PREMESSA

I requisiti essenziali per poter utilizzare questo sistema sono:

- sistemi realizzati mediante l'immagine Solarjessie v2.6 e successive;
- MeterN versione 0.8.3 e successive
- Configurazione standard (primo meter libero n. 6)

Forum di discussione: Modulazione automatica resistenza boiler PWM

DESCRIZIONE E SCHEMI DI COLLEGAMENTO

Con lo stesso Raspberry su cui avete installato 123Solar e MeterN, lo utilizzeremo anche per fare una modulazione della resistenza del boiler mediante l'uscita PWM del Raspberry con un semplice relè SSR zero crossing (acquistato per pochi euro sulla baia).
In questo modo si riesce a fare una regolazione a treni di sinusoidi, regolando la potenza con 20 step (che, ad esempio, su 1200W di resistenza significa gradini incrementali da 60W).

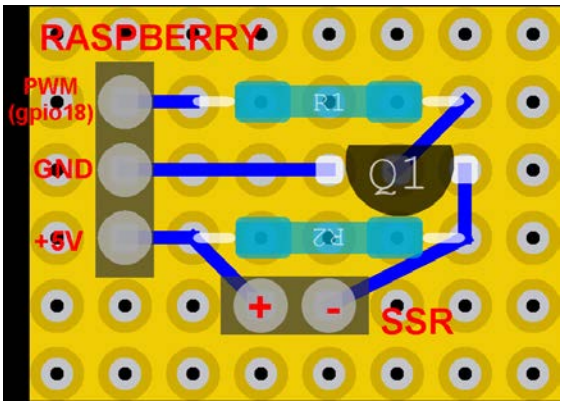
La scelta dei 200ms di periodo non è casuale ma deriva da una sperimentazione che ho fatto confrontando l'energia immessa rilevata dal contatore di scambio, in quanto utilizzando periodi più lunghi si rilevavano delle immissioni in rete che non dovevano esserci, per via dei "buchi" troppo lunghi della regolazione a treni d'onda.

Servono:

- Raspberry con 123solar e MeterN
- Un relè SSR zero crossing FOTEK SSR-25DA
- una resistenza R1=1Kohm 1/4W
- una resistenza R2=100Kohm 1/4W
- un transistor NPN tipo P2N2222A o BC238B
- una basetta millefori, saldatore e un minimo di buona volontà
- tocco finale ..... il mio script per la regolazione PWM

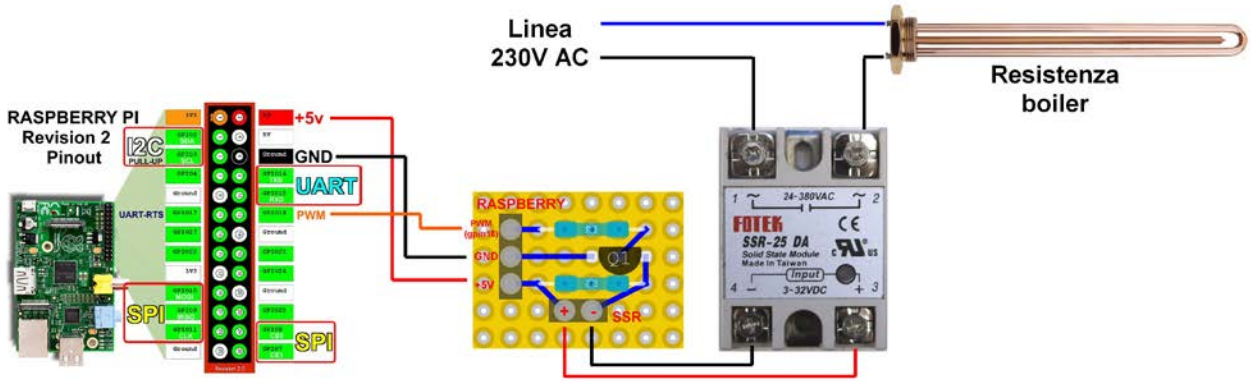
Il transistor e le due resistenze montate opportunamente su una basetta millefori, servono per convertire i 3,3V del raspi a 5V perchè ogni tanto l'SSR non rilevava lo stato logico 1 in ingresso in quanto i 3,3V sono al limite per attivare l'SSR e con attivazioni così veloci ogni tanto ne fallava qualcuna (da datasheet danno 4-32V per l'ingresso dell'SSR). In soldoni è un semplice transistor che lavora in saturazione facendo la semplice funzione di un interruttore ON-OFF.

