปลงพิกัดตำแหน่งภายในระยะเวลาประมาณ 1-2 วินาที ดังนั้น ความเร็วในการเคลื่อนที่ไม่ควรเกิน 60 กิโลเมตร/ชั่วโมง เพื่อไม่ให้เกิดความผิดพลาดของการคำนวณระยะทางในขณะเคลื่อนที่มากเกินไป
3. หากต้องการเพิ่มหรือลดจำนวนลำดับของการบันทึก ไม่สามารถทำได้โดยตรงจากเครื่องบอกพิกัดตำแหน่ง

สิ่งที่ได้รับจากการทำโครงงาน

1. ได้รับความรู้เกี่ยวกับหลักการทำงานของอุปกรณ์ตรวจจับตำแหน่ง (GPS Module)
2. ได้รับความรู้เกี่ยวกับเรื่อง GPS (Global Positioning System)
3. ได้รับความรู้เกี่ยวกับการใช้งานไมโครคอนโทรลเลอร์ในการควบคุมการทำงานของอุปกรณ์ ต่าง ๆ และการประยุกต์ใช้งานไมโครคอนโทรลเลอร์สำหรับใช้งานอย่างอื่น
4. ได้เรียนรู้เกี่ยวกับอุปกรณ์บันทึกเสียง (ISD 25120)
5. ได้เรียนรู้การทำงานร่วมกับผู้อื่น
6. สามารถนำความรู้ที่ได้จากการศึกษาทฤษฎีมาปฏิบัติจริงได้
7. สามารถนำความรู้และประสบการณ์ที่ได้จากโครงงานไปประยุกต์ใช้ในชีวิตจริง

นายปัญญา หันตุลา

เกิดเมื่อวันที่ 8 ธันวาคม พ.ศ. 2530

ภูมิลำเนาอยู่ที่ ตำบลขอนยูง อำเภอกุดจับ จังหวัดอุดรธานี

สำเร็จการศึกษาระดับมัธยมปลายจากโรงเรียนกุดจับประชาสรรค์ อำเภอกุดจับ จังหวัดอุดรธานี เมื่อปี พ.ศ. 2549

ปัจจุบันเป็นนักศึกษาชั้นปีที่ 4 สาขาวิชาวิศวกรรมโทรคมนาคม สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี

นายณัฐพล นฤมลต์

เกิดเมื่อวันที่ 26 กรกฎาคม พ.ศ. 2530

ภูมิลำเนาอยู่ที่ ตำบลบ่อยาง อำเภอสว่างอารมณ์ จังหวัดอุทัยธานี

สำเร็จการศึกษาระดับมัธยมปลายจากโรงเรียนหนองฉางวิทยา อำเภอหนองฉาง จังหวัดอุทัยธานี เมื่อปี พ.ศ. 2549

ปัจจุบันเป็นนักศึกษาชั้นปีที่ 4 สาขาวิชาวิศวกรรมโทรคมนาคม สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี

นางสาวสุพรรณธนา ใจรักษ์

เกิดเมื่อวันที่ 18 สิงหาคม พ.ศ. 2530

ภูมิลำเนาอยู่ที่ ตำบลงิ้วงาม อำเภอเมือง จังหวัดพิษณุโลก

สำเร็จการศึกษาระดับมัธยมปลายจากโรงเรียนจ่านกร้อง ตำบลในเมือง อำเภอเมือง จังหวัดพิษณุโลก เมื่อปี พ.ศ. 2549

ปัจจุบันเป็นนักศึกษาชั้นปีที่ 4 สาขาวิชาวิศวกรรมโทรคมนาคม สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี

บรรณานุกรม

[1] ศูนย์ภูมิภาคเทคโนโลยีอวกาศและภูมิสารสนเทศภาคใต้
คณะการจัดการสิ่งแวดล้อม มหาวิทยาลัยสงขลานครินทร์
Available from : URL : *http://www.rsgis.psu.ac.th*

[2] คร ภักดีชาติ, อรรถพล บุญยะโภคาม, โอภาส ศิริครรชิตถาวร และชัยวัฒน์ ลิ้มพรวิไล (ม.ป.ป.).
คู่มือทดลองไมโครคอนโทรลเลอร์ 32 บิตตระกูล ARM7 เบื้องต้น ฉบับ LPC2148.
บริษัท อินโนเวตีฟ เอ็กเพอริเมนต์ จำกัด.

[3] บริษัทอีทีที จำกัด. คู่มือการใช้งานบอร์ด CP-JR ARM7 USB-LPC2148 EXP [On line] จาก :
http://www.ett.co.th/product/ARM/images/CP_JR_ARM7_LPC2138/MAN_CP_JR_ARM7_LPC2148.pdf

[4] บริษัทอีทีที จำกัด. คู่มือการใช้งานบอร์ด ET-MINI SD/MMC [On line] จาก :
http://www.ett.co.th/product/InterfaceBoard/P-ET-A-00299/MAN_MINI_mp3_SD_AMP_LOGIC_PWR33.pdf

ภาคผนวก

ตารางที่ 2.2.1 PLL Register

ชื่อทั่วไป	ความหมาย	ติดต่อ	ค่าหลังรีเซต	ชื่อและแอดเดรสของ PLL0	ชื่อและแอดเดรสของ PLL1
PLLCON	PLL Control Register เก็บค่ารีจิสเตอร์สำหรับการอัพเดทค่า PLL control bit ค่าที่เขียนให้ยังไม่มีผลจนกว่าจะเขียน PLL feed sequence ที่ถูกต้องให้	อ่าน/เขียน	0	PLL0CON 0xE01FC080	PLL1CON 0xE01FC0A0
PLLCFG	PLL Configuration Register เก็บค่าสำหรับอัพเดทค่าของ PLL Configuration ค่าทีเขียนให้ยังไม่มีผลจนกว่าจะเขียน PLL feed sequence	อ่าน/เขียน	0	PLL0CFG 0xE01FC084	PLL1CFG 0xE01FC0A4
PLLSTAT	PLL Configuration อ่านค่าสถานะของ PLL Control และ configuration	อ่าน/เขียน	0	PLL0STAT 0xE01FC088	PLL1STAT 0xE01FC0A8
PLLFEED	PLL Feed Register เป็นการสั่งโหลดค่าที่เก็บใน PLLCON และ PLLCFG เพื่อสั่งให้ทำงานตามค่าที่กำหนดให้	อ่าน/เขียน	ไม่มี	PLL0FEED 0xE01FC08C	PLL1FEED 0xE01FC0AC

ตารางที่ 2.2.2 PLLCFG Register

ค่าบิตของ PLLCFG	หน้าที่	ความหมาย	ค่าหลังรีเซต
4:0	MSEL 4:0	ค่าตัวคูณ M ของวงจร PLL 00000 คือ M=1 , 00001 คือ M=2,…..,11110 คือ M =31 และ 11111 คือ M=32	0
6:5	PSEL 1:0	ค่าตัวหาร P ของวงจร PLL 00 คือ P=1 ,01 คือ P=2 10 คือ P=2 , 11 คือ P=4	0
7	สงวนไว้	ห้ามเขียนค่า 1 ให้กับบิตนี้	ไม่ได้กำหนด

ตารางที่ 2.2.3 PLLCON Register

ค่าบิตของ PLLCON	หน้าที่	ความหมาย	ค่าหลังรีเซต
0	PLLE	PLL Enable สั่งให้ PLL ทำงาน โดยเขียนค่าเป็น 1 และต้องเขียนค่า PLLFEED ที่เหมาะสมด้วย	0
1	PLLC	PLL Connect เมื่อ PLLC และ PLLE มีค่าเป็น 1 และหลังจากเขียนค่า PLLFEED ที่เหมาะสม จะเป็นการสั่งให้ต่อ PLL เป็นสัญญาณนาฬิกาหลังของวงจร	0
7:2	สงวนไว้	ห้ามเขียนค่า 1 ให้กับบิตนี้	ไม่ได้กำหนด

ตารางที่ 2.2.4 ค่าแต่ละบิตของรีจิสเตอร์ PLLSTAT

ค่าบิตของ PLLSTAT	หน้าที่	ความหมาย	ค่าหลังรีเซต
4:0	MSEL 4:0	อ่านค่าว่าปัจจุบันวงจร PLL ใช้ค่าตัวคูณความถี่เท่ากับเท่าไร	0
6:5	PSEL 1:0	อ่านค่าว่าปัจจุบันวงจร PLL ใช้ค่าตัวหารความถี่เท่ากับเท่าไร	0
7	สงวนไว้	ห้ามเขียนค่า 1 ให้กับบิตนี้	ไม่กำหนด
8	PLLE	อ่านค่าว่าปัจจุบันบิต PLLE มีค่าเป็น 0 หรือ 1	0
9	PLLC	อ่านค่าว่าปัจจุบันบิต PLLC มีค่าเป็น 0 หรือ 1	0
10	PLOCK	ถ้าเป็น 0 แสดงว่า PLL ยังไม่สามารถล็อคค่าความถี่ ถ้าเป็น 1 แสดงว่าสามารถล็อคค่าความถี่ตามที่กำหนดได้แล้ว	0
15:11	สงวนไว้	ห้ามเขียนค่า 1 ให้กับบิตนี้	ไม่กำหนด

ตารางที่ 2.2.5 โหมดการทำงานของวงจรPLL

บิต PLLC	บิต PLLE	การทำงานของ PLL
0	0	PLL ไม่ทำงาน ไม่ได้ต่อเข้ากับไมโครคอนโทรลเลอร์
0	1	PLL ทำงานแต่ไม่ได้ต่อกับไมโครคอนโทรลเลอร์
1	0	เหมือนกับกรณีค่า 00 เพื่อป้องกันไม่ให้ต่อ PLL เข้ากับระบบถ้ายังไม่สั่งให้มันทำงาน
1	1	PLL ทำงาน และต่อเป็นสัญญาณนาฬิกาหลัก

ตารางที่ 2.2.6 VPBDIV Register

แอดเดรส	ชื่อ	ความหมาย	ติดต่อ
0xE10FC100	VPBDIV	ควบคุมค่าตัวหารความถี่ของ VPB Bus	อ่าน/เขียน

ตารางที่ 2.2.7 ค่าแต่ละบิตของรีจิสเตอร์ VPBDIV

ค่าบิตของ VPBDIV	หน้าที่	ความหมาย	ค่าหลังรีเซต
1:0	VPBDIV	VPB bus clock = CCLK/4 VPB bus clock = CCLK VPB bus clock = CCLK/2 สงวนไว้	0
7:2	สงวนไว้	ห้ามเขียนค่า 1 ให้บิตเหล่านี้	0

ตารางที่ 2.2.8 MAM Register

แอดเดรส	ชื่อ	ความหมาย	ติดต่อ
0xE01FC000	MAMCR	Memory Accelerator Module Control Register ใช้กำหนดโหมดการทำงานของ MAM	อ่าน/เขียน
0xE01FC004	MAMTIM	Memory Accelerator Module Timing Control ใช้กำหนดจำนวนสัญญาณนาฬิกาที่ใช้ในการติดต่อกับหน่วยความจำ Flash	อ่าน/เขียน

ตารางที่ 2.2.9 การกำหนดค่าให้กับ MAMCR Register

MAMCR bit	ฟังก์ชัน	ความหมาย
1:0	MAN mode control	00-MAM Disable หยุดการทำงานของ MAM 01-MAM partially enable เปิดการทำงาน MAMบางส่วน 10-MAM fully enable เปิดการทำงาน MAMสมบรูณ์แบบ 11-สงวนไว้
7:2	สงวนไว้	สงวนไว้ ห้ามเขียนค่า 1 ให้บิตเหล่านี้

ตารางที่ 2.2.10 การกำหนดค่าให้กับ MAMTIM Register

MAMTIM bit	ฟังก์ชัน	ความหมาย
2:0	MAN Fetch Cycle timing	000-สงวนไว้ 001-MAM fetch cycles = 1 * CCLK 010-MAM fetch cycles = 2 * CCLK 011-MAM fetch cycles = 3 * CCLK 100-MAM fetch cycles = 4 * CCLK 101-MAM fetch cycles = 5 * CCLK 110-MAM fetch cycles = 6 * CCLK 111-MAM fetch cycles = 7 * CCLK
7:3	สงวนไว้	สงวนไว้ ห้ามเขียนค่า 1 ให้บิตเหล่านี้

ตารางที่ 2.2.11 รีจิสเตอร์ PINSEL0, PINSEL1, PINSEL2

ชื่อ	ความหมาย	การติดต่อ	แอดเดรส
PINSEL0	Pin function select register กำหนดการทำงานของ P0.0-P0.15	อ่าน/เขียน	0xE002 C000
PINSEL1	Pin Function select register 1 กำหนดการทำงานของ P0.16-P0.31	อ่าน/เขียน	0xE002 C004
PINSEL2	Pin Function select register 2 กำหนดการทำงานของ P01.0 –P1.31	อ่าน/เขียน	0xE002 C014

