MONITORING POSISI OPERASI TIM SAR - SRU (*SEARCH AND RESCUE UNIT*) PADA DAERAH BENCANA DENGAN MEMANFAATKAN GPS (*GLOBAL POSITIONING SYSTEM*)

SKRIPSI

KONSENTRASI TEKNIK ELEKTRONIKA

Diajukan untuk memenuhi persyaratan
memperoleh gelar Sarjana Teknik

Disusun oleh:
SEPTIAN ENGGAR SUSANTO
NIM. 115060309111011-63

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN
UNIVERSITAS BRAWIJAYA
FAKULTAS TEKNIK
MALANG
2013

LEMBAR PERSETUJUAN

MONITORING POSISI OPERASI TIM SAR - SRU
(*SEARCH AND RESCUE UNIT*) PADA DAERAH BENCANA
DENGAN MEMANFAATKAN GPS (*GLOBAL POSITIONING SYSTEM*)

SKRIPSI

KONSENTRASI TEKNIK ELEKTRONIKA

Diajukan untuk memenuhi persyaratan
memperoleh gelar Sarjana Teknik

Disusun oleh:

SEPTIAN ENGGAR SUSANTO
NIM. 115060309111011-63

Telah diperiksa dan disetujui oleh :

Dosen Pembimbing I	Dosen Pembimbing II
Ir. M. Julius St., MS	**Eka Maulana, ST., MT., M.Eng**
NIP. 19540720 198203 1 002	**NIK. 841130 06 1 1 0280**

LEMBAR PENGESAHAN

MONITORING POSISI OPERASI TIM SAR - SRU
(*SEARCH AND RESCUE UNIT*) PADA DAERAH BENCANA
DENGAN MEMANFAATKAN GPS (*GLOBAL POSITIONING SYSTEM*)

SKRIPSI

KONSENTRASI TEKNIK ELEKTRONIKA

Diajukan untuk memenuhi persyaratan
memperoleh gelar Sarjana Teknik

Disusun oleh:

SEPTIAN ENGGAR SUSANTO
NIM. 115060309111011-63

Skripsi ini telah diuji dan dinyatakan lulus pada
tanggal 25 Juli 2013

Dosen Penguji

Ir. Nurussa'adah, M.T	**Ir. Ponco Siwindarto, M.Eng.Sc**
NIP. 19680706 199203 2 001	**NIP. 19590304 198903 1 001**

Akhmad Zainuri, S.T., M.T
NIK. 840120 06 1 1 0052

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 "Monitoring Posisi Operasi Tim SAR - SRU (*Search and Rescue Unit*) Pada Daerah Bencana Dengan Memanfaatkan GPS (*Global Positioning System*)" 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.
- Ibu Ir. Nurussa'adah, MT. selaku Ketua Kelompok Dosen Keahlian Elektronika Jurusan Teknik Elektro Universitas Brawijaya atas segala bimbingan, nasehat, pengarahan, motivasi, saran, dan masukan yang telah diberikan.
- Bapak Ir. M. Julius,St., MS selaku Dosen Pembimbing 1 atas segala bimbingan, nasehat, pengarahan, motivasi, saran, dan masukan yang telah diberikan, dan Bapak Eka Maulana, ST., MT., M.Eng 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 2011 dan SAP 2012.

- Semua rekan-rekan di Jurusan Teknik Elektro Universitas Brawijaya Angkatan 2008 – 2010.
- 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, Juli 2013

Penulis

ABSTRAK

Septian Enggar Susanto, Jurusan Teknik Elektro, Fakultas Teknik Universitas Brawijaya, Juli 2013, *Monitoring Posisi Operasi Tim SAR - SRU (Search and Rescue Unit) Pada Daerah Bencana Dengan Memanfaatkan GPS (Global Positioning System)*, Dosen Pembimbing: Ir. M. Julius St., MS dan Eka Maulana, ST., MT., M.Eng

Dalam kegiatan SAR, komunikasi mempunyai peranan yang sangat penting, salah satunya sebagai sarana komando dan pengendalian yang dimaksudkan agar pada saat terjadi musibah, SRU (*Search and Rescue Unit*) di lapangan dapat dikendalikan dan dikoordinasikan secara terpadu oleh tim SAR pemantau yaitu OSC (*On Scene Commender*) atau SMC (SAR *Mission Coordinator*). Komunikasi antar SRU maupun SRU dengan OSC/SMC selama operasi SAR menjadi faktor pendukung dalam pelaksanaan operasi SAR. Komunikasi yang digunakan biasanya adalah komunikasi suara (*voice*) yang dalam hal ini radio komunikasi VHF, HF, UHF atau telepon satelit. Oleh sebab itu sangat dituntut keandalan seluruh peralatan komunikasi SAR sebagai pendukung dalam pelaksanaan operasi SAR. Penggunaan alat komunikasi berupa tampilan visual yang efisien dalam monitoring posisi operasi tim SAR - SRU oleh tim SAR pemantau (SMC/OSC) sangat diperlukan, selain komunikasi suara (*voice*) yang telah biasa digunakan oleh tim SAR saat ini.

Pada monitoring posisi operasi tim SAR - SRU ini memanfaatkan GPS *receiver* dan SMS. Dalam sistem ini, digunakan ATmega162 sebagai pusat pengolah data dan modem GSM *wavecom* untuk komunikasi via SMS. Komputer dengan akses internet dapat menampilkan posisi tim SAR - SRU melaui Google *Map*. Selain itu, tim SAR - SRU juga bisa memberikan informasi tentang jumlah korban yang telah ditemukan di daerah bencana alam.

Kata Kunci **:** GPS *receiver*, SMS, modem GSM *wavecom*, ATmega162, *microsoft visual C# 2005*

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1 Latar Belakang

Wilayah Negara Kesatuan Republik Indonesia memiliki kondisi geologis, geografis, hidrologis, demografis dan sosiologis yang menjadikannya rawan terhadap bencana, baik bencana alam, non-alam, maupun bencana sosial. Data dan Informasi Bencana Indonesia (DIBI) BNPB (http://dibi.bnpb.go.id/) menunjukkan bahwa jumlah kejadian bencana dan korban meninggal per jenis kejadian bencana dalam periode antara tahun 1815 - 2011 terus meningkat. Dapat dikatakan bahwa dalam dua abad terakhir ini Indonesia telah mengalami ribuan bencana geologis maupun hidrometeorologis yang menimbulkan ratusan ribu korban jiwa manusia.

Trend bencana di Indonesia terus meningkat dari tahun ke tahun. Bencana hidrometeorologi seperti banjir, kekeringan, tanah longsor, puting beliung dan gelombang pasang merupakan jenis bencana yang dominan di Indonesia. Bencana hidrometeorologi terjadi rata-rata hampir 70% dari total bencana di Indonesia. Perubahan iklim global, perubahan penggunaan lahan dan meningkatnya jumlah penduduk makin memperbesar ancaman risiko bencana di Indonesia. Bencana tersebut telah menimbulkan korban jiwa dan kerugian yang besar. Pada tahun 2010, bencana di Indonesia terjadi sekitar 644 kejadian bencana. Jumlah orang meninggal mencapai 1.711. Menderita dan hilang sekitar 1.398.923 orang. Rumah rusak berat 14.639 unit, rusak sedang 2.830 unit dan rusak ringan 25.030. Dari 644 kejadian bencana tersebut, sekitar 81,5% atau 517 kejadian bencana adalah bencana hidrometerologi, sedangkan bencana geologi seperti gempa bumi, tsunami, dan gunung meletus masing-masing terjadi 13 kali (2%), 1 kali (0,2%) dan 3 kali (0,5%). Namun jumlah kerugian yang ditimbulkan oleh bencana geologi lebih besar (BNPB, 2011).

Semua musibah yang terjadi merupakan kejadian yang serba tiba-tiba, serta tidak dapat diketahui kapan dan dimana akan terjadi. Pada umumnya berakibat fatal terhadap keselamatan jiwa dan kerugian harta benda. Tetapi, dampak tersebut dapat diantisipasi dan diminimalisir jika ditangani dengan cepat, tepat, dan seksama. Oleh karena itu, kehadiran tim pencari dan penyelamat sangat

dibutuhkan jika terjadi suatu musibah. Sekarang ini dijalankan oleh sebuah organisasi bernama BASARNAS (Badan SAR Nasional).

Lingkup tugas pokok dan fungsi BASARNAS sesuai dengan PP No.36/2006-BASARNAS bertanggungjawab untuk menangani musibah kecelakaan transportasi, bencana alam, dan musibah bencana lainnya, merupakan garda depan (*front line*) dalam proses pencarian, pertolongan, dan evakuasi korban manusia dan harta benda dalam wilayah yurisdiksi NKRI hingga 200 mil laut ZEEI, di samping fungsinya sebagai koordinator seluruh potensi SAR.

Tugas BASARNAS akan dapat terlaksana dengan baik jika didukung dengan ketersediaan dan kesiapan seluruh elemen utama BASARNAS dan institusi pendukung lainnya; secara terintegrasi baik pada tingkatan substrukturnya (institusi/kelembagaan, SDM, pembiayaan), pada tingkatan infrastrukturnya (prasarana dan sarananya), maupun pada tingkatan suprastrukturnya (regulasi, peraturan, perundangan, serta kewenangan lainnya); secara sistemik dan terintegrasi.

Dalam kegiatan SAR, komunikasi mempunyai peranan yang sangat penting, salah satunya sebagai sarana komando dan pengendalian yang dimaksudkan agar pada saat terjadi musibah, SRU (*Search and Rescue Unit*) di lapangan dapat dikendalikan dan dikoordinasikan secara terpadu OSC (*On Scene Commender*) atau SMC (SAR *Mission Coordinator*). Komunikasi antar SRU maupun SRU dengan OSC/SMC selama operasi SAR menjadi faktor pendukung dalam pelaksanaan operasi SAR. Komunikasi yang digunakan dalam lapangan biasanya menggunakan sistem komunikasi suara (*voice*) yang dalam hal ini radio komunikasi VHF, HF, UHF atau telepon satelit. Oleh sebab itu sangat dituntut keandalan seluruh peralatan komunikasi SAR sebagai pendukung dalam pelaksanaan operasi SAR.

Penggunaan alat komunikasi berupa tampilan visual serta efisien dalam monitoring posisi operasi tim SAR - SRU oleh koordinator operasi SAR (SMC/OSC) sangat diperlukan, selain komunikasi suara (*voice*) yang telah biasa digunakan oleh tim SAR saat ini.

Tujuan akhir yang diharapkan dalam penelitian ini adalah merencanakan dan merealisasikan suatu alat yang bisa memonitoring posisi untuk membantu tim

SAR - SRU yang sedang beroperasi melakukan pencarian dan pertolongan korban pada daerah bencana alam. Tim SAR pemantau (SMC/OSC) dapat mengetahui posisi tim SAR - SRU yang sedang bertugas melalui handphone dan komputer via SMS dengan jaringan GSM. Komputer dengan akses internet dapat menampilkan posisi tim SAR - SRU melaui Google *Map*. Selain itu, tim SAR - SRU juga bisa memberikan informasi tentang jumlah korban yang telah ditemukan di daerah bencana alam.

1.2 Rumusan Masalah

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

1) Bagaimana merancang dan membuat sistem elektronika untuk proses akuisisi data dari GPS.
2) Bagaimana merancang dan membuat rangkaian antarmuka mikrokontroler dengan *modem wavecom.*
3) Bagaimana merancang dan membuat perangkat lunak sistem mikrokontroler sebagai pembaca data GPS, *input keypad*, pengolah data untuk sistem transmisi melalui SMS pada *modem wavecom* dan menampilkan data di LCD.
4) Bagaimana merancang dan membuat perangkat lunak komputer agar dapat mengirim SMS dan membaca SMS data GPS serta menampilkannya secara visual dengan memanfaatkan Google *Map*.

1.3 Batasan Masalah

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

1) Parameter yang ditampilkan secara visual adalah posisi tim SAR - SRU dan pengiriman jumlah korban yang ditemukan oleh Tim SAR - SRU.
2) GPS *Receiver* yang digunakan adalah SkyNav SKM53 GPS *module*.
3) Menggunakan *modem* w*avecom* M1306B RS232 dan *modem* w*avecom* M1306B USB untuk pengirim dan penerima SMS dengan komunikasi *serial* RS232 dan USB.

4) Daerah bencana diasumsikan masih terjangkau jaringan GSM.
5) Pengaksesan melalui SMS sesuai format dan nomor yang telah ditentukan.
6) Mikrokontroler yang digunakan adalah AVR ATMega162.
7) Tampilan visual komputer menggunakan program *Microsoft Visual C# 2005* dengan memanfaatkan Google *Map*.

1.4 Tujuan

Penelitian ini bertujuan untuk merancang dan membuat sistem yang dapat memonitoring posisi untuk membantu tim SAR - SRU yang sedang beroperasi melakukan pencarian dan pertolongan korban pada daerah bencana alam. Tim SAR pemantau (SMC/OSC) dapat mengetahui posisi tim SAR - SRU yang sedang bertugas melalui handphone dan komputer via SMS dengan jaringan GSM. Komputer dengan akses internet dapat menampilkan posisi tim SAR - SRU melaui Google *Map*. Selain itu, tim SAR - SRU juga bisa memberikan informasi tentang jumlah korban yang telah ditemukan di daerah bencana alam.

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 perancangan dan pembuatan alat.

BAB III Metodologi Penelitian

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

BAB IV Perancangan dan Pembuatan Alat

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

BAB V Pengujian dan Analisis

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

SAR merupakan singkatan dari *Search And Rescue* yang mempunyai arti usaha untuk melakukan pencarian, pertolongan dan penyelamatan terhadap keadaan darurat yang dialami baik manusia maupun harta benda yang berharga lainnya.

SAR merupakan kegiatan kemanusiaan yang dilakukan secara suka rela dan tanpa pamrih dan merupakan kewajiban moril bagi setiap individu yang terlatih untuk melakukan pertolongan terhadap korban musibah secara cepat, tepat dan efisien dengan memanfaatkan sumber daya/potensi yang ada, baik sarana dan prasarana maupun manusia yang ada.

Operasi SAR akan berhasil dengan baik jika berbagai potensi yang bergabung dalam operasi SAR dikendalikan secara terpadu, melaksanakan operasi SAR sesuai dengan rencana operasi yang telah di buat, sehingga pelaksanaan operasi SAR tidak berjalan masing-masing, organisasi operasi adalah sebagai berikut :

1) **SC** (SAR *Coordinator*) dijabat oleh Kepala Badan SAR Nasional.
2) **Asisten SC** (Asisten SAR *Coordinator*) terdiri dari :
   - **Asisten Operasi** merupakan pejabat SAR yang mempunyai tugas pokok dan fungsi di bidang operasi SAR dan memiliki kualifikasi teknis SAR dan berpengalaman dalam penyelenggaraan operasi SAR.
   - **Asisten Intelijen** merupakan pejabat SAR yang mempunyai tugas pokok dan fungsi di bidang SAR dan memiliki pengetahuan dan kemampuan dalam pengumpulan, pengolahan, dan pendistribusian data dalam penyelenggaraan operasi SAR.
   - **Asisten Komunikasi** merupakan pejabat SAR yang mempunyai tugas pokok dan fungsi di bidang komunikasi dan memiliki kecakapan dan pengalaman dalam komunikasi SAR.
   - **Asisten Administrasi dan Logistik** merupakan pejabat SAR yang mempunyai tugas pokok dan fungsi di bidang sarana dan prasarana untuk mendukung penyelenggaraan operasi SAR.

3) **SMC** (SAR *Mission Coordinator*) dijabat oleh Kepala Kantor SAR setempat.
4) **Staf SMC** (Staf SAR *Mission Coordinator*) ditunjuk oleh dan bertanggungjawab kepada SMC. Staf SMC meliputi :
   - **Staf Operasi** merupakan petugas dari Kantor SAR yang memiliki kualifikasi SAR *Planner* dan berpengalaman dalam penyelenggaraan operasi SAR.
   - **Staf Intelijen** merupakan petugas dari Kantor SAR yang memiliki kualifikasi SAR *Planner*, berpengalaman dalam pengumpulan dan analisis data untuk proses perencanaan dalam pelaksanaan operasi SAR.
   - **Staf Komunikasi** merupakan petugas dari Kantor SAR yang memiliki kualifikasi operator komunikasi SAR, berpengalaman dalam penggunaan dan penguasaan alat komunikasi dan elektronika dalam kegiatan SAR.
   - **Staf Administrasi dan Logistik** merupakan petugas dari Kantor SAR yang memiliki kualifikasi administrasi SAR dan pengelolaan *logistic* dalam kegiatan SAR.
5) **OSC** (*On Scene Coordinator*) dijabat oleh petugas SAR yang ditunjuk oleh SMC untuk mengkoordinasikan dan mengendalikan SRU dalam *search area*.
6) **SRU** (*Search and Rescue Unit*) yaitu petugas SAR yang terlatih dan sarana pendukung yang sesuai dengan kebutuhan operasi SAR. SRU dapat berasal dari berbagai organisasi/instansi yang ingin berpartisipasi dalam kegiatan operasi SAR.

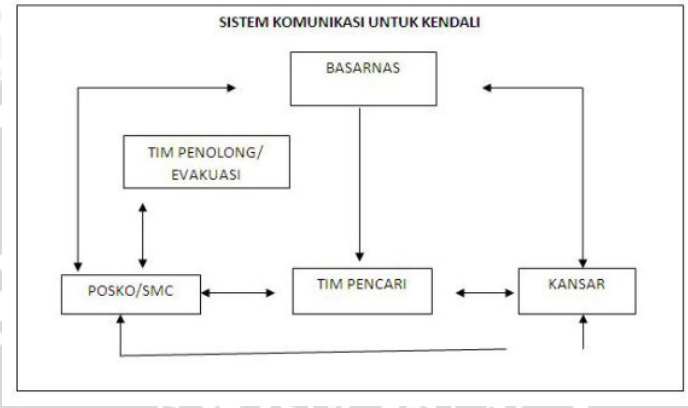
Sumber : www.basarnas.go.id

2.1 Sistem Komunikasi SAR (*Search And Rescue*)

Dalam kegiatan SAR, komunikasi mempunyai peranan yang sangat penting seperti Sarana Pengindera Dini (*early detecting*), Sarana Koordinasi (*early warning*), Sarana Komando dan Pengendali (*command and control*) dan Sarana Administrasi dan Logistik.

Sebagai Sarana komando dan pengendalian dimaksudkan agar pada saat terjadi musibah, SRU (*Search and Rescue Unit*) di lapangan dapat dikendalikan dan dikoordinasikan secara terpadu OSC (*On Scene Commender*) atau SMC (SAR *Mission Coordinator*). Komunikasi antar SRU maupun SRU dengan OSC / SMC

selama operasi SAR menjadi faktor pendukung dalam pelaksanaan operasi SAR. Komunikasi yang digunakan dalam lapangan biasanya menggunakan sistem komunikasi suara (*voice*) yang dalam hal ini radio komunikasi VHF, HF, UHF atau telepon satelit. Oleh sebab itu sangat dituntut keandalan seluruh peralatan komunikasi SAR sebagai pendukung dalam pelaksanaan operasi SAR. Sistem komunikasi untuk kendali ditunjukkan dalam Gambar 2.1.



Gambar 2.1. Sistem Komunikasi Untuk Kendali SAR

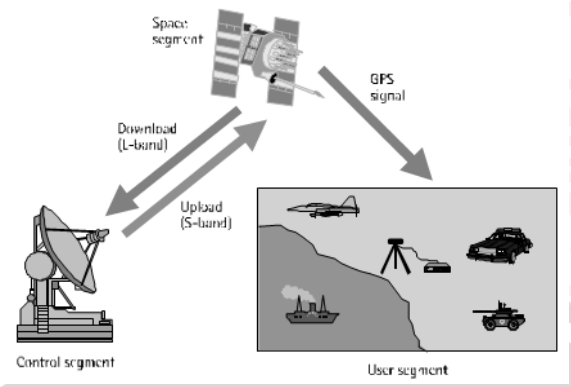
Sumber : www.basarnas.go.id

2.2 Sistem Navigasi GPS

GPS adalah sistem radio navigasi dan penentuan posisi dengan menggunakan satelit navigasi yang dimiliki dan dikelola oleh Departemen Pertahanan Amerika Serikat. Nama formalnya adalah **NAVSTAR GPS** (*Navigation Satellite Timing and Ranging Global Positioning System*). Sistem ini digunakan untuk memberikan informasi mengenai posisi, waktu dan kecepatan kepada siapa saja secara global tanpa ada batasan waktu dan cuaca. Satelit GPS pertama diluncurkan pada tahun 1978 dan secara resmi sistem GPS dinyatakan operasional pada tahun 1994.

2.2.1 Segmen Penyusun Sistem GPS

Sistem GPS tediri atas tiga segmen utama, yaitu segmen angkasa (*space segment*), segmen sistem kontrol (*control system segment*), dan segmen pengguna (*user segment*) (Abidin, 2000). Segmen GPS ditunjukkan dalam Gambar 2.2.



Gambar 2.2. Segmen GPS

Sumber : Ahmad El-Rabbany, 2002

- Segmen Angkasa

  Segmen angkasa terdiri dari 24 buah satelit GPS yang secara kontinyu memancarkan sinyal – sinyal yang membawa data kode dan pesan navigasi yang berguna untuk penentuan posisi, kecepatan dan waktu. Satelit-satelit tersebut ditempatkan pada enam bidang orbit dengan periode orbit 12 jam dan ketinggian orbit 20.200 km di atas permukaan bumi. Keenam orbit tersebut memiliki jarak spasi yang sama dan berinklinasi 55° terhadap ekuator dengan masing-masing orbit ditempati oleh empat buah satelit dengan jarak antar satelit yang tidak sama.

- Segmen Sistem Kontrol

  Segmen sistem kontrol terdiri dari *Master Control Station* (MCS), *Ground Station*, dan beberapa *Monitor Station* (MS) yang berfungsi untuk mengontrol dan memonitor pergerakan satelit.

- Segmen Pengguna

  Segmen pengguna terdiri dari para pengguna satelit GPS baik yang ada di darat, laut maupun udara. Dalam hal ini *receiver* GPS dibutuhkan untuk menerima dan memproses sinyal-sinyal dari GPS untuk digunakan dalam penentuan posisi, kecepatan, dan waktu.

2.2.2 Sinyal GPS

Sinyal GPS yang dipancarkan oleh satelit-satelit GPS menggunakan *band* frekuensi L pada spektrum gelombang elektromagnetik. Setiap satelit GPS memancarkan dua (2) gelombang pembawa yaitu L1 dan L2 yang berisi data kode dan pesan navigasi.

Pada dasarnya sinyal GPS terdiri dari tiga komponen, yaitu: informasi jarak (kode), informasi posisi satelit (*navigation message*), dan gelombang pembawanya (*carrier wave*) (Abidin, 2000).

2.2.2.1 Informasi Jarak

Informasi jarak yang dikirimkan oleh satelit GPS terdiri dari dua buah kode PRN (*Pseudo Random Noise*) yaitu kode-C/A (*Coarse Acquisition/Clear Access*) yang dimodulasikan pada gelombang pembawa L1 dan kode-P(Y) (*Private*) yang dimodulasikan baik pada gelombang pembawa L1 maupun L2. Kedua kode tersebut disusun oleh rangkaian kombinasi bilangan-bilangan biner (0 dan 1).

Setiap satelit GPS mempunyai struktur kode yang unik dan berbeda antara satu satelit dengan satelit lainnya yang memungkinkan *receiver* GPS untuk membedakan sinyal-sinyal yang datang dari satelit-satelit GPS yang berbeda. Sinyal-sinyal tersebut dapat dibedakan oleh *receiver* dengan menggunakan teknik yang dinamakan CDMA (*Code Division Multiple Accsess*) (Kapplan et.al, 1996).

2.2.2.2 Informasi Posisi

Pesan navigasi yang dibawa oleh sinyal GPS terdiri dari informasi *ephemeris* (orbit) satelit yang biasa disebut *broadcast ephemeris* yang terdiri dari parameter waktu, parameter orbit satelit dan parameter perturbasi dari orbit satelit (Abidin, 2000). Parameter – parameter tersebut digunakan untuk menentukan koordinat dari satelit.

Disamping *broadcast ephemeris*, pesan navigasi juga berisi almanak satelit yang memberikan informasi tentang orbit nominal satelit yang berguna bagi *receiver* dalam proses akuisasi awal data satelit maupun bagi para pengguna dalam perencanaan waktu pengamatan yang optimal (Abidin, 2000). Informasi lain yang dibawa oleh pesan navigasi adalah koefisien koreksi jam satelit, parameter koreksi ionosfer, status konstelasi satelit dan informasi kesehatan satelit.

2.2.2.3 Gelombang Pembawa

Kode dan pesan navigasi agar dapat mencapai pengamat harus dimodulasikan terlebih dahulu pada gelombang pembawa. Gelombang pembawa yang digunakan terdiri atas dua gelombang , yaitu gelombang L1 dan L2. Gelombang L1 (1575.42 Mhz) membawa kode-P(Y) dan kode-C/A sedangkan

gelombang L2(1227.60 Mhz) hanya membawa kode-P(Y) saja. Teknik modulasi yang digunakan dalam sinyal GPS adalah BPSK (*Binary Phase Shift Keying*) yang menggunakan modulasi fase (Kapplan et.al, 1996).

2.2.3 Penentuan Posisi Absolut dengan GPS

Penentuan posisi dengan GPS adalah penentuan posisi tiga dimensi yang dinyatakan dalam sistem koordinat kartesian (X,Y,Z) dalam datum WGS (*World Geodetic System*) 1984. Untuk keperluan tertentu, koordinat kartesian tersebut dapat dikonversi ke dalam koordinat geodetik (φ,λ,h). Titik yang akan ditentukan posisinya dapat diam (*static positioning*) maupun bergerak (*kinematic positioning*). Penentuan posisi absolut merupakan metode penentuan posisi yang paling mendasar dan paling banyak digunakan untuk aplikasi-aplikasi yang tidak memerlukan tingkat ketelitian posisi yang tinggi dan tersedia secara instan (*real-time*) seperti pada aplikasi navigasi wahana bergerak (darat, laut dan udara).

2.2.3.1 Prinsip Penentuan Posisi Absolut dengan GPS

Prinsip dasar penentuan posisi absolut dengan GPS adalah dengan reseksi jarak ke beberapa satelit GPS sekaligus yang koordinatnya telah diketahui (Abidin, 2000). Pada penentuan posisi absolut dengan data *pseudorange*, jarak pengamat (*receiver*) ke satelit GPS ditentukan dengan mengukur besarnya waktu tempuh sinyal GPS dari satelit ke *receiver* pengamat.

Waktu tempuh ditentukan dengan menggunakan teknik korelasi kode (*code correlation technique*) dimana sinyal GPS yang datang dikorelasikan dengan sinyal replika yang diformulasikan dalam *receiver*. Jarak dari *receiver* ke pengamat kemudian dapat ditentukan dengan mengalikan waktu tempuh dengan kecepatan cahaya. Karena ada perbedaan waktu pada jam satelit dan jam *receiver* maka data jarak yang diperoleh bukan merupakan jarak yang sebenarnya melainkan jarak *pseudorange* yang persamaannya dapat dirumuskan sebagai berikut (Kapplan et.al, 1996) :

$$\rho = r + c(tu - \delta t) \quad \text{.....(2-1)}$$

$$r = c\,(Tu - Ts) = c\Delta t \quad \text{.....(2-2)}$$

dengan :

ρ = jarak *pseudorange*

r = jarak geometrik

c = kecepatan cahaya

Ts = Waktu GPS pada saat sinyal meninggalkan satelit

Tu = Waktu GPS pada saat sinyal mencapai *receiver*

tu = Perbedaan waktu jam *receiver* dengan waktu GPS

δt = Perbedaan waktu jam satelit dengan waktu GPS

Untuk mendapatkan posisi tiga dimensi (X,Y,Z) maka terdapat empat parameter yang harus diestimasi / dipecahkan yaitu:

- parameter koordinat (X,Y,Z)
- parameter kesalahan jam *receiver* GPS.

Oleh sebab itu untuk memecahkan keempat parameter tersebut dibutuhkan pengamatan terhadap minimal empat (4) buah satelit secara simultan yang dirumuskan dalam persamaan berikut (Kapplan et.all, 1996).

$$\rho_1 = \sqrt{(x_1 - x_u)^2 + (y_1 - y_u)^2 + (z_1 - z_u)^2} + ct_u \quad \text{.....(2-3)}$$

$$\rho_2 = \sqrt{(x_2 - x_u)^2 + (y_2 - y_u)^2 + (z_2 - z_u)^2} + ct_u \quad \text{.....(2-4)}$$

$$\rho_3 = \sqrt{(x_3 - x_u)^2 + (y_3 - y_u)^2 + (z_3 - z_u)^2} + ct_u \quad \text{.....(2-5)}$$

$$\rho_4 = \sqrt{(x_4 - x_u)^2 + (y_4 - y_u)^2 + (z_4 - z_u)^2} + ct_u \quad \text{.....(2-6)}$$

dengan :

ρ = jarak *pseudorange*

xi,yi = koordinat satelit i

xu,yu = koordinat pengamat

ctu = koreksi kesalahan jam *receiver*

2.2.3.2 Ketelitian Posisi Absolut

Ketelitian posisi absolut GPS sangat bergantung pada tingkat ketelitian data *pseudorange* serta geometri dari satelit pada saat pengukuran (Kapplan et.al, 1996).

Ketelitian posisi GPS = Geometri Satelit x Ketelitian *Pseudorange*……(2-7)

Faktor – faktor yang mempengaruhi ketelitian penentuan posisi dengan GPS adalah sebagai berikut :

- Satelit, seperti kesalahan *ephemeris* (orbit) dan jam satelit.

- Medium propagasi, seperti bias ionosfer dan bias troposfer yang mempengaruhi kecepatan (memperlambat) dan arah perambatan sinyal GPS.
- *Receiver* GPS, seperti kesalahan jam *receiver*, kesalahan yang terkait dengan antena dan *noise* (derau). Kesalahan-kesalahan ini bergantung pada kualitas dari *receiver* GPS dan berbanding lurus dengan harga dari *receiver* GPS, semakin tinggi harga *receiver*, semakin tinggi kualitasnya.
- Lingkungan sekitar *receiver* GPS, seperti *multipath* yaitu fenomena dimana sinyal GPS yang tiba di antena *receiver* GPS merupakan resultan dari sinyal yang langsung dari GPS dan sinyal yang dipantulkan oleh benda-benda di sekeliling *receiver* GPS.

Dalam kaitannya dengan ketelitian penentuan posisi dengan GPS, terdapat dua level ketelitian yang diberikan oleh GPS, yaitu SPS (*Standard Positioning Service*) dan PPS (*Precise Positioning Service*). SPS merupakan layanan standar yang diberikan oleh GPS kepada siapa saja tanpa dipungut biaya. Tingkat ketelitian yang diberikan oleh layanan ini adalah ± 100 m pada saat kebijakan SA (*Selective Availability*) masih berlaku dan ± 20 m setelah kebijakan SA dihapus (1 Mei 2000, 00:00 EDT). Sedangkan PPS merupakan jenis layanan yang hanya dikhususkan untuk pihak militer Amerika dan pihak-pihak lain yang diizinkan dengan tingkat ketelitian yang lebih tinggi dari tingkat ketelitian SPS.

2.2.4 Format Data Keluaran GPS

Receiver GPS memiliki format keluaran sebanyak lima (5) jenis yaitu NMEA 0180, NMEA 0182, NMEA 0183, AVIATION, dan PLOTTING (Sitio, 1997). Format data tersebut ditetapkan oleh NMEA (*National Maritime Electronic Association*) dan dapat dikoneksikan ke komputer melalui *port* komunikasi *serial* dengan menggunakan kabel RS-232.

Data keluaran dalam format NMEA 0183 berbentuk kalimat (*string*) yang merupakan rangkaian karakter ASCII 8 *bit*. Setiap kalimat diawali dengan satu karakter '$', dua karakter *Talker ID*, tiga karakter *Sentence ID*, dan diikuti oleh data *fileds* yang masing-masing dipisahkan oleh koma serta diakhiri oleh *optional cheksum* dan karakter *cariage return/line feed*(CR/LF). Jumlah maksimum karakter dihitung dari awal kalimat ($) sampai dengan akhir kalimat (CR/LF) adalah 82 karakter.

Format dasar data NMEA 0183 :

$aaccc,c—c*hh<CR><LF>

keterangan:

aa = *Talker ID*, menandakan jenis atau peralatan navigasi yang digunakan,

ccc = *Sentence ID*, menandakan jenis informasi yang terkandung dalam kalimat,

c—c = *data fileds*, berisi data-data navigasi hasil pengukuran,

*hh = *optional cheksum*, untuk pengecekan kesalahan (error) kalimat,

<CR><LF> = *carriage return/line feed*, menandakan akhir dari kalimat.

beberapa jenis *Talker ID* yang ada pada spesifikasi NMEA 0183 adalah:

- GP, untuk data keluaran GPS *receiver*,
- LC, untuk data keluaran Loran-C *receiver*,
- OM, untuk data keluaran Omega Navigation *receiver*.

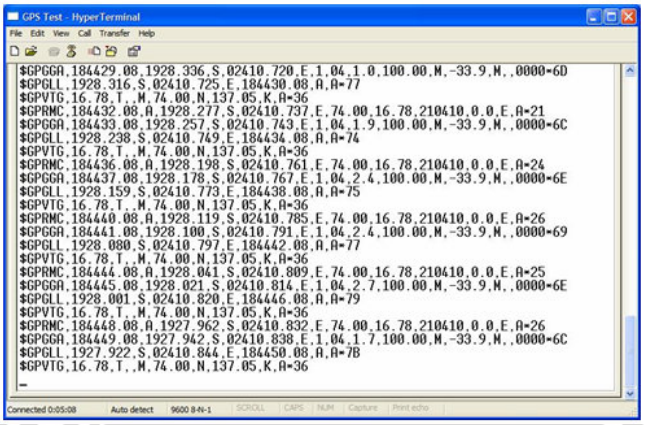
Ada banyak jenis kalimat pada protokol NMEA 0183 pada perangkat GPS *receiver*. Masing-masing jenis kalimat tersebut memiliki data-data yang berbeda. Oleh karena itu pengguna dapat menggunakan data tertentu yang sesuai dengan kebutuhan atau fungsinya. Beberapa jenis kalimat NMEA 0183 yang umum pada perangkat GPS ditunjukkan dalam Tabel 2.1.

Tabel 2.1. Kalimat NMEA 0183 pada GPS

Kalimat	Deskripsi
$GPGGA	*Global positioning system fixed data*
$GPGLL	*Geographic position - latitude / longitude*
$GPGSA	GNSS DOP *and active satellites*
$GPGSV	GNSS *satellites in view*
$GPRMC	*Recommended minimum specific* GNSS *data*
$GPVTG	*Course over ground and ground speed*

Sumber : Taryudi , 2009

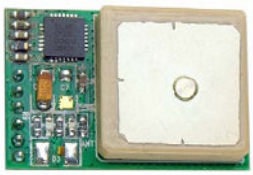
Data keluaran GPS *receiver* pada program *HyperTerminal* komputer ditunjukkan dalam Gambar 2.3.



Gambar 2.3. Data keluaran GPS *receiver* pada program *HyperTerminal*

2.2.5 GPS *Receiver*

GPS *receiver* yang digunakan pada sistem adalah SkyNav SKM53 GPS *module*. Bentuk fisik dari SkyNav SKM53 GPS *module* ditunjukkan dalam Gambar 2.4.



Gambar 2.4. SkyNav SKM53 GPS *module*
Sumber : Skylab, 2009

Fitur :

- Ultra high sensitivity: -165dBm
- Tracking (22) / Acquisition-channel receiver (66)
- WAAS/EGNOS/MSAS/GAGAN support
- NMEA protocols (default speed: 9600bps)
- Internal back-up battery and 1PPS output
- One serial port and USB port (option)
- Embedded patch antenna 18.2 x 18.2 x 4.0 mm
- Operating temperature range: -40 to 85℃
- RoHS compliant (Lead-free)
- Tiny form factor : 30mm x20mm x 11.4mm

Power Supply : Tegangan input Vcc harus 5V, arus tidak kurang dari 150mA.

UART Port : Modul mendukung satu saluran full duplex serial UART. Koneksi serial di LVTTL adalah 2.85V, jika perlu tegangan yang berbeda level, gunakan level yang sesuai. format data: X, N, 8, 1, yaitu X baud rate, no parity, delapan bit data dan satu stop bit. Standar baudrate modul diatur 9600bps. RXD0 & TXD0 direkomendasikan dengan pull up (10KΩ). Hal ini dapat meningkatkan stabilitas data serial.

Port USB : Modul ini menggunakan chip tunggal Silicon CP2102 untuk USB ke UART, sebelum menggunakannya, silahkan install driver yang sesuai.

GPS Status : Status GPS dapat dihubungkan ke LED untuk menunjukkan status sinyal GPS. LED menyala menunjukkan data GPS belum fix, LED berkedip menunjukkan data GPS sudah fix.

Protokol NMEA adalah protokol berbasis ASCII, Data dimulai dengan $ dan dengan carriage return / line feed. Data GPS dimulai dengan $GPxxx dimana xxx adalah tiga huruf pengenal dari data GPS. Data NMEA memiliki checksum, yang memungkinkan deteksi transfer data yang rusak. The SkyNav seri SKM53 mendukung beberapa data NMEA-0183 yaitu GGA, GLL, GSA, GSV, RMC VTG dan ZDA.

Kalimat NMEA 0183 *output* dari GPS *receiver* terdiri dari berbagai macam bentuk. Kalimat NMEA 0183 yang digunakan adalah tipe GPRMC (*Recommended minimum specific* GNSS *data*). Berikut ini adalah format dan penjelasan data dari tipe kalimat GPRMC :

$GPRMC,hhmmss,A,ddmm.mmmm,S,dddmm.mmmm,E,x.xx,y.yy,ddmmyy,*hh

Keterangan :

$GPRMC	: *Recommended Minimum Specific* GPS/TRANSIT *Data*
hhmmss	: waktu UTC (hh = jam, mm = menit, ss = detik)
A	: Status (A = valid, V = invalid)
ddmm.mmmm	: lintang (dd = derajat, mm.mmmm = menit)
S	: N atau S (N = utara, S = selatan)
dddmm.mmmm	: bujur (ddd = derajat, mm.mmmm = menit)
E	: E atau W (E = timur, W = barat)
x.xx	: kecepatan bergerak (dalam knots)

y.yy	: arah pergerakan
ddmmyy	: tanggal UTC (dd = tanggal, mm = bulan, yy = tahun)
*hh	: *checksum*

2.3 Mikrokontroler ATMega162

ATMega162 merupakan seri mikrokontroler CMOS 8-bit buatan Atmel, berbasis arsitektur RISC (*Reduced Instruction Set Computer*). Hampir semua instruksi dieksekusi dalam satu siklus clock. ATMega162 mempunyai 32 register *general-purpose*, *timer/counter* fleksibel dengan mode *compare*, *interrupt* internal dan eksternal, *serial* UART, *programmable Watchdog Timer*, dan *mode power saving*. ATMega162 mempunyai ADC dan PWM internal. ATMega162 juga mempunyai *In-System Programmable Flash on-chip* yang mengijinkan memori program untuk diprogram ulang dalam sistem menggunakan hubungan serial SPI. Atmega162 mempunyai *throughput* mendekati 1 MIPS per MHz membuat disainer sistem dapat mengoptimasi konsumsi daya versus kecepatan proses. Beberapa keistimewaan dari AVR ATmega162 antara lain:

1) *Advanced* RISC *Architecture*
   - *Most Single-clock Cycle Execution* - 131 *Powerful Instructions*
   - *General Purpose Working Registers* (32 x 8)
   - *Up* to 16 MIPS *Throughput at* 16 MHz
   - *On-chip* 2-*cycle Multiplier*
2) *Non-volatile Program and Data Memories*
   - *In-System Self-programmable Flash Endurance* (16K *Bytes*): 1,000 *Write/Erase Cycles*
   - *Optional Boot Code Section with Independent Lock Bits In-System Programming by On*-chip *Boot Program True Rea*d-*While-Write Operation*
   - EEPROM *Endurance* (512 *Bytes*): 100,000 *Write/Erase Cycles*
   - *Internal* SRAM : 1K *Bytes*
   - *Up to* 64K *Bytes Optional External Memory Space*
   - *Programming Lock for Software Security*
3) *Peripheral Features*
   - *Two* 8-*bit Timer/Counters with Separate Prescalers and Compare Modes*

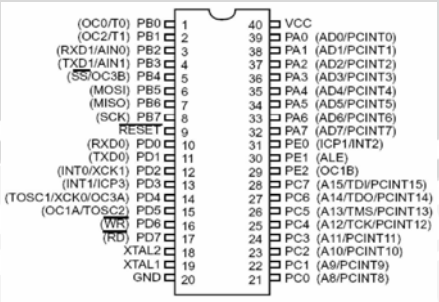
- *Two* 16-*bit Timer/Counters with Separate Prescalers*, *Compare Modes*, *and Capture Modes*
- *Real Time Counter with Separate Oscillator*
- *Six* PWM *Channels*
- *Dual Programmable Serial* USARTs
- *Master/Slave* SPI *Serial Interface*
- *Programmable Watchdog Timer with Separate On-chip Oscillator*
- *On-chip Analog Comparator*

4) *Special Microcontroller Features*
   - *Power-on Reset and Programmable Brown-out Detection*
   - *Internal Calibrated* RC *Oscillator*
   - *External and Internal Interrupt Sources*
   - *Five Sleep Modes*: *Idle*, *Power-save*, *Power-down*, *Standby*, *and Extended Standby*

5) I/O *and Packages*
   - *Programmable* I/O *Lines* (35)
   - PDIP (40-*pin*), 44-*lead* TQFP, *and* 44-*pad* MLF

6) *Operating Voltages*
   - ATmega162V : 1.8 - 3.6V
   - ATmega162L : 2.7 - 5.5V
   - ATmega162 : 4.5 - 5.5V

Untuk memaksimalkan performa dan paralelisme, ATMega162 menggunakan arsitektur Harvard (memori dan bus terpisah untuk program dan data). Gambar 2.5 menunjukkan pin-pin pada ATMega162 dengan kemasan 40-pin DIP (*Dual In-Line Package*).



Gambar 2.5. Pin-Pin pada ATMega162 dengan Kemasan 40-Pin DIP (*Dual In-Line Package*)
Sumber: Atmel, 2009: 2

Adapun fungsi dari pin-pin yang terdapat pada Mikrokontroler ATMega162 dijelaskan sebagai berikut :

a) VCC, suplai tegangan digital
b) GND, pin *ground*
c) PORT A (PA0 – PA7). PORT A merupakan merupakan port I/O 8-*bit bidirectional* yang dilengkapi dengan resistor *pull-up* internal (dapat dipilih untuk tiap *bit*). Selain sebagai port I/O, PORT A juga mempunyai fungsi lain seperti antarmuka memori eksternal dan pin *change interrupt*.
d) PORT B (PB0 – PB7). PORT B merupakan port I/O 8-*bit bidirectional* dengan resistor *pull-up* internal (dapat dipilih untuk tiap *bit*). Selain itu, PORT B juga mempunyai fungsi lain yaitu *Serial Peripheral Interface* (SPI), *Analog Comparator, input/output Timer/Counter*.
e) PORT C (PC0 – PC7). PORT C merupakan port I/O 8-*bit bidirectional* dengan resistor *pull-up* internal (dapat dipilih untuk tiap *bit*). Selain itu PORT C juga mempunyai fungsi lain yaitu JTAG, antarmuka memori eksternal, dan pin *change interrupt*.
f) PORT D (PD0 – PD7). PORT D merupakan port I/O 8-*bit bidirectional* dengan resistor *pull-up* internal (dapat dipilih untuk tiap *bit*). Selain itu PORT D juga mempunyai fungsi lain yaitu USART, *interupsi eksternal, strobe memori eksternal, timer/counter.*
g) PORT E (PE0 – PE2) PORT E merupakan port I/O 3-*bit bidirectional* dengan resistor *pull-up* internal (dapat dipilih untuk tiap *bit*). Selain itu PORT E juga mempunyai fungsi lain yaitu *timer/counter*, *latch enable memori eksternal, interupsi eksternal.*
h) RESET, berfungsi untuk mereset mikrokontroler jika diberikan sinyal *active low* dalam selang waktu tertentu.
i) XTAL1, input ke inverting *oscillator* amplifier dan ke rangkaian detak internal.
j) XTAL2, output dari inverting *oscillator amplifier*.

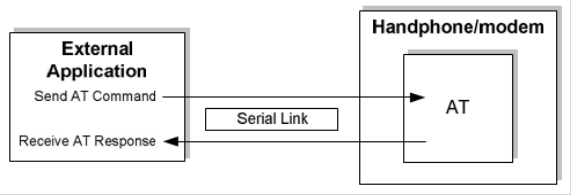
Universal synchronous dan *asynchronous* pemancar dan penerima *serial* adalah suatu alat komunikasi serial sangat fleksibel. Mikrokontroller ATMega162 memiliki dua buah *port* USART untuk komunikasi *serial*, yaitu USART0 dan

USART1. Fasilitas komunikasi serial USART mikrokontroler ini memiliki fitur sebagai berikut:

1) *Operasi full duplex* (register penerima dan pengirim *serial* dapat berdiri sendiri)
2) *Operasi Asychronous* atau *synchronous*
3) *Master* atau *slave* mendapat *clock* dengan operasi *synchronous*
4) Pembangkit *baud rate* dengan resolusi tinggi
5) Dukung *frames serial* dengan 5, 6, 7, 8 atau 9 *data bit* dan 1 atau 2 *stop bit*
6) Tahap *odd* atau *even parity* dan *parity check* didukung oleh *hardware*
7) Pendeteksian data *overrun*
8) Pendeteksi *framing error*
9) Pemfilteran gangguan (*noise*) meliputi pendeteksian *bit false start* dan pendeteksian *low pass filter* digital
10) Tiga *interrupt* terdiri atas TX *complete*, TX *data register empty*, dan RX *complete*
11) Mode komunikasi *multi-processor*
12) Mode komunikasi *double speed asynchronous*

2.4 AT *Command*

AT *command* adalah sebuah bahasa atau instruksi yang digunakan untuk komunikasi antara telepon seluler atau modem dengan komputer, mikrokontroler dan perangkat lainnya melalui melalui *link serial.* AT *Command* ditunjukkan dalam Gambar 2.6.



Gambar 2.6. AT *Command*

Komputer atau mikrokontroler dapat memberikan AT *command* kepada telepon seluler melalui kabel atau *bluetooth*. Dengan penggunaan AT *command*, komputer atau mikrokontroler dapat melakukan suatu perintah seperti mengirim pesan, membaca pesan, menambah item pada daftar telepon dan sebagainya. AT

command merupakan pengembangan dari perintah yang dapat diberikan kepada modem Heyes. Dinamakan AT *command* karena semua perintah diawali dengan karakter A dan T yang merupakan singkatan dari *Attention*. Tiap telepon seluler atau modem mempunyai AT *Command* yang sama. Namun ada beberapa instruksi AT *Command* yang ditambahkan sendiri pada ponsel atau modem oleh vendor pembuatnya. Instruksi AT *Command* dikelompokkan menjadi empat jenis. Jenis-jenis perintah AT *Command* ditunjukkan dalam Tabel 2.2.

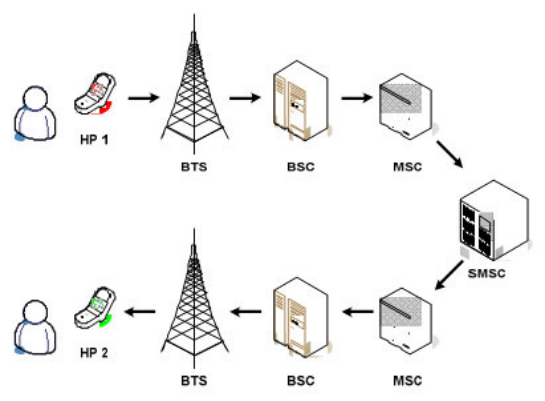
Tabel 2.2. Jenis-jenis Perintah AT *Command*

Perintah	AT Command	Keterangan
Read Command	AT+Cxxx?	Perintah ini digunakan untuk mengetahui nilai parameter yang sedang aktif pada terminal.
Test Command	AT+Cxxx=?	Perintah ini digunakan untuk mengetahui parameter-parameter dan jangkauan nilainya berdasarkan *write command* atau proses internal.
Write Command	AT+Cxxx=<...>	Perintah ini digunakan untuk mengubah nilai parameter.
Execute Command	AT+Cxxx	Perintah ini untuk mengetahui nilai parameter dari terminal yang tidak dapat diubah nilainya.

Sumber : Taryudi, 2009

2.4.1 *Short Message Service* (SMS)

Short Message Service (SMS) adalah protokol layanan pertukaran pesan *text* singkat (sebanyak 160 karakter per pesan) antar telepon. SMS ini pada awalnya adalah bagian dari standar teknologi seluler GSM, yang kemudian juga tersedia di teknologi CDMA, telepon rumah PSTN, dan lainnya. Gambar 2.7 menunjukkan alur pengiriman SMS.



Gambar 2.7. Alur Pengiriman SMS

Keterangan :

BTS : *Base Transceiver Station*

BSC : *Base Station Controller*

MSC : *Mobile Switching Center*

SMSC : *Short Message Service Center*

Alur pengiriman SMS adalah sebagai berikut :

Ketika pengguna mengirim SMS, maka pesan dikirim ke MSC melalui jaringan seluler yang tersedia yang meliputi tower BTS yang sedang menangani komunikasi pengguna, lalu ke BSC, kemudian sampai ke MSC. MSC kemudian meneruskan lagi SMS ke SMSC untuk disimpan. SMSC kemudian mengecek melalui HLR (*Home Location Register*) untuk mengetahui apakah *handphone* tujuan sedang aktif dan dimana lokasi *handphone* tujuan tersebut. Jika *handphone* sedang tidak aktif maka pesan tetap disimpan di SMSC itu sendiri, menunggu MSC memberitahukan bahwa *handphone* sudah aktif kembali untuk kemudian SMS dikirim dengan batas maksimum waktu tunggu yaitu *validity period* dari pesan SMS itu sendiri. Jika *handphone* tujuan aktif maka pesan disampaikan MSC lewat jaringan yang sedang menangani penerima (BSC dan BTS).

Pada AT *Command*, ada dua mode pengiriman SMS yaitu :

a) Mode Teks

Mode teks adalah metode pengiriman pesan yang termudah. Pada mode ini, pesan yang dikirim tidak dilakukan konversi. Teks yang dikirim tetap dalam bentuk aslinya.

b) Mode *Protocol Data Unit* (PDU)

Mode PDU adalah format pesan dalam heksadesimal oktet dan semi-oktet dengan panjang mencapai 160 (7-*bit*) atau 140 (8-*bit*) karakter. Untuk mengirimkan pesan SMS maka teks yang harus dikonversikan dahulu ke dalam bentuk PDU. Begitu juga untuk membaca pesan SMS yang diterima maka bentuk PDU harus dikonversi dahulu ke dalam bentuk teks.

2.4.2 *Modem*

Modem berfungsi untuk mengirimkan dan menerima SMS melalui jaringan GSM. Untuk mengetahui suatu *modem* atau *handphone* mendukung AT *Command* atau tidak, dapat diuji dengan menggunakan program *HyperTerminal* di komputer dengan mengetik karakter AT dan kemudian menekan *Enter*. Apabila mendapat respon "OK", *handphone* atau modem tersebut sudah mendukung AT *Command*. Modem yang digunakan pada sistem ini adalah *modem wavecom* M1306B RS232 dan *modem wavecom* M1306B USB. Bentuk fisik *modem wavecom* M1306B RS232 dan *modem wavecom* M1306B USB ditunjukkan dalam Gambar 2.8.

Gambar 2.8. Modem *Wavecom* M1306B RS232 dan Modem *Wavecom* M1306B USB
Sumber : Wavecom, 2004

Beberapa AT *command* atau AT *response* penting untuk mengoperasikan modem ini adalah (Wavecom, 2006) :

1) AT+CMGR, untuk membaca SMS
2) AT+CMGS, untuk mengirim SMS
3) AT+CMGD, untuk menghapus SMS
4) +CMTI, indikator ada SMS masuk

Sebelum digunakan oleh mikrokontroler, *setting* modem harus dikonfigurasi. Konfigurasi modem pada mikrokontroler ditunjukkan dalam Tabel 2.3.

Tabel 2.3. Konfigurasi Modem Pada Mikrokontroler

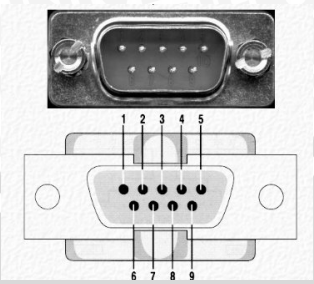
Parameter	Nilai	AT *Command*	Penjelasan
Baud rate	9600	AT+IPR=9600	Mengubah kecepatan link serial modem menjadi 9600 bps. Kecepatan link serial bawaan dari produsen adalah 115200 bps.
Flow control	None	AT+IFC=0,0	Tidak menggunakan *flow control* atau sinyal untuk tunggu.
Echo	Off	ATE0	Menonaktifkan *echo* atau karakter berulang.
Message format	Text	AT+CMGF=1	SMS menggunakan mode teks. Mode 0 adalah mode PDU sedangkan mode 1 adalah mode teks.

Setelah dilakukan setting modem, konfigurasi disimpan dengan menggunakan perintah AT&W.

2.5 Komunikasi *Serial*

Sistem transmisi sinyal RS232 menggunakan level tegangan dengan *respek to sistem common* (*power ground*). Tipe ini bagus untuk komunikasi data secara satu–satu (*point to point communications*). RS232 port pada PC hanya diperuntukkan untuk satu alat (*single device*). Misal, Com1 digunakan untuk *mouse port* sedangkan Com2 digunakan untuk *modem*. Syarat sinyal RS232 dapat berfungsi adalah dengan hubungan ke *ground* antara PC dengan alat (*common ground*). Jarak maksimal jalur komunikasi sangat terbatas hanya 100 / 200 kaki untuk komunikasi data secara asinkron dan hanya 50 kaki untuk komunikasi sinkron. Kecepatan transfer data RS232 cukup rendah, kecepatan maksimal hanya 19200 bits / detik. Singkatnya, RS232 hanya untuk komunikasi area lokal dan hanya untuk satu *driver* dan satu *receiver*. RS232 pada PC mempunyai dua jenis konektor, yaitu konektor dengan 25 Pin (DB25) dan konektor dengan 9 Pin

(DB9). Pada dasarnya hanya 3 pin yang terpakai, yaitu pin kirim, pin terima dan *ground*. Gambar 2.9 menunjukkan RS232 9 pin.



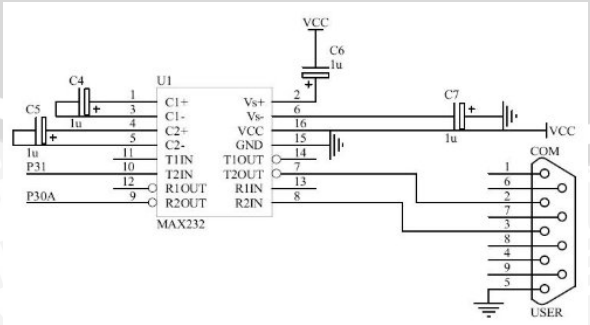
Gambar 2.9. RS232 9 Pin

Konfigurasi pin RS 232 9 pin ditunjukkan dalam Tabel 2.4.

