Progetto di Stazioni Meteo integrate

MeteoICS

L'obiettivo del progetto è quello di associare ad ogni nodo Ninux una piccola stazione meteorologica con la quale rilevare i valori di Temperatura, Umidità dell'aria, Pressione Atmosferica , Direzione e Velocità del Vento, Irraggiamento solare e tutto quello che può rappresentare un monitoraggio ambientale.

Inoltre ogni stazione deve fornire dati statici quali la Posizione Geografica e Altitudine.
I dati rilevati dalla stazione Meteorologica sono raggiungibili attraverso la rete Ninux.
Il rilevamento di questi valori nello stesso momento in più punti della città e anche in città diverse, qualora le isole Ninux siano tra loro collegate (mediante VPN), può costituire una base di dati sulla quale realizzare modelli matematici per lo studio e la previsione dei fenomeni meteorologici.
Ogni stazione condivide con uno dei server dislocati sulla rete i dati rilevati.
La stazione meteorologica deve avere la possibilità di memorizzazione dei dati per un periodo due , tre volte superiore alla tempo di campionamento adottato dai server.

L'accesso a questi server è libero solo dall'interno della rete Ninux (oppure libero e basta ,si può decidere) mediante dei servizi messi a disposizione sui server stessi; ad esempio:

- Pagine Web per la rappresentazione dei dati
- Accesso ai dati attraverso diverse chiavi di ricerca
- Aggregazione di dati secondo criteri prestabiliti e selezionabili
- Integrazione dei dati (in tempo reale) nella pagina Ninux Map attraverso estensione della tabella associata ad ogni nodo.

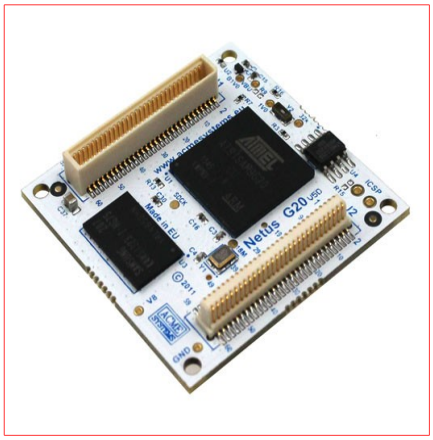
La connessione alla rete è realizzabile attraverso il router (OpenWRT o altro) associato alla rete Ninux.
La soluzione proposta è basata su un hardware piuttosto costoso e poco conosciuto, ma quando ho iniziato a sviluppare il progetto della stazione la Raspberry non era ancora disponibile.
L'adattamento sia dell'hardware che del software è agevolato dall'utilizzo di strumenti standard che

possono essere replicati in un ambiente Linux Debian Embedded. (Raspdebian per esempio)

Lo scopo di questo documento è quello di descrivere i dettagli realizzativi della stazione meteorologica riguardanti le scelte di sistema (Sensoristica,Architettura Software, protocolli di comunicazione, Software OpenSource...) e i metodi adottati per la gestione dei dati rilevati e per l'integrazione insieme ad altre stazioni analoghe. Nessun dettaglio viene dato relativamente alla realizzazione hardware, se non per qualche semplice schema di connessione, né tanto meno alla realizzazione meccanica. Per i dettagli di meteorologia e sul significato delle grandezze rilevate e calcolate si presuppone che ognuno abbia i mezzi e la voglia di andarsele ad approfondire.

1. Architettura Hardware

La scheda di controllo della stazione meteorologica è una FOXG20 della ACME



(http://www.acmesystems.it/FOXG20) basata su CPU Atmel AT91SAM9G20 (ARM926EJ-S – ARM) 400MHz, 64M Bytes SDRAM dalle dimensioni ridotte 40x40mm e peso 10g.
La flash card memory (2GB fino a 16GB) non è compresa nella fornitura. Il file immagine par la scheda può essere scaricato e tutto il software disponibile è open source.
I files da salvare nella prima partizione FAT16/32 della SMD Card possono essere scaricati qui:

cmdline.txt (MD5: ad69d39a63d618eb22a165f88c1a4373)
machtype.txt (MD5: 9043ef9464aecb10ad8071fc13794eb0)
uImage (MD5: 4c95a1188edf9062ce7a1ac3ed8540c5)

I file da decomprimere e copiare nella partizione ETX4 della SD card possono essere scaricati qui:
rootfs.tar.bz2 (MD5:36813f4a71963ec6c26052c7f16a3aae)

Le credenziali di accesso sono
login: root
password: netusg20

Il sistema originale ha bisogno dei seguenti moduli per lavorare in tempo reale con i canali di I/O (GPIO) e i canali Analogici/Digitali :

debarm:~# wget http://www.acmesystems.it/www/adc_kernel_2x/g20/at91-adc.ko

e il modulo gpio_dev.ko (che io ho ma che non trovo più sul sito)

Deve essere installato anche il servizio **incron** (apt-get come al solito su debian) , lo conoscete?, funziona come **cron** ma invece che su eventi temporali reagisce ad eventi generati dal filesystem manager (tipo apertura, chiusura di file , accesso a directory e molto altro) , date una occhiata su http://inotify.aiken.cz/?section=inotify&page=about&lang=en

Per chi ha fretta posso fornire il file immagine **meteo.img** da scaricare sulla SMD da 4GB.

La scheda ha così tutto quello che serve per realizzare il controllo della Stazione Meteo:

- 4 canali conversione Analogica Digitale (ne servono 3 al massimo)
- Input Output digitali (ne servono 4)
- Servizio di web server (httpd)
- Servizio ssh per collegamento remoto a terminale e file system.

- Alimentazione 5V per i sensori
- Alimentazione 3,3 V per i sensori

2. Sensori

Anemometro

Per la misura della velocità e la direzione del vento ho utilizzato l'anemometro TX20 della LaCrosse con annessa la banderuola per la direzione del vento , reperibile online come ricambio a prezzi compresi tra 25,00 € e 36,00 €.

