TUGAS AKHIR – TF145565

RANCANG BANGUN SISTEM MONITORING KONSENTRASI GAS NITROGEN OKSIDA (NOx) SEBAGAI EMISI GAS BUANG MENGGUNAKAN SENSOR GAS MQ – 135 BERBASIS MIKROKONTROLLER STM32F4 *DISCOVERY*

Haryo Arif Wicaksono
NRP 2414.031.056

Dosen Pembimbing
Dr. Ir.Ali Musyafa', M.Sc
NIP. 19600901 198701 1 001

PROGRAM STUDI D3 TEKNIK INSTRUMENTASI
DEPARTEMEN TEKNIK INSTRUMENTASI
FAKULTAS VOKASI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2017

TUGAS AKHIR – TF145565

RANCANG BANGUN SISTEM MONITORING KONSENTRASI GAS NITROGEN OKSIDA (NOx) SEBAGAI EMISI GAS BUANG MENGGUNAKAN SENSOR GAS MQ – 135 BERBASIS MIKROKONTROLLER STM32F4 *DISCOVERY*

Haryo Arif Wicaksono
NRP 2414.031.056

Dosen Pembimbing
Dr. Ir.Ali Musyafa', M.Sc
NIP. 19600901 198701 1 001

PROGRAM STUDI D3 TEKNIK INSTRUMENTASI
DEPARTEMEN TEKNIK INSTRUMENTASI
FAKULTAS VOKASI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA

2017

FINAL PROJECT – **TF145565**

DESIGN OF EMISSION GAS NOx (NITROGEN OXIDE) MONITORING SYSTEM USING MQ – 135 WITH MICROCONTROLLER STM32F4 DISCOVERY

Haryo Arif Wicaksono
NRP 2414.031.056

Advisor Lecturer
Dr. Ir.Ali Musyafa', M.Sc
NIP. 19600901 198701 1 001

DIPLOMA OF INSTRUMENTATION ENGINEERING
DEPARTEMENT OF ENGINEERING INSTRUMENTATION
VACULTY OF DIPLOMA
SEPULUH NOPEMBER INSTITUTE OF TECHNOLOGY
SURABAYA
2017

RANCANG BANGUN SISTEM MONITORING KONSENTRASI GAS NITROGEN OKSIDA (NOx) SEBAGAI EMISI GAS BUANG MENGGUNAKAN SENSOR GAS MQ – 135 BERBASIS MIKROKONTROLLER STM32F4 *DISCOVERY*

TUGAS AKHIR

Oleh :
HARYO ARIF WICAKSONO
NRP. 2414 031 056

Surabaya, 28 Juli 2017
Mengetahui / Menyetujui

Dosen Pembimbing

Dr. Ir. Ali Musyafa', M.Sc
NIP. 19600901 198701 1 001

Ketua Departemen Teknik Instrumentasi FV - ITS

Dr. Ir. Purwadi Agus Darwito, M.Sc.
NIP. 19620822 198803 1 001

RANCANG BANGUN SISTEM MONITORING KONSENTRASI GAS NITROGEN OKSIDA (NOx) SEBAGAI EMISI GAS BUANG MENGGUNAKAN SENSOR GAS MQ-135 BERBASIS MIKROKONTROLER STM32F4 *DISCOVERY*

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Ahli Madya
Pada
Program Studi DIII Teknik Instrumentasi
Jurusan Teknik Instrumentasi
Fakultas Vokasi
Institut Teknologi Sepuluh Nopember

Oleh :
Haryo Arif Wicaksono
NRP. 2414 031 056

Disetujui oleh Tim Penguji Tugas Akhir :

1. Dr. Ir. Ali Musyafa' ..... (Dosen Pembimbing)
2. Dr. Ir. Purwadi Agus D., M.Sc. ..... (Ketua Tim Penguji)
3. Detak Yan Pratama S.T. M.Sc ..... (Dosen Penguji 1)

SURABAYA
25 JULI 2017

RANCANG BANGUN SISTEM MONITORING KONSENTRASI GAS NITROGEN OKSIDA (NOx) SEBAGAI EMISI GAS BUANG MENGGUNAKAN SENSOR GAS MQ – 135 BERBASIS MIKROKONTROLLER STM32F4 *DISCOVERY*

Nama Mahasiswa	**: Haryo Arif Wicaksono**
NRP	**: 2414 031 056**
Jurusan	**: Departemen Teknik Instrumentasi FV Prodi D3 Teknik Instrumentasi**
Dosen Pembimbing	**: Dr. Ir. Ali Musyafa', MSc**

Abstrak

Pencemaran udara dewasa ini semakin memprihatinkan. Hal ini terlihat dimana terjadi perubahan cuaca dan iklim lingkungan yang mempengaruhi suhu bumi dan berbagai pengaruh lain. Gangguan kesehatan dan ancaman kepunahan beberapa spesies makhluk hidup di bumi menjadi fenomena yang tak terhindarkan. Hal ini merupakan permasalahan global yang harus diperhatikan, karena akan berakibat sangat buruk bagi aktivitas kehidupan. Untuk mengetahui besar PPM NOx yang tercemar di perkotaan maka penelitian ini dilakukan perancangan sistem monitoring konsentrasi gas nitrogen oksida (NOx) sebagai emisi gas buang menggunakan sensor gas MQ - 135 berbasis mikrokontroller STM32F4 *discovery*, dan akan ditampilkan pada LCD 16 x 4. Berdasarkan hasil pengujian, bahwa pada pengukuran alat diperoleh nilai 95% hasil pengukuran (*x)* didapatkan 0,14 ± 0,3 ppm.

Kata kunci: Nitrogen Dioksida, PPM, *monitoring,* STM32F4 *discovery*

DESIGN OF EMISSION GAS NOx(NITROGEN OXIDE) MONITORING SYSTEM USING MQ – 135 WITH MICROCONTROLLER STM32F4 DISCOVERY

Name	**: Haryo Arif Wicaksono**
NRP	**: 2414 031 056**
Department	**: *Diploma III Of Instrument Engineering, Department Of Instrument Engineering FV-ITS***
Advisor Lecturer	**: Dr. Ir. Ali Musyafa', MSc**

Abstract

Air pollution today is increasingly alarming. This is seen where there is a change of weather and climate environment that affect the temperature of the earth and various other influences. Health problems and threats of extinction of some species of living things on earth became an inevitable phenomenon. This is a global issue that must be considered, because it will be very bad for life activities. To find out the amount of PPM NOx contaminated in urban area, this research is done by designing monitoring system of nitrogen oxide gas concentration (NOx) as exhaust emission using MQ-135 gas sensor based on microcontroller STM32F4 discovery, and will be displayed on LCD 16 x 4. Based on test result , That on the measurement tool obtained value 95% measurement results (x) obtained 0.14 ± 0.3 ppm.

Keyword : Nitrogen Dioxide, PPM, monitoring, STM32F4 Discovery

KATA PENGANTAR

Puji syukur penulis panjatkan kehadirat Allah SWT atas limpahan rahmat dan hidayah-Nya serta shalawat dan salam kepada Nabi Muhammad SAW sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul **"RANCANG BANGUN SISTEM MONITORING KONSENTRASI GAS NITROGEN OKSIDA (NOx) SEBAGAI EMISI GAS BUANG MENGGUNAKAN SENSOR GAS MQ – 135 BERBASIS MIKROKONTROLLER STM32F4 *DISCOVERY*".** Penulis telah banyak mendapatkan bantuan dari berbagai pihak dalam menyelesaikan Tugas Akhir ini. Untuk itu penulis mengucapkan terima kasih kepada :

1. Dr. Ir. Purwadi Agus Darwinto, MSc Selaku Kepala Departemen Teknik Instrumentasi.
2. Bapak Dr. Ir Ali Musyafa', MSc selaku dosen pembimbing yang senantiasa memberikan motivasi, bimbingan dan arahan dalam menyelesaikan Tugas Akhir ini.
3. Bapak Dr. Ridho Hantoro ST,MT selaku dosen wali penulis yang selalu memberi arahan dan motivasi yang terbaik.
4. Bapak dan Ibu dosen Teknik Fisika dan Teknik Instrumentasi yang telah memberikan bimbingan, ilmu dan arahan selama masa perkuliahan di Teknik Fisika ITS.
5. Orang tua khususnya ibu saya yang tanpa lelah memberikan motivasi dan berkat doanya penulis dapat menyelesaikan tugas akhir ini.
6. Seluruh Staf Departemen Teknik Fisika dan Teknik Instrumentasi yang telah membantu penulis dalam hal administrasi.
7. Teman-teman seperjuangan dalam mengerjakan Tugas Akhir
8. Syahril Arisdianta, Atik Sinawang, Lailatul Mufida selaku *partner* dalam mengerjakan tugas akhir, yang tanpa patah semangat dalam menyelesaikan tugas akhir ini.
9. Keluarga Laboratorium Pengukuran Fisis yang banyak membantu menyemangati penulis dalam myelesaikan tugas akhir ini.

10. Ridha Tentiani yang menyemagati tanpa henti penulis untuk menyelesaikan tugas akhirnya.
11. Serta semua orang yang tidak dapat penulis sebutkan satu persatu.

Penulis menyadari bahwa penulisan laporan Tugas Akhir ini tidaklah sempurna. Oleh karena itu sangat diharapkan kritik dan saran yang membangun dari semua pihak sehingga mencapai sesuatu yang lebih baik lagi. Penulis juga berharap semoga laporan ini dapat menambah wawasan yang bermanfaat bagi pembacanya.

Surabaya, 20 Juli 2017

Penulis.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pencemaran udara dewasa ini semakin memprihatinkan. Hal ini terlihat dimana terjadi perubahan cuaca dan iklim lingkungan yang mempengaruhi suhu bumi dan berbagai pengaruh lain. Keadaan tersebut membawa dampak bagi mencairnya es di kutub yang meningkatkan volume air laut sehingga mengancam tenggelamnya pulau-pulau kecil maupun daratan rendah pulau-pulau besar. Gangguan kesehatan dan ancaman kepunahan beberapa spesies makhluk hidup di bumi menjadi fenomena yang tak terhindarkan. Hal ini merupakan permasalahan global yang harus diperhatikan, karena akan berakibat sangat buruk bagi aktivitas kehidupan.**[1]**

Pencemaran udara adalah masukannya makhluk hidup, zat, energi, dan komponen lain ke udara atau berubahnya tatanan udara oleh kegiatan manusia, oleh proses alam, sehingga terbentuk senyawa-senyawa yang menyebabkan kualitas udara turun sampai ke tingkat tertentu dan berakibat pada kurang atau tidak berfungsinya lagi pada peruntukannya (Peraturan pemerintah no. 29 tahun 1986).**[2]**

Sumber pencemaran udara dapat berasal dari berbagai kegiatan manusia antara lain dalam transportasi, dan industri. Selain itu pencemaran udara juga dapat disebabkan oleh berbagai kegiatan alam, seperti gunung meletus, kebakaran hutan, dan gas alam beracun.

Udara merupakan unsur paling penting bagi kelangsungan kehidupan manusia, binatang dan tumbuh-tumbuhan di muka bumi. Udara mengandung 78% nitrogen, 21% oksigen, serta 1% uap air, karbon dioksida, dan gas-gas lain. Di udara bebas terdapat sekitar enam kelompok senyawa penyebab polutan udara yang berbahaya khususnya bagi kehidupan manusia, yaitu oksida nitrogen (NO_X), oksida belerang (SO_X), karbon monoksida (CO), hidrokarbon (HC), partikulat, dan oksida fotokimia. Oksida nitrogen (NOx) tidak hanya sebagai polutan udara, namun juga

menyebabkan terjadinya hujan asam dan menipisnya ozon pada lapisan toposfer, yang berfungsi melindungi bumi dari radisi matahari. Nilai batas kadar NOx pada udara bersih adalah 0,53 ppm (*Clean Air Technology Center*, 1999). Batas maksimum dalam Batas Mutu Emisi (BME) 1995 pada kendaraan bermotor untuk lepasan total partikel, SO_2 dan NO_2 berturut-turut adalah 300, 1500 dan 1700 mg/m^3, maka pada BME tahun 2000, nilainya turun menjadi 150, 750 dan 850 mg/m^3, setengah dari batas semula (Menteri Negara Lingkungan Hidup No. : KEP.13/MENLH/3/1995 tahap II tahun 2000).**[3]**

Oksida nitrogen lebih dikenal sebagai nitrogen monoksida (NO) dan nitrogen dioksida (NO_2). Nitrogen monoksida merupakan gas yang tidak berwarna dan tidak berbau, sedangkan nitrogen dioksida merupakan gas yang berwarna coklat kemerahan dan berbau. Hampir sebagian besar oksida nitrogen dihasilkan dari aktivitas manusia dalam menggunakan bahan bakar fosil. NO dapat membatasi jumlah kadar O_2 dalam darah karena bila bereaksi dengan O_2 akan membentuk NO_2 dan bila bertemu dengan uap air dalam tubuh maka akan membentuk HNO3.

Pemajanan NO_2 dengan kadar 5 ppm selama 10 menit terhadap manusia dapat menyebabkan kesulitan bernafas (V. M. Aroutiounian., 2007). Peningkatan konsentrasi dan komposisi oksida nitrogen di atmosfer menjadi permasalahan yang mengkhawatirkan, sehingga perlu dirancang instrumen yang dapat mendeteksi gas NOx di udara bebas, agar dapat menjadi tolak ukur kebijakan pemerintah dalam penanganan udara lingkungan. Penentuan kadar NOx di udara bebas telah dilakukan menggunakan peralatan spektroskopi analitik yang bekerja berdasarkan serapan inframerah, spektroskopi resonansi ion, kromatografi gas spektroskopi massa. Namun metode tersebut dibutuhkan peralatan dengan ukuran yang besar dan harganya relatif mahal, serta tidak dapat memonitoring langsung gas NOx di udara (Ono, et. al., 2001). Oleh karena itu pengembangan perangkat sensor yang mempunyai respon cepat, berukuran kecil,

tahan lama, murah, ramah lingkungan , dan dapat merespon cepat gas NOx dalam konsentrasi rendah sekalipun. Hal ini sangat penting untuk mencegah perubahan atmosfir global.**[4]**

Oleh karena itu dibuat alat yang berfungsi memonitoring kadar gas nitrogen oksida (NOx) di udara. Alat ini juga akan bersifat portabel sehingga mempermudah dalam penggunaanya. Ketika proses pengukuran kadar nitrogen oksida (NOx) berlangsung, terjadi perubahan kadar nitrogen oksida (NOx) yang dapat dideteksi oleh sensor gas nitrogen oksida (NOx). Perubahan akan masuk ke dalam rangkaian. Kemudian diproses menggunakan mikrokontroler untuk dihitung konsentrasi gas nitrogen oksida (NOx) yang diukur. Penggunaan mikrokontroler didasarkan pada kemudahan dalam pemrosesan data karena bahasa *C*-nya relatif mudah dan mikrokontroler memiliki fungsi yang dapat diterapkan dalam alat monitoring kadar gas nitrogen oksida (NOx). Alat ini akan menampilkan kadar gas nitrogen oksida (NOx) di udara pada display LCD. Oleh sebab itu dilakukan pembuatan dan perancangan alat monitoring emisi gas nitrogen oksida (NOx) sebagai emisi gas buang berbasis mikrokontroler STM32F4 Discovery yang mudah digunakan sehingga pengukuran terhadap emisi gas dapat dilakukan dengan mudah, dan penerapan *standart* emisi gas dapat dilakukan dengan baik.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang dijelaskan diatas, maka rumusan masalah dalam tugas akhir ini adalah sebagai berikut:

a. Bagaimana merancang suatu sistem *monitoring* kandungan gas nitrogen oksida (NOx) sebagai emisi gas buang berbasis mikrokontroler STM32F4 Discovery?
b. Bagaimana penggunaan data hasil *monitoring* kandungan gas nitrogen oksida (NOx) sebagai emisi gas buang?
c. Bagaimana penggunaan data *logger* untuk melakukan penyimpanan data *monitoring* kandungan gas nitrogen oksida (NOx) sebagai emisi gas buang?

1.3 Tujuan

Tujuan yang dicapai dalam tugas akhir ini adalah sebagai berikut:

a. Merancang suatu sistem monitoring kandungan gas nitrogen oksida (NOx) sebagai emisi gas buang berbasis mikrokontroler STM32F4 Discovery.
b. Mengetahui penggunaan data hasil monitoring kandungan gas nitrogen oksida (NOx) sebagai emisi gas buang.
c. Mengetahui penggunaan data logger pada sistem monitoing kandungan gas nitrogen oksida (NOx) sebagai emisi gas buang.

1.4 Batasan Masalah

Perlu diberikan beberapa batasan masalah agar pembahasan tidak meluas dn menyimpang dari tujuan. Adapun batasan masalah dari sistem yang dirancang ini adalah sebagai berikut:

a. Merancang sistem monitoring konsentrasi gas nitrogen oksida (NOx) menggunakan sebuah mikrokontroler STM32F4 *Discovery*.
b. Merancang sistem monitoring konsentrasi gas nitrogen oksida (NOx) menggunakan sensor MQ-135.
c. Merancang sistem monitoring konsentrasi gas nitrogen oksida (NOx) menggunakan sebuah *display* LCD ukuran 16x4 karakter.
d. Pengujian sistem dari rancang bangun yang telah dibuat dengan menguji performasi alat, baik keakuratan dan keoptimalan alat.
e. Menyusun hasil teori dari pembuatan *hardware*, analisa data dan kesimpulan dari data dan sistem yang ada.

1.5 Manfaat

Manfaat dari tugas akhir ini adalah sebagai sistem monitoring konsentrasi gas nitrogen oksida (NOx) menggunakan mikrokontroler STM32F4 Discovery dan dapat dijadikan sebagai perancangan alat monitoring ISPU (Indeks Standar Pencemaran Udara) masa depan.

1.6 Sistematika Laporan

Sistematika laporan yang digunakan dalam penyusunan laporan tugas akhir ini adalah sebagai berikut:

BAB I PENDAHULUAN

Pada bab ini membahas mengenai latar belakang, rumusan masalah, tujuan, batasan masalah, manfaat, dan sistematika penulisan dalam tugas akhir ini.

BAB II TINJAUAN PUSTAKA

Pada bab ini membahas mengenai teori-teori penunjang yang diperlukan dalam merealisasikan tugas akhir yaitu berupa teori tentang monitoring konsentrasi gas nitrogen oksida (NOx), dan perangkat-perangkat yang digunakan dalam pembuatan tugas akhir ini.

BAB III PERANCANGAN DAN PEMBUATAN ALAT

Pada bab ini diuraikan tentang penjelasan mengenai perancangan dan pembuatan alat.

BAB IV PENGUJIAN DAN ANALISIS DATA

Pada bab ini memuat tentang hasil pengujian dari perangkat yang dibuat beserta pembahasannya.

BAB V PENUTUP

Pada bab ini memuat tentang kesimpulan dan saran dari pembuatan tugas akhir ini.

“Halaman ini sengaja dikosongkan”

BAB II
TINJAUAN PUSTAKA

2.1 Nitrogen Oksida (NOx)

Nitrogen oksida (NOx) adalah senyawa gas yang terdapat di udara bebas (atmosfer) yang sebagian besar terdiri atas nitrit oksida (NO) dan nitrogen dioksida (NO2) serta berbagai jenis oksida dalam jumlah yang lebih sedikit. Kedua macam gas tersebut mempunyai sifat yang sangat berbeda dan keduanya sangat berbahaya bagi kesehatan. Gas NO yang mencemari udara secara visual sulit diamati karena gas tersebut tidak bewarna dan tidak berbau. Sedangkan gas NO2 bila mencemari udara mudah diamati dari baunya yang sangat menyengat dan warnanya merah kecoklatan. Sifat Racun (toksisitas) gas NO2 empat kali lebih kuat dari pada toksisitas gas NO. Organ tubuh yang paling peka terhadap pencemaran gas NO2 adalah paru-paru. Paru-paru yang terkontaminasi oleh gas NO2 akan membengkak sehingga penderita sulit bernafas yang dapat mengakibatkan kematiannya (Fardiaz, 1992).**[5]**

Kadar NOx di udara daerah perkotaan yang berpenduduk padat akan lebih tinggi dibandingkan di pedesaan karena berbagai macam kegiatan manusia akan menunjang pembentukan NOx, misalnya transportasi, generator pembangkit listrik, pembuangan sampah, dan lain-lain. Namun, pencemar utama NOx berasal dari gas buangan hasil pembakaran bahan bakar gas alam (Wardhana, 2004).**[6]**

Udara yang mengandung gas NO dalam batas normal relatif aman dan tidak berbahaya, kecuali bila gas NO yang tinggi dapat menyebabkan gangguan pada sisitem saraf yang menyebabkan kejang-kejang. Bila keracunan ini terus berlanjut akan dapat menyebabkan kelumpuhan. Gas NO akan menjadi lebih berbahaya apabila gas itu teroksidasi oleh oksigen sehingga menjadi gas NO_2. Di udara nitrogen monoksida (NO) teroksidasi sangat cepat membentuk nitrogen dioksida (NO_2) yang pada akhirnya nitrogen dioksida (NO2) teroksidasi secara fotokimia menjadi nitrat (Sastrawijaya, Tresna. 1991).**[7]**

Tabel 2.1 Pengaruh Gas NOx terhadap Kesehatan**[8]**

Efek	**Konsentrasi NOx**		**Waktu terjadi efek**
	mg/m3	ppm	
Batas timbul bau	0.23	0.12	segera
Batas pada adaptasi gelap	0.14	0.075	tidak dilaporkan
Peningkatanresisten pada udara bebas	0.5	0.26	tidak dilaporkan
	1.3-3.8	0.7-2.0	20 menit
	3.0-3.8	1.6-2.0	15 menit
	2.8	1.5	45 menit
	3.8	2	45 menit
	5.6	3	45 menit
	7.5-9.4	4.0-5.0	40 menit
	9.4	5	15 menit
	11.3-75.2	6.0-40.0	5 menit
Penurunan kapasitas difusi paru-paru	7.5-9.4	4.0-5.0	15 menit

2.2 *Exhaust Fan* DC

Exhaust fan berfungsi untuk menghisap udara di dalam ruang untuk dibuang ke luar, dan pada saat bersamaan menarik udara segar di luar ke dalam ruangan. Selain itu exhaust fan juga bisa mengatur volume udara yang akan disirkulasikan pada ruang. Supaya tetap sehat ruang butuh sirkulasi udara agar selalu ada pergantian udara dalam ruangan dengan udara segar dari luar luar ruangan.

