BAB V

Kesimpulan dan Saran

5.1 Kesimpulan

Kesimpulan yang dapat diambil oleh penulis terhadap aplikasi SunTracker adalah:

1. Pengontrolan pada mikrokontroler AT89C51 dilakukan dengan memberikan perintah dari PC, ke sistem mikrokontroler AT89C51 melalui saluran serial.
2. Penentuan arah dari posisi matahari dapat dilakukan dengan memberikan LDR sensor, dimana nilai perubahan resistansi yang didapatkan dapat dimonitor.
3. Desain alat monitoring posisi matahari berbasis AT89C51 terkoneksi dengan computer melalui RS-232, terdiri dari perangkat keras dan perangkat lunak. Perangkat keras meliputi mikrokontroler, sensor, motor stpper, catu daya dan komputer. Sedangkan perangkat lunak meliputi pemrograman assembler pada mikrokontroler dan pemrograman C#.Net 2005 pada komputer.
4. Unjuk kerja dari alat monitoring posisi matahari berbasis AT89C51 terkoneksi dengan komputer melalui RS-232, yaitu; kecepatan pengiriman data

ke computer sebesar 9600bps dan catu daya yang dipakai sebesar +5V DC.

5. Pergerakan dari posisi matahari akan didapatkan dari perubahan resistansi sensor LDR, dimana sensor ini diletakkan pada panel surya, yang secara otomatis akan mengikuti perubahan posisi matahari.
6. Sistem yang dibuat dapat menampilkan perubahan data sensor berupa grafik dan perubahan posisi matahari berupa simulasi perubahan posisi matahari yang dapat dimonitoring oleh *user desktop*.

5.2. Saran

Pada pembuatan SunTracker berbasis mikrokontroler AT89C51 dengan bantuan PC ini penulis menyarankan agar setiap energi yang didapatkan dari *solar cell* dapat digunakan untuk menghidupkan sistem ini sendiri, tanpa perlu adanya catu daya, sehingga sistem ini dapat bekerja secara mandiri. Untuk menggantikan fungsi komputer sebagai alat pengonkontrol sistem, maka sistem Mikrokontroller merupakan salah satu solusi yang dapat digunakan.

Daftar Pustaka

Agfianto Eko Putra, Belajar Mikrokontroler AT89C51/52/55 Teori dan Aplikasi, Gava Media Yogyakarta, Agustus 2002

Bryan , Programmable Controllers theory and Implementation, Industrial Text Company L.A, 1997.

Nalwan, P.A., Teknik Antarmuka dan Pemrograman.

Okan Bingol - Ahmet Altintas - Yusuf Oner ,Microcontroller Based Solar-Tracking System And Its Implementation, Gelis Tarihi,23-06-2005.

www.ilmukomputer.com

www.klik-kanan.com

www.teknisoft.net/project/skpl.dpr

www.AllDatasheet.com

www.MyTutorialCafe.com

LAMPIRAN

SKPL

SPESIFIKASI KEBUTUHAN PERANGKAT LUNAK

Pembangunan Aplikasi Otomatisasi Posisi Panel Sel Surya untuk Mendapatkan Energi Cahaya Matahari Optimal

(Sun Tracker)

Dipersiapkan oleh:

Ign. Hendra Adi W.

