



Matériels - Horloge Temps Réel (HTR) I²C

[Mise à jour le 3/5/2024]

Lecture connexe

- [La gestion du temps : Timer, HTR](#)

1. Généralités

Une horloge temps réel (abrégé en HTR, en anglais *real-time clock* ou *RTC*), est une horloge permettant un décompte très précis du temps (par exemple en nanosecondes) pour un système électronique, en vue de dater ou déclencher des événements selon l'heure. Par rapport à l'horloge à quartz, l'horloge en temps réel réduit le risque de panne mécanique. Il peut donc être utilisé dans un environnement très vibratoire. Wikipédia

2. DS1307



2.1 Généralités

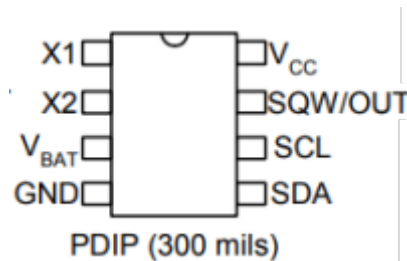
- **Description**

L'horloge temps réel (RTC) à circuit DS1307 est une **horloge / calendrier BCD** (décimal codé binaire) disposant de **56 octets de NV SRAM**. Elle communique avec un microcontrôleur via un bus I²C.

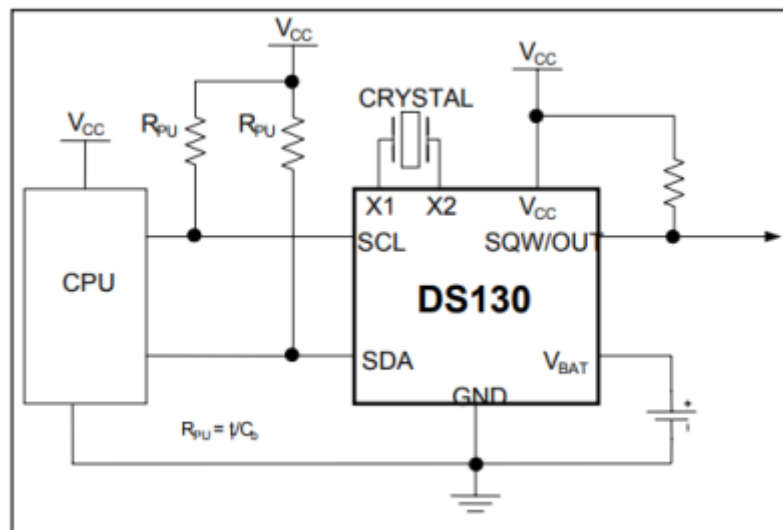
Ce circuit fournit l'année, le mois, le jour, les heures, les minutes et les secondes en tenant compte

des années bissextiles. L'horloge fonctionne au format 24 heures ou 12 heures avec l'indication AM / PM. Un circuit de détection de puissance intégré passe automatiquement sur une alimentation de secours en cas de coupure du courant.

- **Brochage**



- **Schéma**



- **Documentation**

- **Datasheet** du circuit DS1307 à télécharger [ici](#).

2.2 Module MR005 de Microrobot



2.2.1 Présentation

- **Sources**

- Site : microbot.it
- Wiki : [Utiliser un module horloge temps réel DS1307 avec une carte Arduino / Genuino](#)

Module horloge temps réel Microbot basé sur un DS1307 permettant de donner la date et l'heure via le bus I²C.

- **Distributeur** : [Gotronic](#)

- **Caractéristiques**

- Alimentation 4,5 à 5,5 Vcc
- Consommation: 1,5 mA
- Sauvegarde: 1 pile type CR2032
- Informations:
 - secondes, minutes et heures
 - jour, mois, année
- Protocole I²C (**SLA = 0x68**)
- Sortie drain ouvert : signal logique de fréquence réglable (f=1Hz, 4kHz, 8kHz, 32kHz)
- Dimensions: 32 x 23 x 11 mm
- Poids: 5g

2.2.2 Bibliothèques

- [Arduino UNO](#)
- [Rpi Pico \(µPython\)](#)
- *Documentation de la bibliothèque Adafruit pour Arduino* : [RTCLib](#)
- *A partir du gestionnaire de bibliothèques de l'IDE Arduino, installer :*

RTCLib par Adafruit

2.1.4 installed

A fork of Jeelab's fantastic RTC library Works with DS1307, DS3231, PCF8523, PCF8563 on multiple architectures

ou



- *Télécharger le fichier **DFRobot_DS1307-master.zip** sur [Github](#), le dézipper et le placer dans le dossier : ...\\Documents\\Arduino\\Libraries.*

A venir



2.2.3 Exemples de code

- [Arduino UNO](#)
- [Rpi Pico \(µPython\)](#)
- **Exemple** de l'IDE Arduino pour tester le capteur
Dans l'IDE Arduino, sélectionner : Fichier → Exemples → RTCLib → **DS1307**