Tabel 2.4. Konfigurasi Pin RS232 9 Pin

Pin	Signal	Pin	Signal
1	Data Carrier Detect	6	Data Set Ready
2	Received Data	7	Request to Send
3	Transmitted Data	8	Clear to Send
4	Data Terminal Ready	9	Ring Indicator
5	Signal Ground		

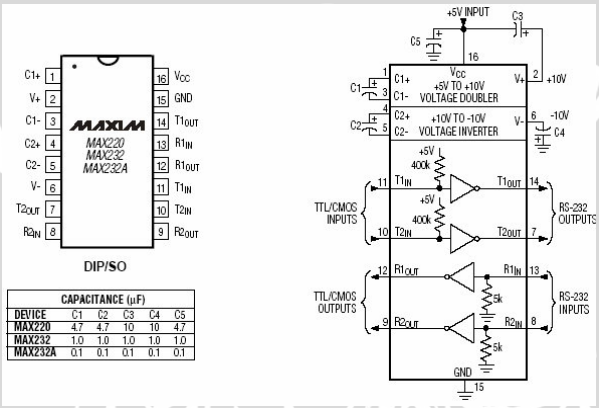
Dalam setiap proses *transfer data* serial, RS232 memerlukan sebuah *Data Terminal Equipment* (DTE) dan *Data Communication Equipment* (DCE) pada masing–masing terminal. Pengiriman data dilakukan secara bit per bit. Kecepatan transfer data harus sama antara pengirim dan penerima, jika tidak sama akan terjadi *overflow*. Kecepatan transmisi transfer data sering disebut dengan *baudrate*. Panjang data bit yang sering digunakan diantaranya adalah 4, 5, 6, 7, dan 8 bit. Gambar 2.10 menunjukkan *wiring diagram* RS232 ke mikrokontroler.



Gambar 2.10. *Wiring Diagram* RS232 ke Mikrokontroler

Pada komunikasi data *serial* pada dasarnya yang dikirimkan adalah tegangan dan kemudian dibaca dalam *bit*. Besar level tegangannya adalah antara -25 volt sampai dengan +25 volt. Untuk *bit* dengan logika 1 maka besar level tegangannya adalah antara -3 volt sampai -25 volt, sedangkan untuk bit dengan logika 0 maka besar level tegangannya antara +3 volt sampai +25 volt.

Pada umumnya komunikasi serial digunakan komponen IC RS232 yaitu pabrikan dari maxim MAX232. Gambar 2.11 menunjukkan konfigurasi pin IC MAX232.



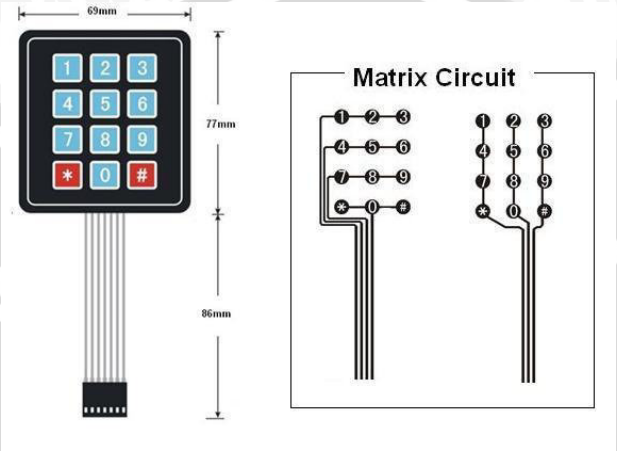
Gambar 2.11. Konfigurasi Pin IC MAX232

Ada beberapa komunikasi data *serial* dari mikrokontroler, mulai dari komunikasi *point to point* menggunakan RS232 dalam satu *frame proto*col. Komunikasi data dilakukan dengan pengiriman beberapa karakter ASCII dari PC ke mikrokontroler dan kembali ke PC. Aplikasi yang digunakan PC untuk menerima karakter ASCII adalah *hyperterminal*. Hal yang perlu diperhatikan dalam komunikasi adalah kesamaan *setting* parameter komunikasi, seperti *baudrate, comport, flowcontrol*, jumlah data bit, paritas, dan sebagainya. Untuk mendapat *setting baudrate* tersebut dilakukan dengan pengaturan register kontrol serial dan *mode timer* SFR pada mikrokontroler.

2.6 *Keypad*

Keypad memiliki bentuk sederhana menyerupai *keyboard* kecil yang difungsikan untuk memberikan masukan data melalui tombol-tombol yang terdapat dalam papan *keypad* tersebut. *Keypad* merupakan penyederhanaan dari tombol-tombol *push button* yang disusun secara matrik. *Keypad* dapat dibedakan

berdasarkan jumlah baris dan kolomnya, salah satunya adalah *keypad* matrik 3×4. *Keypad* matrik 3×4 terdiri atas 3 lajur kolom dan 4 lajur baris, dimana tiap baris dan kolom tersebut dihubungkan dengan tombol *push button* dengan jumlah keseluruhan sebanyak 12 buah. Bentuk dasar *keypad* 3×4 ditunjukkan dalam Gambar 2.12.

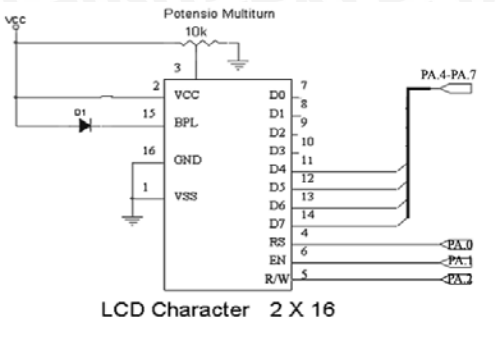


Gambar 2.12. *Keypad* 3×4

Proses pengambilan data pada *keypad*, menggunakan metode *scanning* pada lajur kolom dan lajur baris. Jika terdeteksi adanya persambungan antara baris dan kolom yang valid, maka mikrokontroler akan mengkodekan baris dan kolom mana yang aktif menjadi data *biner*. Apabila ada yang sesuai, maka mikrokontroler akan melakukan instruksi sesuai dengan data yang dimasukkan dari *keypad*.

2.7 *Liquid Crystal Display* (LCD)

Liquid Crystal Display (LCD) merupakan komponen elektronika yang digunakan untuk menampilkan karakter baik berupa karakter angka, huruf, atau karakter lainnya, sehingga tampilan tersebut dapat dilihat secara visual. Gambar 2.13 menunjukkan rangkaian *interface* ke LCD Karakter 2x16 dan Tabel 2.5 menunjukkan Pin-Pin I/O LCD.



Gambar 2.13. Rangkaian *Interface* ke LCD Karakter 2x16

Sumber: Andi Nalwan, 2004

Tabel 2.5. Pin-Pin I/O LCD

No	Simbol	Level	Fungsi
1	Vss		GND
2	Vcc		*Power supply* 5 Volt
3	Vee		LCD Drive
4	RS	H/L	H:*data input* L:*ins input*
5	R/W	H/L	H:*Read* L:*Write*
6	E	H	*Enable signal*
7	DB0	H/L	
8	DB1	H/L	
9	DB2	H/L	
10	DB3	H/L	*Data Bus*
11	DB4	H/L	
12	DB5	H/L	
13	DB6	H/L	
14	DB7	H/L	
15	V-BL		*Power supply* 4 – 4,2 Volt
16	V-BL		GND

Sumber: Andi Nalwan, 2004

Pada perancangan sistem ini memakai LCD modul M1632 yang merupakan sebuah modul LCD dot matrik yang membutuhkan daya kecil. LCD modul M1632 dilengkapi panel LCD dengan tingkat kontras yang cukup tinggi serta pengendali LCD CMOS yang telah terpasang dalam modul tersebut. LCD modul M1632 mempunyai spesifikasi sebagai berikut :

- Memiliki 16 karakter dan 2 baris tampilan yang terdiri atas 5 x 7 dot matrik ditambah dengan kursor.
- Memerlukan catu daya DC 5 V.
- Otomatis *reset* saat catu daya dinyalakan.
- Memiliki data RAM (max 80 karakter) dengan 80 x 8 *display*.
- Menggunakan 4 bit data dan 3 bit kontrol.

2.8 *Buzzer*

Buzzer adalah sebuah transduser yang berfungsi untuk mengubah energi listrik menjadi energi suara atau bunyi. Bunyi yang dihasilkan ini hanya satu nada atau hanya terdengar bunyi tit. Kebanyakan *buzzer* digunakan sebagai pertanda negatif terhadap sesuatu, yang biasanya banyak digunakan pada sensor keamanan, ataupun pada jam alarm. *Buzzer* terdapat banyak jenis, dari yang kecil hingga yang besar yang tentunya penggunaan tegangan dan arusnya juga lebih besar. Berikut ini adalah gambar dari *buzzer*, di mana *buzzer* hanya memiliki dua kaki yaitu kaki positif dan kaki negatif. Gambar 2.14. menunjukkan gambar *buzzer*.

Gambar 2.14. *Buzzer*

BAB III
METODOLOGI PENELITIAN

Penyusunan laporan 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 studi literatur, penentuan spesifikasi alat, perancangan dan pembuatan alat, pengujian alat, dan pengambilan kesimpulan.

3.1 Studi Literatur

Studi literatur dilakukan untuk mempelajari teori penunjang sistem yang dibutuhkan dalam perancangan dan pembuatan alat. Teori yang diperlukan antara lain berkaitan dengan komunikasi tim SAR, ATMega162, GPS *Receiver*, LCD, *keypad*, *buzzer*, komunikasi *serial*, *Microsoft Visual C# 2005* serta pengiriman data via SMS.

3.2 Perancangan dan Pembuatan Alat

3.2.1 Spesifikasi Alat

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

1) Jalur transmisi data yang digunakan adalah SMS dengan menggunakan *modem wavecom* M1306B RS232 dan *modem wavecom* M1306B USB.
2) Menampilkan secara visual posisi tim SAR - SRU dengan memanfaatkan Google *Map*.
3) Tim SAR pemantau bisa memilih dua pilihan pengiriman pesan yaitu data posisi atau data posisi sekaligus jumlah korban yang ditemukan oleh tim SAR – SRU. Jika tim SAR pemantau memilih data posisi maka alat pada tim SAR – SRU akan membunyikan *buzzer* selama dua kali sebagai tanda ada SMS dan secara otomatis akan mencari koordinat posisi menggunakan GPS dan mengirimkannya ke tim SAR pemantau. Untuk pilihan kedua yaitu data posisi sekaligus jumlah korban maka alat akan membunyikan

buzzer selama tiga kali sebagai tanda ada SMS dan tim SAR – SRU harus memasukkan jumlah korban yang ditemukannya melalui *keypad* dan mengirimkannya kembali ke tim SAR pemantau sekaligus koordinat posisinya.

4) Tim SAR – SRU juga dapat mengirimkan data posisi dan jumlah korban yang ditemukan ke tim SAR pemantau tanpa harus diminta dulu oleh tim SAR pemantau untuk mengirimkan datanya.
5) *Microsoft Visual C# 2005* digunakan sebagai *software* pengolah data pada komputer yang terintegrasi dengan Google *Map*.
6) Mikrokontroler yang digunakan adalah AVR ATMega162.
7) GPS *Receiver* yang digunakan adalah SkyNav SKM53 GPS *module*.
8) LCD digunakan untuk menampilkan koordinat posisi dan jumlah korban pada alat tim SAR – SRU.
9) Keypad yang digunakan adalah *Membrane Keypad* 3x4.
10) Menggunakan *Buzzer* DC 3 – 24 Volt.
11) Menggunakan enam *switch* untuk memilih menu.
12) Sistem tim SAR pemantau menggunakan catu daya AC 220 V, Sistem tim SAR- SRU menggunakan dua buah baterai pack DC 9,6 V.

3.2.2 Perancangan Alat

Perancangan alat ini melalui beberapa langkah sebagai berikut :

1) Pembuatan blok diagram sistem secara lengkap dengan mikrokontroler ATMega162 sebagai pengolah data utama.
2) Perancangan perangkat keras (*hardware*) masing-masing bagian, yaitu :
   a) Blok minimum sistem mikrokontroler AVR ATMega162.
   b) Blok rangkaian antarmuka GPS ke mikrokontroler AVR ATMega162.
   c) Blok rangkaian antarmuka Konverter RS232 dan Modem GSM (RS232) ke mikrokontroler AVR ATMega162.
   d) Blok rangkaian antarmuka LCD ke mikrokontroler AVR ATMega162.
   e) Blok rangkaian antarmuka *Keypad* ke mikrokontroler AVR ATMega162.
   f) Blok rangkaian antarmuka *Driver Buzzer* dan *Buzzer* ke mikrokontroler AVR ATMega162.

g) Blok rangkaian antarmuka *Switch* ke mikrokontroler AVR ATMega162.

h) Blok rangkaian antarmuka Modem GSM (USB) ke Komputer.

3) Perancangan perangkat lunak (*software*)

Perancangan perangkat lunak atau program untuk sistem mikrokontroler ATMega162 dan untuk sistem komputer dimulai dengan pembuatan diagram alir (*flowchart*) yang menjelaskan algoritma program yang dirancang sekaligus menggambarkan subrutin-subrutin yang dibutuhkan program.

4) Perancangan keseluruhan sistem

Dengan mengintegrasikan semua bagian rangkaian perangkat keras yang telah dirancang sebagai sebuah sistem terintegrasi yang kemudian akan dioperasikan dengan pengendalian perangkat lunak yang dirancang.

3.2.3 Pembuatan Alat

Pembuatan alat ini meliputi pembuatan perangkat keras (*hardware*) dan pembuatan perangkat lunak (*software*) sebagai berikut :

1) Pembuatan perangkat keras (*hardware*)

Pembuatan alat untuk perangkat keras (*hardware*) meliputi pembuatan tata letak komponen dan *layout* PCB, pelarutan PCB, pengeboran PCB, penyolderan komponen-komponen pada PCB dan pembuatan mekanik box alat.

2) Pembuatan perangkat lunak (*software*)

Dengan diagram alir (*flowchart*) yang telah dirancang, maka dibuatlah program sumber (*source program*) untuk memenuhi kebutuhan pengolahan data dan pengendalian alat dengan program bahasa C dan *Microsoft Visual C# 2005.*

3.3 Pengujian dan Analisis

Pengujian dilakukan dalam dua bentuk, yaitu : pengujian setiap blok rangkaian dan pengujian keseluruhan sistem.

1) Pengujian setiap blok diagram, meliputi :
   a) Pengujian GPS *Receiver*
   b) Pengujian *Keypad*
   c) Pengujian LCD
   d) Pengujian *Driver Buzzer* dan *Buzzer*
   e) Pengujian Konverter RS232 dan Modem GSM
   f) Pengujian Perangkat Lunak

2) Pengujian keseluruhan sistem

   Pengujian keseluruhan sistem dilakukan dengan menyambungkan blok perangkat keras dan mengoperasikan sistem kemudian dapat dianalisis apakah alat ini bekerja sesuai dengan spesifikasi yang diharapkan. Setelah perangkat keras telah beroperasi seperti yang diharapkan, perangkat lunak yang telah dibuat diujikan bersama perangkat kerasnya. Sistem dikatakan berhasil jika perangkat keras dan perangkat lunak yang ada telah tersinergi dan bekerja sebagai sebuah sistem yang sesuai dengan spesifikasi rancangan.

3.4 Pengambilan Kesimpulan dan Saran

Kesimpulan didapat berdasarkan hasil perealisasian sistem monitoring posisi operasi tim SAR – SRU (*Search and Rescue Unit*) pada daerah bencana dengan memanfaatkan GPS (*Global Positioning System*). Beberapa hal hasil pengujian disampaikan dalam kesimpulan disertai realita yang disusun secara berurutan.

BAB IV
PERANCANGAN DAN PEMBUATAN ALAT

Dalam bab ini akan dibahas perancangan dan pembuatan alat. Pembahasan akan dilakukan pada setiap blok rangkaian, cara kerja masing-masing blok rangkaian, perhitungan dan juga fungsi masing-masing blok rangkaian tersebut. Secara garis besar terdapat dua bagian perangkat yang ada yaitu :

- Perancangan perangkat keras (*Hardware*)
- Perancangan perangkat lunak (*Software*)

Pada perancangan perangkat keras akan meliputi *peripheral-peripheral* yang digunakan pada sistem ini. Sedangkan pada perancangan perangkat lunak akan meliputi diagram alir dan *software* secara umum. Akan tetapi kedua perangkat ini dalam kerjanya saling menunjang satu sama lain.

4.1 Spesifikasi Alat

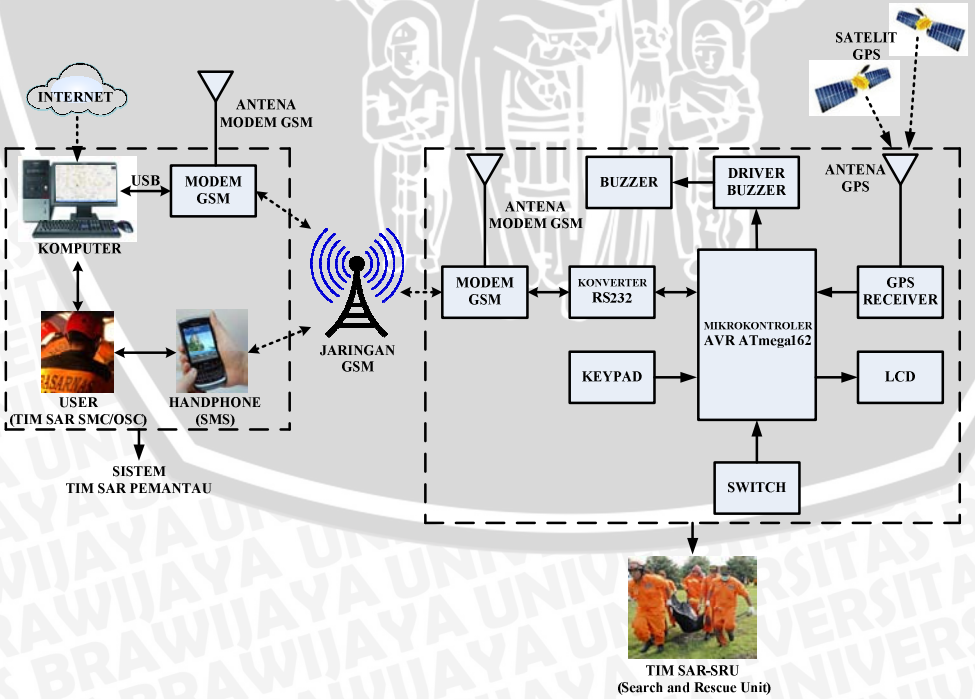
Spesifikasi alat secara global ditetapkan terlebih dahulu sebagai acuan dalam perancangan selanjutnya. Spesifikasi alat yang dirancang yaitu :

1) Jalur transmisi data yang digunakan adalah SMS dengan menggunakan *modem wavecom* M1306B RS232 dan *modem wavecom* M1306B USB.
2) Menampilkan secara visual posisi tim SAR - SRU dengan memanfaatkan Google *Map*.
3) Tim SAR pemantau bisa memilih dua pilihan pengiriman pesan yaitu data posisi atau data posisi sekaligus jumlah korban yang ditemukan oleh tim SAR – SRU. Jika tim SAR pemantau memilih data posisi maka alat pada tim SAR – SRU akan membunyikan *buzzer* selama dua kali sebagai tanda ada SMS dan secara otomatis akan mencari koordinat posisi menggunakan GPS dan mengirimkannya ke tim SAR pemantau. Untuk pilihan kedua yaitu data posisi sekaligus jumlah korban maka alat akan membunyikan *buzzer* selama tiga kali sebagai tanda ada SMS dan tim SAR – SRU harus memasukkan jumlah korban yang ditemukannya melalui *keypad* dan mengirimkannya kembali ke tim SAR pemantau sekaligus koordinat posisinya.

4) Tim SAR – SRU juga dapat mengirimkan data posisi dan jumlah korban yang ditemukan ke tim SAR pemantau tanpa harus diminta dulu oleh tim SAR pemantau untuk mengirimkan datanya.
5) *Microsoft Visual C# 2005* digunakan sebagai *software* pengolah data pada komputer yang terintegrasi dengan Google *Map*.
6) Mikrokontroler yang digunakan adalah AVR ATMega162.
7) GPS *Receiver* yang digunakan adalah SkyNav SKM53 GPS *module*.
8) LCD digunakan untuk menampilkan koordinat posisi dan jumlah korban pada alat tim SAR – SRU.
9) *Keypad* yang digunakan adalah *Membrane Keypad* 3x4.
10) Menggunakan *Buzzer* DC 3 – 24 Volt.
11) Menggunakan enam *switch* untuk memilih menu.
12) Sistem tim SAR pemantau menggunakan catu daya AC 220 V, Sistem tim SAR- SRU menggunakan dua buah baterai pack DC 9,6 V.

4.2 Diagram Blok Sistem

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



Gambar 4.1. Diagram Blok Sistem secara 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.

- Bagian Sistem Alat Tim SAR – SRU :
  1) GPS *Receiver* digunakan untuk memberikan informasi berupa posisi dengan bantuan satelit GPS.
  2) Mikrokontroler AVR ATMega162 sebagai pengolah data dan pusat kendali sistem dari alat tim SAR – SRU.
  3) *Keypad* digunakan untuk memasukkan data jumlah korban jika tim SAR-SRU menemukan korban.
  4) LCD digunakan untuk menampilkan koordinat posisi dan jumlah korban pada sistem alat tim SAR – SRU.
  5) *Driver buzzer* digunakan untuk mengendalikan sinyal dari mikrokontroler untuk menyalakan *buzzer.*
  6) *Buzzer* sebagai tanda adanya SMS masuk. Bunyi dua kali berarti diminta untuk mengirimkan data posisi saja. Bunyi tiga kali berarti diminta untuk mengirimkan data posisi dan jumlah korban yang ditemukan.
  7) *Konverter* RS232 digunakan untuk mengondisikan sinyal dari mikrokontroler ke modem GSM *wavecom*.
  8) Modem GSM *wavecom* pada tim SAR – SRU digunakan untuk mengirim dan menerima data dari dan ke tim SAR pemantau via SMS.
  9) *Switch* digunakan untuk memilih menu.

- Bagian Sistem Alat Tim SAR Pemantau :
  1) Modem GSM *wavecom* pada tim SAR pemantau digunakan untuk mengirim dan menerima data dari dan ke tim SAR – SRU via SMS.
  2) Komputer digunakan untuk menampilkan data secara visual dengan menggunakan *software Microsoft Visual C# 2005* yang terintegrasi dengan Google *Map*.
  3) *Handphone* digunakan untuk menerima SMS jika tim SAR pemantau hanya menginginkan data posisi saja tanpa tampilan visual berupa peta.
  4) User adalah tim SAR pemantau (SMC/OSC).

4.3 Prinsip Kerja Alat

Alat Monitoring Posisi Operasi Tim SAR - SRU (*Search and Rescue Unit*) Pada Daerah Bencana Dengan Memanfaatkan GPS (*Global Positioning System*) ini terdapat dua sistem alat, yaitu Sistem Tim SAR Pemantau dan Sistem Tim SAR – SRU. Prinsip kerja masing-masing sistem adalah sebagai berikut.

1) Sistem Tim SAR Pemantau

Pada sistem tim SAR pemantau terdapat dua alat untuk monitoring posisi yaitu dengan menggunakan *handphone* dan komputer. Akan tetapi untuk melihat posisi tim SAR - SRU secara detail dapat menggunakan komputer dengan tampilan peta Google *Map*, sedangkan pada *handphone* hanya menerima koordinat posisi saja. Monitoring dengan menggunakan komputer terhubung dengan modem GSM yang berfungsi untuk mengirim dan menerima data dari sistem tim SAR - SRU. Komputer dioperasikan oleh operator (*user*) dengan sebuah program *interface* yang bisa mengirim perintah-perintah ke sistem tim SAR - SRU melalui modem GSM. Selain komputer, *handphone* juga bisa digunakan untuk mengirim perintah-perintah ke sistem tim SAR - SRU.

Perintah-perintah yang dapat dikirimkan ke sistem tim SAR - SRU ada dua macam perintah, yaitu :

a) Perintah Permintaan Posisi.

Sistem tim SAR pemantau dapat mengirimkan perintah berupa SMS dengan format pesan (GPS) ke sistem tim SAR - SRU. setelah itu akan mendapatkan balasan sms data posisi tim SAR - SRU berupa *link* yang dapat dilihat di peta Google *Map*.

b) Perintah Permintaan Posisi dan Jumlah Korban yang Ditemukan.

Sistem tim SAR pemantau dapat mengirimkan perintah berupa SMS dengan format pesan (**KORBAN**) ke sistem tim SAR - SRU. setelah itu akan mendapatkan dua balasan SMS yaitu data jumlah korban yang ditemukan dan posisi tim SAR - SRU berupa *link* yang dapat dilihat di peta Google *Map*.

Format penulisan SMS harus benar, penulisan menggunakan huruf kecil atau besar dibedakan. Penulisan SMS yang tidak benar menyebabkan perintah tidak dapat dikenali oleh sistem tim SAR - SRU.

2) Sistem Tim SAR – SRU.

Sistem tim SAR - SRU adalah sebuah alat yang dibawa oleh tim SAR - SRU pada saat pencarian korban atau evakuasi di daerah bencana. Sistem tim SAR – SRU mempunyai enam menu untuk fungsi-fungsi tertentu. Pemilihan menu dengan cara memilih saklar pada alat sistem tim SAR – SRU. Penjelasan menu-menu tersebut, yaitu :

a) Menu *Disable* Modem

Menu *disable* modem ini digunakan untuk men-*disable* modem GSM yaitu untuk menonaktifkan modem GSM sebelum power modem GSM dimatikan. Hal ini bertujuan supaya modem GSM tidak cepat rusak.

b) Menu Terima SMS

Menu terima SMS digunakan untuk menerima perintah dari sistem tim SAR pemantau berupa SMS. Pada menu ini alat tim SAR – SRU siap menerima SMS sesuai format yang telah ditentukan. SMS dengan isi (**GPS**) permintaan posisi maka mikrokontroler akan membunyikan *buzzer* selama dua kali sebagai tanda adanya SMS permintaan posisi dan dilanjutkan dengan pengambilan data posisi dari GPS *receiver*. Setelah data posisi sudah valid maka data tersebut akan dikirim secara otomatis kepada tim SAR pemantau melalui modem *wavecom* via SMS. Jika isi SMS adalah (**KORBAN**) permintaan data posisi dan jumlah korban yang ditemukan maka mikrokontroler akan membunyikan *buzzer* selama tiga kali dan tim SAR – SRU harus memasukkan data jumlah korban yang ditemukannya melalui *keypad*. Dilanjutkan dengan pengambilan data posisi dari GPS *receiver* yang akan dikirimkan ke tim SAR pemantau bersamaan data jumlah korban yang ditemukan. Jika ada SMS masuk dengan format SMS yang salah maka mikrokontroler akan membunyikan *buzzer* selama satu kali dan mengirim SMS ke *handphone* sistem tim SAR pemantau dengan isi pesan (**Format SMS Salah**).

c) Menu *Input Keypad*

Menu *input keypad* berfungsi untuk memasukkan jumlah korban yang ditemukan oleh tim SAR – SRU. Menu ini dapat digunakan ketika ada perintah dari sistem tim SAR pemantau atau dapat juga digunakan sewaktu-waktu tanpa ada perintah dulu dari sistem tim SAR pemantau. Setiap input data *keypad* akan disertai dengan pencarian data posisi, sehingga data yang dikirim adalah data jumlah korban yang ditemukan dan data posisi korban ditemukan.

d) Menu *Checkpoint*

Menu *checkpoint* digunakan untuk melihat koordinat posisi titik awal (posisi tim SAR pemantau). Jika posisi titik awal belum ditentukan, menu ini dapat digunakan bersama dengan menu simpan checkpoint untuk penyimpanan data posisi titik awal tersebut.

e) Menu Simpan *Checkpoint*

Menu simpan *checkpoint* digunakan untuk mencari data posisi titik awal (posisi tim SAR pemantau) dan menyimpan ke dalam EEPROM mikrokontroler. Sebelum berangkat ke daerah bencana (tempat evakuasi), tim SAR – SRU mencari data posisi titik awal ini yang berfungsi sebagai titik acuan untuk perhitungan sudut dan jarak antara tempat evakuasi dengan titik awal (posisi tim SAR pemantau).

f) Menu Sudut Jarak

Menu sudut jarak digunakan untuk menghitung sudut dan jarak antara tempat evakuasi dengan titik awal (posisi tim SAR pemantau). Menu ini dapat digunakan seandainya ada tim SAR yang kesasar pada saat di tempat evakuasi, sehingga dapat terbantu dengan memperkirakan sudut dan jarak ini untuk bisa kembali ke posisi titik awal (posisi tim SAR pemantau) dengan selamat.

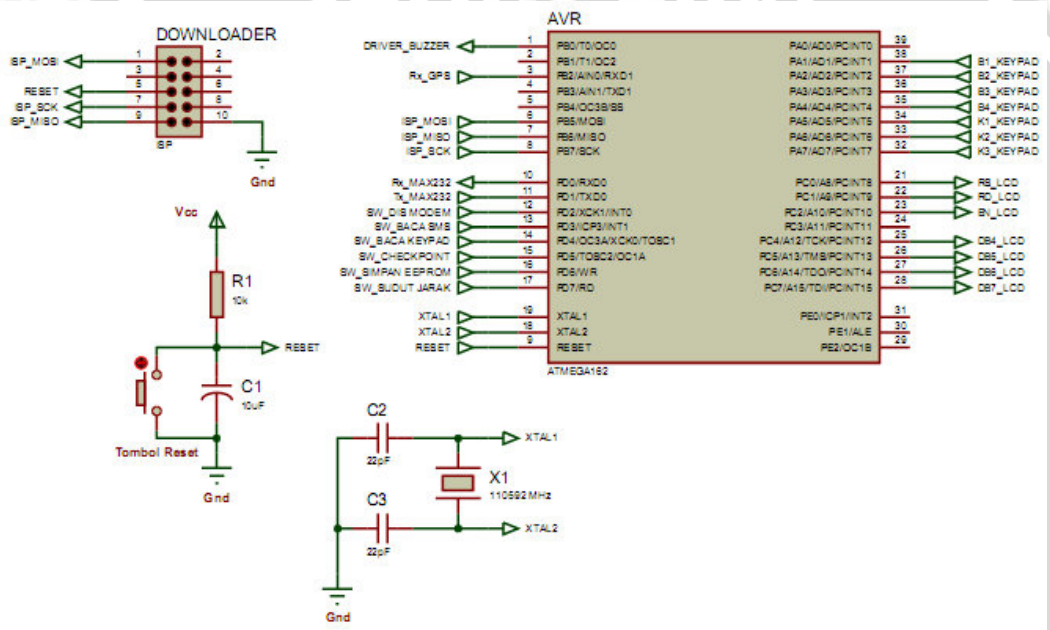
4.4 Perancangan Perangkat Keras (*Hardware*)

4.4.1 Perancangan Rangkaian Sistem Mikrokontroler AVR ATMega162

Mikrokontroler yang digunakan pada rangkaian ini adalah ATMega162 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.

Pada mikrokontroler ATMega162 nilai kristal yang diizinkan berkisar antara 0,4 MHz sampai 16 MHz. Dalam perancangan digunakan kristal sebesar 11.0592 MHz. Sedangkan besarnya nilai C1 dan C2 disesuaikan dengan yang tertera dalam *datasheet* antara 12 pF sampai 22 pF. Dalam perancangan digunakan kapasitor sebesar 22 pF. Rangkaian sistem mikrokontroler ATMega162 ditunjukkan dalam Gambar 4.2.



Gambar 4.2. Rangkaian Sistem Mikrokontroler ATMega162

Dalam rangkaian mikrokontroler diperlukan rangkaian reset. Rangkaian ini berfungsi untuk mengatur waktu me-*restart* ulang program bila terjadi *error* saat tombol *reset* ditekan. Berdasarkan *datasheet*, pin *RESET* harus diberi logika rendah minimal selama 2,5 µs untuk me-*reset* mikrokontroler. Waktu minimal ini dijadikan pedoman untuk menentukan nilai R dan C. Bila nilai C telah ditetapkan sebesar 10µF dan nilai R ditetapkan sebesar 10KΩ, maka perhitungan waktu *reset* adalah sebagai berikut.

t = 0,616 x R x C

t = 0,616 x 10KΩ x 10µF

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

$t = 61{,}6 \times 10^{-3}\ s$

$t = 61{,}6\ ms$

Berdasarkan perhitungan dapat diketahui waktu *reset* yang terbentuk adalah selama 61,6 ms. Nilai tersebut lebih besar dari batas minimal waktu reset sebesar 2,5µs, sehingga nilai C sebesar 10µF dan R sebesar 10KΩ dapat diaplikasikan dalam rangkaian reset mikrokontroler ATMega162.

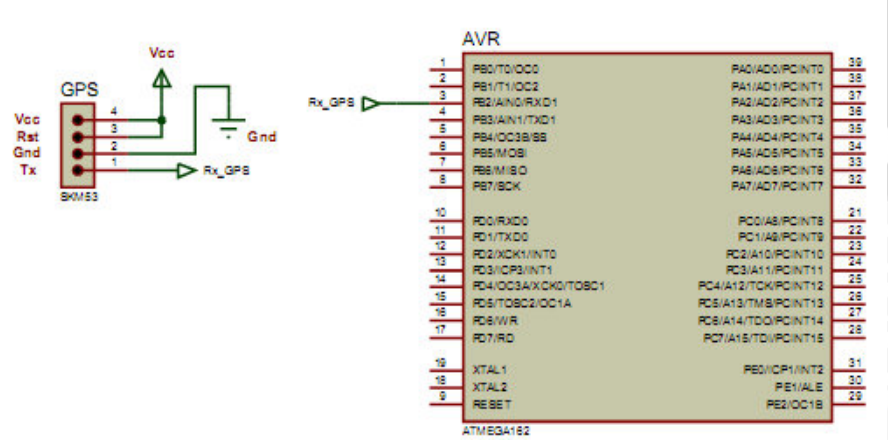
Soket downloader digunakan untuk mengisi program ke dalam *chip* mikrokontroler ATMega162.

4.4.2 Perancangan Rangkaian Antarmuka GPS

GPS *receiver* yang digunakan pada sistem adalah SkyNav SKM53 GPS *module*. Sesuai dengan *datasheet*, konfigurasi untuk komunikasi serial diatur dengan nilai baudrate 9600bps, no parity, delapan bit data dan satu stop bit.

Output GPS *receiver* ini berupa data NMEA 0183. Seperti yang telah dijelaskan dalam Bab 2, kalimat NMEA 0183 output dari GPS *receiver* terdiri dari beberapa bentuk. Pada perancangan sistem ini, kalimat NMEA 0183 yang digunakan adalah tipe GPRMC (*Recommended minimum specific* GNSS *data*).

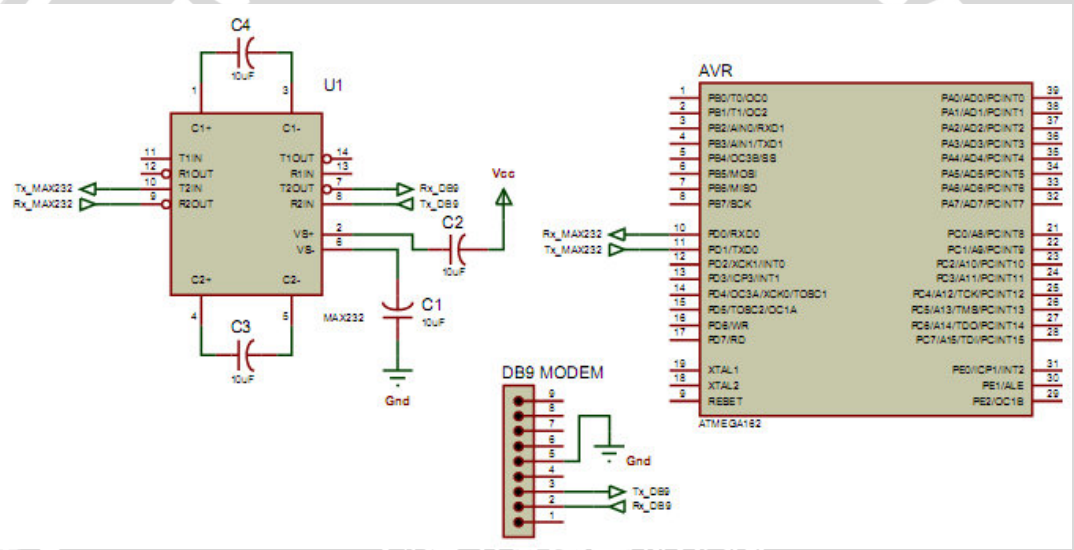
SkyNav SKM53 GPS *module* mempunyai beberapa pin, dalam perancangan ini yang digunakan hanya empat pin, yaitu pin Vcc dan pin *reset* terhubung Vcc, pin Ground terhubung Ground dan pin Tx terhubung RXD1 mikrokontroler yang berfungsi sebagai jalur data GPS. Rangkaian antarmuka GPS dengan mikrokontroler ATMega162 ditunjukkan dalam Gambar 4.3.



Gambar 4.3. Rangkaian Antarmuka GPS dengan Mikrokontroler ATMega162

4.4.3 Perancangan Rangkaian Antarmuka Konverter RS232 dan Modem GSM (RS232)

Level tegangan Modem GSM (RS232) berbeda dengan level tegangan mikrokontroler ATMega162. Untuk itu, dibutuhkan konverter RS232 sebagai pengkonversi data. 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. Kapasitor yang dipasang adalah sebesar 10µF. Rangkaian antarmuka Konverter RS232 dan Modem dengan mikrokontroler ATMega162 ditunjukkan dalam Gambar 4.4.

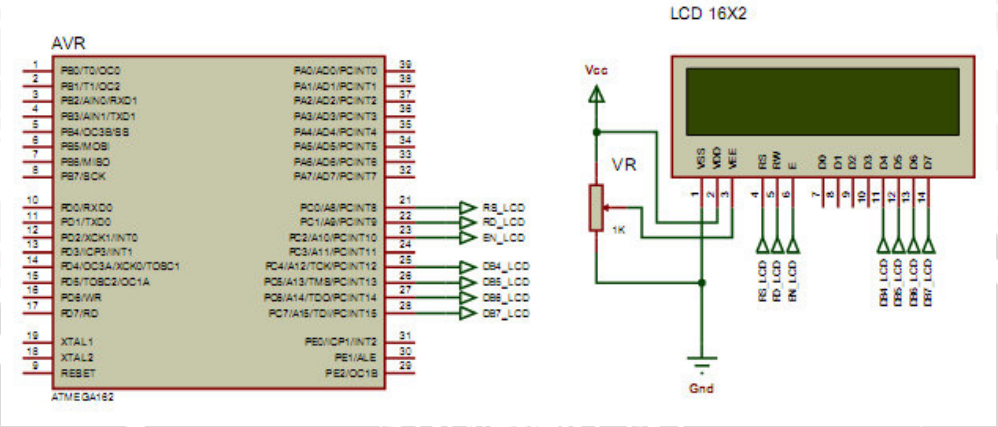


Gambar 4.4. Rangkaian Antarmuka Konverter RS232 dan Modem dengan Mikrokontroler ATMega162

Modem GSM (RS232) menggunakan *connector female* DB9, sehingga dalam perancangan ini IC MAX232 dihubungkan dengan *connector male* DB9. Pin yang dipakai hanya tiga, yaitu pin 2 (Rx MAX232), pin 3(Tx MAX232) dan pin 5 (Ground).

4.4.4 Perancangan Rangkaian Antarmuka LCD

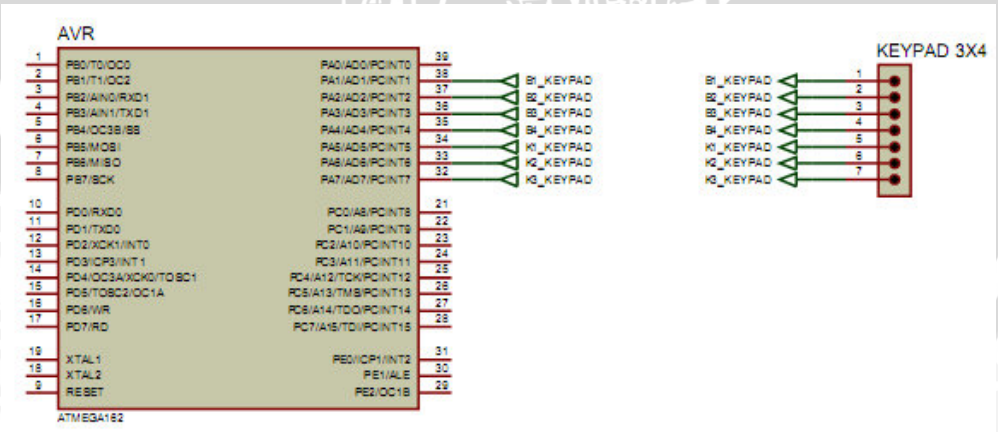
Pada perancangan rangkaian LCD ini dioperasikan sebagai mode 4 bit, maka diperlukan 7 buah *line* (3 *line control* dan 4 *data bus*). Rangkaian antarmuka LCD dengan mikrokontroler ATMega162 ditunjukkan dalam Gambar 4.5.



Gambar 4.5. Rangkaian Antarmuka LCD dengan Mikrokontroler ATMega162

4.4.5 Perancangan Rangkaian Antarmuka *Keypad*

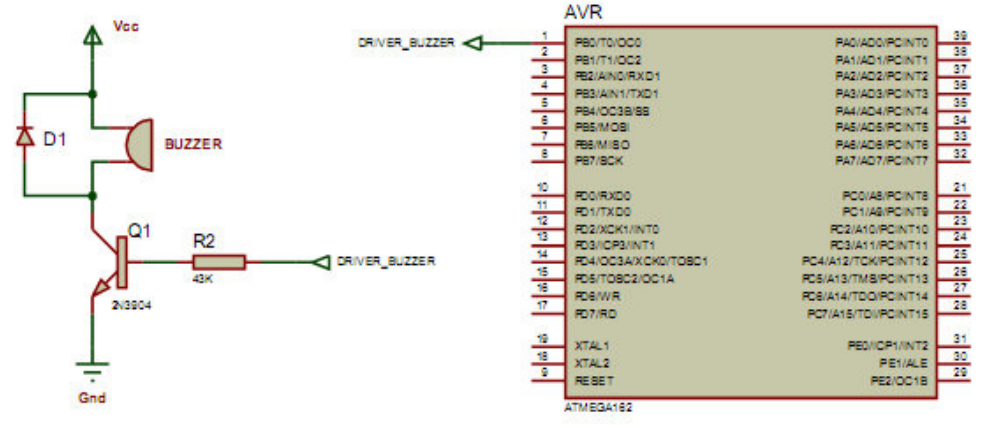
Dalam perancangan ini menggunakan *membrane keypad* 3x4 yang terhubung dengan portA mikrokontroler ATMega162. Rangkaian antarmuka *Keypad* dengan mikrokontroler ATMega162 ditunjukkan dalam Gambar 4.6.



Gambar 4.6. Rangkaian Antarmuka *Keypad* dengan Mikrokontroler ATMega162

4.4.6 Perancangan Rangkaian Antarmuka *Driver Buzzer* dan *Buzzer*

Rangkaian *driver buzzer* yang digunakan untuk menyalakan *buzzer* terdiri dari transistor NPN 2N3904 yang difungsikan sebagai transistor *switching*, seperti yang ditunjukkan dalam Gambar 4.7.



Gambar 4.7. Rangkaian Antarmuka *Driver Buzzer* dan *Buzzer* dengan Mikrokontroler ATMega162

Hfe dari *datasheet* transistor 2N3904 besarnya 100, dan arus maksimal dari *datasheet buzzer* adalah 12 mA, maka :

Ic = Ib x Hfe

Ib = Ic / Hfe

Ib = 12 / 100

= 0,12 mA

maka R2 adalah :

R2 = (Vee – Vbe) / Ib

= (5 – 0,7) / 0,12

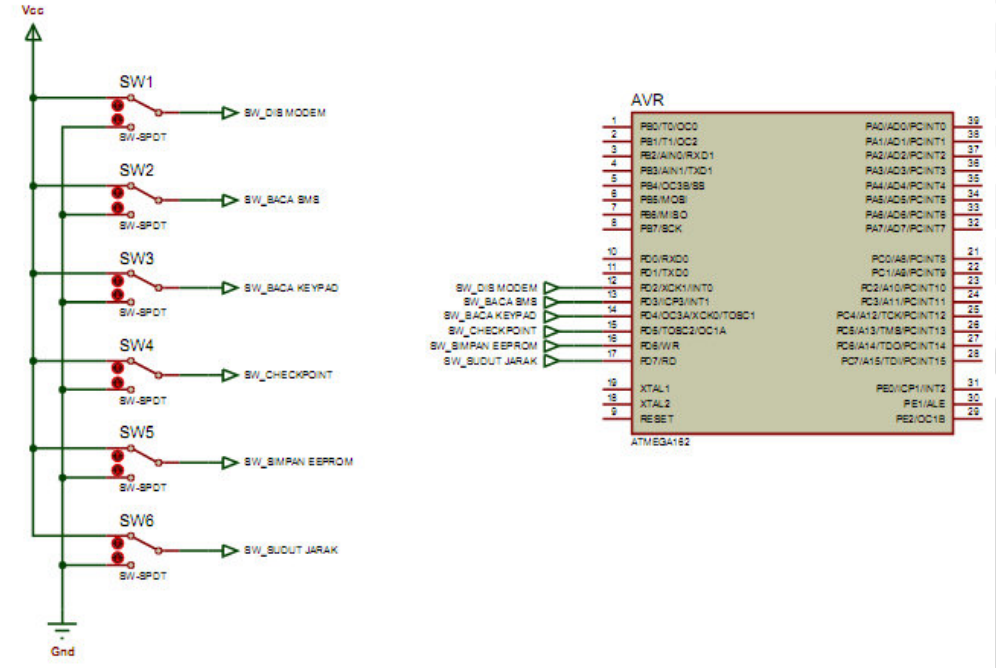
= 35,83 KΩ

Jadi dari hasil perhitungan diatas diperoleh nilai R2 sebesar 35,83 KΩ maka disesuaikan dengan resistor yang ada dipasaran yaitu 43 KΩ.

4.4.7 Perancangan Rangkaian Antarmuka *Switch*

Dalam perancangan ini, menggunakan enam *switch* yang berfungsi untuk memilih menu. *Switch* terhubung dengan Vcc dan Ground untuk memberikan

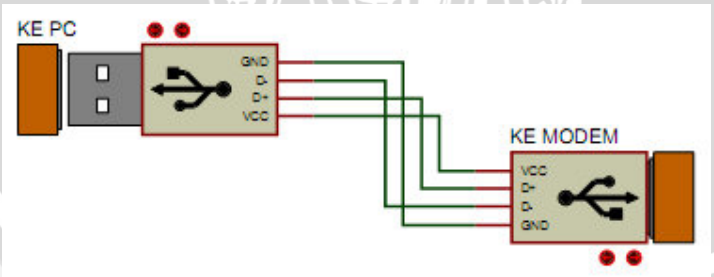
logika masukan pada mikrokontroler ATMega162. Rangkaian antarmuka *Switch* dengan Mikrokontroler ATMega162 ditunjukkan dalam Gambar 4.8.



Gambar 4.8. Rangkaian Antarmuka *Switch* dengan Mikrokontroler ATMega162

4.4.8 Perancangan Rangkaian Antarmuka Modem GSM (USB)

Modem GSM (USB) yang digunakan pada komputer (Sistem tim SAR Pemantau) menggunakan *connector* USB, sehingga modem GSM dapat langsung dihubungkan dengan komputer tanpa sebuah konverter. Rangkaian antarmuka Modem GSM (USB) dengan Komputer ditunjukkan dalam Gambar 4.9.

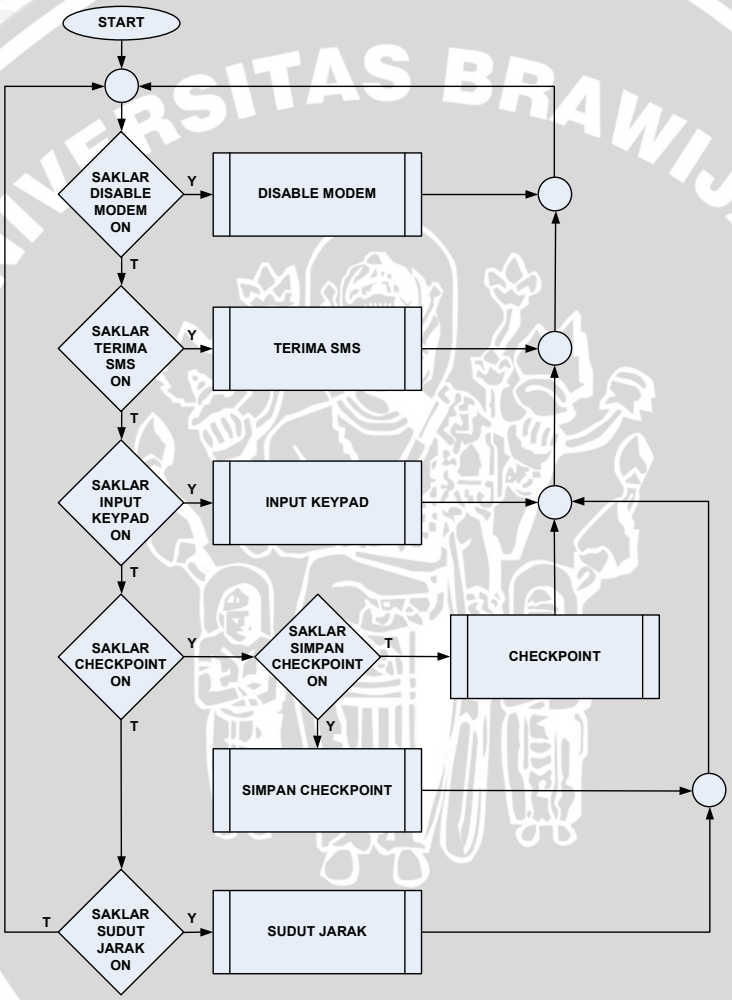


Gambar 4.9. Rangkaian Antarmuka Modem GSM (USB) dengan Komputer

4.5 Perancangan Perangkat Lunak (*Software*)

4.5.1 Perancangan Perangkat Lunak Mikrokontroler

Perancangan perangkat lunak pada mikrokontroler menggunakan bahasa C dengan compiler program CodeVisionAVR. Untuk memberikan gambaran umum jalannya program dan memudahkan dalam pembuatan perangkat lunak, maka dibuat *flowchart* yang memudahkan jalannya program. *Flowchart* program utama mikrokontroler ditunjukkan dalam Gambar 4.10.



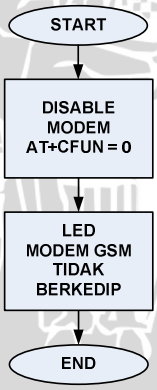
Gambar 4.10. *Flowchart* Program Utama Mikrokontroler

Mikrokontroler merupakan komponen utama yang digunakan untuk mengatur seluruh kerja sistem. Ketika mikrokontroler aktif, mikrokontroler akan mendeteksi input dari enam menu yang ada pada alat yaitu : saklar *disable* modem, saklar terima sms, saklar *input keypad*, saklar *checkpoint*, saklar simpan

checkpoint dan saklar sudut jarak. Ketika saklar *disable* modem aktif maka akan menjalankan sub rutin *disable* modem. Ketika saklar terima sms aktif maka akan menjalankan sub rutin terima sms. Ketika saklar *input keypad* aktif maka akan menjalankan sub rutin *input keypad*. Ketika saklar *checkpoint* aktif maka akan menjalankan sub rutin *checkpoint*., Ketika saklar *checkpoint* dan saklar simpan *checkpoint* aktif maka akan menjalankan sub rutin simpan *checkpoint*. Ketika saklar sudut jarak aktif maka akan menjalankan sub rutin sudut jarak. Program berulang terus sampai saklar *power* dimatikan.

4.5.1.1 Perancangan Sub Rutin *Disable* Modem

Saat saklar *disable* modem aktif maka mikrokontroler akan mengeksekusi sub rutin *disable* modem. Mikrokontroler mengirim AT *command* berupa AT+CFUN=0 ke modem GSM yang berfungsi untuk men-*disable* modem. Hal ini dilakukan ketika akan mematikan *power* modem GSM supaya modem GSM tidak cepat rusak. Modem bisa dimatikan ketika Led indikator modem GSM menyala tidak berkedip. *Flowchart* sub rutin *disable* modem ditunjukkan dalam Gambar 4.11.



Gambar 4.11. *Flowchart* Sub Rutin *Disable* Modem

4.5.1.2 Perancangan Sub Rutin Terima SMS

Sistem tim SAR-SRU ini dapat bekerja berdasarkan perintah dari sistem tim SAR pemantau, dimana perintah tersebut dikirimkan dalam format SMS. SMS yang diterima akan diproses oleh mikrokontroler untuk diseleksi apakah SMS yang diterima formatnya sesuai atau tidak. SMS yang tidak sesuai format akan

diabaikan. Jika SMS yang diterima formatnya sesuai, maka SMS yang diterima akan diproses oleh mikrokontroler untuk diseleksi kembali apa yang diperintahkan oleh sistem tim SAR pemantau. Perintah yang dapat diproses oleh sistem tim SAR - SRU ada dua macam yaitu :

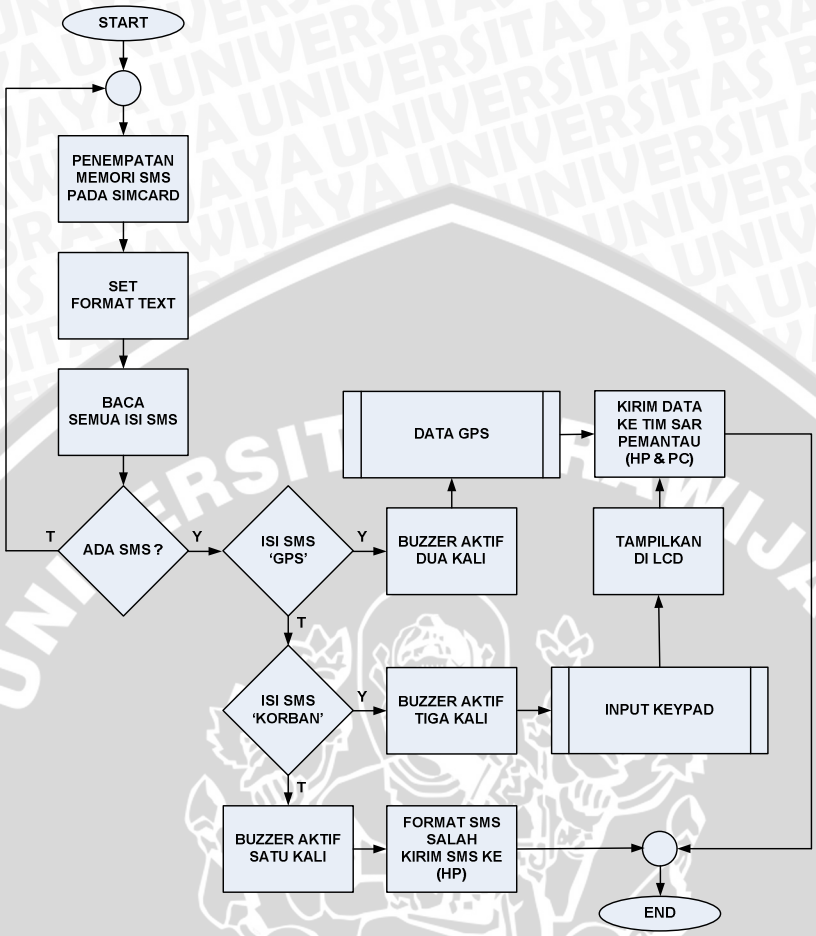
a) Perintah untuk mengirimkan data posisi.

   SMS dengan isi (**GPS**) permintaan posisi maka mikrokontroler akan membunyikan *buzzer* selama dua kali sebagai tanda adanya SMS permintaan posisi dan dilanjutkan dengan pengambilan data posisi pada sub rutin data GPS. Setelah data posisi sudah valid maka data tersebut akan dikirim secara otomatis kepada tim SAR pemantau (*Handphone* dan Komputer) melalui modem *wavecom* via SMS.

b) Perintah untuk mengirimkan jumlah korban dan data posisi.

   Jika isi SMS adalah (**KORBAN**) permintaan data posisi dan jumlah korban yang ditemukan maka mikrokontroler akan membunyikan *buzzer* selama tiga kali dan tim SAR – SRU harus memasukkan data jumlah korban yang ditemukannya melalui *keypad* dilanjutkan dengan pengambilan data posisi dari GPS *receiver* (sub rutin *input keypad*). Data posisi GPS akan dikirimkan ke tim SAR pemantau bersamaan data jumlah korban yang ditemukan.

