BAB III
PERANCANGAN SYSTEM

Perancangan sistem yang dilakukan merupakan perancangan yang dilakukan secara bertahap, dimana tahap tersebut adalah melakukan percobaan dan melakukan pengujian yang menghasilkan sebuah pengukuran dengan pertimbangan hasil pengukuran adalah sesuai yang diharapkan atau belum sesuai dengan yang diharapkan.

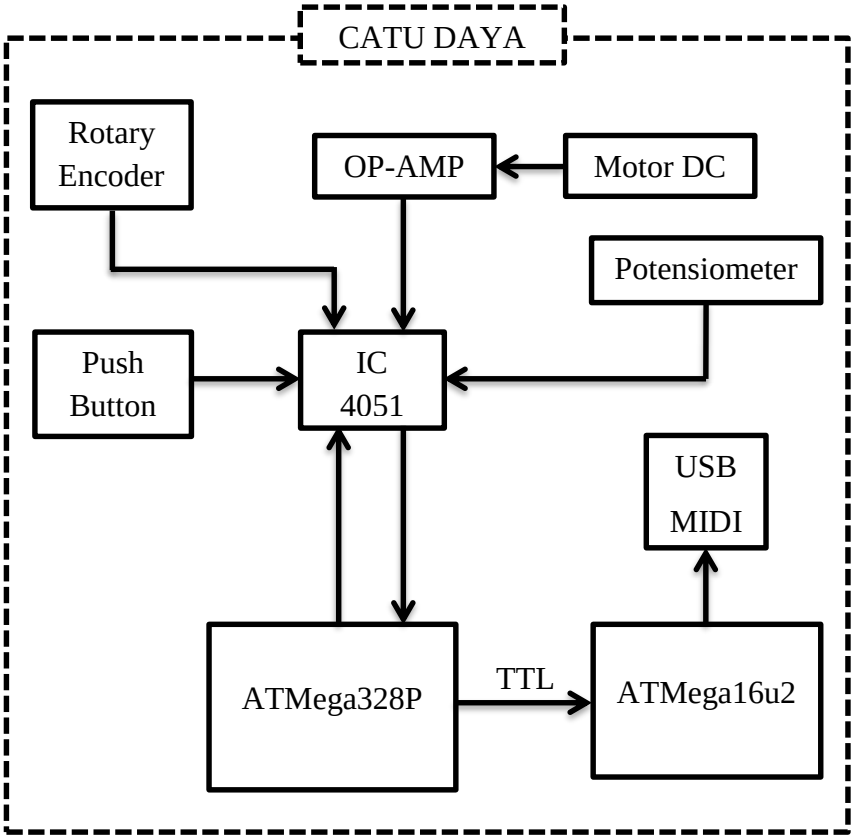
Secara fisik terdapat beberapa penyusun komponen inti yang digunakan untuk dijadikan sensor pada blok *input* yakni potensiometer, *Push Button*, *Rotary Encoder*, motor dc, dan IC *multiplexer*. Potensiometer digunakan untuk melakukan pengendalian *knob* (*limited knob*) dan *slider* (*fader*) pada bagian *mixer* dan deck didalam aplikasi penyaji seperti pengendalian *volume*, *equalizer*, dan sebagainya. *Push Button* digunakan untuk melakukan pengendalian pada panel yang bersifat *button* pada aplikasi antarmuka (Virtual DJ), seperti contohnya tombol *play*, *stop*, dan sebagainya. Sedangkan *Rotary Encoder* digunakan untuk melakukan pengendalian pada panel yang bersifat *unlimited knob* pada aplikasi antarmuka, contohnya adalah pada pemilihan *audio effect* dapat dilakukan dengan komponen ini. Sedangkang motor dc pada prinsipnya sama dengan *Rotary Encoder* namun untuk penggunaan motor dc dikhususkan pada pengendalian panel *jog wheel*, dengan bentuk *design* secara fisik adalah 2 deck dan 1 *mixer*.

Beberapa perangkat lunak digunakan sebagai pendukung agar sistem dapat benar – benar berjalan dengan baik dan sesuai dengan yang diharapkan. Beberapa perangkat lunak yang digunakan adalah aplikasi antarmuka untuk pemprograman arduino (Arduino IDE) dan aplikasi antarmuka untuk pemprograman *Device Firmware Upgrade* (FLIP).

3.1. PERANCANGAN PERANGKAT KERAS

3.1.1. Blok Diagram Sistem

Blok diagram secara keseluruhan digambarkan seperti pada gambar 3.1. Sehingga perancangan dilakukan dengan cara menguji setiap komponen pada blok *input* yang dilanjutkan dengan pengujian pengiriman data di komunikasi serial yang digunakan.



Gambar 3.1 Blok Diagram Perangkat

Setelah hasil yang didapatkan sesuai dengan yang diharapkan pada masing – masing blok *input* yang diteruskan ke blok proses, maka dilanjutkan dengan penggabungan rangkaian yang tersusun secara blok diagram seperti pada gambar 3.1 yakni gambar blok diagram perangangkat secara keseluruhan.

