



**UNIVERSITÀ  
DEGLI STUDI  
DI PADOVA**



**DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE**

**CORSO DI LAUREA IN INGEGNERIA ELETTRONICA**

**Sviluppo di un sistema di trasmissione dati basato su tecnologia satellitare IoT  
per applicazioni di monitoraggio ambientale**

**Relatore: Prof. / Dott. Alessandro Pozzebon**

**Laureando/a: Alberto Cardin**

**ANNO ACCADEMICO 2021 – 2022**

**Data di laurea 23/09/2022**



## Abstract

Questo studio punta alla realizzazione di un sistema basato su un modem sviluppato da SWARM, per la trasmissione di piccole quantità di dati via satellite da ambienti remoti nei quali non vi siano altre possibili soluzioni di connettività. SWARM è un'azienda che sviluppa sistemi per connettività low cost per dispositivi IoT. Al momento conta una costellazione piuttosto limitata di satelliti che, sebbene garantisca copertura sull'intera superficie terrestre, permette di avere finestre di trasmissione ad orari e con tempi differenti in base alla latitudine alla quale verrà posto il sistema. Un'altra limitazione è legata al fatto che i dati vengono conservati dal satellite fino a quando quest'ultimo non abbia la possibilità di intercettare una stazione terrestre di ricezione. Tali vincoli comportano un tempo di latenza tra acquisizione del dato e ricezione da parte dell'utente finale non indifferente. Per tale ragione al momento tali sistemi sono particolarmente indicati per il monitoraggio ambientale; tuttavia, non è da escludere che in futuro saranno adatti anche per sistemi real-time. In particolare, il dispositivo impiegato per questo studio è il modem TILE01 che opera alla tensione di 3.3 V e, attraverso un protocollo di comunicazione UART, può essere interfacciato con diversi dispositivi. In questa applicazione verrà connesso con un modulo ESP-32 che provvederà ad interfacciare il modem con sensori esterni. È stata scelta tale controllore in quanto ha una logica a 3.3 V che non richiede adattamento dei livelli logici, ed è dotato di una potenza di calcolo che lo rende performante anche per applicazioni più complesse.



# Indice

<b>1</b>	<b>Introduzione</b>	<b>1</b>
1.1	Internet of Things (IoT)	1
1.2	Sistemi embedded	1
1.3	La flessibilità dei sistemi	2
1.4	il mercato dei sistemi	3
1.5	Monitoraggio ambientale legato all'IoT	3
<b>2</b>	<b>Il sistema di trasmissione satellitare</b>	<b>5</b>
2.1	Obiettivo della tesi	5
2.2	Approccio al sistema	5
2.3	Hardware	6
2.4	schema elettrico	6
2.4.1	ESP-32	7
2.4.2	TILE-01	8
2.4.3	Antenne	10
2.4.4	Sensori	10
2.4.5	Alimentazione	14
<b>3</b>	<b>Protocolli di comunicazione seriale</b>	<b>17</b>
3.1	Comunicazione seriale asincrona	17
3.2	I2C (Inter Integrated Circuit)	18
<b>4</b>	<b>Software</b>	<b>21</b>
4.1	Ambiente di sviluppo	21
4.2	Librerie	21
4.3	La Macchina a stati finiti	22
4.4	La simulazione ed il codice	22
4.4.1	Batteria scarica	23
4.4.2	Inizializzazione	23

4.4.3	Impacchettamento messaggio . . . . .	23
4.4.4	Attesa invio messaggio . . . . .	24
4.4.5	Misurazione valori e sleep . . . . .	25
	<b>Bibliografia</b>	<b>29</b>

# Capitolo 1

## Introduzione

### 1.1 Internet of Things (IoT)

Il termine "Internet of Things" ha origine nel 1999 dal titolo di una presentazione tenuta dall'innovatore tecnologico Kevin Ashton.

Tuttavia, per questo termine non esiste una definizione univoca, anche se il concetto condiviso dalla comunità scientifica si basa sul fatto che se originariamente internet fosse concepito per i dati creati dagli umani, il futuro di internet si basi su dati creati da cose (casa domotica, contatore intelligente, veicolo elettrico, frigo, logistica, ecc.). Riprendendo quanto detto da Kevin Ashton in un articolo pubblicato presso l'RFID journal [1], le informazioni tecnologiche dipendono così tanto dai dati creati dalle persone, che i computer conoscono meglio le idee che le cose. Se invece vi fossero computer che sapessero ciò che dovrebbero sapere, senza l'aiuto umano, si sarebbe in grado di monitorare e controllare tutto, riducendo sprechi e costi.

### 1.2 Sistemi embedded

I sistemi embedded sebbene rappresentino la maggior parte dei sistemi di elaborazione spesso possono passare inosservati in quanto non hanno bisogno di interazione da parte degli esseri umani. Questi sistemi vengono sviluppati per applicazioni specifiche e si integrano perfettamente con il concetto dell'internet of things. Sono caratterizzati da quattro elementi chiave: alimentazione, elaborazione, memorizzazione, interazione con l'ambiente. I vincoli di progetto a cui può essere sottoposta la progettazione di un sistema embedded possono essere di diverse tipologie, tuttavia è il time-to-market (tempo che si impiega per proget-

tare e mettere in commercio il dispositivo) che tende a vincolare maggiormente le scelte di progetto. Altri aspetti dei quali si tiene conto sono:

- **Peso e Dimensioni:** avere dimensioni e peso contenute soprattutto se il dispositivo deve essere trasportabile;
- **Costo:** determinante per le produzioni in grandi volumi;
- **Consumo energetico:** essendo i dispositivi il più possibile autonomi è un fattore determinante da tenere in considerazione durante la progettazione sia sul lato hardware che software;
- **Prestazioni:** in genere sotto tale aspetto i vincoli sono legati al tempo di reazione a un evento e al tempo di gestione dell'evento;
- **Dimensione del codice:** essendo il codice spesso memorizzato all'interno della memoria del chip deve essere il più possibile contenuto per limitare i costi;
- **Funzionamento in tempo reale:** nel caso in cui determinate operazioni debbano essere iniziate o completate entro un tempo prefissato;
- **Affidabilità:** spesso i vincoli sotto tale aspetto sono dettati da norme di certificazione che prevedono analisi di potenziali guasti;
- **Safety:** è la possibilità che a fronte di un guasto il dispositivo arrechi un danno alle persone o alle cose con cui interagisce;
- **Sicurezza:** è la capacità del sistema di mantenere le informazioni da esso raccolte al sicuro e verificarle.

