TUGAS AKHIR TF 145565

RANCANG BANGUN SISTEM *MONITORING* EMISI KADAR GAS SULFUR DIOKSIDA MENGGUNAKAN SENSOR MQ-136 BERBASIS MIKROKONTROLER STM32F4 *DISCOVERY*

Atik Sinawang Wahyuni
NRP. 2414 031 040

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 – TF 145565

RANCANG BANGUN SISTEM *MONITORING* EMISI KADAR GAS SULFUR DIOKSIDA MENGGUNAKAN SENSOR MQ-136 BERBASIS MIKROKONTROLER STM32F4 *DISCOVERY*

Atik Sinawang Wahyuni
NRP. 2414 031 040

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 – **TF 145565**

DESIGN CONSTRUCTION MONITORING EMISSION OF GAS SULFUR DUOXIDE SYSTEM USING SENSOR MQ 136 BASED ON MICROCONTROLLER STM32F4 DISCOVERY

Atik Sinawang Wahyuni
NRP. 2414 031 040

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

STUDY PROGRAM OF D3 INSTRUMENTATION ENGINEERING
DEPARTMENT OF INSTRUMENTATION ENGINEERING
FACULTY OF VOCATION
SEPULUH NOMPEMBER INSTITUTE OF TECHNOLOGY
SURABAYA 2017

RANCANG BANGUN SISTEM *MONITORING* EMISI KADAR GAS SULFUR DIOKSIDA MENGGUNAKAN SENSOR MQ-136 BERBASIS MIKROKONTROLER STM32F4 *DISCOVERY*

TUGAS AKHIR

Oleh :

Atik Sinawang Wahyuni
NRP. 2414 031 040

Surabaya, 25 Juli 2017
Mengetahui / Menyetujui

Dosen Pembimbing

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

Kepala Departemen
Teknik Instrumentasi FV - ITS

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

RANCANG BANGUN SISTEM *MONITORING* EMISI KADAR GAS SULFUR DIOKSIDA MENGGUNAKAN SENSOR MQ-136 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 :
Atik Sinawang Wahyuni
NRP. 2414 031 040

Disetujui oleh Tim Penguji Tugas Akhir :

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

SURABAYA
25 JULI 2017

RANCANG BANGUN SISTEM *MONITORING* EMISI KADAR GAS SULFUR DIOKSIDA MENGGUNAKAN SENSOR MQ-136 BERBASIS MIKROKONTROLER STM32F4 *DISCOVERY*

Nama Mahasiswa	**: Atik Sinawang Wahyuni**
NRP	**: 2414 031 040**
Program Studi	**: D III Teknik Instrumentasi**
Departemen	**: Teknik Teknik Instrumentasi Fakultas Vokasi-ITS**
Dosen Pembimbing	**: Dr. Ir. Ali Musyafa', M.Sc.**

Abstrak

Gas Sulfur merupakan salah satu polutan berbahaya bagi manusia. Akibat utama polutan SO_2 terhadap manusia adalah terjadinya iritasi pada sistem pernafasan. Terutama kesehatan bagi usia lanjut dan penderita yang mengalami penyakit kronis pada sistem pernafasan dan kardiovaskular yang sangat sensitif jika kontak dengan SO_2. Sehingga dalam tugas akhir ini perlu adanya sebuah rancang bangun sistem *monitoring* kadar gas sulfur dioksida di udara. Sistem *monitoring* ini menggunakan buah *sensor* MQ 136. Untuk *monitoring* ini mempunyai *range* sebesar 0,04 hingga 1,04 dan *error* sebesar 5,5%, Hasil pengukuran (x) dari alat *monitoring* ini sebesar 0,52±0,105 ppm dengan menggunakan faktor cakupan sebesar 2,196 dan *confidence level* (CL) 95%. Alat *monitoring* ini dilengkapi dengan teknologi SMS *Gateway* guna untuk memberitahukan periangatan dini yang terhubung ke mobile *user*. Selain itu alat *monitoring* ini menggunakan *Real Time Clock* sehingga penyimpanan data logger sesuai dengan waktu yang sebenarnya.

Kata kunci: Sulfur Dioksida, *Monitoring* gas, *Sensor* MQ 136, SMS *Gateway*, *Data Logger*.

DESIGN CONSTRUCTION MONITORING EMISSION OF GAS SULFUR DIOXIDE SYSTEM USING SENSOR MQ 136 BASED ON MICROCONTROLLER STM32F4 DISCOVERY

Name of Student	**: Atik Sinawang Wahyuni**
NRP	**: 2414 031 040**
Program Study	**: D III *Instrumentation Engineering***
Department	**: *Instrumentation Engineering Faculty Of Vocation*-ITS**
Advisor Lecturer	**: Dr. Ir. Ali Musyafa', M.Sc.**

ABSTRACT

Sulfur Diooxide gas is one of the most harmful pollutants for humans. The main effect of the SO_2 pollutants on humans is the irritation of the respiratory system. Especially health for the elderly and patients with chronic illness in the respiratory and cardiovascular system are very sensitive even if contact with SO_2. So in this final task need a design construction of monitoring system of gas content of sulfur dioxide in air. This monitoring system uses the MQ 136 sensor. For this monitoring has a range of 0.04 to 1.04 and error of 5.5%, The measurement result (x) of this monitoring tool is 0,52±0,105 ppm using the coverage factor of 2,196 and a confidence level (CL) of 95%. This monitoring tool is equipped with SMS Gateway technology in order to notify the early flutter connected to the mobile user. In addition, this monitoring tool uses Real Time Clock so that the data storage logger in accordance with the actual time.

Keywords: ***Sulfur Dioxide, Gas Monitoring, Sensor MQ 136, SMS Gateway, Data Logger.***

KATA PENGANTAR

Puji syukur kehadirat Allah SWT atas rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan laporan Tugas Akhir yang berjudul **"RANCANG BANGUN SISTEM MONITORING EMISI KADAR GAS SULFUR DIOKSIDA MENGGUNAKAN SENSOR MQ-136 BERBASIS MIKROKONTROLER STM32F4 *DISCOVERY*"** dengan tepat waktu. Terselesaikannya laporan ini juga tak luput dari dukungan dan peran dari orangtua dan keluarga besar serta berbagai pihak. Untuk itulah dalam kesempatan ini penulis mengucapkan terima kasih yang sebesar-besarnya kepada:

1. Bapak Dr. Ir. Purwadi Agus Darwito, M.Sc selaku Ketua Departemen Teknik Instrumentasi.
2. Bapak Dr. Ir. Ali Musyafa', M.Sc selaku pembimbing Tugas Akhir yang telah membina dengan baik dan sabar.
3. Seluruh Asisten Laboratorium *Microprocessor and Microcontroller*, *Workshop Instrumentasi*, Laboratorium Instrumentasi dan Kontrol, Laboratorium Pengukuran Fisis, dan Sahabat Mikro serta Sahabat Zelena yang telah membantu dalam pengerjaan Tugas Akhir penulis.
4. Monitoring *Team*, sahabat terbaik yang penulis cintai (Syahril, Mufida, dan Haryo) yang telah bersama-sama berjuang dalam pengerjaan Tugas Akhir ini hingga selesai.
5. Teman-teman Teknik Instrumentasi dan S1 Teknik Fisika angkatan 2014 FTI-ITS.
6. Serta semua pihak yang tidak dapat disebutkan satu persatu.

Penulis menyadari bahwa laporan ini masih kurang sempurna. Oleh karena itu penulis menerima segala masukan baik berupa saran, kritik, dan segala bentuk tegur sapa demi kesempurnaan lapiran ini.

Demikian laporan Tugas Akhir ini penulis persembahkan dengan harapan dapat bermanfaat dalam akademik baik bagi penulis sendiri maupun bagi pembaca.

Surabaya, 18 Juli 2017

Penulis.

DAFTAR ISI

Hal

DAFTAR GAMBAR

DAFTAR TABEL

Hal

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi semakin berkembang dan mempermudah manusia dalam beraktifitas, namun dalam perkembangan teknologi dalam bidang apapun terkadang tidak diimbangi dengan minimalisasi dampak negatifnya. Polusi udara dapat di karenakan dari berbagai sebab di antaranya adalah reaksi kimia dari limbah yang ada di bumi, sisa pembuangan kendaraaan bermotor, gunung meletus, sisa pembakaran yang ada pada pabrik dan industri serta kebakaran hutan **[1]**. Indonesia, permasalahan lingkungan hidup seolah-olah seperti dibiarkan menggelembung sejalan dengan intensitas pertumbuhan industri, walaupun industrialisasi itu sendiri sedang menjadi prioritas dalam pembangunan menyebabkan kualitas udara menurun. Pencemaran udara biasanya terjadi di kota-kota besar dan juga daerah padat industri yang menghasilkan gas-gas yang mengandung zat di atas batas kewajaran. Gas-gas pencemar udara utama terdiri dari CO, CO_2, NO, NO_2, SO dan SO_2 **[2]**.

Lebih dari 90 persen penduduk dunia menghirup udara dengan kualitas buruk. Data itu dilansir organisasi kesehatan dunia WHO, pada Selasa (27/9/2016). Polusi udara dianggap sebagai salah satu pembunuh terbesar, mencapai 6 juta orang per tahun. The Guardian menulis, di Indonesia, korban tewas karena polusi udara mencapai 61 ribu orang, atau rata-rata 25 orang meninggal per 100 ribu kapita. Laporan WHO menunjukkan 92 persen warga dunia hidup di tempat dimana kualitas udara di bawah ambang batas sehat. Laporan itu didasarkan pada data yang dikumpulkan dari lebih 3,000 kota di dunia, melibatkan belasan ilmuwan dari sejumlah institusi internasional. Sumber polusi udara diantaranya moda transportasi yang tidak efisien, asap rumah tangga dan pembakaran sampah, gunung meletus hingga aktivitas industri**[3]**.

Salah satu polutan yang sangat berbahaya yaitu sulfur dioksida. Semua bahan bakar yang digunakan oleh manusia

mengandung sulfur (belerang) pada kadar tertentu. Kayu memiliki kandungan terendah sekitar 0,1 % atau kurang, batu bara antara 0,5 samapai 3 %, sedangkan minyak kandungan sulfurnya lebih besar daripada kayu, namun lebih kecil dibandingkan batubara. Sulfur dioksida merupakan gas yang tidak berwarna, tidak flammable, maupun tidak explosive. Gas ini memiliki kelarutan dalam air sebesar 11,3 g/100 ml pada suhu 20°C, berat molekulnya 64,06 dan dua kali lebih berat daripada udara.**[4]**

Sebagai pencemar udara SO_2 diperkirakan memiliki waktu tinggal dalam udara antara 2 sampai 4 hari, dan dalam waktu tinggal tersebut, SO_2 dapat ditransportasikan sejauh 1000 km, sehingga dapat dikatakan SO_2 relatif stabil dalam atmosfer. Sehingga masalah pencemaran SO_2 menjadi masalah internasional.

Surabaya menduduki peringkat ketiga kota di kawasan Asia yang memiliki polusi udara tertinggi. Surabaya menduduki peringkat ketiga setelah Bangkok dan Jakarta sebagai kota di kawasan Asia yang polusi udaranya paling buruk **[5]**. Pemantauan kondisi tingkat polusi udara di Surabaya, Pemerintah Surabaya menyediakan sarana prasarana berupa ISPU (Indeks Standar Pencemaran Udara) sejumlah 6 unit. Akan tetapi ISPU yang masih aktif hingga saat ini hanya tersedia 2 unit yang terletak pada jalur MERR dan kawasan Hotel Sahid. Sedangkan 4 unit yang lain sudah tidak berfungsi lagi.**[5]**

Maka dari itu dirancang suatu sistem yang difungsikan untuk melakukan pemantauan adanya gas SO_2. Pemantauan gas SO_2 dilakukan untuk mengetahui adanya emisi gas SO_2 pada suatu area tertentu dari proses yang menimbulkan emisi gas SO_2. Sehingga, tugas akhir ini diberi judul "Rancang Bangun Sistem Monitoring Emisi Kadar Gas Sulfur Dioksida Menggunakan Sensor MQ 136 Berbasis Mikrokontroller *STM32F4 Discovery*".

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 emisi kadar gas sulfur dioksida menggunakan sensor MQ 136 berbasis mikrokontroller *STM32F4 Discovery*?
b. Bagaimana mengetahui karakteristik statik dari alat monitoring gas sulfur dioksida (SO_2)
c. Bagaimana penggunaan data hasil montoring konsentrasi sulfur dioksida (SO_2) yang dihasilkan dari emisi gas buang?

1.3 Tujuan

Tujuan yang dicapai dalam tugas akhir ini adalah sebagai berikut:

a. Merancang suatu sistem monitoring emisi kadar gas sulfur dioksida menggunakan sensor MQ 136 berbasis mikrokontroller *STM32F4 Discovery*
b. Mengetahui karakteristik statik dari alat monitoring gas sulfur dioksida (SO_2)
c. Mengetahui penggunaan data hasil montoring konsentrasi sulfur dioksida (SO_2) yang dihasilkan dari 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. Perancangan sistem monitoring gas sulfur dioksida menggunakan sebuah mikrokontroler *STM32F4 Discovery*.
b. Perancangan sistem monitoring has sulfur dioksida menggunakan sensor MQ 136.
c. Perancangan sistem monitoring gas sulfur diooksida 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. Penyusunan 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 kadar gas sulfur dioksida dengan menggunakan mikrokontroler *STM32F4 Discovery* dan dapat dijadikan sebagai perancangan alat monitor udara pada lingkungan.

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 zat kimia sulfur dioksida, sensor gas, 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.

BAB II
TINJAUAN PUSTAKA

2.1 Sulfur Dioksida (SO_2)

2.1.1 Sifat-sifat Sulfur Dioksida (SO_2)

Belerang oksida atau sering ditulis dengan SOx terdiri atas gas Sulfur Dioksida (SO_2) dan gas Sulfur Trioksida (SO_3) yang keduanya mempunyai sifat berbeda. Pada dasarnya, semua Sulfur yang memasuki atmosfer dirubah dalam bentuk SO_2 dan hanya 1%-2% saja sebagai SO_3. Gas SO_2 berbau tajam dan tidak mudah terbakar. Cairan SO_2 melarutkan banyak senyawaan organik dan anorganik dan digunakan sebagai pelarut dalam pembuatan reaksi. Cairannya tidak melakukan pengionan-diri dan hantarannya terutama merupakan cermin bagi kemurniannya.

Sulfur dioksida mempunyai pasangan-pasangan menyendiri dan dapat bertindak sebagai basa lewis. Meskipun demikian, ia juga bertindak sebagai asam Lewis menghasilkan kompleks, misalnya dengan amina seperti Me_3HSO_2, dan dengan kompleks logam transisi yang kaya elektron. Dalam senyawa kristal SbF_5SO, yang menarik karena penggunaan SO_2 sebagai pelarut bagi sistem super-asam. SO_2 sangat larut dalam air; suatu larutan yang memiliki sifat asam, telah lama dikenal sebagai larutan asam sulfit, H_2SO_3. Gas SO_2 diudara bereaksi dengan uap air atau larut pada tetesan air membentuk H_2SO_4 yang merupakan komponen utama dari hujan asam.

Gas SO2 juga dapat membentuk garam sulfat apabila bertemu dengan oksida logam, yaitu melalui proses kimiawi berikut ini :

$$4MgO + 4SO_2 \rightarrow 3MgSO_4 + MgS$$

Udara yang mengadung uap air akan bereaksi dengan gas SO_2 sehingga membentuk asam sulfit :

$$SO_2 + H_2O \rightarrow H_2SO_3 \text{ (asam sulfit)}$$

Udara yang mengandung uap air juga akan bereaksi dengan gas SO_3 membentuk asam sulfat :

$SO_3 + H_2O \rightarrow H_2SO_3$ (asam sulfat) **[6]**

2.2.2 Sumber-sumber Sulfur dioksida (SO_2)

Pencemaran SO_2 diudara berasal dari sumber alamiah maupun sumber buatan. Sumber alamiah adalah gunung-gunung berapi, pembusukan bahan organik oleh mikroba dan dan reduksi sulfat secara biologis. Proses pembusukan akan menghasilkan H_2S yang akan cepat berubah menjadi SO_2.

Sumber SO_2 buatan adalah pembakaran bahan bakar minyak, gas, dan terutama batubara yang mengandung sulfur tinggi. Sumber-sumber buatan ini diperkirakan memberi kontribusi sebanyak sepertiganya saja dari seluruh SO_2 atmosfir/tahun. Akan tetapi, karena hampir seluruhnya berasal dari buangan industri, maka hal ini bertambah di kemudian hari, maka dalam waktu singkat sumer-sumber ini akan dapat memproduksi lebih banyak SO_2 daripada sumber alamiah.

