Microcontroller 8051 : Interfacing ADC0808/ADC0809 with Microcontroller AT89S52

May 12, 2021

In order to read the physical data from real world and process under CPU in the digital form we need Analog to Digital converter. There are many source of physical data values like temperature, pressure, current, voltage, distance sensor which produce an Analog signal in 0-20ma or 0-5V or 0-10V. To read this type of Analog data from real world we need to convert into digital format. ADC module comes to help as we need to process this data under CPU in digital format.

This tutorial about interfacing **Analog-to-Digital Converter (ADC)** with Microcontroller AT89S52 we will use **ADC ADC0808/ADC0809** to read the analog value. Before starting programming Microcontroller for reading Analog value let know in brief about ADC ADC0808/ADC0809.

About ADC0808/ADC0809 :

ADC0808/ADC0809 is an 8-bit Successive Approximation ADC which is multiplexed among 8 input pins. The ADC0808, ADC0809 data acquisition component is a monolithic CMOS device with an 8-bit analog-to-digital converter, 8-channel multiplexer and microprocessor compatible control logic.

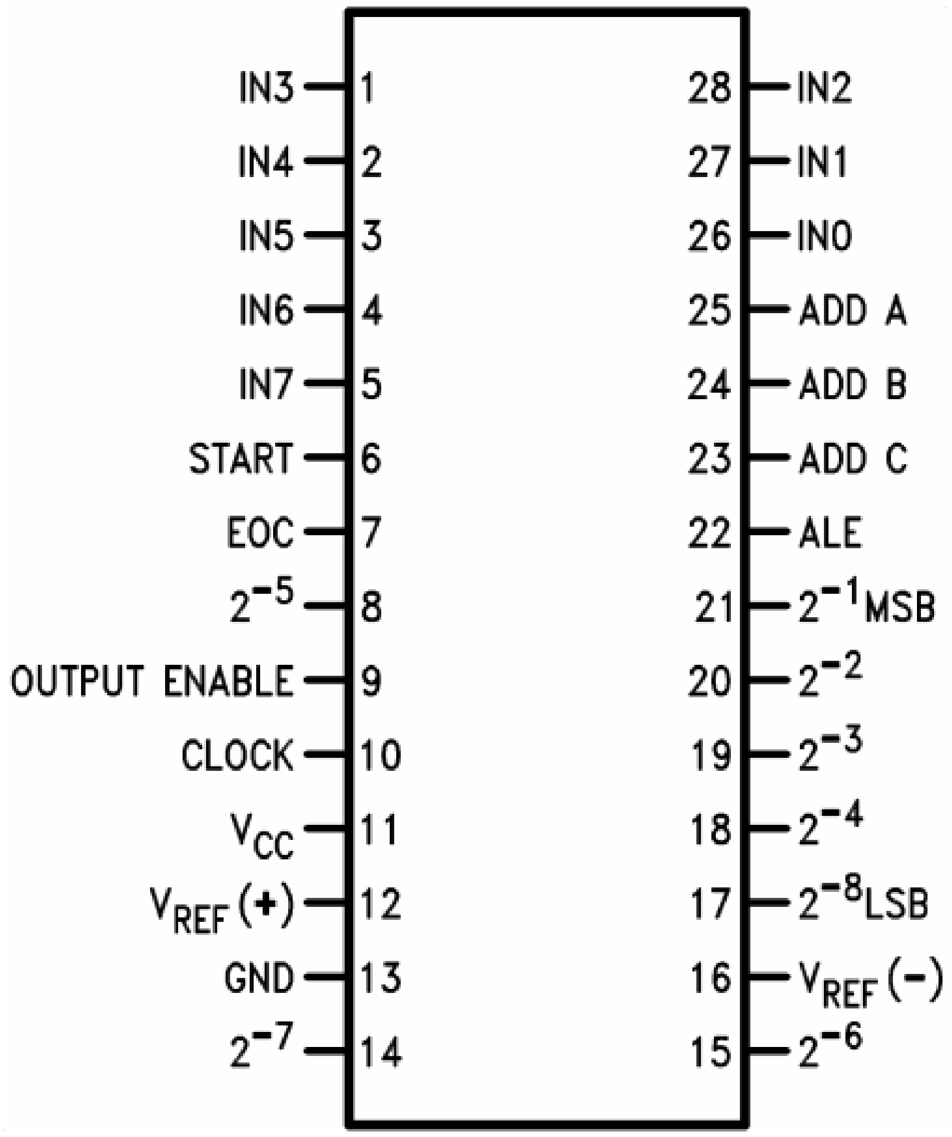
It can measure up to eight ADC values from 0 to 5 volt since it has eight channels. when voltage reference is +5V, its Step size will be 19.53mV. That is, for every increase of 19.53mV on the input side there will be an increase of 1 bit at the output side.
ADC0808 needs an external clock to operate. The ADC needs some specific control signals for its operations like start conversion and bring data to output pins. When the conversion is complete the EOC pins go low to indicate the end of a conversion and that the data is ready to be picked up.

