

**PAPAN LATIH BULUTANGKIS
BERBASIS MIKROKONTROLER AT89S51**

TUGAS AKHIR

Diajukan untuk memenuhi Salah Satu Syarat

Memperoleh Gelar Sarjana Teknik

Jurusan Teknik Elektro



Di susun oleh :

AGUSTINUS GIRI HARTONO

NIM : 995114057

PROGRAM STUDI TEKNIK ELEKTRO

JURUSAN TEKNIK ELEKTRO

FAKULTAS TEKNIK

UNIVERSITAS SANATA DHARMA

YOGYAKARTA

2006

**BADMINTON CIRCUIT TRAINER
BASED ON AT89S51 MICROCONTROLLER**

FINAL PROJECT

Presented as Partial Fulfillment of the Requirements

To Obtain the *Sarjana Teknik* Degree

In Electrical Engineering



By :

AGUSTINUS GIRI HARTONO

Student Number : 995114057

ELECTRICAL ENGINEERING STUDY PROGRAM

ELECTRICAL ENGINEERING DEPARTMENT

ENGINEERING FACULTY

SANATA DHARMA UNIVERSITY

YOGYAKARTA

2006

LEMBAR PERSETUJUAN
LAPORAN TUGAS AKHIR

**PAPAN LATIH BULUTANGKIS
BERBASIS MIKROKONTROLER AT89S51**

Di susun oleh :

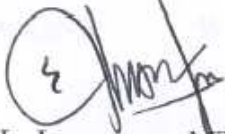
AGUSTINUS GIRI HARTONO

NIM : 995114057

NIRM : 990051123107120056

Laporan Tugas Akhir ini telah diterima dan disetujui oleh :

Pembimbing I,



(Ir. Iswanjono, MT.)

Tanggal : 11 Desember 2006

Pembimbing II,



(Martanto S.T., MT.)

Tanggal : 11 Desember 2006

LEMBAR PENGESAHAN
LAPORAN TUGAS AKHIR

**PAPAN LATIH BULUTANGKIS
BERBASIS MIKROKONTROLER AT89S51**

Di susun oleh :

Agustinus Giri Hartono

NIM : 995114057

NIRM : 990051123107120056

Telah dipertahankan di depan Panitia Penguji
pada tanggal 19 Desember 2006
Dan dinyatakan memenuhi syarat

Susunan Panitia Penguji

Nama Lengkap

Ketua : Ir. Iswanjono, MT.
Sekretaris : Martanto, ST., MT.
Anggota : Ir. Th. Prima Ari Setyani, MT.
Anggota : Wiwien Widyastuti, ST., MT.

Tanda-tangan

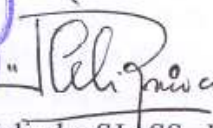


Yogyakarta, 19 Desember 2006

Fakultas Teknik

Universitas Sanata Dharma

Dekan



(Ir. Greg. Heliarko SJ., SS., BST., MA., MSc.)

HALAMAN MOTTO DAN PERSEMBAHAN

**SUNGGUH SUATU HAL YANG LUCU KARENA SEMUA ORANG PUN
TAHU, ADALAH WAJAR JIKA AKU BERHASIL MELAKUKAN APA
YANG AKU DAPAT LAKUKAN , TETAPI JIKA AKU BISA MELAKUKAN
APA YANG TIDAK SEMUA ORANG DAPAT LAKUKAN ITU SEMATA
KARENA DIA ADA DISISIKU
I BELONG TO JESUS
(Giri)**

**“SANADYAN AKU MUNG KODOK, NANGING MENAWA AKU ANA ING
ASTANING DEWI MARIA, SAPA SING WANI MARANG AKU?”
TIDAK ADA DI DUNIA INI YANG KERENDAHAN HATINYA MELEBIHI
KODOK SEBAB KODOK ITUTEKO-TEKO NDHODHOK....
(GP. Sindhunata, SJ)**

**“ ALWAYS WITH ME, ALWAYS WITH YOU ”
(Joe Satriani)**

**DADI WONG KUI OJO RUMANGSA BISA,
NING DADIO WONG SING BISA RUMANGSA
(Semar)**

**“ LEBIH BAIK TERLAMBAT DARI PADA CEPAT TAPI HASILNYA
TIDAK BAIK”
(DEWA BUDJANA)**

**IMPOSSIBLE IS NOTHING ... !
(adidas)**

Dengan segala kerendahan hati dan kejujuran,
secara khusus Tugas Akhir ini kupersembahkan kepada ;
kedua orang tuaku: Bapak karo simbok,
Mas Janus,Mbak Aan,Mas Han, Mas Ipam,Mbak Ima
Mas Budi, Mbak Lin, Mbak Wiwien
Dedariku Krista,Sheila-Sheili,Shera,Sheva
Bulik dan Om-ku, Simbahku,Sepupu-sepupuku, teman-temanku
Tuhan Yesus Kristus Penguasa Alam Semesta Juru Selamatku

PERNYATAAN KEASLIAN KARYA

Saya menyatakan dengan sesungguhnya bahwa karya yang saya tulis ini tidak memuat karya atau bagian karya orang lain, kecuali yang telah disebutkan dalam kutipan pada daftar pustaka, sebagaimana layaknya karya ilmiah.

Yogyakarta, 8 Desember 2006

Penulis,

Agustinus Giri Hartono

PAPAN LATIH BULUTANGKIS BERBASIS MIKROKONTROLER AT89S51

**AGUSTINUS GIRI HARTONO
995114057**

INTI SARI

Pada umumnya pemain bulutangkis berlatih dengan metode *circuit training* secara *manual* dalam durasi yang sudah ditentukan. Secara *manual*, dengan mengikuti instruksi pelatih. Pelatih menunjukan arah kemana pemain harus bergerak. Terdapat 6 titik yang sudah ditentukan di lapangan bulutangkis. Dengan cara yang sama, pelatih menggunakan sebuah alat untuk berlatih. Pelatih menekan satu dari 6 tombol operator untuk menghidupkan salah satu lampu. Pemain harus mengikuti kemana arahan dari lampu yang menunjukkan kepada pemain titik mana yang harus dituju.

Circuit Trainer menggunakan Mikrokontroler AT89S51 sebagai pengendali. *Circuit Trainer* terdiri dari 8 tombol *trainer*, 8 Lampu AC 5 watt, *Speaker*, 2 digit seven segment untuk Penampil Durasi dan Kecepatan, 2 digit seven segment untuk menampilkan skor benar dan skor salah. Alat ini bekerja dengan 9 kombinasi mode. Pemain dapat mengkombinasikan Mode durasi antara 30 detik, 60 detik, atau 90 detik dengan Mode Kecepatan yang terdiri dari 3S, 4S, dan 5S. Mekanisme penyalan lampu AC *trainer* tergantung oleh data pada tabel tengok. Alamat *low byte* tabel tengok diacak dengan menggunakan metode LFSR (*Linear Feedback Shift Register*) 8 bit membentuk sebuah data yang seolah-olah teracak. *Trainer* ini bekerja dengan diawali bunyi *speaker* selama 2 detik dan diakhiri dengan bunyi *speaker* selama 5 detik. Hasil *circuit trainer* akan ditampilkan pada ruas penampil.

Hasil pengamatan menunjukkan alat ini mampu bekerja sebagai pengganti peran pelatih sebagai instruktur sekaligus operator. Proses pengacakan posisi nyala lampu dapat bekerja dengan baik namun tidak terdistribusi secara seragam. *Error* selisih waktu yang terjadi pada mode durasi dapat ditoleransi karena kurang dari 3.0 detik. *Error* untuk mode kecepatan dapat ditoleransi sebab nilainya hanya 0.1 detik, selisih waktu ini tidak dapat dirasakan oleh pemain.

Kata kunci : *Circuit Training*, MCS-51, Aplikasi Mikrokontroler AT89S51,
Pembangkit Angka Acak

BADMINTON CIRCUIT TRAINER BASED ON AT89S51 MICROCONTROLLER

AGUSTINUS GIRI HARTONO
995114057

ABSTRACT

Generally, a badminton player practices by using a circuit training method manually with a setted duration. And manually, the player follows the coach instruction. The coach gives the direction where the player should move. There are six nodes that are set on the badminton pitch. In the same method, the coach uses a tool to do the training. The coach presses one of the six buttons on the operator's buttons to make one of six lamps on. The player should follows the direction of the lamp, that directs the player to the desired node. If the lamp one is on, the player must move to node one and so on.

Circuit Trainer uses AT89S51 microcontroller to drive the system of the trainer. There are 8 trainer buttons, 8 lamps, 1 speaker, 2 digit seven segment used to display the duration trainer and missed score, 2 digit seven segment used to show the trainer speed and the right score. There are 9 combination modes. The player can combine the duration mode between 30 seconds, 60 seconds or 90 seconds with the speed mode 3S, 4S, and 5S. The mechanism that makes one of 8 lamps on depends on the data in the look-up table. The low address of the look-up table will be randomized by using 8 bits LFSR (Linear Feedback Shift Register) method to generate random number. The circuit trainer will be started by the sound of the speaker for 2 seconds, and will be ended with the sound of the speaker for 5 seconds. The result of circuit trainer will be displayed on seven segments.

The result of this study show that, the circuit trainer can do what the coach instructs or the operator does. The random method to make one of the 8 lamps on works but the distribution of the lamp which is on is not the same. The error on the duration mode can be tolerated because the value is less than 3.0 seconds. The error on the speed mode can be tolerated because the value is 0.1 second and the player can not feel it.

Keywords: Circuit Training, Pseudo random, MCS-51, AT89S51
Microcontroller application

KATA PENGANTAR

Puji dan syukur penulis panjatkan kepada Tuhan Yang Maha Esa atas berkat dan rahmat-Nya yang telah diberikan sehingga penulis dapat menyelesaikan Tugas Akhir dengan judul Papan Latih Bulutangkis Berbasis Mikrokontroler AT89S51 (*Badminton Circuit Trainer Based On AT89S51 Microcontroller*). Semoga apa yang telah penulis sampaikan lewat tugas akhir ini dapat memberikan sumbangan pemikiran untuk pengembangan ilmu pengetahuan pada umumnya dan ilmu teknik elektronika pada khususnya.

Tugas Akhir ini merupakan buah dari kerja keras, pemikiran, dan diskusi dengan berbagai sumber yang penulis peroleh sebelumnya di perkuliahan. Namun, penulis menyadari bahwa tugas akhir ini tidak akan selesai jika tidak mendapat bantuan dari banyak pihak yang telah berkenan membantu secara langsung maupun tidak langsung. Penulis juga mengharapkan adanya masukan serta kritik yang membangun dari apa yang telah disampaikan dalam penulisan ini.

Pada kesempatan yang baik ini, dengan segala kerendahan hati penulis mengucapkan banyak terima kasih kepada :

1. Tuhan Yesus Kristus Juru Selamatku, panutan, penjaga, penghibur, guruku, temanku, terima kasih atas nafas yang selalu dapat kuhirup. Walau Kau tak pernah dapat kulihat, tetapi ketika aku jatuh, senang, bahagia Kau dapat kurasakan.

2. Bapak Ir. Iswanjono, MT., selaku pembimbing I, yang telah memberikan bantuan ide, saran, masukan, kritik, serta bimbingannya yang sangat berguna selama penulisan Tugas Akhir ini.
3. Bapak Martanto, ST., MT., selaku pembimbing II sekaligus pembimbing akademik, yang telah memberikan arahan, semangat, dorongan, serta sumbangan pemikiran yang penulis butuhkan untuk menyelesaikan penulisan Tugas Akhir ini.
4. Bapak Augustinus Bayu Primawan, ST., M.Eng., selaku Kepala Jurusan Teknik Elektro Fakultas Teknik Universitas Sanata Dharma Yogyakarta.
5. Ibu Ir. Th. Prima Ari Setyani, MT., selaku Sekretaris Jurusan Teknik Elektro Fakultas Teknik Universitas Sanata Dharma Yogyakarta.
6. Para dosen-dosen Teknik Elektro, yang telah memberikan pengajaran serta pemikiran yang telah diberikan selama penulis berada dalam masa perkuliahan. Pak Petrus, Pak Damar, Pak Tjendro, Bu Wiwien, Bu Wuri, Pak Thomas, Pak Bambang, Pak Linggo, Pak Djoko, Pak Pius.
7. Para karyawan laboratorium Teknik Elektro, yaitu Lik Mardi, Lik Broto, Pakde Suryana, Pak Hardi, yang telah membantu penulis dalam menyelesaikan pembuatan alat yang dibutuhkan Tugas Akhir.
8. Bapak Aris Sukardjito, S.IP, selaku staf Sekretariat Teknik Elektro, yang telah membantu penulis untuk menyiapkan syarat-syarat akademik maupun administrasi yang dibutuhkan selama perkuliahan.

9. Bapak dan Simbok. Bapakku yang sedang sakit PY. Kimin, Ibuku Theresia Sumijati tercinta terima kasih untuk kesetiaannya dan kesediaannya menjaga bapak, sehingga penulis dapat menyelesaikan Tugas Akhir ini.
10. Kakak-kakak-ku Mbak Aan, Mas Han, Mas Ipam, Mbak Ima ,Mas Janus, Mbak Lin, Mas Budi, Mbak Wiwien, Bulik Rum,Om Di, Bulik Ninik, Om Yu, Eron, Dik Iim, Nina, Yuli terima kasih atas dukungannya. Maaf terlalu lama menunggu...!
11. Krista, Sheila-Sheili, Shera, Sheva Dedari-dedariku.
12. ‘Metalicca’ Boim ST, Bagus ‘Brahmana dari Bali’ S.T, ‘Nuno Extreme’ Adrex ST, Si Boss ST ‘Hammerfall’, Ki Gendheng ‘Mbah Jiwo’ Pamungkas ST , si ‘Om GiGi’ Pam ST, ‘ Stingky’ Ableh 07 ST , Si Bro Norick ‘Albe’ ST, Joe Satriani, Kaka, Pirlo, Paolo Maldini, AC Milan-ku, Dewa Budjana, GiGi, Mr.Big. Frater ju-Aprri ‘ SJ’ ST, Wawan ‘misa pagi’ ST, Ian ‘otak yahudi’ Kadir ST, Ari ‘Gijil’, Yuyun ‘Blantik Sapi’ ST, Oscar ST ‘Edan’, Leo Noya’99 ST, Agung ‘Agleng Tbk.’ ST Apache, Rahmad99 ‘orcad’ ST, Dinus ‘Meihong’ ST,sobat-sobatku 99 yang baru berjuang. Alkices Skuad, Pak’e, Bu Gendut, dan seluruh Mabes Minggiran ; SiBund, Mba Datik, Budhe, Lui+Eki. Terima kasih atas dukungan yang selama ini penulis dapatkan dari kalian yang tersebut di atas!
13. Sahabat-sahabatku: Pakde Danur, Om Hari, Bulik Denok, Bulik Ana, Yusan cah bagus , Dik Ikul teman curhatku ‘ojo salah golek dalam’

tengkiu so peri much, Indik, Brabat, Dik Tantri, Bambang, Peyek, Bowo, Robert, Klunthung, Kenuk, Ning ‘teman sms’, Rino Inzaghi, Putra, Marco ‘Gattuso’, Bolo Kurowo Klarangan, Anas bebek ‘buat masukannya, tengkiu nas!’, Upik, Yanti, Bruder Parjo, Romo Teguh, Mas Haryono P.R, Komunitas Mudika Agatha, Mudika Bernadinus Candi, ,Mudika Paroki Maria Assumpta Pakem, Mudika Rayon Sleman dan Yogyakarta. ‘Sudah Kulewati’.

14. Komunitas_CBC: Gendut ‘matur tengkiu sampun pareng ngampil printeripun’, Aryo Bimo ‘Rekayasa’, Kasino buat pinjaman sepatu dasinya, Gelung ‘pasangan doubleku’, Nyiman, Antiq buat ilmu badmintonnya, Genjik Striker kakon ,Maman, Heman, Sugeng; Imam, Agung, Yayur, Diah, Anak buahku di skuad Sinar Muda FC junior. Teman-temanku De Britto di seantero Bumi. Komunitas Angkringan terima kasih untuk dukungannya.

Penulis juga ingin menyampaikan banyak terima kasih kepada pihak-pihak yang tidak dapat penulis sebutkan satu per satu. Karena tanpa bantuan mereka, penulis menyadari tidak akan mampu menyelesaikan Tugas akhir ini dengan baik.

Yogyakarta, 8 Desember 2006

Penulis

DAFTAR ISI

JUDUL	i
LEMBAR PERSETUJUAN	iii
LEMBAR PENGESAHAN	iv
MOTTO DAN PERSEMBAHAN	v
PERNYATAAN KEASLIAN KARYA	vi
INTISARI	vii
ABSTRACT	viii
KATA PENGANTAR	ix
DAFTAR ISI	xiii
DAFTAR GAMBAR	xvii
DAFTAR TABEL	xx
BAB I PENDAHULUAN	1
1.1 Judul	1
1.2 Latar Belakang	1
1.3 Perumusan Masalah	2
1.4 Batasan Masalah	3
1.5 Tujuan Penelitian	3
1.6 Manfaat Penelitian	4
1.7 Sistematika Penulisan	4
BAB II DASAR TEORI	6
2.1. Mikrokontroler AT89S51	6

2.1.1. Organisasi Memori Mikrokontroler AT89S51	6
2.1.1.1. Memori Program	7
2.1.1.2. Memori Data	7
2.1.2. Register Fungsi Khusus	8
2.1.3. <i>Timer /Counter</i>	10
2.1.3.1. Register TCON	11
2.1.3.2. Register TMOD	12
2.1.3.3. Mode <i>Timer/Counter</i>	13
2.1.4. Port Masukan/ Keluaran (<i>I/O Port 0</i>)	15
2.1.5. Sistem Interupsi Mikrokontroler AT89S51	17
2.1.5.1. Pengaktifan Interupsi	17
2.1.5.2. Vektor Interupsi	18
2.2. LED (<i>Light Emitting Diode</i>)	19
2.3. Transistor Sebagai Penggerak <i>Speaker</i>	20
2.4. <i>Triac</i>	22
2.4.1. Penentuan Tahanan <i>Gate Triac</i>	23
2.5. <i>Optocoupler</i>	23
2.6. Penampil <i>Seven Segment</i> (<i>Seven Segment Display</i>)	25
2.7. <i>Saklar / Tombol N.O. SPST Momentary Contact</i>	27
2.8. <i>Pseudo Random</i>	27
BAB III PERANCANGAN	31
3.1. Perancangan Perangkat Keras	31
3.1.1. Konfigurasi Mikrokontroler AT89S51	34

3.1.1.1. Rangkaian Reset	34
3.1.1.2. Rangkaian Osilator	35
3.1.2. Rangkaian 8 Tombol <i>Trainer</i>	36
3.1.3. <i>Interfacing Triac</i>	38
3.1.4. Rangkaian Mode <i>Trainer</i>	40
3.1.4.1. Tombol Pemilih Mode Kecepatan <i>Trainer</i>	40
3.1.4.2. Tombol Pemilih Mode Durasi <i>Trainer</i>	42
3.1.5. Rangkaian Penampil <i>Seven Segment</i>	42
3.1.5.1. LED Penampil <i>Seven Segment</i>	43
3.1.5.2. Resistor Pembatas Arus <i>Seven Segment</i>	44
3.1.5.3. Transistor Penggerak <i>Seven Segment</i>	45
3.1.6. Rangkaian Penggerak <i>Speaker</i>	47
3.2. Perancangan Perangkat Lunak	50
3.2.1. Inisialisasi Awal	52
3.2.2. Pemilih Mode	55
3.2.3. Routine Start_main	58
3.2.4. <i>Trainer</i>	61
3.2.5. Evaluasi	63
3.2.6. Pembacaan Tombol <i>Trainer</i>	65
3.2.7. Proses <i>Random</i>	66
3.2.8. Proses Nyala Lampu	67
3.2.9. Data <i>Seven Segment</i>	69
BAB IV PENGAMATAN DAN ANALISA	70

4.1.	Cara Kerja Alat Secara Umum	70
4.2.	Pengamatan dan Analisa	72
4.2.1.	Pengamatan Tampilan Awal	72
4.2.2.	Pengamatan Penekanan Tombol Mode	73
4.2.3.	Penekanan Tombol <i>Start</i>	74
4.2.4.	Pengamatan Waktu Durasi	75
4.2.5.	Pengamatan Waktu Kecepatan	79
4.2.6.	Pengamatan <i>Tone Speaker</i> Mulai 2 Detik	82
4.2.7.	Pengamatan <i>Tone Speaker</i> Selesai 5 Detik	82
4.2.8.	Pengamatan Frekuensi Bunyi <i>Speaker</i>	83
4.2.9.	Pengamatan <i>Trainer</i> Bekerja Pada Mode 3S/30 detik	84
4.2.10.	Pengamatan <i>Trainer</i> Bekerja Pada Mode 3S/60 detik	85
4.2.11.	Pengamatan <i>Trainer</i> Bekerja Pada Mode 3S/90 detik	86
4.2.12.	Pengamatan Penekanan Tombol <i>Trainer</i>	86
4.2.13.	Pengamatan Penyalaan Acak Lampu <i>Trainer</i>	87
4.2.14.	Pengamatan Akhir <i>Trainer</i>	92
4.2.14.	Pengamatan Penggunaan <i>Trainer</i>	93
BAB V KESIMPULAN DAN SARAN		97
5.1.	Kesimpulan	97
5.2.	Saran	98
DAFTAR PUSTAKA		100
LAMPIRAN		

DAFTAR GAMBAR

Gambar 2.1.	Diagram Blok Papan Latih Bulutangkis	5
Gambar 2.2.	Memori Data dan Memori Program Mikrokontroler AT89S51	6
Gambar 2.3.	Peta Memori Data	8
Gambar 2.4.	Register Program Status Word	9
Gambar 2.5.	Register AUXR1	10
Gambar 2.6.	Register TCON	11
Gambar 2.7.	Register TMOD	12
Gambar 2.8.	Register IE	18
Gambar 2.9.	Rangkaian LED (<i>Light Emitting Diode</i>)	19
Gambar 2.10.	Rangkaian Penggerak <i>Speaker</i>	20
Gambar 2.11.	Satu Periode Gelombang Kotak	22
Gambar 2.12.	Rangkaian Dasar <i>Triac</i>	22
Gambar 2.13.	<i>Optocoupler</i> Menggunakan <i>LED</i> dan <i>Fotodioda</i>	24
Gambar 2.14.	<i>LED Seven Segment</i>	25
Gambar 2.15.	Susunan <i>Segment LED</i>	26
Gambar 2.16.	Tombol N.O. <i>SPST Momentary Contact</i>	27
Gambar 2.17.	LFSR <i>n bit</i>	28
Gambar 2.18.	LFSR <i>8 bit</i>	29
Gambar 3.1.	Lapangan Bulutangkis Ukuran Standar Internasional	31

Gambar 3.2.	Implementasi Rancangan	32
Gambar 3.3.	Penyangga Lampu	33
Gambar 3.4.	Diagram Blok <i>Badminton Circuit Trainer</i>	33
Gambar 3.5.	Rangkaian <i>Reset</i> Mikrokontroler <i>AT89S51</i>	34
Gambar 3.6.	Rangkaian Osilator	35
Gambar 3.7.	Rangkaian 8 Tombol <i>Trainer</i>	37
Gambar 3.8.	<i>Interfacing Triac</i> dengan Mikrokontroler <i>AT89S51</i>	38
Gambar 3.9.	Rangkaian Penampil Skor	42
Gambar 3.10.	Rangkaian Penampil Waktu	43
Gambar 3.11.	Rangkaian Ekuivalen Sebuah <i>Seven Segment</i>	45
Gambar 3.12.	Rangkaian Penggerak Sebuah <i>Seven Segment</i>	46
Gambar 3.13.	Rangkaian Penggerak <i>Speaker</i>	47
Gambar 3.14.	Diagram Alir Perangkat Lunak <i>Circuit Trainer</i>	52
Gambar 3.15.	Diagram Alir Inisialisasi_awal	54
Gambar 3.16.	Diagram Alir Transfer Data <i>Seven Segment</i> ke <i>RAM</i>	55
Gambar 3.17.	Diagram Alir Pemilihan Mode	57
Gambar 3.18.	Diagram Alir Start Main	59
Gambar 3.19.	Gelombang Kotak dengan Frekuensi 1KHz	60
Gambar 3.20.	Diagram Alir <i>Trainer</i>	61
Gambar 3.21.	Diagram Alir Evaluasi	64
Gambar 3.22.	Diagram Alir Pembacaan Tombol	66
Gambar 3.23.	Diagram Alir <i>Random</i>	67
Gambar 4.1.	Durasi 30 detik	78

Gambar 4.2.	Durasi 60 detik	79
Gambar 4.3.	Durasi 90 detik	79
Gambar 4.4.	Kecepatan 3 detik	80
Gambar 4.5.	Kecepatan 4 detik	80
Gambar 4.6.	Kecepatan 5 detik	80
Gambar 4.7.	<i>Speaker</i> Berbunyi Selama 2 detik	82
Gambar 4.8.	<i>Speaker</i> Berbunyi Selama 5 detik	83
Gambar 4.9.	Hasil Pengamatan Periode <i>Speaker</i>	84
Gambar 4.10.	Periode <i>Trainer</i> bekerja pada Mode 3S/30 detik	84
Gambar 4.11.	Periode <i>Trainer</i> bekerja pada Mode 3S/60 detik	85
Gambar 4.12.	Periode <i>Trainer</i> bekerja pada Mode 3S/90 detik	86
Gambar 4.13.	Grafik Pengamatan Distribusi Nyala Lampu Mode 3S/90	91
Gambar 4.14.	Grafik Pengamatan Distribusi Nyala Lampu Mode 4S/90	91
Gambar 4.15.	Grafik Pengamatan Distribusi Nyala Lampu Mode 5S/90	92
Gambar 4.16.	Pemetaan Pergerakan Pemain dengan Cara <i>Manual</i>	93

DAFTAR TABEL

Tabel 2.1.	Peta Memori SFR.	8
Tabel 2.2.	Keterangan Register TCON	12
Tabel 2.3.	Keterangan Register TMOD	13
Tabel 2.4.	Keterangan Kombinasi M0 dan M1 Register TMOD	13
Tabel 2.5.	Fungsi Alternatif Port 1	16
Tabel 2.6.	Fungsi Alternatif Port 3	16
Tabel 2.7.	Register IE	18
Tabel 2.8.	Daftar Vektor Interupsi	18
Tabel 2.9.	<i>Triac</i> yang Tersedia Secara Komersial	23
Tabel 2.10.	Karakter yang Dihasilkan <i>Seven Segment</i>	26
Tabel 2.11.	LFSR (<i>Linier Feedback Shift Register</i>)	28
Tabel 3.1.	Tabel Data Masukan Tombol	37
Tabel 3.2.	Karakteristik <i>Triac</i> Q4004L4	39
Tabel 3.3.	Tabel Data Keluaran <i>Port</i> 0 untuk Penyalan LED	44
Tabel 3.4.	Inisialisasi Variabel	53
Tabel 4.1.	Bagian-bagian Alat Tampak dari Luar	72
Tabel 4.2.	Data Pengamatan Kondisi Tampilan Awal	73
Tabel 4.3.	Data Pengamatan Penekanan Tombol Mode	73
Tabel 4.4.	Data Pengamatan Penekanan Kombinasi Mode	74
Tabel 4.5.	Data Pengamatan Proses Penekanan Tombol “START”	75
Tabel 4.6.	Data Pengamatan Waktu Durasi 30	75

Tabel 4.7.	Data Pengamatan Waktu Durasi 60	76
Tabel 4.8.	Data Pengamatan Waktu Durasi 90	76
Tabel 4.9.	Data Pengamatan Mode 3S / 30 detik	87
Tabel 4.10.	Data Pengamatan Mode 3S / 60 detik	88
Tabel 4.11.	Data Pengamatan Mode 3S / 90 detik	88
Tabel 4.12.	Data Pengamatan Mode 4S / 30 detik	88
Tabel 4.13.	Data Pengamatan Mode 4S / 60 detik	89
Tabel 4.14.	Data Pengamatan Mode 4S / 90 detik	89
Tabel 4.15.	Data Pengamatan Mode 5S / 30 detik	89
Tabel 4.16.	Data Pengamatan Mode 5S / 60 detik	90
Tabel 4.17.	Data Pengamatan Mode 5S / 90 detik	90
Tabel 4.18.	Data Pengamatan Akhir Selesai <i>Trainer</i>	92
Tabel 4.19.	Perbandingan antara Kedua Alat, Secara Manual dan Secara Otomatis	94

BAB I

PENDAHULUAN

1.1. Judul

Papan Latih Bulutangkis Berbasis Mikrokontroler AT89S51
(*Badminton Circuit Trainer Based On AT89S51 Microcontroller*).

1.2. Latar Belakang

Di dalam olahraga bulutangkis terdapat sebuah metode *trainer* yang dalam dunia bulutangkis sering disebut *circuit training*. Metode ini berfungsi untuk melatih ketahanan, reflek, kecepatan, dan melatih pemain untuk memainkan tempo permainan. Berawal dari *hobby* penulis bermain bulutangkis dan pernah berlatih dengan metode ini maka dipilihlah judul “*Badminton Circuit Trainer Based On AT89S51 Microcontroller*”.

Pada awalnya untuk melakukan latihan ini digunakan sebuah sistem dengan beberapa bagian. Bagian tersebut meliputi 6 buah lampu AC, tiang untuk penyangga lampu AC, operator, tombol operator, pewaktu *manual* dengan jam digital eksternal (*stopwatch*), dan bel tanda. Metode *trainer* ini adalah mengikuti gerakan nyala salah satu lampu yang dikendalikan oleh operator melalui tombol operator, kemudian pemain bergerak ke seluruh penjuru lapangan dan berlatih selama waktu yang sudah ditentukan.

Dengan melihat sistem kerja dan bagian-bagian dari alat tersebut, terdapat beberapa kekurangan dan ketidakpraktisan. Dengan sistem tersebut

tidak dapat dipastikan apakah pemain sudah bergerak ke arah salah satu lampu yang menyala atau belum dan tidak terdapatnya sebuah informasi bahwa pemain telah menjalankan metode *trainer* dengan benar serta mengetahui hasil latihan yang sudah dijalankan. Sisi ketidakpraktisan lain adalah diperlukannya operator untuk menjalankan alat, sehingga pemain tidak dapat berlatih secara mandiri.

Dengan mikrokontroler beberapa bagian dari alat itu yang tidak terintegrasi seperti *timer*, bel tanda dapat diintegrasikan kedalam alat. *Timer* sendiri dapat dirancang dengan mikrokontroler. Mikrokontroler juga dapat mengendalikan *speaker* untuk membangkitkan suara. Kelebihan yang lain ialah tidak diperlukannya operator untuk mengendalikan nyala lampu AC, jumlah lampu AC yang akan digunakan juga menjadi 8 buah lampu AC dan memberikan fasilitas tombol masukan kepada pemain. Untuk mengetahui tingkat keberhasilan maka dapat ditambahkan papan evaluasi skor kedalam sistem alat *circuit trainer* tersebut.

1.3. Perumusan Masalah

Pokok permasalahan dalam penelitian ini adalah bagaimana menggantikan peran operator. Kemudian peran dari operator dalam mengendalikan kerja dari *trainer* akan digantikan dengan mikrokontroler AT89S51, mikrokontroler ini akan mengendalikan nyala salah satu lampu dari ke 8 lampu AC secara acak, dan mengolah data masukan dari saklar tombol masukan untuk kemudian hasil *trainer* ditampilkan melalui media papan evaluasi skor.

1.4. Batasan Masalah

Batasan yang ditentukan pada perancangan *Circuit Trainer* adalah:

1. Komponen pengolah yang digunakan dalam perancangan adalah 1 buah mikrokontroler AT89S51.
2. *Trainer* terdiri dari 8 buah tombol dan 8 buah lampu AC 5 watt, 1 buah tombol mode waktu (lambat: 5 detik, sedang: 4 detik, cepat: 3 detik), 1 buah tombol mode lama waktu *trainer* (30 detik, 60 detik, 90 detik), 2 buah *seven segment* sebagai penampil skor, 2 buah *seven segment* sebagai penampil waktu, tombol '*START*' , tiang penyangga lampu dan *speaker*.
3. Lapangan bulutangkis menggunakan ukuran lapangan bulutangkis standar internasional 2x6,7 meter, tinggi tiang net 1,55 meter dan lebar 6,1 meter.

1.5. Tujuan Penelitian

Secara umum tujuan pembuatan tugas akhir ini adalah :

1. Mengimplementasikan dan memanfaatkan mikrokontroler AT89S51 dan teknik antar muka dengan komponen lain dalam perancangan sebagai *circuit trainer* bulutangkis.
2. Membuat *circuit trainer* yang bermanfaat untuk membantu seorang pemain bulutangkis untuk berlatih secara mandiri.

1.6. Manfaat Penelitian

Manfaat perancangan *Circuit Trainer* ini adalah membantu pemain untuk berlatih mandiri agar kualitas permainannya berkembang. Dengan alat ini diharapkan kecepatan gerak reflek dalam merespon pergerakan bola akan meningkat secara bertahap, meningkatnya ketahanan fisik, dan melatih pemain untuk memainkan tempo.

1.7. Sistematika Penulisan

Penulisan laporan penelitian tugas akhir ini disusun dengan sistematika sebagai berikut :

BAB I Pendahuluan

Bab ini meliputi: latar belakang, tujuan penelitian, batasan masalah, manfaat penelitian dan sistematika penulisan.

BAB II Dasar Teori

Menguraikan teori dasar rangkaian elektronika dan mikrokontroler AT89S51 yang digunakan dalam rangkaian.

BAB III Perancangan

Berisi tentang perancangan alat baik *hardware* maupun *software*.

BAB IV Pengamatan dan Analisa

Bab ini berisi pembahasan hasil pengamatan pada perangkat keras, perangkat lunak.

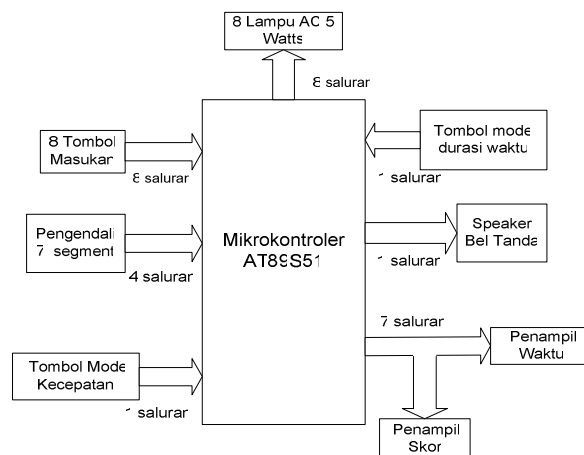
BAB V Kesimpulan dan Saran

BAB II

DASAR TEORI

Pada bab ini akan dijelaskan mengenai teori dasar komponen dan rangkaian, termasuk juga teori dasar yang digunakan untuk merancang perangkat lunak sebagai pembentuk sistem ‘Papan Latih Bulutangkis Berbasis Mikrokontroler AT89S51’. Pembentuk sistem terdiri dari: mikrokontroler AT89S51, rangkaian LED (*Light Emitting Diode*), pengendali lampu AC menggunakan *triac*, pengendali *speaker* menggunakan transistor, penampil 7’segment dan penyaklarannya menggunakan transistor, saklar dan pembangkit angka acak dengan menggunakan metode LFSR (*Linear Feedback Shift Register*) 8 bit.