Le caratteristiche tecniche del sensore sono le seguenti:

Misura Velocità : 0-180 Km/h
Risoluzione = 0.1 m/s →0.36 Km/h
Direzione : 0 - 360° risoluzione 22.5°
Alimentazione : 3.3 -5.0 V

Il sensore viene fornito con un cavo di lunghezza 25 m circa e un connettore RJ11 (tipo telefonico) la cui piedinatura è la seguente:

Pin	*Colore*	*Descrizione*	
1	Brown	TxD	Trasmissione seriale
2	Red	Vcc	3,3-5 V
3	Green	DTR	Enable (GND Abilitato)
4	Yellow	GND	Voltage reference

Con il DTR collegato permanentemente a 0V la trasmissione dei dati avviene circa ogni secondo con un treno di impulsi (41) della durata ognuno di 1,2 ms.
Durante la pausa di trasmissione la linea seriale TxD è tenuta a 0V.
La codifica è a logica invertita: "bit 0" = impulso a Vcc "bit 1" = impulso 0V
Il primo bit è il meno significativo (LSB)
Il frame di una singola stringa di bit (41 impulsi → 49.2 ms) è la seguente:

1-5	5 bit di **Start** sempre 11011
6-9	4 bit **Direzione** (0000 → Nord , 1111 → Nord/Nord/Ovest)
10-21	12 bit (di cui i primi 3 sempre a zero) **Velocità** (Vmax = 511 → 183.96 Km/h)
22-25	4 bit **Checksum** (OR esclusivo)
26-29	4 bit negati **Direzione** (è lo stesso valore di 6-9)
30-41	12 bit negati di **Velocità** (è lo stesso valore di 10-21)

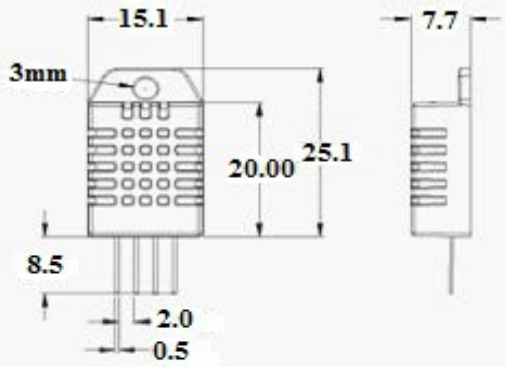
Termometro – Idrometro

Per la misura della temperatura (T) e della Umidità percentuale relativa (RT) ho utilizzato il sensore RTH03 (http://www.humiditycn.com/fl1_1.html) reperibile online al prezzo di circa 4,00 €.

Le caratteristiche tecniche del sensore sono le seguenti:

Modello	**RHT03**
Alimentazione	3.3-5.5V dc
Output signal	digital signal via MaxDetect 1-wire bus
Sensing element	Polymer humidity capacitor
Operating range	humidity 0-100%RH; temperature -40~80Celsius
Accuracy	humidity +-2%RH (at 60%RH, 25Celsius);temperature +-0.5Celsius
Resolution or sensitivity	humidity 0.1%RH; temperature 0.1Celsius
Repeatability	humidity +-0.5%RH; temperature +-0.5Celsius
Humidity hysteresis	+-0.5%RH
Long-term Stability	+-0.5%RH/year
Interchangeability	fully interchangeable

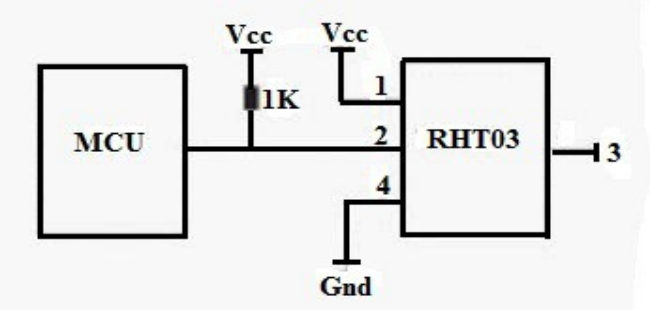
Il sensore che viene fornito è con piedinatura per circuito stampato la cui disposizione dei pin è rappresentata qui di seguito: (il pin 1 è quello più a sinistra)



Pin sequence number: 1 2 3 4 (from left to right direction).

Pin	Function
1	VDD—power supply
2	DATA--signal
3	NULL
4	GND

Lo schema classico di collegamento, che io ho utilizzato, è il seguente (Vcc = 3,3V che non va bene per Arduino) :

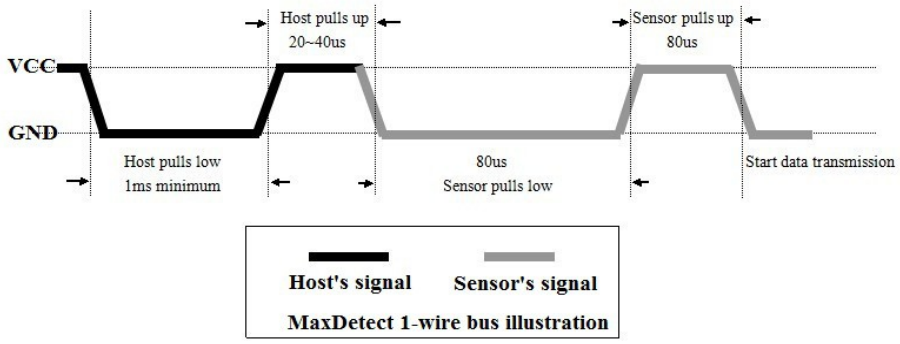


Segnale di comunicazione:

Il sensore utilizza il bus di comunicazione **1-wire** per la trasmissione dei dati tra la MCU and RHT03.
(**1-wire bus** è progettatto specificatamente dalla MaxDetect Technology Co., Ltd. , è differente dallo standard Maxim/Dallas 1-wire bus, quindi con questi incompatibile.), questo vuol dire:
”*state attenti ad usare comunicazioni 1wire predefinite, potrebbero non funzionare !!!*”
(*chi fa da se fa per tre ed è servito come un re)*