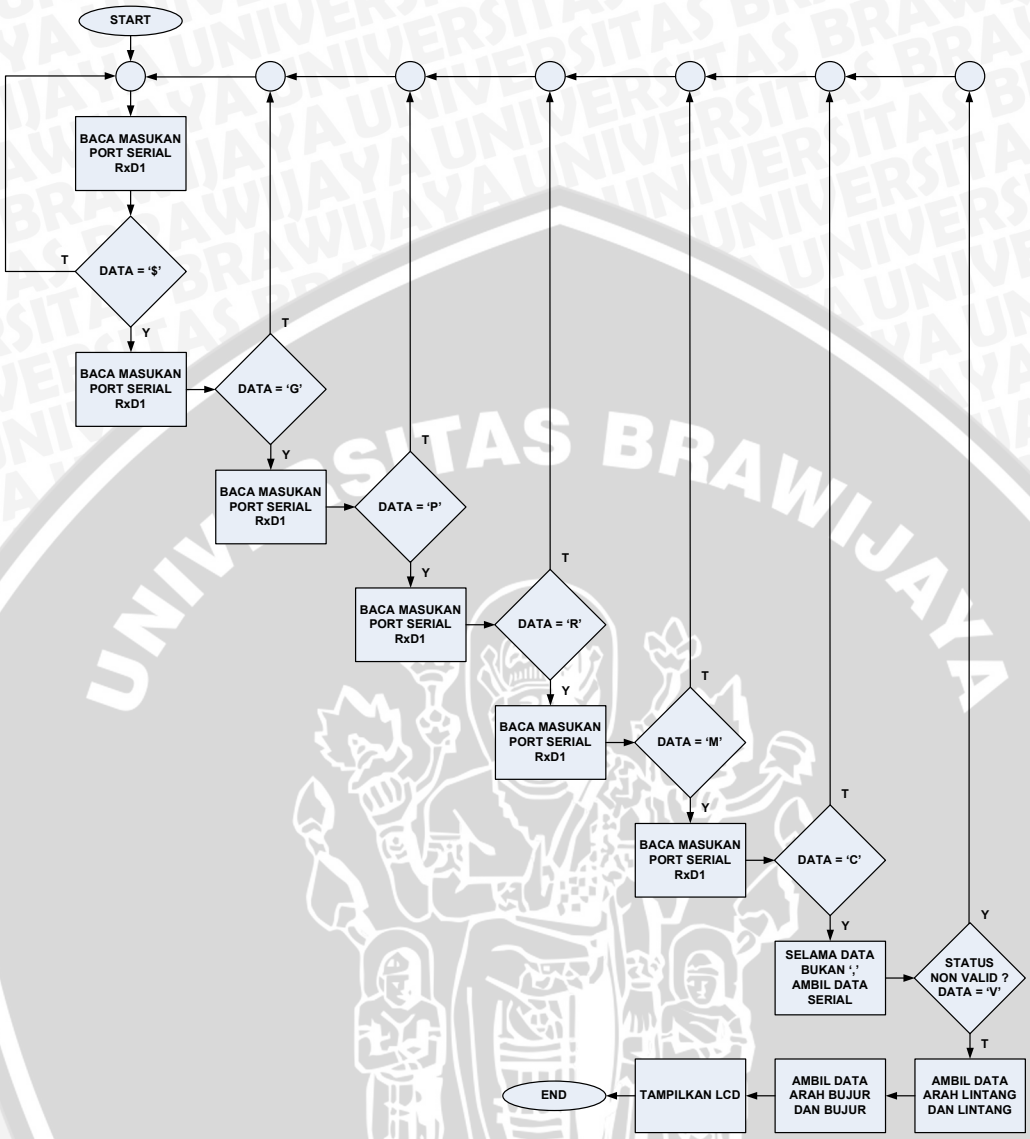
Jika ada SMS masuk dengan format SMS yang salah maka mikrokontroler akan membunyikan *buzzer* selama satu kali dan mengirim SMS ke *handphone* sistem tim SAR pemantau dengan isi pesan (**Format SMS Salah**). *Flowchart* sub rutin terima SMS ditunjukkan dalam Gambar 4.12.



Gambar 4.12. *Flowchart* Sub Rutin Terima SMS

4.5.1.3 Perancangan Sub Rutin Data GPS

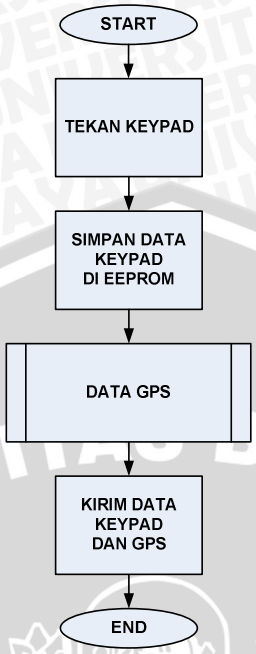
Kalimat NMEA 0183 *output* dari GPS *receiver* terdiri dari berbagai macam bentuk. Kalimat NMEA 0183 yang digunakan dalam sistem ini adalah tipe GPRMC. Pada saat sub rutin data GPS dipanggil maka mikrokontroler akan membaca masukan port serial RxD1 selanjutnya menyeleksi kalimat NMEA 0183 dan hanya memproses kalimat NMEA 0183 yang bertipe GPRMC. Mikrokontroler akan melakukan *parsing* data GPS dan membaca terus masukan port serial sampai data posisi yang diberikan oleh GPS *receiver* valid. Jika data GPS sudah valid maka data status, data lintang dan data bujur akan ditampilkan di LCD. *Flowchart* sub rutin data GPS ditunjukkan dalam Gambar 4.13.



Gambar 4.13. *Flowchart* Sub Rutin Data GPS

4.5.1.4 Perancangan Sub Rutin *Input Keypad*

Saat saklar input keypad aktif maka mikrokontroler akan mengeksekusi sub rutin *input keypad*. Mikrokontroler menunggu *keypad* ditekan oleh tim SAR – SRU. Setelah *keypad* ditekan, data *keypad* akan disimpan di EEPROM dan memanggil subrutin data GPS. Data *keypad* dan data GPS akan dikirim ke sistem tim SAR pemantau. *Flowchart* sub rutin *input keypad* ditunjukkan dalam Gambar 4.14.

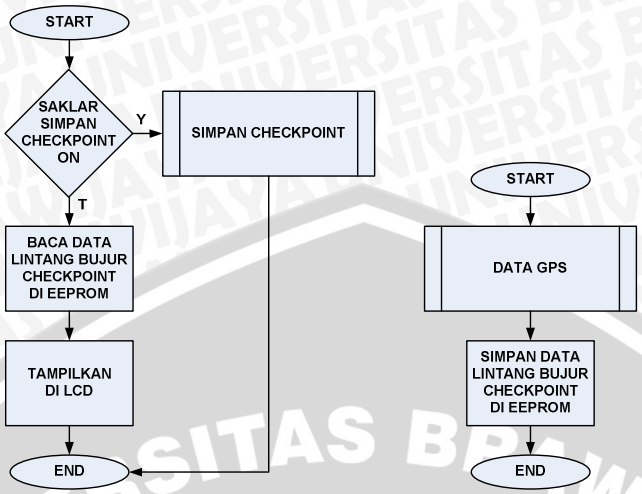


Gambar 4.14. *Flowchart* Sub Rutin *Input Keypad*

4.5.1.5 Perancangan Sub Rutin *Checkpoint* dan Simpan *Checkpoint*

Saat saklar *checkpoint* aktif maka mikrokontroler akan mengeksekusi sub rutin *checkpoint*. Di dalam sub rutin *checkpoint* sendiri terdapat sub rutin simpan *checkpoint*. Tujuan pembuatan subrutin *checkpoint* ini untuk melihat koordinat posisi titik awal (posisi tim SAR pemantau). Jika posisi titik awal belum ditentukan, maka dapat mengaktifkan saklar simpan *checkpoint* dan mikrokontroler akan mengerjakan subrutin simpan *checkpoint* untuk penyimpanan data posisi titik awal berupa data GPS di EEPROM. Ketika data posisi titik awal sudah tersimpan di EEPROM maka untuk melihat data *checkpoint* hanya mengaktifkan saklar checkpoint saja. Data posisi berupa lintang dan bujur akan ditampilkan di LCD.

Sebelum berangkat ke daerah bencana (tempat evakuasi), tim SAR – SRU mencari data posisi titik awal ini yang berfungsi sebagai titik acuan untuk perhitungan sudut dan jarak antara tempat evakuasi dengan titik awal (posisi tim SAR pemantau). *Flowchart* sub rutin *checkpoint* dan sub rutin simpan *checkpoint* ditunjukkan dalam Gambar 4.15. (a) dan Gambar 4.15 (b).



(a) (b)

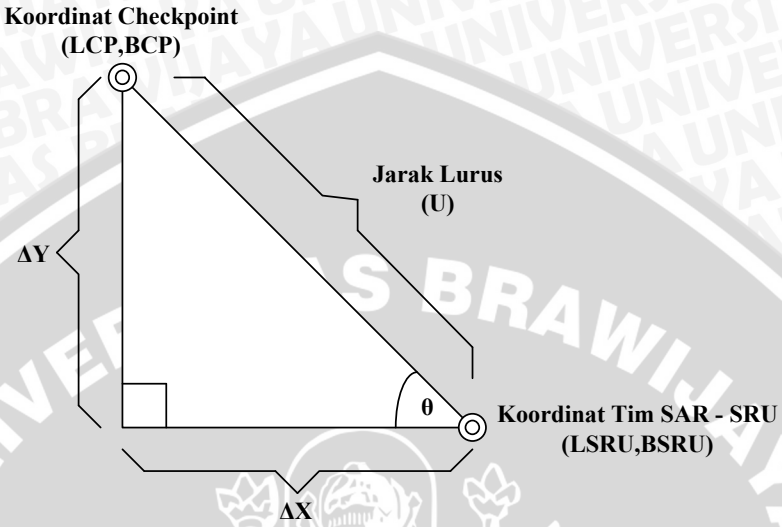
Gambar 4.15. (a) *Flowchart* Sub Rutin *Checkpoint*
(b) *Flowchart* Sub Rutin Simpan *Checkpoint*

4.5.1.6 Perancangan Sub Rutin Sudut Jarak

Saat saklar sudut jarak aktif maka mikrokontroler akan mengeksekusi sub rutin sudut jarak. Di dalam sub rutin sudut jarak sendiri terdapat sub rutin perhitungan jarak, sub rutin perhitungan sudut (1), sub rutin perhitungan sudut (2), sub rutin perhitungan sudut (3) dan sub rutin perhitungan sudut (4). Tujuan sub rutin sudut jarak ini untuk menghitung sudut dan jarak antara tempat tim SAR – SRU terhadap tim SAR pemantau dengan cara memproses dua koordinat yang diketahui, yaitu koordinat *checkpoint* (koordinat tim SAR pemantau) dan koordinat tim SAR – SRU yang berada di daerah bencana. Maka dari itu, tim SAR – SRU harus mencari titik koordinat posisinya dulu sebelum mikrokontroler menghitung sudut dan jaraknya. Setalah data GPS valid, mikrokontroler akan menyimpan data GPS tersebut ke EEPROM. Data GPS dari GPS *receiver* berupa *string*, sebelum diproses kedalam perhitungan maka harus dikonversi kedalam bentuk desimal yaitu dengan cara dikurangi dengan nilai 0x30. Data GPS yang diterima juga masih berupa derajat menit (**ddmm.mmmm**), untuk mempermudah perhitungan maka dikonversi dulu ke bentuk derajat saja (**dd.dddd**). Setelah kedua koordinat dikonversi kedalam bentuk derajat maka selanjutnya mikrokontroler akan memproses perhitungan sudut dan jaraknya dengan cara perhitungan rumus *phytagoras*.

$$\Delta Y^2 + \Delta X^2 = U^2 \text{.....(4-1)}$$

Bentuk dari dua titik koordinat yang di gambarkan dalam segitiga siku-siku ditunjukkan dalam gambar 4.16.



Gambar 4.16. Perhitungan Koordinat *Phytagoras*

Keterangan :

LCP = Lintang *Checkpoint*

BCP = Bujur *Checkpoint*

LSRU = Lintang SRU

BSRU = Bujur SRU

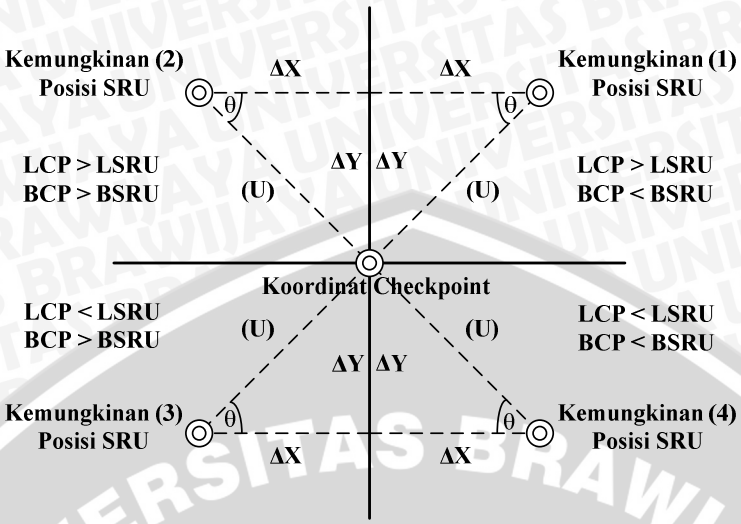
ΔY = Perbedaan LCP dan LSRU

ΔX = Perbedaan BCP dan BSRU

U = Jarak Miring (Jarak antara posisi *checkpoint* dengan posisi tim SRU)

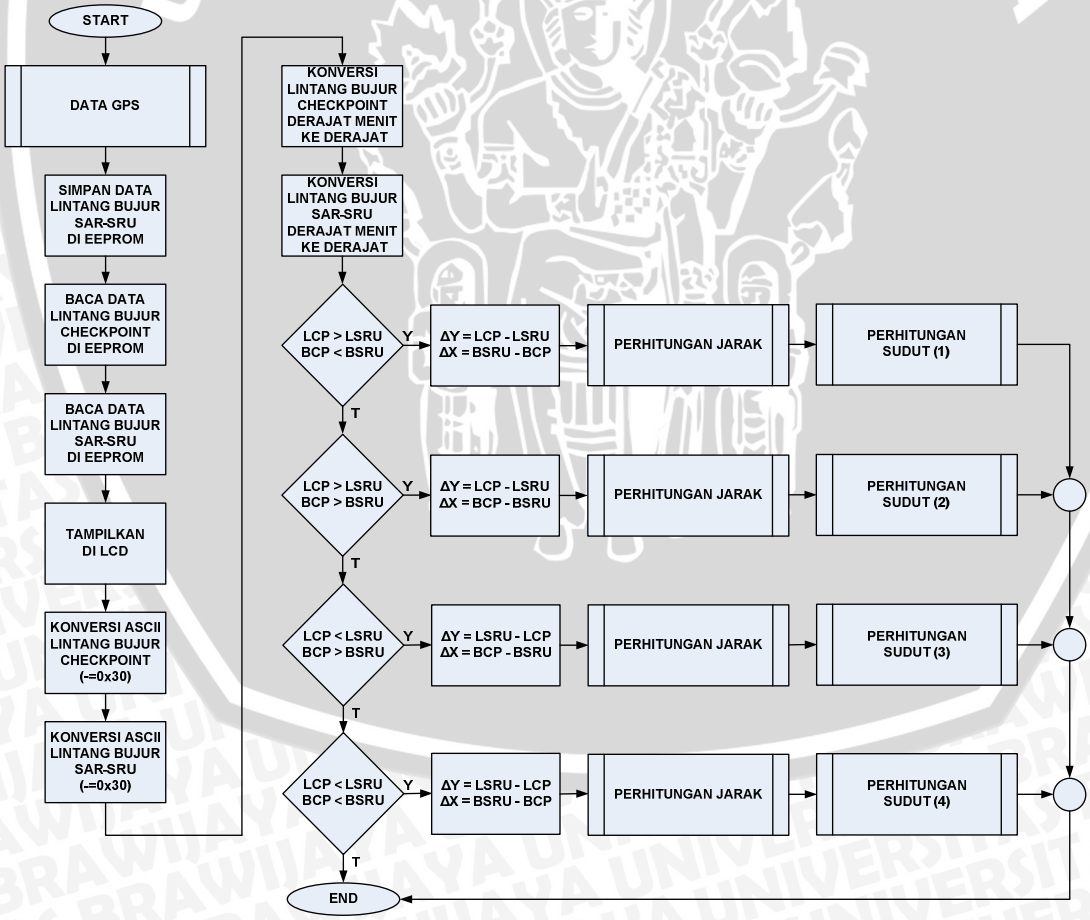
θ = Sudut yang terbentuk antara tim SRU dengan *checkpoint*

Untuk menghitung **ΔY** adalah dengan cara mencari **LCP** atau **LSRU** yang lebih besar dikurangi **LCP** atau **LSRU** yang lebih kecil. Untuk menghitung **ΔX** adalah dengan cara mencari **BCP** atau **BSRU** yang lebih besar dikurangi **BCP** atau **BSRU** yang lebih kecil. Untuk mengetahui yang besar dan kecil dari koordinat tersebut dengan cara melihat kemungkinan dari posisi tim SAR – SRU. Ada empat kemungkinan posisi tim SAR – SRU terhadap posisi *checkpoint*. Bentuk dari empat kemungkinan yang di gambarkan dalam empat kuadran posisi SRU ditunjukkan dalam gambar 4.17.



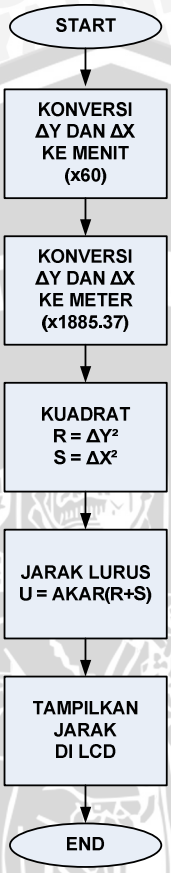
Gambar 4.17. Empat Kemungkinan Posisi SRU

Flowchart sub rutin sudut jarak ditunjukkan dalam Gambar 4.18.



Gambar 4.18. *Flowchart* Sub Rutin Sudut Jarak

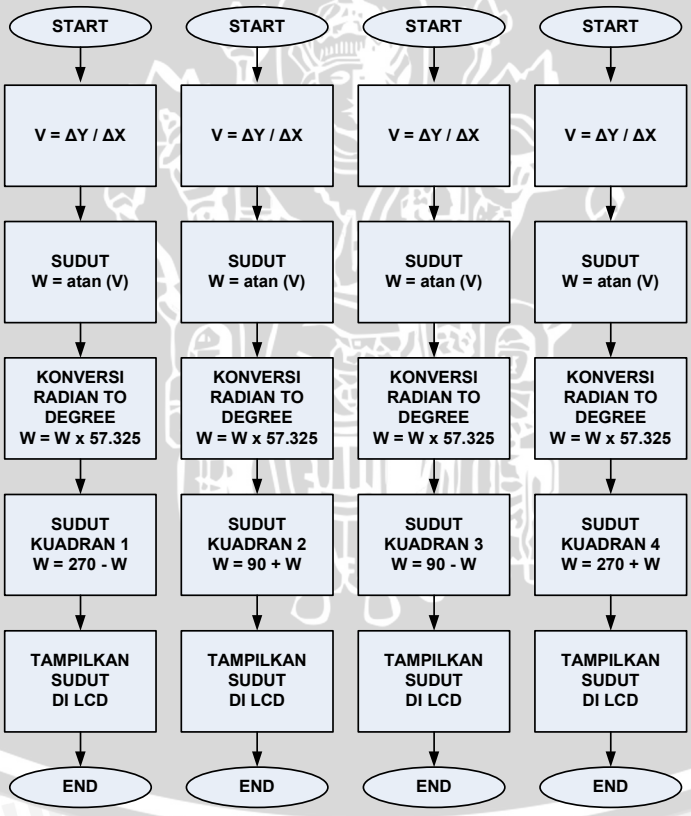
Setelah **ΔY** dan **ΔX** diketahui, selanjutnya mikrokontroler akan memproses kedalam perhitungan jarak lurus (**U**) sesuai dengan rumus *phytagoras*. *Flowchart* sub rutin perhitungan jarak ditunjukkan dalam Gambar 4.19.



Gambar 4.19. *Flowchart* Sub Rutin Perhitungan Jarak

Bentuk nilai **ΔY** dan **ΔX** adalah derajat, untuk mencari jarak maka nilai **ΔY** dan **ΔX** dikonversi kedalam menit. Nilai 1 derajat sama dengan 60 menit, maka untuk mengkonversi derajat ke menit dikalikan dengan 60. Setelah itu, dikonversi lagi ke meter. Nilai 1 menit sama dengan 1885,37 meter, maka untuk mengkonversi menit ke meter dikalikan dengan 1885,37. Jika sudah dalam satuan meter, nilai **ΔY** dan **ΔX** tinggal dimasukkan kedalam rumusan *phytagoras*, yaitu dengan cara menguadratkan nilai **ΔY** dan **ΔX** dan disimpan kedalam variabel **R** dan **S**. Sesuai dengan rumus *phytagoras*, bahwa untuk mencari panjang garis miring (jarak lurus) adalah dengan cara mencari akar dari penjumlahan nilai **R** dan **S**. Setelah itu, nilai jarak ditampilkan di LCD.

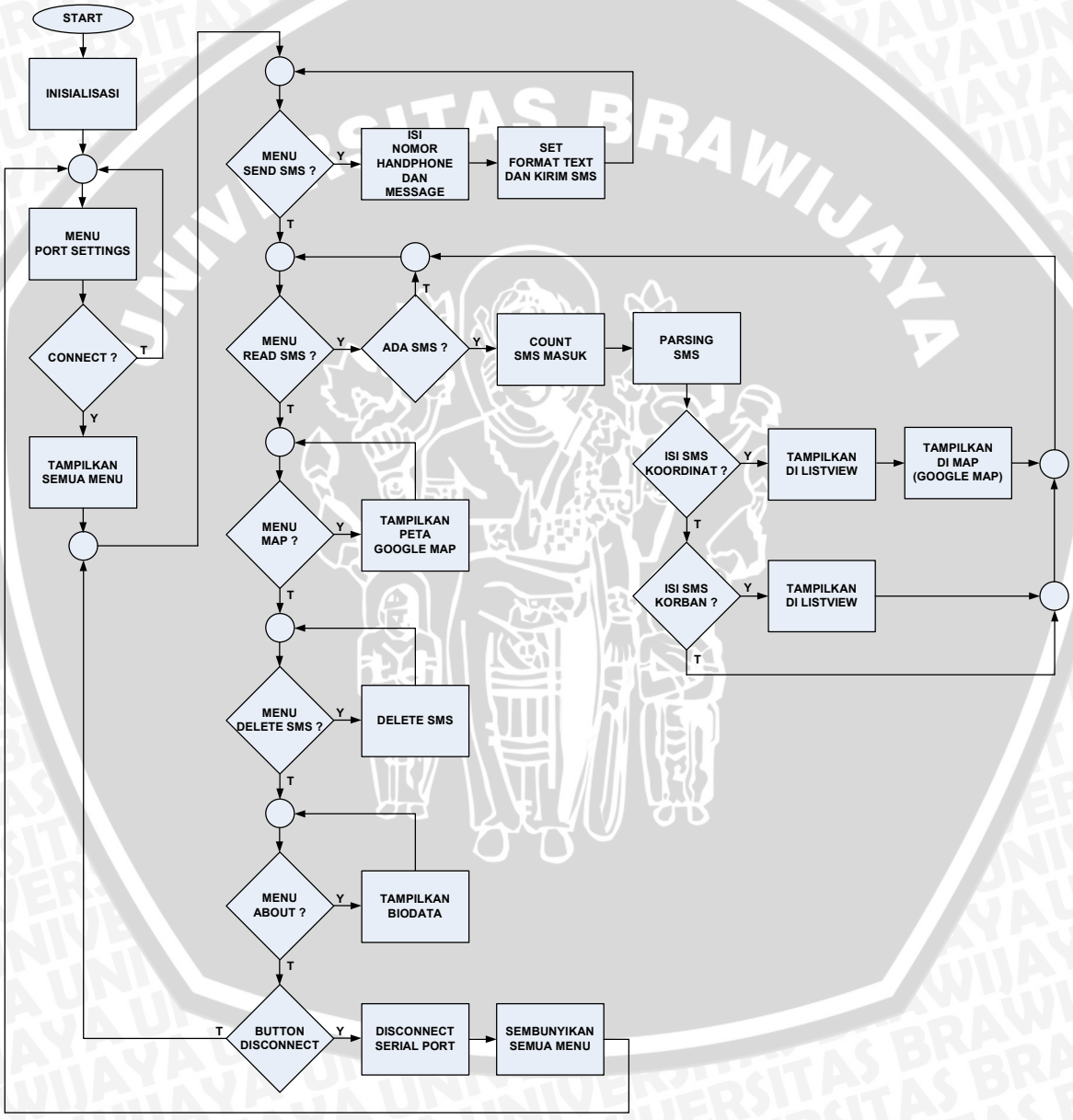
Untuk mencari nilai sudut dengan cara menghitung **ΔY/ΔX** dulu dan disimpan di variabel **V**. Setelah itu menghitung nilai **atan (V)** dan menyimpannya di variabel **W**. *Compiler Codevision* memberikan nilai sudut masih dalam satuan radian, untuk merubah ke dalam satuan derajat yaitu dengan cara mengalikan nilai sudut dengan 57,325. Seperti yang dijelaskan sebelumnya, bahwa ada empat kemungkinan posisi SRU, maka ada empat perhitungan sudut yang berbeda. Pada kemungkinan (1) nilai sudut adalah **270˚ - W**, kemungkinan (2) nilai sudut adalah **90˚ + W**, kemungkinan (3) nilai sudut adalah **90˚ - W**, kemungkinan (4) nilai sudut adalah **270˚ + W**. Setelah itu, nilai sudut ditampilkan di LCD. *Flowchart* sub rutin perhitungan sudut (1), sub rutin perhitungan sudut (2), sub rutin perhitungan sudut (3), sub rutin perhitungan sudut (4) ditunjukkan dalam Gambar 4.20 (a), Gambar 4.20 (b), Gambar 4.20 (c) dan Gambar 4.20 (d).



Gambar 4.20. (a) *Flowchart* Sub Rutin Perhitungan Sudut (1)
(b) *Flowchart* Sub Rutin Perhitungan Sudut (2)
(c) *Flowchart* Sub Rutin Perhitungan Sudut (3)
(d) *Flowchart* Sub Rutin Perhitungan Sudut (4)

4.5.2 Perancangan Perangkat Lunak Komputer

Perancangan perangkat lunak pada komputer menggunakan *Microsoft Visual C# 2005*. Untuk memberikan gambaran umum jalannya program dan memudahkan dalam pembuatan perangkat lunak, maka dibuat *flowchart* yang memudahkan jalannya program. *Flowchart* program utama komputer ditunjukkan dalam Gambar 4.21.



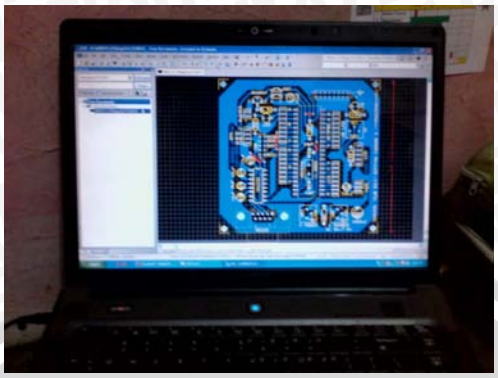
Gambar 4.21. *Flowchart* Program Utama Komputer

Saat pertama kali program dijalankan akan melakukan inisialisasi dan masuk pada menu awal yaitu menu **Port Settings**. User akan menyesuaikan *port, baudrate, data bits, stop bits* dan *parity bits* dari modem GSM. Jika konfigurasi telah sesuai maka modem GSM akan *connect* dan semua menu akan muncul. Menu-menu dalam program yaitu menu **Send SMS**, menu **Read SMS**, menu **Map**, menu **Delete SMS** dan menu **About**. Jika *user* memilih menu **Send SMS** maka *user* bisa mengirim sms dengan mengisi nomor *handphone* tujuan dan *massage*. Menu **Read SMS** berfungsi untuk membaca SMS. Jika ada sms masuk, maka SMS masuk akan dihitung dan di-*parsing* untuk memisahkan tanggal pengiriman, nomor pengirim dan isi SMS. SMS berupa jumlah korban akan ditampilkan di *listview*. SMS berupa data posisi yang berbentuk *link* koordinat akan ditampilkan di *listview* dan di peta Google *Map*. Untuk melihat peta Google *Map*, *user* dapat memilih menu **Map** dan koordinat posisi secara langsung ditampilkan di peta Google *Map*. Menu **Delete SMS** untuk mendelete SMS. Menu **About** untuk menampilkan biodata penulis. Jika *button disconnect* ditekan maka akan *disconnect* serial port dan secara otomatis menu-menu akan disembunyikan. Program akan kembali ke menu awal yaitu menu **Port Settings**. Jika *user* ingin melakukan koneksi lagi, dapat memasukkan konfigurasi port lagi. Program akan terus berjalan sampai user menginginkan untuk menutup program.

4.6 Pembuatan Perangkat Keras (*Hardware*)

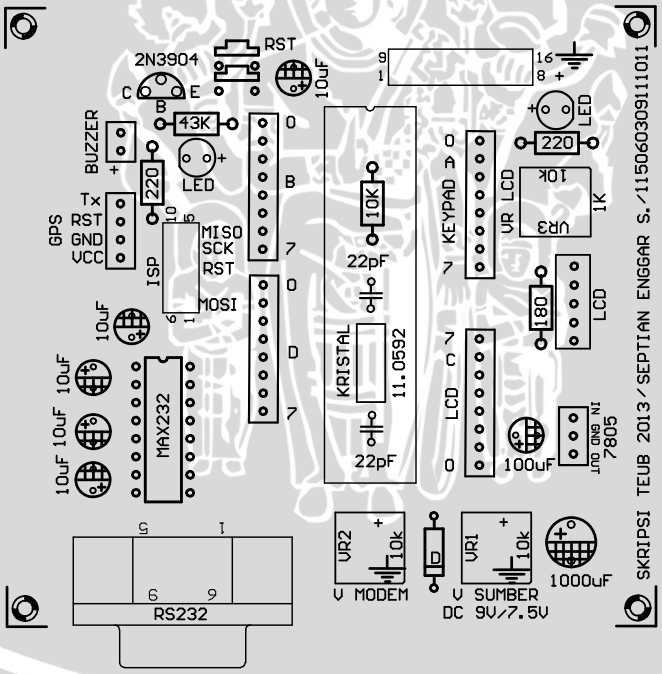
4.6.1 Pembuatan Tata Letak Komponen dan *Layout* PCB

Pembuatan tata letak komponen mempunyai nilai seni yang indah jika meletakkan komponen sesuai dengan yang diharapkan. Setiap komponen mempunyai ukuran dan jenis yang berbeda. Maka dari itu diperlukan suatu pengaturan tata letak komponen sesuai jenis dan ukuran dari masing-masing komponen. Jika perencanaan rangkaian sudah benar, maka tinggal membuat tata letak komponen pada PCB. Pembuatan tata letak komponen ini, dengan menggunakan *software* Protel DXP 2004 yang di mulai dari penggambaran rangkaian sampai perancangan tata letak komponen pada PCB. Gambar 4.22 menunjukkan proses pembuatan layout PCB pada Protel DXP 2004.



Gambar 4.22. Proses Pembuatan Layout PCB pada Protel DXP 2004

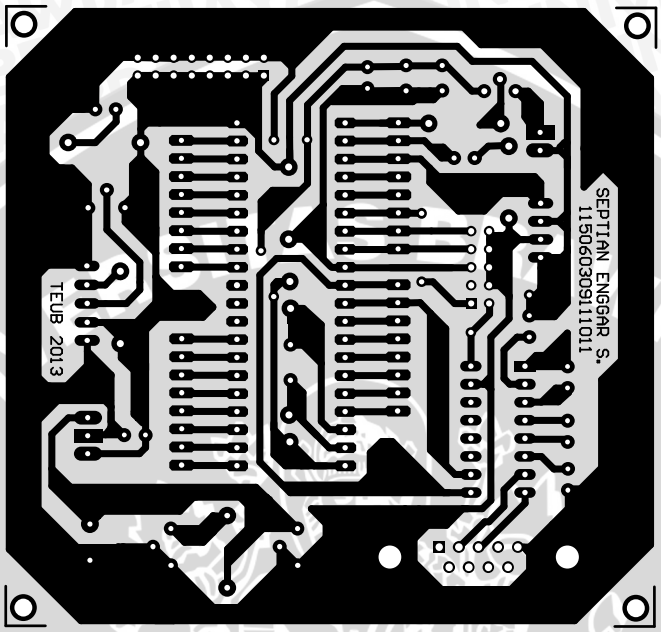
Pertama kali dalam pembuatan tata letak komponen adalah dengan menentukan terlebih dahulu besarnya PCB, setelah itu dibuat tata letak komponen dengan serapi dan seefisien mungkin dengan besarnya komponen yang akan dipergunakan. Gambar 4.23 menunjukkan tata letak minimum sistem ATMega162 dan rangkaian elektrik alat SAR.



Gambar 4.23. Tata Letak Minimum Sistem Atmega162 dan Rangkaian Elektrik Alat SAR.

Setelah pembuatan tata letak komponen, komponen-komponen tersebut dihubungkan satu sama lain sesuai dengan rangkaian yang telah direncanakan dalam perencanaan rangkaian. Jika semua komponen-komponen telah terhubung

sesuai dengan rangkaian yang telah direncanakan, maka layout tersebut kemudian di cetak (print) dikertas kemudian di fotocopy transparan yang nantinya digunakan sebagai bahan untuk penyablonan pada PCB. Gambar 4.24 menunjukkan layout PCB.



Gambar 4.24. *Layout* PCB

4.6.2 Pelarutan PCB

Sebelum melakukan pelarutan, PCB harus disablon dulu. Proses penyablonan yaitu dengan menyiapkan hasil fotocopy transparan dari *layout* yang telah dibuat. Gambar 4.25 menunjukkan hasil fotocopy transparan *layout* PCB.

Gambar 4.25. Hasil Fotocopy Transparan *Layout* PCB

Kemudian menyetrika pada bagian sisi yang tidak terdapat tinta fotocopy sampai dengan tinta tersebut melekat pada tembaga PCB, kemudian memisahkan

antara transparan dengan PCB dengan hati-hati. Gambar 4.26 menunjukkan proses penyablonan PCB.

Gambar 4.26. Proses Penyablonan PCB

PCB yang sudah disetrika tadi kemudian dilarutkan dengan campuran *fericlorit* dan air (lebih baik dengan air panas) sampai yang tersisa hanya bagian tembaga yang kena tinta dan bagian yang tidak terkena tinta akan larut. Tinta pada PCB dibersihkan dengan cairan pembersih cat (tinner atau M3) kemudian dibersihkan dengan sabun dan dibilas dengan air, lalu dikeringkan dan PCB siap untuk dilakukan pengeboran. Gambar 4.27 menunjukkan proses penyablonan PCB.

Gambar 4.27. Proses Pelarutan PCB

4.6.3 Pengeboran PCB

Setelah proses pelarutan PCB, maka selanjutnya dilakukan pengeboran pada kaki-kaki komponen pada pad PCB. Pembuatan lobang komponen (pengeboran) pada PCB harus disesuaikan dengan besarnya setiap penghantar yang ada pada masing-masing kaki komponen. Pengeboran dilakukan dengan menggunakan bor tangan untuk PCB. Gambar 4.28 menunjukkan proses pengeboran PCB.

Gambar 4.28. Proses Pengeboran PCB

Jika semua pad untuk kaki komponen pada PCB telah dibor, maka dilakukan proses selanjutnya yaitu penyolderan PCB.

4.6.4 Penyolderan PCB

Sebelum melakukan penyolderan komponen, terlebih dahulu melakukan pemasangan komponen-komponennya pada PCB sesuai dengan tata letaknya. Penyolderan dilakukan pada kaki-kaki komponen yang telah tertancap di *pad* pada jalur PCB. Gambar 4.29 menunjukkan proses penyolderan PCB. Gambar 4.30 menunjukkan hasil rangkaian elektrik.

Gambar 4.29. Proses Penyolderan PCB

Gambar 4.30. Hasil Rangkaian Elektrik

4.6.5 Pembuatan Mekanik Box Alat

Membeli *box* jadi dan membentuk serta melubangi sesuai kebutuhan komponen. Gambar 4.31 menunjukkan proses pembentukan box alat.

Gambar 4.31. Proses Pembentukan Box Alat

Kemudian melakukan pengecatan pada box alat. Gambar 4.32 menunjukkan proses pengecatan box alat.

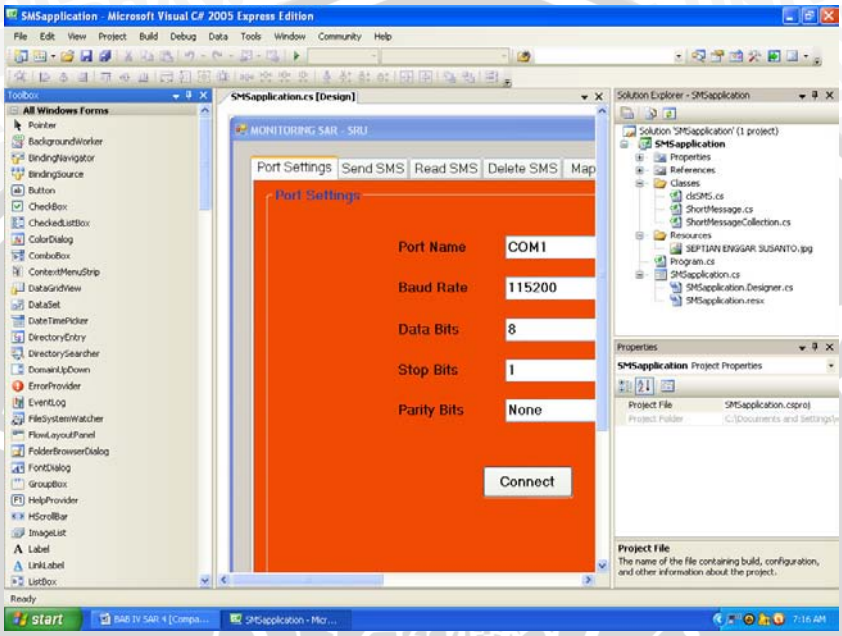
Gambar 4.32. Proses Pengecatan Box Alat

Gambar 4.33 menunjukkan hasil alat secara keseluruhan.

Gambar 4.33. Hasil Alat Secara Keseluruhan

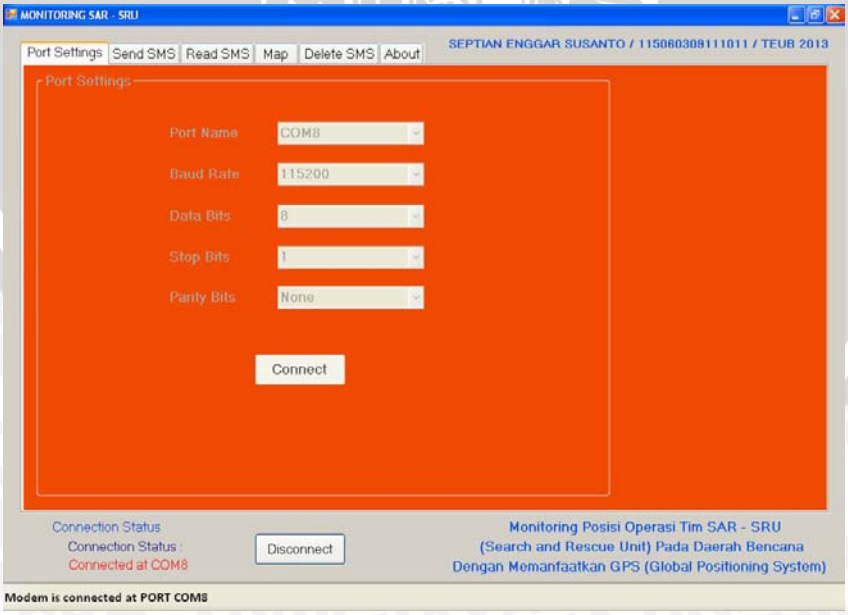
4.7 Pembuatan Perangkat Lunak (*Software*)

Pembuatan perangkat lunak pada komputer sesuai dengan *flowchart* program yang telah direncanakan dengan menggunakan Microsoft Visual C# 2005. Gambar 4.34 menunjukkan proses pembuatan perangkat lunak pada komputer.



Gambar 4.34. Proses Pembuatan Perangkat Lunak Pada Komputer.

Gambar 4.35 menunjukkan hasil perangkat lunak pada komputer.



Gambar 4.35. Hasil Perangkat Lunak pada Komputer

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 GPS *Receiver*.
2) Pengujian *Keypad*.
3) Pengujian LCD.
4) Pengujian *Driver Buzzer* dan *Buzzer*.
5) Pengujian Konverter RS232 dan Modem GSM.
6) Pengujian Perangkat Lunak.
7) Pengujian Keseluruhan Sistem.

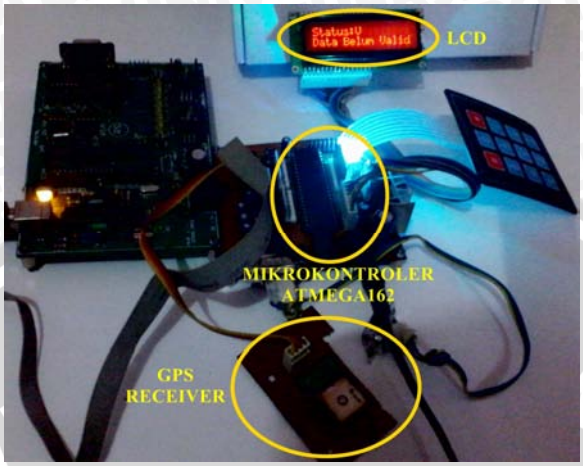
5.1 Pengujian GPS *Receiver*

Pengujian modul GPS *receiver* dilakukan dengan menggunakan mikrokontroler ATMega162 dan LCD. Pengujian ini untuk mencari koordinat posisi yang dihasilkan oleh GPS *receiver*. Output GPS *receiver* ini berupa data NMEA 0183. Kalimat NMEA 0183 yang digunakan adalah tipe GPRMC (*Recommended minimum specific* GNSS *data*). Data yang diterima mikrokontroler ATMega162 dari GPS *receiver* adalah data posisi GPS yang masih mentah, untuk itu diperlukan *parsing* data sehingga bisa dipilih data mana yang dibutuhkan saja. Diagram blok pengujian GPS *receiver* ditunjukkan dalam Gambar 5.1.

Gambar 5.1. Diagram Blok Pengujian GPS *Receiver*

Untuk menentukan posisi, mikrokontroler menunggu data valid dari GPS *receiver*. Ketika GPS *receiver* belum valid berarti GPS *receiver* masih proses

mencari posisi yang tepat. Pengujian GPS *receiver* (Data Belum Valid) ditunjukkan dalam Gambar 5.2.



Gambar 5.2. Pengujian GPS *Receiver* (Data Belum Valid)

Jika GPS sudah valid, maka mikrokontroler akan memproses dan melakukan *parsing* data dari GPS *receiver*. Gambar data hasil GPS *receiver* (Koordinat Lintang) dan data hasil GPS *receiver* (Koordinat Bujur) ditunjukkan dalam Gambar 5.3 (a) dan Gambar 5.3 (b).

(a) (b)

Gambar 5.3. (a) Data HasilGPS *Receiver* (Koordinat Lintang)
(b) Data Hasil GPS *Receiver* (Koordinat Bujur)

Dari Gambar 5.3 didapat posisi dengan koordinat 07˚56’.8692 LS dan 112˚36’.9491 BT. Hal ini menunjukkan bahwa modul GPS *receiver* bekerja dengan baik, yaitu dapat menghasilkan koordinat yang benar.

5.2 Pengujian *Keypad*

Pengujian ini bertujuan untuk melihat data yang ditekan pada *keypad* dapat ditampilkan dengan benar di LCD. Ketika *keypad* ditekan maka mikrokontroler akan mendeteksi data input yang diberikan oleh *keypad*, mikrokontroler akan memproses dan menampilkan data *keypad* pada LCD. Pengujian *keypad* dilakukan dengan menggunakan mikrokontroler ATMega162 dan LCD. Diagram blok pengujian *keypad* ditunjukkan dalam Gambar 5.4.

Gambar 5.4. Diagram Blok Pengujian *Keypad*

Pengujian *keypad* ditunjukkan dalam Gambar 5.5.

Gambar 5.5. Pengujian *Keypad*

Pengujian *keypad* ini dengan cara menekan *keypad* satu persatu untuk melihat data yang diberikan oleh *keypad* apakah sudah sesuai dengan data yang ditampilkan di LCD. Data Hasil Pengujian *keypad* ditunjukkan dalam Tabel 5.1.

Tabel 5.1. Data Hasil Pengujian *Keypad*

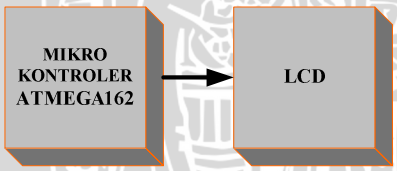
INPUT KEYPAD	HASIL LCD
Ditekan 0	0
Ditekan 1	1
Ditekan 2	2
Ditekan 3	3

Ditekan 4	4
Ditekan 5	5
Ditekan 6	6
Ditekan 7	7
Ditekan 8	8
Ditekan 9	9
Ditekan *	*
Ditekan #	#

Dari data hasil pengujian menunjukkan bahwa *keypad* bekerja dengan baik, yaitu dapat memberikan data yang benar kepada mikrokontroler dan ditampilkan di LCD.

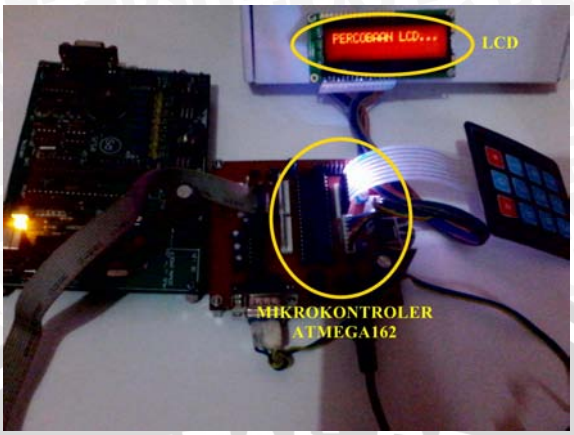
5.3 Pengujian LCD

Pengujian LCD dilakukan untuk menguji *interface* port C mikrokontroler terhadap LCD 2x16. Pengujian tersebut dilakukan dengan cara menulis atau menampilkan karakter pada LCD yang diperintah oleh mikrokontroler melalui program yang dimasukkan kedalamnya. Diagram blok pengujian LCD ditunjukkan dalam Gambar 5.6.



Gambar 5.6. Diagram Blok Pengujian LCD

Prosedur pengujian rangkaian yang pertama pada LCD adalah membuat program mikrokontroler. Program tersebut berisi perintah untuk menampilkan karakter tertentu pada LCD. Pada pengujian ini program akan mengirimkan karakter "**PERCOBAAN LCD...**" untuk ditampilkan di LCD. Pengujian LCD ditunjukkan dalam Gambar 5.7.

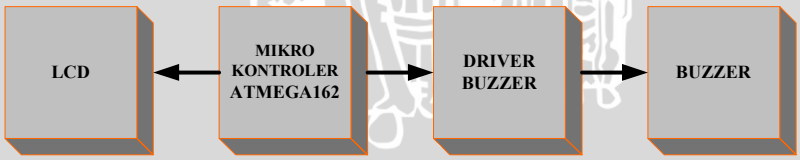


Gambar 5.7. Pengujian LCD

Dari hasil pengujian yang telah dilakukan membuktikan bahwa LCD bekerja dengan baik. LCD bisa menampilkan data berupa karakter dari program yang telah diisikan pada mikrokontroller ATMega162.

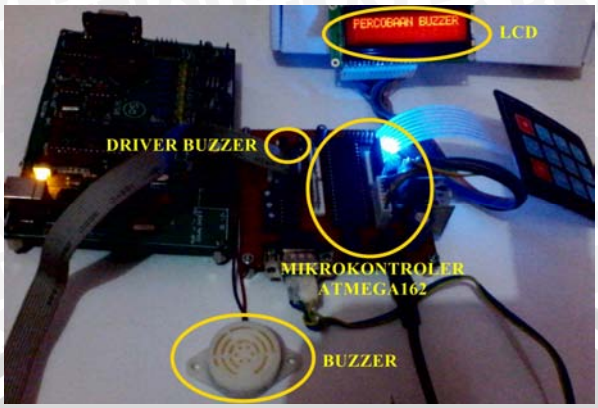
5.4 Pengujian *Driver Buzzer* dan *Buzzer*

Pengujian pada blok ini dilakukan untuk menganalisis kerja *driver buzzer* dan *buzzer*. Pengujian tersebut dilakukan dengan cara mengaktifkan dan menonaktifkan *buzzer* yang diperintah oleh mikrokontroler melalui program yang dimasukkan kedalamnya. Mikrokontroler akan mengendalilakn *driver buzzer* untuk mengaktifkan dan menonaktifkan *buzzer*. Diagram blok pengujian *driver buzzer* dan *buzzer* ditunjukkan dalam Gambar 5.8.



Gambar 5.8. Diagram Blok Pengujian *Driver Buzzer* dan *Buzzer*

Mikrokontroler akan mengeluarkan logika 1 (*High*) selama 800ms ke *driver buzzer* untuk mengaktifkan *buzzer* selama 800ms dan mengeluarkan logika 0 (*Low*) selama 800ms ke *driver buzzer* untuk menonaktifkan *buzzer* selama 800ms. Pengujian *driver buzzer* dan *buzzer* ditunjukkan dalam Gambar 5.9.



Gambar 5.9. Pengujian *Driver Buzzer* dan *Buzzer*

Data Hasil Pengujian *driver buzzer* dan *buzzer* ditunjukkan dalam Tabel 5.2.

Tabel 5.2. Data Hasil Pengujian *Driver Buzzer* dan *Buzzer*

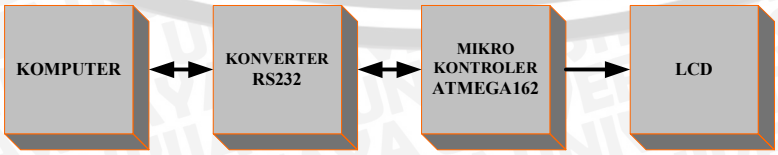
OUTPUT MIKROKONTROLER	***BUZZER***
LOGIKA 1 (*HIGH*)	AKTIF
LOGIKA 0 (*LOW*)	NON AKTIF

Dari data hasil pengujian menunjukkan bahwa *driver buzzer* dan *buzzer* bekerja dengan baik, yaitu dapat aktif ketika mikrokontroler memberikan logika 1 (*High*) dan *buzzer* non aktif ketika mikrokontroler memberikan logika 1 (*Low*).

5.5 Pengujian Konverter RS232 dan Modem GSM

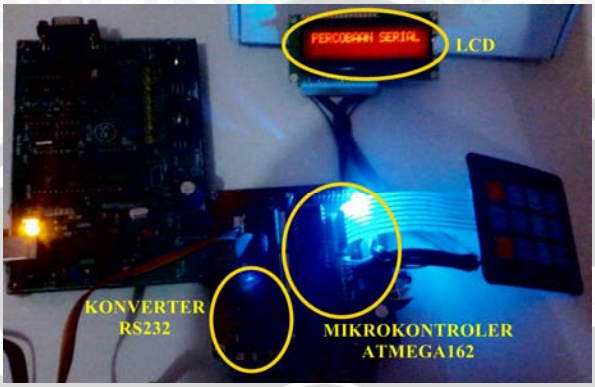
5.5.1 Pengujian Konverter RS232

Sebelum pengujian modem GSM terlebih dahulu akan dilakukan pengujian konverter RS232 untuk memastikan bahwa konverter RS232 dapat mengirimkan data secara serial. Pengujian konverter RS232 dilakukan dengan cara mengirimkan data secara serial dari mikrokontroler ke komputer. Diagram blok pengujian konverter RS232 ditunjukkan dalam Gambar 5.10.



Gambar 5.10. Diagram Blok Pengujian Konverter RS232

Pengujian konverter RS232 dilakukan dengan menggunakan mikrokontroler ATMega162, LCD dan komputer. Mikrokontroler mengirimkan data karakter "**SAR**" ke komputer melalui konverter RS232. Pengujian konverter RS232 ditunjukkan dalam Gambar 5.11.



Gambar 5.11. Pengujian Konverter RS232

Data dari mikrokontroler akan ditampilkan melalui menu terminal *codevisionAVR* dengan konfigurasi yang sama dengan mikrokontroler yaitu:

Baud rate = 115200 bps
Data Bits = 8
Stop Bits = 1
Parity = None

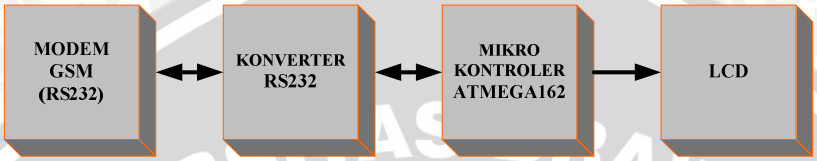
Jika konfigurasi telah dilakukan maka selanjutnya menekan tombol *connect* pada terminal *codevisionAVR*. Hasil pengujian konverter RS232 ditunjukkan dalam Gambar 5.12.

Gambar 5.12. Hasil Pengujian Konverter RS232

Dari hasil pengujian menunjukkan bahwa konverter RS232 bekerja dengan baik, yaitu dapat mengirimkan karakter "**SAR**" ke komputer secara terus menerus.

5.5.2 Pengujian Modem GSM (RS232)

Setelah pengujian konverter RS232 dilakukan dengan baik maka selanjutnya adalah melakukan pengujian modem GSM (RS232). Pengujian modem GSM (RS232) dilakukan dengan menggunakan mikrokontroler ATMega162, konverter RS232 dan LCD. Diagram blok pengujian modem GSM (RS232) ditunjukkan dalam Gambar 5.13.



Gambar 5.13. Diagram Blok Pengujian Modem GSM (RS232)

Program Mikrokontroler diatur dengan konfigurasi yang sama dengan modem GSM (RS232) yaitu:

Baud rate = 115200 bps
Data Bits = 8
Stop Bits = 1
Parity = None

Prosedur pengujian modem GSM (RS232) adalah membuat program mikrokontroler untuk menerima SMS. Program tersebut memerintahkan modem GSM (RS232) untuk siap menerima SMS dan mikrokontroler menyeleksi SMS yang sesuai format. Jika isi SMS sesuai format maka isi SMS tersebut akan ditampilkan di LCD. Format SMS yang diterima adalah "**GPS**" dan "**KORBAN**". Jika isi SMS tidak sesuai format maka LCD akan menampilkan "**Format SMS Salah**". Pengujian modem GSM (RS232) ditunjukkan dalam Gambar 5.14.

Gambar 5.14. Pengujian Modem GSM

Data Hasil Pengujian modem GSM (RS232) ditunjukkan dalam Tabel 5.3.

Tabel 5.3. Data Hasil Pengujian Modem GSM (RS232)

ISI SMS	TAMPILAN LCD
GPS	GPS
KORBAN	KORBAN
COBA	Format SMS Salah

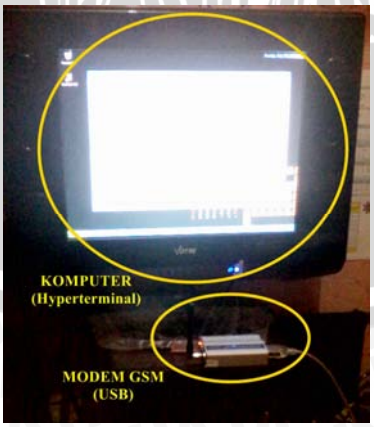
Dari data hasil pengujian menunjukkan bahwa modem GSM (RS232) bekerja dengan baik, yaitu dapat menerima SMS yang sesuai format dan mikrokontroler menampilkannya di LCD. Modem GSM (RS232) juga dapat menerima SMS yang tidak sesuai format dan mikrokontroler menampilkan "**Format SMS Salah**" di LCD.

