PERANCANGAN DAN PEMBUATAN SENSOR CURAH HUJAN TIPE *TIPPING BUCKET* DENGAN TAMPILAN LCD

SKRIPSI

Diajukan Untuk Memenuhi Persyaratan
Memperoleh Gelar Sarjana Teknik

Disusun oleh:
HENDRA DWI SAPUTRA
NIM. 105060309111001-63

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN
UNIVERSITAS BRAWIJAYA
FAKULTAS TEKNIK
MALANG
2013

LEMBAR PERSETUJUAN

PERANCANGAN DAN PEMBUATAN SENSOR CURAH HUJAN TIPE *TIPPING BUCKET* DENGAN TAMPILAN LCD

SKRIPSI

KONSENTRASI TEKNIK ELEKTRONIKA

Diajukan untuk memenuhi persyaratan
memperoleh gelar Sarjana Teknik

Disusun oleh:

HENDRA DWI SAPUTRA

NIM. 105060309111001

Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Ir. Nurussa'adah, MT	**Mochammad Rif'an, ST., MT**
NIP. 19680706 199203 2 001	**NIP. 19710301 200012 1 001**

LEMBAR PENGESAHAN

PERANCANGAN DAN PEMBUATAN SENSOR CURAH HUJAN TIPE *TIPPING BUCKET* DENGAN TAMPILAN LCD

SKRIPSI
KONSENTRASI TEKNIK ELEKTRONIKA

Diajukan untuk memenuhi persyaratan
memperoleh gelar Sarjana Teknik

Disusun oleh:

HENDRA DWI SAPUTRA

NIM. 105060309111001

Skripsi ini telah diuji dan dinyatakan lulus pada
tanggal 11 April 2013

Dosen Penguji

Ir. M. Julius St., M.S	**Ir. Ponco Siwindarto, M.Eng.Sc**
NIP. 19540720 198203 1 002	**NIP. 19590304 198903 1 001**

Dr-Ing. Onny S., S.T., M.T., M.Sc
NIP. 19740417 200003 2 007

Mengetahui,
Ketua Jurusan Teknik Elektro

Dr. Ir. Sholeh Hadi Pramono, M.S.
NIP. 19580728 198701 1 001

PENGANTAR

Puji dan syukur penulis panjatkan kepada Allah SWT, karena dengan rahmat, taufik dan hidayah-Nya lah skripsi ini dapat diselesaikan.

Skripsi berjudul "Perancangan Dan Pembuatan Sensor Curah Hujan Tipe *Tipping Bucket* Dengan Tampilan LCD" ini disusun untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Teknik di Jurusan Teknik Elektro Universitas Brawijaya.

Penulis menyadari bahwa penyusunan skripsi ini tidak terlepas dari bantuan berbagai pihak. Oleh karena itu, dengan ketulusan dan kerendahan hati penulis menyampaikan terima kasih kepada:

- Bapak, Ibu, dan Kakak selaku keluarga yang senantiasa memberikan semangat, doa, kasih sayang, perhatian, serta dukungan baik materi maupun non-materi yang tak ternilai yang telah diberikan.
- Bapak Dr. Ir. Sholeh Hadi Pramono, M.S. selaku Ketua Jurusan Teknik Elektro Universitas Brawijaya dan Bapak M. Aziz Muslim, ST., MT., Ph.D selaku Sekertaris Jurusan Teknik Elektro Universitas Brawijaya.
- Bapak Ir. M. Julius St, MS selaku Ketua Kelompok Dosen Keahlian Elektronika Jurusan Teknik Elektro Universitas Brawijaya atas segala bimbingan, nasehat, pengarahan, motivasi, saran, dan masukan yang telah diberikan.
- Ibu Ir. Nurussa'adah, MT. selaku Dosen Pembimbing 1 atas segala bimbingan, nasehat, pengarahan, motivasi, saran, dan masukan yang telah diberikan,
- dan Bapak Mochammad Rif'an, ST., MT. selaku Dosen Pembimbing 2 atas segala bimbingan, nasehat, pengarahan, motivasi, saran, dan masukan yang telah diberikan,
- Bapak dan Ibu dosen beserta staff dan karyawan Jurusan Teknik Elektro, baik secara langsung maupun tidak langsung telah membantu menyelesaikan skripsi ini,
- Semua teman SAP 2010, Suhartono, Siska Diah, Anas Amrullah, Henri Okta, Zikri Pramudia.
- Semua rekan-rekan di Jurusan Teknik Elektro Universitas Brawijaya Angkatan 2007 – 2010, Deni Satrio, Fikri Aulia, Tunggul Widyamurti, Bagus Ilyas, Aris Prima, Anas Setiawan, Nanang, Akhmad Sulkhan Taufik, Rifa, serta rekan-rekan seperjuangan yang lain yang tidak mungkin disebutkan satu persatu.

- Semua pihak yang tidak dapat penulis sebutkan satu persatu.

Pada akhirnya, penulis menyadari bahwa skripsi ini masih belum sempurna. Oleh karena itu, penulis sangat mengharapkan kritik dan saran yang membangun. Penulis berharap semoga skripsi ini dapat bermanfaat bagi pembangunan ilmu pengetahuan dan teknologi.

Malang, Maret 2013

Penulis

ABSTRAK

Hendra Dwi Saputra, Jurusan Teknik Elektro Fakultas Teknik Universitas Brawijaya, September 2013, Perancangan Dan Pembuatan Sensor Curah Hujan Tipe *Tipping Bucket* Dengan Tampilan LCD, Dosen Pembimbing: Ir. Nurussa'adah, MT. dan Mochammad Rif'an, ST., MT.

Pemanfaatan hujan yang tepat dapat memberikan keuntungan yang banyak. Untuk itu diperlukan pemetaan wilayah curah hujan. Hal ini bertujuan agar dapat menentukan tingkatan siaga bencana untuk masing-masing daerah. Dalam pemetaan dibutuhkan alat untuk menghitung curah hujan yang turun. Di pasaran hanya terdapat alat ukur secara manual, sedangkan tidak tersedia alat ukur curah hujan yang dapat mengukur secara otomatis dan langsung melainkan diperlukan peralatan tambahan secara terpisah. Selain itu hasil pengukuran tidak dapat dilihat oleh publik. Oleh karena itu dibutuhkan alat yang bekerja secara otomatis yang dapat menampilkan hasil pengukuran secara langsung dan dapat dilihat langsung oleh publik. Dengan demikian masyarakat dapat melihat dan mengukur curah hujan yang turun.

Pada perancangan dan pembuatan sensor curah hujan ini, sensor yang akan dibuat adalah tipe *tipping bucket* dengan resolusi pengukuran 0,5 mm. Hasil pengukuran ditampilkan melalui LCD Dot Matrix 2 x 16 karakter. Sensor yang dibuat dapat menyimpan data selama 24 jam. Selain itu, data hasil pengukuran dapat dikirm melalui komunikasi serial dan ditampilkan pada komputer, serta disimpan dalam bentuk *database*.

Kata Kunci: curah hujan, *tipping bucket*, resolusi, *database*

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perubahan iklim secara global dapat mengakibatkan perubahan musim yang signifikan baik secara lokal maupun regional. Hal ini dapat mengakibatkan sulitnya dalam memprediksi cuaca dan kapan terjadinya perubahan musim. Sebagai contoh adalah musim hujan di Indonesia yang kedatangannya selalu berubah dari tahun ketahun dan porsi musim hujan yang lebih panjang dibandingkan dengan musim kemarau. Kondisi ini dipengaruhi oleh wilayah Indonesia yang sebagian besar berupa laut dan berada disekitar wilayah katulistiwa. Kondisi tersebut mengakibatkan curah hujan di Indonesia yang tinggi yaitu berkisar antara 2000 sampai 3000 milimeter tiap tahunnya.

Hujan merupakan salah satu fenomena alam yang terdapat dalam siklus hidrologi dan sangat dipengaruhi iklim. Keberadaan hujan sangat penting dalam kehidupan, karena hujan dapat mencukupi kebutuhan air yang sangat dibutuhkan oleh semua makhluk hidup. Kehidupan di muka bumi akan terganggu jika tidak ada air. Namun disisi lain datangnya hujan dengan intensitas yang sangat tinggi yang tidak setimbang dengan kebutuhan akan terbuang percuma, bahkan dapat menyebabkan bencana. Oleh karena itu diperlukan pembangunan bangunan yang berfungsi mengendalikan dan mengurangi resiko bencana yang mungkin terjadi di musim hujan serta dapat menyimpan dan mengontrol kebutuhan penyediaan air saat musim kemarau.

Dari uraian singkat di atas disimpulkan akan pentingnya data curah hujan untuk mengatur pengelolaan air dalam memenuhi kebutuhan hidup manusia. Mengingat curah hujan antara daerah satu dengan daerah lainnya berbeda-beda dan dapat terjadi setiap saat, oleh karena itu diperlukan alat yang dapat memantau curah hujan secara otomatis, *realtime*, dan mampu menyimpan data curah hujan di masing-masing daerah. Namun pada kenyataannya alat ukur curah hujan yang terdapat di pasaran dijual secara terpisah, dan masih bekerja secara manual serta tidak dapat menyimpan secara otomatis. Hal ini tentu sangat tidak efisien saat digunakan dan dioperasikan. Oleh karena itu diperlukan alat ukur curah hujan yang bekerja secara otomatis dan dapat menyimpan data curah hujan yang turun ke dalam sebuah *database* pada komputer, sehingga data curah hujan yang dihasilkan dapat dimanfaatkan secara optimal.

Standar alat ukur curah hujan tipe *tipping bucket* dapat mengukur dengan resolusi terkecil 0,01 inchi atau sekitar 0,2 mm setiap jam (weathershack.com). Selain itu, diharapkan resolusi dapat diatur sesuai dengan kebutuhan. Fitur ini sangat diperlukan agar alat lebih fleksibel mengingat karakteristik hujan antara daerah satu dengan daerah lain berbeda-beda.

Penelitian yang telah dilakukan sebelumnya oleh Ike Kusuma Dewi (2005), Mohammad Syarief (2006), dan Erdy Prasetiya Kusuma (2007) masih memiliki kelemahan. Alat yang dibuat memerulukan komputer dan bangunan permanen untuk menjaga keamanan alat pada lokasi pemantauan curah hujan. Sehingga diperlukan biaya yang lebih mahal dalam penerapannya. Selain itu penggunaan jalur internet tidak terlalu efektif untuk daerah terpencil yang tidak terdapat jaringan internet.

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, dapat disusun rumusan masalah sebagai berikut:

1) Bagaimana merancang dan membuat sistem elektronika dan pengkondisi sinyal untuk membaca gerakan mekanik sensor curah hujan agar dapat dideteksi oleh mikrokontroler.
2) Bagaimana merancang dan membuat perangkat lunak sistem mikrokontroler sebagai pembaca keluaran sensor, pengolah dan penyimpan data sementara, serta menampilkan pada LCD secara *realtime*.
3) Bagaimana merancang format penyimpanan data agar data tidak saling tumpang tindih saat dibaca oleh komputer.
4) Bagaimana merancang sistem pengiriman data dari mikrokontroler menuju komputer agar data hasil pembacaan pada komputer tidak terjadi kesalahan.

1.3 Batasan Masalah

Dengan mengacu pada permasalahan yang telah dirumuskan, maka hal-hal yang berkaitan dengan alat diberi batasan sebagai berikut:

1) Parameter yang diukur adalah curah hujan (air).
2) Sensor curah hujan yang digunakan tipe *Tipping Bucket*.
3) Resolusi pengukuran *Tipping Bucket* sebesar 0,5 mm.

4) Mikrokontroler yang digunakan ATMega8535.
5) Media transfer data antara alat dengan komputer menggunakan komunikasi serial RS232.
6) *Database* yang digunakan Microsoft Access.

1.4 Tujuan

Tujuan penelitian ini adalah merancang dan membuat sensor curah hujan jenis *tipping bucket* dengan tampilan LCD dan data hasil pemantauan dapat ditransfer ke komputer melalui komunikasi serial, sehingga pengukuran curah hujan dapat dilakukan dengan lebih mudah, akurat, dan hasil pemantauan dapat disimpan dalam *database*.

1.5 Sistematika Penulisan

Sistematika penulisan dalam laporan skripsi ini adalah:

BAB I Pendahuluan

Memuat latar belakang, rumusan masalah, batasan masalah, tujuan, dan sistematika penulisan.

BAB II Tinjauan Pustaka

Membahas teori-teori yang mendukung dalam perencanaan dan pembuatan alat.

BAB III Metodologi Penelitian

Berisi tentang metode-metode yang dipakai dalam melakukan perancangan, pengujian, dan analisis data.

BAB IV Perencanaan dan Pembuatan Alat

Perancangan dan perealisasian alat yang meliputi spesifikasi, perencanaan diagram blok, prinsip kerja, dan realisasi alat.

BAB V Pengujian Alat

Memuat aspek pengujian meliputi penjelasan tentang cara pengujian dan hasil pengujian. Aspek analisis meliputi penilaian atau komentar terhadap hasil-hasil pengujian. Pengujian dan analisis ini terhadap alat yang telah direalisasikan berdasarkan masing-masing blok dan sistem secara keseluruhan.

BAB VI Kesimpulan dan Saran

Memuat intisari hasil pengujian dan menjawab rumusan masalah serta memberikan rekomendasi untuk perbaikan kualitas penelitian di masa akan datang.

BAB II
TINJAUAN PUSTAKA

Dalam merencanakan dan merealisasikan alat pemantau curah hujan dibutuhkan pemahaman tentang berbagai hal yang mendukung dalam pembuatan. Pemahaman ini akan bermanfaat untuk merancang dan merealisasikan alat meliputi dasar teori tentang hujan, *tipping bucket*, *optocoupler*, mikrokontroler, komunikasi serial sinkron dan asinkron, komunikasi I^2C, komunikasi Serial RS232, *Real Time Clock (RTC)* dan LCD.

2.1 Hujan

Hujan adalah titik-titik air yg berjatuhan dari udara karena proses pendinginan. Butir-butir hujan mempunyai garis tengah 0,08 - 6 mm. Hujan terdapat dalam beberapa macam yaitu hujan gerimis, hujan sedang, hujan deras dan hujan badai. Perbedaan terutama pada besarnya butir-butir. Hujan lebat biasanya turun sebentar saja jatuh dari awan *cumulonimbus*. Hujan semacam ini mempunyai intensitas yang besar.

Satuan curah hujan yang umum digunakan oleh Badan Meteorologi, Klimatologi dan Geofisika adalah millimeter per jam ($^{mm}/_{jam}$). Jadi jumlah curah hujan yang diukur sebenarnya adalah tebal atau tingginya permukaan air hujan yang menutupi suatu area di permukaan bumi sebelum mengalami aliran permukaan, evaporasi, dan peresapan ke dalam tanah. Luasan yang tercakup oleh sebuah penakar hujan bergantung pada homogenitas daerahnya maupun kondisi cuaca lainnya. Curah hujan 1 mm artinya dalam area 1 m^2 (1 meter persegi) pada tempat yang datar tertampung air setinggi 1 mm atau tertampung sebanyak 1 liter atau 1000 ml. Jenis-jenis hujan berdasarkan besarnya curah hujan menurut BMKG dibagi menjadi tiga, yaitu :

1) Gerimis — ketika tingkat presipitasinya < 2,5 $^{mm}/_{jam}$
2) Hujan sedang — ketika tingkat presipitasinya antara 2,5 $^{mm}/_{jam}$ – 10 $^{mm}/_{jam}$
3) Hujan deras — ketika tingkat presipitasinya > 10 $^{mm}/_{jam}$ – 50 $^{mm}/_{jam}$
4) Hujan badai — ketika tingkat presipitasinya > 50 $^{mm}/_{jam}$

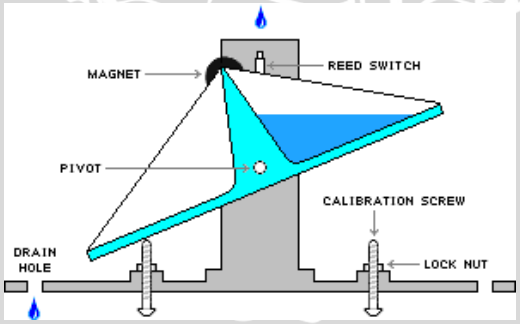
2.2 Penakar Curah Hujan *Tipping Bucket*

Penakar curah hujan adalah instrumen yang digunakan untuk mendapatkan dan mengukur jumlah curah hujan pada satuan waktu tertentu. Penakar hujan mengukur

tinggi hujan seolah-olah air hujan yang jatuh ke tanah menumpuk ke atas merupakan kolom air. Air yang tertampung *volume*-nya dibagi dengan luas corong penampung, hasilnya adalah tinggi atau tebal, satuan yang dipakai adalah milimeter (mm).

Sensor yang dipakai untuk mengukur besarnya curah hujan adalah *rain gauge*. Jenis *rain gauge* bermacam-macam, ada sekitar 50 jenis *rain gauge* yang memenuhi *standard* internasional. Salah satunya adalah jenis *tipping bucket*.

Tipping bucket sensor bekerja dengan cara menghitung pulsa persatuan waktu yang ditentukan dari banyaknya air yang masuk ke dalam corong atau wadah penampung *sensor* tersebut. Sehingga dari pulsa-pulsa tersebut dapat diketahui besarnya curah hujan persatuan luas persatuan waktu. Air hujan ditampung ke dalam bejana yang berjungkit. Bila air mengisi bejana penampung yang setara dengan tinggi hujan 0,5 mm atau sesuai dengan spesifikasi *sensor* akan berjungkit dan air dikeluarkan. Terdapat dua buah bejana yang saling bergantian menampung air hujan. Tiap gerakan bejana berjungkit secara mekanis tercatat pada pias atau menggerakkan *counter* (penghitung). Jumlah hitungan dikalikan dengan 0,5 mm (sesuai dengan spesifikasi *sensor*) merupakan tinggi hujan yang terjadi. Ketika bejana berjungkit, akan menggerakkan saklar (seperti *reed switch*) yang kemudian direkam secara elektronik. Cara kerja alat penakar hujan ditunjukkan dalam Gambar 2.1.



Gambar 2.1 Bagian-bagian *Tipping Bucket*

Sumber: Weathershack

Tipping Bucket memiliki keuntungan dan kelebihan dibandingkan dengan sensor curah hujan yang lain. Keuntungan dari alat pengukur hujan tipe *tipping bucket* adalah karakter dari hujan (ringan, sedang atau berat) dapat dengan mudah diperoleh. Karakter hujan ditentukan oleh jumlah hujan yang turun dalam beberapa waktu (biasanya 1 jam) serta dengan menghitung jumlah jungkitan dalam jangka waktu 10 menit pengamat dapat menentukan karakter dari hujan. Sedangkan kelemahan dari *Tipping Bucket*

tidaklah seteliti instrumen standar lainnya, dikarenakan hujan dapat saja berhenti sebelum bejana berjungkit karena curah hujan belum mencapai nilai 0.5 mm. Sehingga nilai curah hujan di bawah 0,5 mm tidak tercatat. Contoh penakar hujan jenis *tipping bucket* yang terpasang di lapangan ditunjukkan dalam Gambar 2.2

Gambar 2.2 *Tipping Bucket*
Sumber: Okta Veanti, 2011

Kalibrasi pada *tipping bucket sensor* dilakukan dengan cara mengatur keseimbangan jungkitan dengan merubah ketinggian baut penahan jungkitan tersebut. Untuk mendapatkan *volume* yang tertampung dalam curah hujan diperoleh dari luas penampang corong pada *tipping bucket* dikalikan dengan tinggi curah hujan yang diinginkan. Misalnya diameter corong tabung 20 cm dan ketinggian curah hujan yang diinginkan 0.2 mm maka untuk mendapatkan *volume* pada setiap jungkitan dihitung dengan cara :

$V = L \times h$ (2-1)

$V = \pi \times r^2 \times 0{,}2 \text{ mm}$

$= 3{,}14 \times (10 \text{ cm})^2 \times 0{,}02 \text{ cm}$

$= 6{,}28 \text{ cm}^3$

Keterangan:

V = Volume air maksimal yang tertampung

L = Luas penampang *tipping bucket* yang berbentuk lingkaran

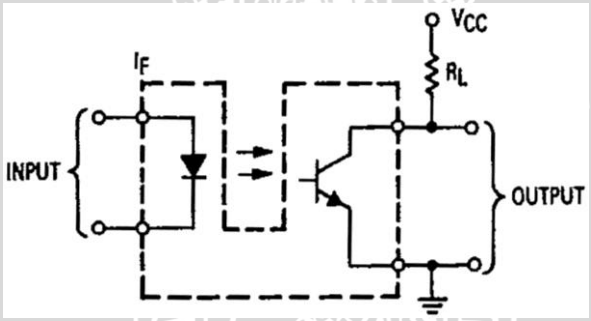
h = Resolusi yang diharapkan

r = Panjang jari-jari penampang *tipping bucket*

Sehingga untuk sekali gerakan, air yang tertampung di dalam bejana penampung sebanyak 6,28 cm^3.

2.3 *Optocoupler*

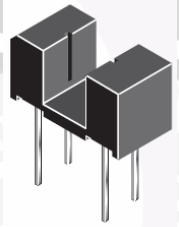
Optocoupler adalah suatu komponen penghubung atau *coupling* yang bekerja berdasarkan picu cahaya optik. *Optocoupler* terdiri dari 2 bagian yang saling berhadapan yaitu *transmitter* atau bagian sumber cahaya dan *receiver* atau bagian deteksi sumber cahaya. Pada bagian transmitter terbuat dari komponen LED infrared. Jika dibandingkan dengan LED biasa, LED infrared memiliki ketahanan yang lebih baik terhadap sinyal tampak. Cahaya yang dipancarkan oleh LED infrared tidak terlihat oleh mata telanjang. Sedangkan pada bagian receiver terbuat dari komponen fototransistor. Hal ini disebabkan fototransistor merupakan suatu transistor yang peka dan respon cepat terhadap cahaya inframerah. Bagian *optocoupler* ditunjukkan dalam Gambar 2.3.



