

# 31.Time\_Clock

## Introduction

In this lesson, you will learn how to use the DS1302 real-time clock module to get the current date and time.

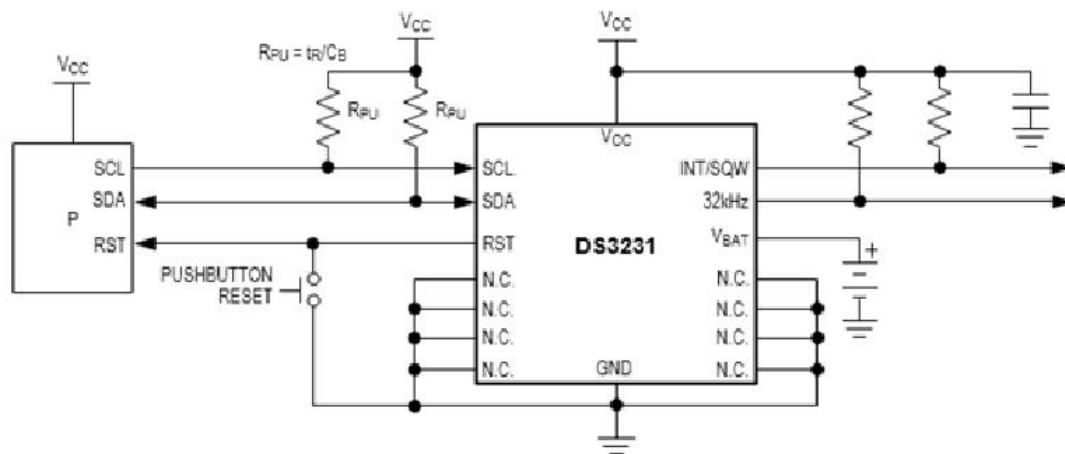
## Hardware Required

- ✓ 1 \* Raspberry Pi
- ✓ 1 \* T-Extension Board
- ✓ 1 \* 40-pin Cable
- ✓ 1 \* DS3231 RTC module
- ✓ Several Jumper Wires
- ✓ 1 \* Breadboard

## Principle

### DS3231

The DS3231 is a simple time-keeping chip. It has an integrated battery, so the clock can continue keeping time even when unplugged.



The DS3231 is a low-cost, highly accurate Real Time Clock which can maintain hours, minutes and seconds, as well as, day, month and year information. Also, it has automatic compensation for leap-years and for months with fewer than 31 days.

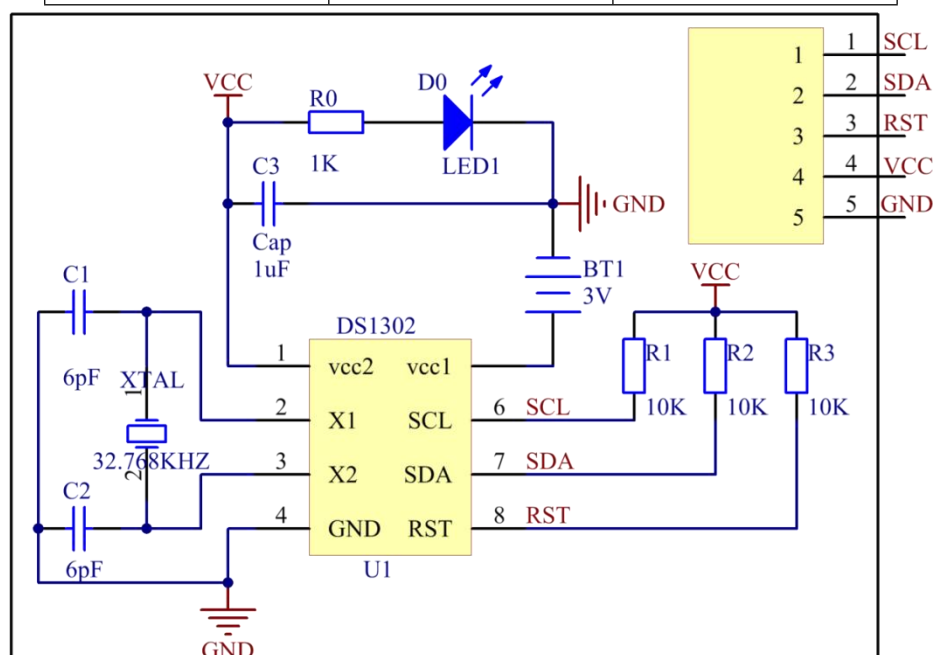
- Counts Hours, Minutes and Seconds
- Day of the Week, Day, Month and Year
- Automatic compensation for leap-years and for months with fewer than 31 days
- Operating voltage from 3.3 to 5V
- 3V Battery
- I2C Communication Protocol

The module can work on either 3.3 or 5 V which makes it suitable for many development platforms or microcontrollers. The battery input is 3V and a typical CR2032 3V battery can power the module and maintain the information for more than a year.