5.5.3 Pengujian Modem GSM (USB)

Pengujian modem GSM (USB) dilakukan dengan menggunakan *hyperterminal* pada komputer. Diagram blok pengujian modem GSM (USB) ditunjukkan dalam Gambar 5.15.

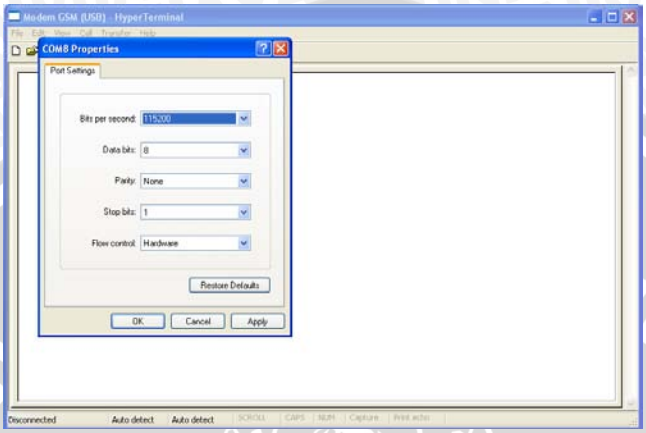
Gambar 5.15. Diagram Blok Pengujian Modem GSM (USB)

Pengujian modem GSM (USB) ditunjukkan dalam Gambar 5.16.



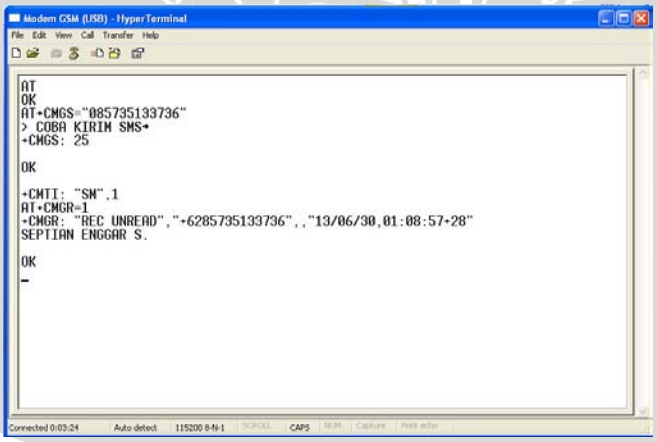
Gambar 5.16. Pengujian Modem GSM (USB)

Untuk menghubungkan koneksi komputer dengan modem GSM (USB) maka pada *hyperterminal* harus dilakukan konfigurasi sesuai dengan konfigurasi modem GSM (USB). Konfigurasi *hyperterminal* pada komputer ditunjukkan dalam Gambar 5.17.



Gambar 5.17. Konfigurasi *Hyperterminal* pada Komputer

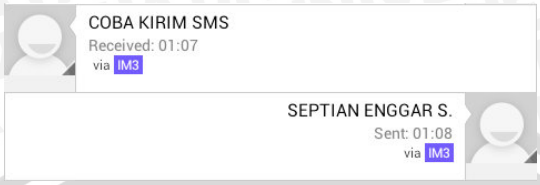
Jika konfigurasi telah dilakukan dan modem GSM (USB) sudah *connect* dengan *hyperterminal* komputer maka pengujian dilakukan dengan mencoba mengetik AT *Command* berupa "**AT**", jika hasilnya "**OK**" maka modem bisa digunakan. Pengujian modem GSM (USB) ditunjukkan dalam Gambar 5.18.



Gambar 5.18. Pengujian Modem GSM (USB)

Pengujian dilanjutkan dengan mengirim SMS yaitu dengan mengetik "**AT+CMGS="085735133736"**", mengetik SMS dengan isi "**COBA KIRIM SMS**" dan mengirim SMS dengan menekan **ctrl+z** pada *keyboard*. Jika ada balasan "**OK**" berarti SMS telah terkirim ke nomor *handphone* tujuan

085735133736. Untuk mengecek apakah SMS sudah diterima dapat dilihat pada layar *handphone*. *Screenshots handphone* pengujiam modem GSM (USB) ditunjukkan dalam Gambar 5.19.



Gambar 5.19. *Screenshots Handphone* Pengujian Modem GSM (USB)

Selanjutnya pengujian modem GSM (USB) untuk menerima SMS. Pada pengujian ini, handphone mengirim SMS ke modem GSM (USB) seperti dalam Gambar 5.19 yaitu dengan isi SMS "**SEPTIAN ENGGAR S.**". Modem GSM (USB) mendeteksi adanya SMS masuk dengan adanya tanda "**+CMTI: "SM" ,1**". Untuk membuka SMS dengan cara mengetik "**AT+CMGR=1**" dan isi SMS "**SEPTIAN ENGGAR S.**" akan terlihat seperti dalam Gambar 5.18.

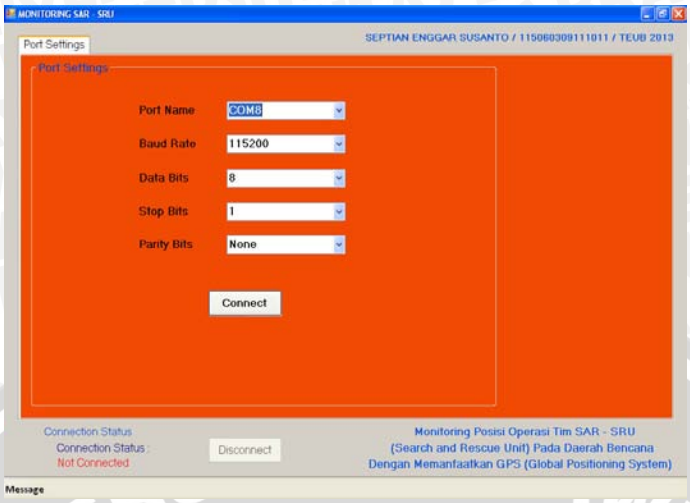
Dari hasil pengujian menunjukkan bahwa modem GSM (USB) bekerja dengan baik, yaitu dapat mengirim SMS dan menerima SMS sesuai dengan isi SMS yang dikirimkan.

5.6 Pengujian Perangkat Lunak

Pengujian perangkat lunak dilakukan dengan menggunakan modem GSM (USB). Diagram blok pengujian perangkat lunak ditunjukkan dalam Gambar 5.20.

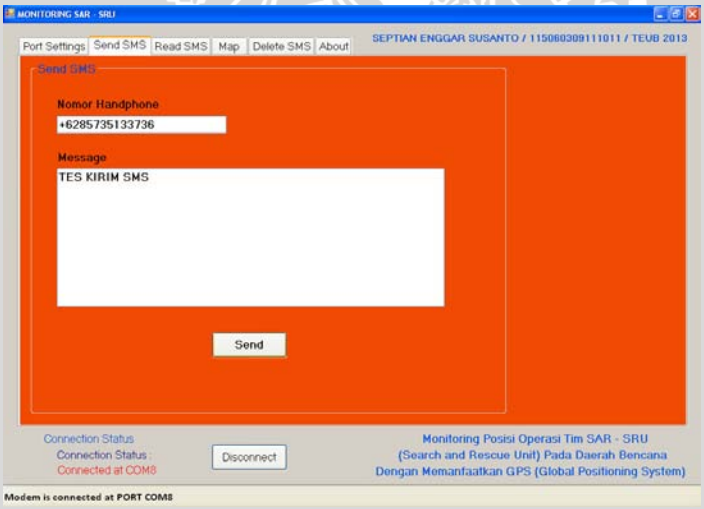
Gambar 5.20. Diagram Blok Pengujian Perangkat Lunak

Perangkat lunak dibuat dengan menggunakan software *Microsoft Visual C# 2005.* Pengujian perangkat lunak ini bertujuan untuk menguji masing-masing menu yang ada dalam perangkat lunak yang telah dibuat. Menu pertama adalah menu **Port Settings**. Pada menu **Port Settings** ini berfungsi untuk melakukan konfigurasi port dengan modem GSM (USB). Menu **Port Settings** ditunjukkan dalam Gambar 5.21.



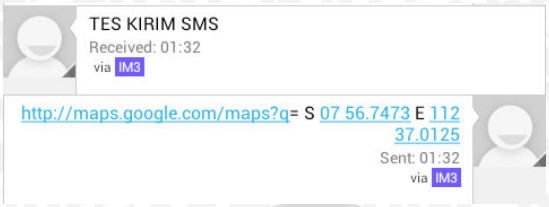
Gambar 5.21. Menu Port Settings

Jika konfigurasi telah dilakukan dan modem GSM (USB) sudah *connect* dengan perangkat lunak komputer maka selanjutnya melakukan pengujian pada menu berikutnya yaitu menu **Send SMS**. Menu **Send SMS** ditunjukkan dalam Gambar 5.22.



Gambar 5.22. Menu Send SMS

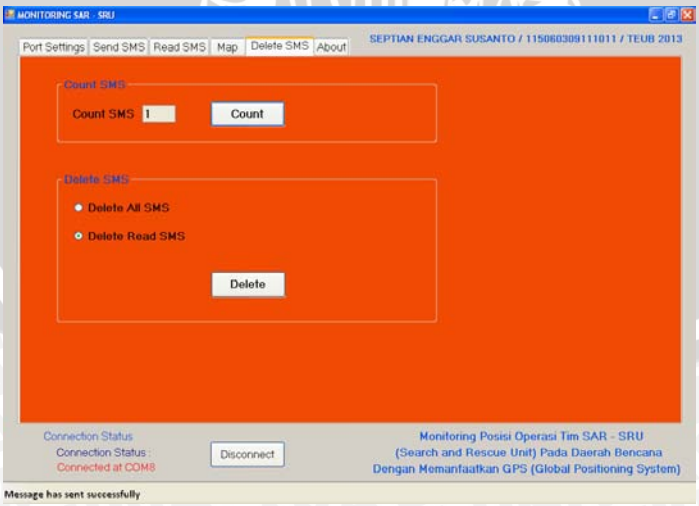
Untuk pengiriman SMS dengan cara mengisi **Nomor Handphone** dan mengetik isi SMS selanjutnya menekan tombol **Send** untuk mengirim SMS. Pada pengujian ini mengirim SMS ke nomor tujuan "**+6285735133736**" dengan isi SMS "**TES KIRIM SMS**". Untuk mengecek apakah SMS sudah diterima dapat dilihat pada layar *handphone*. *Screenshots handphone* pengujian perangkat lunak ditunjukkan dalam Gambar 5.23.



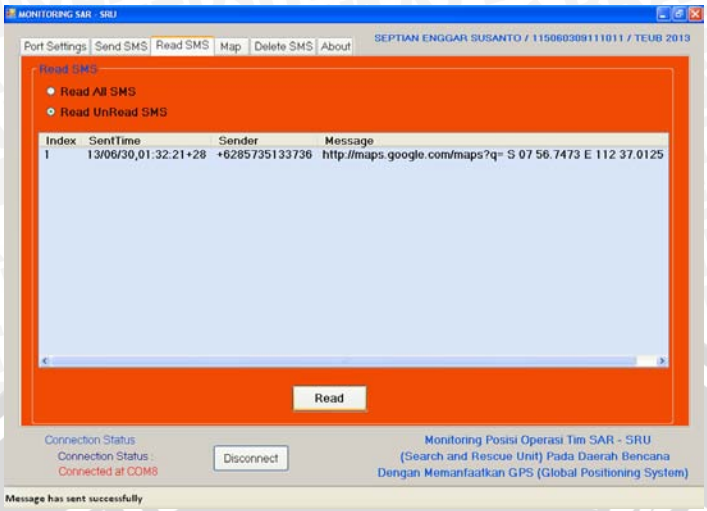
Gambar 5.23. *Screenshots Handphone* Pengujian Perangkat Lunak

Isi SMS "**TES KIRIM SMS**" telah diterima oleh *handphone* tujuan seperti terlihat dalam Gambar 5.23 berarti menu **Send SMS** telah bekerja dengan baik.

Selanjutnya pengujian menu **Read SMS** untuk menerima SMS. Pada pengujian ini, **handphone** mengirim SMS ke perangkat lunak seperti dalam Gambar 5.23 yaitu dengan isi SMS "**http://maps.google.com/maps?q=S07 56.7473 E112 37.0125**". Untuk mendeteksi adanya SMS masuk yaitu dengan cara menekan tombol **Count.** Jika **Count SMS** menunjukkan angka **1** berarti ada SMS masuk. Menu **Count SMS** ditunjukkan dalam Gambar 5.24. Untuk membuka SMS dengan cara masuk ke menu **Read SMS** dan menekan tombol **Read.** Menu **Read SMS** ditunjukkan dalam Gambar 5.25. Setelah tombol **Read** ditekan akan muncul isi SMS yang sudah di-*parsing* dan data isi SMS akan menempati sesuai dengan kolomnya yaitu **Index, SentTime, Sender** dan **Message**. Pada kolom **Message** terlihat bahwa isi SMS **"http://maps.google.com/maps?q=S07 56.7473 E112 37.0125"** telah diterima sesuai isi SMS yang dikirim oleh *handphone* seperti dalam Gambar 5.23. Hal ini berarti menu **Read SMS** telah bekerja dengan baik.

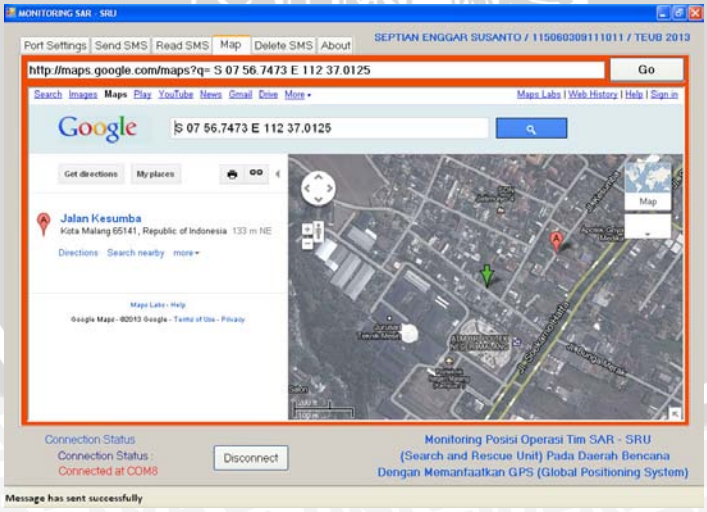


Gambar 5.24. Menu Count SMS



Gambar 5.25. Menu Read SMS

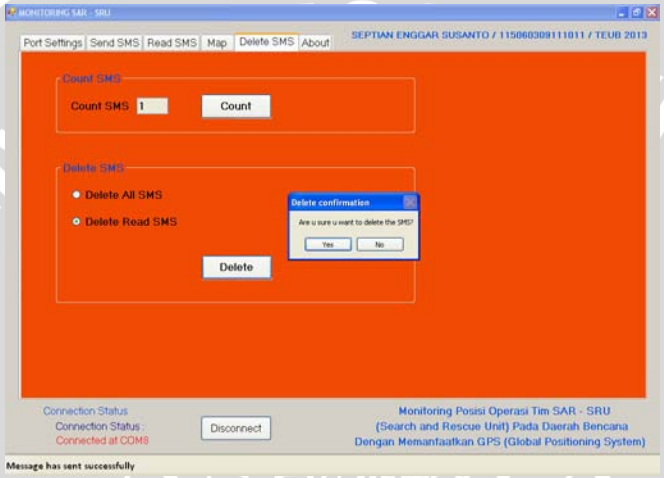
Pengujian menu **Map** bertujuan untuk menguji bahwa seandainya ada SMS masuk berupa posisi yang sudah berbentuk *link* maka menu **Map** bisa menunjukkan lokasi posisi pada peta Google *Map*. Untuk menunjukkan lokasi posisi pada peta Google *Map* yaitu dengan cara menekan tombol **Go**. Isi SMS masuk yang berupa posisi yang sudah berbentuk *link* akan mengisi otomatis pada text URL sehingga *user* tinggal menekan tombol **Go** untuk melihat posisi petanya. Menu **Map** ditunjukkan dalam Gambar 5.26. Lokasi posisi akan ditunjukkan oleh tanda anak panah yang berwarna hijau.



Gambar 5.26. Menu Map

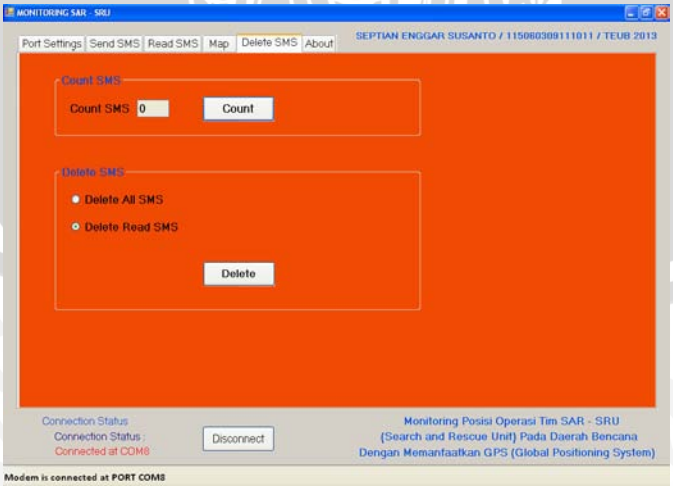
Lokasi posisi telah ditunjukkan dengan benar oleh anak panah yang berwarna hijau pada peta Google *Map* seperti pada Gambar 5.26. Hal ini berarti menu **Map** telah bekerja dengan baik.

Untuk menghapus sms yaitu dengan cara masuk ke menu **Delete SMS**. Jika ingin menghapus SMS yang telah dibaca yaitu dengan meletakkan titik ke **Delete Read SMS** dan menekan tombol **Delete**. Jika ingin menghapus semua SMS yaitu dengan meletakkan titik ke **Delete All SMS** dan menekan tombol **Delete**. Menu **Delete SMS** ditunjukkan dalam Gambar 5.27.



Gambar 5.27. Menu Delete SMS

Selanjutnya akan ada "**Delete Confirmation**", dengan menekan **Yes** maka SMS akan dihapus. Jika SMS telah terhapus semua, maka isi **Count SMS** menjadi **0.** Menu **Count SMS** (Berjumlah 0) ditunjukkan dalam Gambar 5.28.



Gambar 5.28. Menu Count SMS (Berjumlah 0)

Dengan isi **Count SMS** = 0 maka SMS telah terhapus semua dan hal ini berarti menu **Delete SMS** telah bekerja dengan baik.

Yang terakhir adalah menu **About**. Menu ini berfungsi untuk menampilkan biodata dan hasil alat yang telah dibuat. Menu **About** ditunjukkan dalam Gambar 5.29.

Gambar 5.29. Menu About

Dari pengujian semua menu perangkat lunak yang telah dilakukan, semuanya telah bekerja dengan baik. Maka perangkat lunak yang telah dibuat siap untuk dioperasikan dengan perangkat keras untuk pengujian secara keseluruhan sistem.

5.7 Pengujian Keseluruhan Sistem

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 posisi pada komputer.

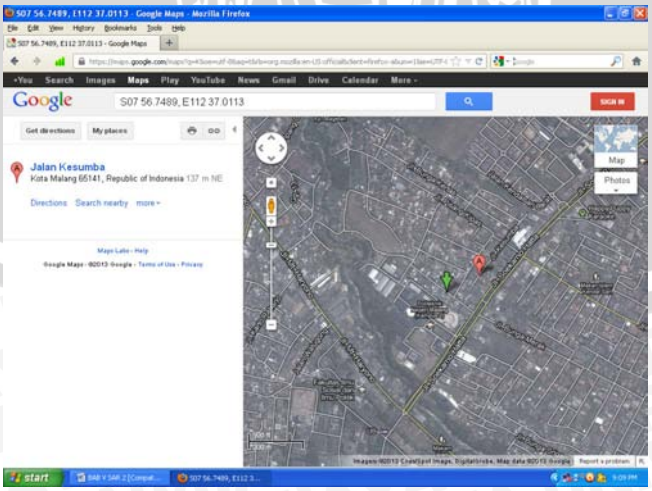
Pengujian sistem secara keseluruhan ini dilakukan di tujuh tempat yaitu di Bundaran Universitas Brawijaya, Museum Brawijaya (Jalan Ijen), Tugu Kapal (Jalan Soekarno-Hatta), Rumah Sakit Universitas Brawijaya, Taman Makam Pahlawan, Dekanat FTUB, dan Politeknik Negeri Malang. Masing-masing tempat

dilakukan pengujian permintaan posisi, pengujian permintaan jumlah korban dan pengujian pengukuran sudut dan jarak.

Sistem alat tim SAR pemantau berupa modem GSM (USB) dan laptop yang terhubung dengan internet untuk Google *Map*. Sistem alat tim SAR – SRU adalah alat yang dibawa ke tiga tempat pengujian. Sebelum melakukan pengujian terlebih dahulu melakukan pengambilan data koordinat *checkpoint* di tempat sistem tim SAR pemantau. Data koordinat *checkpoint* disimpan di EEPROM mikrokontroler. Data *checkpoint* ini berfungsi untuk perhitungan sudut dan jarak pada tempat-tempat pengujian. Koordinat *checkpoint* ditunjukkan dalam Gambar 5.30.

Gambar 5.30. Koordinat *Checkpoint*

Koordinat checkpoint yang telah diambil di tempat sistem tim SAR pemantau dapat dilihat pada peta Google *Map*. Letak posisi *checkpoint* ditunjukkan dalam Gambar 5.31.



Gambar 5.31. Letak Posisi *Checkpoint*

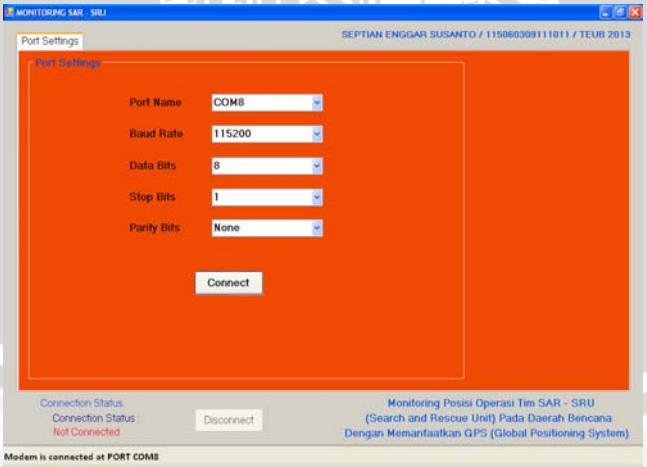
5.7.1 Pengujian di Bundaran Universitas Brawijaya

Pengujian pertama dilakukan di Bundaran Universitas Brawijaya. Dalam pengujian ini akan mencoba respon permintaan posisi, permintaan jumlah korban oleh sistem tim SAR pemantau (komputer) serta pengukuran jarak dan sudut. Pengujian di Bundaran Universitas Brawijaya ditunjukkan dalam Gambar 5.32.

Gambar 5.32. Pengujian di Bundaran Universitas Brawijaya

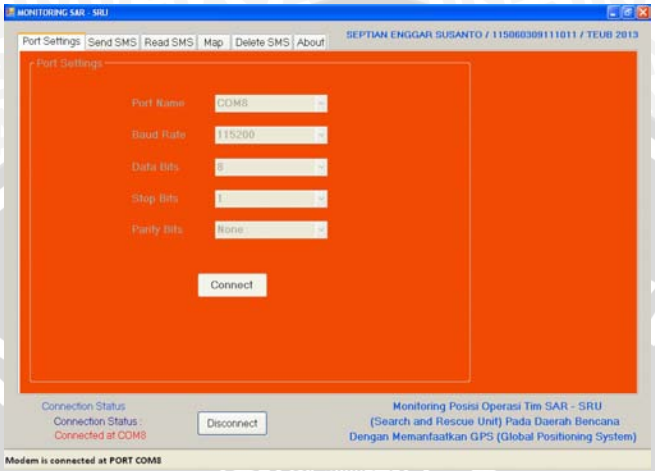
- **Pengujian Permintaan Posisi**

Pada sistem tim SAR pemantau menggunakan perangkat lunak dan modem GSM (USB). Menghubungkan modem GSM (USB) pada komputer dan mengatur konfigurasi port pada menu **Port Settings**. Konfigurasi port modem GSM (USB) ditunjukkan dalam Gambar 5.33.



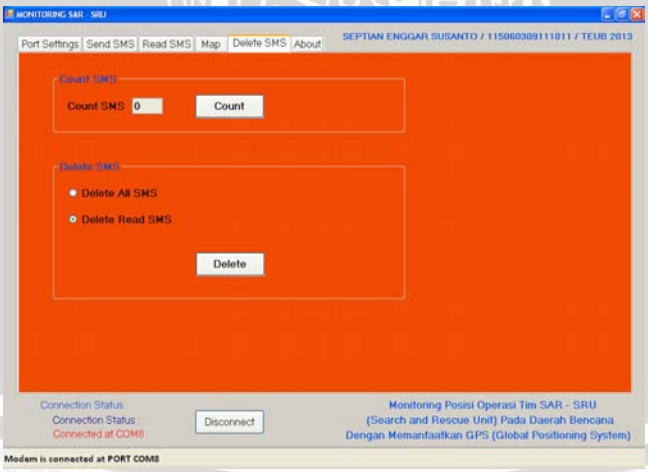
Gambar 5.33. Konfigurasi Port Modem GSM (USB)

Jika konfigurasi port telah sesuai konfigurasi modem GSM (USB) selanjutnya menekan tombol **Connect**. Ketika modem GSM (USB) sudah *connect* dengan perangkat lunak maka semua menu akan muncul. Perangkat lunak setelah *connect* ditunjukkan dalam Gambar 5.34.



Gambar 5.34. Perangkat Lunak Setelah *Connect*

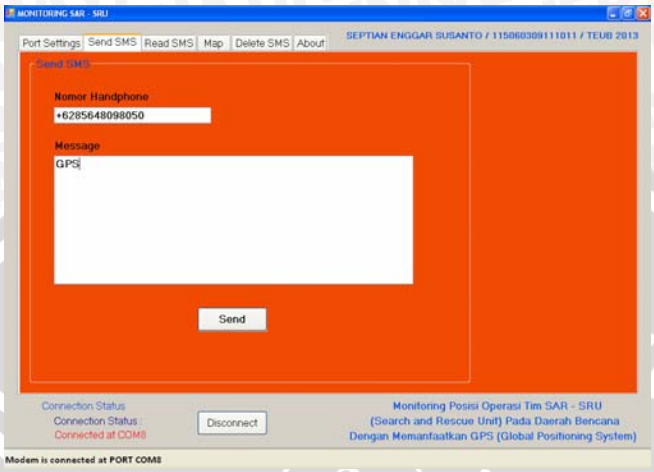
Memastikan bahwa isi *memory* simcard modem GSM (USB) masih kosong dan belum ada SMS yaitu dengan cara menekan tombol **Count**. Jika **Count SMS** berjumlah 0 berarti simcard masih kosong. Tampilan **Count SMS** berjumlah 0 ditunjukkan dalam Gambar 5.35.



Gambar 5.35. Tampilan Count SMS Berjumlah 0

Selanjutnya adalah melakukan pengiriman SMS untuk permintaan posisi ke sistem tim SAR – SRU dengan cara masuk ke menu **Send SMS**. Mengisi **Nomor**

Handphone sistem tim SAR – SRU, mengetik isi SMS dengan "**GPS**" dan menekan tombol **Send** untuk mengirim SMS. Pengiriman SMS untuk permintaan posisi (Bundaran UB) ditunjukkan dalam Gambar 5.36.



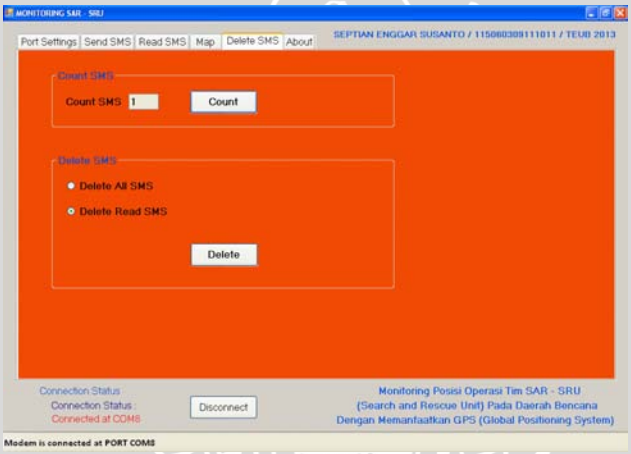
Gambar 5.36. Pengiriman SMS Untuk Permintaan Posisi (Bundaran UB)

Pada sistem tim SAR – SRU di atur pada menu **Terima SMS** untuk siap menerima perintah dari sistem tim SAR pemantau. Proses menunggu SMS (Bundaran UB) ditunjukkan dalam Gambar 5.37 (a). Setelah SMS permintaan posisi telah diterima, maka GPS akan aktif dan mencari koordinat posisinya. Jika koordinat posisi sudah valid selanjutnya data posisi akan dikirim ke sistem tim SAR pemantau (Komputer dan Handphone). Data posisi terkirim (Bundaran UB) ditunjukkan dalam Gambar 5.37 (b).

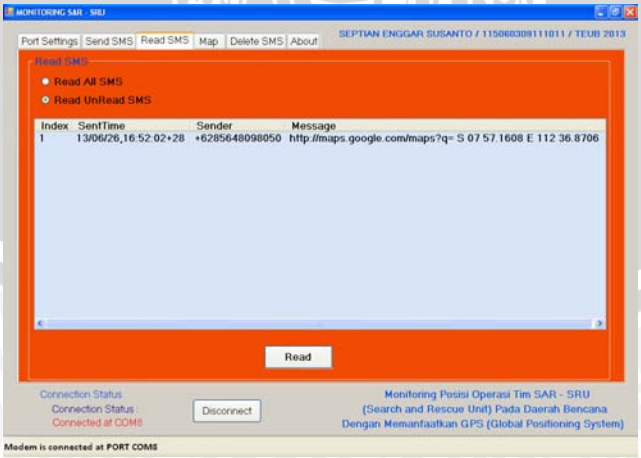
(a) (b)

Gambar 5.37. (a) Proses Menunggu SMS (Bundaran UB)
(b) Data Posisi Terkirim (Bundaran UB)

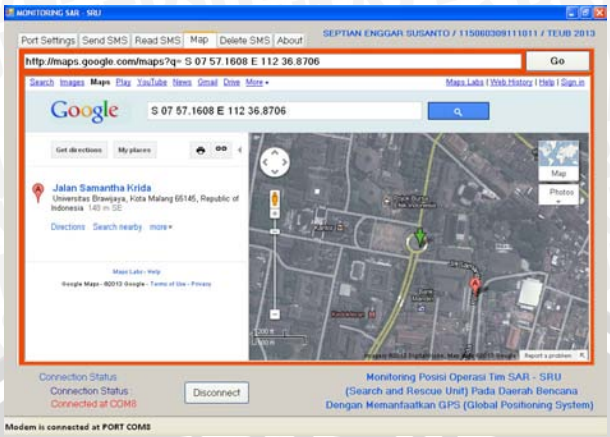
Selanjutnya mengecek SMS masuk pada perangkat lunak dengan menekan tombol **Count**. Jika **Count SMS** berjumlah 1 berarti ada 1 SMS masuk. Tampilan **Count SMS** berjumlah 1 ditunjukkan dalam Gambar 5.38. Melihat isi SMS dengan cara masuk ke menu **Read SMS** dan menekan tombol **Read**. Isi SMS berupa data posisi akan terlihat pada *listview*. Tampilan **Read SMS** dengan isi SMS pada index 1 ditunjukkan dalam Gambar 5.39. Untuk melihat posisi dengan cara masuk menu **Map** dan menekan tombol **Go**. Komputer dengan terhubung internet akan menampilkan posisi sistem tim SAR – SRU pada Google *Map*. Posisi ditunjukkan oleh anak panah yang berwarna hijau. Posisi tersebut berbeda sekitar 5 meter dengan posisi sebenarnya tempat pengujian. Perbedaan tersebut masih termasuk dalam batas range keakuratan GPS *receiver*. Posisi di Google *Map* (Bundaran UB) ditunjukkan dalam Gambar 5.40.



Gambar 5.38. Tampilan Count SMS Berjumlah 1

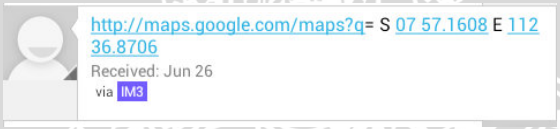


Gambar 5.39. Tampilan Read SMS dengan Isi SMS pada Index 1



Gambar 5.40. Posisi di Google *Map* (Bundaran UB)

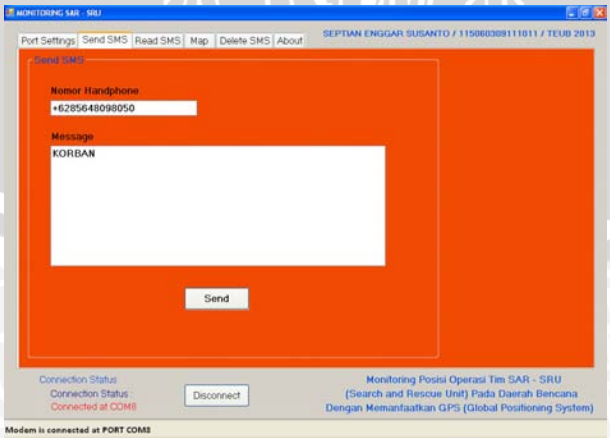
Data posisi juga dikirim ke *handphone* sistem tim SAR pemantau. *Screenshots handphone* data posisi (Bundaran UB) ditunjukkan dalam Gambar 5.41.



Gambar 5.41. *Screenshots Handphone* Data Posisi (Bundaran UB)

➢ **Pengujian Permintaan Jumlah Korban**

Melakukan pengiriman SMS untuk permintaan jumlah korban ke sistem tim SAR – SRU dengan cara masuk ke menu **Send SMS**. Mengisi **Nomor Handphone** sistem tim SAR – SRU, mengetik isi SMS dengan "**KORBAN**" dan menekan tombol **Send** untuk mengirim SMS. Pengiriman SMS untuk permintaan jumlah korban (Bundaran UB) ditunjukkan dalam Gambar 5.42.



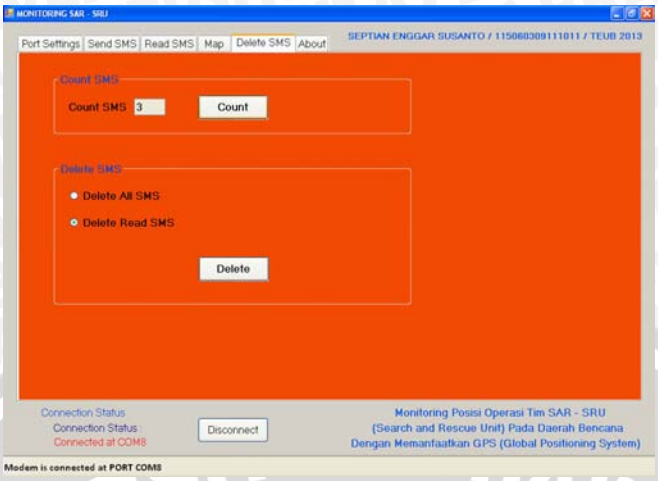
Gambar 5.42. Pengiriman SMS Untuk Permintaan Jumlah Korban (Bundaran UB)

Pada sistem tim SAR – SRU jika ada permintaan korban maka akan menyuruh untuk mengaktifkan menu *keypad* untuk memasukkan jumlah korban yang ditemukan oleh tim SAR – SRU. Perintah Aktifkan menu *keypad* (Bundaran UB) ditunjukkan dalam Gambar 5.43 (a). Pada pengujian ini, menekan tombol 5 pada *keypad* dan di LCD terlihat bahwa jumlah korban adalah 5. Jumlah Korban berjumlah 5 (Bundaran UB) ditunjukkan dalam Gambar 5.43 (b).

(a) (b)

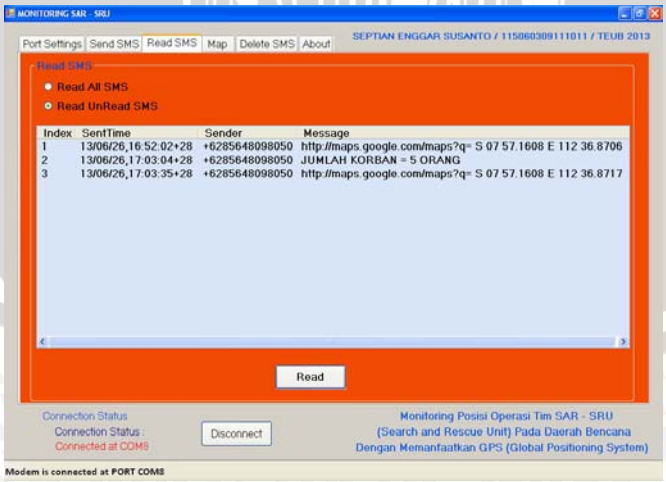
Gambar 5.43. (a) Perintah Aktifkan Menu Keypad (Bundaran UB)
(b) Jumlah Korban Berjumlah 5 (Bundaran UB)

Setelah memasukkan jumlah korban pada *keypad*, data jumlah korban akan dikirimkan ke sistem tim SAR pemantau bersama dengan data posisi. Selanjutnya mengecek SMS masuk pada perangkat lunak dengan menekan tombol **Count**. Jika **Count SMS** berjumlah 3 berarti ada 2 SMS masuk yaitu SMS data jumlah korban dan SMS data posisi. Tampilan **Count SMS** berjumlah 3 (Bundaran UB) ditunjukkan dalam Gambar 5.44.

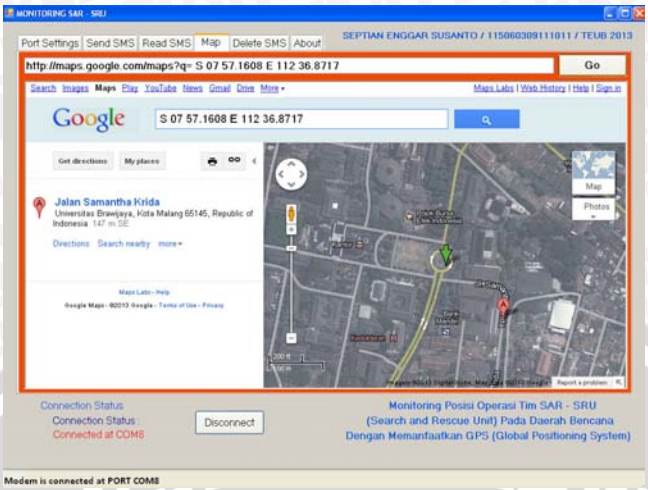


Gambar 5.44. Tampilan Count SMS Berjumlah 3 (Bundaran UB)

Melihat isi SMS dengan cara masuk ke menu **Read SMS** dan menekan tombol **Read**. Isi SMS berupa data jumlah korban dan data posisi akan terlihat pada *listview*. Tampilan Read SMS dengan isi SMS pada index 2 dan 3 ditunjukkan dalam Gambar 5.45. Untuk melihat posisi dengan cara masuk menu **Map** dan menekan tombol **Go**. Komputer dengan terhubung internet akan menampilkan posisi sistem tim SAR – SRU pada Google *Map*. Posisi ditunjukkan oleh anak panah yang berwarna hijau. Posisi tersebut berbeda sekitar 5 meter dengan posisi sebenarnya tempat pengujian. Perbedaan tersebut masih termasuk dalam batas range keakuratan GPS *receiver*. Posisi di Google *Map* (Bundaran UB) ditunjukkan dalam Gambar 5.46.

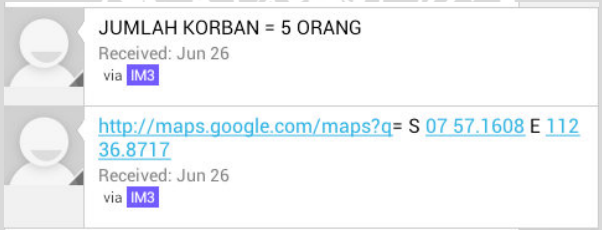


Gambar 5.45. Tampilan Read SMS dengan Isi SMS pada Index 2 dan 3



Gambar 5.46. Posisi di Google Map (Bundaran UB)

Data jumlah korban dan data posisi juga dikirim ke *handphone* sistem tim SAR pemantau. *Screenshots handphone* data jumlah korban dan data posisi (Bundaran UB) ditunjukkan dalam Gambar 5.47.



Gambar 5.47. *Screenshots Handphone* Data Jumlah Korban dan Data Posisi (Bundaran UB)

- **Pengujian Perhitungan Jarak Dan Sudut**

Pengujian perhitungan jarak dan sudut dengan cara mengaktifkan menu **Sudut Jarak** pada sistem tim SAR – SRU. Ketika menu **Sudut Jarak** diaktifkan akan muncul tampilan SUDUT DAN JARAK KE CHECKPOINT. Tampilan SUDUT DAN JARAK KE CHECKPOINT (Bundaran UB) ditunjukkan dalam Gambar 5.48.

Gambar 5.48. Tampilan SUDUT DAN JARAK KE CHECKPOINT (Bundaran UB)

Menu ini berfungsi untuk mengukur jarak dan sudut antara titik pengujian (Bundaran Universitas Brawijaya) dengan *checkpoint* (posisi tim SAR pemantau). Hal ini dapat digunakan seandainya tim SAR – SRU kesasar dan ingin kembali ke tempat *checkpoint* sehingga bisa memperkirakan jarak dan sudut yang harus di tempuh menuju titik asal (*checkpoint*). Perhitungan jarak dan sudut diawali dengan pengambilan data posisi oleh GPS *receiver*. Setelah data posisi valid, mikrokontroler akan melakukan proses perhitungan jarak dan sudut dengan dua data koordinat yaitu koordinat *checkpoint* yang sudah diambil diawal pada waktu berangkat yang disimpan pada EEPROM dan koordinat tempat pengujian. Hasil jarak dan sudut (Bundaran UB) ditunjukkan dalam Gambar 5.49.

Gambar 5.49. Hasil Jarak dan Sudut (Bundaran UB)

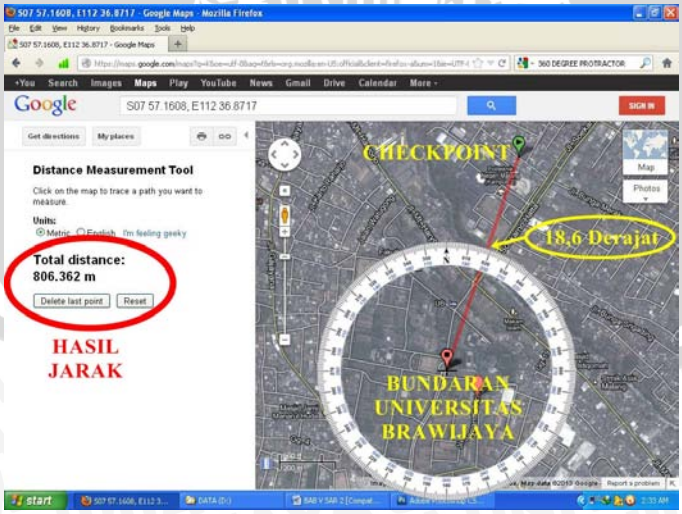
Hasil dari alat tim SAR – SRU menunjukkan **jarak = 819,98632 meter** dan **sudut = 18,68822** derajat terhadap arah utara. Hasil tersebut akan dibandingkan dengan pembacaan jarak pada menu *Distance Measurement Tool Google Map* dan pembacaan sudut menggunakan busur. Pembacaan jarak pada menu *Distance Measurement Tool Google Map* dan pembacaan sudut menggunakan busur ditunjukkan dalam Gambar 5.50.

Pembacaan jarak pada menu Google *Map* menunjukkan **jarak = 806,362 meter**. Pembacaan sudut menggunakan busur menunjukkan **sudut = 18.6 derajat**. *Error* jarak dan *error* sudut dapat dihitung dengan cara sebagai berikut.

$$Error\ jarak = \frac{806,362 - 819,98632}{806,362} \times 100\% = -1,69\%$$

$$Error\ sudut = \frac{18,6 - 18,68822}{18,6} \times 100\% = 0,474\%$$

Error jarak dikarenakan menu Google *Map* mungkin mempunyai algoritma sendiri dalam menghitung jarak sehingga dapat berbeda dengan algoritma yang dibuat untuk menghitung pada alat tim SAR – SRU. Untuk itu perbedaan tersebut dapat menjadi toleransi karena kalibrasi disini berfungsi untuk perbandingan saja. *Error* sudut dikarenakan kepresisian untuk melihat titik yang ditunjukkan pada busur sehingga bisa dikatakan bahwa perhitungan sudut pada alat tim SAR –SRU telah bekerja dengan baik.



Gambar 5.50. Pembacaan Jarak pada Menu *Distance Measurement Tool Google Map* dan Pembacaan Sudut Menggunakan Busur

5.7.2 Pengujian di Museum Brawijaya (Jalan Ijen)

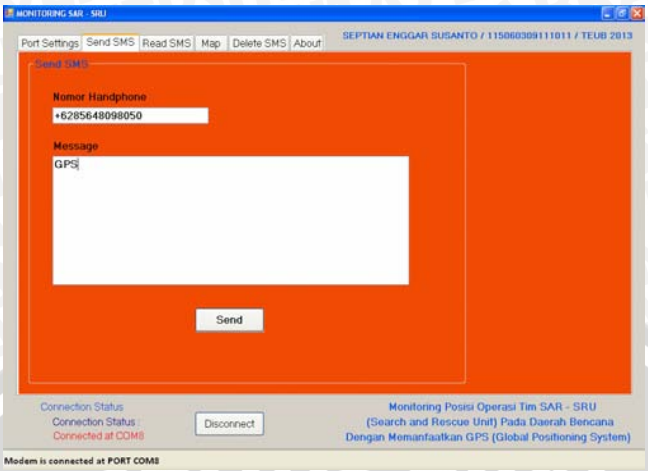
Pengujian kedua dilakukan di Museum Brawijaya. Dalam pengujian ini akan mencoba respon permintaan posisi, permintaan jumlah korban oleh sistem tim SAR pemantau (komputer dan handphone) serta pengukuran jarak dan sudut. Pengujian di Museum Brawijaya ditunjukkan dalam Gambar 5.51.

Gambar 5.51. Pengujian di Museum Brawijaya

- **Pengujian Permintaan Posisi**

Seperti pengujian di Bundaran UB, melakukan pengiriman SMS untuk permintaan posisi ke sistem tim SAR –SRU dengan cara masuk ke menu **Send SMS**. Mengisi **Nomor Handphone** sistem tim SAR – SRU, mengetik isi SMS dengan "**GPS**" dan menekan tombol **Send** untuk mengirim SMS. Pengiriman SMS untuk permintaan posisi (Museum Brawijaya) ditunjukkan dalam Gambar 5.52.

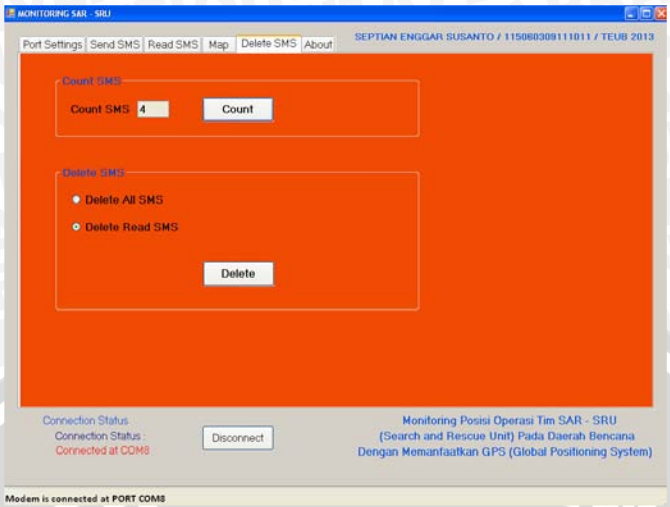
Pada sistem tim SAR – SRU di atur pada menu **Terima SMS** untuk siap menerima perintah dari sistem tim SAR pemantau. Proses Menunggu SMS (Museum Brawijaya) ditunjukkan dalam Gambar 5.53. Setelah SMS permintaan posisi telah diterima, maka GPS akan aktif dan mencari koordinat posisinya. Jika koordinat posisi sudah valid selanjutnya data posisi akan dikirim ke sistem tim SAR pemantau (Komputer dan *Handphone*).



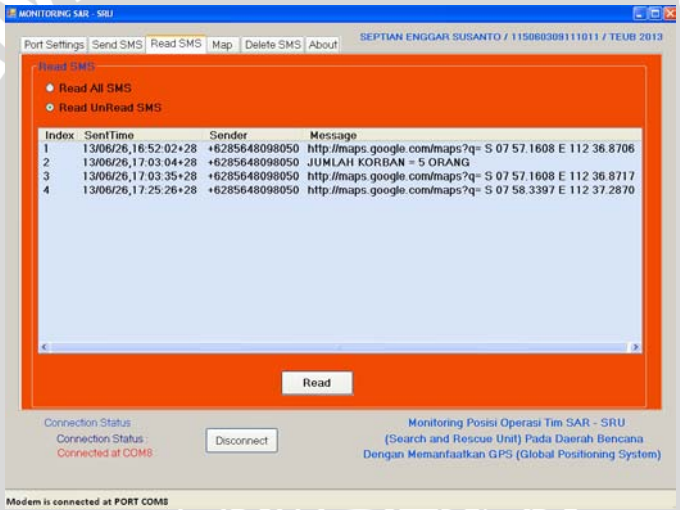
Gambar 5.52. Pengiriman SMS Untuk Permintaan Posisi (Museum Brawijaya)

Gambar 5.53. Proses Menunggu SMS (Museum Brawijaya)

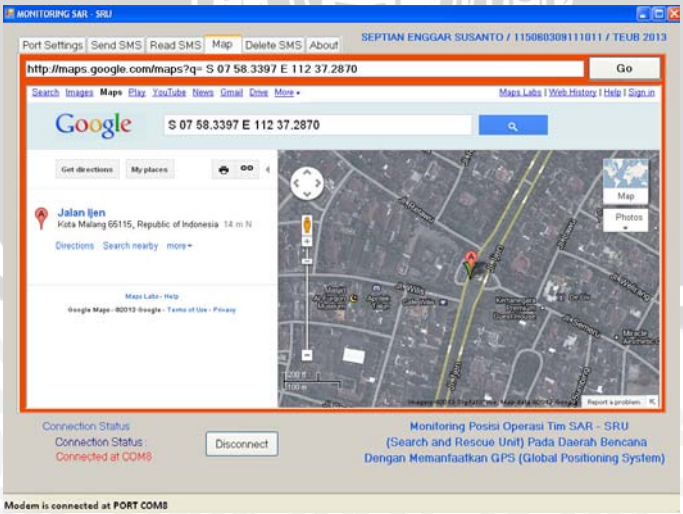
Mengecek SMS masuk pada perangkat lunak dengan menekan tombol **Count**. Jika **Count SMS** berjumlah 4 berarti ada 1 SMS masuk. Tampilan Count SMS berjumlah 4 ditunjukkan dalam Gambar 5.54. Masuk ke menu **Read SMS** dan menekan tombol **Read**. Isi SMS berupa data posisi akan terlihat pada *listview*. Tampilan Read SMS dengan isi SMS pada index 4 ditunjukkan dalam Gambar 5.55. Masuk ke menu **Map** dan menekan tombol **Go**. Posisi ditunjukkan oleh anak panah yang berwarna hijau. Posisi tersebut berbeda sekitar 7 meter dengan posisi sebenarnya tempat pengujian. Perbedaan tersebut masih termasuk dalam batas range keakuratan GPS *receiver*. Posisi di Google *Map* (Museum Brawijaya) ditunjukkan dalam Gambar 5.56.



Gambar 5.54. Tampilan Count SMS berjumlah 4

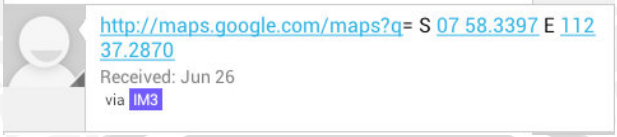


Gambar 5.55. Tampilan Read SMS dengan Isi SMS pada Index 4



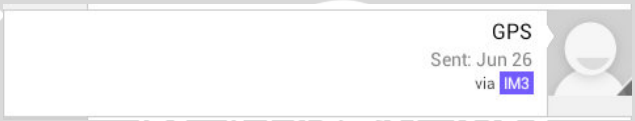
Gambar 5.56. Posisi di Google Map (Museum Brawijaya)

Data posisi juga dikirim ke *handphone* sistem tim SAR pemantau. *Screenshots handphone* data posisi (Museum Brawijaya) ditunjukkan dalam Gambar 5.57.



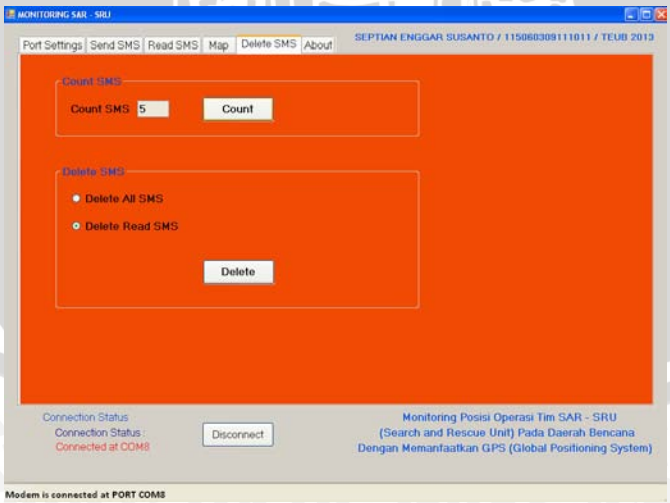
Gambar 5.57. *Screenshots Handphone* Data Posisi (Museum Brawijaya)

Selanjutnya juga mencoba mengirim perintah ke sistem tim SAR – SRU dengan menggunakan *handphone* untuk permintaan posisi yaitu mengirim SMS dengan isi SMS "**GPS**". *Screenshots* Isi SMS "**GPS**" (Museum Brawijaya) ditunjukkan dalam Gambar 5.58.

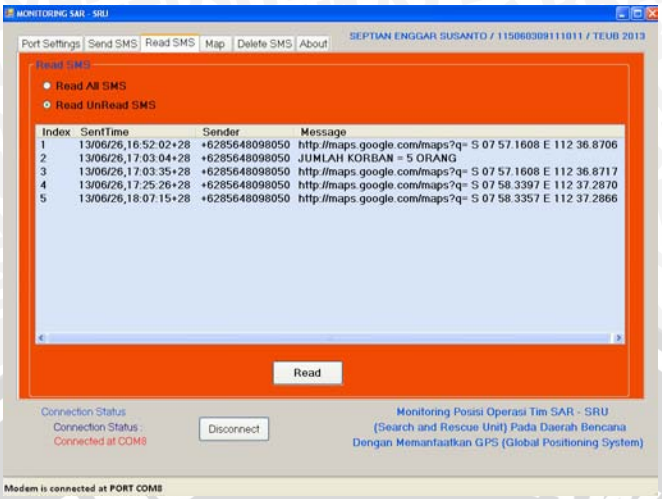


Gambar 5.58. *Screenshots* Isi SMS "GPS" (Museum Brawijaya)

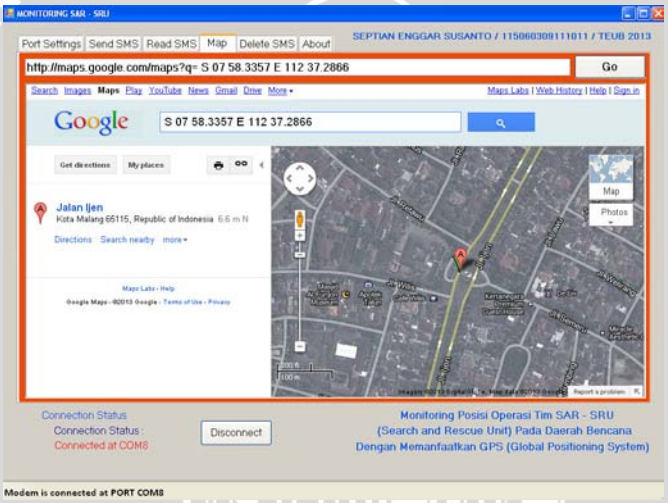
Sistem tim SAR – SRU akan mengirim data posisi ke sistem tim SAR pemantau. **Count SMS** akan bertambah menjadi 5. Tampilan Count SMS berjumlah 5 ditunjukkan dalam Gambar 5.59. Tampilan Read SMS dengan isi SMS pada index 5 ditunjukkan dalam Gambar 5.60. Posisi di Google *Map* (Museum Brawijaya) ditunjukkan dalam Gambar 5.61.



