BAB 2
DASAR TEORI

2.1 TINJAUAN PUSTAKA

Penelitian yang dilakukan oleh Dragos Ioan, Irina Petra, dan Lucian Andrei pada tahun 2017 dengan judul "*Performance Analysis of LoRa Technology in Wireless Sensor Networks*" membahas tentang evaluasi kinerja teknologi *LoRa* pada jaringan WSN dengan mempertimbangan nilai *spreading factor*, *area coverage*, dan *data time on air* serta pada saat yang sama untuk sensor *node* dan konsumsi energi dilakukan. Modul *Node LoPy* dengan *transceiver LoRa SX1276* yang diuji dengan mengimplementasikan *wireless sensor networks* pada kebun buah kenari dan diambil selama periode satu minggu. Dalam pengujiannya ketergantungan *data time on air* pada antena *spreading factor* dan besar data, *spreading factor* berdampak besar pada *data time on air* dan selanjutnya pada besar konsumsi tercatat 155 mA dalam transmisi data [3].

Pada tahun 2019 penelitian yang dilakukan oleh Kais Mekki, Eddy Bajic, Frederic Chaxel, dan Fernand Meyer dengan judul "*Concept and Hardware Considerations for Product Service System Achievement in Internet of Things*" membahas tentang analisis konsumsi energi terhadap teknologi dan teknik baru yang akan dimanfaatkan untuk pencapaian *product service system*. Konsepnya terdiri dari pemanfaatan sumber daya perangkat untuk melakukan komputasi informasi dan layanan, bukan hanya mengumpulkan dan mengirimkan data. Manfaatnya meliputi mobilitas, reaksi peristiwa waktu nyata (latensi kecil), skalabilitas tinggi, dan keandalan sistem. Layanan dan komputasi data harus diimplementasikan dalam perangkat (objek *internet of things*) dengan tetap memenuhi perangkat berdaya rendah dan berukuran kecil. Untuk mengimplementasikan dan membuat prototipe *internet of things product service system*, perlu mempresentasikan *platform* perangkat keras yang menjanjikan yaitu ESP32. Dua model ESP32 ini yaitu *Sparkfun ESP32 Things* dan *HelTec ESP32*

LoRa SX1278 diuji dan mendapatkan hasil bahwa *HelTec ESP32 LoRa SX1278* mampu mengembangkan prototipe *product service system* untuk kasus keperluan industri. Hal ini dibuktikan dengan pengukuran konsumsi energi Sparkfun ESP32 Things dan *HelTec ESP32 LoRa SX127*8 pada saat mode *deepsleep* konsumsi energinya turun dari 258 mA menjadi 14 mA hal ini menjadikan *HelTec ESP32 LoRa SX1278* mampu mengembangkan prototipe *product service system* untuk kasus keperluan industri dibanding kinerja dari *Sparkfun ESP32* pada saat kondisi *deep sleep* sebesar 65 μA dengan selisih 215 kali lebih kecil daripada *HelTec ESP32 LoRa SX1278* [4].

Penelitian yang dilakukan oleh Cosmo Capodiferro dan Mauro Mazzei pada tahun 2020 yang berjudul "*an Approach Adopted For Smart Data Generation And Visualization Problems*" pada penelitian ini membahas tentang implementasi dan eksperimen pada kondisi lapangan untuk aplikasi berbasis *internet of things* dimana pada perangkat sensor (*LoRa* RF) yang digunakan ditenagai oleh baterai dan tujuan dari eksperimen lapangan ini bahwa konsumsi daya perangkat pada sensor bergantung pada banyak faktor dan siklus hidup perangkat itu sendiri dari sumber tenaga yang digunakan. Percobaan dilakukan dengan uji coba lapangan aplikasi IoT untuk *Smart Farming* berbasis LoRaWAN, hasil percobaan ditampilkan dalam kaitannya dengan otonomi dan konsumsi energi perangkat dengan mengirimkan muatan data 12 *bytes* dan konfigurasi perangkat LoRa Kelas A dengan *spreading factor* 7 menghasilkan waktu yang dibutuhkan sekitar 40 ms dan besar arus 30 μA dalam kapasitas baterai 1400mAh. Kesimpulan dari penelitian ini yaitu perangkat dengan tenaga baterai dapat mencapai siklus hidup yang maksimal hanya dengan konfigurasi perangkat keras dan perangkat lunak yang sesuai, hal ini dibuktikan dengan penggunakan mode *deep sleep* pada perangkat *LoRa* yang diaplikasikan untuk kegiatan pertanian [5].

Pada tahun 2021 penelitian yang dilakukan oeleh Omar H. Kombo, Santhi Kumaran, dan Alastair Bovim dengan judul "*Design and Application of a Low-Cost, Low Power, LoRa-GSM, IoT Enabled System for Monitoring of Groundwater Resources with Energy Harvesting Integration*" membahas tentang implementasi jaringan sensor nirkabel untuk pemantauan air tanah berbiaya rendah dan berdaya rendah yang mendukung pengambilan keputusan bijaksana dalam pengelolaan air

tanah. Sistem ini dibuat pada plarform Arduino Uno dan mengggabungkan sensor MS5803-14BA dan MB280 yang murah. Untuk menyediakan skema daya rendah, Arduino UNO bangun dalam interval enam jam untuk pengukuran dan pencatatan data ke kartu SD, dan pada interval 12 jam untuk menyampaikan data (dalam *batch*) ke *gateway LoRa*, sebelum kembali ke mode *deep-sleep* untuk sisa waktu. Konsumsi daya rata-rata untuk node akhir di semua siklus sistem adalah 104.081mW. Otonomi daya semua node disediakan oleh baterai *LiPo* isi ulang 3.7V, 5000mAh, dan baterai *Li-Ion* isi ulang 9V, 600mAh, masing-masing, yang didukung oleh pengisi daya surya 6V dan 3W. Komponen pemrosesan dan penyimpanan data, serta visualisasi data, dibuat menggunakan perangkat lunak *open source* dan gratis [6].

Pada tahun 2021 Penelitian karya Francesco Maita dan Luca Maiolo "*Low Power Wireless Sensor Network for Precision Agriculture: a Battery-less Operation Scenario*" yang membahas tentang arsitektur jaringan sensor nirkabel untuk aplikasi pertanian secara presisi. Jaringan dibuat di atas topologi bintang dengan dua protokol konektifits NB-IoT untuk *gateway* dan *LoRa* untuk *node sensor*. Perangkat keras dirancang untuk tahan terhadap kondisi lingkungan yang keras (suhu, kelembapan, dan air) dan dioptimalkan untuk meminimalkan konsumsi daya yang digunakan selama komunikasi data dan *deepsleep* dari *gateway* dan *node sensor*. Data yang diperoleh tentang energi efektif yang dibutuhkan oleh sistem dan perhitungan berdasarkan pengukuran papan daya pada pemanenan, konsumsi, dan kerugian membuktikan bahwa *node sensor* mendukung penyebaran menggunakan superkapasitor dalam skenario tertentu. Secara khusus, hasil yang diperoleh terakhir tes otonomi menunjukkan fleksibilitas sistem. Faktanya, penurunan tegangan terbatas yang terjadi pada malam hari dan kemungkinan untuk memperpanjang jangkauan operasional hingga empat kali lipat, memungkinkan untuk mempertimbangkan skenario dengan tingkat transmisi yang lebih tinggi, muatan yang lebih besar, superkapasitor yang lebih kecil (lebih murah), aplikasi dalam ruangan, dll [7].

Dari seluruh penelitian pada tinjauan pustaka dapat dikaitkan antara penelitian yang dibuat dengan milik Dragos Ioan, Irina Petra, dan Lucian Andrei dengan judul "*Performance Analysis of LoRa Technology in Wireless Sensor*

Networks" pada penelitian tersebut dapat dikembangkan pada uji konsumsi daya perangkat *LoRa* guna untuk dapat melihat lebih rinci dari konsumsi daya itu sendiri hal ini dilakukan dengan memvariasikan *payload* pada perangkat LoRa dengan data yang akan dikirimkan yaitu 8 *bytes*, 16 *bytes*, dan 32 *bytes* dengan perangkat *LoRa* dalam kondisi *deep sleep*.

Tabel 2.1 Referensi Penunjang Penelitian [3]–[7]

Nama Penulis	Judul Penelitian	Tahun	Parameter Konsumsi Energi
Dragos Ioan, Irina Petra, dan Lucian Andrei	*Performance Analysis of LoRa Technology in Wireless Sensor Networks*	2017	Spreading Factor 7 dan 12 dengan *payload* 10 *bytes* sampai 200 b*ytes*
Kais Mekki, Eddy Bajic, Frederic Chaxel, dan Fernand Meyer	*Concept and Hardware Considerations for Product Service System Achievement in Internet of Things*	2019	Spreading Factor 7 dengan *payload* acak
Cosmo Capodiferro dan Mauro Mazzei	*an Approach Adopted For Smart Data Generation And Visualization Problems*	2020	Spreading Factor 7 dengan *payload* 12 *bytes*
Umar H. Kombo, Santhi Kumaran, dan Alastair Bovim	*Design and Application of a Low-Cost, Low Power, LoRa-GSM, IoT Enabled System for Monitoring of Groundwater Resources with Energy Harvesting Integration*	2021	Spreading Factor 7 dengan *payload* acak
Francesco Maita dan Luca Maiolo	*Low Power Wireless Sensor Network for Precision Agriculture: a Battery-less Operation Scenario*	2021	Spreading Factor 7 dengan *payload* 15 *bytes*

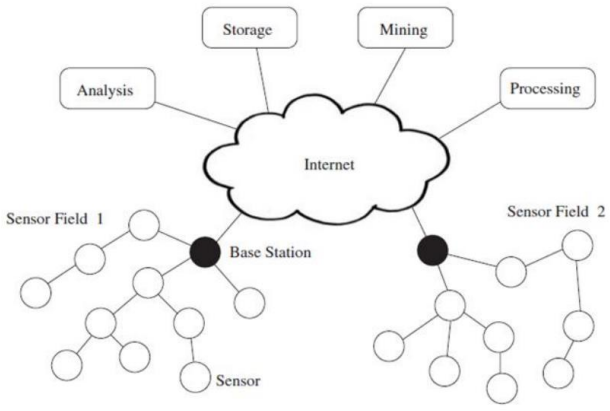
2.2 DASAR TEORI

2.2.1 *Wireless Sensor Network* (*WSN*)

Wireless Sensor Network (WSN) didefinisikan sebagai jaringan yang menghubungkan beberapa perangkat. *Wireless Sensor Network (WSN)* terkenal dengan banyak keuntungannya ketika digunakan dalam berbagai aplikasi. *Wireless*

Sensor Network (*WSN*) juga dikenal sebagai paket keuntungan yang lengkap karena mentransmisikan data yang diterima ke stasiun pangkalan secara kolaboratif memanfaatkan penyimpanan berbiaya rendah, hemat energi, dan *node* mikroprosesor multifungsi yang tidak stabil. *Wireless Sensor Network* adalah suatu sistem di mana sejumlah sensor bekerja sama untuk memantau berbagai kondisi fisik dan lingkungan, antara lain suhu, suara, getaran, gempa bumi, polusi udara, dan sebagainya [8].