No.Mhs : 03755

Program Studi Teknik Informatika – Fakultas Teknologi Industri

Universitas Atma Jaya Yogyakarta

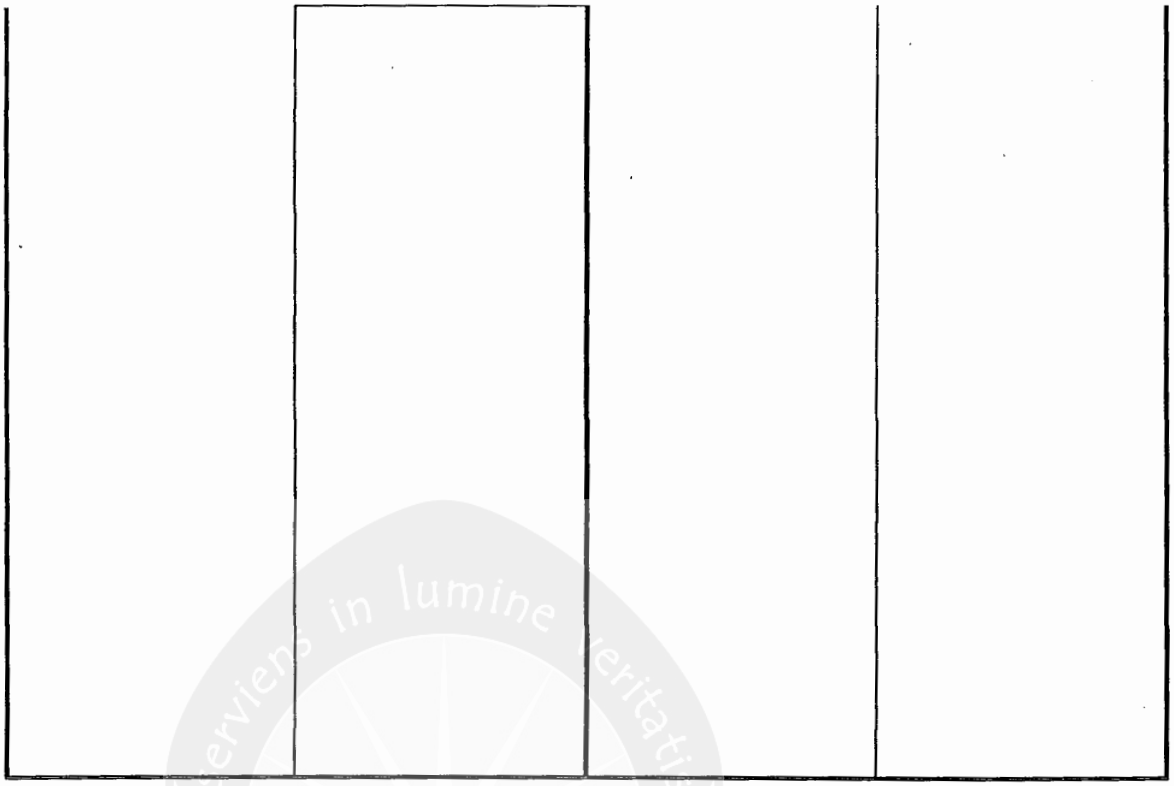
DAFTAR PERUBAHAN

Revisi	Deskripsi
A	1.Penghapusan fungsi login, pada daftar isi 2.Pemberian gambar arsitektur system pada 2.1 Perspektif Produk 3.Penggantian "administrator" menjadi "user" pada 2.3 Karakteristik Pengguna 4.Pembetulan gambar 3.1 Spesifikasi Kebutuhan Fungsionalitas 5.Pembetulan Actors dan Precondition, Includes pada 3.1.1 dan 3.1.2 6.Penghapusan Interaction Diagram untuk Use Case Login 7.Pembetulan Collaboration Diagram untuk Motor Stepper dan LDR Sensor
B	
C	
D	
E	
F	

INDEX TGL	-	A	B	C	D	E	F	G
Ditulis oleh								
Diperiksa oleh								
Disetujui oleh								

Daftar Halaman Perubahan

Halaman	Revisi	Halaman	Revisi



Daftar Isi

Daftar Gambar

Daftar Tabel

1 Pendahuluan

1.1 Tujuan

Dokumen Spesifikasi Kebutuhan Perangkat Lunak (SKPL) ini merupakan dokumen spesifikasi kebutuhan perangkat lunak SunTracker (Pembangunan Aplikasi Otomatisasi Posisi Panel Sel Surya untuk Mendapatkan Energi Cahaya Matahari Optimal) untuk mendefinisikan kebutuhan perangkat lunak, yang meliputi antarmuka eksternal, dan atribut, serta mendefinisikan fungsi perangkat lunak, juga mendefinisikan batasan perancangan perangkat lunak.

1.2 Lingkup Masalah

Perangkat Lunak SunTracker dikembangkan dengan tujuan untuk :

1. Melakukan monitoring terhadap sistem dari perangkat keras SunTracker.
2. Monitoring data berupa data perubahan arah motor stepper terhadap titik-titik *azimuth* dan *tilt*.
3. Monitoring terhadap nilai-nilai LDR sensor yang berupa data grafik.

1.3 Definisi, Akronim dan Singkatan

Daftar definisi akronim dan singkatan :

Keyword/Phrase	Definisi
SKPL	Merupakan spesifikasi kebutuhan dari perangkat lunak yang akan dikembangkan.
UC- SunTracker –XX	Kode yang merepresentasikan kebutuhan pada SunTracker (Pembangunan Aplikasi Otomatisasi Posisi Panel Sel Surya untuk Mendapatkan Energi Cahaya Matahari Optimal).
ERD	Entity Relationship Diagram merupakan teknis grafis/diagram yang menggambarkan objek dan hubungan antar objek.

1.4 Referensi

Referensi yang digunakan pada perangkat lunak tersebut adalah:

1. GLO1, *Spesifikasi Kebutuhan Perangkat Lunak*, Jurusan Teknik Informatika – UAJY
2. Fowler, Martin., *UML Distilled Edisi 3,* Andi Yogyakarta, 2005.

1.5 Deskripsi umum (Overview)

Dokumen SKPL ini dibagi menjadi empat bab. Bab pertama adalah **Pendahuluan,** yang berisi tentang deskripsi dokumen. Bab kedua adalah **Deskripsi Keseluruhan,** yang berisi penjelasan secara umum mengenai sistem yang akan dikembangkan meliputi fungsi-fungsi dari sistem, karakteristik pengguna, batasan dan asumsi yang diambil dalam pengembangan perangkat lunak. Bab ketiga adalah **Spesifikasi Rinci Kebutuhan,** yang berisi penjelasan tentang kebutuhan sistem yang akan dikembangkan secara lebih rinci. Bab keempat adalah **Realisasi Use Case**, yang berisi realisasi use case dalam tahap analisis (konseptual), yang akan digunakan sebagai dasar realisasi use case pada tahap desain.

2 Deskripsi Kebutuhan

2.1 Perspektif produk

SunTracker adalah perangkat lunak yang dikembangkan untuk membantu proses monitoring terhadap SunTracker sistem yang dalam hal ini berupa perangkat keras. Monitoring dilakukan oleh operator dari SunTracker saistem. Data pertama yang didapatkan berupa data dari pergerakan motor stepper, data motor stepper I (pergerakan dari utara-selatan), data motor stepper II (pergerakan dari barat-timur). Data kedua yang didapatkan berupa data-data LDR sensor, dimana data ini berupa grafik.

2.2 Fungsi Produk

Fungsi produk perangkat lunak SunTracker berdasarkan user adalah sebagai berikut :

2.2.1 Use Case : MonitoringMotorStepper

Use Case ini digunakan oleh Aktor (Administrator) untuk melakukan monitoring data-data pergerakan dari motor stepper.

Lihat : **Spesifikasi Use Case : MonitoringMotorStepper**

2.2.2 Use Case : MonitoringLDRSensor

Use Case ini digunakan oleh Aktor (Administrator) untuk melakukan monitoring data-data yang didapatkan dari LDR sensor.

