BAB II

TINJAUAN PUSTAKA

2.1 LoRa

Gambar 2.1 Implementasi Teknologi LoRa[14]

LoRa merupakan teknologi modulasi RF untuk jaringan area luas (LPWAN) yang berdaya rendah. LoRa mengacu pada transmisi data jarak jauh yang dimungkinkan oleh teknologi ini. LoRa menyediakan komunikasi jarak jauh hingga 5 km di daerah perkotaan, dan 15 km atau lebih di daerah pedesaan dengan propagasi *Line of Sight* (LOS). Karakteristik utama LoRa adalah kebutuhan daya yang sangat rendah, yang memungkinkan perangkat yang dioperasikan dapat menggunakan baterai yang dapat bertahan hingga 10 tahun [15].

Jenis modulasi LoRa adalah Chirp Spread Spectrum (CSS). Chirp merupakan kepanjangan dari *Compressed High Intensity Radar Pulse* yang mana teknik ini umum digunakan pada sonar dan radar. Sinyal chirp memiliki amplitudo konstan dengan variable frekuensi. Jika frekuensi berubah dari rendah ke tinggi maka disebut *up-chirp* sedangkan jika sebaliknya maka disebut *down-chirp* [16]. Berikut ini merupakan tabel karakteristik dari modulasi CSS :

Tabel 2.1 Karakteristik Modulasi LoRa [15]

Data Rate (DR)	Spreading Factor (SF)	Channel Frequency	Uplink or Downlink	Bitrate (Bits/Sec)	Maximum User Payload Size (Bytes)
0	SF10	125 kHz	Uplink	980	11
1	SF9	125 kHz	Uplink	1,760	53
2	SF8	125 kHz	Uplink	3,125	125
3	SF7	125 kHz	Uplink	5,470	242
4	SF8	500 kHz	Uplink	12,500	242
5 – 7					
8	SF12	500 kHz	Downlink	980	53
9	SF11	500 kHz	Downlink	1,760	129
10	SF10	500 kHz	Downlink	3,125	242
11	SF9	500 kHz	Downlink	5,470	242
12	SF8	500 kHz	Downlink	12,500	242
13	SF8	500 kHz	Downlink	21,900	242

Terdapat beberapa parameter yang mempengaruhi modulasi pada LoRa yaitu sebagai berikut :

1. *Spreading Factor* (SF)

Data yang akan ditransmisikan melalui komunikasi LoRa direpresentasikan sebagai sinyal CHIRP yang memiliki batasan frekuensi awal dan akhir. Pada LoRa nilai frekuensi awal dari *Compressed High Intensity of Radar Pulse* (CHIRP) direpresentasikan sebagai sebuah simbol. Jumlah bit-bit data yang dikodekan di dalam sebuah simbol adalah parameter yang bisa diubah yang disebut dengan *Spreading Factor* (SF). Maka dari itu, sebuah CHIRP menggunakan *Spreading Factor* (SF) yang direpresentasikan dengan 2^{SF} bit per simbol, yang berarti satu simbol terdiri dari banyak CHIRP yang mana merupakan pulsa dalam kode *spread spectrum,* seperti urutan kode *Pseudo-Noise* (PN) [16]:

$$SF = \frac{chip\ rate}{symbole\ rate} \quad \text{.....(2.1)}$$

Modulasi CSS (*Chirps Spread Spectrum*) menyebabkan energi sinyal akan tersebar pada band yang lebih luas. Spreading factor pada transmisi juga digunakan untuk menentukan periode symbol (T_s) dengan persamaan sebagai berikut :

$T_s = \frac{2^{SF}}{B}$……(2.2)

Dimana B merupakan bandwidth sinyal. Dengan asumsi bahwa modulasi menggunakan bandwidth tetap, peningkatan SF akan meningkatkan durasi symbol T_s. Peningkatan transmisi waktu untuk chirp (*Symbol Time*) memberikan ketahanan pesan yang lebih besar terhadap interferensi atau *noise*. Di sisi lain, efek ini sebagian dapat dikompensasi oleh fakta bahwa jika SF lebih tinggi, maka jumlah simbol meningkat, yang membuat frekuensi kesalahan simbol menjadi lebih besar, di mana sinkronisasi antara penerima dan sinyal menjadi sangat penting. Kerugian lain dari transmisi pesan yang lebih lama adalah kemungkinan tabrakan (*collision*) yang lebih tinggi. Faktanya, SF mempengaruhi sensitivitas (S) penerima yang didefinisikan sebagai berikut :

$S = -174 + 10\log_{10}(B) + NF + SNR$……..……(2.3)

Dimana (-174) disebabkan oleh *noise* termal pada penerima dalam bandwidth 1 Hz, NF adalah Angka Kebisingan (*Noise Figure*) pada penerima yang ditetapkan untuk data konfigurasi perangkat keras dan SNR adalah rasio sinyal terhadap *noise* yang diperlukan untuk modulasi [16]. Laju bit data (R_b) dapat dilihat pada persamaan berikut:

