# **Cahier des charges**

# **ECS\_WEB\_DATA**

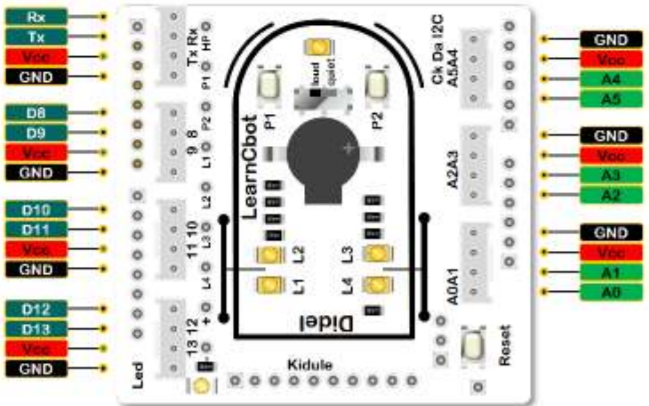
# Hardware

Arduino Yun Rev 2

Display « I2C 2004 204 » 4x20 chr, port I2C, add 0X27

RTC DS3231, port I2C, add standard

## Shield LearnCbot



SW1 « P1 » port 2 / actif LOW

SW2 « P2 » port 3 / actif LOW

LED1 VERTE « L1 » port 4 / actif HIGH

LED2 ROUGE « L2 » port 5 / actif HIGH

LED3 ROUGE « L3 » port 6 / actif HIGH

LED4 VERTE « L4 » port 7 / actif HIGH

## Sondes DS18B20 sur ports Grove suivants

Sonde 1 – input Grove 8 (avec couplage DiGrove)

Sonde 2 – input Grove 10

Sonde 3 – Input Grove 12

## Lecture état pompe

1-Bit AC 220V Optocoupler Isolation Module – input 9 Grove (avec couplage DiGrove)

|  |  |  |  |
| --- | --- | --- | --- |
| Signal | Module Optocopler | Couleur fil | Grove |
| 0V | GND | Noir | 0V |
| 5V | VCC | Rouge | 5V |
| I/O | OUT (pullup 47K) | Blanc | Data 9 |
| I/O | * (inutilisé) | Jaune | Data 8 Sonde 1 |

# Software - Agencement des fonctionnalités

## Sondes - principe d’utilisation

Chaque sonde utilise une entrée définie. On ne se base pas sur le mode OneWire et l’ID de la sonde.

Le soft est monté ainsi : une instance de OneWire(n° canal) par sonde ; un tableau de pointeurs pour les parcourir efficacement. Il est initialisé dans main.cpp .

## Définitions

Le fichiers ecs-web-data.h contient :

* Les constantes hard
* Les variables globales, avec la CLASS de stockage
* Les prototypes

## Démarrage :

* Init I/O
* Instancier objets LCD, horloge RTC et switchs SW1 et SW2
* Init I2C
* Init LCD + info
* Init+Clear variables
* Init Serial à 9600Bd
* ~~Lecture paramètres EEPROM~~
* Init Bridge Yun
* Init Filesystem
* Enable Watch dog à 2 sec.
* Lire le temps Unix
* Start polling 1 sec
* Start polling scan
* Log démarrage

## Boucle principale Arduino :

* Scheduler cycle
* Webservice

## Boucle de scan à 1 seconde :

* Inverser LED témoin
* Lire le temps ~~Unix~~ RTC
* Si secondes modulo 5 :
  + Lire sondes
* Display des valeurs
* Si minutes modulo 10 : (soit toutes les 10 minutes)
  + Imprimer les valeurs mesurées sur sériel
  + Enregistrer les valeurs mesurées
* Test RAM avaible
* Reset Watchdog

## Boucle scan rapide à 30 ms:

* Scan l’état des switchs + état pompe
* Calcule le taux d’activité de la pompe

# Display 4 x 20 caractères

## Agencement des messages

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| y | y | y | y | - | m | m | - | d | d |  | h | h | : | m | m | : | s | s | X |
| i | n | f | o |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| P | A | N |  |  | E | C | S |  |  | S | O | L |  |  | P | O | M | P | E |
| 7 | 0 | . | 4 |  | 6 | 3 | . | 3 |  | 5 | 5 | . | 6 |  | 0 | 7 | 8 | % | \* |

### Ligne 0

Date et heure, avec [X] si valeur de time NOK ; sinon blanc, actualisé à la seconde

