ใบงานครั้งที่ 04

Basic Simulation / Startup.s

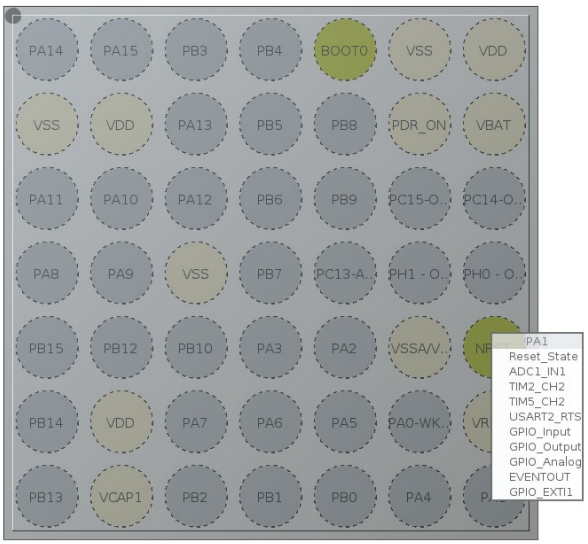
<u>อุปกรณ์ที่ใช้ในการทดลอง</u>

1. STM32CubeIDE
2. Proteus 8.9 SP 2

<u>ขั้นตอนการทดลอง</u>

<u>การ build โปรแกรมสำหรับการทดลองหรือ burn ลงบนบอร์ดจริง</u>

1. เลือก Pin PA1 เพื่อทำการกำหนดเป็น GPIO_Output



2. เมื่อทำการ save โปรแกรม STM32CubeIDE จะทำการสร้างโค้ดให้ทันที

(โค้ดด้านล่าง แสดงโค้ดในส่วนของ Interrupt Service Routine; ISR ในไฟล์ startup_xx.s)

- เมื่อเกิดการ reset จะไปเรียก Interrupt Service Routine(ISR) ที่ชื่อว่า Reset_Handler

```
113 /******************************************************************************
114 *
115 * The minimal vector table for a Cortex M3. Note that the proper constructs
116 * must be placed on this to ensure that it ends up at physical address
117 * 0x0000.0000.
118 *
119 ******************************************************************************/
120   .section  .isr_vector,"a",%progbits
121   .type  g_pfnVectors, %object
122   .size  g_pfnVectors, .-g_pfnVectors
123
124 g_pfnVectors:
125   .word  _estack
126   .word  Reset_Handler
128   .word  HardFault_Handler
129   .word  MemManage_Handler
130   .word  BusFault_Handler
131   .word  UsageFault_Handler
132   .word  0
133   .word  0
134   .word  0
135   .word  0
136   .word  SVC_Handler
137   .word  DebugMon_Handler
138   .word  0
139   .word  PendSV_Handler
140   .word  SysTick_Handler
```

3. ในไฟล์ startup_xx.s เมื่อเกิดการ reset ตัว MCU จะเรียกใช้ function ชื่อ Reset_Handler เมื่อทำงานโปรแกรมเสร็จแล้วจึงไปเรียก function main ในไฟล์ main.c

```
    .section  .text.Reset_Handler
  .weak  Reset_Handler
  .type  Reset_Handler, %function
Reset_Handler:
  ldr   sp, =_estack      /* set stack pointer */

/* Copy the data segment initializers from flash to SRAM */
  movs  r1, #0
  b  LoopCopyDataInit

CopyDataInit:
  ldr  r3, =_sidata
  ldr  r3, [r3, r1]
  str  r3, [r0, r1]
  adds  r1, r1, #4

LoopCopyDataInit:
  ldr  r0, =_sdata
  ldr  r3, =_edata
  adds  r2, r0, r1
  cmp  r2, r3
  bcc  CopyDataInit
  ldr  r2, =_sbss
  b  LoopFillZerobss
/* Zero fill the bss segment. */
FillZerobss:
  movs  r3, #0
  str  r3, [r2], #4

LoopFillZerobss:
  ldr  r3, = _ebss
  cmp  r2, r3
  bcc  FillZerobss

/* Call the clock system intitialization function.*/
  bl  SystemInit
/* Call static constructors */
    bl __libc_init_array
/* Call the application's entry point.*/
  bl  main
  bx  lr
.size  Reset_Handler, .-Reset_Handler
```

