IMPLEMENTASI *HIGH AVAILABILITY* PADA *GATEWAY WIRELESS SENSOR NETWORK* DENGAN PROTOKOL KOMUNIKASI *MESSAGE QUEUING TELEMETRY TRANSPORT*

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Teknik

Disusun oleh:
Bagus Prasetyo
NIM: 145150300111022

PROGRAM STUDI TEKNIK KOMPUTER
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

IMPLEMENTASI *HIGH AVAILABILITY* PADA *GATEWAY WIRELESS SENSOR NETWORK* DENGAN PROTOKOL KOMUNIKASI *MESSAGE QUEUING TELEMETRY TRANSPORT*

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Teknik

Disusun Oleh:
Bagus Prasetyo
NIM: 145150300111022

Skripsi ini telah diuji dan dinyatakan lulus pada
18 Januari 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Sabriansyah Rizqika Akbar, S.T., M.Eng.	Widhi Yahya, S.Kom., M.Sc.
NIP: 19820809 201212 1 004	NIK: 201607 891121 1 001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T., M.T., Ph.D.
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 18 Januari 2018

Bagus Prasetyo

NIM: 145150300111022

KATA PENGANTAR

Puji syukur kehadirat Allah SWT yang telah melimpahkan rahmat, taufik dan hidayah-Nya sehingga laporan skripsi yang berjudul "Implementasi *High Availability* pada *Gateway Wireless Sensor Network* Dengan Protokol Komunikasi *Message Queuing Telemetry Transport*" ini dapat terselesaikan. Laporan skripsi ini disusun dalam rangka memenuhi persyaratan untuk memperoleh gelar Sarjana Komputer di Fakultas Ilmu Komputer

Penulis menyadari bahwa penyusunan laporan skripsi ini tidak lepas dari bantuan berbagai pihak baik secara langsung maupun tidak langsung. Dalam kesempatan ini penulis ingin menyampaikan ucapan terima kasih atas bantuannya kepada semua pihak, sehingga penulis dapat menyelesaikan laporan ini dengan baik. Ucapan terima kasih tersebut khususnya kepada:

1. Allah yang Maha Esa yang selalu memberikan petunjuk dan hikmah dalam penulisan ini.
2. Orang Tua dan Keluarga atas nasehat, kasih sayang serta dukungan materil dan moril. Terutama ayah tercinta karena telah bekerja keras untuk dapat membiayai anaknya kuliah yang biayanya tidak murah.
3. Sabriansyah Rizqika Akbar, S.T., M.Eng. dan Widhi Yahya S.Kom., M.Sc. selaku dosen pembimbing yang telah memberikan banyak dukungan dalam proses penyusunan skripsi ini baik teknis maupun non teknis.
4. Tri Astoto Kurniawan, S.T., M.T., Ph.D. selaku ketua jurusan Teknik Informatika.
5. Alifia Putri atas dukungan, doa, kesabaran, serta penyemangat dalam pengerjaan skripsi ini.
6. Teman-teman The-Ex yang tidak bisa disebutkan satu-satu atas dukungan serta kesediaan untuk menjadi teman diskusi selama pengerjaan skripsi.
7. Dan orang-orang yang selalu mendukung serta mendoakan kelancaran proses skripsi ini yang tidak bisa disebutkan satu per satu atas semua doa dan dukungannya.

Dengan segala keterbatasan pengetahuan yang dimiliki, penulis sadar bahwa penulisan laporan skripsi ini masih jauh dari kesempurnaan. Sehingga penulis berharap agar laporan skripsi ini dapat bermanfaat dan merupakan salah satu informasi yang berguna bagi pembaca.

Malang, 18 Januari 2018

Penulis

Bagustyo92@gmail.com

ABSTRAK

MQTT-SN merupakan pengembangan dari protokol yang banyak digunakan pada *Internet of Things* sekarang ini yaitu MQTT. Pada penerapanya MQTT-SN sama dengan MQTT, tetapi MQTT-SN difokuskan pada *wireless sensor network*. Pada MQTT-SN yang paling terlihat adalah hadirnya *gateway* sebagai kolektif data dari *sensor node* untuk diteruskan ke *broker server*. *Gateway* merupakan bagian terpenting pada MQTT-SN, karena *gateway* merupakan penghubung antara *sensor node* dengan *broker server*. Oleh sebab itu tingkat ketersediaan *gateway* haruslah tinggi untuk tetap dapat meneruskan data ke *broker* ketika terjadi gangguan. Penelitian ini memfokuskan untuk meningkatkan ketersediaan *gateway* dan *broker* MQTT dengan mengimplementasikan 3 buah Raspberry Pi sebagai *gateway* yang dipasangkan Load Balancer Haproxy, dan juga redudansi Keepalived. Dalam penelitian ini juga menggunakan multi *broker* yang saling berbagi pesan atau topik untuk meningkatkan ketersediaan *broker*. Hasil dari penelitian ini adalah *broker* dan *gateway* berhasil ditingkatkan ketersediaanya dengan klasterisasi, pembagian beban trafik, penggunaan multi *broker* dan juga redudansi pada sistem hingga 100% ketika salah satu *gateway* atau *broker server* mengalami gangguan.

Kata kunci: *High Availability,* Load Balancer, Redudansi*, Gateway,* Raspberry Pi*,* MQTT*,* Multi *Broker*

ABSTRACT

MQTT-SN is a development of the widely used protocol on the Internet of Things Technology, MQTT. In common MQTT-SN protocol is same as well as MQTT, but MQTT-SN is focused on the sector of wireless sensor network. In MQTT-SN the most prominent is the presence of gateway. Gateway is used to collective data from multiple sensor nodes and forwarding data to broker server. Gateway is the most important part of MQTT-SN technology, because gateway is communicator between sensor node and broker server. Therefor availability of gateway is crucial when facing failure-system or down. This research will be focus on increasing the availability of MQTT server brokers and gateway by implementing Haproxy Load Balancer and Keepalived that installed on 3 Raspberry Pi's as gateways. This system also used multiple broker servers which sharing message or topic for improve availability on broker server side. Research results is brokers and gateways successfully improved availability by using clusterization, balancer traffic, multiple broker implementation and redudancy up to 100% when one of the gateway or broker has a failure.

Keywords: High Availability, Load Balancer, *Redudancy, Gateway,* Raspberry Pi, MQTT*, Multiple Brokers*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1
PENDAHULUAN

1.1 Latar belakang

Wireless network merupakan sekumpulan komputer yang saling terhubung antara satu dengan yang lainnya sehingga terbentuk sebuah jaringan komputer dengan menggunakan media udara/gelombang sebagai jalur lintas datanya (Ridge, 2013). Seiring berkembangnya kebutuhan manusia akan komunikasi data yang sifatnya tanpa menggunakan kabel (*Wireless*). Mendorong manusia untuk mengembangkan teknologi terbaru. Kebutuhan akan komunikasi data yang sifatnya memiliki data *rate* yang rendah *(Low data Rate)*, biaya murah *(Low Cost)*, dan konsumsi daya yang rendah merupakan salah satu kebutuhan untuk komunikasi data dalam rangka melakukan suatu kontrol pada suatu jaringan. Teknologi *wireless* yang *familiar* digunakan oleh masyarakat seperti radio, bluetooth, maupun wifi sudah banyak diaplikasikan pada *smartphone*, *laptop*, dan beberapa *gadget* lainnya.

Modul ESP8266 merupakan modul *low cost* wifi yang didukung penuh untuk penggunaan TCP/IP ataupun UDP. ESP8266 dikembangkan oleh pengembang asal Tiongkok yaitu "Espressif". Produk ESP866 memiliki banyak varian. Modul wifi ini bersifat SoC (*Sytem on Chip*), sehingga bisa melakukan *programming* langsung ke ESP8266 tanpa memerlukan mikrokontroler tambahan untuk tipe-tipe tertentu. Modul ESP8266 juga menyediakan kemampuan tak tertandingi untuk menanamkan kemampuan wifi dalam sistem yang lain, atau berfungsi sebagai aplikasi *stand alone* dengan biaya yang rendah dan kebutuhan ruang yang minimal (Shobrina, et al., 2016).

Modul NRF24l01 adalah sebuah modul komunikasi jarak jauh yang memanfaatkan pita gelombang RG=F 2.4 GHz ISM (*Industrial, S-cientific and Medical*). Modul ini menggunakan antarmuka SPI untuk berkomunikasi dengan mikrokontroler. Tegangan yang bekerja pada modul yaitu 5V DC (nordicsemi, 2017).

Kedua modul komunikasi tersebut lebih banyak dipilih dalam proses belajar dalam pengiriman data secara *wireless* karena harganya yang murah dan mudah dijumpai di pasaran. Namun, penggunaan modul komunikasi *wireless* seperti ESP8266 dan NRF24L01 untuk media pembelajaran pengiriman data secara *wirelesss* bagi pemula yang menggunakan beberapa mikrokontroler diantaranya seperti Arduino, Nodemcu, dan masih banyak yang lain terkendala saat konfigurasi awal. Hal tersebut karena rumitnya konfigurasi awal yang dibutuhkan untuk membuat perangkat mikrokontroler terhubung secara *wireless*

menggunakan ESP8266 maupun NRF24L01. Hal itu menyebabkan ESP8266 dan NRF24L01 kurang populer di kalangan pemula *hardware.*

Kendala dalam konfigurasi awal penggunaan ESP8266 dan NRF24l01 di mikrokontroler bagi pemula *hadware* ini membuat penulis tertarik untuk melakukan penelitian mengenai "Rancang Bangun Pengenalan Modul Komunikasi dengan Konfigurasi Otomatis Berbasis UART ". Perangkat ESP8266 atau NRF24l01 akan dihubungkan ke *chip* ATmega328P (Arduino nano) atau *chip* ATtiny85 untuk menjadi modul komunikasi *Plug and Play* (PnP). Modul inilah yang nantinya akan terhubung ke mikrokontroler dan langsung terdeteksi serta dapat melakukan pengiriman data tanpa perlu konfigurasi awal lagi. *Chip* ATmega328P atau ATtiny85 yang akan menjadi jembatan dalam komunikasi serial *Universal Asincrhounus Receiver Transmitte* (UART) dari modul komunikasi ke mikrokontroler dengan konfigurasi otomatis atau sering disebut dengan istilah metode PnP.

1.2 Rumusan masalah

Berdasarkan latarbelakang yang telah dikemukakan, rumusan masalah pada penelitian ini antara lain:

1. Bagaimana membuat arsitektur modul komunikasi PnP dengan konfigurasi otomatis berbasis UART ?
2. Bagaimana mendeteksi sekaligus memilih modul komunikasi PnP ESP8266 atau NRF24L01 saat dihubungkan pada mikrokontroler ?
3. Bagaimana mengirimkan data menggunakan modul komunikasi ESP8266 atau NRF24L01 ketika sudah terhubung pada mikrokontroler ?

1.3 Tujuan

Dengan dirumuskannya masalah, didapatkan tujuan yang terkait pada rumusan masalah sebagai berikut :

1. Membuat arsitektur modul komunikasi PnP dengan konfigurasi otomatis berbasis UART.
2. Mendeteksi sekaligus memilih modul komunikasi ESP8266 atau NRF24L01 saat dihubungkan pada mikrokontroler.
3. Mengirimkan data sensor menggunakan modul komunikasi ESP8266 atau NRF24L01 ketika sudah terhubung pada mikrokontroler.

1.4 Manfaat

Melihat dari fungsi sistem yang akan diteliti serta diimpelemetasi, didapatkan manfaat sebagai berikut :

1. Dapat membantu para pemula *hadware* dalam mempelajari serta mengggunakan modul komunikasi *wireless* khususnya ESP8266 dan NRF24L01.

2. Memudahkan dalam komunikasi data bersifat data rate yang rendah *(Low data Rate) .*
3. Membuat pemula *hadware* tertarik dalam mempelajari serta memanfaatkan modul komunikasi *wireless* seperti ESP8266 dan NRF24L01.
4. Kebanggaan bagi diri sendiri dapat menciptakan modul/sistem yang mampu membantu orang dalam proses belajar modul komunikasi *wireless* khususnya ESP8266 dan NRF24l01.

1.5 Batasan masalah

Agar pembahasan dalam penelitian ini dapat dilakukan secara terarah dan mendapatkan hasil sesuai dengan yang diharapkan, maka perlu diterapkan batasan permasalahan yaitu:

1. Sistem hanya diterapkan pada mikrokontroler berbasis ATmega328P Seperti Arduino Uno dan Nano.
2. Modul komunikasi yang digunakan pada penelitian ini yaitu mengunakan ESP8266 seri E-01 dan NRF24l01.
3. *Chip* yang digunakan untuk menghubungkan antara modul komunikasi dan mikrokontroler hanya menggunakan ATtiny85 dan ATmega328P (Arduino nano).
4. Pengiriman data untuk komunikasi *wireless* ESP8266 hanya menggunakan protokol MQTT dan *library* PubSubClient.
5. Pengiriman data untuk komunikasi *wireless* radio NRF24l01 menggunakan *library* RF24.

1.6 Sistematika pembahasan

Sistematika penulisan dalam skripsi ini sebagai berikut:

BAB I Pendahuluan

Membahas tentang latar belakang, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah, dan sistematika pembahasan.

BAB II Landasan Kepustakaan

Membahas tentang tinjauan pustaka dari penelitian sebelumnya untuk memperkuat penelitian ini dan dasar teori yang mendukung dalam pembuatan sistem ini.

BAB III Metodologi Penelitian

Membahas tentang langkah-langkah dalam melakukan penelitian ini, antara lain studi literatur, pengumpulan data, analisis kebutuhan sistem, perancangan sistem, implementasi sistem, pengujian, dan evaluasi sistem.

BAB IV Rekayasa Kebutuhan

Menjelaskan secara keseluruhan kebetuhan sistem yang diperlukan meliputi kebutuhan perangkat keras, kebutuhan perangkat lunak, kebutuhan fungsional serta non fungsional.

BAB V Perancangan dan Implementasi

Membahas tentang proses implementasi sistem dalam mendeteksi modul komunikasi ESP8266 atau NRF24l01 saat dihubungkan dengan mikrokontroler kemudian melakukan pengiriman data secara *wireless*. Proses dimulai dengan membuat arsitektur sistem berupa modul komunikasi Atmega328P (Arduino nano) dengan ESP8266 maupun ATtiny85 dengan NRF24L01, kemudian mendeteksi modul komunikasi saat dihubungkan dengan *board* Arduino Uno (*Core* modul) , melakukan pengiriman data melalui protokol MQTT dan melihat hasil pengiriman data dummy pada aplikasi MQTT *client* untuk modul komunikasi PnP ESP8266. Sedangkan untuk modul komunikasi PnP NRF24l01 menggunakan *library* RF24 untuk melakukan pengiriman data.

BAB VI Pengujian

Membahas tentang cara pengujian serta akurasi hasil sistem dalam mendeteksi modul komunikasi ESP8266 atau NRF24l01 saat dihubungkan ke mikrokontroler Arduino serta ketepatan data yang dikirimkan oleh modul komunikasi.

BAB VII Penutup

Bab ini membahas kesimpulan yang diperoleh dari perancangan, implementasi dan pengujian sistem, serta saran-saran untuk pengembangan penelitian serupa mapun melanjutkan yang sudah ada.

BAB 2
LANDASAN KEPUSTAKAAN

2.1 Tinjauan Pustaka

Berikut merupakan penelitian yang terkait dengan sistem *Plug and Play* pada perangkat mikrokontroler serta penggunaan ESP8266 maupun NRF24l01 untuk pengiriman data secara *wireless*.

Tabel 2.1 Penelitian Terkait

NO	Nama Penulis (Tahun), Judul Penelitian	Persamaan	Perbedaan	
			Penelitian Terdahulu	Rencana Penelitian
1	Alvin Leonardo (2017) Impelementasi *Smart Light Bulb Pervasive* berbasis ESP8266	Penggunaan modul komunikasi yaitu yaitu tipe ESP8266 dan adanya sistem konfigurasi otomatis untuk pengenalan perangkat yang terhubung.	Pengguna dapat mengunakan *Smart Light Bulb* secara otomatis tanpa melakukan konfgurasi awal berbasis ESP8266.	Tidak membuat perangkat *Smart Light Bulb*, namun lebih terfokus pada konfigurasi otomatis dan pengiriman data menggunakan modul komunikasi ESP8266 saat dihubungkan dengan mikrokontroler Arduino Uno
2	Suprayogi, dkk (2016),Implementasi *Pervasive Service Discovery Protocol* pada Rumah Cerdas Berbasis NRF24L01	Penggunaan modul komunikasi *wireless* yaitu NRF24l01 serta terdapat konfigurasi otomatis untuk pengenalan	Sistem yang mampu mengidentifikasi suatu layanan dan perangkat baru pada suatu jaringan sensor yang berbasis *wireless* radio	Sistem tidak mampu mendeteksi suatu layanan seperti sensor namun berfokus pada deteksi dan pengiriman data modul komunikasi

NO	Nama Penulis (Tahun), Judul Penelitian	Persamaan	Perbedaan	
			Penelitian Terdahulu	Rencana Penelitian
		layanan dan perangkat yang baru terhubung .	menggunakan NRF24L01.	NRF24l01 saat terhubung dengan mikrokontroler Arduino Uno

2.2 Dasar Teori

Subbab dasar teori membahas teori pendukung maupun yang diperlukan untuk menyusun penelitian yang diusulkan.

2.2.1 Komunikasi *Wireless*

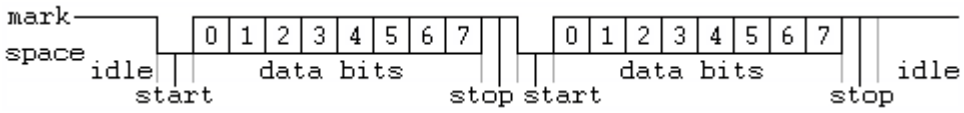
Wireless merupakan teknologi komunikasi yang menggunakan gelombang elektromagnet untuk mengirimkan sinyal dengan jarak yang dekat maupun jarak yang jauh. Dengan menggunakan *wireless* maka komunikasi antar komputer bisa dilakukan tanpa media penguhubung fisik seperti kabel tembaga atau kabel Fiber Optik. Jaringan komunikasi *wireless* merupakan teknologi alternatif yang bersifat lebih modern guna melakukan komunikasi dibandingkan dengan komunikasi menggunakan media fisik.

Jaringan berbasis *wireless* mempunyai kelebihan dan kekurangan sebagaimana jaringan komunikasi yang menggunakan kabel. Adapun kelebihan yang paling mendasar dari sebuah jaringan *wireless* adalah sifat mobilitasnya yang tinggi dan terhindar dari belitan kabel yang terkadang membuat sebagian orang merasa terganggu. Namun disamping kelebihan tersebut terdapat pula kekurangan, diantara kekurangan jaringan *wireless* adalah rentannya gangguan gelombang apabila mendapati cuaca buruk, rentan interferensi dari gelombang radio yang dipancarkan oleh perangkat lain, serta harus benar-benar dipastikan tidak ada hambatan seperti tembok, bukti, dan pohon yang dapat mengurangi kualitas dari pancaran gelombang *wireless* (Mandalamaya,2015).

Jaringan *wireless* yang sangat populer saat ini dan banyak digunakan adalah jaringan *Wireless Local Area Network* (WLAN) yang distandarisasi oleh IEEE, IEEE singkatan dari *Institute of Electrical and Electronic Engineers* . Namun ada juga komunikasi *wireless* yang menggunakan sinyal Radio seperti NRF. Dari banyaknya perangkat komunikasi *wireless* yang digunakan, ada 2 yang paling sering digunakan untuk membentuk sebuah perangkat IoT. Namun memerlukan konfigurasi yang sedikit sulit, sehingga tidak memudahkan bagi para pemula yang ingin belajar tentang perangkat untuk komunikasi *wireless*.

