**COMSATS University**

**Islamabad**



**Lab Report # 08**

**Real Time Embedded Systems**

**(EEE-446)**

|  |
| --- |
| **Implementation of I2C with Devices.** |

**Submitted By:**

**Arwa Aamir (FA16-EEE-002)**

**Submitted To:**

**Dr. Ahsen Malik**

# Lab # 08

## Implementation of I2C with Devices.

### Objectives

* Understand the working of I2C communication.
* Implementing of I2C with digital potentiometer and timer IC.

### Tools

* Arduino
* Proteus ISIS

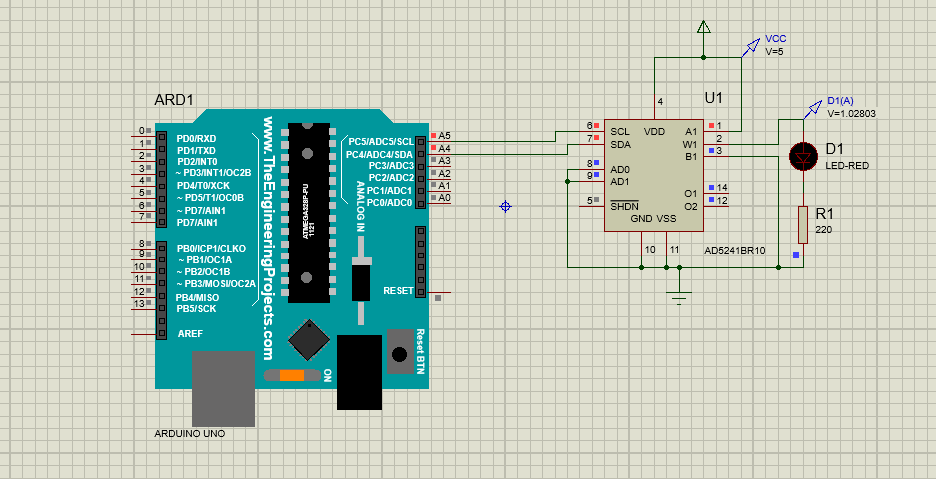
### In-Lab Task 1:

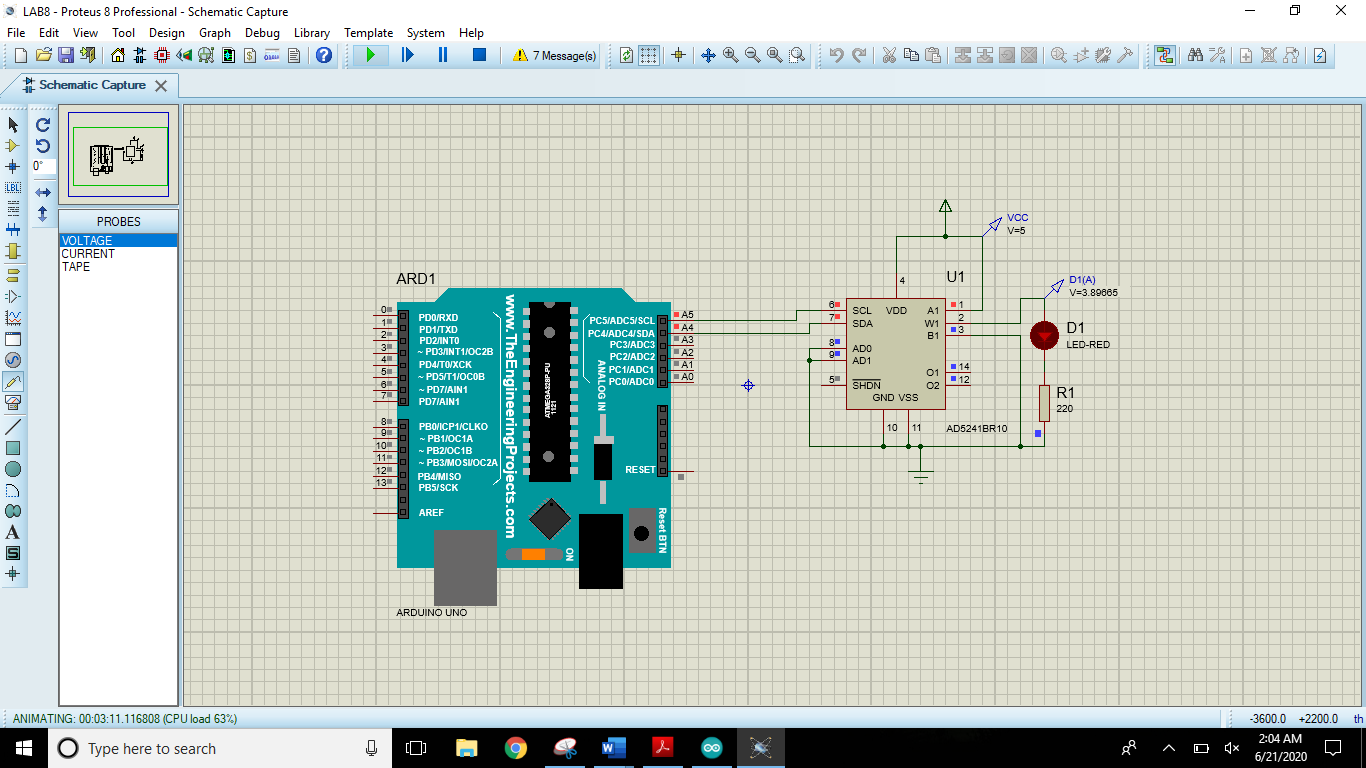
**Implement the task of interfacing digital potentiometer AD5241with Arduino**