A venir

2.3 Module Adafruit ADA3296 (équivalent de MR005)



2.3.1 Présentation

- **Sources**
 - Site : [adafruit](#)
 - Wiki : [Horloge temps réel DS1307 / Genuino](#)
 - Distributeur : [Gotronic](#)
- **Caractéristiques**
 - Alimentation 5 Vcc
 - Sauvegarde: 1 pile 3V type CR1220
 - Informations:
 - secondes, minutes et heures
 - jour, mois, année
 - correction années bissextiles jusque 2100
 - Protocole I²C
 - Sortie drain ouvert : signal logique de fréquence réglable (f=1Hz, 4kHz, 8kHz, 32kHz)
 - Dimensions: 26 x 22 x 5 mm
 - Poids: 2,3g

3. DS3231M

3.1 Présentation



- **Source** : [wiki](#) DFROBOT

Ce RTC adopte la puce DS3231M qui intègre un résonateur MEMS et un compensateur de température. Le DS3231M peut être alimenté par une pile bouton CR1220 et peut toujours maintenir une synchronisation précise même si le secteur est débranché. RTC fournit des informations sur les secondes, les minutes, les heures, les jours, les mois et les années. Il peut ajuster automatiquement la date de fin du mois et la correction des années bissextiles. Le format de l'horloge peut être de 24 heures ou de 12 heures avec indication AM/PM. Il fournit deux alarmes de calendrier programmables et une sortie d'onde carrée de 1 Hz. De plus, la broche de contrôle RST peut être utilisée pour la réinitialisation du microprocesseur.

- **Distributeur** : [GoTronic](#)

- **Caractéristiques**

- Alimentation 3,3 à 5,5 Vcc
- Consommation: 130 μ A
- Sauvegarde: 1 pile type CR1220 (non incluse)
- Informations:
 - secondes, minutes et heures
 - jour, mois, année
 - format: 24 heures ou 12 heures AM/PM
 - gestion des années bissextiles
- Précision: ± 5 ppm ($\pm 0,432$ secondes/jour)
- Interface série I²C : **SLA = 0x68**
- Température de service: -40 à 85 °C
- Dimensions: 22,5 x 21 mm

- **Signaux**

- INT : interruption en sortie de niveau bas ou d'onde carrée de 1 Hz
- RST : Réinitialisation du processeur
- 32K : Sortie 32,768 kHz



- **Documentation**

- Datasheet à télécharger [ici](#)

3.2 Bibliothèques

- [Arduino UNO](#)
- [Rpi Pico \(µPython\)](#)
- Télécharger la fichier **DFRobot_DS3231M-master.zip** disponible sur [GitHub](#) le dézipper et l'installer dans le dossier : ...\\Documents\\Arduino\\Libraries



A venir



3.3 Exemples de code

- [Arduino UNO](#)
- [Rpi Pico \(µPython\)](#)
- **Ressource** : [Wiki](#) DFROBOT
- **Exemple** pour tester l'HTR

*.cpp

```
// Code testé sur Arduino Uno et Arduino MKR 0
// Interruption générée toutes le 1s par une HTR à DS3231M

#include "DFRobot_DS3231M.h"

const byte interruptPin = 2; // 6 pour la MKR Zéro, MKR 1010
int i=0;

DFRobot_DS3231M rtc;

void setup() {
  pinMode(interruptPin, INPUT_PULLUP);
  Serial.begin(115200);
  /*On attend que l'initialisation du DS3231M soit complète*/
  while (rtc.begin() != true) {
    Serial.println("Circuit non initialisé, vérifier les connexions.");
  };
  delay(1000);
}
```

```
}
/*!
  *@brief Set the value of pin sqw
  *@param mode eDS3231M_OFF = 0x01 // Not output square
wave, enter interrupt mode
  *@n      eDS3231M_SquareWave_1Hz = 0x00 // 1Hz square wave
  *@n      eDS3231M_SquareWave_1kHz = 0x08 // 1kHz square wave
  *@n      eDS3231M_SquareWave_4kHz = 0x10 // 4kHz square wave
  *@n      eDS3231M_SquareWave_8kHz = 0x18 // 8kHz square wave
  */

rtc.writeSqwPinMode(eDS3231M_SquareWave_1Hz);
// digitalPinToInterrupt convertit le n° de broche en n°
d'interruption, exemples
// Arduino UNO : interruption n°0 sur connecteur broche 2
// Arduino MKR Zéro : interruption n°6 sur broche 0
attachInterrupt(digitalPinToInterrupt(interruptPin),
interrupt_handler, RISING);
}

void loop() {
  // Mettre le code répétitif ici
}

void interrupt_handler() {
  Serial.println("Hello " + (String)i++);
}
```

A venir

From:

<https://webge.fr/dokuwiki/> - **WEBGE Wikis**

Permanent link:

<https://webge.fr/dokuwiki/doku.php?id=materiels:temps:htr:htri2c&rev=1714807987>

Last update: **2024/05/04 09:33**