2.2.2 Komunikasi Universal Asincrhounus *Receiver* Transmitter (UART)

Komunikasi *Universal Asincrhounus Receiver Transmitter* (UART) adalah mode komunikasi serial secara *asynchronous*, dimana data dikirim secara satu persatu melalui kabel ganda (twisted pair Tx & Rx). Pada gambar 2.1 merupakan proses pembacaan data pada komunikasi UART.



Gambar 2.1 Proses pembacaan komunikasi UART
Sumber : (http://www.aisi555.com, 2017)

Pada gambar 2.1 terdapat "Bit Start" yang ditambahkan pada setiap awal data yang akan ditransmisikan. Bit Start digunakan untuk memperingatkan penerima bahwa data akan segera dikirim, dan memaksa bit-bit sinyal di receiver agar sinkron dengan bit-bit sinyal di pemancar. Sedangkan "Bit Stop" sebagai penanda akhir dari pengiriman tiap bit data.

Komunikasi UART memiliki kecepatan transfer data yg ditentukan oleh pihak pengirim & penerima yg dinamakan *Baudrate*. UART memiliki level TTL 5 volt, dan menggunakan 2 buah I/O port (Tx & Rx). Sebuah UART biasanya berisi komponen-komponen berikut (Yudi,2013):

1. *Generator clock* : biasanya kelipatan dari *bit rate* untuk memungkinkan pengambilan sampel di tengah periode bit.
2. *Register Input Output*
3. Kontrol kirim (TX) dan terima (RX)
4. Kontrol logika *read write*
5. *Buffer* I/O (opsional)
6. *Paralel Bus Buffer* (opsional)
7. *Buffer First In First Out* (FIFO) (opsional)

2.2.3 ESP8266

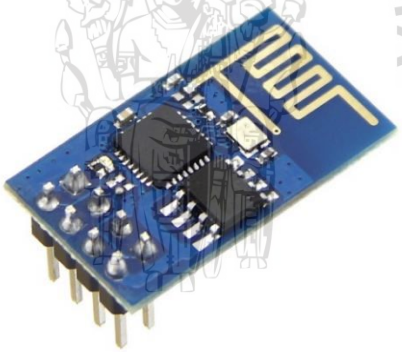
Modul ESP8266 merupakan modul *low cost* wifi yang didukung penuh untuk penggunaan TCP/IP ataupun UDP. ESP8266 dikembangkan oleh pengembang asal tiongkok yaitu "Espressif". Produk ESP866 memiliki banyak varian diantarnya type ESP-01,07, dan 12. Pada penelitian ini digunakan ESP8266 seri ESP-01. Modul ESP8266 juga menyediakan kemampuan tak tertandingi untuk menanamkan kemampuan wifi dalam sistem yang lain, atau berfungsi sebagai aplikasi *stand alone* dengan biaya yang rendah dan kebutuhan ruang yang minimal. (Shobrina, et al., 2016). Kelebihan lain ESP8266 adalah memilki *deep sleep mode*, sehingga penggunaan daya akan relatif jauh lebih efisien

dibandingkan dengan modul WiFI lain. Pada tabel 2.2 merupakan spesifikasi yang dimiliki oleh ESP8266 dan pada gambar 2.2 merupakan tampilan dari modul ESP8266 tipe E-01

Tabel 2.2 Spesifikasi ESP8266

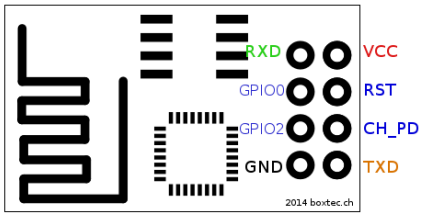
Wifi-protokol	802.11 b/g/n
Wifi-mode	WiFi Direct (P2P / Point-to-Point), Soft-AP / Access Point
Protokol Jaringan	TCP/IP
Peripheral	SDIO 2.0, SPI, UART STBC, 1x1 MIMO, 2x1 MIMO
Wifi-Enkripsi	WEP, TKIP, AES, dan WAPI
Tegangan kerja	3.3v – 5 v
Arus kerja	100 – 170 Ma (Typcial Mode)
Mode sleep to *Delay*	2ms
Power Amplifier	24 dBm

Sumber : Diadaptasi dari (www.seedstudio.com)



Gambar 2.2 ESP8266
Sumber : (SeeedStudio, 2017)

Berikut adalah gambaran dan bagian pin-pin dari modul komunikasi ESP8266 E-01 yang tediri dari 8 pada gambar 2.3 yaitu :



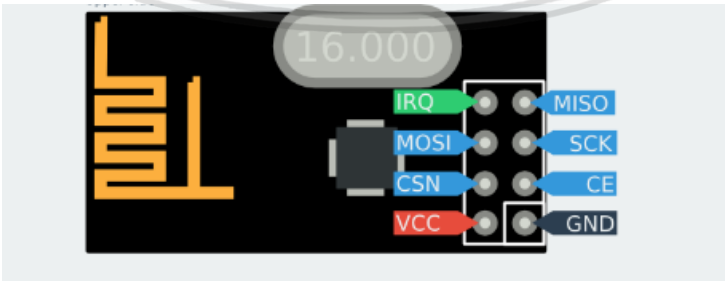
Gambar 2.3 *Pinout* ESP8266
Sumber : (http://fab.cba.mit.edu, 2017)

2.2.4 NRF24L01

Wireless Module NRF24L01 adalah sebuah modul komunikasi jarak jauh yang memanfaatkan pita gelombang RF 2.4GHz ISM (*Industrial, Scientific and Medical*). Modul ini menggunakan antarmuka SPI untuk berkomunikasi. Tegangan kerja dari modul ini adalah 5V DC. NRF24L01 memiliki *baseband logic Enhfanced ShockBurst™ hardware protocol accelerator* yang *support "high-speed* SPI *interface for the application controller*". NRF24L01 memiliki *true ULP solution*, yang memungkinkan daya tahan baterai berbulan-bulan hingga bertahun-tahun. Modul ini dapat digunakan untuk pembuatan *pheriperal* PC, piranti permainan, piranti fitnes dan olahraga, mainan anak-anak dan alat lainnya (Mikrocontroller,2014). Modul ini memiliki 8 buah pin seperti pada gambar 2.4 dibawah ini :

Gambar 2.4 *Wireless module* NRF24l01
Sumber: (www.dx.com, 2017)

Berikut adalah gambaran dan bagian pin-pin dari modul komunikasi NRF24l01 yang tediri dari 8 pin pada gambar 2.5 :



Gambar 2.5 Konfigurasi Pin NRF24L01
Sumber: (www.sunrom.com, 2017)

Spesifikasi lain yang dimiliki oleh modul NRF24L01 dijelaskan pada tabel 2.3 berikut :

Tabel 2.3 Spesfikasi Modul NRF24L01

Power supply	1.9V~3.6V
I/O port working voltage	0~3.3, 5 V
I/O port working current	13.5mA at 2Mbps
Data rate	256kbps / 1Mbps / 2Mbps
Receiving sensitivity	-85dBm at 1Mbps
Transmission range	70~100 meter at 256kbps
Temperatures	Operating:-40°C ~ 85°C / Storage:-40°C ~ 125°C

Sumber: (www.dx.com, 2016)

NRF24L01 memiliki beberapa mode yang dapat diimplementasikan untuk mengatur proses pengiriman dan penerimaan sinyal diantaranya :

1. Mode *Power Down*

   Mode ini membuat NRF24L01 mematikan sistem dengan cara mengkonsumsi arus/daya seminimal mungkin dan bit PWR_UP pada *register config* dimatikan untuk mengaktifkan mode ini.

2. Mode *Standby*

   Mode ini membuat NRF24L01 berada dalam keadaan *standby* yang berarti mode ini meminimalisir penggunaan rata-rata arus pada saat memulai proses *start up*.

3. Mode TX (*Transfer*)

   Mode TX adalah mode yang aktif saat radio NRF24L01 berperan sebagai pengirim (Transfer). NRF24L01 mengaktifkan PWR_UP pada mode *high* dan PRIM_RX pada mode *low*, payload pada pin TX FIFO dan set high pulse dan pin CE lebih dari 10µs. Langkah berikutnya adalah menunggu hingga pin TX fifo kosong , jika tx FIFO masih ada muatan maka pin TX akan melakukan standby 2 dan bersiap untuk melakukan pengiriman selanjutnya.

4. Mode RX (*Receiever*)

   Mode RX adalah mode yang aktif ketika NRF24L01 berperan sebagai penerima. Untuk melakukan proses receive maka PWR_UP harus aktif, bit PRIM_RX dan pin CE harus sudah dalam keadaan aktif. Jika sebuah paket ditemukan maka isi paket tersebut akan dipindahkan ke dalam RX FIFO (P,et al.,2016)

2.2.5 Mikrokontroler

Mikrokontroler adalah sebuah sistem mikroprosesor dimana didalamnya sudah terdapat CPU, Read Only Memory (ROM), Random Accsess Memory (RAM), Input-Output, timer, interrupt, Clock dan peralatan internal lainnya yang sudah saling terhubung dan terorganisasi dengan baik dalam satu chip yang siap dipakai. Berikut adalah tampilah dari microchip Atmega328P pada gambar 2.6 :

Gambar 2.6 ATmega328P
Sumber : (Micro*chip*, 2017)

ATmega328P memberikan beberapa fitur diantaranya 8 Kb system programmable flash dengan kemampuan *read while write*, 1 KB EEPROM, 2 KB SRAM, 8 Kb *system programmable flash* dengan kemampuan *read while write*, 23 *general purpose* I/O, 32 register serba guna, 3 buah *timer/counter*, *Interrupt internal* maupun *eksternal*, serial untuk pemograman dengan menggunakan USART, *peripheral interface* (SPI), *two wire interface* (I2C), 6 port PWM (*Pulse Width Modulation*), 6 port 10 bit ADC dan *Watchdog Timer* dengan *osilator internal* (Sargent,2017). Pada penelitian ini menggunakan mikrokontroler berbasis ATmega328P yaitu Arduino versi Uno R3. Pada tabel 2.4 berikut merupakan spesifikasi Arduino Uno R3.

Tabel 2.4 Spesifikasi Arduino UNO R3

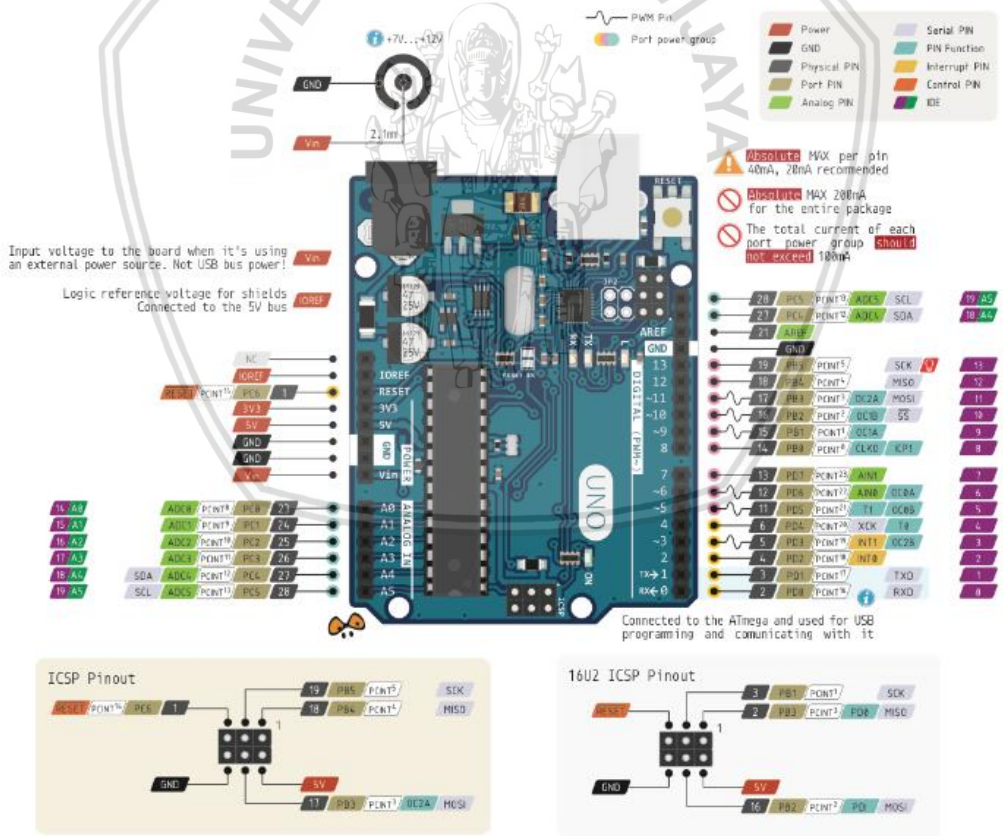
Mikrokontroler	ATmega328P
Tegangan kerja	5v
Tegangan masukan	7-12v
Pin digital I/O	14
Pin PWM I/O	6
Pin Analog I/O	6
Arus DC per pin I/O	20 mA
Arus DC pin 3.3v I/O	50 mA
Flash Memori	32 KB (ATmega328P)
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz

Sumber : (Arduino, 2017)

Berikut adalah tampilan dan bentuk dari mikrokontroler Arduino Uno yang berbasis ATmega328P pada gambar 2.7 :

Gambar 2.7 Arduino Uno Rev3
Sumber : (www.Arduino.cc)

Berikut adalah gambaran dan bagian pin-pin dari Mikrokontroller Arduino Uno serta fungsi beberapa komponen yang terdapat pada board ini pada gambar 2.8 :



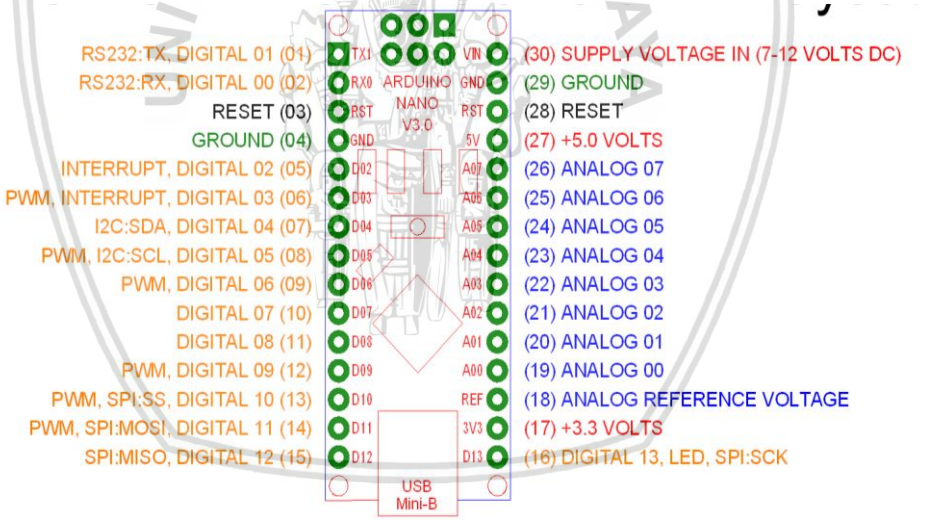
Gambar 2.8 *Pinout* Arduino Uno Rev3
Sumber : (www.Arduino.cc)

2.2.6 Arduino Nano

Arduino nano merupakan salah satu mikrokontroler yang sering dijumpai dipasaran karena harganya yang relatif murah dan mudah untuk digunakan berbasis ATmega328P. Arduino Nano versi 3.0 merupakan bentuk pengembangan Arduino nano sebelumnya (Ecadio,2017) . Arduino nano memiliki *body* yang relatif kecil jenis *breadboard-friendly board* dengan mikroprosesor ATmega328. Bentuk *board* dari Arduino nano dapaat dilihat pada gambar 2.9 berikut :

Gambar 2.9 Arduino Nano v3.0
Sumber: (https://www.Arduino.cc)

Berikut adalah gambaran dan bagian pin-pin dari Arduino nano yang biasa digunakan dalam keperluan pembuatan suatu ide maupun sistem yang ada pada gambar 2.10 :



Gambar 2.10 Arduino Nano Pinout
Sumber: (https://www.Arduino.cc)

Spesifikasi yang dimiliki Arduino Nano v3.0 dapat dilihat pada tabel 2.5 berikut :

Tabel 2.5 Spesifikasi Arduino Nano v3.0

Microcontroller	Atmel ATmega328P
Operating Voltage	5 V
Input Voltage	7-12 V
Digital I/O Pins	14 (of which 6 provide PWM *output*)

Analog *Input* Pins	8
DC Current per I/O Pin	40 mA
Flash Memory	16 KB (ATmega168) or 32 KB (ATmega328) of which 2 KB used by bootloader
SRAM	1 KB (ATmega168) or 2 KB (ATmega328)
EEPROM	512 bytes (ATmega168) or 1 KB (ATmega328)
Clock Speed	16 MHz

Sumber : (https://www.Arduino.cc/)

2.2.7 ATtiny85

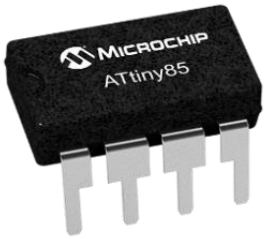
IC (integratot Circuit) ATtiny85 adalah *chip* mikrokontroler *high-performance*, *low-power* berbasis Atmel 8-bit AVR RISC yang memiliki 8KB ISP flash *memory*, 512B EEPROM, 512-Byte SRAM, 6 *general purpose* I/O, 32 *general purpose working registers*, 1 buah 8-bit *timer/counter* dengan *compare modes*, satu buah 8-bit *high speed timer/counter*, USI, internal dan external Interrupts, 4-channel 10-bit A/D converter, programmable *watchdog timer* dengan *internal oscillator*, 3 buah *software* seleksi mode hemat energi, dan debug *wire* untuk on-*chip* debugging. *Chip* ATTiny85 dapat mencapai throughput 20 MIPS pada 20 MHz dan beroperasi pada 2.7 - 5.5 volts (Microchip, 2017). Pada penelitian ini , *chip* ATtiny85 akan menjadi bagian dari hadware komunikasi wifi yang dikonfigurasi menjadi 1 sistem dengan ESP8266 ataupun NRF24L01 dan pada tabel 2.6 berikut adalah spesifikasi dari chip ATtiny85.

Tabel 2.6 Spesifikasi ATtiny85

Tipe Memori Program	Flash
Memori Program (KB)	8
Kecepeatan CPU (MIPS)	20
RAM Bytes	512
Data EEPROM(bytes)	512
Digital Peripheral	1-SPI, 1-I2C
PWM Peripheral	5PWM
Range Tegangan Operasi (V)	1.8 to 5.5
Jumlah pin	8

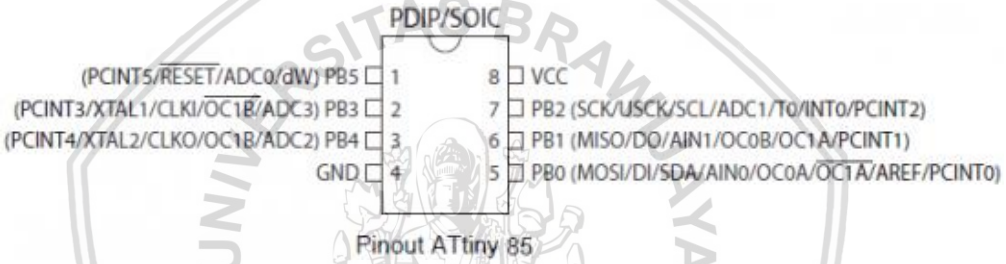
Sumber : (www.microchip.com)

Tampilan dan bentuk dari *Chip* ATtiny85 dapat dilihat pada gambar 2.11 berikut :



Gambar 2.11 *Chip* ATtiny85
Sumber : (www.microchip.com)

Berikut adalah gambaran dan bagian pin-pin dari *Chip* Attinny85 beserta fungsi -fungsi dari tiap pin yang ada pada gambar 2.12.

