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

Pre Lab

Please read the theoretical background of I2C communication.

Digital potentiometer AD5241 interfacing with Arduino using I2C:

The AD5241 provide a single-channel, 256-position, digitally controlled variable resistor (VR) device. These devices perform the same electronic adjustment function as a potentiometer, trimmer, or variable resistor. VR offers a completely programmable value of resistance between the A terminal and the wiper, or the B terminal and the wiper. For the AD5241, the fixed A-to-B terminal resistance of 10 kΩ and has a 1% channel-to-channel matching tolerance.

Wiper position programming defaults to midscale at system power on. When powered, the VR wiper position is programmed by an I2C-compatible, 2-wire serial data interface.

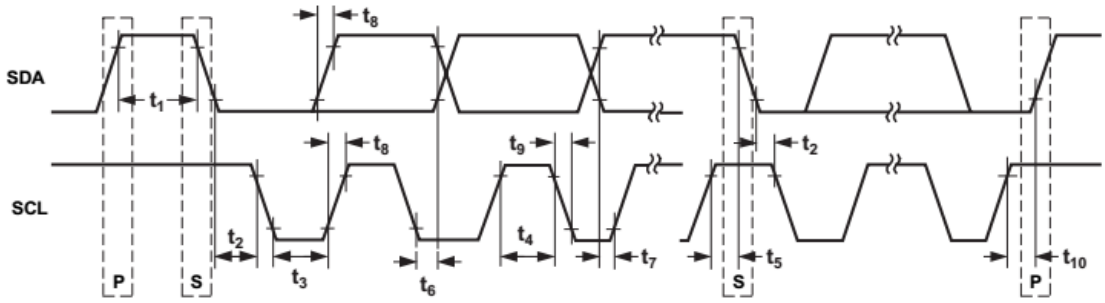

Figure 1: Timing Diagram

S	0	1	0	1	1	AD1	AD0	R/$\overline{W}$	A	$\overline{A}$/B	RS	SD	O_1	O_2	X	X	X	A	D7	D6	D5	D4	D3	D2	D1	D0	A	P
	Slave Address Byte									Instruction Byte									Data Byte									

where:
S = start condition
P = stop condition
A = acknowledge
X = don't care
AD1, AD0 = Package pin programmable address bits. Must be matched with the logic states at Pin AD1 and Pin AD0.
R/$\overline{W}$ = Read enable at high and output to SDA. Write enable at low.
$\overline{A}$/B = RDAC subaddress select; 0 for RDAC1 and 1 for RDAC2.
RS = Midscale reset, active high.
SD = Shutdown in active high. Same as $\overline{SHDN}$ except inverse logic.
O_1, O_2 = Output logic pin latched values
D7, D6, D5, D4, D3, D2, D1, D0 = data bits.

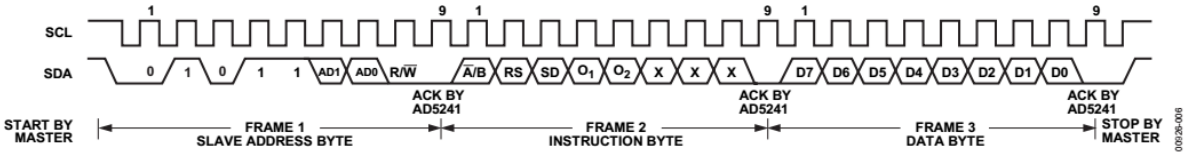

Figure 2 Writing format using I2C to change value of VR

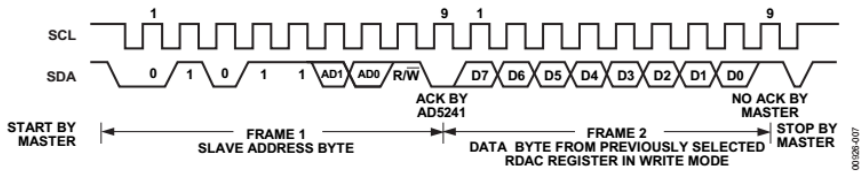

Figure 3: Read Signal using I2C to check current value

Figure 4 contains the circuit diagram of the way the connections are made using Arduino.

