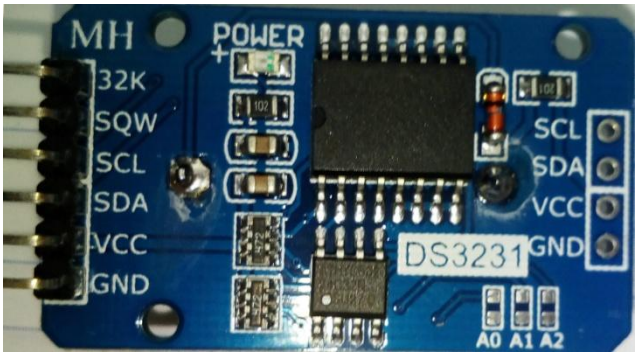
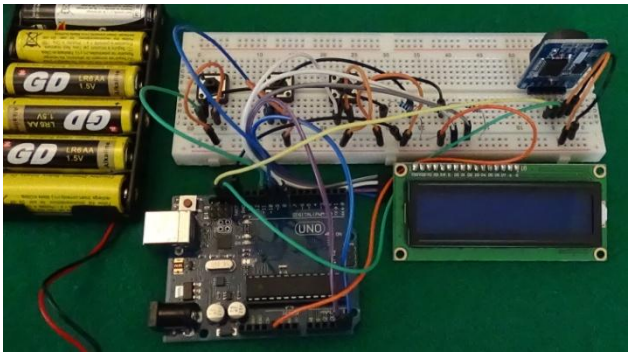
44 - RTC DS3231- orologio digitale regolabile- adjustable digital clock (see notes at end of this section)





Arduino e' uno smemorato. Se si toglie l'alimentazione mantiene memorizzato il programma, ma perde ogni altra informazione acquisita nel corso di ogni sua precedente attivita'. Tra le informazioni "volatili" c'e' anche la data e l'ora. Per superare quest'ultimo problema e' possibile utilizzare un modulo RTC (Real Time Clock) e cioe' un orologio che tiene aggiornata e memorizza data ed ora in sua vece. Il modulo RTC DS3231 rappresentato in figura e' dotato di una sua autonoma fonte di energia (una pila del tipo CR2032 da 3 volt) ed e' in grado di fornire informazioni su data, ora e temperatura ambientale.

Grazie alla pila di cui e' dotato, il modulo, una volta istruito con data ed ora corrente, continua a segnare il tempo, anche quando non e' in uso o e' addirittura tolto dal circuito.

E' anche in grado di gestire due allarmi e quindi di fornire ad Arduino un segnale digitale nel momento in cui matura il momento di un allarme. Questo segnale puo' essere utilizzato per attivare eventuali funzioni a tempo o, addirittura, per risvegliare Arduino dallo stato di "sleep" in cui puo' essere messo al fine di limitare il consumo di energia.

Il modulo fornisce le informazioni attraverso il bus SDA/SCL, il cui protocollo e' gestito dalle porte A4 ed A5 e, in replica, nelle porte a fianco della porta AREF di Arduino

In questo esercizio non ci limiteremo ad utilizzare il modulo RTC DS3231, ma utilizzeremo anche un display lcd gestito da un driver I2C e quindi avremo contemporaneamente in funzione due componenti che utilizzano il bus SDA/SCL. Per comodita' di assemblaggio il modulo RTC DS3231 e' stato collegato alle porte A4 ed A5 mentre il display lcd e' stato collegato alle due porte a fianco del porta AREF.

Il programma presentato in questo progetto e' in grado di "settare" il modulo RTC con data ed ora correnti. Per farlo si deve agire su tre pulsanti. Il primo di sinistra ha la funzione "set" e cioe' appena premuto predispone il sistema all'acquisizione dell'anno. Se viene premuto una seconda volta predispone il sistema alla acquisizione del mese e cosi' via, sino alla acquisizione dei secondi.