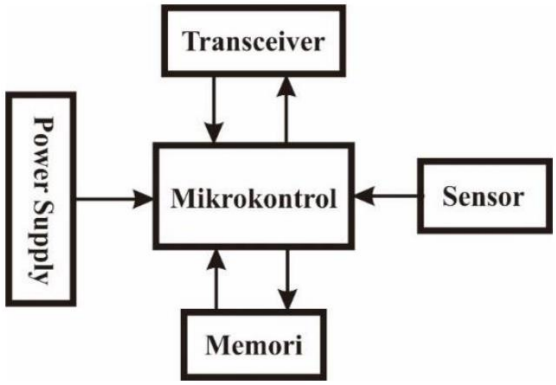
Wireless Sensor Netwotk (WSN) adalah kumpulan *node* yang ditempatkan di jaringan kooperatif, masing-masing dengan kemampuan pemrosesan yang dapat menangani berbagai sensor dan aktuator. Teknologi WSN ini terdiri dari sejumlah kumpulan node sensor yang tersebar dan memiliki cakupan area yang luas yang disebut area sensor, yang mencakup beberapa parameter yang dapat dideteksi. *Node* sensor ini telah dilengkapi dengan banyak komponen yang memungkinkannya untuk melakukan penginderaan, komputasi, dan berinteraksi satu sama lain. Sehingga pengguna dapat mengakses informasi yang mereka butuhkan untuk melakukan pengukuran, observasi, dan menawarkan reaksi terhadap suatu kejadian (insiden) atau fenomena dalam konteks tertentu. Tanpa menggunakan kabel sebagai media transmisi, komunikasi transmisi data antar node sensor dilakukan secara horizontal (dengan *node* sensor yang sama) atau secara vertikal (dengan node koordinasi/*sink node*) (nirkabel). [9]



Gambar 2.1 Ilustrasi Skenario Penggunaan *Wireless Sensor Network* [9]

Komponen *wireless sensor network* diklasifikasikan ke dalam lima kategori: *transceiver*, mikrokontroler, catu daya, memori, dan sensor. *Transceiver* adalah

komponen yang terdiri dari pemancar dan penerima dan digunakan untuk menerima dan mengirimkan data ke perangkat lain melalui protokol IEEE 802.15.4 atau IEEE 802.11b/g. Tugas mikrokontroler adalah melakukan operasi komputasi, mengontrol dan memproses perangkat yang melekat pada mikrokontroler. Catu daya menyediakan energi listrik ke seluruh sistem *wireless sensor network*, memungkinkannya bekerja secara efektif. Memori digunakan untuk menyimpan data dari beberapa *node* sensor. Tugas sensor adalah mendeteksi besaran fisis yang akan diukur, sensor adalah perangkat yang dapat mengubah satu jenis energi menjadi energi lainnya, dalam hal ini menerjemahkan energi terukur menjadi energi listrik, yang kemudian diubah oleh ADC menjadi rangkaian pulsa terkuantisasi yang dapat dibaca oleh mikrokontroler. [9].



Gambar 2.2 Komponen Utama *Wireless Sensor Network* [9]

2.2.2 *Long Range* (*LoRa*)

Aliansi LoRa, yang merupakan singkatan dari *Long Range* teknologi komunikasi nirkabel jarak jauh, mendukung sistem komunikasi nirkabel jarak jauh. Aplikasi yang dimaksudkan sistem ini adalah perangkat bertenaga baterai yang intensif energi dan tahan lama. Meskipun sistem komunikasi *LoRa* juga menyiratkan arsitektur jaringan akses tertentu, istilah "*LoRa*" biasanya mengacu pada dua lapisan berbeda *physical layer*, yang menggunakan teknik modulasi radio *Chirp Spread Spectrum* (CSS), dan protokol *MAC Layer* yang disebut sebagai "LoRaWAN." *Physical Layer* Semtech memungkinkan untuk berkomunikasi jarak jauh dengan daya rendah dan *throughput*, tergantung pada wilayah di mana *LoRa* digunakan, beroperasi pada Band ISM 433 MHz, 868 MHz, atau 915 MHz. Ketika

agregasi saluran digunakan, laju data dapat mencapai hingga 50 Kbps dan muatan setiap transmisi dapat berkisar dari 2 hingga 255 oktet. Teknik modulasi adalah teknologi eksklusif dari Semtech [10].

LoRa adalah metode modulasi *RF* untuk jaringan area luas berdaya rendah. *LoRa* mengacu pada tautan data jarak jauh yang dimungkinkan oleh teknologi ini. *LoRa*, yang dikembangkan oleh Semtech sesuai dengan standar LPWAN, memungkinkan komunikasi jarak jauh hingga 10 mil (15 km) atau lebih (*line of sight*) di daerah pedesaan dan hingga 3 mil (5 km) di daerah perkotaan. Solusi berbasis *LoRa* terkenal karena konsumsi dayanya yang sangat rendah, yang memungkinkan pengembangan perangkat yang dioperasikan dengan baterai dengan masa pakai hingga sepuluh tahun. Jaringan berdasarkan protokol LoRaWAN sangat ideal untuk aplikasi yang memerlukan komunikasi internal jarak jauh atau internal ekstensif antara sejumlah besar perangkat pengumpul data berdaya rendah dalam topologi bintang [11].

Dalam hal masa pakai baterai, energi yang dibutuhkan untuk mengirim paket data dapat diabaikan karena paket data pendek dan hanya ditransfer beberapa kali setiap hari. Selain itu, penggunaan daya dipantau dalam satuan miliwatt (mW) saat *gadget* terakhir dalam mode tidur, memungkinkan baterai perangkat bertahan selama bertahun-tahun. Jaringan LoRaWAN berpotensi menampung jutaan pesan. Namun, jumlah pesan yang didukung di setiap penerapan ditentukan oleh jumlah gateway yang diterapkan. *Gateway* delapan saluran tunggal dapat menangani hingga 1,5 juta pesan per hari. Jika setiap perangkat akhir mengirimkan pesan setiap jam sekali, *gateway* dapat menampung hingga 60.000 per perangkat. Jika sepuluh *gateway* serupa digunakan, jaringan dapat menampung sekitar 100.000 perangkat dan satu juta pesan. Jika diperlukan lebih banyak kapasitas, yang diperlukan hanyalah penambahan *gateway* baru ke jaringan, yang dikenakan biaya. Mengingat kemampuan node akhir dan *gateway* berbasis *LoRa*, hanya beberapa *gateway* yang dikonfigurasi dalam jaringan bintang yang diperlukan untuk mendukung banyak *node* akhir. Ini menyiratkan bahwa biaya modal dan operasi dapat dijaga seminimal mungkin. Selain itu, ketika modul *LoRa* RF tersemat *end-node* berbiaya rendah digabungkan dengan standar LoRaWAN terbuka, laba atas investasi mungkin signifikan [11].

2.2.3 Modulasi *LoRa*

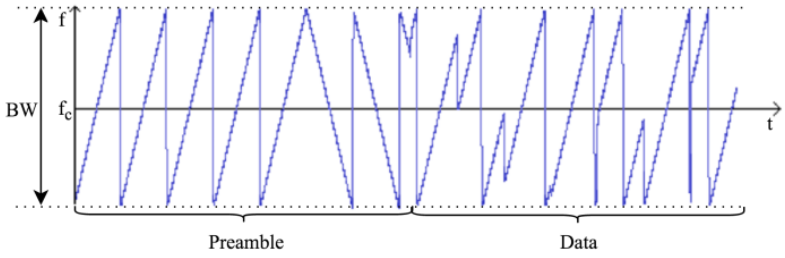
LoRa adalah sistem modulasi *spread-spectrum* yang dipatenkan yang dikembangkan pada teknologi *Chirp Spread Spectrum* (CSS) yang ada yang memungkinkan pertukaran antara sensitivitas dan kecepatan data saat beroperasi pada saluran *bandwidth* tetap 125 KHz atau 500 KHz (untuk saluran *uplink*) dan 500 KHz (untuk saluran *uplink*). Selanjutnya, *LoRa* menggunakan faktor dispersi ortogonal. Dengan melakukan optimalisasi adaptif dari setiap level daya dan kecepatan data node akhir, jaringan dapat menghemat masa pakai baterai node akhir yang terhubung. Misalnya, *end device* yang terletak dekat dengan *gateway* harus mengirimkan data dengan *spread factor* yang rendah, karena *link budget* yang dibutuhkan sangat kecil. Namun, perangkat akhir yang terletak beberapa mil dari *gateway* harus dikirimkan dengan faktor penerapan yang jauh lebih tinggi. *Spreading factor* yang lebih tinggi ini memberikan perolehan pemrosesan yang lebih tinggi, dan sensitivitas penerimaan yang lebih tinggi, meskipun kecepatan data akan lebih rendah [11].

Untuk menyandikan data, *LoRa* menggunakan modulasi *chirp spread spectrum*, frekuensi *chirp* dengan variasi pengulangan langsung dari waktu ke waktu. Karena linearitas *chirp pulse*, *frequency offsets* antara penerima dan pemancar sebanding dengan pengaturan penyeimbang waktu, dengan mudah dihapus di dekoder. Hal ini juga membuat regulasi tahan terhadap efek *doppler*, mirip dengan offset pengulangan. Keseimbangan perulangan di antara pemancar dan penerima pada akhirnya dapat bergantung pada 20% transmisi data tanpa memengaruhi eksekusi penerjemahan. Ini mengurangi biaya pemancar *LoRa*, karena permata yang ditanamkan di pemancar tidak perlu dibuat dengan akurasi yang luar biasa. Penerima *LoRa* dapat mengunci frekuensi yang diterima, menawarkan sensitivitas urutan −130 dBm [12].

Ada sejumlah parameter yang digunakan untuk memodulasi *LoRa*, antara lain *bandwidth* (BW), *spreading factor* (SF), dan *code rate* (CR). *LoRa* mendefinisikan *spreading factor* yang tidak konvensional faktor sebagai logaritma berbasis 2 dalam jumlah *chirps* per simbol. Batas ini memengaruhi modulasi *bitrate* yang efektif, ketahanannya terhadap kebisingan interferensi dan kemudahan

decoding. Parameter modulasi untuk *LoRa* adalah *bandwidth*. 2^{SF} *chirps* adalah sumber sinyal *LoRa* Simbol, yang memiliki probabilitas keberhasilan yang tinggi disertai dengan kicauan berulang di udara. Ketika pita frekuensi maksimum tercapai, frekuensi meningkat, dan frekuensi minimum berkurang. Gambar 2.4 memberikan contoh transmisi *LoRa* dalam variasi frekuensi dari waktu ke waktu. Posisi diskontinuitas dalam frekuensi inilah yang mengkodekan informasi yang dikirimkan. Karena ada 2^{SF} *chirps* dalam simbol, simbol dapat secara efektif menyandikan *bit* informasi SF [12].