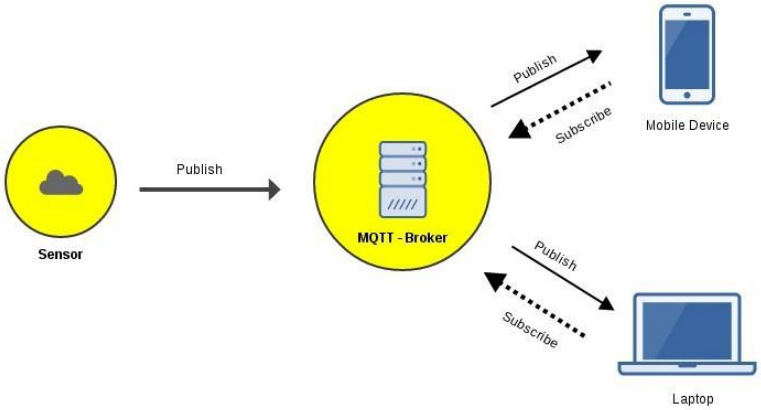


Gambar 2.12 *Pinout* ATtiny85
Sumber : (http://www.electroschematics.com)

2.2.8 MQTT

Message Queuing Telemetry Transport (MQTT) adalah protokol *transport* yang bersifat *client server publish/subscribe*. Protokol yang ringan, terbuka dan sederhana, dirancang agar mudah diimplementasikan. Karakteristik ini membuat MQTT dapat digunakan di banyak situasi, termasuk penggunaanya dalam komunikasi *machine-to-machine* (M2M) dan *Internet of Things* (IoT) sehingga memungkinkan setiap pengguna akan mendapatkan informasi secara *realtime*.. Protokol ini berjalan pada TCP/IP.

Protokol ini adalah jenis protokol *data-agnostic* yang artinya anda bisa mengirimkan data apapun seperti data binary, text bahkan XML ataupun JSON dan protokol ini memakai model **publish/subscribe** daripada model client-server. Pada gambar 2.13 ditunjukan sistem sederhana yang biasanya digunakan untuk mengimplementasikan protokol MQTT, yakni *subscriber* hanya menerima informasi dengan berlangganan pada suatu topik, kemudian *publisher* mengirimkan informasi ke suatu topik, selama *subscriber* dan *publisher* berlangganan topik yang sama maka komunikasi akan selalu terjalin (Equan,2015).



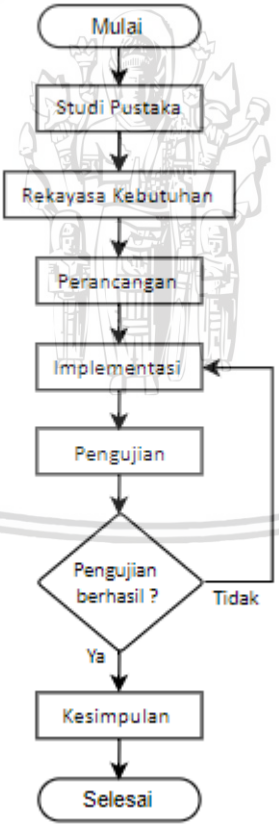
Gambar 2.13 Mekanisme kerja protokol MQTT

Sumber : (Jsiot, 2015)

Keuntungan dari sistem publish/subscribe adalah antara sumber pengirim data (publisher) dan penerima data (klien) tidak saling mengetahui karena ada broker diantara mereka atau istilah kerennya yaitu *space decoupling*dan yang lebih penting lagi yaitu adanya *time decoupling* dimana publisher dan klien tidak perlu terkoneksi secara bersamaan, misalnya klien bisa saja disconnect setelah melakukan subscribe ke broker dan beberapa saat kemudian klien connect kembali ke broker dan klien tetap akan menerima data yang terpending sebelumnya proses ini dikenal dengan mode offline**.**

BAB 3
METODOLOGI PENELITIAN

Bab ini menjelaskan metode yang digunakan dalam melakukan penelitian tentang Rancang Bangun Pengenalan Modul Komunikasi dengan Konfigurasi Otomatis Berbasis UART .Tipe penelitian yang dilakukan adalah implementatif. Implementatif yaitu bersifat observasi dengan menggunakan rancang bangun yang sederhana. Langkah pertama yang dilakukan adalah mengumpulkan teori-teori pendukung penelitian dan dikemas dalam studi literatur. Kemudian dilanjutkan dengan proses analisis kebutuhan. Setelah tahap analisis kebutuhan, proses selanjutnya adalah melakukan proses perancangan yang dilanjutkan dengan proses implementasi hardware serta implementasi software sesuai dengan perancangan. Tahap berikutnya dilakukan pengujian dan analisis pada rancangan yang telah dibangun. Kesimpulan dan saran disertakan sebagai catatan atas rancangan dan kemungkinan arah pengembangan selanjutnya. Langkah-langkah yang dilakukan dalam penelitian ini ditunjukkan pada gambar 3.1 berikut.



Gambar 3.1 Diagram alir metode penelitian

3.1 Studi Pustaka

Studi Literatur adalah tahap awal yang digunakan untuk menambah studi pustaka dan pengetahuan yang mendukung untuk mengerjakan penulisan laporan

dan penelitian. Studi literatur dilaksanakan dengan cara mengumpulkan teori dan pustaka yang berkaitan dengan penelitian ini meliputi :

1. Komunikasi *Wireless*

   Membahas tekait pengertian komunikasi *wireless,* kelebihan dan kekurangan dari komunikasi *wireless,* serta jenis-jenis komunikasi *wireless* yang sering digunakan saat ini.

2. Komunikasi Universal Asincrhounus *Receiver* Transmitter (UART)

   Membahas terkait pengertian komunikasi Serial, spesifikasi komunikasi serial serta komponen-komponen yang terdapat pada komunikasi Serial (UART). Komunikasi Serial inilah yang akan menjadi jembatan penghubung antara modul komunikasi PnP dengan *core* modul (Mikrokontroler Arduino).

3. ESP8266

   Membahas terkait pengertian ESP266 dan tipe ESP-01, spesifikasi ESP8266 ,kelebihan maupun konfigurasi pin yang akan dihubungkan dengan mikrokontroler.

4. NRF24l01

   Membahas terkait pengertian NRF24l01, spesifikasi NRF24l01, kelebihan maupun konfigurasi pin yang akan dihubungkan dengan mikrokontroler. Terdapat juga mode-mode yang bisa diaktifkan saat pengggunaan NRF24l01 yaitu mode *power down*, *stand by* ,RX , dan TX.

5. Mikrokontroler

   Membahas terkait pengertian dan spesifikasi mikorokontroler berbasis ATmega328 yaitu Arduino Uno V3.Terdapat pin out serta fungsi dari pin-pin yang terdapat pada Arduino Uno. Arduino Uno inilah yang akan menjadi *core* modul pendeteksi modul komunikasi PnP saat dihubungkan.

6. Arduino Nano

   Membahas terkait spesifikasi Arduino nano dan kelebihan dibanding mikrokontroler lain yang berbasis ATmega328P dengan bentuk yang mini sangat memungkinkan untuk dijadikan sebagai otak pengontrol modul komunikasi PnP yang terhubung dengan ESP8266 .

7. ATtiny85

   Membahas terkait pengertian maupun spesifikasi *chip* Attiny85 beserta fungsi-fungsi pin *out* yang akan terhubung dengan modul komunikasi NRF24l01. Attiny85 akan menjadi otak pengontrol dari modul komunikasi PnP berbasis NRF24l01.

3.2 Rekayasa Kebutuhan

Analisis kebutuhan ditujukan untuk menganalisis mengenai kebutuhan yang akan mendukung dalam proses penelitian ini. Analisi kebutuhan pada penelitian terdiri dari kebutuhan fungsional yang berisi tentang kebutuhan sistem, kebutuhan perangkat keras maupun perangkat lunak. Serta terdapat kebutuhan non-fungsional yang akan dibahas pada bab selenjutnya. Berikut adalah kebutuhan sistem pada penelitian ini :

1. Modul komunikasi PnP yang berbasis ESP8266 atau NRF24l01 dapat terdeteksi oleh mikrokontroler c*ore* modul (Arduino Uno).
2. Dapat memilih modul komunikasi PnP yang berbasis ESP8266 atau NRF24l01 yang terdeteksi oleh *core* modul.
3. Modul komunikasi PnP berbasis ESP8266 atau NRF24l01 dapat melakukan pengiriman data yang sudah tersimpan pada core modul sebelumnya.

3.2.1 Kebutuhan Perangkat Keras

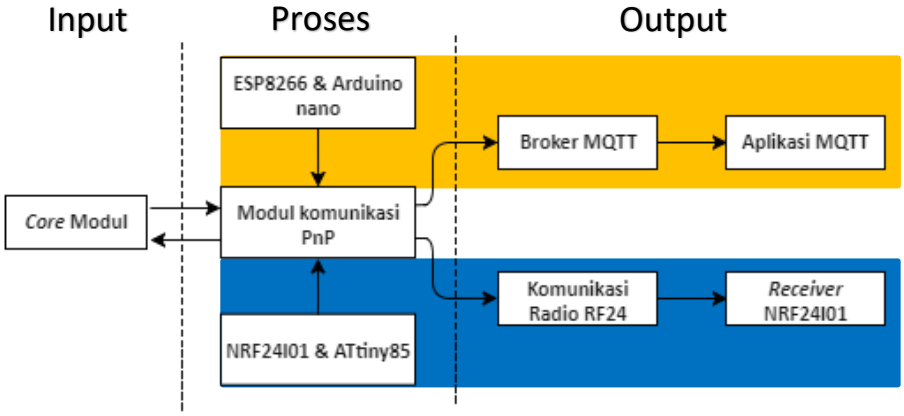
1. Mikrokontroler Arduino Uno
2. Modul Wifi ESP8266
3. Modul Radio NRF24l01
4. *Chip* ATmega328P (Arduino nano)
5. *Chip* ATtiny85

3.2.2 Kebutuhan Perangkat Lunak

1. Arduino IDE
2. Aplikasi MQTT
3. *Library Software Serial*
4. *Library* RF24
5. *Library* PubSubClient

3.3 Perancangan Sistem

Perancangan sistem pada penelitian ini mendefinisikan sistem apa yang akan peneliti bangun. Adapun diagram blok sistem yang akan dibuat seperti pada gambar 3.1 berikut :



Gambar 3.2 Blok diagram sistem

Keterangan :

: Proses modul komunikasi PnP ESP8266

: Proses modul komunikasi PnP NRF24l01

Berdasarkan Gambar 3.2 perancangan sistem dapat dijelaskan pada poin-poin berikut ini :

1. Modul wifi ESP8266 dengan Arduino nano ataupun modul NRF24L01 dengan *chip* ATtiny85 digabungkan dalam 1 minimum sistem menjadi suatu modul komunikasi *wirelesss* yang bersifat *Plug and Play* (PnP). Lalu saat modul komunikasi tersebut dihubungkan ke *core* modul dengan komunikasi serial (UART) , *core* modul dapat mendeteksi modul komunikasi tersebut dengan respon tertentu. Setelah terdeteksi, *core* modul dapat memilih modul terdeteksi untuk melakukan pengiriman data yang sudah tersimpan pada core modul sebelumnya.
2. Untuk modul komunikasi PnP ESP8266, menggunakan protokol MQTT yang terdapat pada *library* PubSubClient untuk mengirimkan data yang telah ditampung oleh core modul (Arduino Uno) melalui modul komunikasi PnP yang telah terdeteksi oleh *core* modul. Protokol yang digunakan untuk pengiriman data yaitu MQTT menggunakan Broker MQTT. Data yang telah dikirimkan oleh modul komunikasi PnP ESP8266 dapat dilihat melalui aplikasi MQTT yang telah ada dengan mengatur nama *server*, *port* dan *topic* yang digunakan sebelumnya pada modul komunikasi PnP ESP8266.
3. Untuk modul Komunikasi PnP NR424L01, *library* RF24 digunakan untuk dapat menggunakan fungsi pengiriman *wireless* dari modul NRF24L01. Untuk dapat melihat apakah data yang dikirimkan berhasil,maka diperlukan *receiver* komunikasi NRF24l01 yang terdiri dari NRF24L01 dan mikrokontroler Arduino Uno.

3.4 Implementasi

Tahap implementasi sistem akan dilakukkan berdasarkan perancangan yang telah ditentukan sebelumnya. Pada tahap ini akan diimplementasikan seluruh gagasan serta ide baik desain maupun hitungan sistem yang telah menjadi tujuan sebelumnya. Terdapat beberapa tahap implementasi yaitu :

1. Implementasi arsitektur modul komunikasi PnP menggunakan Modul ESP8266 dengan Arduino nano maupun modul NRF24L01 dengan ATtiny85.
2. Implementasi pengenalan dan pemilihan modul komunikasi PnP saat dihubungkan dengan *core* modul menggunakan komunikasi serial (UART).
3. Implementasi pengiriman data untuk modul komunikasi PnP ESP8266 dari *core* modul menggunakan protokol MQTT. Data yang telah berhasil dikirim oleh modul komunikasi dapat diakses melalui aplikasi MQTT .
4. Implementasi pengiriman data untuk modul komunikasi PnP NRF24l01 dari *core* modul menggunakan *library* RF24 untuk mengirimkan data secara *wireless*. Dan menyediakan *receiver* NRF24l01 untuk menampilkan hasil pengiriman data yang dilakukan.

3.5 Pengujian dan Analisis

Tahapan pengujian dan analisis dilakukan untuk mengetahui kesesuaian sistem setelah proses implementasi yang dibandingkan dengan spesifikasi sistem pada saat tahap perancangan. Proses pengujian yang dilakukan yakni sebagai berikut :

1. Pengujian deteksi modul komunikasi saat terhubung dengan *core* modul menggunakan komunikasi serial (UART)..
2. Pengujian pemilihan modul komunikasi PnP yang terdeteksi saat terhubung dengan *core* modul.
3. Pengujian pengiriman data modul komunikasi PnP ESP8266 menggunakan protokol MQTT dan data terkirim dapat diakses melalui aplikasi MQTT sebanyak 21 kali pengiriman data.
4. Pengujian pengiriman data modul komunikasi PnP NRF24l01 menggunakan *library* RF24 dan data terkirim dapat dilihat melalui *receiver* NRF24l01 sebanyak 21 kali pengiriman data.

3.6 Kesimpulan

Kesimpulan didapatkan setelah melakukan tahap perancangan, implementasi, pengujian dan analisis terhadap sistem yang dibuat. Kesimpulan disusun berdasarkan hasil dari tahap pengujian dan analisis yang dilakukan pada sistem yang dibuat. Isi pada kesimpulan diharapkan dapat menjadi acuan dasar pada penelitian selanjutnya atau pun peneltian sejenis. Di akhir penulisan terdapat saran yang bertujuan untuk memberikan kemudahan penelitian selanjutnya, apabila akan meneruskan dan mengembangkan penelitian ini.

BAB 4
REKAYASA KEBUTUHAN

Pada bab ini akan dijelaskan secara rinci terkait gambaran umum sistem, kebutuhan fungsional yang terdiri dari kebutuhan sistem, kebutuhan perangkat keras, kebutuhan perangkat lunak dan terdapat pula kebutuhan non-fungsional.

4.1 Gambaran Umum Sistem

Sistem yang akan dibuat yaitu modul komunikasi *wireless* berbasis Plug and Play (PnP) menggunakan komunkasi serial (UART). Modul komunikasi ini terdiri dari 2 jenis modul komunikasi yaitu ESP8266 dan NRF24l01. Cara kerja dari kedua modul ini yaitu akan otomatis dideteksi oleh *core* modul (Arduino Uno) saat dihubungkan menggunakan pin Serial (Rx Tx). Modul komunikasi PnP yang terdeteksi dapat dipilih oleh *core* modul untuk melakukan mekanisme pengiriman data. Pada penelitian ini memiliki batasan permasalahannya yaitu berfokus pada komunikasi serial UART antara modul komunikasi PnP dengan *core* modul, protokol yang digunakan untuk modul komunikasi ESP8266 yaitu MQTT dan modul komunikasi NRF24l01 hanya menggunakan *library* RF24 dalam proses pengiriman data.

4.2 Kebutuhan Fungsional

Kebutuhan fungsional merupakan kebutuhan yang wajib dipenuhi agar sistem dapat berjalan sesuai dengan fungsinya. Kebutuhan fungsional disini terdiri dari kebutuhan sistem, kebutuhan perangkat keras dan kebutuhan perangkat lunak.

4.2.1 Kebutuhan Sistem

4.2.1.1 Modul komunikasi PnP dapat terdeteksi oleh core modul menggunakan komunikasi serial (UART)

Merupakan kebutuhan sistem / fitur yang terkait dengan komunikasi serial (UART). Pada kebutuhan sistem ini modul komunikasi PnP seperti ESP8266 ataupun NRF24l01 akan dimodifikasi menjadi sebuah modul baru yang sudah terhubung dengan Arduino Uno untuk ESP8266 dan *chip* ATtiny85 untuk NRF24l01. ATtiny85 ataupun Arduino nano yang akan menjadi perantara antara modul komunikasi PnP dengan *core* modul agar bisa saling berkomunikasi dengan komunikasi serial. Modul komunikasi PnP akan mengirimkan pesan yang bisa di identifikasi oleh *core* modul. Pesan "1" untuk identifikasi modul komunikasi berbasis ESP8266 dan pesan "2" untuk identifikasi modul komunikasi berbasis NRF24l01 yang merupakan bagian dari proses *Triple Handshaking* untuk pengenalan modul, pemilihan modul, dan konfirmasi kesiapan modul melakukan pengiriman data.

4.2.1.2 Modul komunikasi PnP yang terdeteksi dapat dipilih oleh core modul

Merupakan kebutuhan sistem / fitur terkait dengan pemilihan modul komunikasi PnP. Pada kebutuhan sistem ini modul komunikasi PnP yang telah terhubung ke *core* modul akan menampilkan informasi tentang modul yang telah terhubung. Setelah *core* modul menerima pesan identifikasi dari modul komunikasi , *core* modul maupun modul komunikasi PnP akan melakukan sisa proses *Triple Handshaking*untuk melakukan pemilihan dan konfirmasi kesiapan modul komunikasi PnP dalam melakukan pengiriman data.

4.2.1.3 Modul komunikasi PnP ESP8266 dapat melakukan pengiriman data menggunakan protokol MQTT

Merupakan kebutuhan sistem / fitur yang terkait dengan pengiriman data secara *wireless* menggunakan modul komunikasi PnP ESP8266 dan protokol MQTT. Pada fitur ini , data yang telah ditampung oleh *core* modul sebelumnya akan diteruskan ke modul komunikasi ESP8266 melalui komunikasi serial dan menggunakan protokol MQTT. Namun sebelum mengupload program ke modul komunikasi PnP tersebut harus diberi tambahan pengaturan nama SSID dan password untuk dapat terhubung ke WiFi karena pada dasarnya untuk menggunakan ESP8266 harus terhubung jaringan intenet. Untuk membuktikan pengiriman data menggunakan modul komunikasi PnP ESP8266 berhasil dapat dilakukan menggunakan aplikasi MQTT agar dapat melihat data yang berhasil terkirim.

4.2.1.4 Modul komunikasi PnP NRF24l01 dapat melakukan pengiriman data memanfaatkan *library* RF24

Merupakan kebutuhan sistem/fitur yang terkait dengan pengiriman data secara *wireless* menggunakan modul komunikasi PnP NRF24l01 memanfaatkan *library* RF24. Pada fitur ini , data yang telah ditampung oleh *core* modul sebelumnya akan diteruskan ke modul komunikasi NRF24l01 melalui komunikasi serial kemudian dikiirim menggunakan *library* RF24 yang telah terpasang pada modul komunikasi sebelumnya . Untuk membuktikan data yang dikrim modul komunikasi PnP NRF24l01 sudah berhasil maka diperlukan *receiver* NRF24l01 untuk dapat menerima data yang telah dikirim. Komponen yang dibutukan untuk membentuk *receiver* ini yaitu Arduino Uno dan NRF24l01 serta menggunakan *library* RF24 juga.

