

Politecnico di Milano
Esame di Informatica Industriale
Prof. Mezzalana

Implementazione di un caricatore industriale
veloce per accumulatori da trazione

di Daniele Paolo Scarpazza
Matr. 630655

1 Ringraziamenti

Si ringrazia l'Ing. Roberto Campari che ha progettato e messo a disposizione l'hardware utilizzato per questo progetto ed è stato ampiamente disponibile con chiarimenti e suggerimenti senza i quali non mi sarebbe stato possibile concludere il progetto.

2 Introduzione

Il progetto in questione è un dispositivo di controllo per un caricabatterie industriale veloce nel quale un motore elettrico trifase a 380 V è accoppiato ad un alternatore a 30 V, realizzando in tal modo una conversione meccanica di potenza; si è preferita la conversione meccanica alla conversione statica in quanto per la carica veloce di batterie di grande capacità i valori di corrente in gioco renderebbero non possibile o non conveniente o non affidabile o non efficiente realizzare approcci differenti.

Nella versione qui realizzata, che è un modello applicativo "in scala ridotta", il caricabatterie integra un motore da 1.2 kW e un alternatore in grado di sviluppare 60 A in uscita, ma sono già in corso prove per implementare la stessa circuiteria e logica di controllo a caricabatterie da 60 V e 180 A e con potenze in uscita ancora più elevate.

I processi di carica sono stati realizzati secondo le specifiche DIN 41173 gentilmente forniteci dal committente *Società Industriale Accumulatori S.p.A.*, titolare dei marchi EXIDE e Tudor, la cui linea di prodotti comprende le batterie a gel per veicoli stradali a trazione elettrica che devono essere caricate dal progetto qui descritto.

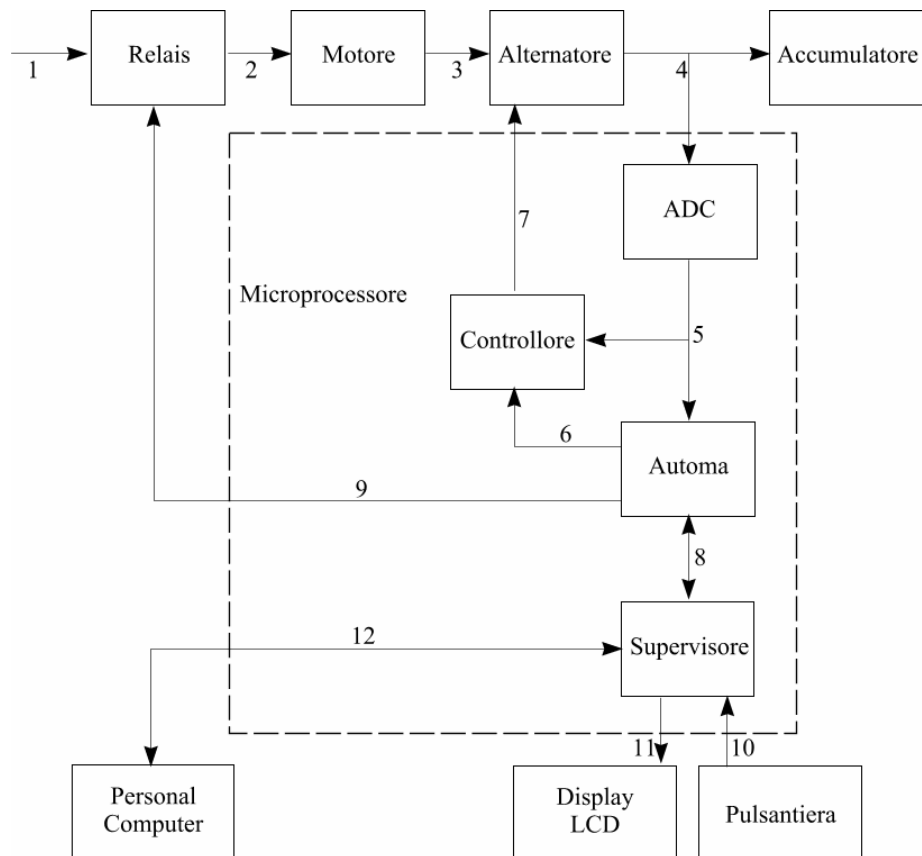
Si è altresì realizzato un software di controllo per personal computer, basato su sistemi operativi Win32, utile per analizzare in tempo reale lo stato del caricabatterie stesso e per monitorare le curve di carica degli accumulatori in carica.