$Rb = SF \times \frac{B}{2^{SF}}$……...(2.4)

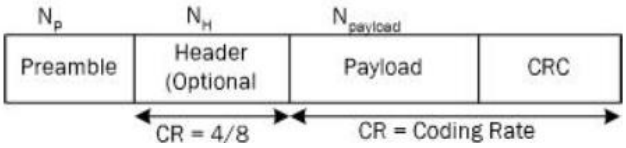
2. *Coding Rate* (CR)

Implementasi modulasi LoRa menggunakan pendeteksi bit error berupa Hamming Code untuk *Forward Error Correction* (FEC), implementasi ini dilakukan dengan melakukan enkode 4 bit data dengan redundansi menjadi 5 bit, 6 bit, 7 bit, ataupun 8 bit. Penggunaan redundansi ini membuat sinyal LoRa lebih tahan terhadap interferensi singkat, nilai *Coding Rate* (CR) perlu diatur sesuai dengan kondisi kanal yang dipakai, jika terdapat banyak interferensi sebaiknya nilai CR ditingkatkan. Lora menawarkan beberapa pilihan CR yaitu 4/5, 4/6, 4/7, dan 4/8. Namun perlu diperhatikan bahwa kenaikan nilai CR juga meningkatkan waktu transmisi. Lora menggunakan sebuah variable pengoreksi error yang akan

menambah performanya dalam mentransmisikan sinyal [16]. Data rate pada persamaan (2.4) dikembangkan menjadi :

$$R_b = SF \times \frac{B}{2^{SF}} \times \frac{4}{4+CR} \quad \text{.....(2.5)}$$

CR adalah *coding rate*, dimana CR $\in\{1,2,3,4\}$. CR berbanding lurus dengan kecepatan transmisi data (*data rate*) dan berbanding terbalik dengan TOA (*Time-on-Air*)[17].



Gambar 2.2 Struktur Paket LoRa [18]

ToA didefinisikan sebagai waktu yang dibutuhkan paket untuk mencapai sisi receiver. ToA dari suatu paket dapat dihitung dengan kombinasi antara CR, SF dan BW. Durasi paket LoRa didefinisikan sebagai kombinasi dari paket yang dikirim dan *preamble* (awalan) yang memiliki persamaan sebagai berikut [18]:

$$T_{preamble} = (n_{preamble} + 4.25)\, T_s \quad \text{.....(2.6)}$$

$n_{preamble}$ mengindikasikan panjang preamble yang diprogram. *Payload* didasarkan pada header yang diaktifkan. Jumlah symbol *payload* dapat dihitung melalui persamaan berikut ini :

$$n_{payload} = 8 + \max\left(\text{ciel}\left[\frac{(8PL-4SF+28+16RC-20IH)}{4(SF-2DE)}\right](\text{CR} + 4), 0\right) \quad \text{.....(2.7)}$$

Dimana :

PL = Jumlah bytes *payload*

SF = Spreading Factor (dari 7 sampai 12)

IH = 0 (Header tidak aktif) atau IH = 1

CR = Coding Rate

Periode *payload* adalah banyaknya symbol *payload* dikali dengan symbol periode sebagaimana persamaan di bawah ini :

$$T_{payload} = n_{payload} * T_s \quad \text{.....(2.8)}$$

ToA adalah penjumlahan dari durasi preamble dan *payload* yaitu sebagai berikut:

$T_{packet} = T_{preamble} + T_{payload}$ ……(2.9)

2.2 Regulasi Frekuensi LPWAN di Indonesia

Berdasarkan Peraturan Jenderal Sumber Daya dan Perangkat Pos Informatika Nomor 3 Tahun 2019 tentang Persyaratan Teknis Alat dan/atau Perangkat Telekomunikasi *Low Power Wide Area* (LPWA), persyaratan perangkat LPWA yang wajib dipenuhi yaitu dapat dijabarkan sebagai berikut :

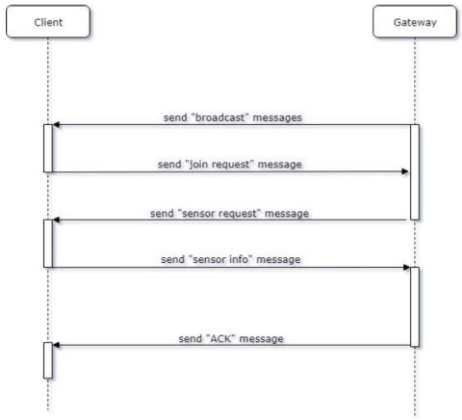
1. Pita frekuensi radio : 920 – 923 MHz
2. Lebar Pita (Bandwidth)
   Wideband : ≤ 250 kHz
   Narrowband : ≤ 200 kHz
3. Duty Cycle
   a. Alat dan/atau Perangkat Telekomunikasi LPWA Nonseluler Wideband
      - Downlink : ≤ 1%
      - Uplink : ≤ 1%
   b. Alat dan/atau Perangkat Telekomunikasi LPWA Nonseluler Narrowband
      - Downlink : ≤ 10%
      - Uplink : ≤ 1%

