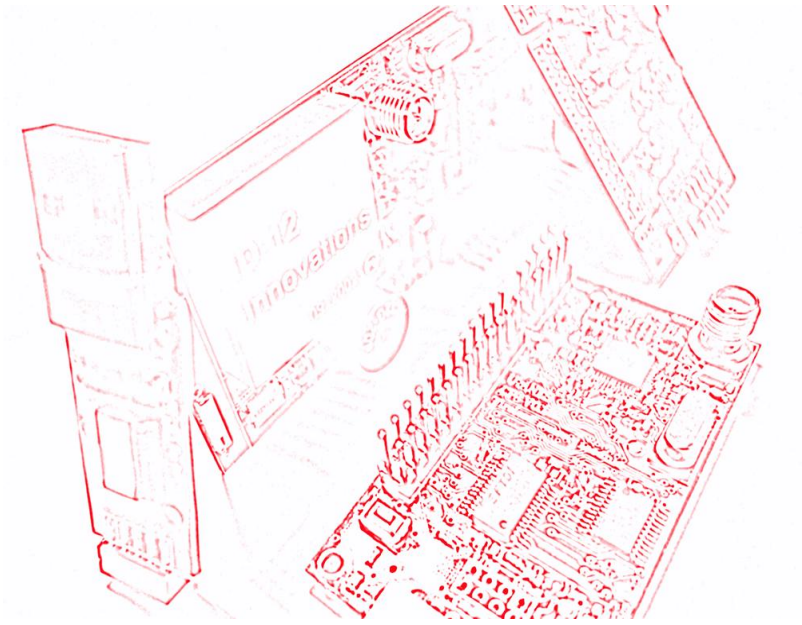
Introduction to Wireless Sensor Networks



W.Suntiamorntut

S.Charoenpanyasak

Department of Computer Engineering, Faculty of Engineering

Prince of Songkla University

Introduction to Wireless Sensor Networks

Theories and Practices

Wannarat Suntiamorntut & Sakuna Charoenpanyasak

Department of Computer Engineering, COE-WSN

Faculty of Engineering

Prince of Songkla University, Thailand

2014

กิตติกรรมประกาศ

ผู้เขียนขอขอบคุณ สมาคมสมองกลฝังตัวไทย (TESA) ที่ริเริ่มโครงการดีๆ ในการเผยแพร่ความรู้ทางด้านเครือข่ายเซนเซอร์ไร้สายซึ่งถือได้ว่าเป็นงานประยุกต์งานหนึ่งทางด้านสมองกลฝังตัว และขอขอบคุณสำนักงานพัฒนาวิทยาศาสตร์และเทคโนโลยีแห่งชาติ (สวทช. หรือ NSTDA) ที่ให้การสนับสนุนโครงการในการเผยแพร่ความรู้ทางด้านสมองกลฝังตัว

นอกจากนี้ผู้เขียนขอขอบคุณบิดา มารดา และครูบาอาจารย์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์ และขอขอบคุณผู้ที่เกี่ยวข้องทุกท่าน มา ณ ที่นี้ด้วย

วรรณรัช สันติอมรทัต สกุณา เจริญปัญญาศักดิ์

คำนำ

หนังสือเล่มนี้เป็นส่วนหนึ่งในโครงการเผยแพร่เอกสารทางด้านเทคโนโลยีสมองกลฝังตัว แก่ผู้ที่สนใจทั่วไปได้สามารถศึกษาหาความรู้ และทดลองปฏิบัติ เพื่อนำไปต่อยอดการสร้างสรรค์ผลงานทางด้านนี้ในประเทศให้มากขึ้น ซึ่งโครงการนี้เป็นการริเริ่มโดยสมาคมสมองกลฝังตัวของไทย ร่วมกับทางสำนักงานพัฒนาวิทยาศาสตร์และเทคโนโลยีแห่งชาติ หรือ สวทช. ผู้เขียนจึงได้มีโอกาสแต่งและเรียบเรียงหนังสือฉบับนี้ขึ้นเพื่อเป็นการถ่ายทอดองค์ความรู้ทั่วไปเกี่ยวกับเทคโนโลยีเครือข่ายเซนเซอร์ไร้สาย และการนำไปประยุกต์ใช้งาน

ก่อนอื่นผู้เขียนขออธิบายโครงสร้าง และการใช้งานหนังสือแนะนำเครือข่ายเซนเซอร์ไร้สายฉบับนี้ไว้ดังนี้ ความตั้งใจในการจัดทำหนังสือทางด้านเครือข่ายเซนเซอร์ไร้สายได้จัดแบ่งไว้เป็น 2 เล่ม สำหรับเล่มแรก เริ่มที่เนื้อหาได้ถูกแบ่งออกไว้เป็น 2 ส่วนคือ ภาคทฤษฎีด้านความรู้ขั้นพื้นฐาน และภาคปฏิบัติขั้นพื้นฐาน ซึ่งในส่วนแรกนั้นจะมีด้วยกัน 5 บท ประกอบด้วย แนะนำให้รู้จักกับเครือข่ายเซนเซอร์ไร้สาย งานประยุกต์ที่นำเทคโนโลยีเครือข่ายเซนเซอร์ไร้สายไปใช้งาน สถาปัตยกรรมและการสร้างโนด (Node) ซึ่งเป็นอุปกรณ์หลักของเครือข่ายเซนเซอร์ไร้สาย จากนั้นให้เข้าใจสถาปัตยกรรมของการนำโนดมาสร้างเป็นเครือข่าย และแนะนำโพรโทคอลพื้นฐานในชั้นเครือข่าย ส่วนในภาคปฏิบัตินั้นแบ่งออกเป็น 3 บท ประกอบด้วย การใช้งานโนด XBee ตามมาตรฐาน Zigbee และการใช้งานโนดที่ compatible กับโนด Telos ดังโครงสร้างที่สรุปไว้ต่อไปนี้

Book I: Introduction to Wireless Sensor Networks

Part I

Chapter 1: Introduction

Chapter 2: WSN Applications

Chapter 3: Node architecture

Chapter 4: Network architecture

Chapter 5: Protocols in Network Layer

Part II

L01: Practice I: Platform ZigBee1

L02: Practice II: Platform ZigBee2

L03: Practice III: Platform Telos compatible.

สำหรับเนื้อหาในหนังสือเล่มที่ 2 ที่จะใช้ชื่อหนังสือว่า ต่อยอดด้วยเทคโนโลยีเครือข่ายเซนเซอร์ไร้สาย (Technologies in Wireless Sensor Networks) จะแบ่งเป็น 4 ส่วน คือ 1) ส่วนของการออกแบบและวิเคราะห์อัลกอริทึมและโพรโทคอลในเครือข่าย เป็นการขยายความความรู้พื้นฐานต่อจากเล่มที่ 1 โดยมีการกล่าวถึงการใช้เครื่องมือจำลองการทำงานเครือข่าย NS-2 มาช่วยในการออกแบบและวิเคราะห์โพรโทคอล 2) รวบรวมหัวข้องานวิจัยในปัจจุบันที่ยังคงน่าสนใจในเครือข่ายเซนเซอร์ไร้สาย และ 3) งานประยุกต์ของเครือข่ายเซนเซอร์ไร้สายที่น่าสนใจในปัจจุบันและอนาคต 4) ภาคปฏิบัติการใช้งาน NS-2 ซึ่งโครงสร้างของหนังสือได้แสดงไว้ด้านล่างนี้

Book II: Technologies in Wireless Sensor Networks

Part I: Algorithms and Protocol Design and Analysis

Chapter 1: MAC Layer

Chapter 2: Network Layer

Chapter 3: Transport & Application Layer

Chapter 4: Cross layer Designs

Part II: Recent Advances and Research Wireless Sensor Networks

Chapter 5: Time Synchronization

Chapter 6: Localization

Chapter 7: Security

Part III: Recent and Future Applications in Wireless Sensor Networks

Chapter 8: Wireless Multimedia Sensor Networks

Chapter 9: Wireless Underwater Sensor Networks

Chapter 10: Wireless Underground Sensor Networks

Chapter 11: Wireless Robot Networks

Part IV

L01: Practice: NS-2

L02: Practice: How to develop protocols in node

ซึ่งหนังสือเล่มที่ 2 นี้จะสามารถหาอ่านได้ ผ่านทางสมาคมสมองกลฝังตัวไทยในปีหน้า สำหรับหนังสือเล่มนี้ทางผู้เขียนได้เขียนและรวบรวมเนื้อหาจากประสบการณ์การวิจัยและพัฒนาเครือข่ายเซนเซอร์ไร้สายที่นำไปใช้งานได้จริงทั้งในฟาร์มกุ้ง แปลงพืชทางการเกษตร ระบบเฝ้าระวังสุขภาพผู้สูงอายุและระบบเตือนภัยน้ำท่วม ที่ได้ผลิตงานร่วมกับนักศึกษาระดับปริญญาตรี โท และเอก จำนวนมากของศูนย์ความรู้เฉพาะด้านเครือข่ายเซนเซอร์ไร้สาย มหาวิทยาลัยสงขลานครินทร์ นอกจากนี้ยังได้รวบรวมและสรุปงานทั้งจากบทความทางวิชาการที่น่าสนใจและหนังสือต่างประเทศ มาย่อไว้ให้ผู้อ่านได้ทำความรู้จักกับเทคโนโลยีของเครือข่ายเซนเซอร์ไร้สาย โดยหวังเป็นอย่างยิ่งว่าผู้อ่านจะได้รับประโยชน์ไม่มากก็น้อยสำหรับการนำไปต่อยอดพัฒนางานวิจัยหรือนวัตกรรมที่ก่อให้เกิดประโยชน์ต่อชุมชน สังคมและประเทศต่อไป

สกุณา เจริญปัญญาศักดิ์ วรรณรัช สันติอมรทัต

ภาควิชาวิศวกรรมคอมพิวเตอร์ ศูนย์เครือข่ายความรู้เฉพาะด้านเครือข่ายเซนเซอร์ไร้สาย (COE-WSN)

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์

E-mail: coewsn@gmail.com

สารบัญ

บทที่ 1

แนะนำเครือข่ายเซนเซอร์ไร้สาย

(Introduction to Wireless Sensor Networks)

จุดประสงค์การเรียนรู้

เพื่อแนะนำเทคโนโลยีเครือข่ายเซนเซอร์ไร้สายและงานประยุกต์ที่เกี่ยวข้อง

เนื่องจากเทคโนโลยีทางด้านอิเล็กทรอนิกส์ และคอมพิวเตอร์ได้ก้าวหน้าไปอย่างมากในช่วง 10 ปีที่ผ่านมา ไม่ว่าจะเป็นเทคโนโลยีขนาดเล็กที่เรียกว่า นาโนเทคโนโลยี (Nano Technology) จึงทำให้ขนาดของทรานซิสเตอร์ และวงจรอิเล็กทรอนิกส์มีขนาดเล็ก ในระดับนาโนเมตร (10^{-9}) ตามด้วยวิวัฒนาการของเทคโนโลยีทางด้าน Micro Electro-Mechanical Systems (MEMS) ที่เราเรียกติดปากว่า “เมมส์” ในภาษาไทยจะเรียกว่าระบบไฟฟ้าเครื่องกลจุลภาค ซึ่งเป็นอุปกรณ์ขนาดเล็กในระดับไมโครเมตร จึงช่วยประหยัดพลังงาน และสามารถทำให้อุปกรณ์อิเล็กทรอนิกส์มีราคาไม่สูง ดังนั้นเมื่อนำเทคโนโลยีเหล่านี้ผสมผสานกับหน่วยประมวลผล (Processing unit) ตัวตรวจวัด (Sensor) และความสามารถในการติดต่อสื่อสารในระยะสั้น (Short range communication) ก่อให้เกิดเทคโนโลยีเครือข่ายเซนเซอร์ไร้สาย (Wireless Sensor Networks: WSNs)

เครือข่ายเซนเซอร์ไร้สายประกอบด้วยอุปกรณ์ขนาดเล็กที่เรียกว่าโนด (Node) เป็นจำนวนมาก ซึ่งจะถูกใช้งานกระจายลงไปอยู่ในพื้นที่ที่ต้องการใช้งาน ตัวโนดเองนั้นจะมีองค์ประกอบที่สำคัญคือ 1) หน่วยประมวลผล โดยทั่วไปจะใช้ไมโครคอนโทรลเลอร์ 2) หน่วยความจำสำหรับเก็บข้อมูลชั่วคราวก่อนที่จะทำการส่งต่อไปยังเครื่องแม่ข่าย 3) ตัวตรวจวัดหรือเรียกว่าเซนเซอร์ (Sensor) จะทำหน้าที่เก็บข้อมูลจากสิ่งแวดล้อมที่สนใจ ตัวอย่างเช่น

ตัวตรวจวัดอุณหภูมิ ความชื้นในอากาศ ตัวตรวจวัดปริมาณน้ำฝน และตัวตรวจวัดปริมาณออกซิเจนในน้ำ เป็นต้น 4) ภาครับส่งคลื่นวิทยุ ทำหน้าที่รับส่งข้อมูลแบบไร้สาย โดยจะต้องรองรับมาตรฐานการเชื่อมต่อแบบ IEEE 802.15.4 มีความสามารถในการสื่อสารแบบ Ad hoc บนย่านความถี่ 2.4 GHz (ตามช่วงความถี่ที่ได้รับอนุญาตในประเทศไทย) 5) แหล่งจ่ายพลังงาน ในที่นี้คือใช้แบตเตอร์รี่ ซึ่งอาจจะมีความสามารถในการหาพลังงานเพิ่มเติม (Energy harvesting) จากสิ่งแวดล้อม เช่น พลังงานแสงอาทิตย์ พลังงานลม และพลังงานน้ำ เป็นต้น

การออกแบบและพัฒนาเครือข่ายเซนเซอร์ไร้สายเริ่มต้นจากโครงการ Smart dust [1] ของมหาวิทยาลัย Berkeley สหรัฐอเมริกา ซึ่งเป็นจุดเริ่มต้นสำหรับงานวิจัยทางด้านนี้ โดยมีการพัฒนาโนดตามข้อจำกัดของการนำระบบเครือข่ายเซนเซอร์ไร้สายไปใช้งาน คือมีพลังงานให้ใช้งานอย่างจำกัด ฉะนั้นในส่วนของการออกแบบโนด รวมถึงการนำโพรโทคอลที่ใช้สำหรับสื่อสาร จึงจำเป็นที่จะต้องคำนึงถึงการใช้พลังงาน (Energy awareness) ระบบมีความจำเป็นที่จะต้องทราบระดับพลังงานคงเหลือของโนด และทำการหาพลังงานเพิ่มเติมจากสิ่งแวดล้อม เมื่อระดับพลังงานลดต่ำกว่าเกณฑ์ที่กำหนด เป็นต้น ดังนั้นหัวข้องานวิจัยจึงเกี่ยวข้องกับการใช้พลังงานอย่างมีประสิทธิภาพ (Energy efficiency) นักวิจัยจึงพยายามลดการใช้พลังงานไฟฟ้าทั้งสองส่วนคือ พลังงานไฟฟ้าในส่วนของไมโครคอนโทรลเลอร์ (E_{cpu}) แสดงไว้ในสมการที่ 1 [2] และพลังงานไฟฟ้าในส่วนของรับส่งข้อมูล (E_{Tx} แล E_{Rx}) ดังแสดงไว้ในสมการที่ 2 และ 3 [3]

$$E_{cpu} = E_{leakage} + E_{dynamic} = I_{leakage} * V_{dd} * t + C_{total} * V_{dd}^2 \quad \text{..... สมการ 1}$$

$$E_{Tx} = E_{elec} * k + \varepsilon_{amp} * k * d^2 \quad \text{..... สมการ 2}$$

$$E_{Rx} = E_{elec} * k \quad \text{..... สมการ 3}$$

ซึ่งในสมการที่ 1 C_{total} คือโหลดรวมทั้งหมดที่มี activity จากการคำนวณประมวลผล V_{dd} คือ แรงดันไฟฟ้า และ $I_{leakage}$ คือกระแสรั่วไหล พลังงานในส่วนของไมโครคอนโทรลเลอร์นี้สามารถประหยัดได้ด้วยการเลือกใช้ทรัพยากรฮาร์ดแวร์เท่าที่จำเป็นเท่านั้น หรือเลือกใช้ไมโครคอนโทรลเลอร์ที่มีโหมดประหยัดพลังงาน เป็นต้น พลังงานนี้เป็นเพียงส่วนหนึ่งของโนดเท่านั้น ยังมีพลังงานอีกส่วนที่สูญเสียไปในการรับส่งข้อมูลดังสมการที่ 2 และ 3 ซึ่ง E_{elec} = 50 nJ/bit ในการรับหรือส่งข้อมูล และ ε_{amp} = 100 pJ/bit/m^2 เป็นค่าของตัวขยายกำลังในการส่ง k คือจำนวนบิต และ d คือระยะทางระหว่างตัวรับและตัวส่ง สำหรับพลังงานส่วนที่สูญเสียในการรับส่ง

ข้อมูลสามารถประหยัดได้โดยการส่งเฉพาะข้อมูลที่จำเป็นเท่านั้น และส่งในระยะทางใกล้ๆเพื่อลดพลังงานในส่วนของตัวขยายกำลังในการส่ง ดังนั้นการพัฒนาโนดหรือโพรโทคอลที่ใช้ในระบบเครือข่ายเซนเซอร์ไร้สายจึงจะคำนึงถึงตัวแปรต่างๆเหล่านี้เพื่อให้เกิดการสูญเสียพลังงานไปอย่างมีประสิทธิภาพมากที่สุด

เครือข่ายเซนเซอร์ไร้สายถูกนำมาประยุกต์ใช้งานและเริ่มเข้ามามีบทบาทในชีวิตประจำวันของเรามากขึ้น โดยที่ตัวอย่างงานประยุกต์ของเครือข่ายเซนเซอร์ไร้สายจะได้อธิบายไว้ในบทที่ 2 โครงสร้างและสถาปัตยกรรมของโนดจะนำเสนออยู่ในบทที่ 3 ในขณะที่สถาปัตยกรรมของเครือข่ายจะเป็นเรื่องถัดไปที่จะอธิบายอยู่ในบทที่ 4 สำหรับการสื่อสารระหว่างโนดในเครือข่ายจำเป็นที่จะต้องมีโพรโทคอลของแต่ละชั้นเครือข่ายจะถูกอธิบายไว้ในบทที่ 5 และการสร้างระบบเครือข่ายเซนเซอร์ไร้สายด้วย XBee ซึ่งเป็นหนึ่งในหลายรูปแบบของโนดที่ใช้มาตรฐานโพรโทคอล Zigbee จะอยู่ในบทที่ 6 และ 7 ตามลำดับ

สำหรับในบทนี้นอกจากที่ได้กล่าวถึงเทคโนโลยีเครือข่ายเซนเซอร์ไร้สายเบื้องต้นไปก่อนนี้แล้ว จะได้สรุปรูปแบบของโนดหรือเรียกว่า Node platform ที่ได้รับการพัฒนาขึ้นให้เป็นอุปกรณ์ชิ้นสำคัญในเครือข่ายเซนเซอร์ไร้สาย ถัดมาจะได้อธิบายมาตรฐานเครือข่ายที่ใช้โนด (Node network standard) และในส่วนสุดท้ายสำหรับบทนี้จะแนะนำสถาปัตยกรรมของเครือข่ายเซนเซอร์ไร้สายและชั้นโพรโทคอลที่เกี่ยวข้อง

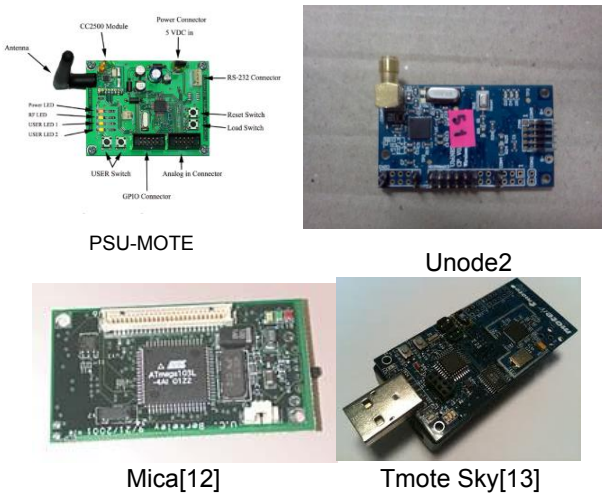
1.1 รูปแบบของโนด (Node platform)

โนดในเครือข่ายเซนเซอร์ไร้สายคือระบบสมองกลฝังตัว (Embedded system) ขนาดเล็กที่มีความสามารถเชื่อมต่อกับเซนเซอร์หลากหลายชนิด สามารถประมวลผลข้อมูลเบื้องต้นบนตัวเองและทำการติดต่อสื่อสารแบบไร้สายเพื่อแลกเปลี่ยนข้อมูลระหว่างกันเป็นเครือข่ายได้ ซึ่งตลอดระยะเวลา 10 กว่าปีที่ผ่านมาได้มีการพัฒนาโนดออกมาใช้งานกันหลากหลายซึ่งสามารถรวบรวมและทำการจัดกลุ่มของรูปแบบของโนดได้ 2 หมวด คือ รูปแบบขนาดเล็ก (Low-end platform) และ รูปแบบขนาดใหญ่ (High-end platform)

1.1.1 รูปแบบขนาดเล็ก (Low-end platform)

โนดรูปแบบนี้จะเน้นให้มีขนาดเล็ก ประหยัดพลังงาน และราคาถูก ซึ่งแน่นอนว่าจะทำให้สามารถเลือกอุปกรณ์หรือวัสดุไอซีใช้งานได้จำกัด เช่น ไมโครคอนโทรลเลอร์จะมีความถี่สัญญาณนาฬิกาที่ต่ำอยู่ในระดับ ไม่เกิน 30 MHz มีขนาดของหน่วยความจำบนโนดไม่มาก และกำลังส่งของคลื่นวิทยุจะต่ำทำให้ไม่สามารถส่งข้อมูลใน

ระยะทางได้ไกล แต่จะใช้คุณลักษณะของเครือข่ายเซนเซอร์ไร้สายที่ส่งแบบ Ad hoc แทน ซึ่งภาพรวมของโนดในกลุ่มรูปแบบขนาดเล็กนี้ได้รวบรวมและแสดงไว้ในภาพประกอบที่ 1.1 และมีรายละเอียดของโนดดังนี้



ภาพประกอบที่ 1.1 โนดหลากหลายชนิดในกลุ่มรูปแบบขนาดเล็ก

โนดตระกูล Mica เป็นโนดที่ถูกพัฒนาโดยบริษัท Crossbow[4] ตามลำดับในแต่ละรุ่นดังนี้ Mica Mica2dot Mica2 MicaZ และ IRIS ซึ่งโนดแต่ละรุ่นจะใช้ไมโครคอนโทรลเลอร์ Atmel AVR ขนาด 8 บิต ที่มีความถี่ของสัญญาณนาฬิกาตั้งแต่ 4-16 MHz และขนาดหน่วยความจำชนิด Flash ขนาด 128-256 kBแตกต่างกันไป โดยโนดรุ่นแรก Mica ใช้ภาครับส่งวิทยุที่ความถี่ 433 MHz มีอัตราการส่งข้อมูลที่ 40 kbps ส่วนใน Mica2 รุ่นต่อมาผู้ใช้สามารถเลือกย่านความถี่วิทยุได้ 3 ย่านคือ 433, 868 หรือ 916 MHz ในขณะที่รุ่น MicaZ และ IRIS จะใช้ภาครับส่งวิทยุไปตามมาตรฐานของ IEEE 802.15.4 คือ 2.4 GHz อัตราการส่งข้อมูลที่ 250 kbps

โนด Telos หรือ Tmote Sky เป็นโนดที่ได้รับการพัฒนาโดยบริษัท Moteiv ซึ่งร่วมพัฒนาระหว่างบริษัท Crossbow และ Tmote Sky ที่มีคุณลักษณะเหมือนโนด MicaZ และ IRIS แต่ต่างตรงที่โนด Telos จะใช้ไมโครคอนโทรลเลอร์ของ TI MSP430f1611 ทำงานด้วยความถี่ของสัญญาณนาฬิกาที่ 8 MHz มีขนาดของหน่วยความจำที่สูงกว่า MicaZ อยู่ที่ 10kB ภาครับส่งคลื่นวิทยุใช้ CC2420 ของบริษัท ChipConนอกจากนี้โนด Telos ยังได้เพิ่มส่วนของเซนเซอร์อุณหภูมิความชื้นในอากาศ และเซนเซอร์วัดความเข้มแสงลงบนโนดอีกด้วย มีการเชื่อมต่อโนดเข้ากับคอมพิวเตอร์ผ่านทางพอร์ด USB ด้วยไอซี FDDI

โนด EYES [5] เป็นการออกแบบและพัฒนาโนดที่ไม่แตกต่างจากโนด Telos ของกลุ่มวิจัยในภาคพื้นยุโรป แตกต่างที่ใช้ไมโครคอนโทรลเลอร์ขนาด 16 บิตมีหน่วยความจำสำหรับเก็บโปรแกรม 60 kB และเก็บข้อมูลที่ 2kB โนด EYES ใช้ภาครับส่งวิทยุผ่านทาง TR1001 ด้วยอัตราการส่งข้อมูลที่ 115.2 kbps และใช้พอร์ตอนุกรม RS232 ในการเชื่อมต่อกับคอมพิวเตอร์

โนด PSU-Mote เป็นโนดที่ได้รับการพัฒนาและออกแบบโดยศูนย์ความรู้เฉพาะด้านเครือข่ายเซนเซอร์ไร้สาย คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์ ซึ่งโนดประกอบด้วยไมโครคอนโทรลเลอร์ ARM LPC2138 เชื่อมต่อกับภาครับส่งคลื่นวิทยุของ ChipCon CC2500

โนด Unode เป็นโนดที่ได้รับการปรับปรุงจากโนด Telos โดยศูนย์ความรู้เฉพาะด้านเครือข่ายเซนเซอร์ไร้สาย คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์ ซึ่งเป็นการลดขนาดของโนดแยกส่วนของการเชื่อมต่อ USB ออกจากตัวโนด Telos เพื่อให้ Unode มีขนาดเล็กลง อีกทั้งทำการเพิ่มขนาดของหน่วยความจำชนิด Flash เพื่อเก็บข้อมูลขนาด 1Mbit

โนด Unode2 เป็นโนดที่ได้รับการพัฒนาจากศูนย์ความรู้เฉพาะด้านเครือข่ายเซนเซอร์ไร้สาย คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์ โดยการเพิ่มช่องการติดต่อสื่อสารผ่านทาง Bluetooth เพื่อให้โนดสามารถแลกเปลี่ยนข้อมูลกับ Mobile device ได้ อีกทั้งเพิ่มอุปกรณ์อ่าน RFID เพื่อให้สามารถนำโนดไปประยุกต์ใช้งานได้หลากหลายมากขึ้น

1.1.2 รูปแบบขนาดใหญ่ (High-end platform)

โนดในรูปแบบขนาดใหญ่จะถูกออกแบบและพัฒนาให้สามารถมีศักยภาพในการประมวลผลข้อมูลได้มากกว่าโนดขนาดเล็ก โดยการประมวลผลนั้นจำเป็นที่จะต้องใช้ข้อมูลจากหลากหลายเซนเซอร์ที่ต่อเชื่อมในเครือข่าย จากนั้นโนดขนาดใหญ่นี้จะถูกคาดหวังให้ทำหน้าที่ต่างๆ เช่น 1) ทำการส่งต่อข้อมูลไปหาโนดขนาดใหญ่ด้วยกันเพื่อให้ข้อมูลสามารถเดินทางไปถึงเครื่องแม่ข่ายได้โดยโนดขนาดใหญ่จะเป็นตัวแทนของกลุ่มโนดขนาดเล็กซึ่งเราเรียกว่า Cluster Head, CH 2) โนดขนาดใหญ่นี้เองยังจะต้องมีความสามารถในการเชื่อมต่อเครือข่ายเซนเซอร์ไร้สายกับเครือข่ายอื่นๆ ซึ่งเราเรียกโนดที่ทำหน้าที่นี้ว่า Gateway ตัวอย่างการทำงานของโนดขนาดใหญ่จะอธิบายในรายละเอียดดังต่อไปนี้

Stargate [7]: เป็นโนดขนาดใหญ่ที่เน้นให้มีการประมวลผลบนโนด ดังนั้น Stargate จึงใช้ไมโครโพรเซสเซอร์ Intel PXA-255 Xscale ทำงานที่ความถี่สัญญาณนาฬิกา 400 MHz ซึ่งเป็นไมโครโพรเซสเซอร์รุ่นเดียวกับที่ใช้ใน Pocket PC หน่วยความจำบน Stargate เป็นแบบชนิด Flash ขนาด 32 MB และหน่วยความจำ SDRAM ขนาด 64 MB สามารถเชื่อมต่อกับโนด Mica เพื่อใช้เป็นช่องทางการติดต่อกับเครือข่ายเซนเซอร์ไร้สาย ในขณะที่มี PCMCIA Bluetooth และ IEEE802.11 บน Stargate เพื่อให้เป็นช่องทางในการติดต่อกับเครือข่ายชนิดอื่น

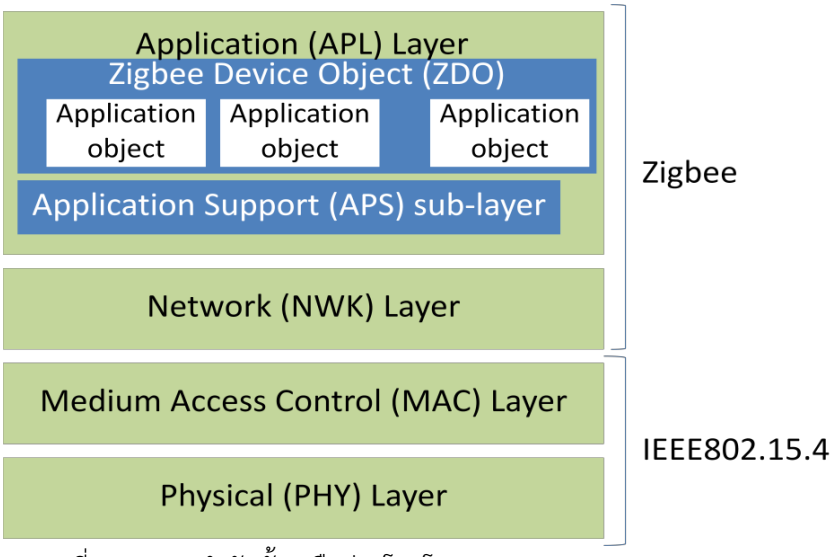
IMote [4] เป็นโนดที่ได้รับการพัฒนาจากบริษัท Crossbow แบ่งออกเป็น 2 รุ่นคือ IMote และ IMote2 โดยใช้ไมโครคอนโทรลเลอร์ ARM7 และ PXA271 XScale ตามลำดับ แต่ทั้ง 2 รุ่นใช้ภาครับส่งวิทยุเป็น ChipCon CC2420 บนมาตรฐาน IEEE802.15.4 และ Imote ใช้ระบบปฏิบัติการ TinyOS ในขณะที่ Imote2 ใช้ระบบปฏิบัติการ Linux

1.2 มาตรฐานเครือข่ายของโนด (Node Network Standard)

จะพบว่ามีความพยายามในการพัฒนาโนดขึ้นมาหลากหลายชนิดในแต่ละรูปแบบ จึงจำเป็นที่จะต้องมีมาตรฐานกลางเพื่อให้โนดแต่ละชนิดสามารถทำงานบนเครือข่ายเซนเซอร์ไร้สายร่วมกันได้ ในที่นี้จะกล่าวถึง 2 มาตรฐานอย่างคร่าวๆคือ IEEE 802.15.4 [7] และ Zigbee [8]

IEEE 802.15.4 เป็นมาตรฐานสำหรับการสื่อสารแบบไร้สายเน้นที่อัตราการส่งข้อมูลแบบต่ำๆ เพื่อให้สามารถใช้งานแบตเตอร์รี่ได้นานมากขึ้น ให้ระบบมีความซับซ้อนน้อยที่สุด โดยกำหนดให้มีย่านความถี่ใช้งานได้ที่ 2.4 GHz (ใช้ทั่วโลก) 915 MHz (ใช้ในทวีปอเมริกา) และ 868 MHz (ใช้ในทวีปยุโรป) กำหนดการทำมอดูเลชั่นสัญญาณในชั้นกายภาพ (Physical layer, PHY) ด้วยรูปแบบ Bit Phase Shift Keying (BPSK) สำหรับย่านความถี่ 868 และ 915 MHz และทำการมอดูเลชั่นด้วยรูปแบบ Offset Quadrature Phase Shift Keying (O-QPSK) สำหรับย่านความถี่ 2.4 GHz สำหรับรายละเอียดในชั้น Medium Access Control (MAC) กำหนดให้มีการเชื่อมต่อเครือข่ายใน 3 รูปแบบคือ Star Mesh และ Cluster tree-based ซึ่งโนดจะมีระยะการส่งที่ 10 ถึง 100 เมตร ในปัจจุบันส่งได้ไกลมากถึง 2-3 กิโลเมตรในที่โล่งรายละเอียดการทำงานและรูปแบบข้อมูลตาม (มาตรฐาน IEEE 802.15.4 จะได้อธิบายเพิ่มเติมไว้ในบทที่ 5

สำหรับมาตรฐาน Zigbee ถูกพัฒนาขึ้นโดย Zigbee Alliance ตามมาตรฐาน Zigbee ถูกออกแบบมาเพื่อให้ราคาของโนดลดลง มีมาตรฐานของเครือข่ายไร้สายที่จะใช้สำหรับส่งข้อมูลขนาดเล็ก ประหยัดพลังงาน ข้อมูลมีความปลอดภัยและน่าเชื่อถือ โดยเน้นการนำไปประยุกต์ใน 5 งานคือ บ้านอัตโนมัติ พลังงาน อาคารอัตโนมัติ บริการโทรคมนาคม และสุขภาพ มาตรฐาน Zigbee ได้กำหนดมาตรฐานต่อจากมาตรฐาน IEEE802.15.4 ดังแสดงในภาพประกอบที่ 1.2 รายละเอียดของชั้นกายภาพ (PHY) และ MAC ถูกกำหนดตามมาตรฐาน IEEE802.15.4 ส่วนชั้น Network (NWK) และ Application (APL) ถูกกำหนดในมาตรฐาน Zigbee ภายในชั้น APL จะประกอบด้วย APS (Application Support) และ Zigbee Device Object (ZDO) ซึ่งใน ZDO จะมี Application Object อยู่ที่สามารถกำหนดได้ด้วยผู้ใช้ เนื่องจากงานประยุกต์มีอยู่หลากหลายดังนั้นจึงต้องกำหนดชนิดของ Traffic ไว้ดังนี้ 1) *Periodic data traffic* ไว้สำหรับงานประยุกต์ประเภทเฝ้าดูข้อมูลต่อเนื่อง (Monitoring) ที่ได้รับจากเซนเซอร์ การแลกเปลี่ยนข้อมูลจะถูกควบคุมด้วย คอนโทรลเลอร์เครือข่าย หรือ Router 2) *Intermittent data traffic* ถูกใช้งานประยุกต์แบบ Event-based อาจจะถูกเรียกใช้งานโดยโปรแกรมประยุกต์หรือเหตุการณ์จากภายนอก Traffic ชนิดนี้จะถูกควบคุมด้วยแต่ละโนดเอง ในช่วงเวลาที่ไม่ใช้งานก็สามารถเข้าสู่โหมด Sleep เพื่อประหยัดพลังงาน 3) *Repetitive low-latency data traffic* ถูกกำหนดไว้สำหรับการสื่อสารเฉพาะอย่าง เช่น เมื่อได้รับสัญญาณจากเมาส์ที่คลิกจะต้องรีบตอบสนองภายในเวลาที่กำหนด โปรแกรมจะถูกเขียนในรูปแบบ Polling ตามมาตรฐาน IEEE802.15.4



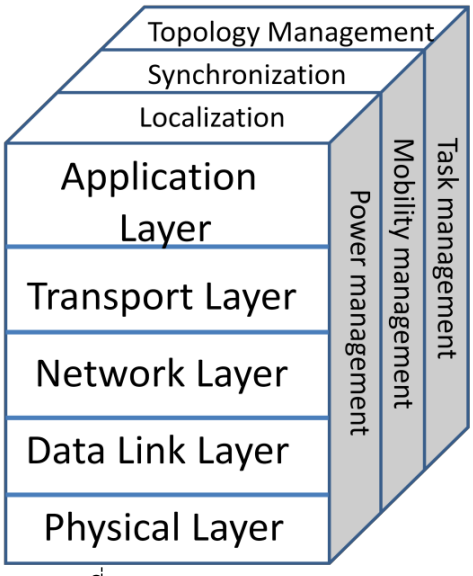
ภาพประกอบที่ 1.2 แสดงลำดับชั้นเครือข่ายโพรโทคอล IEEE 802.15.4 และ Zigbee

ในชั้น NWK ใช้บริการจัดการเครือข่าย มีกระบวนการสร้างเครือข่าย เนื่องจากอุปกรณ์ Zigbee สามารถถูกนำเข้าไปเป็นส่วนหนึ่งของเครือข่ายที่แตกต่างกัน ดังนั้นมาตรฐานจึงต้องกำหนดให้มีกระบวนการให้หมายเลข address ที่ยืดหยุ่นได้ เช่น เมื่อโนดถูกนำเข้าสู่เครือข่ายก็จะถูกกำหนดหมายเลขให้ ชั้น NWK ยังช่วยจัดการเรื่องการทำ synchronization ระหว่างโนดกับไมโครคอนโทรลเลอร์เครือข่าย และสุดท้ายชั้นนี้ยังจัดการเรื่องของการค้นหาเส้นทางอีกด้วย

นอกเหนือจากมาตรฐานของเครือข่ายแล้วยังมีมาตรฐานในส่วนของซอต์แวร์สำหรับโนดตัวอย่างเช่น ระบบปฏิบัติการ TinyOS [9] ซึ่งรองรับโนดตระกูล Mica, Telos และ IMote หรือ ระบบปฏิบัติการ Contiki [10] หรือระบบปฏิบัติการ LiteOS[11] เป็นต้น

1.3 สถาปัตยกรรมของเครือข่ายเซนเซอร์ไร้สาย

ในเครือข่ายเซนเซอร์ไร้สายโนดจะทำหน้าที่อยู่ 2 อย่างคือโนดรับข้อมูลจากเซนเซอร์ โนดจะต้องพยายามสื่อสารในเครือข่ายเพื่อนำส่งข้อมูลจากโนดต้นทาง (Source node) ไปยังโนดปลายทางหรือที่เรียกว่า Sink node และหน้าที่สุดท้ายคือการส่งต่อข้อมูลเรียกโนดนี้ว่า Router การส่งข้อมูลไม่จำเป็นที่จะต้องส่งครั้งเดียวแล้วถึงโนดปลายทาง การส่งอาจจะส่งผ่านหลายโนดไปทีละทอด หรือเรียกว่า (Multi-hop) ในหัวข้อนี้จะได้อธิบายถึงรูปแบบของข้อมูลที่จะถูกส่งผ่านเครือข่ายเซนเซอร์ไร้สายด้วยโพรโทคอลที่เหมาะสม มาตรฐานของโพรโทคอลเครือข่ายถูกแบ่งออกได้เป็น 7 ชั้นดังภาพประกอบที่ 1.3



ภาพประกอบที่ 1.3 Sensor network protocol stack

1.3.1 Physical Layer

ชั้น PHY นี้ถูกใช้ในการเลือกความถี่ การสร้างสัญญาณความถี่ที่ต้องการ การตรวจสอบสัญญาณ การมอดูเลชั่น และการเข้ารหัสข้อมูล สำหรับการสร้างและตรวจสอบสัญญาณความถี่จะถูกออกแบบไว้เป็นฮาร์ดแวร์หรือไอซีที่จะเลือกใช้งานได้ทันที สำหรับในเครือข่ายเซนเซอร์ไร้สายจะเน้นในส่วนของผลกระทบในการกระจายตัวของสัญญาณ การใช้พลังงานอย่างมีประสิทธิภาพ และวิธีการมอดูเลชั่น

1.3.2 Data link Layer

ในชั้นของ Data link รับผิดชอบในการเลือกข้อมูล ตรวจสอบเฟรมข้อมูล และ Medium access and error control เพื่อเพิ่มความน่าเชื่อถือในการเชื่อมต่อจุดต่อจุด (Point to point) และการเชื่อมต่อกับหลายจุด (Point to multipoint) สำหรับเครือข่ายเซนเซอร์ไร้สายได้กล่าวถึงโพรโทคอล MAC ในเครือข่ายไร้สายแบบ Multi-hop นี้ว่าจะต้องสร้างการเชื่อมต่อเพื่อให้เกิดการส่งข้อมูลได้ และจัดการการแบ่งปันช่องสื่อสารให้โนดแต่ละตัวในเครือข่ายอย่างเท่าเทียมกันหรืออย่างมีประสิทธิภาพ อีกหน้าที่ที่สำคัญของชั้นนี้คือการควบคุมข้อผิดพลาดของการส่งข้อมูลซึ่งมีให้เลือกใช้งาน 2 แบบคือ Forward error correction (FEC) และ Automatic repeat request (ARQ) กระบวนการ ARQ มีประโยชน์แต่ก่อให้เกิดภาระ Overhead ของการส่งซ้ำ และมีความซับซ้อนมากกว่า FEC

1.3.3 Network Layer

ในชั้น NWK จะทำหน้าที่ค้นหาเส้นทาง ดังนั้นจะพบโพรโทคอลค้นหาเส้นทาง (Routing protocol) อยู่ในชั้นนี้ ซึ่งมีหลากหลายโพรโทคอลค้นหาเส้นทางที่ถูกพัฒนาขึ้นมาให้เหมาะสมสำหรับใช้ในเครือข่ายเซนเซอร์ไร้สายเช่น จะต้องใช้พลังงานอย่างมีประสิทธิภาพ จะต้องเหมาะสมสำหรับเครือข่ายในรูปแบบ Data centric และเนื่องจากโนดมีจำนวนมากแต่ละโนดจึงไม่สามารถใช้หมายเลขแบบ Unique ได้ รายละเอียดของโพรโทคอลค้นหาเส้นทางที่นิยมใช้ในเครือข่ายเซนเซอร์ไร้สายจะถูกอธิบายในบทที่ 5

1.3.4 Transport Layer

ในชั้น Transport สำหรับเครือข่ายเซนเซอร์ไร้สายจะแตกต่างจากเครือข่ายปกติตรงนี้เน้นเพียงแค่ 2 หน้าที่คือ ความน่าเชื่อถือของเครือข่ายและควบคุมความคับคั่งของข้อมูล แต่ด้วยข้อจำกัดของโนดในเรื่องพลังงาน

ความเร็วในการประมวลผลและขนาดของหน่วยความจำ ทำให้การทำงานของชั้นนี้จะต้องไม่ซับซ้อน กระบวนการที่ซับซ้อนในชั้นนี้อาจจะถูกพัฒนาลงบนโนดขนาดใหญ่แทน

1.3.5 Application Layer

เป็นการรวมฟังก์ชันที่จำเป็นสำหรับใช้งานเครือข่ายเซนเซอร์ไร้สาย เช่น จำเป็นที่จะต้องมี Application ในส่วนของ *Timing* เพื่อให้สามารถส่งข้อมูลให้กับโพรโทคอล Time synchronization เป็นต้น ในบางงานชั้น Application จำเป็นที่จะต้องใช้ข้อมูลในระดับกายภาพของชั้น PHY มาใช้เช่น การทำ Localization ด้วยค่า RSSI จะพบว่าเราจะต้องนำเทคนิคของการทำ Cross-layer เพื่อส่งผ่านข้อมูลจากชั้น PHY ขึ้นมาให้โดยตรง

จะพบว่าเครือข่ายเซนเซอร์ไร้สายสามารถถูกนำไปประยุกต์ใช้งานได้หลากหลาย และได้รับความนิยมเป็นอย่างมากในปัจจุบัน ตัวอย่างของ งานประยุกต์ที่โดดเด่นของเครือข่ายเซนเซอร์ไร้สายจะถูกอธิบายไว้ในบทต่อไป

เอกสารอ้างอิง

[1] V. Hsu, J. M. Kahn, and K. S. J. Pister, “Wireless Communications for Smart Dust,” *Electronics Research Laboratory Technical Memorandum Number M98/2*, February, 1998.

[2] Neil H.E. Weste and K. Eshraghian, “Principles of CMOS VLSI Design a Systems Perspective,” *Addison Wesley Longman,* 2nd edition*,* 1993.

[3] W.R. Heinzelman, A. Sinha, A. Wang and A.P. Chandrakasan, “Energy-Scalable Algorithms and Protocols for Wireless Microsensor Networks,” *Proceedings on IEEE International Conference of the Acoustics, Speech, and Signal Processing (ICASSP)*, pp.3722-3725, 2000.

[4] Crossbow technology. URL:http://www.xbox.com

[5] EYES Project. URL:http://www.eyes.eu.org

[6] Stargate platform. URL:http://www.xbox.com

[7] IEEE standard for information technology – telecommunications and information exchange between systems – local and metropolitan area networks – specific requirement part 15.4: wireless medium access

control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area networks (WPANS). *IEEE Std. 802.15.4a-2007 (Amendment to IEEE Std. 802.15.4-2006)*, pp. 1–203, 2007.

[8] ZigBee Alliance. http://www.zigbee.org

[9] TinyOS. http://www.tinyos.net

[10] Contiki. http://www.contiki-os.org

[11] LiteOS. http://www.linuxliteos.com

[12] David Culler and et.al., "MICA: The commercialization of Microsensor Motes," *Sensor Magazine*, April, 2002.