Gambar 2.3 Bagian-bagian *Optocoupler*
Sumber: Fairchild, 2009:1

Prinsip kerja *optocoupler* adalah :

- Jika antara bagian pengirim mati atau cahaya menuju fototransistor terhalang maka fototransistor tersebut akan *off* sehingga antara kolektor dan emitor tidak terhubung.
- Sebaliknya jika antara bagian pengirim dan penerima tidak terhalang maka fototransistor tersebut akan *on* sehingga antara kolektor dan emitor akan terhubung.



Gambar 2.4 Bentuk Fisik *Optocoupler*
Sumber: Fairchild, 2009:1

2.4 Mikrokontroler ATMega8535

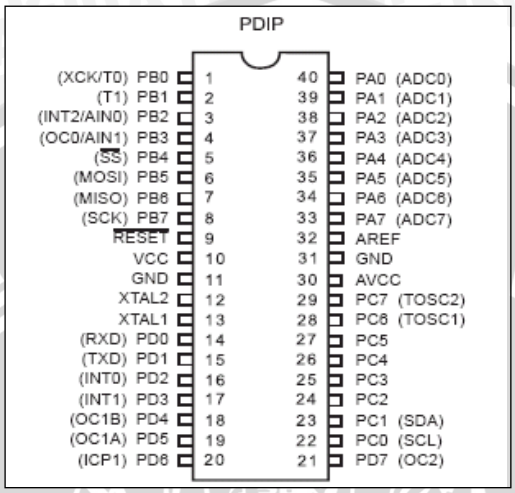
Secara umum, mikrokontroler berfungsi sama dengan komputer. Bedanya adalah mikrokontroler memiliki desain dalam sebuah *single chip* (IC). Mikrokontroler terdapat di hampir semua peralatan elektronik di sekeliling kita, TV, radio, telepon genggam *(Hand Phon*e) dll. Mikrokontroler memiliki kemampuan yang diperlukan untuk membuat keputusan berdasarkan sinyal dari luar dengan kata lain mikrokontroler merupakan otak dari sebuah perangkat elektronik.

ATmega8535 merupakan salah satu mikrokontroler dari buatan ATMEL keluarga ATMEGA yang mempunyai 8 kbyte *Flash* PEROM (*Flash Programmable and Erasable Read Only Memory*), 512 byte SRAM, 32 pin I/O (4 buah *port* I/O bit) yang mana tiap pin tersebut dapat diprogram secara paralel dan tersendiri, mempunyai dua buah *timer/counter* 8 bit dan satu buah *timer/counter* 16 bit, mempunyai 10 bit 8 *channel* ADC, mempunyai *watchdog timer*.

Sebagai suatu sistem kontrol mikrokontroler ATmega8535 bila dibandingkan dengan mikroprosesor memiliki kemampuan dan segi ekonomis yang bisa diandalkan karena dalam mikrokontroler sudah terdapat RAM dan ROM sedangkan mikroprosesor didalamnya tidak terdapat keduanya. Secara umum konfigurasi yang dimiliki mikrokontroler ATmega8535 adalah sebagai berikut :

- Sebuah CPU 8 bit dengan menggunakan teknologi dari Atmel.
- Memiliki memori baca-tulis sebesar 512 byte SRAM.
- Jalur dua arah (*bidirectional*) yang digunakan sebagai saluran masukan atau keluaran yang dikontrol oleh register DDR.
- Sebuah komunikasi serial USART yang dapat diprogram.
- Sebuah *master/slave serial* SPI yang dapat diprogram.
- *Sebuah Two Wire Serial Interface*.
- Dua buah *timer/counter* 8 bit dan sebuah *timer/counter* 16 bit.
- *Watcdog Timer* yang dapat diprogram.
- *Analog comparator* di dalam *chip*.
- *Osilator internal* dan rangkaian pewaktu.
- Flash PEROM yang besarnya 8 kbyte untuk menyimpan program
- Kemampuan melaksanakan operasi perkalian, pembagian, dan operasi *Boolean*.
- Mampu beroperasi sampai 16 MHz.

Masing-masing kaki dalam mikrokontroler ATmega8535 mempunyai fungsi tersendiri. Dengan mengetahui fungsi masing-masing kaki mikrokontroler ATmega8535, perancangan aplikasi mikrokontroler ATmega8535 akan lebih mudah. ATmega8535 mempunyai 40 pin, susunan masing-masing pin ditunjukkan dalam Gambar 2.5.



Gambar 2.5 Konfigurasi Pin ATmega8535
Sumber: Atmel, 2005:2

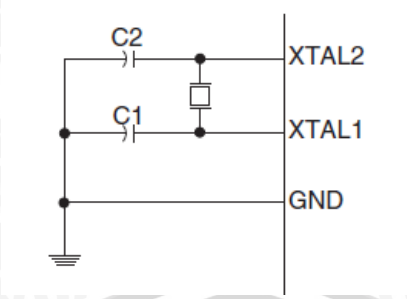
Fungsi kaki-kaki ATmega8535 adalah :

- *Port* A (Pin A0..7), merupakan saluran masukan/keluaran dua arah dan juga mempunyai fungsi khusus. Fungsi khusus *Port* A adalah sebagai ADC (*input* ADC channel 0..7).
- *Port* B (Pin B0..7), merupakan saluran masukkan/keluaran dua arah dan juga mempunyai fungsi khusus. Fungsi khusus Port B diantaranya adalah : *Port* B0 (T0 (*timer/counter0* eksternal *counter input*) & XCK (USART eksternal *clock input*/*output*), *Port* B1 (T1 (*timer/counter* eksternel *counter input*)), *Port* B2 (AIN0 (Analog comparator positive *input*) & INT2 (Eksternal interrupt 2 *input*)), *Port* B3 (AIN0 (Analog comparator negative *input*) & OC0 (*Timer/counter0 output compare match output*)), *Port* B4 (SS (*SPI slave select input*)), *Port* B5 (MOSI (*SPI bus master output/slave input*)), *Port* B6 (MISO (*SPI bus master input/slave output*)), *Port* B7 (SCK (SPI *bus serial clock*)).
- *Port* C (Pin C0..7), merupakan saluran masukkan/keluaran dua arah dan juga mempunyai fungsi khusus. Fungsi khusus dari *Port* C diantaranya adalah : *Port* C0

(SCL (*Two-Wire serial bus clock line*)), *Port* C1 (SDA (*Two-Wire serial bus data input/output line*)), *Port* C6 (TOSC1 (*Timer Oscilator* pin1)), *Port* C7 (TOSC2 (*Timer oscillator* pin2)).

- *Port* D (Pin D0..7), merupakan saluran masukkan/keluaran dua arah dan juga mempunyai fungsi khusus. Fungsi khusus dari *Port* D diantaranya adalah : *Port* D0 (RXD (USART *input* pin)), *Port* D1 (TXD (USART *output* pin)), *Port* D2 (INT0 (Eksternal interupt 0 *input*)), *Port* D3 (INT1 (Eksternal interupt 1 *input*)), *Port* D4 (OC1B (*Timer/counter* 1 *output* compare B match *output*)), *Port* D5 (OC1A (*Timer/counter* 1 *output* compare A match *output*)), *Port* D6 (ICP (*Timer/counter input capture* pin)), *Port* D7 (OC2 (*timer/counter 2 compare match output*)).
- Pin 9 RESET, merupakan saluran dua masukan untuk mereset mikrokontroler dengan cara memberi masukan logika rendah.
- Pin 10 VCC, merupakan saluran masukan untuk catu daya positif sebesar 5 volt DC.
- Pin 11 GND, merupakan *Ground* dari seluruh rangkaian.
- Pin 12 dan 13(XTAL2 dan XTAL1), merupakan saluran untuk mengatur pewaktuan sistem. Untuk pewaktuan dapat dapat menggunakan pewaktuan internal maupun eksternal.
- Pin 32 AREF, merupakan Pin analog referensi untuk masukan ADC.
- Pin 33 GND, merupakan ground dari ADC.
- Pin 34 AVCC, merupakan *supply* untuk *port* A dan juga merupakan *supply* untuk ADC.

Mikrokontroler ATMega8535 memiliki osilator internal (*on chip oscillator*) yang dapat digunakan sebagai sumber *clock* bagi MCU. Untuk menggunakan osilator internal diperlukan sebuah kristal antara pin XTAL 1 dan pin XTAL 2 serta dua buah kapasitor ke *ground.* Komponen kristal dapat digunakan frekuensi dari 0 sampai 33 MHz, sedangkan untuk kapasitor dapat bernilai 12 pF ± 22 pF. Rangkaian osilator eksternal ditunjukkan dalam Gambar 2.6.

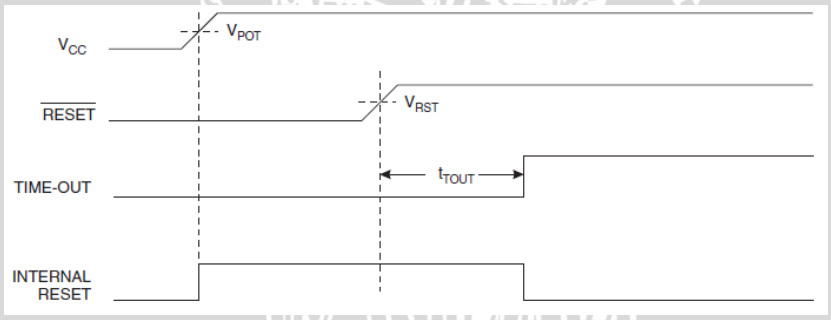


Gambar 2.6 Rangkaian Osilator Mikrokontroler

Sumber: Atmel, 2005:24

Selain membutuhkan rangkaian osilator, mirkorontroler juga memerlukan rangkaian reset. Rangkaian *power on reset* diperlukan untuk m*ereset* mikrokontroler secara otomatis setiap catu daya dinyalakan. Rangkaian ini juga berfungsi untuk mereset secara otomatis pada saat tegangan catu di bawah tegangan *Power-on Reset threshold* (V_{POT}).

Ketika catu daya diaktifkan, rangkaian *power on reset* menahan logika rendah pin RST dengan jangka waktu yang ditentukan oleh lamanya pengisian muatan C, seperti ditunjukkan dalam Gambar 2.7.



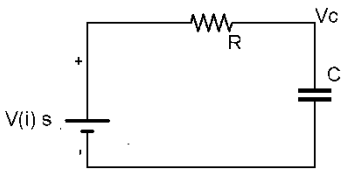
Gambar 2.7 Mikrokontroler *Start-up* dengan *Power On Reset*

Sumber: Atmel, 2005:36

Rangkaian *power on reset* ditunjukkan dalam Gambar 2.8.

Gambar 2.8 Rangkaian *Power On Reset*

Rangkaian ekuivalen *power on reset* ditunjukkan dalam Gambar 2.9.



Gambar 2.9 Rangkaian Ekivalen *Power On Reset*

Dari Gambar 2.9 diperoleh:

$$Vc(s) = \frac{\frac{1}{Cs}}{R + \frac{1}{Cs}} \cdot Vi(s) = \frac{1}{RCs + 1} \cdot Vi(s) \qquad (2\text{-}2)$$

Dengan tegangan V_i adalah tegangan VCC yaitu 5V, dalam fungsi *Laplace* adalah 5/s, sehingga:

$$Vc = \frac{1}{RCs + 1} \times \frac{5}{s} = \frac{\frac{1}{RC}}{1 + \frac{1}{RC}} \times \frac{5}{s} = \frac{5}{s} - \frac{5}{s + \frac{1}{RC}}$$

$$Vc = 5 - 5e^{-\frac{t}{RC}} \qquad (2\text{-}3)$$

$$Vc - 5 = -5e^{-\frac{t}{RC}}$$

$$\frac{5}{5 - Vc} = e^{-\frac{t}{RC}} \rightarrow ln \ \frac{5}{5 - Vc} = \frac{t}{RC}$$

maka:

$$t = R \cdot C \cdot \left(\ln \frac{5}{5 - Vc} \right), \qquad \text{atau} \qquad t = R \cdot C \cdot \frac{\left(\log \frac{5}{5 - Vc} \right)}{\log e} \qquad (2\text{-}4)$$

Dengan nilai $V_{C(MAX)}$ = 2,3 V adalah tegangan logika nominal yang diijinkan oleh pin RST ATMega8535, maka:

$$t = R \cdot C \cdot \frac{\left(\log \frac{5}{2,7} \right)}{\log e}$$, sehingga

$$t = 0,616 \cdot R \cdot C \qquad (2\text{-}5)$$

Keterangan:

V_C = Tegangan keluaran (V)

V_i = Tegangan masukan (V)
t = Waktu pengisian kapasitor (s)
R = Nilai resistor yang dipergunakan (Ω)
C = Nilai kapasitor yang dipergunakan (F)

2.5 Komunikasi Serial

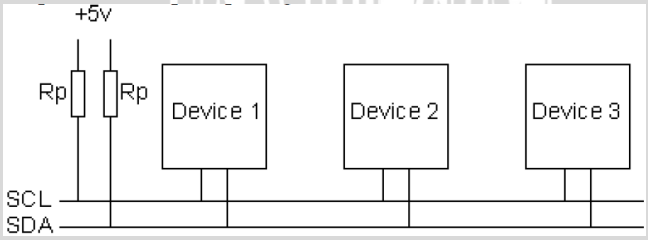
Komunikasi serial adalah salah satu metode komunikasi data di mana hanya satu bit data yang dikirimkan melalui satu jalur kabel pada suatu waktu tertentu. Pada dasarnya komunikasi serial adalah kasus khusus komunikasi paralel dengan nilai n = 1, atau dengan kata lain adalah suatu bentuk komunikasi paralel dengan jumlah kabel hanya satu dan hanya mengirimkan satu bit data secara simultan.

Ada dua jenis komunikasi pada serial yaitu *sinkron* dan *asinkron*. Komunikasi bertipe *sinkron* merupakan sebuah tipe komunikasi serial yang selalu mengirimkan karakter atau penanda setiap selang waktu tertentu untuk menjaga sinkronisasi antara dua alat yang saling berkomunikasi. Pada transmisi sinkron, sebelum terjadi komunikasi, diadakan sinkronisasi *clock* antara pengirim dan penerima. Data dikirim dalam satu blok data (disebut *frame*) yang berisi bit pembuka (*preamble bit*), bit data itu sendiri, dan bit penutup (*postamble bit*). Contoh komunikasi sinkron adalah I^2C yang dikembangkan oleh Philip. Sedangkan komunikasi *asinkron* merupakan suatu komunikasi serial yang tidak membutuhkan *sinkronisasi* untuk melakukan komunikasi antara dua buah alat yang berkomunikasi. Tetapi komunikasi tersebut membutuhkan karakter *idle* yang harus dikirim atau diterima setiap pengiriman satu karakter. Untuk mengirimkan sebuah karakter dalam komunikasi asinkron pada awal dan akhir dari byte data harus ditambahkan *start* dan *stop* bit. *Start* bit menandakan bahwa data tersebut akan mulai dikirim sedangkan *stop* bit menandakan bahwa data tersebut telah berakhir. Komunikasi *sinkron* memungkinkan untuk transfer data yang lebih cepat daripada komunikasi *asinkron* karena pada komunikasi *sinkron* tidak membutuhkan tambahan bit pada awal dan akhir untuk setiap karakter. Serial *port* pada IBM PC adalah *asinkron* karena itu komunikasi yang bisa dilakukan adalah komunikasi serial *asinkron*.

2.5.1 I^{2}C

I^{2}C merupakan singkat dari *Inter Intergrated Circuit,* yang disebut dengan *I-squared-C* atau *I-two-C.* I^{2}C merupakan protocol yang digunakan pada *multi-master serial computer bus* yang diciptakan oleh Philips yang digunakan untuk saling berkomunikasi dengan perangkat *low-speed* lainnya yang diaplikasikan pada *motherboard, embedded system,* atau *cellphone.*

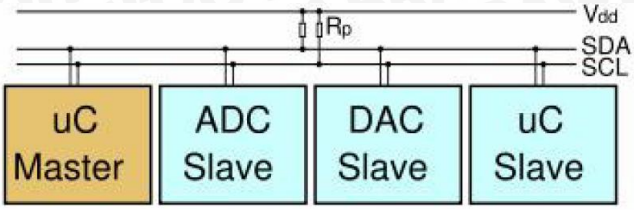
Jalur I^{2}C bus hanya merupakan 2 jalur untuk *clock* dan SDA *line* merupakan jalur untuk data. Semua peralatan yang akan digunakan dihubungkan selurunya pada jalur SDA dan SCL *line* dari I^{2}C bus tersebut. Jenis komunikasi sifat *serial synchronuous half duplex bidirectional,* dimana data yang ditransmisikan dan diterima hanya melalui satu jalur dari SDA *line* (bersifat serial), setiap pengguna jalur data bergantian antar perangkat (bersifat *half duplex*) dan data dapat ditransmisikan dari dan kesebuah perangkat (bersifat *bidirectional*). Sumber *clock* yang digunakan pada I^{2}C bus hanya berasal dari satu perangkat *master* melalui jalur *clock* SCL *line* (bersifat *sychronuous*). Kedua jalur SDA dan SCL merupakan driver yang bersifat "*open drain*", yang berarti IC yang digunakan dapan mendrive *output*-nya *low*, tetapi tidak dapat men-*drive* menjadi *high*. Untuk dapat mendapatkan data yang *high* maka kita harus menyediakan resistor *pull-up* pada tegangan *supply* sebesar 5 volt terhadap jalur SDA dan SCL tersebut. Kita hanya membutuhkan satu set *pull-up* resistor untuk semua jalur I^{2}C bus, tidak untuk semua perangkat yang kita gunakan, pemasangan resistor *pull-up* ditunjukkan dalam Gambar 2.10.



Gambar 2.10 Pemasangan Resistor *Pull-Up*

Sumber: lab.binus.ac.id, 2007: 1

Semua perangkat yang terdapat dalam jalur I^{2}C bus merupakan perangkat *slave* dan *master,* dimana *master* merupakan perangkat yang berfungsi sebagai pengatur (*controller*) dan sumber *clock* bagi perangkat-perangkat *slave* yang terdapat dalam jalur I^{2}C *bus*.



Gambar 2.11 Jaringan I^2C

Sumber: lab.binus.ac.id, 2007: 1

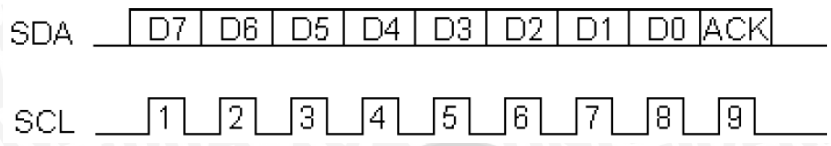
Kecepatan transfer data dari protocol I^2C ditentukan oleh besar *clock speed* yang digunakan pada jalur SCL *line*. Kecepatan *clock* standar yang diberikan pada jalur SCL *line* pada I^2C sebesar 100 KHz. Philips sebagai pencipta protokol I^2C ini membuat standar kecepatan I^2C lainnya yaitu *Fast Mode* yang mempunyai kecepatan *clock* sebesar 400 KHz dan *High Speed Mode* yang mempunyai kecepatan hingga 3.4 MHz. Untuk melakukan transmisi data pada sebuah jalur I^2C bus, dimulai dengan mengirimkan sebuah *start sequence* dan diakhiri dengan mengirim *stop sequence*. *Start sequence* dan *stop sequence* menandakan awal dan akhir dari proses transmisi data dengan perangkat yang lainnya dalam sebuah jalur I^2C. *Start* dan *stop sequence* dalam komunikasi I^2C ditunjukkan dalam Gambar 2.12.

Gambar 2.12 Timming Diagram *Start* dan *Stop*

Sumber: lab.binus.ac.id, 2007: 2

Transmisi data antar perangkat terjadi setelah *start sequence* dan sebelum *stop sequence*. Data yang ditransmisikan sejumlah 8 bit dengan MSB (*Most Significant Bit*) yang dikirim terlebih dahulu hingga kepada LSB (*Least Significant Bit*) kemudian selalu terdapat tambahan satu bit yang merupakan *Acknowledgement* bit (ACK bit). ACK bit digunakan untuk mengetahui kondisi transmisi data, jika ACK bit berupa kondisi *low* maka perangkat yang ada sudah menerima data dan siap untuk menerima data yang selanjutnya, sedangkan ACK bit berupa kondisi *high* maka perangkat yang ada sudah tidak dapat melakukan transmisi data dan *master* harus mengirim *stop sequence* untuk menghentikan komunikasi yang sedang berlangsung. Pada saat berlangsung komunikasi

antar perangkat dalam sebuah jalur I^2C bus, bit data dikirimkan pada saat jalur SCL *line* dalam kondisi *low* ditunjukkan dalam Gambar 2.13.