The module uses the I2C Communication Protocol which makes the connection to the Arduino Board very easy.

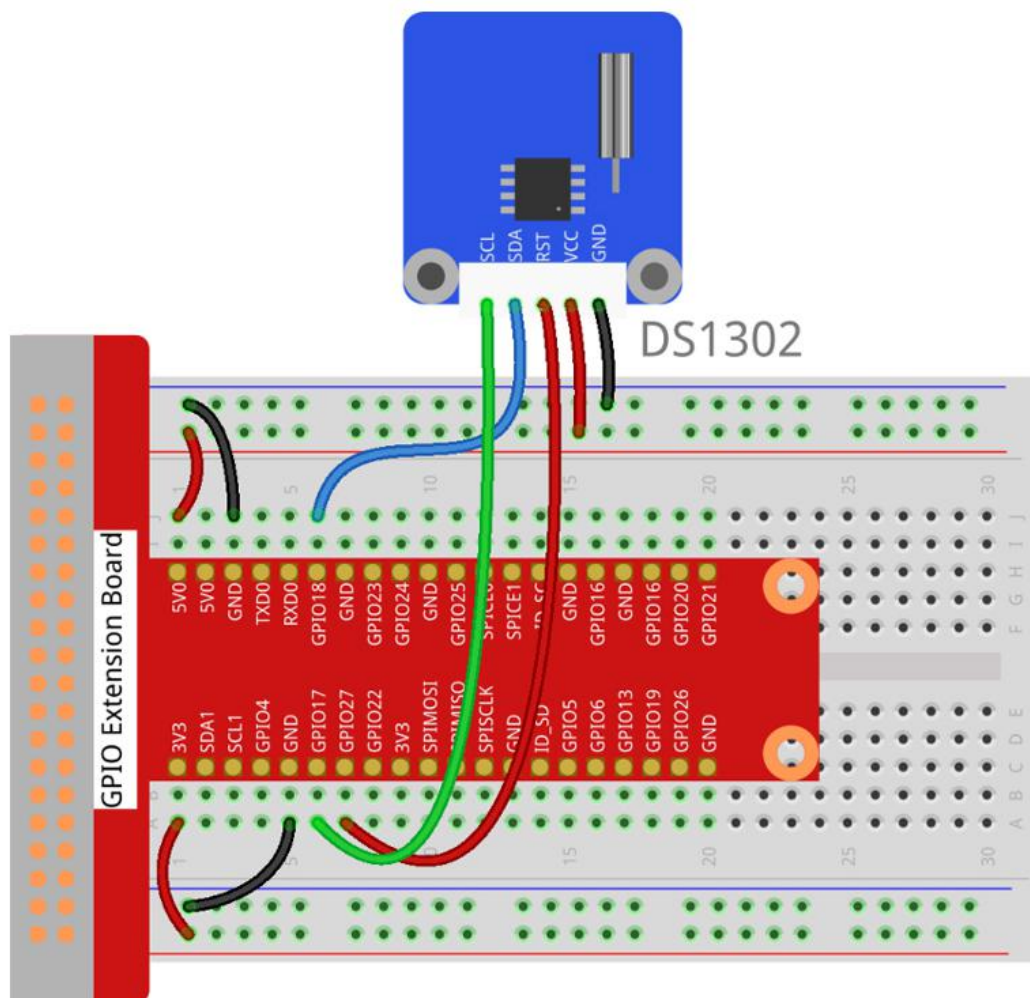
## Schematic Diagram

Raspberry Pi	T-Board Name	DS1302 RTC
GPI00	GPI17	SCL
GPI01	GPI18	SDA
GPI02	GPI27	RST
5V	5V0	VCC
GND	GND	GND



## Experimental Procedures

### Step 1: Build the circuit.



### For C Language Users

#### Step 2: Change directory.

```
cd /home/pi/REXQualis_Raspberry_Pi_Complete_Starter_Kit/C/31.Time_Clock
```

#### Step 3: Compile.

```
gcc 31.Time_Clock.c -o Time_Clock.out -lwiringPi -lwiringPiDev
```

#### Step 4: Set up time by:

```
sudo ./Time_Clock.out -sdsc
```

Set the year, month, and day to: YYMMDD

Set the hours, minutes, and seconds to: HHMMSS

Day of the week (Sunday is 0)