Di seguito le specifiche di connessione:

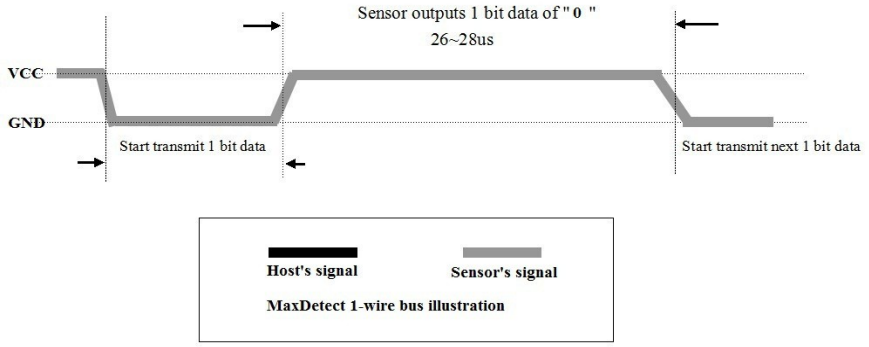
1) Step 1:
<u>MCU invia un segale di Start al sensore RTH03 e questi risposnde con un segnale di Ack alla MCU</u>
Generalmente la linea di Bus è in lata impedenza e allo stato di High Level Voltage (HLV)
Quando deve iniziare una comunicazione tra MCU e RTH03 la MCU “tira” a Low Level Voltage (LLV) la linea di Bus e lo mantiene da 1 a 10 ms in modo che RTH03 possa rilevare il segnale, trascorso questo tempo la MCU “tira” ad HVL il Bus per circa 20-40µS e quindi rilascia il bus in Alta Impedenza (cioè configura il canale come input).
RTH03 quindi interpreta questo come un segnale di start e risponde con un segnale di ACK così composto : 80µS LVL e 80µS HVL.
Vedere la figura di seguito.

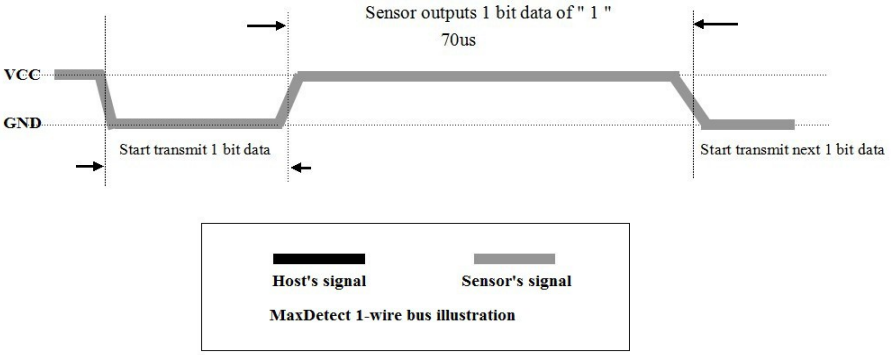


2) Step 2:

RTH03 trasmette i dati alla MCU

Quando il sensore RTH03 trasmette il treno di impulsi alla MCU ogni bit è sempre preceduto da un segnale LLV di durata pari a 50µS, il seguente impulso ad HVL rappresenta uno "1" o un "0" a seconda che la durata sia rispettivamente di 70µS o di 20-28µS





Attenzione:

Se il segnale da RTH03 è sempre allo stato HLV significa che questi non sta lavorando correttamente, è bene controllare quindi le connessioni elettriche.

Il frame della stringa binaria è ben specificato dal seguente esempio:

Esempio: MCU ha ricevuto 40 bit da RHT03 :

0000 0010 1000 1100 — 16 bit RH data
0000 0001 0101 1111 — 16 bit T data
1110 1110 — 8 bit check sum

ovvero:

check sum=0000 0010 + 1000 1100 + 0000 0001 + 0101 1111 = 1110 1110

+ significa OR EXCLUSIVO (0+0=1+1 0 1+0=0+1=1)

RH = bin (0000 0010 1000 1100)/10 = 65.2%RH

T = bin (0000 0001 0101 1111) /10 = 35.1?

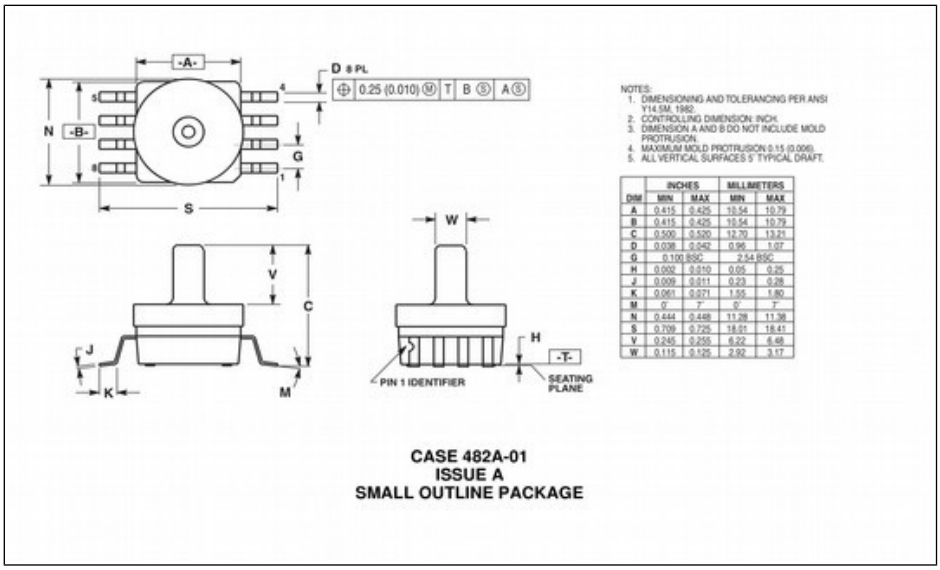
quando il bit più significativo della temperatura è ad "1", allora il valore di temperatura è negativo

Esempio: 1 000 0000 0110 0101 , T= - 10.1?