Questo è lo schema del circuito montato sulla basetta:

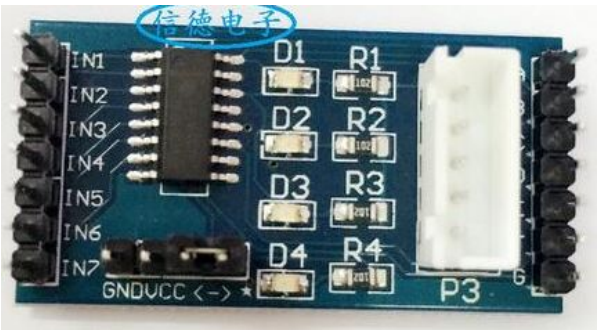


e questo il circuitino finito:

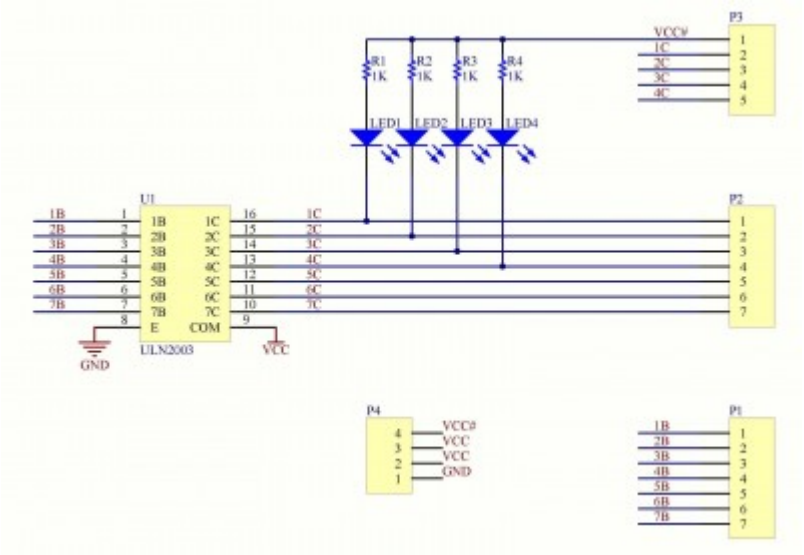
Il tutto andrà collegato come da schema seguente:



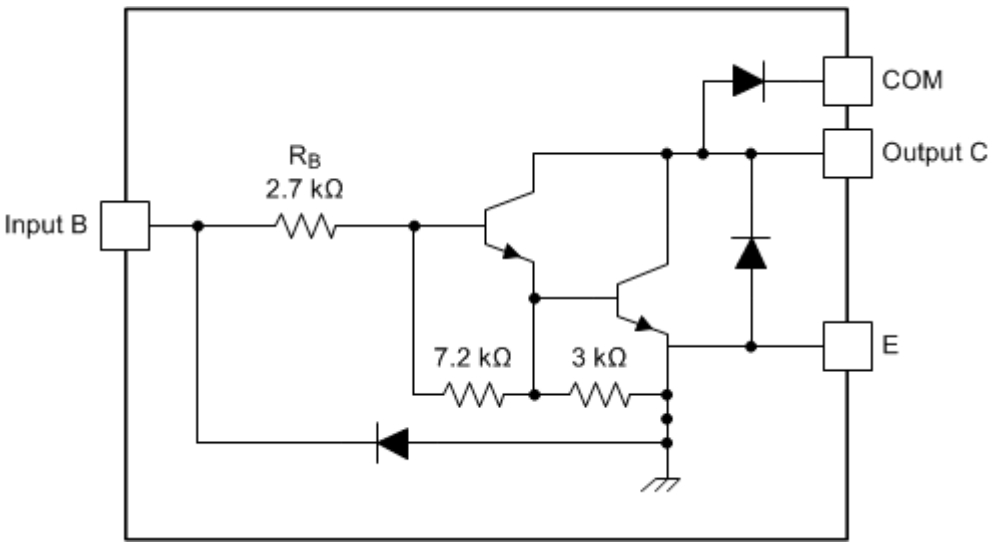
Per chi non se la cava bene con il saldatore, in sostituzione della schedina con il transistor appena vista è anche possibile utilizzare l'integrato ULN2003A, che è un array di 7 transistor darlington, che sopportano correnti fino a 0,5A e tensioni fino a 50V.
Potete prendere direttamente l'integrato e saldarvi i fili, oppure utilizzare queste schedine che si trovano facilmente ad a basso costo in rete:



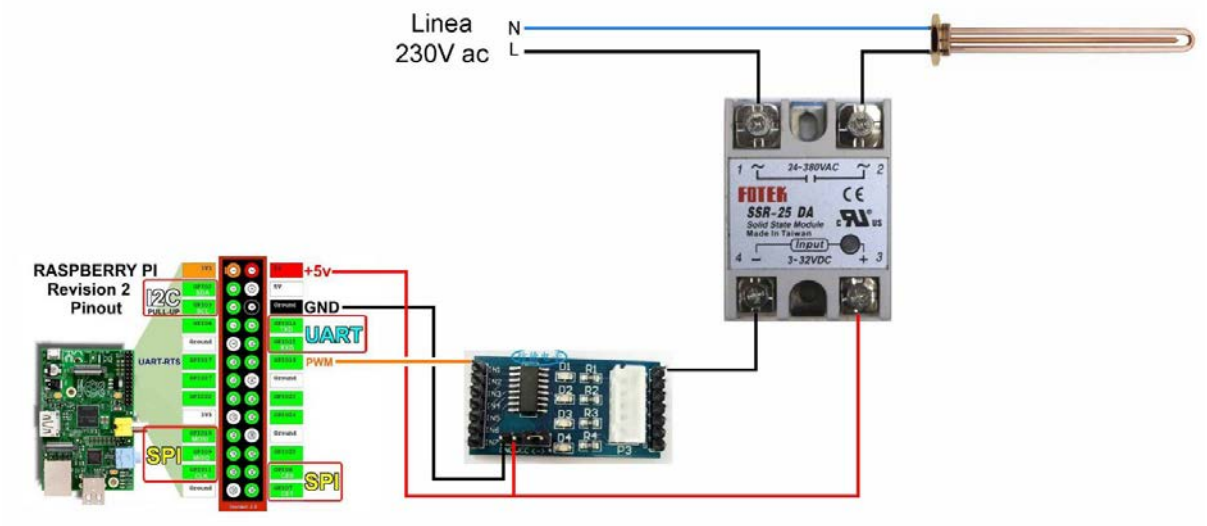
Questo è lo schema della schedina con l'ULN2003A



e questo lo schema interno di funzionamento dell'integrato ULN2003A:



Normalmente sono utilizzate per pilotare i motori passo passo, ma vanno benissimo anche al nostro scopo. Basterà collegare in entrata GND, +5V, ed il segnale PWM su uno dei sette ingressi disponibili, e portare all'SSR la relativa uscita e il +5V come da schema seguente:



Regolazione a treni d'onda

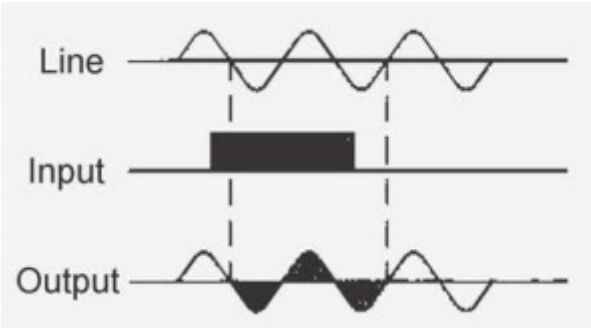
Il software invece provvede mediante l'uscita PWM del raspberry (GPIO18) ad effettuare una regolazione della resistenza a TRENI DI SEMIONDE (QUI spiegano come funziona questo metodo di regolazione)

La regolazione della potenza a treni d'onda presenta notevoli vantaggi.
Rispetto ai sistemi che parzializzano la sinusoide della tensione (sistemi a taglio di fase) presenta infatti i seguenti vantaggi:

- funzionamento del sistema a fattore di potenza (cos fi) unitario se il carico è resistivo;
- assenza di armoniche in rete;
- assenza di disturbi a radio frequenza reiettati in rete;
- l'SSR scalda molto poco (non servono alette per 1200-1500W regolati)

Il mio script PWM

Considerato che una semionda della tensione di rete a 50Hz dura 10ms si devono fare degli intervalli di accensione multipli di 10ms.
Infatti per poter lavorare su periodi PWM più brevi (200ms) ma avere un sufficiente numero di regolazioni (20step) la mia regolazione lavora sulla singola semionda e non su un'onda completa. Questo mi permette appunto di avere 20 step da 10ms su un periodo PWM di 200ms.
In questo modo ci si assicura di beccare un passaggio per lo zero della sinusoide e quindi l'SSR si accende, e così non serve la sincronizzazione dei comandi con la frequenza di rete (una complicazione circuitale in meno).
Di seguito è schematizzato il funzionamento di un relè SSR zero crossing, che si attiva e disattiva sempre e solo al passaggio per lo zero della tensione.



Quindi usando un periodo PWM di 200ms si possono fare 20 intervalli di accensione (20 gradini di regolazione della resistenza):

- accendi per 10ms (1 semionda) e spegni per 190ms (19 semionde) e la potenza è 1/20 (5%) P resistenza
- accendi per 20ms (2 semionde) e spegni per 180ms (18 semionde) e la potenza è 2/20 (10%) P resistenza
- accendi per 100ms (10 semionde) e spegni per 100ms (10 semionde) e la potenza è 10/20 = 1/2 (50%)P resistenza
- accendi per 150ms (15 semionde) e spegni per 50ms (5 semionde) e la potenza è 15/20 = 3/4 (75%) P resistenza
- accendi per 200ms (20 semionde) e spegni per 0ms (0 semionde) e la potenza è 100% P resistenza

Con il PWM la cosa è semplice, solo che tutti i parametri del PWM devono essere impostati correttamente:

- modo funzionamento PWM: Mark Space (di default il raspberry è impostato in modalità "Balanced")
- impostare correttamente PWM range e PWM clock ed utilizzare il corretto multiplo del duty cycle, per far si di avere degli intervalli multipli di 10 ms

Di seguito le impostazioni per avere una regolazione su 200ms da 20 step di 10ms ciascuno

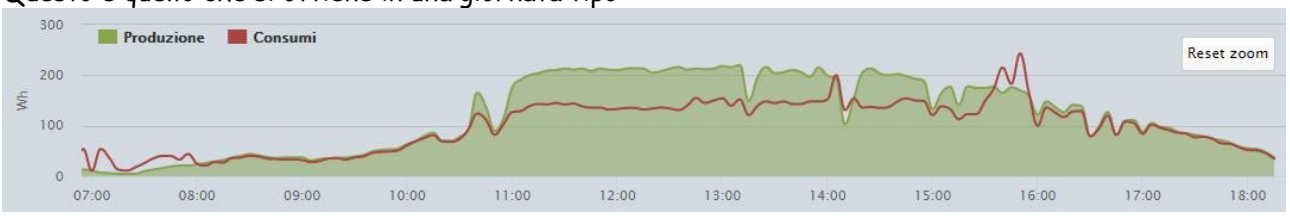
Range	**1600**	
Clock	**2400**	
Freq PWM	5.0000	Hz
T	0.200	s
impulso	0.000125	s
Min step duty cycle	80.00	
N. Step	**20.00**	
Potenza carico	1200	W
Regolazione minima	60	W