Tali aspetti sono estremamente legati al tipo di applicazione a cui è destinato il dispositivo, per cui non esistono standard prestabiliti.

### **1.3 La flessibilità dei sistemi**

Considerato che il Time-to-market è un aspetto chiave, la flessibilità si rivela un aspetto chiave per questi sistemi in quanto un cambio di specifica al progetto, se il sistema non fosse flessibile, comporterebbe un allungamento dei tempi di progetto e sviluppo. Tuttavia la presenza sul mercato di diverse famiglie di microcontrollori e sensori permette di avere una vasta scelta e una grande flessibilità.

## 1.4 il mercato dei sistemi

Nel 2020 il mercato dei sistemi embedded si stima avesse un valore di 86.5 miliardi di dollari, e si stima raggiunga i 116.2 miliardi di dollari entro il 2025, con un tasso annuo di crescita composto del 6.1%.

Il settore che dovrebbe crescere maggiormente in questi anni dovrebbe essere il segmento dell'industria automobilistica. Ciò è dovuto all'aumento della domanda di veicoli ibridi ed elettrici, che sfruttano ampiamente la tecnologia ADAS (Advanced Driver Assistance Systems).

## 1.5 Monitoraggio ambientale legato all'IoT

Secondo l'EEA (European Environment Agency) il monitoraggio ambientale è la: "Misurazione, valutazione e determinazione periodica e/o continua di parametri ambientali e/o livelli di inquinamento al fine di prevenire effetti negativi e dannosi per l'ambiente. Include anche la previsione di possibili cambiamenti nell'ecosistema e/o nella biosfera nel suo insieme." [6]. Per monitoraggio ambientale si intende quindi l'analisi dell'aria, del suolo, dell'acqua sia in ambiente urbano che come strumento di valutazione dell'impatto ambientale.

Principalmente tra i sistemi integrati IoT più diffusi si trovano centraline che raccolgono dati da sensori specifici e, collegate via Wi-Fi o via cavo li trasmettono in cloud consentendo spesso anche un accesso remoto ad esse per la calibrazione o la diagnostica. In genere l'elaborazione dei dati viene effettuata sia dal dispositivo stesso che a livello di rete per calcoli più complessi, non a caso l'elaborazione avviene spesso attraverso algoritmi di AI e machine learning. È ormai largamente diffuso l'uso della tecnologia LoRaWAN per la trasmissione di dati a grande distanza: uno dei limiti di questa tecnologia è che spesso non è disponibile copertura in ambienti remoti come per esempio in mare al largo dalla costa o in alta montagna. È in questi ambienti che la tecnologia satellitare occorre in soccorso, per esempio con la tecnologia utilizzata in questo studio.



# Capitolo 2

## Il sistema di trasmissione satellitare

### 2.1 Obiettivo della tesi

L'obiettivo di questa tesi punta allo sviluppo di un sistema di monitoraggio ambientale per ambienti difficilmente accessibili che sfrutti una trasmissione satellitare per l'invio dei dati, integrato quindi in ambiente IoT. A causa tuttavia di complicazioni, legate allo stato del piano dati stipulato con il rivenditore della scheda ed al rilascio di un aggiornamento firmware lato modem e satellite, non si è riusciti ad effettuare un invio dei dati tramite satellite. Per ovviare a ciò, essendo stati i tempi di sviluppo molto stretti, si è optato per una simulazione che sfruttasse la connettività Wi-Fi del controllore per simulare un invio dei dati nel momento in cui il modem satellitare avesse ricevuto il segnale di test dal satellite. Essendo comunque stato progettato in un'ottica di applicazioni per ambienti remoti, nella fase di progettazione si è tenuto in considerazione l'aspetto della riduzione del consumo energetico.

### 2.2 Approccio al sistema

L'aspetto chiave attorno al quale il progetto doveva svilupparsi stava nella trasmissione satellitare, e quindi nella scheda Tile 01 commercializzata da SWARM. Attorno al funzionamento di essa si è quindi scelto un controllore adatto, per ottenere massima flessibilità e versatilità è stato scelto come un ESP-32. Come sensori sono stati scelti invece tre sensori con interfaccia I2C che hanno principalmente uno scopo illustrativo delle potenzialità del sistema, ma che tuttavia misurano parametri verificabili. Il passo successivo alla scelta dei componenti è

stato il far interagire il controllore ed il modem, per poi aggiungere la lettura dei sensori, ed infine la trasmissione dei dati.

## 2.3 Hardware

In questa sezione verranno esaminati lo schema elettrico e i componenti impiegati.

## 2.4 schema elettrico

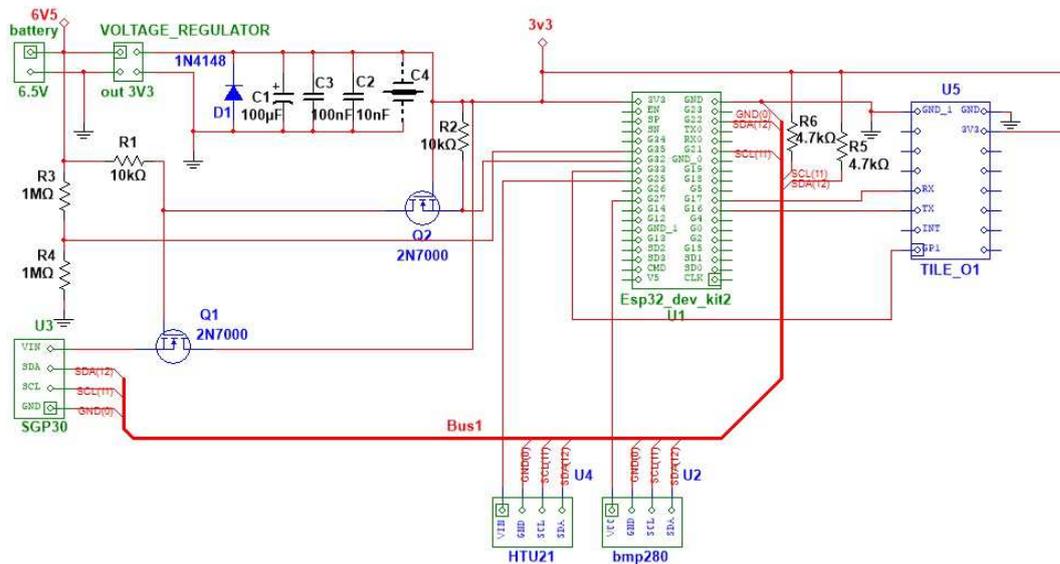


Figura 2.1: Schema elettrico del sistema

Come si può osservare dalla figura 2.1 lo schema elettrico non risulta particolarmente complesso, essendo infatti il sistema sviluppato principalmente dal lato software. Si può tuttavia notare un partitore di tensione posto sulla batteria, utilizzato per misurare la tensione di essa, e sapere quindi quando risulta scarica e scollegare il sistema per evitare che essa si danneggi o che il sistema non funzioni correttamente.

