BAB II
DASAR TEORI DAN TINJAUAN PUSTAKA

2.1 Tinjauan Pustaka

Beberapa penelitian yang dijadikan referensi untuk pembuatan Proyek Akhir ini mencakup penelitian dari berbagai peneliti. Penelitian pertama adalah karya Yuli Febryanti, Fahrudin Mukti Wibowo, dan Anggi Zafia berjudul "Sistem Monitoring Tempat Sampah Pintar Di Pusat Penelitian Konservasi Tumbuhan Dan Kebun Raya-Lipi". Penelitian ini menghasilkan alat monitoring tempat sampah menggunakan NodeMCU di Pusat Penelitian Konservasi Tumbuhan dan Kebun Raya - LIPI. Alat tersebut menggunakan sensor ultrasonik dan *Load Cell* untuk mendeteksi kapasitas dan berat sampah. Data dari sensor kemudian dikirim ke *Firebase* dan dimonitor melalui aplikasi *Android*. Pengujian menunjukkan rata-rata kesalahan sensor jarak tempat sampah pertama sebesar 8,95% dengan akurasi 91,05% dan tempat sampah kedua dengan kesalahan 10,66% serta akurasi 89,34%. Untuk *Load Cell*, kesalahan pada tempat sampah pertama adalah 3,98% dengan akurasi 96,02%, dan tempat sampah kedua memiliki kesalahan 4,56% dengan akurasi 95,44% [3].

Penelitian kedua dari Dani Rohpandi, Shinta Siti Sundari, Ade Taopik Hidayatuloh, dan Saipul Muiz berjudul "Sistem Monitoring Tempat Pembuangan Sampah Sementara Berbasis *Android* dan *Internet of Things*". Penelitian ini merancang alat dengan sensor ultrasonik untuk mengukur ketinggian sampah dan mikrokontroller *NodeMCU ESP8266* sebagai pusat pengolahan data yang terhubung dengan *platform ThingSpeak*. Aplikasi *Android* dikembangkan untuk memberikan informasi status bak sampah secara *real-time*. Sistem ini juga mampu mendeteksi nyala api dan gas monoksida untuk kebutuhan biogas jika diperlukan [4].

Penelitian ketiga dilakukan oleh Aryanto Widigdo, Erna Triawati Christina, dan Desy Kristyawati, dengan judul "Rancang Bangun Monitoring Tempat Sampah Otomatis Berbasis *Internet of Things (IoT) Raspberry 3B+* Menggunakan *Telegram Bot* dan Notifikasi *Gmail*". Penelitian ini menciptakan tempat sampah otomatis yang dapat dimonitor melalui Telegram dan menerima notifikasi melalui *Gmail* ketika penuh. Tempat sampah menggunakan sensor ultrasonik HC-SR04 dan sensor *IR-Photodioda* untuk mendeteksi objek, serta dilengkapi motor servo, LCD, dan modul suara ISD1820 untuk berbagai fungsi. Notifikasi kondisi tempat sampah dikirim melalui *Gmail* dan *Telegram* menggunakan *Raspberry Pi 3B+* [5].

Penelitian keempat dari Ihqrom Pahruzi, Akhlis Munazilin, dan Achmad Baijuri dengan judul "Sistem Monitoring Tempat Penampungan Sementara (TPS) Berbasis Internet of Things (IoT)". Sistem ini menggunakan ESP32 dan sensor ultrasonik HC-SR04 untuk mengukur volume sampah, dan memanfaatkan *web server ThingSpeak* untuk pengamatan jarak jauh. Hasil penelitian menunjukkan bahwa sensor dapat mendeteksi level sampah dengan akurasi tinggi dan semua data dikirim dengan tepat ke *web server ThingSpeak* [6].

Dari tinjauan pustaka tersebut, dapat disimpulkan bahwa semua penelitian memiliki kesamaan dalam penggunaan teknologi IoT untuk melakukan monitoring sampah. Perbedaan utama terletak pada sensor yang digunakan dan metode pengiriman data. Penelitian pertama menggunakan sensor ultrasonik dan Load Cell dengan Firebase sebagai platform data, sementara penelitian kedua menggunakan sensor ultrasonik dan platform ThingSpeak. Penelitian ketiga menggunakan kombinasi sensor ultrasonik dan IR-Photodioda dengan notifikasi melalui Telegram dan Gmail, dan penelitian keempat menggunakan ESP32 dan sensor ultrasonik dengan platform ThingSpeak.

