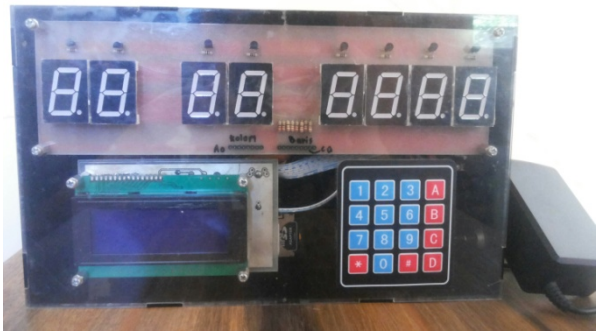
# KALENDER NASIONAL DIGITAL BERBASIS MIKROKONTROLER ATMEGA128 DENGAN TAMPILAN LCD DAN SEVEN SEGMENT

LAPORAN PROYEK AKHIR

Diajukan kepada Fakultas Teknik Universitas Negeri Yogyakarta

Untuk Memenuhi Sebagian Persyaratan

Guna Memperoleh Gelar Ahli Madya Teknik




Disusun Oleh :

FADILLAH NUROHMAH

NIM. 10507131031

JURUSAN PENDIDIKAN TEKNIK ELEKTRONIKA

FAKULTAS TEKNIK

UNIVERSITAS NEGERI YOGYAKARTA


2015

## LEMBAR PERSETUJUAN

Proyek akhir yang berjudul "Kalender Nasional Digital Berbasis Mikrokontroler ATmega128 dengan Tampilan LCD dan *Seven Segment*" ini telah disetujui oleh pembimbing untuk diujikan.

Yogyakarta, Maret 2015

| Mengetahui | Menyetujui |
|---|---|
| Kaprodi Teknik Elektronika D3 | Dosen Pembimbing, |
| **Djoko Santoso, M.Pd** | **Handaru Jati, S.T., M.M., M.T., Ph.D** |
| **NIP. 19580422 198403 1 002** | **NIP. 19740511 1999031 002** |

**LEMBAR PENGESAHAN**

Tugas Proyek Akhir dengan Judul:

**KALENDER DIGITAL NASIONAL BERBASIS ATMEGA128 DENGAN TAMPILAN LCD DAN SEVEN SEGMENT**

Disusun Oleh:
Fadillah Nurohmah
NIM. 10507131031

Telah dipertahankan di depan Tim Penguji Tugas Proyek Akhir Program Studi Teknik Elektronika D3 Fakultas Teknik Universitas Negeri Yogyakarta

Pada tanggal 25 Maret 2015

**DEWAN PENGUJI**

| Nama/Jabatan | Tanda Tangan | Tanggal |
|---|---|---|
| Handaru Jati, S.T., M.M., M.T., Ph.D<br>Ketua Penguji/Pembimbing | ..... | 14/04-2015 |
| Dr. Fatchul Arifin, M.T.<br>Sekertaris Penguji | ..... | 14/04-2015 |
| Nurkhamid, M. Kom., Ph.D<br>Penguji Utama | ..... | 14/04-2015 |

Yogyakarta, April 2015
**Fakultas Teknik Universitas Negeri Yogyakarta**
**Dekan,**

**Dr. Moch. Bruri Triyono, M.Pd.**
**NIP. 19560216 198603 1 003**

## LEMBAR PERNYATAAN KEASLIAN

Yang bertanda tangan di bawah ini:

| | |
|---|---|
| Nama | : Fadillah Nurohmah |
| NIM | : 10507131031 |
| Program Studi | : Teknik Elektronika |
| Judul PA | : Kalender Nasional Digital Berbasis Mikrokontroler Atmega128 dengan Tampilan LCD dan Seven Segment |

Menyatakan bahwa Proyek Akhir ini adalah hasil pekerjaan saya sendiri dan sepanjang pengetahuan saya, tidak berisi materi tertulis orang lain sebagai persyaratan penyelesaian studi di Universitas Negeri Yogyakarta atau Perguruan Tinggi lain, kecuali bagian-bagian tertentu yang saya ambil sebagai acuan dengan mengikuti tata cara dan penulisan karya ilmiah yang lazim. Jika ternyata terbukti pernyataan ini tidak benar, sepenuhnya menjadi tanggung jawab saya.

Yogyakarta, Maret 2015

Penulis

**Fadillah Nurohmah**

**NIM. 10507131031**

## MOTTO

*"Sesungguhnya bersama kesulitan itu ada kemudahan"*

(QS. Al-Insyiroh: 6)

"*Banyak kegagalan dalam hidup ini dikarenakan orang – orang tidak menyadari betapa dekatnya mereka dengan keberhasilan saat mereka menyerah"*

(Thomas Alva Edison)

*"Sabar dan bertindak bijaksana dalam mengatasi kesulitan adalah hal yang paling utama"*

(Penulis)

*"Setiap pekerjaan dapat terselesaikan dengan mudah, apabila dikerjakan tanpa keengganan"*

(Penulis)

## PERSEMBAHAN

*Dengan penuh keyakinan karya ini saya persembahkan untuk :*

*Bapak, Ibu, dan adik atas doa, kasih sayang, dan dukungan yang senantiasa mengiringi perjalanan saya dalam menuntut ilmu.*

*Seluruh keluarga besar yang telah mendoakan dan memberi dukungan moral dan spiritual.*

*Dosen Pembimbing Proyek Akhir, Bapak Handaru Jati, Ph.D yang selalu membimbing dan memotivasi untuk semangat dalam menyelesaikan proyek akhir ini.*

*Janis Uky yang dengan sabar menasehati, menemani, membantu, dan memotivasi agar semangat dalam menyelesaikan Proyek Akhir ini.*

*Rekan – rekan Kelas B 2010 Teknik Elektronika FT UNY atas dukungan, bantuan, dan semangat dari kalian dalam penyelesaian Proyek Akhir ini.*

**Kalender Nasional Digital Berbasis Mikrokontroller ATmega128 dengan Tampilan LCD dan *Seven Segment***

*Oleh : Fadillah Nurohmah*

*NIM : 10507131031*

## ABSTRAK

Kalender adalah suatu sistem dalam sebuah periode waktu yang terdapat daftar hari, tanggal, dan bulan dalam setahun. Proyek akhir ini bertujuan untuk merealisasikan *hardware*, *software*, dan mengetahui unjuk kerja kalender nasional digital. Alat ini diharapkan dapat membantu manusia mengingat *event*/hari peringatan yang akan berbunyi seperti alarm.

Dalam pembuatan proyek akhir ini terdiri dari beberapa tahap yaitu, (1) Identifikasi Kebutuhan, (2) Analisis Kebutuhan, (3) Perancangan Perangkat Keras, (4) Perancangan Perangkat Lunak, (5) Pembuatan Alat, dan (6) Pengujian Alat. Perangkat keras terdiri dari rangkaian yaitu catu daya, rangkaian RTC, rangkaian sistem minimum mikrokontroler ATmega128 dengan tampilan LCD, dan rangkaian sistem minimum mikrokontroler ATmega32 dengan tampilan *seven segment*. Perancangan perangkat lunak (*software*) sebagai pengendali menggunakan bahasa pemrograman *basic* serta *software* BASCOM AVR sebagai *compiler*.

Berdasarkan hasil pengujian telah di dapatkan bahwa dari rangkaian *input* berupa *keypad matrix* dan rangkaian RTC, dan rangkaian catu daya akan terhubung ke pengolah data mikrokontroler ATmega128, kemudian menghasilkan *output* berupa tampilan LCD dan *seven segment*. Kalender akan bunyi sebagai pemberitahuan adanya hari peringatan, ketika terdapat *event* pada tanggal tertentu.

Kata kunci : Kalender nasional digital, Mikrokontroler ATmega128, Mikrokontroler ATmega32, RTC, LCD, *Seven segment*

# KATA PENGANTAR

Puji syukur penulis panjatkan kepada Allah SWT yang telah melimpahkan rahmat serta hidayah-Nya, sehingga dapat menyelesaikan Proyek Akhir dan laporannya dengan judul " Kalender Nasional Digital Berbasis Mikrokontroller ATmega128 dengan Tampilan LCD dan *Seven Segment* ". Pembuatan Proyek Akhir sebagai syarat untuk memperoleh gelar Ahli Madya Fakultas Teknik Universitas Negeri Yogyakarta.

Penulis menyadari bahwa selama pembuatan Proyek Akhir tidak lepas dari bantuan dari berbagai pihak, baik secara langsung maupun tidak langsung. Oleh karena itu dengan kerendahan hati pada kesempatan ini penulis ingin mengucapkan terima kasih kepada :

1. Allah SWT yang telah menuntun, memberikan kesehatan dan memberikan petunjuk kepada kita semua.
2. Bapak dan ibu tercinta yang tak lelah memberikan kasih sayang serta dukungan moral dan spiritual.
3. Bapak Dr. Moch. Bruri Triyono selaku Dekan Fakultas Teknik Universitas Negeri Yogyakarta.
4. Bapak Drs. Muhammad Munir, M.Pd selaku Ketua Jurusan Pendidikan Teknik Elektronika Fakultas Teknik Universitas Negeri Yogyakarta.
5. Bapak Djoko Santoso, M.Pd. selaku Koordinator Proyek Akhir Jurusan Pendidikan Teknik Elektronika Fakultas Teknik Universitas Negeri Yogyakarta.

6. Bapak Handaru Jati, Pd.D selaku Dosen Pembimbing Proyek Akhir yang telah membantu dan selalu memberi motivasi serta arahan dalam bimbingan.
7. Tika Danti Saraswati dan Wasang Juwi Pracihno atas bantuan, nasehat, dukungan, dan masukan – masukannya.
8. Seluruh teman Kelas B 2010 Teknik Elektronika Universitas Negeri Yogyakarta yang telah memberikan bantuan dan semangatnya.
9. Ima Surahmi serta sahabat – sahabat yang selalu mendukung dan memotivasi agar terselesaikannya Proyek Akhir.
10. Janis Uky yang senantiasa sabar menemani, membantu, memberi dukungan, memberi masukan, nasehat dan semangat.
11. Adik perempuan saya yang sangat pengertian dan perhatian.
12. Serta semua pihak yang telah membantu kelancaran Proyek Akhir yang tidak dapat penulis sebutkan satu persatu.

Penulis menyadari bahwa penulisan laporan Proyek Akhir ini masih jauh dari kesempurnaan. Oleh karena itu penulis mengharapkan kritik dan saran yang bersifat membangun duna menyempurnakan laporan ini. Semoga Proyek Akhir ini memberikan manfaat bagi siapa saja yang membacanya.

Yogyakarta, Maret 2015

Penulis

## DAFTAR ISI

Halaman

# DAFTAR TABEL

Halaman

# DAFTAR GAMBAR

Halaman

## DAFTAR LAMPIRAN

Halaman

# BAB I

# PENDAHULUAN

## A. Latar Belakang Masalah

Seiring dengan perkembangan zaman dan perkembangan teknologi yang semakin pesat. Kebutuhan masyarakat akan perangkat-perangkat rumah tangga kini semakin meningkat, maka terciptalah begitu banyak perangkat dengan teknologi yang begitu inovatif sebagai penunjang kebutuhan masyarakat itu sendiri. Kalender adalah suatu sistem dalam sebuah periode waktu yang terdapat daftar hari, tanggal, dan bulan dalam setahun. Kalender berfungsi sebagai penanda waktu hari dan pengatur jadwal kegiatan sehari.

Saat sedang sibuk bekerja atau melakukan kegiatan terkadang kita lupa tanggal, hari, bahkan bulan saat ini. Beberapa masyarakat pernah mengalami, pergi bekerja atau ke sekolah pada hari besar atau hari minggu yang biasanya kegiatan diliburkan. Hal ini terjadi karena tidak ada yang mengingatkan kalau hari itu ada *event* sehingga sekolah atau tempat bekerja libur. Supaya hal tersebut tidak terjadi dibutuhkan alat untuk mengingatkan kita, seperti adanya *event* pada tanggal-tanggal tertentu.

Teknologi saat ini sudah mendekati serba otomatis, dimana hampir semua kegiatan menggunakan peralatan elektronik. Oleh karena itu untuk mempermudah manusia, dalam hal ini dibutuhkan suatu alat yang hampir sama fungsinya dengan jam digital yang sudah tercipta sejak lama. Sebagai contoh cara kerja jam digital, seringkali mengatur alarm untuk mengingatkan

atau membangunkan kita saat tidur. Begitu juga dengan kalender digital sebagai alat bantu manusia untuk mengingatkan ada atau tidak adanya *event* pada setiap harinya.

Selain untuk mengingatkan hari-hari besar, kalender digital tersebut dilengkapi juga dengan jam digital dan pasaran jawa. Dalam kebudayaan masyarakat Indonesia, khususnya di daerah pulau Jawa masih kental dengan pasaran Jawa. Kepercayaan masyarakat pada pasaran Jawa, secara garis besar karena adanya hari baik dan buruk. Entah itu untuk peringatan hajatan seperti, peringatan hari kematian, tanggal baik untuk hari pernikahan dan sebagainya. Pada umumnya masih banyak pasar tradisional di beberapa daerah buka saat hari – hari tertentu pasaran Jawa. Dewasanya masyarakat modern saat ini banyak yang melupakan tradisi pasaran Jawa. Seharusnya kita sebagai warga negara Indonesia yang baik menjunjung tinggi kebudayaan Indonesia. Sudah jelas bahwa pasaran Jawa tersebut hanya ada di negara kita, janganlah meninggalkan kebudayaan yang telah ada sejak dulu.

Dari uraian - uraian di atas penulis membuat proyek akhir yang berjudul " Kalender Nasional Digital Berbasis Mikrokontroller ATmega128 dengan Tampilan LCD dan *Seven Segment* ". Bagian-bagian kalender digital tersebut terdiri dari perangkat kendali mikrokontroller ATmega128, *micro sd* sebagai penyimpan *memory*, dan sebuah LCD *display* sebagai penampil jam, tanggal, pasaran Jawa dan *event*.

## B. Identifikasi Masalah

Dari uraian latar belakang masalah di atas, maka dapat diidentifikasi masalah sebagai berikut :

1. Saat ini belum ada kalender yang dapat digunakan untuk jangka waktu yang lama
2. Belum ada kalender digital yang dilengkapi dengan jam dan pasaran Jawa
3. Belum adanya rancangan program untuk kalender nasional digital yang bekerja secara otomatis

## C. Batasan Masalah

Berdasarkan identifikasi masalah yang ada, perlu adanya pembatasan masalah sehingga ruang lingkup permasalahan jelas. Ruang lingkup batasan masalah dalam proyek akhir ini hanya pada pembuatan kalender digital yang dilengkapi pasaran Jawa dan jam yang dapat bekerja secara otomatis. Kalender digital ini akan menunjukkan adanya *event* pada tanggal-tanggal tertentu. Jika ada *event* pada tanggal tertentu, maka LCD akan menampilkan jenis *event* tersebut dan berbunyi seperti alarm. Sedangkan masukan *input* menggunakan *keypad matrix* sebagai pengaturan.

## D. Rumusan Masalah

Dari identifikasi masalah dan batasan masalah yang dikemukakan di atas maka diperoleh rumusan masalah sebagai berikut :

1. Bagaimana cara membuat kalender nasional digital yang dapat digunakan untuk jangka waktu yang lama?
2. Bagaimana cara menjalankan sistem kalender nasional digital dengan kapasitas *memory* yang besar dan tampilan yang menarik?
3. Bagaimana kinerja kalender nasional digital supaya sesuai dengan perancangan yang dibuat?

## E. Tujuan

Pembuatan proyek akhir kalender nasional digital berbasis mikrokontroller ATmega128 dengan tampilan LCD dan seven segment mempunyai tujuan sebagai berikut :

1. Dapat merealisasikan kalender nasional digital berbasis mikrokontroller ATmega128 dengan tampilan LCD dan seven segment.
2. Dapat merealisasikan *software* untuk kalender nasional digital berbasis mikrokontroller ATmega128 dengan tampilan LCD dan seven segment.
3. Dapat mengetahui unjuk kerja kalender nasional digital berbasis mikrokontroller ATmega128 dengan tampilan LCD dan seven segment.

## F. Manfaat

Dari pembuatan proyek akhir ini penulis mengharapkan dapat memberikan manfaat bagi mahasiswa, lembaga pendidikan, dan dunia usaha/industri. Berbagai manfaat yang diharapkan adalah:

1. Bagi mahasiswa :
    a. Sebagai ilmu pengetahuan yang di dapat di bangku pendidikan maupun dari pengalaman di lapangan.
    b. Sebagai bentuk kontribusi terhadap universitas baik dalam citra maupun daya tawar terhadap masyarakat luas.
2. Bagi Jurusan P.T. Elektronika
    a. Terciptanya inovasi baru dalam dunia pendidikan sebagai sarana ilmu pengetahuan.
    b. Sebagai bahan referensi untuk pengembangan selanjutnya.
    c. Sebagai wujud partisipasi mahasiswa dalam perkembangan ilmu teknologi elektronika.
3. Bagi Dunia Usaha/ Industri
    a. Terciptanya alat sebagai sarana peningkatan teknologi dalam dunia usaha dan industri.
    b. Sebagai bentuk kontribusi terhadap dunia usaha dan industri dalam mewujudkan pengembangan teknologi.

## G. Keaslian Gagasan

Pembuatan proyek akhir ini terinspirasi dari kehidupan sehari-hari, setiap melakukan kegiatan kita sering melihat jam dan tanggal. Saat ini sudah banyak alat-alat yang dilengkapi dengan aplikasi jam dan kalender digital. Akan tetapi masih jarang kita jumpai aplikasi tersebut bisa memberitahu atau mengingatkan hari-hari besar nasional di masing-masing negara. Apalagi

seperti kalender nasional di negara kita yang terdapat pasaran Jawa. Oleh karena itu pembuatan proyek akhir ini mempunyai kelebihan, antara lain :

1. Dapat memberitahukan adanya hari peringatan nasional dengan teks dan bunyi.
2. Kalender nasional yang dilengkapi dengan aplikasi jam.
3. Terdapat pasaran Jawa seperti kalender manual yang kita gunakan sehari-hari.
4. Kalender nasional yang berbasis mikrokontroller ATmega128.
5. Kalender yang sangat praktis dan otomatis.

Dari berbagai pernyataan yang disebutkan di atas maka penulis menyatakan bahwa proyek akhir yang berjudul " Kalender Nasional Digital Berbasis Mikrokontroller ATmega128 dengan Tampilan LCD dan *Seven Segment*" adalah benar-benar rancangan dari penulis sendiri. Dan dari sepengetahuan penulis proyek akhir ini belum pernah dibuat oleh civitas akademik di Universitas Negeri Yogyakarta ataupun di institusi lain.

# BAB II

# PENDEKATAN PEMECAHAN MASALAH

## A. Kalender Nasional Digital

Kalender digital adalah kalender elektronik yang diprogram dan dirancang untuk membantu manusia mengingatkan hari besar dan bisa menyimpan planing yang dibuat sendiri, sehingga pada saat tanggal yang sudah di setting akan berbunyi dan muncul teks. Kalender tersebut menggunakan arus DC dengan tegangan PLN 220 volt sehingga dapat bekerja secara otomatis.

Pada umumnya kalender hanya dibuat secara sederhana dari kertas yang sudah komplit. Pada kalender biasa terdapat hari, tanggal, bulan, tahun, hari peringatan, penanggalan Jawa, dan lain-lain. Memang saat ini sudah ada kalender elektronik bahkan di rumah tangga tak sedikit yang menggunakannya, tetapi masih sederhana dan terbatas fungsinya. Hal ini dikarenakan keterbatasan pada pemrogamannya yang hanya menampilkan penanggalan dan jam saja. Maka dengan keterbatasan tersebut kalender digital mengalami perkembangan bentuk, fungsi dan sebagainya.

## B. Mikrokontroler ATmega128

Mikrokontroler AVR dikelompokkan menjadi empat kelas yaitu keluarga ATtiny, AT90Sxx, ATmega, AT86RFxx. Perbedaan pada masing - masing keluarga AVR adalah kapasitas memori, *peripheral,* dan fungsi.

Mikrokontroler ATmega128 merupakan salah satu varian dari mikrokontroler AVR 8-bit yang diproduksi oleh Atmel.

1. Arsitektur Mikrokontroler ATmega128

   Mikrokontroler ATmega 128 merupakan mikrokontroler keluarga AVR yang mempunyai kapasitas flash memori 128KB. AVR (*Alf and Vegard's Risc Processor*) merupakan seri mikrokontroler CMOS 8-bit buatan Atmel, berbasis arsitektur RISC (*Reduced Instruction Set Computer*). Dengan mengeksekusi instruksi kuat dalam satu siklus clock tunggal, ATmega128 mencapai *throughput* mendekati 1 MIPS per MHz yang memungkinkan perancang sistem untuk mengoptimalkan konsumsi daya melawan kecepatan proses.

2. Fitur Mikrokontroler ATmega128

   Menurut *datasheet* ATmega128 yang diambil dari sebuah situs resmi Atmel (http://html.alldatasheet.com/html-pdf/56260/ATMEL/ ATMEGA128 /1/128/ATMEGA128.html), fitur-fitur pada mikrokontroler ATmega128 antara lain sebagai berikut:

   a. Mikrokontroler AVR 8 bit mempunyai kemampuan tinggi dengan daya rendah.

   b. Arsitektur canggih RISC

      1) 133 intruksi yang kuat. Kebanyakan *Single Clock* siklus eksekusi

      2) 32 x 8 tujuan umum kerja register + *Peripheral* kontrol register

      3) Operasi sepenuhnya statis

      4) Sampai dengan 16 MIPS *troughput* pada 16 MHz

5) *On-chip* 2- siklus *multiplier*

c. Segmen Memory Tinggi Ketahanan *Non-volatile*

*1) 128K Bytes of In-System Reprogrammable Flash Memory*

*2)* 4Kbytes *EEPROM*

*3)* 4Kbytes *Internal SRAM*

4) Menulis / Menghapus siklus*: 10.000 Flash / 100.000 EEPROM*

5) Retensi data: 20 tahun pada 85

*6)* Kode pilihan *Boot* Bagian dengan *Independent Lock Bits*

*a) In-System Programming* secara *On-chip Program Boot*

*b) True Read-While-Write Operation*

*7)* Sampai dengan Ruang 64Kbytes pilihan Memori *Eksternal*

8) Pemrograman *Lock* untuk Software Keamanan

9) *SPI Interface* untuk *In-System Programming*

d. Dukungan *library QTouch*®

*1)* Tombol sentuh kapasitif, *slider* dan *wheels*

*2) Qtouch* dan *Qmatrix acqisition*

3) Sampai dengan 64 saluran akal

e. *JTAG (IEEE std. 1149.1 Compliant) Interface*

1) Kemampuan batas scan Menurut JTAG Standar

*2)* Luas *On-chip Debug Support*

*3)* Pemrograman *Flash, EEPROM,* Sekering dan *Lock Bits* melalui *JTAG Interface*

f. Fitur *Peripheral*

*1)* *Two 8-bit Timer/Counters* dengan *Prescalers* terpisah dan Bandingkan *Modes*

*2)* *Two Expanded 16-bit Timer/Counters* dengan *Separate Prescaler, Compare Mode* dan *CaptureMode*

*3)* *Real Time Counter* dengan *Separate Oscillator*

*4)* *Two 8-bit PWM* saluran

*5)* 6 Saluran *PWM* dengan *Programmable* Resolusi *2-16 Bits*

*6)* Output Bandingkan Modulator

*7)* *Byte* berorientasi *Two-wire Serial Interface*

*8)* *Dual Programmable Serial USARTs*

*9)* *Master/Slave SPI Serial Interface*

*10)* *Programmable Watchdog Timer* dengan *On-chip Oscillator*

*11)* *On-chip Analog Comparator*

12) 8 saluran, 10-bit ADC

*a)* *8 Single-ended Channels*

*b)* *7 Differential Channel*

*c)* *2 Differential Channels with Programmable Gain at 1x, 10x, or 200x*

g. Fitur Mikrokontroler Khusus

1) *Power-on reset* dan *Programmable Brown*

2) *out Detection- internal* dikalibrasi *RC Oscillator*

3) *Eksternal* dan *Internal Interrupt* Sumber

4) Enam *Mode Sleep: Idle, ADC Noise Reduction, Power-save, Power-down,* siaga, *and Extended Standby*

5) *Software* dipilih jam frekuensi

6) *ATmega103 Compatibility Mode* dipilih oleh *Fuse*

7) *Global Pull-up Disable*

h. *I/O* dan paket

1) *53 Programmable I/O Lines*

2) *64-lead TQFP* dan *64-pad MLF*

i. *Operating Voltages*

1) *2.7 - 5.5V for ATmega128L*

2) *4.5 - 5.5V for ATmega128*

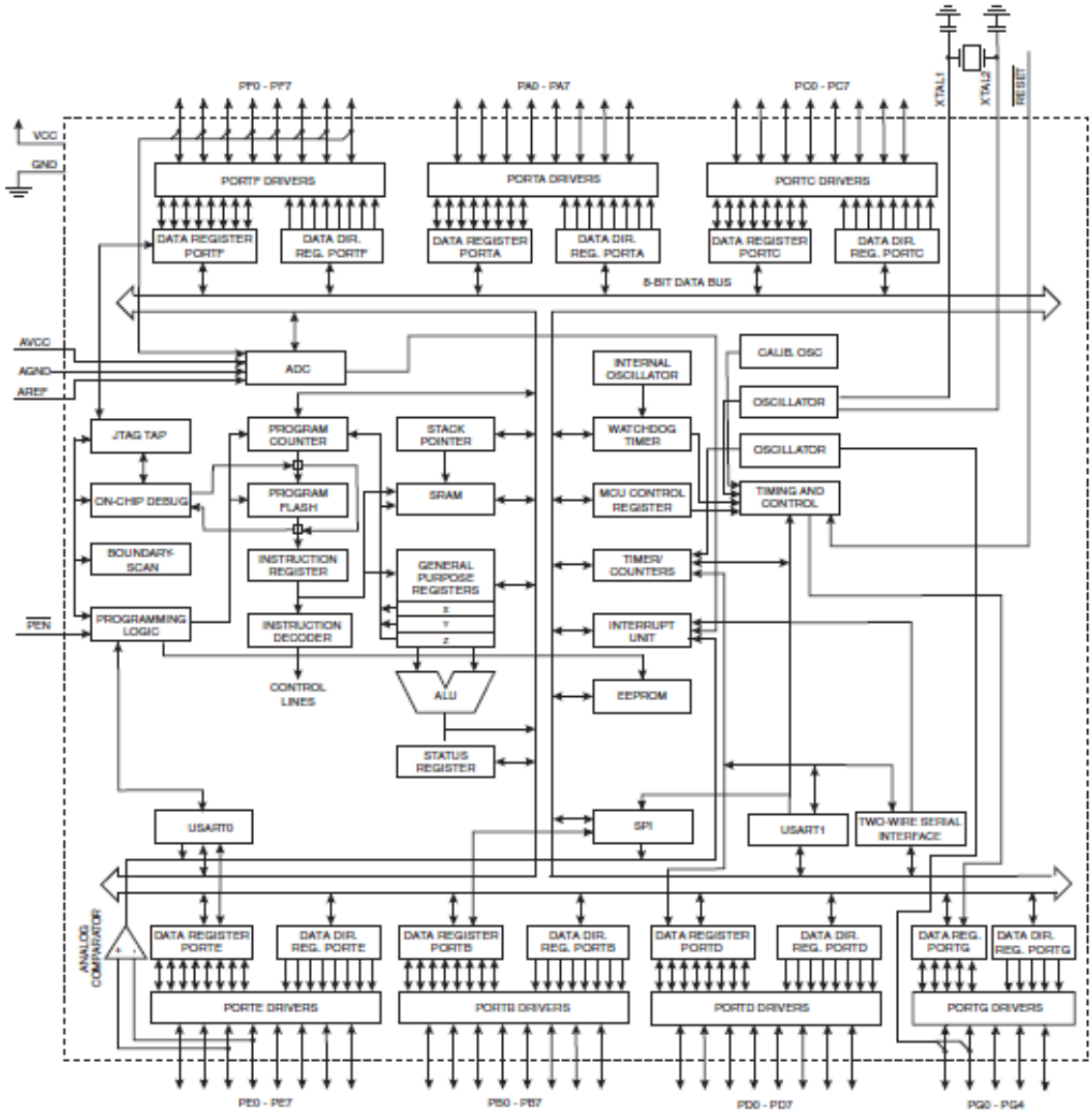
j. *Speed Grades*

1) *0 - 8 MHz for ATmega128L*

2) *0 - 16 MHz for ATmega128*

3. Blok Diagram Mikrokontroler ATmega128

Gambar 1 merupakan blok diagram Mikrokontroler ATmega128.

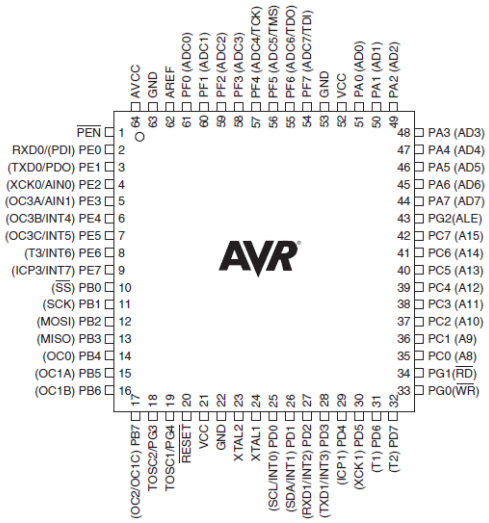



Gambar 1. Blok Diagram ATmega128

(http://html.alldatasheet.com/html-

pdf/56260/ATMEL/ATMEGA128/128/1/ATMEGA128.html)

4. Konfigurasi *Pin* AVR ATmega128




Gambar 2. Konfigurasi *Pin* ATmega128

(http://html.alldatasheet.com/html-pdf/56260/ATMEL/ATMEGA128/128/1/ATMEGA128.html)

Gambar 2 menunjukkan konfigurasi pin ATmega128, berikut penjelasannya:

a. VCC : Suplai tegangan

b. GND : *Ground*

c. Port A(PA7..PA0) : Port A merupakan *bi-directional* 8-bit port I/O

d. Port B(PB7..PB0) : Port B merupakan *bi-directional* 8-bit port I/O dan pin fungsi khusus yaitu *Output Compare* dan PWM *Output, Timer/Counter* dan SPI

e. Port C(PC7..PC0) : Port C merupakan *bi-directional* 8-bit port I/O

f. Port D(PD7..PD0) : Port D merupakan *bi-directional* 8-bit port I/O dan pin fungsi khusus yaitu *Timer/Counter, External Interrupt,* UART, TWI

g. Port E(PE7..PE0) : Port E merupakan *bi-directional* 8-bit port I/O dan pin fungsi khusus yaitu *External Interrupt, Input Capture Pin, Timer/Counter, Output Compare* dan *PWM Output, Analog Comparator Negative, Programming Data Out, Programming Data Input, UART*

*h.* Port F(PF7..PF0) : Port F berfungsi sebagai input analog ke *A/D Converter*. Port F juga berfungsi sebagai *bi-directional* 8-bit port I/O, jika *A/D Converter* tidak digunakan. Port F juga melayani fungsi antarmuka JTAG

i. Port G(PG7..PG0) : Port G merupakan *bi-directional* 8-bit port I/O

j. RESET : Pin yang berfungsi untuk me-*reset* mikrokontroler

k. XTAL1 : Masukan untuk *Inverting Osilator* dan masukan untuk rangkaian operasi internal clock

l. XTAL2 : Keluaran dari *Inverting Osilator amplifier*

m. AVCC : Merupakan pin tegangan suplay untuk port F dan *A/D Converter*

n. AREF : Pin referensi analog bagi *A/D Converter*

o. PEN : Pin pemrograman untuk mengaktifkan mode pemrograman serial SPI

## C. Mikrokontroler ATmega32

Mikrokontroler ATmega32 merupakan mikrokontroler *low-power* CMOS 8 bit berdasarkan arsitektur AVR RISC.