16 bits T data

Barometro

Per la misura della pressione assoluta ho utilizzato il sensore MPX4115AX reperibile online al prezzo di circa 15,00€. (un po' caro eh?).
Il sensore viene fornito in diversi tipi di contenitore , quello che ho utilizzato io , rappresentato in figura è il CASE 482A-01 è con piedinatura per circuito stampato.
Le caratteristiche tecniche del sensore sono le seguenti: (VS = 5.1 Vdc, TA = 25°C)



Characteristic	Symbol	Min	Typ	Max	Unit
Pressure Range	POP	15	—	115	kPa
Supply Voltage	VS	4.85	5.1	5.35	Vdc
Supply Current	I_o	—	7.010		mAdc
Minimum Pressure Offset (0 to 85°C) @ VS = 5.1 Volts	V_{off}	0.135	0.204	0.273	Vdc
Full Scale Output (0 to 85°C) @ VS = 5.1 Volts	VFSO	4.725	4.794	4.863	Vdc
Full Scale Span (0 to 85°C) @ VS = 5.1 Volts(5)	VFSS	4.521	4.59	4.659	Vdc
Accuracy(0 to 85°C) VFSS	———			±1.5%	
Sensitivity	V/P	—	46	—	mV/kPa
Response Time	tR	—	1.0	—	ms
Warm-Up Time		—	20	—	mSec
Offset Stability	—	—	±0.5	—	%VFSS

Lo schema di collegamento e la caratteristica di uscita sono rappresentate in figura

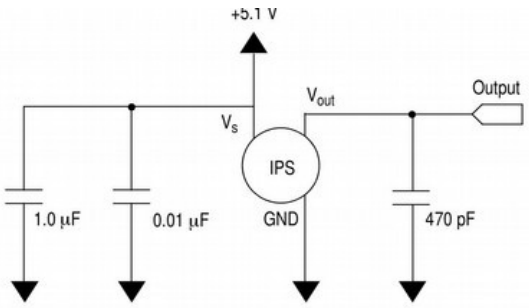


Figure 3. Recommended Power Supply Decoupling and Output Filtering
(For output filtering recommendations, refer to Application Note AN1646.)

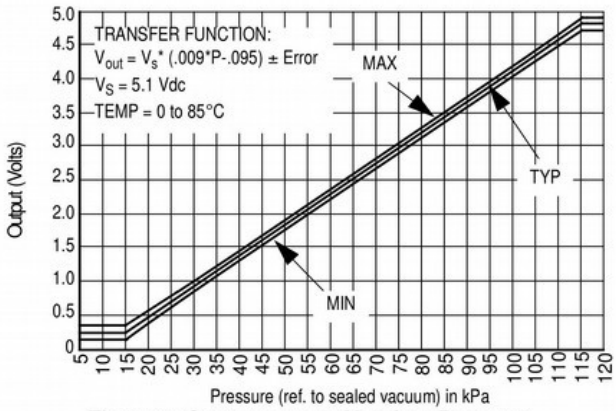


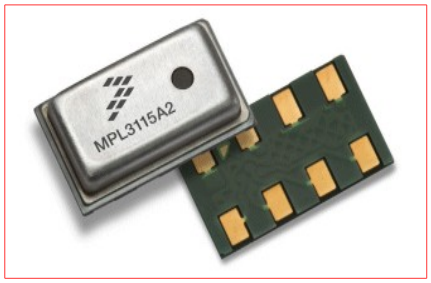
Figure 4. Output versus Absolute Pressure

Dovendo adattare l'uscita analogica all'ingresso del convertitore Analogico/Ditale della scheda (Max 3,22 V) ho partizionato l'uscita con due resistenze da 33 KΩ , in pratica dimezzando il valore della tensione.
Volendo una conversione esatta del valore in volt in KPa si ha:

Vout = VS (P x 0.009 - 0.095) ± (Pressure Error x Temp. Factor x 0.009 x VS)

con

VS = 5.1 V ± 0.25 Vdc
Temp. Factor = 1 da 0 a 85 °C
Pressure Error = ± 1.5 Kpa da 15 a 115 Kpa



Essendo il sensore abbastanza costoso, altre alle controindicazioni di un sistema analogico, ci sarebbe l'alternativa di un sensore con uscita digitale I2c (standard) che però è per circuiti SMD ; quindi di difficile applicazione su basette amatoriali self made.
Comunque per chi fosse interessato la sigla è MPL3115A2 con range da 50 to 110 kPa.
Il costo di questo sensore è di circa 3,50 € reperibile online.

Solarimetro

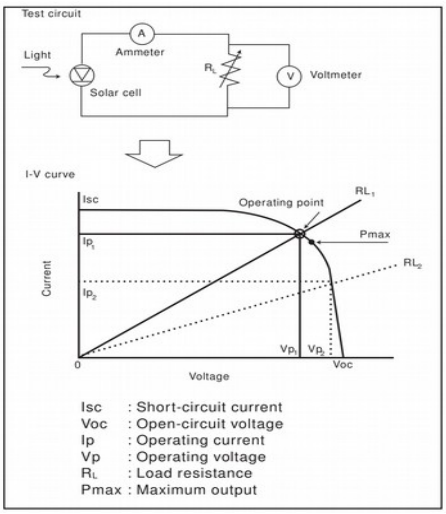
Per realizzare questo strumento ho utilizzato una cella fotovoltaica del costo di circa 3,00 € reperibile dal Paoletti – Firenze ma non so per quanto ancora.

La cella utilizzata è la seguente: Panasonic BP-378234 adatta all'utilizzo esterno e quindi sensibile alla luce solare. Essendo questo un sensore autocostruito cercherò di dare qualche informazione riguardo alla sua realizzazione e alla teoria che ci sta alla base in modo da poter adattare qualunque altra cella di cui si possano conoscere le caratteristiche.