Lihat : **Spesifikasi Use Case : MonitoringLDRSensor.**

2.3 Karakteristik Pengguna

Pengguna perangkat lunak SunTracker tersebut adalah bagian user dengan karakteristik sebagai berikut :

- User
  1. Memahami pengoperasian komputer secara aktif
  2. Memahami sistem komputer tempat perangkat lunak dijalankan

2.4 Batasan-batasan

Batasan-batasan dalam pengembangan perangkat lunak SunTracker tersebut adalah :

1. Kebijaksanaan Umum

   Berpedoman pada tujuan dari pengembangan perangkat lunak SunTracker.

2. Keterbatasan perangkat keras

   Dapat diketahui kemudian setelah sistem ini berjalan (sesuai dengan kebutuhan).

3. Kebutuhan keandalan

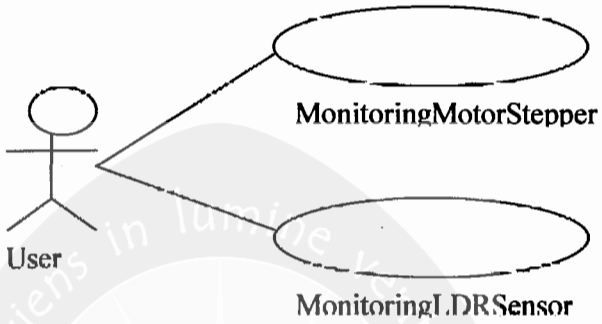
   Pengembangan perangkat lunak ini dibatasi pada kemudahan penggunaan dan kecepatan dalam proses pengolahannya.

2.5 Asumsi dan Ketergantungan

SunTracker akan dikembangkan dengan mengunakan bahasa pemrograman Microsoft Visual C#.NET sebagai antar muka, dan juga SQL Server 2000 sebagai basis data. Oleh karena itu *SunTracker* harus dijalankan dengan sistem operasi Windows 2000 Service Pack 2.

3. Deskripsi Rinci Kebutuhan

3.1 Spesifikasi Kebutuhan Fungsionalitas



Gambar 3.1 Use Case Diagram

3.1.1 Spesifikasi Use Case : MonitoringMotorStepper

Tabel 3.1 Spesifikasi Use Case : MonitoringMotorStepper

Use Case ID	UC-SunTracker-02
Use Case Name	MonitoringMotorStepper
Use Case Type	Essential
Priority	High
Actors	User
Description	Use case ini digunakan oleh aktor untuk melakukan monitoring terhadap data-data motor stepper.
Preconditions	-
Basic Path	Monitoring Motor Stepper 1. Sistem menampilkan antar muka untuk monitoring data motor stepper.
Alternative Paths	-
Postconditions	-
Exception Paths	-

Extends	-
Includes	-
Bussiness Role	-

3.1.2 Spesifikasi Use Case : MonitoringLDRSensor

Tabel 3.2 Spesifikasi Use Case : MonitoringLDRSensor

Use Case ID	UC-SunTracker-03
Use Case Name	MonitoringLDRSensor
Use Case Type	Essential
Priority	High
Actors	User
Description	Use case ini digunakan oleh aktor untuk melakukan monitoring terhadap data-data dari LDR sensor yang berupa data grafik.
Preconditions	-
Basic Path	Monitoring data-data LDR Sensor 1. Sistem menampilkan antar muka untuk monitoring data-data dari LDR sensor.
Alternative Paths	-
Postconditions	-
Exception Paths	-
Extends	-
Includes	-
Bussiness Role	-

3.2 Spesifikasi Kebutuhan Non-Fungsionalitas

3.2.1 Kebutuhan Antarmuka Eksternal

Kebutuhan antar muka eksternal pada sistem SunTracker mencakup kebutuhan antarmuka pemakai, antarmuka perangkat keras dan antarmuka perangkat lunak.

3.2.2 Antarmuka Pemakai

Pemakai berinteraksi langsung dengan sistem SunTracker dengan antarmuka berbasis GUI. Piranti masukan yang digunakan untuk memasukkan data masukan adalah *keyboard* dan *mouse*. Sedangkan keluaran dari sistem berupa data-data yang disimpan dalam basis data dan dalam bentuk file serta akan ditampilkan langsung ke layar monitor.

3.2.3 Antarmuka Perangkat Keras

Antarmuka perangkat keras yang dibutuhkan dalam penggunaan perangkat lunak SunTracker adalah:

- *PC IBM Compatible Pentium 4*
- *Keyboard*
- *Mouse*
- *Printer*

3.2.4 Antarmuka Perangkat Lunak

Perangkat lunak yang dibutuhkan untuk mendukung berjalannya perangkat lunak SunTracker adalah:

- Nama : *Windows Me/NT/2000/XP*
  Sumber : *Microsoft*
  Fungsi : Sistem Operasi Komputer
- Nama : Visual *C# .NET*
  Sumber : *Microsoft*
  Fungsi : *Tool* perancang antarmuka aplikasi
- Nama : *Sql Server 2000*
  Sumber : *Microsoft*
  Fungsi : Basis data

4. Realisasi Use Case

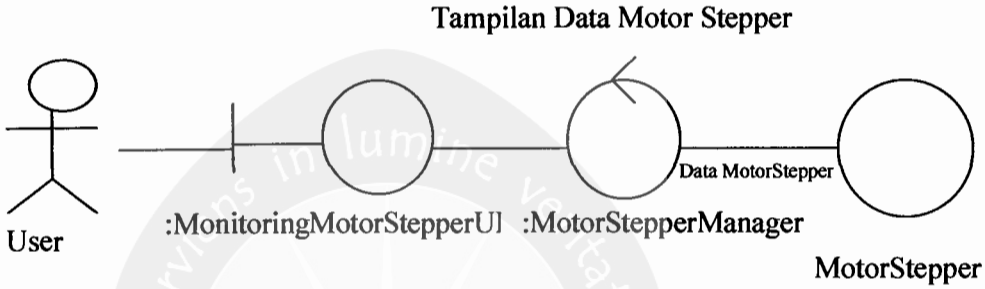
4.1 Basis Data

SUNPOSITION	
ID	integer
TIME	time
ASIMUTH	decimal(10)
TILTH	decimal(10)
VALUE	decimal