Exhaust fan merupakan salah satu jenis kipas angin yg difungsikan untuk sirkulasi udara dalam ruang. Oleh karena itu,peletakkannya diantara indoor dan outdoor.

Dalam memilih *exhaust fan*, hal pertama yang perlu diperhatikan adalah luas ruangan. Kemudian ketahui juga fungsi ruangan. Misalnya, ruangan kerja saya sebenarnya juga berfungsi sebagai ruangan merokok. Setelah itu baru memilih *exhaust fan* dengan spesifikasi yang sesuai luas dan fungsi ruangan. Prinsip kerja dari arus searah adalah membalik phasa tegangan dari gelombang yang mempunyai nilai positif dengan menggunakan komutator, dengan demikian arus yang berbalik arah dengan kumparan jangkar yang berputar dalam medan magnet. Bentuk motor paling sederhana memiliki kumparan satu lilitan yang bisa berputar bebas di antara kutub-kutub magnet permanen. Pada tugas akhir ini, motor DC yang digunakan termasuk pada jenis *fan*, yaitu DC *Exhaust Fan* yang digunakan sebagai penyedot dan pembuang udara pada saat pengujian **[9]**.

Gambar 2.1 Fan DC 12 V

2.3 STM32F4 Discovery

Arsitektur ARM merupakan arsitektur processor 32-bit RISC yang dikembangkan oleh ARM Limited.Dikenal sebagai Advanced RISC Machine dimana sebelumnya dikenal sebagai Acorn RISC Machine. Pada awalnya merupakan prosesor desktop yang sekarang didominasi oleh keluarga x86. Namun desain yang sederhana membuat prosesor ARM cocok untuk aplikasi berdaya rendah.

Hal ini membuat prosesor ARM mendominasi pasar mobile electronic dan karena penggunaan AT MEGA dari ATMEL sudah mulai ditinggalkan dengan sudah terlalu banyak aplikasi dengan AT MEGA maka harus berkembang dengan ARM yang harganya lebih murah dengan teknologi yang lebih canggih. STMicroelectronics adalah salah satu vendor ARM yang memiliki market share terbesar. Harga STM32 Discovery Board yang cukup ekonomis serta memiliki kelengkapan yang excellent, lebih dari sekedar minimum sistem. Bahkan secara keseluruhan, lebih murah development board berbasis mikrokontroler 8-bit.STM32 Discovery Board dapat dijadikan media pembelajaran platform 32-bit ARM Cortex-M yang mumpuni. Di dalamnya sudah dilengkapi dengan ST- LINK/V2 untuk *programming* dan *debugging* melalui koneksi USB.STM32 Discovery Board juga dapat digunakan untuk membangun aplikasi dengan tingkat

kompleksitas algoritma yang cukup tinggi, karena dicatu prosesor kelas 32-bit berkinerja tinggi.**[10]**

Gambar 2.2 STM32F4 Discovery

Mikrokontroller STM32F4 memiliki antarmuka kamera yang dapat terhubung dengan kamera melalui 8 bit sampai 14 bit antarmuka parallel. Gambar yang diterima oleh mikrokontroller disimpan dalam RAM dengan menggunakan DMA (*Direct Memory Acces*). DMA merupakan suatu teknik perpindahan data dari suatu alamat memori ke alamat memori yang lain tanpa mengganggu kerja dari mikroprosessor. **[11]**

Fitur utama

1. Mikrokontroler STM32F407VGT6 menampilkan 32-bit ARM ® Cortex® -M4 dengan FPU inti, 1-Mbyte memori Flash, RAM 192-Kbyte dalam paket LQFP100
2. On-board ST-LINK / V2 pada STM32F4DISCOVERY (referensi tua) atau ST-LINK / V2-A pada STM32F407G-DISC1 (kode orde baru) USB ST-LINK dengan kemampuan re-pencacahan dan tiga antarmuka yang berbeda yakni debug port, virtual port Com (dengan kode orde baru saja), dan mass storage (dengan kode orde baru saja)
3. Dewan power supply: melalui bus USB atau dari tegangan suplai 5 Volt eksternal

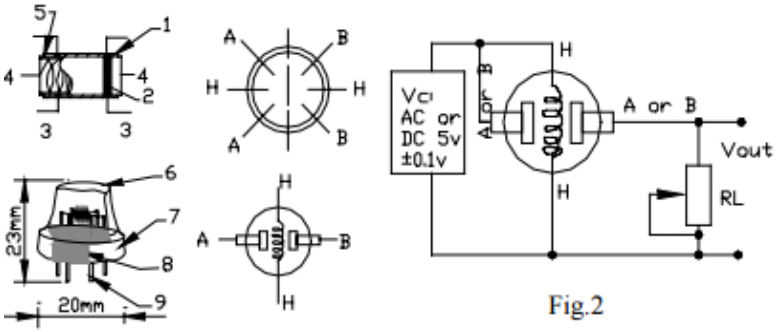
4. Eksternal power supply aplikasi: 3 V dan 5 V
5. LIS302DL atau LIS3DSH ST MEMS accelerometer 3-axis
6. MP45DT02 ST-MEMS sensor audio yang omni-directional microphone digital
7. DAC audio yang CS43L22 dengan kelas yang terintegrasi sopir D speaker
8. Delapan LED, yakni:
   •LD1 (merah / hijau) untuk komunikasi USB
   •LD2 (merah) untuk 3,3 V daya pada
   •Empat LED pengguna, LD3 (oranye), LD4 (hijau), LD5 (merah) dan LD6 (biru)
   •2 USB OTG LED LD7 (hijau) VBUS dan LD8 (merah) over-saat ini
9. Dua push-tombol (pengguna dan reset)
10. USB OTG FS dengan konektor micro-AB extension header untuk semua LQFP100 I / Os untuk koneksi cepat ke papan prototyping dan mudah menyelidik perangkat lunak bebas yang komprehensif termasuk berbagai contoh, bagian dari STM32CubeF4 paket atau STSW-STM32068 menggunakan standar perpustakaan warisan.

2.4 Sensor MQ 135

MQ-135 Air Quality Sensor adalah sensor yang memonitor kualitas udara untuk mendeteksi gas amonia (NH3), nitrogen oksida (NOx), alkohol / ethanol (C2H5OH), benzena (C6H6), karbondioksida (CO2), gas belerang / sulfur-hidroksida (H2S) dan asap / gas-gas lainnya di udara.

Sensor ini melaporkan hasil deteksi kualitas udara berupa perubahan nilai resistensi analog di pin keluarannya. Pin keluaran ini bisa disambungkan dengan pin ADC (analog-to-digital converter) di mikrokontroler / pin analog input Arduino Anda dengan menambahkan satu buah resistor saja (berfungsi sebagai pembagi tegangan / voltage divider). **[12]**.

Gambar 2.3 Bentuk Fisik MQ-135



Gambar 2.4 Struktur Sensor MQ - 135

Tabel 2.2 Spesifikasi Sensor MQ 135

Symbol	Parameter name	Technical	Remarks
Vc	Circuit voltage	5V±0.1	AC OR DC
V_H	Heating voltage	5V±0.1	ACOR DC
R_L	Load resistance	can adjust	
R_H	Heater resistance	33Ω±5%	Room Tem
P_H	Heating	less than 800mw	

2.5 LCD 16x4

Display elektronik adalah salah satu komponen elektronika yang berfungsi sebagai tampilan suatu data, baik karakter, huruf ataupun grafik. LCD (*Liquid Cristal Display*)

adalah salah satu jenis display elektronik yang dibuat dengan teknologi CMOS logic yang bekerja dengan tidak menghasilkan cahaya tetapi memantulkan cahaya yang ada di sekelilingnya terhadap front-lit atau mentransmisikan cahaya dari back-lit. LCD (*Liquid Cristal Display*) berfungsi sebagai penampil data baik dalam bentuk karakter, huruf, angka ataupun grafik. **[13]**.

Gambar 2.5 LCD 16x4

LCD (*Liquid Cristal Display*) terdapat microcontroller yang berfungsi sebagai pengendali tampilan karakter LCD (*Liquid Cristal Display*). Microntroller pada suatu LCD (*Liquid Cristal Display*) dilengkapi dengan memori dan register. Memori yang digunakan microcontroler internal LCD adalah :

- **DDRAM (*Display Data Random Access Memory*)** merupakan memori tempat karakter yang akan ditampilkan berada.
- **CGRAM (*Character Generator Random Access Memory*)** merupakan memori untuk menggambarkan pola sebuah karakter dimana bentuk dari karakter dapat diubah-ubah sesuai dengan keinginan.

- **CGROM (*Character Generator Read Only Memory*)** merupakan memori untuk menggambarkan pola sebuah karakter dimana pola tersebut merupakan karakter dasar yang sudah ditentukan secara permanen oleh pabrikan pembuat LCD (*Liquid Cristal Display*) tersebut sehingga pengguna tinggal mangambilnya sesuai alamat memorinya dan tidak dapat merubah karakter dasar yang ada dalam CGROM.

Register control yang terdapat dalam suatu LCD diantaranya adalah.

- **Register perintah** yaitu register yang berisi perintah-perintah dari mikrokontroler ke panel LCD (*Liquid Cristal Display*) pada saat proses penulisan data atau tempat status dari panel LCD (*Liquid Cristal Display*) dapat dibaca pada saat pembacaan data.
- **Register data** yaitu register untuk menuliskan atau membaca data dari atau keDDRAM. Penulisan data pada register akan menempatkan data tersebut keDDRAM sesuai dengan alamat yang telah diatur sebelumnya.

Pin, kaki atau jalur input dan kontrol dalam suatu LCD (*Liquid Cristal Display*) diantaranya adalah :

- **Pin data** adalah jalur untuk memberikan data karakter yang ingin ditampilkan menggunakan LCD (*Liquid Cristal Display*) dapat dihubungkan dengan bus data dari rangkaian lain seperti mikrokontroler dengan lebar data 8 bit.
- **Pin RS (*Register Select*)** berfungsi sebagai indikator atau yang menentukan jenis data yang masuk, apakah data atau perintah. Logika low menunjukan yang masuk adalah perintah, sedangkan logika high menunjukan data.
- **Pin R/W (*Read Write*)** berfungsi sebagai instruksi pada modul jika low tulis data, sedangkan high baca data.
- **Pin E (*Enable*)** digunakan untuk memegang data baik masuk atau keluar.

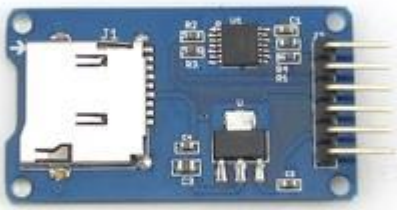
- **Pin VLCD** berfungsi mengatur kecerahan tampilan (kontras) dimana pin ini dihubungkan dengan trimpot 5 Kohm, jika tidak digunakan dihubungkan ke ground, sedangkan tegangan catu daya ke LCD sebesar 5 Volt.

2.6 *Micro SD Shield Module*

Micro Sd shield module adalah kartu memori yang pada umumnya berukuran 11 x 15mm, dengan berbagai ukuran kapasitas yang digunakan untuk keperluan penyimpanan data maupun pembacaan data yang sudah ada didalamnya. Data tersebut bersifat digital yang dapat berupa data gambar, dokumen, video, maupun audio. Peringkat kecepatan transfer rate yang di kenal dengan *Speed Class* yang merupakan standar kecepatan yang ada pada SD Card. Untuk saat ini terdapat beberapa *speed class* antara lain :

- *Class* 2 : dengan kecepatan 2 MB/s
- *Class* 4 : dengan kecepatan 4 MB/s
- *Class* 6 : dengan kecepatan 6 MB/s
- *Class* 10 : dengan kecepatan 10 MB/s
- UHS 1 : dengan kecepatan 10 MB/s
- UHS 3 : dengan kecepatan 30 MB/s

Modul micro sd merupakan modul untuk mengakses memori card yang bertipe micro SD untuk pembacaan maupun penulisan data dengan menggunakan sistem antarmuka SPI (*Serial Parallel Interface*). Modul ini cocok untuk berbagai aplikasi yang membutuhkan media penyimpan data, seperti sistem absensi, sistem antrian, maupun sistem aplikasi *data logging* lainnya **[14]**.

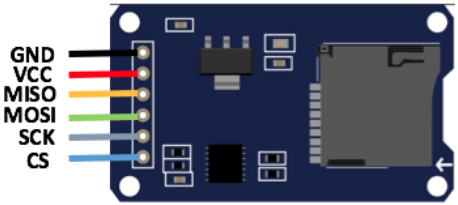


Gambar 2.6 *modul Micro Sd*

Fitur dan spesifikasi :

1. Mendukung pembacaan kartu memori SD Card biasa (<=2G) maupun *SDHC card (high-speed card*) (<=32G)
2. Tegangan operasional dapat menggunakan tegangan 5V atau 3.3V
3. Arus operasional yang digunakan yaitu 80mA (0.2~200mA)
4. Menggunakan antarmuka SPI
5. Pada modul ini sudah terdapat 4 lubang baut guna untuk pemasangan pada rangkaian lainnya
6. Ukuran modul yaitu 42 x 24 x 12 mm

Control Interface



Gambar 2.7 *modul Micro Sd*

- GND *: negatif power supply*
- VCC : *positif power supply*
- MISO, MOSI, SCK : ***SPI bus***
- CS : *chip select signal pin*

2.7 RTC DS3231

Modul RTC **(***Real Time Clock***)** ini memiliki akurasi dan presisi yang sangat tinggi dalam mencacah waktu dengan menggunakan IC RTC DS3231 *extremely accurate temperature compensated* RTC (TCXO). DS3231 memiliki kristal internal dan rangkaian kapasitor *tuning* di mana suhu dari kristal dimonitor secara berkesinambungan dan kapasitor disetel secara otomatis untuk menjaga kestabilan detak frekuensi.

Pencacahan waktu pada solusi RTC lain dapat bergeser (*drift*) hingga hitungan menit per bulannya, terutama pada kondisi perubahan suhu yang ekstrim. Modul ini paling jauh hanya bergeser kurang dari 1 menit per tahunnya, dengan demikian modul ini cocok untuk aplikasi kritis yang sensitif terhadap akurasi waktu yang tidak perlu disinkronisasikan secara teratur terhadap jam eksternal **[15]**.

Gambar 2.8 RTC **(***Real Time Clock***)**

2.8 *Modem GSM Wavecom Fastrack M1306B*

Wavecom M1306B adalah GSM/GPRS modem yang siap digunakan sebagai modem untuk suara, data, fax dan SMS. Kelas ini juga mendukung 10 tingkat kecepatan transfer data. *Wavecom M1306B TCP/IP* dengan mudah dikendalikan dengan menggunakan perintah AT untuk semua jenis operasi karena mendukung fasilitas koneksi RS232 dan juga fasilitas *TCP/IP stacked.* Dapat dengan cepat terhubung ke port serial komputer *desktop* atau *notebook.* casing logam Wavecom M1306B TCP/IP menjadi solusi yang tepat untuk aplikasi berat seperti telemetri atau *Wireless Local Loop* (PLN metering & Telepon Umum). Ukurannya sangat kecil memudahkan dalam peletakkan di berbagai macam area, indoor/outdoor. Cocok sekali digunakan pada aplikasi: Server Pulsa yang menghendaki kemampuan optimal dan usia pakai panjang, telemetri, SMS gateway/broadcast yang handal, PPOB PLN, ATM, Payment Point Systems, Metering Listrik. Modem GSM Serial Wavecom

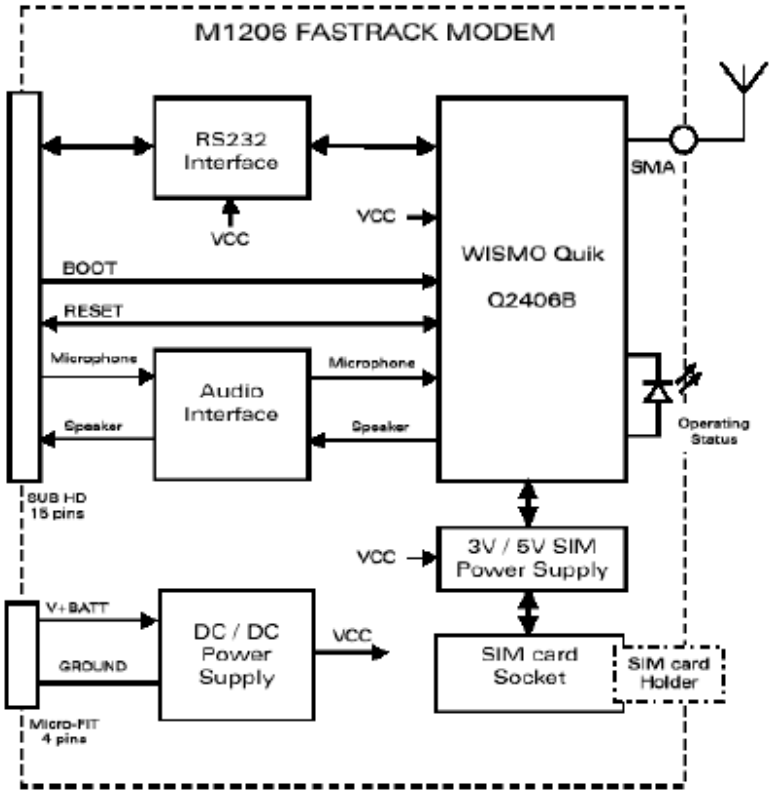
Fastrack M1206B memiliki dua type konektor yaitu serial dan USB**[16]**.

Gambar 2.9 *Modem Wavecom Fastrack M1306B Serial*

Spesifikasi modem WAVECOM FASTRACK M1206B:
· Dual-band GSM 900/1800MHZ & GPRS Class 10
· GSM Dual Band antenna
· Power Supply with 4 pin connector (untuk serial)
· Standard USB 2.0 interface (untuk USB)
· Input Voltage : 5V-32V
· Maximum transmitting speed 253KBps
· Support AT-Command

· Dimensi : 74×54×25mm

Berikut ini arsitektur dari Modem GSM Wavecom Fastrack M1206B:



Gambar 2.10 Arsitektur *Modem Wavecom Fastrack M1306B Serial*

2.9 *MCB (Miniature Circuit Breaker)*

MCB (Miniature Circuit Breaker) atau Miniatur Pemutus Sirkuit adalah sebuah perangkat elektromekanikal yang berfungsi sebagai pelindung rangkaian listrik dari arus yang berlebihan. Dengan kata lain, MCB dapat memutuskan arus listrik secara otomatis ketika arus listrik yang melewati MCB tesebut melebihi

nilai yang ditentukan. Namun saat arus dalam kondisi normal, MCB dapat berfungsi sebagai saklar yang bisa menghubungkan atau memutuskan arus listrik secara manual.

MCB pada dasarnya memiliki fungsi yang hampir sama dengan Sekering (FUSE) yaitu memutuskan aliran arus listrik rangkaian ketika terjadi gangguan kelebihan arus. Terjadinya kelebihan arus listrik ini dapat dikarenakan adanya hubung singkat (Short Circuit) ataupun adanya beban lebih (Overload). Namun MCB dapat di-ON-kan kembali ketika rangkaian listrik sudah normal, sedangkan Fuse/Sekering yang terputus akibat gangguan kelebihan arus tersebut tidak dapat digunakan lagi.**[17]**

Gambar 2.11 MCB Shukaku 220 V 2A

2.10 *ChibiOs*

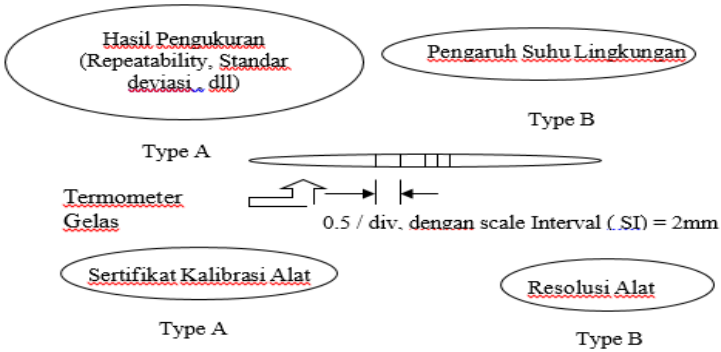
ChibiOs adalah real-time OS yang digunakan untuk embedded system dan mendukung banyak mikroprosesor termasuk stm32. Dengan ChibiOs, kita bisa membangun program-program untuk akuisisi data, control system ataupun pemrosesan sinyal. Selain chibiOs, modul SYM32 ini ‘mencuri’ dari perangkat lunak open source lainnya seperti stm32flash, compiler gcc-arm-mone-eabi dan masih banyak lagi.**[18]**

ChibiOS juga mengintegrasikan komponen Open Source eksternal untuk menawarkan solusi lengkap untuk perangkat embedded.