È stato inoltre inserito un piccolo circuito a due mosfet per controllare l'alimentazione del sensore SGP30, tuttavia le ragioni per cui è stato inserito tale circuito verranno trattate in seguito nella sezione 2.4.4. Sono stati inoltre inseriti dei condensatori per stabilizzare il più possibile la tensione, ed un diodo in polarizzazione inversa che serve a proteggere il circuito posto ai suoi capi da un impulso di tensione negativa che si verifica in seguito al collegamento della batteria. Non

essendo il controllore ESP-32 ed il modem TILE01 componenti standard, il loro simbolo è stato disegnato appositamente.

### 2.4.1 ESP-32

L'ESP-32 è un microcontrollore con Wi-Fi e Bluetooth integrati, successore del ESP-8266 è dotato di un Tensilica Xtensa LX6 come CPU nella versione dual core e di un Tensilica Xtensa LX7 nella versione single core. Supporta una frequenza fino a 240 MHz, è dotato di pipeline a sette stadi e può supportare fino a 70 fonti di interruzioni. La memoria interna comprende 448 kB di ROM per il booting e le funzioni principali, 520 kB di SRAM per i dati e le istruzioni e 8 kB di SRAM chiamata RTC FAST che permette il salvataggio e mantenimento di dati in fase di deep sleep. Gli aspetti che hanno favorito la scelta di questo controllore

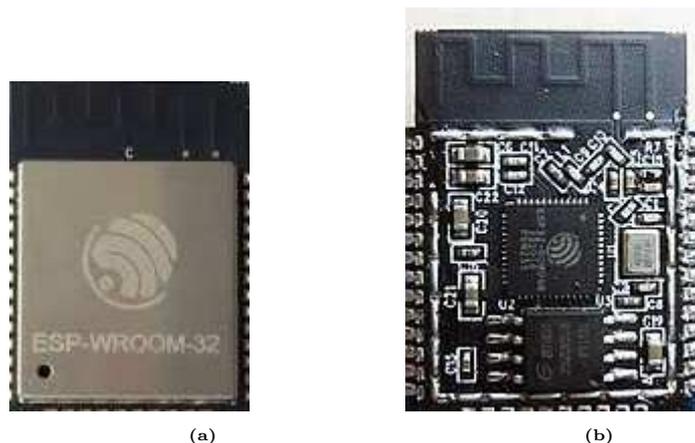


Figura 2.2: esempio di ESP WROOM 32, a sx con schermo di protezione, a dx senza

sono state principalmente:

- Logica a 3.3 V: In questo modo non vi è la necessità di adattamento dei livelli logici di tensione tra modem e controllore;
- Presenza di un modulo Wi-fi integrato: la presenza di tale modulo rende l'ESP-32 estremamente versatile e come si è rivelato in fase di sviluppo del sistema è stata essenziale in fase di simulazione, non avendo la possibilità di trasmettere i dati diversamente.
- 34 GPIO: un tale numero di pin rende possibili molte applicazioni digitali, o analogiche essendo presenti 3 interfacce UART, 2 interfacce I2C, un ADC a 12 bit con fino a 18 canali e 2 DAC a 8bit.

- Risparmio energetico: in modalità deep sleep sono dichiarati 5  $\mu\text{A}$  di assorbimento;
- Ambiente di sviluppo e community: essendo un microcontrollore piuttosto elaborato, è possibile programmarlo nell'ambiente di sviluppo Arduino ed è supportato da una community molto attiva in diversi forum, motivo per cui in caso si dovesse incorrere in qualche problema, probabilmente anche qualcun altro prima di noi ha dovuto affrontarlo e potrebbe aver già trovato una soluzione.

Tuttavia come si può notare dalla figura 2.2a il microcontrollore si presenta in formato SMD e senza alcuna interfaccia di programmazione "Plug and use", fattore molto scomodo in fase di sviluppo di un progetto. Per questa ragione diverse aziende hanno prodotto dei così detti development kit, che integrano al loro interno un convertitore Seriale-Uart e un convertitore di tensione a 3.3 V utili in fase di programmazione via usb. In figura 2.3 si può osservare il modello esatto di Devkit utilizzato nel progetto. Una volta comunque sviluppato il sistema il passaggio ad una versione in stand-alone è banale, in quanto dopo averlo programmato basta saldarlo nella versione finale del pcb senza la necessità di affiungervi quarzi o altri componenti.

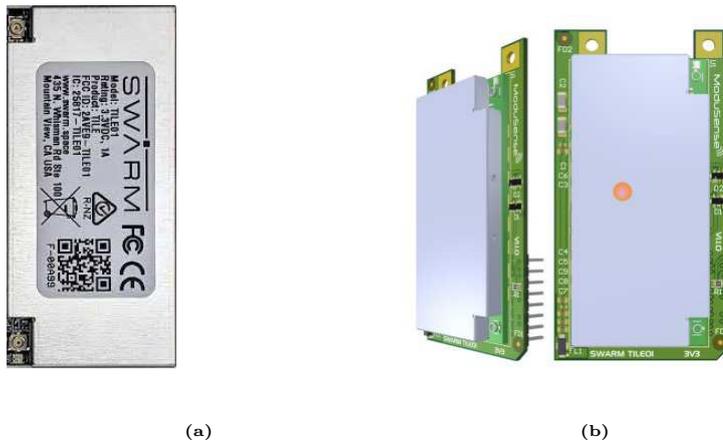


**Figura 2.3:** ESP32 NodeMCU Moduli WiFi Development Board con CP2102

## 2.4.2 TILE-01

Lo Swarm Tile è un modem satellitare per la trasmissione e ricezione di dati da e per la rete Swarm, designato ad essere inserito in prodotti di terze parti. Il modem si presenta in formato SMT come si può osservare in figura 2.4a e comunica attraverso una interfaccia seriale UART standard a 3.3 V. Esso è dotato inoltre di un led di stato il cui comportamento lo si può trovare in tabella 2.1 e di due connettori U.FL. uno per l'antenna GPS e l'altro per l'antenna VHF.

