BAB III

METODOLOGI PENELITIAN

3.1 Waktu dan Tempat Penelitian

Kegiatan utama yang dilakukan pada penelitian terbagi dalam beberapa bagian yaitu penelitian sensor CO_2 MH-Z19, Pembuatan antarmuka I^2C untuk MH-Z19 dan uji coba pada 3 jenis perangkat. Penelitian ini dilakukan mulai dari bulan januari 2018 dan selesai pada bulan mei 2018. Lokasi penelitian akan dilakukan pada Universitas Muhammadiyah Yogyakarta terutama pada laboratorium MRC teknik elektro dan pada ruang kamar pribadi.

3.2 Alat dan Bahan

Terdapat beberapa alat dan bahan yang dibutuhkan dalam tugas akhir mengenai pembuatan antarmuka I^2C atau *converter UART to* I^2C pada sensor CO_2 MH-Z19. Berikut adalah peralatan dan bahan yang diperlukan.

3.2.1 Peralatan Penunjang

Tabel 3.1 Peralatan Penunjang

Perangkat Keras	Perangkat Lunak
Komputer	Arduino IDE
Solder	Proteus 8.5
Atraktor	Fritzing
Bor	Processing
Multimeter Digital	Extreme Burner - AVR

3.2.2 Bahan Penelitian

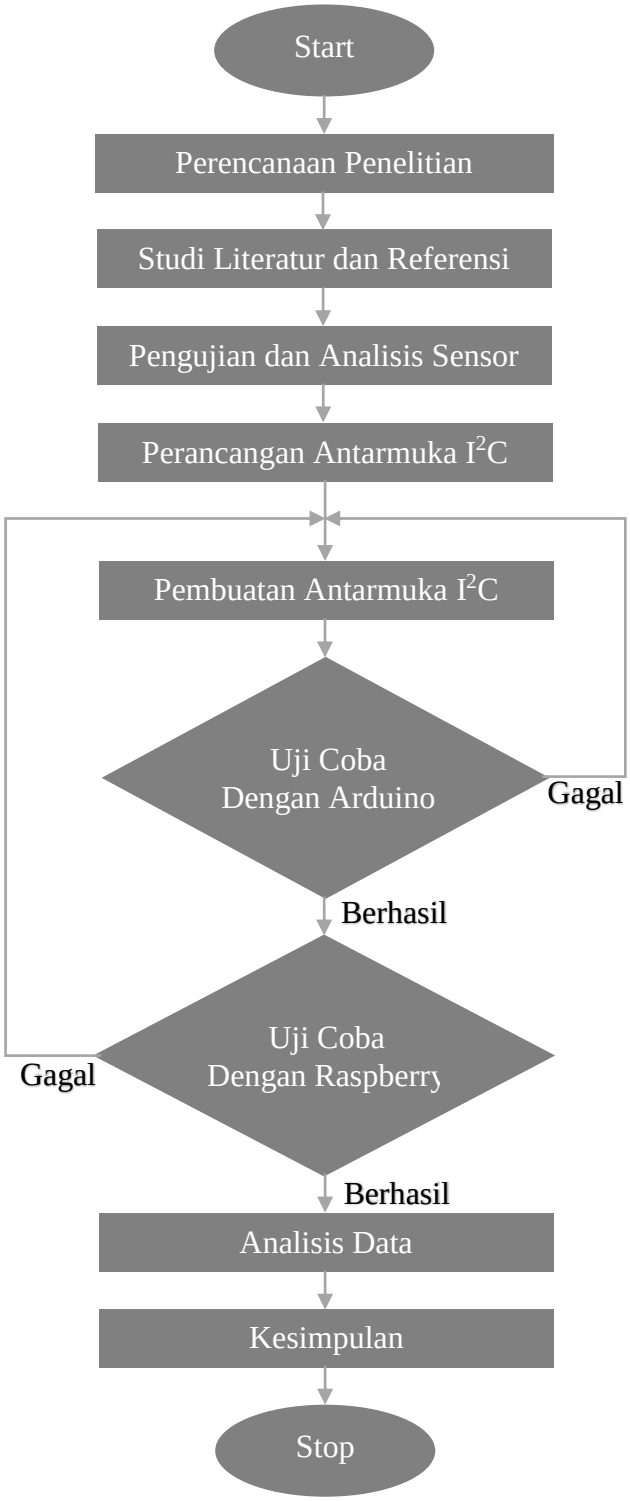
Tabel 3.2 Bahan Penelitian

No	Bahan	Jumlah (buah)
1	Arduino UNO	2
2	Kabel USB Arduino	2
3	Sensor CO_2 MH-Z19	1
4	Atmega 8 SMD	2
5	Oled 128 x 32	2
6	Raspberry pi 3	1
7	Breadboard	1
8	Kabel Jumper	Secukupnya
9	Crystal SMD CSTCE16MOV53-R0	2
10	Resistor 0805 1 MΩ	2
11	Resistor 0805 1 KΩ	4
12	Resistor 0805 10 KΩ	2
13	Resistor 0805 4,7 KΩ	4
14	Kapasitor 100 nF	1
15	Led 1206	4
16	Pin Header Male	2 strip
17	Pin Header Female	2 strip
18	PCB	1
19	FeCl	1
20	Acrylic	Secukupnya
21	Lem Acrylic	1
22	Tenol Cair	1

3.3 Prosedur Penelitian

Dalam penelitian tugas akhir ini terdapat kegiatan perencanaan, perancangan alat dan beberapa uji coba yang harus dilakukan. Setiap kegiatan

penelitian memiliki urutan kerja dan prosedur. Berikut ini adalah diagram alir prosedur penelitian yang dilaksanakan.



Gambar 3.1 Diagram Alir Penelitian

3.3.1 Penjelasan Diagram Alir Penelitian

1. Start

   Tahap ini adalah tahap paling awal dalam penelitian tugas akhir mengenai pembuatan antarmuka I^2C atau *converter UART to* I^2C pada sensor CO_2 MH-Z19.

2. Perencanaan Penelitian

   Pada tahap ini akan ditentukan tujuan dan rencana kegiatan yang akan dikerjakan pada penelitian tugas akhir ini. Sesuai dengan tujuan dan rencana penelitian yang telah disampaikan pada bab 1 pendahuluan.

3. Studi Literatur dan Referensi

   Penelitian akan kurang maksimal apabila tidak mengetahui masalah dan kekurangan yang ada pada beberapa penelitian yang terkait sebelumnya. Oleh sebab itu, dibutuhkan pengumpulan informasi, rerferensi dan studi literature, baik yang berasal dari jurnal, artikel ilmiah, karya tulis, thesis dan disertasi dari topik yang sama.

4. Pengujian dan analisis sensor

   Sensor yang dibahas pada penelitian tugas akhir ini adalah sensor CO_2 MH-Z19. Sebelum memasuki tahap perancangan dan pembuatan alat, alangkah baiknya diketahui secara mendetail mengenai kinerja, efisiensi dan data yang dihasilkan oleh MH-Z19. Selain dari pembuatan antarmuka I^2C, penelitian tugas akhir ini menghasilkan analisis kinerja pada sensor CO_2 MH-Z19.

5. Perancangan Antarmuka I^2C

Perancangan merupakan tahap awal dan utama dari pembuatan suatu alat. Terdapat beberapa perancangan yang akan dilakukan dalam pembuatan antarmuka ini yaitu perancangan sistem, perancangan perangkat keras dan perancangan perangkat lunak. Dalam perancangan perangkat keras akan diketahui bahan-bahan yang diperlukan sebelum tahap pembuatan. Bahan yang diperlukan telah dilampirkan sesuai dengan tabel 3.2 sebelumnya. Perancangan lebih lanjutnya akan dijelaskan pada pembahasan 3.4.

6. Pembuatan Antarmuka I^2C

   Setelah perancangan antarmuka I^2C sesuai dengan tujuan awal maka tahap selanjutnya adalah eksekusi pembuatan antarmuka I^2C. Seperti dalam pembuatan rangkaian elektronika lainnya, alat ini dibuat dari *board* PCB yang telah dibentuk sesuai desain rangkaian. PCB ini akan dirangkai dengan komponen-komponen yang telah ditentukan dalam tahap perancangan.

7. Uji Coba dengan Arduino

   Pengujian antarmuka I^2C dengan arduino adalah tahap 1 pengujian dikarenakan saat uji coba sensor digunakan arduino sebagai mikrokontroler. Baik pada arduino dan antarmuka I^2C akan diprogram sesuai dengan perancangan awal. Apabila terdapat kesalahan dalam pengiriman data, maka antarmuka I^2C akan dievaluasi baik dari perangkat keras dan perangkat lunak. Pengujian terus dilakukan hingga mencapai hasil yang diinginkan.

8. Uji Coba dengan Raspberry Pi

   Perangkat uji coba selanjutnya adalah raspberry pi. Tujuan dilakukan uji coba dengan bermacam-macam perangkat ini adalah untuk mengetahui fleksibilitas dari antarmuka I^2C. pengujian dan evaluasi juga dilakukan hingga mencapai hasil yang sama seperti hasil dari uji coba dengan arduino.

9. Analisis Data

   Seluruh data yang diperoleh baik dari perancangan, pembuatan dan pengujian akan dianalisis untuk mengetahui kelebihan dan kekurangan dari perangkat antarmuka I^2C ini. Data utama yang diperlukan adalah data konsistensi komunikasi, kelancaran komunikasi dan kesamaan data baik yang dikirim dan yang diterima. Analisis ini diperlukan agar mengetahui hasil dari penelitian ini dan akan bisa digunakan untuk penelitian selanjutnya.

10. Kesimpulan

    Data dari setiap pengujian yang telah dianalisis lalu disimpulkan sebagai hasil dari segala kegiatan penelitian yang telah dilaksanakan.

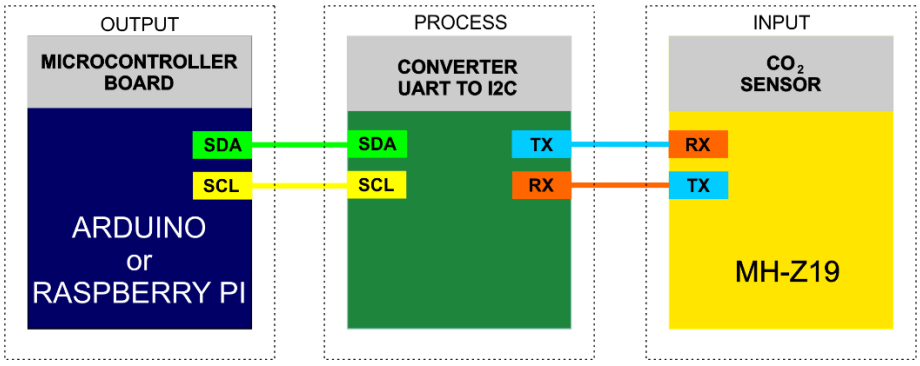
11. Stop

    Pada tahap ini seluruh prosedur pada penelitian tugas akhir mengenai pembuatan antarmuka I^2C pada sensor CO_2 MH-Z19 telah selesai.