La caratteristica di questa cella è di essere utilizzata per irraggiamento solare e quindi adatta, con qualche precauzione, all'utilizzo esterno. La speciale costruzione della cella consente il trasferimento di alte correnti quali quelle generate in presenza di irraggiamento solare.

E' possibile ottenere un range di tensioni di uscita adatto alla applicazione. Questo lo si può ottenere imponendo , attraverso un carico resistivo, il punto di lavoro (Tensione [V] Corrente[I]) della cella.

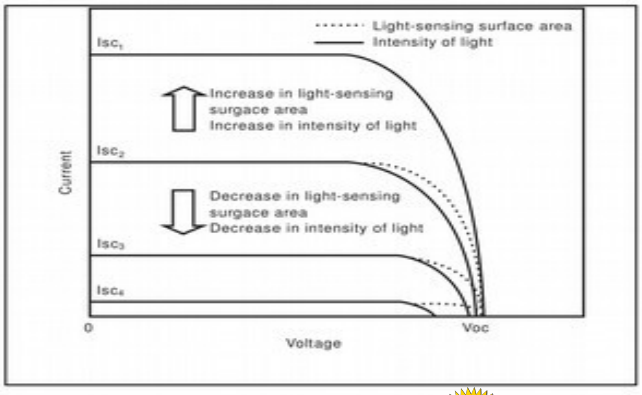
Le caratteristiche di uscita della cella (qualunque cella solare) sono espresse attraverso il diagramma Tensione Corrente , il così detto <u>diagramma I-V</u>.



Una curva I-V tipica di una cella solare è disegnata nella figura accanto. La caratteristica curva si ottiene applicando un carico , resistivo variabile all'uscita della cella per una particolare condizione di irraggiamento espressa generalmente in W/m^2.

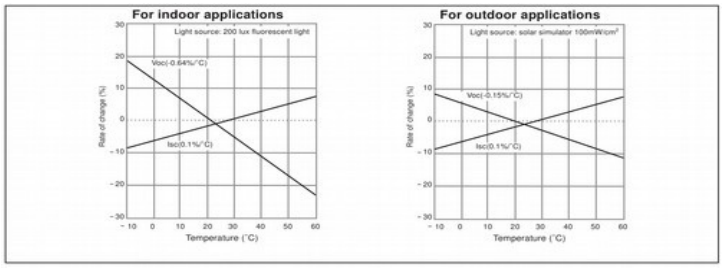
Per ogni valore della resistenza si ottiene una coppia di valori I_p e V_p detti appunto, per quel caratteristico carico , <u>punto di lavoro</u>.

Punti caratteristici della curva sono quelli per cui il valore della resistenza è infinito (circuito aperto I=0 e $V=V_{oc}$) e quello della resistenza R=0 (corto circuito $I=I_{cc}$ e V=0) oltre al punto di Massima Potenza che è quello che massimizza il prodotto $P_{max}= V_p*I_p$



La curva si varia al variare delle condizioni di irraggiamento; in particolare il valore di I_{cc} cambia , praticamente, in maniera direttamente

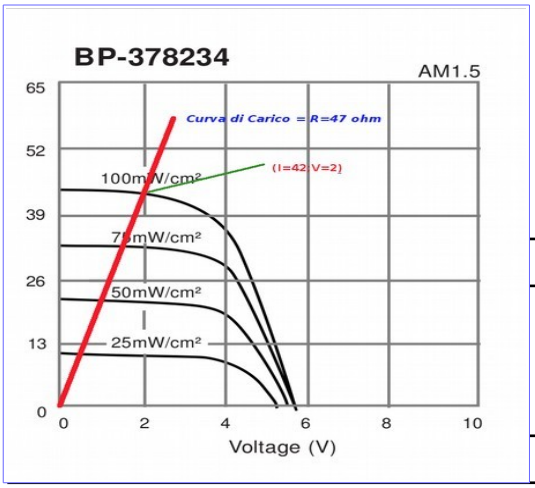
proporzionale con il valore di energia incidente sulla superficie della cella. Cambia, ma di poco anche il valore di V_{oc}.



Un altra caratteristica che influenza il diagramma I-V è la temperatura. Le prestazioni di una cella fotovoltaica sono tali che il valore di correnti di corto circuito I_{cc} aumenta e il valore di circuito aperto V_{oc} diminuisce al crescere della temperatura secondo una legge lineare rappresentata nella figura accanto.

Le caratteristiche della cella BP-378234 sono le seguenti:

Dimensioni	**37.0X82.0 mm**
I_p	**40.0 mA**
V_p	**3.40 V**
V_{oc}	**5.50 V_{oc}**
I_{cc}	**43.0 mA**



La curva di carico scelta in questo caso è quella disegnata in rosso sul diagramma relativo alla cella BP-378234 che corrisponde ad un carico di 48 Ω.
In queste condizioni per un irraggiamento di 1000 W/m^2 si hanno 2 V in uscita. La tensione risulta , al variare della energia irradiante, circa lineare:

$$E_{irr} = V_{out} \times 500$$

Considerando che la massima energia alle nostre latitudini è di circa 1300 W/m^2 avremmo in uscita

$$V_{max} = 1100/500 = 2.2 \text{ V}$$

Devo osservare che su questo sensore c'è ancora molto da lavorare infatti:

- Nessuna compensazione in temperatura è stata effettuata
- Non tutta l'energia irradiata dal sole è misurata: la luce è filtrata (caratteristica spettrale) da 500 a 1100 nm

Questo fa si che il valore di energia irradiata debba essere corretto con metodi empirici facendo riferimento a qualche strumento esistente nella zona.
(vedi ad esempio http://www.lamma.rete.toscana.it/meteo/osservazioni-e-dati/dati-stazioni)

Collegamento dei Sensori

La figura alla pagina seguente rappresenta lo schema di collegamento dei sensori alla unità di controllo FOXG20. Questa è dotata due due connettori denominato J6 e J7 di cui il pinout è presente al seguente indirizzo http://www.acmesystems.it/pinout_foxg20 .

