****

**ภาควิชาวิศวกรรมคอมพิวเตอร์**

**คณะวิศวกรรมศาสตร์**

242-302 ADV COM ENG LAB II

**3HB05,3HB06 AVR Microcontroller**

เสนอ

**อาจารย์ พัชรี เทพนิมิตร**

จัดทำโดย

**นายอรรฆพร ศรีปานรอด**

ID student **5635512057** section **02**

**Prince of Songkla University Phuket Campus**

**Year 2558**

**3HB05,3HB06 AVR Microcontroller**

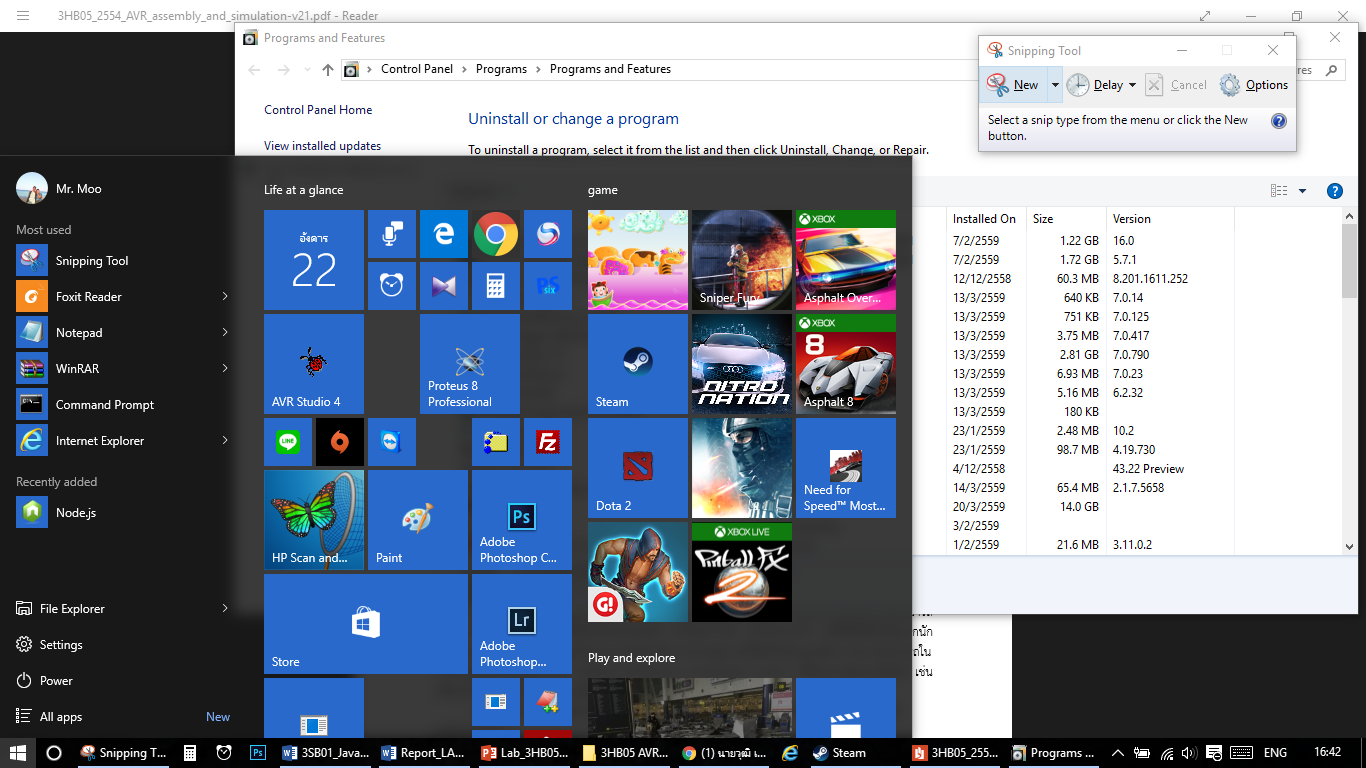
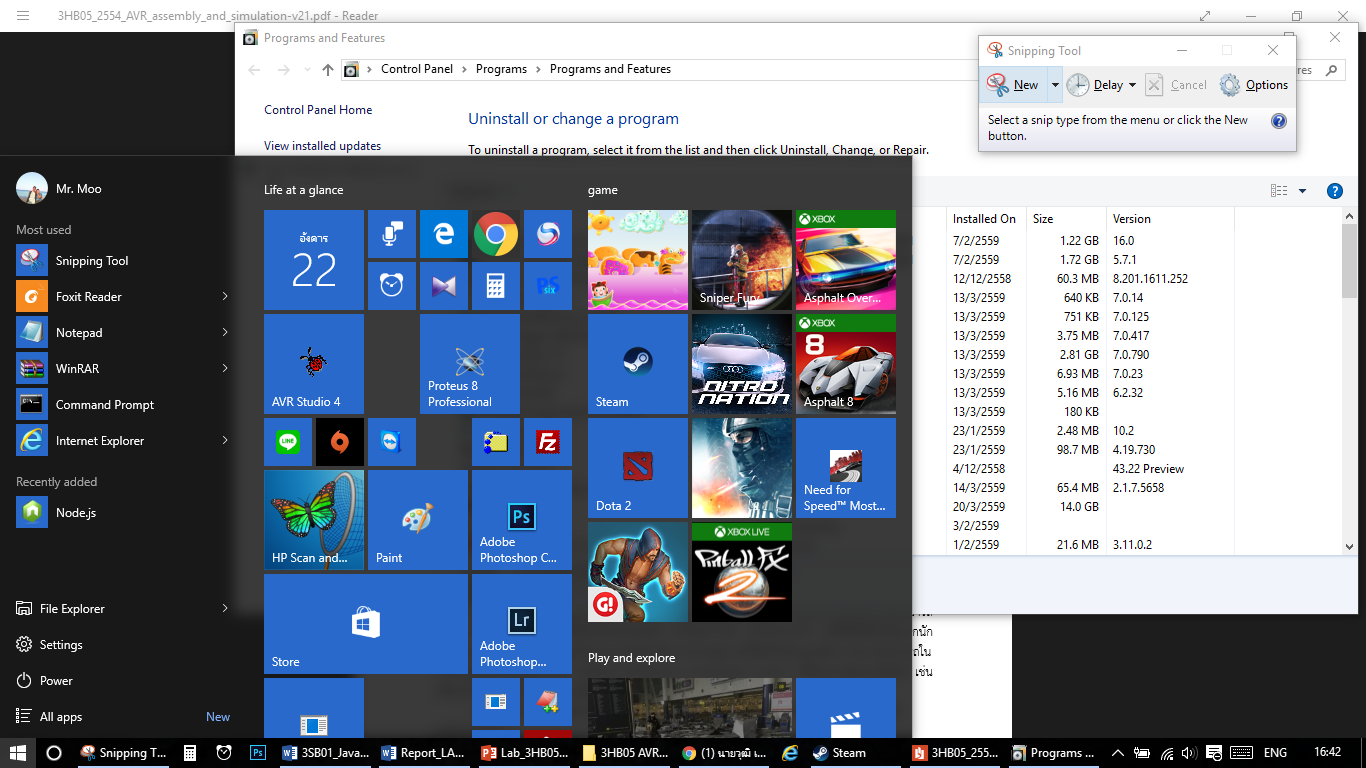
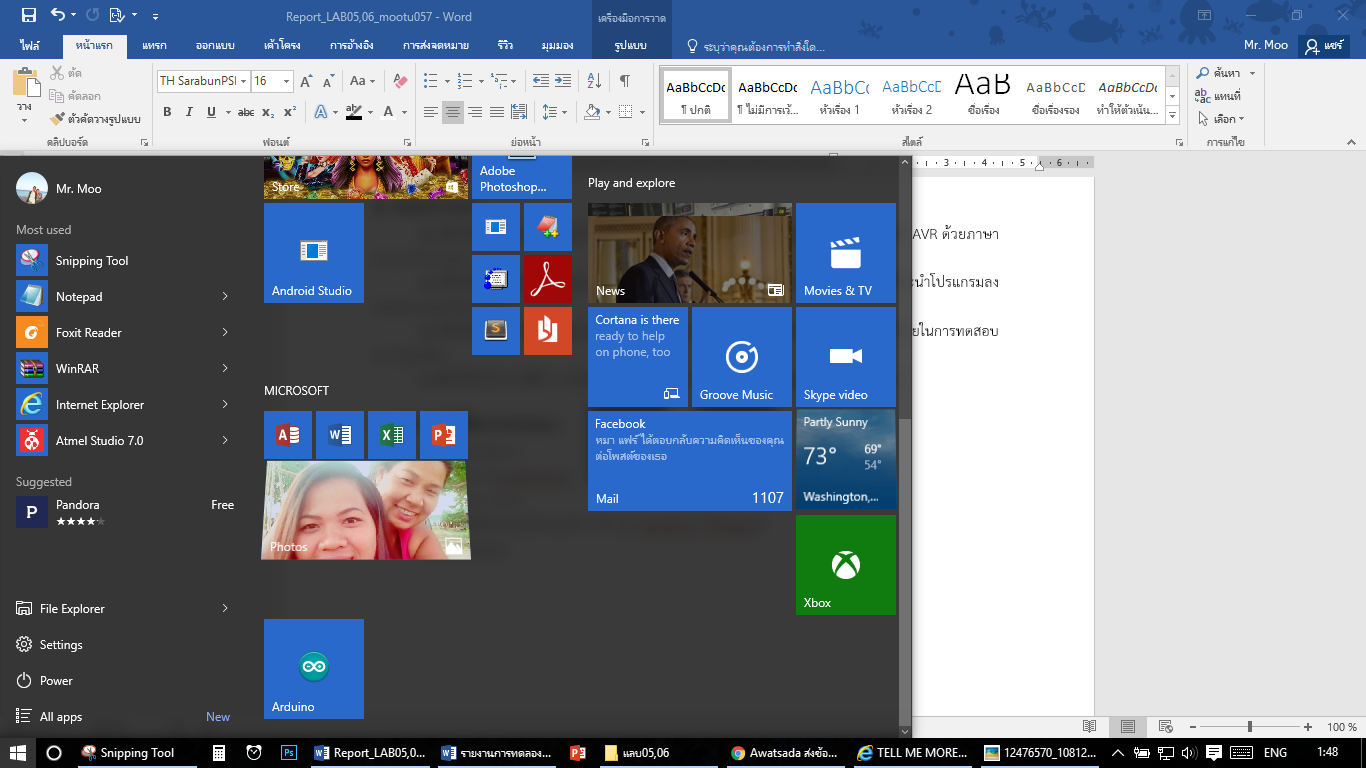
**▶ วัตถุประสงค์**

