In this turorial we will see how we can easily interface a 24C series serial EEPROM with AVR microcontrollers.

What is an EEPROM?

An EEPROM is kinds of novalatile memory, that means it is used for storing digital data permanently without any power suply. EEPROM stands for Electrically Erasable Programmable Read Only Memory. The advantage of these kind of ROMs is that they can be erased Electrically to make them ready for storing new data. Compare this with a CD R disks they can be recorded only once. A small amount of EEPROM is also available internally on the AVR chips. So if the volume of data you want to store is small (say few user names and password) then you can use it. The internal eeprom makes design small and simple.

But if the amount of data you want to store is large, say in order of few tens of kilobytes then you have to interface a External EEPROM chip with your AVR MCU. You can store pictures, sound and long texts in these eeproms.

Their are many kinds of EEPROM chip available from many manufactures. One very common family is 24C series serial EEPROMs. They are available upto 128KB in size. They uses I2C interface with host controller (MCU) which is a very popular serial communication standard. I will write more indept tutorial on I2C in comming days and in this tutorial I will give you easy to use function that you can use without any knowledge of I2C interface.

In this tutorial we will be using 24C64 EEPROM chip which has 8192 bytes = 8 KB = 8x1024 bytes of data storage.

The chip has storage location which have their unique address ranging from 0-8191. Consider these as storage cells so while storing and retriving data you have to tell the chip which cell location you want to read.

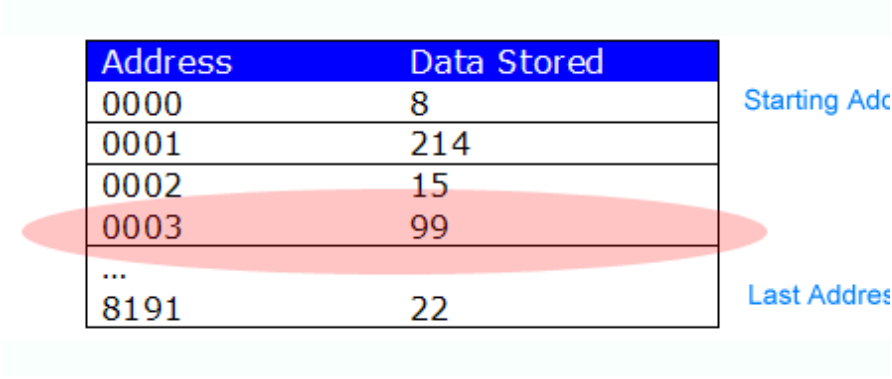

Fig. : Storage Cells inside the 24C64 EEPROM Chip

For exaple if you read location 0003 you will get 99 (see image above). Note each cell can store 8BITs of data so range you can store is 0-255 (-128 to +127). So if you want to store bigget data like int you have to store them in two cells.

Hardware Setup

Connect your ATmega32 with 24C64 chip as shown in the circuit diagram. You can use any avr development board for the purpose or assemble the whole circuit in a Breadboard or Veroboard.

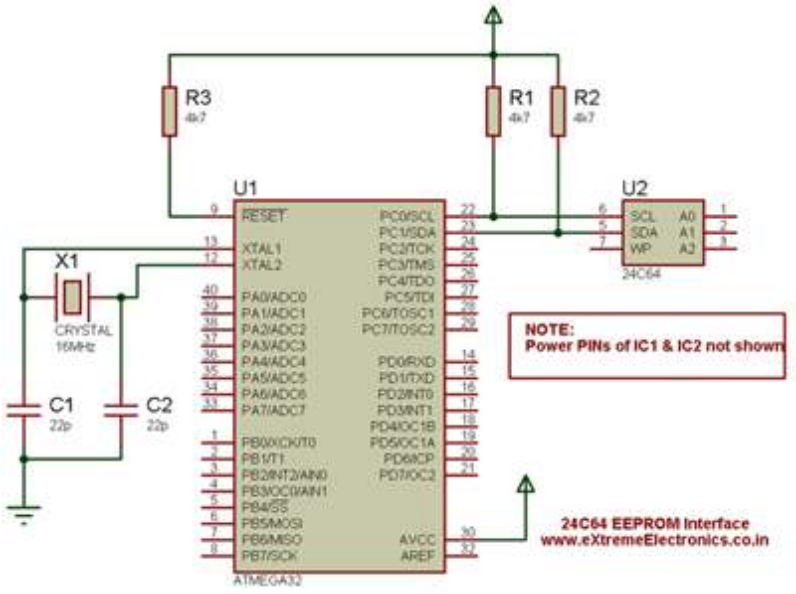

Fig. : Circuit Setup for 24C64.