3 Architettura

Il diagramma in figura 1 descrive gli elementi concettuali del progetto, i quali vengono descritti analiticamente qui di seguito:

- **Motore**
Si tratta di un motore elettrico trifase da 380 V, il quale viene acceso prima di iniziare il processo di carica vero e proprio e lasciato girare a vuoto per alcuni secondi al fine di raggiungere la velocità angolare di regime. Durante tutto il funzionamento del caricabatterie, salvo malfunzionamenti, è ragionevole supporre che la velocità di rotazione del motore resti costante.
- **Alternatore**
Si tratta di un alternatore da 30 V, il cui rotore è fissato meccanicamente all'albero motore del motore elettrico. Ovviamente è collegato in serie ad un dispositivo di raddrizzamento opportunamente dimensionato che non è rappresentato in figura e che porta ai morsetti di carica verso l'accumulatore. L'avvolgimento in continua dello statore dell'alternatore pilotato direttamente dall'uscita uscita PWM del microprocessore tramite un semplice circuito di potenza comprendente un fotocopiatore atto a realizzare l'isolamento galvanico (necessario a proteggere lo stampato su cui si trovano i componenti di elettronica digitale) e un transistor di potenza. In tal modo è possibile per il microprocessore agire sull'intensità del campo magnetico dell'alternatore, variando in tal modo la tensione (e quindi la corrente) in uscita.

Figura 1: Schema architetturale del progetto.



- **Relais**
Trattasi di un semplice relais pilotato a 12 V, atto a realizzare un semplice controllo on/off del motore. Il microprocessore provvede a tenere chiuso questo relais nelle sole fasi necessarie (esaurimento del transitorio di avvio del motore stesso, processo di carica), e ad aprirlo in ogni altra situazione (riposo, allarmi, sgancio, fine carica).
- **Accumulatore**
Gli accumulatori per la cui carica questo caricabatterie e' stato progettato sono accumulatori da trazione per veicoli stradali elettrici, composti da 10 elementi ciascuno da 2.35 V.
I parametri utilizzati nel software del dispositivo di controllo del caricabatterie sono stati tarati opportunamente per la carica delle batterie fornite dal committente, modello Dryfit da 24 V con capacità 180 Ah.
- **Microprocessore**
Il componente racchiuso in una cornice tratteggiata rappresenta il microprocessore, comprendente la logica necessaria per eseguire il software (rappresentato come Automa, Controllore e Supervisore), il convertitore analogico/digitale, il modulatore PWM e il gestore dell'interfaccia seriale RS/232.

Il microprocessore scelto è l' ATMEL 90S4433, processore RISC a 8 bit in logica CMOS della serie AVR, con quarzo a 8 MHz e dotato di 32 registri general purpose, RAM da 128 byte, EEPROM da 256 bytes, Flash ROM da 4 K, contatore a 16 bit con prescaler, PWM a 10 bit, convertitore analogico-digitale a 10 bit e 6 canali, UART, 20 piedini su 28 utilizzabili come linee di ingresso/uscita programmabili.

- ADC

Si tratta dell'analog to digital converter integrato nel microprocessore. I dispositivi collegati con il canale di acquisizione della tensione sono stati tarati in modo da raggiungere il fondo scala dei 10 bit (1023) alla tensione di 102.3 V, in modo da avere esattamente un decimo di volt per ogni tacca del range. La lettura della corrente avviene in maniera indiretta leggendo una tensione ai capi di uno shunt, ancora una volta il campo dei 10 bit viene usato per coprire il range da 0 a 102.3 A.

- Controllore

Il software realizzato comprende tre algoritmi, ognuno dei quali progettato e tarato per il controllo di una data fase di carica ("I" a corrente costante, "U" a tensione costante o "i" ancora a corrente costante). Ad un dato istante ovviamente uno solo (oppure nessuno) dei tre controllori è in esecuzione. Essi realizzano il controllo della tensione o della corrente agendo opportunamente sul PWM che eccita lo statore (e quindi il campo magnetico) dell'alternatore.

- Automa

Le scelte vere e proprie del dispositivo di controllo vengono prese e attuate da un automa a stati finiti sincrono realizzato con implementazione a struttura di programma. Maggiori dettagli sull'automa verranno forniti nel paragrafo dedicato all'automa.

- Supervisore

Con questo elemento concettuale, intendiamo indicare quel componente del software embedded il cui ruolo è gestire l'hardware, leggere gli ingressi e gestire le interruzioni necessarie per realizzare la comunicazione seriale e il campionamento ADC.

- Personal Computer

Il personal computer è un elemento puramente opzionale, nel senso che il microprocessore e gli integrati del chipset che lo assistono sullo stampato integrato nel caricabatterie sono funzionalmente sufficienti a realizzare tutte le funzioni richieste dalle specifiche. In altre parole è perfettamente possibile utilizzare pienamente il caricabatterie in tutte le sue funzionalità senza bisogno di un personal computer collegato, tanto più che il sistema di controllo possiede la sua tastiera e il suo display LCD sufficienti per una interazione "amichevole" e completa con l'utente.

Qualora si volesse connettere il personal computer, è stato realizzato un software per sistemi operativi Win32 (Microsoft Windows 9X, ME, 2000) il quale, oltre a realizzare le funzioni genericamente indicate a lezione come "funzioni di godimento dell'operatore", permette di visualizzare campionare, registrare e visualizzare con livelli arbitrari di zoom le curve di carica della batteria in corso di carica, oltre allo stato corrente del dispositivo di controllo.

È altresì possibile salvare i campioni delle curve su file in modo da poterle ricaricare e confrontare fra loro in tempi successivi ed esportare verso programmi esterni (software di rappresentazione grafica quali Gnuplot o fogli elettronici quali Microsoft Excel).

