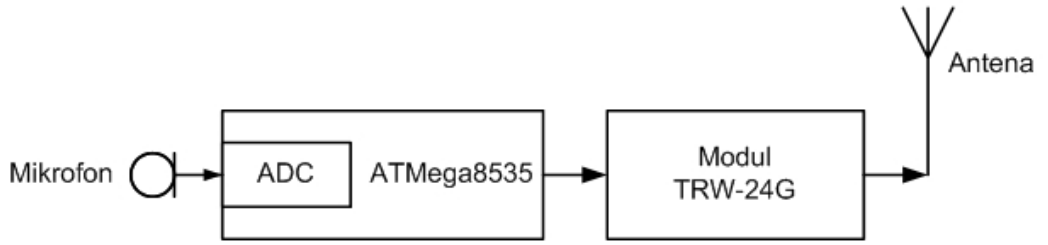
# BAB III
# PERANCANGAN SISTEM

Dalam pembuatan tugas akhir ini akan dirancang sebuah sistem komunikasi suara melalui transmisi gelombang frekuensi radio 2,4 GHz menggunakan modul TRW-2.4G untuk melakukan sistem komunikasi data nirkabel dengan teknik modulasi *Gaussian Frequency Shift Keying* (GFSK). Akan terdapat dua bagian utama yaitu bagian pengirim dan penerima yang masing-masing akan dikendalikan oleh sebuah mikrokontroler ATmega8535 seperti yang diilustrasikan pada Gambar 3.1 berikut.




(a)




(b)

**Gambar 3.1** (a) Diagram Blok Pengirim, (b) Diagram Blok Penerima

Bagian pengirim akan bertanggung jawab untuk melakukan pengubahan sinyal suara yang masih berupa sinyal analog menjadi data-data dalam bentuk digital, melakukan penyimpanan data sementara, dan mengirimkan data-data tersebut ke penerima melalui sebuah protokol tertentu. Sedangkan pada bagian penerima akan bertanggung jawab pada penerimaan data dari pengirim, melakukan penyimpanan data sementara untuk selanjutnya diproses kembali menjadi sinyal analog.

Sinyal suara analog akan diubah menjadi data digital dengan menggunakan perangkat *Analog to Digital Converter* (ADC) dan data digital tersebut akan diubah kembali menjadi sinyal analog dengan menggunakan metode *Pulse Width Modulation* (PWM). Kedua hal tersebut akan dilakukan dengan memanfaatkan fitur yang dimiliki oleh mikrokontroler ATmega8535.

Mikrokontroler ATmega8535 juga akan melakukan pengendalian untuk pengiriman dan penerimaan data pada perangkat TRW-2.4G yang akan mengirimkan atau menerima data digital secara nirkabel dari pengirim ke penerima dengan menggunakan teknik modulasi GFSK.

Secara garis besar, perancangan tugas akhir ini akan terbagi menjadi dua hal yang sangat penting, yaitu:

1. Perancangan perangkat keras meliputi rangkaian mikrofon dan speaker sebagai sarana *Input/Output* (I/O), rangkaian mikrokontroler ATmega8535 sebagai pengendali komunikasi dan pengolahan data I/O, dan rangkaian TRW-2.4G sebagai prasarana untuk melakukan komunikasi nirkabel dengan teknik modulasi GFSK.
2. Perancangan perangkat lunak berupa pembangunan aplikasi pada mikrokontroler dengan menggunakan bahasa pemrograman C untuk mengendalikan proses kerja dari keseluruhan sistem.

## 3.1. Perancangan Perangkat Keras

Pada perancangan perangkat keras ini terdiri dari 3 sub sistem yaitu perancangan rangkaian mikrofon dan speaker, perancangan mikrokontroler ATmega8535, dan perancangan rangkaian TRW-2.4G, yang akan diimplementasikan pada bagian pengirim dan penerima. Rangkaian mikrofon dan speaker akan berfungsi untuk melakukan *filtering* dan penguatan terhadap sinyal analog suara sebelum diproses pada mikrokontroler ATmega8535, dan juga akan melakukan *filtering* dan penguatan terhadap sinyal output dari mikrokontroler ATmega8535.

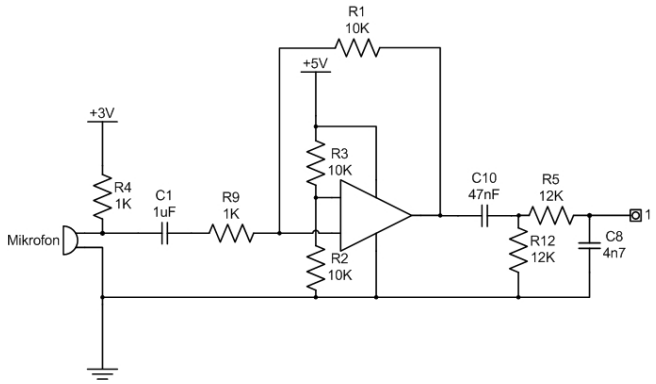
Rangkaian mikrokontroler ATmega8535 akan berfungsi sebagai sebuah prasyarat awal agar mikrokontroler ATmega8535 dapat bekerja dengan baik. Mikrokontroler ATmega8535 akan melakukan proses pengubahan sinyal analog

suara kedalam bentuk data-data digital, melakukan penyimpanan sementara data kontinu hasil dari pengubahan ADC, melakukan pengendalian terhadap TRW-2.4G untuk melakukan komunikasi data, dan melakukan pengubahan data-data digital yang diterima menjadi sinyal analog kembali.