Features : ADC0808/ADC0809 :

- Easy Interface to All Microprocessors
- Operates Ratiometrically or with 5 VDC or Analog Span Adjusted Voltage Reference
- No Zero or Full-Scale Adjust Required
- 8-Channel Multiplexer with Address Logic
- 0V to VCC Input Range
- Outputs meet TTL Voltage Level Specifications
- ADC0808 Equivalent to MM74C949
- ADC0809 Equivalent to MM74C949-1
- 28-pin molded chip carrier package

PIN Diagram :

Lets see the pins of the ADC0809



- **Addres Lines(A,B,C):** As mentioned earlier, ADC0809 has 8-muliplexed ADC channels. A particular input channel is selected by using these address bits as shown below.

- **Address Latch Enable (ALE):** A low-to-high signal at this pin will latch the above-selected address and selected the respective channel for ADC conversion.
- **START Conversion (SOC):** The A/D converter's successive approximation register (SAR) is reset on the positive edge of the start conversion (SC) pulse. Thus we need to generate a LOW-HIGH-LOW pulse for starting the ADC conversion.
- **End of Conversion (EOC):** Once the conversion is over, this pin is pulled HIGH by ADC0809. This pin needs to be monitored for the conversion to complete and then read the data.

- **Output Enable(OE):** ADC0809 does the adc conversion and holds the data in the internal registers. A HIGH signal on this pin will bring the data on the output lines.

ADC Configuration to Interface with Microcontroller:

- Start
- Select the channel using Address pins.
- A Low – High transition on ALE to latch in the address.
- A Low – High transition on Start to reset the ADC's SAR.
- A High – Low transition on ALE.
- A High – Low transition on start to start the conversion.
- Wait for End of cycle (EOC) pin to become high.
- Make Output Enable pin High.
- Take Data from the ADC's output
- Make Output Enable pin Low.
- Stop

Port configuration

You can connect the ADC0808/0809 to any of the PORT pins available on your microcontroller. We will used Port 0 of Our Microcontroller AT89S52.

ADC Voltage Calculation:

The ADC0808, ADC0809 is designed as a complete Data Acquisition System (DAS) for ratiometric conversion systems. In ratiometric systems, the physical variable being measured is expressed as a percentage of full-scale which is not necessarily related to an absolute standard. The voltage input to the ADC0808 is expressed by the equation

$V_{in}/(V_{fs}-V_z)= D_x/(D_{max}-D_{min})$

Where

V_{in} is input voltage for conversion
V_{fs} is full scale Voltage
V_z is zero voltage
D_x is data point being measure
D_{max} is Maximum data limit
D_{min} is Minimum data limit