Tahap berikutnya merupakan pengujian terakhir yakni melakukan pengujian secara keseluruhan dengan cara mendemonstrasikan perangkat, apabila terdapat kesalahan maka dilakukan analisa permasalahan dengan cara melakukan pengujian ulang pada blok tersebut hingga mendapatkan sebuah kesimpulan permasalahan. Apabila perangkat telah dapat berjalan secara semestinya maka dilanjutkan dengan tahap pengambilan kesimpulan secara keseluruhan.

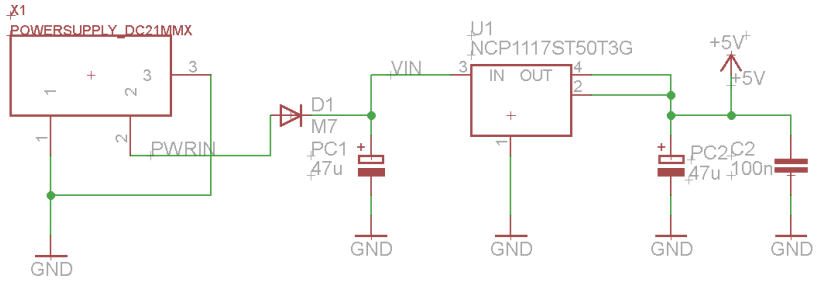
3.1.2. Rangkaian Catu Daya Sistem

Perangkat menggunakan 2 pilihan catu daya yakni; pilihan pertama catu daya menggunakan USB dimana sumber tegangan tersebut didapatkan

dari catu daya pada komputer, dan catu daya alternatif adalah menggunakan catu daya eksternal yang terhubung ke rangkaian regulator pada arduino *board* dengan spesifikasi sebagai berikut :

Arah Arus Masukan	: Arus Searah (DC)
Tegangan Operasi	: 5 Volt
Tegangan Masukan (direkomendasi)	: 7-12 Volt
Batas Tegangan Masukan	: 6-20 Volt

Spesifikasi tersebut didapatkan berdasarkan pada komponen regulator tegangan yang digunakan pada arduino *board* dengan tujuan agar sistem dapat bekerja dengan baik sesuai dengan yang diharapkan. Gambar skematik rangkaian catu daya eksternal yang digunakan pada papan arduino digambarkan pada gambar 3.2.



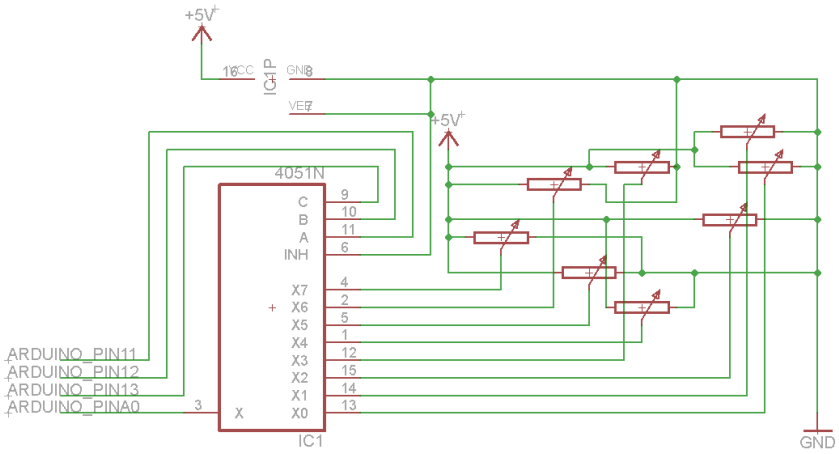
Gambar 3.2 Skematik Rangkaian Catu Daya Eksternal

3.1.3. Perancangan Sensor Potensiometer

Terdapat dua jenis *input* komponen potensiometer yang digunakan yakni potensiometer *rotation* dan potensiometer *slider*. Keduanya digunakan untuk melakukan pengendalian pada panel *fader* dan *knob* yang terdapat pada sisi *mixer* maupun *deck*. Terdapat 8 Komponen *input* yang dapat dimultiplex oleh IC Multiplexer 4051 sehingga 8 masukan tersebut hanya membutuhkan 1 pin yang mendukung *Analog to Digital Converter* (ADC) pada arduino untuk pembacaan nilai. Konfigurasi skematik digambarkan seperti pada gambar 3.3.

Pada gambar 3.3 merupakan susunan rangkaian komponen *input* berupa 1 potensiometer yang keluarannya dihubungkan ke *channel* 0 dan 7 *channel* lainnya terhubung ke masing – masing potensiometer yang

kemudian dimultiplex menggunakan IC 4051 dan menjadi 1 masukan pada arduino di pin A0.