4.2.2 Kebutuhan Perangkat Keras

Pada bagian ini menjelaskan terkait dengan kebutuhan perangkat keras pada sistem ini. Perangkat keras ini meliputi dari sekumpulan komponen elektronika yang akan membentuk suatu sistem perangkat keras baru yang

memiliki fungsi tertentu. Berikut adalah kebutuhan perangkat keras yang digunakan pada proses implementasi sistem, yakni sebagai berikut :

1. Mikrokontroler Arduino Uno

   Mikrokontroler Arduino Uno memiliki *chip* berbasis ATmega328 sehingga bisa digunakan untuk mengupload program dan menjalankan fungsi pengiriman data secara *wireless* dengan menggunakan modul tambahan seperti ESP8266 ataupun NRF24L01. Mikrokontroler ini juga digunakan sebagai perantara untuk mengupload program pada *chip* ATtinny85.
2. Arduino Nano

   Merupakan mikrokontroler berbasis ATmega328P yang akan menjadi perantara antara modul komunikasi ESP8266 dengan *core* modul. Arduino nano yang akan bertugas memberikan informasi dan intruksi agar modul ESP82666 dapat terdeteksi oleh *core* modul saat dihubungkan dengan pin serial sehingga dapat melakukan komunikasi *wireless* dengan kontrol menggunakan komunikasi serial (UART).
3. ATtiny85

   ATtiny85 merupakan *chip* mikrokontroler yang dapat diupload program dan menjalankan fungsi seperti mikrokontroler Arduino. ATtiny85 inilah yang menjadi bagian utama pada modul komunikasi PnP agar saat terhubung dengan *core* modul dapat terdeteksi secara otomatis. ATtiny85 akan diberi logika untuk mengatur komunikasi serial(UART) antara modul komunikasi NRF24l01 dengan *core* modul.
4. Modul ESP8266

   Modul ESP8266 merupakan modul komunikasi tambahan berfungsi untuk mengirimkan data dari mikrokontroler secara *wireless* dengan memanfaatkan jaringan internet terutama layanan Wifi yang ada serta protokol tertentu. Pada sistem menggunakan protokol MQTT untuk melakukan pengiriman data.
5. Modul NRF24l01

   Modul NRF24l01 merupakan modul komunikasi tambahan seperti ESP8266 yang berfungsi untuk mengirimkan data dari mikrokontroler secara *wireless*. Namun perbedaannya yaitu NRF24l01 tidak menggunakan jaringan internet, melainkan sinyal radio dengan frekuensi 2.4GHz. Jadi untuk penggunaan modul ini tidak memerlukan jaringan internet seperti wifi. Namun untuk melakukan kegiatan komunikasi diperlukan 2 buah modul NRF24l01 untuk bertindak sebagai *Transmitter* dan *Receiver*.

4.2.3 Kebutuhan Perangkat Lunak

Berikut adalah kebutuhan perangkat keras yang digunakan pada proses implementasi sistem, yakni sebagai berikut :

1. Arduino IDE

   IDE itu merupakan kependekan dari *Integrated Developtment Enviroenment,* atau secara bahasa mudahnya merupakan lingkungan terintegrasi yang digunakan untuk melakukan pengembangan. Arduino menggunakan bahasa pemrograman sendiri yang menyerupai bahasa C. Bahasa pemrograman Arduino (*Sketch*) sudah dilakukan perubahan untuk memudahkan pemula dalam melakukan pemrograman dari bahasa aslinya. Sebelum dijual ke pasaran, IC mikrokontroler Arduino telah ditanamkan suatu program bernama *Bootlader* yang berfungsi sebagai penengah antara *compiler* Arduino dengan mikrokontroler. Arduino IDE dibuat dari bahasa pemrograman JAVA. Arduino IDE juga dilengkapi dengan *library* C/C++ yang biasa disebut *wiring* yang membuat operasi input dan output menjadi lebih mudah. Arduino IDE ini dikembangkan dari software Processing yang dirombak menjadi Arduino IDE khusus untuk pemrograman dengan Arduino.
2. Aplikasi MQTT

   Aplikasi MQTT merupakan perangkat lunak yang memudahkan user saat memanfaatkan protokol MQTT untuk pengiriman data melalui jaringan internet. Aplikasi MQTT akan memberikan akses bagi para pengguna untuk melihat data yang berhasil dikirimkan ke broker MQTT. Untuk dapat melakukan hal tersebut pengguna haru mengatur nama server, *username*, *password*, beserta topik yang digunakan saat pengiriman data yang diset pada *perangkat* berbasis internet sebelumnya. Pengaturan tersebut harus sama agar data yang diakses dapat sinkron dengan kiriman data oleh perangkat.
3. *Library SoftwareSerial*

   Arduino maupun papan Genuino telah didukung dengan komunikasi serial pad pin 0 dan 1. Namun telah dikembangkan *library SoftwareSerial* untuk memungkinkan komunikasi serial lebih dari 1 dengan menggunakan pin-pin digital pada papan Arduino. *Library* ini sangat membantu pada saat dibutuhkan komunikasi serial yang berjalan pada dua atau lebih *perangkat*. Jadi dengan menggunakan *library* ini, papan Arduino bisa melakukan komunikasi serial lebih dari 1 perangkat.
4. *Library* RF24

   Untuk dapat menggunakan modul komunikasi *wireless* seperti NRF24l01 disediakan beberapa *library* seperti RF24. *Library* RF24 lebih banyak digunakan karena *compatible* pada banyak papan maupun *chip* mikrokontroler seperti Arduino, Attinny85, Raspberry Pi dan beberapa

papan yang jarang digunakan untuk komunikasi *wireless* menggunakan radio.

5. *Library* PubSubClient

   Library PubSubClient merupakan *library* yang akan membantu dalam implementasi protokol MQTT pada perangkat Arduino. *Library* inilah yang akan mengatur agar modul komunikasi PnP ESP8266 dapat terhubung dengan broker melalui jaringan internet.

4.3 Kebutuhan Non Fungsional

Kebutuhan Kebutuhan non-fungsional merupakan kebutuhan yang bersifat opsional agar sistem dapat berjalan lebih baik, namun apabila tidak dijalankan sistem tetep berjalan sesuai dengan fungsionalnya.

4.3.1 Kebutuhan *Portability*

Karena sistem ini bersifat *Plug and* Play (PnP) maka diperlukan kemudahan dalam penggunaaanya dan langkah-langkah untuk menggunakan tidak terlalu banyak. Sehingga pengguna dapat langsung menggunakan fungsi pengiriman data sesuai dengan modul komunikasi PnP yang terdeteksi dan dipilih oleh *core* modul.

4.3.2 Kebutuhan *Reliability*

Kebutuhan *reliability* dari sistem yang dibuat ini yaitu kehandalan pengiriman juga diperhatikan dengan melakukan pengiriman minimal tiap 1.5 detik sekali agar data yang dihasilkan lebih handal dan bisa *real time* setiap ada perubahan data. Pengiriman data memerlukan waktu minimal 1.5 detik karena 0.5 detik pertama sudah digunakan untuk sinkronisasi komunikasi UART antara modul komunikasi dan *core* modul dan selanjutnya waktu pengiriman data selama 1 detik, sehingga total waktu untuk melakukan pengiriman data minimal yaitu 1.5 detik.

BAB 5
PERANCANGAN DAN IMPLEMENTASI

5.1 Perancangan

Setelah proses analisis kebutuhan selesai dilakukan, lanjut ketahap berikutnya yaitu perancangan dan implementasi. Perancangan dilakukan berdasarkan hasil dari analisis kebutuhan yang telah dibuat meliputi kebutuhan fungsional maupun non fungsional. Perancangan dimulai dengan studi kasus mengenai bagaimana merancang modul komunikasi *wireless* yang bersifat *Plug and Play* (PnP), setelah itu melakukan perancangan skematik pada modul komunikasi PnP yang terbagi yaitu modul komunikasi PnP ESP8266 dan modul Komunikasi PnP NRF24L01.

5.1.1 Perancangan Arsitektur Modul Komunikasi PnP

Pada perancangan arsitektur modul komunikasi PnP terbagi menjadi 2 bagian yaitu :

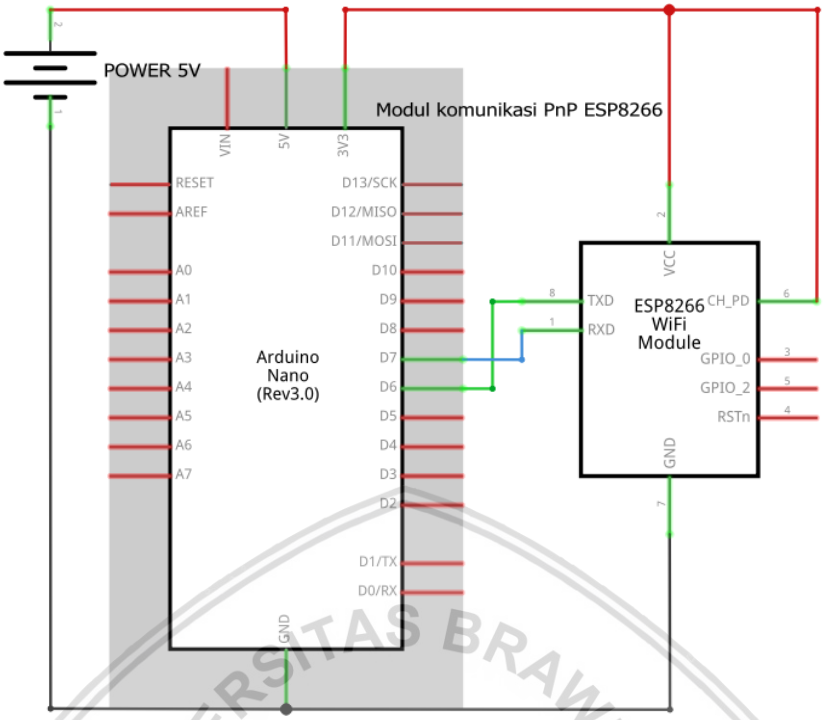
5.1.1.1 Perancangan arsitektur modul komunikas PnP ESP8266

Pada perancangan modul komunikasi PnP ESP8266 terdiri dari beberapa komponen penting yaitu ESP8266 dan Arduino nano. Kedua komponen ini saling dihubungkan dengan aturan pin sebagai berikut :

Tabel 5.1 Hubungan antar pin ESP8266 dan Arduino nano

ESP8266	Arduino Nano
VCC	3.3V
CH_PD	3.3V
TX	RX(D6)
RX	TX(D7)
GND	GND

Dari tabel 5.1 diketahui modul ESP8266 bekerja pada tegangan 3.3V sehingga pin VCC dihubungkan ke pin 3.3V dan seperti biasa pin GND (ground) dihubungkan dengan pin GND juga. Pin CH_PD harus dalam power up untuk dapat mengggunakan fungsi ESP8266 dengan benar, sehingga pin CH-PD yang defaultnya dalam keadaan *power down* dihubungkan ke pin 3.3V agar menjadi *power up*. Terdapat pin TX dan RX pada ESP8288 sehingga agar dapat berkomunikasi ke Arduino nano dapat dilakukan dengan komunikasi serial dimana pin RX ESP8266 dihubungkan ke pin TX Arduino nano begitupun sebaliknya. Namun pin serial (RX TX) yang digunakan pada Arduino Uno bukan pin serial sebenarnya, tetapi pin digital yang dibuat seolah-olah menjadi pin serial dengan bantuan *library SoftwareSerial* yaitu pin D6(Digital 6) sebagai pin RX dan pin D7(Digital 7) sebagai pin TX. Sedangkan pin RX TX yang sebenarnya pada Arduino nano akan digunakan untuk berkomunikasi serial ke *core* modul. Untuk lebih memperjelas visualisasi hubungan pin antara ESP8266 dan Arduino nano, berikut adalah skematiknya pada gambar 5.1 :



Gambar 5.1 Skematik modul komunikasi PnP ESP8266

5.1.1.2 Perancangan arsitektur modul komunikasi PnP NRF24l01

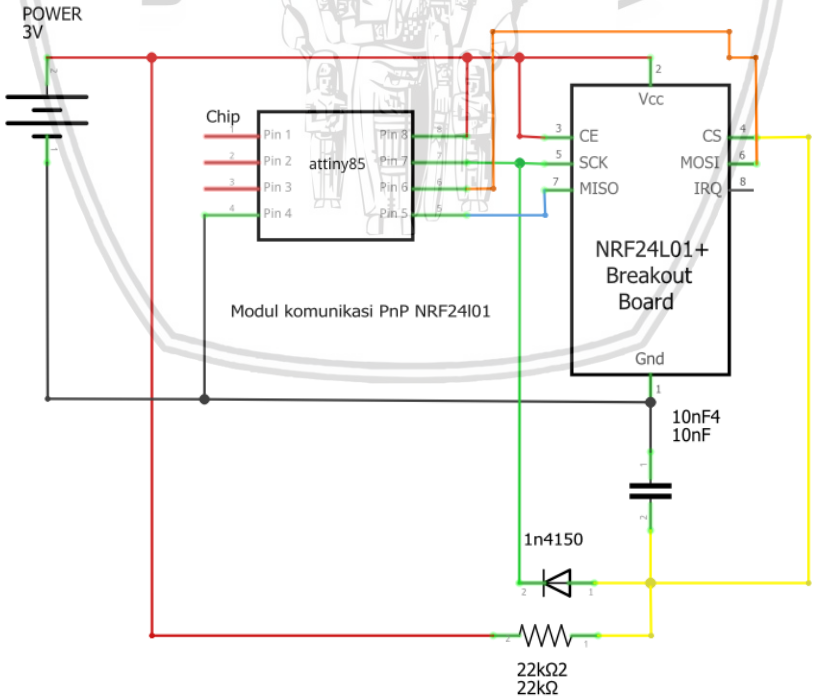
Perancangan arsitektur modul komunikasi PnP NRF24l01 terdiri dari beberapa komponen penting diantaranya NRF24l01 dan ATtiny85. Kedua komponen ini saling dihubungkan dengan aturan pin sebagai berikut :

Tabel 5.2 Hubungan antar pin NRF24l01 dan ATtiny85

NRF24l01	ATtiny85
VCC	Pin 8 (3.3V)
CE	Pin 8 (3.3V)
CSN	Pin 8 (3.3V)
SCK	Pin 7
MOSI	Pin 6
MISO	Pin 5
GND	Pin 4 (GND)

Dari tabel 5.2 diketahui modul NRF24l01 bekerja pada tegangan 3.3V sehingga pin VCC dihubungkan ke pin 3.3V dan seperti biasa pin GND (ground) dihubungkan dengan pin GND juga. Pin CE dan CSN pada NRF24l01 umumnya dihubungkan dengan pin digital, namun karena pin digital ATtiny85 terbatas sehingga dihubungkan saja dengan pin 3.3V dari ATtiny85, hal tersebut dilakukan karena rata-rata pin digital pada mikrokontroler bekerja pada tegangan 3.3V. Pin

SCK NRF24l01 dihubungkan dengan pin 7 ATtiny85 yang bertindak sebagai pin SCK(Serial Clock). Pin MOSI dan MISO NRF24l01 masing-masing dihubungkan dengan pin 6 ATtiny85 sebagi pin MOSI dan pin 5 ATtiny85 sebagai pin MISO.Dan terdapat beberapa komponen tambahan seperti resistor (R1) 22 k ohm untuk memberikan penghambat/penahan arus listrik, *capacitor* (C1) 10 nF yang berfungsi menyimpan energi listrik dan dioda (D1) tipe1n4148 yang berfungsi sebagai penyearah arus listrik. Resistor 22k ohm terhubung dengar power 3.3v dan diteruskan ke pin CSN NRF24l01, D1 dan C1. D1 juga terhubung ke pin SCK NRF24l01 dan ATtiny85 sedangkan C1 terhubung dengan pin GND NRF24L01. Komponen-komponer tersebut ditambahkan karena saat pin SCK ATtiny85 berada pada posisi *high* dalam beberapa *microsecond*, C1 akan mengisi daya melalui R1 dan membuat pin CSN menjadi *High.* Jika pin SCK berada pada posisi *Low* untuk beberapa *microsecond* sebelum memasukan data pada pin SPI (MOSI, MISO dan SCK), C1 akan melepaskan arus listrik melewati D1 dan membuat pin CSN dalam keadaan LOW. *High Pulse* pada pin SCK kurang dari beberapa *microsecond* selama komunikasi NRF24l01 tidak akan bertahan cukup lama untuk mengisi daya pada C1, sehingga pin CSN akan tetap rendah. Oleh karena itu dibutuhkan komponen-komponen tambahan tersebut agar pin CSN tetap stabil pada posisi *High* saat proses komunikasi NRF24l01 berlangsung agar tidak terdapat data *lost* saat pengiriman data. Untuk lebih memperjelas visualisasi hubungan pin antara NRF24l01 dan ATtiny85, berikut adalah skematiknya pada gambar 5.2 :



Gambar 5.2 Skematik modul komunikasi PnP NRF24l01

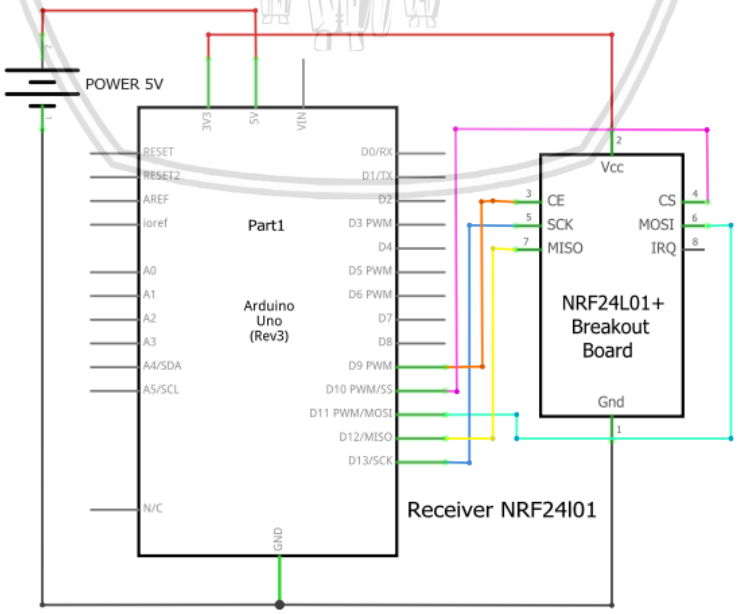
Untuk membuktikan data yang dikirim modul komunikasi PnP RF24l01 berhasil dengan membuat *receiver* NRF24l01. *Receiver* ini akan menerima data

yang dikirim oleh NRF24l01 lain dengan kesepakatan node yang di konfigurasi harus sama antar sisi pengirim dan penerima. Hubungan pin antara NRF24l01 dan Arduino nano untuk membentuk *Receiver* NRF24l01 sebagai berikut :

Tabel 5.3 Hubungan antar pin NRF24l01 dan Arduino Uno

NRF24l01	Arduino Uno
VCC	3.3V
CE	Pin D9
CSN	Pin D10
SCK	Pin D13
MOSI	Pin D11
MISO	Pin D12
GND	GND