3.4 Perancangan Antarmuka I^2C

3.4.1 Perancangan Sistem

Antarmuka I^2C atau *converter UART to* I^2C berfungsi sebagai jembatan antara sensor MH-Z19 dan *microcontroller board* yang akan memproses data dari sensor. Perancangan sistem ini dilakukan menggunakan blok diagram yang terdiri *input, process dan output.* Dari sisi *input* adalah sensor MH-Z19 yang akan mengirim data melalui komunikasi serial UART dan menuju blok *process* yaitu *converter UART to* I^2C. *converter* ini akan meneruskan data sensor menuju *output*. Namun, sebelum itu *converter* akan mengubah protokol komunikasi dari UART menjadi I^2C dengan beberapa program. *Output* yang bisa berupa arduino, raspberry pi atau *microcontroller board* lainnya akan dapat memperoleh dan mengakses data sensor melalui protokol komunikasi I^2C dari *converter*. Perancangan sistem yang telah dibentuk dalam blok diagram ditunjukkan pada gambar 3.2 dibawah ini.



Gambar 3.2 Blok Diagram Perancangan

3.4.2 Perancangan Perangkat Keras

Perancangan perangkat keras antarmuka I^2C untuk sensor MH-Z19 menggunakan *software* proteus 8.5 baik dari perancangan skematik rangkaian hingga desain PCB. Daftar komponen yang akan digunakan pada antarmuka I^2C dapat dilihat pada tabel 3.2 mengenai bahan penelitian. Seluruh komponen yang digunakan berjenis SMD (*Surface Mount Device*) yang rata-rata memiliki ukuran kecil dalam millimeter. Berikut adalah langkah-langkah dalam perancangan perangkat keras.

1. Pembuatan *Library* Komponen SMD

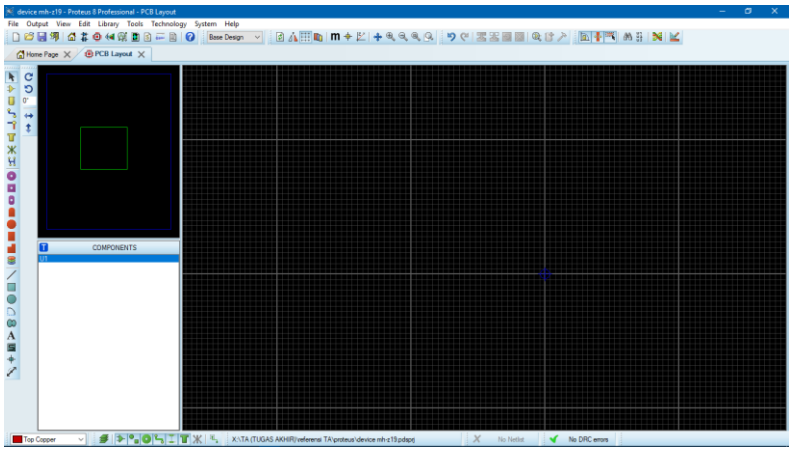
   Proteus adalah salah satu *software* perancangan rangkaian elektronik yang memiliki kelebihan untuk mensimulasikan rangkaian dan memberikan tampilan 3D dari rangkaianyang telah dirancang. Salah satu kelemahan proteus yaitu *library* komponen yang kurang lengkap terutama pada komponen baru seperti modul elektronik, konektor dan komponen SMD. Namun, proteus memiliki fitur untuk membuat *library* komponen secara manual dengan mengikuti ukuran yang sesuai dengan *datasheet* pada tiap komponen yang akan dibuat *library*-nya. Terdapat beberapa komponen yang dibutuhkan dalam perancangan antarmuka I^2C sensor MH-Z19, namun belum memiliki *library* pada proteus yaitu sensor CO_2 MH-Z19, *crystal* 16 MHz SMD dan LED 1206. Oleh sebab itu, sebelum ke tahap perancangan elektronika diharuskan melengkapi kebutuhan komponen dan berikut adalah cara pembuatan *library* dari ketiga komponen di atas.

a. Sensor CO_2 MH-Z19

Pembuatan *library* komponen pada proteus terdiri dari 2 tahap yaitu pembuatan *package* pada *PCB layout* dan pembuatan *device* pada bagian *schematic capture*. Berikut ini adalah penjelasan dari tiap tahap pembuatan library sensor CO_2 MH-Z19.

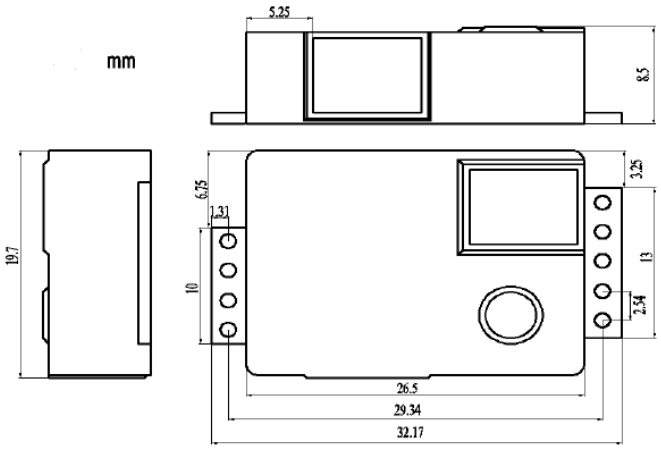
1) Pembuatan *package*

Pembuatan *package* adalah pembuatan *pattern PCB* atau posisi kaki komponen yang akan disolder pada *PCB*. *Package* akan dikerjakan pada *PCB layout* di aplikasi proteus. Berikut adalah tampilan dari *PCB layout* yang sering digunakan dalam mendesain jalur PCB juga pembuatan *package*.



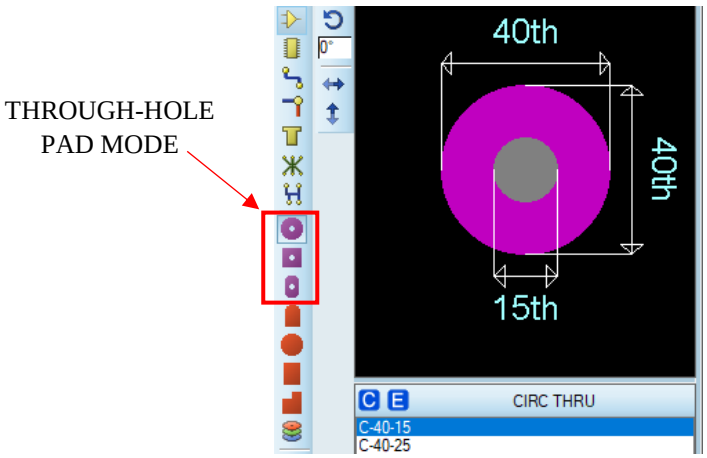
Gambar 3.3 *PCB Layout Proteus*

Sebelum mulai pembuatan *package* komponen diperlukan data komponen terutama data dimensi, struktur atau rekomendasi *pattern* pada *datasheet* komponen. Dimensi pada sensor MH-Z19 dapat dilihat secara detail pada gambar 3.4 pada halaman selanjutnya.



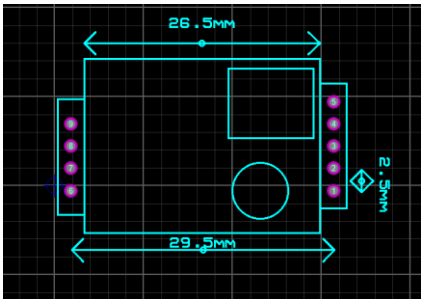
Gambar 3.4 Dimensi Sensor MH-Z19

Pembuatan *package* dapat dilakukan jika diketahui dimensi komponen terutama pada bagian pin atau kaki komponen seperti pada gambar di atas. Selain itu, juga diketahui bahwa pin sensor MH-Z19 berbentuk lubang atau konvensional. Sehingga digunakan *through-hole pad mode* dalam pembuatan *package* sensor MH-Z19 dengan dimnesi yang sesuai dari *datasheet*. Berikut adalah adalah *tools* yang digunakan.



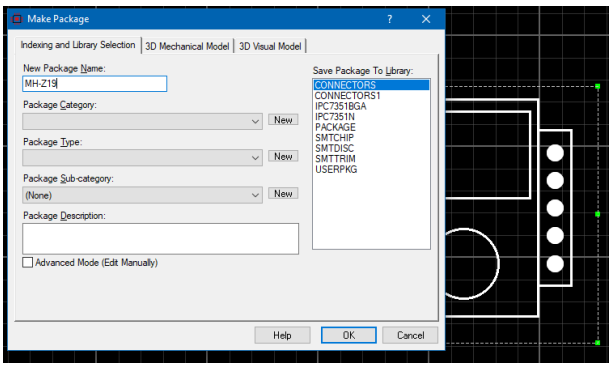
Gambar 3.5 *Through-hole pad mode*

Dengan dimensi dari *datasheet*, nomor pin dan mode dari pin yang akan digunakan, maka *package* bisa dirancang dan akan sesuai dengan komponen fisiknya. Dimensi tiap komponen menggukan satuan millimeter (mm) dengan ketelitian ±0,1 mm. Berikut ini adalah rancangan *package* sensor MH-Z19 sesuai dengan dimensi pada *datasheet*.



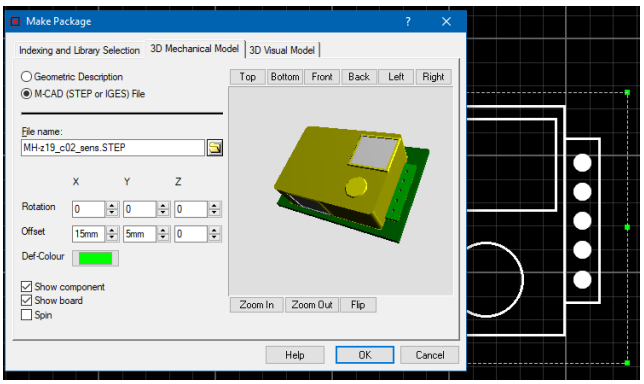
Gambar 3.6 Rancangan *Package* MH-Z19

Jika seluruh dimensi, posisi dan letak rancangan komponen telah sesuai dengan *datasheet*, maka selanjutnya adalah pembuatan *package*. *Select* seluruh rancangan kecuali dimensi lalu klik kanan mouse dan memilih *make package* pada pilihan paling bawah. Isi kolom sesuai dengan spesifikasi dan keterangan sensor. Berikut adalah tampilan dari *make package.*



Gambar 3.7 *Make Package* Proteus

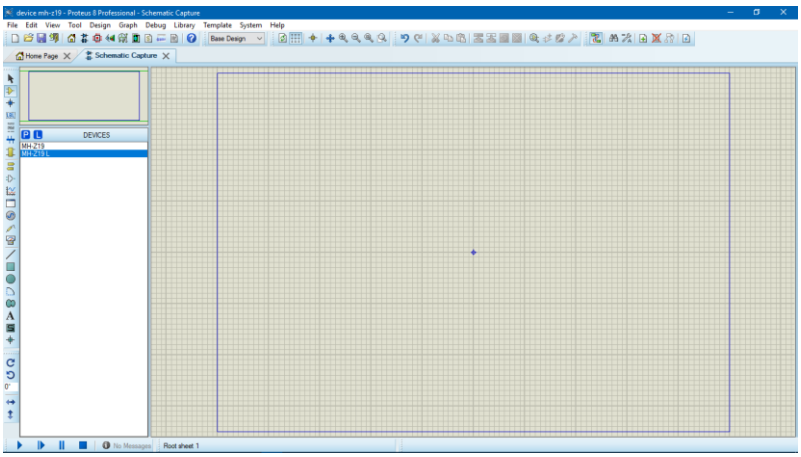
Langkah terakhir dan hanya bersifat opsional adalah menambahkan tampilan 3D komponen. *File* 3D bisa dibuat dengan aplikasi desain 3D atau bisa mendownload dari internet. *File* 3D yang digunakan menggunakan format .STEP atau .STP dan meletakkannya pada folder MCAD proteus. berikut adalah penambahan file 3D dalam *package*.



