Approche : lecture d’un compteur d’énergie

Implémentation Arduino Yun

Projet e230 - contenu

Energie – de quoi parle-t-on?

E230 – définition hardware

E230 – soft embarqué

Mise au point modules

Résultats

Site WEB et présentation

Analyse de consommation

Energie

Unité: Kilowattheure (kWh)

Société à 2000 W, concept de l'ETHZ en 1990

http://www.2000watt.ch/fr/

Cela représente : 2 x 24 x 365 = 8760 kWh / an

Actuellement, on est plutôt à 6000 W…

Consommation électrique d'un ménage : ~4,5 MWh / an

Pointe de puissance: par tranche de 15 minutes

Régulation de puissance au quart d'heure : LKW

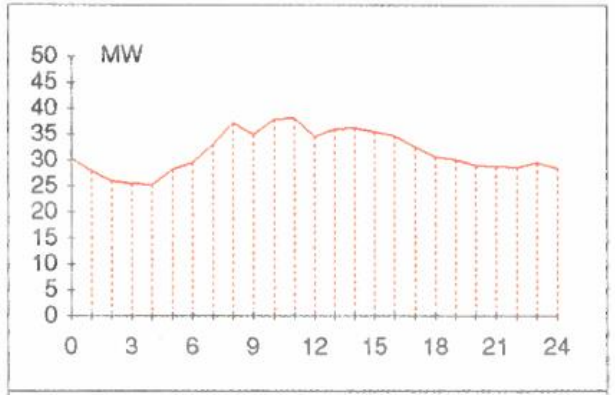
Energie-Info

LIECHTENSTEINISCHE KRAFTWERKE

		22. Juni 1994	Mittwoch
Stromerzeugung	KW Samina	238'160	kWh
	KW Lawena	86'500	kWh
	Klein KW'e und BHKW'e		kWh
	Gesamterzeugung	**324'660**	**kWh**
	Strombezug von NOK	**436'455**	**kWh**
Stromabgabe	an Landesnetz	702'480	kWh
	an NOK	58'635	kWh
	Gesamtabgabe	**761'115**	**kWh**

Netzlast

Höchstlast	11.30	Uhr
KW Samina	9'960	kW
KW Lawena	3'800	kW
Bezug von NOK	**26'340**	**kW**
Total Höchstlast	**40'100**	**kW**

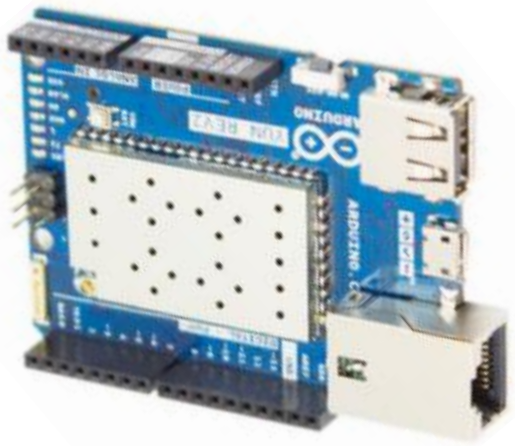


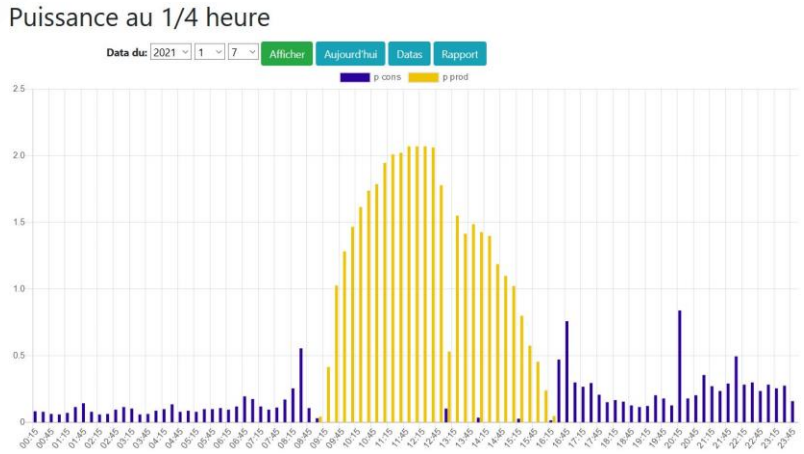
Projet e230 souhait:

- Lecture du compteur en continu
- Affichage local
- Stockage sur mémoire flash
- Copie des données sur l'Internet
- Représentation graphique de la puissance
- Distinguer consommation et production

Facile… comme les projets précédent de Geiger et ECS monitor ?

Flux de données





Projet e230 - contenu

Energie – de quoi parle-t-on?

E230 – définition hardware

E230 – soft embarqué

Mise au point modules

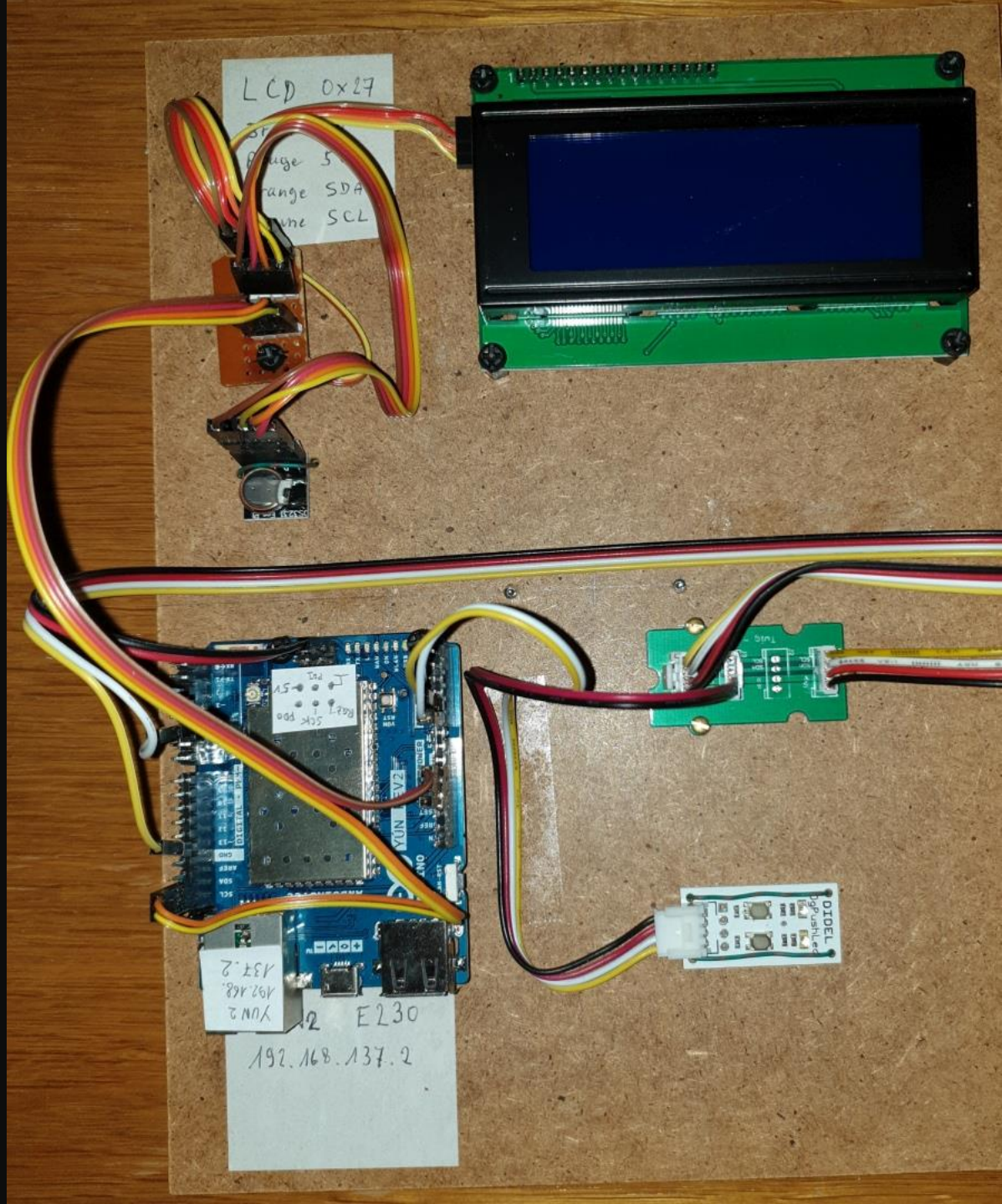
Résultats

Site WEB et présentation

Analyse de consommation

Définitions hardware

- Display I2C
- Horloge I2C
- *Bus I2C*
- Yun avec:
  - mémoire ~~SD~~ USB
  - Wifi
  - Ethernet
- *Bus Grove*
- 2 Boutons



Boutons, afficheur & menus

Principes issus de «Ventilo»

- Boutons
- Menus/affichage
- Réglage RTC

Principes issus de «Geiger»

- Gestion de l’enregistrement
- RTC avec synchro NTP
- Fichier de log

Principes issus de «ECS Monitor»

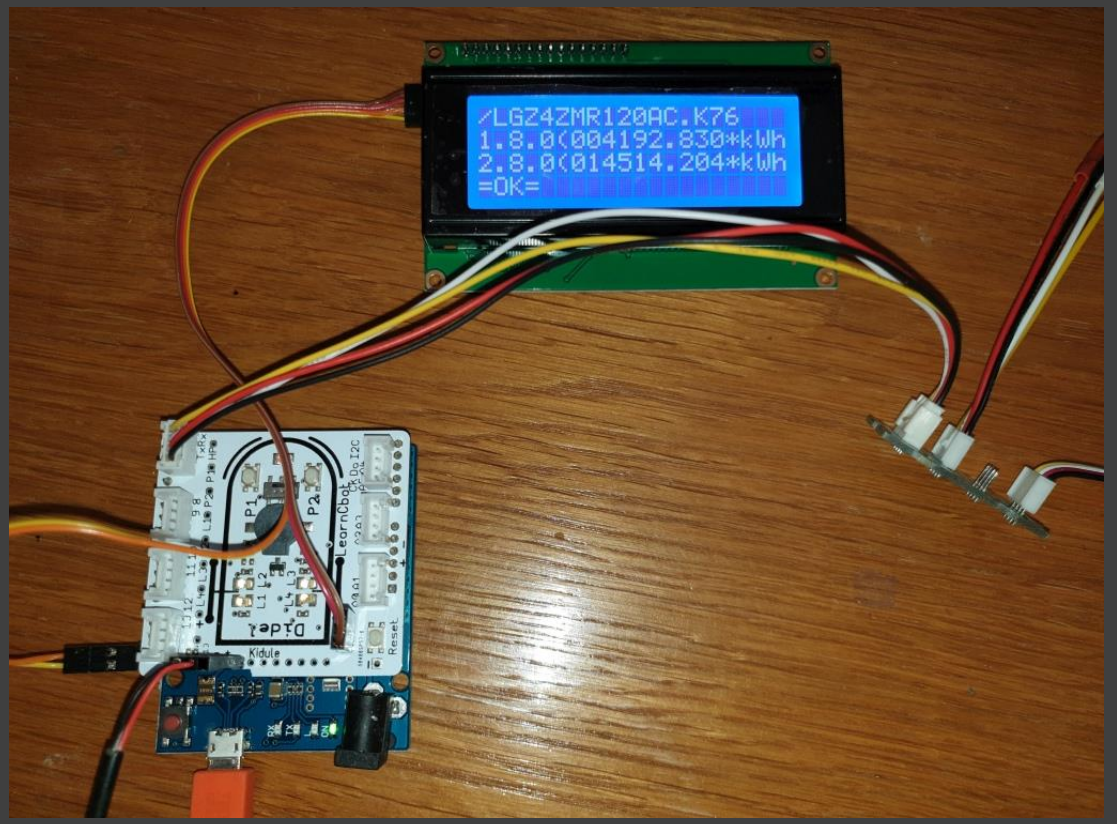
- Gestion des boutons
- Gestion des menus/affichage
- Gestion des messages

Soft : pilote interface compteur Landis&Gyr

- Lecture, pilote «e230»
- Manage 2 vitesses 1200 Bd, 9600 Bd
- Pilote de **J-M Paratte**
  - Un seul fichier Arduino
  - Sortie «compatible Excel»
  - Prévu pour 1 seule interrogation!
  - Relancer par reset!
  - Peu ou pas prévu pour afficheur!