2.2 Dasar Teori

2.1.1 Tempat Penampungan Sementara

Menurut Peraturan Pemerintah Republik Indonesia Nomor 81 Tahun 2012 tentang Pengelolaan Sampah Rumah Tangga dan Sampah Sejenis Sampah Rumah Tangga menjelaskan bahwa Tempat Penampungan Sementara (TPS) adalah tempat

sebelum sampah diangkut ke tempat pendauran ulang, pengolahan, dan atau tempat pengolahan sampah terpadu.

TPS berfungsi sebagai tempat pertama untuk mengumpulkan sampah dari masyarakat sebelum diangkut ke tempat pemrosesan akhir. Dengan adanya TPS, sampah tidak dibuang sembarangan di tempat umum, sehingga dapat mengurangi pencemaran lingkungan, selain itu juga dapat mempermudah petugas kebersihan dalam mengangkut sampah ke TPAS atau TPST karena sampah sudah terpusat di satu lokasi. Di beberapa TPS, dilakukan pemilahan sampah antara sampah organik dan non-organik. Sampah organik dapat diolah menjadi kompos, sedangkan sampah non-organik dapat didaur ulang. Terdapat beberapa jenis TPS, diantaranya adalah sebagai berikut:

- TPS Konvensional yang merupakan TPS paling umum di Indonesia, TPS ini biasanya berupa wadah terbuka yang terbuat dari beton atau plastik.
- TPS 3R (*Reduce, Reuse, Recycle*) merupakan TPS yang dirancang untuk mendukung prinsip 3R itu sendiri, yaitu *reduce* (mengurangi), *reuse* (menggunakan kembali), dan *recycle* (mendaur ulang). TPS 3R biasanya dilengkapi dengan tempat pemilahan sampah dan edukasi tentang 3R.
- TPS Bank Sampah merupakan TPS yang dikelola oleh masyarakat untuk menampung sampah yang bernilai ekonomis, seperti botol plastik, kertas, dan logam. Sampah ini kemudian dijual ke pengepul atau diolah menjadi produk baru.

Pada proyek ini, TPS yang digunakan berbentuk bak kontainer (*Roll-Off Container*) dengan volume kurang lebih 6 liter yang diperoleh dari perhitungan dimensi kontainer dengan panjang 300 cm, lebar 170 cm, dan tinggi 120 cm. Volume kontainer ini dihitung menggunakan rumus volume balok, dimana dari perhitungan tersebut diperoleh volume sebesar 6,12 m^3. Meskipun terdapat sedikit perbedaan dari volume yang diharapkan, perbedaan ini bisa disebabkan oleh pembulatan atau toleransi dalam pembuatan kontainer.

Prototype yang digunakan dalam proyek ini memiliki bentuk gabungan dari prisma dan balok. Namun, dalam pengukuran volume sampah, hanya difokuskan

pada bagian balok. Fokus ini dilakukan untuk menyederhanakan penghitungan dan memastikan bahwa data yang diperoleh dari sensor tetap akurat.