ตารางที่ 2.2.12 ค่าประจำบิตของรีจิสเตอร์ PINSEL0

PINSEL0	ชื่อขา	การทำงาน เมื่อมีค่า 00	การทำงาน เมื่อมีค่า 01	การทำงาน เมื่อมีค่า 10	การทำงาน เมื่อมีค่า 11	ค่าหลัง รีเซต
1:0	P0.0	GPIO Port0.0	TxD0(UART0)	PWM1	Reserved	00
3:2	P0.1	GPIO Port0.1	RxD0(UART0)	PWM3	EINT0	00
5:4	P0.2	GPIO Port0.2	SCL0(I^2C)	Capture 0.0(TIMER0)	Reserved	00
7:6	P0.3	GPIO Port0.3	SDA0(I^2C)	Match0.0 (TIMER0)	EINT1	00
9:8	P0.4	GPIO Port0.4	SCK0(SPI0)	Capture 0.1(TIMER0)	AD0.6	00
11:10	P0.5	GPIO Port0.5	MISO0(SPI0)	Match0.1 (TIMER0)	AD0.7	00
13:12	P0.6	GPIO Port0.6	MISI0(SPI0)	Capture 0.2(TIMER0)	AD1.0	00
15:14	P0.7	GPIO Port0.7	SSEL0(SPI0)	PWM2	EINT2	00
17:16	P0.8	GPIO Port0.8	TxD1 (UART1)	PWM4	AD1.1	00
19:18	P0.9	GPIO Port0.9	RxD1 (UART1)	PWM6	EINT3	00
21:20	P0.10	GPIO Port0.10	RTS1(UART1)	Capture 1.0(TIMER1)	AD1.2	00
23:22	P0.11	GPIO Port0.11	CTS1(UART1)	Capture 1.1(TIMER1)	SCL1(I^2C1)	00
25:24	P0.12	GPIO Port0.12	DSR1(UART1)	Match1.0 (TIMER1)	AD1.3	00
27:26	P0.13	GPIO Port0.13	DTR1(UART1)	Match1.1 (TIMER1)	AD1.4	00
29:28	P0.14	GPIO Port0.14	DCD1(UART1)	EINT1	SDA1(I^2C1)	00
31:30	P0.15	GPIO Port0.15	RI1(UART1)	EINT2	AD1.5	00

*****หมายเหตุ :**

- ขา P0.0 และ P0.1 ทำหน้าที่เป็น UART0 เพื่อติดต่อกับคอมพิวเตอร์ในการเขียน โปรแกรม
- การกำหนด ค่ารีจิสเตอร์ PINSEL0 ต้องระวังให้ P0.0 , P0.1 มีหน้าที่การทำงานเป็นขา TxD0 , RxD0, ของ UART0 เสมอ มิฉะนั้นอาจจะทำให้วงจรเสียได้ ดังนั้นจึงกำหนดค่าเริ่มต้นของ PINSEL0 ได้ดังนี้

```
PINSEL0 = 0x00000005 ;  // set P0.2 –P0.15  to GPIO  Function
```

ตารางที่ 2.2.13 ค่าประจำบิตของรีจิสเตอร์ PINSEL1

PINSEL1	ชื่อขา	การทำงาน เมื่อมีค่า 00	การทำงาน เมื่อมีค่า 01	การทำงาน เมื่อมีค่า 10	การทำงาน เมื่อมีค่า 11	ค่าหลัง รีเซต
1:0	P0.16	GPIO Port0.16	EINT0	Match0.2(TIMER0)	Capture0.2 (TIMER0)	00
3:2	P0.17	GPIO Port0.17	Capture1.2(TIMER1)	SCK1(SPI1)	Match1.2(TIMER1)	00
5:4	P0.18	GPIO Port0.18	Capture1.3(TIMER1)	MISO1(SPI1)	Match1.3(TIMER1)	00
7:6	P0.19	GPIO Port0.19	Match1.2(TIMER1)	MOIS1(SPI1)	Capture1.2 (TIMER1)	00
9:8	P0.20	GPIO Port0.20	Match1.3(TIMER1)	SSEL(SPI1)	EINT3	00
11:10	P0.21	GPIO Port0.21	PWM5	AD1.6	Capture1.3 (TIMER1)	00
13:12	P0.22	GPIO Port0.22	V_{BUS}	Capture0.0 (TIMER0)	Match0.0(TIMER0)	00
15:14	P0.23	GPIO Port0.23	AD0.7	สงวนไว้	สงวนไว้	00
17:16	P0.24	สงวนไว้				00
19:18	P0.25	GPIO Port0.25	AD0.4	AOUT	สงวนไว้	00
21:20	P0.26	สงวนไว้				00
23:22	P0.27	สงวนไว้				00
25:24	P0.28	GPIO Port0.28	AD0.1	Capture0.2 (TIMER0)	Match0.2(TIMER0)	00
27:26	P0.29	GPIO Port0.29	AD0.2	Capture0.3 (TIMER0)	Match0.3(TIMER0)	00
29:28	P0.30	GPIO Port0.30	AD0.3	EINT3	Match0.0(TIMER0)	00
31:30	P0.31	GPIO Port0.31	UP_LED	CONNECT		00

ตารางที่ 2.2.14 ค่าประจำบิตของรีจิสเตอร์ PINSEL2

PINSEL2	ความหมาย	ค่าหลังรีเซต
1:0	สงวนไว้	00
2	เป็น 0 ขา P1.31:26 จะใช้เป็น GPIO ถ้าเป็น1ขา P1.31:26จะเป็น Debug port	P1.26/RTCK
3	เป็น 0 ขา P1.25:16 จะใช้เป็น GPIO ถ้าเป็น1ขา P1.25:16จะเป็นTrace port	P1.20/ TRACESYNC

ตารางที่ 2.2.15 ชื่อและแอดเดรสของรีจิสเตอร์ที่ใช้ในการควบคุม GPIO

ชื่อทั่วไป	ความหมาย	การติดต่อ	ชื่อ และ แอดเดรสของPort0	ชื่อ และ แอดเดรสของPort0
IOPIN	GPIO Port Pin value register ใช้อ่านสถานะปัจจุบันของพอร์ต	อ่าน	0XE002 8000 IOPIN0	0XE002 8010 IOPIN1
IOSET	GPIO Port Output set register ใช้กำหนดค่าให้ขาของพอร์ตมีค่าเป็น 1 ถ้าบิตใดเป็น 1 ขาของพอร์ตที่ตรงกันจะมีค่าเป็น 1	อ่าน/เขียน	0XE002 8004 IOSET0	0XE002 8014 IOSET1
IODIR	GPIO Port Direction control register ใช้กำหนดว่าขาใดเป็นอินพุตขาใดเป็นเอาต์พุต	อ่าน/เขียน	0XE002 8008 IODIR0	0XE002 8018 IODIR1
IOCLR	GPIO Port Output clear register ใช้กำหนดให้ขาของพอร์ตมีค่าเป็น 0	เขียน	0XE002 800C IOCLR0	0XE002 801C IOCLR1

ตารางที่ 2.2.16 แหล่งกำเนิดอินเตอร์รัปต์ทั้ง 32 ตัว

อุปกรณ์	แหล่งกำเนิดอินเตอร์รัปต์	VIC Channel#
WDT	Watchdog Interupt (WDINT)	0
-	Reserved for software interrupts only	1
ARM Core	Embeded ICE, DbgCommRx	2
ARM Core	Embeded ICE, DbgcommTx	3
TIME0	Match 0-3 (MR0, MR1, MR2, MR3) Capture0-3 (CR0, CR1, CR2, CR3)	4
TIME1	Match0-3 (MR0, MR1, MR2, MR3) Capture 0-3 (CR0,CR1,CR2 CR3)	5
UART0	Rx Line Status (RLS) Transmit Holding Register Empty(THRE) Rx Data Available (RDA) Charater Time-out Indicator(CTI)	6
UART1	Rx Line Status (RLS) Transmit Holding Register Empty(THRE) Rx Data Available (RDA) Charater Time-out Indicator(CTI) Modem Status Interrupt(MSI)	7
PWM0	Match 0-6 (MR0, MR1, MR2, MR3, MR4, MR5, MR6)	8
I^2C0	SI(state change)	9
SPI0	SPI Interrupt Flag (SPIF) Mode Fault(MODF)	10
SPI1(SSP)	SPI Interrupt Flag(SPIF) Mode Fault (MODF)	11
PLL	PLL Lock (PLOCK)	12
RTC	Counter Indrement(RTCCIF) Alarm(RTCALF)	13

System Control	External Interrupt 0 (EINT0)	14
System Control	External Interrupt 1 (EINT1)	15
System Control	External Interrupt 2 (EINT2)	16
System Control	External Interrupt 3 (EINT3)	17
ADC0	A/D Converter 0 end of conversion	18
I^2C1	SI(state change)	19
BOD	Brown our detect	20
ADC1	A/D Converter 1 end of conversion	21
USB	USB interrupt, DMA Interrupt	22
	Reserved	23-31

ตารางที่ 2.2.17 รีจิสเตอร์ที่เกี่ยวข้องกับการอินเตอร์รัปต์

ชื่อ	ความหมาย	การติดต่อ	ค่าหลังรีเซต	แอดเดรส
VICIRQStatus	IRQ Status Request อ่านค่าสถานะว่าอนุญาตให้มี interrupt request จากตัวใด และถูกจัดเป็น IRQ	อ่าน	0	0xFFFF F000
VICFIQStatus	FIQ Status Request อ่านค่าสถานะว่าอนุญาตให้มีอินเตอร์รัปต์รีเควสจากตัวใด และถูกจัดเป็น FIQ	อ่าน	0	0xFFFF F004
VICRawltr	Raw Interrupt Status Register ใช้อ่านสถานะของอินเตอร์รัปต์จากทั้ง 32 แหแหล่งหรือจากซอฟต์แวร์โดยไม่สนว่าถูกเปิดใช้หรือถูกจัดประเภทหรือไม่	อ่าน/เขียน	0	0xFFFF F008
VICintselect	Interrupt Select Register ใช้ในการระบุว่าอินเตอร์รัปต์ทั้ง 32 แหล่งแต่ละตัวเป็น FIQ หรือ IRQ	เขียน	0	0xFFFF F00C

ชื่อ	ความหมาย	การติดต่อ	ค่าหลังรีเซต	แอดเดรส
VICIntEnable	Inerrupt E able Register ควบคุมว่าให้อินเตอร์รัปต์จากทั้ง32 แหล่งตัวไหนทำงาน และให้เป็น FIQ หรือ IRQ	อ่าน/เขียน	0	0xFFFF F010
VICIntEnClr	Interrupt Enable Clear Register ใช้ล้างค่าบิตหนึ่งบิตหรือมากกว่าหนึ่งบิตของรีจิสเตอร์ Interrupt Enable Register	เขียน	0	0xFFFF F014
VicSoftlnt	Software Interrupt Register ค่าของรีจิสเตอร์นี้จะถูกจำไป OR กับอินเตอร์รัปต์จากอุปกรณ์ต่างๆ ทั้ง 32 แหล่ง	อ่าน/เขียน	0	0xFFFF F018
VICSoftlntClear	Software Interrupt Clear Register ใช้ล้างค่าบิตหนึ่งบิตหรือมากกว่าหนึ่งบิตของรีจิสเตอร์ Software Interrupt Register	เขียน	0	0xFFFF F01C
VICProtection	Protection enable register ใช้จำกัดการเข้าถึง VIC Register	อ่าน/เขียน	0	0xFFFF F020
VICVectAddr	Vector Address Register เมื่อเกิดการอินเตอร์รัปต์แบบ IRQ ตัว IRQ Sevice routine จะอ่านค่าจากรีจิสเตอร์เพื่อกระโดดไปยังแอดเดรสที่ระบุไว้	อ่าน/เขียน	0	0xFFFF F030
VICDefVectAddr	Default Vector Address Register ใช้เก็บค่าแอดเดรสของ Interrupt service routine(ISR) สำหรับ IRQ non-vectored IRQ	อ่าน/เขียน	0	0xFFFF F100

ชื่อ	ความหมาย	การติดต่อ	ค่าหลังรีเซต	แอดเดรส
VICVEctAddr1	Vector address 1 register	อ่าน/เขียน	0	0xFFFF F104
VICVEctAddr2	Vector address 2 register	อ่าน/เขียน	0	0xFFFF F108
VICVEctAddr3	Vector address 3 register	อ่าน/เขียน	0	0xFFFF F10C
VICVEctAddr4	Vector address 4 register	อ่าน/เขียน	0	0xFFFF F110
VICVEctAddr5	Vector address 5 register	อ่าน/เขียน	0	0xFFFF F114
VICVEctAddr6	Vector address 6 register	อ่าน/เขียน	0	0xFFFF F118
VICVEctAddr7	Vector address 7 register	อ่าน/เขียน	0	0xFFFF F11C
VICVEctAddr8	Vector address 8 register	อ่าน/เขียน	0	0xFFFF F120
VICVEctAddr9	Vector address 9 register	อ่าน/เขียน	0	0xFFFF F124
VICVEctAddr10	Vector address 10 register	อ่าน/เขียน	0	0xFFFF F128
VICVEctAddr11	Vector address 11 register	อ่าน/เขียน	0	0xFFFF F12C
VICVEctAddr12	Vector address 12 register	อ่าน/เขียน	0	0xFFFF F130
VICVEctAddr13	Vector address 13 register	อ่าน/เขียน	0	0xFFFF F134
VICVEctAddr14	Vector address 14 register	อ่าน/เขียน	0	0xFFFF F138
VICVEctAddr15	Vector address 15 register	อ่าน/เขียน	0	0xFFFF F13C
VICVectCntl0	Vector control 0 register รีจิสเตอร์ Vector Control Register0-15 แต่ละตัวควบคุม IRQ Slot แต่ละตัว โดย Slot 0 มีความสำคัญสูงสุดและ Slot 15 มีความสำคัญต่ำสุด	อ่าน/เขียน	0	0xFFFF F200
VICVectCntl1	Vector control 1 register	อ่าน/เขียน	0	0xFFFF F204
VICVectCntl2	Vector control 2 register	อ่าน/เขียน	0	0xFFFF F208
VICVectCntl3	Vector control 3 register	อ่าน/เขียน	0	0xFFFF F20C
VICVectCntl4	Vector control 4 register	อ่าน/เขียน	0	0xFFFF F210
VICVectCntl5	Vector control 5 register	อ่าน/เขียน	0	0xFFFF F214
VICVectCntl6	Vector control 6 register	อ่าน/เขียน	0	0xFFFF F218