▶ เพื่อให้นักศึกษาได้เรียนรู้วิธีการเขียนโปรแกรมประยุกต์ใช้งานไมโครคอนโทรลเลอร์ AVR ด้วยภาษา Assembly และ C

▶ เพื่อให้นักศึกษาได้เรียนรู้เทคนิคการดีบักโปรแกรมภาษา Assembly และ C ก่อนที่จะนำโปรแกรมลงทดสอบบนฮาร์ดแวร์จริง

▶ เพื่อให้นักศึกษาได้เรียนรู้เทคนิคการออกแบบฮาร์ดแวร์และซอฟต์แวร์ร่วมกัน เพื่อให้ง่ายในการทดสอบความถูกต้อง

▶ เพื่อให้นักศึกษาได้ฝึกการเขียนซอฟต์แวร์สำหรับการทดสอบความถูกต้องของฮาร์ดแวร์



**▶ อุปกรณ์ที่ใช้ในการทดลอง**

▶ AVR Studio 4

▶ Proteus 8 professinal

▶ Arduino 1.6.8

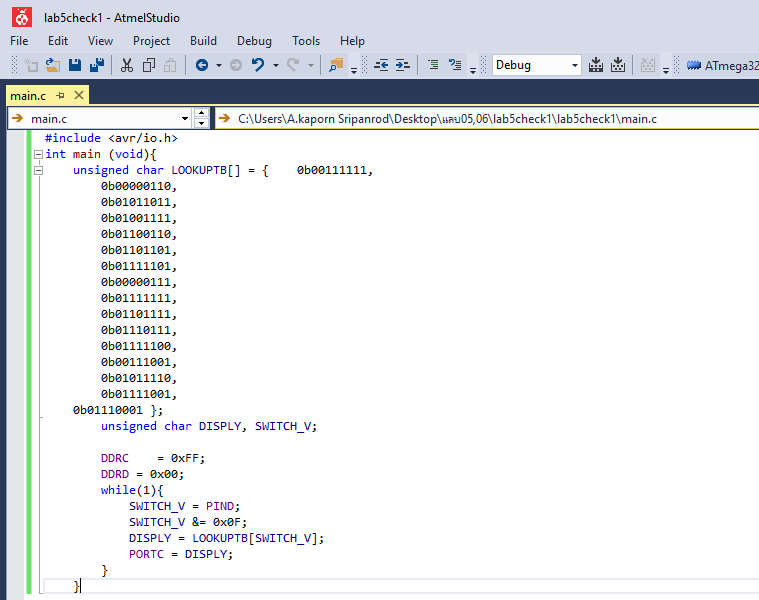
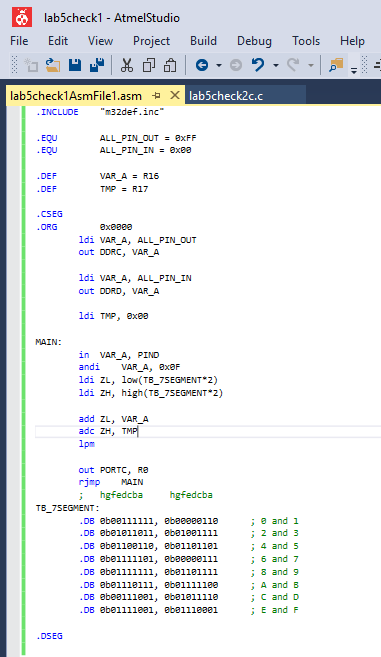
▶ บอร์ดไมโครคอนโทรลเลอร์ AVR รุ่น Diecilima ATMega32

▶ บอร์ดทดลอง

**▶ CHECKPOINT 1** เขียนโปรแกรมเพื่ออ่านค่าจากสวิทช์และแสดงผลทาง 7-segment LED

Assembly

**C Programming**



กำหนดตัวแปรชนิด unsigned

ใน loop while จะเช็คเฉพาะ4บิตล่าง(DIP switch) แล้วนำค่าไปชี้ในตาราง 7-segment แล้วเอาค่าไปแสดงผลใน proteus

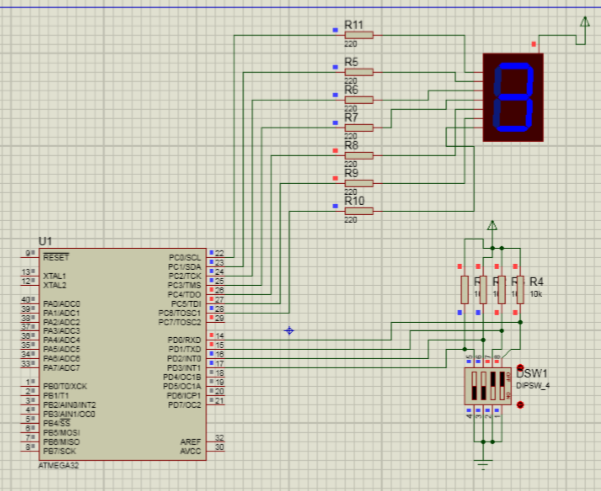
กำหนดตัวแปรชนิด unsigned

ให้ PORTD เป็น Input (รับข้อมูลจาก dipswitch)และ PORTC เป็น Output

>>**ASSAMBLY** : ไฟล์ .hex size = 157 bytes

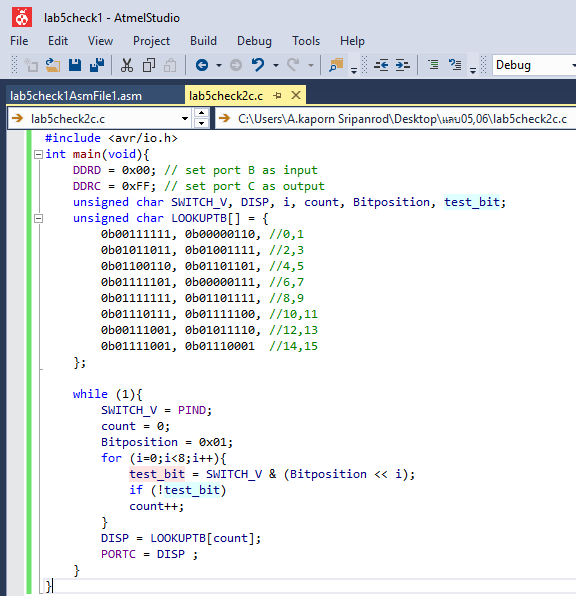
>>**C Programming** : ไฟล์ .hex = 627 bytes