2.1.2 *NodeMCU ESP32*

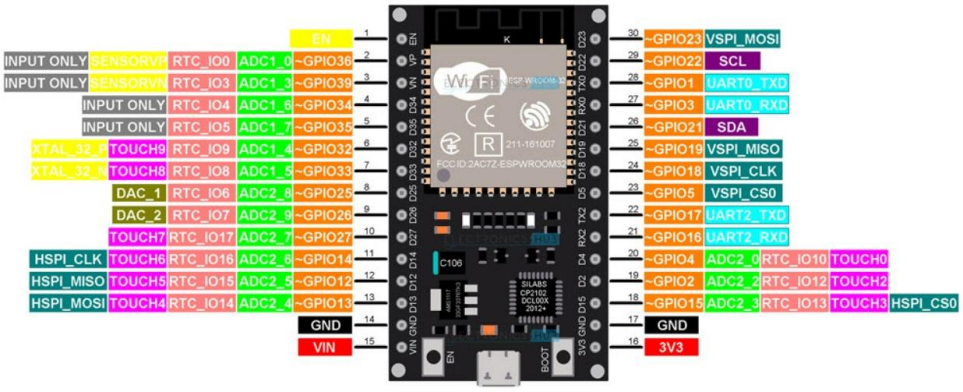
ESP32 dikenalkan oleh *Espressif System* yang merupakan penerus dari mikrokontroler ESP8266 [7]. Mikrokontroler ESP32 memiliki keunggulan yaitu sistem yang berbiaya rendah, dan juga berdaya rendah dengan modul *WiFi* yang terintegrasi dengan *chip* mikrokontroler serta memiliki *bluetooth* dengan mode ganda dan fitur hemat daya menjadikannya lebih fleksibel. ESP32 kompatibel dengan perangkat seluler dan aplikasi IoT (*Internet of Things*). Mikrokontroler ini dapat digunakan sebagai sistem mandiri yang lengkap atau dapat dioperasikan sebagai perangkat pendukung mikrokontroler *host*. ESP32 menggunakan prosesor *dual core* yang berjalan di instruksi *Xtensa* LX16 dan memiliki spesifikasi seperti yang ditampilkan pada tabel 2.1.

Tabel 2.1 Spesifikasi ESP32

No	Atribut	Detail
1	Tegangan	3.3 Volt
2	*Processor*	*Tensilica* L108 32 bit
3	Kecepatan prosesor	*Dual* 160MHz
4	RAM	520K
5	GPIO	34
6	ADC	7
7	Dukungan 802.11	11b/g/n/e/i
8	*Bluetooth*	BLE (*Bluetooth Low Energy*)
9	SPI	3
10	I2C	2
11	UART	3

Board ini memiliki dua versi, yaitu yang 30 dan 36 GPIO. Keduanya berfungsi dengan cara yang sama tetapi versi yang 30 GPIO dipilih karena memiliki dua pin GND. Semua pin diberi label di bagian atas board sehingga mudah untuk dikenali. *Board* ini memiliki *interface* USB-to-UART sehingga mudah diprogram

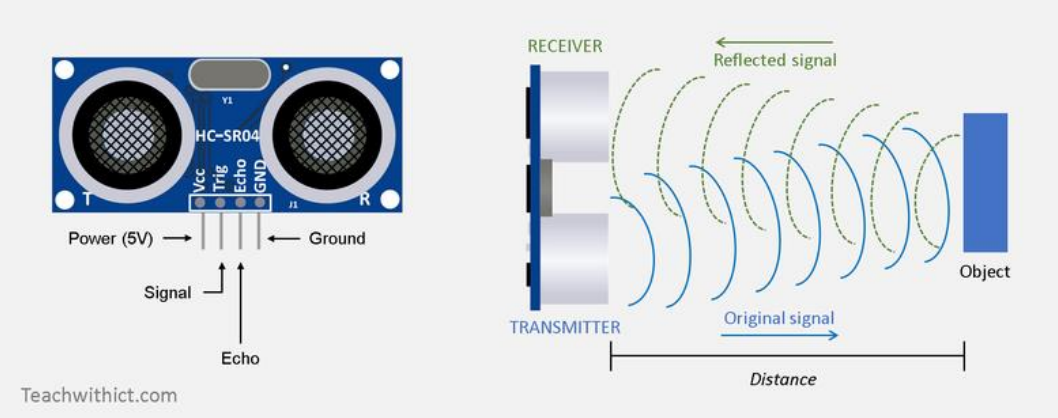
dengan program pengembangan aplikasi seperti Arduino IDE atau yang lainnya. Sumber daya untuk *board* bisa diberikan melalui konektor *micro*-USB. Keunggulan dari ESP32 adalah memiliki banyak pin yang dapat berfungsi sebagai *analog* atau *digital* sesuai dengan konfigurasi yang diinginkan, untuk detail dari pin ESP32 dapat dilihat pada gambar 2.1.