Komponen ChibiOS tersedia di bawah lisensi Open Source, GPL3 atau Apache 2.0, ada juga beberapa pilihan lisensi komersial. **[18]**

2.11 Teori Ketidakpastian

Sumber-sumber ketidakpastian yang turut memberikan kontribusi selain ada pada diri manusia sendiri sebagai pelaku pengukuran/kalibrasi juga pada alat-alat bantu (kalibrator) yang digunakan, juga resolusi alatnya serta pengaruh suhu lingkungan.



Gambar 2.12 Sumber – Sumber Ketidakpastian

Untuk mengevalusi masing- masing sumber ketidakpastian tersebut, diperlukan analisa dengan menggunakan metoda statistik, yang disebut analisa *type* A, dan menggunakan selain metode statistik yang disebut dengan Analisa *type* B. untuk lebih jelasnya dapat dilihat sebagai berikut:

a. Analisa *Type* A, (U_a)

   Pada tipe ini biasanya ditandai dengan adanya data pengukuran, misalnya n kali pengukuran, maka selanjutnya dari data tersebut, akan ditemukan nilai rata-ratanya, standar deviasinya, dan atau *repeatability*-nya.

Bentuk kurva dari tipe ini adalah sebaran Gauss. Rumus umum ketidakpatian untuk tipe A ini adalah:

$$Ua_1 = \frac{\sigma}{\sqrt{n}} \quad \textbf{(2.1)}$$

Dimana:
σ = Standar deviasi
n = Banyaknya data

Rumus standar deviasi (σ) sendiri adalah sebagai berikut:

$$\sigma = \frac{\sqrt{\sum(y_i - \bar{y})^2}}{n-1} \quad \textbf{(2.2)}$$

Dimana:
y_i = nilai koreksi ke-i
$\bar{y}$ = rata-rata nilai koreksi
σ = Standard Deviasi

Sedangkan untuk Ua_2 rumusnya dapat diketahui seperti di bawah ini:

$$Ua_2 = \sqrt{\frac{SSR}{n}} \quad \textbf{(2.3)}$$

Dimana:
SSR *(Sum Square Residual)* = ΣSR(*Square Residual*)
SR = R^2 (*Residu*)

$$SSR = \sum R^2 \quad \textbf{(2.4)}$$

$$R = Y_i - Y_{Reg} \quad \textbf{(2.5)}$$

$$Y_{Reg} = a + bX_i \quad \textbf{(2.6)}$$

$$a = \bar{y} + (b\bar{x}) \quad \textbf{(2.7)}$$

$$b = \frac{n.\sum x_i y_i - \sum xi \sum y_i}{n.\ \sum xi^2 - (\sum xi)^2} \quad \textbf{(2.8)}$$

b. Analisa *Type* B (U_b)
Pada analisa tipe ini akan digunakan selain metode statistik, yaitu berdasarkan adanya sertifikat kalibrasi atau tidak dan spesifikasi dari alat tersebut. Berhubung dalam laporan ini alat ukur standar yang dipakai tidak ada sertifikat kalibrasi, maka rumusnya adalah sebagai berikut:

$$Ub_1 = \frac{0{,}5\ Resolusi}{\sqrt{3}} \quad \textbf{(2.9)}$$

$$Ub_2 = \frac{a}{k} \quad \textbf{(2.10)}$$

Dimana:
Ub_1 = Ketidakpastian resolusi
Ub_1 = Ketidakpastian dari alat standar
a = Ketidakpastian sertifikat kalibrasi
k = faktor cakupan

c. Ketidakpastian Kombinasi (U_c)
Selanjutnya dari semua sumber ketidakpastian tersebut harus dikombinasikan atau digabungkan untuk memberikan gambaran menyeluruh ketidakpstian dari hasil kalibrasi tersebut. Rumus umum ketidakpastian kombinasi adalah:

$$U_c = \sqrt{\sum (U_a)^2 + \sum (U_b)^2} \quad \textbf{(2.11)}$$

Atau secara umum:

$$Uc^2 = \Sigma(C_i.U_i)^2 \quad \textbf{(2.12)}$$

Dengan C_i = Koefisien sensitifitas dari ketidakpastian ke-i

d. Ketidakpastian Diperluas (U_{exp})
Dalam pelaporan ketidakpastian hasil pengukuran/ kalibrasi yang dilaporkan adalah ketidakpatian yang sudah dalam perluasan (*expanded*), sehingga hasil tersebut sangat logis dalam kenyataan, selain itu dengan menggunakan tingkat kepercayaan 95 %, seperti lazimnya dipakai dalam pelaporan-pelaporan saat ini, lain halnya jika ada pengecualian dengan mengambil tingkat kepercayaan tertentu. Rumus ketidakpastian diperluas (*expanded uncertainty*) adalah:

$$U_{95} = k\, U_c \quad \text{..... (2.13)}$$

Dengan:
U_{95} = Ketidakpastian diperluas
k = Faktor Cakupan
U_c = Ketidakpastian kombinasi

e. Derajat Kebebasan Effektif (V_{eff})
Nilai faktor cakupan, k untuk perkalian ketidakpastian diperluas diatas didapat dari derajat kebebasan effektif, V_{eff}, dengan rumus:

$$V_{eff} = \frac{(U_c)^4}{\sum^{(U_i)^4}/_{V_i}} \quad \text{..... (2.14)}$$

Dengan:
U_c = Ketidakpastian kombinasi/gabungan
U_i = Ketidakpastian individual ke-i
V_i = Derajat kebebasan pada ketidakpastian individual ke-i

f. Tingkat Kepercayaan (U_{95})
Tingkat kepercayaan merupakan tingkatan keyakinan akan keberadaan nilai sebenarnya pada suatu tindak pengukuran dengan menggunkan alat tertentu.

g. Faktor Cakupan (k)
Faktor cakupan meruakan faktor pengali pada ketidakpastian, sehingga membentuk cakupan logis pada penggunaan keseharian. Faktor cakupan dicari menggunakan tabel *T-Student Distribution* **[19]**.

Tabel 2.3 *T-Student Distribution*[17]

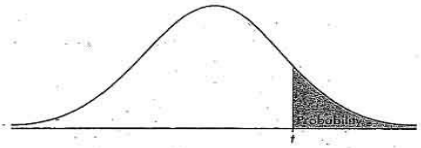


TABLE B: *t*-DISTRIBUTION CRITICAL VALUES

	Tail probability *p*											
df	.25	.20	.15	.10	.05	.025	.02	.01	.005	.0025	.001	.0005
1	1.000	1.376	1.963	3.078	6.314	12.71	15.89	31.82	63.66	127.3	318.3	636.6
2	.816	1.061	1.386	1.886	2.920	4.303	4.849	6.965	9.925	14.09	22.33	31.60
3	.765	.978	1.250	1.638	2.353	3.182	3.482	4.541	5.841	7.453	10.21	12.92
4	.741	.941	1.190	1.533	2.132	2.776	2.999	3.747	4.604	5.598	7.173	8.610
5	.727	.920	1.156	1.476	2.015	2.571	2.757	3.365	4.032	4.773	5.893	6.869
6	.718	.906	1.134	1.440	1.943	2.447	2.612	3.143	3.707	4.317	5.208	5.959
7	.711	.896	1.119	1.415	1.895	2.365	2.517	2.998	3.499	4.029	4.785	5.408
8	.706	.889	1.108	1.397	1.860	2.306	2.449	2.896	3.355	3.833	4.501	5.041
9	.703	.883	1.100	1.383	1.833	2.262	2.398	2.821	3.250	3.690	4.297	4.781
10	.700	.879	1.093	1.372	1.812	2.228	2.359	2.764	3.169	3.581	4.144	4.587
11	.697	.876	1.088	1.363	1.796	2.201	2.328	2.718	3.106	3.497	4.025	4.437
12	.695	.873	1.083	1.356	1.782	2.179	2.303	2.681	3.055	3.428	3.930	4.318
13	.694	.870	1.079	1.350	1.771	2.160	2.282	2.650	3.012	3.372	3.852	4.221
14	.692	.868	1.076	1.345	1.761	2.145	2.264	2.624	2.977	3.326	3.787	4.140
15	.691	.866	1.074	1.341	1.753	2.131	2.249	2.602	2.947	3.286	3.733	4.073
16	.690	.865	1.071	1.337	1.746	2.120	2.235	2.583	2.921	3.252	3.686	4.015
17	.689	.863	1.069	1.333	1.740	2.110	2.224	2.567	2.898	3.222	3.646	3.965
18	.688	.862	1.067	1.330	1.734	2.101	2.214	2.552	2.878	3.197	3.611	3.922
19	.688	.861	1.066	1.328	1.729	2.093	2.205	2.539	2.861	3.174	3.579	3.883
20	.687	.860	1.064	1.325	1.725	2.086	2.197	2.528	2.845	3.153	3.552	3.850
21	.686	.859	1.063	1.323	1.721	2.080	2.189	2.518	2.831	3.135	3.527	3.819
22	.686	.858	1.061	1.321	1.717	2.074	2.183	2.508	2.819	3.119	3.505	3.792
23	.685	.858	1.060	1.319	1.714	2.069	2.177	2.500	2.807	3.104	3.485	3.768
24	.685	.857	1.059	1.318	1.711	2.064	2.172	2.492	2.797	3.091	3.467	3.745
25	.684	.856	1.058	1.316	1.708	2.060	2.167	2.485	2.787	3.078	3.450	3.725
26	.684	.856	1.058	1.315	1.706	2.056	2.162	2.479	2.779	3.067	3.435	3.707
27	.684	.855	1.057	1.314	1.703	2.052	2.158	2.473	2.771	3.057	3.421	3.690
28	.683	.855	1.056	1.313	1.701	2.048	2.154	2.467	2.763	3.047	3.408	3.674
29	.683	.854	1.055	1.311	1.699	2.045	2.150	2.462	2.756	3.038	3.396	3.659
30	.683	.854	1.055	1.310	1.697	2.042	2.147	2.457	2.750	3.030	3.385	3.646
40	.681	.851	1.050	1.303	1.684	2.021	2.123	2.423	2.704	2.971	3.307	3.551
50	.679	.849	1.047	1.299	1.676	2.009	2.109	2.403	2.678	2.937	3.261	3.496
60	.679	.848	1.045	1.296	1.671	2.000	2.099	2.390	2.660	2.915	3.232	3.460
80	.678	.846	1.043	1.292	1.664	1.990	2.088	2.374	2.639	2.887	3.195	3.416
100	.677	.845	1.042	1.290	1.660	1.984	2.081	2.364	2.626	2.871	3.174	3.390
1000	.675	.842	1.037	1.282	1.646	1.962	2.056	2.330	2.581	2.813	3.098	3.300
∞	.674	.841	1.036	1.282	1.645	1.960	2.054	2.326	2.576	2.807	3.091	3.291
	50%	60%	70%	80%	90%	95%	96%	98%	99%	99.5%	99.8%	99.9%
	Confidence level *C*											

2.12 *Memory Card Micro SD*

MicroSD/SDCard adalah kartu memori non-volatile yang dikembangkan oleh SD Card Association yang digunakan dalam perangkat portable. Saat ini, teknologi *microSD* sudah digunakan oleh lebih dari 400 merek produk serta dianggap sebagai standar industri de-facto.

Keluarga *microSD* yang lain terbagi menjadi SDSC yang kapasitas maksimum resminya sekitar 2GB, meskipun beberapa ada yang sampai 4GB. SDHC (*High Capacity*) memiliki kapasitas dari 4GB sampai 32GB. Dan SDXC (Extended Capacity) kapasitasnya di atas 32GB hingga maksimum 2TB. Keberagaman kapasitas seringkali membuat kebingungan karena masing-masing protokol komunikasi sedikit berbeda.

Dari sudut pandang perangkat, semua kartu ini termasuk kedalam keluarga SD. SD adapter memungkinkan konversi fisik kartu SD yang lebih kecil untuk bekerja di slot fisik yang lebih besar dan pada dasarnya ini adalah alat pasif yang menghubungkan pin dari microSD yang kecil ke pin adaptor microSD yang lebih besar.

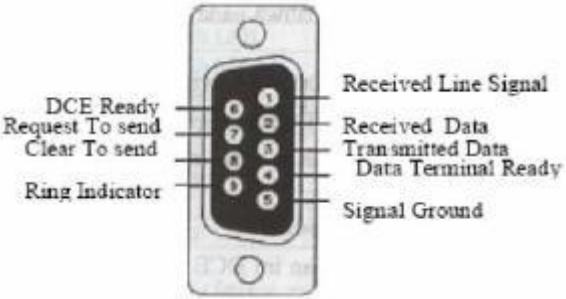
SD mempunyai bentuk fisik yang sama maka sering menyebabkan kebingungan di kalangan konsumen. Contohnya, *MicroSD*, *MicroSDHC*, dan *MicroSDXC* ukuran fisiknya sama tetapi kapabilitasnya berbeda. Protokol komunikasi untuk SDHC/SDXC/SDIO sedikit berbeda dengan *MicroSD* yang sudah mapan karena biasanyahost device keluaran lama tidak bisa mengenali kartu keluaran baru. kebanyakan masalah mengenai inkompatibilitas ini dapat diselesaikan dengan firmware update.

Gambar 2.13 *SD Card 8GB*

2.13 *Serial Port* RS 232

RS232 adalah standard komunikasi serial yang digunakan untuk koneksi peripheral ke peripheral. Biasa juga disebut dengan jalur I/O (input / output). Contoh yang paling sering kita temui adalah koneksi antara komputer dengan modem, atau komputer dengan mouse bahkan bisa juga antara komputer dengan komputer, semua biasanya dihubungkan lewat jalur port serial RS232. Standar ini menggunakan beberapa piranti dalam implementasinya. Paling umum yang dipakai adalah plug / konektor DB9 atau DB25. Untuk RS232 dengan konektor DB9, biasanya dipakai untuk mouse, modem, kasir register dan lain sebagainya, sedang yang konektor DB25, biasanya dipakai untuk joystik game.

Fungsi dari serial port RS232 adalah untuk menghubungkan / koneksi dari perangkat yang satu dengan perangkat yang lain, atau peralatan standart yang menyangkut komunikasi data antara komputer dengan alat-alat pelengkap komputer. Perangkat lainnya itu seperti modem, mouse, cash register dan lain sebagainya. Serial port RS232 pada konektor DB9 memiliki pin 9 buah.



Gambar 2.14 Konfigurasi Serial Port RS 232

Tabel 2.4 Konfigurasi *Pin Serial Port* RS 232

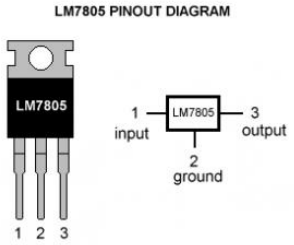
Pin DB25	Pin DB9	Singkatan	Keterangan
Pin 2	Pin 3	TD	*Transmit Data*
Pin 3	Pin 2	RD	*Receive Data*
Pin 4	Pin 7	RTS	*Request To Send*
Pin 5	Pin 8	CTS	*Clear To Send*
Pin 6	Pin 6	DSR	*Data Set Ready*
Pin 7	Pin 5	SG	*Signal Ground*
Pin 8	Pin 1	CD	*Carrier Detect*
Pin 20	Pin 4	DTR	*Data Terminal Ready*
Pin 22	Pin 9	RI	*Ring Indicator*

Singkatan	Keterangan	Fungsi
TD	*Transmit Data*	Untuk pengiriman data serial (TDX)
RD	*Receive Data*	Untuk penerimaan data serial (RDX)
RTS	*Request To Send*	Sinyal untuk menginformasikan modem bahwa UART siap melakukan pertukaran data
CTS	*Clear To Send*	Digunakan untuk memberitahukan bahwa modem siap untuk melakukan pertukaran data
DSR	*Data Set Ready*	Memberitahukan UART bahwa modem siap untuk melakukan pertukaran data
CD	*Carrier Detect*	Saat modem mendeteksi suatu 'carrier' dari modem lain maka sinyal ini akan diaktifkan
DTR	*Data Terminal Ready*	Kebalikan dari DSR untuk memberitahukan bahwa UAT siap melakukan hubungan komunikasi
RI	*Ring Indicator*	Akan aktif jika modem mendeteksi adanya sinyal dering dari saluran telepon

2.14 ***IC 7805***

Voltage regulator IC adalah IC yang digunakan untuk mengatur tegangan .IC 7805 adalah Regulator 5V Voltage yang membatasi output tegangan 5V dan menarik 5V diatur power supply . Muncul dengan ketentuan untuk menambahkan heatsink .Nilai maksimum untuk input ke regulator tegangan 35V . Hal ini dapat memberikan aliran tegangan stabil konstan 5V untuk input tegangan yang lebih tinggi sampai batas ambang 35V . Jika tegangan dekat 7.5V maka tidak menghasilkan panas dan karenanya tidak perlu untuk heatsink . Jika input tegangan lebih , maka kelebihan listrik dibebaskan sebagai panas dari 7805 .Ini mengatur output stabil 5V jika tegangan input adalah marah dari 7.2V ke 35V . Oleh karena itu untuk menghindari kehilangan daya mencoba mempertahankan input ke 7.2V . Dalam beberapa fluktuasi tegangan sirkuit fatal (untuk misalnya Microcontroller) , untuk situasi semacam itu untuk memastikan tegangan konstan IC 7805 Voltage Regulator digunakan . Untuk informasi lebih lanjut tentang spesifikasi dari 7805 Voltage Regulator silakan lihat lembar data di sini (IC 7805 Voltage R

egulator Data Sheet) .IC 7805IC 7805 adalah serangkaian 78XX regulator tegangan . Ini standar , dari nama dua digit terakhir menunjukkan 05 jumlah tegangan yang mengatur . Oleh karena itu 7805 akan mengatur 5V dan 7806 akan mengatur 6V dan seterusnya .Skema yang diberikan di bawah ini menunjukkan bagaimana menggunakan IC 7805 , ada 3 pin di IC 7805 , pin 1 mengambil tegangan input dan pin 3 menghasilkan tegangan output. The GND dari kedua input dan out yang diberikan ke pin 2 .



Gambar 2.15 *IC 7805*

2.15 Karakteristik Statik

Berikut ini merupakan perhitungan karakteristik statis dan kalibrasi yang digunakan untuk menganalisa data:

a. Range
b. Span
c. Resolusi
d. Sensitivitas $= \frac{\Delta O}{\Delta I} = \frac{4{,}096-0}{4{,}4-0} = 0{,}93$

e. Non- Linieritas

(N(I)) = O(I) – (KI + *a*)

*data yang dihitung adalah data pembacaan naik

Non – linieritas maksimum per unit $= \frac{N}{Omax-Omin} \boldsymbol{x\ 100}\%$

Dimana :

K (sensitivitas) =

a (*zero bias*) = $O_{min} - KI_{min}$

f. Histerisis

$H(I) = O(I)_{I\uparrow} - O(I)_{I\downarrow,}$ $\hat{H} = H(I)_{max}$ sehingga :

% maksimum histerisis =

$$= \frac{\hat{H}}{Omax-Omin} \boldsymbol{x\ 100}\%$$

g. Akurasi

$$\boldsymbol{A = 1 - \left| \frac{Yn - Xn}{Yn} \right| x\ 100}\%$$

Dengan :

Yn = Pembacaan Standar

Xn = Pembacaan Alat

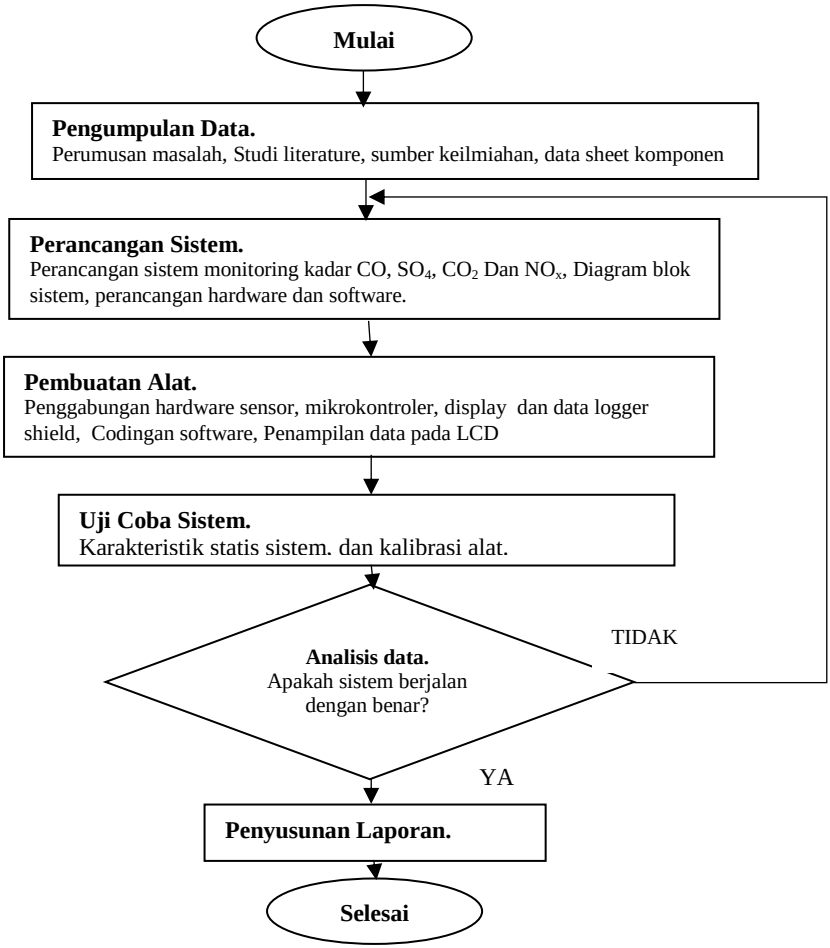
h. Kesalahan(Error)

e = 1 – A

BAB III
PERANCANGAN DAN PEMBUATAN ALAT

3.1 *Flowchart* dan Diagram Blok Perancangan Alat

Langkah-langkah perancangan alat ini digambarkan dalam *flowchart* yang dapat dilihat pada gambar 3.1 di bawah ini.