Karena durasi simbol LoRa lebih lama dari gangguan AM yang dihasilkan oleh sistem *Frequency Hopping Spread Spectrum* (FHSS), kesalahan yang dihasilkan oleh gangguan tersebut mudah diperbaiki melalui *Forward Error-correction Codes* (FECs). Selektivitas *out-of-channel* yang khas (rasio daya maksimum antara interferensi di *neighboring band* dan sinyal *LoRa*) dan penolakan *co-channel* (rasio daya maksimal antara interferensi di saluran yang sama dan sinyal *LoRa*) dari Penerima *LoRa* masing-masing adalah 90 dB dan 20 dB. Ini mengungguli skema modulasi tradisional, seperti *Frequency-Shift Keying* (FSK), dan membuat *LoRa* cocok untuk transmisi berdaya rendah dan jarak jauh [13], [14].



Gambar 2.3 Variasi frekuensi dari waktu ke waktu dari sinyal sampel yang dipancarkan oleh pemancar *LoRa* [12]

Pada *LoRa*, *chirp rate* hanya bergantung pada *bandwidth chirp rate* sama dengan *bandwidth* (satu *chrip* per detik per *Hertz bandwidth*). Ini memiliki beberapa konsekuensi pada modulasi peningkatan salah satu *spreading factor* akan membagi rentang frekuensi *chirp* menjadi dua (karena 2^{SF} *chirp* menutupi seluruh *bandwidth*) dan juga mengalikan durasi simbol dengan dua. Namun, ini tidak akan membagi kecepatan *bit* menjadi dua, karena satu *bit* lagi akan ditransmisikan di

setiap simbol. Selain itu, laju simbol dan laju *bit* pada *spreading factor* tertentu sebanding dengan *bandwidth* frekuensi, sehingga penggandaan bandwidth secara efektif akan menggandakan laju transmisi. Dapat dilihat dalam persamaan (2.1), yang menghubungkan durasi simbol (TS) dengan *bandwidth* dan *spreading factor*.

$$Ts\frac{2^{FS}}{BW} \quad \textbf{(2.1)}$$

Selain itu, LoRa menyertakan kode koreksi kesalahan maju. Tingkat *code rate* (CR) sama dengan $4/(4+n)$, dengan $n \in \{1,2,3,4\}$. Mempertimbangkan hal ini, serta fakta bahwa bit informasi *SF* ditransmisikan per simbol, persamaan (2.2) memungkinkan seseorang untuk menghitung kecepatan *bit* yang berguna (Rb).

$$Rb = SF \times \frac{BW}{2^{FS}} \times CR \quad \textbf{(2.2)}$$

Misalnya, pengaturan dengan BW = 125 kHz, SF = 7, CR = 4/5 memberikan bit rate Rb = 5,5 kbps. Parameter ini juga mempengaruhi sensitivitas *decoder*. Secara umum, peningkatan *bandwidth* menurunkan sensitivitas penerima, sedangkan peningkatan *spreading factor* meningkatkan sensitivitas penerima. Menurunkan *code rate* membantu mengurangi *Packet Error Rate* (PER) di hadapan interferensi singkat yaitu, paket yang ditransmisikan dengan *code rate* 4/8 akan lebih toleran terhadap interferensi daripada sinyal yang ditransmisikan dengan *code rate* 4/ 5 akan lebih toleran terhadap interferensi daripada sinyal yang dikirimkan pada *code rate* 4/5 akan lebih toleran terhadap interferensi daripada sinyal yang dikirimkan pada *code rate* 4/5 akan lebih toleran terhadap interferensi daripada.

Tabel 2.2 Sensitivitas penerima Semtech SX1276 LoRa dalam dBm pada *bandwidth* dan *spreading factor* yang berbeda [13]

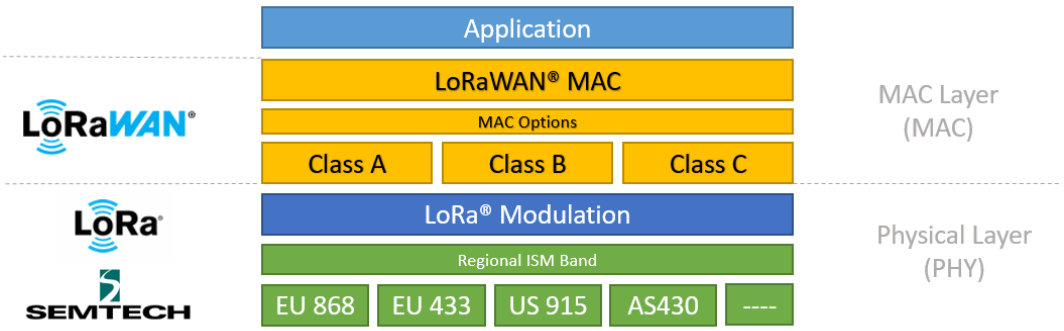
BW / SF	7	8	9	10	11	12
125 kHz	-123	-126	-129	-132	-133	-136
250 kHz	-120	-123	-125	-128	-130	-133
500 kHz	-116	-119	-122	-125	-128	-130

Parameter modulasi *LoRa* lainnya, yang diimplementasikan dalam *transceiver* adalah pengoptimalan kecepatan data yang rendah. Parameter ini wajib pada *LoRa* saat menggunakan *spreading factor* 11 dan 12 dengan *bandwidth* 125 kHz atau lebih rendah. Pengaruh parameter ini tidak didokumentasikan namun,

persamaan (2.3) menunjukkan bahwa ini mengurangi jumlah bit yang ditransmisikan per simbol menjadi dua [13].

2.2.2 Dasar Jaringan LoRaWAN

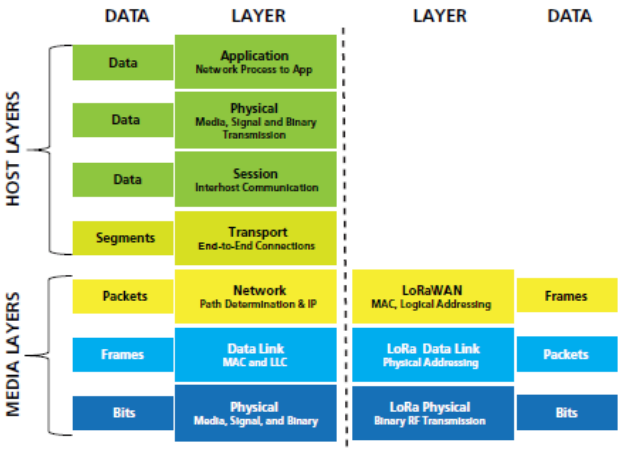
Untuk benar-benar memahami jaringan LoRaWAN, pertama-tama kita harus memeriksa tumpukan teknisnya. *LoRa* adalah lapisan fisik (PHY), seperti yang digambarkan pada Gambar 2.3 dan modulasi nirkabel digunakan untuk menghasilkan jalur komunikasi jarak jauh. Aliansi *LoRa* membakukan dan mengelola LoRaWAN, sebuah teknologi jaringan terbuka yang memungkinkan layanan komunikasi, mobilitas, dan lokasi dua arah yang aman. [11].



Gambar 2.4 Tumpukan Teknologi LoRaWAN [11]

2.2.4 *LoRa Physical Layer*

LoRa hanyalah implementasi lapisan fisik (PHY), atau "bit", seperti yang ditentukan oleh Model Jaringan tujuh lapis OSI (ditunjukkan pada Gambar 2.5). Alih-alih menggunakan kabel, udara digunakan sebagai pembawa untuk mentransfer gelombang radio *LoRa* dari Pemancar *Radio Frequency* di perangkat *Internet of Things* ke penerima *Radio Frequency* di gateway.

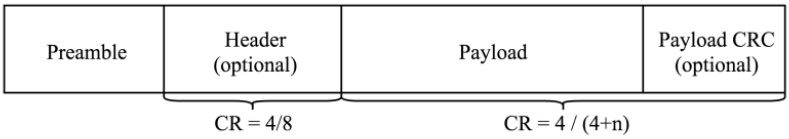


Gambar 2.5 Tujuh Lapisan Jaringan OSI Layer [11]

Walaupun modulasi *LoRa* sanggup digunakan buat mentransmisikan *frame arbitrer*, format *physical frame* ditetapkan serta diterapkan pada pemancar serta penerima. *Bandwidth* serta *spreading factor* konstan buat *frame*. *Frame LoRa* diawali dengan pembukaan. Pembukaan diawali dengan urutan upchirp konstan yang mencakup segala *band frekuensi*. 2 *upchirp* terakhir menyandikan kata sinkronisasi. Kata sinkronisasi merupakan nilai satu *byte* yang digunakan buat membedakan jaringan *LoRa* yang memakai band frekuensi yang sama. Fitur yang dikonfigurasi dengan kata sinkronisasi yang diberikan hendak menyudahi menjajaki transmisi bila kata sinkronisasi yang didekodekan tidak sesuai dengan konfigurasinya. Kata sinkronisasi diiringi oleh 2 seperempat *downchirps*, dengan durasi 2, 25 simbol. Total durasi pembukaan ini bisa dikonfigurasi antara 10. 25 serta 65. 539, 25 simbol. Struktur pembukaan dapat dilihat pada Foto 2. 2 [12].

Sesudah pembukaan, terselip *header* opsional. Apabila terdapat, *header* ini ditransmisikan dengan laju kode 4/ 8. Ini menampilkan ukuran *payload* (dalam *byte*), tingkat pengkodean yang digunakan pada akhir transmisi, dan apakah CRC 16-bit *payload* ada atau tidak di akhir *frame*. CRC juga disertakan di *header* untuk memungkinkan penerima mengabaikan paket dengan *header* yang salah. Dimensi muatan disimpan dengan satu *bytes*, membatasi dimensi muatan hingga 255 *bytes*. *Header* bertabiat opsional buat membolehkan penonaktifan dalam suasana yang tidak dibutuhkan, misalnya pada saat panjang *payload*, laju pengkodean, dan keberadaan CRC dikenal lebih dahulu. *Payload* dikirim sesudah header, dan di

akhir *frame* merupakan CRC opsional. Gambar 2.6 menunjukkan ringkasan *format frame*. [10].