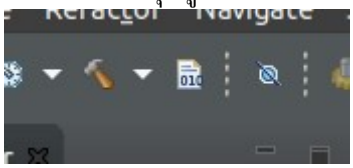
4. function main ในไฟล์ main.c จะทำการเรียก function ต่างๆ เพื่อทำการ configure MCU ทั้งในส่วนของ NVIC และ RCC

```
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  /* USER CODE BEGIN 2 */

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}
```

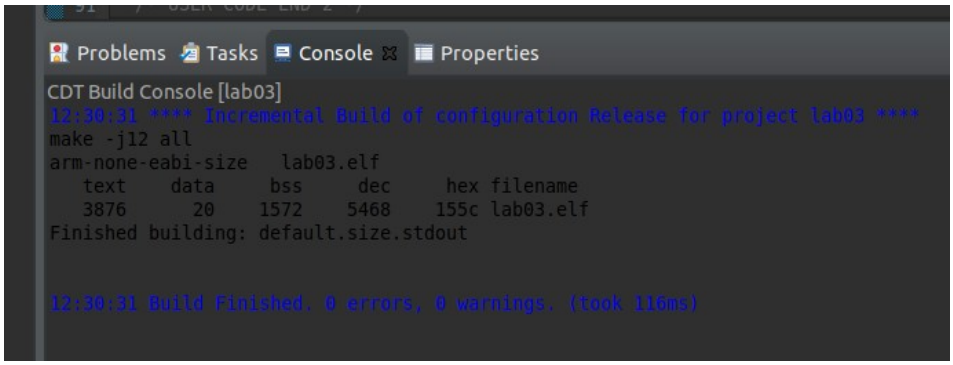
5. เขียนโค้ดเพิ่มเพื่อทำการ Toggle สัญญาณออกที่ Port A ขาที่ 1

```
94   /* USER CODE BEGIN WHILE */
95   while (1)
96   {
97     /* USER CODE END WHILE */
98
99     /* USER CODE BEGIN 3 */
100    HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_1);
101    HAL_Delay(500);
102  }
103  /* USER CODE END 3 */
104 }
105
```

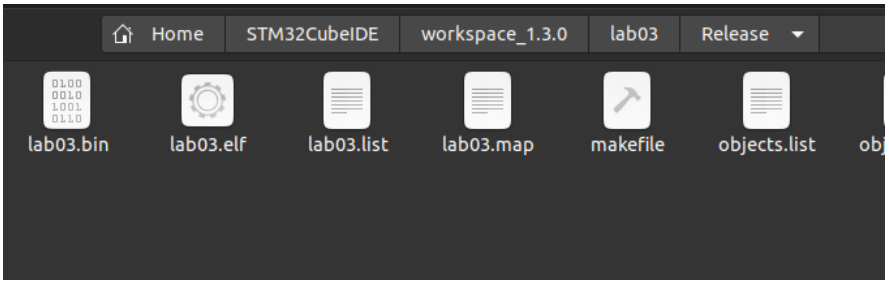
6. ทำการกำหนดการ build ให้เป็น Release โดยกดที่ปุ่มลูกศรชี้ลง แล้วเลือกเมนู Release



7. สามารถทำการ build โปรแกรมได้โดยกดปุ่มที่เป็นรูปฆ้อน หากไม่เกิด error จะแสดงผลการ build ที่ได้ตามรูปด้านล่าง



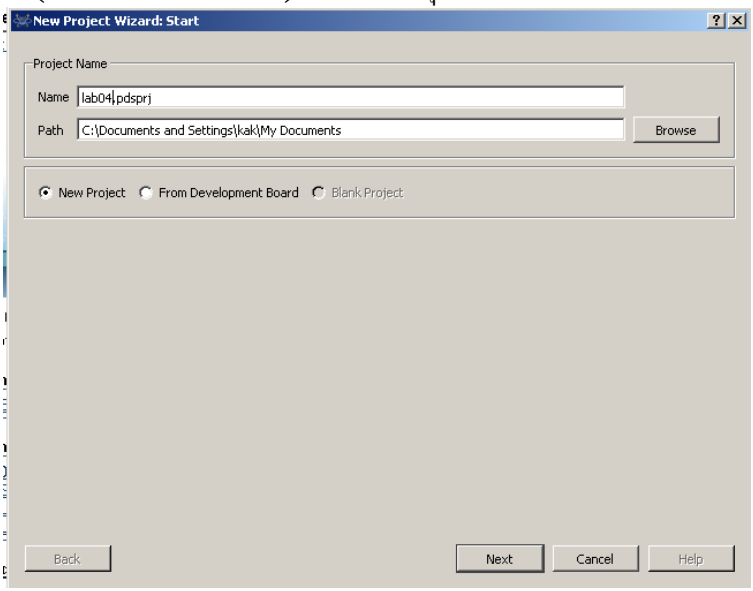
8. ใน folder ของโปรเจค labxx → Release จะพบไฟล์ที่ build สำเร็จดังนี้
- xx.bin => binary ไฟล์สำหรับ burn ลงบอร์ดจริง
- xx.elf => elf ไฟล์ที่มีตารางของสัญลักษณ์ที่สามารถโหลดลงตำแหน่งหน่วยความจำ (จะใช้ไฟล์นี้ในการ simulate การทำงานของ MCU)



