Arduino Based Multitasking

A mini-project report submitted for

Embedded System and RTOS (Semester VI)

by

Aarti Bandekar (Roll No. 8296)

Mounita Bhagat (Roll No. 8453)

(Roll No.)

$\boxed{\textbf{(Group Code: 01)}}$

Under the guidance of

Prof. Sapna Prabhu

(Sign with date)

DEPARTMENT OF ELECTRONICS ENGINEERING

Fr. Conceicao Rodrigues College of Engineering

Bandra (W), Mumbai - 400050

University of Mumbai

Apri 18, 2020

Mini-Project Report - (mini_project/EsRtos/2019-20/01)

Arduino Based Multitasking

Aarti Bandekar[1], Mounita Bhagat[2],

Abstract
This project deals with the Arduino Based Multitasking with the following applications (loops executing at different rates) : 1. Motors running in the main loop. 2. PIR sensed LED Toggle, headlight indicator when vechicle approaching. 3. LDR based LED ON/OFF, headlight turns on when darkness is detected.

Keywords
ARTe Sketch, Arduino, LED

[1] *Fr. Conceicao Rodrigues College Of Engineering*
[2] *Department of Electronics Engineering*
[3] *Bandra, Mumbai*

Contents

1. Introduction

Multitasking term used in a modern computer system. Multitasking is a logical extension of a multiprogramming system that supports multiple programs to run concurrently. In a multitasking operating system, more than one task is executed at the same time. In this technique, multiple tasks, also known as processes, share common processing resources such as a CPU. Arduino does not support concurrency and a program is limited to a single block of instructions cyclically repeated. The instructions written in a particular section of the program are cyclically repeated sequentially. This means that it is not possible run processes in parallel (Multitasking) or activities in Real-Time mode. In order to run periodic processes with deadlines, it is necessary to implement a sequential program that, at each cycle, controls what activity to be performed. In order to run periodic processes with deadlines, it is necessary to implement a sequential program that, at each cycle, controls what activity to be performed. To overcome the limitation explained above, a special version of the Arduino framework has been used, called ARTe (Arduino Real-Time Extension). ARTe, which introduces a support for implementing concurrent real-time applications in the standard Arduino framework with a minimal impact on the original programming model. In addition to the classical Arduino programming model, consisting in a single main loop containing the code to be executed, ARTe allows the user to specify a number of different loops, each to be executed with a given desired period.

2. ARTe (Arduino Real-Time extension)

ARTe (Arduino Real-Time extension) is an extension to the Arduino framework that supports multitasking and real-time preemptive scheduling. Thanks to ARTe, the user can easily specify and run multiple concurrent loops at differents rates, in addition to the single execution cycle provided by the standard Arduino framework. Today ARTe supports the most popular platforms: Arduino UNO and Arduino DUE. In addition to the single loop present in the standard Arduino approach, the user can easily specify a number of concurrent loops to be executed at specific rates. Concurrency and real-time scheduling is provided by the ERIKA Enterprise (http://erika.tuxfamily.org/drupal/) open-source real-time kernel. Loops can use the standard Arduino libraries, which have been enhanced to guarantee mutual exclusion in the access of shared data structures. The impact of ARTe in terms of footprint and runtime overhead has beed evaluated by extensive tests and resulted to be negligible. You can use ARTe and develop Arduino code only with Windows platforms.

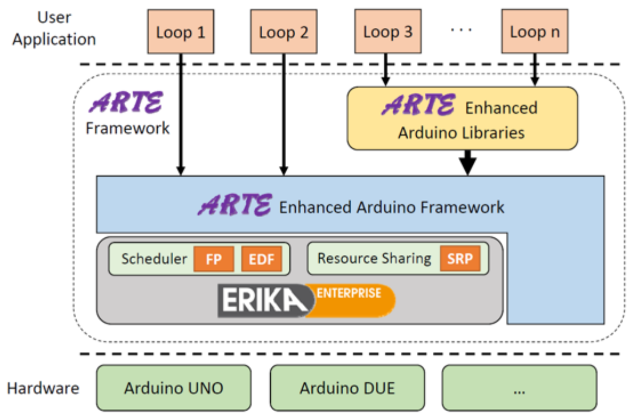
2.1 ARTe Installation

In order to develop applications with ARTe you have to download the development tool. The version of ARTe used to develop this project can be found on the following web page: http://retis.sssup.it/?q arte ARTe is evolved and the new version can be downloaded from the site:

http://arte.retis.santannapisa.it/index.html .