1. Arsitektur Mikrokontroler ATmega32

   Mikrokontroler ATmega32 merupakan mikrokontroler RISC 8 bit, dimana semua instruksi dikemas dalam kode 16-bit dan sebagian besar instruksi dieksekusi dalam satu siklus clock.

2. Menurut datasheet ATmega32 yang diambil dari situs resmi Atmel (http://html.alldatasheet.com/html-pdf/77378/ATMEL/ATMEGA32/127/1/ATMEGA32.html), fitur mikrokontroler ATmega32 adalah sebagai berikut :

   a. Mikrokontroler AVR 8 bit mempunyai kemampuan tinggi dengan daya rendah.

   b. Arsitektur canggih RISC

      1) 133 intruksi yang kuat. Kebanyakan *Single Clock* siklus eksekusi

2) 32 x 8 tujuan umum kerja register + *Peripheral* kontrol register

3) Operasi sepenuhnya statis

4) Sampai dengan 16 MIPS *troughput* pada 16 MHz

5) On-chip 2- siklus *multiplier*

c. Program *Non-volatile* dan Data *Memory*

*1) 32K Bytes of In-System Self-programmable Flash*

*a) Endurance : 10,000 Write/Erase Cycles*

2) Kode pilihan *Boot* bagian dengan *Independent Lock Bits*

*a) In-System Programming* secara *On-chip Boot Program*

*b) True Read-While-Write Operation*

*3) 1024Bytes EEPROM*

*a) Endurance : 100,000 Write/Erase Cycles*

*4) 2K Byte Internal SRAM*

*5) Programming Lock* untuk *Software Security*

b. *JTAG (IEEE std. 1149.1 Compliant) Interface*

*1)* Kemampuan batas scan menurut JTAG standar

*2) Extensive On-chip Debug Support*

*3)* Pemrograman *Flash, EEPROM, Fuses,* dan *Lock Bits* melalui *JTAG Interface*

c. Fitur *Peripheral*

*1) Two 8-bit Timers/Counters with Separate Prescaler and Compare Mode*

2) *One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode*
3) *Real Time Counter with Separate Oscillator*
4) *Four PWM Channels*
5) *8-channel,10-bit ADC*
    a) *8 Single-ended Channels*
    b) *7 Differential Channels in TQFP Package Only*
    c) *2 Differential Channels with Programmable Gain at 1x, 10x,200x*
6) *Byte-oriented Two-wire Serial Interface*
7) *Programmable Serial USART*
8) *Master/Slave SPI Serial Interface*
9) *Programmable Watchdog Timer with Separate On-Chip Oscillator*
10) *On-Chip Analog Comparator*

d. *Special Microcontroller features*

1) *Power-On Reset and Programmable Brown-out Detection*
2) *Internal Calibrated RC Oscillator*
3) *External and Internal Interrupt Sources*
4) *Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby*

e. *I/O and Packages*

1) *32 Programmable I/O Lines*

2) *40-pin PDIP, 44-lead TQFP, and 44-pad QFN/MLF*

f. *Operating Voltages*

1) *2.7 – 5.5V (ATmega32L)*

2) *4.5 – 5.5V (ATmega32)*

g. *Speed Grades*

1) *0 – 8MHz (ATmega32L)*

2) *0 – 16MHz (ATmega32)*

h. *Power Consumption at 1MHz, 3V, 25`C for ATmega32L*

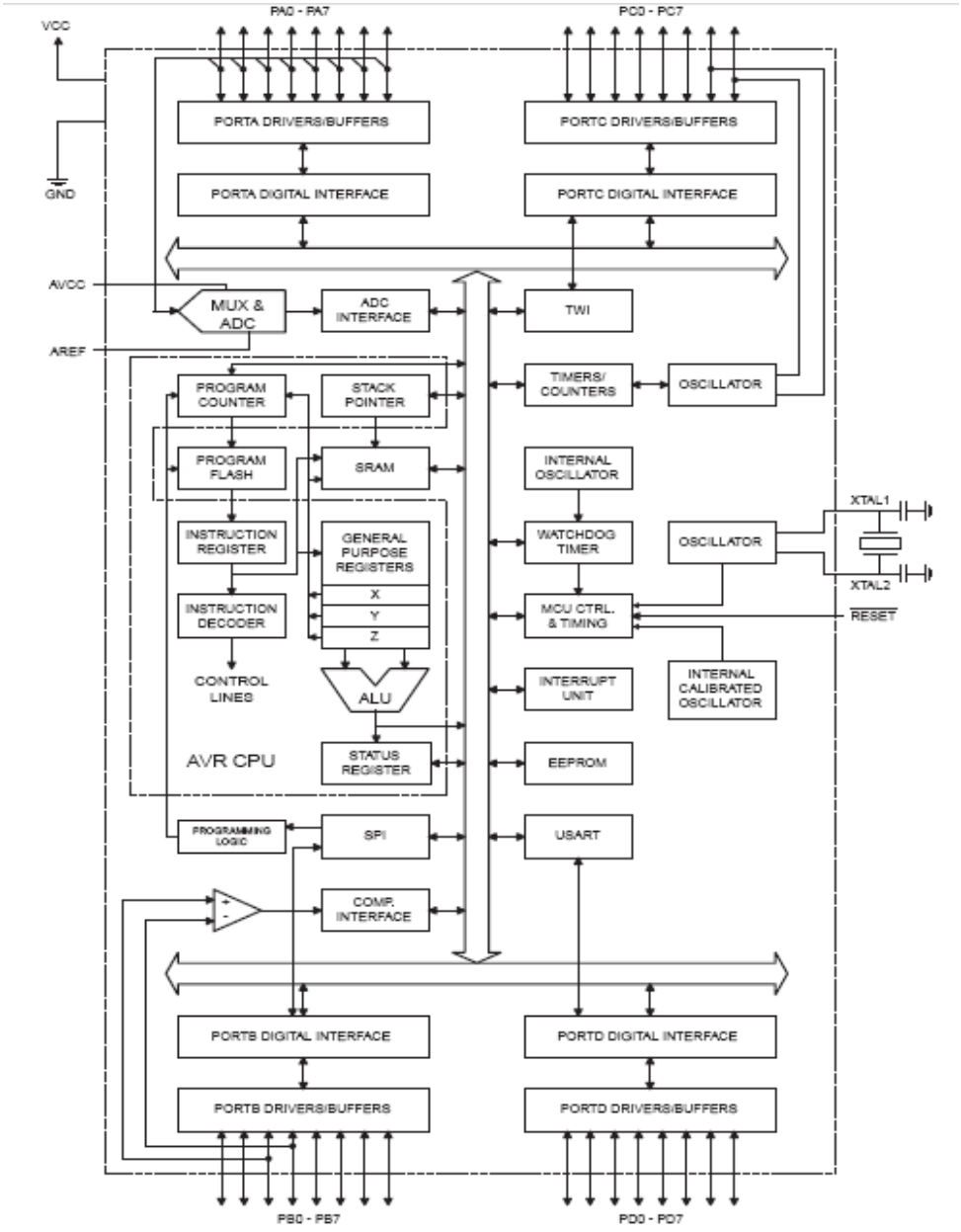
1) *Active: 1.1 mA*

2) *Idle Mode: 0.35 mA*

3) *Power-Down Mode: <1 uA*

3. Blok Diagram Mikrokontroler ATmega32

Gambar 3 adalah blok diagram Mikrokontroler ATmega32.

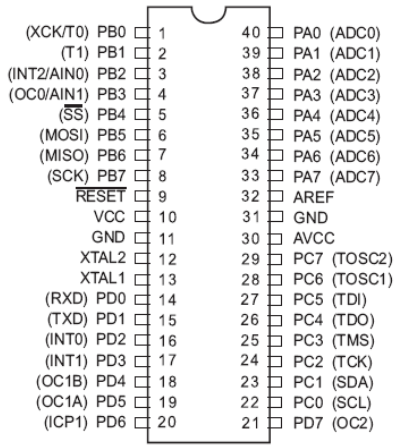



Gambar 3. Blok Diagram ATmega32

(http://html.alldatasheet.com/html-

pdf/77378/ATMEL/ATMEGA32/127/1/ATMEGA32.html)

4. Konfigurasi *Pin* AVR ATmega32




Gambar 4. Konfigurasi Pin ATmega32

(http://html.alldatasheet.com/html-pdf/77378/ATMEL/ATMEGA32/127/1/ATMEGA32.html)

Gambar 4 menunjukkan konfigurasi pin ATmega128, berikut penjelasannya:

a. VCC merupakan pin yang berfungsi sebagai pin masukan catudaya
b. GND merupakan pin ground
c. Port A (PA0.PA7) merupakan pin I/O dua arah dan bisa dikonfigurasikan juga untuk ADC
d. Port B (PB0.PB7) merupakan pin I/O dua arah dan pin fungsi khusus, yaitu *Timer/Counter*, komparator analog, dan SPI
e. Port C (PC0.PC7) merupakan pin I/O dua arah dan pin fungsi khusus, yaitu TWI (*Two Wire Interface*), komparator analog, dan *Timer Oscillator*

f. Port D (PD0.PD7) merupakan pin I/O dua arah dan pin fungsi khusus, yaitu komparator analog, interupsi eksternal, dan komunikasi serial

g. RESET merupakan pin yang digunakan untuk me-*reset* mikrokontroler

h. XTAL1 dan XTAL2 merupakan pin masukan *clock* eksternal

i. AVCC merupakan pin masukan tegangan untuk ADC

j. AREF merupakan pin masukan tegangan referensi ADC

## D. Perangkat Lunak BASCOM AVR

Bahasa BASIC atau bahasa dasar adalah salah satu bahasa pemograman tingkat tinggi yang dapat digunakan oleh pemula maupun yang sudah ahli. Bahasa BASIC terdiri dari kode-kode intruksi yang mempunyai tujuan menjalankan program *software*.

BASCOM AVR adalah pemrograman program BASIC *compiler* berbasis Windows untuk mikrokontroler keluarga AVR. BASCOM AVR merupakan pemrograman dengan tingkat tinggi BASIC yang dikembangkan dan dikeluarkan oleh MCS *Electronics*. Selain itu program hasil pemrograman yang telah dikompilasi langsung bisa di *download* ke mikrokontroler dengan *software* khusus *downloader* untuk *download* melalui SPI.

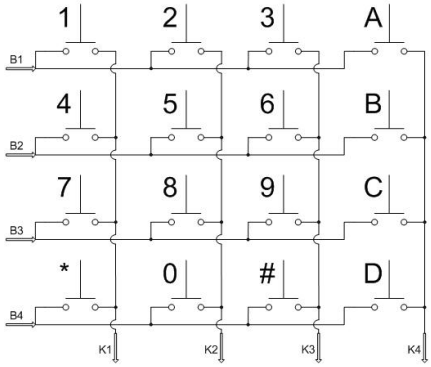
## E. LCD (*Liquid Crystal Display*)

*Liquid Crystal Display* (LCD) berfungsi sebagai indikator status dalam kalender digital. Modul LCD relatif jauh lebih sedikit memerlukan daya daripada modul – modul *display* dengan LED. Dengan mikrokontroler kita dapat menampilkan tampilan pada LCD agar dapat bekerja secara otomatis. Proyek akhir ini menggunakan LCD karakter 4 x 20, cara mengakses LCD harus melakukan konfigurasi pin dari LCD dengan pin I/O mikrokontroler tersebut. Berikut deskripsi pin pada LCD:

a. VSS adalah ground
b. VCC adalah + 5V power suplay
c. VEE adalah power suplay *source to control contrast*
d. RS adalah *Register select*: RS = 0 *to select* instruksi, *Command* register: RS =1 *to select* data reg
e. R/W adalah *Read/Write*: R/W =0 *for write*, R/W= 1 *for read*
f. E adalah *enable*
g. DB0 adalah 8-bit data bus
h. DB1 adalah 8-bit data bus
i. DB2 adalah 8-bit data bus
j. DB3 adalah 8-bit data bus
k. DB4 adalah 8-bit data bus
l. DB5 adalah 8-bit data bus
m. DB6 adalah 8-bit data bus
n. DB7 adalah 8-bit data bus

## *F. Keypad Matrix*

*Keypad Matrix* adalah tombol-tombol yang disusun secara maktriks (baris x kolom) sehingga dapat mengurangi penggunaan pin input. *Keypad matrix* berfungsi sebagai *interface* antara perangkat elektronik dengan manusia. Contohnya *keypad Matrix* 4×4 cukup menggunakan 8 pin untuk 16 tombol. Rangkaian tombol disusun secara horizontal membentuk baris dan secara vertikal membentuk kolom terlihat pada Gambar 5.




Gambar 5. Rangkaian Tombol *Keypad Matrix* 4x4

## *G. Memory Card*

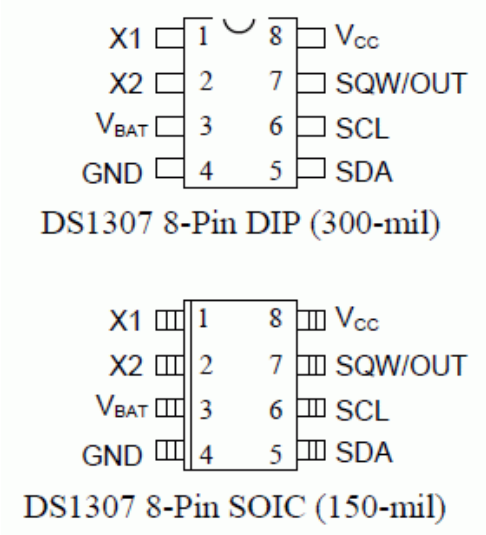
*Memory card* atau kartu memori merupakan alat yang berfungsi sebagai tempat penyimpanan data pada gadget seperti, kamera digital, PDA dan *handphone*. Ukuran dari kartu memori ini bermacam - macam yaitu 128MB, 512MB, 1GB, 32GB, dan seterusnya.

Contoh *memory card* yaitu SD card, bentuk SD card seukuran dengan MMC dengan kecepatan transfer data yang lebih cepat. SD card banyak

digunakan mulai dari *handphone*, kamera dan komputer. Pada proyek akhir ini menggunakan salah satu jenis SD card yaitu *micro* SD. Ukurannya lebih kecil daripada Mini SD dan banyak digunakan pada *handphone* sekarang.

## H. *Real Time Clock* (RTC)

*Real time clock* DS1307 memiliki kristal yang dapat mempertahankan frekuensinya dengan baik. IC tersebut dapat menghitung waktu (mulai detik hingga tahun) dengan akurat dan menjaga data waktu tersebut secara *real time*. Konsumsi daya RTC ini kurang dari 500nA menggunakan baterai cadangan dengan operasi osilator. RTC DS1307 terdapat pendeteksi otomatis kegagalan-daya (*power-fail*) dan rangkaian *switch.* Berikut daftar pin RTC DS1307 dan konfigurasi pin RTC DS1307 seperti Gambar 6.




Gambar 6. Konfigurasi *Pin* RTC DS1307

(http://html.alldatasheet.com/html-pdf/58481/DALLAS/DS1307/180/1/DS1307.html)

Keterangan Konfigurasi *pin* RTC DS1307 adalah sebagai berikut :

a. VCC adalah *power suplay*
b. X1, X2 adalah 32.768kHz *Crystal Connection*
c. VBAT adalah +3V *Battery Input*
d. GND adalah *ground*
e. SDA adalah serial data
f. SCL adalah *Serial Clock*
g. SQW/OUT adalah *Square Wave/Output Driver*

## *I. Seven Segment*

*Seven segment* merupakan komponen yang berfungsi sebagai penampil karakter angka dan karakter huruf. Dengan dilengkapi karakter titik (dot) yang sering dibutuhkan untuk karakter koma atau titik pada saat menampilkan suatu bilangan. Seven segment mempunyai tujuh buah penampil dasar dari LED (*Light Emiting Diode*) yang dinamakan karakter A-F dan karakter dot.

*Seven segment* mempunyai dua jenis rangkaian dasar yaitu *display seven segment common anoda* (CA) dan *common cathoda* (CC). Pada *display common anoda* untuk mengaktifkan karakter *display segment segment* diperlukan logika low (0) pada jalur A-F dan DP dan sebaliknya untuk *display seven segment common cathoda* (CA)

# BAB III

# PERANCANGAN DAN PEMBUATAN ALAT

Perancangan dari alat terdiri dari dua bagian yaitu perangkat lunak (*software*) dan perangkat keras (*hardware*). Perancangan perangkat keras meliputi sistem minimum mikrokontroler ATmega128, sistem minimum mikrokontroler ATmega32, RTC, LCD, *keypad matrix, seven segment,* dan *memori card.* Pada masing – masing komponen atau rangkaian harus dapat bekerja sesuai sistem yang diperintahkan oleh *software.*

Perangkat lunak (*software*) meliputi pengolahan data pada mikrokontroler untuk pembuatan *software* program menggunakan bahasa pemrograman Bascom AVR. Pada pembuatan *software* harus sesuai dengan urutan perintah agar terjadi kesinambungan antara *software* dan *hardware.*

## A. Identifikasi Kebutuhan

Dalam proyek akhir ini dapat dilakukan identifikasi kebutuhan sebagai berikut :

1. Perlu adanya pengumpulan data hari peringatan yang akan dimasukan sebagai inputan pada program *software.*
2. Dibutuhkan rangkaian catudaya sebagai sumber tegangan pada masing - masing rangkaian.
3. Dibutuhkan sistem minimum sebagai pengendali sistem pada kalender digital.

4. Dibutuhkan RTC (*real time clock*) sebagai penghitung waktu (mulai detik hingga tahun) dengan akurat dan menjaga data waktu tersebut secara *real time*.
5. Dibutuhkan bahasa pemrogaman sebagai pengaplikasian argoritma pada kalender nasional digital.
6. Dibutuhkan *memory card* sebagai penyimpan data memori.
7. Dibutuhkan unit penampil *display* untuk menampilkan pengoperasian kalender nasional digital.

## B. Analisis Kebutuhan

Berdasarkan dari identifikasi kebutuhan di atas maka diperoleh beberapa analisis kebutuhan terhadap pengembangan alat yang akan dibuat sebagai berikut:

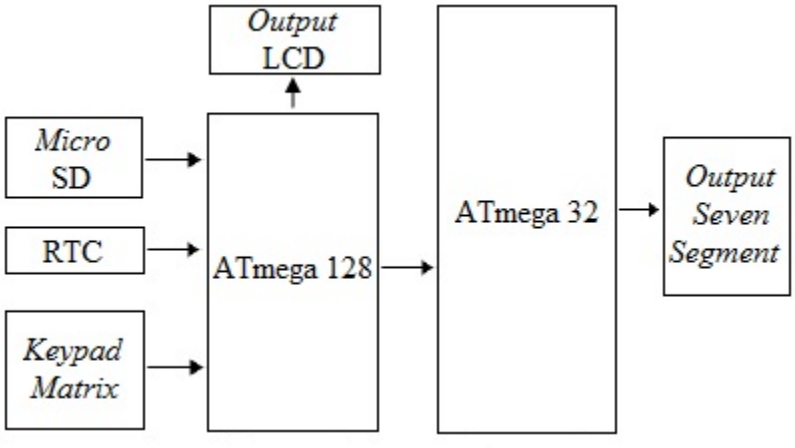
1. Rangkaian catu daya sebagai sumber tegangan untuk seluruh rangkaian sebesar 5 volt.
2. Rangkaian sistem minimum menggunakan ATmega128 sebagai pengontrol seluruh kinerja dari kalender digital sesuai dengan perintah program *software.*
3. Rangkaian sistem minimum menggunakan ATmega32 sebagai pengontrol dan pendukung kinerja dari kalender digital kemudian ditampilkan pada *display seven segment*.

8. Menggunakan RTC (*real time clock*) sebagai penghitung waktu (mulai detik hingga tahun) dengan akurat dan menjaga data waktu tersebut secara *real time*.
4. Menggunakan *micro SD* sebagai penyimpan data memori hari peringatan dan *event* tambahan.
5. Menggunakan bahasa BASIC sebagai pengaplikasian pengolahan data.
6. Menggunakan LCD 4x20 sebagai penampil utama seluruh kinerja dari kalender nasional digital.
7. Penambahan seven segment sebagai penampil tanggal, bulan dan tahun.

## C. Perancangan Sistem

1. Perancangan Blok Diagram Rangkaian

Perancangan blok diagram rangkaian berdasarkan analisis kebutuhan yang diperlukan alat. Susunan blok diagram rangkaian dapat dilihat pada Gambar 7.



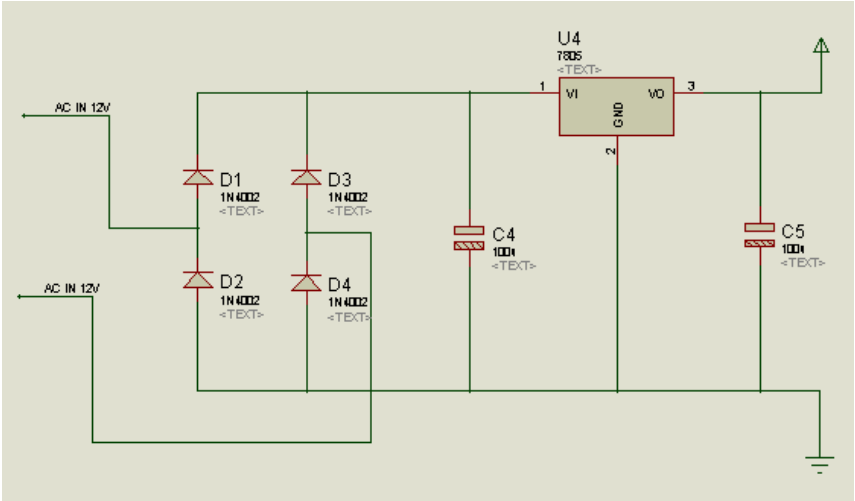

Gambar 7. Blok Diagram Perancangan Rangkaian

Keterangan blok diagram perancangan rangkaian pada Gambar 7, sebagai berikut :

a. Input RTC (*real-time clock*) DS1307 sebagai penghitung waktu dengan akurat dan menjaga data waktu secara *real time.*
b. Input *keypad matrix* 4x4 digunakan sebagai tombol untuk memilih menu dan menambahkan data yang akan tersimpan secara otomatis.
c. LCD sebagai monitor kegiatan sistem seperti jam, tanggal, hari peringatan, pasaran jawa, dan menu. LCD ini terhubung dengan PORT A pada mikrokontroler ATmega128.
d. *Seven segment* sebagai komponen pendukung untuk menampilkan tanggal, bulan dan tahun. *Seven segment* terhubung dengan PORT A dan D pada mikrokontroler ATmega32.

2. Perancangan Catu Daya

Rangkaian catu daya merupakan rangkaian yang digunakan sebagai sumber listrik pada setiap perangkat elektronika yang terdiri atas dioda dan kapasitor. Secara keseluruhan rangkaian catu daya berfungsi untuk menyearahkan tegangan AC sehingga menjadi DC, dan menstabilkan tegangan DC.

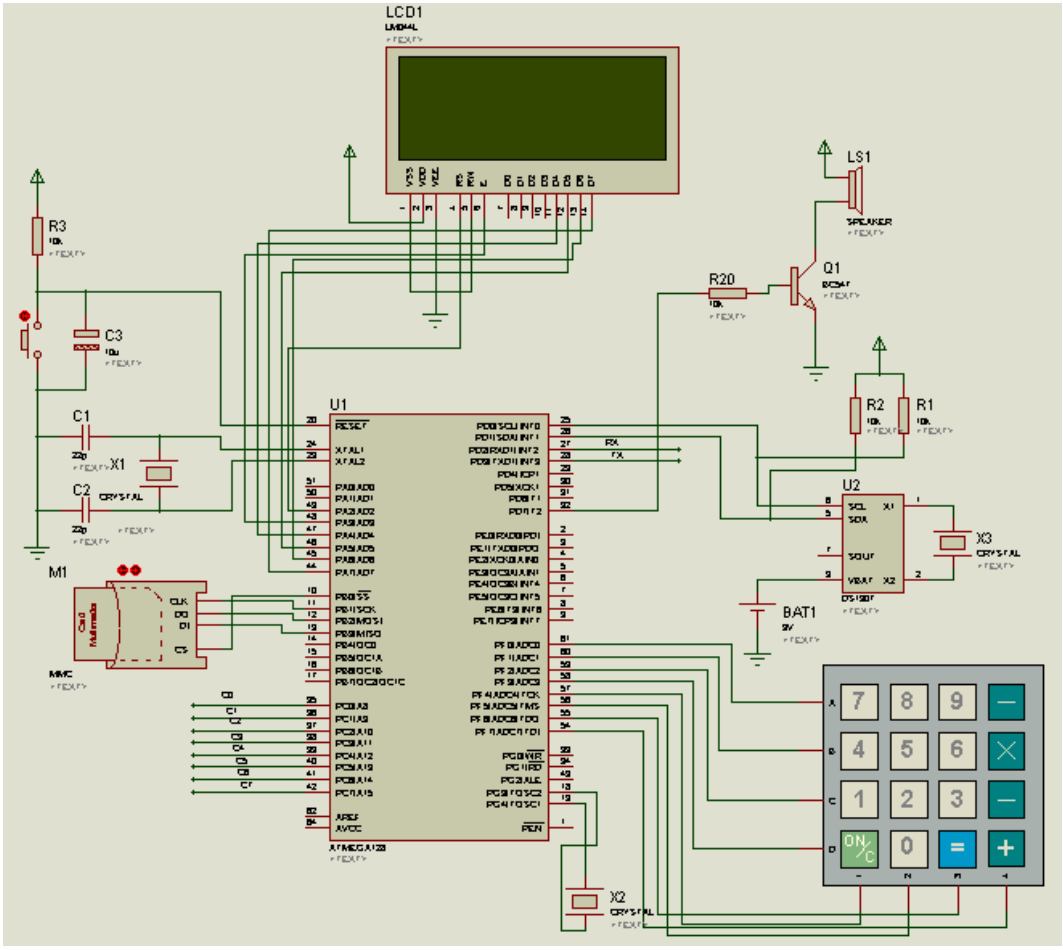



Gambar 8. Rangkaian Catu Daya

Gambar 8 menunjukkan rangkaian catu daya. Rangkaian catu daya mengeluarkan *output* 5V DC, sebagai catu sistem minimum mikrokontroler ATmega32, sistem minimum ATmega128, rangkaian RTC, LCD, dan *seven segment.*

3. Rangkaian Utama Mikrokontroler ATmega128

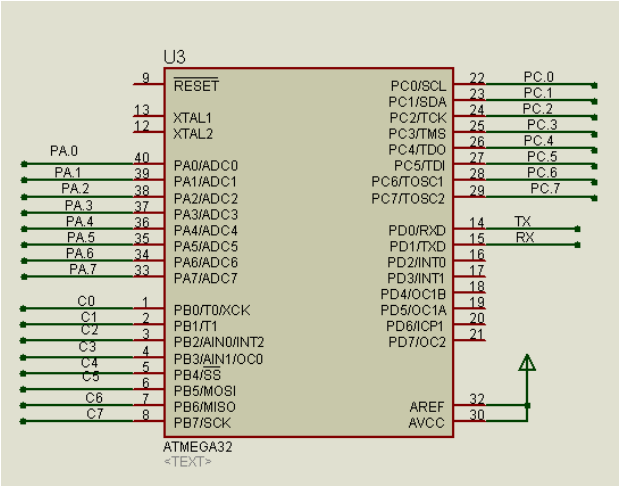
Rangkaian sistem minimum mikrokontroler ATmega128 merupakan rangkaian inti dari pembuatan kalender nasional digital. Alat ini dapat bekerja karena terdapat program software yang akan menjalankan sistem secara otomatis. Gambar 9 merupakan rangkaian utama mikrokontroler ATmega128.




Gambar 9. Rangkaian Utama Mikrokontroler ATmega128

4. Rangkaian Sistem Minimum Mikrokontroler ATmega32

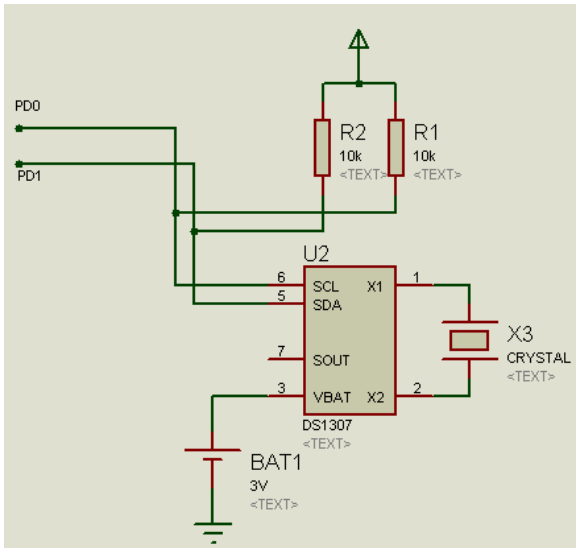
Rangkaian sistem minimum mikrokontroler ATmega32 berfungsi sebagai input untuk menampilkan *seven segment.* Port B mikrokontroler ATmega32 pada rangkaian alat ini terhubung dengan port C ATmega128. Gambar 10 merupakan rangkaian sistem minimum ATmega32.




Gambar 10. Rangkaian Sistem Minimum ATmega32

5. Rangkaian RTC (*Real Time Clock*)

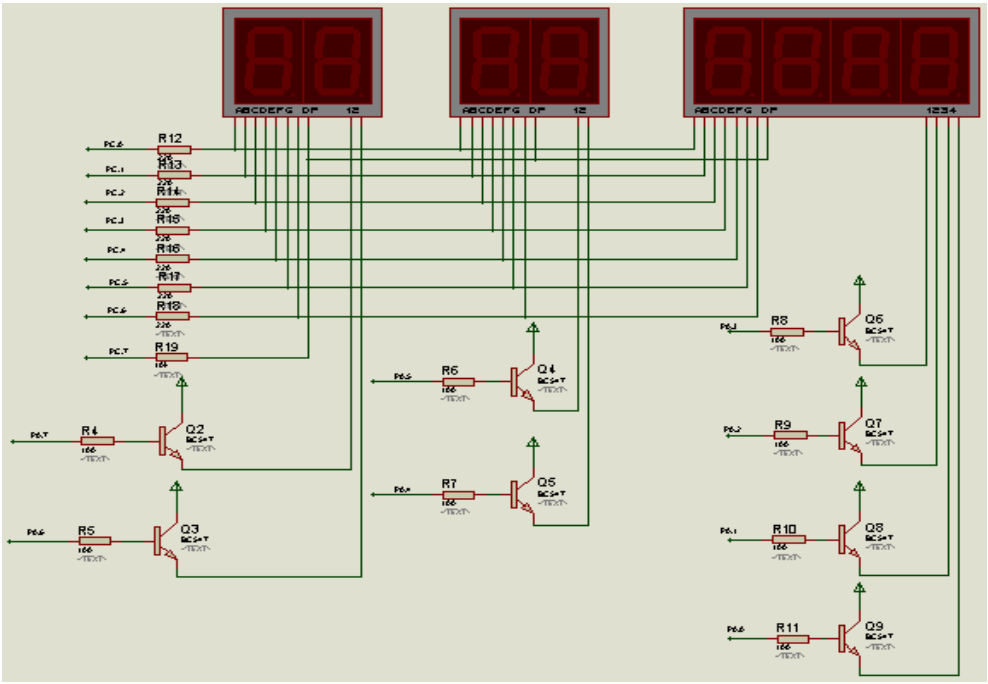
Rangkaian RTC (*real time clock*) berfungsi menghitung waktu (mulai detik hingga tahun) dengan akurat dan menjaga data waktu tersebut secara *real time*. RTC dilengkapi baterai sebagai penyuplai tegangan, sehingga jam tetap berjalan walaupun alat dalam keadaan mati. Rangkaian RTC (*real time clock*) dapat dilihat pada Gambar 11.