Essendo la vendita di tale modem disponibile sul sito ufficiale SWARM (<https://www.swarm.com>):



**Figura 2.4:** (a) Modem Tile01, (b) ModuSense SWARM Mikro Bus Module Board

Colore	Comportamento	Stato
verde	Fisso 3 secondi	Sequenza di bootup.
	Lampeggio acceso 100ms ogni 5 secondi	Indicatore di heartbeat, il modem è acceso e funziona correttamente.
rosso	fisso 10 secondi	Dopo esser stato alimentato finché non inizia le sequenza di bootup.
	flash veloce	Dopo il bootup, fintanto che non ha una posizione GPS valida.
	Singolo flash ogni 5 secondi	dopo esser stato svegliato dallo sleep, finché non ottiene una posizione GPS valida.
	Fisso	Durante la sequenza di spegnimento finché non viene tolta dalla scheda.

**Tabella 2.1:** Indicatore di stato LED

//swarm.space) a partire da un minimo di 25 pz. ci si è affidati a un rivenditore per l'acquisto il quale fornisce inoltre un'interfaccia di backend per la visualizzazione dei messaggi trasmessi oltre ad un Socket al quale è saldato il modem che lo adatta al formato MicroBUS (Figura 2.4b). Per la comunicazione il modulo impiega una versione del protocollo NMEA con comandi a due lettere ed un checksum a due cifre esadecimali alla fine preceduti da un \*.

Il modem dispone inoltre di un pin GPIO configurabile attraverso l'invio di un comando via seriale e di essere posto in sleep, limitando quindi il suo assorbimento di corrente come può essere visto dalla tabella 2.2. Si può anche, nel caso di necessità, spegnere via software la scheda, tuttavia in tal caso per riaccendere il modem bisogna togliere e ridare alimentazione. Il modem non è resistente all'acqua: nel caso in cui si dovesse quindi farlo operare all'aperto, vi è la necessità

Modalità	Tipico	Picco
Sleep	-	80 $\mu$ A
Ricezione attiva	26mA	40mA
Trasmissione attiva	850mA	1000mA

**Tabella 2.2:** corrente assorbita dal modem nelle varie funzioni, con alimentazione a 3.3 V

di predisporre un alloggiamento adeguato. Può tuttavia operare in ambienti con temperature comprese tra i -40 °C e +85 °C, con umidità comprese tra lo 0% ed il 95%.

È inoltre sensibile a disturbi elettromagnetici, per questa ragione è suggerito schermare l'alimentatore nel caso si utilizzi un alimentatore switching: è in ogni caso raccomandato l'inserimento di filtri ai pin di alimentazione.

### 2.4.3 Antenne

Come accennato in precedenza il modem presenta due connettori per le due antenne richieste al suo funzionamento.

Sono supportate le antenna GPS che lavorano a una tensione di 3.3 V e fino a 35 mA di corrente assorbita, in figura 2.5b si può osservare l'antenna utilizzata.

Tuttavia per l'antenna per la comunicazione satellitare è suggerito l'utilizzo della antenna fornita dall'azienda stessa per essere certi che sia sintonizzata correttamente. L'antenna fornita è un'antenna a  $\frac{\lambda}{4}$  con guadagno di 2.0 dBi e una potenza nominale in uscita di 30 dBm equivalenti ad 1 W. La frequenza alla quale lavora in ricezione è tra 137.000 MHz e 138.000 MHz mentre in trasmissione tra 148.000 MHz e 150.000 MHz. Per il suo corretto funzionamento è richiesta la presenza di un piano di massa, e che sia situato ad almeno 1 m di altezza rispetto il suolo. Funziona per temperature comprese tra i -55 °C e i +130 °C e una umidità tra lo 0% ed il 100%. Deve essere inoltre posta ad una distanza minima di 36 cm dal corpo umano per ragioni di sicurezza.

### 2.4.4 Sensori

#### SGP30

L' SGP-30 è un sensore di gas digitale multi-pixel a basso consumo con interfaccia I2C. Il sensore è composto da una piastra riscaldante e degli elementi sensibili che restituiscono un valore analogico corrispondente alla concentrazione di gas



**Figura 2.5:** (a) Antenna VHF SWARM, (b) Antenna GPS



**Figura 2.6:** Modulo SGP-30, BlueDot

nell'aria, valore che viene poi convertito in un dato digitale e, a richiesta, inviato attraverso l'interfaccia I2C.

É sensibile alla presenza di etanolo per concentrazioni tra 0.3 ppm e 30 ppm con una accuratezza del 15% e di idrogeno per concentrazioni tra 0.5 ppm e 10 ppm con una accuratezza del 10% con una risoluzione<sup>1</sup> dello 0.2%.

Tuttavia non è stato scelto per misurare la presenza di tali gas, ma per la sua possibilità di misurare due aspetti della qualità dell'aria: concentrazione di CO<sub>2</sub> e di TVOC<sup>2</sup> (Total Volatile Organic Compounds) Il TVOC è un raggruppamento di un'ampia gamma di composti chimici organici definito per semplificare la segnalazione quando questi sono presenti nell'aria ambiente o nelle emissioni. La CO<sub>2</sub> viene rilevata per concentrazioni da 400 ppm a 60000 ppm mentre i TVOC tra 0 ppb e 60000 ppb con una risoluzione visualizzabile in tabella 2.3 e può operare tra i -40 °C ed i 125 °C, con umidità tra il 10% ed il 95%.

Il sensore si presenta in formato SMD con una dimensione di 2.45 mm x 2.45mm x 0.9 mm, e richiede una alimentazione a 1.8 V con una corrente di 48 mA. Si sarebbe dunque dovuto realizzare un PCB specifico per utilizzarlo, per tali ragioni si è proceduto all'acquisto di un modulo che contenesse il sensore già saldato

<sup>1</sup>L'accuratezza indica quanto il valore misurato si discosta dal valore reale, mentre la risoluzione è la minima variazione che il sensore può misurare.

TVOC	0 ppb - 2008 ppb	1 ppb
	2008 ppb - 11110 ppb	6 ppb
	11110 ppb - 60000 ppb	32 ppb
CO <sub>2</sub>	400 ppm - 1479 ppm	1 ppm
	1479 ppm - 5144 ppm	3 ppm
	5144 ppm - 17597 ppm	9 ppm
	17597 ppm - 60000 ppm	31 ppm

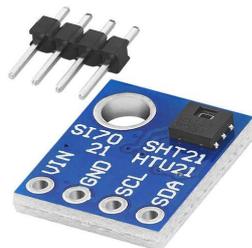
**Tabella 2.3:** Risoluzione qualità dell'aria SGP 30

su di esso, con un regolatore di tensione per l'alimentazione e due adattatori di livelli logici a mosfet per la comunicazione I2C.