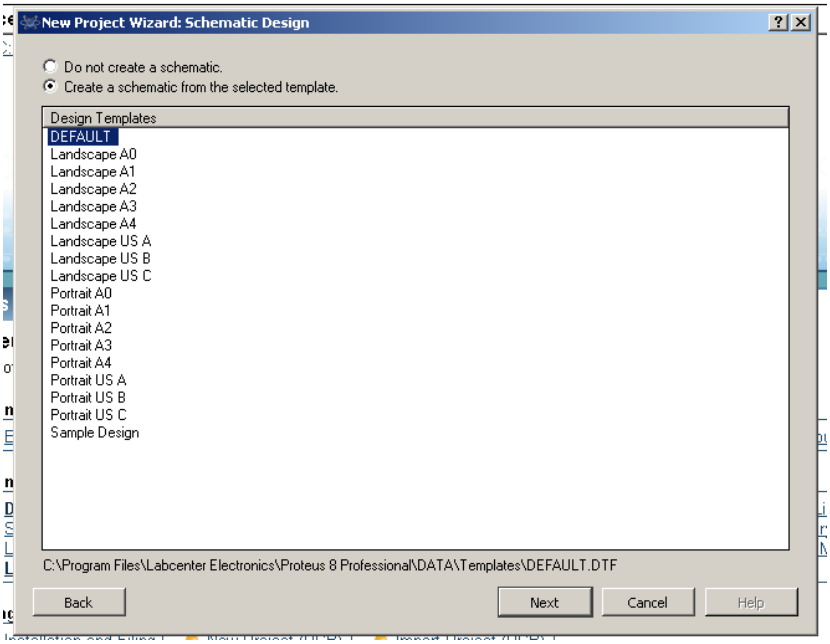
การจำลองการทำงานของบอร์ด STM32F401CC ด้วยโปรแกรม Proteus 8.9 SP2

1. เปิดโปรแกรม Proteus 8.9 ขึ้นมา จากนั้นเลือกสร้าง project ขึ้นมาใหม่ (New Project)

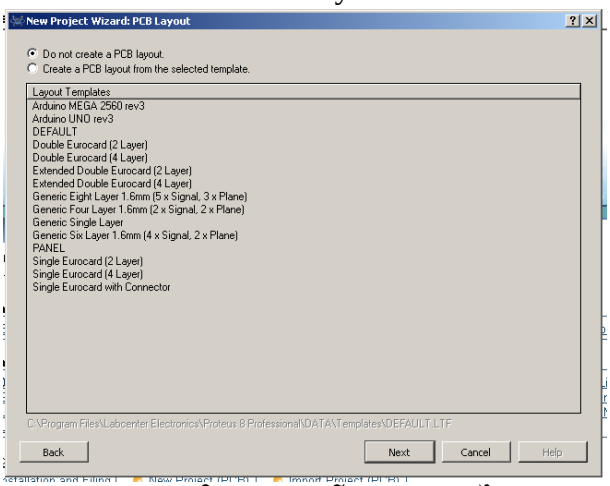
2. ตั้งชื่อโปรเจคที่ต้องการ (ในที่นี้ใช้ชื่อว่า lab04) จากนั้นกดปุ่ม next