Gambar 11. Rangkaian RTC (*Real Time Clock*)

6. Rangkaian *Seven Segment*

Rangkaian *seven segment* pada kalender nasional digital berfungsi untuk menampilkan tanggal, bulan dan tahun. Rangkaian ini juga berfungsi sebagai komponen pendukung terutama untuk penglihatan jarak jauh agar terlihat lebih jelas. *Seven segment* tersebut terhubung dengan port A dan port C pada mikrokontroler ATmega32. Rangkaian *seven segment* dapat dilihat pada Gambar 12.




Gambar 12. Rangkaian *Seven Segment*

## D. Perancangan Perangkat Lunak (*Software*)

Perancangan perangkat lunak (*Software*) dilakukan dengan menentukan langkah kerja yang digambarkan dengan algoritma dan diagram alir (*flowchart*). Hal ini dimaksudkan agar memperoleh langkah – langkah

yang efektif dan efisien. Selanjutnya langkah yang telah diperoleh akan direalisasikan ke dalam sebuah program. Program tersebut akan di *download* ke dalam *flash memory* pada IC ATmega128.

Pembuatan kalender nasional digital menggunakan bahasa BASIC dengan program *software* BASCOM AVR. Pemrogram bahasa BASIC memberikan kemudahan bagi programmer untuk melakukan pemrograman disamping juga dilengkapi *simulator* yang cukup membantu dalam mensimulasikan hasil pemrograman.selain itu program hasil pemrograman yang telah dikompilasi langsung bisa di *download* ke mikrokontroler dengan BASCOM AVR.

Untuk mendesain rangkaian alat dan menganalisi program menggunakan program *software ISIS Proteus 7.7*. Program software yang berupa file berekstensi *.hex dimasukkan pada mikrokontroler agar dapat melakukan simulasi.

1. Algoritma

   Algoritma pembuatan program kalender nasional digital sebagai berikut:

   a. Mulai
   b. Inisialisasi Port I/O
   c. LCD menampilkan tampilan utama
   d. Apakah menu dipilih?
   e. Jika ya pilih menu 1 sampai dengan 7
      1) Jika menu 1 dipilih, simpan di EEPROM

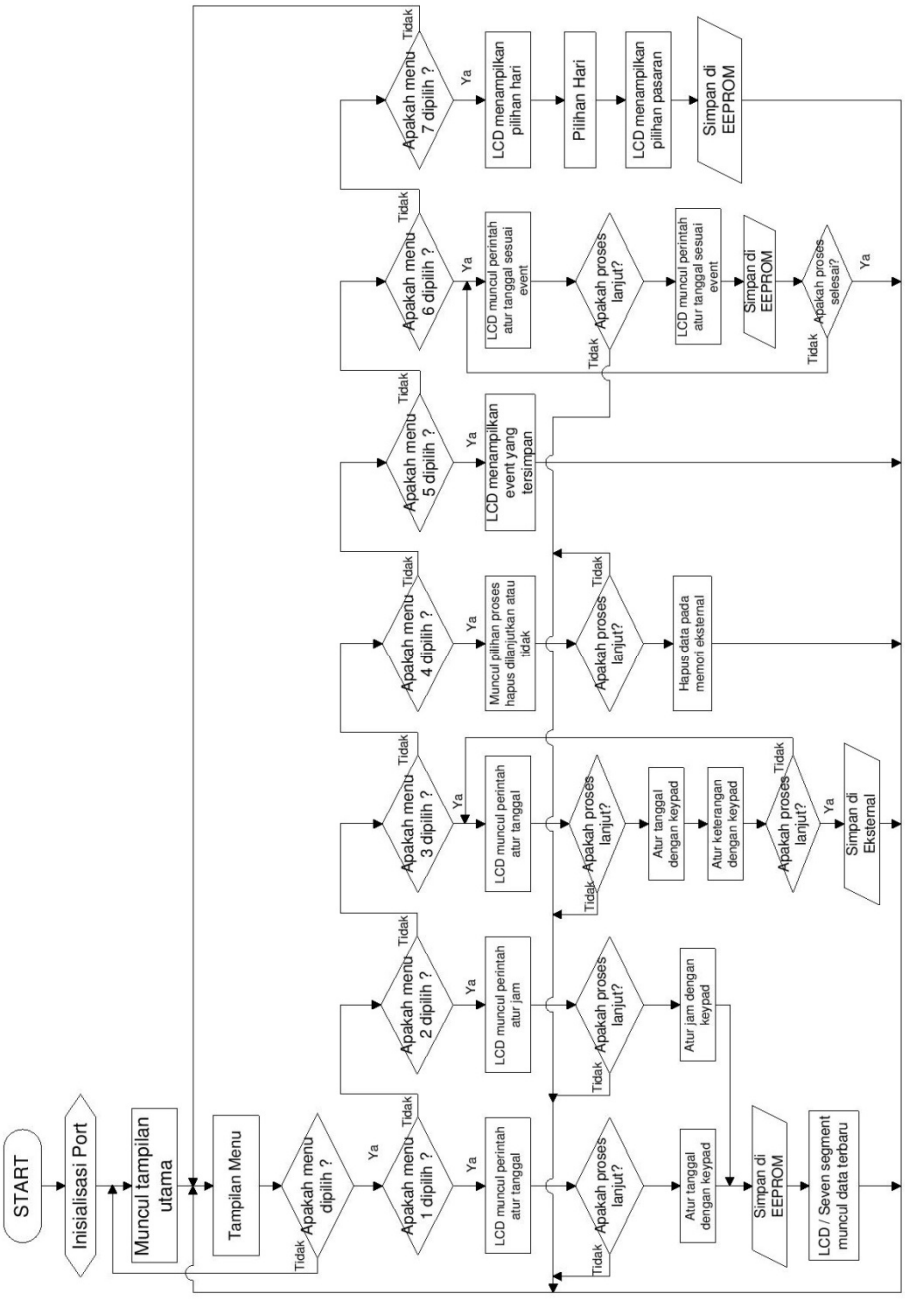
2) Jika menu 2 dipilih, simpan di EEPROM
3) Jika menu 3 dipilih, atur tanggal, tambah keterangan lalu simpan di EEPROM
4) Jika menu 4 dipilih, hapus lalu simpan di EEPROM
5) Jika menu 5 dipilih, muncul event
6) Jika menu 6 dipilih, atur tanggal lalu simpan di EEPROM
7) Jika menu 7 dipilih, atur hari, atur pasaran lalu simpan di EEPROM

f. Kembali ke tampilan menu

g. Kembali ke tampilan utama

2. Diagram Alir (*Flowchart*)

Diagram alir (*flowchart*) dapat dilihat pada Gambar 13, seperti berikut :




Gambar 13. Diagram Alir (*Flowchart*)

## E. Peralatan yang digunakan

Dalam pembuatan kalender nasional digital ini memerlukan beberapa alat yang harus disediakan, yaitu :

1. Solder
2. Setrika
3. Obeng set
4. Bor dan mata bor
5. Tang potong
6. *Cutter*
7. Gunting
8. Multimeter

Sedangkan bahan – bahan yang dibutuhkan, yaitu :

1. Komponen untuk catudaya dan sistem minimum, seperti RTC, IC LM 7805, IC ATmega128, IC ATmega32, LCD, *seven segment*, *keypad matrix*, dan komponen pendukung misalnya crystal 12 Mhz, resistor, transistor, kapasitor, dioda, LED,dll.
2. PCB polos dan PCB *dot*
3. Timah
4. Pelarut $FeCl_3$
5. Larutan *Thinner*
6. *Photo copy* desain PCB
7. Kabel
8. Amplas

9. Mur dan baut
10. *Box acrylic*
11. Spiser

## F. Langkah Pembuatan Alat

Pembuatan alat proyek akhir ini terdiri dari pembuatan *printed circuit board* (PCB) untuk rangkaian catu daya, rangkaian utama mikrokontroler ATmega128, dan rangkaian sistem minimum ATmega32, rangkaian *seven segment* serta pemasangan komponen, pengujian laporan, pembuatan box rangkaian, dan perakitan rangkaian pada box.

1. Pembuatan *Printed Circuit Board* (PCB)
   a. Pembuatan *layout* PCB

      Langkah awal pembuatan PCB adalah menggambar *layout* rangkaian dengan perangkat lunak. Perangkat lunak yang digunakan yaitu PCB Express.

   b. Penyablonan PCB

      Setelah *layout* selesai dibuat maka langkah selanjutnya yaitu penyablonan. Proses penyablonan dilakukan dengan cara sebagai berikut :

      1) Mencetak *layout* PCB yang telah dibuat pada kertas glossi.
      2) Membersihkan PCB menggunakan amplas dengan cara digosok-gosokkan.

3) Desain *layout* yang sudah dicetak pada kertas glossi dipindahkan atau disablonkan ke PCB polos dengan cara disetrika dengan panas sedang sampai dirasa gambar sudah terpindah semua.
4) Setelah gambar *layout* menempel semua pada PCB maka tunggu sebentar sampai dingin, kemudian bersihkan kertas yang menempel pada PCB dengan menggunakan air bersih.

c. Pelarutan dan Pengeboran PCB

Proses melarutkan PCB atau sering disebut juga pembuatan jalur pada PCB. Pada proses ini PCB yang sudah di sablon kemudian dilarutkan menggunakan cairan kimia $FeCL_3$ ( *Feri Chloride* ). Setelah dilarutkan ambil PCB dari cairan *Feri Chloride* dan bersihkan dengan air. Langkah selanjutnya yaitu pengeboran PCB, PCB di bor sesuai dengan titik titik yang telah ditentukan.

d. Pemasangan Komponen

Jika PCB selesai dibor, maka selanjutnya yaitu pemasangan seluruh komponen pada PCB, dengan urutan sebagai berikut :

1) Menyiapkan seluruh komponen yang dibutuhkan.
2) Memasang komponen dari ukuran terkecil dan pemasangan jumper terlebih dahulu.
3) Menyolder kaki komponen hingga semua komponen terpasang.
4) Memotong kaki komponen yang menggangu agar terlihat rapi.

2. Pengujian Rangkaian

Setelah komponen telah terpasang semua, langkah selanjutnya yaitu melakukan pengujian rangkaian. Hal ini perlu dilakukan agar kita mengetahui bahwa cara kerja rangkaian sudah baik atau belum.

3. Pembuatan *Box*

a. Perencanaan Ukuran

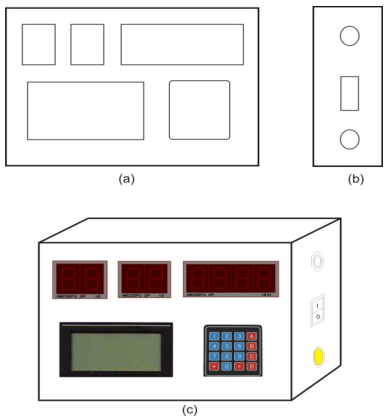
Panjang : 27,7 cm

Lebar : 16,5 cm

Tinggi : 5,5 cm

b. Pembuatan *Box*

Bahan yang dibutuhkan untuk membuat *box* adalah *arcylic*. Langkah pembuatannya yaitu membuat desain, memotong *arcylic*, pengeboran dan penggabungan. Gambar 14 menunjukkan desain *box* dari tampak depan, samping dan keseluruhan.




Gambar 14. Desain *Box*

(a) Tampak Depan (b) Tampak Samping Kiri (c) Tampak Keseluruhan

4. Pemasangan Rangkaian pada *Box*

Pemasangan rangkaian pada *box* dilakukan setelah semua bagian rangkaian selesai dirakit dan diuji sudah bisa bekerja dengan baik. Bagian rangkaian yang dirakit di dalam *box* yaitu rangkaian catu daya, rangkaian utama mikrokontroler ATmega128, dan rangkaian sistem minimum ATmega32, rangkaian *seven segment* dan LCD.

## G. Pengujian Alat

Pengujian alat dilakukan untuk mendapat data dari kalender nasional digital yang telah dibuat. Pengujian dilakukan dalam beberapa tahap sebagai berikut :

1. Uji fungsional

Pengujian yang dilakukan yaitu pengujian pada tiap bagian rangkaian. Pengujian ini dilakukan untuk memastikan pada setiap bagian rangkaian bekerja sesuai dengan fungsinya agar tidak terjadi kendala.

2. Uji seluruh sistem

Setelah menguji pada setiap bagian rangkaian hingga sesuai dengan fungsinya, langkah selanjutnya melakukan pengujian untuk seluruh sistem. Pengujian ini dilakukan dengan tujuan untuk mengetahui alat dapat bekerja sesuai fungsinya atau tidak. Hasil dari pembuatan alat tidak hanya mencapai keberhasilan, tetapi juga terdapat kekurangan. Kekurangan ini yang diharapkan dapat diperbaiki di lain kesempatan. Hasil pengujian dan pembahasan terdapat pada BAB IV.

# BAB IV

# PENGUJIAN DAN PEMBAHASAN

## A. Hasil Pengujian

Setelah pembuatan kalender nasional digital telah selesai, maka tahap selanjutnya yaitu pengujian *hardware* dan *software.* Proses pengujian alat dilakukan pada setiap bagian rangkaian, pengujian *software* dan sistem keseluruhan. Pengujian pada blok catu daya, sistem minimum mikrokontroler ATmega128, sistem minimum ATmega32, LCD dan *seven segment* menggunakan multimeter.

1. Rangkaian Catu Daya

   Pengukuran ini dilakukan untuk mengetahui besarnya keluaran tegangan dari catu daya. Tabel 1 adalah hasil dari pengukuran tegangan catu daya saat saklar *ON*.

   Tabel 1. Pengukuran Tegangan Catu Daya saat Saklar *ON*

| Pengukuran | Tegangan (V) |
|---|---|
| I | 5,05 |
| II | 5,04 |
| III | 5,06 |
| IV | 5,07 |

   Dari Tabel 1, diperoleh tegangan output rata – rata sebesar 5,055 V dengan cara perhitungan sebagai berikut :

Rata – rata = jumlah hasil percobaan : banyak percobaan

= 20,22 : 4

= 5,055 V

Besarnya tegangan *supply* minimum yang dibutuhkan untuk menyalakan sebuah LCD adalah sebesar 2,45 V, sedangkan untuk menyalakan *seven segment* sebesar 2,9 V. Kondisi LCD dan *seven segment* pada saat tegangan minimum yaitu menyala redup. Jika diberi tegangan sebesar 3 V, maka LCD dan *seven segment* dapat menyala terang dan jelas.

2. Rangkaian *Seven Segment*

*Seven segment* pada kalender digital berfungsi sebagai penampil tanggal, bulan dan tahun. Tabel 2 merupakan pengukuran tegangan BE (*Basis-Emittor*) transistor sebagai saklar pada *common anoda.*

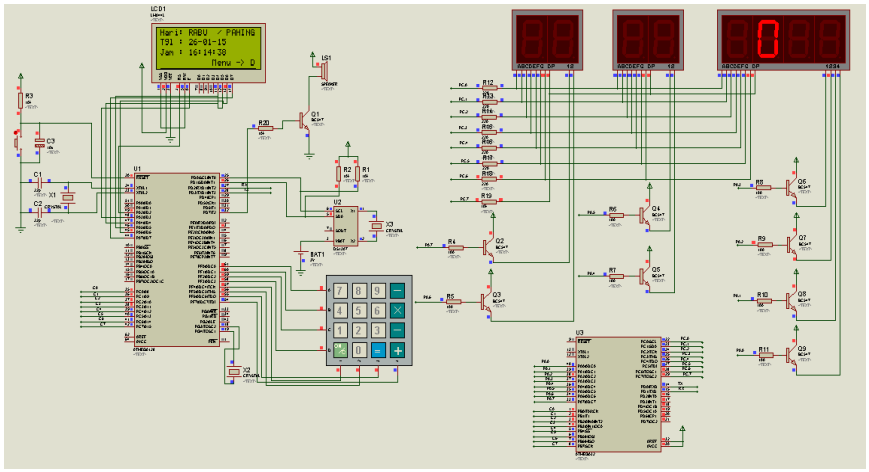
Tabel 2. Pengukuran Tegangan BE pada *Seven Segment* saat saklar *ON*

| No | Kolom *Seven segmen* ke - | Tegangan BE(V) |
|---|---|---|
| 1 | I | 0,4 |
| 2 | II | 0,6 |
| 3 | III | 0,6 |
| 4 | IV | 0,7 |
| 5 | V | 0,5 |
| 6 | VI | 0,5 |
| 7 | VII | 0,6 |
| 8 | VIII | 0,5 |

3. Pengujian pada Simulasi (*Software*)

Pengujian dilakukan dengan menggunakan *software* ISIS *Proteus 7.7* untuk mengetahui hasil analisa program yang telah dibuat. Mikrokontroller ATmega128 pada kalender nasional digital diprogram untuk mengendalikan *keypad matrix* dan tampilan LCD. Sedangkan mikrokontroller ATmega32 diprogram untuk mengendalikan tampilan *seven segment.*

*Keypad matrix* berfungsi sebagai *input* yang dihubungkan dengan PORTF pada mikrokontroler ATmega128. Sedangkan LCD berfungsi *output* penampil utama dan menu yang terhubung dengan PORTA pada mikrokontroler ATmega. Alat ini terdapat 2 (dua) mikrokontroler yaitu mikrokontroler ATmega128 dan mikrokontroler ATmega32. PORTB mikrokontroler ATmega 32 terhubung dengan PORTB mikrokontroler ATmega128. Sedangkan *seven segment* terhubung dengan PORTC mikrokontroler ATmega32. Gambar 17 merupakan tampilan simulasi dengan menggunakan *software* ISIS *Proteus 7.7.*



Gambar 15. Tampilan Simulasi pada *software* ISIS *Proteus 7.7*

Pada Gambar 15 dapat dilihat saat saklar *ON*, maka *seven segment* akan menyala secara bergantian setelah itu diikuti LCD menampilkan hari, tanggal, dan jam. Kemudian untuk memilih menu tekan D pada *keypad matrix*, pilih opsi menu sesuai keinginan.

4. Uji Keseluruhan Sistem

Pengujian keseluruhan sistem dilakukan untuk mengetahui kinerja alat setelah menggabungkan seluruh bagian dengan program *software*. Pada kalender digital nasional ini terdapat dua penampil yaitu LCD dan *seven segment.* Pengujian juga bertujuan untuk mengetahui apakah kedua penampil menunjukan tanggal yang sama atau tidak. Hasil pengujian tampilan kalender nasional digital dengan *seven segment* dapat terlihat pada Tabel 3.

Tabel 3. Pengujian Tampilan LCD dengan *Seven segment*

| **No** | **Tanggal** | **Hasil Tampilan LCD** | **Hasil Tampilan *Seven segment*** | **Keterangan** |
|---|---|---|---|---|
| 1. | 29 Januari 2015 |  |  | Sesuai |
| 2. | 2 Februari 2015 |  |  | Sesuai |

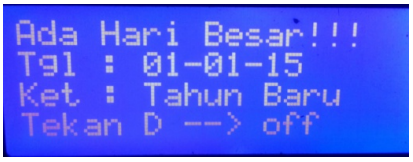
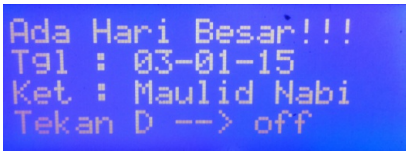
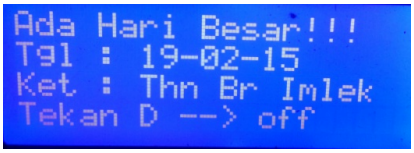
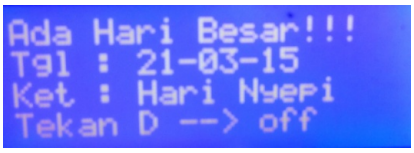
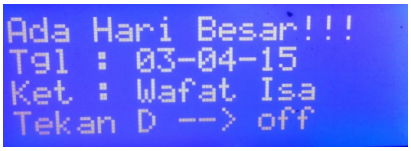
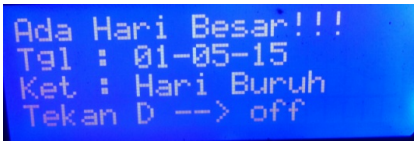
Kalender nasional digital juga dilengkapi dengan pemberitahuan hari-hari besar (*event*) yang akan muncul dengan bunyi alarm dan keterangan pada LCD. Dalam kalender nasional terdapat 2 (dua) macam hari peringatan yaitu tetap dan tidak tetap. Hal ini disebabkan hari peringatan pada setiap tahun ada yang berubah dan ada yang tidak berubah. Tabel 4 merupakan daftar hari peringatan pada tahun 2015.

Tabel 4. Daftar Hari Peringatan Tahun 2015

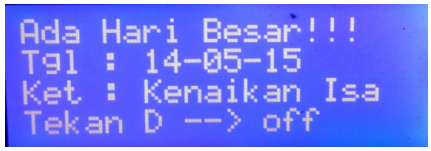
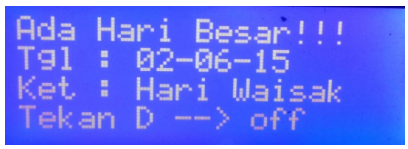
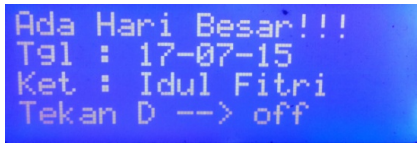
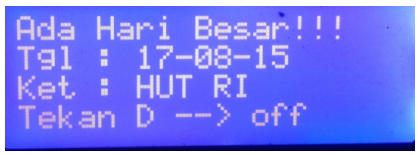
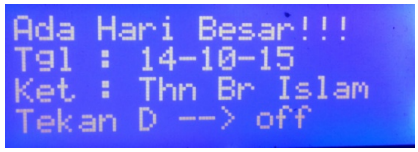
| Tahun 2015 | | | |
|---|---|---|---|
| **Bulan** | **Tanggal** | **Hari Peringatan** | **Keterangan** |
| Januari | 1 | tahun baru masehi | Tetap |
| | 14 | maulid nabi Muhammad SAW | Tidak Tetap |
| Februari | 19 | tahun baru imlek | Tidak Tetap |
| Maret | 21 | Nyepi | Tidak Tetap |
| April | 3 | wafat isa almasih | Tidak Tetap |
| Mei | 1 | hari buruh | Tetap |
| | 14 | kenaikan isa almasih | Tidak Tetap |
| | 16 | isra' mi'raj | Tidak Tetap |
| Juni | 2 | Waisak | Tidak Tetap |
| Juli | 17 | idul fitri | Tidak Tetap |
| Agustus | 17 | hut RI | Tetap |
| September | 24 | idul adha | Tidak Tetap |
| Oktober | 25 | tahun baru hijriyah | Tidak Tetap |
| November | | | |
| Desember | 25 | Natal | Tetap |

Pengujian ini dilakukan untuk mengetahui apakah kalender digital menampilkan hari besar (*event*) sesuai dengan kalender analog. Hasil pengujian ketepatan hari besar (*event*) antara kalender analog dengan kalender nasional digital dapat dilihat pada Tabel 4.

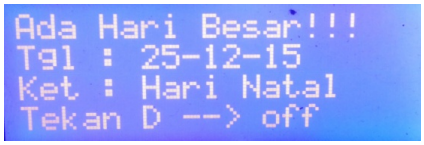
Tabel 5. Ketepatan antara Kalender Analog dengan Kalender Digital

| No | Tanggal | Hari Besar | Hasil Output pada LCD | Keterangan |
|---|---|---|---|---|
| 1. | 1 Januari 2015 | Tahun baru masehi |  | Sesuai |
| 2. | 3 Januari 2015 | Maulid Nabi Muhammad SAW |  | Sesuai |
| 3. | 19 Februari 2015 | Tahun baru Imlek |  | Sesuai |
| 4. | 21 Maret 2015 | Hari Raya Nyepi |  | Sesuai |
| 5. | 3 April 2015 | Wafat Isa Al-Masih |  | Sesuai |
| 6. | 1 Mei 2015 | Hari Buruh internasional |  | Sesuai |

(Lanjutan)

| | | | | |
|---|---|---|---|---|
| 7. | 14 Mei 2015 | Kenaikan Isa Al-Masih |  | Sesuai |
| 8. | 16 Mei 2015 | Isra' Mi'raj Nabi Muhammad SAW |  | Sesuai |
| 9. | 2 Juni 2015 | Waisak |  | Sesuai |
| 10. | 17 Juli 2015 | Idul Fitri 1436 H |  | Sesuai |
| 11. | 17 Agustus 2015 | Hut Kemerdekaan RI |  | Sesuai |
| 14. | 24 September 2015 | Idul Adha 1436 H |  | Sesuai |
| 15. | 14 Oktober 2015 | Tahun baru Hijriah 1437 |  | Sesuai |

**(**Lanjutan**)**

| | | | | |
|---|---|---|---|---|
| 16. | 25 Desember 2015 | Natal |  | Sesuai |

## B. Pembahasan

Setelah melakukan pengujian pada bagian-bagian rangkaian dan komponen dari alat ini, diperoleh bahwa keseluruhan rangkaian dapat bekerja dengan baik dan sesuai fungsinya. Hasil dari pengukuran tersebut terdapat perbedaan antara hasil pengukuran dengan perhitungan secara teori maupun *datasheet* komponen. Ada beberapa faktor yang mempengaruhi perbedaan tersebut, antara lain toleransi tegangan atau nilai komponen dari pabrik, nilai komponen yang tidak sesuai dengan label, kondisi alat ukur,dan kesalahan pengukuran. Namun hal ini tidak mengganggu kinerja dari kalender nasional digital. Berikut pembahasan beberapa fungsi rangkaian yang telah di uji:

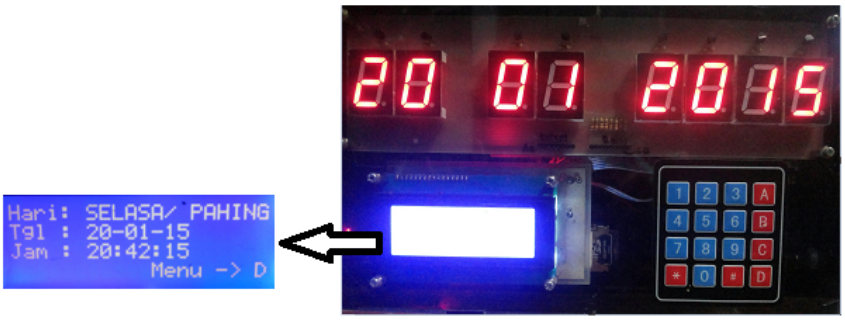
1. Perangkat Keras (*Hardware*)
   a. Rangkaian Catu Daya

      Pada perencanaan catu daya *output* yang dihasilkan alat ini sebesar 5 V, sedangkan pada pengukuran diperoleh sebesar 5, 05 V. Akan tetapi dari hasil pengukuran catu daya dapat disimpulkan bahwa catu daya dapat bekerja dengan baik meskipun mengalami sedikit kenaikan tegangan *output*. Kemungkinan hal ini disebabkan oleh kondisi alat

ukur yang kurang baik atau nilai komponen yang tak sesuai dengan label.

b. Sistem Minimum ATmega128 dan LCD

Pengujian pada sistem minimum ATmega128 dan LCD menggunakan *software* dan *hardware*. Proses pengujian ini yaitu dengan memasukkan program *software* pada alat dan pengecekan port untuk LCD agar dapat berfunsi sebagai tampilan. Hasil dari pengujian sistem minimum ATmega128 dan *LCD* dapat dilihat pada Gambar 18.




Gambar 16. Pengujian Sistem Minimum ATmega128 dan LCD

Pada pengujian tersebut seperti pada gambar 16 membuktikan bahwa sistem minimum ATmega128 berfungsi dengan baik. Sedangkan LCD dapat beroperasi dengan baik dan menunjukkan hari, pasaran, tanggal, jam, dan menu.

c. Sistem Minimum ATmega32 dan *Seven Segment*

Proses pengujian sistem minimum ATmega32 dan *seven segment* sama seperti proses pengujian sistem minimum ATmega128 dan LCD. Gambar 18 merupakan pengujian sistem minimum ATmega32 dan *seven segment.*


Gambar 17. Pengujian sistem Minimum ATmega32 dan *Seven Segment*

Bisa dilihat pada gambar 17, bahwa pengujian sistem minimum ATmega32 dapat berfungsi dengan baik sesuai program yang telah dibuat. Begitu juga dengan *seven segment* sudah berfungsi sesuai dengan rancangan atau menunjukan tanggal, bulan, tahun yang sesuai.

d. Perbandingan Ketepatan Tanggal pada LCD dan *seven segment*

Pada perbandingan tanggal antara LCD dengan *seven segment* dapat diperoleh hasil berdasarkan tabel 3 yang menunjukan bahwa secara otomatis kedua penampil memunculkan tanggal yang sudah sesuai. Hal ini disebabkan adanya RTC pada kalender digital yang memakai suplai baterai sendiri untuk menjaga ketepatan tanggal dan waktu yang ditampilkan di LCD.

