Progetto

Sistemi Embedded

Federico Guida, Giovanni Raccuglia, Alessio Princiotta

Università degli Studi di Palermo

Sommario

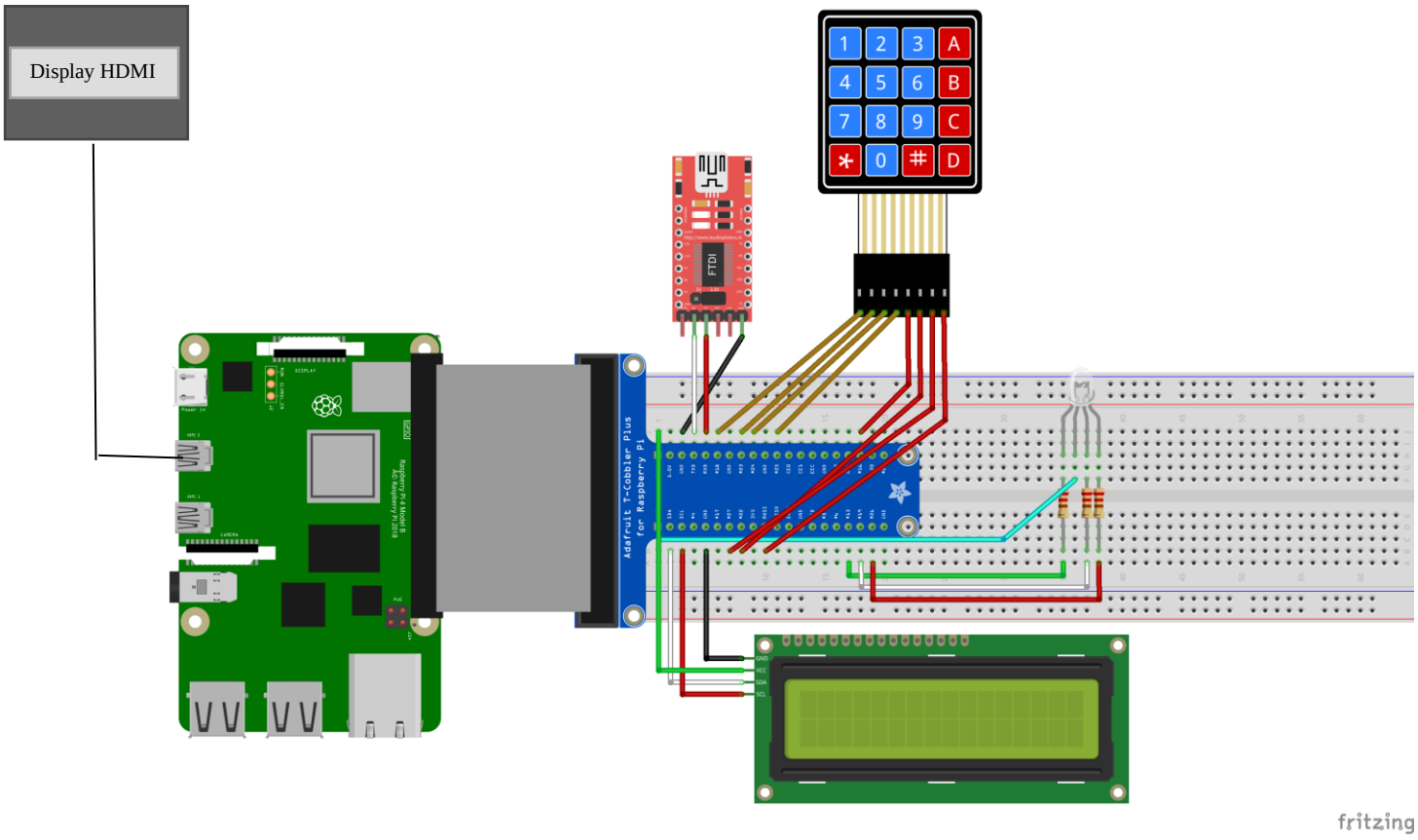
Sunto

Il progetto è stato realizzato per il corso di Sistemi Embedded dell'Università degli Studi di Palermo tenuto dal docente Daniele Peri. Il corso prevede che lo studente acquisisca i concetti necessari per la comprensione della struttura dei sistemi embedded e inoltre acquisisca una conoscenza dei linguaggi di programmazione dedicati a quest'ultimi.
Lo scopo di questo progetto consiste nel realizzare un sistema di sicurezza domestico tramite l'utilizzo di un Raspberry Pi 4B. Il sistema assicura all'utente semplice un'autenticazione tramite l'ausilio di una password, mentre all'utente amministratore è consentito l'accesso a una serie di funzionalità che hanno lo scopo di gestire gli altri utenti. Inoltre, è compresa nel sistema una funzione che permette di gestire lo storico degli accessi.
La seguente documentazione descrive la struttura, il funzionamento e l'uso del sistema sviluppato. La documentazione è costituita di tre macrosezioni:

- *Hardware*, sezione che descrive i componenti che sono utilizzati per il corretto funzionamento del sistema.
- *Ambiente di sviluppo*, sezione che descrive gli strumenti software deputati al caricamento ed esecuzione del codice.
- *Software*, sezione che descrive i moduli sviluppati e il processo decisionale che hanno portato alla loro forma definitiva.