จะเห็นว่าไฟล์ .hex ของ ASSAMBLY มีขนาดน้อยกว่า ไฟล์ .hex ของ C Programming

**Proteus จำลองการทำงานของโปรแกรม**

**▶ CHECKPOINT 2** เขียนโปรแกรม เพื่อนับลอจิกต่ำจากดิปสวิทช์

โปรแกรมจะทำการอ่านค่าจากดิปสวิทช์จำนวน 8 ตัว ที่ต่อกับพอร์ท D จากนั้น ทำการตรวจสอบบิตที่มาค่าลอจิกต่ำ และแสดงผลออก 7-segment ที่ต่อที่พอร์ท C



**C Programming**

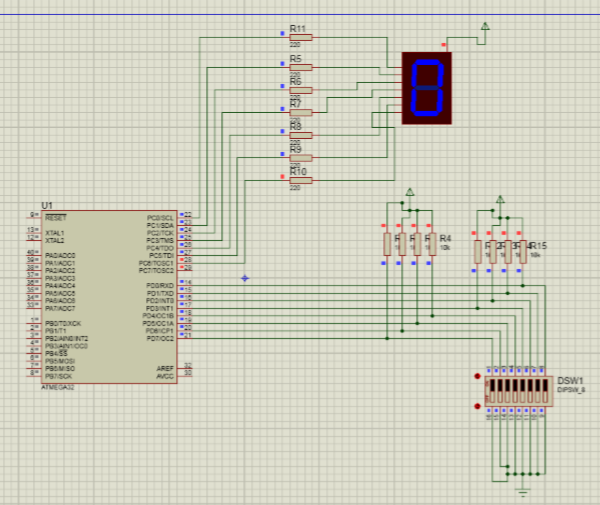
ให้ PORTD เป็น Input (รับข้อมูลจาก dipswitch)และ PORTC เป็น Output

Count เป็นตัวนับ ใน loop while

While loop ใช้วนนับโดย จะทำการชิป(Bitposition << 1) แล้วเช็คว่า เป็น High หรือไม่

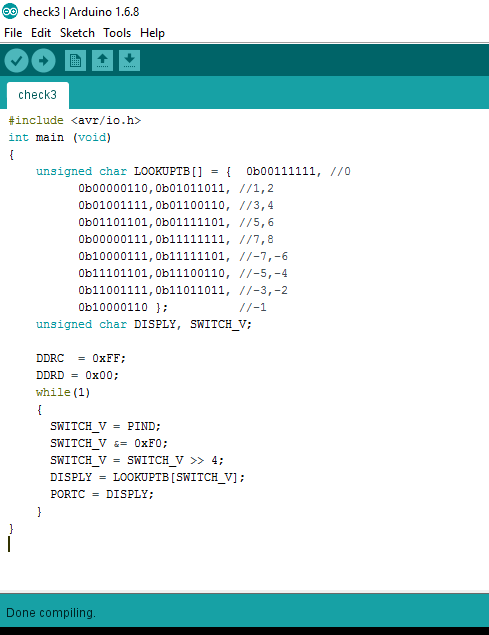
\*หากเป้น High ให้เพิ่มค่า Count++ แล้ววนเช็คบิตถัดไปจนครบ 8 บิต ก็จะได้จำนวนของ logic low แล้วนำค่าที่ได้ไปชี้ใน LOOKUPTB[ ] และให้แสดงผลออกมาทาง 7-segment

**Proteus จำลองการทำงานของโปรแกรม**



**▶ CHECKPOINT 3 เขียนโปรแกรม เพื่ออ่านค่าจากสวิทช์ โดยข้อมูลเป็นเลข signed number**

จงเขียนโปรแกรมเพื่อแปลงค่าจากสวิทช์เป็นเลขฐานสิบแบบมีเครื่องหมาย ซึ่งมีค่าอยู่ระหว่าง -8...+7 ออกแสดงผลทางแอลอีดี แบบ 7 เซกเมนต์ 2 ตัว

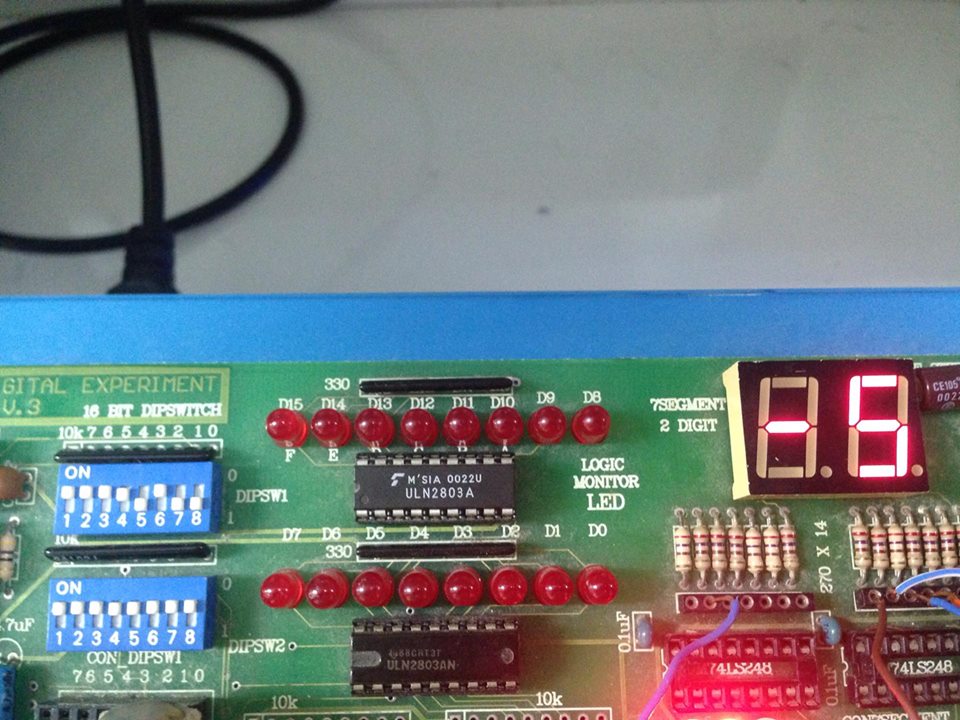


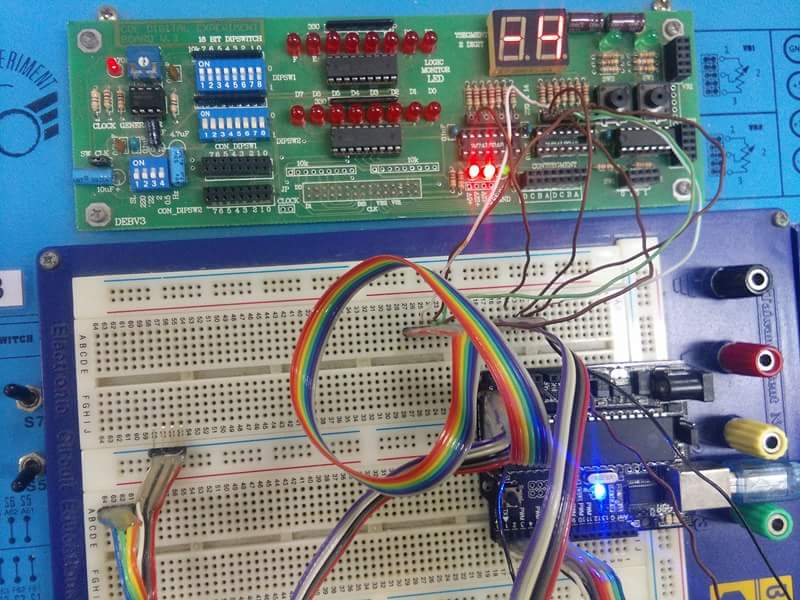
LOOKUPTB[] ให้แสดงผลออกมาทาง 7-segment โดยกำหนด ให้มีค่าเลข ติดลบ โดยในวงจรจะใช้ 7-segment 2 ตัว

ให้ DDRC เป็น output

และ DDRD เป้น input ใช้ 4 บิตล่าง

Shift 4 bit และให้ชี้ตำแหน่งไปที่ LOOKUPTABLE[ ] โดยในตารางจะมีค่าระหว่าง -8 ถึง +7 ในส่วนที่เป้น ลบ จะกำหนด sign bit = 1 หากไม่มีก็กำหนดให้เป้น 0