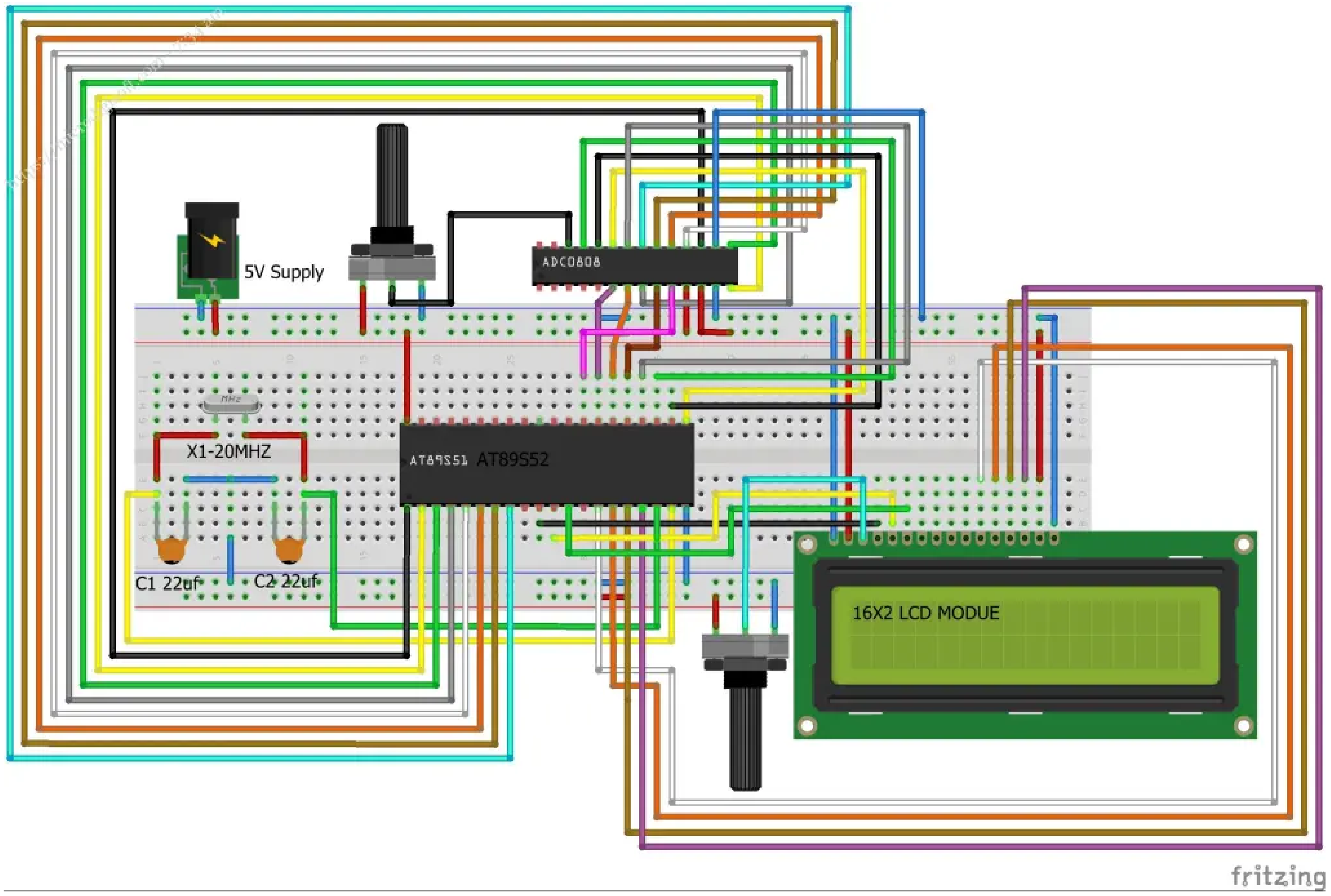
Download ADC0808/0808 Datasheet

Example: Reading ADC with Potentiometer

Let us take simple example of **Reading Potentiometer Value with ADC0808/0809** using Microcontroller and LCD. To build this project we need Potentiometer of 1kOhm Which we need to connect at ADC input IN0 and **16X2 LCD display** to monitor the ADC Value. In my previous tutorial we have seen **Interfacing LCD display with microcontroller AT89S5**2 and **Interfacing RTC with Microcontroller AT89S52.**

Also we have seen how to Read the Temperature data from LM35 Sensor using Arduino and Proteus.
Here we are going to use Microcontroller AT89S52 instead Arduino. Let us take variable potentiometer Input on ADC input IN0 from ADC0808/0809.

As I explain in previous tutorial to getting start with microcontroller programming we need install **Keil Microvision software** on our system. Also we need **Proteus** simulator to simulate the circuit with program.
So let us prepare circuit diagram to Interface 2-Digit Seven Segment display with Microcontroller AT89S52. let us do the connection as per connection requirement decided above:

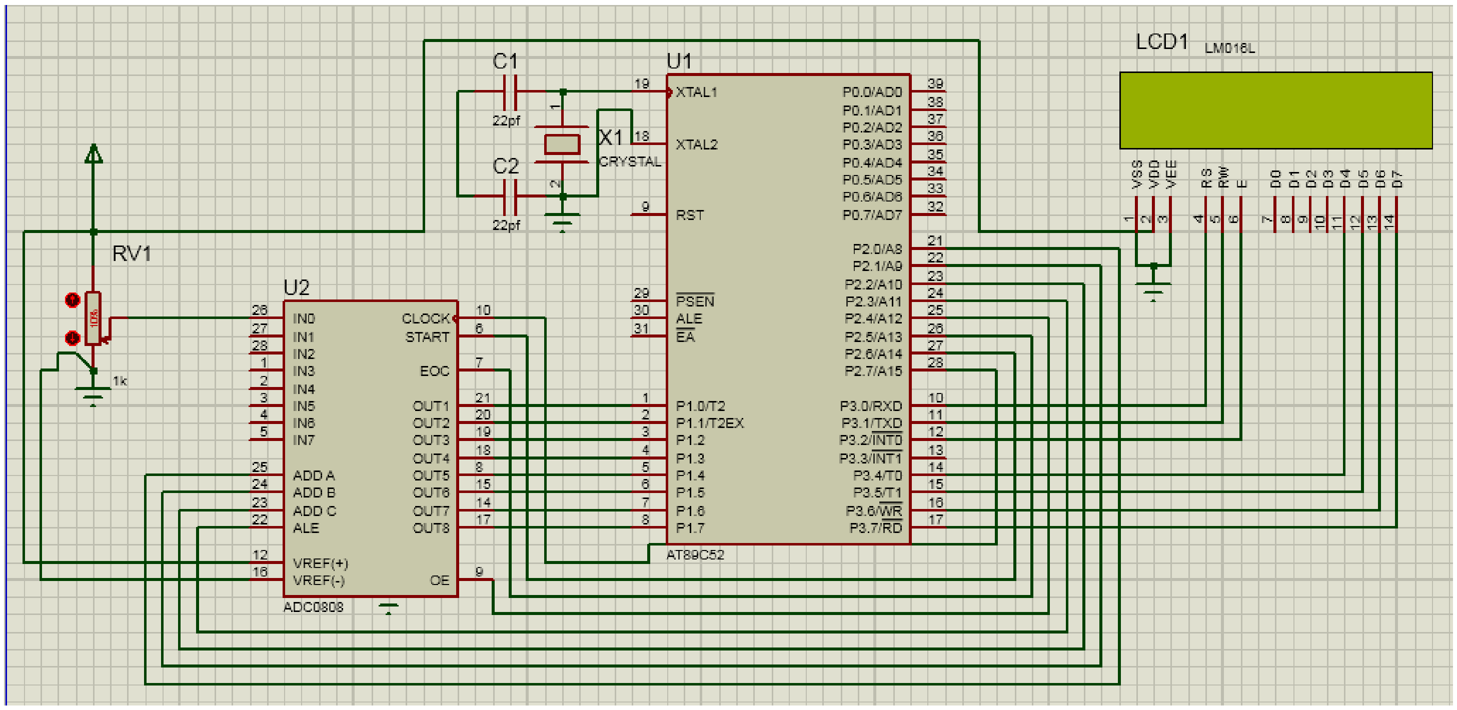


Program to Upload

```
#include < reg51.h>
#include "lcd.h"
#include "ds1307.h"
#include "i2c.h"
#include "delay.h"
#include"stdutils.h"
#include< stdio.h>
sbit ADDA     =P2^0;  //Address pins for selecting input channels.
sbit ADDB     =P2^1;
sbit ADDC     =P2^2;
sbit ALE   =P2^3;
sbit OE    =P2^4;
sbit EOC   =P2^5;
sbit START =P2^6;
sbit clk       =P2^7;
#define ADCPORT P1  //ADC Data port
 int ADCValue, adcresult[3],number;
 float volt;
void ADC_Init(void)
{
    ADDA=0;        //Initialize all the control lines to zero.
    ADDB=0;
    ADDC=0;
    ADCPORT=0xff;   //configure adc_databus as input
          EOC=1;
    ALE=0;
    OE=0;
    START=0;
}
ADC_Read()
{
   ADCValue=0;
   ALE=1;
   START=1;
   delay_us(1);
   ALE=0;
   START=0;
   while(EOC==1);
   while(EOC==0);
   OE=1;
   ADCValue=ADCPORT;
   delay_us(1);
   OE=0;;

  }

void timer0() interrupt 1
{
    clk=~clk;
}


/* main program */
void main()
{

    TMOD=0x02;
    TH0=0xFD;
    lcd_Init();
          IE=0x82;
    TR0=1;
    lcd_DisplayString(" MICRODIGISOFT ");
    lcd_Line_posnY();
    lcd_DisplayString("ADC0808 with LCD");
          delay_ms(1000);
          lcd_Clear();
    while(1)
    {
                 ADC_Init();        //Initialize the ADC module
     ADC_Read();       // Read the ADC value of channel zeroO
                 sprintf(adcresult," %d",ADCValue);
                 lcd_DisplayString(" ADC Reading:");
                 lcd_Line_posnY();
     lcd_DisplayString(adcresult);
                 ADCValue=0;
     delay_ms(5000);
                 lcd_Clear();
     }
}
```