Gambar 3.8 3D *Mechanical Package* Proteus

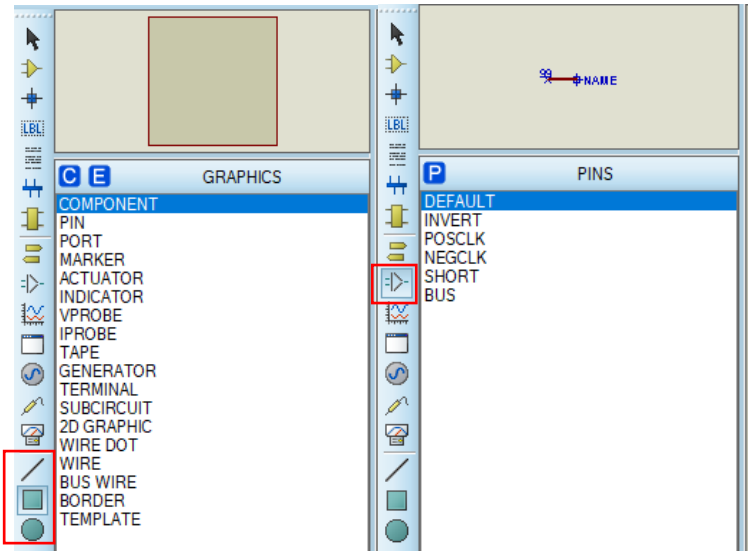
2) Pembuatan *Device*

Device dirancang pada bagian *schematic capture* dalam aplikasi proteus dengan tampilan sebagai berikut.



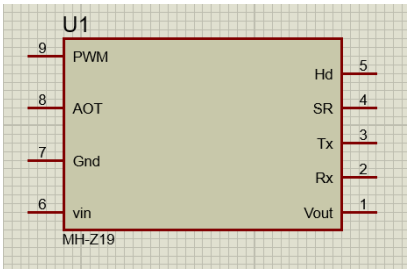
Gambar 3.9 *Schematic Capture* Proteus

Pada tahap ini dilakukan perancangan komponen dan penentuan tiap pin sesuai dengan *datasheet*. *device* bisa dirancangan dengan membuat dan meniru simbol komponen elektronika yang sudah ada atau membuat manual jika tidak ada simbol yang sesuai. *Tools* yang dominan digunakan pada pembuatan *device* ini adalah *tools graphics* sebagai pembuat simbol dan *tools pins* untuk menambahkan pin komponen beserta pengaturan nomor dan jenis pin. Berikut adalah tampilan kedua *tools* tersebut.



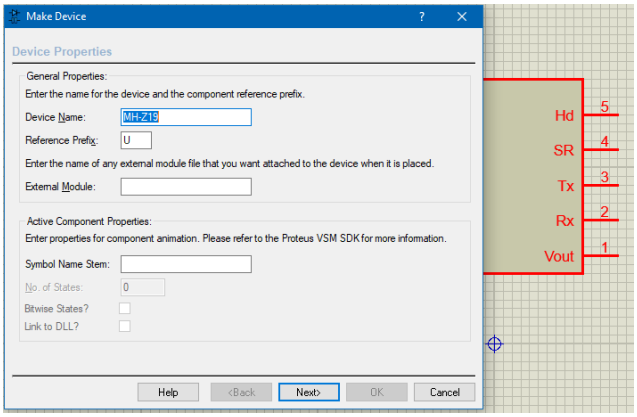
Gambar 3.10 *Tools Graphic* dan *Pins* Proteus

Kedua *tools* pada gambar 3.10 di atas sudah sangat cukup untuk membuat simbol *device*. Berikut adalah simbol dari hasil perancangan *library* sensor MH-Z19 yang ditunjukkan pada gambar 3.11 pada halaman selanjutnya.



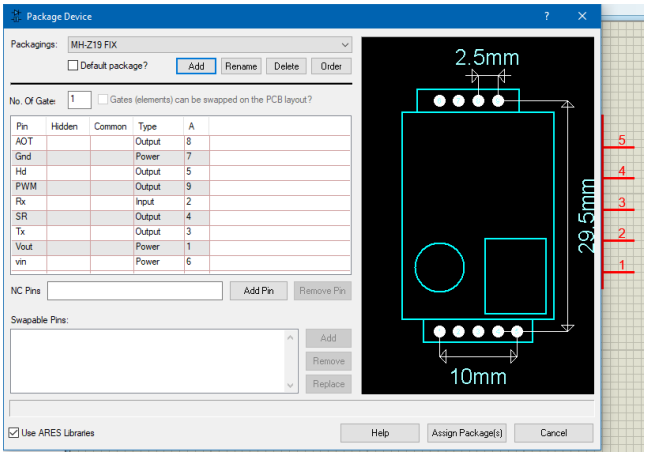
Gambar 3.11 Simbol Sensor MH-Z19

Agar simbol ini bisa digunakan bersamaan dengan *package* sensor MH-Z19 yang telah dikerjakan sebelumnya maka, simbol harus dijadikan sebuah *device.* Caranya adalah dengan klik kanan pada simbol yang telah di-*select* dan memilih pilihan *make device.* Maka akan muncul jendela baru untuk pengisian spesifikasi *device* seperti yang ditunjukkan pada gambar 3.12 berikut ini.



Gambar 3.12 Jendela *Make Device* Proteus

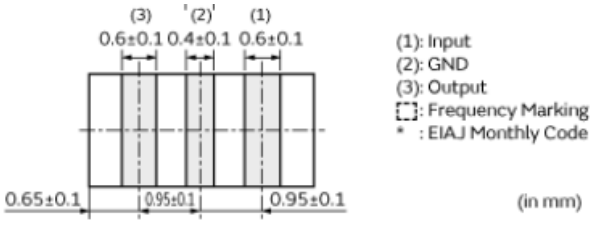
Pada jendela berikutnya terdapat pengaturan untuk menambahkan *package* yang sebelumnya telah dibuat dan akan digunakan saat perancangan *PCB layout.* Jendela *package* ditampilkan pada gambar 3.13 beserta pengaturan pin *device*.



Gambar 3.13 *Package Device MH-Z19*

b. Crystal 16 MHz SMD CSTCE16MOV53-R0

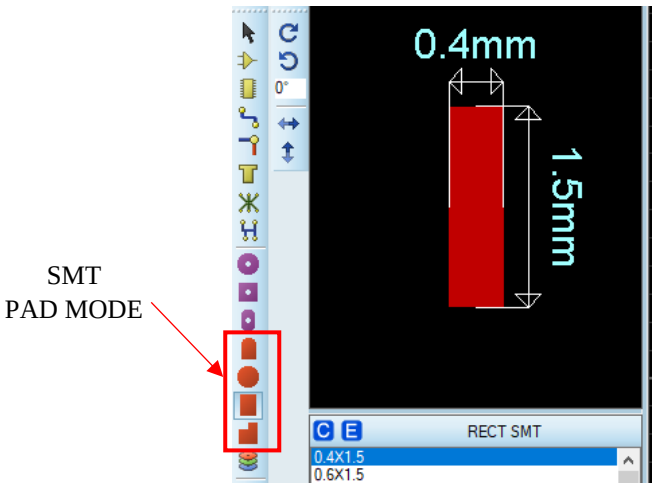
Pembuatan *library* pada *crystal* CSTCE16MOV53-R0 sama seperti pada pembuatan *library* sensor CO_2 MH-Z19 sebelumnya yaitu pembuatan *package* dan *device*. Namun, terdapat sedikit perbedaan dalam pembuatan *package* karena jenis komponen komponen *crystal* berbeda. Hal ini ditunjukkan pada data dimensi dan struktur pada *datasheet* berikut.



Gambar 3.14 Dimensi *Crystal CSTCE16MOV53-R0*

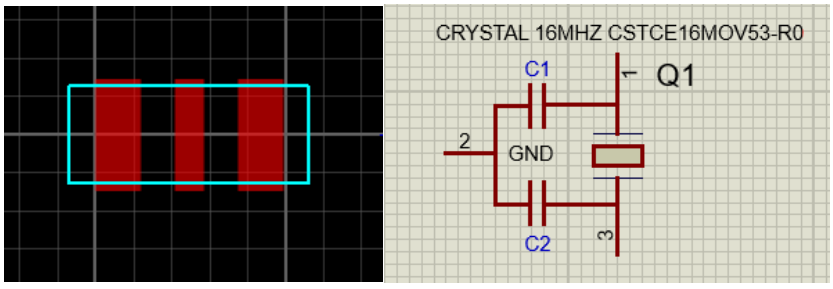
Berdasarkan gambar 3.14 di atas diketahui bahwa tipe *pad* yang digunakan berbeda dikarenakan komponen ini berjenis SMD

sehingga pin atau kaki komponen langsung menempel pada sisi tembaga PCB. Oleh sebab itu, *tools* yang digunakan dalam perancangan *package* berbeda dari yang digunakan pada perancangan sensor MH-Z19. *Tools* tersebut adalah *SMT pad mode.* Berikut adalah *tools* yang digunakan.



Gambar 3.15 *Tools SMT Pad Mode*

Selain perbedaan pada *tools* yang digunakan tidak ada lagi perbedaan dari proses pembuatan *library crystal* baik dari *package* maupun *device.* Berikut adalah *package* dan *device* pada *library crystal* CSTCE16MOV53-R0.

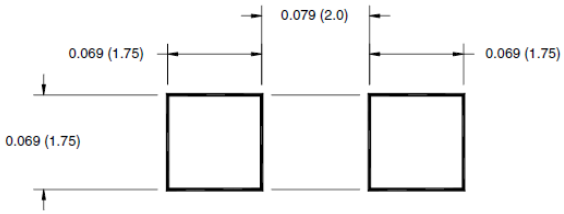


(A) (B)

Gambar 3.16 *Package (A) dan Device (B) CSTCE16MOV53-R0*

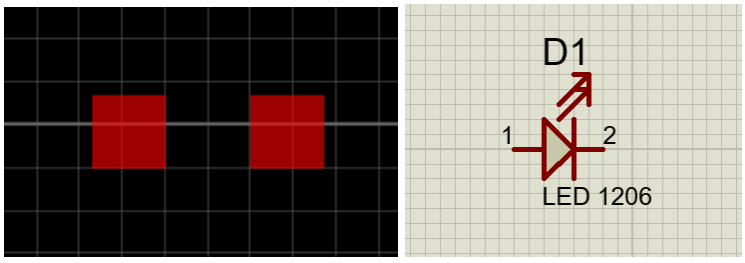
c. LED 1206

Sama seperti *crystal* CSTCE16MOV53-R0, LED 1206 juga berjenis SMD sehingga juga menggunakan *pad mode.* Namun, pada *datasheet* LED ini terdapat rekomendasi ukuran atau dimensi *pattern*. Sehingga mempermudah dalam perancangan *pattern* LED. Berikut adalah gambar dimensi yang dimaksud.