ชื่อ	ความหมาย	การติดต่อ	ค่าหลังรีเซต	แอดเดรส
VICVectCntl7	Vector control 7 register	อ่าน/เขียน	0	0xFFFF F21C
VICVectCntl8	Vector control 8 register	อ่าน/เขียน	0	0xFFFF F220
VICVectCntl9	Vector control 9 register	อ่าน/เขียน	0	0xFFFF F224
VICVectCntl10	Vector control10 register	อ่าน/เขียน	0	0xFFFF F228
VICVectCntl11	Vector control 11 register	อ่าน/เขียน	0	0xFFFF F22C
VICVectCntl12	Vector control 12 register	อ่าน/เขียน	0	0xFFFF F230
VICVectCntl13	Vector control 13 register	อ่าน/เขียน	0	0xFFFF F234
VICVectCntl14	Vector control 14 register	อ่าน/เขียน	0	0xFFFF F238
VICVectCntl15	Vector control 15 register	อ่าน/เขียน	0	0xFFFF F23C

ตารางที่ 2.2.18 รีจิสเตอร์ที่เกี่ยวข้องกับการอินเตอร์รัปต์จากภายนอก

แอดเดรส	ชื่อ	ความหมาย	การติดต่อ
0xE01FC140	EXTINT	External Interrupt Flag Register เก็บค่าอินเตอร์รัปต์แฟล็กของ EINT0, EINT1 และ EINT2	อ่าน/เขียน
0xE01FC144	EXTWAKE	External Interrupt Wakeup Register เก็บค่าบิตที่ควบคุมว่าจะให้อินเตอร์รัปต์จากภายนอกนี้สามารถกระตุ้นซีพียูจาก Power mode หรือไม่	อ่าน/เขียน
0xE01FC148	EXTMODE	External Interrupt Mode Register ควบคุมให้แต่ละระดับจะตอบสนองต่อระดับลอกจิกหรือขอบสัญญาณ	อ่าน/เขียน
0xE01FC14C	EXTPOLAR	External Interrupt Polarity Register ควบคุมว่าแต่ละขาจะตอบสนองระดับลอกจิก 0 หรือ1 หรือตอบสนองต่อขอบขาขึ้นขอบขาลงของสัญญาณ	อ่าน/เขียน

ตารางที่ 2.2.19 ค่าประจำบิตของรีจิสเตอร์ EXTINT

ค่าบิตของ EXTINT	หน้าที่	ความหมาย	ค่าหลังรีเซ็ต
0	EINT0	เมื่อขา EINT0 ทำงานมีระดับลอจิกหรือมีขอบของสัญญาณตามที่กำหนดเกิดขึ้น บิตนี้จะมีค่าเป็น 1 เมื่อเขียนค่า 1 ให้มันเป็นการล้างค่า ยกเว้นกรณีที่ทำงานตามระดับลอจิกต้องรอให้ค่าลอจิกหยุดการทำงานก่อน ขาที่เป็น EINT0 คือ P0.1 และ P0.16	0
1	EINT1	เมื่อขา EINT1 ทำงานมีระดับลอจิกหรือมีขอบของสัญญาณตามที่กำหนดเกิดขึ้น บิตนี้จะมีค่าเป็น 1 เมื่อเขียนค่า 1 ให้มนเป็นการล้างค่า ยกเว้นกรณีที่ทำงานตามระดับลอจิกต้องรอให้ค่าบอจิกหยุดการทำงานก่อน ขาที่เป็น EINT1 คือ P0.3 และ P0.14	0
2	EINT2	เมื่อขา EINT2 ทำงานมีระดับลอจิกหรือมีขอบของสัญญาณตามที่กำหนดเกิดขึ้น บิตนี้จะมีค่าเป็น 1 เมื่อเขียนค่า 1 ให้มนเป็นการล้างค่า ยกเว้นกรณีที่ทำงานตามระดับลอจิกต้องรอให้ค่าบอจิกหยุดการทำงานก่อน ขาที่เป็น EINT2 คือ P0.7 และ P0.15	0
3	EINT3	เมื่อขา EINT3 ทำงานมีระดับลอจิกหรือมีขอบของสัญญาณตามที่กำหนดเกิดขึ้น บิตนี้จะมีค่าเป็น 1 เมื่อเขียนค่า 1 ให้มนเป็นการล้างค่า ยกเว้นกรณีที่ทำงานตามระดับลอจิกต้องรอให้ค่าบอจิกหยุดการทำงานก่อน ขาที่เป็น EINT3 คือ P0.9, P0.20 และ P0.30	0
7:4	สงวนไว้	ห้ามเขียนค่า 1 ให้กับบิตเหล่านี้ ค่าที่อ่านได้ไม่มีความหมาย	ไม่กำหนด

ตารางที่ 2.2.20 ค่าประจำบิตของรีจิสเตอร์ EXTMODE

ค่าบิตของ EXTWAKE	หน้าที่	ความหมาย	ค่าหลังรีเซ็ต
0	EXTWAKE0	โหมดการทำงานของ EINT0 เมื่อมีค่าเป็น 0 ทำงานที่ระดับสัญญาณ เมื่อมีค่าเป็น 1 ทำงานที่ขอบของสัญญาณ	0
1	EXTWAKE1	โหมดการทำงานของ EINT1 เมื่อมีค่าเป็น 0 ทำงานที่ระดับสัญญาณ เมื่อมีค่าเป็น 1 ทำงานที่ขอบของสัญญาณ	0
2	EXTWAKE2	โหมดการทำงานของ EINT2 เมื่อมีค่าเป็น 0 ทำงานที่ระดับสัญญาณ เมื่อมีค่าเป็น 1 ทำงานที่ขอบของสัญญาณ	0
3	EXTWAKE3	โหมดการทำงานของ EINT3 เมื่อมีค่าเป็น 0 ทำงานที่ระดับสัญญาณ เมื่อมีค่าเป็น 1 ทำงานที่ขอบของสัญญาณ	0
7:4	สงวนไว้	ห้ามเขียนค่า 1 ให้กับบิตเหล่านี้ ค่าที่อ่านได้ไม่มีความหมาย	ไม่กำหนด

ตารางที่ 2.2.21 ค่าประจำบิตของรีจิสเตอร์ EXTPOLAR

ค่าบิตของ EXTPOLAR	หน้าที่	ความหมาย	ค่าหลังรีเซ็ต
0	EXTPOLAR0	เป็น 0 EINT0 ทำงานที่ลอจิก 0 หรือขอบขาลงของสัญญาณ เป็น1 EINT0 ทำงานที่ลอจิก 1 หรือขอบขาขึ้นของสัญญาณ	0
1	EXTPOLAR1	เป็น 0 EINT1 ทำงานที่ลอจิก 0 หรือขอบขาลงของสัญญาณ เป็น1 EINT1 ทำงานที่ลอจิก 1 หรือขอบขาขึ้นของสัญญาณ	0
2	EXTPOLAR2	เป็น 0 EINT2 ทำงานที่ลอจิก 0 หรือขอบขาลงของสัญญาณ เป็น1 EINT2 ทำงานที่ลอจิก 1 หรือขอบขาขึ้นของสัญญาณ	0

ค่าบิตของ EXTPOLAR	หน้าที่	ความหมาย	ค่าหลังรีเซ็ต
3	EXTPOLAR3	เป็น 0 EINT3 ทำงานที่ลอจิก 0 หรือขอบขาลงของสัญญาณ เป็น1 EINT3 ทำงานที่ลอจิก 1 หรือขอบขาขึ้นของสัญญาณ	0
7:4	สงวนไว้	ห้ามเขียนค่า 1 ให้กับบิตเหล่านี้ ค่าที่อ่านได้ไม่มีความหมาย	ไม่กำหนด

ตารางที่ 2.2.22 ค่าประจำบิตของรีจิสเตอร์ EXTPILAR

ค่าบิตของ EXTMODE	หน้าที่	ความหมาย	ค่าหลังรีเซ็ต
0	EXTMODE0	เป็น 1 สัญญาณที่ขา EINT0 สามารถกระตุ้นให้ออกจาก Power Down ได้	0
1	EXTMODE1	เป็น 1 สัญญาณที่ขา EINT1 สามารถกระตุ้นให้ออกจาก Power Down ได้	0
2	EXTMODE2	เป็น 1 สัญญาณที่ขา EINT2 สามารถกระตุ้นให้ออกจาก Power Down ได้	0
3	EXTMODE3	เป็น 1 สัญญาณที่ขา EINT0 สามารถกระตุ้นให้ออกจาก Power Down ได้	0
7:4	สงวนไว้	ห้ามเขียนค่า 1 ให้กับบิตเหล่านี้ ค่าที่อ่านได้ไม่มีความหมาย	ไม่กำหนด

ตารางที่ 2.2.23 รีจิสเตอร์ที่เกี่ยวข้องกับ UART0

ชื่อ	ความหมาย	บิต 7	บิต 6	บิต 5	บิต 4	บิต 3	บิต 2	บิต 1	บิต 0	การติดต่อ	ค่าหลังรีเซต	แอดเดรส
U0RBR	Receiver Buffer Register	MSB	อ่านข้อมูล						LSB	อ่าน	ไม่กำหนด	0xE000C000 DLAB=0
U0THR	Transmit Holding Register	MSB	เขียนข้อมูล						LSB	เขียน	ไม่กำหนด	0xE000C000 DLAB=0
U0IER	Interrupt Enable Register	0	0	0	0	0	Enable Rx Line status Interrupt	Enable THRE Interrupt	Enable Rx Data Available Interrupt	อ่าน/เขียน	0	0xE000C004 DLAB=0
U0IIR	Interrupt ID Register	FIFOs Enable		0	0	IIR3	IIR2	IIR1	IIR0	อ่าน	0x01	0xE000C008
U0FCR	FIFO Control Register	Rx Trigger		สงวนไว้		-	Tx FIFO Reset	Rx FIFO Reset	FIFO Enable	เขียน	0	0xE000C008
U0LCR	Line Control Register	DLAB	Set Break	Stick Parity	Even Parity Select	Parity Enable	Number of Stop Bit	World Length Select		อ่าน/เขียน	0	0xE000C00C
U0LSR	Line Status Register	Rx FIFO Error	TEMT	THRE	BI	FE	PE	OE	DR	อ่าน	0X60	0xE000C014
U0SCR	Scratch Pad Register	MSB LSB								อ่าน/เขียน	0	0xE000C01C
U0DLL	Divisor Latch LSB	MSB LSB								อ่าน/เขียน	0x01	0xE000C000 DLAB=1
U0DLM	Divisor Latch MSB	MSB LSB								อ่าน/เขียน	0	0xE000C004 DLAB=1

ตารางที่ 2.2.24 ค่าประจำบิตของรีจิสเตอร์ UART0 Line Control Register(U0LCR)

ค่าของบิต U0LCR	หน้าที่	ความหมาย	ค่าหลังรีเซ็ต
1:0	World Length Select	ตัวอักษรขนาด 5 บิต ตัวอักษรขนาด 6 บิต ตัวอักษรขนาด 7 บิต ตัวอักษรขนาด 8 บิต	0
2	Stop Bit Select	0 Stop bit 1 บิต 1 Stop Bit 2 บิต(1.5บิต ถ้า U0LCR[1:0]=00)	0
3	Parity Enable	0 ยกเลิกการเพิ่มพาริตี้บิตและการตรวจพาริตี้ 1 ใช้งานการเพิ่มพาริตี้บิตและการตรวจพาริตี้	0
5:4	Parity Select	Odd parity Even Parity ให้พาริตี้บิตมีค่าเป็น 1 ตลอดเวลา ให้พาริตี้บิตมีค่าเป็น 0 ตลอดเวลา	0
6	Break Control	ยกเลิกการหยุดการสั่ง อนุญาตให้หยุดการส่งให้ ขาที่เอาต์พุตของ UART0 TxD จะมีค่าลอจิก 0	0
7	Divisor Latch Access Bit	ไม่อนุญาตให้แก้ไขค่าตัวหาร Divisor Latch อนุญาตให้แก้ไขค่าตัวหาร Divisor Latch	0

ตารางที่ 2.2.25 แสดงค่าประจำบิตของ UART0 Line Status Register (U0LSR)

ค่าบิตของ U0LSR	หน้าที่	ความหมาย	ค่าหลังรีเซต
0	Receiver Data Ready (RDR)	0 : U0RBR ว่าง 1 : U0RBR มีรับข้อมูลที่ถูกต้อง U0LSR0 เป็น 1 เมื่อ U0RBR มีค่าตัวอักษรที่ยังไม่อ่าน และถูกล้างค่าเมื่อ UART0 RBR FIFO ว่าง	0
1	Overrun Error (OE)	0 : ไม่มี Overrun error 1 : มี Overrun error เกิดขึ้น Overrun error เกิดขึ้นเมื่อ UART0 RSR ได้รับตัวอักษรตัวใหม่ในขณะที่บัฟเฟอร์ FIFO เต็ม ในกรณีนี้ข้อมูลตัวใหม่ใน UART0 RSR จะสูญหายไป เมื่ออ่านค่าของ U0LSR จะล้างค่าบิตนี้	0
2	Parity Error (PE)	0 : ไม่มี Parity Error 1 : มี Parity Error เกิดขึ้น ใช้ตรวจสอบว่าข้อมูลที่รับมาใน UART0 RBR FIFO มีการผิดพลาดที่พาริตี้บิตหรือไม่ เมื่ออ่านค่าของ U0LSR จะล้างค่าของบิตนี้	0
3	Framing Error (FE)	0 : ไม่มี Framing Error 1 : มี Framing Error เกิดขึ้น เมื่อ Stop bit ของข้อมูลที่รับมามีค่าเป็น 0 แสดงว่าเกิดการผิดพลาดที่เฟรมข้อมูลในขณะที่เกิดการผิดพลาดที่เฟรมข้อมูลนี้ UART0 จะพยายามรับข้อมูลโดยนำ Stop bit ที่ผิดพลาดนี้มาใช้เป็น Start bit ข้อมูลตัวถัดไป ซึ่งอาจจะอ่านข้อมูลได้ถูกต้องหรือไม่ถูกต้อง	0
4	Break Interrupt (BI)	0 : ไม่ใช้ Break Interrupt 1 : ใช้ Break Interrupt เมื่อ RxD0 ได้รับข้อมูลที่เป็น 0 ทุกบิต (start, data, parity, stop) จะเกิด Break Interrupt ภาครับจะหยุดทำงาน จนกว่าจะได้รับข้อมูลที่เป็น 1 ทุกบิตอีกครั้งหนึ่ง	0