Gas SO_2 diproduksi terutama oleh insinerator yang menggunakan bahan bakar fosil seperti batu bara dan minyak bumi. SO_2 diemisikan oleh pabrik kimia, pabrik pemroses besi dan baja, pembuatan semen, pabrik batu bata, industri keramik, pembuatan kaca dan pelepasan asap buangan.**[7]**

2.2.3 Dampak Pencemaran Sulfur Dioksida (SO_2)

Akibat utama polutan SO_2 terhadap manusia adalah terjadinya iritasi pada sistem pernafasan. Dalam Tabel 2.1 ditunjukkan konsentrasi SO_2 yang berpengaruh terhadap manusia. Beberapa hasil penelitian menunjukkan bahwa iritasi pada tenggorokan terjadi pada konsentrasi SO_2 sebesar 5 ppm atau lebih, bahkan pada beberapa individu yang sensitif, iritasi terjadi pada konsentrasi 1-2 ppm. SO_2 dianggap polutan yang berbahaya bagi kesehatan terutama terhadap manusia usia lanjut dan penderita yang mengalami penyakit kronis pada sistem pernafasan dan kardiovaskular. Individu dengan gejala tersebut sangat sensitif jika kontak dengan SO_2 walaupun dengan konsentrasi yang relatif rendah, misalnya 0,2 ppm atau lebih.**[8]**

Selain pengaruhnya terhadap kesehatan, sulfur dioksida juga berpengaruh terhadap tanaman dan hewan. Pengaruh SO_2 terhadap hewan sangat menyerupai efek SO_2 pada manusia. Efek SO_2 terhadap tumbuhan tampak terutama pada daun yang menjadi putih atau terjadi nekrosis, daun yang hijau dapat berubah menjadi kuning, ataupun terjadi bercak-bercak putih. Pengaruh pada daun ini terjadi terutama di siang hari sewaktu stomata daun sedang terbuka. Apabila yang terpapar SO_2 itu adalah sayuran, maka perubahan pada warna daun tentunya sangat mempengaruhi harga jual sayuran.

Harta benda dapat juga terpengaruh oleh SO_2. Gedung-gedung yang mempunyai arti sejarah, patung-patung bernilai seni dapat rusak karena SO_2 mudah menjadi H_2SO_4 yang sangat korosif.**[9]**

Tabel 2.1 Pengaruh SO_2 terhadap manusia

Konsentrasi ppm	Pengaruh
3-5	- Jumlah minimum yang dapat dideteksi dari baunya
8-12	- Jumlah minimum yang segera mengakibatkan iritasi pada tenggorokan
20	- Jumlah minimum yang mengakibatkan iritasi pada mata - Jumlah minimum yang segera mengakibatkan batuk - Jumlah maksimum yang diperkenankan untuk kontak dalam

2.2 Gas Ambien

Kualitas udara ambien merupakan tahap awal untuk memahami dampak negatif cemaran udara terhadap lingkungan. Kualitas udara ambien ditentukan oleh kuantitas emisi cemaran dari sumber cemaran dan proses transportasi, konversi dan penghilangan cemaran di atmosfer. Baku mutu kualitas udara lingkungan/ambien ditetapkan untuk cemaran O_3 (ozon), CO, NOx, SO_2, hidrokarbon non metana dan partikulat.**[10]**

Tabel 2.2 Baku Mutu Udara Ambien Indonesia

Parameter	Waktu Pengukuran	Baku Mutu	Metoda Analisa
SO2	24 jam	260 μg/m^3 (0,10 ppm)	Para-rosanilin
CO	8 jam	2260 μg/m^3 (20 ppm)	Non Dispersive Infrared (NDIR)
Nox	24 jam	92,5 μg/m^3 (0,05 ppm)	Saltzman
Oksidan	1 jam	200 μg/m^3 (0,10 ppm)	Chemiluminescent
Debu	24 jam	0,26 μg/m^3	Gravimetri
Timah Hitam	24 jam	0,06 μg/m^3	Gravimetrik Absorbsi Atom

2.3 Sensor MQ 136

Sensor MQ-136 adalah suatu komponen semikonduktor yang berfungsi sebagai pengindera bau gas tin oksida (SnO_2). Sensor gas MQ136 memiliki sensitivitas tinggi terhadap SO_2, juga bisa digunakan untuk mendeteksi uap lain yang mengandung Sulfur. Ini memiliki sensitivitas rendah terhadap gas yang mudah terbakar normal, dengan biaya rendah dan sesuai untuk aplikasi yang berbeda.**[11]**

Gambar 2.1 Sensor MQ 136

Sensor ini membutuhkan tegangan *input* sebesar 5V. Pada sensor ini terdapat nilai resistansi sensor (Rs) yang dapat berubah

bila terkena gas dan juga sebuah pemanas yang digunakan sebagai pembersihan ruangan sensor dari kontaminasi udara luar. Sensor ini memerlukan rangkaian sederhana serta memerlukan tegangan pemanas (power heater) sebesar 5V, resistansi beban (load resistance). Output sensor berupa data analog.

Tabel 2.3 Spesifikasi Sensor MQ 136

Model No.			MQ136
Sensor Type			Semiconductor
Standard Encapsulation			Bakelite (Black Bakelite)
Detection Gas			SO_2
Concentration			1-200ppm (SO_2)
Circuit	Loop Voltage	Vc	≤24V DC
	Heater	VH	5.0V±0.2V ACorDC
	Load Resistance	RL	Adjustable
Character	Heater Resistance	RH	31Ω±3Ω（Room Tem.）
	Heater consumption	PH	≤900mW
	Sensing Resistance	Rs	2KΩ-20KΩ(in 50ppm SO2)
	Sensitivity	S	Rs(in air)/Rs(50ppm SO2)≥3
	Output Voltage	ΔVs	≥ 0.5 V(in 500 ppm SO_2)
	Slope	α	≤0.6(R100ppm/R50ppm SO2)
Condition	Tem. Humidity		20℃±2℃；65%±5%RH
	Standard test circuit		Vc: 5.0V±0.1V； VH: 5.0V±0.1V
	Preheat time		Over 48 hours

Sensor gas terdiri dari elemen sensor, dasar sensor dan tudung sensor. Elemen sensor terdiri dari bahan sensor dan bahan pemanas untuk memanaskan elemen. Elemen sensor menggunakan bahan-bahan seperti timah (IV) oksida SnO2, wolfram (VI) oksida WO3, dan lain-lain, tergantung pada gas yang hendak dideteksi.

Bila suatu kristal oksida logam seperti SnO2 dipanaskan pada suhu tinggi tertentu di udara, oksigen akan teradsorpsi pada permukaan kristal dengan muatan negatif . Elektron-elektron donor pada permukaan kristal ditransfer ke oksigen teradsorpsi, sehingga menghasilkan suatu lapisan ruang bermuatan positip. Akibatnya potensial permukaan terbentuk, yang akan menghambat aliran elektron. Di dalam sensor, arus listrik mengalir melalui bagian-bagian penghubung (batas butir) kristal-kristal mikro SnO2. Pada batas-batas antar butir, oksigen yang teradsorpsi membentuk penghalang potensial yang menghambat muatan bebas bergerak. Tahanan listrik sensor disebabkan oleh penghalang potensial ini.

Dalam lingkungan adanya gas pereduksi, kerapatan oksigen teradsorpsi bermuatan negatif pada permukaan semikonduktor sensor menjadi berkurang, sehingga ketinggian penghalang pada batas antar butir berkurang. Ketinggian penghalang yang berkurang menyebabkan berkurangnya tahanan sensor butir dalam lingkungan gas.

Hubungan antar tahanan sensor dan konsentrasi gas pereduksi pada suatu rentang konsentrasi gas dapat dinyatakan dengan persamaan berikut :

Rs = A [C] -a , dengan :
Rs = tahanan listrik sensor
A = konstanta
[C] = konsentrasi gas
A = gradien kurva Rs

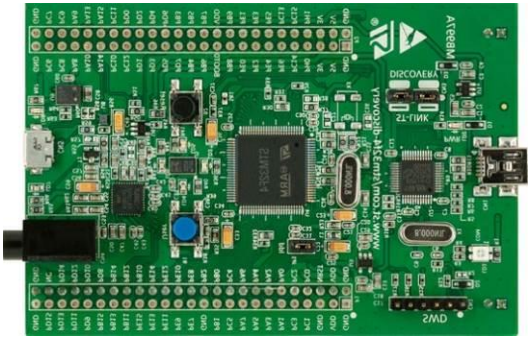
2.4 Mikrokontroler

Sebuah komputer mikro memiliki tiga komponen utama yaitu, unit pengolah pusat *Central Processing Unit* (CPU),

memori dan sistem *input/output* (I/O) untuk dihubungkan dengan perangkat luar. *Central Processing Unit* (CPU), yang mengatur sistem kerja komputer mikro, dibangun oleh sebuah mikroprosesor. Memori terdiri atas EEPROM untuk menyimpan program dan RAM untuk menyimpan data. Sistem I/O bisa dihubungkan dengan perangkat luar misalnya sebuah *keyboard* dan sebuah monitor, bergantung pada aplikasinya. Apabila *Central Processing Unit* (CPU), memori dan sistem I/O dibuat dalam sebuah *chip* semikonduktor, maka inilah yang dinamakan mikrokontroler **[12]**.

2.5 Mikrokontroler 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 embedded sistem dimana membutuhkan daya dan harga yang rendah. 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.**[13]**



Gambar 2.2 STM32F407 *Discovery* **[14]**

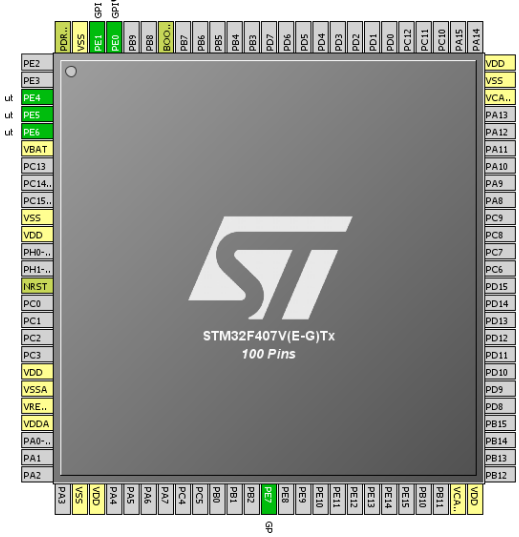
Fitur utama

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.

Dalam mikrokontroler STM32F4 *Discovery* ini terdapat konfigurasi tesendiri agar dapat digunakan dengan baik dan benar. Konfigurasi tersebut digunakan untuk pengontrolan suatu instrumen yang digunakan. Selain itu, dengan konfigurasi kita dapat menentukan di *port* manakah kita memasukkan kodingan yang telah disusun dalam *software* tertentu yang ada pada komputer atau laptop. Di bawah ini merupakan konfigurasi *pin* STM32F4 *Discovery*:



Gambar 2.3 Konfigurasi pin STM32F4 *Discovery*

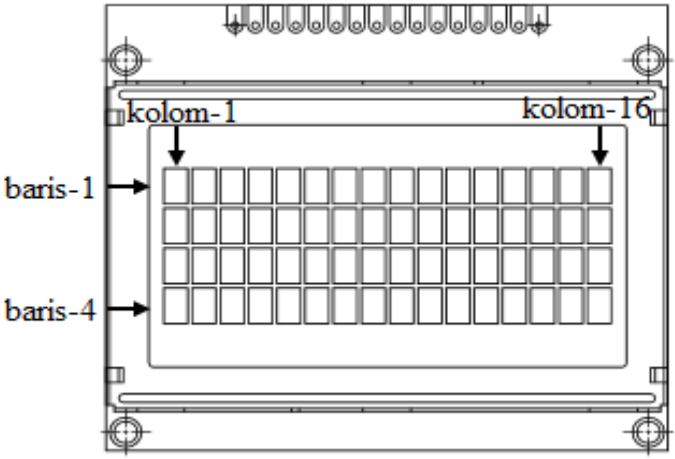
2.6 *Liquid Crystal Display* (LCD)

Liquid Crystal Display (LCD) adalah suatu alat untuk *display* berbagai *character*. LCD ini mempunyai beberapa ukuran mengikuti bilangan *character* seperti 16x4. 16x4 *character* bermakna LCD tersebut mempunyai 16 kolom dan 4 baris. LCD ini mempunyai 16 *pin*.

Gambar 2.4 LCD *Character* 16x4

LCD karakter dalam pengendaliannya cenderung lebih mudah dibandingkan dengan LCD grafik. Namun ada kesamaan diantara keduanya, yaitu inisialisasi. Inisialisasi adalah prosedur awal yang perlu dilakukan dan dikondisikan kepada LCD agar LCD dapat bekerja dengan baik. Hal ini sangat penting ditentukan dalam proses inisialisasi adalah jenis *interface* (antar muka) antara yang dapat digunakan dalam pengendalian LCD karakter **[15]**.

Untuk dapat mengendalikan LCD karakter dengan baik, tentu perlu koneksi yang benar. Koneksi yang benar dapat diwujudkan dengan cara mengetahui *pins* antarmuka yang dimiliki oleh LCD karakter tersebut yaitu seperti pada gambar 2.9 dibawah ini.



Gambar 2.5 Penunjukkan Kolom dan Baris pada LCD 16x4[17]

Keterangan:

1. *Pin* 1 dihubungkan ke *Ground*.
2. *Pin* 2 dihubungkan ke Vcc (+5V).
3. *Pin* 3 dihubungkan ke bagian tengah potensiometer sebagai pengatur kontras.
4. *Pin* 4 untuk *Register Selection* (RS). Jika diberi nilai logika 1 (*High*) = *display* data dan jika diberi nilai logika 0 (*Low*) = *Write Operational*.
5. *Pin* 5 digunakan untuk mengatur fungsi LCD. Jika di-*set* ke logika 1 (*high*, +5V) maka LCD berfungsi untuk membaca data, jika *pin* ini di-*set* ke logika 0 (low, 0V) akan berfungsi untuk menulis data.
6. *Pin* 6 adalah terminal *enable* (*Enable Signal*). Berlogika 1 setiap kali pengiriman atau pembaca data.
7. *Pin* 7 – 14 adalah saluran dua arah (*bi-directional*) data 8 *bit* dan 4 *bit bus data* (untuk 4 *bit pin* data yang digunakan *pin* 11 – 14).
8. *Pin* 15 dan 16 adalah tegangan untuk menyalakan LCD **[16]**.

2.7 RTC DS3231

DS3231 adalah I2C sangat akurat *real-time clock* (RTC) dengan temperature compensated terintegrasi osilator kristal (TCXO) dan kristal. Perangkat menggabungkan input baterai, dan memelihara ketepatan waktu yang akurat ketika listrik utama ke perangkat terganggu. RTC mempertahankan detik, menit, jam, hari, tanggal,bulan, dan informasi tahun. Tanggal pada akhir bulan secara otomatis disesuaikan dengan bulan dengan lebih sedikit dari 31 hari, termasuk koreksi untuk tahun kabisat. Jam beroperasi baik dalam 24 jam atau 12 jam dengan format AM indikator / PM. Diprogram dua waktu-of-hari alarm dan output gelombang persegi programmable yang disediakan. Alamat dan data yang ditransfer serial melalui dua arah bus I2C.
Referensi tegangan kompensasi suhu presisi dan sirkuit komparator memonitor status VCC yang berfungsi untuk mendeteksi kegagalan daya, memberikan keluaran reset, dan digunakan secara otomatis beralih ke persediaan cadangan bila diperlukan. Selain itu, pin RST dipantau sebagai masukan tombol tekan untuk menghasilkan reset secara eksternal. **[18]**

Gambar 2.6 RTC DS 3231

2.8 *Micro SD Sield* Modul

Modul (*MicroSD Card Adapter*) adalah modul pembaca kartu *MicroSD*, dan antarmuka SPI melalui *driver* sistem *file*, sistem mikrokontroler untuk melengkapi kartu *MicroSD* membaca dan menulis file. Pengguna Arduino bisa langsung

menggunakan Arduino *IDE* hadir dengan kartu SD untuk melengkapi inisialisasi kartu perpustakaan dan baca tulis. Terdapat enam pin (GND, VCC, MISO, MOSI, SCK, CS), GND ke ground, VCC adalah *power supply*, MISO, MOSI, SCK adalah bus SPI, CS adalah *chip select signal pin*.**[19]**

Gambar 2.7 *Module SD Card*

2.9 *Software* ChibiOS/RT

ChibiOS/RT merupakan salah satu dari sekian banyaknya RTOS yang ada pada saat ini. Kata *chibi* yang ada pada ChibiOS/RT merupakan bahasa Jepang yang mempunyai arti kecil. ChibiOS/RT merupakan RTOS yang menggunakan bahasa pemrograman C dan C++. Untuk saat ini ChibiOS/RT sudah bisa di-*porting* ke dalam beberapa arsitektur mikrokontroler di antaranya jenis AVR, ARM7, ARM9, ARMMCx (ARMv6/7-M), STM, MSP430 dan PowerPC.