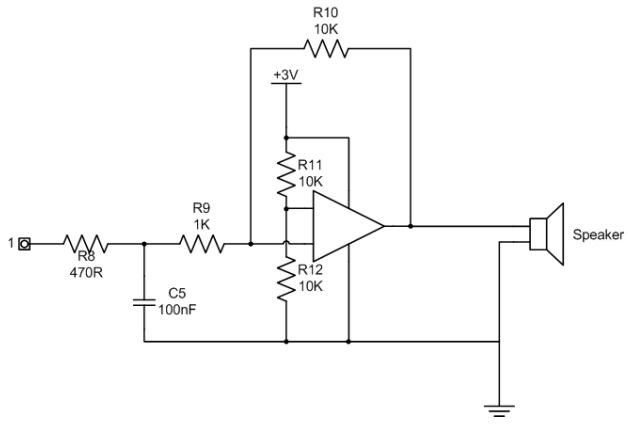
TRW-2.4G merupakan perangkat yang akan digunakan oleh mikrokontroler ATmega8535 untuk melakukan komunikasi data nirkabel dengan menggunakan teknik modulasi GFSK. Rangkaian TRW-2.4G merupakan rangkaian antarmuka antara TRW-2.4G dan mikrokontroler ATmega8535 sehingga TRW-2.4G dapat diatur untuk bekerja sesuai dengan kebutuhan.

### 3.1.1. Rangkaian Mikrofon dan Speaker

Rangkaian mikrofon dan speaker akan melakukan *filtering* terhadap sinyal analog suara baik dari mikrofon sebelum diproses oleh mikrokontroler ATmega8535 maupun sebelum dikeluarkan pada sebuah speaker.




(a)




(b)

**Gambar 3.2** (a) Rangkaian Mikrofon, (b) Rangkaian Speaker

*Filtering* akan menggunakan rangkaian *band-pass filter* pasif dengan lebar frekuensi sebesar 3000 Hz. Meskipun batas frekuensi minimum dan maksimum yang dapat didengar oleh telinga manusia adalah sekitar 20 Hz hingga 20 kHz, tapi pada aplikasi ini besar frekuensi yang akan diloloskan hanya pada rentang 300 Hz hingga 3000 Hz.

Rangkaian *band-pass filter* pasif yang digunakan merupakan penggabungan antara *high-pass filter* dan *low-pass filter*. Untuk dapat meloloskan frekuensi antara 300 Hz hingga 3000 Hz akan digunakan rangkaian RC filter dengan menggunakan Persamaan (2.12) dan (2.13). Berdasarkan pada persamaan tersebut, maka besar frekuensi *cut-off* yang akan diloloskan oleh rangkaian *band-pass filter* tersebut antara lain:

1. Frekuensi *cut-off* pada rangkaian *high-pass filter* pasif:

$$f_c = \frac{1}{2\pi RC} \tag{3.3}$$

$$f_c = \frac{1}{2\pi \cdot 12 \cdot 10^3 \cdot 47 \cdot 10^{-9}}$$

$$f_c = \frac{1}{3{,}5437 \cdot 10^{-3}}$$

$$f_c = 282{,}1896 \text{ Hz}$$

Besar frekuensi *cut-off* yang akan diloloskan oleh *high-pass filter* yaitu diatas 282,1896 Hz.

2. Frekuensi cut-off pada rangkaian *low-pass filter* pasif:

$$f_c = \frac{1}{2\pi \cdot 12 \cdot 10^3 \cdot 4{,}7 \cdot 10^{-9}}$$

$$f_c = \frac{1}{3{,}5437 \cdot 10^{-4}}$$

$$f_c = 2821{,}896 \text{ Hz}$$

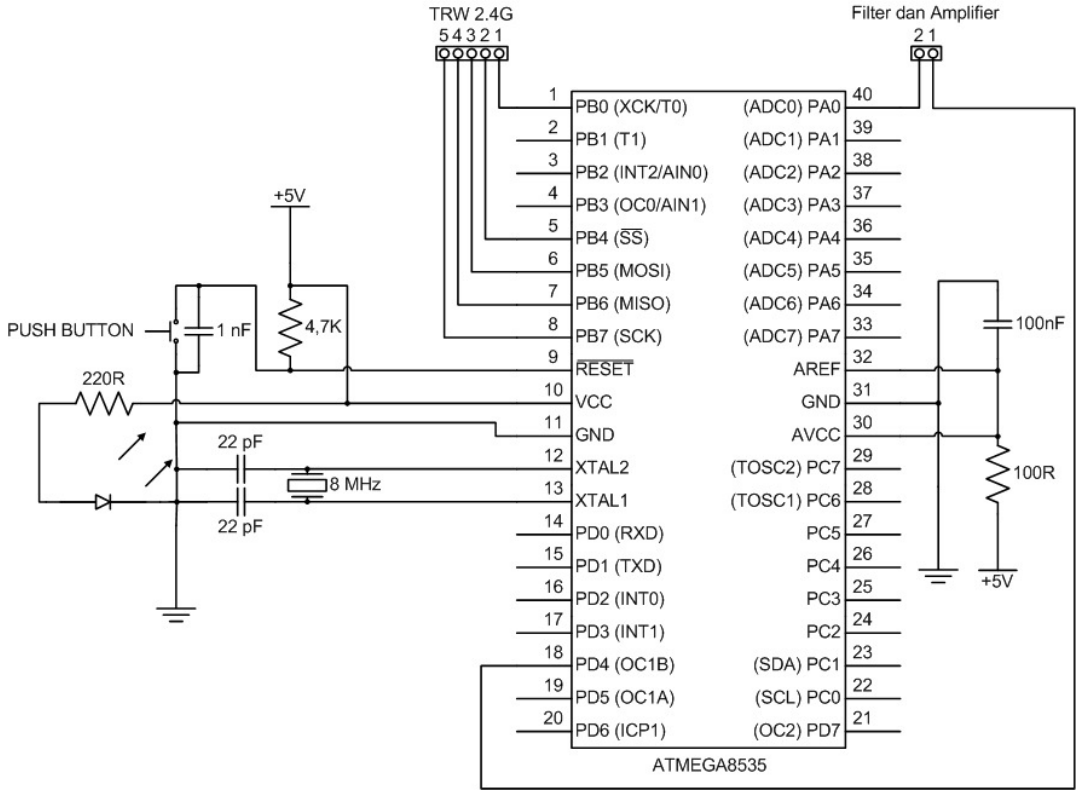
Besar frekuensi yang akan diloloskan oleh *low-pass filter* yaitu dibawah 2821,896 Hz.

Maka, besar frekuensi yang akan diloloskan oleh band-pass filter yaitu antara 282,1896 Hz hingga 2821,896 Hz.