%	step	Duty cycle	Tempo ON (s)	N. semionde	Potenza regolata (W)
0	0	0	0.0000	0.000	0
5	1	80	0.0100	1.000	60
10	2	160	0.0200	2.000	120
15	3	240	0.0300	3.000	180
20	4	320	0.0400	4.000	240
25	**5**	**400**	**0.0500**	**5.000**	**300**
30	6	480	0.0600	6.000	360
35	7	560	0.0700	7.000	420
40	8	640	0.0800	8.000	480
45	9	720	0.0900	9.000	540
50	**10**	**800**	**0.1000**	**10.000**	**600**
55	11	880	0.1100	11.000	660
60	12	960	0.1200	12.000	720
65	13	1040	0.1300	13.000	780
70	14	1120	0.1400	14.000	840
75	**15**	**1200**	**0.1500**	**15.000**	**900**
80	16	1280	0.1600	16.000	960
85	17	1360	0.1700	17.000	1020
90	18	1440	0.1800	18.000	1080
95	19	1520	0.1900	19.000	1140
100	**20**	**1600**	**0.2000**	**20.000**	**1200**

dove il "duty cycle" è il valore che si invia all'uscita PWM.

Risultato finale

Questo è quello che si ottiene in una giornata tipo:



Ovviamente quando la produzione e alta, la resistenza da 1200W non è sufficiente per mangiarsi tutta la produzione, ma comunque basta ed avanza ☺

INSTALLAZIONE SCRIPT PWM

```
cd /var/www/MyScripts
mkdir PWM
```

```
cd PWM
git clone https://github.com/flanesi/Raspberry-PWM-Water-heater.git /var/www/MyScripts/PWM
sudo chmod a+x *
```

ed infine creiamo un symlink per poterlo avviare o stoppare in maniera più semplice:

```
sudo ln -s /var/www/MyScripts/PWM/pwm_ssr_dimmer.sh /usr/bin/pwm
```

a questo punto inseriamo in crontab l'esecuzione automatica all'ora desiderata (alle 6.45 del mattino)

```
sudo -s
cd /etc
nano crontab
```

aggiungere come ultima riga:

```
45 6 * * * root /var/www/MyScripts/PWM/pwm_ssr_dimmer.sh start
```

ctrl+O per salvare e ctrl+X per uscire

Lo script è poi impostato per fermarsi automaticamente dopo 20 cicli con produzione = 0 (dopo le ore 16.00 cioè alla sera in assenza di sole).

AVVIO e STOP MANUALE

Se comunque volete avviare manualmente lo script basterà digitare da putty:

```
pwm start
```

Invece per fermarlo:

```
pwm stop
```

TEST e CONFIGURAZIONE SCRIPT

Ora non dovete fare altro che editare il file pwm_ssr_dimmer.php solo per inserire il valore in W della vostra resistenza.
Sicuramente conoscerete il valore nominale in Watt (W) della vostra resistenza, ma è importante conoscerlo nel modo più preciso possibile.

Lo script infatti, per evitare di dover utilizzare un ulteriore contatore solo per la resistenza, basa il suo funzionamento sulla conoscenza del preciso valore in W della resistenza che deve comandare.

Per poterlo ottenere in modo semplice ho predisposto un semplice script (pwm_test.php).

