บทที่ 5

สรุปผลการวิจัยและข้อเสนอแนะ

หลังจากที่ได้ทำการทดสอบและเสนอผลการทดสอบในบทที่แล้ว บทนี้จะกล่าวถึงบทสรุปเกี่ยวกับงานวิจัย ปัญหาและอุปสรรค วิจารณ์และข้อเสนอแนะสำหรับการออกแบบและทดสอบระบบปฏิบัติการเชิงเวลาจริง uC/OS-II ที่ตระหนักถึงกำลังงานที่ใช้เพื่อเป็นประโยชน์สำหรับผู้ที่ต้องการศึกษาและทำการวิจัยที่เกี่ยวข้องต่อไป

5.1 สรุปผล

จากการศึกษาและทดสอบการลดกำลังงานที่ใช้ของระบบปฏิบัติการเชิงเวลาจริง uC/OS-II ที่ทำงานบนหน่วยประมวลผล ARM-7 เทคนิคการลดการใช้พลังงานสามารถทำได้โดยการปรับเปลี่ยนสถานะในการทำงานของหน่วยประมวลผลให้อยู่ในสถานะ idle หรือ power down เมื่อไม่มีงานใด ๆ ต้องการใช้หน่วยประมวลผลกลาง สำหรับการทำงานบนระบบเชิงเวลาจริงการควบคุมพลังงานโดยใช้สถานะ idle จะมีความน่าเชื่อถือมากกว่าสถานะ power down แต่ลดการใช้กำลังงานได้น้อยกว่าสถานะ power down โดยลดการใช้พลังงานลงเหลือเพียง 37.93% เมื่อทดสอบด้วยแอพลิเคชัน Modbus ดังนั้นการปรับระบบให้คำนึงถึงกำลังงานที่ใช้ในระบบเชิงเวลาจริงต้องหาจุดที่เหมาะสมที่สุดระหว่างกำลังงานที่ใช้และความน่าเชื่อถือของระบบ เนื่องจากแอพลิเคชันแต่ละชนิดมีความต้องการในการตอบสนองเชิงเวลาจริงของระบบที่แตกต่างกัน

จากการวิเคราะห์ตัวโปรแกรมของระบบปฏิบัติการเชิงเวลาจริง uC/OS-II และจากการวัดกิจกรรมของหน่วยประมวลผลกลางในสภาวะการใช้งานจริงพบว่า uC/OS-II จะดำเนินการให้งานทุกงานได้ใช้หน่วยประมวลผลได้ครบถ้วนตามความต้องการก่อนที่ระบบปฏิบัติการจะเข้าสู่ idle task นั่นหมายความว่างานทุกงานจะรันไปตามลำดับโดยไม่มีการหยุดและรันอย่างต่อเนื่องจนกระทั่งเข้าสู่ idle task เช่นเดียวกับที่ผู้วิจัยได้ออกแบบไว้ ปัจจัยการสูญเสียกำลังงานอันเนื่องจากการจัดตารางงานที่ไม่เหมาะสมจึงไม่เกิดในระบบปฏิบัติการ uC/OS-II ดังนั้นจึงไม่มีความจำเป็นต้องจัดตารางงานใหม่

จากการทดสอบปรับเปลี่ยนระบบปฏิบัติการ uC/OS-II ในส่วนของ Timer tick โดยปรับช่วงระยะเวลาในการ tick ที่ต่างกันเพื่อพิจารณากำลังงานที่ใช้เปรียบเทียบกับความถี่ของ

timer tick พบว่ากำลังงานที่ใช้เปลี่ยนแปลงเมื่อปรับให้ระบบทำงานในสถานะ idle ในขณะที่รัน idle task พลังงานที่ใช้จะเพิ่มสูงขึ้นถ้าความถี่ของ timer tick มีค่าสูง โดยเฉพาะถ้าปรับให้เกิด timer tick ที่ความถี่สูงมากจะทำให้ระบบใช้พลังงานเต็มที่โดยไม่มีโอกาสได้รัน idle task หรือเข้าสู่สถานะ idle ค่า timer tick ที่เหมาะสมขึ้นอยู่กับความต้องการในการตอบสนองของระบบ สำหรับความถี่ที่ต่ำที่สุดที่ยังสามารถตอบสนองเชิงเวลาจริงได้ดีและตระหนักถึงกำลังงานที่ใช้ สำหรับงานวิจัยนี้มีค่าเป็น 1 ms เพราะถ้าปรับให้ความถี่ต่ำกว่าค่านี้ระบบก็ยังคงใช้พลังงานเท่าเดิมแต่อาจส่งผลกระทบต่อการตอบสนองเชิงเวลาจริง