- Utilise 2 ports série
- **J-M Paratte** a testé son pilote sur:
  - Arduino Uno,
  - Leonardo
  - ESP32
  - ESP8266

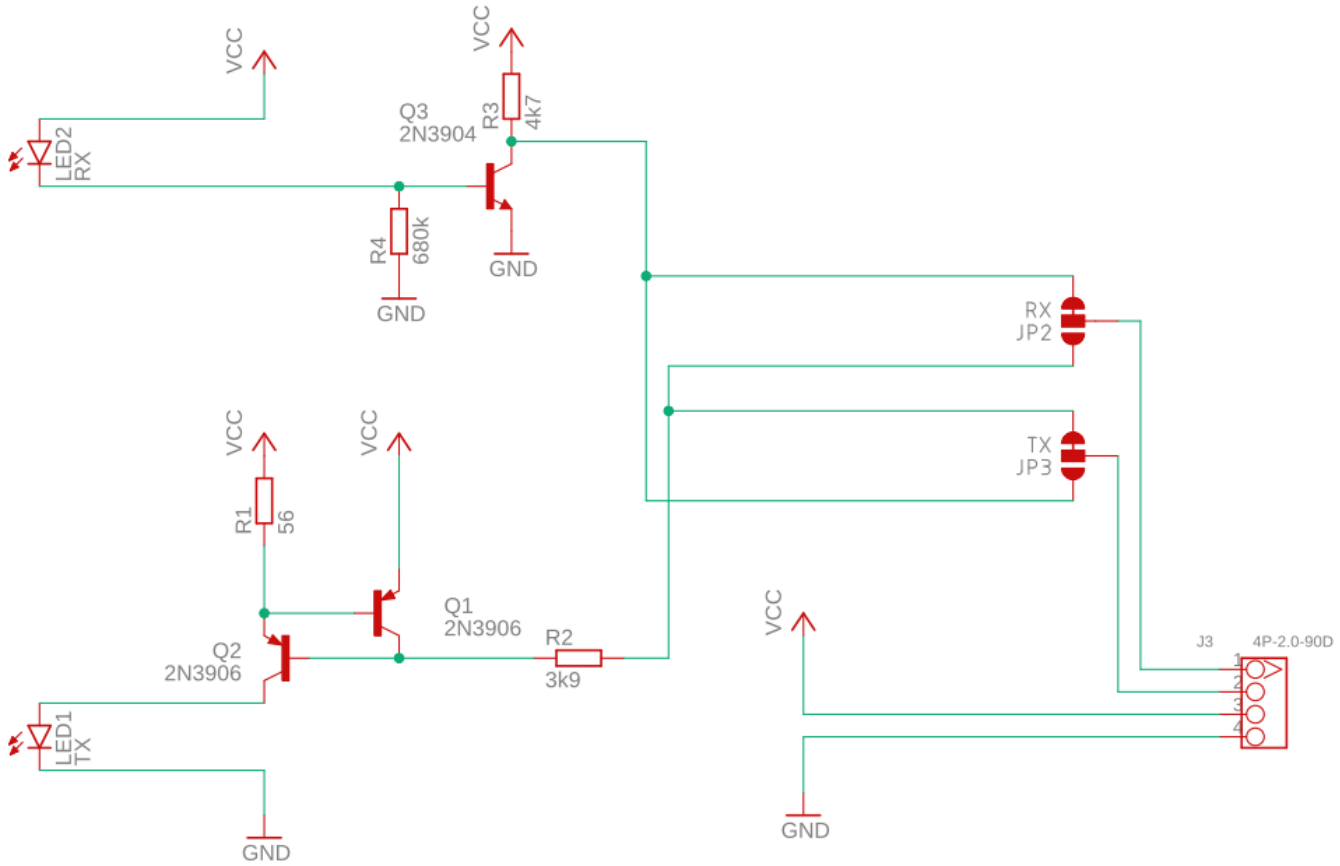
- Essai des librairies ~~SoftwareSerial~~ et AltSoftSerial pour port #2



Test lecture

Allonger le cable Grove évite de travailler sous l'escalier!

Schéma original JMP



Premiers tests

Serial #2

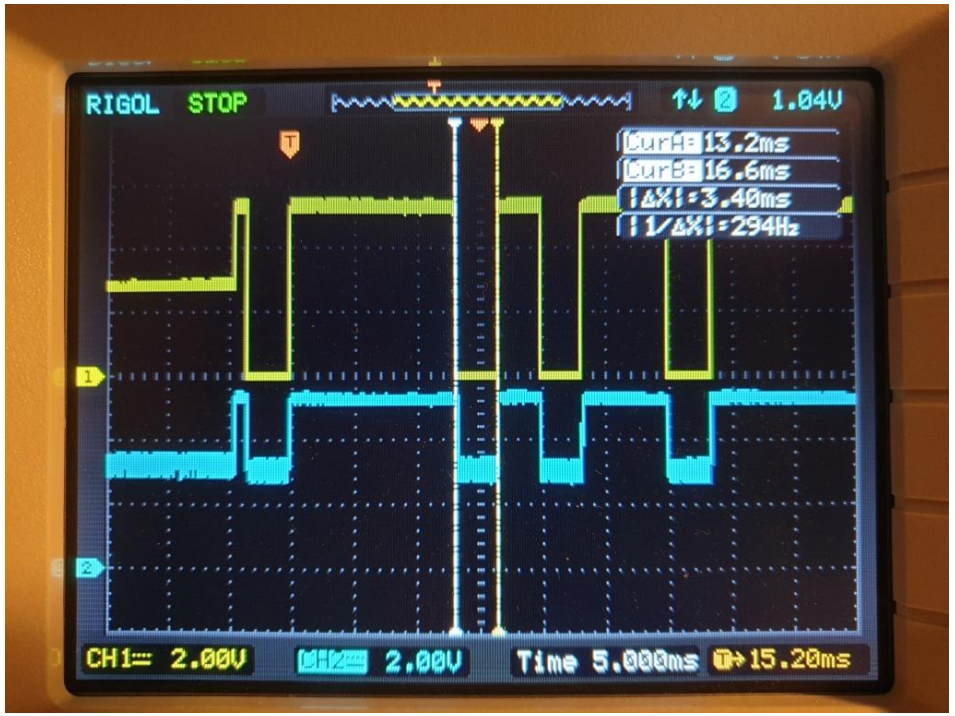
- PINs Rx-Tx imposées
- Lecture compteur aléatoire
- Leonardo défectueux…
- Test : bouclage optique
- Réception Ok avec Yun!

Bouclage: montage miroir ☺

Premiers tests

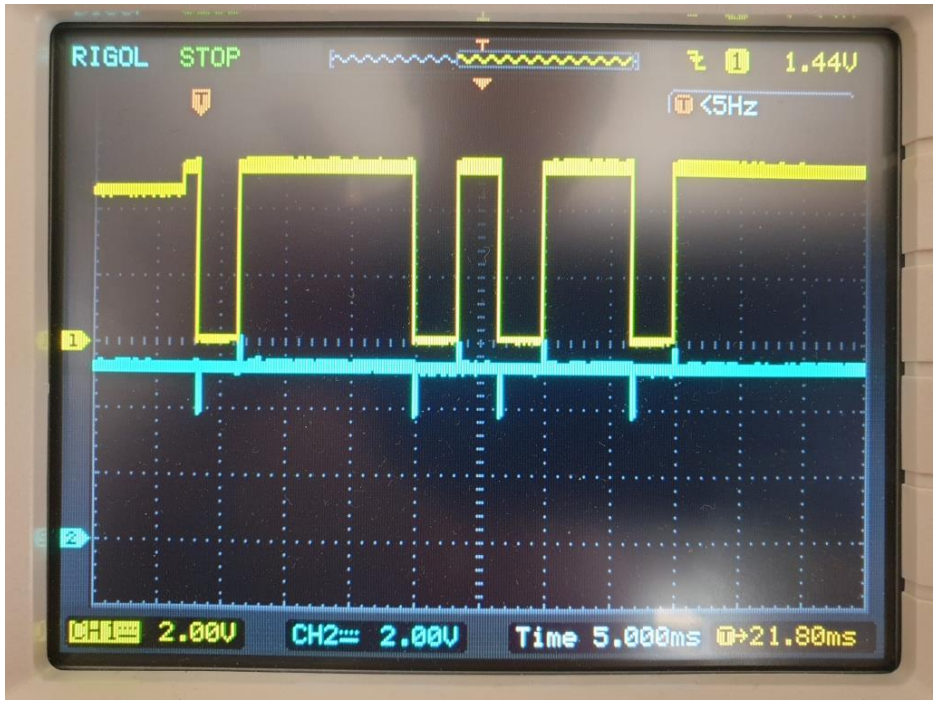
Sériel – scope Rx - Tx

- Diaphonie importante



Pull-Up : R3 à 1K5 (au lieu de 4K7)

- Après correction



Allure des données du compteur

Champs

`1.8.0(004281.837*kWh)`

«1.8.0»

«004281.837»

«*»

«kWh»

Exemple

Ligne complète

- Identifiant
- Valeur en ASCII
- Séparateur
- Unité de mesure

Buffer complet

Issu du pilote V05 de J-M Paratte.

Chaque élément est séparé par ‘\0’ .

Exemple pour 32.7.0:

«32.7.0\0238\0V\0»

Le CR/LF est aussi un ‘\0’ :

Économie: 31 octets

```
/LGZ4ZMR120AC.K76
F.F.0(00000000)
0.0.2(  978880)
C.1.0(30863358)
C.1.1(        )
1.8.1(000000.000*kWh)
1.8.2(004281.837*kWh)
1.8.0(004281.837*kWh)
2.8.1(000000.000*kWh)
2.8.2(014605.516*kWh)
2.8.0(014605.516*kWh)
32.7.0(238*V)
52.7.0(236*V)
72.7.0(237*V)
31.7.0(000.67*A)
51.7.0(000.86*A)
71.7.0(000.99*A)
36.7.0(000.09*kW)
56.7.0(000.11*kW)
76.7.0(000.07*kW)
33.7.0(0.81)
53.7.0(0.57)
73.7.0(0.32)
C.7.0(0010)
C.7.1(0029)
C.7.2(0026)
C.7.3(0024)
0.2.0(K76-0-9)
C.5.0(0004E0F0)
C.90.1(000000030863358)
!
```

Projet e230 - contenu

Energie – de quoi parle-t-on?

E230 – définition hardware

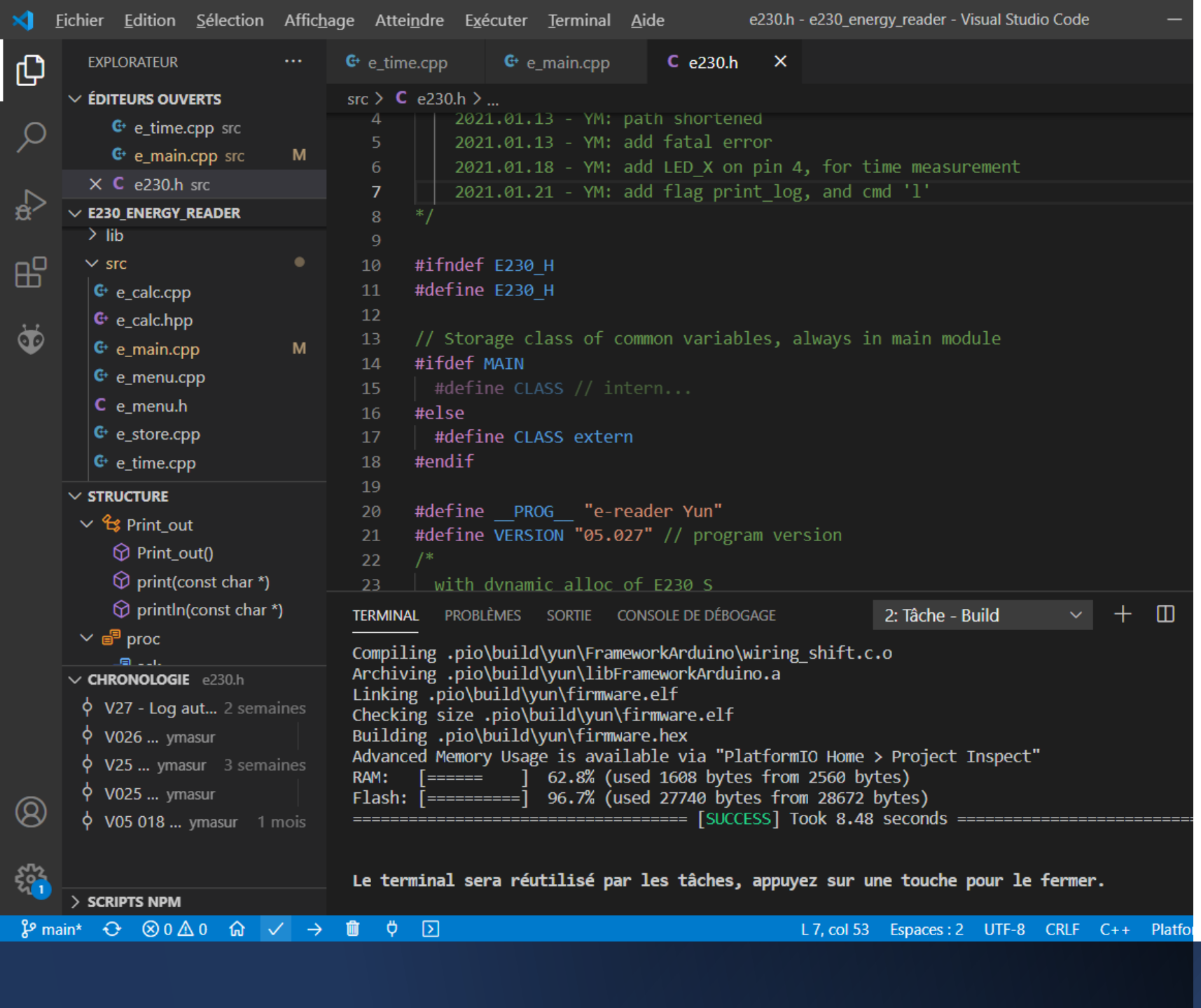
E230 – soft embarqué

Mise au point modules

Résultats

Site WEB et présentation

Analyse de consommation



Méthode de développement

- VS-CODE et
- PLatformIo

Module CALC

- Toucher aussi peu que possible à e230_05.h, le pilote
- Commenter les parties identifiables
- Identifier et manipuler les données sous une forme structurée
- Tenir compte du sens de l'énergie