Selain itu juga bisa digunakan sebagai *simulator* pada OS Linux dan Windows x86. Sedangkan terdapat beberapa *compiler* yang sudah *support* dengan ChibiOS/RT di antaranya GCC, IAR, Keil RVCT, Cosmic dan Raisonance (RC).

ChibiOS/RT sendiri merupakan perangkat lunak gratis dengan lisensi GNU *general public license* 3 atau GPL3. Di mana merupakan perangkat lunak ini bisa diperbanyak atau disebarluaskan kepada umum tetapi tidak boleh mengklaim tentang hak cipta dari ChibiOS/RT dan tidak disarankan mengubah konfigurasi dan fungsi yang sudah ada (kecuali konfigurasi untuk mikrokontroler yang digunakan) untuk lebih jelasnya dapat dilihat pada *homepage* ChibiOS/RT

http://www.chibios.org. Berikut merupakan fitur-fitur pada ChibiOS/RT antara lain:

1. Perangkat lunak gratis dengan lisensi GPL3.
2. Dirancang untuk aplikasi RTOS.
3. *Portable.*
4. *Preemptive scheduling.*
5. 256 tingkat prioritas, di mana bisa terdapat dua atau lebih *task* dengan prioritas yang sama.
6. *Round robin scheduling* untuk *task* dengan prioritas yang sama.
7. Terdapat *task/thread, virtual timers, semaphores, mutexes, condvars, event flags, messages, mailboxes,* I/O *queues.*
8. Perancangan bisa dilakukan pada PC dengan Windows atau Linux.
9. Terdapat fungsi opsional *heap allocator subsystem* dan *memory pools allocator subsystem.*
10. *Blocking* dan *non-blocking* jalur I/O dengan kemampuan *timeout* dan pembuat *event*.
11. Hampir semua tertulis dalam bahasa C dengan sedikit bahasa *assembler* untuk *porting*.
12. Terdapat *hardware abstraction layer* (HAL) yang mendukung untuk berbagai macam peralatan seperti, serial, ADC, CAN I2C, MAC, MMC, PWM, SPI, UART, uIP, lwIP, dan FatFs.

Sedangkan untuk menggunakan ChibiOS/RT diperlukan mikrokontroler yang memenuhi spesifikasi minimum sebagai berikut:

1. Arsitektur minimum CPU dengan 8-bits.
2. Mendukung untuk bahasa standar C89 dan C99.
3. Mendukung untuk maskable interrupt sources.
4. Memiliki RAM minimal sebesar 2 KB.
5. Memiliki memori untuk program sebesar 16 KB **[20]**.

2.10 *Qt Creator*

QT Creator merupakan *cross-platdorm* IDE (*Integrated Development Environment*) yang lengkap untuk pengembangan aplikasi dengan target berbagai *platform desktop* dan berbagai *platform mobile*. QT Creator dapat diinstall pada Linux, OS X dan Micorosoft Windows.

QT Creator merupakan IDE yang menyediakan *tools* untuk mendisain dan mengembangkan aplikasi menggunakan *framework* aplikasi Qt. QT merancang tampilan dan mengembangkan aplikasi sekali kemudian menyebarkan aplikasi tersebut ke berbagai *platform desktop* dan *platform mobile*. QT Creator menyediakan *tool-tool* menyeluruh dalam mengembangkan aplikasi dimulai dari memulai project dan menyebarkan aplikasi ke berbagai target *platform*.

Salah satu kelebihan QT Creator adalah memungkinkan sebuah team pengembang aplikasi bekerja sama mengembangkan aplikasi dari berbagai platform dengan menggunakan *tool-tool* dan debugging yang sama.**[21]**

Qt Creator mempunyai sebuah *code editor* and mengintegrasikan Nokia's Qt Designer untuk mendesain dan membangun aplikasi GUI dari Qt widgets. Karena Qt Creator adalah sebuah *Integrated Development Enviroment* (IDE), maka Qt Creator memisahkan antara *text editor* untuk *build* dan *editor* untuk menjalankan (*run*) aplikasi-aplikasi. Qt Creator bukan hanya bisa membaca *text file* biasa, akan tetapi juga bisa membaca file C++ dan bahasa QML.

Keunggulan dari Code Editor sebagai berikut:

1. Dapat menulis *code* dengan format yang benar.
2. Mengantisipasikan apa yang akan *programer* tulis dan *code* yang komplit.
3. Menampilkan baris-baris yang *error* dan pesan-pesan *warning*.
4. Memandu *programer* secara semantik untuk menulis *classes, functions*, dan *symbols*.
5. Menyediakan fasilitas bantuan *context-sensitive* pada *classes, functions*, dan *symbols.*

6. Me-*rename* simbol-simbol dengan langkah *intelligent*, sehingga simbol-simbol yang lain dengan nama yang sama tidak ter-*rename*.
7. Menampilkan lokasi *function*, *class* yang dideklarasikan atau yang dipanggil.**[22]**

2.11 Modem GSM Wavecom

Wavecom adalah pabrikan asal Perancis (bermarkas di kota Issy les Moulineaux, Perancis) yaitu Wavecom SA yang berdiri sejak 1993 bermula sebagai biro konsultan teknologi dan sistim jaringan nirkabel GSM, dan pada 1996 Wavecom mulai membuat desain daripada modul wireless GSM pertamanya dan diresmikan pada 1997, bentuk modul GSM pertama berbasis GSM dan pengkodean khusus yang disebut AT Command. Sulit mencari referensi module tipe apa yang pertama dibuat oleh Wavecom SA.

Modem Wavecom Fastrack ini di Indonesia cukup dikenal digunakan pada industri bisnis rumahan dan bahkan skala besar – mulai dari fungsi untuk kirim SMS massal hingga fungsi sebagai penggerak perangkat elektronik. Beberapa fungsi kegunaan modem ini di masyarakat adalah antara lain:

1. SMS Broadcast application
2. SMS Quiz application
3. SMS Polling
4. SMS auto-reply
5. M2M integration
6. Aplikasi Server Pulsa
7. Telemetri
8. Payment Point Data
9. PPOB
10. Dsb

Kelebihan dari menggunakan Modem Wavecom Fastrack ketimbang Modem GSM/HP:

1. Wavecom jauh lebih stabil dibanding Modem GSM/HP
2. Wavecom tidak gampang panas dibanding Modem GSM/HP

3. Pengiriman SMS yang lebih cepat dibanding Modem GSM/HP (1000 s/d 1200 SMS per jam)
4. Support AT Command, bisa cek sisa pulsa, cek point, cek pemakaian terakhir, dll
5. Tidak semua Modem GSM/HP support AT Command
6. Tidak memakai baterai sehingga lebih praktis digunakan
7. Dan masih banyak lainnya**[23]**

Gambar 2.8 Modem Wavecom 1306B

2.12 SMS Gateway

SMS (*Short Message Service*) merupakan suatu teknologi yang memungkinkan untuk mengirim dan menerima pesan antar pengguna *mobile phone*. Sms *Gateway* adalah suatu *platform* yang menyediakan mekanisme untuk EUA menghantar dan menerima SMS dari peralatan mobile (HP, phone, dll) melalui SMS *Gateway's shortcode*.

SMS *gateway* merupakan sistem aplikasi untuk mengirim dan/atau menerima SMS, terutama digunakan dalam aplikasi bisnis, baik untuk kepentingan promosi, servis kepada *customer*, pengadaan content produk atau jasa, dan seterusnya. Karena merupakan sebuah aplikasi, maka fitur-fitur yang terdapat di dalam SMS *gateway* dapat dimodifi kasi sesuai dengan kebutuhan.

Beberapa fitur yang umum dikembangkan dalam aplikasi SMS *gateway* adalah:

1. *Auto-reply*.

SMS *gateway* secara otomatis akan membalas SMS yang masuk. Contohnya untuk keperluan permintaan informasi tertentu (misalnya kurs mata uang atau jadwal perjalanan), di mana pengirim mengirimkan SMS dengan format tertentu yang dikenali aplikasi, kemudian aplikasi dapat melakukan *auto-reply* dengan membalas SMS tersebut, berisi informasi yang dibutuhkan.

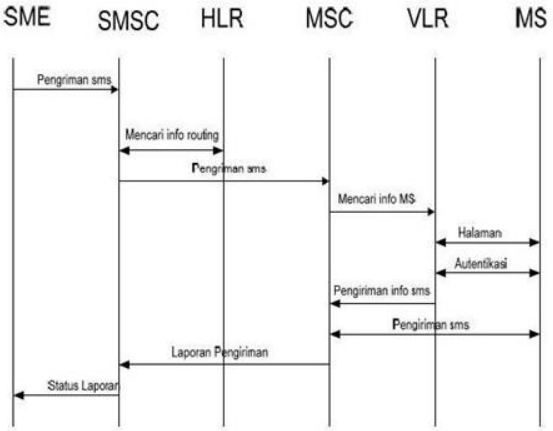
2. Pengiriman massal.

Disebut juga dengan istilah SMS *broadcast*, bertujuan untuk mengirimkan SMS ke banyak tujuan sekaligus. Misalnya, untuk informasi produk terbaru kepada pelanggan.

3. Pengiriman terjadwal.

Sebuah SMS dapat diatur untuk dikirimkan ke tujuan secara otomatis pada waktu tertentu. Contohnya untuk keperluan mengucapkan selamat ulang tahun.

Salah satu hal yang memegang peranan penting dalam pengiriman SMS adalah SMSC (*Short Message Service Center*), yang merupakan jaringan telepon selular yang menangani pengiriman SMS. Jadi, pada saat seseorang mengirimkan sebuah pesan SMS melalui ponselnya, SMSC-lah yang bertugas mengirimkan pesan tersebut ke nomor tujuan. Jika nomor tujuan tidak aktif, maka SMSC akan menyimpan pesan tersebut dalam jangka waktu tertentu. Jika SMS tetap tidak dapat terkirim sampai jangka waktu tersebut berakhir, maka SMS tersebut akan dihapus dari penyimpanan SMSC.



Gambar 2.9 Diagram Proses Pengiriman SMS

Penjelasan Dari Diagram diatas, Proses pengiriman SMS (*Short Message Services*) pertama kali dimulai ketika SMS akan diterima oleh SMSC (SMS *Center*) dari SME (*Short Message Entity*). Setelah dilakukan pengontrolan parameter, maka SMSC–GMSC akan mencari suatu informasi tentang MS pelanggan di HLR (*Home Location Register*) yang berisi informasi *administrative* dari semua pelanggan yang terdaftar dari suatu jaringan GSM beserta lokasi dari *mobile station*. Selanjutnya SMSC akan mengirimkan pesan melalui SMS–GMSC kepada MS (*Mobile Station*) yang dituju dengan format *forward short* message. Setelah proses pengiriman SMC selesai maka SMSC akan mencari suatu informasi yang akan kita dituju dari VLR (*Visitor Location Register*) yang berisi informasi *administrative* terpilih dari HLR yang dibutuhkan untuk kontrol panggilan dan izin bagi pengguna *service* berlangganan. Dimana dalam hal ini akan mengirimkan suatu proses autentifikasi yang akan kita kirimkan. Selanjutnya MSC (*Mobile Station Center*) akan mengirimkan pesan ke MS (*mobile station*), kemudian MSC mengirimkan kembali pesan tersebut. Tetapi bedanya MSC ini akan mengirimkan *fordward* SMS ke MSC bukan ke MS lagi. Apabila SME (*Short Message Entity*) meminta laporan status

maka SMSC akan mengirimkan laporan status ke SME yang mengindikasikan terkirimnya pesan. Di balik tampilan menu *Messages* pada sebuah ponsel sebenarnya terdapat AT Command-AT Command yang bertugas mengirim atau menerima data ke dan dari SMS *Centre*. AT Command tiap–tiap SMS *device* bisa berbeda–beda, tetapi pada dasarnya sama. Perintah–perintah AT Command biasanya disediakan oleh vendor alat komunikasi yang kita beli.**[24]**

Beberapa AT Command yang penting unutk SMS :

1. AT+CMGS : Untuk mengirim SMS
2. AT+CMGL : Untuk memeriksa SMS
3. AT+CMGD : Untuk menghapus SMS

AT Command untuk SMS biasanya diikuti oleh data I/O yang diwakili oleh unit–unit PDU. Data yang mengalir ke/ dari SMS-*Center* harus berbentuk PDU (*Protocol Data Unit*). PDU berisi bilangan–bilangan heksadesimal yang mencerminkan bahasa I/O. PDU terdiri atas beberapa *Header*, *header* untuk mengirim SMS ke SMS-*Center* berbeda dengan SMS yang diterima dari SMS *Center*. Saat kita menerima SMS/MMS dari *handphone* (*mobile originated*) pesan tersebut tidak langsung dikirimkan ke *handphone* tujuan (*mobile terminated*), akan tetapi dikirim terlebih dahulu ke SMSC (SMS *Center*) yang biasanya berada di kantor operator telepon dan kemudian pesan tersebut diteruskan ke *handphone* tujuan.**[25]**

2.13 Teori Ketidakpastian

Ketidakpastian adalah ukuran sebaran yang secara layak dapat dikaitkan dengan nilai terukur. Yang memberikan rentang, terpusat pada nilai terukur, dimana di dalam rentang tersebut terletak nilai benar dengan kemungkinan tertentu.

2.14 Klasifikasi Komponen Ketidakpastian

Ketidakpastian pengukuran terdiri dari beberapa komponen yang dapat diklasifikasikan menurut metode yang digunakan untuk menaksir nilai numeriknya:

1. Tipe A : yang dievaluasi dengan analisis statistik dari serangkaian pengamatan.
2. Tipe B : yang dievaluasi dengan cara selain analisis statistik dari serangakaian pengamatan.

Klasifikasi komponen ketidakpastian ke dalam tipe A dan tipe B tidak selalu mempunyai hubungan langsung dengan klasifikasi komponen ketidakpastian sebagai ketidakpastian acak dan sistematik.[26]

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)

   Bila pengukuran diulangi beberapa kali, nilai rata-rata dan simpangan baku-nya dapat dihitung. Simpangan baku menggambarkan sebaran nilai yang dapat digunakan untuk mewakili seluruh populasi nilai terukur.

   Rumus umum ketidakpatian untuk tipe A ini adalah:

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

   Dimana:

   σ = Standar deviasi (simpangan baku)

   n = Banyaknya data

   Simpangan baku adalah suatu taksiran sebaran populasi dimana n nilai tersebut diambil.

   Rumus standar deviasi (σ) sendiri adalah sebagai berikut:

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

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*)

$$SSR = \sum(yi - a - bxi)^2 \quad \textbf{(2.4)}$$

$$a = \bar{y} + (b\bar{x}) \quad \textbf{(2.5)}$$

$$b = \frac{n.\sum x_i y_i - \sum xi \sum y_i}{n.\ \sum xi^2 - (\sum xi)^2} \quad \textbf{(2.6)}$$

b. Analisa *Type* B (U_b)
Evaluasi ketidakpastian baku tipe B diperoleh dengan cara selain analisis statistik dari serangkaian pengamatan yang biasanya didasarkan pada justifikasi ilmiah menggunakan semua informasi relevan yang tersedia, yang dapat meliputi:
1. Data pengukuran sebelumnya
2. Pengalaman dengan, atau pengetahuan umum tentang tingkah laku dan sifat instrumen dan bahan yang relevan
3. Spesifikasi pabrik
4. Data yang diberikan dalam sertifikat atau laporan lainnya

5. Ketidakpastian yang diberikan untuk data acuan yang diambil dari data *book*.

Berhubung dalam laporan ini alat ukur standar yang dipakai tidak ada sertifikat kalibrasi, maka rumusnya adalah sebagai berikut:

$$Ub_1 = \frac{0{,}5\ Resolusi}{\sqrt{3}} \quad \textbf{(2.7)}$$

$$Ub_2 = \frac{a}{k} \quad \textbf{(2.8)}$$

Dimana:

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

c. Ketidakpastian Kombinasi (U_c)

Ketidakpastian baku gabungan dari suatu pengukuran, dinotasikan dengan u_c, diambil untuk mewakili taksiran simpangan baku (*estimated standard deviation*) dari hasil pengukuran, yang diperoleh dengan menggabungkan ketidakpastian baku dari setiap taksiran masukan berdasarkan pendekatan deret Taylor orde satu dari model pengukuran. Metode penggabungan ketidakpastian baku ini sering disebut dengan hukum propagasi ketidakpastian. Rumus umum ketidakpastian kombinasi adalah:

$$U_c = \sqrt{\sum (U_a)^2 + \sum (U_b)^2} \quad \textbf{(2.9)}$$

Atau secara umum:

$$Uc^2 = \Sigma (C_i.U_i)^2 \quad \textbf{(2.10)}$$

Dengan C_i = Koefisien sensitifitas dari ketidakpastian ke-i

d. Derajat Kebebasan Effektif (V_{eff})

Perlunya perhitungan derajat kebebasan efektif terkait dengan komponen ketidakpastian adalah untuk memperoleh pemelihan nilai faktor pengali yang tepat untuk distribusi Student's t dan juga memberikan indikasi kehandalan penaksiran ketidakpastian.