2. Perangkat Lunak (*Software*)

Perangkat lunak (*Software*) pada proyek akhir ini menggunakan bahasa basic dengan memanfaatkan aplikasi *compiler* BASCOM AVR. Pembuatan program *software* dirancang menggunakan diagram alir (*flowchart*). Kemudian memasukkan program ke dalam mikrokontroler ATmega128 dan ATmega32. Program yang telah dibuat dapat menjalankan semua perintah sehingga alat ini dapat bekerja sesuai rancangan. Berikut ini adalah sebagian program pada mikrokontroler ATmega128 :

```
$regfile = "m128def.dat"
```

Maksud dari intruksi di atas merupakan pengarah - pengarah preprosesor bahasa BASIC yang memerintahkan untuk meyisipkan file lain, dalam hal ini adalah file m128def.dat yang berisi deklarasi register dari mikrokontroler ATmega128.

```
$crystal = 11059200
```

Nilai frekuensi kristal sebesar 11059200 Hz.

```
$baud = 9600
```

Intruksi ini menyatakan bahwa komunikasi serial dengan baudrate 9600.

```
$swstack = 128
```

Ruang yang tersedia untuk memanggil routine dalam bahasa mesin pada perangkat lunak yaitu sebesar 128.

```
$framesize = 128
```

Ukuran frame yang digunakan sebesar 128.

```
Enable Interrupts
```

Intruksi di atas menyatakan bahwa interupt aktif memungkinkan mikrokontroler untuk menanggapi *request* dari pengguna.

```
Declare Sub Load_mmc
Declare Sub File_baru
Declare Sub Selesai
```

Pada bagian ini mendeklarasi variabel – variabel yang digunakan.

```
Config Lcdpin = Pin , Rs = Porta.2 , E = Porta.3 ,
Db4 = Porta.4 , Db5 = Porta.5 , Db6 = Porta.6 ,
Db7 = Porta.7
```

Intruksi tersebut melakukan inisialisasi pin pada LCD, pada bagian ini mengkonfigurasikan pin yang ada di LCD terhubung dengan PORT mikrokontroler ATmega128.

```
Config Lcd = 20 * 4
```

Bagian ini mendefinisikan LCD memiliki 20 baris dan 4 kolom.

```
Const Ds1307w = &HD0
Const Ds1307r = &HD1
```

Intruksi di atas melakukan inisialisasi *socket* mikrokontroler, pada bagian ini mendefinisikan konstanta – konstanta soket yang digunakan.

```
Dim Keypad As Byte
Dim Ff As Byte , B As Byte
```

Kode di atas menunjukkan bahwa dalam program menentukan beberapa penentuan variabel yaitu variabel ‘keypad’, ‘Ff’, dan ‘B’ bertipe data byte.

```
Do
   Gosub Baca_keypad
   Gosub Getdatetime
   Gosub Cek
   Locate 1 , 1
   Lcd "Hari: " ; Hri ; "/ " ; Psr
   Locate 2 , 1
   Lcd "Tgl : " ; Date$
   Locate 3 , 1
   Lcd "Jam : " ; Time$
   Locate 4 , 1
   Lcd "           Menu -> D"
   If Keypad = 15 Then Gosub Menu
Loop
```

Intruksi ini berisi perulangan dengan melakukan pemanggilan menggunakan sub program. Intruksi tersebut menginisialisasi *keypad* dan memunculkan data waktu dan tanggal pada LCD. Penentuan letak

hari/pasaran pada kolom 1, tanggal pada kolom 2, jam pada kolom 3 dan menu pada kolom 4.

```
Select Case Keypad
         Case 1 : Gosub Atur_tanggal
         Case 2 : Gosub Atur_waktu
         Case 3 : Gosub Set_agenda
         Case 4 : Gosub Hapus_agenda
         Case 11 : Gosub Lanjut
         Case 12 : Gosub Mulai
         Case 15 : Gosub Menu
      End Select
```

Perintah tersebut digunakan untuk pengambilan keputusan terhadap banyak kondisi.

3. Kerja Alat secara Keseluruhan

Kalender nasional digital berbasis mikrokontroler ATmega128 ini bekerja berdasarkan intruksi yang telah dibuat dan dimasukkan ke dalam mikrokontroler ATmega128. Mikrokontroler ATmega berfungsi sebagai pengendali *keypad matrix*, tampilan LCD, dan tampilan *seven segment.* Berikut ini unjuk kerja dari kalender nasional digital secara keseluruhan.

a. Pada kalender nasional digital tersedia *keypad matrix* yang berfungsi sebagai *input* dan untuk memilih menu yang ada pada LCD. Menu yang ada pada LCD yaitu atur tanggal, atur waktu, tambah hari besar, hapus hari besar, lihat hari besar, setting STD, dan atur hari/ pasaran.
b. Saat tombol saklar *ON/OFF* ditekan, LCD akan menyala lalu di ikuti nyala *seven segment*. Kemudian LCD akan menampilkan hari/pasaran, tanggal, jam dan menu.
c. Jika ingin memilih menu tekan tombol D, maka akan muncul 7 (tujuh) opsi menu.

d. Setelah LCD menampilkan beberapa pilihan menu, *user* dapat memilih salah satu menu yang tersedia. Jika ingin mengatur tanggal tekan tombol 1, maka akan ada perintah untuk memasukkan data tanggal. Setelah memasukkan data tanggal, tekan tombol D untuk menyimpan. Untuk membatalkan penyimpanan tekan tombol A.

e. Apabila ingin mengatur waktu tekan tombol 2, kemudian masukkan data jam yang sesuai. Setelah selesai memasukkan data waktu yang sesuai, tekan tombol D untuk menyimpan. Untuk membatalkan penyimpanan tekan tombol A.

f. Jika ingin menambah *event*/hari besar tekan tombol 3, maka LCD akan menampilkan 'agenda baru' dan terdapat nomor urut agenda yang akan ditambahkan. Kemudian atur tanggal dengan memasukkan data tanggal, setelah selesai tekan D. Lalu beri keterangan agenda tersebut, setelah selesai tekan D. Pada LCD akan muncul agenda yang sudah tersimpan, kemudian tekan tombol D untuk kembali ke menu.

g. Apabila ingin menghapus agenda yang sudah tersimpan tekan tombol 4, maka akan muncul 'Anda Yakin??'. Pertanyaan tersebut untuk mengonfirmasi apakah akan dilanjutkan menghapus atau tidak. Dan di bawah pertanyaan tersebut terdapat dua perintah, jika yakin akan dihapus tekan D dan jika tidak jadi dihapus tekan A. Setelah menekan salah satu tombol, tampilan LCD akan kembali ke menu.

h. Untuk melihat agenda yang sudah tersimpan, *user* dapat menekan tombol 5. Kemudian akan muncul semua data angenda yang sudah tersimpan, untuk kembali ke menu tekan tombol D.

i. Menu Setting STD berfungsi sebagai penyimpanan hari – hari besar nasional yang standar seperti yang ada di kalender analog. Diantaranya tahun baru, maulid nabi, natal, idul fitri, idul adha, nyepi, dan lain- lainnya. Akan tetapi seperti hari besar keagamaan pada setiap tahun tidak selalu sama, maka setiap tahun harus diberi masukkan data tanggal pada hari – hari besar tersebut. Untuk mengatur hari besar tekan tombol 6, kemudian akan muncul keterangan hari besar yang akan dimasukkan data tanggal. Setelah memasukkan data tanggal tekan tombol D untuk menyimpan, ulangi penyimpanan data tersebut sampai selesai lalu tekan D.

j. Jika ingin mengatur hari dan pasaran tekan tombol 7, kemudian pilih hari dengan menekan tombol pilihan yang ada. Setelah memilih hari akan muncul pilihan pasaran, lalu tekan tombol pilihan pasaran. Kemudian LCD akan kembali menampilkan menu.

k. Apabila sudah tidak memilih menu lagi tekan tombol A, LCD akan kembali menampilkan tampilan utama.

l. Tombol *reset* digunakan apabila sewaktu – waktu terjadi *error*.

m. Untuk mematikan seluruh sistem tekan tombol saklar *ON/OFF*.

# BAB V

# KESIMPULAN DAN SARAN

## A. Kesimpulan

Berdasarkan dari pembuatan proyek akhir ini dapat diperoleh kesimpulan sebagai berikut :

1. Kalender nasional digital berbasis mikrokontroler ATmega128 dirancang dari perangkat keras (*hardware*) yaitu rangkaian catu daya, rangkaian sistem minimum ATmega128, rangkaian sistem minimum ATmega32, rangkaian RTC, rangkaian *seven segment*, speaker, dan LCD. *Input* alat ini dari *keypad matrix*, sedangkan *output* berupa tampilan pada LCD dan *seven segment.*
2. Kalender nasional digital berbasis mikrokontroler ATmega128 dirancang dari perangkat lunak (*software*) yang digunakan dalam merancang kalender nasional digital adalah bahasa BASIC dengan program *software* BASCOM AVR. Perangkat lunak lainnya menggunakan ISIS *Proteus 7.7* untuk mendesain dan sebagai simulator rangkaian.
3. Unjuk kerja dari kalender nasional digital dimulai dengan pasang kabel *power* ke alat dan sambungkan dengan sumber tegangan AC. Kemudian mulai dengan menekan saklar *on/off* untuk menghidupkan alat.Setelah saklar ditekan, LCD akan menampilkan tampilan utama dan *seven segment* akan menampilkan tanggal, bulan, serta tahun.

Tombol *keypad matrix* berfungsi sebagai *inputan* untuk memilih menu dan mengatur tampilan yang akan muncul pada *display*. Pada tampilan utama pada LCD terdapat perintah jika ingin ke menu tekan huruf D. Kemudian akan muncul 7 menu yaitu atur tanggal, atur waktu, tambah hari besar, hapus hari besar, lihat hari besar, setting standar, dan atur hari/pasaran. Pilih menu sesuai keinginan atau yang akan dituju dengan menekan *keypad.* Jika ingin menyimpan data, maka tekan huruf D pada *keypad*. Setelah menyimpan data, tampilan LCD akan kembali ke pilihan menu. Jika ingin kembali ke tampilan utama, maka tekan huruf A pada *keypad*. Tombol *reset* digunakan untuk mengatur ulang sistem dari awal, apabila terjadi *eror* pada alat.

## B. Keterbatasan Alat

Pada pembuatan alat ini ada pula keterbatasannya, yaitu :

1. Untuk hari besar keagamaan tidak dapat secara otomatis memperbarui setiap tahunnya, jadi harus di input setiap tahunnya dengan *keypad matrix.*
2. *Keypad* huruf sebagai *input* masih terbatas.
3. Dikarenakan program *software* yang cukup banyak dan kapasitas memori yang terbatas, maka proses pengoperasian dan penyimpanan data memerlukan waktu yang relatif lama.
4. Kalender nasional digital ini belum menggunakan *output* suara atau modul mp3 untuk memberitahukan *event* yang ada.

## C. Saran

Dalam pembuatan proyek akhir ini tentu saja terdapat kekurangan sehimgga perlu pengembangan guna menyempurnakan proyek akhir. Oleh karena itu penulis memberikan saran sebagai berikut :

1. Pengembangan selanjutnya alat ini dapat dirancang secara otomatis, sehingga untuk tiap tahunnya tidak perlu memperbarui.
2. Guna pengembangan selanjutnya *keypad* pada alat ini dapat diganti dengan *keypad* huruf yang lengkap dari A sampai dengan Z.
3. Pengembangan selanjutnya mengganti memori dengan kapasitas yang memuat lebih banyak.
4. Kalender nasional digital ini masih menggunakan keluaran bunyi yang bersifat *monotone*, sebaiknya untuk pengembangan selanjutnya menggunakan modul mp3 yang keluarannya rekaman suara.

# DAFTAR PUSTAKA


ATMEL. (2011). *Datasheet ATmega128.* Diakses dari http://html.alldatasheet.com/html-pdf/56260/ATMEL/ATMEGA128/128/1/ATMEGA128.html pada tanggal 24Agustus 2014

ATMEL. (2003). *Datasheet ATmega32.* Diakses dari http://html.alldatasheet.com/html-pdf/77378/ATMEL/ATMEGA32/127/1/ATMEGA32.html pada tanggal 6 September 2014

DALLAS. (2000). *Datasheet RTC DS1307.* Diakses dari http://html.alldatasheet.com/html-pdf/58481/DALLAS/DS1307/180/1/DS1307.html pada tanggal 15 September 2014

Kurniawan, Fitra Mega. (2014). *Jam Digital Untuk Tunanetra Dengan Output Suara Menggunakan Pengendali Jarak Jauh Berbasis Mikrokontroler Atmega16.*Yogyakarta : Fakultas Teknik Universitas Negeri Yogyakarta

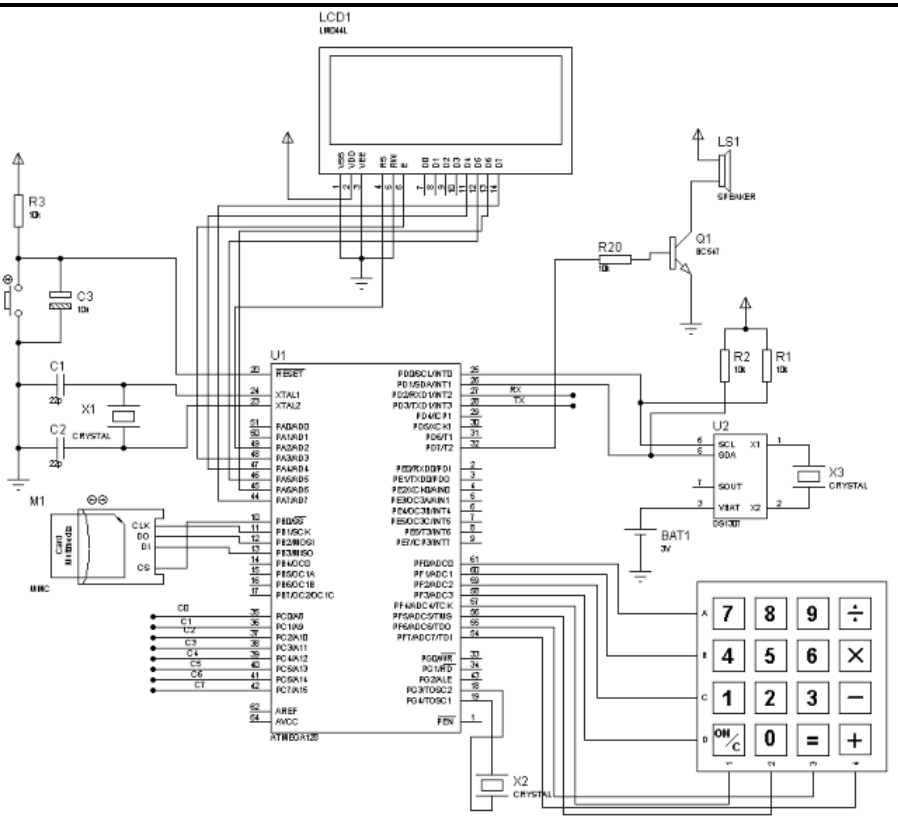
Pratama, Fery. (2013). *Robot Pemadam Api Berkaki Enam Berbasis Mikrokontroller Atmega32 Dan Atmega128.* Yogyakarta : Fakultas Teknik Universitas Negeri Yogyakarta

Tim Penyusun. (2013). *Pedoman Proyek Akhir D3*. Yogyakarta: Fakultas Teknik Universitas Negeri Yogyakarta.

Iswanto. (2008). *Design dan Implementasi Sistem Embedded Mikrokontroller ATmega8535 dengan Bahasa Basic*. Yogyakarta: Gava Media

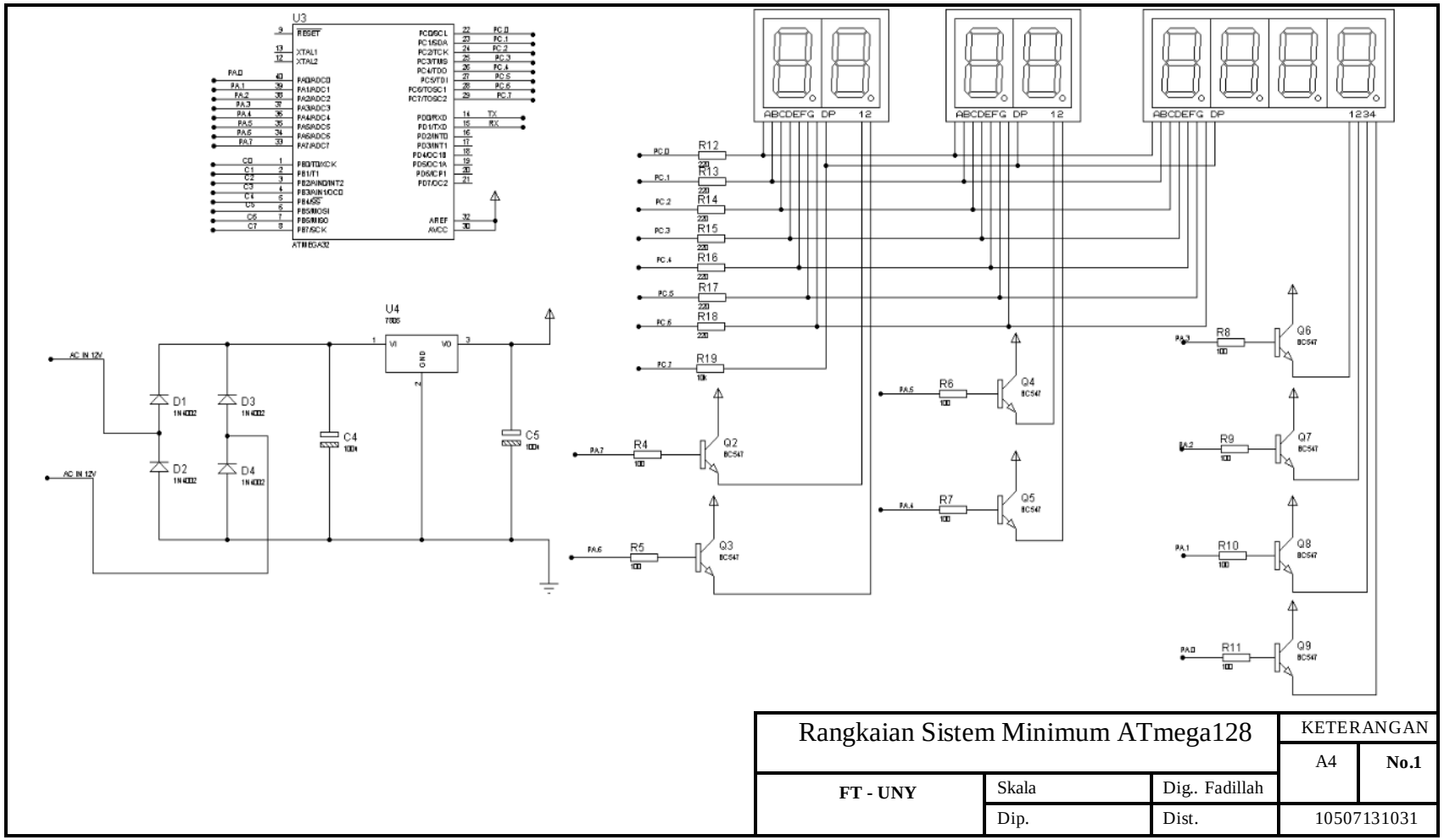
# LAMPIRAN

Lampiran 1. Rangkaian Sistem Minimum ATmega128



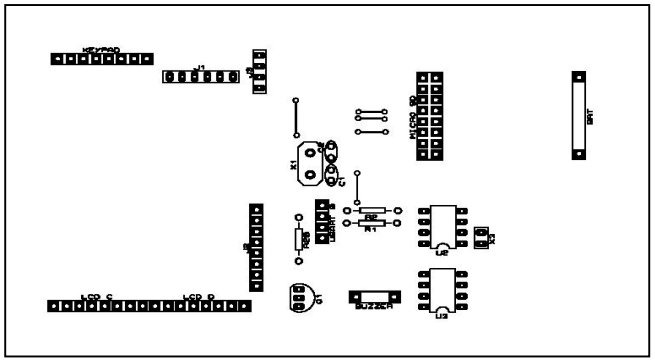

| Rangkaian Sistem Minimum ATmega128 | | | KETERANGAN | |
|---|---|---|---|---|
| | | | A4 | **No.1** |
| **FT - UNY** | Skala | Dig.. Fadillah | | |
| | Dip. | Dist. | 10507131031 | |

Lampiran 2. Rangkaian Sistem Minimum ATmega32

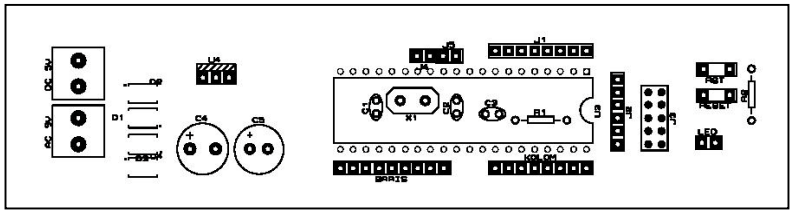



| Rangkaian Sistem Minimum ATmega128 | | | KETERANGAN | |
|---|---|---|---|---|
| | | | A4 | **No.1** |
| **FT - UNY** | Skala | Dig.. Fadillah | | |
| | Dip. | Dist. | 10507131031 | |

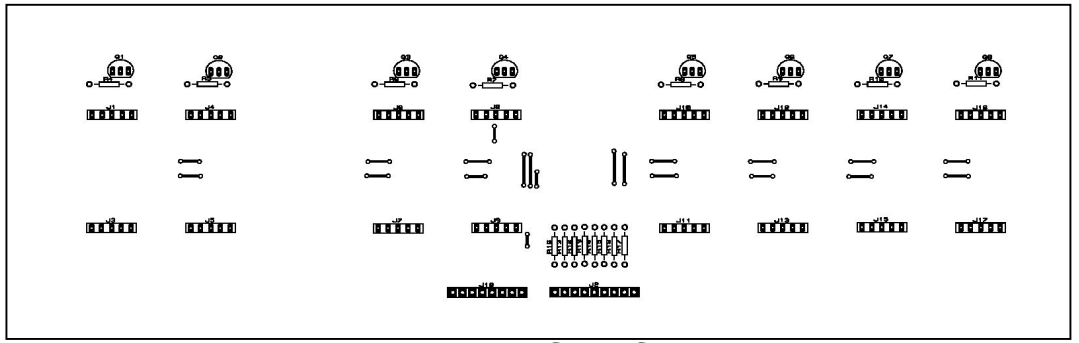
Lampiran 3. Layout Komponen



Layout Komponen Sismin ATmega128 dan RTC



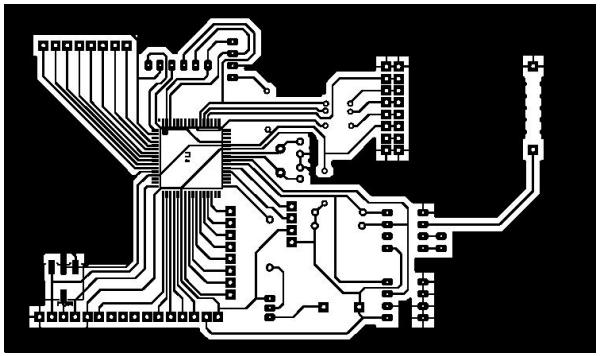
Layout Komponen Sismin ATmega32



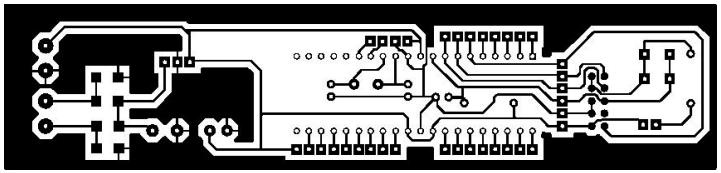
Layout Komponen Seven Segment

| LAYOUT KOMPONEN | | | KETERANGAN | |
|---|---|---|---|---|
| | | | A4 | **No.2** |
| **FT - UNY** | Skala | Dig.. Fadillah | | |
| | Dip. | Dist. | 10507131031 | |

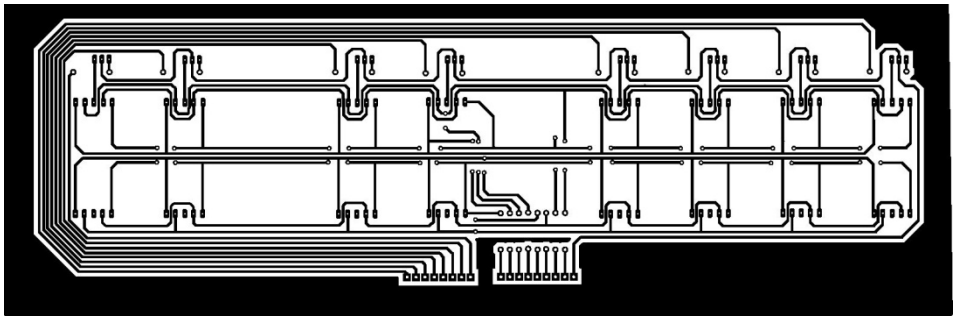
Lampiran 4. Layout PCB



Layout PCB Sismin ATmega128 dan RTC



Layout PCB Sismin ATmega32



Layout PCB Seven Segment

| | | | | |
|---|---|---|---|---|
| LAYOUT PCB | | | KETERANGAN | |
| | | | A4 | **No.3** |
| **FT - UNY** | Skala | Dig.. Fadillah | | |
| | Dip. | Dist. | 10507131031 | |