Structuration des données:

- Par l'identifiant
- Par la phase
- Par nom symbolique (V, kW, A…)
- Conversion ASCII -> float
- Calculs de valeurs globales
- Calcul de la P moyenne pendant 15 minutes

e_calc.hpp

Test et mise au point de CALC

- MS Visual Studio 2019
- Compilateur Windows natif, EXE en mode DEBUG
- Fichier d'essai «\test\test_e230\test_e230.cpp»
- Comporte des données de test = un buffer complet
- Inclusion relative des sources Arduino:
- `#include ".\..\..\src\e230.h"`
- `#include ".\..\..\src\e_calc.hpp"`

Génère d'importants fichiers OBJ, PDB, IDB, …

Modification du code

Inclusion/exclusion selon le mot clé **_WIN32**


```
#ifndef _WIN32
#include <Arduino.h>
#include <SoftwareSerial.h>
#include <FileIO.h>
#include <avr/wdt.h>
#include <string.h>
#include <Wire.h>
#include <time.h>
#include <RTClib.h>
// #include <jm_Scheduler.h>
#include <jm_LCM2004A_I2C.h>

#else // _WIN32
#warning "only for test purpose"
#include <stdio.h>
#include <string>
#include <stdlib.h>     /* atof */

#endif // _WIN32 not defined
```

Partie Arduino

Partie Windows

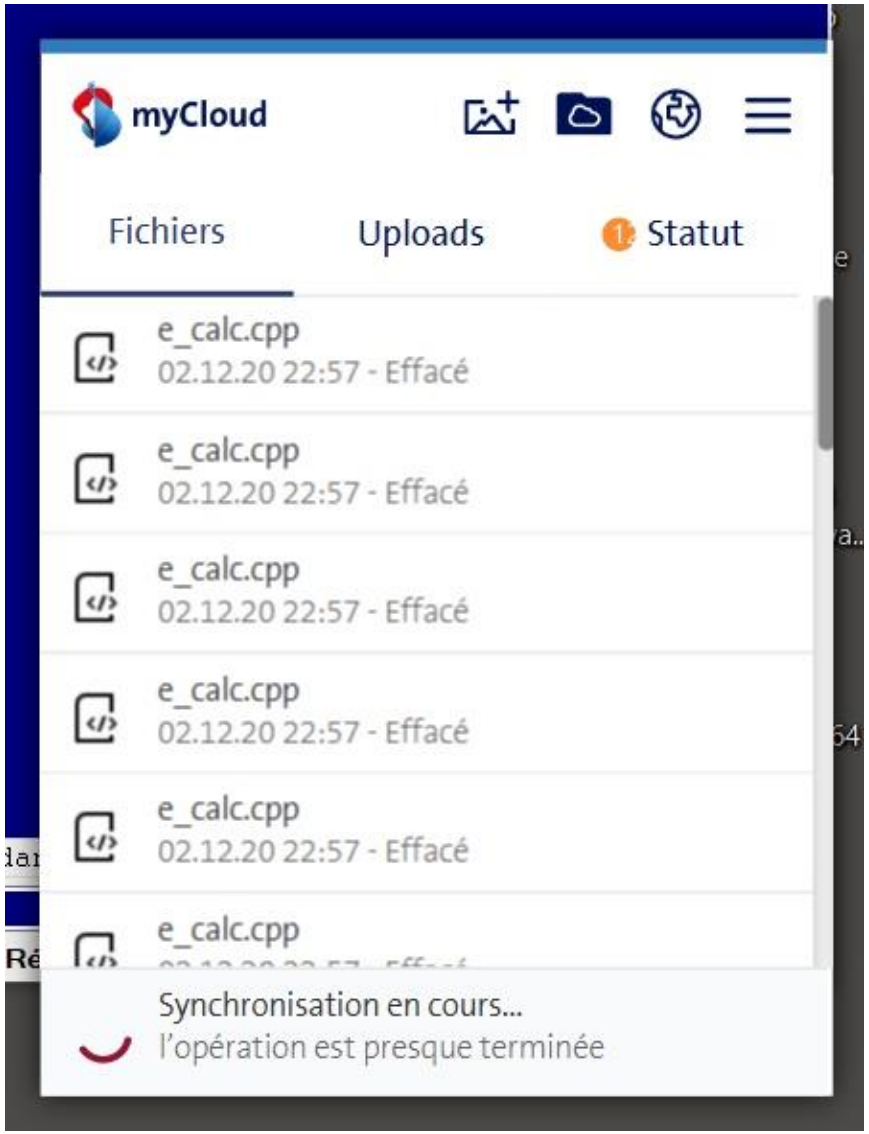
Adaptation des fonctions print

- Imprimer un float sur Arduino donne: ‘?’
- En C, on écrit: printf(«var= %f», var);
- Arduino, fct dtostrf(var, champ, décimales, *buffer);
- Macro, pour permettre le DEBUG

```
 // Arduino use dtostrf(), unused in WIN32. This macro expand into a compatible sprintf()
#define dtostrf(VAR, UNIT, DECIM, BUF) (sprintf(BUF, "%" #UNIT "." #DECIM "f", VAR))
```

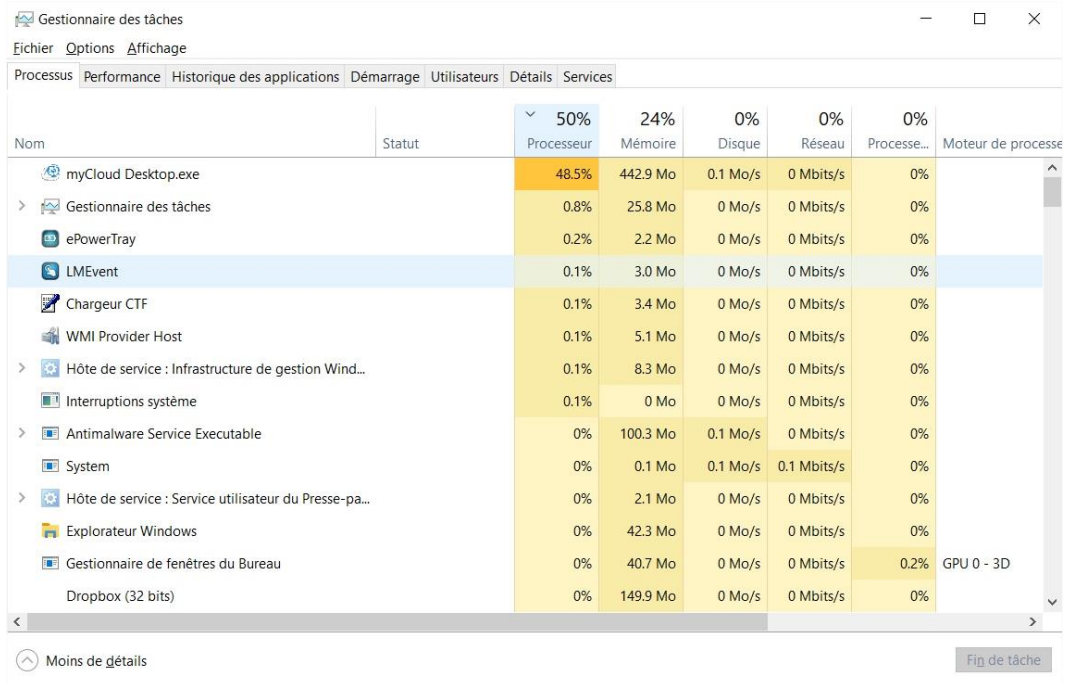
Ça patine !!

Intermède: Problème de cloud



Patinage confirmé!

Intermède cloud



Panne de myCloud

- Méli-mélo de fichiers, conflits
- **Où sont les modifications??**
- Pollution entre le PC principal et le PC utilisé pour les tests
- myCloud boucle et fait chauffer le CPU!
- Le support indique qu'il ne faut pas d'EXE dans les répertoires cloudés
- Nettoyage…
- Suite momentanée avec Dropbox
- Tests avec Kdrive d'Infomaniak ✓

Leçon

N’utiliser qu’un PC pour développer –OU–

Un cloud fiable

Pousser les fichiers sur un cloud qu’après un pas en avant effectué

Mise ensemble des modules

- Création d'un fichier de référence (proto, variables) dans «230.h»
- Contient la CLASSe de stockage des variables
- Création/adaptation des menus et display selon CDC
- Faire gober le tout à la compilation…

Problème:

Mémoire RAM utilisée 102% !

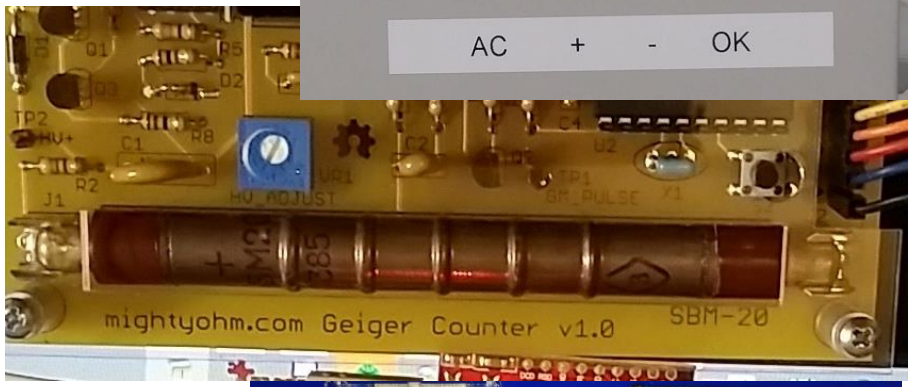
Mémoire flash utilisée 115.9% !

Mémoire, comparaison d'autres projets

- **Ventilo**: **Arduino, 4 Menus, RTC, LCD**

RAM: 47.0% (used 962 bytes from 2048 bytes)

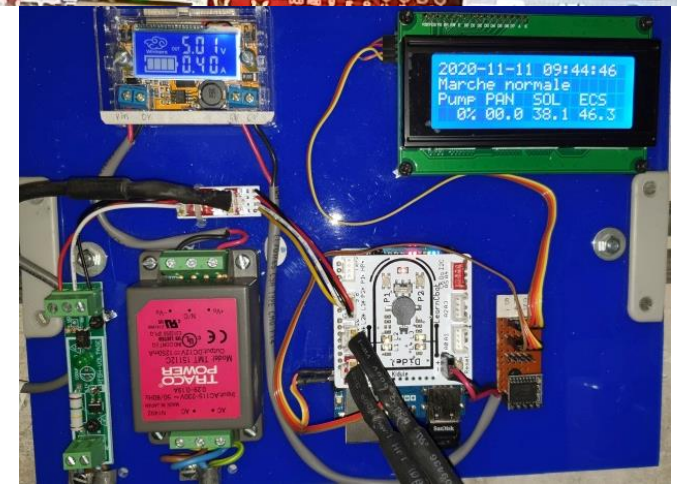
Flash: 49.1% (used 15832 bytes from 32256 bytes)



- **Geiger** : **Yun, Web, RTC, LCD, enregistrements, log**

RAM: 69.9% (used 1790 bytes from 2560 bytes)

Flash: 86.0% (used 24644 bytes from 28672 bytes)



- **ECS Monitor : Yun, Web, 4 menus, RTC, LCD, enregistrements, log**

RAM: 57.8% (used 1480 bytes from 2560 bytes)

Flash: 95.7% (used 27442 bytes from 28672 bytes)

Leçon:

Ne pas charger un soft avec >= 100% d’utilisation mémoire.

Risque de ne plus charger un nouveau HEX!

Projet e230 - contenu

Energie – de quoi parle-t-on?

E230 – définition hardware

E230 – soft embarqué

Mise au point modules

Résultats

Site WEB et présentation

Analyse de consommation

Mise au point du soft

Assembler, retailler et adapter des modules existants.

Réduction 1

- Suppression des services WEB :
  - RAM: 86.7%
  - Flash: 87.5%

Mais… rien ne marche bien longtemps.

Ne passe même pas le setup() !

Plante dans les fct d'affichage LCD.

Ecriture simplifiée sur LCD: lecture de e230 ✓

Décision: réduire l'occupation de la RAM.

Réduction 2

- Recherche des .OBJ potentiellement supprimables
- Abandon de la lib multitâche JM_scheduler :-(
- Utilisation de millis() pour les appels récurrents, soit:
  - Scan des boutons à 20 ms
  - Gestion des données à la seconde

A la première tentative d'enregistrement de log sur la mémoire USB, **plantage** ☒

-> suppression du log

A l'enregistrement des 15 minutes, **plantage** ☒

Reste un problème d'enregistrement!!