Il settaggio dei vari parametri (anno, mese, giorno, ora, minuto e secondo) avviene tramite i due successivi pulsanti. Il primo aumenta di una unita' il parametro in via di definizione mentre il secondo provoca la diminuzione di una unita'. Una volta settati tutti i parametri il sistema esce dalla modalita' di settaggio ed inizia a lavorare esponendo la data e l'ora.

Prima di procedere alla compilazione del programma e' necessario installare, se non gia' fatto, le seguenti librerie:

- timeLib.h reperibile qui;
- DS3232RTC.h reperibile qui'.

- LiquidCrystal_I2C.h reperibile qui'.

Per le modalita' di installazione si rimanda all'iter gia' illustrato nei precedenti esercizi e comunque riassumibile in:

- download delle librerie in forma compressa (normalmente ogni sito che tratta una libreria ha un punto dal quale si puo' attivare il download);
- installazione via IDE->sketch->include library->add .zip library
- verifica di avvenuta installazione (la libreria deve essere nell'elenco presente in IDE->sketch->include library->contributed library)

Il filmato di questo esercizio e' reperibile qui'.

Nota: Questo esercizio e questa nota sono parte di una serie che vede protagonisti Arduino ed alcuni dei componenti ad esso collegabili. Per la maggior parte degli esercizi e' anche disponibile un filmato su youtube.

- Esercizi facenti parte della raccolta
- Filmati presenti su youtube
- Informazioni su arduino e sui componenti collegabili (PDF scaricato nell'area di download)
- Breve manuale di programmazione (PDF scaricato nell'area di download)

Per eventuali chiarimenti o suggerimenti sul contenuto di questa scheda scrivere a giocarduino@libero.it

Here some notes about this project, translated by google translator

Arduino is a dazed. If power is off, it keeps the program, but loses any other information generated in the previous activity. Among the lost information there are also date and time. To overcome this problem we can use a RTC (Real Time Clock), an external watch that maintain date and time in his stead.

The DS3231 module shown in the figure is equipped with an autonomous power source (a battery CR2032 3v type) and can provide information about date, time and ambient temperature. Once trained with current date and time, continue to mark time, even when not in use or even removed from the circuit.

It also can handle two alarms, and then to provide, at Arduino, a digital signal when matures an alarm. This signal can be used to start a scheduled activity or, even, to awaken Arduino from the state of "sleep" in which can be put, in order to save energy.

The module provides information via "SDA/SCL" bus, whose protocol runs on A4 and A5 pins and on pins near (on the left) the AREF pin.

This project will not only uses a DS3231 device, but also uses an LCD display, both controlled by an I2C driver, that uses "SDA/SCL" bus.

In this project we can set the RTC module acting on three buttons. The first left button is the "set" function. The first pressing prepares system to year acquisition. When pressed a second time prepare acquisition of month and so on, until seconds acquisition.

Parameters setting (year, month, day, hour, minute and second) takes place via the following two buttons. The first increases by one unit the selected parameter, while the second causes a decreasing. Once setup all parameters, the system leaves the setting mode and begins work, exposing date and time.

Before proceeding to program compilation must be installed, if not already done, the libraries:

- ds3232rtc.h found here.
- timelib.h found here
- LiquidCrystal_I2C.h found here

For library installation, see process shown in previous examples, and summarized in:

- library download in compressed form;
- Installation via IDE-> sketch-> includes Library-> add .zip library
- After installation need verification: the library must be present in IDE-> sketch-> includes Library-> Contributed library

Note: This project and this note is part of a series that sees, as main characters, Arduino and some of connectable components. For most projects there is also a video on youtube.