2.3 Multi-node Sensor

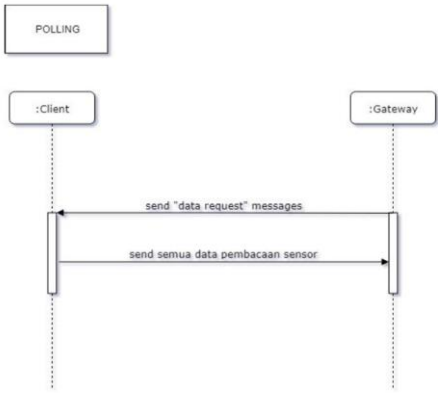
Multi-node sensor (*multiple node sensor*) didefinisikan sebagai banyaknya *node* yang terhubung melalui komunikasi nirkabel. Dalam setiap *node* atau *client* terdapat satu atau lebih sensor-sensor yang bekerja sesuai kebutuhan. Biasanya, semua sensor dapat dikonfigurasi sebagai jaringan seperti jaringan area lokal [19].

2.4 *Simple Lora Protocol* (SLP)

Simple Lora Protocol (SLP) merupakan protokol pengembangan dari LoRaWAN yaitu pada penelitian [13]. Protokol ini bekerja pada *data-link layer* suatu jaringan. *Simple Lora Protocol* mengontrol urutan pengiriman pesan antara klien dan *gateway*. Proses komunikasi dibagi menjadi dua kondisi, yaitu *set* dan *polling*. *Setup mode* adalah proses identifikasi *client* yang masuk ke dalam jaringan *gateway* sedangkan *polling mode* adalah proses *gateway* menerima data dari *client*. Dalam kedua mode, *gateway* dan klien bertukar pesan untuk komunikasi. Pada setup mode inisialisasi dimulai saat *gateway* mengirimkan pesan *broadcast* ke *client* yang kemudian diterima oleh *client*. Sedangkan mode *polling* dimulai setelah mode setup selesai. Detail alur pengiriman pesan pada mode *setup* digambarkan pada diagram sekuen berikut ini.

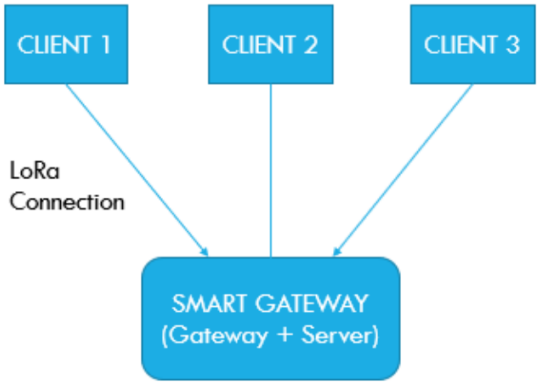


Gambar 2.3 Diagram Sekuen Mode Setup [13]



Gambar 2.4 Diagram Sekuen Mode Polling [13]

Arsitektur jaringan SLP terdiri dari *Gateway* cerdas dan beberapa *client* dengan sensor terpasang. Tiap *client* dapat mengoperasikan maksimal 3 sensor. Tiap-tiap *client* terhubung ke *Gateway* melalui koneksi LoRa.



Gambar 2.5 Arsitektur Jaringan *Simple Lora Protocol* [13]

2.5 Hypertext Transfer Protocol (HTTP)

HTTP merupakan protokol komunikasi jaringan yang awalnya dikembangkan oleh Tim Berners-Lee. Seperti CoAP, HTTP menggunakan *Universal Resource Identifier* (URI). Server mengirimkan data menggunakan URI dan klien menerima data menggunakan URI tertentu. HTTP menggunakan TCP sebagai protokol transport default dan TLS/SSL untuk keamanan. Dengan demikian, komunikasi antara klien dan server berorientasi koneksi[20].