4.2 Interaction Diagram

4.2.1 Analysis Collaboration Diagram : Use Case Monitoring Motor Stepper

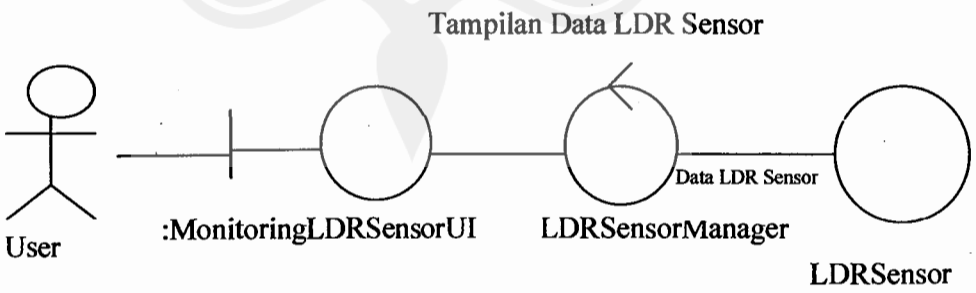
4.2.2.1 Monitoring Motor Stepper



Gambar 4.1 Analysis Collaboration Diagram : Use Case Monitoring Motor Stepper

4.2.2 Analysis Collaboration Diagram : Use Case Monitoring LDR Sensor

4.2.2.1 Monitoring LDR Sensor



Gambar 4.2 Analysis Collaboration Diagram : Use Case Monitoring LDR Sensor

DPPL

DESKRIPSI PERANCANGAN PERANGKAT LUNAK

Pembangunan Aplikasi Otomatisasi Posisi Panel Sel Surya untuk Mendapatkan Energi Cahaya Matahari Optimal (Sun Tracker)

Dipersiapkan oleh:

Ign. Hendra Adi W.

No.Mhs : 03755

Program Studi Teknik Informatika – Fakultas Teknologi Industri

Universitas Atma Jaya Yogyakarta

		Revisi		*Tgl : 30-05-2007*

DAFTAR PERUBAHAN

Revisi	Deskripsi
A	
B	
C	
D	
E	
F	

INDEX TGL	-	A	B	C	D	E	F	G
Ditulis oleh								
Diperiksa oleh								
Disetujui oleh								

DAFTAR HALAMAN PERUBAHAN

Halaman	Revisi	Halaman	Revisi

PEMBANGUNAN APLIKASI OTOMATISASI POSISI PANEL SEL SURYA UNTUK MENDAPATKAN ENERGI CAHAYA MATAHARI OPTIMAL

(SUN TRAKCER)

Pendahuluan

1.1 Tujuan

Dokumen DPPL ini dibuat untuk menyediakan deskripsi lengkap mengenai desain Pembangunan Aplikasi Otomatisasi Posisi Panel Sel Surya untuk Mendapatkan Energi Cahaya Matahari Optimal(Sun Tracker), versi 1.0. Dokumen ini khususnya ditujukan untuk pembuat perangkat lunak, dan orang orang lain yang tertarik untuk mengembangkan perangkat lunak ini lebih lanjut.

1.2 Lingkup Dokumen

Dokumen DPPL ini menyediakan deskripsi lengkap perancangan perangkat lunak untuk SunTracker, versi 1.0. Perancangan ini merupakan arsitektur sistem yang dijelaskan melalui perancangan class/modul, detail operasi apa yang akan dilakukan oleh masing-masing class/modul, dan layout database. Metodologi yang digunakan dalam perancangan adalah metode USDP (Unified Software Development Process) dari Rational Software.

1.3 Istilah dan singkatan

Untuk definisi istilah dan singkatan yang digunakan dalam dokumen ini dapat mengacu pada **Apendiks A : Daftar Istilah dan Singkatan .**

1.4 Referensi

- Martin Fowler, Kendall Scott. *UML Distilled – Second Edition*, Adisson Wesley, 1999.

1.5 Deskripsi Umum Dokumen

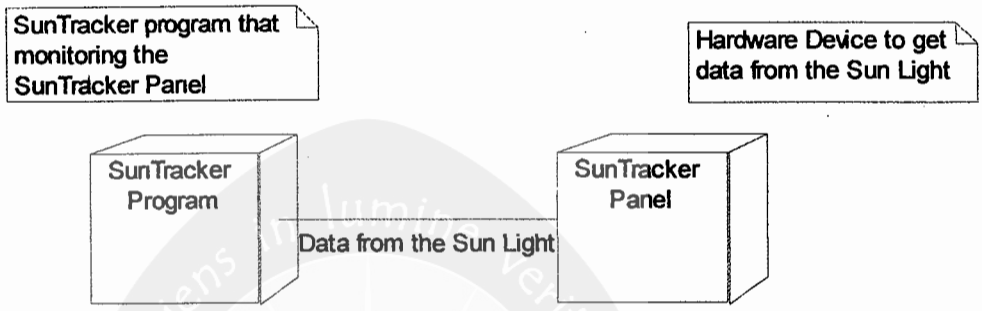
Dokumen ini terdiri dari empat bab. Bab pertama adalah **Pendahuluan**, yang berisi deskripsi dokumen. Bab kedua adalah **Deskripsi Perancangan Arsitektural**, yang berisi deskripsi arsitektur sistem. Bab ketiga adalah **Deskripsi Perancangan Persistent Data**, yang berisi deskripsi data-data yang akan disimpan pada persistent storage. Bab keempat adalah **Deskripsi Perancangan**

Antarmuka, yang berisi deskripsi rancangan GUI yang digunakan sistem untuk berinteraksi dengan user.