In figura 2.6 si può osservare il modulo impiegato.

Il modulo assorbe quindi anch'esso 48 mA, per tale ragione, volendo ridurre il consumo di corrente, averlo lasciato permanentemente alimentato sarebbe stato un inutile spreco di energia. Tuttavia, il pin di alimentazione non poteva essere controllato tramite un pin GPIO del ESP-32, in quanto essi possono erogare una corrente massima di 40mA contro i 48 richiesti dal modulo. Si è quindi ricorsi all'utilizzo di un circuito a mosfet che sfruttasse il fatto che dopo ogni riavvio il sensore entri in una modalità di sleep il cui consumo è di appena 2  $\mu$ A. Per riavviare il sensore è necessario togliere l'alimentazione per un paio di secondi e poi ridarla. All'avvio del sensore, dopo l'invio del comando di inizializzazione, prima di ottenere una misura valida bisogna attendere che la fase di inizializzazione termini. Durante questa fase, che dura all'incirca 15 secondi, alla richiesta di fornire le misure effettuate verranno restituiti i valori fissi di 400 ppm per la CO<sub>2</sub> e 0 ppb per i TVOC. Dopo tale fase le misure possono essere considerate valide.

## HTU21



**Figura 2.7:** Sensore di umidità e temperatura GY-21 HTU21, Az-Delivery

Si tratta di un sensore digitale di umidità e temperatura a basso consumo energetico, anticipatamente calibrato, con un tempo di risposta breve. Deve essere alimentato a una tensione di 3 V ed in fase di misura assorbe 450  $\mu\text{A}$ .

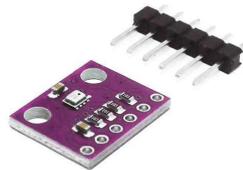
Anche in questo caso si è utilizzato un modulo pre-assemblato a causa delle piccole dimensioni del sensore e del suo formato SMD. In figura 2.7 si può osservare il modello utilizzato: esso è inoltre dotato di un regolatore di tensione ed un adattatore di livelli logici per la comunicazione seriale. Infatti tale modulo supporta tensioni fino a 5 V, tuttavia assorbendo una corrente inferiore a 40 mA l'alimentazione verrà fornita dall'ESP-32 attraverso un pin GPIO nel momento in cui sarà richiesto di effettuare una misura.

Il sensore può operare tra i  $-40\text{ }^\circ\text{C}$  ed i  $125\text{ }^\circ\text{C}$  con umidità tra lo 0% ed il 100%, ed ha una tolleranza del 3%.

Essendo la misura dell'umidità fortemente dipendente dalla temperatura la misura viene compensata internamente dal sensore attraverso la formula 2.1. É quindi fortemente consigliato di lasciare il sensore a contatto con l'aria in modo che possa acquisirne la temperatura nel miglior modo possibile.

$$RH_{compensato} = RH_{attuale} + (25 - T_{attuale}) \times CoefTemp \quad (2.1)$$

## BMP280



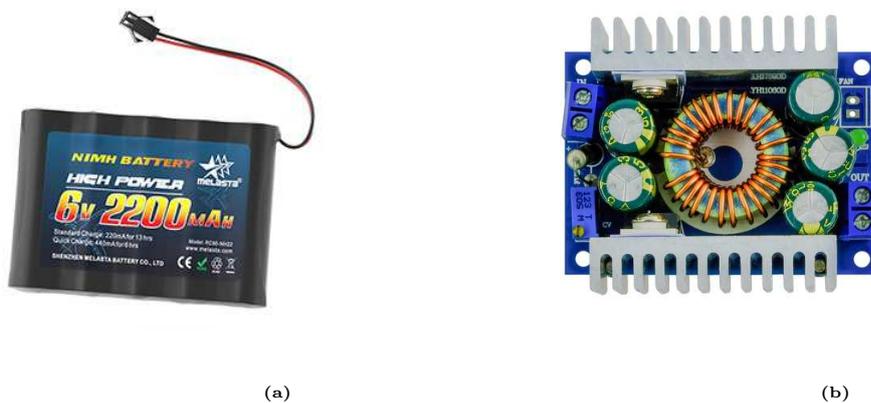
**Figura 2.8:** Sensore barometrico GY-BMP280

Il BMP - 280 è sensore di pressione dell'aria Piezo-resistivo prodotto dalla BOSCH pensato per applicazioni dove è richiesta una misura precisa della pressione. Le piccole dimensioni ed il basso consumo energetico rendono il BMP-280 ottimo per applicazioni mobili. Il sensore è operativo tra i  $-40\text{ }^\circ\text{C}$  ed i  $+85\text{ }^\circ\text{C}$ , e pressioni tra i 300 hPa ed i 1100 hPa. Nel datasheet non sono state trovate dipendenze dalla umidità, tuttavia per applicazioni di precisione il comportamento

del sensore andrebbe verificato al variare dell'umidità all'interno di una camera climatica.

L'accuratezza del sensore è dipendente dalla temperatura, in particolare tra  $-20\text{ }^{\circ}\text{C}$  e  $0\text{ }^{\circ}\text{C}$  è di  $\pm 1.7\text{ hPa}$ , mentre tra  $0\text{ }^{\circ}\text{C}$  e  $65\text{ }^{\circ}\text{C}$  aumenta a  $\pm 1.0\text{ hPa}$ . Il sensore è controllabile sia via SPI che via I2C, ed essendo dotato nativamente al suo interno di un regolatore di tensione, può essere alimentato tra  $1.8\text{ V}$  e  $3.6\text{ V}$ . Tuttavia sempre a causa delle sue piccole dimensioni è stato utilizzato attraverso il modulo in figura 2.8. In fase di misura il sensore assorbe  $720\text{ }\mu\text{A}$ , per tale ragione la sua alimentazione viene controllata dall'ESP-32.