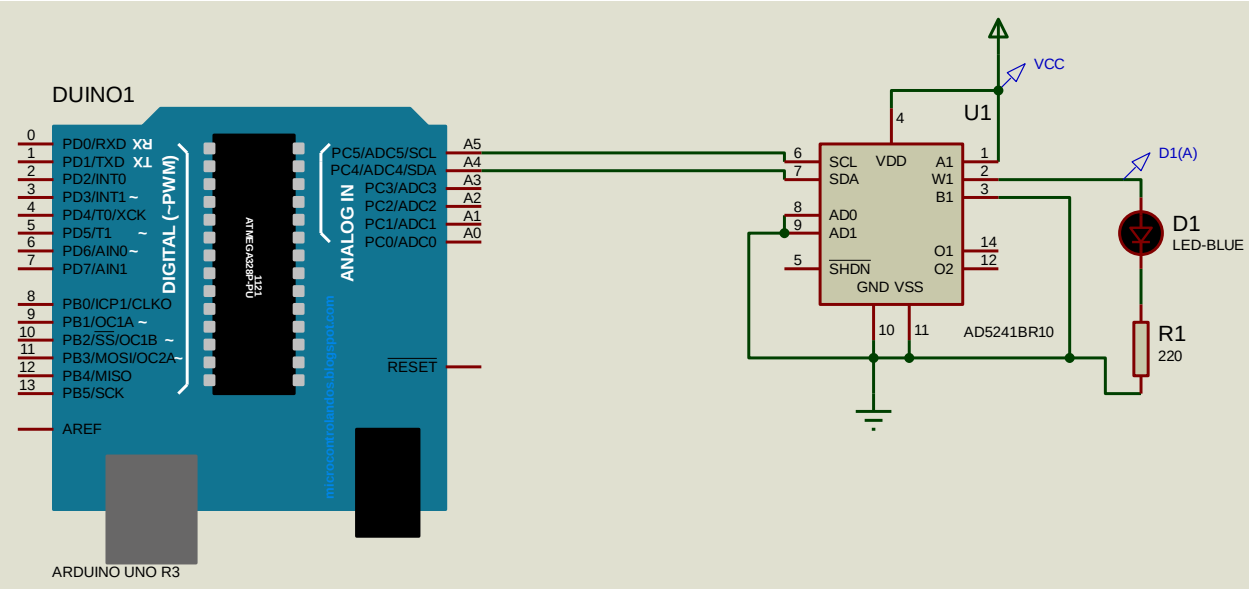

Figure 4: Proteus based circuit diagram

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);
}
```

In-lab Task 1:

Implement the above task of interfacing digital potentiometer AD5241with Arduino.

Interfacing DS3231 or DS3232 Timer IC with Arduino:

The DS3231 is a low-cost, extremely accurate I2C real-time clock (RTC). The RTC maintains seconds, minutes, hours, day, date, month, and year information. The date at the end of the month is automatically adjusted for months with fewer than 31 days, including corrections for leap year. The clock operates in either the 24-hour or 12-hour format with an AM/PM indicator. Two programmable time-of-day alarms and a programmable square-wave output are also provided. Address and data are transferred serially through an I2C bidirectional bus.

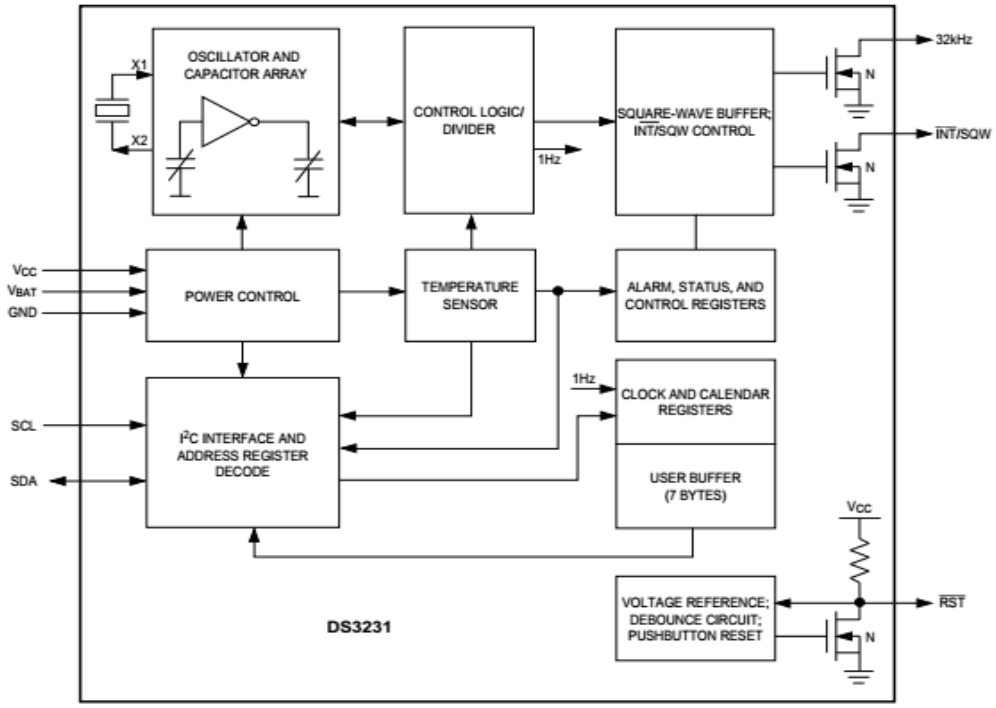

Figure 5: Internal block diagram of ds3231

ADDRESS	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0 LSB	FUNCTION	RANGE
00h	0	10 Seconds			Seconds				Seconds	00–59
01h	0	10 Minutes			Minutes				Minutes	00–59
02h	0	12/$\overline{24}$	$\overline{AM}$/PM 20 Hour	10 Hour	Hour				Hours	1–12 + $\overline{AM}$/PM 00–23
03h	0	0	0	0	0	Day			Day	1–7
04h	0	0	10 Date		Date				Date	01–31