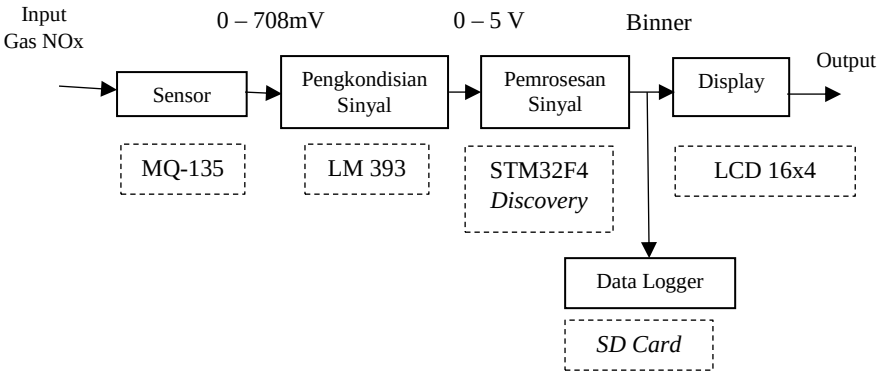


Gambar 3.1 *Flowchart* Pengerjaan Tugas Akhir

Flowchart diatas merupakan *flowchart* pengerjaan tugas akhir mulai dari *start* hingga selesai. Tahap awal pada *flowchart* tugas akhir ini dimulai dengan adanya studi literatur sebagai upaya pemahaman terhadap materi yang menunjang tugas akhir mengenai "Rancang Bangun Sistem Monitoring Konsentrasi Gas Nitrogen oksida (NOx) Sebagai Emisi Gas Buang Menggunakan Sensor MQ 135 Berbasis Mikrokontroler STM32F4 Discovery". Setelah melakukan studi literatur, selanjutnya adalah melakukan perancangan sistem dan mempersiapkan komponen yang dibutuhkan. Kemudian dibuat perancangan *hardware*, *software*, dan mekanik dari sistem monitoring gas nitrogen oksida (NOx) berbasis STM32F4 Discovery. Setelah itu dari sistem monitoring gas nitrogen dioksida (NOx) yang telah dibuat, dilakukan pengujian alat dengan memberi *input gas* pada box , sehingga dapat diketahui konsentrasi gas Nitrogen oksida (NOx). Apabila semua rancang bangun sistem monitoring gas Nitrogen oksida (NOx) dapat bekerja dengan baik, maka selanjutnya dilakukan pengambilan data pada *plant monitoring gas*. setelah pembuatan rancangan telah selesai dengan hasil yang sesuai dengan yang diinginkan, kemudian dilakukan analisis data dengan memanfaatkan hasil dari uji performansi dan sistem pengendalian. Setelah semua hasil yang diinginkan tercapai mulai dari studi literatur hingga analisa data dan kesimpulan dicantumkan dalam sebuah laporan.

Diagram merupakan pernyataan hubungan yang berurutan dari suatu atau lebih komponen yang memiliki kesatuan kerja tersendiri, dan setiap blok komponen mempengaruhi komponen lainnya. Diagram blok merupakan salah satu cara yang paling sederhana untuk menjelaskan cara kerja dari suatu sistem. Dengan diagram blok dapat menganalisa cara kerja rangkaian dan merancang *hardware* yang akan dibuat secara umum.

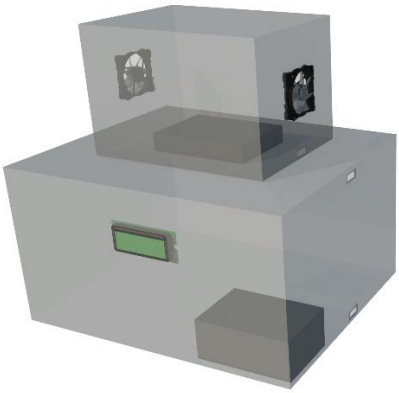
Adapun diagram blok dari sistem yang dirancang, seperti yang diperlihatkan pada gambar 3.2.



Gambar 3.2 Diagram Blok Sistem monitoring gas buang

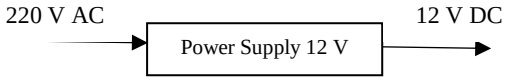
Gambar 3.2 di atas merupakan gambar diagram blok sistem monitoring emisi gas buang berupa konsentrasi gas karbon dioksida (NO_x) menggunakan sensor gas MG-811 yang dikontrol oleh mikrokontroler STM32F4 Discovery. Dalam perancangan mekanik terdiri dari bebrapa komponen mekanik seperti box, LCD dan sensor. Box disini berfungsi sebgaai tempat komponen dan rangkaian elektrik. Selain itu terdapat juga perancangan elektrik yang meliputi rangkaian power supply, rangkaian minimum sistem mikrokontroler, rangkaian driver sensor dan rangkaian LCD. Untuk menjalankan sistem juga terdapat perancangan software yang menggunakan C *Compiler*. Dalam rancang bangun sistem monitoring konsentrasi gas Nitrogen oksida (NOx) sebagai emisi gas buang pada kendaraan bermotor dimana terdiri dari beberapa bagian penting yaitu sensor, sinyal pengkondisian, sinyal pemrosesan dan representasi data atau *display* berupa LCD (*Liquid Crystal Display*) 16x4 yang ditampilkan dalam satuan ppm.

3.2 Gambaran Umum *Plant*



Gambar 3.3 Skematik *Plant* Monitoring Gas Buang

3.3 Rangkaian *Power Supply*

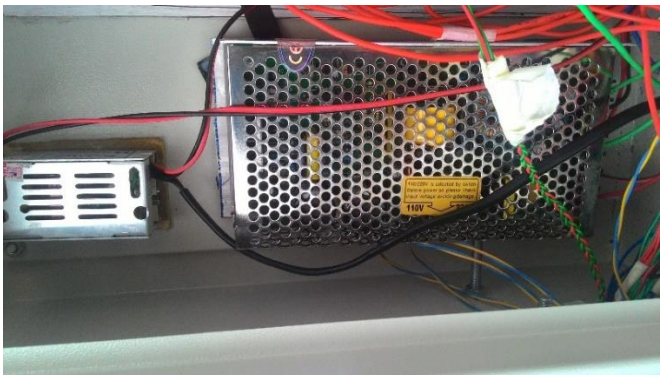


Gambar 3.4 Diagram Blok *Power Supply*

Power supply merupakan sumber tenaga yang dibutuhkan suatu rangkaian elektronika untuk bekerja. Besar *power supply* ini tergantung oleh spesifikasi dari alat masing-masing. Pada perancangan sistem pengendali ini *power supply* digunakan untuk men-*supply* rangkaian mikrokontroler STM32F4 discovery, rangkaian *sensor MG 811*, rangkaian *sensor MQ 7,* rangkaian *sensor MQ 136*, rangkaian *sensor MQ 135*, modul RTC DS3231, modul sd card shield, modul sms gateway, dan LCD 16x4.

Pada rangkaian *power supply* pada umumnya sering menggunakan IC regulator dalam mengontrol tegangan yang diinginkan. Regulator tegangan menjadi sangat penting gunanya apabila mengaplikasikan *power supply* tersebut untuk rangkaian-rangkaian yang membutuhkan tegangan yang sangat stabil.

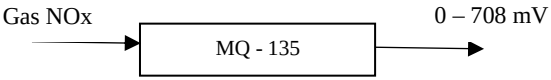
Misalkan untuk sistem *digital*, terutama untuk mikrokontroler yang sangat membutuhkan tegangan dan arus yang stabil.



Gambar 3.5 *Power Supply* DC

Power supply untuk tegangan DC digunakan sebagai *supply* untuk perangkat yang membutuhkan tegangan DC 12 V dan 5 V. IC regulator yang umum digunakan untuk mengontrol tegangan adalah IC keluarga 78XX. IC ini dapat mengontrol tegangan dengan baik. Keluaran tegangan yang diinginkan tinggal melihat tipe yang ada. Misalkan tipe 7805 dapat memberikan keluaran tegangan 5 V dengan toleransi ±1 dengan arus keluaran maksimal 1500 mA.

3.4 Perancangan *Sensor* MQ 135

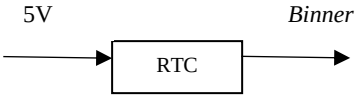


Gambar 3.6 Diagram Blok Sensor MQ - 135

Sensor MG-811 digunakan untuk mendeteksi senyawa gas *NOx*. Penggunaan sensor MQ-135 menggunakan prinsip reaksi elektrokimia untuk tegangan outputnya. Reaksi elektrokimia yang

terjadi menghasilkan *emf (electromotive force)* diantara dua elektroda. Perubahan nilai *emf* menunjukan perubahan kosentrasi *NOx* yang dideteksi sensor dan untuk gambar rangkaian pengkondisian sinyal sensor MQ135.

3.5 Perancangan *Real Time Clock (RTC)*



Gambar 3.7 Diagram blok *RTC*

RTC yang digunakan adalah IC RTC DS3231 sebagai input pemberi referensi waktu terhadap data yang akan diperoleh. Cara kerjanya adalah alamat dan data ditransmisikan secara serial melalui sebuah jalur data dua arah I2C. Karena menggunakan jalur data I2C maka hanya memerlukan dua buah pin saja untuk berkomunikasi yaitu pin data dan pin untuk sinyal clock (SDA dan SCL) sehingga STM32F4 discovery dapat mengolah data dan clock yang diterima dari RTC untuk dijadikan referensi waktu. Pin SDA dihubungkan ke port ?? dan pin SCL ke port ?? STM32F4 discovery.

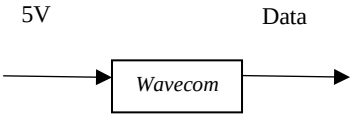
3.6 *Perancangan Data Logger*

Perancangan ini digunakan dalam sistem monitoring ini. Modul SD card digunakan sebagai data logger dalam penyimpanan informasi dari hasil ppm yang telah diukur. Penyimpanan data ini disimpan dalam format excel atau .cgsv sehingga dapat mudah dilihat.Terdapat 6 pin dalam modul sd card ini yakni Ground, VCC, MISO, MOSI, SCK , dan CS.

Dalam perancangannya, menghubungkan pin pada Modul SD Card kedalam mikrokontroller STM32F4 Discovery yang terhubung dengan shield yang telah dibuat. Pin Ground pada modul SD Card disambungkan ke Ground STM32F4 Discovery.

Pin VCC pada modul SD Card disambungkan ke pin 5 volt STM32F4 Discovery. Pin MISO pada modul SD Card disambungkan ke pin STM32F4 Discovery. Pin MOSI pada modul SD Card disambungkan ke pin STM32F4 Discovery . Pin SCK pada modul SD Card disambungkan ke pin STM32F4 Discovery dan Pin CS pada modul SD Card disambungkan ke pin STM32F4 Discovery.

3.7 Perancangan *Modem Wavecom*



Gambar 3.8 Diagram blok *Wavecom*

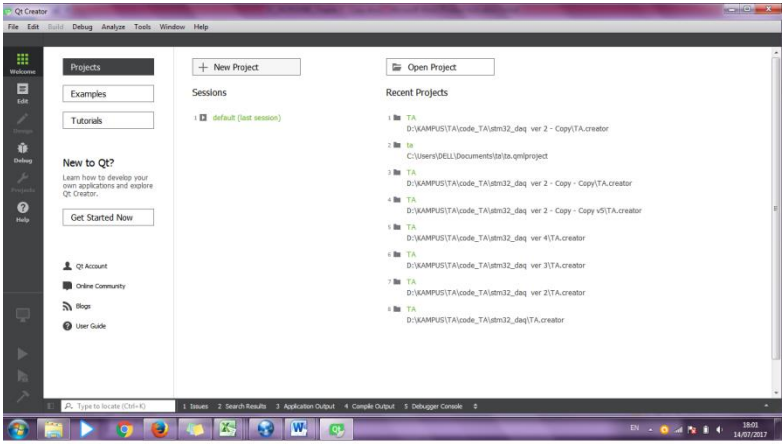
Pada bagian rancang bangun monitoring gas ini terdapat modem wavecom M1306B serial RS232. Dimana modem ini banyak digunakan oleh oleh pengguna layanan sms gateway untuk brodcast sms, kirim sms massal dan compatible dengan engine sms seperti gammu dan quick gateway. Modem ini juga dilengkapi dengan AT Command untuk semua jenis operasi karena mendukung fasilitas koneksi RS232 dan dan dapat dengan cepat terhubung ke port serial komputer maupun mikrokontroller sehingga sangat mudah untuk dikendalikan. Kelas ini juga mendukung 10 tingkat kecepatan transfer data.

Dalam perancangan penggunaan modem wavecom M1306B serial RS232 dilakukan dengan cara mengambil pin Rx, Tx, dan Ground dari serial RS232 modem wavecom M1306B dan menghubungkan pin Rx, Tx, dan Ground ke dalam mikrokontroler STM32F4 Discovery yang terhubung dengan shield yang telah dibuat. Pin Ground pada modul disambungkan ke Ground STM32F4 Discovery. Pin VCC pada modul SD Card disambungkan ke pin 5 volt STM32F4 Discovery. Pin MISO pada modul SD Card disambungkan ke pin STM32F4 Discovery.

Pin MOSI pada modul SD Card disambungkan ke pin STM32F4 Discovery . Pin SCK pada modul SD Card disambungkan ke pin STM32F4 Discovery dan Pin CS pada modul SD Card disambungkan ke pin STM32F4 Discovery.

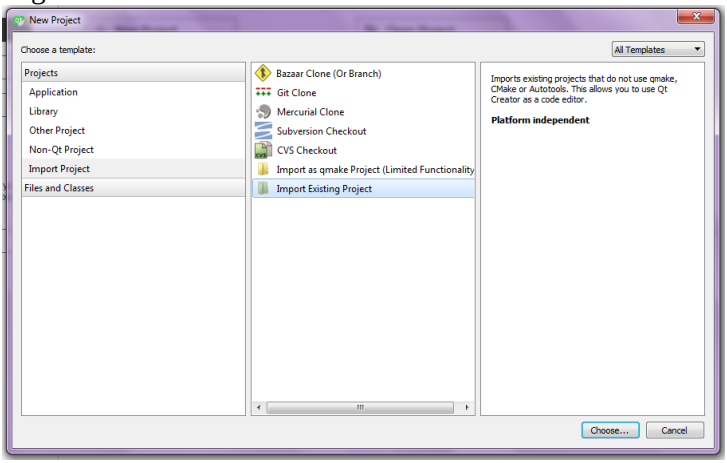
3.8 Langkah – Langkah Meng-*compile* Program

a. *Qt Creator* dibuka.
b. Klik *New project*. Kemudian akan muncul gambar seperti dibawah ini.



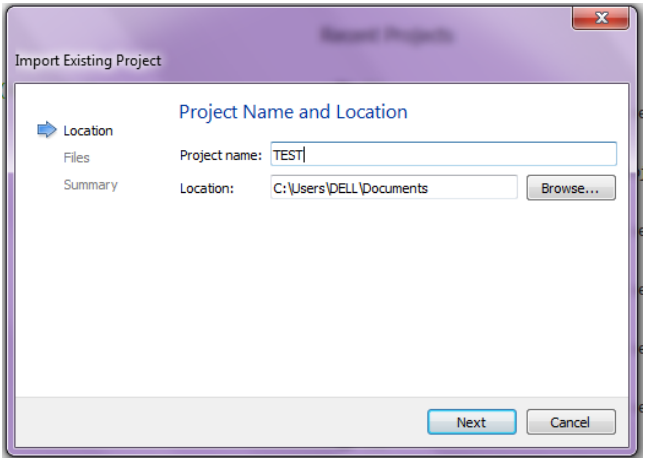
Gambar 3.9 *Create New Project*

c. Pilih *import project*. Lalu klik *import existing project* seperti gambar dibawah ini:



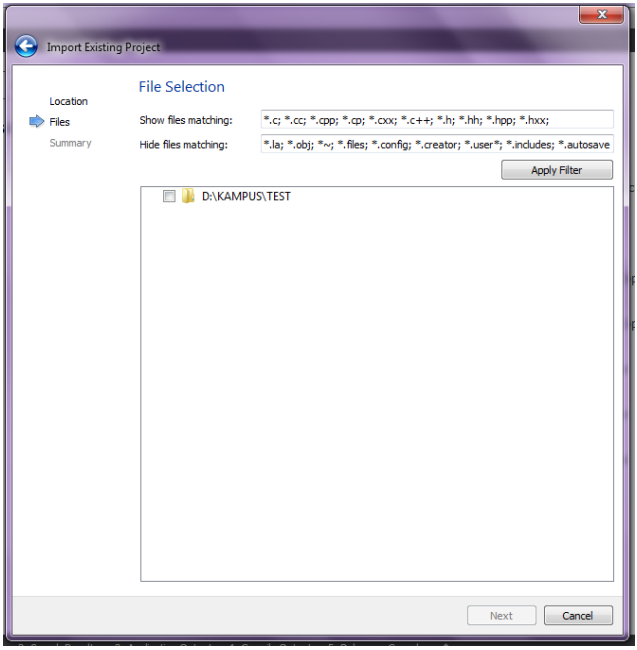
Gambar 3.10 *Project Qt Creator*

d. Kemudian akan muncul dialog *import existing project*. Beri nama *project* sesuai yang diinginkan dan pilih lokasi yang diinginkan dalam laptop. Kemudian klik *next*.



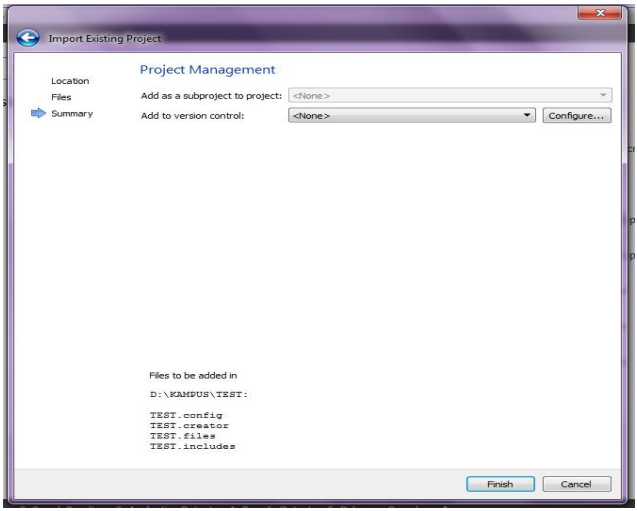
Gambar 3.11 *import existing project name and location*

e. Kemudian akan muncul kotak *file selection*. Setelah itu centang *location file* dan klik *next.*



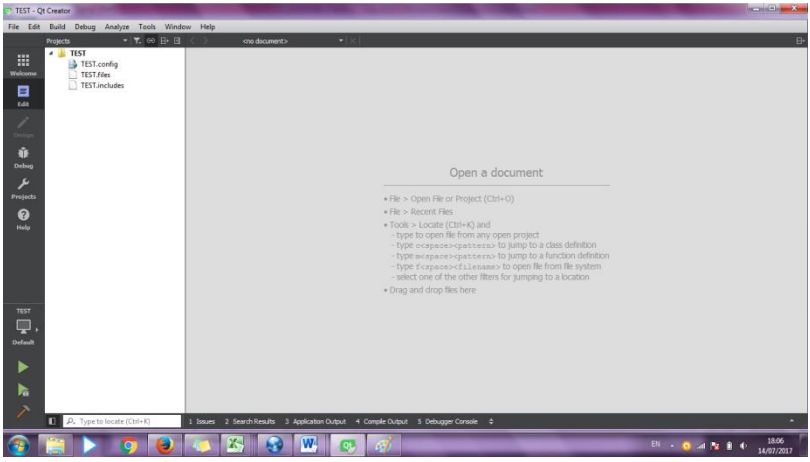
Gambar 3.12 *import existing project file selection*

f. Setelah itu akan muncul kotak *project management*. Lalu klik *finish*.