ค่าบิตของ U0LSR	หน้าที่	ความหมาย	ค่าหลังรีเซต
5	Transmitter Holding Register Empty (THRE)	0 : U0THR มีข้อมูลที่ถูกต้อง 1 : U0THR ว่าง บิต THRE จะถูกเซ็ตค่าเป็น 1 ทันทีที่พบว่า UART0 THR ว่าง และถูกล้างค่าทันทีมีการเขียนค่าไปยัง U0THR	1
6	Transmitter Empty (TEMT)	0 : U0THRและ/หรือ U0TSR มีข้อมูลที่ถูกต้อง 1 : U0THR และ U0TSR ว่าง บิตนี้จะถูกล้างค่าเมื่อ U0THR หรือ U0TSR มีข้อมูลที่ถูกต้อง	1
7	Error in Rx FIFO (RXFE)	0 : ข้อมูลที่เก็บใน UART0 RBR FIFO ไม่ผิดพลาด 1 : ข้อมูลที่เก็บใน UART0 RBR FIFO อย่างน้อยหนึ่งตัวมีการผิดพลาด	0

ตารางที่ 2.4.1 สรุปข้อมูลสำคัญของการติดต่อกับ SD การ์ดทั้งแบบบัส SD และ SPI

การติดต่อ SD การ์ดด้วยบัส SD	การติดต่อ SD การ์ดด้วยบัส SPI
ใช้สายสัญญาณ 6 เส้น สัญญาณนาฬิกา สัญญาณคำสั่ง (Command) สัญญาณข้อมูล 4 สาย	ใช้สายสัญญาณอนุกรม 3 เส้น สัญญาณนาฬิกา สัญญาณข้อมูลเข้า(DI) สัญญาณข้อมูลออก(DO) สัญญาณเลือกการ์ดCS
มีการป้องกันความผิดพลาดในการถ่ายทอดข้อมูล	สามารถเลือกหรือไม่เลือกการป้องกันความผิดพลาดในการถ่ายทอดข้อมูล
สามารถถ่ายทอดข้อมูลได้ทั้งแบบล็อกเดี่ยวหรือหลายบล็อก	สามารถถ่ายทอดข้อมูลได้ทั้งแบบบล็อกเดี่ยวหรือหลายบล็อก

ตารางที่ 2.4.2 เป็นการจัดขาเมื่อติดต่อ SD การ์ดด้วยบัส SD

หมายเลขขา	ชื่อขาสัญญาณ	ชนิด	คำอธิบาย
1	CD/DAT3	อินพุต/เอาต์พุต	ตรวจสอบการ์ด/สายข้อมูลบิต 3
2	CMD	อินพุต/เอาต์พุต	สัญญาณคำสั่ง/ตรวจสอบการตอบสนอง
3	Vss	สายแหล่งจ่ายไฟ	กราวด์
4	VDD	สายแหล่งจ่ายไฟ	ไฟเลี้ยง
5	CLK	อินพุต	สัญญาณนาฬิกา
6	Vss	สายแหล่งจ่ายไฟ	กราวด์
7	DAT0	อินพุต/เอาต์พุต	สายข้อมูลบิต0
8	DAT1	อินพุต/เอาต์พุต	สายข้อมูลบิต1
9	DAT2	อินพุต/เอาต์พุต	สายข้อมูลบิต2

ตารางที่ 2.4.3 เป็นการจัดขาเมื่อติดต่อ SD การ์ดด้วยบัส SD

หมายเลขขา	ชื่อขาสัญญาณ	ชนิด	คำอธิบาย
1	CS	อินพุต	สัญญาณเลือกติดต่อ(ลอจิก “0”)
2	DI	อินพุต	สัญญาณข้อมูลเข้าจากโฮสต์
3	Vss1	สายแหล่งจ่ายไฟ	กราวด์
4	VDD	สายแหล่งจ่ายไฟ	ไฟเลี้ยง
5	CLK	อินพุต	สัญญาณนาฬิกา
6	Vss2	สายแหล่งจ่ายไฟ	กราวด์
7	DO0	เอาต์พุต	สัญญาณข้อมูลส่งออกจากการ์ด
8	RSV	อินพุต	สำรองไว้
9	RSV	อินพุต	สำรองไว้

ตารางที่ 2.4.4 การแสดงรีจิสเตอร์ใน SD การ์ด

ชื่อริจิสเตอร์	ขนาด	รายละเอียด
OCR	32 บิต	รีจิสเตอร์เก็บสภาวะการทำงาน(Operation Condition Register)
CID	128 บิต	รีจิสเตอร์เก็บค่ารหัสเฉพาะตัวของ SD การ์ด (Card Identification number)
CSD	128 บิต	รีจิสเตอร์เก็บข้อมูลคุณสมบัติเฉพาะของ SD การ์ด (Card Specific Data)
RCA	16 บิต	รีจิสเตอร์กำหนดค่าแอดเดรสแบบสัมพัทธ์ (Relative Card Address) สามารถกำหนดได้จากโฮสต์ไม่ใช่ในกรณีติดต่อ SD การ์ดในโหมด SPI
SCR	64 บิต	รีจิสเตอร์เก็บค่าคุณสมบัติพิเศษของ SD การ์ด(SD Configuration Register) เป็นรีจิสเตอร์พิเศษ ไม่มีใช้ใน MMC (เนื่องจาก MMC มีการติดต่อคล้ายกับ SD การ์ดมาก ดังนั้นใน MMC จะมีรีจิสเตอร์ 4 ตัว ให้ใช้งาน)
DSR	16 บิต	รีจิสเตอร์เสริมสำหรับเก็บค่าคุณสมบัติของไดรเวอร์ทางเอาต์พุต (Driver Stage Register) - ใช้กับSDIO การ์ด

ตารางที่ 2.4.5 แสดงสายสัญญาณของการติดต่อกับSD การ์ดทั้งแบบผ่านบัส SD และ SPI

ขา	การติดแบบบัส SD			การติดแบบบัส SPI		
	ชื่อขา	**ชนิดวงจร**	**คำอธิบาย**	**ชื่อขา**	**ชนิดวงจร**	**คำอธิบาย**
1	CD/DAT3	I/O และ พุชพูล	ตรวจสอบการมีอยู่ของการ์ด/สายข้อมูล DAT3	CS	อินพุต	สัญญาณเลือกกราวด์ (Chip select)
2	CMD	พุชพูล	สัญญาณคำสั่ง สัญญาณตอบสนอง	DI	อินพุต	สายสัญญาณข้อมูลเข้า
3	VSS1	อินพุต กราวด์	กราวด์	VSS	อินพุต กราวด์	กราวด์
4	VDD	อินพุต ไฟเลี้ยง	ไฟเลี้ยง	VDD	อินพุต ไฟเลี้ยง	ไฟเลี้ยง
5	CLK	อินพุต	สัญญาณนาฬิกา	SCK	อินพุต	สัญญาณนาฬิกา
6	VSS2	อินพุต กราวด์	กราวด์	VSS2	อินพุต กราวด์	กราวด์
7	DAT0	I/O และ พุชพูล	สายข้อมูล DAT0	D0	อินพุต พุชพูล	สายสัญญาณข้อมูลออก
8	DAT1	I/O และ พุชพูล	สายข้อมูล DAT1	RESERVE		สำรองไว้
9	DAT2	I/O และ พุชพูล	สายข้อมูล DAT2	RESERVE		สำรองไว้

ตารางที่ 2.4.6 รายละเอียดขาสัญญาณต่างๆ เมื่อใช้การเชื่อมต่อในโหมด SD MODE

SPI Mode			
1	CS	I	Chip Select (active low)
2	DataIn	I	Host-to-card Commands and Data
3	V_{SS1}	S	Supply voltage ground
4	V_{DD}	S	Supply voltage
5	CLK	I	Clock
6	V_{SS2}	S	Supply voltage ground
7	DataOut	O	Card-to-host Data and Status
8	RSV[4]	---	Reserved
9	RSV[5]	---	Reserved

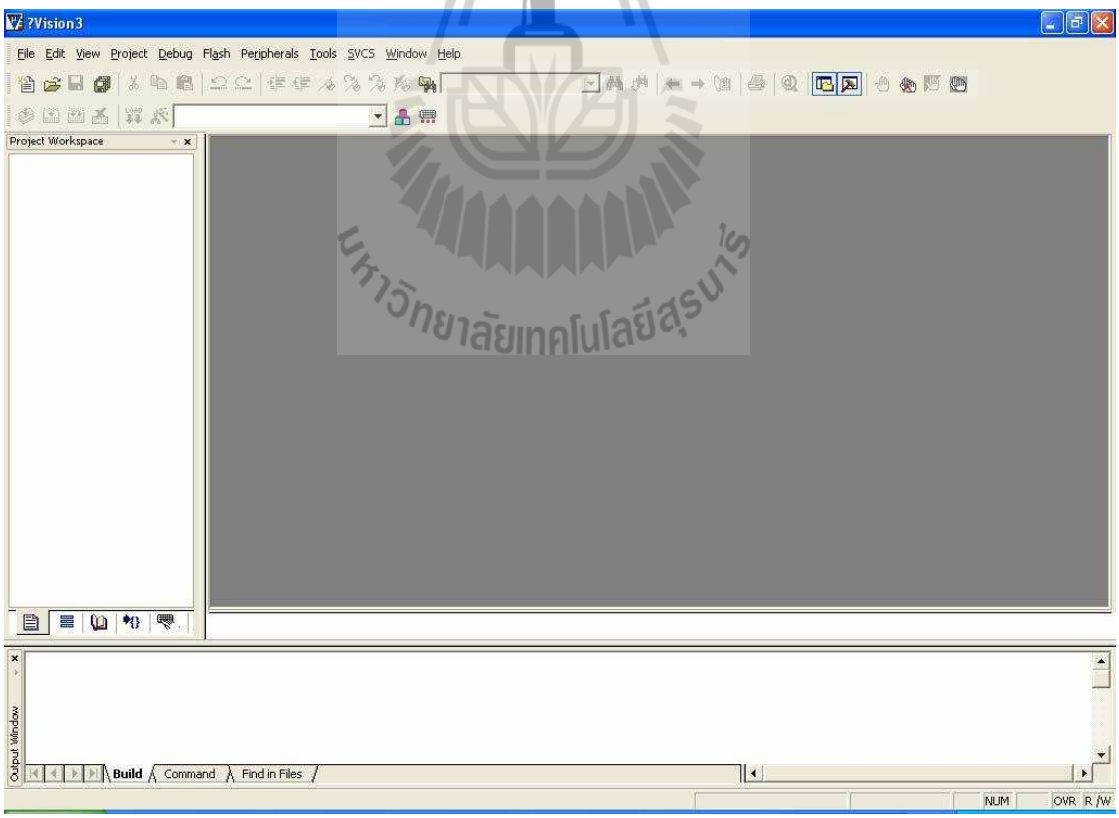
ตารางที่ 2.4.7 รายละเอียดขาสัญญาณต่างๆ เมื่อใช้การเชื่อมต่อในโหมด SPI MODE

Pin No.	Name	Type[1]	Description
SD Mode			
1	CD/DAT3[2]	I/O[3], PP	Card detect/Data line [Bit 3]
2	CMD	I/O, PP	Command/Response
3	V_{SS1}	S	Supply voltage ground
4	V_{DD}	S	Supply voltage
5	CLK	I	Clock
6	V_{SS2}	S	Supply voltage ground
7	DAT0	I/O, PP	Data line [Bit 0]
8	DAT1	I/O, PP	Data line [Bit 1]
9	DAT2	I/O, PP	Data line [Bit 2]