Gambar 3.17 Dimensi *Pattern* LED 1206

LED memiliki simbol tersendiri. Oleh karena itu, simbol LED 1206 bisa menggunkan simbol LED yang telah ada. Berikut ini adalah hasil dari perancangan *package* dan *device* dalam pembuatan *library* LED 1206 yang ditunjukkan pada gambar 3.18 berikut.

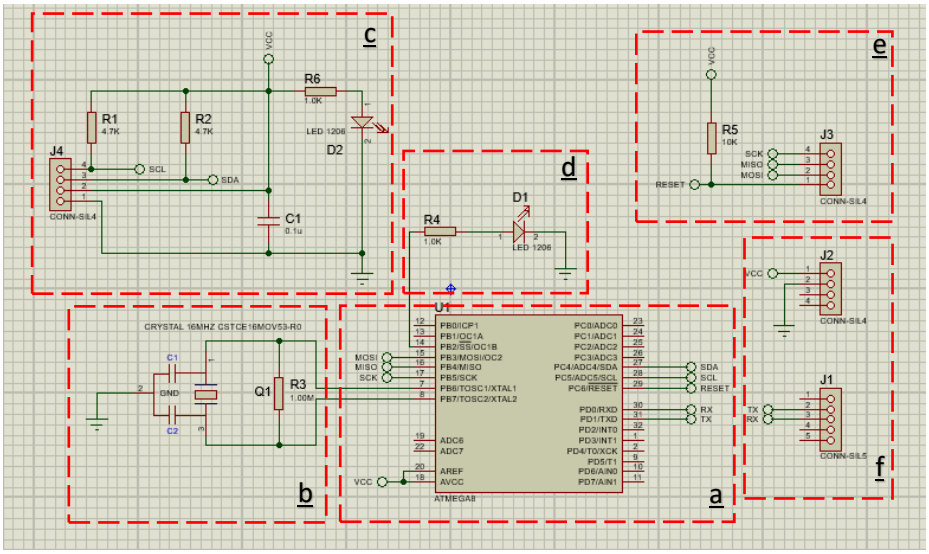


(A) (B)

Gambar 3.18 *Package (A) dan Device (B) LED 1206*

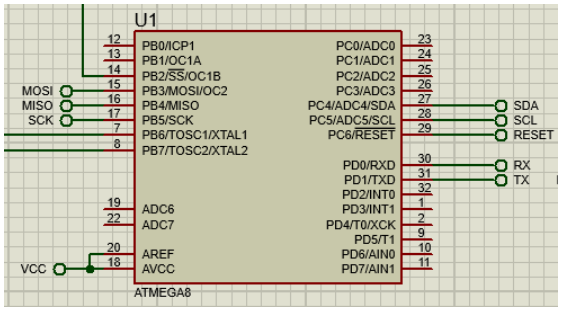
2. Perancangan Skematik Antarmuka I^2C

Perancangan skematik dilakukan untuk menghubungkan tiap komponen yang akan digunakan. Pada aplikasi proteus, perancangan skematik atau rangkaian elektronika dilakukan pada *schematic capture*. Antarmuka I^2C menggunakan komponen yang sama dalam rangkaian sistem minimum mikrokontroler yaitu mikrokontroler, *crystal*, resistor, kapasitor dan sebagainya. Seluruh komponen yang digunakan berjenis SMD yang memiliki ukuran lebih kecil dibandingkan ukuran komponen pada umumnya. Hal ini bertujuan untuk menghasilkan modul antarmuka I^2C yang berukuran sama dan terhubung langsung dengan *board sensor* MH-Z19. Adapun perancangan antarmuka I^2C menggunakan aplikasi proteus ditunjukkan pada gambar 3.19 di bawah ini dan disertai penjelasan rangkaian.



Gambar 3.19 Skematik Antarmuka I^2C Sensor MH-Z19

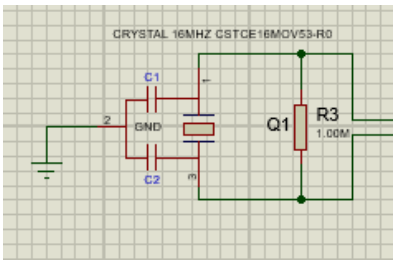
a. Mikrokontroler



Gambar 3.20 Mikrokontroler Atmega 8

Mikrokontroler yang akan digunakan sebagai pusat pengendali adalah atmega 8, karena kebutuhan pin yang tidak terlalu banyak dan program yang akan dimasukkan pada antarmuka I^2C ini tidak terlalu besar dan kompleks. Sehingga, spesifikasi atmega 8 yang hanya memiliki memori *flash* sebesar 8 Kb sudah sangat tercukupi. Dapat diperhatikan bahwa tidak semua pin pada atmega 8 digunakan. Hanya pin yang berhubungan dengan power, antarmuka dan komunikasi serta pin *crystal (XTAL)* yang digunakan. Terdapat 3 antarmuka atau komunikasi yang digunakan yaitu SPI, UART dan I^2C.

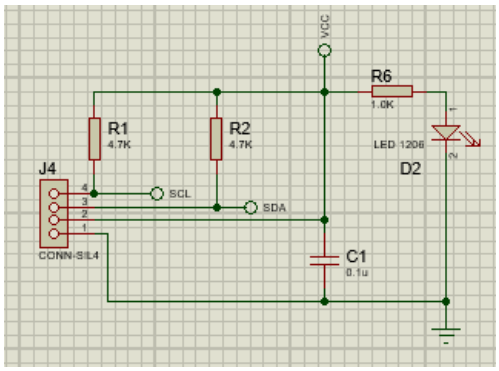
b. *Crystal*



Gambar 3.21 Rangkaian *Crystal*

Crystal yang digunakan adalah *ceramic resonator* 16 Mhz CSTCE16MOV53-R0 yang berjenis SMD. Pada *datasheet*, *crystal* ini telah memiliki 2 kapasitor 22 pf. Jadi tidak dibutuhkan lagi penambahan kapasitor pada *crystal.* Penggunaan *crystal* ini berdasarkan pada rancangan arduino uno R3. Terdapat penambahan resistor 1 MΩ secara paralel. Masih belum diketahui pasti fungsi dari resistor tersebut. Namun, pada pada artikel yang dibuat oleh Ramon Cerda (2008) mengenai *Pierce-Gate Crystal Oscillator* mengatakan bahwa penambahan resistor secara paralel pada *crystal* berfungsi sebagai *resistor feedback.* Resistor ini untuk melinierisasikan *digital CMOS inverter* dan mengubah *logic gate* menjadi *analog inverter*. Selain itu *resistor feedback* ini juga membantu meningkatkan *start-up time crystal* saat seluruh sistem baru dihidupkan. (Cerda, 2008)

c. Rangkaian *power* dan I^2C



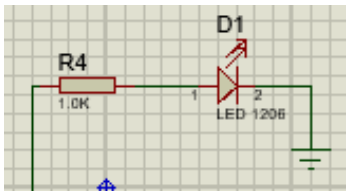
Gambar 3.22 Rangkaian *Power* dan I2C

Antarmuka I^2C ini dirancang untuk memperoleh daya sebesar 5v dari perangkat lain seperti arduino atau raspberry pi.

Diantara VCC dan *ground* dipasang kapasitor 100 nf sebagai *bypass capacitor*. Fungsi utama kapasitor ini adalah sebagai *filter* dari *noise* tegangan maupun tegangan riak yang bisa membahayakan komponen. Sehingga tegangan bisa lebih stabil.

Pada pin antarmuka I^2C dipasang *resistor pull-up* 4,7 kΩ baik pada pin SDA dan SCL. Hal ini dilakukan agar beberapa *device* bisa saling berkomunikasi tanpa ada konflik data, dimana satu *device* memberikan logika *low* sementara yang lain memberikan logika *high*. Selain itu, pada rangkaian ini terdapat LED yang berfungsi sebagai indikator *power on.*

d. Test LED



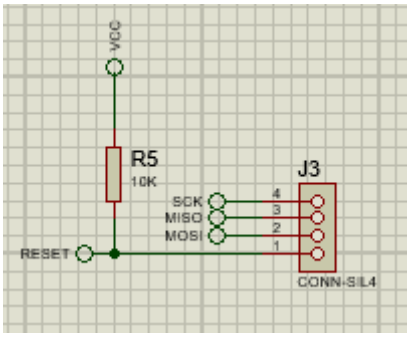
Gambar 3.23 Rangkaian *Test LED*

Fungsi pada rangkaian ini cukup sederhana yaitu sebagai rangkaian percobaan awal. Sebelum mikrokontroler dimasukkan program utama, terlebih dahulu dilakukan pengujian dengan menggunakan rangkaian di atas. Jika LED bisa dikendalikan, maka mikrokontroler sudah bersedia untuk program selanjutnya.

e. Rangkaian SPI atau ISP

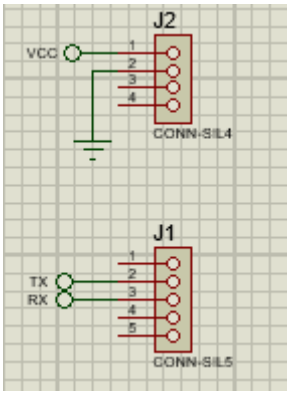
ISP (*in-circuit serial programming)* adalah rangkaian yang berfungsi sebagai media untuk memasukkan atau *upload* program ke

mikrokontroler. Sistem ini memanfaatkan komunikasi SPI yang terdiri dari MISO, MOSI, SCK dan disertai *reset*. Terdapat tambahan perangkat untuk memasukkan program melalui ISP ini yaitu downloader atau bisa menggunakan *arduino as ISP*. Pada pin *reset* menggunakan resistor *pull-up* 10 kΩ yang berfungsi untuk mencegah pin *reset* tidak menerima logika *low* (0) secara tidak sengaja. Hal ini dikarenakan saat pin *reset* memperoleh logika *low*, maka mikrokontroler akan me*reset* program dari awal.