- Display LCD

Si tratta di un display LCD intelligente a matrice di punti retroilluminato, da 2 righe e 16

colonne. Il display integra un controllore Hitachi HD44780, ed è collegato al microprocessore con un opportuno bus. Sul display LCD vengono visualizzati

- Pulsantiera
Si tratta di quattro pulsanti semplici collegati a quattro pin in ingresso del microprocessore e utilizzati per l'input dell'utente, realizzando (ove serve) un antirimbazzo software.

Collegamenti

- 1: linea elettrica di alimentazione in corrente alternata trifase a 380 V 50 Hz;
- 2: alimentazione del motore presente o assente a seconda dell'apertura del relais;
- 3: albero motore che realizza l'accoppiamento meccanico fra motore e alternatore;
- 4: tensione e corrente di carica dell'accumulatore, raddrizzate;
- 5: codifica digitale della tensione e corrente di carica dell'accumulatore;
- 6: parametri del controllore scelti dall'automa a stati finiti;
- 7: tensione di alimentazione del filamento in corrente continua dello statore dell'alternatore, modulato in PWM, da 0 a 120 V;
- 8: linea di comunicazione virtuale per le informazioni di stato e delle uscite, realizzata in software con variabili globali e parametri di chiamata di procedura;
- 9: canale di comunicazione seriale RS/232 a 9600 bps con il personal computer per invio di comandi e la ricezione dello stato;
- 10: bus proprietario della pulsantiera;
- 11: bus digitale proprietario per il controllo del display LCD;

Si vogliono notare i collegamenti 4, 5 e 7 che realizzano l'anello di retroazione per la regolazione in tensione o in corrente a seconda della fase di carica in cui ci si trova.

4 Processo di carica

Il seguente processo di carica è denominato "IUI", o a tre fasi, ed è specificato dalla normativa DIN 41173:

- nella fase "I" l'accumulatore viene sottoposto a carica a corrente costante (nel caso specifico a 30 A). In questa fase è compito del dispositivo di controllo mantenere la corrente costante comportandosi da opportuno regolatore; il termine di questa fase si ha quando l'accumulatore sotto carica raggiunge la tensione di 27 V;
- nella fase "U" l'accumulatore viene sottoposto a carica a tensione costante (nel caso specifico a 27 V). Ancora una volta è compito del dispositivo realizzato fare in modo che la tensione sia tenuta costante al valore desiderato;
- nella fase "i" ("i piccolo") l'accumulatore viene sottoposto ad una corrente di mantenimento di 2 A. Tale fase ha una durata di 15 (?) minuti.

5 Automa

L'automa realizzato è un automa di Moore (le uscite sono legate allo stato), sincrono (viene eseguito regolarmente in maniera temporalmente indipendente dal pervenimento degli eventi), realizzato con controllo a struttura di programma.

L'implementazione a struttura di programma è stata scelta in quanto, in questo caso specifico è la più compatta da realizzare ed empiricamente ho visto che **non era possibile contenere** l'implementazione a struttura dati nei 4K di Flash ROM disponibile.

In ogni caso ho provveduto a specificare azioni di ingresso, cicliche e di uscita per ogni stato, lo spazio a disposizione non avrebbe reso possibile tenere in memoria array di puntatori a procedure e di identificatori di stato prossimo per tutti gli stati e le transizioni specificate.

Gli stati utilizzati sono un soprainsieme degli stati di carica suggeriti dalla specifica DIN 41173 (stato "I", stato "U" e stato "i") e comprendono, oltre ad essi la coppia di stati "pronto" e "spegnimento" (i quali rappresentano l'avvio e la fine della carica) e gli stati necessari per la segnalazione delle anomalie (descritti estesamente di seguito).

In figura 2 è rappresentato il diagramma degli stati implementati, con le seguenti convenzioni:

- gli stati sono identificati da una etichetta composta da una 'S' e seguita da un numero (es. S1, S2, ...);
- le transizioni sono identificate da una etichetta composta da una 'C' e seguita da un numero (es. C1, C2, ...); per chiarezza una transizione che può essere scatenata da due differenti condizioni (in OR logico) è rappresentata sul diagramma come due differenti transizioni;
- alcune transizioni sono scatenate da un ritardo: si intende che la permanenza nello stesso stato (senza che altre transizioni in uscita scattino) per il tempo specificato scateni una transizione. I ritardi sono identificati da una etichetta composta da una 'T' e seguita da un numero (es. T1, T2, ...), in particolare dallo stesso numero dello stato da cui la transizione esce.

Segue l'elenco completo degli stati:

- S0: Pronto
Questo stato rappresenta la situazione di quiete iniziale dell'apparato, quando l'accumulatore da caricare non è ancora stato collegato ai morsetti. In questo stato il relais che alimenta il motore elettrico è aperto (il motore è chiuso) e i valori di tensione e corrente letti ai morsetti di carica sono nulli.
- S1: Accensione
L'automa si porta nello stato di accensione quando rileva che l'accumulatore da caricare è stato collegato ai morsetti di carica (la tensione ai morsetti supera una soglia V_P di valore nominale 18 V, pari ad una tensione minima che qualsiasi batteria scarica ma in buono stato di salute è in grado di raggiungere).
All'ingresso nello stato di accensione viene chiuso il relais che pilota il motore elettrico ma l'eccitazione PWM dello statore dell'alternatore viene tenuta a zero.
Lo stato di accensione è stato pensato per consentire al motore di raggiungere la sua velocità di rotazione di regime intanto che l'alternatore -il cui statore non è eccitato- non provoca carico meccanico.
L'unica transizione in uscita da questo stato dovuta allo scadere del ritardo T_1 , al termine del quale si suppone che il transitorio di avvio del motore sia concluso.

