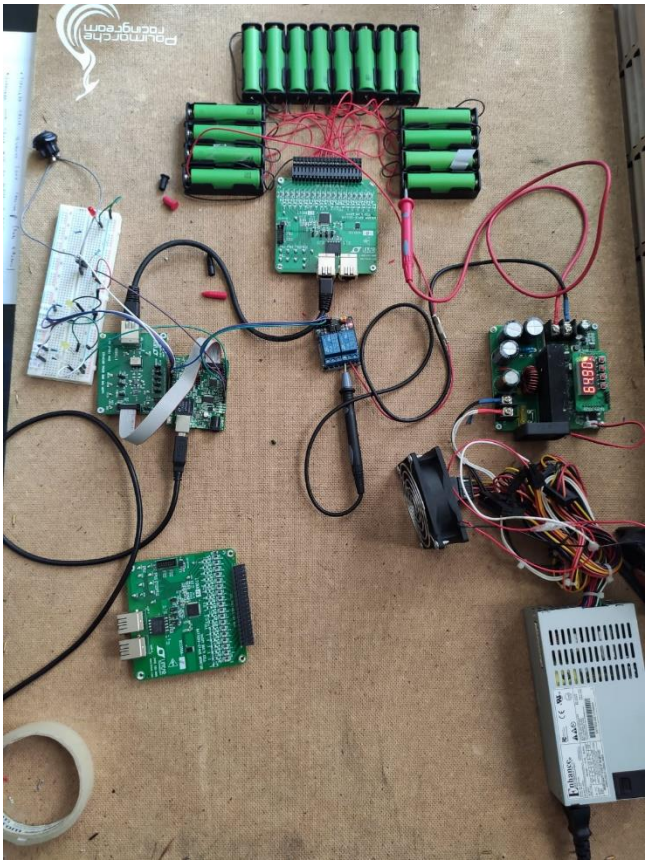
Configurazione

Il pacco batteria è formato da 16 celle formato 18650 modello VTC5 della SONY, queste sono collegate in serie una all'altra e la tensione di ciascuna di loro è misurata dall'ADC LT6813 come descritto nelle specifiche del file README.md su github.

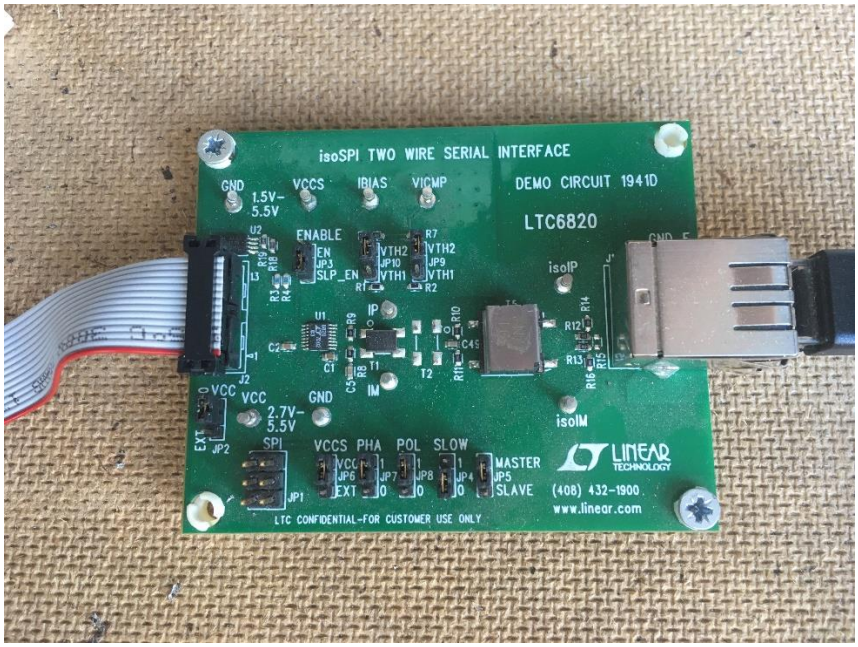


Ogni cella rappresenta un parallelo di 6 celle che verranno implementate nel modulo finale. Attualmente il pacco è implementato da un modulo solo, questa decisione è stata momentaneamente presa per velocizzare il processo di test.

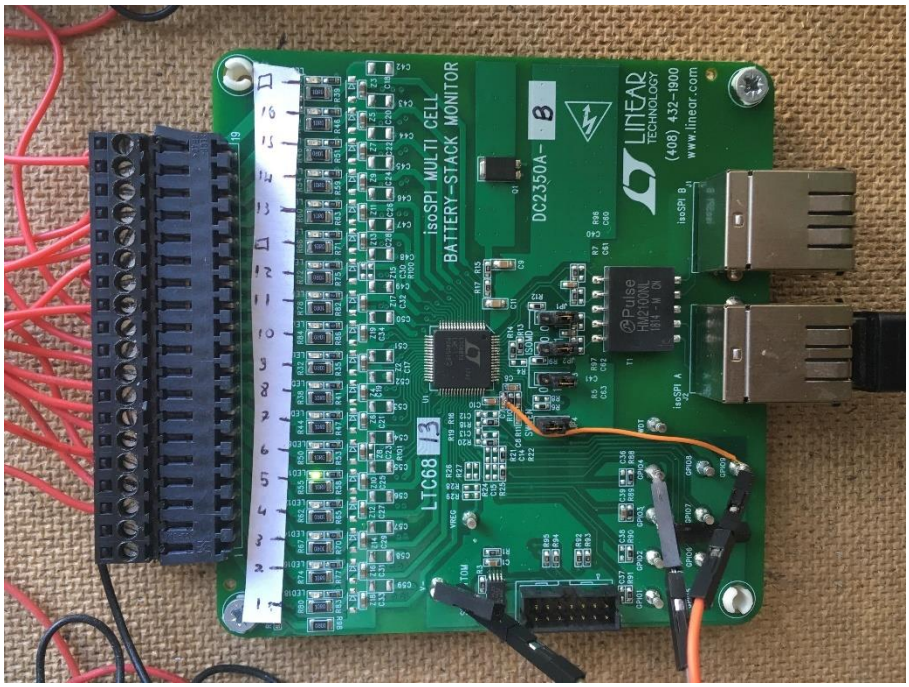
Cose utilizzate

- **Linduino:** E’ il microcontrollore che viene utilizzato come master per i nostri ADC e per comunicare con il nostro compiuter. Al suo posto è possibili utilizzare un qualsiasi Arduino con processore ATMega (l’Arduino UNO è stato testato ed è funzionante).
  Non siamo riusciti a far funzionare il codice con scheda NUCLEO-F446RE per problemi con la comunicazione SPI. (codice convertito tramite STM32duino).