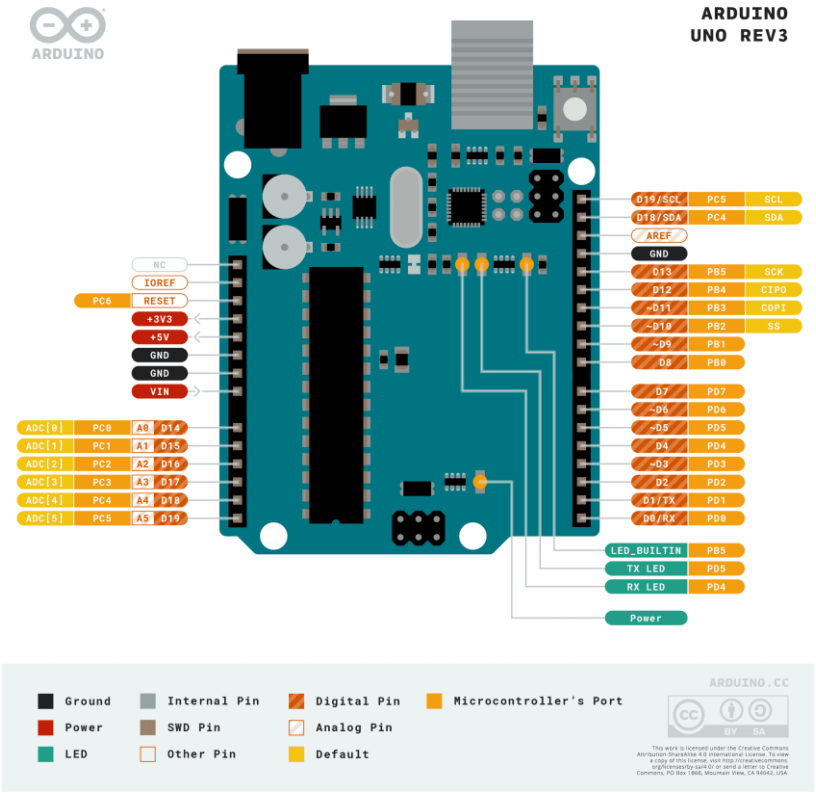
2.6 Prototyping

Prototyping merupakan suatu teknik pengembangan pada suatu sistem yang menggunakan *Prototype* untuk menggambarkan sistem, sehingga pengguna atau pemilik sistem mempunyai gambaran pengembangan sistem yang akan dilakukannya. Teknik ini sering digunakan apabila pemilik sistem tidak terlalu menguasai sistem yang akan dikembangkannya sehingga dia memerlukan gambaran dari sistem yang akan dikembangkannya tersebut [21].

2.7 *Hardware* dalam Rancangan *Prototype*

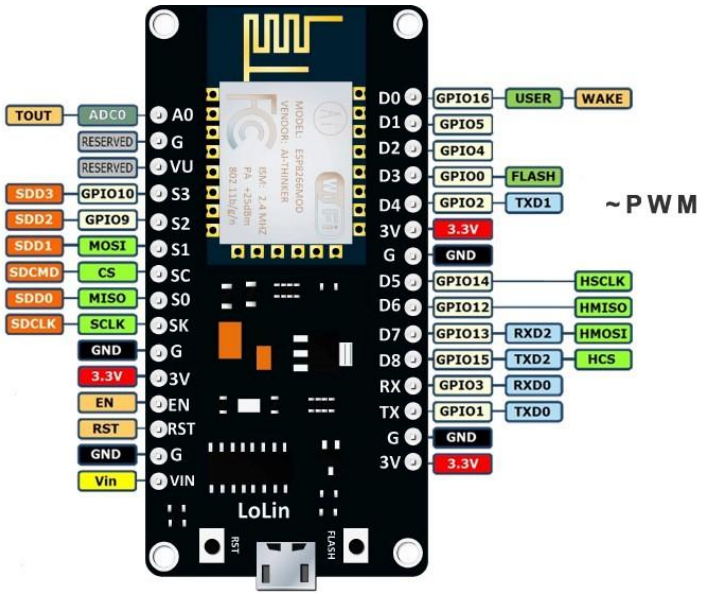
2.7.1 Arduino Uno

Arduino Uno adalah papan yang menggunakan mikrokontroler ATMega328. Arduino Uno memiliki 14 pin digital (dapat digunakan 6 pin Output PWM), 6 input analog, osilator kristal 16MHz, antarmuka USB, konektor daya, *header* dan tombol ICSP. Arduino Uno disuplai melalui USB atau masukan daya DC dari baterai atau adaptor AC-DC Arduino Uno menggunakan ATmega16U2 yang diprogram sebagai konverter USB ke serial untuk komunikasi serial dengan computer melalui port USB. Board Arduino UNO dapat beroperasi pada sebuah suplai eksternal 6 sampai 20 Volt[22]. "Uno" berarti satu dalam bahasa Italia dan dinamai sesuai dengan perilisannya Arduino 1.0 yang mana versi 1.0 adalah model referensi versi lanjutan dari Arduino. Arduino Uno R3 adalah versi terbaru dan model referensi dari seri *board* Arduino[23].



Gambar 2.6 Arduino Uno Pin Out[23]

2.7.2 NodeMCU ESP8266



Gambar 2.7 NodeMCU ESP8266 Pin Out[24]

NodeMCU adalah sebuah firmware berbasis LUA untuk ESP8266 WiFi SOC. ESP8266 adalah modul WiFi yang berfungsi sebagai mikrokontroler tambahan seperti Arduino untuk terhubung langsung ke WiFi dan membentuk koneksi TCP/IP. NodeMCU bekerja dengan cara yang sama seperti Arduino, meskipun IC yang digunakan, GPIO dan bahasa pemrograman yang digunakan berbeda, tetapi tujuannya sama yaitu untuk mengontrol sistem,

NodeMCU bisa dianalogikaan sebagai *board* arduino yang terkoneksi dengan ESP8622. NodeMCU telah me-package ESP8266 ke dalam sebuah *board* yang sudah terintegrasi dengan berbagai fitur selayaknya mikrokontroler dan akses terhadap WiFi dan juga chip komunikasi yang berupa USB to serial[25].