05h	Century	0	0	10 Month	Month				Month/ Century	01–12 + Century
06h	10 Year				Year				Year	00–99
07h	A1M1	10 Seconds			Seconds				Alarm 1 Seconds	00–59
08h	A1M2	10 Minutes			Minutes				Alarm 1 Minutes	00–59
09h	A1M3	12/$\overline{24}$	$\overline{AM}$/PM 20 Hour	10 Hour	Hour				Alarm 1 Hours	1–12 + $\overline{AM}$/PM 00–23
0Ah	A1M4	DY/$\overline{DT}$	10 Date		Day				Alarm 1 Day	1–7
					Date				Alarm 1 Date	1–31
0Bh	A2M2	10 Minutes			Minutes				Alarm 2 Minutes	00–59
0Ch	A2M3	12/$\overline{24}$	$\overline{AM}$/PM 20 Hour	10 Hour	Hour				Alarm 2 Hours	1–12 + $\overline{AM}$/PM 00–23
0Dh	A2M4	DY/$\overline{DT}$	10 Date		Day				Alarm 2 Day	1–7
					Date				Alarm 2 Date	1–31
0Eh	$\overline{EOSC}$	BBSQW	CONV	RS2	RS1	INTCN	A2IE	A1IE	Control	—
0Fh	OSF	0	0	0	EN32kHz	BSY	A2F	A1F	Control/Status	—
10h	SIGN	DATA	DATA	DATA	DATA	DATA	DATA	DATA	Aging Offset	—
11h	SIGN	DATA	DATA	DATA	DATA	DATA	DATA	DATA	MSB of Temp	—
12h	DATA	DATA	0	0	0	0	0	0	LSB of Temp	—

Figure 6: Timekeeping Registers

How to Write/Set Time in DS3231 using wire.h

S.No.	Steps	Code
1	Start I2C	Wire.beginTransmission(104)
2	Select start address of register to set value	Wire.write(0x00)
3	1: Write value of register 1	Wire.write(0)//seconds
4	N: Write value of register N	Wire.write(1)//minutes
