MSc in Computer Science
at University of Milan

CHIP-8 STM32

Proposta per il Progetto di PROS,
corso tenuto da **Danilo Bruschi**

Email:
federico.bruzzone@studenti.unimi.it
lorenzo.ferrante1@studenti.unimi.it
andrea.longoni3@studenti.unimi.it

Creato da:
Federico Bruzzone
Lorenzo Ferrante
Andrea Longoni

Anno accademico 2022/2023

1 Introduzione

CHIP-8 é un linguaggio di programmazione creato a metà degli anni '70 da Joseph Weisbecker per semplificare lo sviluppo di videogiochi per microcomputer a 8 bit. I programmi CHIP-8 vengono interpretati da una macchina virtuale che é stata estesa parecchie volte nel corso degli anni, tra le versioni più adottate citiamo S-CHIP e la più recente XO-CHIP.

La semplicità dell'interprete in aggiunta alla sua lunga storia e popolarità hanno fatto sì che emulatori e programmi CHIP-8 vengano realizzati ancora oggi. Nel corso degli anni molti videogiochi storici sono stati "portati" su CHIP-8 tra cui Pong, Space Invaders e Tetris.

Lo scopo del progetto é quello di costruire un emulatore CHIP-8 e S-CHIP in grado di funzionare su un microcontrollore STM32.

2 Analisi di Mercato

Al giorno d'oggi risulta difficile ottenere un numero esatto di utenti, ma possiamo basarci sul topic "chip8" di GitHub che raggruppa quasi un migliaio di repository.

Tra queste la più popolare è Octo, un'implementazione scritta in JavaScript capace di eseguire la versione base di CHIP-8, S-CHIP e XO-CHIP nel browser. La repository è mantenuta da John Earnest, l'inventore di XO-CHIP che nel 2014 ha riportato in vita CHIP-8 modernizzandolo e aggiungendo nuove funzionalità.

Inoltre ogni anno viene organizzata la Octojam, una game jam dove ogni partecipante prova a sviluppare un videogioco per CHIP-8 (o per le sue estensioni) partendo da zero.

Grazie al suo instruction set ridotto e alla sua limitata richiesta di risorse hardware è stato portato su un elevato numero di piattaforme, tra cui il Game Boy Color, calcolatrici grafiche serie HP 48 e Emacs (il famoso editor di testo).

Sebbene CHIP-8 e S-CHIP siano stati tradizionalmente implementati tramite software esistono anche implementazioni hardware. Ne citiamo una in particolare scritta nel linguaggio Verilog per schede FPGA.

3 Componenti Hardware

Il componente principale é il microcontrollore `STM32F334R8T6` basato su architettura ARM con processore Cortex-M4 da 72 `MHz`, 64 `Kb` di memoria flash e 16 `Kb` di SRAM. Abbiamo deciso di utilizzare questa scheda perché le ROM dei giochi CHIP-8 e S-CHIP hanno dimensione massima di 4 `Kb`. Considerando questo e il fatto che la macchina virtuale necessita 4 Kb per poter funzionare non è stato potuto utilizzare la scheda fornitaci durante il corso a causa della sua quantità limita di SRAM.

Necessiteremo inoltre di un display TFT LCD (thin-film-transistor liquid-crystal display) a colori retroilluminato, per poterci interfacciare col gioco. Questo display, da 2.4 pollici, é basato sul controller `ILI9341` e ha una risoluzione di 320×240 `px`.

Il display dispone di un lettore di schede microSD, che verrà utilizzato per caricare una delle ROM presenti sulla SD in SRAM.

Per interagire con l'emulatore utilizzeremo una tastiera matriciale 4×4 corrispondente alla tastiera esadecimale originale.

Per riprodurre gli effetti sonori generati dal gioco utilizzeremo un beeper a frequenza variabile. In CHIP-8 e S-CHIP vi é un solo tipo di suono che può essere riprodotto durante tutta l'esecuzione del gioco.

Infine sarà possibile aggiungere uno slot per l'alimentazione tramite due batterie `AA`.

4 Componenti Software

Le parti software che saranno implementate sono:

1. La **Macchina Virtuale** (VM) che interpreta programmi CHIP-8 e S-CHIP. Non volendoci affidare ad implementazioni già esistenti abbiamo deciso di svilupparla da zero, per questo motivo verrà effettuato il testing con un'interfaccia emulata su un calcolatore x86 a 64 `bit` tramite SDL2 o Raylib (due librerie per creare interfacce multimediali in C). Successivamente verrà portata sul microcontrollore basato su ARM Cortex-M.
2. Il **Driver Video** che permette l'interfacciamento tra il microcontrollore e il display TFT LCD. Lo schermo supporta interfacce parallele a 8 `bit` e SPI a 4 pin, per ragioni di prestazioni noi implementeremo il driver utilizzando l'interfaccia parallela a 8 `bit`.
3. Il **Driver microSD** che permette l'interfacciamento tra il microcontrollore e il lettore della scheda SD, verrà sviluppato utilizzando l'interfaccia SPI. Per questo driver utilizzeremo le librerie fornite da ST per la gestione del filesystem `FAT32` e facilitare lo sviluppo del trasferimento dati.
4. Il **Driver Keypad** che permette l'interfacciamento tra il microcontrollore e la tastiera matriciale 4×4. La gestione degli input da tastiera non verrà effettuata in polling, ma bensí tramite interrupt.
5. Il **Driver Sound** che permette l'interfacciamento tra il microcontrollore e il beeper. Il driver permette di generare suoni a frequenza variabile.

Inoltre verrà implementato un menù per la selezione del gioco all'avvio dell'emulatore. Infine abbiamo deciso di non usare librerie esterne ma di sviluppare una nostra libreria per effettuare il rendering del font e disegnare sul display.

5 Architettura Software

6 Analisi del Consumo Energetico

Come abbiamo giá accennato in precedenza il microcontrollore durante lo sviluppo verrà alimentato tramite USB e successivamente utilizzando batterie `AA`. Le batterie hanno una tensione di `3V` e una capacitá di ∼2500 `mA` e usandone due in serie otterremo 5000 `mA`.

Stimando un consumo dello schermo di 100 `mAh` e un consumo di 25 `mAh` per il microcontrollore e le sue periferiche, avremo un consumo totale di 125 `mAh`.

$$\begin{aligned}\text{time (h)} &= \frac{\text{capacity (mA)}}{\text{consumption (mAh)}} \\ &= \frac{2 * 2500\ \texttt{mA}}{125\ \texttt{mAh}} \\ &= 40\ \texttt{h}\end{aligned}$$

Possiamo stimare una durata di 40 ore di gioco.

7 Analisi di Tempi e Costi

Nome	**Modello**	**Costo unitario**	**Unità**	**Costo**
Schermo	ILI9341 2.4"	6.44	1	6.50
Matrix keypad	AZ-Delivery 4×4	3.99	1	3.99
Microcontrollore	STM32 F334R8T6	14.99	1	14.99
Beeper		0.99	1	0.99
Breadboard e cablaggio		4.99	1	4.99
			Totale	31.50€

Table 1: Materiali previsti per la costruzione del progetto. I costi indicati provengono da negozi online come Amazon e eBay

Nome	**Tempo di lavoro**
Macchina virtuale	3 settimane
Driver video	1 settimana
Driver microSD	3 giorni
Driver keypad	1 giorno
Driver audio	5 giorni
Selezione gioco	5 giorni
Ottimizzazione software	2 settimane
Totale	2 mesi

Table 2: Tempi previsti per la realizzazione dei componenti software

8 UML test