### Ligne 1 :

Info courante / messages de log

### Ligne 2 :

Texte fixe, pour champs ligne 3

### Ligne 3 :

Valeurs instantanées, actualisées tts les 5 secondes

## Menus

### Changement par les switchs

SW1 activé, à chaque impulsion : incrément de l’index menu

SW2, décrément de l’index menu

### Changement par le sériel

‘+’ : incrément index menu

‘-‘ : décrément index menu

### Index menu (0..4)

0) affichage normal : Ln1 « marche normale »

1) Ln1 « Dernier enregistrement » ; Ln3 montre les valeurs

2) Ln1 « Dernier log » ; Ln2 date/heure ; L3 log

3) Ln1 « Version soft » ; Ln2 date/heure du HEX ; Ln3 Nom et N° de version

4) Ln1 « Dernier boot » ; Ln2 date/heure du boot

# Lecture des sondes

## Méthode d’échantillonnage

Avec un temps de conversion pas forcément maitrisé : indiqué de 750 ms, mais avec le mode parasite en ONE WIRE, il s’agit de décomposer le lancement de la conversion et la récupération des résultats.

L’exemple original de lecture est modifié ainsi, dans DS18B20\_Temp\_3sondes

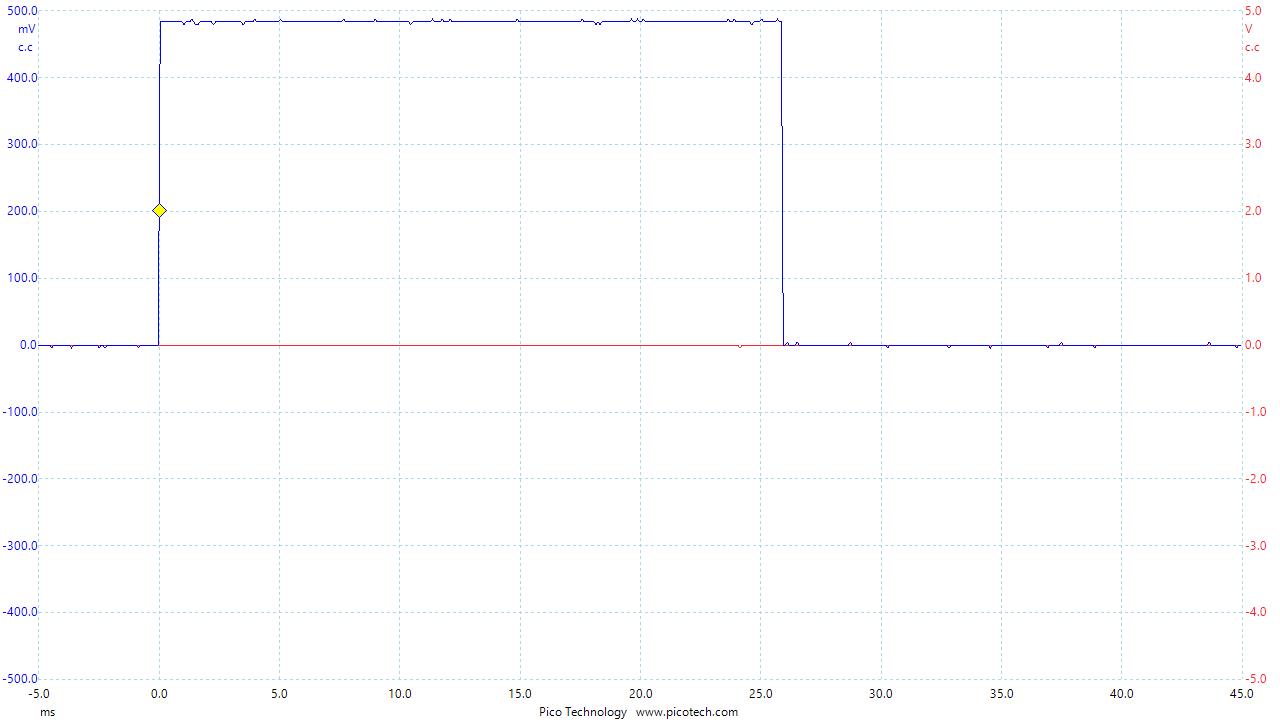
( c:\Users\Masur\myCloud\Dev\PlatformIO\DS18B20\_Temp\_3sondes\src\DS18B20\_Temp\_3sondes.cpp )

* Chaque sonde à son port
* On ne teste pas le modèle de sonde : fixé sur DS18B20
* Fonction démarrage de la conversion : byte start\_temperature\_conversion(byte n)
* Fonction de lecture de la température : float get\_temperature(byte n)

Les fonctions seront appelées par la boucle principale, alternativement à la seconde. Une boucle avec n [0..2] permet de traiter les 3 sondes.