Derajat kebebasan efektif yang tinggi mewakili jumlah pengukuran yang besar, sebaran yang sempit, dan keyakinan yang tinggi terhadap nilai tersebut, dan sebaliknya. Rumusnya adalah sebagai berikut:

$$V_{eff} = \frac{(U_c)^4}{\sum^{(U_i)^4}/_{V_i}} \quad \textbf{(2.11)}$$

Dengan:

U_c = Ketidakpastian kombinasi/gabungan

U_i = Ketidakpastian individual ke-i

V_i = Derajat kebebasan pada ketidakpastian individual ke- i

e. Ketidakpastian bentangan (U_{exp})

Ukuran ketidakpastian perlu untuk memenuhi kemungkinan yang memadai yang diistilahkan dengan ketidakpastian bentangan (U_{exp}).

Praktek internasional yang biasa diterapkan adalah memberikan tingkat kepercayaan sekitar 95% (95.45%). Untuk tingkat kepercayaan tertentu, nilai faktor cakupan bervariasi terhadap derajat kebebasan efektif. Rumus ketidakpastian diperluas (*expanded uncertainty*) adalah:

$$U_{95} = k\, U_c \quad \textbf{(2.12)}$$

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 numerik yang digunakan sebagai pengali terhadap ketidakpastian baku gabungan untuk memperoleh ketidakpastian bentangan. Faktor cakupan dicari menggunakan tabel *T-Student Distribution* **[27]**.

Tabel 2.4 *T-Student Distribution*[28]

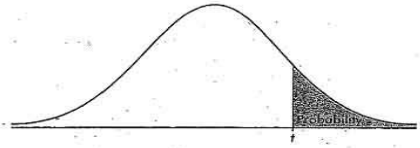


TABLE B: *t*-DISTRIBUTION CRITICAL VALUES

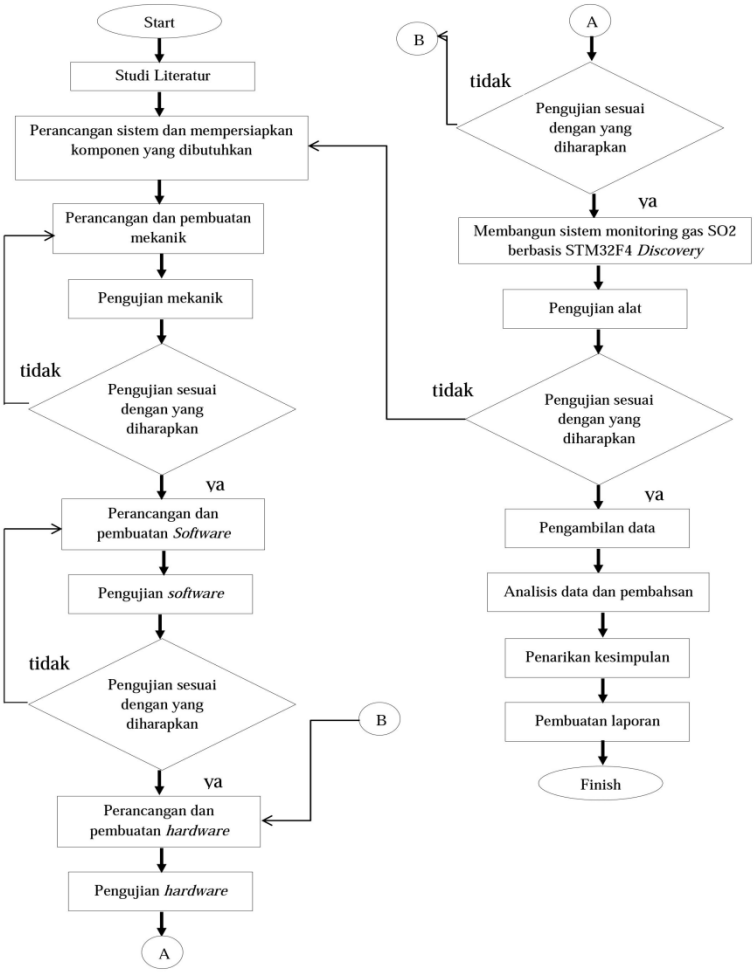
	Tail probability *p*											
df	.25	.20	.15	.10	.05	.025	.02	.01	.005	.0025	.001	.0005
1	1.000	1.376	1.963	3.078	6.314	12.71	15.89	31.82	63.66	127.3	318.3	636.6
2	.816	1.061	1.386	1.886	2.920	4.303	4.849	6.965	9.925	14.09	22.33	31.60
3	.765	.978	1.250	1.638	2.353	3.182	3.482	4.541	5.841	7.453	10.21	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*											

(*Halaman ini sengaja dikosongkan*)

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.

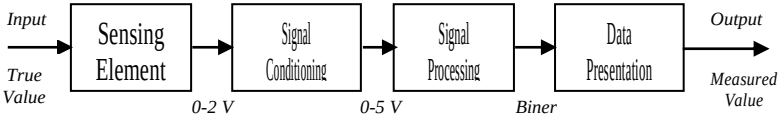


Gambar 3.1 *Flowchart* Pengerjaan Tugas Akhir

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 Emisi Kadar Gas Sulfur Dioksida Menggunakan Sensor MQ 136 Berbasis Mikrokontroller *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 kadar gas sulfur dioksida (SO_2) berbasis STM32F4 *Discovery*. Setelah itu dari sistem monitoring gas yang telah dibuat, dilakukan pengujian alat dengan memberikan *input* berupa serbuk sulfur dioksida yang dibakar untuk untuk mengetahui apakah sensor MQ 136 sudah dapat bekerja dengan baik. Apabila semua rancang bangun sistem monitoring gas dapat bekerja dengan baik, maka selanjutnya dilakukan pengambilan data pada rancang bangun 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 kalibrasi. Setelah semua hasil yang diinginkan tercapai mulai dari studi literatur hingga analisa data dan kesimpulan dicantumkan dalam sebuah laporan.

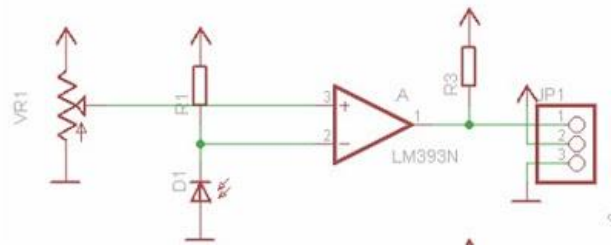
Diagram merupakan pernyataan hubungan yang berurutan dari suatu atau lebih komponen yang memiliki kesatuan kerja tersendiri, dan setiap blok komponen mempengaruhi komponen lainnya. Diagram blok merupakan salah satu cara yang paling sederhana untuk menjelaskan cara kerja dari suatu sistem. Dengan diagram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

Pada perancangan sistem monitoring konsentrasi gas sulfur dioksida (SO_2) ini terdapat diagram blok pengukuran. Berikut merupakan diagram blok sistem pengukuran secara umum.



Gambar 3.2 Diagram Blok Sistem Monitoring Gas

Gambar 3.2 di atas merupakan gambar diagram blok sistem monitoring gas SO_2. Pada sistem monitoring gas ini menggunakan pada rangkaian sensor MQ-136 Sensor MQ 136 merupakan *sensing element*. Keluaran dari sensing element merupakan tegangan yang masih sangat kecil sehingga diteruskan ke pengkondisisan sinyal. Pengkondisian sinyal dalam modul sensor MQ 136 menggunakan IC LM 393**.** IC LM 393 berfungsi untuk membandingkan dua macam tegangan yang terdapat pada kedua inputnya. Dalam aplikasinya biasanya salah satu pin input dari komparator sebagai tegangan reverensi sedangkan pin input lainya sebagai tegangan yang akan dibandingkan.



Gambar 3.3 Skematik IC LM 393

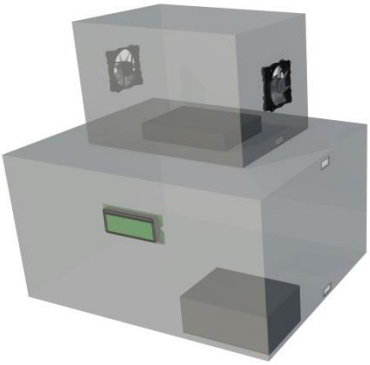
Pada rangkaian tersebut tegangan referensinya diperoleh dari sebuah VR (Variable Resistor) dan tegangan yang akan dibandingkan berasal dari sensor MQ-136 yang dirangkai menjadi rangkaian pembagi tegangan. Dengan tegangan reverensi dari VR maka output dari komparator dapat diatur "pada konsentrasi berapa output dari regulator akan bernilai nol". Dalam aplikasinya output dari komparator LM 393, membutuhkan resistor pullup dengan tegangan V+ yaitu untuk menjaga tegangan output supaya memiliki logika satu ketika kondisi diam. LM 393 akan mengubah tegangan (mv) menjadi tegangan (v). Keluaran

rangkaian sensor MQ 136 berupa digital output, maka tahap selanjutnya adalah pemrosesan sinyal. Pemrosesan sinyal dilakukan oleh mikrokontroller STM32F4 *Discovery.* Sinyal digital yang diterima adalah berupa bilangan digital dimana bilangan digital ini menunjukkan pengukukuran konsentrasi gas karbonmonoksia (CO). Maka, agar dapat ditampilkan dalam angka, sinyal tersebut diolah di mikrokontroller STM32F4 *Discovery*. Setelah diolah untuk penampilan data digunakan *LCD (Liquid Crystal Display)*. Ketika *sensor* MQ 136 telah mendapat *input* gas maka oleh *sensor* akan diubah menjadi tegangan dan kemudian *output* ini dibaca oleh ADC (*Analog to Digital Converter*) internal dari mikrokontroler STM32F4 yang kemudian data dikalkulasikan dengan rumusan tertentu sehingga pada tahap berikutnya sistem dapat menentukan apakah tekanan yang dihasilkan sudah sesuai dengan nilai yang seharusnya atau belum. Konsentrasi gas yang terbaca oleh *sensor* akan ditampilkan ke display LCD (*Liquid Crystal Display*) 16x2 yang ditampilkan dalam satuan ppm.

3.2 Desain Rancang Bangun Sistem Monitoring Gas SO_2

Rancang bangun sistem monitoring SO_2 terdiri dari 2 *box* yang memiliki ukuran berbeda. Penempatan box secara vertical. *Box* pertama yang bertempat diatas berukuran 15 x 15 x 15 cm, dengan menggunakan bahan dari *acrylic* yang diharapkan *box* tidak mudah memanas dan mempengaruhi sensor. Terdapat *exhaust fan* pada *box* ini yang berfungsi untuk menghisap udara di dalam *box* untuk dibuang ke *box*, dan pada saat bersamaan menarik gas di luar ke dalam *box*.

Box kedua yang bertempat dibawah berukuran 30 x 30 x 30 cm, dengan menggunakan bahan *acrylic* supaya komponen alat yang digunakan tidak terlihat dari luar *box*.



Gambar 3.4 Desain Sistem Monitoring gas SO_2

3.3 Perancangan *Sensor MQ 136*

Sensor Gas MQ-136 merupakan *sensor* yang digunakan untuk mengetahui kadar atau konsentrasi gas (SO_2), *sensor* ini bisa mengukur dengan *range* 1〜200ppm SO_2, dan tegangan *output* berada ≥ 0.5 V(in 500 ppm SO_2).

Sensor ini yang bereaksi terhadap kadar gas sulfur dioksida yang terdapat dalam udara. Modul ini dapat diaplikasikan sebagai alarm peringatan dini, ataupun gas detector untuk membantu proses industri yang melibatkan gas dioksida.

Sensor ini nantinya akan mengirimkan data kepada mikrokontroler STM32F4 *Discovery* dan selanjutnya dapat ditampilkan dalam *Liquid Crystal Display* (LCD) dan data hasil pembacaan konsentrasi gas tersebut dapat disimpan melalui *data looger* pada *SD Card* dan dikirimkan melalui *SMS Gateway*.

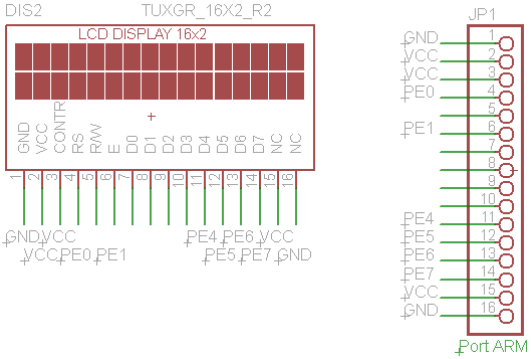
Gambar 3.5 SO_2 Gas Sensor *Module*

Pada modul sensor MQ-136 ini terdapat 6 kaki. Pada kaki 1 merupakan Vin yang berfungsi sebagai tegangan sumber untuk menhidupkan sensor. Pada kaki 2 merupakan Aout yang berfungsi sebagai keluaran analog dari sensor. Pada kaki 3 merupakan Dout yang berfungsi sebagai keluaran digital dari sensor. Pada kaki 4 merupakan Ground.

Dalam perancangannya,Modul sensor MQ 136 dihubungkan dengan mikrokontroller STM32F4 Discovery yang sudah tersambung dengan shield yang telah dibuat. kaki Vin pada modul disambungkan ke terminal Vcc pada shield STM32F4 Discovery. kaki Ground pada modul MQ 136 disambungkan ke terminal Ground STM32F4 Discovery.

3.4 Perancangan *Liquid Crystal Display* (LCD)

Rangkaian *Liquid Crystal Display* (LCD) ke mikrokontroler STM32F4 *Discovery* dapat dilihat pada gambar dibawah ini:



Gambar 3.6 Rangkaian LCD pada STM32F4 *Discovery*

Gambar 3.5 diatas merupanakan skematik rangkaian lcd dengan STM32F4 *Discovery*. *Display* LCD adalah suatu *modul* penampil. Dalam hal ini untuk menampilkan data yang terdeteksi pada *sensor* sehingga dapat ditampilkan data berupa *digital* yang menunjukkan *display* nilai *pressure* pada *hydrogen storage*. LCD

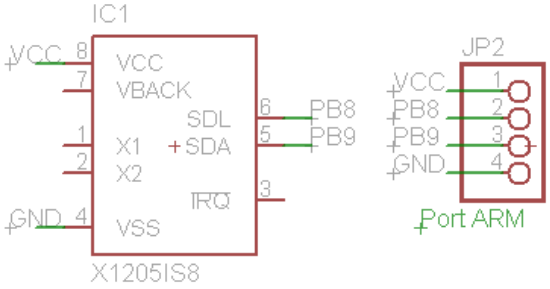
yang digunakan dalam *plant* pengendalian *pressure* ini adalah LCD 16 kolom x 4 baris.

Gambar 3.7 Penempatan LCD untuk *Display*.

3.5 *Modul RTC*

RTC merupakan komponen yang diperlukan untuk memberikan informasi mengenai waktu. Waktu disini dapat berupa detik, menit, hari, bulan dan tahun. Agar tetap dapat bekerja, sebuah RTC dilengkapi dengan baterai, yang umumnya orang-orang menyebutkannya sebagai baterai "CMOS".