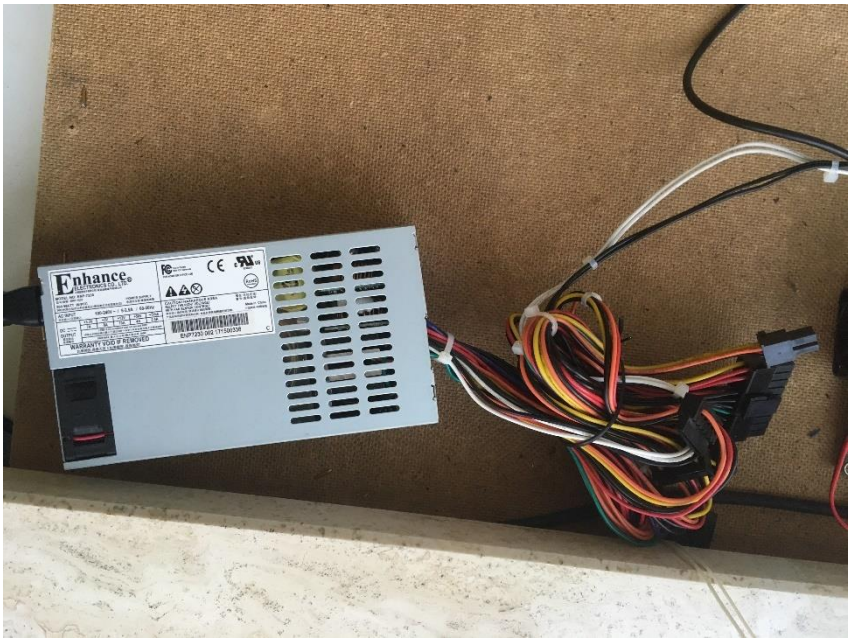
- **DC1941D:** Provvede all’isolamento tra la tensione delle batterie e la scheda Linduino, è collegata con un cavo direttamente al Linduino (il cavo collega l’alimentazione e i pin 13-12-11-10 del Linduino, è utilizzato per semplice comodità).
  Oltre a ciò questa scheda provvede a convertire i segnali SPI dal Linduino a Iso-SPI mandati gli ADC, questa è una comunicazione proprietaria della Linear che migliora affidabilità e velocità.



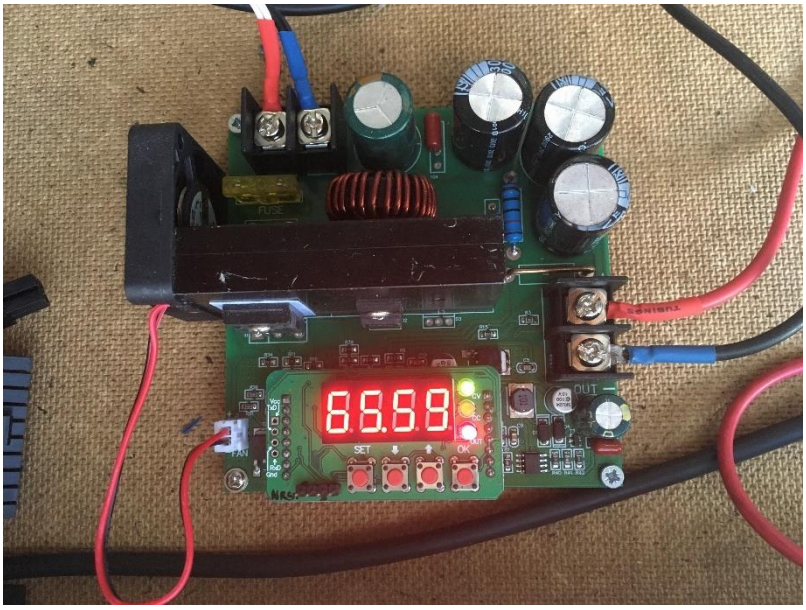
- **DC2350A-b:** E’ l’evaluation board che viene utilizzata (basata sull’ADC LTC6813), ci consente di gestire un massimo di 18 tensioni per le nostre celle (nel codice ne vengono utilizzate 16) e 9 entrate general purpose (che verranno utilizzate per vari sensori, come gli NTC).



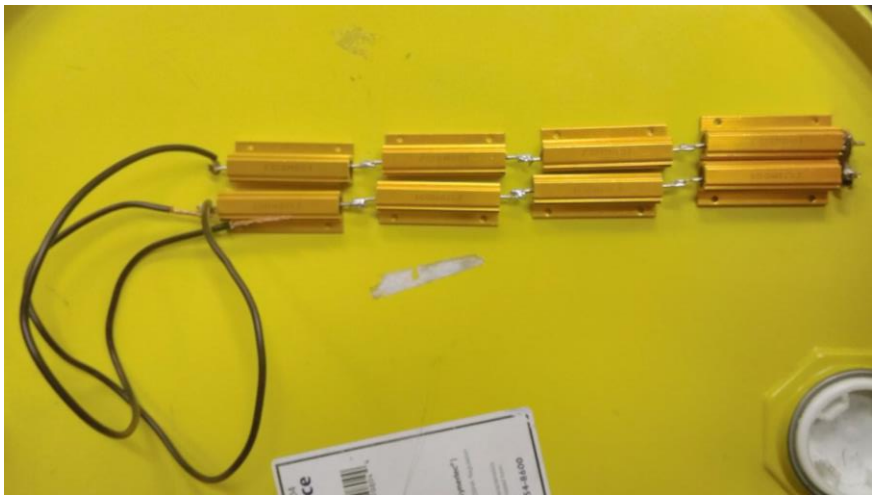
- **Alimentatore:** E’ un alimentatore rigenerato da un lettore dvd avente oltre ai classici connettori ATX, anche una uscita 50V 5A (che verrà utilizzata per alimentare il nostro step up).