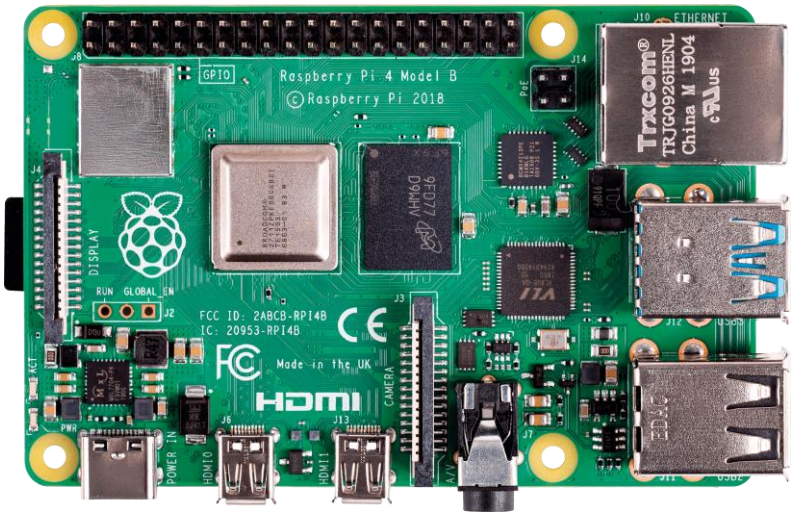
Hardware

Nello schema riportiamo lo stato di collegamento delle componenti hardware utilizzate per il corretto funzionamento del sistema. Lo schema è stato realizzato grazie all'ausilio di un software esterno Fritzing, che permette la realizzazione di schemi professionali per i collegamenti di un sistema embedded.



Nella parte sottostante verranno analizzate singolarmente le componenti.

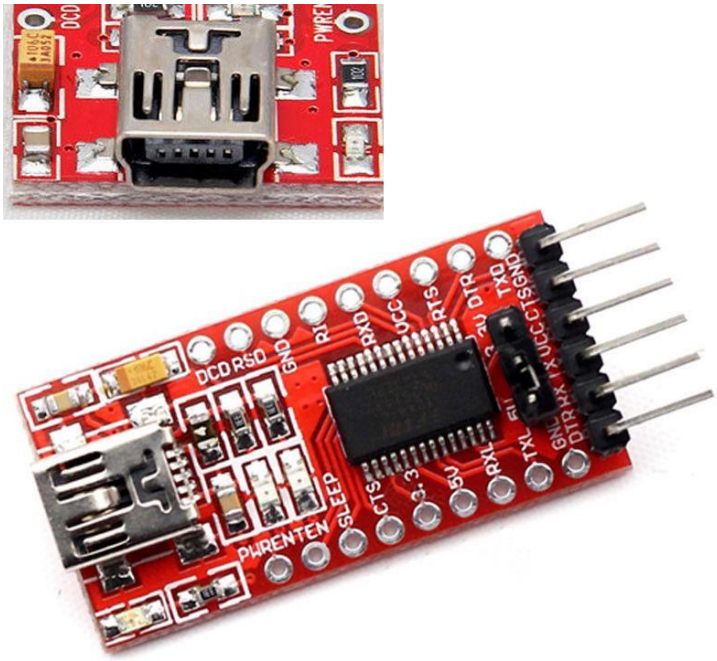
Raspberry Pi 4B



Il target di questo progetto prevede l'ausilio di un Raspberry Pi 4B, una general-purpose Single-Board Computer. Quest'ultimo possiede le seguenti caratteristiche:

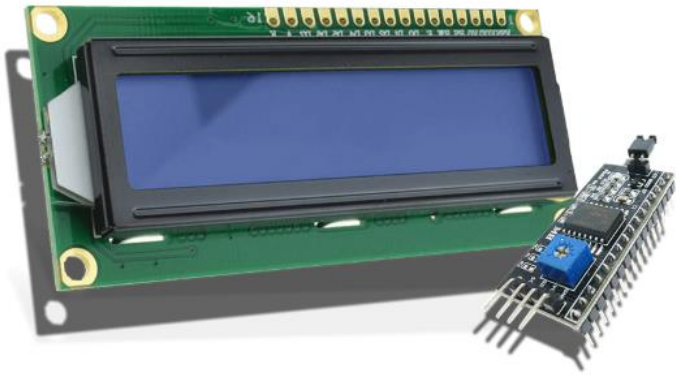
- una CPU ARM Cortex-A72 quad-core a 64-bit da 1,5 GHz
- 2 GB di LPDDR4 SDRAM (nostra configurazione)
- Gigabit Ethernet
- Rete wireless dual-band 802.11ac
- Bluetooth 5.0
- Due porte USB 3.0 e due porte USB 2.0
- Supporto per doppio monitor, con risoluzione fino a 4K
- Grafica VideoCore VI, con supporto OpenGL ES 3.x
- Decodifica hardware 4Kp60 del video HEVC
- Retrocompatibilità con i precedenti Raspberry Pi