Gambar 3.3 Skematik Rangkaian Sensor Potensiometer

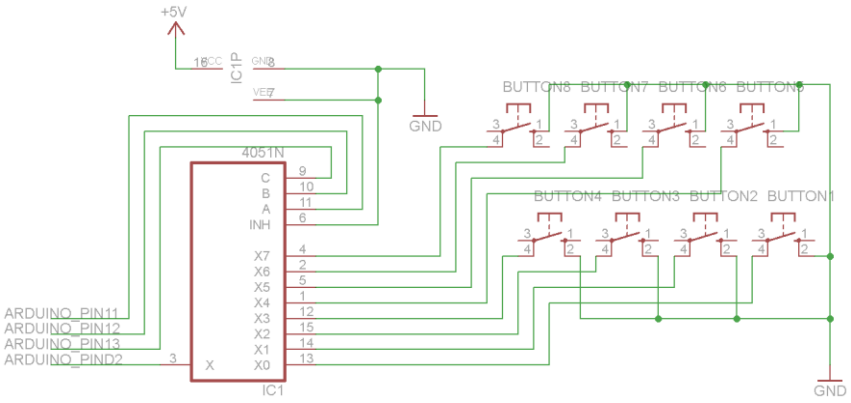
Pada rangkaian utuh atau rangkaian keseluruhan yang telah tergabung dari beberapa komponen akan memaksimalkan pin channel I/O, atau dengan kata lain hampir seluruh channel pada pin I/O yang terdapat pada IC multiplexer akan digunakan, namun apabila terdapat beberapa pin channel pada IC multiplexer tertentu yang tidak digunakan atau berlebih maka akan diabaikan, dimana pengaturan tersebut dilakukan pada perancangan perangkat lunak atau pada saat penulisan instruksi pada *sketch* program perangkat.

3.1.4. Perancangan Sensor *Push Button*

Push Button digunakan sebagai sensor yang bertujuan untuk pengendalian panel pada *software* aplikasi antarmuka yang bersifat button. Dimana skematik rangkaian yang digunakan adalah seperti pada gambar 3.4 yang menggambarkan hubungan sensor terhadap perangkat pemproses dan pengirim data ke komputer.

Berdasarkan skematik rangkaian, untuk penentukan nilai awal digunakan *pull-up internal* dengan memanfaatkan *pull-up* pada MCU (ATmega328P). Dengan kata lain dapat dikatakan bahwa nilai awal saat

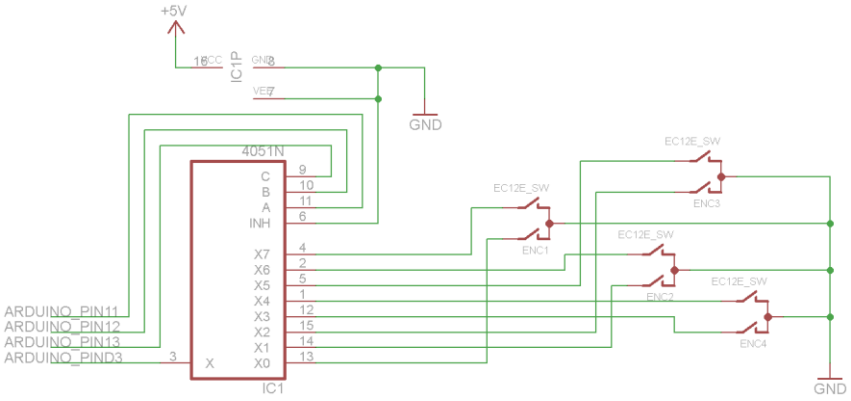
tidak terjadi perubahan pada posisi *Push Button* (NC) dilogikakan 1 dan saat posisi berubah (NO) maka logika akan berganti menjadi logika 0.



Gambar 3.4 Skematik Rangkaian Sensor *Push Button*

3.1.5. Perancangan Sensor *Rotary Encoder*

Dalam perancangan perangkat, sensor *rotary encoder* digunakan untuk mendeteksi arah putaran dimana arah putaran tersebut adalah searah jarum jam atau berlawanan arah jarum jam. Skematik rangkaian yang dirancang untuk sensor *rotary encoder* adalah seperti pada gambar 3.5.



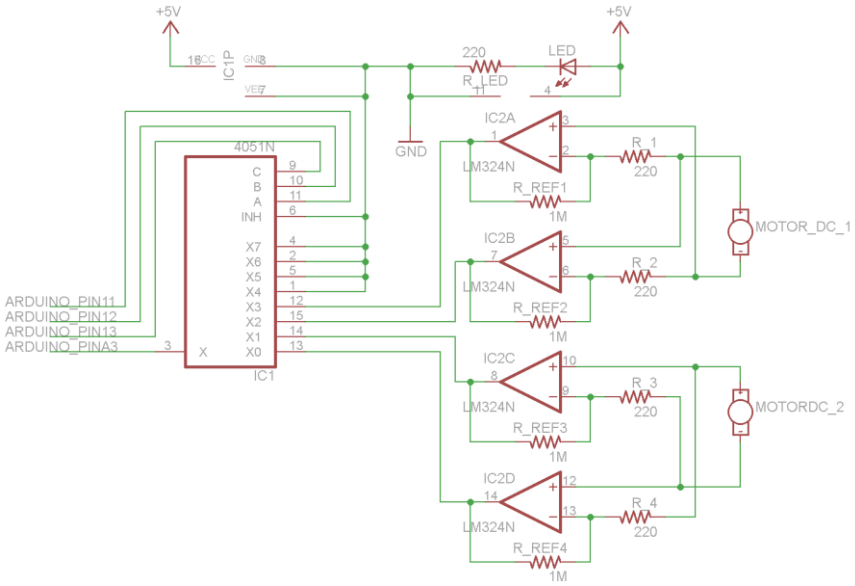
Gambar 3.5 Skematik Rangkaian Sensor *Rotary Encoder*

Dengan memanfaatkan pendeteksian arah putaran tersebut, salah satunya dapat dimanfaatkan untuk mengendalikan fungsi *browser* atau pemilihan musik pada panel *browser* dimana song yang terpilih akan di *load* ke *deck*. Berdasarkan pada gambar 3.5, rangkaian sensor *Rotary Encoder* menggunakan *pull-up internal* dimana hal tersebut guna menentukan posisi awal saat tidak terjadi perubahan pada pin arduino yang diteruskan ke IC

multiplexer untuk pembacaan nilai tegangan dengan didefinisikan sebagai logika 0 atau 1 (*HIGH* atau *LOW*).