Gambar 2.1 Pin ESP32

2.1.3 Sensor HC-SR04

Sensor ultrasonik HC-SR04 memiliki tingkat akurasi hingga 3 mm dan mampu mengukur jarak secara efektif antara 2 hingga 400 cm. Sensor ini terdiri dari dua unit utama: pemancar dan penerima. Ketika ada objek yang menghalangi, gelombang ultrasonik yang dipancarkan akan memantul dan diterima oleh unit penerima, sebagaimana dijelaskan pada Gambar 2.2.



Gambar 2.2 Sensor HC-SR04

Pada Gambar 2.2 dijelaskan prinsip kerja sensor Ultrasonik bahwa pemancar mengirimkan gelombang, sementara penerima menangkap pantulan gelombang tersebut. Jarak suatu objek diukur berdasarkan perbedaan waktu antara pengiriman gelombang dan penerimaan pantulannya. Mikrokontroler menghitung waktu yang dibutuhkan gelombang untuk bergerak ke objek dan kembali, dan dari informasi ini, jarak objek dihitung menggunakan metode pengukuran jarak berbasis waktu penerbangan (*Time of Flight*). Modul ultrasonik HC-SR04 dapat mengukur jarak antara 2 cm hingga 400 cm dengan akurasi 3 mm. Rumus yang digunakan dalam menentukan jarak dari sensor terhadap objek yaitu:

$$\text{Jarak} = \text{kecepatan suara} \times T / 2$$

Dimana kecepatan suara sebesar 343 m/s, variabel T yang merupakan waktu tempuh saat sinyal ultrasonik dipantulkan dari transmitter dan kembali diterima oleh *receiver*.

Modul HC-SR04 terdiri dari beberapa komponen utama: *Chip* penghasil sinyal 40 kHz yang memancarkan gelombang ultrasonik, speaker ultrasonik yang mengubah sinyal ini menjadi gelombang yang dikirimkan ke objek, dan mikrofon ultrasonik yang mendeteksi pantulan gelombang tersebut. Mikrofon mengirimkan sinyal kembali ke modul untuk diolah, dan dari waktu perjalanan gelombang, jarak antara sensor dan objek dihitung dengan akurasi tinggi. Prinsip ini dikenal sebagai pengukuran jarak berdasarkan waktu penerbangan (*Time of Flight*).

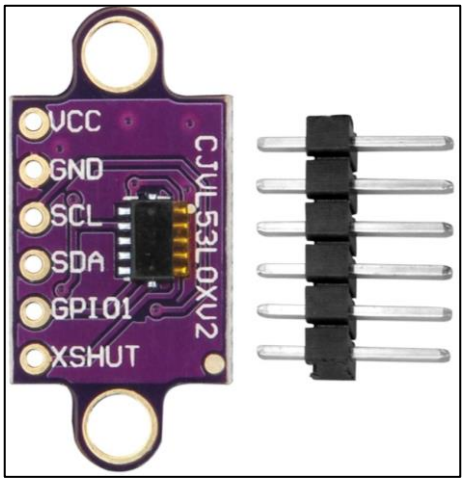
Sensor HC-SR04 memiliki empat pin, seperti yang ditunjukkan pada Gambar 2.2. Pin VCC terhubung ke sumber daya 5V DC, Pin Trig terhubung ke pin digital Arduino untuk mengirimkan sinyal gelombang, dan Pin Echo terhubung ke pin digital Arduino untuk menerima pantulan gelombang ultrasonik dari objek. Ketika gelombang belum terpantul kembali, logika Pin Echo berada pada status “HIGH”, dan berubah menjadi “LOW” setelah gelombang diterima. Pin GND dihubungkan ke ground pada papan Arduino untuk melengkapi rangkaian.