**ARDUINO IDE CODE:**

|  |
| --- |
| // I2C Digital Potentiometer interfacing with Arduino  // Controls AD5241 digital potentiometer via I2C/TWI  #include <Wire.h>  void setup()  {  Wire.begin(); // join i2c bus (address optional for master)  }  byte val = 0;  void loop()  {  Wire.beginTransmission(44); // transmit to device #44 (0x2c)  // device address is specified in datasheet  Wire.write(byte(0x00)); // sends instruction byte  Wire.write(val); // sends potentiometer value byte  Wire.endTransmission(); // stop transmitting  val++; // increment value  if (val == 255)  { // if reached max value 255  val = 0; // start over from lowest value  }  delay(100);  } |

**PROTEUS SIMULATIONS:**

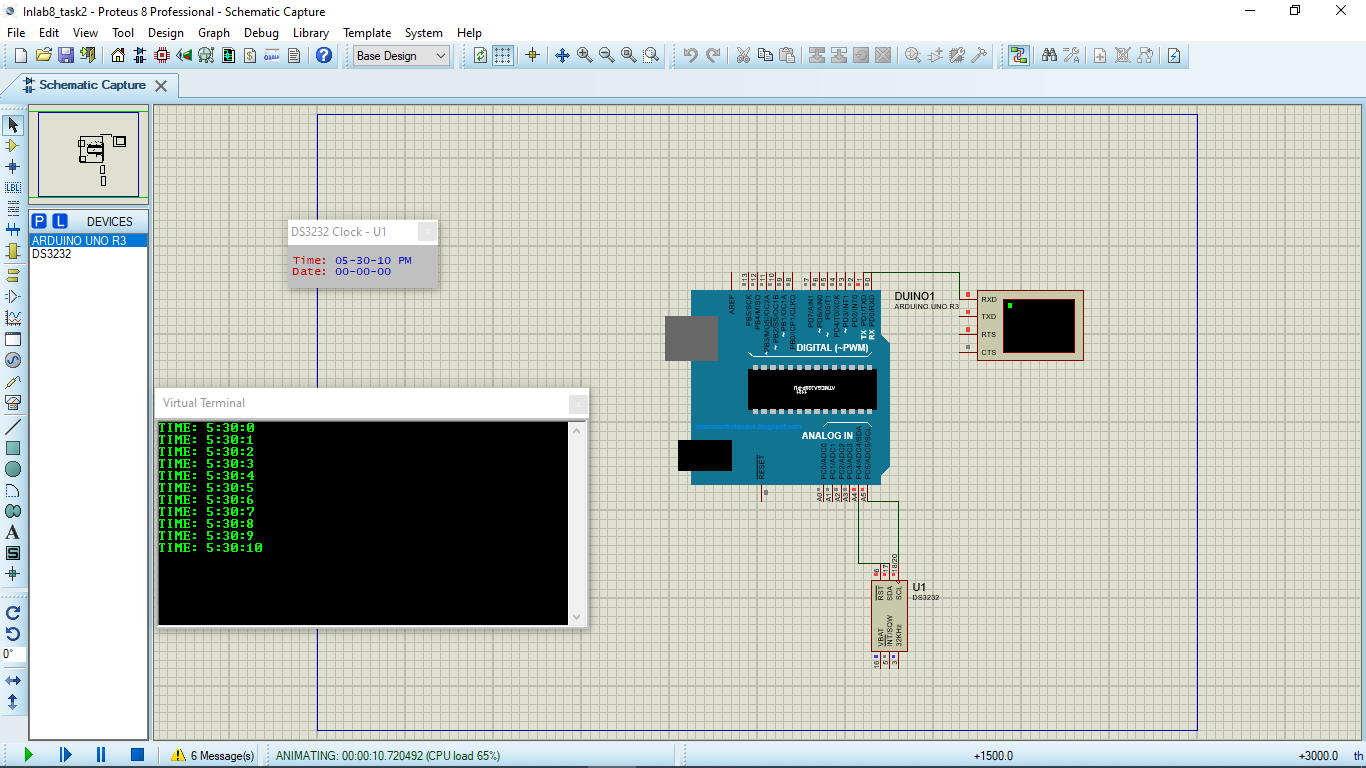




### In Lab Task 2:

**Implement the I2C to interface DS3232 with Arduino and update current time in time-keeping register and show it on attached virtual terminal**