3.1.6. Perancangan Motor DC Sebagai Sensor

Motor DC dirancang sebagai sensor yang bertujuan sama dengan *Rotary Encoder* dimana secara fungsi yakni untuk mendapatkan arah putaran. Namun pada perancangannya, motor dc digunakan untuk pengendalian panel *jogwheel* pada *deck*. Berdasarkan perancangan secara fisik dimana *deck* yang dirancang adalah berjumlah 2 *deck*, sehingga jumlah motor dc yang digunakan dalam penelitian ini adalah sebanyak 2 motor dc. Pada gambar 3.6 merupakan skematik rangkaian motor dc yang digunakan sebagai sensor.

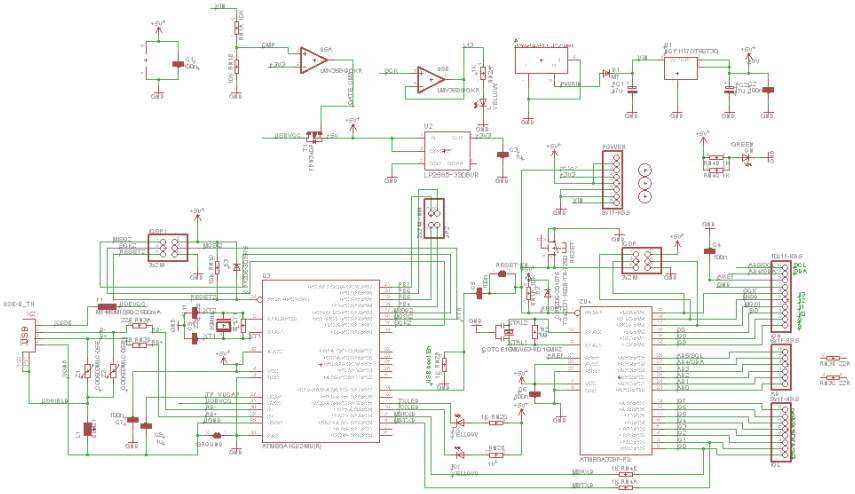


Gambar 3.6 Skematik Sensor Motor DC

Dari skematik pada gambar 3.6, untuk dapat mendapatkan nilai berdasarkan arah putaran digunakan rangkaian penguat dan pembanding terbalik agar pembacaan nilai menggunakan ADC dapat lebih akurat, sebab tegangan yang dihasilkan oleh perputaran dari motor dc sebagai generator tegangan sangat kecil. Dengan demikian perubahan yang mungkin terjadi adalah apabila perputaran terjadi ke arah kiri atau kanan, maka salah satu keluaran akan bernilai 0 unit ADC.

3.1.7. Perancangan Perangkat Pemproses dan Pengirim Data

Perangkat pemproses dan pengirim data menggunakan rangkaian perangkat yang berbasis pada Arduino *Board*, secara utuh pada penelitian ini menggunakan papan Arduino UNO R3 sebagai pemproses data masukkan yang didapat dari sensor dan sebagai pengirim data berupa pesan MIDI ke komputer. Semua hasil pendeteksian yang telah diolah diteruskan ke komputer berupa pesan MIDI. Skematik rangkaian pada Arduino yang digunakan adalah seperti pada gambar 3.7.



Gambar 3.7 Skematik Rangkaian Arduino UNO R3

Pada papan Arduino UNO R3 terdapat dua IC pemproses yaitu Atmega328P dan Atmega16u2. Dimana Atmega328P bekerja sebagai pemproses atau mengolah data masukkan dan kemudian mengirimkan sebagai pesan dalam format MIDI. Sedangkan Atmega16u2 bekerja sebagai pengubah level tegangan antara level tegangan *Transistor-Transistor-Logic* (TTL) dan *Universal Serial Bus* (USB) serta pada Atmega16u2 diterapkan *Firmware* MIDI dimana dengan kata lain Atmega16u2 dapat dikatakan sebagai antarmuka perangkat MIDI melalui USB.

3.2. PERANCANGAN PERANGKAT LUNAK

3.2.1. *Flowchart* Program

Secara umum urutan pekerjaan yang dilakukan perangkat adalah mendeteksi secara terus – menerus setiap perubahan yang terjadi pada

sensor dan kemudian dari informasi tersebut diolah menjadi sebuah data yang berupa pesan MIDI dan kemudian dikirimkan melalui jalur komunikasi serial.

Algoritma pekerjaan yang terjadi pada perangkat adalah dimulai dari perangkat diaktifkan. Pada tahap awal yakni melakukan inisialisasi, dimana inisialisasi mencakup pekerjaan yang dilakukan oleh *bootloader*, penentuan pin I/O, deklarasi variabel, dan konfigurasi protokol komunikasi serial.

