13th-Dec-2020

Winter term 2020/21

**Networked Embedded Systems**

**Homework 02 : Energy Consumption**

**Task 1:** Toggle red LED for every 1s

We implemented the protothread (led\_pt) which starts automatically.

* A Timer from etimer module is initiated for every second using the CLOCK\_SECOND constant and the function etimer\_set().
* process waits for the etimer to get expired. This was verified using the function etimer\_expired().
* Each time the etimer is expired the red LED is toggled using the function leds\_toggle(LEDS\_RED)
* After toggling, the timer is again reset to CLOCK\_SECOND using the function etimer\_set().

Protothread Implementaion:

Text

Description automatically generated

Output 1: led off

Graphical user interface, text, application

Description automatically generated

Output 2: led on

Graphical user interface, application

Description automatically generated

**Task 2:** Toggle red LED for unequal time intervals on time – 1s / off time – 0.5s

We extended the above implementation by adding a flag variable (i) to track the state of led(on/off) and based on the state of the led the timer is reset to new time using the etimer\_set() function. Here, leds\_on and leds\_off is used to toggle the LED on and off.

Protothread Implementation:

Text

Description automatically generated

Output 1: led off

Graphical user interface, application

Description automatically generated

Output 2: led on

Graphical user interface, application

Description automatically generated

**Task 3:** Toggle the LED in different Brightness level (10% -> 50%->90%->10%..) using PWM on click of a button

We extended the above the implementation by adding one more protothread(btn\_pt) which listen for the button press event. When the button is pressed the brightness level is toggled to next level(10 - > 50 -> 90 -> 10 ->50 -> ,….).

Code:

A picture containing text, screen, screenshot

Description automatically generated

* + Next, the on / off time for the brightness level is calculated dynamically in change\_time( ) method based on the brightness level which is duty\_cycle using the formula
    - *T= 1 / frequency;*

*period = time\_on + time\_off;*

* + - *duty\_Cycle = (time\_on / period ) \* 100;*
    - *time\_on = (duty\_Cycle \* period) / 100;*
    - *time\_off = period – time\_on;*

*code:*

*Text

Description automatically generated*

* + Since, time\_on, time\_off and duty\_cycle variables are static. The updated values variables can be accessed from all the threads.
  + Next, the on and off time of etimer which was implemented in the last task in the protothread(led\_pt) will be updated for each button press and LED’s will blink according to the duty cycle.

Code:

Text

Description automatically generated

Output 1: 10%

Graphical user interface, text, application

Description automatically generated

Output 2: 50%

Graphical user interface, application

Description automatically generated

Output 3: 90%

Graphical user interface, application

Description automatically generated

**Task 4:** Calculate and print the Energy consumption of CPU and LED for every second for the above implementation Using energest module

We extended the above implementation by adding one more protothread(energy\_pt)

Which will be calulating the energy consumption as below.

* + Intially, energest is initialized by #define ENERGEST\_CONF\_ON 1 in global config and energest\_init() inside the protothread (energy\_pt).
  + Next, initial active time of CPU and LED is noted using the function energest\_type\_time() and the constants ENERGEST\_TYPE\_CPU for CPU active time and ENERGEST\_TYPE\_LED\_RED for LED active time.
  + A Timer using etimer is initialized for every second using the constant CLOCK\_SECOND.
  + When the timer expires, the new active time of CPU and LED is noted as described above.
  + Now, To find the active time of CPU and LED in last second, The difference of new active time and initial active is calculated.
  + This is converted to milliseconds by multiplying by 1000 and diving it by the constant RTIMER\_SECOND because the energest module will provide the output in time ticks which needs to be converted to milliseconds.
  + Energy consumption is calculated by multiplying the voltage , current and active\_time of the component. Voltage and current for the component is available in the datasheet of the microcontroller. The energy consumption is printed in mJ and Time is printed in ms.
  + Finally, The initial active time value is updated with the new active time to calculate the active time for next second.

Prothothread Implementation:

Text

Description automatically generated

Output:

Graphical user interface, application

Description automatically generated