Gambar 5.59. Tampilan Count SMS adalah 5

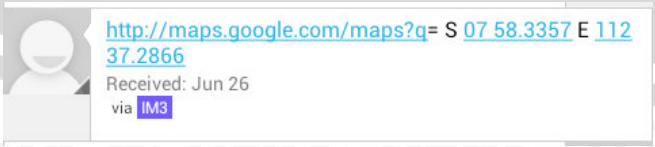


Gambar 5.60. Tampilan Read SMS dengan Isi SMS pada Index 5



Gambar 5.61. Posisi di Google *Map* (Museum Brawijaya)

Data posisi juga dikirim ke *handphone* sistem tim SAR pemantau. *Screenshots handphone* data posisi (Museum Brawijaya) ditunjukkan dalam Gambar 5.62.



Gambar 5.62. *Screenshots Handphone* Data Posisi (Museum Brawijaya)

➢ **Pengujian Permintaan Jumlah Korban**

Melakukan pengiriman SMS untuk permintaan jumlah korban ke sistem tim SAR – SRU dengan cara mengirim SMS dengan isi SMS "**KORBAN**". Isi SMS "KORBAN" (Museum Brawijaya) ditunjukkan dalam Gambar 5.63.

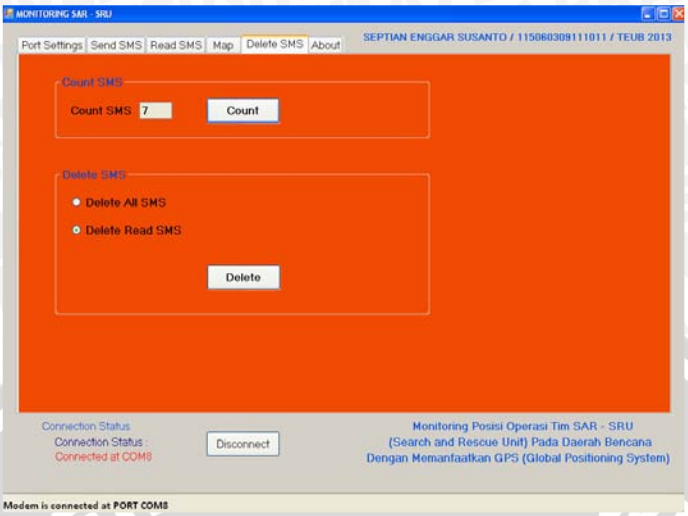
Gambar 5.63. Isi SMS "KORBAN" (Museum Brawijaya)

Pada sistem tim SAR – SRU jika ada permintaan korban maka akan menyuruh untuk mengaktifkan menu *keypad* untuk memasukkan jumlah korban yang ditemukan oleh tim SAR – SRU. Menu **Input Keypad** (Museum Brawijaya) ditunjukkan dalam Gambar 5.64 (a). Pada pengujian ini, menekan tombol 3 pada *keypad* dan di LCD terlihat bahwa jumlah korban adalah 3. Jumlah Korban berjumlah 3 ditunjukkan dalam Gambar 5.64 (b).

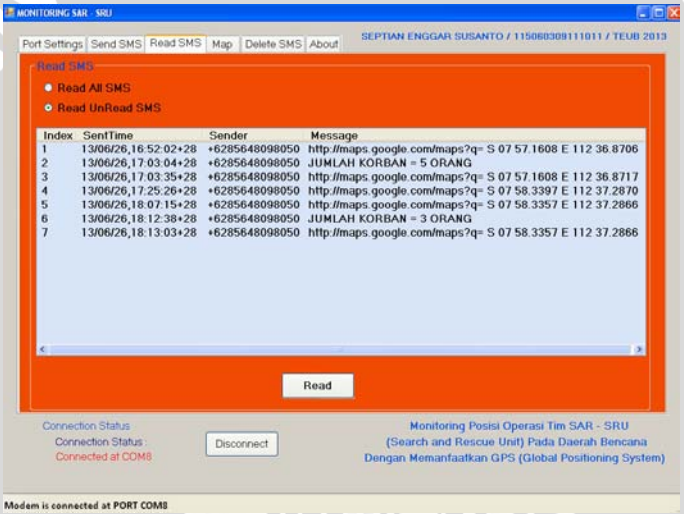
(a) (b)

Gambar 5.64. (a) Menu Input Keypad (Museum Brawijaya)
(b) Jumlah Korban Berjumlah 3

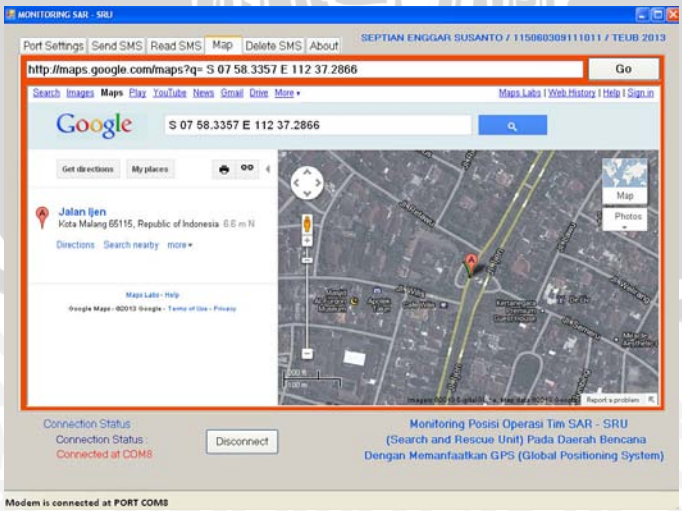
Setelah memasukkan jumlah korban pada *keypad*, data jumlah korban akan dikirimkan ke sistem tim SAR pemantau bersama dengan data posisi. Tampilan **Count SMS** berjumlah 7 ditunjukkan dalam Gambar 5.65. Tampilan Read SMS dengan isi SMS pada index 6 dan 7 ditunjukkan dalam Gambar 5.66. Posisi di Google *Map* (Museum Brawijaya) ditunjukkan dalam Gambar 5.67.



Gambar 5.65. Tampilan Count SMS adalah 7

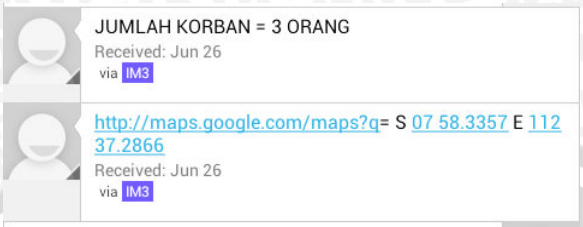


Gambar 5.66. Tampilan read SMS dengan Isi SMS pada Index 6 dan 7



Gambar 5.67. Posisi di Google *Map* (Museum Brawijaya)

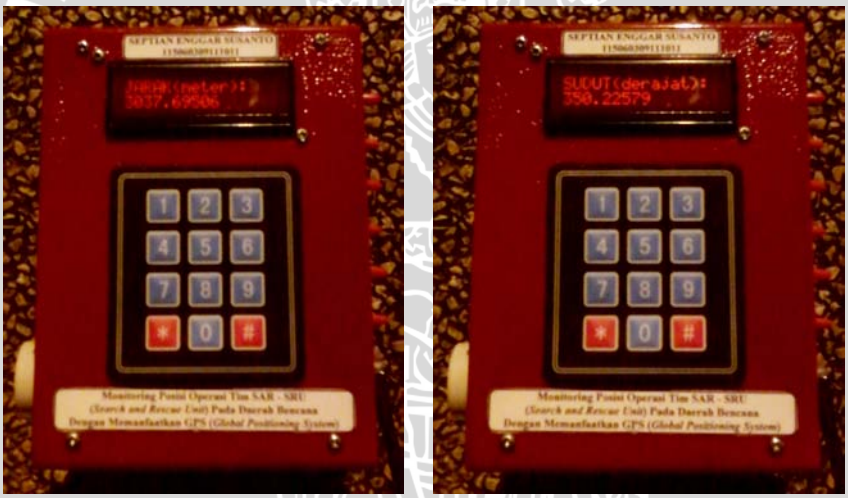
Data jumlah korban dan data posisi juga dikirim ke *handphone* sistem tim SAR pemantau. *Screenshots handphone* data jumlah korban dan data posisi (Museum Brawijaya) ditunjukkan dalam Gambar 5.68.



Gambar 5.68. *Screenshots Handphone* Data Jumlah Korban dan Data Posisi (Museum Brawijaya)

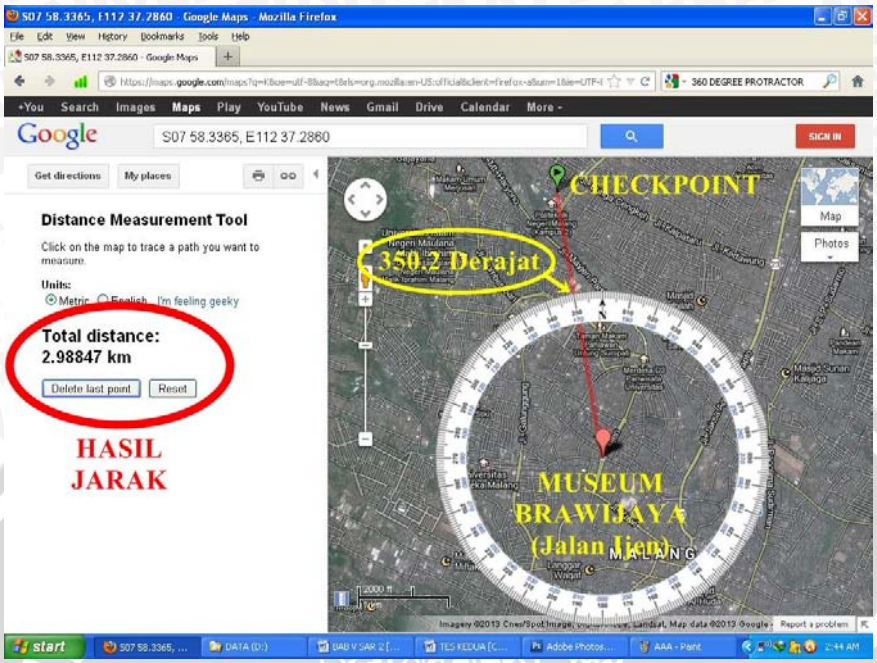
➢ **Pengujian Perhitungan Jarak Dan Sudut**

Pengujian perhitungan jarak dan sudut seperti pengujian jarak dan sudut di Bundaran Universitas Brawijaya. Hasil jarak dan sudut (Museum Brawijaya) ditunjukkan dalam Gambar 5.69.



Gambar 5.69. Hasil Jarak dan Sudut (Museum Brawijaya)

Hasil dari alat tim SAR – SRU menunjukkan **jarak = 3037,69506 meter** dan **sudut = 350,22579 derajat** terhadap arah utara. Seperti sebelumnya, Hasil tersebut akan dibandingkan dengan pembacaan jarak pada menu *Distance Measurement Tool Google Map* dan pembacaan sudut menggunakan busur. Pembacaan jarak pada menu *Distance Measurement Tool Google Map* dan pembacaan sudut menggunakan busur (Museum Brawijaya) ditunjukkan dalam Gambar 5.70.



Gambar 5.70. Pembacaan Jarak Pada Menu Distance Measurement Tool Google *Map* dan Pembacaan Sudut Menggunakan Busur (Museum Brawijaya)

Pembacaan jarak pada menu Google *Map* menunjukkan **jarak = 2,98847 kilometer (2988,47 meter)**. Pembacaan sudut menggunakan busur menunjukkan **sudut = 350,2 derajat**. *Error* jarak dan *error* sudut dapat dihitung dengan cara sebagai berikut.

$$Error\ jarak = \frac{2988,47 - 3037,69506}{2988,47} \times 100\% = 1,65\%$$

$$Error\ sudut = \frac{350,2 - 350,22579}{350,2} \times 100\% = 0,007\%$$

Error jarak dikarenakan menu Google *Map* mungkin mempunyai algoritma sendiri dalam menghitung jarak sehingga dapat berbeda dengan algoritma yang dibuat untuk menghitung pada alat tim SAR – SRU. Untuk itu perbedaan tersebut dapat menjadi toleransi karena kalibrasi disini berfungsi untuk perbandingan saja. *Error* sudut dikarenakan kepresisian untuk melihat titik yang ditunjukkan pada busur sehingga bisa dikatakan bahwa perhitungan sudut pada alat tim SAR – SRU telah bekerja dengan baik.

5.7.3 Pengujian di Tugu Kapal (Jalan Soekarno-Hatta)

Pengujian ketiga dilakukan di Tugu Kapal. Dalam pengujian ini akan mencoba respon permintaan posisi, pengiriman jumlah korban secara langsung kepada sistem tim SAR pemantau serta pengukuran jarak dan sudut. Pengujian di Tugu Kapal ditunjukkan dalam Gambar 5.71.

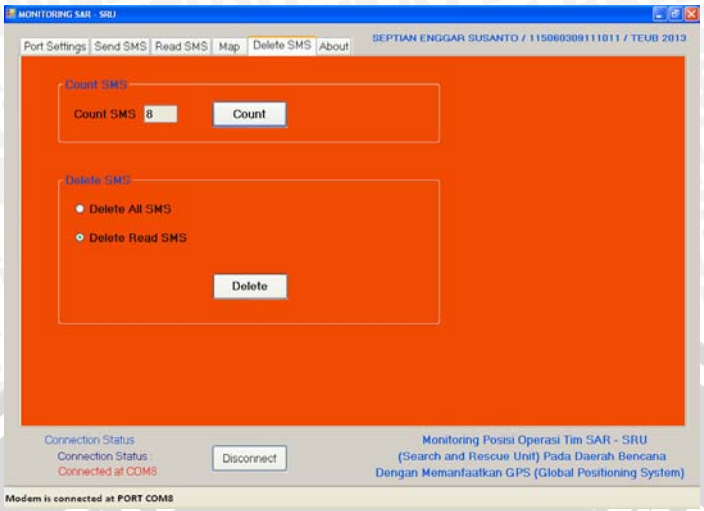
Gambar 5.71. Pengujian di Tugu Kapal

- **Pengujian Permintaan Posisi**

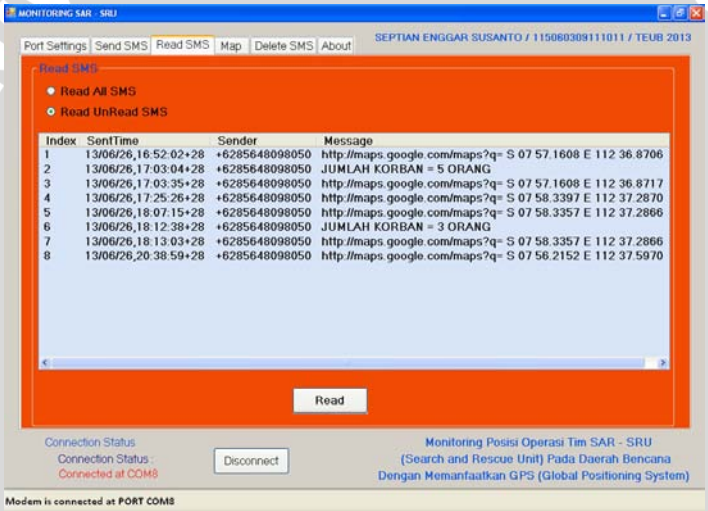
Mengirim perintah ke sistem tim SAR – SRU dengan *handphone* untuk permintaan posisi yaitu mengirim SMS dengan isi SMS "**GPS**". Isi SMS "**GPS**" (Tugu Kapal) ditunjukkan dalam Gambar 5.72.

Gambar 5.72. *Screenshots* Isi SMS "GPS" (Tugu Kapal)

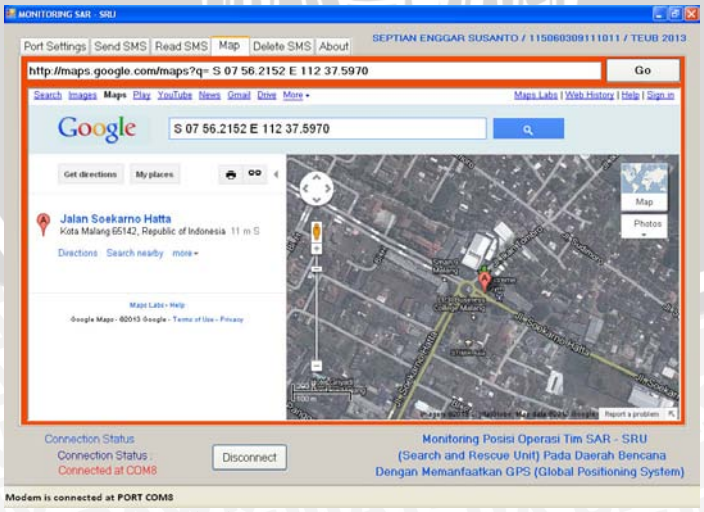
Sistem tim SAR – SRU akan mengirim data posisi ke sistem tim SAR pemantau. **Count SMS** akan bertambah menjadi 8. Tampilan Count SMS berjumlah 8 ditunjukkan dalam Gambar 5.73. Tampilan Read SMS dengan isi SMS pada index 8 ditunjukkan dalam Gambar 5.74. Posisi di Google *Map* (Tugu Kapal) ditunjukkan dalam Gambar 5.75.



Gambar 5.73. Tampilan Count SMS Berjumlah 8

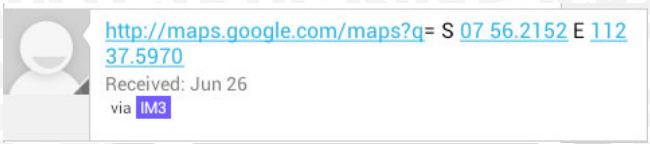


Gambar 5.74. Tampilan Read SMS dengan Isi SMS pada Index 8



Gambar 5.75. Posisi di Google *Map* (Tugu Kapal)

Data posisi juga dikirim ke *handphone* sistem tim SAR pemantau. *Screenshots handphone* data posisi (Tugu Kapal) ditunjukkan dalam Gambar 5.76.



Gambar 5.76. *Screenshots Handphone* Data Posisi (Tugu Kapal)

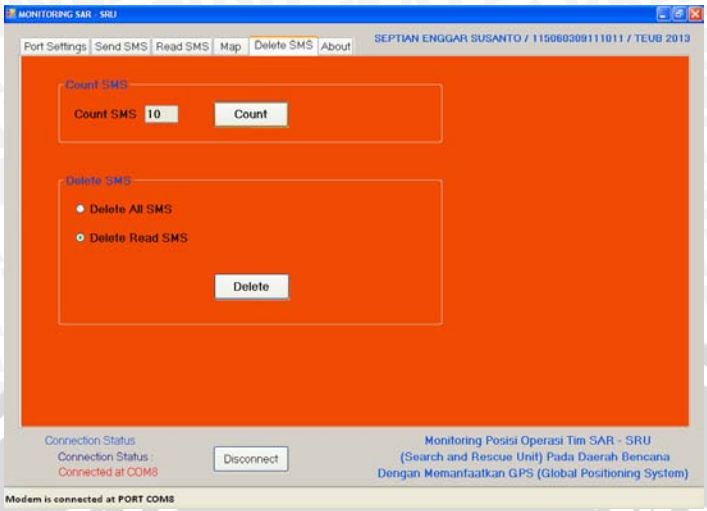
➢ **Pengujian Pengiriman Jumlah Korban Secara Langsung**

Pada pengujian ini akan dilakukan pengiriman jumlah korban secara langsung kepada sistem tim SAR pemantau tanpa dikirimi perintah terlebih dahulu. Hal ini berfungsi seandainya tim SAR – SRU menemukan korban dan ingin langsung mengirimkannya ke tim SAR pemantau.

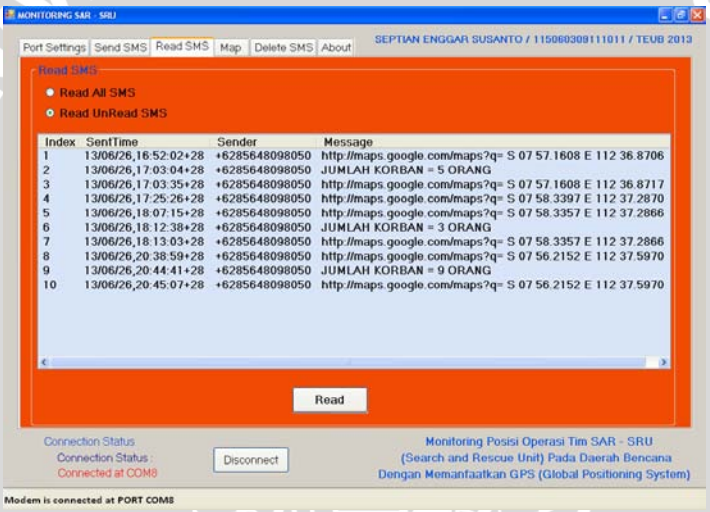
Prosedur pertama yang dilakukan oleh tim SAR – SRU adalah mengaktifkan menu **Input Keypad** pada alat untuk memasukkan jumlah korban yang ditemukan oleh tim SAR – SRU. Pada pengujian ini, menekan tombol 9 pada *keypad* dan di LCD terlihat bahwa jumlah korban adalah 9. Jumlah Korban berjumlah 9 ditunjukkan dalam Gambar 5.77.

Gambar 5.77. Jumlah Korban Berjumlah 9

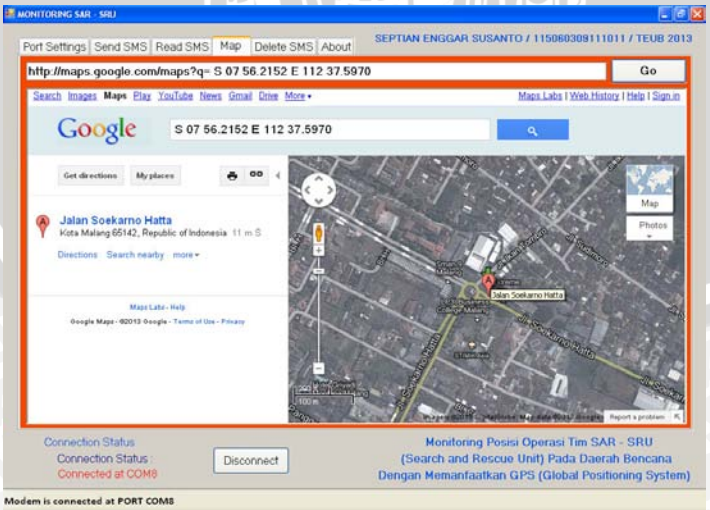
Setelah memasukkan jumlah korban pada *keypad*, sistem tim SAR - SRU akan mencari koordinat posisi melalui GPS *receiver*. Data jumlah korban akan dikirimkan ke sistem tim SAR pemantau bersama dengan data posisi. Tampilan Count SMS berjumlah 10 ditunjukkan dalam Gambar 5.78. Tampilan Read SMS dengan isi SMS pada index 9 dan 10 ditunjukkan dalam Gambar 5.79. Posisi di Google *Map* (Tugu Kapal) ditunjukkan dalam Gambar 5.80.



Gambar 5.78. Tampilan Count SMS Berjumlah 10

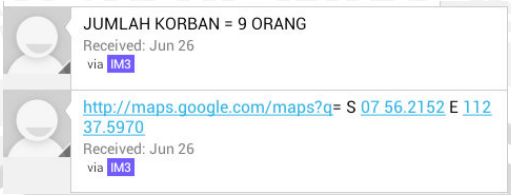


Gambar 5.79. Tampilan Read SMS dengan Isi SMS pada Index 9 dan 10



Gambar 5.80. Posisi di Google *Map* (Tugu Kapal)

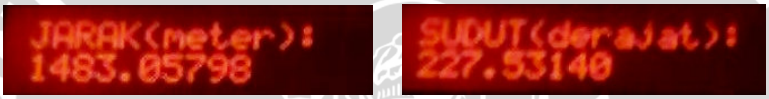
Data jumlah korban dan data posisi juga dikirim ke *handphone* sistem tim SAR pemantau. *Screenshots handphone* data jumlah korban dan data posisi (Tugu Kapal) ditunjukkan dalam Gambar 5.81.



Gambar 5.81. *Screenshots Handphone* Data Jumlah Korban dan Data Posisi (Tugu Kapal)

➢ **Pengujian Perhitungan Jarak Dan Sudut**

Pengujian perhitungan jarak dan sudut seperti pengujian jarak dan sudut sebelumnya. Hasil jarak dan sudut (Tugu Kapal) ditunjukkan dalam Gambar 5.82.



Gambar 5.82. Hasil Sudut dan Jarak (Tugu Kapal)

Hasil dari alat tim SAR – SRU menunjukkan **jarak = 1483,05798 meter** dan **sudut = 227,53140 derajat** terhadap arah utara. Seperti sebelumnya, Hasil tersebut akan dibandingkan dengan pembacaan jarak pada menu *Distance Measurement Tool Google Map* dan pembacaan sudut menggunakan busur. Pembacaan jarak pada menu *Distance Measurement Tool Google Map* dan pembacaan sudut menggunakan busur (Tugu Kapal) ditunjukkan dalam Gambar 5.83.

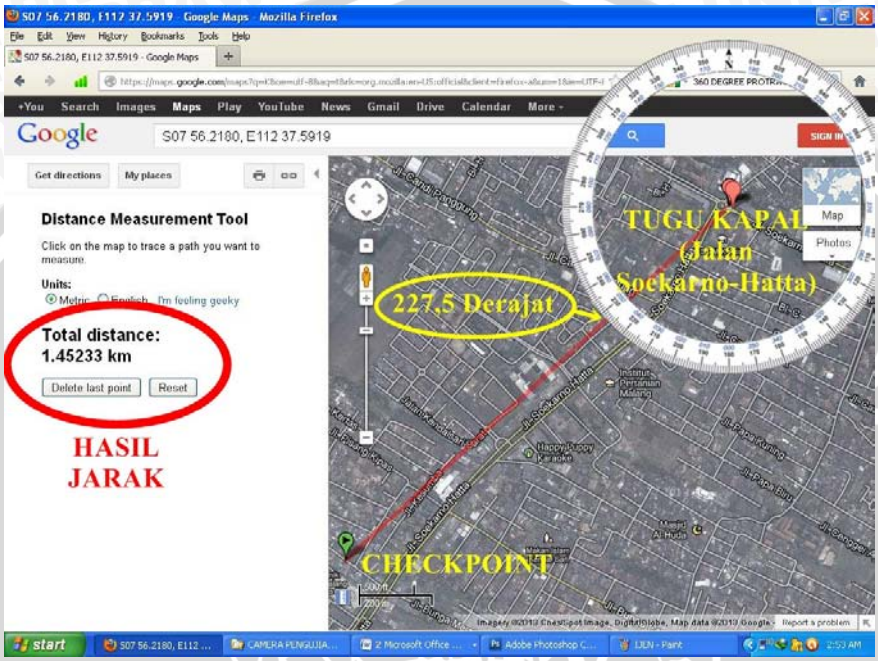
Pembacaan jarak pada menu Google *Map* menunjukkan **jarak = 1,45233 kilometer (1452,33 meter)**. Pembacaan sudut menggunakan busur menunjukkan **sudut = 227,5 derajat**. *Error* jarak dan *error* sudut dapat dihitung dengan cara sebagai berikut.

$$Error\ jarak = \frac{1452.33 - 1483.05798}{1452.33} \times 100\%$$
$$= 2.12\%$$

$$Error\ sudut = \frac{227.5 - 227.5314}{227.5} \times 100\%$$
$$= 0.014\%$$

Error jarak dikarenakan menu Google *Map* mungkin mempunyai algoritma sendiri dalam menghitung jarak sehingga dapat berbeda dengan algoritma yang

dibuat untuk menghitung pada alat tim SAR – SRU. Untuk itu perbedaan tersebut dapat menjadi toleransi karena kalibrasi disini berfungsi untuk perbandingan saja. *Error* sudut dikarenakan kepresisian untuk melihat titik yang ditunjukkan pada busur sehingga bisa dikatakan bahwa perhitungan sudut pada alat tim SAR – SRU telah bekerja dengan baik.



Gambar 5.83. Pembacaan Jarak Pada Menu *Distance Measurement Tool Google Map* dan Pembacaan Sudut Menggunakan Busur (Tugu Kapal)

5.7.4 Data Pengujian

➢ **Pengujian Waktu Respons SMS**

Pengujian waktu respons SMS ini adalah untuk menguji lama waktu yang dibutuhkan ketika sistem tim SAR - SRU diberi perintah untuk penentuan posisi sampai data posisi GPS valid diterima oleh sistem tim SAR pemantau. Data hasil pengujian waktu respons SMS ditunjukkan dalam Tabel 5.4.

Tabel 5.4. Data Hasil Pengujian Waktu Respons SMS

No	Tempat Pengujian	Waktu (detik)
1	Bundaran UB	48
2	Museum Brawijaya	51
3	Tugu Kapal	58
4	Rumah Sakit UB	56
5	Taman Makam Pahlawan	48
6	Dekanat FTUB	45
7	POLINEMA	47
Waktu rata-rata		**50,43**

Dari data hasil pengujian menunjukkan bahwa waktu rata-rata respons SMS ketika sistem tim SAR - SRU diberi perintah untuk penentuan posisi sampai data posisi diterima oleh sistem tim SAR pemantau adalah **50,43 detik**.

➢ **Pengujian Akurasi Data GPS**

Pengujian akurasi data GPS ini adalah untuk menguji perbedaan posisi koordinat yang diberikan oleh GPS *receiver* dengan posisi tempat sebenarnya. Data hasil pengujian akurasi data GPS ditunjukkan dalam Tabel 5.5.

Tabel 5.5. Data Hasil Pengujian Akurasi Data GPS

No	Tempat Pengujian	Pembacaan (meter)
1	Bundaran UB	5,0
2	Museum Brawijaya	7,0
3	Tugu Kapal	4,8
4	Rumah Sakit UB	9,1
5	Taman Makam Pahlawan	9,0
6	Dekanat FTUB	15,1
7	POLINEMA	3,9
Akurasi rata-rata		**7,7**

Untuk mengukur perbedaan posisi koordinat yang diberikan oleh GPS *receiver* dengan posisi tempat sebenarnya dengan cara menggunakan metode pembacaan jarak pada menu *Google Map* (*Distance Measurement Tool).*

Gambar 5.84 (a) menunjukkan pembacaan jarak di Bundaran UB. Gambar 5.84 (b) menunjukkan pembacaan jarak di Museum Brawijaya. Gambar 5.84 (c) menunjukkan pembacaan jarak di Tugu Kapal. Gambar 5.84 (d) menunjukkan pembacaan jarak di Rumah Sakit UB. Gambar 5.84 (e) menunjukkan pembacaan jarak di Taman Makam Pahlawan. Gambar 5.84 (f) menunjukkan pembacaan jarak di Dekanat FTUB. Gambar 5.84 (g) menunjukkan pembacaan jarak di POLINEMA.

Gambar 5.84. (a) Pembacaan Jarak di Bundaran UB (b) Pembacaan Jarak di Museum Brawijaya
(c) Pembacaan Jarak di Tugu Kapal (d) Pembacaan Jarak di Rumah Sakit UB
(e) Pembacaan Jarak di Taman Makam Pahlawan (f) Pembacaan Jarak di Dekanat FTUB
(g) Pembacaan Jarak di POLINEMA

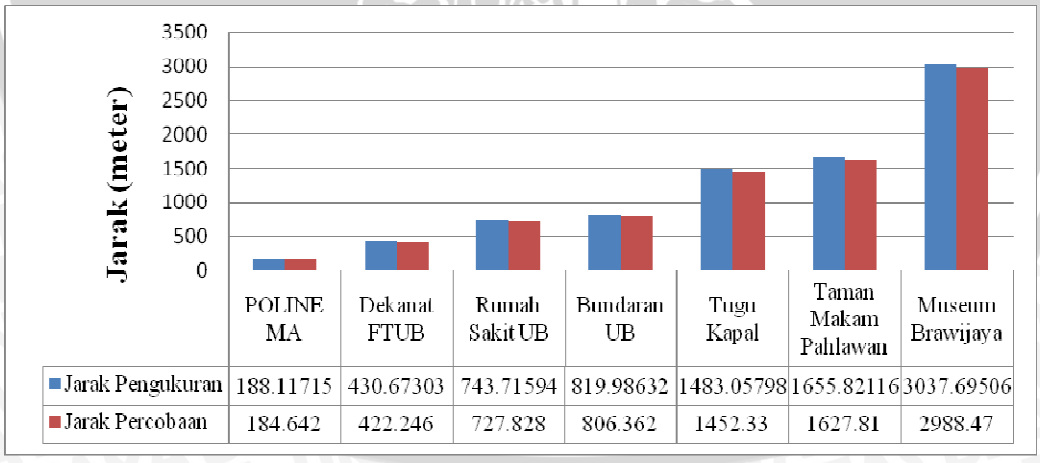
Dari data hasil pengujian menunjukkan bahwa akurasi rata-rata antara koordinat posisi yang diberikan oleh GPS *receiver* dengan posisi tempat sebenarnya adalah **7,7 meter**.

➢ **Pengujian Jarak tim SAR - SRU terhadap *Checkpoint***

Pengujian ini adalah untuk menguji jarak tim SAR - SRU terhadap *checkpoint* (titik awal). Koordinat *checkpoint* adalah 07˚56',7489 LS dan 112˚37',0113 BT. Data hasil pengujian jarak ditunjukkan dalam Tabel 5.6. Grafik hasil pengujian jarak tim SAR - SRU terhadap *checkpoint* ditunjukkan dalam Gambar 5.85.

Tabel 5.6. Data Hasil Pengujian Jarak

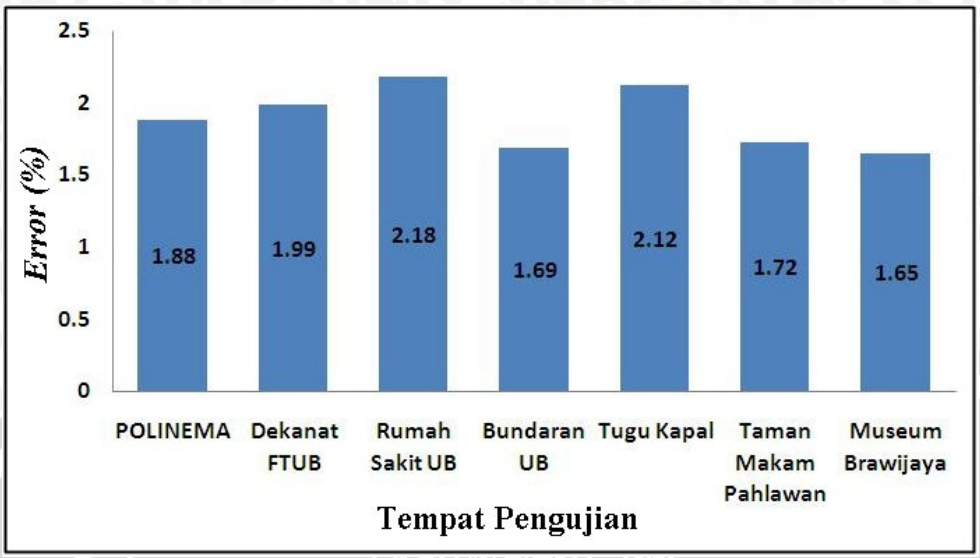
No	Tempat Pengujian	Jarak Pengukuran(meter)	Jarak Percobaan(meter)	*Error*(%)
1	POLINEMA	188.11715	184.642	1.88
2	Dekanat FTUB	430.67303	422.246	1.99
3	Rumah Sakit UB	743.71594	727.828	2.18
4	Bundaran UB	819.98632	806.362	1.69
5	Tugu Kapal	1483.05798	1452.33	2.12
6	Taman Makam Pahlawan	1655.82116	1627.81	1.72
7	Museum Brawijaya	3037.69506	2988.47	1.65
		***Error* rata-rata**		**1.89**



Gambar 5.85. Grafik Hasil Pengujian Jarak Tim SAR - SRU terhadap *Checkpoint*

Jarak pengukuran adalah hasil pengukuran jarak pada alat tim SAR - SRU. Jarak percobaan adalah hasil percobaan dengan menggunakan menu *Google Map* (*Distance Measurement Tool)* untuk menghitung jarak.

Gambar 5.86 menunujukkan grafik *presentase error* pengukuran jarak tim SAR - SRU terhadap *checkpoint.*



Gambar 5.86. Grafik *Presentase Error* Pengukuran Jarak Tim SAR - SRU Terhadap *Checkpoint*

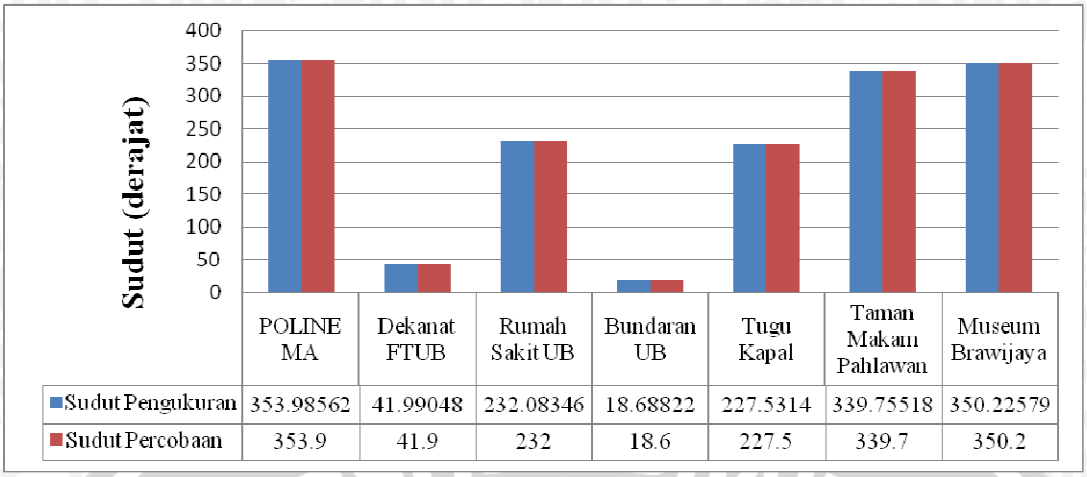
Presentase error rata-rata pengukuran jarak tim SAR - SRU terhadap *checkpoint* adalah **1.89%**.

- **Pengujian Arah Sudut tim SAR - SRU Menuju *Checkpoint***

Pengujian ini adalah untuk menguji arah sudut tim SAR - SRU menuju *checkpoint* (titik awal). Koordinat *checkpoint* adalah 07˚56',7489 LS dan 112˚37',0113 BT. Data hasil pengujian arah sudut ditunjukkan dalam Tabel 5.7. Grafik hasil pengujian arah sudut tim SAR - SRU menuju *checkpoint* ditunjukkan dalam Gambar 5.87.

Tabel 5.7. Data Hasil Pengujian Arah Sudut

No	Tempat Pengujian	Sudut Pengukuran(derajat)	Sudut Percobaan(derajat)	*Error*(%)
1	POLINEMA	353.98562	353.9	0.024
2	Dekanat FTUB	41.99048	41.9	0.216
3	Rumah Sakit UB	232.08346	232	0.036
4	Bundaran UB	18.68822	18.6	0.474
5	Tugu Kapal	227.5314	227.5	0.014
6	Taman Makam Pahlawan	339.75518	339.7	0.016
7	Museum Brawijaya	350.22579	350.2	0.007
		***Error* rata-rata**		**0.11**

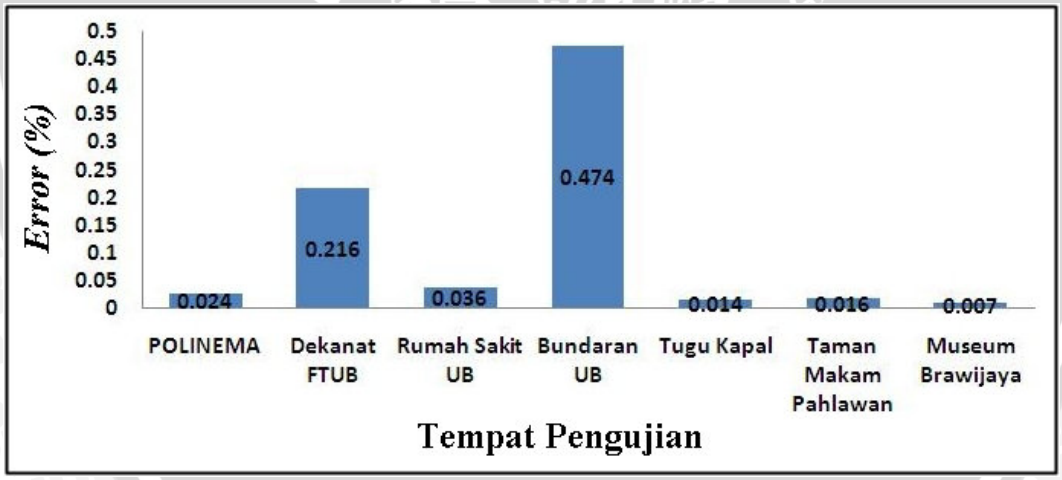


	POLINEMA	Dekanat FTUB	Rumah Sakit UB	Bundaran UB	Tugu Kapal	Taman Makam Pahlawan	Museum Brawijaya
Sudut Pengukuran	353.98562	41.99048	232.08346	18.68822	227.5314	339.75518	350.22579
Sudut Percobaan	353.9	41.9	232	18.6	227.5	339.7	350.2

Gambar 5.87. Grafik Hasil Pengujian Arah Sudut Tim SAR - SRU Menuju *Checkpoint*

Sudut pengukuran adalah hasil pengukuran sudut pada alat tim SAR - SRU. Sudut percobaan adalah hasil percobaan dengan menggunakan busur untuk menghitung sudut.

Gambar 5.88 menunujukkan grafik *presentase error* pengukuran arah sudut tim SAR - SRU menuju *checkpoint.*



Gambar 5.88. Grafik *Presentase Error* Pengukuran Arah Sudut Tim SAR - SRU Menuju *Checkpoint*

Presentase error rata-rata pengukuran arah sudut tim SAR - SRU menuju *checkpoint* adalah **0.11%**.

➢ **Pengujian Format Data GPS**

Pengujian ini adalah untuk menguji beberapa format data yang dapat dikeluarkan oleh GPS *receiver*. GPS *Receiver* yang digunakan adalah SkyNav SKM53 GPS *module*. Pada *datasheet* SkyNav SKM53 GPS *module* dijelaskan bahwa ada 7 format data GPS yang bisa dikeluarkan. Akan tetapi setelah diuji hanya beberapa format data GPS yang bisa dikeluarkan. Data hasil pengujian format data ditunjukkan dalam Tabel 5.8.

Tabel 5.8. Data Hasil Pengujian Format Data

Format Data	Uji Coba	Data
GPGGA	**Mengeluarkan Data**	**Koordinat**
GPGLL	**Tidak Mengeluarkan Data**	**-**
GPGSA	**Mengeluarkan Data**	**Bukan Koordinat**
GPGSV	**Mengeluarkan Data**	**Bukan Koordinat**
GPRMC	**Mengeluarkan Data**	**Koordinat**
GPVTG	**Tidak Mengeluarkan Data**	**-**
GPZDA	**Tidak Mengeluarkan Data**	**-**

Dari data hasil pengujian format data diatas hanya 4 format data yang dapat dikeluarklan oleh GPS *receiver* yaitu GPGGA, GPGSA, GPGSV dan GPRMC. Pada alat ini dibutuhkan data koordinat, sedangkan format data GPS yang mengeluarkan data koordinat hanya 2 format data yaitu GPGGA dan GPRMC. Untuk itu, akan diuji kedua format data tersebut untuk mengetahui tingkat akurasi data koordinat yang dikeluarkan. Pengujian ini dilakukan di tiga tempat yaitu di Bundaran UB, di Dekanat FTUB dan di Tugu Kapal. Data hasil pengujian format data GPGGA ditunjukkan dalam Tabel 5.9. Data hasil pengujian format data GPRMC ditunjukkan dalam Tabel 5.10.

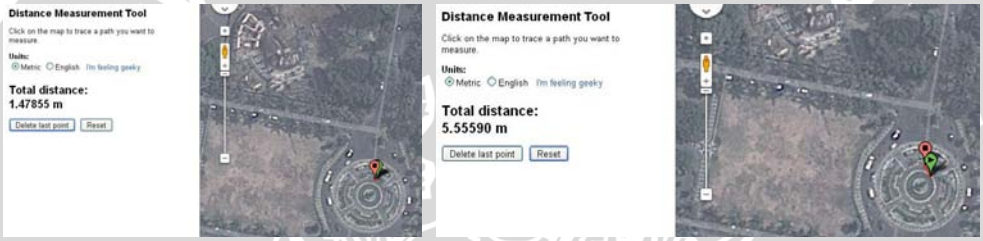
Tabel 5.9. Data Hasil Pengujian Format Data GPGGA

No	Tempat Pengujian	Koordinat GPGGA	Pembacaan Jarak (meter)
1	Bundaran UB	07° 57',1504 112° 36',8698	1.47855
2	Dekanat FTUB	07° 56',9188 112° 36',8492	2.38412
3	Tugu Kapal	07° 56',2181 112° 37',5970	2.07005

Tabel 5.10. Data Hasil Pengujian Format Data GPRMC

No	Tempat Pengujian	Koordinat GPRMC	Pembacaan Jarak (meter)
1	Bundaran UB	07° 57',1527 112° 36',8700	5.55590
2	Dekanat FTUB	07° 56',9201 112° 36',8484	4.36616
3	Tugu Kapal	07° 56',2192 112° 37',5983	2.92750

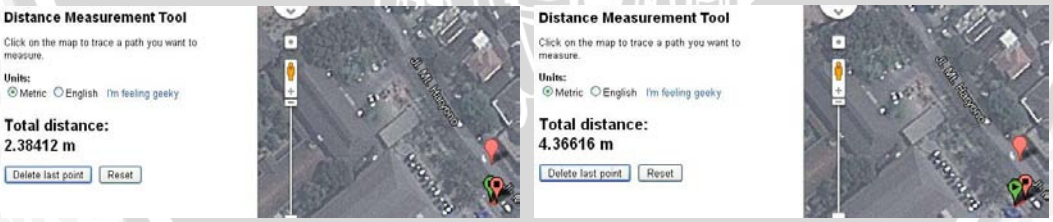
Gambar 5.89 (a) menunjukkan pembacaan jarak format data GPGGA di Bundaran UB. Gambar 5.89 (b) menunjukkan pembacaan jarak format data GPRMC di Bundaran UB.



(a) (b)

Gambar 5.89. (a) Pembacaan Jarak Format Data GPGGA di Bundaran UB
(b) Pembacaan Jarak Format Data GPRMC di Bundaran UB

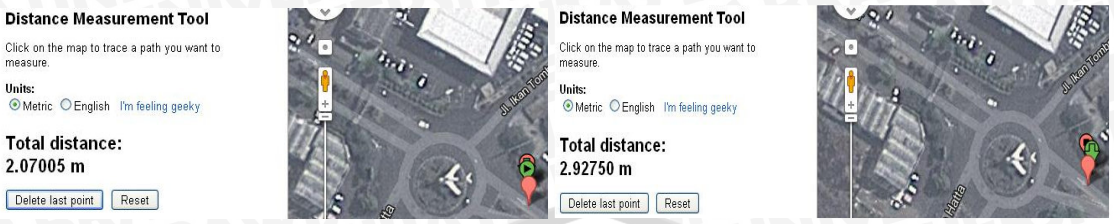
Gambar 5.90 (a) menunjukkan pembacaan jarak format data GPGGA di Dekanat FTUB. Gambar 5.90 (b) menunjukkan pembacaan jarak format data GPRMC di Dekanat FTUB.



(a) (b)

Gambar 5.90. (a) Pembacaan Jarak Format Data GPGGA di Dekanat FTUB
(b) Pembacaan Jarak Format Data GPRMC di Dekanat FTUB

Gambar 5.91 (a) menunjukkan pembacaan jarak format data GPGGA di Tugu Kapal. Gambar 5.91 (b) menunjukkan pembacaan jarak format data GPRMC di Tugu Kapal.



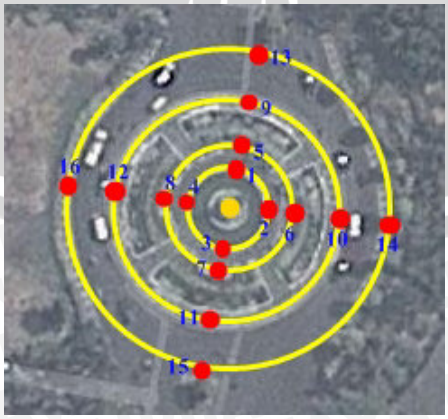
(a) (b)

Gambar 5.91. (a) Pembacaan Jarak Format Data GPGGA di Tugu Kapal
(b) Pembacaan Jarak Format Data GPRMC di Tugu Kapal

Dari hasil data pengujian kedua format data GPS diatas menunjukkan bahwa format data GPGGA memiliki nilai pembacaan jarak lebih kecil dari pada format data GPRMC. Pembacaan jarak tersebut adalah jarak antara koordinat yang diberikan GPS dengan posisi tempat sebenarnya. Hal ini berarti format data GPGGA lebih akurat daripada format data GPRMC.

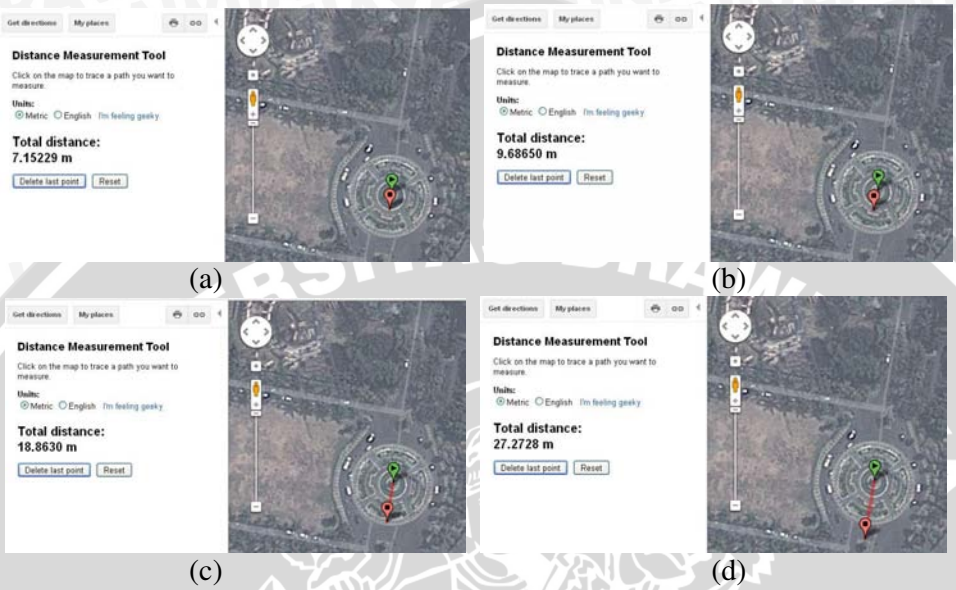
- **Metode Pengukuran**

Pengujian ini adalah untuk menguji akurasi data GPS dengan 4 radius jarak yang berbeda, masing-masing radius dilakukan pengujian 4 titik koordinat. Pengujian dilakukan di Bundaran UB dengan menggunakan format data GPGGA. Penggunaan format data GPGGA dikarenakan format data GPGGA lebih akurat dibandingkan dengan format data GPRMC. Gambar 5.92 menunjukkan gambaran radius dan titik pengukuran. Radius 1 adalah titik 1,2,3 dan 4 dengan jarak 7,15229 meter dari titik pusat. Radius 2 adalah titik 5,6,7 dan 8 dengan jarak 9,68650 meter dari titik pusat. Radius 3 adalah titik 9,10,11 dan 12 dengan jarak 18,8630 meter dari titik pusat. Radius 4 adalah titik 13,14,15 dan 16 dengan jarak 27,2728 meter dari titik pusat.



Gambar 5.92. Gambaran Radius dan Titik Pengukuran

Gambar 5.93 (a) menunjukkan pembacaan jarak radius 1. Gambar 5.93 (b) menunjukkan pembacaan jarak radius 2. Gambar 5.93 (c) menunjukkan pembacaan jarak radius 3. Gambar 5.93 (d) menunjukkan pembacaan jarak radius 4.



Gambar 5.93. (a) Pembacaan Jarak Radius 1 (b) Pembacaan Jarak Radius 2 (c) Pembacaan Jarak Radius 3 (d) Pembacaan Radius Jarak 4

Setelah masing-masing radius jarak dan titik pengujian ditentukan maka selanjutnya adalah melakukan pengukuran jarak dengan menggunakan sistem alat tim SAR - SRU. Tabel 5.11 menunjukkan data hasil pengukuran jarak radius 1. Tabel 5.12 menunjukkan data hasil pengukuran jarak radius 2. Tabel 5.13 menunjukkan data hasil pengukuran jarak radius 3. Tabel 5.14 menunjukkan data hasil pengukuran jarak radius 4.

Tabel 5.11. Data Hasil Pengukuran Jarak Radius 1

Tempat Pengujian	Hasil Koordinat GPS	Pengukuran Jarak (meter)
Titik 1	07° 57',1545 112° 36',8711	6.04
Titik 2	07° 57',1564 112° 36',8724	8.30
Titik 3	07° 57',1590 112° 36',8670	8.03
Titik 4	07° 57',1547 112° 36',8641	7.74
Pengukuran Jarak rata-rata		**7.53**

Tabel 5.12. Data Hasil Pengukuran Jarak Radius 2

Tempat Pengujian	Hasil Koordinat GPS	Pengukuran Jarak (meter)
Titik 5	07° 57',1505 112° 36',8703	9.18
Titik 6	07° 57',1567 112° 36',8727	9.29
Titik 7	07° 57',1599 112° 36',8649	11.29
Titik 8	07° 57',1565 112° 36',8635	9.16
Pengukuran Jarak rata-rata		**9.73**

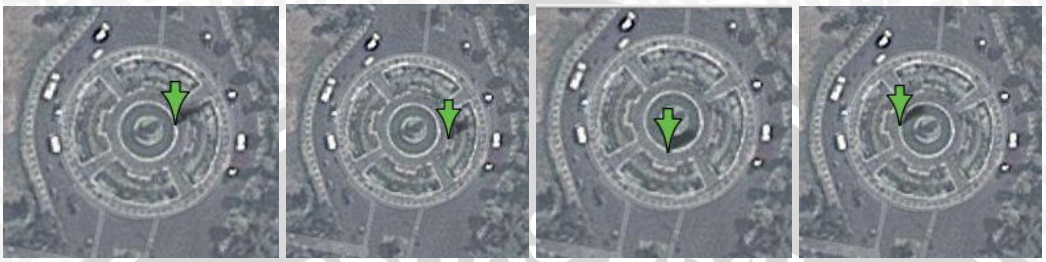
Tabel 5.13. Data Hasil Pengukuran Jarak Radius 3

Tempat Pengujian	Hasil Koordinat GPS	Pengukuran Jarak (meter)
Titik 9	07° 57',1485 112° 36',8737	15.70
Titik 10	07° 57',1579 112° 36',8769	17.30
Titik 11	07° 57',1657 112° 36',8692	20.50
Titik 12	07° 57',1559 112° 36',8590	17.30
Pengukuran Jarak rata-rata		**17.70**

Tabel 5.14. Data Hasil Pengukuran Jarak Radius 4

Tempat Pengujian	Hasil Koordinat GPS	Pengukuran Jarak (meter)
Titik 13	07° 57',1418 112° 36',8737	26.50
Titik 14	07° 57',1614 112° 36',8794	24.79
Titik 15	07° 57',1695 112° 36',8642	28.40
Titik 16	07° 57',1553 112° 36',8509	31.80
Pengukuran Jarak rata-rata		**27.87**

Data koordinat yang dihasilkan GPS dapat dilihat pada peta Google *Map*. Gambar 5.94 (a) menunjukkan koordinat titik 1 radius 1. Gambar 5.94 (b) menunjukkan koordinat titik 2 radius 1. Gambar 5.94 (c) menunjukkan koordinat titik 3 radius 1. Gambar 5.94 (d) menunjukkan koordinat titik 4 radius 1.