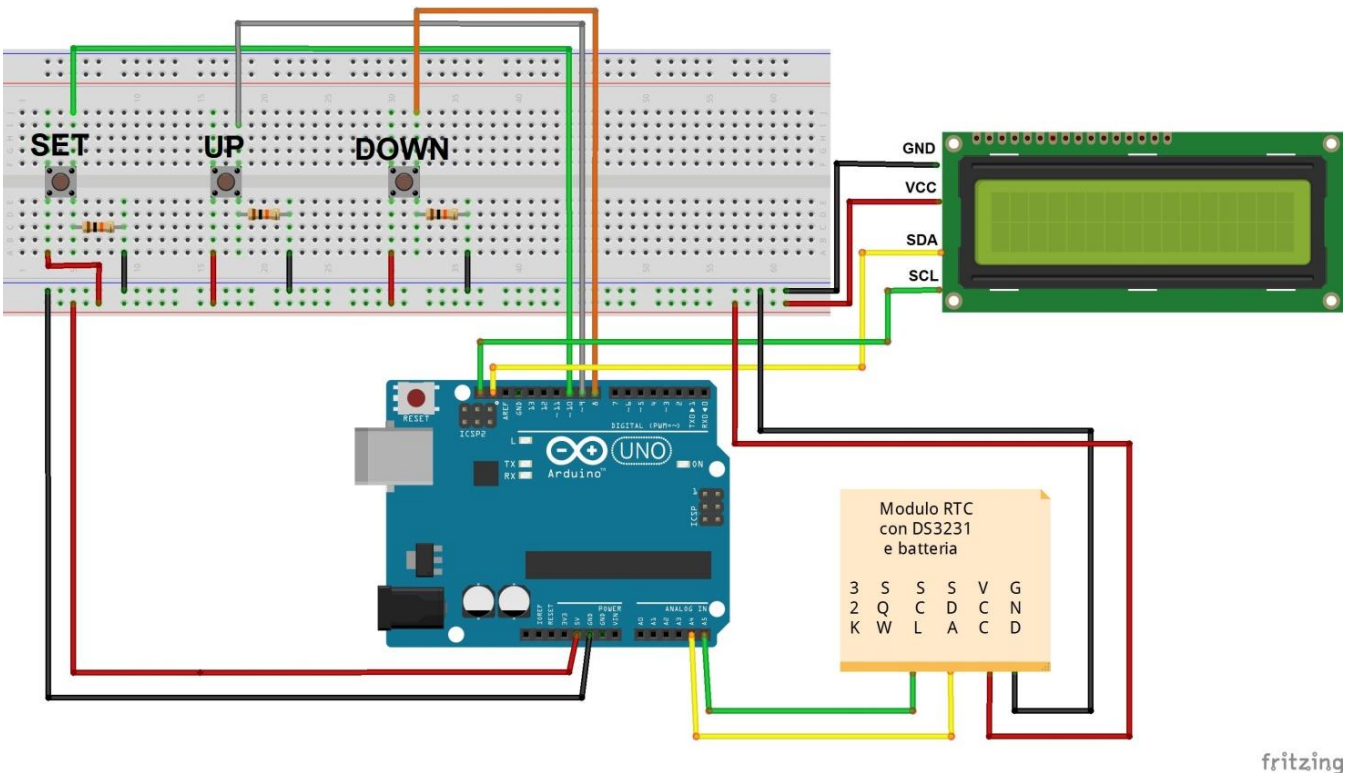
- Projects collection
- Movies on youtube
- About Arduino and components (italian; pdf will be downloaded in your download area
- Quick programming guide (almost english; pdf will be downloaded in your download area)

For any questions or suggestions about this note (and on its english translation), please write to giocarduino@libero.it (simple words and short sentences, please)

Materiali

- Un modulo RTC DS3231
- Un display lcd con driver I2C
- Tre pulsanti
- Tre resistenze da 10k ohm
- Un po' di cavetteria