Metodologi penelitian ini menggunakan gelombang suara dengan frekuensi 40 kHz. Pengaturan logika “HIGH-LOW” pada pin Trig memungkinkan sensor ultrasonik menghasilkan dan memancarkan gelombang ultrasonik. Pin Echo pada

sensor ini bertugas menerima pantulan gelombang dari objek yang berada di depannya. Sensor tetap dalam status “HIGH” hingga pantulan gelombang diterima kembali, kemudian berubah menjadi “LOW”. Pin GND terhubung ke *ground* pada papan Arduino untuk memastikan koneksi ground yang diperlukan dalam rangkaian. [8]

2.1.4 Sensor VL53L0X

Selain menggunakan sensor HC-SR04 dalam mendeteksi jarak sensor terhadap objek dapat digunakan sensor VL53L0X. Sensor VL53L0X merupakan sensor LIDAR (*Light Detection and Ranging*) dimana sensor ini merupakan sensor jarak jauh yang memanfaatkan cahaya sinar laser untuk mendapatkan informasi tentang jarak suatu objek. Penggunaan sensor LIDAR VL53L0X tidak hanya digunakan sebagai pendeteksi jarak namun dapat dimanfaatkan juga sebagai sensor *scanner* 3D, dimana sensor VL53L0X digunakan untuk membaca dan mendeteksi bentuk atau kontur dari suatu objek.



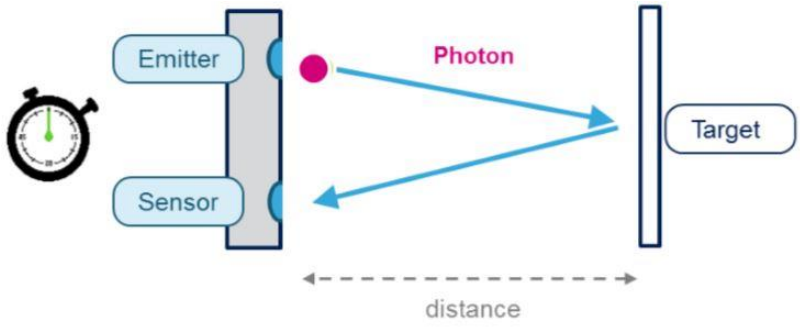
Gambar 2.3 Sensor VL53L0X

Dari Gambar 2.3 dapat diketahui bahwa sensor ini memiliki 6 pin, diantaranya pin VCC yang digunakan untuk sumber daya dengan tegangan 2,6 V hingga 3,5 V, pin GND yang merupakan pin ground untuk referensi nol volt, pin SCL (*Serial Clock Line*) yang merupakan pin clock dari I2C digunakan untuk sinkronisasi transfer data antara sensor dan mikrokontroler, pin SDA (*Serial Data Line*) yang merupakan pin data dari I2C dengan kegunaan seperti pin SCL, pin

XSHUT merupakan pin shutdown eksternal yang ketika ditarik ke level rendah (GND) maka sensor akan dimatikan dan ketika dilepas atau ditarik ke level tinggi maka sensor akan aktif sehingga dapat menghemat daya yang digunakan, serta pin GPIO1 (*General Purpose Input/Output*) yang digunakan untuk fungsi tambahan atau indikasi status tertentu.

Sensor VL53L0X dapat digunakan dengan menggunakan komunikasi I2C dalam pembacaan data sensornya. Pada Arduino sudah tersedia *library* yang dapat digunakan untuk mengakses sensor VL53L0X sehingga pengguna dapat dengan mudah mengakses atau mengolah data dari sensor tersebut karena data pembacaan sensor sudah berupa data matang dengan satuan millimeter (mm). Pada projek ini, satuan millimeter (mm) dikonversikan kedalam satuan centimeter (cm) agar data yang diolah sama dengan data sensor HC-SR04.

Sensor VL53L0X mengintegrasikan SPAD (*Single Photon Avalance Diodes*) terdepan array dan menanamkan *FlightSenseTM* generasi kedua ST teknologi yang dipatenkan. Pemancar VCSEL (*Vertical Cavity Surface-Emitting Laser*) 940 nm VL54L0X sama sekali tidak terlihat oleh mata manusia dan ketika digabungkan dengan fisik filter inframerah memungkinkan jarak jangkau lebih jauh, kekebalan yang lebih tinggi terhadap cahaya sekitar dan struktur lebih baik untuk mengurangi *cross-talk optik* dari kaca penutup.