## Step 5: Run the executable file above.

```
sudo ./Time_Clock.out
```

## Code

```
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include <string.h>
#include <time.h>
#include <wiringPi.h>
#include <ds1302.h>

// Register defines

#define RTC_SECS  0
#define RTC_MINS  1
#define RTC_HOURS 2
#define RTC_DATE  3
#define RTC_MONTH 4
#define RTC_DAY    5
#define RTC_YEAR  6
#define RTC_WP     7
#define RTC_TC     8
#define RTC_BM     31

static unsigned int masks [] = { 0x7F, 0x7F, 0x3F, 0x3F, 0x1F, 0x07, 0xFF }

// bcdToD: dToBCD:

static int bcdToD (unsigned int byte, unsigned int mask)
{
    unsigned int b1, b2 ;

    byte &= mask ;

    b1 = byte & 0x0F ;
```

```

    b2 = ((byte >> 4) & 0x0F) * 10 ;

    return b1 + b2 ;
}

static unsigned int dToBcd (unsigned int byte)
{
    return ((byte / 10) << 4) + (byte % 10) ;
}

// ramTest:

static int ramTestValues [] =
    { 0x00, 0xFF, 0xAA, 0x55, 0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80, 0x00, 0xF0,
0x0F, -1 } ;

static int ramTest (void)
{
    int addr ;
    int got ;
    int i = 0 ;
    int errors = 0 ;
    int testVal ;

    printf ("DS1302 RAM TEST\n") ;

    testVal = ramTestValues [i] ;

    while (testVal != -1)
    {
        for (addr = 0 ; addr < 31 ; ++addr)
            ds1302ramWrite (addr, testVal) ;

        for (addr = 0 ; addr < 31 ; ++addr)
            if ((got = ds1302ramRead (addr)) != testVal)
            {
                printf ("DS1302 RAM Failure: Address: %2d, Expected: 0x%02X, Got: 0x%02X\n",
                    addr, testVal, got) ;

                ++errors ;
            }
    }
}

```

```

    }

    testVal = ramTestValues [++i] ;
}

for (addr = 0 ; addr < 31 ; ++addr)
    ds1302ramWrite (addr, addr) ;

for (addr = 0 ; addr < 31 ; ++addr)
    if ((got = ds1302ramRead (addr)) != addr)
    {
        printf ("DS1302 RAM Failure: Address: %2d, Expected: 0x%02X, Got: 0x%02X\n",
            addr, addr, got) ;

        ++errors ;
    }

if (errors == 0)
    printf ("-- DS1302 RAM TEST: OK\n") ;

else
    printf ("-- DS1302 RAM TEST FAILURE. %d errors.\n", errors) ;

return 0 ;
}

// setLinuxClock:

static int setLinuxClock (void)
{
    char dateTime [20] ;
    char command [64] ;
    int clock [8] ;

    printf ("Setting the Linux Clock from the DS1302... ") ; fflush (stdout) ;
    ds1302clockRead (clock) ;

    // [MMDDhhmm[[CC]YY][.ss]]

    sprintf (dateTime, "%02d%02d%02d%02d%02d%02d.%02d",
        bcdToD (clock [RTC_MONTH], masks [RTC_MONTH]),
        bcdToD (clock [RTC_DATE], masks [RTC_DATE]),

```

```

        bcdToD (clock [RTC_HOURS], masks [RTC_HOURS]),
        bcdToD (clock [RTC_MINS], masks [RTC_MINS]),
        20,
        bcdToD (clock [RTC_YEAR], masks [RTC_YEAR]),
        bcdToD (clock [RTC_SECS], masks [RTC_SECS])) ;
    sprintf (command, "/bin/date %s", dateTime) ;
    system (command) ;
    return 0 ;
}

// setDSclock:
static int setDSclock (void)
{
    struct tm t ;
    time_t now ;
    int clock [8] ;
    int time = 0 ;
    int date = 0 ;
    int weekday = 0 ;

    printf ("Setting the clock in the DS1302 from type in... ") ;
    printf ("\n\nEnter Date(YMMDD): ") ;
    scanf ("%d", &date) ;
    printf ("Enter time(HHMMSS, 24-hour clock): ") ;
    scanf ("%d", &time) ;
    printf ("Enter Weekday(0 as sunday): ") ;
    scanf ("%d", &weekday) ;

    // printf("\ndate: %d, time: %d\n\n", date, time) ;

    clock [ 0] = dToBcd (time % 100) ;// seconds
    clock [ 1] = dToBcd (time / 100 % 100) ; // mins
    clock [ 2] = dToBcd (time / 100 / 100) ;// hours
    clock [ 3] = dToBcd (date % 100) ;// date