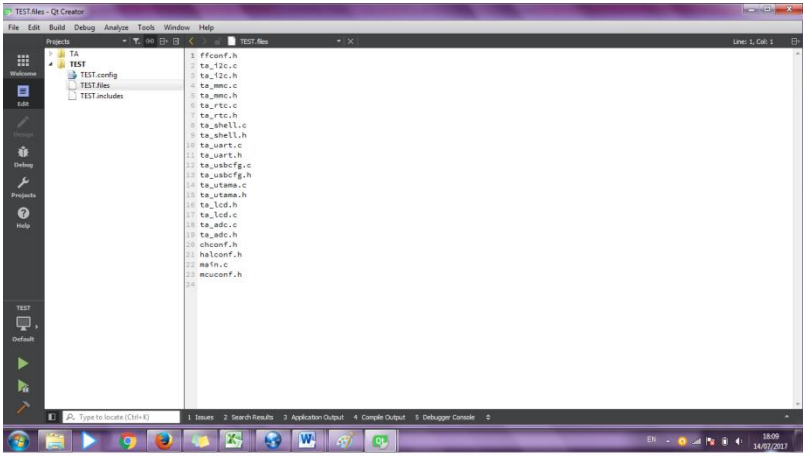
Gambar 3.13 *import existing project management*

g. Setelah itu akan muncul *project* yang berisi *file* utama meliputi .config, .files, .includes.



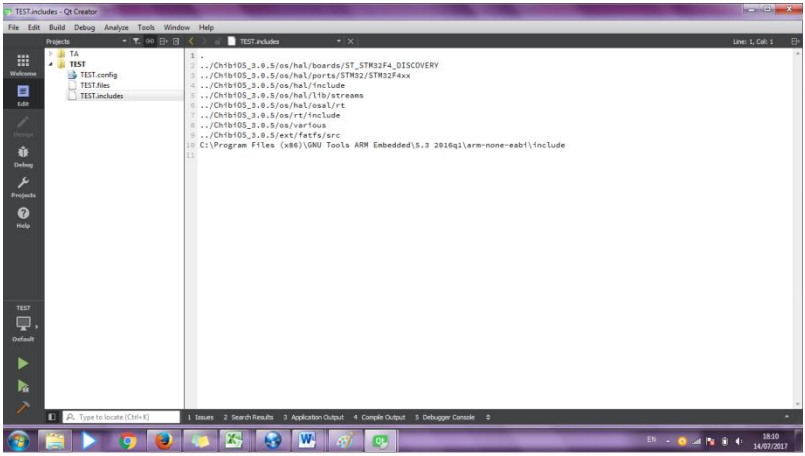
Gambar 3.14 *Tampilan awal program*

h. Kemudian pada project.files diisi fungsi-fungsi yang digunakan sesuai gambar dibawah ini:



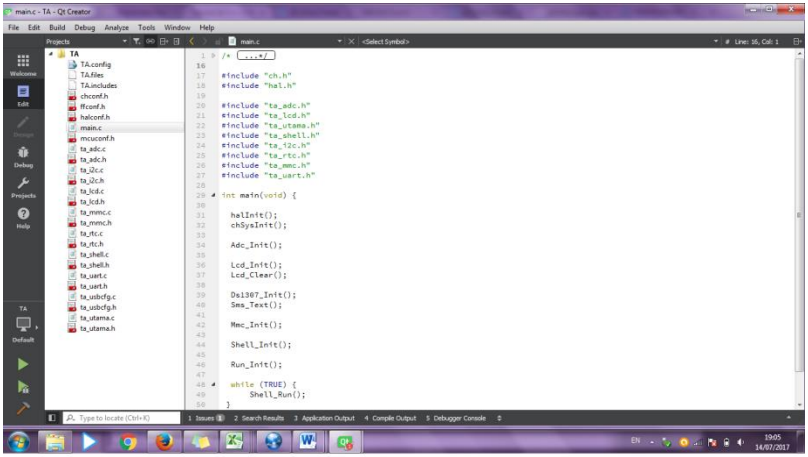
Gambar 3.15 *Program project .files*

i. Pada project.includes diisi dengan *library software* yang digunakan yakni ChibiOS seperti gambar dibawah ini:



Gambar 3.16 *Program project .includes*

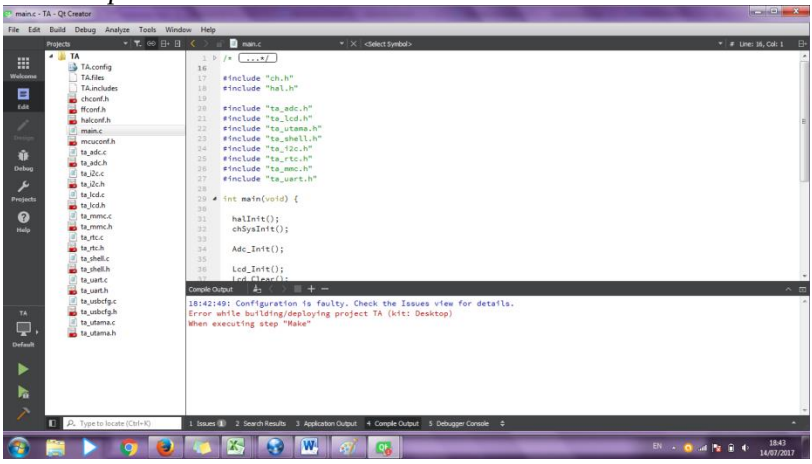
j. Setelah konfigurasi telah dibuat, kemudian ditambahkan *source* dan *header*



Gambar 3.17 *Class pada project*

k. Setelah itu akan muncul program yang telah diatur dan pemrograman bisa dilakukan.

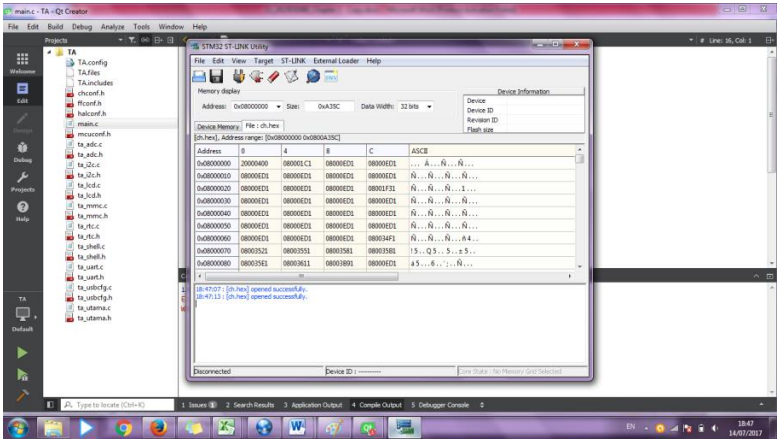
l. Setelah selesai menyusun program, program dapat di-*compile*.



Gambar 3.18 *Build Project Qt Creator*

+ +

m. *Download* ST-LINK V2 sesuai



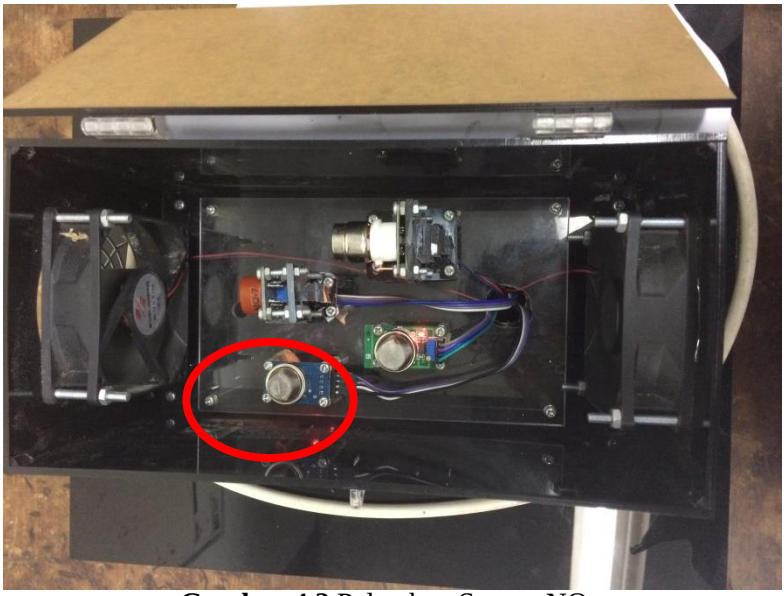
Gambar 3.19 *Download ST-LINK V2*

BAB IV
PENGUJIAN DAN ANALISIS DATA

4.1 Pengujian Sensor Gas MQ - 135

Berikut merupakan hasil pengujian sensor gas NOx menggunakan sensor MQ 135. Pengambilan data dilakukan 2 type yaitu pengambilan saat tanpa terkena gas dan pada saat diberi gas dari sepeda motor supra x 125r berbasis STM32F4 *Discovery*.

Gambar 4.1 *Detector Emission Gass NOx Plant*

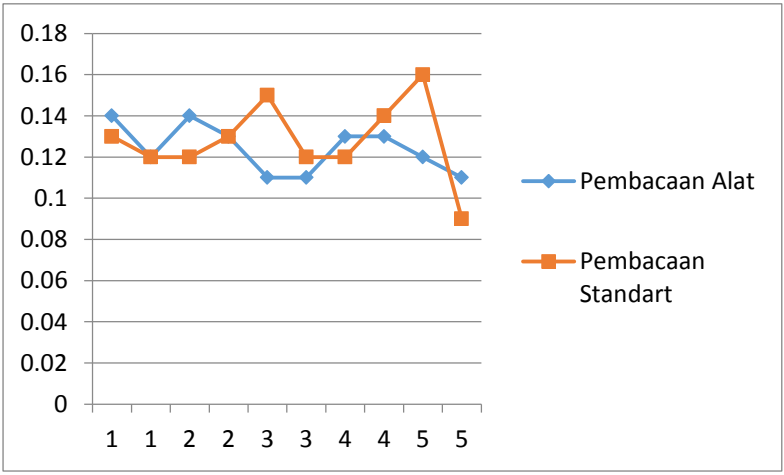


Gambar 4.2 Peletakan Sensor NOx

Pada alat *monitoring* emisi gas buang ini terdapat 4 sensor yaitu sensor CO, SO_2, CO_2, NO_2. Sensor yang digunakan pada gas NO_2 ini yaitu MQ – 135 dimana sensor ini memiliki *range* pengukuran sebesar 0.01ppm – 10ppm.

Tabel 4.1 Hasil Pengambilan Data Emisi Gas NOx pada Pukul 09.45

No.	Jarak	Pembacaan Alat	Pembacaan Standart
1	1	0.14	0.13
2	1	0.12	0.12
3	2	0.14	0.12
4	2	0.13	0.13
5	3	0.11	0.15
6	3	0.11	0.12
7	4	0.13	0.12
8	4	0.13	0.14
9	5	0.12	0.16
10	5	0.11	0.09

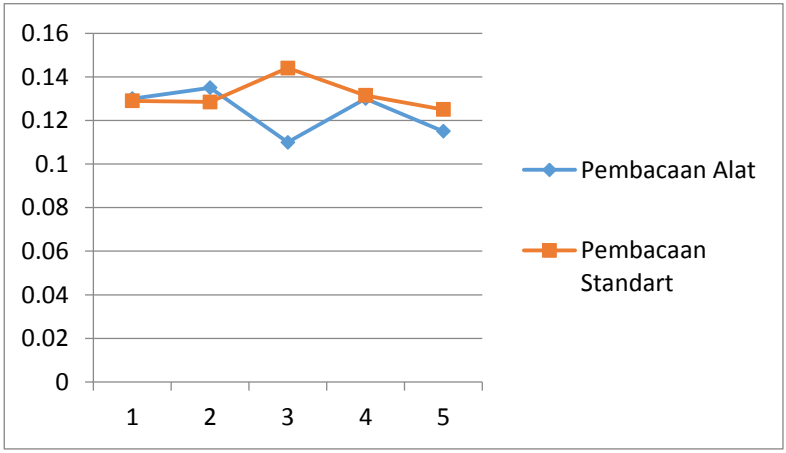


Gambar 4.3 Grafik Pembacaan Alat Satndart dan Uji

Dilakukan pengambilan data pada pukul 09.45 dengan menggunakan sumber gas yaitu berasal dari pembuangan gas sepeda motor.

Tabel 4.2 Pembacaan Sensor pada Gas Buang Kendaraan Sepeda Motor Supra X 125r Menggunakan Variasi Jarak.

Variasi Jarak (Meter)	Pembacaan Alat (xi) PPM	Pembacaan Standart (Xi) PPM
1	0.13	0.12
2	0.13	0.12
3	0.11	0.14
4	0.13	0.13
5	0.11	0.12



Gambar 4.4 Grafik Pembacaan Standart dan Alat

4.1.1 Prosedur Kalibrasi

Rangkaian kegiatan kalibrasi secara sederhana dapat digambarkan sebagai kegiatan persiapan kalibrasi, pelaksanaan kalibrasi, perhitungan data kalibrasi, penentuan ketidakpastian dan penerbitan laporan kalibrasi.

1. Persiapan kalibrasi

- Persiapan alat standar dan alat yang dikalibrasi

Alat yang akan dikalibrasi dan alat standar dikondisikan pada kondisi yang sama sesuai metode kalibrasi, hal ini diperlukan untuk menghindarkan perbedaan hasil ukur akibat pengaruh lingkungan.

- Pelaksana kalibrasi

Pelaksana kalibrasi harus dipilih orang yang mengerti tentang kalibrasi yang akan dilaksanakan, misalnya telah pernah mengikuti kursus kalibrasi, telah berpengalaman dibidangnya, dan dalam hal tertentu memerlukan persyaratan latar belakang pendidikan atau persyaratan fisik tertentu (misalnya tidak boleh buta warna). Hal ini diperlukan untuk menghindari kesalahan pengambilan data ukur.

- Kondisi lingkungan kalibrasi

Kondisi lingkungan kalibrasi harus diatur sedemikian sesuai persyaratan metode kalibrasi umpama suhu dan kelembaban. Tidak selamanya kalibrasi harus dilakukan pada ruang yang terkondisi dengan ketat. Pengkondisian lingkungan kalibrasi biasanya dilakukan untuk kalibrasi peralatan yang mudah berubah akibat pengaruh suhu, kelembaban, getaran, cahaya, dan sebagainya.

- Metode kalibrasi

Metode kalibrasi dapat mengacu kepada metode standar internasional maupun metode standar lainnya semisal text book, jurnal, buletin, dan manual peralatan, namun perlu diperhatikan bahwa acuan tersebut harus merupakan publikasi yang diakui masyarakat luas. Selain itu dari beberapa pilihan metode kalibrasi dapat dipilih metode yang mudah dilaksanakan, karena sulitnya mengikuti metode kalibrasi dapat berakibat kesalahan dalam pengambilan data kalibrasi.

2. Pelaksanaan kalibrasi

- Pengamatan awal

Jika alat yang dikalibrasi berupa instrumen, pastikan bahwa alat tersebut dapat beroperasi normal. Jika alat berupa objek ukur pastikan bahwa alat mempunyai bentuk sempurna. Pada prinsipnya pelaksanaan kalibrasi tidak bertujuan untuk memperbaiki alat, karenanya alat yang tidak normal seyogyanya tidak boleh dikalibrasi. Alat demikian harus diperbaiki dulu oleh petugas yang khusus menangani perbaikan alat hingga alat tersebut diyakini beroperasi normal.

- Penyetelan

Penyetelan alat yang akan dikalibrasi biasanya diperlukan untuk menghindari kesalahan titik nol. Penyetelan dapat berupa menyetel kedataran, pembersihan alat dari kotoran, menyetel titik nol, dalam hal misalnya kalibrasi neraca elektronik penyetelan dapat berupa kalibrasi internal sesuai prosedur dalam manual.

- Pengamatan kewajaran hasil ukur

Pengamatan ini dimaksudkan untuk memastikan kewajaran penunjukan alat. Jika alat menunjukan hasil ukur yang tidak wajar mungkin perlu penyetelan kembali atau perlu dicari penyebab ketidakwajaran penunjukan alat tersebut.

- Pengukuran

Pengukuran dilakukan pada titik ukur tertentu seperti dinyatakan dalam dokumen acuan kalibrasi sesuai kapasitas alat atau rentang ukur tertentu yang biasa digunakan oleh pengguna alat. Jika dokumen acuan kalibrasi tidak menyatakan titik ukur, biasanya pengukuran dilakukan dalam selang 10% dari kapasitas ukur alat. Titik uku harus dibuat mudah dibaca oleh pengguna alat. Pada waktu pengukuran hanyalah melakukan pengambilan data dan tidak boleh melakukan kegiatan lainnya yang mungkin menyebabkan pembacaan atau pencatatan menjadi salah.

- Pencatatan

Pencatatan hasil ukur harus berdasar kepada apa yang dilihat bukan kepada apa yang dirasakan. Pencatatan dilakukan seobjektif mungkin menggunakan format yang telah dirancang dengan teliti sesuai dengan ketentuan metode kalibrasi. Selain

data ukur hal yang perlu dicatat adalah identitas alat selengkapnya serta faktor yang mempengaruhi kalibrasi seperti suhu ruangan, kelembaban, tekanan udara dan sebagainya.

3. Perhitungan

Data kalibrasi yang diperoleh dihitung sesuai metode kalibrasi. Perhitungan biasanya melibatkan pekerjaan mengkonversi satuan, menghitung nilai maksimum-minimum, nilai rata-rata, standar deviasi, atau menentukan persamaan regresi. Hasil perhitungan akan menjadi dasar dalam penarikan kesimpulan dan penentuan ketidakpastian kalibrasi.

4. Penentuan ketidakpastian

Penentuan ketidakpastian kalibrasi diperlukan karena ternyata bahwa hasil kalibrasi yang diperoleh dipengaruhi oleh berbagai faktor antara lain operator, alat kalibrasi, alat bersangkutan, lingkungan, metode kalibrasi. Besarnya pengaruh faktor-faktor tersebut ada yang dominan dan ada pula yang dapat diabaikan tergantung jenis kalibrasi yang dilakukan. Dengan demikian nilai telusur atau kesalahan sistematik yang diperoleh dari kalibrasi tidak berada di satu titik tertentu melainkan dalam suatu rentang nilai sebesar nilai ketidakpastian kalibrasi. Untuk keterangan lebih rinci termuat dalam butir 8.

5. Laporan kalibrasi

Format laporan kalibrasi hendaknya mengacu kepada pedoman SNI 19-17025. Proses penerbitan laporan kalibrasi secara sederhana meliputi tahap:

- Pengkonsepan

Pengkonsepan laporan berdasarkan hasil pengukuran, perhitungan data, dan perhitungan ketidakpastian;

- Pemeriksaan konsep

Pemeriksaan konsep oleh petugas yang berwenang untuk mengecek kesalahan identitas alat, pengambilan data, kesalahan perhitungan data dan perhitungan ketidakpastian;

- Pengetikan konsep

Pengetikan konsep laporan dan pemeriksaan kebenaran pengetikan dengan cara membandingkan antara konsep laporan dengan konsep net laporan.