**PROTEUS SIMULATIONS:**



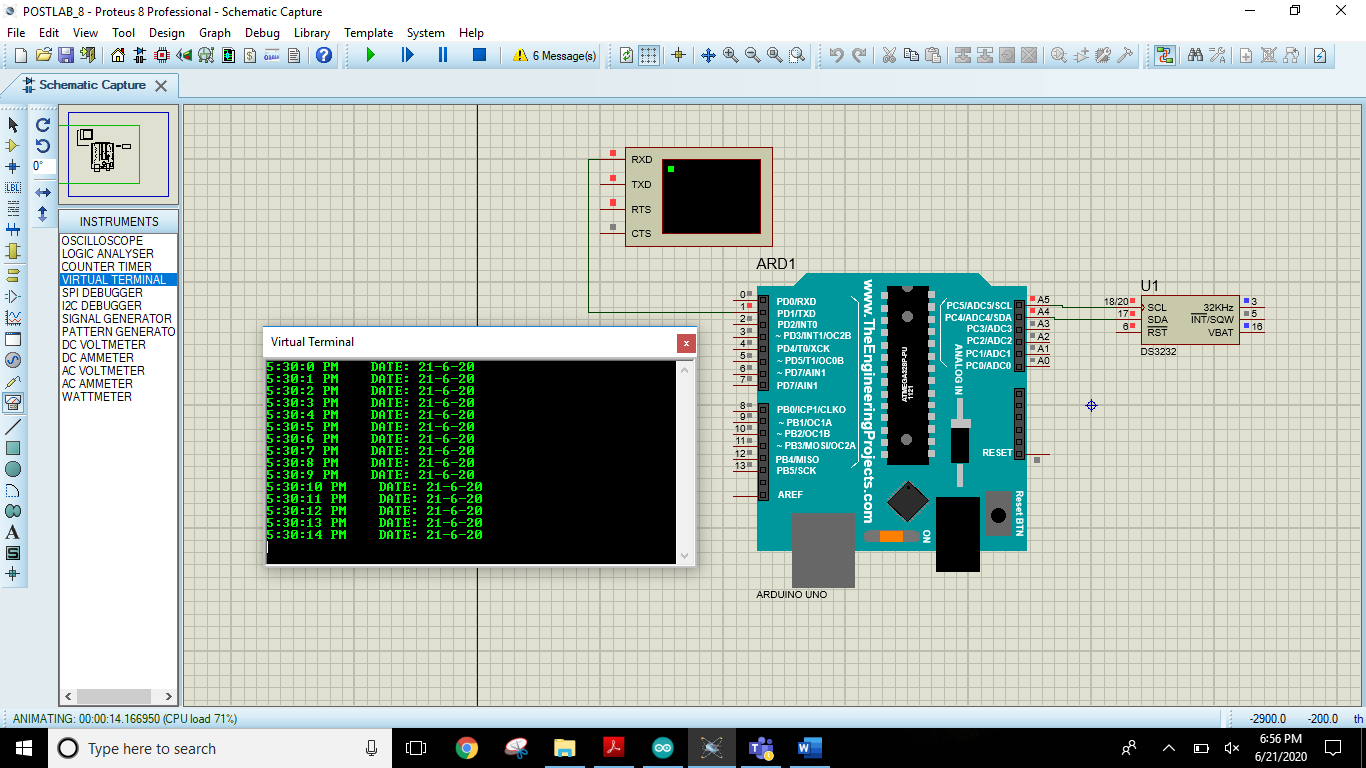
**ARDUINO IDE CODE:**

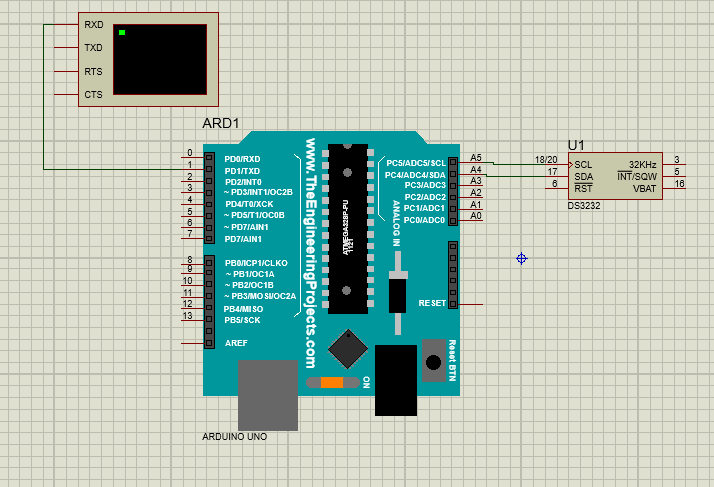
|  |
| --- |
| #include<Wire.h>  byte sec\_bcd, min\_bcd, h\_bcd;  unsigned int seconds\_1, minutes\_1, hour\_1;  void set\_time() {  Wire.beginTransmission(104);  Wire.write(byte(0x00));  Wire.write(0);  Wire.write(byte(0x30));  Wire.write(byte(0x65));  Wire.endTransmission();  }  void set\_date(){  Wire.beginTransmission(104);  Wire.write(byte(0x00));  Wire.write(0);  Wire.write(byte(0x30));  Wire.write(byte(0x65));  Wire.endTransmission();  }  void read\_time() {  Wire.beginTransmission(104);  Wire.write(0x00);  Wire.endTransmission();  Wire.requestFrom(104,3);  sec\_bcd=Wire.read() & 0x7f;  min\_bcd=Wire.read() & 0x7f;  h\_bcd=Wire.read() & 0x1f;  seconds\_1=(sec\_bcd>>4)\*10 + (sec\_bcd & 0x0f);  minutes\_1=(min\_bcd>>4)\*10 + (min\_bcd & 0x0f);  hour\_1 =(h\_bcd>>4)\*10 + (h\_bcd & 0x0f);  }  void setup(){  Wire.begin();  Serial.begin(9600);  set\_time();  }  void loop(){  read\_time();  Serial.print("TIME: ");  Serial.print(String(hour\_1));  Serial.print(":");  Serial.print(String(minutes\_1));  Serial.print(":");  Serial.println(String(seconds\_1));  delay(1000);  } |