Secara umum diagram blok Papan Latih Bulutangkis Berbasis Mikrokontroler AT89S51 dapat terlihat pada gambar 2.1.



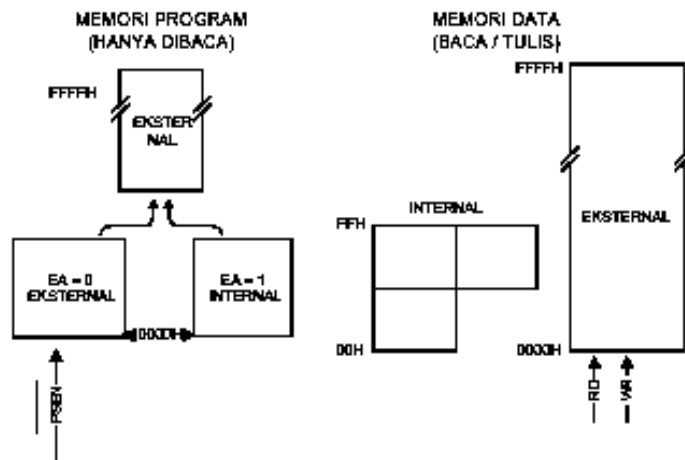
Gambar 2.1. Diagram blok Papan Latih Bulutangkis

2.1. Mikrokontroler AT89S51

Mikrokontroler AT89S51 adalah mikrokontroler CMOS 8-bit yang berkemampuan tinggi dengan 4 Kbytes *in-system programmable Flash Memory*. AT89S51 ini dibuat dengan teknologi Atmel, memori *non-volatile* dan sesuai dengan standar industri *pinout* dan instruksi set MCS 51. AT89S51 yang dipakai memiliki fitur: 4KB *In-System Programmable Flash*, 128 Bytes RAM, 32 jalur I/O, dua 16-bit *timers / counters*, *Watchdog Timer*, 2 *data pointer*, 5 vektor dua level interupsi, serial port *full duplex*, osilator *on-chip* dan *clock circuitry*.

2.1.1 Organisasi Memori AT89S51

Mikrokontroler AT89S51 memiliki ruang alamat untuk memori program dan memori data yang terpisah, seperti terlihat pada gambar 2.2 . Setiap memori program dan memori data eksternal dapat dialamati hingga 64 Kbytes.



Gambar 2.2. Memori Data dan Memori Program Mikrokontroler AT89S51

Pemisahan program dan data memori ini memungkinkan pengaksesan data memori dengan pengalamatan 8 bit, sehingga dapat langsung disimpan dan

dimanipulasi oleh mikrokontroler dengan kapasitas akses 8 bit. Namun demikian, untuk pengaksesan data memori dengan alamat 16 bit, harus dilakukan dengan menggunakan register DPTR (*Data Pointer*). Program memori hanya dapat dibaca saja (diletakkan pada ROM/ EPROM). Untuk membaca program memori eksternal, mikrokontroler akan mengirim sinyal PSEN (*Program Store Enable*). Sebagai data memori eksternal dapat digunakan RAM eksternal (maksimum 64 Kbyte). Dalam pengaksesannya mikrokontroler akan mengirimkan sinyal RD (*Read*, melakukan operasi pembacaan data) dan WR (*Write*, melakukan operasi penulisan data).

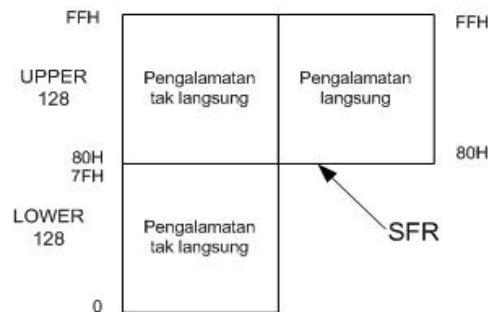
2.1.1.1 Memori Program

Memori program atau sering disebut dengan *flash memory* pada mikrokontroler AT89S51 memiliki kapasitas sebesar 4KB yang hanya bisa dibaca saja. Bila pin \overline{EA} dihubungkan pada *ground* program memori dapat diakses secara eksternal, bila pin \overline{EA} dihubungkan pada V_{CC} memori program 4KB dapat diakses langsung pada alamat 0000H-0FFFFH secara internal dan pada alamat 1000H-FFFFH secara eksternal.

2.1.1.2. Memori Data

Memori data menggunakan memori jenis RAM. RAM merupakan memori yang dapat dibaca dan ditulis. RAM dipakai sebagai penyimpan data pada saat program bekerja. Isi RAM akan hilang bila catu daya mati (*Volatile Memory*).

Mikrokontroler AT89S51 memiliki memori data 256 *bytes* dan dapat diakses secara pengalamatan langsung dan pengalamatan tidak langsung. Pengoperasian *stack* adalah contoh dari pengalamatan tidak langsung, jadi 128 *bytes* RAM data tersedia sebagai ruang *stack*. Peta memori data dapat dilihat pada Gambar 2.3.



Gambar 2.3. Peta Memori Data

2.1.2 Register Fungsi Khusus (*Special Function Register*)

Peta dari memori *on-chip* disebut dengan ruang register fungsi khusus (*Special Function Register*) yang diperlihatkan pada Tabel 2.1.

Tabel 2.1. Peta memori SFR

0F8H								0FFH
0F0H	B 00000000							0F7H
0EBH								0EFH
0E0H	ACC 00000000							0E7H
0D8H								0DFH
0D0H	PSW 00000000							0D7H
0C8H								0CFH
0C0H								0C7H
0B8H	IP XX000000							0BFH
0B0H	P3 11111111							0B7H
0A8H	IE 0X000000							0AFH
0A0H	P2 11111111		AUXR1 XXXXXXXX0				WDTRST XXXXXXXXX	0A7H
98H	SCON 00000000	SBUF XXXXXXXXX						9FH
90H	P1 11111111							97H
88H	TCN 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000	AUXR XX00XX0	8FH
80H	P0 11111111	SP 00000000	DP0L 00000000	DP0H 00000000	DP1L 00000000	DP1H 00000000	PCON 0XXX0000	87H

Akumulator

ACC atau akumulator yang menempati lokasi E0H digunakan sebagai register untuk penyimpanan data sementara, dalam program.

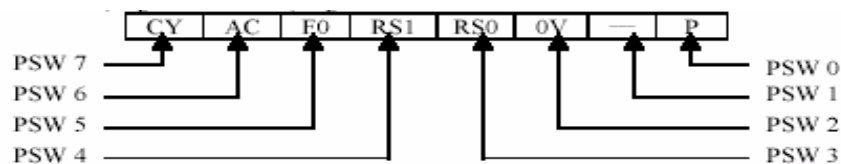
Register B

Register B (lokasi F0) digunakan selama operasi perkalian dan pembagian, untuk intruksi lain dapat diperlakukan sebagai register *scratch pad* (papan coret-core).

Program Status Word (PSW)

Register PSW (lokasi D0H) mengandung informasi status program seperti terlihat pada Gambar 2.4. Fungsi bit pada PSW sebagai berikut:

CY (PSW7)	: <i>carry</i> setelah operasi aritmatika.
AC (PSW6)	: <i>auxiliary carry</i> setelah operasi aritmatika.
F0 (PSW5)	: <i>flag</i> untuk fungsi umum.
RS0 (PSW4)	: untuk memilih bank register.
RS1 (PSW3)	: untuk memilih bank register.
OV (PSW2)	: <i>overflow</i> setelah operasi aritmatika.
(-) (PSW1)	: <i>user definable flag</i>
P (PSW0)	: <i>Parity Flag (P)</i> , merupakan bit even-parity dari akumulator. Jika banyaknya angka 1 dalam Akumulator ganjil, maka P akan di-set menjadi 1; sebaliknya P akan di-clear menjadi 0.



Gambar 2.4. Register Program Status Word

Stack Pointer

Register *Stack Pointer* (lokasi 81H) merupakan register dengan panjang 8-bit, digunakan dalam proses simpan dan ambil dari atau ke *stack*.

Data Pointer

Register *Data Pointer* mengandung DPTR untuk *byte* tinggi (DPH) dan *byte* rendah (DPL). Pada AT89S51 memiliki 2 buah DPTR untuk memudahkan pengaksesan baik internal maupun eksternal, yaitu DP0 di lokasi 82H-83H dan DP1 di lokasi 84H-85H. Untuk menggunakannya harus menginisialisasi bit DPS pada register AUXR1 (lokasi A2H). Bila DPS = 0, maka memilih register DPTR DP0L - DP0H dan bila DPS = 1, maka memilih register DPTR DP1L - DP1H. Register AUXR1 dapat dilihat pada Gambar 2.5.

—	—	—	—	—	—	—	DPS
7	6	5	4	3	2	1	0

Gambar 2.5. Register AUXR1

Kontrol Register

Register-register IP, IE, TMOD, dan TCON berisi bit-bit kontrol dan status untuk sistem interupsi, pencacah / pewaktu dan serial *port* .

2.1.3. Timer / Counter

Mikrokontroler AT89S51 mempunyai dua buah register *timer/counter* 16 bit , *Timer* 0 dan *Timer* 1. Pada saat sebagai *Timer*, register naik satu (*increment*) setiap satu *cycle*. Jika digunakan osilator 12 Mhz, maka satu *cycle* sama dengan

$1/12$ frekuensi osilator = $1\mu s$. Pada saat sebagai counter, register naik satu (*increment*) pada saat transisi 1 ke 0 dari input eksternal, T0 atau T1.

Apabila periode tertentu telah dilampaui, *timer/counter* segera menginterupsi mikrokontroler untuk memberitahukan bahwa perhitungan periode waktu telah selesai dilaksanakan. Periode waktu *timer/counter* secara umum ditentukan oleh persamaan berikut:

a. Sebagai T/C 8 bit

$$T = (255 - TLx) * 1\mu s \quad \dots\dots\dots(2.1)$$

Dimana TLx adalah isi register TL0 atau TL1.

b. Sebagai T/C 16 bit

$$T = (65535 - THx TLx) * 1\mu s \quad \dots\dots\dots(2.2)$$

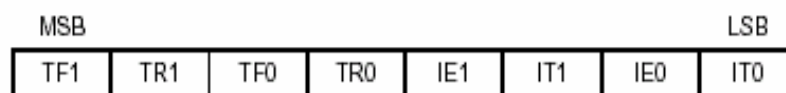
THx = isi register TH0 atau TH1

TLx = isi register TL0 atau TL1

2.1.3.1. Register TCON

Pengontrol kerja *timer/counter* ada pada register *timer control* (TCON).

Adapun definisi dari bit-bit pada *timer control* adalah sebagai berikut:



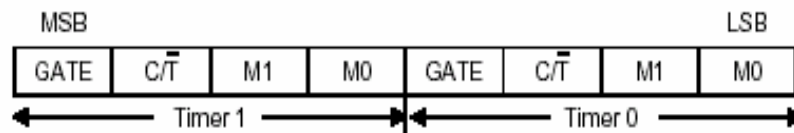
Gambar 2.6. Register TCON

Tabel 2.2. Keterangan register TCON

Simbol	Fungsi
TF1	<i>Timer 1 overflow flag. Di set oleh hardware pada saat timer/counter overflow. Di clear oleh hardware pada saat menjalankan rutin interupsi.</i>
TR1	<i>Timer 1 Run control bit. Di set/clear oleh software. Digunakan untuk mengaktifkan/nonaktifkan timer/counter</i>
TF0	<i>Timer 0 overflow flag. Di set oleh hardware pada saat timer/counter overflow. Di clear oleh hardware pada saat menjalankan rutin interupsi.</i>
TR0	<i>Timer 0 Run control bit. Di set/clear oleh software. Digunakan untuk mengaktifkan / nonaktifkan timer/counter</i>
IE1	<i>Interrupt 1 Edge flag. Di set oleh hardware ketika interupsi eksternal mendeteksi adanya edge. Di clear ketika proses interupsi</i>
IT1	<i>Interrupt 1 Type control bit. Di set / clear oleh software untuk menentukan pen-trigger-an interupsi eksternal pada transisi turun / low level</i>
IE0	<i>Interrupt 0 Edge flag. Di set oleh hardware ketika interupsi eksternal mendeteksi adanya edge. Di clear ketika proses interupsi</i>
IR0	<i>Interrupt 0 Type control bit. Di set/clear oleh perangkat lunak untuk menentukan pen-trigger-an interupsi eksternal pada transisi turun / low level</i>

2.1.3.2. Register TMOD

Pengontrol pemilihan mode operasi *timer/counter* ada pada register *timer mode* (TMOD) . Definisi bit-bitnya adalah sebagai berikut:

**Gambar 2.7. Register TMOD**

Tabel 2.3. Keterangan Register TMOD

Simbol	Fungsi
GATE	<i>Gate control set. Timer/counter 'x' akan aktif jika pin "INT" high dan kondisi pin "TRx" sedang set. Gate control clear. Timer"x" akan aktif jika "TRx" set</i>
C / T	<i>Selector timer/counter. Clear untuk mode timer (input dari internal clock) dan set untuk mode counter (input dari pin "Tx")</i>
M1	<i>Bit untuk memilih mode timer/counter</i>
M0	<i>Bit untuk memilih mode timer/counter</i>

x = 0 atau 1.

Tabel 2.4. Keterangan Kombinasi M0 dan M1 Register TMOD

M1	M0	Mode	Operasi
0	0	0	<i>Timer 13 bit</i>
0	1	1	<i>Timer/counter 16 bit</i>
1	0	2	<i>Timer/counter pengisian otomatis (auto reload) 8 bit</i>
1	1	3	<i>(Timer 0) TL0 adalah T/C 8 bit yang dikontrol oleh kontrol bit standar timer 0. TH0 adalah timer 8 bit dan dikontrol oleh kontrol bit timer 1 (Timer 1) Timer/counter 1 tidak aktif</i>

2.1.3.3. Mode *Timer/Counter*

Terdapat empat jenis mode yang dapat ditentukan melalui pengisian register TMOD yang dibahas pada bagian 2.1.3.3, yaitu:

a. Mode 0

Pada mode ini timer bekerja sebagai timer 13 bit yang terdiri dari counter 8-bit dengan pembagi 32 (pembagi 5 bit). Setelah perhitungan selesai, mikrokontroler akan mengeset *Timer Interrupt Flag* (TF1). Dengan membuat

GATE = 1, timer dapat dikontrol oleh *input* dari luar (INT1), untuk fasilitas pengukuran lebar pulsa. Register 13 bit yang digunakan terdiri dari 8 bit dari TH1 dan 5 bit bawah dari TL1 (bit 6,7,8 tidak digunakan). Mengeset TR1 tidak akan menghapus isi register. Operasi pada mode 0 untuk *Timer 0* dan *Timer 1* adalah sama.

b. Mode 1

Mode 1 sama dengan mode 0 kecuali register *timer* akan bekerja dalam mode 16 bit.

c. Mode 2

Mode 2 menyusun register *timer* sebagai 8 bit *counter* (TL1) dengan kemampuan pengisian otomatis. *Overflow* dari TL1 tidak hanya men-set TF1 tetapi juga mengisi TL1 dengan isi TH1 yang diisi sebelumnya oleh *software*. Pengisian ulang ini tidak mengubah nilai TH1.

d. Mode 3

Dalam operasi mode 3 *timer 1* akan berhenti, hitungan yang sedang berjalan dipegang. Efeknya sama seperti mengatur TR1 = 0. Timer 0 dalam mode 3 membuat TL0 dan TH0 sebagai dua *counter* terpisah. TL0 menggunakan kontrol bit *timer 0* yaitu C/T, GATE, TR0, INT0 dan TF0.. TH0 berfungsi hanya sebagai *timer* dan mengambil alih penggunaan TR1 dan TF1 dari *timer 1* dan sekarang TH0 mengontrol interupsi *timer 1*. Mode 3 diperlukan untuk aplikasi yang membutuhkan ekstra *timer/counter* 8 bit. Dengan *timer 0* dalam mode 3, mikrokontroler AT89S51 seperti memiliki 3 *timer/counter*. Saat *timer 0* dalam

mode 3, *timer* 1 dapat dihidupkan atau dimatikan, atau dapat digunakan oleh *port* serial sebagai pembangkit *baud rate* dalam aplikasi komunikasi serial.

2.1.4. Port masukan/keluaran (I/O port)

Sama seperti keluarga MCS-51 lainnya mikrokontroler AT89S51 memiliki 4 *port* masukan/keluaran (I/O *port*) yang diberi nama port 0, port 1, port 2 dan port 3. Setiap *port* selain sebagai jalur masuk atau keluar data, juga memiliki karakteristik masing-masing.

a. Port 0

Port 0 merupakan *port* keluaran/masukan (I/O) bertipe *open drain bidirectional*. *Port* 0 juga dapat dikonfigurasi sebagai bus alamat/ data bagian rendah selama proses pengaksesan memori data dan program eksternal. *Port* ini berada di alamat 80H SFR.

b. Port 1

Port 1 merupakan *port* I/O dwiarah yang dilengkapi dengan *pull-up* internal. Jika '1' dituliskan ke kaki-kaki *port* 1, masing-masing kaki akan di *pull high* dengan *pull up* internal sehingga dapat digunakan sebagai masukan. *Port* 1 berada di alamat 90H juga menerima alamat bagian rendah (*low bit*) selama pemrograman dan verifikasi *flash*. Selain sebagai piranti I/O, *port* 1 mempunyai fungsi yang lain seperti terlihat pada Tabel 2.5.

Tabel 2.5. Fungsi Alternatif Port 1

Pin Port	Fungsi Alternatif
P1.5	MOSI (digunakan untuk <i>In-System Programming</i>)
P1.6	MISO (digunakan untuk <i>In-System Programming</i>)
P1.7	SCK (digunakan untuk <i>In-System Programming</i>)

c. Port 2

Port 2 berada di alamat A0H dan memiliki karakteristik yang mirip dengan *port 1*. *Port 2* akan memberikan byte alamat bagian tinggi selama pengambilan instruksi dari memori program eksternal dan selama pengaksesan memori data eksternal yang menggunakan perintah dengan alamat 16-bit (misalnya: `MOVX @DPTR`). *Port* ini juga menerima alamat bagian tinggi selama pemrograman dan verifikasi *flash*.

d. Port 3

Port 3 terletak di alamat B0H. Selain berfungsi untuk menerima sinyal-sinyal kontrol untuk pemrograman dan verifikasi *flash*, dapat juga digunakan untuk fungsi-fungsi yang lain seperti terlihat pada Tabel 2.6.

Tabel 2.6. Fungsi Alternatif Port 3

Pin Port	Fungsi Alternatif
P3.0	RXD (masukan <i>port</i> serial)
P3.1	TXD (keluaran <i>port</i> serial)
P3.2	$\overline{INT0}$ (interupsi 0 eksternal)
P3.3	$\overline{INT1}$ (interupsi 1 eksternal)
P3.4	T0 (<i>input</i> eksternal timer 0)
P3.5	T1 (<i>input</i> eksternal timer 1)
P3.6	\overline{WR} (memori data eksternal jalur tulis)
P3.7	\overline{RD} (memori data eksternal jalur baca)

2.1.5. Sistem Interupsi Mikrokontroler AT89S51

Interupsi merupakan suatu sarana dalam mikrokontroler yang sangat berperan dalam penanganan sistem input output. Dalam proses interupsi, terjadinya sesuatu pada perangkat keras akan dicatat pada flip-flop yang sering disebut petanda (*flag*). Catatan dalam petanda tersebut diatur sedemikian rupa sehingga merupakan sinyal permintaan interupsi pada prosesor.

Program yang dijalankan dengan cara tersebut dinamakan sebagai program pelayanan interupsi (ISR - *Interrupt Service Routine*). Saat prosesor menjalankan ISR, pekerjaan yang sedang dilakukan dalam program utama ditinggalkan sementara, selesai menjalankan ISR program utama kembali dijalankan.

AT89S51 memiliki 5 sumber interupsi, yaitu: 2 interupsi pewaktu yaitu interupsi yang terjadi pada saat timer mengalami *overflow* dari semua bit 1 ke semua bit 0, atau bit TF=1. 2 interupsi eksternal yaitu interupsi yang terjadi karena adanya sinyal interupsi yang berasal dari luar, dan sebuah interupsi *port* serial yaitu interupsi yang terjadi pada saat diterimanya/ dikirimnya sinyal serial antara 2 mikrokontroler.

2.1.5.1. Pengaktifan Interupsi

Masing-masing sumber interupsi dapat diaktifkan dan dimatikan secara individual dengan menset bit EA dalam register IE (lihat gambar 2.8).



Gambar 2.8. Register IE

Penjelasan simbol pada register IE dapat dilihat pada Tabel 2.7.

Tabel 2.7. Register IE

Simbol	Posisi	Fungsi
EA	IE.7	Untuk mengaktifkan (IE =1) dan menon-aktifkan (IE =0)
—	IE.6	Cadangan
—	IE.5	Cadangan
ES	IE.4	Bit aktivasi interupsi <i>Port Serial</i>
ET1	IE.3	Bit aktivasi interupsi <i>Timer 1</i>
EX1	IE.2	Bit aktivasi interupsi Eksternal 1
ET0	IE.1	Bit aktivasi interupsi <i>Timer 0</i>
EX0	IE.0	Bit aktivasi interupsi Eksternal 0

2.1.5.2. Vektor Interupsi

Saat suatu interupsi diterima, nilai yang disimpan ke PC sebagai alamat RLI selanjutnya disebut sebagai vektor interupsi, yang sekaligus merupakan awal alamat RLI yang bersangkutan. Daftar vektor interupsi terdapat pada Tabel 2.8.

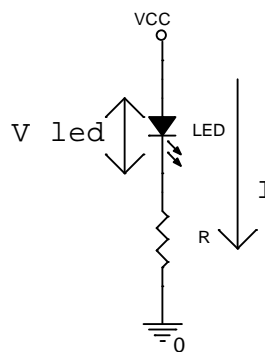
Tabel 2.8. Daftar Vektor Interupsi

Interupsi	Tanda (Flag)	Alamat Vektor
<i>Reset sistem</i>	RST	0000H
Eksternal 0	IE0	0003H
<i>Timer 0</i>	TF0	000BH
Eksternal 1	IE1	0013H
<i>Timer 1</i>	TF1	001BH
<i>Port Serial</i>	RI atau TI	0023H

2.2. LED (*Light Emitting Diode*)

LED (*Light Emitting Diode*) merupakan salah satu komponen elektronika, yaitu *diode*, yang dapat memancarkan cahaya. LED terdiri dari unsur-unsur seperti *Gallium*, *Arsen* dan *Fosfor* yang dapat memancarkan cahaya berwarna merah, kuning, hijau, biru, putih, jingga.

Gambar 2.9 menunjukkan rangkaian skematik LED dan hambatan pembatas arus.



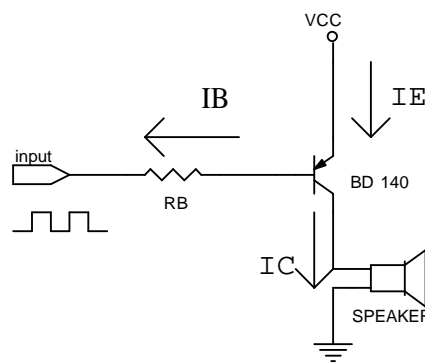
Gambar 2.9. Rangkaian LED (*Light Emitting Diode*)

Hal terpenting dalam penggunaan LED adalah arus agar dapat memancarkan cahaya. Biasanya masing-masing LED membutuhkan tegangan yang berbeda untuk mengalirkan arus 20 mA agar intensitas cahaya yang dihasilkan 100 %. Arus yang mengalir pada LED menentukan intensitas cahaya yang dihasilkan sesuai dengan *datasheet*. Dibutuhkan juga sebuah hambatan sebagai pembatas arus untuk menjaga kemungkinan LED rusak, karena LED mempunyai resistansi dalam yang kecil. Penentuan nilai hambatan pada gambar 2.9. adalah :

$$R = \frac{V_{CC} - V_{LED}}{I} \dots\dots\dots(2.3)$$

2.3. Transistor Sebagai Penggerak *Speaker*

Speaker merupakan komponen yang digunakan untuk menghasilkan bunyi. Frekuensi bunyi *speaker* dapat diatur dengan menggunakan fasilitas *timer* pada mikrokontroler AT89S51 dengan penggunaan *delay*. Pengaturan kondisi *ON* dan *OFF speaker* diatur dengan transistor penggerak *speaker*. Rangkaian penggerak *speaker* menggunakan transistor PNP BD 140 pada gambar 2.10.



Gambar 2.10. Rangkaian Penggerak *Speaker*

Transistor BD 140 berjenis PNP sehingga untuk mengaktifkan transistor diperlukan arus basis yang mengalir dari Vcc ke kaki basis. Transistor dalam keadaan *ON* karena adanya arus *LOW* pada basis dengan pemberian logika 0 pada basis transistor, *speaker* mendapatkan tegangan Vcc. Sebaliknya jika transistor dalam keadaan *OFF* karena maka *speaker* juga dalam keadaan *OFF*. Jika kondisi *ON* dan *OFF* dilakukan secara periodik maka akan diperoleh gelombang kotak dengan frekuensi atau periode tertentu.

Speaker membutuhkan arus (I_{speaker}) agar bekerja (aktif) yang nilainya dapat ditentukan dengan spesifikasi daya (P) dan hambatan (R) pada *speaker*. Nilai arus tersebut adalah:

$$I_{spea\ ker} = \sqrt{\frac{P}{R}} \quad (\text{Ampere}) \quad \dots\dots\dots (2.4)$$

Nilai hambatan basis (R_B) dapat ditentukan dengan menghitung nilai arus basis (I_B) dengan arus kolektor (I_C) dan penguatan arus transistor (h_{FE}), yang besarnya:

$$I_B = \frac{I_C}{h_{FE}} \quad \dots\dots\dots (2.5)$$

Arus speaker ($I_{speaker}$) pada gambar 2.11 sama dengan arus I_C transistor, sehingga nilai arus basis adalah:

$$I_B = \frac{I_{spea\ ker}}{h_{FE}} \quad \dots\dots\dots (2.6)$$

Nilai hambatan basis dengan mensubstitusi persamaan 2.6 adalah:

$$R_B = \frac{V_{CC} - V_{EB}}{I_B} \quad \text{atau} \quad R_B = \frac{V_{CC} - V_{EB}}{I_{spea\ ker} / h_{FE}} \quad \dots\dots\dots (2.7)$$

Frekuensi dapat dibangkitkan oleh mikrokontroler dengan pengaturan tunda waktunya melalui siklus mesin instruksi perangkat lunak mikrokontroler.

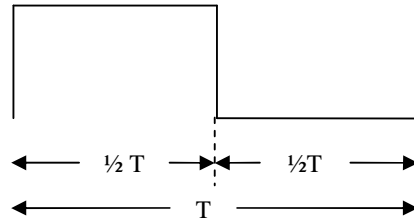
Perhitungan tunda waktu untuk pembangkit frekuensi *speaker* adalah:

$$f = \frac{1}{T} \quad \text{atau} \quad T = \frac{1}{f} \quad \dots\dots\dots (2.8)$$

T = periode bunyi *speaker* (detik)

f = frekuensi bunyi *speaker* (Hertz)

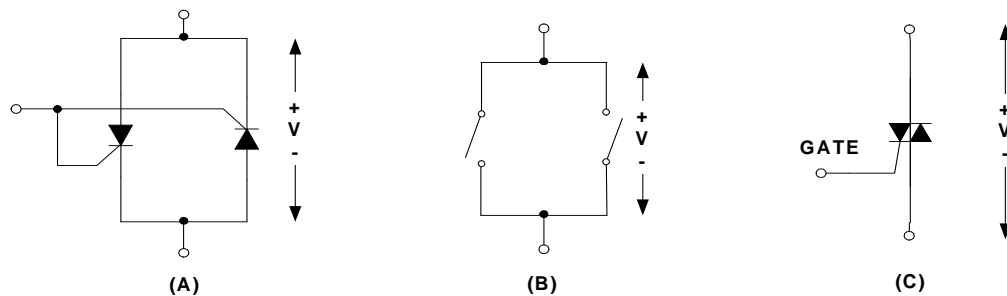
Speaker akan *ON* dan *OFF* dalam satu periode agar menghasilkan bunyi, sehingga diperlukan tunda waktu sebesar $\frac{1}{2} T$ dalam keadaan *ON* dan *OFF* sehingga menghasilkan gelombang kotak seperti pada gambar 2.11.



Gambar 2.11. Satu Periode Gelombang Kotak

2.4. *Triac*

Triac merupakan sebuah *thyristor* dua arah, yang sangat ideal untuk menghantarkan arus bolak-balik. *Triac* dapat bertindak sebagai *SCR* yang terpasang paralel, seperti dalam gambar 2.12.(A) yang ekuivalen dengan dua buah saklar, seperti pada gambar 2.12.(B). Oleh karena itu, *triac* dapat mengendalikan arus pada kedua arah. Tegangan penyalan biasanya tinggi, sehingga cara yang normal untuk menyalakan *triac* adalah dengan menerapkan pemicu berprategangan maju. Lembaran data untuk *triac* selalu mencantumkan tegangan pemicu dan arus pemicu ini.



Gambar 2.12. Rangkaian Dasar *Triac*

Apabila tegangan V mempunyai polaritas seperti yang tampak pada gambar 2.12.(A), maka harus diterapkan pemicu positif, yang akan menutup saklar di sebelah kiri. Bila tegangan V mempunyai polaritas yang berlawanan dibutuhkan pemicu negatif yang akan menutup saklar di sebelah kanan. Gambar 2.12.(C) merupakan lambang skematik dari sebuah *triac*.

Tabel 2.9. *Triac* yang Tersedia Secara Komersial

Seri Piranti	V_{GT} (V)	I_{GT} (mA)	I_{max} (A)	V_{max} (V)
Q201E3	2	10	1	200
Q4004L4	2,5	25	4	400
Q5010R5	2,5	50	10	500
Q6015R5	2,5	50	15	600

2.4.1. Penentuan Tahanan Gate *Triac*

Untuk menentukan R_{gate} dapat dihitung dengan rumus :

$$V_{gate} = I_{gate} \cdot R_{gate}$$

$$R_{gate} = \frac{V_{gate}}{I_{gate}} \dots\dots\dots(2.9)$$

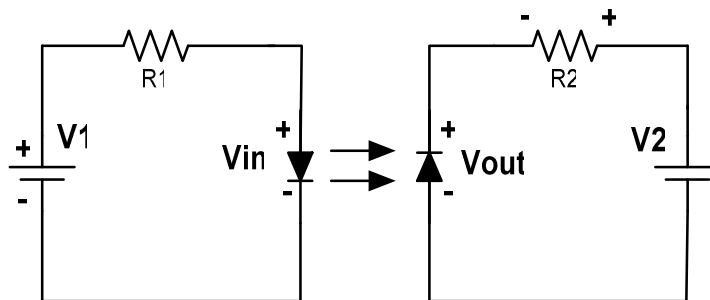
2.5. *Optocoupler*

Sebuah *optocoupler* (juga disebut optoisolator) menggabungkan LED dan *fotodioda* dalam satu kemasan. Pada *optocoupler* terdapat pada sisi *input* dan *fotodioda* pada sisi *output*. Sumber tegangan sebelah kiri dan resistor seri mengatur arus melalui LED. Kemudian cahaya dari LED *fotodioda*, dan akan mengatur arus balik pada rangkaian *output*. Arus balik ini akan menghasilkan

tegangan jepit pada resistor *output*. Tegangan *output* kemudian sama dengan *output* tegangan penyedia daya dikurangi tegangan pada resistor.

Saat tegangan *input* berubah, jumlah cahaya berubah-ubah. Ini berarti bahwa tegangan *output* berubah bersama-sama dengan tegangan *input*. Hal inilah yang menyebabkan kombinasi LED dan *fotodioda* disebut dengan *optocoupler*. Alat ini dapat menghubungkan isyarat *input* dengan rangkaian *output*.

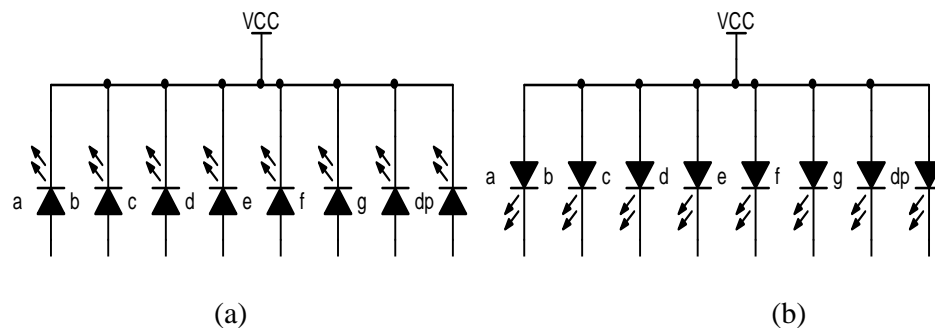
Keuntungan pokok *optocoupler* adalah terjadinya isolasi elektrik antara rangkaian *input* dan *output*. Dengan *optocoupler*, hanya terdapat kontak *input* dan *output* dalam bentuk pancaran sinar. Oleh karena itu, dimungkinkan untuk mengisolasi resistansi antara dua rangkaian dalam orde ribuan *megaohm*. Isolasi yang seperti ini berguna dalam aplikasi tegangan tinggi antara beda potensial dua rangkaian yang mencapai ribuan volt. Gambar 2.13 menunjukkan *optocoupler*.



Gambar 2.13. *Optocoupler* Menggunakan LED dan *Fotodioda*

2.6. Penampil Seven Segment (*Seven Segment Display*)

Penampil *seven segment* adalah salah satu penampil yang biasa digunakan untuk menampilkan karakter angka 0 sampai 9. *Seven segment* terdiri dari 7 buah LED yang disatukan pada kutub anoda (*Common Anode*) atau katodanya (*Common Cathode*). Ketujuh LED pada *seven segment* diberi label a sampai g. Gambar *seven segment* dapat dilihat pada gambar 2.14 :



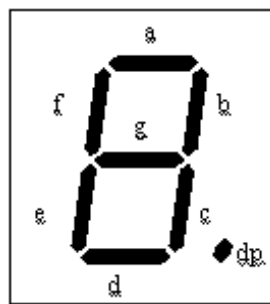
Gambar 2.14. LED Seven Segment
(a) *Common Cathode* (b) *Common Anode*

Seven segment memiliki karakteristik yang sama dengan LED, yaitu: konsumsi arus yang cukup kecil dan sangat peka terhadap polaritasnya (anoda dan katoda), sehingga perlu diperhatikan jenis yang digunakan agar tidak terjadi *bias* mundur yang menyebabkan LED rusak.

LED pada *seven segment* hanya dapat menerima tegangan sekitar 1,7 Volt pada terminal-terminalnya dan arus per *segment* sebesar 5 mA. Untuk mencegah arus yang berlebih, digunakan hambatan sebagai pengaman. Nilai hambatan yang digunakan ditentukan dengan rumus :

$$R = \frac{V_{cc} - V_{LED}}{I_{led}} \dots\dots\dots(2.10)$$

Untuk menampilkan angka diperlukan arus yang melewati masing-masing *segment* yang diinginkan untuk aktif. Gambar 2.15 memperlihatkan susunan LED sehingga dapat menampilkan karakter angka desimal (0 sampai 9) dan huruf-huruf tertentu bila *segment* itu aktif.