- S2: Fase I
Fase di carica dell'accumulatore a corrente costante. Durante questa fase viene attivato il regolatore tarato per la fase I, il cui scopo è regolare l'eccitazione PWM dello statore dell'alternatore in modo da mantenere la corrente in ingresso alla batteria costante ad un valore I il cui valore è impostabile dall'utente dall'utente fino a 60 A.
Questa fase continua fino a quando la tensione rilevata sull'accumulatore sale fino a raggiungere il valore prefissato V_D (il cui valore nominale è 28 V); a questo punto si passa in fase U.
- S3: Fase U
Fase di carica dell'accumulatore a tensione costante. Viene attivato un regolatore del tutto simile a quello utilizzato nella fase precedente, ma progettato per tenere costante la tensione ad un valore U , impostabile dall'utente, normalmente posto a 27.5 V. Questa fase continua fino a quando che la corrente rilevata sull'accumulatore scende fino a raggiungere il valore prefissato I_D , di valore nominale 24 A. A questo punto si passa in fase i.
- S4: Fase i
Si tratta di una fase di carica di mantenimento ad una corrente di 2 A, della durata impostabile dall'utente (ritardo T4), nominalmente di 15 minuti. Ancora una volta si utilizza un regolatore a corrente costante, ma con parametri ovviamente differenti da quelli impostati per la fase I.
- S5: Spegnimento
In questo stato si giunge al termine della fase i, e quindi la carica dell' accumulatore è stata completata, il sistema non intraprender più alcuna azione fino a nuovo ordine dell'utente. È ora possibile per l'utente scollegare i morsetti e, se lo desidera premere un tasto che riporta l'automa in fase di Pronto.
- S6: Allarme V
Si giunge in questo stato quando la tensione misurata sull'accumulatore superiore ad una tensione massima di emergenza V_E , pari a 28.5 V. Questo stato è pensato per proteggere l'accumulatore da una sovratensione che può danneggiare il gel degli elementi degli accumulatori e per allarmare l'utente in modo che venga messo a conoscenza dell'anomalia. Da questo stato è infatti possibile tornare allo stato di pronto solo dopo la tacitazione da parte dell'utente. Si tratta in sostanza di uno stato del tipo "allarme attivo e da tacitare". A causa della particolare gravità dell'allarme, non è prevista la possibilità del rientro dell'allarme né uno stato di "allarme rientrato ma non tacitato". Con la pressione di un tasto è possibile tornare in stato di pronto.
- S7: Sgancio
L'automa si porta nello stato di sgancio quando si rileva una corrente di carica nulla, condizione che indica una anomalia causata dall'utente che scollega l'accumulatore dai morsetti, oppure dallo stato di scarsa salute (solfatazione) dell'accumulatore che non accetta corrente, oppure da un malfunzionamento generico. In ogni caso necessario spegnere il motore elettrico e disalimentare l'eccitazione PWM dello statore dell'alternatore. Ancora una volta è richiesta la tacitazione dell'utente prima di tornare nello stato di pronto.
- S8: Fase I allarmata
Se la fase I si protrae oltre il tempo impostato dall'utente (ritardo T2), l'automa si porta in questo stato, che allerta l'utente sullo stato di salute imperfetta dell'accumulatore pur concedendo una sorta di "tempo supplementare" per la carica a corrente costante. Qualora

la tensione della batteria non salisse oltre la soglia indicata in un ulteriore tempo (ritardo $T_8 =$ ritardo T_2 per costruzione), ci si porta in stato di spegnimento, altrimenti si prosegue come se l'anomalia non si fosse mai verificata.

Figura 2: Diagramma degli stati.

