Study Notes

Generated on 5/4/2025

Study Content

Dispensa di Studio su Arduino

Fonte principale: Playlist YouTube - Introduzione ad Arduino

Indice

• Introduzione ad Arduino

1. 1 Cos’è Arduino?
• Definizione: Piattaforma open-source per prototipazione elettronica, basata su hardware (schede) e software (IDE). 2 Architettura Hardware
• Microcontrollore: ATmega328 (Arduino Uno) con 32 KB di memoria flash.
• Pin:
• 14 digitali (6 PWM), 6 analogici.
• Alimentazione: 5V (USB) o 7-12V (esterna).

• Componenti Hardware

2. 1 Schede Comuni

Modello	Pin Digitali	Memoria Flash	Dimensioni
Uno	14	32 KB	Standard
Nano	14	32 KB	Compatta
Mega 2560	54	256 KB	Large

2. 2 Sensori e Attuatori
• Input:
• Fotoresistore (luce), termistore (temperatura), pulsante. 3 Elettronica di Base
• Legge di Ohm: $V = R \cdot I$ (esempio: calcolo resistenza per un LED).
• Circuiti Protetti:
• Resistenza in serie con LED: $R = \frac{V_{sorgente} - V_{LED}}{I_{LED}}$.

• Programmazione e Ambiente IDE

3. 1 Struttura di un Sketch

3. 2 Funzioni Principali

Funzione	Descrizione
`digitalRead()`	Legge un valore digitale (HIGH/LOW). 1 Controllo di un Servomotore

Circuitazione:
• Servo collegato a pin PWM (es. 2 Lettura Sensore di Temperatura (LM35)

Formula:
$T (°C) = \frac{\text{valore analogico} \times 5}{1023} \times 100$. 1 Esercizi
• Allarme Fotoresistente:
• Accendi un buzzer se la luce ambientale scende sotto una soglia. 2 Domande di Verifica
• Cosa succede se si collega un LED senza resistenza? Arduino + Python). 3 Risorse Consigliate
• Libri: "Arduino Cookbook" di Michael Margolis.
• Siti: Arduino Project Hub.

Concetti fondamentali e applicazioni pratiche

Indice

• 6. 4 Controllo a distanza di dispositivi
• Approfondimenti e risorse aggiuntive
• 7. 1 Analisi dell'efficacia della playlist
• 8. 1 Cos'è Arduino

Arduino è una piattaforma hardware e software open-source pensata per rendere l'elettronica e la programmazione accessibili a tutti. Costituisce un ecosistema completo che permette di creare progetti interattivi di varia complessità, dai semplici esperimenti educativi fino a sistemi embedded professionali.

Il cuore di questa piattaforma è rappresentato dalle schede Arduino, dispositivi hardware basati su microcontrollori che possono essere programmati per leggere input (come la luce su un sensore, la pressione di un pulsante o un messaggio su Twitter) e convertirli in output (attivando un motore, accendendo un LED, pubblicando qualcosa online).

La filosofia alla base di Arduino è quella di rendere la tecnologia accessibile a tutti, indipendentemente dal background tecnico dell'utente. Questo approccio ha reso Arduino uno strumento ideale sia per principianti che vogliono muovere i primi passi nel mondo dell'elettronica digitale, sia per professionisti che necessitano di prototipare rapidamente le proprie idee. 2 Storia e filosofia del progetto

Arduino nacque nel 2005 presso l'Interaction Design Institute di Ivrea, in Italia. Fu concepito da un gruppo di docenti e studenti, tra cui Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino e David Mellis, con l'obiettivo di creare uno strumento semplice ed economico per gli studenti di design che non avevano competenze specifiche in elettronica o programmazione.

Il nome "Arduino" deriva da quello di un bar di Ivrea frequentato dai fondatori del progetto, che a sua volta prende il nome da Arduino d'Ivrea, un re italiano del X secolo.