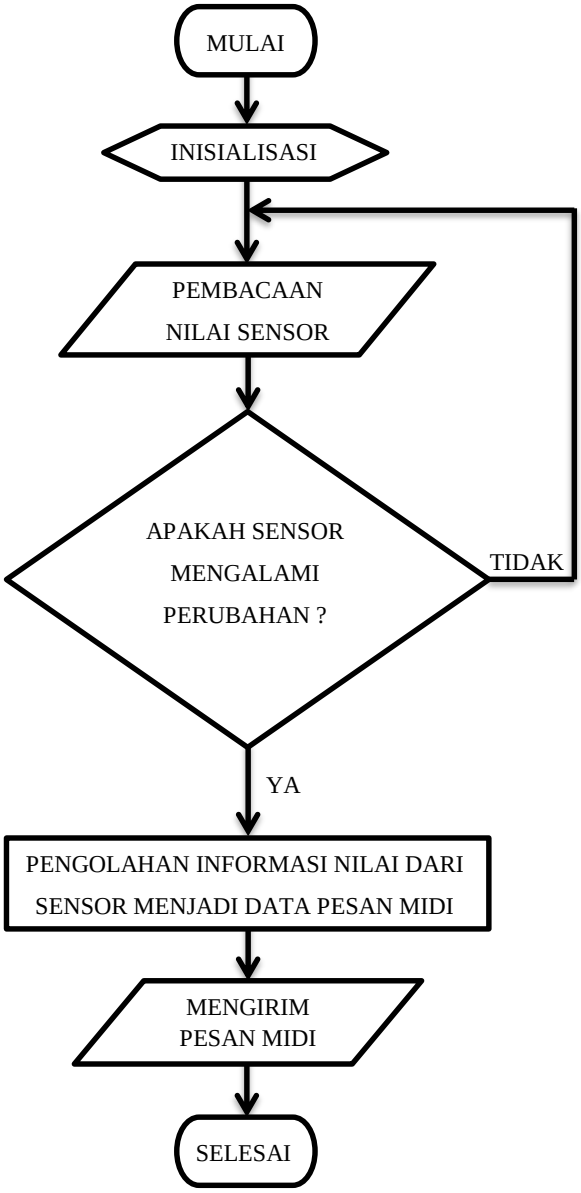
Urutan berikutnya adalah menjalankan program utama, dimana urutan yang pertama adalah melakukan pembacaan nilai pada seluruh sensor dengan berurutan atau secara bergantian. Dimana nilai yang didapatkan dari masing – masing sensor disimpan kedalam variabel sebagai nilai hasil pembacaan yang terbaru.

Berikutnya dilakukan pengambilan keputusan dengan logika jika nilai hasil pembacaan yang terbaru berbeda dengan nilai hasil pembacaan sebelumnya maka akan melakukan pengiriman data yang berisi nilai terbaru dari sensor, namun jika kondisi tersebut tidak terpenuhi maka pembacaan nilai terus diulangi. Metode pengambilan keputusan tersebut digunakan untuk mencegah pengiriman data agar tidak dilakukan secara terus menerus namun tidak mempengaruhi waktu pembacaan pada sensor. Sehingga agar komunikasi serial tidak dibanjiri oleh data yang sama seperti dengan data yang telah dikirim sebelumnya maka diterapkanlah metode pengiriman data dengan cara tersebut.

Data yang dikirimkan melalui komunikasi serial berisi nilai yang didapatkan dari hasil pembacaan tegangan pada pin yang terhubung dengan sisi *input*. Kemudian data tersebut diolah dan diatur dengan format berdasarkan pada protokol MIDI, sehingga data yang dikirim tersebut merupakan data yang berupa pesan MIDI.

Secara umum *flowchart* program pada perangkat digambarkan seperti pada gambar 3.8, dimana pada *Flowchart* di gambar 3.8 menggambarkan menggambarkan urutan – urutan pekerjaan berdasarkan instruksi yang telah dirancang pada *sketch* program dan dijalankan oleh perangkat secara sistematis. Serta terdapat instruksi yang dilakukan berdasarkan beberapa

kemungkinan – kemungkinan yang terjadi pada masing – masing sensor dalam satu kali kerja



Gambar 3.8 *Flowchart* Program

.

Masing – masing blok pada *flowchart* di gambar 3.8 adalah hasil penyederhanaan proses dari beberapa urutan proses dengan algoritma yang kompleks. Dengan kata lain, terdapat urutan – urutan pekerjaan didalam masing – masing blok pada *Flowchart* program di gambar 3.8. Urutan proses atau instruksi tersebut dituliskan secara lengkap pada *sketch* program. Penulisan susunan instruksi ditulis dalam bahasa pemprograman

Arduino, dimana bahasa tersebut merupakan bahasa turunan dari bahasa C/C++ atau dengan kata lain bahasa tersebut merupakan bahasa yang dikembangkan dari bahasa C/C++. Antarmuka yang digunakan untuk melakukan penulisan instruksi – instruksi tersebut adalah menggunakan aplikasi Arduino IDE.