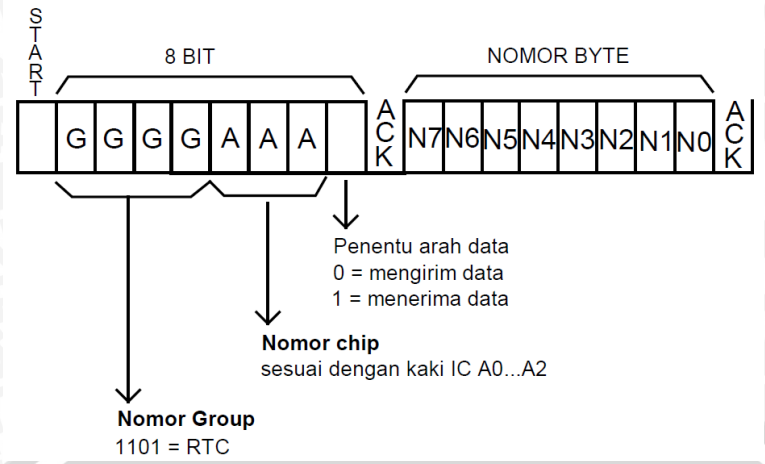
Gambar 2.13 Timming Diagram Saat Pengiriman Data

Sumber: lab.binus, 2007: 2

Pada sebuah jalur I^2C bus ditunjukkan untuk mengendalikan beberapa perangkat *slave* dengan menggunakan sebuah perangkat *master*. Setiap perangkat *slave* pada jalur I^2C bus masing-masing mempunyai alamat I^2C sebesar 7 bit alamat, sehigga pada sebuah jalur I^2C bus dapat digunakan perangkat *slave* sebanyak 2^7 perangkat dengan alamat yang berkisar antara 0 sampai dengan 127. Pada saat mengirimkan 7 bit alamat sebuah perangkat *slave* kita selalu mengirim 8 bit data, yaitu bit alamat + 1 bit R/W. Bit alamat itu sendiri (terdiri dari 7 bit) terbagi menjadi 2 bagian, yaitu:

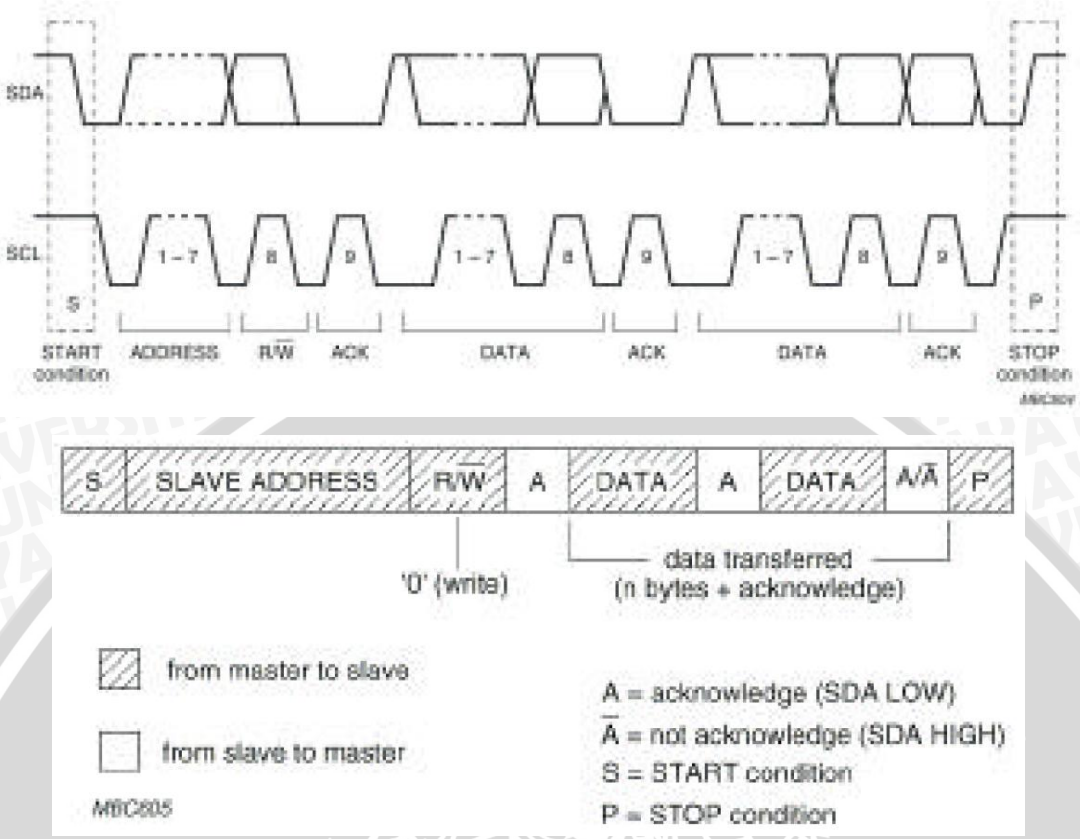
1) Nomor Group (G) adalah nomor yang diberikan oleh Philips (sebagai pencipta I2C) pada kelompok-kelompok IC I2C. Sebagai contoh nomor group untuk Serial *RTC* adalah 1101 (biner).
2) Nomor Chip (C) adalah nomor yang diberikan pada masing-masing *chip* lewat kaki *A0,A1* dan *A2* dari masing-masing *IC*. Dalam *IC* I2C tertentu, *A0...A2* tidak dihubungkan ke kaki IC, tapi dipakai didalam *IC* untuk menomori register/memori di dalam *IC* bersangkutan. DS1307 tidak memiliki A0, A1, dan A2.

Bit R/W (*Read*/*Write*) digunakan untuk memberitahukan kepada perangkat *slave* yang dipanggil tindakan yang akan dilakukan perangkat *master* pada perangkat *slave* tersebut, dimana *Read* → perangkat *master* akan melakukan pembacaan data dari perangkat *slave* tersebut dan *Write* → perangkat *master* akan melakukan pengiriman data pada perangkat *slave* tersebut. Untuk melakukan *Read* maka pada bit R/W diberikan kondisi logic *high*, sedangkan untuk melakukan *Write* maka pada bit R/W diberikan konsisi logic *low*. Pengiriman alamat perangkat *slave* pada sebuah sequence protokol I^2C ditunjukkan dalam Gambar 2.14.



Gambar 2.14 Format Data Pengalamatan *Slave*
Sumber: Agus Pracoyo, 2007: 93

Sebuah perangkat *master* dapat melakukan dua tindakan pada perangkat *slave* yang terhubung dalam sebuah jalur I^2C bus, yaitu melakukan *read* dari perangkat *slave* atau melakukan *write* ke perangkat *slave*. Penggunaan protokol untuk melakukan *read* atau *write* sedikit berbeda, namun untuk memulai komunikasi antara *master* dan *slave* sama untuk *write* atau *read*, yaitu dengan mengirimkan *start sequence* kemudian alamat perangkat *slave* yang dituju dan tindakan yangakan dilakukan *write* atau *read*, dan pada akhirnya mengirimkan *stop sequence* untuk menyelesaikan komunikasi yang sedang berlangsung. Komunikasi yang terjadi pada protokol I^2C secara lengkap ditunjukkan dalam Gambar 2.15.



Gambar 2.15 Protokol I^2C

Sumber: lab.binus.ac.id, 2007: 3

2.5.2 Komunikasi Serial RS-232

Komunikasi serial merupakan suatu proses transfer data yang memanfaatkan *port* yang sudah tersedia pada komputer yaitu Port DB9. Sehingga tidak memerlukan *hardware* lain selain konektor dan kabel data. Serial mereferensikan untuk metransfer data satu bit tiap waktu, dimana setiap bit adalah on atau off.

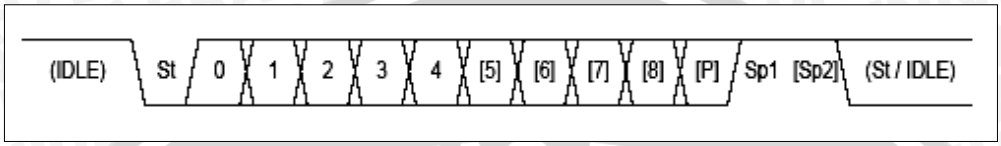
RS-232 merupakan salah satu jenis antar muka *(interface)* dalam proses transfer data antar komputer dalam bentuk serial. RS-232 merupakan kependekan dari *Recommended Standart Number* 232. RS-232 dibuat untuk *interface* antara peralatan terminal data dan peralatan komunikasi data, dengan menggunakan data biner serial sebagai data yang ditransmisikan. Serial *interface* RS-232 memberi ketentuan *logic level* sebagai berikut:

Logika 1 disebut "*mark*" terletak antara –3 Volt hingga –15 Volt.

Logika 0 disebut "*space*" terletak antara +3 Volt hingga +15 Volt.

Daerah tegangan antara -3Volt hingga +3Volt adalah *invaled level*, yaitu daerah tegangan yang tidak memiliki logika sehingga daerah itu harus dihindari. Suatu saluran data RS-232 yang terdapat tegangan ini berarti ada kesalahan. Demikian pula pada saluran pada daerah lebih negatif dari -15 Volt daerah lebih positif dari +15 Volt.

Protokol komunikasi serial dalam pengiriman data ditunjukkan dalam Gambar 6.16.



Gambar 2.16 Format Data Pengiriman
Sumber: Atmel, 2006:134

Keterangan gambar :

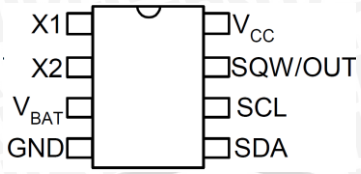
St = Bit *start* selalu berlogika rendah
(n) = Banyaknya data yang dikirim (0-8)
P = Bit paritas (ganjil atau genap)
Sp = Bit *stop* selalu berlogika tinggi (bit *stop* bisa berjumlah 1 atau 2
IDLE = Tidak ada data yang ditransfer pada RX dan TX, IDLE selalu berlogika tinggi

2.6 *Real Time Clock* (RTC) DS1307

Real Time Clock atau lebih dikenal RTC adalah jenis pewaktu yang bekerja berdasarkan waktu yang sebenarnya atau dengan kata lain berdasarkan waktu yang ada pada jam kita. Agar dapat berfungsi, pewaktu ini membutuhkan dua parameter utama yang harus ditentukan, yaitu pada saat mulai (*start*) dan pada saat berhenti (*stop*) yang sesuai dengan satuan waktu. Kebanyakan RTC menggunakan sebuah pembangkit *clock* dengan frekuensi *clock* 32,786 kHz. *Clock* tersebut sama dengan frekuensi yang digunakan pada jam tangan dan untuk alasan kesesuaian dengan 2^{15} siklus tiap detik yang mana mudah digunakan untuk perhitungan biner. (Wikipedia, 2012)

DS1307 merupakan RTC dengan konsumsi daya rendah yang berkisar 500nA dengan memori sebesar 56 byte SRAM dengan akses secara serial sinkron I^2C. DS1307 mampu menyimpan dan membaca data detik, menit, jam, bulan, dan tahun. RTC jenis ini mampu mensetting secara otomatis untuk bulan yang mempunyai umur kurang dari 31 hari. Selain itu mampu mendeteksi kehilangan catu daya dan otomatis berpindah

pada *supply* baterai dan dapat terus beroperasi. Konfigurasi pin IC DS1307 ditunjukkan dalam Gambar 2.17.



Gambar 2.17 Konfigurasi Pin RTC DS1307

Sumber: Dallas, 2008: 1

Keterangan pin:

X1 dan X2 : Saluran untuk pembangkit sistem *clock*.

V_{BAT} : Saluran tegangan baterai cadangan lithium standar 3V untuk RTC jika tidak mendapatkan catu daya.

GND : *Ground*.

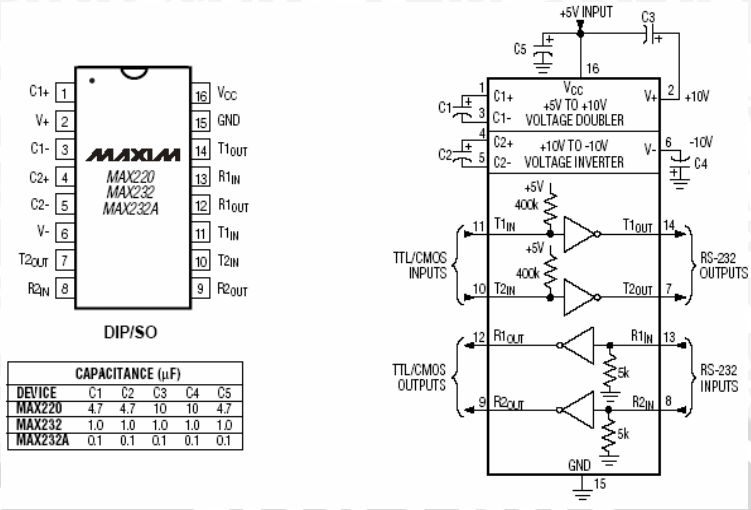
VCC : Catu daya utama.

SCL : *Serial Clock Input* untuk sinkronisasi data saat terjadi komunikasi I^2C.

SDA : *Serial Data Input/Output* sebagai jalur komunikasi data pada komunikasi I^2C.

2.7 MAX232

Komunikasi serial RS232 merupakan komunikasi dengan menggunakan level tegangan yang berkisar antara +3 volt sampai +15 volt atau -15 volt sampai -3 volt. Namun kebanyakan IC mikrokontroler hanya beroperasi pada tegangan TTL. Untuk itu diperlukan suatu rangkaian yang berfungsi untuk mengubah level tegangan dari TTL menjadi level tegangan yang sesuai dengan level tegangan RS232. Pada umumnya komunikasi konverter tegangan serial yang digunakan komponen IC RS232 yaitu pabrikan dari Maxim MAX232, di bawah ini konfigurasi pin IC MAX232 berikut rangkaian minimum yang dibutuhkan agar IC tersebut dapat bekerja. Konfigurasi Pin IC MAX232 ditunjukkan dalam Gambar 2.18.



CAPACITANCE (µF)					
DEVICE	C1	C2	C3	C4	C5
MAX220	4.7	4.7	10	10	4.7
MAX232	1.0	1.0	1.0	1.0	1.0
MAX232A	0.1	0.1	0.1	0.1	0.1

Gambar 2.18 Konfigurasi Pin IC MAX232

Sumber: Maxim, 2006: 17

BAB III
METODOLOGI PENELITIAN

Penyusunan proposal ini didasarkan pada masalah yang bersifat aplikatif, yaitu perencanaan dan perealisasian alat agar dapat bekerja sesuai dengan yang direncanakan dengan mengacu pada rumusan masalah. Langkah-langkah yang perlu dilakukan untuk merealisasikan alat yang dirancang adalah penentuan spesifikasi alat, studi literatur, perancangan dan pembuatan alat, pengujian alat, dan pengambilan kesimpulan.

3.1 Studi Literatur

Dalam merencanakan dan merealisasikan alat pemantau curah hujan dibutuhkan pemahaman tentang berbagai hal yang mendukung. Pemahaman ini akan bermanfaat untuk merancang dan merealisasikan alat meliputi dasar teori tentang hujan, *tipping bucket*, *optocoupler*, mikrokontroler, komunikasi serial sinkron dan asinkron, komunikasi I^2C, komunikasi Serial RS232, *Real Time Clock (RTC)* dan LCD.

3.2 Penentuan Spesifikasi Alat

Spesifikasi alat secara global ditetapkan terlebih dahulu sebagai acuan dalam perancangan selanjutnya. Spesifikasi alat yang direncanakan yaitu:

1) Sensor curah hujan yang digunakan adalah tipe *Tipping Bucket*.
2) Resolusi sensor curah hujan sebesar 0,5 mm.
3) Komponen *push button* digunakan untuk men-*setting* menit, jam, tanggal, bulan, tahun pada komponen RTC.
4) LCD 2x16 digunakan untuk menampilkan jam, jumlah *counter* jungkitan, data hasil pengukuran curah hujan dan sebagai tampilan saat proses kalibrasi RTC.
5) Data hasil pengukuran ditransfer ke komputer melalui komunikasi serial RS232.
6) Komputer berfungsi untuk mengolah dan menyimpan data hasil pengukuran ke dalam *database* serta menampilkan dalam bentuk grafik.
7) *Database* yang digunakan dalam komputer server adalah Microsoft Access.

3.3 Perancangan dan Pembuatan Alat

Pembuatan alat dalam skripsi ini dibagi menjadi dua bagian, yaitu perancangan perangkat keras (*hardware*) dan perangkat lunak (*software*).

1) Pembuatan perangkat keras (*hardware*) dibagi menjadi dua, yaitu:
   a. Pembuatan Mekanik

      Pembuatan mekanik disesuaikan dengan alat dan bahan yang tersedia dan dapat memenuhi spesifikasi alat yang berkaitan dengan ketelitian sensor sebesar 0,5 mm.

   b. Pembuatan Rangkaian Elektrik

      Pembuatan rangkaian elektrik disesuaikan dengan komponen elektronika yang tersedia dipasaran yang dapat memenuhi spesifikasi alat yang berkaitan dengan kehandalan rangkaian dan akurasi hasil pengukuran curah hujan.

2) Pembuatan perangkat lunak (*software*) dibagi menjadi dua, yaitu:
   a. Pembuatan *Software* Mikrokontroler

      Pembuatan *software* pada mikrokontroler menggunakan bahasa C dengan mengggunakan *compiler* CodeVision AVR.

   b. Pembuatan *Software* pada Komputer

      Pembuatan *software* pada mikrokontroler menggunakan program Borland Delphi 7 untuk menampilkan hasil pengukuran dan mengatur penyimpanan ke *database* dalam komputer

3.4 Pengujian Alat

Untuk menganalisis kinerja alat apakah sesuai dengan yang direncanakan, maka dilakukan pengujian sistem. Pengujian dilakukan pada masing-masing blok dan secara keseluruhan. Pengujian yang dilakukan meliputi:

1) Pengujian sensor curah hujan *tipping bucket*

   Pengujian sensor curah hujan *tipping bucket* dilakukan dengan cara mengkondisikan sensor dalam beberapa variasi keadaan kemudian menganalisis respon yang diberikan sensor. Pengujian ini bertujuan untuk memastikan sensor dapat mendeteksi curah hujan yang turun dengan resolusi pengukuran 0,5 mm/jam, serta memberikan analisis terhadap hasil pengujian.

2) Pengujian rangkaian pengkondisi sinyal sensor *optocoupler*

Pengujian rangkaian pengkondisi sinyal dilakukan dengan cara menggerakkan bagian jungkat-jungkit *tipping bucket* dan mendeteksi sinyal keluaran apakah telah sesuai dengan sinyal masukan untuk mikrokontroler. Pengujian ini bertujuan untuk memastikan pengkondisi sinyal telah berfungsi dan mampu mengeluarkan sinyal atau tegangan yang sesuai atau dapat dideteksi oleh mikrokontroler.

3) Pengujian LCD

Pengujian LCD dilakukan dengan cara menampilkan karakter tertentu pada LCD yang dikirim melalui mikrokontroler. Pengujian ini bertujuan untuk memastikan LCD telah berfungsi dan dapat bekerja sesuai dengan harapan untuk menampilkan data hasil pengukuran.

4) Pengujian RTC

Pengujian RTC dilakukan dengan cara menyimpan data menit, jam, tanggal, bulan, dan tahun pada RTC dengan menggunakan masukan dari mikrokontroler dan mematikan rangkaian. Bila RTC dihidupkan kembali maka RTC harus menunjukkan data saat ini. Pengujian ini bertujuan untuk memastikan RTC telah berfungsi dan dapat menyimpan data waktu serta dapat beroperasi saat tidak mendapat *supply* utama.

5) Pengujian rangkaian MAX232

Pengujian ini dilakukan dengan cara menghubungkan kaki masukan pada IC MAX232 dan mendeteksi keluaran menggunakan multimeter, serta menguji dengan menggunakan masukan pulsa dan mendeteksi tegangan keluaran menggunakan osiloskop. Pengujian ini bertujuan untuk memastikan tegangan keluaran pada konverter MAX232 telah sesuai dengan standar RS232.

6) Pengujian komunikasi serial

Pengujian ini dilakukan dengan cara mengirimkan data serial melalui pin TX pada mikrokontroler dengan *port serial* komputer dan melihat karakter keluaran pada Hyperterminal. Pengujian ini bertujuan untuk memastikan komunikasi serial antara mikrokontroler dengan komputer dapat berjalan sesuai dengan harapan.

7) Pengujian sistem secara keseluruhan

Pengujian ini dilakukan dengan cara menggabungkan semua bagian alat yang dibuat dan melihat kinerja alat. Pengujian ini bertujuan untuk menganalisis kinerja

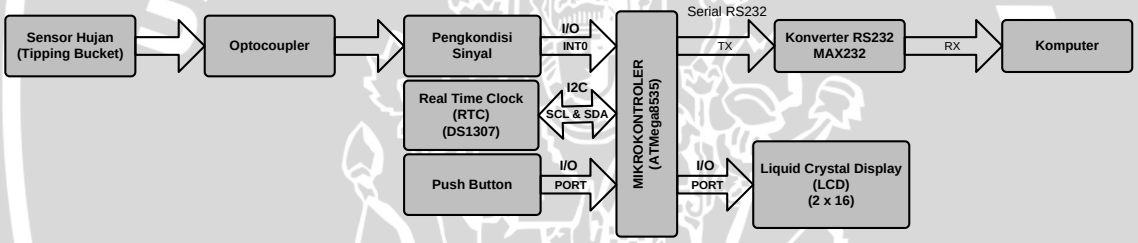
keseluruhan bagian alat yang dibuat apakah telah sinkron antara satu bagian dengan bagian yang lainnya.

BAB IV
PERANCANGAN DAN PEMBUATAN ALAT

Penyusunan laporan ini didasarkan pada masalah yang bersifat aplikatif, yaitu perencanaan dan pembuatan alat agar dapat bekerja sesuai dengan yang direncanakan dan mengacu pada rumusan masalah. Langkah-langkah yang perlu dilakukan untuk merealisasikan alat yang dirancang adalah penentuan spesifikasi alat, pembuatan diagram blok sistem secara keseluruhan, perancangan perangkat keras (*hardware*), perancangan protokol komunikasi data, dan perancangan perangkat lunak (*software*).

4.1 Diagram Blok Sistem

Secara garis besar, diagram blok perancangan *hardware* sistem secara keseluruhan ditunjukkan dalam Gambar 4.1.



Gambar 4.1 Blok Diagram Sistem Keseluruhan

Berdasarkan diagram blok dalam Gambar 4.1, dapat dijelaskan secara umum mengenai bagian-bagian yang menyusun keseluruhan sistem alat ini. Penjelasan bagian-bagian tersebut adalah sebagai berikut.

1) Sensor Hujan (*Tipping Bucket*)

   Berfungsi untuk mengumpulkan air hujan yang turun dan mengukur air yang telah terkumpul.

2) Pengkondisi Sinyal

   Berfungsi untuk mendeteksi gerakan mekanik pias dari sensor curah hujan dan mengubah keluaran menjadi sinyal elektrik yang sesuai dengan tegangan masukan untuk mikrokontroler.