The developer site provides all the information on how to use the development tool, providing documentation and examples. Once installed the tool is quite similar to Arduino's editor but with the ability to use the Arte Extension. You can then enable or disable the ARTe extension by clicking on the Tools item in the menu bar. The item ARTe will appear and click on it you can select enable or disable.

2.2 ARTe Architecture



The ARTE architecture.

3. Working

ARTe allows us to implement a system with 3 periodic processes, however for this project we will be using only 2 periodic processes along with standard loop function. The loops will be executing at different time period.

1. Motors running in the same loop :

The Motor Driver is a module for motors that allows you to control the working speed and direction of two motors simultaneously.This Motor Driver is designed and developed based on L293D IC.L293D is a 16 Pin Motor Driver IC. This is designed to provide bidirectional drive currents at voltages from 5 V to 36 V. Output pin 1 and 2 of motor driver L293D are connected to motor 1 and output pins 3 and 4 of L293D are connected to motor 2. Pin no. 4 and 5 of Arduino are connected to the input pins 1 and 2 of L293D. And pin no. 10 and 11 of Arduino are connected to input 3 and 4 of motor driver L293D.

3.1 Motors running in the main loop

```
/* Arduino based multitasking with the following applications:
(loops executing at differenct time periods)

1. Motors running in the main loop
2. PIR sensed LED Toggle ,headlight indicator when vehicle approaching
3. LDR based LED ON/OFF, headlights turn on when darkness is detected
*/

//Pin definition

int a;
const int ledPin = 9;
int ledState = LOW;                // ledState used to set the LED


int sensorPin = A0; // select the input pin for the LDR
int sensorValue = 0;
// variable to store the value coming from the sensor
int LDRled = 3;


static int outputPin1=5;   // PINS FOR MOTOR
static int outputPin2=6;
static int outputPin3=10;
static int outputPin4=11;

void setup() {
  //Serial communication
  Serial.begin(9600);

  //Pin setup
 pinMode(ledPin,OUTPUT); //OUTPUT TO GLOW LED
  pinMode(7,INPUT);  //INPUT FROM PIR

 pinMode(LDRled, OUTPUT);

  pinMode(outputPin1, OUTPUT);
  //Setting the dc motor pins as output to L293D
  pinMode(outputPin2, OUTPUT);
  pinMode(outputPin3, OUTPUT);
  pinMode(outputPin4, OUTPUT);
```

This loop is responsible for always rotating the motors and it follows the following logic chart :

After which we provide it with a 20 sec delay.

3.2

```
}

void loop() {

  //This loop is responsible for always rotating
  the motors and it follows the following logic chart:

  /*Input 1     Input 2               Result
    0            0                    Stop
    0            1                    Anti Clockwise
    1            0                    Clockwise
    1            1                    Stop
*/
    digitalWrite(outputPin1, HIGH);
    digitalWrite(outputPin2, LOW);
    digitalWrite(outputPin3, HIGH);
    digitalWrite(outputPin4, LOW);
    delay(20000);
}
```

3.3 PIR sensed LED toggle, headlight indicator when vechicle approaches.

```
void loop1(1000){
  //This task uses a PIR to sense a forthcoming vehicle
  and toggle the headlights

  //Send a HIGH pulse to the trigger pin

//This loop will toggle the LEDs When PIR detects object

  a=digitalRead(7);  //this variable is set to read the
  pir inputs and it gets stored here
  Serial.println(a);


  if(a==1){  //PIR has detected object
  for(int i=0; i<5;i++)
  {
  digitalWrite(ledPin, HIGH);
  delay(100);
  digitalWrite(ledPin, LOW);
  delay(100);
  }

  }
  else
  digitalWrite(ledPin,LOW);


}
```

This loop runs at a rate of 1sec.This task uses a PIR to sense a forthcoming vehicle and toggles the headlights.PIR sensor detects a moving object around within approximately 10m from the sensor. This is an average value, as the actual detection range is between 5m and 12m.PIR are fundamentally made of a pyro electric sensor, which can detect levels of infrared radiation. For numerous essential projects or items that need to discover when an individual has left or entered the area. PIR sensors are incredible, they are flat control and minimal effort, have a wide lens range, and are simple to interface with. The sensor value from the out pin of the PIR is stored in a variable ”a”. When the PIR detects an object it outputs a 5V signal to the ARDUINO and triggers an interrupt owing to which the led will toggle with a delay of 0.1 sec, else the led will remain off.