## Mesure du temps de lecture sonde

La durée totale de la lecture et conversion est de 26 ms, indépendamment de la sonde.



### Cas d’erreur (sonde coupée)

La lecture retourne un statut de -1 et le message « CRC error » ; la conversion rend 0.0 °C. Le rétablissement de la liaison rétablit des données correctes.

# Affichage

L’affichage dépend du menu ; donc des infos d’enregistrement et de log.

## Menu 0 – date et heure, datas

L’affichage de la date/heure est indépendant du menu : fonction appelée chaque seconde display\_date() utilise la ligne 0.

* Lire la date/heure
* Ajouter le chr ‘X’ si non synchronisé par le NTP
* Afficher le string en Ln0

Si le menu est 0 depuis plus de 5 minutes (300 sec.), afficher en Ln1 « Marche normale »

A la seconde modulo 5, soit toutes les 5 secondes, afficher les températures et % d’utilisation de la pompe par la fonction display\_meas() en Ln2 et Ln3.

## Menu 1 - enregistrement

Afficher le dernier enregistrement par la fct display\_last\_rec() avec mention « Dernier enregistrement » en Ln1 ; date/heure champs en Ln2 et valeurs en Ln3.

## Menu 2 - log

Afficher le dernier log par la fct display\_last\_log() avec mention « Dernier log » en Ln1 ; date/heure champs en Ln2 et log en Ln3.

## Menu 3 – version

Affiche par la fct display\_vers() « Version soft »  en Ln1 ; date/heure du HEX en Ln2 et nom en Ln3.

## Menu 4 – boot

Affiche le message « Dernier boot » en Ln1 et la date/heure du soft en Ln2.

Affiche le nb d’enregistrements en Ln4.

## Copie des textes sur le sériel

Afin d’éviter la répétition du message lors du rafraichissement de l’afficheur, un mécanisme à bit est mis en place.

Variable globale pour faire le lien : byte ser\_copy.

Chaque ligne du LCD à un bit de contrôle. Une demande de copie de toutes les lignes se fait par :

ser\_copy = 0xF ;

Lors d’un changement de mode ou demandé via interrogation du sériel, les bits des lignes concernées (0..3) sont mis à 1. Lors de l’écriture sur l’afficheur :

* Si le bit est à 1, copier le texte sur le sériel et mettre le bit à 0

Effet de bord / inconvénient : en cas de demande de toutes les lignes, elles ne s’affichent pas dans l’ordre sur le sériel, mais dans l’ordre du rafraichissement (accepté).

# Gestion des logs

## Sorties

Le message est affiché en Ln 1 du LCD.

L’enregistrement avec timestamp dans fichier de log

Affiché sur sériel ~~avec timestamp~~ tel quel, via le mécanisme §5.6

## Temporisation d’affichage

Si le log est informatif,

* Affiché sur LCD pendant 10 s max
* Affichage dernière erreur, hors tempo

## Tampon message d’erreur

Si erreur : stockée dans tampon err\_msg[]

## Logique affichage

Appelée chaque seconde :

* Si décompte tempo\_msg : décrémenter et sortir
* Sinon :
  + Si err\_act : afficher err\_msg
  + Sinon : afficher « marche normale »

# Gestion du sériel

## Input

Lecture chr entrant :

* ‘+’ : incrémenter n° du menu
* ‘-‘ : décrémenter n° du menu
* ‘ ‘ (espace) : afficher dernier log
* Chiffres 0..4, n° du menu

## Output

Messages selon logs §6.1

# Gestion des pannes de sonde

## Mémorisation

Si une sonde ne répond pas, l’info err\_probe est positionnée ; et le n° de la sonde enregistré dans err\_probe\_n (1, 2, ou 3).

## Log d’erreur de sonde

A la transition de l’état d’erreur, un message est logué en mentionnant le n° de sonde.

Lorsque l’erreur disparait, le log correspondant affiche le retour du fonctionnement.