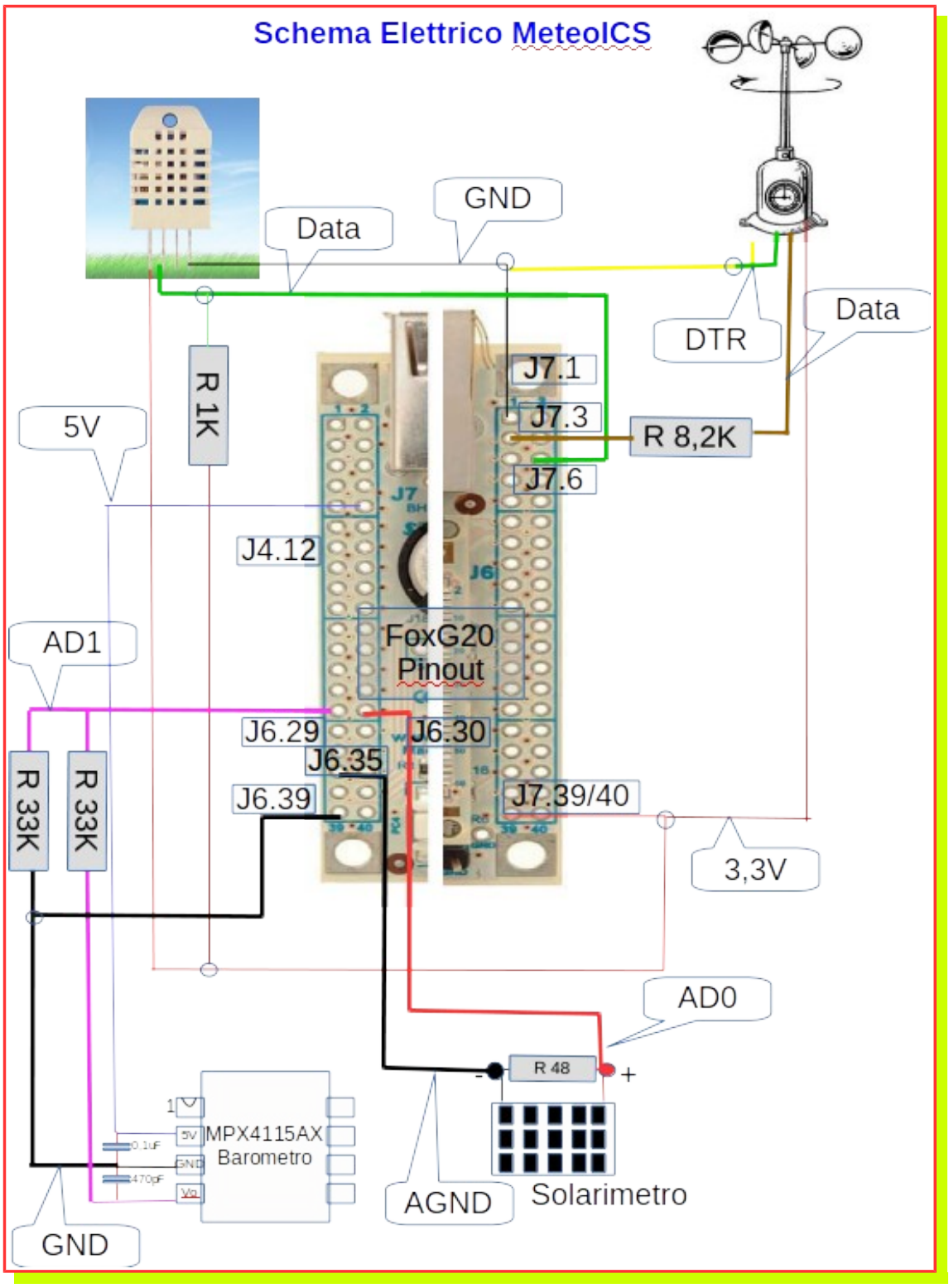
La rappresentazione è puramente teorica, è ovvio che a seconda delle condizioni di montaggio e installazione devono essere fatte delle scelte in base al tipo di connettori da utilizzare e come disporre i componenti e come installare e proteggere i sensori dal sole e da fattori meteorologici.

L'anemometro viene alimentato a 3,3 V dalla tensione di scheda attraverso i Pin J7.39-J7.40 (3,3V) e J7.1 (GND). Il DTR è mantenuto a livello “0” collegandolo a GND, il Data Bus e collegato, attraverso una resistenza da 8,2K Ω, al pin J7.3 (GPIO 82) che viene configurato dal software come linea di Input.

Il sensore di temperatura e umidità è alimentato a 3,3 V mediante i pin J7.39-40 e J7.1, il data Bus è collegato al pin J7.6 con una resistenza di pull-up di 1K Ω in modo da mantenerlo a livello ”1” quando è in alta impedenza.

Il sensore di Pressione è alimentato a 5V attraverso il pin J6.12 e J6.39 (GND), Il segnale analogico di uscita opportunamente partizionato (½) e stabilizzato mediante un condensatore da 470 pF, è collegato al canale ADC1 del pin J6.29. Il condensatore da 0.1 uF sulla alimentazione è richiesto nelle specifiche applicative.

Alla cella solare che costituisce il **solarimetro** è applicata una resistenza di carico da 48 Ω la cui tensione ai capi , proporzionale alla intensità dell'irraggiamento solare, è riferita alla massa analogica della scheda (AGND) pin J6.35, ed è misurata sul pin J6.29 che è il canale analogico ADC0 della scheda FOXG20.