3.4 LDR based ON/OFF, headlights turn ON when darkness is detected.

```
void loop2(1500) {

  // This task senses the value of the LDR
  and glows the LED accordingly

  // Application : Headlights turn on when it gets dark,
  automated headlights


sensorValue = analogRead(sensorPin);

Serial.println(sensorValue);

    if (sensorValue < 100)

    {

      Serial.println("LED light on");

      digitalWrite(LDRled,HIGH);

       delay(1000);

    }

      digitalWrite(LDRled,LOW);

       delay(sensorValue);
       // delay to make output readable on serial monitor.

}
//}

void loop3(11000) {
  // NO APPLICATION HERE

}
```

This loop runs at a rate of 1.5sec This task senses the value of the LDR and glows the LED accordingly, when the darkness is detected by the LDR then the LED glows and vice versa. Application : Headlights turn on when it gets dark, automated headlights

To achieve this we use the Light Dependent Resistor(LDR)

Hardware Connections :

1. Arduino pin 3 connected to LED +ve
2. Arduino GND connected to LED -ve through 10k
3. Arduino +5v is connected to LDR One End
4. Arduino A0 pin is connected to LDR other end
5. Arduino GND is connected to LDR other end with 10k

A Light Dependent Resistor (also known as a photoresistor or LDR) is a device whose resistivity is a function of the incident electromagnetic radiation. Hence, they are light-sensitive devices. Photoresistors work based off of the principle of photoconductivity. Photoconductivity is an optical phenomenon in which the material’s conductivity is increased when light is absorbed by the material.Photoresistor LDR’s are light-dependent devices whose resistance is decreased when light falls on them and that is increased in the dark. When a light dependent resistor is kept in dark, its resistance is very high. This resistance is called as dark resistance. It can be as high as 1012 ohms and if the device is allowed to absorb light its resistance will be decreased drastically. If a constant voltage is applied to it and the intensity of light is increased the current starts increasing.

The input from the LDR is input on pin A0 ,an analog pin of the arduino because voltage is very easy for the Arduino to measure, while resistance is not, and most sensors such as

Photoresistor (LDR), flex sensor, thermistors and more - are actually a variable resistors.

The main reason it's hard to measure resistance changes is that the Arduino (and most IC) contain a tiny system called Analog to Digital Converter (ADC). This system translates changes in analog voltage to a series of 1's and 0's that can be in turn converted to an Integer for example.

The ADC is designed to read voltage changes, and if we want to use the Arduino's analogRead (which utilizes the ADC) to get the Photoresistor readings for example, we will need a way to convert the changes in resistance to changes in voltage - and a voltage divider is the easiest way to do it.

It is true that the sensor is already a resistor, and as such it should change the voltage across it. But you would have trouble measuring the voltage changes, since there is no reference point except for Vcc (5V) and Ground.In contrary, when using a voltage divider, you have a well defined reference point to measure the voltage changes.

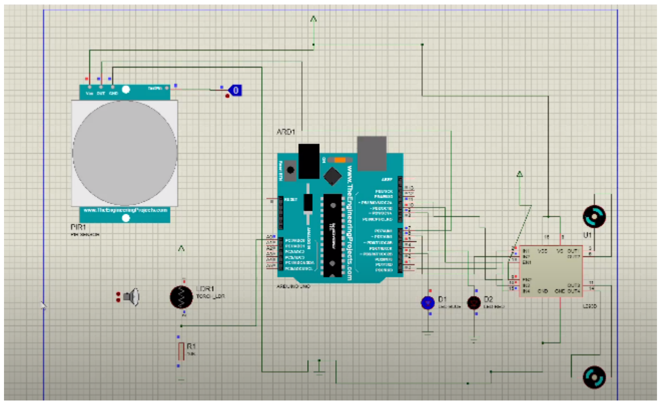
This value is stored in a variable named sensorValue, if the input from the LDR is below 100, as its name implies, the Light Dependent Resistor (LDR) is made from a piece of exposed semiconductor material such as cadmium sulphide that changes its electrical resistance from several thousand Ohms in the dark to only a few hundred Ohms when light falls upon it by creating hole-electron pairs in the material.