Schema



Programma

```
/* Attenzione: facendo il copia/incolla dal PDF all'IDE si perde la formattazione del
 testo. Per rendere piu' facilmente leggibile il programma e' opportuno formattarlo
 subito dopo il trasferimento nell'IDE, premendo CTRL+T.
*
 Orologio digitale regolabile, gestito da DS3231
*
   Questo programma sincronizza il timer interno di arduino con la
   data e l'ora fornita da un orologio esterno (un modulo RTC con
   DS3231) ed espone data, ora e temperatura su di un display 1620
   governato da un circuito I2C
*
   La data e l'ora sono regolabili mediante tre pulsanti; per attivare
   la regolazione dei parametri bisogna premere il pulsante di settaggio
   (collegato alla porta 10);
   Ogni successiva pressione del pulsante di settaggio abilita la
   regolazione del successivo parametro (mese,giorno, ora, minuto e secondo).
*
   Nel momento in cui viene regolato l'ultimo parametro (secondi)
   l'orologio inizia a computare il tempo partendo dai
   dati immessi e non avra' piu' bisogno di essere regolato, anche se
   nel frattempoa rduino verra' spento (a meno che sul modulo ds3231
   non si scarichi la pila o la si tolga).
*
   ATTENZIONE:
   il programma esegue scarsi controlli di validita' dei dati immessi,
   per cui se si inseriscono dati errati (es. 30 febbraio) il risultato
   potrebbe essere imprevedibile.
  ----------------------------------------------------------------------
*
WARNING: cut&paste from PDF to IDE loses formatting. to restore it press CTRL + T.
*
DS3231 Real time clock
*
This program synchronizes the arduino's internal timer with the date and time provided by an
external clock (RTC DS3231) and displays date, time and temperature on a 1620 lcd display,
managed by an I2C circuit.

It's necessary, if not already done, upload the RTC appliance with current date and time, using
three button to set and adjuste.
*
After setting, clock begins work and does not needs to be set again (unless you drain or takes
out battery).
*
WARNING: Program does not perform any validity check on input data, so if you enter
incorrect data, results may be unpredictable.
*
*-----------------------------------------------------------------------------------------------
*/
#include <Wire.h>              //http://arduino.cc/en/Reference/Wire
#include <DS3232RTC.h>         //http://github.com/JChristensen/DS3232RTC
#include <TimeLib.h>           //http://playground.arduino.cc/Code/Time

#include <LiquidCrystal_I2C.h>  // libreria di gestione del display lcd
// . . . . . .  addr, en,rw,rs,d4,d5,d6,d7,bl,blpol
LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);  // definisce la
//  tipologia del display
time_t t;
tmElements_t tm;
//
//***** inizio elenco variabili ed elementi necessari al settaggio dell'orologio digitale  ********
// ****************** variables and elements needed to set phase **********************
#define settaggio 10
#define avanti 9
#define indietro 8
int indiceset      = 0;  // indice per avviare e gestire la routine di settaggio dei parametri
int setta          = 0;  // valore pulsante settaggio
int su             = 0;  // valore pulsante avanti
int giu            = 0;  // valore pulsante indietro
int incremento     = 0;  // utilizzato per aumentare o diminuire il parametro in via di settaggio
int indicedati     = 0;  // indice di scorrimento della tabella tabdati
int riga           = 0;  // riga del display sulla quale si sta agendo
int datodaesporre  = 0;  // parametro in via di esposizione
long tempoprec     = 0;  // zona di memorizzazione del momento dell'ultima esposizione dei dati
long tempocorrente = 0;  // zona di memorizzazione del momento corrente
int intermittenza  = 0;  // semaforo intermittenza (0 espone, 1 non espone il parametro)
```