Software Setup

Download and add the following files to your AVR Studio project. Now you can use the following functions for EEPROM interfacing.

- 24c64.h
- 24c64.c

Download Here.

EEOpen()

Arguments - None

Returns - Nothing

Description: This function should be called before any read/write operation. This functions initialize the communication channel.

EEWriteByte(unsigned int address,uint8_t data)

Arguments

- **address :** where you want to store data. The address must be within the range of EEPROM chip used.
- **data :** 8 bit data you want to store at the address given.

Returns

- 1 on success
- 0 on failure

Description: Use this function to store a 8bit value in any EEPROM storage cell.

EEReadByte(unsigned int address)

Arguments

- **address :** from where you want to read data. The address must be within the range of EEPROM chip used.

Returns

- The value stored in the specified EEPROM storage cell.

Description: Use this function to read a 8bit value from any EEPROM storage cell.

Sample Program.

The following sample program demonstrate the use of external EEPROM interfaing functions. The program makes use of the LCD library for AVRs to display information in a 16x2 LCD display. The program first writes 8Kbytes of data to a 24c64 eeprom to fill the whole eeprom with '7' and then it reads back to see if all the location has 7. If the condition is met the screen shows "Write Successfull" message.

I have used my xBoard - An Advance ATmega32 development board to test the routines. You can use any devboard with ATmega16 or ATmega32 MCUs. The 24C64 was mounted on a Breadboard.

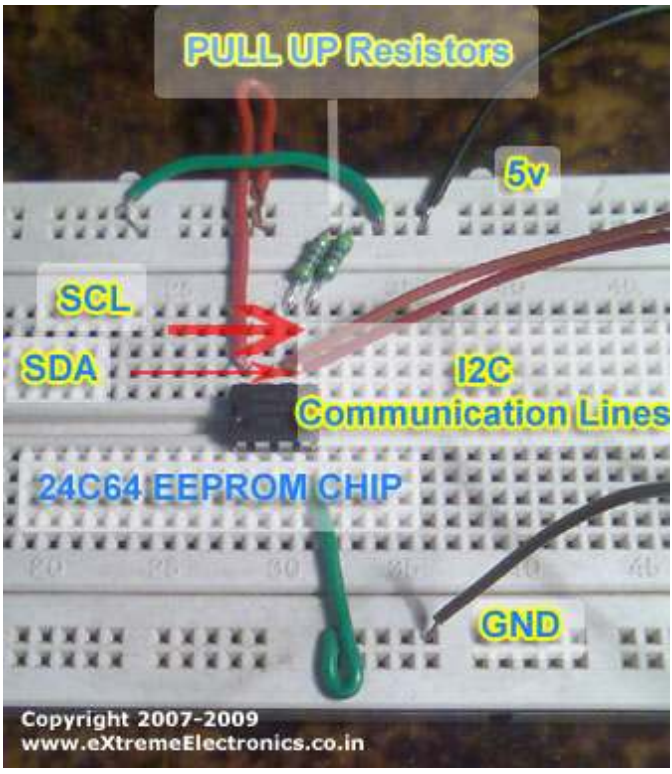

Fig. : 24C64 Serial EEProm Interface with ATmega32.

The 5v,GND,SDA,SCL were connected to the xBoard development board.