**เมื่อนำโปรแกรมอัดลงบอร์ด**

****

**▶ CHECKPOINT 4 เขียนโปรแกรม เพื่อแสดงค่าจากสวิทช์ โดยข้อมูลเป็นเลขฐานสิบ**

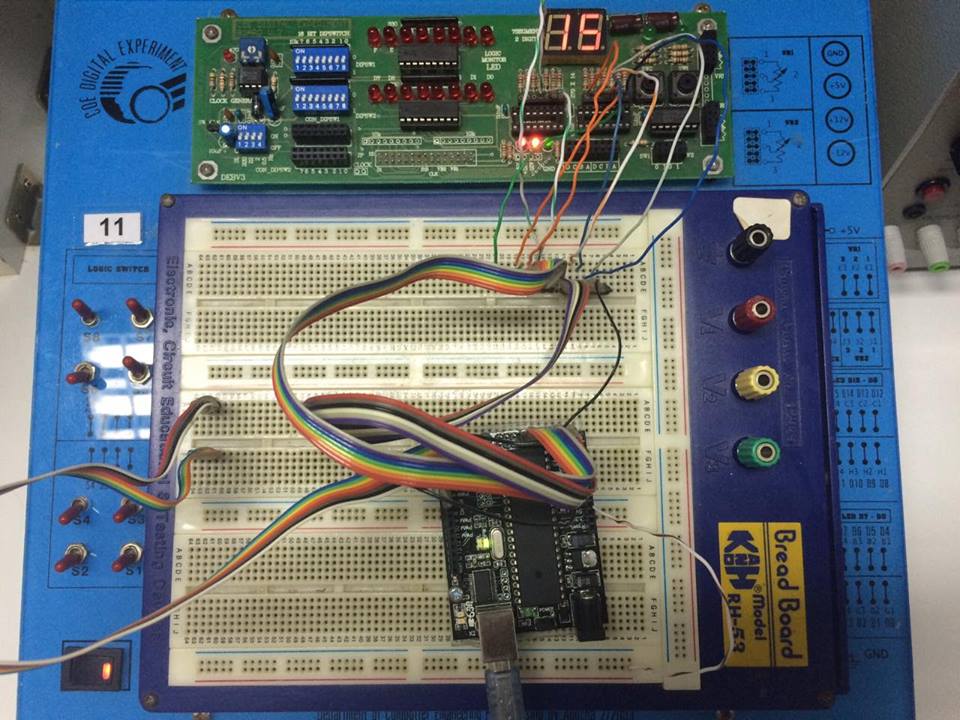
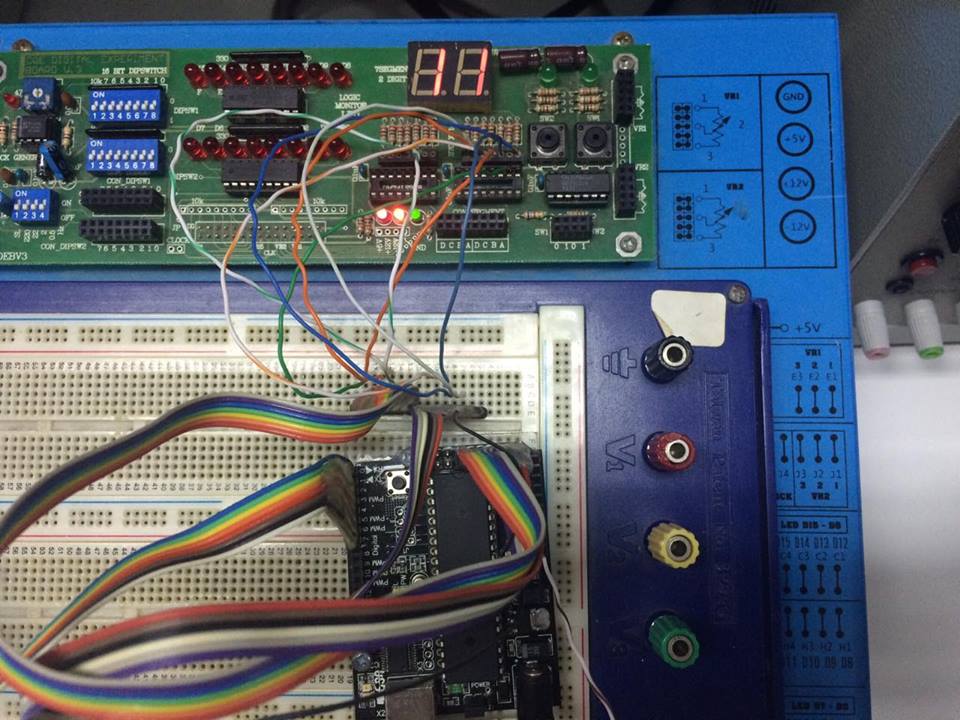
จงเขียนโปรแกรมเพื่อแปลงค่าจากสวิทช์เป็นเลขฐานสิบแบบไม่มีเครื่องหมาย ซึ่งมีค่าอยู่ระหว่าง 0-15 ออกแสดงผลทางแอลอีดี แบบ 7 เซกเมนต์ 2 ตัว



LOOKUPTB[] ให้แสดงผลออกมาทาง 7-segment โดยกำหนด ให้แสดงเลขตั้งแต่ 0-15 โดยในวงจรจะใช้ 7-segment 2 ตัว

เลข 1 ข้างหน้า (10-15) กำหนดให้ signed bit เป้น 1 แต่จะมีการต่อของสายจุด b กับ C ใน7-segment ของบอร์ดทดลอง ดยหลักการจะคล้าย check 3

**เมื่อนำโปรแกรมอัดลงบอร์ด**

****

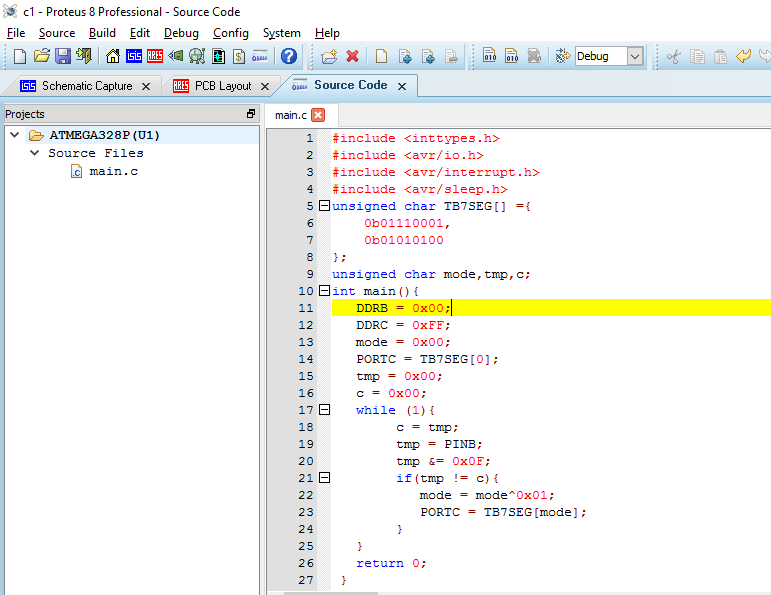
**การทดลองที่ 5 ออกแบบและทดสอบระบบสวิทช์สัมผัสสำหรับเปิดปิดเครื่องใช้ไฟฟ้า**