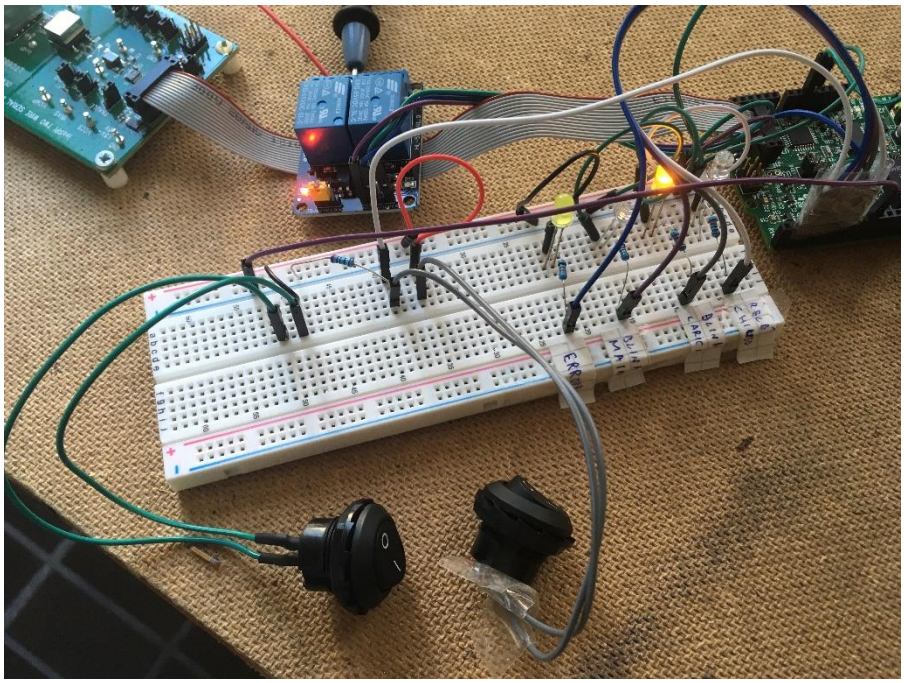
- **Step up:** Capace di erogare 0-15A con ingresso 8-60V e uscita 10-120V facilmente reperibile su amazon o altri siti di e-commerce. Per i nostri scopi è alimentato da 50V e tramite il display viene settata la tensione massima e la corrente massima desiderata (per i nostri scopi 2A 65.60V) questo provvederà da solo a fornire CC o CV a seconda di quanto richiesto dalle batterie.



- **Resistenze:** Abbiamo formato una serie di 8 resistenze tarate a 100W: 4 da 1 ohm e 4 da 8 ohm, per un totale di 36 ohm capace di dissipare 100w.



- **Led e cavetteria varia:** In foto un esempio della nostra workbench.

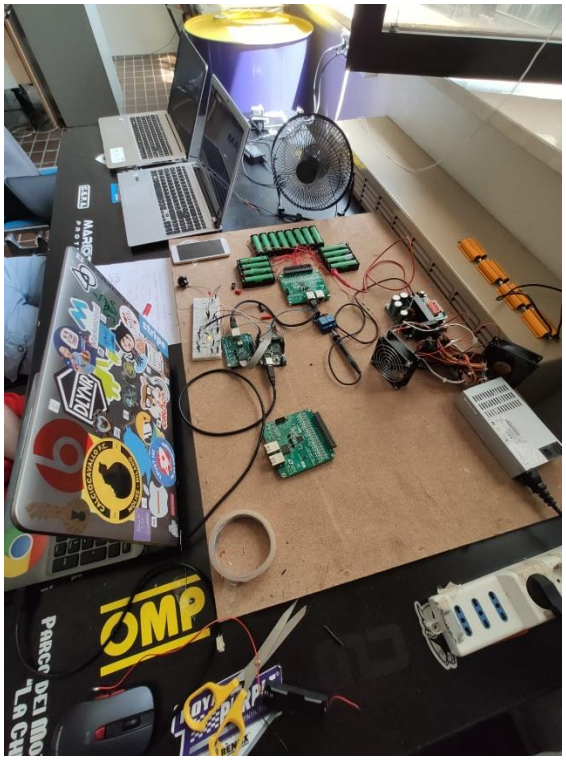


Scarica del pacco

La scarica è stata effettuata collegando ai capi del pacco una serie di 8 resistenze per un totale di 36 ohm. I terminali della resistenza di scarica vengono collegati (attraverso un relè che controlla il circuito) tra il capo negativo della prima batteria e il capo positivo dell'ultima.
Un piccolo accorgimento che consiglio è quello di aggiungere qualche avvolgimento al filo che collega l'alimentatore alle celle per evitare le interferenze elettromagnetiche.
Una volta collegato il tutto, accendiamo CoolTerm (un software che ci permette di visualizzare il monitor seriale del Linduino e di salvarlo in un file txt), chiudiamo il relè e spettiamo che le tensioni delle celle si abbassino fin quando una arrivi a 2.5V (tensione di cutoff per le celle VTC5).
Il raggiunginmento della tensione di cutoff ci viene segnalata dall'accensione del led di errore.

Carica del pacco

I terminali dell'insieme alimentatore - step up vengono collegati (sempre attraverso un relè che controlla il circuito) tra il capo negativo della prima batteria e il capo positivo dell'ultima.
Viene suggerito lo stesso accorgimento di aggiungere qualche avvolgimento al filo che collega l'alimentatore alle celle per evitare le interferenze elettromagnetiche.
Una volta collegato il circuito accendiamo l'alimentatore, impostiamo lo step up a 2A 65.60V e dal Linduino diamo il comando (attraverso un pulsante) di avviare la carica. Da qui in poi abbiamo salvato tutte le tensioni delle batterie grazie al software CoolTerm.
Non avendo il sensore di corrente (attualmente) la carica verrà fermata una volta che dallo schermo dello step up verrà mostrata una corrente inferiore di 0.1A.
L'algoritmo, esaustivamente descritto all'interno del codice, attiva a seconda della necessità una resistenza di scarica collegata in parallelo con ciascuna delle celle: se la differenza di tensione tra la cella più scarica e quella controllata supera un DELTA definito all'interno del codice, allora viene accesa la resistenza, se il DELTA viene superato di molto allora si ferma la carica e si aspetta un minuto per dare tempo alla cella di scaricarsi (questo controllo avviene temporalmente perché la tensione una volta staccata l'alimentazione cade rapidamente e le differenze di tensione in generale diminuiscono molto).