2.7.3 LoRa Chip RFM95

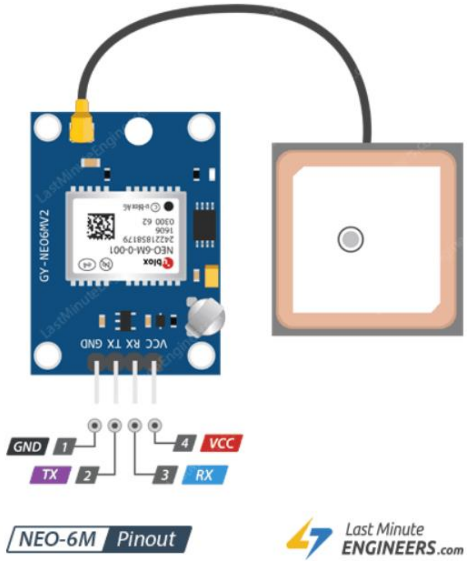
Modul LoRa dengan *chip* RFM95/96/97/98(W) merupakan modem LoRa jarak jauh yang menyediakan komunikasi spektrum tersebar dan ketahanan tinggi dengan konsumsi daya minimal. Menggunakan teknologi

modulasi yang dipatenkan oleh Hope RF, RFM95/96/97/98(W) dapat mencapai sensitivitas lebih dari -148dBm serta menggunakan kristal dan material yang murah. Sensitivitas tinggi dipadukan dengan *power amplifier* +20dBm terintegrasi menghasilkan *link budget* terdepan di industri, menjadikannya ideal untuk aplikasi apa pun yang membutuhkan jangkauan atau daya tahan baterai yang awet. LoRaTM juga menawarkan keunggulan pemblokiran dan selektivitas yang signifikan dibandingkan teknik modulasi tradisional, desain tradisional antara jangkauan, kekebalan pemblokiran, dan konsumsi daya rendah. Perangkat ini juga mendukung mode FSK (G) performa tinggi dalam sistem seperti WMBus, IEEE802.15.4g. RFM95/96/97/98(W) menawarkan *noise* fase, selektivitas, linearitas *receiver*, dan IIP3 dengan konsumsi daya yang jauh lebih rendah daripada perangkat pesaing[26].

- 168 dB *link budget* maksimum.
- +20 dBm - 100 mW output RF konstan vs. tegangan.
- +14 dBm efisiensi tinggi PA.
- Bit rate 300 kbps.
- Sensitivitas tinggi hingga -148 dBm.
- 127 dB *dynamic range* RSSI.
- Ukuran modul :16*16mm

Gambar 2.8 Lora Chip RFM95 [27]

2.7.4 Modul GPS



Gambar 2.9 Pin Out GPS Ublox Neo 6m [28]

GPS biasa digunakan untuk menentukan sebuah posisi, dimana posisi yang didapat GPS mengambil data dari satelit. Data yang dicakup biasanya adalah waktu, *latitude*, *longitude*, ketinggian, dan kecepatan. Dengan beberapa data tersebut, kita dapat menggunakannya sebagai *tracking device*. Pada penelitian ini penulis menggunakan Modul Ublox Neo 6m yang cukup murah namun sangat baik dalam menerima data dari satelit. Modul GPS ini adalah keluarga dari *stand-alone GPS receivers*. Ukuran modul ini cukup kecil hanya 16x12.2x2.4 mm saja. Dapat diaplikasikan ke laptop dengan komunikasi menggunakan USB dengan protokol UART Terdapat LED pada modul GPS NEO-6M yang mengindikasikan status 'Pemantapan Posisi'. LED akan berkedip dengan kecepatan yang berbeda, tergantung pada statusnya sebagai berikut:

1) Tidak berkedip : sedang mencari satelit.
2) Berkedip setiap 1 detik : Penentuan Posisi ditemukan (modul dapat melihat cukup banyak satelit)[28]. Untuk gambar dari GPS tersebut dapat dilihat pada Gambar 2.9.

2.7.5 Sensor Kelembaban

Gambar 2.10 Modul DHT22
Sumber : [Dokumentasi Penulis]

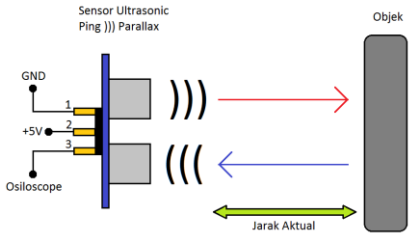
Sensor kelembaban merupakan sensor yang dapat membaca tingkat kelembaban ruangan. Pada penelitian ini, penulis menggunakan sensor DHT22 yang mana di dalam sensor ini terdapat sebuah thermistor dengan tipe NTC (*Negative Temperature Coefficient*). Cara kerja dari thermistor adalah dengan naik dan turunnya suhu berpengaruh terhadap naik dan turunnya resistansi thermistor. Berdasarkan naik turunnya resistansi tersebut maka sensor akan mengeluarkan output berupa nilai analog yang akan dibaca dan dikonversi oleh arduino menjadi nilai suhu (dalam bentuk ºC) dan kelembaban ruangan (dalam bentuk %).