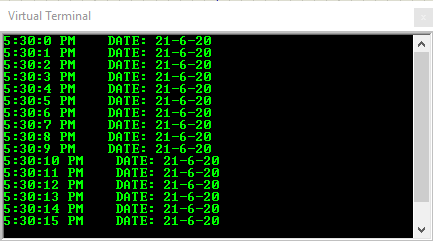
### POST LAB TASK:

**Store the current date and time on DS3232 IC and show it on attached 16x2 LCD. Time must contain AM/PM part. And year field must be complete i.e. 2020**

**PROTEUS SIMULATIONS:**





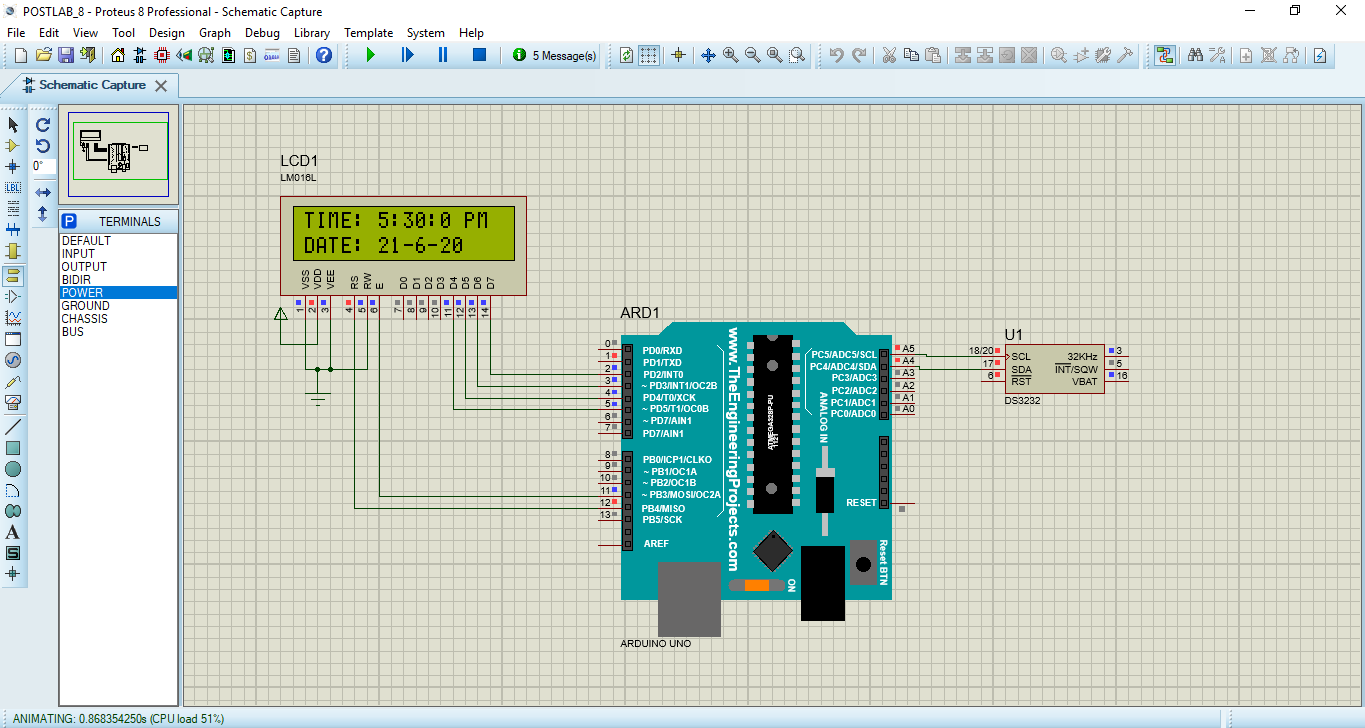


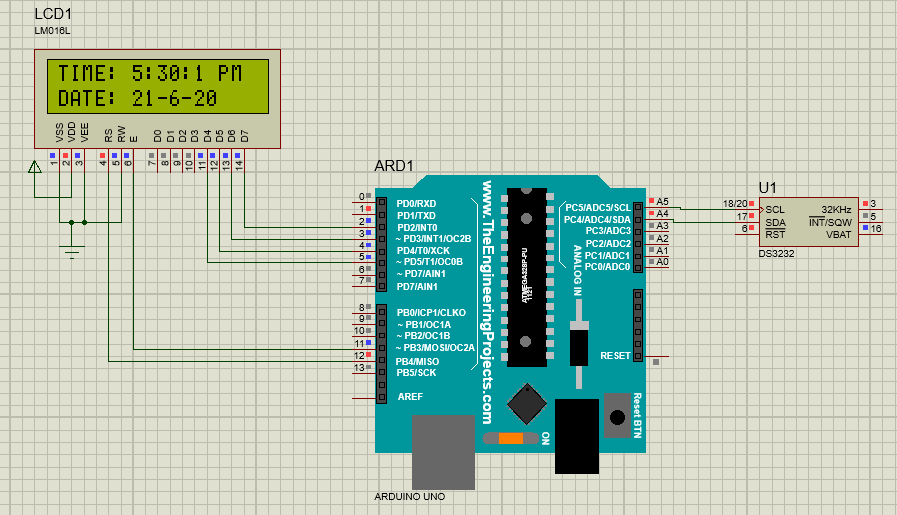
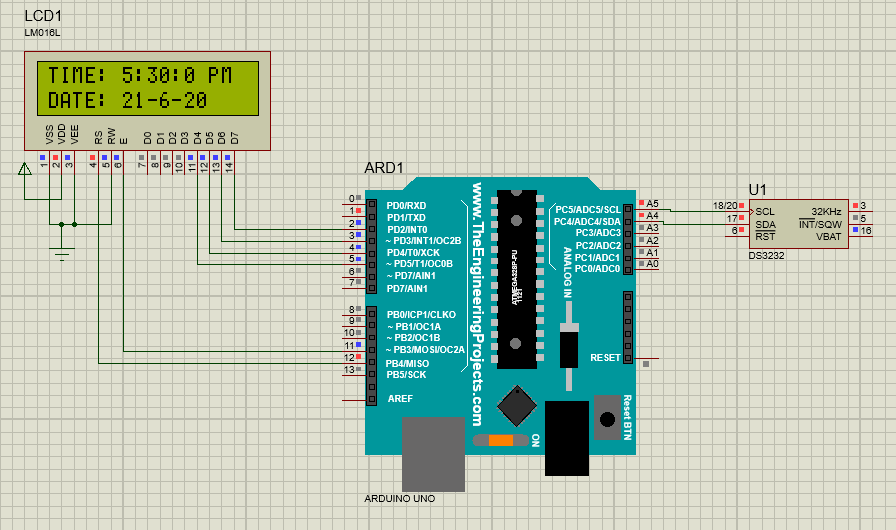
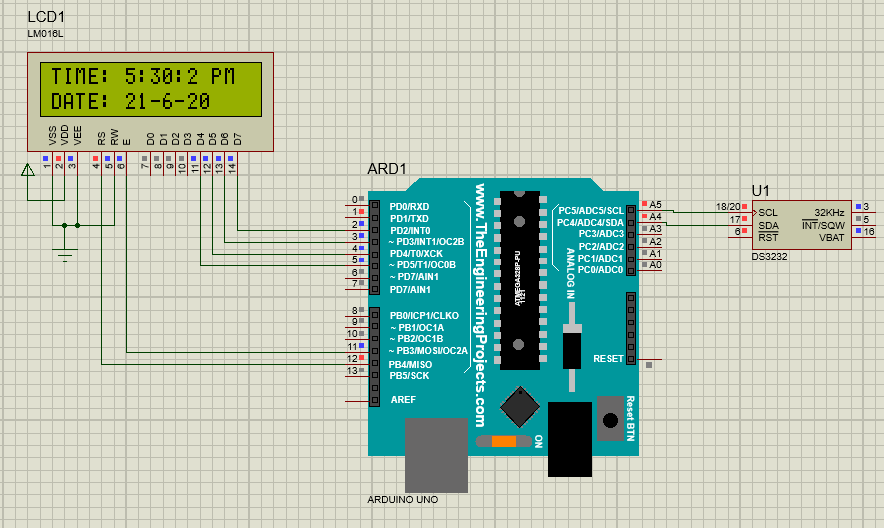
**ARDUINO IDE CODE:**