Gambar 2.6 Struktur *Frame LoRa.* $n \epsilon \{1..4\}$ [10]

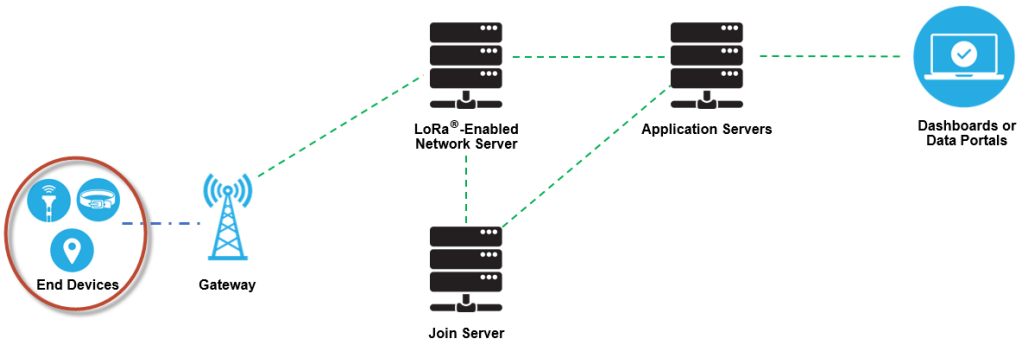
Persamaan (3) memberikan jumlah simbol yang diperlukan untuk mengirimkan *payload* n_s sebagai fungsi dari semua parameter ini. Nomor ini harus ditambahkan ke nomor simbol pembukaan, untuk menghitung ukuran total paket dalam simbol. Dalam persamaan ini, PL adalah ukuran *payload* dalam *byte*, CRC adalah 16 jika CRC diaktifkan dan nol jika tidak, H adalah 20 jika *header* diaktifkan dan nol jika tidak, dan DE adalah dua jika pengoptimalan *data rate* rendah diaktifkan dan nol jika tidak. Persamaan ini juga menunjukkan bahwa ukuran minimum sebuah paket adalah delapan simbol [13].

$$n_s = 8 + \max\left(\left\lceil \frac{8PL-4SF+8+CRC+H}{4\times(SF-DE)} \right\rceil \times \frac{4}{CR}, 0\right) \quad \textbf{(2.3)}$$

2.2.5 Protokol *LoRaWAN*

LoRaWAN adalah protokol MAC yang dibuat untuk menggunakan *physical layer LoRa*, dirancang terutama untuk jaringan sensor, di mana sensor bertukar paket dengan server dengan kecepatan data rendah dan interval waktu yang relatif lama (satu transmisi per jam atau bahkan hari). Beberapa komponen jaringan ditentukan dalam spesifikasi *LoRaWAN* dan diperlukan untuk membentuk jaringan *LoRaWAN*, seperti *end device*, *gateway* (misal *base stasion*), dan *network server*.

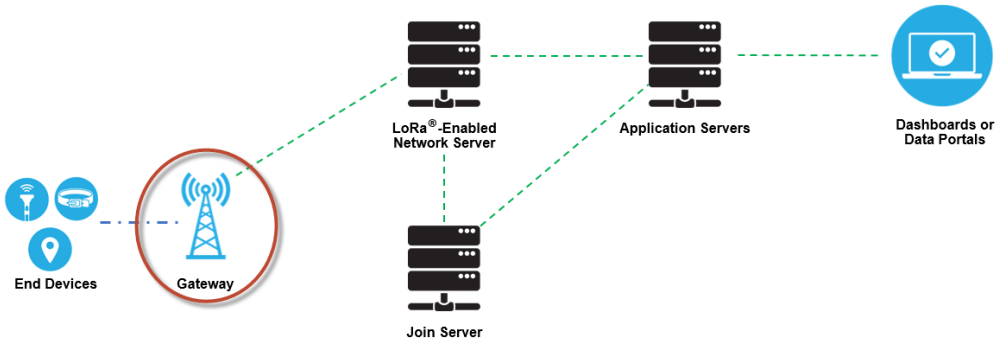
a. *End Device* berfungsi sebagai *interface* untuk mengumpulkan informasi data yang terkumpul, kemudian dikirim ke *physical layer LoRa* dan dikirim ke *gateway*. Jika perangkat akhir berada di dua wilayah jangkauan *gateway*, perangkat akhir akan mengirim data ke kedua *gateway*.



Gambar 2.7 End Device LoRa[11]

Sensor atau aktuator yang memungkinkan LoRaWAN terhubung secara nirkabel ke jaringan LoRaWAN melalui *gateway* radio menggunakan Modulasi *Radio Frequency LoRa*. Perangkat akhir di sebagian besar aplikasi adalah sensor otonom, umumnya bertenaga baterai, yang mendigitalkan keadaan fisik dan kejadian lingkungan. Aktuator umumnya digunakan untuk lampu jalan, kunci nirkabel, penutup katup air, pencegahan kebocoran, dan aplikasi lainnya. Perangkat berbasis *LoRa* diberi identifikasi unik saat diproduksi. ID ini digunakan untuk mengaktifkan dan mengatur perangkat secara aman, untuk menyediakan pengiriman paket yang aman melalui jaringan pribadi atau publik, dan untuk mengirim data terenkripsi ke *Cloud*. [11].

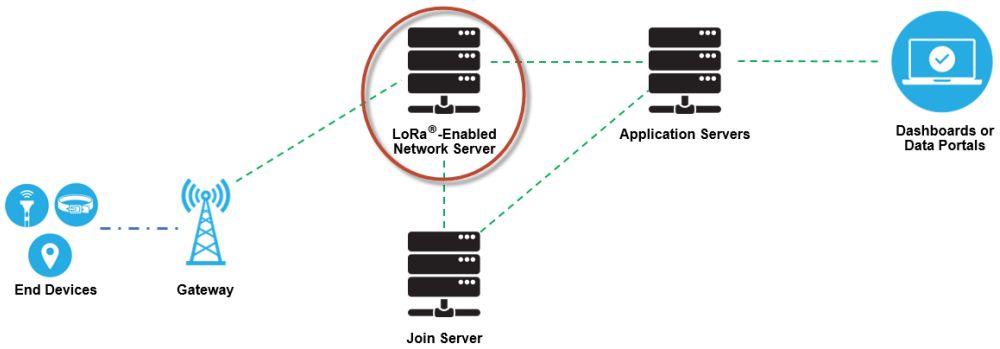
b. *Gateway*, perangkat perantara yang merutekan paket masuk dari perangkat akhir ke server jaringan menggunakan antarmuka backhaul IP dengan *throughput* yang lebih tinggi, seperti Ethernet atau 3G.. Mungkin ada beberapa *gateway* dalam penerapan *LoRa*, dan paket data yang sama dapat diterima (dan diteruskan) oleh lebih dari satu *gateway*



Gambar 2.8 Gateway dalam penerapan jaringan LoRaWAN pada umumnya [11]

Gateway LoRaWAN menerima sinyal RF termodulasi LoRa dari perangkat akhir mana pun dalam jarak pendengaran dan mengirimkannya ke LoRaWAN *Network Server* (LNS) LoRaWAN, yang dihubungkan oleh LoRa dan LoRaWAN. Tidak ada koneksi yang stabil antara perangkat akhir dan *gateway* tertentu. Sebaliknya, sensor yang sama dapat disuplai oleh banyak *gateway* di sekitarnya. Seperti yang ditunjukkan pada Gambar 2.8, setiap paket uplink yang disediakan oleh perangkat akhir diterima oleh semua *gateway* dalam jangkauan. Konfigurasi ini sangat menurunkan tingkat kesalahan paket (karena kemungkinan setidaknya satu *gateway* menerima pesan cukup tinggi), secara signifikan menurunkan *overhead* baterai untuk sensor seluler/nomaden, dan memungkinkan geolokasi berbiaya rendah (dengan asumsi gateway yang bersangkutan memiliki kemampuan geolokasi). Lalu lintas IP dapat dialihkan dari gateway ke server jaringan menggunakan Wi-Fi, Ethernet bawaan, atau koneksi seluler. *Gateway* LoRaWAN berfungsi sepenuhnya pada lapisan fisik dan pada dasarnya tidak lebih dari pembawa pesan radio LoRa. Mereka hanya memeriksa integritas data dari setiap komunikasi RF LoRa yang datang. Pesan ditolak jika integritasnya tidak utuh, yaitu jika CRC salah. Jika benar, gateway mengirimkannya ke LNS bersama dengan metadata tertentu seperti tingkat penerimaan RSSI pesan dan stempel waktu opsional. *Gateway* memproses permintaan transmisi dari LNS untuk *downlink* LoRaWAN tanpa interpretasi muatan apa pun. Karena kemungkinan beberapa *gateway* menerima pesan *LoRa* RF yang sama dari perangkat ujung tunggal, LNS mengeksekusi de-duplikasi data dan menghapus semua salinan. Saat mentransmisikan pesan *downlink*, server jaringan biasanya memilih *gateway* yang mendapatkan pesan dengan RSSI terbaik berdasarkan tingkat RSSI dari pesan identik karena *gateway* tersebut adalah yang terdekat dengan perangkat akhir yang dimaksud. [11].

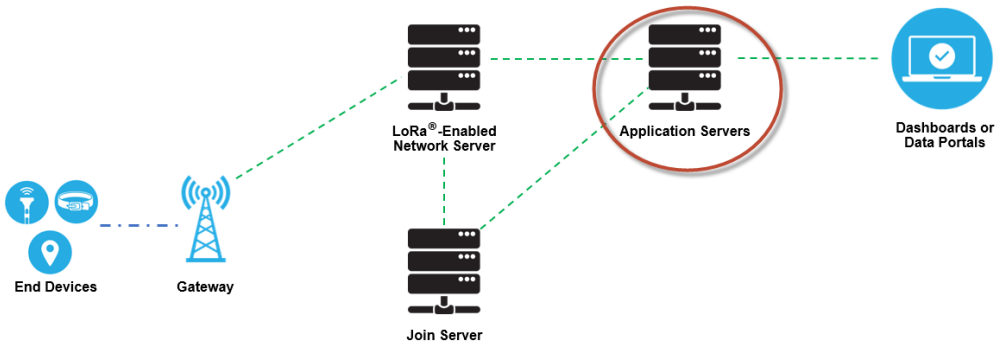
c. *Network Server*, bertanggung jawab atas konfigurasi jaringan, pemfilteran paket, pemeriksaan keamanan, dan penyesuaian *adaptive data rate* (ADR). Akibatnya, pengguna dapat memantau keefektifan alat secara *real time*.



Gambar 2.9 *Network Server* dalam penyebaran jaringan LoRaWAN[11]

LoRaWAN *Network Server* (LNS) mengelola seluruh jaringan, secara dinamis mengontrol parameter jaringan untuk menyesuaikan sistem dengan kondisi yang berubah, dan membuat koneksi AES 128-bit yang aman untuk transportasi data *end-to-end* (dari perangkat akhir LoRaWAN ke pengguna akhir Aplikasi di *Cloud*) serta kontrol lalu lintas dari perangkat akhir LoRaWAN ke LNS (dan sebaliknya). Server jaringan menjaga validitas semua sensor jaringan dan integritas semua pesan. Secara bersamaan, server jaringan tidak dapat membaca atau mengakses data aplikasi [11].