3) Real Time *Clock* (DS1307)

Berfungsi untuk mengukur selisih waktu jungkitan pertama dengan jungkitan selanjutnya pada *tipping bucket* serta digunakan untuk mengatur waktu penyimpanan hasil pengukuran setiap 1 jam.

4) Push Button

Berfungsi sebagai tombol untuk mengkalibrasi data menit, jam, tanggal, bulan, dan tahun pada RTC serta untuk mengirim data hasil pengukuran menuju ke komputer melalu komunikasi serial RS232.

5) Mikrokontroler

Berfungsi sebagai pengontrol kerja keseluruhan alat.

6) LCD (2 x 16)

Berfungsi untuk menampilkan data hasil pengukuran secara *realtime* serta untuk menampilkan data menit, jam, tanggal, bulan, dan tahun.

7) Komputer

Berfungsi untuk menyimpan data hasil pengukuran ke dalam *database* dan menampilkan dalam bentuk grafik.

Prinsip kerja alat adalah mengukur hujan berdasarkan volume air yang jatuh pada *tipping bucket* setiap volume tertentu. Saat volume air yang ditampung telah sesuai dengan volume yang diharapkan maka akan dibuang dan mengisi kembali bagian penampung yang lain. Saat proses pembuangan ini terjadi pergerakan secara mekanik yang dideteksi oleh *optocoupler*. Keluaran *optocoupler* akan memicu pengkondisi sinyal untuk mengeluarkan sinyal keluaran yang sesuai dengan *level* tegangan TTL untuk memicu interupsi pada mikrokontroler. Saat terjadi interupsi maka mikrokontroler mengecek waktu pada RTC dan mengukur selisih waktu antara gerakan *tipping bucket* yang sebelumnya. Setelah diketahui selisih waktu akan dihitung curah hujan yang turun dan ditampilkan pada LCD. Untuk kebutuhan dokumentasi hasil pengukuran disimpan di dalam mikrokontroler untuk sementara yang selanjutnya dapat dikirim ke komputer melalui komunikasi serial RS232. Data yang diterima oleh komputer akan disimpan ke dalam *database* dan ditampilkan pada monitor dalam bentuk grafik.

4.2 Perancangan Perangkat Keras (*Hardware*)

4.2.1 Perancangan Mekanik

Dalam perancangan perangkat keras disebutkan bahwa sensor curah hujan yang digunakan adalah tipe *tipping bucket* dengan resolusi pengukuran sebesar 0,5 mm. Terdapat dua faktor yang mempengaruhi perancangan sebuah *tipping bucket*, yaitu luas bagian atas corong dan resolusi *tipping bucket* yang diinginkan.

Sesuai dengan dasar teori tentang pengukuran curah hujan yaitu jumlah curah hujan yang diukur sebenarnya adalah tebal atau tingginya permukaan air hujan setiap satu jam yang menutupi suatu area di permukaan bumi sebelum mengalami aliran permukaan, evaporasi, dan peresapan ke dalam tanah. Bila air hujan yang jatuh dikumpulkan dalam suatu wadah maka akan menggenangi wadah tersebut dengan ketinggian (h). Perhitungan curah hujan adalah kecepatan naiknya ketinggian air setiap satu jam.

Dalam rancangan *tipping bucket* dengan bentuk penampang bagian atas adalah lingkaran, sehingga dapat dihitung dengan menggunakan rumus luas lingkaran dengan resolusi pengukuran sebesar 0,5 mm dan jari-jari corong sebesar 12,5 cm, maka volume air setiap satu kali *clock* adalah:

Volume setiap jungkitan (V) = Luas corong (L) x Tinggi tampungan air hujan (h) (4-1)

$V = \pi \times r^2 \times 0{,}5 \text{ mm}$

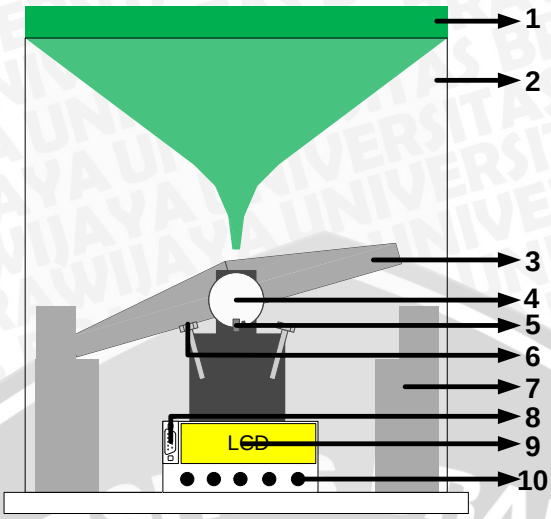
$= 3{,}14 \times (12{,}5 \text{ cm})^2 \times 0{,}05 \text{ cm}$

$= 3{,}14 \times 156{,}25 \text{ cm}^2 \times 0{,}05 \text{ cm}$

$= 24{,}53 \text{ cm}^3 = 24{,}53 \text{ ml} = 0{,}02453 \text{ liter}$

Dengan demikian curah hujan diukur dengan cara menghitung total jumlah *clock* atau gerakan pias pada *tipping bucket* dikalikan dengan resolusi pengukuran dalam waktu satu jam.

Perancangan mekanik alat pada ini terdapat beberapa komponen utama yang ditunjukkan dalam Gambar 4.2. Air hujan yang turun akan dikumpulkan oleh corong agar dapat menetes kebagian pias atau jungkat-jungkit. Bila volume air di dalam pias telah penuh maka akan dibuang melalui saluran pembuangan. Saat pias bergerak pada proses pengosongan air bagian piringan berlubang akan ikut bergerak. Pergerakan dari pias dideteksi oleh sensor *optocoupler* yang selanjutnya diolah oleh bagian elektrik.



Gambar 4.2 Rancangan Bentuk Mekanik

Keterangan gambar:

1) Corong.
2) Penutup alat.
3) Pias.
4) Piringan berlubang.
5) *Optocoupler*.
6) Baut kalibrasi.
7) Saluran pembuangan air dari Pias.
8) *Port Serial*.
9) LCD.
10) *Push button*

4.2.2 Perancangan Elektrik

Perencanaan bagian perangkat keras ini dapat diuraikan sebagai berikut.

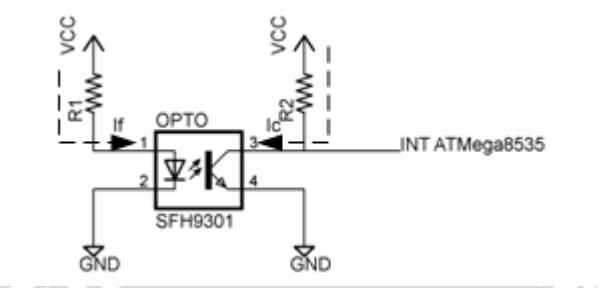
1) Perancangan rangkaian pengkondisi sinyal *optocoupler*.
2) Perancangan rangkaian *push button*.
3) Perancangan rangkaian minimum sistem ATMega8535.
4) Perancangan rangkaian minimum sistem RTC DS1307.
5) Perancangan rangkaian MAX232.

4.2.2.1 Perancangan Rangkaian Pengkondisi Sinyal *Optocoupler*

Keluaran *tipping bucket* yang masih berupa gerakan mekanik memerlukan sensor gerakan yang berfungsi untuk mendeteksi gerakan dari pias *tipping bucket*. Pendeteksian gerakan *tipping bucket* tersebut dengan menggunakan *optocoupler*. Agar komponen *optocoupler* ini dapat bekerja dengan baik diperlukan rangkaian pengkondisi sinyal. Selain itu, pengkondisi sinyal berfungsi sebagai rangkaian yang menjembatani

antara *tipping bucket* dengan mikrokontroler. Oleh karena itu tengangan keluaran rangkaian pengkondisi sinyal harus sesuai dengan masukan untuk kaki INT0 pada mikrokontroler.

Bentuk rangkaian pengkondisi sinyal untuk komponen *optocoupler* ditunjukkan dalam Gambar 4.3.



Gambar 4.3 Pengkondisi Sinyal *Optocoupler*

Dalam perancangan ini digunakan *optocoupler* tipe SFH 9301. Dari *datasheet* diketahui spesifikasi arus LED atau I_F = 20 mA. Saat I_F = 20 mA fototransistor mengalami kondisi saturasi dengan arus kolektor atau I_C = 0,2 mA dan tegangan $V_{CE(SAT)}$ = 0,4 V. Sedangkan kebutuhan arus input yang masuk ke mikrokontroler saat high atau I_{IH} = 1 µA. Sehingga untuk menentukan besarnya nilai R1 dan R2 diperoleh dari hitungan sebagai berikut.

$$R1 = \frac{V_{CC} - V_{LED}}{I_F} \quad (4\text{-}2)$$

$$R1 = \frac{5\ V - 0{,}7\ V}{20\ mA}$$

$$R1 = \frac{4{,}3\ V}{20 \times 10^{-3}\ A}$$

$$R1 = 215\ \Omega \approx 220\ \Omega$$

$$R2 = \frac{V_{CC} - V_{CE}}{I_C} \quad (4\text{-}3)$$

$$R2 = \frac{5\ V - 0{,}4\ V}{0{,}2\ mA}$$

$$R2 = \frac{4{,}6\ V}{0{,}2 \times 10^{-3} A}$$

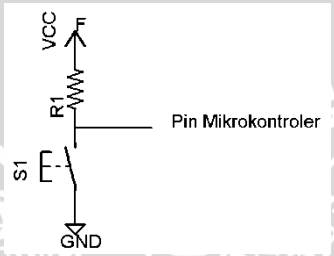
$$R2 = \frac{V_{CC} - V_{CE}}{I_F}$$

$$R2 = 23000\ \Omega = 23\ K\Omega \approx 22K\Omega$$

4.2.2.2 Perancangan Rangkaian Tombol *Push Button*

Push button pada alat ini digunakan sebagai tombol untuk mengkalibrasi RTC dan mengoperasikan alat. Pada alat ini direncanakan terdapat 5 buah tombol *push button*. Penggunaan 5 buah *push button* tersebut antara lain untuk kalibrasi RTC, up/down saat kalibrasi, dan tombol untuk transfer hasil pengukuran menuju ke computer. Untuk tombol *mode* kalibrasi dihubungkan dengan pin INT1 pada mikrokontroler, sedangkan tombol *Up* dan *Down* dihubungkan dengan pin D4 dan D5. Untuk tombol transfer data hasil pengukuran dihubungkan dengan pin INT2. Agar tombol dapat berfungsi dengan baik maka tegangan harus sesuai dengan masukan untuk mikrokontroler.

Bentuk rangkaian pengkondisi sinyal untuk *push button* ditunjukkan dalam Gambar 4.4.



Gambar 4.4 Rangkaian *Push Button*

Pemasangan R1 bertujuan untuk membatasi arus ketika *push button* ditekan. Jika arus yang masuk terlalu besar dapat mengakibatkan arus supply rangkaian yang lain terganggu. Dalam perancangan ditetapkan besar arus yang melewati resistor dibatasi 0,5 mA, karena pada saat tombol belum ditekan kebutuhan arus masuk mikrokontroler saat high atau I_{IH} = 1 µA. Sehingga besarnya R1 diperoleh dari hitungan sebagai berikut.

$$R1 = \frac{VCC}{I_{R1}} \quad (4\text{-}4)$$

$$R1 = \frac{5\ V}{0{,}5\ mA}$$

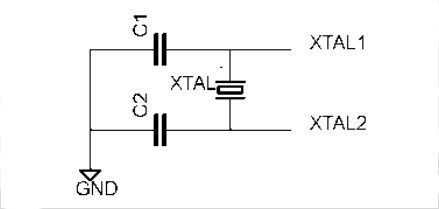
$$R1 = \frac{5\ V}{0{,}5 \times 10^{-3}}$$

$$R1 = 10000\ \Omega = 10\ K\Omega$$

4.2.2.3 Perancangan Rangkaian Minimum Sistem ATMega8535

Mikrokontroler yang digunakan pada rangkaian ini adalah ATMega8535 yang termasuk dalam seri AVR. Agar sebuah mikrokontroler dapat bekerja, maka diperlukan

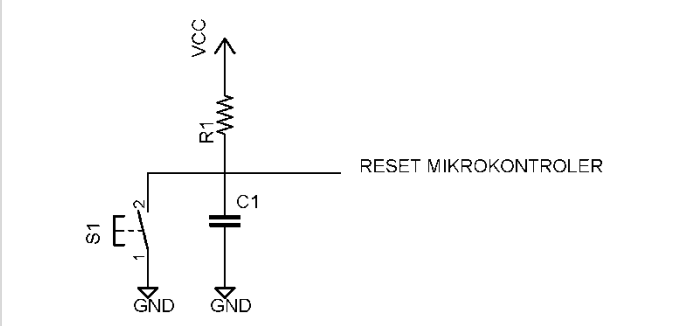
rangkaian pembangkit *clock* dan rangkaian *reset*. Rangkaian pembangkit *clock* sendiri terdiri dari komponen utama yang berupa kristal dan kapasitor. Sesuai dengan *datasheet* rangkaian pembangkit *clock* ditunjukkan dalam Gambar 4.5.



Gambar 4.5 Rangkaian Pembangkit *Clock*

Pada mikrokontroler ATMega8535 nilai Kristal yang diizinkan berkisar antara 0,4 MHz sampai 16 MHz. Dalam perancangan digunakan kristal sebesar 12 MHz. Sedangkan besarnya nilai C1 dan C2 disesuaikan dengan yang tertera dalam *datasheet* antara 12 pF sampai 22 pF. Dalam perancagan digunakan digunakan kapasitor sebesar 22 pF.

Dalam rangkaian mikrokontroler diperlukan rangkaian *reset*. Rangkaian ini berfunsi untuk mengatur waktu me-*restart* ulang program bila terjadi *error* saat tombol *reset* ditekan. Rangkaian *reset* ditunjukkan dalam Gambar 4.6.



Gambar 4.6 Rangkaian *Reset*

Berdasarkan *datasheet*, pin *RESET* harus diberi logika rendah minimal selama 1,5 µs untuk me-*reset* mikrokontroler. Waktu minimal ini dijadikan pedoman untuk menentukan nilai R dan C. Bila nilai C telah ditetapkan sebesar 10 µF dan nilai R ditetapkan sebesar 1 KΩ, maka berdasarkan Persamaan 2.5 nilai R adalah sebagai berikut.

$$t = 0{,}616 \times R \times C \quad (2\text{-}5)$$

$$t = 0{,}616 \times 1\ K\Omega \times 10\ \mu F$$

$$t = 0{,}616 \times 10^{3}\ \Omega \times 10 \times 10^{-6} F$$

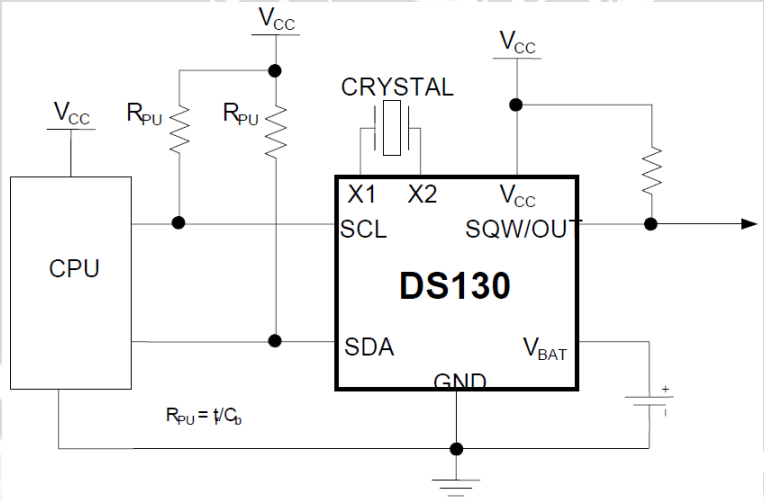
$$t = 6{,}16 \times 10^{-3}\ s$$

$$t = 6{,}16\ ms$$

Berdasarkan perhitungan dapat diketahui waktu *reset* yang terbentuk adalah selama 6,16 ms. Nilai tersebut lebih besar dari batas minimal waktu *reset* sebesar 1,5 µs, sehingga nilai C sebesar 10 µs dan R sebesar 1 KΩ dapat diaplikasikan dalam rangkaian *reset* mikrokontroler.

4.2.2.4 Perancangan Rangkaian Minimum Sistem RTC DS1307

RTC yang digunakan pada rangkaian ini adalah DS1307. Agar sebuah RTC dapat bekerja diperlukan suplai daya dan pembangkit *clock*. Suplai daya pada RTC terbagi menjadi dua, yaitu suplai daya utama dan suplai daya baterai. Supplai utama digunakan sebagai suplai keseluruhan sistem. Suplai baterai berfunsi sebagai suplai pengganti bila suplai utama tidak ada, sehingga RTC dapat tetap berjalan tanpa suplai utama. Sesuai dengan *datasheet* minimum sistem IC DS1307 ditunjukkan dalam Gambar 4.7.



Gambar 4.7 Minimum Sistem DS1307

Agar RTC dapat bekerja diperlukan rangkaian pembangkit *clock*. Komponen yang digunakan untuk membangkitkan *clock* adalah Kristal. Berdasarkan *datasheet* nilai

Kristal yang dapat digunakan antara range 0 sampa 100 KHz. Namun dalam perancangan ini digunakan kristal 32,768 KHz.

Resistor pull up (Rp) berfungsi untuk meningkatkan skaligus membatasi arus yang dibutuhkan untuk komunikasi I^2C, bila diketahui C_b DS1307 sebesar 400 pF. Berdasarkan *datasheet* ATMega8535 perhitungan nilai Rp minimum (Rp_{MIN}) dan maksimum (Rp_{MAX}) adalah sebagai berikut.

$$Rp_{MIN} = \frac{VCC - 0{,}4\ V}{3\ mA} \quad (4\text{-}5)$$

$$Rp_{MIN} = \frac{5\ V - 0{,}4\ V}{3\ mA}$$

$$Rp_{MIN} = \frac{4{,}6\ V}{3\ \times 10^{-3} A}$$

$$Rp_{MIN} = 1533{,}33\ \Omega = 1{,}5\ K\Omega$$

$$Rp_{MAX} = \frac{1000\ ns}{C_b} \quad (4\text{-}6)$$

$$Rp_{MAX} = \frac{1000\ ns}{400\ pF}$$

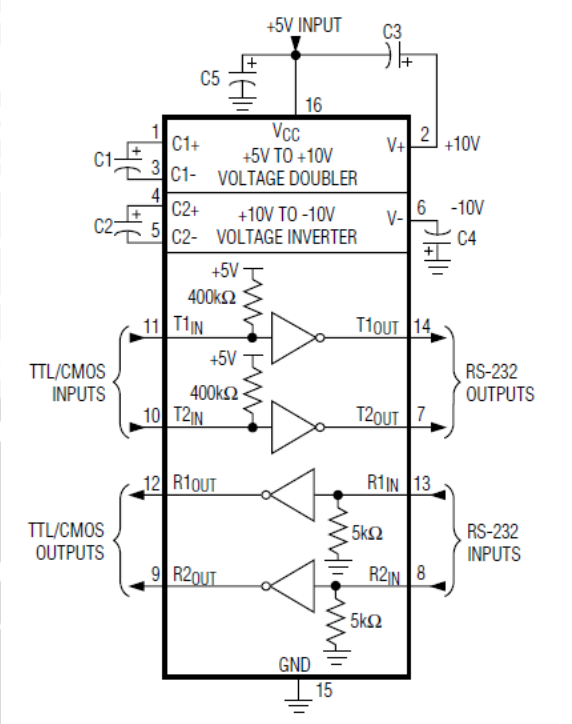
$$Rp_{MAX} = \frac{1000\ \times 10^{-9} s}{400\ \times 10^{-12} F}$$

$$Rp_{MAX} = 2500\ \Omega = 2K5\ \Omega$$

Berdasarkan hasil perhitungan nilai Rp yang dapat dipergunakan antara 1K5 Ω sampai 2K5 Ω. Dalam perancangan ini ditentukan nilai resistor yang terdapat di pasaran, yaitu sebesar 2K2 Ω.

4.2.2.5 Perancangan rangkaian MAX232

Pada perancangan ini IC yang digunakan adalah MAX232 yang diproduksi oleh Maxim. Berdasarkan *datasheet* agar IC ini dapat bekerja diperlukan komponen kapasitor sebagai pengganda tegangan dan pembalik tegangan agar tegangan keluaran sesuai dengan *level* tegangan RS232. Sesuai dengan yang tertera dalam *datasheet* kapasitor yang dipasang adalah sebesar 1 µF Rangkaian minimum sistem MAX232 ditunjukkan dalam Gambar 4.8.