La filosofia alla base del progetto Arduino si fonda su alcuni principi fondamentali:
• Open Source: sia l'hardware che il software sono rilasciati con licenze open source, permettendo a chiunque di studiare, modificare e migliorare il progetto.
• Accessibilità: le schede Arduino sono progettate per essere economiche e facili da utilizzare, anche per chi non ha conoscenze avanzate di elettronica.
• Comunità: Arduino ha sviluppato una vasta comunità di utenti che condividono progetti, soluzioni e idee.
• Interdisciplinarità: la piattaforma è pensata per essere utilizzata in contesti diversi, dall'arte all'educazione, dalla domotica alla robotica. 3 La famiglia Arduino e le diverse schede

Nel corso degli anni, la famiglia Arduino si è ampliata notevolmente, offrendo diverse schede adatte a vari tipi di progetti e esigenze. Alcune delle principali schede Arduino sono:

Arduino UNO: È la scheda più popolare e utilizzata, ideale per principianti. Basata sul microcontrollore ATmega328P, offre un buon equilibrio tra facilità d'uso, performance e costo.

Arduino Nano: Una versione compatta dell'UNO, perfetta per progetti dove lo spazio è limitato.

Arduino Mega: Dotata di un microcontrollore più potente e di un maggior numero di pin, è adatta per progetti più complessi che richiedono molte connessioni I/O.

Arduino Leonardo: Utilizza il microcontrollore ATmega32u4 che consente alla scheda di apparire al computer come una tastiera o un mouse.

Arduino Due: Basata su un processore ARM Cortex-M3, è molto più potente delle schede basate su AVR, adatta per applicazioni che richiedono elevate capacità di calcolo.

Arduino MKR series: Una famiglia di schede compatte focalizzate su specifici ambiti, come IoT (Internet of Things), connettività wireless, sicurezza.

Arduino Portenta: Schede di fascia alta per applicazioni industriali e AI (Intelligenza Artificiale).

Ogni scheda ha caratteristiche specifiche in termini di dimensioni, capacità di elaborazione, numero di pin I/O, supporto per comunicazioni wireless, e altro ancora. La scelta della scheda più adatta dipende dalle esigenze specifiche del progetto che si intende realizzare. 4 Perché imparare Arduino

Imparare a utilizzare Arduino offre numerosi vantaggi, sia per studenti che per professionisti:

Approccio pratico all'elettronica e alla programmazione: Arduino permette di apprendere concetti di programmazione ed elettronica in modo concreto e tangibile, rendendo l'apprendimento più efficace.

Versatilità: Con Arduino è possibile realizzare un'ampia gamma di progetti, dall'automazione domestica ai giochi, dai sistemi di monitoraggio ambientale ai robot.

Competenze trasferibili: Le conoscenze acquisite con Arduino sono applicabili in molti campi, dalla domotica all'industria 4.

Comunità di supporto: Esiste una vasta comunità di utenti Arduino che condivide progetti, codice e soluzioni, facilitando l'apprendimento e la risoluzione dei problemi.

Costo contenuto: Rispetto ad altre piattaforme di sviluppo, Arduino ha un costo accessibile, permettendo anche a scuole e singoli studenti di avvicinarsi al mondo dell'elettronica.

Innovazione e creatività: Arduino stimola la creatività, permettendo di trasformare idee in prototipi funzionanti con relativamente poco sforzo.

Preparazione al futuro: In un mondo sempre più dominato dalla tecnologia, le competenze di programmazione e di elettronica forniscono un vantaggio significativo nel mercato del lavoro.

• Componenti hardware di Arduino

2. 1 Anatomia della scheda Arduino UNO

La scheda Arduino UNO, essendo la più diffusa e utilizzata, costituisce un riferimento ideale per comprendere l'anatomia generale di una scheda Arduino. Analizziamo i suoi componenti principali:

Microcontrollore: Il "cervello" della scheda, un ATmega328P per Arduino UNO. Si tratta di un chip che contiene un processore, memoria e periferiche di input/output.