```

```

clock [ 4] = dToBcd (date / 100 % 100) ;    // months 0-11 --> 1-12
clock [ 5] = dToBcd (weekday) ;    // weekdays (sun 0)
clock [ 6] = dToBcd (date / 100 / 100) ;    // years
clock [ 7] = 0 ;    // W-Protect off
ds1302clockWrite (clock) ;
printf ("OK\n") ;
return 0 ;
}

int main (int argc, char *argv [])
{
    int i ;
    int clock [8] ;
    int year ;
    int month ;
    int date ;
    int hour ;
    int minute ;
    int second ;
    int weekday ;
    wiringPiSetup () ;
    ds1302setup    (0, 1, 2) ;
    if (argc == 2)
    {
        /**/ if (strcmp (argv [1], "-slc") == 0)
            return setLinuxClock () ;
        else if (strcmp (argv [1], "-sdsc") == 0)
            return setDSclock () ;
        else if (strcmp (argv [1], "-rtest") == 0)
            return ramTest () ;
        else

```



```

{
    printf ("Usage: ds1302 [-slc | -sdsc | -rtest]\n");
    return EXIT_FAILURE ;
}
}
for (i = 0 ;; ++i)
{
    printf ("%5d:  ", i) ;
    ds1302clockRead (clock) ;

    hour   = bcdToD (clock [2], masks [2]) ;
    minute = bcdToD (clock [1], masks [1]) ;
    second = bcdToD (clock [0], masks [0]) ;
    date   = bcdToD (clock [3], masks [3]) ;
    month  = bcdToD (clock [4], masks [4]) ;
    year   = bcdToD (clock [6], masks [6]) + 2000 ;
    weekday = bcdToD (clock [5], masks [5]) ;
    printf ("  %04d-%02d-%02d", year, month, date) ;
    printf ("  %02d:%02d:%02d", hour, minute, second) ;
    switch (weekday){
        case 0: printf ("  SUN") ; break;
        case 1: printf ("  MON") ; break;
        case 2: printf ("  TUE") ; break;
        case 3: printf ("  WED") ; break;
        case 4: printf ("  THU") ; break;
        case 5: printf ("  FRI") ; break;
        case 6: printf ("  SAT") ; break;
    }
    printf ("\n") ;
    delay (200) ;
}

```

```
    return 0 ;  
}
```

## For Python Language Users

### Step 2: Change directory.

```
cd /home/pi/REXQualis_Raspberry_Pi_Complete_Starter_Kit/Python/31.Time_Clock
```

### Step 3: Run.

```
sudo python3 31.Time_Clock.py
```

### Code

```
#!/usr/bin/env python3  
  
#   This is a program for DS1302 RTC Module.  
#   It provide precision timmer.  
#  
#   This program depend on rpi_time.py.  
#   ds1302 Module           Pi  
#       VCC ----- 5 V (Must be 5v)  
#       GND ----- GND  
#       SCL ----- GPIO23  
#       I/O ----- GPIO24  
#       RST ----- GPIO25  
#  
#-----  
  
from datetime import datetime  
from ds1302 import DS1302  
from sys import version_info  
import time  
  
if version_info.major == 2:  
    input = raw_input
```

```

rtc = DS1302()

def setup():
    print("")
    print("")
    print(rtc.get_datetime())
    print("")
    print("")
    a = input("Do you want to setup date and time?(y/n) ")
    if a == 'y' or a == 'Y':
        date = input("Input date:(YYYY MM DD) ")
        time = input("Input time:(HH MM SS) ")
        date = list(map(lambda x: int(x), date.split()))
        time = list(map(lambda x: int(x), time.split()))
        print("")
        print("")
        rtc.set_datetime(datetime(date[0], date[1], date[2], time[0], time[1], time[2]))
        dt = rtc.get_datetime()
        print("You set the date and time to:", dt)

def loop():
    while True:
        a = rtc.get_datetime()
        print(a)
        time.sleep(0.5)

def destory():
    pass # Release resource

if __name__ == '__main__': # Program start from here
    setup()
    try:
        loop()
    except KeyboardInterrupt: # When 'Ctrl+C' is pressed, the child program

```

**destroy() will be executed.//**

destory()

## Phenomenon Picture

