



UNIVERSITÀ DEGLI STUDI DELL'INSUBRIA

DIPARTIMENTO DI SCIENZE TEORICHE E APPLICATE
CORSO DI STUDIO TRIENNALE IN INFORMATICA - F004

Sviluppo di un sistema embedded per il controllo della temperatura in una camera di collaudo

Relatore:

Prof. Carlo Dossi

Tutor Aziendale:

Edoardo Scaglia

Tesi di Laurea di:

Mattia Papaccioli - 747053

Azienda ospitante:

AMEL SRL

Anno Accademico:

2025/2026

Indice

1	Introduzione	1
1.1	Abstract	1
1.2	Struttura del codice sorgente	2
2	Control	3
2.1	Graphical User Interface	3
2.1.1	Funzioni di callback nel ciclo principale della GUI LVGL	4
2.1.2	Albero dei widget nel ciclo principale	5
2.2	Compilazione della GUI	6
2.2.1	<code>cross_compile_setup.cmake</code>	6
2.3	Admin Control	7
2.4	Logging and Monitoring	7
3	Embedded Linux	8
3.1	Hardware	8
3.2	Costruzione del sistema	8
3.3	Personalizzazione di buildroot	9
3.3.1	Il pacchetto amel-temp-control	9
3.3.2	Il pacchetto amel-pid	9
3.3.3	Modifica pacchetti networking	10
3.4	File I/O	10
3.5	PID Control	10
3.6	MODBUS RTU	10
4	Conclusione	11
5	Einleitung	12
6	Erläuterungen	13
6.1	Ausdrücke und Abkürzungen	13
6.2	Listen	13
6.3	Abbildungen und Tabellen	14
6.3.1	Abbildungen	14
6.3.2	Tabellen	14
6.4	Programm Quellcode	14
6.5	Verweise	15
7	Fazit	16
A	Riferimenti	17
B	Glossario	18

1

Introduzione

1.1 Abstract

La camera di collaudo, situata presso AMEL s.r.l., è utilizzata per testare carichi resistivi. Questi producono calore, rendendo necessario il raffreddamento dell'ambiente, soprattutto nei mesi più caldi.

Il sistema embedded controlla la frequenza di una ventola di raffreddamento mediante un controllo PID:

1. La temperatura ambiente viene letta tramite un sensore.
2. Questa viene utilizzata come ingresso del sistema PID.
3. L'algoritmo PID calcola la tensione da inviare a un inverter, il quale determina la frequenza della ventola in base all'errore attuale (azione proporzionale) e a quello accumulato (azione integrativa).
4. Questo sistema di retroazione negativa viene applicato continuamente per mantenere costante la temperatura ambiente.

La comunicazione tra l'inverter e il sistema embedded avviene tramite il protocollo MODBUS RTU.

Il sistema embedded fornisce inoltre un'interfaccia per regolare la temperatura target dell'ambiente mediante un display touchscreen. La GUI è sviluppata utilizzando la libreria LVGL.

Il sistema embedded è collegato alla rete aziendale tramite Ethernet; attraverso un web server sarà possibile regolare la temperatura target anche da remoto.

La comunicazione tra il sistema embedded e il web server avviene mediante scrittura su file o IPC (Inter-Process Communication).

È ancora da valutare l'impiego di un database per registrare la temperatura nel tempo. In tal caso, il web server gestirà l'interazione con esso.