|  |
| --- |
| #include<Wire.h>  byte sec\_bcd, min\_bcd, h\_bcd, day\_bcd, date\_bcd, month\_bcd, year\_bcd, h\_val, am\_pm\_val;  unsigned int seconds\_1, minutes\_1, hour\_1, day\_1, date\_1, month\_1, year\_1;  void set\_data() {  Wire.beginTransmission(104); // START ADDRESS OF DEVICE  Wire.write(byte(0x00)); // START ADDRESS POINTER  Wire.write(0); // ADDRESS = 0x00 , DATA = 0 ( 0 SECONDS)  Wire.write(byte(0x30)); // ADDRESS = 0X01 , DATA = 0x30 ( 30 MINUTES)  Wire.write(byte(0x65)); // ADDRESS = 0X02 , DATA = 0x65 ( 5 PM IN 12 hour FORMAT) (HOURS)  Wire.write(byte(0x02));// ADDRESS = 0X03 , DATA = 0x02 (DAY = SUNDAY)  Wire.write(byte(0x21));// ADDRESS = 0X04 , DATA = 0x21 (DATE = 21)  Wire.write(byte(0x06));// ADDRESS = 0X05 , DATA = 0x06 (6th Month = June)  Wire.write(byte(0x20));// ADDRESS = 0X06 , DATA = 0x20 (Year 2020)  Wire.endTransmission();  }  void read\_data() {  Wire.beginTransmission(104);  Wire.write(0x00);  Wire.endTransmission();  Wire.requestFrom(104,7);  sec\_bcd=Wire.read() & 0x7f;  min\_bcd=Wire.read() & 0x7f;  h\_bcd=Wire.read() & 0x3f;  day\_bcd=Wire.read() & 0x07;  date\_bcd=Wire.read() & 0x3f;  month\_bcd=Wire.read() & 0x1f;  year\_bcd=Wire.read() & 0xff;    seconds\_1=(sec\_bcd>>4)\*10 + (sec\_bcd & 0x0f);  minutes\_1=(min\_bcd>>4)\*10 + (min\_bcd & 0x0f);  h\_val=h\_bcd & 0x1f;  hour\_1=(h\_val>>4)\*10 + (h\_val & 0x0f);  am\_pm\_val = h\_bcd>>5; // (0 = AM , 1= PM)  day\_1=day\_bcd;  date\_1=(date\_bcd>>4)\*10 + (date\_bcd & 0x0f);  month\_1=(month\_bcd>>4)\*10 + (month\_bcd & 0x0f);  year\_1=(year\_bcd>>4)\*10 + (year\_bcd & 0x0f);  }  void setup(){  Wire.begin();  Serial.begin(9600);  set\_data();  }  void loop(){  read\_data();  Serial.print(String(hour\_1));  Serial.print(":");  Serial.print(String(minutes\_1));  Serial.print(":");  Serial.print(String(seconds\_1));  Serial.print(" ");  if (am\_pm\_val>0)  {  Serial.print("PM");  }  else  {  Serial.print("AM");  }  Serial.print(" ");  Serial.print("DATE: ");  Serial.print(String(date\_1));  Serial.print("-");  Serial.print(String(month\_1));  Serial.print("-");  Serial.println(String(year\_1));    delay(1000);  } |