3.2.1.1. Blok Persiapan

Pada bagian persiapan merupakan bagian yang hanya dilaksanakan atau dikerjakan oleh perangkat sebanyak satu kali pada saat perangkat diaktifkan. Dimana pada bagian ini, perangkat melakukan inisialisasi, dengan urutan pekerjaan yakni melakukan deklarasi variabel dan dilanjutkan dengan melakukan konfigurasi pada komunikasi serial, serta menetapkan fungsi I/O pada pin – pin tertentu yang digunakan. Penentuan penggunakan resistor *pull-up* atau *pull-down* secara internal atau eksternal juga dilakukan pada bagian ini.

Pada tahap awal secara urutan di *sketch* dilakukan deklarasi variabel, pada *prototype* perangkat ditulisankan menurut bahasa arduino adalah seperti :

```
unsigned long previousMillis = 0;
const long interval = 13;
int i,e = 0;
int b[] = {0,0,0,0,0,0,0,0};
int balt[] = {0,0,0,0,0,0,0,0};
int p[] = {0,0,0,0,0,0,0,0};
int palt[] = {0,0,0,0,0,0,0,0};
int val;
int encoder0PinALast[] = {0,0,0,0,0,0,0,0};
int n[] = {0,0,0,0,0,0,0,0};
int bit1 = 0;
int bit2 = 0;
int bit3 = 0;
```

Dari potongan code tersebut bertujuan untuk melakukan deklarasi variabel, dimana struktur penulisan adalah dengan menentukan tipe data terlebih dahulu dan kemudian menuliskan nama variabel yang dideklarasikan dilanjutkan dengan nilai awalnya (jika diperlukan).

Sedangkan potongan *sketch* pada program untuk melakukan konfigurasi pin dan pengaturan baudrate adalah seperti :

```
void setup() {
  Serial.begin(31250);
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, INPUT_PULLUP);
  pinMode(A1, INPUT_PULLUP);
  pinMode(A0, INPUT);
}
```

Serial.begin(baudrate,config) adalah perintah untuk melakukan pengaturan komunikasi serial. Dimana nilai config akan diatur ke posisi default apabila nilai tidak ditetapkan, sedangkan nilai baudrate wajib diisi.

Perintah *pinMode(pin,mode)* adalah perintah untuk menentukan fungsi I/O pada pin tersebut, dan dapat juga menentukan penggunaan resistor *pull-up* atau resistor *pull-down* apabila pin digunakan sebagai *input*.

3.2.1.2. Blok *Input*

Pada bagian *input* terdapat beberapa komponen yang berfungsi sebagai pembaca konsisi dengan representasi nilai tertentu, pembacaan dapat dilakukan dengan memerintahkan pemproses melalui instruksi *digitalRead(pin)* untuk pembacaan data secara digital, dimana dapat diketahui bahwa dengan metode tersebut hanya akan terdapat dua kemungkinan yakni logika 1 (*HIGH*) dan logika 0 (*LOW*). Serta terdapat penggunaan perintah *analogRead(pin)* adalah untuk pembacaan nilai analog, dimana secara teknis adalah dengan memanfaatkan fasilitas *Analog to Digital Converter* (ADC) yang terdapat pada ATMega328P dengan resolusi ADC sebesar 10 bit.

3.2.1.3. Blok Pengambilan Keputusan

Pengambilan keputusan merupakan hal penting dalam sebuah program. Untuk melakukan pengambilan keputusan berdasarkan kondisi yang terdaftar adalah dengan menggunakan perintah *if (kondisi) {instruksi}* dimana kondisi merupakan tolak ukur yang harus terpenuhi untuk menjalankan instruksi. Sedangkan perintah *else {instruksi}* digunakan untuk menjalankan instruksi apabila pada kenyataannya kondisi belum terpenuhi, dengan kata lain kondisi yang didapatkan belum memenuhi persyaratan atau belum sesuai dengan kondisi yang telah didaftarkan.

3.2.1.4. Blok Pengolahan

Pengiriman nilai pada pesan MIDI memiliki format yang berbeda dengan representasi nilai yang diperoleh dari ADC, dimana nilai hasil representasi pada ADC yang berupa analog memiliki *range* 0-1023. Hasil representasi tersebut belum sesuai dengan format MIDI sehingga pemetaan harus dilakukan guna menyesuaikan informasi yang didapatkan dari komponen *input* terhadap format data MIDI. Pemetaan nilai merupakan metode untuk merubah nilai berdasarkan resolusi, dimana hasil resolusi didapatkan dari perbandingan jumlah dari nilai asli berbanding nilai hasil. Untuk melakukan pemetaan nilai dapat dilakukan dengan perintah *map(nilai_referensi, nilai_minimum_referensi, nilai_maksimum_ referensi nilai_minimum_hasil, nilai_maksimum_hasil)* dimana contoh penggunaan pada program seperti *p[i] = map (analogRead(A0),0,1023,0,127)* yang berarti bahwa melakukan perubahan nilai dari *range* 0-1023 menjadi 0-127 maka resolusi yang didapatkan adalah 1024 : 128 atau 8:1. Sehingga nilai yang akan tersimpan pada variabel p adalah nilai yang didapatkan dari nilai referensi yang telah diresolusikan.

Terdapat pemetaan lain yaitu pemetaan pin yang bertujuan untuk menentukan arah *switching* pin *common* yang terhubung pada arduino dengan pin *channel* yang terhubung pada komponen *input* di IC Multiplexer.