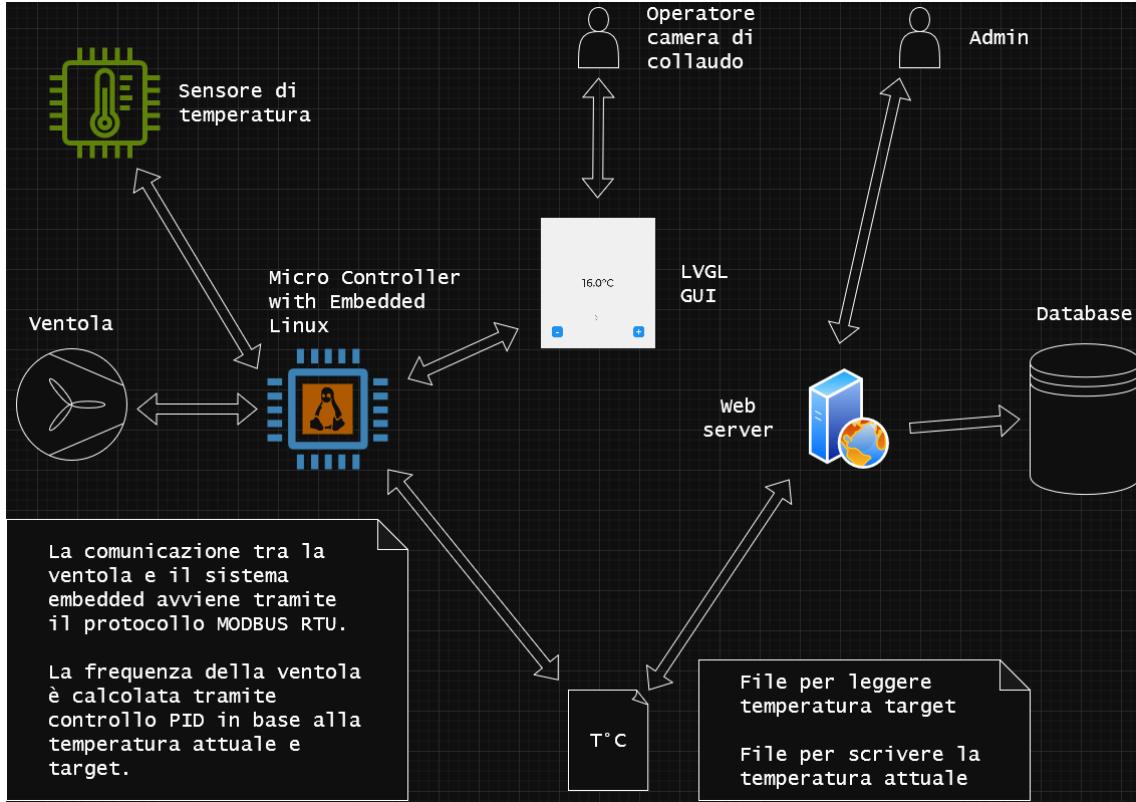


Figura 1 — Diagramma che illustra il sistema

1.2 Struttura del codice sorgente

Il codice sorgente del progetto è pubblicato su GitHub [1]. Questa repository è composta da più sottomoduli:

- tesi-gui [2]
- tesi-overleaf [3]
- tesi-modbus

Questo approccio è stato adottato per garantire modularità e organizzazione del software [4].

2

Control

2.1 Graphical User Interface

Per controllare la temperatura della camera di collaudo, l'operatore imposta la temperatura target mediante un display touchscreen «NOME DISPLAY». L'interfaccia grafica è sviluppata utilizzando la libreria LVGL [5] e si è utilizzato un template [6] contenente il porting su Linux fornito dagli sviluppatori della libreria.

2.1.1 Funzioni di callback nel ciclo principale della GUI LVGL

```
1 // TODO CHANGEME READ THIS FROM A FILE
2 static float target_temperature = read_temperature_file();
3 lv_obj_t * screen;
4 lv_obj_t * target_temperature_label;
5
6 const int padding_button = 50;
7 const int height_button = 50;
8 const int width_button = 50;
9
10 static void increment_temperature(lv_event_t * e)
11 {
12     if(lv_event_get_code(e) != LV_EVENT_CLICKED) {
13         return;
14     }
15     target_temperature++;
16     lv_label_set_text_fmt(
17         target_temperature_label, "%.1f°C", target_temperature
18     );
19 }
20
21 static void decrement_temperature(lv_event_t * e)
22 {
23     if(lv_event_get_code(e) != LV_EVENT_CLICKED) {
24         return;
25     }
26     target_temperature--;
27     lv_label_set_text_fmt(
28         target_temperature_label, "%.1f°C", target_temperature
29     );
30 }
```

Codice 1 – Funzioni di callback per i bottoni di incremento e decremento della temperatura target