Gambar 2.4 Prinsip Sensor VL53L0X

Seperti pada gambar 2.4 sensor VL53L0X pertama kali akan memancarkan pulsa cahaya laser inframerah pendek melalui pemancar VCSEL, kemudian cahaya bergerak menuju objek lalu memantul kembali ke sensor untuk diterima menggunakan detektor SPAD, selanjutnya sensor menghitung waktu yang dibutuhkan cahaya untuk melakukan perjalanan dari sensor ke objek dan kembali,

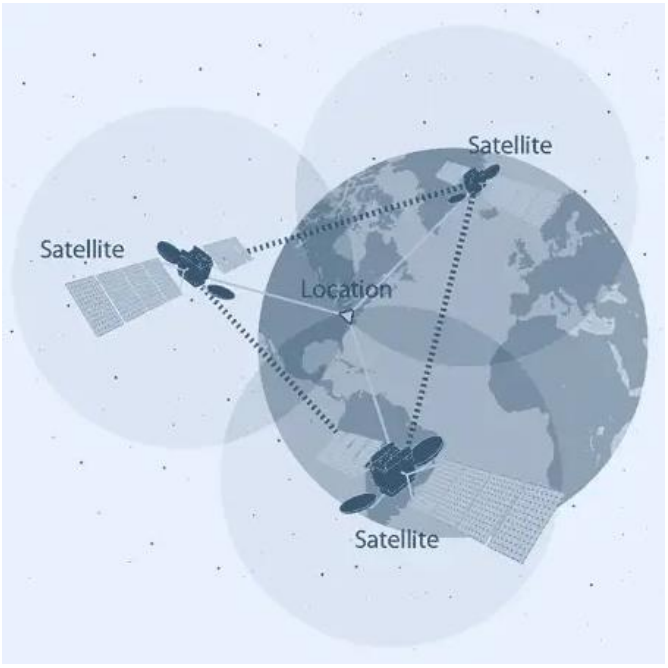
jarak dihitung berdasarkan waktu penerbangan dengan menggunakan kecepatan cahaya yang dapat dirumuskan:

$$\text{Jarak} = \text{Kecepatan Cahaya} \times T / 2$$

Dimana kecepatan cahaya sebesar 3×10^8 m/s atau sekitar 3 juta m/s dan T adalah waktu tempuh dari sensor. Dengan menggunakan prinsip ini, sensor dapat mengukur jarak dengan akurasi tinggi dan respon cepat, membuatnya cocok untuk berbagai aplikasi termasuk deteksi objek dan pemetaan 3D.

2.1.5 Modul GPS Ublox Neo 6m

Global Positioning System (GPS) adalah sistem navigasi berbasis satelit yang memungkinkan perangkat penerima GPS untuk menentukan lokasi di Bumi dengan akurasi tinggi. GPS bekerja dengan memanfaatkan jaringan satelit yang mengorbit Bumi, yang mengirimkan sinyal radio berisi informasi posisi dan waktu kepada penerima GPS di permukaan bumi. Dengan menerima sinyal dari setidaknya tiga satelit, perangkat GPS dapat menghitung lokasi pengguna menggunakan proses yang dikenal sebagai trilaterasi.

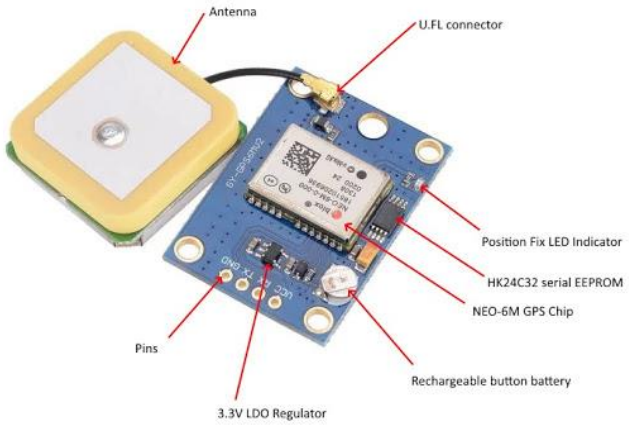


Gambar 2.5 Proses Trilaterasi

Penerima GPS bekerja dengan menentukan jarak mereka dari sejumlah satelit GPS. Setiap satelit mengirimkan sinyal radio yang berisi informasi tentang posisi