FTDI FT232RL



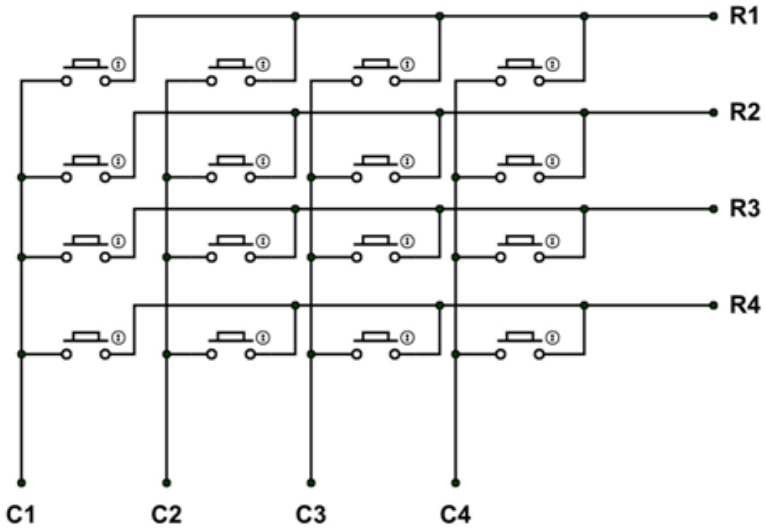
La FT232RL è un'interfaccia da USB a UART seriale. Per mezzo di questa periferica è possibile inviare dati da e verso il dispositivo target. Il modulo FTDI è collegato al Raspberry Pi 4B tramite la porta UART1. La periferica è dotata di un buffer di ricezione di 128 byte e da un buffer in trasmissione di 256 byte, che garantiscono robustezza in trasmissione fino a 3Mbaund/s. Collegandola inoltre alla porta usb, il PC la riconoscerà come dispositivo UART e permetterà la comunicazione.

LCD 16x02



I moduli LCD sono molto comunemente utilizzati nella maggior parte dei progetti embedded, il motivo è il prezzo economico, la disponibilità e la facilità di programmazione. È chiamato così perché ha 16 colonne e 2 righe. Ci sono molte combinazioni disponibili come, 8 × 1, 8 × 2, 10 × 2, 16 × 1, ecc. La versione la più utilizzata è l'LCD 16 × 2. Quindi, avrà (16 × 2 = 32) 32 caratteri in totale e ogni carattere sarà composto da 5 × 8 Pixel. Può mostrare numeri, lettere simboli, codici ASCII, ecc. Dispone di un potenziometro sul retro per regolare il contrasto. Per gestire l'utilizzo di questo display viene utilizzato un ulteriore dispositivo che permette di utilizzare l'interfaccia I2C del dispositivo target. Il dispositivo utilizzato è HD44780**,** che è montato sul retro del modulo LCD stesso. La funzione dell'interfaccia I2C è quella di prendere i comandi e i dati dalla MCU, processarli e mandarli al display per visualizzare le informazioni all'interno dello schermo LCD.

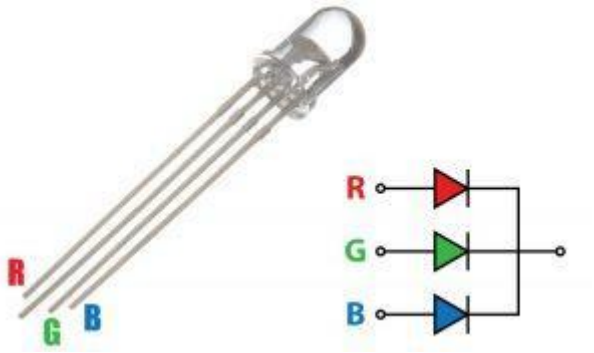
Keypad Matrix 4x4



Il dispositivo seguente è una periferica di input in cui i tasti sono disposti in formato matriciale. La sua struttura righe e colonne, permette di verificare facilmente con l'ausilio di un microcontrollore quale dei tasti è stato digitato tramite la lettura dei pin di output del tastierino stesso. Il tastierino seguente ha 16 tasti e presenta un'uscita con 8 pin che permettono la lettura della matrice righe/colonne.

Nella figura a sinistra viene presentato lo schema elettrico del tastierino.

Led RGB



Il led RBG sono led capaci di produrre tre differenti lunghezze d'onda, rosso, verde e blu. La mescolanza dei tre colori dà luogo ad una luce di un determinato colore che dipende dall'intensità di ciascuno dei tre canali primari.