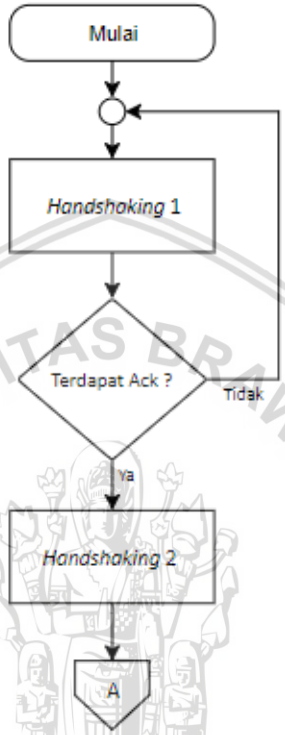
Dari tabel 5.3 diketahui modul NRF24l01 bekerja pada tegangan 3.3 volt sehingga pin VCC dihubungkan ke pin 3.3V dan seperti biasa pin GND (ground) dihubungkan dengan pin GND juga. Pin CE dan CSN NRF24l01 dihubungkan masing-masing ke pin D7 dan D8 Arduino Uno disesuaikan dengan inisilisai pada program Arduino sehinggan pin Arduino yang dihubungkan dengan pin CE dan CSN NRF24l01 bisa selain pin D7 dan D8. Pin SPI (SCK, MOSI dan MISO) NRF24l01 terhubung ke pin SPI Arduino Uno yaitu masing-masing pin D13 ke pin SCK, D11 ke pin MOSI dan D12 ke pin MISO. Untuk lebih memperjelas visualisasi hubungan pin antara NRF24l01 dan Arduino Uno, berikut adalah skematiknya pada gambar 5.3 :



Gambar 5.3 Skematik *receiver* NRF24l01

5.1.2 Perancangan Pengenalan Modul Komunikasi PnP

Untuk pengenalan kedua modul komunikasi yang ada yaitu modul komunikasi PnP ESP8266 dan NRF24l01 menggunakan *core* modul (Arduino Uno) dimanfaatkan *library SoftwareSerial* agar bisa berkomunikasi secara serial lebih dari 1 perangkat. Berikut merupakan *flowchart* program pengenalan modul komunikasi PnP :



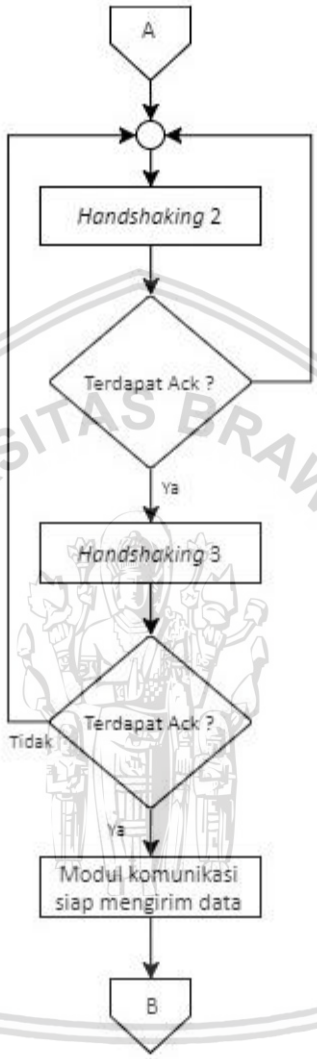
Gambar 5.4 *Flowchart* pengenalan modul komunikasi PnP

Pada gambar 5.4 merupakan *flowchart* pengenalan modul yaitu modul komunikasi PnP yang telah terhubung menggunakan komunikasi serial (UART) mengirimkan kode perangkat yang telah didefinisikan pada *core* modul sebelumnya pada proses *Handshaking* 1 . Kode "1" untuk modul komunikasi PnP ESP8266 dan kode "2" untuk modul komunikasi PnP NRF24l01. Jika kode "1" ataupun kode "2" terdeteksi, maka core modul akan mengirim Ack (*acknowledgement*) agar modul komunikasi PnP masuk pada tahap *Handshaking* 2 dan menampilkan modul komunikasi PnP terdeteksi di serial monitor *core* modul.

5.1.3 Perancangan Pemilihan Modul Komunikasi PnP yang Terdeteksi

Setelah modul komunikasi terhubung secara serial dengan *core* modul dan masuk pada bagian *Handshaking* 2. Core modul pun mengirimkan pesan lagi sebagai bentuk pesan bahwa modul siap dipilih untuk melakukan pengiriman. Jika

terdapat Ack lagi dari core modul atas pesan tadi, maka modul komunikasi PnP pun masuk pada bagian *Handshaking* 3. *Handshaking* 3 sebagai tahap terakhir untuk memberi tahu *core* modul bahwa modul komunikasi PnP siap melakukan pengiriman data. Berikut merupakan *flowchart* program pemilihan modul komunikasi PnP :

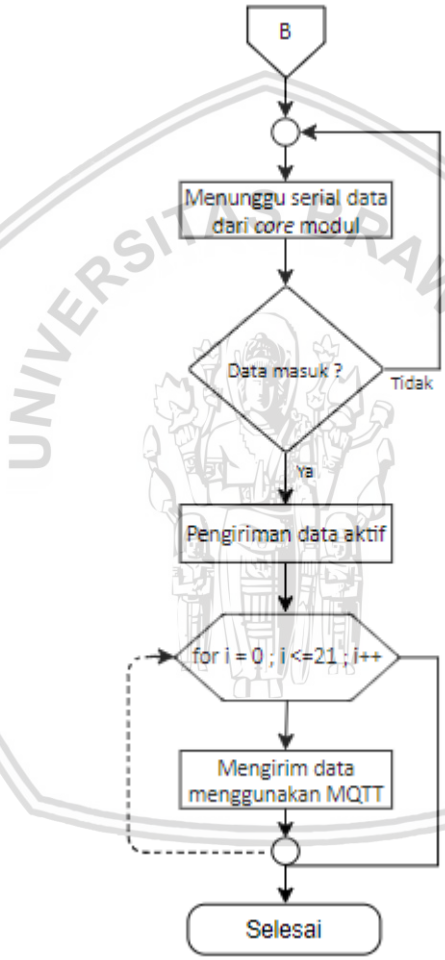


Gambar 5.5 *Flowchart* pemilihan modul komunikasi PnP

Pada gambar 5.5 merupakan *flowchart* yang menjelaskan proses pemilihan modul komunikasi PnP yang sudah terdeteksi hingga siap untuk melakukan pengiriman data. Saat modul komunikasi mengirim pesan pada tahap *Handshaking* 2 dan terdapat ack (*acknowledgement*) dari *core* modul maka modul komunikasi masuk ke tahap *Handshaking* 3 untuk memberitahu ke *core* modul bahwa modul komunikasi sudah siap mengirim data. Jika terdapat ack dari core modul terhadap pesan *Handshaking* 3. Modul komunikasi pun siap untuk mengirimkan data dari *core* modul.

5.1.4 Perancangan Pengiriman Data Menggunakan Modul Komunikasi PnP ESP8266

Saat proses deteksi dan pemilihan modul berhasil, modul komunikasi terpilih pun seperti ESP8266 akan menunggu data dari *core* modul melalui komunikasi serial. Saat data diterima oleh modul komunikasi, langsung dikirim menggunakan protokol MQTT sebanyak 21 kali setiap 1.1 detik (1100 ms). Untuk melihat data yang telah berhasil dikrim oleh modul komunikasi yaitu dengan menggunakan aplikasi MQTT. Berikut merupakan *flowchart* pengiriman data modul komunikasi PnP ESP8266 :



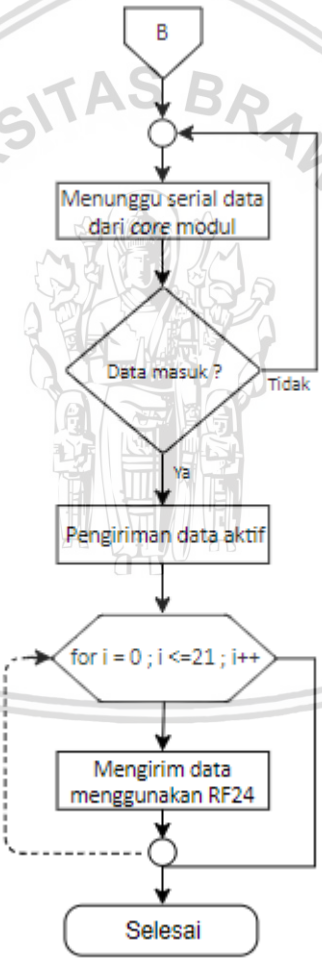
Gambar 5.6 *Flowchart* pengiriman data modul komunikasi PnP ESP8266

Pada gambar 5.6 terdapat *flowchart* yang menjelaskan pengiriman data pada modul komunikasi PnP ESP8266. Setelah proses pengenalan dan pemilihan modul yang terdeteksi selesai, modul komunikasi terpilih yaitu ESP8266 akan menungggu data dari *core* modul melalui komunikasi serial (UART). Jika terdapat data yang masuk pada modul komunikasi, fungsi kirim akan aktif dan akan

melakukan pengiriman data yang masuk melalui komunikasi serial sebanyak 21 kali. Dan data yang telah dikirim dapat diakses menggunakan aplikasi MQTT.

5.1.5 Perancangan Pengiriman Data Menggunakan Modul Komunikasi PnP NRF24l01

Saat proses deteksi dan pemilihan modul berhasil, modul komunikasi terpilih pun seperti NRF24l01 akan menunggu data dari *core* modul melalui komunikasi Serial. Saat data diterima oleh modul komunikasi, langsung dikirim menggunakan modul NRF24l01 sebanyak 21 kali setiap 1 detik (1000 ms) memanfaatkan *library* RF24. Untuk melihat data yang telah berhasil dikrim oleh modul komunikasi NRF24l01 yaitu dengan menggunakan *receiver* NRF24l01 yang terdiri dari modul NRF24l01 dan Arduino Uno/nano. Berikut merupakan *flowchart* pengiriman data modul komunikasi PnP NRF24l01 :



Gambar 5.7 *Flowchart* pengiriman data modul komunikasi PnP ESP8266

Pada gambar 5.7 terdapat *flowchart* yang menjelaskan pengiriman data pada modul komunikasi PnP NRF24l01. Setelah proses pengenalan dan pemilihan modul yang terdeteksi selesai, modul komunikasi terpilih yaitu NRF24l01 akan menungggu data dari *core* modul melalui komunikasi serial (UART). Jika terdapat

data yang masuk pada modul komunikasi, fungsi kirim akan aktif dan akan melakukan pengiriman data yang masuk melalui komunikasi serial sebanyak 21 kali. Data yang berhasil dikrimkan oleh modul komunikasi NRF24l01 dapat dilihat melalui *receiver* NRF24l01 melalui serial monitor.

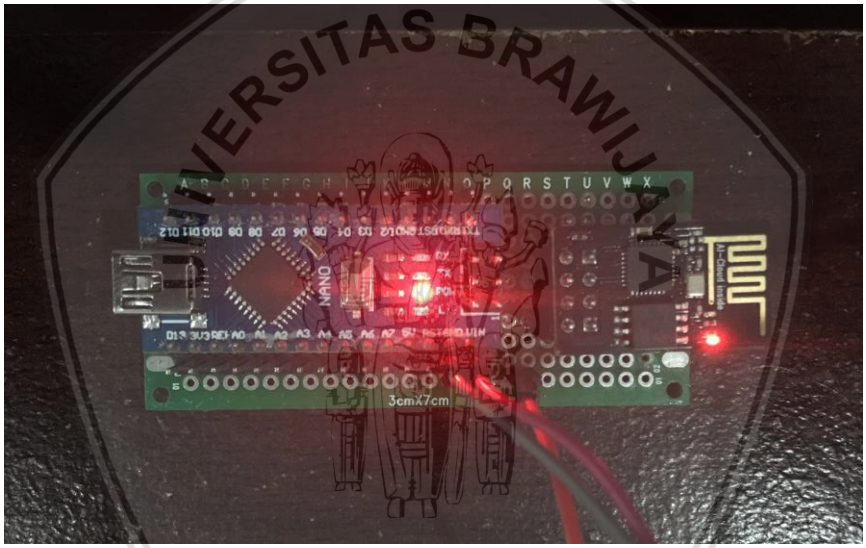
5.2 Implementasi

Setelah proses perancangan selesai dilakukan, lanjut ketahap berikutnya yaitu implementasi. Implementasi yang akan dilakukan sesuai dengan perancangan yang telah dibuat sebelumnya.

5.2.1 Implementasi Arsitektur Modul Komunikasi PnP

5.2.1.1 Implementasi arsitektur modul komunikasi PnP ESP8266

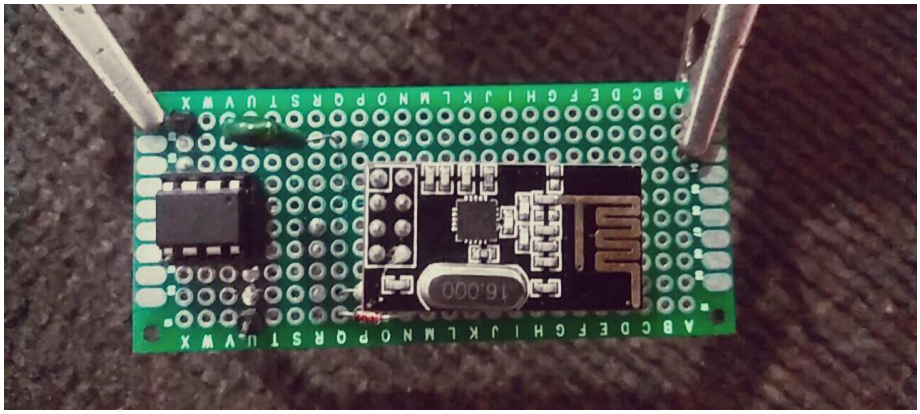
Pada implementasi modul komunikasi PnP ESP8266 seperti gambar 5.8 terdiri dari Arduino nano dan ESP8266 dan dihubungkan agar kedua komponen utama tersebut dapat terhubung dan bekerja dengan baik.



Gambar 5.8 Arsitektur modul komunikasi PnP ESP8266

5.2.1.2 Implementasi arsitektur modul komunikasi PnP NRF24l01

Pada implementasi modul komunikasi PnP ESP8266 seperti gambar 5.9 terdiri dari ATtiny85 dan NRF24l01 dan dihubungkan jumper dengan agar kedua komponen utama tersebut dan terhubung dan bekerja dengan baik. Serta ditambahkan beberapa komponen agar pengiriman data menggunakan NRF24l01 lebih stabil saat menggunakan ATtiny85 seperti kapasitor keramik 10nF, resistor 22k ohm, serta dioda tipe 1n4148.



Gambar 5.9 Arsitektur modul komunikasi PnP NRF24l01

5.2.2 Implementasi Mekanisme Pengenalan Modul Komunikasi PnP

Untuk implementasi pengenalan modul komunikasi seperti pada perangcangan sebelumya yaitu terdapat source code untuk mengirimkan pesan dari modul komunikasi PnP ke *core* modul. Modul komunikasi akan mengirimkan pesan "1" pada *core* modul untuk pengenalan modul komunikasi ESP8266 atau pesan "2" untuk untuk pengenalan modul komunikasi NRF24l01. Untuk proses pengenalan modul komunikasi PnP ini mengikuti bagian pola *Triple Handshaking, yaitu Handshaking 1*.

5.2.2.1 *Source code* pengenalan modul komunikasi PnP ESP8266

Pada gambar 5.10 menunjukkan penggalan *source code* pengenalan modul komunikasi PnP ESP8266.

modul_esp8266 § arduino_esp

```
#include <SoftwareSerial.h>
SoftwareSerial cetak(2, 3); //(RX , TX)
SoftwareSerial esp8266 (6, 7); //(RX , TX)
int data, codeEsp , isi;
unsigned long previousMillis = 0;

void setup() {
  Serial.begin(9600);
  cetak.begin(9600);
  bool cekGlobal = true;
  bool cek1 = true;
  bool cek2 = false;
  while (cekGlobal) {
    unsigned long currentMillis = millis();
    if (currentMillis - previousMillis >= 1500) {
      previousMillis = currentMillis;
      cetak.print(1);
      Serial.println("Handshaking 1 \n");
```

Gambar 5.10 Penggalan *source code* pengenalan modul komunikasi PnP ESP8266

Dari penggalan *source code* gambar 5.10 terlihat *include library SofwareSerial* untuk berkomunikasi serial antara modul komunikasi dan *core* modul . Terdapat inisialisasi variabel yang akan digunakan pada pada sistem ini serta terdapat fungsi perintah "cetak.print(1)" yang berfungsi untuk mengirim data 1 sebagai kode identifikasi pada *core* modul agar bisa mendeteksi modul komunikasi ESP8266 saat terhubung menggunakan pin serial (RX , TX) atau disebut dengan proses *Handshaking* 1.

5.2.2.2 *Source code* pengenalan modul komunikasi PnP NRF24l01

Pada gambar 5.11 menunjukkan penggalan *source code* pengenalan modul komunikasi PnP NRF24l01.

modul_nrf24l01

```
#include <SoftwareSerial.h>
#include "RF24.h"
SoftwareSerial cetak(3 , 4); // (RX TX)
RF24 radio(3, 3);

byte address[11] = "SimpleNode";
unsigned long payload = 0;
unsigned long previousMillis = 0;

void setup() {
  radio.begin();
  radio.openWritingPipe(address);
  cetak.begin(9600);
  int isi1;
  bool cekGlobal = true;
  bool cek1 = true;
  bool cek2 = false;
  while (cekGlobal) {
    unsigned long currentMillis = millis();
    if (currentMillis - previousMillis >= 1500) {
      previousMillis = currentMillis;
      cetak.print(2); //Handshaking1
```

Gambar 5.11 Penggalan *source code* pengenalan modul komunikasi PnP NRF24l01

Dari penggalan *source code* gambar 5.11 terlihat *include library SofwareSerial* untuk berkomunikasi serial antara modul komunikasi dan *core* modul serta Include RF24 agar dapat menggunakan fungsi pengiriman data NRF24l01 . Terdapat inisialisasi variabel dan fungsi yang akan digunakan pada pada sistem ini serta terdapat fungsi perintah "cetak.print(2)" yang berfungsi untuk mengirim data 2 sebagai kode identifikasi pada *core* modul agar bisa mendeteksi modul komunikasi NRF24l01 saat terhubung menggunakan pin serial (RX , TX) atau disebut dengan proses *Handshaking* 1.

5.2.3 Implementasi Pemilihan Modul Komunikasi PnP yang Terdeteksi

Pada implemetasi pemilihan modul komunikasi PnP menggunakan pola *Triple Handshaking* dari pengenalan hingga pemilihan modul komunikasi.

5.2.3.1 *Source code* pemilihan modul komunikasi PnP ESP8266

Pada gambar 5.12 menunjukkan penggalan *source code* pengenalan-pemilihan modul komunikasi PnP ESP8266.

modul_esp8266 §

```
14  while (cekGlobal) {
15    unsigned long currentMillis = millis();
16    if (currentMillis - previousMillis >= 1500) {
17      previousMillis = currentMillis;
18      cetak.print(1);
19      Serial.println("Handshaking 1 \n");
20    } else {
21      if (cetak.available() > 0) {
22        isi = cetak.parseInt();
23        if (cek1) {
24          if (isi == 1) {
25            Serial.println("Handshaking 2 \n");
26            cetak.println("11");
27            cek1 = false;
28            cek2 = true;
29          } delay(100);
30        }
31        if (cek2) {
32          isi = cetak.parseInt();
33          if (isi == 11) {
34            Serial.println("Handshaking 3 & Modul Siap Kirim \n");
35            cetak.println("111");
36            cek1 = false;
37            cek2 = false;
38            cekGlobal = false;
39            break;
40          } delay(100);
41        }
42      }
43    }
44  }
45  esp8266.begin(115200);
```

Gambar 5.12 Penggalan *source code* pengenalan-pemilihan modul komunikasi PnP ESP8266

Dari penggalan *source code* gambar 5.12 setelah modul komunikasi mengirim pesan 1 melalui perintah "cetak.print(1)" modul komunikasi akan mengecek ack dari *core* modul yaitu berupa data "1", jika terdapat ack tersebut maka proses *Handshaking* 2 dijalankan. Pada proses ini modul komunikasi mengirimkan pesan "11" melalui perintah "cetak.println("11")"untuk memberi informasi ke *core* modul bahwa modul komunikasi yang terdeteksi siap dipilih. Kemudian modul komunikasi menunggu ack dari core modul berupa pesan "11". Saat ack tersebut diterima maka proses *Handshaking* 3 akan dijalankan, dan modul komunikasi mengirimkan pesan "111" melalui perintah

"cetak.println("111") ke *core* modul bahwa modul komunikasi siap mengirimkan data.