Réduction 3

- Suppression de tout «Stream»
- Gros travail de redesign des fct (appel, variables…)
- Utilisation de const char * au lieu de char *
- Minimiser la taille et l'utilisation de buffers statiques

Suppression de la mémorisation du start, et minimiser les messages de logs à 20 bytes.

Résultats:

- Menus affichage énergie et tension ✓
- Enregistrement de log ✓
- Plantage de l'enregistrement du quart d'heure ☒

Décision à mi-parcours

- J’avance dans les menus.

Menu 0: énergie

Réduction 4

- Modifier l'objet E230_S (soit le pilote) pour l'activer dynamiquement: new et delete
  - Utilisation à éviter dans de l'embarqué…
- Variables en float au lieu de double*
- Toutes les chaînes de chr en mode F(«message»)
  - Diminue l'emprise mémoire
- Raccourcir les textes
  - Diminue l'emprise Flash
- Vérification de la mémoire libre: freemem()
- Appeler les fcts si possibles en phase «wait»
  - Ex. interpréteur 'P' -> print()
- Suppression du menu «dernier log»

* Vérifier l'effet!!

Précision: float?

- Float : 32 bits
  - 8 bits exposant
  - 24 bits mantisse
  - ~7 chiffres significatifs

Programme de test, avec valeur de départ et increment de delta, arrêt lorsque l'incrément n'a plus d'effet.

Résultat, test pour float:

Delta	Val	Occurences
0.001	32768.000	10149545
0.002	65536.000	17657685
0.010	262144.000	21427235

Précision: double ?

- Double 64 bits
  - 11 exposant
  - 53 mantisse
  - ~16 chiffres significatifs
- Test avec delta = 0.001 pas terminé après 2h…
- Ici, 9 chiffres suffisent.

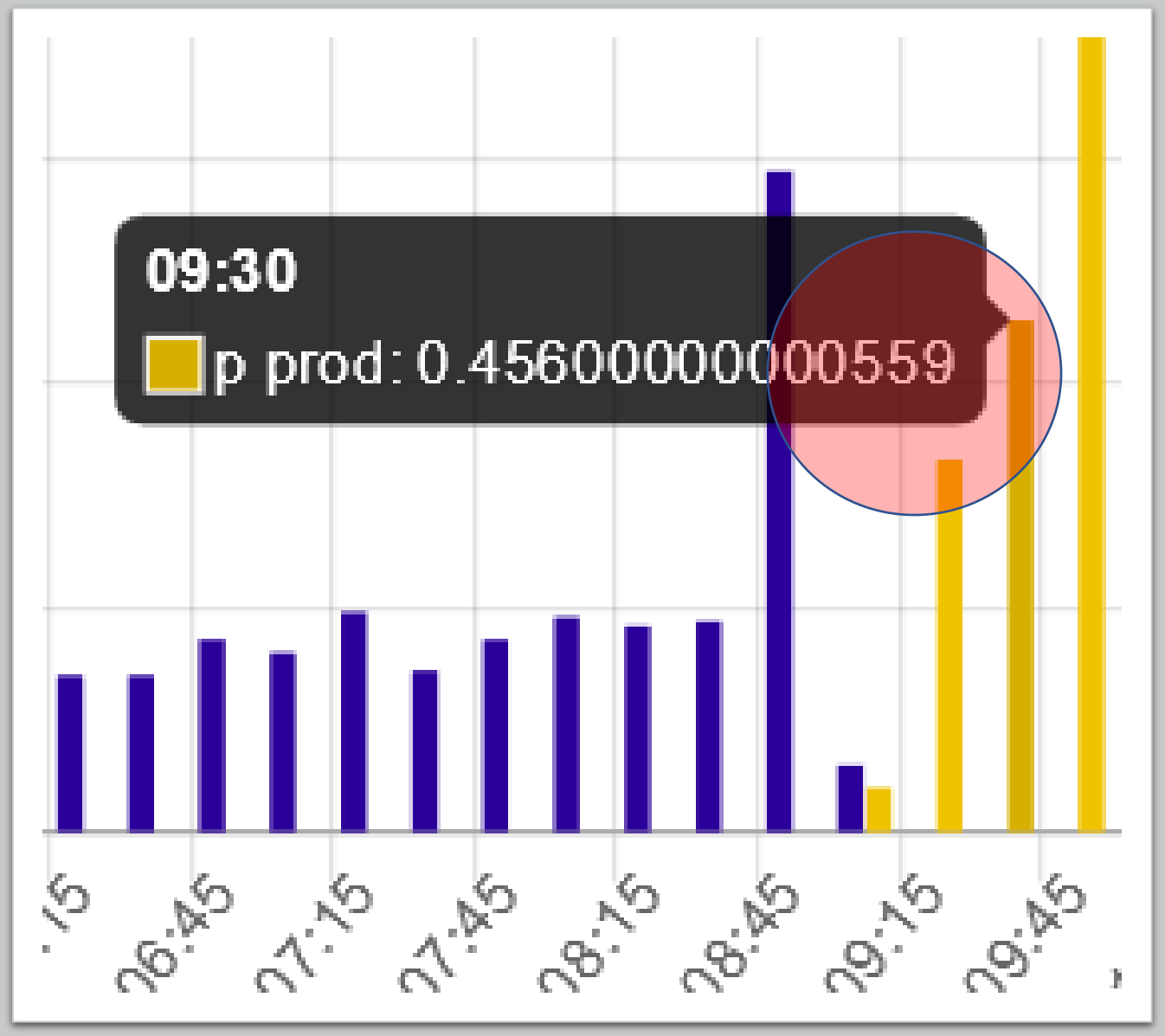
Résultat, test pour float:

Delta	Val	Occurences
0.001	32768.000	10149545
0.002	65536.000	17657685
0.010	262144.000	21427235

Float – avec PHP

Le delta laisse apparaître des scories sur les 3 derniers digits.

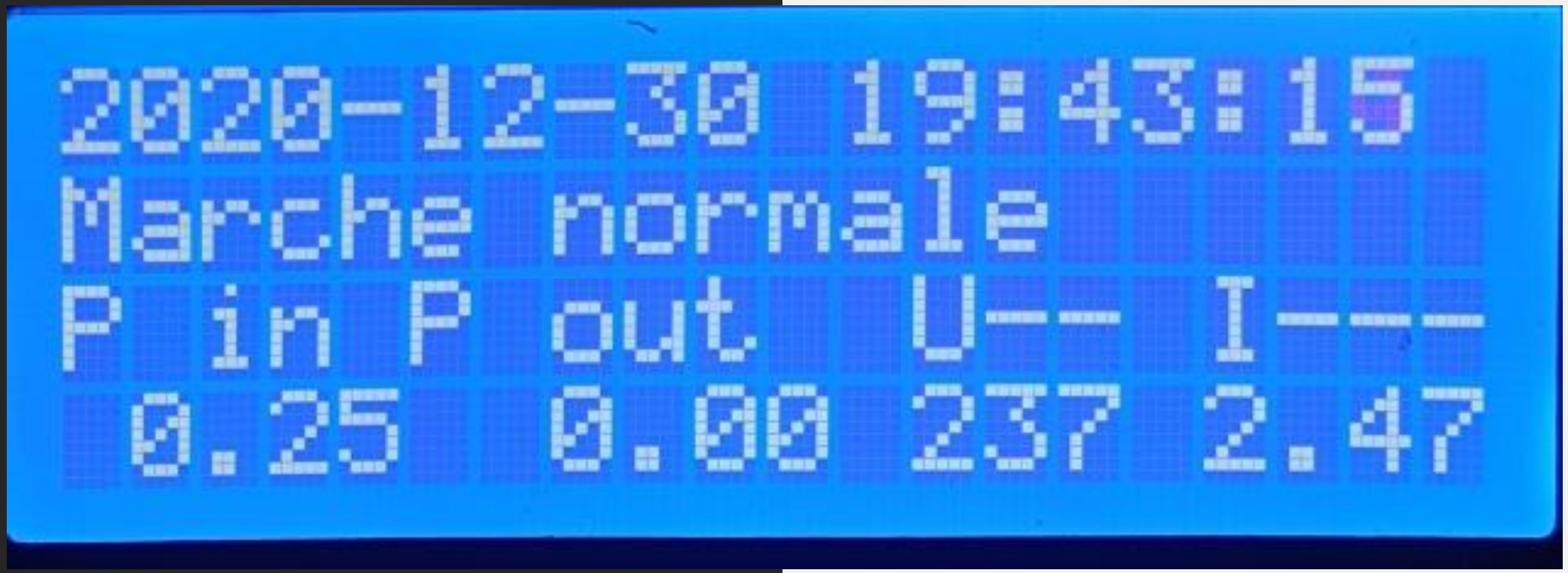
Aucune importance…



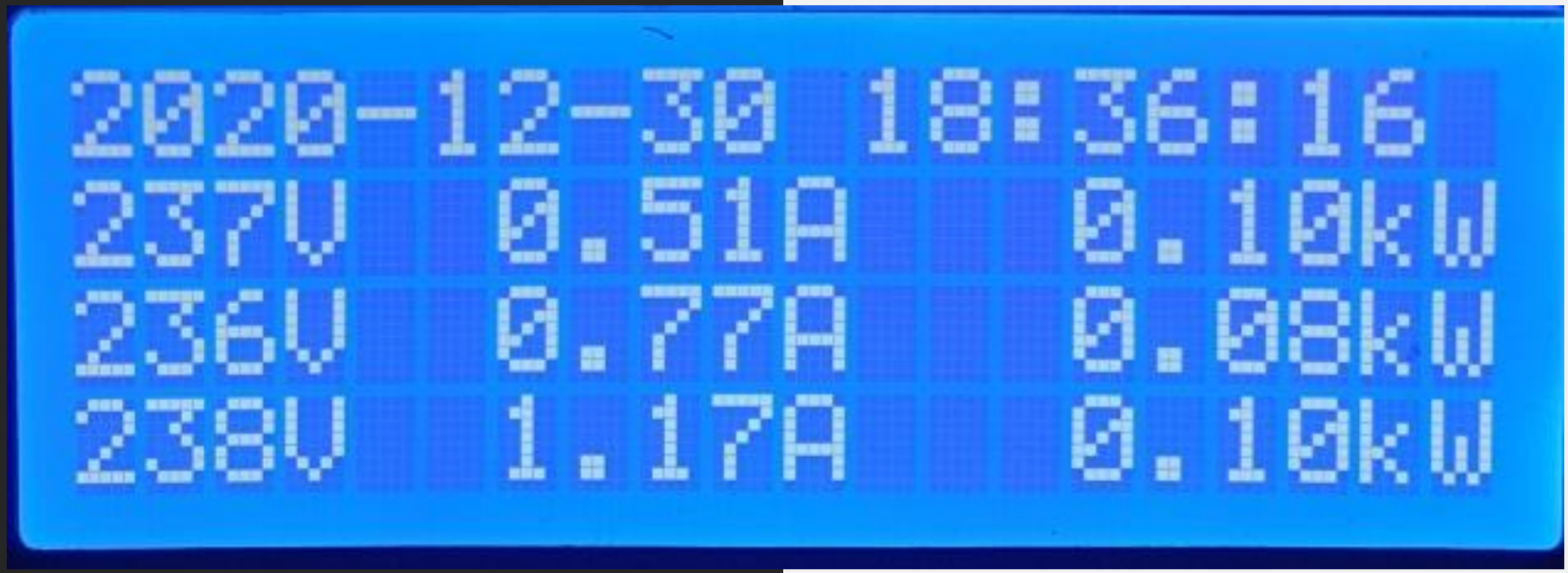
Séquence de 1 sec, déroulement du processus.