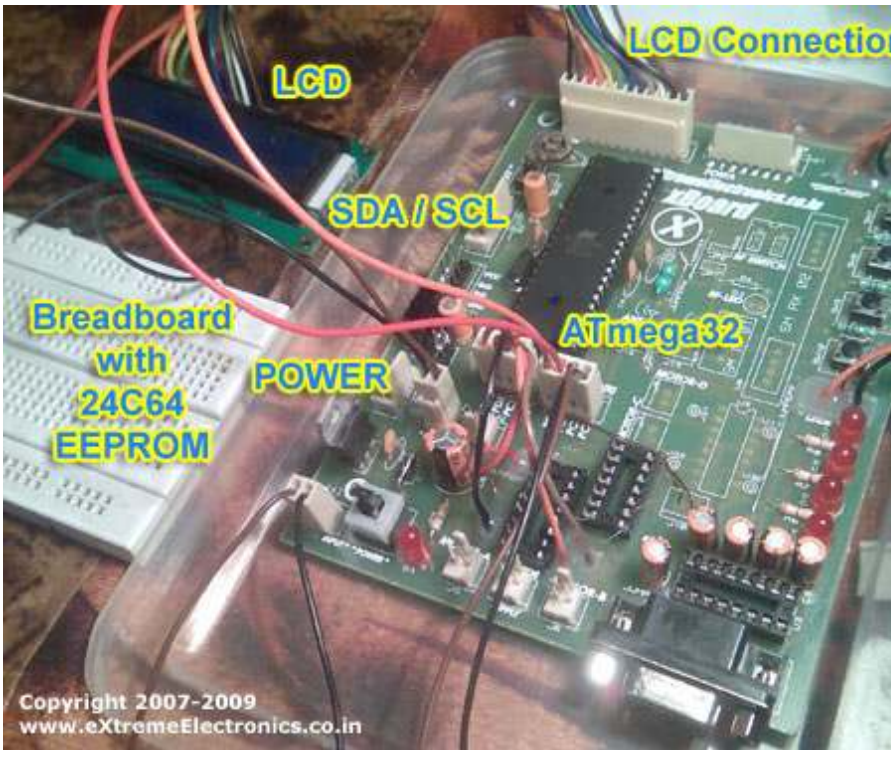

Fig. : xBoard with MCU ATmega32.

Fig. : Output of 24C64 test program.

```
/*

A sample program to test the Extrenal EEPROM interfacing

routines. The program fills the whole EEPROM with 7 and
then reads the whole EEPROM memory to see if all of them
contains 7.

This helps in quick testing of you hardware /software befo
using these routines in your own programs.

The target for this program is ATmega8, ATmega16, or ATmeg
running at 16MHz. If you use slower crystal the program wi
simply run slow but without any problems.

This program also makes use of eXtreme Electronics 16x2 LC
Interfacing routines. See the related page for more info

Author : Avinash Gupta
Date : 16 March 2009
Mail : me@avinashgupta.com
Web : www.eXtremeElectronics.co.in

NOTE: IF YOU USE THIS LIBRARY IN YOUR PROGRAMS AND FINDS
```

```
IT USEFUL PLEASE MENTION US IN CREDIT AND RECOMEND OTHERS.


*/

#include <avr/io.h>
#include "util/delay.h"

#include "24c64.h"

#include "lcd.h"


/*********************************

A simple delay routine to wait
between messages given to user
so that he/she gets time to read them

*********************************/

void Wait()
{
   uint8_t i;

   for(i=0;i<100;i++)
      _delay_loop_2(0);
}

void main()
{
   //Varriables
   uint8_t failed;
   unsigned int address;

   //Initialize LCD

   LCDInit(LS_BLINK);

   //Init EEPROM
   EEOpen();

   _delay_loop_2(0);

   LCDClear();
   LCDWriteString("External EEPROM");
```

```
LCDWriteStringXY(0,1,"Test");

Wait();

LCDClear();
LCDWriteString("Writting...");

//Fill whole eeprom 8KB (8192 bytes)

//with number 7
failed=0;
for(address=0;address<8192;address++)
{
   if(EEWriteByte(address,7)==0)
   {
      //Write Failed

      LCDClear();
      LCDWriteString("Write Failed !");
      LCDWriteStringXY(0,1,"Addess = ");
      LCDWriteInt(address,4);
      failed=1;
      Wait();
      break;
   }
}

LCDClear();

if(!failed)
   LCDWriteString("Written 8192bytes");

Wait();

LCDClear();
LCDWriteString("Verifying ...");

//Check if every location in EEPROM has

//number 7 stored
failed=0;
for(address=0;address<8192;address++)
{
   if(EEReadByte(address)!=7)
   {
      //Failed !
```

```
            LCDClear();
            LCDWriteString("Verify Failed");
            LCDWriteStringXY(0,1,"Addess = ");
            LCDWriteInt(address,4);
            failed=1;
            Wait();
            break;
        }
    }

    if(!failed)
    {
        //We have Done it !!!

        LCDClear();
        LCDWriteString("Write Success !");
    }

    while(1);

}
```

Downloads

All the required files for 24C eeprom interfacing with AVR MCUs.