**▶ CHECKPOINT 5.1 ออกแบบและทดสอบระบบสวิทช์สัมผัสสำหรับเปิดปิดเครื่องใช้ไฟฟ้า**

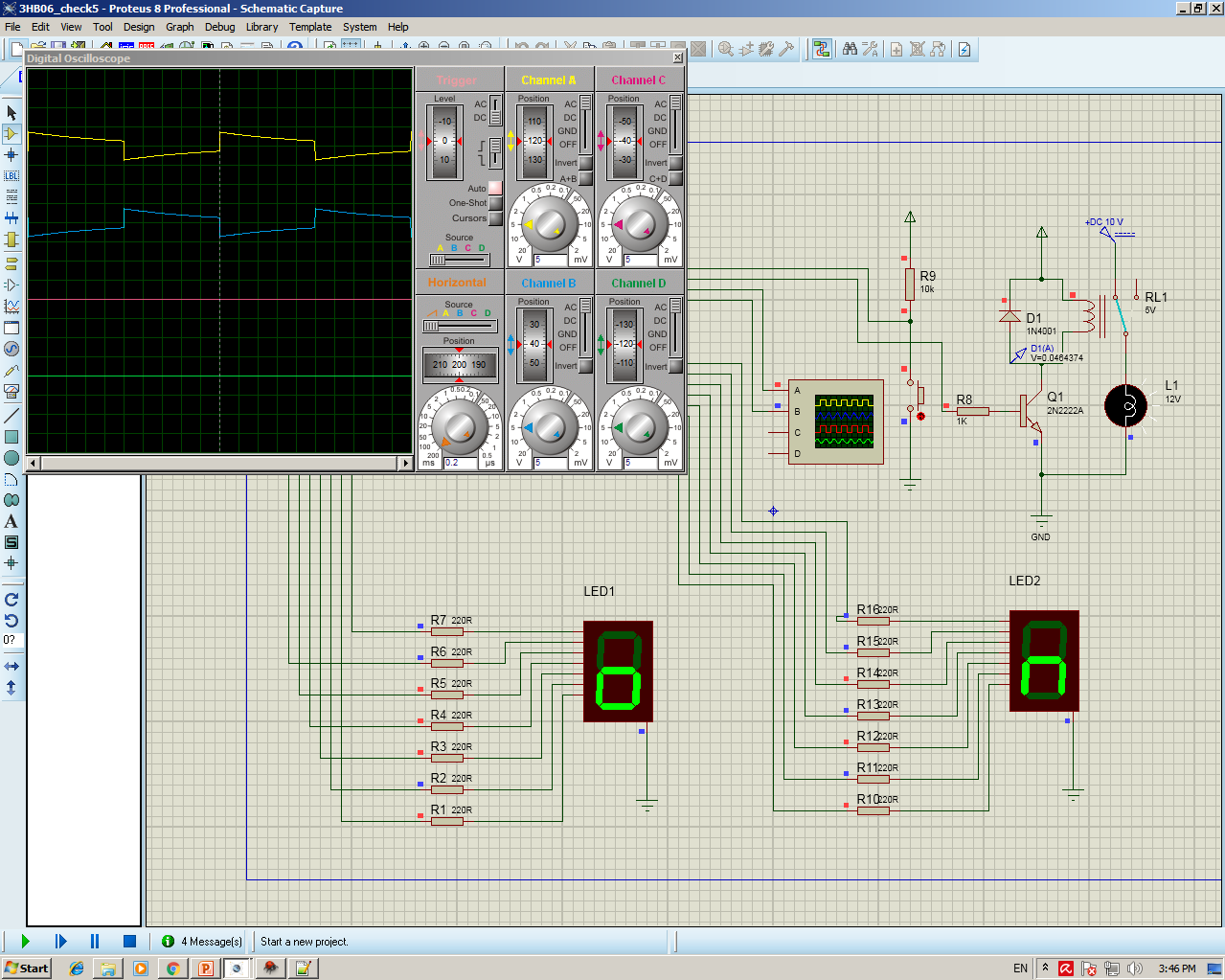
****

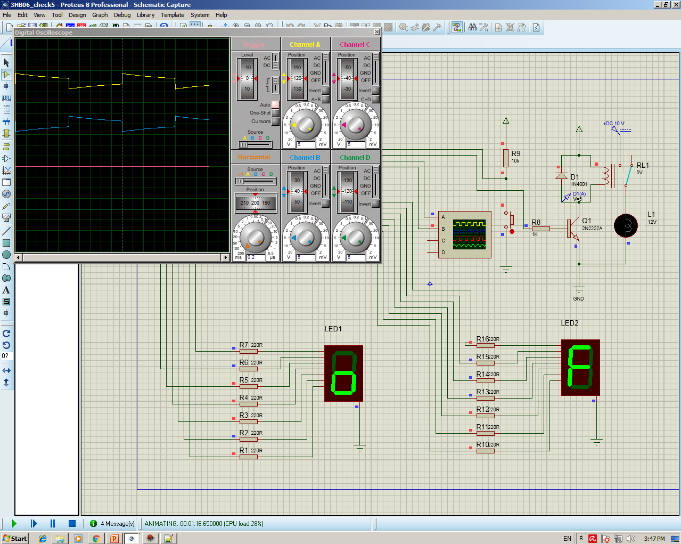
- เมื่อเริ่มจ่ายไฟให้กับโปรแกรม หลอด L1 จะดับอยู่ และ 7-segment แสดงคำว่า

 - เมื่อกดสวิทช์ หลอด L1 จะติด และ 7-segment แสดงคำว่า

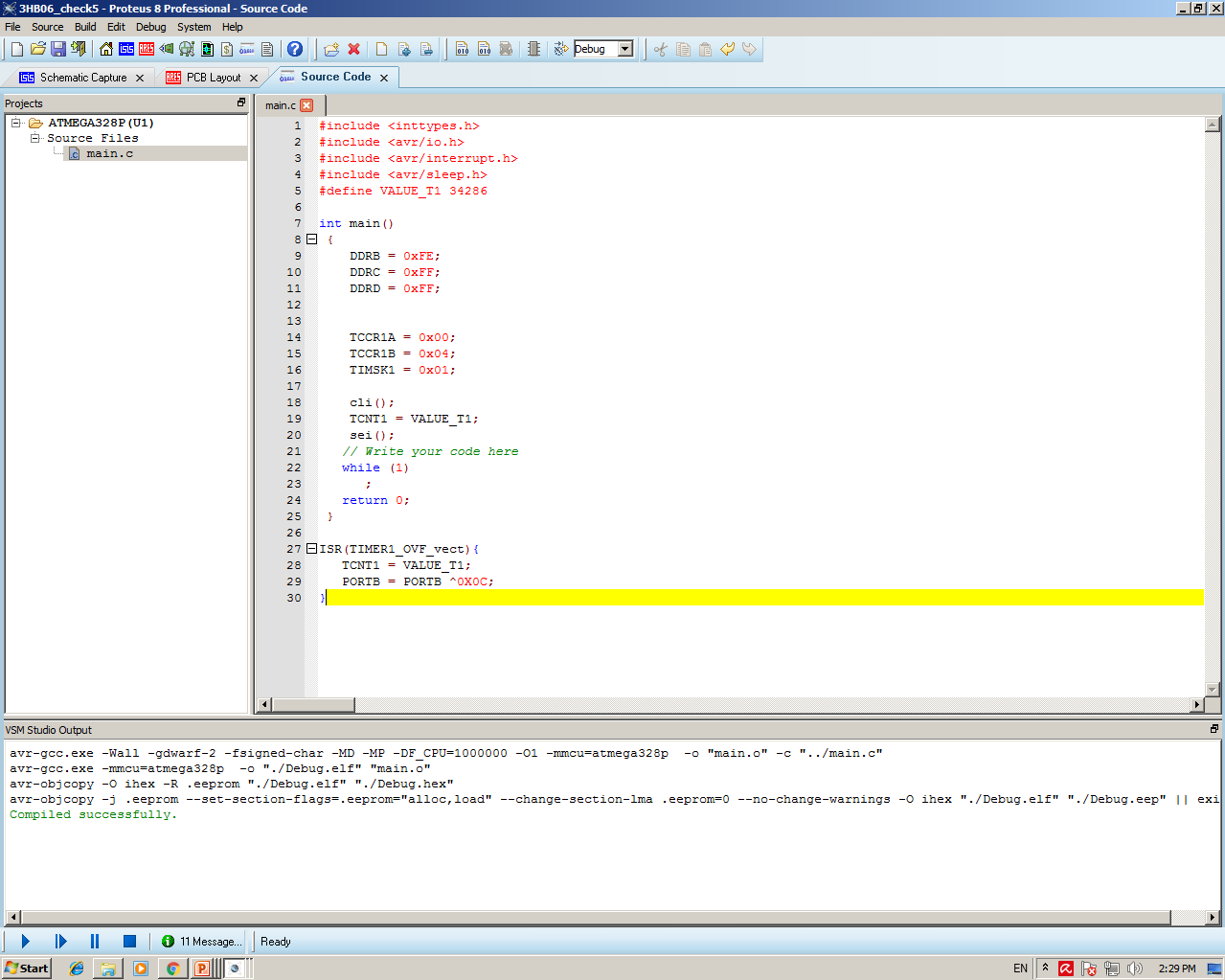
 - เมื่อกดสวิทช์อีกครั้ง หลอด L1 จะดับ และ 7-segment แสดงคำว่า