Interface atau antarmuka untuk mengakses modul ini yaitu menggunakan i2c atau *two wire* (SDA dan SCL). Sehingga apabila diakses menggunakan mikrontroler misal Arduino Uno pin yang dibutuhkan 2 pin saja dan 2 pin power.



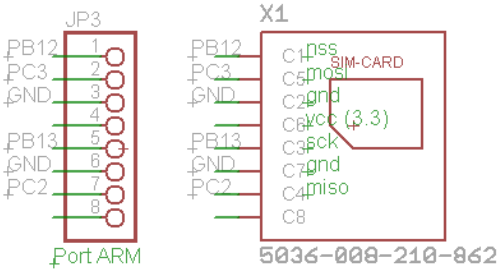
Gambar 3.8 Skematik RTC pada ARM

Dari gambar 3.7 diatas maka modul dapat dirangkai dengan cara sambungkan modul RTC dengan Mikrokontroler STM32F4 dengan menhubungkan 4 buah pin yaitu VCC, GND, SDA, dan SCL. SDA (Serial Data) dan SCL (Serial Clock). Pin Vcc dan GND disambungkan dengan 5V dan Ground sedangkan pin SDA dan SCL disambungkan pin PB9 dan PB8 pada Mikrokontroler.

Gambar 3.9 Modul RTC DS3231

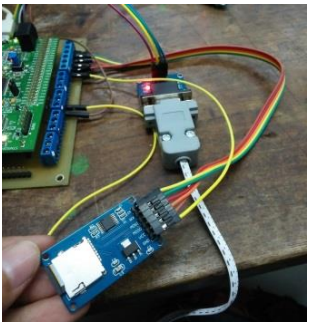
3.6 Perancangan Modul *SD card*

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.



Gambar 3.10 Skematik SD card pada ARM

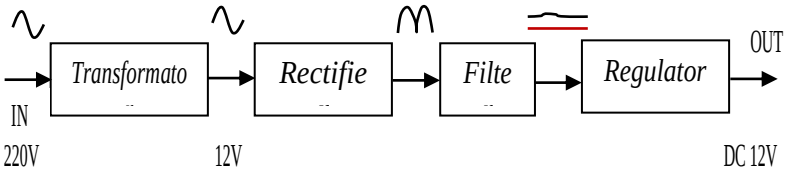
Dari gambar 3.9 diatas maka modul dapat dirangkai dengan cara Sambungkan modul *SD Card* dengan Mikrokontroler STM32F4 dengan menhubungkan 6 buah pin yaitu VCC, GND, MOSI, MOSO, SCK, NSS. Pin Vcc dan GND disambungkan dengan 5V dan Ground sedangkan pin MOSI, MOSO, SCK dan NSS disambungkan pin PC3, PC2, PB13 dan PB 12 pada Mikrokontroler.



Gambar 3.11 Modul *SD Card*

3.7 Rangkaian *Power Supply*

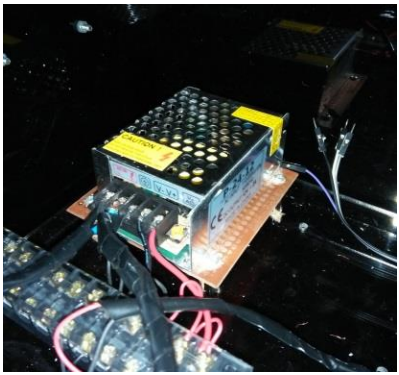
Power supply merupakan sumber tenaga yang dibutuhkan suatu rangkaian elektronika untuk bekerja. Besar *power supply* ini tergantung oleh spesifikasi dari alat masing-masing. Pada perancangan sistem pengendali ini *power supply* digunakan untuk men-*supply* rangkaian mikrokontroler STM32F4, rangkaian *sensor* MQ 136, modul SD Card, RTC, dan LCD 16x4.



Gambar 3.12 Blok Diagram *Power Supply*

Gambar 3.11 diatas dapat dijelaskan bahwa pada rangkaian *power supply* memiliki *input* tegangan yang masuk sebesar 220V

kemudian pada *Transformator* yang berfungsi sebagai penurun tegangan akan menurunkan tegangan menjadi 12 V. Ketika tegangan sudah diturunkan, tegangan masih berupa tegangan AC dan akan dirubah ke tegangan DC dengan *Rectifier*. *Filter* digunakan untuk meratakan sinyal arus yang keluar dari *Rectifier*. Untuk menghasilkan Tegangan dan Arus DC (arus searah) yang tetap dan stabil, diperlukan *Regulator* yang berfungsi untuk mengatur tegangan sehingga tegangan *Output* tidak dipengaruhi oleh suhu, arus beban dan juga tegangan input yang berasal *Output Filter*.

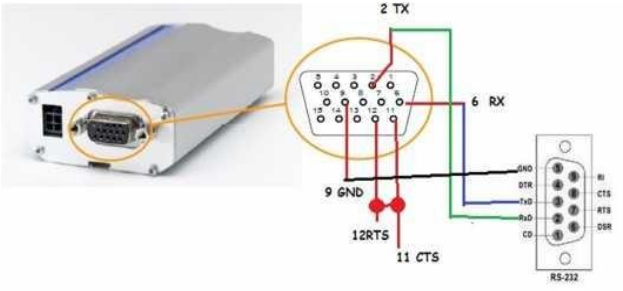


Gambar 3.13 *Power Supply* DC

Power supply untuk tegangan DC digunakan sebagai *supply* untuk perangkat yang membutuhkan tegangan DC 12 V.

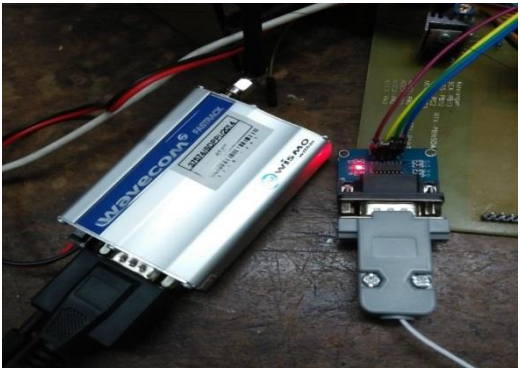
3.8 Perancangan Modem GSM Wavecom

Pada perancangan Modem GSM Wavecom ini menggunakan kabel komunikasi serial RS232 yang biasa digunakan untuk menghubungkan periferal eksternal seperti modem dengan komputer. Modem memiliki level tegangan yang berbeda dengan level tegangan TTL ataupun RS232, tetapi untuk kompatibilitas modem agar bisa terkoneksi dengan PC guna berbagai keperluan maka disediakan kabel data yang compatible dengan standar RS232 sebagai interface untuk koneksi ke PC, untuk konfigurasi port data modem yang digunakan yaitu wavecom m1306b.



Gambar 3.14 Konfigurasi port modem wavecom

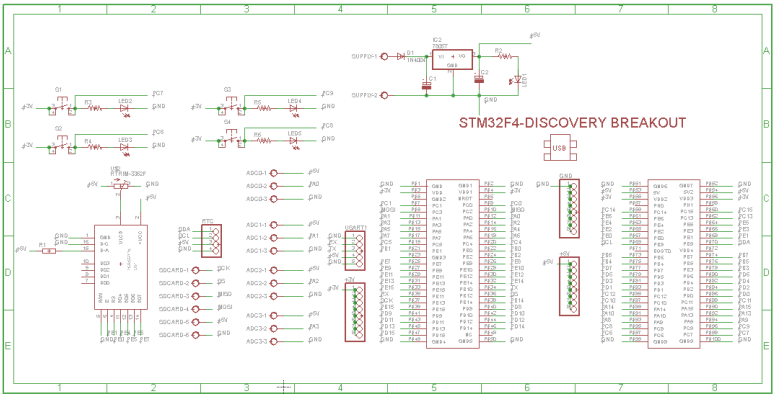
Karena tegangan keluaran dari modem wavecom tidak compatible dengan mikrokontroller maka harus dihubungkan dengan modul RS232 agar sesuai dengan tegangan pada mikrokontroller.



Gambar 3.15 RS232 to TTL

3.9 Perancangan dan Pembuatan *Software*

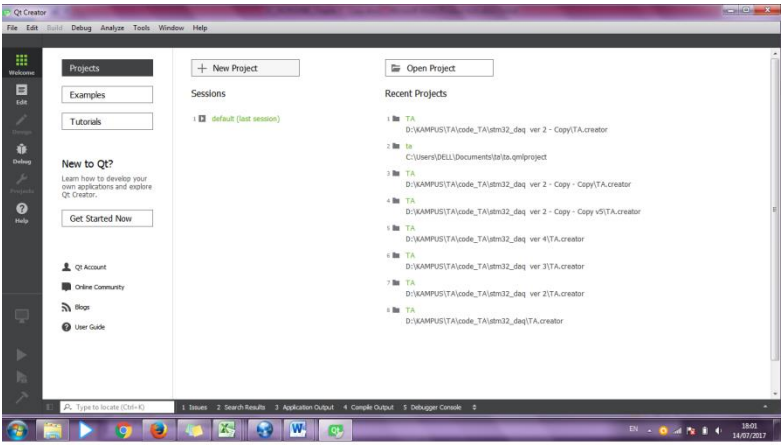
Pada percancangan dan pembuatan *software* dilakukan dengan menggunakan *software eagle*, dengan mengintegrasikan *sensor* MQ 136, SD *Card*, LCD 16x4, RTC, *SMS Gateway* dan mikrokontroler SM32F4 *Discovery*. Berikut merupakan rangkaian skematik mikrokontroler STM32F4, *sensor* MQ136, SD *Card*, RTC, Uart (SMS *Gateway*) dan LCD 16x42 menggunakan *software eagle*.



Gambar 3.16 Rangkaian Skematik Mikrokontroler STM32F4, *sensor* MQ136, SD *Card*, RTC, Uart (SMS *Gateway*) dan LCD 16x4

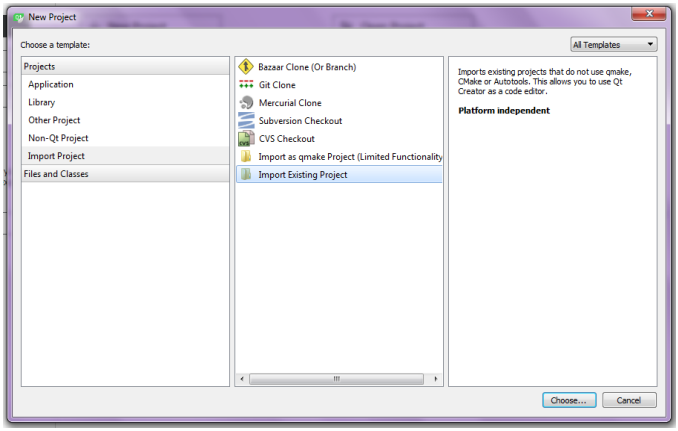
Kemudian untuk pemrograman menggunakan *Qt Creator* dilakukan dengan langkah-langkah sebagai berikut:

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



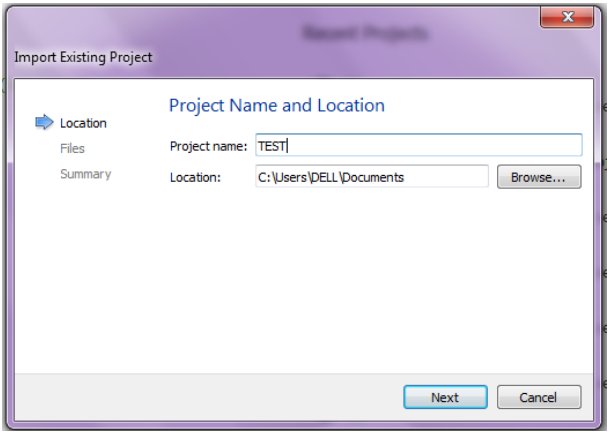
Gambar 3.17 *Create New Project*

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



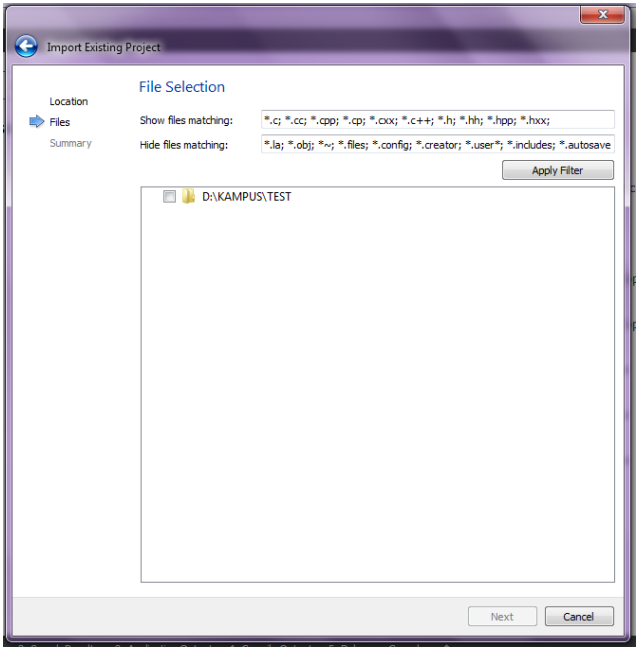
Gambar 3.18 *Project Qt Creator*

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



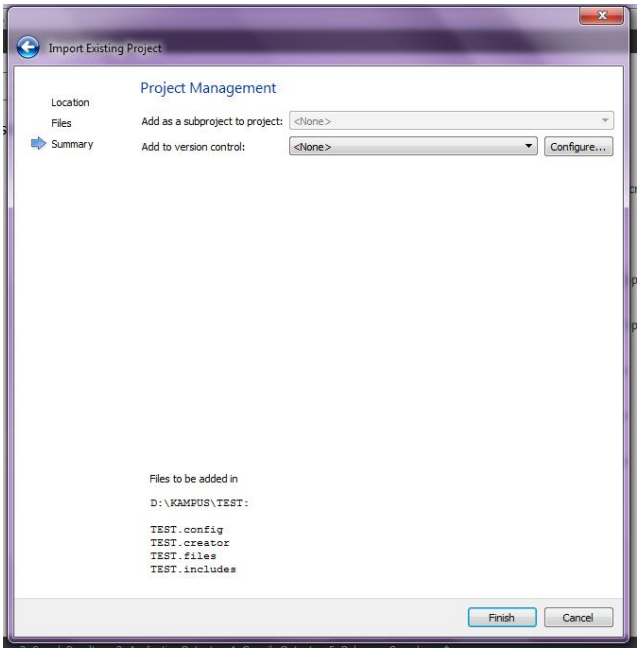
Gambar 3.19 *Import existing project name and location*

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



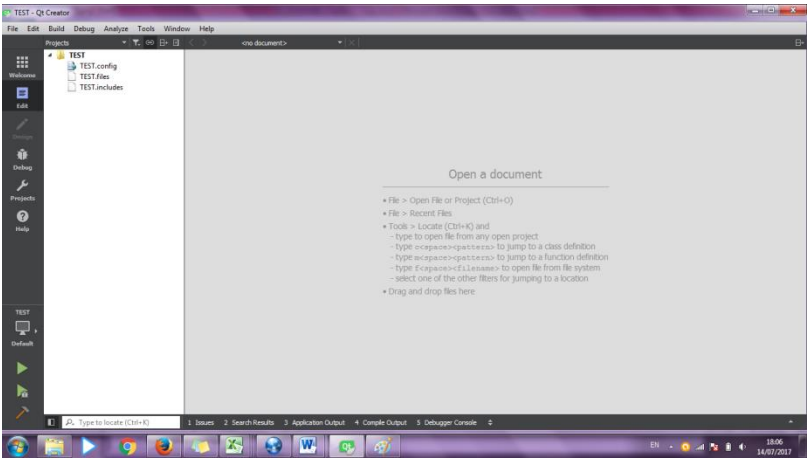
Gambar 3.20 *Import existing project file selection*

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



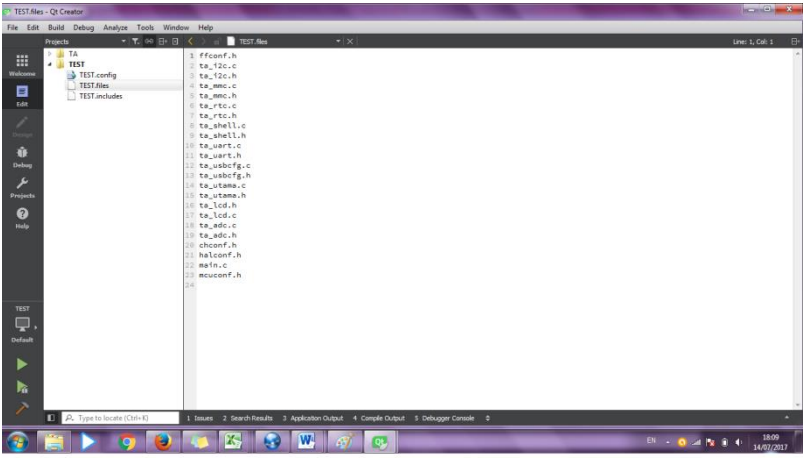
Gambar 3.21 *Import existing project management*