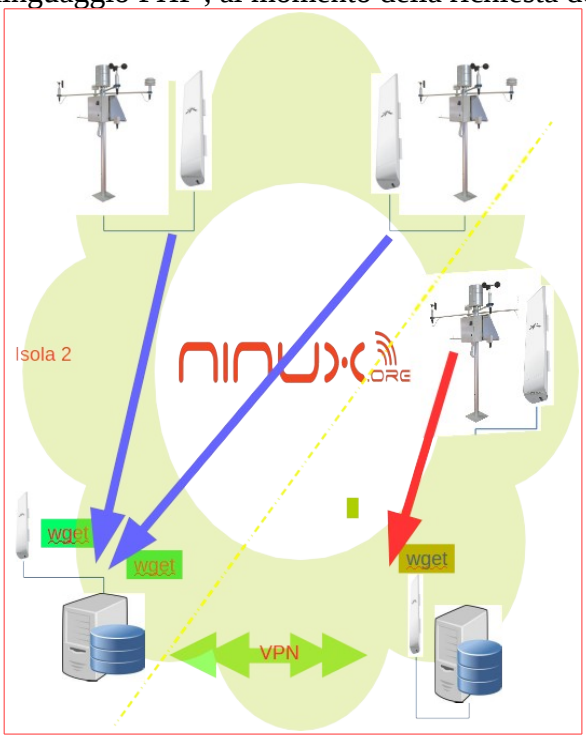

Schema Elettrico MeteoICS
Data
GND
DTR
Data
R 1K
5V
J7.1
J7.3
R 8,2K
J7.6
J4.12
FoxG20
Pinout
AD1
J6.29
J6.30
J6.35
J6.39
J7.39/40
R 33K
R 33K
3,3V
AD0
R 48
MPX4115AX
Barometro
AGND
Solarimetro
GND

3. Architettura Software

Il software che gestisce il sistema è residente in due ambienti:

- La scheda FoxG20 (Debian su architettura ARM)
- Il Server che ospita il sito Web e il DataBase (Una qualunque distribuzione LINUX)

La parte sviluppata ad hoc, praticamente quella che gira sulla scheda, è quella relativa ai driver di gestione dei sensori e quella per la strutturazione dei dati da trasferire verso il server.
Il server è "attrezzato" con LAMP (Apache,MySql e PHP).
I linguaggi utilizzati sono il C standard cross-compilato in ambiente ECLIPSE , il Python per gli script di trasferimento dati e il PHP per le pagine WEB.
Il trattamento dei dati per il calcolo e la rappresentazione dei valori meteorologici è affidata al linguaggio PHP , al momento della richiesta delle pagine, ed è residente sul server.



Il sistema di integrazione delle stazioni in un unica rete prevede che ogni stazione sia in grado di accumulare dati in maniera autonoma strutturati sotto forma di file i cui record sono formattati secondo una regola comune.

Ogni server , attualmente solo uno, contiene una lista di Stazioni Client, dalle quali prelevare i dati sulla base di eventi temporali (ogni giorno, ogni ora, ogni 15 minuti). Dopo che i file sono stati acquisiti vengono automaticamente cancellati sul Client.

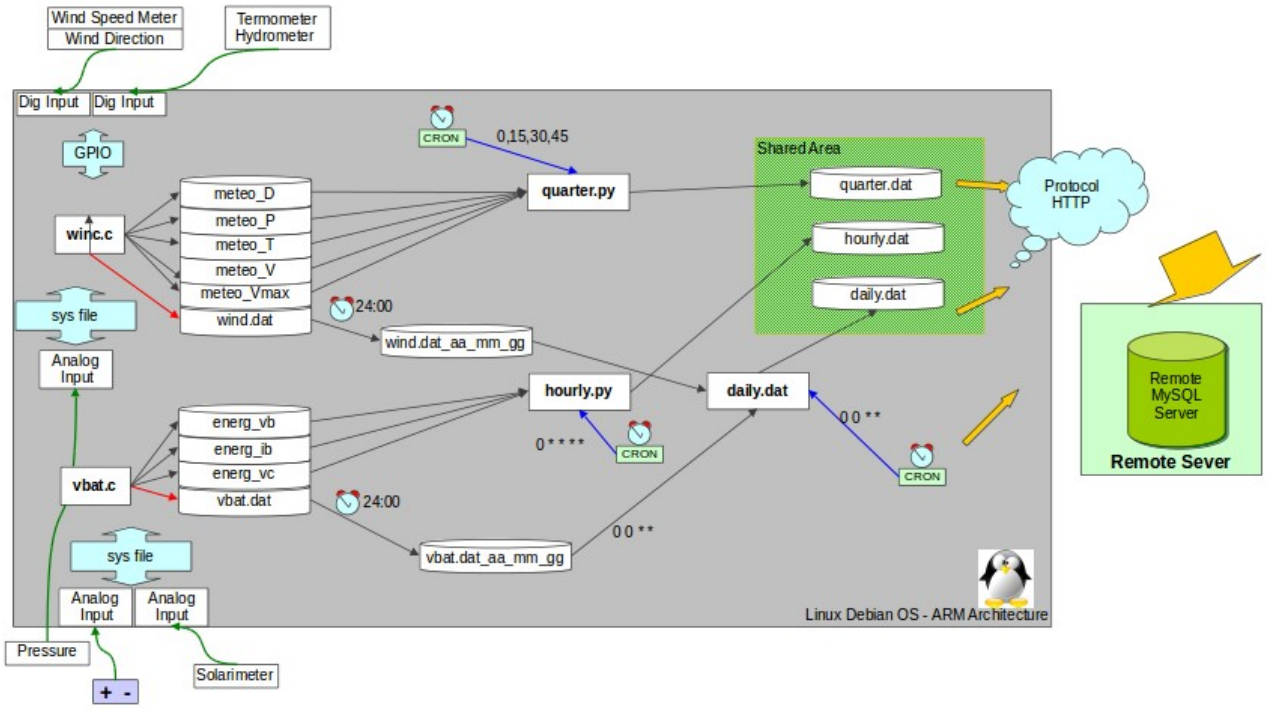
Il protocollo adottato è HTTP, ovvero il server esegue una "wget" verso ogni stazione. Questo presuppone che il Client abbia attivo un servizio http server e che sia raggiungibile attraverso un ip pubblico o sia noto l'indirizzo ip nella rete comunitaria.