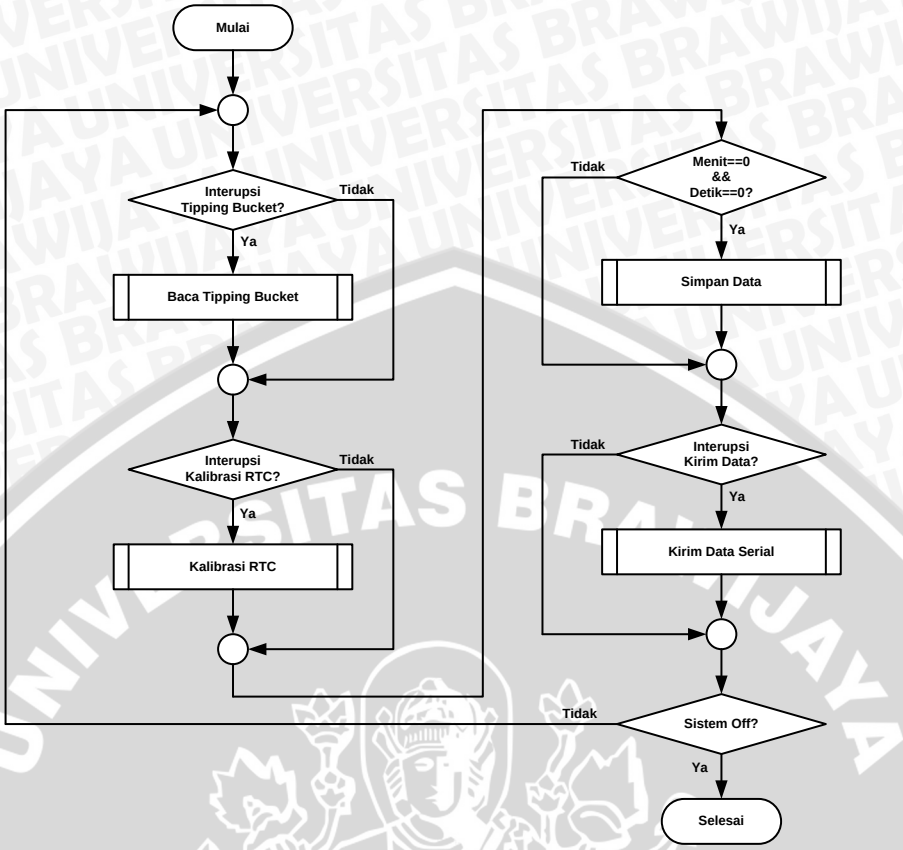


Gambar 4.8 Minimum Sistem MAX232

4.3 Perancangan Perangkat Lunak (*Software*)

4.3.1 Perancangan Perangkat Lunak Mikrokontroler

Perancangan perangkat lunak pada mikrokontroler menggunakan bahasa C dengan melalui *compiler* program CodeVisionAVR. Untuk memberikan gambaran umum jalannya program dan memudahkan dalam pembuatan perangkat lunak, maka dibuat diagram alir yang memudahkan jalannya program, diagram alir program utama ditunjukkan dalam Gambar 4.9.



Gambar 4.9 Diagram Alir Program Utama

Mikrokontroler merupakan komponen utama yang digunakan untuk mengatur seluruh kerja sistem. Ketika mikrokontroler aktif, mikrokontroler akan mendeteksi interupsi dari *optocoupler*. Jika terjadi interupsi, maka akan mengerjakan sub rutin baca *tipping bucket*. Proses selanjutnya mikrokontroler akan mendeteksi interupsi *push button*. Bila terjadi interupsi kalibrasi RTC, maka akan mengerjakan sub rutin kalibrasi RTC. Mikrokontroler akan mengechek data jam pada RTC. Hal ini bertujuan untuk melakukan penyimpanan data hasil pemantauan secara berkala. Saat terjadi interupsi untuk pengiriman data, mikrokontroler akan mengerjakan sub rutin kirim data serial untuk mengirimkan data yang tersimpan sementara di dalam mikrokontroler.

4.3.1.1 Perancangan Sub Rutin Baca *Tipping Bucket* Mikrokontroler

Saat terjadi interupsi mikrokontroler akan mengeksekusi sub rutin baca *tipping bucket*. Mikrokontroler mengecek selisih waktu dengan sebelumnya. Bila selisih waktu antara jungkitan sebelumnya lebih dari 5 jam, maka mikrokontroler akan me-*reset* data curah hujan dan menampilkan langsung pada LCD. Bila selisih waktu kurang dari 5

jam, maka mikrokontroler akan menghitung selisih waktu antara interupsi sebelumnya dan meng-*increament counter*. Data selisih waktu interupsi digunakan untuk menghitung curah hujan yang turun. Perhitungan curah hujan ditunjukkan dalam Persamaan 4-7. Diagram alir sub rutin baca *tipping bucket* ditunjukkan dalam Gambar 4.10.

Curah hujan tiap jam:

$$CH = \frac{n}{1\ jam} \times Res \quad (4\text{-}7)$$

$$CH = \frac{n}{1\ jam} \times 0{,}5$$

Curah hujan tiap satuan waktu atau Δt:

$$CH = \frac{1}{\frac{\Delta t}{3600}} \times 0{,}5 \quad (4\text{-}8)$$

$$CH = \frac{3600}{\Delta t} \times 0{,}5$$

$$Curah\ hujan = \frac{1800}{\Delta t}$$

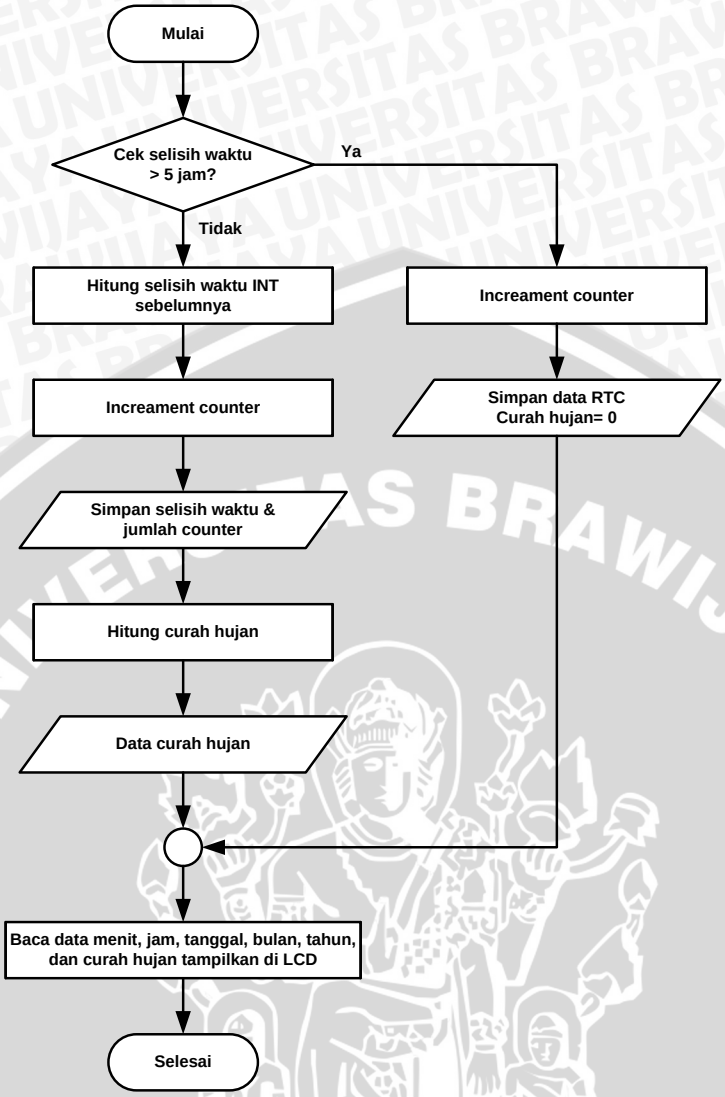
Keterangan:

n = Jumlah jungkitan

Δt = Jeda jungkitan (detik)

Res = Resolusi

CH = Curah Hujan ($\frac{mm}{jam}$)

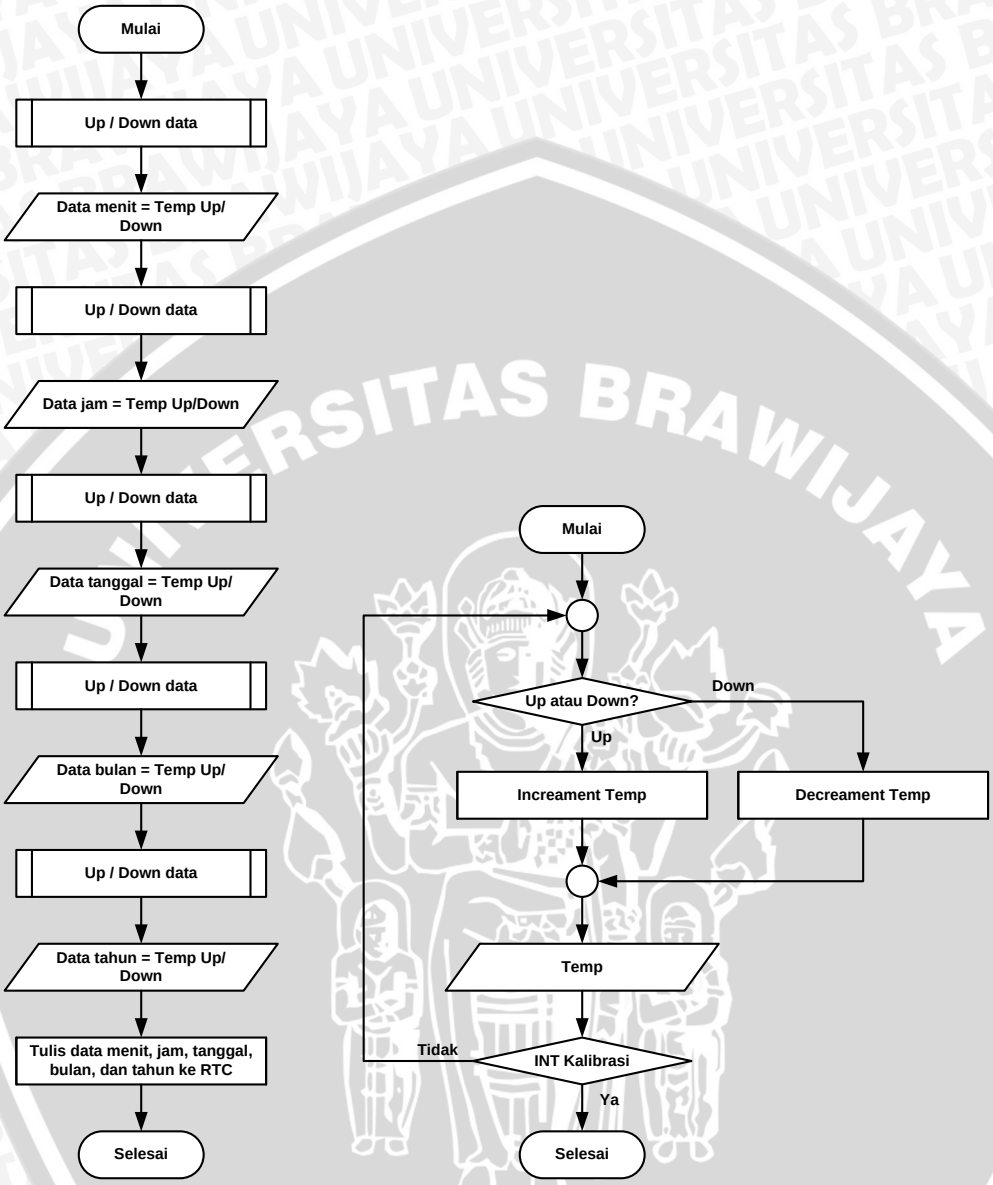


Gambar 4.10 Diagram Alir Sub Rutin Baca *Tipping bucket*

4.3.1.2 Perancangan Sub Rutin Kalibrasi RTC

Saat terjadi interupsi mikrokontroler akan mengeksekusi sub rutin kalibrasi RTC. Di dalam sub rutin kalibrasi RTC sendiri terdapat sub rutin *Up/Down* data. Tujuan pembuatan sub rutin ini adalah untuk menghemat memori program. Hal ini disebabkan mikrokontroler hanya mengulang-ulang program yang telah tersedia. Saat bila tombol *Up* ditekan maka data temp akan di-*increament* dan sebaliknya bila tombol *down* ditekan maka data temp akan di-*decreament.* Bila tombol kalibrasi RTC ditekan lagi maka mikrokontroler akan mengkopi data temp ke variabel menit. Selanjutnya mikrokontroler akan mengeksekusi sub rutin *Up/Down* Data kembali. Hingga proses

copy data temp ke data tahun. Diagram alir sub rutin kalibrasi RTC dan sub rutin *Up/Down* data ditunjukkan dalam Gambar 4.11 (a) dan Gambar 4.11 (b).

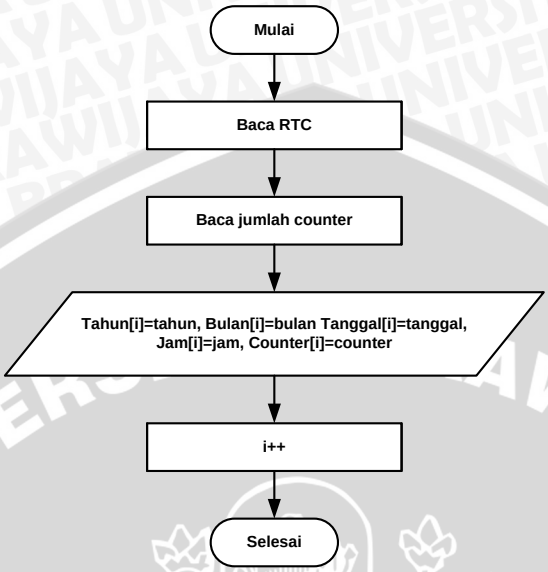


Gambar 4.11 (a) Diagram Alir Sub Rutin Kalibrasi RTC
(b) Diagram Alir Sub Rutin *Up/Down* Data

4.3.1.3 Perancangan Sub Rutin Simpan Data

Dalam sub rutin ini mikrokontroler bertugas mengecek data menit dan detik pada RTC. Jika data menit = 00 dan detik = 00, maka mikrokontroler akan membaca jumlah *clock* yang terjadi dalam satu jam terakhir. Untuk menghindari tumpang tindih data dalam sub rutin ini, penyimpanan antara data sebelumnya dibedakan dengan cara

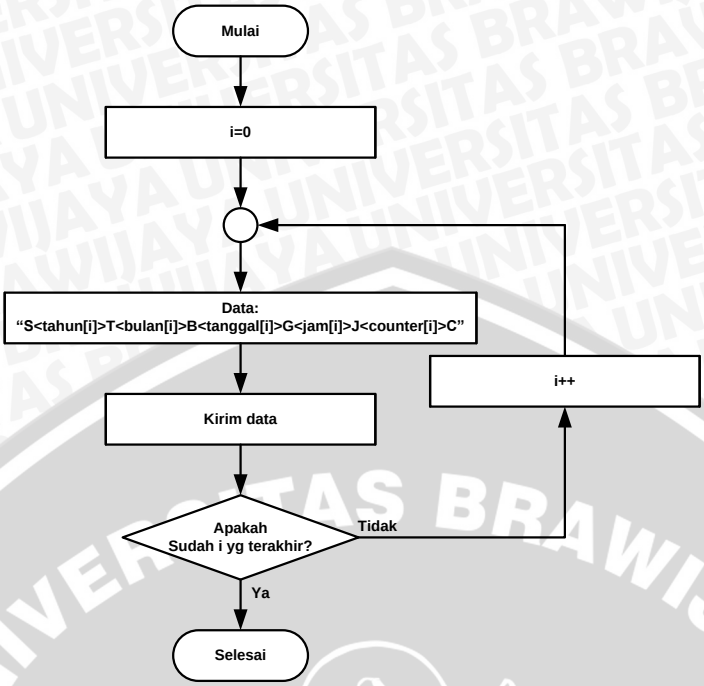
memberikan angka indek pada *variable* yang digunakan. Data yang disimpan dalam variabel tersebut adalah data tahun, bulan, tanggal, jam, dan *counter*. Diagram alir sub rutin simpan data ditunjukkan dalam Gambar 4.12.



Gambar 4.12 Diagram Alir Sub Rutin Simpan Data

4.3.1.4 Perancangan Sub Rutin Kirim Data Serial

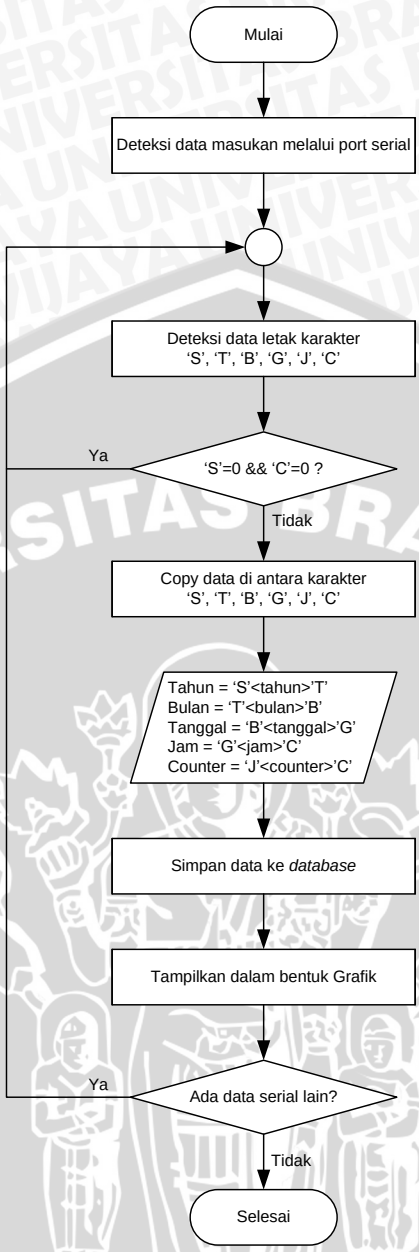
Saat terjadi interupsi mikrokontroler akan mengeksekusi sub rutin kirim data serial. Dalam sub rutin ini mikrokontroler akan membaca variabel hasil penyimpanan data setiap satu jam sebelum ditekan *reset* (*reset* utama). Data hasil penyimpanan dibaca berdasarkan indeks yang diberikan pada variabel. Data dengan angka indeks yang sama akan dibaca dan dikumpulkan menjadi satu dan dikirim melalui *port* TX pada mikrokontroler. Setelah data pertama terkirim angka indek akan dibanding kan dengan angka indek terakhir dari mikrokontroler. Bila masih belum sama berarti data tersebut bukan merupakan data terakhir, maka mikrokontroler akan meng-*increment* variabel. Jika angka indek telah sama dengan indek terakhir yang tersimpan didalam mikrokontroler, maka mikrokontroler akan kembali ke main program. Diagram alir sub rutin kirim data serial ditunjukkan dalam Gambar 4.13.



Gambar 4.13 Diagram Alir Sub Rutin Kirim Data Serial

4.3.2 Perancangan Perangkat Lunak pada Komputer

Saat pertama kali program dieksekusi akan membaca masukan dari *port* serial dengan baudrate yang telah disesuaikan dengan baudrate mikrokontroler. Data dari *port* serial dideteksi karakter penanda data utama, karakter penanda tersebut yaitu huruf ’S’, ’T’, ’B’, ’G’, ’J’, dan ’C’. Setelah diketahui letak 6 karakter tersebut, data dicopy diantara karakter penanda tersebut dan disimpan dalam variabel sementara. Setelah data tahun, bulan, tanggal, jam dan jumlah *counter* lengkap, data tersebut disimpan ke *database* file Microsoft Acces. Setelah proses penyimpanan selesai program akan membaca masukan dari *port* serial. Bila masih terdapat data yang masuk maka akan dideteksi ulang letak karakter pemisah pada data yang baru tersebut.



Gambar 4.14 Diagram Alir Program Delphi

BAB V
PENGUJIAN DAN ANALISIS

Pengujian dan analisis dilakukan untuk menganalisis apakah sistem telah bekerja sesuai perancangan. Pengujian dilakukan per blok kemudian secara keseluruhan. Pengujian yang perlu dilakukan adalah sebagai berikut:

1) Pengujian sensor curah hujan *tipping bucket.*
2) Pengujian rangkaian penkondisi sinyal sensor.
3) Pengujian LCD.
4) Pengujian RTC.
5) Pengujian Rangkaian MAX232.
6) Pengujian komunikasi serial.
7) Pengujian sistem secara keseluruhan.

5.1 Pengujian Sensor Curah Hujan *Tipping Bucket*

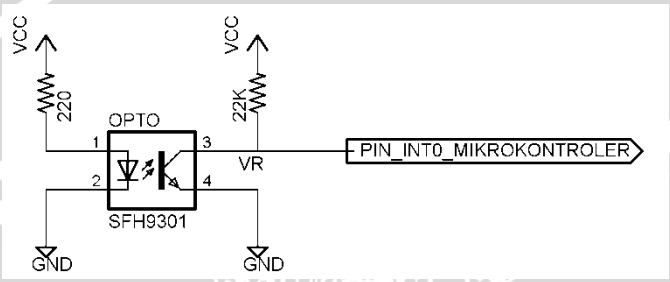
Pengujian pada blok ini dilakukan untuk menganalisis kerja *tipping bucket* dalam mengukur volume air hujan yang turun dan terkumpul di dalam pias *tipping bucket*. Volume air yang dapat ditampung oleh pias dapat diatur dengan memutar baut penahan pias. Volume air yang dapat ditampung oleh pias ruas kiri diatur oleh baut sebelah kanan dan sebaliknya. Sesuai dengan perancangan, volume air tersebut adalah sebesar 24,53 ml. Data hasil kalibrasi volume air pada pias *tipping bucket* ditunjukkan dalam Tabel 5.1.

Tabel 5.1 Data Hasil Pengujian Tipping Bucket

Resolusi		Jumlah Pengujian	Volume					Ketinggian Air						Error (%)	
Ketinggian (mm)	Volume (ml)		Perhitungan (ml)	Pengukuran (ml)				Perhitungan (mm)	Pengukuran						
				Kiri	Error	Kanan	Error		Kiri	Error	Kanan	Error			
0.5	24.53	1	24.53	25	0.47	25	0.47	0.5	0.5096	0.0096	0.5096	0.0096		1.911	1.9108
		2	49.06	50	0.94	50	0.94	1	1.0191	0.0191	1.0191	0.0191		1.911	1.9108
		3	73.59	76	2.41	74	0.41	1.5	1.5490	0.0490	1.5083	0.0083		3.270	0.5520
		4	98.12	102	3.88	94	4.12	2	2.0790	0.0790	1.9159	0.0841		3.949	4.2038
		6	147.18	148	0.82	142	5.18	3	3.0166	0.0166	2.8943	0.1057		0.552	3.5244
		8	196.24	192	4.24	194	2.24	4	3.9134	0.0866	3.9541	0.0459		2.166	1.1465
		10	245.3	242	3.30	242	3.30	5	4.9325	0.0675	4.9325	0.0675		1.350	1.3503

5.2 Pengujian Rangkaian Penkondisi Sinyal Sensor

Pengujian pada blok ini dilakukan untuk menganalisis kerja *optocoupler* dalam mendeteksi gerakan dari piringan berlubang pada *tipping bucket*. Keluaran dari pengkondisi sinyal *optocoupler* terhubung dengan pin INT0 pada mikrokontroler. Pada pin INT0 pada mikrokontroler diseting untuk mendeteksi logika 0 atau *active low*, sehingga saat *optocoupler* mendeteksi lubang maka keluaran dari pengkondisi sinyal harus 0 Volt dan kembali 5 Volt saat *optocoupler* terhalang. Pengujian rangkaian pengkondisi sinyal *optocoupler* ditunjukkan dalam Gambar 5.1.



