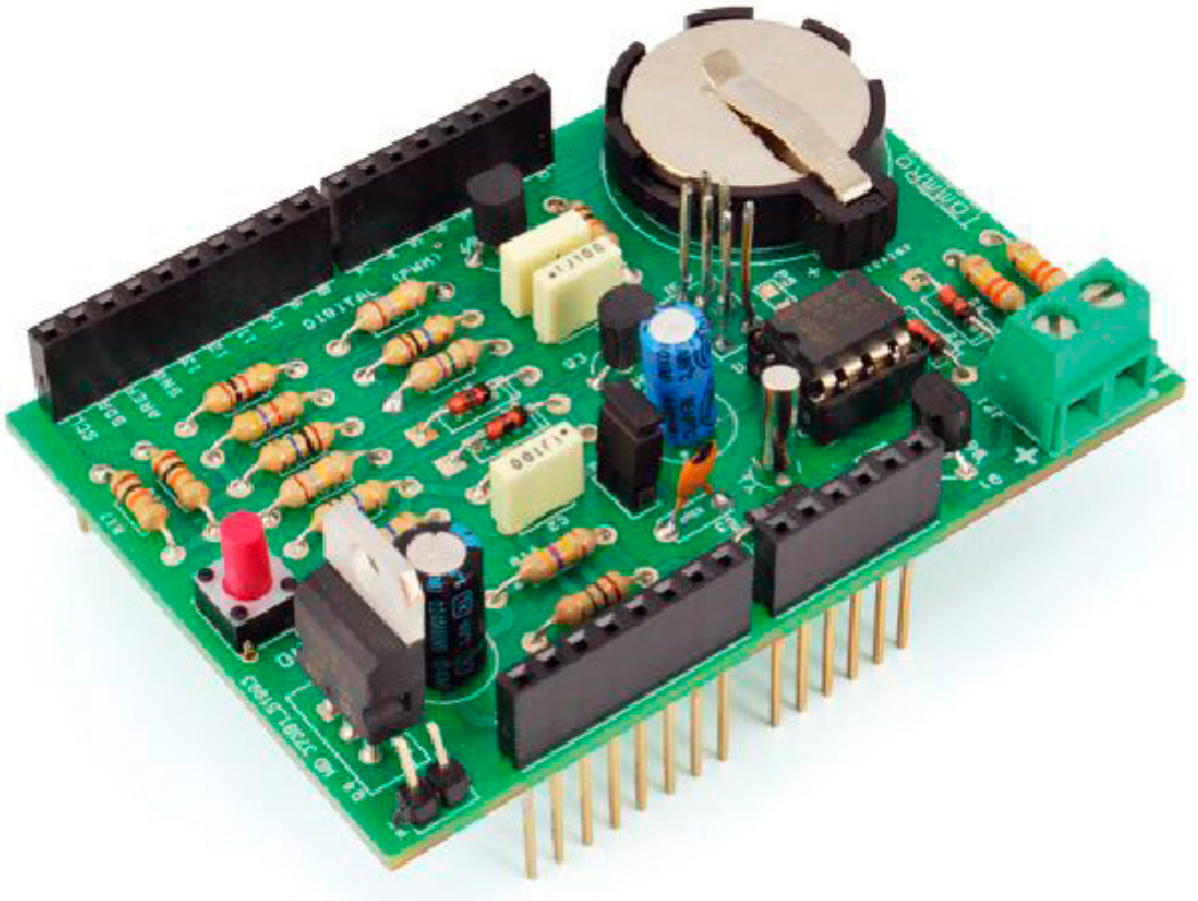
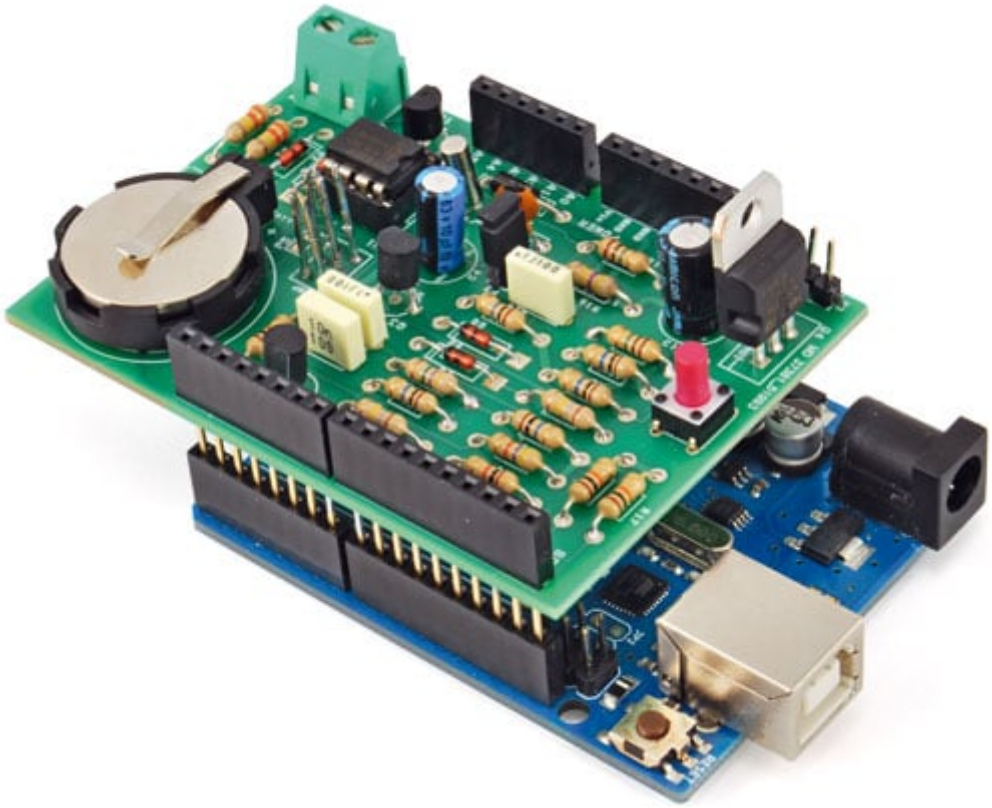
Shield batterie per Arduino - in kit da saldare