2.1.2 Albero dei widget nel ciclo principale

```

1 screen = lv_scr_act();
2 lv_obj_t * increment_temperature_button = lv_btn_create(screen);
3 lv_obj_align(
4     increment_temperature_button, LV_ALIGN_BOTTOM_RIGHT,
5     -padding_button, -padding_button);
6 lv_obj_set_height(increment_temperature_button, height_button);
7 lv_obj_set_width(increment_temperature_button, width_button);
8 lv_obj_add_event_cb(
9     increment_temperature_button, increment_temperature,
10    LV_EVENT_ALL, NULL
11 );
12
13 lv_obj_t * increment_temperature_label = lv_label_create(
14     increment_temperature_button
15 );
16 lv_label_set_text(increment_temperature_label, "+");
17 lv_obj_set_style_text_font(
18     increment_temperature_label, &lv_font_montserrat_48, 0
19 );
20 lv_obj_center(increment_temperature_label);
21
22 lv_obj_t * decrement_temperature_button = lv_btn_create(screen);
23 lv_obj_align(
24     decrement_temperature_button, LV_ALIGN_BOTTOM_LEFT,
25     padding_button, -padding_button);
26 lv_obj_set_height(decrement_temperature_button, height_button);
27 lv_obj_set_width(decrement_temperature_button, width_button);
28 lv_obj_add_event_cb(
29     decrement_temperature_button, decrement_temperature,
30    LV_EVENT_ALL, NULL
31 );
32
33 lv_obj_t * decrement_temperature_label = lv_label_create(
34     decrement_temperature_button
35 );
36 lv_label_set_text(decrement_temperature_label, "-");
37 lv_obj_set_style_text_font(
38     decrement_temperature_label, &lv_font_montserrat_48, 0
39 );
40 lv_obj_center(decrement_temperature_label);
41
42 target_temperature_label = lv_label_create(screen);
43 lv_label_set_text_fmt(
44     target_temperature_label, "%.1f°C", target_temperature
45 );
46 lv_obj_set_style_text_font(
47     target_temperature_label, &lv_font_montserrat_48, 0
48 );
49 lv_obj_align(target_temperature_label, LV_ALIGN_CENTER, 0, 0);

```

Codice 2 – Creazione dei widget nell'interfaccia grafica

I backend utilizzati da LVGL per l'I/O sono libevdev e il framebuffer device. Sono stati scelti per la loro semplicità e il ridotto utilizzo di risorse.

Libevdev [7] è una libreria che gestisce gli eventi di input: riceve i tocchi dal touchscreen e li passa all’interfaccia grafica.

Il framebuffer device è semplicemente il file `/dev/fb0`, scritto dalla GUI, che contiene il colore di ciascun pixel dello schermo.

2.2 Compilazione della GUI

Per la compilazione dell’applicazione è necessaria una toolchain adatta all’architettura ARM. Nel nostro caso, ci affidiamo al compilatore e alle librerie fornite da Buildroot.

2.2.1 cross_compile_setup.cmake

```
1 set(CMAKE_SYSTEM_NAME Linux)
2 set(CMAKE_SYSTEM_PROCESSOR arm)
3
4 set(tools ~/buildroot/output/host/bin/arm-buildroot-linux-gnueabihf-)
5 set(CMAKE_C_COMPILER ${tools}gcc)
6 set(CMAKE_CXX_COMPILER ${tools}g++)
7
8 set(EVDEV_INCLUDE_DIRS ~/buildroot/output/staging/usr/include/libevdev/)
9 set(EVDEV_LIBRARIES ~/buildroot/output/staging/usr/lib/libevdev.so)
10
11 set(BUILD_SHARED_LIBS ON)
```

Codice 3 – cross_compile_setup.cmake

Il comando `cmake -DCMAKE_TOOLCHAIN_FILE=./cross_compile_setup.cmake -B build -S .` genera i Makefile necessari per la cross-compilazione, che vengono poi eseguiti con `make -C build -j`.

LVGL viene compilata come libreria condivisa, mentre l’applicazione come eseguibile.