Lampiran 5. Program BASCOM AVR pada Mikrokontroler ATmega128

```
$regfile = "m128def.dat"
$crystal = 11059200
$baud = 9600
$hwstack = 128
$swstack = 128
$framesize = 128

Enable Interrupts

Declare Sub Load_mmc
Declare Sub File_baru
Declare Sub Selesai

Config Lcdpin = Pin , Rs = Porta.2 , E = Porta.3 , Db4 = Porta.4 , Db5 = Porta.5 ,
Db6 = Porta.6 , Db7 = Porta.7
Config Lcd = 20 * 4
Cursor Off

Config Sda = Portd.1
Config Scl = Portd.0
Const Ds1307w = &HD0                    ' Addresses of Ds1307 clock
Const Ds1307r = &HD1

Config Clock = User
Config Date = Dmy , Separator = -
'Date$ = "14-05-14"
'Time$ = "22:28:11"

Portf = 255
Ddrf = &HF0
Portc = 255
Ddrc = 255
Portd.2 = 1
Portd.3 = 1
Ddrd.2 = 1
Ddrd.3 = 1
Portd.7 = 1
Ddrd.7 = 1
Portd.7 = 0

Buzz Alias Portd.7
Dat Alias Portc
Sbl Alias Portd.3
Sbh Alias Portd.2
```

```
Sbh = 0
Sbl = 1

Dim Keypad As Byte
Dim Ff As Byte , B As Byte
Dim I As Byte , N As Eram Byte , M As Byte
Dim Nilai(12) As Byte , Ni(4) As Byte
Dim Jam As Byte , Menit As Byte , Detik As Byte
Dim Tahun As Byte , Bulan As Byte , Tanggal As Byte
Dim Bln_agenda As Byte , Tgl_agenda As Byte
Dim Temp As Long , Temp1 As Long , Temp2 As Long
Dim Now As Long
Dim Status As Byte
Dim Hr As Byte
Dim Str_tanggal As String * 5
Dim Str_bulan As String * 5
Dim Str_tahun As String * 5
Dim Str_no_file As String * 5
Dim No_file As Byte
Dim Char_tanggal As String * 26
Dim Eep_no_file As Eram Byte
Dim Status_mmc As Bit
Dim Dt_tgl As Long , Tgl_agendanya(34) As Eram Long
Dim Keterangan As String * 12 , Ket(34) As Eram String * 12
Dim X(12) As String * 1
Dim Mati As Bit , Now_ref As Eram Byte , Nr As Byte
Dim O As Byte
Dim _1 As Byte , _2 As Byte , _3 As Byte
Dim Bb As Byte

Dim Pasar_ep(5) As String * 6 , Psr As String * 6
Dim Hari_ep(7)as String * 6 , Hri As String * 6
Dim Count_psr As Eram Byte , Count_hr As Eram Byte
Dim Count1 As Byte , Count2 As Byte

Pasar_ep(1) = "PON   "
Pasar_ep(2) = "WAGE  "
Pasar_ep(3) = "KLIWON"
Pasar_ep(4) = "LEGI  "
Pasar_ep(5) = "PAHING"

Hari_ep(1) = "SENIN "
Hari_ep(2) = "SELASA"
Hari_ep(3) = "RABU  "
Hari_ep(4) = "KAMIS "
Hari_ep(5) = "JUM'AT"
```

```
Hari_ep(6) = "SABTU "
Hari_ep(7) = "MINGGU"

Mati = 0
Mulai:
Hr = _day
Reset Buzz
Status = 0

Waitms 500
Cls

Gosub Conv
Gosub Kirim_tanggal

Do
  Gosub Baca_keypad
  Gosub Getdatetime
  Gosub Cek
  Locate 1 , 1
  Lcd "Hari: " ; Hri ; "/ " ; Psr
  Locate 2 , 1
  Lcd "Tgl : " ; Date$
  Locate 3 , 1
  Lcd "Jam : " ; Time$
  Locate 4 , 1
  Lcd "          Menu -> D"
  If Keypad = 15 Then Gosub Menu
Loop

'=============================================================
====================
Kirim_tanggal:
  Temp = Now / 10000
  _3 = Temp - 2000
  Temp = Temp * 10000
  Temp1 = Now - Temp
  Temp2 = Temp1 / 100
  _2 = Temp2
  Temp = _2
  Temp = Temp * 100
  _1 = Temp1 - Temp

  Sbh = 0
  Sbl = 1
  Dat = _2
```

```
  Waitms 100

  Sbh = 1
  Sbl = 0
  Dat = _1
  Waitms 100

  Sbh = 1
  Sbl = 1
  Dat = _3
  Waitms 100

  Sbh = 0
  Sbl = 0
Return
'=============================================================
====================
Conv:
  Gosub Getdatetime
  Gosub Setdate
  _day = Makedec(_day)
  _month = Makedec(_month)
  _year = Makedec(_year)
  Tanggal = _day
  Bulan = _month
  Tahun = _year
  Now = 2000 + Tahun
  Now = Now * 100
  Now = Now + Bulan
  Now = Now * 100
  Now = Now + Tanggal
  Nr = Now_ref

  Count1 = Count_hr
  If Count1 < 1 Then Count1 = 1
  If Count1 > 7 Then Count1 = 7
  Select Case Count1
    Case 1 : Hri = Hari_ep(1)
    Case 2 : Hri = Hari_ep(2)
    Case 3 : Hri = Hari_ep(3)
    Case 4 : Hri = Hari_ep(4)
    Case 5 : Hri = Hari_ep(5)
    Case 6 : Hri = Hari_ep(6)
    Case 7 : Hri = Hari_ep(7)
  End Select
```

```
  Count2 = Count_psr
  If Count2 < 1 Then Count2 = 1
  If Count2 > 5 Then Count2 = 5
  Select Case Count2
    Case 1 : Psr = Pasar_ep(1)
    Case 2 : Psr = Pasar_ep(2)
    Case 3 : Psr = Pasar_ep(3)
    Case 4 : Psr = Pasar_ep(4)
    Case 5 : Psr = Pasar_ep(5)
  End Select
Return
'============================================================
====================
Cek:
  Gosub Conv
  If Nr <> Tanggal Then
    Mati = 0
    Now_ref = Tanggal
    Gosub Kirim_tanggal
    Incr Count1
    Incr Count2
    If Count1 > 7 Then Count1 = 1
    If Count2 > 5 Then Count2 = 1
    Count_hr = Count1
    Count_psr = Count2
  End If
  For I = 1 To 34
    Dt_tgl = Tgl_agendanya(i)
    Keterangan = "           "
    Keterangan = Ket(i)
    If Now = Dt_tgl And Mati = 0 Then Gosub Buzzer
  Next
Return
'============================================================
====================
Buzzer:
  Cls
  Locate 1 , 1
  Lcd "Ada Hari Besar!!!"
  Locate 2 , 1
  Lcd "Tgl : " ; Date$
  Locate 3 , 1
  Lcd "Ket : " ; Keterangan
  Locate 4 , 1
  Lcd "Tekan D --> off"
```

```
  Do
    Gosub Baca_keypad
    Toggle Buzz
    Waitms 100
  Loop Until Keypad = 15
  Mati = 1

  Gosub Tulis_mmc
Gosub Mulai
'=============================================================
====================
Tulis_mmc:
  Cls
  Locate 1 , 1
  Lcd "Tulis ke MMC"
  Locate 2 , 1
  Lcd "Initialization..."

  $include "Config_AVR-DOS.BAS"
  $include "Config_MMC.bas"
  Ff = Drivereset()
  Ff = Driveinit()
'   $external Waitms

  If Gbdriveerror <> 0 Then
    Cls
    Lcd "Error Config..."
    Locate 2 , 1
    Lcd Gbdriveerror
    Wait 2
    Goto Mulai
  End If

  B = Initfilesystem(1)                    'init file system
  If B <> 0 Then Lcd "Error MMC"
  Ff = Freefile()

  Wait 1
  Cls
  Lcd "Tulis file baru..."
  Locate 2 , 1
  Lcd "Please wait...."
  Wait 1
  Incr No_file
  Gosub Getdatetime
  Str_tanggal = Str(_day)
```

```
  Str_bulan = Str(_month)
  Str_tahun = Str(_year)
  Str_no_file = Str(no_file)
  Str_no_file = Format(str_no_file , "00")
  Char_tanggal = Str_tanggal + Str_bulan + Str_tahun + "-" + Str_no_file + ".txt"

  Status_mmc = 1
  If No_file > 99 Then
   No_file = 0
  End If
  Eep_no_file = No_file

  Open Char_tanggal For Output As #ff
  Print #ff , "+-------------------------+"
  Print #ff , "| Tanggal |    Agenda    |"
  Print #ff , "+-------------------------+"
  Print #ff , "| " ; Date$ ; " | " ; Keterangan ; " |"
  Print #ff , "+-------------------------+"
  Close #ff
  Status_mmc = 0
  Cls
  Lcd "Tulis file selesai.."
  Wait 1
Return
'============================================================
====================
Menu:
  Status = 1
  Waitms 500
  Do
    Gosub Baca_keypad
  Loop Until Keypad = 16
  Waitms 100
  Cls
  Locate 1 , 1
  Lcd "1. Atur Tanggal"
  Locate 2 , 1
  Lcd "2. Atur Waktu"
  Locate 3 , 1
  Lcd "3. Tambah Hari Besar"
  Locate 4 , 1
  Lcd "4. Hapus Hari Besar"
  Do
    Gosub Baca_keypad
    Select Case Keypad
      Case 1 : Gosub Atur_tanggal
```

```
        Case 2 : Gosub Atur_waktu
        Case 3 : Gosub Set_agenda
        Case 4 : Gosub Hapus_agenda
        Case 11 : Gosub Lanjut
        Case 12 : Gosub Mulai
        Case 15 : Gosub Menu
      End Select
    Loop
    Waitms 200
Return
'============================================================
====================
Lanjut:
    Waitms 200
    Cls
    Locate 1 , 1
    Lcd "5. Lihat Hari Besar"
    Locate 2 , 1
    Lcd "6. Setting STD"
    Locate 3 , 1
    Lcd "7. Atur Hari/Pasaran"
    Locate 4 , 1
    Lcd "8. --"
    Do
      Gosub Baca_keypad
      Select Case Keypad
        Case 5 : Gosub Lihat_agenda
        Case 6 : Gosub Set_std
        Case 7 : Gosub Set_psr
        Case 10 : Gosub Menu
        Case 12 : Gosub Mulai
      End Select
    Loop
Return
'============================================================
====================
Set_psr:
    Waitms 500
    Cls
    Locate 1 , 1
    Lcd "Pilih Hari :"
    Waitms 500
Xx:
    Cls
    Locate 1 , 1
    Lcd "1. Senin   5. Jum'at"
```

```
  Locate 2 , 1
  Lcd "2. Selasa  6. Sabtu"
  Locate 3 , 1
  Lcd "3. Rabu    7. Minggu"
  Locate 4 , 1
  Lcd "4. Kamis"
  Do
    Gosub Baca_keypad
  Loop Until Keypad <> 16
  Waitms 200
  If Keypad > 7 Then Gosub Xx
  Count_hr = Keypad

  Cls
  Locate 1 , 1
  Lcd "Pilih Pasaran :"
  Waitms 500
Yy:
  Cls
  Locate 1 , 1
  Lcd "1. Pon     4. Legi"
  Locate 2 , 1
  Lcd "2. Wage    5. Pahing"
  Locate 3 , 1
  Lcd "3. Kliwon"
  Do
    Gosub Baca_keypad
  Loop Until Keypad <> 16
  Waitms 200
  If Keypad > 5 Then Gosub Yy
  Count_psr = Keypad

  Cls
  Lcd "Selesai.."
  Waitms 500
Gosub Lanjut
'=============================================================
====================
Set_std:
  Waitms 500
  Tahun = _year

  Cls
  Lcd "Setting hari besar"
  Waitms 500
_3:
```

```
    Status = 3
    Cls
    Lcd "Tanggal Maulid Nabi"
    Waitms 500
    Gosub Atur_tanggal
_4:
    Status = 4
    Cls
    Lcd "Tahun Baru Imlek"
    Waitms 500
    Gosub Atur_tanggal
_5:
    Status = 5
    Cls
    Lcd "Hari Raya Nyepi"
    Waitms 500
    Gosub Atur_tanggal
_6:
    Status = 6
    Cls
    Lcd "Wafat Isa Almasih"
    Waitms 500
    Gosub Atur_tanggal
_7:
    Status = 7
    Cls
    Lcd "Hari Raya Waisak"
    Waitms 500
    Gosub Atur_tanggal
_8:
    Status = 8
    Cls
    Lcd "Isra' Mi'raj"
    Waitms 500
    Gosub Atur_tanggal
_9:
    Status = 9
    Cls
    Lcd "Kenaikan Isa"
    Waitms 500
    Gosub Atur_tanggal
_10:
    Status = 10
    Cls
    Lcd "Idul Fitri"
    Waitms 500
```

```
    Gosub Atur_tanggal
_11:
    Status = 11
    Cls
    Lcd "Idul Adha"
    Waitms 500
    Gosub Atur_tanggal
_12:
    Status = 12
    Cls
    Lcd "Tahun Baru Hijriyah"
    Waitms 500
    Gosub Atur_tanggal
_13:
    Cls
    Lcd "Selesai"
    Waitms 500
Return
'=============================================================
====================
Atur_tanggal:
  Waitms 500
  Cls
  Locate 1 , 1
  Lcd "Atur tanggal :"
  Locate 2 , 1
  Lcd Date$ ; "  "
  Do
  Atur_tgl:
    For I = 1 To 2
      Do
        Gosub Baca_keypad
        Select Case Keypad
          Case 0 : Nilai(i) = 0
          Case 1 : Nilai(i) = 1
          Case 2 : Nilai(i) = 2
          Case 3 : Nilai(i) = 3
          Case 4 : Nilai(i) = 4
          Case 5 : Nilai(i) = 5
          Case 6 : Nilai(i) = 6
          Case 7 : Nilai(i) = 7
          Case 8 : Nilai(i) = 8
          Case 9 : Nilai(i) = 9
          'Case 10 : Gosub Atur_tahun
          'Case 11 : Gosub Atur_bulan
          Case 12 : Gosub Menu
```

```
        Case 14 : Gosub Mulai
        Case 15 : Gosub Simpan_tanggal
      End Select
      Locate 2 , 1
      Lcd "  "
      Waitms 100
      Locate 2 , 1
      Lcd Nilai(1) ; Nilai(2)
      Waitms 100
    Loop Until Keypad <> 16
    Waitms 200
  Next
Atur_bulan:
  For I = 3 To 4
    Do
      Gosub Baca_keypad
      Select Case Keypad
        Case 0 : Nilai(i) = 0
        Case 1 : Nilai(i) = 1
        Case 2 : Nilai(i) = 2
        Case 3 : Nilai(i) = 3
        Case 4 : Nilai(i) = 4
        Case 5 : Nilai(i) = 5
        Case 6 : Nilai(i) = 6
        Case 7 : Nilai(i) = 7
        Case 8 : Nilai(i) = 8
        Case 9 : Nilai(i) = 9
        'Case 10 : Gosub Atur_tgl
        'Case 11 : Gosub Atur_tahun
        Case 12 : Gosub Menu
        Case 14 : Gosub Mulai
        Case 15 : Gosub Simpan_tanggal
      End Select
      Locate 2 , 4
      Lcd "  "
      Waitms 100
      Locate 2 , 4
      Lcd Nilai(3) ; Nilai(4)
      Waitms 100
    Loop Until Keypad <> 16
    Waitms 200
  Next
Atur_tahun:
  For I = 5 To 6
    Do
      Gosub Baca_keypad
```

```
          Select Case Keypad
            Case 0 : Nilai(i) = 0
            Case 1 : Nilai(i) = 1
            Case 2 : Nilai(i) = 2
            Case 3 : Nilai(i) = 3
            Case 4 : Nilai(i) = 4
            Case 5 : Nilai(i) = 5
            Case 6 : Nilai(i) = 6
            Case 7 : Nilai(i) = 7
            Case 8 : Nilai(i) = 8
            Case 9 : Nilai(i) = 9
            'Case 10 : Gosub Atur_bulan
            'Case 11 : Gosub Atur_tgl
            Case 12 : Gosub Menu
            Case 14 : Gosub Mulai
            Case 15 : Gosub Simpan_tanggal
          End Select
          Locate 2 , 7
          Lcd "  "
          Waitms 100
          Locate 2 , 7
          Lcd Nilai(5) ; Nilai(6)
          Waitms 100
        Loop Until Keypad <> 16
        Waitms 200
      Next
    Loop
Return
'=============================================================
====================
Atur_waktu:
  Waitms 500
  Cls
  Locate 1 , 1
  Lcd "Atur waktu :"
  Locate 2 , 1
  Lcd Time$ ; "  "
  Do
  Atur_jam:
    For I = 7 To 8
      Do
        Gosub Baca_keypad
        Select Case Keypad
          Case 0 : Nilai(i) = 0
          Case 1 : Nilai(i) = 1
          Case 2 : Nilai(i) = 2
```

```
            Case 3 : Nilai(i) = 3
            Case 4 : Nilai(i) = 4
            Case 5 : Nilai(i) = 5
            Case 6 : Nilai(i) = 6
            Case 7 : Nilai(i) = 7
            Case 8 : Nilai(i) = 8
            Case 9 : Nilai(i) = 9
            'Case 10 : Gosub Atur_detik
            'Case 11 : Gosub Atur_menit
            Case 12 : Gosub Menu
            Case 14 : Gosub Mulai
            Case 15 : Gosub Simpan_waktu
          End Select
          Locate 2 , 1
          Lcd "  "
          Waitms 100
          Locate 2 , 1
          Lcd Nilai(7) ; Nilai(8)
          Waitms 100
        Loop Until Keypad <> 16
        Waitms 200
    Next
Atur_menit:
    For I = 9 To 10
      Do
        Gosub Baca_keypad
        Select Case Keypad
          Case 0 : Nilai(i) = 0
          Case 1 : Nilai(i) = 1
          Case 2 : Nilai(i) = 2
          Case 3 : Nilai(i) = 3
          Case 4 : Nilai(i) = 4
          Case 5 : Nilai(i) = 5
          Case 6 : Nilai(i) = 6
          Case 7 : Nilai(i) = 7
          Case 8 : Nilai(i) = 8
          Case 9 : Nilai(i) = 9
          'Case 10 : Gosub Atur_jam
          'Case 11 : Gosub Atur_detik
          Case 12 : Gosub Menu
          Case 14 : Gosub Mulai
          Case 15 : Gosub Simpan_waktu
        End Select
        Locate 2 , 4
        Lcd "  "
        Waitms 100
```

```
          Locate 2 , 4
          Lcd Nilai(9) ; Nilai(10)
          Waitms 100
        Loop Until Keypad <> 16
        Waitms 200
      Next
  Atur_detik:
    For I = 11 To 12
      Do
        Gosub Baca_keypad
        Select Case Keypad
          Case 0 : Nilai(i) = 0
          Case 1 : Nilai(i) = 1
          Case 2 : Nilai(i) = 2
          Case 3 : Nilai(i) = 3
          Case 4 : Nilai(i) = 4
          Case 5 : Nilai(i) = 5
          Case 6 : Nilai(i) = 6
          Case 7 : Nilai(i) = 7
          Case 8 : Nilai(i) = 8
          Case 9 : Nilai(i) = 9
          'Case 10 : Gosub Atur_menit
          'Case 11 : Gosub Atur_jam
          Case 12 : Gosub Menu
          Case 14 : Gosub Mulai
          Case 15 : Gosub Simpan_waktu
        End Select
        Locate 2 , 7
        Lcd "  "
        Waitms 100
        Locate 2 , 7
        Lcd Nilai(11) ; Nilai(12)
        Waitms 100
      Loop Until Keypad <> 16
      Waitms 200
    Next
  Loop
Return
'============================================================
====================
Simpan_waktu:
  Waitms 500
  Jam = Nilai(7) * 10
  Jam = Jam + Nilai(8)
  Menit = Nilai(9) * 10
  Menit = Menit + Nilai(10)
```

```
  Detik = Nilai(11) * 10
  Detik = Detik + Nilai(12)

  _hour = Jam
  _min = Menit
  _sec = Detik
  Gosub Settime

  If Jam > 24 Or Menit > 59 Or Detik > 59 Then
    Cls
    Locate 1 , 1
    Lcd "Jam/Menit Salah!!"
    Locate 3 , 1
    Lcd "Ulangi input!!"
    Wait 1
    Gosub Atur_waktu
  End If
Gosub Menu
'============================================================
====================
Simpan_tanggal:
  Waitms 500
  Bulan = Nilai(3) * 10
  Bulan = Bulan + Nilai(4)
  Tanggal = Nilai(1) * 10
  Tanggal = Tanggal + Nilai(2)
  Tahun = Nilai(5) * 10
  Tahun = Tahun + Nilai(6)

  If Status = 1 Then
    _month = Bulan
    _day = Tanggal
    _year = Tahun
    Gosub Setdate
    Gosub Bigdays
    Gosub Conv
    Gosub Kirim_tanggal
  End If
  If Status = 2 Then Gosub Key_huruf
  If Status = 3 Then Gosub Maulid_nabi
  If Status = 4 Then Gosub Imlek
  If Status = 5 Then Gosub Nyepi
  If Status = 6 Then Gosub Wafat_isa
  If Status = 7 Then Gosub Waisak
  If Status = 8 Then Gosub Isra
  If Status = 9 Then Gosub Kenaikan_isa
```

```
   If Status = 10 Then Gosub Idul_fitri
   If Status = 11 Then Gosub Idul_adha
   If Status = 12 Then Gosub Thn_islam

   If Bulan > 12 Or Bulan < 1 Or Tanggal > 31 Or Tanggal < 1 Then
      Cls
      Locate 1 , 1
      Lcd "Bulan/hari Salah!!"
      Locate 3 , 1
      Lcd "Ulangi input!!"
      Wait 1
      Gosub Atur_tanggal
   End If
Gosub Menu
'============================================================
====================
Maulid_nabi:
   Gosub Konversi
   Keterangan = "Maulid Nabi "
   Tgl_agendanya(21) = Dt_tgl
   Ket(21) = Keterangan
Gosub _4

Imlek:
   Gosub Konversi
   Keterangan = "Thn Br Imlek"
   Tgl_agendanya(22) = Dt_tgl
   Ket(22) = Keterangan
Gosub _5

Nyepi:
   Gosub Konversi
   Keterangan = "Hari Nyepi  "
   Tgl_agendanya(23) = Dt_tgl
   Ket(23) = Keterangan
Gosub _6

Wafat_isa:
   Gosub Konversi
   Keterangan = "Wafat Isa   "
   Tgl_agendanya(24) = Dt_tgl
   Ket(24) = Keterangan
Gosub _7

Waisak:
   Gosub Konversi
```

```
   Keterangan = "Hari Waisak "
   Tgl_agendanya(25) = Dt_tgl
   Ket(25) = Keterangan
Gosub _8

Isra:
   Gosub Konversi
   Keterangan = "Isra' Mi'raj"
   Tgl_agendanya(26) = Dt_tgl
   Ket(26) = Keterangan
Gosub _9

Kenaikan_isa:
   Gosub Konversi
   Keterangan = "Kenaikan Isa"
   Tgl_agendanya(27) = Dt_tgl
   Ket(27) = Keterangan
Gosub _10

Idul_fitri:
   Gosub Konversi
   Keterangan = "Idul Fitri  "
   Tgl_agendanya(28) = Dt_tgl
   Ket(28) = Keterangan
Gosub _11

Idul_adha:
   Gosub Konversi
   Keterangan = "Idul Adha   "
   Tgl_agendanya(29) = Dt_tgl
   Ket(29) = Keterangan
Gosub _12

Thn_islam:
   Gosub Konversi
   Keterangan = "Thn Br Islam"
   Tgl_agendanya(30) = Dt_tgl
   Ket(30) = Keterangan
Gosub _13
'============================================================
====================
Bigdays:
   Gosub Getdatetime
   Gosub Setdate
   _year = Makedec(_year)
   Tahun = _year
```

```
   Dt_tgl = 2000 + Tahun
   Dt_tgl = Dt_tgl * 10000
   Dt_tgl = Dt_tgl + 100
   Dt_tgl = Dt_tgl + 1
   Keterangan = "Tahun Baru  "
   Tgl_agendanya(31) = Dt_tgl
   Ket(31) = Keterangan

   Dt_tgl = 2000 + Tahun
   Dt_tgl = Dt_tgl * 10000
   Dt_tgl = Dt_tgl + 500
   Dt_tgl = Dt_tgl + 1
   Keterangan = "Hari Buruh  "
   Tgl_agendanya(32) = Dt_tgl
   Ket(32) = Keterangan

   Dt_tgl = 2000 + Tahun
   Dt_tgl = Dt_tgl * 10000
   Dt_tgl = Dt_tgl + 800
   Dt_tgl = Dt_tgl + 17
   Keterangan = "HUT RI      "
   Tgl_agendanya(33) = Dt_tgl
   Ket(33) = Keterangan

   Dt_tgl = 2000 + Tahun
   Dt_tgl = Dt_tgl * 10000
   Dt_tgl = Dt_tgl + 1200
   Dt_tgl = Dt_tgl + 25
   Keterangan = "Hari Natal  "
   Tgl_agendanya(34) = Dt_tgl
   Ket(34) = Keterangan

Return
'=========================================================
====================
Set_agenda:
   M = N
   M = M + 1
   Waitms 500
   Status = 2
   Cls
   Lcd "   -Agenda Baru!-"
   Locate 2 , 1
   Lcd "No : " ; M
   Wait 1
```

```
   Gosub Atur_tanggal
Gosub Menu
'============================================================
====================
Lihat_agenda:
   Waitms 500
   I = 1
   Cls
   Do
      Gosub Baca_keypad
      Dt_tgl = Tgl_agendanya(i)
      Keterangan = ""
      Keterangan = Ket(i)
      Locate 1 , 1
      Lcd "No  : " ; I
      Locate 2 , 1
      Lcd "Tgl : " ; Dt_tgl
      Locate 3 , 1
      Lcd "Ket : " ; Keterangan
      If Keypad = 10 Then
         Waitms 200
         Cls
         Decr I
      Elseif Keypad = 11 Then
         Waitms 200
         Cls
         Incr I
      End If
      If I < 1 Then I = 34
      If I > 34 Then I = 1
   Loop Until Keypad = 15
   Waitms 200
Gosub Menu
'============================================================
====================
Key_huruf:
   For I = 1 To 12
      X(i) = " "
   Next
   I = 1
'   M = N
   O = 1
   Cls
   Locate 1 , 1
   Lcd "Keterangan :"
   Do
```

```
Gosub Baca_keypad
Select Case O
  Case 1 : X(i) = "A"
  Case 2 : X(i) = "B"
  Case 3 : X(i) = "C"
  Case 4 : X(i) = "D"
  Case 5 : X(i) = "E"
  Case 6 : X(i) = "F"
  Case 7 : X(i) = "G"
  Case 8 : X(i) = "H"
  Case 9 : X(i) = "I"
  Case 10 : X(i) = "J"
  Case 11 : X(i) = "K"
  Case 12 : X(i) = "L"
  Case 13 : X(i) = "M"
  Case 14 : X(i) = "N"
  Case 15 : X(i) = "O"
  Case 16 : X(i) = "P"
  Case 17 : X(i) = "Q"
  Case 18 : X(i) = "R"
  Case 19 : X(i) = "S"
  Case 20 : X(i) = "T"
  Case 21 : X(i) = "U"
  Case 22 : X(i) = "V"
  Case 23 : X(i) = "W"
  Case 24 : X(i) = "X"
  Case 25 : X(i) = "Y"
  Case 26 : X(i) = "Z"
  Case 27 : X(i) = " "
  Case 28 : X(i) = "1"
  Case 29 : X(i) = "2"
  Case 30 : X(i) = "3"
  Case 32 : X(i) = "4"
  Case 33 : X(i) = "5"
  Case 34 : X(i) = "6"
  Case 35 : X(i) = "7"
  Case 36 : X(i) = "8"
  Case 37 : X(i) = "9"
  Case 38 : X(i) = "0"
End Select
Locate 2 , I
Lcd X(i)
Waitms 100

If Keypad = 10 Then
  Decr I
```

```
      Waitms 100
    Elseif Keypad = 11 Then
      Incr I
      Waitms 100
    Elseif Keypad = 13 Then
      Incr O
      Waitms 100
    Elseif Keypad = 14 Then
      Decr O
      Waitms 100
    End If

    If I < 1 Then I = 1
    If I > 12 Then I = 12
    If O > 38 Then O = 1
    If O < 1 Then O = 38

  Loop Until Keypad = 15
  Waitms 200
  Keterangan = X(1) + X(2) + X(3) + X(4) + X(5) + X(6) + X(7) + X(8) + X(9) +
X(10) + X(11) + X(12)
Gosub Simpan_eeprom
Waitms 200
Gosub Menu
'============================================================
====================
Konversi:
  If Bulan > 12 Or Bulan < 1 Or Tanggal > 31 Or Tanggal < 1 Then
    Cls
    Locate 1 , 1
    Lcd "Bulan/hari Salah!!"
    Locate 3 , 1
    Lcd "Ulangi input!!"
    Wait 1
    Gosub Atur_tanggal
  End If

  Dt_tgl = 2000 + Tahun
  Dt_tgl = Dt_tgl * 100
  Dt_tgl = Dt_tgl + Bulan
  Dt_tgl = Dt_tgl * 100
  Dt_tgl = Dt_tgl + Tanggal
Return
'============================================================
====================
Simpan_eeprom:
```

```
'   M = N
'   Incr M

   If M > 20 Then
      Cls
      Lcd "Kapasitas habis"
      Wait 1
      Gosub Menu
   End If

   Gosub Konversi

   If M < 21 Then
      Tgl_agendanya(m) = Dt_tgl
      Ket(m) = Keterangan
      N = M
      Cls
      Locate 1 , 1
      Lcd "Agenda Tersimpan.."
      Locate 2 , 1
      Lcd "No  : " ; M
      Locate 3 , 1
      Lcd "Tgl : " ; Tanggal ; "-" ; Bulan ; "-" ; Tahun
      Locate 4 , 1
      Lcd "Ket : " ; Keterangan
      Waitms 200
      Do
        Gosub Baca_keypad
      Loop Until Keypad = 15
   End If
Return
'============================================================
====================
Hapus_agenda:
   Waitms 500
   Cls
   Lcd "    Anda Yakin??"
   Locate 4 , 1
   Lcd "Yes->D     Cancel->A"
   Do
      Gosub Baca_keypad
      If Keypad = 12 Then Gosub Menu
   Loop Until Keypad = 15
   Keterangan = "            "
   For I = 1 To 20
      Tgl_agendanya(i) = 0
```

```
    Ket(i) = Keterangan
  Next
  M = 0
  N = M
  Cls
  Lcd "Agenda terhapus....."
  Wait 1
  Gosub Bigdays
Gosub Menu
'=============================================================
====================
Baca_keypad:
  Keypad = 16
  Portf.4 = 0
  Portf.5 = 1
  Portf.6 = 1
  Portf.7 = 1
  If Pinf.0 = 0 Then Keypad = 1
  If Pinf.1 = 0 Then Keypad = 4
  If Pinf.2 = 0 Then Keypad = 7
  If Pinf.3 = 0 Then Keypad = 10
  Portf.4 = 1
  Portf.5 = 0
  Portf.6 = 1
  Portf.7 = 1
  If Pinf.0 = 0 Then Keypad = 2
  If Pinf.1 = 0 Then Keypad = 5
  If Pinf.2 = 0 Then Keypad = 8
  If Pinf.3 = 0 Then Keypad = 0
  Portf.4 = 1
  Portf.5 = 1
  Portf.6 = 0
  Portf.7 = 1
  If Pinf.0 = 0 Then Keypad = 3
  If Pinf.1 = 0 Then Keypad = 6
  If Pinf.2 = 0 Then Keypad = 9
  If Pinf.3 = 0 Then Keypad = 11
  Portf.4 = 1
  Portf.5 = 1
  Portf.6 = 1
  Portf.7 = 0
  If Pinf.0 = 0 Then Keypad = 12
  If Pinf.1 = 0 Then Keypad = 13
  If Pinf.2 = 0 Then Keypad = 14
  If Pinf.3 = 0 Then Keypad = 15
Return
```

```
'=========================================================
====================
Getdatetime:

  I2cstart
  I2cwbyte Ds1307w
  I2cwbyte 0

  I2cstart
  I2cwbyte Ds1307r
  I2crbyte _sec , Ack
  I2crbyte _min , Ack
  I2crbyte _hour , Nack
  I2cstop

  I2cstart
  I2cwbyte Ds1307w
  I2cwbyte 4

  I2cstart
  I2cwbyte Ds1307r
  I2crbyte _day , Ack
  I2crbyte _month , Ack
  I2crbyte _year , Nack
  I2cstop

  _sec = Makedec(_sec)
  _min = Makedec(_min)
  _hour = Makedec(_hour)
  _day = Makedec(_day)
  _month = Makedec(_month)
  _year = Makedec(_year)

Return
'=========================================================
====================
Setdate:
  _day = Makebcd(_day)
  _month = Makebcd(_month)
  _year = Makebcd(_year)

  I2cstart
  I2cwbyte Ds1307w
  I2cwbyte 4

  I2cwbyte _day
```

```
  I2cwbyte _month
  I2cwbyte _year
  I2cstop
Return
'=============================================================
====================
Settime:
  _sec = Makebcd(_sec)
  _min = Makebcd(_min)
  _hour = Makebcd(_hour)

  I2cstart
  I2cwbyte Ds1307w
  I2cwbyte 0

  I2cwbyte _sec
  I2cwbyte _min
  I2cwbyte _hour
  I2cstop
Return
'=============================================================
====================
```

Lampiran 6. Program BASCOM AVR pada Mikrokontroler ATmega32

```
$regfile = "m32def.dat"
$crystal = 11059200


Porta = 255
Ddra = 255
Portb = 255
Ddrb = 0
Portc = 255
Ddrc = 255
Portd = 255
Ddrd = 0

Dim _1 As Byte , _2 As Byte , _3 As Byte , _4 As Byte
Dim _5 As Byte , _6 As Byte , _7 As Byte , _8 As Byte
Dim D As Byte , M As Byte , Y As Byte
Dim Tmp As Byte , Tmp1 As Byte

Datasegment Alias Portc
Dat Alias Pinb
Status Alias Pind

D = 0
M = 0
Y = 0

Gosub Konversi

Do

  If Status <> 252 Then
    If Status = 253 Then M = Dat
    If Status = 254 Then D = Dat
    If Status = 255 Then Y = Dat
    If D > 31 Then D = 31
    If M > 12 Then M = 12
    If M < 1 Then M = 1
    If D < 1 Then D = 1
    Gosub Konversi
  Else
    Porta = 1
    Datasegment = Lookup(_8 , Segment)
    Waitms 2
    Porta = 2
```

```
    Datasegment = Lookup(_7 , Segment)
    Waitms 2
    Porta = 4
    Datasegment = Lookup(_6 , Segment)
    Waitms 2
    Porta = 8
    Datasegment = Lookup(_5 , Segment)
    Waitms 2
    Porta = 16
    Datasegment = Lookup(_4 , Segment)
    Waitms 2
    Porta = 32
    Datasegment = Lookup(_3 , Segment)
    Waitms 2
    Porta = 64
    Datasegment = Lookup(_2 , Segment)
    Waitms 2
    Porta = 128
    Datasegment = Lookup(_1 , Segment)
    Waitms 2
  End If

Loop

Konversi:
  If D < 10 Then
    _1 = 0
    _2 = D
  Else
    Tmp = D / 10
    _1 = Tmp
    Tmp = Tmp * 10
    _2 = D - Tmp
  End If

  If M < 10 Then
    _3 = 0
    _4 = M
  Else
    Tmp = M / 10
    _3 = Tmp
    Tmp = Tmp * 10
    _4 = M - Tmp
  End If

  If Y > 9 And Y < 100 Then
```

```
    Tmp = Y / 10
    _5 = 2
    _6 = 0
    _7 = Tmp
    Tmp = Tmp * 10
    _8 = Y - Tmp
  Elseif Y < 10 Then
    _5 = 2
    _6 = 0
    _7 = 0
    _8 = Y
  Else
    _5 = 2
    Tmp = Y / 100
    Tmp1 = Y / 10
    _6 = Tmp
    _7 = Tmp1
    Tmp = Tmp * 100
    Tmp1 = Tmp1 * 10
    Tmp = Tmp + Tmp1
    _8 = Y - Tmp
  End If

  If D = 0 And M = 0 And Y = 0 Then
    _1 = 2
    _2 = 6
    _3 = 0
    _4 = 5
    _5 = 2
    _6 = 0
    _7 = 1
    _8 = 4
  End If
Return

Segment:
'Data 0        1        2        3        4        5        6        7        8
9
Data &B11000000 , &B11111001 , &B10100100 , &B10110000 , &B10011001 ,
&B10010010 , &B10000010 , &B11111000 , &B10000000 , &B10010000
```

## Features

- High-performance, Low-power Atmel®AVR®8-bit Microcontroller
- Advanced RISC Architecture
  - 133 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers + Peripheral Control Registers
  - Fully Static Operation
  - Up to 16MIPS Throughput at 16MHz
  - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory segments
  - 128Kbytes of In-System Self-programmable Flash program memory
  - 4Kbytes EEPROM
  - 4Kbytes Internal SRAM
  - Write/Erase cycles: 10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/100 years at 25°C(1)
  - Optional Boot Code Section with Independent Lock Bits
    - In-System Programming by On-chip Boot Program
    - True Read-While-Write Operation
  - Up to 64Kbytes Optional External Memory Space
  - Programming Lock for Software Security
  - SPI Interface for In-System Programming
- QTouch® library support
  - Capacitive touch buttons, sliders and wheels
  - QTouch and QMatrix acquisition
  - Up to 64 sense channels
- 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 Expanded 16-bit Timer/Counters with Separate Prescaler, Compare Mode and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Two 8-bit PWM Channels
  - 6 PWM Channels with Programmable Resolution from 2 to 16 Bits
  - Output Compare Modulator
  - 8-channel, 10-bit ADC
    - 8 Single-ended Channels
    - 7 Differential Channels
    - 2 Differential Channels with Programmable Gain at 1x, 10x, or 200x
  - Byte-oriented Two-wire Serial Interface
  - Dual Programmable Serial USARTs
  - Master/Slave SPI Serial Interface
  - Programmable Watchdog Timer with 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
  - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby
  - Software Selectable Clock Frequency
  - ATmega103 Compatibility Mode Selected by a Fuse
  - Global Pull-up Disable
- I/O and Packages
  - 53 Programmable I/O Lines
  - 64-lead TQFP and 64-pad QFN/MLF
- Operating Voltages
  - 2.7 - 5.5V ATmega128L
  - 4.5 - 5.5V ATmega128
- Speed Grades
  - 0 - 8MHz ATmega128L
  - 0 - 16MHz ATmega128