Penentuan alamat *channel* tujuan adalah dengan cara mengkombinasikan 3 nilai logika *selection* pin yang terdapat di IC Multiplexer (pin 11,10,9). Dimana tabel pengalamatan pada tabel 3.1 menunjukkan pengalamatan pin *channel* yang terhubung dengan pin *common*.

Tabel 3.1 Pengalamatan Jalur IC Multiplexer

Selection A (Pin 11)	Selection B (Pin 10)	Selection C (Pin 9)	Channel (Pin)
0	0	0	0 (Pin 13)
0	0	1	1 (Pin 14
0	1	0	2 (Pin 15)
0	1	1	3 (Pin 12)
1	0	0	4 (Pin 1)
1	0	1	5 (Pin 5)
1	1	0	6 (Pin 2)
1	1	1	7 (Pin 4)

Sehingga berdasarkan tabel 3.1 disusun *sketch* program untuk pengalamatkan seperti :

```
for (i = 0 ; i <= 7; i++ ) {
  bit1 = bitRead(i, 0);
  bit2 = bitRead(i, 1);
  bit3 = bitRead(i, 2);
  digitalWrite(2, bit1);
  digitalWrite(3, bit2);
  digitalWrite(4, bit3);
}
```

Sketch program yang berfungsi sebagai perulangan tersebut bertujuan untuk melakukan *switching* ke seluruh *channel* secara bergantian pada IC Multiplexer 4051.

3.2.1.5. Blok *Output*

Pengiriman pesan MIDI melalui komunikasi serial adalah dengan memanfaatkan perintah *Serial.write(nilai_desimal)* dimana nilai

tersebut didapatkan dari hasil penyesuaian terhadap format MIDI (penyesuaian format dan hasil resolusi). Pemanfaatan fungsi perintah tersebut adalah metode yang tepat agar data yang dikirim sesuai dengan format pesan MIDI. Dimana contoh potongan code untuk mengirim data berdasarkan format pesan MIDI yang adalah seperti :

```
Serial.write(176);
Serial.write(i);
Serial.write(p[i]);
```

Dapat diketahui bahwa data yang terkirim berukuran 3 byte dimana nilai pada byte pertama adalah 176 (dalam desimal), byte kedua adalah nilai berdasarkan variabel i, sedangkan nilai byte ketiga diambil dari variabel p dengan index dari nilai i. Maka dengan susunan pengiriman data tersebut telah sesuai dengan format pesan MIDI untuk melakukan perubahan nilai pada suatu panel yang bersifat analog atau dengan kata lain format pengiriman data tersebut mewakili format untuk melakukan fungsi Control Change yang berdasarkan pada protokol MIDI.

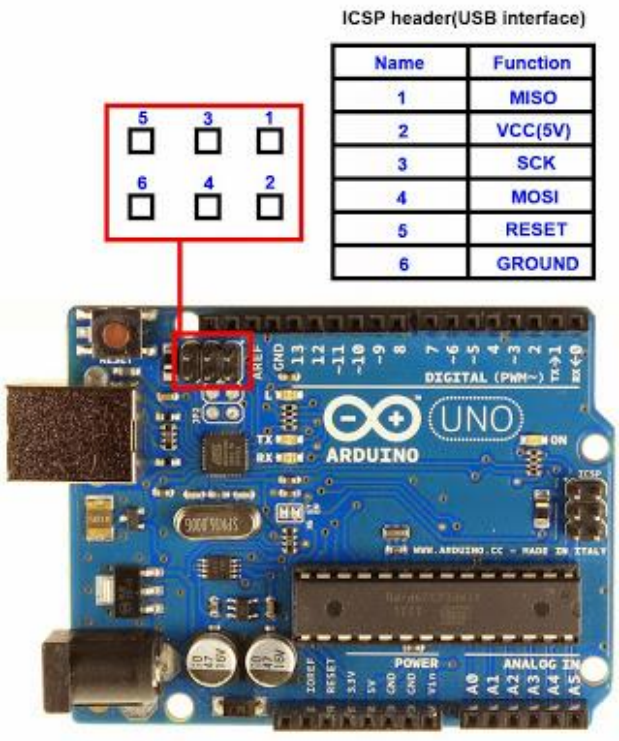
3.2.2. Perancangan *Firmware*

Perancangan *firmware* MIDI merupakan penggantian *firmware* yang digunakan oleh platform ardunio menjadi *firmware* MIDI yaitu MocoLUFA. Penggantian *firmware* adalah dengan cara pemprograman perangkat untuk mengganti *firmware* (*Device Firmware Upgrade Programming*) menjadi *firmware* MIDI yang bertujuan untuk menetapkan perangkat yang berbasis mikrokontroler tersebut menjadi sebuah perangkat MIDI yang dapat terintegrasi dengan perangkat komputer. *Firmware* merupakan sebuah program yang dapat mengintegrasi suatu perangkat dengan perangkat lainnya untuk melakukan komunikasi berdasarkan pada suatu protokol dengan antarmuka tertentu. Pemprograman atau penggantian *firmware* tersebut dilakukan melalui antarmuka perangkat lunak Flexible in-system Programmer (FLIP) dengan versi 3.4.7. Dimana pemprograman

tersebut bertujuan agar pada saat perangkat terhubung dengan komputer, komputer akan mengenali perangkat tersebut sebagai perangkat MIDI.