Gambar 2.15. Susunan Segment LED

Karakter yang dapat dihasilkan dari nyala *segment* LED yang aktif ditunjukkan pada Tabel 2.10:

Tabel 2.10. Karakter yang Dihasilkan Seven Segment

Karakter	Segment yang aktif
0	a b c d e f
1	b c
2	a b d e g
3	a b c d g
4	b c f g
5	a c d f g
6	a c d e f g
7	a b c
8	a b c d e f g
9	a b c d f g

2.7. Saklar / Tombol N.O. SPST *Momentary Contact*

Tombol N.O. SPST (*Normally Open, Single Pole, Single Throw*) *momentary contact* adalah salah satu jenis saklar yang dalam keadaan normal berkeadaan *OFF* (*normally open*), berupa satu kutub (berasal dari satu sumber) dan menghantarkan arus hanya ke satu beban. Penghantaran arus dan tegangan (kondisi *ON*) dari suatu sumber terjadi jika tombol ditekan, dan pemutusan arus dan tegangan (kondisi *OFF*) terjadi saat tombol dilepas atau tidak di tekan, sehingga disebut sebagai saklar *momentary contact*. Skematik tombol N.O. SPST *momentary contact* pada gambar 2.16 :



Gambar 2.16. Tombol N.O. SPST *Momentary Contact*

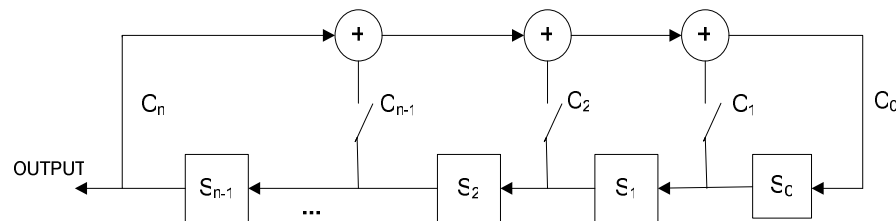
2.8. *Pseudo Random*

Salah satu metode yang dapat digunakan untuk mengimplementasikan pembangkit angka acak (*random number generator*) adalah *Linear Feedback Shift Register* (LFSR). Pembangkit kode dengan penggunaan LFSR sebenarnya merupakan '*pseudo-random*' sebab pada suatu periode angka yang diacak dapat berulang. Salah satu cara mengatasinya adalah dengan penggunaan pergeseran bit yang lebih panjang, sehingga perulangan pola terjadi dengan waktu yang lebih lama. Operasi dasar dari LFSR adalah penggunaan pergeseran bit dan operasi XOR dari tap (*carry bit* setelah pergeseran) yang telah ditentukan seperti

ditunjukkan pada gambar 2.17. Tabel 2.11 adalah tabel jumlah urutan maksimum dan posisi tap *feedback* (umpan balik) dari LFSR sesuai jumlah bit.

Tabel 2.11. LFSR (*Linear Feedback Shift Register*)

Jumlah bit	Panjang urutan	Posisi tap
2	3	[0,1]
3	7	[0,2]
4	15	[0,3]
5	31	[1,4]
6	63	[0,5]
7	127	[0,6]
8	255	[1,2,3,7]
9	511	[3,8]
10	1023	[2,9]



Gambar 2.17. LFSR n bit

C_0, C_1, \dots, C_{n-1} merupakan tap *feedback*, jika $C = 0$, tidak terhubung; $C = 1$, terhubung. Persamaan karakteristik *polynomial* LFSR n bit adalah:

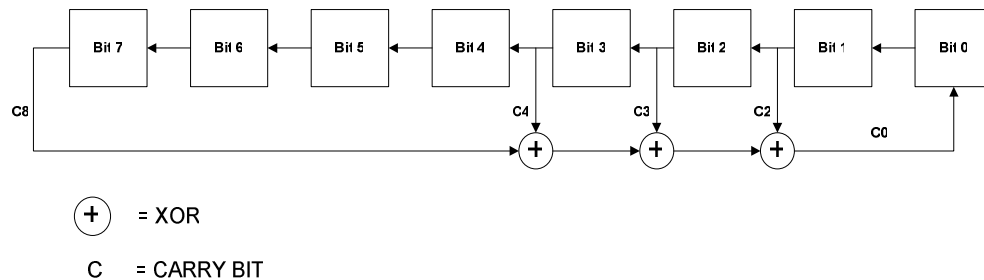
$$C(x) = C_0 + C_1x + C_2x^2 + \dots + C_{n-1}x^{n-1} + C_n x^n \dots\dots\dots(2.11)$$

Panjang maksimum (L) urutan dari suatu LFSR n-bit adalah :

$$L = 2^n - 1 \dots\dots\dots(2.12)$$

Dari tabel 2.11, LFSR dengan jumlah bit 8 memiliki panjang urutan 255, artinya LFSR 8 bit menghasilkan 255 angka acak. Posisi tap umpan balik untuk LFSR 8 bit terletak pada posisi bit 1, 2, 3 dan 7; keempat bit ini di-XOR-kan dan

hasilnya akan menjadi bit 0 (LSB). Gambar pengoperasian LFSR 8 bit ditunjukkan pada gambar 2.18.



Gambar 2.18. LFSR 8 Bit

Bentuk persamaan *polynomial* LFSR 8 bit dari gambar 2.18, dapat ditentukan berdasarkan persamaan 2.13 ,yaitu:

$$C(x) = 1 + x^2 + x^3 + x^4 + x^8 \quad \dots\dots\dots(2.13)$$

Persyaratan utama dari penggunaan LFSR adalah pengisian angka pertama sebelum operasi adalah bukan nol, karena akan mengakibatkan perulangan angka nol (angka tidak teracak).

Angka acak yang dihasilkan adalah pengacak alamat/lokasi memori yang menyimpan 256 data penyalan lampu yang ada pada tabel *look-up* dan tersimpan dalam ROM AT89S51. Mikrokontroler AT89S51 lebih mudah dalam pengoperasian 8 bit, sehingga dipilih metode pengacakan 8 bit. Hasil angka acak berupa 8 bit dan akan ditambahkan dengan alamat awal data penyalan lampu dan menghasilkan 16 bit penunjuk alamat.

Data penyalan lampu yang ada pada tabel *look-up* berupa 8 bit dan dirancang hanya untuk menyalakan salah satu lampu, yaitu : 1111 1110 (FEH), 1111 1101 (FDH), 1111 1011 (FBH), 1111 0111 (F7H), 1110 1111 (EFH), 1101 1111 (DFH), 1011 1111 (BFH), 0111 1111 (7FH). Data penyalan lampu ini

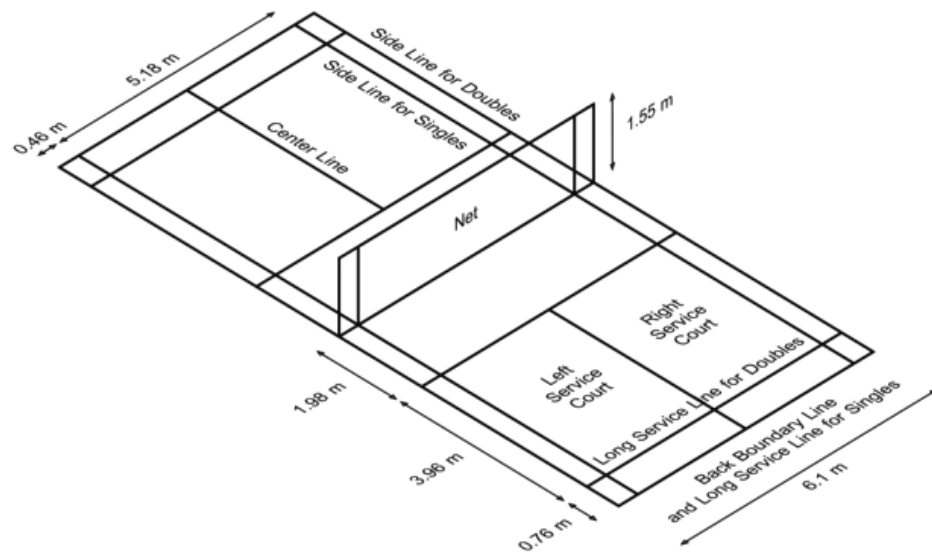
akan diacak juga pada 255 lokasi alamat memori untuk menghasilkan nyala salah satu Lampu yang bervariasi.

BAB III

PERANCANGAN

3.1. Perancangan Perangkat Keras

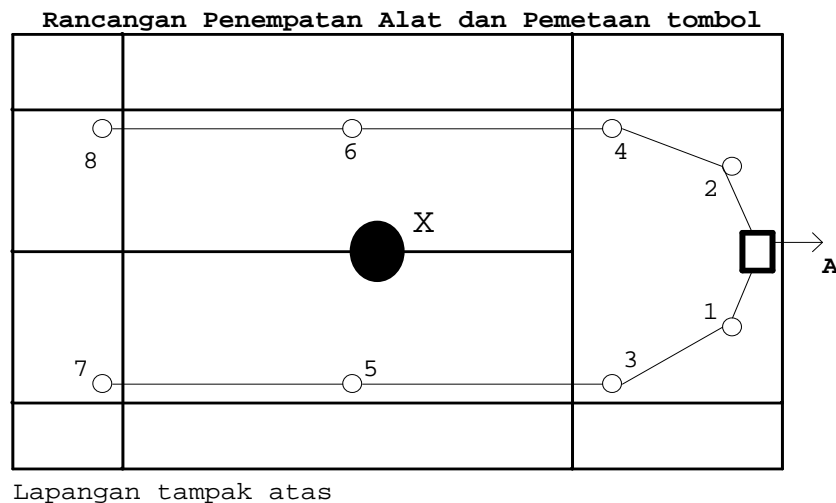
Sistem *trainer* ini secara nyata terdiri dari 8 buah tombol *trainer*, 8 buah lampu AC 5 watts, 1 buah *speaker*, blok alat, penampil skor, penampil waktu, tombol mode, dan tiang penyangga. Kedelapan bagian tersebut akan diletakan diatas sebuah lapangan bulutangkis dengan ukuran standar internasional. Gambar 3.1 adalah sebuah lapangan bulutangkis standar internasional dengan panjang 2x 6,7 meter, tinggi tiang *net* 1,55 meter dan lebar 6,1 meter.



Gambar 3.1. Lapangan Bulutangkis Ukuran Standar Internasional

Sistem *trainer* ini dapat dijelaskan sebagai berikut. Delapan tombol *trainer* masing-masing akan diletakan di setiap titik lapangan yang sudah

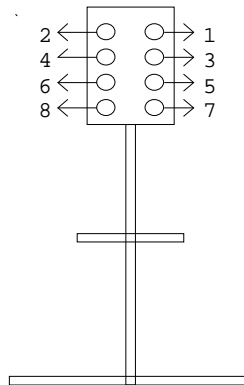
ditentukan sebelumnya ditunjukkan pada Gambar 3.2. Delapan Lampu AC akan berada di depan, tepatnya didekat net dengan posisi berada di atas net dengan sebuah penyangga tiang setinggi 1,65 meter. Mekanisme pelatihan adalah, ketika salah satu lampu menyala maka pemain akan mengikuti gerakan dan menekan tombol yang ditentukan sebagai pasangan dari lampu tersebut. Jika lampu nomor 5 menyala maka pemain harus lari ke tombol dengan nomor 5 pula. Demikian juga ketika lampu 2 menyala, maka pemain harus lari dan menekan tombol 2, dan seterusnya. Kesalahan akan muncul bila tombol yang ditekan tidak sesuai dengan pasangannya atau tidak ditekan sama sekali.



Gambar 3.2 Implementasi Rancangan

Tanda O dengan penomorannya pada Gambar 3.2 adalah letak tombol *trainer*, tanda X dengan lingkaran oval berwarna hitam adalah posisi awal pemain, A adalah posisi tiang dengan 8 lampu serta letak blok alat. Untuk perkabelan tombol, kabel akan diletakkan di tepi garis bagian dalam membentuk huruf U. Untuk tombol 7, tombol 5, tombol 3 dan tombol 1 memiliki *ground*

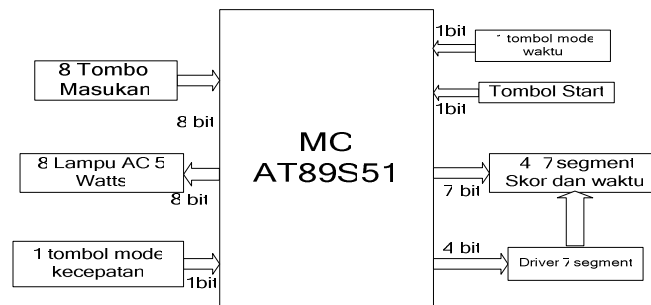
yang sama pada 1 titik, demikian juga tombol 8 sampai tombol 2. Jika bagian kutub *ground* pada tombol saling terhubung, maka untuk bagian kutub tombol yang lain masing-masing akan menjadi masukan untuk pin mikrokontroler.



Gambar 3.3. Penyangga Lampu

Gambar 3.3 diatas adalah rancangan tiang penyangga dengan tinggi 1,6 meter. Lingkaran kecil di dalam kotak adalah ke 8 lampu, dengan disertai nomornya.

Diagram Blok *Badminton Circuit Trainer* yang akan dirancang dapat dilihat pada gambar 3.4.



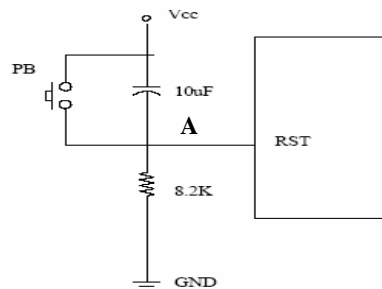
Gambar 3.4. Diagram Blok *Badminton Circuit Trainer*

3.1.1. Konfigurasi Mikrokontroler AT89S51

Konfigurasi mikrokontroler AT89S51 dijelaskan pada sub bab berikut.

3.1.1.1. Rangkaian Reset

Pada saat catu daya diaktifkan, kapasitor $10\mu\text{F}$ akan terhubung singkat. Arus mengalir dari V_{cc} langsung ke kaki RST, sehingga kaki RST memperoleh masukan berlogika 1. Kapasitor $10\mu\text{F}$ akan terisi hingga tegangan pada kapasitor yaitu tegangan antara V_{cc} dan titik A mencapai V_{cc} , otomatis tegangan pada resistor atau tegangan kaki RST akan turun menjadi 0, sehingga kaki RST memperoleh masukan logika 0 dan proses *reset* selesai. Skematik rangkaian *reset* ditunjukkan pada gambar 3.5 :



Gambar 3.5. Rangkaian *Reset* Mikrokontroler AT89S51

Saat saklar PB ditekan, *reset* bekerja secara *manual*, tegangan pada resistor $8,2\text{ K}\Omega$ akan sama dengan V_{cc} , sehingga kaki RST memperoleh masukan berlogika 1. Saat saklar dilepas, aliran arus ke resistor $8,2\text{ K}\Omega$ akan terhenti dan tegangan pada kaki RST akan turun menjadi 0, sehingga logika pada kaki ini berubah menjadi 0 dan proses *reset* selesai. Setelah kondisi pin RST kembali *LOW*, mikrokontroler akan mulai menjalankan program dari alamat 0000H. Kondisi *internal RAM* tidak terjadi perubahan selama *reset*.

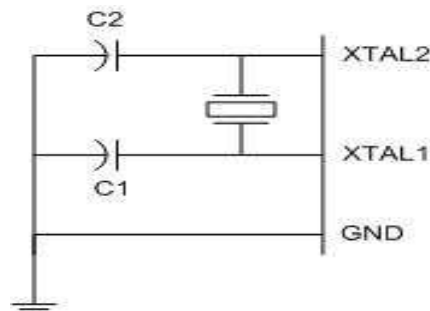
Pada perancangan digunakan osilator kristal 12 MHz dan satu siklus mesin dikerjakan dalam 12 periode osilator, maka satu siklus mesin dikerjakan selama :

$$T_{\text{cycle}} = \frac{12}{f_{\text{osc}}} = \frac{12}{12\text{MHz}} = 1\mu\text{d}$$

Reset terjadi saat adanya logika 1 selama 2 siklus mesin pada kaki 9 (RST), sehingga untuk reset membutuhkan waktu selama $2 \times 1\mu\text{d} = 2\mu\text{d}$.

3.1.1.2. Rangkaian Osilator

Kedua mikrokontroler AT89S51 masing-masing membutuhkan rangkaian osilator. Rangkaian osilator untuk mikrokontroler AT89S51 ditunjukkan pada gambar 3.6 .



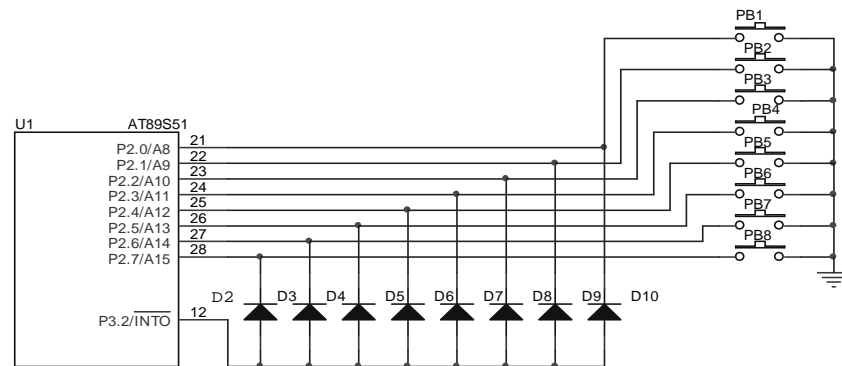
Gambar 3.6. Rangkaian Osilator

Osilator *on-chip* digunakan sebagai sumber detak (*clock*) ke CPU. Pada perancangan, digunakan sebuah resonator kristal 12 Mhz di antara kaki-kaki XTAL1 dan XTAL2 serta 2 buah kapasitor C1 dan C2 sebesar 30pF ke *ground*. Penggunaan kristal dengan frekuensi 12 MHz mengakibatkan satu siklus mesin mikrokontroler dikerjakan dalam 12 periode osilator atau 1 μs .

$$T_{\text{cycle}} = \frac{12}{f_{\text{osc}}} = \frac{12}{12\text{MHz}} = 1\mu\text{s}$$

3.1.2. Rangkaian 8 Tombol *Trainer*

Rangkaian tombol terdiri dari 8 saklar tekan yang bersifat *momentary*. Saat tombol ditekan maka kondisinya terhubung dan sebaliknya kondisinya terputus saat tombol dilepas atau tidak ditekan. Kedelapan tombol tersebut dihubungkan menjadi satu pertanahan (*common ground*). Masing-masing tombol terhubung ke *port* 2 mikrokontroler AT89S51 yang terdiri dari 8 pin (P2.0 sampai P2.7). Konfigurasi tombol seperti gambar 3.7 memungkinkan masukan salah satu tombol difungsikan sebagai interupsi . Jika salah satu tombol ditekan, kemudian P3.2 akan terhubung ke *ground*. Setelah interupsi terdeteksi lewat P3.2, data tombol *trainer* akan dibaca oleh mikrokontroler lewat P2 dan dibandingkan dengan data lampu AC yang menyala. Fungsi dioda adalah ketika salah satu tombol ditekan, P3.2 juga akan menjadi *low* dan terhubung ke *ground*. Posisi saklar tekan diatur di ke delapan titik sudut lapangan bulutangkis dengan urutan saklar 1 di posisi kiri atas sampai saklar 8 di posisi kanan bawah. Gambar rangkaian 8 tombol *trainer* dapat dilihat pada gambar 3.7.



Gambar 3.7. Rangkaian 8 tombol *Trainer*

Pada saat sistem dihubungkan dengan catu daya, isi dari register *port 2* mikrokontroler mula-mula adalah FFH atau *port 2* berada dalam keadaan logika 1. Bila salah satu tombol ditekan, maka salah satu pin pada *port 2* terhubung dengan *ground* dan mendapatkan masukan dengan logika 0. Tabel 3.1 adalah tabel data masukan ke *port 2* mikrokontroler bila ada salah satu tombol yang ditekan (0 = posisi saklar tertutup, 1 = posisi saklar terbuka) :

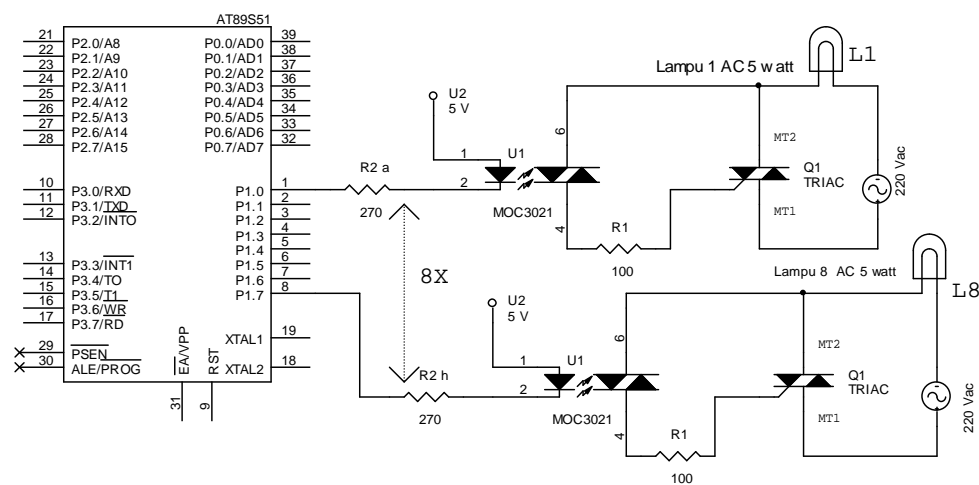
Tabel 3.1. Tabel Data Masukan Tombol

Tombol	Koneksi	Data biner	Data Heksadesimal
1	P2.0	1111 1110	FE
2	P2.1	1111 1101	FD
3	P2.2	1111 1011	FB
4	P2.3	1111 0111	F7
5	P2.4	1110 1111	EF
6	P2.5	1101 1111	DF
7	P2.6	1011 1111	BF
8	P2.7	0111 1111	7F

3.1.3. Interfacing Triac

Dibentuk oleh komponen R1, MOC3021, R2_{a-h}, *triac*, dan lampu AC 10 watt / 220 volt sebagai beban. Bagian ini langsung berhubungan dengan sumber tegangan jala-jala listrik 220 volt, supaya tegangan jala-jala terpisah dari bagian lainnya, dipakai *optoisolator* MOC3021 untuk menghubungkan AT89S51 dengan *triac*.

Bagian *input* dari MOC3021 merupakan LED yang dinyala-padamkan oleh AT89S51 lewat resistor R2_{a-h} dan menyalurkan arus dari Vcc lewat LED dan R2_{a-h} ke *pin* P1.0 sehingga LED menyala. Cahaya LED mengakibatkan diac di bagian *output* MOC3021 menjadi “ON” dan mengalirlah arus *gate triac* lewat R1, selanjutnya *triac* akan “ON” dan lampu AC akan menyala . *Interfacing triac* dengan mikrokontroler AT89S51 ditunjukkan dalam gambar 3.8.



Gambar 3.8. *Interfacing Triac* dengan Mikrokontroler AT89S51

Karakteristik *triac* Q4004L4 yang digunakan terlihat dalam tabel 3.2.

Tabel 3.2. Karakteristik *Triac* Q4004L4

Seri Piranti	V_{GT} (V)	I_{GT} (mA)	I_{max} (A)	V_{max} (V)
Q4004L4	2,5	25	4	400

Menentukan nilai R2 (a-h)

$$R_2 = \frac{V_{cc} - V_{forward_MOC}}{I_{forward_MOC}}$$

$$V_{FORWARD} = 1,5 \text{ volt (datasheet MOC)}$$

$$I_{FORWARD} = 15 \text{ mA (datasheet MOC)}$$

$$R_2 = \frac{5V - 1,5V}{15mA}$$

$$R_2 = 233,3 \Omega$$

Karena di pasaran tidak terdapat nilai $R_2 = 233,3 \Omega$, maka dipilih nilai hambatan sebesar yaitu **270 Ω** .

Menentukan nilai Rgate R1 mengacu rumus (2.9) :

$$V_{gate} = I_{gate} \cdot R_{gate}$$

$$I_{gate} = 25 \text{ mA (data sheet triac Q4004L4)}$$

$$R_{gate} = \frac{V_{gate}}{I_{gate}}$$

$$R_{gate} = \frac{2,5V}{25mA} = \mathbf{100 \Omega}$$

3.1.4. Rangkaian Mode *Trainer*

Mode ini terdiri dari rangkaian pemilih dan indikatornya, yang terhubung menjadi satu pada port mikrokontroler (P3.5, P3.6). Terdiri dari 2 bagian, yaitu Mode Kecepatan *Trainer* dan Mode Durasi Lama *Trainer*.

3.1.4.1. Tombol Pemilih Mode Kecepatan *Trainer*

Pemilih mode *trainer* menggunakan tombol N.O. SPST (*Single Pole Single Throw*) yang bersifat *momentary*. Penekanan tombol akan menghubungkan mikrokontroler dengan *ground* sehingga *port* yang terhubung akan menerima masukan berlogika 0. Pemilih mode terdiri dari 1 tombol yang dikoneksikan ke mikrokontroler pada P 3.5 dan mekanisme berapa kali tombol tersebut ditekan menentukan berapa mode kecepatan *trainer* yang dipilih. Penekanan 1 kali akan memberikan mode 3 detik, 2 kali tekan untuk 4 detik, dan 3 kali tekan untuk 5 detik.

LED berfungsi sebagai indikator mode *trainer* yang dipilih sesuai dengan penekanan tombol pemilihan mode *trainer* oleh pemain. LED yang digunakan berwarna merah sebagai indikator mode. LED tersebut dihubungkan dengan sumber tegangan 5 Volt (V_{cc}) dan sebuah hambatan sebagai pembatas arus.

Indikator menyala saat *port* mikrokontroler yang terhubung dengan LED berlogika 0. Keadaan ini terjadi setelah penekanan tombol pemilih mode *trainer* yang menyebabkan mikrokontroler terhubung dengan *ground*.

LED dihubungkan dengan konfigurasi *Common Anode* karena kutub anoda LED tersebut dihubungkan menjadi satu ke catu daya 5 Volt (V_{cc}). LED membutuhkan tegangan yang berbeda sesuai warna yang dipancarkan, untuk menghasilkan intensitas cahaya 100% (lampiran *data sheet*) di butuhkan arus sampai 20 mA, namun LED dapat menyala dengan arus 10 mA meskipun intensitas cahayanya tidak sampai 100%.

Bila mendapat logika 0 dari *port 1* yang tegangannya (V_{OL})= 0,45 V, sehingga perhitungan penentuan nilai hambatan sebagai berikut :

$$R = \frac{(V_{CC} - V_{OL}) - V_{LED}}{I_{LED}}$$

Jika penyalan LED merah membutuhkan tegangan 2,1 V (lampiran *data sheet*) untuk menghasilkan arus 10 mA, maka nilai hambatan yang dibutuhkan adalah :

$$R = \frac{(5V - 0,45V) - 2,1V}{10mA}$$

$$R = \frac{2,45V}{10mA} = 245 \Omega$$

Karena nilai hambatan tidak terdapat di pasaran, maka dapat digunakan hambatan 220 Ω , sehingga arus yang mengalir melalui LED merah :

$$I_{LED} = \frac{(5V - 0,45) - 2,1V}{220\Omega} = \frac{2,45V}{220\Omega} = 11,14 \text{ mA}$$

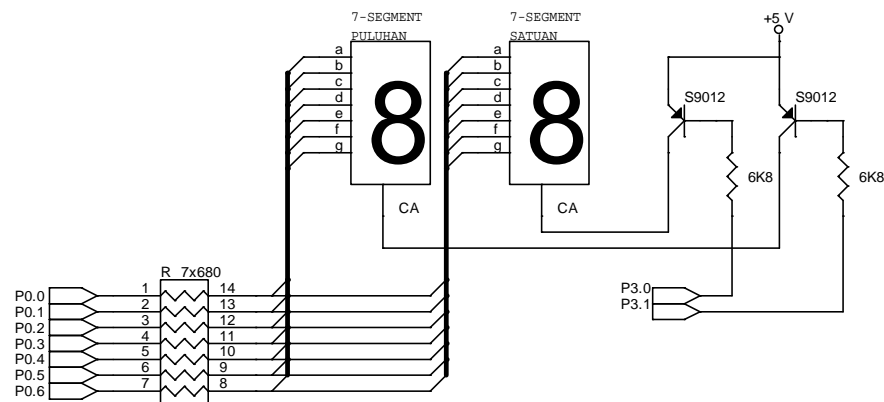
Dengan arus sebesar 11,14 mA, LED merah dapat bekerja (menyala).

3.1.4.2. Tombol Pemilih Mode Durasi *Trainer*

Pemilih mode ini menggunakan tombol N.O. SPST (*Single Pole Single Throw*) yang bersifat *momentary*. Penekanan tombol akan menghubungkan mikrokontroler dengan *ground* sehingga *port* yang terhubung akan menerima masukan berlogika 0. Pemilih mode terdiri dari 1 tombol yang dikoneksikan mikrokontroler pada P 3.6 dengan ketentuan, penekanan tombol 1 kali adalah mode 30 detik, 2 kali tekan mode 60, dan penekanan 3 kali untuk mode 90 detik.

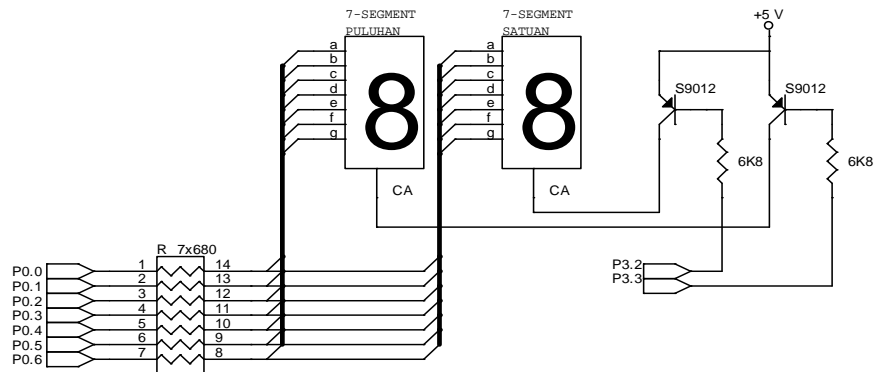
3.1.5. Rangkaian Penampil *Seven Segment*

Seven segment digunakan sebagai penampil skor pada *trainer* bulutangkis, berupa 2 digit (satuan, puluhan). *Seven segment* juga digunakan sebagai penampil waktu *trainer* (*count down*) selama (90 detik, 60 detik, dan 30 detik), berupa 2 digit puluhan dan satuan. Skematik rangkaian penampil *seven segment* skor *trainer* dapat dilihat pada gambar 3.9.



Gambar 3.9. Rangkaian Penampil Skor

Skematik rangkaian penampil *seven segment* waktu *trainer* dapat dilihat pada gambar 3.10.



Gambar 3.10. Rangkaian Penampil Waktu

Penampil data pada *seven segment* menggunakan metode *scanning display*. Metode ini bekerja menampilkan data pada dua *seven segment* secara bergantian. Namun, dengan kecepatan tinggi akan tampak seolah bersamaan. Rangkaian terdiri dari rangkaian LED dengan hambatan dan rangkaian penggerak dengan transistor.

3.1.5.1. LED Penampil Seven Segment

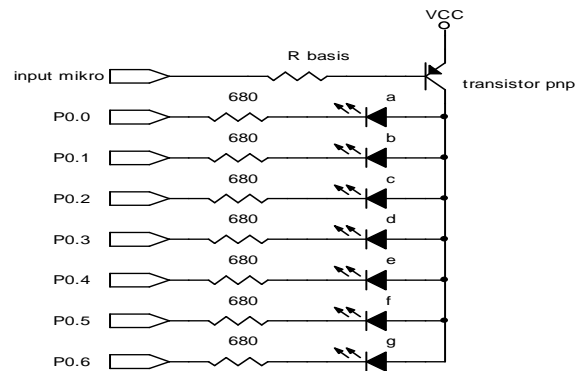
Pada gambar 3.9 dan 3.10, LED *seven segment* dihubungkan dengan port 0 mikrokontroler AT89S51. Port 0 (kecuali P0.7) digunakan untuk mengirimkan data *segment* sebanyak 7 bit, yaitu a, b, c, d, e, f, dan g yang secara berurutan mendapatkan masukan dari P0.0 sampai P0.6 mikrokontroler. Bit untuk *segment* dp tidak digunakan sehingga dibiarkan dalam keadaan logika 1. Tabel 3.3 adalah tabel data keluaran dari port 0 untuk menyalakan *segment* (0 = menyala, 1 = tidak menyala).

Tabel 3.3. Tabel Data Keluaran Port 0 untuk Penyalaan LED

Karakter	Segment								Data Heksadesimal
	dp	g	f	e	d	c	b	a	
0	1	1	0	0	0	0	0	0	C0
1	1	1	1	1	1	0	0	1	F9
2	1	0	1	0	0	1	0	0	A4
3	1	0	1	1	0	0	0	0	B0
4	1	0	0	1	1	0	0	1	99
5	1	0	0	1	0	0	1	0	92
6	1	0	0	0	0	0	1	1	83
7	1	1	1	1	1	0	0	0	F8
8	1	0	0	0	0	0	0	0	80
9	1	0	0	1	0	0	0	0	90

3.1.5.2. Resistor Pembatas Arus *Seven Segment*

Seven segment mendapatkan tegangan masukan dari *port 0* mikrokontroler. Setiap *segment* pada *seven segment* adalah LED yang membutuhkan arus cukup agar dapat menyala dan arus tidak melebihi batas maksimal yang diijinkan agar LED tidak rusak. Tegangan setiap LED *segment* pada *seven segment* yaitu 1,7 V; arus 5 mA (lampiran *data sheet 7 segment*); dan V_{OL} (tegangan keluaran logika 0 AT89S51) = 0,45V (sesuai *data sheet* AT89S51). Gambar rangkaian ekuivalen dari *seven segment common anode* dapat dilihat pada gambar 3.11.