Ambiente di sviluppo

PijFORTHOS è un sistema operativo bare-metal per Raspberry Pi, basato su Jonesforth-ARM, un interprete Forth sviluppato per architettura ARM. La versione utilizzata per il progetto è stata modificata e adattata per il suo utilizzo dal docente universitario Daniele Peri. Grazie all'utilizzo della UART e di alcuni comandi da terminale possiamo comunicare e programmare con questo linguaggio di programmazione interattivo con il nostro dispositivo target. Ad esempio per il progetto è stato utilizzato Minicom e Picocom.
Minicom è un emulatore di terminale per sistemi operativi Unix-like. Usato comunemente quando si configura una console seriale remota.
Picocom, simile al software precedente, è uno strumento semplice e manuale, che permette di configurare e gestire una porta seriale, collegandosi all'emulatore di terminale già in uso. Il software in questione permette la comunicazione seriale con il dispositivo target.
Poichè la comunicazione attraverso la UART è asincrona, ASCII-XFR permette di introdurre, durante l'invio di un codice sorgente al Raspberry, un ritardo tra ogni carattere. In tal modo, il ritardo introdotto consente di processare le istruzioni ricevute prima di riceverne di nuove.
Grazie all'utilizzo di questo ambiente di sviluppo è permessa la programmazione e il testing delle funzionalità introdotte in real-time con il dispositivo target.

```
sudo picocom --b 115200 /dev/ttyUSB0 --imap delbs -s "ascii-xfr -sv -l100 -c10"
```

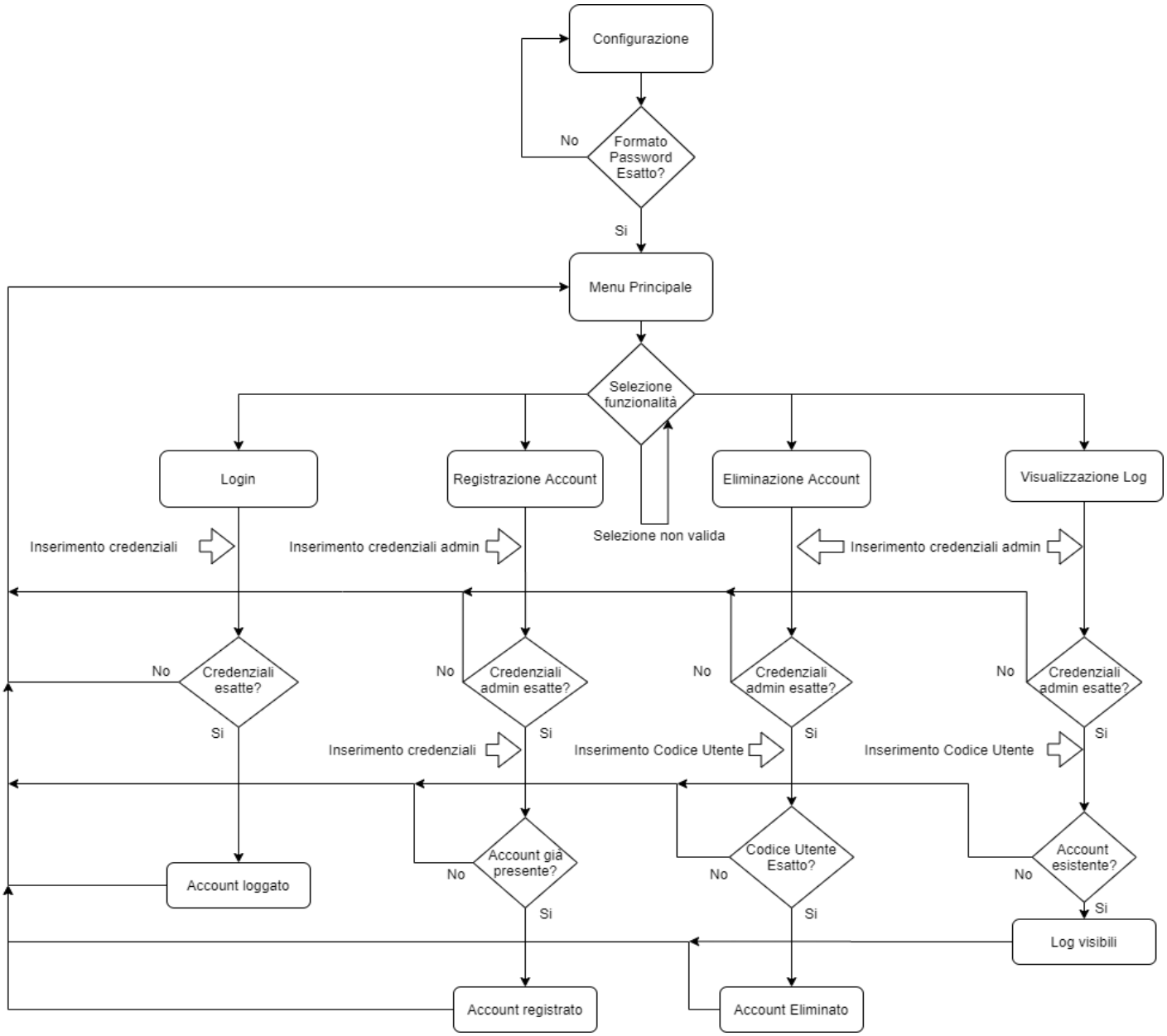
- `--b 115200`: 115200 bit/s bit rate.
- `--imap delbs`: Permette l'utilizzo del backspace per eliminare i caratteri.
- `-s "ascii-xfr -sv -l100 -c10"`: specifica ascii-xfr come comando esterno da utilizzare per trasmettere i files.
  - `-sv`: modalità di invio verbose.
  - `-l100`: imposta un ritardo di 100ms dopo ogni linea inviata, questo solitamente dà il tempo necessario a eseguire un comando ed essere pronti per il prossimo...
  - `-c10`: Aspetta 10ms dall'invio di un carattere e l'altro.