การใช้ Keil uVision3 ในการสร้าง Project File ของ Keil ARM

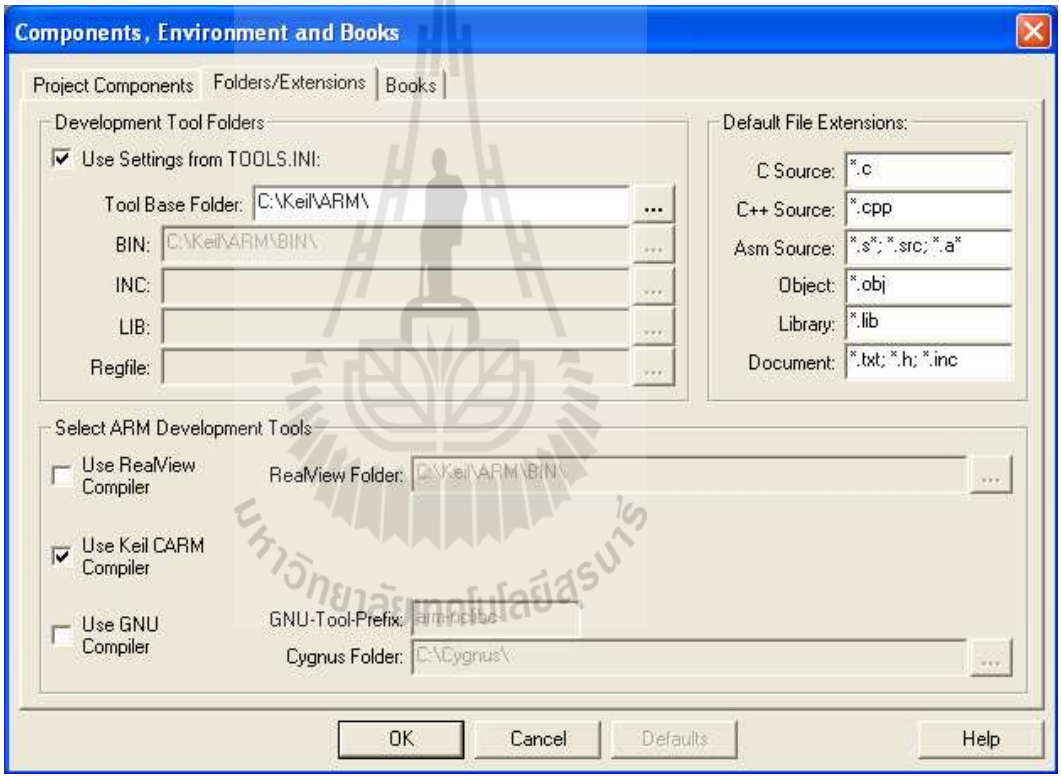
ในที่นี้จะขอแสดงแนวทางการเขียนโปรแกรมภาษาซีโดยใช้ Keil-CARMในการแปลคำสั่งภายใต้โปรแกรม Text Editor ของ Keil (Keil uVision3) โดยจะขออธิบายถึงเฉพาะวิธีการกำหนดค่า Option สำหรับเชื่อมโยงคำสั่งในการสั่งแปลโปรแกรมด้วย Keil-CARM ผ่านทาง Keil uVision3 เท่านั้น ส่วนรายละเอียดคำสั่งและการใช้งานฟังก์ชันต่างๆในการเขียนโปรแกรมของ Keil-CARM นั้นขอให้ผู้ใช้ศึกษาจากคู่มือคำสั่งของ Keil-CARM เอง โดยวิธีการกำหนดค่าตัวเลือกของ Keil uVision3 ให้ใช้งานกับ Keil-CARM นั้นมีขั้นตอนพอสังเขปดังนี้คือ

1. เปิดโปรแกรม Keil uVision3 ซึ่งเป็นโปรแกรม Text Editor ของ Keil-CARM ใช้สำหรับใช้ในการเขียนโปรแกรมที่เป็น Source Code ภาษาซี โดยจะมีลักษณะดังรูป



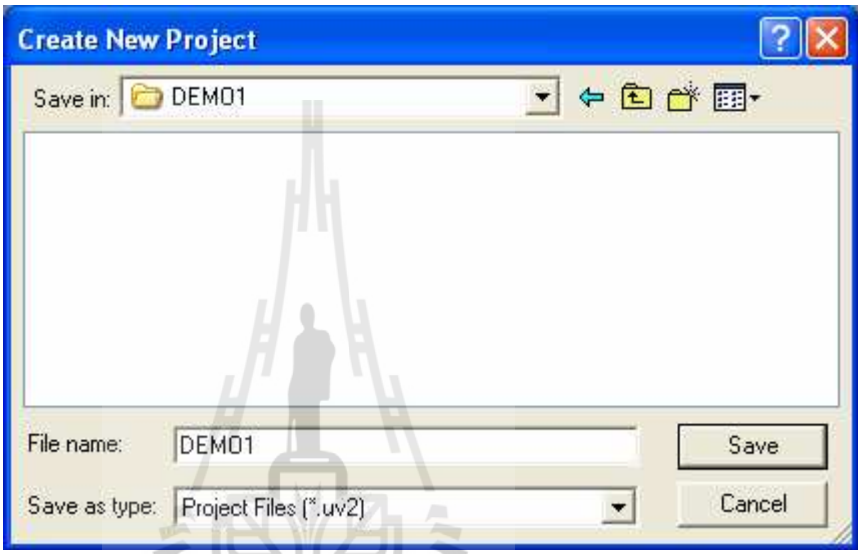
รูปที่ 5.1 หน้าต่างของโปรแกรม Keil uVision3

2. ทำการกำหนดค่าตัวเลือกในการแปลคำสั่งของ uVision3 ให้ใช้งานกับโปรแกรม Keil uVision3 และ Keil-CARM โดยให้เลือกคลิกเมาส์ที่เมนูคำสั่ง Project ➔ Components ,Environment, Books… จากนั้นให้เลือกค่าตัวเลือกสำหรับกำหนดการใช้งาน Complier จากหัวข้อ Select ARM Development Tools ซึ่งจะมีค่าตัวเลือกอยู่ 3 แบบ คือ Use Keil-CARM Tools ,Use GNU Tools และ Use ARM Tools โดยให้เลือกเป็น “Use Keil ARM Tools” จากนั้นให้ทำการกำหนดตำแหน่ง Folder สำหรับเก็บค่าตัวเลือกการทำงานของโปรแกรม Keil ARM ซึ่งตามปรกติแล้วจะเป็น “C:\Keil\ARM\” แต่ถ้าติดตั้ง Keil ไว้ที่อื่นก็ต้องปรับเปลี่ยนให้ถูกต้องตามความเป็นจริงด้วยดังรูป



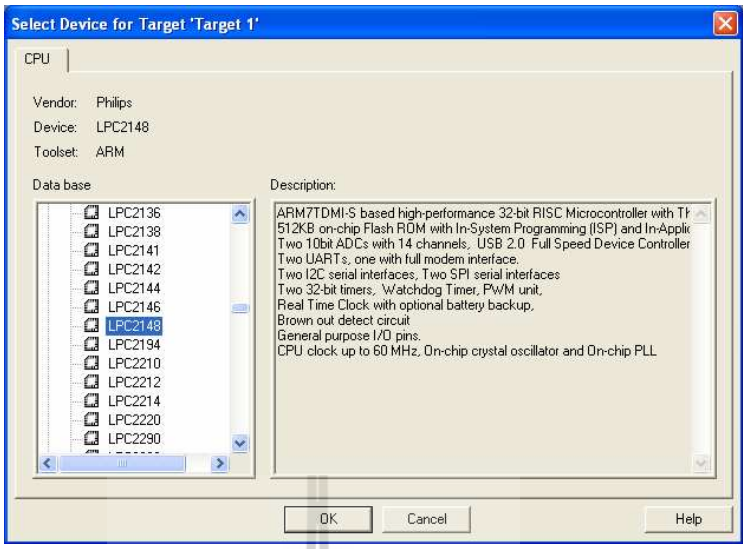
รูปที่ 5.2 การกำหนดค่าตัวเลือกในการแปลคำสั่งของ uVision3

3.ทำการสร้าง Project File ขึ้นมาใหม่ โดยเรียกเมนูคำสั่ง Project ➔ New Project จากนั้นให้เลือกกำหนดหรือสร้างตำแหน่ง Folder ที่จะบันทึก Project File พร้อมกับกำหนดชื่อ Project File ตามต้องการ เช่น ถ้าต้องการสร้างProject File ชื่อ DEMO1 โดยเก็บไว้ใน Folder ชื่อ DEMO1 ก็สามารถกำหนดตำแหน่ง Folder และชื่อ Project Fileได้เอง โดยเมื่อกำหนดชื่อในช่อง File name แล้วให้เลือก Save เพื่อบันทึก Project File ไว้ ดังรูป



รูปที่ 5.3 หน้าต่างของการสร้าง Project ใหม่

หลังจากกำหนดชื่อและสั่ง Save Project File แล้ว โปรแกรมจะรอให้ผู้ใช้ทำการกำหนดเบอร์ MCU ที่จะใช้งานใน Project ที่สั่ง Save นั้น ซึ่งในกรณีที่ใช้งานกับบอร์ด CP-JR ARM7 USB-LPC2148 นั้น ให้เลือกกำหนดเบอร์ของ MCU เป็น LPC2148 ของ Philips แล้วเลือก OK ดังรูป



รูปที่ 5.4 การกำหนดเบอร์ MCU ที่จะใช้งาน

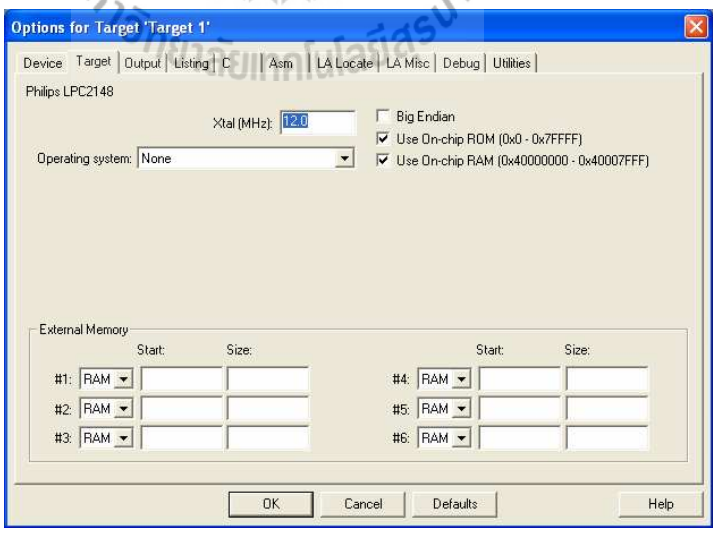
หลังจากเลือกกำหนดเบอร์ของ MCU เป็นที่เรียบร้อยแล้ว ในขั้นตอนนี้โปรแกรมจะรอให้ผู้ใช้ยืนยันว่าต้องการจะทำการ Copy ไฟล์ Startup ของ Keil เพื่อใช้งานกับ MCU ของ Philips มาใช้ใน Project ด้วยหรือไม่ โดยStartup ไฟล์จะเป็นส่วนของการกำหนดค่าเริ่มต้นการทำงานให้กับ MCU เช่น การกำหนดค่า Stack และการกำหนดค่าการทำงานให้กับ Phase-Lock-Loop ต่างๆ ก่อนที่จะเริ่มต้นทำงานตามโปรแกรมที่เราเขียนขึ้น ไม่เช่นนั้นแล้วโปรแกรมที่เราเขียนขึ้นมานั้นจะต้องเพิ่มคำสั่งในการเตรียมการทำงานส่วนเหล่านี้ให้ MCU เองทั้งหมดแต่เนื่องจากไฟล์ Startup ของ Keil-ARM นั้น เป็นไฟล์ภาษา แอสแซมบลี้(Assembly) ซึ่งกำหนดค่าการทำงานไว้กับชุดพัฒนาของ Keil เอง ดังนั้นข้อกำหนดและการกำหนดค่าบางอย่างจะมีความแตกต่างกันอยู่กับค่าที่ต้องการสำหรับบอร์ด “CP-JR ARM7 USB-LPC2148” ไม่สามารถใช้งานไฟล์ Startup ได้ทันที ต้องมีการแก้ไขค่าตัวเลือกใหม่ดังนั้นก่อนที่จะใช้โปรแกรม Keil-CARM ในการแปลคำสั่งให้นั้น ผู้ใช้จะต้องเข้าไปแก้ไขไฟล์ Startup ใหม่โดยต้องกำหนดรูปแบบให้ถูกต้องตรงกับความต้องการของบอร์ดด้วย ดังนั้นในที่นี้ขอแนะนำให้เลือก “No” เพื่อไม่ให้ Keil uVision3 ทำการ Copy ไฟล์ Startup ของ Keil-CARM มาใช้ใน Project นี้ด้วย

รูปที่ 5.5 ข้อความการกำหนด Startup File

4. ให้ทำการ Copy File ชื่อ “Startup.s” ที่ทาง ผู้จัดทำจัดเตรียมไว้ใน CD-ROM ชื่อ “Startup.s” มาไว้ในตำแหน่ง Folder เดียวกันกับ Project File ที่สร้างขึ้นมาใหม่นี้โดยไฟล์ “Startup.s” จะเป็นไฟล์ซึ่งบรรจุคำสั่งภาษาแอสแซมบลี้ของ ARM7 สำหรับทำหน้าที่กำหนดค่าเริ่มต้นการทำงานที่จำเป็นให้กับ MCU ซึ่งได้แก่การ กำหนดค่า Stack ให้กับ MCU การ Initial Phase-Lock-Loopการกำหนดค่าให้กับ MAM Function และการกำหนดตำแหน่ง Vector ต่างๆของ MCU สำหรับใช้งานร่วมกับบอร์ด “CP-JR ARM7 USB-LPC2148” ซึ่งถ้าสั่ง Add ไฟล์ “Startup.s” จาก Keil หรือ Copy ไฟล์ดังกล่าวมาจากแหล่งอื่นๆ อาจมีการทำงานของโปรแกรมใน Startup ไม่เหมือนกัน ซึ่งจะส่งผลต่อการทำงานของโปรแกรมด้วย

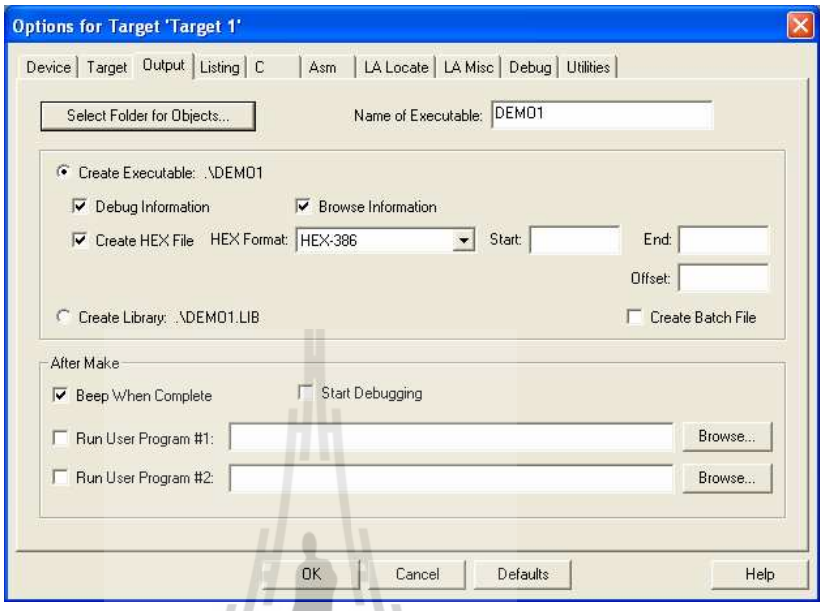
5. ให้ทำการกำหนดค่า Option ของ Project File โดยเลือกเมนูคำสั่ง Project ➔ Option for Target ’Target 1’ จากนั้นเลือกที่ Tab ของ Target เพื่อกำหนดค่าของ MCU Target โดยให้กำหนดดังนี้