Dovete eseguire questo test possibilmente in queste condizioni:

- in un momento della giornata che il vostro impianto fotovoltaico stà producendo (questo per via delle lievi variazioni della tensione)
- durante il test non dovete possibilmente avere grosse variazioni di carichi elettrici in casa

Il funzionamento è semplice:

- lo script misura i vostri consumi domestici con resistenza spenta
- accende la resistenza al 100% per 10 secondi, rileva nuovamente il consumo totale e fa la differenza con il precedente valore
- ripete l'operazione per 3 volte
- restituisce per ogni test i valori rilevati ed infine fa un media dei tre valori ricavati

Per un risultato ottimale è quindi importante che durante il test i carichi elettrici di casa rimangano pressoché costanti.

Per eseguire il test:

```
cd /var/www/MyScripts/PWM
php pwm_test.php
```

Dovrebbe restituire qualcosa del genere:

```
Start TEST .... attendere....

Consumo OFF 1 :        272.00 W
Consumo ON  1 :        1513.70 W
Pot. Resistenza 1:     1241.7 W

Consumo OFF 2 :        274.70 W
Consumo ON  2 :        1492.90 W
Pot. Resistenza 2:     1218.2 W

Consumo OFF 3 :        269.90 W
Consumo ON  3 :        1501.40 W
Pot. Resistenza 3:     1231.5 W

POT. RESISTENZA MEDIA   1230 W
```

Se nei vari test viene rilevata una grossa differenza fra i 3 valori, è necessario ripetere il test.

Editate ora il file pwm_ssr_dimmer.php ed inserite il valore rilevato

```
cd /var/www/MyScripts/PWM
nano pwm_ssr_dimmer.php
```

scorrete il file fino alla riga

```
$resistenza=1200;
```

inserite il valore rilevato e ctrl+O per salvare e ctrl+X per uscire

MODIFICA FILE reqsdm.php

Editate ora il file /var/www/comapps/reqsdm.php in modo che risulti come il seguente (in rosso le modifiche):

```
#!/usr/bin/php
<?php
// This script will output a meterN compatible format for the main or live command
```

```
// You'll need to setup correct permission chmod +x
// then ln -s /var/www/comapps/reqsdm.php /usr/bin/reqsdm
// Request command with 'reqsdm tensione' or 'reqsdm corrente' or .....

if (isset($_SERVER['REMOTE_ADDR'])) {
   die('Direct access not permitted');
}
if (!isset($argv[1])) {
    die("Abording: no valid argument given.\n");
                } elseif ($argv[1] == 'tensione') {
                        $outstr =  exec('cat /dev/shm/metern2.txt | egrep "^2_1\(" | grep "*V)"');
                } elseif ($argv[1] == 'corrente') {
                        $outstr =  exec('cat /dev/shm/metern2.txt | egrep "^2_2\(" | grep "*A)"');
                } elseif ($argv[1] == 'freq') {
                        $outstr =  exec('cat /dev/shm/metern2.txt | egrep "^2_3\(" | grep "*Hz)"');
                } elseif ($argv[1] == 'cospi') {
                        $outstr =  exec('cat /dev/shm/metern2.txt | egrep "^2_4\(" | grep "*F)"');
                } elseif ($argv[1] == 'boiler-status') {
                        $outstr =  exec('echo "pwm_ssr_dimmer("`pgrep -c pwm_ssr_dimmer`"*X)"');
                } elseif ($argv[1] == 'boiler-live') {
                        $outstr =  exec('cat /dev/shm/boiler6.txt | egrep "^6\(" | grep "*W)"');
                } elseif ($argv[1] == 'boiler-main') {
                        $outstr =  exec('cat /dev/shm/boiler6.txt | egrep "^6\(" | grep "*Wh)"');
                } elseif ($argv[1] == 'resistenza') {
                        $outstr =  exec('cat /dev/shm/boiler6.txt | egrep "^6_1\(" | grep "*%)"');
                } elseif ($argv[1] == 'cpu-temp') {
                        $outstr =  exec('cat /sys/class/thermal/thermal_zone0/temp');
                        $outstr = $outstr/1000;
                        $outstr = "cpu($outstr*°C)";
                // and so on ....
    } else {
    die("Usage: reqsdm (tensione|corrente|freq|cospi|boiler-status|boiler-live|boiler-main|resistenza|cpu-temp)\n");
    }
echo "$outstr";
?>
```