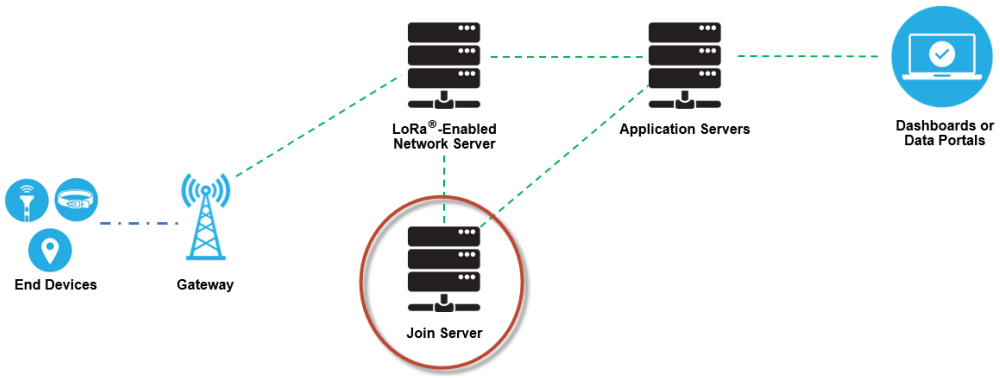
d. *Aplication Servers*, Server aplikasi bertugas memproses, mengelola, dan menganalisis data aplikasi sensor dengan cara yang aman. Mereka juga membuat muatan downlink seluruh lapisan aplikasi untuk perangkat akhir terkait.



Gambar 2.10 *Aplication Servers* dalam penyebaran jaringan LoRaWAN [11]

e. *Join Servers*, fungsi bertugas mengelola prosedur aktivasi *over-the-air* untuk perangkat akhir jaringan baru. *Join server* menyimpan informasi yang dibutuhkan untuk memproses uplink *join-request frame* dan membuat *downlink*

join-accept frame. Ini memberi tahu server jaringan tentang server aplikasi mana yang harus terhubung ke perangkat akhir dan menghasilkan kunci enkripsi sesi aplikasi dan jaringan. Ini mengirimkan Kunci Sesi Jaringan perangkat ke server jaringan dan Kunci Sesi Aplikasi ke server aplikasi yang relevan. [11]

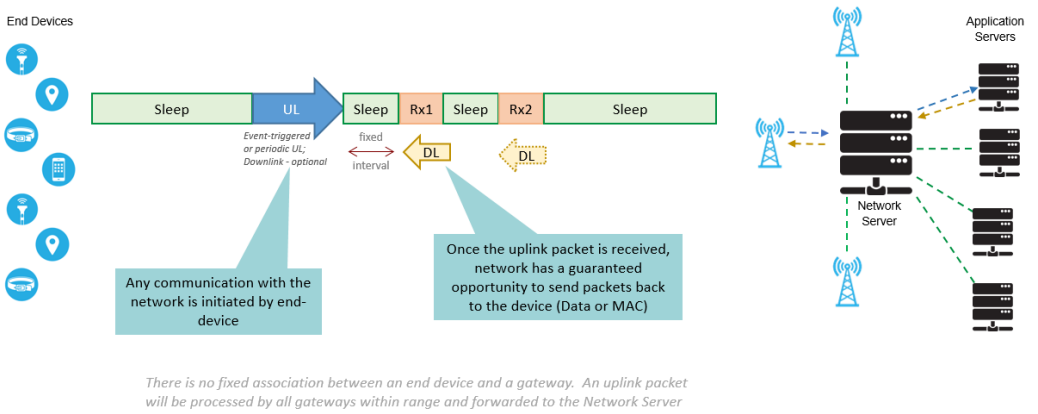


Gambar 2.11 *Join Servers* dalam penyebaran jaringan LoRaWAN [11]

Tidak seperti jaringan seluler tradisional, *end device* tidak dikaitkan dengan *gateway* tertentu agar memiliki akses ke jaringan. *Gateway* hanya berfungsi sebagai *relay* dan penerusan lapisan paket yang diterima dari perangkat akhir ke *network server* setelah menambahkan informasi mengenai kualitas penerimaan. Dengan demikian, *end device* dikaitkan dengan *network server* yang bertanggung jawab untuk mendeteksi paket duplikat, memilih *gateway* yang sesuai untuk mengirim balasan (jika ada), akibatnya mengirim kembali paket ke *end device*. *LoRaWAN* memiliki tiga kelas perangkat akhir yang berbeda untuk memenuhi berbagai kebutuhan aplikasi :

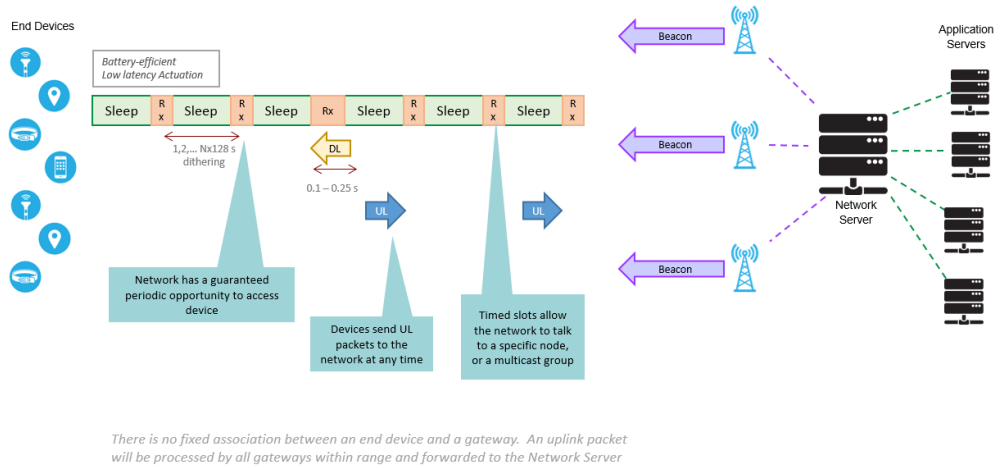
a. Kelas A (*bi-directional*): *End Device* Kelas A dapat menjadwalkan transmisi *uplink* berdasarkan kebutuhannya sendiri, dengan *jitter* kecil (variasi acak sebelum transmisi). Kelas perangkat ini memungkinkan komunikasi dua arah, di mana setiap transmisi *uplink* diikuti oleh dua jendela penerima *downlink* pendek. Transmisi *downlink* dari server pada waktu lain harus menunggu hingga transmisi *uplink* berikutnya terjadi. Perangkat Kelas A memiliki konsumsi daya terendah, tetapi juga menawarkan lebih sedikit fleksibilitas pada transmisi *downlink*. Dalam hal ini, perangkat akhir pada dasarnya tidak aktif (yaitu, dalam mode tidur). Saat lingkungan terkait dengan perangkat apa pun yang dirancang

untuk memantau perubahan, gadget bangun dan mengaktifkan uplink, mengirim data tentang kondisi yang berubah kembali ke jaringan (Tx). Gadget kemudian menunggu tanggapan dari jaringan, yang biasanya memerlukan waktu satu detik (walaupun durasi ini dapat dikonfigurasi). Jika tidak mendapatkan *downlink* selama jendela penerimaan ini (Rx1), ia akan kembali tidur untuk beberapa saat, kemudian bangun dan menunggu respons (Rx2). Jika tidak ada respons yang diterima dalam jendela Rx kedua ini, perangkat akan kembali ke mode *sleep* hingga memiliki data untuk dilaporkan lagi. Penundaan antara Rx1 dan Rx2 disesuaikan dengan waktu antara akhir transmisi uplink dan dimulainya transmisi *downlink*. [11].



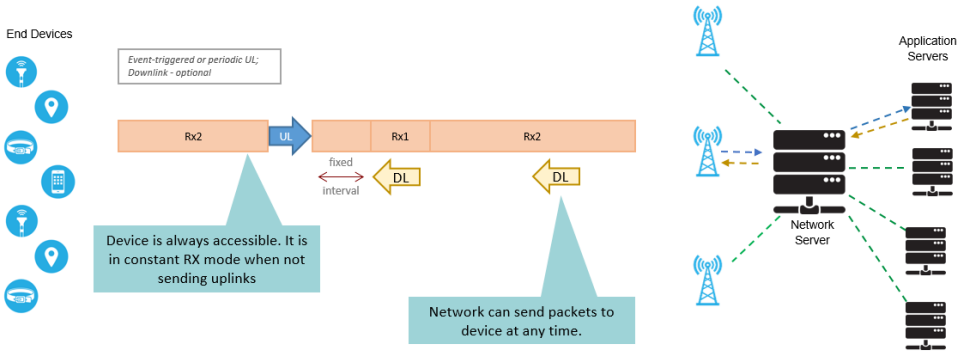
Gambar 2.12 Operasi Kelas A [11]

b. Kelas B (*bi-directional* dengan slot *receive* terjadwal): *End Device* Kelas B membuka jendela penerimaan tambahan pada waktu yang dijadwalkan. Oleh karena itu, *beacon* yang disinkronkan dari *gateway* diperlukan, sehingga *network server* dapat mengetahui kapan *end device* sedang berkomunikasi. Sebuah teknik yang dikenal sebagai *beacon* diperlukan agar mode komunikasi Kelas B berfungsi. *Beacon* yang disinkronkan waktu harus ditransmisikan secara teratur oleh jaringan melalui *gateway* selama prosedur *beacon*, seperti yang ditunjukkan pada Gambar 2.13. Perangkat akhir harus menerima salah satu suar jaringan ini secara teratur untuk menyinkronkan referensi waktu internalnya dengan jaringan.



Gambar 2.13 Operasi Suar Kelas B [11]

c. Kelas C (*bi-directional* dengan slot *receive* maksimal): *End Device* Kelas C memiliki jendela penerimaan yang hampir terus menerus. Dengan demikian mereka memiliki konsumsi daya maksimum. Perangkat Kelas C selalu "*on*", yang berarti tidak bergantung pada daya baterai. Lampu jalan, meteran listrik, dan peralatan Kelas C lainnya adalah contohnya. Kecuali mereka mengirim uplink, perangkat ini terus menunggu sinyal downlink. Akibatnya, mereka memberikan latensi terendah untuk komunikasi *server-to-end-device*. Perangkat akhir Kelas C mengimplementasikan dua *window* penerima yang sama dengan perangkat Kelas A, tetapi jendela Rx2 tidak ditutup hingga transmisi berikutnya dikirim kembali ke server. Akibatnya, mereka dapat menerima *downlink* hampir setiap saat di jendela Rx2. Seperti yang ditunjukkan pada Gambar 2.14, *window* pada frekuensi Rx2 dan kecepatan data juga dibuka antara akhir transmisi dan awal *window* penerimaan Rx1. [11]



Gambar 2.14 Operasi Kelas C [11]

Perlu dicatat bahwa *LoRaWAN* tidak mengaktifkan komunikasi *device to device*, paket hanya dapat dikirim dari *end device* ke *network server* atau sebaliknya. Komunikasi *device to device* (jika diperlukan) dengan demikian harus ditembakkan melalui *network server* (dan akibatnya, melalui dua transmisi *gateway*). Spesifikasi *LoRaWAN* menyatakan bahwa jaringan *LoRaWAN* harus menggunakan *band* frekuensi ISM. *Band* ini berada pada peraturan tentang daya transmisi maksimum dan siklus kerja. Batasan siklus tugas ini diterjemahkan ke dalam penundaan antara *frame* berurutan yang dikirim oleh perangkat. Jika batasannya 1%, perangkat harus menunggu 100 kali durasi *frame* terakhir sebelum mengirim lagi di saluran yang sama [10].