[13] Sentilla. http://www.sentilla.com/

บทที่ 2

งานประยุกต์ของเครือข่ายเซนเซอร์ไร้สาย

(Wireless Sensor Networks Applications)

จุดประสงค์การเรียนรู้

เพื่อให้ทราบงานประยุกต์ที่นำเทคโนโลยีเครือข่ายเซนเซอร์ไร้สายไปใช้งาน

เทคโนโลยีเครือข่ายเซนเซอร์ไร้สายถูกนำไปประยุกต์ใช้งานในหลากหลายด้าน ซึ่งในบทนี้จะได้ยกตัวอย่างงานวิจัย และการนำเทคโนโลยีเครือข่ายเซนเซอร์ไร้สายไปประยุกต์ใช้งานทางด้านที่เกี่ยวข้องกับสิ่งแวดล้อม (Environment) ด้านการเกษตร (Agriculture) ด้านการทหาร (Military) ด้านอาคาร (Building) และด้านสุขภาพ (Health care)

2.1 การเฝ้าสังเกตการณ์สิ่งแวดล้อม (Environmental Observation)

ระบบเครือข่ายเซนเซอร์ไร้สายสามารถถูกนำไปใช้ในการเฝ้าระวังการเปลี่ยนแปลงของสิ่งต่างๆในธรรมชาติ ตัวอย่างที่ชัดเจนได้แก่ การนำเซนเซอร์โหนดไปติดตั้งตามลุ่มแม่น้ำ แหล่งน้ำ ที่อยู่ในบริเวณโรงงานอุตสาหกรรม เพื่อทำการตรวจสอบปริมาณสารเคมีที่ปนเปื้อนลงสู่แหล่งน้ำธรรมชาติ ซึ่งจะช่วยให้สามารถตรวจสอบปริมาณสารปนเปื้อนและสามารถป้องกันอันตรายจากสารเคมีที่จะเกิดแก่คน สัตว์และสิ่งแวดล้อมที่อยู่รอบๆบริเวณโรงงานอุตสาหกรรม นอกจากนี้การติดตั้งเซนเซอร์โหนดที่มีขนาดเล็กทำให้ไม่มีข้อจำกัดของการเข้าตรวจสอบในบริเวณที่เข้าถึงได้ลำบาก อีกตัวอย่างหนึ่งคือการเฝ้าสังเกตการณ์สัตว์ หรือสิ่งมีชีวิตที่นักวิทยาศาสตร์สนใจและต้องการเก็บข้อมูลถึงพฤติกรรม รวมทั้งการเฝ้าสังเกตการณ์สัตว์สงวนและสัตว์ป่าหายากได้อีกด้วย ในปี พ.ศ. 2545 ห้องปฏิบัติการ Intel ของมหาวิทยาลัย Berkeley ร่วมกับ College of the Atlantic in Bar Harbor

ร่วมกันทำการนำเอาระบบเซนเซอร์ไร้สายจำนวน 32 ตัวกระจายติดตั้งบนเกาะ Great Duck เมือง Maine ประเทศสหรัฐอเมริกา ดังภาพประกอบที่ 2.1 เพื่อพัฒนาระบบการเฝ้าสังเกตการณ์และเฝ้าระวังสิ่งมีชีวิต ทำให้นักวิทยาศาสตร์ได้ข้อมูลจำนวนมหาศาลที่จะเป็นประโยชน์ในงานวิจัย โดยในปี พ.ศ. 2546 ได้ติดตั้งเซนเซอร์โหนดเพิ่มขึ้นเป็น ๆ ตัวสำหรับรับข้อมูลทั่วไปที่อยู่ตามโพรงไม้ หลุมต่าง 60 กระจายทั่วเกาะ และมีโหนดอีก 25 ตัวทำหน้าที่ในการเก็บข้อมูลทางด้านอากาศกระจายอยู่ทั่วทั้งเกาะโดยทิ้งระยะห่างกันได้ถึง ฟุตในป่าลึก ซึ่ง 1000 ข้อมูลที่ได้จากเซนเซอร์จะถูกรายงานผ่านทางเว็ปไซต์http://www.greatduckisland.net รายละเอียดของงานวิจัยสามารถศึกษาเพิ่มเติมได้จาก [1-4]

ภาพประกอบที่ 2.1 เกาะ Great Duck ที่ทำการติดตั้งระบบเครือข่ายเซนเซอร์ไร้สาย [4]

ทั้งนี้การนำระบบเซนเซอร์ไร้สายประยุกต์ใช้งานบนเกาะแห่งนี้จำเป็นต้องพิจารณาถึงปัจจัยต่างๆอาทิ เช่น

- ทำการติดตั้งระบบอินเตอร์เน็ต
- โครงสร้างสถาปัตยกรรมของระบบเครือข่ายที่จะต้องใช้งานร่วมกัน เซนเซอร์โหนดอาจจะถูกจัดเป็นกลุ่มย่อยๆโดยในแต่ละกลุ่มจะเชื่อมต่อระหว่างกันผ่านทาง Gateway ของแต่ละกลุ่ม โดยที่ระยะห่างระหว่างกลุ่มอาจจะทิ้งห่างเป็นกิโลเมตร รวมทั้งระบบฐานข้อมูลที่จะใช้ในการเก็บข้อมูลจำนวนมหาศาลที่ได้จากแต่ละเซนเซอร์โหนด
- ระยะเวลาในการใช้งานเซนเซอร์โหนด ปัจจัยสำคัญที่เป็นตัวกำหนดอายุการใช้งานของโหนดคือระบบของพลังงานหรือแบตเตอร์รี่ที่จะสามารถจ่ายให้แก่โหนด ดังนั้นจึงจำเป็นที่จะต้องมีหน่วยของการจัดการใช้พลังงาน ซึ่งพลังงานส่วนใหญ่จะสูญเสียไปกับการส่งข้อมูล (มูลส่งข้อ-ภาควิทยุรับ) เนื่องจากหน่วยประมวลผลบนเซนเซอร์โหนดนั้นจะมีโหมดการทำงานที่สามารถ idle ตัวเองเพื่อช่วยประหยัดพลังงานได้ในระดับหนึ่ง ฉะนั้นจึงอาจจะต้องมีการกรองข้อมูลขั้นต้นก่อนที่จะทำการส่งเพื่อลดระยะเวลาหรือจำนวนครั้งในการจัดส่งข้อมูล
- ชนิดของเซนเซอร์ที่ต้องเลือกใช้ได้เหมาะสมกับข้อมูลที่ต้องการ

2.2 การประยุกต์ใช้ทางการเกษตร (Agriculture Systems)

การนำเอาระบบเซนเซอร์ไร้สายมาประยุกต์ใช้ในด้านการเกษตรสามารถทำได้ทั้งส่วนของการเฝ้าระวัง (Monitoring) เฝ้าสังเกตการณ์ (Observation) หรือผสมผสานร่วมกับระบบการควบคุม (Control systems) เพื่อให้เป็นระบบอัตโนมัติ ซึ่งความสามารถที่จะต้องมีเพิ่มขึ้นในตัวเซนเซอร์โหนดคือการควบคุม สั่งงานอุปกรณ์อิเล็กทรอนิกส์ โหนดที่ใช้ในการควบคุมเหล่านี้อาจจะไม่จำเป็นต้องคำนึงถึงระบบพลังงานมากนัก ถ้าอุปกรณ์อิเล็กทรอนิกส์นั้นมีแหล่งจ่ายพลังงานติดอยู่ และตัวโหนดที่ควบคุมสามารถติดอยู่บนอุปกรณ์นั้นๆได้ ทั้งนี้ทั้งนั้นขึ้นอยู่กับระบบที่ได้ทำการออกแบบ ตัวอย่างการใช้งานระบบเครือข่ายเซนเซอร์ไรสายในภาคเกษตรได้แก่ งานของ Prof. Aline Baggio มหาวิทยาลัย Delft ประเทศเนเธอร์แลนด์ ซึ่งได้มีทำโครงการวิจัยที่ชื่อ Lofar Agro [5] เพื่อเฝ้าระวังสภาพอากาศในพื้นที่การเกษตร โดยที่ข้อมูลที่ได้จะเก็บเป็นสถิติสำหรับการใช้งานระบบเครือข่ายเซนเซอร์ไร้สายในงานทางด้านนี้ เซนเซอร์โหนดที่ใช้ในงานมีลักษณะดังภาพประกอบที่ 2.2 โดยโหนดจะทำการเก็บข้อมูลเกี่ยวกับอุณหภูมิ และความชื้น ทุกๆนาที แต่เพื่อประหยัดพลังงานโหนดจะมีการรายงานผลกลับมาทุกๆ นาที ดังนั้นเซนเซอร์โหนดแบบนี้จึงจำเป็นจะต้องมีหน่วยความจำเพื่อเก็บข้อมูลพักไว้บนตัวโหนดซึ่งข้อมูล 10 เหล่านี้จะมีประโยชน์สำหรับศึกษาสภาพแวดล้อมที่เหมาะสมกับผลิตผลทางการเกษตร เพื่อนำไปศึกษาและขยายผลไปสู่ให้มีการควบคุมและจัดสภาพแวดล้อมให้เหมาะสมกับพืชการเกษตรนั้นๆ ทำให้ได้ผลผลิตที่ดีขึ้น

ภาพประกอบที่ 2.2 Lofar Node [5]

ในบทความ [6] ที่ได้ทดสอบระบบเครือข่ายเซนเซอร์ไร้สายในไร่องุ่นดังภาพประกอบที่ 2.3 เพื่อศึกษาถึงความเป็นไปได้ในการนำเอาระบบนี้ไปใช้งานกับเกษตรชาวไร่องุ่น โดยเน้นศึกษาถึงปัจจัยต่างๆที่จะทำให้ระบบเซนเซอร์ไร้สายนำไปใช้งานได้จริงกับชาวไร่ ซึ่งแตกต่างจากการงานอื่นๆที่ผู้ใช้งานเป็นนักวิจัยที่อยู่กันคนละสาขา กลุ่มนักวิจัยพบว่าการทำความเข้าใจถึงความจำเป็นที่ผู้ใช้ต้องการใช้งานระบบเซนเซอร์ไร้สายและเข้าใจถึงกิจกรรมต่างที่เกิดขึ้นจริงในไร่เป็นสิ่งสำคัญอย่างยิ่งที่จะช่วยให้นักวิจัยสามารถออกแบบระบบทั้งส่วนของฮาร์ดแวร์และซอฟต์แวร์ให้เข้ากับการใช้งานจริง ตัวอย่างเช่นตัวแปรต่างๆที่ต้องการในส่วนองุ่นจะเกิดขึ้นมากมายในช่วงเวลากลางวัน แต่ในทางตรงกันข้ามช่วงเวลากลางคืนจะพบตัวแปรที่ต้องเฝ้าติดตามน้อย ดังนั้นนักวิจัยสามารถ

ลดความถี่ในการอ่านข้อมูลจากเซนเซอร์ในช่วงกลางคืนและทำให้ช่วยประหยัดพลังงานลงได้ นอกจากนี้ฤดูกาลก็สามารถที่จะกำหนดความถี่ของการอ่านข้อมูลจากเซนเซอร์ได้เช่นกัน เนื่องจากในช่วงฤดูหนาวชาวไร้จะต้องเฝ้าระวังองุ่นเป็นพิเศษเนื่องจากความเย็นจะสามารถทำให้ผลผลิตเสียหายได้ และระบบเตือนภัย (Alarm systems) จะมีประโยชน์เฉพาะในช่วงฤดูหนาว อีกตัวอย่างหนึ่งของสิ่งที่ได้จากการทดลองนี้คือระบบเก็บผลผลิตอัตโนมัติซึ่งจะมีความจำเป็นเฉพาะในประเทศที่ต้องเสียค่าแรงสูงเท่านั้น

ภาพประกอบที่ 2.3 ระบบเครือข่ายเซนเซอร์ไรสายในไร่องุ่น [6]

2.3 การเฝ้าระวังทางการทหาร (Military Monitoring)

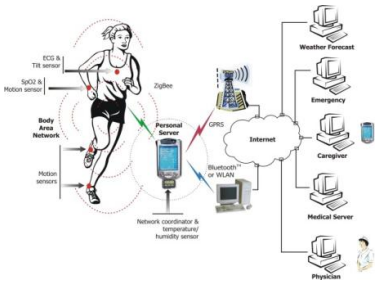
ในทางการทหารอาจจะนำระบบเครือข่ายเซนเซอร์ไร้สายประยุกต์ใช้ในสนามรบเพื่อตรวจสอบหาผู้รอดชีวิต หรือเฝ้าติดตามการเคลื่อนไหวของฝ่ายตรงข้าม หรือแม้แต่ในการทำระบบเฝ้าระวังอาวุธ ระบบเครือข่ายเซนเซอร์ไร้สายสามารถนำไปใช้เพื่อให้ทหารได้รับทราบข้อมูลแทนการติดต่อสื่อสารผ่านทางวิทยุที่อาจจะถูกจำกัดบริเวณ หรือมีโอกาสสูงในการถูกลักลอบดักฟัง หรือแม้แต่นำเซนเซอร์ให้ทหารพกติดตัวทำให้ทราบตำแหน่งของทหารที่ถูกส่งลงพื้นที่ ทราบจำนวนอาวุธที่คงเหลือ เป็นต้น ข้อมูลเหล่านี้จะเป็นประโยชน์สำหรับผู้วางแผนการรบ รวมถึงการติดตามทหารที่อาจจะหลงออกนอกพื้นที่ ซึ่งเริ่มมีงานวิจัยที่จะนำเอาระบบเซนเซอร์ไร้สายประยุกต์ใช้ในทางการทหาร เช่น [7] ศึกษาถึงการออกแบบระบบชั้นโพรโทคอลติดต่อสื่อสารผ่านระบบเครือขายไร้สายให้ง่ายและมีความเสถียรมากที่สุด

2.4 การเฝ้าระวังอาคารขนาดใหญ่ (Large Building Monitoring)

เนื่องจากการเฝ้าระวังอาคารขนาดใหญ่ได้รับความสนใจและกำลังเป็นที่สนใจในปัจจุบัน ระบบเครือข่ายเซนเซอร์ไร้สายสามารถนำมาประยุกต์ใช้เพื่อตรวจสอบโครงสร้างของอาคาร เฝ้าระวังตึกเกิดการสั่น หรือผลที่เกิดจากแรงสั่นสะเทือน ควบคุมและเฝ้าระวังสภาพแวดล้อมภายในตัวอาคารอาทิเช่น ระบบให้แสงสว่าง ระบบให้ความร้อน ระบบทำความเย็น หรือตรวจสอบหาเส้นทางที่ปลอดภัยและสั้นที่สุดเพื่ออพยพคนออกจากตัวอาคารกรณีที่มีเหตุฉุกเฉิน ในรายงานเชิงเทคนิค [8] ของภาควิชาวิทยาศาสตร์และวิศวกรรมศาสตร์คอมพิวเตอร์ มหาวิทยาลัย Buffalo ได้นำระบบเซนเซอร์ไร้สายทดลองประยุกต์ใช้ในตัวอาคารขนาดใหญ่ ทดลองใช้เซนเซอร์ตรวจวัดการสั่นของตัวอาคารที่ออกแบบโดย Nagayama et. al. ที่สามารถวัดได้ในช่วง 1 – 2,000 microstrain และเสียงที่ระดับ 1 microstrain [9] และระบบเซนเซอร์อื่นๆที่จำเป็นทางด้านวิศวกรรมโยธา [10-11] ซึ่งภายในตัวอาคารจะติดเซนเซอร์ที่ตัววัด NOx, COx, Nerve Gases, Anthrax ในการทดลองนี้ใช้เซนเซอร์โหนดที่ชื่อว่า Tmote [12]

2.5 สุขภาพการแพทย์ (Healthcare)

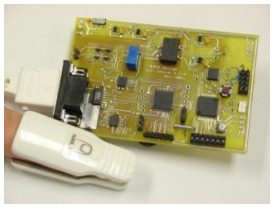
ในทางการแพทย์ระบบเครือข่ายเซนเซอร์ไร้สายสามารถถูกนำไปใช้เพื่อปรับปรุงคุณภาพของการให้บริการทางด้านการแพทย์ เซนเซอร์จะถูกติดไว้กับคนเพื่อคอยเฝ้าระวังปัญหาทางด้านสุขภาพ อาทิเช่นผู้ป่วยโรคหัวใจก็จะมีเซนเซอร์คอยวัดระบบการเต้นของหัวใจเพื่อคอยระวังความผิดปกติที่อาจจะเกิดขึ้น การนำระบบเครือข่ายเซนเซอร์ไร้สายไปประยุกต์ใช้งานทางด้านนี้กำลังเป็นที่สนใจ ดังจะดูได้จากงานวิจัยของมหาวิทยาลัยชั้นนำทั่วไป เช่น ระบบ Wireless Body Area Network ของเซนเซอร์ที่ชาญฉลาดเพื่อตรวจเฝ้าติดตามสุขภาพแบบเคลื่อนที่ ของภาควิชาวิศวกรรมอิเล็กทรอนิกส์คอมพิวเตอร์ มหาวิทยาลัย Alabama in Huntsville [13-15] ดังภาพประกอบที่ 2.4 และ 2.5



(ที่มาของภาพ : http:www.ece.uah.edu/~jovanov/whrms/)

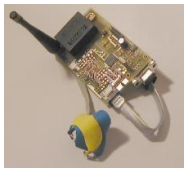
ภาพประกอบที่ 2.4 Wireless Body Area Network of Intelligent Sensors for Ambulatory Health Monitoring

(ก) เซนเซอร์วัดอัตราการเต้นของหัวใจ



(ข) เซนเซอร์วัดชีพจร (Pulse Oximeter)

(ค) เซนเซอร์ไร้สายที่มี ECG amplifier



(ง) เซนเซอร์ไร้สายวัดการหายใจ

(จ) เซนเซอร์ไร้สายตรวจการนั่งการเดิน

(ที่มาของภาพ : http:www.ece.uah.edu/~jovanov/whrms/)

ภาพประกอบที่ 2.5 เซนเซอร์ทางการแพทย์รูปแบบต่างๆ

งานวิจัยที่นำเอาระบบเครือข่ายเซนเซอร์ไร้สายไปใช้งานในงานทางด้านการแพทย์อีกชิ้นที่น่าสนใจคือ Codeblue [16-17] ของมหาวิทยาลัยฮาวาร์ดที่ร่วมมือกับมหาวิทยาลัย องค์กรและศูนย์การแพทย์ต่างๆในประเทศอเมริกา ตัวอย่างการใช้งาน Codeblue ในหน่วย Emergency แสดงไว้ในภาพประกอบที่ 2.6 ซึ่งซอฟต์แวร์ในระบบ CodeBlue นี้ได้จัดทำเป็นแบบ Open-source ทำให้นักวิจัยที่อื่นสามารถนำไปประยุกต์ใช้กับเซนเซอร์โหนดแบบไร้สาย สามารถศึกษาข้อมูลเพิ่มเติมได้ที่ http://www.eecs.harvard.edu/~mdw/proj/codeblue/

(ก) CodeBlue ที่ถูกนำไปใช้งานในแผนกฉุกเฉิน

(ข) Pluto Mote ขนาดเล็กที่สามารถพกพาได้

(ที่มาของรูปภาพ : http://www.eecs.harvard.edu/~mdw/proj/codeblue/)

ภาพประกอบที่ 2.6 ระบบ CodeBlue

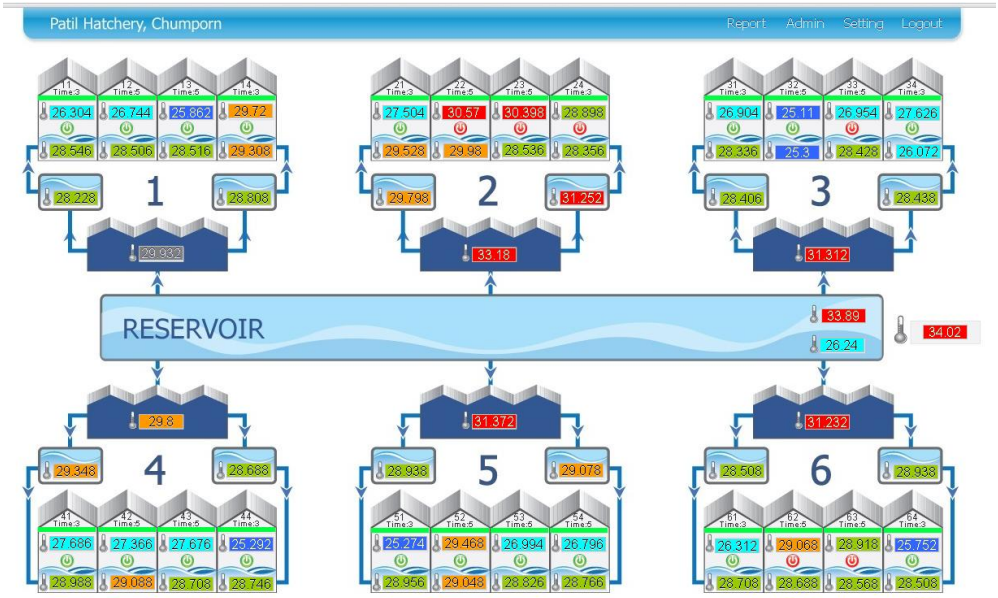
บทความนี้ได้แนะนำระบบเครือข่ายเซนเซอร์ไร้สาย และตัวอย่างการนำไปประยุกต์ใช้งาน จะพบว่าเราสามารถนำเอาระบบเครือข่ายเซนเซอร์ไร้สายนี้ไปใช้งานได้ในหลากหลาย ทั้งนี้การนำเอาเซนเซอร์โหนดไร้สายมาสร้างเป็นระบบเครือข่ายจะต้องเข้าใจว่าระบบเดิมที่มีอยู่มีการทำงานอย่างไร หรือเซนเซอร์โหนดจะเข้าไปช่วยแก้ปัญหาได้อย่างไร ในบทความตอนต่อไปเราจะเข้าไปดูถึงการทำงานของระบบเครือข่ายเซนเซอร์ไร้สายว่ามีองค์ประกอบอะไรบ้าง เซนเซอร์โหนดทำงานอย่างไร และในตอนท้ายจะเรียนรู้ว่าเซนเซอร์โหนดจะทำการติดต่อสื่อสารกันแบบใด

2.6 งานประยุกต์เครือข่ายเซนเซอร์ไร้สายของไทย

ในหัวข้อนี้จะขอยกตัวอย่างงานประยุกต์เครือข่ายเซนเซอร์ไร้สายที่ถูกนำไปใช้ในการเพาะเลี้ยงลูกกุ้งที่พัฒนาขึ้นโดยทีมคณาจารย์และนักวิจัย ของศูนย์เครือข่ายความรู้เฉพาะด้านเครือข่ายเซนเซอร์ไร้สาย คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์ ซึ่งได้รับทุนสนับสนุนการจัดตั้งศูนย์ฯจาก สวทช. เพื่อให้วิจัยและพัฒนาเทคโนโลยีทางด้านเครือข่ายเซนเซอร์ไร้สายในประเทศ

ในปีค.ศ. 2010 ทางฝ่าย IT ของบริษัทเครือเจริญโภคภัณฑ์อาหาร มหาชน (CPF) ได้สนับสนุนเงินทุนในการพัฒนาระบบเครือข่ายเซนเซอร์ไร้สายสำหรับใช้ในโรงเรือนเพาะเลี้ยงลูกกุ้ง ที่ อำเภอปะทิว จังหวัดชุมพร โดยมีจุดประสงค์เพื่อตรวจสอบและรักษาระดับของอุณหภูมิในบ่อพ่อพันธุ์แม่พันธุ์กุ้งให้เหมาะสมกับการผสมพันธุ์มากที่สุด ข้อมูลจากเซนเซอร์เกือบ 70 จุดทั่วทั้งฟาร์มจะถูกส่งผ่านระบบเครือข่ายเซนเซอร์ไร้สายบันทึกลงในฐานข้อมูล รวมทั้งสามารถควบคุมการเปิด-ปิดเครื่องปรับอากาศจำนวน 24 เครื่องผ่านเครือข่ายเซนเซอร์ไร้สายเช่นกันโดยการพัฒนาในระยะที่ 1 นั้นได้ออกแบบโหนดที่ตรงตาม Platform TinyOS ขึ้นใช้งานเองดังภาพประกอบที่ 2.7 ใช้ไมโครคอนโทรลเลอร์ MSP430F1611 และมีหน่วยความจำชนิด Flash 1 Mbit เพื่อใช้งานเป็นบัฟเฟอร์ชั่วคราวบนโหนด ในแต่ละกลุ่มของโหนดในโรงเรือนจะมีหัวหน้าโหนด (Cluster Head) เป็นตัวแทนรวบรวมข้อมูลเพื่อส่งข้อมูลกลับไปยังเครื่องแม่ข่าย

ภาพประกอบที่ 2.7 โหนดที่ใช้ในการรับค่าจากเซนเซอร์และใช้ควบคุม



ภาพประกอบที่ 2.8 โปรแกรมแสดงค่าอุณหภูมิ แถบสีอุณหภูมิและสถานะของเครื่องปรับอากาศ

จากภาพประกอบที่ 2.8 เป็นโปรแกรมแสดงผลค่าอุณหภูมิของน้ำในบ่อเลี้ยงกุ้งของแต่ละโรงเรือน โดยมีการจัดแบ่งเฉดสีของค่าอุณหภูมิในแต่ละระดับ ซึ่งผู้ดูแลสามารถกำหนดได้เอง มีการแสดงสถานะของเครื่องปรับอากาศด้วยว่าเปิดหรือปิดอยู่ และสามารถควบคุมการเปิด-ปิดได้จากโปรแกรม หรือจะสามารถตั้งค่าให้เครื่องปรับอากาศสามารถเปิดหรือปิดแบบอัตโนมัติ

สรุปท้ายบท

เทคโนโลยีเครือข่ายเซนเซอร์ไร้สายเป็นการรวมหลากหลายศาสตร์ที่เกี่ยวข้องได้แก่ ระบบสมองกลฝังตัวเพื่อพัฒนาอุปกรณ์สื่อสารขนาดเล็ก เซนเซอร์เพื่อใช้สำหรับตรวจวัด ระบบเครือข่ายแบบไร้สายเพื่อใช้ในการรับส่งข้อมูล และยังต้องเชื่อมต่อไปยังเครือข่ายทั่วไปเช่น อินเตอร์เน็ต ข้อมูลที่สามารถรวบรวมได้จากเซนเซอร์ก็จะสามารถเก็บบันทึกลงในฐานข้อมูล หรือสามารถใช้ประมวลผลในทันทีเพื่อนำไปใช้ในงานควบคุม จากตัวอย่างการนำเครือข่ายเซนเซอร์ไร้สายไปประยุกต์ใช้งานที่ได้กล่าวมาแล้วในบทนี้จึงถือได้ว่าเป็นเทคโนโลยีที่จะต้องถูกใช้งานอย่างแพร่หลายในอนาคตอันใกล้ ซึ่งก็สอดคล้องกับทิศทางเทคโนโลยีในเรื่องของ Internet of Things (IOT)

เอกสารอ้างอิง

[1] Robert Szewczyk, Joseph Polastre, Alan Mainwaring and David Culler**,** "Lessons from a Sensor Network Expedition," *1st European Workshop on Wireless Sensor Networks (EWSN '04)*, Berlin, Germany, January 19-21, 2004.

[2] Joseph Polastre, "Design and Implementation of Wireless Sensor Networks for Habitat Monitoring," *Master's Thesis, University of California at Berkeley*, Spring 2003.

[3] Mainwaring Alan, Polastre Joseph, Szewczyk Robert, Culler David and Anderson John, "Wireless Sensor Networks for Habitat Monitoring**,"** *First ACM Workshop on Wireless Sensor Networks and Applications*, Atlanta, GA, USA, September 28, 2002.

[4] Cerpa Alberto, Elson Jeremy, Estrin Deborah, Girod Lewis, Hamilton Michael and Zhao Jerry, "Habitat monitoring: Application driver for wireless communications technology," *ACM SIGCOMM Workshop on Data Communications* in Latin America and the Caribbean, 2001.

[5] Aline Baggio, "Wireless Sensor Networks in Precision Agriculture," *Workshop on Real-World Wireless Sensor Networks (REALWSN'05),* Sweden, June, 2005.

[6] Jenna Burrell, Tim Brooke and Richard Beckwith Vineyard, "Computing: Sensor Networks in Agriculture Production," *IEEE CS and IEEE ComSoc*, pp.38-45, January-March, 2004.

[7] Lizhi Charlie Zhong, Jan Rabaey, Chunlong Guo and Rahul Shah, "Data Link Layer Design for Wireless Sensor Networks," *Proceedings of IEEE MILCOM 2001*, Washington D.C., October 28-31, 2001.

[8] Demirbas Murat, "Wireless Sensor Networks for Monitoring of Large Public Buildings," *Technical Reports at Department of Computer Science and Engineering*, University at Buffalo, December 8, 2005.

[9] T. Nagayama, M. Ruiz-Sandoval, B. Spencer, K. Mechitov and G. Agha, "Wireless strain sensor development for civil infrastructure," *Proceedings of First International Workshop on Networked Sensing Systems*, 2004.

[10] S. Pakzad, S. Kim, G. Fenves, S. Glaser, D. Culler and J. Demmel, "Multi-purpose wireless accelerometers for civil infrastructure monitoring," *5th International Workshop on Structural Health Monitoring (IWSHM)*, 2005.

[11] M. Ruiz-Sandoval, B. Spencer and N. Kurata, "Development of a high sensitivity accelerometer for the mica platform," *Proceedings of International Workshop on Advanced Sensors, Structural Health Monitoring, and Smart Structures*, 2003.

[12] http://www.moteiv.com

[13] Chris Otto, Aleksandar Milenkovic, Corey Sanders and Emil Jovanov, "System Architecture of a Wireless Body Area Sensor Network for Ubiquitous Health Monitoring," *Journal of Mobile Multimedia*, vol. 1, No. 4, pp. 307-326, 2006.

[14] Aleksandar Milenkovic, Chris Otto and Emil Jovanov, "Wireless Sensor Networks for Personal Health Monitoring: Issues and an Implementation," to appear in *Computer Communications* (Special issue: Wireless Sensor Networks: Performance, Reliability, Security, and Beyond), *Elsevier*, 2006.

[15] Emil Jovanov, Aleksandar Milenkovic, Chris Otto and Piet C. de Groen, "A wireless body area network of intelligent motion sensors for computer assisted physical rehabilitation," *Journal of Neuro Engineering and Rehabilitation*, March 1, 2005.

[16] Lorincz K., Malan, D.J. Fulford-Jones, T.R.F. Nawoj, A.k Clavel, A. Shnayder, V. Mainland, G. Welsh, M. and Moulton S., "Sensor networks for emergency response: challenges and opportunities," *IEEE Pervasive Computing*, vol.3, pp.16-23, 2003.

[17] Fulford-Jones, T.R.F., Gu-Yeon Wei and Welsh M, "A portable, low-power, wireless two-lead EKG system," *International Conference of the Engineering in Medicine and Biology Society,* vol.3, pp. 2141- 2144, 2004.

บทที่ 3

สถาปัตยกรรมของโนด

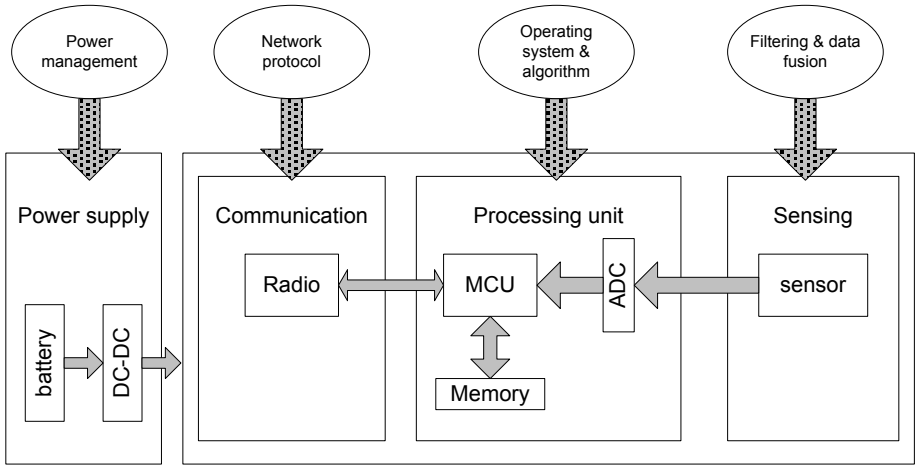
(Node Architecture)

จุดประสงค์การเรียนรู้

เพื่อเข้าใจสถาปัตยกรรมของโนด และสามารถพัฒนาโนดขึ้นใช้งานได้

3.1 สถาปัตยกรรมของโนด

องค์ประกอบที่สำคัญในเครือข่ายเซนเซอร์ไร้สายคือ อุปกรณ์ขนาดเล็กที่เรียกว่าโนด (Node) ซึ่งเป็นฮาร์ดแวร์ที่มีสถาปัตยกรรมดังแสดงในภาพประกอบที่ 3.1 ส่วนแรกคือไมโครคอนโทรลเลอร์ทำหน้าที่ในการประมวลผล (Processing unit) ข้อมูลที่ได้มาจากเซนเซอร์ก่อนที่จะทำการส่งข้อมูลออกไป หรือใช้ในการควบคุมอุปกรณ์ต่อพ่วงอื่นๆ เช่น โมดูลการสื่อสาร และหน่วยความจำ เป็นต้น ส่วนที่สองคือโมดูลสื่อสารแบบไร้สายผ่านทางคลื่นวิทยุ (RF) ส่วนที่สามคือเซนเซอร์ (Sensor) หรือตัวตรวจรับข้อมูล และส่วนสุดท้ายคือแหล่งจ่ายพลังงาน (Power supply) ซึ่งนักพัฒนาเทคโนโลยีเครือข่ายเซนเซอร์ไร้สายสามารถทำการพัฒนาสร้างเซนเซอร์โนดขึ้นใช้เองตามแนวคิดของสถาปัตยกรรมดังกล่าว ทั้งนี้นักพัฒนาจะพิจารณาถึงความเหมาะสมในการนำไปใช้งาน ตัวอย่างเช่น โนดรูปแบบของ Mica [1], Telos [2] และ EYES [3] ซึ่งเป็นโนดขนาดเล็กประหยัดพลังงาน ไม่เหมาะสมกับการประมวลผลที่ซับซ้อน จึงมักจะนำไปใช้ในการตรวจวัดข้อมูลอย่างง่าย ได้แก่ อุณหภูมิ หรือความชื้นในอากาศ เป็นต้น เมื่อได้ข้อมูลมาแล้วก็จะดำเนินการส่งข้อมูลกลับไปยังเครื่องแม่ข่าย หลักสำคัญของการพัฒนาโนดก็คือราคาที่จะต้องถูกเพียงพอสำหรับที่จะใช้เป็นจำนวนมากได้ และโนดชนิดนี้จะขอเรียกว่าโนดขนาดเล็กหรือโนดที่ทำหน้าที่เป็น End-device หรือ Router ในเครือข่าย



ภาพประกอบที่ 3.1 สถาปัตยกรรมของเซนเซอร์โนดไร้สาย

สำหรับโนดที่เพิ่มความสามารถในการประมวลผลมากขึ้น อาจจะใช้ไมโครคอนโทรลเลอร์ขนาด 16 บิต มีหน่วยความจำมากขึ้น และมีพอร์ตสำหรับให้สามารถต่ออุปกรณ์เพิ่มเติมได้ โนดชนิดนี้จึงเหมาะกับงานที่ต้องการการประมวลผลซับซ้อนได้แก่ การประมวลผลภาพ และการประมวลผลสัญญาณเป็นต้น ตัวอย่างของโนดชนิดนี้ได้แก่ SunSPOT [4] และ IMote [5] ขอเรียกโนดชนิดนี้ว่าโนดขนาดกลาง โนดชนิดนี้อาจจะถูกนำไปใช้เป็นโนดแม่ (Cluster head, CH) ของเครือข่ายโดยตัดส่วนของพอร์ตที่จะให้เซนเซอร์ออกไปเพราะไม่จำเป็นต้องใช้งานถ้าทำหน้าที่เป็นโนดแม่

จะพบว่าการออกแบบและพัฒนาโนดไร้สายนั้นสามารถยืดหยุ่นและไม่มีกฎเกณฑ์ตายตัว ผสมผสานเทคโนโลยีที่เหมาะสมกับงานที่จะใช้ ดังนั้นการสร้างโนดไร้สายจึงไม่นิยมเป็นเรียกเป็น Platform เนื่องจากมีหลากหลายรูปแบบ และที่สำคัญโนดแต่ละรูปแบบก็ไม่สามารถสื่อสารกันได้ ดังนั้นจึงจำเป็นที่จะต้องมีอุปกรณ์ช่วยคือ Gateway ที่จะทำหน้าที่เป็นตัวกลางในการสื่อสารระหว่างโนดที่หลากหลายรูปแบบ รวมทั้งสื่อสารระหว่างโนดภายในเครือข่ายเซนเซอร์ไปยังเครือข่ายอื่นๆภายนอก

เนื่องจากโนดในเครือข่ายเซนเซอร์ไร้สายจะใช้พลังงานจากแบตเตอร์รี่ (ยกเว้นในบางงานประยุกต์ที่โนดมีแหล่งจ่ายพลังงานที่ไม่จำกัดเช่น ใช้พลังงานจากไฟบ้านได้โดยตรง เป็นต้น) ดังนั้นการเลือกใช้และพัฒนาโนดจะต้องคำนึงถึงเรื่องของการใช้พลังงานเป็นสำคัญ การพิจารณาเลือกใช้ไมโครคอนโทรลเลอร์ควรจะเลือกใช้งานไมโครคอนโทรลเลอร์ที่มีหลายโหมดการทำงาน เนื่องจากเวลาส่วนใหญ่ในระบบเครือข่ายเซนเซอร์ไร้สายโนดจะอยู่ในสภาวะ Idle เพราะเมื่อทำการอ่านค่าจากเซนเซอร์แล้วก็สามารถให้ไมโครคอนโทรลเลอร์เข้าสู่สภาวะพัก เปิดไว้

เฉพาะส่วนของโมดูลการสื่อสาร เพื่อรอรับสัญญาณร้องขอให้ตื่นจากโนดอื่น การทำเช่นนี้ก็จะช่วยประหยัดพลังงานได้ นอกจากนี้ยังสามารถจัดการให้ไมโครคอนโทรลเลอร์และโมดูลการสื่อสารเข้าสู่สภาวะพักทั้งคู่ไปจนกว่าจะถึงเวลาอ่านค่าจากเซนเซอร์ในรอบถัดไป การจัดการเช่นนี้ก็จะช่วยให้ประหยัดพลังงานได้มากขึ้น แต่จะต้องคำนึงถึงการตื่นขึ้นมาทำงาน (Wake up) ว่าจะต้องทำให้รวดเร็วเพื่อสามารถอ่านค่าจากเซนเซอร์ได้ทัน นอกจากนี้จะต้องพิจารณาถึงปัจจัยอื่นๆ เช่น โนดตัวนั้นจะต้องไม่ใช่ทางผ่านข้อมูลของโนดอื่น เพราะไม่เช่นนั้นถ้าโนดเข้าสู่สภาวะพักทั้งหมดจะทำให้ข้อมูลไม่สามารถส่งต่อข้อมูลผ่านไปยังปลายทางได้

สำหรับการเลือกใช้โมดูลรับส่งแบบไร้สายด้วยคลื่นความถี่วิทยุก็ต้องคำนึงถึงเรื่องการใช้พลังงานเช่นกัน ควรเลือกใช้โมดูลภาครับส่งคลื่นวิทยุที่มีโหมดการทำงานที่หลากหลายเช่นเดียวกับไมโครคอนโทรลเลอร์ เพื่อให้สามารถเข้าสู่สภาวะพักในช่วงเวลาที่ไม่จำเป็นต้องการใช้งาน นอกจากนี้แล้วการเลือกใช้โมดูลภาครับส่งคลื่นวิทยุจะต้องสอดคล้องกับหลักการของเครือข่ายเซนเซอร์ไร้สาย หมายถึงโมดูลจะต้องรองรับมาตรฐาน IEEE 802.15.4 (จะกล่าวรายละเอียดไว้ในบทที่ 5) ที่จะทำให้มีการเชื่อมต่อแบบเครือข่ายได้ การเลือกโมดูลภาครับส่งคลื่นวิทยุจำเป็นที่จะต้องพิจารณาถึงความแรงในการส่งสัญญาณประกอบด้วย เพราะความแรงของสัญญาณจะแปรผันโดยตรงกับระยะทางของการส่งข้อมูล แต่ก็จะแปรผันตรงกับการใช้พลังงานด้วยเช่นกัน กล่าวคือถ้ามีความแรงของสัญญาณมากก็จะส่งข้อมูลไปได้ไกล แต่จะเสียพลังงานมากเช่นกัน ดังนั้นนักพัฒนาจึงควรที่จะต้องเลือกให้เหมาะสมกับงาน

ดังนั้นสำหรับแนวทางของการวิจัยและพัฒนาของเทคโนโลยีเครือข่ายเซนเซอร์ไร้สายจึงจะเกี่ยวข้องกับทุกปัจจัยที่ส่งผลไปถึงการใช้พลังงานทั้งสิ้น ตัวอย่างแนวทางของการวิจัยมีดังนี้

- พัฒนาให้มีการใช้พลังงานอย่างมีประสิทธิภาพในโนด
- พัฒนาให้มีการใช้พลังงานอย่างมีประสิทธิภาพในแต่ละระดับของชั้นเครือข่าย
- สร้างเซนเซอร์หรือตัวตรวจวัดที่ประหยัดพลังงาน
- พัฒนาเทคโนโลยีของแบตเตอร์รี่ให้สามารถจุพลังงานได้มาก มีขนาดเล็ก หรือการจัดการพลังงาน หรือการหาพลังงานจากสิ่งแวดล้อม (Energy harvesting) เป็นต้น
- นอกจากนี้เครือข่ายเซนเซอร์ไร้สายจะต้องมีระบบที่สามารถจัดการตัวเองได้อัตโนมัติ เพราะในบางงานประยุกต์เครือข่ายเซนเซอร์ไร้สายจะอยู่ในที่ห่างไกล หรือในบางงานประยุกต์ที่เครือข่ายเซนเซอร์มีจำนวนโนดจำนวนมาก (มากกว่า 100 ตัว) การเข้าไปจัดการหรือซ่อมบำรุงที่ตัวโนดจึงไม่สามารถทำได้ จำเป็นที่โนดจะต้องมีความสามารถดังต่อไปนี้ เช่น การปรับเทียบค่าที่ได้จากเซนเซอร์ได้เอง (Self calibration)

ระบุตัวตนได้เอง (Self identification) ค้นหาบริการที่มีอยู่บนเครือข่ายได้เอง (Self discovery) หรือวินิจฉัยความผิดปกติของสิ่งที่ตรวจวัดได้ที่ตัวโนดเอง (Self diagnosis) เป็นต้น

- งานอื่นๆ ที่เกี่ยวข้อง เช่น รูปแบบการเชื่อมต่อระหว่างเครือข่ายเซนเซอร์ไร้สายกับเครือข่ายอื่นๆ การออกแบบมาตรฐานการเชื่อมต่อของเซนเซอร์ และความสามารถในการขยายจำนวนโนดในเครือข่าย (Scalability) เป็นต้น

3.2 เซนเซอร์

ตัวตรวจวัดหรือเรียกกันทั่วไปว่าเซนเซอร์ ใช้สำหรับตรวจวัดข้อมูลทางกายภาพที่ต้องการตัวอย่างเช่น เซนเซอร์วัดความเร่ง (Accelerometer) เซนเซอร์วัดสัญญาณกล้ามเนื้อ (EMG: Electromyography) เซนเซอร์วัดอุณหภูมิ และความชื้นในอากาศ (Temperature and Humidity) เซนเซอร์วัดความชื้นในดิน (Soil moisture) เป็นต้น เซนเซอร์เหล่านี้จะทำการวัดค่าในทางไฟฟ้าซึ่งอาจจะเป็นค่าความต้านทาน ค่าความจุของประจุ หรือค่าความต่างศักย์ทางไฟฟ้า เป็นต้นและให้ไมโครคอนโทรลเลอร์ทำหน้าที่แปลค่าทางไฟฟ้าเป็นค่ามาตรฐานที่ต้องการ โดยไมโครคอนโทรลเลอร์ที่ทำหน้าที่ในส่วนนี้อาจจะเป็นตัวเดียวกันกับไมโครคอนโทรลเลอร์ของโนด หรืออาจจะใช้ไมโครคอนโทรลเลอร์ที่แยกจากกัน ซึ่งถ้าหากค่าที่ได้จากเซนเซอร์เป็นค่าทางไฟฟ้าและจะต้องแปลค่าที่ไมโครคอนโทรลเลอร์ของโนดก็จะนิยมใช้การเชื่อมต่อผ่านทางพอร์ตมาตรฐานเช่น ADC หรือขา I/O ทั่วไป แต่ถ้าหากบนเซนเซอร์เหล่านั้นมีการแปลค่าด้วยไมโครคอนโทรลเลอร์เรียบร้อยแล้วจะใช้การส่งค่าที่อ่านได้ให้กับโนดได้ทันทีผ่านทางการเชื่อมต่อมาตรฐานเช่น UART หรือ SPI หรือ I^2C เป็นต้น

3.3 ไมโครคอนโทรลเลอร์

ส่วนสำคัญของโนดคือไมโครคอนโทรลเลอร์ที่ทำหน้าที่สำหรับการประมวลผล และการควบคุมให้โนดทำงานตามที่ต้องการ ซึ่งการเลือกใช้ไมโครคอนโทรลเลอร์ก็ควรจะต้องให้เหมาะสมกับงานประยุกต์ที่จะนำไปใช้ สาเหตุที่โนดนิยมใช้ไมโครคอนโทรลเลอร์มากกว่าใช้งานไมโครโพรเซสเซอร์ก็เนื่องจากว่า ไมโครคอนโทรลเลอร์นั้นมีส่วนประกอบอื่นที่จำเป็นสำหรับการทำงานของโนดตัวอย่างเช่น ส่วนของการเชื่อมต่อที่หลากหลายเช่น ADC, UART, I2C หรือ SPI เป็นต้น มีส่วนของหน่วยความจำภายในของไมโครคอนโทรลเลอร์ทั้งแบบ RAM ROM หรือ Flash มีส่วนการจัดการพลังงาน (Power management unit) เป็นต้น แต่ทั้งนี้หากผู้พัฒนาต้องการความสามารถของโนดมากกว่าการทำงานปกติเช่น ต้องการให้สามารถประมวลผลงานที่ซับซ้อนทางด้านการประมวลผลภาพ หรือการประมวลสัญญาณ ก็สามารถพิจารณานำโพรเซสเซอร์ประมวลผลสัญญาณ (Digital

Signal Processor, DSP) หรืออุปกรณ์ที่สามารถโปรแกรมได้ (Field Programmable Gate Arrays, FPGAs) มาใช้บนโนดได้ แต่ทั้งนี้จะต้องพึงระวังว่าการเพิ่มความสามารถเหล่านี้จะส่งผลต่อการใช้พลังงานของโนดด้วยเช่นกัน

3.3.1 การเชื่อมต่อกับภาครับส่งคลื่นวิทยุ

การเลือกใช้โมดูลภาครับส่งคลื่นวิทยุจำเป็นที่จะต้องคำนึงถึงการใช้พลังงาน ดังนั้นเทคโนโลยีภาคการรับส่งแบบ RF (Radio Frequency) CMOS จึงถูกใช้งานอย่างแพร่หลาย สำหรับคลื่นความถี่ที่ได้รับอนุญาตให้ใช้งานได้ในประเทศไทยคือ 2.4 GHz ซึ่งเป็นย่าน ISM (Industrial, Scientific and Medical) สำหรับการเชื่อมต่อภาครับส่งคลื่นวิทยุเข้ากับไมโครคอนโทรลเลอร์จะต้องคำนึงถึงความเร็วและการใช้พลังงานให้มีประสิทธิภาพ ดังนั้นจึงควรเลือกใช้การเชื่อมต่อแบบอนุกรม (Serial) มากกว่าการเชื่อมต่อแบบขนาน (Parallel) ถึงแม้ว่าการเชื่อมต่อแบบขนานจะสามารถส่งข้อมูลได้เร็วกว่าแบบอนุกรม แต่การเชื่อมต่อแบบขนานจะใช้ขนาดของบัสส่งข้อมูลจำนวนมาก ทำให้สิ้นเปลืองพลังงานและไมโครคอนโทรลเลอร์จำเป็นต้องมี I/O จำนวนมาก การเชื่อมต่อแบบอนุกรมที่นิยมใช้งานคือ SPI (Serial Peripheral Interface) และ I2C (Inter-Integrated Circuit) ในหนังสือนี้ขอกล่าวถึงแต่การเชื่อมต่อแบบอนุกรมชนิด SPI เพราะชิปโมดูลภาครับส่งคลื่นวิทยุของบริษัท ChipCon เช่น CC2420 หรือ CC2430 หรือ CC2530 เป็นต้น ได้รับความนิยมเลือกใช้งานจำนวนมากนั้นมีการเชื่อมระหว่างชิปกับไมโครคอนโทรลเลอร์ผ่านทาง SPI

การเชื่อมต่อแบบอนุกรมชนิด SPI จะมี 4 ขาสัญญาณได้แก่ MOSI (Master-Out/Slave-In), MISO (Master-In/Slave-out), SCLK (Serial Clock) และ CS (Chip Select) ขาสัญญาณ MOSI ใช้สำหรับการส่งข้อมูลจากอุปกรณ์หลัก (Master) ไปยังอุปกรณ์ลูก (Slave) ถ้าอุปกรณ์ถูกตั้งค่าให้เป็นอุปกรณ์หลัก ในทางตรงกันข้ามถ้าอุปกรณ์ถูกตั้งค่าให้เป็นอุปกรณ์ลูก ขาสัญญาณ MOSI นี้จะถูกใช้สำหรับรับข้อมูลจากอุปกรณ์หลัก สำหรับการทำงานของสัญญาณ MISO นั้นก็จะทำงานตรงกันข้ามกับ MOSI ส่วนขาสัญญาณ SCLK อุปกรณ์หลักจะใช้ในการส่งสัญญาณนาฬิกานี้ไปยังอุปกรณ์ลูกตัวอื่นๆเพื่อให้สามารถทำงานสอดคล้องกันได้ หลักการทำงานของ SPI จะเริ่มต้นจากอุปกรณ์หลักส่งสัญญาณผ่านทางขา CS ไปยังอุปกรณ์ลูกว่าต้องการทำการสื่อสาร สำหรับเซนเซอร์โนดไมโครคอนโทรลเลอร์จะถือว่าเป็นอุปกรณ์หลัก และชิปโมดูลภาครับส่งคลื่นวิทยุเป็นอุปกรณ์ลูก นอกจากนี้ SPI จะมีอุปกรณ์หลักได้เพียง 1 ตัว ดังนั้นถ้าหากมีโมดูลหน่วยความจำอื่นพ่วงต่อใน SPI ด้วย หน่วยความจำจะสื่อสารกับชิปโมดูลภาครับส่งคลื่นวิทยุได้จะต้องผ่านไมโครคอนโทรลเลอร์

SPI รองรับโพรโทคอลการสื่อสารแบบเข้าจังหวะ (Synchronous) ดังนั้นเพื่อให้สามารถทำงานสอดคล้องกันระหว่างไมโครคอนโทรลเลอร์และโมดูลภาครับส่งคลื่นวิทยุ ขาสัญญาณ SCLK จะต้องถูกตั้งค่าเป็นค่าความถี่สูงสุดของสัญญาณนาฬิกาของโมดูลภาครับส่งคลื่นวิทยุ นอกจากนี้แล้วอุปกรณ์ทั้งสองส่วนจะต้องทำงานร่วมกันผ่านทาง 2 ตัวแปรคือ CPOL (Clock Polarity) และ CPHA (Clock Phase) ค่า CPOL คือการตรวจสอบว่าขาสัญญาณนาฬิกานั้นทำงานแบบ active-high หรือ active-low ส่วนค่า CPHA ใช้กำหนดเวลาเมื่อข้อมูลในรีจิสเตอร์มีการเปลี่ยนแปลงและเมื่อข้อมูลที่เขียนใส่รีจิสเตอร์ได้ถูกอ่านออกไป ดังนั้นทั้ง 2 ค่าตัวแปรนี้จึงสามารถเกิดเป็นรูปแบบการทำงานของ SPI ได้ 4 โหมด ดังตารางต่อไปนี้

โหมดการทำงาน SPI	CPOL	CPHA	คำอธิบาย
0	0	0	SCLK ทำงาน active low Sampling ที่ขอบขาของสัญญาณนาฬิกาลูกคลื่นที่เป็นคี่ ส่วนข้อมูลจะเปลี่ยนแปลงตอนขอบขาสัญญาณนาฬิกาลูกคลื่นที่เป็นคู่
1	0	1	SCLK ทำงาน active low Sampling ที่ขอบขาของสัญญาณนาฬิกาลูกคลื่นที่เป็นคู่ ส่วนข้อมูลจะเปลี่ยนแปลงตอนขอบขาสัญญาณนาฬิกาลูกคลื่นที่เป็นคี่
2	1	0	SCLK ทำงาน active high Sampling ที่ขอบขาของสัญญาณนาฬิกาลูกคลื่นที่เป็นคี่ ส่วนข้อมูลจะเปลี่ยนแปลงตอนขอบขาสัญญาณนาฬิกาลูกคลื่นที่เป็นคู่
3	1	1	SCLK ทำงาน active high Sampling ที่ขอบขาของสัญญาณนาฬิกาลูกคลื่นที่เป็นคี่ ส่วนข้อมูลจะเปลี่ยนแปลงตอนขอบขาสัญญาณนาฬิกาลูกคลื่นที่เป็นคู่

3.3.2 หน่วยความจำ

ที่เก็บข้อมูลของโนดนิยมใช้หน่วยความจำภายในไมโครคอนโทรลเลอร์เพื่อเป็นการประหยัดพื้นที่ของบอร์ดและประหยัดพลังงาน ดังนั้นจึงไม่นิยมเก็บข้อมูลจำนวนมากไว้บนโนด ในขณะที่หน่วยความจำภายใน

ไมโครคอนโทรลเลอร์อาจจะเป็นปัจจัยสำคัญในการเลือกใช้ตระกูลหรือรุ่นของไมโครคอนโทรลเลอร์ ตัวอย่างเช่น ถ้าต้องการให้โนดมีระบบปฏิบัติการขนาดเล็กดังนั้นหน่วยความจำชนิด ROM จะต้องมีขนาดใหญ่เพียงพอที่จะใส่ระบบปฏิบัติการนั้น โนด Telos จะใช้หน่วยความจำชนิด ROM ในการเก็บโปรแกรมและระบบปฏิบัติการ TinyOS อยู่ที่ 48 Kbytes สำหรับโนดที่ต้องการมีการทำงานที่รวดเร็วอาจจะต้องพิจารณาเพิ่มหน่วยความจำชนิด RAM ให้มีขนาดใหญ่ขึ้น ตัวอย่างเช่นโนด SunSPOT ใช้หน่วยความจำชนิด RAM สูงถึง 256 Kbytes ในขณะที่ใช้พื้นที่เก็บโปรแกรมและระบบปฏิบัติการที่ 2 Mbytes แต่ถ้าในบางงานประยุกต์ที่ต้องการเก็บข้อมูลที่ได้จากเซนเซอร์ไว้ในโนดก่อน ก็สามารถที่จะพิจารณาการต่อพ่วงหน่วยความจำเพิ่มเช่น หน่วยความจำชนิด Flash ได้เช่นกัน สรุปได้ว่าหน่วยความจำบนโนดจะมีหน้าที่ 3 ส่วนด้วยกันคือ 1) เก็บโปรแกรมหรือระบบปฏิบัติการนิยมใช้ ROM ภายในไมโครคอนโทรลเลอร์ 2) เก็บตัวแปรในระหว่างการประมวลผล (ใช้หน่วยความจำชนิด RAM ภายในไมโครคอนโทรลเลอร์) และ 3) เก็บข้อมูลชั่วคราวที่ได้จากเซนเซอร์ (พิจารณาใช้การต่อพ่วง Flash)

3.4 โมดูลภาครับส่งคลื่นวิทยุ

การเลือกใช้โมดูลภาครับส่งคลื่นวิทยุควรจะต้องคำนึงถึงการนำโนดไปใช้งานว่าจำเป็นที่จะต้องใช้ในลักษณะใด ซึ่งสามารถสรุปสิ่งที่ควรจะต้องพิจารณาไว้ได้ดังต่อไปนี้

- ความสามารถในการเชื่อมต่อกับชั้นเครือข่ายที่สูงขึ้น โดยส่วนมากจะต้องรองรับไปจนถึงระดับชั้น Medium Access Control (MAC) เนื่องจากภาครับส่งคลื่นวิทยุจะเชื่อมต่อกับไมโครคอนโทรลเลอร์และส่งข้อมูลในระดับบิตหรือไบต์ ดังนั้นการยอมให้ชั้น MAC สามารถเข้ามาจัดการเฟรมของการส่งข้อมูล หรือสามารถเข้าถึงตัวข้อมูลที่เก็บไว้ในหน่วยความจำชั่วคราวได้

- พลังงานเป็นอีกปัจจัยที่จะต้องคำนึงถึงเวลาเลือกใช้โมดูลภาครับส่งคลื่นวิทยุ เนื่องจากโมดูลภาครับส่งคลื่นวิทยุนี้เป็นส่วนที่ใช้พลังงานมากที่สุดบนสถาปัตยกรรมของโนด และที่สำคัญโมดูลนี้จะต้องสามารถเลือกปรับเปลี่ยนโหมดการทำงานของตัวเองได้เช่น เมื่อไม่ได้รับส่งข้อมูลก็สามารถเข้าสู่สภาวะ Sleep ได้ เป็นต้น

- ความถี่และการมีหลายช่องสัญญาณเป็นส่วนที่จะต้องพิจาณา ในหลายประเทศเช่นประเทศไทยไม่สามารถใช้งานที่ความถี่ช่วง 900 MHz ได้ ดังนั้นการเลือกใช้ความถี่ที่ 2.4 GHz จึงได้รับความนิยม นอกจากนี้หากต้องการการใช้งานช่องสัญญาณมากกว่า 1 ช่องสัญญาณ (Multiple channels) ในการสร้างเทคนิคบางประการก็จำเป็นที่จะต้องเลือกโมดูลภาครับส่งคลื่นวิทยุให้สอดคล้องกันด้วย