- Pengesahan laporan

Pengesahan laporan. Biasanya yang mengesahkan laporan kalibrasi adalah kepala laboratorium kalibrasi atau seseorang yang ditunjuk atas dasar pengetahuannya di bidang kalibrasi

Tabel 4.3 Data Kalibrasi Pada Sensor MQ – 135 (A)

No.	Pembacaan Alat (xi)	Pembacaan Standart (Xi)	Koreksi (Yi)
1	0.13	0.12	-0.001
2	0.13	0.12	-0.0065
3	0.11	0.14	0.034
4	0.13	0.13	0.0015
5	0.11	0.12	0.01
Jumlah	0.62	0.65	0.03
Rata – Rata	0.12	0.13	0.0076

Tabel 4.4 Data Kalibrasi Pada Sensor MQ 135 (B)

No.	XiYi	Xi^2	Yi^2
1	-0.00013	0.016641	0.000001
2	-0.00084	0.0165123	4.23E-05
3	0.004896	0.020736	0.001156
4	0.000197	0.0172923	0.00000225
5	0.00125	0.015625	0.0001
Jumlah	0.005379	0.0868065	0.0013015

Tabel 4.5 Data Kalibrasi Pada Sensor MQ 135 (C)

No.	b	a	Yi-Ybar	D-(bM+a)^2
1	1.769	0.225	0.0086	1.59888E-05
2			0.0141	7.4196E-05
3			0.0264	1.98456E-05
4			0.0061	3.50822E-05
5			0.0024	0.00019826
Jumlah			0.0576	0.000343373

A. Analisa *Type* A, (U_a)

Pada tipe ini biasanya ditandai dengan adanya data pengukuran, misalnya n kali pengukuran, maka selanjutnya dari data tersebut, akan ditemukan nilai rata-ratanya, standar deviasinya, dan atau *repeatability*-nya. Bentuk kurva dari tipe ini adalah sebaran Gauss. Rumus umum ketidakpatian untuk tipe A ini adalah:

$$Ua_1 = \frac{\sigma}{\sqrt{n}} \quad \textbf{(4.1)}$$

Dimana:
σ = Standar deviasi
n = Banyaknya data

Rumus standar deviasi (σ) sendiri adalah sebagai berikut:

$$\sigma = \frac{\sqrt{\sum(y_i - \bar{y})^2}}{n-1} \quad \textbf{(4.2)}$$

$$\sigma = \frac{\sqrt{\sum(0)^2}}{5-1}$$

$$\sigma = \mathbf{0}$$

Dimana:
y_i = nilai koreksi ke-i
$\bar{y}$ = rata-rata nilai koreksi
σ = Standard Deviasi

Ua_1 = 0

Sedangkan untuk Ua_2 rumusnya dapat diketahui seperti di bawah ini:

$$Ua_2 = \sqrt{\frac{SSR}{n}} \quad \textbf{(4.3)}$$

$$Ua_2 = \sqrt{\frac{0.000343373}{5}}$$

$$Ua_2 = 0.0106$$

Dimana:
SSR *(Sum Square Residual)* = ΣSR(*Square Residual*)
SR = R^2 (*Residu*)

$$SSR = \sum R^2 \quad \textbf{(4.4)}$$

$$SSR = 0.000343373$$

$$R = Y_i - Y_{Reg} \quad \textbf{(4.5)}$$

R_1 = -0.4546

R_2 = -0.45922

R_3 = -0.446

R_4 = -0.456

R_5 = -0.436

$Y_{Reg} = a + bX_i$ ..... **(4.6)**

$Y_{Reg1} = 0.453$

$Y_{Reg2} = 0.452$

$Y_{Reg3} = 0.480$

$Y_{Reg4} = 0.458$

$Y_{Reg5} = 0.446$

$a = \bar{y} + (b\bar{x})$ ..... **(4.7)**

$a = 0.0076 + (1.76977071.1.76977071)$

$a =$ **0.225**

$b = \frac{n.\sum x_i y_i - \sum xi \sum y_i}{n.\ \sum xi^2 - (\sum xi)^2}$ ..... **(4.8)**

$b = \frac{5.\sum 0.005379 - \sum 0.658 \sum 0.0076}{5.\ \sum 0.0868065\ \ - (\sum 0.658)^2}$

$b =$ **1.769**

B. Nilai Ketidakpastian *Type* B :

Pada tipe ini terdapat 2 parameter ketidakpastian, yaitu ketidakpastian Resolusi (U_{b1}) dan Ketidakpastian alat standar (U_{b2}). Dengan perhitungan sebagai berikut:

$\mathbf{U_{b1}} = \frac{a}{\sqrt{3}}$,

$\mathbf{U_{b1}} = \frac{\mathbf{0.225}}{\sqrt{3}}$,

$\mathbf{U_{b1}}$ = 0.1325

$\mathbf{U_{b2}} = \frac{\frac{1}{2}x\ Resolusi}{\sqrt{3}} = \frac{\frac{1}{2}x\ 0.01}{\sqrt{3}} =$ **0.0029412**

C. Nilai ketidakpastian kombinasi U_c:

$U_c = \sqrt{U_{a1}{}^2 + U_{a2}{}^2 + U_{b1}{}^2 + U_{b2}{}^2}$ .....**(4.9)**

$U_c = \sqrt{0^2 + 0{,}0106^2 + 0{,}0029^2 + 0{,}1325^2}$

U_c = 0.132

Dengan kondisi V atau derajat kebebasan dari kedua tipe ketidakpastian, sebagai berikut :

V = n-1, sehingga :

V1 = 4; V2 = 4; V3 = 50; V4 = 0 (berdasarkan table *T-Student*)

Dengan nilai Veff (Nilai derajat kebebasan effektif) sebagai berikut :

$Veff = \frac{(U_C)^4}{\sum^{(U_i)^4}/_{V_i}}$.....**(4.10)**

$Veff = \frac{(0.0549436)^4}{(0)^4/_4 + (0.0106)^4/_4 + (0{,}0029412)^4/_{50} + (0{,}1325)^4/_{50}}$

Veff = 4.426

Pada saat diketahui Veffnya maka dimasukan ke dalam rumus interpolasi untuk mencari nilai k (derajat kebebasan)

$$\text{Interpolasi} = \frac{x - x1}{x2 - x1} = \frac{y - y1}{y2 - y1}$$

$$= \frac{4{,}4 - 4}{5 - 4} = \frac{y - 2{,}776}{2{,}57 - 2{,}776}$$

$$= 2{,}693$$

Oleh karena itu, hasil nilai ketidakpastian diperluang sebesar :

$U_{exp} = k \; x \; U_c$ ..... **(4.11)**

$$\boldsymbol{U_{exp}} = 2{,}693 \; x \; 0{,}132 = 0{,}3$$

Sehingga berdasarkan perhitungan ketidakpastian diperluas di atas menghasilkan nilai ketidakpastian alat sebesar :

x = 0,14 ± 0,3
cl = 95%
k = 2,693

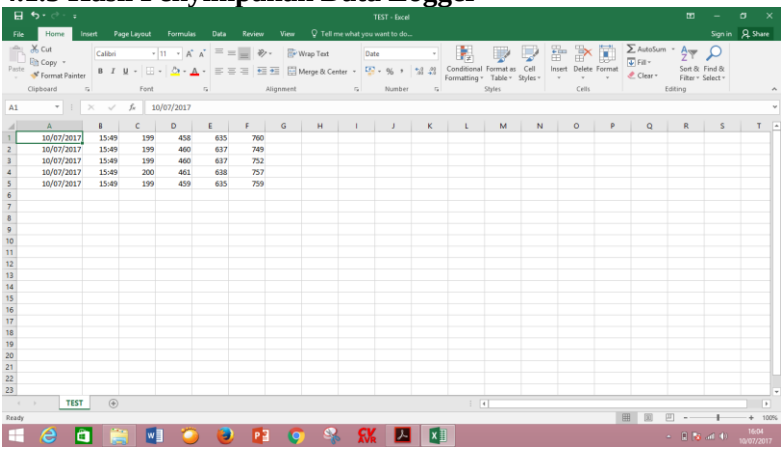
4.1.2 Karakteristik Statik pada Sensor MQ – 135

Karakteristik statis suatu alat ukur adalah karakteristik yang harus diperhatikan apabila alat tersebut digunakan untuk mengukur suatu kondisi yang tidak berubah karena waktu atau hanya berubah secara lambat laun.Karakteristik statis adalah hal-hal yang harus diperhitungkan bila alat ukur dipergunakan untuk mengukur suatu keadaan yang tidak bergantung pada waktu, yaitu :

1. *Range* : 0,01 ppm – 10ppm
2. Akurasi : 84,52 %
3. *Span* : 9,99 ppm
4. Sensitivitas : 0,42
5. *Non* linieritas : 5,23
6. Resolusi : 0,01

7. Kesalahan : 16 %

4.1.3 Hasil Penyimpanan Data Logger

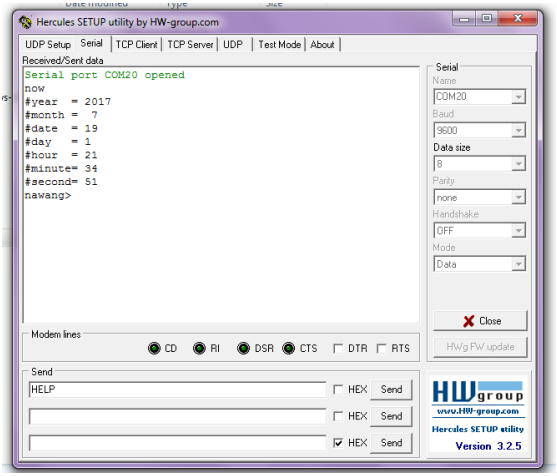


	A	B	C	D	E	F
1	10/07/2017	15:49	199	458	635	760
2	10/07/2017	15:49	199	460	637	749
3	10/07/2017	15:49	199	460	637	752
4	10/07/2017	15:49	200	461	638	757
5	10/07/2017	15:49	199	459	635	759

Gambar 4.5 Penyimpanan Data *Logger*

Pada gambar diatas ditampilkan data logger pada excel. Data logger tersebut meliputi tanggal, waktu, ADC pada ke empat sensor.

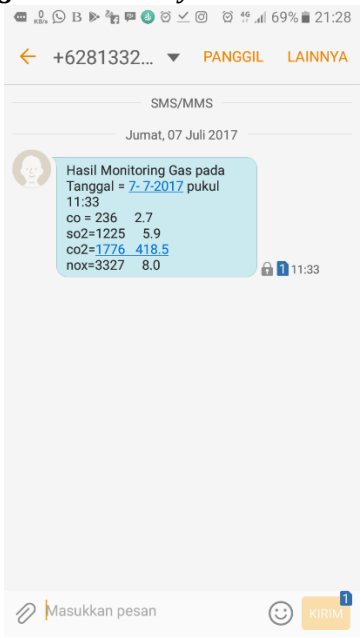
4.1.4 Hasil Pembuatan *Real Time Clock*



Gambar 4.6 Hasil Pembuatan *Real Time Clock*

RTC (Real Time Clock) merupakan sebuah IC yang memiliki fungsi untuk menghitung waktu, mulai dari detik, menit, jam, tanggal, bulan, serta tahun.

4.1.5 Hasil *Testing SMS Gateway*



Gambar 4.7 Hasil Pengiriman *SMS Gateway*

Sms akan diterima oleh *user* ketika nilai ppm yang ditentukan oleh *user* mencapai ppm yang ditentukan atau bahkan melebihi ppm yang ditentukan oleh *user*

4.2 Analisis Sistem

Pada saat melakukan uji emisi gas tempat yang digunakan yaitu parkiran teknik fisika pada jam 09.30 – 12.00 dengan menggunakan gas buang kendaraan motor supra x 125r dengan menggunakan variasi jarak dengan rentang 1 meter. Pada saat udara normal nilai ppm pada alat uji yaitu sebesar 0,05 hal ini

sesuai dengan kualitas udara normal yaitu sebesar 0,03ppm – 0,09ppm. Pada saat pengambilan data dilakukan pengambilan data sebanyak 10 kali dengan catatan diambil 2 data pada jarak yang sama. Setelah tercapai 10 data maka diambil rata – rata pembacaan dari setiap jarak tersebut. Pada jarak 1- 5 meter data yang didapatkan dari pembacaan alat yaitu sebesar 0,13ppm; 0,135ppm; 0,11ppm; 0,13ppm; 0,115ppm. Pada pembacaan standart didapatkan data sebesar 0,129ppm; 0,1285ppm; 0,144 ppm; 0,1315 ppm; 0,125ppm.

Dari data tersebut kemudian diolah untuk mencari nilai ketidakpastian agar alat yang diuji sesuai dengan alat standart. Dari pengolahan data yang telah dilakukan didapaatkan hasil analisa tipe A (Ua) yaitu Ua1 sebesar 0,053 dan Ua2 sebesar 0,0106. Dan analisa tipe B (Ub) yaitu Ub1 sebesar 0,0029 dan Ub2 sebesar 0, $\mathbf{U}_{b2}$ = 0,1325. Setelah diketahui nilai analisa tipe A dan B selanjutnya dicari Nilai ketidakpastian kombinasi dan didapatkan nilai sebesar 0,0549. Dan derajat kebebasan effektifnya didapatkan nilai sebesar 4,42 Sehingga berdasarkan perhitungan ketidakpastian diperluas di atas menghasilkan nilai ketidakpastian alat sebesar : x = 0,14 ± 0,3;cl = 95%;k = 2,693 Alat yang digunakan pada uji emisi gas NOx ini terbilang layak untuk digunakan karena tidak melebihi 5% dari nilai rata – rata pembacaan alat dan nilai rata – rata pembacaan alat tidak melebihi nilai 3 σ.

Pada data yang diambil seharusnya dari jarak terdekat hingga terjauh akan mengalami penurunan nilai ppm. Sedangkan pada saat menguji alat nilai ppm yang didapatkan tidak menentu tetapi menunjukan *trend* penurunan nilai ppm. Hal ini disebabkan bahwa nilai NOx diudara dan di gas buang dari kendaraan bermotor cukup kecil dan disebabkan oleh pengaruh angin yang berhembus dapat menyebabkan gas yang dikeluarkan oleh sepeda motor hilang bahkan sudah tercampur dengan yang lain.

Jika dilihat dari karakteristik statik sensor MQ – 135 didapatkan hasil perhitungan diantaranya yaitu : *Range* kemampuan sensor untuk mambaca nilai ppm antara 0,01 ppm – 10 ppm, pada akurasi didapatkan nilai sebesar 84,52% jadi tingkat

akurasi sensor MQ – 135 sebesar 84,52%, nilai span didapatkan sebesar 9,99 ppm, nilai pada sensitivitas dan linieritas sebesar 0,42 dan 5,23. Nilai tingkat kesalahan pembacaan pada sensor sebesar 16%.

“Halaman ini sengaja dikosongkan”

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan pada hasil penelitian tugas akhir yang sudah dilakukan dapat disimpulkan sebagai berikut:

1. Telah berhasil dibuat rancang bangun monitoring gas NOx menggunakan sensor MQ – 135 dengan berbasis mikrokontroler STM32F4 *Discovery* dengan spesifikasi alat yaitu *range* : 0,01 ppm – 10 ppm; akurasi : 84,52%; *span* 9,99 ppm; Sensitivitas :0,42; *non* linieritas : 5,23; resolusi : 0,01; kesalahan : 16%
2. Dari kalibrasi alat yang telah dilakukan dapat diketahui bahwa alat monitoring gas kadar NOx ini dengan faktor cakupan sebesar 2,693 dengan tingkat kepercayaan 95% hasil pengukuran (x) didapatkan $0{,}14 \pm 0{,}3$ ppm.
3. Alat monitoring ini dapat menyimpan data dengan menggunakan *SD Card* yang dapat ditampilkan data *monitoring* pengukurannya nya pada program *excel*.

5.2 Saran

Saran yang diberikan untuk dilakukan penelitian selanjutnya yaitu:

1. Dibutuhkan Jumper yang baik karena pada saat merangkai *LCD* beberapa kali *jumper* sering putus dikarenakan kualitas *jumper* yang kurang memadai.
2. Alat ini dapat disempurnakan denan menambahkan *Safety* berupa MCB (*Mini Circuit Breaker)* untuk mencegah terjadinya konsleting pada arus listrik

“Halaman ini sengaja dikosongkan”

DAFTAR PUSTAKA

[1] http://herypurba-fst.web.unair.ac.id/artikel_detail-41623UmumPERUBAHAN%20IKLIM%20GLOBAL.html. Diakses pada tanggal 14 Juni 2017.

[2] www.generasibiologi.com › Lingkungan. Diakses pada tanggal 14 Juni 2017

[3] www.energi.lipi.go.id/utama.cgi?artikel&1065978843&10 Diakses pada tanggal 14 Juni 2017.

[4] ejurnal.bppt.go.id/index.php/JTL/article/view/269/169 Diakses pada tanggal 6 Desember 2015.

[5] aldilah-bagas-d.blog.ugm.ac.id/2012/06/17/pencemaran-udara/ Diakses pada tanggal 15 Juni 2017

[6] https://www.slideshare.net/Linda.../pencemaran-udara-28796890 Diakses pada tanggal 15 Juni 2017

[7] http://webcache.googleusercontent.com/search?q=cache:KteKs_DcYEkJ:airpollution2014.weebly.com/dampakpencemaran-udara---nitrogen-oksida+&cd=2&hl=en&ct=clnk&gl=id.Diakses pada tanggal 17 Juni 2017

[8] http://airpollution2014.weebly.com/dampak-pencemaran-udara---nitrogen-oksida Diakses pada tanggal 17 Juni 2017

[9] http://www.lamudi.co.id/journal/pengertian-exhaust-fan-dan-cara-memilihnya/ Diakses pada tanggal 17 Juni 2017

[10] blog.nextsys.web.id/2013/01/stm32-discovery-board/ Diakses pada tanggal 17 Juni 2017

[11] www.academia.edu/9903603/Arsitektur_Komputer Diakses pada tanggal 17 Juni 2017

[12] https://www.academia.edu/11491199/Alat_Ukur_Kualitas_Udara_Menggunakan_Sensor_Gas_MQ135_Berbasis_Mikrokontroller_Atmega16A Diakses pada tanggal 18 Juni 2017

[13] Agfianto Eko Putra,Teknik antar muka computer : konsep & aplikasi,Penerbit Graha Ilmu, Yogyakarta, 2002

[14] angenano.blog.pcr.ac.id/2017/04/17/pertemuan-12-modul-11-lcd/ Diaksess pada tanggal 18 Juni 2017

[15] www.ngarep.net/tutorial-arduino-mengakses-modul-micro-sd/ Diakses pada tanggal 18 Juni 2017
[16] andidinata.com/2017/06/rtc/ Diakses pada tanggal 19 Juni 2017
[17] kiswara.net › M2M Wavecom › Modem Wavecom Diakses pada tanggal 20 Juni 2017
[18] teknikelektronika.com › Komponen Elektronika Diakses pada tanggal 20 Juni 2017
[19] www.atl123.com/chibios-rt.html Diakses pada tanggal 20 Juni 2017
[20] BSN, Persyaratan Umum Kompetensi Laboratorium Pengujian dan Laboratorium Kalibrasi, ISO/IEC 17025, Edisi Kedua, Jakarta, 2005.
[21] JULIA KANT ASUBRA TA, Dasar Ketidakpastian Pengukuran, Pusat Penelitian Kimia LIPI,Jakarta,2003.

LAMPIRAN A
(SPESIFIKASI SENSOR MQ - 135)

A. **Spesifikasi Sensor MQ – 135**

TECHNICAL DATA MQ-135 GAS SENSOR

FEATURES

Wide detecting scope Fast response and High sensitivity Stable and long life Simple drive circuit

APPLICATION

They are used in air quality control equipments for buildings/offices, are suitable for detecting of NH3,NOx ,etc.

SPECIFICATIONS

A.Standard work condition

Symbol	Parameter name	Technical	Remarks
Vc	Circuit voltage	5V±0.1	AC OR DC
VH	Heating voltage	5V±0.1	ACOR DC
RL	Load resistance	can adjust	
RH	Heater	33Ω±5%	Room Tem
PH	Heating	less than 800mw	

B. Environment condition

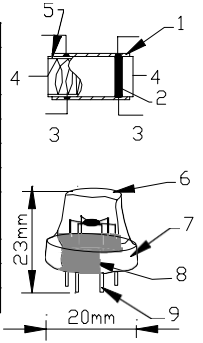
Symbol	Parameter name	Technical condition	Remarks
Tao	Using Tem	-10 -45	
Tas	Storage Tem	-20 -70	
R_H	Related humidity	less than 95%Rh	
O_2	Oxygen concentration	21%(standard condition)Oxygen concentration can affect sensitivity	minimum value is over 2%

C. Sensitivity characteristic

Symbol	Parameter name	Technical parameter	Ramark 2
Rs	Sensing Resistance	30KΩ-200KΩ (100ppm NH_3)	Detecting concentration scope 10ppm-300ppm NH_3 NOx 0.01ppm-10ppm
Standard Detecting Condition	Temp: 20 ±2 Vc:5V±0.1 Humidity: 65%±5% Vh: 5V±0.1		
Preheat time	Over 24 hour		

D. Structure and configuration, basic measuring circuit

	Parts	Materials
1	Gas sensing layer	Au
2	Electrode Electrode line	Pt
3	Heater coil Tubular ceramic	Ni-Cr alloy Al_2O_3
4	Anti-explosion network	Stainless steel gauze (SUS316 100 mesh)



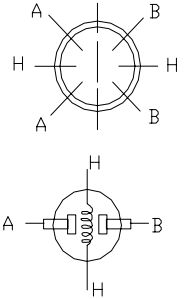


Fig. 1

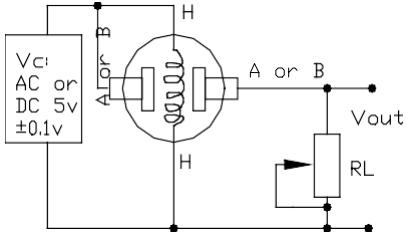
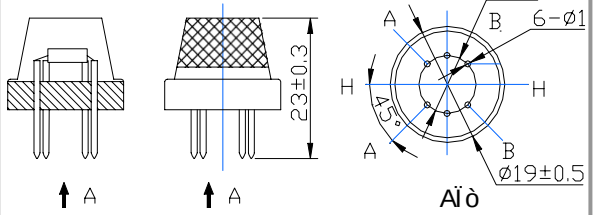
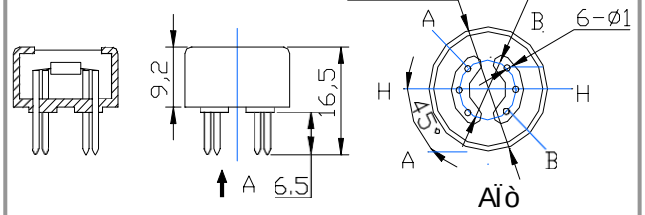


Fig.2

Configuration **A**



Configuration **B**



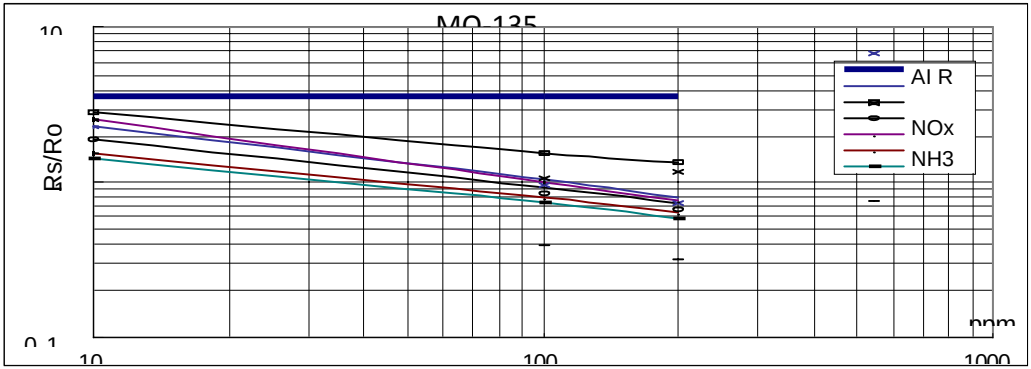
Structure and configuration of MQ-135 gas sensor is shown as Fig. 1 (Configuration **A or B),** sensor composed by micro AL2O3 ceramic tube, Tin Dioxide (SnO2) sensitive layer, measuring electrode and heater are fixed into a crust made by plastic and stainless steel net. The heater provides necessary work conditions for work of sensitive components. The enveloped MQ-135 have 6 pin ,4 of them are used to fetch signals, and other 2 are used for providing heating current.