2.2.6 ESP32

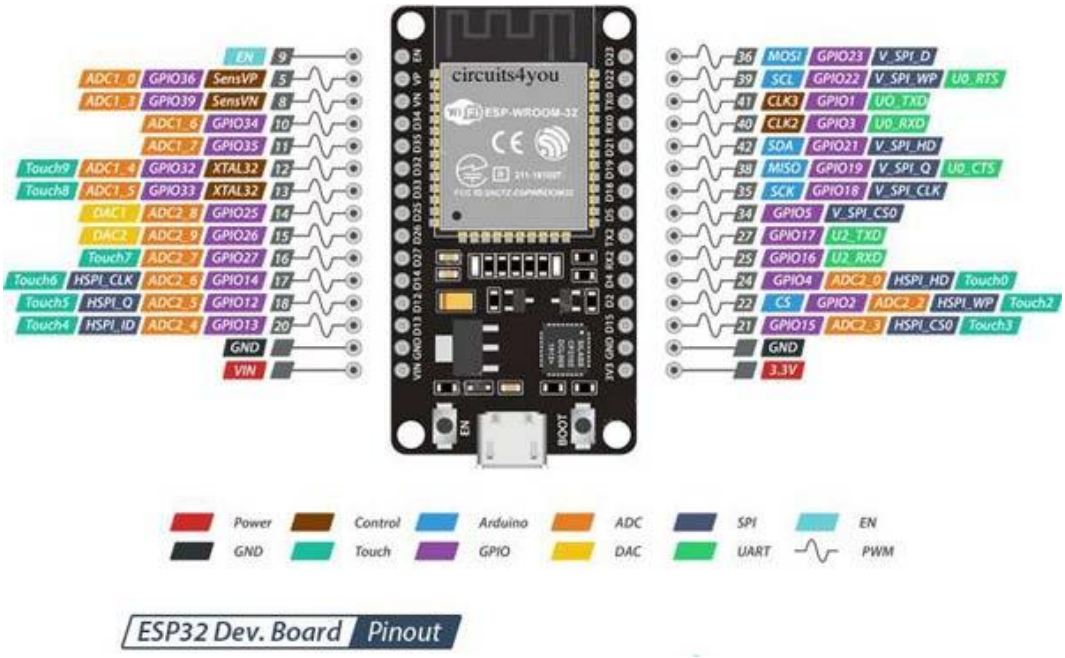
ESP32 adalah mikrokontroler SoC (*System on Chip*) yang mencakup *Wi-Fi* 802.11 b/g/n, mode *Bluetooth* ganda tipe 4.2, dan periferal yang berbeda. *Chip* ESP32 merupakan kelanjutan dari chip ESP8266 utama dalam aplikasi router inti kembar dengan berbagai jenis jam hingga 240 MHz. Ini tidak hanya memiliki fungsi ini, tetapi juga menaikkan jumlah pin GPIO dari 17 menjadi 36, jumlah saluran PWM per 16 dan dilengkapi dengan memori *flash* 4MB. Espressif *Systems* menciptakan *chip* ESP32, dan perusahaan saat ini menyediakan banyak *SoC* tipe ESP32, termasuk ESP32 *Developer Kit*, ESP32 *Wrover Kit*, yang berisi kartu SD dan layar LCD 3,2 inci, dan kit ESP32 Azure *IoT*, yang termasuk Jembatan USB dan sensor bawaan lainnya. Selain Sistem Espresif, pabrikan tambahan

berkomitmen untuk chip ini*:* SparkFun dengan ESP32 Thing DB-nya, WeMoS dengan TTGO, D1, Lolin32 dan Lolin D32, Adafruid (dengan Huzzah32), Robot DF (ESP32 *FireBeatle*), dan banyak lagi. [15].

Tabel 2.3 Perbandingan ESP8266 dengan ESP32 [15]

Spesifikasi	Board	
	ESP8266	**ESP32**
MCU	Xtensa Single-core 32-bit L106	Xtensa Dual-Core 32-bit LX6 with
Wi-Fi	802.11 b/g/n tipe HT20	802.11 b/g/n tipe HT40
Bluetooth T	Tidak Ada	Bluetooth 4.2 dan BLE
Frekuensi	80 MHz	160 MHz
SRAM	Tidak Ada	Ada
Total GPIO	17 pin	36 pin
Total ADC pin	1 pin	15 pin
Total Digital pin	9 pin	2 pin
Tegangan Output	3.3 – 5 Volt	3.3 – 5 Volt
Total SPI-UART-I2C-I2S	2-2-1-2	4-2-2-2
Resolusi ADC	10 bit	12 bit
Suhu operasional kerja	-40°C hingga 125°C	-40°C hingga 125°C
Sensor dalam modul	Tidak ada	Touch Sensor, Temp-erature Sensor, Hall Effect Sensor
Harga di pasaran	Rp. 30.000 – 350.000	Rp. 70.000 – 650.000

ESP32 memiliki dua inti (prosesor Xtensa LX6 dibuat dengan teknologi 40nm). Inti CPU dapat diatur secara independen. *Chip* ini mencakup SRAM 520 KB untuk data dan instruksi. Beberapa SoC, seperti ESP32 *Rover*, menyertakan Flash SPI eksternal 4MB dan PSRAM SPI (*pseudostatic* RAM) 8MB. Bergantung pada jenis papan, kita dapat menggunakan SPI, I2S, I2C, CAN, UART, *Ethernet* MAC, dan sejumlah variabel IR. Peralatan standar juga mencakup sensor efek *hall*, sensor suhu, dan sensor sentuh, sensor tersemat lainnya yang diterapkan di Azure IoT dan kit pengembang. SoC juga menawarkan kriptografi yang dipercepat perangkat keras: *Elliptic Curve Encryption* (ECC), AES, SHA-2, RSA, dan *Random Number Generator* (RNG). Gambar 2.15 menggambarkan versi mikrokontroler ESP32 Developer Kit yang peneliti gunakan [15].



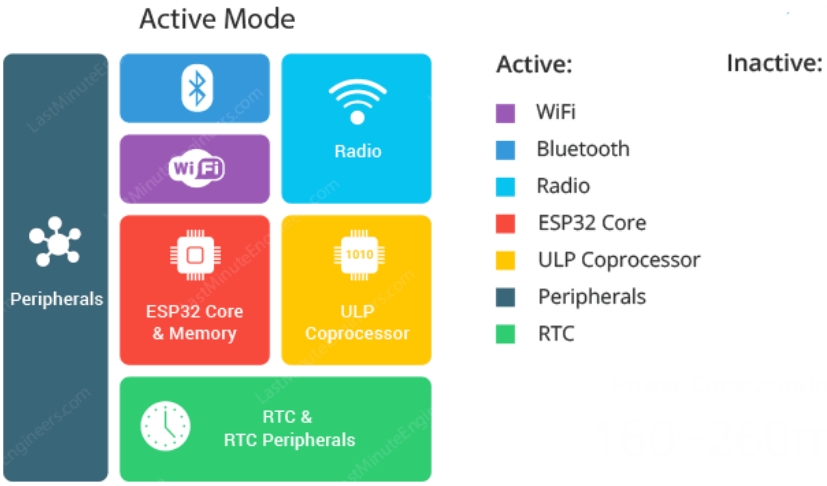
Gambar 2.15 Papan ESP32 Developer Kit [15]

ESP32 tidak dapat disangkal sebagai pesaing yang layak bagi banyak SoC WiFi/MCU, karena mengungguli mereka dalam hal kinerja dan harga. Namun, tergantung pada mode yang digunakan, ESP32 dapat menjadi perangkat yang relatif boros daya. Ketika proyek IoT Anda ditenagai oleh stopkontak listrik, konsumsi daya menjadi perhatian kecil, namun jika Anda berencana untuk memberi daya pada proyek Anda dengan baterai, setiap mA sangat diperhitungkan. Solusinya adalah memanfaatkan salah satu mode tidur ESP32 untuk mengurangi konsumsi daya. Ini adalah strategi yang sangat baik untuk meningkatkan masa pakai baterai secara signifikan untuk proyek yang tidak perlu aktif sepanjang waktu. Mengetahui apa yang ada di dalam chip akan membantu kita lebih memahami bagaimana ESP32 mengelola penghematan daya. Diagram blok chip ESP32 ditunjukkan pada gambar

Gambar 2.16 Diagram blok chip ESP32 [2]

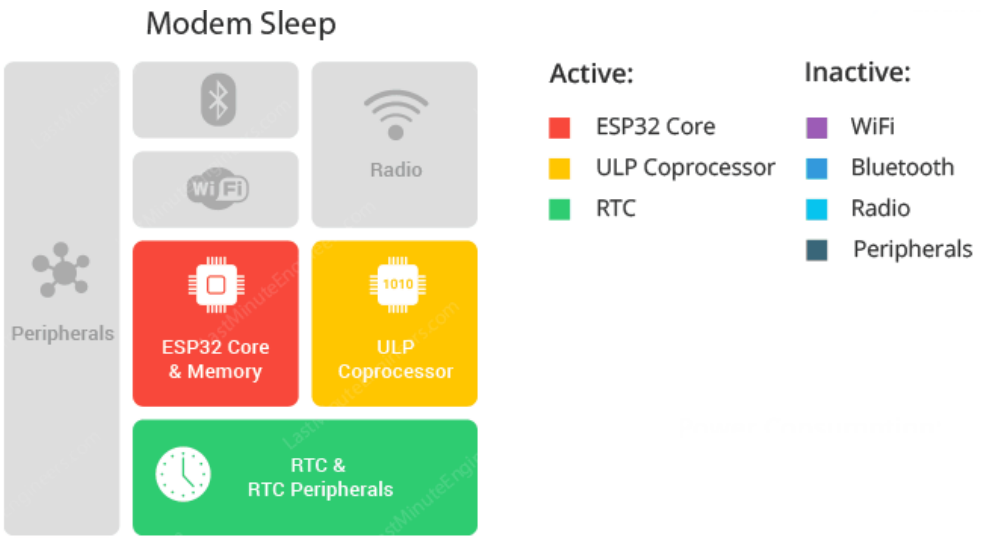
Chip ESP32 berisi mikroprosesor *dual-core* 32-bit bersama dengan ROM 448 KB, SRAM 520 KB, dan memori flash 4 MB. Selain itu, *chip* tersebut berisi modul *WiFi*, modul *Bluetooth*, *cryptographic accelerator* (prosesor bersama yang dirancang khusus untuk melakukan operasi kriptografi), modul RTC, dan sejumlah periferal. Mode Daya pada ESP32 adalah serangkaian konfigurasi yang dapat diatur untuk mengatur penggunaan daya pada perangkat ESP32. Ada beberapa mode daya yang tersedia di ESP32, yaitu:

a. *Active Mode*, mode penggunaan daya penuh pada ESP32, di mana semua komponen perangkat aktif dan beroperasi pada kecepatan penuh.