Software

Come già accennato nell'introduzione, il sistema sviluppato in questione si prefigge l'obiettivo di garantire un accesso sicuro, tramite l'ausilio di una password, in ambienti domestici e non, in cui esso è installato, solamente al personale autorizzato.
Durante il primo avvio, il sistema effettuerà una serie di inizializzazioni atte a configurare tutte le periferiche utilizzate dal sistema, indispensabili per il suo corretto funzionamento. Inoltre, durante questa prima fase, all'utente verrà mostrata, nel display LCD, una schermata in cui gli sarà permesso di impostare le credenziali di amministratore. L'immissione delle credenziali avverrà attraverso l'ausilio di un tastierino matriciale 4x4.
Le credenziali per un utente con permessi di amministratore sono costituite da un *codice utente* (999), impostato di default nel sistema, e da una *password* scelta in fase di configurazione. Per quanto riguarda l'utente semplice, sia il codice utente che la password vengono scelti in fase di registrazione.
Il formato del codice utente e della password, per entrambe le tipologie di utente, è il medesimo, ed è costituito da tre caratteri numerici per quanto riguarda il codice utente, mentre da sei caratteri numerici per quanto riguarda la password.
Una volta completata la fase di configurazione, sul display LCD, verrà mostrata la schermata principale del sistema. Essa presenta quattro funzionalità, ognuna etichettata con una lettera; *login* (A), *registrazione* (B), *cancellazione* (C), *visualizzazione log* (D). Le seguenti funzionalità potranno essere selezionate tramite l'immissione dei rispettivi caratteri attraverso

l'ausilio del tastierino; se il sistema rileverà una non corretta immissione da parte dell'utente, il sistema gestirà questo evento riportando l'utente nella schermata principale.
Il *login*, come già si presuppone, consente, tramite l'inserimento di codice utente e password, all'utente di effettuare correttamente l'autenticazione e quindi di accedere alla zona protetta dal sistema. Se le credenziali inserite risulteranno errate, il sistema impedirà all'utente in questione l'accesso all'area protetta. Per semplicità, il nostro sistema simula lo sblocco e quindi l'accesso sicuro all'area tramite l'impiego di un led RGB che, nel caso di esito positivo si illuminerà di verde, mentre nel caso di esito negativo, si illuminerà di rosso; verrà anche mostrato a schermo un messaggio di benvenuto o di accesso negato, dipendente dall'esito del login.
Il sistema prevede la visualizzazione in tempo reale degli utenti che si sono effettivamente autenticati, mostrando in uno schermo HDMI le informazioni riguardanti il codice utente e la data. Il formato della data è il seguente, AAA/DDD/HH:MM:SS ed esprime il tempo assoluto trascorso dall'accensione del sistema (si assuma che il sistema rimarrà attivo per un tempo indefinito).
La *registrazione* ha lo scopo di inserire nuovi utenti, funzionalità riservata esclusivamente all'utente amministratore. In questa fase, il sistema richiederà, prima di effettuare l'inserimento di un utente, le credenziali di amministratore, e solamente dopo aver ricevuto esito positivo si potrà proseguire alla scelta delle credenziali del nuovo utente. Il sistema inserirà in memoria le nuove credenziali solamente se queste ultime soddisferanno una serie di requisiti, cioè, il codice utente non dovrà essere già presente all'interno del sistema e i caratteri che comporranno sia quest'ultimo sia la password dovranno essere esclusivamente numerici; se non verrà soddisfatto anche un solo dei seguenti requisiti, il sistema, dopo aver visualizzato un messaggio di errore appropriato, riporterà l'utente alla schermata principale.
La *cancellazione* concede all'utente amministratore la possibilità di rimuovere dal sistema uno specifico utente impedendo a quest'ultimo in futuro l'accesso all'area protetta. Quindi, una volta inserite le credenziali di amministratore, il sistema permetterà la cancellazione di un utente tramite l'inserimento del rispettivo codice utente. Se il codice utente sarà presente in memoria, il sistema effettuerà correttamente la cancellazione, in caso contrario verrà visualizzato un messaggio di errore e il sistema verrà riportato nelle condizioni iniziali.
Infine, l'ultima funzionalità consente all'utente amministratore di poter visualizzare tutti i log di un determinato utente. I *log* in questione verranno visualizzati in uno schermo HDMI e daranno informazioni sugli accessi e/o tentati accessi di uno specifico utente. Questa

funzionalità è cruciale in un sistema di sicurezza poiché permette di memorizzare tutto lo storico degli accessi, catalogato per ogni utente.



Qui di seguito viene riportato il diagramma di flusso del sistema.

Vengono adesso illustrati e discussi i moduli che contengono il codice sorgente scritto in FORTH. L'ordine con cui verranno discussi è il medesimo che bisogna seguire per caricare il codice sul dispositivo target tramite Picocom, e quindi eseguire correttamente il software.
Il caricamento del codice sorgente verrà automatizzato tramite l'ausilio di un *makefile* allegato al codice sorgente.