4 lectures par minute:
00 – 15 – 30 - 45

Etat	ask	rec	calc	store	stored	wait
T (sec.)	0	1..4	5	6	6	7..
Proc			± × ≥			



Menu 1: Résumé



Menu 2: details phases

Premier enregistrement – V016

Fichier 2012engy.txt

20/12/30	14:00:06	4363.189	14646.530
20/12/30	14:15:06	4363.189	14646.937
20/12/30	14:30:06	4363.189	14647.124
20/12/30	14:45:06	4363.189	14647.413
20/12/30	15:00:06	4363.189	14647.492
20/12/30	15:15:06	4363.189	14647.500
20/12/30	15:30:06	4363.189	14647.516
20/12/30	15:45:06	4363.206	14647.516
20/12/30	16:00:06	4363.238	14647.516
20/12/30	16:15:06	4363.284	14647.516
20/12/30	16:30:06	4363.363	14647.516
20/12/30	16:45:06	4363.418	14647.516
20/12/30	17:00:06	4363.469	14647.516

Commentaires

- La date est YY/MM/JJ (au lieu de YYYY.MM.JJ)
- Il manque <TAB> entre les valeurs d'énergie
- L'enregistrement est à la seconde 6, de la fenêtre pour «store»
- Jusqu'à 15h30, production par les panneaux
- Depuis 15h30, consommation
- Les '0' non-significatifs sont remplacés par des espaces

Réduction 5

- Suppression de la RTC!
  - Le module de synchro NTP ne pouvait pas être activé…
- RTC remplacée par le NTP
  - Le temps Unix n'est dispo qu'après ~1'30'' du «power on»
  - Pas d'influence sur l'enregistrement du 1/4H
  - Convient pour mesures sur la durée
- Redesing des fonctions du module «time»

Résultat:

```
RAM:   58.2% (au lieu de 65.1%)
Flash: 93.3% (au lieu de 93.4%)
```

- Pilote instancié à l'initialisation ✓
  - Plus de new et delete

Complément V20

- Redesign des menus
- Energie en double, adaptation du code
- Traitement et menu «erreur»
- macro CYCLE(), ajout de CYCLE() aux point «lents»

Stack min: 208 (au lieu de 102)

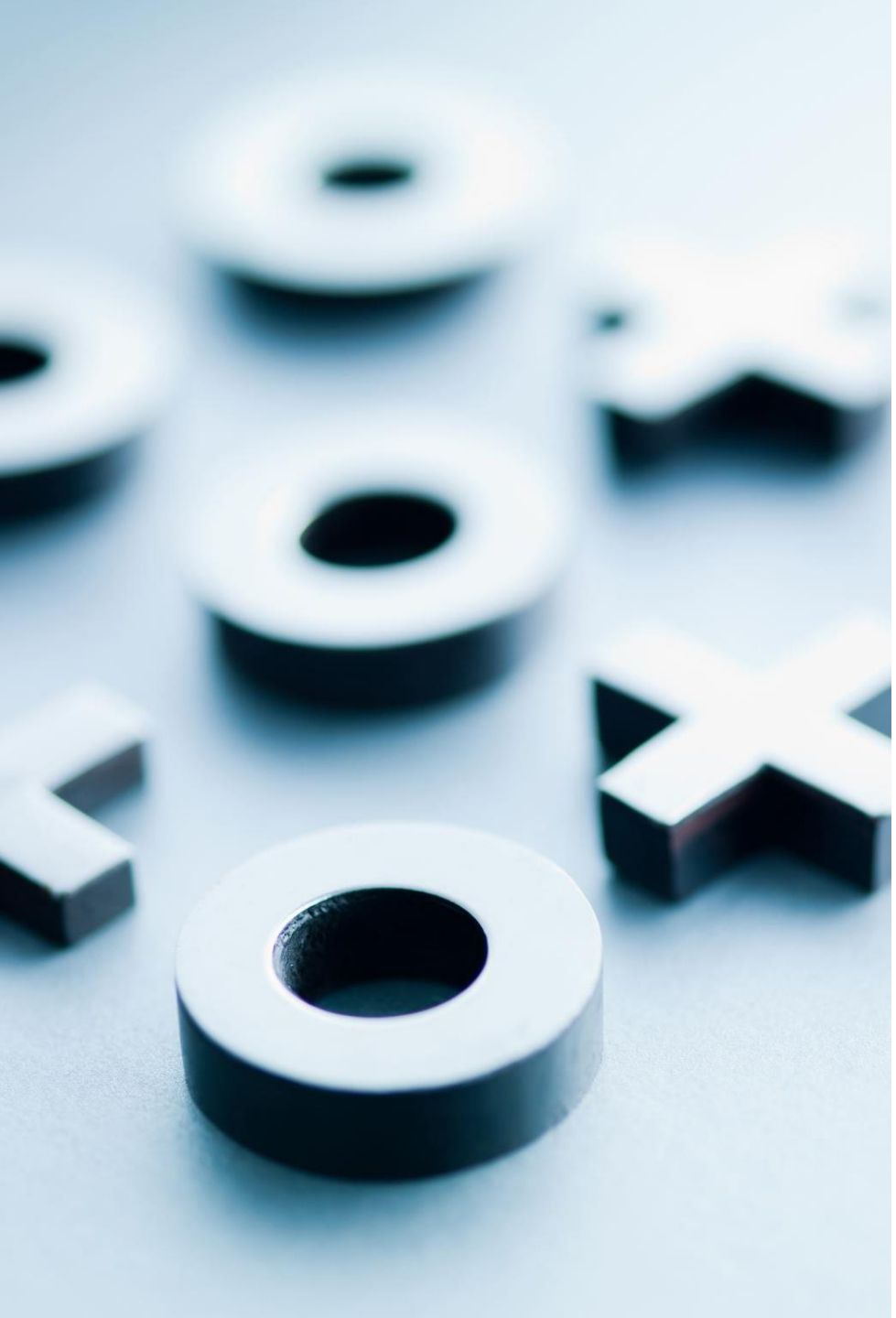
RAM: [======   ] 63.3% (used 1620 bytes from 2560 bytes)

Flash: [==========] 95.8% (used 27474 bytes from 28672 bytes)

Modification structure CALC pour les 2 valeurs de l'énergie

- Structure de base, signature valeur

- Structure dérivée: base et valeur double
- Utilisation de classe dérivée C++



Instabilité de la V020

- Enregistrement «pourri»
  - L'énergie ne change plus, ni IN, ni OUT ??
  - Timeout de lecture du compteur
  - Chrs bizarres dans la date!
  - Seconde d'enregistrement tardive (au lieu de +6 .. +7)

2021-01-12 05:00:20	4411.225	14697.550
2021-01-12 05:15:07	4411.225	14697.550
2021-01-12 05:15:34	Read timeout!	
2021-01-12 05:30:19	4411.225	14697.550
2021-01ûûÒ:	4411.225	14697.550

Réduction 6

- Augmenter le buffer e230_05 contamine l'horloge!!
  - Le cycle devient aléatoire
  - Données pourries
- Chasse aux octets RAM
- Modification de processus: pas de conversion si erreur de parité, on garde les dernières valeurs
- Idée, raccourcir les path d'enregistrement:
- `//const char flog[] = "/mnt/sd/arduino/www/e230.log";`
- `const char flog[] = "/m/e230.log";`

Grâce à un lien symbolique Unix!!

Analyse et résultats

Enregistrement stable

Valeurs correctes

Résultats, version V20

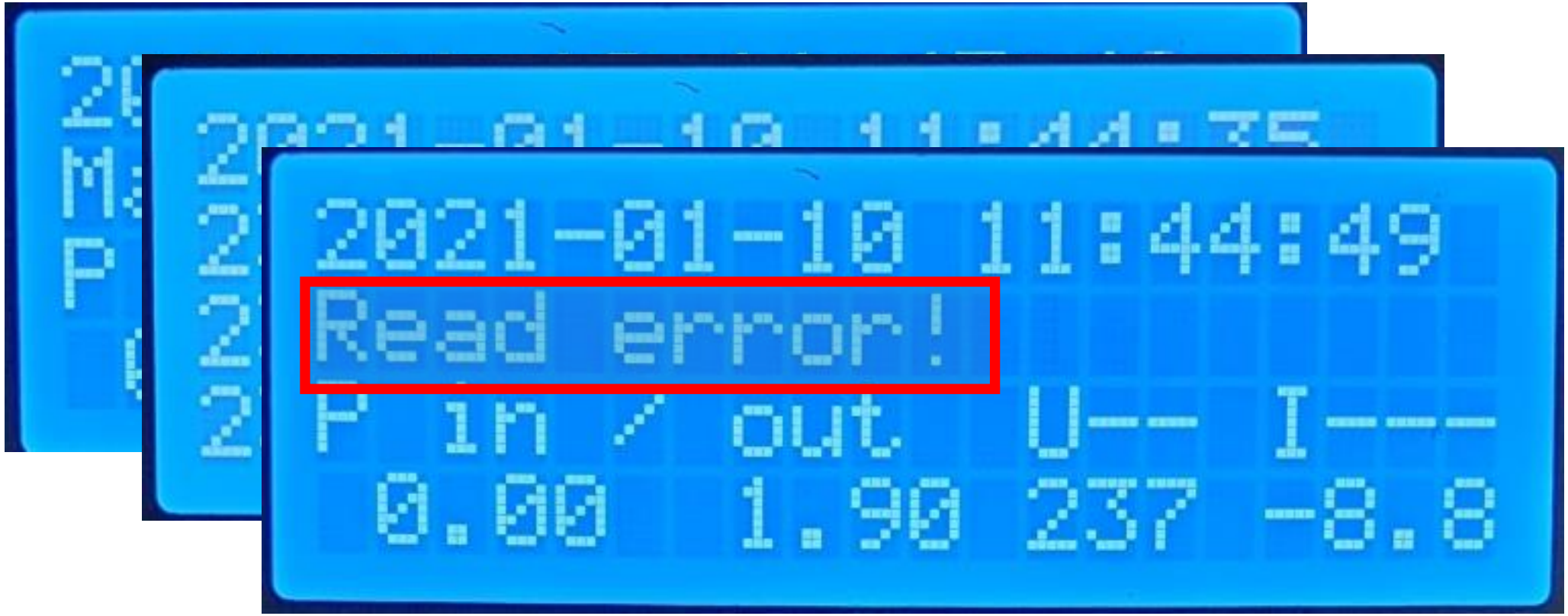
Mission	**Remarques**
Lecture ~	• 32% d'erreurs …
Enregistrement	• Risque de mauvaise lectures
Précision	• OK, avec énergie en double
Menus, affichage	• Informatif et suffisant
Economicité	• Yun utilisé à l'extrême!
Analyse, rendu	• Local OK, sur WEB OK avec PHP

Taux de lecture

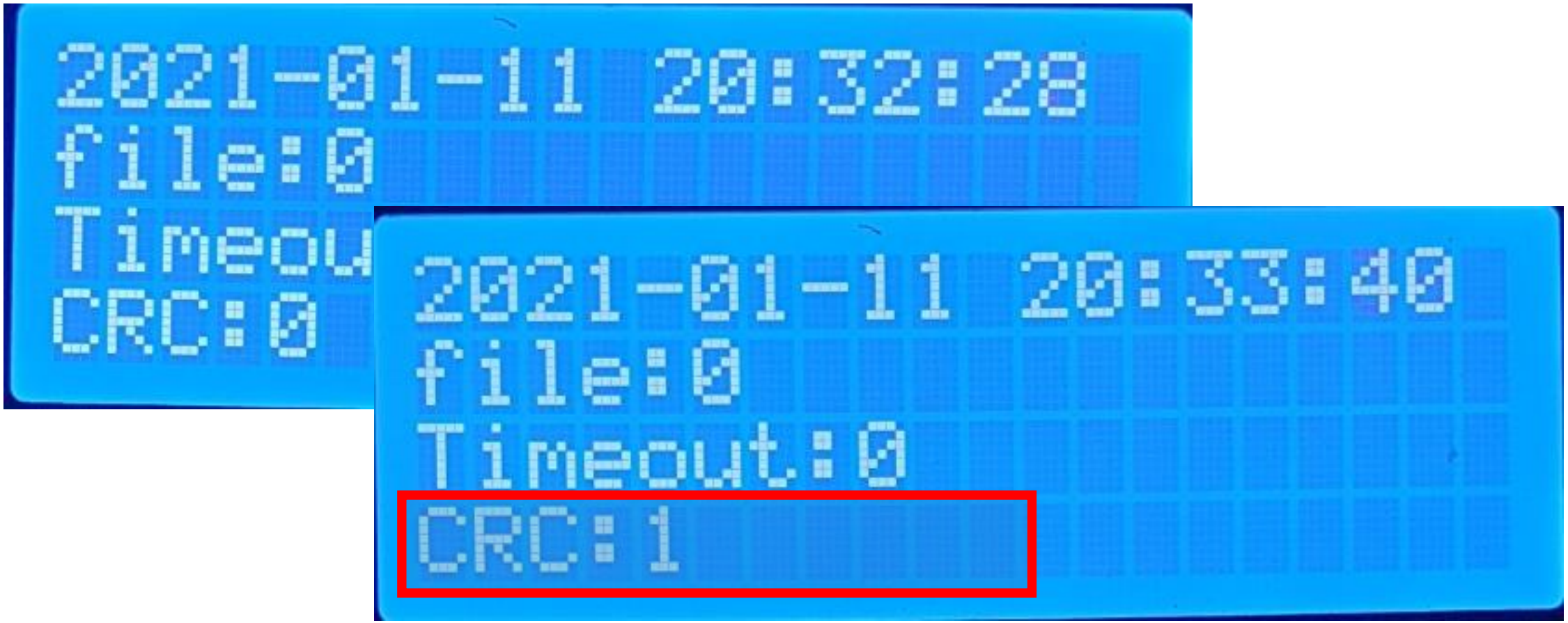
- Analyse log enregistré sur ~10 heures
  - 2708 Ask / 880 Read error (32.4%)
  - 0 Timeout
  - 0 Record pourri

Peut-on améliorer ce point?

Menus avec production solaire



Menu 3 – sans / avec erreur



Est-ce un produit fini?

Pas vraiment…

- Taux d’erreur avec le compteur trop important
- Problème de Serial USB, pour monitorer et/ou des commandes avec le PC

Réduire le taux d’erreur

Comment?

Chercher les fonctions consommatrices de temps bloquant

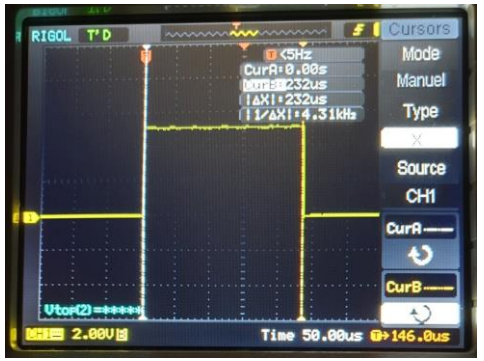
Contrôle de timing

Par le hardware

Méthode:

- utiliser une sortie libre X
- Enclencher X
- Appel routine()
- Déclencher X

Et mesurer avec le scope le délai.