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



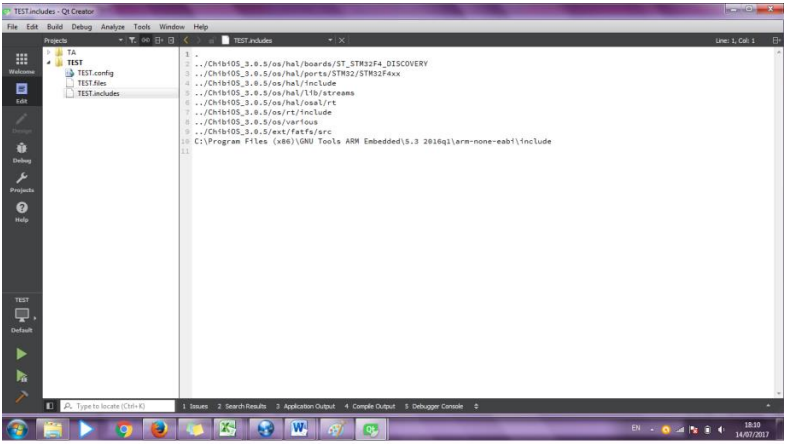
Gambar 3.22 Tampilan awal program

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



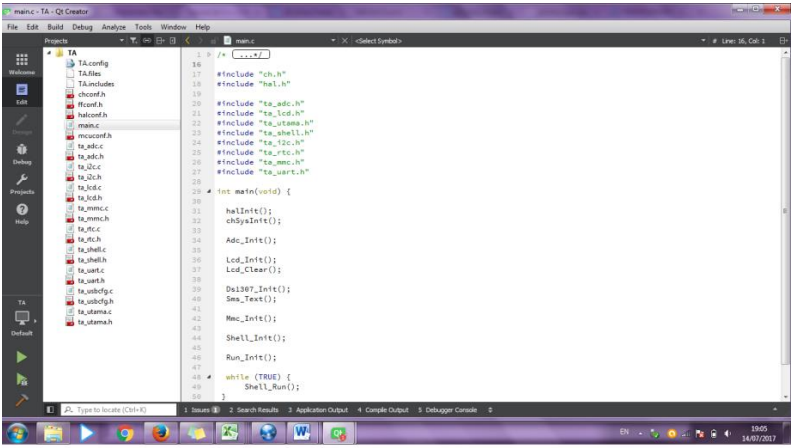
Gambar 3.23 Program *project .files*

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



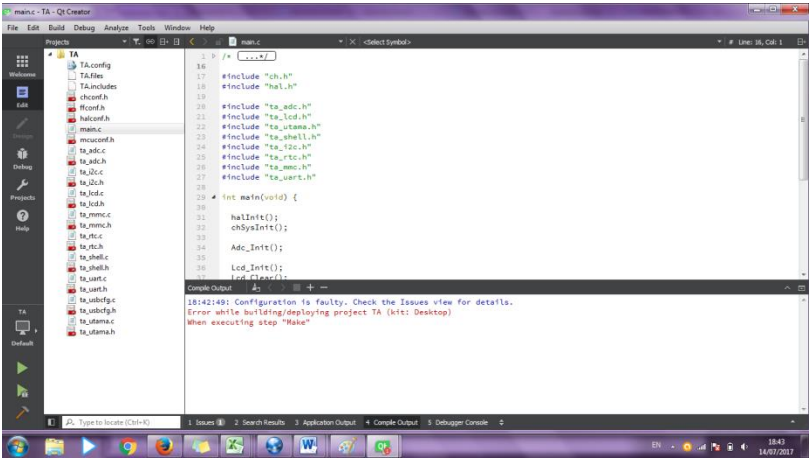
Gambar 3.24 Program *project .includes*

j. Setelah konfigurasi telah dibuat, kemudian ditambahkan *source* dan *header* yang digunakan seperti ADC, I2C, MMC, RTC, SHELL, UART, UTAMA, LCD, MAIN, dll



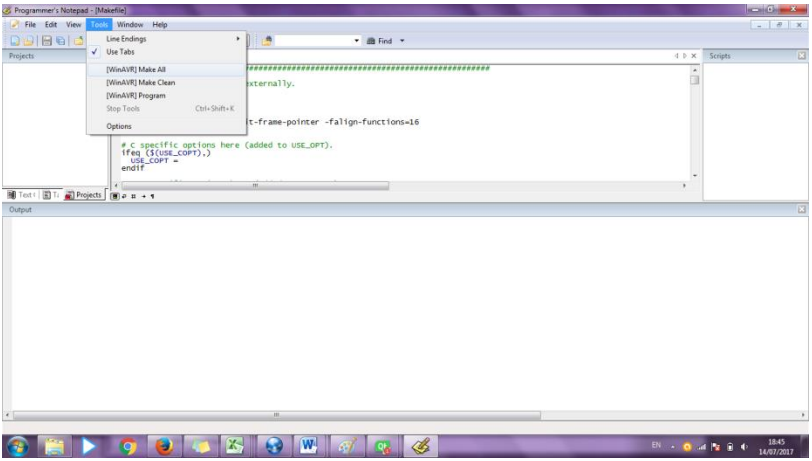
Gambar 3.25 *Class* pada *project*

k. Setelah itu akan muncul program yang telah diatur dan pemrograman bisa dilakukan.
l. Setelah selesai menyusun program, program dapat di-*compile*.



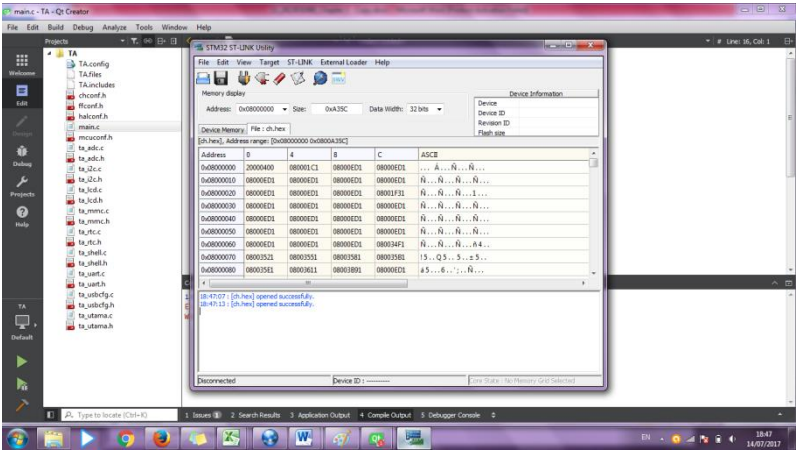
Gambar 3.26 *Build Project Qt Creator*

m. *Software* Qt Creator dan PC tidak *compatible* sehingga diperlukan aplikasi tambahan berupa *notepad++* untuk meng-*compile*. Pada *notepad++* masukkan *project* yang dibuat (Makefile). Kemudian klik *tools* dan *make all*, seperti gambar dibawah ini:



Gambar 3.27 *Make all project* di *Notepad++*

n. *Download project* yang telah di-*compile* menggunakan downloader ST-LINK V2 sesuai gambar dibawah ini:



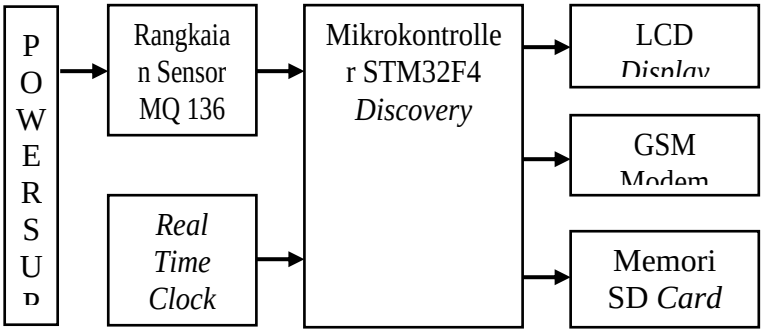
Gambar 3.28 *Download project* di ST-LINK V2

(Halaman ini sengaja dikosongkan)

BAB IV
PENGUJIAN DAN ANALISIS DATA

4.1 Rancang Bangun Alat

Berikut pada gambar 4.1 merupakan blok diagram alur sistem dari sistem monitoring gas SO_2 yang telah dibuat. Sistem monitoring gas sulfur dioksida (SO_2) ini dirangkai untuk memonitoring kadar gas SO_2 yang terdapat di udara bebas.



Gambar 4.1 Blok Diagram Alur Sistem

Alat *Monitoring* Gas SO_2 merupakan detektor gas SO_2 yang memiliki fasilitas sistem pemberitahuan dan pemantauan konsentrasi dan status kondisi gas SO_2 dari jarak jauh menggunakan teknologi *SMS gateway*.

SMS Gateway yang dirancang berupa pemberitahuan dini terhadap kondisi gas SO_2 berdasarkan tingkat konsenstrasi gas yang dideteksi. Pemberitahuan ini akan dikirim melalui SMS ke handphone *user* apabila konsentrasi gas yang dideteksi melampaui ambang batas konsentrasi gas SO_2 yang diprogram dalam mikrokontroler.

Selain dilengkapi dengan teknologi *SMS Gateway,* alat monitoring ini juga dilengkapi penyimpanan data menggunakan SD Card dan juga menunjukkan waktu pengambilan data yang diperoleh menggunakan RTC (*Real Time Clock*)

Berikut ini adalah perancangan sistem monitoring gas SO2 pada berbasis STM32F4 *Discovery*.

Gambar 4.2 Alat Monitoring Gas

4.2 Pengujian Rangkaian Sensor MQ 136

Pada pengujian rangkaian sensor MQ 136 ini menggunakan cara pengambilan data yang didapat dari uji udara dengan dua *variable* yakni udara tanpa polutan dan udara dengan polutan. Sebelum melakukan uji, sensor MQ 136 diharuskan *pre heating* minimal 24 jam. Dilakukan pengambilan data pada uji sensor dan didapatkan nilai bahwa perubahan output sensor sesuai dengan *variable* yang diujikan. Berikut adalah hasil monitoring yang telah dilakukan :

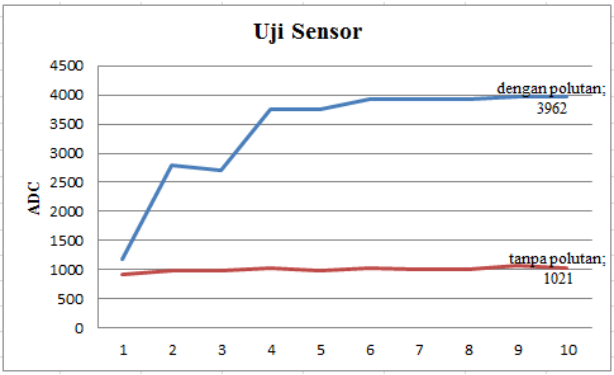
Tabel 4.1 Tabel Hasil Monitoring Kadar Gas SO_2

No	Udara tanpa polutan	Udara dengan polutan
1	1021	1176
2	1067	2800
3	1012	2708
4	1003	3763
5	1024	3763

Tabel 4.1 (lanjutan)

6	973	3924
7	1018	3924
8	983	3924
9	974	3962
10	928	3962

Dari pengambilan data diatas dapat dilihat hasil grafik sebagai berikut



Gambar 4.3 Grafik uji sensor

Dapat dilihat bahwa nilai kadar gas SO2 yang terdapat pada udara tanpa polutan bahwa output memiliki perubahan ppm dalam keadaan stabil dan terdapat kenaikan ketika kondisi udara dengan polutan.

Dilakukan pengambilan data untuk mengetahui apakah sensor sudah bekerja dengan baik maka dilakukan pengambilan data untuk kalibrasi alat. Berikut hasil pengambilan data yang diperoleh:

Tabel 4.2 Tabel Pengambilan data monitoring

No	ADC	Standar
1	1275	1,110
2	1266	0,985

Tabel 4.2 (lanjutan)

3	1195	0,810
4	1051	0,680
5	1025	0,590
6	964	0,523
7	910	0,355
8	810	0,175
9	784	0,156
10	722	0,084
11	700	0,076
12	687	0,068

Dari data diatas diperoleh grafik perbandingan perbedaan antara alat monitoring dengan alat standar. Alat standar yang digunakan adalah Aeroqual Source. Karena adanya perbedaan hasil maka dilakukan perhitungan persamaan regresi dengan rumus sebagai berikut:

$$Y = a + bx \quad \textbf{(4.1)}$$

Dimana,

$$a = \frac{(\sum y)(\sum x^2) - (\sum x)(\sum xy)}{n(\sum x^2) - (\sum x)^2} \quad \textbf{(4.2)}$$

$$a = \frac{(5{,}616)(11323037) - (11389)(6205{,}775)}{12(11323037) - (129709321)} = -1{,}205$$

$$b = \frac{n(\sum xy) - (\sum x)(\sum y)}{n(\sum x^2) - (\sum x)^2} \quad \textbf{(4.3)}$$

$$b = \frac{12(6205{,}775) - (11389)(5{,}616)}{12(11323037) - (129709321)} = 0{,}0018$$

Dari perhitungan diatas diperoleh hasil y = 0,0018x - 1,205 maka data yang digunakan seperti berikut:

Tabel 4.3 Tabel Konversi Data ADC ke PPM

No	Kadar Gas (ppm)	ADC	V_{out}	Nilai ppm alat
1	1,110	1275	1,027	1,0487
2	0,985	1266	1,001	0,9943
3	0,810	1195	0,920	0,8226
4	0,680	1051	0,846	0,6679
5	0,590	1025	0,826	0,6237
6	0,523	964	0,776	0,5200
7	0,355	910	0,733	0,4282
8	0,175	810	0,652	0,2582
9	0,156	784	0,631	0,2140
10	0,084	722	0,581	0,1086
11	0,076	700	0,564	0,0712
12	0,068	687	0,553	0,0491

Data keluaran sensor kadar gas SO_2 didapat pada percobaan dengan *range* pengukuran yaitu 0.068 – 1.110ppm dan dilakukan pengambilan data sebanyak 1 kali setiap data.