Utility.f

Questo modulo comprende una serie di *word*, definizioni scritte in FORTH, che sono ampiamente utilizzate per semplificare la scrittura del codice nei successivi moduli.
Poiché JonesForth non è conforme ad ANSI, alcune parole appartenenti allo standard FORTH non sono definite oppure non si comportano come ci si aspetterebbe. Per ovviare a ciò, all'interno di questo modulo abbiamo inserito le definizioni delle word presenti nel file *se-ans.f* fornito come materiale didattico del corso di Sistemi Embedded. Il file in questione ridefinisce alcune word garantendo la loro conformità secondo lo standard ANS. Il codice presente all'interno di questo modulo è il primo ad essere caricato così da garantire l'esecuzione corretta dei successivi moduli.

Assembly.f

In questo modulo sono contenute le definizioni di due word fondamentali per la gestione del tempo utilizzate nel modulo che tratteremo successivamente. Le due word, UM/ e TIME@, sono state definite tramite la compilazione di istruzioni macchina ottenute disassemblando il file binario relativo al codice assembly per architetture ARM-32. Le istruzioni macchina ottenute, sono eseguite dall'ambiente FORTH tramite la word , (comma).
Qui di seguito vengono mostrati i comandi da terminale.

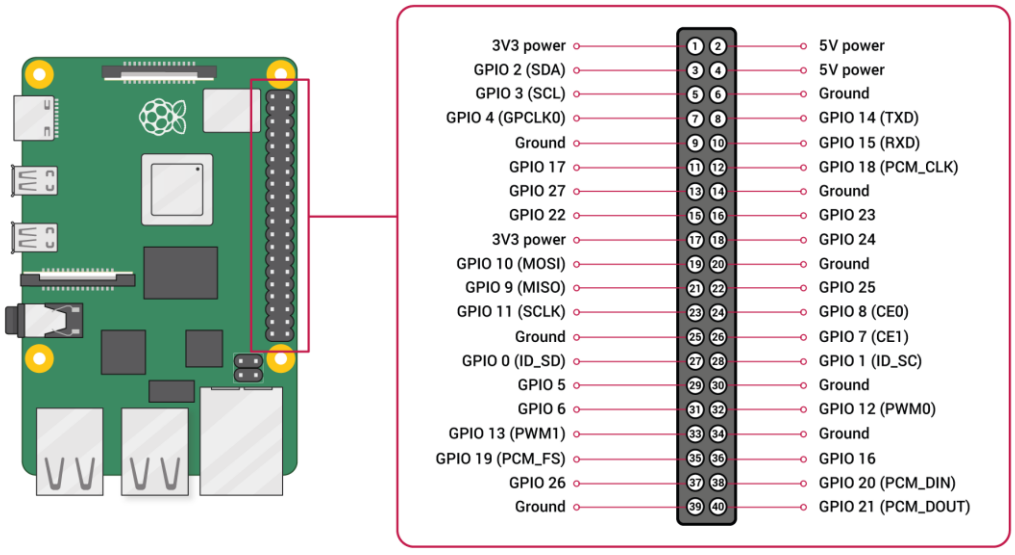
```
arm-none-eabi-as *.s -o *.o ;
arm-none-eabi-ld *.o -o *.elf ;
arm-none-eabi-objdump -d *.elf > *.elf.list ;
```

Tramite l'impiego della GNU ARM Embedded Toolchain siamo riusciti ad assemblare, e linkare correttamente il codice assembly, ottenendo il file .elf (Executable and Linkable Format), per poi effettuare l'objdump.
La prima delle due word, UM/, consente di introdurre nell'ambiente pijFORTHos una nuova operazione fondamentale per il nostro sistema, cioè la divisione tra un valore a 64-bit ed un

valore a 32-bit, restituendo un quoziente a 32-bit. Mentre la word, TIME@, permette una volta eseguita, di leggere contemporaneamente i registri basso e alto del SYSTEM_TIMER_CLOCK.

GPIO.f

All'interno di questo modulo sono definiti come costanti, diversi registri hardware del dispositivo target, impiegati per implementare determinate funzionalità del sistema. Inoltre sono definite delle word, come GPIO_INPUT oppure GPIO_OUTPUT che rispettivamente permettono di impostare un determinato pin GPIO come input oppure come output.
Nella figura sottostante vengono illustrati i pin GPIO del Raspberry Pi 4B. L'intestazione GPIO è costituita da 40 pin ognuno dei quali può essere impostato come pin di input o di output per una vasta gamma di scopi.