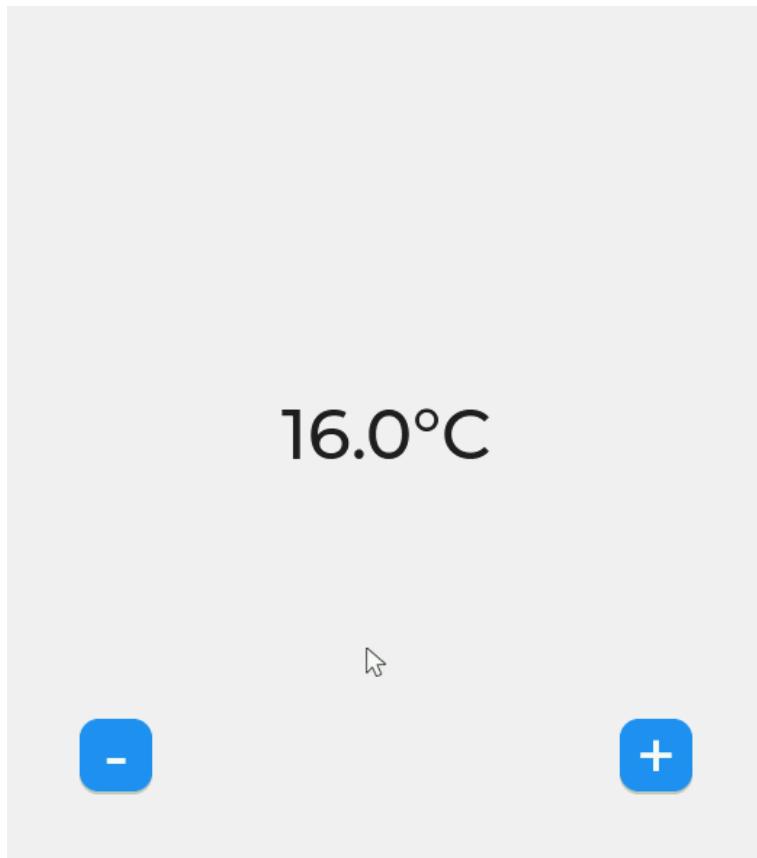


Figura 2 – Interfaccia grafica per il controllo della temperatura

2.3 Admin Control

2.4 Logging and Monitoring

3

Embedded Linux

3.1 Hardware

La scheda utilizzata per il sistema embedded è sviluppata da AMEL e comprende:

- una host-board Ganador (rev. 4);
- un system-on-module Vulcano-A5;
- una CPU ARM;
- una memoria SD utilizzata come disco fisso;
- un'interfaccia Ethernet;
- un'interfaccia seriale;
- un display touchscreen.

3.2 Costruzione del sistema

Per orchestrare il sistema è stato utilizzato Linux. È stato creato un kernel personalizzato con soli i moduli essenziali per il funzionamento, data la limitatezza delle risorse hardware. Lo strumento utilizzato per completare l'opera è Buildroot [8], che consiste essenzialmente in una serie di Makefile per installare e cross-compilare tutte le librerie e i pacchetti necessari alla costruzione e all'esecuzione del sistema. Inoltre, si occupa di creare il filesystem e di prepararlo in un'immagine pronta per essere scritta sulla scheda SD del sistema embedded.

Per aggiungere l'interfaccia grafica sviluppata con LVGL è stato necessario creare un nuovo pacchetto in Buildroot.

All'avvio del sistema, il bootloader del chip attiva AT91bootstrap, che a sua volta avvia Barebox, il quale carica il kernel in memoria.

3.3 Personalizzazione di buildroot

Per aggiungere un pacchetto a Buildroot è necessario inserire una nuova voce nella cartella package, comprendente un file `Config.in` e un file `.mk`.

Questi due file contengono le istruzioni che consentono a Buildroot di risolvere le dipendenze, scaricare e installare il pacchetto nel filesystem del dispositivo target.

3.3.1 Il pacchetto amel-temp-control

```

1 AMEL_TEMP_CONTROL_VERSION = 44c17c6f2c492f1f3c7d8a6767df390c8d13eb9c
2 AMEL_TEMP_CONTROL_SITE = git@git.amelchem.com:mpapaccioli/temp-
3 control.git
4 AMEL_TEMP_CONTROL_SITE_METHOD = git
5 AMEL_TEMP_CONTROL_DEPENDENCIES = libevdev
6 AMEL_TEMP_CONTROL_GIT_SUBMODULES = YES
7
8 define AMEL_TEMP_CONTROL_BUILD_CMDS
9   cmake -DCMAKE_TOOLCHAIN_FILE=$(@D)/user_cross_compile_setup.cmake \
10         -B $(@D)/build -S $(@D)
11   make -C $(@D)/build -j
12
13 endef
14
15 define AMEL_TEMP_CONTROL_INSTALL_TARGET_CMDS
16   $(INSTALL) -d $(TARGET_DIR)/opt/amel-temp-control/
17   cp $(@D)/build/bin/lvglSim $(TARGET_DIR)/opt/amel-temp-control/main
18   cp -r $(@D)/build/lvgl/lib/* $(TARGET_DIR)/usr/lib
19
20 endef
21
22 $(eval $(generic-package))

```

Codice 4 – amel-temp-control.mk

Inizialmente, viene clonata la repository temp-control, contenente la GUI, al commit specificato, inizializzando i sottomoduli e verificando la presenza della dipendenza `libevdev`.

Successivamente, vengono cross-compilate la libreria LVGL e l'applicazione con interfaccia grafica utilizzando il compilatore ARM fornito da Buildroot.

Infine, i binari della libreria e l'eseguibile dell'applicazione vengono installati sulla macchina target.

3.3.2 Il pacchetto amel-pid

Analogamente, è stato creato il pacchetto `amel-pid-control` per cross-compilare e installare la libreria PID sviluppata in C++.

3.3.3 Modifica pacchetti networking

Dopo aver ripetuto questo processo per tutti i pacchetti desiderati, il filesystem e l'immagine del kernel vengono assemblati in un file `sdcards.img` pronto per essere scritto su una scheda SD e avviato sul dispositivo embedded.