4.3 Data Spesifikasi Alat

Karakteristik statik adalah karakteristik yang harus di-perhatikan apabila alat tersebut digunakan untuk mengukur suatu kondisi yang tidak berubah karena waktu atau hanya berubah secara lambat laun. Untuk itu perlu dilakukan perhitungan untuk mengetahui nilai karakteristik dari *sensor pressure* MPX5700AP diantaranya sebagai berikut:

Tabel 4.4 Pengambilan Data pada *Sensor* MQ136

No	Standart	Alat	(STD-alat)/std	Nonlinearitas
1	1,110	1,048	0,055	0,000
2	0,985	0,994	-0,009	0,065
3	0,810	0,822	-0,015	0,062

Tabel 4.4 (lanjutan)

4	0,680	0,667	0,019	0,031
5	0,590	0,623	-0,057	0,074
6	0,523	0,520	0,006	0,034
7	0,355	0,428	-0,205	0,104
8	0,175	0,258	-0,469	0,106
9	0,156	0,21	-0,366	0,081
10	0,084	0,108	-0,292	0,045
11	0,076	0,071	0,068	0,015
12	0,068	0,049	0,286	0,000
Jumlah	5,62	5,81	0,98	
Rata-rata	0,33	0,34	0,06	

Sehingga dihasilkan nilai:

Range : 0,04 – 1,04 ppm
Span : 1
Resolusi : 0,01
Non-linieritas : 10,65%
Akurasi : 94.5%
Kesalahan (*Error*) : 5,5%

Berikut ini hasil perhitungan nilai karakteristik statik *pressure* berdasarkan data pada tabel 4.11:

a. Non – Linieritas $(N(I)) = O(I) - (KI + a)$
*(Berdasarkan data naik)
Non – Linieritas maksimum per unit
$= \frac{\hat{N}}{O_{max} - O_{min}} x\ 100\%$ ..... **(4.4)**

Dimana:
$K\ (\text{sensitifitas}) = \frac{\Delta O}{\Delta I} = \frac{1,04 - 0,04}{1,11 - 0,06} = 0,95$
$a\ (zero\ bias) = O_{min} - KI_{min}$ ..... **(4.5)**

$a = 0{,}04 - (0{,}95\ x\ 0{,}06)$

$a = -0{,}017$

Sehingga:

Non – Linieritas maks. Per unit

$= \frac{0{,}106}{1{,}04-0{,}04}\ x\ 100\%$

Non – linieritas = 10,6%

b. Akurasi:

$$A = 1 - \Sigma \left| \frac{Y_n - X_n}{Y_n} \right| x100\% \quad \textbf{(4.6)}$$

Dengan:

Y_n = Pembacaan standar (I) dan

X_n = Pembacaan alat (O)

$A = 1 - (0.055)x100\%$

$A = 0{,}945x100\%$

$A = 94{,}5\%$

c. *Error*:

$$e = 1 - A \quad \textbf{(4.7)}$$

$e = 1 - 0{,}945$

$e = 0{,}055x100\% = 5{,}5\%$

Berikut ini merupakan hasil pengukuran kalibrasi untuk mencari nilai ketidakpastian alat ukur, dimana kalibrasi dilakukan di ruangan terbuka:

A. Nilai Ketidakpastian untuk *Sensor* MQ 136

Tabel 4.5 Data Kalibrasi pada *Sensor* MQ 136 (1)

No.	Standar, Xi (PPM)	Alat	Koreksi Yi	Xi^2
1	1,1100	1,0487	0,0613	1,23210
2	0,9856	0,9943	-0,0086	0,65639
3	0,8101	0,8226	-0,0124	0,65639

Tabel 4.5 (lanjutan)

4	0,6806	0,6679	0,0127	0,46327
5	0,5900	0,6237	-0,0336	0,34819
6	0,5233	0,5200	0,0033	0,27388
7	0,3552	0,4282	-0,0729	0,12621
8	0,1757	0,2582	-0,0824	0,03087
9	0,1566	0,2140	-0,0573	0,02453
10	0,0840	0,1086	-0,0245	0,00706
11	0,076	0,0712	0,0052	0,00583
12	0,0687	0,0491	0,0196	0,00472
Jumlah	5,6167	5,8065	-0,2424	3,82947
Rata-rata	0,4680	0,4838	-0,0242	0,31912

Tabel 4.6 Data Kalibrasi pada *Sensor* MQ 136 (2)

No.	**Standar, Xi (PPM)**	**Yi-Y'**	**$(Yi-Y')^2$**	**XiYi**
1	1,1100	0,0855	0,0073	0,06804
2	0,9856	0,0156	0,0002	-0,0085
3	0,8101	0,0118	0,0001	-0,0100
4	0,6806	0,0369	0,0013	0,0086
5	0,5900	-0,0093	8,79E-05	-0,0198
6	0,5233	0,0275	0,0007	0,0017
7	0,3552	-0,0486	0,0023	-0,0259
8	0,1757	-0,0582	0,0033	-0,0144
9	0,1566	-0,0331	0,0010	-0,0089
10	0,0840	-0,0003	9,86E-08	-0,0020
11	0,0764	0,0294	0,0008	0,0003
12	0,0687	0,0439	0,0019	0,0013
Jumlah	5,6167		0,0195	-0,0096
Rata-rata	0,4680		0,0016	-0,0008

Tabel 4.7 Data Kalibrasi pada *Sensor* MQ 136 (3)

No.	Standar, Xi (PPM)	Yregresi	Residu (R)	R^2 (SR)
1	1,1100	0,0312	0,0300	0,00090
2	0,9856	0,0205	-0,0291	0,00084
3	0,8101	0,0053	-0,0177	0,00031
4	0,6806	-0,0058	0,0185	0,00034
5	0,5900	-0,0136	-0,0199	0,00039
6	0,5233	-0,0194	0,0228	0,00052
7	0,3552	-0,0340	-0,0389	0,00151
8	0,1757	-0,0495	-0,0329	0,00108
9	0,1566	-0,0511	-0,0061	3,83E-05
10	0,0840	-0,0574	0,0329	0,00108
11	0,0764	-0,0581	0,0633	0,00400
12	0,0687	-0,0587	0,0784	0,00615
Jumlah	5,6167			0,01721
Rata-rata	0,4680			0,00143

a. Nilai Ketidakpastian *Type* A:
Sehingga nilai ketidakpastian hasil pengukuran:

$$U_{a1} = \frac{\sigma}{\sqrt{n}} \quad \textbf{(4.8)}$$

$$U_{a1} = 0$$

Sedangkan nilai ketidakpastian regresi $U_{a2} = \sqrt{\frac{SSR}{n-v}}$

Dimana:
SSR (*Sum Square Residual*) = ∑SR(*Square Residual)*
SR = R^2 (*Residu*)
Y_i = Nilai koreksi ke-i

$$Yreg = a + (b.xi) \quad \textbf{(4.9)}$$

$$a = \bar{y_i} + (b.\bar{x_i}) \quad \textbf{(4.10)}$$

$$b = \frac{n.\sum x_i y_i - \sum xi \sum y_i}{n.\ \sum xi^2 - (\sum xi)^2} \quad \textbf{(4.11)}$$

Dimana:
$xi = Pemb.standar,$
$yi = Nilai\ koreksi,$
$n = Jumlah\ data$
$v = variabel\ bebas$

$$b = \frac{(-0{,}163) - (-1{,}065)}{(70{,}459) - (31{,}55)} = 0{,}023181049$$

Sehingga nilai:

$$a = (-0{,}016) + (-0{,}023\ x\ 0{,}47)$$
$$a = -0{,}004964878$$

Jadi, persamaan regresi menjadi:

$$Yreg = (-0{,}00496) + (xi\ x\ (0{,}023))$$

Yang menghasilkan nilai SSR = 0,02152

$$U_{a2} = \sqrt{\frac{SSR}{n-v}} = \sqrt{\frac{0{,}023}{17-2}} = 0{,}048173364$$

b. Nilai Ketidakpastian *Type* B :
Pada tipe ini terdapat 2 parameter ketidakpastian, yaitu ketidakpastian Resolusi (U_{b1}) dan Ketidakpastian alat standar *pressure gauge* (U_{b2}). Dengan perhitungan sebagai berikut:

$$U_{b1} = \frac{a}{\sqrt{3}} = \frac{-0{,}0049}{\sqrt{3}} = -0{,}002866474$$

$$U_{b2} = \frac{\frac{1}{2} x\ Resolusi}{\sqrt{3}} = \frac{\frac{1}{2} x\ 0{,}01}{\sqrt{3}} = 0{,}002886751$$

c. Nilai ketidakpastian kombinasi U_c:

$$U_c = \sqrt{U_{a1}{}^2 + U_{a2}{}^2 + U_{b1}{}^2 + U_{b2}{}^2} \quad \text{.....(4.12)}$$

$$U_c = \sqrt{0^2 + 0{,}048^2 + 0{,}0028^2 + 0{,}0028^2}$$

U_c = 0,048344834

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

V = n-1, sehingga :

V1 = 11; V2 = 11; V3 = 0,5; V4 = 0,5 (berdasarkan table *T-Student*)

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

$$Veff = \frac{(U_c)^4}{\sum {(U_i)^4}/{V_i}} \quad \text{.....(4.13)}$$

$$Veff = \frac{(0{,}044)^4}{{(0)^4}/{11} + {(0{,}048)^4}/{11} + {(0{,}002)^4}/{0{,}5} + {(0{,}002)^4}/{0{,}5}}$$

Veff = 11,15121479

Dari hasil derajat kebebasan yang didapatkan dicari faktor cakupan dengan cara interpolasi

$$\frac{x - x_1}{x_2 - x_1} = \frac{y - y_1}{y_2 - y} \quad \text{.....(4.14)}$$

$$\frac{x - 2{,}228}{2{,}201 - 2{,}228} = \frac{11{,}15121479 - 10}{11 - 10}$$

$$\frac{x - 2{,}228}{-0{,}027} = \frac{1{,}15121479}{1}$$

$$x - 2{,}228 = -0{,}031082799$$
$$x = 2{,}196917201$$

Oleh karena itu, hasil nilai ketidakpastian diperluang sebesar :

$$U_{exp} = k \; x \; U_c \quad \text{.....(4.15)}$$

$$U_{exp} = 2{,}196 \; x \; 0{,}048 = 0{,}105$$

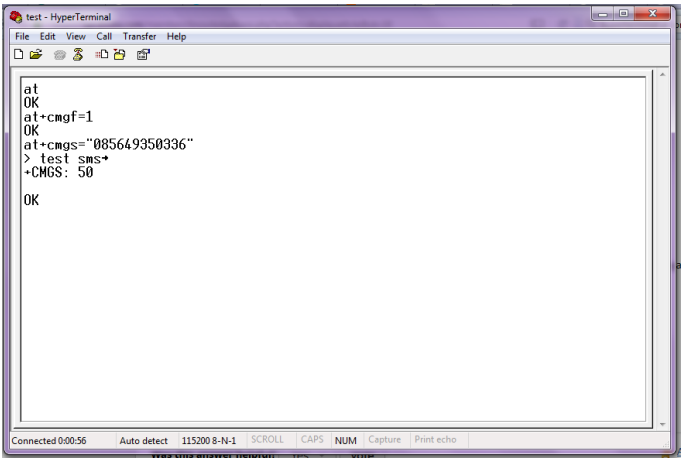
Diperoleh hasil pengukuran

$x = \bar{x} \pm Uncertainty$
$x = 0{,}52 \pm 0{,}105$ ppm
$CL = 95\%$
$k = 2{,}196$

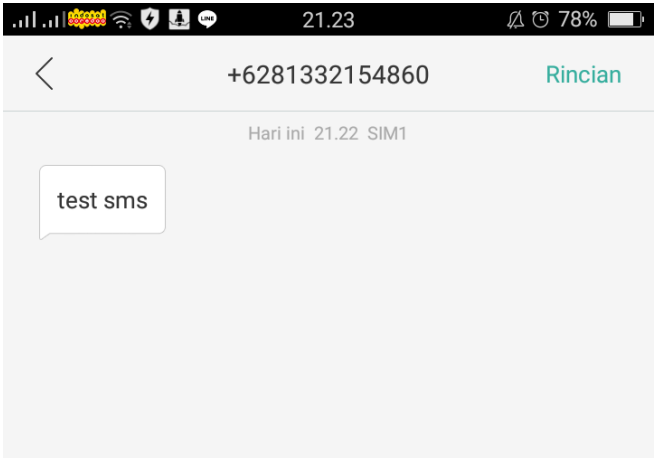
4.4 Pengujian Modem untuk SMS Gateway

Pengujian sistem sms gateway ini terdiri dari pengiriman dan penerimaan sms. Sistem pengiriman sms diprogram hanya untuk satu nomer telepon. Seperti yang telah dijelaskan sebelumnya bahwa alat monitoring gas SO_2 ini akan mengirimkan sms sebagai peringatan sekaligus pemberitahuan konsentrasi gas SO_2 dan kondisinya apabila melebihi ambang batas yang telah ditetapkan yaitu 20 ppm.

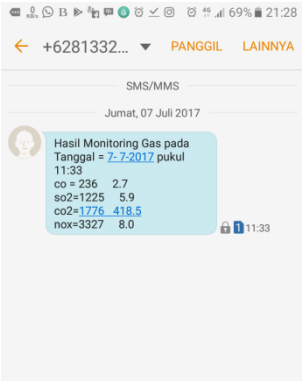
Berikut merupakan hasil pengujian yang telah dilakukan:



Gambar 4.4 *Screenshoot* AT Command pada Hyperterminal



Gambar 4.5 *Screenshoot* SMS yang diterima HP



Gambar 4.6 *Screenshoot* SMS yang diterima HP (ppm)

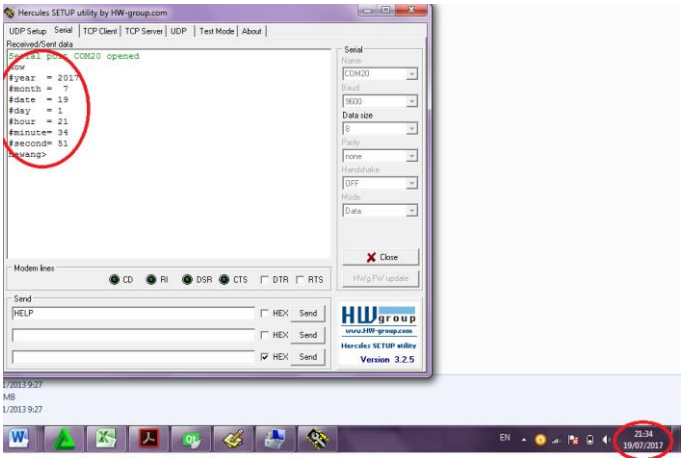
Untuk mengatasi permasalahan agar sms tidak dikirim terus-menerus dalam wilayah ambang batas yang masih sama maka digunakan fungsi *if* dengan kondisi awal seperti berikut ini.

```
//kondisi awal
if (so2 >= 3)
if(udhkirim==0)
Kirim SMS();
udhkirim=1;
else
udhkirim=0; //(Lampiran)
```

Jika gas SO_2 lebih dari 20 ppm maka akan mengirimkan SMS, jika kondisi SMS sudah terpenuhi maka SMS akan berhenti.

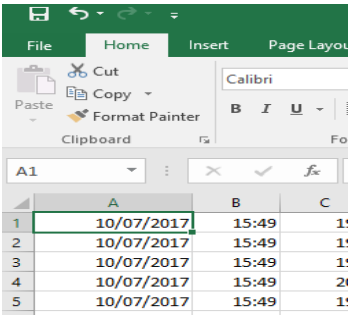
4.5 Pengujian RTC (*Real Time Clock*)

Pengujian rangkaian Real Time Clock dilakukan dengan menerapkan program pada mikrokontroller STM32F4 *Discovery*, dimana RTC DS1307 sebagai timer, dan dibutuhkan tegangan 5V untuk mengaktifkan rangkaian RTC DS1307. Untuk mengatur jam dan tanggal pada RTC dilakukan pengujian melalui Hyperteminal. Bisa dilihat pada gambar 4.7 dibawah ini,



Gambar 4.7 Pengujian data waktu RTC pada Hyperterminal

Pada saat pengujian terlihat bahwa RTC DS1307 bisa bekerja dengan baik yaitu bisa menampilkan jam dan tanggal sesuai dengan program yang sudah dimasukan sebelumnya. Data waktu yang sudah sesuai dicantumkan ke format excel yang akan disimpan ke dalam SD Card. Bisa dilihat pada Gambar 4.9 di bawah ini.



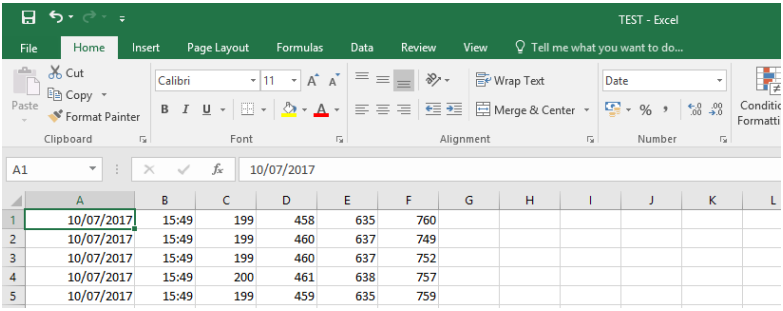
Gambar 4.8 Pengujian data waktu

4.6 Pengujian penyimpanan memori ke SD Card

Pada pengujian modul SD card ini menggunakan cara pengambilan data yang didapat dari empat sensor (Sistem

Monitoring Gas) sensor yang digunakan yaitu sensor MQ 136, sensor MQ 135, sensor MQ 7 dan sensor MG 811. Data yang didapatkan dari sensor akan disimpan ke dalam memori SD card. Data yang disimpan pada SD Card diperoleh setiap 5 detik.

Berikut adalah gambar dari hasil pengujian pembacaan dan penulisan ke dalam memori SD card :



	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 4.9 Pengujian SD Card

Pada gambar 4.9 menjelaskan tentang hasil pembacaan dari empat sensor yang digunakan dan dijelaskan juga kapan waktu pembacaan dan penulisan pada SD card dilakukan. Pada hasil pengujian gambar 4.11 sudah didapatkan hasil yang sesuai dengan yang diinginkan data dapat dibaca atau ditulis kedalam memori SD card dengan hasil yang ditulis berasal dari pembacaan sensor MQ 136, sensor MQ 135, sensor MQ 7 dan sensor MG 811.