2. Deskripsi Perancangan Arsitektural

2.1 Deployment Diagram

Deployment diagram ini dibuat untuk menunjukkan semua node pada sistem, hubungan di antara mereka, dan proses yang akan dijalankan di masing-masing node.



Gambar 2.1 Deployment Diagram SunTracker

2.1.1 Node : SunTracker Program

Node ini merupakan aplikasi yang berada dikomputer *user*, yang digunakan untuk melakukan monitoring terhadap perubahan dari posisi matahari, lewat aplikasi simulasi pergerakan. Proses yang ada di dalamnya adalah :

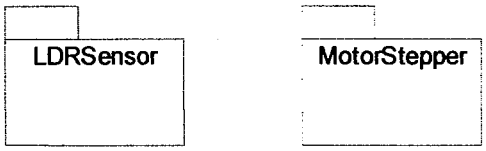
1. SunTracker Program, aplikasi ini mendapatkan data dari SunTracker Panel, berupa data perubahan resistansi sensor, dan juga perubahan data DC motor, untuk disimulasikan.

2.1.2 Node : SunTracker Panel

Node ini merupakan perangkat keras yang digunakan oleh untuk menangani pencarian posisi matahari(*changing the sun state*). Proses yang ada di dalamnya adalah :

1. Melakukan penerimaan data berupa *infrared* dari cahaya matahari, yang kemudian diolah, lalu dikirimkan ke SunTracker Program.

2.2 Design Kelas



Gambar 2.2 Package Depencies SunTracker

2.2.1 Pengantar

Nama class yang digunakan dalam design class adalah nama class yang valid, termasuk nama packagenya. Untuk class-class yang berasal dari framework .Net juga digunakan nama class dengan package lengkap, misalnya System.IO.File. Untuk penjelasan tipe data yang utuh dapat dilihat pada bagian deskripsi class, sedangkan gambar design class tidak akan menggunakan nama package yang lengkap. Stereotype yang digunakan dalam design class adalah :

- << boundary >>

  Boundary class merupakan class yang berfungsi untuk menghubungkan sistem dengan user di luar sistem.

- << control >>

  Control class adalah suatu class yang objek-nya melakukan interaksi antar sekelompok objek lain. Control class biasanya memiliki karakteristik yang spesifik untuk satu use case, dan objek class ini biasanya hanya aktif pada realisasi use case.

- << entity >>

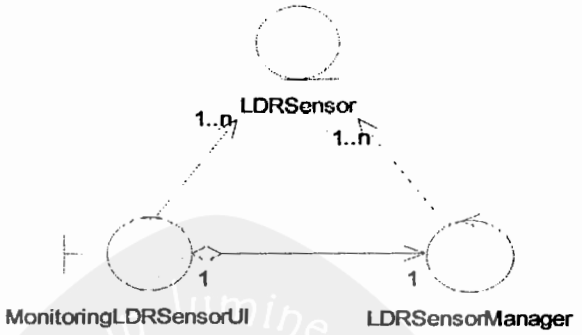
  Entity class adalah class yang bersifat pasif, dalam arti class tersebut tidak memulai interaksi dengan class lain. Entity class ini biasanya merepresentasikan suatu objek yang disimpan dalam persistent storage. Untuk hirarki class design berdasarkan package, dapat dilihat pada

2.2.2 Package Dependencies

2.2.3 Package LDRSensor Management (SunTracker.LDRSensor)

Package ini menyediakan class-class yang digunakan oleh program untuk mengelola hal-hal yang berhubungan dengan data sensor LDR.

2.2.3.1 Class Diagram Package SunTracker.LDRSensor



Gambar 2.3 Class Diagram Package SunTracker.LDRSensor

2.2.3.2 Class SunTracker.LDRSensor.LDRSensor

<<entity>> LDRSensor
- NoSensor : string - Direction : string - Resistance : string
+ LDRSensor() + LDRSensor(NoSensor : string, Direction : string, Resistance : string) + getNoSensor() : string + getDirection() : string + getResistance() : string + setNoSensor() : string + setDirection() : string + setResistance() : string

Gambar 2.4 Class SunTracker.LDRSensor.LDRSensor

Deskripsi

Class ini merepresentasikan nilai dari resistansi LDR Sensor dari sistem SunTracker

Atribut

- - NoSensor : string

merepresentasikan data LDR berasal dari sensor yang nomor berapa (No 1 : Utara, No 2 : Selatan, No 3 : Barat, No 4 : Timur)

- - Direction : string

  merepresentasikan Arah posisi dari panel, apakah ke arah Utara, Selatan, Barat, atau Timur.

- Resistance : string

  merepresentasikan nilai perubahan hambatan dari LDR Sensor, seiring dengan perubahan arah cahaya matahari.

Method

- + LDRSensor ()

  Default Konstruktor class SunTracker.LDRSensor.LDRSensor. Membuat instance baru SunTracker.LDRSensor.LDRSensor. LDRSensor tanpa atribut terdefinisi.

- + LDRSensor (NoSensor : string, Direction : string, Resistance : string)

  Konstruktor class SunTracker.LDRSensor.LDRSensor. Membuat instance baru SunTracker.LDRSensor.LDRSensor dengan atribut terdefinisi.