Prosedur untuk tahap – tahap dalam melakukan upgrade *firmware* pada arduino *board* adalah sebagai berikut :

1. Melakukan reset *firmware* yang dilakukan dengan cara menghubungkan pin *reset* dengan pin *ground* pada icsp *header* yang terhubung dengan chip atmega16u2 pada arduino. Posisi icsp *header* chip atmega16u2 adalah seperti yang digambarkan pada gambar 3.9.



Gambar 3.9 Letak Posisi ICSP *Header* pada Arduino *Board*[6]

Penghubungan pin *reset* dengan pin *ground* hanya dilakukan satu kali dan dalam waktu ± 1 detik, dan dilanjutkan dengan instalasi *usb driver* pada komputer agar perangkat dapat dikenali dan *upload* dapat dilakukan ke perangkat.

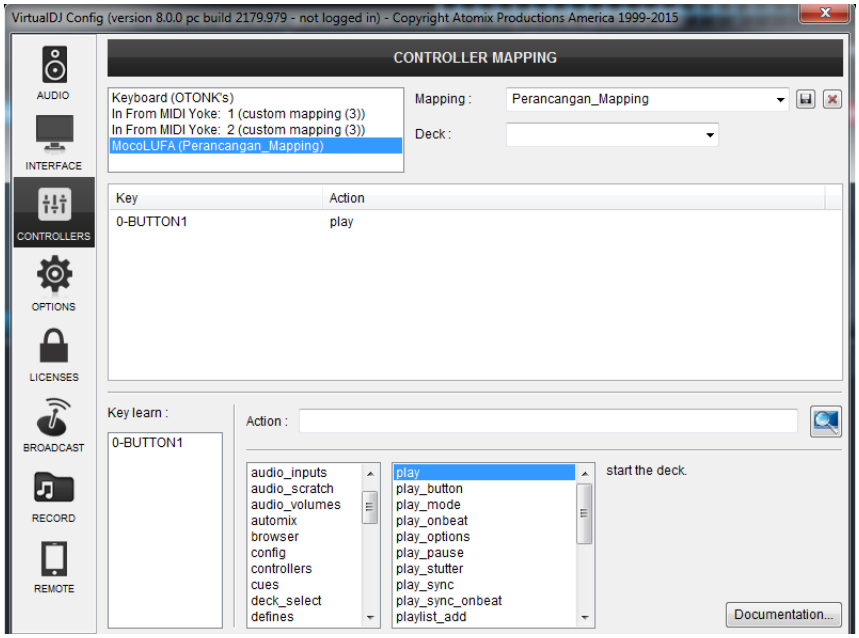
2. Tahap berikutnya yaitu melakukan konfigurasi pada aplikasi antarmuka (FLIP 3.4.7), yang meliputi persiapan untuk melakukan *upgrade firmware*, penghapusan *firmware* terdahulu dan menggantikan dengan *firmware* yang baru (*upload*).

3. Tahap terakhir adalah mereset papan arduino. Dimana pada tahap ini, perangkat telah siap digunakan dengan *firmware* baru.

Adapun hal lain yang dilakukan dalam perancangan *firmware* MIDI dimana pada tahap eksperimen menggunakan *software* aplikasi yang berfungsi sebagai virtual untuk pengganti *firmware*. Mengingat fleksibilitas yang diperlukan dalam perangcangan sehingga hal tersebut harus dilakukan. Dimana *software* aplikasi tersebut adalah Hairless – MIDI Serial dan MIDI yoke yang dimana Hairless – MIDI Serial bertujuan untuk dapat melakukan routing pengiriman data ke perangkat MIDI dan MIDI yoke bertujuan untuk dapat menjadi virtual driver untuk perangkat MIDI.

3.2.3. MIDI *Mapping*

Pemetaan pada sisi *software* aplikasi antarmuka merupakan hal wajib dilakukan agar fungsi sensor dapat merujuk ke aplikasi antarmuka dengan benar dan tepat secara fungsi yang sesuai dengan yang diharapkan.



Gambar 3.10 Tampilan Halaman Pengaturan *Controllers*.

Pemetaan yang dilakukan pada *software* aplikasi antarmuka yakni memetakan setiap *input* terhadap panel – panel yang terdapat pada *software* aplikasi antarmuka (Virtual DJ). Sehingga masing – masing sensor dapat mengendalikan satu panel pada aplikasi antarmuka. Pada perancangannya, system dirancang untuk dapat terintegrasi dengan perangkat Virtual DJ yang

dimana pemetaan dapat dilakukan pada halaman pengaturan *Controllers*. Pada gambar 3.10 menggambarkan tampilan halaman untuk melakukan pemetaan komponen *input*. Setiap komponen *input* (sensor) harus didaftarkan terlebih dahulu kedalam list *key* agar kemudian dapat difungsikan sebagai pengendali terhadap panel – panel yang terdapat pada aplikasi Vitrual DJ sesuai dengan fungsi komponen *input*. Pada gambar 3.10 juga menggambarkan pemetaan 1 buah sensor *Push Button* yang difungsikan sebagai panel *play* pada *deck*.