- อัตราการรับส่งข้อมูล โดยมากจะถูกกำหนดมากับความถี่ของคลื่นสัญญาณที่เลือกใช้ เช่น ถ้าเลือกใช้คลื่นความถี่ที่ 2.4 GHz อัตราการรับส่งข้อมูลจะอยู่ที่ 250 kbps

- Gain หรืออัตราของกำลังของสัญญาณส่งต่อกำลังของสัญญาณขารับ มีหน่วยเป็น dB ซึ่งถ้าโนดต้องการให้มีกำลังส่งมากๆสามารถออกแบบวงจรขยายกำลังส่งเพิ่มเติมได้

ตัวอย่างของโมดูลภาครับส่งคลื่นวิทยุสำหรับโนดในเครือข่ายเซนเซอร์ไร้สายที่ได้รับความนิยมมีดังนี้

RFM เป็นโมดูลภาครับส่งคลื่นวิทยุในตระกูล TR1000 [6] ใช้ความถี่ในช่วง 868 – 916 MHz สามารถส่งข้อมูลในอัตราสูงสุดไม่เกิน 115.2 kbps ทำการมอดูเลชั่นทั้งแบบ on-off-keying หรือ แบบ ASK กำลังส่งสูงสุดที่ 1.5 dBm หรือประมาณ 1.4 mW โนด Mica ใช้โมดูลภาครับส่งคลื่นวิทยุ RFM TR1000 นี้

Chipcon [7] นำเสนอโมดูลภาครับส่งคลื่นวิทยุในตระกูล CC1000และ CC2420 เป็นต้น ซึ่งได้รับความนิยมใช้อย่างแพร่หลายในเครือข่ายเซนเซอร์ไร้สาย ชิปตระกูล CC1000 สามารถทำงานได้ที่ความถี่ตั้งแต่ 300 – 1000 MHz และใช้มอดูเลชั่นแบบ FSK รวมทั้งให้ข้อมูลค่าความแรงของสัญญาณ (RSSI, Received Signal Strength Indicator) และสามารถโปรแกรมกำลังส่งได้ สำหรับชิปตระกูล CC2420 เป็นโมดูลภาครับส่งคลื่นวิทยุที่รองรับการทำงานชั้นกายภาพ (Physical layer) ตามมาตรฐาน IEEE802.15.4 ซึ่งถือได้ว่าเป็นชิปตัวแรกที่รองรับมาตรฐานนี้ของเครือข่ายเซนเซอร์ไร้สาย ทำงานที่ความถี่ 2.4 GHz มีอัตราการส่งข้อมูลที่ 250 kbps ดังนั้นจึงได้รับความนิยมใช้งานอย่างแพร่หลาย ปัจจุบันสามารถเลือกหาซื้อโมดูลภาครับส่งคลื่นวิทยุของ Chipcon ได้จากบริษัท TI โนดตระกูล Mica2 หรือ Telos ก็ใช้โมดูล CC2420 นี้

Ember [8] เป็นอีกเจ้าของเทคโนโลยีโมดูลภาครับส่งคลื่นวิทยุที่ได้รับความนิยมมีใช้อยู่ใน XBee ของบริษัท Digi เช่น Ember EM2420 กำลังการส่งที่ -0.5 dBm ทำงานที่ 3.3 V มีการใช้กระแสในช่วงการส่งอยู่ที่ประมาณ 22.7 mA และกระแสช่วงรับข้อมูลที่ 25.2 mA อีกทั้งสามารถทำงานในโหมด Sleep และใช้กระแสเพียง 12 uA สำหรับการทำงานที่ความถี่ 2.4 GHz สามารถเลือกช่องสัญญาณได้ 16 ช่อง และทำการมอดูเลชั่นแบบ BPSK (Binary Phase Shift Keying)

3.5 พลังงานของโนด (Energy Consumption of Node)

การใช้พลังงานของโนดเป็นสิ่งสำคัญเนื่องจากโนดมีแหล่งจ่ายพลังงานที่จำกัดเช่น ได้รับพลังงานจากแบตเตอร์รี่ เป็นต้น ดังนั้นถ้าส่วนที่ใช้พลังงานหลักของโนดก็จะทำให้สามารถพัฒนาเทคนิคในการประหยัด

พลังงานหรือเทคนิคการให้โมดูลส่วนนั้นใช้พลังงานน้อยลง ซึ่งจะส่งผลให้อายุการทำงานของเครือข่ายยาวนานขึ้น เมื่อพิจารณาความสัมพันธ์ระหว่างแหล่งจ่ายพลังงานกับการใช้พลังงานของโนด อาจจะทำได้อย่างง่ายเช่น โนดมีแบตเตอร์รี่ที่เก็บความจุได้ 2 J ดังนั้นโนดจะต้องใช้พลังงานประมาณไม่เกิน 2/24*60*60 = 23 uW (อย่างต่อเนื่อง) ซึ่งเป็นอัตราการใช้พลังงานที่ต่ำมาก ดังนั้นการออกแบบให้เกิดการใช้พลังงานอย่างประหยัดจึงเป็นสิ่งจำเป็นอีกทั้งจำเป็นที่จะต้องมีระบบจัดการการใช้กำลังงาน (Power management) หรือโหมดการทำงานในสอดคล้องกับความจำเป็นเช่น เมื่อไม่มีการทำงานใดๆโนดก็ควรจะเข้าสู่สภาวะหลับ (Sleep) เป็นต้น

เมื่อพิจารณาตามโมดูลที่มีใช้งานบนโนดดังนั้นเราสามารถเลือกพิจารณาการใช้พลังงานอย่างมีประสิทธิภาพได้ดังนี้

ไมโครคอนโทรลเลอร์ ควรเลือกใช้ไมโครคอนโทรลเลอร์ที่มีอัตราการใช้พลังงานต่ำ และรองรับการทำงานในหลายโหมด ตัวอย่างเช่น ไมโครคอนโทรลเลอร์ของ TI ในตระกูล MSP430 มีอัตราการใช้พลังงานที่ 1.2 mW (ทำงานที่ความถี่ 1 MHz แรงดันไฟฟ้าที่ 3 V) และมีโหมดการหลับอยู่ถึง 4 ระดับ ได้แก่ LPM0, LPM2, LPM3 และ LPM4 ตัวอย่างการทำงานได้แก่ โหมด Deep sleep (LPM4) มีอัตราการใช้พลังงานที่ 0.3 uW สามารถปลุกให้ทำงานด้วยอินเตอร์รัปป์ภายนอก (External interrupt), LPM3 ในการหลับชั้นนี้สัญญาณนาฬิกายังคงทำงานตามปกติ ดังนั้นสามารถใช้การปลุกด้วยวิธีการตั้งตารางเวลาไว้ (Schedule) ซึ่งมีอัตราการใช้พลังงานที่ 6 uW ดังนั้นเมื่อสามารถเลือกได้ว่าจำเป็นที่จะต้องใช้ไมโครคอนโทรลเลอร์ขนาดกี่บิต สิ่งที่จะต้องพิจารณานอกเหนือจากขนาดของหน่วยความจำภายในไมโครคอนโทรลเลอร์แล้วคือ การพิจารณาอัตราการใช้พลังงานในแต่ละโหมดด้วย สำหรับไมโครคอนโทรลเลอร์ขนาดใหญ่อาจจะต้องพิจารณาเทคนิคของการทำ Dynamic Voltage Scaling (DVS) ที่สามารถลดอัตราการใช้พลังงานด้วยการลดระดับแรงดันไฟฟ้า แต่การทำเช่นนี้ความเร็วของสัญญาณนาฬิกาจะลงด้วย

หน่วยความจำ พลังงานที่จะต้องสูญเสียในการใช้หน่วยความจำคือการอ่านและเขียนข้อมูลจากหน่วยความจำ โดยทั่วไปนิยมใช้หน่วยความจำภายในไมโครคอนโทรลเลอร์ดังนั้น อัตราการใช้พลังงานของหน่วยความจำจึงจะรวมอยู่กับไมโครคอนโทรลเลอร์อยู่แล้ว สำหรับหน่วยความจำชนิด Flash ที่นิยมต่อพ่วงกับไมโครคอนโทรลเลอร์จะต้องพิจารณาในเรื่องของการเขียนข้อมูลลงหน่วยความจำชนิดนี้ เนื่องจากมีกระบวนการที่ซับซ้อนจึงทำให้สูญเสียการใช้พลังงานมาก ซึ่งทำให้ควรหลีกเลี่ยงการเขียนลงหน่วยความจำหากไม่จำเป็น

ภาครับส่งคลื่นวิทยุ หน้าที่ของโมดูลภาครับส่งคลื่นวิทยุคือการรับและการส่งข้อมูล เวลาส่วนใหญ่ของการทำงานของโมดูลภาครับส่งคลื่นวิทยุในโนดจะถูกปิด หรือมี Duty cycle ที่ต่ำ การเลือกใช้โมดูลภาครับส่งคลื่นวิทยุจะต้องพิจารณาอัตราการใช้พลังงานในการรับและการส่งข้อมูล ตัวอย่างเช่น โมดูลของ RFM TR1000 จะมีอัตราการใช้พลังงานสำหรับส่งข้อมูลที่ 1 นJ สำหรับส่งข้อมูล 1 บิต และใช้พลังงาน 0.5 นJ สำหรับรับข้อมูล 1 บิต [9] ซึ่งข้อมูลเหล่านี้สามารถตรวจสอบได้จาก Datasheet ของโมดูลที่เลือกใช้ ดังนั้นจึงแนะนำให้ตรวจสอบอัตราการใช้พลังงานในการรับหรือส่งข้อมูลต่อ 1 บิต เพื่อให้สามารถเลือกโมดูลภาครับส่งคลื่นวิทยุได้อย่างเหมาะสม

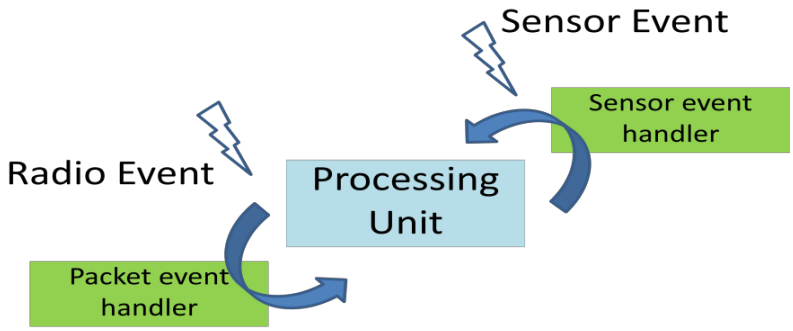
3.6 ระบบปฏิบัติการและทำงานของโปรแกรม

ระบบปฏิบัติการที่ใช้ในโนดจะเรียกว่าเป็นระบบปฏิบัติการสมองกลฝังตัว (Embedded Operating System) ที่ไม่ใช่ระบบปฏิบัติการที่ใช้งานในเครื่องคอมพิวเตอร์ทั่วไป เนื่องจากมีทรัพยากรของโนดที่จำกัด ทั้งส่วนของไมโครคอนโทรลเลอร์ที่ประมวลผลไม่ซับซ้อน และขนาดของหน่วยความจำมีจำกัด โดยระบบปฏิบัติการสมองกลฝังตัวนี้เป็นเพียงทางเลือกของโนดเท่านั้น ไม่จำเป็นที่โนดในเครือข่ายเซนเซอร์ไร้สายจะต้องมี โดยให้พิจารณาความจำเป็นจากความซับซ้อนของการใช้งาน หากต้องการป้องกันและควบคุมการใช้ทรัพยากรฮาร์ดแวร์บนโนดจากผู้ใช้ที่มีหลากหลายก็ควรใช้งานระบบปฏิบัติการสมองกลฝังตัว ตัวอย่างของการใช้ระบบปฏิบัติการสมองกลฝังตัวควบคุมทรัพยากรบนโนดได้แก่ ควบคุมการทำ Dynamic Voltage Scaling (DVS) หรือการจัดการของเครือข่าย เป็นต้น

การโปรแกรมโนดในรูปแบบ Concurrent programming เพื่อให้แต่ละโปรแกรมย่อยสามารถทำงานได้พร้อมๆกัน เช่นในขณะที่ประมวลผลข้อมูลก็ยังสามารถใช้ทรัพยากรที่ไม่เกี่ยวข้องทำการรับส่งข้อมูลไปพร้อมๆกัน ถ้าให้โปรแกรมทำงานในรูปแบบ Polling เมื่อไปอ่านข้อมูลจากเซนเซอร์แล้วจะต้องทำการรับส่งข้อมูลด้วยอาจจะมีโอกาสที่ทำให้การอ่านเซนเซอร์หรือการรับส่งข้อมูลผิดพลาดได้ ดังนั้นจึงมีการนำเสนอการโปรแกรมแบบ Process-based concurrency ทำให้สามารถเกิดการทำงานหลายๆอย่างพร้อมกันได้บนหน่วยประมวลผลเดี่ยว (Single CPU) เมื่อต้องการอ่านค่าจากเซนเซอร์ พร้อมๆกับการรับส่งข้อมูลก็จะต้องสร้างโพรเซสสำหรับทั้งสองงานขึ้นคือ โพรเซส Sensor (Handle sensor process) และโพรเซส Packet (Handle packet process) โดยมีระบบปฏิบัติการทำหน้าที่เป็นตัวสลับการทำงานระหว่างสองโพรเซส การทำงานของโปรแกรมชนิดนี้มีข้อเสียในกรณีที่แต่ละโพรเซสมีการทำงานเพียงเล็กน้อยแต่จะต้องเสียเวลาในจังหวะการสลับการทำงานของแต่ละโพรเซส

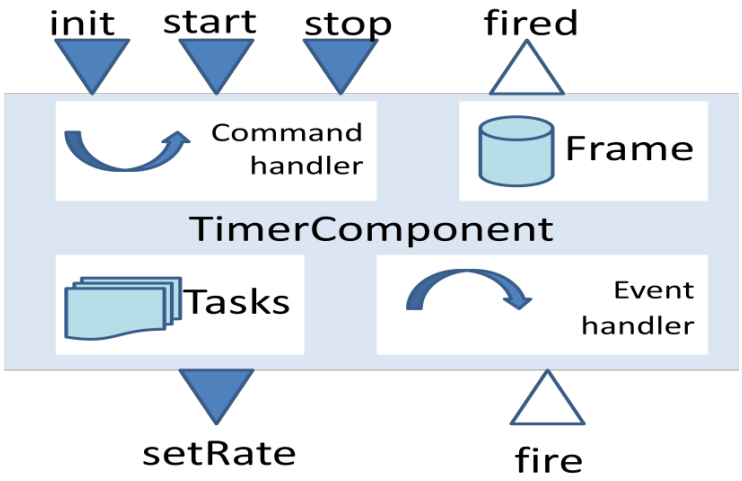
และแต่ละโพรเซสจะต้องมีหน่วยความจำส่วนตัว ดังนั้นจึงเป็นข้อเสียเปรียบของโปรแกรมแบบ Process-based concurrency

อีกรูปแบบของการโปรแกรมคือ Event-based programming [9] ดังแสดงในภาพประกอบที่ 3.2 จะมีการเก็บลำดับของคำสั่งที่จะไปเรียกแต่ละ Event ขึ้นมาทำงาน เมื่อมีการเรียกใช้เซนเซอร์หรือภาครับส่งคลื่นวิทยุ โดยในส่วนของโปรแกรมในแต่ละ Handler ก็ควรจะต้องสั้น และแต่ละ Handler จะไม่สามารถอินเตอร์รัปป์กันเองได้ เพื่อป้องกันไม่ให้กระบวนการจัดการการอินเตอร์รัปป์ต้องยุ่งยาก และการเชื่อมต่อกับระบบปฏิบัติการจะกระทำผ่าน Application Programming Interface (API)



ภาพประกอบที่ 3.2 โมเดลของ Event-based programming

ตัวอย่างของระบบปฏิบัติการสมองกลฝังตัวในเครือข่ายเซนเซอร์ไร้สายที่รองรับรูปแบบโปรแกรมแบบ Event-based programming คือ TinyOS [10] ที่ใช้ภาษา NesC [11] ระบบปฏิบัติการ TinyOS รองรับการทำงานแบบ Event-based programming ด้วยแนวคิดแบบ Component ซึ่งแต่ละ Component จะประกอบด้วยข้อมูลของสถานะทำงานในรูปแบบของ *Frame* โปรแกรมทั่วไปที่ทำงานเรียก *Task* และมี Handler สำหรับ *Events* และ *Commands* ดังภาพประกอบที่ 3.3 เป็นตัวอย่างของ Timer Component ซึ่งเป็นโมดูลที่เข้าใจง่ายเหมาะแก่การศึกษาโครงสร้างของ Component โดยจะประกอบด้วยคำสั่ง (Commands) 3 คำสั่งคือ “init”, “start”, และ “stop” และมี Events คือ “fire” ไปยัง Component อื่น ใน Component นี้ยังเชื่อมโยงกับโมดูลฮาร์ดแวร์ที่เป็น Timer คือคำสั่ง “setRate” ที่สั่งหรือกำหนดค่าให้กับ Component นี้และยังสามารถทำการส่ง Event “fired” ได้อีกด้วย สำหรับงานการประมวลผลจะอยู่ในส่วนของ Task ตัวอย่างของการเขียนโปรแกรมด้วย NesC ใช้งานบนระบบปฏิบัติการ TinyOS นั้นสามารถศึกษาได้จากในเว็ปไซต์ www.tinyos.net



ภาพประกอบที่ 3.3 ตัวอย่างของ Timer Component ใน TinyOS

สรุปท้ายบท

ในบทนี้เป็นการอธิบายโครงสร้างสถาปัตยกรรมของโนดทั้งในส่วนของฮาร์ดแวร์และซอฟต์แวร์เพื่อให้สามารถเป็นความรู้พื้นฐานให้พัฒนาโนดขึ้นใช้งานได้เอง ปัจจัยที่ให้คำนึงมากที่สุดคือเรื่องของการใช้พลังงาน เพราะโนดมีพลังงานที่ใช้งานได้จำกัด (ใช้งานจากแบตเตอร์รี่) สำหรับระบบปฏิบัติการสมองกลฝังตัวในส่วนสุดท้ายนั้นอาจจะจำเป็นก็ต่อเมื่อต้องการที่จะควบคุมการใช้ทรัพยากรฮาร์ดแวร์บนโนดให้มีประสิทธิภาพและการใช้งานโนดที่ต้องการป้องกันไม่ให้ผู้ใช้เข้ามาควบคุมหรือเปลี่ยนแปลงรูปแบบการควบคุมฮาร์ดแวร์ได้โดยตรง สำหรับความคิดเห็นของผู้เขียนแนะนำให้ใช้งานระบบปฏิบัติการสมองกลฝังตัวก็ต่อเมื่อต้องการให้ระบบมีประสิทธิภาพหรือเป็นการสร้างระบบเครือข่ายเซนเซอร์ไร้สายที่สมบูรณ์แบบ ถ้าต้องการเพียงแค่การทดลองรับส่งข้อมูลเพียง 2-3 โนดและใช้งานเซนเซอร์เพียงโนดละ 1-2 ตัว ก็ไม่จำเป็นที่จะต้องใช้ระบบปฏิบัติการสมองกลฝังตัวบนโนด เนื่องจากจะมีความซับซ้อนและใช้เวลาในการพัฒนามากขึ้นเมื่อเปรียบเทียบกับการเขียนโปรแกรมควบคุมไมโครคอนโทรลเลอร์บนโนด

เอกสารอ้างอิง

[1] David Culler and et.al, “MICA: The commercialization of Microsensor Motes,” *Sensor Magazine*, April, 2002.

[2] Sentilla. http://www.sentilla.com.

[3] EYES Project. http://www.eyes.eu.org.

[4] SunSPOT mote specifications. http://www.sunspotworld.com.

[5] iMote Crossbow technology. http://www.xbow.com.

[6] R.F.Monolithics. http://www.rfm.com.

[7] Chipcon. http://www.chipcon.com.

[8] R. F. Ember, Embedded, "Design of an IEEE 802.15.4 Compliant," *EmberNet Ready and ZigBee Ready Communication Module using the EM2420 RF Transceiver*, 2004.

[9] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. E. Culler, and K. S. J. Pister, "System Architecture Directions for Networked Sensors," *Proceedings of the 9th International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 93–104, 2000.

[10] TinyOS. http://www.tinyos.net.

[11] D. Gay, P. Levis, R. von Behren, M. Welsh, E. Brewer, and D. Culler, "The nesC Language: A Holistic Approach to Networked Embedded Systems," *Proceedings of ACM SIGPLAN Conference on Programming Language Design and Implementation*, ACM Press, pp. 1–11, 2003.

บทที่ 4

สถาปัตยกรรมของเครือข่าย

(Network Architecture)

จุดประสงค์การเรียนรู้

ในบทนี้จะได้เรียนรู้การนำโนดมาออกแบบให้เป็นเครือข่ายเซนเซอร์ไร้สาย การเชื่อมต่อบริการของเครือข่ายเซนเซอร์ไร้สาย และแนวทางการทำงานของตัวเชื่อมต่อระหว่างเครือข่าย (Gateway)

ช่วงหลายปีที่ผ่านมามีงานวิจัยที่เกี่ยวข้องกับเครือข่ายเซนเซอร์ไร้สายเกิดขึ้นจำนวนมาก เช่นหัวข้อทางด้าน Self-organizing เครือข่ายที่โนดมีการเคลื่อนที่ (Mobile) หรือเครือข่ายแบบ Ad hoc ในช่วงเวลานั้นเองได้มีการนำเทคโนโลยีเครือข่ายเซนเซอร์ไร้สาย ไปประยุกต์ใช้งานตามความต้องการที่หลากหลายเช่น ต้องการเครือข่ายที่เป็นแบบ Decentralized หรือมีรูปแบบ Distributed หรือการทำงานที่มีการรองรับการประมวลผลแบบเรียวไทม์ เป็นต้น ดังนั้นเพื่อให้สามารถบรรลุเป้าหมายของการนำไปใช้งาน นักวิจัย และพัฒนาควรที่จะต้องเข้าใจรูปแบบการเชื่อมต่อของเครือข่ายเซนเซอร์ไร้สาย หลักการออกแบบ รูปแบบการเชื่อมต่อบริการที่เครือข่ายเซนเซอร์ไร้สายสามารถรองรับได้ และสุดท้ายเป็นการอธิบายการเชื่อมต่อจากระบบเครือข่ายเซนเซอร์ไร้สายออกสู่เครือข่ายอื่นๆ ภายนอก

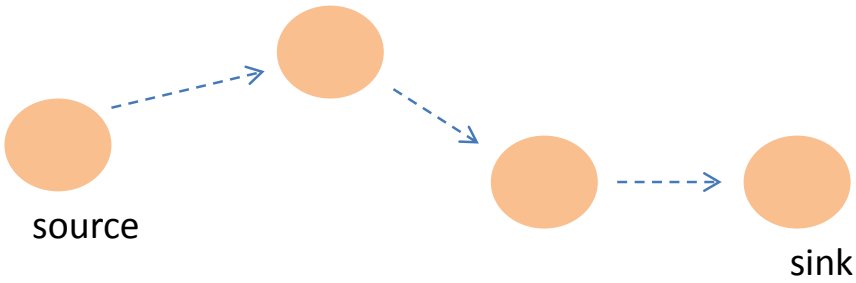
4.1 รูปแบบเครือข่ายเซนเซอร์ไร้สาย

เครือข่ายเซนเซอร์ไร้สายมีโนดเป็นองค์ประกอบหลักที่สำคัญ และมีการนำเทคโนโลยีเครือข่ายเซนเซอร์ไร้สายไปประยุกต์ใช้งานที่หลากหลายเช่น การตรวจจับเหตุการณ์ การเฝ้าวัดข้อมูลทางกายภาพ และการติดตามวัตถุ เป็นต้น แม้ว่าจะมีหลากหลายรูปแบบการประยุกต์ใช้งาน โนดที่ใช้ในเครือข่ายเซนเซอร์ไร้สายนั้นมี 3 หน้าที่ คือ โนดที่ทำหน้าที่เป็นต้นทาง หรือเรียกว่า Source หมายถึงโนดที่ทำหน้าที่ตรวจวัดรับข้อมูลจากเซนเซอร์

โดยตรง กับโนดที่ทำหน้าที่เป็นปลายทาง หรือเรียกว่า Sink และโนดที่ทำหน้าที่ส่งต่อข้อมูลหรือเรียกว่า Repeater โนดปลายทางสามารถมีได้หลากหลายรูปแบบอาจจะเป็นโนดที่ถูกออกแบบและพัฒนาขึ้นมาโดยเฉพาะ และมีหน่วยความจำมากกว่าโนดทั่วไปเพื่อทำการเก็บรวบรวมข้อมูล หรือโนดอาจจะมีการเชื่อมต่อกับอุปกรณ์ชนิดอื่นเพื่อให้เชื่อมโยงไปยังเครือข่ายรูปแบบอื่นเช่น โนดปลายทางเชื่อมต่อกับคอมพิวเตอร์ เป็นต้น ดังนั้นอาจจะเป็นไปได้ว่าโนดปลายทางนี้จะทำหน้าที่เป็น Gateway ไปด้วยในตัวเดียวกัน

4.1.1 Single hop และ Multi hop

การส่งข้อมูลระหว่างโนดในเครือข่ายเซนเซอร์ไร้สายจะคำนึงถึงการใช้พลังงานให้มีประสิทธิภาพ ดังนั้นรูปแบบการส่งข้อมูลระหว่างโนดถึงโนดที่เรียกว่า Single hop จึงถูกออกแบบใหม่เพื่อให้ระยะทางของการส่งลดน้อยลง เนื่องจากระยะทางของการส่งข้อมูลจะแปรผันตรงกับกำลังไฟของสัญญาณในการส่งข้อมูล นอกจากเรื่องพลังงานแล้วการส่งต่อข้อมูลเป็นทอดจะช่วยให้สามารถส่งข้อมูลหลบหลีกสิ่งกีดขวางไปยังโนดปลายทางได้ดังแสดงในภาพประกอบที่ 4.1 แต่ทั้งนี้การส่งข้อมูลเป็นทอดเช่นนี้อาจจะส่งผลกระทบในเรื่องของเวลาโดยรวมในการส่งข้อมูลจากต้นทางไปยังโนดปลายทาง

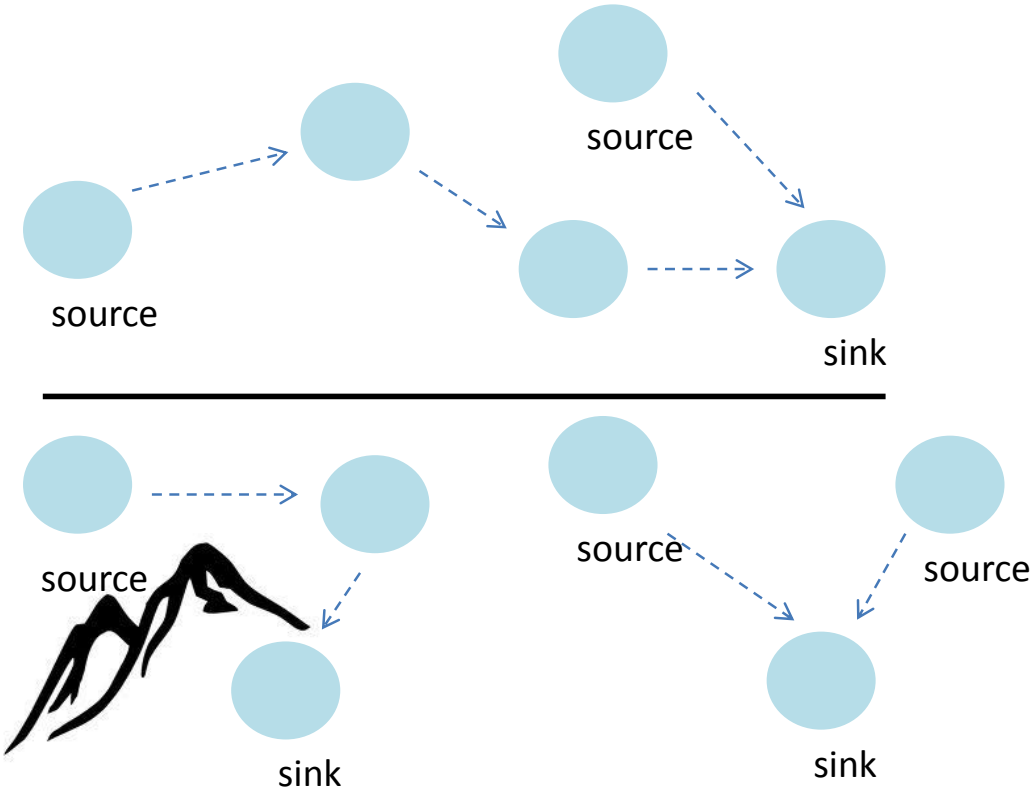


ภาพประกอบที่ 4.1 การส่งข้อมูลแบบส่งต่อเป็นทอด Multi hop

4.1.2 โนดเคลื่อนที่

ในการนำไปใช้งานจริงเครือข่ายเซนเซอร์ไร้สายไม่ได้มีโนดต้นทางและโนดปลายทางอย่างละ 1 โนด หากแต่ระบบสามารถในโนดต้นทางและโนดปลายทางได้มากกว่า 1 โนด ดังภาพประกอบที่ 4.2 ดังนั้นเมื่อโนดต้นทางที่มากกว่า 1 โนดทำการส่งข้อมูลต่างก็มุ่งที่จะส่งไปยังโนดปลายทางที่แตกต่างกันบ้างหรือต่างก็ส่งข้อมูลไปยังโนดปลายทางโนดเดียวกันบ้าง จึงเป็นเหตุให้เกิดหลากหลายเส้นทางของการส่งข้อมูลภายในเครือข่าย ปัญหาเรื่อง

ของการชนกันของข้อมูลจึงสามารถเกิดขึ้นได้ ทำให้เป็นประเด็นของหัวข้อวิจัยที่สำคัญหัวข้อหนึ่งของการพัฒนาเครือข่ายเซนเซอร์ไร้สาย



ภาพประกอบที่ 4.2 รูปแบบการส่งข้อมูลแบบ Multiple sources และ Sinks

นอกจากการส่งข้อมูลที่มีโนดต้นทางและโนดปลายทางมากกว่า 1 โนดแล้วก็ยังมีโอกาสที่โนดจะเกิดการเคลื่อนที่ ซึ่งจะส่งผลต่อการส่งข้อมูลเช่นกัน โดยที่สามารถแยกรูปแบบของการเคลื่อนที่ของโนดได้ดังนี้

Node mobility เป็นรูปแบบที่โดยธรรมชาติแล้วโนดจะเคลื่อนที่อยู่ตลอดเวลาภายในเครือข่าย ซึ่งเครือข่ายเซนเซอร์ไร้สายที่มีโนดเคลื่อนที่อยู่ตลอดเวลาเช่นนี้จะมักถูกใช้งานกับระบบการขนส่ง โนดจะถูกติดกับยานพาหนะ หรือระบบในปศุสัตว์โดยที่โนดจะติดอยู่กับสัตว์เลี้ยง เป็นต้น ทำให้จะต้องมีการพัฒนาเครือข่ายเซนเซอร์ไร้สายให้รองรับสถานการณ์ต่างๆที่อาจจะเกิดขึ้นอันเนื่องจากที่โนดเคลื่อนที่อยู่ตลอดเวลา เช่น เมื่อโนดเคลื่อนตัวออกจากรัศมีของการรับส่งข้อมูล ทำให้ไม่สามารถมีเส้นทางส่งข้อมูลกลับไปยังโนดปลายทางได้ หรือประเด็นที่จะต้องพิจารณาถึงความเร็วในการเคลื่อนที่ของโนดเพื่อยังคงให้สามารถรับส่งข้อมูลได้ตามปกติ เป็นต้น

Sink mobility เป็นรูปแบบที่โนดปลายทางมีการเคลื่อนที่หรือเคลื่อนที่ ซึ่งอาจจะเป็นกรณีที่โนดปลายทางมีการติดตั้งอยู่กับอุปกรณ์พกพาได้เช่น สมาร์ทโฟน หรือ ติดกับยานพาหนะสำรวจ เป็นต้น ทั้งนี้ขึ้นกับรูปแบบของงานประยุกต์ ในกรณีนี้รูปแบบของการรับส่งข้อมูลอาจจะเกิดปัญหาได้ถ้าหากโนดปลายทางเกิดการเคลื่อนที่ในระหว่างที่โนดกำลังพยายามส่งข้อมูลหรือยังส่งข้อมูลไม่เสร็จ ดังนั้นโพรโทคอลของการติดต่อกับโนดปลายทางจะต้องมีความแตกต่างจากรูปแบบทั่วไป หรืออาจจะต้องมีการเข้าจังหวะ (Synchronization) ระหว่างโนดปลายทางกับโนดทั่วไปได้แก่ เมื่อส่งข้อมูลเรียบร้อยแล้วจึงจะสามารถทำการเคลื่อนที่ได้ เป็นต้น แต่รูปแบบหรือวิธีการแก้ไขปัญหาจะต้องคำนึงถึงงานประยุกต์ที่นำเอาเครือข่ายเซนเซอร์ไร้สายไปใช้งาน

Event mobility เป็นกรณีที่โนดทั่วไปบางตัวมีการเคลื่อนที่ในเฉพาะบางเวลาหรือบางโอกาส ตัวอย่างเช่นการประยุกต์ใช้ในการติดตามวัตถุ (Object tracking) ในพื้นที่ที่ครอบคลุมด้วยเครือข่ายเซนเซอร์ไร้สาย

4.2 หลักการออกแบบเครือข่ายเซนเซอร์ไร้สาย

การออกแบบเครือข่ายเซนเซอร์ไร้สายจะต้องมีองค์ความรู้หลากหลายด้าน (Multidisciplinary) ที่ประกอบด้วยศาสตร์ทางด้าน การสื่อสารไร้สาย ระบบเครือข่าย ระบบสมองกลฝังตัว การประมวลผลสัญญาณ และการพัฒนาซอฟต์แวร์ สำหรับการออกแบบเครือข่ายเซนเซอร์ไร้สายในหัวข้อนี้จะกล่าวถึงประเด็นที่สำคัญที่จะต้องคำนึงถึงในการออกแบบได้แก่ ข้อจำกัดทางฮาร์ดแวร์ของโนด ราคาของโนด สื่อกลางในการส่งข้อมูล การใช้พลังงาน และหลักการออกแบบเบื้องต้น

4.2.1 ข้อจำกัดของฮาร์ดแวร์หรือโนด

โนดเป็นอุปกรณ์ที่สำคัญในเครือข่ายเซนเซอร์ไร้สายซึ่งประกอบด้วย หน่วยประมวลผล เซนเซอร์ ภาครับส่งคลื่นวิทยุ หน่วยความจำ และแหล่งจ่ายพลังงาน โดยที่ทั้งหมดนี้จะต้องอยู่ในภายเงื่อนไขและข้อจำกัดของระบบสมองกลฝังตัวที่จะต้องมีขนาดเล็กกะทัดรัด ซึ่งโนดที่มีขนาดเล็กก็จะสามารถตอบสนองความต้องการในการนำไปใช้งานในหลากหลายงานประยุกต์ตัวอย่างเช่น ระบบเฝ้าระวังตรวจวัดสุขภาพเพราะอุปกรณ์ต้องติดอยู่กับคนจำเป็นที่จะต้องมีขนาดเล็กพกพาง่ายและมีน้ำหนักเบา แนวคิดของระบบสมองกลฝังตัวที่สำคัญอีกประการคือจะต้องประหยัดพลังงานซึ่งก็สอดคล้องกับแนวคิดของเครือข่ายเซนเซอร์ไร้สาย เพราะโนดจะใช้พลังงานจากแหล่งจ่ายพลังงานอย่างเช่นแบตเตอร์รี่เท่านั้น หรือเมื่อนำโนดไปกระจายใช้งานในพื้นที่เป็นจำนวนมากก็จะไม่

สามารถเข้าไปเปลี่ยนแบตเตอร์รี่ให้กับโนด และประการสุดท้ายของแนวคิดระบบสมองกลฝังตัวคือโนดควรต้องมีราคาถูก เพื่อให้สามารถใช้โนดได้จำนวนมาก

สำหรับเครือข่ายเซนเซอร์ไร้สายเรื่องของการใช้พลังงานมีความสำคัญมากที่สุดและเป็นตัวกำหนดการออกแบบเครือข่ายที่จะต้องให้สอดคล้องกับฮาร์ดแวร์ที่จะสามารถใช้งานได้ในข้อจำกัดเช่น โมดูลภาครับส่งคลื่นวิทยุจะส่งได้ในระยะทางที่สั้นกว่าโมดูลภาครับส่งคลื่นวิทยุในเครือข่ายไร้สายอื่นๆ ประกอบกับราคาที่จะต้องไม่แพงมากทำให้ตัวเลือกของไอซีที่มาทำหน้าที่เป็นโมดูลภาครับส่งคลื่นวิทยุจะมีคุณสมบัติที่จะส่งข้อมูลได้ไม่ไกลมาก และขนาดของข้อมูลที่จะส่งได้ภายในเวลาหน่วยวินาทีจะน้อยกว่าโมดูลภาครับส่งคลื่นวิทยุในเครือข่ายไร้สายอื่นๆเช่นกัน

หน่วยความจำในโนดจะนิยมเลือกใช้แบบที่มาพร้อมกับไมโครคอนโทรลเลอร์คือหน่วยความจำแบบ RAM และ Flash ตัวอย่างเช่น ในโนด Smart Dust ใช้ไมโครคอนโทรลเลอร์ Atmel AVR 8535 มีหน่วยความจำแบบ Flash เพื่อเก็บคำสั่งขนาด 8 kB และมีหน่วยความแบบ RAM ขนาด 512 Bytes และหน่วยความจำแบบ EEPROM ขนาด 512 Bytes

4.2.2 ราคาของโนด

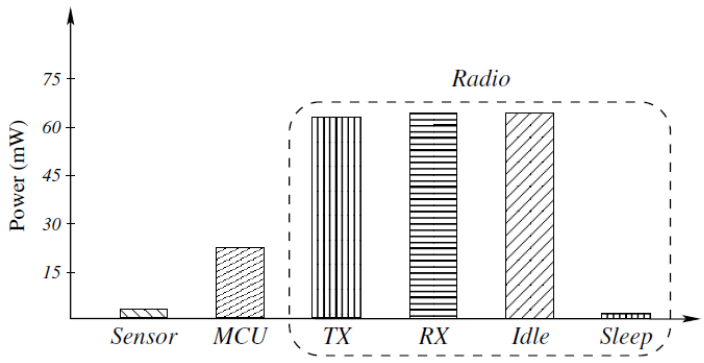
เนื่องจากเครือข่ายเซนเซอร์ไร้สายจะใช้โนดเป็นจำนวนมาก ดังนั้นราคาโนดต่อตัวจะส่งผลต่อค่าใช้จ่ายทั้งหมดของระบบเครือข่ายเซนเซอร์ไร้สาย การเลือกใช้ไมโครคอนโทรลเลอร์หรือโมดูลภาครับส่งคลื่นวิทยุเพื่อสร้างโนดก็จะต้องมีราคาไม่สูง นอกเหนือจากการเลือกใช้อุปกรณ์ที่ราคาไม่แพงแล้วก็อาจจะสามารถทำได้โดยการสร้างหรือผลิตโนดในจำนวนมากก็จะช่วยทำให้ราคาการผลิตต่อชิ้นลดต่ำลงได้ พบว่าในปัจจุบันเซนเซอร์มีราคาสูงกว่าไมโครคอนโทรลเลอร์และโมดูลอื่นๆมาก

4.2.3 สื่อกลางการส่งข้อมูล

สื่อกลางสำหรับรับส่งข้อมูลคือคลื่นความถี่วิทยุในย่าน ISM แม้ว่าจะมีหลากหลายคลื่นความถี่ให้เลือกใช้งาน แต่ข้อจำกัดของเครือข่ายเซนเซอร์ไร้สายในเรื่องของพลังงานและราคาดังนั้นจึงไม่เลือกใช้คลื่นความถี่ย่าน Ultra High Frequency (UHF) แม้ว่าจะสามารถส่งข้อมูลได้ไกลและส่งข้อมูลได้มาก ในประเทศไทยควรเลือกใช้คลื่นความถี่วิทยุที่ 2.4 GHz ซึ่งสามารถใช้งานได้และรองรับมาตรฐาน IEEE 802.15.4

4.2.4 พลังงาน

พลังงานถือเป็นเรื่องที่สำคัญมากที่สุดในเครือข่ายเซนเซอร์ไร้สาย ดังนั้นจึงมีการวิเคราะห์การใช้กำลังงานไฟฟ้าของโนดรูปแบบ MicaZ [1] ที่ได้ทำการแยกแยะการใช้กำลังงานไฟฟ้าของแต่ละส่วนบนโนดแสดงได้ดังภาพประกอบที่ 4.3 พบว่าโมดูลภาครับส่งคลื่นวิทยุเป็นส่วนที่ใช้กำลังงานมากที่สุด ในขณะที่ไมโครคอนโทรลเลอร์และเซนเซอร์ใช้กำลังงานรองลงมาตามลำดับ แต่ทั้งนี้ในส่วนของเซนเซอร์ที่ในภาพเป็นเซนเซอร์วัดอุณหภูมิและความชื้นในอากาศที่พัฒนาด้วยเทคโนโลยี Complementary Metal Oxide Semiconductor (CMOS) ซึ่งถูกออกแบบมาเพื่อให้ใช้กำลังงานไฟฟ้าต่ำ สำหรับการใช้กำลังงานของเซนเซอร์นั้นจะขึ้นอยู่กับชนิดและเทคโนโลยีของเซนเซอร์ที่เลือกมาใช้งาน



ภาพประกอบที่ 4.3 แผนภาพการใช้กำลังงาน (Power) ในแต่ละโมดูลบน MicaZ [1]

จุดที่น่าสนใจคือไมโครคอนโทรลเลอร์นั้นมีการใช้กำลังงานไฟฟ้าที่ต่ำกว่าโมดูลภาครับส่งคลื่นวิทยุอยู่มากหรือน้อยกว่าไม่ต่ำกว่า 3 เท่า ดังนั้นทำให้การประมวลผลข้อมูลดิบจากเซนเซอร์หรือการคัดกรองข้อมูลบนโนดจึงนิยมทำกันมาก เพื่อลดปริมาณของข้อมูลที่จะต้องทำการรับส่งกันภายในเครือข่าย

สำหรับส่วนของโมดูลภาครับส่งคลื่นวิทยุก็จะพบเห็นได้ชัดเจนว่า การทำให้โมดูลภาครับส่งคลื่นวิทยุเข้าสู่สภาวะหลับ (Sleep) จะช่วยให้ประหยัดกำลังงานไฟฟ้าได้มากอย่างเห็นได้ชัด และการใช้กำลังงานไฟฟ้าของการรับและการส่งข้อมูลจะมีปริมาณที่ใกล้เคียงกัน เนื่องจากวงจรการรับส่งข้อมูลเป็นแบบการส่งระยะใกล้ (Short range) ที่กำลังการส่งประมาณ 0 dBm

4.2.5 หลักการเบื้องต้นในการออกแบบเครือข่ายเซนเซอร์ไร้สาย

โครงสร้างเครือข่ายเซนเซอร์ไร้สายจะใช้หลักการของ Distributed เพื่อให้สามารถขยายขนาดของเครือข่ายได้ง่าย ไม่ควรออกแบบในลักษณะที่เป็นแบบ Centralization โนดแต่ละตัวในเครือข่ายควรถูกใช้งานเป็นส่วนหนึ่งของเครือข่าย รวมทั้งใช้อัลกอริทึมและโพรโทคอลแบบ Distributed หรือมักจะเรียกคุณสมบัติเช่นนี้ในเครือข่ายเซนเซอร์ไร้สายว่า Self-organization โนดแต่ละตัวไม่เพียงทำหน้าที่ในการรับส่งข้อมูลเท่านั้นแต่จะต้องทำหน้าที่ร่วมตัดสินใจในการทำงานของเครือข่ายโดยนำข้อมูลที่เกี่ยวข้องกับเครือข่ายมาประมวลผลและตัดสินใจ เช่นการทำ Aggregation แนวคิดของการทำ Aggregation ก็เพื่อให้ปริมาณของข้อความที่จะถูกส่งภายในเครือข่ายไปยัง Sink โนดมีจำนวนลดลง หรือหมายถึงการพยายามบีบอัดข้อมูลให้มีจำนวนบิตที่จะต้องส่งภายในเครือข่ายลดลงแต่ยังคงได้ข้อมูลที่จำเป็นเหมือนเดิม หลักการบีบอัดข้อมูลแบบ Distributed สามารถศึกษาได้จากบทความวิจัย [2] นอกเหนือจากการทำ Distributed compression แล้วการทำ Distributed signal processing มีความจำเป็นสำหรับการทำการประมวลผลติดตามวัตถุ (Target tracking) สามารถศึกษาข้อมูลงานวิจัยที่เกี่ยวข้องได้จากบทความวิจัยของ Zhao และ Guibas [3]

ข้อมูลที่ได้จากเซนเซอร์ถือเป็นหัวใจสำคัญในการใช้งานเครือข่ายเซนเซอร์ไร้สาย หรืออาจจะเรียกว่าเป็น Data Centric ไม่ได้สนใจกับตัวตนของโนด แต่ในบางสถานการณ์เครือข่ายเซนเซอร์ไร้สายจำเป็นที่จะต้องให้โนดขอสมัครเข้าเป็นสมาชิกของเครือข่ายเซนเซอร์ไร้สายก่อนที่จะเริ่มทำการรับส่งข้อมูล กระบวนการตรงนี้เราเรียกว่า Publish/Subscribe [4] ซึ่งได้รับความนิยมอย่างมากในเครือข่ายเซนเซอร์ไร้สาย

4.3 ปัจจัยชี้วัดของเครือข่ายเซนเซอร์ไร้สาย

การออกแบบโนดและเครือข่ายเซนเซอร์ไร้สายสามารถทำได้หลากหลายรูปแบบ แต่ตัวชี้วัดความสำเร็จของการใช้งานหรือตัวบ่งบอกว่าเครือข่ายเซนเซอร์ไร้สายนี้ดีหรือไม่จะพิจารณาจากสิ่งต่างๆดังต่อไปนี้

คุณภาพของบริการ (Quality of service, QoS)

ความสามารถในการให้บริการของเครือข่ายเซนเซอร์ไร้สายนิยมวัดด้วยคุณภาพของบริการในแต่ละระดับชั้น ตัวอย่างเช่นในระดับชั้นกายภาพหรือชั้นล่างจะวัดจากค่าหน่วงเวลา (Delay) อัตราการสูญเสียข้อมูล (Packet loss rate) และ Bandwidth หรือการบอกคุณภาพของบริการในระดับชั้นสูงขึ้นไปอาจจะดูจากสิ่งที่ได้รับมาจากเครือข่ายเช่น คุณภาพของเสียง หรือคุณภาพของภาพที่ได้ ในที่นี้จะขอยกตัวอย่างการรายงานคุณภาพของบริการในเครือข่ายเซนเซอร์ไร้สายในระดับชั้น Application ได้แก่

- Event detection สำหรับแจ้งรายงานเหตุการณ์ต่างๆที่เกิดขึ้น เช่นในระบบเฝ้าระวังการล้มของผู้สูงอายุนั้นเพื่อเป็นการตรวจสอบความผิดพลาดของการส่งข้อมูล ดังนั้นจึงอาจจะจำเป็นที่จะต้องรายงานพฤติกรรมของผู้สูงอายุที่ตรวจวัดได้ทุกๆช่วงเวลาที่กำหนด แต่รูปแบบนี้อาจจะทำให้เกิด Overhead ในเครือข่าย รวมทั้งปริมาณของข้อมูลที่มาเก็บยังตัว Sink โนดได้

- Event detection delay มีความจำเป็นที่อาจจะต้องรายงานค่าหน่วงเวลาสำหรับการตรวจสอบเหตุการณ์ที่รายงานมายัง Sink โนด เมื่อมีความผิดพลาดในการรับส่งข้อมูลก็ควรจะต้องมีการเก็บบันทึกและรายงานผลให้กับผู้ดูแลระบบด้วย

การใช้พลังงานอย่างมีประสิทธิภาพ (Energy Efficiency)

คนส่วนมากมักจะเข้าใจผิดถึงวิธีการใช้พลังงานกับระบบสมองกลฝังตัวที่มีข้อจำกัดในเรื่องของพลังงานว่าจะออกแบบให้มีการใช้พลังงานต่ำที่สุด แต่ความจริงแล้วตัวชี้วัดที่ควรจะต้องพิจารณาคือการใช้พลังงานอย่างมีประสิทธิภาพ เพราะการใช้พลังงานต่ำแต่จะต้องใช้เวลาในการทำงานมากขึ้นจากเดิมจนกว่าจะเสร็จงานก็จะส่งผลให้ปริมาณการใช้พลังงานโดยรวมมากกว่ารูปแบบการทำงานที่ใช้พลังงานสูงกว่าแต่เสร็จได้รวดเร็วกว่า (สอดคล้องกับสมการ E = P*t) การใช้พลังงานอย่างมีประสิทธิภาพในเครือข่ายเซนเซอร์นั้นอาจจะวัดได้จาก พลังงานที่ใช้ในการส่งข้อมูล 1 บิตได้อย่างถูกต้อง (Energy per correctly received bit) การทำ trade-off ระหว่างค่าหน่วงเวลาและพลังงาน เช่นบางงานที่จำเป็นเร่งด่วนสามารถให้ใช้พลังงานได้มากกว่าปกติ แต่จะต้องทำให้เสร็จอย่างรวดเร็ว และสุดท้ายตัวชี้วัดที่สำคัญคืออายุการทำงานของเครือข่าย (Network lifetime) โดยส่วนใหญ่นิยมตรวจสอบจาก 1) ระยะเวลาการทำงานจนกว่าจะเริ่มมีโนดตัวแรกหมดพลังงานหรือหยุดการทำงาน 2) ระยะเวลาการทำงานจนกว่าโนดในเครือข่ายหยุดการทำงานหรือหมดพลังงานไป 50% เราเรียกว่า Network half-life 3) ระยะเวลาทำงานจนกว่าโนดในเครือข่ายจะหยุดทำงานแล้วทำให้เกิดการตัดขาดของเครือข่ายออกเป็น 2 เครือข่าย เราเรียกว่า Time to partition และ 4) ระยะเวลาทำงานจนกว่าโนดในเครือข่ายจะไม่สามารถรับข้อมูลจากเซนเซอร์จุดใดจุดหนึ่งได้หรือไม่สามารถมีเครือข่ายครอบคลุมพื้นที่ที่ต้องการได้เราเรียกว่า Time to loss of coverage ตัวชี้วัดที่กล่าวมาเหล่านี้สามารถจะช่วยอธิบายการใช้พลังงานอย่างมีประสิทธิภาพในเครือข่ายเซนเซอร์ไร้สายได้ทั้งนี้ขึ้นอยู่กับสิ่งที่ผู้พัฒนาสนใจว่าจะใช้ตัวชี้วัดตัวใด

4.4 การให้บริการการเชื่อมต่อของเครือข่ายเซนเซอร์ไร้สาย

เพื่อให้เกิดความสะดวกในการใช้งานโนดและการพัฒนาเครือข่ายเซนเซอร์ไร้สาย จึงมีการจัดทำโครงสร้างหรือรูปแบบของการเชื่อมต่อกับชิ้นส่วน (Component) ต่างๆของระบบปฏิบัติการหรือโพรโทคอล ทำให้การพัฒนาโปรแกรมบนโนดสามารถทำได้สะดวกและรวดเร็ว นักพัฒนาเพียงเรียกใช้และควบคุมโพรโทคอลหรือชิ้นส่วนของโปรแกรมที่มีอยู่แล้วให้ทำงานตามที่ต้องการ แต่เพื่อให้เป็นมาตรฐานและใช้งานได้อย่างมีประสิทธิภาพ จึงมีแนวคิดในการพัฒนาการให้บริการการเชื่อมต่อ (Service Interface) เหมือนอย่างการพัฒนาโปรแกรมบนเครือข่ายทั่วไปอย่างเช่นอินเตอร์เน็ต ที่มีการให้บริการการเชื่อมต่อผ่านทางช่องทางมาตรฐาน (Socket) เป็นต้น ฟังก์ชันที่สำคัญการให้บริการการเชื่อมต่อในเครือข่ายเซนเซอร์ไร้สายประกอบด้วย