The net effect is an improvement in its conductivity with a decrease in resistance for an increase in illumination. Also, photoresistive cells have a long response time requiring many seconds to respond to a change in the light intensity ,the led turns on otherwise it remains off ,however this loop runs at a rate of 1.5 sec.

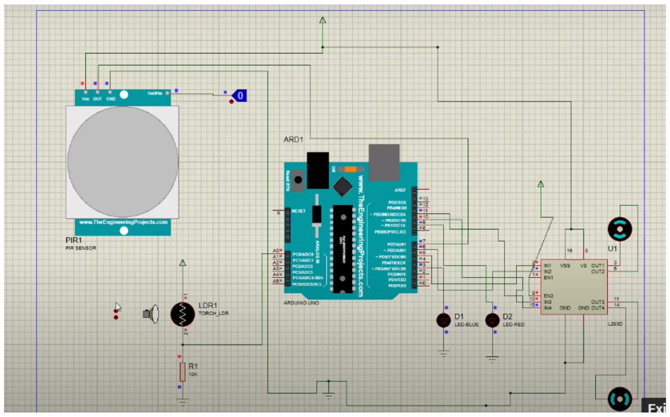
The ARTe programming model is as similar as possible to the original Arduino programming model. Each periodic loop defined by the user is specified as shown in 3, where i equals to 1, 2, 3, . . . and period represents the time interval (in milliseconds) with which the loop is executed. As in the original Arduino programming model, the setup() function is also available under ARTe with the same syntax and semantics. Similarly, the original loop() function can also be used under ARTe, offering the programmer the possibility to execute background activities when no other pending loops are running.

4. Output

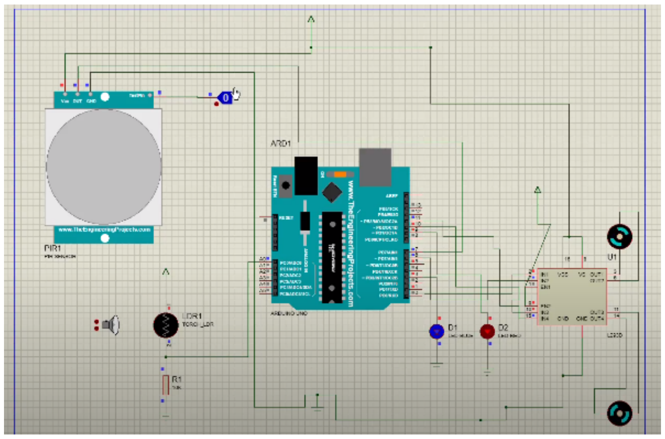
4.1 Motors running in the main loop



4.2 PIR sensed LED toggle, headlight indicator when vechicle approaches



4.3 LDR based ON/OFF, headlights turn ON when darkness is detected



5. Reference

1. https://create.arduino.cc/projecthub/Menphis/multitasking-and-real-time-arduino-system-789d6f
2. http://arte.retis.santannapisa.it/index.html
3. https://create.arduino.cc/projecthub/104900/getting-started-with-real-time-multi-tasking-in-arte-86494b

4. https://www.hackster.io/Menphis/multitasking-and-real-time-arduino-system-789d6f

5. https://howtomechatronics.com/tutorials/arduino/how-pir-sensor-works-and-how-to-use-it-with-arduino/

6. https://www.youtube.com/watch?v NpAYjpWYMuw feature youtu.be

7. https://www.instructables.com/id/Auotmatic-Street-lights-control-using-LDR-and-Ardu/

8. https://roboindia.com/tutorials/motor-driver-arduino/

GitHub References :

1. https://github.com/melvin-mancini/Multitasking-RealTime-Arduino-System/blob/master/README.md ARTe-installation

2. https://github.com/melvin-mancini/Multitasking-RealTime-Arduino-System readme

Libraries Used :

1. https://www.theengineeringprojects.com/2016/01/pir-sensor-library-proteus.html

2. https://etechnophiles.com/add-arduino-library-proteus-simulate-arduino-projects-2018-edition/