Arduino: modulo RTC DS3231 orologio digitale regolabile – adjustable digital clock

```
int tabdati [19]   // tabella di acquisizione dei parametri
{
// min   max  valore
  10,     99,      0,   // anno year
  1,      12,      0,   // mese month
  1,      31,      0,   // giorno day
  0,      23,      0,   // ora hour
  0,      59,      0,   // minuto minute
  0,      59,      0,   // secondo second
};
// ******** fine delle variabili e degli elementi necessari al settaggio dell'orologio ***********
//
int digits         = 0;    // zona di memorizzazione dei minuti e dei secondi da esporre
//                             sul display lcd
char tabmesi [37]  =  "genfebmaraprmaggiulugagosetottnovdic";  // tabella descr. Mesi
// above: italian month description, on three digits
int mese           = 0;    // zona di memorizzazione del mese corrente
int lavoro         = 0;    // zona di lavoro, utilizzata per calcoli intermedi
//
//
//********* routine di esposizione di data, ora e temperatura sul display lcd
// ************** time and temperature display routine *************
//
void esponidati(void)
{
// ****** esposizione dei dati provenienti dal timer di arduino  *****
// *************** exposing data from arduino internal timer ***********
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("   ");
  lcd.print(day());
  lcd.print(' ');
  mese = month();    // mememorizza il mese
  mese = mese - 1;   // diminuisce di 1 il valore del mese poiche' la descrizione del primo
// mese (gen) inizia all'indirizzo zero della tabella mesi
  lcd.print (tabmesi [(mese * 3)]);      // espone la prima lettera del mese
  lcd.print (tabmesi [(mese * 3 + 1)]);  // espone la seconda lettera del mese
  lcd.print (tabmesi [(mese * 3 + 2)]);  // espone la terza lettera del mese
  lcd.print(' ');
  lcd.print(year());
  lcd.setCursor(4, 1);
  lcd.print(hour());
  digits = (minute());
  printDigits();  // routine esposizione zeri non significativi
  digits = (second());
  printDigits();  // routine esposizione zeri non significativi
}

//
//****routine visualizzazione minuti e secondi comprensivi di zeri non significativi********
//
void printDigits()
{
  lcd.print(':');
  if (digits < 10)
    lcd.print('0');
  lcd.print(digits);
}
//
// *************** inizio delle routine necesarie al settaggio dell'orologio ********
// ********************setting routine start from here ************************
//
//******** routine di sincronizzazione del timer di arduino con il timer del ds3231
//*************** syncronize arduino internal timer with ds3231 data
//
void sincronizzarduino ()
{
  lcd.clear ();
  lcd.setCursor(0, 0);       // posiziona il cursore all'inizio della prima riga
  setSyncProvider(RTC.get);  // sincronizza il timer di Arduino con i dati su RTC
  lcd. clear ();
  if (timeStatus() != timeSet)  // verifica se la sincronizzazione e' andata  a  buon fine
    lcd.print(" orologio non");
  else
    lcd.print("   orologio");
  lcd.setCursor (0, 1);
  lcd.print  (" sincronizzato");
  delay (1500);
```

```
}
//
//******** routine di settaggio dell'orologio *****************
// *********** parameters setting routine **************
//
void acquisiscidati ()
{
  lcd.clear ();
  // inserisce in tabdati i valori provenienti dal timer di arduino
  // al fine di usarli come valori iniziali per il settaggio
  // initial data from arduino interna timer
  tabdati [2] = year();
  tabdati [2] = tabdati [2] - 2000;
  tabdati [5] = month ();
  tabdati [8] = day ();
  tabdati [11] = hour ();
  tabdati [14] = minute ();
  tabdati [17] = second ();
  //
  // scorre tabdati e consente il settaggio dei vari parametri
//   runs on tabdati to permit parameters adjust
  for (indiceset = 0; indiceset < 6; indiceset = indiceset)
  {
    esponitabdati ();
    setta = 0;
    su    = 0;
    giu   = 0;
    incremento = 0;
    setta = digitalRead (settaggio);
    su    = digitalRead (avanti);
    giu   = digitalRead (indietro);
    if ((setta == 1) || (su == 1) || (giu == 1))
      delay (500);   // attende 5 decimi di secondo per evitare di interpretare una doppia pressione