Sensor ini memiliki 4 kaki pin, dan terdapat juga sensor DHT22 dengan *breakout* PCB yang terdapat hanya memiliki 3 kaki[29].

- Tegangan masukan : 5 Vdc
- Rentang temperatur :0-50 ° C kesalahan ± 2 ° C
- Kelembaban :20-90% RH ± 5% RH error

2.7.6 Sensor Ultrasonik

Sensor ultrasonik adalah sensor yang memiliki frekuensi 40KHz. Sensor ini bekerja berdasarkan prinsip pantulan gelombang suara dimana sensor ini akan memantulkan gelombang suara kemudian akan menangkapnya kembali. Perbedaan waktu ketika dipancarkan kemudian kembali inilah yang menjadi dasar penginderaannya. Gelombang ultrasonik merambat melalui udara dengan kecepatan 344 m/s, mengenai obyek dan memantul kembali ke sensor ultrasonik.



Gambar 2.11 Cara Kerja Sensor Ultrasonik[30]

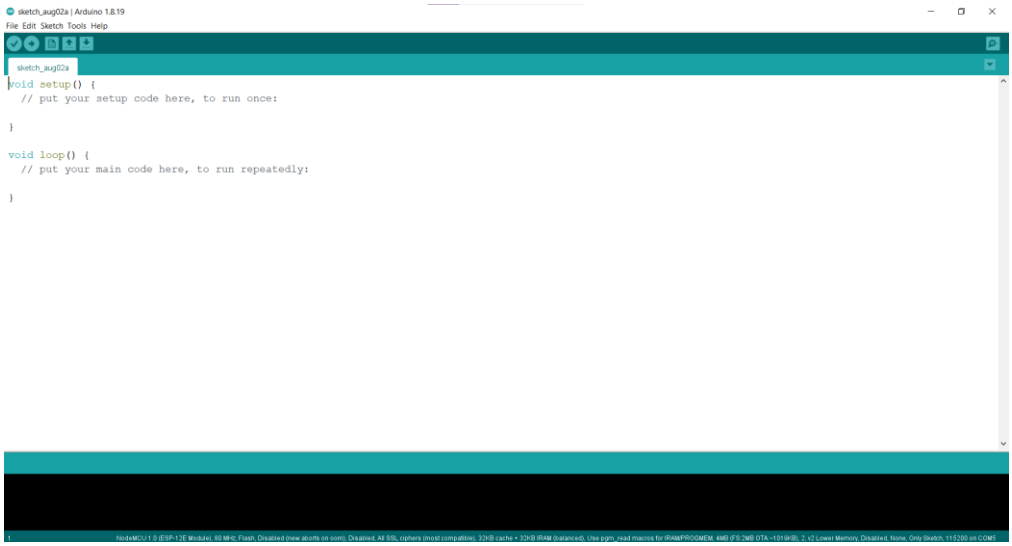
Gambar 2.12 Pin Out Sensor Ultrasonik HC-SR04[31]

Sifat dari gelombang ultrasonik yang melalui medium menyebabkan getaran partikel dengan medium amplitudo sama dengan arah rambat longitudinal sehingga menghasilkan partikel medium yang membentuk suatu rapatan atau biasa disebut *strain* dan tegangan yang biasa disebut *strees*. Gelombang ultrasonik merambat melalui udara dengan kecepatan 344 meter per detik, mengenai obyek dan memantul kembali ke sensor ultrasonik[32].

2.8 *Software* dalam Rancangan *Prototype*

2.8.1 *Software* Arduino IDE

Arduino IDE (*Integrated Development Environment*) merupakan *software* bawaan Arduino yang memiliki fungsi untuk menulis program, menyimpan dan mengunggah ke memori mikrokontroler. Arduino menyebut program sebagai sketsa (*sketch*)[33]. IDE memiliki sub menu yang ditunjukkan pada tombol-tombol di bagian atas seperti yang ditunjukkan pada gambar di bawah ini.