5.2 ปัญหาและอุปสรรค

5.2.1 โปรแกรม comDebug ซึ่งเป็นเครื่องมือสำหรับรับส่งข้อมูลด้วย Modbus protocol ติดตั้งทางฝ่าย master มีความผิดพลาดที่เกิดจากโปรแกรมไม่รองรับข้อมูลที่เป็นศูนย์ (0x00) ทำให้เข้าใจผิดเกี่ยวกับความถูกต้องในการทำงานของ Modbus ทางฝ่าย slave ดังนั้นจึงพัฒนาโปรแกรมเพื่อใช้ในการทดสอบทางฝ่าย master ด้วยโปรแกรม Microsoft Visual Basic 6.0 ตรวจสอบความถูกต้องของโปรแกรมที่พัฒนาด้วยโปรแกรม Modscan ซึ่งมีมาตรฐานและทำงานได้อย่างถูกต้อง

5.2.2 ตัวแปลภาษา IAR ต้องลงทะเบียนทุก 30 วัน เนื่องจากเป็นเวอร์ชันทดลองใช้

5.2.3 Oscilloscope ที่ใช้ในงานวิจัยสามารถรับสัญญาณได้เพียง 2 ช่องสัญญาณเท่านั้นทำให้ไม่สามารถเปรียบเทียบลักษณะสัญญาณได้มากกว่า 2

5.3 วิจารณ์และข้อเสนอแนะ

5.3.1 การนำเทคนิคในการควบคุมการใช้กำลังงานไปใช้งานต้องพิจารณาความน่าเชื่อถือที่ระบบยอมรับได้ร่วมกับกำลังงานที่ต้องการลด ถ้าไม่วิเคราะห์ความต้องการของระบบให้ชัดเจนอาจก่อให้เกิดความเสียหายร้ายแรงได้

5.3.2 แอพลิเคชัน Modbus ที่พัฒนาในงานวิจัยนี้ไม่สมบูรณ์ครบทุกฟังก์ชัน เนื่องจากข้อจำกัดทางด้านเวลาจึงพัฒนาให้มีเฉพาะฟังก์ชันที่รองรับการทดสอบเชิงเวลาจริงสำหรับงานวิจัยเท่านั้น ดังนั้นถ้าต้องการนำ Modbus ดังกล่าวไปใช้งานจริงต้องพัฒนาเพิ่มเติมให้มีความสมบูรณ์มากยิ่งขึ้น

5.3.3 โพรโตคอล Modbus เป็นโพรโตคอลที่ยังไม่สนับสนุนการทำงานเชิงเวลาจริงที่ตระหนักถึงกำลังงานที่ใช้เนื่องจากลักษณะการทำงานจะถูกควบคุมด้วยการส่งคำถามมาจาก master ดังนั้นการออกแบบโพรโตคอล Modbus ที่ตระหนักถึงกำลังงานที่ใช้ควรให้ slave มีส่วนร่วมในการควบคุมการทำงานนั่นคือเมื่อ slave ตื่นจากสถานะ power down ก็ทำการส่งสัญญาณไปบอก master ให้สามารถส่งคำร้องขอมาได้ (slave จะต้องรอด้วยระยะเวลาไม่ต่ำกว่าระยะเวลาที่ใช้ในการส่งข้อมูลไปกลับ) จากนั้น slave จึงจะเข้าสู่สถานะ power down ได้ สำหรับกรณีที่ master ต้องการร้องขอข้อมูลหรือสั่งให้ slave ทำงานตามที่ขอ slave จะต้องทำงานดังกล่าวให้เสร็จก่อนที่จะเข้าสู่สถานะ power down

5.3.4 การออกแบบโพรเซสเซอร์ที่ตระหนักถึงกำลังงานที่ใช้สำหรับแอพลิเคชันที่มีการรับส่งข้อมูลผ่าน UART ไม่ควรปิดการทำงานของ UART เมื่อระบบเข้าสู่สถานะ power down เพื่อให้ระบบสามารถรองรับการทำงานได้ตลอดเวลา และลดปริมาณกำลังงานที่ได้ได้มากโดยไม่ส่งผลกระทบต่อการทำงานเชิงเวลาจริง

5.3.5 แนวทางการแก้ไขดัดแปลงระบบปฏิบัติการสำหรับงานวิจัยนี้สามารถนำไปประยุกต์ใช้กับระบบปฏิบัติการอื่น ๆ ได้โดยมีเงื่อนไขคือไมโครโพรเซสเซอร์หรือฮาร์ดแวร์ที่ใช้ต้องรองรับการลดกำลังงานเช่นเดียวกับ idle mode หรือ power down mode ใน ARM-7 และระบบปฏิบัติการนั้นต้องมีโปรแกรมต้นฉบับเพื่อที่จะสามารถแก้ไขส่วนของ idle task ให้รองรับการตระหนักถึงกำลังงานที่ใช้ได้ ตัวอย่างเช่น ถ้าต้องการนำแนวทางนี้ไปใช้สำหรับระบบปฏิบัติการ Linux ที่ทำงานบนเครื่องคอมพิวเตอร์ส่วนบุคคลสามารถทำได้โดยการแก้ไขต้นฉบับโปรแกรมใน idle task (ฟังก์ชัน default_idle ซึ่งอยู่ในไฟล์ process.c) ให้รองรับการลดกำลังงานในขณะที่ไม่มีงานที่ให้บริการ โดยผ่านกระบวนการของ ACPI (Advanced Configuration and Power Interface)