**ผลลัพธ์ของวงจร ที่ได้ใน program proteus**

****

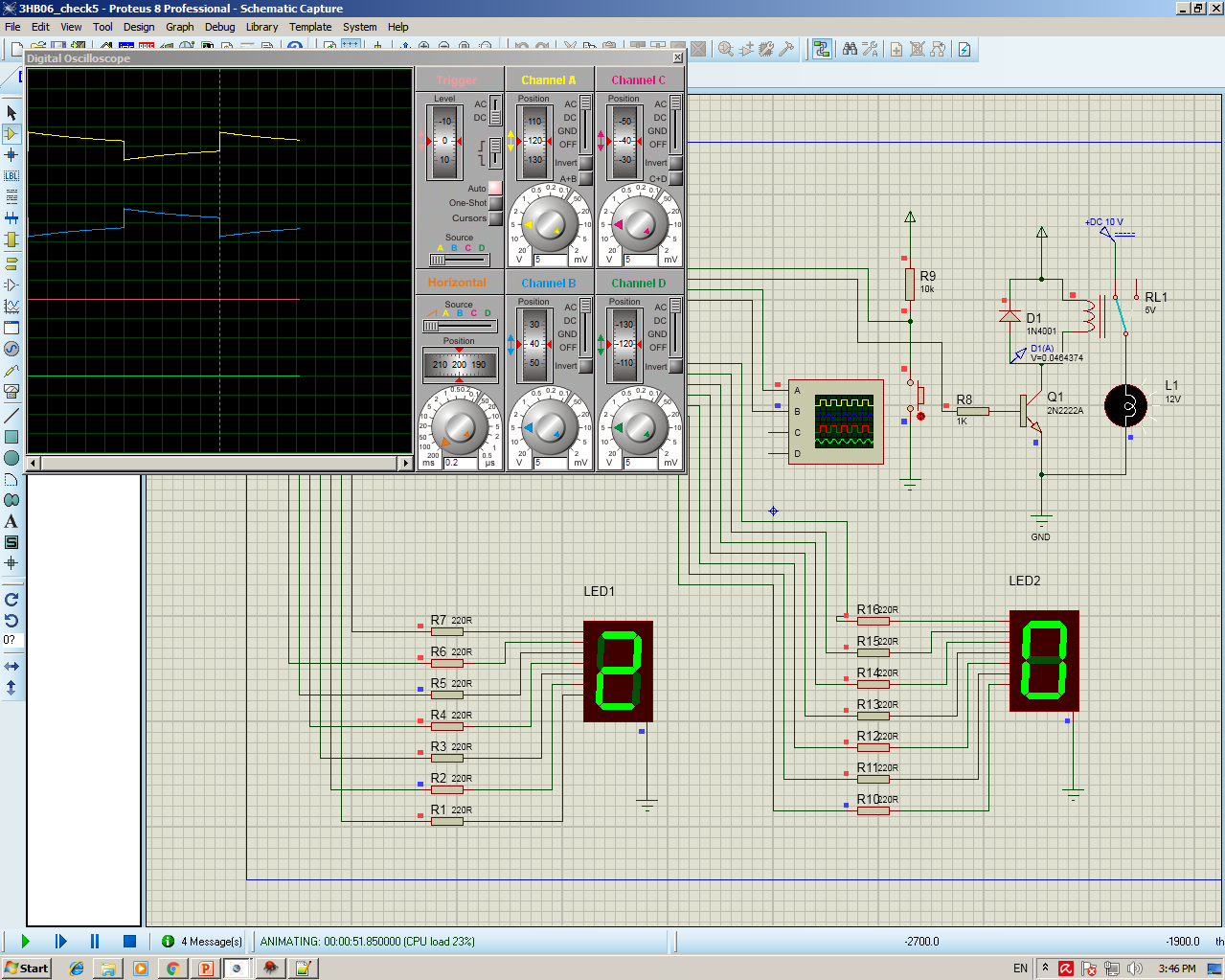
****

**▶ CHECKPOINT 5.2เขียนโปรแกรมสร้างพัลส์ ออก Oscilloscope**

**** - เมื่อเริ่มจ่ายไฟให้กับโปรแกรม สัญญาณพัลส์ที่วัดได้ที่ Oscilloscope จะมีความถี่ 0.5 Hz (คาบ 2 sec)

#define VALUE\_T1 34286 เป็นตัวกำหนดความเร็วในการนับถอยหลัง

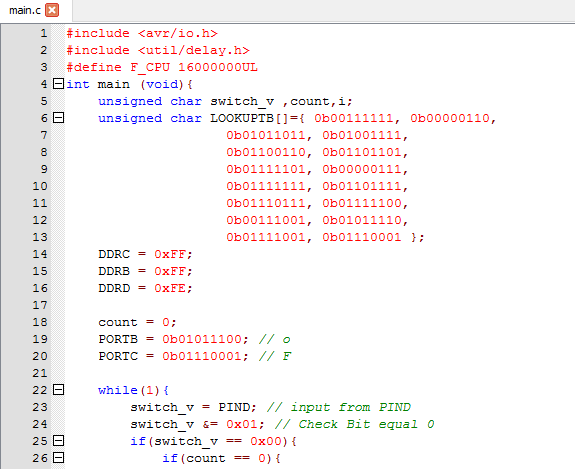
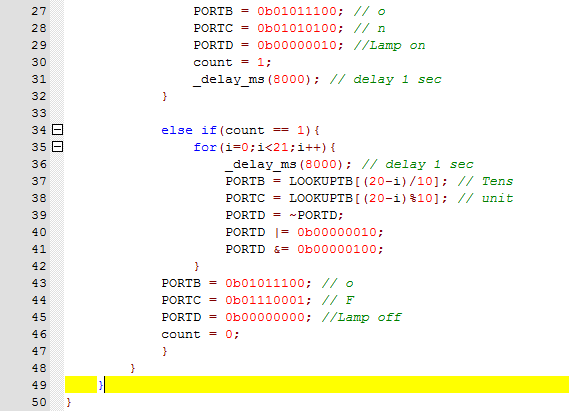
**ผลลัพธ์ของวงจร ที่ได้ใน program proteus**

****

**▶ CHECKPOINT 5.3 เขียนโปรแกรมตั้งเวลา**

- เมื่อเริ่มจ่ายไฟให้กับโปรแกรม 7-segment แสดงเลข 20 เมื่อกดสวิทช์ 7-segment จะเริ่มนับถอยหลังจากทุกๆ 1 วินาที จะมีค่าลดลงทีละหนึ่ง จนเหลือ 00

- ถ้ากดสวิทช์อีกครั้ง 7-segment จะรีเซตจากค่า 00 มาเป็น 20

 - ถ้ากดสวิทช์อีกครั้ง 7-segment จะนับถอยหลังไปที่เลข 00 การทำงานจะเป็นสเตทสลับไปมา