Gambar 3.24 Rangkaian ISP

f. Konektor Sensor MH-Z19



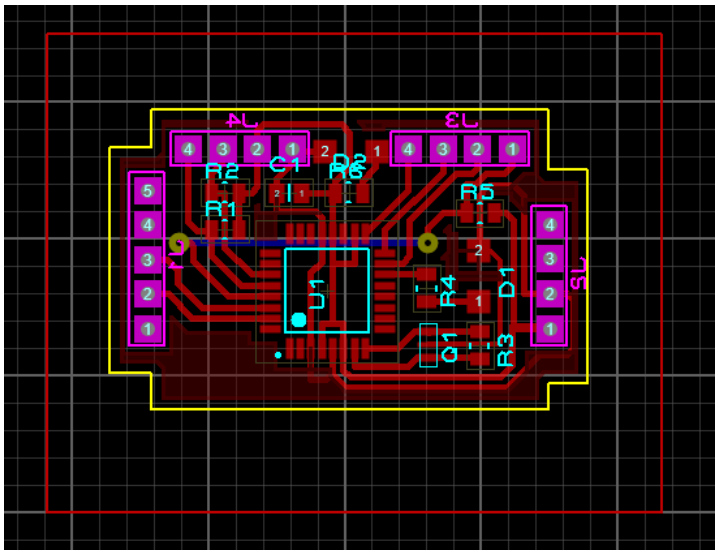
Gambar 3.25 Rangkain Konektor MH-Z19

Rangkaian pada gambar 3.25 adalah rangkaian konektor yang akan menghubungkan antarmuka I^2C dengan sensor CO_2 MH-

Z19. Konektor tersebut menghubungkan pin power sensor dan pin komunikasi UART. Antaramuka I^{2}C akan mengambil data dari MH-Z19 melalui komunikasi UART pada sensor.

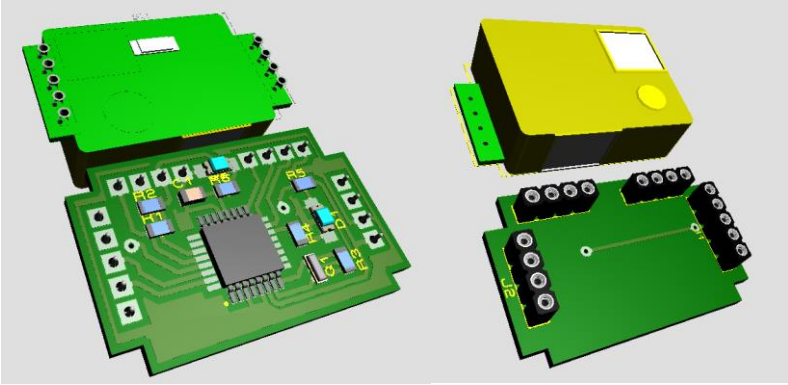
3. Perancangan Jalur PCB Antarmuka I^{2}C

Setelah skematik rangkaian selesai tanpa ada kesalahan, tahap selanjutnya adalah desain jalur PCB. Hal terpenting pada tahap ini adalah peletakan tiap komponen agar jalur yang dihasilkan tidak bersilangan dengan jalur yang lain. Ukuran PCB akan disesuaikan dengan ukuran sensor MH-Z19, sehingga bisa menyebabkan letak tiap komponen menjadi lebih rapat. Karena semua komponen yang digunakan berjenis SMD, maka *track* atau jalur tembaga menggunakan *top copper*. Gambar 3.26 berikut adalah *Layout PCB* antarmuka I^{2}C sensor MH-Z19.



Gambar 3.26 *Layout PCB Antarmuka I^{2}C*

Seperti sistem *plug and play*, antarmuka I^2C ini akan langsung terhubung dengan sensor MH-Z19 melalui *pin header* dan tanpa kabel tambahan. Ukuran antarmuka juga disesuaikan dengan dimensi sensor MH-Z19. Pada saat perancangan *layout*, *layer* yang digunakan oleh sensor MH-Z19 dan *pin header* adalah *solder side.* sehingga posisinya akan membelakangi antarmuka I^2C. Agar lebih mudah diamati baik dimensi maupun letak komponen, berikut adalah gambar 3D dari *layout PCB* dan sensor MH-Z19 menggunakan *3D visualizer* pada aplikasi proteus.



Gambar 3.27 Tampilan 3D Antarmuka I^2C dan MH-Z19

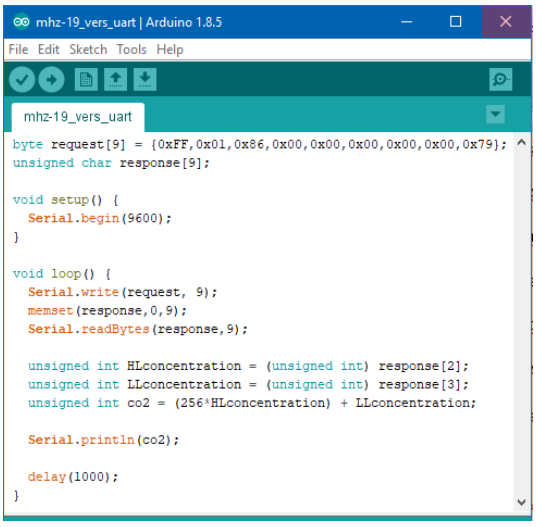
Antarmuka ini memiliki dimensi 3.5 cm x 2.3 cm. dibandingkan dengan sensor MH-Z19, perbedaan dimensi mencapai ±0.2 cm. kelebihan dimensi ini dilakukan karena terdapat beberapa Jalur yang harus melewati bagian luar *pin header*. Struktur PCB juga dibentuk mengikuti struktur dari sensor MH-Z19.

3.4.3 Perancangan Perangkat Lunak

Perancangan perangkat lunak adalah perancangan program mikrokontroler pada perangkat antarmuka I^2C atau *converter UART to* I^2C dan pada perangkat *microcontroller board* yang mengakses *converter* tersebut. Pemrograman dilakukan menggunakan arduino IDE dan *upload* program menuju mikrokontroler menggunakan sistem ISP (*In-System Chip Programming*). ISP memanfaatkan protokol komunikasi SPI yang menggunakan pin MISO, MOSI, dan SCK.

1. Program Pengukuran CO_2 Sensor MH-Z19

   Program utama pada antarmuka I2C adalah pembacaan gas CO_2 yang dilakukan oleh sensor MH-Z19. Dari dua jenis *output* data sensor yang dihasilkan, dipilih data hasil keluaran dari UART yang menggunakan pin Tx dan Rx sesuai dengan perancangan perangkat keras sebelumnya. Berikut adalah program utama dari pembacaan CO_2 sensor MH-Z19 menggunakan arduino IDE.



Gambar 3.28 Program CO_2 Sensor MH-Z19

Berdasarkan kode pemrograman pada gambar 3.28, MH-Z19 bekerja dengan *array byte* yang berjumlah 9 *byte*. Untuk memperoleh data pengukuran CO_2, arduino atau antarmuka I^2C harus mengirimkan *command*. *Command* yang dikirim juga berupa *array* 9 *byte* data melalui UART. Berikut adalah *command* dan data *command* yang dikirim.

(a)
```
byte request[9] = {0xFF,0x01,0x86,0x00,0x00,0x00,0x00,0x00,0x79};
```
(b)
```
Serial.write(request, 9);
```

Gambar 3.29 Data *Command (a)* dan *Command (b)*

Setelah *command* diterima, sensor CO_2 akan mengirimkan *array* 9 *byte* data yang diinisialisasikan dalam variabel *response* yang juga memiliki data pengukuran CO_2. Terdapat 2 fungsi yang digunakan untuk membaca data dalam sebuah *array* baru yaitu *Serial.readBytes()* dan membersihkan *array* untuk digunakan kembali yaitu *memset*. Dengan menggunakan *memset*, *array* yang sudah berisi data bisa dibersihkan dan diganti dengan data yang baru. Berikut ini adalah fungsi atau kode program yang digunakan.

(a)
```
memset(response,0,9);
```
(b)
```
Serial.readBytes(response,9)
```

Gambar 3.30 *Clear Array (a)* dan *Read Bytes* Data Dari Serial *(b)*

Data yang diperoleh dari sensor MH-Z19 masih berupa 9 *byte array* atau masih mentahan data. Data ini perlu diolah kembali untuk memperoleh nilai dari pengukuruan CO_2. Tidak seluruh *byte* yang terdapat pada 9 *byte array* tersebut yang berisi data pengukuran.

Berdasarkan *datasheet* dari MH-Z19, data pengukuran CO_2 hanya terdapat pada *byte* ke-dua dan *byte* ke-3. Kedua nilai *byte* ini akan dikombinasikan menjadi 2 *byte* data pengukuran CO_2. Rumus yang digunakan juga terdapat pada *datasheet* MH-Z19. Berikut adalah kode untuk menghitung data pengukuran dari kedua *byte* dalam *array response*.

```
unsigned int HLconcentration = (unsigned int) response[2];
unsigned int LLconcentration = (unsigned int) response[3];
unsigned int co2 = (256*HLconcentration) + LLconcentration;
```

Gambar 3.31 Kode Pembacaan Data Pengukuran CO_2 MH-Z19

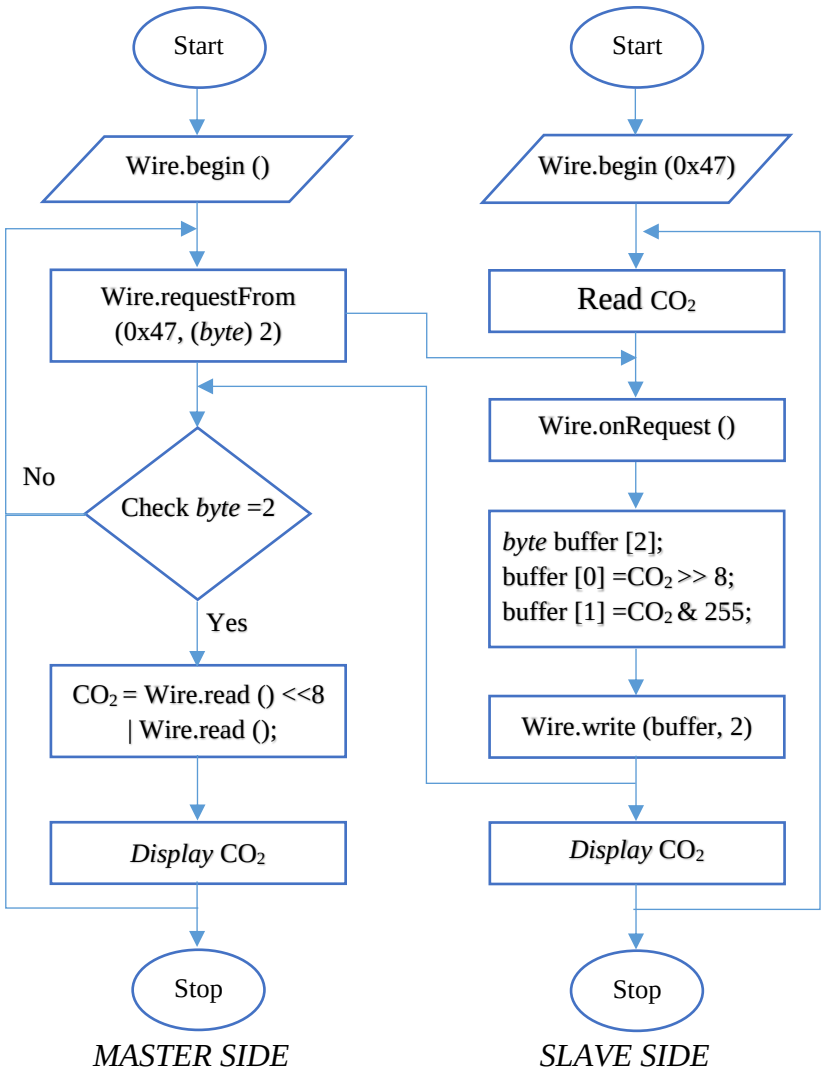
2. Program Transmisi Data Via I^2C

Antarmuka I^2C akan bekerja persis seperti arduino yang bersifat *slave*. Terdapat 3 metode pemrograman transmisi data antar arduino melalui protokol I^2C. Pada tiap metode yang akan dipaparkan terdapat arduino *master* sebagai *microcontroller board* dan arduino *slave* sebagai antarmuka I^2C yang akan mentransmisiskan data sensor MH-Z19 menuju arduino *master*. Ketiga metode tersebut adalah sebagai berikut.