Prezzo: 15.98 €

Tasse: 3.52 €

Prezzo totale (con tasse): 19.50 €





Shield per Arduino Uno, predisposto per ricevere in ingresso (attraverso una morsettiera) la tensione di alimentazione proveniente da una batteria (compresa tra 5 e 12 volt) e, sfruttando una particolare funzionalità del chip PCF8593T (un clock/calendar utilizzato generalmente come orologio e calendario), permette di accendere il sistema ad intervalli fissi selezionabili da software a seconda delle condizioni riscontrate. Permette al software Arduino di scegliere quando “essere risvegliato” ed, una volta acceso, attraverso lo stato di un output digitale, per quanto tempo rimanere acceso ed infine quando spegnersi. Lo shield comprende al suo interno il chip PCF8593T, quindi può essere utilizzato anche come shield RTC per fornire ad Arduino le funzionalità tipiche di un orologio e calendario. Lo shield utilizza due pin digitali (D6 e D7) per la gestione del sistema di accensione; la porta I²C per la comunicazione/programmazione del componente PCF8593T; è prevista la possibilità (tramite ponticello) di usare il pin analogico A0 per leggere la tensione di alimentazione proveniente dalla batteria in modo da identificare il relativo livello di carica. Inoltre è presente un pulsante per accendere Arduino anche durante i tempi di spegnimento previsti (chiaramente la condizione andrà prevista in fase di scrittura software) ed un jumper che, se inserito, permetterà di mantenere Arduino sempre acceso (possibilità da utilizzare per particolari casi; magari collegato a un relé proveniente da una seconda scheda per realizzare particolari applicazioni). **N.B.** Arduino non è compreso.

Funzionamento

- il sistema è spento da un determinato istante temporale;
- ad un certo punto l'orologio risveglia Arduino; quest'ultimo decide di rimanere sveglio e passa ad analizzare il proprio stato interno nonché quello di eventuali pin/sensori esterni;
- terminata l'analisi degli input ed aggiornamento delle uscite, Arduino programma l'orologio indicando l'istante in cui dovrà essere risvegliato ed in seguito decide di spegnersi;
- per tutto il restante tempo il sistema rimane spento ed alla riaccensione successivo tutto il meccanismo viene ripetuto.