5	Stop I2C	Wire.endTramission()

How to read time from DS3231 using Wire.h

S.No.	Steps	Code
1	Start I2C	Wire.beginTransmission(104)
2	Write start address where start reading	Wire.write(0)
3	End I2C	Wire.endTranmission()
4	Request number of bytes to read from I2C	Wire.requestFrom(104,3)//first 3 bytes
5	Start Reading byte 1	Second_bcd=wire_read() & 0x7f
6	Read till byte N	hour_bcd=wire_read() & 0x1F
7	Do masking for number of bits required	Done above
8	Convert BCD values into integer for 1	Second_i = (seconds_bcd)>>4 *10 + (seconds_bcd & 0x0F)
9	Convert BCD values into integer till N	hour_i = (hour_bcd)>>4 *10 + (hour_bcd & 0x0F)
10	Display Value after converting to string for 1	Serial.print(String(Second_i)
11	Display Value after converting to string for 1	Serial.println(String(hour_i)

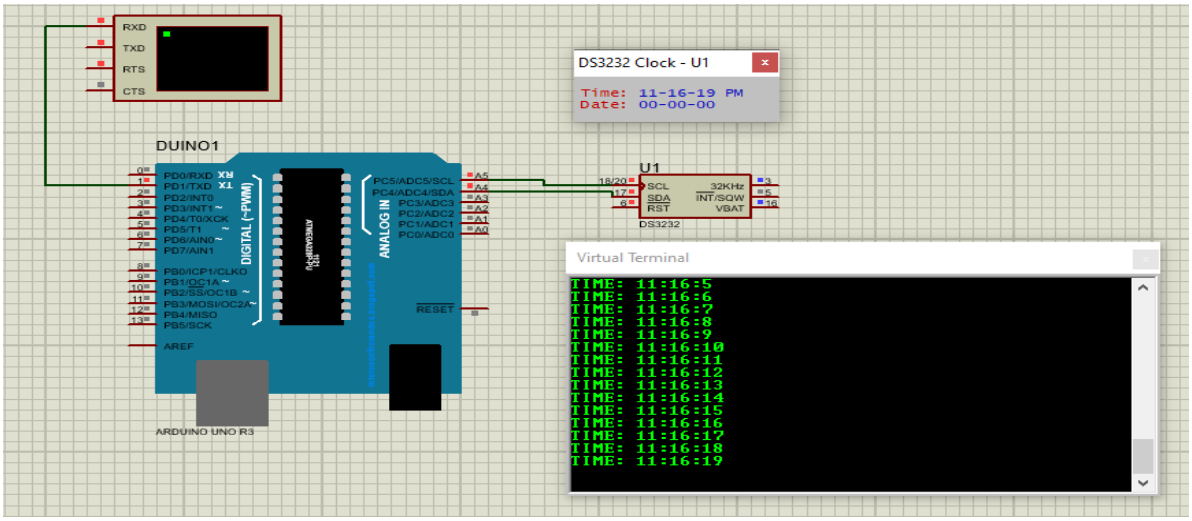

Figure 7: Circuit Diagram in Proteus

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.

Post-Lab Task 1:

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

Critical Analysis / Conclusion

(By Student about Learning from the Lab)

Lab Assessment				
Pre Lab			**/1**	**/10**
In Lab			**/5**	
Post Lab	**Data Analysis**	**/4**	**/4**	
	Data Presentation	**/4**		
	Writing Style	**/4**		
Instructor Signature and Comments				