Fonction	Durée min	…	max
getTimeStamp()	42 ms		150 ms
display_menu()	80 ms		
Serial.print()	86 us	232 us	1020 ms
Traitement ‘pstate’	38 us	180 uS	1.34 ms

Mesures de timing au scope

Modif du process

- Bypass l'appel de getTimeStamp() à Linux lors de transaction,
- Modif locale de la seconde


```
void poll_loop_1_s()
{
  if (!pstate)  // Q: Is Serial ask energy running?
  {             // A: no, get the Unix time
    getTimeStamp(dateTimeStr+2, sizeof(dateTimeStr)-3);
  }
  else          // the Unix time cannot be read for 6-7 seconds
  {             // make the seconds up to date
    char *ps = dateTimeStr + TIME_S;  // point seconds digit
    *ps == '9' ? *ps = '0' : ++*(ps);   // increment it, if < 9
  }
```

Normal…

Seconde traitée

Fiabilité de lecture du compteur

Sur 2311 lectures,

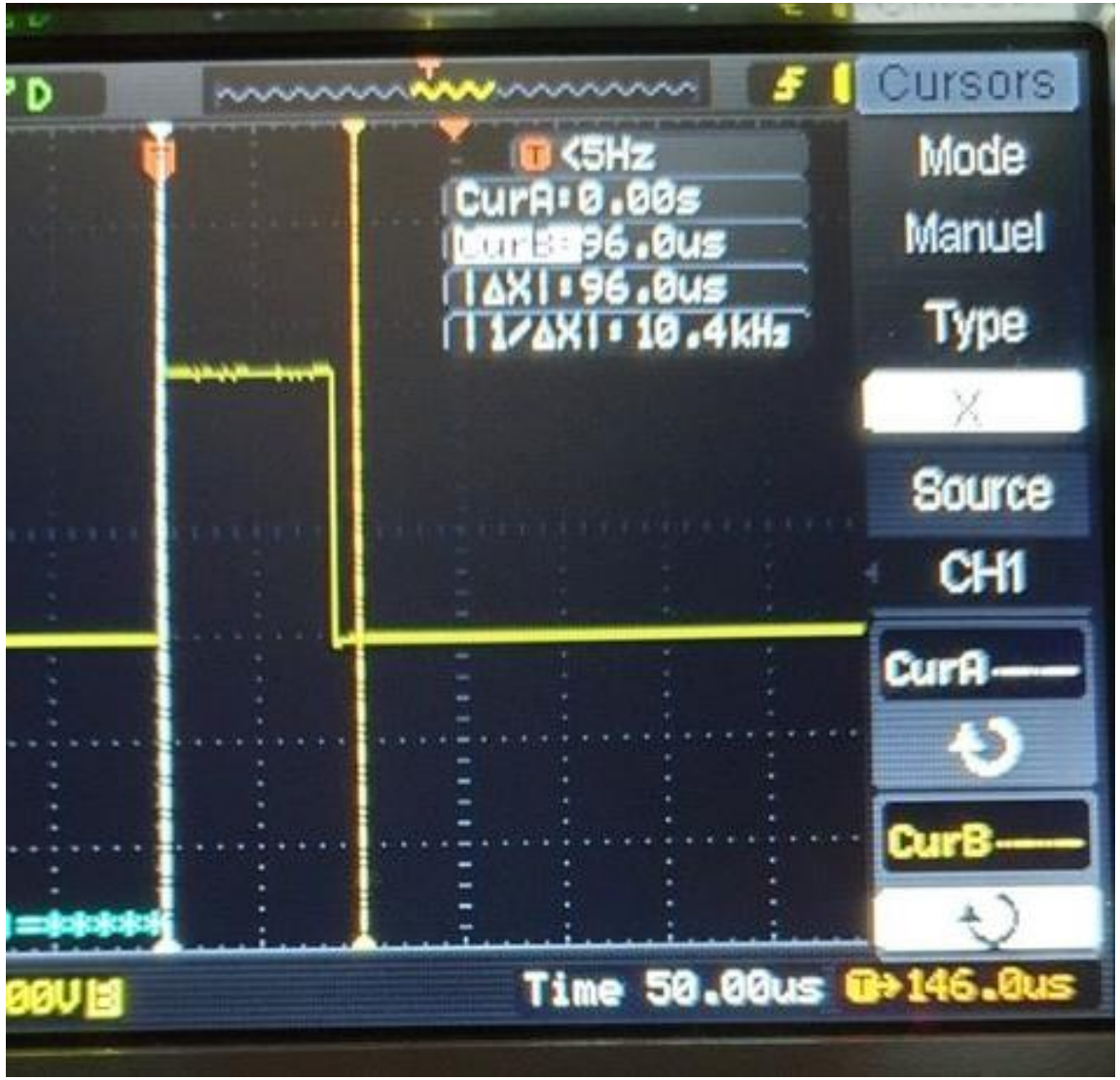
21 ‘read error’ → 0.9 % ✓

Reste le PB du serial.print()…

Résultats pour Serial.print()

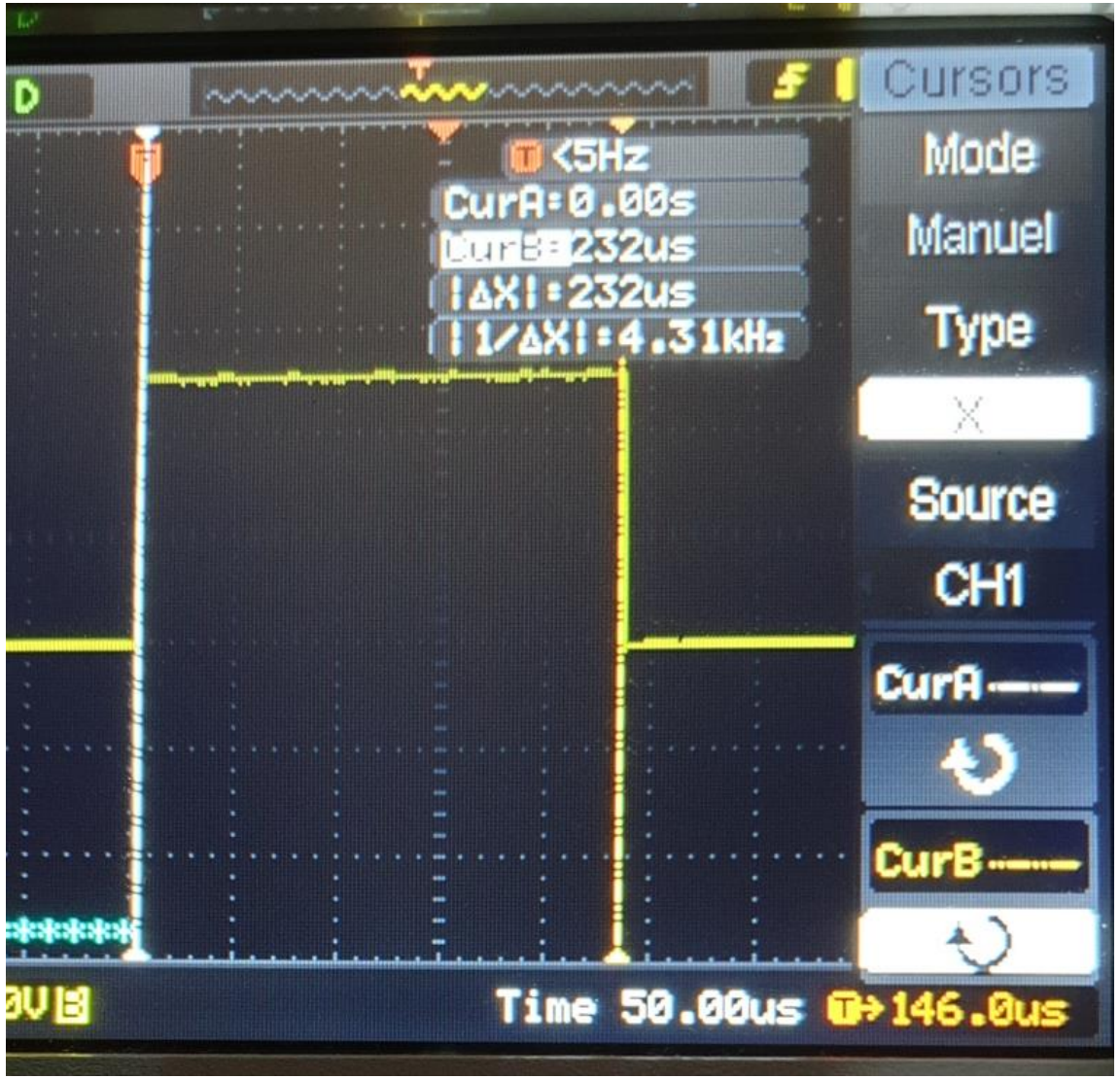
Serial, avec USB non connecté

(après un reset)



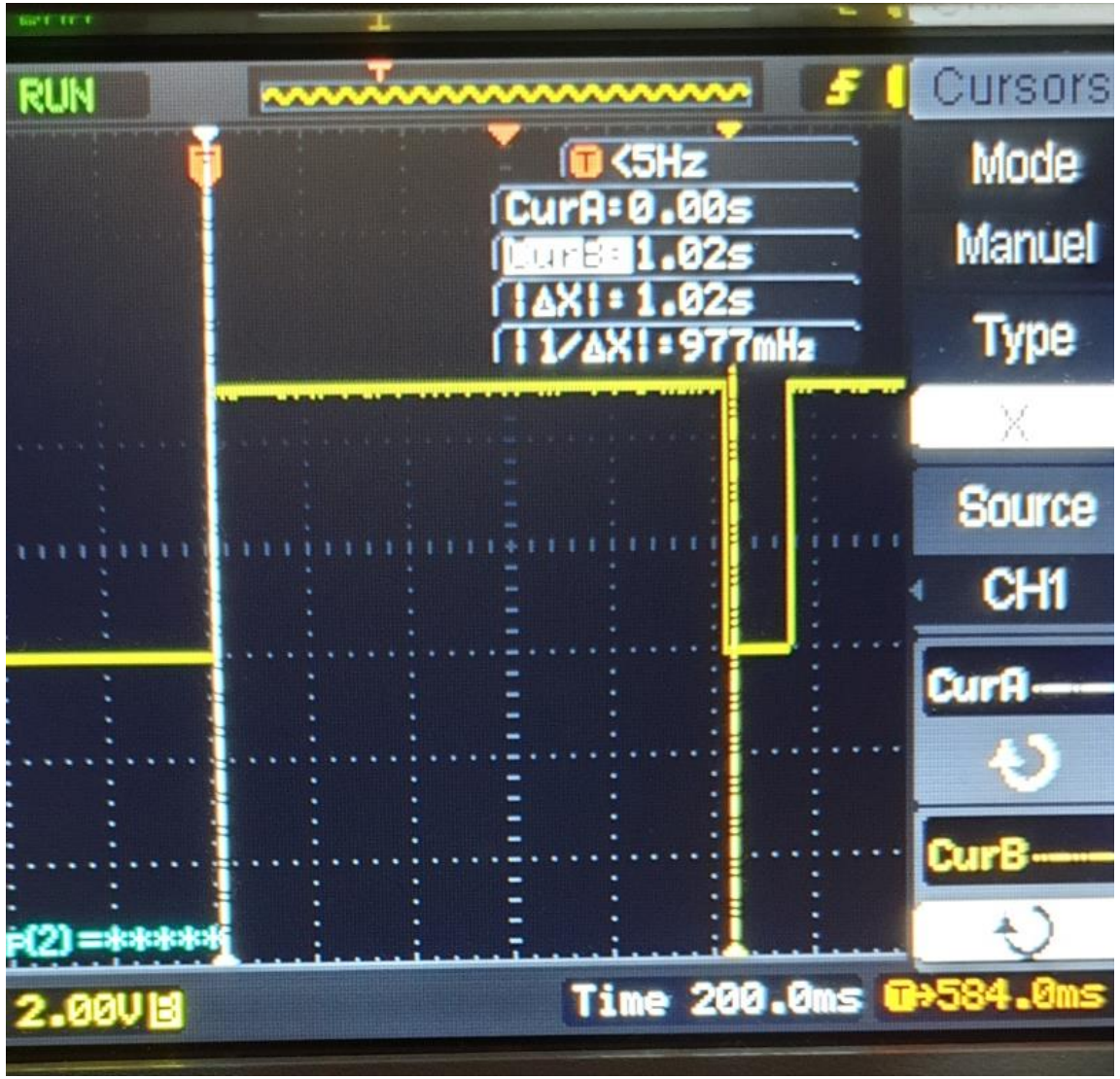
Résultats pour Serial.print()

Sérial, avec USB connecté et log actif



Serial USB déconnecté: problème!

- Ne se récupère qu’avec un reset.