Gambar 3.11. Rangkaian Ekivalen Sebuah *Seven Segment*

Perhitungan nilai hambatan sebagai pembatas arus LED dengan menggunakan persamaan 3.2 diperoleh hasil sebagai berikut :

$$R = \frac{5V - (1,7V + 0,45V)}{5mA} = 570 \Omega$$

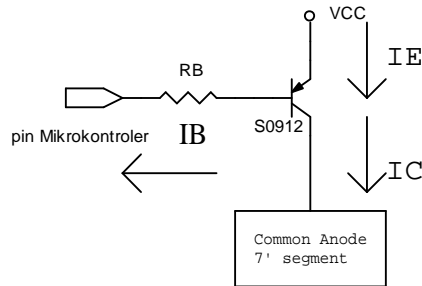
Karena nilai hambatan 570Ω tidak tersedia di pasaran maka dipilih nilai hambatan 680Ω untuk menyalakan LED. Besar arus yang mengalir melalui tiap *segment* adalah:

$$I_{LED} = \frac{5V - (1,7V + 0,45V)}{680\Omega} = 0,00419 A = 4,19 mA$$

3.1.5.3. Transistor Penggerak *Seven Segment*

Pada penampil skor, P3.0 dan P3.1 mikrokontroler digunakan untuk mengaktifkan *common* dari kedua *seven segment* secara bergantian. Pada penampil waktu pengendali *common* kedua *seven segment* adalah P3.2 dan P3.3. Sebagai pemultipleks tampilan digunakan sebuah transistor yang berfungsi sebagai saklar yang akan mengalirkan arus ke *common anode seven segment* (I_C)

agar dapat menyala. Pemilihan nilai hambatan basis (R_B) untuk membias basis transistor sangat menentukan agar transistor dapat bekerja secara ideal.



Gambar 3.12. Rangkaian Penggerak Sebuah *Seven Segment*

Pada gambar 3.12, nilai arus yang mengalir pada masing-masing LED adalah 5 mA, sehingga untuk tujuh LED membutuhkan I_C sebesar :

$$I_C = \sum I_{LED} = 7 \times 5 \text{ mA} = 35 \text{ mA}$$

Penentuan nilai hambatan basis (R_B) dengan transistor CS 9012 (β minimum = 64) dengan mencari nilai arus basis :

$$I_B = \frac{35 \text{ mA}}{64} = 0,546875 \text{ mA}$$

$$I_B \cong 0,55 \text{ mA}$$

Dengan $V_{CC} = 5 \text{ V}$ dan $V_{OL} = 0,45 \text{ V}$ (lembar data AT89S51), V_{BB} transistor = $V_{CC} - V_{OL}$, maka nilai hambatan basis (R_B) adalah :

$$R_B = \frac{V_{BB} - V_{BE}}{I_B}$$

$$R_B = \frac{(5 - 0,45) - 0,7 \text{ V}}{0,55 \text{ mA}} = 7000 \, \Omega$$

Maka, dengan nilai hambatan yang terdapat di pasaran dipilih nilai R_B sebesar 6,8 K Ω . Arus kolektor yang mengalir pada transistor (I_C) adalah :

$$I_B = \frac{V_{BB} - V_{BE}}{R_B}$$

$$I_B = \frac{(5V - 0,45V) - 0,7V}{6800\Omega}$$

$$I_B = \frac{3,85V}{6800\Omega} = 0,566 \text{ mA}$$

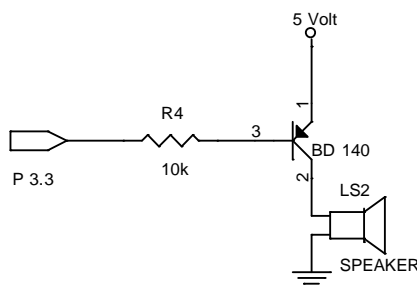
Nilai I_C yang mengalir dengan menggunakan persamaan 2.5 adalah :

$$I_C = (64) \cdot (0,566 \text{ mA}) = 32,832 \text{ mA}$$

Dengan demikian transistor dapat dilewati arus 32,832 mA.

3.1.6. Rangkaian Penggerak *Speaker*

Bunyi *speaker* dihasilkan dari frekuensi yang dibangkitkan oleh keluaran pin 3.3 dengan perangkat lunak mikrokontroler. Dalam penggunaannya dibutuhkan sebuah penggerak berupa transistor yang berfungsi sebagai saklar. Gambar 3.13 adalah rangkaian penggerak *speaker*.



Gambar 3.13. Rangkaian Penggerak *Speaker*

Speaker berbunyi jika terdapat arus I_C yang mengalir (lihat gambar 3.13).

Besar nilai arus I_C sama dengan arus yang dibutuhkan *speaker* untuk bekerja.

Dari spesifikasi speaker yang digunakan dalam perancangan yaitu 0,5 Watt; 8 Ω ,

dapat ditentukan nilai arus maksimal untuk mengaktifkan *speaker* tersebut yaitu:

$$I_{\text{spea ker}} = \sqrt{\frac{P_{\text{spea ker}}}{R_{\text{spea ker}}}}$$

$$I_{\text{spea ker}} = \sqrt{\frac{0,5}{8}} = \sqrt{0,0625}$$

$$I_{\text{spea ker}} = 0,25 \text{ A} = 250 \text{ mA}$$

Arus kolektor dihasilkan bila transistor BD 140 dalam keadaan *ON* dengan adanya arus basis (I_B). Arus kolektor (I_C) = $I_E - I_B$, karena I_B sangat kecil dibandingkan I_E , maka dianggap $I_C \approx I_E$. Pada perancangan nilai I_C maksimal sama dengan nilai arus maksimum *speaker* = 250 mA.

Berdasarkan *data sheet* transistor BD 140, dapat ditentukan nilai $V_{EC} = 2$ V dan nilai penguatan arus (h_{FE}) dengan $I_C = 250$ mA adalah 125. Hambatan basis (R_B) berfungsi sebagai pembatas arus agar transistor tidak rusak dan dapat menghasilkan I_C yang aman untuk mengaktifkan *speaker*. Hambatan basis ditentukan dengan menghitung arus basis maksimum (I_B) menggunakan persamaan 2.6 :

$$I_B = \frac{I_C}{h_{FE}} = \frac{I_{\text{spea ker}}}{h_{FE}}$$

$$I_B = \frac{250 \text{ mA}}{125} = 2 \text{ mA}$$

Nilai R_B dengan arus basis sebesar 2mA, V_{EB} maksimum = 1 V dan $V_{EC} = 2$ V (*data sheet* BD 140) serta V_{OL} mikrokontroler sebesar 0,45 V adalah:

$$R_B = \frac{V_{CC} - V_{EB} - V_{OL}}{I_B}$$

$$R_B = \frac{(5 - 1 - 0,45)V}{2mA} = 1775 \Omega.$$

Nilai hambatan basis 1775Ω adalah minimum yang menyebabkan kinerjanya buruk karena dapat mengakibatkan kerusakan *speaker*. Pada perancangan digunakan hambatan yang lebih besar nilainya, yaitu $10 K\Omega$, agar arus kolektor transistor tidak melebihi nilai maksimum yaitu 250 mA yang mengakibatkan kerusakan pada *speaker*. Arus basis dengan hambatan $10 K\Omega$ adalah :

$$I_B = \frac{(5 - 1 - 0,45)V}{10K\Omega} = 0,35 \text{ mA}$$

Dengan penguatan arus maksimum transistor BD 140 sebesar 250, maka arus kolektor maksimum yang dihasilkan sebesar:

$$I_C = 250 \cdot 0,35 = 87,5 \text{ mA}$$

Nilai $R_B = 10 K\Omega$ dapat digunakan berdasarkan hasil perhitungan di atas. Arus kolektor sebesar $87,5 \text{ mA}$ lebih kecil dari nilai arus kolektor maksimum agar transistor bekerja, yaitu sebesar 500 mA dan dapat digunakan untuk mengaktifkan *speaker*.

3.2. Perancangan Perangkat Lunak

Untuk dapat merancang sebuah perangkat lunak maka perlu sebuah petunjuk bagaimana mekanisme alat tersebut bekerja. Pengoperasian alat menjelaskan bagaimana cara kerja dan cara menggunakan “*Circuit Trainer*”.

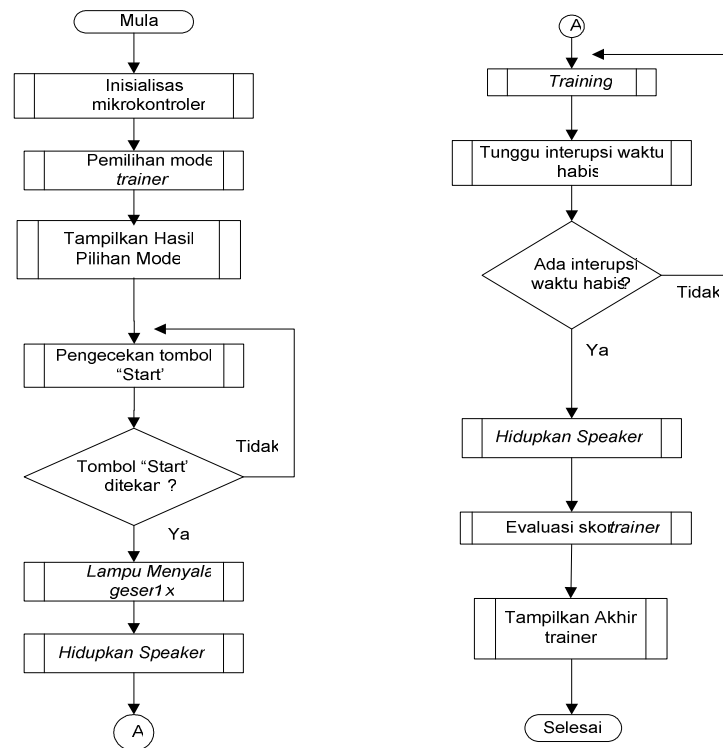
Langkah-langkah pengoperasian alat dijelaskan sebagai berikut:

1. Menekan tombol *ON*, untuk mengaktifkan catu daya untuk menghidupkan alat.
2. Pemain memilih mode *trainer* dengan menekan tombol mode. LED indikator yang terletak di dalam kotak yang transparan akan terlihat menyala jika ada penekanan. Pada 2 digit *seven segment* bawah dan atas akan ditampilkan mode yang dipilih, meliputi mode kecepatan *trainer* untuk 2 digit *seven segment* atas dan mode durasi *trainer* untuk 2 digit *seven segment* bawah. Jumlah penekanan akan membagi mode menjadi sub-mode. Untuk mode kecepatan *trainer*, jika tombol ini ditekan 1 kali maka akan ditampilkan sub-mode 3S artinya kecepatan *trainer* adalah 3 detik. Jika 2 kali tekan akan ditampilkan 4S, dan 3 kali tekan akan ditampilkan 5S. Untuk mode durasi, jumlah penekanan tombol akan membagi mode menjadi sub-mode durasi, untuk 1 kali tekan akan ditampilkan angka 30 detik, 2 kali tekan 60 detik, dan 3 kali tekan untuk durasi 90 detik.
3. Setelah mode dipilih, tekan tombol *start*. Penekanan tombol *start* akan menghapus tampilan 2 digit *seven segment* atas dan menggantikan fungsi 2 digit *seven segment* tersebut sebagai penampil skor dengan angka 00 sebagai

permulaan. Untuk 2 digit *seven segment* bawah akan tetap menampilkan durasi waktu.

4. Setelah tombol start ditekan, ke 8 lampu menyala selama 2 detik kemudian kedelapan lampu akan mati 1 per 1 secara bergeser dengan *delay* 1 detik. Tunggu selama 8 detik untuk memberi waktu untuk pemain bergerak menuju tengah lapangan dan berkonsentrasi. Setelah siap akan terdengar suara speaker selama 2 detik dan pelatihan dimulai.
5. Pemain melakukan penekanan tombol *trainer*, jika posisi tombol yang ditekan sesuai dengan posisi lampu yang menyala, maka skor benar bertambah satu.
6. Pelatihan akan berakhir jika waktu *trainer* sudah menampilkan angka 00 dan terdengar bunyi *speaker* selama 5 detik.
7. Untuk skor di evaluasi skor akan terlihat jumlah skor *trainer*. 2 digit *seven segment* atas untuk skor benar dan 2 digit *seven segment* bawah untuk skor salah.
8. Untuk mengulangi pelatihan atau berganti mode *trainer* tekan tombol *reset* atau tekan tombol *ON/OFF*. Kemudian menuju Langkah ke 2.

Untuk dapat mendukung perancangan perangkat keras, di rancang sebuah perangkat lunak yang bertugas mengatur kerja dari perangkat keras. Algoritma perancangan perangkat lunak dapat dilihat pada gambar 3.14.



Gambar 3.14. Diagram Alir Perangkat Lunak *Circuit Trainer*

3.2.1. Inisialisasi Awal

Pada bagian awal program ini akan diinisialisasi variabel-variabel dan konstanta yang akan dipergunakan didalam program serta pengkondisian awal terhadap variabel yang diinginkan ketika program akan dijalankan, meliputi inisialisasi port, menghidupkan interupsi secara global dan inisialisasi timer. Tabel inisialisasi awal dapat dilihat di table 3.4.

Inisialisasi port meliputi:

<i>Ruas_Penampil</i>	<i>equ</i>	<i>P0</i>	<i>; Label Port 0</i>
<i>Lampu_AC</i>	<i>equ</i>	<i>P1</i>	<i>; Label Port 1</i>
<i>Tombol_trainer</i>	<i>equ</i>	<i>P2</i>	<i>; Label Port 2</i>
<i>Speker_TOA</i>	<i>bit</i>	<i>P3.3</i>	<i>; p3.3 kendali Speaker</i>
<i>Puluhan_skor</i>	<i>bit</i>	<i>P3.1</i>	<i>; p3.1 kendali ruas skor puluhan</i>
<i>Satuan_skor</i>	<i>bit</i>	<i>P3.0</i>	<i>; p3.0 kendali ruas skor satuan</i>

<i>Satuan_waktu</i>	<i>bit</i>	<i>P3.4</i>	<i>; p3.4 kendali ruas satuan waktu</i>
<i>Puluhan_waktu</i>	<i>bit</i>	<i>P3.5</i>	<i>; p3.5 kendali ruas puluhan waktu</i>
<i>Mode1_kecepatan</i>	<i>bit</i>	<i>P3.6</i>	<i>; p3.6 tombol mode kecepatan</i>
<i>Mode2_durasi</i>	<i>bit</i>	<i>P3.7</i>	<i>; p3.7 tombol mode durasi</i>
<i>Tombol_start</i>	<i>bit</i>	<i>P0.7</i>	

Inisialisasi konstanta:

<i>Tone_Mulai</i>	<i>equ</i>	<i>0F0h</i>	<i>;Tone_Mulai bernilai 0F0h</i>
<i>Tone_Selesai</i>	<i>equ</i>	<i>0Fh</i>	<i>;Tone_Selesai bernilai 0Fh</i>
<i>Trainer_Mulai</i>	<i>equ</i>	<i>70h</i>	<i>;Trainer_Mulai bernilai 70h</i>
<i>Trainer_Selesai</i>	<i>equ</i>	<i>07h</i>	<i>;Trainer_Selesai bernilai 07h</i>

Tabel 3.4. Inisialisasi Variabel

Nama Variabel	Alamat Memori
Tone_Mulai	0F0h
Tone_Selesai	0Fh
Trainer_Mulai	70h
Trainer_Selesai	07h
Random_temp	23h
Hasil_random	30h
Data_Lampu_AC	31h
Skor_Benar	32h
Skor_Salah	33h
Cacah_start	34h
Data_Temp_Kecs	35h
Data_Per_Kec	36h
Data_Temp_1s	37h
Counter_50ms	38h
Counter_05ms	39h
Tekan_apa	3Ah
Jumlah_random	3Bh
Data_Kecepatan	3Ch
Data_Durasi	3Dh
Data_Skor	3Eh
Kol_Puluhan_skor	3Fh
Kol_Satuan_skor	40h
Kol_Puluhan_waktu	41h
Kol_Satuan_waktu	42h
Tanda_Tone	43h
Tanda_Trainer	44h
Awal_data_7seg_RAM	50h

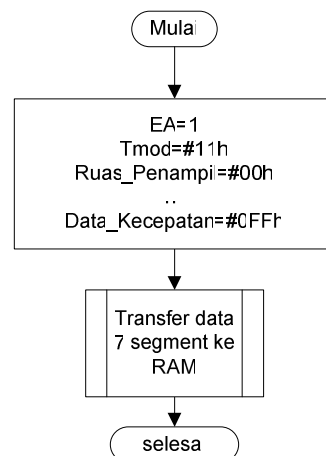
Untuk sub-routine Inisialisasi_awal diawali dengan pengaktifan interupsi secara global, penggunaan timer 1 dan timer 0 pada mode 1 (16 bit). Tampilan

ruas penampil dimatikan, variabel Skor_Benar dan Skor_Salah diisi #00h, Cacah_start diisi #10, Kol_Satuan_skor diisi dengan #0FFh (demikian juga untuk Kol_Puluhan_skor, Kol_Satuan_waktu, Kol_Puluhan_waktu), Data_Durasi dan Data_Kecepatan diisi #0FFh. Pengaktifan interupsi dilakukan dengan perintah:

Setb EA ; interupsi global diaktifkan

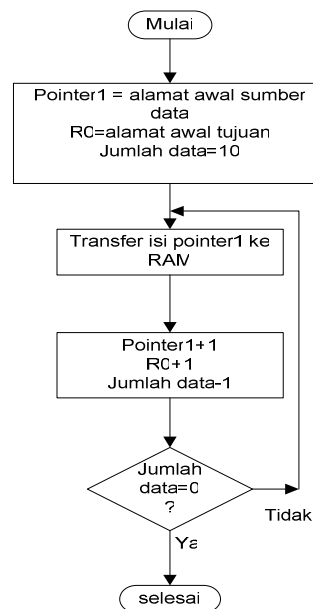
Inisialisasi timer dengan mengisi register TMOD dengan #11h, yang artinya timer yang dipakai adalah timer 1 dan timer 0 dengan mode 1 (16 bit).

Proses transfer data *seven segment* ke RAM berguna untuk memindahkan data penampil *seven segment* pada tabel *look-up* ke RAM pada alamat 50h. Proses pemindahan dilakukan agar data lebih mudah diolah, mulai dari alamat 50h yang berisi data penampil angka 0, sampai 5AH yang berisi data penampil angka 9. Aliran program Inisialisasi_awal dapat dilihat pada gambar 3.15.



Gambar 3.15. Diagram Alir Inisialisasi_awal

Sub-routine *transfer data seven segment* ke RAM dapat dilihat pada diagram alir gambar 3.16.



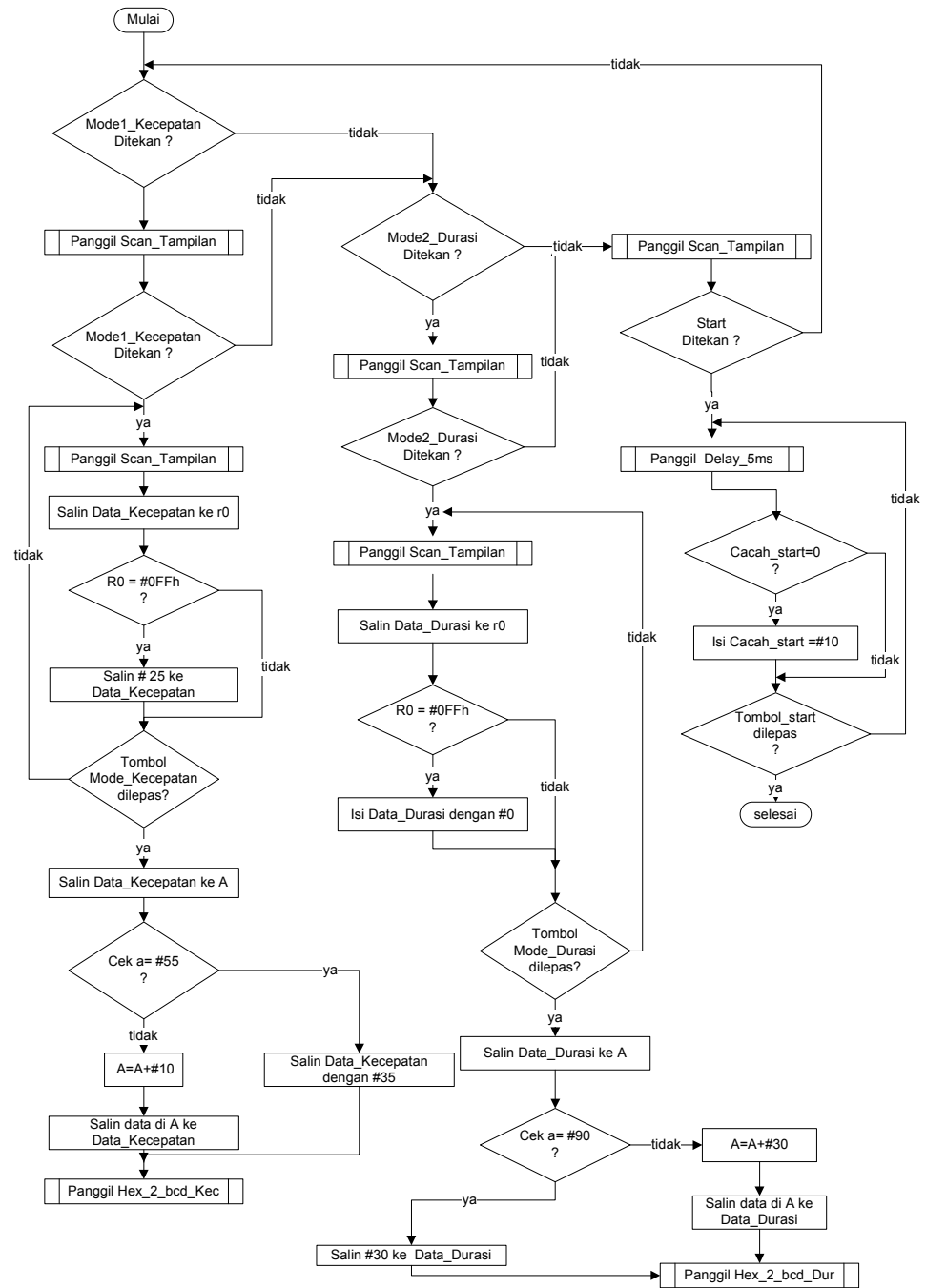
Gambar 3.16. Diagram Alir Transfer Data *Seven Segment* ke RAM

3.2.2 Pemilihan Mode

Routine ini akan menentukan pemilihan mode yang dipilih, yaitu mode kecepatan (3S, 4S, dan 5S) dan mode durasi (30, 60, 90 detik). Jumlah penekanan pada 2 tombol tersebut akan menentukan mode yang dipilih. Cara ini dipilih untuk meringkas perangkat keras karena hanya menggunakan 2 pin mikrokontroler. *Routine* Scan_Tampilan selain melakukan *scanning* tampilan di ruas penampil juga berfungsi mengatasi masalah *bouncing* yang terjadi pada saklar mekanik. Setiap terjadi penekanan kedua tombol mode *routine*

Scan_Tampilan akan dipanggil, *routine* ini memerlukan waktu kurang lebih 24,8 ms.

Langkah pertama ialah mengecek adanya penekanan tombol mode kecepatan, jika tidak ditekan lompat ke M_Dura, panggil Scan_Tampilan. Cek lagi tombol mode kecepatan ditekan atau tidak, jika tidak lompat ke M_Dura, panggil Scan_Tampilan. Jika tombol kecepatan yang ditekan maka cek apakah data kecepatan sama dengan 0FFh, jika sama dengan 0FFh maka angka 25 diisikan ke Data_Kecepatan, tunggu tombol kecepatan dilepas. Jika data di R0 tidak sama dengan 0FFh lompat ke Cek1. Jika tombol dilepas data kecepatan akan dibandingkan dengan angka 55. Jika sama maka data kecepatan adalah 35. Selanjutnya lompat ke Terus1 untuk panggil *routine* Hex_2_bcd_Kec. Jika tidak, lompat ke Lewat1 tambahkan data tersebut dengan 10. Salin data di akumulator ke Data_Kecepatan, kemudian lompat ke Terus1 untuk panggil *routine* Hex_2_bcd_Kec. Perlu diperhatikan bahwa angka 35, 45, dan 55 yang dihasilkan pada *routine* ini akan diambil digit puluhannya saja. Angka 5 disetiap digit satuan adalah lambang *second* (S). Untuk itu *routine* Cek_Perulangan berguna untuk mengambil digit puluhan Data_ Kecepatan dan disimpan ke Data_Per_Kec.



Gambar 3.17. Diagram Alir Pemilihan Mode

Diagram alir gambar 3.17 adalah algoritma program pemilihan mode dan juga pengecekan penekanan tombol *start*. Lama penekanan tombol *start* sangat mempengaruhi nyala *random* lampu. Dengan menekan tombol *start*, akan diperoleh variabel *Cacah_start* yang menjadi angka *random* pertama yang nantinya akan digunakan dalam rutin *random*. Angka *random* pertama tersebut antara 1 sampai 10. Dalam perancangan ditentukan cacahan pertama adalah 10 yang akan mencacah turun setiap 5 ms dan akan berhenti setelah tombol “START” dilepas. Angka *random* tidak boleh bernilai 0 karena mekanisme *random* tidak akan berjalan. Ketika penekanan tombol *start*, cacahan yang diperoleh angka 0 maka secara *software* angka cacahan itu diubah menjadi 10 dengan perintah *mov Cacah_start, #10*.

<i>M_Start:</i>		<i>;Cek Tombol Start</i>
<i>call</i>	<i>Scan_Tampilan</i>	<i>;Panggil Scan_Tampilan</i>
<i>jb</i>	<i>Tombol_start,M_Kec</i>	<i>;Start ditekan? Tidak = M_Kec</i>
<i>Lagi:</i>		
<i>call</i>	<i>Delay_5ms</i>	<i>;Ya = tunda 5 ms</i>
<i>djnz</i>	<i>Cacah_start,Isi_random</i>	<i>;Cacahan -1, Cacahan=0?</i>
<i>mov</i>	<i>Cacah_start,#10</i>	<i>;Ya -> isi cacah =10</i>
<i>Isi_random:</i>		
<i>jnb</i>	<i>Tombol_start,Lagi</i>	<i>;Tombol Start dilepas? Tidak = Cacahan-1</i>
<i>ret</i>		

3.2.3. Routine Start_main

Bagian *Start_main* ini akan memanggil *Cek_Perulangan* yang akan memperoleh data berupa data yang disimpan di *Data_Per_Kec*. Diagram alir dapat dilihat pada Gambar 3.18.



Gambar 3.18. Diagram Alir Start Main

Potongan program *Cek_Perulangan* akan menghasilkan *Data_Per_Kec* yaitu 3 detik, 4 detik, dan 5 detik. *Routine* ini akan menyimpan angka 5(=Second) di register B sebagai hasil bagi instruksi *div ab* (S artinya *second* / detik) di belakang 3, 4, atau 5 yang tertampil di ruas atas *digit* satuan mode kecepatan.

Cek_Perulangan:

```

mov    a,Data_Kecepatan    ;Isi akumulator dengan data di Data_Kecepatan
mov    b,#10               ;Isi register b dengan angka 10
div     ab                 ;Skor dibagi 10, hasil di A, sisa di B1
mov     Data_Per_Kec,a      ;Hasil bagi disimpan di Data_Per_Kec
ret

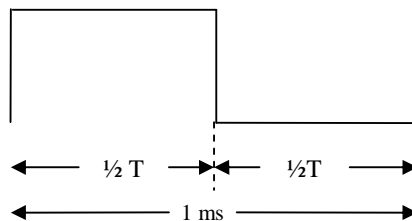
```

Tanda_Siap akan menampilkan data skor 00 kemudian menyalakan ke 8 lampu serempak ketika tombol start ditekan dan mematikannya secara bergeser dengan interval waktu 1 detik untuk tiap lampunya.

Panggil Tone_Speaker merupakan *routine* untuk membangkitkan suara dari speaker dengan frekuensi 1 KHz yang berguna sebagai tanda ketika *trainer* akan dimulai, selama 2 detik. Sehingga total waktu untuk start main adalah 10

detik. Waktu 10 detik ini digunakan pemain untuk bersiap-siap dan berkonsentrasi ditengah lapangan.

Speaker akan *ON* dan *OFF* dalam satu periode agar menghasilkan bunyi, sehingga diperlukan tunda waktu sebesar $\frac{1}{2} T$ dalam keadaan *ON* dan *OFF* sehingga menghasilkan gelombang kotak seperti pada gambar 3.19.



Gambar 3.19. Gelombang Kotak dengan Frekuensi 1KHz

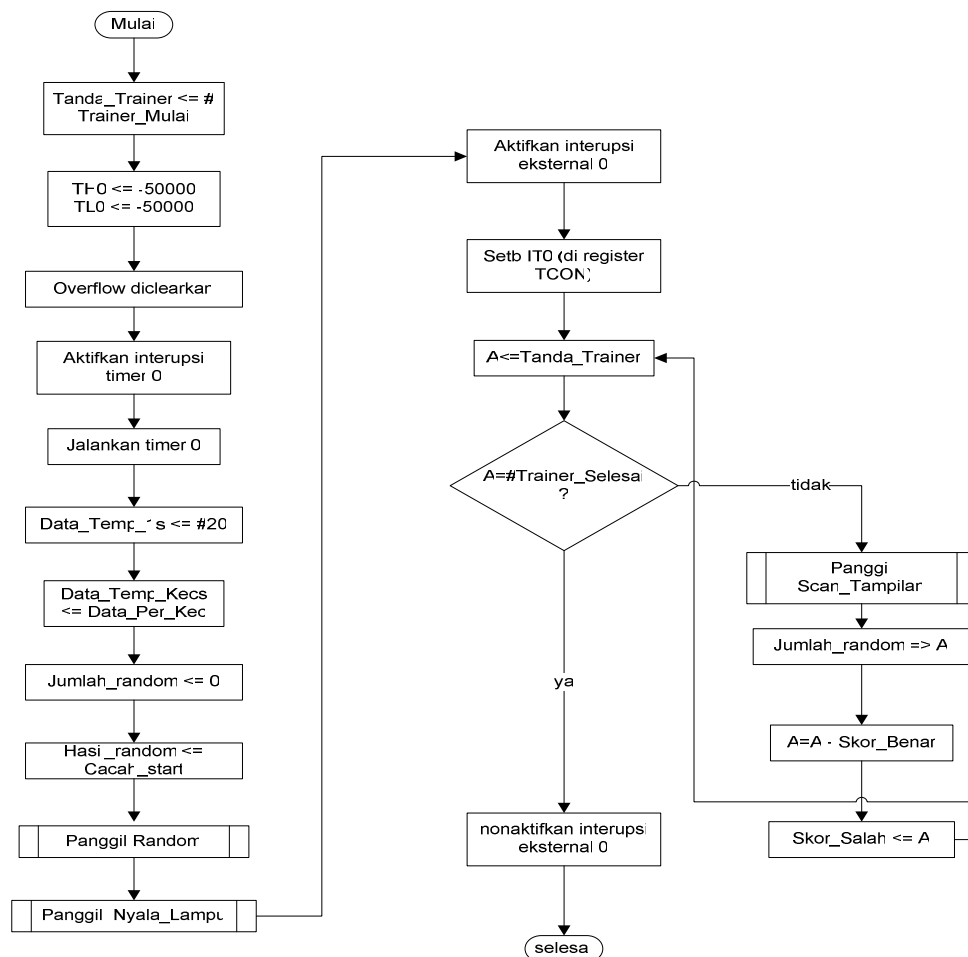
Jika frekuensi yang diinginkan adalah 1 KHz, maka nilai periode $T = 1$ ms. Untuk $\frac{1}{2} T$ *ON* akan diperoleh waktu sebesar 0,5 ms, dan $\frac{1}{2} T$ *OFF* juga 0,5 ms.

Maka untuk menghasilkan tundaan $\frac{1}{2}$ gelombang sebesar 0,5 ms, register TH1 dan TL1 diisi dengan angka -500. Setiap pergantian 1 siklus dari -500 => -499, membutuhkan $1\mu s$. Jika timer berjalan dan menghitung maju dari -500 ke 0 akan dibutuhkan sebanyak 500 siklus artinya tundaan yang dihasilkan adalah $500 \times 1\mu s = 0,5$ ms.

Untuk mengaktifkan *speaker* selama 2 detik maka dibutuhkan 0,5ms sebanyak 4000 kali. Maka Counter_50ms diisi dengan 40 dan Counter_05ms diisi 100.

3.2.4. Trainer

Bagian trainer ini adalah bagian yang cukup kompleks, karena mikrokontroler bekerja untuk mengerjakan beberapa tugas hampir bersamaan. Pertama mikrokontroler harus melakukan *scanning* tampilan, kedua memeriksa tombol *trainer* mana yang ditekan, kemudian menjalankan proses *random*, menghitung mundur durasi, dan menyalakan lampu AC. Untuk lebih jelasnya lihat diagram alir *routine trainer* Gambar 3.20.



Gambar 3.20. Diagram Alir *Trainer*

Interupsi Eksternal 0 ini merupakan interupsi yang berasal dari masukan tombol trainer. Jika dilihat di rangkaian bagian tombol trainer, setiap tombol dihubungkan ke pin P3.2 dari mikrokontroler melewati dioda. Pin ini memiliki fungsi khusus sebagai masukan interupsi eksternal 0.

Angka -50000 yang diisikan ke TH0 dan TL0 akan menghasilkan waktu tunda sebesar 50 ms, itu dengan ketentuan mikrokontroler bekerja dengan kristal 12 Mhz. Ketika cacahan -50000 mencapai 0 maka itu membutuhkan 50000 siklus, 1 siklus memerlukan waktu $1\mu s$. Sehingga waktu untuk mencacah sampai 0 dibutuhkan waktu $50.000 \times 1\mu s = 50\text{ ms}$.

Jumlah *random* diberikan sebesar 0 sebagai angka awal. Pada program Jumlah_random akan di-*increment* sehingga hasil *increment* akan bertambah. Secara sederhana jumlah *random* dapat dihitung, di setiap 1 kali nyala lampu maka itu diartikan jumlah *random* bertambah 1, misalnya mode durasi yang dipilih 30; mode kecepatan 3S (3 Second) maka jumlah *random* adalah 30 dibagi 3 yaitu = 10 kali nyala lampu *random*.

Cacah_start adalah nilai yang diperoleh ketika penekanan tombol *start* dilakukan, nilai ini disalin ke Hasil_random untuk diproses ke *routine random*. Penekanan tombol “*START*” dilakukan untuk memulai *trainer* dan berfungsi untuk menghasilkan angka pertama yang akan diacak (*random*). Angka pertama yang akan diacak diperoleh dari lamanya penekanan tombol “*START*”. Dalam perancangan ditentukan cacahan pertama adalah 10 (lihat di Inisialisasi_awal) yang akan mencacah turun setiap 5 ms dan akan berhenti setelah tombol “*START*” dilepas.

Selanjutnya *routine random* akan dipanggil, kemudian proses penyalaan lampu dengan memanggil *Nyala_Lampu*. Ketika proses *random* dan penyalaan lampu berjalan, akan diinterupsi oleh 2 sumber, yaitu interupsi eksternal 0 berupa penekanan tombol dan interupsi timer 0.