5.1 X-TAL ให้กำหนดเป็น 12 MHz พร้อมกับเลือกกำหนดให้ใช้หน่วยความจำที่มีอยู่ใน MCU เป็นเงื่อนไข ในการแปลโปรแกรมของ Keil-CARM ด้วย ดังรูป



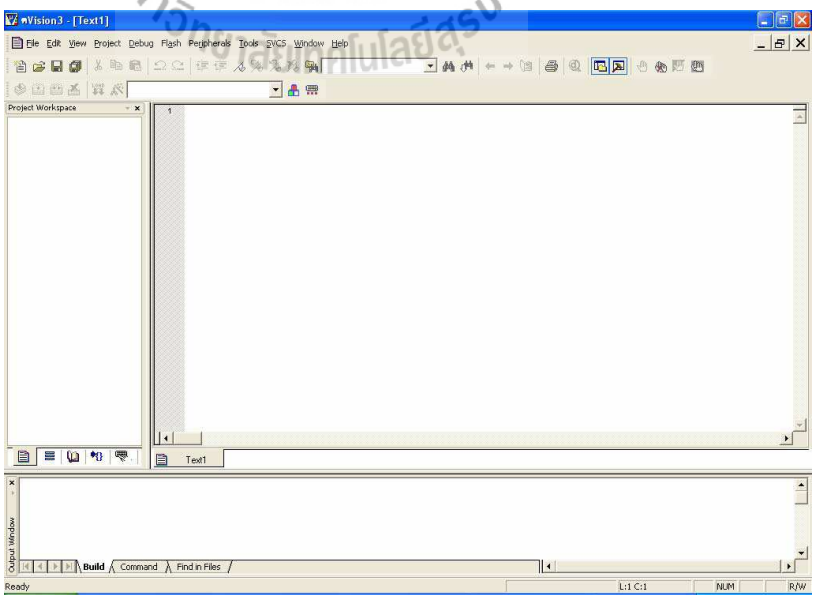
รูปที่ 5.6 การกำหนดค่า XTAL ของ MCU ที่ใช้งาน

5.2 เลือกที่ Output ให้เลือกคลิกเมาส์ที่ค่าตัวเลือก Create HEX File พร้อมกับเลือกกำหนดรูปแบบของ Hex ให้เป็นแบบ HEX-386 แล้วเลือก OK ดังรูป



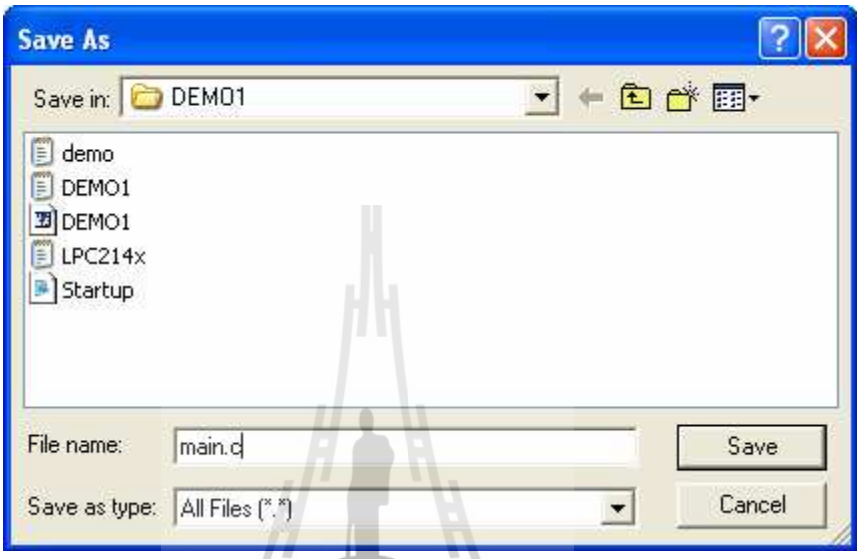
รูปที่ 5.7 การกำหนดค่าตัวเลือก Create HEX File

6. เริ่มต้นเขียน Source Code ภาษาซี โดยให้เลือกคลิกเมาส์ที่เมนูคำสั่ง File ➔ New… ซึ่งจะได้พื้นที่ในการเขียน Text File เกิดขึ้นมา โดยในครั้งแรกจะกำหนดชื่อตามค่า Default เป็น “Text1” ดังรูป



รูปที่ 5.8 หน้าต่างของโปรแกรม Keil uVision3 หลังจากตั้งค่าต่าง ๆแล้ว

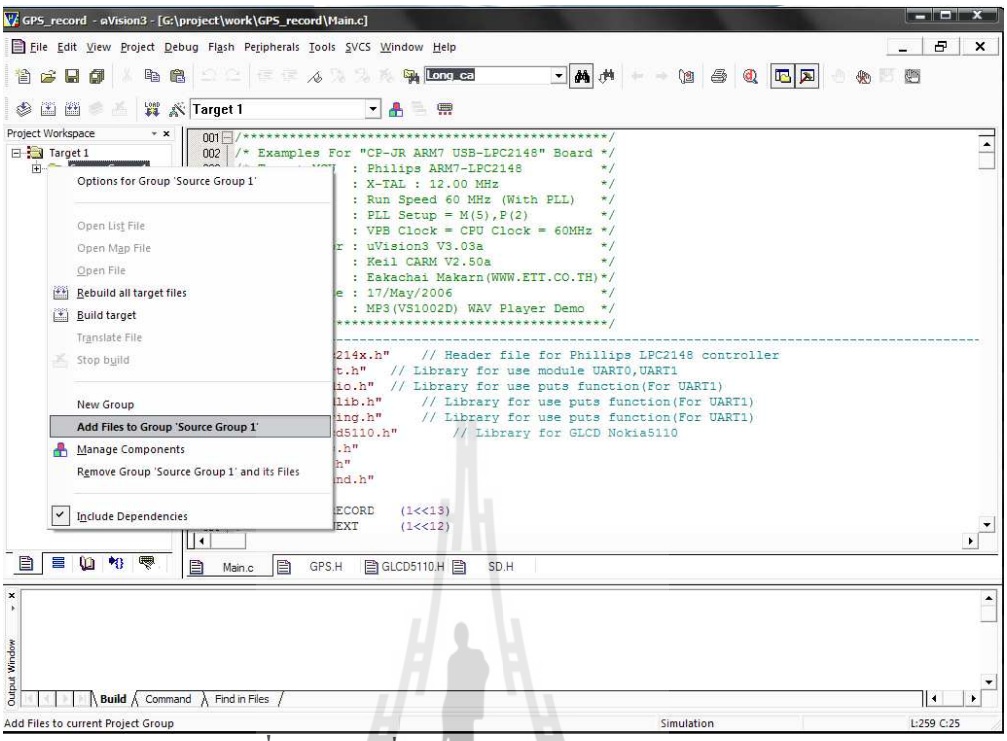
ในขั้นตอนนี้ให้ทำการพิมพ์ Source Code ภาษาซี ตามข้อกำหนดของ Keil-CARM ในพื้นที่เขียนโปรแกรมตามต้องต้องการ หลังจากพิมพ์คำสั่งภาษาซีเสร็จเรียบร้อยตามต้องการแล้ว ให้สั่ง Save ไฟล์ดังกล่าว โดยต้องกำหนดเป็นไฟล์ที่มีนามสกุลเป็น ".C" ในที่นี้ขอแนะนำให้สั่ง Save โดยใช้คำสั่ง File ➔ Save As… แล้วกำหนดชื่อและนามสกุลของไฟล์เป็น .main.c" ดังรูป



รูปที่ 5.9 หน้าต่างของการ Save ไฟล์

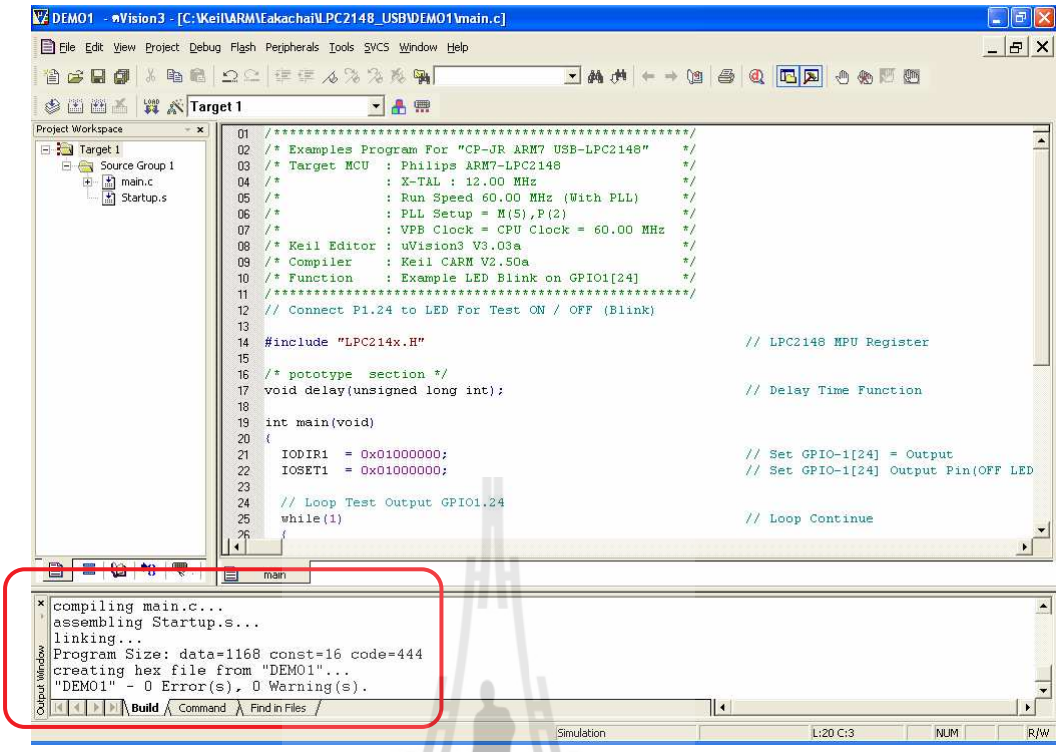
ซึ่งหลังจากที่สั่ง Save ไฟล์เป็น "main.c" แล้วจะเห็นว่าลักษณะสีของตัวอักขระต่างๆในโปรแกรมจะเกิดการเปลี่ยนแปลงไปตามหน้าที่ เช่น Comment, ตัวแปร และ คำสั่ง เป็นต้น ซึ่งส่วนนี้เป็นข้อดีของ Keil uVision3 ซึ่งสามารถแยกและแสดงตัวอักขระได้อย่างเป็นหมวดหมู่ ทำให้ง่ายต่อการอ่านโปรแกรมด้วย

7. ทำการสั่ง Add File ต่างๆเข้ากับ Project File โดยให้เลือกคลิกเมาส์ที่ด้านซ้ายของหน้าต่าง จากนั้นให้เลือกที่ Add Files to Group "Source Group 1 " แล้วเลือกที่ Add File ที่ต้องการจะเพิ่มเข้าไปใช้งานร่วมกับ Project Fileโดยในครั้งแรกให้เลือก Files of type เป็น "C Source files(*.c)" ซึ่งจะปรากฏชื่อไฟล์ต่างๆที่เป็น Source Code ภาษาซีให้เห็น โดยในที่นี้ให้เลือกคลิกเมาส์ที่ไอคอนของไฟล์ชื่อ "main.c" แล้วเลือก Add เพื่อสั่งเพิ่มไฟล์ชื่อ "Startup.s" เข้าไปรวมกับ Project Files ที่เราสร้างไว้



รูปที่ 5.10 การสั่ง Add File ต่างๆเข้ากับ Project File

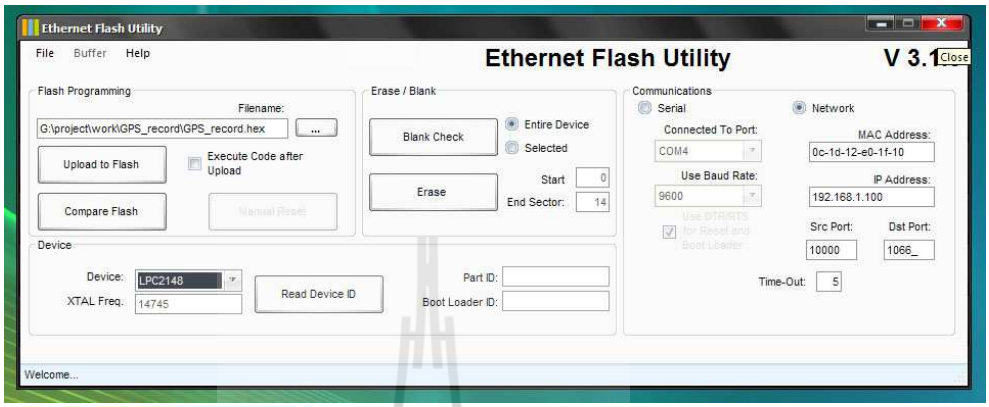
8. ให้ทำการสั่งแปลโปรแกรมที่เราเขียนขึ้นเรียบร้อยแล้ว โดยให้คลิกเมาส์ที่เมนูคำสั่ง Projects ➔ Rebuild all target files ซึ่งโปรแกรม Keil uVision3 จะทำการสั่งให้โปรแกรม Keil-CARM ทำการแปลคำสั่งให้ทันที ซึ่งหลังจากสั่งแปลโปรแกรมแล้วได้ผลถูกต้องและไม่เกิดข้อผิดพลาดใดๆขึ้น (0 Error และ 0 Warning) ก็ได้ Hex File ซึ่งมีชื่อเหมือนกันกับชื่อของ Project File ที่สร้างไว้ ซึ่งผู้ใช้สามารถนำ Hex File ดังกล่าวไปทำการ Download ให้กับ MCU ได้ทันที