Modif du process

- Ajouter un flag ‘print_log’
- Ajout d’une commande ‘l’
  - Toggle en log / pas de log
- Ajout d’une surveillance de timing

```
if (ms() > (currentMs+10)) // Q: is Serial gobing too long time?
  print_log = false;       // A: Yes, probably disconnected, stop log
```

Projet e230 - contenu

Energie – de quoi parle-t-on?

E230 – définition hardware

E230 – soft embarqué

Mise au point modules

Résultats

Site WEB et présentation

Analyse de consommation

Site WEB - présentation

- L’afficheur local montre les valeurs instantanées
- Un CRON pousse les données par ftp
- Le site WEB doit montrer :
  - L’historique de puissance, à 15 minutes près
  - La consommation / production d’énergie
  - Un rapport mensuel / journalier des valeurs intéressantes
- Les pages doivent être compatibles avec un écran de mobile

Site WEB - composants

PHP – codage de fonctions (plages, date, etc…)

Bootstrap – boutons

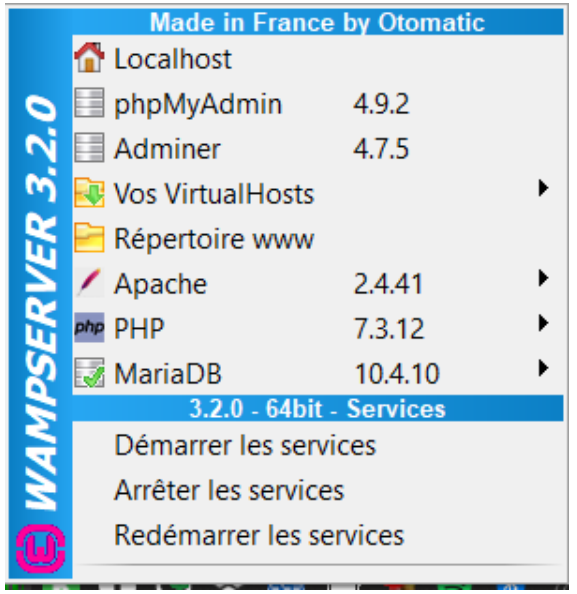
Chart.js – données en graphique

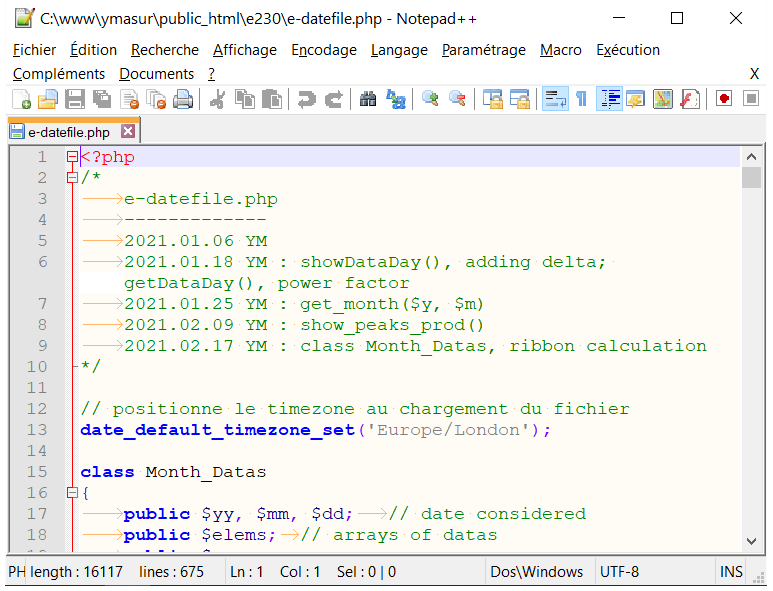
Site web:

https://e230.yvesmasur.ch

Site WEB - méthode

- Browser
- Wampserver
- Notepad++





Site WEB - fonctions

- Positionner la date (défaut: aujourd'hui)
- Lire le fichier du mois
- Données en tableau PHP
- Calcul de la puissance: P = $\frac{dE}{dT}$
- Calculs annexes: pointe, ruban

Projet e230 - contenu

Energie – de quoi parle-t-on?

E230 – définition hardware

E230 – soft embarqué

Mise au point modules

Résultats

Site WEB et présentation

Analyse de consommation

Affichage U – I – P

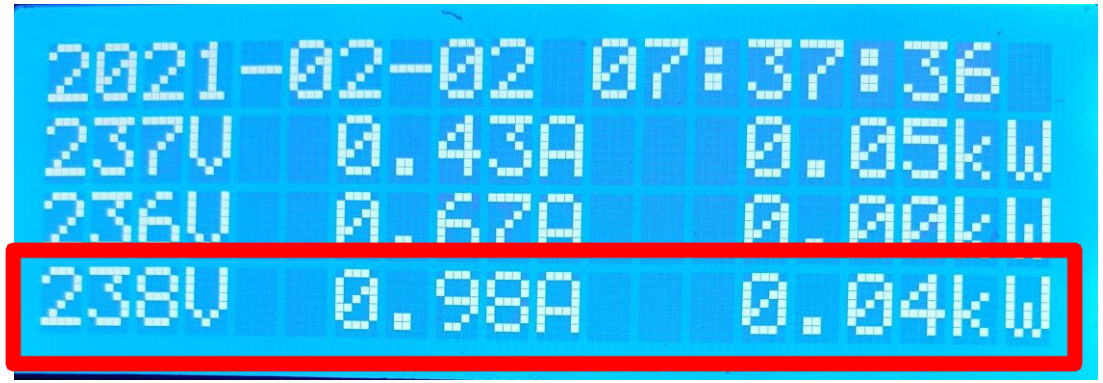
Energie au cours du jour

Quelques résultats et analyse

Puissance réactive

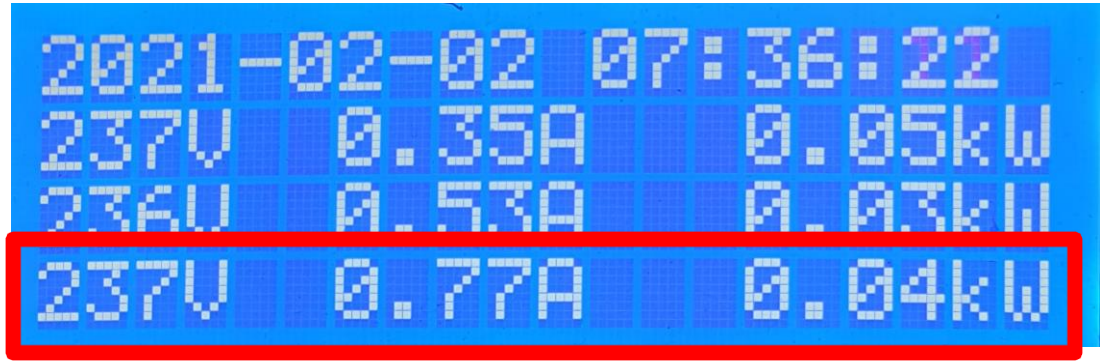
Matin, ph3 :

238[V] x 0.98[A] = 0.233 [kVA]



Influence PV?

237[V] x 0.77[A] = 0.182 [kVA]



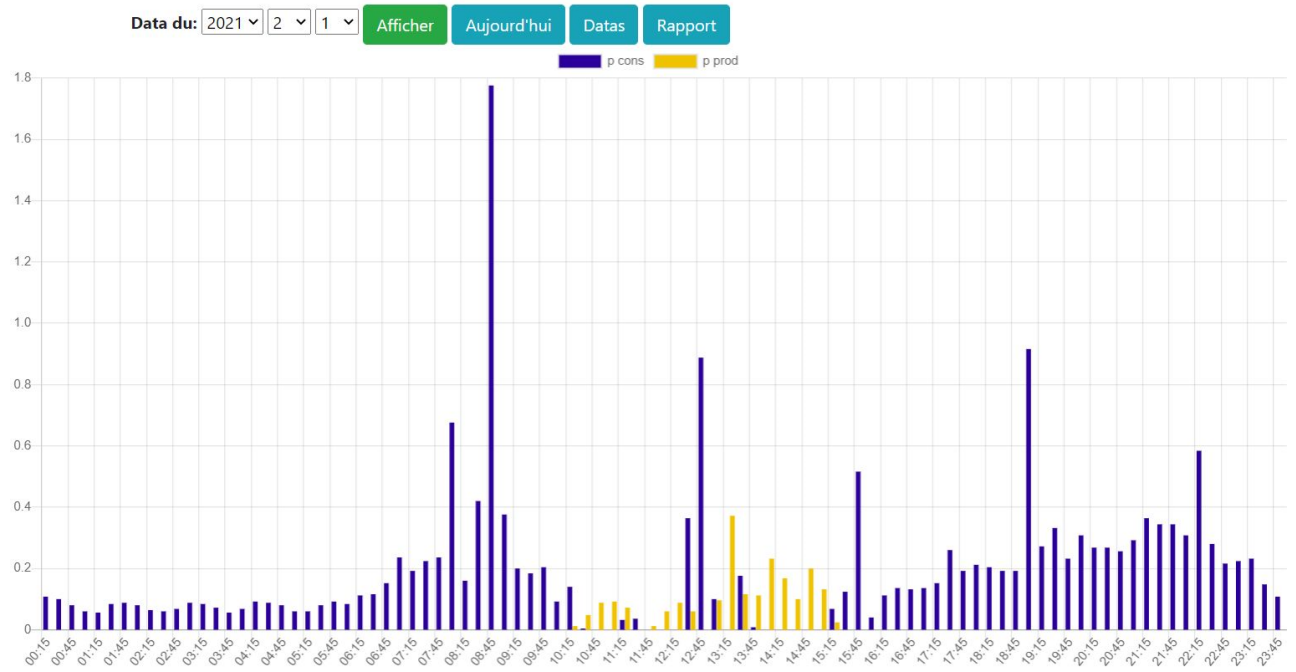
Sans PV, disjoncteur OFF

Puissance

Cuisinière, 2 plaques utilisées

234[V] x 14.37[A] = 3.36 [kVA]