# 8-bit Atmel Microcontroller with 128KBytes In-System Programmable Flash

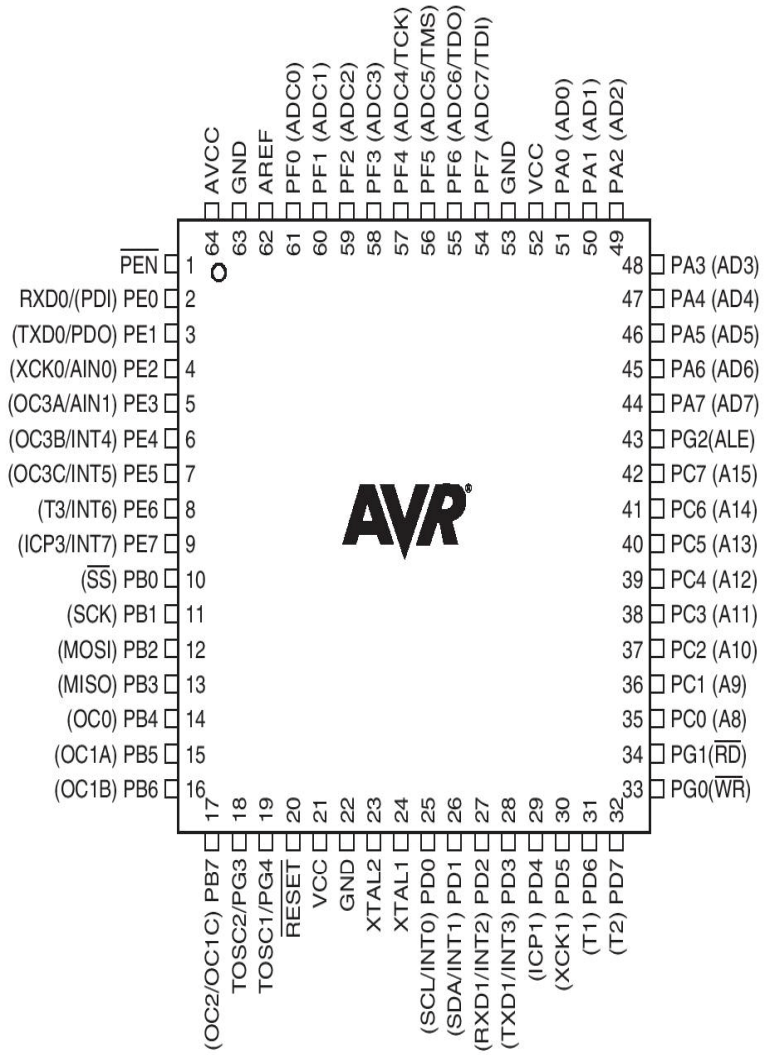
# ATmega128 ATmega128L

## Summary

Rev. 2467XS–AVR–06/11

## Pin Configurations

**Figure 1.** Pinout ATmega128



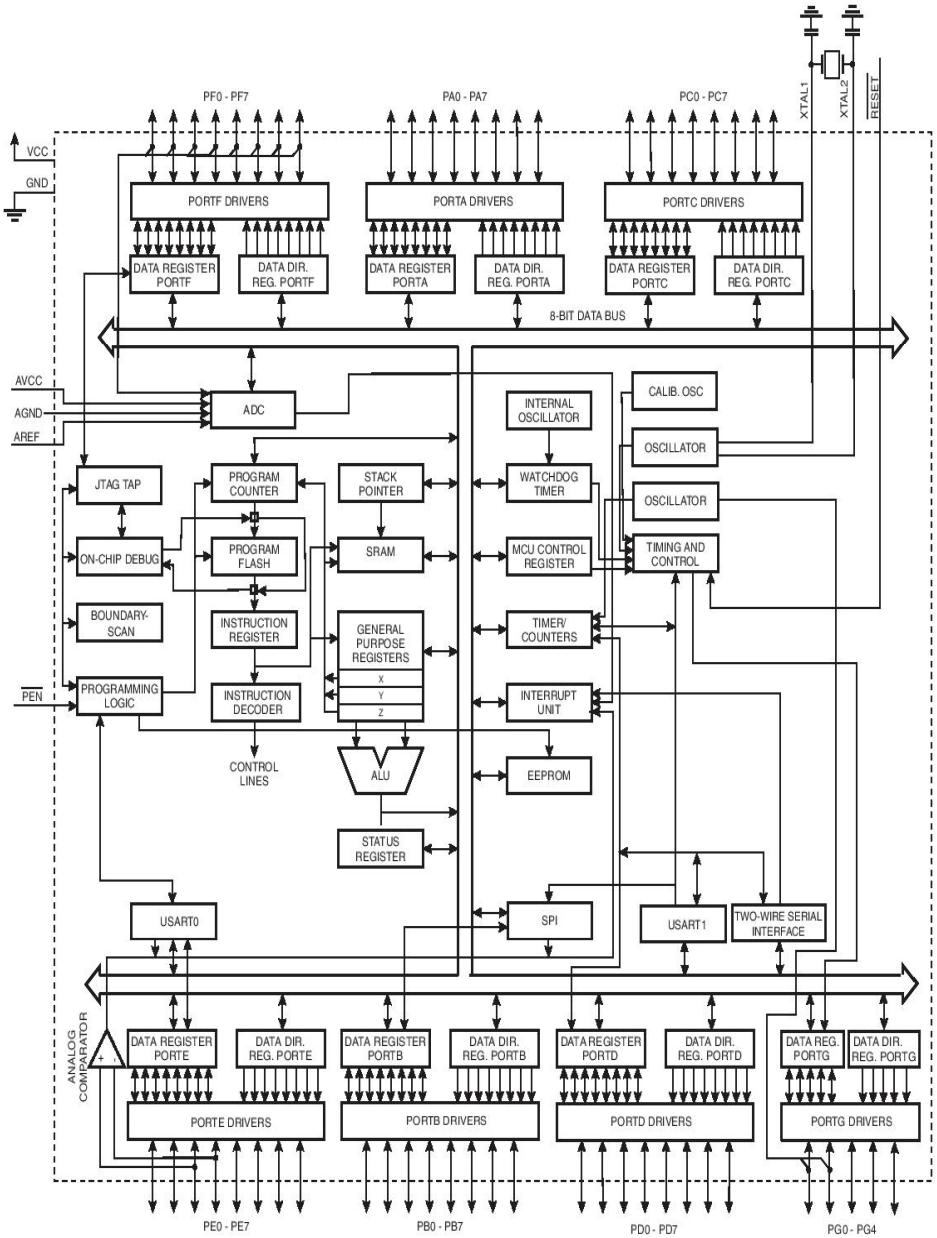

Note: The Pinout figure applies to both TQFP and MLF packages. The bottom pad under the QFN/MLF package should be soldered to ground.

## Overview

The Atmel® AVR® ATmega128 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega128 achieves throughputs approaching 1MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

## Block Diagram

**Figure 2.** Block Diagram



The Atmel® AVR® core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The ATmega128 provides the following features: 128Kbytes of In-System Programmable Flash with Read-While-Write capabilities, 4Kbytes EEPROM, 4Kbytes SRAM, 53 general purpose I/O lines, 32 general purpose working registers, Real Time Counter (RTC), four flexible Timer/Counters with compare modes and PWM, 2 USARTs, a byte oriented Two-wire Serial Interface, an 8-channel, 10-bit ADC with optional differential input stage with programmable gain, programmable Watchdog Timer with Internal Oscillator, an SPI serial port, IEEE std. 1149.1 compliant JTAG test interface, also used for accessing the On-chip Debug system and programming and six software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or Hardware Reset. In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except Asynchronous Timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the Crystal/Resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low power consumption. In Extended Standby mode, both the main Oscillator and the Asynchronous Timer continue to run.

Atmel offers the QTouch® library for embedding capacitive touch buttons, sliders and wheels functionality into AVR microcontrollers. The patented charge-transfer signal acquisition offers robust sensing and includes fully debounced reporting of touch keys and includes Adjacent Key Suppression® (AKS™) technology for unambiguous detection of key events. The easy-to-use QTouch Suite toolchain allows you to explore, develop and debug your own touch applications.

The device is manufactured using Atmel's high-density nonvolatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed in-system through an SPI serial interface, by a conventional nonvolatile memory programmer, or by an On-chip Boot program running on the AVR core. The boot program can use any interface to download the application program in the application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega128 is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega128 device is supported with a full suite of program and system development tools including: C compilers, macro assemblers, program debugger/simulators, in-circuit emulators, and evaluation kits.

## ATmega103 and ATmega128 Compatibility

The ATmega128 is a highly complex microcontroller where the number of I/O locations supersedes the 64 I/O locations reserved in the AVR instruction set. To ensure backward compatibility with the ATmega103, all I/O locations present in ATmega103 have the same location in ATmega128. Most additional I/O locations are added in an Extended I/O space starting from $60 to $FF, (i.e., in the ATmega103 internal RAM space). These locations can be reached by using LD/LDS/LDD and ST/STS/STD instructions only, not by using IN and OUT instructions. The relocation of the internal RAM space may still be a problem for ATmega103 users. Also, the increased number of interrupt vectors might be a problem if the code uses absolute addresses. To solve these problems, an ATmega103 compatibility mode can be selected by programming the fuse M103C. In this mode, none of the functions in the Extended I/O space are in use, so the internal RAM is located as in ATmega103. Also, the Extended Interrupt vectors are removed.

The ATmega128 is 100% pin compatible with ATmega103, and can replace the ATmega103 on current Printed Circuit Boards. The application note “Replacing ATmega103 by ATmega128” describes what the user should be aware of replacing the ATmega103 by an ATmega128.

**ATmega103 Compatibility Mode**

By programming the M103C fuse, the Atmel®ATmega128 will be compatible with the ATmega103 regards to RAM, I/O pins and interrupt vectors as described above. However, some new features in ATmega128 are not available in this compatibility mode, these features are listed below:

- One USART instead of two, Asynchronous mode only. Only the eight least significant bits of the Baud Rate Register is available.
- One 16 bits Timer/Counter with two compare registers instead of two 16-bit Timer/Counters with three compare registers.
- Two-wire serial interface is not supported.
- Port C is output only.
- Port G serves alternate functions only (not a general I/O port).
- Port F serves as digital input only in addition to analog input to the ADC.
- Boot Loader capabilities is not supported.
- It is not possible to adjust the frequency of the internal calibrated RC Oscillator.
- The External Memory Interface can not release any Address pins for general I/O, neither configure different wait-states to different External Memory Address sections.

In addition, there are some other minor differences to make it more compatible to ATmega103:

- Only EXTRF and PORF exists in MCUCSR.
- Timed sequence not required for Watchdog Time-out change.
- External Interrupt pins 3 - 0 serve as level interrupt only.
- USART has no FIFO buffer, so data overrun comes earlier.

Unused I/O bits in ATmega103 should be written to 0 to ensure same operation in ATmega128.

## Pin Descriptions

**VCC**

Digital supply voltage.

**GND**

Ground.

**Port A (PA7..PA0)**

Port A is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port A pins that are externally pulled low will source current if the pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port A also serves the functions of various special features of the ATmega128 as listed on page 72.

**Port B (PB7..PB0)**

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port B also serves the functions of various special features of the ATmega128 as listed on page 73.

**Port C (PC7..PC0)**

Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port C also serves the functions of special features of the Atmel® AVR®ATmega128 as listed on page 76. In ATmega103 compatibility mode, Port C is output only, and the port C pins are not tri-stated when a reset condition becomes active.

Note: The ATmega128 is by default shipped in ATmega103 compatibility mode. Thus, if the parts are not programmed before they are put on the PCB, PORTC will be output during first power up, and until the ATmega103 compatibility mode is disabled.

**Port D (PD7..PD0)**

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port D also serves the functions of various special features of the ATmega128 as listed on page 77.

**Port E (PE7..PE0)**

Port E is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port E output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port E pins that are externally pulled low will source current if the pull-up resistors are activated. The Port E pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port E also serves the functions of various special features of the ATmega128 as listed on page 80.

**Port F (PF7..PF0)**

Port F serves as the analog inputs to the A/D Converter.

Port F also serves as an 8-bit bi-directional I/O port, if the A/D Converter is not used. Port pins can provide internal pull-up resistors (selected for each bit). The Port F output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port F pins that are externally pulled low will source current if the pull-up resistors are activated. The Port F pins are tri-stated when a reset condition becomes active, even if the clock is not running. If the JTAG interface is enabled, the pull-up resistors on pins PF7(TDI), PF5(TMS), and PF4(TCK) will be activated even if a Reset occurs.

The TDO pin is tri-stated unless TAP states that shift out data are entered.

Port F also serves the functions of the JTAG interface.

In ATmega103 compatibility mode, Port F is an input Port only.

**Port G (PG4..PG0)**

Port G is a 5-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port G output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port G pins that are externally pulled low will source current if the pull-up resistors are activated. The Port G pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port G also serves the functions of various special features.

The port G pins are tri-stated when a reset condition becomes active, even if the clock is not running.

In ATmega103 compatibility mode, these pins only serves as strobes signals to the external memory as well as input to the 32kHz Oscillator, and the pins are initialized to PG0 = 1, PG1 = 1, and PG2 = 0 asynchronously when a reset condition becomes active, even if the clock is not running. PG3 and PG4 are oscillator pins.

**$\overline{\text{RESET}}$** Reset input. A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running. The minimum pulse length is given in Table 19 on page 50. Shorter pulses are not guaranteed to generate a reset.

**XTAL1** Input to the inverting Oscillator amplifier and input to the internal clock operating circuit.

**XTAL2** Output from the inverting Oscillator amplifier.

**AVCC** AVCC is the supply voltage pin for Port F and the A/D Converter. It should be externally connected to $V_{CC}$, even if the ADC is not used. If the ADC is used, it should be connected to $V_{CC}$ through a low-pass filter.

**AREF** AREF is the analog reference pin for the A/D Converter.

**PEN** PEN is a programming enable pin for the SPI Serial Programming mode, and is internally pulled high . By holding this pin low during a Power-on Reset, the device will enter the SPI Serial Programming mode. $\overline{\text{PEN}}$ has no function during normal operation.

## Resources

A comprehensive set of development tools, application notes, and datasheets are available for download on http://www.atmel.com/avr.

## Data Retention

Reliability Qualification results show that the projected data retention failure rate is much less than 1 PPM over 20 years at 85°C or 100 years at 25°C

## About Code Examples

This datasheet contains simple code examples that briefly show how to use various parts of the device. These code examples assume that the part specific header file is included before compilation. Be aware that not all C compiler vendors include bit definitions in the header files and interrupt handling in C is compiler dependent. Please confirm with the C compiler documentation for more details.

For I/O registers located in extended I/O map, “IN”, “OUT”, “SBIS”, “SBIC”, “CBI”, and “SBI” instructions must be replaced with instructions that allow access to extended I/O. Typically “LDS” and “STS” combined with “SBRS”, “SBRC”, “SBR”, and “CBR”.

## Capacitive touch sensing

The Atmel QTouch Library provides a simple to use solution to realize touch sensitive interfaces on most Atmel AVR microcontrollers. The QTouch Library includes support for the QTouch and QMatrix acquisition methods.

Touch sensing can be added to any application by linking the appropriate Atmel QTouch Library for the AVR Microcontroller. This is done by using a simple set of APIs to define the touch channels and sensors, and then calling the touch sensing API’s to retrieve the channel information and determine the touch sensor states.

The QTouch Library is FREE and downloadable from the Atmel website at the following location: www.atmel.com/qtouchlibrary. For implementation details and other information, refer to the Atmel QTouch Library User Guide - also available for download from the Atmel website.

## Instruction Set Summary

| Mnemonics | Operands | Description | Operation | Flags | #Clocks |
|---|---|---|---|---|---|
| **ARITHMETIC AND LOGIC INSTRUCTIONS** | | | | | |
| ADD | Rd, Rr | Add two Registers | Rd ← Rd + Rr | Z,C,N,V,H | 1 |
| ADC | Rd, Rr | Add with Carry two Registers | Rd ← Rd + Rr + C | Z,C,N,V,H | 1 |
| ADIW | Rdl,K | Add Immediate to Word | Rdh:Rdl ← Rdh:Rdl + K | Z,C,N,V,S | 2 |
| SUB | Rd, Rr | Subtract two Registers | Rd ← Rd - Rr | Z,C,N,V,H | 1 |
| SUBI | Rd, K | Subtract Constant from Register | Rd ← Rd - K | Z,C,N,V,H | 1 |
| SBC | Rd, Rr | Subtract with Carry two Registers | Rd ← Rd - Rr - C | Z,C,N,V,H | 1 |
| SBCI | Rd, K | Subtract with Carry Constant from Reg. | Rd ← Rd - K - C | Z,C,N,V,H | 1 |
| SBIW | Rdl,K | Subtract Immediate from Word | Rdh:Rdl ← Rdh:Rdl - K | Z,C,N,V,S | 2 |
| AND | Rd, Rr | Logical AND Registers | Rd ← Rd • Rr | Z,N,V | 1 |
| ANDI | Rd, K | Logical AND Register and Constant | Rd ← Rd • K | Z,N,V | 1 |
| OR | Rd, Rr | Logical OR Registers | Rd ← Rd v Rr | Z,N,V | 1 |
| ORI | Rd, K | Logical OR Register and Constant | Rd ← Rd v K | Z,N,V | 1 |
| EOR | Rd, Rr | Exclusive OR Registers | Rd ← Rd ⊕ Rr | Z,N,V | 1 |
| COM | Rd | One's Complement | Rd ← $FF – Rd | Z,C,N,V | 1 |
| NEG | Rd | Two's Complement | Rd ← $00 – Rd | Z,C,N,V,H | 1 |
| SBR | Rd,K | Set Bit(s) in Register | Rd ← Rd v K | Z,N,V | 1 |
| CBR | Rd,K | Clear Bit(s) in Register | Rd ← Rd • ($FF - K) | Z,N,V | 1 |
| INC | Rd | Increment | Rd ← Rd + 1 | Z,N,V | 1 |
| DEC | Rd | Decrement | Rd ← Rd – 1 | Z,N,V | 1 |
| TST | Rd | Test for Zero or Minus | Rd ← Rd • Rd | Z,N,V | 1 |
| CLR | Rd | Clear Register | Rd ← Rd ⊕ Rd | Z,N,V | 1 |
| SER | Rd | Set Register | Rd ← $FF | None | 1 |
| MUL | Rd, Rr | Multiply Unsigned | R1:R0 ← Rd x Rr | Z,C | 2 |
| MULS | Rd, Rr | Multiply Signed | R1:R0 ← Rd x Rr | Z,C | 2 |
| MULSU | Rd, Rr | Multiply Signed with Unsigned | R1:R0 ← Rd x Rr | Z,C | 2 |
| FMUL | Rd, Rr | Fractional Multiply Unsigned | R1:R0 ← (Rd x Rr) << 1 | Z,C | 2 |
| FMULS | Rd, Rr | Fractional Multiply Signed | R1:R0 ← (Rd x Rr) << 1 | Z,C | 2 |
| FMULSU | Rd, Rr | Fractional Multiply Signed with Unsigned | R1:R0 ← (Rd x Rr) << 1 | Z,C | 2 |
| **BRANCH INSTRUCTIONS** | | | | | |
| RJMP | k | Relative Jump | PC ← PC + k + 1 | None | 2 |
| IJMP | | Indirect Jump to (Z) | PC ← Z | None | 2 |
| JMP | k | Direct Jump | PC ← k | None | 3 |
| RCALL | k | Relative Subroutine Call | PC ← PC + k + 1 | None | 3 |
| ICALL | | Indirect Call to (Z) | PC ← Z | None | 3 |
| CALL | k | Direct Subroutine Call | PC ← k | None | 4 |
| RET | | Subroutine Return | PC ← STACK | None | 4 |
| RETI | | Interrupt Return | PC ← STACK | I | 4 |
| CPSE | Rd,Rr | Compare, Skip if Equal | if (Rd = Rr) PC ← PC + 2 or 3 | None | 1 / 2 / 3 |
| CP | Rd,Rr | Compare | Rd – Rr | Z, N,V,C,H | 1 |
| CPC | Rd,Rr | Compare with Carry | Rd – Rr – C | Z, N,V,C,H | 1 |
| CPI | Rd,K | Compare Register with Immediate | Rd – K | Z, N,V,C,H | 1 |
| SBRC | Rr, b | Skip if Bit in Register Cleared | if (Rr(b)=0) PC ← PC + 2 or 3 | None | 1 / 2 / 3 |
| SBRS | Rr, b | Skip if Bit in Register is Set | if (Rr(b)=1) PC ← PC + 2 or 3 | None | 1 / 2 / 3 |
| SBIC | P, b | Skip if Bit in I/O Register Cleared | if (P(b)=0) PC ← PC + 2 or 3 | None | 1 / 2 / 3 |
| SBIS | P, b | Skip if Bit in I/O Register is Set | if (P(b)=1) PC ← PC + 2 or 3 | None | 1 / 2 / 3 |
| BRBS | s, k | Branch if Status Flag Set | if (SREG(s) = 1) then PC←PC+k + 1 | None | 1 / 2 |
| BRBC | s, k | Branch if Status Flag Cleared | if (SREG(s) = 0) then PC←PC+k + 1 | None | 1 / 2 |
| BREQ | k | Branch if Equal | if (Z = 1) then PC ← PC + k + 1 | None | 1 / 2 |
| BRNE | k | Branch if Not Equal | if (Z = 0) then PC ← PC + k + 1 | None | 1 / 2 |
| BRCS | k | Branch if Carry Set | if (C = 1) then PC ← PC + k + 1 | None | 1 / 2 |
| BRCC | k | Branch if Carry Cleared | if (C = 0) then PC ← PC + k + 1 | None | 1 / 2 |
| BRSH | k | Branch if Same or Higher | if (C = 0) then PC ← PC + k + 1 | None | 1 / 2 |
| BRLO | k | Branch if Lower | if (C = 1) then PC ← PC + k + 1 | None | 1 / 2 |
| BRMI | k | Branch if Minus | if (N = 1) then PC ← PC + k + 1 | None | 1 / 2 |
| BRPL | k | Branch if Plus | if (N = 0) then PC ← PC + k + 1 | None | 1 / 2 |
| BRGE | k | Branch if Greater or Equal, Signed | if (N ⊕ V= 0) then PC ← PC + k + 1 | None | 1 / 2 |
| BRLT | k | Branch if Less Than Zero, Signed | if (N ⊕ V= 1) then PC ← PC + k + 1 | None | 1 / 2 |
| BRHS | k | Branch if Half Carry Flag Set | if (H = 1) then PC ← PC + k + 1 | None | 1 / 2 |
| BRHC | k | Branch if Half Carry Flag Cleared | if (H = 0) then PC ← PC + k + 1 | None | 1 / 2 |
| BRTS | k | Branch if T Flag Set | if (T = 1) then PC ← PC + k + 1 | None | 1 / 2 |
| BRTC | k | Branch if T Flag Cleared | if (T = 0) then PC ← PC + k + 1 | None | 1 / 2 |
| BRVS | k | Branch if Overflow Flag is Set | if (V = 1) then PC ← PC + k + 1 | None | 1 / 2 |
| BRVC | k | Branch if Overflow Flag is Cleared | if (V = 0) then PC ← PC + k + 1 | None | 1 / 2 |

## Instruction Set Summary (Continued)

| Mnemonics | Operands | Description | Operation | Flags | #Clocks |
|---|---|---|---|---|---|
| BRIE | k | Branch if Interrupt Enabled | if ( I = 1) then PC ← PC + k + 1 | None | 1 / 2 |
| BRID | k | Branch if Interrupt Disabled | if ( I = 0) then PC ← PC + k + 1 | None | 1 / 2 |
| **DATA TRANSFER INSTRUCTIONS** | | | | | |
| MOV | Rd, Rr | Move Between Registers | Rd ← Rr | None | 1 |
| MOVW | Rd, Rr | Copy Register Word | Rd+1:Rd ← Rr+1:Rr | None | 1 |
| LDI | Rd, K | Load Immediate | Rd ← K | None | 1 |
| LD | Rd, X | Load Indirect | Rd ← (X) | None | 2 |
| LD | Rd, X+ | Load Indirect and Post-Inc. | Rd ← (X), X ← X + 1 | None | 2 |
| LD | Rd, - X | Load Indirect and Pre-Dec. | X ← X - 1, Rd ← (X) | None | 2 |
| LD | Rd, Y | Load Indirect | Rd ← (Y) | None | 2 |
| LD | Rd, Y+ | Load Indirect and Post-Inc. | Rd ← (Y), Y ← Y + 1 | None | 2 |
| LD | Rd, - Y | Load Indirect and Pre-Dec. | Y ← Y - 1, Rd ← (Y) | None | 2 |
| LDD | Rd,Y+q | Load Indirect with Displacement | Rd ← (Y + q) | None | 2 |
| LD | Rd, Z | Load Indirect | Rd ← (Z) | None | 2 |
| LD | Rd, Z+ | Load Indirect and Post-Inc. | Rd ← (Z), Z ← Z+1 | None | 2 |
| LD | Rd, -Z | Load Indirect and Pre-Dec. | Z ← Z - 1, Rd ← (Z) | None | 2 |
| LDD | Rd, Z+q | Load Indirect with Displacement | Rd ← (Z + q) | None | 2 |
| LDS | Rd, k | Load Direct from SRAM | Rd ← (k) | None | 2 |
| ST | X, Rr | Store Indirect | (X) ← Rr | None | 2 |
| ST | X+, Rr | Store Indirect and Post-Inc. | (X) ← Rr, X ← X + 1 | None | 2 |
| ST | - X, Rr | Store Indirect and Pre-Dec. | X ← X - 1, (X) ← Rr | None | 2 |
| ST | Y, Rr | Store Indirect | (Y) ← Rr | None | 2 |
| ST | Y+, Rr | Store Indirect and Post-Inc. | (Y) ← Rr, Y ← Y + 1 | None | 2 |
| ST | - Y, Rr | Store Indirect and Pre-Dec. | Y ← Y - 1, (Y) ← Rr | None | 2 |
| STD | Y+q,Rr | Store Indirect with Displacement | (Y + q) ← Rr | None | 2 |
| ST | Z, Rr | Store Indirect | (Z) ← Rr | None | 2 |
| ST | Z+, Rr | Store Indirect and Post-Inc. | (Z) ← Rr, Z ← Z + 1 | None | 2 |
| ST | -Z, Rr | Store Indirect and Pre-Dec. | Z ← Z - 1, (Z) ← Rr | None | 2 |
| STD | Z+q,Rr | Store Indirect with Displacement | (Z + q) ← Rr | None | 2 |
| STS | k, Rr | Store Direct to SRAM | (k) ← Rr | None | 2 |
| LPM | | Load Program Memory | R0 ← (Z) | None | 3 |
| LPM | Rd, Z | Load Program Memory | Rd ← (Z) | None | 3 |
| LPM | Rd, Z+ | Load Program Memory and Post-Inc | Rd ← (Z), Z ← Z+1 | None | 3 |
| ELPM | | Extended Load Program Memory | R0 ← (RAMPZ:Z) | None | 3 |
| ELPM | Rd, Z | Extended Load Program Memory | Rd ← (RAMPZ:Z) | None | 3 |
| ELPM | Rd, Z+ | Extended Load Program Memory and Post-Inc | Rd ← (RAMPZ:Z), RAMPZ:Z ← RAMPZ:Z+1 | None | 3 |
| SPM | | Store Program Memory | (Z) ← R1:R0 | None | - |
| IN | Rd, P | In Port | Rd ← P | None | 1 |
| OUT | P, Rr | Out Port | P ← Rr | None | 1 |
| PUSH | Rr | Push Register on Stack | STACK ← Rr | None | 2 |
| POP | Rd | Pop Register from Stack | Rd ← STACK | None | 2 |
| **BIT AND BIT-TEST INSTRUCTIONS** | | | | | |
| SBI | P,b | Set Bit in I/O Register | I/O(P,b) ← 1 | None | 2 |
| CBI | P,b | Clear Bit in I/O Register | I/O(P,b) ← 0 | None | 2 |
| LSL | Rd | Logical Shift Left | Rd(n+1) ← Rd(n), Rd(0) ← 0 | Z,C,N,V | 1 |
| LSR | Rd | Logical Shift Right | Rd(n) ← Rd(n+1), Rd(7) ← 0 | Z,C,N,V | 1 |
| ROL | Rd | Rotate Left Through Carry | Rd(0)←C,Rd(n+1)← Rd(n),C←Rd(7) | Z,C,N,V | 1 |
| ROR | Rd | Rotate Right Through Carry | Rd(7)←C,Rd(n)← Rd(n+1),C←Rd(0) | Z,C,N,V | 1 |
| ASR | Rd | Arithmetic Shift Right | Rd(n) ← Rd(n+1), n=0..6 | Z,C,N,V | 1 |
| SWAP | Rd | Swap Nibbles | Rd(3..0)←Rd(7..4),Rd(7..4)←Rd(3..0) | None | 1 |
| BSET | s | Flag Set | SREG(s) ← 1 | SREG(s) | 1 |
| BCLR | s | Flag Clear | SREG(s) ← 0 | SREG(s) | 1 |
| BST | Rr, b | Bit Store from Register to T | T ← Rr(b) | T | 1 |
| BLD | Rd, b | Bit load from T to Register | Rd(b) ← T | None | 1 |
| SEC | | Set Carry | C ← 1 | C | 1 |
| CLC | | Clear Carry | C ← 0 | C | 1 |
| SEN | | Set Negative Flag | N ← 1 | N | 1 |
| CLN | | Clear Negative Flag | N ← 0 | N | 1 |
| SEZ | | Set Zero Flag | Z ← 1 | Z | 1 |
| CLZ | | Clear Zero Flag | Z ← 0 | Z | 1 |
| SEI | | Global Interrupt Enable | I ← 1 | I | 1 |
| CLI | | Global Interrupt Disable | I ← 0 | I | 1 |
| SES | | Set Signed Test Flag | S ← 1 | S | 1 |
| CLS | | Clear Signed Test Flag | S ← 0 | S | 1 |

## Instruction Set Summary (Continued)

| Mnemonics | Operands | Description | Operation | Flags | #Clocks |
|---|---|---|---|---|---|
| SEV | | Set Twos Complement Overflow. | V ← 1 | V | 1 |
| CLV | | Clear Twos Complement Overflow | V ← 0 | V | 1 |
| SET | | Set T in SREG | T ← 1 | T | 1 |
| CLT | | Clear T in SREG | T ← 0 | T | 1 |
| SEH | | Set Half Carry Flag in SREG | H ← 1 | H | 1 |
| CLH | | Clear Half Carry Flag in SREG | H ← 0 | H | 1 |
| **MCU CONTROL INSTRUCTIONS** | | | | | |
| NOP | | No Operation | | None | 1 |
| SLEEP | | Sleep | (see specific descr. for Sleep function) | None | 1 |
| WDR | | Watchdog Reset | (see specific descr. for WDR/timer) | None | 1 |
| BREAK | | Break | For On-chip Debug Only | None | N/A |

## Ordering Information

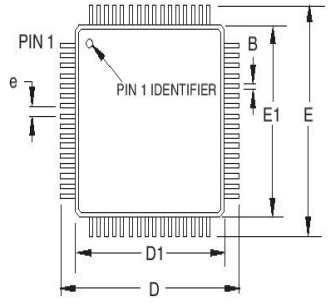
| Speed (MHz) | Power Supply | Ordering Code[1] | Package[2] | Operation Range |
|---|---|---|---|---|
| 8 | 2.7 – 5.5V | ATmega128L-8AU<br>ATmega128L-8AUR[3]<br>ATmega128L-8MU<br>ATmega128L-8MUR[3] | 64A<br>64A<br>64M1<br>64M1 | Industrial (-40°C to 85°C) |
| 16 | 4.5 – 5.5V | ATmega128-16AU<br>ATmega128-16AUR[3]<br>ATmega128-16MU<br>ATmega128-16MUR[3] | 64A<br>64A<br>64M1<br>64M1 | |
| 8 | 3.0 – 5.5V | ATmega128L–8AN<br>ATmega128L–8ANR[3]<br>ATmega128L–8MN<br>ATmega128L–8MNR[3] | 64A<br>64A<br>64M1<br>64M1 | Extended (-40°C to 105°C) |
| 16 | 4.5 – 5.5V | ATmega128–16AN<br>ATmega128–16ANR[3]<br>ATmega128–16MN<br>ATmega128–16MNR[3] | 64A<br>64A<br>64M1<br>64M1 | |