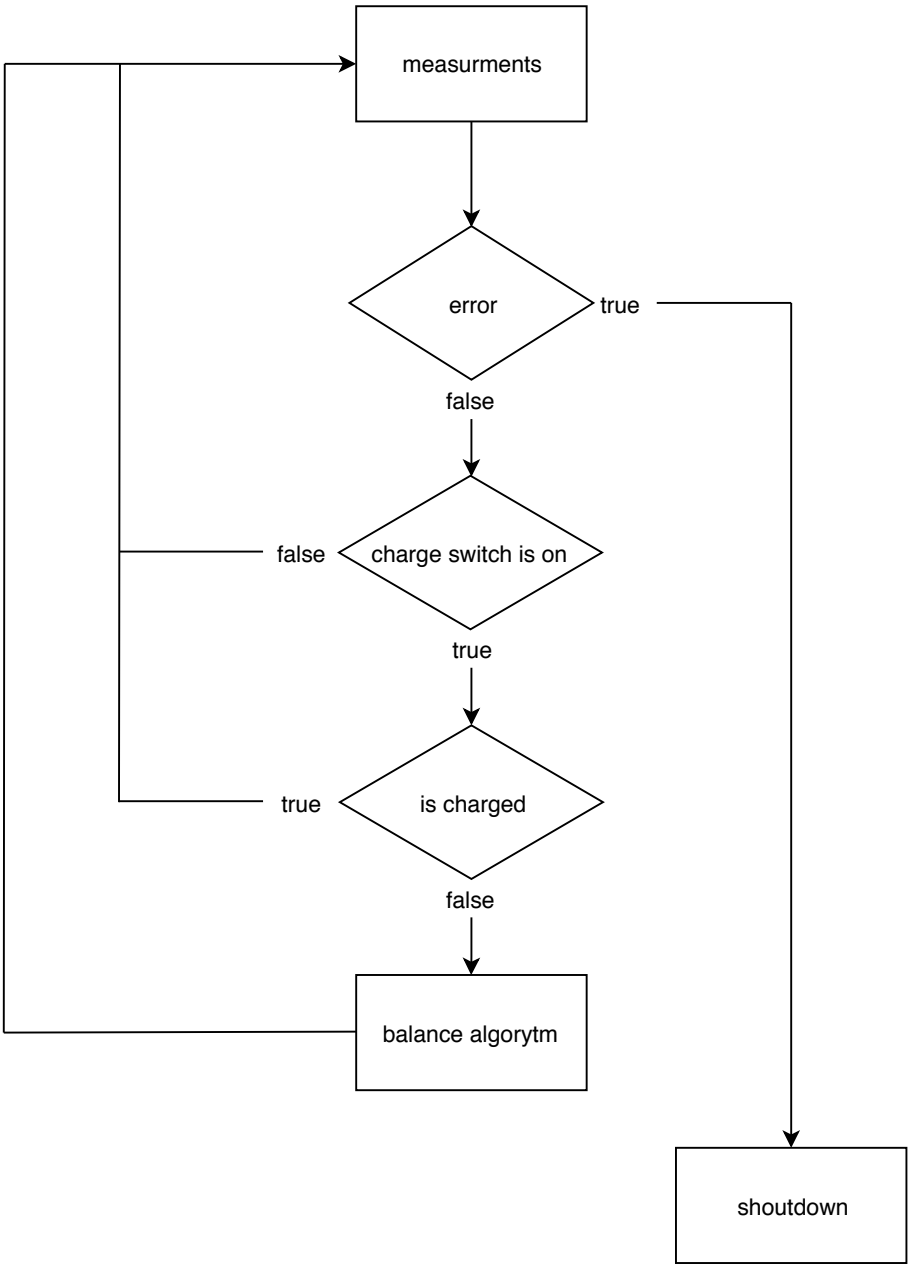


codice

Per descrivere meglio il codice, di seguito i diagrammi a blocchi del MainLoop e dell'algoritmo di carica, assieme al diagramma che descrive la creazione preliminare degli oggetti.
E' una descrizione molto semplificata, il suo scopo è quello di far capire le informazioni base sul funzionamento del codice, per una spiegazione a più basso livello invitiamo a leggere il codice, il quale è stato commentato nella sua interezza.