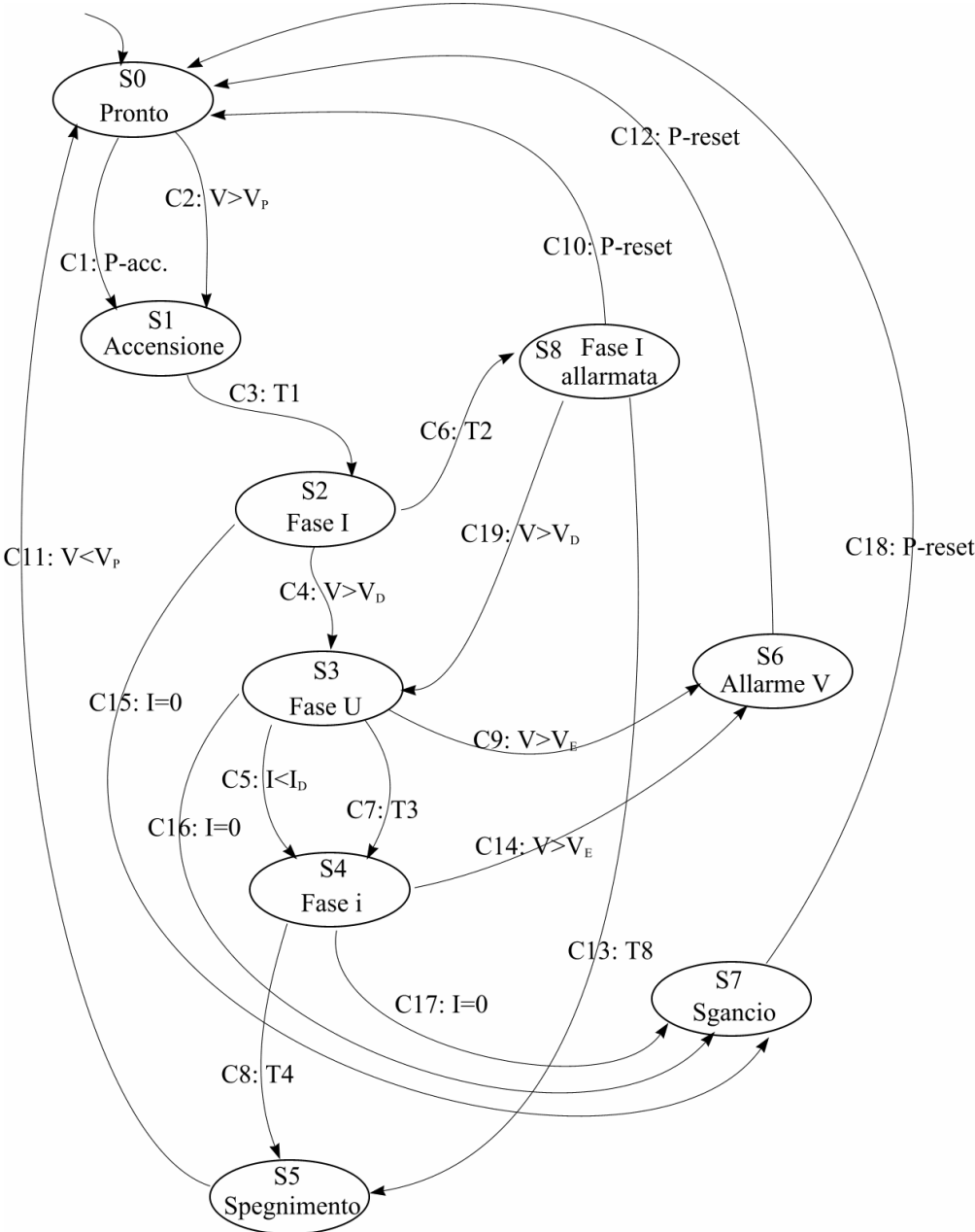
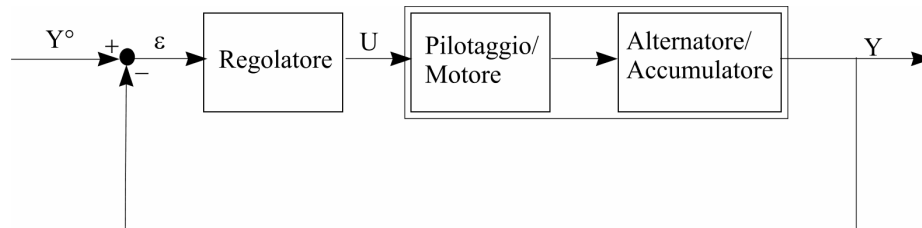


Figura 3: Schema semplificato del sistema di controllo realizzato.



Parametri

- T_1 : tempo richiesto per l'esaurimento del transitorio di avvio del motore elettrico, valore nominale = 5 secondi;
- V_D : tensione desiderata per il completamento della fase I;
- I_D : corrente desiderata per il completamento della fase U;
- T_2 : tempo di permanenza in stato 2, timeout;
- V_E : tensione di emergenza che non deve mai essere oltrepassata per la salvaguardia della salute dell'accumulatore
- T_3 : tempo di permanenza in stato di carica I allarmata senza raggiungimento tensione desiderata V_D ;
- V_P : tensione di presente

6 Problema di controllo

Il mantenimento della tensione o della corrente (a seconda della fase scelta) è stato realizzato mediante un opportuno regolatore ad anello chiuso.

In figura 3 è rappresentato uno schema semplificato del sistema di controllo realizzato. L'uscita desiderata, indicata nello schema con la lettera Y^o sarà di volta in volta il valore desiderato per la corrente I, per la tensione U o per la corrente i (valori tutti impostabili dall'utente e salvati nella EEPROM interna del processore).

La variabile rappresentata in figura con la lettera U (da non confondere con la tensione U della fase di carica omonima) è la variabile di controllo, in particolare il valore di eccitazione PWM a 10 bit (da 0 a 1023) che controlla direttamente il filamento dello statore dell'alternatore.