Download Code

Circuit Diagram and Simulation with Proteus

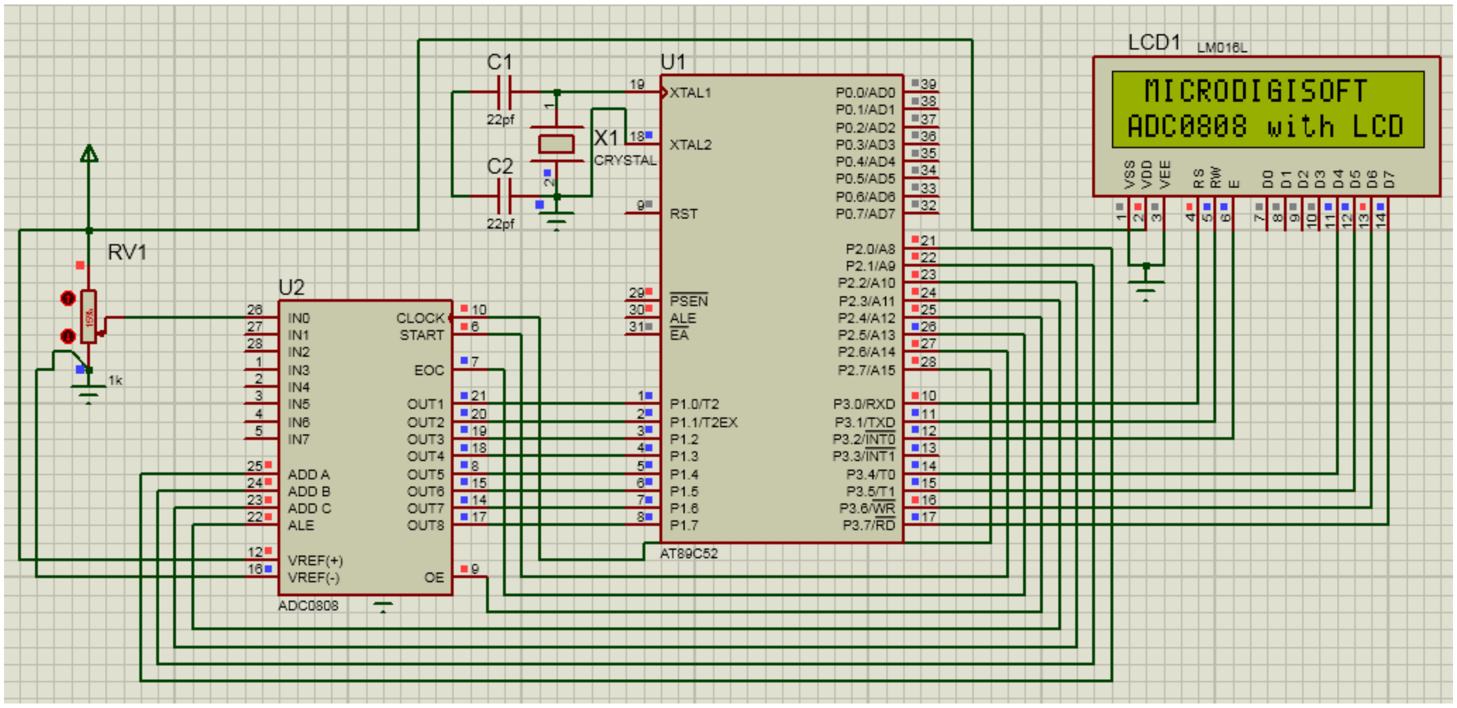


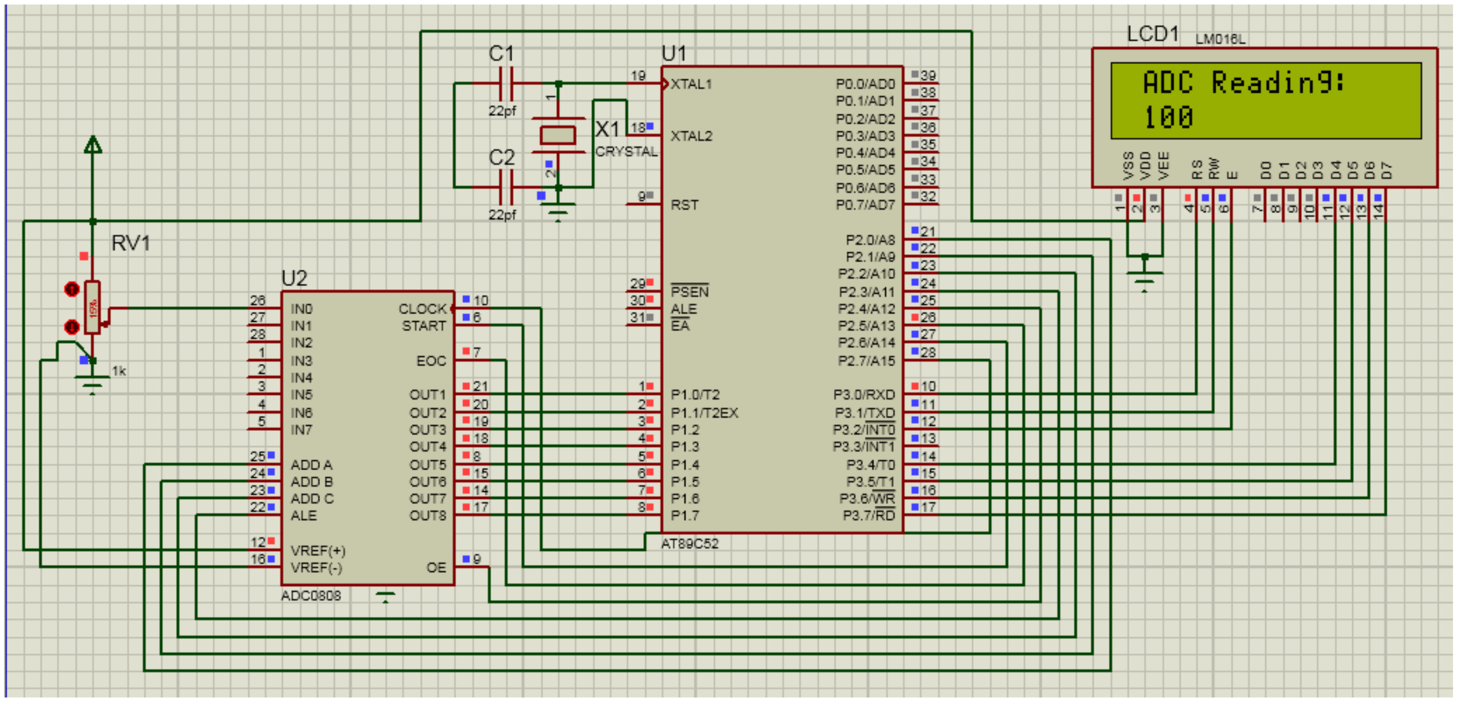
Now our project code is ready for simulating for Interfacing ADC0808/ADC0809 with Microcontroller AT89S52. Once your circuit ready build the Project code and if Build is successful, a HEX file would have been generated at the background of our IDE. This HEX file can be found inside the below directory

/…/ObjectsADC_LCD.hex

Please refer my previous tutorial for **How To Simulate Arduino Program In Proteus Software?** to understand stepwise process to build the circuit in Proteus software.
Here instead of using Arduino controller going to use microcontroller AT89S52 to simulate with HEX file.

Proteus Results:


LCD1
LM016L
MICRODIGISOFT
ADC0808 with LCD
U1
AT89C52
U2
ADC0808
RV1
C1
C2
22pf
X1
CRYSTAL


LCD1
LM016L
ADC Reading:
100
U1
AT89C52
U2
ADC0808
RV1
C1
C2
22pf
X1
CRYSTAL