(a) (b) (c) (d)

Gambar 5.94. (a) Koordinat Titik 1 Radius 1 (b) Koordinat Titik 2 Radius 1 (c) Koordinat Titik 3 Radius 1 (d) Koordinat Titik 4 Radius 1

Gambar 5.95 (a) menunjukkan koordinat titik 5 radius 2. Gambar 5. 95 (b) menunjukkan koordinat titik 6 radius 2. Gambar 5. 95 (c) menunjukkan koordinat titik 7 radius 2. Gambar 5. 95 (d) menunjukkan koordinat titik 8 radius 2.

(a) (b) (c) (d)

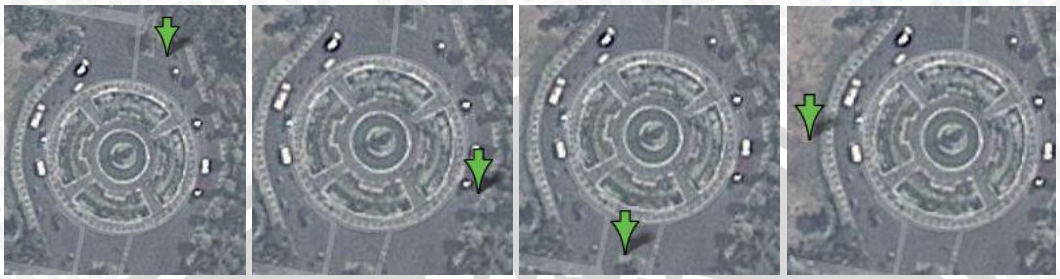
Gambar 5.95. (a) Koordinat Titik 5 Radius 2 (b) Koordinat Titik 6 Radius 2 (c) Koordinat Titik 7 Radius 2 (d) Koordinat Titik 8 Radius 2

Gambar 5.96 (a) menunjukkan koordinat titik 9 radius 3. Gambar 5.96 (b) menunjukkan koordinat titik 10 radius 3. Gambar 5.96 (c) menunjukkan koordinat titik 11 radius 3. Gambar 5.96 (d) menunjukkan koordinat titik 12 radius 3.

(a) (b) (c) (d)

Gambar 5.96. (a) Koordinat Titik 9 Radius 3 (b) Koordinat Titik 10 Radius 3 (c) Koordinat Titik 11 Radius 3 (d) Koordinat Titik 12 Radius 3

Gambar 5.97 (a) menunjukkan koordinat titik 13 radius 4. Gambar 5.97 (b) menunjukkan koordinat titik 14 radius 4. Gambar 5.97 (c) menunjukkan koordinat titik 15 radius 4. Gambar 5. 97 (d) menunjukkan koordinat titik 16 radius 4.



(a) (b) (c) (d)

Gambar 5.97. (a) Koordinat Titik 13 Radius 4 (b) Koordinat Titik 14 Radius 4 (c) Koordinat Titik 15 Radius 4 (d) Koordinat Titik 16 Radius 4

Dari data pengukuran melalui alat tim SAR - SRU dan pembacaan pada menu Google *Map* diatas dapat dicari nilai *error* rata-rata jarak pada masing-masing radius.

Pembacaan jarak radius 1 = 7,15229 meter

Pengukuran jarak rata-rata radius 1 = 7,53 meter

$$Error\ rata-rata\ radius\ 1 = \frac{7.15229 - 7.53}{7.15229} \times 100\%$$

$$= 5.281\%$$

Pembacaan jarak radius 2 = 9,68650 meter

Pengukuran jarak rata-rata radius 2 = 9,73 meter

$$Error\ rata-rata\ radius\ 2 = \frac{9.68650 - 9.73}{9.68650} \times 100\%$$

$$= 0.449\%$$

Pembacaan jarak radius 3 = 18,8630 meter

Pengukuran jarak rata-rata radius 3 = 17,70 meter

$$Error\ rata-rata\ radius\ 3 = \frac{18.8630 - 17.70}{18.8630} \times 100\%$$

$$= 6.165\%$$

Pembacaan jarak radius 4 = 27,2728 meter

Pengukuran jarak rata-rata radius 4 = 27,87 meter

$$Error\ rata-rata\ radius\ 4 = \frac{27.2728 - 27.87}{27.2728} \times 100\%$$

$$= 0.026\%$$

Dari keempat *error* rata-rata masing - masing radius tersebut dapat dihitung rata-rata *error* radius sebagai berikut :

$$Error\ rata-rata\ radius = \frac{5,281 + 0,449 + 6,165 + 0,026}{4} = 2,980\%$$

Dari pengujian format data GPS dan pengujian metode pengukuran dapat disimpulkan bahwa format data GPS yang akurat adalah format data **GPGGA**, metode pengukuran jarak dengan beberapa titik koordinat dan jarak radius mempunyai *error* rata-rata sebesar **2,980%**.

BAB VI
KESIMPULAN DAN SARAN

6.1 Kesimpulan

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

1) Mikrokontroler dapat melakukan proses akuisisi data dari GPS *receiver*. Akurasi rata-rata data GPS antara koordinat posisi yang diberikan oleh GPS *receiver* dengan posisi tempat sebenarnya adalah 7,7 meter. Format data GPS yang akurat adalah format data GPGGA.
2) Sistem tim SAR - SRU dapat menerima respons berupa SMS dengan format SMS yang sesuai. Pada saat ada perintah penentuan posisi dengan isi SMS (GPS) maka akan dikirimkan data berupa koordinat posisi berbentuk *link* kepada sistem tim SAR pemantau (Komputer dan *Handphone*). Ketika ada perintah permintaan jumlah korban dengan isi SMS (KORBAN) maka data jumlah korban akan dimasukkan melalui *keypad* oleh tim SAR - SRU dan akan dikirimkan ke tim SAR pemantau (Komputer dan *Handphone*) bersama dengan data koordinat posisinya. Waktu rata-rata respons SMS ketika sistem tim SAR - SRU diberi perintah untuk penentuan posisi sampai data posisi diterima oleh sistem tim SAR pemantau adalah 50,43 detik.
3) Perangkat lunak sistem mikrokontroler bekerja dengan baik dalam mengolah data GPS, *input keypad*, mengolah data SMS pada modem *wavecom* dan menampilkan data di LCD. *Error* rata-rata pengujian jarak tim SAR - SRU dengan *checkpoint* adalah 1,89% dan *Error* rata-rata pengujian arah sudut tim SAR - SRU menuju *checkpoint* adalah 0,11%. *Error* rata-rata pengukuran jarak dengan metode pengukuran menggunakan jarak radius adalah 2,980%.
4) Perangkat lunak pada sistem tim SAR pemantau dapat mengirim perintah dan menerima data berupa SMS dari sistem tim SAR - SRU. Data jumlah korban dapat ditampilkan pada *listview* dan data posisi yang berupa *link* dengan jumlah 53 karakter dapat ditampilkan secara visual pada peta Google *Map*.

6.2 Saran

Beberapa hal yang dapat dikembangkan untuk kesempurnaan alat ini adalah sebagai berikut :

1) Untuk memperoleh tingkat ketelitian penentuan posisi yang lebih tinggi, perlu dipertimbangkan penggunaan GPS *receiver* yang lebih akurat. Disarankan menggunakan format data GPGGA, karena format data GPGGA lebih akurat dalam penentuan koordinat daripada format data GPRMC.
2) Perlu dilakukan penelitian atau pengembangan selanjutnya untuk aplikasi SAR di tempat yang tidak ada sinyal GSM seperti di gunung sehingga pengiriman data dapat menggunakan komunikasi sinyal radio.

DAFTAR PUSTAKA

Abidin, Hasanuddin Z. 2000. *Penentuan Posisi dengan GPS dan Aplikasinya*, CV Pradnya Paramita, Jakarta.

Andi Nalwan, Paulus. 2004. *Penggunaan dan Antarmuka Modul LCD M1632*. Jakarta: PT Elek Media Komputindo Kelompok Gramedi

Atmel. 2009. *ATMega162 Datasheet*. San Jose: Atmel Corporation

Basarnas. 2012. *Pengendalian Operasi.* http://www.basarnas.go.id/index.php/halaman/52/pengendalian-operasi. Diakses pada tanggal 27 Februari 2013

Basarnas. 2012. Sistem Komunikasi SAR. http://www.basarnas.go.id/index.php/halaman/47/sistem-komunikasi-sar. Diakses pada tanggal 27 Februari 2013

BNPB. 2011. *Data Dan Informasi Bencana Indonesia*. http://dibi.bnpb.go.id/ /DesInventar/ dashboard.jsp?countrycode=id&continue=y&lang=ID. Diakses pada tanggal 27 Februari 2013

El-Rabbany, Ahmad. 2002. *Introduction to GPS: The Global Positioning System*. Norwood: Artech House.

Kapplan, Elliott D. 1996. *Understanding GPS Priciples and Application*, Artech House, London.

Sitio, Aprianto. 1997. *Pembuatan Purwarupa (Prototype) Peta Elektronik Untuk Keperluan Transportasi*. Tugas Akhir. Jurusan Teknik Geodesi, Fakultas Teknik Sipil dan Perencanaan. Institut Teknologi Bandung.

Skylab. 2009. *SkyNav SKM53 Series Ultra High Sensitivity and Low Power The Smart Antenna GPS Module Datasheet*. http://www.webtronico.com. Diakses Pada Tanggal 17 April 2013

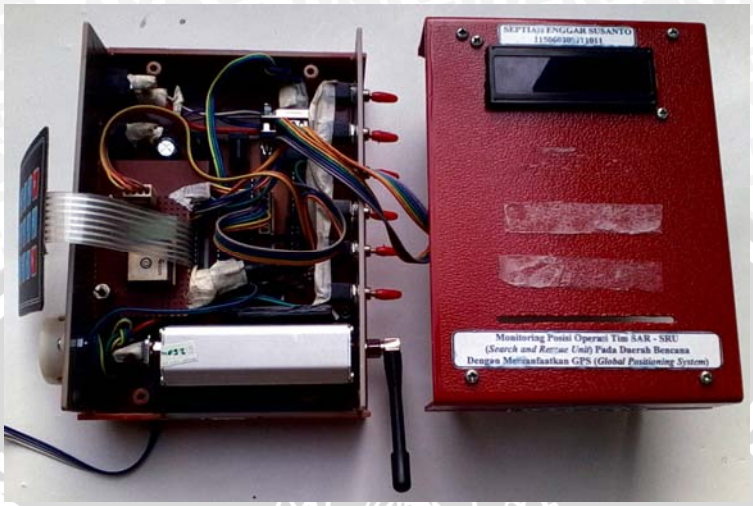
Taryudi. 2009. *Implementasi dan Uji Kerja Sistem Pemantau Posisi dan Tingkat Pencemaran Udara Bergerak, Tesis*. Depok: Universitas Indonesia

Wavecom. 2006. *AT Commands Interface Guide for 6.57 Release Datasheet*. http://www.wavecom.com. Diakses Pada Tanggal 27 Februari 2013

Wavecom. 2004. *Fastrack Modem M1306B User Guide Datasheet*. http://www.wavecom.com. Diakses Pada Tanggal 27 Februari 2013

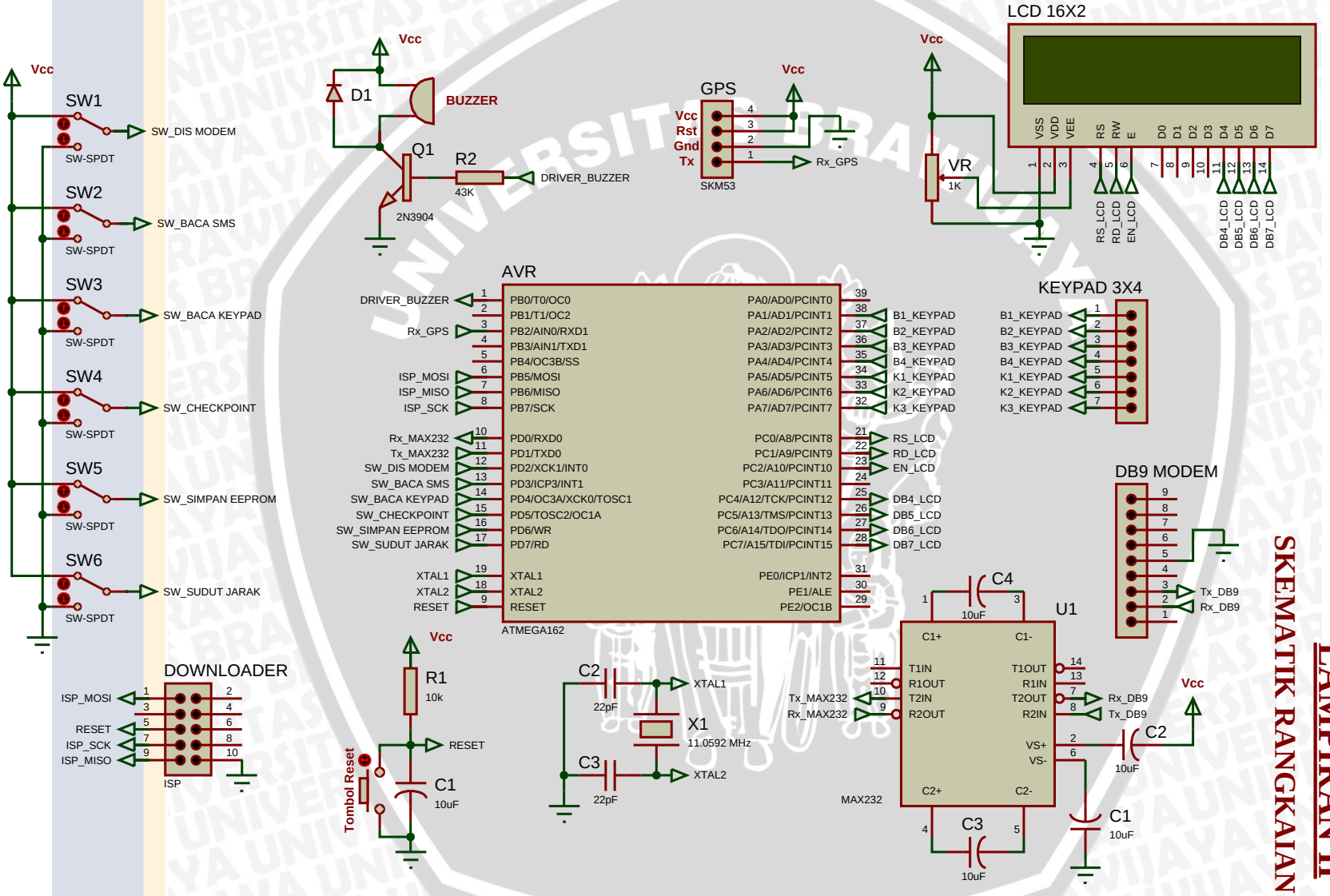
LAMPIRAN I

FOTO ALAT SISTEM TIM SAR - SRU



Gambar 1. Foto Komponen Alat

Gambar 2. Foto Alat Sistem Tim SAR - SRU Keseluruhan


LAMPIRAN II
SKEMATIK RANGKAIAN
Vcc
SW1
SW_DIS MODEM
SW-SPDT
SW2
SW_BACA SMS
SW3
SW_BACA KEYPAD
SW4
SW_CHECKPOINT
SW5
SW_SIMPAN EEPROM
SW6
SW_SUDUT JARAK
D1
BUZZER
Q1
2N3904
R2
43K
DRIVER_BUZZER
GPS
Vcc
Rst
Gnd
Tx
SKM53
Rx_GPS
LCD 16X2
VSS
VDD
VEE
RS
RW
E
D0
D1
D2
D3
D4
D5
D6
D7
VR
1K
RS_LCD
RD_LCD
EN_LCD
DB4_LCD
DB5_LCD
DB6_LCD
DB7_LCD
AVR
PB0/T0/OC0
PB1/T1/OC2
PB2/AIN0/RXD1
PB3/AIN1/TXD1
PB4/OC3B/SS
PB5/MOSI
PB6/MISO
PB7/SCK
PD0/RXD0
PD1/TXD0
PD2/XCK1/INT0
PD3/ICP3/INT1
PD4/OC3A/XCK0/TOSC1
PD5/TOSC2/OC1A
PD6/WR
PD7/RD
XTAL1
XTAL2
RESET
PA0/AD0/PCINT0
PA1/AD1/PCINT1
PA2/AD2/PCINT2
PA3/AD3/PCINT3
PA4/AD4/PCINT4
PA5/AD5/PCINT5
PA6/AD6/PCINT6
PA7/AD7/PCINT7
PC0/A8/PCINT8
PC1/A9/PCINT9
PC2/A10/PCINT10
PC3/A11/PCINT11
PC4/A12/TCK/PCINT12
PC5/A13/TMS/PCINT13
PC6/A14/TDO/PCINT14
PC7/A15/TDI/PCINT15
PE0/ICP1/INT2
PE1/ALE
PE2/OC1B
ATMEGA162
KEYPAD 3X4
B1_KEYPAD
B2_KEYPAD
B3_KEYPAD
B4_KEYPAD
K1_KEYPAD
K2_KEYPAD
K3_KEYPAD
DB9 MODEM
Tx_DB9
Rx_DB9
DOWNLOADER
ISP_MOSI
RESET
ISP_SCK
ISP_MISO
ISP
R1
10k
Tombol Reset
C1
10uF
C2
22pF
C3
22pF
X1
11.0592 MHz
XTAL1
XTAL2
U1
C1+
C1-
T1IN
T1OUT
R1OUT
R1IN
T2IN
T2OUT
R2OUT
R2IN
VS+
VS-
C2+
C2-
MAX232
C4
10uF
C2
10uF
C3
10uF
C1
10uF

LAMPIRAN III

LISTING PROGRAM MIKROKONTROLER ATMEGA162

(*CODEVISION AVR*)

```
/*****************************************
*********
This program was produced by the
CodeWizardAVR V2.03.4 Standard Automatic
Program Generator © Copyright 1998-2008 Pavel
Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project : SKRIPSI SAR
Version : 1.0
Date    : 5/1/2013
Author  : SEPTIAN ENGGAR SUSANTO
Company : SAP2011-TEUB

Chip type           : ATmega162
Clock frequency     : 11.059200 MHz
*****************************************
********/

#include <mega162.h>
#include <stdio.h>
#include <stdlib.h>
#include <lcd.h>
#include <delay.h>
#include <math.h>

#asm
   .equ __lcd_port=0x15 ;PORTC
#endasm

#define RXB8 1
#define TXB8 0
#define UPE 2
#define OVR 3
#define FE 4
#define UDRE 5
#define RXC 7

#define FRAMING_ERROR (1<<FE)
#define PARITY_ERROR (1<<UPE)
#define DATA_OVERRUN (1<<OVR)
#define DATA_REGISTER_EMPTY (1<<UDRE)
#define RX_COMPLETE (1<<RXC)
#define buzzer PORTB.0

// Get a character from the USART1 Receiver
#pragma used+
char getchar1(void)
{
    char status,data;
    while (1)
    {
        while ((status=UCSR1A) & RX_COMPLETE)==0);
        data=UDR1;
        if ((status & (FRAMING_ERROR | PARITY_ERROR | DATA_OVERRUN))==0)
        return data;
    };
}
#pragma used-

// Declare your global variables here
//SMS
unsigned char cek[100];
unsigned int a;

//GPS
char status;
float arah_lintang,arah_bujur,data,lintang[10],bujur[10],kecepatan[6];
char i;

//EEPROM
eeprom float E_lsimpan1, E_lsimpan2, E_lsimpan3, E_lsimpan4, E_lsimpan5, E_lsimpan6, E_lsimpan7, E_lsimpan8;
float lsimpan1, lsimpan2, lsimpan3, lsimpan4, lsimpan5, lsimpan6, lsimpan7, lsimpan8;

eeprom float E_bsimpan1, E_bsimpan2, E_bsimpan3,E_bsimpan4,E_bsimpan5, E_bsimpan6, E_bsimpan7, E_bsimpan8, E_bsimpan9;
float bsimpan1, bsimpan2, bsimpan3, bsimpan4, bsimpan5, bsimpan6, bsimpan7, bsimpan8, bsimpan9;

eeprom float E_lsimpan1_kedua, E_lsimpan2_kedua,E_lsimpan3_kedua,E_lsimpan4_kedua, E_lsimpan5_kedua, E_lsimpan6_kedua, E_lsimpan7_kedua, E_lsimpan8_kedua;
float lsimpan1_kedua, lsimpan2_kedua, lsimpan3_kedua,lsimpan4_kedua,lsimpan5_kedua, lsimpan6_kedua,lsimpan7_kedua,lsimpan8_kedua;

eeprom float E_bsimpan1_kedua, E_bsimpan2_kedua,E_bsimpan3_kedua,E_bsimpan4_kedua, E_bsimpan5_kedua, E_bsimpan6_kedua, E_bsimpan7_kedua,E_bsimpan8_kedua,E_bsimpan9_kedua;
float bsimpan1_kedua, bsimpan2_kedua, bsimpan3_kedua,bsimpan4_kedua,bsimpan5_kedua, bsimpan6_kedua,bsimpan7_kedua,bsimpan8_kedua, bsimpan9_kedua;

eeprom float E_simpankeypad;
float simpankeypad;

//KEYPAD 4 x 3
unsigned char key;

//KONVERSI
float cx,d,f,g,jx,k,m,n;
unsigned long ax,b,c,h,ix,j;

float cx2,d2,f2,g2,jx2,k2,m2,n2;
unsigned long ax2,b2,c2,h2,ix2,j2;

//HITUNG JARAK DAN SUDUT
float p,q,r,s,t,u,v,w;
unsigned char temp[6];

float atan(float x);

//PROTOTYPE FUNGSI
void lintang_sms(void);
void bujur_sms(void);
void tampil_lintang(void);
void tampil_bujur(void);
void tampil_status(void);
void send_sms(char flash *fmtstr1);
void gps(void);
void baca_sms(void);

void tulis_data_ke_eeprom11();
void baca_data_ke_eeprom11();
void tulis_data_ke_eeprom12();
void baca_data_ke_eeprom12();
void tulis_data_ke_eeprom13();
void baca_data_ke_eeprom13();
void tulis_data_ke_eeprom14();
void baca_data_ke_eeprom14();
void tulis_data_ke_eeprom15();
void baca_data_ke_eeprom15();
void tulis_data_ke_eeprom16();
void baca_data_ke_eeprom16();
```

```
void tulis_data_ke_eeproml7();
void baca_data_ke_eeproml7();
void tulis_data_ke_eeproml8();
void baca_data_ke_eeproml8();

void tulis_data_ke_eepromb1();
void baca_data_ke_eepromb1();
void tulis_data_ke_eepromb2();
void baca_data_ke_eepromb2();
void tulis_data_ke_eepromb3();
void baca_data_ke_eepromb3();
void tulis_data_ke_eepromb4();
void baca_data_ke_eepromb4();
void tulis_data_ke_eepromb5();
void baca_data_ke_eepromb5();
void tulis_data_ke_eepromb6();
void baca_data_ke_eepromb6();
void tulis_data_ke_eepromb7();
void baca_data_ke_eepromb7();
void tulis_data_ke_eepromb8();
void baca_data_ke_eepromb8();
void tulis_data_ke_eepromb9();
void baca_data_ke_eepromb9();

void tulis_data_ke_eeproml1_kedua();
void baca_data_ke_eeproml1_kedua();
void tulis_data_ke_eeproml2_kedua();
void baca_data_ke_eeproml2_kedua();
void tulis_data_ke_eeproml3_kedua();
void baca_data_ke_eeproml3_kedua();
void tulis_data_ke_eeproml4_kedua();
void baca_data_ke_eeproml4_kedua();
void tulis_data_ke_eeproml5_kedua();
void baca_data_ke_eeproml5_kedua();
void tulis_data_ke_eeproml6_kedua();
void baca_data_ke_eeproml6_kedua();
void tulis_data_ke_eeproml7_kedua();
void baca_data_ke_eeproml7_kedua();
void tulis_data_ke_eeproml8_kedua();
void baca_data_ke_eeproml8_kedua();

void tulis_data_ke_eepromb1_kedua();
void baca_data_ke_eepromb1_kedua();
void tulis_data_ke_eepromb2_kedua();
void baca_data_ke_eepromb2_kedua();
void tulis_data_ke_eepromb3_kedua();
void baca_data_ke_eepromb3_kedua();
void tulis_data_ke_eepromb4_kedua();
void baca_data_ke_eepromb4_kedua();
void tulis_data_ke_eepromb5_kedua();
void baca_data_ke_eepromb5_kedua();
void tulis_data_ke_eepromb6_kedua();
void baca_data_ke_eepromb6_kedua();
void tulis_data_ke_eepromb7_kedua();
void baca_data_ke_eepromb7_kedua();
void tulis_data_ke_eepromb8_kedua();
void baca_data_ke_eepromb8_kedua();
void tulis_data_ke_eepromb9_kedua();
void baca_data_ke_eepromb9_kedua();

void simpan_gps_eeprom();
void simpan_gps_eeprom_kedua();

void konversi_L_B();
void sudut_jarak();

unsigned char keypad4x3();
void tes_keypad();

void main(void)
{
// Crystal Oscillator division factor: 1
#pragma optsize-
CLKPR=0x80;
CLKPR=0x00;
#ifdef _OPTIMIZE_SIZE_
#pragma optsize+
#endif

PORTA=0xFE;
DDRA=0x1F;

PORTB=0x00;
DDRB=0x03;    //PB.0 [Buzzer], PB.1 [Led]

// USART0 initialization    [SERIAL MODEM]
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART0 Receiver: On
// USART0 Transmitter: On
// USART0 Mode: Asynchronous
// USART0 Baud Rate: 115200
UCSR0A=0x00;
UCSR0B=0x18;
UCSR0C=0x86;
UBRR0H=0x00;
UBRR0L=0x05;

// USART1 initialization    [SERIAL GPS]
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART1 Receiver: On
// USART1 Transmitter: Off
// USART1 Mode: Asynchronous
// USART1 Baud Rate: 9600
UCSR1A=0x00;
UCSR1B=0x10;
UCSR1C=0x86;
UBRR1H=0x00;
UBRR1L=0x47;

// LCD module initialization
lcd_init(16);

//ALLOHU AKBAR
lcd_gotoxy(14,0);
lcd_putchar(0x7C);
lcd_gotoxy(14,1);
lcd_putchar(0x7C);
lcd_gotoxy(13,0);
lcd_putchar(0x7C);
lcd_gotoxy(13,1);
lcd_putchar(0x7C);
lcd_gotoxy(12,1);
lcd_putchar(0x5F);
lcd_gotoxy(11,0);
lcd_putchar(0x7C);
lcd_gotoxy(11,1);
lcd_putchar(0x7C);
lcd_gotoxy(10,1);
lcd_putchar(0x5F);
lcd_gotoxy(9,0);
lcd_putchar(0x34);
lcd_gotoxy(9,1);
lcd_putchar(0x7C);

lcd_gotoxy(6,0);
lcd_putchar(0x7C);
lcd_gotoxy(6,1);
lcd_putchar(0x7C);
lcd_gotoxy(5,1);
lcd_putchar(0x5D);
lcd_gotoxy(4,0);
lcd_putchar(0x3C);
lcd_gotoxy(4,1);
lcd_putchar(0x5F);
lcd_gotoxy(3,1);
lcd_putchar(0x21);
lcd_gotoxy(2,1);
lcd_putchar(0x5F);
lcd_gotoxy(1,1);
lcd_putchar(0x3E);

delay_ms(2000);
lcd_clear();

lcd_gotoxy(0,0);
lcd_putsf("SEPTIAN ENGGAR S");
lcd_gotoxy(0,1);
```

```
    lcd_putsf("115060309111011");
    delay_ms(2000);
    lcd_clear();

    while (1)
    {
        DDRD = 0x00;
        PORTD = 0xFF;

        lcd_gotoxy(0,0);
        lcd_putsf("....MENU SAR....");
        delay_ms(1000);
        lcd_clear();
        lcd_gotoxy(0,1);
        lcd_putsf(">>PILIH SAKLAR<<");
        delay_ms(1000);
        lcd_clear();

        while (PIND.2 == 0)       // PILIHAN
UNTUK DISABLE MODEM WAVECOM (OK)
        {
            lcd_gotoxy(0,0);
            lcd_putsf("MODEM TIDAK BER-");
            lcd_gotoxy(0,1);
            lcd_putsf("KEDIP = OFF");

            printf("AT+CFUN=0");
            putchar(0x0D);
            while(getchar()!='K'){};
            while(getchar()!=0x0A){};

            delay_ms(8000);
            lcd_clear();
        }

        while (PIND.3 == 0)       // PILIHAN
UNTUK TERIMA SMS (OK)
        {
             baca_sms();
        }

        while (PIND.4 == 0)       // PILIHAN
UNTUK INPUT KEYPAD (OK)
        {
            lcd_gotoxy(0,0);
            lcd_putsf("TEKAN KEYPAD!");

            tes_keypad();

            if (key != '@')
            {
                simpankeypad= E_simpankeypad;
                printf("AT+CMGF=1");
                putchar(0x0D);
                while(getchar()!= 'K'){};
                while(getchar()!=0x0A){};
                printf("AT+CMGS=");
                putchar('"');
                printf("085735133736");
                putchar('"');
                putchar(0x0D);
                while(getchar()!=0x3E){};
                delay_ms(10);
                 printf("JUMLAH KORBAN = ");
                putchar(simpankeypad);
                printf(" ORANG");
                putchar(0x1A);
                while(getchar()!='K'){};
                lcd_clear();
                lcd_gotoxy(0,0);
                lcd_putsf("SMS1Terkirim.Ok");
                delay_ms(2000);

                printf("AT+CMGF=1");
                putchar(0x0D);
                while(getchar()!= 'K'){};
                while(getchar()!=0x0A){};
                printf("AT+CMGS=");
                putchar('"');
                printf("085648096965");
                putchar('"');
                putchar(0x0D);
                while(getchar()!=0x3E){};
                delay_ms(10);
                printf("JUMLAH KORBAN = ");
                putchar(simpankeypad);
                printf(" ORANG");
                putchar(0x1A);
                while(getchar()!='K'){};

                lcd_gotoxy(0,1);
               lcd_putsf("SMS2 Terkirim.Ok");
                delay_ms(2000);
                lcd_clear();

                while(status!='A')
                {
                    gps();
                }
                send_sms("085735133736");
                send_sms("085648096965");
                status='V';
            }

          while (PIND.4 == 0 && PIND.2 == 0)
           {
                lcd_gotoxy(0,0);
                 lcd_putsf("ENABLE MODEM..");
                delay_ms(2000);

                printf("AT+CFUN=1");
                putchar(0x0D);
                while(getchar()!='K'){};
                while(getchar()!=0x0A){};

                lcd_clear();
                lcd_gotoxy(0,0);
                lcd_putsf("MODEM OK..");
                delay_ms(5000);
                lcd_clear();
            }
        }

        while (PIND.5 == 0)
        {
          konversi_L_B();
        }

        while (PIND.7 == 0)
        {
            sudut_jarak();
        }
    };


    //***FUNGSI-FUNGSI***//

    void sudut_jarak()
    {
        lcd_gotoxy(0,0);
        lcd_putsf("SUDUT DAN JARAK");
        lcd_gotoxy(0,1);
        lcd_putsf("KE CHECKPOINT...");
        delay_ms(2000);
        lcd_clear();

        while(status!='A')
        {
            gps();
        }

        simpan_gps_eeprom_kedua();

        lcd_gotoxy(0,0);
        lcd_putsf("   DATA KEDUA   ");
        lcd_gotoxy(0,1);
        lcd_putsf("  TERSIMPAN...  ");
        delay_ms(2000);
        lcd_clear();
```

```
baca_data_ke_eeproml1();
baca_data_ke_eeproml2();
baca_data_ke_eeproml3();
baca_data_ke_eeproml4();
baca_data_ke_eeproml5();
baca_data_ke_eeproml6();
baca_data_ke_eeproml7();
baca_data_ke_eeproml8();

baca_data_ke_eepromb1();
baca_data_ke_eepromb2();
baca_data_ke_eepromb3();
baca_data_ke_eepromb4();
baca_data_ke_eepromb5();
baca_data_ke_eepromb6();
baca_data_ke_eepromb7();
baca_data_ke_eepromb8();
baca_data_ke_eepromb9();

baca_data_ke_eeproml1_kedua();
baca_data_ke_eeproml2_kedua();
baca_data_ke_eeproml3_kedua();
baca_data_ke_eeproml4_kedua();
baca_data_ke_eeproml5_kedua();
baca_data_ke_eeproml6_kedua();
baca_data_ke_eeproml7_kedua();
baca_data_ke_eeproml8_kedua();

baca_data_ke_eepromb1_kedua();
baca_data_ke_eepromb2_kedua();
baca_data_ke_eepromb3_kedua();
baca_data_ke_eepromb4_kedua();
baca_data_ke_eepromb5_kedua();
baca_data_ke_eepromb6_kedua();
baca_data_ke_eepromb7_kedua();
baca_data_ke_eepromb8_kedua();
baca_data_ke_eepromb9_kedua();

//TAMPILKAN EEPROM LINTANG KE LCD
lcd_gotoxy(0,0);
lcd_putchar(lsimpan1_kedua);
delay_ms(800);

lcd_gotoxy(1,0);
lcd_putchar(lsimpan2_kedua);
delay_ms(800);

lcd_gotoxy(2,0);
lcd_putchar(lsimpan3_kedua);
delay_ms(800);

lcd_gotoxy(3,0);
lcd_putchar(lsimpan4_kedua);
delay_ms(800);

lcd_gotoxy(4,0);
lcd_putchar(lsimpan5_kedua);
delay_ms(800);

lcd_gotoxy(5,0);
lcd_putchar(lsimpan6_kedua);
delay_ms(800);

lcd_gotoxy(6,0);
lcd_putchar(lsimpan7_kedua);
delay_ms(800);

lcd_gotoxy(7,0);
lcd_putchar(lsimpan8_kedua);
delay_ms(800);

//TAMPILKAN EEPROM BUJUR KE LCD
lcd_gotoxy(0,1);
lcd_putchar(bsimpan1_kedua);
delay_ms(800);

lcd_gotoxy(1,1);
lcd_putchar(bsimpan2_kedua);
delay_ms(800);

lcd_gotoxy(2,1);
lcd_putchar(bsimpan3_kedua);
delay_ms(800);

lcd_gotoxy(3,1);
lcd_putchar(bsimpan4_kedua);
delay_ms(800);

lcd_gotoxy(4,1);
lcd_putchar(bsimpan5_kedua);
delay_ms(800);

lcd_gotoxy(5,1);
lcd_putchar(bsimpan6_kedua);
delay_ms(800);

lcd_gotoxy(6,1);
lcd_putchar(bsimpan7_kedua);
delay_ms(800);

lcd_gotoxy(7,1);
lcd_putchar(bsimpan8_kedua);
delay_ms(800);

lcd_gotoxy(8,1);
lcd_putchar(bsimpan9_kedua);
delay_ms(800);
lcd_clear();

lsimpan1 -= 0x30;
lsimpan2 -= 0x30;
lsimpan3 -= 0x30;
lsimpan4 -= 0x30;
lsimpan5 -= 0x30;
lsimpan6 -= 0x30;
lsimpan7 -= 0x30;
lsimpan8 -= 0x30;

bsimpan1 -= 0x30;
bsimpan2 -= 0x30;
bsimpan3 -= 0x30;
bsimpan4 -= 0x30;
bsimpan5 -= 0x30;
bsimpan6 -= 0x30;
bsimpan7 -= 0x30;
bsimpan8 -= 0x30;
bsimpan9 -= 0x30;

lsimpan1_kedua -= 0x30;
lsimpan2_kedua -= 0x30;
lsimpan3_kedua -= 0x30;
lsimpan4_kedua -= 0x30;
lsimpan5_kedua -= 0x30;
lsimpan6_kedua -= 0x30;
lsimpan7_kedua -= 0x30;
lsimpan8_kedua -= 0x30;

bsimpan1_kedua -= 0x30;
bsimpan2_kedua -= 0x30;
bsimpan3_kedua -= 0x30;
bsimpan4_kedua -= 0x30;
bsimpan5_kedua -= 0x30;
bsimpan6_kedua -= 0x30;
bsimpan7_kedua -= 0x30;
bsimpan8_kedua -= 0x30;
bsimpan9_kedua -= 0x30;

    //PROSES KONVERSI LINTANG BUJUR
KESATU(CHECKPOINT)
    ax = ((lsimpan1 * 10) + (lsimpan2 * 1));
// 0 + 7 = 7
    b = ((lsimpan3 * 10) + (lsimpan4 * 1));
//56
    c = ((lsimpan5 * 1000) + (lsimpan6 * 100)
+ (lsimpan7 * 10) + (lsimpan8 * 1));  //7536
    cx = c * 0.0001; //0.7536
    d = b + cx;  //56.7536
    f=d*  0.016666666666666666666666666666667;
// 0.94589 (konversi menit ke derajat) 1/60
```

```
    g = ax + f;  //7.94589    (hasil konversi
satuan derajat)

    ftoa(g,5,temp);     //menampilkan di LCD
    lcd_gotoxy(0,0);
    lcd_puts(temp);
    delay_ms(5000);

    h = ((bsimpan1 * 100) + (bsimpan2 * 10) +
(bsimpan3 * 1));    // 100+10+2 = 112
    ix = ((bsimpan4 * 10) + (bsimpan5 * 1));
//37
    j = ((bsimpan6 * 1000) + (bsimpan7 * 100)
+ (bsimpan8 * 10) + (bsimpan9 * 1));  //112
    jx = j * 0.0001; //0.0112
    k = ix + jx;  //37.0112
    m=k*  0.01666666666666666666666666666667;
// 0.61685 (konversi menit ke derajat) 1/60
    n = h + m;  //112.61685    (hasil konversi
satuan derajat)

    ftoa(n,5,temp);
    lcd_gotoxy(0,1);
    lcd_puts(temp);
    delay_ms(5000);
    lcd_clear();

    //PROSES     KONVERSI     LINTANG     BUJUR
KEDUA(SRU)
    ax2    =    ((lsimpan1_kedua    *    10)    +
(lsimpan2_kedua * 1));    // 0 + 7 = 7
    b2    =    ((lsimpan3_kedua    *    10)    +
(lsimpan4_kedua * 1)); //56
    c2    =    ((lsimpan5_kedua    *    1000)    +
(lsimpan6_kedua  *  100)  +  (lsimpan7_kedua  *
10) + (lsimpan8_kedua * 1));  //7536
    cx2 = c2 * 0.0001; //0.7536
    d2 = b2 + cx2;  //56.7536
f2=d2*0.01666666666666666666666666667;  //
0.94589 (konversi menit ke derajat) 1/60
    g2  =  ax2  +  f2;    //7.94589      (hasil
konversi satuan derajat)

    ftoa(g2,5,temp);
    lcd_gotoxy(0,0);
    lcd_puts(temp);
    delay_ms(5000);

    h2    =    ((bsimpan1_kedua    *    100)    +
(bsimpan2_kedua  *  10)  +  (bsimpan3_kedua  *
1));    // 100+10+2 = 112
    ix2    =    ((bsimpan4_kedua    *    10)    +
(bsimpan5_kedua * 1)); //37
    j2    =    ((bsimpan6_kedua    *    1000)    +
(bsimpan7_kedua  *  100)  +  (bsimpan8_kedua  *
10) + (bsimpan9_kedua * 1));  //112
    jx2 = j2 * 0.0001; //0.0112
    k2 = ix2 + jx2;  //37.0112

m2=k2*0.01666666666666666666666666666667;  //
0.61685 (konversi menit ke derajat) 1/60
    n2  =  h2  +  m2;    //112.61685      (hasil
konversi satuan derajat)

    ftoa(n2,5,temp);
    lcd_gotoxy(0,1);
    lcd_puts(temp);
    delay_ms(5000);
    lcd_clear();

    //PROSES PERHITUNGAN SUDUT DAN JARAK
    //VARIABEL g = LINTANG 1
    //VARIABEL n = BUJUR 1
    //VARIABEL g2 = LINTANG 2
    //VARIABEL n2 = BUJUR 2

    if ((g < g2) && (n > n2))
    {
        p = g2 - g;       // Delta Lintang (y)
        p = p * 60;       // Konversi ke menit
        p = p * 1885.37; // Konversi ke meter

        q = n - n2;      // Delta Bujur (x)
        q = q * 60;       // Konversi ke menit
        q = q * 1885.37; // Konversi ke meter

        r = p * p;      // p kuadrat
        s = q * q;      // q kuadrat
        t = r + s;    // t kuadrat = p kuadrat
+ q kuadrat
        u = sqrt(t);    // hasil jarak miring

        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("JARAK(meter):");
        ftoa(u,5,temp);
        lcd_gotoxy(0,1);
        lcd_puts(temp);
        delay_ms(5000);
        lcd_clear();

     v = p / q;         //Delta y / Delta x
    w = atan(v);    // hasil sudut
    w = w * 57.325;//convert radian to degree
        w = 90 - w;
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("SUDUT(derajat):");
        ftoa(w,5,temp);
        lcd_gotoxy(0,1);
        lcd_puts(temp);
        delay_ms(5000);
        lcd_clear();
    }

    if ((g < g2) && (n < n2))
    {
        p = g2 - g;       // Delta Lintang (y)
        p = p * 60;       // Konversi ke menit
        p = p * 1885.37; // Konversi ke meter

        q = n2 - n;      // Delta Bujur (x)
        q = q * 60;       // Konversi ke menit
        q = q * 1885.37; // Konversi ke meter

        r = p * p;      // p kuadrat
        s = q * q;      // q kuadrat
        t = r + s;    // t kuadrat = p kuadrat
+ q kuadrat
        u = sqrt(t);    // hasil jarak miring

        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("JARAK(meter):");
        ftoa(u,5,temp);
        lcd_gotoxy(0,1);
        lcd_puts(temp);
        delay_ms(5000);
        lcd_clear();

        v = p / q;         //Delta y / Delta x
     w = atan(v);    // hasil sudut
    w = w * 57.325;//convert radian to degree
        w = 270 + w;
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("SUDUT(derajat):");
        ftoa(w,5,temp);
        lcd_gotoxy(0,1);
        lcd_puts(temp);
        delay_ms(5000);
        lcd_clear();
    }

    if ((g > g2) && (n < n2))
    {
        p = g - g2;       // Delta Lintang (y)
        p = p * 60;      // Konversi ke menit
        p = p * 1885.37; // Konversi ke meter
```

```
        q = n2 - n;   // Delta Bujur (x)
        q = q * 60;     // Konversi ke menit
        q = q * 1885.37; // Konversi ke meter

        r = p * p;     // p kuadrat
        s = q * q;     // q kuadrat
        t = r + s;    // t kuadrat = p kuadrat
+ q kuadrat
        u = sqrt(t);    // hasil jarak miring

        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("JARAK(meter):");
        ftoa(u,5,temp);
        lcd_gotoxy(0,1);
        lcd_puts(temp);
        delay_ms(5000);
        lcd_clear();

        v = p / q;        //Delta y / Delta x
     w = atan(v);    // hasil sudut
    w = w * 57.325; //convert radian to degree
        w = 270 - w;
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("SUDUT(derajat):");
        ftoa(w,5,temp);
        lcd_gotoxy(0,1);
        lcd_puts(temp);
        delay_ms(5000);
        lcd_clear();
    }

    if ((g > g2) && (n > n2))
    {
        p = g - g2;       // Delta Lintang (y)
        p = p * 60;       // Konversi ke menit
        p = p * 1885.37; // Konversi ke meter

        q = n - n2;   // Delta Bujur (x)
        q = q * 60;     // Konversi ke menit
        q = q * 1885.37; // Konversi ke meter

        r = p * p;     // p kuadrat
        s = q * q;     // q kuadrat
        t = r + s;    // t kuadrat = p kuadrat
+ q kuadrat
        u = sqrt(t);   // hasil jarak miring

        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("JARAK(meter):");
        ftoa(u,5,temp);
        lcd_gotoxy(0,1);
        lcd_puts(temp);
        delay_ms(5000);
        lcd_clear();

        v = p / q;        //Delta y / Delta x
     w = atan(v);   // hasil sudut
    w = w * 57.325; //convert radian to degree
        w = 90 + w;
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("SUDUT(derajat):");
        ftoa(w,5,temp);
        lcd_gotoxy(0,1);
        lcd_puts(temp);
        delay_ms(5000);
        lcd_clear();
    }
    if ((g == g2) && (n == n2))
    {
        lcd_gotoxy(0,0);
        lcd_putsf("SAMA CHECKPOINT");
        delay_ms(1000);
        lcd_clear();
    }
}
```

```
void konversi_L_B()
{
    DDRD = 0x00;
    PORTD = 0xFF;

    baca_data_ke_eeproml1();
    baca_data_ke_eeproml2();
    baca_data_ke_eeproml3();
    baca_data_ke_eeproml4();
    baca_data_ke_eeproml5();
    baca_data_ke_eeproml6();
    baca_data_ke_eeproml7();
    baca_data_ke_eeproml8();

    baca_data_ke_eepromb1();
    baca_data_ke_eepromb2();
    baca_data_ke_eepromb3();
    baca_data_ke_eepromb4();
    baca_data_ke_eepromb5();
    baca_data_ke_eepromb6();
    baca_data_ke_eepromb7();
    baca_data_ke_eepromb8();
    baca_data_ke_eepromb9();

    while(PIND.5 == 0 && PIND.6 == 0)
    {
        gps();

        if(status == 'A')
        {
            simpan_gps_eeprom();
            lcd_clear();
            lcd_putsf("SIMPAN KE EEPROM");
            delay_ms(2000);
            lcd_clear();
            lcd_putsf("DATA TERSIMPAN..");
            delay_ms(2000);
            lcd_clear();
        }
    }

    //TAMPILKAN EEPROM LINTANG KE LCD
    lcd_gotoxy(0,0);
    lcd_putchar(lsimpan1);
    delay_ms(800);

    lcd_gotoxy(1,0);
    lcd_putchar(lsimpan2);
    delay_ms(800);

    lcd_gotoxy(2,0);
    lcd_putchar(lsimpan3);
    delay_ms(800);

    lcd_gotoxy(3,0);
    lcd_putchar(lsimpan4);
    delay_ms(800);

    lcd_gotoxy(4,0);
    lcd_putchar(lsimpan5);
    delay_ms(800);

    lcd_gotoxy(5,0);
    lcd_putchar(lsimpan6);
    delay_ms(800);

    lcd_gotoxy(6,0);
    lcd_putchar(lsimpan7);
    delay_ms(800);

    lcd_gotoxy(7,0);
    lcd_putchar(lsimpan8);
    delay_ms(800);

    //TAMPILKAN EEPROM BUJUR KE LCD
    lcd_gotoxy(0,1);
    lcd_putchar(bsimpan1);
    delay_ms(800);

    lcd_gotoxy(1,1);
```

```
    lcd_putchar(bsimpan2);
    delay_ms(800);

    lcd_gotoxy(2,1);
    lcd_putchar(bsimpan3);
    delay_ms(800);

    lcd_gotoxy(3,1);
    lcd_putchar(bsimpan4);
    delay_ms(800);

    lcd_gotoxy(4,1);
    lcd_putchar(bsimpan5);
    delay_ms(800);

    lcd_gotoxy(5,1);
    lcd_putchar(bsimpan6);
    delay_ms(800);

    lcd_gotoxy(6,1);
    lcd_putchar(bsimpan7);
    delay_ms(800);

    lcd_gotoxy(7,1);
    lcd_putchar(bsimpan8);
    delay_ms(800);

    lcd_gotoxy(8,1);
    lcd_putchar(bsimpan9);
    delay_ms(800);
    lcd_clear();
}

void simpan_gps_eeprom()
{
    lsimpan1 = lintang[0];   //SIMPAN LINTANG
    tulis_data_ke_eeproml1();

    lsimpan2 = lintang[1];
    tulis_data_ke_eeproml2();

    lsimpan3 = lintang[2];
    tulis_data_ke_eeproml3();

    lsimpan4 = lintang[3];
    tulis_data_ke_eeproml4();

    lsimpan5 = lintang[5];
    tulis_data_ke_eeproml5();

    lsimpan6 = lintang[6];
    tulis_data_ke_eeproml6();

    lsimpan7 = lintang[7];
    tulis_data_ke_eeproml7();

    lsimpan8 = lintang[8];
    tulis_data_ke_eeproml8();

    bsimpan1 = bujur[0];  //SIMPAN BUJUR
    tulis_data_ke_eepromb1();

    bsimpan2 = bujur[1];
    tulis_data_ke_eepromb2();

    bsimpan3 = bujur[2];
    tulis_data_ke_eepromb3();

    bsimpan4 = bujur[3];
    tulis_data_ke_eepromb4();

    bsimpan5 = bujur[4];
    tulis_data_ke_eepromb5();

    bsimpan6 = bujur[6];
    tulis_data_ke_eepromb6();

    bsimpan7 = bujur[7];
    tulis_data_ke_eepromb7();

    bsimpan8 = bujur[8];
    tulis_data_ke_eepromb8();

    bsimpan9 = bujur[9];
    tulis_data_ke_eepromb9();
}

//EEPROM LINTANG
void tulis_data_ke_eeproml1()
{
    E_lsimpan1 = lsimpan1;
}

void baca_data_ke_eeproml1()
{
    lsimpan1 = E_lsimpan1;
}

void tulis_data_ke_eeproml2()
{
    E_lsimpan2 = lsimpan2;
}

void baca_data_ke_eeproml2()
{
     lsimpan2 = E_lsimpan2;
}

void tulis_data_ke_eeproml3()
{
    E_lsimpan3 = lsimpan3;
}

void baca_data_ke_eeproml3()
{
     lsimpan3 = E_lsimpan3;
}

void tulis_data_ke_eeproml4()
{
    E_lsimpan4 = lsimpan4;
}

void baca_data_ke_eeproml4()
{
     lsimpan4 = E_lsimpan4;
}

void tulis_data_ke_eeproml5()
{
    E_lsimpan5 = lsimpan5;
}

void baca_data_ke_eeproml5()
{
     lsimpan5 = E_lsimpan5;
}

void tulis_data_ke_eeproml6()
{
    E_lsimpan6 = lsimpan6;
}

void baca_data_ke_eeproml6()
{
     lsimpan6 = E_lsimpan6;
}

void tulis_data_ke_eeproml7()
{
    E_lsimpan7 = lsimpan7;
}

void baca_data_ke_eeproml7()
{
     lsimpan7 = E_lsimpan7;
}
```

```
void tulis_data_ke_eeprom18()
{
   E_lsimpan8 = lsimpan8;
}

void baca_data_ke_eeprom18()
{
    lsimpan8 = E_lsimpan8;
}
//EEPROM BUJUR
void tulis_data_ke_eepromb1()
{
    E_bsimpan1 = bsimpan1;
}
void baca_data_ke_eepromb1()
{
    bsimpan1 = E_bsimpan1;
}

void tulis_data_ke_eepromb2()
{
    E_bsimpan2 = bsimpan2;
}

void baca_data_ke_eepromb2()
{
    bsimpan2 = E_bsimpan2;
}

void tulis_data_ke_eepromb3()
{
    E_bsimpan3 = bsimpan3;
}

void baca_data_ke_eepromb3()
{
    bsimpan3 = E_bsimpan3;
}

void tulis_data_ke_eepromb4()
{
    E_bsimpan4 = bsimpan4;
}

void baca_data_ke_eepromb4()
{
    bsimpan4 = E_bsimpan4;
}

void tulis_data_ke_eepromb5()
{
    E_bsimpan5 = bsimpan5;
}

void baca_data_ke_eepromb5()
{
    bsimpan5 = E_bsimpan5;
}

void tulis_data_ke_eepromb6()
{
    E_bsimpan6 = bsimpan6;
}

void baca_data_ke_eepromb6()
{
    bsimpan6 = E_bsimpan6;
}

void tulis_data_ke_eepromb7()
{
    E_bsimpan7 = bsimpan7;
}

void baca_data_ke_eepromb7()
{
    bsimpan7 = E_bsimpan7;
}

void tulis_data_ke_eepromb8()
{
   E_bsimpan8 = bsimpan8;
}

void baca_data_ke_eepromb8()
{
    bsimpan8 = E_bsimpan8;
}

void tulis_data_ke_eepromb9()
{
   E_bsimpan9 = bsimpan9;
}

void baca_data_ke_eepromb9()
{
    bsimpan9 = E_bsimpan9;
}

void simpan_gps_eeprom_kedua()
{
    lsimpan1_kedua=lintang[0];//SIMPAN
LINTANG KEDUA
    tulis_data_ke_eeprom11_kedua();

    lsimpan2_kedua = lintang[1];
    tulis_data_ke_eeprom12_kedua();

    lsimpan3_kedua = lintang[2];
    tulis_data_ke_eeprom13_kedua();

    lsimpan4_kedua = lintang[3];
    tulis_data_ke_eeprom14_kedua();

    lsimpan5_kedua = lintang[5];
    tulis_data_ke_eeprom15_kedua();

    lsimpan6_kedua = lintang[6];
    tulis_data_ke_eeprom16_kedua();

    lsimpan7_kedua = lintang[7];
    tulis_data_ke_eeprom17_kedua();

    lsimpan8_kedua = lintang[8];
    tulis_data_ke_eeprom18_kedua();

    bsimpan1_kedua = bujur[0]; //SIMPAN BUJUR
KEDUA
    tulis_data_ke_eepromb1_kedua();

    bsimpan2_kedua = bujur[1];
    tulis_data_ke_eepromb2_kedua();

    bsimpan3_kedua = bujur[2];
    tulis_data_ke_eepromb3_kedua();

    bsimpan4_kedua = bujur[3];
    tulis_data_ke_eepromb4_kedua();

    bsimpan5_kedua = bujur[4];
    tulis_data_ke_eepromb5_kedua();

    bsimpan6_kedua = bujur[6];
    tulis_data_ke_eepromb6_kedua();

    bsimpan7_kedua = bujur[7];
    tulis_data_ke_eepromb7_kedua();

    bsimpan8_kedua = bujur[8];
    tulis_data_ke_eepromb8_kedua();

    bsimpan9_kedua = bujur[9];
    tulis_data_ke_eepromb9_kedua();
}
```

```
//EEPROM LINTANG KEDUA
void tulis_data_ke_eeproml1_kedua()
{
    E_lsimpan1_kedua = lsimpan1_kedua;
}

void baca_data_ke_eeproml1_kedua()
{
    lsimpan1_kedua = E_lsimpan1_kedua;
}

void tulis_data_ke_eeproml2_kedua()
{
    E_lsimpan2_kedua = lsimpan2_kedua;
}
void baca_data_ke_eeproml2_kedua()
{
     lsimpan2_kedua = E_lsimpan2_kedua;
}
void tulis_data_ke_eeproml3_kedua()
{
    E_lsimpan3_kedua = lsimpan3_kedua;
}

void baca_data_ke_eeproml3_kedua()
{
     lsimpan3_kedua = E_lsimpan3_kedua;
}

void tulis_data_ke_eeproml4_kedua()
{
    E_lsimpan4_kedua = lsimpan4_kedua;
}

void baca_data_ke_eeproml4_kedua()
{
     lsimpan4_kedua = E_lsimpan4_kedua;
}

void tulis_data_ke_eeproml5_kedua()
{
    E_lsimpan5_kedua = lsimpan5_kedua;
}

void baca_data_ke_eeproml5_kedua()
{
     lsimpan5_kedua = E_lsimpan5_kedua;
}

void tulis_data_ke_eeproml6_kedua()
{
    E_lsimpan6_kedua = lsimpan6_kedua;
}

void baca_data_ke_eeproml6_kedua()
{
     lsimpan6_kedua = E_lsimpan6_kedua;
}

void tulis_data_ke_eeproml7_kedua()
{
    E_lsimpan7_kedua = lsimpan7_kedua;
}

void baca_data_ke_eeproml7_kedua()
{
     lsimpan7_kedua = E_lsimpan7_kedua;
}

void tulis_data_ke_eeproml8_kedua()
{
   E_lsimpan8_kedua = lsimpan8_kedua;
}

void baca_data_ke_eeproml8_kedua()
{
     lsimpan8_kedua = E_lsimpan8_kedua;
}
```

```
//EEPROM BUJUR KEDUA
void tulis_data_ke_eepromb1_kedua()
{
    E_bsimpan1_kedua = bsimpan1_kedua;
}

void baca_data_ke_eepromb1_kedua()
{
    bsimpan1_kedua = E_bsimpan1_kedua;
}

void tulis_data_ke_eepromb2_kedua()
{
    E_bsimpan2_kedua = bsimpan2_kedua;
}

void baca_data_ke_eepromb2_kedua()
{
     bsimpan2_kedua = E_bsimpan2_kedua;
}
void tulis_data_ke_eepromb3_kedua()
{
    E_bsimpan3_kedua = bsimpan3_kedua;
}

void baca_data_ke_eepromb3_kedua()
{
     bsimpan3_kedua = E_bsimpan3_kedua;
}

void tulis_data_ke_eepromb4_kedua()
{
    E_bsimpan4_kedua = bsimpan4_kedua;
}

void baca_data_ke_eepromb4_kedua()
{
     bsimpan4_kedua = E_bsimpan4_kedua;
}

void tulis_data_ke_eepromb5_kedua()
{
    E_bsimpan5_kedua = bsimpan5_kedua;
}

void baca_data_ke_eepromb5_kedua()
{
     bsimpan5_kedua = E_bsimpan5_kedua;
}

void tulis_data_ke_eepromb6_kedua()
{
    E_bsimpan6_kedua = bsimpan6_kedua;
}

void baca_data_ke_eepromb6_kedua()
{
    bsimpan6_kedua = E_bsimpan6_kedua;
}

void tulis_data_ke_eepromb7_kedua()
{
    E_bsimpan7_kedua = bsimpan7_kedua;
}

void baca_data_ke_eepromb7_kedua()
{
     bsimpan7_kedua = E_bsimpan7_kedua;
}

void tulis_data_ke_eepromb8_kedua()
{
   E_bsimpan8_kedua = bsimpan8_kedua;
}

void baca_data_ke_eepromb8_kedua()
{
     bsimpan8_kedua = E_bsimpan8_kedua;
}
```

```
void tulis_data_ke_eepromb9_kedua()
{
   E_bsimpan9_kedua = bsimpan9_kedua;
}

void baca_data_ke_eepromb9_kedua()
{
    bsimpan9_kedua = E_bsimpan9_kedua;
}

void lintang_sms()
{
    printf("http://maps.google.com/maps?q=");

    if (arah_lintang=='S')
    {
        printf(" S ");
    }

    else
    {
        printf(" U ");
    }
    putchar(lintang[0]);
    putchar(lintang[1]);
    printf(" ");
    putchar(lintang[2]);
    putchar(lintang[3]);
    printf(".");
    putchar(lintang[5]);
    putchar(lintang[6]);
    putchar(lintang[7]);
    putchar(lintang[8]);
}

void bujur_sms()
{
    if (arah_bujur=='E')
    {
        printf(" E ");
    }
    else
    {
        printf(" W ");
    }

    putchar(bujur[0]);
    putchar(bujur[1]);
    putchar(bujur[2]);
    printf(" ");
    putchar(bujur[3]);
    putchar(bujur[4]);
    printf(".");
    putchar(bujur[6]);
    putchar(bujur[7]);
    putchar(bujur[8]);
    putchar(bujur[9]);
}

void tampil_lintang()
{
    for (i=0;i<9;i++)
    {
        lcd_gotoxy(0,0);
        lcd_putsf("Koord. Lintang:");
        lcd_gotoxy(i,1);
        lcd_putchar(lintang[i]);
        delay_ms(100);
    }
    if (arah_lintang=='S')
    {
        lcd_putsf(" LS");
    }
    else
    {
        lcd_putsf(" LU");
    }
    delay_ms(2000);
    lcd_clear();
}

void tampil_bujur()
{
    for (i=0;i<10;i++)
    {
        lcd_gotoxy(0,0);
        lcd_putsf("Koord. Bujur:");
        lcd_gotoxy(i,1);
        lcd_putchar(bujur[i]);
        delay_ms(100);
    }
    if (arah_bujur=='E')
    {
        lcd_putsf(" BT");
    }
    else
    {
        lcd_putsf(" BB");
    }
    delay_ms(2000);
    lcd_clear();
}

void tampil_status()
{
    lcd_gotoxy(0,0);
    lcd_putsf("Status:");
    lcd_putchar(status);
    delay_ms(500);
}

void send_sms(char flash *fmtstr1)
{
    char dat;
    awal_sms:

    delay_ms(500);
    lcd_clear();
    lcd_gotoxy(0,0);
    lcd_putsf("Kirim SMS....");

    printf("AT+CMGF=1");
    putchar(0x0D);//ENTER
    while(getchar()!=0x0A){};    //0x0A = New
Line
    while(getchar()!=0x0A){};
    delay_ms(500);

    printf("AT+CMGS=\"");
    printf(fmtstr1);
    printf("\"");
    putchar(0x0D);  //ENTER
    while(getchar()!=0x20){}; //spasi
    delay_ms(500);
    lintang_sms();
    bujur_sms();
    putchar(0x1A);  //ctrl+z (kirim)

    while((dat=getchar())!=0x0A)
    {
        if (dat=='O')
        {
            if(getchar()=='R')
            {
                getchar();
                 delay_ms(500);
                lcd_clear();
                lcd_gotoxy(0,0);
                lcd_putsf("SMS Gagal");
                printf("AT+CFUN=1");
                putchar(0x0D);
                while(getchar()!='K'){};
                while(getchar()!=0x0A){};
                delay_ms(15000);
                 goto awal_sms;
            }
        }
    };

    delay_ms(500);
```