Si noti che tutta la teoria di controllo automatico lineare NON può essere sfruttata appieno nella progettazione di questo controllore in quanto il complesso indicato con la scatola "Pilotaggio/Motore" rappresenta esso stesso una relazione ingresso uscita NON lineare (durante le prove ha mostrato anzi un comportamento dotato di isteresi).

È stato implementato un regolatore digitale di tipo PI (azione proporzionale e integrale), con alcuni accorgimenti atti a tener conto delle seguenti condizioni particolari:

- mancanza del supporto ai numeri in virgola mobile;
- realizzazione a tempo discreto;

- quantizzazione delle variabili di controllo e sotto controllo;

In questo di controllare la derivata della variabile di controllo è determinata dal prodotto fra una costante di guadagno (da determinare sperimentalmente, associata al tempo di risposta del sistema) e l'errore (scostamento fra valore desiderato e valore attuale della variabile sotto controllo, rappresentata in figura con la lettera ε).

Più esattamente, essendo il controllore a tempo discreto, ad ogni occorrenza temporale del ciclo di controllo (eseguito a periodi regolari) la variabile di controllo viene incrementata o decrementata di una quantità determinata proporzionalmente in base all'errore:

$$U(n+1) = U(n) + \Delta U \quad (1)$$

$$\Delta U = \frac{\mu \cdot \varepsilon(n)}{f} = \frac{\mu}{f} (Y^o - Y(n)) \quad (2)$$

L'implementazione dei controllori non è stata troppo agevole a causa della mancanza del supporto ai numeri in virgola mobile, infatti si rende immediatamente evidente che, avendo a disposizione la sola matematica intera, una volta fissato il valore del guadagno μ (opportunamente diviso per una costante f che tenga conto della frequenza di esecuzione del ciclo di controllo), vi sarà sempre un range di valori della variabile ε , nell'intorno dello zero, il quale pur rappresentando uno scostamento non nullo, si trova a far risultare un prodotto inferiore in valore assoluto all'unità e quindi, in matematica intera, nullo.

La conseguenza diretta di ciò è che in un certo intorno dello zero per l'errore, pari a

$$-\frac{f}{\mu} < \varepsilon < \frac{f}{\mu}$$

(che a frequenze elevate può essere anche grande), l'azione di controllo si congela e il controllore perde ogni efficacia, perdendo così anche la possibilità di raggiungere il valore desiderato.

Si è allora proceduto a distinguere, all'interno dell'algoritmo di controllo, due casi, nel primo dei quali la ΔU è superiore all'unità (l'errore è grande, anche ΔU è grande, non emergono problemi numerici), mentre nel secondo ΔU è inferiore all'unità e viene rappresentato, per comodità, con il più vicino numero nella forma $1/n$, e la variabile sotto controllo viene incrementata o decrementata a seconda del bisogno di una unità, solamente uno ogni n cicli di controllo.

È ragionevole dire che fintanto che i valori sono abbastanza grandi da poter trascurare l'errore dovuto all'uso degli interi al posto dei reali il controllore si comporta in modo proporzionale continuo, mentre quando l'approssimazione non è più valida il controllore si avvicina ad essere un controllore di tipo on/off (detto anche MB/2).

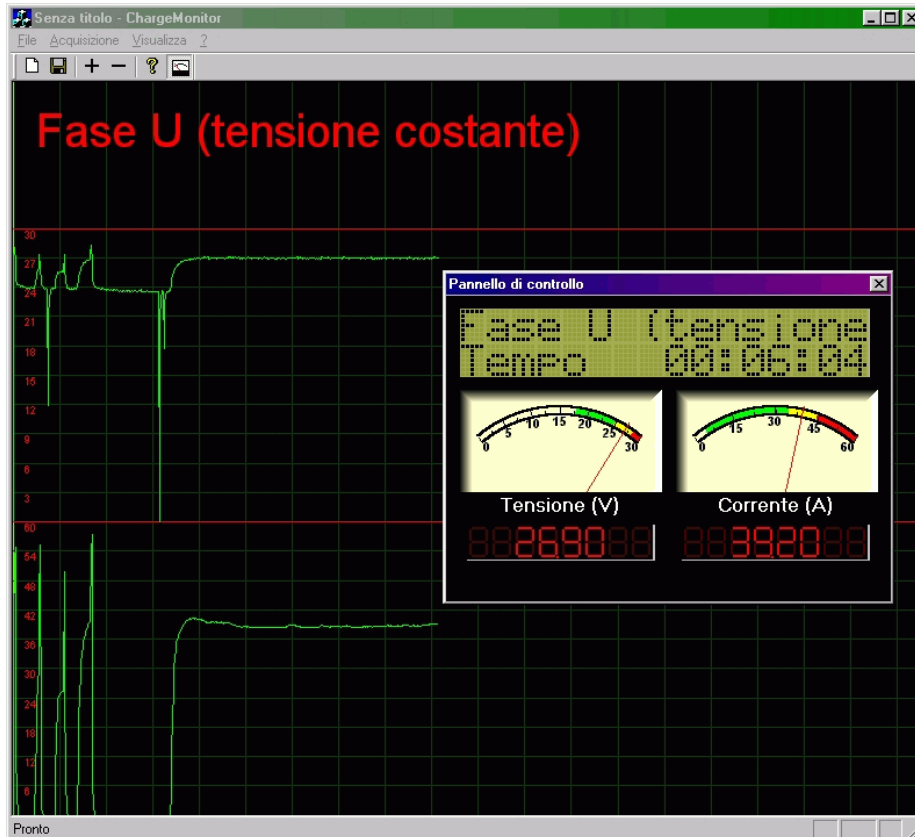
Al fine di ottenere un avviamento morbido e di evitare problemi di sicurezza e di sovratensione, il valore iniziale della variabile di controllo U (eccitazione PWM) viene fissato a zero. Sebbene questa scelta allunghi leggermente il tempo di raggiungimento del regime, una volta scelti i parametri opportuni per il guadagno, ho verificato sperimentalmente che il transitorio si esaurisce comunque in pochi secondi e i vantaggi sotto il profilo della sicurezza e della minore sollecitazione dei componenti meccanici sono irrinunciabili.