Connettore USB: Permette di collegare la scheda al computer per programmarla e per la comunicazione seriale.

Jack di alimentazione: Consente di alimentare la scheda con una fonte esterna (7-12V).

Pin di alimentazione:
• VIN: Pin per fornire alimentazione esterna alla scheda.

Pin digitali (0-13): Utilizzabili come input o output digitali. Alcuni di questi pin hanno funzionalità speciali:
• Pin 0 e 1: Utilizzati per la comunicazione seriale (RX e TX).
• Pin 2 e 3: Possono essere utilizzati per gli interrupt esterni.
• Pin 10, 11, 12, 13: Utilizzati per la comunicazione SPI.
• Pin 13: Collegato al LED integrato sulla scheda.

Pin analogici (A0-A5): Utilizzati principalmente come input analogici, ma possono fungere anche da pin digitali. I pin A4 e A5 sono usati anche per la comunicazione I2C.

Pulsante di reset: Permette di riavviare il programma sulla scheda.

LED integrati:
• Power LED: Indica quando la scheda è alimentata.
• LED TX/RX: Lampeggiano durante la comunicazione seriale.

Regolatore di tensione: Converte la tensione in ingresso (da USB o jack) a 5V e 3.

Oscillatore al quarzo: Fornisce il clock a 16 MHz per il microcontrollore.

ICSP (In-Circuit Serial Programming) header: Connettore per la programmazione del microcontrollore attraverso un programmatore esterno. 2 Microcontrollore e sue caratteristiche

Il microcontrollore è il componente centrale di ogni scheda Arduino. Nel caso di Arduino UNO, si tratta dell'ATmega328P prodotto da Microchip (precedentemente Atmel). Vediamo le sue principali caratteristiche:

Architettura: L'ATmega328P è basato sull'architettura AVR a 8-bit, una versione modificata dell'architettura RISC (Reduced Instruction Set Computer). Questa memoria non volatile contiene il programma (sketch) caricato sulla scheda. Questa memoria volatile viene utilizzata per memorizzare le variabili durante l'esecuzione del programma. Memoria non volatile che

può essere utilizzata per memorizzare dati che devono persistere anche quando la scheda viene spenta.

Input/Output: 23 linee di I/O programmabili (14 digitali e 6 analogiche nell'Arduino UNO, con alcune linee con funzionalità multiple).

Periferiche integrate:
• ADC (Analog-to-Digital Converter): Converte segnali analogici in valori digitali.
• Timer/Counter: Utilizzati per operazioni temporizzate e conteggio.
• PWM (Pulse Width Modulation): Per generare segnali analogici attraverso tecniche digitali.
• Interfacce di comunicazione: UART (seriale), SPI, I²C.

Tensione operativa: Generalmente 5V, anche se alcuni microcontrollori (come quelli usati nelle schede Arduino a 3. 3 Pin digitali e analogici

I pin di Arduino rappresentano l'interfaccia tra il microcontrollore e il mondo esterno. Comprendere la loro natura e funzionamento è fondamentale per creare progetti efficaci.

Pin digitali: I pin digitali possono assumere solo due stati: HIGH (5V) o LOW (0V). Possono essere configurati come:
• Input: Per leggere il valore di un segnale esterno (ad esempio, un pulsante).
• Output: Per controllare componenti esterni (ad esempio, un LED).

Alcuni pin digitali hanno funzionalità speciali:
• PWM (Pulse Width Modulation): I pin contrassegnati con "~" possono generare segnali analogici mediante la tecnica PWM. Questa tecnica alterna rapidamente il pin tra stato HIGH e LOW, creando un valore medio che simula un'uscita analogica.
• Comunicazione seriale: I pin 0 (RX) e 1 (TX) sono utilizzati per la comunicazione seriale.
• Interrupt esterni: I pin 2 e 3 possono essere configurati per generare interruzioni nel programma in risposta a eventi esterni.
• Comunicazione SPI: I pin 10 (SS), 11 (MOSI), 12 (MISO) e 13 (SCK) sono utilizzati per la comunicazione SPI.
• Comunicazione I²C: I pin A4 (SDA) e A5 (SCL) sono utilizzati per la comunicazione I²C.