### 2.4.5 Alimentazione



**Figura 2.9:** (a) Batteria ricaricabile 6V 2200 mAh Ni-MH, (b) AZDelivery, Regolatore di Tensione mod. XL4016E1

Essendo pensato per applicazioni in ambienti remoti è necessario che il sistema sia alimentato a batterie. In particolare le batterie utilizzate in questo sistema sono delle batterie Ni-MH (ossia nichel – metallo idruro), batterie che si prestano bene anche all'utilizzo a basse temperature fino a  $-20\text{ }^{\circ}\text{C}$ , con una lieve riduzione dell'autonomia. Sono state utilizzate due batterie da  $6\text{ V}$  e  $2200\text{ mAh}$  poste in parallelo per aumentarne la capacità. Il modello è visionabile in figura 2.9a. Hanno un tasso di scarica continua di  $10\text{C}$  ossia che possono essere scaricate a con una corrente di  $22\text{ A}$ . Per tale ragione la corrente erogata può soddisfare la richiesta del sistema (maggiore di  $1\text{ A}$  in fase di trasmissione). Tuttavia essendo il sistema funzionante a  $3.3\text{ V}$  è stato necessario inserire un regolatore di tensione, essendo inoltre un prototipo sviluppato su breadboard e poi su millefori, si è optato per utilizzare un convertitore buck pre-assemblato visibile in figura 2.9b. La tensione di uscita di tale regolatore può viene regolata tramite un trimmer, e

---

la corrente massima erogabile raggiunge gli 8 A. Tuttavia qualora si procedesse alla produzione di un circuito stampato è consigliato l'impiego di un LTC3113 prodotto dalla Analog Devices, il quale oltre ad essere reperibile ad un costo inferiore è adatto ad applicazioni in radio frequenza. Tuttavia in tal caso andrebbe cambiata batteria o almeno rimossa una cella in quanto la massima tensione in ingresso da esso supportata è di 5 V.



# Capitolo 3

## Protocolli di comunicazione seriale

A differenza della comunicazione parallela dove i bit vengono trasmessi simultaneamente, nella comunicazione seriale i bit che compongono un dato vengono trasmessi in sequenza. Per questa ragione il numero di linee richieste per il trasferimento dei dati si limita ad un minimo di due. Per tali ragioni nel mondo dei microcontrollori è largamente diffusa la presenza di periferiche di comunicazione e l'utilizzo di protocolli di comunicazione seriale.

Esistono due modalità di comunicazione seriale: asincrona e sincrona. Quest'ultima sebbene possa raggiungere velocità più elevate richiede una maggiore complessità fisica.

Pertanto, a meno di particolari esigenze, viene prediletta la comunicazione asincrona.

### 3.1 Comunicazione seriale asincrona

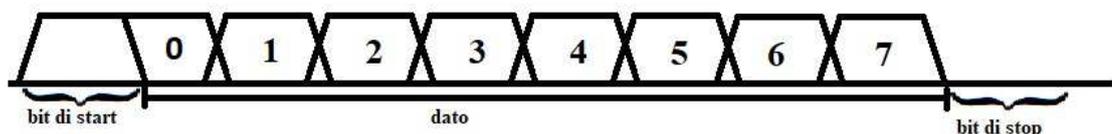


Figura 3.1: Diagramma temporale di una trasmissione asincrona

La comunicazione seriale asincrona avviene senza che i clock del ricevitore e del trasmettitore siano sincronizzati, in quanto il sincronismo viene ricavato direttamente dalla linea dati.

La convenzione che si adotta impone che il dato sia preceduto da un bit di start

e uno sia seguito da uno di stop come può esser visto in figura 3.1.

Vi è inoltre la possibilità di inserire prima del bit di stop un bit di parità, posto a 0 o 1 affinché il numero di 1 del dato sia un numero pari, tuttavia non viene molto utilizzato. Questa comunicazione viene utilizzata spesso in applicazioni in cui è richiesto un trasferimento di dati senza particolari esigenze di velocità.

## 3.2 I2C (Inter Integrated Circuit)

L'Inter Integrated Circuit è uno standard per bus di comunicazione largamente diffuso, è stato sviluppato dalla Philips nel 1982 e nel 1992 è stata prodotta la prima versione del protocollo. Il bus I2C è composto da un master, generalmente un microcontrollore, e uno o più slave connessi attraverso due linee: una di clock e una dati. Data la sua larga diffusione molti microcontrollori includono apposite interfacce per la gestione di questo tipo di comunicazione. Nello standard I2C sono disponibili diversi modi di funzionamento: il modo normale che opera fino a 100 kbps, il modo fast fino a 400 kbps e il modo ultra fast fino a 5 Mbps.