- ฟังก์ชันพื้นฐานสำหรับการร้องขอและการตอบสนองการวัดค่าจากเซนเซอร์ การตั้งค่าพื้นฐานของเซนเซอร์เช่น ความถี่ของการอ่านค่าจากเซนเซอร์ เป็นต้น

- ฟังก์ชันสำหรับการแจ้งเหตุการณ์ที่เข้ามาโดยไม่ได้คาดการณ์ไว้ (Asynchronous event) ไว้สำหรับใช้ในกรณีที่เงื่อนไขถูกต้องตามที่ตั้งไว้การร้องขอโนดก็จะสามารถทำงานได้ ฟังก์ชันนี้จะเน้นการจัดการกับเหตุการณ์ที่ไม่สามารถคาดเดาได้ว่าจะเกิดขึ้นเมื่อไหร่

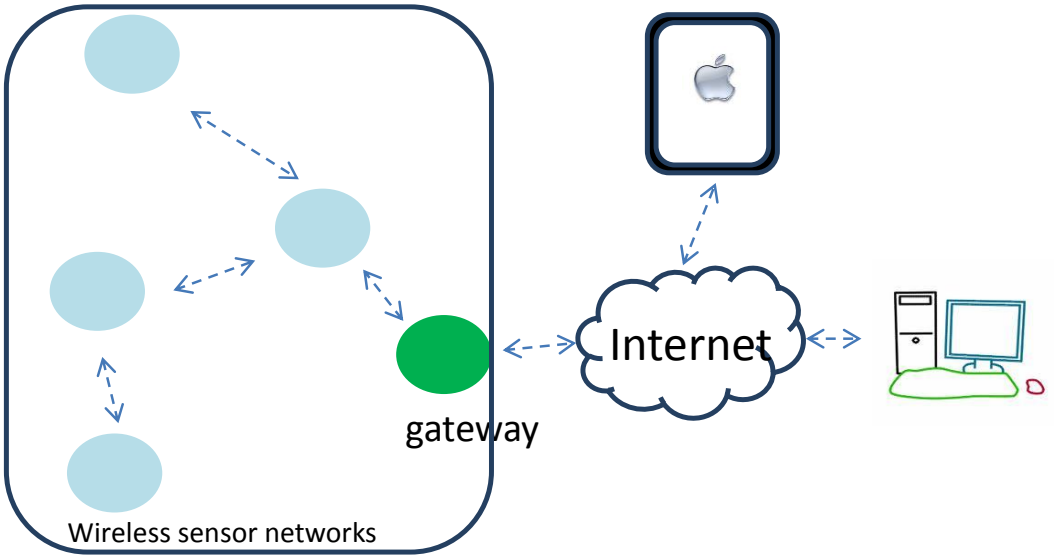
- การทำงานของทั้งสองฟังก์ชันนั้นจำเป็นที่จะต้องมีการกำหนดตัวเลขของโนด (Address) ดังนั้นฟังก์ชันของการกำหนดค่าเลขประจำโนดจึงเป็นที่ต้องการ นอกจากนี้อาจจะมีฟังก์ชันของการทำ publish/subscribe สำหรับโนดที่เป็นสมาชิกในกลุ่มโดยการจะแบ่งกลุ่มของโนดตามสถานที่ติดตั้ง เช่นกลุ่มของโนดที่อยู่ทางด้านตะวันออกของพื้นที่ หรืออาจจะแบ่งกลุ่มของโนดตามรูปแบบของการเฝ้าระวังค่าจากเซนเซอร์ เช่น กลุ่มของโนดที่จะทำการอ่านค่าจากเซนเซอร์ก็ต่อเมื่อมีอุณหภูมิสูงกว่า 20 องศาเซลเซียส เป็นต้น

- ฟังก์ชันที่ไม่เกี่ยวข้องกับเวลา ใช้สำหรับสั่งการตรงไปยังโนดเพื่อทำงานที่สำคัญตัวอย่างเช่น สั่งให้โนดตื่น หรือการสั่งให้โนด restart ใหม่เป็นต้น

- ฟังก์ชันที่ต้องการข้อมูลหรือสถานะของโนดเช่น ต้องการระดับพลังงานที่คงเหลือของโนด ต้องการตำแหน่งของโนด หรือส่งคำสั่งเพื่อตรวจสอบว่าโนดยังสามารถตอบสนองกลับมาได้หรือไม่ เป็นต้น

4.5 ตัวเชื่อมต่อระหว่างเครือข่าย (Gateway)

คุณสมบัติหลักของตัวเชื่อมต่อระหว่างเครือข่ายหรือ Gateway คือทำหน้าที่เป็นตัวกลางรับข้อมูลจากเซนเซอร์แล้วเชื่อมต่อการสื่อสารออกสู่เครือข่ายอื่นเช่น อินเตอร์เน็ต หรือ เครือข่ายโทรศัพท์มือถือ เป็นต้น เพื่อให้ข้อมูลสามารถส่งต่อไปยังผู้ใช้งาน (User) ดังภาพประกอบที่ 4.4 โดยอุปกรณ์ที่จะมาทำหน้าที่ของตัวเชื่อมต่อระหว่างเครือข่ายนี้จะต้องมีโมดูลภาครับส่งคลื่นวิทยุตามมาตรฐานของเครือข่ายเซนเซอร์ไร้สายและมีโมดูลเชื่อมต่ออินเตอร์เน็ต ตัวอย่างเช่น เราอาจจะเลือกใช้บอร์ดสมองกลฝังตัวที่มีพอร์ตอนุกรมที่สามารถเชื่อมต่อกับโมดูลรับส่งคลื่นวิทยุ 2.4 GHz ชิปของ XBee หรือ ChipCon แล้วบอร์ดสมองกลฝังตัวจะต้องมีโมดูล WiFi หรือ Ethernet เป็นต้น



ภาพประกอบที่ 4.4 รูปแบบการเชื่อมต่อระหว่างเครือข่าย

ในทางตรงกันข้ามเมื่อผู้ใช้หรือทางฝั่ง Client ต้องการติดต่อถึงโนดภายในเครือข่ายเซนเซอร์ไร้สายก็จะต้องทำผ่านทางตัวเชื่อมต่อระหว่างเครือข่ายนี้ ซึ่งรูปแบบการเข้าถึงโนดก็มีหลากหลายวิธีการตั้งแต่อย่างง่ายคือการกำหนดหมายเลขของโนดภายในเครือข่ายไว้โดยปกติจะสามารถกำหนดได้ตั้งแต่หมายเลข 0 – 255 หรือถ้าซับซ้อนมากยิ่งขึ้นก็สามารถใช้เลข IP อย่างเช่นมาตรฐาน 6LoWPAN ก็ได้เช่นกัน ทั้งนี้ขึ้นกับความต้องการในการใช้งาน ตรงที่ตัวเชื่อมต่อระหว่างเครือข่ายก็ควรจะต้องมีบริการที่เรียกว่า Service Discovery (SD) ไว้เพื่อให้ผู้ใช้ภายนอกเครือข่ายสามารถทราบได้ว่ามีบริการหรือมีข้อมูลที่ได้จากเซนเซอร์อะไรอยู่บ้าง ตัวอย่างงานวิจัยที่มีได้แก่

[5-9] นอกจากนี้แล้วตัวเชื่อมต่อระหว่างเครือข่ายอาจจะสามารถพิจารณาเรื่องของความปลอดภัย (Security) ร่วมด้วยเพื่อเป็นการป้องกันบุคคลภายนอกหรือป้องกันไม่ให้ข้อมูลถูกส่งต่อไปยังผู้ที่ไม่สมควรจะได้รับ ในปัจจุบันตัวเชื่อมต่อระหว่างเครือข่ายยังมีบทบาทสำคัญเมื่อมีนิยามของการพยายามให้อุปกรณ์ต่างๆสามารถเชื่อมต่อเครือข่ายอินเตอร์เน็ตได้ หรือที่เรียกว่า Internet of Thing (IoT) ดังนั้นตัวเชื่อมต่อระหว่างเครือข่ายหรือ Gateway ของเครือข่ายเซนเซอร์ไร้สายจึงมีความสำคัญอย่างมาก จะต้องพิจารณาการเชื่อมต่อทั้ง 2 ด้านให้เหมาะสม 1) ด้านที่เชื่อมต่อกับเครือข่ายเซนเซอร์ไร้สายที่อาจจะเป็น ZigBee หรือ 6LowPAN หรือ IEEE802.15.4 มาตรฐานปกติ ในขณะที่ 2) ด้านที่จะออกสู่อินเตอร์เน็ตจะต้องพิจารณาว่าต้องการเชื่อมต่อไปยังผู้ใช้หรือเครือข่ายอื่นๆผ่านทางช่องทางใดที่จะสะดวกที่สุดเช่น 3G, 4G หรือ Wifi เป็นต้น

สรุปท้ายบท

ในบทนี้เป็นการสรุปรูปแบบการเชื่อมต่อของเครือข่ายเซนเซอร์ไร้สาย และให้แนวคิดหลักการของการออกแบบเครือข่ายเซนเซอร์ไร้สาย รูปแบบการเชื่อมต่อบริการที่เครือข่ายเซนเซอร์ไร้สายสามารถรองรับได้ เพื่อให้สามารถนำศักยภาพของเครือข่ายเซนเซอร์ไร้สายออกมาใช้ประโยชน์ได้สูงสุด และตอนท้ายของบทเป็นการอธิบายตัวเชื่อมต่อระหว่างเครือข่าย เพื่อให้ข้อมูลจากระบบเครือข่ายเซนเซอร์ไร้สายสามารถออกสู่เครือข่ายอื่นๆภายนอกหรือในทางตรงกันข้ามผู้ใช้จากภายนอกสามารถเข้าถึงข้อมูลหรือควบคุมอุปกรณ์ภายในเครือข่ายเซนเซอร์ไร้สายได้

เอกสารอ้างอิง

[1] Crossbow MicaZ mote specifications. http://www.xbow.com.

[2] S. S. Pradhan, J. Kusuma and K. Ramchandran, “Distributed Compression in a Dense Microsensor Network,” *IEEE Signal Processing Magazine*, 19(2): 51–60, 2002.

[3] F. Zhao and L. Guibas, “*Wireless Sensor Networks – An Information Processing Approach*,” Elsevier/Morgan-Kaufman, Amsterdam, NY, 2004.

[4] P. Th. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, “The Many Faces of Publish/Subscribe”, *ACM Computing Surveys (CSUR)*, 35(2): 114–131, 2003.

[5] L. Steenkamp and et.al, “Wireless sensor network gateway,” *IEEE International Conference on AFRICON*, pp.1-6, 2009.

[6] Ahmet Burask Gokbayrak and Oguzhan Urhan, “Wireless sensor network gateway design for home automation applications,” *Signal processing and Communication Applications Conference (SIU),* pp.1770-1773, 2014.

[7] B. da Silva Campos, E. Freire Nakamura, C.M.S. Figueiredo and J.J.P.C. Rodrigues, “On the design of UPnP gateways for service discovery in wireless sensor networks,” *IEEE Symposium on Computers and Communications (ISCC),* pp.719-722, 2011.

[8] F.M.Anwar, Seung-wha Yoo and Ki-Hyung Kim, “Survey on service discovery for wireless sensor networks,” *International Conference on Ubiquitous and Future Networks*, pp.17-21, 2010.

[9] D. Singh and Daeyeoul Kim, “Performance analysis of gateway discovery techniques: IPv6-based wireless sensor networks,” *International Conference on Evolving Internet (INTERNET)*, pp.142-146, 2010.

บทที่ 5

โพรโทคอลในชั้นเครือข่าย

(Protocols in Network Layer)

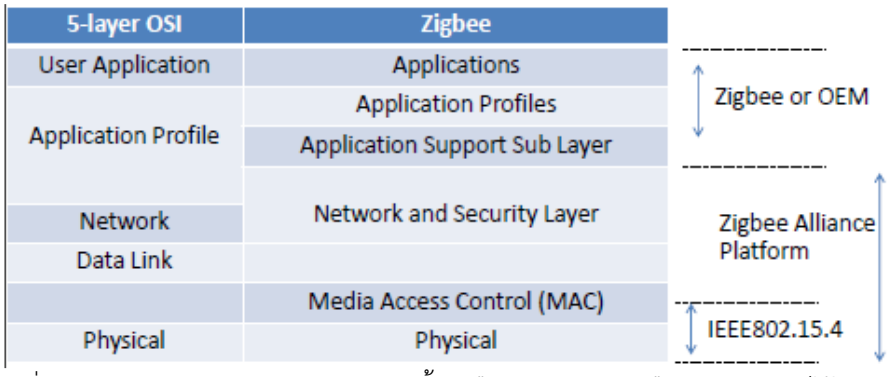
จุดประสงค์การเรียนรู้

เพื่อให้สามารถเข้าใจการทำงานพื้นฐานของระดับชั้นในเครือข่ายเซนเซอร์ไร้สายและเข้าใจการทำงานของโพรโทคอลในชั้นเครือข่าย

เครือข่ายเซนเซอร์ไร้สายเป็นส่วนหนึ่งของระบบเครือข่ายซึ่งมีมาตรฐานที่รู้จักกันในชื่อ IEEE802.15.4 และมาตรฐาน Zigbee ซึ่งสามารถอธิบายเปรียบเทียบความแตกต่างของมาตรฐานชั้นเครือข่ายระหว่างเครือข่ายเซนเซอร์ไร้สาย และเครือข่ายปกติได้ดังภาพประกอบที่ 5.1 ชั้นเครือข่ายทางซ้ายมือคือมาตรฐาน OSI 5-layer ประกอบด้วยชั้นกายภาพ (Physical layer) ชั้น Data Link layer ชั้นเครือข่าย (Network layer) ชั้น Application Profile layer (ประกอบด้วยชั้น Transport, Session และ Presentation ใน OSI 7-layer) และชั้น User Application layer ส่วนลำดับชั้นเครือข่ายทางขวามือคือมาตรฐาน Zigbee และ IEEE802.15.4 ที่ใช้ในเครือข่ายเซนเซอร์ไร้สาย ซึ่งประกอบด้วยชั้นล่างสุดคือชั้นกายภาพเหมือนกัน ลำดับต่อมาคือชั้น Media Access Control (MAC) ซึ่งก็คือชั้น Data Link ใน OSI ปกติ ชั้นถัดไปคือชั้น Network and security เทียบได้กับชั้นเครือข่ายที่อาจจะรวมถึงชั้น Transport ของ OSI ปกติ ต่อมาคือชั้น Application support sub layer และชั้น Application profile เทียบได้กับบางส่วนของชั้น Application profile สุดท้ายคือชั้น Applications

พบว่าในชั้นกายภาพและบางส่วนของชั้น MAC ของเครือข่ายเซนเซอร์ไร้สายจะเป็นไปตามมาตรฐาน IEEE802.15.4 และบางส่วนของชั้น MAC ชั้น Network and security ชั้น Application support ที่วิ่งอยู่บนมาตรฐาน IEEE802.15.4 ก็จะเป็นไปตามมาตรฐาน Zigbee Alliance (ขึ้นกับผู้ผลิตฮาร์ดแวร์ platform) ส่วนใน

ชั้น Application profile และ Applications จะเป็นมาตรฐาน Zigbee กลางเพื่อให้อุปกรณ์สามารถสื่อสารและใช้งานร่วมกันได้ ในการออกแบบและพัฒนาเครือข่ายเซนเซอร์ไร้สายใช้งานจริงผู้พัฒนาสามารถพิจารณาลดรูปของชั้นต่างๆในเครือข่ายเหล่านี้ลงได้เพื่อให้สามารถทำงานได้เร็วขึ้นหรือการใช้พลังงานให้มีประสิทธิภาพมากขึ้น เทคนิคเหล่านี้สามารถพบในงานวิจัยประเภท Cross-layer design สำหรับในบทนี้จะขอกล่าวถึงเฉพาะชั้นกายภาพ ชั้น MAC และเน้นทำความเข้าใจกับโพรโทคอลค้นหาเส้นทางในชั้นเครือข่ายเนื่องจากมีความสำคัญต่อประสิทธิภาพและการใช้พลังงานของเครือข่าย นอกจากนี้แล้วถ้าเข้าใจวิธีการค้นหาเส้นทางก็จะเข้าใจกระบวนการทำงานของโนดภายในเครือข่ายเซนเซอร์ไร้สายที่ทำให้สามารถต่อยอดงานอื่นๆได้ เช่นเรื่องความปลอดภัยของข้อมูล การแก้ไขการโจมตีโนด (Attack) และการหาตำแหน่ง (Location) เป็นต้น



ภาพประกอบที่ 5.1 ความแตกต่างระหว่างมาตรฐานชั้นเครือข่ายระหว่างเครือข่ายเซนเซอร์ไร้สายและเครือข่ายทั่วไป

5.1 Physical Layer

ชั้นกายภาพทำหน้าที่ในการแปลงข้อมูลดิจิตอลระดับบิตให้เป็นสัญญาณที่เหมาะสมในการส่งผ่านโมดูลคลื่นวิทยุ ชั้นกายภาพยังทำหน้าที่เลือกคลื่นความถี่ การสร้างคลื่นความถี่ที่เป็นพาหะ (Carrier frequency) การตรวจจับสัญญาณ การมอดูเลชั่น และการเข้ารหัส เทคโนโลยีของการสื่อสารคลื่นวิทยุในเครือข่ายเซนเซอร์ไร้สายสามารถแบ่งได้เป็น 3 ชนิดได้แก่

1) Narrow-band เน้นการลดรูปให้ใช้ Bandwidth อย่างมีประสิทธิภาพโดยใช้วิธีการมอดูเลชั่นแบบ M-ary ในช่วงต้นของการพัฒนาโนดสำหรับเครือข่ายเซนเซอร์ไร้สายมีการใช้เทคนิค Narrow-band เช่นโนด

Platform Mica2 ที่ใช้ชิป CC1000 ทำงานที่ความถี่ 433, 868 และ 915 MHz ด้วยขนาดของ Bandwidth สูงถึง 175 kHz มีอัตราการส่งข้อมูลที่ 76 kbps

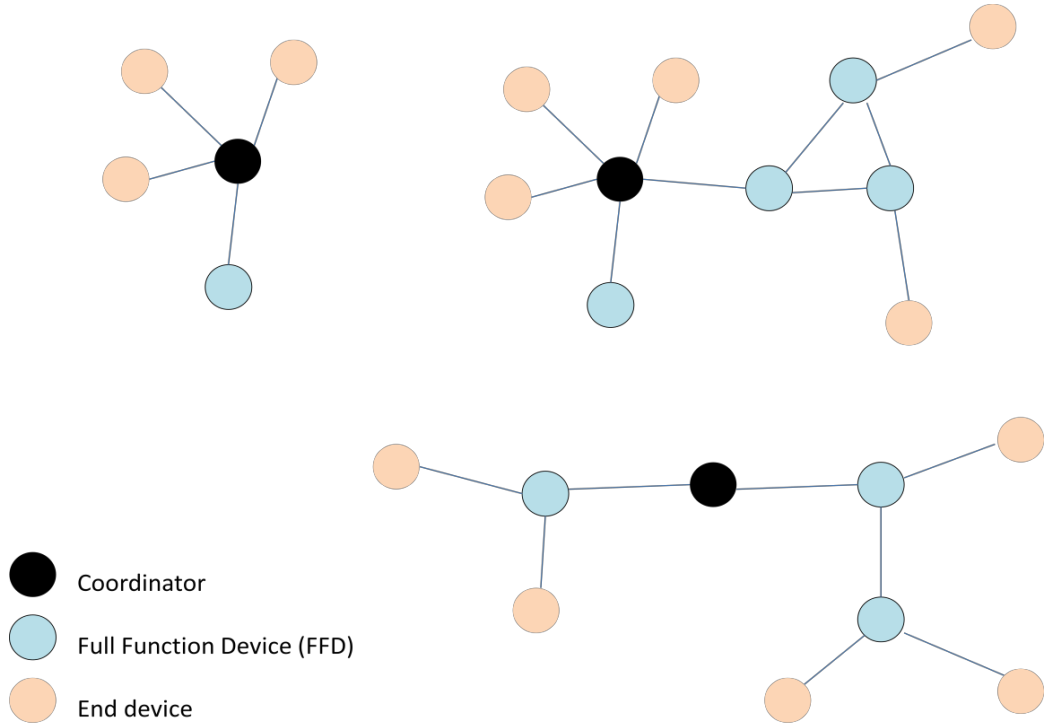
2) Spread-spectrum นี้ถูกใช้ในการสื่อสารผ่านคลื่นวิทยุเพื่อปรับปรุงอัตราการส่งข้อมูลและให้ทนทานต่อสัญญาณรบกวน ซึ่งเทคนิคนี้มีอยู่ด้วยกัน 2 ชนิดคือ Frequency Hopping Spread Spectrum (FHSS) และ Direct Sequence Spread Spectrum (DSSS) โดยที่ FHSS เกิดจากการที่ Wide-band spectrum ถูกแบ่งย่อยไปเป็นช่องความถี่ย่อยๆ โดยที่การรับและส่งจะถูก pair กับเข้าช่องสัญญาณตามวิธีการ hopping ที่กำหนดไว้ ตัวอย่างการสื่อสารผ่านคลื่นวิทยุด้วยวิธีการ FHSS นี้ได้แก่ Bluetooth ในขณะที่เทคนิค DSSS จะอยู่บนพื้นฐานของรหัส Pseudo-noise (PN) หรือที่เรียกว่า *Chips* โดยที่ Chips ที่มีอัตราการไหลที่สูงกว่าข้อมูลบิตระดับบิต จะถูกนำไปมอดูเลชั่นกับข้อมูลระดับบิตก่อนที่จะมีการส่งออก ซึ่ง DSSS นี้เองเป็นมาตรฐานที่ใช้ในเครือข่ายเซนเซอร์ไร้สาย IEEE802.15.4 ได้แก่ชิป CC2420 ใช้คลื่นความถี่ที่ 2.4 GHz มีอัตราการการส่ง chips ที่ 2 Mchips/s และ อัตราการส่งข้อมูลที่ 250 kbps

3) Ultra-wide-band (UWB) เป็นอีกรูปแบบหนึ่งของการส่งข้อมูลผ่านคลื่นวิทยุ UWB อาศัยการรับส่งข้อมูลแบบ Baseband ดังนั้นจึงไม่ต้องใช้คลื่นพาหะ โดยทั่วไปจะใช้วิธีการมอดูเลชั่นแบบ Pulse Position Modulation (PPM) ซึ่ง UWB ได้รับความสนใจที่จะนำมาใช้ในระดับชั้นกายภาพของเครือข่ายเซนเซอร์ไร้สายที่ต้องการรับส่งข้อมูลภาพและเสียง แม้ว่า UWB จะมีข้อดีเหนือว่า Spread spectrum เพราะไม่ได้ใช้คลื่นพาหะในการส่งข้อมูลทำให้มีวงจรภาครับส่งที่ซับซ้อนน้อยกว่า แต่ก็มีข้อจำกัดที่จะสามารถทำการส่งข้อมูลได้ดีภายในรัศมีน้อยกว่า 10 เมตร

สำหรับเทคนิคการสื่อสารในเครือข่ายเซนเซอร์ไร้สายรูปแบบอื่นที่น่าสนใจได้แก่ Acoustic communication ที่เหมาะสำหรับการพัฒนาเครือข่ายเซนเซอร์ไร้สายใต้น้ำ (Underwater Wireless Sensor Networks, UWSNs) เนื่องจากพาหะของการสื่อสารถูกเปลี่ยนจากอากาศเป็นน้ำ คลื่นเสียง หรือ Acoustic wave สามารถเคลื่อนที่ผ่านตัวกลางที่เป็นน้ำได้อย่างดีและสามารถส่งสัญญาณไปได้เป็นระยะทางไกล แต่การสื่อสารในน้ำก็ยังมีข้อจำกัดในหลายเรื่องเช่น Path loss, สัญญาณรบกวน (Noise), Doppler และค่าหน่วงเวลาในการส่ง เป็นต้น ทำให้ยังคงเป็นเรื่องที่น่าสนใจในการทำวิจัยในปัจจุบัน ซึ่งสามารถอ่านงานวิจัยที่เกี่ยวเนื่องกับ UWSNs ได้จาก [1-4]

ส่วนหลักการพื้นฐานของการเข้ารหัสช่องสัญญาณ (Channel coding) หรือการมอดูเลชั่นหรือทฤษฎีอื่นๆ ที่เกี่ยวข้องจะไม่ขอกล่าวไว้ในหนังสือเล่มนี้ เนื้อหาภาคทฤษฎีและงานวิจัยเชิงลึกจะถูกอธิบายไว้ในหนังสือเครือข่ายเซนเซอร์ไร้สายเล่มที่ 2 มีเฉพาะรายละเอียดของมาตรฐาน IEEE802.15.4 ที่จะถูกอธิบายไว้ในที่นี้

มาตรฐาน IEEE802.15.4 [5] ได้กำหนดคุณลักษณะของการรับส่งข้อมูลแบบไร้สายที่มีอัตราการส่งข้อมูลต่ำ ใช้พลังงานต่ำ และมีความซับซ้อนน้อย และได้กำหนดรูปแบบของโนดเพื่อให้สามารถเชื่อมต่อเครือข่ายได้หลากหลายดังภาพประกอบที่ 5.2 โดยโนดมีด้วยกัน 3 แบบคือ Coordinator (ทุกเครือข่ายจะต้องมีโนดชนิดนี้และมี Coordinator 1 ตัวเพื่อทำหน้าที่ติดต่อกับเครื่องแม่ข่ายหรือส่งข้อมูลออกนอกวงจรเครือข่ายเซนเซอร์ไร้สาย) Full Function Device (FFD) ทำหน้าที่เป็นทั้งโนดรับข้อมูลจากเซนเซอร์หรือเป็น Router ได้ด้วย และสุดท้ายคือโนด End device หรือเรียกว่า Reduce Function Device โนดแบบนี้ทำหน้าที่รับข้อมูลจากเซนเซอร์แล้วส่งต่อข้อมูลให้โนด FFD หรือ Coordinator เท่านั้น



ภาพประกอบที่ 5.2 ตัวอย่างการเชื่อมต่อเครือข่ายด้วยโนด 3 รูปแบบ

ในชั้นกายภาพกำหนดให้มีคลื่นความถี่ใช้งานได้ 3 ย่านได้แก่ 1) 2.4 GHz สามารถใช้ได้ทั่วโลกเป็นคลื่นความถี่สาธารณะ ประกอบด้วย 16 ช่องสัญญาณ 2) 915 MHz ใช้ในทวีปอเมริกาประกอบด้วย 30 ช่องสัญญาณ และ 3) 868 MHz ใช้ในยุโรปมีเพียง 1 ช่องสัญญาณให้ใช้งานเท่านั้น สำหรับในประเทศไทยเราสามารถใช้คลื่นความถี่ที่ 2.4 GHz ส่วนคลื่นย่าน UWB ที่ความถี่ 1, 3-5 และ 6-10 GHz ได้ถูกกำหนดไว้ในมาตรฐาน IEEE802.15.4a

5.2 Media Access Control (MAC)

ในส่วนของ MAC ของเครือข่ายเซนเซอร์ไร้สายจะกล่าวถึงการควบคุมการเชื่อมต่อระหว่างโนด รวมถึงการควบคุมชนกันของข้อมูล (Collisions) เมื่อโนดอยู่ใกล้กันหรือต้องการส่งข้อมูลในเวลาเดียวกัน มีงานวิจัยหลายชิ้นได้นำเสนอโพรโทคอล MAC สำหรับใช้ในเครือข่ายเซนเซอร์ไร้สาย โดยที่สามารถจัดแบ่งออกได้เป็น 3 กลุ่ม ได้แก่

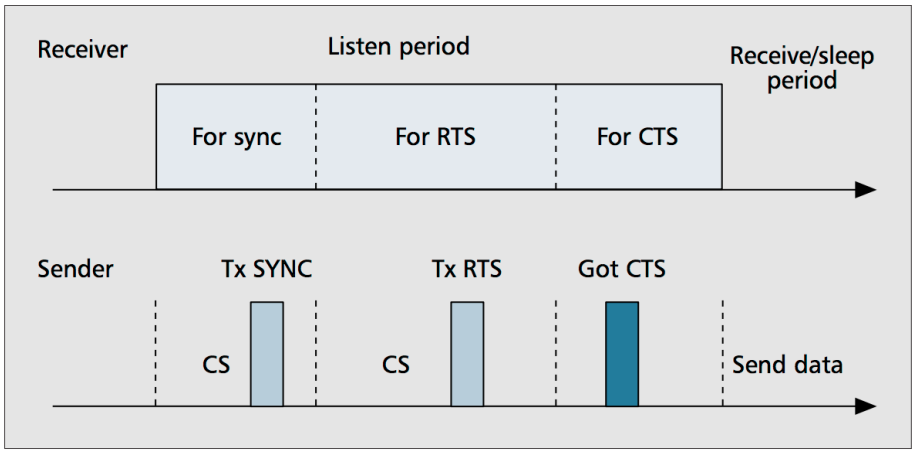
1) Contention-based medium access ได้แก่ B-MAC [6], S-MAC [7], WiseMAC [8] และ DSMAC [9] เป็นต้น
2) Reservation-based medium access ได้แก่ BMA-MAC [10] และ TRAMA [11] เป็นต้น
3) Hybrid solution ในกลุ่มนี้มีโพรโทคอล MAC ตามมาตรฐาน IEEE802.15.4 [5] และ Z-MAC [12]

สำหรับการอธิบายการทำงานของ S-MAC, WiseMAC และ DSMAC เป็นส่วนของหนึ่งการทบทวนวรรณกรรมที่เกี่ยวข้องในงานวิทยานิพนธ์ระดับปริญญาโทของนายจิระศักดิ์ รักษาชุม และการทำงานของ MAC มาตรฐาน IEEE802.15.4 เป็นส่วนหนึ่งของการทบทวนวรรณกรรมที่เกี่ยวข้องในงานวิทยานิพนธ์ระดับปริญญาโทของนายอนิรุธ ทองกลิ่น

5.2.1 การทำงานของ S-MAC

S-MAC เป็น MAC โพรโทคอลที่ถูกออกแบบขึ้นมาเพื่อใช้งานกับเครือข่ายเซนเซอร์ไร้สายในยุคแรก ซึ่งได้ปรับปรุงโพรโทคอลมาจาก โหมดประหยัดพลังงานของมาตรฐาน IEEE 802.11 ทำงานโดยการกำหนดช่วงเวลาหลับ และตื่น ของอุปกรณ์เพื่อประหยัดพลังงาน โดยการสื่อสารจะทำได้เฉพาะช่วงเวลาที่อุปกรณ์ตื่นเท่านั้น และการเข้าถึงช่องสัญญาณจะใช้วิธีการแย่งชิงช่องสัญญาณ โดยการส่งเฟรมควบคุมดังภาพประกอบที่ 5.3 ซึ่งประกอบด้วย SYNC RTS/CTS และ ACK โดย SYNC จะทำหน้าที่กำหนดจุดเริ่มต้นของการสื่อสาร และหลังจากนั้นหากมีอุปกรณ์ตัวใดต้องการส่งข้อมูลจะส่ง RTS ออกมาเพื่อขอส่งข้อมูล และอุปกรณ์ตัวที่ต้องการรับข้อมูลจะ

ตอบ CTS กลับมาหากพร้อมที่จะรับข้อมูล เมื่อผู้ที่ต้องการจะส่งข้อมูลได้รับสัญญาณ CTS ก็จะส่งข้อมูลออกมาให้ผู้รับ เมื่อผู้รับสามารถรับข้อมูลได้ครบก็จะตอบ ACK ไปยังผู้ส่งเพื่อยืนยันการได้รับข้อมูล



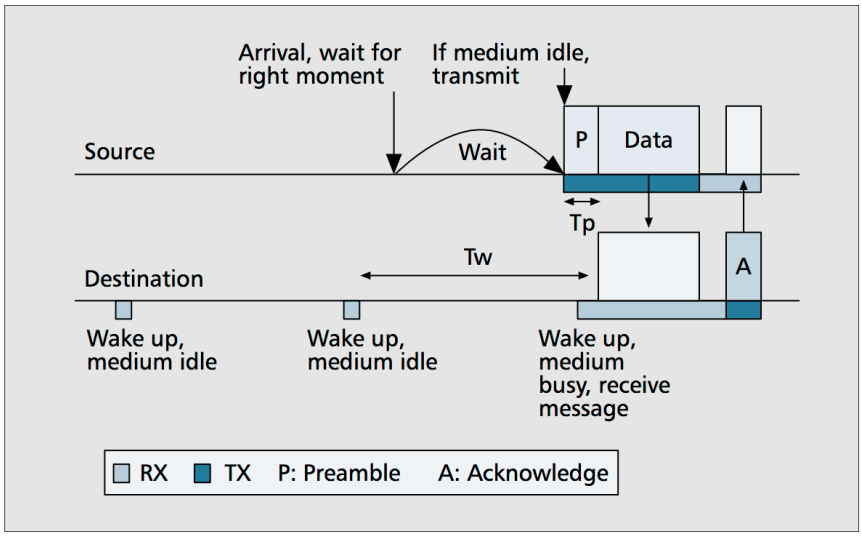
ภาพประกอบที่ 5.3 วิธีการรับส่งข้อมูลของ S-MAC [7]

ข้อดีของ S-MAC คือ สามารถลดการใช้พลังงานที่เกิดจาก การฟังโดยไม่มีข้อมูลส่งออกมาได้ ด้วยการหลับ และตื่น ตามคาบเวลา นอกจากนี้ยังเป็นโพรโทคอลที่สามารถนำไปพัฒนาได้ง่าย เนื่องจากไม่มีความซับซ้อนมากนัก ข้อเสียของ S-MAC คือ การส่งข้อมูลแบบ Broadcast นั้นจะไม่ใช้เฟรมควบคุม RTS/CTS ซึ่งทำเพิ่มโอกาสการชนกันของข้อมูล และคาบการหลับ และตื่นของอุปกรณ์นั้น ถูกกำหนดค่าไว้ล่วงหน้า ไม่สามารถเปลี่ยนแปลงได้ ซึ่งจะทำให้ประสิทธิภาพของการใช้พลังงานลดลงเมื่อใช้งานในเครือข่ายที่มีอัตราการส่งข้อมูลที่ไม่แน่นอน

5.2.2 การทำงานของ WiseMAC

WiseMAC ได้พัฒนามาจากแนวคิดของโพรโทคอลชื่อ “Spatial TDMA and CSMA with Preamble Sampling” ที่คิดค้นโดย A. El-Hoiydi ซึ่งใช้การสื่อสารแบบสองช่องสัญญาณ โดยช่องสัญญาณแรกจะใช้สำหรับการส่งข้อมูล และใช้การเข้าถึงช่องสัญญาณแบบ TDMA ส่วนช่องสัญญาณที่สอง ใช้สำหรับส่งคำสั่งควบคุม และใช้การเข้าถึงช่องสัญญาณแบบ CSMA แต่ WiseMAC จะใช้ช่องสัญญาณเพียงช่องเดียว และใช้วิธีการเข้าถึงช่องสัญญาณแบบ non-persistent CSMA (np-CSMA) [13] โดยใช้กระบวนการ Preamble sampling แบบเดียวกับโพรโทคอล Spatial TDMA and CSMA with Preamble Sampling เพื่อลดการรอรับข้อมูลโดยไม่มีข้อมูลส่งออกมา ซึ่งวิธีการทำงานของ Preamble Sampling นั้นได้แสดงไว้ดังภาพประกอบที่ 5.4 คือ จะมีการส่ง

สัญญาณ Preamble ออกมาก่อนที่จะส่งข้อมูลแต่ละเฟรม เพื่อใช้สำหรับบอกโนดตัวที่จะรับข้อมูล ให้รอรับข้อมูล ซึ่งเมื่อโนดตัวที่ต้องการจะรับข้อมูลตื่นขึ้นมา แล้วพบว่าช่องสัญญาณไม่ว่าง โนดตัวนั้นก็จะรอรับข้อมูลต่อไป จนกว่าจะได้รับข้อมูลหรือจนกว่าช่องสัญญาณจะว่างอีกครั้ง และเมื่อได้รับข้อมูลแล้วก็จะตอบ Acknowledge กลับไปยังตัวที่ส่งข้อมูล เพื่อยืนยันการได้รับข้อมูล



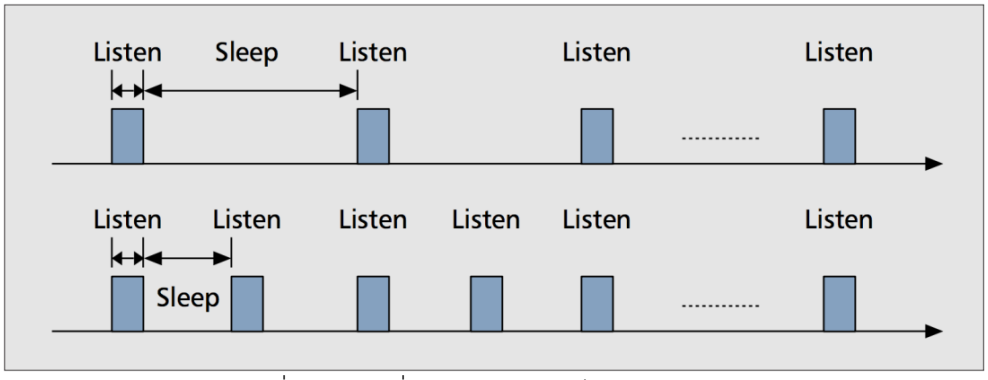
ภาพประกอบที่ 5.4 วิธีการรับส่งข้อมูลแบบ WiseMAC [8]

ข้อดีของ WiseMAC คือ ผลจากการจำลองการทำงานพบว่าประสิทธิภาพของ WiseMAC นั้นทำได้ดีกว่า S-MAC ภายได้สภาพเครือข่ายที่มีอัตราการส่งข้อมูลไม่คงที่ และยังสามารถจัดการปัญหา ความเหลื่อมล้ำของเวลาได้ ทำให้ลดการพึ่งพาการ Synchronize เวลาจากภายนอก ข้อเสียของ WiseMAC คือ การที่ WiseMAC ไม่ได้มีการ Synchronize ตารางการหลับและตื่นให้ตรงกันในแต่ละโนดที่อยู่ใกล้กัน ทำให้เกิดปัญหาคือ เมื่อมีการส่งข้อมูลแบบ Broadcast ข้อมูลจะต้องถูกเก็บไว้สำหรับโนดที่อยู่ใกล้กัน หากโนดนั้นหลับอยู่ และจะส่งออกไปก็ต่อเมื่อโนดนั้นได้ตื่นขึ้นมา ซึ่งทำให้มี Latency เพิ่มขึ้น และใช้พลังงานมากขึ้น

5.2.3 Dynamic Sensor – MAC (DSMAC)

DSMAC เป็นโพรโทคอลที่พัฒนาต่อมาจาก S-MAC ซึ่งเพิ่มความสามารถในการปรับเปลี่ยน Duty-cycle เพื่อลด Latency ของเครือข่าย โดยในช่วงการ SYNC ทุกโนดจะรายงาน Latency ของตัวเอง (เวลาตั้งแต่รับข้อมูลเข้ามาในคิว จนกระทั่งได้ส่งข้อมูลออกไปและในช่วงเริ่มต้น ทุก (โนดจะมี Duty-cycle ที่ตรงกัน เมื่อตัวที่ต้องการรับข้อมูลพบว่าค่าเฉลี่ย Latency มีค่าสูงเกินไป มันตัดสินใจลดเวลาการหลับของตัวเองลง และประกาศ

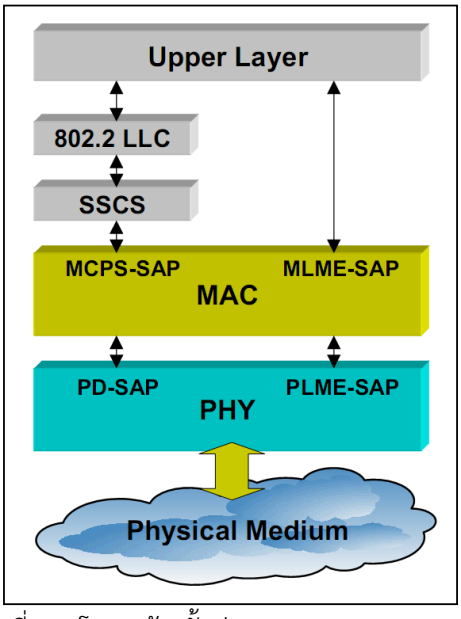
ให้โนดอื่นรับรู้ผ่าน ช่วงเวลา SYNC เมื่อตัวที่ต้องการจะส่งข้อมูล ได้รับสัญญาณการลดช่วงเวลาการหลับของตัวรับข้อมูล มันจะตรวจสอบคิวสำหรับส่งข้อมูลไปยังปลายทาง หากมีค่าเป็นหนึ่ง มันจะตัดสินใจเพิ่ม Duty-cycle เป็นสองเท่าดังภาพประกอบที่ 5.5 หากพลังงานในแบตเตอรี่อยู่ในช่วงที่กำหนด เพื่อลด Latency ลง



ภาพประกอบที่ 5.5 การเพิ่ม Duty-cycle เป็นสองเท่าของ DSMAC[9]

5.2.5 การทำงานของ IEEE802.15.4 MAC

ชั้นย่อยการควบคุมการเข้าใช้สื่อกลางนำเสนออินเตอร์เฟสการเชื่อมต่อระหว่างชั้นย่อย SSCS (Service Specific Convergence Sublayer) และชั้นกายภาพดังภาพประกอบที่ 5.6 ในชั้นนี้ได้มีการนำเสนอบริการ 2 ชนิดเช่นเดียวกับชั้นกายภาพคือMAC data unit และ MAC management unit ชั้นย่อย MAC จะเป็นชั้นที่กำหนดว่าสื่อกลางควรได้รับการเข้าใช้งานอย่างไรจากอุปกรณ์ ชั้นย่อย MAC นำเสนอการเข้าใช้ช่องสัญญาณทางกายภาพสำหรับการส่งทุกประเภท ชั้นที่อยู่สูงกว่าคือชั้น IEEE 802.2 LLC (Logical Link Control) ซึ่งเป็นชั้นที่รับผิดชอบพื้นฐานเกี่ยวกับลอจิคัลแอดเดรส การควบคุมความผิดพลาด และเป็นชั้นที่สามารถเข้าถึงชั้น MAC ผ่านชั้น SSCS สำหรับชั้น 802.2 LLC นี้ได้รับการออกแบบมาเพื่อสนับสนุนการควบคุมและการดูแลงานประเภทอุตสาหกรรมและอุปกรณ์ภายในบ้านจำนวนมาก ซึ่งงานประเภทนี้ต้องการอัตราการส่งข้อมูลต่ำถึงปานกลางและมีการรับประกันความล่าช้าในการส่ง (Delay) ที่สามารถยืดหยุ่นได้



ภาพประกอบที่ 5.6 โครงสร้างชั้นย่อย MAC มาตรฐาน IEEE802.15.4

หน้าที่และฟังก์ชันของชั้น MAC ตามมาตรฐาน IEEE802.15.4 ประกอบด้วย

1) สร้างเบคอนเครือข่ายเมื่อเป็นอุปกรณ์ประเภท Coordinator ซึ่งจะตัดสินใจว่าจะทำงานในโหมดเบคอนหรือไม่ (beacon enabled mode) ซึ่งจะเกี่ยวข้องกับการใช้ Superframe โดยปกติ Superframe จะเต็มไปด้วยเบคอนเครือข่ายและมีการแบ่งเป็น 16 ช่อง (slot) เท่าๆ กัน และจะแบ่งออกเป็นคาบ Active และคาบ Inactive กล่าวคือ คาบ Active จะมีการแบ่งย่อยเป็นสองส่วนคือ Contention Access Period (CAP) ที่ใช้ขอเข้าใช้สื่อกลางด้วยอัลกอริทึม CSMA-CA และ Contention Free Period (CFP) ที่มีกลไก Guaranteed Time Slot (GTS) สนับสนุนการรับประกันความน่าเชื่อถือการเชื่อมต่อสื่อสาร ส่วนคาบ Inactive Coordinator จะไม่สามารถโต้ตอบกับเครือข่ายและใช้โหมดประหยัดพลังงานได้ โดยปกติ Coordinator จะส่งเบคอนด้วยคาบเวลาที่กำหนดไว้เพื่อเทียบจังหวะสัญญาณนาฬิกาให้ตรงกัน (Synchronization) หรือเพื่อวัตถุประสงค์อื่น และอุปกรณ์ประเภท FFD จะเริ่มส่งเฟรมเบคอนเมื่อเชื่อมต่อกับ Coordinator อย่างสมบูรณ์
2) การเทียบจังหวะสัญญาณนาฬิกาให้ตรงกับเบคอน (Synchronizing to the beacons) อุปกรณ์ที่เชื่อมต่อกับ Coordinator จะทำงานตามเบคอนที่ได้ซิงโครไนซ์กับ Coordinator การทำเข้าจังหวะกันด้วยกระบวนการทำ polling ซึ่งเป็นรูปแบบการทำงานตามคิว การทำงานในโหมดประหยัดพลังงานและการค้นหาอุปกรณ์ที่ไม่ได้รับการกำหนดลงในเฟรมเบคอน
3) สนับสนุนการเชื่อมต่อและยกเลิกการเชื่อมต่อเครือข่ายส่วนบุคคล (PAN) ชั้น MAC จะมีฟังก์ชันการเชื่อมต่อและยกเลิกการเชื่อมต่ออยู่ภายในสำหรับการเริ่มระบบเครือข่ายด้วยตัวเอง (Self-configuration)

ที่มีรูปแบบการเชื่อมต่อแบบดาวและจุดต่อจุด Coordinator จะส่งเบคอนเป็นระยะๆ ตามเบคอนที่ได้รับกาจัดสรรจาก PAN coordinator กระบวนการขอการเชื่อมต่อ อุปกรณ์จะส่งข้อความขอการเชื่อมต่อไปยัง PAN coordinator ซึ่งจะได้รับข้อความตอบกลับขึ้นอยู่กับว่าทรัพยากรมีเพียงพอหรือไม่ ส่วนการยกเลิกการเชื่อมต่อนั้นจะทำได้โดยอุปกรณ์เองหรือ PAN coordinator ก็ได้

4) ฟังก์ชันการขอเข้าใช้สื่อกลางด้วยอัลกอริทึม CSMA-CA อัลกอริทึมการขอเข้าใช้สื่อกลางชนิดนี้เหมือนกับอัลกอริทึมการขอเข้าใช้สื่อกลางสำหรับเครือข่ายไร้สายอื่นๆ แต่จะไม่มีกลไก Request-To-Send (RTS) และ Clear-To-Send (CTS) ในการพิจารณาว่าอุปกรณ์จะใช้โหมดการทำงานแบบ slotted CSMA-CA หรือ unslotted CSMA-CA นั้นจะขึ้นอยู่กับว่า PAN coordinator มีการทำงานในโหมดเบคอนหรือไม่ตามลำดับ

5.3 Network Layer

ระบบเครือข่ายเซนเซอร์ไร้สายมีคุณลักษณะที่สามารถทำงานได้ด้วยตนเองหรือที่เรียกว่า Self-organization เครือข่ายแบบนี้จะประกอบไปด้วยอุปกรณ์ขนาดเล็กที่ใช้พลังงานจากแบตเตอร์รี่ และราคาถูก จำนวนหลายร้อยหรือหลายพันตัว โดยมีความท้าทายที่ว่าจะต้องทำให้ระบบเครือข่ายนี้สามารถทำงานได้ยาวนานที่สุด เนื่องจากอุปกรณ์ขนาดเล็กมีพลังงานให้สามารถใช้ได้ในจำนวนที่จำกัด โดยมีการทำงานที่เป็นพื้นฐานสำคัญได้แก่ การค้นหาโนดทั้งหมดในระบบ การสร้างเชื่อมต่อเส้นทาง และการบำรุงรักษาเส้นทางเพื่อให้ระบบยังคงสามารถรักษาสถานภาพการเชื่อมต่อแบบเครือข่ายไว้ได้

สิ่งสำคัญของการจัดการเกี่ยวกับการสร้างการเชื่อมต่อจากต้นทางไปยังปลายทางก็คือ โพรโทคอลในระดับเครือข่าย ซึ่งจะใช้วิธีการหลักๆ 2 ส่วน ดังนี้ 1) ส่งสัญญาณจากต้นทางไปยังปลายทาง เพื่อทำการค้นหาเส้นทางหลัก และ 2) ปลายทางจะเลือกเส้นทางที่ดีที่สุดตามวิธีการของโพรโทคอลนั้นๆ แล้วส่งสัญญาณกลับมายังต้นทาง นอกจากนั้นระบบเครือข่ายเซนเซอร์ไร้สาย ถ้าจะมองในส่วนของโพรโทคอลระดับเครือข่าย ลักษณะของการวางตำแหน่งโนด การกระจายตัวของโนด ความเร็วของการเคลื่อนที่ของโนด จะมีผลต่อสมรรถนะในการทำงาน ดังนั้นจะต้องหาโพรโทคอลให้เหมาะสมกับงานที่จะนำมาใช้ ยกตัวอย่างเช่น เราต้องการออกแบบระบบที่โนดเคลื่อนที่ช้าจำนวนโนดน้อย และไม่ต้องการคุณภาพของการบริการที่มากนัก ก็ให้เลือกใช้โพรโทคอล Geographic and Energy Aware Routing (GEAR) หรือ Sensor Protocols for Information via Negotiation (SPIN) [14] หรือถ้าหากออกแบบให้สถานีฐาน (Base Station) อยู่กับที่ และมีโนดข้างเคียงจำนวนมาก ลักษณะเช่นนี้ควรจะเลือกใช้โพรโทคอล Low Energy Adaptive Clustering Hierarchy (LEACH) [15], Threshold-Sensitive

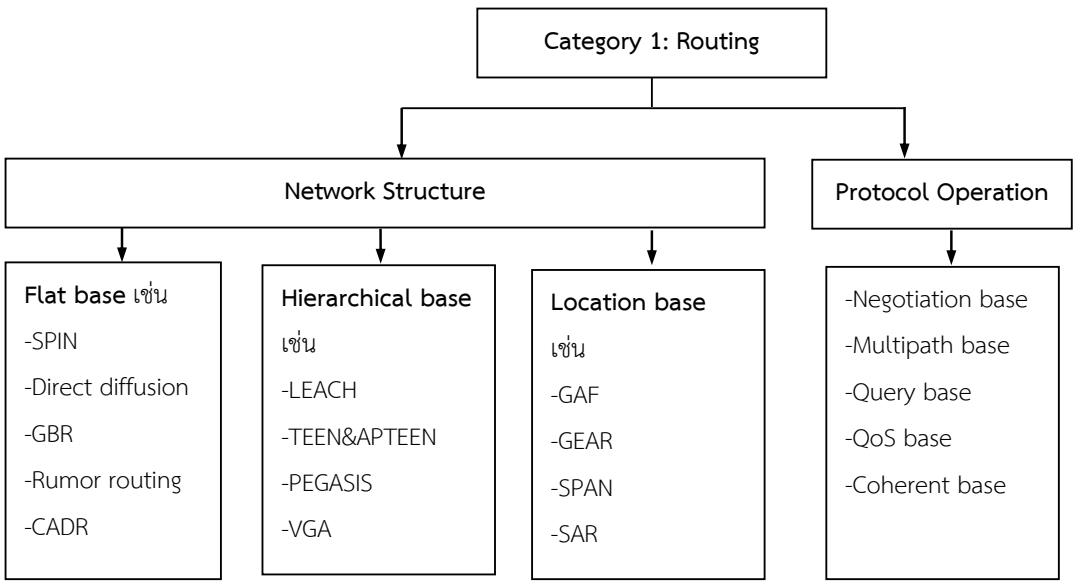
Energy Efficient Sensor Network Protocols (TEEN) [16] และ Power-Efficient Gathering in Sensor Information Systems (PEGASIS) [17] เป็นต้น โพรโทคอลการค้นหาเส้นทางสำหรับระบบเครือข่ายเซนเซอร์ไร้สาย จะขึ้นอยู่กับปัจจัยเหล่านี้