Timer.f

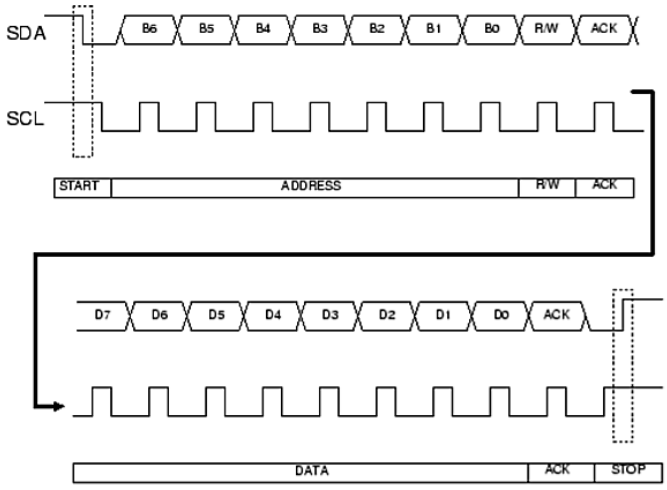
Questo modulo permette di gestire il tempo tramite l'impiego dei registri SYSTEM_TIMER_CLOCK. Questi registri hardware tengono conto del tempo trascorso, in microsecondi, dall'accensione del dispositivo. A tal proposito abbiamo definito delle word che ci permettono di effettuare *delay* di specifici intervalli di tempo all'interno del nostro sistema, e inoltre tramite la word DATETIME, siamo in grado di calcolare e quindi poi

stampare su schermo HDMI il tempo trascorso, in un formato appropriato, per un periodo di tempo indefinito, grazie all'ausilio di UM/.

I2C.f

All'interno di questo modulo sono presenti tutte le word che consentono la comunicazione attraverso il protocollo I2C, fondamentale per interfacciarci con il display LCD. In particolare, la word, denominata WRITE_I2C, si occupa di effettuare il trasferimento dei dati, rispettando il protocollo I2C, tra il master (Raspberry) e lo slave (display LCD).
Il protocollo I2C permette la comunicazione di dati tra due o più dispositivi I2C utilizzando un *bus* a due fili, più uno per il riferimento comune di tensione. In tale protocollo le informazioni sono inviate serialmente usando una linea per i dati SDA (Serial Data Line), ed una per il Clock, SCL (Serial Clock Line). Deve infine essere presente una terza linea, ovvero la massa. Ci sono quattro distinti modi di operare: un master trasmette, un master riceve, uno slave trasmette, uno slave riceve. Il dispositivo *master* è quello che controlla il bus in un certo istante, e controlla il segnale di Clock e genera i segnali di START e di STOP. I dispositivi *slave* "ascoltano" il bus ricevendo dati dal master o inviandone qualora questo ne faccia loro richiesta. Una sequenza di lettura o scrittura di dati tra master e slave segue il seguente ordine:

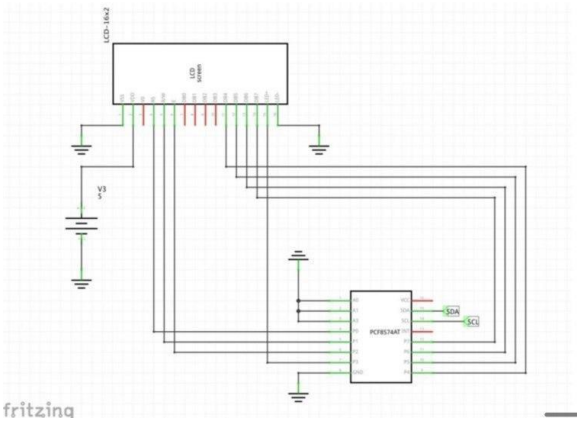
- Invio del bit di START (S) da parte del master;
- Invio dell'indirizzo dello slave (ADDR) ad opera del master;
- Invio del bit di Read (R) o di Write (W), che valgono rispettivamente 1 o 0 (sempre ad opera del master);
- Attesa/invio del bit di Acknowledge (ACK);
- Invio/ricezione del byte dei dati (DATA);
- Attesa/invio del bit di Acknowledge (ACK);
- Invio del bit di STOP (P) da parte del master.



LCD.f

Il modulo in questione ci permette di visualizzare a schermo stringhe di testo utili al funzionamento del nostro sistema. Le stringhe, vengono inviate e quindi visualizzate su schermo attraverso il protocollo I2C implementato e discusso nella sezione precedente.
Il display necessita di una inizializzazione prima di effettuare qualsiasi operazione. L'inizializzazione prevede una serie di istruzioni volte a configurare la modalità di invio (a 4-bit), il numero di righe nello schermo, la dimensione dei caratteri, l'accensione del display, l'accensione del cursore, la retro-illuminazione, e infine il settaggio della modalità *entry-mode*. Inoltre sono state introdotte una serie di funzionalità che consentono la navigazione all'interno del display LCD.
La word, SEND_STRING, che permette il corretto invio delle stringhe, utilizzando la modalità di invio a 4-bit prevista dal dispositivo HD44780, divide ogni singolo carattere della stringa in modo da essere processato secondo la modalità di invio.
La modalità di invio a 4-bit prevede la suddivisione in 4 messaggi distinti: i primi due messaggi, sono composti dai 4 bit più significativi del carattere che prendono posto nei primi 4 bit più significativi (P7-P4) del messaggio con l'unica differenza di avere il bit P2 di *enable*, impostato una volta in *high* (1) e una volta a *low* (0). La medesima cosa accade per i 4 bit meno significativi del carattere. Il processo di invio si ripete finché non verrà processato l'ultimo carattere della stringa da inviare all' LCD.
Qui in basso vi è raffigurato lo schema di invio utilizzato.