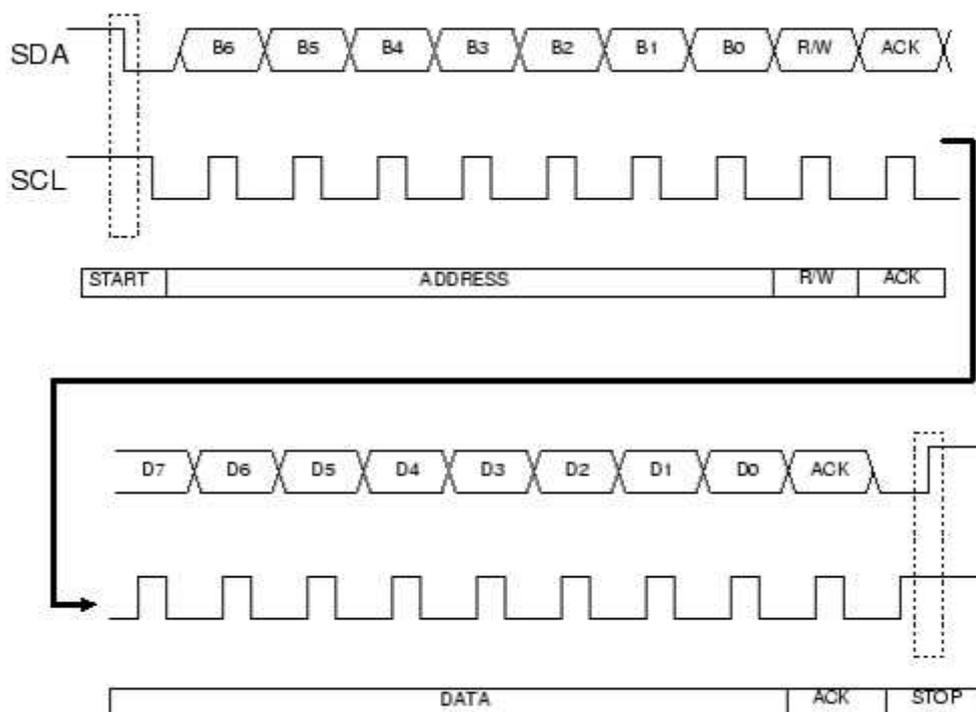


Figura 3.2: Diagramma temporale di una trasmissione I2C

Ogni messaggio è composto da pacchetti di 9 bit. Il messaggio è aperto da una condizione di start e chiuso da una condizione di stop generate dal microcontrollore. Le condizioni di start e stop possono essere riconosciute in quanto vengono effettuate nella fase alta del clock, mentre tutti gli altri bit vengono impostati nella fase bassa del clock. I primi 8 bit di ogni pacchetto vengono settati dall'unità che sta trasmettendo in quel momento, mentre il nono viene settato dal ricevitore, tale bit prende il nome di acknowledgement. La sua funzione è di permettere al trasmettitore di capire se il messaggio è stato ricevuto correttamente. Il primo dato trasmesso in seguito al segnale di start è l'indirizzo della periferica alla quale il microcontrollore vuole connettersi. L'indirizzo ha una lunghezza di 7 bit, l'ottavo serve a definire la direzione del trasferimento dei dati, e quindi se in lettura o scrittura. Nel momento in cui la periferica viene chiamata quest'ultima abbassa la linea dati nel nono bit time, facendo così capire al microcontrollore di essere pronta. Si procederà quindi con le istruzioni necessarie.



# Capitolo 4

## Software

La componente software, e quindi il funzionamento vero e proprio del sistema si è basato sul risparmio energetico e che per questo tipo di monitoraggio ambientale i dati richiesti fossero due temperature e una lettura di pressione, umidità, CO<sub>2</sub> e TVOC.

Inoltre i tempi di sviluppo richiesti per una applicazione sviluppata in C per un microcontrollore sono piuttosto lunghi, per tale ragione il sistema è stato sviluppato in ambiente Arduino. Inoltre nell'ultima sezione si analizzerà il codice utilizzato nella simulazione.

### 4.1 Ambiente di sviluppo

L'Arduino IDE (Integrated Development Environment), creato da Massimo Banzì, David Cuartielles, Tom Igoe, Gianluca Martino, e David Mellis, è un ambiente di sviluppo open source derivato dall'IDE per il linguaggio di programmazione Processing e per il progetto Wiring. Il linguaggio di programmazione utilizzato in tale IDE è un linguaggio derivato dal C/C++ che originariamente era pensato per avvicinare le persone al mondo della programmazione. Originariamente era stato sviluppato per le schede Arduino, tuttavia ora attraverso l'installazione di specifici pacchetti è possibile anche l'utilizzo di altre schede come l'ESP-32, l'STM32, ATtiny e altre ancora.

### 4.2 Librerie

Essendo ormai l'utilizzo di Arduino largamente diffuso, trovare librerie specifiche per ogni sensore spesso si rivela una strategia ulteriore per ridurre i tempi di svi-

luppo. Inoltre, per una larga parte di sensori sono gli stessi produttori dei moduli a rendere disponibili librerie specifiche, ad esempio SparkFun o Adafruit. Tuttavia sono spesso anche i membri della community di Arduino a rendere disponibili delle librerie: un esempio ne è la libreria utilizzata per il controllo del modem, che sebbene abbia necessitato di una analisi ed alcune modifiche ha comunque velocizzato il processo di sviluppo del sistema.

### 4.3 La Macchina a stati finiti

L'approccio a macchina a stati è un modello matematico che permette di descrivere con precisione il comportamento di un sistema. Per poter utilizzare questo modello il sistema deve essere dinamico, deterministico e discreto. In figura 4.1 il diagramma del sistema trattato in questo studio.

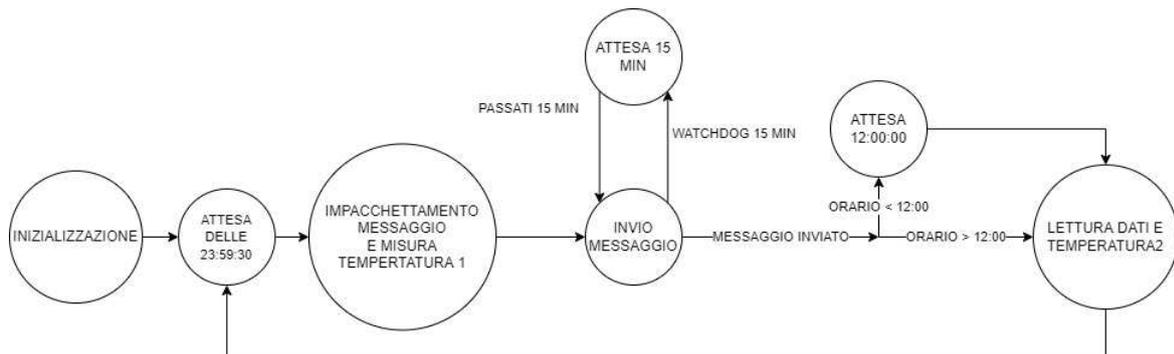


Figura 4.1: Diagramma degli stati del sistema

### 4.4 La simulazione ed il codice

A causa dei problemi riportati in precedenza, per la simulazione del sistema, l'invio dei dati è stato effettuato attraverso la compilazione di un form google appositamente creato per la ricezione dei messaggi.

Le variabili come lo stato e le misure effettuate sono state inizializzate come RTC DATA ATTR in modo essere salvate presso la parte di memoria RTC e non essere perse durante il deep-sleep. La parte di codice contenuta nel void setup viene eseguita ad ogni accensione anche dopo il deep sleep ed in tale fase viene anche controllata la tensione ai capi della batteria, così che nel caso fosse inferiore

ad un livello prestabilito si proceda allo spegnimento del sistema fintanto che la batteria non venga sostituita.

Verranno ora analizzati singolarmente i vari stati del sistema.

#### **4.4.1 Batteria scarica**

Qualora si verificasse tale condizione il microcontrollore dopo aver stabilito una connessione con il modem invierà il comando di power off.

A seguito di tale comando è possibile riaccendere il modem solo dopo aver tolto e riassegnato completamente tensione al sistema.

Invece non esistendo un comando per spegnere il microcontrollore, lo si mette in deep sleep senza però assegnare alcun metodo di risveglio. In tal modo esso rimarrà in deep sleep fino al successivo reset.

#### **4.4.2 Inizializzazione**

Subito dopo la prima accensione, e dopo ogni cambio di batteria, il sistema procede inizializzando il modem ad uno stato conosciuto. In questo stato il microcontrollore verifica che il modem sia connesso e risponda, attraverso l'invio della richiesta della versione del firmware.