- **ปริมาณของโนด** ในพื้นที่สื่อสารพบว่าปริมาณอาจจะมี 100 โนด หรืออาจจะมากกว่านั้น ดังนั้นจะทำอย่างไรที่จะหาวิธีการจัดหาเส้นทางให้เหมาะสมกับปริมาณของโนดที่มากขึ้น ซึ่งเป็นเหตุให้มีการออกแบบโพรโทคอล การค้นหาเส้นทางที่รองรับจำนวนโนดในสภาวะแวดล้อมต่างๆกัน โดยเฉพาะอย่างยิ่งเมื่อกรณีของเครือข่ายเกิดปัญหาขาดการติดต่อสื่อสาร โนดเซนเซอร์ไร้สายก็จะส่งสัญญาณไปบอกต้นทาง และถ้าโนดมีปริมาณมากจะทำให้เกิด overhead มากขึ้น จึงเกิดการทำวิจัยเช่น กระบวนการทำ Cooperation system หรือ Clustering Hierarchical [18] เป็นต้น
- **รูปแบบของเครือข่าย** ในระบบเครือข่ายเซนเซอร์ไร้สาย จะมีทั้งโนดอยู่กับที่ ซึ่งส่วนใหญ่จะเป็นสถานีฐาน และโนดเคลื่อนที่ แต่จะมีการใช้งานบางอย่าง ที่ทั้งสถานีฐาน และโนดเซนเซอร์ไร้สาย สามารถเป็นโนดเคลื่อนที่ ลักษณะดังกล่าวจะมีผลต่อการวิจัยในด้านพลังงาน หรือแบนด์วิดท์ ยิ่งไปกว่านั้นลักษณะการกระจายตัวของโนดจะเป็นปัจจัยสำคัญ ถ้าหากโนดต้นทางอยู่ใกล้โนดปลายทาง ก็จะทำให้ใช้พลังงานน้อยลง แต่ถ้าหากโนดต้นทางหลายตัวส่งไปยังสถานีฐานตัวเดียว จะสร้างปัญหาให้กับการจัดการของแบนด์วิดท์ ดังนั้นจึงมีการแก้ปัญหาโดยการทำ Multichannel หรือ Multi-rate เกิดขึ้น
- **สื่อกลางสำหรับการส่งข้อมูล** สำหรับระบบนี้จะใช้วิธีการแบบ Multi-hop หมายความว่าโนดเพื่อนบ้านจะทำหน้าที่ช่วยส่งข้อมูลไปยังปลายทาง ดังนั้นสื่อกลางจึงเป็นตัวสำคัญ ซึ่งจะถูกออกแบบที่โพรโทคอล Medium Access Control (MAC) ในยุคแรกของการวิจัยจะใช้มาตรฐานของ IEEE802.11 เป็นมาตรฐานของระบบเครือข่ายที่ชื่อว่า Ad hoc หลังจากนั้นก็มีการพัฒนามาเป็น IEEE802.15.4 เป็นการออกแบบให้เหมาะสมกับเครือข่ายเซนเซอร์ไร้สาย มาในปัจจุบันก็มีเทคโนโลยีต่างๆ เข้ามาใช้กับเครือข่ายนี้ได้แก่ Bluetooth หรือ General Packet Radio Services (GPRS) เป็นต้น
- **การเชื่อมต่อระหว่างโนด** จะแบ่งออกเป็น 2 ช่วงคือ 1) การทำ Route discovery เพื่อหาเส้นทางสำหรับการส่งข้อมูล และ 2) การทำ Route maintenance เมื่อเกิดปัญหาของ Route failure หรือ Link failure ดังนั้นจะทำอย่างไรกับการรักษาการเชื่อมต่อระหว่างโนด ให้อยู่ได้ตลอดการส่งข้อมูล ในปัญหานี้จึงเกิดการวิจัยทางด้านโพรโทคอลระดับเครือข่ายสำหรับการค้นหาเส้นทาง หรือโพรโทคอลระดับ

MAC จึงมีงานวิจัยที่สนใจใช้หลักการที่เรียกว่า Cross layer [19, 20, 21, 22] ซึ่งเป็นการทำงานร่วมกันระหว่างระดับชั้นสื่อสาร นอกจากนั้นก็มีการพัฒนาที่เรียกว่า การทำ Optimization ที่พัฒนาระบบเครือข่าย ในแต่ละระดับชั้นให้ทำงานแก้ปัญหาเฉพาะด้าน

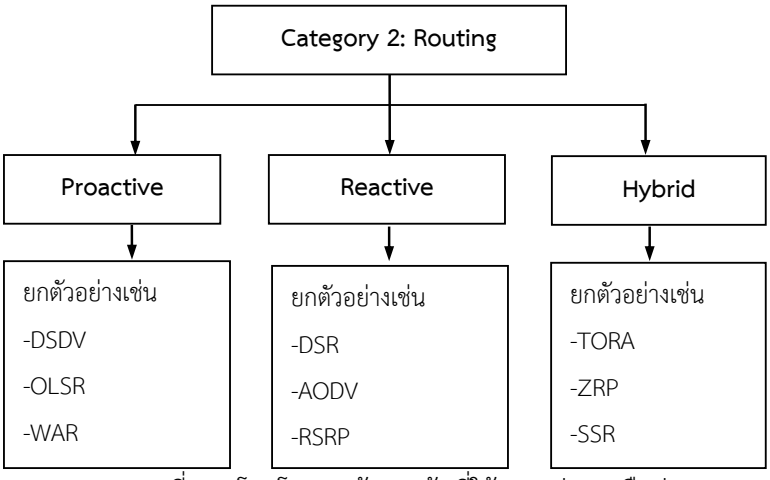
จากการศึกษาค้นคว้าจากบทความทางวิชาการ รวมทั้งประสบการณ์ในการออกแบบและพัฒนาระบบเครือข่ายเซนเซอร์ไร้สาย ผู้เขียนพบว่ามีกลุ่มงานวิจัยพัฒนาที่เกี่ยวข้องกับเครือข่ายบนเซนเซอร์ไร้สาย 2 กลุ่มใหญ่ คือ 1) กลุ่มที่ออกแบบและพัฒนาโพรโทคอลการเชื่อมต่อสำหรับโนดบนแนวคิดที่เป็นเซนเซอร์ไร้สาย 2) กลุ่มพัฒนาที่เน้นการนำโพรโทคอลการเชื่อมต่อเดิมที่ใช้งานอยู่บนเครือข่าย Ad hoc นำมาพัฒนาโดยเน้นข้อจำกัดของโนดในเรื่องของพลังงาน อัตราการส่งข้อมูลที่ 250 kbps และมีหน่วยความจำจำกัด

สำหรับกลุ่มแรกมีผู้วิจัย [23] แบ่งโพรโทคอลสำหรับการค้นหาเส้นทางออกเป็น 2 แบบคือ 1) Network Structure ซึ่งกลุ่มนี้จะขึ้นอยู่กับโครงสร้างของเครือข่าย ยกตัวอย่างเช่น ลักษณะแบบ Flat Base ทุกโนดจะถูกกำหนดให้มีหน้าที่การทำงาน และฟังก์ชันต่างๆมีลักษณะเดียวกัน ส่วน Hierarchical base โนดแต่ละกลุ่มจะมีหน้าที่ตามลำดับความสำคัญ และสุดท้าย Location base เป็นลักษณะการระบุตำแหน่งของโนดในเครือข่าย และ 2) Protocol Operation ในกลุ่มนี้จะขึ้นอยู่กับการจัดการโพรโทคอล อย่างเช่น Negotiation base, Multipath base, Query base, QoS base และ Coherent base วิธีการดังกล่าวก็เพื่อเพิ่มสมรรถนะการทำงานให้กับระบบเครือข่าย ดังแสดงในภาพประกอบที่ 5.7



ภาพประกอบที่ 5.7 แผนภาพแสดงการแบ่งโพรโทคอลค้นหาเส้นทาง [23]

กลุ่มที่สอง ดังภาพประกอบที่ 5.8 เป็นโพรโทคอลที่มาจากระบบเครือข่ายแบบ Ad hoc สำหรับกลุ่มนี้จะแบ่งออกเป็น 3 แบบได้แก่ Proactive Reactive และ Hybrid โดยที่โพรโทคอล Proactive จะค้นหาเส้นทางทุกเส้นทางก่อนที่ระบบต้องการส่งข้อมูล ในขณะที่โพรโทคอล Reactive จะค้นหาเส้นทางไปยังปลายทางก็ต่อเมื่อมีการร้องขอเส้นทางเกิดขึ้น ส่วนโพรโทคอล Hybrid จะเป็นการรวมกันของทั้ง 2 วิธีข้างต้น



ภาพประกอบที่ 5.8 โพรโทคอลค้นหาเส้นที่ใช้งานอยู่บนเครือข่าย Ad hoc

ดังนั้นโพรโทคอลการค้นหาเส้นทางที่ถูกพัฒนาขึ้น เพื่อทำให้ระบบเครือข่ายเซนเซอร์ไร้สายสามารถทำงานให้มีประสิทธิภาพมากที่สุด ทำอย่างไรเพื่อลดการใช้พลังงาน ทำอย่างไรกับข้อจำกัดในเรื่องหน่วยความจำ ผู้เขียนพบว่าโพรโทคอลจะขึ้นกับการนำไปประยุกต์ใช้งาน โดยที่จะต้องเลือกให้เหมาะสมกับงานที่จะนำไปใช้ จึงจะได้ประสิทธิภาพสูงสุด

จากบทความนี้แสดงให้เห็นว่าไม่เพียงแต่โพรโทคอลการค้นหาเส้นทางที่ต้องเลือกอัลกอริทึมการทำงานเพื่อวัดประสิทธิภาพ คุณภาพการให้บริการ รวมไปถึงปัจจัยต่างๆ ที่มีผลต่อเครือข่ายเซนเซอร์ไร้สาย อย่างเช่น ค่าของพลังงานที่จะใช้ในการค้นหาเส้นทาง พลังงานที่ต้องเสียไปเมื่อเกิดการสูญหายของข้อมูลเพื่อทำการค้นหาเส้นทางใหม่อีกครั้ง ยิ่งไปกว่านั้น การจัดสรรแบนด์วิดท์ยังมีอิทธิพลต่อลักษณะของระบบเครือข่ายแบบนี้ ซึ่งปัจจัยต่างๆที่ส่งผลต่อการเลือกใช้โพรโทคอลของเครือข่ายเซนเซอร์ไร้สายนั้นมาจากข้อจำกัดของฮาร์ดแวร์บนตัวโนด นั้นหมายรวมถึง เรื่องของพลังงานที่มีให้ใช้งานอย่างจำกัดจากแบตเตอร์รี่ ความแรงของสัญญาณที่ใช้ในรับส่งข้อมูลด้วยคลื่นวิทยุ เป็นต้น ดังนั้นผู้ที่สนใจจะต้องศึกษาความต้องการสำหรับการนำไปประยุกต์ใช้งาน และ

ข้อจำกัดของเครือข่ายเซนเซอร์ไร้สายเหล่านี้ เพื่อให้สามารถออกแบบและพัฒนาระบบเครือข่ายเซนเซอร์ไร้สายได้อย่างเหมาะสม

การจัดประเภทของโพรโทคอลการค้นหาเส้นทาง

จากการศึกษาพบว่างานวิจัยจำนวนมาก [24-28] มีการพัฒนาการจัดการเส้นทางการส่งข้อมูล เพื่อให้ได้โพรโทคอลการค้นหาเส้นทาง ที่สามารถรองรับการสื่อสารไร้สาย โดยเฉพาะการสื่อสารในรูปแบบเซนเซอร์ไร้สายที่มีข้อจำกัดทั้งแบนด์วิดท์ หน่วยความจำ พลังงาน และการประมวลผล ดังนั้นการออกแบบโพรโทคอลการค้นหาเส้นทาง จึงเป็นส่วนสำคัญที่จะทำให้ระบบสามารถทำงานได้ นอกจากนั้นก็จะพิจารณาถึงขนาดของเครือข่าย เพื่อรองรับจำนวนโนด หรือขนาดของข้อมูลรวมทั้งรูปแบบ (Topology) ของการสื่อสาร

สำหรับการจัดประเภทของโพรโทคอลการค้นหาข้อมูลมีการแบ่งประเภทตามข้อมูลที่แสดงไว้ข้างต้น ดังนั้นขอสรุป หรือหาวิธีการค้นหาเส้นทางตามข้อจำกัดต่างๆ ของเครือข่ายเซนเซอร์ไร้สายดังนี้

1. การจัดการพลังงาน (Energy Management)
   การจัดการพลังงานเป็นประเด็นที่สำคัญตัวหนึ่ง จากประเด็นของ ความน่าจะเป็นจากการสูญเสียเส้นทาง (Routing failure) และ ความน่าจะเป็นจากการสูญหายของเครือข่าย (Network failure) เป็นต้น การสูญหายของพลังงานจากการใช้ในการรับ และส่งข้อมูล รวมถึงการจัดการค้นหาเส้นทาง ดังนั้นเมื่อเซนเซอร์โนด มีข้อจำกัดในเรื่องนี้ จึงต้องหาวิธีการออกแบบโพรโทคอลการค้นหาเส้นทางให้สามารถใช้พลังงานได้อย่างมีประสิทธิภาพ เพื่อรักษาความคงอยู่ของเครือข่าย
   จากคุณลักษณะเฉพาะของเครือข่ายเซนเซอร์ไร้สายในเรื่องแบตเตอร์รี่ ความสามารถในการประมวลผล และการจัดการข้อมูล ซึ่งคุณสมบัติเหล่านี้ จึงจำเป็นต้องออกแบบโพรโทคอลการค้นหาเส้นทาง ที่ต้องสนใจต่อการใช้พลังงานให้มีประสิทธิภาพมากที่สุด
2. การจัดการโทโปโลยี (Topology Management)
   รูปแบบของโครงสร้างเครือข่ายมีหลายแบบ และต้องเลือกให้เหมาะสมกับงาน เพื่อจะเพิ่มสมรรถนะให้กับระบบสื่อสาร ยกตัวอย่างเช่น Star Topology เหมาะสมกับระบบเครือข่ายที่มีขนาดเล็ก รวมถึงระยะเวลาในการส่งข้อมูล Latency ต่ำ โดยมีโนดตัวกลางคอยจัดการการเข้าใช้ช่องสัญญาณ ส่วนกรณีของ Peer-to-Peer Topology เหมาะสมกับรูปแบบเครือข่ายที่มีขนาดใหญ่ และมีความซับซ้อนของโนดสูง แต่ไม่คำนึงถึงระยะเวลาในการส่งข้อมูล โดยที่การทำงานเป็นแบบหลายฮอบ

(Multihops) แต่ละโนดสามารถสื่อสารกันเอง ดังนั้นการเลือกรูปแบบของโทโปโลยี จะต้องดูจากงานที่จะใช้เป็นหลัก

3. การจัดการเครือข่าย (Network Management)
   การจัดการเส้นทางเป็นส่วนประกอบสำคัญต่อเครือข่ายเซนเซอร์ไร้สาย ที่ต้องการให้ข้อมูลสามารถส่งไปยังปลายทางได้ โดยที่ข้อมูลมีการสูญหายน้อย และเมื่อมีข้อมูลสูญหายเกิดขึ้น ก็สามารถที่จะจัดการกับข้อมูลเหล่านั้น โดยใช้ทรัพยากร อย่างเช่น พลังงานน้อยที่สุด ดังนั้นคุณลักษณะเฉพาะของเครือข่ายเซนเซอร์ไร้สาย ที่เกี่ยวกับการจัดการส่งข้อมูล สามารถสรุปจาก ความน่าจะเป็นที่จะเกิดปัญหาการสูญเสียเส้นทางการสื่อสาร (Link failure) ความน่าจะเป็นจากการสูญเสียเส้นทาง และความน่าจะเป็นจากการสูญหายของเครือข่าย

การออกแบบระบบสถาปัตยกรรม

ลักษณะของสถาปัตยกรรมที่จะพิจารณาในการออกแบบของเครือข่ายเซนเซอร์ไร้สายขึ้นอยู่กับงานประยุกต์ (Application) ความแตกต่างของสถาปัตยกรรม และจุดประสงค์ของการออกแบบ โดยเฉพาะอย่างยิ่งการออกแบบสถาปัตยกรรมนี้ จะมีผลต่อการเลือกใช้โพรโทคอลการค้นหาเส้นทาง เพื่อให้ได้ประสิทธิภาพของงานสูงสุด ในหัวข้อนี้จึงต้องระบุสิ่งที่จำเป็นต่อการพิจารณา การออกแบบสถาปัตยกรรมในเครือข่ายเซนเซอร์ไร้สายดังนี้

Network Dynamics

กระบวนการนี้จำเป็นต่อสภาวะของเครือข่ายแบบไร้สาย เพราะสถานะของเครือข่ายมีความไม่แน่นอนสูง เครือข่ายมีการเคลื่อนที่ตลอดเวลา จำเป็นต้องมีโพรโทคอลการค้นหาเส้นทางในการจัดการและตรวจสอบเส้นทางอยู่ตลอดเวลา เนื่องจากมีโอกาสที่ต้องทำการหาเส้นทางใหม่บ่อยครั้ง ดังนั้นการออกแบบสถาปัตยกรรมของเครือข่ายเซนเซอร์ไร้สาย จึงจำเป็นต้องการเลือกใช้โพรโทคอลในแต่ละระดับชั้นให้เหมาะสมกับงานที่จะใช้จริง โดยเฉพาะโพรโทคอลการค้นหาเส้นทาง สามารถเลือกใช้เส้นทางอื่น เมื่อเส้นทางสื่อสารเดิมไม่สามารถทำงานได้ หรือหากเกิดสภาวะที่โนด เกิดการสูญหายจากปัญหาของ Link failure โพรโทคอลระดับ Medium Access Control (MAC) ก็สามารถจัดการกับระบบให้ยังคงส่งข้อมูลไปยังปลายทางได้

Node Deployment

การกระจายตัวของโนดขึ้นอยู่กับความต้องการของงานประยุกต์ และต้องทำงานร่วมกับโพรโทคอลการค้นหาเส้นทาง ดังนั้นการจัดการตัวโนด มีได้ทั้งแบบการจัดการด้วยตัวเอง โดยการวางโนดด้วยการให้ผู้ใช้กำหนดตำแหน่งที่ชัดเจนและอีกแบบจะเป็นการจัดการแบบสุ่ม (Random) ซึ่งผลของการจัดการตัวโนดจะส่งผลต่อประสิทธิภาพของพลังงาน และสมรรถนะการทำงานของระบบ

Node Capabilities

ในเซนเซอร์โนด ต้องการให้มีการคำนวณ การสื่อสาร และพลังงาน ซึ่งทั้งสามตัวนี้ก็เป็นคีย์หลักต่อสมรรถนะของระบบ นอกจากคีย์หลักที่จะต้องออกแบบให้มีประสิทธิภาพ ความสามารถของเซนเซอร์โนด ก็จำเป็นเพื่อรองรับคีย์หลักเหล่านั้น การนำเซนเซอร์โนดมาใช้งาน จะต้องเลือกให้เหมาะสมกับงานประยุกต์ สำหรับการนำโนดไปเก็บข้อมูลต่างๆ เช่น อุณหภูมิ ความชื้น ความเร็วลม และอื่นๆ สามารถที่จะเก็บข้อมูลเป็นแบบเวลาจริง หรือแบบสุ่มก็ได้ ดังนั้นงานวิจัยทางด้านนี้ก็มีการพัฒนาโนด เพื่อรองรับกับงานประยุกต์ในอนาคตมากขึ้น

Energy Considerations

กระบวนการที่มีผลต่อพลังงานมาก จะมีการกระบวนการจัดการค้นหาเส้นทาง กระบวนการซ่อมแซมเส้นทาง และเส้นทางที่โนดต้นทางอยู่ห่างจากโนดปลายทาง ซึ่งกระบวนการดังกล่าวจะต้องการพลังงานสูง ดังนั้นในหัวข้อนี้ ถ้าหากสามารถจัดการระบบให้อยู่ในสภาวะปกติ ก็จะช่วยให้ใช้พลังงานน้อยลง หรือ การจัดการการกระจายตัวของโนดให้มีความยืดหยุ่นต่อสภาวะต่างๆ ในส่วนนี้จึงทำให้เกิดงานวิจัยจำนวนมากเพื่อรักษาให้เกิดสภาพแวดล้อมที่ระบบยังคงทำงานได้นานที่สุด และมีระบบเตือนให้สามารถจัดการระบบให้ได้ประสิทธิภาพสูงสุด

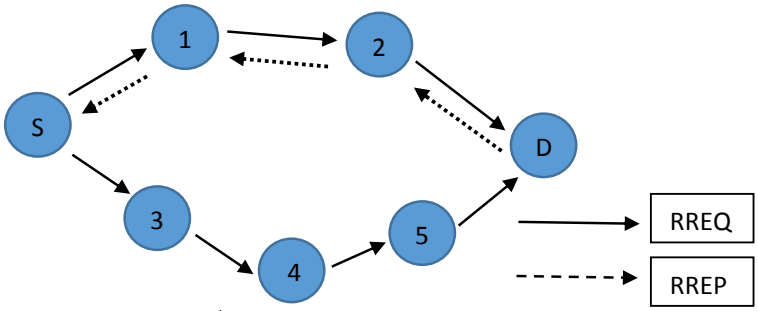
5.4 ตัวอย่างโพรโทคอลค้นหาเส้นทางที่น่าสนใจ

ในปัจจุบันโพรโทคอลที่นำมาใช้งานจริง และใช้ร่วมกับฮาร์ดแวร์จริง ได้แก่ โพรโทคอล Ad-hoc On-Demand Distance Vector (AODV) และโพรโทคอล LEACH ดังนั้นจึงขออธิบายการทำงานของโพรโทคอลที่น่าสนใจเพียง 2 ตัวนี้ในหนังสือเล่มนี้ สำหรับโพรโทคอลอื่นๆ และการศึกษาโพรโทคอลด้วยวิธีจำลองการทำงานบนโปรแกรม Network Simulator (NS-2) นั้นจะได้กล่าวและวิเคราะห์ไว้ในหนังสือเครือข่ายเซนเซอร์ไร้สายเล่มที่ 2

5.4.1 โพรโทคอล Ad-hoc On-Demand Distance Vector (AODV)

โพรโทคอลการค้นหาเส้นทาง AODV [29-30] จะทำการค้นหาเส้นทางเมื่อมีการร้องขอ โดยแบ่งออกเป็น 2 ส่วนคือ กระบวนการค้นหาเส้นทาง (Route Discovery) และ กระบวนการบำรุงรักษาเส้นทาง (Route Maintenance)

1. เริ่มต้นจาก เมื่อโนดต้นทางต้องการเส้นทางในการส่งข้อมูล โนดจะทำการตรวจสอบข้อมูลเส้นทางในตารางเส้นทางของตนเอง ในกรณีที่มีเส้นทาง ข้อมูลก็สามารถส่งออกไปได้ทันที แต่ในกรณีที่ไม่มีข้อมูลเส้นทางโนดจะเริ่มกระบวนการค้นหาเส้นทาง โดยส่งข้อความควบคุมที่เรียกว่า Route Request (RREQ) ไปยังโนดเพื่อนบ้าน เมื่อโนดเพื่อนบ้านได้รับข้อความควบคุม RREQ จะทำการตรวจสอบข้อมูลในตารางเส้นทาง กรณีที่มีข้อมูลเส้นทางของโนดปลายทาง โนดจะทำการส่งข้อความควบคุม Route Reply (RREP) กลับไปยังโนดต้นทาง แต่หากไม่ใช่โนดปลายทางโนดก็จะทำการส่งข้อความควบคุม RREQ ต่อไปจนกระทั่งถึงโนดปลายทาง โนดปลายทางจะตอบกลับด้วยข้อความควบคุม RREP โนดต้นทางจะทำการพิจารณาข้อมูลจาก เลขลำดับปลายทาง (Destination Sequence number) เพื่อความใหม่ของข้อมูล (Freshness) และจำนวนระยะห่างระหว่างต้นทางและปลายทาง (Hop count) เพื่อเลือกเส้นทางที่สั้นที่สุด โดยยกตัวอย่างการทำงานของโพรโทคอลการค้นหาเส้นทาง AODV ตามภาพประกอบที่ 5.9



ภาพประกอบที่ 5.9 การค้นหาเส้นทางของโพรโทคอล AODV

จากภาพประกอบที่ 5.9 เป็นต้นอย่างของการค้นหาเส้นทางของโพรโทคอล AODV โดยโนด S เป็นโนดต้นทาง ต้องการทำการส่งข้อมูลไปยังโนด D ซึ่งเป็นโนดปลายทาง โนด S จะทำการกระจาย (Broadcast) ข้อความควบคุม RREQ ไปยังโนดเพื่อนบ้าน คือ โนดหมายเลข 1 และโนด 3 เพื่อทำการส่งข้อความ RREQ ไปยังโนดถัดไป เมื่อโนด D ซึ่งเป็นโนดปลายทาง ได้รับข้อความควบคุม RREQ จะทำการตอบกลับด้วยข้อความควบคุม RREP กลับไปแบบโดยตรง (Unicast) ผ่านเส้นทางที่สั้นที่สุดในกรณีนี้คือเส้นทางของโนดหมายเลข 1 และ 2 เมื่อโนด

ได้รับข้อความควบคุม RREP จะทำการบันทึกข้อมูลเส้นทางลงในตารางเส้นทาง และสามารถส่งข้อมูลไปยังปลายทาง จนกระทั้งเส้นทางในการส่งข้อมูลจะเสียหาย

2. กระบวนการบำรุงรักษาเส้นทาง จะทำการตรวจสอบเส้นทางในการส่งข้อมูล กรณีที่โนดมีการเคลื่อนที่หรือโนดหายไป (เนื่องจากพลังงานหมด หรืออยู่ในสภาวะหลับ (Sleep)) ซึ่งเป็นสาเหตุทำให้เส้นทางในการส่งข้อมูลเสียหาย ดังนั้นโพรโทคอลการค้นหาเส้นทาง AODV จึงใช้ข้อความควบคุม RREP ที่กำหนดให้มีระยะการส่ง 1 Hop ซึ่งอาจเรียกว่าข้อความควบคุม Hello เพื่อใช้ในการตรวจสอบสถานะของโนดเพื่อนบ้าน ในกรณีที่โนดเพื่อนบ้านสูญหายไป จึงทำให้ไม่สามารถส่งข้อมูลไปยังปลายทางได้ ดังนั้นเมื่อโนดพบความเสียหายจะทำการตรวจสอบว่าเกิดความเสียหายใกล้ต้นทางหรือใกล้ปลายทาง ในกรณีที่โนดอยู่ใกล้ปลายทางโนดจะเริ่มกระบวนการซ่อมแซมเฉพาะที่ ซึ่งเรียกว่า Local Repair โดยพยายามที่จะซ่อมแซมเส้นทางด้วยกระบวนการค้นหาเส้นทางไปยังโนดปลายทาง แต่อาจจะไม่สามารถค้นหาเส้นทางได้ ถ้าหากเกิดเหตุการณ์นี้ จะมีการส่งข้อความควบคุม Route Error (RERR) ไปยังโนดต้นทางเพื่อเริ่มกระบวนการค้นหาเส้นทางใหม่อีกครั้ง

5.4.2 โพรโทคอล Low-Energy Adaptive Clustering Hierarchy (LEACH)

LEACH [15] เป็นโพรโทคอลที่ได้รับความสนใจมากตัวหนึ่งในกลุ่มโพรโทคอลแบบ ลำดับชั้นสำหรับเครือข่ายเซนเซอร์ไร้สาย แนวความคิดหลักของโพรโทคอลตัวนี้คือ แบ่งตัวโนดในเครือข่ายออกเป็นกลุ่มๆ โดยพิจารณาจากความเข้มของสัญญาณเป็นหลักในการแบ่งกลุ่ม และให้ตัวโนดหัวหน้าเป็นทางเชื่อมต่อไปยังตัวสถานีฐาน ด้วยวิธีการนี้จะสามารถประหยัดพลังงานจากการส่งข้อมูลได้เพราะการส่งข้อมูลถึงสถานีฐานจะมีเพียงแค่ตัวโนดที่เป็นหัวหน้ากลุ่มเท่านั้น โนดอื่นๆไม่ต้องส่งข้อมูลมายังสถานีฐานโดยตรงแต่จะใช้โนดหัวหน้าเป็นตัวส่งต่อข้อมูลไปยังสถานีฐานแทน ซึ่งจำนวนของตัวโนดที่เป็นหัวหน้าจะมีประมาณ 5% ของตัวโนดทั้งหมดภายในเครือข่าย

โนดหัวหน้าแต่ละตัวจะมีภาระหน้าที่ในการรวบรวมข้อมูลทั้งหมดภายในกลุ่มและประมวลผล บีบอัดก่อนส่งต่อไปยังสถานีฐาน เนื่องจากโนดหัวหน้าจะมีภาระหน้าที่เยอะกว่าตัวโนดอื่นๆทำให้มีการใช้พลังงานค่อนข้างสูงกว่า โนดหัวหน้าในเครือข่ายจึงจะถูกสุ่มเปลี่ยนตามเวลาที่กำหนดเพื่อกระจายการใช้พลังงานของตัวโนดให้สมดุลภายในเครือข่าย โดยการตัดสินใจดังกล่าวว่าโนดตัวใดจะเป็นโนดหัวหน้า เริ่มแรกโนดแต่ละตัวจะกระทำการสุ่มเลือกหมายเลขระหว่าง 0 กับ 1 ขึ้นมาและโนดตัวใดสุ่มเลือกตัวเลขได้ค่าที่มีค่าต่ำกว่าค่าเกณฑ์ จะเป็นโนดหัวหน้า โดยค่าเกณฑ์จะคำนวณได้จากสมการที่ (1) ดังนี้

$$T(n) = \begin{cases} \frac{P}{1-P*\left(rmod\frac{1}{P}\right)} & if n \in G, \\ 0 & otherwise \end{cases} \qquad (5.1)\ [15]$$

โดยที่ P เป็นจำนวนเปอร์เซ็นต์ของตัวโนดหัวหน้าที่ต้องการในเครือข่ายต่อจำนวนโนดทั้งหมดในเครือข่าย r เป็นหมายเลขของรอบ n เป็นหมายเลขของโนด G เป็นชุดของตัวโนดที่ยังไม่เคยทำหน้าที่เป็นโนดหัวหน้าใน 1/P รอบ จากสมการที่ (5.1) จะเห็นว่าโนดทุกตัวในเครือข่ายจะต้องทำหน้าโนดหัวหน้า 1 รอบภายใน 1/P รอบ เมื่อโนดตัวใดทำหน้าที่เป็นโนดหัวหน้าแล้ว ในรอบที่เหลือภายใน 1/P รอบจะมีค่าความน่าจะเป็นเท่ากับ 0 หมายความว่าโนดดังกล่าวจะไม่ได้รับเลือกเป็นโนดหัวหน้าอีก เริ่มต้นโนดแต่ละตัวจะทำการสุ่มค่าตัวเลขขึ้นมาหนึ่งค่า โดยค่าดังกล่าวจะมีค่าระหว่าง 0 ถึง 1 และนำค่าดังกล่าวมาเทียบกับค่าเกณฑ์ที่ได้ในสมการที่ (5.1) หากว่าค่าที่สุ่มได้มีค่าต่ำกว่าค่าเกณฑ์ตัวโนดที่สุ่มค่านั้นจะทำหน้าที่เป็นโนดหัวหน้า

สรุปท้ายบท

ในบทนี้ได้อธิบายการทำงานในแต่ละระดับชั้นของเครือข่ายเซนเซอร์ไร้สายตั้งแต่ชั้นกายภาพ (Physical) ระดับชั้น MAC ระดับชั้น Networks มาตรฐาน IEEE802.15.4 มาตรฐาน Zigbee รวมทั้งได้อธิบายการทำงานเบื้องต้นของโพรโทคอลในชั้น MAC ได้แก่ S-MAC, WiseMAC, DSMAC และ MAC ตามมาตรฐาน IEEE802.15.4 และสุดท้ายได้อธิบายการทำงานเบื้องต้นของโพรโทคอลค้นหาเส้นทางที่ได้รับความนิยมเพื่อใช้งานจริงคือ โพรโทคอล AODV และ LEACH

เอกสารอ้างอิง

1. H. Riksfjord, O.T.Haung and J.M.Hovem, “Underwater Acoustic Networks – Survey on Communication Challenges with Transmission Simulations,” *International Conference on Sensor Technologies and Applications*, pp. 300-305, 2009.
2. A. Gkikopouli, G.Nikolakopoulous and S. Manesis, “A Survey on Underwater Wireless Sensor Networks and applications,” *20th Mediterranean Conference on Control and Automation,* pp.1147-1154, 2012.
3. K. Chen, M. Ma and et.al, “A Survey on MAC protocols for Underwater Wireless Sensor Networks,” *IEEE Communications Survey & Tutorials,* pp.1-15, 2014.
4. A. Davis and Hwa Chang, “Underwater Wireless Sensor Networks,” *Oceans*, pp.1-5, 2012.
5. IEEE standard for information technology – telecommunications and information exchange between systems – local and metropolitan area networks – specific requirement part 15.4: wireless medium access control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area networks (WPANS). *IEEE Std. 802.15.4a-2007 (Amendment to IEEE Std. 802.15.4-2006)*, pp. 1–203, 2007.
6. J. Polastre, J. Hill, and D. Culler, “Versatile low power media access for wireless sensor networks,” *Proceedings of SenSys’04*, pp. 95–107, 2004.

7. W. Ye, J. Heidemann, and D. Estrin “An energy-efficient MAC protocol for wireless sensor networks,” *Proceedings of IEEE INFOCOM’02*, volume 3, pp. 1567–1576, 2002.
8. A. El-Hoiydi and J.-D. Decotignie, “WiseMAC: an ultra-low power MAC protocol for the downlink of infrastructure wireless sensor networks,” *Proceedings of the International Symposium on Computers and Communications (ISCC’04)*, pp. 244–251, 2004.
9. Cheng Yin, Ya Li and et.al, “DSMAC: An energy-efficient MAC protocol in event-driven sensor networks,” *International Conference on Advanced Computer Control*, pp. 422-425, 2010.
10. J. Li and G. Y. Lazarou, “A bit-map-assisted energy-efficient MAC scheme for wireless sensor networks,” *Proceedings of ACM IPSN’04*, pp. 55–60, 2004.
10. V. Rajendran, K. Obraczka, and J. J. Garcia-Luna-Aceves, “Energy-efficient, collision-free medium access control for wireless sensor networks,” *Proceedings of ACM SenSys’03*, 2003.
11. I. Rhee, A. Warrier, M. Aia, and J. Min, “Z-MAC: a hybrid MAC for wireless sensor networks,” *Proceedings of ACM SenSys’05*, pp. 90–101, 2005.
12. J. Galtier, “Analysis of the slotted non-persistent CSMA protocol with poissonian packet size using a semi-Markov graph representation,” *International Conference on Transparent Optical Networks,* pp.258-262, 2006.
13. J. Kulik, W.Heinzelman and H. Balakrishman, “Negotiation-based protocols for disseminating information in wireless sensor networks,” *Wireless Sensor Networks,* pp. 168-185, 2002.
14. W.B. Heizelman, A.P. Chandrakasan, and H. Balakrishnan, “An Application-Specific Protocol Architecture for Wireless Microsensor Networks,” *IEEE Transactions on Wireless Communications*, vol 1, No. 4, pp. 660-670, 2002.
15. K. Akkaya and M. Younis, “*A survey on routing protocols for wireless sensor networks*,” *Ad Hoc Networks*, vol.3, no.3, pp.325–349, 2005.
16. A.Manjeshwar and D.P. Agrawal, “TEEN: A routing protocol for enhanced efficiency in wireless sensor networks”, *International Conference on Parallel and Distributed Processing Symposium,* pp.2009-2015, 2001.
17. S. Lindsey and C.S.Raghavendra, “PEGASIS: Power-efficient gathering in sensor information systems,” *IEEE Aerospace Conference*, pp.1125-1130, 2002.
18. J.Y. Yu and P.H.J. Chong, “A survey of clustering schemes for mobile ad hoc networks,” *IEEE Communication Survey and Tutorial*, vol.7, No.1, pp. 32-48, 2005.

19. Sakuna Charoenpanyasak, “Real Multiroute System (RMS) for Mobile Ad hoc Networks: A Cross layer Approach,” *ACM First International Workshop on Ad Hoc & Ubiquitous Computing (AUC-2009)*, Kuala Lumpur, Malaysia, December 14-16, 2009.
20. Amel B and Zoulikha M.M., “Routing technique with cross-layer approach in Ad hoc network,” 2nd International *Conference on the Applications of Digital Information and Web Technologies (ICADIWT '09),* 2009.
21. Lixin Li and Huisheng Zhang, “Research on Cross-Layer Design for MANET,” *Third International Symposium on Intelligent Information Technology Application (IITA 2009)*, vol. 2, 2009.

22. Heping Wang, Xiaobo Zhang, Nait-Abdesselam, F and Khokhar, A, “Cross-Layer Optimized MAC to Support Multihop QoS Routing for Wireless Sensor Networks,” *IEEE Transactions on Vehicular Technology*, vol. 59, issue 5, 2010.
23. Jamal N. Al-Karaki, Ahmed E. Kamal, “Routing Techniques in Wireless sensor networks: A Survey,” *IEEE Wireless Communications*, 2004.
24. Shijin Dai, Xiaorong Jing and Lemin Li, “Research and analysis on routing protocols for wireless sensor networks,” *International Conference on Communications, Circuits and Systems*, vol. 1, May 2005.
25. Baghyalakshmi D, Ebenezer J and Satyamurty S.A.V., “Low latency and energy efficient routing protocols for wireless sensor networks,” *International Conference on Wireless Communication and Sensor Computing (ICWCSC)*, 2010.
26. Hongbin Chen, Tse, C.K. and Jiuchao Feng, “Impact of Topology on Performance and Energy Efficiency in Wireless Sensor Networks for Source Extraction,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 20, issue 6, 2009.
27. Kemal Akkaya and Mohamed Younis, “A survey on routing protocols for wireless sensor networks,” *Journal of Ad Hoc Networks*, vol. 3, pp. 325-349, 2005.
28. Nikolaos A. Pantazis, Stefanos A. Nikolidakis and Dimitrios D. Vergados, “Energy-Efficient Routing Protocols in Wireless Sensor Networks: A Survey,” *IEEE COMMUNICATIONS SURVEYS & TUTORIALS*, vol. 15, No. 2, 2013.
29. Charles E. Perkins, Elisabeth M. Belding-Royer and Samir R. Das, “Ad hoc On-Demand Distance Vector (AODV) Routing,” *RFC 3561*, July 2003.
30. Che-Aron Z, Al-Khateeb W.F.M and Anwar F, “The Enhanced Fault-Tolerant AODV Routing Protocol for Wireless Sensor Network,” *Second International Conference on Computer Research and Development*, 2010.

Lab Sheet ภาคที่ 1

แนะนำการสร้างเครือข่ายเซนเซอร์ไร้สายด้วย XBee ภาค 1

(Introduction to Building WSNs using XBee Part I)

จุดประสงค์การเรียนรู้

แนะนำและเรียนรู้ XBee พร้อมทำการติดตั้งโปรแกรมที่เกี่ยวข้องให้สามารถทำงานได้อย่างง่าย

เมื่อได้เรียนรู้ทฤษฎีที่จำเป็นและเกี่ยวข้องกับเครือข่ายเซนเซอร์ไร้สายแล้ว ในบทนี้และบทถัดไปจะได้แนะนำการใช้งานและนำเครือข่ายเซนเซอร์ไร้สายไปประยุกต์ใช้งาน ในที่นี้จะเลือกใช้งาน XBee Platform ซึ่งรองรับการทำงานโพรโทคอล ZigBee สำหรับบทนี้เป็นการแนะนำวิธีการเลือกหาอุปกรณ์และซอฟต์แวร์สำหรับความรู้เบื้องต้นของ XBee การตั้งค่าก่อนเริ่มใช้งาน การนำ XBee ไปใช้งานอย่างง่าย และการใช้งาน XBee ร่วมกับไมโครคอนโทรลเลอร์ เนื้อหาส่วนหนึ่งในบทนี้ได้ทำการสรุปมาจากหนังสือ ***Building Wireless Sensor Networks, O'Reilly, ของ Robert Faludi ปี 2011*** ส่วนเนื้อหาที่เหลือได้รวบรวมจากเว็ปไซต์ของบริษัท Digi และประสบการณ์ของนักพัฒนาที่ใช้ XBee สำหรับในบทถัดไปหรือบทสุดท้ายจะทำการยกตัวอย่าง Project ที่นำ XBee และไมโครคอนโทรลเลอร์ไปใช้งาน พร้อมทั้งแนะนำการใช้งาน Programmable XBee เบื้องต้น

L1.1 การเตรียมพร้อม XBee และซอฟต์แวร์ที่เกี่ยวข้อง

L1.1.1 การเลือกใช้ XBee

ในท้องตลาดมี XBee ให้เลือกหลากหลายรุ่น จนอาจจะทำให้สับสนได้ ดังนั้นจึงขอสรุปรุ่นของ XBee ไว้คร่าวๆก่อนที่จะเตรียมพร้อมซอฟต์แวร์ XBee เป็นโนด Platform หนึ่งที่สามารถหาซื้อได้ในประเทศ ผลิตโดย

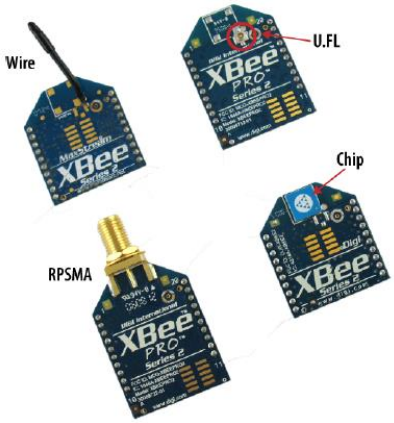
บริษัท Digi จนถึงปัจจุบัน XBee มีด้วยกัน 2 รุ่นคือ XBee Series 1 และ XBee Series 2 ในหนังสือนี้ขอกล่าวถึงเฉพาะ Series 2 เพราะเป็นรุ่นที่มีใช้อยู่ในปัจจุบัน โดยข้อแตกต่างคร่าวๆระหว่าง Series 1 และ Series 2 ได้แก่ 1) Series 1 จะใช้ Standard IEEE 802.15.4 สำหรับการเชื่อมต่อแบบ Point-to-point หรือแบบ Star 2) Series 2 ได้เพิ่ม Zigbee mesh firmware มีการใช้พลังงานต่ำกว่ารุ่นเก่า แต่การสร้างการเชื่อมต่อจะซับซ้อนและยุ่งยากกว่าแบบแรก

รุ่น XBee series 2 จะมี 3 แบบด้วยกันคือ ZNet 2.5, ZB และ S2B โดยรุ่น ZNet2.5 ถือเป็นรุ่นแรกในการพัฒนา XBee series 2 ของบริษัท Digi ในปี 2006 อุปกรณ์ภายในไม่แตกต่างจากรุ่น ZB สามารถทำการ update ZB firmware ได้, ถัดมาคือรุ่น ZB เป็นโครงสร้างของ ZigBee Pro ส่วนรุ่นในปัจจุบันเป็น S2B ที่ทำการปรับปรุงเรื่องของกำลังในการส่งข้อมูล ประหยัดกำลังงาน และมีประสิทธิภาพมากขึ้น

าอากาศของนอกจากรุ่นแล้วจะมีลักษณะของเส XBee ให้เลือกได้ 4 แบบคือ SI: RPSMA, WI: wire whip, UI: U.FL และ CI: on board ceramic chip ดังภาพประกอบที่ L1.1 ซึ่งอักษรย่อเหล่านี้จะปรากฏอยู่ด้านหลังของ XBee นอกจากตัวย่อเสาอากาศเหล่านี้แล้วยังมีอักษรย่อดังนี้

A	802.15.4 (Series 1)
DM	Digimesh (Series 1)
Z7	Zigbee Series 2
BZ7	Zigbee บน Series 2B

นอกจากที่จะต้องจัดหา XBee แล้วจำเป็นที่จะต้องมีฐานรอง XBee เนื่องจาก Pitch ของ Pin XBee มีขนาดเล็กอยู่ที่ 2 mm ไม่สามารถนำไปใช้งานได้ทันที โดยผู้พัฒนาสามารถทำฐานรองได้เอง หรืออาจจะสามารถหาซื้อ XBee Breakout board ซึ่งขยายฐานรองเป็นขนาด 0.1” และ Dongle ที่ไว้สำหรับเชื่อมต่อ XBee กับคอมพิวเตอร์ผ่านทางพอร์ต USB ดังภาพประกอบที่ L1.2



ภาพประกอบที่ L1.1 ภาพแสดงลักษณะเสาอากาศของ XBee

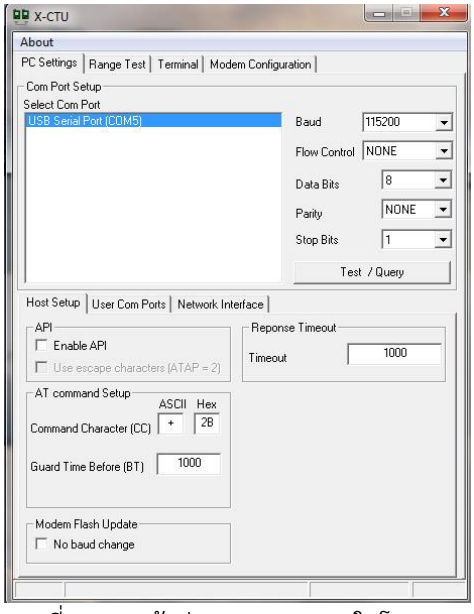
ภาพประกอบที่ L1.2 XBee Breakout และ Dongle ของบริษัท Thaieasy Elec.