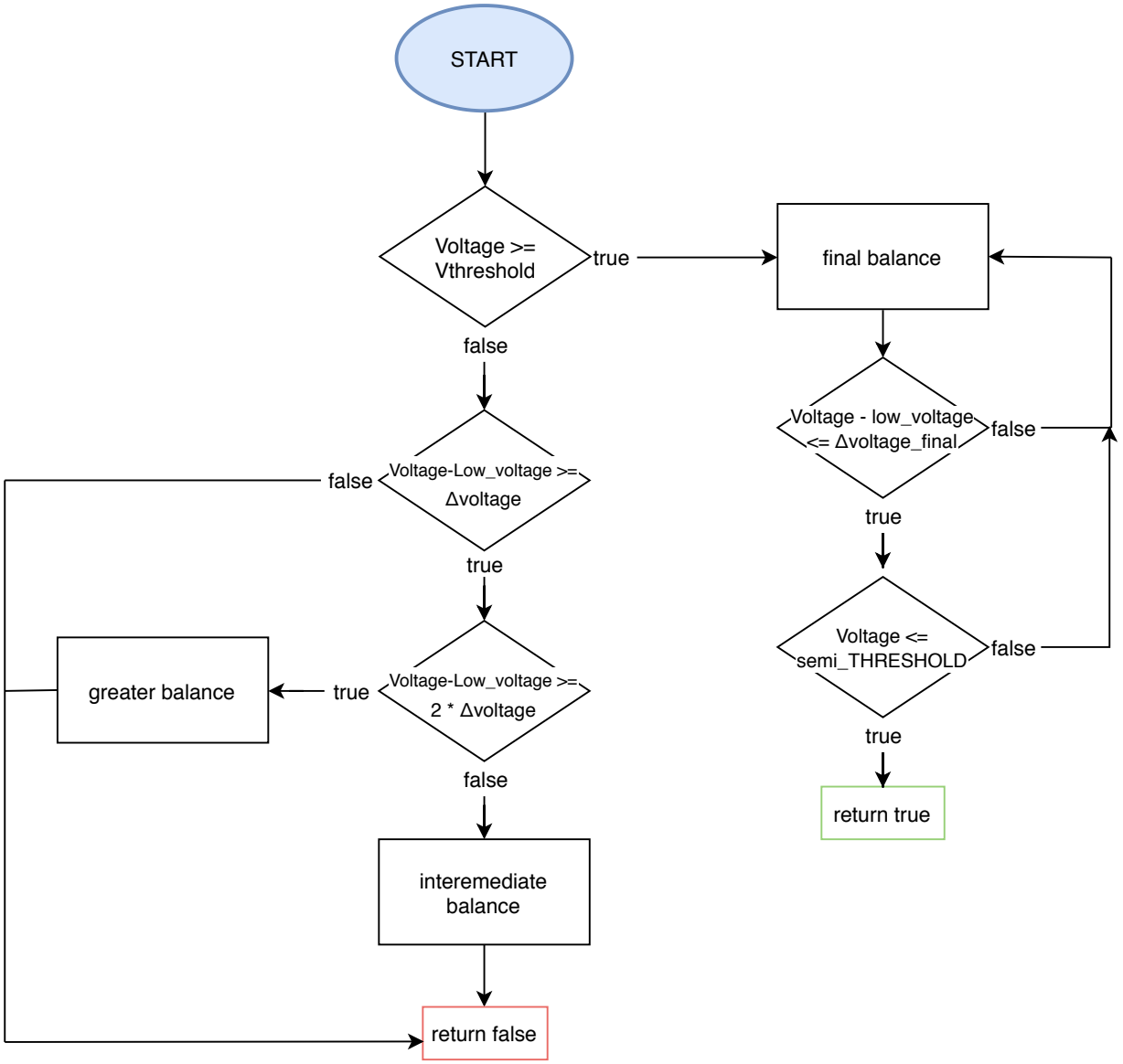
void loop()

semplified schematic of the loop()
function.



carica(cell_asic bms_ic[])

inside the object: CELLA



VARIABILES:

Voltage : Voltege of the current cell

low_voltage : lowest voltage between the cells of the pack

Vthreshold : threshold voltage to stop the charge, usually set 4.1V to give a gap for the rest of the battery pack to rise up, and to not involve in a OV_Error

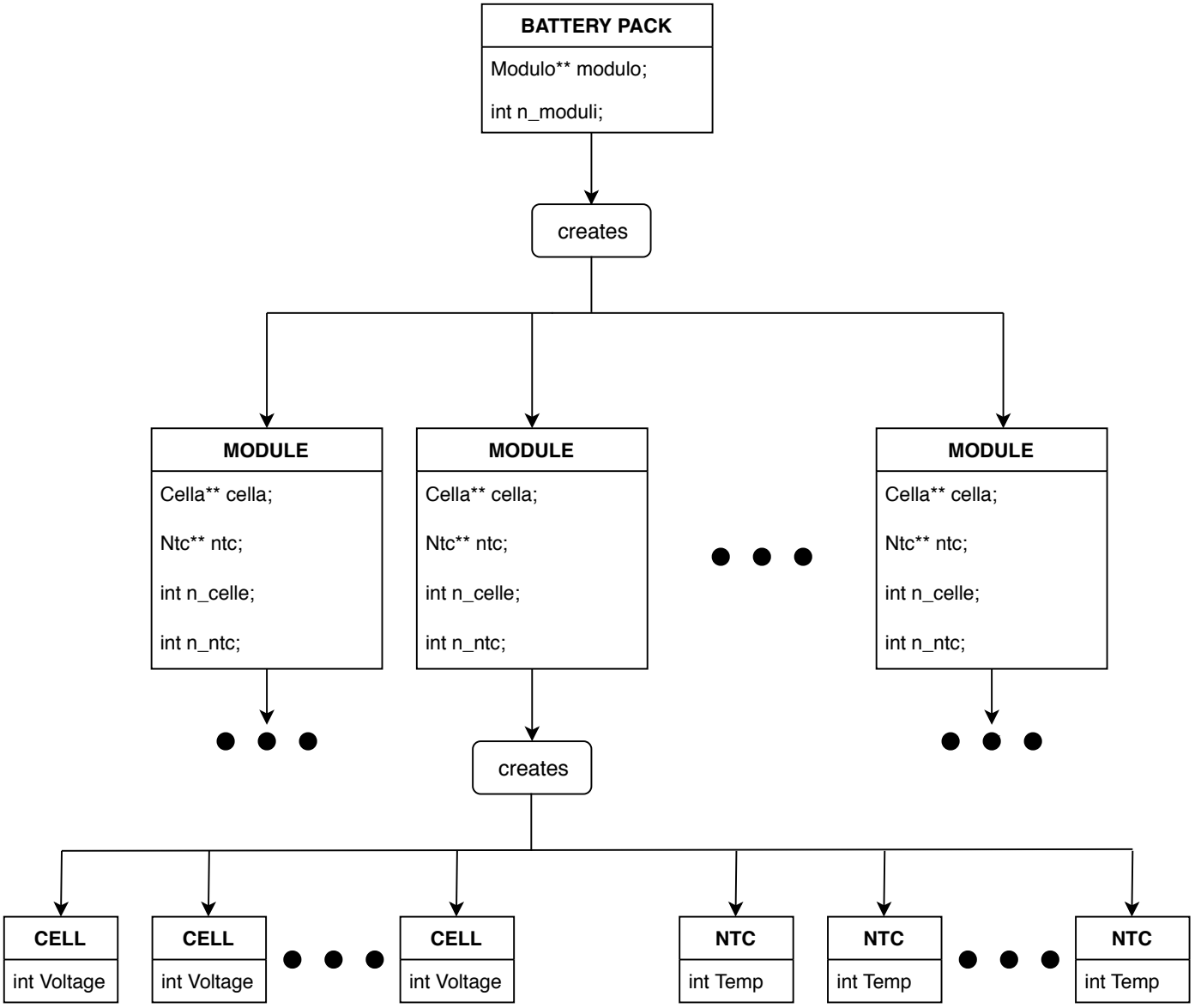
semi_THRESHOLD : a voltage near the OV_Threshold, to not involve in an OV_error, usually set to 4.19V