Method-method berikut merupakan accessor untuk atribut class SunTracker.LDRSensor.LDRSensor

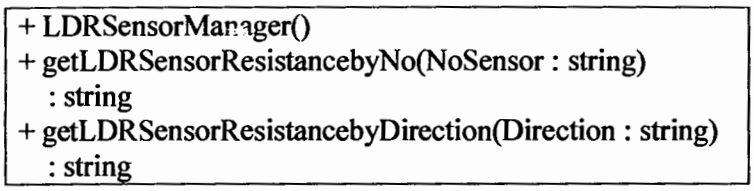
- + getNoSensor () : string
- + getDirection () : string
- + getResistance () : string

Method-method berikut merupakan mutator untuk atribut class SunTracker.LDRSensor.LDRSensor

- + setNoSensor () : string
- + setDirection () : string
- + setResistance () : string

2.2.3.3 Class SunTracker.LDRSensor.LDRSensorManager

<<control>> LDRSensorManager



Gambar 2.5 Class SunTracker.LDRSensor.LDRSensorManager

Deskripsi

Class yang berperan sebagai control class untuk masalah yang berhubungan dengan LDR Sensor pada Sun Tracker system. Masalah ini antara lain :

- mendapatkan nilai data perubahan resistansi LDR sensor berdasarkan nomor sensor.
- mendapatkan nilai data perubahan resistansi LDR sensor berdasarkan arah (Utara, Selatan, Barat, Timur).

Atribut

-

Method

- + LDRSensorManager()
  Konstruktor class SunTracker.LDRSensor.LDRSensorManager. Membuat instance baru SunTracker.LDRSensor.LDRSensorManager.
- + getLDRSensorResistancebyNo(NoSensor : string) : string
  Digunakan untuk mendapatkan nilai resistansi dari LDR Sensor berdasarkan nomor sensor yang diminta.
  Parameters:
  NoSensor - string yang ingin dicari data nilainya.
  Returns:
  Mengembalikan nilai angka (nilai perubahan resistansi LDR Sensor) yang bertipe string.
- + getLDRSensorResistancebyDirection(Direction : string) : string
  Digunakan untuk mendapatkan nilai resistansi dari LDR Sensor berdasarkan arah, yaitu apakah arah Utara, Selatan, Barat, Timur.

Parameters:

Direction - string yang ingin dicari data nilainya.

Returns:

Mengembalikan nilai angka (nilai perubahan resistansi LDR Sensor) yang bertipe string.

2.2.3.4 Class SunTracker.LDRSensor.MonitoringLDRSensorUI

<<boundary>> MonitoringLDRSensorUI
- ldrsensorMan : LDRSensorManager
+ MonitoringLDRSensorUI()

Gambar 2.6 Class SunTracker.LDRSensor.MonitoringLDRSensorUI

Deskripsi

Class ini merupakan GUI untuk melakukan monitoring LDR Sensor dari SunTracker *System*, yang meliputi penampilan perubahan nilai resistansi data dari ke empat sensor yang ada, dalam hal ini sensor untuk arah Utara, Selatan, Barat dan Timur.

Atribut

- - ldrsensorMan : SunTracker.LDRSensor.LDRSensorManager

  Merupakan control class untuk mendapatkan nilai data perubahan resistansi LDR sensor berdasarkan nomor sensor, atau juga bisa berdasarkan arah yang diinginkan.

Method

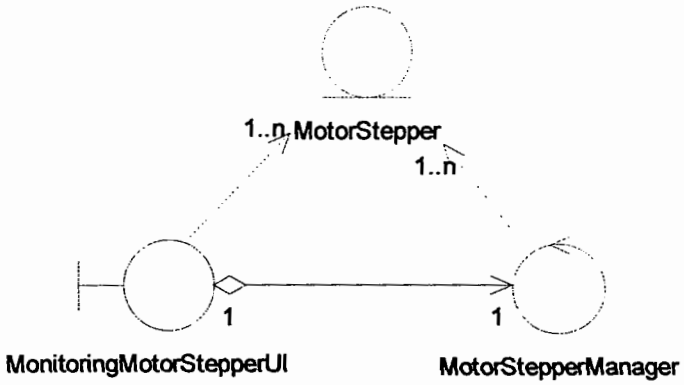
- + MonitoringLDRSensorUI()

  Buat instance baru SunTracker.LDRSensor.MonitoringLDRSensorUI.

2.2.4 Package MotorStepper (SunTracker.MotorStepper)

Package ini menyediakan class-class yang diperlukan untuk mendapatkan perubahan arah dari posisi matahari, dengan menggerakkan motor stepper tegak lurus dengan matahari.

2.2.4.1 Class Diagram Package SunTracker.MotorStepper



Gambar 2.8 Class Diagram Package SunTracker.MotorStepper

2.2.4.2 Class SunTracker.MotorStepper.MotorStepper

<<entity>> MotorStepper
- MotorStepperI : boolean - MotorStepperII : boolean - DetakMaju : boolean - DetakMundur : boolean
+ MotorStepper () + MotorStepper(DataMS : string) + getMotorStepperI() : boolean + getMotorStepperII() : boolean + getDetakMaju() : boolean + getDetakMaju() : boolean + setMotorStepperI(MSI : boolean) + setMotorStepperII(MSII : boolean) + setDetakMaju(DetakMaju : boolean) + setDetakMundur(DetakMundur : boolean)

Gambar 2.9 Class SunTracker.MotorStepper.MotorStepper

Deskripsi

Merepresentasikan nilai dari pergerakan dari motor stepper dari sistem SunTracker

Atribut

- - MotorStepperI : boolean

  Merepresentasikan pergerakan motor stepper I (berupa pergerakan dari Utara ke Selatan, atau sebaliknya) apakah diaktifkan atau tidak, jika aktif bernilai *true*, sedangkan jika tidak aktif bernilai *false*.

- - MotorStepperII : boolean

  Merepresentasikan pergerakan motor stepper II (berupa pergerakan dari Timur ke Barat) apakah diaktifkan atau tidak, jika aktif bernilai *true*, sedangkan jika tidak aktif bernilai *false*

- - DetakMaju : boolean

  Merepresentasikan pergerakan dari motor stepper apakah maju atau diam, bergantung dari motor stepper mana yang aktif, jika motor stepper I yang aktif, jika nilainya *true* berarti mengalami pergerakan ke arah selatan(dari arah Utara), sedangkan jika *false* berarti motor stepper I diam. Hal yang sama terjadi untuk motor stepper II, jika ternyata yang aktif motor stepper II, maka jika detak maju bernilai *true*, maka motor stepper II dianggap telah mengalami pergerakan dari arah Timur ke arah Barat, namun jika bernilai *false*, maka motor stepper II ini tidak mengalami pergerakan(diam).