5.2.3.2 *Source code* pemilihan modul komunikasi PnP NRF24l01

Pada gambar 5.13 menunjukkan penggalan *source code* pengenalan-pemilihan modul komunikasi PnP NRF24l01.

modul_nrf24l01 §

```
18  while (cekGlobal) {
19    unsigned long currentMillis = millis();
20    if (currentMillis - previousMillis >= 1500) {
21      previousMillis = currentMillis;
22      cetak.print(2); //Handshaking 1
23    } else {
24      if (cetak.available() > 0) {
25        isi1 = cetak.parseInt();
26        if (cek1) {
27          if (isi1 == 2) {
28            cetak.println(22);//Handshaking 2
29            cek1 = false;
30            cek2 = true;
31          } delay(500);
32        }
33        if (cek2) {
34          isi1 = cetak.parseInt();
35          if (isi1 == 22) {
36            cetak.println(222);//Handshaking 3
37            cek1 = false;
38            cek2 = false;
39            cekGlobal = false;
40            break;
41          } delay(500);
42        }
43      }
44    }
45  }
```

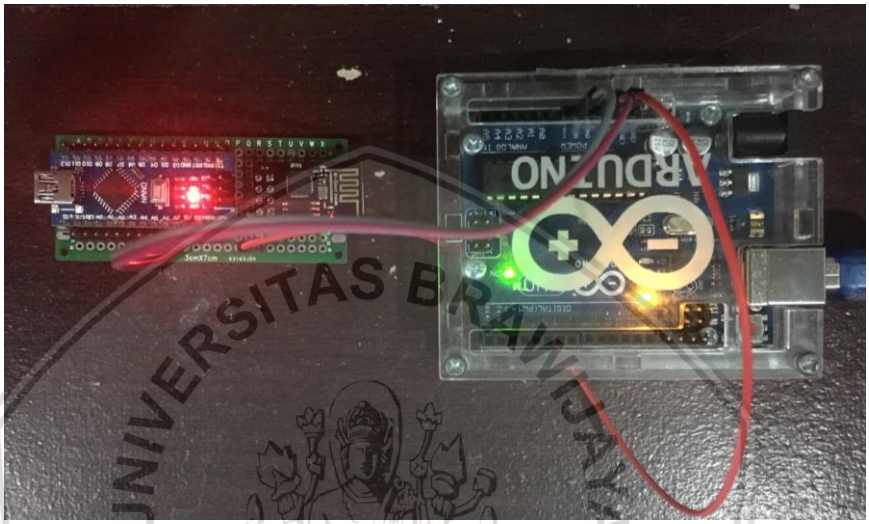
Auto Format finished.

Gambar 5.13 Penggalan *source code* pengenalan-pemilihan modul komunikasi PnP NRF24l01

Dari penggalan *source code* gambar 5.13 setelah modul komunikasi mengirim pesan 1 melalui perintah "cetak.print(2)" modul komunikasi akan mengecek ack dari *core* modul yaitu berupa data "2", jika terdapat ack tersebut maka proses *Handshaking 2* dijalankan. Pada proses ini modul komunikasi mengirimkan pesan "22" melalui perintah "cetak.println("22") untuk memberi informasi ke *core* modul bahwa modul komunikasi yang terdeteksi siap dipilih. Kemudian modul komunikasi menunggu ack dari *core* modul berupa pesan "22". Saat ack tersebut diterima maka proses *Handshaking* 3 akan dijalankan, dan modul komunikasi mengirimkan pesan "222" melalui perintah "cetak.println("222")" ke *core* modul bahwa modul komunikasi siap mengirimkan data.

5.2.4 Implementasi Pengiriman Data Menggunakan Modul komunikasi PnP ESP8266

Pada implementasi pengiriman data menggunakan modul komunikasi PnP ESP8266 seperti gambar 5.14 *core* modul dan modul komunikasi dihubungkan secara serial (UART). Saat modul komunikasi sudah terdeteksi dan dipilih oleh *core* modul, maka modul komunikasi akan menjalankan fungsi pengiriman menggunakan protokol MQTT dan untuk melihat data yang dikirim menggunakan protokol MQTT dapat menggunakan aplikasi MQTT.



Gambar 5.14 Modul komunikasi terhubung dengan *core* modul menggunakan komunikasi serial (UART)

Pada gambar 5.15 menunjukkan penggalan *source code* modul komunikasi saat proses pengiriman data oleh modul komunikasi ESP8266 aktif.

modul_esp8266 § | arduino_esp

```
48 void loop() {
49   previousMillis = 0;
50   if (esp8266.available()) {
51     delay(100);
52     data = esp8266.parseInt();
53     if (data == 0)  Serial.println("connecting wifi ");
54     else if (data == 1) Serial.println("wifi connected  ");
55     else if (data == 2) Serial.println("connectingBroker");
56     else if (data == 3) {
57       Serial.println("broker connected"); ; //membaca kode "3"
58       int isi = 0;
59       int count = 0;
60       while (count != 21) {
61         esp8266.println(isi++);
62         Serial.print("mengirim data : ");
63         Serial.println(isi);
64         count++;
65         delay(1100);
66       }
67     }
68     else if (data == 4) Serial.println("error");
69   } else {
70     unsigned long currentMillis = millis();
71     if (currentMillis - previousMillis >= 5000) {
72       previousMillis = currentMillis;
73       Serial.println("ESP belum terdeteksi");
74       data = esp8266.parseInt();
75     }
76   }
77 }
```

Gambar 5.15 Penggalan *source code* modul komunikasi saat proses pengiriman data aktif

Dari penggalan *source code* gambar 5.15 setelah modul komunikasi PnP ESP8266 terdeteksi dan siap melakukan pengiriman. Terdapat fungsi untuk melakukan koneksi ke internet hingga terhubung ke broker MQTT agar modul komunikasi ESP8266 dapat mengirimkan data. Dari source code diatas diketahui Arduino nano akan menunggu pesan dari ESP8266 melalui komunikasi serial. Ketika terdapat pesan 1 – 3 dari ESP8266, maka pengiriman data bisa dilakukan, namun saat pesan 4 yang diterima maka pengiriman tidak bisa dilakukan. Pada proses pesan 3 diterima ESP8266 yang menandakan modul komunikasi ini sudah terhubung dengan MQTT broker. Modul Komunikasi PnP (Arduino nano & ESP8266) akan menunggu data dan melakukan pengiriman data sebanyak 21 kali.

5.2.5 Implementasi Pengiriman Data Menggunakan Modul komunikasi PnP NRF24L01

Pada implementasi pengiriman data menggunakan modul komunikasi PnP NRF24l01 seperti gambar 5.16 *core* modul dan modul komunikasi dihubungkan secara serial (UART). Saat modul komunikasi sudah terdeteksi dan dipilih oleh *core*

modul, maka modul komunikasi akan menjalankan fungsi pengiriman data memanfaatkan *library* RF24 yang *compatible* dengan modul NRF24l01.

Gambar 5.16 Modul komunikasi terhubung dengan *core* modul menggunakan komunikasi serial (UART)

5.2.5.1 *Source code* pengiriman data menggunakan modul komunikasi PnP NRF24L01

Pada gambar 5.18 menunjukkan penggalan *source code* modul komunikasi saat proses pengiriman data oleh modul komunikasi NRF24l01 aktif.

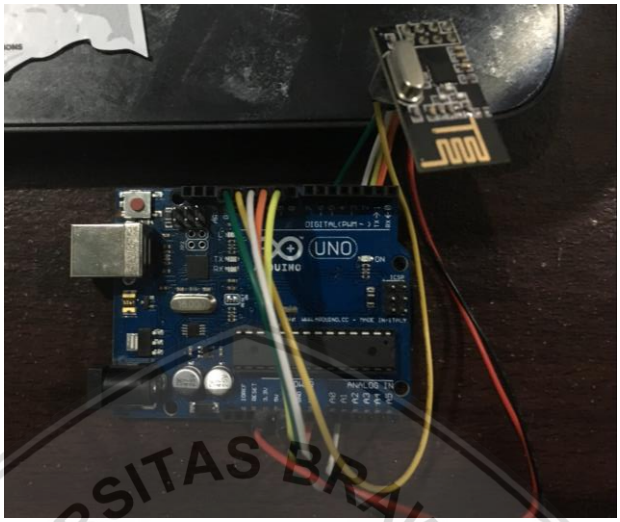
```
modul_nrf24l01 §
47
48 void loop(void) {
49   int count = 0;
50   if (cetak.available() > 0) {
51     while (count != 21) {
52       payload = cetak.parseInt();
53       radio.write( &payload, sizeof(int) ); //Send data to 'Receiver' ever second
54       count++;
55       delay(1000);
56     }
57   }
58 }
59
60
```

Gambar 5.17 Penggalan *source code* modul komunikasi saat proses pengiriman data NRF24l01

Dari penggalan *source code* gambar 5.17 setelah modul komunikasi PnP NRF24l01 terdeteksi dan siap melakukan pengiriman. Terdapat fungsi " void loop(void) " modul komunikasi NRF24l01 akan menunggu data dari core modul melalui komunikasi serial (RX , TX). Saat data dari core modul masuk maka terdapat fungsi perulangan aktif sebanyak 21 kali untuk melakukan pengiriman data menggunakan modul NRF dengan memanfaatkan *library* RF24.

5.2.5.2 *Receiver* NRF24L01

Pada gambar 5.18 terlihat rangkaian Arduino no dan NRF24l01 untuk membentuk *receiver* NRF24l01. *Receiver* inilah yang akan menampilkan data terkirim dari modul komunikasi PnP NRF24l01.



Gambar 5.18 *Receiver* NRF24l01 untuk menampung data terkirim dari NRF24l01

Serta terdapat penggalan source code yang akan diimplementasikan ke *receiver* NRF24l01 agar dapat menampilkan data yang berhasil dikirimkan oleh modul komunikasi khusunya NRF24l01.

receiver §

```
#define CE_PIN 9
#define CSN_PIN 10
#include <SPI.h>
#include "RF24.h"

RF24 radio(CE_PIN, CSN_PIN);
byte address[11] = "SimpleNode";
byte addresss[11] = "Node";
unsigned long payload = 0;

void setup() {
  Serial.begin(115200);
  radio.begin(); // Start up the radio
  radio.openReadingPipe(1, address); // Write to device address 'Simple
  radio.startListening();
}

void loop(void) {
  radio.read( &payload, sizeof(unsigned long) );
  if (payload != 0) {
    Serial.print("Got Payload ");
    Serial.println(payload);
  } else {
    Serial.println("NRF24L01 belum Terdeteksi");
  }
  delay(1000);
}
```

Gambar 5.19 Penggalan *source code receiver* NRF24l01

Dari penggalan *source code* gambar 5.19 terdapat inisialisai variabel dan *library* RF24. Terdapat "*void setup()*" untuk inisialisai objek dan "*void loop()*" untuk melakukan perulangan peneriamaan data. Pada fungsi perulangan tersebut terdapat 2 kondisi yaitu ketika tidak terdapat modul NRF24l01 yang aktif maka menampilkan informasi "NRF24l01 belum terdeteksi" dan jika terdapat modul NRF24l01 yang aktif dan mengirimkan data akan ditampilkan "*Got Payload* = data_terima".

BAB 6
PENGUJIAN DAN ANALISIS

Pada bab ini akan membahas mengenai tahap pengujian dan analisis dari hasil implementasi sistem yang telah dibuat melalii tahap perancangan & implementasi. Pengujian dilakukan untuk mengetahui apakah kebutuhan sistem yang dibuat sudah terpenuhi. Pengujian sistem ini dibagi menjadi beberapa tahap agar sesuai dengan tujuannya dan lebih mudah dalam proses analisisnya.

6.1 Pengujian Pengenalan Modul Komunikasi PnP

Pada pengujian pengenalan PnP terbagi menjadi 2 karena modul yang dibuat juga terdapat 2 jenis yang berbeda yaitu :

6.1.1 Pengujian Pengenalan Modul Komunikasi PnP ESP8266

6.1.1.1 Tujuan Pengujian

Tujuan pengujian ini untuk mengetahui dapatkah modul komunikasi PnP ESP8266 terdeteksi dan seberapa cepat modul terdeteksi oleh *core* modul (Arduino Uno).

6.1.1.2 Prosedur pengujian

Prosedur yang dikerjakan pada pengujian ini yaitu :

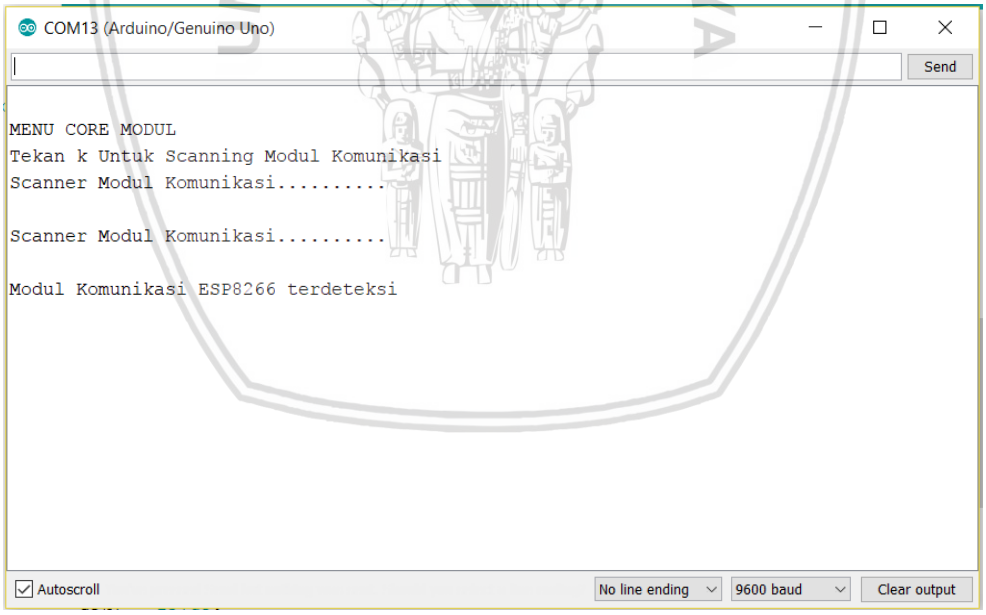
1. *Core* modul sudah dalam keadaan aktif dan melakukan pendeteksian secara terus menerus.
2. Modul komunikasi PnP ESP8266 diaktifkan dan sudah di upload program mengirim id 1 dan menampilkan proses *Handshaking* 1 pada serial monitor(jika diperlukan).
3. *Core* modul akan menampilkan modul komunikasi ESP pada serial monitor dan mengirim ACK ke modul komunikasi bahwa modul ESP8266 sudah terdeteksi.
4. Modul komunikasi PnP ESP8266 akan masuk pada tahap (*Handshaking*2) dan dapat ditampilkan di serial monitor (jika diperlukan).
5. Pengujian dilakukan sebanyak 10 kali.

6.1.1.3 Hasil pengujian

Hasil pengujian ditampilkan berupa pesan terdeteksinya modul komunikasi ESP8266 pada serial monitor *core* modul dan modul komunikasi masuk tahap *Handshaking* 2 dan dapat ditampilkan pada serial monitor. Berikut adalah tampilan serial monitor proses modul komunikasi mengirim id “1” atau disebut dengan proses *Handshaking* 1 dan menerima ack (*acknowledgment*) dari *core* modul sehingga masuk pada proses *Handshaking* 2. Hasil dari proses pengenalan pada modul komunikasi dapat dilihat pada gambar 6.1 berikut.

Gambar 6.1 Proses *Handshaking* 1-2 modul komunikasi ESP8266

Pada serial monitor *core* modul akan menampilkan informasi modul yang terdeteksi yaitu "Modul Komunikasi ESP8266 terdeteksi" dan mengirim ack pada modul komunikasi. Hasil dari proses pengenalan pada *core* modul dapat dilihat pada gambar 6.2 berikut.



Gambar 6.2 *Core* modul mendeteksi modul komunikasi ESP8266

Pengujian pengenalan modul komunikasi PnP ESP8266 ini dilakukan sebanyak 10 kali dengan tingkat keberhasilan serta waktu yang sedikit berbeda tergantung waktu menghubungkan ke *core* modul. Hasil percobaan dapat dilihat pada tabel 6.1 berikut :

Tabel 6.1 Tabel hasil percobaan pengenalan modul komunikasi ESP8266

Percobaan ke -	Berhasil terdeteksi	Waktu (mS)
1	Ya	1500
2	Ya	1500
3	Ya	1500
4	Ya	1500
5	Ya	1500
6	Ya	1500
7	Ya	1500
8	Ya	1500
9	Ya	1500
10	Ya	1500

6.1.1.4 Analisis pengujian

Hasil pengujian dari 10 kali pengenalan modul komunikasi PnP ESP8266 oleh core modul adalah berhasil sehingga presentase keberhasil mencapai 100%. Waktu yang dibutuhkan untuk pengenalan modul komunikasi rata-rata adalah 1.5 detik (1500 mS). Waktu 1.5 detik merupakan waktu minimal terdeteksinya modul.

6.1.2 Pengujian Pengenalan Modul Komunikasi PnP NRF24l01

6.1.2.1 Tujuan pengujian

Tujuan pengujian ini untuk mengetahui dapatkah modul komunikasi PnP NRF24l01 terdeteksi dan seberapa cepat modul terdeteksi oleh *core* modul (Arduino Uno).

6.1.2.2 Prosedur pengujian

Prosedur yang dikerjakan pada pengujian ini yaitu :

1. *Core* modul sudah dalam keadaan aktif dan melakukan pendeteksian secara terus menerus.
2. Modul komunikasi PnP NRF24l01 diaktifkan dan sudah di upload program mengirim id 2 dan menampilkan proses *Handshaking* 1 pada serial monitor (jika diperlukan).
3. *Core* modul akan menampilkan modul komunikasi NRF24l01 pada serial monitor dan mengirim ack ke modul komunikasi bahwa modul komunikasi sudah terdeteksi.
4. Modul komunikasi PnP NRF24l01 akan masuk pada tahap *Handshaking* 2 dan dapat ditampilkan di serial monitor (jika diperlukan).
5. Pengujian dilakukan sebanyak 10 kali.

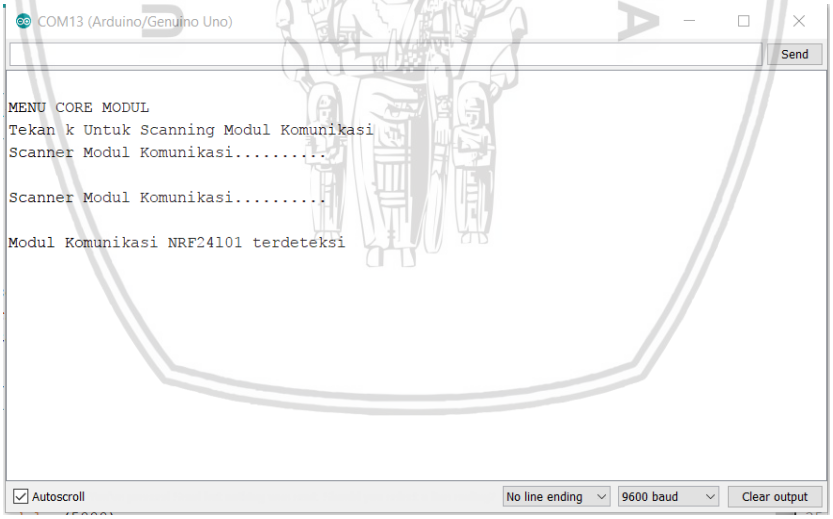
6.1.2.3 Hasil pengujian

Hasil pengujian ditampilkan berupa pesan terdeteksinya modul komunikasi ESP8266 pada serial monitor core modul dan modul komunikasi masuk tahap *Handshaking* 2 dan dapat ditampilkan pada serial monitor. Berikut adalah tampilan proses modul komunikasi mengirim id “1” atau disebut dengan proses

Handshaking 1 dan menerima ACK dari core modul sehingga masuk pada proses *Handshaking* 2.