Proses ini akan selesai ketika *Tanda_Trainer* sama dengan *#Trainer_Selesai* dan setelah itu interupsi eksternal 0 akan dinonaktifkan. Jika belum selesai lompat ke *Loop3*. *Loop3* akan diawali dengan *Scan_Tampilan* dan proses penyimpanan nilai *Skor_Benar* dan perhitungan *Skor_Salah*. Kemudian lompat ke *Ulang_main*.

```

Loop3:
    call    Scan_Tampilan          ;Panggil Scan_Tampilan
    mov     a,Jumlah_random        ;Akumulator <- Jumlah_random
    subb    a,Skor_Benar          ;Jumlah_random - Skor_Benar -> A
    mov     Skor_Salah,a          ;Skor_Salah <- A
    jmp     Ulang_main            ;Ulang trainer sampai ada interupsi

```

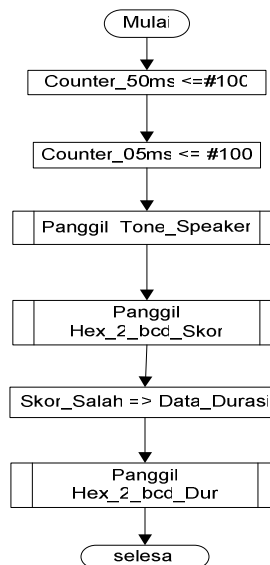
3.2.5. Evaluasi

Evaluasi adalah bagian yang akan menampilkan *Skor_Benar* dan *Skor_Salah* pada *seven segment*. *Skor_Benar* akan ditampilkan di 2 digit atas, *Skor_Salah* akan ditempatkan di 2 digit bawah yang sebelumnya dipakai sebagai *Data_Durasi*. *Routine Evaluasi* juga akan membunyikan *speaker* selama 5 detik yang menandakan waktu pelatihan telah usai.

Counter_50ms dan *Counter_05ms* diisi dengan #100, kemudian panggil *Tone_Speaker*. Setelah *routine Tone_Spaker* sudah selesai dijalankan, panggil *Hex_2_bcd_Skor*.

Salin *Skor_Salah* ke *Data_Durasi* karena nilai skor salah ini akan ditampilkan di ruas penampil yang sebelumnya digunakan sebagai penampil

data durasi. Selanjutnya panggil Hex_2_bcd_Dur. Diagram alir bagian evaluasi dapat dilihat pada Gambar 3.21.



Gambar 3.21. Diagram Alir Evaluasi

<i>Hex_2_bcd_Dur:</i>		
<i>mov b,#10</i>		<i>;Register b diisi dengan 10</i>
<i>mov a,Data_Durasi</i>		<i>;Data_Durasi salin ke Akumulator</i>
<i>div ab</i>		<i>;Skor dibagi 10</i>
<i>mov Kol_Satuan_waktu,b</i>		<i>;Nilai satuan = sisa hasil bagi</i>
<i>mov Kol_Puluhan_waktu,a</i>		<i>;Nilai puluhan = hasil bagi</i>
<i>jmp BCD_ke_7</i>		<i>;Lompat ke BCD_ke_7</i>
<i>Hex_2_bcd_Skor:</i>		<i>;Skor diubah dari Heksa ke BCD</i>
<i>mov b,#10</i>		<i>;Register b diisi dengan 10</i>
<i>mov a,Skor_Benar</i>		<i>;Skor_Benar => Akumulator</i>
<i>div ab</i>		<i>;Skor dibagi 10</i>
<i>mov Kol_Satuan_skor,b</i>		<i>;Nilai satuan = sisa hasil bagi</i>
<i>mov Kol_Puluhan_skor,a</i>		<i>;Nilai puluhan = hasil bagi</i>
<i>jmp BCD_ke_7</i>		<i>;Lompat ke BCD_ke_7</i>

Penggalan program diatas adalah proses pengubahan data *hexadecimal* ke BCD. Untuk Hex_2_bcd_Dur, Skor_Salah disalin ke Data_Durasi, kemudian dibagi dengan 10. Hasil bagi disimpan di akumulator kemudian disalin ke Kol_Puluhan_waktu, sedangkan sisa hasil bagi disimpan di register B dan

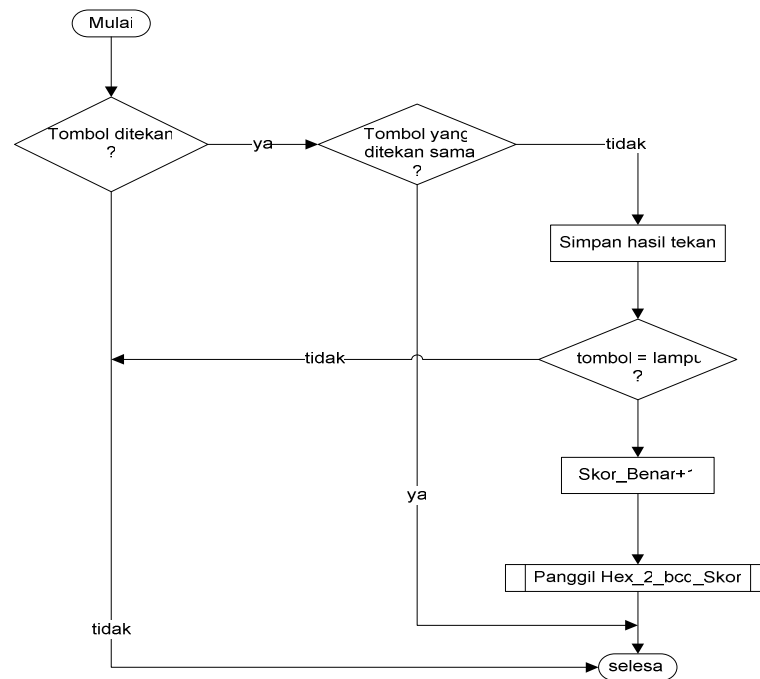
disalin ke Kol_Satuan_waktu, langkah selanjutnya lompat ke BCD_ke_7 .
Demikian juga untuk Hex_2_bcd_Skor, untuk skor proses yang dilakukan hampir sama hanya saja skor benar ditampilkan di ruas penampil skor.

```
BCD_ke_7:
    orl    Kol_Satuan_skor,#Awal_data_7seg_RAM    ;Nilai satuan+50h(data7segment)
    orl    Kol_Puluhan_skor,#Awal_data_7seg_RAM    ;Nilai puluhan+50h
    orl    Kol_Satuan_waktu,#Awal_data_7seg_RAM    ;Nilai satuan+50h(data7segment)
    orl    Kol_Puluhan_waktu,#Awal_data_7seg_RAM    ;Nilai puluhan+50h
    ret
```

Data di variabel Kol_Satuan_skor, dan Kol_Puluhan_skor akan di-or-kan dengan #Awal_data_7seg_RAM, misalnya Skor_Benar adalah 10, maka angka 0 akan di-or-kan dengan #Awal_data_7seg_RAM dan hasilnya adalah kombinasi 8 angka biner yang nantinya akan menghasilkan tampilan angka 0 pada ruas penampil satuan (11000000b). Angka 1 juga akan di-or-kan dengan #Awal_data_7seg_RAM dan hasilnya adalah kombinasi 8 angka biner yang menghasilkan tampilan angka 1 pada ruas penampil puluhan, yaitu 11111001b. Agar data skor tersebut dapat ditampilkan maka proses selanjutnya adalah panggil Scan_Tampilan.

3.2.6. Pembacaan Tombol *Trainer*

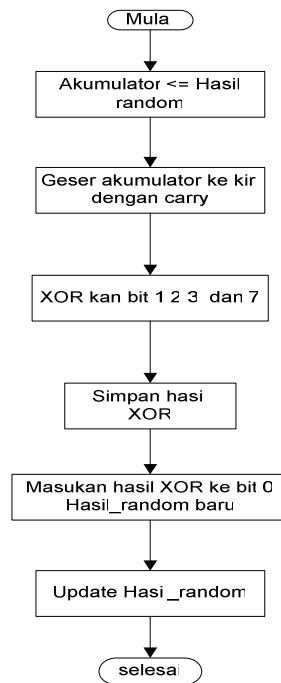
Bagian ini akan membandingkan antara masukan tombol yang ditekan apakah sesuai dengan lampu yang menyala. Jika lampu 1 menyala maka tombol 1 ditekan, demikian juga untuk tombol yang lain. Apabila benar maka skor benar akan bertambah satu, kemudian hasil skor benar akan ditampilkan pada ruas penampil skor. Diagram alir pembacaan tombol *trainer* dapat dilihat pada Gambar 3.22.



Gambar 3.22. Diagram Alir Pembacaan Tombol

3.2.7. Proses *Random*

Proses *random* adalah bagian penting dari sistem *trainer* yang menjadikan *trainer* ini tidak hanya sekedar seperti lampu hias. Pembangkit *random* diterapkan dalam perancangan agar hanya ada sebuah nyala lampu yang dikendalikan oleh mikrokontroler muncul secara acak sehingga nyala lampu berikutnya tidak dapat dihafal oleh pemain. Pembangkit *random* menggunakan operasi LFSR 8 bit yang telah dijelaskan pada bab 2.8 sebelumnya Operasi LFSR 8 bit dilakukan dengan pergeseran 1 bit ke kiri dengan mengikutsertakan *carry*. Diagram alir dapat dilihat pada gambar 3.23



Gambar 3.23. Diagram Alir *Random*

3.2.8. Proses Nyala Lampu

Proses penyalan salah satu lampu sangat tergantung dari data bilangan *hexadecimal* yang disimpan di tabel *look-up* dalam memori ROM mikrokontroler. Data dalam tabel look-up ini dirancang agar hanya dapat menyalakan satu dari 8 lampu dalam *trainer* sehingga 256 data keseluruhan terdiri dari 8 macam data. Urutan 256 data pada tabel *look up* disusun dengan terlebih dahulu mengamati hasil *random* 8 bit, sehingga tidak terjadi penyalan lampu yang sama sesudahnya. Isi tabel *look-up* adalah data 8 *bit* yang dirancang untuk nyala 1 lampu.

: Data lampu yang akan dirandom disimpan pada prom							
:-----							
Data_random_Lampu:							
:00-07		00	01	02	03	04	05 06 07
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH					
:08-0F		08	09	0A	0B	0C	0D 0E 0F
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0FEH					
:10-17		10	11	12	13	14	15 16 17
	DB	0DFH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH					
:18-1F		18	19	1A	1B	1C	1D 1E 1F
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH					
:20-27		20	21	22	23	24	25 26 27
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH					
:28-2F		28	29	2A	2B	2C	2D 2E 2F
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0FDH					
:30-37		30	31	32	33	34	35 36 37
	DB	0F7H,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH					
:38-3F		38	39	3A	3B	3C	3D 3E 3F
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0EFH					
:40-47		40	41	42	43	44	45 46 47
	DB	0BFH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH					
:48-4F		48	49	4A	4B	4C	4D 4E 4F
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0EFH					
:50-57		50	51	52	53	54	55 56 57
	DB	0FBH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH					
:58-5F		58	59	5A	5B	5C	5D 5E 5F
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH					
:60-67		60	61	62	63	64	65 66 67
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH					
:68-6F		68	69	6A	6B	6C	6D 6E 6F
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0BFH					
:70-77		70	71	72	73	74	75 76 77
	DB	07FH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH					
:78-7F		78	79	7A	7B	7C	7D 7E 7F
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0FEH					
:80-87		80	81	82	83	84	85 86 87
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH					
:88-8F		88	89	8A	8B	8C	8D 8E 8F
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0FEH					
:90-97		90	91	92	93	94	95 96 97
	DB	0EFH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH					
:98-9F		98	99	9A	9B	9C	9D 9E 9F
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH					
:A0-A7		A0	A1	A2	A3	A4	A5 A6 A7
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH					
:A8-AF		A8	A9	AA	AB	AC	AD AE AF
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0DFH					
:B0-B7		B0	B1	B2	B3	B4	B5 B6 B7
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH					
:B8-BF		B8	B9	BA	BB	BC	BD BE BF
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0FBH					
:C0-C7		C0	C1	C2	C3	C4	C5 C6 C7
	DB	0EFH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH					
:C8-CF		C8	C9	CA	CB	CC	CD CE CF
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0EFH					
:D0-D7		D0	D1	D2	D3	D4	D5 D6 D7
	DB	07FH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH					
:D8-DF		D8	D9	DA	DB	DC	DD DE DF
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH					
:E0-E7		E0	E1	E2	E3	E4	E5 E6 E7
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH					
:E8-EF		E8	E9	EA	EB	EC	ED EE EF
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0F7H					
:F0-F7		F0	F1	F2	F3	F4	F5 F6 F7
	DB	07FH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH					
:F8-FF		F8	F9	FA	FB	FC	FD FE FF
	DB	0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0F7H					

3.2.9. Data Seven Segment

Data untuk menampilkan *digit* satuan, dan puluhan pada penampil skor, durasi, kecepatan disimpan di tabel *look-up* dalam memori ROM mikrokontroler. Tabel *look-up* berisi data untuk menampilkan angka 0 sampai 9 pada 7 *segment*. Cuplikan programnya sebagai berikut

Data_7segment:

<i>db 11000000b</i>	<i>;data penampil angka 0</i>
<i>db 11111001b</i>	<i>;data penampil angka 1</i>
<i>db 10100100b</i>	<i>;data penampil angka 2</i>
<i>db 10110000b</i>	<i>;data penampil angka 3</i>
<i>db 10011001b</i>	<i>;data penampil angka 4</i>
<i>db 10010010b</i>	<i>;data penampil angka 5</i>
<i>db 10000010b</i>	<i>;data penampil angka 6</i>
<i>db 11111000b</i>	<i>;data penampil angka 7</i>
<i>db 10000000b</i>	<i>;data penampil angka 8</i>
<i>db 10010000b</i>	<i>;data penampil angka 9</i>

BAB IV

PENGAMATAN DAN ANALISA

Pada bab ini akan dijelaskan cara kerja alat secara umum, hasil akhir dari perancangan, pengamatan terhadap : bentuk fisik, tampilan awal, kecepatan *trainer*, durasi *trainer*, penekanan tombol *start*, *trainer*, evaluasi skor, dan *speaker*. Hasil pengamatan merupakan tes uji alat, apakah alat yang sudah dirancang sudah sesuai dengan tujuan penelitian.

Pembahasan selanjutnya adalah pengamatan yang menunjukkan alat ini dapat beroperasi untuk membantu seorang pemain bulutangkis secara mandiri tanpa bantuan instruksi dari pelatih.

4.1 Cara Kerja Alat Secara Umum

Langkah-langkah pengoperasian alat dijelaskan sebagai berikut:

1. Menekan tombol *ON* untuk mengaktifkan catu daya untuk menghidupkan alat.
2. Pemain memilih mode *trainer* dengan menekan tombol mode. LED indikator yang terletak di dalam kotak yang transparan akan terlihat menyala jika ada penekanan. Pada 2 *digit seven segment* bawah dan atas akan ditampilkan mode yang dipilih, meliputi mode kecepatan *trainer* untuk 2 *digit seven segment* atas dan mode durasi *trainer* untuk 2 *digit seven segment* bawah. Jumlah penekanan akan membagi mode menjadi sub-mode. Untuk mode kecepatan *trainer*, jika tombol ini ditekan 1 kali maka akan

ditampilkan sub-mode 3S artinya kecepatan *trainer* adalah 3 detik. Jika 2 kali tekan akan ditampilkan 4S, dan 3 kali tekan akan ditampilkan 5S. Untuk mode durasi, jumlah penekanan tombol akan membagi mode menjadi sub-mode durasi, untuk 1 kali tekan akan ditampilkan angka 30 detik, 2 kali tekan 60 detik, dan 3 kali tekan untuk durasi 90 detik.

3. Setelah mode dipilih, tekan tombol *start*. Penekanan tombol *start* akan menghapus tampilan 2 *digit seven segment* atas dan menggantikan fungsi 2 *digit seven segment* tersebut sebagai penampil skor dengan angka 00 sebagai permulaan. Untuk 2 *digit seven segment* bawah akan tetap menampilkan durasi waktu.
4. Setelah tombol *start* ditekan, ke 8 lampu menyala 2 detik kemudian kedelapan lampu akan mati secara bergeser dengan *delay* 1 detik. Tunda selama 8 detik memberi waktu untuk pemain bergerak menuju tengah lapangan dan berkonsentrasi. Setelah siap akan terdengar suara *speaker* selama 2 detik dan pelatihan dimulai.
5. Pemain melakukan penekanan tombol *trainer*, jika posisi tombol yang ditekan sesuai dengan posisi lampu yang menyala, maka skor keberhasilan bertambah satu.
6. Pelatihan akan berakhir jika waktu durasi *trainer* sudah menampilkan angka 00 dan terdengar bunyi *speaker* selama 5 detik.
7. Untuk skor hasil *trainer* dapat dilihat pada penampil skor pada *seven segment*. 2 *digit seven segment* atas untuk skor benar dan 2 *digit seven segment* bawah untuk skor salah.

8. Untuk mengulangi pelatihan atau berganti *mode trainer* tekan tombol *reset* atau tekan tombol ON/OFF. Kemudian menuju Langkah ke 2.

4.2 Pengamatan dan Analisa

Bentuk akhir perancangan *circuit trainer* ini dapat dilihat pada lampiran.

Bagian-bagian alat yang tampak dari luar ditunjukkan pada tabel 4.1.

Tabel 4.1. Bagian-bagian Alat Tampak dari Luar

No.	Panel sistem	Komponen pembentuk
1.	Penampil mode durasi waktu / Skor benar	2 digit <i>seven segment</i> bawah
2.	Penampil mode Kecepatan / Skor salah	2 digit <i>seven segment</i> atas
3.	<i>Mode trainer</i> dan indikatornya 1. Mode Kecepatan 2. Mode Durasi	2 tombol dan 2 LED
4.	Tombol “START”	1 tombol
5.	<i>Trainer</i>	8 tombol dan 8 Lampu
6.	Pembangkit suara	<i>Speaker</i> 0,5 Watt; 8 Ω
7.	Tiang Penyangga 8 Lampu	Tiang Kayu
8.	Aktifasi catu dan <i>reset</i>	Saklar

4.2.1 Pengamatan Tampilan Awal

Pengamatan tampilan awal dilakukan untuk mengecek kondisi penampil setelah tombol ON ditekan dalam kondisi kedua tombol mode tersebut belum ditekan. Tabel pengamatan kondisi tampilan awal ditunjukkan pada tabel 4.2.

Tabel 4.2. Data Pengamatan Kondisi Tampilan Awal

No.	Panel	Hasil Tampilan
1.	Penampil Mode Kecepatan Mode 3S Mode 4S Mode 5S	Kosong Kosong Kosong
2.	Penampil Mode Durasi Mode 30 Mode 60 Mode 90	Kosong Kosong Kosong

Dari pengamatan dapat disimpulkan sebelum terjadi penekanan tombol mode, ke dua penampil mode terlihat kosong.

4.2.2 Pengamatan Penekanan Tombol Mode

Pengamatan proses pemilihan mode dilakukan dengan melihat kondisi yang terjadi pada penampil setelah menekan tombol mode. Pengamatan dilakukan sesuai jumlah mode yang ada untuk memperoleh data dan diperoleh hasil pengamatan yang ditunjukkan pada tabel 4.3.

Tabel 4.3. Data Pengamatan Penekanan Tombol Mode

No.	Tombol mode	Jumlah Tekan Tombol			Hasil Penekanan Tombol		
1.	Durasi	1x /4x/7x ...	2x/ 5x/8x ...	3x/6x/9x ...	30	60	90
2.	Kecepatan	1x /4x /7x ...	2x/ 5x/8x ...	3x/6x/9x ...	3S	4S	5S

Jika terjadi jumlah penekanan melebihi 3 kali akan menghasilkan perulangan data yang tertampil. Jika 1 kali penekanan pada mode kecepatan menghasilkan data 3S maka jika tombol mode tersebut ditekan 3 kali lagi maka

akan kembali menampilkan data 3S. Hal ini berlaku juga untuk mode durasi, namun data yang ditampilkan berbeda.

Dengan *trainer* ini pemain juga dapat menentukan kombinasi mode yang diperoleh dari kombinasi penekanan tombol mode. Dengan kombinasi mode ini memungkinkan pemain memilih secara bebas mode durasi dan mode kecepatan yang diinginkan. Kombinasi data yang dihasilkan dapat dilihat pada tabel 4.4.

Tabel 4.4. Data Pengamatan Penekanan Kombinasi Mode

No	Jumlah Tekan Mode Durasi	Jumlah Tekan Mode Kecepatan	Hasil	
			Durasi	Kecepatan
1	1x /4x/7x ...	1x /4x/7x ...	30	3S
2	1x /4x/7x ...	2x/ 5x/8x ...	30	4S
3	1x /4x/7x ...	3x/6x/9x ...	30	5S
4	2x/ 5x/8x ...	1x /4x/7x ...	60	3S
5	2x/ 5x/8x ...	2x/ 5x/8x ...	60	4S
6	2x/ 5x/8x ...	3x/6x/9x ...	60	5S
7	3x/6x/9x ...	1x /4x/7x ...	90	3S
8	3x/6x/9x ...	2x/ 5x/8x ...	90	4S
9	3x/6x/9x ...	3x/6x/9x ...	90	5S

4.2.3 Penekanan Tombol Start

Pengamatan bagian tombol start ini diperlukan agar memastikan proses *random* berjalan dengan baik.

Tabel 4.5. Data Pengamatan Proses Penekanan Tombol “START”

No.	Kondisi tombol “START”	Proses
1.	Belum ditekan	Lampu mati dan tampilan 7 segment sesuai mode yang dipilih
2.	Sudah ditekan, belum dilepas	8 Lampu menyala serempak
3.	Sudah ditekan kemudian dilepas	Lampu mati secara bergeser dengan <i>delay</i> waktu 1 detik

4.2.4 Pengamatan Waktu Durasi

Pengamatan waktu Durasi dilakukan dengan membandingkan waktu pada alat yang ditampilkan pada 2 buah 7 *segment* yang digunakan sebagai penampil waktu *trainer* dengan waktu pada *stopwatch*, dengan ketelitian 2 digit dibelakang koma. Pengamatan ini dilakukan untuk mengetahui apakah penampil waktu durasi telah berjalan sesuai dengan yang diinginkan, yaitu sebagai pencacah turun selama 30, 60 atau 90 detik. Pengamatan dilakukan mulai pada saat penampil menunjukkan angka 89 sampai 00, atau 59 sampai 00, dan atau 29 sampai 00. Tabel 4.6 adalah hasil pengamatan. Pengamatan dilakukan sebanyak sepuluh kali.

Tabel 4.6. Data Pengamatan Waktu Durasi 30

Durasi yang diinginkan (Detik)	Waktu pada <i>stopwatch</i> (Detik)	Selisih waktu (Detik)
00 : 30.00	00:30.03	0,03
00 : 30.00	00:30.23	0,23
00 : 30.00	00:30.21	0,21
00 : 30.00	00:30.12	0,12
00 : 30.00	00:30.22	0,22
00 : 30.00	00:30.11	0,11
00 : 30.00	00:30.09	0,09
00 : 30.00	00:29.98	0,02
00 : 30.00	00:30.09	0,09
00 : 30.00	00:29.98	0,02

Tabel 4.7. Data Pengamatan Waktu Durasi 60

Durasi yang diinginkan (Detik)	Waktu pada <i>stopwatch</i> (Detik)	Selisih waktu (Detik)
00 : 60.00	00:60.14	0,14
00 : 60.00	00:59.96	0,04
00 : 60.00	00:60.10	0,10
00 : 60.00	00:60.16	0,16
00 : 60.00	00:60.20	0,20
00 : 60.00	00:60.08	0,08
00 : 60.00	00:60.11	0,11
00 : 60.00	00:60.14	0,14
00 : 60.00	00:60.13	0,13
00 : 60.00	00:59.97	0,03

Tabel 4.8. Data Pengamatan Waktu Durasi 90

Waktu yang diinginkan (Detik)	Waktu pada <i>stopwatch</i> (Detik)	Selisih waktu (detik)
00 : 90.00	00:89.95	0,05
00 : 90.00	00:89.92	0,08
00 : 90.00	00:90.07	0,07
00 : 90.00	00:90.11	0,11
00 : 90.00	00:90.07	0,07
00 : 90.00	00:90.15	0,15
00 : 90.00	00:90.17	0,17
00 : 90.00	00:90.07	0,07
00 : 90.00	00:90.12	0,12
00 : 90.00	00:89.90	0,10

Nilai selisih waktu merupakan selisih waktu pada *stopwatch* dengan waktu yang diinginkan pada perancangan. Nilai rata-rata dari selisih waktu durasi adalah:

1. Mode Durasi 30 detik :

$$= 0,114 \text{ detik}$$

2. Mode Durasi 60 detik:

$$= 0,113 \text{ detik}$$

3. Mode Durasi 90 detik:

$$= 0,099 \text{ detik}$$

Persentase nilai rata-rata selisih waktu terhadap waktu yang diinginkan yaitu:

1. 30 detik :

$$\text{Persentase } error = (0,114 / 30) \times 100\%$$

$$\text{Persentase } error = 0,38 \%$$

2. 60 detik:

$$\text{Persentase } error = (0,113 / 60) \times 100\%$$

$$\text{Persentase } error = 0,188 \%$$

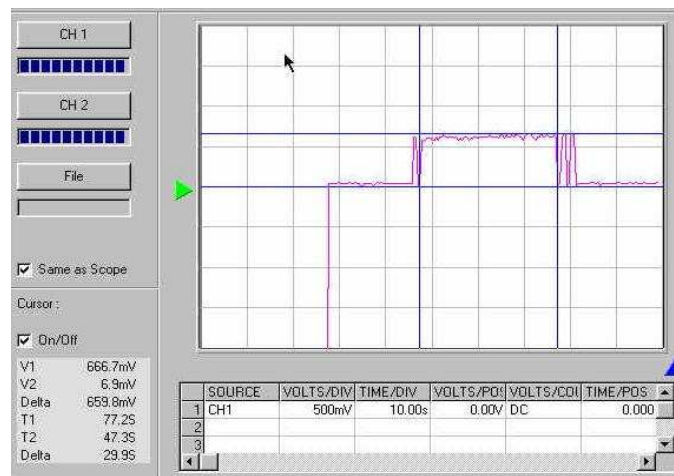
3. 90 detik:

$$\text{Persentase } error = (0,099 / 90) \times 100\%$$

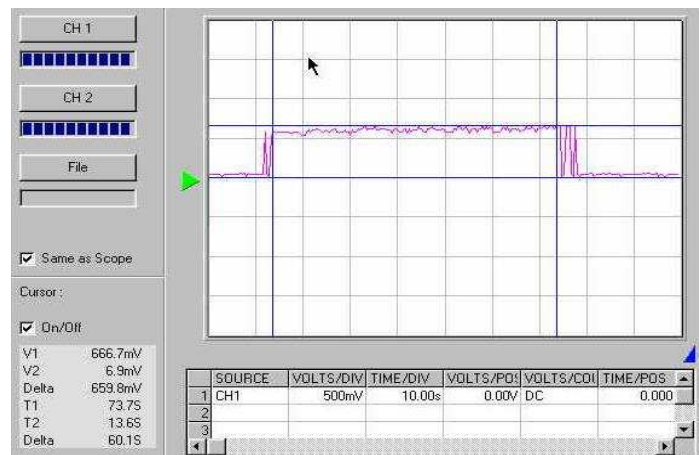
$$\text{Persentase } error = 0,11 \%$$

Error dapat terjadi karena penekanan *stopwatch* yang tidak tepat bersamaan dengan mulainya cacahan penampil durasi waktu *trainer*. Nilai *error* sebesar 0,114; 0,113; 0,099 (detik) dapat ditoleransi dalam *trainer* karena nilai tersebut relatif lebih kecil dari waktu penyalan lampu paling kecil yaitu mode 3S sebesar 3 detik.

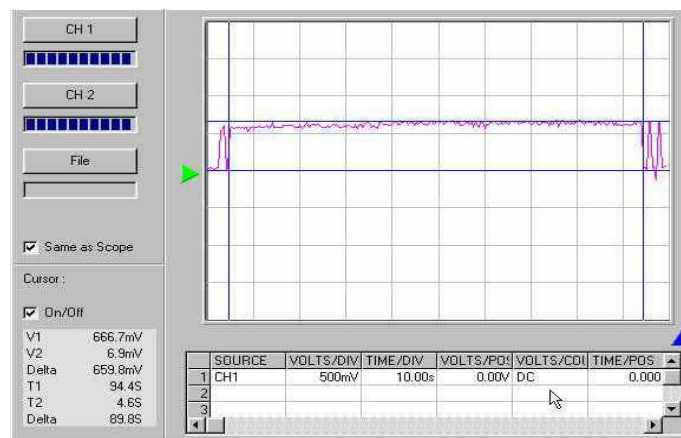
Selain dengan cara *manual* yaitu membandingkan dengan *stopwatch*, digunakan osiloskop digital sebagai data pembanding. Hasil pengamatan dapat dilihat pada Gambar 4.1 sampai Gambar 4.3. Terdapat selisih waktu yang cukup kecil antara perancangan dengan pengamatan menggunakan osiloskop digital. Untuk 30 dan 60 detik selisih 0,1 detik. Untuk 90 detik terjadi selisih 0,2 detik. Nilai selisih untuk tiap mode durasi tersebut masih dapat ditoleransi karena dibawah 1 detik. Untuk durasi 30 detik *delta* bernilai 29.9S , artinya durasi yang dihasilkan adalah 29,9 detik. Durasi 60, *delta* bernilai 60.1S, sedangkan untuk durasi 90 diperoleh *delta* 89.8S.



Gambar 4.1. Durasi 30 detik



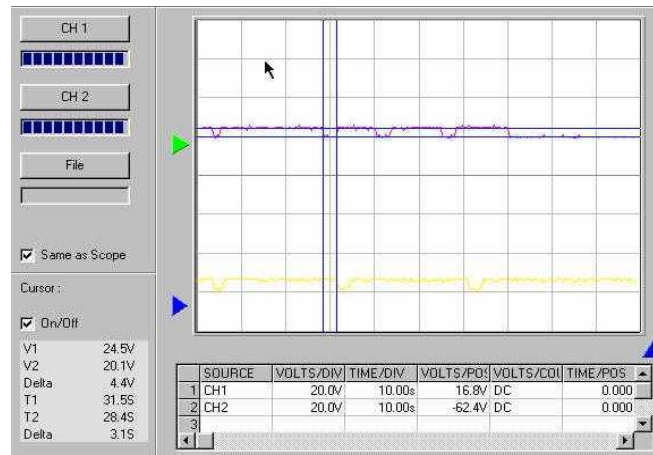
Gambar 4.2. Durasi 60 detik



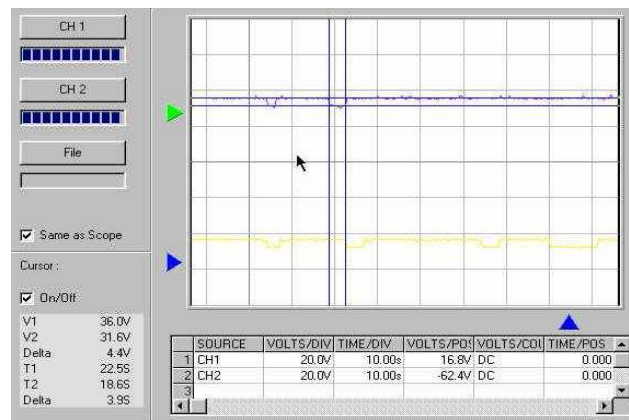
Gambar 4.3. Durasi 90 detik

4.2.5 Pengamatan Waktu Kecepatan

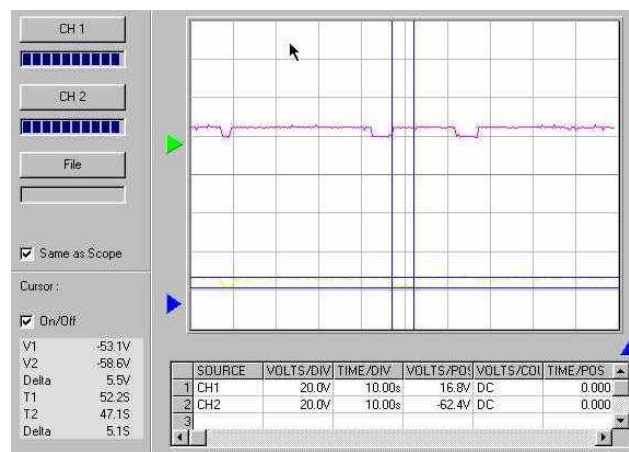
Pengamatan ini dilakukan untuk memastikan bahwa ketika *trainer* bekerja pada beberapa mode kecepatan diperoleh waktu kecepatan yang tepat. Untuk melakukan pengamatan digunakan piranti osiloskop digital. Hasil pengamatan dapat dilihat pada gambar 4.4, gambar 4.5, dan gambar 4.6.



Gambar 4.4. Kecepatan 3 detik



Gambar 4.5. Kecepatan 4 detik



Gambar 4.6. Kecepatan 5 detik

Dari hasil pengamatan diperoleh bahwa terjadi selisih sebesar 0,1 detik pada setiap mode kecepatan antara perancangan dan pengamatan hasil perancangan, data ini diperoleh dengan melihat besar nilai *delta* terhadap data perancangan.. Sehingga akan diperoleh galat sebesar :

$$\text{Galat (\%)} = \frac{\text{Data perancangan} - \text{Data pengamatan}}{\text{Data perancangan}} \times 100\%$$

1. Mode 3 detik :

$$= \frac{3-3,1}{3} \times 100\% = 3,33 \%$$

2. Mode 4 detik :

$$= \frac{4-3,9}{4} \times 100\% = 3,33 \%$$

3. Mode 5 detik :

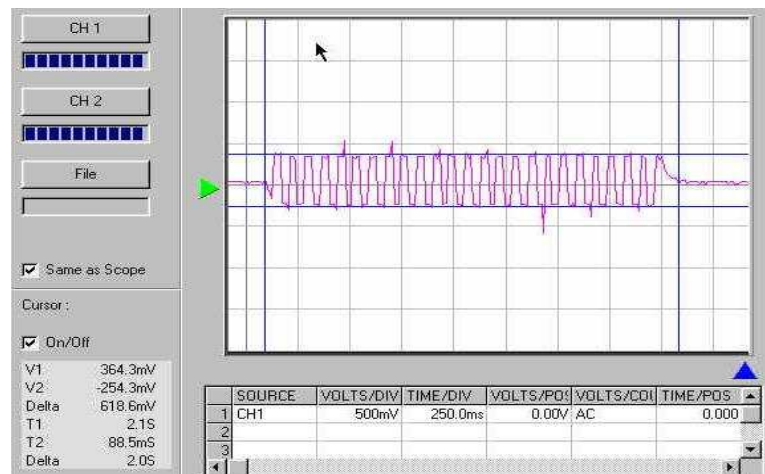
$$= \frac{5-5,1}{5} \times 100\% = 3,33\%$$

Meskipun terjadi kesalahan, namun galat yang terjadi masih dapat diabaikan. Waktu sebesar 0,1 detik terlalu kecil dan tidak terlalu berpengaruh terhadap mode kecepatan *trainer*. Perbedaan selisih waktu ini tidak dapat dirasakan oleh pemain ketika menggunakan *trainer*.

Channel 1 dan 2 pada osiloskop digunakan untuk mengamati waktu nyala dua buah lampu, diambil *sample* yaitu lampu 1 dan lampu 2. Metode ini ditempuh untuk mengamati proses pergantian nyala antar lampu. Dari pengamatan tampak bahwa setelah 3 atau 4 , dan 5 detik lampu 1 menyala akan langsung dilanjutkan lampu ke 2.

4.2.6 Pengamatan Tone *Speaker* Mulai 2 Detik

Pengamatan ini dilakukan agar dapat dipastikan *speaker* berbunyi dengan periode waktu 2 detik ketika *trainer* akan dijalankan. Gambar hasil pengamatan dapat dilihat pada gambar 4.7.