Gambar 5.1 Pengujian Rangkaian *Optocoupler*

Data hasil pengujian rangkaian pengkondisi sinyal *optocoupler* ditunjukkan dalam Tabel 5.2.

Tabel 5.2 Hasil Pengujian Rangkaian Pengkondisi Sinyal *Optocoupler*

Kondisi *Optocoupler*	**Tegangan Titik V_R**
Terhalang	4,75 V
Tidak terhalang	0,16 V

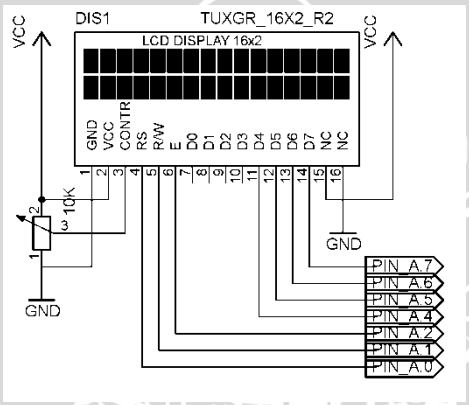
Pada saat *tipping bucket* tidak bergerak, kondisi tersebut sama dengan saat kondisi *optocoupler* terhalang. Cahaya dari LED transmiter tidak dapat langsung menuju pada basis fototransistor. Berdasarkan hasil percobaan saat optocupler terhalang tegangan titik V_R adalah 4,75 volt. Tegangan keluaran dari pengkondisi sinyal *optocoupler* tersebut telah sesuai dengan range tegangan masukan untuk logika high pada mikrokontroler, karena dalam *datasheet* mikrokontroler syarat input mikrokontroler dapat dideteksi logika 1 adalah antara $V_{IH\ (Min)}$ = 3 Volt sampai $V_{IH\ (Max)}$ = 5,5 Volt.

Pada saat piringan berlubang *tipping bucket* bergerak melewati *optocoupler*, cahaya dari LED dapat langsung menuju basis fototransistor. Kondisi tersebut sama dengan kodisi *optocoupler* saat tidak terhalang. Berdasarkan hasil percobaan tegangan keluaran saat kondisi tersebut adalah 0,16 volt. Tegangan keluraran dari pengkondisi

sinyal *optocoupler* tersebut telah sesuai dengan range tegangan masukan untuk logika *low* pada mikrokontroler, karena dalam *datasheet* disebutkan syarat input mikrokontroler dapat dideteksi logika 0 adalah antara $V_{IL\ (Min)}$ = -0,5 volt sampai $V_{IL\ (Max)}$ = 1 volt.

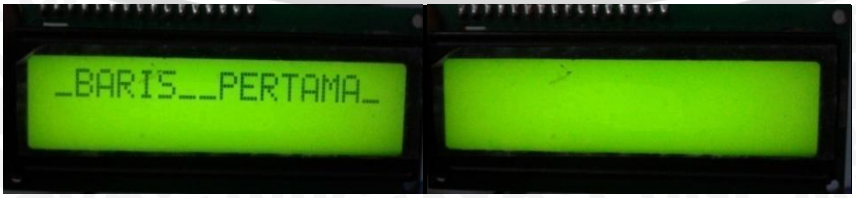
5.3 Pengujian LCD

Pengujian LCD dilakukan untuk menguji *interface port* A mikrokontroler terhadap LCD 2 x 16. Pengujian tersebut dilakukan dengan cara menulis atau menampilkan karakter tertentu pada LCD yang diperintah oleh mikrokontroler melalui program yang dimasukkan kedalamnya. Selain dapat ditulis, tampilan LCD juga harus dapat dihapus oleh mikrokontroler melalui program yang dimasukkan kedalamnya. Rangkaian pengujian LCD ditunjukkan dalam Gambar 5.2.



Gambar 5.2 Rangkaian Pengujian LCD

Prosedur pengujian rangkaian yang pertama LCD adalah membuat program mikrokontroler. Program tersebut berisi perintah untuk menampilkan karakter tertentu pada LCD. Selanjutnya karakter yang telah ditampilkan harus dapat dihapus dan digantikan karakter yang lain. Hal ini bertujuan untuk menguji kemampuan LCD dalam menampilkan data terbaru yang telah diolah oleh mikrokontroler. Hasil pengujian LCD ditunjukkan dalam Gambar 5.3.



(a) (b)

(c) (d)

Gambar 5.3 (a) Data Hasil Pengujian Baris 1 (b) Data Hasil Pengujian *Clear* LCD (c) Data Hasil Pengujian Baris 2 (d) Data Hasil Pengujian Baris 1 dan Baris 2

Berdasarkan data hasil percobaan dalam Gambar 5.3 (a) menunjukkan bahwa LCD pada baris pertama dan baris kedua telah bekerja sesuai dengan harapan. Hal ini dibuktikan mulai dari kolom peratama hingga kolom terakhir dapat menampilkan karakter yang sesuai dengan program yang telah dibuat dan baris kedua dalam keadaan kosong. Tampilan kosong pada baris kedua tersebut disebabkan memang tidak ada data yang harus ditampilkan di baris ke dua.

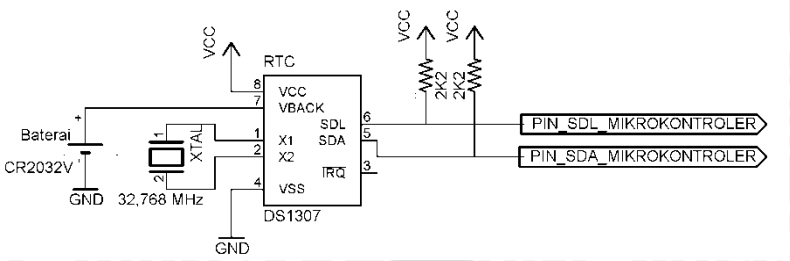
Tampilan senjutnya dalam gambar 5.3 (b) baris pertama dan kedua kosong. tampilan tersebut disebabkan pada program berisi perintah untuk menghapus semua tampilan yang muncul pada LCD. Tampilan tersebut membuktikan tampilan LCD dapat dihapus melalui perintah dari mikrokontroler.

Tampilan selanjutnya dalam Gambar 5.3 (c) menunjukkan bahwa LCD pada baris pertama kosong dan pada baris ke dua berisi karakter. Pada baris kedua berisi karakter yang sesuai dengan program yang telah dibuat dan tidak terdapat tampilan yang cacat. Berdasarkan Gambar 5.3 (c) menjunjukkan bahwa LCD telah berkerja sesuai dengan harapan.

Tampilan selanjutnya dalam Gambar 5.3 (d) menunjukkan bahwa LCD pada baris pertama dan kedua berisi karakter yang muncul secara bersamaan. Hal ini membuktikan bahwa mikrokontroler dapat menampilkan karakter baru dalam dua baris secara langsung.

5.4 Pengujian RTC

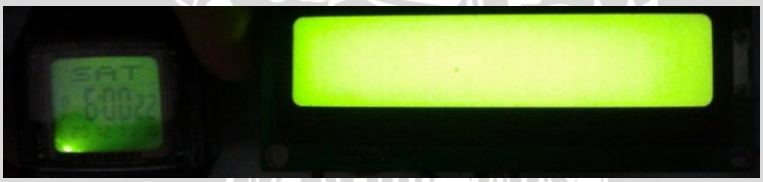
Pengujian pada blok ini dilakukan untuk menganalisis kerja RTC dalam menyimpan data detik, menit, jam, tanggal, bulan, dan tahun. Rangkaian RTC yang digunakan mempunyai 2 jenis *supply*, yaitu utama dan baterai. *Supply* baterai akan bekerja menggantikan *supply* utama ketika *supply* utama diputus. Bentuk rangkaian RTC ditunjukkan dalam Gambar 5.4.



Gambar 5.4 Rangkaian RTC

Pengisian data tahun, bulan, tanggal, jam, menit, dan detik pada RTC melalui komunikasi I^2C dengan menggunakan mikrokontroler. Bentuk pengujian dari rangkaian RTC yaitu data pertama yang dimasukkan pada RTC berupa data waktu saat ini, selanjutnya supply utama RTC dilepas dan mikrokontroler diisi program yang berfungsi untuk membaca data RTC. Pada proses selanjutnya data dalam RTC kembali dibaca ulang. Data hasil pembacaan RTC ditampilkan pada LCD harus menunjukkan waktu saat ini. Data hasil pengujian rangkaian RTC ditunjukkan dalam Gambar 5.5

(a)



(b)

(c)

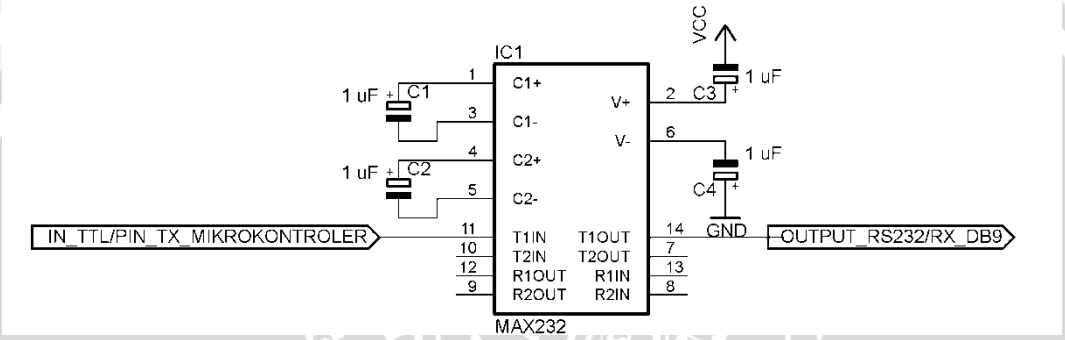
Gambar 5.5 (a) Memasukkan Data Ke Dalam Mikrokontroler (b) Pengujian Tanpa *Supply* Utama (c) Pengujian Setelah *Supply* Utama Kembali Terhubung

Berdasarkan data hasil percobaan dalam Gambar 5.5 (a) merupakan proses kalibrasi data detik, menit, jam, tanggal, bulan, dan tahun. Sebagai data acuan dalam proses kalibrasi menggunakan jam tangan. Selanjutnya pada Gambar 5.5 (b) merupakan tampilan LCD saat *supply* utama dan mikrokontroler dilepas, sehingga pada LCD hanya

terdapat tampilan kososng. Hal ini bertujuan untuk pengujian kemampuan RTC dalam menyimpan data dengan menggunakan *supply* baterai atau tanpa *supply* utama. Selanjutnya dalam Gambar 5.5 (c) merupakan bukti bahwa rangkaian RTC dapat menyimpan data, selain itu data jam dan tanggal tetap berjalan saat tanpa *supply* utama.

5.5 Pengujian Rangkaian MAX232

Pengujian pada blok ini dilakukan untuk menganalisis tegangan keluaran dari MAX232. Sesuai dengan fungsinya rangkaian MAX232 adalah untuk mengubah tegangan TTL menjadi tegangan standar RS232. Bentuk rangkaian MAX232 ditunjukkan dalam Gambar 5.6.



Gambar 5.6 Rangkaian MAX232

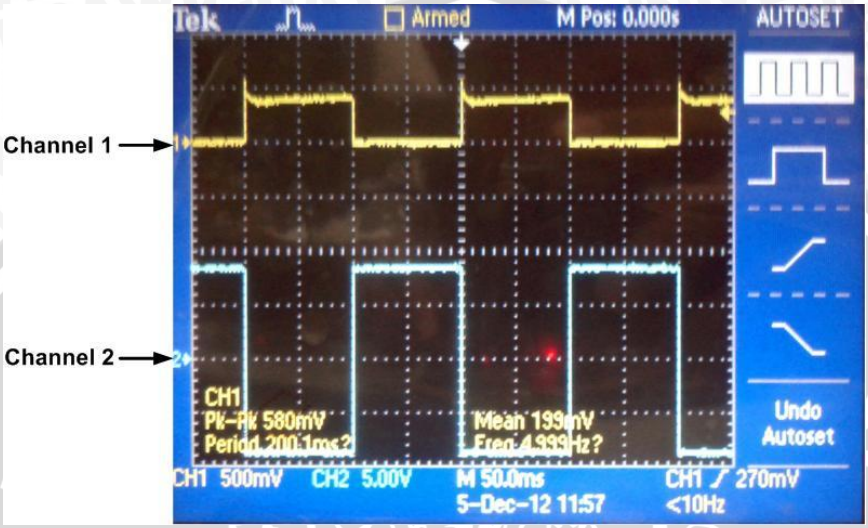
Pengujian rangkaian MAX232 diuji menjadi dua bagian, yaitu pengujian masukan tegangan secara manual dan pengujian masukan pulsa. Pengujian masukan tegangan TTL untuk membuktikan rangkaian dapat mengubah tegangan TTL menjadi standar RS232, sedangkan pengujian masukan pulsa bertujuan untuk menguji respon kecepatan rangkaian dalam merubah tegangan TTL menjadi tegangan yang sesuai dengan tegangan RS232. Data hasil pengujian tegangan keluaran dari rangkaian MAX232 ditunjukkan dalam Tabel 5.3.

Tabel 5.3 Hasil Pengujian Rangkaian MAX232

Masukan	Keluaran
0 V	8,56 V
4,88 V	-9,06 V

Dari hasil pengujian saat mendapat logika 0 atau tegangan sebesar 0 volt keluaran dari rangkaian MAX232 adalah sebesar 8,56 volt. Tegangan tersebut telah sesuai dengan standar RS232 saat mendapatkan logika 0, yaitu sebesar +3 volt hingga +15

volt. Sedangkan pada saat mendapat logika 1 atau tegangan sebesar 4,88 volt keluaran dari rangkaian MAX232 adalah sebesar -9,06 volt. Tegangan tersebut telah sesuai dengan standar RS232 saat mendapat logika 1, yaitu sebesar -3 volt hingga -15 volt. Data hasil pengujian masukan berupa pulsa pada rangkaian MAX232 ditunjukkan dalam Gambar 5.7

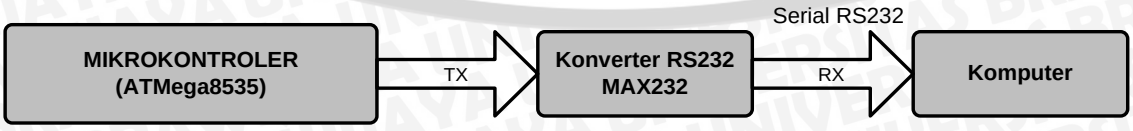


Gambar 5.7 Grafik Keluaran Rangkaian MAX232

Dari hasil pengujian dengan masukan pulsa (channel 1) pada rangkaian MAX232, pulsa keluaran (Channel 2) mampu mengikuti masukan dari rangkaian MAX232. Dapat disimpulkan rangkaian MAX232 telah mampu dan dapat digunakan sebagai rangkaian penghubung antara mikrokontroler dengan terminal masukan RS232.

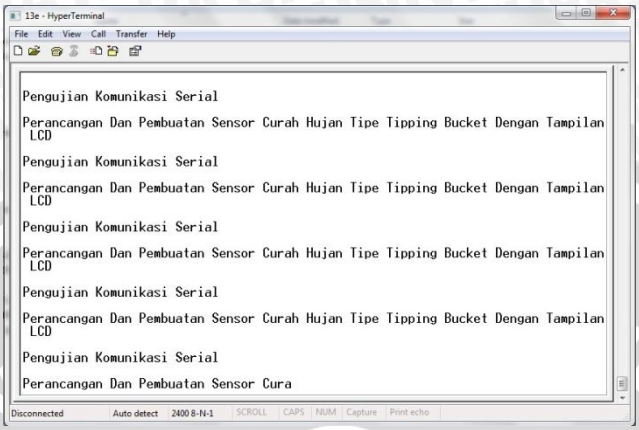
5.6 Pengujian Komunikasi Serial

Pengujian komunikasi serial ini dilakukan untuk menganalisis kinerja *hardware* dalam komunikasi serial. Pengujian dilakukan dengan merangkai mikrokontroler dengan rangkaian MAX232 dan menghubungkan ke pin RX pada *port serial* serta memantau hasil keluaran pada program hyperterminal. Blok diagram rangkaian pengujian komunikasi serial ditunjukkan dalam Gambar 5.8.



Gambar 5.8 Blok Diagram Pengujian Komunikasi Serial

Karakter yang akan ditampilkan pada hyperterminal dikirim melalui komunikasi serial yang dibangkitkan oleh mikrokontroler. Kecepatan komunikasi yang digunakan adalah sebesar 9600 bps dengan protokol komunikasi 8 bit data, 1 *stop* bit, dan tanpa *parity*. Data hasil pengujian komunikasi serial ditunjukkan dalam Gambar 5.9.

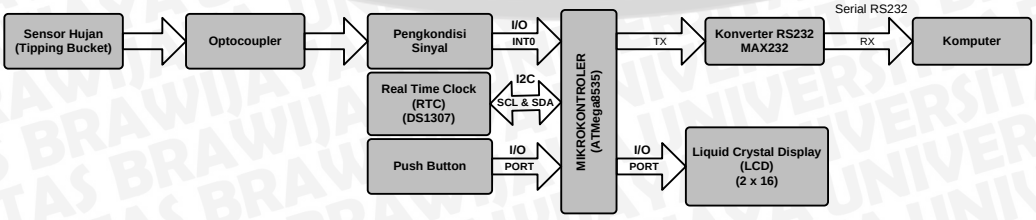


Gambar 5.9 Tampilan Hyperterminal

Pada tampilan hyperterminal karakter yang terbaca sama dengan karakter yang dibangkitkan oleh mikrokontroler, yaitu ”Perancangan Dan Pembuatan Sensor Curah Hujan Tipe *Tipping Bucket* Dengan Tampilan LCD” dan “Pengujian Komunikasi Serial”. Data hasil pembacaan pada hyperterminal juga tidak terdapat cacat karakter. Saat tombol reset pada mikrokontroler ditekan tampilan hyperterminal juga berhenti. Hal ini membuktikan hardware telah mampu mendukung komunikasi serial dengan kecepatan 9600 bps dengan 8 bit data, 1 bit *stop*, dan tanpa *parity*.

5.7 Pengujian Sistem Secara Keseluruhan

Pengujian sistem secara keseluruhan ini dilakukan untuk menganalisis kesesuaian kinerja dari *hardware* dan *software*. Pengujian dilakukan dengan merangkai keselurahan alat serta melakukan pengujian mulai dari pengambilan data hingga menampilkan data pada komputer dalam bentuk grafik. Bentuk diagram blok pengujian keseluruhan ditunjukkan dalam Gambar 5.10



Gambar 5.10 Blok Diagram Pengujian Keseluruhan Rangkaian

Pengujian keseluruhan sistem diawali ketika pias pada *tipping bucket* bergerak. Pergerakan dari pias tersebut dideteksi oleh sensor *optocoupler*. Selanjutnya *optocoupler* akan memicu interupsi pada mikrokontroler. Saat interupsi mikrokontroler meng-*increament counter* dan menampilkan hasilnya pada LCD. Pengujian dilakukan dengan cara menggerakkan pias dan melihat keluarannya pada tampilan LCD. Hasil pengujian interupsi dari sensor ditunjukkan dalam Gambar 5.11.

(a) (b)

Gambar 5.11 (a) Posisi Pias dan Tampilan LCD Sebelum Pengujian
(b) Posisi Pias dan Tampilan LCD Setelah Pengujian

Pengujian selanjutnya adalah pengujian kalibrasi pada komponen RTC. Pengujian ini dilakukan untuk menguji kemampuan RTC dalam menyimpan perubahan data. Pengujian ini dilakukan dengan mengaktifkan mode kalibrasi RTC dengan menekan tombol kalibrasi RTC. Mikrokontroler akan melihat data jam yang tersimpan dalam RTC untuk dikalibrasi. Setelah pembacaan data, selanjutnya mikrokontroler mendeteksi masukan dari tombol *up/down* untuk memasukkan data kalibrasi pada RTC. Proses berikutnya mikrokontroler akan menulis data pada RTC sesuai dengan data yang dimasukkan pada proses kalibrasi. Data terakhir yang dimasukkan pada RTC selanjutnya akan ditampilkan pada LCD. Hasil pengujian kalibrasi RTC ditunjukkan dalam Gambar 5.12, 5.13, dan 5.14.

Gambar 5.12 Tampilan Sebelum Kalibrasi

Gambar 5.13 Tampilan Saat Proses Kalibrasi

Gambar 5.14 Tampilan Setelah Proses Kalibrasi

Pengujian selanjutnya adalah pengujian hasil pengukuran curah hujan. Pengujian ini didasarkan jarak waktu antara interupsi pertama dengan berikutnya. Saat terjadi interupsi mikrokontroler akan menghitung selisih waktu antara interupsi sekarang

dengan interupsi yang sebelumnya. Semakin lama jarak antar interupsi, maka curah hujan semakin rendah. Selisih waktu antar jungkitan dimasukkan kedalam rumus untuk menghitung curah hujan. Pengujian dilakukan dengan cara mengatur rentang waktu antara jungkitan pertama dengan berikutnya. Hasil pengujian pengukuran curah hujan ditunjukkan dalam Gambar 5.15 dan Tabel 5.4.