Questo è il motivo per cui la scheda che controlla la stazione ha un sistema operativo Embedded. (vi ci vorrei vedere con Arduino!)
L'infrastruttura di rete attualmente è quella di Internet (dove serve un ip pubblico e statico), ma niente vieta che questa possa essere quella di Ninux (dove l'ip è noto e statico).
Le stazioni di un isola possono essere asservite ad un unico server il quale è collegato in VPN ai server di altre Isole.
Il Web server, dove risiede il sito web del servizio meteorologico, può essere unico e differenziato

dai server di stazione o replicato su più macchine distribuite nella rete. Su questo argomento lascio la parola a chi ne sa più di me.
In questo modo ogni stazione meteorologica , comunque essa sia realizzata può essere integrata in questo sistema purché si riesca a definire un driver per prelevare i dati grezzi.
Nel router su cui risiede OpenWRT può anche essere implementato il sistema di organizzazione dei files formattati in modo da poter essere condivisi con il server di Isola (si tratta di qualche script in python).

Architettura Software a bordo Scheda

Il digramma seguente rappresenta l'organizzazione del software a bordo della scheda FOXG20



Il sistema si compone di diversi processi concorrenti:

wind.	*sviluppato in linguaggio c*
vbat.	*sviluppato in linguaggio c*
quarter.py	*sviluppato in python*
hourly.py	*sviluppato in python*
daily.py	*sviluppato in python*

I programmi in linguaggio C sono sviluppati in ambiente Eclipse, la toolchain necessaria alla cross-compilazione dei sorgenti è scaricabile dal sito della ACME (http://www.acmesystems.it/FOXG20).

wind

Esegue la lettura dei dati di direzione del vento, di velocità del vento , di temperatura , di umidità e di pressione atmosferica. Si compone di due thread che realizzano i driver di interfaccia verso l'anemometro e verso il sensore di temperatura per la decodifica delle sequenze seriali attraverso canali GPIO.
La lettura del valore di pressione avviene tramite canale analogico opportunamente convertita in unità ingegneristiche.
I dati sono condivisi con gli altri processi attraverso memoria condivisa (mmap : memory-mapped file) un a sorta di file dove sono mappate le variabile e trattati a livelo di codice come una sorta di memoria virtuale. A questo tipologia di dati apertengono i dati a cuisiti in te,mpo reale di:

Velocità del Vento (Km/h)	meteo_V
Velocità Massima del vento (Km/h)	meteo_Vmax
Direzione del Vento (gradi sessagesimali, Punti cardinali (N;S;E;O)	meteo_D
Temperatura in gradi centigradi	meteo_T
Umità relativa in percentuale	meteo_U.

Inoltre il processo aggiunge una volta ogni minuto circa un record al file wind.dat che è un file in formato ascii composto da record così strutturati:

Data	Time	Km/h	MAX Km/h	Gradi	Direzione	Checksum	Inversione	sleep	Temperatura	Umidita	validate	mbar
23/Oct/2014	00:01:08	0.00	0.00	292.5	WNW	5	0	0	13.0	38.0	1	1010
23/Oct/2014	00:01:55	0.00	0.00	292.5	WNW	10	0	0	13.0	38.0	1	1009

Questo file viene acquisito una volta ogni giorno dal server per archiviazione. Se questo avviene con successo il file viene eliminato dalla memoria flash della scheda.

vbat

Esegue la lettura dei dati di tensione batteria , di tensione fotocellula solarimetro,di corrente batteria e valori medi. Questi dati sono tutti acquisito mediante convertitori analogico/digitali

Come per il processo precedente, wind.c, i dati sono condivisi con gli altri processi attraverso memoria condivisa (mmap : memory-mapped file) un a sorta di file dove sono mappate le variabile e trattati a livello di codice come una sorta di memoria virtuale. A questo tipologia di dati appartengono i dati acquisiti in tempo reale di:

Tensione Batteria (V)	meteo_vb
Tensione fotocellula (V)	meteo_vc
Corrente Batteria (A)	meteo_ib

Inoltre il processo aggiunge una volta ogni minuto circa un record al file vbat.dat che è un file in formato ascii composto da record così strutturati:

Data	Time	Vbat	Vcell	Ampere	Vbat M	Vcel M
23/Oct/2014	00:00:00	11.797	0.016	-0.251	11.825	0.007
23/Oct/2014	00:01:00	11.846	0.013	-0.331	11.825	0.007

Questo file viene acquisito una volta ogni giorno dal server per archiviazione. Se questo avviene con successo il file viene eliminato dalla memoria flash della scheda.

Processi a tempo:
Attraverso l'uso di cron a scadenza di tempo regolare vengono avviati tre processi :

quarter.py	ogni 15 minuti
hourly.py	ogni ora
daily.py	ogni giorno allo scadere della mezzanotte

La scopo di questi processi è quello di predisporre rispettivamente i seguenti file alla lettura del server remoto :

/var/www/meteox/data/quarter/quarter.dat
/var/www/meteox/data/hourly/hourly.dat
/var/www/meteox/data/daily/daily.dat

Queste cartelle sono accessibili da remoto attraverso richieste HTTP (wget) con le quali il server aggiorna il proprio data base. Una volta che questi file sono stati letti vengono cancellati. Il sistema è basato sul processo “icron” analogamente a “cron” reagisce ad eventi su agenti su file o cartelle. Nel caso particolare l'evento di chiusura in lettura del file è sincronizzato con uno script che cancella il file.
Se la lettura non avviene allora il nuovo record di dati vengono aggiunti ai precedenti di modo che i dati non vengano persi, ma mantenuti sulla scheda fino a che il server no li abbia acquisiti.
I processi accedono alle memorie condivise di wind e vbat per generare record nei file condivisi con Mysql server remoto.

L'intero progetto Software è scaricabile da github…..... ed è utilizzabile in tutte le sue parti perché coperto da licenza GPL.