a. *Master Request To Slave*

Pada metode ini *slave* bekerja mengirimkan data sensor MH-Z19 berdasarkan *request* dari *master*. *Slave* akan memiliki alamat tersendiri yaitu 0x47, kemudian *master* akan melakukan request ke alamat tersebut dengan fungsi *Wire.requestFrom(address slave, byte data).* Jumlah data yang dikirim oleh *slave* sebanyak 2 *byte*

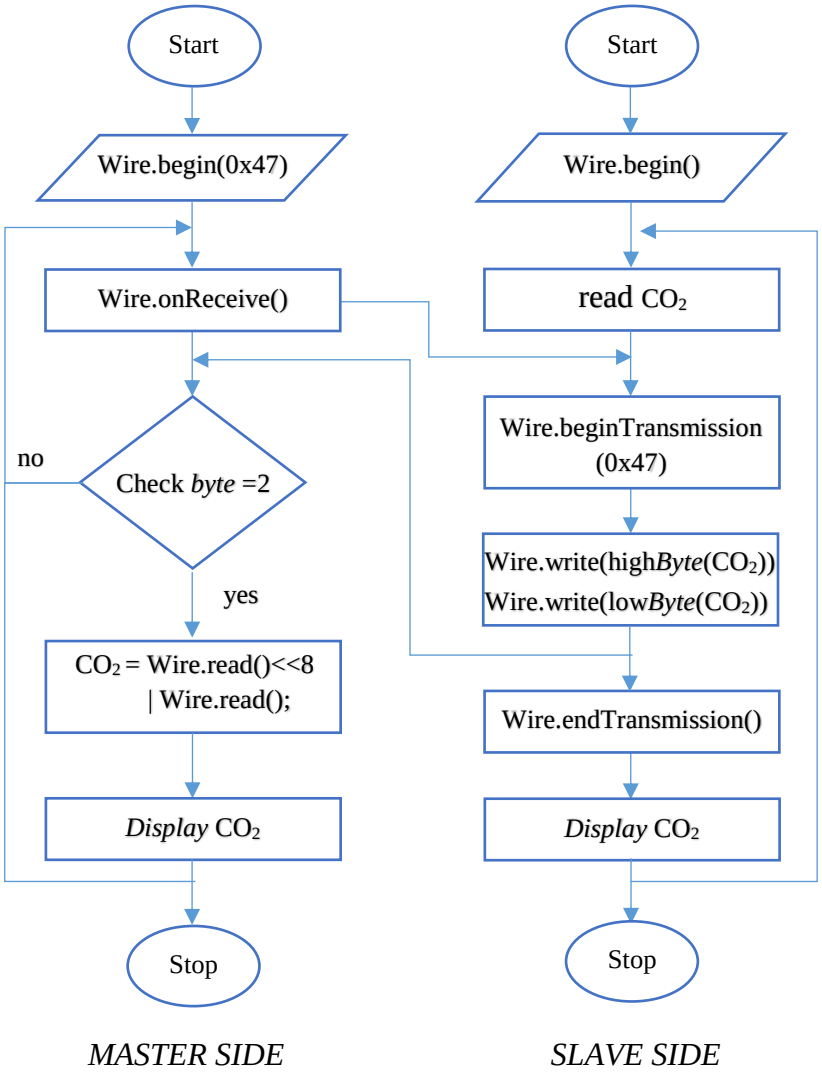
dikarenakan data CO_2 bisa mencapai 2000 PPM. 2 *byte* data ini akan dipecah terlebih dahulu menjadi *array* yang memiliki 2 *byte* data dan dikirim menuju *master*. *Master* menerima data *array* yang memiliki 2 *byte* data dan mengkombinasikannya menjadi 2 *byte* yang utuh Berikut adalah *flowchart* pemrograman baik pada sisi *master* dan *slave*.



Gambar 3.32 *Flowchart Master Request To Slave*

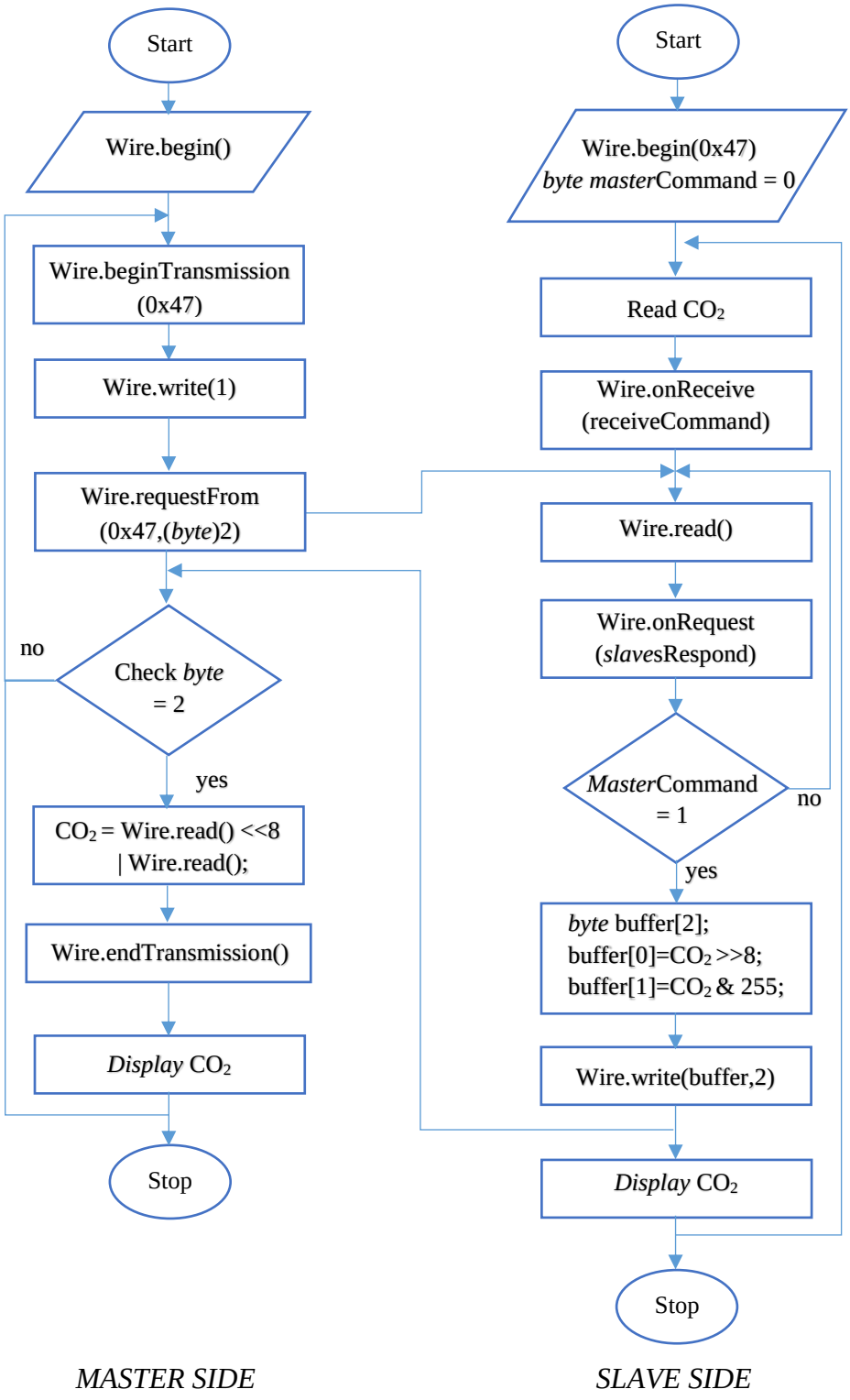
b. *Master Receive From Slave*

Pada metode ini *slave* akan mengirim langsung data MH-Z19 menuju *master* tanpa ada perintah apapun dari *master*. *Master* akan memiliki address tersendiri yang berkebalikan dari metode sebelumnya. *Master* memiliki fungsi *Wire.onReceive* dan bersedia menerima data apapun dan *slave* mengirim per *byte* data dari 2 *byte* data CO_2. Berikut adalah *flowchart*-nya.



Gambar 3.33 *Flowchart Master Receive From Slave*

c. *Master Send Command to Slave*



Gambar 3.34 *Flowchart Master Send Command to Slave*

Metode yang pada *flowchart* gambar 3.34 menunjukkan bahwa *slave* akan mengirim data menuju *master* saat *master* mengirim *command* yaitu angka "1". Saat *command* telah diterima, *slave* akan mengirim data seperti pada metode "*master* request from *slave*".

3.5 Pengujian dan Analisis Sensor

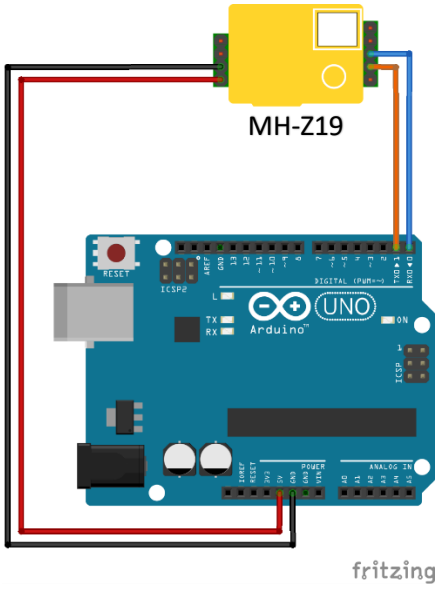
3.5.1 Pengujian Data Sensor CO_2 MH-Z19

Pengujian data pengukuran konsentrasi CO_2 pada sensor CO_2 MH-Z19 dilakukan pada 4 kondisi lingkungan tertentu berdasarkan sumber penghasil CO_2, yaitu:

1. Lingkungan tertutup
2. Lingkungan udara bebas siang dan malam
3. Lingkungan ruangan dengan manusia dan nafas manusia
4. Asap kendaraan bermotor

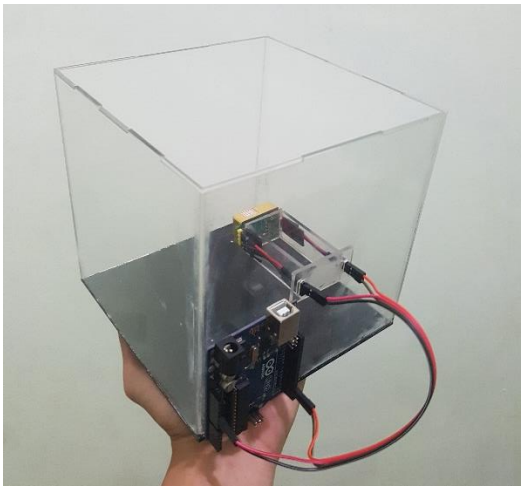
Pengujian ini dilakukan untuk mengetahui kecepatan *refresh-rate* sensor, kriteria MH-Z19 dan data konsentrasi CO_2 yang diukur pada lingkungan yang berbeda. Dalam uji coba ini dibutuhkan arduino uno, MH-Z19, Komputer dan wadah uji coba khusus untuk lingkungan tertutup. Selain itu pengujian ini hanya dilakukan untuk membuktikan kinerja sensor dengan mengamati perubahan data ditiap perubahan lingkungan yang ada. Data pengukuran yang dihasilkan tidak diteliti lebih lanjut karena tugas akhir ini lebih berfokus pada pembuatan

antarmuka I^2C sensor. Berikut adalah pengkabelan pada sensor MH-Z19 dan arduino melalui protokol UART.