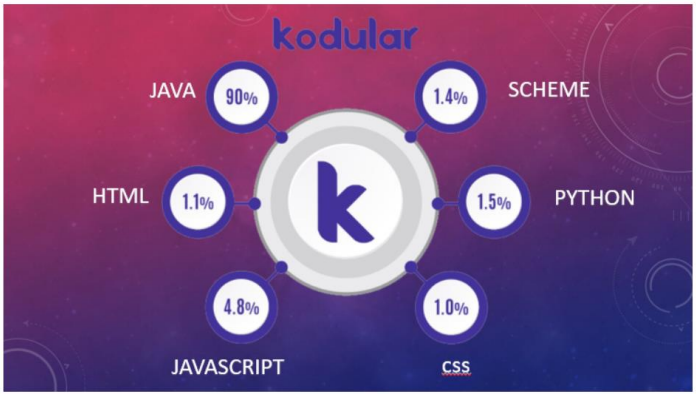
Gambar 2.13 Tampilan Arduino IDE Versi 1.8.19

Sumber : [Dokumentasi Penulis]

Bagian-bagian menu dari Arduino IDE :

1. *Compile*: Sebelum "kode" program Anda dapat dikirim ke papan, kode tersebut perlu diubah menjadi instruksi yang dimengerti oleh papan. Proses ini disebut kompilasi.
2. *Upload*: mengunggah program (*sketch*) ke board
3. *New*: membuat *sketch* baru, ini membuka jendela baru untuk membuat sketch baru.
4. *Open*: membuka sketsa yang telah ada, ini memuat sketsa dari file di computer.

2.8.2 Kodular



Gambar 2.14 Kodular[34]

Kodular adalah *website* yang menyediakan *tools* yang menyerupai MIT App Inventor untuk membuat aplikasi Android dengan menggunakan *block programming*. Dengan kata lain, tidak perlu mengetik kode program secara manual untuk membuat aplikasi Android. Saat ini, Kodular dan AppyBuilder telah bersatu dalam kontribusi untuk menciptakan sebuah aplikasi bagi orang-orang yang masih awam atau tidak ada kemampuan *coding* bisa membuat aplikasi Android sendiri dengan fitur dan layanan hampir mirip dengan Android Studio secara simpel dan mudah. Kodular Creator merupakan versi yang lebih lanjut dari MIT APP Inventor dengan penambahan fitur sensor[35].

2.8.3 Firebase

Gambar 2.15 Logo Firebase[36]

Firebase Realtime Database adalah platform database untuk aplikasi *real-time*. Saat data berubah, aplikasi yang terhubung ke Firebase secara otomatis diperbarui di semua perangkat, baik itu web maupun seluler. Firebase memiliki pustaka yang luas untuk sebagian besar platform web dan seluler. Firebase dapat diintegrasikan dengan *framework* lain seperti Node, Java, Javascript, dan lainnya. Firebase menawarkan beberapa fitur sebagai berikut[37]:

1. Analytics, fitur ini digunakan untuk memantau perilaku pengguna saat menggunakan aplikasi dan ditampilkan di *dashboard*.
2. Develop, fungsi ini mencakup komunikasi cloud, autentikasi, database *real-time*, penyimpanan, hosting, lab pengujian, dan pelaporan kerusakan.
3. Grow, fungsi ini digunakan untuk mem-*publish* aplikasi.

2.9 Parameter *Quality of Service* (QoS)

Beberapa parameter yang menjadi indikator dari QoS yang menunjukkan performansi dari sistem komunikasi yaitu sebagai berikut:

2.9.1 *Received Signal Strength Indicator* (RSSI)

Received Signal Strength Indicator (RSSI) merupakan parameter yang menunjukkan daya terima dari seluruh sinyal pada band frekuensi saluran yang digunakan [38].

Rumus untuk menentukan RSSI:

$$RSSI(dBm)=TX\ Power(dBm) + TX\ Gain(dBi) - Free\ Space\ Path\ Loss + Rx\ Gain\ (dBi) \quad (2.10)$$

Rumus untuk mencari *Free Space Path Loss* yaitu sebagai berikut :

$$FSPL(dB) = 20log10(d) + 20log10(f) + K \quad (2.11)$$

Nilai RSSI semakin dekat dengan 0, maka semakin baik sinyal tersebut. Nilai RSSI dinyatakan dalam dBm dan merupakan nilai negatif, nilai minimum RSSI adalah –110 dBm.

2.9.2 *Packet loss*

Packet loss merupakan banyaknya paket yang gagal mencapai tempat tujuan saat pengiriman paket. Jika paket gagal dikirim maka paket tersebut tidak akan dikirim kembali, atau dengan kata lain paket tersebut hilang [38].

Rumus untuk menentukan *Packet loss* yaitu sebagai berikut:

$$\textit{Packet loss} = \frac{\text{Paket yang dikirim} - \text{paket yang diterima}}{Paket\ yang\ dikirim} \text{ x } 100\% \quad \text{.....(2.12)}$$

2.9.3 *Delay*

Delay merupakan banyaknya waktu yang diperlukan sebuah paket untuk melakukan perjalanan dari sumber ke tujuan. Bersama dengan *delay*, mendefinisikan kecepatan dan kapasitas dalam jaringan [38].