L1.1.2 การเตรียมซอฟต์แวร์ที่เกี่ยวข้อง

บริษัท Digi จะให้ซอฟต์แวร์ที่ใช้กับ XBee เพื่อติดต่อ ตั้งค่าและโปรแกรมลงบน XBee ผ่านทาง USB to Serial ซอฟต์แวร์ตัวนี้ชื่อ X-CTU ดังภาพประกอบที่ L1.3 ผู้ใช้สามารถหาดาวน์โหลดได้จากเว็ปไซต์ของบริษัท Digi ได้โดยตรง เมื่อเปิดโปรแกรม X-CTU จะพบว่าโปรแกรมประกอบด้วย 4 ส่วนด้วยกัน ได้แก่ 1) PC Setting 2) Range Test 3) Terminal และ 4) Modem Configuration

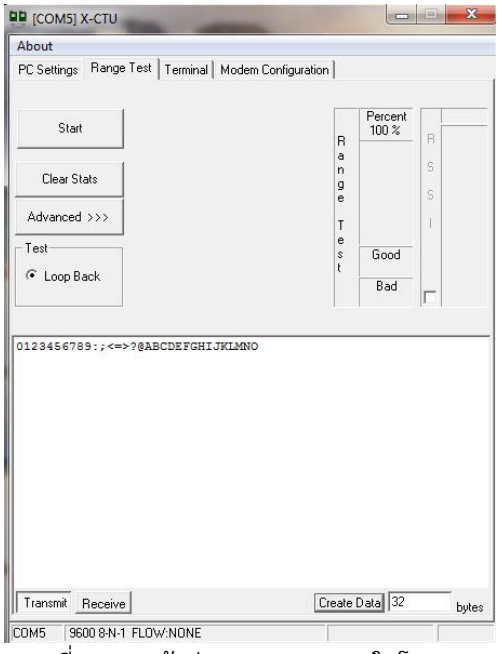
ภาพประกอบที่ L1.3 About X-CTU

1) PC Settings มีหน้าต่างดังภาพประกอบที่ L1.4 ซึ่งให้ผู้ใช้เลือก Com Port และตั้งค่าอัตราการรับส่งข้อมูล



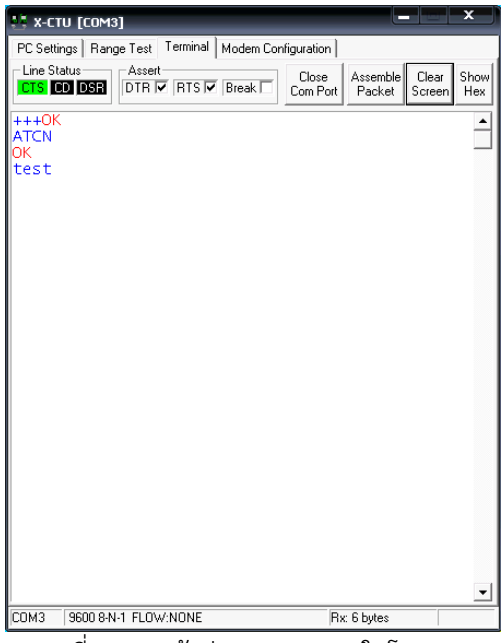
ภาพประกอบที่ L1.4 หน้าต่าง PC Setting ในโปรแกรม X-CTU

2) Range Test ใช้ในการทดสอบระยะทางในการส่งข้อมูลระหว่าง XBee ดังภาพประกอบที่ L1.5



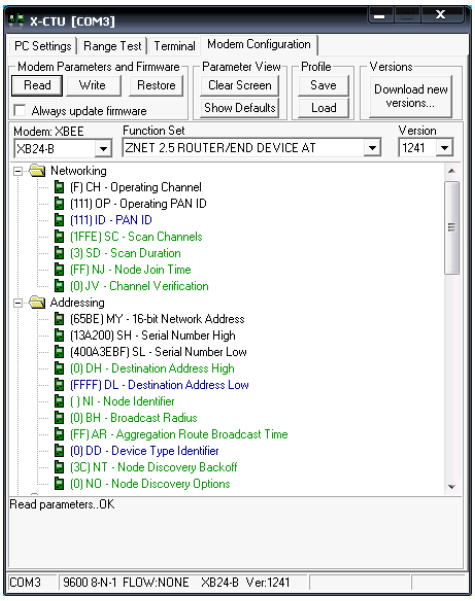
ภาพประกอบที่ L1.5 หน้าต่าง Range Test ในโปรแกรม X-CTU

3) Terminal เป็นหน้าต่างที่ใช้ในการติดต่อระหว่างคอมพิวเตอร์กับ XBee ด้วยคำสั่งในรูปแบบ AT ดังภาพประกอบที่ L1.6 โดยสีฟ้าเป็นข้อมูลที่ส่งออกไปยัง XBee และสีแดงเป็นข้อมูลที่รับเข้ามา



ภาพประกอบที่ L1.6 หน้าต่าง Terminal ในโปรแกรม X-CTU

4) Modem Configuration เป็นส่วนกำหนดการตั้งค่าค่างๆ ของ XBee รวมทั้งการอัพเดต Firmware ดังภาพประกอบที่ L1.7 ค่าของ Firmware ที่ถูกอ่านขึ้นมาจะมีแสดงอยู่ด้วยกัน 3 สี คือ 1) สีดำไม่สามารถตั้งค่าได้ หรือถือได้ว่า Read-only 2) สีฟ้า เป็นสีที่ผู้ใช้จะต้องเป็นคนกำหนด 3) สีเขียวเป็นค่า Default



ภาพประกอบที่ L1.7 หน้าต่าง Modem Configuration ในโปรแกรม X-CTU

L1.2 แนะนำมาตรฐาน ZigBee

โดยทั่วไปเรามักจะสับสนระหว่าง XBee กับ ZigBee ซึ่งทั้งคู่ไม่เหมือนกัน ตรงที่ว่า ZigBee คือโพรโทคอลมาตรฐานสำหรับการสื่อสารบน IEEE 802.15.4 ในขณะที่ XBee คือชื่อของอุปกรณ์โนดสื่อสารที่มีทั้งแบบ IEEE 802.15.4 แบบ ZigBee หรือแม้แต่ Wifi ดังนั้นอุปกรณ์โนดสื่อสารอื่นใดที่ใช้มาตรฐาน ZigBee ก็สามารถถูกนำมาประยุกต์ใช้กับเครือข่ายเซนเซอร์ไร้สายได้ทั้งสิ้น แต่อาจจะต้องผ่านกระบวนการที่ซับซ้อนหากต้องการให้อุปกรณ์สื่อสารที่มาจากต่างยี่ห้อคุยกันได้

ZigBee นั้นเป็นกลุ่มก้อนของ Layer ที่ถูกกำหนดและสร้างอยู่บนมาตรฐาน IEEE 802.15.4 ซึ่งจุดสำคัญของการเพิ่มกลุ่ม Layer ของ ZigBee เข้าไปก็เพื่อการทำ 1) Routing เป็นโพรโทคอลที่จะช่วยค้นหาเส้นทางให้โนดสามารถส่งข้อความจากต้นทางไปยังปลายทางได้ 2) Ad hoc network เป็นกระบวนการอัตโนมัติที่จะช่วยให้โนดสามารถสร้างเครือข่ายขึ้นมาได้เอง 3) Self-healing mesh เป็นกระบวนการที่จะช่วยซ่อมแซมเครือข่ายใน

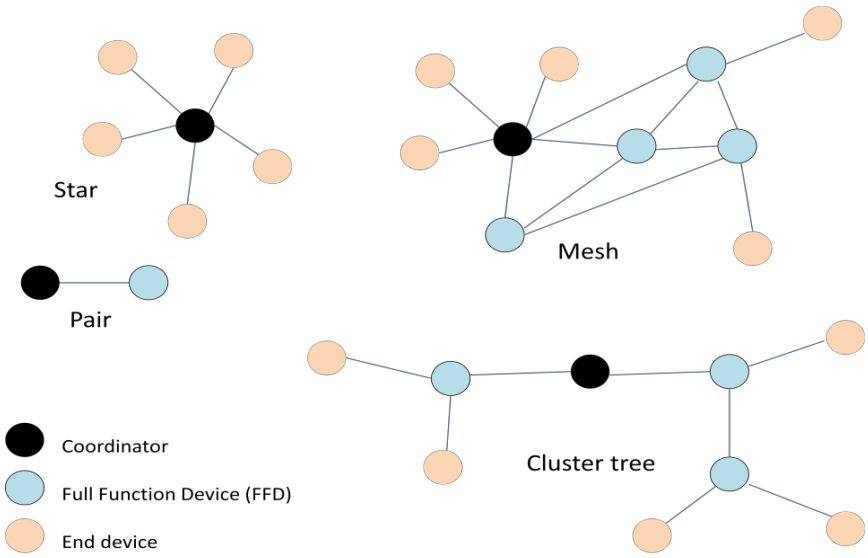
กรณีที่สัญญาณคลื่นวิทยุของโนดบางตัวหลุดออกไปจากเครือข่าย การทำ Self-healing นี้จะทำให้เครือข่ายถูกซ่อมแซมหรือปรับโครงสร้างใหม่ รวมทั้งการซ่อมแซมเส้นทางที่ได้รับความเสียหายด้วย

โนดหรือผู้เล่นในเครือข่ายเซนเซอร์ไร้สายแบบ ZigBee จะมีอยู่ 3 แบบเท่านั้นคือ

1) Coordinator ซึ่งจำเป็นที่จะต้องมีอยู่ในทุกรูปแบบของโครงสร้างเครือข่าย และมีเพียง 1 ตัวต่อ 1 เครือข่ายเท่านั้น ทำหน้าที่เป็นคนจัดการสร้างเครือข่าย, Address หรือหน้าที่อื่นๆที่เกี่ยวข้องเช่น เรื่องความปลอดภัยของข้อมูล (Security)
2) Router โนดชนิดนี้มีรูปแบบการทำงานของ ZigBee เต็มรูปแบบ สามารถเข้าสู่เครือข่ายที่มีอยู่เดิม ส่งข้อมูล รับข้อมูล มีข้อมูลเส้นทาง Router ทำหน้าที่เหมือน Messenger ช่วยให้อุปกรณ์โนดอื่นๆที่อยู่ไกลจาก Coordinator สามารถสื่อสารกันได้ ดังนั้นโนดชนิด Router นี้จำเป็นที่จะต้องมีแหล่งจ่ายพลังงานอยู่ตลอดเวลาและสามารถมี Router ได้มากกว่า 1 ตัว ไม่เช่นนั้นอุปกรณ์อื่นๆมีโอกาสที่จะถูกตัดขาดจาก Coordinator ได้
3) End device เป็นโนดที่ทำหน้าที่เชื่อมต่อกับเซนเซอร์ แล้วส่งข้อมูลที่ได้จากเซนเซอร์ไปยังโนด Router นอกจากนั้นโนดชนิดนี้สามารถหลับ (Sleep หรือ Idle) เพื่อประหยัดพลังงานให้มากที่สุด เนื่องจากโนดชนิด End device นี้จะใช้พลังงานจากแบตเตอร์รี่

L1.2.1 รูปแบบเครือข่ายที่มีใช้งานใน ZigBee

รูปแบบการเชื่อมต่อเป็นเครือข่ายเมื่อมีการนำ ZigBee ไปใช้งานมีหลากหลายรูปแบบดังภาพประกอบที่ L1.8 เช่น 1) การเชื่อมแบบจุดต่อจุด (Pair) ระหว่างโนด 2 ตัวซึ่งจะต้องกำหนด 1 โนดให้ทำหน้าที่เป็น Coordinator 2) การเชื่อมต่อแบบ Star ถือเป็นเครือข่ายชนิดที่ง่ายที่สุด โนด End device ทุกตัวจะส่งข้อมูลเข้าหา Coordinator เท่านั้น และโนด End device จะไม่ส่งข้อมูลถึงกันโดยตรง 3) การเชื่อมต่อแบบ Mesh จะมี Coordinator 1 ตัว โดยมีโนด Router อยู่หลายตัวเชื่อมต่อถึงกันในทุกทิศทางเพื่อเชื่อมต่อกับ Coordinator ให้ได้ โดยโนดที่จะรับข้อมูลจากเซนเซอร์หรือ End device จะคุยกับโนด Router เพื่อให้ช่วยส่งไปยัง Coordinator หรือสามารถเชื่อมต่อตรงกับ Coordinator ได้และแบบสุดท้าย 4) การเชื่อมต่อแบบ Cluster จะใช้โนด Router เชื่อมต่อเส้นทางเป็นเสมือน Backbone ของเครือข่าย โครงสร้างไม่แตกต่างจากการเชื่อมต่อแบบ Mesh เพียงแต่ Router ไม่ได้เชื่อมต่อกันหลากหลายเส้นทาง



ภาพประกอบที่ L1.8 Network Topologies ของ ZigBee

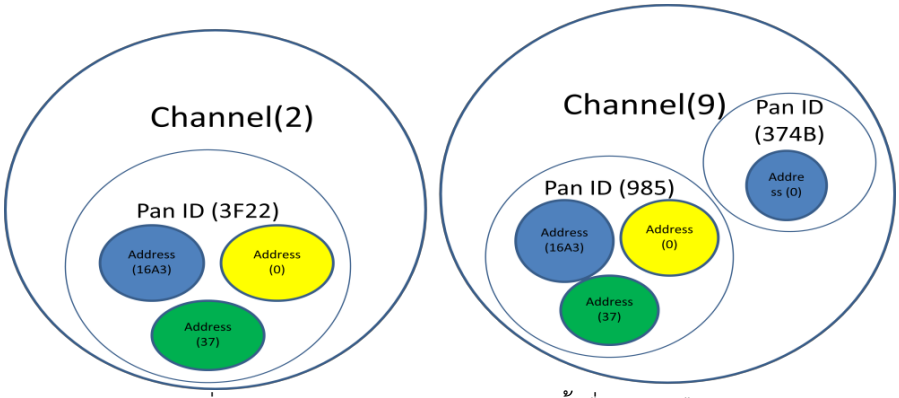
L1.2.2 Addressing

การมีหมายเลขระบุตัวตนของโนดทำให้ทราบว่าโนดใดต้องการส่งข้อมูลและต้องการส่งข้อมูลไปให้โนดใดในเครือข่าย สำหรับในมาตรฐานของ ZigBee จะสามารถเลือกใช้ได้ 3 แบบดังตารางที่ L1.1 แบบแรกคือการใช้หมายเลขของภาครับส่งคลื่นวิทยุจะมีความเป็นเอกลักษณ์โดยใช้จำนวน 64 บิตในการกำหนดหมายเลข นั่นหมายความว่าจะมีหมายเลขได้จำนวนมหาศาลทำให้อุปกรณ์ ZigBee แต่ละตัวจะมี Serial number นี้ไม่เหมือนกัน แต่ถ้าหากผู้ใช้ต้องการให้โนด Coordinator ในเครือข่ายเป็นผู้กำหนดหมายเลขระบุตัวเองของโนดกันเอง ก็จะสามารถกำหนดได้ 16 บิต และรูปแบบสุดท้ายของการกำหนดหมายเลขหรือการระบุตัวตนของโนดใน ZigBee สามารถทำได้โดยการกำหนดเป็นชื่อ ซึ่งก็จะสามารถทำให้โนดไม่ซ้ำกันได้ แต่แน่นอนว่าไม่สามารถรับประกันได้ว่า ผู้พัฒนาจากคนละที่จะกำหนดชื่อไม่ตรงกันหรือไม่

ตารางที่ L1.1 รูปแบบของการกำหนดหมายเลขระบุตัวตน

รูปแบบของการกำหนดหมายเลขระบุตัวตน	**ตัวอย่างและความหมาย**
64 บิต	0011A299403F0680
16 บิต	14F7
ตั้งชื่อ (Node identifier)	COE-WSN

ในเครือข่าย ZigBee เราสามารถสร้างชื่อของเครือข่ายขึ้นเองได้ เราเรียกว่า ***PAN Address*** เปรียบเสมือนการตั้งชื่อเมืองและมีโนดเป็นประชากรของเมืองที่สร้างขึ้น แต่ชื่อที่สร้างขึ้นในเครือข่าย ZigBee จะใช้เป็นตัวเลขไม่ใช่ตัวอักษร ทำให้สามารถมีชื่อของเครือข่ายเองได้ทั้งหมด 65,536 ชื่อ นอกจากการกำหนดหมายเลขระบุตัวตนแล้วยังสามารถมีช่องทางของการแบ่งกลุ่มของเครือข่ายได้ด้วยการแยกช่องสัญญาณ (Channel) การผสมผสานระหว่างการใช้หมายเลขระบุตัวตน การตั้งค่า PAN address และการแยกช่องสัญญาณจะทำให้สามารถสร้างเครือข่าย ZigBee ได้โดยไม่ซ้ำกันดังตัวอย่างในภาพประกอบที่ L1.9



ภาพประกอบที่ L1.9 แนวทางการผสมผสานการตั้งชื่อของเครือข่าย ZigBee

L1.2.3 โหมดการทำงานของ XBee

L1.2.3.1 การทำงานในโหมด AT Command

โหมดการทำงานในแบบแรกของ XBee เรียกว่า AT สามารถตั้งค่าภาครับส่งวิทยุได้ 2 รูปแบบคือ Transparent และ Command สำหรับรูปแบบ AT แบบ Transparent เป็นการสั่งให้ส่งข้อมูลไปยังโนดปลายทาง (Send *through)* เมื่อโนดปลายทางรับข้อมูลก็จะพ่นออกทางพอร์ตอนุกรมทันที ส่วนรูปแบบ Command สำหรับในกรณีที่ไม่ต้องการส่งข้อมูล แต่ต้องการคุยกับภาครับส่งวิทยุ (Talk to) ในแบบนี้ภาครับส่งวิทยุจะต้องหยุดเพื่อรับฟังคำสั่งและทำสิ่งที่ได้รับคำสั่งนั้นมา โดยปกติแล้ว XBee ที่อยู่ในโหมด AT จะถูกตั้งค่าให้เป็นแบบ Transparent ถ้าต้องการเปลี่ยนเข้าสู่รูปแบบการทำงานแบบ Command จะต้องใช้คำสั่ง “+++” แต่ถ้าไม่มีการสั่งการใดๆ ภายในเวลา 10 วินาทีก็จะกลับเข้าสู่รูปแบบ Transparent ทันที คำสั่ง AT Command สามารถสรุปได้คร่าวๆดังตารางที่ L1.2 ซึ่งรูปแบบของคำสั่งที่ส่งในแบบ AT Command มีลักษณะดังภาพด้านล่างนี้

“AT” Prefix	+	ASCII Command	+	Space (Optional)	+	Parameter (Optional, HEX)	+	Carriage Return

Example: AT + DL + 1 + F + <CR> (ATDL 1F<CR>)

ตารางที่ L1.2 สรุปคำสั่ง AT Command

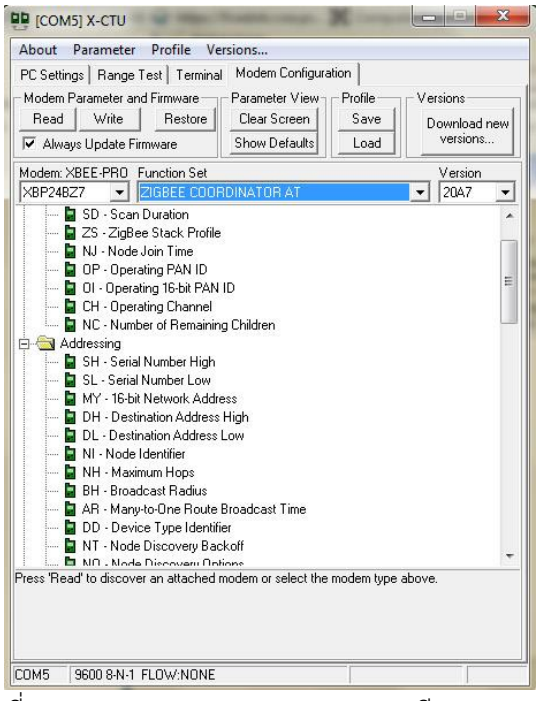
คำสั่ง	**ความหมาย**
+++	เข้าสู่โหมด AT Command
AT	Attention: XBee ควรจะต้องตอบด้วยข้อความว่า ‘OK’
ATID	ถามค่า PAN ID
ATSH/ ATSL	แสดงค่า 64 บิต serial number ของ XBee
ATDH/ATDL	แสดงค่าหมายเลขปลายทางของ XBee
ATWR	เขียนค่าที่ได้ตั้งไว้ในรูปแบบของ AT command ลงใน Firmware
ATMY	MY ID แสดงค่าหมายเลขระบุตัวตนของ XBee ขนาด 16 บิต
ATD0 … ATD7	I/O Pin Configuration หมายเลข Pin 0 – 7 ซึ่งจะต้องกำหนดหมายเลขกำกับโดยมีความหมายดังนี้ 0: สั่ง disable I/O ในหมายเลข Pin นั้น, 1: Built in function ถ้า I/O นั้นรองรับ 2: Analog input ใช้ได้เฉพาะ D0-D3, 3: Digital input, 4: Digital output, low (0 V), 5: Digital output, high (3.3V)
ATP0 … ATP1	I/O Pin configuration หมายเลข Pin 10 และ 11 สำหรับความหมายการใช้งานเหมือนด้านบน
ATIR	ใช้ในการตั้งค่า Sampling rate ของ I/O
AT%V	ต้องการให้ XBee แสดงค่าระดับแรงดันในเวลาปัจจุบัน เพื่อตรวจสอบสถานะภาพของการใช้งานแบตเตอร์รี่
ATPR	สั่งให้มีการ pull-up resistor (ภายใน 30K โอม)
ATRE	สั่ง reset การตั้งค่าของ XBee

L1.2.3.2 การทำงานในโหมด API Command

การทำงานของ XBee ในอีกโหมดคือ API (Application Programming Interface) คือโหมดทำงานแบบ Frame-based สำหรับการรับและส่งข้อมูล โดยสามารถเพิ่มศักยภาพในการทำงานมากขึ้นในเรื่องของการเปลี่ยนแปลงค่าพารามิเตอร์ของ XBee โดยที่ไม่ต้องเข้าสู่โหมด Command สามารถดูค่า RSSI (Receive Signal Strength) โดยรายละเอียดของการใช้งานในโหมด API ของ XBee จะแสดงในบทที่ 7

L1.3 ทดสอบการทำงานของ XBee เบื้องต้น

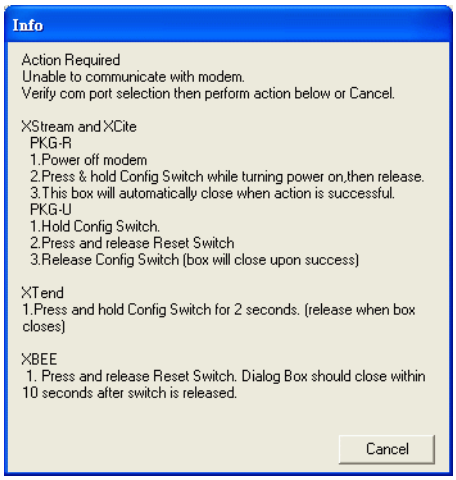
ในส่วนนี้เป็นการทดสอบการทำงานของ XBee เบื้องต้นโดยเลือกใช้ XBee รุ่น XBee Pro S2BZ7 ซึ่งจะทดสอบการเชื่อมต่อแบบ Point-to-Point ระหว่าง XBee 2 ตัว กำหนดให้ XBee 1 ตัวเป็น Coordinator และอีกตัวเป็น Router สามารถเริ่มต้นกระบวนโดยการเชื่อมต่อ XBee เข้ากับ Dongle แล้วต่อไปยังคอมพิวเตอร์ผ่านทางสาย USB

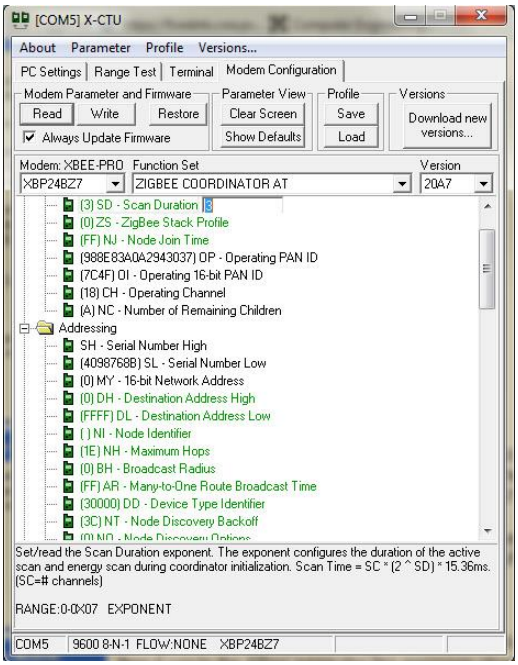


ภาพประกอบที่ L1.10 Modem Configuration – เตรียม update firmware

จากนั้นไปยังหน้าต่าง Modem Configuration เลือกรุ่น เป็น **XBP24BZ7** จากนั้นตั้งค่า Function Set เป็น **“Zigbee coordinator AT”** พบว่าเป็น version 20A7 และเลือก Click ที่ **“Always Update**

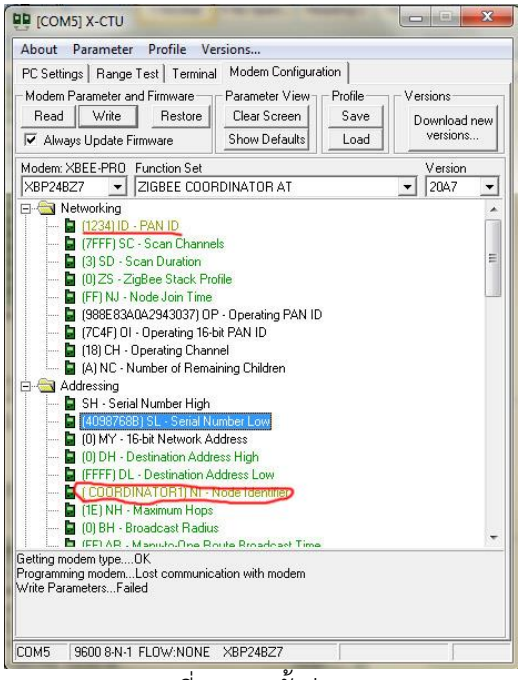
Firmware” ดังภาพประกอบที่ L1.10 หลังจากตั้งค่าเรียบร้อยแล้วสามารถทำการเริ่มกระบวนการเขียน Firmware ใหม่โดยเลือก **“Write”** ซึ่งจะพบมีหน้าต่าง Pop-up ขึ้นมาดังภาพประกอบที่ L1.11a ก็ให้กดปุ่ม Reset บนบอร์ด Dongle หน้าต่าง Dialog ก็จะหายไปแล้วเริ่มต้นการเขียน Firmware เมื่อทำการเขียนเรียบร้อยแล้วจะพบรายละเอียดของ Firmware ตัวใหม่แสดงดังภาพประกอบที่ L1.11b





a) dialog ที่ pop-up b) หน้า modem configuration หลังเขียนเสร็จ

ภาพประกอบที่ L1.11 ขั้นตอนการเขียน firmware

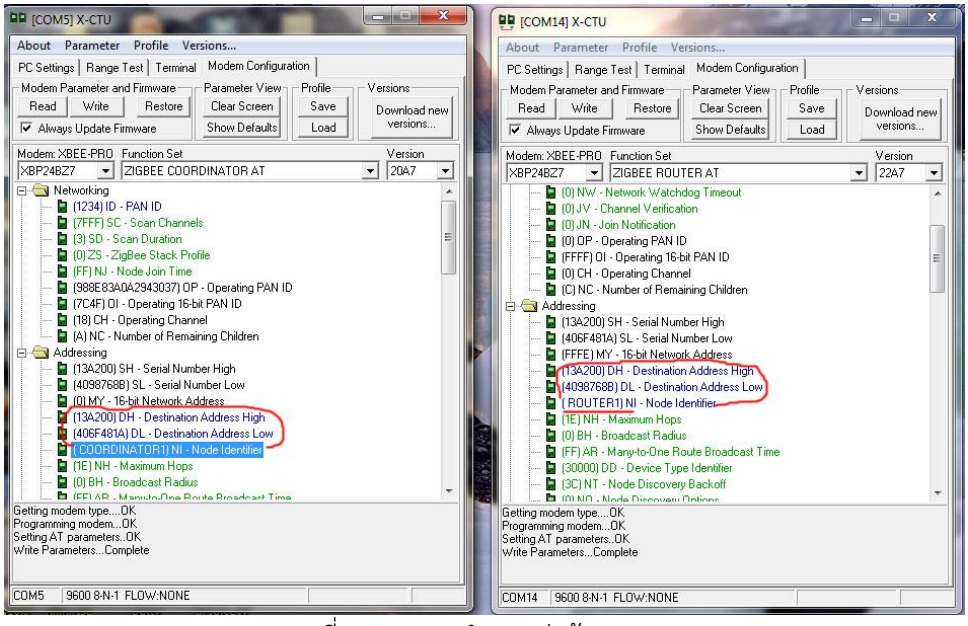


ภาพประกอบที่ L1.12 ตั้งค่า Coordinator

หลังจากที่ได้ firmware ที่ update สำหรับ XBee แล้วจะต้องทำการกำหนดค่า parameter เริ่มต้นจาก

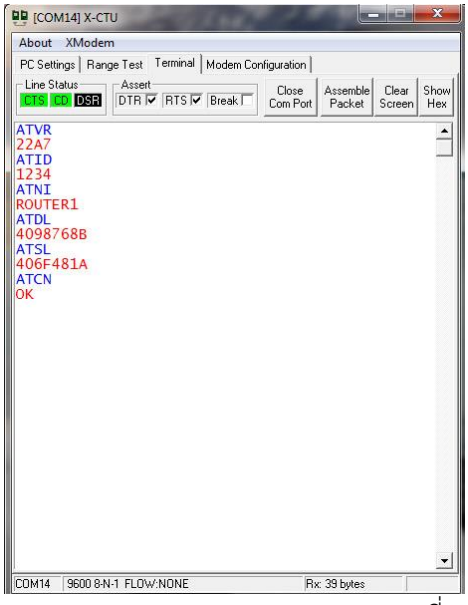
- **PAN ID** : ตั้งค่าหมายเลขของวง XBee ที่ใช้งาน ในที่นี้ตั้งเป็น ‘1234’
- **Node ID (NI)** : ตั้งเป็นชื่อได้ ในที่นี้คือ ‘COORDINATOR1’

เมื่อตั้งค่าเหล่านี้แล้วทำการ ‘write’ ใหม่อีกรอบ จากนั้นจะพบว่าค่าต่างๆจะเป็นตามปรากฏในภาพประกอบที่ L1.12 สำหรับโนดที่จะทำหน้าที่เป็น Router ก็ทำเช่นเดียวกับขั้นตอนที่ผ่านมา แต่ให้เลือก Function set เป็น **“ZIGBEE ROUTER AT”** เมื่อเสร็จสิ้นกระบวนการเรียบร้อยแล้ว จะพบค่าของทั้งโนด Coordinator และ Router ดังแสดงในภาพประกอบที่ L1.13 ซึ่งหน้าต่างทางด้านขวามือคือ Router และซ้ายมือคือ Coordinator จากนั้นให้ทำการกำหนด Destination Address ทั้ง Byte High และ Low ของโนด Router เป็นหมายเลข Serial number high และ Low ของโนด Coordinator สำหรับโนด coordinator ก็กำหนด Destination Address เป็นของ Router



ภาพประกอบที่ L1.13 การกำหนดค่าต้นทางและปลายทาง

เมื่อตั้งค่าโนดทั้งสองเสร็จเรียบร้อยแล้วเราจะลองทดสอบพูดคุยกับโนด router ด้วยคำสั่งแบบ AT เริ่มต้นด้วยคำสั่ง “+++” หมายถึงการบอก XBee ให้เข้าสู่โหมด Command ซึ่ง XBee จะตอบสนองด้วย “OK” จากนั้นลองทดสอบคำสั่งดังแสดงในภาพประกอบที่ L1.14



ATVR : Request firmware version ซึ่ง 22A7 คือ firmware สำหรับ XBP24BZ7 – Zigbee Router AT

ATID: Request for PAN ID

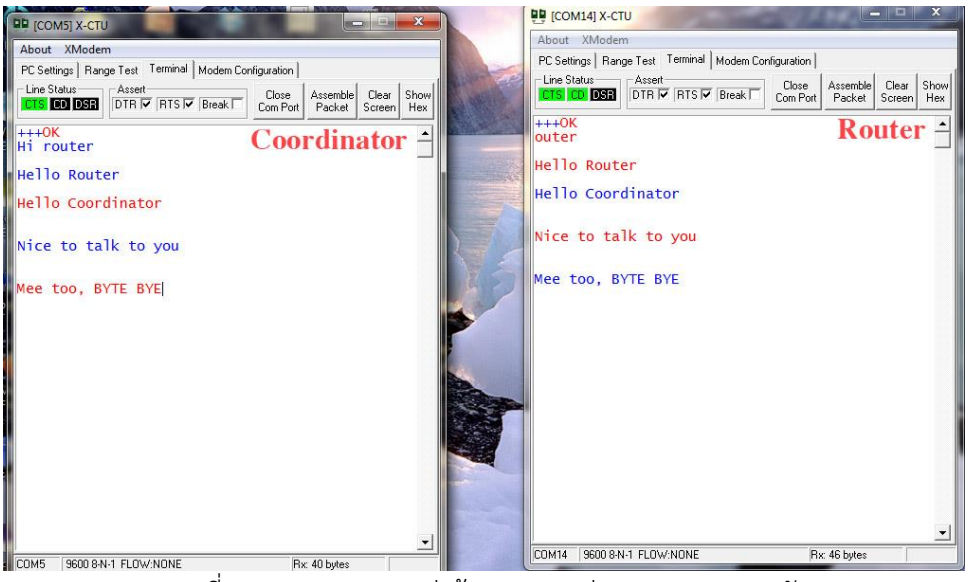
ATNI: Request for NODE ID

ATDL: Request for Destination low address

ATSL: Request for Source low address

ATCN: Force XBee to exit command mode

ภาพประกอบที่ L1.14 การทดสอบโนด Router



ภาพประกอบที่ L1.15 การทดสอบส่งข้อความระหว่าง Coordinator กับ Router

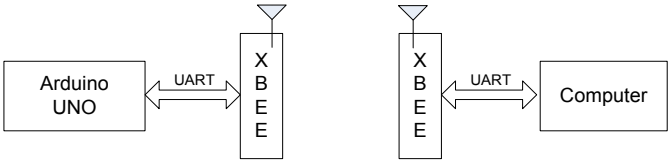
จากนั้นทำการทดสอบการส่งข้อความระหว่างโนด Coordinator และ Router ในที่นี้ทางด้านซ้ายคือโนด Coordinator ที่ต่อเข้ากับพอร์ต COM5 และโนด Router ต่ออยู่ที่พอร์ต COM14 เมื่อมีการพิมพ์ข้อความลงบนหน้าต่าง Terminal ของโนด Coordinator ข้อความจะถูกส่งไปยังโนด Router แบบไร้สาย จากนั้นข้อความจะถูกส่งผ่านพอร์ต COM14 เข้าสู่คอมพิวเตอร์แล้วถูกนำไปแสดงผลบน Terminal ของโนด Router

Project 1

ให้นักศึกษาจับคู่ให้คนใดคนหนึ่งเป็น Router อีกคนเป็น Coordinator ทำซ้ำจากเนื้อหาด้านบน เพื่อให้สามารถทำการส่งข้อความระหว่างกันแบบไร้สายได้

Project 2

ทดสอบการส่งข้อมูลจากบอร์ดไมโครคอนโทรลเลอร์ไปยังคอมพิวเตอร์แบบไร้สายผ่านโมดูล XBee ดังแสดงในภาพประกอบที่ L1.16



ภาพประกอบที่ L1.16 รูปแบบการเชื่อมต่อระหว่างไมโครคอนโทรลเลอร์กับคอมพิวเตอร์ผ่าน XBee

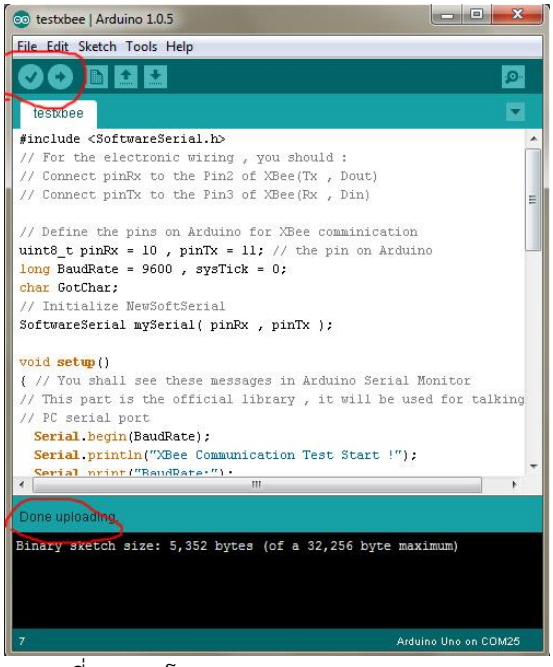
ในการทำ Project นี้จะต้องติดตั้งซอฟต์แวร์สำหรับพัฒนาบอร์ด Arduino UNO โดยสามารถดาวน์โหลดได้จากเว็ปไซต์ http://arduino.cc/en ทำการสร้าง code ตามภาพประกอบที่ L1.17 โดยโปรแกรมจะถูกเก็บไว้ใน Folder /MyDocuments/Arduino/ ตั้งชื่อตามที่ต้องการ ในการทดลองนี้เชื่อมต่อ XBee โนดที่เป็น router เข้ากับ Arduino และ XBee โนด coordinator เข้ากับคอมพิวเตอร์ สำหรับ XBee โนด router ให้ทำการต่อ 3.3V, GND, Rx, Tx ของ XBee เข้ากับขาของ Arduino โดยที่ Rx (XBee) จะต่อเข้ากับ Arduino ขา Tx (ขา 11) ในขณะที่ Tx (XBee) ต่อเข้ากับ Arduino ที่ขา Rx (ขา 10) จากนั้นจะต้องกำหนดการเชื่อมต่อ UART ที่ Baud rate 9600 โดยเราสามารถตั้งค่าโนด XBee ได้เหมือนกับตัวอย่างที่แล้ว

```
#include <SoftwareSerial.h>
uint8_t pinRx = 10 , pinTx = 11; // the pin on Arduino
long BaudRate = 9600;
char GotChar;
// Initialize SoftwareSerial
SoftwareSerial mySerial( pinRx , pinTx );

void setup()
{
  Serial.begin(BaudRate);
  Serial.println("XBee Communication Test Start !");
  Serial.print("BaudRate:");
  Serial.println(BaudRate);
  Serial.print("NewSoftSerial Rx Pin#");
  Serial.println(pinRx,DEC);
  Serial.print("NewSoftSerial Tx Pin#");
  Serial.println(pinTx,DEC);

  mySerial.begin(BaudRate);
```

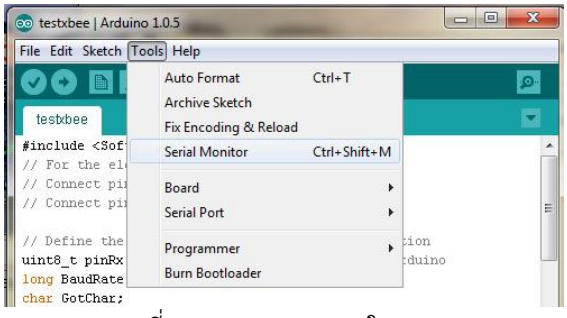
```
  mySerial.println("Powered by SoftwareSerial !");
}
void loop()
{
  if ( Serial.available() ) {
      GotChar = Serial.read();
      mySerial.print(GotChar);
  }
  if ( mySerial.available() ) {
      GotChar = mySerial.read();
      Serial.print(GotChar);
  }
}
```

ภาพประกอบที่ L1.17 โปรแกรม Arduino ส่งข้อมูลผ่าน UART - XBee

เมื่อทำการบันทึกไฟล์เรียบร้อยแล้วก็สามารถเริ่มทำการ Compile และ Upload ผลลัพธ์ใส่ลงบนบอร์ด Arduino โดยเลือกปุ่ม ที่อยู่ใต้เมนู ปุ่มแรก (Verify) และปุ่มที่สองคือ upload ดังภาพประกอบที่ L1.18 แต่ก่อนที่จะเริ่มการ verify ให้ไปที่เมนู Tools -> board -> Arduino Uno เพื่อตั้งค่าของบอร์ดที่จะใช้งาน แล้วเลือกพอร์ตที่เชื่อมต่อกับบอร์ด Arduino ให้ถูกต้อง Tools -> Serial Port จากนั้นเตรียมเปิด Terminal เพื่อรอดูผลการทำงานโดยการเลือก Tools-> Serial Monitor ตามภาพประกอบที่ L1.19

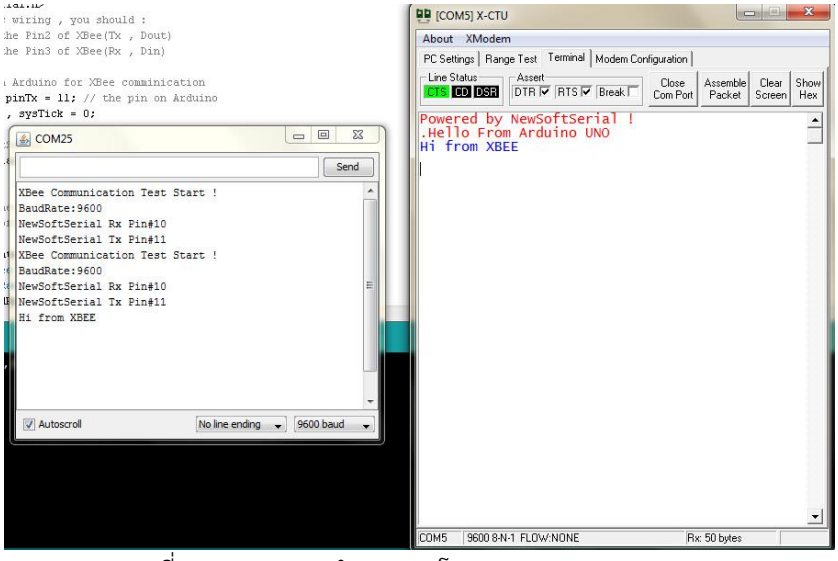


ภาพประกอบที่ L1.18 โปรแกรมทดสอบ XBee บน Arduino Dev.



ภาพประกอบที่ L1.19 เมนู Tools ใน Arduino Dev.

หน้าต่าง Terminal ของทั้ง Arduino Dev. แสดงอยู่ทางซ้ายมือในภาพประกอบ L1.20 เป็นผลของการทำงานของ Arduino + XBee ในขณะที่หน้าต่าง Terminal ของ X-CTU แสดงอยู่ทางขวามือในภาพประกอบ L1.20 พบว่าทั้งสองฝั่งสามารถเชื่อมต่อแลกเปลี่ยนข้อมูลได้



ภาพประกอบที่ L1.20 ผลการทำงานของโปรแกรมทดสอบ XBee + Arduino

เอกสารอ้างอิง

Robert Faludi, “Building Wireless Sensor Networks,” *O’Reilly*, 2011.

URL: http://www.digi.com

URL: http://arduino.cc/en

URL: https://sites.google.com/site/xbeetutorial/example/arduino-test-program-for-xbee

Lab Sheet ภาคที่ 2

แนะนำการสร้างเครือข่ายเซนเซอร์ไร้สายด้วย XBee ภาค 2

(Introduction to Building WSNs using Xbee Part II)

จุดประสงค์การเรียนรู้

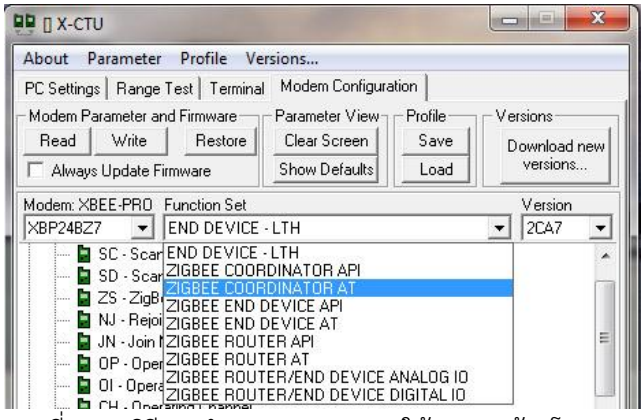
เพื่อให้สามารถใช้งาน XBee ในโหมด API และประยุกต์ใช้งานได้

รูปแบบการใช้งานของ XBee มีอยู่ 2 แบบคือ แบบ AT และ แบบ API (Application Programming Interface) ซึ่งรูปแบบการใช้งานแบบ API จะเปิดโอกาสให้โปรแกรมแต่ละฝ่ายสามารถสื่อสารกันได้โดยตรงผ่านทางมาตรฐาน API ที่เปิดไว้ให้ ในบทที่ 6 เป็นการทดสอบการใช้งาน XBee ในรูปแบบของ AT ที่ทำให้เราสามารถใส่ข้อความเพื่อสื่อสารระหว่าง XBee 2 ตัว โดยในขั้นตอนแรกจะเป็นการใส่คำสั่ง AT ในรูปแบบ Command เมื่อเรียบร้อยแล้วก็จะเข้าสู่รูปแบบ Transparent จะพบว่าการใช้งานแบบนี้จะง่ายและสะดวกถ้าเป็นการใช้งานโดยตรงกับมนุษย์ที่สามารถกรอกคำสั่งทีละคำสั่ง แต่ถ้าต้องการให้โปรแกรมสามารถสั่งงานกันได้เอง มีความรวดเร็วสำหรับการส่งข้อมูลไปยังอุปกรณ์โนดหลายๆตัว ทำให้เราทราบต้นทาง ปลายทางของข้อความเพราะมีการแสดง Source และ Destination Address อยู่ในข้อความ หรือถ้าต้องการใช้ความสามารถ Over-the-Air Firmware update ได้ ดังนั้นรูปแบบการใช้งานแบบ API จะเหมาะสมกว่า โดยในบทที่นี้จะได้อธิบายรูปแบบการใช้งานแบบ API พร้อมทั้งให้ตัวอย่างของการใช้งานแบบ API อย่างง่าย โดยที่ในบทนี้จะใช้ไมโครคอนโทรลเลอร์ Arduino ดังนั้นโปรแกรมที่ใช้ยกตัวอย่างในการอธิบายจึงเป็นโปรแกรมสำหรับไมโครคอนโทรลเลอร์ Arduino

L2.1 โพรโทคอล API ใน XBee

เราจะสามารถกำหนดให้ XBee ทำงานในรูปแบบของ API โดยการกำหนด “Function Set” ในโปรแกรมของ X-CTU ซึ่งจะเป็นการเตรียมโหลด Firmware สำหรับการทำงานในรูปแบบนี้ลงใน XBee ก่อนที่จะ

เริ่มใช้งานซึ่งสามารถศึกษาได้จากภาพประกอบที่ L2.1 ซึ่งเราสามารถเลือกใช้งานได้ 3 แบบคือ XBee ที่ต้องการเป็น Coordinator API, End Device API และ Router API



ภาพประกอบที่ L2.1 วิธีการกำหนด Firmware ให้ XBee ด้วยโปรแกรม X-CTU

เมื่อกำหนดและติดตั้ง Firmware ให้กับ XBee เพื่อให้ทำงานในแบบ API ได้แล้ว ก็จะต้องเริ่มศึกษาทำความเข้าใจถึงโครงสร้างข้อความแบบ API ก่อนดังแสดงในภาพประกอบที่ L2.2 เริ่มต้นข้อความด้วย Start delimiter (0x7E) เราเรียกง่ายๆว่า Start byte ถัดมาเป็น Byte แสดงขนาดของข้อความโดยใช้ 2 bytes ในการแสดงขนาดข้อความ ต่อมาก็เป็น Frame Data ที่เป็นโครงสร้างพิเศษเฉพาะสำหรับ API โดยจะมีรายละเอียดในส่วนต่อไป สุดท้ายจะเป็นข้อมูลขนาด 1 byte เป็นค่า Checksum

Start delimiter	Length	Frame Data	Checksum
Byte 1	Byte 2-3	Byte 4 ... Byte n	Byte n+1
0x7E	MSB, LSB	API- Specific Structure	One byte

ภาพประกอบที่ L2.2 โครงสร้างข้อความแบบ API

วิธีการคำนวณค่า Checksum ทำได้โดยการนำเฉพาะข้อมูลในส่วนของ Frame Data มาทำการบวกทีละ Byte แล้วนำเฉพาะ Byte สุดท้ายของผลลัพธ์ไปลบออกจากค่า 0xFF ก็จะได้ค่า Checksum ใน Byte สุดท้าย

โดยเมื่อทำการนำค่า Checksum กลับไปบวกแต่ละ Byte ในส่วนของ Frame Data ก็จะได้ผลลัพธ์เป็น 0xFF คือเป็นการทวนสอบคำตอบอีกครั้ง

L2.2 โครงสร้างของ API ใน Frame Data

รูปแบบโครงสร้างของ API ที่จะถูกบรรจุในส่วนของ Frame data นั้นจะมีความหมายเฉพาะสำหรับใช้ในการรับส่งข้อมูล โดยข้อมูลใน byte แรก เรียกว่า Frame type จะประกอบด้วยความหมายดังตารางที่ L2.1

ตารางที่ L2.1 แสดงความหมายของ Frame type ในส่วนของ API-Specific Structure

Frame type	ความหมาย
0x08	เป็นการบ่งบอกว่า frame นี้เป็น AT Command (immediate)
0x09	เป็นการบ่งบอกว่า frame นี้เป็น AT Command (queued)
0x17	ร้องขอใช้งานแบบ Remote command
0x88	AT command response
0x8A	Modem status
0x10	Tx request
0x8B	Tx response
0x90	Rx received
0x92	Rx I/O data received
0x95	Node identification indicator
0x97	Remote command response

AT Command

เมื่อเราทราบชนิดของ Frame ที่มีให้ใช้งานแล้วก็ต้องตรวจสอบโครงสร้างองค์ประกอบในแต่ละ byte ของ Frame ในแต่ละชนิดด้วย ดังตัวอย่างแรกคือในการใช้งานชนิดของ API Frame แบบ AT Command ผ่านข้อความแบบ API เพื่อเพื่อจะทำการตั้งค่าการใช้งานคลื่นวิทยุหรือตัวแปรต่างๆที่จำเป็นบน XBee ซึ่งจะได้ผลเหมือนกับการส่งคำสั่ง AT Command ในโหมดของ transparent/command ในบทที่ผ่านมา โดยที่รูปแบบโครงสร้างของ API Frame สำหรับการส่ง AT Command แสดงไว้ในตารางที่ L2.2

ตารางที่ L2.2 API Frame สำหรับใช้งานแบบ AT Command

Frame field		Offset	ตัวอย่าง	คำอธิบาย
Start delimiter		0	0x7E	
Length		1(MSB)		ขนาดของข้อความ
		2(LSB)		
Frame-specific data	Frame type	3	0x08	บอกว่าจะใช้งาน API แบบ ส่ง AT Command
	Frame ID	4		XBee จะต้องรับข้อมูลดังนั้นจะต้องมีหมายเลขของ Frame เพื่อใช้เป็นข้อมูลตอบกลับมาบอกว่าได้รับ Frame ของหมายเลขใด
	AT command	5	0x4E	สำหรับใส่ ASCII ตัวอักษรที่เป็นคำสั่งของ AT Command แต่เราจะต้องเปลี่ยน ASCII ตัวอักษรให้เป็นเลขฐาน 16 เช่น N = 0x4E เป็นต้น
		6	0x4E	
	Parameter value			(optional) สำหรับใส่ค่าที่จะต้องกำหนดต่อท้าย AT Command
Checksum		7		0xFF – 8-bit ของการ sum byte ทั้งหมดในข้อความจาก byte ที่ 3 เป็นต้นมา

AT Response

API Frame ในแบบนี้จะพบเมื่อได้รับการตอบกลับ (Acknowledge) จาก XBee ที่สามารถรับ AT Command โดยอาจจะมีค่าที่เราต้องการกลับมา ดังนั้นรูปแบบของ API Frame นี้จึงจะถูกสร้างโดยอัตโนมัติจาก XBee เพื่อสำหรับอ่านเพื่อเข้าใจการตอบกลับหรือเขียนโปรแกรมให้ไมโครคอนโทรลเลอร์ทำการรับค่าที่อ่านมาได้ API Frame ชนิดนี้จึงจะไม่ถูกสร้างหรือเขียนขึ้นมาเองมีรายละเอียดดังตารางที่ L2.3

ตารางที่ L2.3 API Frame สำหรับใช้งานแบบ AT Response

Frame field		Offset	ตัวอย่าง	คำอธิบาย
Start delimiter		0	0x7E	
Length		1(MSB)		ขนาดของข้อความ
		2(LSB)		
Frame-specific	Frame	3	0x88	บอกว่าจะใช้งาน API แบบ ส่ง AT Response

data	type			
	Frame ID	4		XBee จะต้องรับข้อมูลดังนั้นจะต้องมีหมายเลขของ Frame เพื่อใช้เป็นข้อมูลตอบกลับมาบอกว่าได้รับ Frame ของหมายเลขใด
	AT command	5	0x4E	สำหรับใส่ ASCII ตัวอักษรที่เป็นคำสั่งของ AT Command
		6	0x4E	
	Command status	7		0 = OK, 1 = ERROR, 2= Invalid command, 3=invalid parameter, 4=Tx failure
	Command data			รีจิสเตอร์สำหรับเก็บข้อมูลใน format binary ถ้าเป็น '1' (set) หมายถึงไม่มีการ return ค่ากลับมา
Checksum		8		0xFF – 8-bit ของการ sum byte ทั้งหมดในข้อความจาก byte ที่ 3 เป็นต้นมา

Zigbee Transmit Request

ในส่วนนี้เป็นการส่งข้อมูลกันจริงๆคือการใช้ Frame specific แบบ Zigbee transmit request ซึ่งจะมีส่วนของข้อมูลหรือ Payload อยู่ในข้อมูลชุดนี้ มีรายละเอียดดังตารางที่ L2.4

ตารางที่ L2.4 API Frame สำหรับใช้งานแบบ Zigbee Transmit Request

Frame field		**Offset**	**ตัวอย่าง**	**คำอธิบาย**
Start delimiter		0	0x7E	
Length		1(MSB)		ขนาดของข้อความ
		2(LSB)		
Frame-specific data	Frame type	3	0x10	บอกว่าจะใช้งาน API แบบ Zigbee transmit request
	Frame ID	4		XBee จะต้องรับข้อมูลดังนั้นจะต้องมีหมายเลขของ Frame เพื่อใช้เป็นข้อมูลตอบกลับมาบอกว่าได้รับ Frame ของหมายเลขใด
	64-bit destination Address	5-12		0x0000000000000000 – Reserved 64-bit address for the coordinator. 0x000000000000FFFF – Broadcast address

	16-bit destination network address	13-14		ตั้งค่าเป็น 0xFFFE ถ้าไม่รู้ค่า address หรือหมายถึงจะส่งแบบ broadcast
	Broadcast radius	15		ตั้งค่าจำนวน hop สูงสุด ถ้าเป็น '0' หมายถึงให้มี hop ได้สูงสุด
	Option	16		กำหนดรูปแบบการส่งข้อมูล 0 หมายถึงไม่ใช้ 0x01 – Disable ACK, 0x20 – Enable APS encryption (if EE=1), 0x40 – ใช้การยืดเวลา timeout สำหรับส่งข้อมูล
	RF Data	17-24		ข้อมูลที่จะส่งไปยังโนดปลายทาง ซึ่งอาจจะสามารถขยายให้สูงสุดได้ถึง 84 byte
Checksum		25		0xFF – 8-bit ของการ sum byte ทั้งหมดในข้อความจาก byte ที่ 3 เป็นต้นมา

Zigbee Transmit Status

เป็นการใช้เมื่อต้องการข้อมูลสถานะเต็มรูปแบบของโนด และข้อมูลอื่นๆที่จำเป็น แต่ไม่ใช่ข้อมูลหรือ data ดังรายละเอียดในตารางที่ L2.5

ตารางที่ L2.5 API Frame สำหรับใช้งานแบบ Zigbee Transmit Status

Frame field		**Offset**	**ตัวอย่าง**	**คำอธิบาย**
Start delimiter		0	0x7E	
Length		1(MSB)		ขนาดของข้อความ
		2(LSB)		
Frame-specific data	Frame type	3	0x8B	บอกว่าจะใช้งาน API แบบ Zigbee transmit Status
	Frame ID	4		XBee จะต้องรับข้อมูลดังนั้นจะต้องมีหมายเลขของ Frame เพื่อใช้เป็นข้อมูลตอบกลับมาบอกว่าได้รับ Frame ของหมายเลขใด
	16-bit destination	5-6		ถ้าส่งได้สำเร็จถูกต้อง ค่าของ 16-bit network address นี้จะเป็นค่าของโนดที่เราส่งไป

	network address			
	Transmit retry counter	7		มีการ retry ทั้งหมดกี่ครั้ง
	Delivery status	8		0x00 = success 0x01 = MAC ACK failure 0x02 = CCA failure 0x15 = invalid destination endpoint 0x21 = network ACK failure 0x22 = not joined to network 0x23 = self-addressed 0x24 = Address not found 0x25 = route not found 0x26 = broadcast source fail to hear a neighbor relay the message 0x2B = invalid blinding table index 0x2C = resource error, lack of free buffers, timers, etc. 0x2D = attempted broadcast with APS transmission 0x2E = attempted unicast with APS transmission 0x32 = resource error, lack of free buffer, timers, etc. 0x74 = payload too large 0x75 = indirect message unrequested
	Discovery status	9		0x00 = no discovery overhead 0x01 = address discovery 0x02 = route discovery 0x03 = address and route 0x40 = extended timeout discovery
Checksum		10		0xFF – 8-bit ของการ sum byte ทั้งหมดในข้อความจาก byte ที่ 3 เป็นต้นมา