ΔVoltage & ΔVoltage_final : max voltage difference that we want in uor pack during the first and the last part of the charge

return false — this cell is not completely charge

return true — this cell is charged

OBJECT INITIALIZE



This is **NOT a block diagram**, BATTERY_PACK is not the father and MODULE does not extend it, that is also true for CELL and NTC.
here I'm showing how every object is created.

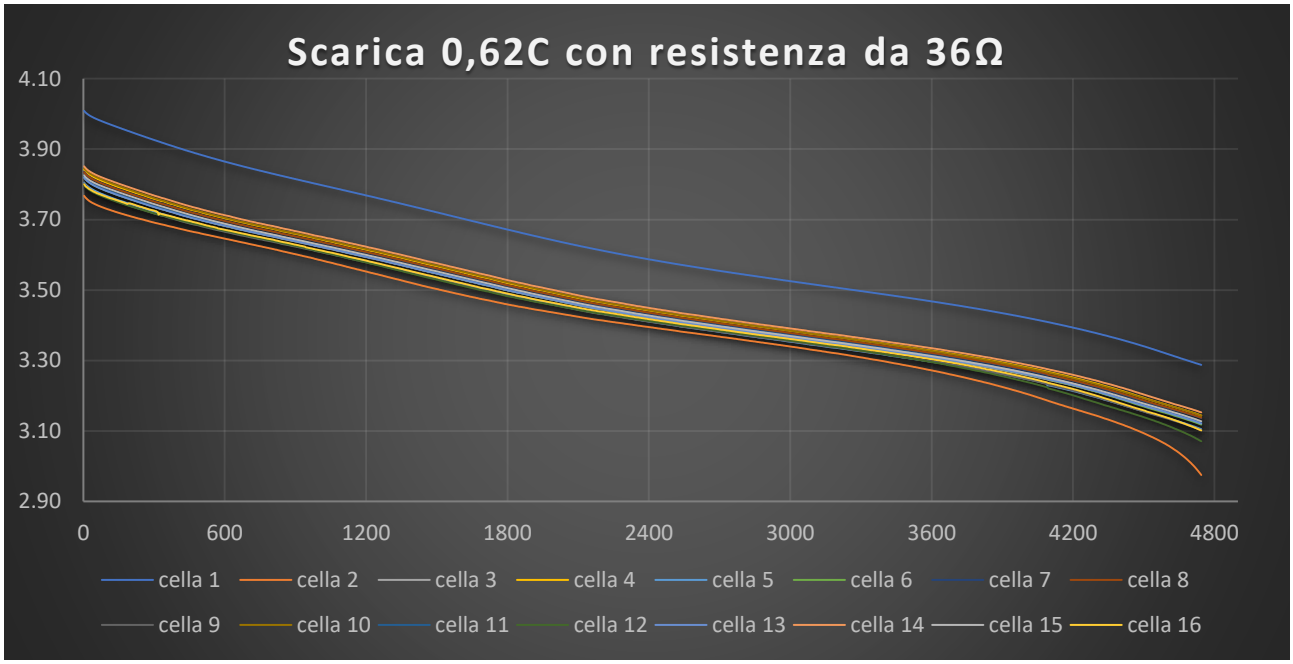
Only 1 BATTERY_PACK objet is created, it is important because in the main_loop() every method call start from that (pacco.error() etc..), and it call the same method for every MODULE, that obviously calls the proper method in CELL and NTC.

So the algorythm is structured by an itereative method call for every battery pack object, making it a **solid** but most important **modular** code, that is very important for our goals

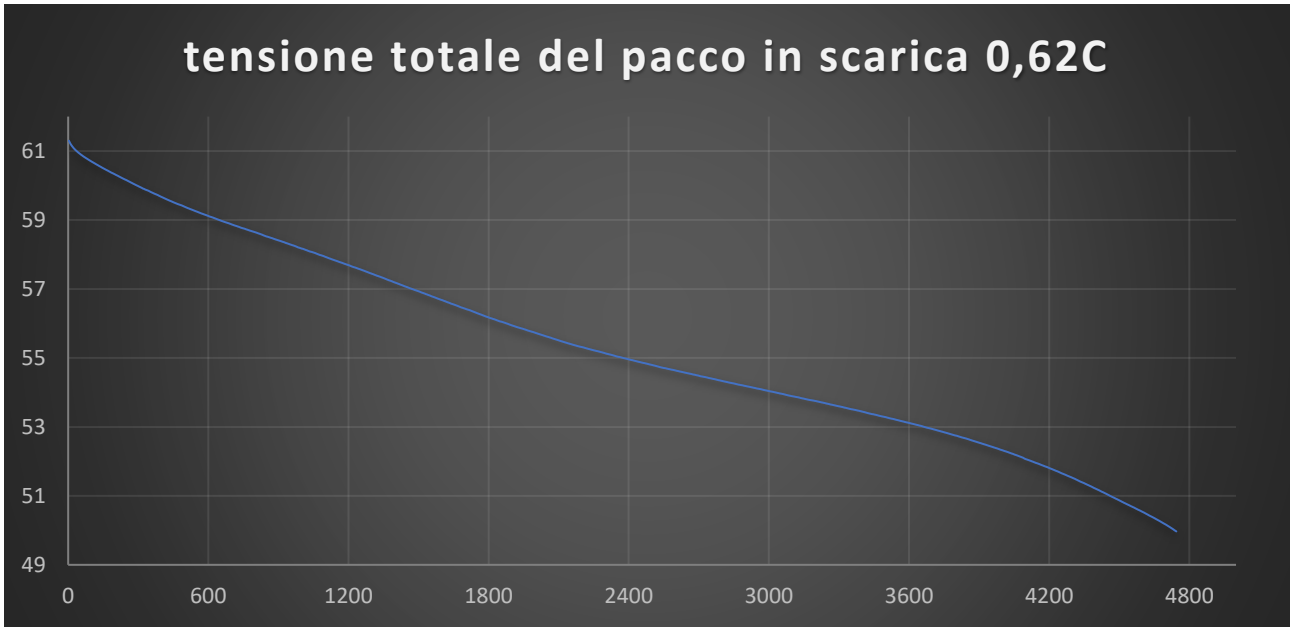
Risultati scarica del pacco (prima prova)