\*\*TCCR1A= 0x00เป็นการบอกว่า เป็น timer ขนาด 16-bit แบบ normal

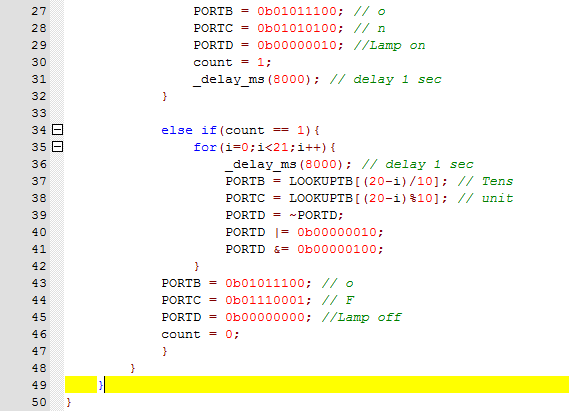
\*\*TCCR1B = 0x04 ใช้ prescalar = 256

เริ่มต้นให้ count = 20 และเริ่มนับถอยหลัง โดยให้

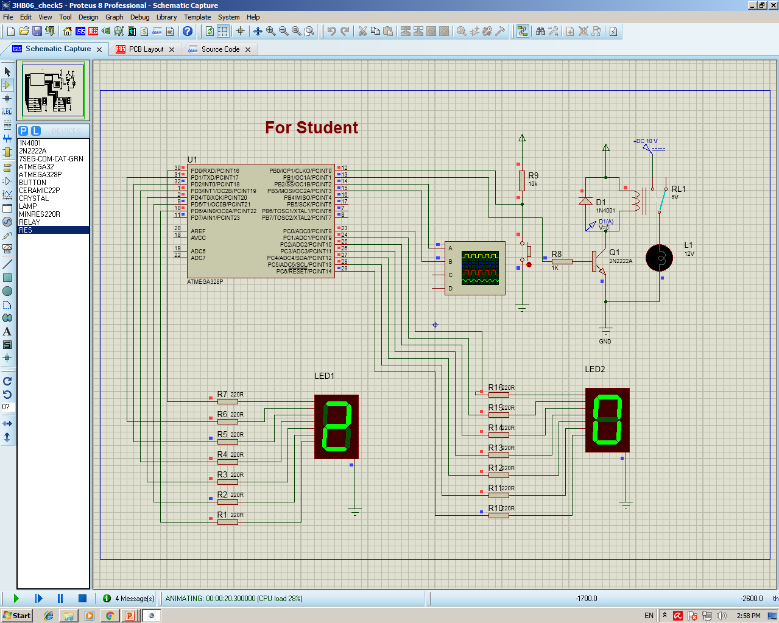
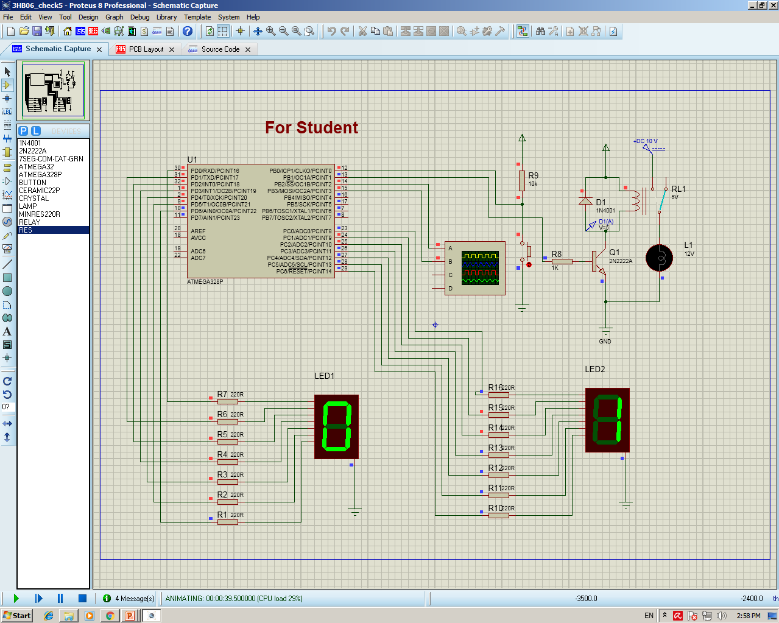
\*port c เป็นหลักหน่วย (count mod 10)

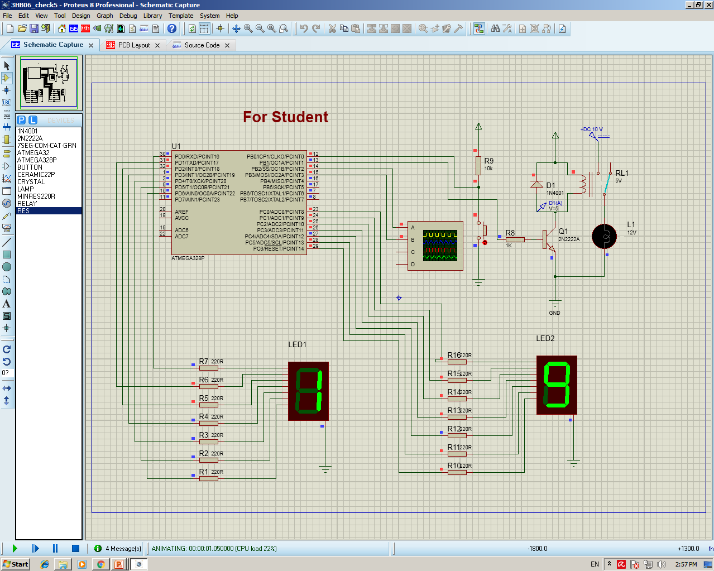
\*port d เป้นหลักสิบ (count หาร 10)

ให้ PORTC, D, B เป้น input

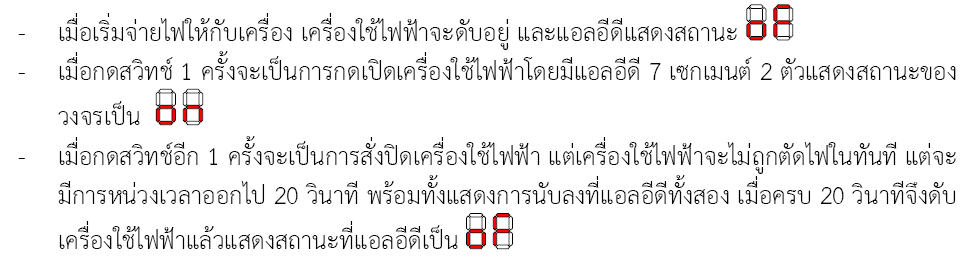


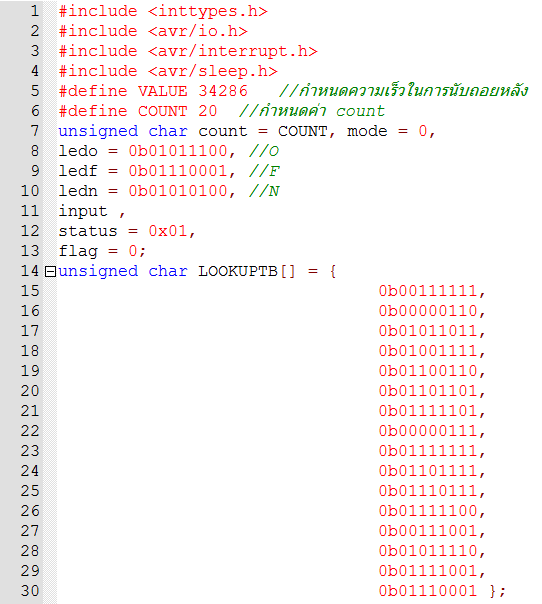
ISR จะเคลียร์บิตที่ 0, 1 โดยการ OR กัน และลดค่า count 1 จนมีค่า เท่ากับ 0 ก็จะ รีเซตให้เป็น20อีกครั้ง แสดงทาง 7-segment

**ผลลัพธ์ของวงจร ที่ได้ใน program proteus**

****

**▶ CHECKPOINT 5.4**

****

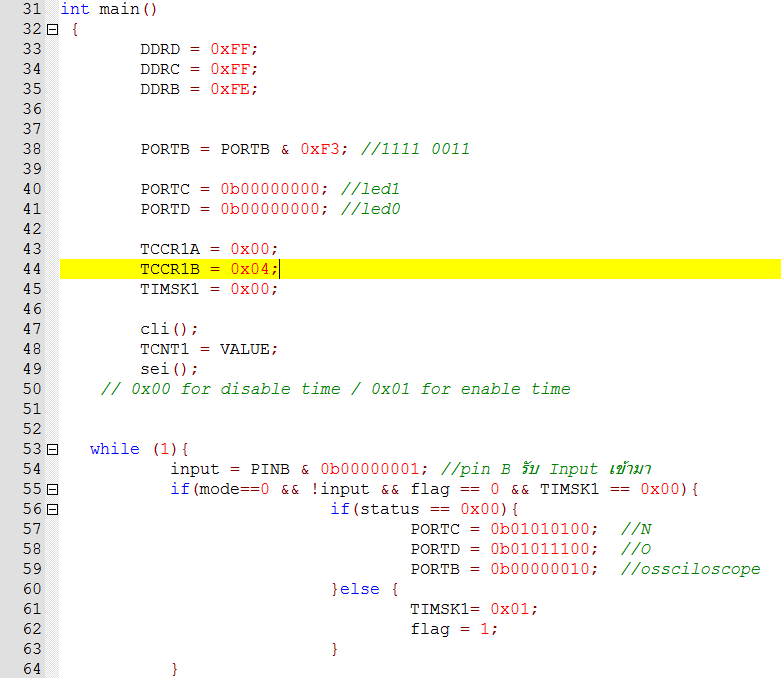
****

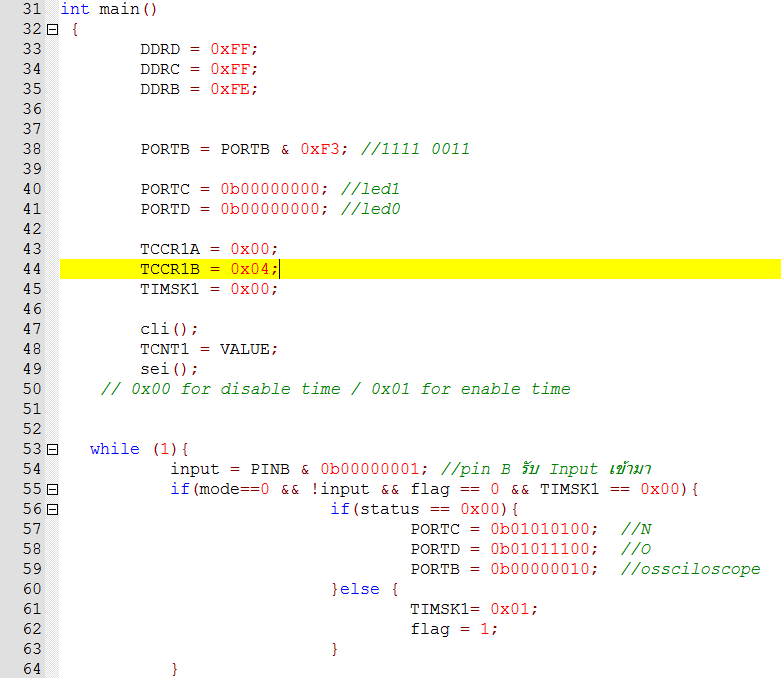
ในข้อนี้จะนำ โค้ดข้อ 5.1มาผสมกับ 5.3 รวมกัน ที่ต่างกันคือ เริ่มต้นให้ 7-seg แสงผลเป็น of เมื่อกด button Register ก้จะทำงาน(TIMER1\_OVF\_vect)