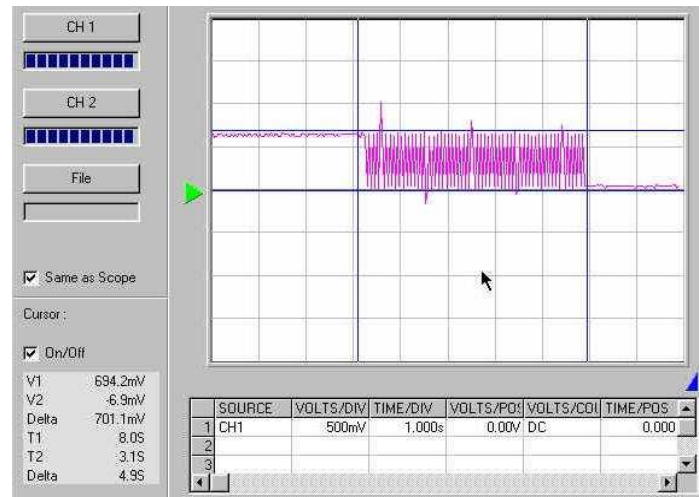


Gambar 4.7 *Speaker* Berbunyi Selama 2 detik

Pada gambar terlihat *delta* bernilai 2.0S, artinya tunda waktu yang dihasilkan ketika *speaker* berbunyi adalah 2 detik. Sehingga hasil pengamatan sesuai dengan perancangan .

4.2.7 Pengamatan Tone *Speaker* Selesai 5 Detik

Untuk mengakhiri *trainer* maka dibutuhkan tanda berupa bunyi *speaker* selama 5 detik. Dari hasil pengamatan diperoleh data hasil perancangan periode waktu *speaker* berbunyi. *Delta* memiliki nilai 4.9S, artinya tunda waktu *speaker* berbunyi adalah 4,9 detik.



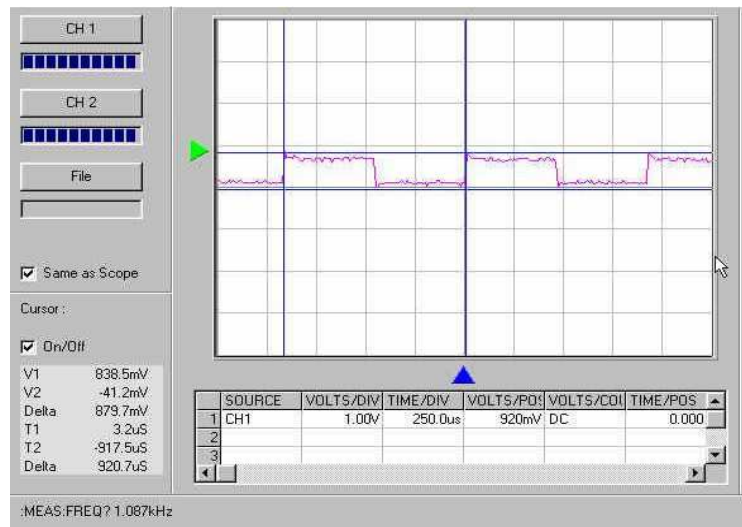
Gambar 4.8 *Speaker* Berbunyi Selama 5 detik

Dari hasil pengamatan diperoleh selisih waktu sebesar 0,1 detik ketika *speaker* berbunyi selama periode 5 detik.

4.2.8 Pengamatan Frekuensi Bunyi *Speaker*

Pengamatan bunyi *speaker* dilakukan dengan menggunakan osiloskop digital untuk mengetahui frekuensi bunyi yang dibangkitkan, yaitu 1 KHz. Data hasil pengamatan yang ditampilkan dengan menggunakan osiloskop digital ditunjukkan pada gambar 4.9.

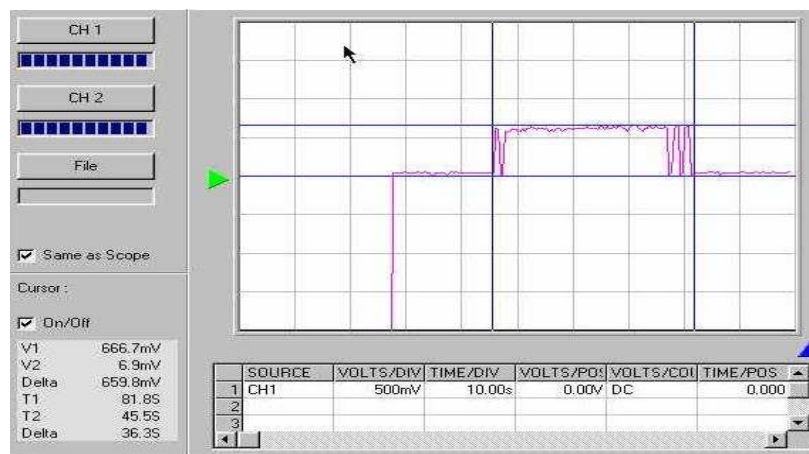
Hasil pengamatan menunjukkan frekuensi yang dibangkitkan *speaker* sebesar 1,087 KHz ditunjukkan pada nilai *delta* T (periode) pada 1 periode gelombang (gambar 4.9) , meskipun terdapat selisih angka 0,087 KHz itu dapat diabaikan sehingga *speaker* bekerja sesuai dengan perancangan yang diinginkan.



Gambar 4.9. Hasil Pengamatan Periode *Speaker*

4.2.9 Pengamatan *Trainer* Bekerja Pada Mode 3S / 30 detik

Pengamatan dilakukan agar dapat diketahui total waktu yang dihasilkan ketika *trainer* bekerja pada salah satu pilihan mode. Hasil pengamatan terdapat pada gambar 4.10.

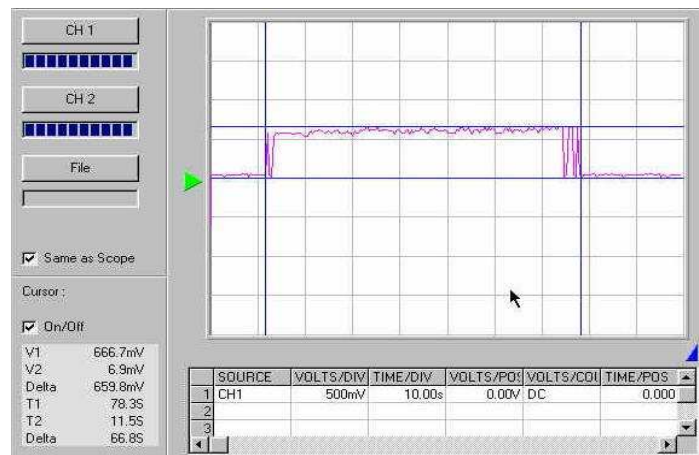


Gambar 4.10 Periode *Trainer* bekerja pada Mode 3S/30 detik

Dari pengamatan diperoleh waktu total yang dihasilkan ketika *speaker* tanda mulai berbunyi (2 detik), diteruskan dengan durasi *trainer* (30 detik), dan diakhiri dengan *speaker* tanda selesai (5 detik) adalah 36,3 detik. Pada perancangan secara matematis akan dihasilkan total waktu sebesar 37 detik. Sehingga dapat diketahui adanya selisih waktu sebesar 0,7 detik.

4.2.10 Pengamatan *Trainer* Bekerja Pada Mode 3S / 60 detik

Hasil pengamatan pada mode 3S / 60 detik dapat dilihat pada gambar 4.11.



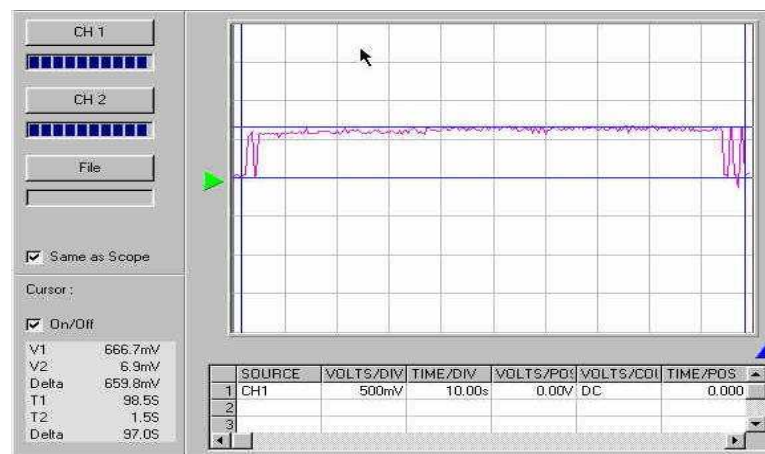
Gambar 4.11 Periode *Trainer* Bekerja pada Mode 3S/60 detik

Total waktu yang dirancang adalah 67 detik. Hasil pengamatan diperoleh adalah 66,8 detik. Sehingga diperoleh selisih waktu antara perancangan dan pengamatan adalah 0,2 detik.

4.2.11 Pengamatan *Trainer* Bekerja Pada Mode 3S / 90 detik

Hasil pengamatan pada mode 3S / 90 detik dapat dilihat pada gambar

4.12.



Gambar 4.12 Periode *Trainer* Bekerja pada Mode 3S/90 detik

Total waktu yang dirancang adalah 97 detik. Hasil pengamatan diperoleh adalah 97,0 detik. Sehingga tidak ada selisih waktu antara perancangan dan pengamatan.

4.2.12 Pengamatan Penekanan Tombol *Trainer*

Pengamatan penekanan tombol dilakukan untuk mengetahui hasil dari proses penekanan tombol.

Pada saat *trainer* berjalan dan proses penekanan tombol dilakukan, diperoleh hasil berupa skor benar. Saat *trainer*, tombol tidak ditekan atau yang ditekan salah maka skor kegagalan akan bertambah. Penekanan tombol ini tidak berpengaruh pada waktu kecepatan *trainer* dan durasi. Durasi *trainer* tetap berjalan mundur sampai durasi habis walaupun tombol ditekan, tidak ditekan

atau salah tekan. Ketika terjadi kesalahan penekanan tombol, pemain dapat melakukan koreksi dengan penekanan kembali terhadap tombol yang seharusnya ditekan. Hal ini dapat dilakukan dengan syarat waktu masih memungkinkan untuk melakukan koreksi. Jika pemain terlambat melakukan koreksi ketika lampu selanjutnya menyala maka kasus tersebut dianggap sebagai kesalahan penekanan dan skor salah akan bertambah satu.

Saat pengamatan, skor benar ditampilkan pada dua digit ruas penampil bagian atas. Namun untuk skor kegagalan ditampilkan setelah trainer selesai di 2 digit bawah ruas penampil. Hal ini diakibatkan saat *trainer* berjalan tampilan 2 digit ruas bawah digunakan sebagai tampilan durasi.

4.2.13 Pengamatan Penyalaan Acak Lampu *Trainer*

Untuk memastikan bahwa lampu menyala secara acak, maka pengamatan untuk 9 kombinasi tipe mode dilakukan dengan 5 kali pengambilan data. Pengambilan data dilakukan dengan cara mencatat posisi nyala lampu.

1. Mode 3S / 30 detik :

(No : Pengamatan ke n, $n : 1 \Rightarrow 5$)

Tabel 4.9. Data Pengamatan Mode 3S / 30 detik

No	Posisi Nyala Lampu	Jumlah Penyalaan Lampu
1	2,4,8,7,6,4,7,5,2,3	10
2	2,3,6,4,8,7,6,3,6,4	10
3	7,6,3,5,2,3,5,1,2,4	10
4	4,8,7,6,4,7,5,2,3,5	10
5	5,1,2,3,6,3,6,4,7,5	10

2. Mode 3S/ 60 detik :

Tabel 4.10. Data Pengamatan Mode 3S / 60 detik

No	Posisi Nyala Lampu	Jumlah Nyala Lampu
1	8,7,6,3,5,2,3,5,1,2,4,8,7,5,1,2,3,5,1,6	20
2	4,8,7,6,4,7,5,2,3,5,2,3,5,2,4,7,5,1,4,1	20
3	7,6,3,5,2,3,5,1,2,4,8,7,5,1,2,3,5,1,6,1	20
4	3,6,4,7,5,1,2,4,8,1,7,6,3,5,1,8,2,4,8,1	20
5	2,4,8,7,6,4,7,5,2,3,5,2,3,5,2,4,7,5,1,4	20

3. Mode 3S/ 90 detik :

Tabel 4.11. Data Pengamatan Mode 3S / 90 detik

No	Posisi Nyala Lampu	Jumlah Penyalaan Lampu
1	3,6,4,7,5,1,2,4,8,1,7,6,3,5,1, 8,2,4,8,1,8,5,1,4,7,5,2,3,5,1	30
2	2,4,8,7,6,4,7,5,2,3,5,2,3,5,2, 4,7,5,1,4,1,5,2,4,8,7,6,3,5,2	30
3	7,6,3,5,2,3,5,1,2,4,8,7,5,1,2, 3,5,1,6,1,7,1,2,3,6,4,7,5,1,2	30
4	6,4,7,5,1,2,4,8,1,7,6,3,5,1,8, 2,4,8,1,8,5,1,4,7,5,2,3,5,1,5	30
5	5,2,4,8,5,8,7,6,3,6,3,6,3,6,4, 8,7,5,1,8,1,2,4,7,5,1,2,3,6,3	30

4. Mode 4S/ 30 detik :

Tabel 4.12. Data Pengamatan Mode 4S / 30 detik

No	Posisi nyala lampu	Jumlah Penyalaan Lampu
1	2,3,6,4,8,7,6,3	8
2	2,3,6,4,8,7,6,3	8
3	7,6,3,5,2,3,5,1	8
4	5,2,4,8,5,8,7,6	8
5	6,4,7,5,1,2,4,8	8

5. Mode 4S/ 60 detik :

Tabel 4.13. Data Pengamatan Mode 4S / 60 detik

No	Posisi nyala lampu	Jumlah Penyalaan Lampu
1	8,7,6,3,5,2,3,5,1,2,4,8,7,5,1	15
2	4,8,7,6,4,7,5,2,3,5,2,3,5,2,4	15
3	7,6,3,5,2,3,5,1,2,4,8,7,5,1,2	15
4	2,3,6,4,8,7,6,3,6,4,8,6,7,6,4	15
5	5,1,2,3,6,3,6,4,7,5,2,4,7,5,2	15

6. Mode 4S/ 90 detik :

Tabel 4.14. Data Pengamatan Mode 4S / 90 detik

No	Posisi nyala lampu	Jumlah Penyalaan Lampu
1	8,7,6,3,5,2,3,5,1,2,4,8,7,5,1,2,3,5,1,6,1,7,1	23
2	4,8,7,6,4,7,5,2,3,5,2,3,5,2,4,7,5,1,4,1,5,2,4	23
3	5,1,2,3,6,3,6,4,7,5,2,4,7,5,2,3,6,4,8,2,8,3,7	23
4	5,1,2,3,6,3,6,4,7,5,2,4,7,5,2,3,6,4,8,2,8,3,7	23
5	2,4,8,7,6,4,7,5,2,3,5,2,3,5,2,4,7,5,1,4,1,5,2	23

7. Mode 5S/ 30 detik :

Tabel 4.15. Data Pengamatan Mode 5S / 30 detik

No	Posisi nyala lampu	Jumlah Penyalaan Lampu
1	4,7,5,1,2,4	6
2	2,4,8,7,6,4	6
3	8,7,6,3,5,2	6
4	4,8,7,6,4,7	6
5	7,6,3,5,2,3	6

8. Mode 5S/ 60 detik :

Tabel 4.16. Data Pengamatan Mode 5S / 60 detik

No	Posisi nyala lampu	Jumlah Penyalaan Lampu
1	8,7,6,3,5,2,3,5,1,2,4,8	12
2	4,8,7,6,4,7,5,2,3,5,2,3	12
3	7,6,3,5,2,3,5,1,2,4,8,7	12
4	3,6,4,7,5,1,2,4,8,1,7,6	12
5	5,1,2,3,6,3,6,4,7,5,2,4	12

9. Mode 5S/ 90 detik :

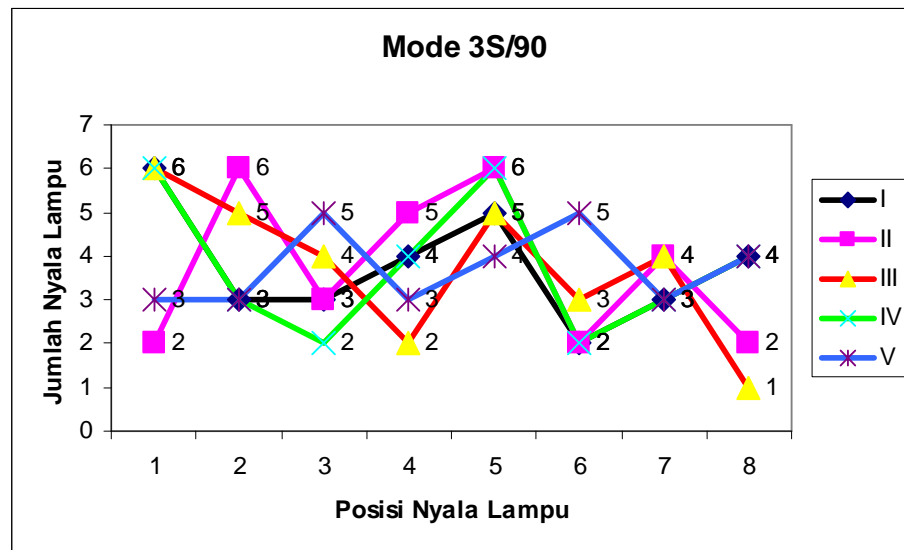
Tabel 4.17. Data Pengamatan Mode 5S / 90 detik

No	Posisi nyala lampu	Jumlah Penyalaan Lampu
1	8,7,6,3,5,2,3,5,1,2,4,8,7,5,1,2,3,5	18
2	2,3,6,4,8,7,6,3,6,4,8,6,7,6,4,7,6,4	18
3	5,2,4,8,5,8,7,6,3,6,3,6,3,6,4,8,7,5	18
4	3,6,4,7,5,1,2,4,8,1,7,6,3,5,1,8,2,4	18
5	4,7,5,1,2,4,8,1,7,6,3,5,1,8,2,4,8,1	18

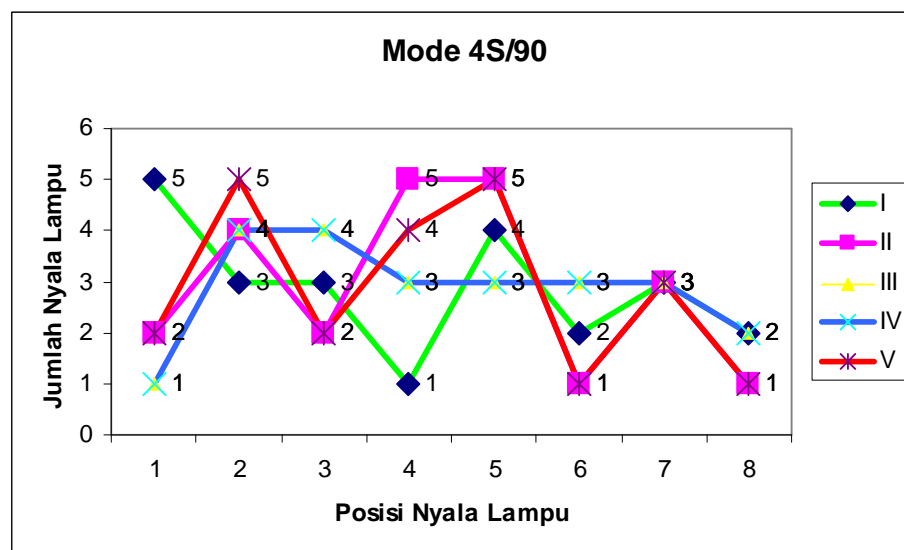
Pengamatan *random* nyala lampu juga memperhatikan distribusi atau sebaran data penyalaan lampu. *Trainer* menjadi lebih baik jika mempunyai sebaran *data random* yang seragam atau tiap lampu mempunyai peluang yang sama untuk menyala. Grafik *distribusi* nyala lampu diambil dari beberapa pengamatan pada 3 mode pilihan, yaitu mode 3S/90 detik, mode 4S/90 detik, dan mode 5S/90 detik.

Pada Gambar 4.13, 4.14, dan 4.15 adalah hasil pengamatan yang dilakukan berdasarkan acuan data pada Tabel 4.11, 4.14, dan 4.17 pengamatan mode. Terlihat bahwa sebaran data posisi nyala lampu tidak seragam untuk 5 kali pengamatan pada salah satu kombinasi mode. Garis dengan warna yang

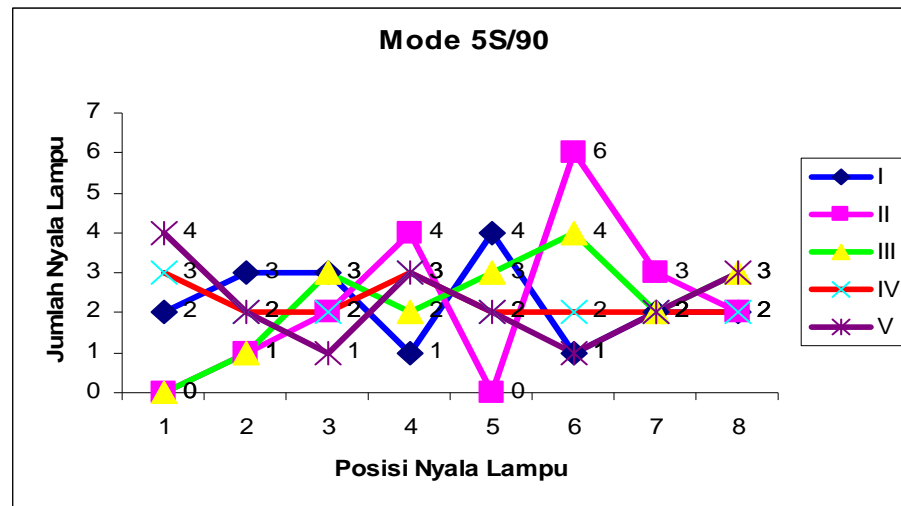
berbeda pada gambar 4.13 sampai 4.15 adalah pengamatan ke-n. Pengamatan ke-n artinya pengamatan dari pengamatan I sampai V.



Gambar 4.13 Grafik Pengamatan Distribusi Nyala Lampu
Mode 3S / 90 detik



Gambar 4.14 Grafik Pengamatan Distribusi Nyala Lampu
mode 4S / 90 detik



**Gambar 4.15 Grafik Pengamatan Distribusi Nyala Lampu
mode 5S / 90 detik**

4.2.14. Pengamatan Akhir *Trainer*

Pengamatan akhir *trainer* dilakukan dengan mengamati kondisi akhir setelah proses *trainer* selesai. Hasil pengamatan ditunjukkan pada tabel 4.18 berikut ini:

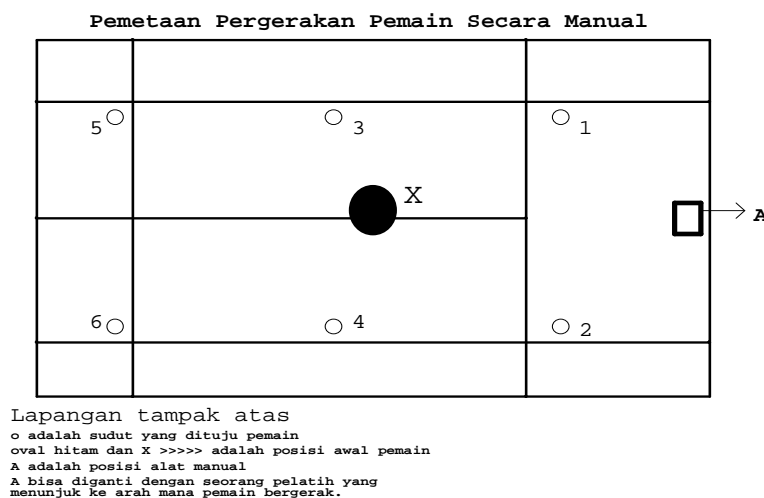
Tabel 4.18. Data Pengamatan Akhir Selesai *Trainer*

No.	Panel	Proses
1.	<i>Speaker</i>	Berbunyi selama 5 detik
2.	Ruas Penampil Durasi	00
3.	Penampil skor benar	Menampilkan hasil skor benar pada 2 <i>digit</i> atas ruas penampil
4.	Penampil skor salah	Setelah Durasi 00, skor salah akan ditampilkan pada ruas penampil Durasi bawah
4.	8 lampu trainer	Akan berhenti pada salah 1 titik lampu

4.2.15. Pengamatan Penggunaan *Trainer*

Pengamatan penggunaan *trainer* ini dibutuhkan untuk melihat fungsi *trainer* ini sebagai *trainer* yang bersifat mandiri. Metode yang digunakan dengan membandingkan antara cara berlatih secara *manual* dan secara otomatis dengan menggunakan alat *badminton trainer*. Kemudian dilihat perbandingan antara kedua cara tersebut.

Secara *manual*, proses berlatih adalah demikian. Pemain akan bergerak sesuai dengan perintah pelatih yang menggunakan tombol masukan untuk menyalakan 6 lampu AC menuju ke 6 titik sudut lapangan. Posisi nyala lampu sesuai dengan keinginan pelatih dengan kecepatan yang tidak menentu. Durasi yang biasa digunakan adalah 30, 60, dan 90 detik. Tidak terdapat papan evaluasi skor. Gambar 4.16. adalah pemetaan pergerakan pemain secara *manual*.



**Gambar 4.16 Pemetaan Pergerakan Pemain dengan
Cara Manual**

Parameter yang digunakan untuk melihat kelebihan dari *trainer* yang dirancang adalah pemilihan mode *trainer* yang bertahap. Terdapat 9 kombinasi

mode yang akan membantu pemain berlatih secara mandiri mulai dari mode mudah 5S/30 detik sampai mode paling cepat dan paling lama 3S/90 detik. Parameter lainnya adalah adanya papan evaluasi skor yang akan memperlihatkan hasil pemain dalam berlatih bergerak secara acak dalam kecepatan yang konstan. Jika pada pengamatan terlihat bahwa skor kegagalan semakin kecil atau skor benar mendekati maksimum dan durasi yang dipilih adalah 3S/90 detik maka mengindikasikan terjadi peningkatan ketahanan fisik pemain dan kecepatan respon pemain itu sendiri. Dengan melihat sendiri hasil skor yang diperoleh, pemain akan dapat mengetahui seberapa hasil yang diperoleh. Parameter selanjutnya adalah jumlah titik yang akan dituju pemain untuk bergerak menjadi 8 titik. Dengan jumlah titik yang lebih banyak maka jangkauan pergerakan pemain juga menjadi lebih luas. Untuk lebih jelasnya dapat dilihat pada tabel 4.19.

Tabel 4.19 Perbandingan antara Kedua Alat, Secara *Manual* dan Secara Otomatis.

No	Parameter	<i>Manual</i>	Otomatis dengan <i>Trainer</i>
1	Durasi	30/60/90 detik	30/60/90 detik
2	Interval Penyalaan Lampu	Tidak tentu	3S/4S/5S
3	Sistem Penyalaan	Sesuai perintah pelatih / operator atau dengan tombol	Acak/ Random
4	Evaluasi Skor	Tidak ada	Skor benar Skor salah
5	Masukan tombol pemain	Tidak ada	Ada 8 buah tombol
6	Jumlah posisi arah gerakan	6 titik	8 titik
7	Jumlah Pergerakan	Tidak tentu	Tertentu
8	Operator/ Pelatih	Harus ada	Tidak perlu

Data-data pengamatan yang diambil diperoleh dari 3 pemain yang bersedia untuk melakukan proses *training*. Pengambilan data tidak hanya dilakukan pada 1 kesempatan tetapi perlu beberapa kali pengambilan data di hari yang berbeda. Hal ini dilakukan untuk dapat melihat perkembangan pemain itu sendiri. Hasil pengamatan ini juga sangat dipengaruhi oleh kondisi pemain saat pengamatan dilakukan.

Setelah dilakukan pengamatan ada beberapa hal yang dapat disimpulkan. Saat pertama kali pemain melakukan latihan dengan *trainer*, mereka membutuhkan adaptasi terhadap alat. Kesalahan terjadi karena pemain kadang mengalami kebingungan tombol mana yang harus ditekan. Ketika mereka sudah hafal posisi lampu dan pasangan tombol yang harus ditekan baru kemudian kesalahan-kesalahan tersebut menjadi semakin jarang terjadi.

Pada saat proses penekanan dilakukan kadang terjadi proses penekanan yang kurang sempurna. Untuk si pemain merasa sudah menekan tombol yang benar tetapi sebetulnya tombol belum ditekan dengan baik sehingga terjadi kesalahan. Hal ini disebabkan dimensi dari tombol yang kurang besar dan pegas pada tombol yang kurang baik karena terlalu sering ditekan. Dimensi tombol *trainer* yang telah dirancang berdiameter 1,5 cm dan diameter mangkok tombol 15 cm.

Faktor fisik menjadi hambatan. Untuk mode dengan kecepatan yang tinggi dan durasi yang lama pemain cenderung kehilangan konsentrasi karena kelelahan fisik yang dialami dan kehabisan nafas. Kesalahan yang biasa terjadi

adalah keterlambatan menekan tombol. Data pengamatan dapat dilihat pada lampiran.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Mikrokontroler AT89S51 dapat digunakan sebagai kendali utama *Circuit Trainer Badminton*. Berdasarkan perancangan dan hasil pengamatan yang diperoleh, maka dapat disimpulkan :

1. Lama penekanan tombol “*START*” yang dilakukan pemain menentukan nyala lampu yang pertama dan nyala lampu berikutnya akan tampil secara acak (*random*) dengan penggunaan metode acak LFSR (*Linear Feedback Shift Register*) 8 bit. Lampu dapat menyala *random*, tetapi tidak terdistribusi seragam.
2. Pada mode Durasi, nilai galat sebesar 0,114; 0,113; 0,099 (detik) dapat ditoleransi dalam *trainer* karena nilai tersebut relatif lebih kecil dari waktu penyalaan lampu paling kecil yaitu mode 3*S* sebesar 3 detik.
3. Untuk Mode 30 detik, selisih waktu yang terjadi adalah 0,1 detik. Mode 60 detik diperoleh juga 0,1 detik. Mode 90 detik diperoleh selisih 0,2 detik.
4. Galat yang dihasilkan untuk masing-masing mode kecepatan adalah:

$$\text{Galat (\%)} = 3,33\%$$

Galat yang terjadi masih dapat diabaikan. Waktu sebesar 0,1 detik terlalu kecil dan tidak terlalu berpengaruh terhadap mode kecepatan

trainer. Perbedaan selisih waktu ini tidak dapat dirasakan oleh pemain ketika menggunakan *trainer*.

5.2 Saran

Alat hasil perancangan ini masih memiliki banyak kekurangan dan keterbatasan pada perangkat keras dan perangkat lunaknya, sehingga penulis mencoba untuk memberikan saran-saran untuk pengembangan lebih lanjut agar menjadi lebih baik, yaitu:

1. Tampilan skor dan waktu berupa *7'segment* akan lebih baik jika digunakan yang berukuran lebih besar.. Ketika *trainer* berjalan pemain dapat langsung melihat durasi waktu yang berjalan dan skor yang dihasilkan.
2. Penggunaan 8 tombol *trainer* yang lebih besar, sehingga mempermudah pemain untuk menekan dan melihat obyek tombol, kesalahan yang diperoleh oleh pemain selain terlambat menekan tombol adalah faktor bentuk tombol yang kurang besar. Disarankan lebih besar dari dimensi tombol yang telah dirancang . Diameter tombol yang telah dirancang 1,5 cm. Akan lebih baik jika diameter tombol berukuran kurang lebih 7 cm.
3. Pilihan mode dapat diperbanyak, sehingga pilihan mode untuk berlatih lebih *variatif*.
4. Sistem menjadi lebih baik jika *random* penyalan lampu terdistribusi seragam dengan memperbaiki isi tabel *look-up* data penyalan lampu.

Tingkat kesulitan akan menjadi lebih tinggi jika nyala lampu tidak dapat dihafalkan oleh pemain.

5. Sistem menjadi lebih realistis dengan perubahan jeda waktu nyala lampu atau kecepatan *trainer* juga dibuat random.
6. Sistem akan menjadi lebih baik apabila dilengkapi dengan catu daya cadangan seperti *battery recharged* untuk menghindari terputusnya catu daya secara tiba-tiba tiba (listik padam) di saat *trainer* dilakukan.

DAFTAR PUSTAKA

- Andi Nalwan, Paulus, 2003, *Panduan Praktis Teknik Antarmuka dan Pemrograman Mikrokontroler AT89C51*, Penerbit Elex Media Komputindo, Jakarta
- Eko Putra, Agfianto, 2002, *Belajar Mikrokontroler AT89C51 / 52 / 55*, Penerbit Gava Media, Yogyakarta
- Malik, Norbert R., 1995, *Electronics Circuit Analysis, Simulation and Design*, Prentice Hall Inc., New Jersey, USA
- Sklar, Bernard, 1988, *Digital Communications Fundamental & Applications*, Prentice Hall Inc., New Jersey, USA
- Stewart, James W. and Miao, Kai X., 1999, *The 8051 Microcontroller : Hardware, Software and Interfacing*, Prentice Hall, Inc., USA
- _____, *Flash Microcontroller : Architectural Overview*, Atmel Inc., <http://www.atmel.com>
- _____, *Pseudo-Random Number Generation Routine For The MAX765X Microprocessor*, Maxim Inc., http://www.Maxim-ic.com/appnotes.cfm/appnotes_number/1743
- _____, *Light Emitting Diodes (LEDs), Transistors*, <http://www.kpsec.freeuk.com/components.htm>
- Terjemahan Ir. Alb. Joko Santoso, M.T. 2003. *Prinsip-Prinsip Elektronika Buku Satu*. Malvino. A. P.: Penerbit: Salemba Teknika.
- Ir Balza Achmad, M.Sc.E, IrAgus Arif, M.T. 2004. *Handbook Microcontroller Application Workshop*. Yogyakarta: Teknik Elektro UGM.