- - DetakMundur : boolean

  Merepresentasikan pergerakan dari motor stepper apakah mundur atau diam, bergantung dari motor stepper mana yang aktif, jika motor stepper I yang aktif, jika nilainya *true* berarti mengalami pergerakan ke arah Utara(dari arah Selatan), sedangkan jika *false* berarti ke motor stepper I diam. Hal yang sama terjadi untuk motor stepper II, jika ternyata yang aktif motor stepper II, maka jika detak mundur bernilai *true*, maka motor stepper II dianggap telah mengalami pergerakan dari arah Barat ke arah Timur, namun jika bernilai *false*, maka motor stepper II ini tidak mengalami pergerakan(diam).

Method

- + MotorStepper()

  Membuat instance baru MotorStepper tanpa atribut terdefinisi

- + MotorStepper(DataMS : string)

  Membuat instance baru MotorStepper dengan atribut terdefinisi, DataMS ini berisi string 4 bit data, detailnya seperti dibawah ini.

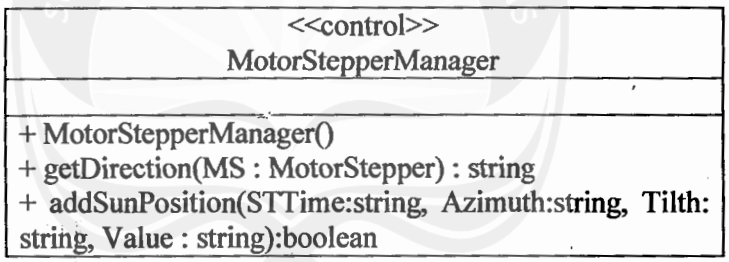
Method-method berikut merupakan accessor untuk atribut class siparsel.data.kelas

- + getMotorStepperI() : boolean
- + getMotorStepperII() : boolean
- + getDetakMaju() : boolean
- + getDetakMundur() : boolean

Method-method berikut merupakan mutator untuk atribut class siparsel.data.kelas

- + setMotorStepperI(MSI : boolean) : boolean
- + setMotorStepperII(MSII : boolean) : boolean
- + setDetakMaju(DetakMaju : boolean) : boolean
- + setDetakMundur(DetakMundur : boolean) : boolean

2.2.4.3 Class SunTracker.MotorStepper.MotorSteperManager



Gambar 2.10 Class SunTracker.MotorStepper. MotorStepperManager

Deskripsi

Class yang berperan sebagai control class untuk masalah yang berhubungan dengan MotorStepper pada Sun Tracker system. Masalah ini antara lain :

- mendapatkan nilai data perubahan pergerakan dari motor stepper, apakah pergerakan ke Utara, Selatan, Timur, atau ke Barat.

Atribut

-

Method

- + MotorStepperManager()

Konstruktor class SunTracker.MotorStepper.MotorStepperManager.
Membuat instance baru
SunTracker.MotorStepper.MotorStepperManager.

- + getDirection(MS : MotorStepper) : string
  Digunakan untuk mendapatkan data nilai perubahan dari arah posisi matahari, berdasarkan pergerakan dari motor stepper.
  Parameters:
  MS - MotorStepper parameter data yang akan digunakan untuk bisa dilakukan kalkulasi.
  Returns:
  Mengembalikan nilai angka (nilai pergerakan motor stepper), apakah bergerak dari Utara ke Selatan(SMI) atau sebaliknya dan juga pergerakan dari arah Timur ke Barat(SMII) atau sebaliknya, dan dengan nilai *Azimuth* dan *Tilth*-nya juga, yang bertipe string.
- + addSunPosition(STTime:string, Azimuth:string, Tilth: string, Value : string):boolean
  Digunakan untuk memasukkan data perbubahan posisi matahari ke dalam basis data.
  Parameters:
  STTime – string parameter data yang digunakan untuk mencatat waktu perubahan posisi matahari.
  Azimuth – string parameter data yang digunakan untuk mencatat posisi secara azimuth dari matahari.
  Tilth – string parameter data yang digunakan untuk mencatat posisi secara tilth dari matahari.
  Value – string parameter data yang digunakan untuk menyimpan nilai data.
  Returns:
  Mengembalikan nilai true jika data telah berhasil diinputkan

2.2.4.4 Class SunTracker.MotorStepper.MonitoringMotorStepperUI

<<boundary>> KamarManager
- MSManager : MontorStepperManager
+ MonitoringMotorStepperUI()

Gambar 2.11 Class SunTracker.MotorStepper.MonitoringMotorStepperUI

Deskripsi

Class ini merupakan GUI untuk melakukan monitoring Motor Stepper dari SunTracker *System*, yang meliputi penampilan perubahan data dari pergerakan motor stepper berdasarkan perubahan nilai resistansi data dari ke empat sensor yang ada, dalam hal ini sensor untuk arah Utara, Selatan, Barat dan Timur, yang berupa simulasi.

Atribut

- - MSManager : SunTracker.MotorStepper.MotorStepperManager

  Merupakan control class untuk mendapatkan nilai data perubahan pergerakan dari motor stepper berdasarkan perubahan resistansi LDR sensor.