La prima tabella mostra l'andamento della scarica di ognuna delle 16 celle che formano il pacco:
sulle ascisse troviamo la tensione e sulle ordinate c'è il tempo in secondi, quindi ogni linea di separazione indica 10 minuti.

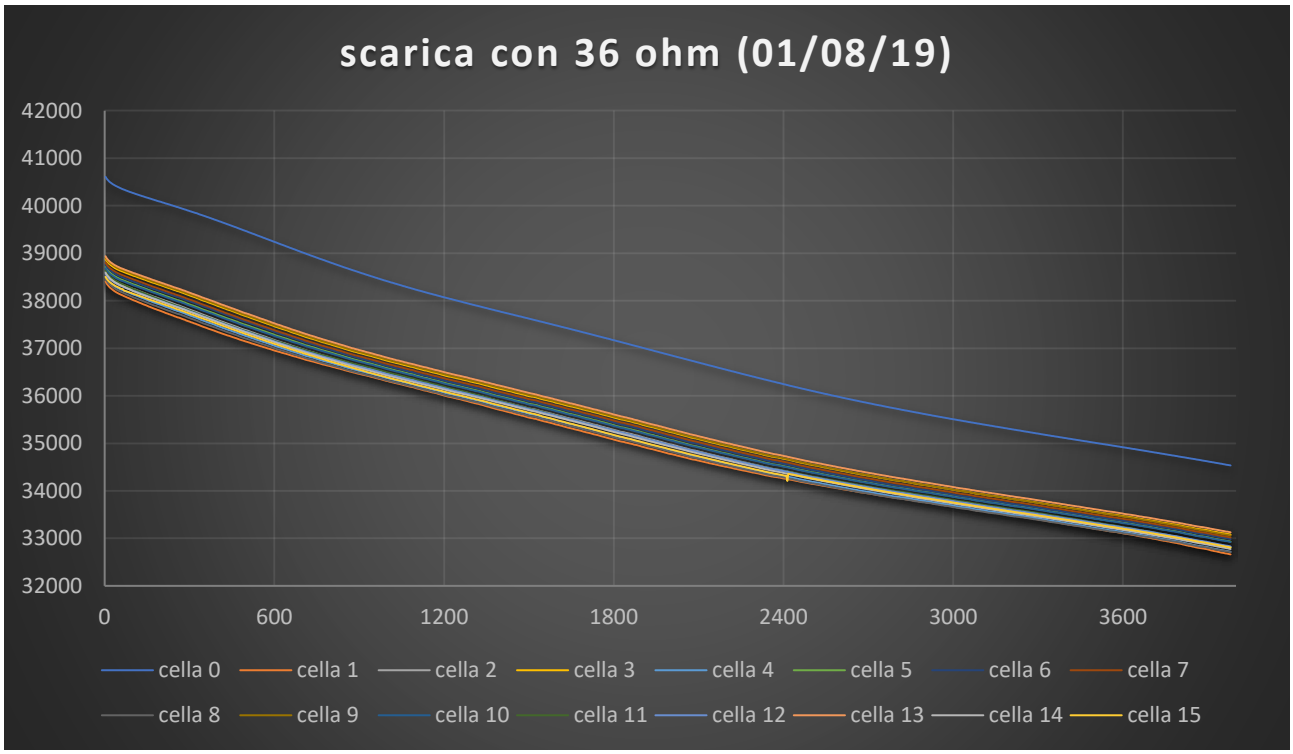
Notiamo che la cella 2 ha iniziato la scarica da una tensione più bassa delle altre e come previsto durante la parte finale del processo la tensione cade molto più velocemente delle altre.
c'è una situazione anomala, per la quale la cella 1 ha una tensione molto più alta delle altre durante tutto il processo di scarica, ma siamo arrivati alla conclusione che sia un errore di misurazione da parte dell'ADC.



Il secondo grafico deriva dalla raccolta dei dati precedenti e indica la tensione totale del pacco dal terminale più positivo al più negativo.