LAMPIRAN

I. Pengamatan Penggunaan *Trainer* :

1. Hari I , Hari Rabu 29 November 2006,

1.a. *Manual*:

Tabel 4.20. Pengamatan Manual Hari I

No	Pemain ke	Durasi	Jumlah Pergerakan	Perubahan Posisi Arah Gerak
1	I	30	9	3,4,5,1,1,2,4,6,5
		60	18	4,1,2,3,4,5,2,1,4, 1,4,3,1,6,5,2,3,5
		90	28	2,3,1,4,2,5,6,1,2,5,3,6,2,4, 6,2,6,5,4,5,4,2,1,3,1,4,1,4
2	II	30	10	4,6,5,1,2,3,6,5,3,6,
		60	19	3,1,3,5,4,2,1,6,4, 2,4,1,3,4,3,6,1,2,4,
		90	29	3,2,4,1,2,6,4,1,2,3,1,5,6,4, 1,2,6,5,3,1,3,4,6,4,2,1,3,4,2
3	III	30	8	2,1,5,1,4,6,2,3
		60	16	2,1,2,3,6,5,2,1, 3,4,3,1,6,5,5,4
		90	27	1,4,1,4,1,5,6,1,2,6,3,6,2,4,3,4, 6,1,2,1,6,5,4,5,4,2,1

1.b. Dengan *Trainer* :

Tabel 4.21. Pengamatan Penggunaan *Trainer*

No	Pemain ke	Pilihan Mode	Skor Benar	Skor Salah	Perubahan Posisi Arah Gerak	Jumlah Pergerakan
1	I	3S/30	6	4	5,1,2,3,6,3,6,4,7,5	10
		3S/60	12	8	2,4,8,7,6,4,7,5,2,3, 5,2,3,5,2,4,7,5,1,4	20
		3S/90	20	10	2,4,8,7,6,4,7,5,2,3,5,2,3,5,2, 4,7,5,1,4,1,5,2,4,8,7,6,3,5,2	30
		4S/30	4	4	2,3,6,4,8,7,6,3	8

		4S/60	9	6	7,6,3,5,2,3,5,1, 2,4,8,7,5,1,2	15
		4S/90	16	7	4,8,7,6,4,7,5,2,3,5,2, 3,5,2,4,7,5,1,4,1,5,2,4	23
		5S/30	5	1	2,4,8,7,6,4	6
		5S/60	10	2	3,6,4,7,5,1,2,4,8,1,7,6	12
		5S/90	15	3	2,3,6,4,8,7,6,3,6,4,8,6,7,6,4,7, 6,4	18
2	II	3S/30	5	5	4,8,7,6,4,7,5,2,3,5	10
		3S/60	14	6	3,6,4,7,5,1,2,4,8,1,7,6,3,5,1,8, 2,4,8,1	20
		3S/90	22	8	5,2,4,8,5,8,7,6,3,6,3,6,3,6,4, 8,7,5,1,8,1,2,4,7,5,1,2,3,6,3	30
		4S/30	5	3	7,6,3,5,2,3,5,1	8
		4S/60	8	7	8,7,6,3,5,2,3,5,1,2,4,8,7,5,1	15
		4S/90	14	9	8,7,6,3,5,2,3,5,1,2,4,8, 7,5,1,2,3,5,1,6,1,7,1	23
		5S/30	6	0	4,7,5,1,2,4	6
		5S/60	10	2	5,1,2,3,6,3,6,4,7,5,2,4	12
		5S/90	14	4	5,2,4,8,5,8,7,6,3, 6,3,6,3,6,4,8,7,5	18
3	III	3S/30	4	6	2,4,8,7,6,4,7,5,2,3	10
		3S/60	13	7	7,6,3,5,2,3,5,1,2,4, 8,7,5,1,2,3,5,1,6,1	20
		3S/90	21	9	6,4,7,5,1,2,4,8,1,7,6,3,5,1,8, 2,4,8,1,8,5,1,4,7,5,2,3,5,1,5	30
		4S/30	5	3	2,3,6,4,8,7,6,3	8
		4S/60	11	4	4,8,7,6,4,7,5,2,3,5,2,3,5,2,4	15
		4S/90	18	5	5,1,2,3,6,3,6,4,7,5,2, 4,7,5,2,3,6,4,8,2,8,3,7	23
		5S/30	5	1	8,7,6,3,5,2	6
		5S/60	9	3	3,6,4,7,5,1,2,4,8,1,7,6	12
		5S/90	16	2	8,7,6,3,5,2,3,5,1,2,4,8,7,5,1,2, 3,5	18

2. Hari II , Hari Kamis 30 November 2006

1.a. *Manual*:

Tabel 4.22. Pengamatan Manual Hari II

No	Pemain ke	Durasi	Jumlah Pergerakan	Perubahan Posisi Arah Gerak
1	I	30	10	4,6,5,3,2,3,4,5,3,5
		60	20	3,4,3,2,1,4,5,2,6,4, 5,1,4,3,1,6,5,3,2,5
		90	27	1,4,3,5,4,1,2,5,4,6,2,4,5 2,6,5,4,3,2,1,3,1,4,2,4,1,6
2	II	30	8	3,1,5,2,4,6,2,6
		60	18	1,4,3,2,6,4,5,3,6, 4,6,1,4,3,1,6,4,3
		90	26	2,4,3,6,4,1,2,5,4,6,2,4,6 2,6,5,4,3,2,6,3,1,4,2,3,1
3	III	30	9	2,3,5,2,4,5,2,6,2
		60	16	3,4,3,2,6,4,5,3, 4,6,5,4,3,2,6,2
		90	24	4,2,3,6,4,1,3,5,4,6,5,4, 3,6,5,4,3,2,6,3,1,4,2,3

1.b. Dengan *Trainer* :

Tabel 4.23. Pengamatan Penggunaan *Trainer*

No	Pemain ke	Pilihan Mode	Skor Benar	Skor Salah	Perubahan Posisi Arah Gerak	Jumlah Pergerakan
1	I	3S/30	7	3	7,6,3,5,2,3,5,1,2,4	10
		3S/60	14	6	7,6,3,5,2,3,5,1,2,4,8,7,5,1,2, 3,5,1,6,1	20
		3S/90	23	7	6,4,7,5,1,2,4,8,1,7,6,3,5,1,8, 2,4,8,1,8,5,1,4,7,5,2,3,5,1,5	30
		4S/30	6	2	6,4,7,5,1,2,4,8	8
		4S/60	12	3	2,3,6,4,8,7,6,3,6,4,8,6,7,6,4	15
		4S/90	19	4	5,1,2,3,6,3,6,4,7,5,2, 4,7,5,2,3,6,4,8,2,8,3,7	23
		5S/30	5	1	7,6,3,5,2,3	6

		5S/60	10	2	3,6,4,7,5,1,2,4,8,1,7,6	12
		5S/90	15	3	5,2,4,8,5,8,7,6,3, 6,3,6,3,6,4,8,7,5	18
2	II	3S/30	6	4	5,1,2,3,6,3,6,4,7,5	10
		3S/60	15	5	8,7,6,3,5,2,3,5,1,2,4,8,7,5,1, 2,3,5,1,6	20
		3S/90	23	7	7,6,3,5,2,3,5,1,2,4,8,7,5,1,2, 3,5,1,6,1,7,1,2,3,6,4,7,5,1,2	30
		4S/30	5	3	5,2,4,8,5,8,7,6	8
		4S/60	10	5	2,3,6,4,8,7,6,3,6,4,8,6,7,6,4	15
		4S/90	15	8	4,8,7,6,4,7,5,2,3,5,2,3,5,2,4,7, 5,1,4,1,5,2,4	23
		5S/30	4	2	7,6,3,5,2,3	6
		5S/60	10	2	3,6,4,7,5,1,2,4,8,1,7,6	12
		5S/90	14	5	4,7,5,1,2,4,8,1,7, 6,3,5,1,8,2,4,8,1	18
3	III	3S/30	6	4	2,3,6,4,8,7,6,3,6,4	10
		3S/60	16	4	4,8,7,6,4,7,5,2,3,5,2,3,5,2,4, 7,5,1,4,1	20
		3S/90	24	6	5,2,4,8,5,8,7,6,3,6,3,6,3,6,4, 8,7,5,1,8,1,2,4,7,5,1,2,3,6,3	30
		4S/30	6	2	6,4,7,5,1,2,4,8	8
		4S/60	12	3	2,3,6,4,8,7,6, 3,6,4,8,6,7,6,4	15
		4S/90	20	3	5,1,2,3,6,3,6,4,7,5, 2,4,7,5,2,3,6,4,8,2,8,3,7	23
		5S/30	4	2	8,7,6,3,5,2	6
		5S/60	10	2	4,8,7,6,4,7,5,2,3,5,2,3	12
		5S/90	15	3	4,7,5,1,2,4,8,1,7, 6,3,5,1,8,2,4,8,1	18

3. Hari III , Hari Jumat 1 Desember 2006

1.a. *Manual*:

Tabel 4.24. Pengamatan *Manual* Hari III

No	Pemain ke	Durasi	Jumlah Pergerakan	Perubahan Posisi Arah Gerak
1	I	30	11	6,4,1,3,2,3,2,5,6,5,1
		60	19	1,4,6,2,1,4,6,2,4,5, 6,1,4,3,1,6,5,3,4
		90	29	3,2,1,4,3,5,4,1,2,5,4,6,2,4,5 3,6,5,4,6,2,1,3,1,5,2,4,1,6
2	II	30	9	4,1,3,2,4,5,2,6,5
		60	20	3,4,3,2,6,1,5,3,6,5 4,6,1,4,3,1,5,4,3,2
		90	28	1,4,3,6,4,1,2,5,4,6,2,4,6,3 3,6,5,4,3,2,6,2,1,4,2,3,1,2
3	III	30	9	1,3,4,2,4,5,6,1,2
		60	19	3,4,3,2,6,4,5,3,1 4,6,5,4,3,2,6,2,3,2
		90	26	4,2,3,6,4,1,3,5,4,6,5,4,1 3,6,5,4,3,2,6,3,1,4,2,3,2

1.b. Dengan *Trainer* :

Tabel 4.25. Pengamatan Penggunaan *Trainer*

No	Pemain ke	Pilihan Mode	Skor Benar	Skor Salah	Perubahan Posisi Arah Gerak	Jumlah Pergerakan
1	I	3S/30	8	2	2,3,6,4,8,7,6,3,6,4	10
		3S/60	15	5	8,7,6,3,5,2,3,5,1, 2,4,8,7,5,1,2,3,5,1,6	20
		3S/90	24	6	5,2,4,8,5,8,7,6,3,6,3,6,3,6,4 8,7,5,1,8,1,2,4,7,5,1,2,3,6,3	30
		4S/30	6	2	5,2,4,8,5,8,7,6	8
		4S/60	13	2	2,3,6,4,8,7,6,3,6,4,8,6,7,6,4	15
		4S/90	20	3	5,1,2,3,6,3,6,4,7,5, 2,4,7,5,2,3,6,4,8,2,8,3,7	23

		5S/30	4	2	2,4,8,7,6,4	6
		5S/60	10	2	5,1,2,3,6,3,6,4,7,5,2,4	12
		5S/90	16	2	4,7,5,1,2,4,8,1,7,6,3,5,1,8,2,4,8,1	18
2	II	3S/30	7	3	2,3,6,4,8,7,6,3,6,4	10
		3S/60	16	4	3,6,4,7,5,1,2,4,8,1,7,6,3,5,1,8,2,4,8,1	20
		3S/90	24	6	2,4,8,7,6,4,7,5,2,3,5,2,3,5,2,4,7,5,1,4,1,5,2,4,8,7,6,3,5,2	30
		4S/30	6	2	6,4,7,5,1,2,4,8	8
		4S/60	10	5	8,7,6,3,5,2,3,5,1,2,4,8,7,5,1	15
		4S/90	16	7	5,1,2,3,6,3,6,4,7,5,2,4,7,5,2,3,6,4,8,2,8,3,7	23
		5S/30	5	1	4,7,5,1,2,4	6
		5S/60	10	2	7,6,3,5,2,3,5,1,2,4,8,7	12
		5S/90	16	2	2,3,6,4,8,7,6,3,6,4,8,6,7,6,4,7,6,4	18
3	III	3S/30	6	4	5,1,2,3,6,3,6,4,7,5	10
		3S/60	16	4	8,7,6,3,5,2,3,5,1,2,4,8,7,5,1,2,3,5,1,6	20
		3S/90	22	8	6,4,7,5,1,2,4,8,1,7,6,3,5,1,8,2,4,8,1,8,5,1,4,7,5,2,3,5,1,5	30
		4S/30	5	3	5,2,4,8,5,8,7,6	8
		4S/60	11	4	7,6,3,5,2,3,5,1,2,4,8,7,5,1,2	15
		4S/90	19	4	2,4,8,7,6,4,7,5,2,3,5,2,3,5,2,4,7,5,1,4,1,5,2	23
		5S/30	5	1	7,6,3,5,2,3	6
		5S/60	11	1	5,1,2,3,6,3,6,4,7,5,2,4	12
		5S/90	16	2	4,7,5,1,2,4,8,1,7,6,3,5,1,8,2,4,8,1	18

4. Hari IV , Hari Sabtu 2 Desember 2006

1.a. *Manual*:

Tabel 4.26. Pengamatan *Manual* Hari IV

No	Pemain ke	Durasi	Jumlah Pergerakan	Perubahan Posisi Arah Gerak
1	I	30	9	5,4,1,4,2,3,5,2,6
		60	17	2,4,6,2,1,4,6,2, 5,1,4,3,1,6,5,4,1
		90	26	3,2,1,4,3,5,4,1,2,5,4,6,2, 3,6,5,4,6,2,1,3,1,5,2,4,1
2	II	30	10	2,4,1,3,2,4,5,2,6,5
		60	21	1,3,4,3,2,6,1,5,3,6,5 4,6,1,4,3,1,5,4,3,2
		90	29	5,1,4,3,6,4,1,2,5,4,6,2,4,6,3, 3,6,5,4,3,2,6,2,1,4,2,3,1,2
3	III	30	11	1,3,4,2,4,5,6,1,2,1,4
		60	18	5,4,3,2,6,4,5,3,1, 6,3,4,3,2,6,2,3,1
		90	29	1,2,1,2,3,6,4,1,3,5,4,6,5,4,1, 4,3,6,5,4,3,2,6,3,1,4,2,3,2

1.b. Dengan *Trainer* :

Tabel 4.27. Pengamatan Penggunaan *Trainer*

No	Pemain ke	Pilihan Mode	Skor Benar	Skor Salah	Perubahan Posisi Arah Gerak	Jumlah Pergerakan
1	I	3S/30	9	1	2,4,8,7,6,4,7,5,2,3	10
		3S/60	18	2	4,8,7,6,4,7,5,2,3,5,2,3,5,2, 4,7,5,1,4,1	20
		3S/90	27	3	3,6,4,7,5,1,2,4,8,1,7,6,3,5,1, 8,2,4,8,1,8,5,1,4,7,5,2,3,5,1	30
		4S/30	6	2	6,4,7,5,1,2,4,8	8
		4S/60	13	2	5,1,2,3,6,3,6,4,7,5,2,4,7,5,2	15
		4S/90	21	2	2,4,8,7,6,4,7,5,2,3,5,2,3,5,2,4, 7,5,1,4,1,5,2	23
		5S/30	6	0	4,7,5,1,2,4	6
		5S/60	11	1	8,7,6,3,5,2,3,5,1,2,4,8	12

		5S/90	16	2	3,6,4,7,5,1,2,4,8,1,7,6,3,5, 1,8,2,4	18
2	II	3S/30	9	1	5,1,2,3,6,3,6,4,7,5	10
		3S/60	18	2	4,8,7,6,4,7,5,2,3,5,2,3,5,2, 4,7,5,1,4,1	20
		3S/90	26	4	3,6,4,7,5,1,2,4,8,1,7,6,3,5,1, 8,2,4,8,1,8,5,1,4,7,5,2,3,5,1	30
		4S/30	7	1	7,6,3,5,2,3,5,1	8
		4S/60	12	3	2,3,6,4,8,7,6,3,6,4,8,6,7,6,4	15
		4S/90	18	5	2,4,8,7,6,4,7,5,2,3,5,2, 3,5,2,4,7,5,1,4,1,5,2	23
		5S/30	6	0	7,6,3,5,2,3	6
		5S/60	11	1	4,8,7,6,4,7,5,2,3,5,2,3	12
		5S/90	16	2	3,6,4,7,5,1,2,4,8,1,7,6,3,5, 1,8,2,4	18
3	III	3S/30	8	2	4,8,7,6,4,7,5,2,3,5	10
		3S/60	17	3	4,8,7,6,4,7,5,2,3,5,2,3,5,2, 4,7,5,1,4,1	20
		3S/90	26	4	7,6,3,5,2,3,5,1,2,4,8,7,5,1,2, 3,5,1,6,1,7,1,2,3,6,4,7,5,1,2	30
		4S/30	6	2	2,3,6,4,8,7,6,3	8
		4S/60	13	2	5,1,2,3,6,3,6,4,7,5,2,4,7,5, 2	15
		4S/90	20	3	4,8,7,6,4,7,5,2,3,5,2,3,5,2,4,7, 5,1,4,1,5,2,4	23
		5S/30	6	0	4,7,5,1,2,4	6
		5S/60	12	0	3,6,4,7,5,1,2,4,8,1,7,6	12
		5S/90	16	2	5,2,4,8,5,8,7,6,3,6,3,6,3,6, 4,8,7,5	18

5. Hari V , Hari Minggu 3 Desember 2006

1.a. *Manual*:

Tabel 4.28. Pengamatan *Manual* Hari V

No	Pemain ke	Durasi	Jumlah Pergerakan	Perubahan Posisi Arah Gerak
1	I	30	10	5,4,1,4,2,3,5,2,6,1
		60	15	2,4,6,2,1,4,6, 1,4,3,1,6,5,4,1
		90	24	2,1,4,3,5,4,1,2,5,4,6,2, 6,5,4,6,2,1,3,1,5,2,4,1
2	II	30	9	2,4,1,3,2,4,5,2,5
		60	23	1,3,4,3,2,6,1,5,3,6,5,4 4,6,1,4,3,1,5,4,3,2,1
		90	27	5,1,4,3,6,4,1,2,5,4,6,2,4, 3,6,5,4,3,2,6,2,1,4,2,3,1,2
3	III	30	12	1,3,4,2,4,5,6,1,2,1,4,1
		60	19	5,4,3,2,6,4,5,3,1, 6,3,4,3,2,6,2,3,1,4
		90	27	2,1,2,3,6,4,1,3,5,4,6,5,4,1, 4,3,6,5,4,3,2,6,3,1,4,2,3,

1.b. Dengan *Trainer* :

Tabel 4.29. Pengamatan Penggunaan *Trainer*

No	Pemain ke	Pilihan Mode	Skor Benar	Skor Salah	Perubahan Posisi Arah Gerak	Jumlah Pergerakan
1	I	3S/30	10	0	2,3,6,4,8,7,6,3,6,4	10
		3S/60	18	2	4,8,7,6,4,7,5,2,3,5,2,3,5,2, 4,7,5,1,4,1	20
		3S/90	26	4	2,4,8,7,6,4,7,5,2,3,5,2,3,5,2, 4,7,5,1,4,1,5,2,4,8,7,6,3,5,2	30
		4S/30	7	1	5,2,4,8,5,8,7,6	8
		4S/60	14	1	5,1,2,3,6,3,6,4,7,5, 2,4,7,5,2	15

		4S/90	22	1	5,1,2,3,6,3,6,4,7,5,2, 4,7,5,2,3,6,4,8,2,8,3,7	23
		5S/30	5	1	7,6,3,5,2,3	6
		5S/60	11	1	3,6,4,7,5,1,2,4,8,1,7,6	12
		5S/90	15	3	5,2,4,8,5,8,7,6,3,6 ,3,6,3,6,4,8,7,5	18
2	II	3S/30	8	2	7,6,3,5,2,3,5,1,2,4	10
		3S/60	17	3	8,7,6,3,5,2,3,5,1,2,4,8,7,5,1, 2,3,5,1,6	20
		3S/90	26	4	2,4,8,7,6,4,7,5,2,3,5,2,3,5,2, 4,7,5,1,4,1,5,2,4,8,7,6,3,5,2	30
		4S/30	8	0	2,3,6,4,8,7,6,3	8
		4S/60	12	3	4,8,7,6,4,7,5,2,3,5,2,3,5,2,4	15
		4S/90	19	4	5,1,2,3,6,3,6,4,7,5,2,4,7,5,2, 3,6,4,8,2,8,3,7	23
		5S/30	6	0	4,7,5,1,2,4	6
		5S/60	12	0	3,6,4,7,5,1,2,4,8,1,7,6	12
		5S/90	15	3	4,7,5,1,2,4,8,1,7,6,3,5,1,8,2, 4,8,1	18
3	III	3S/30	9	1	7,6,3,5,2,3,5,1,2,4	10
		3S/60	17	3	2,4,8,7,6,4,7,5,2,3,5,2,3,5, 2,4,7,5,1,4	20
		3S/90	25	5	6,4,7,5,1,2,4,8,1,7,6,3,5,1,8, 2,4,8,1,8,5,1,4,7,5,2,3,5,1,5	30
		4S/30	7	1	7,6,3,5,2,3,5,1	8
		4S/60	12	3	2,3,6,4,8,7,6,3,6, 4,8,6,7,6,4	15
		4S/90	17	6	2,4,8,7,6,4,7,5,2,3,5,2,3,5, 2,4,7,5,1,4,1,5,2	23
		5S/30	6	0	4,8,7,6,4,7	6
		5S/60	11	1	7,6,3,5,2,3,5,1,2,4,8,7	12
		5S/90	15	3	3,6,4,7,5,1,2,4,8,1,7,6,3,5, 1,8,2,4	18

II. *Badminton Circuit Trainer:*

a. Lampu, Blok Alat dan Penyangga:



b. Blok Alat Tampak Atas :

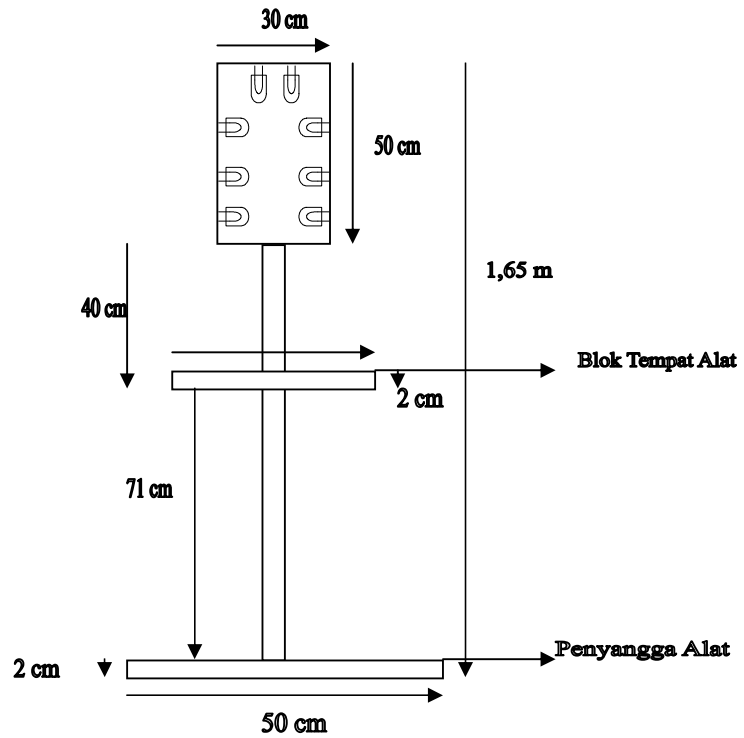


c. Tombol *Trainer*:

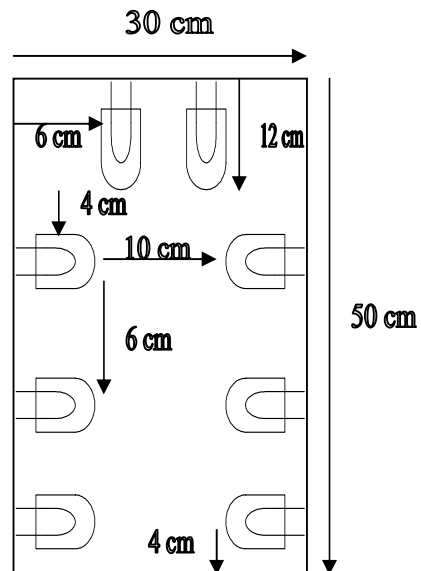


Rancangan Tiang Penyangga Papan Latihan Bulutangkis

1. Tampak Depan :



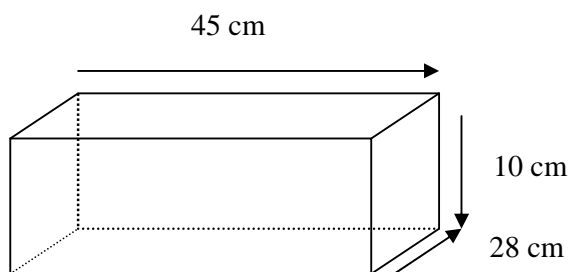
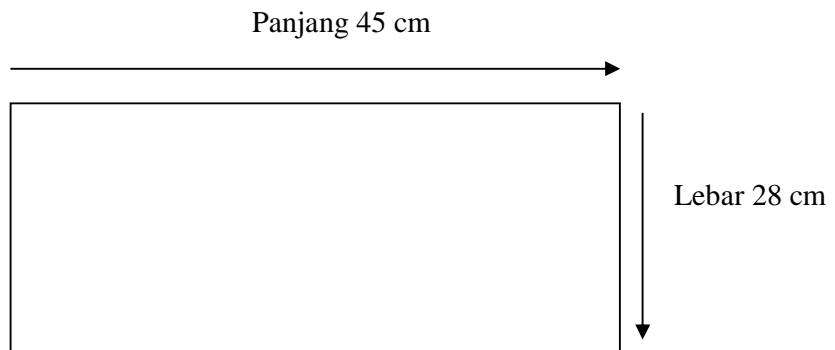
2. Blok Lampu Tampak Depan:



3. Blok Tempat Alat:

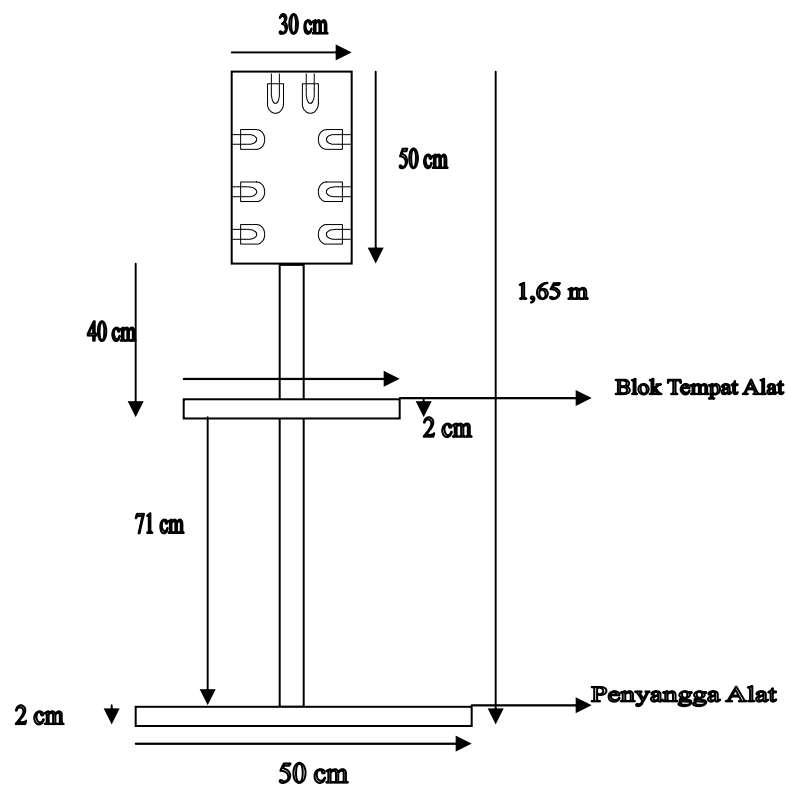
Panjang: 45 cm, Lebar: 28 cm, Tinggi: 10 cm

Tampak Atas:

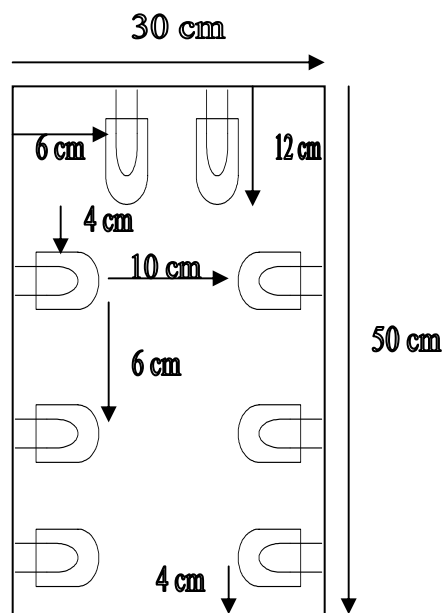


Rancangan Tiang Penyangga Papan Latihan Bulutangkis

1. Tampak Depan :



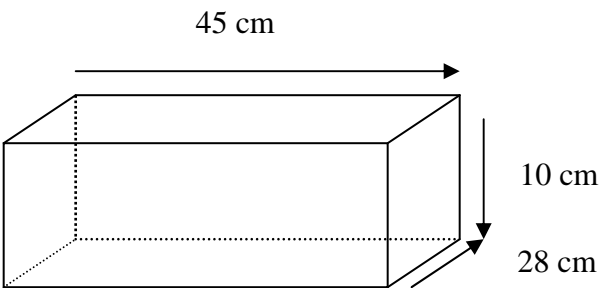
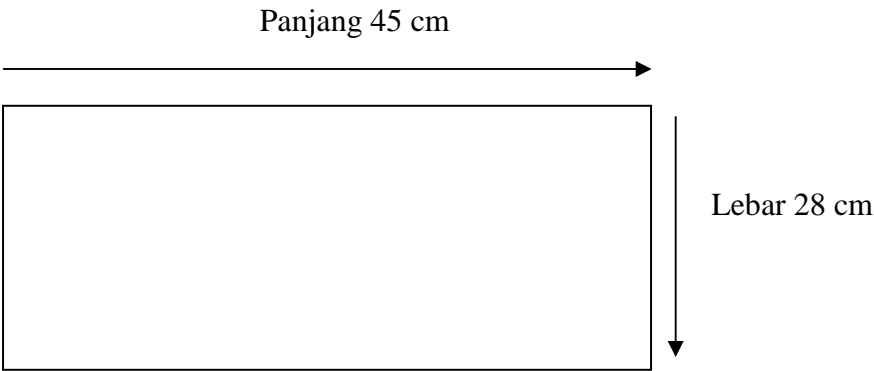
2. Blok Lampu Tampak Depan:



3. Blok Tempat Alat:

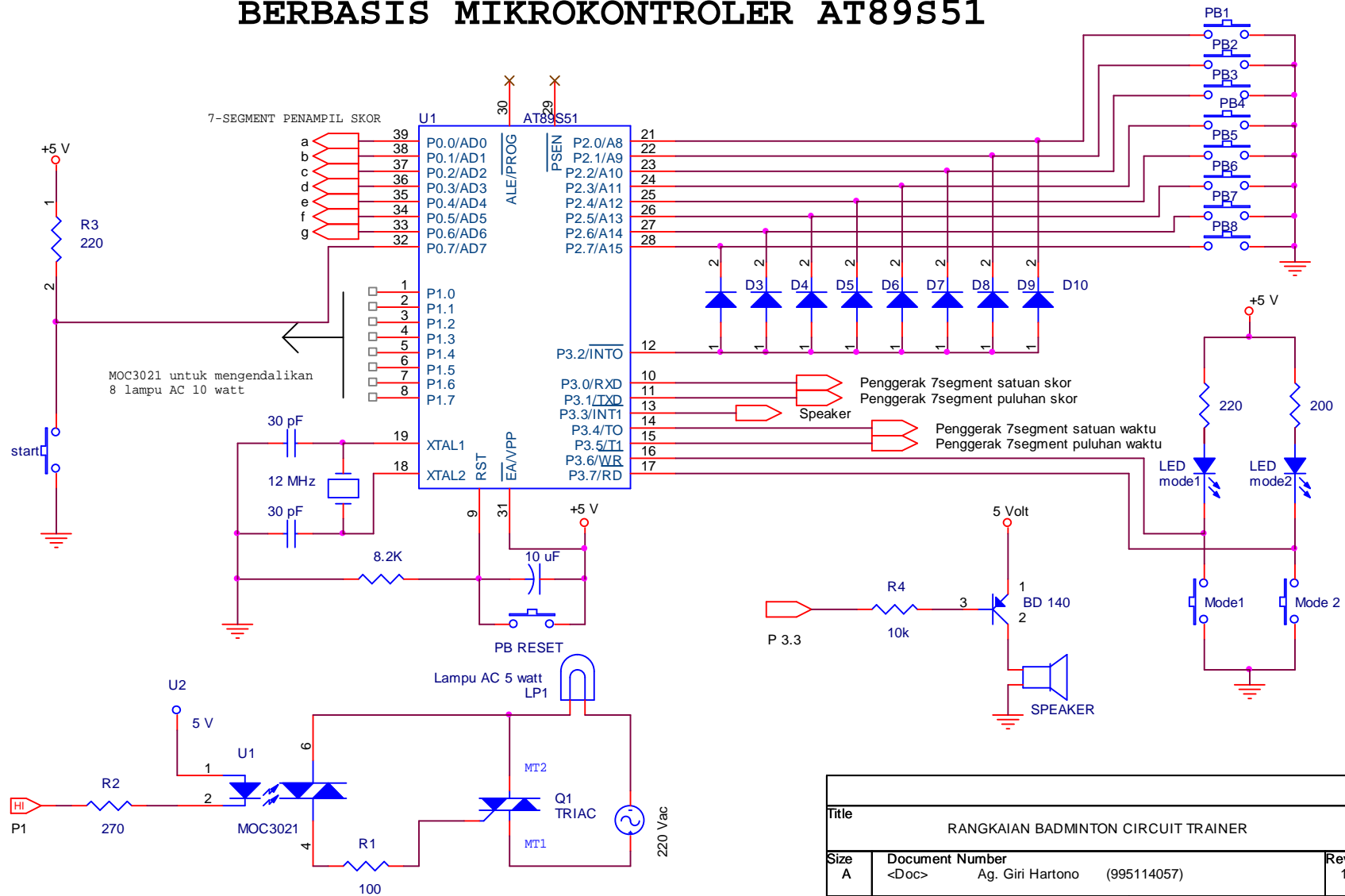
Panjang: 45 cm, Lebar: 28 cm, Tinggi: 10 cm

Tampak Atas:



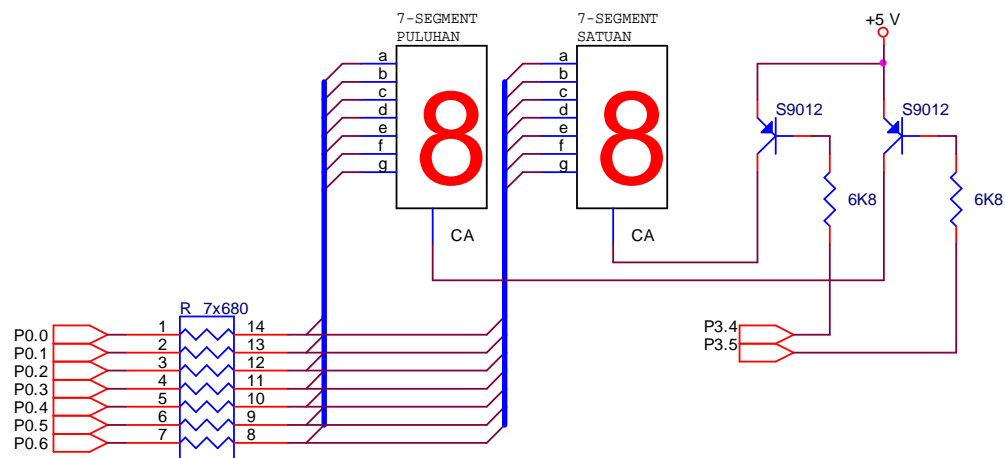
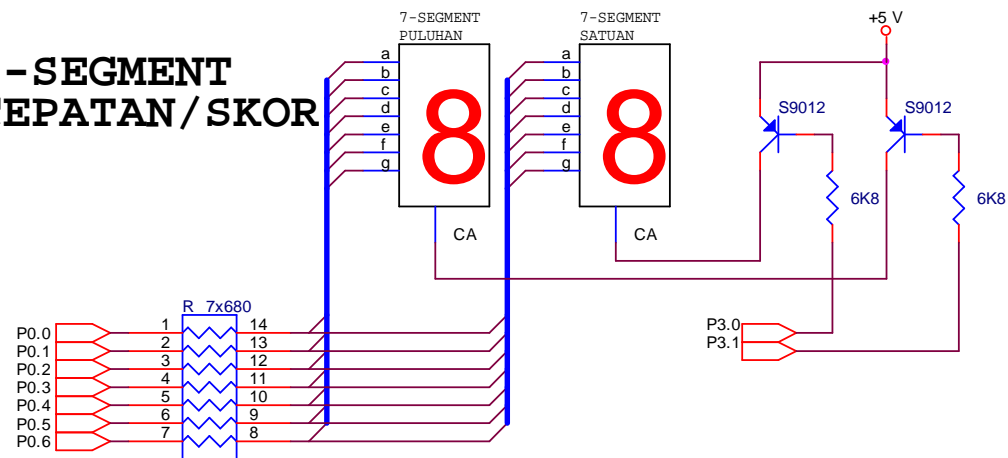
RANGKAIAN BADMINTON CIRCUIT TRAINER

BERBASIS MIKROKONTROLER AT89S51

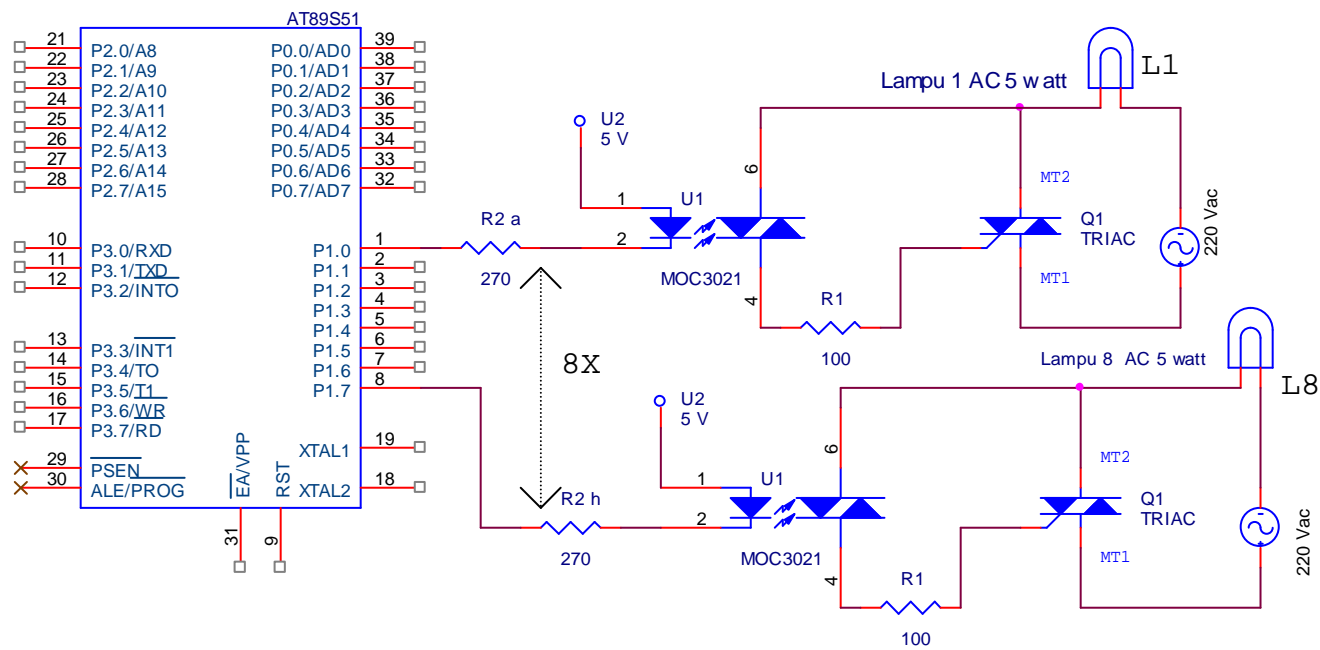


Title			
RANGKAIAN BADMINTON CIRCUIT TRAINER			
Size	Document Number		Rev
A	<Doc> Ag. Giri Hartono (995114057)		1
Date:	Tuesday, January 09, 2007	Sheet	1 of 1

PENAMPIL 7-SEGMENT DURASI /KECEPATAN/ SKOR



Title		
Penampil		
Size A	Document Number Agustinus Giri / 995114057	Rev 4
Date:	Tuesday , January 09, 2007	Sheet 1 of 1



Title		
Triac dan MOC		
Size A	Document Number	Rev 2
Agustinus Giri Hartono		
Date: Tuesday, January 09, 2007	Sheet 1 of 1	


```

;*****
;   Papan Latih Bulutangkis Berbasis Mikrokontroler AT89S51
;       Nama : Agustinus Giri Hartono
;       NIM  : 995114057
;*****
;-----
;       Inisialisasi port
;-----
Ruas_Penampil      equ P0      ; Label Port 0
Lampu_AC           equ P1      ; Label Port 1
Tombol_trainer     equ P2      ; Label Port 2
Speker_TOA         bit p3.3    ; p3.3 kendali Speaker
Puluhan_skor       bit p3.1    ; p3.1 kendali ruas skor puluhan
Satuan_skor        bit p3.0    ; p3.0 kendali ruas skor satuan
Satuan_waktu       bit P3.4    ; p3.4 kendali ruas satuan waktu
Puluhan_waktu      bit P3.5    ; p3.5 kendali ruas puluhan waktu
Model_kecepatan    bit P3.6    ; p3.1 tombol mode kecepatan
Mode2_durasi       bit P3.7    ; p3.7 tombol mode durasi
Tombol_start       bit P0.7

Tone_Mulai         equ 0F0h
Tone_Selesai       equ 0Fh
Trainer_Mulai      equ 70h
Trainer_Selesai    equ 07h

;-----
;       Inisialisasi alamat penyimpanan data pada RAM
;-----
Random_temp        equ 23h     ;Penyimpan sementara data operasi random
Hasil_random       equ 30h     ;Penyimpan data hasil random
Data_Lampu_AC      equ 31h     ;Penyimpan data Lampu yang ditampilkan
Skor_Benar         equ 32h     ;Penyimpan data skor benar
Skor_Salah         equ 33h     ;Penyimpan data skor salah
Cacah_start        equ 34h     ;Penyimpan data random dari tombol start
Data_Temp_Kecs     equ 35h     ;Penyimpan sementara Data kecepatan
Data_Per_Kec       equ 36h     ;Penyimpan data kecepatan (3,4,5)
Data_Temp_1s       equ 37h     ;Penyimpan data counter delay
Counter_50ms       equ 38h     ;Penyimpan data counter delay
Counter_05ms       equ 39h     ;Penyimpan data counter delay
Tekan_apa          equ 3Ah     ;Penyimpan data tombol yang ditekan
Jumlah_random      equ 3Bh     ;Penyimpan data jumlah random
Data_Kecepatan     equ 3Ch     ;Penyimpan data kecepatan (3S,4S,5S)
Data_Durasi        equ 3Dh     ;Penyimpan data durasi (30,60,90)
Data_Skor          equ 3Eh     ;Penyimpan data skor
Kol_Puluhan_skor   equ 3Fh     ;Penyimpan data skor digit puluhan
Kol_Satuan_skor    equ 40h     ;Penyimpan data skor digit satuan
Kol_Puluhan_waktu  equ 41h     ;Penyimpan data waktu digit puluhan
Kol_Satuan_waktu   equ 42h     ;Penyimpan data waktu digit satuan
Tanda_Tone         equ 43h     ;Penyimpan data tanda tone
Tanda_Trainer      equ 44h     ;Penyimpan data tanda trainer
Awal_data_7seg_RAM equ 50h     ;Penyimpan awal data 7segment

;-----
;       Inisialisasi Vektor reset dan interupsi
;-----

org      0h
ajmp     Mulai          ;Vektor reset
org      03h
ljmp     Interrupt_Exter0
org      0bh
ljmp     Interrupt_Timer0
org      13h
reti
org      1bh
ljmp     Interrupt_Timer1
org      23h
reti

;=====
;       Awal Program

```

```

;=====
Mulai:
    call    Inisialisasi_awal    ;Panggil Rutin Inisialisasi_awal
    call    Pilih_mode          ;Panggil Rutin Pilih_mode
    call    Start_main          ;Panggil Rutin Start_main
    call    Trainer              ;Panggil Rutin Permainan
    call    Evaluasi             ;Ulangi sampai terjadi interupsi

Balik:
    call    Scan_Tampilan       ;Panggil Scan_Tampilan
    sjmp    Balik               ;Lompat ke Balik
;=====
;  Inisialisasi Program
;=====
Inisialisasi_awal:
    setb    EA                  ;Aktifkan interupsi
    mov     tmod,#11h           ;Timer 1 dan timer 0 mode 1 (16 bit)
    mov     Ruas_Penampil,#0FFh ;Inisialisasi port yang digunakan
    mov     Skor_Benar,#00h     ;Isi Skor_Benar dengan 00h
    mov     Skor_Salah,#00h     ;Isi Skor_Salah dengan 00h
    mov     Cacah_start,#10     ;Cacahan lama penekanan Start
    mov     Kol_Satuan_skor,#0FFh ;Kol_Satuan_Skor <- 0FFh
    mov     Kol_Puluhan_skor,#0FFh ;Kol_Puluhan_Skor<- 0FFh
    mov     Kol_Satuan_waktu,#0FFh ;Kol_Satuan_waktu <- 0FFh
    mov     Kol_Puluhan_waktu,#0FFh ;Kol_Puluhan_waktu <- 0FFh
    mov     Data_Durasi,#0FFh    ;Data_Durasi <- 0FFh
    mov     Data_Kecepatan,#0FFh ;Data_Kecepatan <- 0FFh
    mov     dptr,#Data_7segment ;Alamat pertama sumber data angka 7's
    mov     r0,#Awal_data_7seg_RAM ;Alamat data pertama = 50H
    mov     r1,#10              ;Banyaknya data = 10
Copy_ke_RAM:
    clr     a                   ;Proses transfer sebanyak 10 data
    movc    a,@a+dptr           ;Akumulator diclearkan
    mov     @r0,a               ;Akumulator <- data di alamat @a+dptr
    inc     r0                  ;Akumulator -> dialamat yang ditunjukan r0
    inc     r0                  ;R0=R0+1
    inc     dptr                ;Dptr=Dptr+1
    djnz    r1,Copy_ke_RAM      ;r1=0?, ulangi Copy_ke_RAM
    mov     dptr,#Data_random_Lampu ;DPTR <- data di Data_random_Lampu
    clr     a                   ;clearkan A
    ret

;=====
;  Pemilihan Mode trainer
;=====
Pilih_mode:
M_Kec:
    jb      Model_kecepatan,M_Dura ;Mode Kecepatan
    call    Scan_Tampilan          ;Mode Kecepatan ditekan? Tidak=M_Dura
    call    Scan_Tampilan          ;Panggil Scan_Tampilan
    jb      Model_kecepatan,M_Dura ;Cek Mode Kecepatan ditekan?Tidak = M_Dura
Ulang1:
    call    Scan_Tampilan          ;Panggil Scan_Tampilan
    mov     R0,Data_Kecepatan      ;R0 diisi dengan data di Data_Kecepatan
    cjne    R0,#0FFh,Cek1         ;R0=#0FFh,tidak ke Cek1
    mov     Data_Kecepatan,#25     ;Ya, isi Data_Kecepatan dengan 25
Cek1:
    jnb     Model_kecepatan,Ulang1 ;Tunggu tombol Mode_kecepatan dilepas
    mov     a,Data_Kecepatan       ;Akumulator <- dengan data di Data_Kecepatan
    cjne    a,#55,Lewat1          ;Akumulator=#55,tidak ke Lewat1
    mov     Data_Kecepatan,#35     ;Ya, isi Data_Kecepatan dengan 35
    jmp     Terus1                ;Lompat ke Terus1
Lewat1:
    add     a,#10                 ;A=A+10
    mov     Data_Kecepatan,a       ;Data_Kecepatan <- Akumulator
Terus1:
    call    Hex_2_bcd_Kec          ;panggil Hex_2_bcd_Kec

M_Dura:
    jb      Mode2_durasi,M_Start   ;Mode Durasi
    call    Scan_Tampilan          ;Mode Durasi ditekan? Tidak = M_Start
    call    Scan_Tampilan          ;Panggil Scan_Tampilan
    jb      Mode2_durasi,M_Start   ;Cek Mode Durasi ditekan? Tidak = M_Start
Ulang2:

```

```

        call    Scan_Tampilan          ;Panggil Scan_Tampilan
        mov     R0,Data_Durasi         ;R0 diisi data di Data_Durasi
        cjne    R0,#0FFh,Cek2         ;R0=#0FFh,tidak ke Cek2
        mov     Data_Durasi,#0         ;Data_Durasi <- 0
Cek2:
        jnb     Mode2_durasi,Ulang2    ;Tunggu tombol dilepas
        mov     a,Data_Durasi          ;Data di Data_Durasi -> ke akumulator
        cjne    a,#90,Lewat2           ;kumulator=#90, tidak ke Lewat2
        mov     Data_Durasi,#30        ;Ya, isikan 30 ke Data_Durasi
        jmp     Terus2                ;Lompat ke Terus2
Lewat2:
        add     a,#30                  ;A=A+30
        mov     Data_Durasi,a          ;Isi Data_Durasi dengan isi Akumulator
Terus2:
        call    Hex_2_bcd_Dur         ;Panggil Hex_2_bcd_Dur

M_Start:
        call    Scan_Tampilan          ;Cek Tombol Start
        jnb     Tombol_start,M_Kec     ;Start ditekan? Tidak = M_Kec
Lagi:
        call    Delay_5ms             ;Ya = tunda 5 ms
        djnz    Cacah_start,Isi_random ;Cacahan -1, Cacahan=0?
        mov     Cacah_start,#10        ;Ya -> isi cacah =10
Isi_random:
        jnb     Tombol_start,Lagi      ;Tombol Start dilepas? Tidak = Cacahan-1
        ret

;=====
;      Start_main
;=====
Start_main:
        call    Cek_Perulangan        ;Panggil Cek_Perulangan
        call    Tanda_Siap            ;Panggil Tanda_Siap
        mov     Counter_50ms,#40      ;Isikan 40 ke Counter_50ms
        call    Tone_Speaker          ;Panggil Tone_Speaker
        ret

Tanda_Siap:
        mov     Kol_Satuan_skor,#50h   ;Nilai satuan+50h (data 7segment)
        mov     Kol_Puluhan_skor,#50h  ;Nilai puluhan+50h
        mov     a,#00h                 ;Salin 00h ke Akumulator
        mov     Lampu_AC,a             ;Salin isi akumulator ke Lampu_AC
Loop1:
        call    Delay_1s              ;Delay 1 s
        rlc     a                     ;Lampu bergeser 1x
        mov     Lampu_AC,a            ;Lampu_AC <- A
        cjne    a,#0FFh,Loop1         ;Akumulator=#0FFh,tidak ke Loop1
        ret

Tone_Speaker:
        mov     Tanda_Tone,#Tone_Mulai
        mov     th1,#high(-500)        ; th1=-500
        mov     tl1,#low(-500)         ; tl1=-500
        clr     tfl                    ; Clearkan bit tfl
        setb    etl                    ; Hidupkan etl
        setb    trl                    ; Matikan timer1
Cek_Tone:
        mov     a,Tanda_Tone           ;Isi akumulator dengan data di Tanda_Tone
        cjne    a,#Tone_Selesai,Loop2 ;A = #Tone_Selesai ?, Tidak ke Loop2
        ret
Loop2:
        call    Scan_Tampilan          ;Panggil Scan_Tampilan
        jmp     Cek_Tone              ;Lompat ke Cek_Tone

;=====
;      Trainer
;=====
Trainer:
        mov     Tanda_Trainer,#Trainer_Mulai;Trainer_Mulai -> Tanda_Trainer
        mov     th0,#high(-50000)      ;th0=-50000
        mov     tl0,#low(-50000)       ;tl0=-50000

```

```

        clr      tf0                      ;Clearkan bit tf0
        setb     et0                      ;Hidupkan et0
        setb     tr0                      ;Hidupkan timer 0
        mov      Data_Temp_ls,#20        ;Data_Temp_ls <- 20
        mov      Data_Temp_Kecs,Data_Per_Kec ;Data_Temp_Kecs <-data di Data_Per_Kec
        mov      Jumlah_random,#0        ;Jumlah_random = 0
        mov      Hasil_random,Cacah_start ;Angka random pertama
        call     Random                  ;Panggil sub rutin proses random 8 bit
        call     Nyala_Lampu             ;Panggil sub rutin penyalaan LED
        setb     ex0                      ;Hidupkan Interupsi Eksternal 0
        setb     it0                      ;Hidupkan Interupsi Timer 0
Ulang_main:
        mov      a,Tanda_Trainer          ;data di Tanda_Trainer -> Akumulator
        cjne     a,#Trainer_Selesai,Loop3;A = # Trainer_Selesai ? Tidak ke loop3
        clr      ex0                      ;Ya, Matikan interupsi Eksternal 0
        ret
Loop3:
        call     Scan_Tampilan            ;Panggil Scan_Tampilan
        mov      a,Jumlah_random          ;Akumulator <- Jumlah_random
        subb     a,Skor_Benar             ;Jumlah_random - Skor_Benar -> A
        mov      Skor_Salah,a             ;Skor_Salah <- A
        jmp      Ulang_main               ;Ulang trainer sampai ada interupsi

;=====
;  Cek skor untuk tampilan akhir (evaluasi skor)
;=====
Evaluasi:
        mov      Counter_50ms,#100        ;Counter_50ms <- 100
        mov      Counter_05ms,#100       ;Counter_05ms <- 100
        call     Tone_Speaker             ;Panggil Tone_Speaker
        call     Hex_2_bcd_Skor           ;Subrutin pengubah Heksa ke BCD
        mov      Data_Durasi,Skor_Salah   ;Data_Durasi <- data Skor_Salah
        call     Hex_2_bcd_Dur           ;Panggil Ke Hex_2_bcd_Dur
        ret

;-----
;  Interupsi Eksternal 0 ( Masukan Tombol )
;-----
Interrup_Exter0:
        Mov      r7,a                     ;Data di akumulator -> r7
Cek_tombol:
        mov      a,Tombol_trainer         ;Simpan data tombol trainer P2
        cjne     a,#0ffh,Cek_error        ;Tombol trainer ditekan? Ya = cek error
        jmp      Keluar_IETx0             ;Lompat ke Keluar_IETx0
Cek_error:
        cjne     a,Tekan_apa,Cek_tekan    ;Tombol=tombol sebelumnya?tidak=Cek_tekan
        jmp      Keluar_IETx0             ;Lompat ke Keluar_IETx0
Cek_tekan:
        mov      Tekan_apa,a              ;Simpan data di akumulator ke Tekan_apa
        anl      a,Data_Lampu_AC          ;Data tombol=data Lampu?
        cjne     a,Data_Lampu_AC,Keluar_IETx0 ;Tidak = lompat ke Keluar_IETx0
        Inc      Skor_Benar               ;Ya = Skor_Benar + 1
        call     Hex_2_bcd_Skor           ;Subrutin pengubah Heksa ke BCD skor
        jmp      Keluar_IETx0             ;Lompat ke Keluar_IETx0
Keluar_IETx0:
        Mov      a,r7                     ;Salin data di r7 ke akumulator
        reti

;-----
;  Interupsi Timer 0 untuk waktu Trainer
;-----
Interrup_Timer0:
        clr      tf0                      ;Clearkan bit tf0
        mov      th0,#high(-50000)        ;th1=-50000
        mov      tl0,#low(-50000)         ;tl1=-50000
        djnz     Data_Temp_ls,Keluar_IntT0 ;Data_Temp_ls=0?
                                                ;Tidak, ke Keluar_IntT0

```

```

    djnz    Data_Durasi,Cek_Kecepatan ;Ya, Data_Durasi=0?
                                         ;Tidak, ke Cek_Kecepatan
    mov     Tanda_Trainer,#Trainer_Selesai
                                         ;Ya, Trainer_Selesai->Tanda_Trainer
    call    Hex_2_bcd_Dur                 ;Panggil Hex_2_bcd_Dur
    clr     tr0                          ;Matikan timer 0
    clr     tf0                          ;Clearkan bit tf0
    clr     et0                          ;Clearkan bit et0
    reti

Cek_Kecepatan:
    mov     Data_Temp_1s,#20             ;Isi Data_Temp_1s dengan 20
    call    Hex_2_bcd_Dur                 ;Panggil Hex_2_bcd_Dur
    djnz    Data_Temp_Kecs,Keluar_IntT0 ;Cek Data_Temp_Kecs=0?
                                         ;Tidak,Keluar_IntT0
    mov     Data_Temp_Kecs,Data_Per_Kec ;Data di Data_Per_Kec->Data_Temp_Kecs
    call    Random                       ;Panggil Random
    call    Nyala_Lampu                  ;Panggil Nyala_Lampu
Keluar_IntT0:
    reti                                  ;Akhir interupsi timer 0

;-----
;      Interupsi Timer 1 untuk tunda tone speaker
;-----
Interrup_Timer1:
    clr     tf1                          ; Clearkan bit tf0
    mov     th1,#high(-500)              ; th1=-500
    mov     tl1,#low(-500)               ; tl1=-500
    djnz    Counter_05ms,Tunggu_05ms    ; Counter_05ms = 0 ?, Tidak Tunggu_05ms
    djnz    Counter_50ms,Tunggu_50ms    ; Counter_50ms = 0 ?, Tidak Tunggu_50ms
    mov     Tanda_Tone,#Tone_Selesai    ; Salin isi Tone_Selesai ke Tanda_Tone
    clr     tr1                          ; Matikan timer 0
    clr     tf1                          ; Bit overflow di-nolkan
    clr     et1                          ; Matikan interupsi timer 1
    reti

Tunggu_50ms:
    mov     Counter_05ms,#100            ; Isi Counter_05ms dengan 100
Tunggu_05ms:
    cpl     Speker_TOA                   ; Ubah/membalikkan kondisi Speaker P3.3
Keluar_IntT1:
    reti                                  ; Akhir interupsi timer 0

;-----
;      Scan Tampilan
;-----
Scan_Tampilan:
    call    Delay_segment                ;Delay tombol mode belum ditekan
    mov     R0,Kol_Puluhan_skor          ;Data di Kol_Puluhan_skor-> R0
    mov     Ruas_Penampil,@R0           ;P0 = data angka puluhan skor
    clr     Puluhan_skor                 ;Hidupkan digit puluhan skor
    setb    Satuan_skor                 ;Matikan digit satuan skor
    setb    Puluhan_waktu               ;Matikan digit puluhan waktu
    setb    Satuan_waktu                ;Matikan digit satuan waktu
    call    Delay_segment                ;Delay tombol mode belum ditekan

    mov     R0,Kol_Satuan_skor           ;Data di Kol_Satuan_skor -> R0
    mov     Ruas_Penampil,@R0           ;P0 = data angka satuan skor
    setb    Puluhan_skor                 ;Matikan digit puluhan skor
    clr     Satuan_skor                 ;Hidupkan digit satuan skor
    setb    Puluhan_waktu               ;Matikan digit puluhan waktu
    setb    Satuan_waktu                ;Matikan digit satuan waktu
    call    Delay_segment                ;Delay tombol mode belum ditekan

    mov     R0,Kol_Puluhan_waktu         ;Data di Kol_Puluhan_waktu -> R0
    mov     Ruas_Penampil,@R0           ;P0 = data angka puluhan waktu
    setb    Puluhan_skor                 ;Matikan digit puluhan skor
    setb    Satuan_skor                 ;Matikan digit satuan skor
    clr     Puluhan_waktu               ;Hidupkan digit puluhan waktu
    setb    Satuan_waktu                ;Matikan digit satuan waktu
    call    Delay_segment                ;Delay tombol mode belum ditekan

```

```

        mov     R0,Kol_Satuan_waktu    ;Data di Kol_Satuan_waktu -> R0
        mov     Ruas_Penampil,@R0      ;P0= data angka satuan waktu
        setb    Puluhan_skor           ;Matikan digit puluhan skor
        setb    Satuan_skor            ;Matikan digit satuan skor
        setb    Puluhan_waktu          ;Matikan digit puluhan waktu
        clr     Satuan_waktu           ;Hidupkan digit satuan waktu
        call    Delay_segment           ;Delay tombol mode belum ditekan
        ret

;-----
;   Hex_to_bcd
;-----
Hex_2_bcd_Kec:
        mov     b,#10                  ;Skor diubah dari Heksa ke BCD
        mov     a,Data_Kecepatan       ;Register b diisi dengan 10
        div     ab                      ;Salin Data_Kecepatan ke Akumulator
        div     ab                      ;Skor dibagi 10
        mov     Kol_Satuan_skor,b       ;Nilai satuan = sisa hasil bagi
        mov     Kol_Puluhan_skor,a      ;Nilai puluhan = sisa hasil bagi
        jmp     BCD_ke_7               ;Lompat ke BCD_ke_7
Hex_2_bcd_Dur:
        mov     b,#10                  ;Register b diisi dengan 10
        mov     a,Data_Durasi           ;Data_Durasi salin ke Akumulator
        div     ab                      ;Skor dibagi 10
        mov     Kol_Satuan_waktu,b      ;Nilai satuan = sisa hasil bagi
        mov     Kol_Puluhan_waktu,a     ;Nilai puluhan = sisa hasil bagi
        jmp     BCD_ke_7               ;Lompat ke BCD_ke_7
Hex_2_bcd_Skor:
        mov     b,#10                  ;Skor diubah dari Heksa ke BCD
        mov     a,Skor_Benar            ;Register b diisi dengan 10
        div     ab                      ;Skor_Benar diisikan ke Akumulator
        div     ab                      ;Skor dibagi 10
        mov     Kol_Satuan_skor,b       ;Nilai satuan = sisa hasil bagi
        mov     Kol_Puluhan_skor,a      ;Nilai puluhan = sisa hasil bagi
        jmp     BCD_ke_7               ;Lompat ke BCD_ke_7

;-----
;   BCD ke 7 segment
;-----
BCD_ke_7:
        orl     Kol_Satuan_skor,#Awal_data_7seg_RAM ;Nilai satuan+50h(data7segment)
        orl     Kol_Puluhan_skor,#Awal_data_7seg_RAM ;Nilai puluhan+50h
        orl     Kol_Satuan_waktu,#Awal_data_7seg_RAM ;Nilai satuan+50h(data7segment)
        orl     Kol_Puluhan_waktu,#Awal_data_7seg_RAM ;Nilai puluhan+50h
        ret

;-----
;   Delay 2 detik
;-----
Delay_2s:
        mov     Counter_50ms,#40       ;Isi Counter_50ms dengan 40
Tunggu_2s:
        call    Debounce                ;Panggil Debounce
        djnz    Counter_50ms,Tunggu_2s ;Counter_50ms = 0 ? ,tidak ke Tunggu_2s
        ret

;-----
;   Delay 1 detik
;-----
Delay_1s:
        mov     Counter_50ms,#20       ;Counter=20
Tunggu_1s:
        call    Debounce                ;Delay 50 ms
        djnz    Counter_50ms,Tunggu_1s ;20 x 50 ms = 1 detik
        ret                             ;Counter= 0, sudah 1 s

;-----
;   Delay Segment = 4,983 ms
;-----

```

```

Delay_segment:
    mov     R3,#9                ;Isi R3 dengan 9
    mov     R2,#245              ;Isi R2 dengan 245
Tundal:
    djnz    R2,$                 ;R2 = 0 ?, tidak = ulangi instruksi ini
    djnz    R3,Tundal            ;R3 = 0 ?, Tidak = lompat ke Tundal
    ret

;-----
;   Subroutine Debounce = 50ms
;-----
Debounce:
    call    Scan_Tampilan        ;Panggil Scan_Tampilan
    call    Scan_Tampilan        ;Delay 50 ms
    ret

;-----
;   Subroutine Delay 5 ms
;   - Menggunakan timer 1 mode 1 ; - TH0= -5000 dan TL0= -5000
;-----
Delay_5ms:
    clr     tf0
    mov     th0,#high (-5000)    ; Simpan ke timer0
    mov     tl0,#low (-5000)     ; Simpan ke timer0
    setb    tr0                  ; Aktifkan timer0
    jnb     tf0,$                ; Bit overflow = 1 ? Ya = sudah 5 ms
    clr     tr0                  ; Hentikan timer 0
    ret

Cek_Perulangan:
    mov     a,Data_Kecepatan     ;Isi akumulator dengan data di Data_Kecepatan
    mov     b,#10                ;Isi register b dengan angka 10
    div     ab                    ;Skor dibagi 10
    mov     Data_Per_Kec,a        ;Hasil bagi disimpan di Data_Per_Kec
    ret

;-----
;   Proses random low byte alamat data nyala Lampu
;-----
Random:
    clr     a                    ;Nol-kan akumulator
    mov     a,Hasil_random        ;Hasil_random -> ke akumulator
    mov     c,acc.1                ;Simpan nilai bit ke-1 dalam temp random
    mov     Random_temp,#0        ;Random_temp <- dengan 0
    mov     Random_temp.0,c        ;Random_temp <- bit ke 0 dengan c
    mov     c,acc.2                ;Simpan nilai bit ke-2
    push    acc                    ;Angka random di simpan sementara
    clr     a                    ;Nol-kan akumulator
    rlc     a                    ;Nilai bit ke-2 -> akumulator
    xrl     a,Random_temp          ;A= bit 2 xor bit 1
    mov     Random_temp,acc        ;Simpan hasilnya pada temp random
    pop     acc                    ;A= angka random
    mov     c,acc.3                ;Simpan bit ke-3
    push    acc                    ;Simpan angka random sementara
    clr     a                    ;Nol-kan akumulator
    rlc     a                    ;Simpan bit ke-3 dalam akumulator
    xrl     a,Random_temp          ;A= bit ke-3 xor bit ke-2 xor bit ke-1
    mov     Random_temp,acc        ;Simpan hasilnya pada temp random
    pop     acc                    ;A=angka random
    mov     c,acc.7                ;Simpan bit ke-7
    push    acc                    ;Simpan angka random sementara
    clr     a                    ;Nol-kan akumulator
    rlc     a                    ;Simpan bit ke-7 dalam akumulator
    xrl     a,Random_temp          ;A=bit ke-7 xor bit ke-3 xor bit ke-2 xor bit ke-1
    mov     Random_temp,acc        ;Simpan hasilnya pada temp random
    mov     c,Random_temp.0        ;Masukan nilai A bit ke-0 pada carry
    pop     acc                    ;A=angka random
    rlc     a                    ;Geser ke kiri dengan carry 1 kali
    mov     Hasil_random,a        ;Simpan hasil random untuk di-update
    ret

```

```

;-----
; Subroutine menyalakan Lampu sesuai alamat yang dituju
; -> alamat tujuan = DPTR (16 bit) + Hasil random (8 bit)
;-----
Nyala_Lampu:
    mov     dptr,#Data_random_Lampu ;Data random lampu -> ke DPTR
    movc    a,@a+dptr               ;A = Hasil random + pointer awal nyala Lampu
    mov     Lampu_AC,a               ;Tampilkan pada Lampu
    inc     Jumlah_random            ;Jumlah_random=Jumlah_random + 1
    mov     Data_Lampu_AC,Lampu_AC  ;Simpan data P1
    ret

;-----
; Data 7 segment untuk menampilkan angka 0 sampai 9
;-----
Data_7segment:
    db 11000000b ;data penampil angka 0
    db 11111001b ;data penampil angka 1
    db 10100100b ;data penampil angka 2
    db 10110000b ;data penampil angka 3
    db 10011001b ;data penampil angka 4
    db 10010010b ;data penampil angka 5
    db 10000010b ;data penampil angka 6
    db 11111000b ;data penampil angka 7
    db 10000000b ;data penampil angka 8
    db 10010000b ;data penampil angka 9

;-----
; Data lampu yang akan dirandom disimpan pada prom
;-----
Data_random_Lampu:
;00-07
    00 01 02 03 04 05 06 07
    DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH
;08-0F
    08 09 0A 0B 0C 0D 0E 0F
    DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0FEH
;10-17
    10 11 12 13 14 15 16 17
    DB 0DFH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH
;18-1F
    18 19 1A 1B 1C 1D 1E 1F
    DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH
;20-27
    20 21 22 23 24 25 26 27
    DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH
;28-2F
    28 29 2A 2B 2C 2D 2E 2F
    DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0FDH
;30-37
    30 31 32 33 34 35 36 37
    DB 0F7H,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH
;38-3F
    38 39 3A 3B 3C 3D 3E 3F
    DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0EFH
;40-47
    40 41 42 43 44 45 46 47
    DB 0BFH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH
;48-4F
    48 49 4A 4B 4C 4D 4E 4F
    DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0EFH
;50-57
    50 51 52 53 54 55 56 57
    DB 0FBH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH
;58-5F
    58 59 5A 5B 5C 5D 5E 5F
    DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH
;60-67
    60 61 62 63 64 65 66 67
    DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH
;68-6F
    68 69 6A 6B 6C 6D 6E 6F
    DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0BFH
;70-77
    70 71 72 73 74 75 76 77
    DB 07FH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH
;78-7F
    78 79 7A 7B 7C 7D 7E 7F
    DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0FEH
;80-87
    80 81 82 83 84 85 86 87
    DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH
;88-8F
    88 89 8A 8B 8C 8D 8E 8F
    DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0FEH
;90-97
    90 91 92 93 94 95 96 97
    DB 0EFH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH
;98-9F
    98 99 9A 9B 9C 9D 9E 9F
    DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH

```



```
;A0-A7      A0  A1  A2  A3  A4  A5  A6  A7
DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH
;A8-AF      A8  A9  AA  AB  AC  AD  AE  AF
DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0DFH
;B0-B7      B0  B1  B2  B3  B4  B5  B6  B7
DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH
;B8-BF      B8  B9  BA  BB  BC  BD  BE  BF
DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0FBH
;C0-C7      C0  C1  C2  C3  C4  C5  C6  C7
DB 0EFH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH
;C8-CF      C8  C9  CA  CB  CC  CD  CE  CF
DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0EFH
;D0-D7      D0  D1  D2  D3  D4  D5  D6  D7
DB 07FH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH
;D8-DF      D8  D9  DA  DB  DC  DD  DE  DF
DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH
;E0-E7      E0  E1  E2  E3  E4  E5  E6  E7
DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH
;E8-EF      E8  E9  EA  EB  EC  ED  EE  EF
DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0F7H
;F0-F7      F0  F1  F2  F3  F4  F5  F6  F7
DB 07FH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH
;F8-FF      F8  F9  FA  FB  FC  FD  FE  FF
DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,0F7H
end
```