Electric parameter measurement circuit is shown as Fig.2

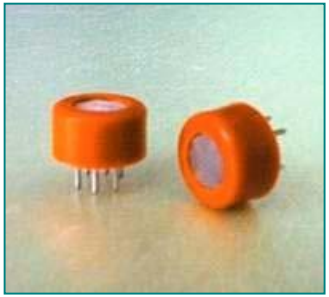
E. Sensitivity characteristic curve

Fig.2 sensitivity characteristics of the MQ-135

Fig.3 is shows the typical sensitivity characteristics of the MQ-135 for several gases.
in their: Temp: 20 Humidity: 65%
O_2 concentration 21% RL=20kΩ
Ro: sensor resistance at 10ppm of NOx in the clean air.
Rs:sensor resistance at various concentrations of gases.



Gambar A.1 Grafik Karekteristik Sensitivitas

Gambar A.2 Bentuk Fisik Sensor MQ - 135

LAMPIRAN B
(*LISTING* PROGRAM PADA *CHIBIOS*)

//main.c

```
#include "ch.h"
#include "hal.h"
#include "ta_adc.h"
#include "ta_lcd.h"
#include "ta_utama.h"
#include "ta_shell.h"
#include "ta_i2c.h"
#include "ta_rtc.h"
#include "ta_mmc.h"
#include "ta_uart.h"
```

```
int main(void) {
  halInit();
  chSysInit();
  Adc_Init();
  Lcd_Init();
  Lcd_Clear();
  Ds1307_Init();
  Sms_Text();
  Mmc_Init();
  Shell_Init();
  Run_Init();
  while (TRUE) {
     Shell_Run();
 }
}
```

//adc.c

```
#include "ta_adc.h"

static  adcsample_t  samples[ADC_GRP1_NUM_CHANNELS  *
        ADC_GRP1_BUF_DEPTH];
adcsample_t adc0,adc1,adc2,adc3;
uint32_t sum_adc0,sum_adc1,sum_adc2,sum_adc3;
void adccb(ADCDriver *adcp, adcsample_t *buffer, size_t n){
  (void) buffer; (void) n;
  int i;
  if (adcp->state == ADC_COMPLETE) {
    sum_adc0=0;
    sum_adc1=0;
    sum_adc2=0;
    sum_adc3=0;
    for(i=0;i<ADC_GRP1_BUF_DEPTH;i++){
         sum_adc0=sum_adc0+samples[0+(i*ADC_GRP1_NUM_
         CHANNELS)];
         sum_adc1=sum_adc1+samples[1+(i*ADC_GRP1_NUM_
         CHANNELS)];
```

```
        sum_adc2=sum_adc2+samples[2+(i*ADC_GRP1_NUM_
        CHANNELS)];
        sum_adc3=sum_adc3+samples[3+(i*ADC_GRP1_NUM_
        CHANNELS)];
    }
    adc0=sum_adc0/ADC_GRP1_BUF_DEPTH;
    adc1=sum_adc1/ADC_GRP1_BUF_DEPTH;
    adc2=sum_adc2/ADC_GRP1_BUF_DEPTH;
    adc3=sum_adc3/ADC_GRP1_BUF_DEPTH;
   }
 }
static const ADCConversionGroup adcgrpcfg = {
  FALSE,
  ADC_GRP1_NUM_CHANNELS,
  adccb,
  NULL,
  /* HW dependent part.*/
  0,
  ADC_CR2_SWSTART,
  0,
  ADC_SMPR2_SMP_AN0(ADC_SAMPLE_112)                |
        ADC_SMPR2_SMP_AN1(ADC_SAMPLE_112)          |
        ADC_SMPR2_SMP_AN2(ADC_SAMPLE_112)          |
        ADC_SMPR2_SMP_AN3(ADC_SAMPLE_112),
  ADC_SQR1_NUM_CH(ADC_GRP1_NUM_CHANNELS),
  0,
  ADC_SQR3_SQ1_N(ADC_CHANNEL_IN0)                  |
        ADC_SQR3_SQ2_N(ADC_CHANNEL_IN1)            |
        ADC_SQR3_SQ3_N(ADC_CHANNEL_IN2)            |
        ADC_SQR3_SQ4_N(ADC_CHANNEL_IN3)
};
static THD_WORKING_AREA(wa_adcThread, 128);
static THD_FUNCTION(adcThread, arg) {
  (void)arg;
  chRegSetThreadName("ADC Run");
  while (TRUE) {
```

```
    chThdSleepMilliseconds(100);
    palSetPad(GPIOD, 12); /* Yellow.  */
    adcStartConversion(&ADCD1,  &adcgrpcfg,  samples,
        ADC_GRP1_BUF_DEPTH);
    chThdSleepMilliseconds(100);
    palClearPad(GPIOD, 12); /* Yellow.  */
  }
}
void Adc_Init(){
    palSetPadMode(GPIOA,0,PAL_MODE_INPUT_ANALOG);
    palSetPadMode(GPIOA,1,PAL_MODE_INPUT_ANALOG);
    palSetPadMode(GPIOA,2,PAL_MODE_INPUT_ANALOG);
    palSetPadMode(GPIOA,3,PAL_MODE_INPUT_ANALOG);
    adcStart(&ADCD1, NULL);
    adcSTM32EnableTSVREFE();
        palSetPadMode(GPIOD,12,PAL_MODE_OUTPUT_PUS
        HPULL);
    chThdCreateStatic(wa_adcThread,  sizeof(wa_adcThread),
        NORMALPRIO, adcThread, NULL);
}
```

//adc.h

```
#ifndef TA_ADC_H
#define TA_ADC_H

#include <stdio.h>
#include <string.h>
#include <stdint.h>
#include <stdarg.h>
#include <stdlib.h>
#include <math.h>
#include "ch.h"
#include "hal.h"
#include "chprintf.h"
#include "memstreams.h"
```

```
#if                !defined(CHPRINTF_USE_FLOAT)                ||
        defined(__DOXYGEN__)
#define CHPRINTF_USE_FLOAT         FALSE
#endif
#define MAX_FILLER 16
#define FLOAT_PRECISION 100
#define ADC_GRP1_NUM_CHANNELS   4
#define ADC_GRP1_BUF_DEPTH      100
void Adc_Init(void);
#endif
```

//i2c.c

```
#include "ta_i2c.h"

static const I2CConfig i2cconfig= {
   OPMODE_I2C,
   400000,
   FAST_DUTY_CYCLE_2,
};
uint8_t readByteI2C(uint8_t addr){
   uint8_t data;
   i2cAcquireBus(&I2CD1);
   (void)
        i2cMasterReceiveTimeout(&I2CD1,addr,&data,1,TIME_I
        NFINITE);
   i2cReleaseBus(&I2CD1);
   return data;
}
void writeByteI2C(uint8_t addr, uint8_t reg, uint8_t val){
   uint8_t cmd[] = {reg, val};
   i2cAcquireBus(&I2CD1);
   (void) i2cMasterTransmitTimeout(&I2CD1, addr, cmd, 2,
        NULL, 0, TIME_INFINITE);
   i2cReleaseBus(&I2CD1);
}
void I2c_Init(void){
```

```
    palSetPadMode(GPIOB,8,PAL_MODE_ALTERNATE(4)      |
        PAL_STM32_OTYPE_OPENDRAIN);
    palSetPadMode(GPIOB,9,PAL_MODE_ALTERNATE(4)      |
        PAL_STM32_OTYPE_OPENDRAIN);
    i2cStart(&I2CD1, &i2cconfig);
}
```

//i2c.h

```
#ifndef TA_I2C_H
#define TA_I2C_H
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <stdint.h>
#include "ch.h"
#include "hal.h"
#include "chprintf.h"
#include "memstreams.h"
#include "chstreams.h"
uint8_t readByteI2C(uint8_t addr);
void writeByteI2C(uint8_t addr, uint8_t reg, uint8_t val);
void I2c_Init(void);
#endif
```

//lcd.c

```
#include "ta_lcd.h"

LcdStream myLCD;
static msg_t put(void *ip, uint8_t chr) {
    (void)ip;
    Lcd_Write_Data(chr);
    return MSG_OK;
}
static const struct LcdStreamVMT vmt = {NULL, NULL, put,
        NULL};
void lsObjectInit(LcdStream *msp) {
```

```
   msp->vmt = &vmt;
}
void Lcd_Pin_Dir(void){
        palSetPadMode(LCD_PORT_CRTL,LCD_PIN_RS,LCD_
        PORT_MODE);
        palSetPadMode(LCD_PORT_CRTL,LCD_PIN_EN,LCD_
        PORT_MODE);
        palSetPadMode(LCD_PORT_DATA,LCD_PIN_D4,LCD_
        PORT_MODE);
        palSetPadMode(LCD_PORT_DATA,LCD_PIN_D5,LCD_
        PORT_MODE);
        palSetPadMode(LCD_PORT_DATA,LCD_PIN_D6,LCD_
        PORT_MODE);
        palSetPadMode(LCD_PORT_DATA,LCD_PIN_D7,LCD_
        PORT_MODE);
}
void Lcd_Write_Data(uint8_t chr){
   palWritePort(LCD_PORT_DATA,(chr & 0xf0));
   palSetPad(LCD_PORT_CRTL,LCD_PIN_RS);
   palSetPad(LCD_PORT_CRTL,LCD_PIN_EN);
   chThdSleepMilliseconds(10);
   palClearPad(LCD_PORT_CRTL,LCD_PIN_EN);
   palClearPad(LCD_PORT_CRTL,LCD_PIN_RS);
   chThdSleepMilliseconds(10);

   palWritePort(LCD_PORT_DATA,((chr & 0x0f)<<4));
   palSetPad(LCD_PORT_CRTL,LCD_PIN_RS);
   palSetPad(LCD_PORT_CRTL,LCD_PIN_EN);
   chThdSleepMilliseconds(10);
   palClearPad(LCD_PORT_CRTL,LCD_PIN_EN);
   palClearPad(LCD_PORT_CRTL,LCD_PIN_RS);
   chThdSleepMilliseconds(10);
}
void Lcd_Write_Command(uint8_t cmd){
   palWritePort(LCD_PORT_DATA,(cmd & 0xf0));
   palSetPad(LCD_PORT_CRTL,LCD_PIN_EN);
```

```
    chThdSleepMilliseconds(10);
    palClearPad(LCD_PORT_CRTL,LCD_PIN_EN);
    chThdSleepMilliseconds(10);

    palWritePort(LCD_PORT_DATA,((cmd & 0x0f)<<4));
    palSetPad(LCD_PORT_CRTL,LCD_PIN_EN);
    chThdSleepMilliseconds(10);
    palClearPad(LCD_PORT_CRTL,LCD_PIN_EN);
    chThdSleepMilliseconds(10);
}
void Lcd_Cursor(uint8_t column, uint8_t line){
    uint8_t position = 0x00;
    if(column>=TLCD_MAXX) column=0;
    if(line>=TLCD_MAXY) line=0;
    switch(line)
    {
        case 0: position = LCD_LINE0_DDRAMADDR+column;
          break;
        case 1: position = LCD_LINE1_DDRAMADDR+column;
          break;
        case 2: position = LCD_LINE2_DDRAMADDR+column;
          break;
        case 3: position = LCD_LINE3_DDRAMADDR+column;
          break;
    }
    Lcd_Write_Command(1<<LCD_DDRAM | position);
}
void Lcd_Init(void){
    lsObjectInit(&myLCD);
    Lcd_Pin_Dir();
    chThdSleepMilliseconds(500);
    palWritePort(LCD_PORT_CRTL,0x00);
    palWritePort(LCD_PORT_DATA,0x00);

    palSetPad(LCD_PORT_DATA,LCD_PIN_D5);
    palSetPad(LCD_PORT_CRTL,LCD_PIN_EN);
```

```
    chThdSleepMilliseconds(40);
    palClearPad(LCD_PORT_CRTL,LCD_PIN_EN);
    chThdSleepMilliseconds(40);

    Lcd_Write_Command(0x28);
    chThdSleepMilliseconds(10);
    Lcd_Write_Command(0x0c);
    chThdSleepMilliseconds(10);
}
void Lcd_Clear (void){
    Lcd_Write_Command(0x01);
    chThdSleepMilliseconds(10);
}
void Lcd_Example(){
    Lcd_Clear();
    Lcd_Cursor(0,0);
    chprintf((BaseSequentialStream *)&myLCD,"A-LCD");
    Lcd_Cursor(0,1);
    chprintf((BaseSequentialStream *)&myLCD,"Works");
    Lcd_Cursor(0,2);
    chprintf((BaseSequentialStream *)&myLCD,"horee");
    Lcd_Cursor(0,3);
    chprintf((BaseSequentialStream *)&myLCD,"yeee");
}
```

//lcd.h

```
#ifndef TA_LCD_H
#define TA_LCD_H
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <stdint.h>
#include "ch.h"
#include "hal.h"
#include "chprintf.h"
#include "memstreams.h"
```

```
#include "chstreams.h"
#define _lcd_stream_data _base_sequential_stream_data
#define LCD_PIN_RS 0
#define LCD_PIN_EN 1
#define LCD_PORT_CRTL GPIOE
#define LCD_PIN_D4 4
#define LCD_PIN_D5 5
#define LCD_PIN_D6 6
#define LCD_PIN_D7 7
#define LCD_PORT_DATA GPIOE
#define                                    LCD_PORT_MODE
        PAL_MODE_OUTPUT_PUSHPULL
#define  TLCD_MAXX          16  // max x-Position (0...15)
#define  TLCD_MAXY           4  // max y-Position (0...1)
#define  LCD_DDRAM           7
#define LCD_LINE0_DDRAMADDR             0x00
#define LCD_LINE1_DDRAMADDR             0x40
#define LCD_LINE2_DDRAMADDR             0x10
#define LCD_LINE3_DDRAMADDR             0x50

struct LcdStreamVMT {
   _base_sequential_stream_methods
};
typedef struct {
   const struct LcdStreamVMT *vmt;
   _base_sequential_stream_data
} LcdStream;

#ifdef __cplusplus
extern "C" {
#endif
void lsObjectInit(LcdStream *msp);
#ifdef __cplusplus
}
#endif
```

```
void Lcd_Pin_Dir(void);
void Lcd_Write_Command(uint8_t cmd);
void Lcd_Write_Data(uint8_t chr);
void Lcd_Init(void);
void Lcd_Cursor(uint8_t column, uint8_t line);
void Lcd_Clear (void);
void Lcd_Example (void);
#endif // LIB_LCD_H
```

//mmc.c

```
#include "ta_mmc.h"

FATFS MMC_FS;
MMCDriver MMCD1;
bool fs_ready = FALSE;
FRESULT err;
uint32_t clusters;
FATFS *fsp;
uint8_t fbuff[1024];
static SPIConfig hs_spicfg = {NULL, GPIOB, 12, 0};
static SPIConfig ls_spicfg = {NULL, GPIOB, 12,SPI_CR1_BR_2
        | SPI_CR1_BR_1};
static MMCConfig mmccfg = {&SPID2, &ls_spicfg,
        &hs_spicfg};
FRESULT scan_files(BaseSequentialStream *chp, char *path) {
  FRESULT res;
  FILINFO fno;
  DIR dir;
  int i;
  char *fn;

#if _USE_LFN
  fno.lfname = 0;
  fno.lfsize = 0;
#endif
  res = f_opendir(&dir, path);
```

```
  if (res == FR_OK) {
    i = strlen(path);
    for (;;) {
      res = f_readdir(&dir, &fno);
      if (res != FR_OK || fno.fname[0] == 0)
        break;
      if (fno.fname[0] == '.')
        continue;
      fn = fno.fname;
      if (fno.fattrib & AM_DIR) {
        path[i++] = '/';
        strcpy(&path[i], fn);
        res = scan_files(chp, path);
        if (res != FR_OK)
          break;
        path[--i] = 0;
      }
      else {
        chprintf(chp, "%s/%s\r\n", path, fn);
      }
    }
  }
  return res;
}
void Mmc_Mount(void) {
    if (fs_ready) {
        return;
    }
    if (mmcConnect(&MMCD1)) {
        return;
    }
    err = f_mount(&MMC_FS,"/",1);
    if (err != FR_OK) {
        mmcDisconnect(&MMCD1);
        fs_ready = FALSE;
        return;
```

```
    }
    fs_ready = TRUE;
}
void Mmc_Unmount(void) {
  f_mount(NULL,"/",1);
  mmcDisconnect(&MMCD1);
  fs_ready = FALSE;
}
FRESULT f_append (
    FIL* fp,           /* [OUT] file object to create */
    const char* path    /* [IN]  file name to be opened */
)
{
    FRESULT fr;
    /* Opens an existing file. If not exist, creates a new file. */
    fr  =  f_open(fp,  path,  FA_WRITE  |  FA_OPEN_ALWAYS  |
         FA_READ);
    if (fr == FR_OK) {
       /* Seek to end of the file to append data */
       fr = f_lseek(fp, f_size(fp));
       if (fr != FR_OK)
          f_close(fp);
    }
    return fr;
}
void Mmc_Init(){
    palSetPadMode(GPIOB,13,PAL_MODE_ALTERNATE(5)     |
         PAL_STM32_OSPEED_HIGHEST); //SCK

         palSetPadMode(GPIOB,12,PAL_MODE_OUTPUT_PUS
         HPULL | PAL_STM32_OSPEED_HIGHEST); //NSS
    palSetPadMode(GPIOC,2,PAL_MODE_ALTERNATE(5));
         //MISO
    palSetPadMode(GPIOC,3,PAL_MODE_ALTERNATE(5)       |
         PAL_STM32_OSPEED_HIGHEST); //MOSI
    palSetPad(GPIOB, 12);
```

```
    mmcObjectInit(&MMCD1);
    mmcStart(&MMCD1, &mmccfg);
    chThdSleepMilliseconds(50);
    Mmc_Mount();
}
```

//mmc.h

```
#ifndef TA_MMC_H
#define TA_MMC_H
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <stdint.h>
#include "ch.h"
#include "hal.h"
#include "chprintf.h"
#include "memstreams.h"
#include "chstreams.h"
#include "evtimer.h"
#include "chvt.h"
#include "ff.h"
#include "ffconf.h"
#define buffer_size 16
FRESULT f_append (FIL* fp,const char* path);
FRESULT scan_files(BaseSequentialStream *chp, char *path);
void Mmc_Mount(void);
void Mmc_Unmount(void);
void Mmc_Init(void);
#endif // TA_MMC_H
```

//rtc.c

```
#include "ta_rtc.h"

struct ds1307_t calendar;
static uint8_t rxbuf[DS1307_RX_DEPTH];
static uint8_t txbuf[DS1307_TX_DEPTH];
```

```
static i2cflags_t errors = 0;
uint8_t bcd2Dec ( uint8_t val )
{
   uint8_t res = ((val/16*10) + (val % 16));
   return res;
}
uint8_t dec2Bcd ( uint8_t val )
{
   uint8_t res = ((val/10*16) + (val%10));
   return res;
}
void setDs1307Date ( msg_t *status, systime_t *tmo, struct
        ds1307_t dsData )
{
   txbuf[0] = DS1307_SECONDS_REG;
   txbuf[1] = dec2Bcd( dsData.seconds );
   txbuf[2] = dec2Bcd( dsData.minutes );
   txbuf[3] = dec2Bcd( dsData.hours );
   txbuf[4] = dec2Bcd( dsData.day );
   txbuf[5] = dec2Bcd( dsData.date );
   txbuf[6] = dec2Bcd( dsData.month );
   txbuf[7] = dec2Bcd( dsData.year - 2000);

   i2cAcquireBus ( &I2CD1 );
   *status = i2cMasterTransmitTimeout ( &I2CD1,
        DS1307_ADDRESS, txbuf, DS1307_TX_DEPTH, rxbuf,
        0, *tmo );
   i2cReleaseBus ( &I2CD1 );
}

struct ds1307_t getDs1307Date ( msg_t *status, systime_t *tmo )
{
   struct ds1307_t dsData;
   txbuf[0] = DS1307_SECONDS_REG;
   i2cAcquireBus( &I2CD1 );
```

```
    *status = i2cMasterTransmitTimeout ( &I2CD1,
        DS1307_ADDRESS, txbuf, 1,rxbuf, 7, *tmo );
    i2cReleaseBus ( &I2CD1 );
    if ( *status != MSG_OK )
    {
       errors = i2cGetErrors ( &I2CD1 );
    }
    else
    {
       dsData.seconds  = bcd2Dec ( rxbuf[0] & 0x7F );
       dsData.minutes  = bcd2Dec ( rxbuf[1] );
       dsData.hours    = bcd2Dec ( rxbuf[2] & 0x3F );
       dsData.day      = bcd2Dec ( rxbuf[3] );
       dsData.date     = bcd2Dec ( rxbuf[4] );
       dsData.month    = bcd2Dec ( rxbuf[5] );
       dsData.year     = bcd2Dec ( rxbuf[6] ) + 2000;
    }
    return dsData;
}
static THD_WORKING_AREA(waRTC, 128);
static THD_FUNCTION(ThdRTC, arg)  {
  (void)arg;
  msg_t status = MSG_OK;
  systime_t timeOut = MS2ST ( 4 );
  chRegSetThreadName("RTC Request");
  while (TRUE) {
    calendar = getDs1307Date ( &status, &timeOut );
    palSetPad(GPIOD, 14);       /* Red.  */
    chThdSleepMilliseconds(500);
    palClearPad(GPIOD, 14);     /* Red.  */
    chThdSleepMilliseconds(500);
  }
}
void Ds1307_Init ( void )
{
    I2c_Init();
```

```
   chThdSleepMilliseconds(500);
       palSetPadMode(GPIOD,14,PAL_MODE_OUTPUT_PUS
       HPULL);
   chThdCreateStatic(waRTC,  sizeof(waRTC),  NORMALPRIO,
       ThdRTC, NULL);
}
```