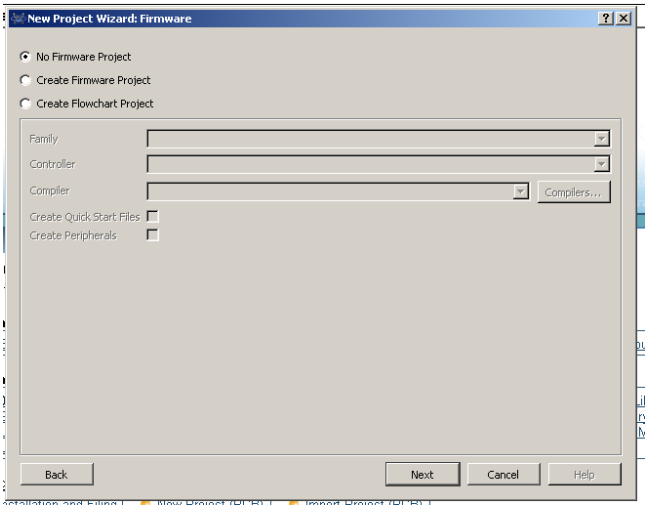
3. เลือกสร้าง schematic เพื่อใช้สำหรับการวาดวงจรที่ต้องการทดลอง กดปุ่ม next



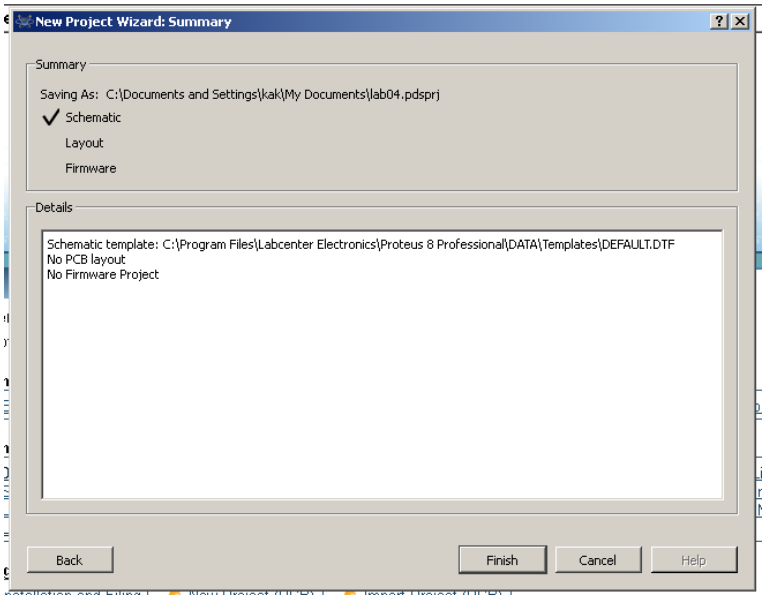
4. เนื่องจากไม่ต้องการสร้าง PCB จึงเลือกไม่สร้าง PCB Layout จากนั้นกด next



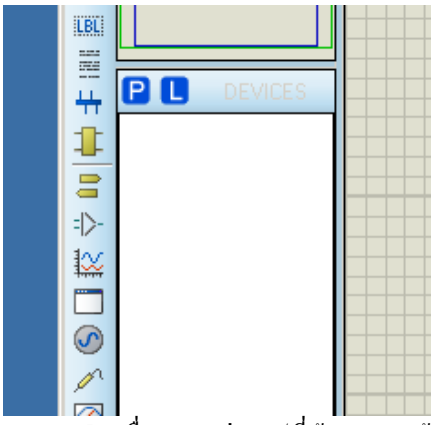
5. เนื่องจากจะใช้โปรแกรม STM32CubeIDE ในการพัฒนาโปรแกรม จึงไม่ต้องสร้าง firmware ให้กดปุ่ม next