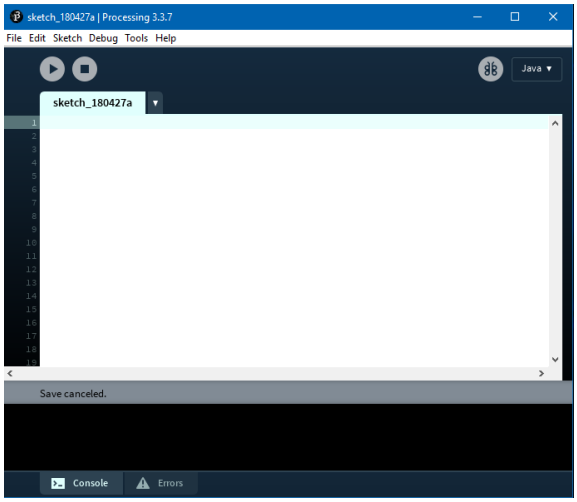
Gambar 3.35 *Wiring Diagram* Sensor MH-Z19 dan Arduino

Khusus pada pengujian sensor di lingkungan tertutup, digunakan sebuah wadah uji khusus. Wadah uji ini berfungsi untuk melindungi sensor dari pengaruh udara luar atau sumber CO_2 lain. Sehingga diharapkan hasil dari pengukuran pada lingkungan tertutup akan selalu konstan. Oleh karena itu, dibuat sebuah wadah tertutup yang terbuat dari *acrylic* berbentuk kubus dengan dimensi 15 cm x 15 cm x 15 cm. sensor diletakkan didalam wadah uji dengan kabel yang menuju ke luar wadah uji agar arduino bisa mengakses sensor dari luar. Berikut adalah gambar dari wadah uji sensor MH-Z19 yang ditunjukkan pada gambar 3.36 pada halaman selanjutnya.



Gambar 3.36 Wadah Uji Pengujian Sensor

Program yang digunakan dalam pengujian data sensor sama dengan program utama pengukuran CO_2 sensor MH-Z19 pada gambar 3.28 pada perancangan perangkat lunak. Data yang diambil berasal dari data UART sensor MH-Z19. Karena pengujian ini dilakukan untuk mengambil data sensor, maka diperlukan program tambahan untuk menyimpan seluruh data pengukuran CO_2 dari sensor MH-Z19. Jika dilakukan pencatatan data manual oleh media kertas dan sebagainya akan mempersulit pengujian. Oleh karena itu, digunakan aplikasi *processing* yang mampu menghasilkan aplikasi berbasis GUI (*graphic user interface)* dan menyimpan data kedalam komputer. *Processing* sendiri menggunakan bahasa java sebagai bahasa pemrograman. Arduino akan mampu berkomunikasi dengan *processing* melalui komunikasi serial menggunakan kabel USB. Tampilan *processing IDE* sebagai media pembuatan dan pemrograman GUI ditunjukkan pada gambar 3.37 pada halaman selanjutnya.



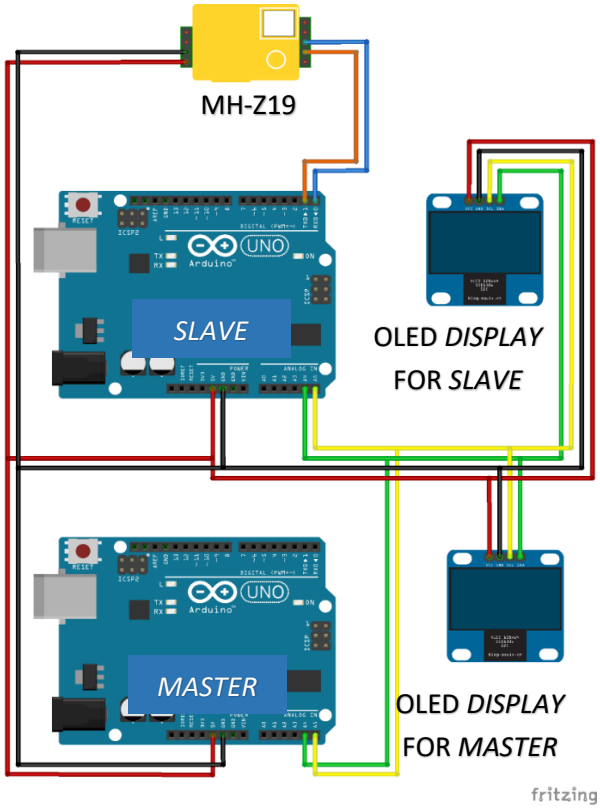
Gambar 3.37 *Processing IDE*

Data pengukuran sensor akan dikirim melalui serial menuju komputer sama halnya dengan serial monitor pada arduino IDE. Namun, *processing* mampu melakukan lebih banyak hal seperti memberikan perintah melalui tombol, menampilkan data pada layar komputer dengan lebih menarik sambil menyimpan data tersebut. Data yang disimpan menggunakan format .csv dan bisa diproses pada *microsoft excel*. Berikut adalah program sederhana *processing* untuk menyimpan data pengukuran CO_2 pada pengujian data sensor yang ditunjukkan pada gambar 3.38 di bawah.

Gambar 3.38 Program Monitoring dan Penyimpan Data Sensor

3.5.2 Pengujian Transmisi Data Antar Arduino Melalui Protokol I^2C

Pengujian ini dilakukan dengan mengirim data sensor MH-Z19 dan menerima datanya antar arduino melalui komunikasi I^2C. Tujuan pengujian ini adalah untuk mengetahui keberhasilan dan sistem transmisi data antar perangkat dalam 1 bus protokol I^2C. Arduino yang bersifat *slave* akan bertugas mengirim data sensor MH-Z19 menuju arduino *master*. Data sensor pada tiap arduino akan ditampilkan pada tiap *display oled*. Program yang digunakan adalah program arduino yang transmisi data via I^2C pada perancangan perangkat lunak di pembahasan sebelumnya. Berikut adalah pengkabelan (wiring diagram) pengujian yang dilakukan.



Gambar 3.39 *Wiring Diagram* Pengujian Transmisi Antar Arduino

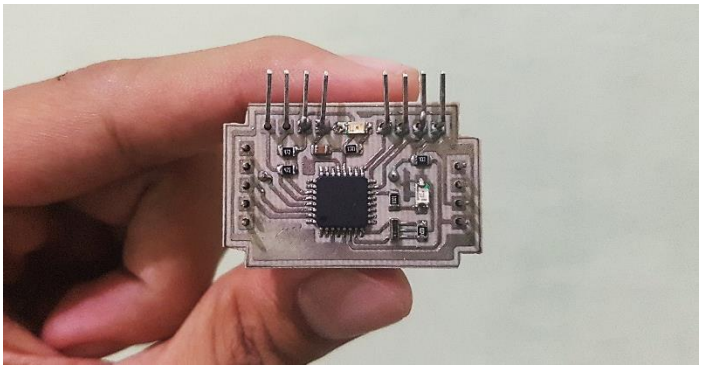
3.6 Pembuatan Antarmuka I^2C

Antarmuka I^2C adalah rangkaian sistem minimum dengan skala kinerja yang lebih rendah daripada arduino uno. Tahap pembuatan antarmuka I^2C ini akan dilakukan setelah perancangan selesai, terutama pada perancangan perangkat keras. Proses Pembuatan antarmuka I^2C dari sisi perangkat keras ini sama seperti tahap dalam pembuatan rangkaian elektronik lainnya, yaitu:

1. Pembuatan skematik dan perancangan *layout* PCB yang telah dilakukan pada tahap perancangan perangkat keras.
2. Pencetakan *layout* PCB pada kertas *glossy*, transparan atau *transfer paper*.
3. Penyetrikaan PCB yang telah ditempel kertas berisi *layout* PCB sebelumnya. Hal ini untuk menempelkan tinta kertas *layout* pada PCB.
4. Pelarutan PCB atau *etching* menggunakan pelarut PCB atau FeCl (*Ferri Chloride).* Larutan FeCl akan melarutkan tembaga PCB yang tidak ditempeli oleh tinta dari kertas *layout*.
5. *Troubleshooting* jalur PCB menggunakan multimeter. Hal ini bertujuan untuk mengetahui kondisi tiap jalur yang ada dan menghindari *short-circuit* pada jalur yang seharusnya tidak terhubung.
6. Pengeboran PCB yang disesuaikan dengan *layout* yang ada.
7. Proteksi PCB dengan cairan $AgNO_3$ atau silver nitrat. Dengan melapisi PCB dengan cairan ini akan melindungi PCB dari korosi, mempermudah tenol menempel di tembaga dan membuat warna tembaga menjadi silver.
8. Pemasangan dan penyolderan komponen merupakan tahap akhir dari pembuatan PCB. Teknik penyolderan yang akan digunakan adalah

penyolderan komponen SMD yang membutuhkan ketelitian, karena ukuran tiap komponen yang sangat kecil.

Setelah menyelesaikan tahap-tahap pembuatan PCB antarmuka I^2C tersebut, antarmuka I^2C akan dicek ulang lagi agar tidak terjadi *short-circuit* yang tidak diinginkan dari hasil solderan. Berikut ini adalah gambar dari hasil pembuatan antarmuka I^2C menggunakan komponen SMD.



Gambar 3.40 Rangkaian Antarmuka I^2C

Sebelum masuk ke tahap pemrograman, antarmuka I^2C akan masuk tahap *fuse-bits* mikrokontroler. Hal ini dikarenakan, mikrokontroler atmega 8 menggunakan sumber *clock* eksternal yaitu *crystal* CSTCE16MOV53-R0. Sehingga perlu beberapa perubahan pada *hfuse* (*high fuse byte*) dan *lfuse* (*low fuse byte).* Perubahan ini disesuaikan dengan jenis *crystal* yang digunakan. Terdapat beberapa aplikasi yang mampu mengatur *fuse-bit* dan salah satunya adalah *extreme-burner*. Berikut adalah setting *fuse-bits* untuk antarmuka I^2C.