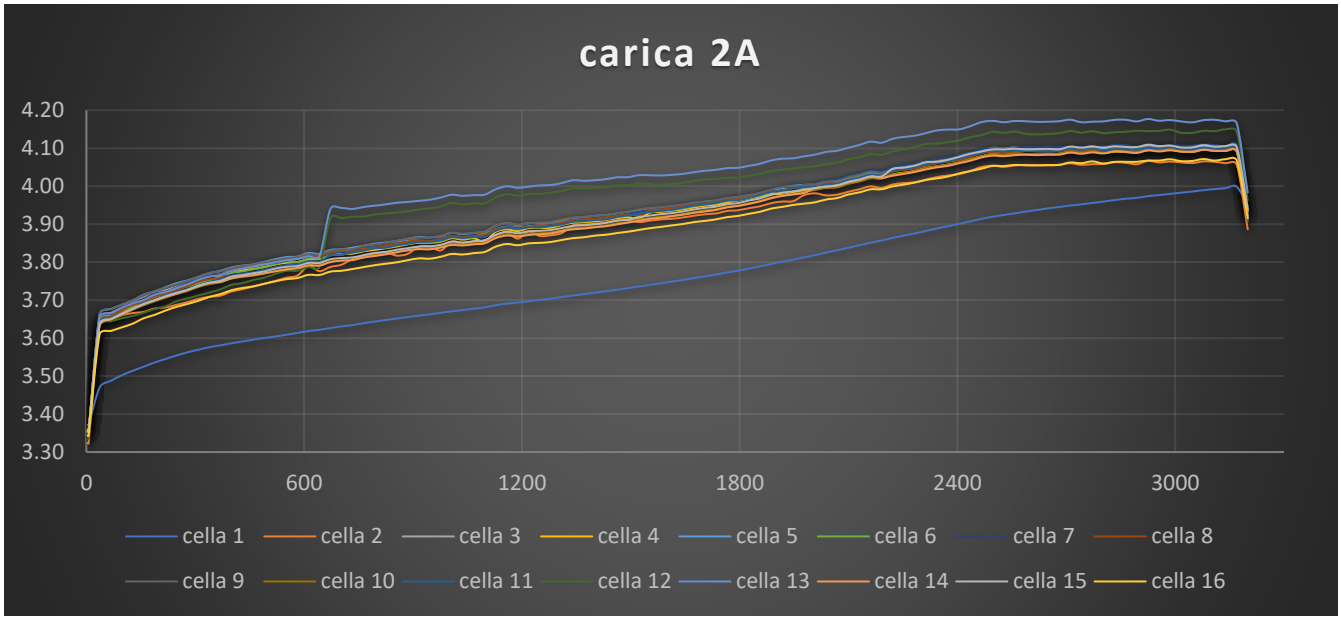
Successive scariche (01/08/19)



Risultati carica del pacco (prima prova)

Come per i risultati precedenti, la prima tabella mostra la tensione durante la carica di ognuna delle 16 celle che formano il pacco:
sulle ascisse troviamo la tensione e sulle ordinate c'è il tempo in secondi, quindi ogni linea di separazione indica 10 minuti.

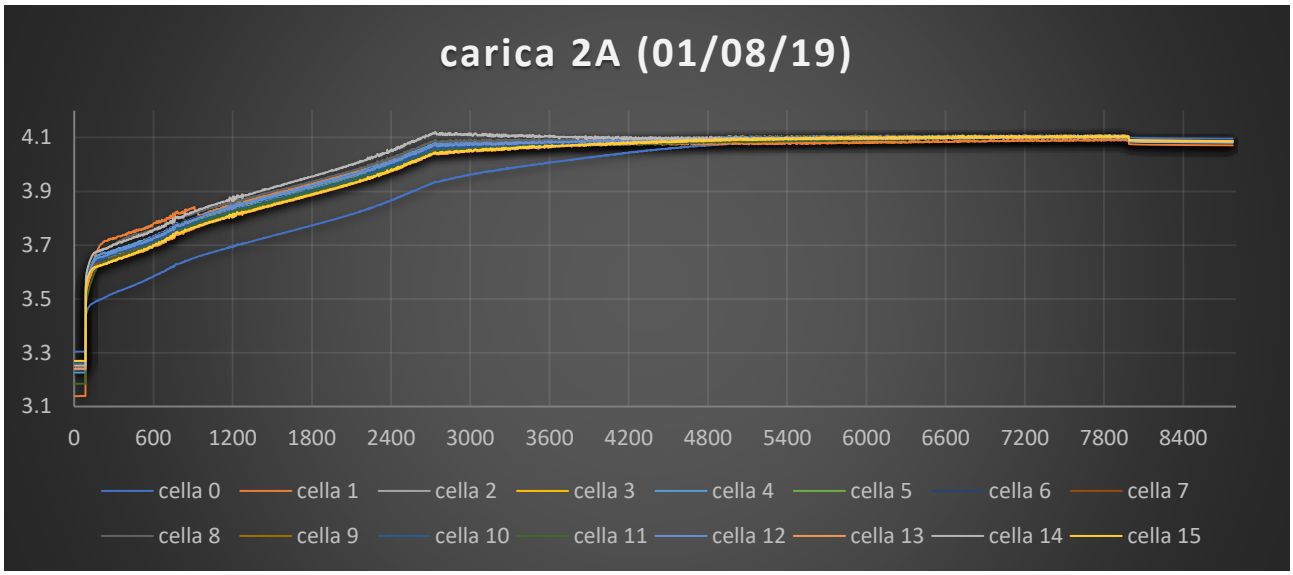
Dal grafico possiamo notare che dopo 600 secondi circa l'algoritmo attiva il bilanciamento delle celle più cariche e l'andamento rispetto alla cella 1 diventa più rumoroso (causato dalla misura delle tensioni con la resistenza in serie) e inizia a crescere più lentamente.
Si nota che la cella 13 e 14 aumentano molto la propria tensione appena si attiva la resistenza di scarica, questo ci fa pensare ad un errore di misurazione, dato che adiacente a queste celle c'è il corto che simula una cella a 0V (questo è necessario quando ci sono meno celle dell'entrate dell'adc, nel nostro caso 16-18);
Inoltre la cella 1 inizia dalla stessa tensione delle altre ma non cresce rapidamente come le altre, questo assieme agli scorsi grafici ci fa pensare che la cella 1 non goda di ottima salute, verrà indagato a riguardo.



Anche in questo caso la seconda tabella mostra l'andamento della tensione totale del pacco durante la scarica. la caduta di tensione finale è dovuta alla mancante seconda parte in CV.

Carica da celle completamente scariche (01/08/19)

In questo caso oltre ad avere l'informazione dell'andamento della carica completa (avendo implementato la seconda fase dell'algoritmo di carica CV) abbiamo anche la temperatura della seconda cella data dal nostro NTC.
Dal grafico vediamo come l'algoritmo di bilanciamento agisce egregiamente nella parte finale della carica.
Il piccolo drop finale avviene quando si staccano le batterie dall'alimentazione, il quale è molto piccolo, dato che abbiamo fermato la carica una volta arrivati a 0.09A .
Abbiamo completato la carica in 2h 26m.



La temperatura è maggiore nella fase di CC, dove la corrente è massima (2A).