IMPOSTARE IL METER (ENERGIA e POTENZA BOILER)

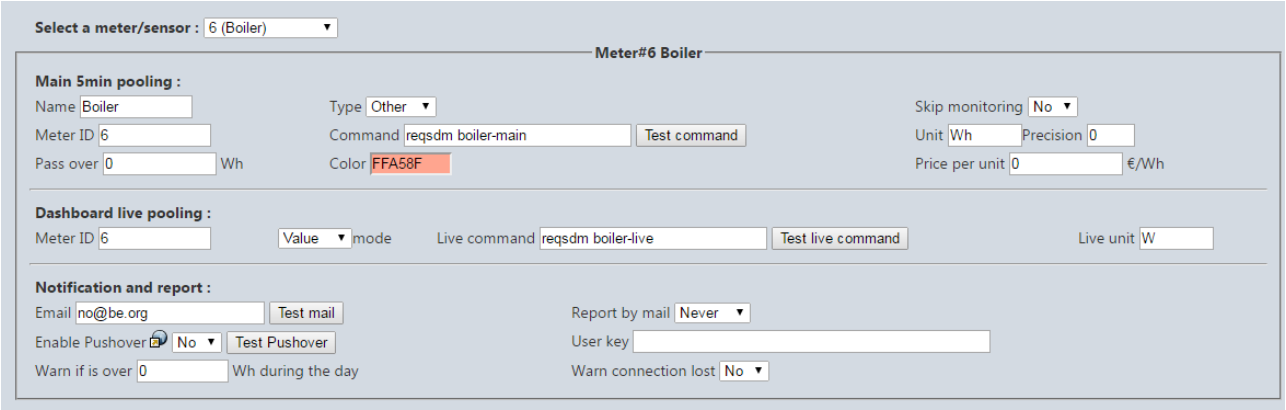
Lo script genera durante il suo funzionamento un file nella cartella temporanea /dev/shm/boiler6.txt simile a quello per i contatori, in cui vengono riportati:

- il valore dell'energia in Wh indirizzata al boiler (è un contatore continuo)
- il valore dalla potenza in W regolata della resistenza
- la % di regolazione della potenza della resistenza

Il valore dell'energia viene stimato con un semplice calcolo e non misurato, ma vi assicuro che risulta molto affidabile (provato confrontando il valore calcolato con quello misurato mediante un contatore specifico per il boiler ed il valore a fine giornata differisce di pochi Wh).
E' però possibile che la stima non sia precisa nel caso che nel corso della giornata la tensione di rete subisca delle variazioni importanti rispetto al valore nominale di 230V.

E' quindi possibile inserire in MeterN i seguenti meters ed indicators:

Energia Boiler

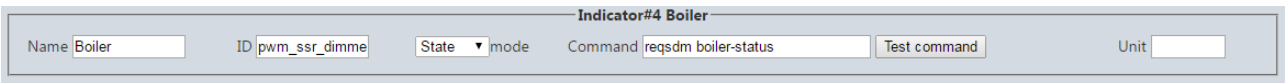


Main pooling:
ID: 6
command: reqsdm boiler-main

Live pooling
ID: 6
Mode: Value
Command: reqsdm boiler-live

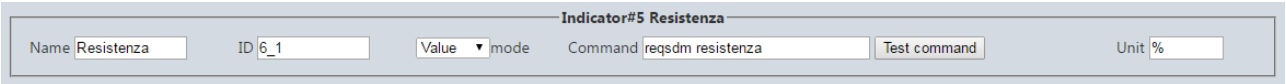
IMPOSTARE GLI INDICATORS

Indicatore dello script pwm attivo (ON/OFF)



ID: pwm_ssr_dimmer
Mode: State
Command: reqsdm boiler-status

Indicatore della % di regolazione della resistenza



ID: 6_1
Mode: Value
Command: reqsdm resistenza