satelit dan waktu saat sinyal tersebut dikirimkan. Penerima GPS di bumi menerima sinyal ini dan menghitung waktu tempuh sinyal untuk mencapai penerima. Dengan mengetahui waktu tempuh dan kecepatan sinyal radio (sama dengan kecepatan cahaya), penerima dapat menghitung jarak ke setiap satelit. Setelah menerima sinyal dari minimal tiga satelit, penerima GPS dapat menentukan posisi tiga dimensi (*latitude, longitude,* dan *altitude*) pengguna.

GPS *receiver* U-blox Neo-6m adalah penerima sinyal GPS *stand-alone* yang memiliki performa baik. *Time to First* (TIFF) adalah salah satu parameter penting dimana didefenisikan sebagai kecepatan modul GPS untuk dapat mengakses data almanac dan emepheris dari satelit. TIFF, dan Neo-6m ini memiliki TIFF paling lama 27 detik.



Gambar 2.6 Modul GPS U-blox Neo 6m

Spesifikasi dari GPS U-blox Neo-6m memiliki tegangan input 3~5 volt, dengan sensitivitas navigasi -161 dBm (reakuisisi dari blank-spot: -160 dBm) dan sensitivitas pada saat mulai -147 dBm pada cold-start, -156 dBm pada hot star. Serta, memiliki kecepatan pembaruan data 5hz, memiliki akurasi penetapan lokasi 2,5 meter, memiliki rentang frekuensi 0,25 Hz hingga 1KHz, dengan memiliki akurasi kecepatan 0,1 meter/detik dan akurasi arah 0,5°, dengan Batasan operasi daya tarik maksimum 4x gravitasi, ketinggian maksimum 50 km, kecepatan maksimum 500 meter/detik (1800km/jam) [9].

2.1.6 Arduino IDE

Untuk memprogram board ESP32, maka diperlukan aplikasi IDE (*Integrated Development Environment*) bawaan dari Arduino. Aplikasi ini berguna untuk membuat, membuka, dan mengedit *source code* ESP32 (*Sketches*, para *programmer* menyebut *source code arduino* dengan istilah "*sketch*s"). *Sketch* merupakan *source code* yang berisi logika dan algoritma yang akan diupload ke dalam IC mikrokontroller.

Interface Arduino IDE dibagi menjadi beberapa bagian yaitu:

- *Verify* : pada versi sebelumnya dikenal dengan istilah *Compile*. Sebelum aplikasi diupload ke *board Arduino,* biasakan untuk memverifikasi terlebih dahulu *sketch* yang dibuat. Jika ada kesalahan pada *sketch*, nanti akan muncul *error*. Proses *Verify / Compile* mengubah *sketch* ke *binary code* untuk diupload ke mikrokontroller.
- *Upload* : tombol ini berfungsi untuk mengupload *sketch* ke *board Arduino*. Walaupun tidak mengklik tombol *verify*, maka *sketch* akan di-*compile*, kemudian langsung diupload ke *board*. Berbeda dengan tombol *verify* yang hanya berfungsi untuk memverifikasi *source code* saja.
- *New Sketch* : Membuka *window* dan membuat *sketch* baru.
- *Open Sketch* : Membuka *sketch* yang sudah pernah dibuat. *Sketch* yang dibuat dengan IDE *Arduino* akan disimpan dengan ekstensi file .ino
- *Save Sketch* : menyimpan *sketch*, tapi tidak disertai mengcompile.
- *Serial Monitor* : Membuka *interface* untuk komunikasi serial, nanti akan didiskusikan lebih lanjut pada bagian selanjutnya.
- Keterangan Aplikasi : pesan-pesan yang dilakukan aplikasi akan muncul, misal "*Compiling*" dan "*Done Uploading*" ketika mengcompile dan mengupload *sketch* ke *board Arduino*.
- Konsol : Pesan-pesan yang dikerjakan aplikasi dan pesan-pesan tentang *sketch* akan muncul pada bagian ini. Misal, ketika aplikasi mengcompile atau ketika ada kesalahan pada *sketch* yang kita buat, maka informasi *error* dan baris akan diinformasikan di bagian ini.