Pin analogici: I pin analogici (A0-A5 su Arduino UNO) possono leggere valori analogici continui, non solo HIGH o LOW. Sono collegati all'ADC (Analog-to-Digital Converter) del microcontrollore, che converte la tensione analogica (0-5V) in un valore digitale (0-1023 su Arduino UNO, che utilizza un ADC a 10 bit).

I pin analogici possono essere utilizzati anche come pin digitali quando necessario.

È importante notare che, mentre Arduino può leggere valori analogici attraverso i pin analogici, può generare output analogici solo attraverso la tecnica PWM sui pin digitali supportati. 4 Alimentazione

L'alimentazione è un aspetto cruciale per il corretto funzionamento di Arduino e dei componenti a esso collegati. Arduino può essere alimentato in diversi modi:

Via USB: Quando collegato a un computer tramite cavo USB, Arduino riceve 5V. Questa modalità è comoda durante la fase di sviluppo e debug.

Jack di alimentazione: Arduino può essere alimentato tramite un alimentatore esterno collegato al jack di alimentazione. La tensione consigliata è di 7-12V; tensioni inferiori potrebbero non essere sufficienti, mentre tensioni superiori potrebbero surriscaldare il regolatore di tensione.

Pin VIN: Si può fornire alimentazione direttamente attraverso il pin VIN, specialmente quando si utilizza Arduino in un circuito più ampio.

Batterie: Per progetti portatili, Arduino può essere alimentato con batterie (ad esempio, un pacco di batterie AA o una batteria LiPo).

Considerazioni importanti sull'alimentazione:
• Regolatore di tensione: La scheda Arduino UNO contiene un regolatore di tensione che converte la tensione in ingresso (da USB, jack o VIN) a 5V stabilizzati.
• Pin di alimentazione: Arduino fornisce pin per alimentare componenti esterni: 5V, 3.
• Limitazioni di corrente: I pin di Arduino possono fornire o assorbire una quantità limitata di corrente (generalmente 20-40 mA per pin, con un totale di circa 500 mA per tutta la scheda con Arduino UNO). Progetti che richiedono correnti maggiori necessitano di circuiti di driver esterni.
• Protezione da sovracorrente: Alcune schede Arduino includono fusibili o protezioni contro le sovracorrenti, ma è sempre buona pratica verificare i limiti di corrente e adottare misure protettive aggiuntive quando necessario. 5 Componenti elettronici fondamentali

Per realizzare progetti con Arduino, è essenziale familiarizzare con alcuni componenti elettronici di base. Ecco i principali:

Resistori: Componenti passivi che limitano il flusso di corrente in un circuito. Vengono utilizzati per proteggere LED, creare partitori di tensione, pullup/pulldown, e altro ancora. Il valore di una resistenza si misura in ohm (:'• e viene identificato tramite codici colore o stampato direttamente sul componente. Vengono utilizzati per filtraggio, accoppiamento/disaccoppiamento, temporizzazione, e altro.

LED (Light Emitting Diode): Diodi che emettono luce quando attraversati da corrente. Hanno una polarità specifica (anodo e catodo) e necessitano di una resistenza in serie per limitare la corrente.

Transistor: Componenti semiconduttori che possono funzionare come interruttori o amplificatori. Sono fondamentali per controllare carichi che richiedono correnti maggiori di quelle fornibili direttamente da Arduino.

Diodi: Permettono il passaggio della corrente in una sola direzione. Utili per proteggere circuiti da inversioni di polarità o per realizzare circuiti logici.