### 3.1.2. Rangkaian Mikrokontroler ATmega8535

Jenis mikrokontroler yang akan digunakan dalam tugas akhir ini adalah mikrokontroler keluarga ATmega yang diproduksi oleh Atmel, yaitu ATmega8535. ATmega8535 telah dilengkapi dengan fitur-fitur yang dapat dimanfaatkan dalam pembangunan aplikasi pada tugas akkhir ini, antara lain:

1. Memiliki fasilitas ADC internal yang dapat digunakan dengan kecepatan konversi data sebesar 125µs pada resolusi bit sebanyak 8-bit.
2. Memiliki timer yang telah dilengkapi dengan kemampuan untuk membangkitkan sinyal PWM.
3. Memiliki 3 buah timer (2 timer 8-bit dan 1 timer 16-bit) sehingga selain untuk membangkitkan sinyal PWM, masih terdapat timer lain untuk membuat slot-slot waktu pada kepentingan pembangunan perangkat lunak dari aplikasi tugas akhir ini.




**Gambar 3.3** Rangkaian Mikrokontroler ATMega8535

Rangkaian mikrokontroler ATmega8535 pada tugas akhir ini akan bekerja pada sistem *clock* sebesar 8 MHz. Sistem *clock* tersebut dibangkitkan oleh sebuah

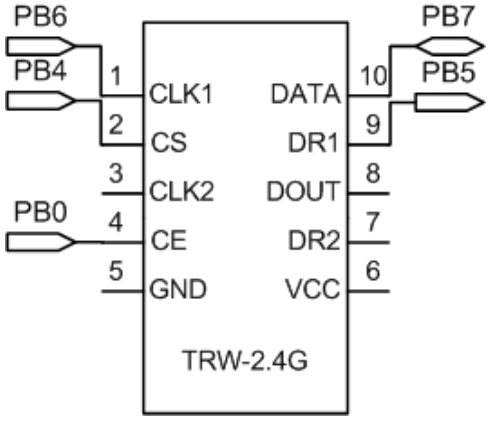
kristal osilator 8 MHz yang dilengkapi dengan dua buah *decoupling capacitor* masing-masing sebesar 22 pF untuk menstabilkan frekuensi *clock*. Pin AVCC, *analog voltage*, dihubungkan ke sumber tegangan VCC melalui sebuah *low-pass filter* pasif. AVCC akan digunakan sebagai tegangan referensi pada ADC internal. Low-pass filter yang dihubungkan pada AVCC bertujuan agar *ripple* supply analog pada pin AVCC kecil sehingga ADC internal akan memiliki tingkat presisi yang tinggi.

Pada rangkaian mikrokontroler ATmega8535, PORTB akan digunakan sebagai port antarmuka antara ATmega8535 dengan TRW-2.4G untuk mengendalikan kerja TRW-2.4G sebanyak 5 pin I/O. PORTA akan digunakan sebagai input ADC (1 pin I/O) dan PORTD sebagai output PWM (1 pin I/O).

Jenis ATmega8535 yang akan digunakan pada tugas akhir ini adalah ATmega8535L. ATmega8535L dapat beroperasi pada tegangan 2,7 – 5,5V dengan kecepatan *clock* maksimum sebesar 8 MHz. Hal ini ditujukan agar mikrokontroler ATmega8535L dapat bekerja pada level tegangan yang sama dengan Wenshing TRW-2.4G yaitu pada level tegangan 3V.

### 3.1.3. Rangkaian TRW-2.4G

TRW-2.4G merupakan modul transceiver komunikasi data nirkabel yang bekerja dengan teknik modulasi GFSK. TRW-2.4G harus dikendalikan oleh sebuah mikrokontroler agar dapat bekerja. Level tegangan yang dibutuhkan agar TRW-2.4G dapat bekerja dengan baik yaitu antara 1,9 – 3,6V.




**Gambar 3.4** Rangkaian TRW-2.4G

Pada rangkaian ini, TRW-2.4G dihubungkan secara langsung ke pin I/O pada mikrokontroler karena mikrokontroler ATmega8535L dan TRW-2.4G bekerja pada level tegangan yang sama, yaitu 3V. Aplikasi tugas akhir ini hanya akan menggunakan satu *channel* penerimaan data yaitu *channel* 1, sehingga hanya pin-pin yang bersesuaian dengan *channel* 1 yang akan dihubungkan ke pin I/O pada mikrokontroler ATmega8535L.

### 3.2. Perancangan Perangkat Lunak

Perangkat lunak pada tugas akhir ini dibangun dengan menggunakan aplikasi pembangunan perangkat lunak Eclipse yang merupakan sebuah IDE (*Integrated Development Environment*) untuk mengembangkan perangkat lunak dan dapat dijalankan di semua platform sistem operasi (*platform-independent*). Kompilasi perangkat lunak pada mikrokontroler akan dibangun dengan menggunakan *compiler* AVR-GCC yang menggunakan bahasa pemrograman C.

Perancangan perangkat lunak ditujukan agar mikrokontroler ATmega8535L dapat bekerja dengan baik dalam mengendalikan perangkat-perangkat baik perangkat internal maupun perangkat eksternal dari mikrokontroler ATmega8535L seperti ADC, pembangkitan sinyal PWM dengan menggunakan Timer 1, Timer 0 untuk pembentukan slot waktu, dan konfigurasi TRW-2.4G untuk melakukan pengiriman atau penerimaan data nirkabel.

Sinyal suara analog memiliki rentang frekuensi hingga 4 kHz. Sehingga, berdasarkan Nyquist, maka pada rentang frekuensi 4 kHz akan dibutuhkan kecepatan frekuensi sampling sebesar 8 kHz dengan waktu sampling sebesar 125μs. Proses digitalisasi sinyal suara membutuhkan resolusi bit minimal sebesar 8-bit.

Perangkat lunak pada aplikasi dalam tugas akhir ini akan didefinisikan sebagai berikut:

1. Frekuensi ADC *sampling* dan PWM *update* (*fs*) adalah 8 kHz dengan resolusi sebesar 8-bit.
2. Pewaktuan pada perangkat lunak akan membentuk slot-slot waktu yang memiliki besar frekuensi 8 kHz.
3. Sebuah frame data tersusun atas 12 slot waktu sebesar 125μs.