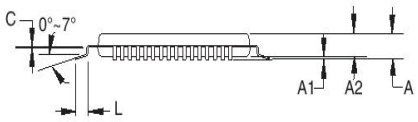
Notes:
1. Pb-free packaging complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
2. The device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
3. Tape and Reel

| Package Type | |
|---|---|
| **64A** | 64-lead, 14 x 14 x 1.0mm, Thin Profile Plastic Quad Flat Package (TQFP) |
| **64M1** | 64-pad, 9 x 9 x 1.0mm, Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF) |

## Packaging Information

### 64A




**COMMON DIMENSIONS**
(Unit of Measure = mm)

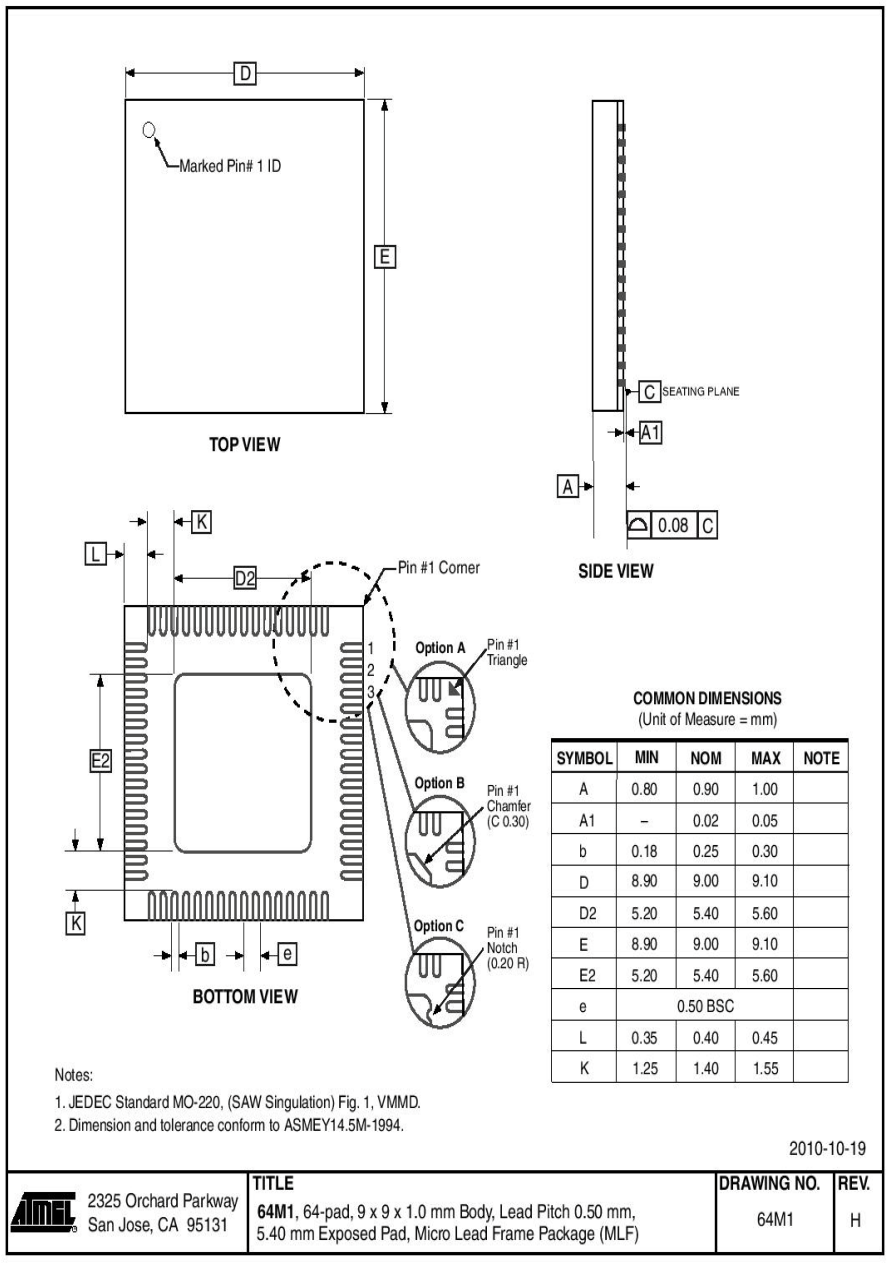
| SYMBOL | MIN | NOM | MAX | NOTE |
|---|---|---|---|---|
| A | – | – | 1.20 | |
| A1 | 0.05 | – | 0.15 | |
| A2 | 0.95 | 1.00 | 1.05 | |
| D | 15.75 | 16.00 | 16.25 | |
| D1 | 13.90 | 14.00 | 14.10 | Note 2 |
| E | 15.75 | 16.00 | 16.25 | |
| E1 | 13.90 | 14.00 | 14.10 | Note 2 |
| B | 0.30 | – | 0.45 | |
| C | 0.09 | – | 0.20 | |
| L | 0.45 | – | 0.75 | |
| e | 0.80 TYP | | | |

Notes:
1. This package conforms to JEDEC reference MS-026, Variation AEB.
2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is 0.25 mm per side. Dimensions D1 and E1 are maximum plastic body size dimensions including mold mismatch.
3. Lead coplanarity is 0.10 mm maximum.

2010-10-20

| | TITLE | DRAWING NO. | REV. |
|---|---|---|---|
| 2325 Orchard Parkway<br>San Jose, CA 95131 | **64A**, 64-lead, 14 x 14 mm Body Size, 1.0 mm Body Thickness, 0.8 mm Lead Pitch, Thin Profile Plastic Quad Flat Package (TQFP) | 64A | C |

## 64M1




**COMMON DIMENSIONS**
(Unit of Measure = mm)

| SYMBOL | MIN | NOM | MAX | NOTE |
|---|---|---|---|---|
| A | 0.80 | 0.90 | 1.00 | |
| A1 | – | 0.02 | 0.05 | |
| b | 0.18 | 0.25 | 0.30 | |
| D | 8.90 | 9.00 | 9.10 | |
| D2 | 5.20 | 5.40 | 5.60 | |
| E | 8.90 | 9.00 | 9.10 | |
| E2 | 5.20 | 5.40 | 5.60 | |
| e | 0.50 BSC | | | |
| L | 0.35 | 0.40 | 0.45 | |
| K | 1.25 | 1.40 | 1.55 | |

Notes:
1. JEDEC Standard MO-220, (SAW Singulation) Fig. 1, VMMD.
2. Dimension and tolerance conform to ASMEY14.5M-1994.

2010-10-19

| | TITLE | DRAWING NO. | REV. |
|---|---|---|---|
| 2325 Orchard Parkway<br>San Jose, CA 95131 | **64M1**, 64-pad, 9 x 9 x 1.0 mm Body, Lead Pitch 0.50 mm,<br>5.40 mm Exposed Pad, Micro Lead Frame Package (MLF) | 64M1 | H |

## Errata

The revision letter in this section refers to the revision of the ATmega128 device.

### ATmega128 Rev. F to M

- **First Analog Comparator conversion may be delayed**
- **Interrupts may be lost when writing the timer registers in the asynchronous timer**
- **Stabilizing time needed when changing XDIV Register**
- **Stabilizing time needed when changing OSCCAL Register**
- **IDCODE masks data from TDI input**
- **Reading EEPROM by using ST or STS to set EERE bit triggers unexpected interrupt request**

1. **First Analog Comparator conversion may be delayed**

   If the device is powered by a slow rising $V_{CC}$, the first Analog Comparator conversion will take longer than expected on some devices.

   **Problem Fix/Workaround**

   When the device has been powered or reset, disable then enable theAnalog Comparator before the first conversion.

2. **Interrupts may be lost when writing the timer registers in the asynchronous timer**

   The interrupt will be lost if a timer register that is synchronous timer clock is written when the asynchronous Timer/Counter register (TCNTx) is 0x00.

   **Problem Fix/Workaround**

   Always check that the asynchronous Timer/Counter register neither have the value 0xFF nor 0x00 before writing to the asynchronous Timer Control Register (TCCRx), asynchronous Timer Counter Register (TCNTx), or asynchronous Output Compare Register (OCRx).

3. **Stabilizing time needed when changing XDIV Register**

   After increasing the source clock frequency more than 2% with settings in the XDIV register, the device may execute some of the subsequent instructions incorrectly.

   **Problem Fix / Workaround**

   The NOP instruction will always be executed correctly also right after a frequency change. Thus, the next 8 instructions after the change should be NOP instructions. To ensure this, follow this procedure:

   1.Clear the I bit in the SREG Register.

   2.Set the new pre-scaling factor in XDIV register.

   3.Execute 8 NOP instructions

   4.Set the I bit in SREG

   This will ensure that all subsequent instructions will execute correctly.

   Assembly Code Example:

```
CLI              ; clear global interrupt enable
OUT  XDIV, temp  ; set new prescale value
NOP              ; no operation
NOP              ; no operation
NOP              ; no operation
NOP              ; no operation
NOP              ; no operation
NOP              ; no operation
NOP              ; no operation
NOP              ; no operation
```

```
SEI                 ; set global interrupt enable
```

4. **Stabilizing time needed when changing OSCCAL Register**

   After increasing the source clock frequency more than 2% with settings in the OSCCAL register, the device may execute some of the subsequent instructions incorrectly.

   **Problem Fix / Workaround**

   The behavior follows errata number 3., and the same Fix / Workaround is applicable on this errata.

5. **IDCODE masks data from TDI input**

   The JTAG instruction IDCODE is not working correctly. Data to succeeding devices are replaced by all-ones during Update-DR.

   **Problem Fix / Workaround**

   – If ATmega128 is the only device in the scan chain, the problem is not visible.
   – Select the Device ID Register of the ATmega128 by issuing the IDCODE instruction or by entering the Test-Logic-Reset state of the TAP controller to read out the contents of its Device ID Register and possibly data from succeeding devices of the scan chain. Issue the BYPASS instruction to the ATmega128 while reading the Device ID Registers of preceding devices of the boundary scan chain.
   – If the Device IDs of all devices in the boundary scan chain must be captured simultaneously, the ATmega128 must be the fist device in the chain.

6. **Reading EEPROM by using ST or STS to set EERE bit triggers unexpected interrupt request.**

   Reading EEPROM by using the ST or STS command to set the EERE bit in the EECR register triggers an unexpected EEPROM interrupt request.

   **Problem Fix / Workaround**

   Always use OUT or SBI to set EERE in EECR.

## 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
- Nonvolatile Program and Data Memories
  - 32K 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
  - 1024 Bytes EEPROM
    Endurance: 100,000 Write/Erase Cycles
  - 2K Byte Internal SRAM
  - 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
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Four PWM Channels
  - 8-channel, 10-bit ADC
    8 Single-ended Channels
    7 Differential Channels in TQFP Package Only
    2 Differential Channels with Programmable Gain at 1x, 10x, or 200x
  - Byte-oriented Two-wire Serial Interface
  - Programmable Serial USART
  - 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
  - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby and Extended Standby
- I/O and Packages
  - 32 Programmable I/O Lines
  - 40-pin PDIP, 44-lead TQFP, and 44-pad MLF
- Operating Voltages
  - 2.7 - 5.5V for ATmega32L
  - 4.5 - 5.5V for ATmega32
- Speed Grades
  - 0 - 8 MHz for ATmega32L
  - 0 - 16 MHz for ATmega32
- Power Consumption at 1 MHz, 3V, 25°C for ATmega32L
  - Active: 1.1 mA
  - Idle Mode: 0.35 mA
  - Power-down Mode: < 1 µA

# 8-bit AVR® Microcontroller with 32K Bytes In-System Programmable Flash

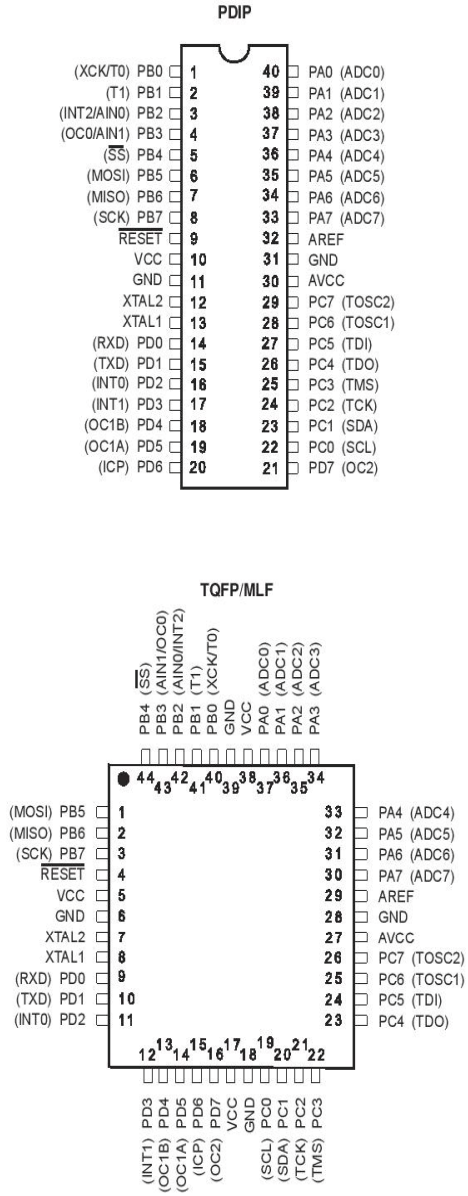
## ATmega32
## ATmega32L

## Preliminary

2503F–AVR–12/03

## Pin Configurations

**Figure 1.** Pinouts ATmega32




## Disclaimer

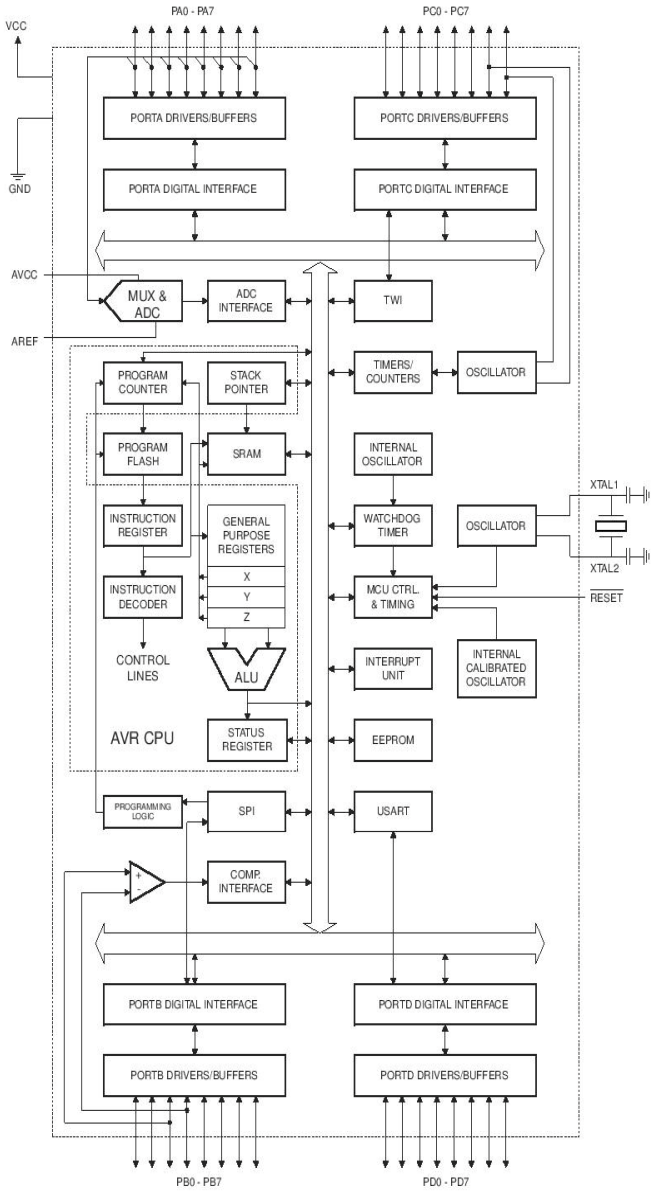
Typical values contained in this datasheet are based on simulations and characterization of other AVR microcontrollers manufactured on the same process technology. Min and Max values will be available after the device is characterized.

## Overview

The ATmega32 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega32 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

## Block Diagram

**Figure 2.** Block Diagram



The AVR core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The ATmega32 provides the following features: 32K bytes of In-System Programmable Flash Program memory with Read-While-Write capabilities, 1024 bytes EEPROM, 2K byte SRAM, 32 general purpose I/O lines, 32 general purpose working registers, a JTAG interface for Boundary-scan, On-chip Debugging support and programming, three flexible Timer/Counters with compare modes, Internal and External Interrupts, a serial programmable USART, a byte oriented Two-wire Serial Interface, an 8-channel, 10-bit ADC with optional differential input stage with programmable gain (TQFP package only), a programmable Watchdog Timer with Internal Oscillator, an SPI serial port, and six software selectable power saving modes. The Idle mode stops the CPU while allowing the USART, Two-wire interface, A/D Converter, SRAM, Timer/Counters, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next External Interrupt or Hardware Reset. In Power-save mode, the Asynchronous Timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except Asynchronous Timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low-power consumption. In Extended Standby mode, both the main Oscillator and the Asynchronous Timer continue to run.

The device is manufactured using Atmel's high density nonvolatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed in-system through an SPI serial interface, by a conventional nonvolatile memory programmer, or by an On-chip Boot program running on the AVR core. The boot program can use any interface to download the application program in the Application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega32 is a powerful microcontroller that provides a highly-flexible and cost-effective solution to many embedded control applications.

The ATmega32 AVR is supported with a full suite of program and system development tools including: C compilers, macro assemblers, program debugger/simulators, in-circuit emulators, and evaluation kits.

## Pin Descriptions

**VCC** Digital supply voltage.

**GND** Ground.

**Port A (PA7..PA0)** Port A serves as the analog inputs to the A/D Converter.

Port A also serves as an 8-bit bi-directional I/O port, if the A/D Converter is not used. Port pins can provide internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

**Port B (PB7..PB0)**

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port B also serves the functions of various special features of the ATmega32 as listed on page 55.

**Port C (PC7..PC0)**

Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running. If the JTAG interface is enabled, the pull-up resistors on pins PC5(TDI), PC3(TMS) and PC2(TCK) will be activated even if a reset occurs.

The TD0 pin is tri-stated unless TAP states that shift out data are entered.

Port C also serves the functions of the JTAG interface and other special features of the ATmega32 as listed on page 58.

**Port D (PD7..PD0)**

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port D also serves the functions of various special features of the ATmega32 as listed on page 60.

**$\overline{\text{RESET}}$**

Reset Input. A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running. The minimum pulse length is given in Table 15 on page 35. Shorter pulses are not guaranteed to generate a reset.

**XTAL1**

Input to the inverting Oscillator amplifier and input to the internal clock operating circuit.

**XTAL2**

Output from the inverting Oscillator amplifier.

**AVCC**

AVCC is the supply voltage pin for Port A and the A/D Converter. It should be externally connected to $V_{CC}$, even if the ADC is not used. If the ADC is used, it should be connected to $V_{CC}$ through a low-pass filter.

**AREF**

AREF is the analog reference pin for the A/D Converter.

## About Code Examples

This documentation contains simple code examples that briefly show how to use various parts of the device. These code examples assume that the part specific header file is included before compilation. Be aware that not all C Compiler vendors include bit definitions in the header files and interrupt handling in C is compiler dependent. Please confirm with the C Compiler documentation for more details.

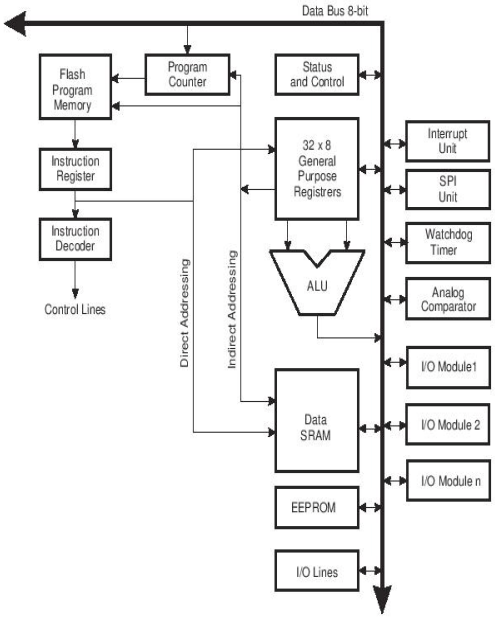
## AVR CPU Core

### Introduction

This section discusses the AVR core architecture in general. The main function of the CPU core is to ensure correct program execution. The CPU must therefore be able to access memories, perform calculations, control peripherals, and handle interrupts.

### Architectural Overview

**Figure 3.** Block Diagram of the AVR MCU Architecture




In order to maximize performance and parallelism, the AVR uses a Harvard architecture – with separate memories and buses for program and data. Instructions in the program memory are executed with a single level pipelining. While one instruction is being executed, the next instruction is pre-fetched from the program memory. This concept enables instructions to be executed in every clock cycle. The program memory is In-System Reprogrammable Flash memory.

The fast-access Register File contains 32 x 8-bit general purpose working registers with a single clock cycle access time. This allows single-cycle Arithmetic Logic Unit (ALU) operation. In a typical ALU operation, two operands are output from the Register File, the operation is executed, and the result is stored back in the Register File – in one clock cycle.

Six of the 32 registers can be used as three 16-bit indirect address register pointers for Data Space addressing – enabling efficient address calculations. One of the these address pointers can also be used as an address pointer for look up tables in Flash Program memory. These added function registers are the 16-bit X-, Y-, and Z-register, described later in this section.

The ALU supports arithmetic and logic operations between registers or between a constant and a register. Single register operations can also be executed in the ALU. After

an arithmetic operation, the Status Register is updated to reflect information about the result of the operation.

Program flow is provided by conditional and unconditional jump and call instructions, able to directly address the whole address space. Most AVR instructions have a single 16-bit word format. Every program memory address contains a 16- or 32-bit instruction.

Program Flash memory space is divided in two sections, the Boot program section and the Application Program section. Both sections have dedicated Lock bits for write and read/write protection. The SPM instruction that writes into the Application Flash memory section must reside in the Boot Program section.

During interrupts and subroutine calls, the return address Program Counter (PC) is stored on the Stack. The Stack is effectively allocated in the general data SRAM, and consequently the Stack size is only limited by the total SRAM size and the usage of the SRAM. All user programs must initialize the SP in the reset routine (before subroutines or interrupts are executed). The Stack Pointer SP is read/write accessible in the I/O space. The data SRAM can easily be accessed through the five different addressing modes supported in the AVR architecture.

The memory spaces in the AVR architecture are all linear and regular memory maps.

A flexible interrupt module has its control registers in the I/O space with an additional global interrupt enable bit in the Status Register. All interrupts have a separate interrupt vector in the interrupt vector table. The interrupts have priority in accordance with their interrupt vector position. The lower the interrupt vector address, the higher the priority.

The I/O memory space contains 64 addresses for CPU peripheral functions as Control Registers, SPI, and other I/O functions. The I/O Memory can be accessed directly, or as the Data Space locations following those of the Register File, $20 - $5F.

### ALU – Arithmetic Logic Unit

The high-performance AVR ALU operates in direct connection with all the 32 general purpose working registers. Within a single clock cycle, arithmetic operations between general purpose registers or between a register and an immediate are executed. The ALU operations are divided into three main categories – arithmetic, logical, and bit-functions. Some implementations of the architecture also provide a powerful multiplier supporting both signed/unsigned multiplication and fractional format. See the “Instruction Set” section for a detailed description.

DALLAS SEMICONDUCTOR

# DS1307
# 64 X 8 Serial Real Time Clock

www.dalsemi.com

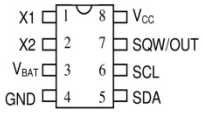
## FEATURES

- Real time clock counts seconds, minutes, hours, date of the month, month, day of the week, and year with leap year compensation valid up to 2100
- 56 byte nonvolatile RAM for data storage
- 2-wire serial interface
- Programmable squarewave output signal
- Automatic power-fail detect and switch circuitry
- Consumes less than 500 nA in battery backup mode with oscillator running
- Optional industrial temperature range -40°C to +85°C
- Available in 8-pin DIP or SOIC
- Recognized by Underwriters Laboratory

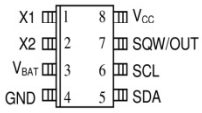
## ORDERING INFORMATION

| | |
|---|---|
| DS1307 | 8-Pin DIP |
| DS1307Z | 8-Pin SOIC (150 mil) |
| DS1307N | 8-Pin DIP (Industrial) |
| DS1307ZN | 8-Pin SOIC (Industrial) |

## PIN ASSIGNMENT




DS1307 8-Pin DIP (300 mil)




DS1307Z 8-Pin SOIC (150 mil)

## PIN DESCRIPTION

| | |
|---|---|
| $V_{CC}$ | - Primary Power Supply |
| X1, X2 | - 32.768 kHz Crystal Connection |
| $V_{BAT}$ | - +3V Battery Input |
| GND | - Ground |
| SDA | - Serial Data |
| SCL | - Serial Clock |
| SQW/OUT | - Square wave/Output Driver |

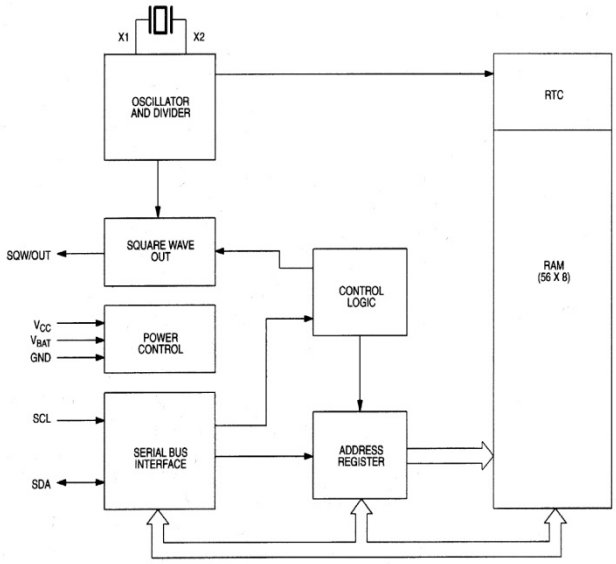
## DESCRIPTION

The DS1307 Serial Real Time Clock is a low power, full BCD clock/calendar plus 56 bytes of nonvolatile SRAM. Address and data are transferred serially via a 2-wire bi-directional bus. The clock/calendar provides seconds, minutes, hours, day, date, month, and year information. The end of the month date is automatically adjusted for months with less than 31 days, including corrections for leap year. The clock operates in either the 24-hour or 12-hour format with AM/PM indicator. The DS1307 has a built-in power sense circuit which detects power failures and automatically switches to the battery supply.

## OPERATION

The DS1307 operates as a slave device on the serial bus. Access is obtained by implementing a START condition and providing a device identification code followed by a register address. Subsequent registers can be accessed sequentially until a STOP condition is executed. When $V_{CC}$ falls below 1.25 x $V_{BAT}$ the device terminates an access in progress and resets the device address counter. Inputs to the device will not be recognized at this time to prevent erroneous data from being written to the device from an out of tolerance system. When $V_{CC}$ falls below $V_{BAT}$ the device switches into a low current battery backup mode. Upon power up, the device switches from battery to $V_{CC}$ when $V_{CC}$ is greater than $V_{BAT}$ +0.2V and recognizes inputs when $V_{CC}$ is greater than 1.25 x $V_{BAT}$. The block diagram in Figure 1 shows the main elements of the Serial Real Time Clock.

## DS1307 BLOCK DIAGRAM Figure 1




## SIGNAL DESCRIPTIONS

**$V_{CC}$, GND -** DC power is provided to the device on these pins. $V_{CC}$ is the +5 volt input. When 5 volts is applied within normal limits, the device is fully accessible and data can be written and read. When a 3-volt battery is connected to the device and $V_{CC}$ is below 1.25 x $V_{BAT}$, reads and writes are inhibited. However, the Timekeeping function continues unaffected by the lower input voltage. As $V_{CC}$ falls below $V_{BAT}$ the RAM and timekeeper are switched over to the external power supply (nominal 3.0V DC) at $V_{BAT}$.

**$V_{BAT}$** - Battery input for any standard 3-volt lithium cell or other energy source. Battery voltage must be held between 2.0 and 3.5 volts for proper operation. The nominal write protect trip point voltage at which access to the real time clock and user RAM is denied is set by the internal circuitry as 1.25 x $V_{BAT}$ nominal. A lithium battery with 48 mAhr or greater will back up the DS1307 for more than 10 years in the absence of power at 25 degrees C.

**SCL (Serial Clock Input)** - SCL is used to synchronize data movement on the serial interface.

**SDA (Serial Data Input/Output) -** SDA is the input/output pin for the 2-wire serial interface. The SDA pin is open drain which requires an external pullup resistor.

**SQW/OUT (Square Wave/ Output Driver) -** When enabled, the SQWE bit set to 1, the SQW/OUT pin outputs one of four square wave frequencies (1 Hz, 4 kHz, 8 kHz, 32 kHz). The SQW/OUT pin is open drain which requires an external pullup resistor. SQW/OUT will operate with either Vcc or Vbat applied.

**X1, X2** - Connections for a standard 32.768 kHz quartz crystal. The internal oscillator circuitry is designed for operation with a crystal having a specified load capacitance (CL) of 12.5 pF.

For more information on crystal selection and crystal layout considerations, please consult Application Note 58, "Crystal Considerations with Dallas Real Time Clocks." The DS1307 can also be driven by an external 32.768 kHz oscillator. In this configuration, the X1 pin is connected to the external oscillator signal and the X2 pin is floated.

Please review Application Note 95, "Interfacing the DS1307 with a 8051-Compatible Microcontroller" for additional information.

## RTC AND RAM ADDRESS MAP

The address map for the RTC and RAM registers of the DS1307 is shown in Figure 2. The real time clock registers are located in address locations 00h to 07h. The RAM registers are located in address locations 08h to 3Fh. During a multi-byte access, when the address pointer reaches 3Fh, the end of RAM space, it wraps around to location 00h, the beginning of the clock space.

## DS1307 ADDRESS MAP Figure 2


## CLOCK AND CALENDAR

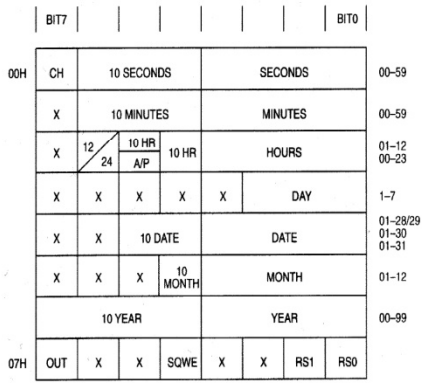
The time and calendar information is obtained by reading the appropriate register bytes. The real time clock registers are illustrated in Figure 3. The time and calendar are set or initialized by writing the appropriate register bytes. The contents of the time and calendar registers are in the Binary-Coded Decimal (BCD) format. Bit 7 of Register 0 is the Clock Halt (CH) bit. When this bit is set to a 1, the oscillator is disabled. When cleared to a 0, the oscillator is enabled.