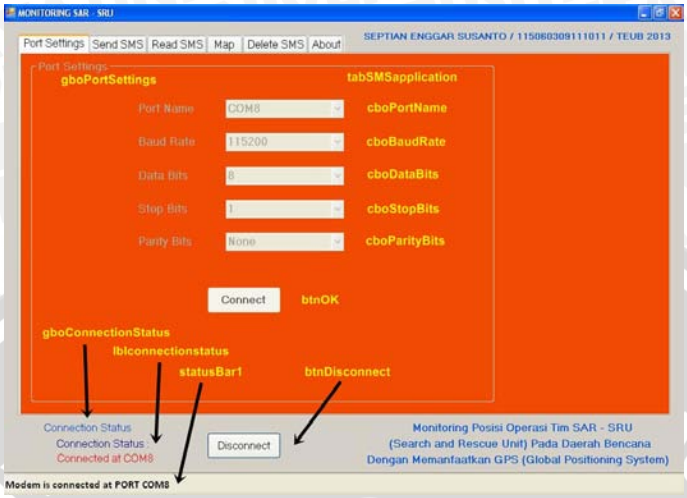
```
    printf("AT+CMGD=1");    //HAPUS SMS
    putchar(0x0D);//ENTER
    while(getchar()!=0x0A){};
    while(getchar()!=0x0A){};
    delay_ms(500);

    printf("AT+CMGD=2");
    putchar(0x0D);//ENTER
    while(getchar()!=0x0A){};
    while(getchar()!=0x0A){};
    delay_ms(500);


    printf("AT+CMGD=3");
    putchar(0x0D);//ENTER
    while(getchar()!=0x0A){};
    while(getchar()!=0x0A){};

    lcd_clear();
    lcd_gotoxy(0,0);
    lcd_putsf("SMS Terkirim..Ok");
    delay_ms(5000);
    lcd_clear();
}

void gps()
{
    loop:
        data=getchar1();
        if(data=='$')
        {
            data=getchar1();
            if(data=='G')
            {
                data=getchar1();
                if(data=='P')
                {
                    data=getchar1();
                    if(data=='R')
                    {
                        data=getchar1();
                        if(data=='M')
                        {
                            data=getchar1();
                            if(data=='C')
                            {
data=getchar1();
data=getchar1();
while (data!=',')
{
data=getchar1();
}
status=getchar1();
if (status=='V')
{
lcd_clear();
tampil_status();
lcd_gotoxy(0,1);
lcd_putsf("Data Belum Valid");
delay_ms(500);
goto loop;
}
data=getchar1();
lintang[0]=getchar1();
data=lintang[0];
for (i=1;data!=',';i++)
{
lintang[i]=getchar1();
data=lintang[i];
}
arah_lintang=getchar1();
data=getchar1();
bujur[0]=getchar1();
data=bujur[0];
for (i=1;data!=',';i++)
{
bujur[i]=getchar1();
data=bujur[i];
arah_bujur=getchar1();
data=getchar1();
kecepatan[0]=getchar1();
data=kecepatan[0];
for (i=1;data!=',';i++)
{
kecepatan[i]=getchar1();
data=kecepatan[i];
}
lcd_clear();
lcd_putsf("Data Valid");
delay_ms(500);
lcd_clear();
tampil_status();
lcd_clear();
tampil_lintang();
tampil_bujur();
                            }
                        }
                    }
                }
            }
        }
}

void baca_sms()
{
    lcd_gotoxy(0,0);
    lcd_putsf("WAVECOM BLM SIAP");
    delay_ms(1000);

    printf("ATE0");// perintah agar tidak ada
echo
    putchar(0x0D);              // enter
    while(getchar()!=   'K'){};     //  tunggu
sampai menerima K / OK
    while(getchar()!=0x0A){};      //   tunggu
sampai newline

    lcd_clear();
    lcd_gotoxy(0,0);
    lcd_putsf("WAVECOM SIAP....");
    delay_ms(800);
    reload:
      lcd_clear();
      lcd_gotoxy(0,0);
      lcd_putsf("TUNGGU SMS");
      delay_ms(600);
      lcd_gotoxy(10,0);
      lcd_putsf(".");
      delay_ms(600);
      lcd_gotoxy(11,0);
      lcd_putsf(".");
      delay_ms(600);
      lcd_gotoxy(12,0);
      lcd_putsf(".");
      printf("AT+CPMS=");        // penempatan
memori sms
      putchar('"');
      printf("SM");
      putchar('"');
      putchar(0x0D);
      while(getchar()!= 'K'){};
      while(getchar()!=0x0A){};
      printf("AT+CMGF=1");
      putchar(0x0D);
      while(getchar()!= 'K'){};
      while(getchar()!=0x0A){};
      printf("AT+CMGL=");
      putchar('"');
      printf("ALL");
      putchar('"');
      putchar(0x0D);
      while(getchar()!=0x0A){};
      a=0;
      do
      {
          a++;
          cek[a]=getchar();
      }
      while(cek[a]!=0x0A);
      if(cek[1]=='O')
```

```
        {
          for(a=0;a<70;a++){cek[a]=' ';}
          delay_ms(50);
          goto reload;
        }
        a=0;
        do
        {
          a++;
          cek[a]=getchar();
        }
        while(cek[a]!=0x0A);
if((cek[1]=='G')&&(cek[2]=='P')&&
(cek[3]=='S'))
        {
          buzzer = 1;
          delay_ms(800);
          buzzer = 0;
          delay_ms(800);
          buzzer = 1;
          delay_ms(800);
          buzzer = 0;
          lcd_clear();
          lcd_gotoxy(0,0);
          lcd_putsf("ISI SMS:");
          lcd_gotoxy(0,1);
          lcd_putsf("GPS");
          printf("AT+CMGD=1");
          putchar(0x0D);
          while(getchar()!='K'){};
          while(getchar()!=0x0A){};
          printf("AT+CMGD=1");
          putchar(0x0D);
          while(getchar()!='K'){};
          while(getchar()!=0x0A){};
          lcd_clear();

          while(status!='A')
          {
              gps();
          }
          send_sms("085735133736");
          send_sms("085648096965");
          status='V';
        }
if((cek[1]=='K')&&(cek[2]=='O')&&(cek[3]=='R'
)&&(cek[4]=='B')&&(cek[5]=='A')&&
(cek[6]=='N'))
        {
          buzzer = 1;
          delay_ms(800);
          buzzer = 0;
          delay_ms(800);
          buzzer = 1;
          delay_ms(800);
          buzzer = 0;
          delay_ms(800);
          buzzer = 1;
          delay_ms(800);
          buzzer = 0;
          lcd_clear();
          lcd_gotoxy(0,0);
          lcd_putsf("ISI SMS:");
          lcd_gotoxy(0,1);
          lcd_putsf("KORBAN");
          printf("AT+CMGD=1");
          putchar(0x0D);
          while(getchar()!='K'){};
          while(getchar()!=0x0A){};
          printf("AT+CMGD=1");
          putchar(0x0D);
          while(getchar()!='K'){};
          while(getchar()!=0x0A){};
          lcd_clear();
          lcd_gotoxy(0,0);
          lcd_putsf(" AKTIFKAN  MENU ");
          lcd_gotoxy(0,1);
          lcd_putsf("     KEYPAD     ");
          delay_ms(5000);
         }
if((cek[1]!='K'&&cek[1]!='G')&&(cek[2]!='O'&&
cek[2]!='P')&&(cek[3]!='R'&&cek[3]!='S')&&(ce
k[4]!='B')&&(cek[5]!='A')&&(cek[6]!='N'))
        {
          buzzer = 1;
          delay_ms(800);
          buzzer = 0;
          printf("AT+CMGD=1");
          putchar(0x0D);
          while(getchar()!='K'){};
          while(getchar()!=0x0A){};
          printf("AT+CMGD=1");
          putchar(0x0D);
          while(getchar()!='K'){};
          while(getchar()!=0x0A){};
          printf("AT+CMGF=1");
          putchar(0x0D);
          while(getchar()!= 'K'){};
          while(getchar()!=0x0A){};
          printf("AT+CMGS=");
          putchar('"');
          printf("085735133736");
          putchar('"');
          putchar(0x0D);
          while(getchar()!=0x3E){};
          delay_ms(10);
          printf("Format SMS Salah");
          lcd_clear();
          lcd_gotoxy(0,0);
          lcd_putsf("ISI SMS:");
          lcd_gotoxy(0,1);
          lcd_putsf("Format SMS Salah");
          putchar(0x1A);
          while(getchar()!='K'){};
          lcd_clear();
        }
       delay_ms(50);
}

unsigned char keypad4x3()
{
    unsigned char key_press = '@';

    PORTA=0b11111101;
    if(PINA.5==0)   key_press = '*';
    if(PINA.6==0)   key_press = '2';
    if(PINA.7==0)   key_press = '3';
    PORTA=0b11111011;
    if(PINA.5==0)   key_press = '1';
    if(PINA.6==0)   key_press = '5';
    if(PINA.7==0)   key_press = '6';
    PORTA=0b11110111;
    if(PINA.5==0)   key_press = '4';
    if(PINA.6==0)   key_press = '8';
    if(PINA.7==0)   key_press = '9';
    PORTA=0b11101111;
    if(PINA.5==0)   key_press = '7';
    if(PINA.6==0)   key_press = '0';
    if(PINA.7==0)   key_press = '#';

    delay_ms(50);
    return key_press;
}

void tes_keypad()
{
    key = keypad4x3();
    if (key !='@')
    {
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("JMLH KORBAN: ");
        lcd_putchar(key);
        simpankeypad = key;
        E_simpankeypad = simpankeypad;
        delay_ms(2000);

    }
}
```

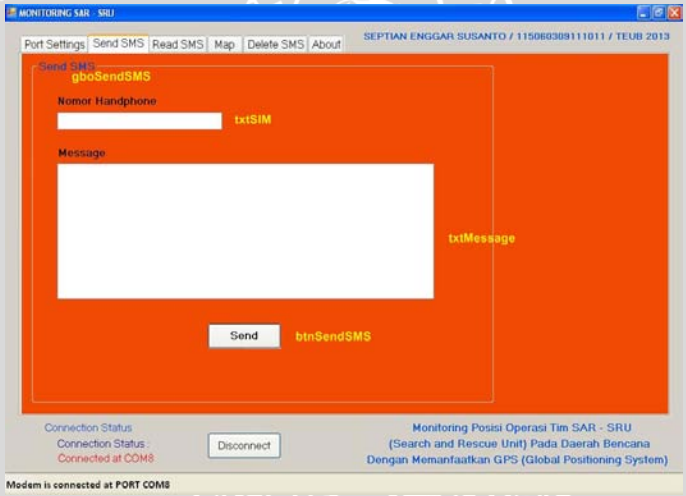
LAMPIRAN IV

LISTING PROGRAM SISTEM TIM SAR PEMANTAU

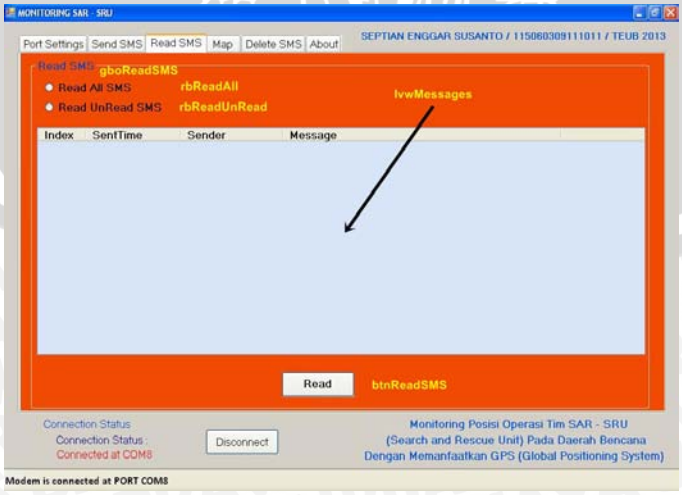
(*MICROSOFT VISUAL C# 2005*)



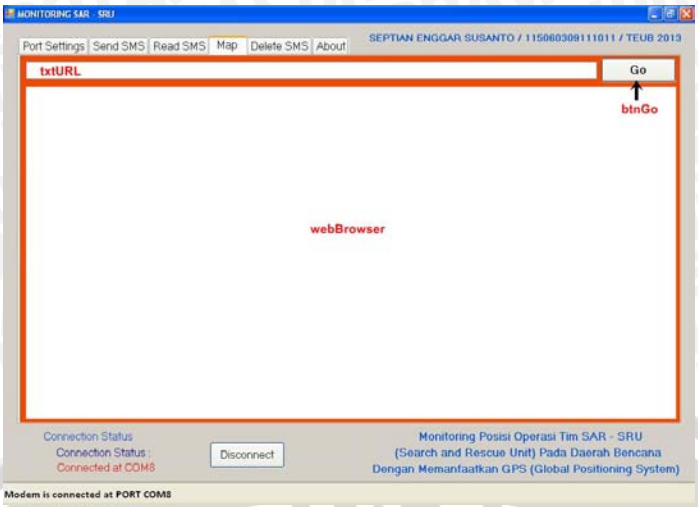
Gambar 4. Menu Port Settings



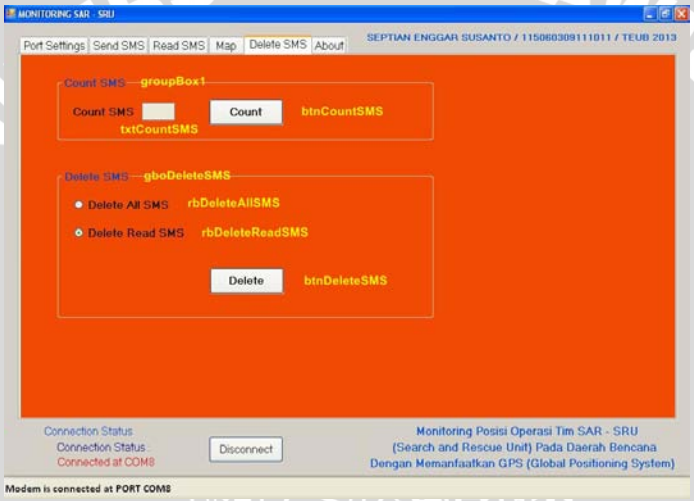
Gambar 5. Menu Send SMS



Gambar 6. Menu Read SMS



Gambar 7. Menu Map



Gambar 8. Menu Delete SMS

Gambar 9. Menu About

```
//***SMSapplication.Cs***//
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.IO;
using System.IO.Ports;

namespace SMSapplication
{
   public partial class SMSapplication : Form
    {
        #region Constructor
        public SMSapplication()
        {
            InitializeComponent();
        }
        #endregion

        #region Private Variables
        SerialPort port = new SerialPort();
        clsSMS objclsSMS = new clsSMS();
    ShortMessageCollection
objShortMessageCollection        =        new
ShortMessageCollection();
        #endregion

        #region Private Methods

        #region Write StatusBar
        private   void   WriteStatusBar(string
status)
        {
            try
            {
                statusBar1.Text = "Message: "
+ status;
            }
            catch (Exception ex)
            {

            }
        }
        #endregion

        #endregion

        #region Private Events

     private void SMSapplication_Load(object
sender, EventArgs e)
        {
            try
            {
                #region Display all available
COM Ports
                string[]        ports        =
SerialPort.GetPortNames();

    // Add all port names to the combo box:
                foreach   (string   port   in
ports)
                {
     this.cboPortName.Items.Add(port);
                }
                #endregion

                //Remove tab pages
this.tabSMSapplication.TabPages.Remove(tbSend
SMS);
this.tabSMSapplication.TabPages.Remove(tbRead
SMS);
this.tabSMSapplication.TabPages.Remove(tbMap)
;
this.tabSMSapplication.TabPages.Remove(tbDele
teSMS);
this.tabSMSapplication.TabPages.Remove(tbAbou
t);
        this.btnDisconnect.Enabled = false;
            }
            catch(Exception ex)
            {
                ErrorLog(ex.Message);
            }
        }

       private    void    btnOK_Click(object
sender, EventArgs e)
        {
            try
            {
          //Open communication port
this.port=
objclsSMS.OpenPort(this.cboPortName.Text,
Convert.ToInt32(this.cboBaudRate.Text),
Convert.ToInt32(this.cboDataBits.Text));

                if (this.port != null)
                {
this.gboPortSettings.Enabled = false;

 //MessageBox.Show("Modem   is   connected   at
PORT " + this.cboPortName.Text);
                    this.statusBar1.Text     =
"Modem    is    connected    at    PORT    "    +
this.cboPortName.Text;

 //Add tab pages
this.tabSMSapplication.TabPages.Add(tbSendSMS
);
this.tabSMSapplication.TabPages.Add(tbReadSMS
);
this.tabSMSapplication.TabPages.Add(tbMap);
this.tabSMSapplication.TabPages.Add(tbDeleteS
MS);
this.tabSMSapplication.TabPages.Add(tbAbout);
this.lblConnectionStatus.Text = "Connected at
" + this.cboPortName.Text;
this.btnDisconnect.Enabled = true;
                }
                else
                {

//MessageBox.Show("Invalid port settings");
                    this.statusBar1.Text     =
"Invalid port settings";
                }
            }
            catch (Exception ex)
            {
                ErrorLog(ex.Message);
            }

        }
       private void btnDisconnect_Click(object
sender, EventArgs e)
        {
            try
            {
                this.gboPortSettings.Enabled
= true;

objclsSMS.ClosePort(this.port);

//Remove tab pages
this.tabSMSapplication.TabPages.Remove(tbSend
SMS);
this.tabSMSapplication.TabPages.Remove(tbRead
SMS);
this.tabSMSapplication.TabPages.Remove(tbMap)
;
this.tabSMSapplication.TabPages.Remove(tbDele
teSMS);
this.tabSMSapplication.TabPages.Remove(tbAbou
t);
```

```
this.lblConnectionStatus.Text = "Not Connected";
            this.btnDisconnect.Enabled = false;
            }
            catch (Exception ex)
            {
                ErrorLog(ex.Message);
            }
        }

        private void btnSendSMS_Click(object sender, EventArgs e)
        {
//...Send SMS...//
            try
            {
             if (objclsSMS.sendMsg(this.port, this.txtSIM.Text, this.txtMessage.Text))
                {
   //MessageBox.Show("Message has sent successfully");
                    this.statusBar1.Text = "Message has sent successfully";
                }
                else
                {
                    //MessageBox.Show("Failed to send message");
                    this.statusBar1.Text = "Failed to send message";
                }
            }
            catch (Exception ex)
            {
                ErrorLog(ex.Message);
            }
        }
        private void btnReadSMS_Click(object sender, EventArgs e)
        {
            try
            {
                //count SMS
                int uCountSMS = objclsSMS.CountSMSmessages(this.port);
                if (uCountSMS > 0)
                {
                    #region Command
                    string strCommand = "AT+CMGL=\"ALL\"";
                  if (this.rbReadAll.Checked)
                    {
                        strCommand = "AT+CMGL=\"ALL\"";
                    }
              else if (this.rbReadUnRead.Checked)
                    {
        strCommand = "AT+CMGL=\"REC UNREAD\"";
                    }
                    #endregion

  // If SMS exist then read SMS
                    #region Read SMS
//...Read all SMS...//
                    objShortMessageCollection = objclsSMS.ReadSMS(this.port, strCommand);
                    foreach (ShortMessage msg in objShortMessageCollection)
                    {
                        ListViewItem item = new ListViewItem(new string[] { msg.Index, msg.Sent, msg.Sender, msg.Message });
                        item.Tag = msg;
                        txtURL.Text = msg.Message;
lvwMessages.Items.Add(item);
                    }
                    #endregion
                }
                if (uCountSMS == 0)
                {
                    lvwMessages.Clear();
//MessageBox.Show("There is no message in SIM");
                    this.statusBar1.Text = "There is no message in SIM";
                }
            }
            catch (Exception ex)
            {
                ErrorLog(ex.Message);
            }
        }
        private void btnDeleteSMS_Click(object sender, EventArgs e)
        {
            try
            {
//Count SMS
                int uCountSMS = objclsSMS.CountSMSmessages(this.port);
                if (uCountSMS > 0)
                {
                    DialogResult dr = MessageBox.Show("Are u sure u want to delete the SMS?", "Delete confirmation", MessageBoxButtons.YesNo);
                    if (dr.ToString() == "Yes")
                    {
                        #region Delete SMS
                        if (this.rbDeleteAllSMS.Checked)
                        {

//...Delete all SMS...//
                            #region Delete all SMS
                            string strCommand = "AT+CMGD=1,4";
                            if (objclsSMS.DeleteMsg(this.port, strCommand))
                            {
  //MessageBox.Show("Messages has deleted successfuly ");
this.statusBar1.Text = "Messages has deleted successfuly";
                            }
                            else
                            {
//MessageBox.Show("Failed to delete messages ");
this.statusBar1.Text = "Failed to delete messages";
                            }
                            #endregion
                        }
                    else if (this.rbDeleteReadSMS.Checked)
                        {

//...Delete Read SMS...//
                            #region Delete Read SMS
                            string strCommand = "AT+CMGD=1,3";
                            if (objclsSMS.DeleteMsg(this.port, strCommand))
                            {

//MessageBox.Show("Messages has deleted successfuly");
this.statusBar1.Text = "Messages has deleted successfuly";
                            }
                            else
                            {
//MessageBox.Show("Failed to delete messages ");
this.statusBar1.Text = "Failed to delete messages";
                            }
```

```
                #endregion
            }
            #endregion
        }
    }
    catch (Exception ex)
    {
        ErrorLog(ex.Message);
    }
}
private void btnCountSMS_Click(object
sender, EventArgs e)
{
    try
    {
//Count SMS
        int uCountSMS =
objclsSMS.CountSMSmessages(this.port);
        this.txtCountSMS.Text =
uCountSMS.ToString();
    }
    catch (Exception ex)
    {
        ErrorLog(ex.Message);
    }
}
#endregion
#region Error Log
public void ErrorLog(string Message)
{
    StreamWriter sw = null;
    try
    {
        WriteStatusBar(Message);
        string sLogFormat =
DateTime.Now.ToShortDateString().ToString() +
"                " +
DateTime.Now.ToLongTimeString().ToString() +
" ==> ";
        //string sPathName = @"E:\";
        string sPathName =
@"SMSapplicationErrorLog_";
        string sYear =
DateTime.Now.Year.ToString();
        string sMonth =
DateTime.Now.Month.ToString();
        string sDay =
DateTime.Now.Day.ToString();
        string sErrorTime = sDay + "-
" + sMonth + "-" + sYear;
        sw = new
StreamWriter(sPathName + sErrorTime + ".txt",
true);
        sw.WriteLine(sLogFormat +
Message);
        sw.Flush();
    }
    catch (Exception ex)
    {
        //ErrorLog(ex.ToString());
    }
    finally
    {
        if (sw != null)
        {
            sw.Dispose();
            sw.Close();
        }
    }
}
#endregion

private void btnGo_Click(object sender,
EventArgs e)
{
    webBrowser.Navigate(txtURL.Text);
}
```

```
//***clsSMS.Cs***//
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.IO.Ports;
using System.Threading;
using System.Text.RegularExpressions;

namespace SMSapplication
{
    public class clsSMS
    {
        #region Open and Close Ports
        //Open Port
        public SerialPort OpenPort(string
p_strPortName, int p_uBaudRate, int
p_uDataBits)
        {
            receiveNow = new
AutoResetEvent(false);
            SerialPort port = new
SerialPort();
            try
            {
                port.PortName =
p_strPortName;                //COM1
                port.BaudRate = p_uBaudRate;
//9600
                port.DataBits = p_uDataBits;
//8
                port.StopBits = StopBits.One;
//1
                port.Parity = Parity.None;
//None
                //port.ReadTimeout =
p_uReadTimeout;            //300
                //port.WriteTimeout =
p_uWriteTimeout;           //300
                port.Encoding =
Encoding.GetEncoding("iso-8859-1");
                port.DataReceived += new
SerialDataReceivedEventHandler(port_DataRecei
ved);
                port.Open();
                port.DtrEnable = true;
                port.RtsEnable = true;
            }
            catch (Exception ex)
            {
                throw ex;
            }
            return port;
        }

        //Close Port
        public void ClosePort(SerialPort
port)
        {
            try
            {
                port.Close();
                port.DataReceived -= new
SerialDataReceivedEventHandler(port_DataRecei
ved);
                port = null;
            }
            catch (Exception ex)
            {
                throw ex;
            }
        }
        #endregion
```

```
//Execute AT Command
        public string ExecCommand(SerialPort port,string command, int responseTimeout, string errorMessage)
        {
            try
            {
                port.DiscardOutBuffer();
                port.DiscardInBuffer();
                receiveNow.Reset();
                port.Write(command + "\r");
                string input = ReadResponse(port, responseTimeout);
                if ((input.Length == 0) || ((!input.EndsWith("\r\n>")) && (!input.EndsWith("\r\nOK\r\n"))))
                    throw new ApplicationException("No success message was received.");
                return input;
            }
            catch (Exception ex)
            {
                throw ex;
            }
        }

//Receive data from port
        public void port_DataReceived(object sender, SerialDataReceivedEventArgs e)
        {
            try
            {
                if (e.EventType == SerialData.Chars)
                {
                    receiveNow.Set();
                }
            }
            catch (Exception ex)
            {
                throw ex;
            }
        }
        public string ReadResponse(SerialPort port,int timeout)
        {
            string buffer = string.Empty;
            try
            {
                do
                {
                    if (receiveNow.WaitOne(timeout, false))
                    {
                        string t = port.ReadExisting();
                        buffer += t;
                    }
                    else
                    {
                       if (buffer.Length > 0)
                            throw new ApplicationException("Response received is incomplete.");
else
                            throw new ApplicationException("No data received from phone.");
                    }
                }
                while (!buffer.EndsWith("\r\nOK\r\n") && !buffer.EndsWith("\r\n>") && !buffer.EndsWith("\r\nERROR\r\n"));
            }
            catch (Exception ex)
            {
                throw ex;
            }
            return buffer;
        }

        #region Count SMS
        public int CountSMSmessages(SerialPort port)
        {
            int CountTotalMessages = 0;
            try
            {
                #region Execute Command
                string recievedData = ExecCommand(port, "AT", 300, "No phone connected at ");
                recievedData = ExecCommand(port, "AT+CMGF=1", 300, "Failed to set message format.");
                String command = "AT+CPMS?";
                recievedData = ExecCommand(port, command, 1000, "Failed to count SMS message");
                int uReceivedDataLength = recievedData.Length;
                #endregion

                #region If command is executed successfully
                if ((recievedData.Length >= 45) && (recievedData.StartsWith("AT+CPMS?")))
                {

                    #region Parsing SMS
                    string[] strSplit = recievedData.Split(',');
                    string strMessageStorageArea1 = strSplit[0]; //SM
                    string strMessageExist1 = strSplit[1];  //Msgs exist in SM
                    #endregion

                    #region Count Total Number of SMS In SIM
                    CountTotalMessages = Convert.ToInt32(strMessageExist1);
                    #endregion
                }
                #endregion

                #region If command is not executed successfully
                else if (recievedData.Contains("ERROR"))
                {
                    #region Error in Counting total number of SMS
                    string recievedError = recievedData;
                    recievedError = recievedError.Trim();
                    recievedData = "Following error occured while counting the message" + recievedError;
                    #endregion

                }
                #endregion
                return CountTotalMessages;
            }
            catch (Exception ex)
            {
                throw ex;
            }
        }
        #endregion

        #region Read SMS
        public AutoResetEvent receiveNow;
        public ShortMessageCollection
```

```
ReadSMS(SerialPort port, string p_strCommand)
        {
// Set up the phone and read the messages
            ShortMessageCollection messages =
null;
            try
            {
                #region Execute Command
// Check connection
                ExecCommand(port,"AT",    300,
"No phone connected");
// Use message format "Text mode"
                ExecCommand(port,"AT+CMGF=1",
300, "Failed to set message format.");
// Use character set "PCCP437"
ExecCommand(port,"AT+CSCS=\"PCCP437\"",   300,
"Failed to set character set.");
// Select SIM storage
ExecCommand(port,"AT+CPMS=\"SM\"",        300,
"Failed to select message storage.");
// Read the messages
                string         input         =
ExecCommand(port, p_strCommand, 5000, "Failed
to read the messages.");
                #endregion

                #region Parse messages
                messages                      =
ParseMessages(input);
                #endregion
            }
            catch (Exception ex)
            {
                throw ex;
            }
            if (messages != null)
                return messages;
            else
                return null;
        }
        public          ShortMessageCollection
ParseMessages(string input)
        {
            ShortMessageCollection messages =
new ShortMessageCollection();
            try
            {
                Regex r = new Regex(@"\+CMGL:
(\d+),""(.+)"",""(.+)"",(.*),""(.+)""\r\n(.+)
\r\n");
                Match m = r.Match(input);
                while (m.Success)
                {
                    ShortMessage   msg   =   new
ShortMessage();
                    //msg.Index                =
int.Parse(m.Groups[1].Value);
                  msg.Index = m.Groups[1].Value;
                 msg.Status = m.Groups[2].Value;
                msg.Sender = m.Groups[3].Value;
               msg.Alphabet = m.Groups[4].Value;
             msg.Sent = m.Groups[5].Value;
              msg.Message = m.Groups[6].Value;
                    messages.Add(msg);
                    m = m.NextMatch();
                }
            }
            catch (Exception ex)
            {
                throw ex;
            }
            return messages;
        }
        #endregion

        #region Send SMS
         static AutoResetEvent readNow = new
AutoResetEvent(false);
        public bool sendMsg(SerialPort port,
string PhoneNo, string Message)
        {
            bool isSend = false;
            try
            {
                string      recievedData      =
ExecCommand(port,"AT",    300,    "No    phone
connected");
                recievedData                  =
ExecCommand(port,"AT+CMGF=1", 300, "Failed to
set message format.");
                String command = "AT+CMGS=\""
+ PhoneNo + "\"";
                recievedData                  =
ExecCommand(port,command,   300,   "Failed   to
accept phoneNo");
                command     =     Message     +
char.ConvertFromUtf32(26) + "\r";
                recievedData                  =
ExecCommand(port,command,   3000,   "Failed   to
send message"); //3 seconds
                if
(recievedData.EndsWith("\r\nOK\r\n"))
                {
                    isSend = true;
                }
                else                          if
(recievedData.Contains("ERROR"))
                {
                    isSend = false;
                }
                return isSend;
            }
            catch (Exception ex)
            {
                throw ex;
            }
        }
        static    void    DataReceived(object
sender, SerialDataReceivedEventArgs e)
        {
            try
            {
                if       (e.EventType       ==
SerialData.Chars)
                    readNow.Set();
            }
            catch (Exception ex)
            {
                throw ex;
            }
        }
        #endregion

        #region Delete SMS
        public bool DeleteMsg(SerialPort port
, string p_strCommand)
        {
            bool isDeleted = false;
            try
            {
                #region Execute Command
                string      recievedData      =
ExecCommand(port,"AT",    300,    "No    phone
connected");
                recievedData                  =
ExecCommand(port,"AT+CMGF=1", 300, "Failed to
set message format.");
                String        command        =
p_strCommand;
                recievedData                  =
ExecCommand(port,command,   300,   "Failed   to
delete message");
                #endregion

                if
(recievedData.EndsWith("\r\nOK\r\n"))
                {
                    isDeleted = true;
                }
```

```
                if
(recievedData.Contains("ERROR"))
                {
                    isDeleted = false;
                }
                return isDeleted;
            }
            catch (Exception ex)
            {
                throw ex;
            }
        }
        #endregion
    }
}
```

```
//***Program.Cs***//
using System;
using System.Collections.Generic;
using System.Windows.Forms;

namespace SMSapplication
{
    static class Program
    {
        /// <summary>
        ///  The  main  entry  point  for  the
application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault
(false);
            Application.Run(new
SMSapplication());
        }
    }
}
```

```
//***ShortMessage.Cs***//
using System;
using System.Collections.Generic;
using System.Text;

namespace SMSapplication
{
    public class ShortMessage
    {

        #region Private Variables
        private string index;
                private string status;
                private string sender;
                private string alphabet;
                private string sent;
                private string message;
        #endregion
        #region Public Properties
        public string Index
                {
                        get { return index;}
                        set { index = value;}
                }
                public string Status
                {
                        get { return status;}
                        set { status = value;}
                }
                public string Sender
                {
                        get { return sender;}
                        set { sender = value;}
                }
                public string Alphabet
                {
                        get { return alphabet;}
                        set    {    alphabet    =
value;}
                }
                public string Sent
                {
                        get { return sent;}
                        set { sent = value;}
                }
                public string Message
                {
                        get { return message;}
                        set { message = value;}
                }
        #endregion
    }

    public   class   ShortMessageCollection   :
List<ShortMessage>
    {
    }
}
```

```
//***ShortMessageCollection.Cs***//
using System;
using System.Collections.Generic;
using System.Text;

namespace SMSapplication
{
}
```

LAMPIRAN V

DATASHEET KOMPONEN

1. **DATASHEET MIKROKONTROLER ATMEGA162**
2. **DATASHEET GPS SKYNAV SKM53**
3. **DATASHEET TRANSISTOR 2N3904**

Features

- High-performance, Low-power AVR® 8-bit Microcontroller
- Advanced RISC Architecture
  - 131 Powerful Instructions – Most Single-clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 16 MIPS Throughput at 16 MHz
  - On-chip 2-cycle Multiplier
- Non-volatile Program and Data Memories
  - 16K Bytes of In-System Self-programmable Flash
    Endurance: 10,000 Write/Erase Cycles
  - Optional Boot Code Section with Independent Lock Bits
    In-System Programming by On-chip Boot Program
    True Read-While-Write Operation
  - 512 Bytes EEPROM
    Endurance: 100,000 Write/Erase Cycles
  - 1K Bytes Internal SRAM
  - Up to 64K Bytes Optional External Memory Space
  - Programming Lock for Software Security
- JTAG (IEEE std. 1149.1 Compliant) Interface
  - Boundary-scan Capabilities According to the JTAG Standard
  - Extensive On-chip Debug Support
  - Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes
  - Two 16-bit Timer/Counters with Separate Prescalers, Compare Modes, and Capture Modes
  - Real Time Counter with Separate Oscillator
  - Six PWM Channels
  - Dual Programmable Serial USARTs
  - Master/Slave SPI Serial Interface
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated RC Oscillator
  - External and Internal Interrupt Sources
  - Five Sleep Modes: Idle, Power-save, Power-down, Standby, and Extended Standby
- I/O and Packages
  - 35 Programmable I/O Lines
  - 40-pin PDIP, 44-lead TQFP, and 44-pad MLF
- Operating Voltages
  - 1.8 - 5.5V for ATmega162V
  - 2.7 - 5.5V for ATmega162
- Speed Grades
  - 0 - 8 MHz for ATmega162V (see Figure 113 on page 268)
  - 0 - 16 MHz for ATmega162 (see Figure 114 on page 268)

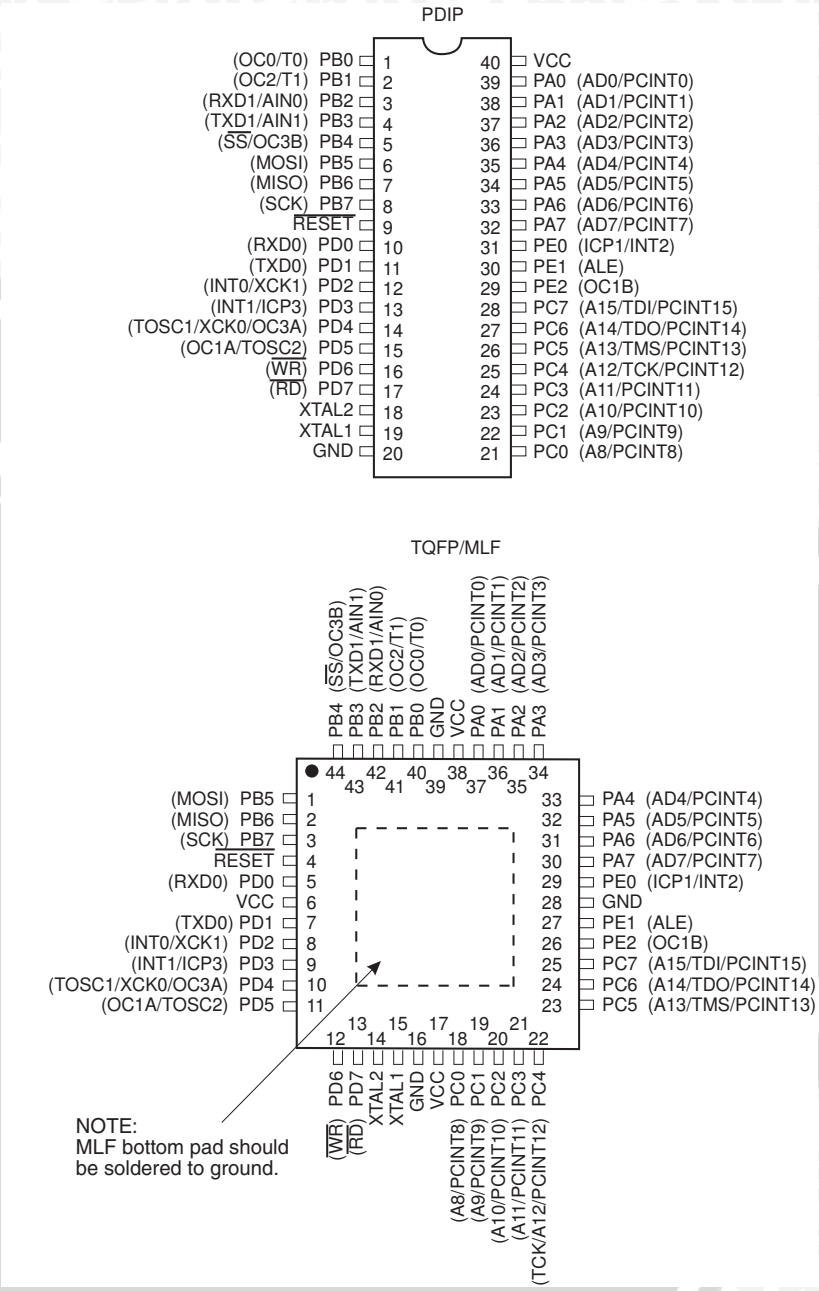
8-bit AVR® Microcontroller with 16K Bytes In-System Programmable Flash

ATmega162
ATmega162V

2513H–AVR–04/06

Pin Configurations

Figure 1. Pinout ATmega162



Disclaimer

Typical values contained in this datasheet are based on simulations and characterization of other AVR microcontrollers manufactured on the same process technology. Min and Max values will be available after the device is characterized.

Clock Sources

The device has the following clock source options, selectable by Flash Fuse bits as shown below. The clock from the selected source is input to the AVR clock generator, and routed to the appropriate modules.

Table 5. Device Clocking Options Select

Device Clocking Option	CKSEL3..0
External Crystal/Ceramic Resonator	1111 - 1000
External Low-frequency Crystal	0111 - 0100
Calibrated Internal RC Oscillator	0010
External Clock	0000
Reserved	0011, 0001

Note: For all fuses “1” means unprogrammed while “0” means programmed.

The various choices for each clocking option is given in the following sections. When the CPU wakes up from Power-down or Power-save, the selected clock source is used to time the start-up, ensuring stable Oscillator operation before instruction execution starts. When the CPU starts from Reset, there is an additional delay allowing the power to reach a stable level before commencing normal operation. The Watchdog Oscillator is used for timing this realtime part of the start-up time. The number of WDT Oscillator cycles used for each Time-out is shown in Table 6. The frequency of the Watchdog Oscillator is voltage dependent as shown in “ATmega162 Typical Characteristics” on page 277.

Table 6. Number of Watchdog Oscillator Cycles

Typ Time-out (V_{CC} = 5.0V)	Typ Time-out (V_{CC} = 3.0V)	Number of Cycles
4.1 ms	4.3 ms	4K (4,096)
65 ms	69 ms	64K (65,536)

Default Clock Source

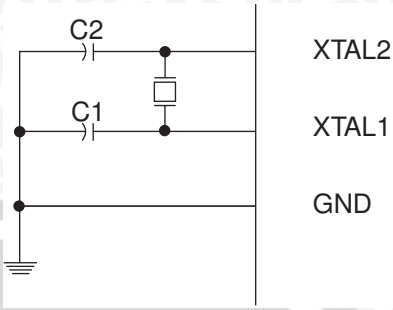
The device is shipped with CKSEL = “0010”, SUT = “10” and CKDIV8 programmed. The default clock source setting is therefore the Internal RC Oscillator with longest startup time and an initial system clock prescaling of 8. This default setting ensures that all users can make their desired clock source setting using an In-System or Parallel programmer.

Crystal Oscillator

XTAL1 and XTAL2 are input and output, respectively, of an inverting amplifier which can be configured for use as an On-chip Oscillator, as shown in Figure 19. Either a quartz crystal or a ceramic resonator may be used.

C1 and C2 should always be equal for both crystals and resonators. The optimal value of the capacitors depends on the crystal or resonator in use, the amount of stray capacitance, and the electromagnetic noise of the environment. Some initial guidelines for choosing capacitors for use with crystals are given in Table 7. For ceramic resonators, the capacitor values given by the manufacturer should be used.

Figure 19. Crystal Oscillator Connections



The Oscillator can operate in four different modes, each optimized for a specific frequency range. The operating mode is selected by the fuses CKSEL3:1 as shown in Table 7.

Table 7. Crystal Oscillator Operating Modes

CKSEL3:1	Frequency Range (MHz)	Recommended Range for Capacitors C1 and C2 for Use with Crystals (pF)
100(1)	0.4 - 0.9	–
101	0.9 - 3.0	12 - 22
110	3.0 - 8.0	12 - 22
111	8.0 -	12 - 22

Note: 1. This option should not be used with crystals, only with ceramic resonators.

The CKSEL0 Fuse together with the SUT1..0 Fuses select the start-up times as shown in Table 8.

Table 8. Start-up Times for the Crystal Oscillator Clock Selection

CKSEL0	SUT1:0	Start-up Time from Power-down and Power-save	Additional Delay from Reset (V_{CC} = 5.0V)	Recommended Usage
0	00	258 CK(1)	4.1 ms	Ceramic resonator, fast rising power
0	01	258 CK(1)	65 ms	Ceramic resonator, slowly rising power
0	10	1K CK(2)	–	Ceramic resonator, BOD enabled
0	11	1K CK(2)	4.1 ms	Ceramic resonator, fast rising power
1	00	1K CK(2)	65 ms	Ceramic resonator, slowly rising power
1	01	16K CK	–	Crystal Oscillator, BOD enabled
1	10	16K CK	4.1 ms	Crystal Oscillator, fast rising power
1	11	16K CK	65 ms	Crystal Oscillator, slowly rising power

System Control and Reset

Resetting the AVR

During Reset, all I/O Registers are set to their initial values, and the program starts execution from the Reset Vector. The instruction placed at the Reset Vector must be a JMP – Absolute Jump – instruction to the reset handling routine. If the program never enables an interrupt source, the Interrupt Vectors are not used, and regular program code can be placed at these locations. This is also the case if the Reset Vector is in the Application section while the Interrupt Vectors are in the Boot section or vice versa. The circuit diagram in Figure 21 shows the Reset Logic. Table 18 defines the electrical parameters of the reset circuitry.

The I/O ports of the AVR are immediately reset to their initial state when a reset source goes active. This does not require any clock source to be running.

After all reset sources have gone inactive, a delay counter is invoked, stretching the Internal Reset. This allows the power to reach a stable level before normal operation starts. The Time-out period of the delay counter is defined by the user through the CKSEL Fuses. The different selections for the delay period are presented in “Clock Sources” on page 36.

Reset Sources

The ATmega162 has five sources of reset:

- Power-on Reset. The MCU is reset when the supply voltage is below the Power-on Reset threshold (V_{POT}).
- External Reset. The MCU is reset when a low level is present on the $\overline{\text{RESET}}$ pin for longer than the minimum pulse length.
- Watchdog Reset. The MCU is reset when the Watchdog Timer period expires and the Watchdog is enabled.
- Brown-out Reset. The MCU is reset when the supply voltage V_{CC} is below the Brown-out Reset threshold (V_{BOT}) and the Brown-out Detector is enabled. The device is guaranteed to operate at maximum frequency for the V_{CC} voltage down to V_{BOT}. V_{BOT} must be set to the corresponding minimum voltage of the device (i.e., minimum V_{BOT} for ATmega162V is 1.8V).
- JTAG AVR Reset. The MCU is reset as long as there is a logic one in the Reset Register, one of the scan chains of the JTAG system. Refer to the section “IEEE 1149.1 (JTAG) Boundary-scan” on page 206 for details.

Figure 21. Reset Logic

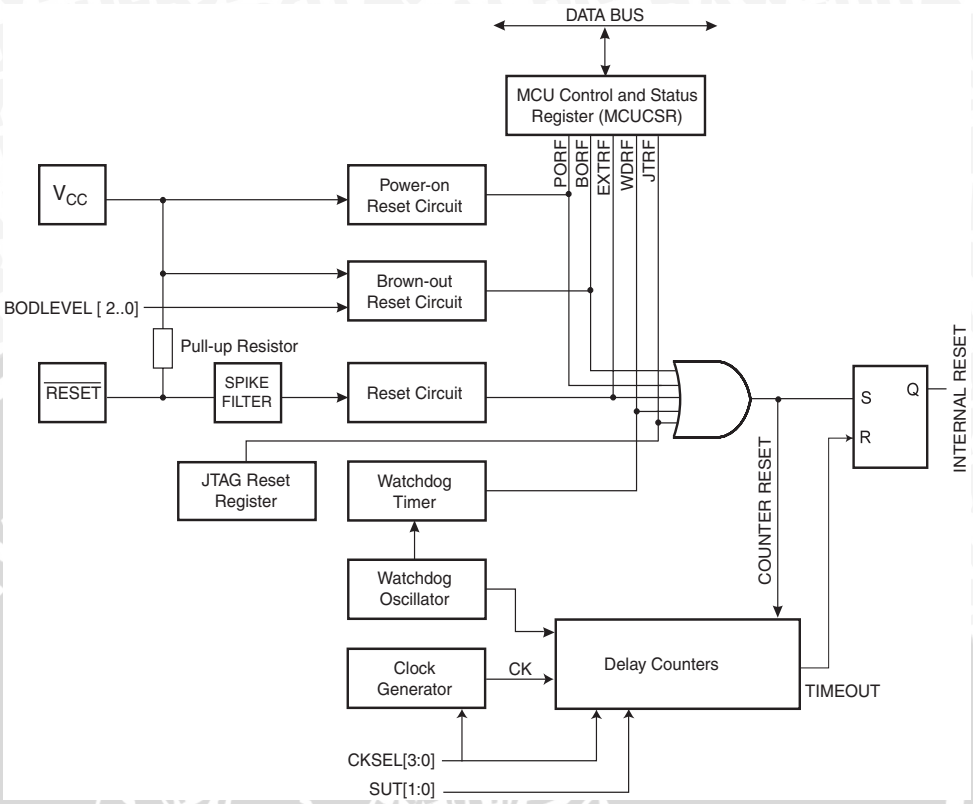


Table 18. Reset Characteristics

Symbol	Parameter	Condition	Min.	Typ.	Max.	Units
V_{POT}	Power-on Reset Threshold Voltage (rising)	T_A = -40 - 85°C	0.7	1.0	1.4	V
	Power-on Reset Threshold Voltage (falling)(1)	T_A = -40 - 85°C	0.6	0.9	1.3	V
V_{RST}	$\overline{\text{RESET}}$ Pin Threshold Voltage	V_{CC} = 3V	0.1 V_{CC}		0.9 V_{CC}	V
t_{RST}	Minimum pulse width on $\overline{\text{RESET}}$ Pin	V_{CC} = 3V			2.5	µs

Note: 1. The Power-on Reset will not work unless the supply voltage has been below V_{POT} (falling)

Power-on Reset

A Power-on Reset (POR) pulse is generated by an On-chip detection circuit. The detection level is defined in Table 18. The POR is activated whenever V_{CC} is below the detection level. The POR circuit can be used to trigger the Start-up Reset, as well as to detect a failure in supply voltage.

A Power-on Reset (POR) circuit ensures that the device is Reset from Power-on. Reaching the Power-on Reset threshold voltage invokes the delay counter, which determines how long the device is kept in RESET after V_{CC} rise. The RESET signal is activated again, without any delay, when V_{CC} decreases below the detection level.

Figure 22. MCU Start-up, $\overline{\text{RESET}}$ Tied to V_{CC}.