4. Sebuah paket dari 12 sampel data akan ditransmisikan pada tiap frame (1,5 ms).

Perangkat lunak pada aplikasi dalam tugas akhir ini dibangun dengan menggunakan sistem penjadwalan yang diimplementasikan pada mikrokontroler ATmega8535L. Secara umum perangkat lunak akan dibagi menjadi dua bagian penting, yaitu sebuah *interrupt service routine* (ISR) dan *loop* program utama. ISR akan dipanggil pada tiap 125μs slot waktu.

### 3.2.1. Perancangan Konfigurasi Slot Waktu 125 μs

Slot waktu 125 μs pada aplikasi dalam tugas akhir ini ditujukan agar konversi data dari ADC ATmega8535L dan pembangkitan sinyal PWM memiliki kecepatan yang sama. Slot waktu 125 μs dibangun dengan memanfaatkan fasilitas Timer 0 8-bit pada ATmega8535L.

Konfigurasi Timer 0 untuk membangun slot waktu 125 μs ditentukan oleh tiga register pengendali yaitu TCCR0, OCR0, dan TIMSK. Register TCCR0 digunakan untuk mengatur besar *prescaler* dan mode operasi yang digunakan. Pada aplikasi dalam tugas akhir ini besar *prescaler* yang digunakan pada Timer0 adalah 8 (CS01 = 1) dan dengan mode operasi Clear Timer on Compare (CTC) yang diaktifkan dengan mengaktifkan register WGM01 (WGM01 = 1). Pada mode ini counter akan menghitung mundur dengan kecepatan 1 μs. Nilai tertinggi pada mode CTC ditentukan oleh nilai pada register OCR0. Sehingga untuk mendapatkan *counter* sebesar 125 μs maka nilai register OCR0 adalah 125.

Potongan kode program inisialisasi Timer 0 untuk membangun slot waktu 125 μs adalah sebagai berikut:

```
// Inisialisasi Timer 0
TCCR0 = (1<<WGM01) | (1<<CS01);
OCR0 = 125;
TIMSK = (1<<OCIE0);
```

Register TIMSK digunakan untuk mengatur aktif atau tidaknya *interrupt* sesuai dengan konfigurasi pada TCCR0. *Interrupt* akan diaktifkan ketika *Compare Match* terjadi antara Timer0 dan OCR0 (OCF0 = 1). Setelah *interrupt* pada register TIMSK diaktifkan, maka *global interrupt* juga harus diaktifkan sehingga ISR dapat dieksekusi sesuai dengan waktu diinginkan yaitu 125 μs.

### 3.2.2. Perancangan Konfigurasi ADC Internal ATmega8535L

Perangkat ADC internal ATmega8535L akan digunakan untuk melakukan sampling terhadap sinyal input suara analog. Pada aplikasi dalam tugas akhir ini, data sampel hasil konversi ADC hanya akan diambil sesuai dengan slot waktu 125 μs atau dengan kecepatan frekuensi *sampling* sebesar 8 kHz dan resolusi bit untuk merepresentasikan sinyal analog sebesar 8-bit.

Konfigurasi ADC pada mikrokontroler ATmega8535L ditentukan oleh tiga register pengendali yaitu ADMUX, ADCSRA, dan ADC Data Register. Register ADMUX digunakan untuk mengatur tegangan referensi yang digunakan, format data sampel hasil konversi, dan *channel* ADC yang digunakan. Pada aplikasi dalam tugas akhir ini, ADC akan menggunakan tegangan referensi dari pin AREF (REFS1 = 0; REFS0 = 0) dan *channel* input ADC0 pada pin I/O PORTA.0 (MUX3..0 = 0000). Karena besar resolusi bit hasil sampling ADC yang dibutuhkan hanya 8-bit, maka register ADLAR akan diaktifkan (ADLAR = 1) untuk mempermudah proses pembacaan data hasil konversi ADC.

Register ADCSRA digunakan untuk mengatur kendali dan status ADC. Pada aplikasi dalam tugas akhir ini, ADC akan diatur pada kecepatan frekuensi *clock* sebesar 250 kHz (ADPS2..0 = 101). Kecepatan konversi ADC diatur dengan menentukan besar *prescaler* ADC untuk menentukan kecepatan *clock* ADC. ADC diaktifkan dan langsung memulai konversi (ADEN = 1; ADSC = 1).

Potongan kode program inisialisasi ADC untuk melakukan sampling terhadap sinyal input suara analog adalah sebagai berikut:

```
// Inisialisasi ADC
ADMUX = (0<<REFS1) | (0<<REFS0) | (1<<ADLAR);
ADCSRA = ADCSRA | (1<<ADEN) | (1<<ADSC) | (1<<ADPS2) |
(0<<ADPS1) | (1<<ADPS0);
```

Hasil konversi ADC akan dibaca pada tiap slot waktu 125 μs untuk mendapatkan kecepatan frekuensi *sampling* sebesar 8 kHz. Pembacaan nilai hasil konversi ADC dilakukan dengan membaca nilai pada register ADCH sebesar 8-bit.

### 3.2.3. Perancangan Konfigurasi Pembangkitan Sinyal PWM

Pembangkitan sinyal PWM pada aplikasi dalam tugas akhir ini dibangun dengan memanfaatkan fasilitas Timer 1 16-bit pada ATmega8535L. Untuk pembangkitan sinyal PWM, Timer 1 akan menggunakan port PWM output OCR1B (PORTD.4).

Konfigurasi pada Timer 1 dikendalikan oleh register TCCR1A dan TCCR1B. Pembangkitan sinyal PWM Timer 1 dioperasikan pada mode PWM, *Phase Correct*, 8-bit. Mode ini dioperasikan dengan mengaktifkan bit WGM10 pada register TCCR1A (WGM10 = 1). Besar *prescaler* yang digunakan pada Timer 1 yaitu 1 atau tanpa *prescaler* sehingga PWM dapat beroperasi pada frekuensi tertinggi yang paling mungkin. *Prescaler* diatur dengan mengaktifkan bit CS10 pada register TCCR1B (CS10 = 1).