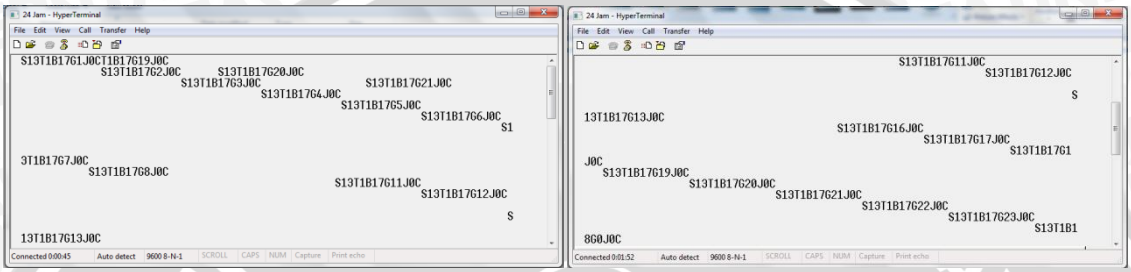
Gambar 5.15 Hasil Pengujian Pengukuran Curah Hujan

Tabel 5.4 Hasil Pengujian Pengukuran Curah Hujan

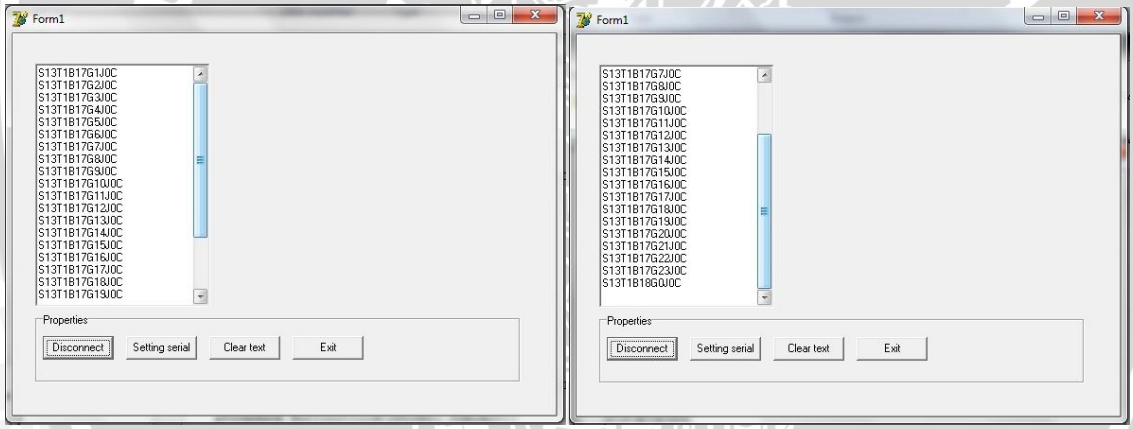
Selisih Waktu (menit)	curah hujan (mm/jam)		Error	
	Perhitungan	Pengukuran	(mm/jam)	(%)
1	30	29.51	0.49	1.633333
2	15	14.75	0.25	1.666667
3	10	9.89	0.11	1.1
4	7.5	7.44	0.06	0.8
5	6	5.98	0.02	0.333333
10	3	3	0	0
15	2	2	0	0
30	1	1	0	0
45	0.666666667	0.67	0.00333333	0.5
60	0.5	0.5	0	0

Pengujian selanjutnya adalah pengujian data hasil penyimpanan di dalam eeprom. Proses penyimpanan data ke dalam eeprom terjadi setiap 1 jam sekali atau saat menit

dan detik menunjukkan 00. Saat proses penyimpanan mikrokontroler akan membaca data tahun, bulan, tanggal, dan jam yang tersimpan pada RTC serta jumlah *clock* terakhir yang tersimpan. Data hasil pembacaan tersebut disimpan dalam bentuk *array*. Sehingga untuk pengujian hasil penyimpanan dilakukan dengan cara menghubungkan alat dengan komputer melalui *port serial* dan menekan tombol kirim data. Data hasil pengiriman dilihat melalui program. Hasil pengujian data hasil penyimpanan ditunjukkan dalam Gambar 5.16 dan Gambar 5.17.

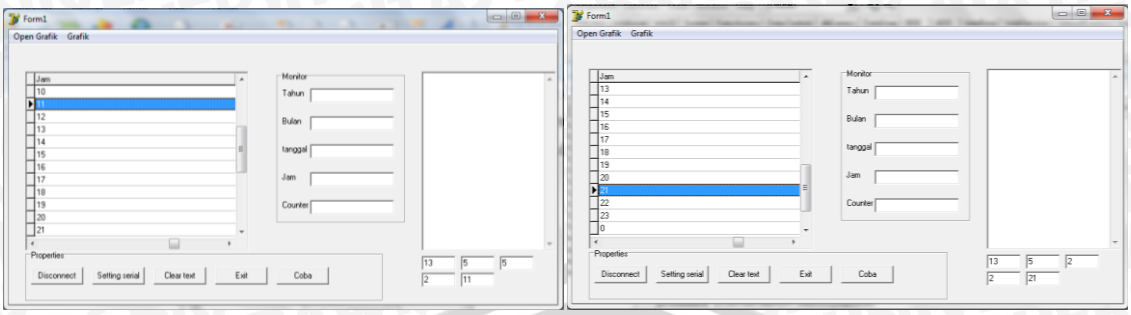


Gambar 5.16 Pengujian Pembacaan Data Menggunakan Hyperterminal



Gambar 5.17 Pengujian Pembacaan Data Menggunakan Delphi

Pengujian selanjutnya adalah pengujian program Delphi dalam mendeteksi data tahun, bulan, tanggal, jam, dan jumlah *counter* diantara karakter penanda. Pada proses ini program akan mendeteksi letak karakter penanda. Selanjutnya program akan mengkopi karakter yang terdapat diantara karakter penanda. Hasil pengujian pemisahan data hasil pengiriman ditunjukkan dalam Gambar 5.18.



Gambar 5.18 Pengujian Pemisahan Data Hasil Pengiriman

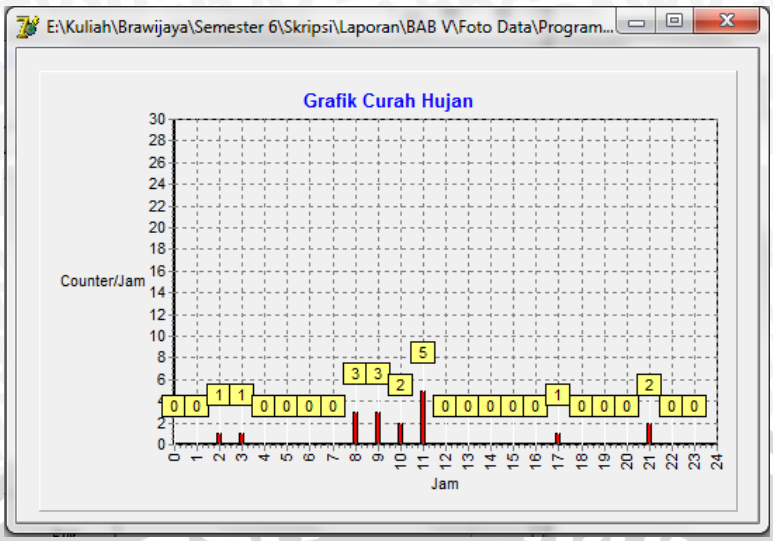
Pengujian berikutnya adalah pengujian penyimpanan data kedalam *database* Ms Microsoft acces. Data tahun, bulan, tanggal, jam, dan jumlah *counter* selanjutnya disimpan ke dalam *database* sesuai dengan kolom data masing-masing. Hasil pengujian penyimpanan data ke dalam *database* ditunjukkan dalam Gambar 5.19.

Tahun	Bulan	Tanggal	Jam	Counter
13	2	5	1	0
13	2	5	2	1
13	2	5	3	1
13	2	5	4	0
13	2	5	5	0
13	2	5	6	0
13	2	5	7	0
13	2	5	8	3
13	2	5	9	3
13	2	5	10	2
13	2	5	11	5
13	2	5	12	0
13	2	5	13	0
13	2	5	14	0
13	2	5	15	0
13	2	5	16	0
13	2	5	17	1
13	2	5	18	0
13	2	5	19	0
13	2	5	20	0
13	2	5	21	2
13	2	5	22	0
13	2	5	23	0
13	2	6	0	0

Record: 1 of 24 No Filter Search

Gambar 5.19 Pengujian Penyimpanan Data ke *Database* Access

Pengujian beriktunya adalah pengujian tampilan data dalam bentuk grafik. Data yang tersimpan di dalam *database* akan dibaca pada masing-masing jam dan ditampilkan dalam bentuk grafik. Sumbu x pada grafik menunjukkan jam dan sedangkan sumbu y menunjukkan jumlah *counter*. Hasil pengujian tampilan data dalam bentuk grafik ditunjukkan dalam Gambar 5.20.



Gambar 5.20 Pengujian Tampilan Data Dalam Bentuk Grafik

Dari seluruh pengujian di atas, dapat disimpulkan bahwa keseluruhan sistem dapat bekerja sesuai dengan spesifikasi yang diharapkan. Sistem telah mampu menghitung curah hujan, menyimpan data secara sementara, mengirim menuju komputer melalui komunikasi serial, menyimpan data ke dalam database, dan menampilkan dalam bentuk grafik.

BAB VI
KESIMPULAN DAN SARAN

6.1 Kesimpulan

Berdasarkan hasil perancangan dan pengujian yang telah dilakukan, dapat ditarik kesimpulan sebagai berikut:

1. Gerakan mekanik dari pias *tipping bucket* dapat dideteksi dengan menghubungkan pias dengan piringan berlubang. Lubang pada piringan tersebut berfungsi untuk mengatur cahaya yang masuk ke basis fototransistor pada komponen *optocoupler*. Saat cahaya terhalang keluaran dari *optocoupler* bernilai logika 1 (4,75 volt), sedangkan saat cahaya tidak terhalang keluaran dari optocoupler bernilai logika 0 (0,16 volt). Sehingga gerakan mekanik dari *tipping bucket* dapat dideteksi dan dapat dirubah menjadi sinyal elektrik yang sesuai dengan tegangan TTL.
2. Agar program dapat mendeteksi gerakan pias *tipping bucket* yang telah dirubah kedalam bentuk sinyal yang sesuai dengan tegangan TTL dapat dilakukan dengan mengaktifkan fasilitas interupsi didalam program yang terdapat pada mikrokontroler. Sedangkan penyimpanan data tahun, bulan, tanggal, jam, dan jumlah *counter* yang bersifat sementara selama 24 jam di dalam mikrokontroler dapat disimpan ke dalam memori EEPROM. Dalam satu kali penyimpanan memori yang terpakai sebesar 5 byte atau sebesar 120 byte untuk satu hari dengan rincian data tahun, bulan, tanggal, jam, dan jumlah *counter* yang berjenis *char* yang membutuhkan memori sebesar 1 byte untuk masing-masing data. Metode penyimpanan dapat dilakukan dengan cara membuat variabel secara otomatis dengan menggunakan array untuk masing-masing jenis data.
3. Untuk menghindari data saling tumpang tindih antara data sekarang dengan berikutnya di dalam EEPROM mikrokontroler dapat dilakukan dengan menyimpan data dengan menggunakan angka indeks atau array yang berbeda untuk masing-masing data. Sehingga saat proses transfer data melalui *port serial*, mikrokontroler membaca data tahun, bulan, tanggal, jam dan jumlah *counter* untuk angka indeks array yang sama. Setelah proses pengiriman data berakhir, diperlukan menekan tombol reset. Hal ini bertujuan untuk mengosongkan data yang telah dikirim agar saat pengiriman data berikutnya ke komputer tidak dibaca kembali.

4. Agar data yang dikirim dari mikrokontroler menuju komputer tidak salah dalam pembacaan dapat dilakukan dengan menyisipkan karakter unik di awal dan akhir data, serta diantara masing-masing data tahun, bulan, tanggal, jam, dan jumlah *counter*. Format pengiriman data adalah "S<data_tahun[i]>T<data_bulan[i]>B<data_tanggal[i]>G<data_jam[i]>J<data_counter[i]>C". Sehingga saat proses pembacaan data dalam komputer, Delphi hanya perlu mendeteksi letak karakter S, T, B, G, J, dan C tersebut. Selanjutnya Delphi mengkopi data yang disisipkan di antara karakter tersebut.

6.2 Saran

Saran-saran dalam implementasi maupun peningkatan unjuk kerja sistem ini dapat diuraikan sebagai berikut:

1. Dalam sistem ini hasil pengukuran ditampilkan pada LCD ukuran kecil. Saat terjadi hujan penggunaan LCD ukuran kecil akan menyulitkan untuk mengetahui curah hujan yang turun. Oleh karena itu, pengembangan selanjutnya sebaiknya perlu ditambahkan indikator visual yang dapat berupa warna lampu LED. Dengan penambahan indikator berupa warna lampu tersebut, diharapkan informasi bersifat untuk umum yang sekaligus bertujuan memberikan peringatan kepada masyarakat tentang jenis curah hujan yang turun (gerimis, sedang, deras, atau badai).
2. Dalam sistem yang dibuat ini media penyimpanan yang digunakan adalah EEPROM yang ada didalam mikrokontroler. Ukuran dari EEPROM ini yang sangat kecil, sehingga penyimpanan ini hanya bersifat sementara selama 24 jam. Untuk pengembangan selanjutnya sebaiknya media penyimpan dapat diganti dengan menggunakan media penyimpan data dengan ukuran yang lebih besar, sehingga mampu menampung data hingga 1 bulan.

DAFTAR PUSTAKA

Atmel. 2005. 8-bit AVR Microcontroller with 8K Bytes In-System Programmable Flash.

Bahri, Kusnassriyanto Saiful. 2008. Teknik Pemrograman Delphi. Bandung. Informatika

Dalas Semiconductor. 2008. DS1307 64 x 8, Serial, I^2C Real-Time Clock.

Dewi, Ike Kusuma. 2005. Pengontrol Sensor Pada Sistem Telemetri Stasiun Pengamat Cuaca Dilengkapi Dengan Sensor Suhu, Kelembaban, Curah Hujan, Kecepatan dan Arah Angin Diakses Melalui Internet. Malang: Tugas Akhir Jurusan Teknik Elektro Fakultas Teknik Universitas Brawijaya Malang.

Fairchild. 2009. Phototransistor Optical Interrupter Switch.

Kusuma, Erdy Prasetiya. 2007. Penyimpan Data Pada Sistem Telemetri Stasiun Pengamat Cuaca Dilengkapi Dengan Sensor Kelembaban, Temperatur, Curah Hujan, Kecepatan Angin dan Arah Angin Diakses Melalui Internet. Malang: Tugas Akhir Jurusan Teknik Elektro Fakultas Teknik Universitas Brawijaya Malang.

M, Reinaldo Kevin. 2009. Sensor *Optocoupler* Komponen Sistem Kontrol: http://fisikainstrumentasiukm.files.wordpress.com/2012/10/uts2012-sensor-*optocoupler*.docx. Diakses tanggal: 20 November 2012.

Maxim. 2006. *+5V-Powered, Multichannel RS-232 Drivers/Receivers.*

Osram. 2001. Slotted Interrupters.

Pracoyo, Agus. 2008. MENGAKSES SERIAL REAL TIME CLOCK (RTC) DALLAS TIPE DS1307 DENGAN BASCOM KOMPILER: http://isjd.pdii.lipi.go.id/admin/jurnal/61088895.pdf. Diakses Tanggal: 8 November 2012.

Priyahitajuniarfan. 2009. Komunikasi I2C. http://priyahitajuniarfan.wordpress.com/2009/05/17/komunikasi-i2c. Diakses tanggal: 8 November 2012.

Putra, Agfianto Eko. 2009. RTC DS12C887: Pendahuluan. RTC http://agfi.staff.ugm.ac.id/blog/index.php/2009/01/rtc-ds12c887-pendahuluan.htm. Diakses tanggal: 19 Juli 2012.

Sastaviyana, Adelia Revani. 2011. Rancang Bangun *Server* Sistem Pemantau Banjir *Real Time* Terpusat dengan Fasilitas Sms *Public Access*. Malang: Tugas Akhir S1 Jurusan Teknik Elektro Fakultas Teknik Universitas Brawijaya Malang.

Surya, Frans. 2007. I^2C Protokol: http://lab.binus.ac.id/pk/download/artikel/6/i2c.rar. Diakses tanggal: 8 November 2012.

Syarief, Mohammad. 2006. Stasiun Pusat Pengolah Data Pada Sistem Telemetri Stasiun Pengamat Cuaca Dilengkapi Dengan Sensor Kelembaban, Temperatur, Curah Hujan, Kecepatan dan Arah Angin Diakses Melalui Internet. Malang: Tugas Akhir S1 Jurusan Teknik Elektro Fakultas Teknik Universitas Brawijaya Malang.

Veanti, Okta. 2011. *Penakar Hujan Type Tipping Bucket.* http://kamus pengetahuan.blogspot.com/2011/04/penakar-hujan-type-tipping-bucket. html. Diakses tanggal: 19 Juli 2012.

Wangready. 2011. Komunikasi I2C pada Microcontroller AVR: http://blog.ub.ac.id /ayoxsz/2010/03/15/memahami-sistem-komuniksi-data-serial-menggunakan-protocol-i2c. Diakses tanggal: 8 November 2012.

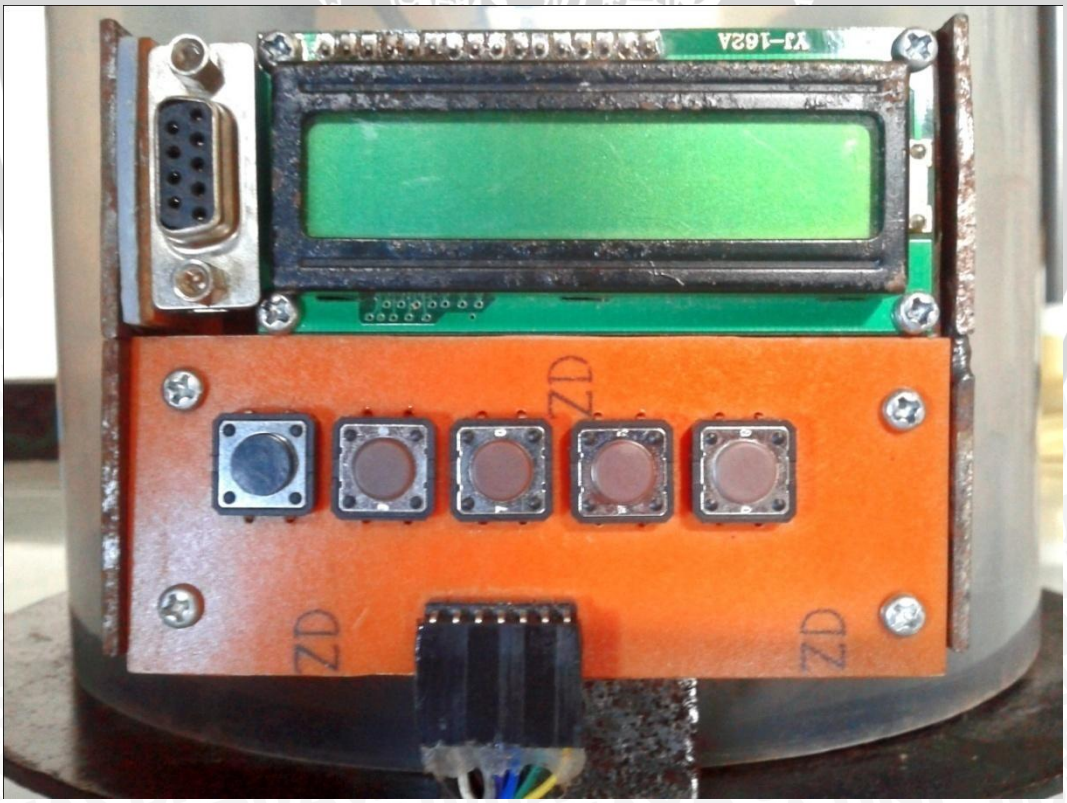
Winoto, Ardi. 2008. Mikrokontroler AVR ATMega8/32/16/8535 dan Pemrograman dengan Bahasa C pada WinAVR. Bandung: Informatika.

Weathershack. 2010. *Tipping Bucket Rain Gauge The Most Common Type Of Automated Rain Sensor*. http://www.weathershack.com/education/ tipping-bucket-rain-gauge.html. Diakses tanggal: 19 Juli 2012.