Gambar 6.3 Proses *Handshaking* 1-2 modul komunikasi NRF24l01

Pada serial monitor *core* modul akan menampilkan informasi modul yang terdeteksi yaitu "Modul Komunikasi ESP8266 terdeteksi" dan mengirim ack pada modul komunikasi. Hasil dari proses pengenalan pada *core* modul dapat dilihat pada gambar 6.4 berikut :



Gambar 6.4 *Core* modul mendeteksi modul komunikasi NRF24l01

Pengujian pengenalan modul komunikasi PnP NRF24l01 ini dilakukan sebanyak 10 kali dengan tingkat keberhasilan serta waktu yang sedikit berbeda berdasarkan waktu dihubungkan ke *core* modul. Hasil percobaan dapat dilihat pada tabel 6.2 berikut :

Tabel 6.2 Tabel hasil percobaan pengenalan modul komunikasi NRF24l01

Percobaan ke -	Berhasil terdeteksi	Waktu (mS)
1	Ya	1500
2	Ya	1500
3	Ya	1500
4	Ya	1500
5	Ya	1500
6	Ya	1500
7	Ya	1500
8	Ya	1500
9	Ya	1500
10	Ya	1500

6.1.2.4 Analisis pengujian

Hasil pengujian dari 10 kali pengenalan modul komunikasi PnP NRF24l01 oleh *core* modul adalah berhasil sehingga presentase keberhasilan mencapai 100%. Waktu yang dibutuhkan untuk pengenalan modul komunikasi rata-rata adalah 1.5 detik (1500 mS). Waktu 1.5 detik merupakan waktu minimal terdeteksinya modul

6.2 Pengujian Pemilihan Modul Komunikasi PnP

6.2.1 Pengujian Pemilihan Modul Komunikasi PnP ESP8266

6.2.1.1 Tujuan pengujian

Tujuan pengujian ini yaitu untuk mengetahui dapatkah modul memilih modul komunikasi PnP Esp8266 yang telah terdeteksi oleh *core* modul.

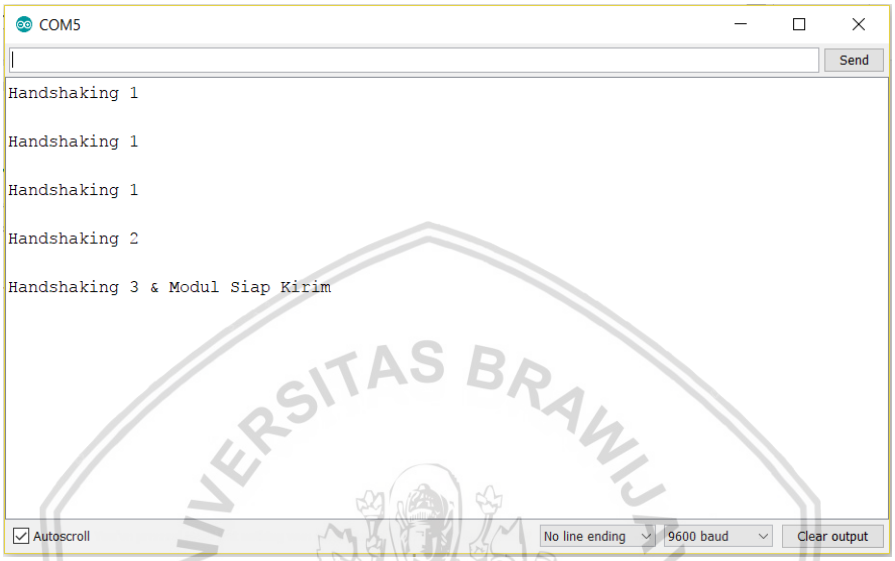
6.2.1.2 Prosedur Pengujian

Prosedur yang dikerjakan pada pengujian ini yaitu :

1. *Core* modul sudah dalam keadaan aktif dan telah mendeteksi modul komunikasi PnP yang terhubung.
2. Modul komunikasi PnP Esp8266 telah masuk pada proses *Handshaking* 2 dan mengirimkan data “11”. Data tersebut berfungsi sebagai informasi bahwa modul komunikasi sudah siap dipilih dan melakukan proses pengiriman melalui intruksi dari core modul. Pada serial monitor modul komunikasi pun ditampilkan (jika diperlukan) proses *Handshaking 3* jika ack dari proses *Handshaking* 2 telah didapatkan *core* modul. Proses *Handshaking* 3 mengirimkan data ”111”.
3. *Core* modul akan menampilkan informasi modul komunikasi telah dipilih dan siap melakukan pengiriman data setelah proses *Handshaking* 2 dan *Handshaking* 3 dilakukan oleh modul komunikasi.
4. Pengujian dilakukan sebanyak 10 kali.

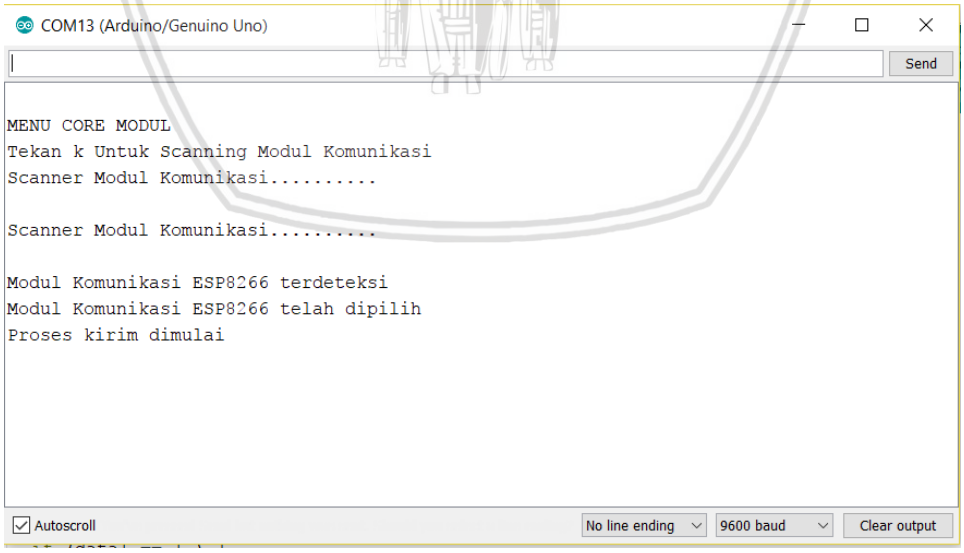
6.2.1.3 Hasil Pengujian

Hasil pengujian ditampilkan berupa pesan modul komunikasi telah dipilih dan telah siap melakukan pengiriman data. Pesan modul komunikasi yang telah dipilih merupakan bantuk ack dari *core* modul terhadap proses *Handshaking* 2 modul komunikasi. Sedangkan pesan modul komunikasi siap melakukan pengiriman sebagai bentuk ack dari proses "*Handshaking*3"modul komunikasi.



Gambar 6.5 Proses *Handshaking* 1-3 modul komunikasi ESP8266

Pada gambar 6.5 ditampilkan *Handshaking* 1-3 dan informasi modul komunikasi sudah siap melakukan pengiriman data dan menunggu data melalui komunikasi serial dari *core* modul.



Gambar 6.6 *Core* modul berhasil memilih modul komunikasi ESP8266

Pada gambar 6.6 ditampilkan informasi "modul komunikasi ESP8266 telah dipilih" di serial monitor sebagai bentuk ack dari proses *Handshaking 2* modul

komunikasi. Setelah proses *Handshaking 2* selesai, core modul pun menampilkan pesan "proses kirim dimulai" pada serial monitor sebagai ack dari proses "*Handshaking*3" modul komunikasi.

Pengujian pemilihan modul komunikasi PnP NRF24l01 ini dilakukan sebanyak 10 kali. Hasil pengujiannya dapat dilihat pada tabel 6.3 berikut:

Tabel 6.3 Tabel hasil percobaan pemilihan modul komunikasi ESP8266

Percobaan ke -	Berhasil terdeteksi	Waktu (mS)
1	Ya	400
2	Ya	400
3	Ya	400
4	Ya	400
5	Ya	400
6	Ya	400
7	Ya	400
8	Ya	400
9	Ya	400
10	Ya	400

6.2.1.4 Analisis pengujian

Hasil pengujian dari 10 kali pemilihan modul komunikasi PnP ESP8266 oleh core modul adalah berhasil sehingga presentase keberhasil mencapai 100%. Waktu yang dibutuhkan untuk pemilihan modul komunikasi rata-rata adalah 0.4 detik (400 mS). Hal tersebut karena pada tiap proses *Handshaking* 2 dan 3 masing-masing di beri *delay* (200 mS) sehingga waktu total untuk proses pemilihan hingga modul komunikasi siap melakukan pengiriman yaitu (400 mS). Waktu tersebut merupakan waktu minimal agar modul komunikasi dan dan core modul dapat saling menerima dan mengirim pesan melalui komunikasi serial (UART).

6.2.2 Pengujian Pemilihan Modul Komunikasi PnP NRF24l01

6.2.2.1 Tujuan pengujian

Tujuan pengujian ini yaitu untuk mengetahui dapatkah modul memilih modul komunikasi PnP NRF24l01 yang telah terdeteksi oleh *core* modul.

6.2.2.2 Prosedur pengujian

Prosedur yang dikerjakan pada pengujian ini yaitu :

1. *Core* modul sudah dalam keadaan aktif dan telah mendeteksi modul komunikasi PnP yang terhubung.
2. Modul komunikasi PnP Esp8266 telah masuk pada proses *Handshaking* 2 dan mengirimkan data "11". Data tersebut berfungsi sebagai informasi bahwa modul komunikasi sudah siap dipilih dan melakukan proses pengiriman melalui intruksi dari core modul. Pada serial monitor modul komunikasi pun ditampilkan (jika diperlukan) proses *Handshaking* 3 jika ack dari proses

Handshaking 2 telah didapatkan *core* modul. Proses *Handshaking* 3 mengirimkan data ”111”.

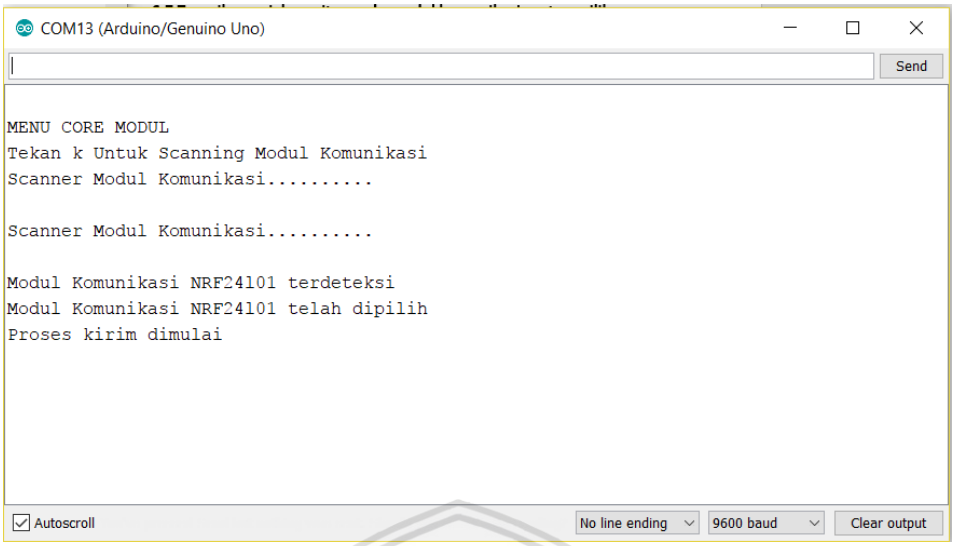
3. *Core* modul akan menampilkan informasi modul komunikasi telah dipilih dan siap melakukan pengiriman data setelah proses “*Handshaking*2” dan *Handshaking* 3 dilakukan oleh modul komunikasi.
4. Pengujian dilakukan sebanyak 10 kali.

6.2.2.3 Hasil pengujian

Hasil pengujian ditampilkan berupa pesan modul komunikasi telah dipilih dan telah siap melakukan pengiriman data pada serial monitor. Pesan modul komunikasi yang telah dipilih merupakan bantuk ack dari *core* modul terhadap proses *Handshaking* 2 modul komunikasi. Sedangkan pesan modul komunikasi siap melakukan pengiriman sebagai bentuk ack dari proses *Handshaking* 3 modul komunikasi.

Gambar 6.7 Proses *Handshaking* 1-3 modul komunikasi NRF24l01

Pada gambar 6.7 ditampilkan *Handshaking* 1-3 dan informasi modul komunikasi sudah siap melakukan pengiriman data dan menunggu data melalui komunikasi serial dari *core* modul.



Gambar 6.8 *Core* modul berhasil memilih modul komunikasi NRF24l01

Gambar 6.8 ditampilkan informasi "Modul komunikasi NRF24l01 telah dipilih" di serial monitor sebagai bentuk ack dari proses *Handshaking 2* modul komunikasi. Setelah proses *Handshaking 2* selesai, core modul pun menampilkan pesan "Proses kirim dimulai" pada serial monitor sebagai ack dari proses *Handshaking* 3 modul komunikasi.

Pengujian pemilihan modul komunikasi PnP NRF24l01 ini dilakukan sebanyak 10 kali. Hasil pengujiannya dapat dilihat pada tabel 6.4 berikut:

Percobaan ke -	Berhasil terdeteksi	Waktu (mS)
1	Ya	400
2	Ya	400
3	Ya	400
4	Ya	400
5	Ya	400
6	Ya	400
7	Ya	400
8	Ya	400
9	Ya	400
10	Ya	400

Tabel 6.4 Hasil percobaan pemilihan modul komunikasi ESP8266

6.2.2.4 Analisis pengujian

Hasil pengujian dari 10 kali pemilihan modul komunikasi PnP NRF24l01 oleh core modul adalah berhasil sehingga presentase keberhasilan mencapai 100%. Waktu yang dibutuhkan untuk pemilihan modul komunikasi rata-rata adalah 0.4 detik (400 mS). Hal tersebut karena pada tiap proses *Handshaking* 2 dan 3 masing-masing di beri *delay* 0.2 detik (200 mS) sehingga waktu total untuk proses pemilihan yaitu (400 mS). Waktu tersebut merupakan waktu minimal agal

modul komunikasi dan dan core modul dapat saling menerima dan mengirim pesan melalui komunikasi serial (UART).

6.3 Pengujian Pengiriman Data Menggunakan Modul Komunikasi PnP ESP8266

6.3.1 Tujuan Pengujian

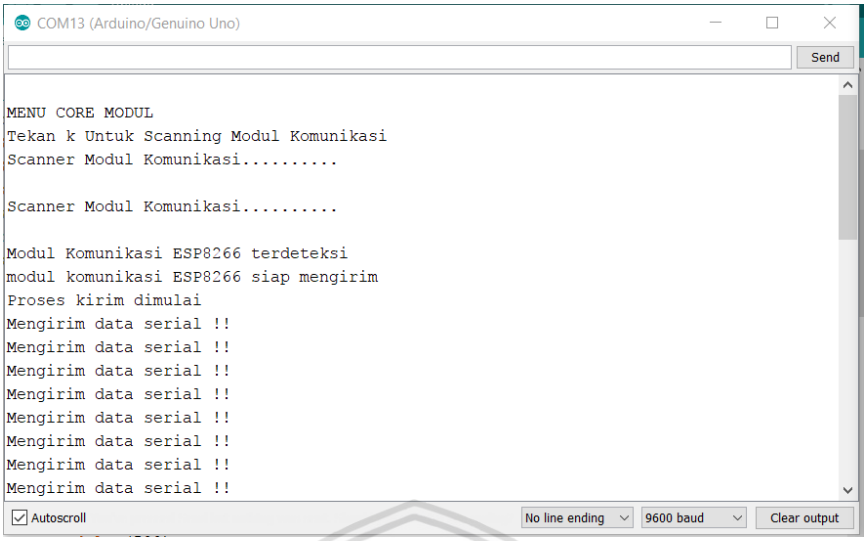
Tujuan pengujian ini yaitu untuk mengetahui kemampuan pengiriman data data dari *core* modul melalui komunikasi serial ke modul komunikasi PnP ESP8266 dan diteruskan ke broker MQTT melalui modul ESP8266 menggunakan protokol MQTT.

6.3.2 Prosedur Pengujian

1. *Core* modul sudah dalam keadaan aktif dan telah memiliki data yang akan dikirimkan ke modul komunikasi PnP ESP8266.
2. Modul komunikasi PnP ESP8266 telah masuk pada proses "*Handshaking*3" dan menunggu data melalui komunikasi serial dari *core* modul.
3. Saat data dikirim *core* modul dan diterima oleh modul komunikasi, fungsi perulangan pengiriman data akan diaktifkan sehingga data akan diterima sebanyak 21 data dari *core* modul. Dan modul komunikasi telah terhubung ke *core* modul.
4. Data yang diterima oleh modul komunikasi akan dikirim ke broker MQTT
5. Data yang dikirim ke Broker MQTT dapat diakses dengan Aplikasi MQTT client dengan menyamakan pengaturan pengiriman MQTT antara modul ESP8266 dan aplikasi MQTT *client*.
6. Pengujian ini dilakukan 1 kali namun pengiriman data dilakukan sebanyak 21 kali dengan 21 data yang berbeda.

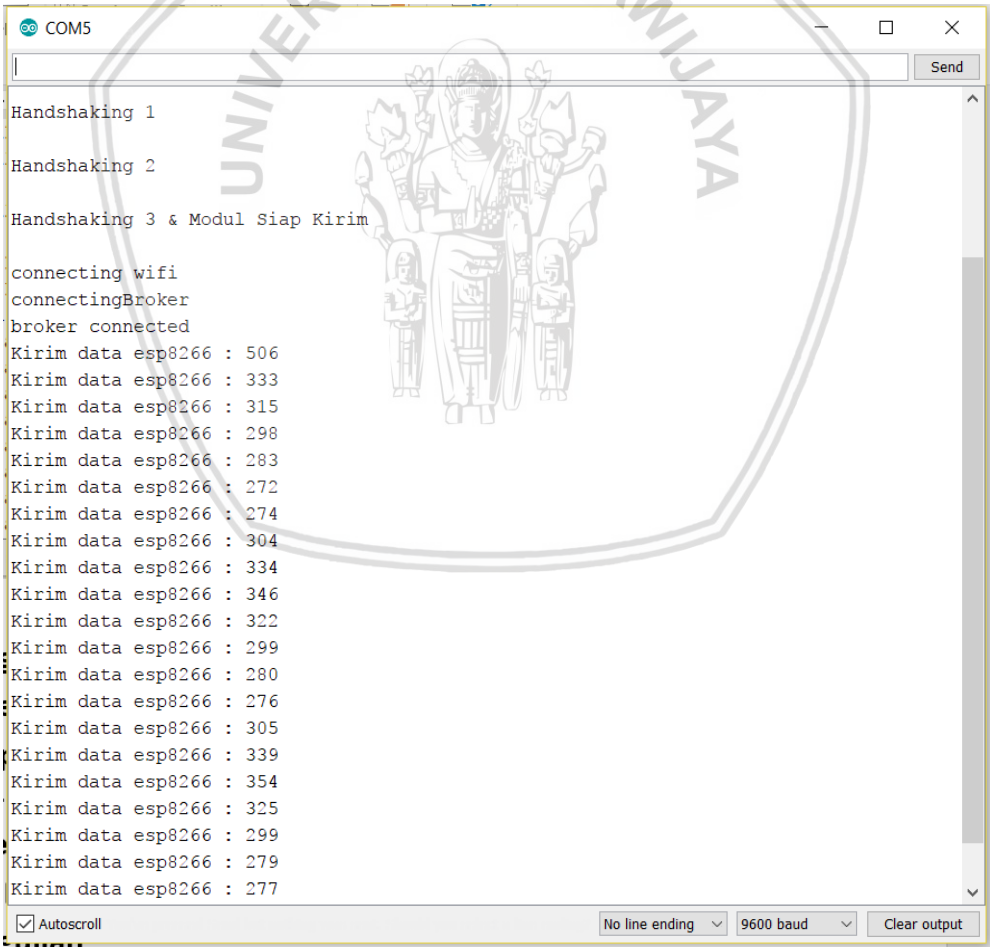
6.3.3 Hasil Pengujian

Hasil pengujian ini menampilkan pesan pada serial monitor pada *core* modul yaitu "Mengirim data serial !!" , data yang diterima oleh modul komunikasi ESP8266 pada serial monitor serta data yang berhasil dikirimkan oleh modul komunikasi diakses melalui aplikasi MQTT *client*.