Potongan kode program inisialisasi Timer 1 untuk membangkitkan sinyal PWM adalah sebagai berikut:

```
// Inisialisasi Timer 1 PWM
TCCR1A = 0x21;
TCCR1B = 0x01;
OCR1B = 0x00;
```

Proses *update* nilai register OCR1B akan dilakukan sesuai dengan besar slot waktu yang telah ditentukan oleh slot waktu pada Timer 0 yaitu 125 μs atau 8 kHz.

### 3.2.4. Perancangan Konfigurasi TRW-2.4G

Sistem komunikasi yang digunakan antara TRW-2.4G dan mikrokontroler ATmega8535L merupakan sistem komunikasi serial. Komunikasi serial ini dikendalikan sepenuhnya menggunakan perangkat lunak pada mikrokontroler ATmega8535L.

Metode yang digunakan untuk melakukan komunikasi serial antara TRW-2.4G dan ATmega8535L disebut dengan metode *bit-banging*. *Bit-banging* merupakan metode untuk melakukan komunikasi serial dengan sepenuhnya menggunakan perangkat lunak pada sebuah sistem mikrokontroler. Pada aplikasi dalam tugas akhir ini, metode *bit-banging* akan digunakan untuk melakukan

penulisan bit ke register TRW-2.4G dan juga pembacaan untuk tiap 8-bit data dari register TRW-2.4G.

Fungsi untuk melakukan penulisan dan pembacaan 8-bit data pada mikrokontroler ATmega8535L adalah sebagai berikut:

1. Fungsi untuk melakukan penulisan 8-bit data ke register TRW-2.4G

```
static void trw24g_send_byte(uint8_t byte)
{
    uint8_t i;
    for (i = 0x80; i; i >>= 1)
      {
        if (byte & i)
          {
            trw24g_set_data1_high();
          }
        else
          {
            trw24g_set_data1_low();
          }
        delay_500ns();
        trw24g_set_clk1_high();
        delay_500ns();
        trw24g_set_clk1_low();
      }
}
```

Pada fungsi tersebut tiap bit dalam sebuah 8-bit data yang akan ditulis ke register TRW-2.4G diperiksa, dimulai dari MSB. Jika bit bernilai 1, maka fungsi akan mengaktifkan pin output pada kondisi high (fungsi `trw24g_set_data1_high()`). Jika bit bernilai 0, maka fungsi akan mengaktifkan pin output pada kondisi low (fungsi `trw24g_set_data1_low()`). Fungsi `trw24g_set_clk1_high()` dan `trw24g_set_clk1_low()` memberikan pulsa *clock* untuk mengindikasikan bahwa nilai bit harus dibaca oleh TRW-2.4G. Fungsi `delay_500ns()` merupakan fungsi untuk menciptakan waktu tunda sebesar 500ns sesuai dengan petunjuk di dalam datasheet TRW-2.4G. Kecepatan penulisan tiap bit menggunakan fungsi ini adalah 1 μs atau 1 Mbps.

2. Fungsi untuk melakukan pembacaan 8-bit data dari register TRW-2.4G

```
static uint8_t trw24g_recv_byte()
{
    uint8_t ret = 0;
    uint8_t i;
    for (i = 0x80; i; i >>= 1)
      {
        delay_500ns();
        trw24g_set_clk1_high();
        delay_500ns();
        if (trw24g_data1_is_high())
          {
            ret |= i;
          }
        trw24g_set_clk1_low();
      }
    return ret;
}
```

Pada fungsi tersebut nilai bit dalam sebuah 8-bit data pada register TRW-2.4G dibaca, dimulai dari MSB, dengan melakukan *polling* data pada pin input mikrokontroler ATmega8535L (fungsi `trw24g_data1_is_high()`). Jika pin input bernilai 1, maka fungsi akan melakukan operasi OR antara variable `ret` dengan indeks `i`. Jika pin input bernilai 0, maka fungsi akan membiarkan data tersebut bernilai 0 dan melakukan pergeseran indeks `i`. Hasil pembacaan keseluruhan 8-bit data pada register TRW-2.4G akan disimpan ke dalam variable `ret`.

Secara umum, TRW-2.4G memiliki 3 mode operasi utama yang dapat diatur berdasarkan 3 pin kontrol. Mode operasi tersebut antara lain:

**Tabel 3.1** *Mode Operasi Wenshing TRW-2.4G*

| Mode | PWR_UP | CE | CS |
|---|---|---|---|
| Active (RX/TX) | 1 | 1 | 0 |
| Configuration | 1 | 0 | 1 |
| Stand By | 1 | 0 | 0 |

Mode *configuration* harus diaktifkan untuk dapat melakukan konfigurasi TRW-2.4G. TRW-2.4G membutuhkan byte-byte konfigurasi sebesar 15 byte yang akan disimpan di dalam sebuah *configuration register* pada TRW-2.4G.

Definisi konfigurasi TRW-2.4G yang akan digunakan pada aplikasi dalam tugas akhir ini antara lain:

1. Besar *payload* untuk tiap paket data RF adalah 96-bit.
2. Besar pengalamatan yang digunakan pada buffer TRW-2.4G adalah 24-bit dimulai dari alamat `0x030405`.
3. Menggunakan 16-bit CRC untuk menjamin integritas data.
4. Kecepatan transfer data yang digunakan pada TRW-2.4G adalah 1 Mbps dengan mode operasi Shockburst™ pada *channel* 204 (menggunakan frekuensi 2604 MHz).

Setelah TRW-2.4G selesai dikonfigurasikan sesuai dengan definisi diatas, cukup 1 bit RXEN yang digunakan untuk mengatur TRW-2.4G pada mode pengirim atau penerima.

Pengiriman dan penerimaan paket melalui TRW-2.4G dengan mode operasi Shockburst™ harus melalui beberapa protokol yang telah ditentukan oleh datasheet TRW-2.4G.