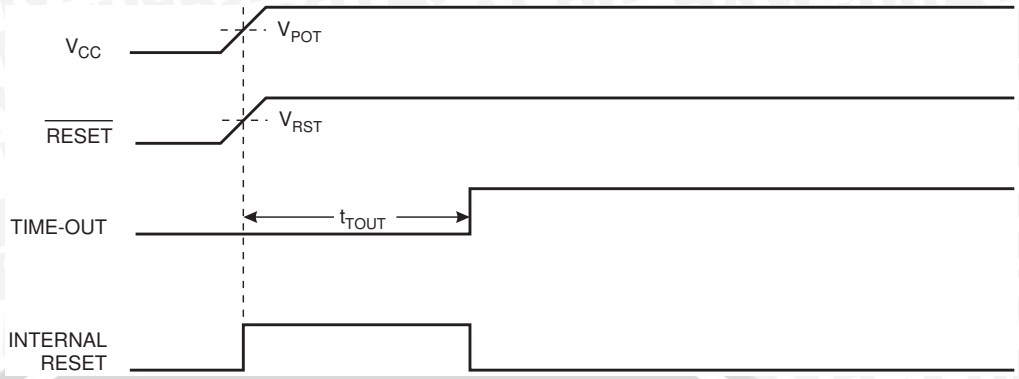
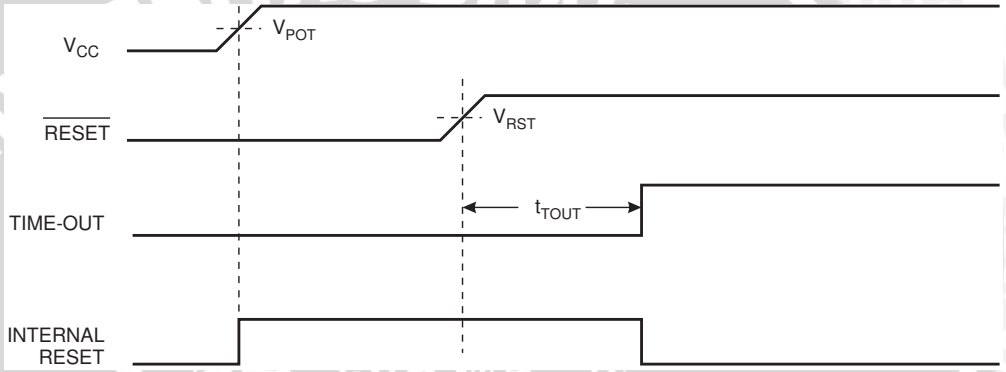


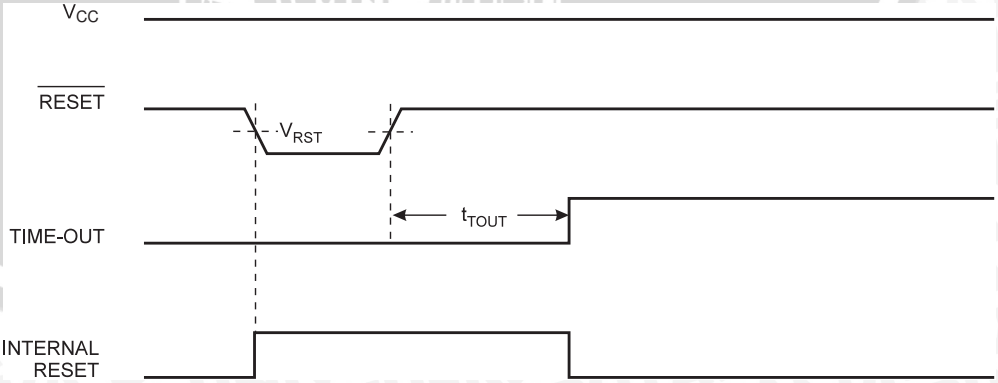
Figure 23. MCU Start-up, $\overline{\text{RESET}}$ Extended Externally



External Reset

An External Reset is generated by a low level on the $\overline{\text{RESET}}$ pin. Reset pulses longer than the minimum pulse width (see Table 18) will generate a Reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a Reset. When the applied signal reaches the Reset Threshold Voltage – V_{RST} on its positive edge, the delay counter starts the MCU after the Time-out period t_{TOUT} has expired.

Figure 24. External Reset During Operation



USART

The Universal Synchronous and Asynchronous serial Receiver and Transmitter (USART) is a highly flexible serial communication device. The main features are:

- **Full Duplex Operation (Independent Serial Receive and Transmit Registers)**
- **Asynchronous or Synchronous Operation**
- **Master or Slave Clocked Synchronous Operation**
- **High Resolution Baud Rate Generator**
- **Supports Serial Frames with 5, 6, 7, 8, or 9 Data Bits and 1 or 2 Stop Bits**
- **Odd or Even Parity Generation and Parity Check Supported by Hardware**
- **Data OverRun Detection**
- **Framing Error Detection**
- **Noise Filtering Includes False Start Bit Detection and Digital Low Pass Filter**
- **Three Separate Interrupts on TX Complete, TX Data Register Empty and RX Complete**
- **Multi-processor Communication Mode**
- **Double Speed Asynchronous Communication Mode**

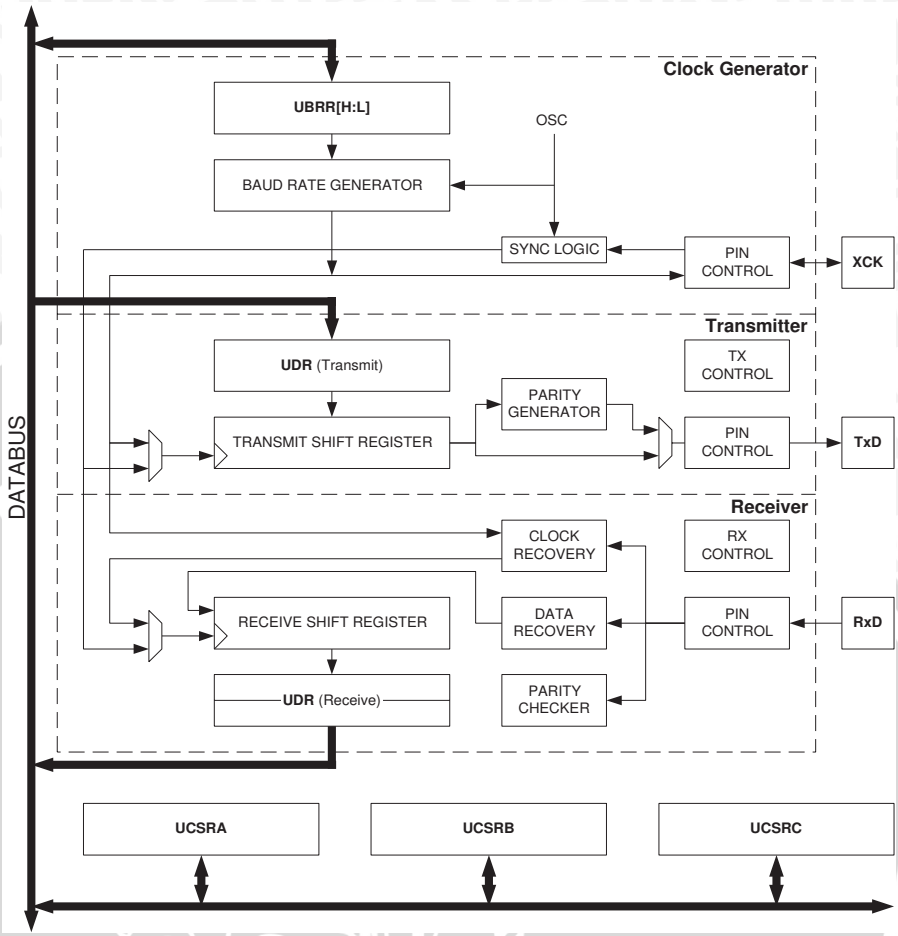
Dual USART

The ATmega162 has two USARTs, USART0 and USART1. The functionality for both USARTs is described below.

USART0 and USART1 have different I/O Registers as shown in “Register Summary” on page 306. Note that in ATmega161 compatibility mode, the double buffering of the USART Receive Register is disabled. For details, see “AVR USART vs. AVR UART – Compatibility” on page 170. Note also that the shared UBRRHI Register in ATmega161 has been split into two separate registers, UBRR0H and UBRR1H, in ATmega162.

A simplified block diagram of the USART Transmitter is shown in Figure 75. CPU accessible I/O Registers and I/O pins are shown in bold.

Figure 75. USART Block Diagram[1]



Note: 1. Refer to Figure 1 on page 2, Table 34 on page 75, Table 39 on page 81, and Table 40 on page 81 for USART pin placement.

The dashed boxes in the block diagram separate the three main parts of the USART (listed from the top): Clock Generator, Transmitter and Receiver. Control registers are shared by all units. The Clock Generation logic consists of synchronization logic for external clock input used by synchronous slave operation, and the baud rate generator. The XCK (Transfer Clock) pin is only used by synchronous transfer mode. The Transmitter consists of a single write buffer, a serial Shift Register, parity generator and control logic for handling different serial frame formats. The write buffer allows a continuous transfer of data without any delay between frames. The Receiver is the most complex part of the USART module due to its clock and data recovery units. The recovery units are used for asynchronous data reception. In addition to the recovery units, the Receiver includes a Parity Checker, Control logic, a Shift Register and a two level receive buffer (UDR). The receiver supports the same frame formats as the Transmitter, and can detect Frame Error, Data OverRun and Parity Errors.

AVR USART vs. AVR UART – Compatibility

The USART is fully compatible with the AVR UART regarding:

- Bit locations inside all USART Registers
- Baud Rate Generation
- Transmitter Operation
- Transmit Buffer Functionality
- Receiver Operation

However, the receive buffering has two improvements that will affect the compatibility in some special cases:

- A second Buffer Register has been added. The two buffer registers operate as a circular FIFO buffer. Therefore the UDR must only be read once for each incoming data! More important is the fact that the Error Flags (FE and DOR) and the ninth data bit (RXB8) are buffered with the data in the receive buffer. Therefore the status bits must always be read before the UDR Register is read. Otherwise the error status will be lost since the buffer state is lost.
- The Receiver Shift Register can now act as a third buffer level. This is done by allowing the received data to remain in the serial Shift Register (see Figure 75) if the Buffer Registers are full, until a new start bit is detected. The USART is therefore more resistant to Data OverRun (DOR) error conditions.

The following control bits have changed name, but have same functionality and register location:

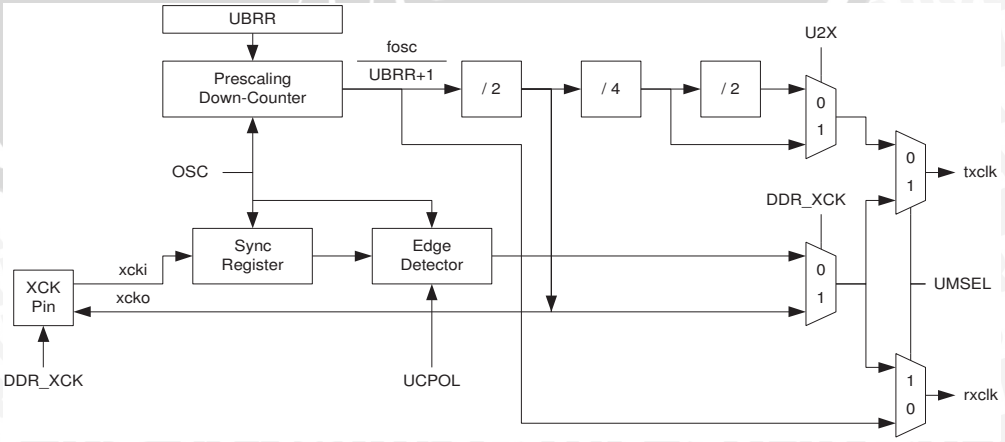
- CHR9 is changed to UCSZ2.
- OR is changed to DOR.

Clock Generation

The Clock Generation logic generates the base clock for the Transmitter and Receiver. The USART supports four modes of clock operation: Normal asynchronous, Double Speed asynchronous, Master synchronous and Slave synchronous mode. The UMSEL bit in USART Control and Status Register C (UCSRC) selects between asynchronous and synchronous operation. Double Speed (asynchronous mode only) is controlled by the U2X found in the UCSRA Register. When using synchronous mode (UMSEL = 1), the Data Direction Register for the XCK pin (DDR_XCK) controls whether the clock source is internal (Master mode) or external (Slave mode). The XCK pin is only active when using synchronous mode.

Figure 76 shows a block diagram of the clock generation logic.

Figure 76. Clock Generation Logic, Block Diagram



Signal description:

txclk Transmitter clock. (Internal Signal)

rxclk Receiver base clock. (Internal Signal)

xcki Input from XCK pin (internal Signal). Used for synchronous slave operation.

xcko Clock output to XCK pin (Internal Signal). Used for synchronous master operation.

fosc XTAL pin frequency (System Clock).

Internal Clock Generation – The Baud Rate Generator

Internal clock generation is used for the asynchronous and the synchronous master modes of operation. The description in this section refers to Figure 76.

The USART Baud Rate Register (UBRR) and the down-counter connected to it function as a programmable prescaler or baud rate generator. The down-counter, running at system clock (fosc), is loaded with the UBRR value each time the counter has counted down to zero or when the UBRRL Register is written. A clock is generated each time the counter reaches zero. This clock is the baud rate generator clock output (= fosc/(UBRR+1)). The Transmitter divides the baud rate generator clock output by 2, 8 or 16 depending on mode. The baud rate generator output is used directly by the receiver's clock and data recovery units. However, the recovery units use a state machine that uses 2, 8 or 16 states depending on mode set by the state of the UMSEL, U2X and DDR_XCK bits.

Table 70 contains equations for calculating the baud rate (in bits per second) and for calculating the UBRR value for each mode of operation using an internally generated clock source.

Table 70. Equations for Calculating Baud Rate Register Setting

Operating Mode	Equation for Calculating Baud Rate[1]	Equation for Calculating UBRR Value
Asynchronous Normal Mode (U2X = 0)	$BAUD = \frac{f_{OSC}}{16(UBRR+1)}$	$UBRR = \frac{f_{OSC}}{16BAUD} - 1$
Asynchronous Double Speed Mode (U2X = 1)	$BAUD = \frac{f_{OSC}}{8(UBRR+1)}$	$UBRR = \frac{f_{OSC}}{8BAUD} - 1$
Synchronous Master Mode	$BAUD = \frac{f_{OSC}}{2(UBRR+1)}$	$UBRR = \frac{f_{OSC}}{2BAUD} - 1$

Note: 1. The baud rate is defined to be the transfer rate in bit per second (bps).

BAUD Baud rate (in bits per second, bps)

f_{osc} System Oscillator clock frequency

UBRR Contents of the UBRRH and UBRRL Registers, (0 - 4095)

Some examples of UBRR values for some system clock frequencies are found in Table 78 (see page 193).

Double Speed Operation (U2X)

The transfer rate can be doubled by setting the U2X bit in UCSRA. Setting this bit only has effect for the asynchronous operation. Set this bit to zero when using synchronous operation.

Setting this bit will reduce the divisor of the baud rate divider from 16 to 8, effectively doubling the transfer rate for asynchronous communication. Note however that the Receiver will in this case only use half the number of samples (reduced from 16 to 8) for data sampling and clock recovery, and therefore a more accurate baud rate setting and system clock are required when this mode is used. For the Transmitter, there are no downsides.

External Clock

External clocking is used by the synchronous slave modes of operation. The description in this section refers to Figure 76 for details.

External clock input from the XCK pin is sampled by a synchronization register to minimize the chance of meta-stability. The output from the synchronization register must then pass through an edge detector before it can be used by the Transmitter and Receiver. This process introduces a two CPU clock period delay and therefore the maximum external XCK clock frequency is limited by the following equation:

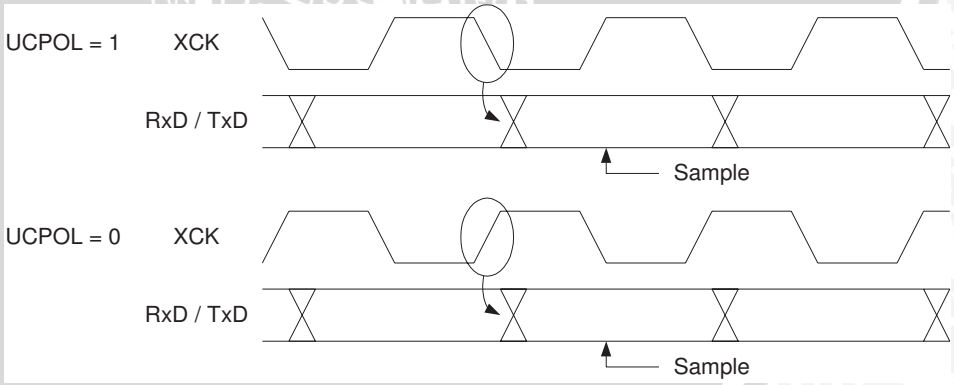
$$f_{XCK} < \frac{f_{OSC}}{4}$$

Note that f_{osc} depends on the stability of the system clock source. It is therefore recommended to add some margin to avoid possible loss of data due to frequency variations.

Synchronous Clock Operation

When synchronous mode is used (UMSEL = 1), the XCK pin will be used as either clock input (Slave) or clock output (Master). The dependency between the clock edges and data sampling or data change is the same. The basic principle is that data input (on RxD) is sampled at the opposite XCK clock edge of the edge the data output (TxD) is changed.

Figure 77. Synchronous Mode XCK Timing.



The UCPOL bit UCRSC selects which XCK clock edge is used for data sampling and which is used for data change. As Figure 77 shows, when UCPOL is zero the data will be changed at rising XCK edge and sampled at falling XCK edge. If UCPOL is set, the data will be changed at falling XCK edge and sampled at rising XCK edge.

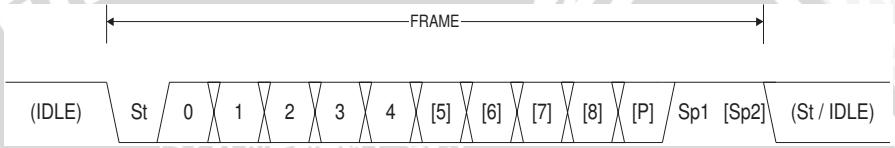
Frame Formats

A serial frame is defined to be one character of data bits with synchronization bits (start and stop bits), and optionally a parity bit for error checking. The USART accepts all 30 combinations of the following as valid frame formats:

- 1 start bit
- 5, 6, 7, 8, or 9 data bits
- no, even or odd parity bit
- 1 or 2 stop bits

A frame starts with the start bit followed by the least significant data bit. Then the next data bits, up to a total of nine, are succeeding, ending with the most significant bit. If enabled, the parity bit is inserted after the data bits, before the stop bits. When a complete frame is transmitted, it can be directly followed by a new frame, or the communication line can be set to an idle (high) state. Figure 78 illustrates the possible combinations of the frame formats. Bits inside brackets are optional.

Figure 78. Frame Formats



St Start bit, always low.

(n) Data bits (0 to 8).

P Parity bit. Can be odd or even.

Sp Stop bit, always high.

IDLE No transfers on the communication line (RxD or TxD). An IDLE line must be high.

The frame format used by the USART is set by the UCSZ2:0, UPM1:0 and USBS bits in UCSRB and UCSRC. The Receiver and Transmitter use the same setting. Note that changing the setting of any of these bits will corrupt all ongoing communication for both the Receiver and Transmitter.

The USART Character SiZe (UCSZ2:0) bits select the number of data bits in the frame. The USART Parity mode (UPM1:0) bits enable and set the type of parity bit. The selection between one or two stop bits is done by the USART Stop Bit Select (USBS) bit. The receiver ignores the second stop bit. An FE (Frame Error) will therefore only be detected in the cases where the first stop bit is zero.

Parity Bit Calculation

The parity bit is calculated by doing an exclusive-or of all the data bits. If odd parity is used, the result of the exclusive or is inverted. The relation between the parity bit and data bits is as follows::

$$P_{even} = d_{n-1} \oplus \ldots \oplus d_3 \oplus d_2 \oplus d_1 \oplus d_0 \oplus 0$$
$$P_{odd} = d_{n-1} \oplus \ldots \oplus d_3 \oplus d_2 \oplus d_1 \oplus d_0 \oplus 1$$

P_{even} Parity bit using even parity

P_{odd} Parity bit using odd parity

d_n Data bit n of the character

If used, the parity bit is located between the last data bit and first stop bit of a serial frame.

USART Initialization

The USART has to be initialized before any communication can take place. The initialization process normally consists of setting the baud rate, setting frame format and enabling the Transmitter or the Receiver depending on the usage. For interrupt driven USART operation, the Global Interrupt Flag should be cleared (and interrupts globally disabled) when doing the initialization.

Before doing a re-initialization with changed baud rate or frame format, be sure that there are no ongoing transmissions during the period the registers are changed. The TXC Flag can be used to check that the Transmitter has completed all transfers, and the RXC Flag can be used to check that there are no unread data in the receive buffer. Note that the TXC Flag must be cleared before each transmission (before UDR is written) if it is used for this purpose.

The following simple USART initialization code examples show one assembly and one C function that are equal in functionality. The examples assume asynchronous operation using polling (no interrupts enabled) and a fixed frame format. The baud rate is given as a function parameter. For the assembly code, the baud rate parameter is assumed to be stored in the r17:r16 Registers. When the function writes to the UCSRC Register, the URSEL bit (MSB) must be set due to the sharing of I/O location by UBRRH and UCSRC.

Assembly Code Example[1]

```
USART_Init:
  ; Set baud rate
  out   UBRRH, r17
  out   UBRRL, r16
  ; Enable receiver and transmitter
  ldi   r16, (1<<RXEN)|(1<<TXEN)
  out   UCSRB,r16
  ; Set frame format: 8data, 2stop bit
  ldi   r16, (1<<URSEL)|(1<<USBS)|(3<<UCSZ0)
  out   UCSRC,r16
  ret
```

C Code Example[1]

```
void USART_Init( unsigned int baud )
{
  /* Set baud rate */
  UBRRH = (unsigned char)(baud>>8);
  UBRRL = (unsigned char)baud;
  /* Enable receiver and transmitter */
  UCSRB = (1<<RXEN)|(1<<TXEN);
  /* Set frame format: 8data, 2stop bit */
  UCSRC = (1<<URSEL)|(1<<USBS)|(3<<UCSZ0);
}
```

Note: 1. The example code assumes that the part specific header file is included.

More advanced initialization routines can be made that include frame format as parameters, disable interrupts and so on. However, many applications use a fixed setting of the

Baud and Control Registers, and for these types of applications the initialization code can be placed directly in the main routine, or be combined with initialization code for other I/O modules.

Data Transmission – The USART Transmitter

The USART Transmitter is enabled by setting the *Transmit Enable* (TXEN) bit in the UCSRB Register. When the Transmitter is enabled, the normal port operation of the TxD pin is overridden by the USART and given the function as the transmitter's serial output. The baud rate, mode of operation and frame format must be set up once before doing any transmissions. If synchronous operation is used, the clock on the XCK pin will be overridden and used as transmission clock.

Sending Frames with 5 to 8 Data Bit

A data transmission is initiated by loading the transmit buffer with the data to be transmitted. The CPU can load the transmit buffer by writing to the UDR I/O location. The buffered data in the transmit buffer will be moved to the Shift Register when the Shift Register is ready to send a new frame. The Shift Register is loaded with new data if it is in idle state (no ongoing transmission) or immediately after the last stop bit of the previous frame is transmitted. When the Shift Register is loaded with new data, it will transfer one complete frame at the rate given by the Baud Register, U2X bit or by XCK depending on mode of operation.

The following code examples show a simple USART transmit function based on polling of the *Data Register Empty* (UDRE) Flag. When using frames with less than eight bits, the most significant bits written to the UDR are ignored. The USART has to be initialized before the function can be used. For the assembly code, the data to be sent is assumed to be stored in Register R16

Assembly Code Example[1]

```
USART_Transmit:
  ; Wait for empty transmit buffer
  sbis  UCSRA,UDRE
  rjmp  USART_Transmit
  ; Put data (r16) into buffer, sends the data
  out   UDR,r16
  ret
```

C Code Example[1]

```
void USART_Transmit( unsigned char data )
{
  /* Wait for empty transmit buffer */
  while ( !( UCSRA & (1<<UDRE)) )
        ;
  /* Put data into buffer, sends the data */
  UDR = data;
}
```

Note: 1. The example code assumes that the part specific header file is included.

The function simply waits for the transmit buffer to be empty by checking the UDRE Flag, before loading it with new data to be transmitted. If the Data Register Empty interrupt is utilized, the interrupt routine writes the data into the buffer.

Sending Frames with 9 Data Bit

If 9-bit characters are used (UCSZ = 7), the ninth bit must be written to the TXB8 bit in UCSRB before the low byte of the character is written to UDR. The following code examples show a transmit function that handles 9-bit characters. For the assembly code, the data to be sent is assumed to be stored in Registers R17:R16.

Assembly Code Example[1]

```
USART_Transmit:
  ; Wait for empty transmit buffer
  sbis  UCSRA,UDRE
  rjmp  USART_Transmit
  ; Copy 9th bit from r17 to TXB8
  cbi   UCSRB,TXB8
  sbrc  r17,0
  sbi   UCSRB,TXB8
  ; Put LSB data (r16) into buffer, sends the data
  out   UDR,r16
  ret
```

C Code Example[1]

```
void USART_Transmit( unsigned int data )
{
  /* Wait for empty transmit buffer */
  while ( !( UCSRA & (1<<UDRE)) )
        ;
  /* Copy 9th bit to TXB8 */
  UCSRB &= ~(1<<TXB8);
  if ( data & 0x0100 )
    UCSRB |= (1<<TXB8);
  /* Put data into buffer, sends the data */
  UDR = data;
}
```

Note: 1. These transmit functions are written to be general functions. They can be optimized if the contents of the UCSRB is static. For example, only the TXB8 bit of the UCSRB Register is used after initialization.

The ninth bit can be used for indicating an address frame when using multi processor communication mode or for other protocol handling as for example synchronization.

Transmitter Flags and Interrupts

The USART Transmitter has two flags that indicate its state: USART Data Register Empty (UDRE) and Transmit Complete (TXC). Both flags can be used for generating interrupts.

The Data Register Empty (UDRE) Flag indicates whether the transmit buffer is ready to receive new data. This bit is set when the transmit buffer is empty, and cleared when the transmit buffer contains data to be transmitted that has not yet been moved into the Shift Register. For compatibility with future devices, always write this bit to zero when writing the UCSRA Register.

When the Data Register Empty Interrupt Enable (UDRIE) bit in UCSRB is written to one, the USART Data Register Empty Interrupt will be executed as long as UDRE is set (provided that global interrupts are enabled). UDRE is cleared by writing UDR. When interrupt-driven data transmission is used, the Data Register Empty Interrupt routine must either write new data to UDR in order to clear UDRE or disable the Data Register

Empty Interrupt, otherwise a new interrupt will occur once the interrupt routine terminates.

The Transmit Complete (TXC) Flag bit is set one when the entire frame in the Transmit Shift Register has been shifted out and there are no new data currently present in the transmit buffer. The TXC Flag bit is automatically cleared when a transmit complete interrupt is executed, or it can be cleared by writing a one to its bit location. The TXC Flag is useful in half-duplex communication interfaces (like the RS-485 standard), where a transmitting application must enter Receive mode and free the communication bus immediately after completing the transmission.

When the Transmit Compete Interrupt Enable (TXCIE) bit in UCSRB is set, the USART Transmit Complete Interrupt will be executed when the TXC Flag becomes set (provided that global interrupts are enabled). When the transmit complete interrupt is used, the interrupt handling routine does not have to clear the TXC Flag, this is done automatically when the interrupt is executed.

Parity Generator

The Parity Generator calculates the parity bit for the serial frame data. When parity bit is enabled (UPM1 = 1), the transmitter control logic inserts the parity bit between the last data bit and the first stop bit of the frame that is sent.

Disabling the Transmitter

The disabling of the Transmitter (setting the TXEN to zero) will not become effective until ongoing and pending transmissions are completed, i.e., when the Transmit Shift Register and Transmit Buffer Register do not contain data to be transmitted. When disabled, the Transmitter will no longer override the TxD pin.

Data Reception – The USART Receiver

The USART Receiver is enabled by writing the Receive Enable (RXEN) bit in the UCSRB Register to one. When the receiver is enabled, the normal pin operation of the RxD pin is overridden by the USART and given the function as the receiver's serial input. The baud rate, mode of operation and frame format must be set up once before any serial reception can be done. If synchronous operation is used, the clock on the XCK pin will be used as transfer clock.

Receiving Frames with 5 to 8 Data Bits

The Receiver starts data reception when it detects a valid start bit. Each bit that follows the start bit will be sampled at the baud rate or XCK clock, and shifted into the Receive Shift Register until the first stop bit of a frame is received. A second stop bit will be ignored by the Receiver. When the first stop bit is received, i.e., a complete serial frame is present in the Receive Shift Register, the contents of the Shift Register will be moved into the receive buffer. The receive buffer can then be read by reading the UDR I/O location.

The following code example shows a simple USART receive function based on polling of the Receive Complete (RXC) Flag. When using frames with less than eight bits the most significant bits of the data read from the UDR will be masked to zero. The USART has to be initialized before the function can be used.

Assembly Code Example[1]

```
USART_Receive:
  ; Wait for data to be received
  sbis  UCSRA, RXC
  rjmp  USART_Receive
  ; Get and return received data from buffer
  in    r16, UDR
  ret
```

C Code Example[1]

```
unsigned char USART_Receive( void )
{
  /* Wait for data to be received */
  while ( !(UCSRA & (1<<RXC)) )
        ;
  /* Get and return received data from buffer */
  return UDR;
}
```

Note: 1. The example code assumes that the part specific header file is included.

The function simply waits for data to be present in the receive buffer by checking the RXC Flag, before reading the buffer and returning the value.

SkyNav SKM53 Series
Ultra High Sensitivity and Low Power
The Smart Antenna GPS Module

http://www.webtronico.com

General Description

The SkyNav SKM53 Series with embedded GPS antenna enables high performance navigation in the most stringent applications and solid fix even in harsh GPS visibility environments.

It is based on the high performance features of the MediaTek 3327 single-chip architecture, Its –165dBm tracking sensitivity extends positioning coverage into place like urban canyons and dense foliage environment where the GPS was not possible before. The 6-pin and USB connector design is the easiest and convenient solution to be embedded in a portable device and receiver like PND, GPS mouse, car holder, personal locator, speed camera detector and vehicle locator.

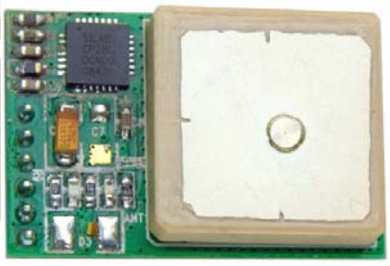


Figure 1: SKM53 series Top View

Applications

- LBS (Location Based Service)
- Vehicle navigation system
- PND (Portable Navigation Device)
- GPS mouse and Bluetooth GPS receiver
- Timing application

Features

- Ultra high sensitivity: -165dBm
- 22 tracking/66 acquisition-channel receiver
- WAAS/EGNOS/MSAS/GAGAN support
- NMEA protocols (default speed: 9600bps)
- Internal back-up battery and 1PPS output
- One serial port and USB port (option)
- Embedded patch antenna 18.2 x 18.2 x 4.0 mm
- Operating temperature range: -40 to 85℃
- RoHS compliant (Lead-free)
- Tiny form factor ： 30mm x20mm x 11.4mm

Pin Assignment

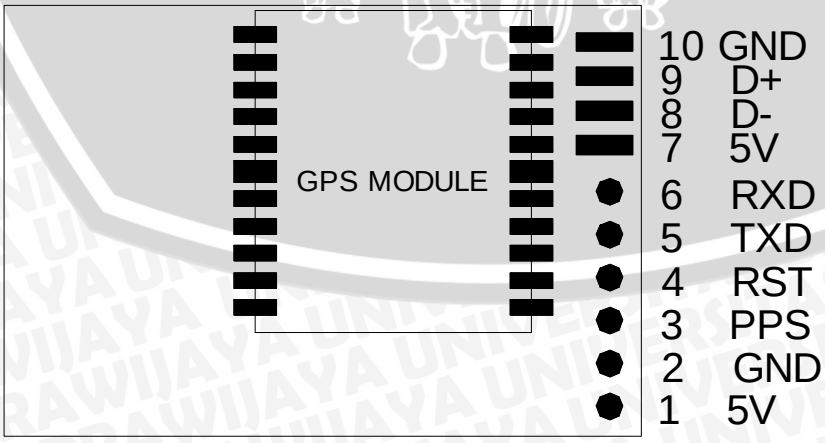


Figure 2: SKM53 Series Pin Package

Performance Specification

Parameter	Specification	
GPS receiver		
Receiver Type	L1 frequency band, C/A code, 22 Tracking / 66 Acquisition-Channel	
Sensitivity	Tracking Acquisition	-165dBm -148dBm
Accuracy	Position Velocity Timing (PPS)	3.0m 3D RMS without SA 0.1m/s without SA 60ns RMS
Acquisition Time	Cold Start Warm Start Hot Start Re-Acquisition	36s 33s 1s <1s
Power Consumption	Tracking Acquisition Sleep/Standby	<30mA @3.0V 40mA @3.0V TBD
Navigation Data Update Rate	1Hz	
Operational Limits	Altitude Velocity Acceleration	Max 18,000m Max 515m/s Less than 4g
Antenna Specifications		
Outline Dimension	18.2 x 18.2 x 4.0 mm	
Center Frequency	1575 ± 3 MHz	
Bandwidth	10 MHz min	
Impedance	50 Ω	
Axial Ratio	3 dB max	
Polarization	RHCP	
Mechanical requirements		
Dimension	30mm x20mm x 11.4mm	
Weight	9g	
Power consumption		
VCC	5V ±5%	
Current	55mA(typical)	
Environment		
Operating temperature	40 ~ +85 ℃ (w/o backup battery)	
Storage temperature	40 ~ +125 ℃	
Humidity	≦95%	

Hardware Interfaces Configuration

Power Supply: Regulated power for the SKM53 series is required. The input voltage Vcc should be 5V, current is no less than 150mA. Suitable decoupling must be provided by external decoupling circuitry(10uF and 1uF). It can reduce the Noise from power supply and increase power stability.

UART Ports: The module supports one full duplex serial channels UART. The serial connections are at 2.85V LVTTL logic levels, if need different voltage levels, use appropriate level shifters. the data format is however fixed: X, N, 8, 1, i.e. X baud rate, no parity, eight data bits and one stop bit, no other data formats are supported, LSB is sent first. The modules default baud rate is set up 9600bps. The RXD0 & TXD0 recommended to pull up (10KΩ). It can increase the stability of serial data.

USB Ports: The module uses single-chip USB to UART bridge by Silicon CP2102, It is a USB 2.0 compliant full-speed device with integrated transceiver. Before using it, please install the appropriate driver.

GPS Status: GPS Status can be connected to a LED to indicate the status of GPS signal. Lights indicate GPS not fix and flashing indicate fix.

Pin Description

Pin No.	Pin name	I/O	Description	Remark
UART Port				
1	5V	P	Module Power Supply	VCC:5V±5%
2	GND	G	Module Power Ground	Reference Ground
3	PPS	O	Time pulse Signal (Default 200ms pulse/sec)	Leave Open in not used
4	RST	I	Module Reset (Active Low Status)	
5	TXD	I	TTL:VOH≥0.75 *VDD VOL≤0.25VDD	Pull up if not used
6	RXD	O	TTL:VIH ≥0.7 *VDD VIL ≤0.3 *VDD	Leave Open in not used
USB Port				
7	5V	P	USB Power Supply	
8	D-	I\O	Data-	
9	D+	I\O	Data+	
10	GND	G	USB Power Supply	

Ordering Information

SKM53S：UART Port Interface
SKM53U：USB Port Interface

Mechanical Specification

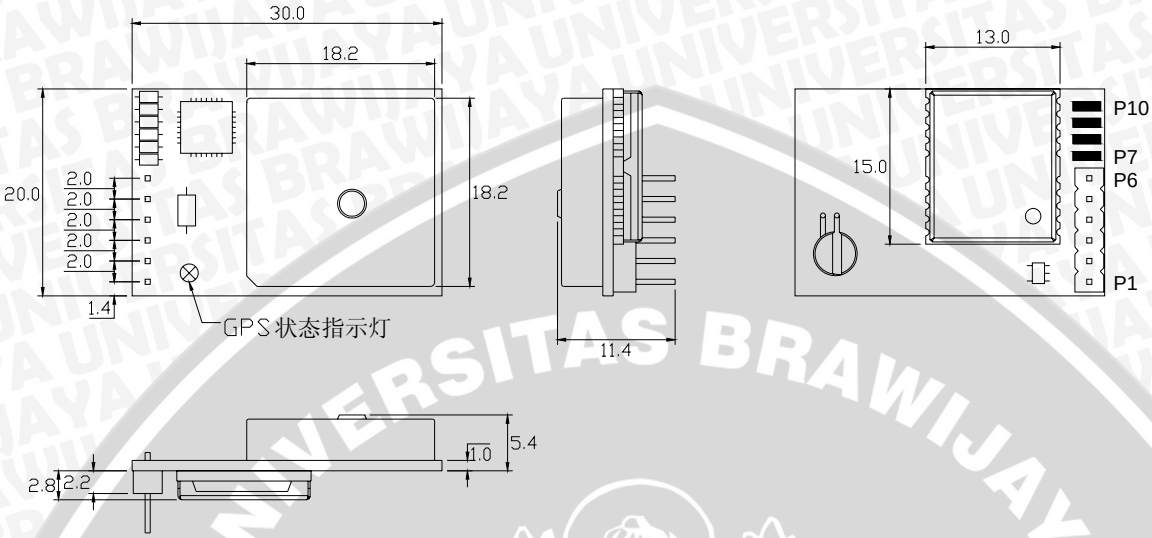


Figure 3: SKM53 Series Dimensions

Software Protocol

NMEA 0183 Protocol

The NMEA protocol is an ASCII-based protocol, Records start with a $ and with carriage return/line feed. GPS specific messages all start with $GPxxx where xxx is a three-letter identifier of the message data that follows. NMEA messages have a checksum, which allows detection of corrupted data transfers.

The SkyNav SKM53 series supports the following NMEA-0183 messages: GGA, GLL, GSA, GSV, RMC VTG, ZDA.

Table 1: NMEA-0183 Output Messages

NMEA Record	DESCRIPTION
GGA	Global positioning system fixed data
GLL	Geographic position—latitude/longitude
GSA	GNSS DOP and active satellites
GSV	GNSS satellites in view
RMC	Recommended minimum specific GNSS data
VTG	Course over ground and ground speed
ZDA	Time and Date

GGA-Global Positioning System Fixed Data

Table 2 contains the values of the following example:

$GPGGA, 083559.00,3723.2475,N, 12158.3416,W, 1,07,1.0,9.0,M, ,M, ,0000*18

Table 2: GGA Data Format

Name	Example	Units	Description
Message ID	$GPGGA		GGA protocol header
UTC Time	083559.00		hhmmss.sss
Latitude	3723.2457		ddmm.mmmm
N/S indicator	N		N=north or S=south
Longitude	12158.3416		ddmm.mmmm
E/W Indicator	W		E=east or W=west
Position Fix Indicator	1		See Table 2-1
Satellites Used	07		Range 00 to 12
HDOP	1.0		Horizontal Dilution of Precision
MSL Altitude	9.0	meters	Altitude above mean seal level
Units	M	meters	
Geoids Separation		meters	Separation from Geoids can be bank
Units	M	meters	
Age of Diff.Corr.		second	Null fields when DGPS is not Used
Diff.Ref.Station ID	0000		Null fields when DGPS is not Used
Checksum	*18		
<CR> <LF>			End of message termination(ASCII 13, ASCII 10)

Table 2-1: Position Fix Indicators

Value	Description
0	Fix not available or invalid
1	GPS SPS Mode, fix valid
2	Differential GPS, SPS Mode, fix valid
3	GPS PPS Mode, fix valid

GLL-Geographic Position – Latitude/Longitude

Table 3 contains the values of the following example:
$GPGLL , 3723.2475, N,12158.3416, W, 083559.00, A*2C.

Table 3: GLL Data Format

Name	Example	Units	Description
Message ID	$GPGLL		GLL protocol header
Latitude	3723.2475		Ddmm.mmmm
N/S Indicator	N		N=north or S=south
Longitude	12158.3416		Ddmm.mmmm
E/W Indicator	W		E=east or W=west
UTC Time	083559.00		Hhmmss.sss
Status	A		A=data valid or V=data not valid

Checksum	*2C		
<CR> <LF>			End of message temination(ASCII 13, ASCII 10)

GSA-GNSS DOP and Active Satellites

Table 4 contains the values of the following example:
$GPGSA , A, 3, 07, 02, 26,27, 09, 04,15, , , , , , 1.8,1.0,1.5*33.

Table 4: GSA Data Format

Name	Example	Units	Description
Message	$GPGSA		GSA protocol header
Mode 1	A		See Table 4-2
Mode 2	3		See Table 4-1
Satellite Used	07		Sv on Channel 1
Satellite Used	02		Sv on Channel 2
…	…		…
Satellite Used			Sv on Channel 12
PDOP	1.8		Position Dilution of Precision
HDOP	1.0		Horizontal Dilution of Precision
VDOP	1.5		Vertical Dilution of Precision
Checksum	*33		
<CR> <LF>			End of message termination(ASCII 13, ASCII 10)

Table 4-1: Mode 1

Value	Description
1	Fix not available
2	2D
3	3D

Table 4-2: Mode 2

Value	Description
M	Manual-forced to operate in 2D or 3D mode
A	Automatic-allowed to automatically switch 2D/3D

GSV-GNSS Satellites in View

Table 5 contains the values of the following example:
$GPGSV , 2, 1, 07, 07, 79, 048, 42, 02, 51,062, 43, 26, 36,256, 42, 27, 27, 138,42*71
$GPGSV , 2, 2, 07, 09, 23,313, 42, 04, 19, 159, 41, 15,12,041, 42*41.

Table 5: GGA Data Format

Name	Example	Units	Description
Message ID	$GPGSV		GSV protocol header
Number of	2		Range 1 to 3

Message			
Message Number	1		Range 1 to 3
Satellites in View	07		
Satellite ID	07		Channel 1(Range 1 to 32)
Elevation	79	degrees	Channel 1(Maximum 90)
Azinmuth	048	degrees	Channel 1(True, Range 0 to 359)
SNR(C/NO)	42	dBHz	Range 0 to 99,null when not tracking
…			…
Satellite ID	27		Channel 4(Range 1 to 32)
Elevation	27	degrees	Channel 4(Maximum 90)
Azimuth	138	degrees	Channel 4(True, Range 0 to 359)
SNR(C/NO)	42	dBHz	Range 0 to 99, null when not tracking
Checksum	*71		
<CR> <LF>			End of message termination(ASCII 13, ASCII 10)

Depending on the number of satellites tracked multiple messages of GSV data may be required.

RMC-Recommended Minimum Specific GNSS Data

Table 6 contains the values of the following example:
$GPRMC, 083559.00, A, 3723.2475, N, 12158.3416, W, 0.13, 309.62, 120598, , *10
Table 6: RMC Data Format

Name	Example	Units	Description
Message ID	$GPRMC		RMC protocol header
UTC Time	083559.00		hhmmss.sss
Status	A		A=data valid or V=data not valid
Latitude	3723.2475		ddmm.mmmm
N/S Indicator	N		N=north or S=south
Longitude	12158.3416		Ddmm.mmmm
E/W Indicator	W		E=east or W=west
Speed Over Ground	0.13	Knots	
Course Over Ground	309.62	Degrees	True
Date	120598		Dummy
Magnetic variation		Degrees	Not used
E/W indicator			Not used
Mode			Only NMEA0183 version 3.00 output
Checksum	*10	hexadecimal	
<CR> <LF>			End of message termination(ASCII 13, ASCII 10)

VTG-Course Over Ground and Ground Speed

Table 7 contains the values of the following example:
$GPVTG, 309.62, T, ,M, 0.13, N, 0.2, K*6E

Table 7: VTG Data Format

Name	Example	Units	Description
Message ID	$GPVTG		VTG protocol header
Course	309.62	Degrees	Measured heading
Reference	T		True
Course		Degrees	Measured heading
Reference	M		Magnetic
Speed	0.13	Knots	Measured horizontal speed
Units	N		Knots
Speed	0.2	Km/hr	Measured horizontal speed
Units	K		Kilometer per hour
Checksum	*6E		
<CR> <LF>			End of message termination

ZDA-Date and Time

Table 8 contains the values of the following example:
$GPZDA, 082710.00,04,07,2002,00,00*60

Name	Example	Units	Description
Message ID	$GPZDA		ZDA protocol header
UTC Time	082710.00		hhmmss.sss
Day	04		UTC time: day （01 ... 31）
Month	07		UTC time: month （01 ... 12）
Year	2002		UTC time: year （4 digit year）
local zone hours	00		Not supported (fixed to 00)
local zone minutes	00		Not supported (fixed to 00)
Checksum	*60		
<CR> <LF>			End of message termination

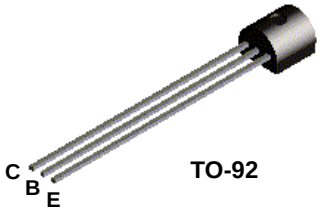
How to Reach Us:

Skylab M&C Technology Co., Ltd.
Address: Room.801, Bldg.211, Terra Industrial Park, Futian District, Shenzhen
Phone: 86-755 8340 8210（Sales Support）
Phone: 86-755 8340 8130（Technical Support）
Fax: 86-755-8340 8560
E-Mail: sales@skylab.com.cn

National Semiconductor™

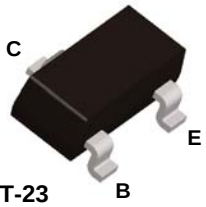
Discrete Power & Signal Technologies

2N3904



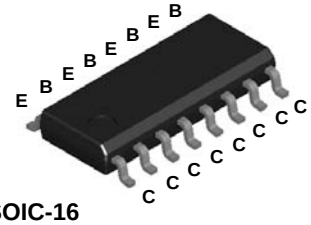
TO-92

MMBT3904



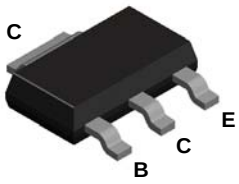
SOT-23
Mark: 1A

MMPQ3904



SOIC-16

PZT3904



SOT-223

NPN General Purpose Amplifier

This device is designed as a general purpose amplifier and switch. The useful dynamic range extends to 100 mA as a switch and to 100 MHz as an amplifier. Sourced from Process 23.

Absolute Maximum Ratings*

TA = 25°C unless otherwise noted

Symbol	Parameter	Value	Units
V_{CEO}	Collector-Emitter Voltage	40	V
V_{CBO}	Collector-Base Voltage	60	V
V_{EBO}	Emitter-Base Voltage	6.0	V
I_C	Collector Current - Continuous	200	mA
T_J, T_{stg}	Operating and Storage Junction Temperature Range	-55 to +150	°C

*These ratings are limiting values above which the serviceability of any semiconductor device may be impaired.

NOTES:
1) These ratings are based on a maximum junction temperature of 150 degrees C.
2) These are steady state limits. The factory should be consulted on applications involving pulsed or low duty cycle operations.

NPN General Purpose Amplifier
(continued)

Electrical Characteristics
TA = 25°C unless otherwise noted

Symbol	Parameter	Test Conditions	Min	Max	Units
OFF CHARACTERISTICS					
$V_{(BR)CEO}$	Collector-Emitter Breakdown Voltage	I_C = 10 mA, I_B = 0	40		V
$V_{(BR)CBO}$	Collector-Base Breakdown Voltage	I_C = 10 µA, I_E = 0	60		V
$V_{(BR)EBO}$	Emitter-Base Breakdown Voltage	I_E = 10 µA, I_C = 0	6.0		V
I_{BL}	Base Cutoff Current	V_{CE} = 30 V, V_{EB} = 0		50	nA
I_{CEX}	Collector Cutoff Current	V_{CE} = 30 V, V_{EB} = 0		50	nA
ON CHARACTERISTICS*					
h_{FE}	DC Current Gain	I_C = 0.1 mA, V_{CE} = 1.0 V I_C = 1.0 mA, V_{CE} = 1.0 V I_C = 10 mA, V_{CE} = 1.0 V I_C = 50 mA, V_{CE} = 1.0 V I_C = 100 mA, V_{CE} = 1.0 V	40 70 100 60 30	 300 	
$V_{CE(sat)}$	Collector-Emitter Saturation Voltage	I_C = 10 mA, I_B = 1.0 mA I_C = 50 mA, I_B = 5.0 mA		0.2 0.3	V V
$V_{BE(sat)}$	Base-Emitter Saturation Voltage	I_C = 10 mA, I_B = 1.0 mA I_C = 50 mA, I_B = 5.0 mA	0.65	0.85 0.95	V V
SMALL SIGNAL CHARACTERISTICS					
f_T	Current Gain - Bandwidth Product	I_C = 10 mA, V_{CE} = 20 V, f = 100 MHz	300		MHz
C_{obo}	Output Capacitance	V_{CB} = 5.0 V, I_E = 0, f = 1.0 MHz		4.0	pF
C_{ibo}	Input Capacitance	V_{EB} = 0.5 V, I_C = 0, f = 1.0 MHz		8.0	pF
NF	Noise Figure (except MMPQ3904)	I_C = 100 µA, V_{CE} = 5.0 V, R_S =1.0kΩ, f=10 Hz to 15.7 kHz		5.0	dB
SWITCHING CHARACTERISTICS (except MMPQ3904)					
t_d	Delay Time	V_{CC} = 3.0 V, V_{BE} = 0.5 V,		35	ns
t_r	Rise Time	I_C = 10 mA, I_{B1} = 1.0 mA		35	ns
t_s	Storage Time	V_{CC} = 3.0 V, I_C = 10mA		200	ns
t_f	Fall Time	I_{B1} = I_{B2} = 1.0 mA		50	ns

*Pulse Test: Pulse Width ≤ 300 µs, Duty Cycle ≤ 2.0%

Spice Model

NPN (Is=6.734f Xti=3 Eg=1.11 Vaf=74.03 Bf=416.4 Ne=1.259 Ise=6.734 Ikf=66.78m Xtb=1.5 Br=.7371 Nc=2 Isc=0 Ikr=0 Rc=1 Cjc=3.638p Mjc=.3085 Vjc=.75 Fc=.5 Cje=4.493p Mje=.2593 Vje=.75 Tr=239.5n Tf=301.2p Itf=.4 Vtf=4 Xtf=2 Rb=10)

NPN General Purpose Amplifier
(continued)

Thermal Characteristics
TA = 25°C unless otherwise noted

Symbol	Characteristic	Max		Units
		2N3904	*PZT3904	
P_D	Total Device Dissipation Derate above 25°C	625 5.0	1,000 8.0	mW mW/°C
$R_{\theta JC}$	Thermal Resistance, Junction to Case	83.3		°C/W
$R_{\theta JA}$	Thermal Resistance, Junction to Ambient	200	125	°C/W

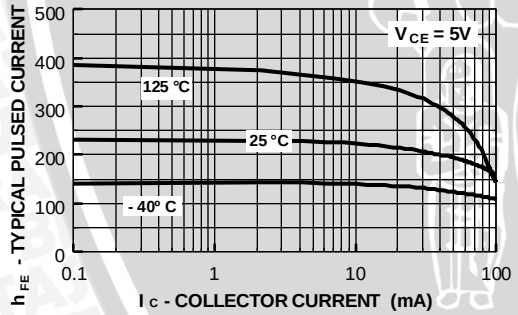
Symbol	Characteristic	Max		Units
		**MMBT3904	MMPQ3904	
P_D	Total Device Dissipation Derate above 25°C	350 2.8	1,000 8.0	mW mW/°C
$R_{\theta JA}$	Thermal Resistance, Junction to Ambient Effective 4 Die Each Die	357	 125 240	°C/W °C/W °C/W

*Device mounted on FR-4 PCB 36 mm X 18 mm X 1.5 mm; mounting pad for the collector lead min. 6 cm^2.

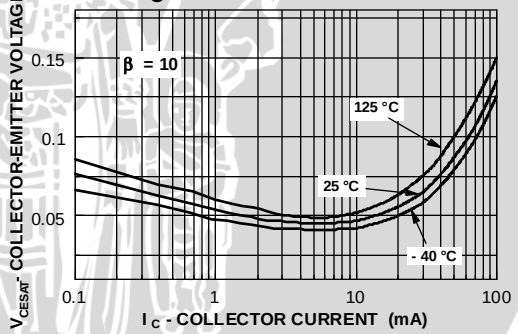
**Device mounted on FR-4 PCB 1.6" X 1.6" X 0.06."

Typical Characteristics

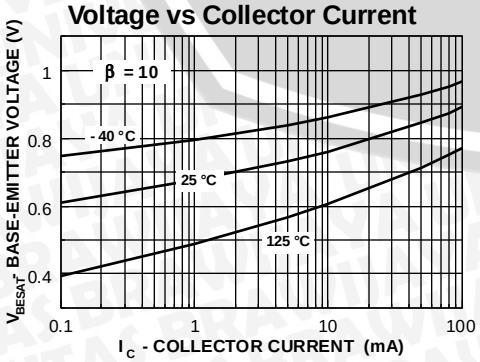
Typical Pulsed Current Gain vs Collector Current



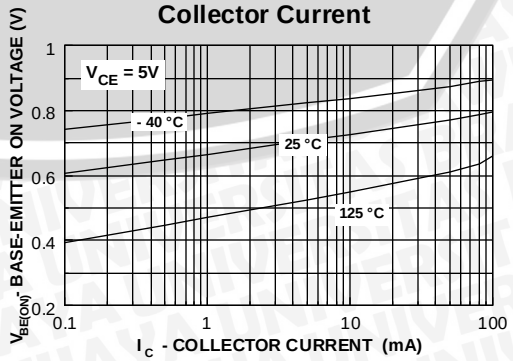
Collector-Emitter Saturation Voltage vs Collector Current



Base-Emitter Saturation Voltage vs Collector Current

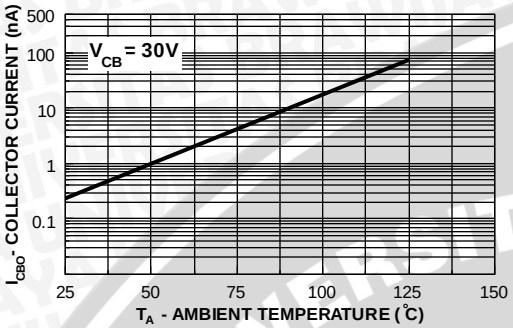


Base-Emitter ON Voltage vs Collector Current

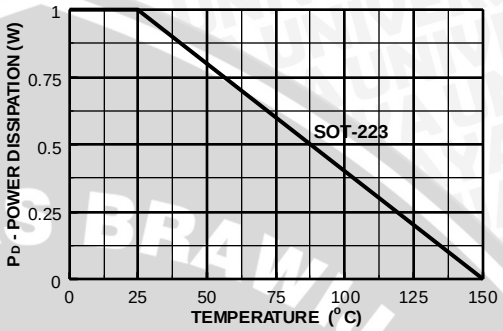


Typical Characteristics (continued)

Collector-Cutoff Current vs Ambient Temperature



POWER DISSIPATION vs AMBIENT TEMPERATURE



Test Circuits

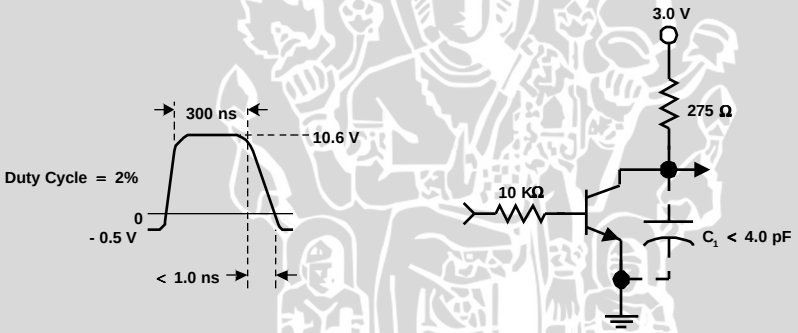


FIGURE 1: Delay and Rise Time Equivalent Test Circuit

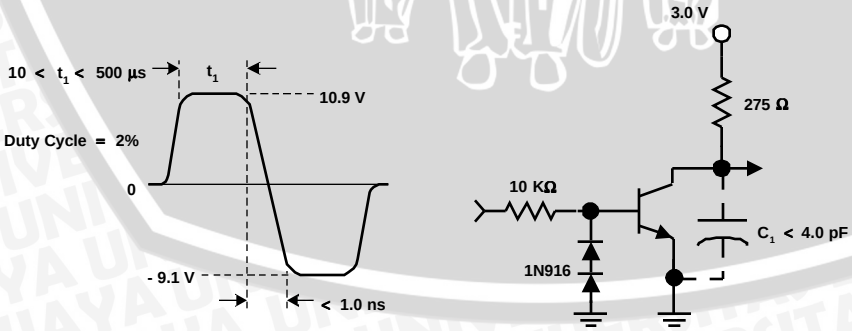


FIGURE 2: Storage and Fall Time Equivalent Test Circuit