6. กดปุ่ม Finish เพื่อสร้างโปรเจคตามที่ได้เลือกไว้

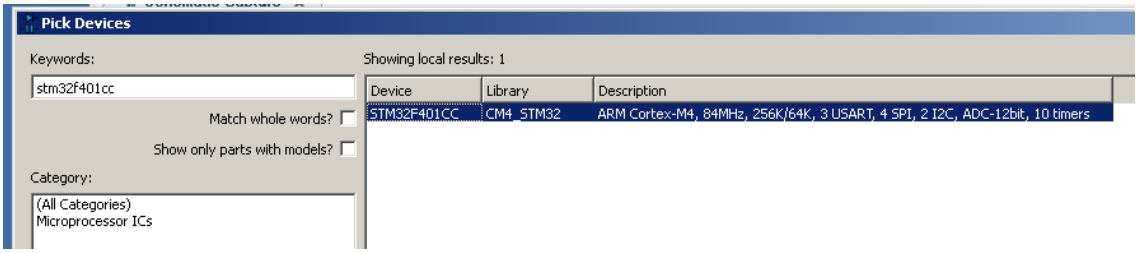


7. กดที่ตัวอักษร P เพื่อทำการเพิ่มอุปกรณ์ที่ต้องการใช้เข้ามา

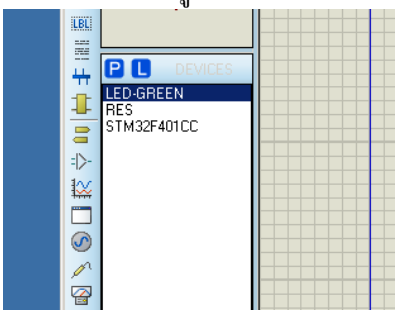


8. ค้นหาอุปกรณ์ที่ต้องการได้ที่ช่อง Keywords เมื่อเจออุปกรณ์ที่ต้องการแล้วให้กด double click ตัวอุปกรณ์ที่ต้องการ โดยอุปกรณ์ที่ต้องการเป็นไปตามข้างล่าง

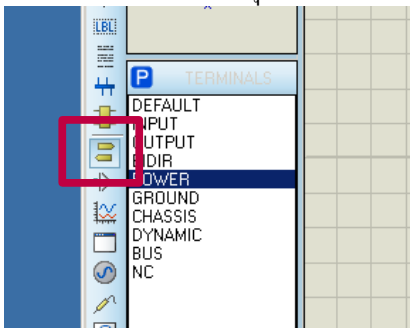
- STM32F401CC
- Res
- LED-GREEN



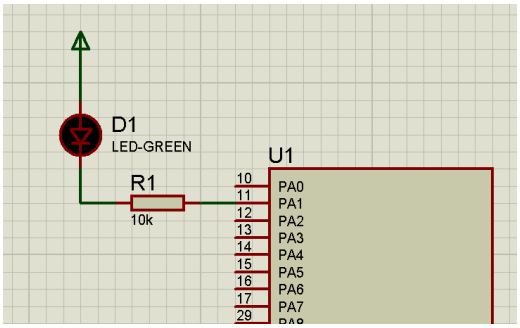
9. เมื่อเพิ่มอุปกรณ์แล้วจะได้รายชื่ออุปกรณ์ที่ต้องการดังรูปด้านล่าง



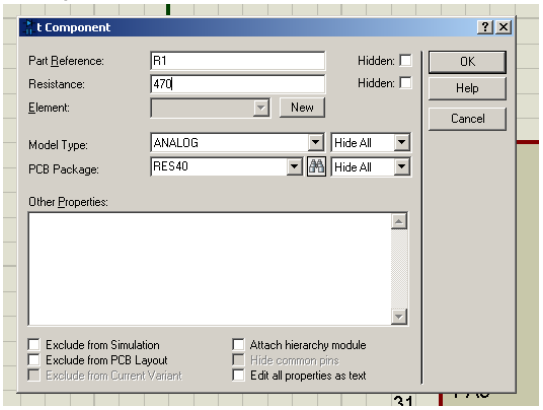
10. เมื่อต้องการจุดเชื่อมต่อที่เป็นแหล่งจ่ายไฟฟ้าให้เลือกได้ที่อุปกรณ์ bus terminal



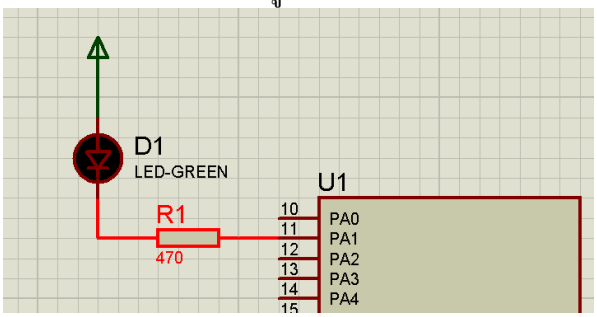
11. ต่อวงจรตามรูปด้านล่าง
- เนื่องจากในโปรแกรมที่ได้เขียนไว้กำหนดสัญญาณออกที่ PA1 จึงเชื่อมต่อ LED สีเขียวและตัวต้านทานเพื่อจำกัดกระแสตามรูป