Proses yang harus dilakukan untuk mengirimkan paket data melalui TRW-2.4G antara lain:

1. Mengatur konfigurasi TRW-2.4G pada mode pengirim.
2. Mengaktifkan kondisi pin CE dalam keadaan high (`trw24g_set_ce_high()`) yang selanjutnya akan mengaktifkan *on-board data processing* pada TRW-2.4G.
3. Melakukan clocking-in alamat penerimaan data dan data ke TRW-2.4G dengan menggunakan metode *bit-banging*.
4. Mikrokontroler menonaktifkan kondisi pin CE dalam keadaan low (`trw24g_set_ce_low()`) untuk mengaktifkan transmisi TRW-2.4G pada mode Shockburst™.

Proses penerimaan paket data dari TRW-2.4G antara lain:

1. Mengatur konfigurasi TRW-2.4G pada mode penerima.
2. Mengaktifkan kondisi pin CE dalam keadaan high (`trw24g_set_ce_high()`) untuk mengaktifkan mode penerima.
3. TRW-2.4G akan mengaktifkan pin DR1 untuk menandakan bahwa data pada *buffer* TRW-2.4G telah siap untuk dikeluarkan.

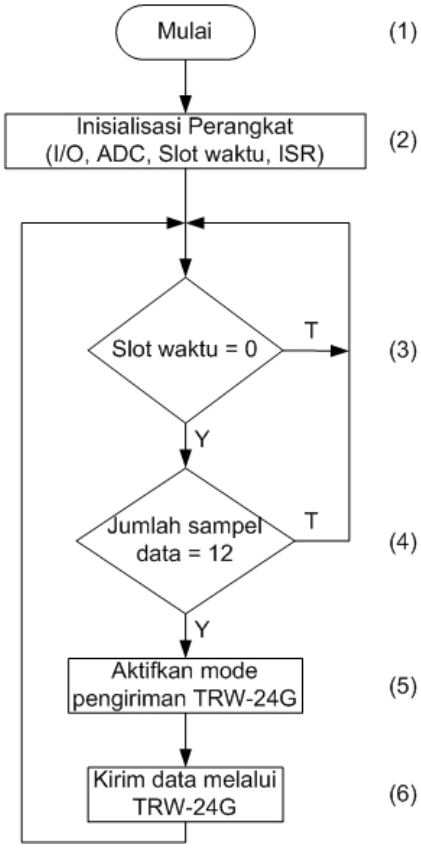
4. Mikrokontroler mengeluarkan data dari buffer TRW-2.4G dan menonaktifkan kondisi pin CE dalam keadaan low (`trw24g_set_ce_low()`).

### 3.2.5. Perancangan Alur Pengiriman Paket Data

Perancangan alur pengiriman paket data ditujukan untuk membangun perangkat lunak pada sisi pengirim. Ketika pengirim dinyalakan, perangkat lunak akan langsung mengeksekusi perintah yang terdapat pada *loop* program utama dan perintah yang terdapat di dalam ISR. Diagram alir loop program utama diilustrasikan pada Gambar 3.5.

Secara umum, proses yang akan dijalankan di dalam *loop* pada program utama pada pengirim antara lain:

1. Menunggu hingga slot waktu ke 0.
2. Jika jumlah sampel telah siap, aplikasi akan memanggil fungsi pengaktifan mode pengirim dan mengirimkan sebuah paket data.




**Gambar 3.5** Diagram Alir *Loop* Program Utama Pengirim

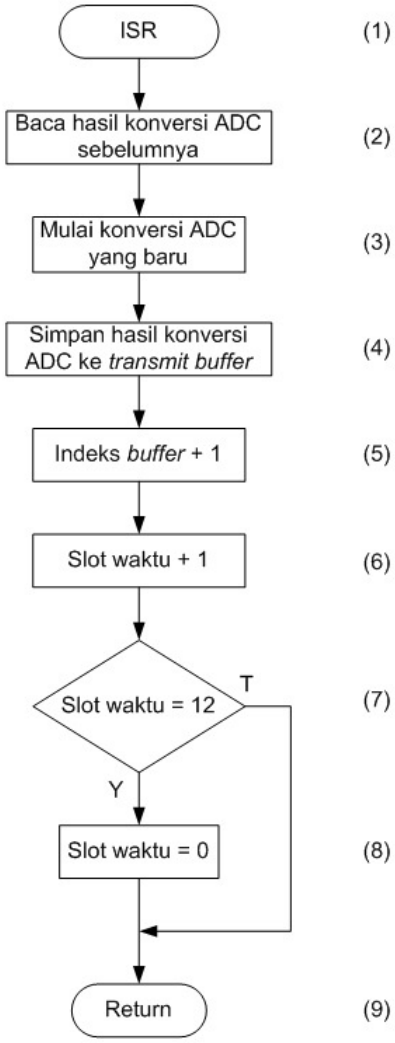
Keterangan:

1. Awal program.
2. Inisialisasi atau konfigurasi untuk perangkat-perangkat internal mikrokontroler ATmega8535L dan TRW-2.4G. Pada bagian ini, program akan melakukan konfigurasi port I/O, ADC internal ATmega8535L, Timer 0 sebagai slot waktu, TRW-2.4G,dan global *interrupt*.
3. Program akan memeriksa apakah *counter* slot waktu bernilai 0. Jika *counter* slot waktu bernilai 0, maka program akan melakukan eksekusi terhadap proses selanjutnya. Jika *counter* slot waktu tidak bernilai 0, maka program akan melakukan proses perulangan hingga *counter* slot waktu bernilai 0.
4. Pada langkah ini, program akan melakukan pemeriksaan terhadap jumlah paket data yang telah dimiliki. Program hanya akan mengeksekusi proses selanjutnya jika nilai *counter* untuk jumlah sampel data adalah 12. Nilai 12 pada *counter* jumlah sampel data menandakan bahwa *transmit buffer* telah memiliki data dan siap dikirim.
5. Jika *counter* untuk jumlah sampel data telah bernilai 12, maka program akan melakukan konfigurasi terhadap TRW-2.4G untuk segera mengaktifkan mode pengiriman data.
6. Setelah mode pengiriman data pada TRW-2.4G telah aktif, maka program akan mengirimkan data yang ada pada *transmit buffer* ke dalam *buffer* pada TRW-2.4G dan memerintahkan TRW-2.4G untuk segera melakukan pengiriman data nirkabel ke penerima yang telah siap untuk menerima paket data.