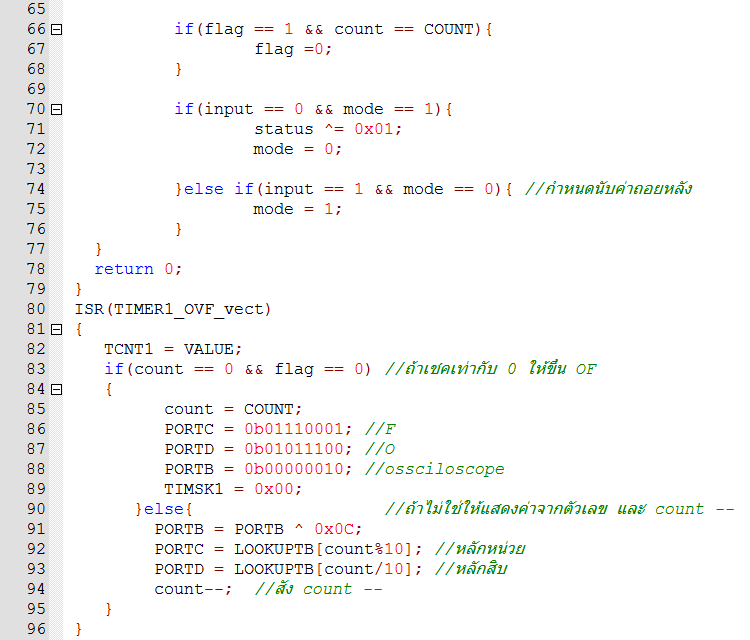
\*\*เมื่อ 7-seg แสดง คำว่า on ค่า count ก้จะเริ่มนับที่ 20 และลดค่าลงเรื่อยๆ

\*\*เมื่อค่า count = 0 7-seg ก็จะแสดงคำว่า of

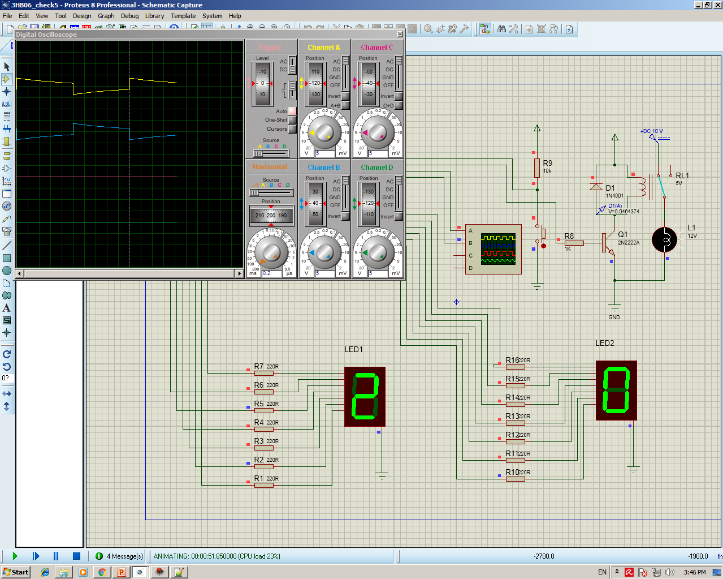
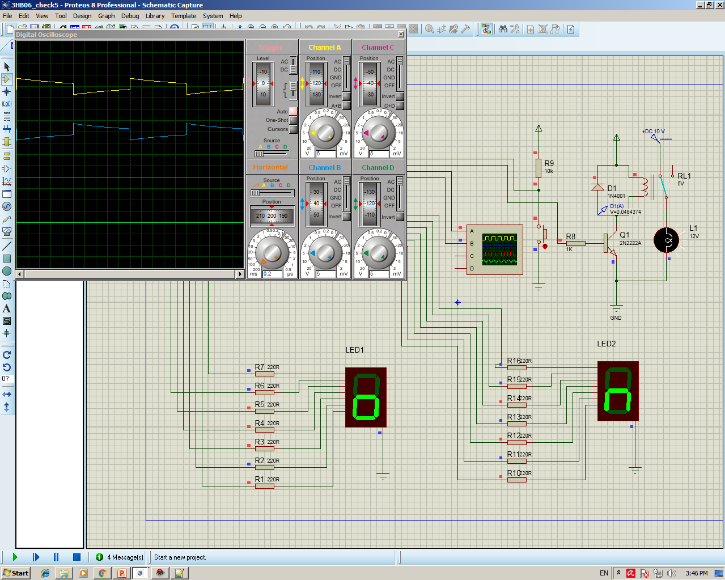
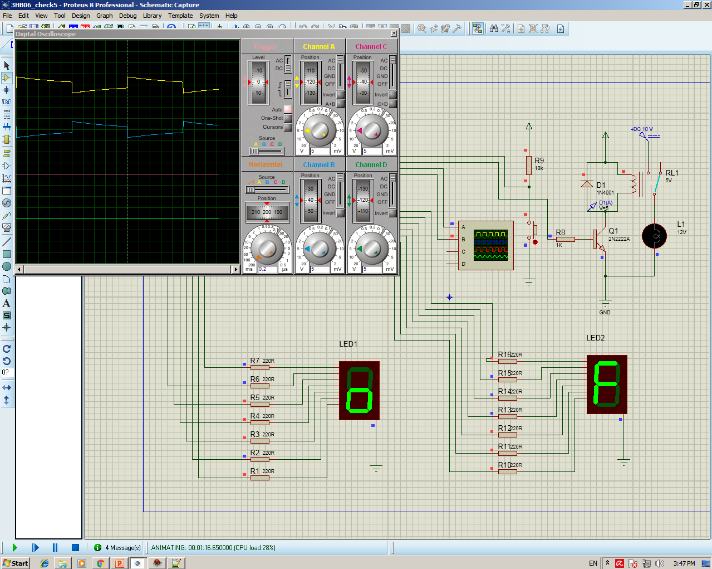
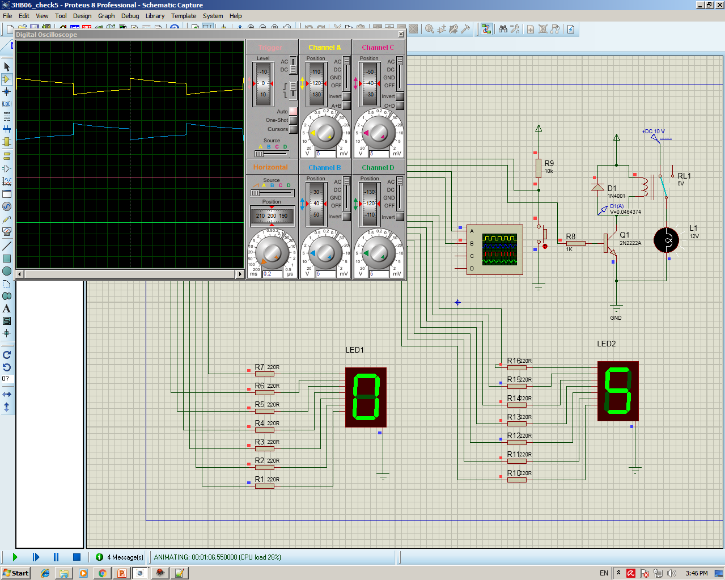
หากกด button อีก ก็จะนับอีกครั้ง (วนลูป)

****

****

****

**ผลลัพธ์ของวงจร ที่ได้ใน program proteus**



เมื่อนับจนถึง 00 ก็จะขึ้น of และเมื่อกดอีกครั้งก็จะนับอีก

กด button อีก 1 ครั้ง ก็จะเริ่มนับถอยหลังเรื่อยๆจาก 20

กด button 1 ครั้ง ก็จะขึ้น on และจะมีไฟติด