Libreria Arduino Battery Shield

Come visto nell'articolo, per il corretto funzionamento del sistema è necessario che tutto il software venga scritto avendo in mente come viene realizzato il meccanismo di accensione e spegnimento dell'elettronica. Per facilitare l'integrazione da parte degli utilizzatori finali abbiamo realizzato una libreria software che implementa le funzionalità basilari. Vengono definite sia le funzioni di "basso livello" tra cui inizializzazione del sistema; accensione, auto alimentazione e spegnimento di Arduino; lettura del valore analogico della tensione di alimentazione; sia quelle di "medio livello" tra cui avvio del chip PCF8593T e relativa programmazione degli istanti di accensione; sia quelle di "alto livello" tipiche di un componente orologio/calendario per cui programmazione e lettura data e ora; verifica del capodanno (incremento dell'anno); passaggio automatico dall'ora legale a solare e viceversa, ecc. Come vedremo meglio più avanti analizzando l'esempio di programmazione, utilizzando queste funzioni il programmatore non si deve preoccupare della gestione diretta dell'hardware ma avrà una interfaccia software che se ne fa carico e nasconde tutti i relativi dettagli. La libreria definisce un oggetto "Battery" avente i seguenti metodi pubblici (cioè utilizzabili nel programma Arduino): - void begin(): inizializza i pin D6 e D7 come input ed output digitali ed inoltre avvia l'orologio se questo viene rilevato fermo; - void powerON(): abbassa il pin D7 in modo che Arduino possa rimanere acceso; - void turnOFF(): al contrario di powerON(), alza il pin D6 in modo che Arduino possa spegnersi; - bool onFromButton(): nell'analisi dello schema hardware abbiamo visto che è presente un pulsante per accendere Arduino bypassando la programmazione dell'orologio; per distinguere se l'accensione è stata effettuata dal PCF8593T oppure dalla pressione del pulsante si utilizza il pin digitale D6 di Arduino. Richiamando la funzione onFromButton() si testa lo stato di D6 e di conseguenza si identifica se l'accensione deriva dal pulsante (risultato della funzione true) oppure dall'orologio (risultato false); - float getBatteryVoltage(): come visto analizzando lo schema elettrico dello shield, è possibile selezionare di utilizzare il pin analogico A0 di Arduino per rilevare la tensione di alimentazione della batteria. Questa funzione può essere utilizzate per leggere tale valore; la funzione esegue già tutte le conversioni e tiene conto delle varie cadute di tensioni dovute ai diversi componenti (in particolare il diodo D15) e ritorna in formato float (numero a virgola mobile) la tensione della batteria; - void startSecAlarmPCF8593, void startMinAlarmPCF8593, void startHourAlarmPCF8593: con queste funzioni è possibile selezionare l'istante del prossimo evento di accensione. In particolare è possibile specificare (prendendo come riferimento l'istante di esecuzione delle funzioni) rispettivamente per quanti secondi, minuti oppure ore Arduino debba rimanere spento prima di essere risvegliato dall'orologio. Le funzioni accettano come ingresso un parametro di un byte i cui valori validi sono solo quelli compresi tra 1 e 99 estremi inclusi; - void writeClockANDDataPCF8593(): come detto il componente PCF8593T oltre che essere utilizzato per determinare gli istanti di accensione, può essere usato anche come RTC e orologio/calendario. Questa funzione permette di programmare la data e ora corrente (passati come parametri ore, minuti, secondi, giorno, mese ed anno). La funzione determina e programma in automatico il giorno della settimana e se è attiva l'ora solare o legale; - void readClockANDDataPCF8593(): al contrario della precedente funzione, permette di leggere la data e ora corrente indicata dal chip PCF8593T. Come calendario vengono ritornati il giorno, mese e anno più il giorno della settimana; come orologio vengono ritornati l'ora, i minuti, i secondi e i centesimi di secondo; - bool checkProgrammedPCF8593(): permette di testare se l'orologio/calendario del chip PCF8593T sono stati avviati e programmati. In caso affermativo ritorna true; in caso negativo ritorna false (in questo caso andrà avviato richiamando l'opportuna funzione); - bool checkHourLegPCF8593(): permette di verificare se per un determinato giorno (identificato da giorno della settimana, giorno e mese) debba essere attiva l'ora solare o legale (utile per eseguire in automatico il relativo passaggio); - void checkNewYear(): testa se il calendario è passato dal capodanno ed, in caso affermativo, aggiorna (incrementa di 1) il numero dell'anno; - byte getDayOfWeek(): determina il giorno della settimana di una data (identificata da giorno, mese ed anno); - bool YearBisestile(): determina se un determinato anno è bisestile.

PRIMO FIRMWARE ARDUINO Per capire il funzionamento e la logica di programmazione da utilizzare con il battery shield, vediamo due esempi di codice Arduino, il primo dei quali è riportato nel Listato 1. Il software, semplicemente programma il battery shield affinché Arduino rimanga

Documentazione e link utili

- ***Libreria e Sketch***