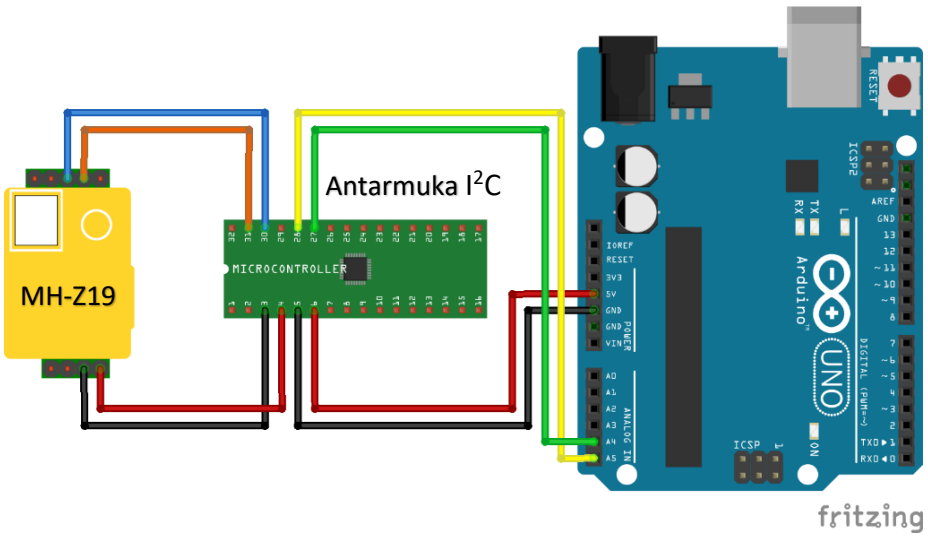
hfuse awal : D9 ⟶ hfuse baru : C9

lfuse awal : E1 ⟶ lfuse baru : CF

3.7 Pengujian Antarmuka I²C

3.7.1 Pengujian Antarmuka I²C dengan Arduino

Tahap pengujian diperlukan untuk melakukan analisis kinerja dari antarmuka I²C yang telah dikerjakan. Pengujian awal menggunakan arduino sebagai *board microcontroller* yang memperoleh data terusan MH-Z19 dari antarmuka I²C. Arduino dipilih sebagai pengujian awal dikarenakan pada tahap uji transmisi data via I²C memanfaatkan arduino. Berikut adalah *wiring diagram* pengujian antarmuka I²C. kondisi fisik antarmuka I²C pada gambar 3.41 di bawah hanya perumpaan dan tidak sesuai dengan hasil perancangan.



Gambar 3.41 Percobaan Antarmuka I²C dengan Arduino

Pemrograman yang digunakan pada antarmuka I²C adalah arduino. Antarmuka I²C akan dimasukkan program *bootloader* arduino terlebih dahulu sehingga program bisa di*upload* langsung melalui

arduino IDE. Program *bootloader* ini memiliki kapasitas 0,5 Kb memori *flash*.

Kode program yang digunakan sama seperti program transmisi antar 2 arduino via I^2C. Antarmuka I^2C akan menjadi *slave* yang mengirim data pengukuran sensor menuju *master* yaitu arduino.

```
#include <Wire.h>

#define SLAVE_ADDRESS 0x47

void setup() {
  Wire.begin(SLAVE_ADDRESS);
  Serial.begin(9600);
  Wire.onRequest(requestEvent);
}
```

(a)

```
#include <Wire.h>

void setup() {
  Wire.begin();
  Serial.begin(9600);
}
```

(b)

Gambar 3.42 *Setup* Antarmuka I^2C (a) dan *Setup* Arduino (b)

Antarmuka I^2C yang bersifat *slave* memiliki alamat tersendiri untuk mempermudah *master* mengirim *command* atau meminta data. Untuk mendukung komunikasi I^2C dibutuhkan *library* <Wire.h>. Pada antarmuka I^2C juga ditambahan kode *Wire.onRequest(requestEvent)* yang nantinya memberikan data dari fungsi *requestEvent* sesuai dari *request* atau permintaan *master*. Berikut ini adalah fungsi *requestEvent* yang berisi pengiriman data CO_2 dari sensor MH-Z19.

```
void requestEvent() {
  byte buffer[2];
  buffer[0] = co2 >> 8;
  buffer[1] = co2 & 255;
  Wire.write(buffer, 2);
  Serial.println(co2);
}
```

Gambar 3.43 *RequestEvent* Antarmuka I^2C

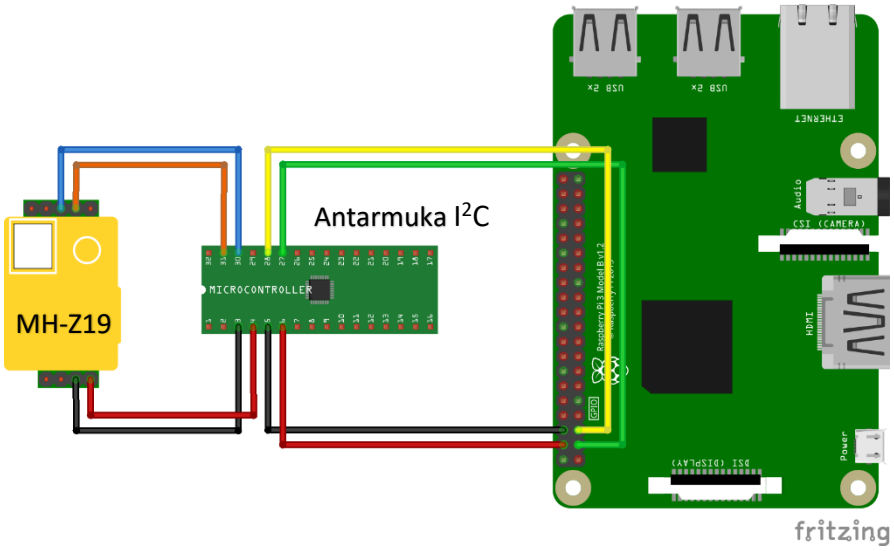
Komunikasi I^2C tidak bisa langsung mengirim 2 *byte* data utuh, namun bertahap tiap 1 *byte*. Oleh karena itu, pada *requestEvent* digambar 3.39 data CO_2 yang berukuran 2 *byte* dipisah menjadi 2 data berukuran 1 *byte* dan dikirim menuju *master*. Saat *master* menerima 2 data tersebut, *master* akan menggabungkan kedua data untuk menjadi 1 data berukuran 2 *byte*. Berikut adalah kode pada *master*.

```
int check = Wire.requestFrom(0x47,(byte)2);
  if(check == 2)
  {
    co2 = Wire.read() << 8 | Wire.read();
  }
```

Gambar 3.44 *Request Master*

3.7.2 Pengujian Antarmuka I^2C dengan Raspberry Pi

Pengujian selanjutnya adalah dengan menggunakan perangkat mikrokontroler atau mikroprosesor yaitu raspberry pi 3. Hal ini bertujuan untuk menentukan tingkat fleksibilitas dari antarmuka I^2C sehingga bisa digunakan oleh berbagai *device.* Raspberry pi merupakan komputer mini yang sering digunakan untuk proyek elektronika, IOT, dan sebagainya. Fungsi dan kinerja layaknya sebuah komputer dengan tampilan grafis, wifi dan tambahan pin dan port yang bisa dimanfaatkan. Pemrograman yang digunakan pada raspberry pi 3 adalah python 2 maupun python 3. Pin I^2C juga tersedia sehingga antarmuka I^2C bisa mengirim data menuju raspberry pi. Gambar 3.45 adalah *wiring diagram* pengujian antarmuka I^2C dengan raspberry pi.



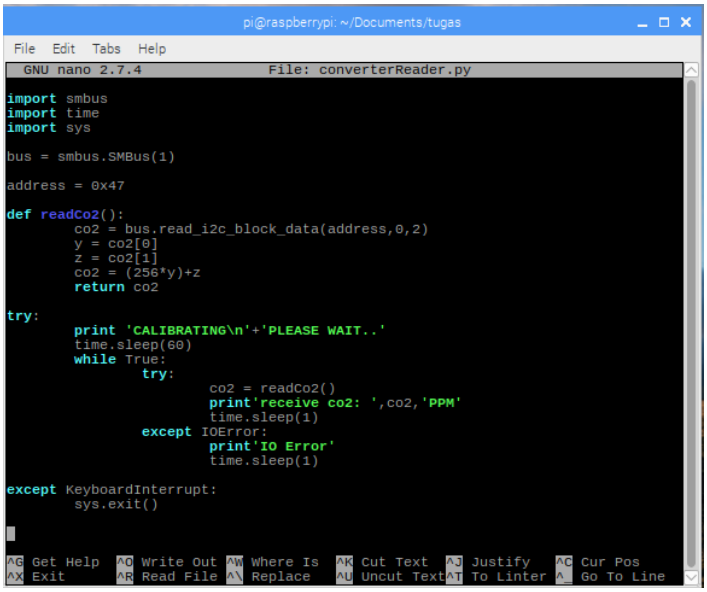
Gambar 3.45 Percobaan Antarmuka I^2C dengan Raspberry Pi

Keterangan kabel :

= Power 5v = Tx *to* Rx = SDA

= Ground = Rx *to* Tx = SCL

Kode program yang digunakan pada antarmuka I^2C sama seperti kode pada pengujian antarmuka I^2C dengan arduino. Data pengukuran CO_2 dibagi menjadi 2 data per 1 *byte* dalam sebuah *array* dan dikirim menuju *master* yaitu raspberry pi. Karena menggunakan bahasa python sebagai Bahasa pemrograman, sehingga kodenya sedikit berbeda dari arduino yang menggunakan Bahasa C yang diperbaharui. Namun, cara kerja program utama tetap sama yaitu mengkombinasikan 2 data dalam *array* menjadi sebuah data utuh berukuran 2 *byte*. Berikut adalah kode program pada raspberry pi yang ditunjukkan pada gambar 3.46.



Gambar 3.46 Program pada Raspberry Pi

Pada program di atas, ditunjukkan bahwa raspberry bekerja dengan membaca *block* data berupa *array* berjumlah 2 *byte* dari *slave* yang memiliki address 0x47 yaitu antarmuka I^2C. Raspberry pi membaca data via I^2C menggunakan kode *bus.read_i2c_block_data().* Terdapat beberapa kode lain yang berhubungan dengan I^2C pada python tergantung dari jenis dan jumlah data. *Array* adalah jenis data yang diterima oleh raspberry dan berisi 2 data pengukuran CO_2 yang telah dipecah. Berikut adalah rumus atau kode yang digunakan untuk memperoleh data pengukuran CO_2 secara utuh pada gambar 3.47 di bawah ini.

```
def readCo2():
        co2 = bus.read_i2c_block_data(address,0,2)
        y = co2[0]
        z = co2[1]
        co2 = (256*y)+z
        return co2
```

Gambar 3.47 Kode Kombinasi 2 *byte* data