Pulsanti e interruttori: Utilizzati per fornire input manuali. I pulsanti sono normalmente aperti (NO) o normalmente chiusi (NC) e richiedono spesso resistenze di pullup o pulldown.

Potenziometri e trimmer: Resistori variabili che permettono di regolare manualmente la resistenza. Utilizzati frequentemente come input analogici (ad esempio, per controlli di volume).

Sensori: Dispositivi che convertono grandezze fisiche in segnali elettrici. Esempi comuni sono sensori di temperatura, luce, umidità, distanza, movimento.

Attuatori: Dispositivi che convertono segnali elettrici in azioni fisiche. Esempi sono motori, servo, relay, solenoid. Possono essere display a LED, LCD (Liquid Crystal Display), OLED, e altri.

Breadboard: Non è un componente elettronico in senso stretto, ma è fondamentale per prototipare circuiti senza saldature. Permette di inserire componenti e fili, creando connessioni temporanee.

Jumper e cavi: Utilizzati per collegare componenti tra loro o alla scheda Arduino. 6 Shield e moduli di espansione

Gli shield sono schede di espansione progettate per essere inserite direttamente sopra Arduino, estendendone le funzionalità in modo semplice e standardizzato. I moduli, invece, sono componenti esterni che si collegano ad Arduino tramite cavi.

Caratteristiche degli shield:
• Si inseriscono direttamente sui pin di Arduino.
• Mantengono lo stesso form factor della scheda Arduino.
• Possono essere impilati (sebbene con alcune limitazioni, come conflitti di pin).

Shield comuni:
• Ethernet Shield: Aggiunge connettività di rete Ethernet.
• WiFi Shield: Fornisce connettività WiFi.
• Motor Shield: Facilita il controllo di motori (DC, passo-passo, servo).
• Display Shield: Integra display LCD o touchscreen.
• Data Logger Shield: Include lettore di schede SD e RTC (Real-Time Clock).
• Proto Shield: Fornisce un'area per saldare circuiti personalizzati.
• GSM/GPRS Shield: Aggiunge connettività cellulare.

Moduli comuni:
• Moduli sensore: Temperatura, umidità, pressione, distanza, movimento.
• Moduli di comunicazione: Bluetooth, RF, IR, NFC.
• Moduli display: LCD, OLED, segmenti.
• Moduli di controllo motori: Driver per vari tipi di motori.
• Moduli di alimentazione: Regolatori, battery management.
• Moduli di memorizzazione: SD card, EEPROM esterna.

Vantaggi dell'utilizzo di shield e moduli:
• Riducono la complessità del circuito.
• Spesso includono librerie dedicate che semplificano la programmazione.
• Permettono di aggiungere funzionalità complesse senza conoscenze approfondite di elettronica.

Considerazioni importanti:
• Compatibilità: Non tutti gli shield sono compatibili con tutte le schede Arduino, a causa di differenze nel layout dei pin o nelle tensioni operative.
• Conflitti di pin: Alcuni shield utilizzano gli stessi pin, rendendo impossibile il loro utilizzo simultaneo senza modifiche.
• Consumo energetico: Shield e moduli possono aumentare significativamente il consumo energetico del sistema.

• Ambiente di programmazione Arduino IDE

3. 1 Installazione e configurazione dell'IDE

L'Arduino IDE (Integrated Development Environment) è l'ambiente di sviluppo ufficiale per programmare le schede Arduino. x (la versione più recente con interfaccia rinnovata e funzionalità avanzate).