//rtc.h

```
#ifndef TA_RTC_H
#define TA_RTC_H
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <stdint.h>
#include "ch.h"
#include "hal.h"
#include "chprintf.h"
#include "memstreams.h"
#include "chstreams.h"
#include "shell.h"
#include "ta_i2c.h"
#define DS1307_RX_DEPTH 7
#define DS1307_TX_DEPTH 8
#define DS1307_ADDRESS      0x68
#define DS1307_SECONDS_REG        0x00
typedef struct ds1307_t
{
   uint8_t     seconds;
   uint8_t     minutes;
   uint8_t     hours;
   uint8_t     day;
   uint8_t     date;
   uint8_t     month;
   uint16_t    year;
} ds1307;
uint8_t bcd2Dec ( uint8_t val );
```

```
uint8_t dec2Bcd ( uint8_t val );
void Ds1307_Init ( void );
void setDs1307Date ( msg_t *status, systime_t *tmo, struct
        ds1307_t dsData );
struct ds1307_t getDs1307Date ( msg_t *status, systime_t *tmo );
#endif
```

//shell.c

```
#include "ta_shell.h"

extern uint16_t adc_co,adc_so2,adc_co2,adc_nox;
thread_t *shelltp = NULL;
extern const USBConfig usbcfg;
extern SerialUSBConfig serusbcfg;
extern struct ds1307_t calendar;
extern FATFS MMC_FS;
extern bool fs_ready;
extern uint32_t clusters;
extern FATFS *fsp;
extern uint8_t fbuff[1024];
extern FRESULT err;
SerialUSBDriver SDU1;
static void cmd_mem(BaseSequentialStream *chp, int argc, char
        *argv[]) {
  size_t n, size;
  (void)argv;
  if (argc > 0) {
    chprintf(chp, "Usage: mem\r\n");
    return;
  }
  n = chHeapStatus(NULL, &size);
  chprintf(chp, "core free memory : %u bytes\r\n",
        chCoreGetStatusX());
  chprintf(chp, "heap fragments   : %u\r\n", n);
  chprintf(chp, "heap free total  : %u bytes\r\n", size);
}
```

```
static void cmd_threads(BaseSequentialStream *chp, int argc,
        char *argv[]) {
  static const char *states[] = {CH_STATE_NAMES};
  thread_t *tp;
  (void)argv;
  if (argc > 0) {
    chprintf(chp, "Usage: threads\r\n");
    return;
  }
  chprintf(chp, "    addr    stack prio refs     state time\r\n");
  tp = chRegFirstThread();
  do {
    chprintf(chp, "%08lx %08lx %4lu %4lu %9s\r\n",
           (uint32_t)tp, (uint32_t)tp->p_ctx.r13,
           (uint32_t)tp->p_prio, (uint32_t)(tp->p_refs - 1),
           states[tp->p_state]);
    tp = chRegNextThread(tp);
  } while (tp != NULL);
}
static void cmd_now(BaseSequentialStream *chp, int argc, char
        *argv[]) {

  (void)argv;
  if (argc > 0) {
    chprintf(chp, "Usage: now\r\n");
    return;
  }
  chprintf(chp, "#year  = %4i\r\n",calendar.year);
  chprintf(chp, "#month = %2i\r\n",calendar.month);
  chprintf(chp, "#date  = %2i\r\n",calendar.date);
  chprintf(chp, "#day   = %1i\r\n",calendar.day);
  chprintf(chp, "#hour  = %2i\r\n",calendar.hours);
  chprintf(chp, "#minute= %2i\r\n",calendar.minutes);
  chprintf(chp, "#second= %2i\r\n",calendar.seconds);
}
```

```
static void cmd_settime(BaseSequentialStream *chp, int argc,
        char *argv[]) {
  msg_t status = MSG_OK;
  systime_t timeOut = MS2ST ( 4 );
  if (argc != 3) {
    chprintf(chp, "Usage: settime sec min hr\r\n");
    return;
  }
  calendar.seconds    = atoi(argv[0]);
  calendar.minutes    = atoi(argv[1]);
  calendar.hours      = atoi(argv[2]);
  setDs1307Date( &status, &timeOut, calendar);
  chprintf(chp, "time was set\r\n");
}
static void cmd_setdate(BaseSequentialStream *chp, int argc,
        char *argv[]) {
  msg_t status = MSG_OK;
  systime_t timeOut = MS2ST ( 4 );
  if (argc != 3) {
    chprintf(chp, "Usage: setdate date month year\r\n");
    return;
  }
  calendar.date       = atoi(argv[0]);
  calendar.month      = atoi(argv[1]);
  calendar.year       = atoi(argv[2]);
  setDs1307Date( &status, &timeOut, calendar);
  chprintf(chp, "date was set\r\n");
}
static void cmd_mmctree(BaseSequentialStream *chp, int argc,
        char *argv[]) {
    (void)argv;
    if (argc > 0) {
    chprintf(chp, "mmctree\r\n");
       return;
    }
```

```
    if (!fs_ready) {
      chprintf(chp, "File System not mounted\r\n");
      return;
    }
    err = f_getfree("/", &clusters, &fsp);
    if (err != FR_OK) {
       chprintf(chp, "FS: f_getfree() failed (%i)\r\n",err);
       return;
    }
    chprintf(chp,"FS: %lu free clusters, %lu sectors per cluster,
         %lu bytes free\r\n",clusters,
         (uint32_t)MMC_FS.csize,clusters *
         (uint32_t)MMC_FS.csize *
         (uint32_t)MMC_SECTOR_SIZE);
    fbuff[0] = 0;
    scan_files(chp, (char *)fbuff);
}
static void cmd_mmctest(BaseSequentialStream *chp, int argc,
        char *argv[]) {
    (void)argv;
    if (argc > 0) {
    chprintf(chp, "mmctest\r\n");
       return;
    }
    if (!fs_ready) {
      chprintf(chp, "File System not mounted\r\n");
      return;
    }
    FIL FDLogFile;
    memset(&FDLogFile, 0, sizeof(FIL));
    FRESULT err_file;
    UINT bw;
    char buffer[buffer_size];
    err_file = f_open(&FDLogFile, "Test.txt", FA_WRITE |
        FA_OPEN_ALWAYS );
```

```
    if (err_file == FR_OK || err_file == FR_EXIST){
        err_file = f_lseek(&FDLogFile, f_size(&FDLogFile));
        if(err_file == FR_OK){
            chsnprintf(buffer,buffer_size,"Aku Jomblo!!!\n\r");
            f_write(&FDLogFile, buffer, strlen(buffer), &bw);
            f_close(&FDLogFile);
            chprintf(chp, "Some text written\r\n");
            return;
        }else{
            chprintf(chp, "Failed to seek file\r\n");
            return;
        }
    }else{
        chprintf(chp, "Cannot Write file\r\n");
        return;
    }
}
static void cmd_mmcadc(BaseSequentialStream *chp, int argc,
        char *argv[]) {
    (void)argv;
    if (argc > 0) {
    chprintf(chp, "mmcadc\r\n");
        return;
    }
    if (!fs_ready) {
      chprintf(chp, "File System not mounted\r\n");
      return;
    }
    Tulis_Adc();
}
static void cmd_dataadc(BaseSequentialStream *chp, int argc,
        char *argv[]) {
  (void) argv;
  if (argc > 0) {
    chprintf(chp, "Usage: dataadc\r\n");
    return;
```

```
  }
{
  chprintf(chp, "adc_co = %4i\r\n",adc_co);
  chprintf(chp, "adc_so2= %4i\r\n",adc_so2);
  chprintf(chp, "adc_co2= %4i\r\n",adc_co2);
  chprintf(chp, "adc_nox= %4i\r\n",adc_nox);
}
}
static void cmd_testsms(BaseSequentialStream *chp, int argc,
        char *argv[]) {
  (void) argv;
  if (argc > 0) {
    chprintf(chp, "Usage: testsms\r\n");
    return;
  }
  Sms_Test();
}
static const ShellCommand commands[] = {
  {"mem", cmd_mem},
  {"threads", cmd_threads},
  {"now", cmd_now},
  {"settime", cmd_settime},
  {"setdate", cmd_setdate},
  {"mmctree", cmd_mmctree},
  {"mmctest", cmd_mmctest},
  {"mmcadc",  cmd_mmcadc },
  {"dataadc", cmd_dataadc},
  {"testsms", cmd_testsms},
  {NULL, NULL}
};
static const ShellConfig shell_cfg = {
  (BaseSequentialStream *)&SDU1,
  commands
};
void Shell_Init(void){
   sduObjectInit(&SDU1);
```

```
    sduStart(&SDU1, &serusbcfg);
    usbDisconnectBus(serusbcfg.usbp);
    chThdSleepMilliseconds(1000);
    usbStart(serusbcfg.usbp, &usbcfg);
    usbConnectBus(serusbcfg.usbp);
    shellInit();
}
void Shell_Run(void){
    if      (!shelltp      &&      (SDU1.config->usbp->state      ==
        USB_ACTIVE))
      shelltp     =     shellCreate(&shell_cfg,     SHELL_WA_SIZE,
        NORMALPRIO);
    else if (chThdTerminatedX(shelltp)) {
      chThdRelease(shelltp);    /* Recovers memory of the previous
        shell.   */
      shelltp = NULL;           /* Triggers spawning of a new shell.
        */
    }
    chThdSleepMilliseconds(1000);
}
```

//shell.h

```
#ifndef TA_SHELL_H
#define TA_SHELL_H
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <stdint.h>
#include "ch.h"
#include "hal.h"
#include "chprintf.h"
#include "memstreams.h"
#include "chstreams.h"
#include "shell.h"
#include "ta_usbcfg.h"
#include "ta_utama.h"
```

```
#include "ta_mmc.h"
#include "ta_uart.h"
#define                                          SHELL_WA_SIZE
        THD_WORKING_AREA_SIZE(4096)
#define                                           TEST_WA_SIZE
        THD_WORKING_AREA_SIZE(256)

void Shell_Init(void);
void Shell_Run(void);
#endif // TA_SHELL_H
```

//uart.c

```
#include "ta_uart.h"

extern struct ds1307_t calendar;
extern uint16_t adc_co,adc_so2,adc_co2,adc_nox;
extern float v_co,v_so2,v_co2,v_nox;
void Uart_Init(void){
   palSetPadMode(GPIOB,11,PAL_MODE_ALTERNATE(7));
   palSetPadMode(GPIOB,10,PAL_MODE_ALTERNATE(7));
   sdStart(&SD3,NULL);
}
void Sms_Text(void){
   Uart_Init();
   chThdSleepMilliseconds(500);
}
void Sms_Test(void){
   chprintf((BaseSequentialStream *)&SD3,"AT+CMGF=1\n");
   chThdSleepMilliseconds(100);
   chprintf((BaseSequentialStream *)&SD3,"AT+CMGS=\"");
   chprintf((BaseSequentialStream *)&SD3,"+6282244105564");
   chprintf((BaseSequentialStream *)&SD3,"\"\n");
   chThdSleepMilliseconds(100);
   chprintf((BaseSequentialStream    *)&SD3,"Hasil    Monitoring
        Gas pada\n");
```

```
    chprintf((BaseSequentialStream *)&SD3,"Tanggal =%2i-%2i-
        %4i                                    pukul
        %2i:%2i\n",calendar.date,calendar.month,calendar.year,cal
        endar.hours,calendar.minutes);
    chprintf((BaseSequentialStream      *)&SD3,"co      =%4i
        %7.1f\n",adc_co,v_co);
    chprintf((BaseSequentialStream          *)&SD3,"so2=%4i
        %7.1f\n",adc_so2,v_so2);
    chprintf((BaseSequentialStream          *)&SD3,"co2=%4i
        %7.1f\n",adc_co2,v_co2);
    chprintf((BaseSequentialStream          *)&SD3,"nox=%4i
        %7.1f\n",adc_nox,v_nox) ;
    chThdSleepMilliseconds(100);

    chSequentialStreamPut((BaseSequentialStream
        *)&SD3,0x1A);
    chSequentialStreamPut((BaseSequentialStream
        *)&SD3,0x0D);
    chSequentialStreamPut((BaseSequentialStream
        *)&SD3,0x0A);
    chprintf((BaseSequentialStream *)&SD3,"\n");
}
```

//uart.h

```
#ifndef TA_UART_H
#define TA_UART_H
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <stdint.h>
#include "ch.h"
#include "hal.h"
#include "chprintf.h"
#include "memstreams.h"
#include "chstreams.h"
#include "ta_rtc.h"
```

```
#include "ta_utama.h"

void Uart_Init(void);
void Sms_Text(void);
void Sms_Test(void);
#endif // TA_UART_H
```

//utama.c

```
#include "ta_utama.h"

extern LcdStream myLCD;
extern adcsample_t adc0,adc1,adc2,adc3;
uint16_t adc_co,adc_so2,adc_co2,adc_nox;
float v_co,v_so2,v_co2,v_nox,v_in,R_S;
extern struct ds1307_t calendar;
extern SerialUSBDriver SDU1;
extern FATFS MMC_FS;
uint8_t udhkirim=0;
static THD_WORKING_AREA(waBlink, 128);
static THD_FUNCTION(Blink, arg)  {

  (void)arg;
  chRegSetThreadName("Blinker");
  while (TRUE) {
    palSetPad(GPIOD, 13);       /* Orange.  */
    chThdSleepMilliseconds(500);
    palClearPad(GPIOD, 13);     /* Orange.  */
    chThdSleepMilliseconds(500);
  }
}
static THD_WORKING_AREA(waADCLCD, 128);
static THD_FUNCTION(ADCLCD, arg)  {

  (void)arg;
  chRegSetThreadName("ADC LCD");
  while (TRUE) {
```

```
    Hasil_Adc();
  }
}
static THD_WORKING_AREA(waRECORD, 1024);
static THD_FUNCTION(RECORD, arg)  {

  (void)arg;
  chRegSetThreadName("RECORD");
  while (TRUE) {
    Tulis_Adc();
    palClearPad(GPIOD, 15);
    chThdSleepMilliseconds(5000);
      if ((v_co >= 25) || (v_so2 >= 20) || (v_co2 >= 600) || (v_nox
          >= 4)){
          if(udhkirim==0){
          Sms_Test();
             udhkirim=1;
          }
      }
      else{
          udhkirim=0;
      }
  }
}
void Run_Init(void){
    chThdCreateStatic(waADCLCD,          sizeof(waADCLCD),
          NORMALPRIO,          ADCLCD,          NULL);
          palSetPadMode(GPIOD,13,PAL_MODE_OUTPUT_PUS
          HPULL);
          palSetPadMode(GPIOD,15,PAL_MODE_OUTPUT_PUS
          HPULL);
    chThdCreateStatic(waBlink, sizeof(waBlink), NORMALPRIO,
          Blink, NULL);
    chThdSleepMilliseconds(1000);
    chThdCreateStatic(waRECORD,          sizeof(waRECORD),
          NORMALPRIO, RECORD, NULL);
```

```
}
void Hasil_Adc(void){
   adc_co =adc0;
   adc_so2=adc1;
   adc_co2=adc2;
   adc_nox=adc3;

   v_co = (0.2279*adc_co)-32.492;
   v_so2 = (0.0018*adc_so2)-1.205;
   v_co2 = ((-0.2446*adc_co2)+926.94);
   v_nox = ((adc3* 9.9)/4095)*0.1;

   Lcd_Cursor(0,0);
   chprintf((BaseSequentialStream      *)&myLCD,"co      =%7.1f
        ppm",v_co);
   Lcd_Cursor(0,1);
   chprintf((BaseSequentialStream         *)&myLCD,"so2=%7.3f
        ppm",v_so2);
   Lcd_Cursor(0,2);
   chprintf((BaseSequentialStream         *)&myLCD,"co2=%7.1f
        ppm",v_co2);
   Lcd_Cursor(0,3);
   chprintf((BaseSequentialStream         *)&myLCD,"nox=%7.2f
        ppm",v_nox) ;
}

void Tulis_Adc(void){
   FIL FDLogFile;
   memset(&FDLogFile, 0, sizeof(FIL));
   FRESULT err_file;
   UINT bw;
   char buffer[64];

   palSetPad(GPIOD, 15);
```

```
    err_file = f_open(&FDLogFile, "Data Monitoring.csv",
        FA_WRITE | FA_OPEN_ALWAYS );
    if (err_file == FR_OK || err_file == FR_EXIST){
       err_file = f_lseek(&FDLogFile, f_size(&FDLogFile));
       if(err_file == FR_OK){

        chsnprintf(buffer,64,"%2i-%2i-
         %4i;%2i:%2i;%4i;%4i;%4i;%4i\r\n",calendar.date,calendar
         .month,calendar.year,calendar.hours,calendar.minutes,adc_
         co,adc_so2,adc_co2,adc_nox);
          f_write(&FDLogFile, buffer, strlen(buffer), &bw);
          f_close(&FDLogFile);
          chprintf((BaseSequentialStream *)&SDU1, "Some text
        written\r\n");
          return;
       }else{
          chprintf((BaseSequentialStream *)&SDU1, "Failed to
        seek file\r\n");
          return;
       }
    }else{
       chprintf((BaseSequentialStream *)&SDU1, "Cannot Write
         file\r\n");
       return;
    }
}
```

//utama.h

```
#ifndef TA_UTAMA_H
#define TA_UTAMA_H
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <stdint.h>
#include "ch.h"
#include "hal.h"
```

```
#include "chprintf.h"
#include "memstreams.h"
#include "chstreams.h"
#include "ta_adc.h"
#include "ta_lcd.h"
#include "ta_rtc.h"
#include "ta_mmc.h"
#include "ta_uart.h"

void Run_Init(void);
void Hasil_Adc(void);
void Tulis_Adc(void);

#endif // TA_UTAMA_H
```

LAMPIRAN C
(DATA KALIBRASI ALAT DETEKTOR PENCEMAR UDARA)

LABORATORIUM PENCEMARAN UDARA DAN PERUBAHAN IKLIM
DEPARTEMEN TEKNIK LINGKUNGAN
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
INSTITUT TEKNOLOGI SEPULUH NOPEMBER

KAMPUS ITS SUKOLILO SURABAYA
TELPON (031)5948886, FAX. (031)5928397

DATA KALIBRASI ALAT DETEKTOR PENCEMAR UDARA

Pengirim : Haryo Arif Wicaksono / 2414031056

Jenis Pencemar : Sulfit (NO_2)

Pengukuran	A D C	Standard (ppm)
1	506	0,13
2	506	0,12
3	662	0,12
4	556	0,13
5	487	0,15
6	468	0,13
7	559	0,12
8	538	0,14
9	475	0,16
10	461	0,09

Surabaya, 20 Juli 2017
Kepala Laboratorium Pencemaran Udara dan Perubahan Iklim
Departemen Teknik Lingkungan FTSP-ITS

Dr.Eng. Arie Dipareza Syafei,ST.MEPM
NIP. 1982 0119 200501 1 001

Gambar C1. Data Kalibrasi Alat Detektor Pencemar Udara

BIODATA PENULIS

Nama Penulis Haryo Arif Wicaksono, dilahirkan di Surabaya, 14 Mei 1996. Riwayat Pendidikan Penulis dimulai dari TK Mutiara Islam dilanjutkan di SDN Kendangsari 1 / 276 Surabaya dan dilanjtukan lagi di SMPN 19 Surabaya dan dilanjutkan lagi di SMAN 1 Surabaya dan pada tahun 2014 masuk di prodi D3 Metrologi & Instrumentasi, Jurusan Teknik Fisika, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember (ITS) Surabaya dengan NRP : 24 14 031 056. Apabila terdapat pertanyaan tentang tugas akhir ini maka dapat menghubungi nomor telpon penulis yaitu : 082232159203, dan dapat juga melalui email penulis yaitu haryoarifw@gmail.com