Procede quindi alla misurazione dei parametri ambientali con tutti i sensori, in questo caso non conoscendo l'orario alla quale il dispositivo sarà acceso, nell'arco della giornata si effettuerà un'unica misurazione della temperatura.

Elimina quindi gli eventuali messaggi in coda non ancora spediti per poi attendere di aver ricevuto la data e l'ora attraverso il segnale GPS.

Imposta quindi il GPIO del modem nella funzione 13, che da datasheet indica la modalità di output a livello logico basso in fase di sleep, e ordina al modem di andare in sleep fino alle 23.59.30 del giorno stesso. A questo punto si aggiorna la variabile indicante lo stato, si configura quindi la ESP-32 in modo tale che venga svegliata dalla fase di sleep attraverso un cambio di stato nel pin GPIO alla quale è connessa con il modem e la si manda in deep-sleep.

#### **4.4.3 Impacchettamento messaggio**

Il sistema si sveglierà ora alle 23:59:30, si attende quindi che il modem sia pronto a trasmettere i dati, ossia che quindi abbia ricevuto la data, l'ora e la posizione dal GPS e si esegue una attesa di 30 secondi per essere sicuri che passi la mezzanotte. Si procede quindi a creare un messaggio in formato stringa contenente tutti i dati

in un ordine specifico, separati da un ";", si mette quindi in coda il messaggio per l'invio e si effettua la misurazione della prima temperatura.

Si aggiorna quindi lo stato e si passa allo stato successivo.

#### 4.4.4 Attesa invio messaggio

In questo stato si attende che il messaggio venga spedito, quindi come prima cosa si verifica che il modem sia pronto a trasmettere, si inizializza quindi un timer che fungerà da watchdog per controllare il passaggio del tempo, e quindi se dopo quindici minuti il messaggio sia ancora in coda. In tal caso si procederà ad uno sleep di altri quindici minuti. Infatti controllando il passaggio dei satelliti attraverso la piattaforma SWARM SATELLITE PASS CHECKER, reperibile presso il sito ufficiale SWARM (<https://swarm.space/>), si possono notare intervalli privi di passaggi anche piuttosto lunghi, quindi nel caso sfortunato ci si trovasse in tale finestra, mantenere il sistema acceso per tutto il tempo sarebbe uno spreco di energia. Quindi mantenendo acceso il sistema per metà del tempo si può ovviare a questo problema.

Non appena invece il sistema trasmetterà il dato si procede all'aggiornamento della variabile di stato e ad un controllo dell'ora. Questo perché qualora fossero passate le 12:00, ossia l'orario stabilito per la seconda misurazione dei valori ambientali, si procederà subito alla misurazione di questi e alla fase di sleep fino le 23:59:30. Nel caso invece in cui così non fosse si procederà ad una nuova fase di sleep come visto precedentemente questa volta però fino alle 12:00.

É comunque fortemente improbabile che passino 12 ore senza che il sistema riesca ad inviare il messaggio, vi è comunque la possibilità di aggiungere un tempo oltre il quale il messaggio si auto-elimina nel caso in cui non riesca ad essere inviato.

Si noti che questo è lo stato in cui si effettua la distinzione tra la simulazione e la versione del software effettiva, ossia per effettuare l'invio del messaggio via Wi-Fi si attende di ricevere il segnale di beacon del satellite, e quindi solo una volta ricevuto questo si procede all'invio dei dati.

Il comando impiegato è stato aggiunto alla libreria in quanto non presente, esso considera il satellite agganciato solo nel momento in cui viene ricevuto un ID del satellite valido.

#### **4.4.5 Misurazione valori e sleep**

In quest'ultimo stato, dopo aver ricevuto la data e l'ora si procede alla misurazione dei valori ambientali, si aggiorna lo stato a "impacchettamento messaggio" e si avvia nuovamente il deep-sleep sino le 23:59:30, in questo modo si chiude il ciclo del sistema.



## Conclusioni

Questo sistema, sebbene piuttosto semplice, è un ottimo esempio delle potenzialità di questa tecnologia, che in un anno ha visto aumentare il numero di satelliti della propria rete, aumentando la frequenza delle finestre disponibili per la trasmissione.

Al momento un recente studio che vede l'applicazione della comunicazione attraverso i satelliti SWARM per il rilevamento di incendi boschivi ha dato un esito molto positivo in quanto utilizzando sensori particolarmente sensibili si è riusciti a rilevare principi d'incendio entro 60 minuti dall'innescio. L'unico vincolo temporale al momento rimane legato alle finestre di trasmissione, ma per una applicazione tale, dove le alternative sono meno efficienti e non vi sono possibilità alternative per trasmettere i dati, questo rimane un ottimo sistema.

Si spera quindi che con il continuo sviluppo di questa tecnologia e l'incremento dei satelliti dispiegati, si possa arrivare ad un livello in cui è possibile il monitoraggio real time con tempi di latenza ancora più brevi.



# Bibliografia

- [1] Kevin Ashton, That 'internet of things' thing, RFID journal <http://www.itrco.jp/libraries/RFIDjournal-That%20Internet%20of%20Things%20Thing.pdf>, 22 Giugno, 2009.
- [2] Alessandro Bassi, Martin Bauer, Martin Fiedler, Thorsten Kramp, Rob van Kranenburg, Sebastian Lange, Stefan Meissner, *Enabling Things to Talk*, capitolo 1, Introduction to the Internet of Things.
- [3] Carlo Brandolese, William Fornaciari, Mondo Digitale n.3, Settembre 2009.
- [4] Embedded System Market, marketsandmarkets, <https://www.marketsandmarkets.com/Market-Reports/embedded-system-market-98154672.html>
- [5] Soluzioni IoT per il monitoraggio ambientale: quali sono e a cosa servono, Josephine Condemi, 14 Giugno 2021
- [6] environmental monitoring <https://www.eea.europa.eu/help/glossary/gemet-environmental-thesaurus/environmental-monitoring>
- [7] ESP32 Series Datasheet, [https://espressif.com/sites/default/files/documentation/esp32\\_datasheet\\_en.pdf](https://espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf)
- [8] Swarm Tile Product Manual.
- [9] Datasheet SGP30, Sensirion Gas Platform.
- [10] Datasheet HTU21D(F) Sensor, MEAS.
- [11] Datasheet BMP280 Digital Pressure Senso, BOSCH.

- [12] Simone Buso, Introduzione alle applicazioni industriali di microcontrollori e dsp.
- [13] Carsten Brinkschulte, Dryad CEO, Preventing wildfires through ultra-early detection.