P7	P6	P5	P4	P3	P2	P1	P0
D7	D6	D5	D4	Backlight	Enable	R/W'	RS



Keypad.f

La seguente modulo comprende tutte le funzioni relative al tastierino matriciale e alla sua configurazione. Tramite la word KEYPAD_INIT vengono inizializzate tutte le porte GPIO necessarie per l'interfacciamento del tastierino e inoltre viene attivata la funzione di *falling edge detection* per le righe. Quest'ultima permette di settare un bit a 1 *nell'event detect register* nel caso in cui ci sia stata una transizione di *falling edge*, cioè un cambiamento di voltaggio da alto a basso. Per interfacciare il tastierino vengono impostate in input le porte relative alle righe e in output quelle relative alle colonne. Tramite un loop, richiamato dalla parola READ_VALUES, per ogni colonna, viene mandato un output e viene fatto un controllo sul valore delle quattro righe, se viene letto il bit di *event detection* per una delle righe si effettua un match riga-colonna e quindi l'output del relativo valore. Nel caso in cui i valori da inserire facciano parte di una password l'output a video diventa un asterisco per ogni valore. Prima di collegare il tastierino è stato effettuato un controllo sulle resistenze di Pull Up e Pull Down sulle porte utilizzate. Sono state scelte le porte con resistenza di Pull Down impostate di default. Quest'ultima permette di mantenere uno stato logico stabile all'ingresso del pin del Raspberry Pi senza oscillazioni e quindi evitare letture spurie.

HDMI.f

In questo modulo si trovano tutte le word che permettono di gestire la scrittura (nel nostro caso dei log), su un display HDMI.
Il display HDMI è generato dal Videocore, e non è altro che una matrice di *width x height* pixels. I parametri, di ampiezza e altezza, possono essere scelti e definiti dalla risoluzione spaziale del display.
Abbiamo definito delle word che ci hanno permesso di mappare l'intero set di caratteri stampabili ASCII in modo da poter gestire e quindi visualizzare sul display qualsiasi tipo di stringa. Le word in questione sono in grado di spostarsi sull'array di pixel e accenderlo salvando all'interno di esso un valore che esprime il colore. I valori dei pixels sono salvati in un *framebuffer*. Quest'ultimo è organizzato come una matrice righe-colonne, i cui pixels sono predisposti uno dopo l'altro come in un array lineare.
La word HDMI_INIT permette di inizializzare il display selezionando il colore e inizializzando le variabili che ci permettono di suddividere lo schermo in righe su cui scrivere testo. Una volta terminato lo spazio del framebuffer, la word SCROLLING effettuerà lo spostamento verso l'alto del testo, lasciando posto ad una nuova riga vuota; tutto ciò cerca di simulare lo scrolling verticale di un tradizionale terminale. Per velocizzare l'esecuzione dello scrolling, abbiamo sfruttato delle istruzioni assembly che ci hanno permesso di spostare in maniera quasi immediata diversi valori dei registri contemporaneamente.

Led.f

Questo modulo comprende semplicemente delle word che consentono di inizializzare e quindi poter in seguito accendere il led RGB con degli specifici colori. Il colore che assumerà il led RGB dipenderà da un particolare evento del sistema che si vuole segnalare all'utente.

Log.f

Il modulo in questione permette gestire i log. La word USERLOG_INIT consente innanzitutto di inizializzare il buffer circolare di una dimensione prefissata che conterrà i log degli utenti, mentre la word REGISTER_LOG consente di memorizzare sugli array le informazioni utili che poi verranno stampate su schermo HDMI in un secondo momento.

Per creare la stringa adatta da visualizzare viene utilizzata la word CREATE_STRING. L'output finale viene mandato a schermo tramite la word PRINT_LOG.

Security.f

L'ultimo modulo definisce il *core* di funzioni del sistema. Il sistema viene avviato chiamando la word BOOT che innesca una serie di funzioni necessarie alla preparazione del sistema. La configurazione iniziale del sistema avviene tramite la word CONFIGURATION che inizializza gli array deputati a memorizzare le credenziali di accesso dell'utente, abilita il tastierino, attiva il protocollo I2C e il display LCD, e infine il display HDMI. La word inoltre permette l'impostazione della password di amministratore. La word MAIN consente di visualizzare tramite LCD e selezionare le funzionalità del sistema tramite l'ausilio del tastierino.

Come usare il sistema

1. Scaricare ed estrarre tutto il software contenuto nell'archivio **ProgettoEmbedded.rar**
2. Se non presente nel sistema scaricare e installare i pacchetti Minicom e Picocom.
3. Collegare il **Raspberry** e **FT232RL** con il PC, *si consiglia di alimentare entrambi i dispositivi con il PC per evitare configurazioni elettriche differenti.*
4. Aprire un terminale e recarsi all'interno della cartella ***Codice***
5. Collegare il Raspberry alle periferiche [LCD, HDMI,KEYPAD,LEDRGB].
6. Eseguire il comando **make**
7. Premere la combinazione di tasti **CTRL-A-S**
8. Trascinare il file **merged.f** all'interno del terminale.
9. Aspettare il caricamento del programma.
10. Eseguire il comando **boot**.
11. Il programma è in esecuzione da questo momento è possibile rimuovere la UART.