**Please note that the initial power on state of all registers is not defined. Therefore it is important to enable the oscillator (CH bit=0) during initial configuration.**

The DS1307 can be run in either 12-hour or 24-hour mode. Bit 6 of the hours register is defined as the 12- or 24-hour mode select bit. When high, the 12-hour mode is selected. In the 12-hour mode, bit 5 is the AM/PM bit with logic high being PM. In the 24-hour mode, bit 5 is the second 10 hour bit (20-23 hours).

On a 2-wire START, the current time is transferred to a second set of registers. The time information is read from these secondary registers, while the clock may continue to run. This eliminates the need to re-read the registers in case of an update of the main registers during a read.

## DS1307 TIMEKEEPER REGISTERS Figure 3




## CONTROL REGISTER

The DS1307 Control Register is used to control the operation of the SQW/OUT pin.

| BIT 7 | BIT 6 | BIT 5 | BIT 4 | BIT 3 | BIT 2 | BIT 1 | BIT 0 |
|---|---|---|---|---|---|---|---|
| OUT | X | X | SQWE | X | X | RS1 | RS0 |

OUT (Output control): This bit controls the output level of the SQW/OUT pin when the square wave output is disabled. If SQWE=0, the logic level on the SQW/OUT pin is 1 if OUT=1 and is 0 if OUT=0.

SQWE (Square Wave Enable): This bit, when set to a logic 1, will enable the oscillator output. The frequency of the square wave output depends upon the value of the RS0 and RS1 bits.

RS (Rate Select): These bits control the frequency of the square wave output when the square wave output has been enabled. Table 1 lists the square wave frequencies that can be selected with the RS bits.

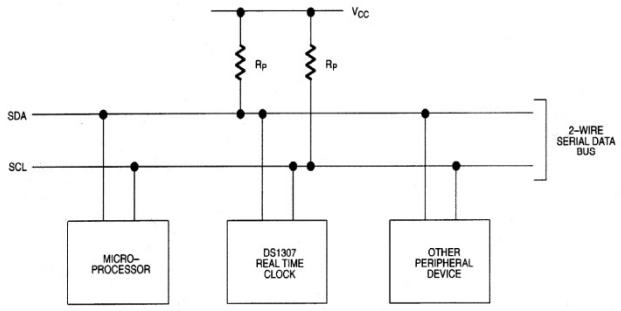
## SQUAREWAVE OUTPUT FREQUENCY Table 1

| RS1 | RS0 | SQW OUTPUT FREQUENCY |
|---|---|---|
| 0 | 0 | 1 Hz |
| 0 | 1 | 4.096 kHz |
| 1 | 0 | 8.192 kHz |
| 1 | 1 | 32.768 kHz |

## 2-WIRE SERIAL DATA BUS

The DS1307 supports a bi-directional 2-wire bus and data transmission protocol. A device that sends data onto the bus is defined as a transmitter and a device receiving data as a receiver. The device that controls the message is called a master. The devices that are controlled by the master are referred to as slaves. The bus must be controlled by a master device which generates the serial clock (SCL), controls the bus access, and generates the START and STOP conditions. The DS1307 operates as a slave on the 2-wire bus. A typical bus configuration using this 2-wire protocol is show in Figure 4.

## TYPICAL 2-WIRE BUS CONFIGURATION Figure 4




Figures 5, 6, and 7 detail how data is transferred on the 2-wire bus.

- Data transfer may be initiated only when the bus is not busy.
- During data transfer, the data line must remain stable whenever the clock line is HIGH. Changes in the data line while the clock line is high will be interpreted as control signals.

Accordingly, the following bus conditions have been defined:

**Bus not busy:** Both data and clock lines remain HIGH.

**Start data transfer:** A change in the state of the data line, from HIGH to LOW, while the clock is HIGH, defines a START condition.

**Stop data transfer:** A change in the state of the data line, from LOW to HIGH, while the clock line is HIGH, defines the STOP condition.

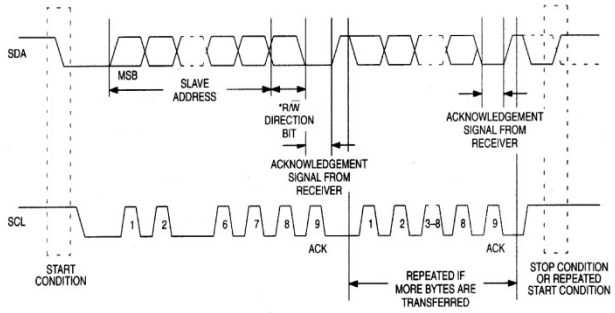
**Data valid:** The state of the data line represents valid data when, after a START condition, the data line is stable for the duration of the HIGH period of the clock signal. The data on the line must be changed during the LOW period of the clock signal. There is one clock pulse per bit of data.

Each data transfer is initiated with a START condition and terminated with a STOP condition. The number of data bytes transferred between START and STOP conditions is not limited, and is determined by the master device. The information is transferred byte-wise and each receiver acknowledges with a ninth bit. Within the 2-wire bus specifications a regular mode (100 kHz clock rate) and a fast mode (400 kHz clock rate) are defined. The DS1307 operates in the regular mode (100 kHz) only.

**Acknowledge:** Each receiving device, when addressed, is obliged to generate an acknowledge after the reception of each byte. The master device must generate an extra clock pulse which is associated with this acknowledge bit.

A device that acknowledges must pull down the SDA line during the acknowledge clock pulse in such a way that the SDA line is stable LOW during the HIGH period of the acknowledge related clock pulse. Of course, setup and hold times must be taken into account. A master must signal an end of data to the slave by not generating an acknowledge bit on the last byte that has been clocked out of the slave. In this case, the slave must leave the data line HIGH to enable the master to generate the STOP condition.

## DATA TRANSFER ON 2-WIRE SERIAL BUS Figure 5




Depending upon the state of the R/$\overline{W}$ bit, two types of data transfer are possible:

1. **Data transfer from a master transmitter to a slave receiver.** The first byte transmitted by the master is the slave address. Next follows a number of data bytes. The slave returns an acknowledge bit after each received byte. Data is transferred with the most significant bit (MSB) first.

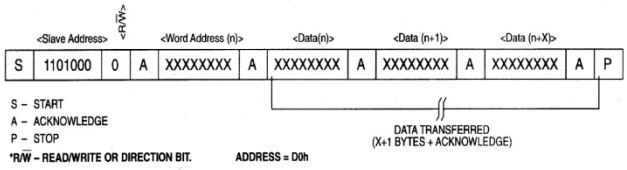
2. **Data transfer from a slave transmitter to a master receiver.** The first byte (the slave address) is transmitted by the master. The slave then returns an acknowledge bit. This is followed by the slave transmitting a number of data bytes. The master returns an acknowledge bit after all received bytes other than the last byte. At the end of the last received byte, a 'not acknowledge' is returned.

The master device generates all of the serial clock pulses and the START and STOP conditions. A transfer is ended with a STOP condition or with a repeated START condition. Since a repeated START condition is also the beginning of the next serial transfer, the bus will not be released. Data is transferred with the most significant bit (MSB) first.

The DS1307 may operate in the following two modes:

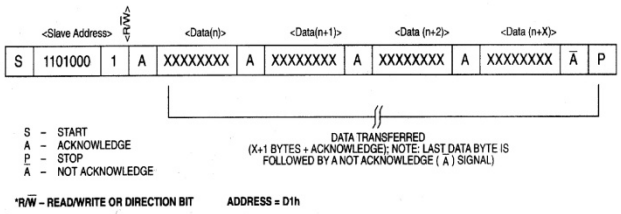
1. **Slave receiver mode (DS1307 write mode):** Serial data and clock are received through SDA and SCL. After each byte is received an acknowledge bit is transmitted. START and STOP conditions are recognized as the beginning and end of a serial transfer. Address recognition is performed by hardware after reception of the slave address and *direction bit (See Figure 6). The address byte is the first byte received after the start condition is generated by the master. The address byte contains the 7 bit DS1307 address, which is 1101000, followed by the *direction bit ($R/\overline{W}$) which, for a write, is a 0. After receiving and decoding the address byte the device outputs an acknowledge on the SDA line. After the DS1307 acknowledges the slave address + write bit, the master transmits a register address to the DS1307 This will set the register pointer on the DS1307. The master will then begin transmitting each byte of data with the DS1307 acknowledging each byte received. The master will generate a stop condition to terminate the data write.

## DATA WRITE - SLAVE RECEIVER MODE Figure 6




2. **Slave transmitter mode (DS1307 read mode):** The first byte is received and handled as in the slave receiver mode. However, in this mode, the *direction bit will indicate that the transfer direction is reversed. Serial data is transmitted on SDA by the DS1307 while the serial clock is input on SCL. START and STOP conditions are recognized as the beginning and end of a serial transfer (See Figure 7). The address byte is the first byte received after the start condition is generated by the master. The address byte contains the 7-bit DS1307 address, which is 1101000, followed by the *direction bit ($R/\overline{W}$) which, for a read, is a 1. After receiving and decoding the address byte the device inputs an acknowledge on the SDA line. The DS1307 then begins to transmit data starting with the register address pointed to by the register pointer. If the register pointer is not written to before the initiation of a read mode the first address that is read is the last one stored in the register pointer. The DS1307 must receive a Not Acknowledge to end a read.

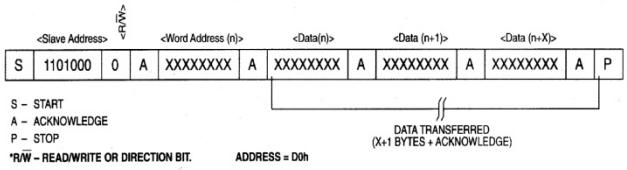
## DATA READ - SLAVE TRANSMITTER MODE Figure 7



The DS1307 may operate in the following two modes:

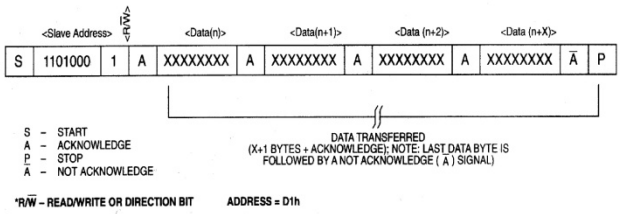
1. **Slave receiver mode (DS1307 write mode):** Serial data and clock are received through SDA and SCL. After each byte is received an acknowledge bit is transmitted. START and STOP conditions are recognized as the beginning and end of a serial transfer. Address recognition is performed by hardware after reception of the slave address and *direction bit (See Figure 6). The address byte is the first byte received after the start condition is generated by the master. The address byte contains the 7 bit DS1307 address, which is 1101000, followed by the *direction bit ($R/\overline{W}$) which, for a write, is a 0. After receiving and decoding the address byte the device outputs an acknowledge on the SDA line. After the DS1307 acknowledges the slave address + write bit, the master transmits a register address to the DS1307 This will set the register pointer on the DS1307. The master will then begin transmitting each byte of data with the DS1307 acknowledging each byte received. The master will generate a stop condition to terminate the data write.

## DATA WRITE - SLAVE RECEIVER MODE Figure 6




2. **Slave transmitter mode (DS1307 read mode):** The first byte is received and handled as in the slave receiver mode. However, in this mode, the *direction bit will indicate that the transfer direction is reversed. Serial data is transmitted on SDA by the DS1307 while the serial clock is input on SCL. START and STOP conditions are recognized as the beginning and end of a serial transfer (See Figure 7). The address byte is the first byte received after the start condition is generated by the master. The address byte contains the 7-bit DS1307 address, which is 1101000, followed by the *direction bit ($R/\overline{W}$) which, for a read, is a 1. After receiving and decoding the address byte the device inputs an acknowledge on the SDA line. The DS1307 then begins to transmit data starting with the register address pointed to by the register pointer. If the register pointer is not written to before the initiation of a read mode the first address that is read is the last one stored in the register pointer. The DS1307 must receive a Not Acknowledge to end a read.

## DATA READ - SLAVE TRANSMITTER MODE Figure 7



## ABSOLUTE MAXIMUM RATINGS*

| | |
|---|---|
| Voltage on Any Pin Relative to Ground | -0.5V to +7.0V |
| Operating Temperature | 0°C to 70°C (-40°C to 85°C for industrial) |
| Storage Temperature | -55°C to +125°C |
| Soldering Temperature | 260°C for 10 seconds DIP<br>See JPC/JEDEC Standard J-STD-020A for Surface Mount Devices |

* This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operation sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect reliability.

## RECOMMENDED DC OPERATING CONDITIONS

(0°C to 70°C or -40°C to +85°C)

| PARAMETER | SYMBOL | MIN | TYP | MAX | UNITS | NOTES |
|---|---|---|---|---|---|---|
| Supply Voltage | $V_{CC}$ | 4.5 | 5.0 | 5.5 | V | 1 |
| Logic 1 | $V_{IH}$ | 2.2 | | $V_{CC}$+0.3 | V | 1 |
| Logic 0 | $V_{IL}$ | -0.3 | | +0.8 | V | 1 |
| $V_{BAT}$ Battery Voltage | $V_{BAT}$ | 2.0 | | 3.5 | V | 1 |

## DC ELECTRICAL CHARACTERISTICS

(0°C to 70°C or -40°C to +85°C; $V_{CC}$ =4.5V to 5.5V)

| PARAMETER | SYMBOL | MIN | TYP | MAX | UNITS | NOTES |
|---|---|---|---|---|---|---|
| Input Leakage | $I_{LI}$ | | | 1 | µA | 10 |
| I/O Leakage | $I_{LO}$ | | | 1 | µA | 11 |
| Logic 0 Output | $V_{OL}$ | | | 0.4 | V | 2 |
| Active Supply Current | $I_{CCA}$ | | | 1.5 | mA | 9 |
| Standby Current | $I_{CCS}$ | | | 200 | µA | 3 |
| Battery Current (OSC ON); SQW/OUT OFF | $I_{BAT1}$ | | 300 | 500 | nA | 4 |
| Battery Current (OSC ON); SQW/OUT ON (32 kHz) | $I_{BAT2}$ | | 480 | 800 | nA | 4 |

## AC ELECTRICAL CHARACTERISTICS

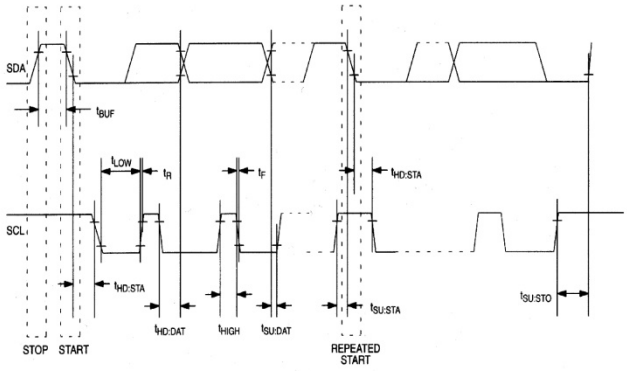
(0°C to 70°C or -40°C to +85°C; $V_{CC}$=4.5V to 5.5V)

| PARAMETER | SYMBOL | MIN | TYP | MAX | UNITS | NOTES |
|---|---|---|---|---|---|---|
| SCL Clock Frequency | $f_{SCL}$ | 0 | | 100 | kHz | |
| Bus Free Time Between a STOP and START Condition | $t_{BUF}$ | 4.7 | | | µs | |
| Hold Time (Repeated) START Condition | $t_{HD:STA}$ | 4.0 | | | µs | 5 |
| LOW Period of SCL Clock | $t_{LOW}$ | 4.7 | | | µs | |
| HIGH Period of SCL Clock | $t_{HIGH}$ | 4.0 | | | µs | |
| Set-up Time for a Repeated START Condition | $t_{SU:STA}$ | 4.7 | | | µs | |
| Data Hold Time | $t_{HD:DAT}$ | 0 | | | µs | 6, 7 |
| Data Set-up Time | $t_{SU:DAT}$ | 250 | | | ns | |
| Rise Time of Both SDA and SCL Signals | $t_R$ | | | 1000 | ns | |
| Fall Time of Both SDA and SCL Signals | $t_F$ | | | 300 | ns | |
| Set-up Time for STOP Condition | $t_{SU:STO}$ | 4.7 | | | µs | |
| Capacitive Load for each Bus Line | $C_B$ | | | 400 | pF | 8 |
| I/O Capacitance | $C_{I/O}$ | | 10 | | pF | |
| Crystal Specified Load Capacitance | | | 12.5 | | pF | |

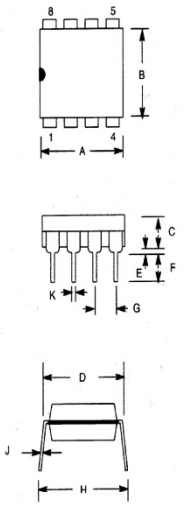
## NOTES:

1. All voltages are referenced to ground.
2. Logic zero voltages are specified at a sink current of 5 mA at $V_{CC}$=4.5V, $V_{OL}$=GND for capacitive loads.
3. $I_{CCS}$ specified with $V_{CC}$=5.0V and SDA, SCL=5.0V.
4. $V_{CC}$=0V, $V_{BAT}$=3V.
5. After this period, the first clock pulse is generated.
6. A device must internally provide a hold time of at least 300 ns for the SDA signal (referred to the $V_{IHMIN}$ of the SCL signal) in order to bridge the undefined region of the falling edge of SCL.
7. The maximum $t_{HD:DAT}$ has only to be met if the device does not stretch the LOW period ($t_{LOW}$) of the SCL signal.
8. $C_B$ - total capacitance of one bus line in pF.
9. $I_{CCA}$ - SCL clocking at max frequency = 100 kHz.
10. SCL only.
11. SDA and SQW/OUT

## TIMING DIAGRAM Figure 8

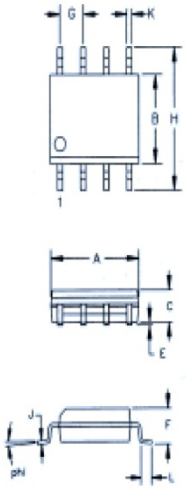



## DS1307 64 X 8 SERIAL REAL TIME CLOCK
## 8-PIN DIP MECHANICAL DIMENSIONS




| PKG | 8-PIN | |
|---|---|---|
| DIM | MIN | MAX |
| A IN. | 0.360 | 0.400 |
| MM | 9.14 | 10.16 |
| B IN. | 0.240 | 0.260 |
| MM | 6.10 | 6.60 |
| C IN. | 0.120 | 0.140 |
| MM | 3.05 | 3.56 |
| D IN. | 0.300 | 0.325 |
| MM | 7.62 | 8.26 |
| E IN. | 0.015 | 0.040 |
| MM | 0.38 | 1.02 |
| F IN. | 0.120 | 0.140 |
| MM | 3.04 | 3.56 |
| G IN. | 0.090 | 0.110 |
| MM | 2.29 | 2.79 |
| H IN. | 0.320 | 0.370 |
| MM | 8.13 | 9.40 |
| J IN. | 0.008 | 0.012 |
| MM | 0.20 | 0.30 |
| K IN. | 0.015 | 0.021 |
| MM | 0.38 | 0.53 |

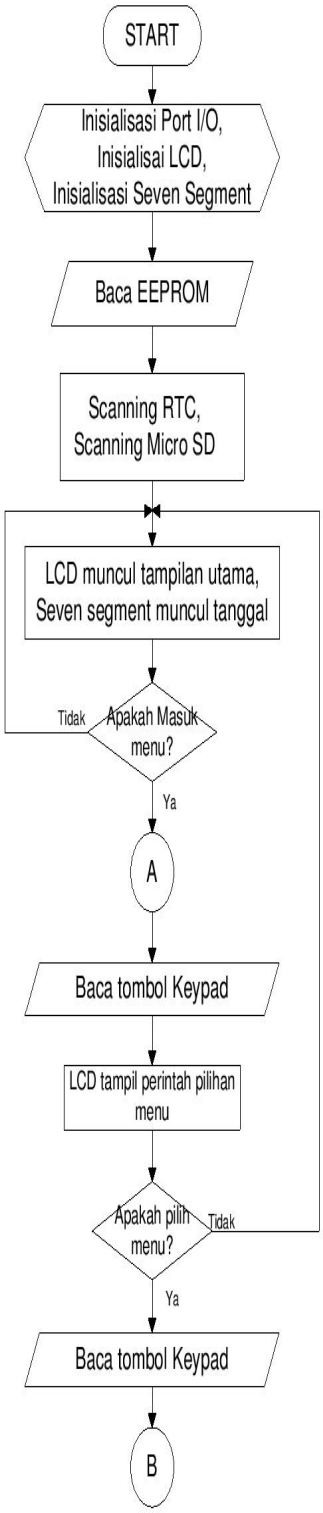
# DS1307Z 64 X 8 SERIAL REAL TIME CLOCK
# 8-PIN SOIC (150-MIL) MECHANICAL DIMENSIONS



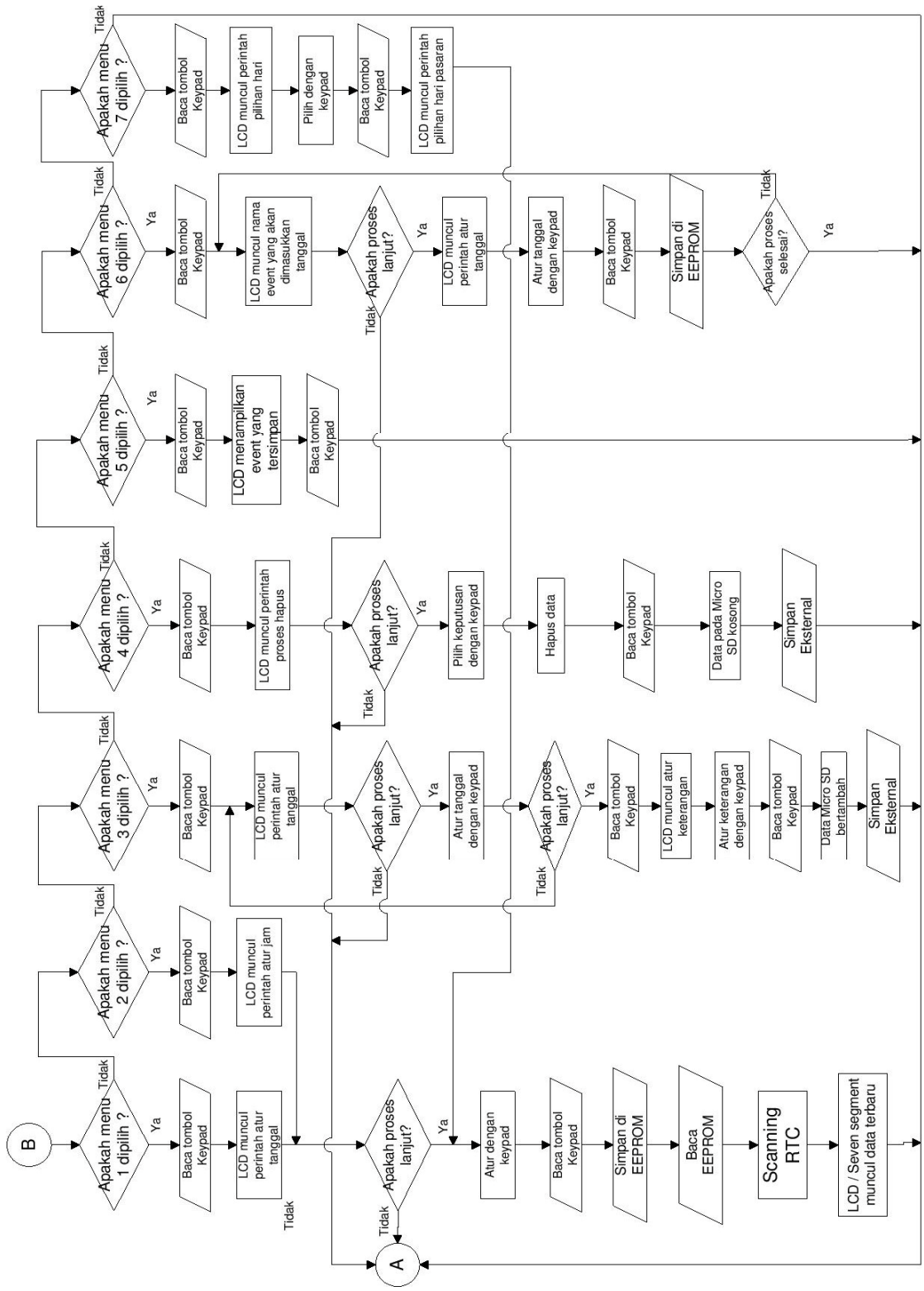

| PKG | 8-PIN (150 MIL) | |
|---|---|---|
| DIM | MIN | MAX |
| A IN. | 0.188 | 0.196 |
| MM | 4.78 | 4.98 |
| B IN. | 0.150 | 0.158 |
| MM | 3.81 | 4.01 |
| C IN. | 0.048 | 0.062 |
| MM | 1.22 | 1.57 |
| E IN. | 0.004 | 0.010 |
| MM | 0.10 | 0.25 |
| F IN. | 0.053 | 0.069 |
| MM | 1.35 | 1.75 |
| G IN. | 0.050 BSC | |
| MM | 1.27 BSC | |
| H IN. | 0.230 | 0.244 |
| MM | 5.84 | 6.20 |
| J IN. | 0.007 | 0.011 |
| MM | 0.18 | 0.28 |
| K IN. | 0.012 | 0.020 |
| MM | 0.30 | 0.51 |
| L IN. | 0.016 | 0.050 |
| MM | 0.41 | 1.27 |
| phi | 0° | 8° |

56-G2008-001

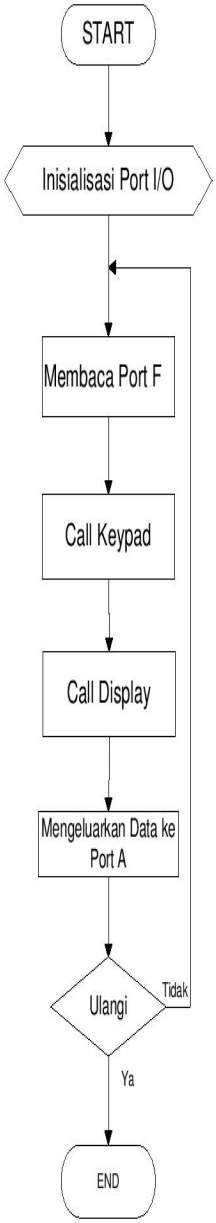
Lampiran 10. Flowchart Utama



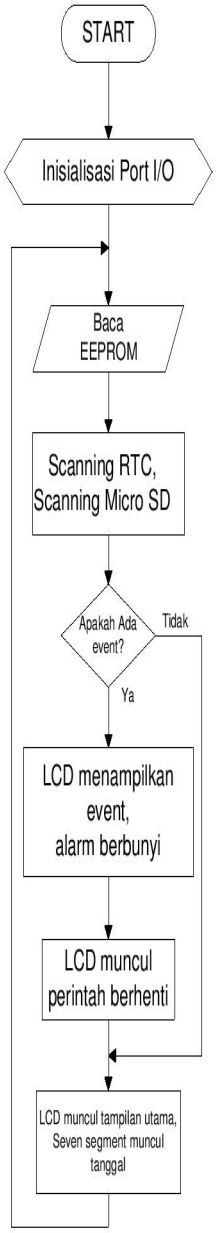
Lampiran 11. Flowchart Menu



Lampiran 12. Flowchart Pembacaan Keypad dengan Display



Lampiran 13. Flowchart Event



Lampiran 14. Manual Pengoperasian Alat

**Panduan Pengoperasian Kalender Nasional Digital Berbasis Mikrokontroler Atmega128 Dengan Tampilan Lcd Dan Seven Segment**

Oleh : Fadillah Nurohmah

NIM : 10507131031

Dosen Pembimbing : Handaru Jati, S.T., M.M., M.T., Ph.D

1. Pasang kabel *power* ke alat, lalu menghubungkan ke sumber tegangan AC 220 volt.
2. Tekan tombol saklar *ON/OFF* ditekan
3. LCD akan menyala lalu di ikuti nyala *seven segment*. Kemudian LCD akan menampilkan hari/pasaran, tanggal, jam dan menu.
4. Jika ingin memilih menu tekan tombol D, maka akan muncul 7 (tujuh) opsi menu.
5. Jika ingin mengatur tanggal tekan tombol 1.
    a. Muncul perintah atur tanggal, kemudian masukan data tanggal dengan menggunakan *keypad matrix.*
    b. Tekan tombol D untuk menyimpan atau untuk membatalkan penyimpanan tekan tombol A.
6. Apabila ingin mengatur waktu tekan tombol 2.
    a. Muncul perintah atur jam, kemudian masukan data jam dengan menggunakan *keypad matrix.*

b. Tekan tombol D untuk menyimpan atau untuk membatalkan penyimpanan tekan tombol A.

7. Jika ingin menambah *event*/hari besar tekan tombol 3.
   a. LCD akan menampilkan ‘agenda baru’ dan terdapat nomor urut agenda yang akan ditambahkan.
   b. Kemudian atur tanggal dengan memasukkan data tanggal dengan menggunakan *keypad matrix*
   c. Tekan tombol D untuk menyimpan.
   d. Lalu beri keterangan agenda tersebut
   e. Tekan D untuk menyimpan.
   f. Pada LCD akan muncul agenda yang sudah tersimpan, kemudian tekan tombol D untuk kembali ke menu.
8. Apabila ingin menghapus agenda yang sudah tersimpan tekan tombol 4.
   a. Pada LCD akan muncul ‘Anda Yakin??’.
   b. Jika yakin akan dihapus, maka tekan tombol D.
   c. Jika tidak jadi dihapus, maka tekan tombol A.
   d. Tampilan LCD akan kembali ke menu.
9. Jika ingin melihat agenda yang sudah tersimpan, maka tekan tombol 5.
   a. Kemudian akan muncul semua data agenda yang sudah tersimpan.
   b. Tekan tombol D untuk kembali ke menu.
10. Jika ingin mengatur tanggal pada hari peringatan, maka tekan tombol 6.
   a. Pada LCD akan menampilkan hari peringatan Maulid Nabi dan muncul perintah untuk memasukan tanggal.

b. Masukkan data tanggal dengan menggunakan *keypad matrix.*

c. Tekan tombol D untuk menyimpan.

d. Lakukan langkah a s/d d untuk mengisi data tanggal untuk hari peringatan berikutnya.

e. Tekan tombol D untuk menyimpan dan kembali ke menu.

11. Jika ingin mengatur hari dan pasaran, maka tekan tombol 7.

    a. Pada LCD akan muncul pilihan hari, tekan salah satu tombol pilihan untuk menentukan hari yang sesuai.

    b. Setelah memilih hari akan muncul pilihan pasaran, lalu tekan tombol pilihan pasaran.

    c. Kemudian LCD akan kembali menampilkan menu.

12. Apabila sudah tidak memilih menu lagi tekan tombol A, LCD akan kembali menampilkan tampilan utama.

13. Tombol *reset* digunakan apabila sewaktu – waktu terjadi *error.*

14. Untuk mematikan seluruh sistem tekan tombol saklar *ON/OFF.*

15. Cabut kabel *power* dari sumber tegangan AC, lalu lepas kabel *power* dari alat.