Procedura di installazione:
• Download:
• Visitare il sito ufficiale di Arduino (arduino. cc)
• Navigare alla sezione "Software"
• Scaricare la versione appropriata per il proprio sistema operativo (Windows, macOS, Linux)
• Installazione su Windows:
• Eseguire il file . exe scaricato
• Seguire le istruzioni della procedura guidata
• Installare i driver USB quando richiesto
• Installazione su macOS:
• Aprire il file . dmg scaricato
• Trascinare l'icona dell'Arduino IDE nella cartella Applicazioni
• Al primo avvio, potrebbe essere necessario autorizzare l'esecuzione di app scaricate da Internet
• Installazione su Linux:
• Decomprimere l'archivio scaricato
• Eseguire lo script di installazione o utilizzare il gestore pacchetti della propria distribuzione
• Configurare i permessi per le porte seriali (ad esempio, aggiungendo l'utente al gruppo "dialout")

Configurazione iniziale:
• Selezione della scheda:
• Dal menu "Strumenti" !’ "Scheda", selezionare il tipo di Arduino in uso (es. Arduino UNO)
• Selezione della porta:
• Dal menu "Strumenti" !’ "Porta", selezionare la porta seriale a cui è collegato Arduino
• Su Windows, sarà indicata come COMx (es. COM3)
• Su macOS e Linux, sarà indicata come /dev/ttyXXX (es. /dev/ttyUSB0)
• Installazione di librerie aggiuntive:
• Dal menu "Sketch" !’ "Includi libreria" !’ "Gestione librerie"
• Cercare e installare le librerie necessarie per i propri progetti

• Installazione di schede aggiuntive:
• Dal menu "Strumenti" !’ "Scheda" !’ "Gestore schede"
• Cercare e installare il supporto per schede Arduino non standard o di terze parti

Personalizzazione dell'IDE:
• Preferenze: Dal menu "File" !’ "Preferenze", è possibile modificare varie impostazioni come la dimensione del testo, il comportamento dell'editor, la posizione della cartella degli sketch.
• Scorciatoie da tastiera: È possibile utilizzare e personalizzare scorciatoie da tastiera per le operazioni più comuni. 2 Struttura dell'IDE

L'Arduino IDE presenta un'interfaccia semplice ma funzionale, progettata per essere accessibile anche ai principianti. Analizziamo le principali componenti dell'interfaccia:

Barra dei menu:
• File: Contiene comandi per aprire, salvare e gestire gli sketch, oltre alle preferenze dell'IDE.
• Modifica: Offre funzioni di editing come copia, incolla, ricerca.
• Sketch: Permette di verificare/compilare il codice, includere librerie, e altre operazioni sullo sketch.
• Strumenti: Contiene opzioni per selezionare la scheda, la porta, il programmatore, e altre impostazioni hardware.
• Aiuto: Fornisce accesso alla documentazione e alle risorse di supporto.

Barra degli strumenti: Contiene pulsanti per le operazioni più comuni:
• Verifica: Compila il codice senza caricarlo sulla scheda.
• Carica: Compila il codice e lo carica sulla scheda Arduino.
• Salva: Salva lo sketch corrente.
• Monitor seriale: Apre il monitor seriale per comunicare con Arduino.

Area di editing: È lo spazio principale dove si scrive il codice. Supporta evidenziazione della sintassi, indentazione automatica e altre funzionalità tipiche degli editor di codice.

Area dei messaggi: Mostra i messaggi del compilatore, inclusi errori e avvisi durante la compilazione e il caricamento.

Barra di stato: Visualizza informazioni come la scheda e la porta selezionate.

Caratteristiche dell'editor:
• Indentazione automatica: Aiuta a mantenere il codice ordinato.
• Evidenziazione della sintassi: Colora differentemente parole chiave, variabili, commenti per migliorare la leggibilità.
• Numerazione delle righe: Facilita il riferimento a specifiche parti del codice.
• Ricerca e sostituzione: Permette di trovare e modificare testo in tutto il codice.