BAB V
PENUTUP

5.1 Kesimpulan

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

1. Telah berhasil dibuat alat monitoring gas SO_2 dengan spesifikasi alat yang dimiliki yaitu range antara 0,04 – 1,04 ppm, span sebesar 1 dengan resolusi 0,01, Non-linieritas 10,65%, Akurasi 94.5%, dan kesalahan (Error) 5,5%
2. Dari kalibrasi yang telah dilakukan dapat diketahui bahwa alat monitoring gas kadar SO_2 ini dengan factor cakupan sebesar 2,196 dengan tingkat kepercayaan 95% hasil pengukuran (x) didapatkan $0{,}52 \pm 0{,}105$ ppm.
3. Rancang bangun sistem monitoring ini dilengkapi dengan penyimpanan SD Card yang sesuai dengan waktu sesungguhnya menggunakan RTC dan terdapat SMS *Gateway* komunikasi dengan *mobile*.

5.2 Saran

Saran yang diberikan untuk dilakukan penelitian selanjutnya yaitu:

1. Dibutuhkan pengujian kalibrasi dengan alat standar untuk mengetahui keakurasi dan ketepatan dari sistem monitoing gas ini.
2. Alat ini dapat disempurnakan dengan menambahkan *RTC* sebagai Real Time Clock maka harus dilakukan pengaturan penyesuaian waktu sesungguhnya
3. Penggunaan rangkaian shield sangat dianjurkan untuk mempermudah pemasangan setiap komponen.

(Halaman ini sengaja dikosongkan)

DAFTAR PUSTAKA

[1] Maya Sari. 2015. 10 Penyebab Pencemaran Udara dan Penyebabnya. http://ilmugeografi.com/ilmu-bumi/udara/penyebab-pencemaran-udara (Diakses 10 Juni 2017, 22.00 WIB)

[2] Mahida, U.N. 1981. Pencemaran Udara dan Pemanfaatan Limbah Industri, diterjemahkan oleh Prof. DR.Ir. Otto Soemarsoto. C.V. Rajawali, Jakarta.

[3] Lukman Agus. 2016. WHO: Dunia Darurat Udara Kotor, 60 Ribu Warga Indonesia Meninggal karena Polusi. http://kbr.id/09-2016/who__dunia_darurat_udara_kotor__60_ribu_warga_indonesia_meninggal_karena_polusi/85412.html (Diakses 16 Januari 2017)

[4] Alfiyah, Tati. 2009. Pencemaran Udara. Teknik Lingkungan,ITATS,Surabaya.

[5] Kementerian Lingkungan Hidup dan Kehutanan Direktorat Jenderal Pengendalian Pencemaran & Kerusakan Lingkungan. 2017. Stasiun Pemantau Kualitas Udara. http://iku.menlhk.go.id/ (Diakses 11 Juni 2017, 23.17)

[6] Wardhana, Arya Wisnu. 2001. Dampak Pencemaran Lingkungan. Edisi Revisi Penerbit Andi. Yogyakarta.

[7] Nugroho, Astri. 2005. Bioindikator Kualitas Udara. Penerbit Universitas Trisakti. Jakarta.

[8] Kristanto,P. 2002. Ekologi Industri. Andi, Yogyakarta.

[9] Slamet, Juli Soemirat. 2009. Kesehatan Lingkungan. Cetakan Kedelapan. Gadjah Mada University Press. Yogyakarta.

[10] Anonim. 2016. Prediksi pencemaran udara. http://dinus.ac.id/repository/docs/ajar/Prediksi_Pencemaran_Udara.pdf (Diakses 12 Juni 16.13)

[11] *Datasheet* MQ136. MQ136 Semiconductor Sensor for Sulfur Dioxide. http://www.china-total.com (Diakses 18 Juni 2017, 16.40)

[12] Ardhianto, A. 2010. Pemanfaatan Mikrokontroler ATMega8535 dan Sensor PIR sebagai Pengendali Alat Pengering Tangan. UNS, Surakarta.
[13] http://sir.stikom.edu/1701/4/BAB_II.pdf (Diakses 26 Mei 2017, 16.00 WIB)
[14] UM1472 User manual. Discovery kit with STM32F407VG MCU.http://www.st.com/content/ccc/resource/technical/document/user_manual/70/fe/4a/3f/e7/e1/4f/7d/DM00039084.pdf/files/DM00039084.pdf/jcr:content/translations/en.DM00039084.pdf (Diakses 26 Mei 2017, 16.15 WIB)
[15] Fadillah, I. 2014. Perancangan Alat Ukur Tekanan Udara dengan Menggunakan *Sensor Pressure Gauge* MPX5700 Berbasis ATMega8535. USU, Medan.
[16] Manalu, M., I., A. 2014. Perancangan Alat Ukur Konduktivitas Air (*Conductivity Meter*) *Digital* dengan Sensor Resistif. USU, Medan.
[17] *Datasheet* LCD 16x4. 16x4 Character LCD. http://www.vishay.com/docs/37306/lcd016n004b.pdf (Diakses 19 Juni 2017, 15:56)
[18] *Datasheet* RTC DS3231. *Extremely Accurate I^2C-Integrated RTC/TCXO/Crystal*. https://web.wpi.edu/Pubs/E-project/Available/E-project-010908-124414/unrestricted/DS3231-DS3231S.pdf (Diakses 27 Mei 2017, 18.46 WIB)
[19] MMM999. *Micro SD card reader module for Arduino*. https://www.tindie.com/products/mmm999/micro-sd-card-reader-module-for-arduino/ (Diakses 27 Mei 2017, 10.00 WIB)
[20] HYPERLINK http://www.chibios.org/dokuwiki/doku.php (Diakses 19 Juni 2017, 16.37 WIB)
[21] Iwan. 2017. Mengenal Qt Creator. http://www.proweb.co.id/articles/mobile_development/qt_creator.html (Diakses 19 Juni 2017, 18:40)
[22] Wahid, N. 2016. Koding C++ dengan QT Creator. *Leanpub Publishing*. Jakarta

[23] HYPERLINK. http://library.binus.ac.id/eColls/eThesisdoc/Bab2/2013-1-01231-IF%20Bab2001.pdf (Diakses 16 Juli 2017, 08.44 WIB)
[24] Arif. 2014. Aplikasi Sms Gateway Presensi Siswa Berbasis Web Dengan Php Dan Mysql, Aplikasi Sms Gateway. Universitas Negeri Yogyakarta. Yogjakarta.
[25] Ali.2011. Pengembangan Sistem Informasi Monitoring Tugas Akhir Berbasis *Short Message Service* (SMS) *Gateway* di Fasilkom Unsri, Universitas Sriwijaya Kampus UNSRI Indralaya, Indralaya, JUSI Vol. 1, No. 2 ISSN 2087-8737 September 2011. Yogjakarta.
[26] Komite Akreditasi Nasional. 2003. Pedoman Evaluasi Dan Pelaporan Ketidakpastian Pengukuran. Jakarta.
[27] Asy'ari, M., K. 2014. Kalibrasi *Flow Meter* dalam Aliran Fluida pada Sistem *Manifold*. Program Studi D3 Metrologi dan Instrumentasi, Jurusan Teknik Fisika, FTI-ITS, Surabaya.
[28] Laboratorium Pengukuran Fisis. 2013. Modul Teknik Pengukuran dan Kalibrasi. Teknik Fisika, FFI-ITS, Surabaya.

LAMPIRAN A
(*LISTING* PROGRAM PADA *CODE VISION AVR*)

//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 B
(*DATA SHEET MQ 136*)

MQ136 Semiconductor Sensor for Sulfur Dioxide

Sensitive material of MQ136 gas sensor is SnO_2 which with lower conductivity in clean air. When the target SO2 gas exist, the sensor's conductivity is more higher along with the gas concentration rising. Please use simple electrocircuit, Convert change of conductivity to correspond output signal of gas concentration.

MQ136 gas sensor has high sensitity to SO2, also could be used to detect other vapor which contains Sulfur. It has low sensitivity to normal combustible gases, which is with low cost and suitable for different application.

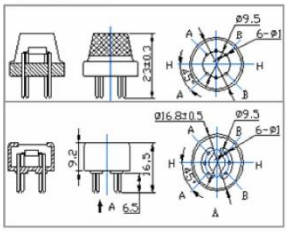
Character

* Good sensitivity to SO2
* Long life and low cost
* Simple drive circuit

Application

* Domestic SO2 concentration detector
* Industrial SO2 leakage detector
* Portable SO2 detector

Configuration

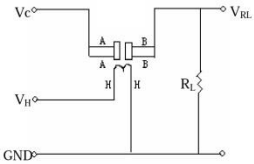


Technical Data

Basic test loop

Model No.			MQ136
Sensor Type			Semiconductor
Standard Encapsulation			Bakelite (Black Bakelite)
Detection Gas			SO2
Concentration			1-200ppm (SO2)
Circuit	Loop Voltage	V_c	≤24V DC
	Heater Voltage	V_H	5.0V±0.2V ACorDC
	Load Resistance	R_L	Adjustable
Character	Heater Resistance	R_H	31Ω±3Ω（Room Tem.）
	Heater consumption	P_H	≤900mW
	Sensing Resistance	R_s	2KΩ-20KΩ(in 50ppm SO_2)
	Sensitivity	S	$Rs(in\ air)/Rs(50ppm\ SO_2) \geqslant 3$
	Slope	α	$\leqslant 0.6(R_{150ppm}/R_{50ppm}\ SO_2)$
Condition	Tem. Humidity		20℃±2℃；65%±5%RH
	Standard test circuit		Vc: 5.0V±0.1V； V_H: 5.0V±0.1V
	Preheat time		Over 48 hours

Power of Sensitivity body(Ps): $Ps=Vc^2 \times Rs/(Rs+RL)^2$



The above is basic test circuit of the sensor. The sensor need to be put 2 voltage, heater voltage（VH） and test voltage（VC）. VH used to supply certified working temperature to the sensor, while VC used to detect voltage (VRL) on load resistance （RL） whom is in series with sensor. The sensor has light polarity, Vc need DC power. VC and VH could use same power circuit with precondition to assure performance of sensor. In order to make the sensor with better performance, suitable RL value is needed:

Resistance of sensor(Rs): Rs=(Vc/VRL-1)×RL

Sensitivity Characteristics

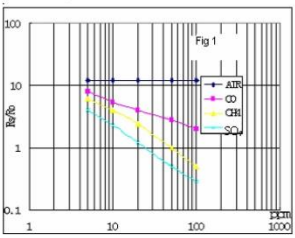


Fig.1 shows the typical sensitivity characteristics of the MQ136, ordinate means resistance ratio of the sensor (Rs/Ro), abscissa is concentration of gases. Rs means resistance in different gases, Ro means resistance of sensor in 50ppm SO2. All test are under standard test conditions.

Influence of Temperature/Humidity

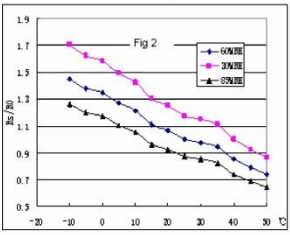
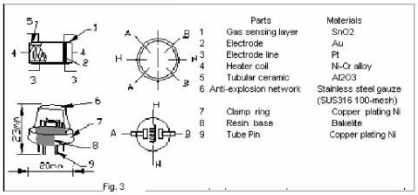


Fig.2 shows the typical temperature and humidity characteristics. Ordinate means resistance ratio of the sensor (Rs/Ro), Rs means resistance of sensor in 50ppm SO2 under different tem. and humidity. Ro means resistance of the sensor in environment of 50ppm SO2, 20℃/65%RH

Structure and configuration



Structure and configuration of MQ136 gas sensor is shown as Fig. 3, 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-4 have 6 pin, 4 of them are used to fetch signals, and other 2 are used for providing heating current.

Notification

1 Following conditions must be prohibited

1.1 Exposed to organic silicon steam

Organic silicon steam cause sensors invalid, sensors must be avoid exposing to silicon bond, fixature, silicon latex, putty or plastic contain silicon environment

1.2 High Corrosive gas

If the sensors exposed to high concentration corrosive gas (such as H_2Sz, SO_X, Cl_2, HCl etc), it will not only result in corrosion of sensors structure, also it cause sincere sensitivity attenuation.

1.3 Alkali, Alkali metals salt, halogen pollution

The sensors performance will be changed badly if sensors be sprayed polluted by alkali metals salt especially brine, or be exposed to halogen such as fluorin.

1.4 Touch water

Sensitivity of the sensors will be reduced when spattered or dipped in water.

1.5 Freezing

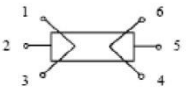
Do avoid icing on sensor'surface, otherwise sensor would lose sensitivity.

1.6 Applied voltage higher

Applied voltage on sensor should not be higher than stipulated value, otherwise it cause down-line or heater damaged, and bring on sensors' sensitivity characteristic changed badly.

1.7 Voltage on wrong pins

For 6 pins sensor, if apply voltage on 1、3 pins or 4、6 pins, it will make lead broken, and without signal when apply on 2、4 pins



2 Following conditions must be avoided

2.1 Water Condensation

Indoor conditions, slight water condensation will effect sensors performance lightly. However, if water condensation on sensors surface and keep a certain period, sensor' sensitivity will be decreased.

2.2 Used in high gas concentration

No matter the sensor is electrified or not, if long time placed in high gas concentration, if will affect sensors characteristic.

2.3 Long time storage

The sensors resistance produce reversible drift if it's stored for long time without electrify, this drift is related with storage conditions. Sensors should be stored in airproof without silicon gel bag with clean air. For the sensors with long time storage but no electrify, they need long aging time for stbility before using.

2.4 Long time exposed to adverse environment

No matter the sensors electrified or not, if exposed to adverse environment for long time, such as high humidity, high temperature, or high pollution etc, it will effect the sensors performance badly.

2.5 Vibration

Continual vibration will result in sensors down-lead response then repture. In transportation or assembling line, pneumatic screwdriver/ultrasonic welding machine can lead this vibration.

2.6 Concussion

If sensors meet strong concussion, it may lead its lead wire disconnected.

2.7 Usage

For sensor, handmade welding is optimal way. If use wave crest welding should meet the following conditions:

2.7.1 Soldering flux: Rosin soldering flux contains least chlorine
2.7.2 Speed: 1-2 Meter/ Minute
2.7.3 Warm-up temperature：100±20℃
2.7.4 Welding temperature：250±10℃
2.7.5 1 time pass wave crest welding machine

If disobey the above using terms, sensors sensitivity will be reduced.

LAMPIRAN C
(LAPORAN HASIL PENGAMBILAN DATA)

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 : Atik Sinawang / 2414031040

Jenis Pencemar : Sulfur Dioksida (SO_2)

Pengukuran	A D C	Standard (ppm)
1	687	0,06
2	700	0,07
3	722	0,08
4	784	0,11
5	810	0,17
6	910	0,35
7	964	0,52
8	1025	0,59
9	1051	0,68
10	1142	0,81
11	1243	0,98
12	1275	1,11

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

BIODATA PENULIS

Penulis dilahirkan di Nganjuk pada tanggal 09 Oktober 1995 dengan diberi nama Atik Sinawang Wahyuni. Bapak bernama Duril Hidayat, Ibu bernama Nikatul Shofiyah, dan kakak laki-laki bernama Anang Rosyadi. Penulis telah menyelesaikan studi di SDN I Sonoageng Nganjuk pada tahun 2008, SMP Negeri 1 Tanjunganom Nganjuk pada tahun 2011, SMA Negeri 7 Kediri pada tahun 2014, dan kemudian melanjutkan kuliah di Institut Teknologi Sepuluh Nopember (ITS), Departemen Teknik Instrumentasi, Program Studi D3 Teknik Instrumentasi pada tahun 2014. Pengalaman magang (*on job training* / kerja praktek) di PT. Aneka Gas Industri sier, Jawa Timur dengan judul: Studi Sistem Pengendalian *Flow* Pada Proses *Heating* Menggunakan *Butterfly Valve* Di *Moleculer Sieve Tower* Di PT. Aneka Gas Industri Sier, Jawa Timur. Bagi pembaca yang memiliki kritik, saran, atau ingin berdiskusi lebih lanjut mengenai tugas akhir ini, dapat menghubungi penulis melalui nomor telefon 085649350336 atau email atik.jewel04@gmail.com.

Motto hidup: *Do your best, and Allah will take care of the rest*