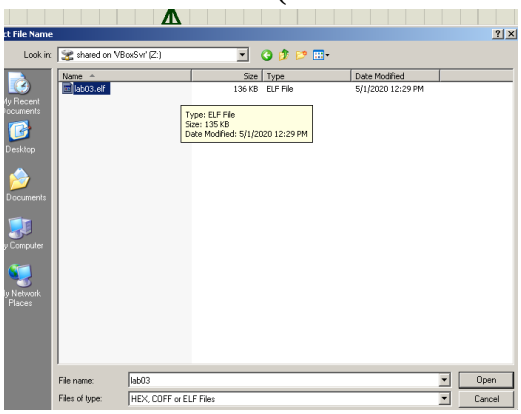
12. เนื่องจากต้องการกำหนดค่าความต้านทานของ R1 ให้มีค่าเป็น 470 โอห์ม ให้ทำการ double click ตัวอุปกรณ์ที่เป็นตัวต้านทานที่ชื่อว่า R1 จะปรากฏหน้าต่างสหรับกำหนดมาให้ ทำการเปลี่ยนค่าเป็น 470 จากนั้นกดปุ่มยอมรับ (OK) (ในกรณีอุปกรณ์อื่นๆ ก็สามารถกระทำได้เช่นเดียวกัน)



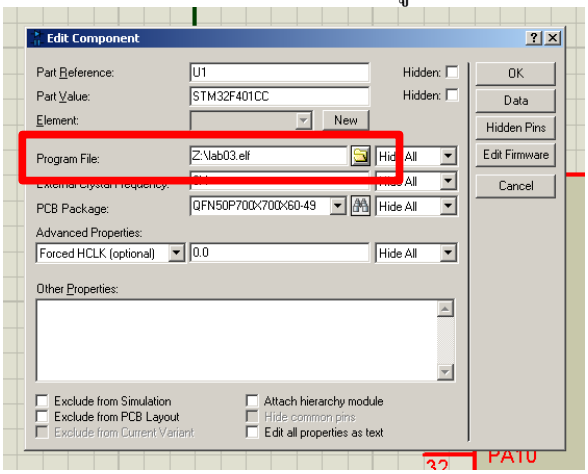
13. ค่าความต้านทานที่กำหนดไว้ก็จะเปลี่ยนแปลงได้ดังรูป



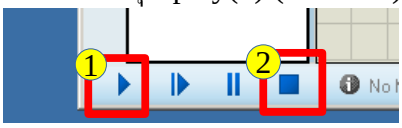
14. ทำการกำหนดโปรแกรมที่จะให้ MCU STM32F401CC ทำงานโดยการ double click และในช่อง Program File: ให้เลือกโปรแกรมที่เราจะใช้ในการจำลองการทำงาน (ในที่นี้ชื่อว่า lab03.elf)



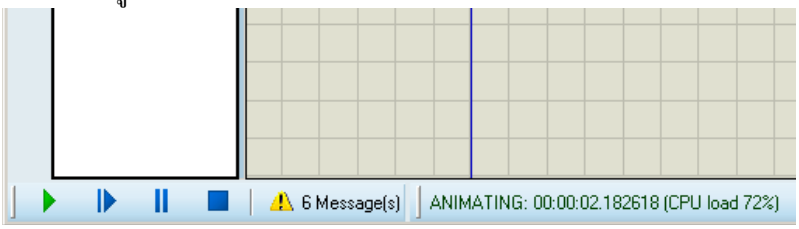
15. เมื่อทำการเลือกโปรแกรมที่ต้องการจำลองเข้ามาแล้ว จะได้ตามรูปด้านล่าง



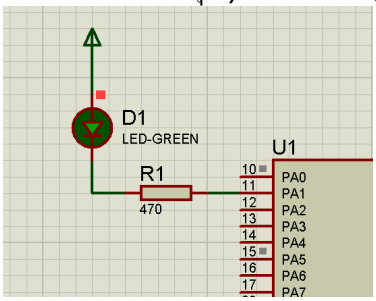
16. เมื่อต้องการสั่ง run การจำลองการทำงานให้กดปุ่ม play(1) (simulate) หรือสั่งหยุดการจำลองโดยปุ่ม stop(2)



17. รูปแสดงเมื่อโปรแกรมกำลังถูกจำลองการทำงาน



18. ขณะจำลองการทำงานหลอดแอลอีดีสีเขียวจะกระพริบทุกๆ 0.5 วินาที (เวลาในการจำลองไม่ใช่เวลาจริงที่รู้สึก)



<u>คำถามท้ายใบงาน</u>

1. ให้นิสิตทำการจำลองโปรแกรมโดยใช้ Proteus 8.9 เพื่อสั่งงานให้ LED กระพริบทุกๆ 1 วินาที โดยมี LED ต่อวงจรดังนี้

- LED สีเขียวอยู่ที่ PB0

(ในการส่งตรวจให้อัดวีดีโอในการทดลองตั้งแต่ต้นจน simulate เสร็จสิ้น)