Zigbee received packet

ข้อความชนิดนี้ถูกใช้เพื่อส่งข้อมูลตอบกลับหลังจากที่โนดได้รับข้อความชนิด transmit ไปแล้วซึ่งจะมีการระบุ ID ของโนดที่ส่งกลับมาด้วยทำให้ทราบว่าตอบกลับมาจากโนดใด มีรายละเอียดดังตารางที่ L2.6

ตารางที่ L2.6 API Frame สำหรับใช้งานแบบ Zigbee received packet

Frame field		Offset	ตัวอย่าง	คำอธิบาย
Start delimiter		0	0x7E	
Length		1(MSB)		ขนาดของข้อความ
		2(LSB)		
Frame-specific data	Frame type	3	0x90	บอกว่าจะใช้งาน API แบบ Zigbee Rx packet
	64-bit source address	4-11		64-bit address ของผู้ส่ง ถ้ากำหนดเป็น 0xFFFFFFFF หมายถึงไม่ทราบ address
	16-bit destination network address	12-13		16-bit address ของผู้ส่ง
	Receive option	14		0x01 – packet acknowledge 0x02 – packet was broadcasted 0x20 – packet encryption with APS 0x40 – packet was sent from an end device.
	Received data	15-20		Received RF Data
Checksum		21		0xFF – 8-bit ของการ sum byte ทั้งหมดในข้อความจาก byte ที่ 3 เป็นต้นมา

Zigbee I/O Data Sample Rx Indicator

ข้อความในรูปแบบของ I/O Data sample Rx indicator นั่นเป็นรูปแบบที่ขยายต่อจากข้อมูลในรูปแบบ ซึ่งโดยปกติแล้วการทำงานแบบ AT จะไม่สามารถเข้าถึง I/O ได้เลย การใช้รูปแบบข้อความแบบ API ทำให้สามารถได้ข้อมูลตรงจาก I/O ของ XBee โดยมีรายละเอียดของข้อความดังตารางที่ L2.7

ตารางที่ L2.7 API Frame สำหรับใช้งานแบบ Zigbee I/O Data Sample Rx indicator

Frame field		**Offset**	**ตัวอย่าง**	**คำอธิบาย**
Start delimiter		0	0x7E	
Length		1(MSB)		ขนาดของข้อความ
		2(LSB)		
Frame-specific data	Frame type	3	0x92	บอกว่าจะใช้งาน API แบบ Zigbee I/O Data Sample Rx indicator
	64-bit source address	4-11		64-bit address ของผู้ส่ง ถ้ากำหนดเป็น 0xFFFFFFFF หมายถึงไม่ทราบ address
	16-bit destination network address	12-13		16-bit address ของผู้ส่ง
	Receive option	14		0x01 – packet acknowledge 0x02 – packet was broadcasted
	Number of sample	15	0x01	Always set to 1
	Digital channel mask	16-17		bit mask to indicate which digital I/O lines on the remote have sampling enabled. First byte = “n/a n/a n/a D12 D11 D10 n/a n/a” และ second byte = “D7 … D0”
	Analog channel mask	18		bit mask to indicate which digital I/O lines on the remote have sampling enabled.
	Digital sample	19-20		สำหรับกำหนด sample ของทุก digital I/O
	Analog sample	21-22		สำหรับกำหนด sample ของทุก Analog I/O
Checksum		23		0xFF – 8-bit ของการ sum byte ทั้งหมดในข้อความจาก byte ที่ 3 เป็นต้นมา

Remote AT Command Request

ข้อความแบบ API ที่ใช้สำหรับการส่งคำสั่งแบบ AT ผ่านบนข้อความแบบ API ได้ เพื่อร้องขอให้ทำงานอย่างใดอย่างหนึ่งโดยมีรายละเอียดของข้อความดังตารางที่ L2.8

ตารางที่ L2.8 API Frame สำหรับใช้งานแบบ Zigbee Remote AT Command Request

Frame field		Offset	ตัวอย่าง	คำอธิบาย
Start delimiter		0	0x7E	
Length		1(MSB)		ขนาดของข้อความ
		2(LSB)		
Frame-specific data	Frame type	3	0x17	บอกว่าจะใช้งาน API แบบ Remote AT Command request
	Frame ID	4		
	64-bit source address	5-12		64-bit address ของผู้ส่ง ถ้ากำหนดเป็น 0xFFFFFFFF หมายถึงไม่ทราบ address
	16-bit destination network address	13-14		16-bit address ของผู้ส่ง
	Remote command option	15		0x01 – Disable ACK 0x02 – apply changes on remote 0x40 – used extended transmission timeout for this destination 0x00 - unused
	AT Command	16-17		The name of command
	Command parameter	18		
Checksum		19		0xFF – 8-bit ของการ sum byte ทั้งหมดในข้อความจาก byte ที่ 3 เป็นต้นมา

Remote AT Command Response

ข้อความแบบ API ที่ใช้สำหรับการส่งคำสั่งแบบ AT ผ่านบนข้อความแบบ API ได้ เพื่อตอบรับการทำงานที่เกิดขึ้นจาก AT Command Request โดยมีรายละเอียดของข้อความดังตารางที่ L2.9

ตารางที่ L2.9 API Frame สำหรับใช้งานแบบ Zigbee Remote AT Command Response

Frame field		Offset	ตัวอย่าง	คำอธิบาย
Start delimiter		0	0x7E	
Length		1(MSB)		ขนาดของข้อความ
		2(LSB)		
Frame-specific data	Frame type	3	0x97	
	Frame ID	4		ช่องนี้จะเป็นค่าเดียวกับที่ส่ง request
	64-bit source address	5-12		64-bit address ของผู้ส่ง ถ้ากำหนดเป็น 0xFFFFFFFF หมายถึงไม่ทราบ address
	16-bit destination network address	13-14		16-bit address ของผู้ส่ง
	AT Command	15-16		The name of command
	Command parameter	17		0 = OK, 1 = ERROR 2 = invalid command, 3 = invalid parameter 4 = Remote command transmission failed
	Command data	18-21		ค่า register data ในรูปแบบของ Binary
Checksum		22		0xFF – 8-bit ของการ sum byte ทั้งหมดในข้อความจาก byte ที่ 3 เป็นต้นมา

เมื่อนักพัฒนาเข้าใจรูปแบบโครงสร้างของข้อความแบบ API แล้วก็จะสามารถใช้งานได้อย่างถูกต้องและมีประสิทธิภาพ ตัวอย่างเช่น ถ้าเราใช้ API แบบ I/O Data sample เราสามารถนำค่าในแต่ละ Byte ตามรูปแบบที่กำหนดไว้มาใช้งานได้ทันที โดยที่โปรแกรมไม่จำเป็นที่จะต้องเขียนสอบตรวจขนาดของข้อมูล หรือทำให้เราทราบขนาดข้อมูลว่ามีทั้งหมดกี่ Byte ในที่นี้ถ้าไม่คิด checksum) Byte ที่ 23) การได้ข้อความ API แบบ I/O Data sample จะมีขนาด 23 Byte (0 – 22) ซึ่งข้อมูลของเซนเซอร์ประเภท Analog จะถูกเก็บไว้ใน Byte ลำดับที่ 21 – 22 เริ่มต้นจาก) Byte ที่ 0) เมื่อเขียนโปรแกรมตรวจสอบให้แน่ชัดว่าข้อความที่ได้รับมานี้เป็นข้อความ API แบบ

I/O Data sample จริงๆแล้ว (if (Serial.read() == 0x7E) เราก็สามารถทำการอ่านค่าใน Byte อื่นๆทิ้งออกจาก buffer ได้ เมื่อถึงลำดับ Byte ที่ต้องการจึงทำการดึงค่าออกมาใช้งานดังตัวอย่างโปรแกรมด้านล่างนี้

```
if (Serial.read() == 0x7E) {
        //flush the data out of the buffer
        for (int i = 0; i < 19; i++) {
                byte flush = Serial.read();
        }
        int analogHigh = Serial.read();
        int analogLow = Serial.read();
}
```

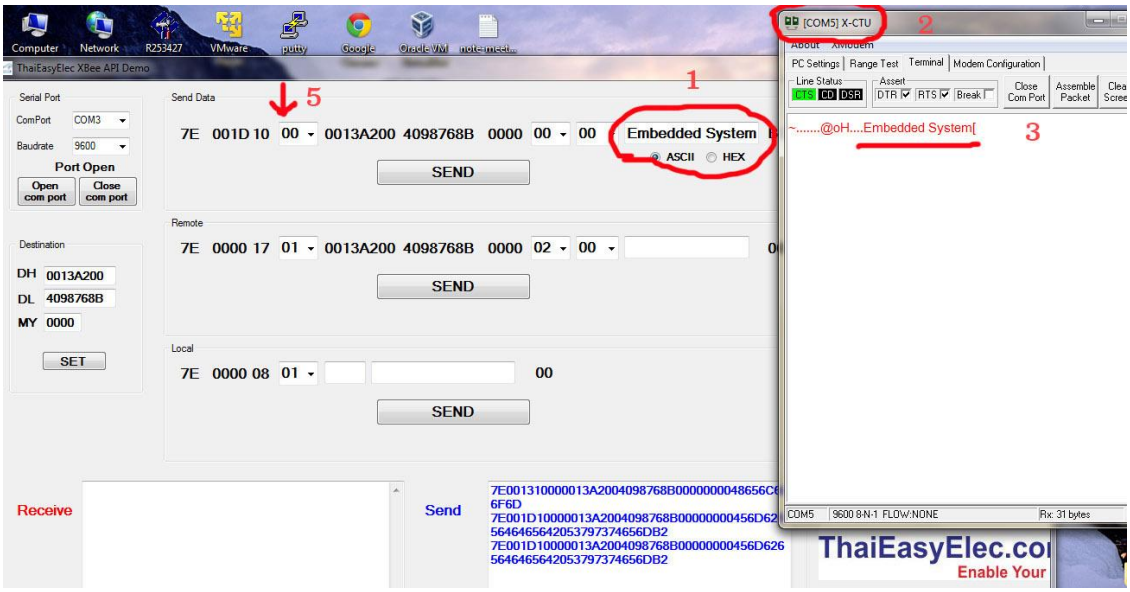
การใช้งาน Library ของ Arduino ที่รองรับการเรียกใช้ API ก็จะช่วยให้สามารถพัฒนาโปรแกรมได้อย่างมีประสิทธิภาพ ตัวอย่างของ Library สำหรับ Arduino จะสามารถศึกษารายละเอียดและหา Library มาติดตั้งได้จาก http://code.google.com/p/xbee-api/ ซึ่งพัฒนาโดย Andrew Rapp

L3.3 การใช้งาน XBee ในแบบ API ด้วย XBee API Demo ของ Thaieasyelec

ในหัวข้อนี้เป็นการแนะนำการทดลองการส่งข้อมูลในแบบ API อย่างง่าย โดยทำการทดสอบร่วมกับโปรแกรม XBee API Demo ของบริษัท Thaieasyelec ซึ่งสามารถหาดาวน์โหลดได้จาก http://www.thaieasyelec.net/archives/Manual/XBeeAPIDemo.zip เพื่อให้ทดสอบความถูกของในการทำงานของ XBee ก่อนที่จะพัฒนาขั้นต่อไป โดยหน้าที่ของโปรแกรม XBee API Demo ก็คือจะทำการติดต่อกับโนด Coordinatorเชื่อมต่ออยู่กับพอร์ต) COM6) ที่ถูกโปรแกรมด้วย X-CTU แล้วว่าให้ทำหน้าที่เป็น Coordinator API เพื่อส่งข้อมูลไปยังโนด Routerเชื่อมต่ออยู่กับพอร์ต) COM5) ที่ได้ถูกโปรแกรมด้วย X-CTU แล้วว่าเป็น Router API ดังภาพประกอบที่ L2.3 โดยจะทำการเปิดโปรแกรม X-CTU ให้เชื่อมกับโนด Router ที่เป็นพอร์ต COM5 ไว้ตรวจสอบข้อความที่ได้รับจากโนด Coordinator

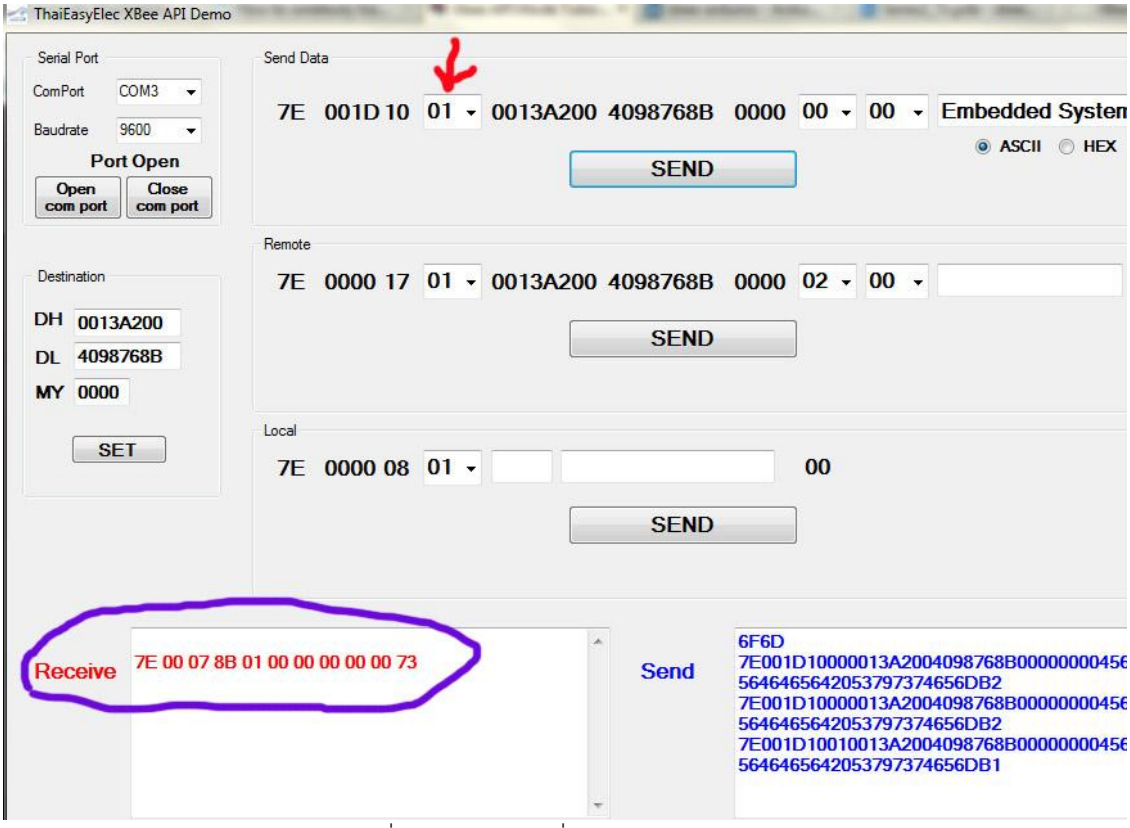
ภาพประกอบที่ L2.3 การเชื่อมต่อโนดเพื่อทดสอบการส่งข้อมูลแบบ API แบบที่ 1

บนโปรแกรม XBee API Demo เราสามารถเพิ่มข้อความที่จะใช้ส่งในช่องหมายเลข 1 ดังแสดงในภาพประกอบที่ L2.4 โดยพิมพ์ข้อความในรูปแบบของ ASCII ได้โดยตรง ในที่นี้กำหนดให้ส่งข้อความ “Embedded System” ในทางด้านของโนด Router ที่เชื่อมต่ออยู่กับพอร์ต COM5 ก็ทำการเปิดโปรแกรม X-CTU ไปยังหน้า Terminal ตรวจสอบให้เรียบร้อยว่าเป็นการเชื่อมต่อกับโนด Router จริงๆโดยสังเกตที่ชื่อของหน้าต่างโปรแกรม X-CTUหมายเลข) 2 ในภาพประกอบที่ L2.4) จากนั้นทำการกดปุ่ม “SEND” บนโปรแกรม XBee API Demo ถ้าสามารถสื่อสารได้สำเร็จ ก็จะปรากฏข้อความ “Embedded System” บนหน้าต่างของโนด Routerหมายเลข) 3 ในภาพประกอบที่ L2.4) และไม่พบข้อความตอบกลับใดๆในช่องของ Receive บนโปรแกรม XBee API Demo เพราะที่การกำหนดให้ไม่ต้องตอบกลับด้วยเลข “0” หรือ Byte ที่ 5หมายเลข) 5 ในภาพประกอบที่ L2.4) ในข้อความ API แบบ Tx Request (0x10) ในส่วนของข้อความที่ส่งออกจากโนด Coordinator ไปยังโนด Router สามารถตรวจสอบได้จากในช่อง Send บนโปรแกรม XBee API Demo



ภาพประกอบที่ L2.4 เมื่อ Router รับข้อมูลจาก Coordinator ที่ส่งผ่านทางโปรแกรม XBee API Demo

เมื่อต้องการให้มีการตอบกลับเมื่อโนดได้รับข้อความแล้วเราสามารถกำหนดได้ใน Byte ที่ 5 ให้เป็น 0x01 ก็จะได้รับข้อความตอบกลับมาแสดงในช่อง Receive ของโปรแกรม XBee API Demo ดังภาพประกอบที่ L2.5 โดยข้อความดังกล่าวคือ “7E 00 07 8B 01 00 00 00 00 00 00 73” โดยใน Byte ที่ 4 จะแสดงค่า 8B หมายถึงข้อความ API แบบ Tx Response มีค่า 16-bit address ของโนดปลายทางเป็น “00 00” ใน Byte ที่ 8 เป็นค่า 0x00 หมายถึงส่งสำเร็จ ดังนั้นในการนำไปประยุกต์ใช้งาน นักพัฒนาสามารถอ่านข้อมูลใน Byte ที่ 8 นี้ เพื่อตรวจสอบว่าส่งสำเร็จหรือไม่ แล้วใช้เป็นเงื่อนไขในการแสดงผลของการรับส่งข้อมูลผ่านการ Blink ของ LED



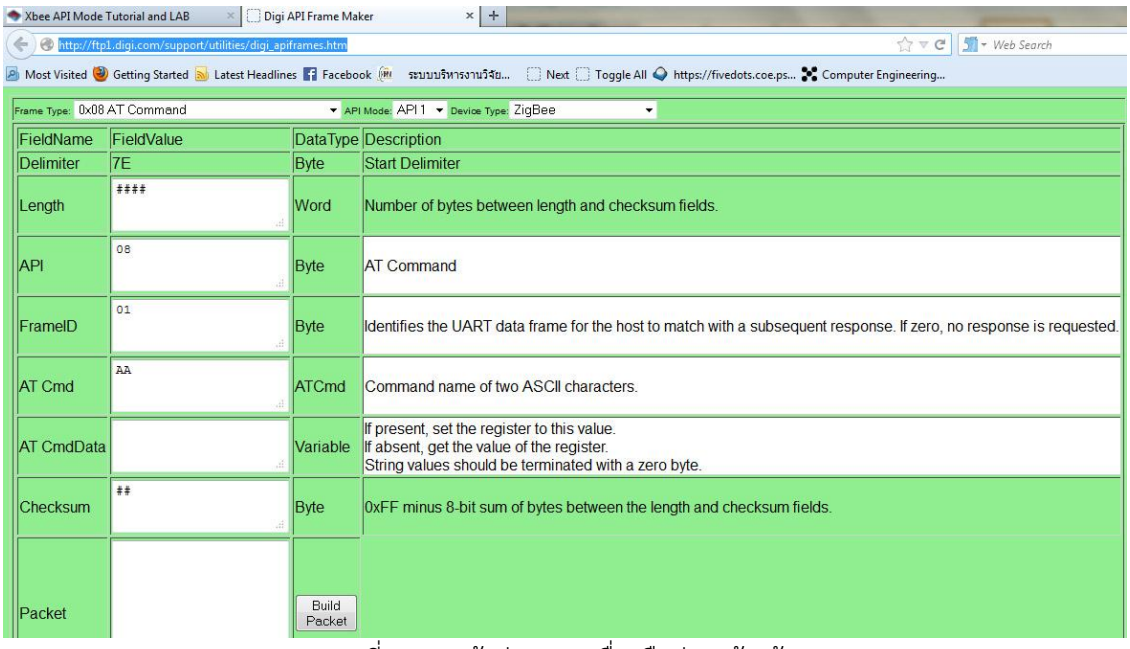
ภาพประกอบที่ L2.5 ข้อความที่ได้รับตอบกลับจากโนด Router

L2.4 การใช้งาน XBee ในแบบ API ด้วย X-CTU

ในการใช้งาน XBee แบบ API เราสามารถใช้และทดสอบการทำงานได้ด้วยโปรแกรม X-CTU ที่ใช้ในการดาวน์โหลดตั้งค่า XBee เช่นกัน โดยทดสอบการทำงานของข้อความ API แบบ Modem Status ด้วยการทำการ reset บน XBee หรือเรียกว่า Hardware reset จะพบข้อความบนหน้าต่าง Terminal ของ X-CTU เป็นข้อความ API 2 ชุดดังภาพประกอบที่ L2.6 Byte ที่ 3 คือค่า 0x8A แสดงชนิดของ API แบบ Modem status โดยค่า 0x00 ใน Byte ถัดไปคือสถานะที่บอกว่าเกิด Hardware reset ในข้อความ API ชุดที่สองคือ “7E 00 02 8A 06 6F” เป็นการแสดงสถานะว่าโนด Coordinator ได้เริ่มทำงานดูได้จากค่า 0x06 ใน Byte ที่ 4 ของข้อความชุดนี้ เราสามารถสรุปสถานะของการทำงานได้เป็น 6 ค่าได้แก่ 0x00 หมายถึง hardware reset, 0x01 หมายถึง watchdog reset, 0x02 หมายถึง joined network, 0x03 หมายถึง disassociated, 0x06 หมายถึง coordinator started และ 0x80+ หมายถึง stack error

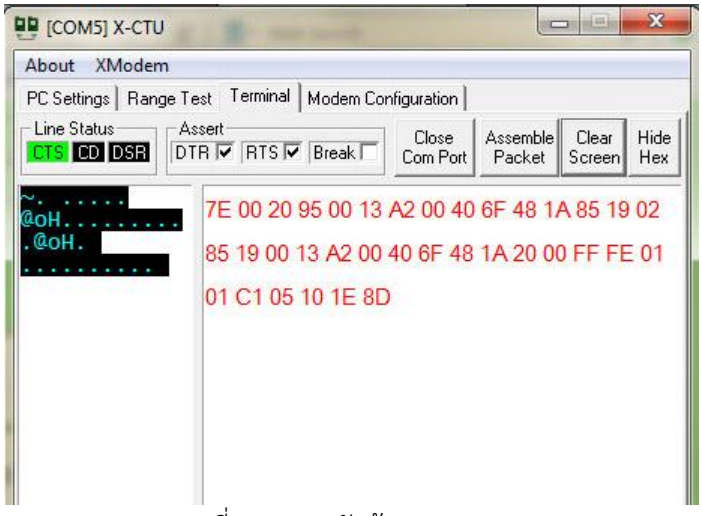
ภาพประกอบที่ L2.6 Hardware reset ของข้อความ API แบบ Modem Status

ก่อนที่จะเริ่มการรับส่งข้อความ API บน X-CTU เราจำเป็นที่จะต้องมีเครื่องมือช่วยในการสร้างข้อความ ซึ่งสามารถเข้าใช้งานได้จาก http://ftp.1digi.com/support/utilities/digi_apiframes.htm ของ บริษัท Digi ดังภาพประกอบที่ L2.7



ภาพประกอบที่ L2.7 หน้าต่างของเครื่องมือช่วยสร้างข้อความ API

การทดสอบข้อความ API ชนิด Node identification indicator เมื่อทำการกดปุ่ม D0 โปรแกรม X-CTU แสดงข้อความดังภาพประกอบที่ L2.8 Byte ที่ 3 คือ 0x95 เป็นการบ่งบอกว่าข้อความ API นี้เป็นชนิด Node identification indicator โดยที่ความหมายของข้อความจะแสดงในตารางที่ L2.10

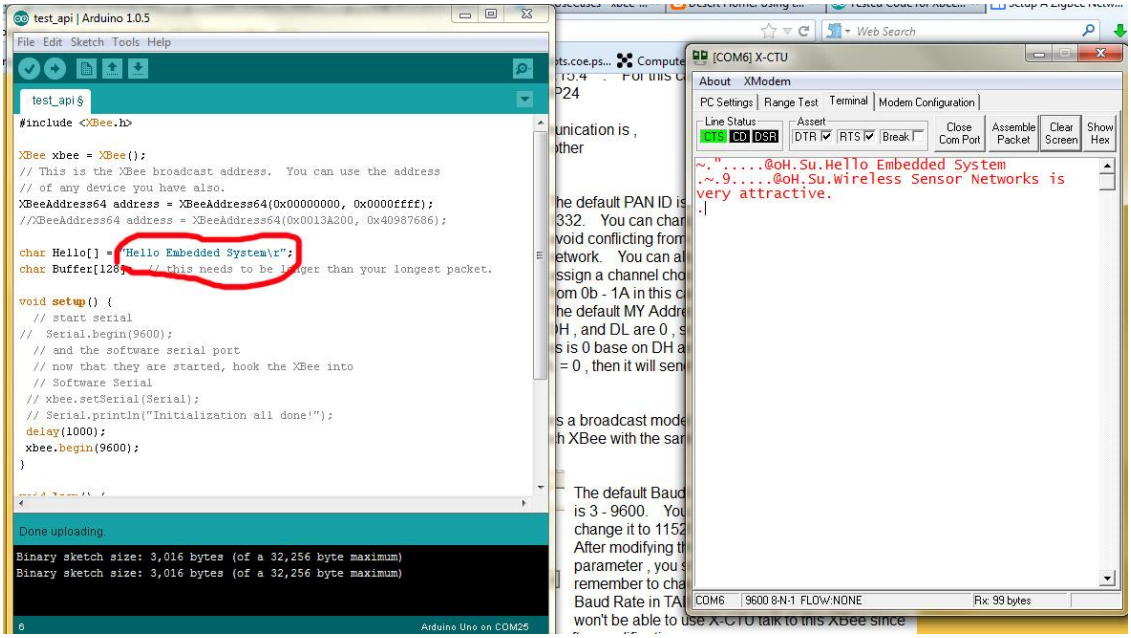


ภาพประกอบที่ L2.8 การรับข้อความ API บน X-CTU

ตารางที่ L2.10 ความหมายของข้อความ API แบบ Node identification indicator ในส่วนของ Frame Data

Byte	ขนาด	ความหมาย
1-8	64-bit sender address	ข้อมูล address ขนาด 8 bytes ของโนดต้นทาง
9-10	16-bit sender address	ข้อมูล address ขนาด 2 bytes ของโนดต้นทาง
11	Receive options	0x01 หมายถึง packet acknowledged 0x02 หมายถึง packet broadcast 0x20 หมายถึง Packet encrypted with APS encryption 0x40 - Packet was sent from an end device (if known)
12-13	16-bit remote address	ข้อมูล address ขนาด 2 bytes ของโนดปลายทาง
14-21	64-bit remote address	ข้อมูล address ขนาด 8 bytes ของโนดปลายทาง
22-23	16-bit	NI String เป็นชื่อของโนด
24-25	Remote parent	เป็นค่า 0xFFFE
26	Device type	0 – coordinator, 1 – Router, 2 – end device
27	Source event	1 - Frame sent by node identification push button event (see D0 command). 2 - Frame sent after joining event occurred (see JN command). 3 - Frame sent after power cycle event occurred (see JN command).
28-29	DigiProfileID	
30-31	DigiManuID	
32	Checksum	

การทดลองสุดท้ายคือการทดสอบการส่งข้อความในที่นี้คือ (Hello Embedded System) จากบอร์ด Arduino ไปยังคอมพิวเตอร์ด้วย Xbee แบบ API มีผลการทำงานดังภาพประกอบที่ L2.9 ภาพทางขวาคือโปรแกรม X-CTU ที่เชื่อมต่อผ่านพอร์ตอนุกรมและต่อพ่วงกับ Xbee เพื่อรอรับข้อความที่จะส่งมายังคอมพิวเตอร์ ในขณะที่ภาพทางซ้ายเป็นโปรแกรมที่เขียนให้กับบอร์ด Arduino อย่างง่ายเพื่อให้ส่งข้อความออก



ภาพประกอบที่ L2.9 ตัวอย่างการส่งข้อความจากบอร์ด Arduino ผ่าน Xbee ไปยังคอมพิวเตอร์

เอกสารอ้างอิง

Robert Faludi, “Building Wireless Sensor Networks,” *O’Reilly*, 2011.

URL: http://www.science.smith.edu/~jcardell/Courses/EGR328/Readings/XBeeCookbook.pdf

URL: http://code.google.com/p/xbee-api/

Lab Sheet ภาคที่ 3

แนะนำการสร้างเครือข่ายเซนเซอร์ไร้สายด้วยโนด Telos Compatible

(Introduction to Building WSNs using Telos Compatible Node)

ผู้เขียนร่วมในบทนี้ - ดร.ชัชนันท์ จันแดง

จุดประสงค์การเรียนรู้

เพื่อให้สามารถใช้งานโนดที่ทำขึ้นเอง (In-house node) ที่รองรับการทำงานตามรูปแบบของโนด Telos

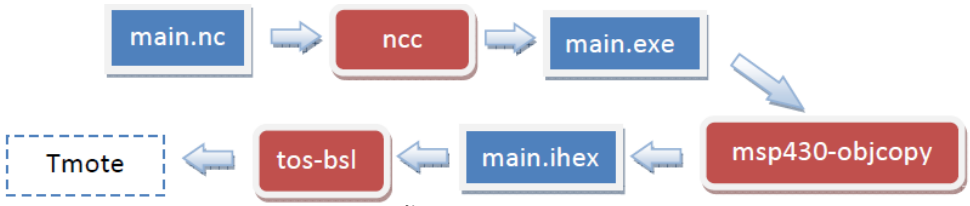
บทนี้เป็นการแนะนำการใช้งานโนดที่ได้พัฒนาขึ้นเองรองรับการทำงานตามรูปแบบของโนด Telos ของ Tmote Sky เพื่อลดการนำสั่งซื้อและนำเข้าจากต่างประเทศ โนดที่ผลิตขึ้นเองนี้ใช้ไมโครคอนโทรลเลอร์ของบริษัท TI MSP430F1611 มีหน่วยความจำชนิด RAM ขนาด 10 kB และหน่วยความจำชนิด ROM ขนาด 48 kB ใช้ชิปภาครับส่งคลื่นวิทยุ CC2420 โดยที่มีการต่อพ่วงโมดูลเซนเซอร์วัดแสง เซนเซอร์วัดอุณหภูมิและความชื้นในอากาศ เป็นต้น และต่อเพิ่มขยายหน่วยความจำชนิด Flash อีก 1 Mbit โนดจะใช้ระบบปฏิบัติการ TinyOS ซึ่งต้องการพื้นที่จัดเก็บเพียงประมาณ 400 ไบต์ (เฉพาะแกนของระบบปฏิบัติการ) ระบบปฏิบัติการ TinyOS เป็นระบบปฏิบัติการที่ทำงานแบบ multitasking และ concurrency ซึ่งหมายถึง ในช่วงเวลาหนึ่งระบบปฏิบัติการจะประมวลผลโพรเซสได้พร้อมกันหลายๆงาน โพรเซสของระบบปฏิบัติการแบ่งออกเป็นสองชนิด ได้แก่ task และ event

- task เป็นโพรเซสซึ่งมีหน้าที่สำหรับดำเนินงานต่างๆ จากชุดคำสั่งแรกจนกระทั่งคำสั่งสุดท้ายของงานนั้นๆ

- event เป็นโพรเซสที่ถูกสั่งงานโดยโมดูลหรือการอินเตอร์รัปป์ (interrupt) จากโมดูลต่างๆ โพรเซส

ชนิดนี้จะหยุดการทำงานของ task อื่นๆ หลังจากนั้นดำเนินการตามชุดคำสั่งในส่วนของของการอินเตอร์รัปป์จนจบการทำงาน และกลับไปทำงานตาม task เดิมที่เคยทำงานค้างไว้

สำหรับภาษาที่จะใช้ในการพัฒนาโปรแกรมบนโนดที่ใช้ระบบปฏิบัติการ TinyOS คือภาษา nesC เป็นภาษาที่ขยายมาจากภาษา C สำหรับขั้นตอนของการพัฒนาโปรแกรมด้วยภาษา nesC จะเริ่มจากไฟล์ของภาษา nesC (.ns) จากนั้นจะคอมไพล์ด้วยตัวแปลภาษา ncc ผลลัพธ์ที่ได้คือไฟล์ .exe ซึ่งเป็นไบนารีไฟล์ ซึ่งสามารถเปลี่ยนเป็นไฟล์ hex ได้ด้วย msp43-objcopy ผลลัพธ์ที่ได้คือไฟล์ .ihex จากนั้นสามารถดาวน์โหลดไฟล์ไปยังโนดได้ด้วย tos-bsl อย่างไรก็ตามกระบวนการตามภาพประกอบที่ L3.1 สามารถถูกดำเนินการด้วยชุดคำสั่งของ Tool-chain เพียงคำสั่งเดียว



ภาพประกอบที่ L3.1 ขั้นตอนการพัฒนาโปรแกรมด้วย nesC

L3.1 เตรียมพร้อมก่อนเริ่มปฏิบัติการ

สำหรับปฏิบัติการนี้เครื่องคอมพิวเตอร์จะใช้ระบบปฏิบัติการ Linux *Ubuntu*

1. ให้เริ่มทำการดาวน์โหลดไฟล์แพคเก็ตจากเว็ปไซต์ http://www.tinyos.net หรือ http://202.28.99.231/download/tinyos
2. ในกรณีที่ไฟล์ที่ดาวนโหลดมาเป็นนามสกุล .rpm จะต้องทำการแปลงให้อยู่ในรูปแบบของไฟล์ deb ด้วยโปรแกรม alien ก่อน

```
$ sudo apt-get install alien
$ sudo alien *.rpm
```

3. ติดตั้งโปรแกรมด้วยคำสั่ง dpkg

```
$ sudo dpkg -i msp430tools*.deb
$ sudo dpkg -i tinyos*.deb
$ sudo dpkg -i nesC*.deb
```

4. ปรับแต่งไฟล์ setup กันเล็กน้อย ด้วย gedit ดังต่อไปนี้

```
$ gedit ~/.bashrc
```

เติมเนื้อความในไฟล์ .bashrc ดังต่อไปนี้

```
export TOSROOT="/opt/tinyos-2.x"
export TOSDIR="$TOSROOT/tos"
export MAKERULES="$TOSROOT/support/make/Makerules"
export PATH="/opt/msp430/bin:$PATH"
```

5. แล้วปรับแต่งระบบด้วยคำสั่งดังต่อไปนี้

```
$ sudo chown -R `whoami` /opt/tinyos-2.x
$ sudo apt-get install build-essential automake
$ cd /opt/tinyos-2.x/support/sdk/c
$ ./bootstrap
$ ./configure
$ make
```

แพคเก็ตต่างที่ของ TinyOs จะอยู่จัดเก็บไว้ในไดเร็กทอรีต่างๆดังแสดงในรายละเอียดต่อไปนี้

/apps ตัวอย่างโปรแกรมภาษา NesC

/support โปรแกรมภาษา C, C++, Java สำหรับสนับสนุนการทำงานของ TinyOS ซึ่งทำงานทางฝั่ง Server

/tools เครื่องมือสำหรับสนับสนุนการคอมไพล์และติดตั้งโปรแกรมบน Tmote

/tos Source code ของระบบปฏิบัติการ TinyOS

- /interface ไฟล์ interface ของระบบปฏิบัติการ
- /lib ไลบรารีมาตรฐานของระบบปฏิบัติการ
- /platform ไลบรารีสำหรับแต่ละแพลตฟอร์ม เช่น telosb, micaz เป็นต้น
- /sensorboards ไลบรารีสำหรับ sensor แต่ละชนิด
- /system ไลบรารีสำหรับ kernel ของระบบปฏิบัติการ
- /types ไฟล์ Header สำหรับอธิบายชนิดของข้อมูลใน TinyOS
- /chips ไลบรารีสำหรับ chip ต่างๆ เช่น CC2420 (โมดูลสำหรับ RF) sht11 (โมดูล sensor วัดความชื้นและอุณหภูมิ)

L3.2 การคอมไพล์และติดตั้ง TinyOS บนโนด

จากที่กล่าวมาแล้วก่อนหน้านี้ tool-chain ของ TinyOS ได้อำนวยความสะดวกในการคอมไฟล์โปรแกรม tool-chains ชุดนี้ได้ให้บริการคำสั่งที่สำคัญสองคำสั่งได้แก่คำสั่ง motelist และคำสั่ง make

motelist เป็นคำสั่งสำหรับตรวจสอบ tmote ที่ติดต่อกับเครื่องคอมพิวเตอร์ ดังภาพประกอบที่ L3.1 จากตัวอย่างนี้แสดงว่า มี Tmote จำนวนหนึ่งตัวเชื่อมต่อผ่านพอร์ต USB โดยเชื่อมต่อผ่าน */dev/ttyUSB0* ในขั้นตอนของการติดตั้งโปรแกรม ต้องระบุชื่อพอร์ตดังกล่าวเพื่อระบุ Tmote ปลายทางในกรณีที่มี Tmote มากกว่าหนึ่งตัว

ภาพประกอบที่ L3.1 ผลจากการใช้คำสั่ง motelist

make เป็นเครื่องมือสำหรับช่วยในการคอมไฟล์โปรแกรม ก่อนที่จะคอมไพล์โปรแกรมด้วยคำสั่ง make ผู้พัฒนาระบบต้องสร้าง makefile เพื่อใช้สำหรับอธิบายขั้นตอนการคอมไพล์โปรแกรมขึ้นมาก่อน รูปแบบของการคอมไพล์โปรแกรมระบุตามรูปแบบต่อไปนี้

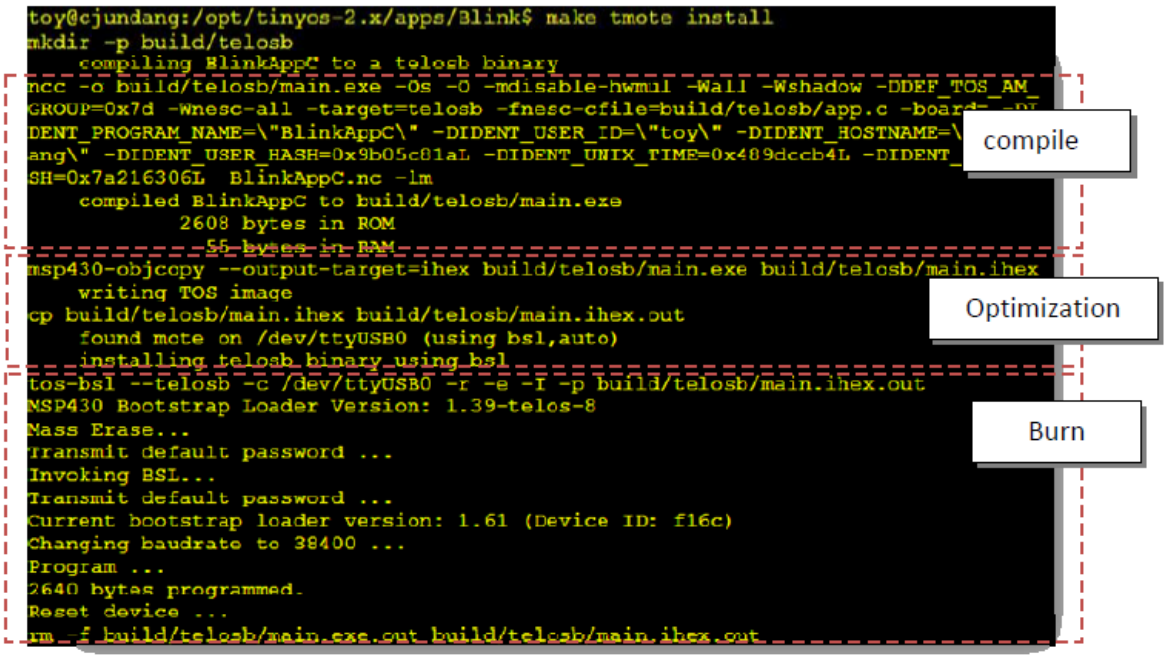
```
$ make <target> [extras]
```

<target> เป็นส่วนของการระบุแพลตฟอร์มเป้าหมายซึ่งได้แก่ tmote, telosb หรือ micaz เป็นต้น [extras] เป็นส่วนที่ระบบเพิ่มเติมในกรณีที่ต้องระบุพารามิเตอร์อื่นเพิ่มเติมซึ่งได้แก่

make tmote	คอมไฟล์โปรแกรมสำหรับ tmote
make tmote install	คอมไฟล์โปรแกรมสำหรับ tmote และติดตั้งโปรแกรมยัง tmote ตัวแรกที่ตรวจสอบพบ
make tmote reinstall	ติดตั้งโปรแกรมโดยที่ไม่ต้องคอมไพล์ใหม่
make tmote install,5	คอมไพล์โปรแกรมพร้อมกับติดตั้งโปรแกรมและระบุหมายเลขของ tmote
make tmote install bsl,/dev/ttyUSB1	คอมไฟล์โปรแกรมพร้อมกับติดตั้งโปรแกรมไปยัง tmote ที่ติดตั้งอยู่ที่พอร์ต /dev/ttyUSB1

ตัวอย่างของผลการคอมไพล์โปรแกรมและติดตั้งโปรแกรมแสดงดังภาพประกอบที่ L3.2

```
$ make tmote install
```



ภาพประกอบที่ L3.2 ผลการคอมไพล์และติดตั้งโปรแกรม

L3.3 การพัฒนาโปรแกรมด้วยภาษา nesC

โครงสร้างของภาษา

ภาษา nesC เป็นภาษาที่ขยายมาจากภาษาซี ดังนั้นโครงสร้างพื้นฐานของภาษาจะอิงกับภาษาซี ซึ่งได้แก่ ตัวแปร (variable) การดำเนินการ (operator) คำสั่งเงื่อนไข (condition expression) คำสั่งวนรอบ (loop expression) ชนิดข้อมูลขั้นสูง ได้แก่ ข้อมูลเชิงโครงสร้าง (struct) ข้อมูลแบบอาเรย์ (array) และข้อมูลแบบพอยเตอร์ (pointer) เป็นต้น

ชนิดข้อมูล

สำหรับชนิดข้อมูลพื้นฐานในภาษา nesC นั้น ไม่มีการกำหนดชนิดข้อมูลแบบ floating point, long หรือ char แต่ในภาษา nesC ได้ให้บริการชนิดข้อมูล integer ขนาด 8,16, 32 และ 64 บิตทั้งแบบคิดเครื่องหมายและไม่คิดเครื่องหมาย โดยกำหนดรูปแบบการประกาศข้อมูลเป็น uint8_t และ int8_t เพื่อแทนข้อมูลขนาด 8 แบบไม่คิดเครื่องหมายและคิดเครื่องหมายตามลำดับ

คำสั่งแบบเงื่อนไขและคำสั่งแบบวนรอบ

คำสั่งแบบเงื่อนไขซึ่งได้แก่ if, if...else สามารถใช้ได้บน nesC แต่คำสั่ง switch นั้นไม่สามารถได้งานได้ ในขณะที่คำสั่งวนลูปซึ่งได้แก่ for, while และ do...while นั้นสามารถให้งานได้ตามปกติ

ฟังก์ชัน

ภาษา nesC รองรับการทำงานด้วยฟังก์ชัน นอกจากนั้นยังนิยามรูปแบบของฟังก์ชั่นเพิ่มเติมขึ้นมาเพื่ออำนวยความสะดวก และรองรับคุณลักษณะพิเศษของภาษา nesC ดังต่อไปนี้

ฟังก์ชันแบบทั่วไป	ฟังก์ชันที่ถูกประกาศและเรียกใช้ภายในโมดูลซึ่งรูปแบบการทำงานเช่นเดียวกันกับภาษาซี
วิธีการประกาศ	*return_type functionname (argument list)*
วิธีการเรียกใช้	*ret_var = functionname(argument)*
ฟังก์ชันแบบ command	ฟังก์ชันที่ถูกประกาศและให้บริการโดย interface
วิธีการประกาศ	*command return_type functionname (argument list)*
วิธีการเรียกใช้	*rev_var = call functionname(argument)*
ฟังก์ชันแบบ event	ฟังก์ชันที่ถูกประกาศและให้บริการโดย interface แต่ผู้เรียกใช้ interface นั้นต้อง implement ส่วนของรายละเอียดของคำสั่งเอง
วิธีการประกาศ *event*	*return_type functionname (argument list)*
วิธีการเรียกใช้	ถูกเรียกใช้งานเองอัตโนมัติ เมื่อเกิด event ตามเงื่อนไขที่ถูกกำหนดไว้ในกรณีที่ต้องการเรียกใช้ฟังก์ชันนี้เองต้องเรียกใช้ผ่านคำสั่ง signal *signal fucntionname();*
ฟังก์ชันแบบ task	ฟังก์ชันพิเศษเมื่อถูกเรียกใช้งาน ระบบปฏิบัติการจะสร้าง task ใหม่ขึ้นมาและทำงานแบบ concurrent กับ task หลัก
วิธีการประกาศ *task*	*taskname (argument tlist)*
วิธีการเรียกใช้ *post*	*taskname (argument list)*

คอมโพแนนต์

จากที่กล่าวมาแล้วข้างต้นภาษา nesC ประกอบไปด้วยสองส่วนหลักได้แก่ Interface และ Component โดยที่ interface เป็นรูปแบบของการเชื่อมต่อในขณะที่ component เป็นส่วนหลักของการพัฒนาโปรแกรมด้วยภาษา nesC

- Interface รองรับการสื่อสารแบบสองทางซึ่งประกอบด้วย command และ event โดยที่

- Command เป็นฟังก์ชันที่ถูกเรียกใช้โดยคอมโพแนนต์เพื่อร้องขอบริการจากคอมโพแนนต์อื่น
- Event เป็นฟังก์ชันที่ถูกเรียกโดยคอมโพแนนต์ที่เรียกใช้ command จาก interface เดียวกัน ซึ่งถูกเรียกใช้โดยอัตโนมัติเมื่อมีเหตุการณ์ที่สัมพันธ์กับฟังก์ชันนั้นๆ

ตัวอย่างของโปรแกรมต่อไปนี้ เป็น Timer Interface

```
interface Timer
{
command void startPeriodic(uint32_t dt);
command void startOneShot(uint32_t dt);
command void stop();
event void fired();
……
}
```

interface timer ซึ่งใช้สำหรับควบคุมการทำงาน timer ในระบบปฏิบัติการ ภายใน interface ดังกล่าวประกอบไปด้วยฟังก์ชันสองประเภทได้แก่ command และ event โดยที่ *command void startPeriodic(...)* ใช้สำหรับการสั่งให้ timer เริ่มทำงานในขณะที่ *event void fired()* เป็นฟังก์ชันแบบ event จะถูกเรียกใช้อัตโนมัติ สำหรับงานที่จะให้ระบบปฏิบัติการทำงานนั้นผู้ใช้ต้องนิยามเพิ่มเติมเอง

Component เป็นส่วนประกอบที่สำคัญของโปรแกรม เนื่องจากคอมโพแนนต์มีหน้าที่สำหรับการกำหนดรายละเอียดการทำงานของโปรแกรม ซึ่งประกอบไปด้วยสองส่วนย่อย ได้แก่ module และ configuration แต่ละส่วนย่อยมีรายละเอียดดังนี้

Module เป็นไฟล์สำหรับการพัฒนาโมดูลใหม่ขึ้นมา แต่ละโมดูลนี้จะนิยามส่วนของการ use และ provide ส่วนของ interface ซึ่งได้กล่าวมาแล้วข้างต้น ตัวอย่างของ Module แสดงในลำดับต่อไป

```
module M{
provides interface A;
provides interface B;
uses interface C;
uses interface Boot;
}

implementation{
event void Boot.booted(){
/*… Implement something here*/
}
}
interface Boot{
event void Booted();
}
```

ตัวอย่างข้างต้นเป็นการสร้างโมดูลใหม่ชื่อ *M* ซึ่งประกอบไปด้วยสองส่วน ได้แก่ module และ implementation ส่วนของ module นั้นเป็นการอธิบายว่า โมดูลดังกล่าวให้บริการและเรียกใช้ interface ชนิดใดบ้าง จากตัวอย่างข้างต้น โมดูลดังกล่าวให้บริการ (provides) interface ชื่อ A, B และเรียกใช้ (uses) interface ชื่อ C และ Boot ส่วนที่สองของไฟล์โมดูลคือส่วนของ implementation ภายในส่วนนี้เป็นการอธิบายการทำงานของโมดูลด้วยไวยากรณ์ภาษาซีพื้นฐาน ภายใน interface Boot นั้นมีฟังก์ชันแบบ event ชื่อ booted ดังนั้นในส่วนของการ implementation ของโมดูล M จำเป็นต้องมีการ implement ฟังก์ชัน booted ไว้ด้วย โดยที่ฟังก์ชันดังกล่าวจะถูกเรียกใช่งานทันทีเมื่อระบบปฏิบัติการบูทเสร็จเรียบร้อยแล้ว

Configuration เป็นไฟล์สำหรับการอธิบายความสัมพันธ์ของโมดูลต่างๆ ในโปรแกรมผ่านการ wire ผ่าน interface ของแต่ละโมดูล ดังตัวอย่างต่อไปนี้

```
configuration TestAppC {
provides interface My_Interface;
}
implementation {
components MainC, TestM;
TestM -> MainC.Boot;
}
```

จากตัวอย่างข้างต้น ไฟล์ Configuration เป็นส่วนของการอธิบายถึงความสัมพันธ์ของโมดูล TestAppC โดยไฟล์ดังกล่าวแบ่งออกเป็นสองส่วน ได้แก่ configuration และ implementation ส่วนของ Configuration เป็นการอธิบายว่า โมดูลตัวนี้ได้ provide อะไรให้แก่โมดูลอื่น จะนิยามไว้ในกรณีที่โมดูลนี้ไม่ใช่โมดูลหลักของโปรเจคส่วน Implementation เป็นส่วนของการสร้างความสัมพันธ์ระหว่างโมดูลต่างๆ ผ่าน interface สำหรับโมดูลที่เป็นโมดูลหลักของโปรเจคนั้นต้องเรียกใช้คอมโพแนนต์ MainC เพื่อเป็นจุดเริ่มต้นของการทำงานของระบบโปรแกรมข้างต้น เรียกใช้คอมโพแนนต์สองตัวคือ MainC และ TestM โดยที่

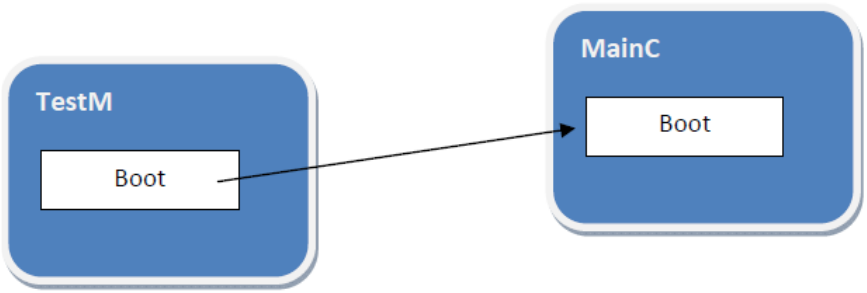
MainC เป็นคอมโพแนนต์หลักให้บริการ Interface Boot สำหรับการสั่งงานระบบปฏิบัติการ