Proses yang akan dilakukan oleh ISR pada pengirim antara lain:

1. Membaca satu sampel dari ADC dan menyimpan hasil pembacaan ADC ke *transmit buffer*.
2. Memulai konversi ADC.
3. Melakukan *update counter* slot waktu.

Diagram alir untuk proses yang dilakukan oleh ISR pada pengirim diilustrasikan pada Gambar 3.6.




**Gambar 3.6** Diagram Alir ISR Pengirim

Keterangan:

1. Awal rutin ISR. ISR akan dieksekusi tiap 125µs, yaitu ketika terjadi *overflow interrupt* pada Timer 0.
2. ISR melakukan pembacaan hasil konversi ADC dari konversi ADC sebelumnya.
3. ISR memberikan perintah kepada ADC untuk melakukan konversi ADC yang baru.
4. Melakukan penyimpanan hasil pembacaan konversi ADC ke dalam *transmit*

*buffer*.

5. *Increment* indeks *transmit buffer*.
6. *Increment counter* slot waktu.
7. Melakukan pemeriksaan terhadap nilai counter slot waktu. Jika *counter* slot waktu bernilai 12, maka ISR akan mengeksekusi perintah selanjutnya. Jika tidak, maka ISR akan melakukan *increment* terhadap *counter* slot waktu pada *interrupt* selanjutnya.
8. ISR memberikan counter slot waktu dengan nilai 0.
9. Proses dalam ISR telah selesai dieksekusi, dan kontrol akan dikembalikan pada *loop* program utama hingga *interrupt* selanjutnya.

Berdasarkan dua fungsi utama pada perancangan alur pengiriman paket data diatas maka alur pengiriman paket data dapat diilustrasikan pada Gambar 3.7.

| Slot waktu (125µs) | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Pengirim | TX | | | | | | | | | | | |

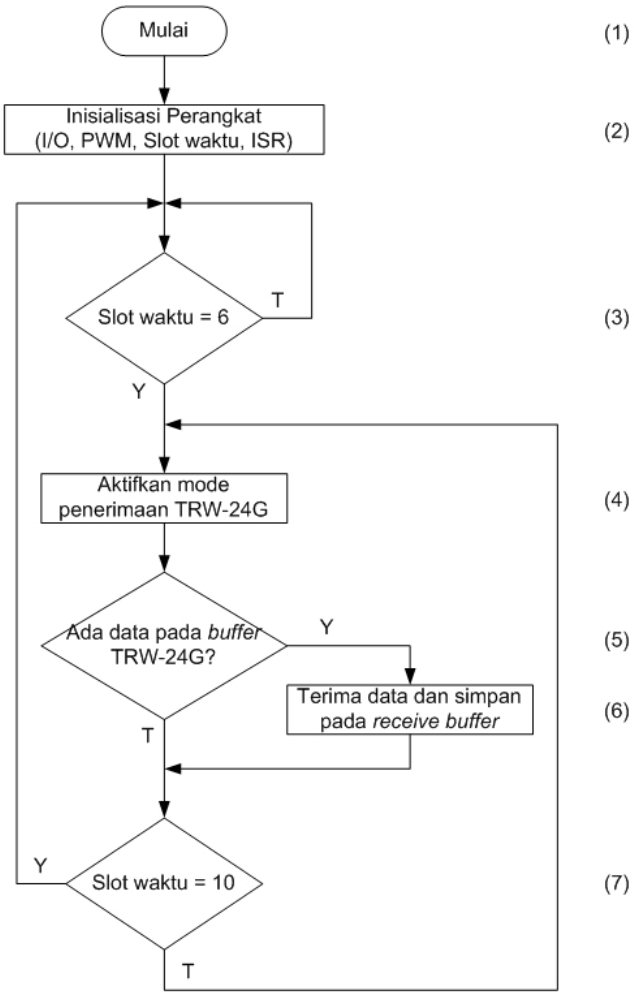
**Gambar 3.7** Ilustrasi Pengiriman Paket Data

### 3.2.6. Perancangan Alur Penerimaan Paket Data

Perancangan alur penerimaan paket data ditujukan untuk membangun perangkat lunak pada sisi penerima. Ketika penerima dinyalakan, perangkat lunak akan langsung mengeksekusi perintah yang terdapat pada *loop* program utama dan perintah yang terdapat di dalam ISR. Penerima akan menggunakan penerimaan sebuah paket untuk melakukan sinkronisasi dengan men-set counter slot waktu pada slot waktu ke 9. Diagram alir loop program utama diilustrasikan pada Gambar 3.8.

Secara umum, proses yang akan dijalankan di dalam *loop* pada program utama pada penerima antara lain:

1. Menunggu hingga slot waktu ke 6.
2. Mengaktifkan mode penerima dan melakukan penerimaan sebuah paket data. Time out pada slot waktu ke 10.




**Gambar 3.8** Diagram Alir *Loop* Program Utama Penerima

Keterangan:

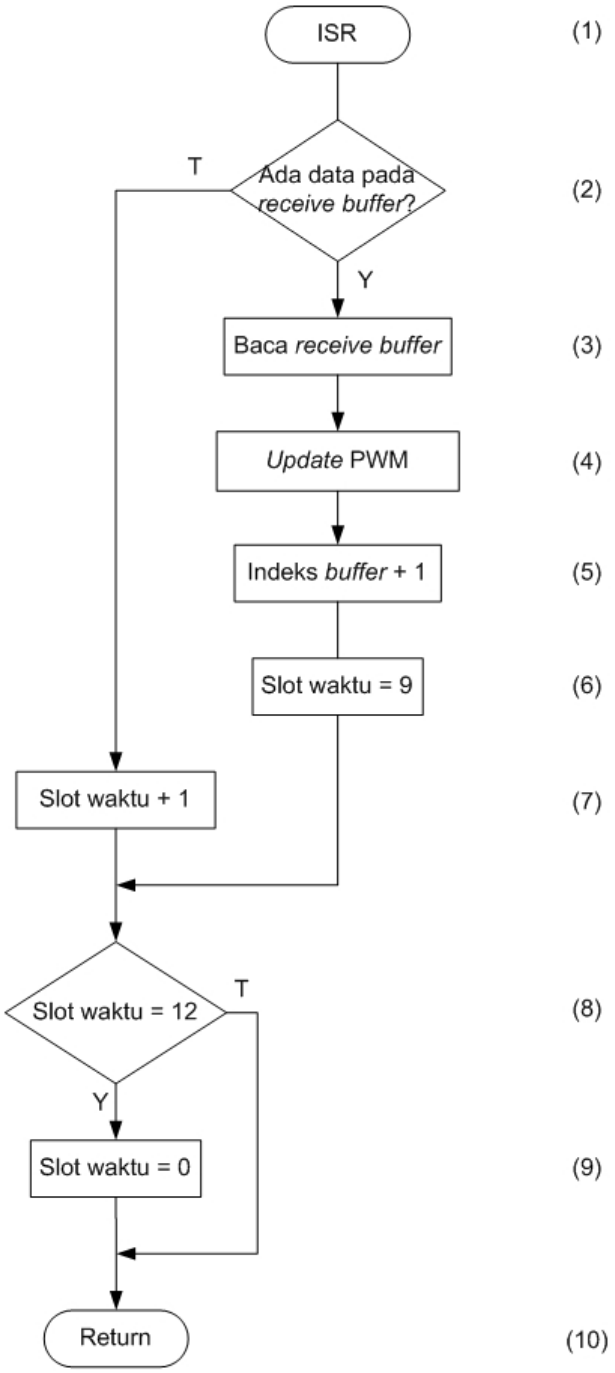
1. Awal program.
2. Inisialisasi atau konfigurasi untuk perangkat-perangkat internal mikrokontroler ATmega8535L dan TRW-2.4G. Pada bagian ini, program akan melakukan konfigurasi port I/O, Timer 1 ATmega8535L untuk membangkitkan sinyal PWM, Timer 0 sebagai slot waktu, TRW-2.4G, dan global *interrupt*.
3. Program akan memeriksa apakah *counter* slot waktu bernilai 6. Jika *counter* slot waktu bernilai 6, maka program akan melakukan eksekusi terhadap proses selanjutnya. Jika *counter* slot waktu tidak bernilai 6, maka program akan melakukan proses perulangan hingga *counter* slot waktu bernilai 6.

4. Jika *counter* slot waktu bernilai 6, maka program akan melakukan konfigurasi terhadap TRW-2.4G untuk segera mengaktifkan mode penerimaan data.
5. Program akan melakukan pemeriksaan terhadap keberadaan data pada *buffer* TRW-2.4G. TRW-2.4G akan mengaktifkan pin DR1 (PINB5 pada mikrokontroler ATmega8535L) untuk menandakan bahwa data pada *buffer* TRW-2.4G telah siap untuk dikeluarkan untuk selanjutnya akan disimpan di dalam *receive buffer* pada mikrokontroler ATmega8535L. Jika tidak, maka program tidak akan melakukan pengambilan data dari TRW-2.4G.
6. Pada langkah ini mikrokontroler ATmega8535L akan melakukan pengambilan data dari *buffer* TRW-2.4G untuk disimpan ke dalam *receive buffer* pada mikrokontroler ATmega8535L.
7. Program akan melakukan pemeriksaan kembali terhadap nilai *counter* slot waktu. Jika *counter* slot waktu bernilai 10, maka program akan menganggap bahwa proses penerimaan data telah mengalami *time-out* atau sama sekali tidak ada data yang diterima. Pada kondisi ini, program akan kembali mengeksekusi perintah pada langkah ke-3. Sedangkan jika counter slot waktu belum bernilai 10, program akan mengulang eksekusi perintah pada langkah ke-4.

Proses yang akan dilakukan oleh ISR pada penerima antara lain:

1. Membaca satu sampel dari *receive buffer* dan melakukan *update* PWM.
2. Melakukan *update counter* slot waktu.

Diagram alir untuk proses yang dilakukan oleh ISR pada penerima diilustrasikan pada Gambar 3.9.




**Gambar 3.9** Diagram Alir ISR Penerima

Keterangan:

1. Awal rutin ISR. ISR akan dieksekusi tiap 125µs, yaitu ketika terjadi *overflow interrupt* pada Timer 0.
2. ISR melakukan pemeriksaan terhadap ada atau tidaknya data pada *receive buffer*. Jika *receive buffer* telah berisi data, maka ISR akan segera

melanjutkan pembacaan data pada langkah ke-3. Jika *receive buffer* belum terisi oleh data baru, maka ISR akan melompat ke proses *increment* slot waktu pada langkah ke-7.

3. ISR melakukan pembacaan data yang terdapat di *receive buffer*.
4. ISR melakukan update terhadap *register* data PWM dengan nilai yang sesuai dengan nilai yang dibaca di *receive buffer*.
5. *Increment* indeks *transmit buffer*.
6. Nilai *counter* slot waktu akan langsung diupdate dengan nilai 9 untuk menandakan bahwa data yang diterima oleh penerima telah sinkron dengan data yang dikirim oleh pengirim.
7. *Increment counter* slot waktu oleh ISR yang terjadi jika tidak ada update data pada *receive buffer*.
8. Melakukan pemeriksaan terhadap nilai counter slot waktu. Jika *counter* slot waktu bernilai 12, maka ISR akan mengeksekusi perintah selanjutnya. Jika tidak, maka ISR akan melakukan *increment* terhadap *counter* slot waktu pada *interrupt* selanjutnya.
9. ISR memberikan *counter* slot waktu dengan nilai 0.
10. Proses dalam ISR telah selesai dieksekusi, dan kontrol akan dikembalikan pada *loop* program utama hingga *interrupt* selanjutnya.

Berdasarkan dua fungsi utama pada perancangan alur penerimaan paket data diatas maka alur penerimaan paket data dapat diilustrasikan pada Gambar 3.10.

| Penerima | | RX | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Slot waktu (125µs) | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 0 | 1 | 2 | 3 | 4 |

**Gambar 3.10** Ilustrasi Penerimaan Paket Data