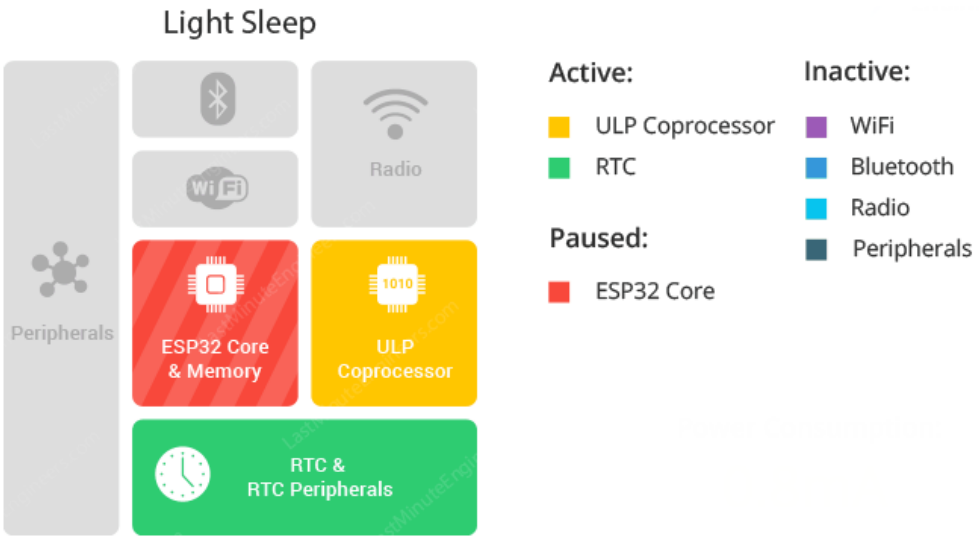
Gambar 2.17 *Active Mode* [2]

b. *Sleep Mode*, Mode ini mengurangi konsumsi daya dengan menonaktifkan beberapa bagian perangkat, seperti CPU dan modul WiFi. Namun, perangkat masih dapat menerima sinyal dan memproses beberapa input.



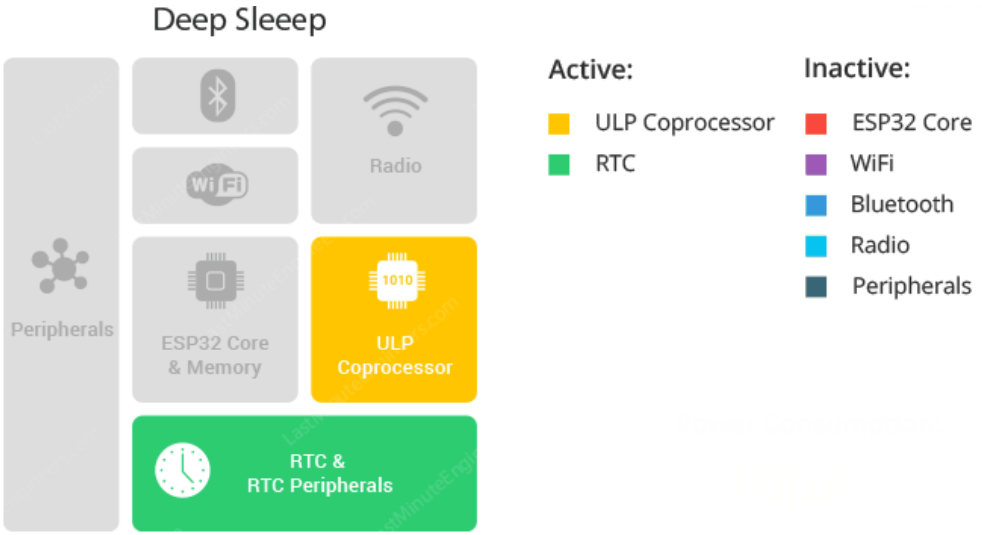
Gambar 2.18 *Sleep Mode* [2]

c. *Light Sleep Mode*, *light sleep* mirip dengan *sleep mode* karena chip mengikuti Pola Tidur Asosiasi. Satu-satunya perbedaan adalah bahwa dalam mode tidur ringan, CPU, sebagian besar RAM, dan periferal digital memiliki *clock gateway*.



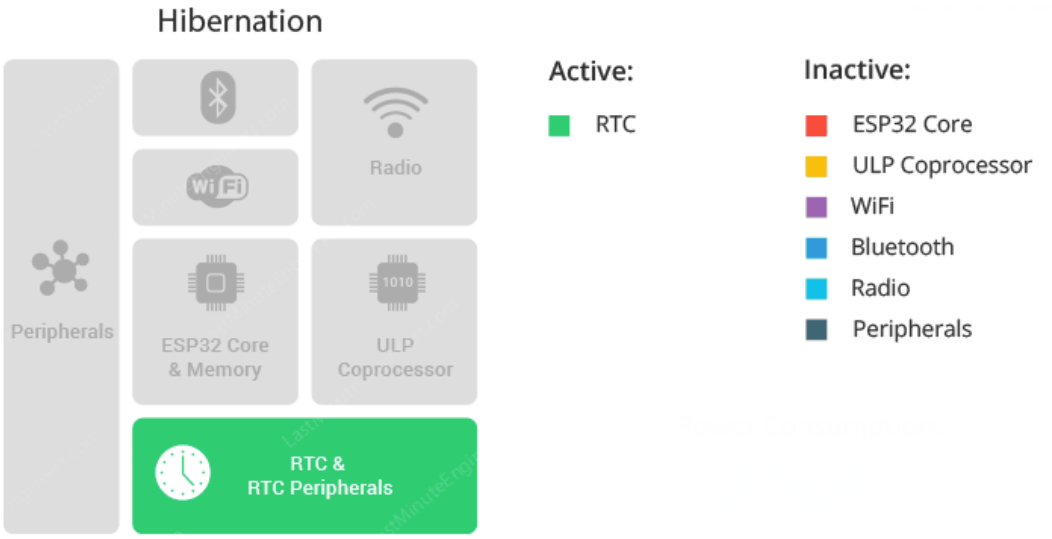
Gambar 2.19 *Light Sleep Mode* [2]

d. *Deep Sleep Mode*, pengguaan hemat daya yang paling ekstrem pada ESP32. Pada mode ini, sebagian besar perangkat dalam kondisi mati, termasuk *CPU* dan memori. Hanya beberapa bagian yang tetap aktif, seperti RTC (*Real Time Clock*) dan beberapa pin eksternal. Modus ini sangat cocok untuk aplikasi yang membutuhkan waktu istirahat yang lama dan konsumsi daya yang sangat rendah.



Gambar 2.20 *Deep Sleep Mode* [2]

e. *Hibernation Mode*, mode hibernasi sangat mirip dengan tidur nyenyak. Satu - satunya perbedaan adalah bahwa dalam mode hibernasi, *chip* menonaktifkan osilator internal 8 MHz serta koprosesor ULP, hanya menyisakan satu pengatur waktu RTC (pada jam lambat) dan beberapa GPIO RTC untuk membangunkan chip. Karena memori pemulihan RTC juga dimatikan, kami tidak dapat menyimpan data apa pun saat dalam mode hibernasi.



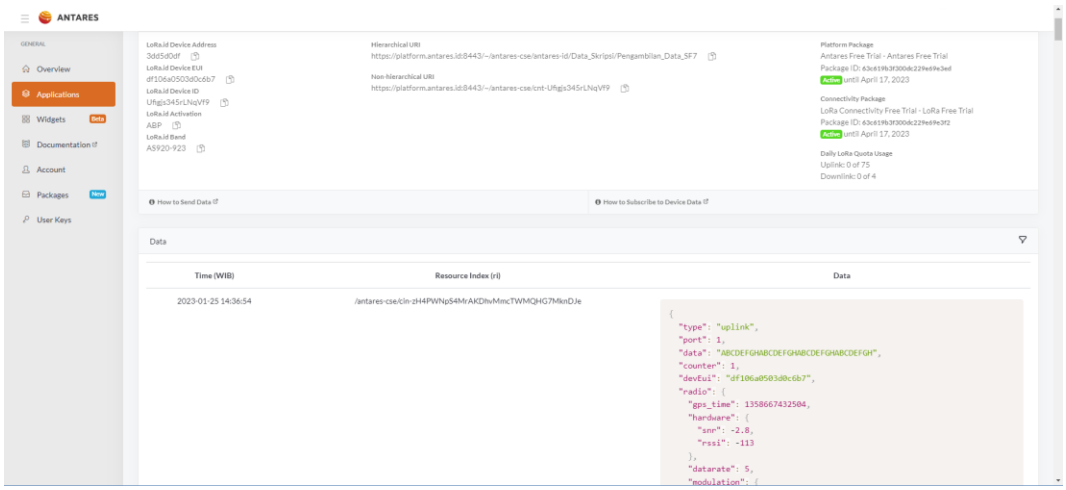
Gambar 2.21 *Hibernation Mode* [2]

Pengaturan mode daya pada ESP32 dapat dilakukan melalui perangkat lunak, seperti Arduino IDE atau ESP-IDF (*Espressif IoT Development Framework*). Pengaturan mode daya yang tepat dapat membantu mengoptimalkan penggunaan daya pada perangkat dan meningkatkan efisiensi energi.