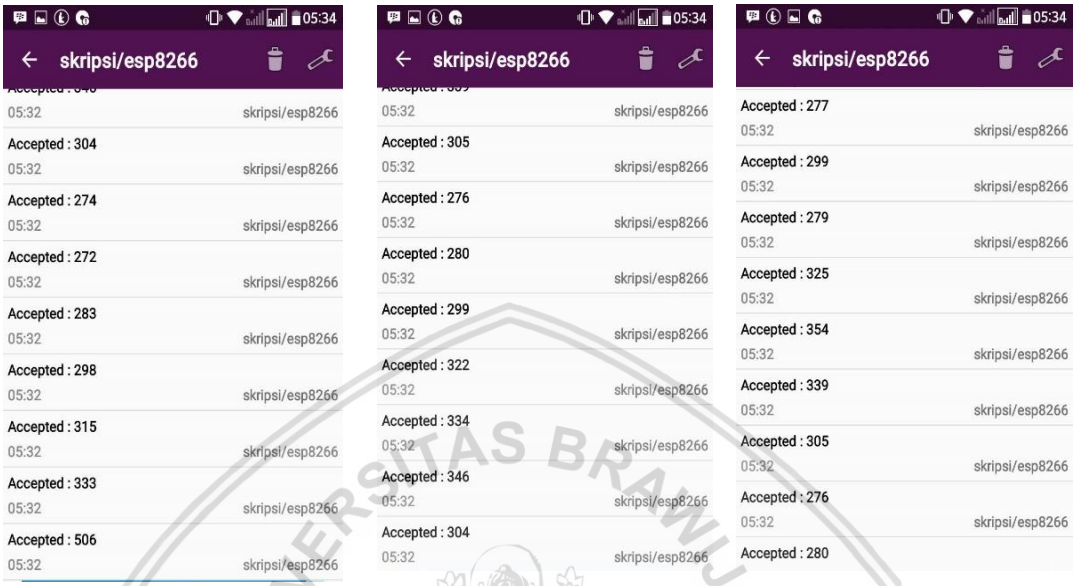
Gambar 6.9 *Core* modul mengirim data serial

Gambar 6.9 ditampilkan pesan "Mengirim data serial !!" pada serial monitor *core* modul saat mengirimkan data ke modul komunikasi setelah ack terhadap proses *Handshaking* 3 modul komunikasi berhasil dilakukan.



Gambar 6.10 Modul komunikasi ESP8266 melakukan pengiriman data

Gambar 6.10 menampilkan pesan "Kirim data esp8266 : " beserta data yang diterima oleh modul komunikasi pada serial monitor. Pesan yang dicetak oleh serial monitor sebanyak 21 kali dengan 21 data berbeda. Data tersebut kemudian diteruskan ke modul ESP8266 dengan komunikasi serial (Rx Tx).



Gambar 6.11 Tampilan aplikasi MQTT *client*

Pada gambar 6.11 terdapat data yang berhasil dikirimkan oleh modul komunikasi PnP ESP8266 ke broker MQTT menggunakan protokol MQTT . Data tersebut diakses menggunakan aplikasi MQTT *client*. Jumlah data yang diterima sesuai dengan data yang ditampilkan pada gambar 6.10 yaitu 21 data yang berbeda.

Pengujian pengiriman data modul komunikasi PnP NRF24l01 ini dilakukan 21 kali. Hasil pengujiannya dapat dilihat pada tabel 6.5 berikut:

Tabel 6.5 Hasil percobaan pengiriman data modul komunikasi ESP8266

Pengiriman data ke -	Data kirim	Data terima	Kecocokan data	Waktu (mS)
1	506	506	Cocok	1100
2	333	333	Cocok	1100
3	315	315	Cocok	1100
4	298	298	Cocok	1100
5	283	283	Cocok	1100
6	272	272	Cocok	1100
7	274	274	Cocok	1100
8	304	304	Cocok	1100
9	334	334	Cocok	1100
10	346	346	Cocok	1100
11	322	322	Cocok	1100
12	299	299	Cocok	1100

Pengiriman data ke -	Data kirim	Data terima	Kecocokan data	Waktu (mS)
13	280	280	Cocok	1100
14	276	276	Cocok	1100
15	305	305	Cocok	1100
16	339	339	Cocok	1100
17	354	354	Cocok	1100
18	325	325	Cocok	1100
19	299	299	Cocok	1100
20	279	279	Cocok	1100
21	277	277	Cocok	1100

6.3.4 Analisis Pengujian

Hasil pengujian dari 21 kali pengiriman data pada modul komunikasi PnP ESP8266 ke broker MQTT adalah berhasil dengan presentase kecocokan/keberhasilan data mencapai 100%. Jumlah data ada yang dikirim oleh modul komunikasi dan diterima oleh aplikasi yaitu 21 dengan nilai yang berbeda-beda untuk memastikan tidak ada pengiriman data yang sama berulang kali. Waktu pengiriman data setiap 1.1 detik (1100 mS) dan diterima oleh aplikasi MQTT *client* pada saat itu juga sehingga pada gambar terlihat waktu data diterima untuk 21 data adalah sama pada pukul 05.32. Hal juga itu diperkuat dengan inisialisai pada program pengiriman data ke Broker MQTT diberi *delay* 1.1 detik (1100 mS) agar data yang dikirimkan sesuai dan tidak terjadi kehilangan data (*data loss*).

6.4 Pengujian Pengiriman Data Menggunakan Modul Komunikasi PnP NRF24l01

6.4.1 Tujuan Pengujian

Tujuan pengujian ini yaitu untuk mengetahui kemampuan pengiriman data data dari *core* modul melalui komunikasi serial ke modul komunikasi PnP NRF24l01 dan diteruskan ke *receiver* NRF24l01 memanfaatkan *library* RF24 pada modul radio NRF24l01.

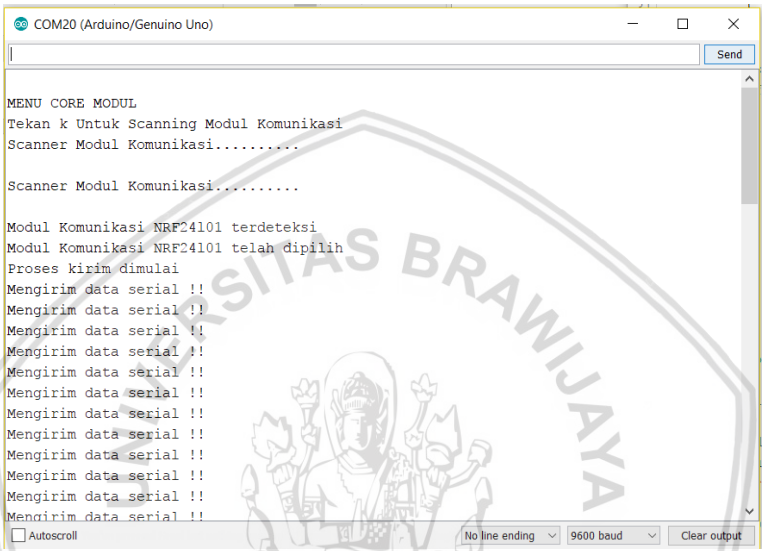
6.4.2 Prosedur Pengujian

1. *Core* modul sudah dalam keadaan aktif dan telah memiliki data yang akan dikirimkan ke modul komunikasi PnP NRF24l01.
2. Modul komunikasi PnP NRF24l01 telah masuk pada proses *Handshaking* 3 dan menunggu data melalui komunikasi serial dari *core* modul.
3. Saat data dikirim *core* modul dan diterima oleh modul komunikasi, fungsi perulangan pengiriman data akan diaktifkan sehingga data akan diterima sebanyak 21 data dari *core* modul.
4. Data yang diterima oleh modul komunikasi akan dikirim ke *receiver* NRF24l01.

5. Data yang dikirim ke *receiver* NRF24l01 dapat dilihat dengan mengakses serial monitornya.
6. Pengujian ini dilakukan 1 kali namun pengiriman data dilakukan sebanyak 21 kali dengan 21 data yang berbeda.

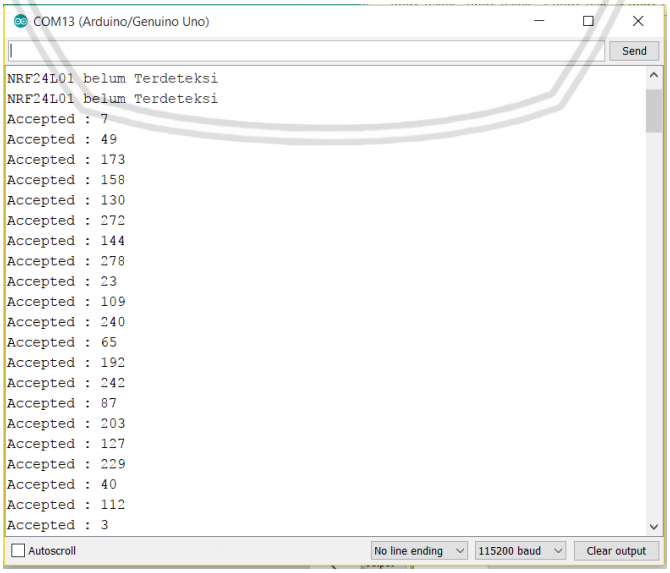
6.4.3 Hasil Pengujian

Hasil pengujian ini menampilkan pesan pada serial monitor pada *core* modul yaitu "Mengirim data serial !!" , data yang diterima oleh modul komunikasi NRF24l01 pada serial monitor serta data yang berhasil dikirimkan oleh modul komunikasi diakses pada serial monitor *receiver* NRF24l01.



Gambar 6.12 Tampilan serial monitor pada *core* modul saat pengiriman data

Pada gambar 6.12 ditampilkan pesan "Mengirim data serial !!" pada serial monitor *core* modul saat mengirimkan data ke modul komunikasi NRF24l01 setelah ack terhadap proses *Handshaking* 3 modul komunikasi berhasil dilakukan.



Gambar 6.13 Tampilan serial monitor *receiver* NRF24l01

Pada gambar 6.13 terdapat data yang berhasil dikirimkan oleh modul komunikasi PnP NRF2l01 pada *receiver* NRF24l01. Awalnya saat tidak terdapat modul komunikasi yang mengirimkan data, serial monitor *receiver* NRF24l01 menampilkan "NRF24l01 belum terdeteksi". Saat modul komunikasi PnP NRF24l01 sudah terdeteksi dan dipilih oleh core modul, maka tampilan *receiver* NRF24l01 akan berupa menjadi "*Accepted* : " beserta data yang dikirim oleh core modul melalui komunikasi serial ke modul komunikasi NRF24l01 lalu diteruskan ke *receiver* NRF24l01 menggunakan komunikasi radio dan memanfaatkan *library* RF24.

Pengujian pengiriman data modul komunikasi PnP NRF24l01 ini dilakukan 21 kali. Hasil pengujiannya dapat dilihat pada tabel 6.6 berikut:

Tabel 6.6 Hasil percobaan pengiriman data modul komunikasi NRF24l01

Pengiriman data ke -	Data kirim	Data terima	Kecocokan data	Waktu (mS)
1	7	7	Cocok	1100
2	49	49	Cocok	1100
3	173	173	Cocok	1100
4	158	158	Cocok	1100
5	130	130	Cocok	1100
6	272	272	Cocok	1100
7	144	144	Cocok	1100
8	278	278	Cocok	1100
9	23	23	Cocok	1100
10	109	109	Cocok	1100
11	240	240	Cocok	1100
12	65	65	Cocok	1100
13	192	192	Cocok	1100
14	242	242	Cocok	1100
15	87	87	Cocok	1100
16	203	203	Cocok	1100
17	127	127	Cocok	1100
18	229	229	Cocok	1100
19	40	40	Cocok	1100
20	112	112	Cocok	1100
21	3	3	Cocok	1100

6.4.4 Analisis Pengujian

Hasil pengujian dari 21 kali pengiriman data pada modul komunikasi PnP NRF24l01 ke *receiver* NRF24l01 adalah berhasil dengan presentase kecocokan/keberhasilan data mencapai 100%. Jumlah data ada yang dikirim oleh modul komunikasi dan diterima oleh aplikasi yaitu 21 dengan nilai yang berbeda-beda untuk memastikan tidak ada pengiriman data yang sama berulang kali. Waktu pengiriman data setiap 1.1 detik (1100 mS) sesuai dengan *delay* yang diberi

pada program modul komunikasi dan *receiver* NRF24l01 agar tidak terjadi kehilangan data (*data loss)* saat pengiriman berlangsung.

BAB 7
PENUTUP

Bab ini berisi penarikan kesimpulan berdasarkan tahapan-tahapan yang telah dikerjakan selama perancangan, implementasi, pengujian, dan analisis. Serta terdapat saran dari penulis agar penelitian ini bisa dilanjutkan dan lebih bermanfaat bagi pembaca.

7.1 Kesimpulan

Berdasarkan perancangan, implementasi, pengujian serta analisis yang dilakukan penulis, maka dapat diambil kesimpulan yaitu hal yaitu :

1. Arsitektur modul komunikasi PnP dapat dibuat dengan menggunakan *chip* ATmega 328p (Arduino nano) dengan modul ESP8266 dan *chip* ATtiny85 dengan modul NRF24l01 menggunakan komunikasi serial (UART).
2. Proses pengenalan modul komunikasi atau *Handshaking* 1 berhasil dilakukan dan membutuhkan waktu minimal 1.5 detik (1500 mS). Hal itu berdasarkan *delay* yang diberikan pada modul komunikasi tersebut. Jika diberi *delay* dibawah 1.5 detik, maka proses pengenalan tidak dapat dilakukan. Sedangkan proses pemilihan hingga modul siap untuk melakukan pengiriman data atau proses *Handshaking* 2 dan *Handshaking* 3 membutuhkan waktu total minimal 0.4 detik. 0.2 detik awal digunakan untuk pemilihan/*Handshaking* 2 dan 0.2 detik lain digunakan untuk kesiapan modul melakukan pengiriman data/*Handshaking* 3.
3. Proses pengiriman data pada modul komunikasi PnP berhasil dilakukan dengan waktu minimal pengiriman untuk modul komunikasi ESP8266 yaitu 1.1 detik (1100 ms) , sedangkan modul komunikasi NRF24l01 membutuhkan waktu minimal 1 detik (1000 ms). Data yang berhasil dikirim modul komunikasi ESP8266 dapat diakses menggunakan aplikasi MQTT *client* dengan menyamakan pengaturan yang ada pada perangkat ESP8266. Untuk modul komunikasi NRF24l01, data yang berhasil terkirim dapat dilihat pada serial monitor *receiver* NRF24l01

7.2 Saran

Adapun saran yang dapat dijadikan acuan untuk perbaikan dan pengembangan penelitian ini maupun penelitian serupa untuk kedepannya adalah sebagai berikut :

1. Untuk arsitektur modul komunikasi PnP ESP8266 penggunaan Arduino nano sebagai perantara komunikasi serial bisa diganti dengan board atau *microchip* lain yang lebih kecil dan *powerfull* agar ukurannya pun bisa lebih proporsional untuk ukuran sebuah modul.

2. Melakukan penelitian lebih lanjut tentang perbandingan komunikasi antar board (micro*chip*) atau modul yang menggunakan komunikasi serial (UART) ataupun komunikasi *Inter Integrated Circuit* (I2C).
3. Dapat terintegrasi dengan aplikasi agar pengguna mudah memilih dan melihat aktifitas modul komunikasi terdeteksi tanpa harus membuka serial monitor *device* lagi.

DAFTAR PUSTAKA

Arduino, 2017. Arduino UNO Rev3. [online] Tersedia di :< http://store.Arduino.cc/> [Diakses 28 Agustus 2017]

Bhatti, N., Dhomeja, L, D., Malkani, Y, A., 2014. *Service Discovery Protocols in Pervasive Computing: A review.* Pakistan [Diakses 25 Agustus 2017]

Ecadio. (2017, December 05). *Berkenalan dengan Arduino Nano*. Retrieved from http://ecadio.com: http://ecadio.com/mengenal-dan-belajar-arduino-nano

Equan , 2015 . *Mengenal MQTT Protokol IoT* . [online] Tersedia di :< https://jsiot.pw/mengenal-mqtt-998b6271f585> [Diakses 02 September 2017

http://www.mobnasesemka.com/category/dunia-telekomunikasi/perangkat-*wireless*/ || diakses tangg 24 November 2017

Leonardo, A., Akbar, S. R. & Maulana, R., 2017. *Implementasi Smart Light Bulb Pervasive Berbasis ESP8266.* Malang

Madakam, S., Ramaswamy, R. And Tripathi, S. 2015 . *Journal of Cumputer and Communications. Internet of Thinks(IoT) : A literature review*. [Diakses 25 Agustus 2017]

Mandalamaya. (2015, February 27). *Pengertian Wireless Dan Standar Jaringan Wireless*. Retrieved from www.mandalamaya.com: https://www.mandalamaya.com/pengertian-wireless-dan-standar-jaringan-wireless/

Micro*chip*, 2017. *Attiny85 in Production*. [online] Tersedia di :< http://micro*chip*.com/> [Diakses 27 Agustus 2017]

Mikrocontroller. (2014, April 18). *NRF24L01 Tutorial*. Retrieved from www.mikrocontroller.net: https://www.mikrocontroller.net/articles/NRF24L01_Tutorial

Nordic Semiconductor, 2017. NRF24L01 Product Spesification v2_0. Katalog. Trondheim

Ridge, R. (2013, January 14). *Archives for : Jaringan Wireless*. Retrieved from http://www.mobnasesemka.com:http://www.mobnasesemka.com/tag/jaringan-wireless/

Sargent. (2016, September 12). *Wireless Untuk Mengukur Suhu Dan Kelembaban Berbasis Arduino Uno R3 Atmega328p*. Retrieved from www.coursehero.com: https://www.coursehero.com/file/p5enjoj/Wireless-Untuk-Mengukur-Suhu-Dan-Kelembaban-Berbasis-Arduino-Uno-R3-Atmega328p/

SeeedStudio, 2017. *Getting Started With ESP8266*. Tersedia di :< http://seedstudio.com>[Diakses 25 Agustus 2017]

Shobrina, U. J., Primananda, R. & Maulana, R., 2016. *Analisis Kinerja Pengiriman Data Modul Transceiver NRF24l01, Xbee dan Wifi ESP8266 Pada Wireless Sensor Network*. Malang

Sonic, 2014 . *Wifi Serial transceiver Module w/ ESP8266*. [online] Tersedia di: <http://www.seeedstudio.com/blog/2014/09/11/getting-started-with-ESP8266/> [Diakses 25 Agustus 2017]

Suprayogi, D, A., Akbar, S, R., Ichsan, M, H, H., 2016. *Implementasi Pervasive Service Discovery Protocol pada Rumah Cerdas Berbasis NRF24L01.* Malang