NIELIT, Gorakhpur

Course Name: A-level (1st Sem.) **Subject: IoT**

Topic: Time and Interrupts in Arduino **Date: 23.04.2020**

Let's see some basic functions related to time and interrupts that are frequently used in Arduino IDE.

Time Functions

a) millis()

- This function returns the number of milliseconds passed since the Arduino board started running the current program.
- This number overflows (rolls back to zero) after approximately 50 days.
- Value returned by millis is an unsigned long int.
- Example unsigned long time
  time = millis()

b) micros()

- This function returns the number of microseconds passed since the Arduino board started running the current program.
- This number overflows (rolls back to zero) after approximately 70 minutes.
- Value returned by micros is an unsigned long int.
- Example unsigned long time
  time = micros()

c) delay(value)

- value : the number of milliseconds to pause the program.
- This function pauses the program for the number of milliseconds specified.

d) delayMicroseconds(value)

- value : the number of microseconds to pause the program.
- This function pauses the program for the number of microseconds specified.

Printing time elapsed in microseconds and milliseconds on the serial monitor of Arduino from the time the Arduino is powered on

```
/* Printing time elapsed in microseconds and milliseconds on the serial monitor of Arduino fr
om the time the Arduino is powered on */
```

```
unsigned long time_value;
/* Setup is run once at the start (Power-On or Reset) of sketch */
void setup()
{
  pinMode(13, OUTPUT); /* Pin 13 is defined as Output */
  Serial.begin(9600); /* opens serial port, sets data rate to 9600 bps */
}
/* Loop runs over and over after the startup function */
void loop()
{
  digitalWrite(13, HIGH); /* Make pin 13 HIGH */
  delay(2000); /* Wait for 2 seconds */
  digitalWrite(13, LOW);
  delayMicroseconds(100); /* Wait for 100 microseconds */
  Serial.print("Time passed since power on in milliseconds:");
  time_value = millis(); /* Read value of number of microseconds since program started exec
uting */
  Serial.println(time_value);
  delay(2000);
  Serial.print("Time passed since power on in microseconds:");
  time_value = micros(); /* Read value of number of milliseconds since program started exec
uting */
  Serial.println(time_value);
  delay(1000);
}
```

Interrupt Functions

a) interrupts()

- This function re-enables interrupts after they have been disabled by noInterrupts().

b) noInterrupts()

- This function disables interrupts.
- Interrupts can be re-enabled using interrupts().

c) attachInterrupt(digitalPinToInterrupt(pin), ISR, mode)

- The first parameter to attachInterrupt is pin number.
  digitalPinToInterrupt(pin) must be used to specify pin on which external interrupt is used.
  pin is the pin number to which external interrupt functionality is to be enabled.
  For Arduino UNO, only pins 2 and 3 can be used for external interrupts.
- ISR : The ISR(Interrupt Service Routine) to call when an interrupt event occurs. This function must take no parameters and return nothing.
- mode : Decides which event should cause an interrupt. 4 options are available to select from.

  i) LOW
     Interrupt generated whenever pin is LOW.
  ii) CHANGE
     Interrupt generated whenever pin changes value.
  iii) FALLING
     Interrupt generated whenever pin changes from HIGH to LOW.
  iv) RISING
     Interrupt generated whenever pin changes from LOW to HIGH.

d) detachInterrupt(digitalPinToInterrupt(pin))

- Turns off the interrupt specified by the digitalPinToInterrupt(pin)

Turning interrupts on and off

```
/* Interrupt Functions */
/* Setup is run once at the start (Power-On or Reset) of sketch */
void setup()
{
}
/* Loop runs over and over after the startup function */
void loop()
{
  noInterrupts();
  /* critical, time-sensitive code here */
  interrupts();
  /* other code here */
}
```

Toggling LED connected to pin 13 of Arduino using interrupt events on pin 2 of Arduino. A tactile switch is used to cause the interrupt events

```
/* Toggling LED connected to pin 13 of Arduino using interrupt events on pin 2 of Arduino.
A tactile switch is used to cause the interrupt events. */


volatile bool led_state = LOW;

/* Setup is run once at the start (Power-On or Reset) of sketch */

void setup()

{

  pinMode(13, OUTPUT); /* Pin 13 is defined as Output */

  pinMode(2, INPUT_PULLUP); /* Interrupt pin */

  /* Connect pin 2 to ground using a tactile switch between the pin and ground */

  attachInterrupt(digitalPinToInterrupt(2), blink, FALLING); /* Call ISR blink whenever a fal
ling edge event is detected on pin 2 */

  /* Press switch to cause falling edge event on the pin */

}


/* Loop runs over and over after the startup function */

void loop()

{

  digitalWrite(13, led_state); /* Write state value to pin 13 which has on-board LED */

}

/* ISR */

void blink() {

  led_state = !led_state; /* Toggle state value of LED */

}
```