2.2.7 Platform Antares

PT Telkom Indonesia (Persero) Tbk terus berinovasi dalam pengembangan platform *Internet of Things* (IoT) Indonesia. Ini juga dicapai melalui dua jalur, yaitu koneksi LoRaWAN dan platform IoT Antares. *LoRa* adalah solusi komunikasi IoT nirkabel yang memungkinkan komunikasi jarak jauh dan berdaya rendah. Konsumsi daya *LoRa* yang rendah disebabkan oleh paradigma komunikasi asinkronnya, yang menyiratkan bahwa sebuah node hanya akan berkomunikasi jika ada daya untuk dikirim. *LoRa* juga dapat digunakan untuk berbagai aplikasi IoT, seperti kota pintar, di mana sensor dapat berkomunikasi secara langsung. LoRaWAN adalah protokol jaringan untuk perangkat *LoRa* open source yang didukung oleh Aliansi *LoRa*. Teknik ini juga menghemat masa pakai baterai dan beroperasi pada frekuensi 920-923 MHz. Selain itu, LoRaWAN memenuhi kebutuhan IoT penting seperti komunikasi dua arah, keamanan end-to-end, mobilitas, dan layanan lokasi. Perangkat dan protokol LoRaWAN memungkinkan aplikasi IoT untuk mengatasi beberapa masalah kemanusiaan yang paling

mendesak, termasuk manajemen energi, manajemen sumber daya alam (SDA), pengendalian polusi, efisiensi infrastruktur, penghindaran bencana, dan banyak lagi. LoRaWAN secara demografis sangat mendukung pengembangan di Indonesia untuk memungkinkan IoT, dengan lebih dari 158 juta perangkat terhubung ke jaringan di 92 negara dan diperkirakan akan meningkat di masa mendatang. Telkom LoRaWAN dan Antares adalah dua teknologi yang saling berhubungan. Antares dengan sertifikasi internasional dari OneM2M dapat mendukung protokol internasional dan standar teknis untuk menawarkan interoperabilitas dan komunikasi yang lancar untuk semua perangkat yang mendukung IoT. Utilitas pintar, PJU pintar, pelacakan aset, pemantauan lingkungan, unggas pintar, dan lebih banyak aplikasi dapat dibuat menggunakan platform IoT Antares dan Telkom LoRaWAN. Konektivitas dalam Solusi Layanan IoT tidak terbatas pada LoRaWAN, tetapi juga dapat mencakup BLE, RFID, Wi-Fi, NB-IoT, GSM, satelit, dan teknologi lainnya. LoRaWAN Telkom juga telah terhubung ke hampir seluruh Indonesia, dengan 592 gateway *LoRa* yang dikerahkan dari Sabang hingga Merauke untuk mendukung IoT secara memadai di Indonesia [16].



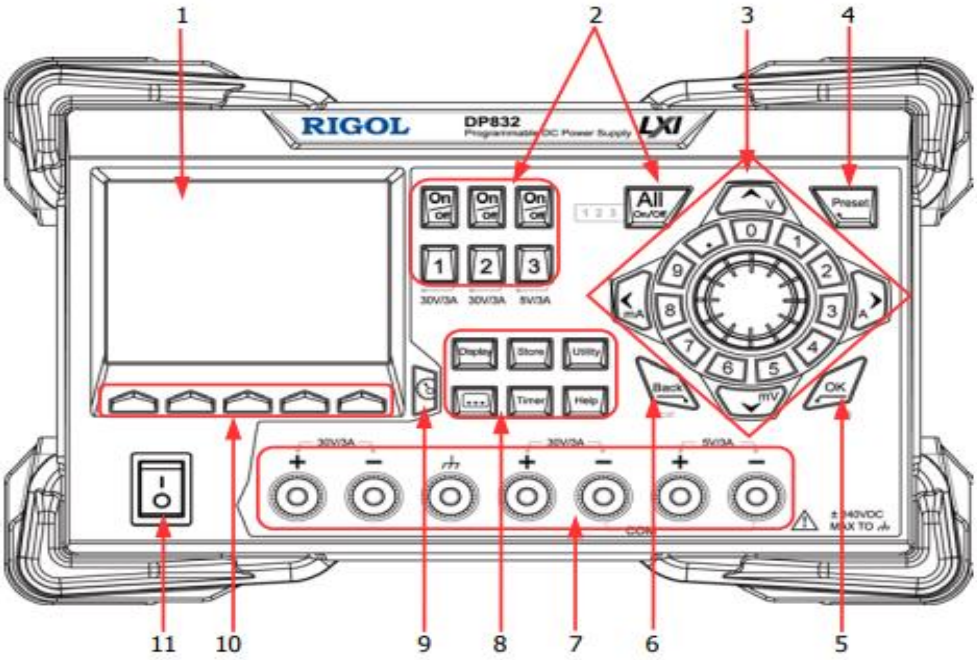
Gambar 2.22 Tampilan *Device* Antares

2.2.8 Catu Daya

Power supply atau sering dikenal dengan catu daya adalah perangkat listrik yang berfungsi sebagai sumber tenaga bagi perangkat lain. Secara umum, catu daya adalah sistem penyearah filter yang mengubah arus bolak-balik (AC) menjadi arus

searah (DC). Meskipun sumber DC mungkin sering menggerakkan perangkat secara langsung, beberapa cara untuk mengatur dan mempertahankan ggl bahkan ketika beban berfluktuasi mungkin diperlukan. Energi yang disediakan adalah arus bolak-balik, yang harus diubah atau diperbaiki menjadi arus searah berdenyut sebelum disuling atau disaring menjadi tegangan konstan. Tegangan DC juga membutuhkan manajemen tegangan agar rangkaian dapat bekerja dengan baik. Sebagian besar sirkuit elektronik memerlukan sumber daya DC 5 sampai 30 volt yang khas. Dalam beberapa kasus, daya ini dapat diambil langsung dari baterai (misal 6 V, 9 V, 12 V), tetapi di banyak kasus lainnya, adaptor daya AC standar lebih disukai. Selain itu, sirkuit listrik tertentu hanya membutuhkan 5 *volt*. Gerbang logika dibuat menggunakan sirkuit terpadu. [17].

Catu daya adalah sirkuit yang mengubah arus searah dari arus bolak-balik. Ini biasanya digunakan dalam elektronik untuk menyalakan peralatan yang membutuhkan arus searah daripada arus bolak-balik. Penyearah arus AC ke DC menggunakan empat buah dioda sebagai penyearah, serta berbagai komponen pendukung seperti rangkaian terpadu (IC) pengatur tegangan, kapasitor, resistor, transistor, dan potensiometer [18].



Gambar 2.23 Bagian Panel RIGOL DP832 [19]

Tabel 2.4 Keterangan bagian dari gambar 2.18 [19]

Nomor	Bagian	Keterangan
1	*LCD*	Digunakan untuk menampilkan pengaturan parameter sistem, sistem status keluaran, opsi menu, pesan prompt, dll
2	*Channel (Range) Selection and Output Switch*	Untuk model multi-saluran, fungsiny adalah pemilihan saluran dan sakelar keluaran. Untuk model saluran tunggal, fungsinya adalah pemilihan jangkauan dan saklar keluaran.
3	*Parameter Input Area*	Area input parameter, mencakup tombol arah (tombol pemilihan unit), keyboard numerik, dan kenop.
4	*Preset*	Mengembalikan semua pengaturan instrumen ke nilai default atau kembali konfigurasi voltase/arus saluran yang ditentukan pengguna.
5	*OK*	Konfirmasikan pengaturan parameter.
6	*Back*	Hapus karakter saat ini sebelum kursor. Saat instrumen dalam mode jarak jauh, tekan tombol ini untuk kembali ke mode lokal.
7	*Output Terminals*	Mengeluarkan besar tegangan dan arus yang telah terkonfigurasi
8	*Function Menu Area*	Pemilihan Menu fungsi yang akan digunakan pada instrumet
9	*Display Mode Switch Key*	Beralih antara mode tampilan saat ini dan mode tampilan yang dipilih.
10	*Menu Keys*	Tombol menu sesuai dengan menu di atasnya. Tekan sembarang tombol menu untuk memilih menu yang sesuai.
11	*Power Switch Keys*	Hidupkan atau matikan instrumen.

2.2.8 Konsumsi Energi Listrik

Jumlah daya dan waktu pengoperasian, atau berapa banyak energi listrik yang digunakan dalam waktu tertentu, dikenal sebagai konsumsi energi listrik. Biaya yang kita bayarkan untuk listrik ditentukan oleh berapa banyak energi yang kita gunakan. Karena konsumsi daya dalam waktu bukan *watt* yang menentukan jumlah listrik yang dikonsumsi, kita mungkin merasa bahwa sedikit *watt* akan menghemat uang, meskipun faktanya tidak selalu demikian. Karena mereka perlu menemukan banyak informasi spesifik tentang suatu perangkat sebelum mereka dapat mengetahui berapa banyak energi yang digunakannya, hal ini mungkin sulit dipahami oleh konsumen sehari-hari. Akibatnya, produsen peralatan listrik harus diwajibkan untuk memasukkan data konsumsi energi. Perhitungan di bawah ini

(dengan asumsi bahwa kedua lemari es memiliki ukuran, kualitas, suhu, dan desain yang identik) berfungsi sebagai bukti [20].

Kulkas / Lemari Es 1 : Kapasitasnya 200 liter, dan kompresor mengkonsumsi 70 W. Misalkan lemari es 70 W bekerja selama 3.000 detik per jam, yaitu 70J/s x 3000s/h x 24h = 5.040.000 Joule setiap hari. Alhasil, penggunaan energi lemari es adalah 1 = 5 MJ per hari.

Kulkas / Lemari Es 2 : Kapasitasnya 200 liter, dan kompresor mengkonsumsi 100 W. Misalkan lemari es 2 adalah 100 W dan menyala 1.000 detik per jam, mengkonsumsi 100J/s x 1000s/h x 24h = 2.400.000 Joule per hari. Hasilnya, penggunaan energi kulkas 2 adalah 2,4 MJ per hari.

Ternyata lemari es dengan daya 100 watt lebih hemat energi 50% daripada lemari es dengan daya 70 watt. Sepintas mungkin ada anggapan bahwa kulkas 70 *watt* lebih hemat energi daripada kulkas 100 *watt*. Oleh karena itu, kami dapat menunjukkan bahwa peringkat watt alat listrik memberikan sedikit wawasan tentang konsumsi energinya. Sejalan dengan itu, kita harus tahu berapa banyak konsumsi energi dalam perangkat keras listrik. TV dan lampu harus diberi label dalam MJ atau kJ per jam karena biasanya digunakan setiap jam. Karena lemari es biasanya digunakan terus menerus sepanjang hari atau bulan, lebih baik untuk menunjukkan jumlah MJ yang mereka konsumsi setiap hari atau bulan. Daya peralatan serta jumlah waktu pengoperasiannya memengaruhi berapa banyak energi listrik yang digunakan pada suatu beban. Semakin lama durasi waktu beroperasi, semakin banyak energi yang dikonsumsi, persamaannya adalah sebagai berikut:

$$W = P \times t \qquad \textbf{(2.4)}$$

Dimana :

W = Energi Listrik (*kWh*)

P = Daya Listrik (*Watt*) = V

t = Satuan Waktu (*Hour*)

Untuk dapat menghitung waktu pemakaian beban pada suatu perangkat yang hanya membutuhkan suplai energi melalui baterai atau penyimpanan energi berjenis *direct current power supply* dapat menggunakan persamaan berikut untuk menghitung waktu pemakaian beban dapat menggunakan persamaan 2.5 dibawah ini:

$$t = \frac{Kapasitas\ Baterai}{Daya\ yang\ digunakan} \quad \textbf{(2.5)}$$

Dimana :

t = Waktu pemakaian (*Hour*)

Kapasitas Baterai = Ukuran baterai dalam *miliampere* (*mA)*

Daya digunakan = $P = V \times I$ (*Watt)*