Strumenti aggiuntivi:
• Monitor Seriale: Permette di visualizzare i dati inviati da Arduino al computer via seriale e di inviare dati ad Arduino. 6): Visualizza in forma grafica i dati ricevuti via seriale.
• Esempi: Una raccolta di sketch precompilati che dimostrano l'uso di varie funzionalità. x include un debugger (su schede supportate). x ha una gestione librerie e schede migliorata. 3 Il

linguaggio di programmazione Arduino

Il linguaggio di programmazione Arduino, spesso chiamato "Arduino Language", è in realtà un insieme di funzioni C/C++ raccolte in un framework che semplifica la programmazione dei microcontrollori. Questo linguaggio è stato progettato per essere accessibile anche a chi non ha esperienza pregressa di programmazione, pur mantenendo la potenza e la flessibilità di C/C++.

Caratteristiche principali:
• Basato su C/C++: Il codice Arduino è essenzialmente C/C++ con alcune semplificazioni e funzioni specifiche.
• Astrazioni hardware: Funzioni di alto livello che nascondono la complessità del controllo diretto dell'hardware.
• Librerie estese: Una vasta collezione di librerie che facilitano l'interazione con vari componenti e periferiche.
• Sintassi semplificata: Alcune caratteristiche di C/C++ sono semplificate per facilitarne l'apprendimento.

Elementi fondamentali della sintassi:
• Tipi di dati:
• int: Numeri interi (16 bit su Arduino UNO)
• long: Numeri interi a 32 bit
• float: Numeri a virgola mobile
• double: Generalmente uguale a float su Arduino AVR
• char: Singolo carattere
• byte: Valori da 0 a 255
• boolean: true o false
• String: Sequenza di caratteri (tipo di classe)
• Array: Collezione di valori dello stesso tipo
• Strutture di controllo:
• if, else, else if: Per l'esecuzione condizionale
• for, while, do-while: Per l'iterazione
• switch-case: Per la selezione multipla
• break, continue: Per il controllo del flusso all'interno dei cicli

Dispensa di Studio Introduttiva ad Arduino

Questa dispensa si basa sui contenuti della playlist di tutorial introduttivi ad Arduino fornita, con l'obiettivo di presentare in maniera chiara e dettagliata i concetti fondamentali e le applicazioni pratiche di questa piattaforma.

Sezione 1: Introduzione ad Arduino e Panoramica Generale

Arduino è una scheda elettronica nata in Italia ed è un progetto completamente Open Source, sia per la parte hardware che software.

L'ambiente di sviluppo Arduino IDE serve per scrivere i programmi (chiamati sketch) che vengono poi trasferiti sulla scheda per l'esecuzione. Essendo un progetto Open Source, è possibile consultare sia il codice sorgente del software che gli schemi circuitali della scheda.

Arduino è stato pensato per scopi didattici e per realizzare facilmente prototipi funzionanti. Tuttavia, il suo utilizzo si estende anche a settori come la robotica e la domotica.

Tipologie di schede Arduino

Le schede più utilizzate sono:
• Arduino Uno (la più comune)
• Arduino Mega (con più porte I/O e memoria)
• Arduino Leonardo (con USB nativa)
• Arduino LilyPad (adatta a indumenti smart)

Sezione 2: Componenti Hardware e Loro Funzionalità

Componenti della scheda Arduino Uno
• Microcontrollore ATmega: il cuore della scheda.
• Porta USB: per la programmazione e l'alimentazione.
• Jack di alimentazione: per alimentare la scheda da fonte esterna (6-20V, consigliato 7-12V). 3V: per alimentare altri componenti.
• Vin: per fornire tensione esterna.
• Pin di ingresso analogico (A0-A5): ricevono segnali analogici (0-5V, valori 0-1023).
• Pin di I/O digitale (0-13): configurabili come ingresso/uscita digitale (HIGH/LOW), alcuni supportano PWM (~).
• Pulsante di reset: riavvia il programma sulla scheda.
• LED indicatori:
• ON: indica l'alimentazione.
• TX/RX: lampeggiano durante la comunicazione seriale.
• LED utente (collegato al PIN 13): utilizzabile nei progetti.