P 15 @12H45 : 0.9 kWh

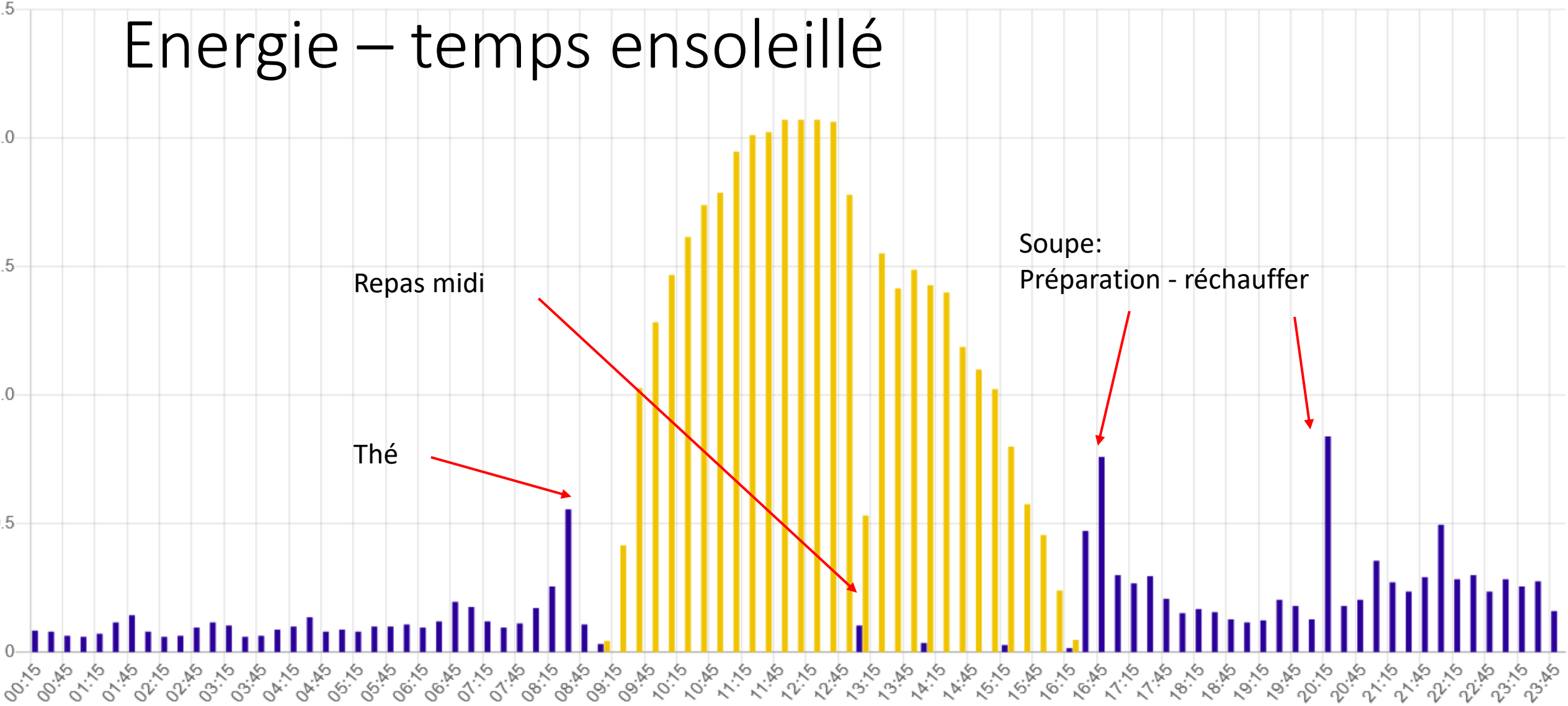


Puissance au 1/4 heure

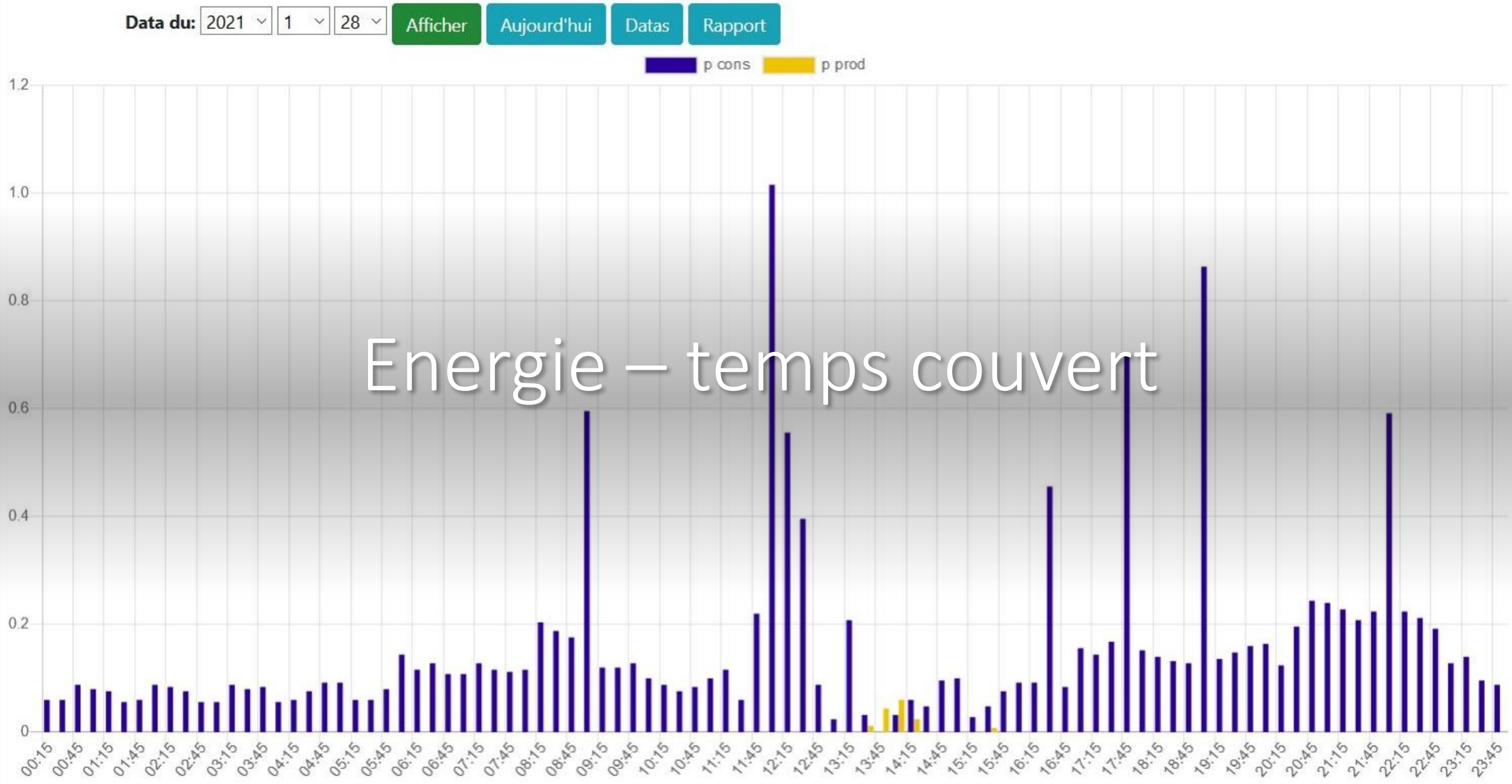
Data du: 2021 1 7 Afficher Aujourd'hui Datas Rapport

p cons p prod

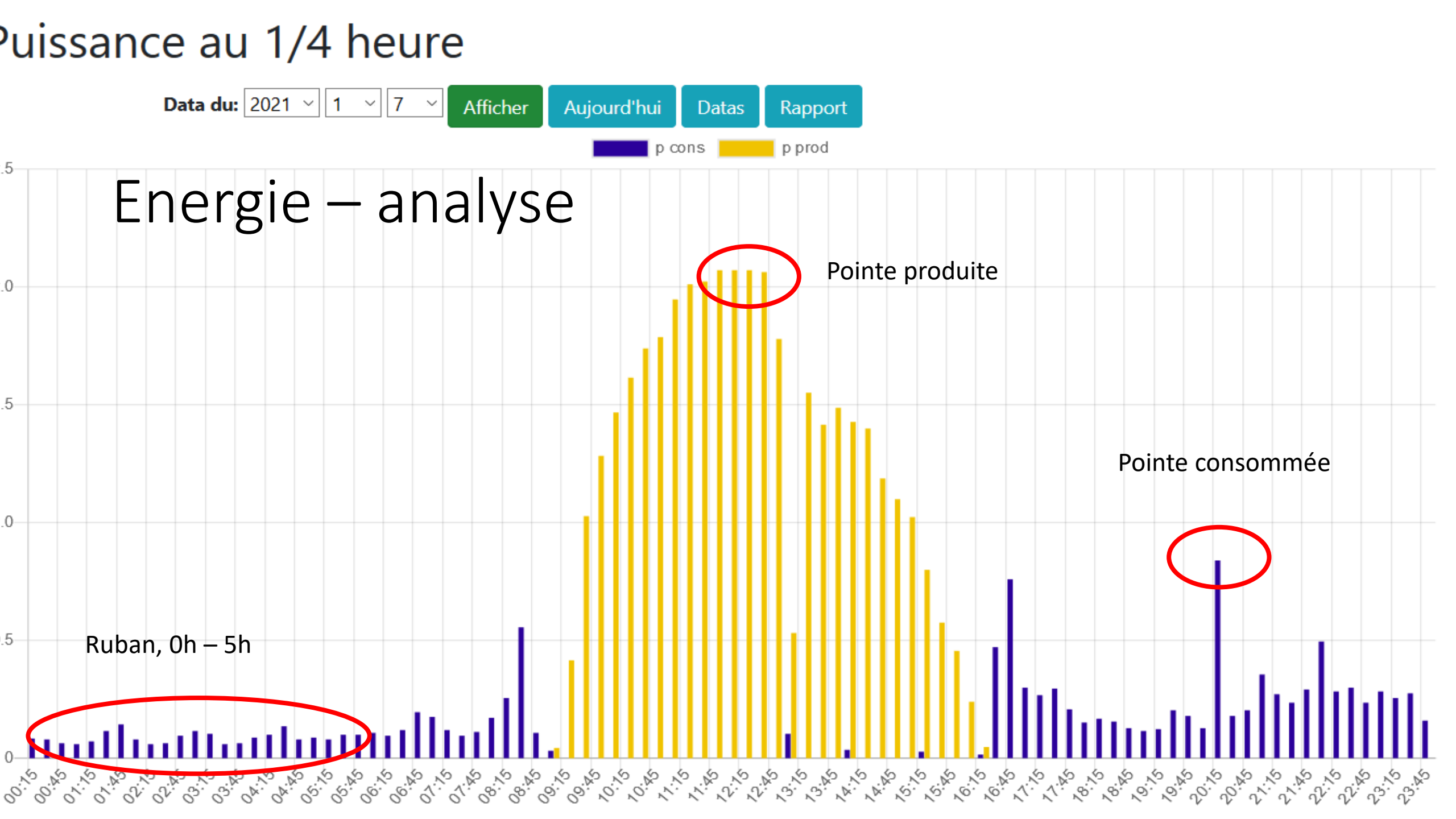
Energie – temps ensoleillé



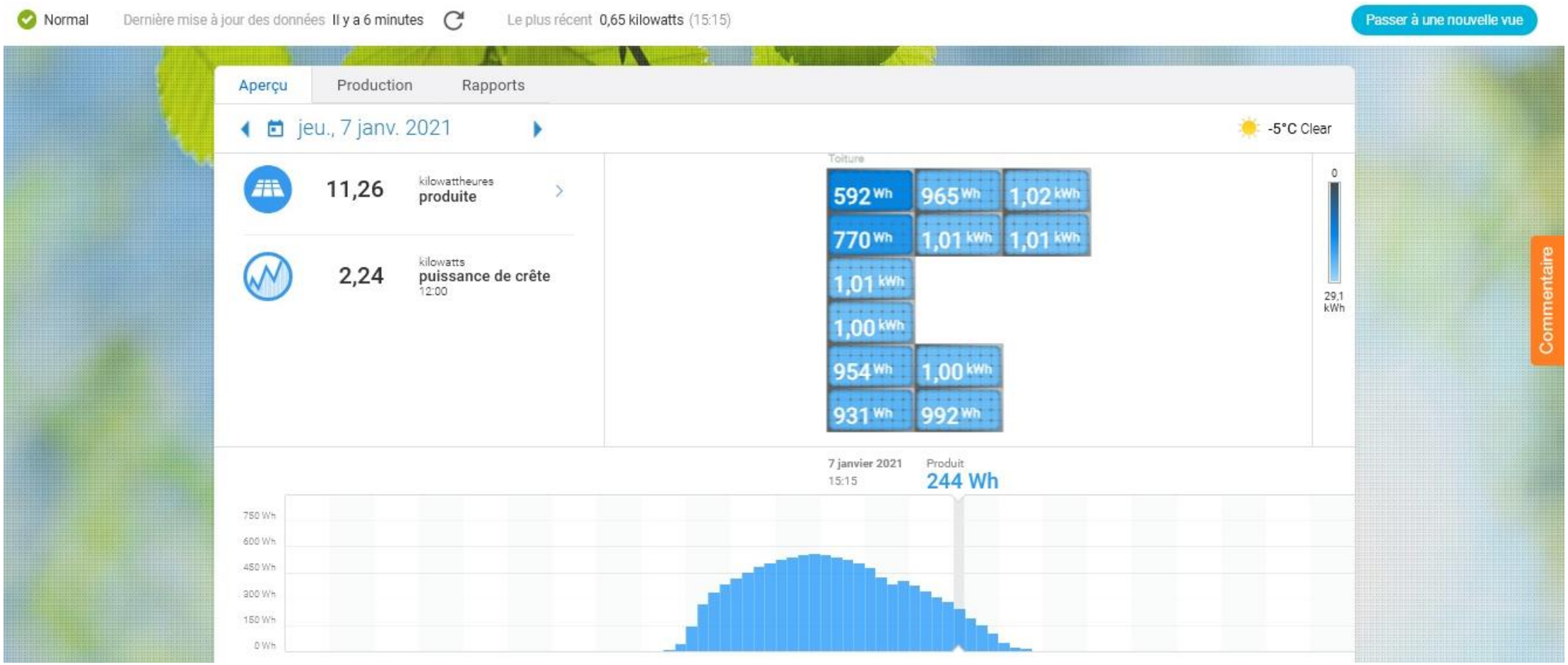
Puissance au 1/4 heure



Energie – temps couvert


Puissance au 1/4 heure
Data du: 2021 1 7
Afficher
Aujourd'hui
Datas
Rapport
p cons
p prod
Energie – analyse
Pointe produite
Pointe consommée
Ruban, 0h – 5h

Energie – selon PV



Rapport de Enlighten

Rapport mensuel de production d'énergie pour Famille Masur

Enphase Energy maximise la production de votre énergie solaire et vous tient informé au sujet de votre système. Votre rapport mensuel d'énergie indique la manière dont votre système s'est comporté et votre contribution à la compensation de l'empreinte carbone mondiale.

Semaine	Puissance de pointe	Énergie produite
01/01/2021 - 07/01/2021	2,28 kW	21,3 kWh
08/01/2021 - 14/01/2021	2,33 kW	45,2 kWh
15/01/2021 - 21/01/2021	1,41 kW	25,1 kWh
22/01/2021 - 28/01/2021	2,67 kW	23,7 kWh
29/01/2021 - 31/01/2021	1,18 kW	6,88 kWh
	janvier 2021 Total :	**122 kWh**

Rapport énergie

Datas du: 2021 1 7 Afficher Ce mois Datas Graphique

Consommée [kWh]	Produite [kWh]
122.865	086.532
Valeur 1.8.2	**Valeur 2.8.2**
004490.906	014736.596

Pic consommé [kW]	Pic produit [kW]
Valeur du mois	**Valeur du mois**
6.76 *Faux...*	3.46 *Faux...*
Valeur du jour (07)	**Valeur du jour (07)**
0.84	2.07

Puissance en ruban [kW]

Ruban mensuel : 0.085
Ruban du jour (07) : 0.088

Pour contrôler le stand-by

Rapport de e230

Améliorations possibles

- Rapport:
  - tenir compte des trous de données
  - Alerte en cas de manque
- Fiabilité
  - À contrôler – mise à jour Linux?

C’est fini!

Merci de votre intérêt