- Baris *Sketch* : bagian ini akan menunjukkan posisi baris kursor yang sedang aktif pada *sketch*.
- Informasi *Port* : bagian ini menginformasikan *port* yang dipakah oleh *board Arduino*.

2.1.7 *Android Studio*

Android Studio adalah *Integrated Development Environment* (IDE) atau *software* yang bisa digunakan untuk mengembangkan aplikasi *android*. Dengan menggunakan *Android Studio* proses menciptakan aplikasi *android* lebih efisien karena terdapat beberapa fitur yang tersedia, seperti:

- Sistem Versi *Gradle* yang fleksibel.
- Emulator yang cepat dan dilengkapi kaya fitur.
- Lingkungan yang menyatu mengembangkan aplikasi android bagi semua perangkat android (*Smartphone, Tablet, Smarttv*, dan *Smartwatch*).
- Template kode dan integrasi dengan GitHub untuk membuat fitur aplikasi yang sama dan mengimport kode-kode contoh.
- Alat pengujian dan kerangka kerja yang ekstensif.
- Dukungan C++ dan NDK.
- Dukungan *Google Cloud* Platform sehingga mempermudah saat pengintegrasian *Google Cloud Messaging* dan *App Engine*.

2.1.8 *Firebase*

Firebase atau BaaS (*Backend as a Service*) merupakan layanan dari *Google* yang dapat mempermudah *developer* dalam mengembangkan aplikasi khususnya pada bagian *backend* [10].

Firebase memiliki beberapa fitur, yaitu:

- *Firebase Analytics* yang digunakan sebagai koleksi data dan *reporting* yang memungkinkan untuk membuat segmentasi *user* berdasarkan *user attribute*. Sebagai contoh pada aplikasi *online store*, dengan *user attribute* maka dapat

diketahui jumlah *user* yang membeli barang dengan merk tertentu, serta dapat mengetahui waktu transaksi yang dilakukan user.

- *Firebase Cloud Messaging* (FCM) *and Notifications* digunakan untuk menyediakan koneksi antar *server* dan *device* sehingga dapat mengirim dan menerima pesan serta notifikasi di *Android*, iOS, dan *web* tanpa biaya. Untuk menargetkan pesan lanjutan dengan mudah menggunakan segment. Pesan notifikasi telah terintegrasi dengan *Google Analytics for Firebase*, sehingga user memiliki akses pada interaksi dan *tracking* konversi secara detail.
- *Firebase Authentication* yang digunakan untuk mendukung autentikasi menggunakan nomor telepon, sandi, penyedia identitas gabungan seperti *Google*, *Facebook*, dan sebagainya. *Firebase Authentication* terintegrasi dengan fitur layanan *Firebase* yang memanfaatkan berbagai jenis standar industri, seperti *Oauth* 2.0 dan *OpenID Connect* yang memudahkan integrasi dengan *backend* khusus.
- *Firebase Cloud Firestore* merupakan *database* NoSQL yang dihosting di *cloud* dan dapat diakses melalui SDK *real* oleh aplikasi. *Cloud Firestore* membuat data tetap terkoneksi di aplikasi *user* melalui *listener realtime* dan menawarkan layanan secara *offline* untuk aplikasi seluler dan *web*.
- *Firebase Realtime Database* merupakan *database* yang di hosting melalui *cloud*, data akan disimpan dan dieksekusi dalam bentuk JSON dan disinkron secara *realtime* ke setiap *user* yang terkoneksi, *Realtime Database* juga akan tetap responsif saat *offline*, karena SDK *Firebase Realtime Database* menyimpan data langsung ke *disk device* atau memori lokal, setelah perangkat terhubung ke Internet, perangkat *user* akan menerima setiap perubahan yang terjadi.
- *Firebase Hosting* merupakan layanan hosting konten *web* yang dapat menayangkan konten melalui koneksi yang aman, mengirimkan konten secara cepat, dan mendukung semua jeis konten untuk di *hosting*.