Le prestazioni ottenute del sistema di controllo, come detto, sono molto buone: errore statico nullo, tempo di risposta breve e oscillazioni dovute alla quantizzazione del tutto trascurabili.

7 Software per PC

In figura 4 è rappresentata una schermata del programma realizzato per Windows, atto a monitorare lo stato e l'andamento nel tempo del dispositivo di controllo.

Figura 4: Schermata del programma realizzato.



Come si può vedere, il programma si compone di due finestre, la prima delle quali mostra il grafico con l'andamento di tensione e corrente nel tempo (è possibile effettuare operazioni di zoom e di scrolling) mentre la seconda mostra una sorta di replica del display presente sul dispositivo stesso, che rappresenta lo stato corrente, il tempo di carica e i valori di tensione e corrente istantanei.

Il protocollo di comunicazione fra personal computer e dispositivo è molto semplice e si compone di tre sole primitive:

- la primitiva di interrogazione di versione, che permette di ottenere una stringa di identificazione del dispositivo; si realizza inviando al dispositivo un singolo carattere ASCII '?', e causa la risposta da parte del dispositivo stesso con la stringa "CA810, v0.2", seguita da un carattere ritorno carrello e avanzamento riga;
- la primitiva di interrogazione di stato e tempo di carica, consistente nell'invio di un singolo carattere 'S', che causa il ritorno di una stringa il cui primo carattere è una cifra da '0' a '8', che indica lo stato corrente nel diagramma degli stati (da S0 a S8) e i cui caratteri successivi sono la codifica decimale del numero di secondi di carica della batteria;
- la primitiva di interrogazione sui valori di tensione e corrente, consistente nell'invio di un

singolo carattere 'G', che causa il ritorno di una stringa contenente le codifiche decimali del numero di volt e di ampere misurati, separate da uno spazio.

Essendo il protocollo molto semplice, in particolare avendo ridotto i comandi ad un solo carattere, e facendo l'ipotesi che il personal computer sia veloce a sufficienza, è risolto ogni problema di controllo di flusso in quanto il dispositivo si trova ad avere tutto il tempo di processare il comando fra un carattere e l'altro, senza dover gestire la linea CTS del canale RS/232, potendola tranquillamente cortocircuitare sulla linea RTS.

8 Modello implementativo

In questa sezione si esamina la struttura dell'implementazione realizzata nel software a bordo del microcontrollore, alla luce del modello presentato nel Cap. 12 del manuale del corso.

8.1 Struttura generale

- **Gestione delle informazioni digitali**

Le informazioni digitali da acquisire in questo progetto (che non ricadano in altre categorie strutturali di questa classificazione) sono solamente le pressioni dei tasti del tastierino collegato al dispositivo. Tale acquisizione viene eseguita in maniera banale con la lettura del particolare bit della porta C del microcontrollore a cui è associato il pin che ha collegato il tasto. Per buon progetto, tali dettagli sono nascosti in quattro opportune macro del linguaggio C:

```
#define PULSANTE_A() (PINC & 0x04)
#define PULSANTE_B() (PINC & 0x08)
#define PULSANTE_C() (PINC & 0x10)
#define PULSANTE_D() (PINC & 0x20)
```

Ove necessario è applicato l'antirimbazzo software.

Le informazioni da mantenere e gestire in generale sono le informazioni di stato (comprese a temporizzazioni allarmi) associate allo stato in cui l'automa si trova, che saranno meglio descritte nell'apposito paragrafo del modello funzionale.

- **Gestione delle misure**

Sotto questa categoria possono essere enumerate i dettagli di implementazione qui di seguito elencati.

Ingresso di segnali analogici: si acquisiscono i segnali di tensione e corrente ai morsetti collegati alla batteria; la taratura della misura è realizzata per semplicità in maniera solo hardware, viene inoltre applicato un filtraggio a media mobile con ampiezza del campione = 2, empiricamente dimostratasi sufficiente, atta ad eliminare gli effetti della granularità della rappresentazione intera.