Rumus untuk menentukan *delay* yaitu sebagai berikut:

Delay = Waktu paket diterima− waktu paket dikirim……(2.13)

2.9.4 *Throughput*

Throughput merupakan kecepatan (rate) transfer data efektif, yang diukur dalam bps[13]. *Throughput* merupakan jumlah data persatuan waktu yang dikirim dari suatu titik jaringan ke titik jaringan yang lain. *Throughput* diukur dengan membandingkan data yang diterima dalam satuan bit dengan waktu pengamatan. 1 bytes adalah sama dengan 8 bit[39]. Berikut adalah persamaan untuk mendapatkan nilai *throughput* :

$$\textit{Throughput} = \frac{\text{Dt}}{w} \quad \text{.....(2.14)}$$

T : *Throughput*

Dt : Data terkirim

W : Waktu *Throughput*

2.10 Penelitian Terdahulu

Tabel 2.2 Studi Pustaka Sejenis

No.	Judul Penelitian	Peneliti	Tahun Penelitian	Kelebihan	Kekurangan
1.	Smart Waste Management using IOT	S. Neema dan K. Gor	2022	Dapat mendeteksi level, kelembaban dan berat sampah secara *real-time.* Pemberitahuan sampah penuh melalui sms.	Sistem manajemen sampah hanya berfokus pada satu tempat sampah sehingga pengimplementasiannya sempit.
2.	IoT Based Waste Management System Using NodeMCU	K. V. S. Reddy, P. M. Kumar, T. K. Kumar, T. Srinivasulu, and M. Seenu	2022	Dapat membedakan sampah basah dan kering serta mengirimkan lokasi bak sampah penuh melalui notifikasi sms yang disertai *link* lokasi.	Dibutuhkan biaya yang besar, baterai tidak tahan lama karena terlalu banyak sensor yang diaplikasikan, dan tidak ada pengujian performansi sistem.

3.	IoT Based Waste Management	M. Chavan, V. Swapna, H. Sune, and P. M. Yewale.	2022	Secara otomatis mendeteksi dan memisahkan sampah basah dan kering. Saat kotak sampah penuh, diberitahu melalui suara *buzzer* dan LCD.	Diimplementasikan untuk satu kotak sampah, boros daya baterai karena terlalu banyak sensor.
4.	*Prototype* sistem monitoring volume bak sampah online berbasis arduino nano Dan Orange Pi	A. Mappatoba, I. Haruna, and A. Agussalim	2020	Menggunakan 4 sensor ultrasonik pada satu bak sampah, pemantauan dilakukan di web.	Hanya bisa diaplikasikan di satu bak sampah serta tidak dilakukan pengujian performansi sistem.
5.	IoT based smart city garbage bin for waste management.	Mrs. E.Shanthini, Ms.Sangeetha. V, Dr. M. Jagadeeswari	2020	Dapat membedakan banyak jenis sampah, serta menyediakan fitur pemecah sampah menjadi bagian kecil.	Hanya berfokus pada satu tempat sampah.

6.	Digitalisasi Sistem Monitoring Sampah Rumahan Berbasis Internet of Things	M.S. Nur Pasha, T. Supriyadi dan R. Hanifatunnisa	2019	Melakukan pemisahan sampah logam dan non logam secara otomatis, adanya pengukuran berat sampah, serta monitoring sampah melalui aplikasi.	Hanya berfokus pada satu tempat sampah dan tidak bisa diterapkan di skala yang lebih besar.
7.	An IoT-based smart cities infrastructure architecture applied to a waste management skenario	P. Marques, D. Manfroi, E. Deitos, J. Cegoni, R. Castilhos, J. Rochol, E. Pignaton, and R. Kunst	2019	Menambahkan RFID pada bak sampah untuk autorisasi	Penggunaan daya yang besar karena banyak menggunakan sensor.

Tabel 2.3 Perbandingan Penelitian

No	Judul Penelitian	Tahun Penelitian	Jenis Modul				Modul LoRa	Platform Monitoring			
			Arduino	Raspberry Pi	Orange Pi	Node MCU		Blynk	SMS	Website	App
1.	Perancangan Sistem Manajemen Sampah dengan Pengaplikasian Multi-Node Sensor Menggunakan *Simple Lora Protocol* (SLP)	2023	−	-	-	☑	☑	-	-	-	☑
2.	Smart Waste Management using IOT	2022	−	☑	-	-	-	-	☑	-	-

3.	IoT Based Waste Management System Using NodeMCU	2022	☑	-	-	☑	-	-	-	☑	-
4.	IoT Based Waste Management	2022	☑	-	-	☑	–	☑	-	-	-
5.	*Prototype* sistem monitoring volume bak sampah online berbasis arduino nano Dan Orange Pi	2020	☑	-	☑	-	-	-	-	☑	-
6.	IoT based smart city garbage bin for waste management.	2020	-	-	-	☑	-	-	-	-	☑

7.	Digitalisasi Sistem Monitoring Sampah Rumahan Berbasis Internet of Things	2019	☑	-	-	☑	-	-	-	-	☑
8.	An IoT-based smart cities infrastructure architecture applied to a waste management skenario	2019	☑	-	-	-	☑	-	-	-	-