รูปที่ 5.11 การแปลงเป็น Hex File

การโหลดโปรแกรมทดสอบ

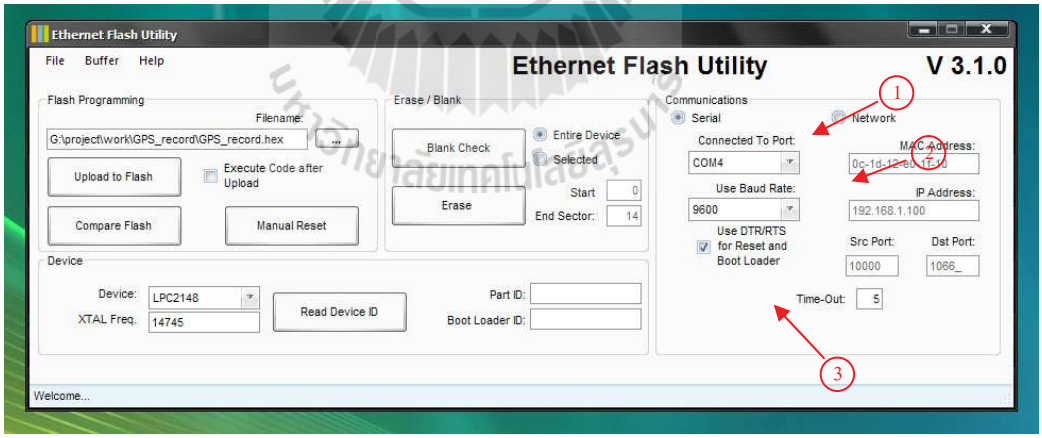
เปิดโปรกแกรม Ethernet Flash Utility จะได้หน้าต่างดังรูป



รูปที่ 5.12 หน้าต่างโปรแกรม Ethernet Flash Utility

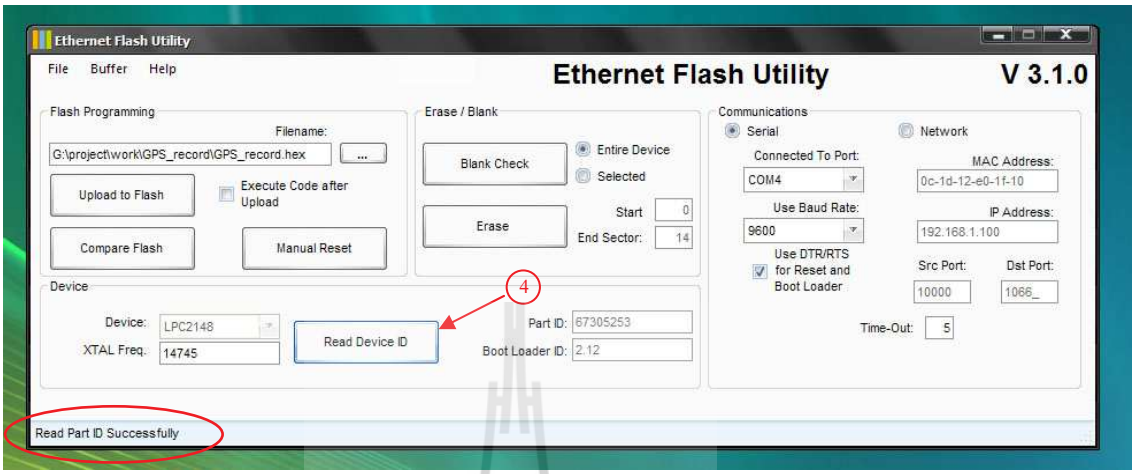
เลือกรูปแบบการเชื่อมต่อเป็นแบบ Serial ➔ เลือก COM ให้ถูกต้อง ➔ Baud Rate 9600

➔ เลือกที่ช่อง Use DTR/RTS for Reset and Boot Loader



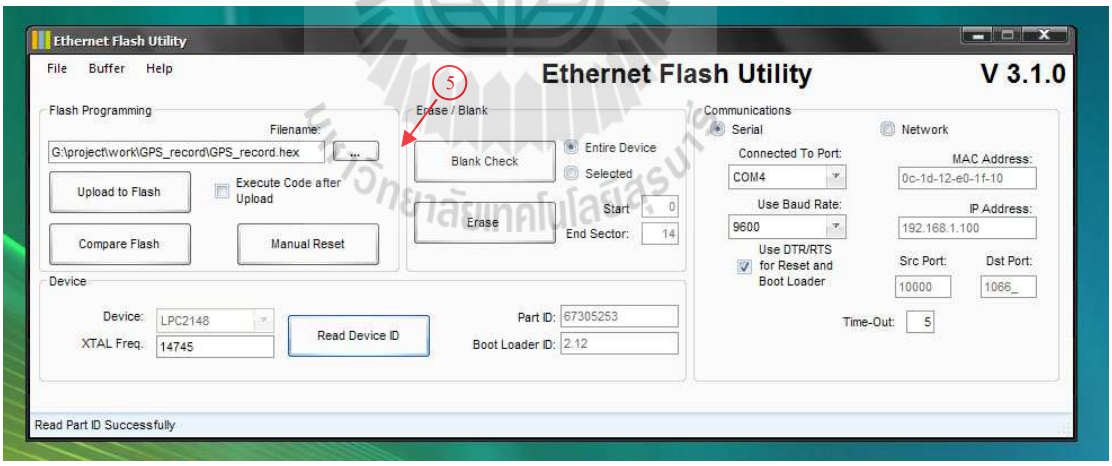
รูปที่ 5.13 หน้าต่างรูปแบบการเชื่อมต่อ

กด Switch Reset ที่ บอร์ดไมโครคอนโทรลเลอร์ แล้วเลือกที่ Read Device ID หาก เชื่อมต่อกับไมโครคอนโทรลเลอร์สำเร็จจะแสดงข้อความ Read Part ID Successfully

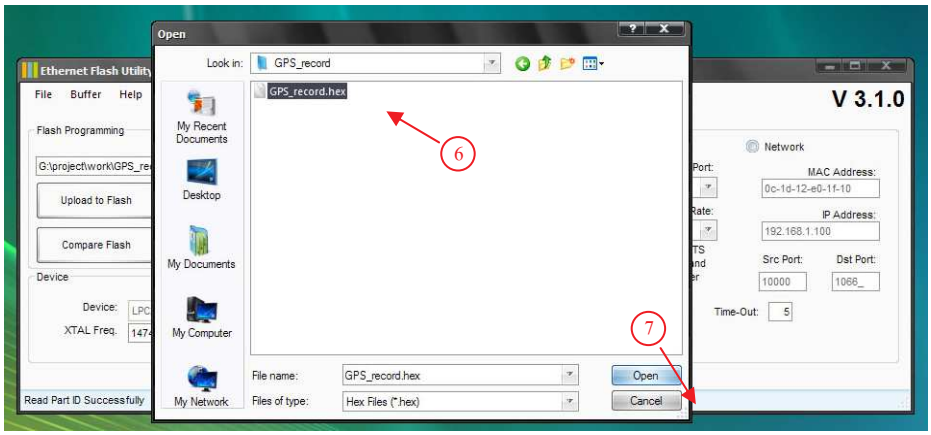


รูปที่ 5.14 หน้าต่างการเชื่อมต่อสำเร็จ

เมื่อเชื่อมต่อเรียบร้อยแล้ว ทำการโหลดไฟล์ .HEX โดยเลือกที่ Brown เลือกไฟล์ .HEX ที่ ต้องการ แล้วเลือก Open ตามรูปที่ 5.15 และรูปที่ 5.16

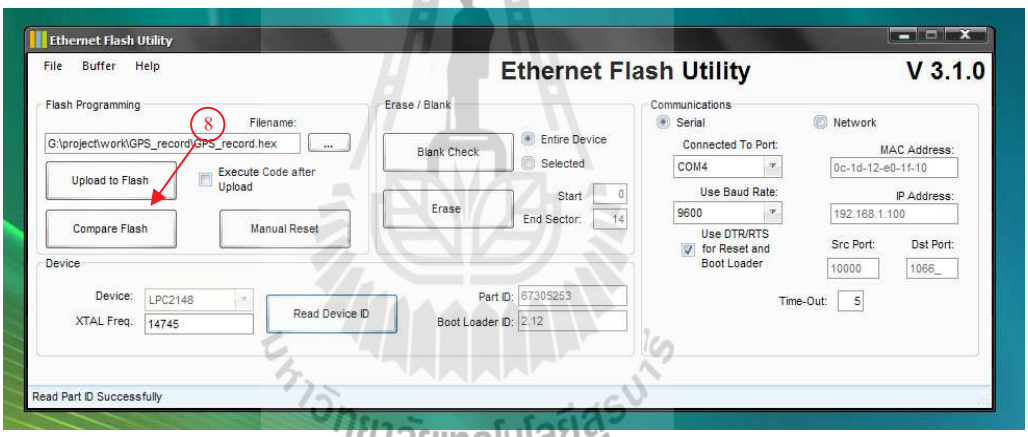


รูปที่ 5.15 หน้าต่างการโหลดไฟล์ .HEX

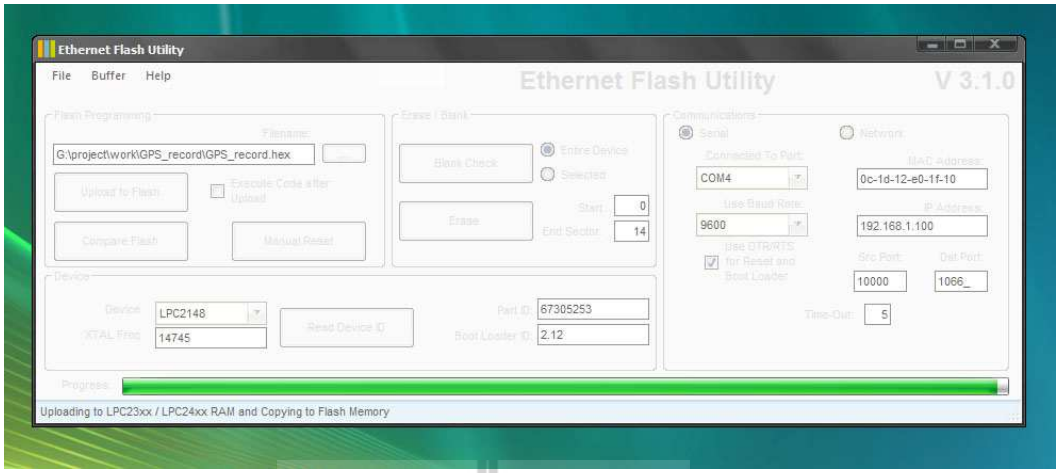


รูปที่ 5.16 หน้าต่างการโหลดไฟล์ .HEX

เลือกที่ Upload to Flash และรอจนกว่าโปรแกรมจะทำการโหลดเสร็จตามรูปที่ 5.17 และรูปที่ 5.18