**WITH LCD:**

**PROTEUS SIMULATIONS:**





**ARDUINO IDE CODE:**

|  |
| --- |
| #include<Wire.h>  #include <LiquidCrystal.h>  const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;  LiquidCrystal lcd(rs, en, d4, d5, d6, d7);  byte sec\_bcd, min\_bcd, h\_bcd, day\_bcd, date\_bcd, month\_bcd, year\_bcd, h\_val, am\_pm\_val;  unsigned int seconds\_1, minutes\_1, hour\_1, day\_1, date\_1, month\_1, year\_1;  void set\_data() {  Wire.beginTransmission(104); // START ADDRESS OF DEVICE  Wire.write(byte(0x00)); // START ADDRESS POINTER  Wire.write(0); // ADDRESS = 0x00 , DATA = 0 ( 0 SECONDS)  Wire.write(byte(0x30)); // ADDRESS = 0X01 , DATA = 0x30 ( 30 MINUTES)  Wire.write(byte(0x65)); // ADDRESS = 0X02 , DATA = 0x65 ( 5 PM IN 12 hour FORMAT) (HOURS)  Wire.write(byte(0x02));// ADDRESS = 0X03 , DATA = 0x02 (DAY = SUNDAY)  Wire.write(byte(0x21));// ADDRESS = 0X04 , DATA = 0x21 (DATE = 21)  Wire.write(byte(0x06));// ADDRESS = 0X05 , DATA = 0x06 (6th Month = June)  Wire.write(byte(0x20));// ADDRESS = 0X06 , DATA = 0x20 (Year 2020)  Wire.endTransmission();  }  void read\_data() {  Wire.beginTransmission(104);  Wire.write(0x00);  Wire.endTransmission();  Wire.requestFrom(104,7);  sec\_bcd=Wire.read() & 0x7f;  min\_bcd=Wire.read() & 0x7f;  h\_bcd=Wire.read() & 0x3f;  day\_bcd=Wire.read() & 0x07;  date\_bcd=Wire.read() & 0x3f;  month\_bcd=Wire.read() & 0x1f;  year\_bcd=Wire.read() & 0xff;  seconds\_1=(sec\_bcd>>4)\*10 + (sec\_bcd & 0x0f);  minutes\_1=(min\_bcd>>4)\*10 + (min\_bcd & 0x0f);  h\_val=h\_bcd & 0x1f;  hour\_1=(h\_val>>4)\*10 + (h\_val & 0x0f);  am\_pm\_val = h\_bcd>>5; // (0 = AM , 1= PM)  day\_1=day\_bcd;  date\_1=(date\_bcd>>4)\*10 + (date\_bcd & 0x0f);  month\_1=(month\_bcd>>4)\*10 + (month\_bcd & 0x0f);  year\_1=(year\_bcd>>4)\*10 + (year\_bcd & 0x0f);  }  void setup(){  Wire.begin();  Serial.begin(9600);  lcd.begin(16,2); // Initializes the interface to the LCD screen, and specifies the dimensions (width and height) of the display }  set\_data();  }  void loop(){  read\_data();  lcd.print("TIME: ");  lcd.print(String(hour\_1));  lcd.print(":");  lcd.print(String(minutes\_1));  lcd.print(":");  lcd.print(String(seconds\_1));  lcd.print(" ");  if (am\_pm\_val>0)  {  lcd.print("PM");  }  else  {  lcd.print("AM");  }  lcd.setCursor(0,1);  lcd.print("DATE: ");  lcd.print(String(date\_1));  lcd.print("-");  lcd.print(String(month\_1));  lcd.print("-");  lcd.print(String(year\_1));  delay(1000); // 1 second delay  lcd.clear(); // Write a character to the LCD  } |

# Critical Analysis/Conclusion:

|  |
| --- |
| In this lab we learnt how to use I2C Serial Protocol, its benefits, and applications.  I2C Protocol uses only two lines SDA and SCK for data transfer and synchronization and can support up to 127 devices at a time. It is easier to daisy chain devices in I2C than in SPI. It can also support variable supply devices.  In this lab we used the AD5241 IC that is a variable potentiometer controlled using I2C protocol. It varied the luminescence of LED by varying the resistance.  We also used the IC DS3232 that is a Real time Clock IC that can also store dates, alarms, and temperature. We viewed the time and date on virtual terminal and LCD using this IC. |