La conversione analogica digitale avviene per mezzo dell' ADC integrato nel microprocessore, il quale viene inizializzato per mezzo dell'istruzione

```
ADCSR = 0xCF; // On ADC Interrupt-Flag, ADC Enable, Prescaler=128
```

e armato ciclicamente con frequenza di 4 Hz all'interno del gestore dell'interrupt del timer, nel seguente modo:

```
interrupt [TIMER0_OVF_vect] void ricarica(void)
{
    TCNT0 = TOL;
    // ricarica il valore TOL, che causa un interrupt @ 40 Hz
    if (++crono.ticks % 10) return;
    // divido la frequenza per dieci
    ADCSR|=0x40;
    // ADCSR bit 6: ADSC -- start conversion (v. pag. 56 databook)
    ...
}
```

Al termine della conversione si scatena un ulteriore interrupt dedicato, il cui gestore estrae i valori una volta del primo canale (tensione), una volta del secondo (corrente), rendendo quindi disponibili nuovi valori ciclicamente (in modo interlacciato) ad una frequenza di 2 Hz. Tale interlacciamento viene implementato cambiando il valore dell'ultimo bit (MUX0) del registro ADMUX che seleziona il canale ADC attivo.

```
void interrupt[ADC_vect] GetValue(void)
{
    char aux1,aux2;
    char c;
    aux1=ADCL;
    aux2=ADCH;

    if (ADMUX==0)
    {
        vecchia_tensione = tensione;
        ADMUX++;
        tensione = aux1+256*aux2;
        tensione += vecchia_tensione;
        tensione /=2;
    } else {
        vecchia_corrente = corrente;
        corrente = aux1+256*aux2;
        corrente += vecchia_corrente;
        corrente /=2;
        ADMUX=0;
    }
}
```

Nello stesso frammento di codice si può anche notare l'implementazione (peraltro banale) di un **filtraggio software a media mobile**. Si è prestata attenzione ad evitare problemi di overflow numerici, utilizzando per i calcoli della media valori a 16 bit mentre l'ADC genera

campioni di cui solo 10 bit sono significativi (tutti i bit di ADCL e i 2 bit meno significativi di ADCH, v. databook pag. 57).

Sotto questa categoria ricadono anche l'**implementazione del regolatore PI** di cui si è già discusso ampiamente, e l'**uscita di un segnale analogico** di controllo, in particolare la generazione di tale segnale in PWM, realizzata in maniera semplice con il generatore PWM integrato nel microprocessore, inizializzato come segue:

```
TCCR0=0x5;    // prescaler 1024
TCCR1A=0x83;  // PWM 10 bit - v. pag. 32 databook
```

e utilizzato per mezzo di una macro C:

```
#define set_pwm(_level_) { OCR1H = _level_ / 256; OCR1L = _level_ % 256; }
```

; opportune precauzioni di tipo numerico sono prese nel regolatore al fine di evitare un overflow oltre i valori rappresentabili dai 10 bit del registro di controllo del PWM, implementando una saturazione.

- **Gestione dell'operatore** La gestione dell'operatore è in parte demandata al software che gira su personal computer a cui il microcontrollore, è collegato, nondimeno vi sono alcune funzioni di rilevante importanza che vengono svolte localmente e in modo interattivo, quali la regolazione dei parametri impostabili, i comandi di tacitazione degli allarmi e di ripartenza.
- **Gestione delle comunicazioni**
Il dispositivo comunica in maniera digitale verso il display LCD (in sola uscita) e verso il personal computer. Ci focalizziamo sui dettagli a basso livello di quest'ultima comunicazione (visto che quelli ad alto livello sono già stati esaminati): la gestione viene effettuata con un approccio interrupt-driven implementando un semplice interprete dei messaggi all'interno del gestore di interrupt legato alla ricezione di un byte da parte dell' UART integrato nel processore:

```
interrupt [UART_RX_vect] void rxbyte(void) //RS232
{
    switch(UDR)
    {
        case '?' : serial_fputs(id_versione);
                  break;
        case 'G' : outwordser(tensione,1);
                  SERIAL_PUTCHAR(' ');
                  outwordser(corrente,1);
                  break;
        case 'S' : SERIAL_PUTCHAR('0'+stato_corrente);
                  outwordser(crono.secondi,0);
                  break;
    }
    SERIAL_PUTCHAR(13);
    SERIAL_PUTCHAR(10);
}
```

. L'uso dell'UART è del tutto simile (cambiano i dettagli architetturali) a quello descritto nel paragrafo 9.4 del testo. L'UART viene pilotato con le due seguenti macro C:

```
#define SERIAL_GETCHAR(_ch_) { _ch_ = UDR; }  
#define SERIAL_PUTCHAR(_ch_) { while(!(UCSRA&0x20)); UDR=_ch_; }
```

Le quali operano in ingresso o in uscita sul registro UDR, eventualmente controllando lo stato del bit 5 del registro UCSRA (Bit 5 - UDRE: UART Data Register Empty, v. pag. 47 databook). L'inizializzazione dell'UART avviene invece con le istruzioni:

```
UBRR=51; // 9600 baud con CK @ 8 MHz, errore dello 0.2%  
UCSRB|=0x9a; // Bit 7 - RXCIE: RX Complete Interrupt Enable  
// Bit 4 - RXEN: Receiver Enable  
// Bit 3 - TXEN: Transmitter Enable
```

(vedi pagg. 48 e 49 databook).