3.4 File I/O

3.5 PID Control

3.6 MODBUS RTU

4

Conclusione

5

Einleitung

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliquam quaerat voluptatem. Ut enim aequo doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos iridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliquam quaerat voluptatem. Ut enim aequo doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos iridente, statua est in quo a nobis philosophia defensa et.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliquam quaerat voluptatem. Ut enim aequo doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos iridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum.



Erläuterungen

Im folgenden werden einige nützliche Elemente und Funktionen zum Erstellen von Typst-Dokumenten mit diesem Template erläutert.

6.1 Ausdrücke und Abkürzungen

Verwende die `gls`-Funktion, um Ausdrücke aus dem Glossar einzufügen, die dann dorthin verlinkt werden. Ein Beispiel dafür ist:

Im diesem Kapitel wird eine [Softwareschnittstelle](#) beschrieben. Man spricht in diesem Zusammenhang auch von einem [Application Programming Interface \(API\)](#). Die Schnittstelle nutzt Technologien wie das [Hypertext Transfer Protocol \(HTTP\)](#).

Das Template nutzt das `glossarium`-Package für solche Glossar-Referenzen. In der zugehörigen [Dokumentation](#) werden noch weitere Varianten für derartige Querverweise gezeigt. Dort ist auch im Detail erläutert, wie das Glossar aufgebaut werden kann.

6.2 Listen

Es gibt Aufzählungslisten oder nummerierte Listen:

- Dies
- ist eine
- Aufzählungsliste
 1. Und
 2. hier wird
 3. alles nummeriert.

6.3 Abbildungen und Tabellen

Abbildungen und Tabellen (mit entsprechenden Beschriftungen) werden wie folgt erstellt.

6.3.1 Abbildungen



Figura 3 – Eine Abbildung

6.3.2 Tabellen

	Area	Parameters
cylinder.svg	$\pi h \frac{D^2 - d^2}{4}$ (1)	h : height D : outer radius d : inner radius
tetrahedron.svg	$\frac{\sqrt{2}}{12} a^3$ (2)	a : edge length

Tabella 1 – Eine Tabelle

6.4 Programm Quellcode

Quellcode mit entsprechender Formatierung wird wie folgt eingefügt:

```

1 const ReactComponent = () => {
2   return (
3     <div>
4       <h1>Hello World</h1>
5     </div>
6   );
7 }
8 export default ReactComponent;

```

Codice 5 – Ein Stück Quellcode

```
1 def hello_world():
2     print("Hello, World!")
```

6.5 Verweise

Für Literaturverweise verwendet man die `cite`-Funktion oder die Kurzschreibweise mit dem @-Zeichen:

Tabellen, Abbildungen und andere Elemente können mit einem Label in spitzen Klammern gekennzeichnet werden (die Tabelle oben hat z.B. das Label `<table>`). Sie kann dann mit `@table` referenziert werden. Das ergibt im konkreten Fall: Tabella 1



Fazit

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliquam quaerat voluptatem. Ut enim aequo doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliquam quaerat voluptatem. Ut enim aequo doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distingue possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitibus aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliquam quaerat voluptatem. Ut enim aequo doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distingue possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et.

A Riferimenti

- [1] M. Papaccioli, «Tesi». [Online]. Disponibile su: <https://github.com/sbOogway/tesi>
- [2] M. Papaccioli, «GUI». [Online]. Disponibile su: <https://github.com/sbOogway/tesi-gui>
- [3] M. Papaccioli, «LaTeX source code». [Online]. Disponibile su: <https://github.com/sbOogway/tesi-overleaf>
- [4] G. contributors, «7.11 Git Tools - Submodules». [Online]. Disponibile su: <https://git-scm.com/book/en/v2/Git-Tools-Submodules>
- [5] L. Contributors, «LVGL». [Online]. Disponibile su: <https://github.com/lvgl/lvgl>
- [6] L. Contributors, «Linux template for LVGL». [Online]. Disponibile su: https://github.com/lvgl/lv_port_linux
- [7] [Online]. Disponibile su: <https://www.freedesktop.org/wiki/Software/libevdev/>
- [8] B. contributors, [Online]. Disponibile su: <https://buildroot.org/>

B Glossario

API – Application Programming Interface [13](#)

HTTP – Hypertext Transfer Protocol [13](#)

Softwareschnittstelle: Ein logischer Berührungs punkt in einem Softwaresystem: Sie ermöglicht und regelt den Austausch von Kommandos und Daten zwischen verschiedenen Prozessen und Komponenten. [13](#)