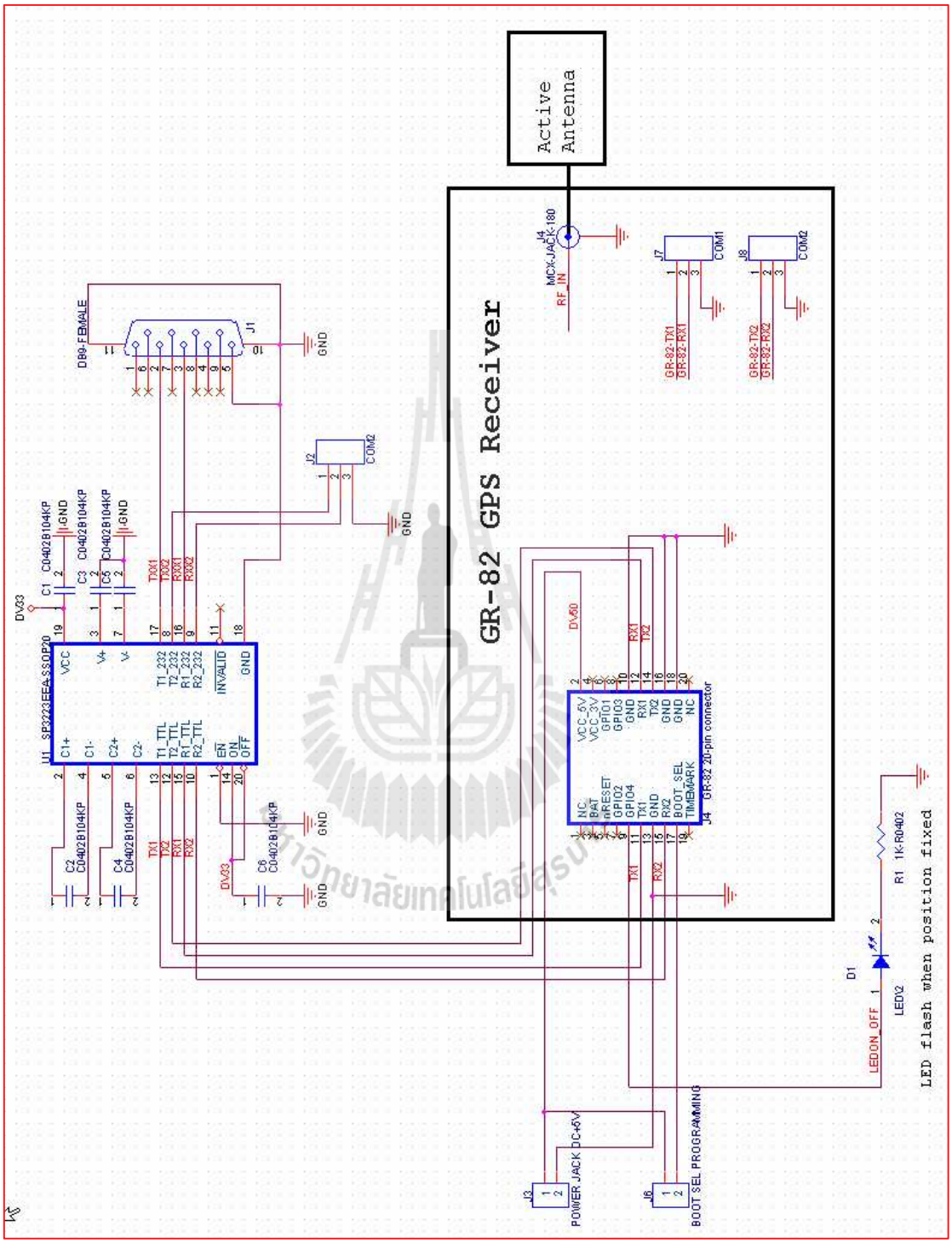
รูปที่ 5.17 หน้าต่างการโหลดไฟล์ .HEX



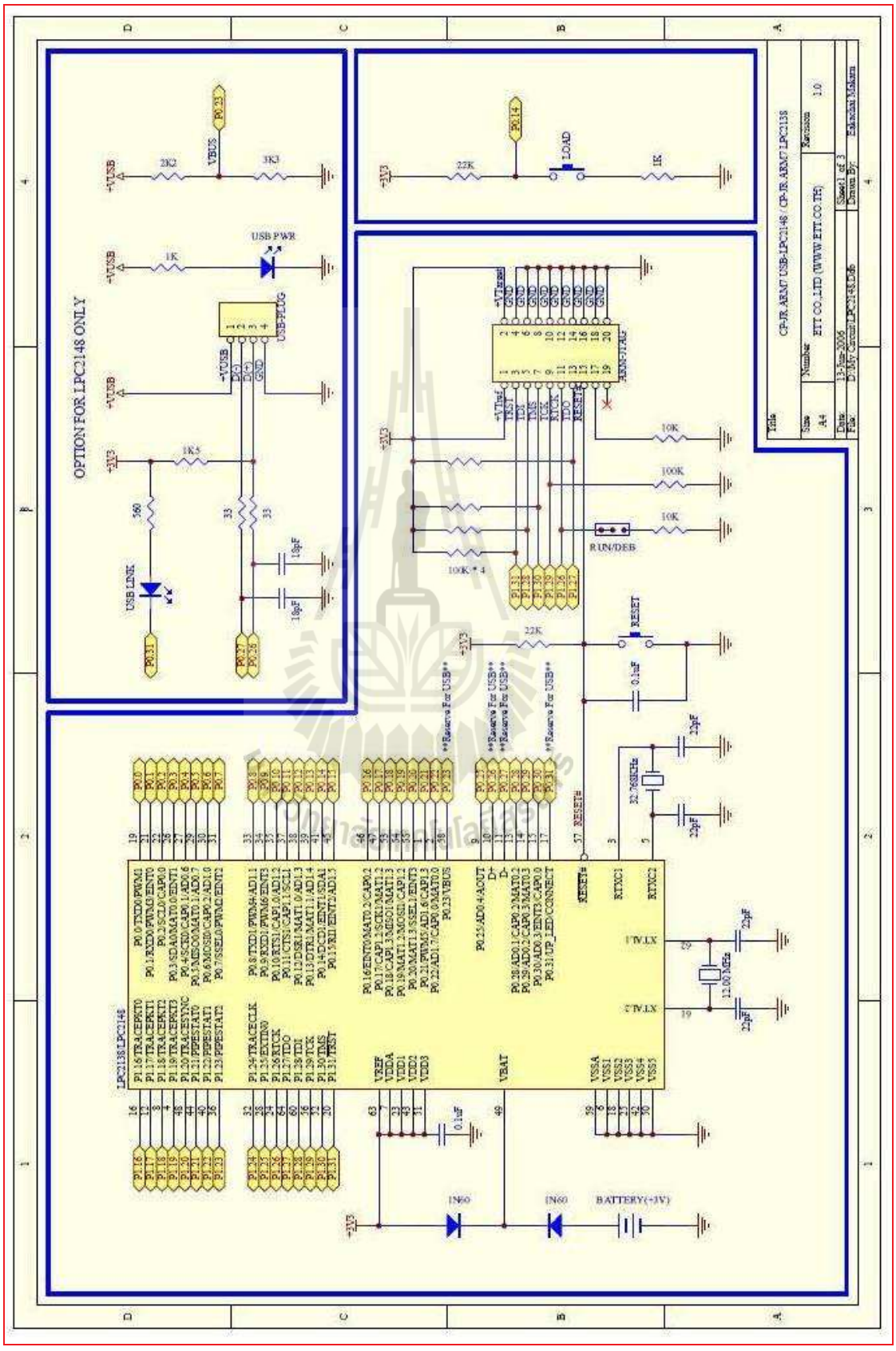
รูปที่ 5.18 หน้าต่างการโหลดไฟล์ .HEX

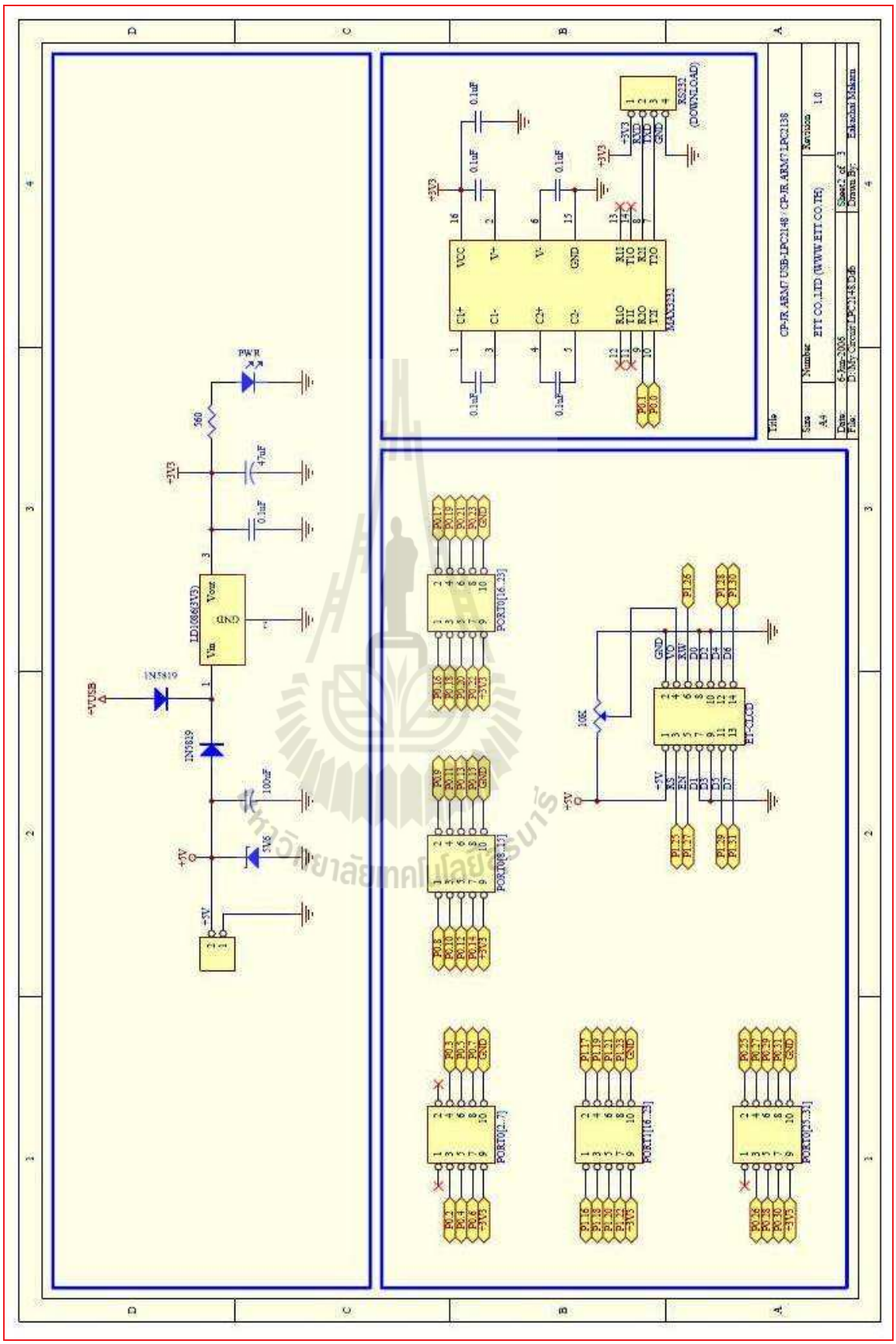
Datasheet

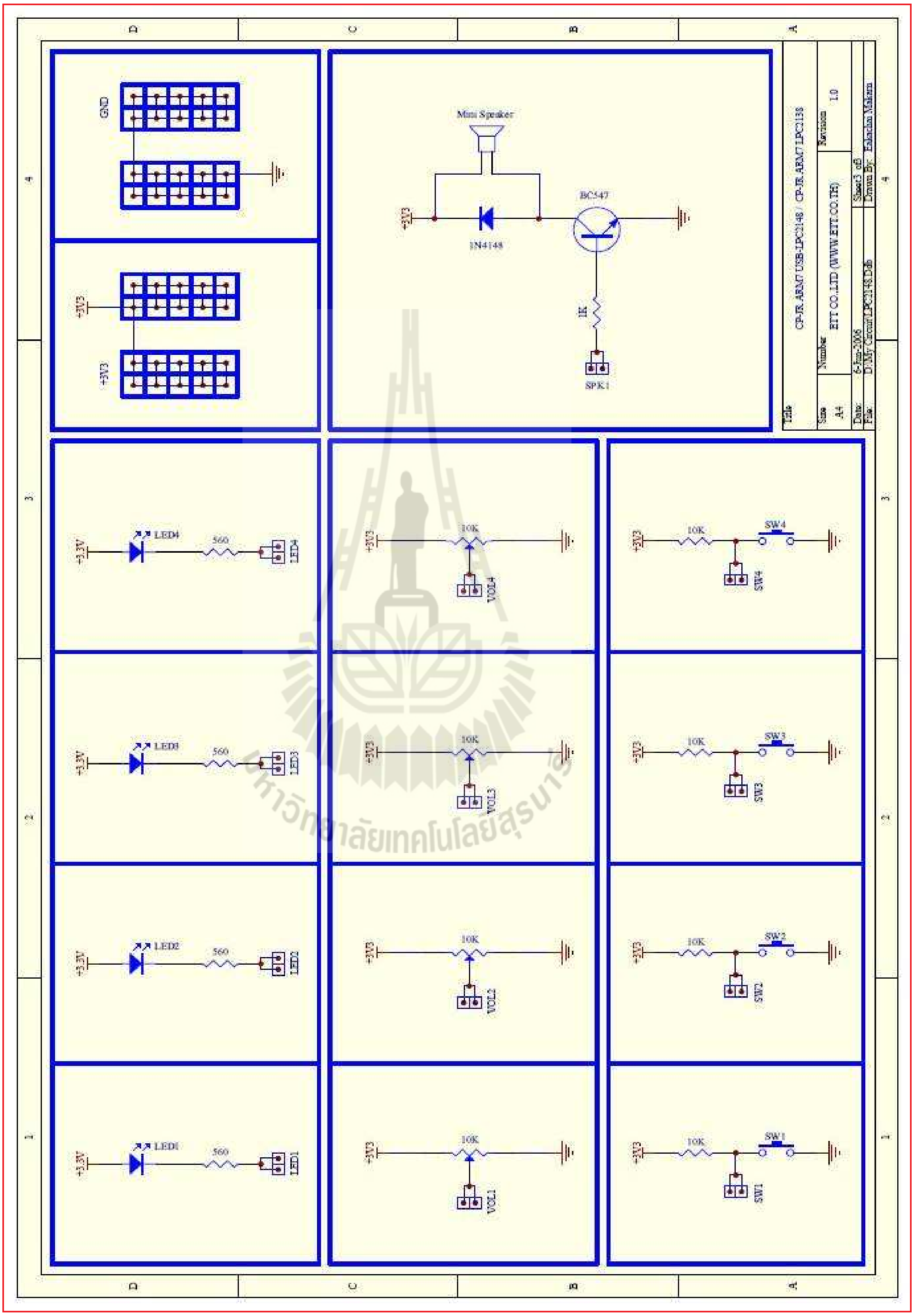
DATASHEET GPS MODULE (GM-82)



DATASHEET MICROCONTROLLER (CP_JR_ARM7_USB_LPC2148)




CP-IR ARM7 USB-LPC2148 / CP-IR ARM7-LPC2138
EIT CO.,LTD (WWW.EIT.CO.TH)
MAX232
LD1086(3V3)
IN5819
PWR
EY-CLCD


Mini Speaker
1N4148
BC547
1K
SPK1
GND
+3V3
LED1
LED2
LED3
LED4
560
10K
VOL1
VOL2
VOL3
VOL4
SW1
SW2
SW3
SW4

DATASHEET VOICE RECORD MODULE