Componenti esterni utili
• Breadboard: basetta con fori per il montaggio dei circuiti senza saldature.
• Pulsanti: necessitano di resistenze pull-up/down per funzionare correttamente.
• Resistori: regolano il flusso di corrente (valori in Ohm :’’à
• Potenziometri e trimmer: resistenze variabili per segnali analogici.
• Fotoreistenze: variano la resistenza in base alla luce.

Sezione 3: Panoramica sull'Ambiente di Programmazione

L'IDE di Arduino è disponibile per Windows, macOS e Linux.

Struttura base di un programma (Sketch)

Ogni sketch Arduino deve contenere due funzioni principali:

Funzionalità principali dell'IDE
• Verifica ('): controlla il codice per errori.
• Carica (!'): trasferisce il codice sulla scheda.
• Monitor Seriale: permette la comunicazione seriale con la scheda.
• Esempi predefiniti: forniti nel menu File > Esempi.

Concetti chiave del linguaggio Arduino (C/C++)
• Variabili: memorizzano dati (es.
• Librerie (`#include`): estendono le funzionalità.

Sezione 4: Esempi Pratici e Applicazioni di Base
• Lampeggio del LED Integrato (PIN 13)

Concetti: output digitale, `digitalWrite()`, `delay()`.

Codice

Circuito: LED integrato, nessun componente aggiuntivo.

• Controllo di un LED Esterno

Concetti: uso della breadboard, resistenze, output digitale variabile.

• Controllo di un LED con un Pulsante

Concetti: input digitale, `digitalRead()`, resistenze pull-down.

Circuito: pulsante tra PIN 2 e 5V, resistore pull-down da 100k:' `erso GND.

Glossary

ambiente
: Definition for "ambiente" would be provided here.

analisi
: Definition for "analisi" would be provided here.

analogici
: Definition for "analogici" would be provided here.

arduino
: Definition for "arduino" would be provided here.

autovalutazione
: Definition for "autovalutazione" would be provided here.

che
: Definition for "che" would be provided here.

codice
: Definition for "codice" would be provided here.

componenti
: Definition for "componenti" would be provided here.

comunicazione
: Definition for "comunicazione" would be provided here.

con
: Definition for "con" would be provided here.

concetti
: Definition for "concetti" would be provided here.

critica
: Definition for "critica" would be provided here.

del
: Definition for "del" would be provided here.

della
: Definition for "della" would be provided here.

digitale
: Definition for "digitale" would be provided here.

digitali
: Definition for "digitali" would be provided here.

esempi
: Definition for "esempi" would be provided here.

esercitazioni
: Definition for "esercitazioni" would be provided here.

essere
: Definition for "essere" would be provided here.

funzionalit
: Definition for "funzionalit" would be provided here.

hardware
: Definition for "hardware" would be provided here.

ide
: Definition for "ide" would be provided here.

introduzione
: Definition for "introduzione" would be provided here.

led
: Definition for "led" would be provided here.

per
: Definition for "per" would be provided here.

pin
: Definition for "pin" would be provided here.

possono
: Definition for "possono" would be provided here.

pratici
: Definition for "pratici" would be provided here.

programmazione
: Definition for "programmazione" would be provided here.

pulsante
: Definition for "pulsante" would be provided here.

pwm
: Definition for "pwm" would be provided here.

resistenza
: Definition for "resistenza" would be provided here.

resistenze
: Definition for "resistenze" would be provided here.

scheda
: Definition for "scheda" would be provided here.

schede
: Definition for "schede" would be provided here.

segnali
: Definition for "segnali" would be provided here.

sezione
: Definition for "sezione" would be provided here.

shield
: Definition for "shield" would be provided here.

sketch
: Definition for "sketch" would be provided here.

sono
: Definition for "sono" would be provided here.

sulla
: Definition for "sulla" would be provided here.

temperatura
: Definition for "temperatura" would be provided here.

uno
: Definition for "uno" would be provided here.

utilizzati

Definition for "utilizzati" would be provided here.