LAMPIRAN I

FOTO ALAT

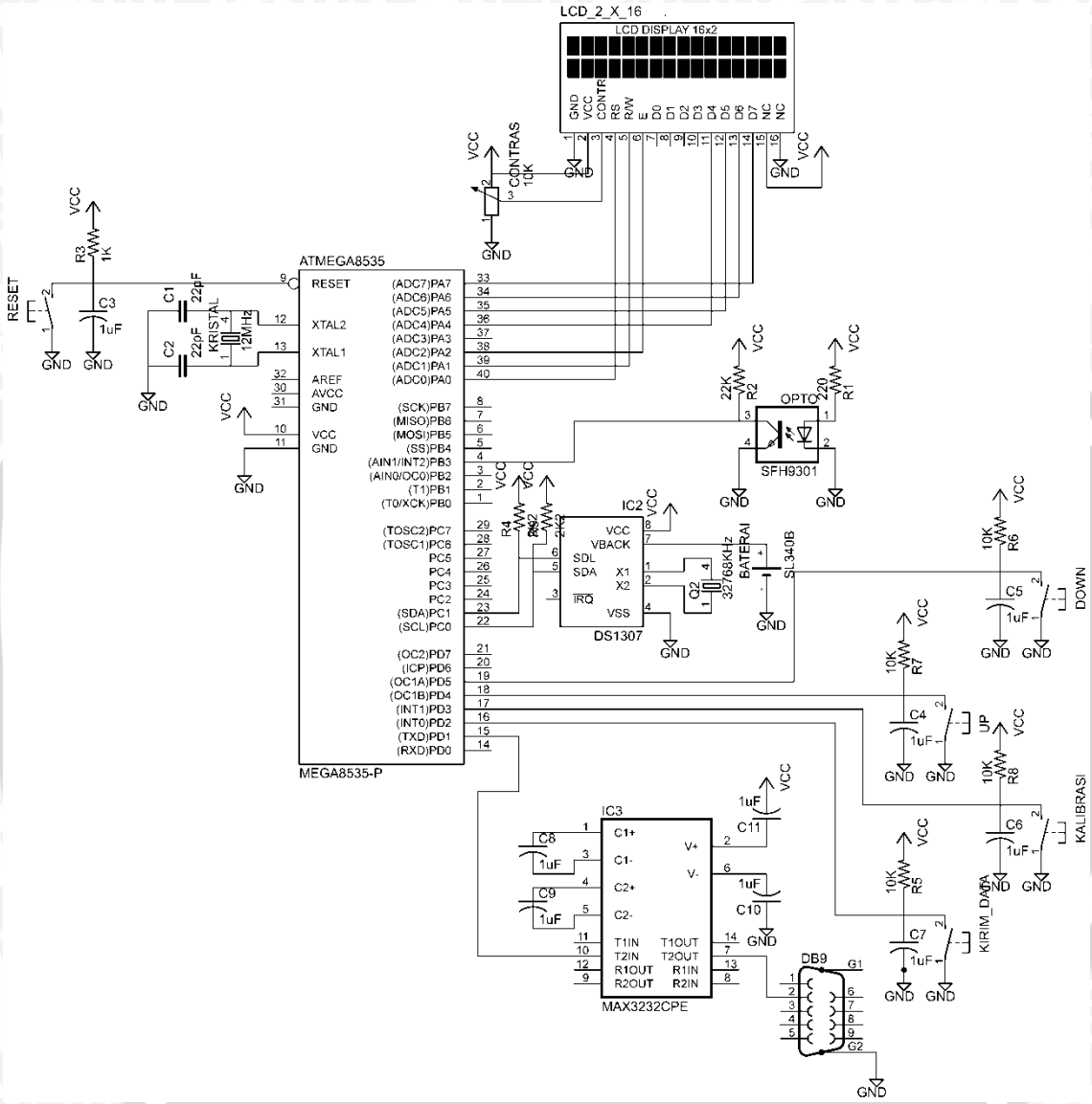
Gambar 1 Foto Alat



Gambar 2 Foto LCD dan Tombol

LAMPIRAN II

GAMBAR RANGKAIAN



Gambar 3 Rangkaian Keseluruhan

LAMPIRAN III

LIST PROGRAM ATMega8535

```
#include <mega8535.h>
#include <stdio.h>
#include <delay.h>
#include <math.h>

unsigned char s, m, h, dd, mm, yy, temp_tanggal;
int updown, counter=0, i=0, a;
eeprom unsigned char
T[24]={0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
B[24]={0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
G[24]={0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
J[24]={0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},
C[24]={0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};
char baris1[16], baris2[16], data[16];
float jam_ke_detik, menit_ke_detik, detik_ke_detik,
konversi_detik, selisih_detik, konversi_sebelumnya,
curah_hujan_skrg=0;

// I2C Bus functions
#asm
   .equ __i2c_port=0x15 ;PORTC
   .equ __sda_bit=1
   .equ __scl_bit=0
#endasm
#include <i2c.h>

// DS1307 Real Time Clock functions
#include <ds1307.h>

// Alphanumeric LCD Module functions
#asm
   .equ __lcd_port=0x1B ;PORTA
#endasm
#include <lcd.h>

void TULIS_LCD()
{
      lcd_clear();
      lcd_gotoxy(0,0);lcd_puts(baris1);
      lcd_gotoxy(0,1);lcd_puts(baris2);
}

void BACA_RTC()
{
      rtc_get_time(&h,&m,&s);
      rtc_get_date(&dd,&mm,&yy);
}

void PUSH_BUTTON()
{
      delay_ms(100);
      while (PIND.3==1)
      {
            if (PIND.4==0)
            {
                  delay_ms(10);
                  while (PIND.4==0)
                  {}
                  delay_ms(10);
                  updown++;
            }
            else if (PIND.5==0)
            {
                  delay_ms(10);
                  while (PIND.5==0)
                  {}
                  delay_ms(10);
                  updown=updown-1;
            }
            sprintf(baris2,"%i",updown);
            TULIS_LCD();
      }
      delay_ms(10);
      while (PIND.3==0)
      {}
      delay_ms(100);
}

void TULIS_RTC()
{
      rtc_set_time(h,m,s);
      rtc_set_date(dd,mm,yy);
}

void HITUNG_DETIK()
{
      BACA_RTC();
      jam_ke_detik=3.6*h;
      delay_ms(1);
      menit_ke_detik=0.06*m;
      delay_ms(1);
      detik_ke_detik=0.001*s;
      delay_ms(1);
      konversi_detik=jam_ke_detik+menit_ke_detik+detik_k
      e_detik;
      delay_ms(1);
}

void SELISIH_WAKTU()
{
      delay_ms(10);
      HITUNG_DETIK();
      delay_ms(1);
      if(dd==temp_tanggal)
      {
            delay_ms(1);
            selisih_detik=konversi_detik-
konversi_sebelumnya;
      }
      else
      {
            konversi_detik=86.4+konversi_detik;
            delay_ms(1);
            selisih_detik=konversi_detik-
konversi_sebelumnya;
      }
      konversi_sebelumnya=konversi_detik;
      selisih_detik=selisih_detik*1000;
      temp_tanggal=dd;
      delay_ms(1);
}

void RUTIN_SIMPAN()
{
      delay_ms(10);
      T[i]=yy;
```

```
        delay_ms(10);
        B[i]=mm;
        delay_ms(10);
        G[i]=dd;
        delay_ms(10);
        J[i]=h;
        delay_ms(10);delay_ms(10);
        C[i]=counter;
        delay_ms(10);

sprintf(data,"S%uT%uB%uG%uJ%iC",T[i],B[i],G[i],J[i],C[
i]);
        lcd_clear();
        lcd_gotoxy(0,0);lcd_puts(data);
        delay_ms(1000);
        a=i;
        i++;
        counter=0;
}

// External Interrupt 0 service routine
interrupt [EXT_INT0] void ext_int0_isr(void)
{
// Place your code here
        GICR&=0b00011111;
        delay_ms(1000);
        while(PIND.3==0)
        {}
        for (i=0;i<=a;i++)
        {

sprintf(data,"S%uT%uB%uG%uJ%iC",T[i],B[i],G[i],J[i],C[
i]);
                puts(data);
                delay_ms(1000);
        }
}

// External Interrupt 1 service routine
interrupt [EXT_INT1] void ext_int1_isr(void)
{
// Place your code here
        GICR&=0b00011111;
        delay_ms(10);
        while(PIND.3==0)
        {}
        BACA_RTC();
        //-----------------------  KALIBRASI MENIT
        sprintf(baris1,"MENIT ?");
        TULIS_LCD();
        updown=m;
        PUSH_BUTTON();
        m=updown;
        //----------------------- KALIBRASI JAM
        sprintf(baris1,"JAM ?");
        TULIS_LCD();
        updown=h;
        PUSH_BUTTON();
        h=updown;
        //----------------------- KALIBRASI TANGGAL
        sprintf(baris1,"TANGGAL ?");
        TULIS_LCD();
        updown=dd;
        PUSH_BUTTON();
        dd=updown;
        //----------------------- KALIBRASI BULAN
        sprintf(baris1,"BULAN ?");
        TULIS_LCD();
        updown=mm;
        PUSH_BUTTON();
        mm=updown;
        //----------------------- KALIBRASI TAHUN
        sprintf(baris1,"TAHUN ?");
        TULIS_LCD();
        updown=yy;
        PUSH_BUTTON();
        yy=updown;
        //----------------------- KALIBRASI DETIK
        s=00;
        TULIS_RTC();
        delay_ms(100);
        BACA_RTC();
        sprintf(baris1,"%u:%u %u/%u/%u",h,m,dd,mm,yy);
        TULIS_LCD();
        delay_ms(1000);
}

// External Interrupt 2 service routine
interrupt [EXT_INT2] void ext_int2_isr(void)
{
// Place your code here
        delay_ms(500);
        while(PINB.2==0)
        {}
        delay_ms(500);
        SELISIH_WAKTU();
        delay_ms(1);
        if (selisih_detik<=18000)
        {
                curah_hujan_skrg=1800/selisih_detik;
        }
        else
        {
                curah_hujan_skrg=0;
        }
        counter++;
}

// Standard Input/Output functions

// Declare your global variables here

void main(void)
{
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In
Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T
State1=T State0=T
PORTA=0x00;
DDRA=0x00;

// Port B initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In
Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T
State1=T State0=T
PORTB=0xFF;
DDRB=0x00;

// Port C initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In
Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T
State1=T State0=T
PORTC=0x00;
DDRC=0x00;

// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In
Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T
State1=T State0=T
```

```
PORTD=0x00;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=FFh
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer 1 Stopped
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer 2 Stopped
// Mode: Normal top=FFh
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: On
// INT0 Mode: Low level
// INT1: On
// INT1 Mode: Low level
// INT2: On
// INT2 Mode: Falling Edge
GICR|=0xE0;
MCUCR=0x00;
MCUCSR=0x00;
GIFR=0xE0;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;

// USART initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART Receiver: Off
// USART Transmitter: On
// USART Mode: Asynchronous
// USART Baud rate: 9600
UCSRA=0x00;
UCSRB=0x08;
UCSRC=0x86;
UBRRH=0x00;
UBRRL=0x4D;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1:
Off
ACSR=0x80;
SFIOR=0x00;

// I2C Bus initialization
i2c_init();

// DS1307 Real Time Clock initialization
// Square wave output on pin SQW/OUT: Off
// SQW/OUT pin state: 0
rtc_init(0,0,0);

// LCD module initialization
lcd_init(16);

// Global enable interrupts
#asm("sei")

while(1)
{
BACA_RTC();
delay_ms(10);
temp_tanggal=dd;
delay_ms(10);
HITUNG_DETIK();
konversi_sebelumnya=konversi_detik;
delay_ms(10);
while (1)
{
// Place your code here
      BACA_RTC();
      delay_ms(1);
      if ((m==0)&&(s==0))
      {
            RUTIN_SIMPAN();
      }
      sprintf(baris1,"%02u:%02u
%02u/%02u/%02u",h,m,dd,mm,yy);
      sprintf(baris2,"%03i    %03.2f
mm",counter,curah_hujan_skrg);
      TULIS_LCD();
      PORTB=0xFF;
      GICR|=0b11100000;
}
}
}
```

LAMPIRAN IV

LIST PROGRAM DELPHI

UNIT 1

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Variants,
Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, CPort, ExtCtrls, Menus, DB,
ADODB, Grids, DBGrids,
  Mask, DBCtrls;

type
  TForm1 = class(TForm)
    Panel1: TPanel;
    ComPort1: TComPort;
    Memo1: TMemo;
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    Button4: TButton;
    GroupBox1: TGroupBox;
    GroupBox2: TGroupBox;
    Edit1: TEdit;
    Label1: TLabel;
    Label2: TLabel;
    Edit2: TEdit;
    Label3: TLabel;
    Edit3: TEdit;
    Label4: TLabel;
    Edit4: TEdit;
    Button5: TButton;
    MainMenu1: TMainMenu;
    Grafik1: TMenuItem;
    ADOConnection1: TADOConnection;
    ADOTable1: TADOTable;
    DataSource1: TDataSource;
    DBGrid1: TDBGrid;
    DBEdit1: TDBEdit;
    DBEdit2: TDBEdit;
    DBEdit3: TDBEdit;
    DBEdit4: TDBEdit;
    Edit5: TEdit;
    Label5: TLabel;
    DBEdit5: TDBEdit;
    OpenGrafik1: TMenuItem;
    Timer1: TTimer;
    Label6: TLabel;
    Edit6: TEdit;
    Edit7: TEdit;
    Edit8: TEdit;
    procedure mengkopy(sender:Tobject);
    procedure terima_data(Sender: TObject;
Count: Integer);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure Button5Click(Sender: TObject);
    procedure Grafik1Click(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure OpenGrafik1Click(Sender:
TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
  data:string;
  logic: boolean;
  counter:integer;
  tanggal_sebelumnya:string;
  //++++++++++++++++++++++++++++++++
  // var penangkapan data melalui port serial

data_parsing,data_olahan,data_tahun,data_bula
n,    data_tanggal,data_jam,data_counter:string;
pos_S,pos_T,pos_B,pos_G,pos_J,pos_C:integer
;
//**************************************
  // var create directory
  p_dir_name_lama,p_dir_name_baru : pchar;
  P_temp:pchar;

dir_name,dir_name_lama,dir_name_baru,name
_temp:string;
  waktu:string;
  pemisahan:string;
  i_:integer;
//**************************************
implementation

uses u_grafik, u_lihat_grafik;

//**************************************
  // FORMAT PENGIRIMAN
  // S <tahun> T <bulan> B <tanggal> G <Jam>
J <counter> C
//================================
{$R *.dfm}
```

```
procedure TForm1.terima_data(Sender:
TObject; Count: Integer);
var data_serial:string;
begin
 ComPort1.ReadStr(data_serial,count);
 data:=data+data_serial;
 data_parsing:=data;
 Memo1.Text:=data;
 pos_S:=pos('S',data_parsing);
 pos_T:=pos('T',data_parsing);
 pos_B:=pos('B',data_parsing);
 pos_G:=pos('G',data_parsing);
 pos_J:=pos('J',data_parsing);
 pos_C:=pos('C',data_parsing);
 if ((pos_S<>0)and(pos_C<>0)) then
 begin
  delete(data,1,pos_C);

data_tahun:=copy(data_parsing,pos_S+1,pos_T-
pos_S-1);

data_bulan:=copy(data_parsing,pos_T+1,pos_B
-pos_T-1);

data_tanggal:=copy(data_parsing,pos_B+1,pos_
G-pos_B-1);

data_jam:=copy(data_parsing,pos_G+1,pos_J-
pos_G-1);

data_counter:=copy(data_parsing,pos_J+1,pos_
C-pos_J-1);
//================================
   if (tanggal_sebelumnya<>data_tanggal)then
     begin
      ADOConnection1.Connected:=false;
      ADOTable1.Active:=false;
      createdir(dir_name+'\temp');
      MoveFile(p_dir_name_baru,p_temp);
      mengkopy(sender);
ADOConnection1.ConnectionString:=dir_name
_baru;
      Form2.Series1.Clear;
      tanggal_sebelumnya:=data_tanggal;
     end;
//================================
   ADOConnection1.Connected:=true;
   ADOTable1.Active:=true;
   ADOTable1.Edit;
   DBEdit1.Text:=data_tahun;
   Edit1.Text:=DBEdit1.Text;
   DBEdit2.Text:=data_bulan;
   Edit2.Text:=DBEdit2.Text;
   DBEdit3.Text:=data_tanggal;
   Edit3.Text:=DBEdit3.Text;
   DBEdit4.Text:=data_jam;
   Edit4.Text:=DBEdit4.Text;
   DBEdit5.Text:=data_counter;
   Edit5.Text:=DBEdit5.Text;
   ADOTable1.Append;
Form2.Chart1.BottomAxis.DateTimeFormat:=
DateToStr(date);
Form2.Series1.AddXY(strtoint(data_jam),strtoi
nt(data_counter),'',clred);
 end;
end;
procedure TForm1.Button1Click(Sender:
TObject);
 begin
 If (button1.Caption='Connect')then
 begin
    ComPort1.Connected:=true;
    Button1.Caption:='Disconnect';
 end
 else
 begin
    ComPort1.Connected:=false;
    Button1.Caption:='Connect';
 end;
end;
procedure TForm1.Button2Click(Sender:
TObject);
begin
ComPort1.ShowSetupDialog;
end;
procedure TForm1.Button4Click(Sender:
TObject);
begin
close;
end;
procedure TForm1.Button5Click(Sender:
TObject);
begin
 data_parsing:=Memo1.Text;
 pos_S:=pos('S',data_parsing);
 pos_T:=pos('T',data_parsing);
 pos_B:=pos('B',data_parsing);
 pos_G:=pos('G',data_parsing);
 pos_J:=pos('J',data_parsing);
 pos_C:=pos('C',data_parsing);
 delete(data_parsing,1,pos_C);

Edit1.Text:=copy(Memo1.Text,pos_S+1,pos_T-
pos_S-1);

Edit2.Text:=copy(Memo1.Text,pos_T+1,pos_B
-pos_T-1);

Edit3.Text:=copy(Memo1.Text,pos_B+1,pos_G
-pos_B-1);

Edit4.Text:=copy(Memo1.Text,pos_G+1,pos_J-
pos_G-1);
```

```
Edit5.Text:=copy(Memo1.Text,pos_J+1,pos_C-
pos_J-1);
//================================
   if (tanggal_sebelumnya<>Edit3.Text)then
    begin
      ADOConnection1.Connected:=false;
      ADOTable1.Active:=false;
      createdir(dir_name+'\temp');
      MoveFile(p_dir_name_baru,p_temp);
      mengkopy(sender);

ADOConnection1.ConnectionString:=dir_name
_baru;
      Form2.Series1.Clear;
      tanggal_sebelumnya:=Edit3.Text;
    end;
//================================
  ADOConnection1.Connected:=true;
  ADOTable1.Active:=true;
  ADOTable1.Edit;
  DBEdit1.Text:=Edit1.Text;
  DBEdit2.Text:=Edit2.Text;
  DBEdit3.Text:=Edit3.Text;
  DBEdit4.Text:=Edit4.Text;
  DBEdit5.Text:=Edit5.Text;
  ADOTable1.Append;
Form2.Series1.AddXY(StrToInt(Edit4.Text),Str
ToInt(Edit5.Text),",clred );
  Memo1.Clear;
end;
procedure          TForm1.Grafik1Click(Sender:
TObject);
begin
Form2.Show;
end;
procedure          TForm1.Timer1Timer(Sender:
TObject);
begin
Label6.Caption:=TimeToStr(time);
if (Label6.Caption = Edit6.Text)and(logic =
true)then
  begin
    logic:=false;
    Memo1.Text:=IntToStr(counter);
    counter:=counter+1;
  end;
if (Label6.Caption = Edit7.Text)and(logic =
false)then
  begin
    logic:=true;
  end;
end;
procedure          TForm1.FormCreate(Sender:
TObject);
begin
  counter:=1;
  logic:=true;
  dir_name:=GetCurrentDir;
  CreateDir(dir_name+'\temp');
  mengkopy(sender);
  tanggal_sebelumnya:='0';
end;
procedure Tform1.mengkopy(sender:Tobject);
begin
dir_name_lama:=dir_name+'\data_curah_hujan.
mdb';
  //waktu:=DateToStr(date);
//pemisahan:=copy(waktu,1,2)+copy(waktu,4,2)
+copy(waktu,7,4);
  //waktu:=timetostr(time);
//pemisahan:=pemisahan+'_'+copy(waktu,length
(waktu)-1,2)+copy(waktu,length(waktu)-
4,2)+copy(waktu,1,length(waktu)-6);
//pemisahan:=Edit3.Text+'_'+Edit2.Text+'_'+Ed
it1.Text;
pemisahan:=data_tanggal+'_'+data_bulan+'_'+d
ata_tahun;
dir_name_baru:=dir_name+'\'+pemisahan+'.md
b';
name_temp:=dir_name+'\temp\'+pemisahan+'.m
db';
  i_:= 1;
  p_dir_name_lama                    :=
Addr(dir_name_lama[i_]);
  p_dir_name_baru := Addr(dir_name_baru[i_]);
  p_temp:=Addr(name_temp[i_]);
CopyFile(p_dir_name_lama,p_dir_name_baru,tr
ue);
end;
procedure    TForm1.OpenGrafik1Click(Sender:
TObject);
begin
if (Form3.OpenDialog1.Execute) then
  Begin
      Form3.Series1.Clear;
      Form3.ADOConnection1.Connected:=false;
      Form3.ADOTable1.Active:=false;
Form3.ADOConnection1.ConnectionString:=Fo
rm3.OpenDialog1.FileName;
Form3.Caption:=Form3.OpenDialog1.FileName
;
      Form3.ADOConnection1.Connected:=true;
      Form3.ADOTable1.Active:=true;
      Form3.Show;
      Form3.Series1.Clear;
      while not (Form3.ADOTable1.Eof) do
      begin
Form3.Series1.AddXY(StrToFloat(Form3.DBE
dit1.Text),StrToFloat(Form3.DBEdit2.Text),",cl
red);
        Form3.ADOTable1.Next;
      end;
  End;
end;
end
```

UNIT U_GRAFIK

```
unit u_grafik;

interface

uses
  Windows, Messages, SysUtils, Variants,
Classes, Graphics, Controls, Forms,
  Dialogs, TeeProcs, TeEngine, Chart, ExtCtrls,
Series, OleCtrls, VCFI;

type
  TForm2 = class(TForm)
    Panel1: TPanel;
    Chart1: TChart;
    Series1: TBarSeries;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form2: TForm2;

implementation

{$R *.dfm}

end.
```

UNIT U_LIHAT_GRAFIK

```
unit u_lihat_grafik;

interface

uses
  Windows, Messages, SysUtils, Variants,
Classes, Graphics, Controls, Forms,
  Dialogs, TeEngine, Series, TeeProcs, Chart,
ExtCtrls, DB, ADODB,
  StdCtrls, Mask, DBCtrls;

type
  TForm3 = class(TForm)
    Panel1: TPanel;
    Chart1: TChart;
    Series1: TBarSeries;
    ADOConnection1: TADOConnection;
    ADOTable1: TADOTable;
    DataSource1: TDataSource;
    OpenDialog1: TOpenDialog;
    DBEdit1: TDBEdit;
    DBEdit2: TDBEdit;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form3: TForm3;

implementation

{$R *.dfm}

end.
```

LAMPIRAN V

DATA SHEET KOMPONEN