TestM เป็นโมดูลที่ออกแบบมาเพื่อทำงานบางอย่าง ซึ่งในโมดูลนี้ต้องเรียกใช้ (uses) interface ชื่อ Boot เช่นกัน

```
TestAppC.nc
configuration TestAppC {
}
implementation {
components MainC, TestM;
TestM -> MainC.Boot;
}
```

```
TestM.nc
module TestM{
uses interface Boot;
}
implementation
{
event void Boot.booted(){
/*Implement here*/
}
}
```

โปรแกรม TestAppC เรียกใช้ MainC และ TestM โดยที่เชื่อมโยง interface Boot ของ TestM เข้ากันกับ Boot ของ MainC เนื่องจาก TestM ต้องการที่จะบูทระบบปฏิบัติการ แต่เนื่องจากในโมดูล TestM ไม่ได้ Implement ส่วนของกระบวนการบูทไว้ จึงจำเป็นที่จะต้องเรียกใช้กระบวนการบูทผ่าน interface Boot ของโมดูล MainC จากโครงสร้างของโปรแกรมข้างต้นอธิบายได้ด้วยภาพประกอบดังภาพประกอบที่ L3.3



ภาพประกอบที่ L3.3 โครงสร้างของ TestM.nc และ MainC.nc

คอมโพแนนต์ LedsC

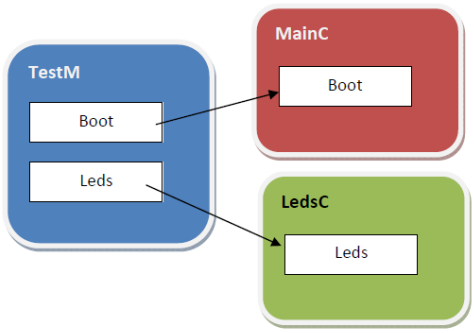
LedsC เป็นคอมโพแนนต์สำหรับควบคุมการแสดงผลของ Led ทั้งสามตัวบน tmote โดยแทน LED แต่ละตัวด้วย LED0, LED1 และ LED2 ผู้พัฒนาระบบสามารถเรียกใช้ฟังก์ชันผ่าน interface LedsC ซึ่งใน interface นี้ ประกอบไปด้วยฟังก์ชันที่ให้บริการและความหมายของแต่ละฟังก์ชันดังภาพประกอบที่ L3.4 และตัวอย่างของโปรแกรมควบคุม LED ดังภาพประกอบที่ L3.5

```
interface Leds {
async command void led0On();
async command void led0Off();
async command void led0Toggle();
async command void led1On();
async command void led1Off();
async command void led1Toggle();
async command void led2On();
async command void led2Off();
async command void led2Toggle();
async command uint8_t get();
async command void set(uint8_t val);
}
```

ฟังก์ชัน	ความหมาย
led0On()	สั่งเปิด LED
led0Off()	สั่งปิด LED
led0Toggle()	สั่งงานให้ LED กระพริบหนึ่งครั้ง
get()	อ่านสถานะของ LED
set(uint8_t)	กำหนดสถานะของ LED โดยการกำหนดค่าแบบ bitwise LEDS_LED0 หมายถึง LED0 LEDS_LED1 หมายถึง LED1 LEDS_LED2 หมายถึง LED2

ภาพประกอบที่ L3.4 คำอธิบายของฟังก์ชันที่เกี่ยวกับ LED

```
/*File: LedAppC.nc*/
configuration LedAppC {
}
implementation {
components MainC, LedM;
components LedsC;
LedM.Boot -> MainC.Boot;
LedM.Leds -> LedsC.Leds;
}
/*File: LedM.nc*/
module LedM{
uses interface Boot;
uses interface Leds;
}
implementation
{
void wait(uint16_t x){
while(x-- >0)
;
}
event void Boot.booted(){
call Leds.set(LEDS_LED1);
while(1){
call Leds.led0Toggle();
wait(0xFFFF);
}
}
}
```



ภาพประกอบที่ L3.5 ตัวอย่างโปรแกรมควบคุมการทำงานของ LED

คอมโพแนนต์ TimerC

TimerC เป็นคอมโพแนนต์สำหรับควบคุมการทำงานของตัวจับเวลา (Timer) ซึ่งสั่งงานผ่าน Interface ชื่อ TimerC ระบบปฏิบัติการ TinyOS ให้บริการตัวจับเวลาจำนวนสามแบบได้แก่ ตัวจับเวลาที่ความละเอียดหนึ่งในพันส่วน (TMilli) หนึ่งในสามหมื่นสองพันส่วน (T32khz) หนึ่งในล้านส่วน (TMicro) ฟังก์ชันที่ให้บริการผ่าน interface TimerC ที่น่าสนใจประกอบไปด้วย

```
interface Timer<precision_tag>
{
command void startPeriodic(uint32_t dt);
command void startOneShot(uint32_t dt);
command void stop();
event void fired();
}
```

เมื่อสังเกต Interface ข้างต้นจะพบว่ามีความแตกต่างจาก interface ก่อนหน้า เนื่องจาก TimerC เป็นแบบ Interface ที่ต้องการค่าพารามิเตอร์ interface ชนิดนี้ เป็น interface ที่รองรับการประมวลผลข้อมูลที่ต่างชนิดกัน (รูปแบบเดียวกันกับ template ในภาษา C++) ดังนั้น TimerC ต้องได้รับการกำหนดค่าพารามิเตอร์ให้แก่ interface ค่าพารามิเตอร์ที่ต้องการได้แก่ TMilli, T32khz และ TMicro เพื่อกำหนดความละเอียดของเวลาดังที่กล่าวมาแล้วข้างต้น การเรียกใช้ Interface TimerC แสดงดังตัวอย่างต่อไปนี้

```
#include "timer.h"
uses interface Timer<TMiil> as Timer0;
uses interface Timer<T32khz> as Timer1;
```

จากตัวอย่างข้างต้นได้เรียกใช้ interface timer จำนวนสองตัว โดยที่แต่ละตัวมีความละเอียดที่แตกต่างกันตามพารามิเตอร์ที่ระบบ นอกจากนี้คำสั่งข้างต้นได้เรียกใช้ คำสำคัญ(keyword) ชื่อ “as” ซึ่งใช้สำหรับการกำหนดชื่อของ interface เป็นชื่ออื่น

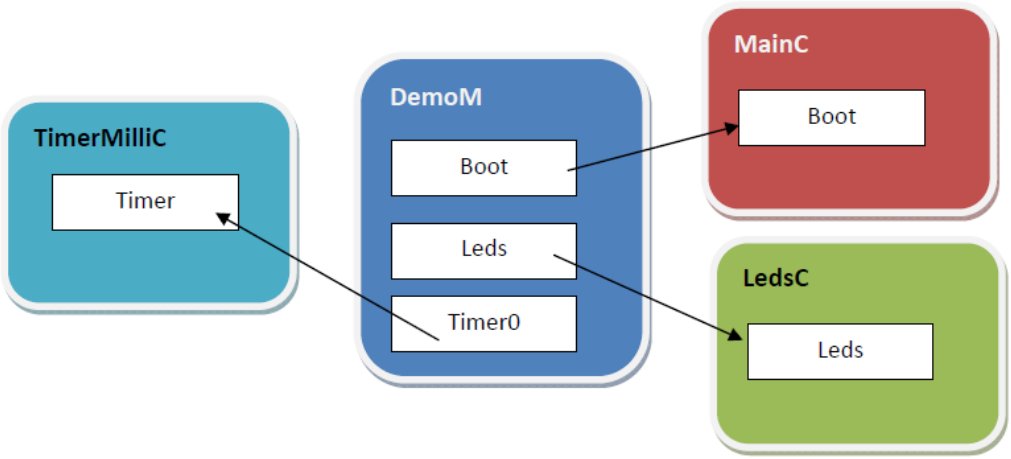
สำหรับระบบปฏิบัติการที่ติดตั้งใหม่นั้น รองรับการเรียกใช้เฉพาะพารามิเตอร์ชื่อ TMilli เท่านั้น สำหรับพารามิเตอร์อีกสองตัวนั้นยังไม่ได้รับการ implement ชุด คอมโพแนนต์ที่รองรับการทำงานของ Interface สิ่งที่ต้องพิจารณาเพิ่มเติมของการเรียกใช้ Interface คือ ในกรณีที่ interface ใดๆ มีการประกาศใช้ event function ผู้พัฒนาโปรแกรมต้องเขียนโปรแกรมเพื่ออธิบายส่วนของ body ของโปรแกรมเพื่อดำเนินการในกรณีที่เกิดเหตุการณ์ที่เป็นไปตามที่กำหนด มีการอธิบายการทำงานของฟังก์ชันที่เกี่ยวกับ Timer ดังภาพประกอบที่ L3.6

ฟังก์ชัน	ความหมาย
startPeriodic()	กำหนดให้ timer ทำงานทุกๆ เวลาที่กำหนดในพารามิเตอร์ ซึ่งในที่นี้หน่วยเป็น ms
startOneShot	กำหนดให้ timer ทำงานเพียงครั้งเดียว เมื่อครบกำหนดเวลาที่กำหนดในพารามิเตอร์ ซึ่งในที่นี้หน่วยเป็น ms
stop()	สั่งให้ timer หยุดการทำงาน
fired()	เป็นฟังก์ชันแบบ event โดยจะถูกเรียกใช้อัตโนมัติเมื่อ timer ทำงานครบตามเงื่อนไขจากการกำหนดโดยฟังก์ชัน startPeriodic() และ startOneShot

ภาพประกอบที่ L3.6 ฟังก์ชันที่เกี่ยวกับ Timer

โดยตัวอย่างของโปรแกรมควบคุมการทำงานของ LED ด้วย Timer ดังตัวอย่างด้านล่างและมีภาพโครงสร้างของโปรแกรมดังภาพประกอบที่ L3.7

```
/*File: TimerAppC.nc*/
configuration TimerAppC{
}
implementation{
components MainC, DemoM;
components LedsC;
components new TimerMilliC() as Timer;
DemoM.Boot -> MainC.Boot;
DemoM.Leds -> LedsC.Leds;
DemoM.Timer0 -> Timer.Timer;
}

/*File: DemoM.nc*/
module DemoM{
uses interface Boot;
uses interface Leds;
uses interface Timer<TMilli> as Timer0;
}
implementation
{
event void Boot.booted(){
call Timer0.startPeriodic(250);
}
event void Timer0.fired(){
call Leds.led0Toggle();
}
}
```



ภาพประกอบที่ L3.7 โครงสร้างของโปรแกรมควบคุม LED ด้วย Timer

จากไฟล์ TimerAppC.nc สังเกตได้ว่า คอมโพแนนต์ชื่อ TimerMilliC มีรูปแบบการเรียกใช้งานที่แตกต่างออกไป นั้นคือ ใช้คำสำคัญ “new” เข้าร่วมการประกาศค่า เนื่องจากคอมโพแนนต์ TimerMilliC ได้รับการนิยามที่แตกต่างจากคอมโพแนนต์ปกตินั่นคือ เป็นคอมโพแนนต์แบบ generic

โดยปกติ ในโปรแกรมเดียวกันเรียกใช้คอมโพแนนต์เดียวกันได้มากกว่า 1 ครั้ง แต่อย่างไรก็ตามแต่ละคอมโพแนนต์นั้นจะใช้ข้อมูลในคอมโพแนนต์เดียวกันหรือ share กันนั้นเอง แต่สำหรับคอมโพแนนต์ TimerMilliC แต่ละตัวที่เรียกใช้นั้น ใช้ตัวแปรสำหรับการเก็บตัวนับที่แตกต่างกัน จึงจำเป็นที่ต้องแตกตัวแปรภายในคอมโพแนนต์ออกจากกัน ดังนั้นเมื่อต้องการสร้างคอมโพแนนต์ที่ต้องการทำงานในรูปแบบดังกล่าวต้องระบุว่าคอมโพแนนต์ดังกล่าวเป็นแบบ generic และ เมื่อผู้ใช้นำคอมโพแนนต์ดังกล่าวไปใช้งานต้องเรียกใช้ด้วยคำสั่ง new ดังตัวอย่างต่อไปนี้

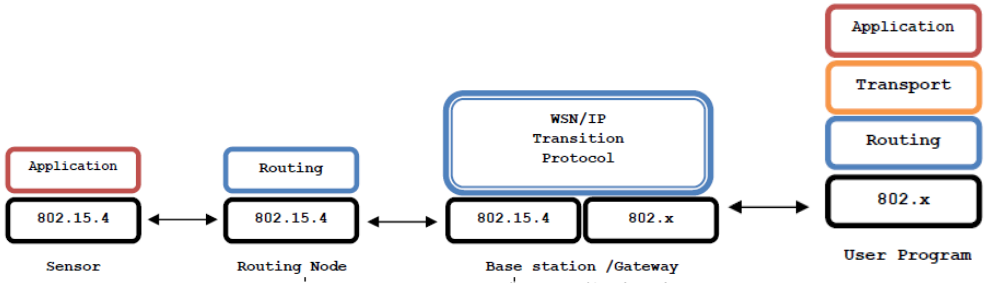
```
generaic configuration TimeMiilC() {
…
}
```

เรียกใช้งานคอมโพแนนต์ดังกล่าวด้วยคำสั่งต่อไปนี้

```
components new TimerMilliC() as Timer0;
```

L3.4 การรับส่งข้อมูลผ่าน CC2420 ด้วย nesC

โปรแกรมประยุกต์ของระบบเครือข่ายเซนเซอร์ไร้สายเป็นการทำงานร่วมกันกับเครือข่ายสัญญาณวิทยุกับเครือข่ายอินเตอร์เน็ตระบบแบบ IP รูปแบบการเชื่อมต่อของเครือข่ายอธิบายได้ด้วยโพรโทคอล stack ดังภาพประกอบที่ L3.8



ภาพประกอบที่ L3.8 รูปแบบการเชื่อมต่อด้วยโพรโทคอล stack

Sensor Node เป็นอุปกรณ์ที่ติดตั้งโปรแกรมสำหรับการอ่านข้อมูลจากเซนเซอร์หลังจากนั้นส่งข้อมูลดังกล่าวไปยัง base station ผ่านเครือข่าย 802.15.4 ซึ่งเป็นมาตรฐานที่กำลังใช้งานอยู่ในขณะนี้

Routing Node เป็นโนดที่ทำหน้าที่ส่งต่อข้อความเนื่องจาก ข้อจำกัดของสัญญาณวิทยุเป็นผลให้ข้อมูลถูกส่งไปได้ไม่ไกล ในกรณีที่ต้องการขยายขอบเขตของการติดตามข้อมูล ต้องอาศัยโนดอื่นมารับข้อมูลแล้วส่งข้อมูลไปยังปลายทาง ซึ่งเรียกโนดนั้นว่า routing node โดยโนดดังกล่าวต้องพัฒนาโปรแกรมสำหรับการค้นหาเส้นทางเพื่อที่สั้นและคุ้มค่าที่สุดในการเลือกเส้นทางในการรับส่งข้อมูล

Base station เป็นโนดหลักทำหน้าที่เชื่อมต่อเครือข่ายเซนเซอร์ไร้สายเข้ากันกับเครือข่ายที่ใช้งานเช่นเครือข่าย IP โนดตัวนี้ต้องมีสมรรถนะที่สูงเนื่องจากต้องการการคำนวณ หน่วยความจำและพลังงานที่สูงกว่าโนดอื่นๆ ในระบบ บนโนดชุดนี้ต้องพัฒนาโปรแกรมสำหรับการแปลงข้อความสำหรับเครือข่ายเซนเซอร์ไร้สายเป็นเครือข่าย IP เพื่อที่จะส่งข้อมูลนั้นไปยังโปรแกรมเครื่องของผู้ใช้ หรือระบบอื่นๆ ที่ต้องการใช้ข้อมูลดังกล่าว

User Program เป็นโปรแกรมสำหรับการอ่านและประมวลผลข้อมูลที่ได้จากเครือข่ายเซนเซอร์ไร้สายหลังจากที่แปลความข้อมูลดังกล่าวแล้วอาจจะจัดเก็บข้อมูลชุดนั้นไว้ในฐานข้อมูลหรือแสดงผลทางจอภาพในรูปแบบของข้อมูลดิบหรือกราฟซึ่งขึ้นอยู่กับวัตถุประสงค์ของผู้ใช้

หลังจากที่ติดตั้ง tool chain แล้วภายใน /opt/tinyos-2.x/support/sdk ได้ให้บริการโปรแกรมสำหรับ base station ซึ่งพัฒนาด้วยภาษาต่างๆ ได้แก่ ภาษา C, java, C++ หรือ python ซึ่งขึ้นอยู่ผู้พัฒนาโปรแกรมจะเลือกใช้ภาษาใดเป็นภาษาในการพัฒนาระบบ

ขั้นตอนของการสร้าง Test bed สำหรับการสื่อสารข้อมูลผ่านเครือข่าย

1. ต่อ Tmote ตัวที่ 1 เข้าไปยังพอร์ต USB ของติวเตอร์ ตรวจสอบชื่อพอร์ตที่ได้

$ motelist

2. เปลี่ยน working directory ไปยัง /opt/tinyos-2.x/apps/Basestation

$ cd /opt/tinyos-2.x/apps/Basestation

3. คอมไพล์โปรแกรม และติดตั้งโปรแกรมไปยัง Tmote

$ make tmote install,0x00 bsl,/dev/ttyUSB0

4. ต่อ Tmote ตัวที่ 2 เข้าไปยังพอร์ต USB ของติวเตอร์ ตรวจสอบชื่อพอร์ตที่ได้

$ motelist

5. เปลี่ยน working directory ไปยัง /opt/tinyos-2.x/apps/RadioCountToLeds

$ cd /opt/tinyos-2.x/apps/RadioCountToLeds

6. คอมไพล์โปรแกรม และติดตั้งโปรแกรมไปยัง Tmote ตัวที่สอง

$ make tmote install,0xFF bsl,/dev/ttyUSB1

7. เปลี่ยน working directory ไปยัง /opt/tinyos-2.x/support/sdk/c

$ cd /opt/tinyos-2.x/support/sdk/c

8. สั่งงานโปรแกรมเพื่ออ่านข้อมูลจาก Tmote วิธีการที่ 1

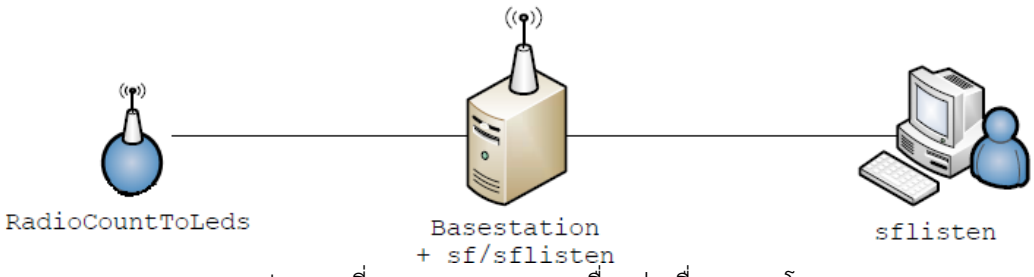
$./seriallisten /dev/ttyUSB0 tmote

9. สั่งงานโปรแกรมเพื่ออ่านข้อมูลจาก Tmote วิธีการที่ 2

$ ./sf 9000 /dev/ttyUSB0 tmote

เปิด Terminal ที่สอง

$ ./sflisten localhost 9000



ภาพประกอบที่ L3.9 ภาพแสดงการเชื่อมต่อเพื่อทดสอบโนด

ลองสังเกตจากภาพประกอบที่ L3.9 ว่า โปรแกรม RadioCountToLeds เป็นโปรแกรมสำหรับนับค่าตัวเลยแล้วส่งข้อมูลยังกล่าวมาที่ Base station ซึ่งมี Tmote เชื่อมต่ออยู่ผ่านพอร์ต USB บนเครื่อง Basestation มีการสั่งงานโปรแกรม seriallisten สำหรับอ่านค่า message จาก USB แล้วแสดงผลข้อมูลนั้นบนหน้า console หรือสั่งงานโปรแกรม sf เพื่ออ่านข้อมูลจาก USB แล้วเปิด พอร์ต TCP socket ไว้ หลังจากนั้นให้ผู้ใช้สั่งงานโปรแกรม sflisten ที่เครื่องผู้ใช้เพื่อแสดงข้อมูลบนหน้า console ของผู้ใช้

message_t

ตั้งแต่ระบบปฏิบัติการ TinyOS รุ่น 2.x ข้อความมาตรฐานสำหรับการรับส่งข้อมูลบนระบบเครือข่ายเซนเซอร์ไร้สายคือ message_t ซึ่งมีโครงสร้างของ message_t นั้นถูกนิยามไว้ที่ tos/types/message.h โดยมีโครงสร้างดังภาพประกอบที่ L3.10

```
typedef nx_struct message_t{
nx_unint8_t header [sizeof(message_header_t)];
nx_unint8_t data [sizeof(message_header_t)];
nx_unint8_t footer [sizeof(message_header_t)];
nx_unint8_t metadata[sizeof(message_header_t)];
}
```

ภาพประกอบที่ L3.10 แสดงโครงสร้างของข้อความใน TinyOS รุ่น 2.x

โครงสร้างของข้อความประกอบด้วยส่วนย่อยสี่ส่วนได้แก่ header, data. footer และ metadata โดยที่แต่ละส่วนมีรายละเอียดดังนี้

- *header* เป็นส่วนหัวของข้อความซึ่งประกอบไปด้วยโครงสร้างย่อย ๆ ขึ้นอยู่กับชนิดของสื่อที่ใช้ในการรับส่งข้อมูล ในที่นี้ได้แก่ CC2420 ซึ่งเป็น chip ใช้สำหรับควบคุมการทำงานของภาค RF หรือ serial เป็น

header ของข้อความที่ถูกใช้สำหรับการรับส่งข้อมูลผ่านพอร์ตอนุกรม ความยาวของ header นี้ขึ้นอยู่กับชนิดของสื่อที่ใช้ในการสื่อสารนั้นเอง

- data เป็นส่วนที่ถูกออกแบบมาโดยขึ้นอยู่กับชนิดของโปรแกรมประยุกต์ โครงสร้างดังกล่าวนั้นมีความยืดหยุ่นมากเมื่อเปรียบกันกับส่วนอื่นๆ ของข้อความ วิธีการสร้างส่วนของ data จะอธิบายในลำดับถัดไป

- footer เป็นส่วนท้ายของข้อความที่ถูกออกแบบมาเพื่อที่จะรองรับการทำงานอื่นๆ โครงสร้างส่วนนี้สามารถละได้

- metadata เป็นฟิลด์ที่ใช้สำหรับการเก็บข้อมูลของการสื่อสารข้อมูลหนึ่งฮอป (single hop communication) มักจะถูกใช้สำหรับการเก็บข้อมูลพื้นฐานของ Message ได้แก่ RSSI, LQI,CRC, ACK, กำลังส่ง, เวลาที่ได้รับ message โดยที่ข้อมูลเหล่านี้จะไม่ถูกส่งออกไป

โปรแกรมฝั่ง sensor node

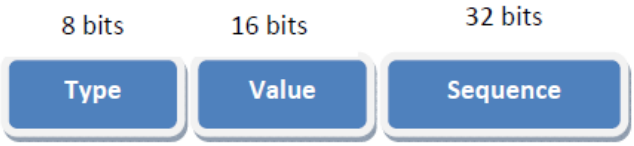
โปรแกรมฝั่งนี้เป็นโปรแกรมที่ใช้สำหรับในการอ่านข้อมูลจากเซนเซอร์หลังจากนั้นสร้าง 802.15.4 message TinyOS ให้บริการไลบรารีสำหรับการสร้างข้อความ รับ/ส่ง message กระบวนการอ่านข้อมูลจาก เซนเซอร์นั้นจะกล่าวในหัวข้อถัดไป สำหรับในหัวข้อนี้จะกล่าวถึงเฉพาะการสร้างข้อความและรับส่งข้อมูลผ่านเครือข่าย หลังจากนั้นแสดงผลข้อมูลบน console เท่านั้น

การสร้าง Payload สำหรับ TinyOS message

เมื่อโปรแกรมทางฝั่ง sensor node ต้องการส่งข้อมูลไปยังโนดอื่นๆ โนดดังกล่าวต้องออกแบบ message_t ในส่วนของ data เพื่อบรรจุโครงสร้างดังกล่าวเข้าสู่ message_t ดังตัวอย่างต่อไปนี้

```
// file: RadioCountToLeds.h
#ifndef RADIO_COUNT_TO_LEDS_H
#define RADIO_COUNT_TO_LEDS_H
typedef nx_struct radio_count_msg {
nx_uint16_t counter;
} radio_count_msg_t;
enum {
AM_RADIO_COUNT_MSG = 6,
};
#endif
```

ตัวอย่างข้างต้นเป็นไฟล์ header ของโปรแกรม RadioCountToLeds ซึ่งประกอบไปด้วยส่วนประกอบที่สำคัญได้แก่ ชื่อของ nx_struct,รายการของข้อมูลในข้อความ และหมายเลขข้อความ AM สำหรับข้อมูลในข้อความนั้นขึ้นอยู่กับโพรโทคอลที่ออกแบบมา ตัวอย่าง การสร้างข้อความสำหรับการส่งข้อมูลจากเซนเซอร์โนด โดยในโครงสร้างประกอบไปส่วนต่างๆดังภาพประกอบที่ L3.11




```
typedef nx_struct radio_data_msg {
nx_uint8_t type;
nx_uint16_t value;
nx_uint32_t seq;
} radio_count_msg_t;
```

ภาพประกอบที่ L3.11 โครงสร้างของข้อความเก็บข้อมูลจากเซนเซอร์

interface Receive และ AMSend

Interface ที่เกี่ยวข้องกับการเขียนโปรแกรมรับส่งข้อมูลผ่าน 802.15.4 ได้แก่ Receive และ AMSend โดยที่ interface สองตัวแรกนั้นใช้สำหรับการรับและส่งข้อความ ในขณะที่ interface อีกตัวนั้นใช้สำหรับการควบคุมการทำงานของโพรเซส

Interface ตัวนี้ใช้สำหรับการรับข้อความซึ่ง interface กล่าวทำงานร่วมกันกับคอมโพแนนท์ชื่อ AMReceiverC ซึ่งคอมโพแนนต์ดังกล่าวได้ provide interface Receive ไว้ ดังนั้น เมื่อโมดูลใดๆ เรียกใช้ interface Receive แล้ว จำเป็นที่จะต้องเรียกใช้ (wire) interface ดังกล่าวกับ AMReceiverC เสมอ ตามไวยากรณ์ต่อไปนี้ (กำหนดให้โมดูลที่เรียกใช้ interface ชื่อ DemoM และ message_type คือชนิดของข้อความ เช่น AM_RADIO_COUNT_MSG)

```
component DemoM
components AMReceiverC( message_type) ;
DemoM.Recieve -> AMReceiverC
```

ในขณะที่เมื่อมีการเรียกใช้ AMSend นั้นไฟล์ configuration ของโปรแกรมต้องเรียกใช้โมดูลชื่อ AMSenderC เพื่อเรียกใช้ฟังก์ชันการทำงานของ AMSender เช่นเดียวกันกับ Receive การเชื่อมโยงโมดูลระหว่าง interface สองตัวนี้คล้ายคลึงกันกับ Receive ได้แก่

```
component DemoM
components AMSenderC( message_type) ;
DemoM.AMSend -> AMSenderC
```

สำหรับ Interface Receive, AMSend และ Packet นั้นได้ให้บริการฟังก์ชั่นไว้ทั้งหมดดังแสดงไว้ด้านล่างและอธิบายความหมายไว้ในภาพประกอบที่ L3.12

```
interface Reciver{
event message_t * receive(message_t * msg,
void *payload,
uint8_t len);
command void getPayload(message_t * msg,
uint8_t len);
command uint8_t payloadLength(message_t *msg);
}
interface AMSend{
command error_t send(am_addr_t addr, message_t
* msg,
uint8_t len);
event void sendDone(message_t * msg, , error_t
error);
command uint8_t maxPayloadLength();
command void getPayload(message_t * msg);
}
interface Packet{
command void clear(message_t * msg);
command uint8_t payload(message_t * msg);
command void * getpayload(message_t * msg,
uint8_t len);
}
```

ฟังก์ชัน	ความหมาย
receive()	ฟังก์ชันแบบ event โดยจะถูกเรียกใช้อัตโนมัติเมื่อระบบปฏิบัติการอ่านข้อมูลในแต่ละข้อความจนจบ
getPayload()	ฟังก์ชันสำหรับการอ่าน Payload ออกจากข้อความโดยตัดส่วนของ Header ออกไป
payloadLength()	ฟังก์ชันสำหรับหาขนาดของ Payload
send()	ฟังก์ชันสำหรับส่งข้อความ
sendDone()	ฟังก์ชันแบบ event จะถูกเรียกใช้งานเมื่อกระบวนการส่งทำงานเสร็จสิ้น
maxPayloadLength()	ฟังก์ชันสำหรับอ่านขนาดของ Payload สูงสุดของโพรโทคอล

ภาพประกอบที่ L3.12 คำอธิบายฟังก์ชันสำหรับ Interface receive()

ในกรณีของการออกแบบและพัฒนาโปรแกรมสำหรับการเฝ้าระวังนั้นอาจจะไม่จำเป็นต้องเริ่มเขียนโปรแกรมจากไฟล์เปล่า เราอาจจะให้โปรแกรมตัวอย่างใน apps มาแก้ไขให้ตรงกับงานที่เราต้องการ แต่ถ้าหากต้องการพัฒนาโปรแกรมในส่วนของการค้นหาเส้นทางในชั้นเครือข่าย จำเป็นที่จะต้องศึกษาการเขียนโปรแกรมรับส่งข้อมูลให้มากขึ้น

สำหรับโปรแกรมบนเครื่อง Basestation ประกอบไปด้วยโปรแกรมสองส่วนได้แก่ โปรแกรม Basestation และโปรแกรม SerialForwarder

- Basestation เป็นโปรแกรมที่เขียนด้วยภาษา nesC ซึ่งใช้สำหรับรับข้อมูลจากสัญญาณวิทยุประกอบแต่ละไบต์เป็นข้อความและส่งข้อมูลทั้งหมดที่รับเข้ามาไปยังโปรแกรมระดับบนด้วยพอร์ตอนุกรม โปรแกรม Basestation นี้เป็นโปรแกรมพร้อมใช้ สามารถนำโปรแกรมดังกล่าวไปใช้งานได้ทันที

- SerialForwarder เป็นโปรแกรมที่พัฒนาด้วยภาษาระดับสูงเพื่ออ่าน message_t จากพอร์ตอนุกรมแล้วนำข้อมูลนั้นไปใช้งานต่อ เช่นแสดงผลทางจอภาพ หรือจัดเก็บข้อมูลนั้นในฐานข้อมูล

L3.5 การข้อมูลจากพอร์ตอนุกรม

การอ่านข้อมูลจากพอร์ตอนุกรมทำได้สองวิธีการคือการแสดงข้อมูลบน console หรืออ่านข้อมูลจากพอร์ตอนุกรมแล้วส่งต่อข้อมูลเหล่านั้นผ่านโพรโทคอล TCP/IP หลังจากนั้นเขียนโปรแกรมฝั่งเครื่องลูกข่ายเพื่อประมวลผลและแสดงหน้าจอ จากการพัฒนาโปรแกรมทั้งสองวิธีข้างต้นจะได้รับผลลัพธ์ที่หน้าจอเหมือนกันดังแสดงไว้ในตัวอย่างต่อไปนี้

ข้อความข้างต้นประกอบไปด้วยส่วนย่อยสองส่วนนั่นคือ ส่วนของ Header และส่วนของ data โดยส่วนของ Header นั้นถูกนิยามโดย struct ของ Serial และส่วนของ Data ถูกนิยามโดย struct ของ RadioCount-ToLed ซึ่งแต่ละโครงสร้างนั้นมีรายละเอียดดังนี้

serial_header

```
typedef nx_struct serial_header {
  nx_am_addr_t dest;
  nx_am_addr_t src;
  nx_uint8_t length;
  nx_am_group_t group;
  nx_am_id_t type;
} serial_header_t;
```

radio_count_msg

```
typedef nx_struct radio_count_msg
{
  nx_uint16_t counter;
} radio_count_msg_t;

enum {
  AM_RADIO_COUNT_MSG = 0xAB,
};
```

จากข้อมูลที่แสดงข้างต้นประกอบกับโครงสร้างดังต่อไปนี้อธิบายร่วมกันได้ว่า โครงสร้างส่วนแรก serial header เป็นส่วนหัวของพอร์ตอนุกรมซึ่งประกอบไปด้วยส่วนต่างๆ เมื่อจับคู่กันกับข้อมูลแล้วแสดงดังภาพประกอบที่ L3.13

ชื่อฟิลล์	ขนาด (บิต)	ค่า	แปลค่า
nx_am_addr_t dest	16	ff ff	ไม่ระบุฝ่ายรับ
nx_am_addr_t src	16	ff ff	ไม่ระบุฝ่ายส่ง
nx_uint8_t length	8	02	ความยาวของ data
nx_am_group_t group	8	00	ไม่ระบุกลุ่ม
nx_am_id_t type	8	ab	ข้อมูลแบบ AM_RADIO_COUNT_MSG

ภาพประกอบที่ L3.13 คำอธิบายโครงสร้างของ serial header

ในขณะที่ส่วนของ data นั้นประกอบไปด้วยข้อมูลขนาดสองไบต์นั่นคือ counter ซึ่งเมื่อคำนวณค่าที่ส่งมาจากเซนเซอร์โนด จะได้ค่าของ 0x02 0x14 ซึ่งเท่ากับ 2 (0x02) * 255 + 20 (0x14) = 530 เนื่องจากข้อความที่ถูกจัดส่งมานั้นจัดเรียงแบบ Most Significant Bits (MSB) หลังจากนั้นแสดงข้อมูลดังกล่าวทาง console

L3.6 การอ่านข้อมูลจากเซนเซอร์

เป้าหมายของระบบเครือข่ายเซนเซอร์ไร้สายคือการติดตามเฝ้าระวังค่าของสิ่งแวดล้อมรอบตัว บน Tmote มีเซนเซอร์ที่ติดมากับบอร์ดจำนวนสี่ชนิด ได้แก่ แรงดันไฟฟ้าของบอร์ด ความชื้น อุณหภูมิ ความเข้ม แสง พอร์ต IO พอร์ตอนุกรม โดยเซนเซอร์แต่ละตัวมีรายละเอียดดังนี้

แรงดันไฟ

MSP430 มีเซนเซอร์สำหรับการวัดแรงดันไฟและอุณหภูมิ โดยอ่านข้อมูลผ่าน ADC ของไมโครคอนโทรลเลอร์ โดยที่แรงดันไฟ (voltage port) เป็นการอ่านจากขา input ขนาด 12 บิตมาประมวลผลเพื่อที่จะหาค่าจากข้อมูลดิบที่ถูกส่งมาจากเซนเซอร์โนด

แรงดันฟ้าคำนวณได้จาก

*Voltage = RawData * 1.5 * 2 /4096*

เมื่ออ่านข้อมูลขนาด 12 บิตจากข้อความ นำมาเทียบอัตราส่วนกับข้อมูลที่เป็นไปได้ทั้งหมดนั่นคือ 4096 หลังจากนั้นคูณด้วยค่าความต่างศักย์ของไฟแบตเตอรี่นั้นคือ 1.5 * 2 เมื่อ 1.5 คือค่าความต่างศักย์ของแบตเตอรี่หนึ่งก้อน

ความชื้นและอุณหภูมิ

เซนเซอร์อุณหภูมิและความชื้นบน Tmote ผลิตจาก Sensirion AG ซึ่งใช้ชื่อรุ่น SHT11 และ SHt15 โมดูลดังกล่าวตรวจจับอุณหภูมิและความร้อนให้เป็นผลลัพธ์เป็นค่าดิจิตอล เซนเซอร์รุ่น SHT11 ให้ค่าที่แม่นยำกว่า SHT15 สำหรับบอร์ด Tmote ชุดนี้ได้ติดตั้ง SHT11 ไว้การแปลงข้อมูลดิบของอุณหภูมิจากสภาพแวดล้อมภายนอกระบบมีหน่วยเป็นองศาเซลเซียส มีสูตรคำนวณดังนี้

ExternalTemperture = -39.60 + (0.01 x RawData)

การแปลงข้อมูลดิบของความชื้นสัมพัทธ์มีหน่วยเป็นเปอร์เซ็นต์ต้องมีการคำนวณสองขั้นตอนนั่นคือ

*Humidity = -4 + (0.0405 x RawData) + (-2.8 x 1e-6 x RawData*RawData)*
Humidity(true) = ((ExternalTemperature – 25) x (0.01 + (0.00008 x RawData))) +
Humidity

ความเข้มแสง

การแปลงอุณหภูมิดิบของความสว่างของแสงมีอยู่สองแบบคือ TSR และ PAR โดยที่ TSR เป็นค่าความสว่างที่มนุษย์มองเห็นทั้งหมดรวมถึงแสงอินฟาเรด ในขณะที่ PAR เป็นค่าความสว่างแบบ Photosynthetically active radiation ซึ่งค่าความสว่างของแสงที่คำนวณได้มีหน่วยเป็นลักซ์ การคำนวณจะต้องมีการคำนวณค่ากระแสไฟที่ใช้ตามกฎของโอห์ม โดย

V = I R หรือ *I = V/R*

สูตรที่ใช้ในการคำนวณหากระแสไฟก่อนที่จะนำไปคำนวณหาค่าความสว่างต่อไป

I = Voltage / 1e5

นำค่า I ที่ได้ไปคำนวณค่าความสว่างของแสงจริงมีสูตรการคำนวณค่าความสว่างของแสงจริงมีสูตรการคำนวณทั้งหมดดังนี้

TSR = 0.769 x 1e5 x I x 1e3

PAR = 0.625 x 1e6 x I x 1e3

การเขียนโปรแกรมเพื่ออ่านข้อมูลจากเซนเซอร์

การเขียนโปรแกรมเพื่ออ่านข้อมูลจากเซนเซอร์ต้องกำหนดส่วนของโปรแกรมที่สำคัญสามส่วน นั้นคือการกำหนดให้โปรแกรมอ่านค่าจากเซนเซอร์ กำหนดส่วนของ uses ส่วนของ Interface Read สำหรับในไฟล์ configuration ต้อง wire interface Read กับคอมโพแนนต์ของโมดูลนั้นๆ ดังตารางต่อไปนี้

เซ็นเซอร์	Component	Wire
Internal Voltage	Msp430InternalVoltageC	*-> sensor
Internal Temperature	Msp430InternalTemperatureC	*-> sensor
External Temperature	SensirionSht11C	*-> sensor. Temperature
External Humidity	SensirionSht11C	*-> sensor. Humidity
Par Photo	HamamatsuS1087ParC	*-> sensor
Tsr Photo	HamamatsuS10871TsrC	*-> sensor

ตัวอย่างการเขียนโปรแกรม

```
configuration TestApp{
}
Implememtation {
component TestM as App;
component new Msp430InternalTemperatureC() as sensor;
…
App.Read -> sensor;
…
}
// file: TestM.nc
module TestM{
uses interface Read as sensor;
}
implementation {
event void sensor.Readdone(error_t error, uint16_t data){
}
}
```

ใบงานปฏิบัติการ

ปฏิบัติการที่ 1 การติดตั้ง tool-chain สำหรับ nesC

Hint: ถ้าหากนักศึกษาดาวน์โหลดแพ็คเก็จจากอินเตอร์เน็ตนั้นให้นักศึกษาติดตั้ง Tool-chain ตามลำดับ ตั้งแต่ข้อที่ 1 ถึงข้อที่ 5 แต่ถ้าหากติดตั้งโปรแกรมผ่าน CD Xbuntos 2.0 ระบบปฏิบัติการสามารถใช้งาน tool-chain ได้ทันที

ปฏิบัติการที่ 2 การฝึกใช้งานคำสั่ง motelist

เมื่อนักศึกษาบูทเครื่องคอมพิวเตอร์ด้วย CD Xbuntos เรียบร้อยแล้ว ให้นักศึกษาสั่งงานโปรแกรม Terminal ขึ้นมาหนึ่งโปรแกรมแล้วทำการพิมพ์คำสั่งดังต่อไปนี้แล้วกดปุ่ม enter ทีละคำสั่ง แล้วสังเกตผลการทดลอง

คำสั่ง	**ผล**
$ motelist	
เสียบ Tmote ทางพอร์ต USB แล้วพิมพ์ $ motelist	
เสียบ Tmote ตัวที่สอง ทางพอร์ต USB แล้วพิมพ์ $ motelist	
ดึง Tmote ตัวตัวแรกออก แล้วพิมพ์ $ motelist	

ปฏิบัติการที่ 3 การคอมไพล์โปรแกรมและติดตั้งโปรแกรมแบบต่างๆ

ขั้นตอนการคอมไพล์โปรแกรมและเขียนโปรแกรมบน Tmote นั้นมี 3 ขั้นตอน นั้นคือ compilation, optimization และ write to tmote

เมื่อนักศึกษาบูทเครื่องคอมพิวเตอร์ด้วย CD Xbuntos เรียบร้อยแล้ว ให้นักศึกษาสั่งงานโปรแกรม Terminal ขึ้นมาหนึ่งโปรแกรมแล้วหลังจากนั้นเสียบ Tmote จำนวนสองตัวทางพอร์ต USB หลังจากนั้นพิมพ์คำสั่งต่อไปนี้ สังเกตและตอบคำถามในตาราง

```
$ cd /opt/tinyos-2.x/apps/Blink [Enter]
```

พิมพ์คำสั่งต่อไปนี้	ผลการทำงาน		
	Compilation	Optimization	Write to Tmote
1. make tmote			
2. make tmote install			
3. make tmote reinstall			
4. make tmote install,1			
5. make tmote install,0xAB			
6. make tmote install bsl,/dev/ttyUSB0			
7. make tmote reinstall,5 bsl,/dev/ttyUSB1			

ปฏิบัติการที่ 4 การเขียนโปรแกรมภาษา nesC อย่างง่าย

เมื่อนักศึกษาบูทเครื่องคอมพิวเตอร์ด้วย CD Xbuntos เรียบร้อยแล้ว ให้นักศึกษาสั่งงานโปรแกรม Terminal ขึ้นมาหนึ่งโปรแกรมแล้ว ทำตามคำสั่งต่อไปนี้ ตามลำดับ

1. สร้างไดเร็กทอรีใหม่ ชื่อ Test

```
$ cd
$ mkdir Test
$ cd Test
```

2. สร้างไฟล์ด้วย editor ใดๆ ที่นักศึกษาถนัด เช่น vim, mousepad เป็นต้น แล้วคอมไพล์โปรแกรม

```
//file:Makefile
COMPONENT= TestAppC
$nclude $(MAKERULES)
//file: TestAppC.nc
configuration TestAppC {
}
implementation {
components MainC, TestM;
TestM -> MainC.Boot;
}
//file:TestM.nc
module TestM{
uses interface Boot;
}
implementation
{
event void Boot.booted(){
/*Implement here*/
}
}
```

ปฏิบัติการที่ 5 การเขียนโปรแกรมเพื่อควบคุม Led

เมื่อนักศึกษาบูทเครื่องคอมพิวเตอร์ด้วย CD Xbuntos เรียบร้อยแล้ว ให้นักศึกษาสั่งงานโปรแกรม Terminal ขึ้นมาหนึ่งโปรแกรมแล้ว ทำตามคำสั่งต่อไปนี้ ตามลำดับ

1. สร้างไดเร็กทอรีใหม่ ชื่อ Led

```
$ cd
$ mkdir Led
$ cd led
```

2. สร้างไฟล์ด้วย editor ใดๆ ที่นักศึกษาถนัด เช่น vim, mousepad เป็นต้น

```
//file:Makefile
COMPONENT= LedAppC
$include $(MAKERULES)
//file: LedAppC.nc
configuration LedAppC{
}
implementation{
}
//file:LedM.nc
module LedM{
}
implementation{
}
```

3. จงแก้ไขโปรแกรมข้างต้นเพื่อให้ Led ดวงที่ 1 สว่างเมื่อระบบปฏิบัติการ บูทเสร็จเรียบร้อยแล้ว
4. จงแก้ไขโปรแกรมข้างต้นเพื่อให้ Led ดวงที่ 2 กระพริบทุกๆ ช่วงเวลาหนึ่งๆ
5. จงเขียนโปรแกรมนับวนรอบ ตั้งแต่ 0 – 7 โดยแสดงผลทาง Led ทั้งสามดวง

ปฏิบัติการที่ 6 การเขียนโปรแกรมเพื่อควบคุม Timer

เมื่อนักศึกษาบูทเครื่องคอมพิวเตอร์ด้วย CD Xbuntos เรียบร้อยแล้ว ให้นักศึกษาสั่งงานโปรแกรม Terminal ขึ้นมาหนึ่งโปรแกรมแล้ว ทำตามคำสั่งต่อไปนี้ ตามลำดับ

1. สร้างไดเร็กทอรีใหม่ ชื่อ Time

$ cd

$ mkdir Timer

$ cd Timer

2. สร้างไฟล์ด้วย editor ใดๆ ที่นักศึกษาถนัด เช่น vim, mousepad เป็นต้น

```
//file:Makefile
COMPONENT=DemoTimerAppC
$include $(MAKERULES)
//file: DemoLedAppC.nc
configuration DemoTimerAppC{ ... }
implementation{ ... }
//file:DemoLedM.nc
module DemoTimerM{ ... }
implementation{ ... }
```

3. ให้นักศึกษาแก้ไขโปรแกรมข้างต้นเพื่อให้ Led 3 สว่าง หลังจากบูทระบบเสร็จแล้ว 1 s โดยใช้ Timer (Tips ให้เลือกใช้ ฟังก์ชัน startOneShot())

4. ให้นักศึกษาแก้ไขโปรแกรมในข้อที่ 3 เพื่อให้ Led กระพริบทุกๆ 500 ms โดยใช้ Timer (Tips ให้เลือกใช้ ฟังก์ชัน startOneShot())

5. ให้นักศึกษาเขียนโปรแกรมเพื่อให้ Led กระพริบทุกๆ 1 s โดยใช้ Timer

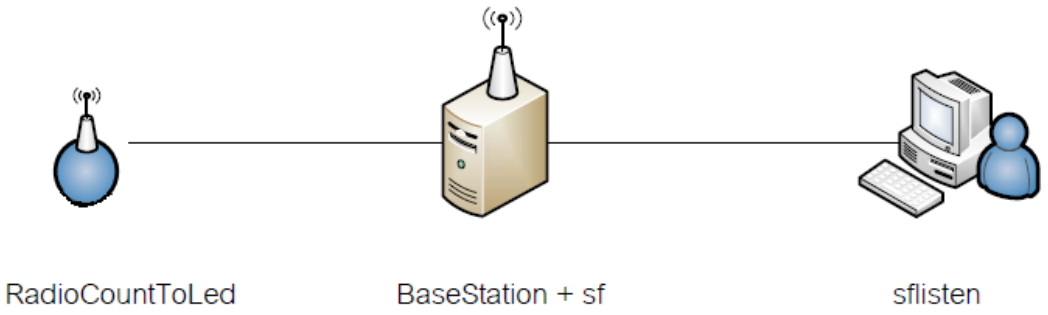
6. จงเขียนโปรแกรมนับวนรอบ ตั้งแต่ 0 – 7 โดยแสดงผลทาง Led ทั้งสามดวง โดยใช้ Timer

7. จงเขียนโปรแกรมเพื่อให้ Led 1 กระพริบทุกๆ 500 ms และ Led 2 กระพริบทุกๆ 700 ms

ปฏิบัติการที่ 7 ทบทวนการเขียนโปรแกรม

ให้นักศึกษาเขียนโปรแกรมภาษา nesC เพื่อให้ Led ทั้งสามดวงกระพริบพร้อมกัน โดยกำหนดให้ความถี่ในการกระพริบเริ่มต้นที่ 5 วินาที และกระพริบถี่ขึ้นเป็นครึ่งหนึ่งของเวลาก่อนหน้า จนกระทั่งเวลามีค่าน้อยกว่า 100 มิลลิวินาที ให้ Led ทั้งสามดวงดับ

ปฏิบัติการที่ 8 การ setup test bed สำหรับการทดสอบการสื่อสารข้อมูลผ่านเครือข่ายบนเซนเซอร์ไร้สาย



ขั้นตอนของการสร้าง Test bed สำหรับการสื่อสารข้อมูลผ่านเครือข่าย

1. ต่อ Tmote ตัวที่ 1 เข้าไปยังพอร์ต USB ของคอมพิวเตอร์ ตรวจสอบชื่อพอร์ตที่ได้

$ motelist

2. เปลี่ยน working directory ไปยัง /opt/tinyos-2.x/apps/Basestation

$ cd /opt/tinyos-2.x/apps/Basestation

3. คอมไพล์โปรแกรม และติดตั้งโปรแกรมไปยัง Tmote ที่เชื่อมต่ออยู่ตามพอร์ตในข้อที่ 1

$ make tmote install,0x00 bsl,/dev/ttyUSB0

4. ต่อ Tmote ตัวที่ 2 เข้าไปยังพอร์ต USB ของติวเตอร์ ตรวจสอบชื่อพอร์ตที่ได้

$ motelist

5. เปลี่ยน working directory ไปยัง /opt/tinyos-2.x/apps/RadioCountToLeds

$ cd /opt/tinyos-2.x/apps/RadioCountToLeds

6. คอมไพล์โปรแกรม และติดตั้งโปรแกรมไปยัง Tmote ตัวที่สอง ที่เชื่อมต่ออยู่ตามพอร์ตในข้อที่ 4 และกำหนด Node ID ตามที่กำหนดให้ ในที่นี้กำหนดเป็น 0xFF

$ make tmote install,0xFF bsl,/dev/ttyUSB1

7. เปลี่ยน working directory ไปยัง /opt/tinyos-2.x/support/sdk/c

$ cd /opt/tinyos-2.x/support/sdk/c

8. สั่งงานโปรแกรมเพื่ออ่านข้อมูลจาก Tmote วิธีการที่ 1

$./seriallisten /dev/ttyUSB0 tmote

อธิบายผลของการทำงานที่เกิดขึ้น

ทำการทดลองตั้งแต่ข้อที่ 1 ใหม่อีกครั้งโดย แก้ไขคำสั่ง ในข้อที่ 3 เป็น

$ CC2420_CHANNEL=xx make tmote install,0 bsl,/dev/ttyUSB0

และข้อที่ 6 เป็น

$ CC2420_CHANNEL=xx make tmote install,yy bsl,/dev/ttyUSB1

เมื่อ xx คือหมายเลขช่องสัญญาณ และ yy คือหมายเลขโนดฝ่ายส่ง

ปฏิบัติการที่ 9 การออกแบบและการแปลค่าจากข้อมูลใน Payload

ให้นักศึกษาออกแบบและพัฒนาโปรแกรม โดยมีคุณสมบัติดังต่อไปนี้

1. Payload บรรจุหมายเลขโนด และ Counter โดยกำหนดให้ หมายเลขโนดอยู่ในลำดับแรก
2. คอมไพล์โปรแกรมโดยกำหนดให้ใช้ Default CHANNEL
3. แก้ไขโปรแกรม serialisten โดยแสดงเฉพาะค่า Counter ที่ได้รับการแปลผลแล้วเท่านั้น และต้องเป็นค่าที่มีจากหมายเลยโนดที่กำหนดให้เท่านั้น

ปฏิบัติการที่ 10 การเขียนโปรแกรมเพื่ออ่านค่าจากเซนเซอร์

ให้นักศึกษาออกแบบและพัฒนาโปรแกรม โดยที่ ออกแบบ Payload โดยบรรจุหมายเลขโนด และข้อมูลดิบจากเซนเซอร์ ซึ่งได้แก่ Internal Voltage Temperature Humidity และ TSR โดยแต่ละข้อความต้องบรรจุข้อมูลของเซนเซอร์ทุกชนิด