    if (setta == 1)
      indiceset ++;
    if (su == 1)
      incremento = 1;
    if (giu == 1)
      incremento = -1;
    //
    // modifica il parametro corrente di tabadati con il valore presente in  "incremento" e verifica
    // che il nuovo valore in tabdati non ecceda i limiti  - Edit current tabdati  parameter with
    // the "incrementa" value and verify that the new value does not exceed limits
    tabdati [(indiceset * 3) + 2] = tabdati [(indiceset * 3) + 2] + incremento;
    if (tabdati [(indiceset * 3) + 2] > tabdati [(indiceset * 3) + 1])
      tabdati [(indiceset * 3) + 2] = tabdati [(indiceset * 3) + 1];
    if ( tabdati [(indiceset * 3) + 2] < tabdati [(indiceset * 3) ] )
      tabdati [(indiceset * 3) + 2] = tabdati [(indiceset * 3)];
  }
  //  inserisce in ds3231 i nuovi dati - update ds3231 data
  tm.Year = y2kYearToTm(tabdati [2]);
  tm.Month  = tabdati [5];
  tm.Day    = tabdati [8];
  tm.Hour   = tabdati [11];
  tm.Minute = tabdati [14];
  tm.Second = tabdati [17];
  t         = makeTime(tm);
  RTC.set(t);      // aggiorna (setta) il modulo RTC
  setTime(t);      // aggiorna (setta) il modulo RTC
  indiceset = 0;  //  azzera l'indice di tabdati e riprende le normali funzioni dell'orologio
}
//
//****** routine di esposizione dei dati in via di settaggio ****
// ********** exposes setting parameter ***********
//
void esponitabdati ()
{
  tempocorrente = millis ();
  if ((tempocorrente - tempoprec) > 500) // espone solo se sono passati piu' di 5 decimi di secondo
  {
    tempoprec = tempocorrente;
    riga = 0;
    lcd.setCursor (6, 0);
    lcd.print ("/");
    lcd.setCursor (9, 0);
    lcd.print ("/");
    lcd.setCursor (6, 1);
    lcd.print (":");
```

```
    lcd.setCursor (9, 1);
    lcd.print (":");
    for (indicedati = 0; indicedati < 6; indicedati++)
    {
    // **** espone i parametri presenti in tabdati *****
      if (indicedati > 2)
        riga = 1;
      datodaesporre = tabdati [(indicedati * 3) + 2];
      lcd.setCursor (((indicedati - riga * 3) * 3) + 4, riga);
      if (datodaesporre < 10)
        lcd.print ("0");
      lcd.print (datodaesporre);
    }
    riga = 0;
    if (indiceset > 2)
      riga = 1;
    if (intermittenza == 1)  // gestisce il blinking del parametro corrente
    {
      intermittenza = 0;
      lcd.setCursor (((indiceset - riga * 3) * 3) + 4, riga);
      lcd.print ("00");
    }
    else
    {
      intermittenza = 1;
      datodaesporre = tabdati [(indiceset * 3) + 2];
      lcd.setCursor (((indiceset - riga * 3) * 3) + 4, riga);
      if (datodaesporre < 10)
        lcd.print ("0");
      lcd.print (datodaesporre);
    }
  }
}
//
// *************** fine delle routine necessarie al settaggio dell'orologio ******
// ************************** end setting routines ************************
//
void setup(void)
{
  Serial.begin (9600);       // inizializza il monitor seriale
  // ********** le seguenti tre istruzioni sono funzionali al settaggio dell'orologio
  pinMode (settaggio, INPUT);
  pinMode (avanti, INPUT);
  pinMode (indietro, INPUT);
  lcd.begin(16, 2);          // inizializza il display (16 caratteri per due righe) e accende
  lcd.backlight();           // lo sfondo
  lcd.print ("   buongiorno");
  delay (1500);
  sincronizzarduino ();  // sincronizza il timer di arduino con i dati forniti da ds3231
}
//
//
void loop(void)
{
  // ********** inizio delle istruzioni necessarie al settaggio dell'orologio ********
  indiceset = 0;
  indiceset = digitalRead (settaggio);
  if (indiceset == 1)    // se e' stato premuto il pulsante di settaggio
  { delay (1000);  // attende un secondo per evitare l'interpretazione di una doppia pressione
    acquisiscidati ();   // lancia la routine di settaggio
  }
  // ********* fine delle istruzioni di settaggio ***********************************
  //
  tempocorrente = millis();
  if ((tempocorrente - tempoprec ) > 1000)// se e' passato piu' di un secondo dall'ultima
  // visualizzazione
  {
    tempoprec = tempocorrente;
    esponidati ();
  }
}
```