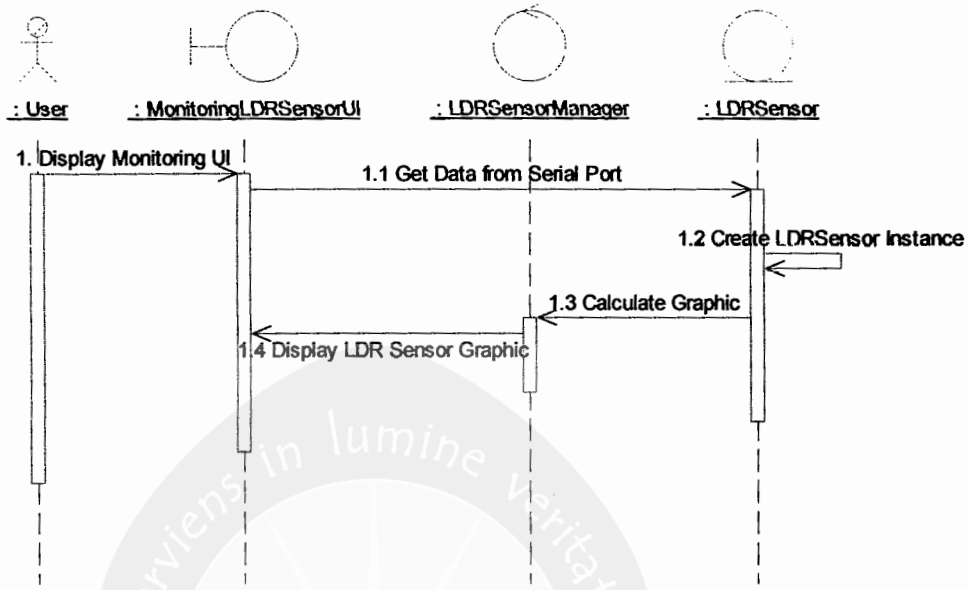
Method

- + MonitoringMotorStepperUI()

  Membuat instance baru SunTracker.MotorStepper.MonitoringMotorStepperUI.

2.3 Realisasi Use Case

2.3.1 Use Case : Monitoring LDR Sensor

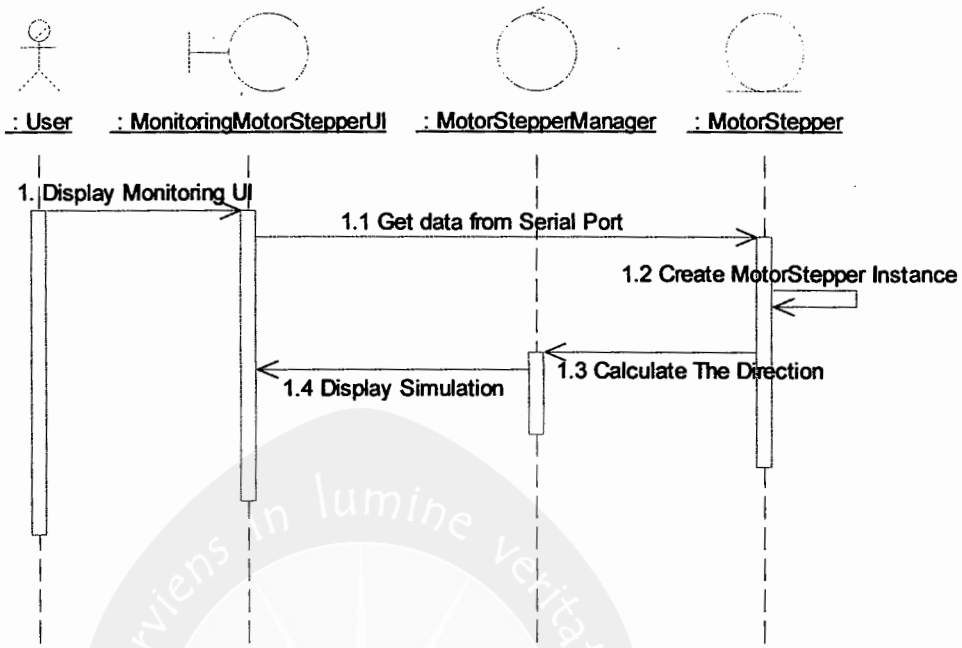


Gambar 2.37 Design sequence diagram use case Monitoring LDR Sensor

Flow of events :

1. User menampilkan antarmuka untuk Monitoring LDR Sensor, yaitu boundary class MonitoringLDRSensorUI.
   1.1 System secara otomatis akan mengambil data dari serial port, dimana data LDR Sensor dikirimkan oleh SunTracker Panel ke port serial (Rx).
   1.2 Menciptakan sebuah objek LDRSensor berdasarkan data yang didapatkan dari serial port.
   1.3 Melakukan perhitungan terhadap data yang didapatkan, untuk bisa dibuat sebuah grafik, dimana akan ditampilkan 4 buah grafik berdasarkan 4 buah sensor.
   1.4 Menampilkan data kalkulasi perhitungan, yang ditampilan berupa grafik

2.3.2 Use Case : Monitoring Motor Stepper



Gambar 2.36 Design sequence diagram use case Monitoring Motor Stepper

Flow of events :

1. User menampilkan antarmuka untuk monitoring motor stepper, yaitu boundary class MonitoringMotorStepperUI.
   1.1 System secara otomatis akan mengambil data dari serial port, dimana data pergerakan motor stepper dikirimkan oleh SunTracker Panel ke port serial (Rx).
   1.2 Menciptakan sebuah objek MotorStepper berdasarkan data yang didapatkan dari serial port.
   1.3 Melakukan perhitungan terhadap data yang didapatkan, untuk bisa ditentukan arah pergerakan dari posisi matahari.
   1.4 Menampilkan data kalkulasi perhitungan, yang berupa simulasi pergerakan, serta melakukan penyimpanan data ke dalam basis data nilai posisinya.

3. Deskripsi Perancangan Persintent Data

3.1 Basis Data

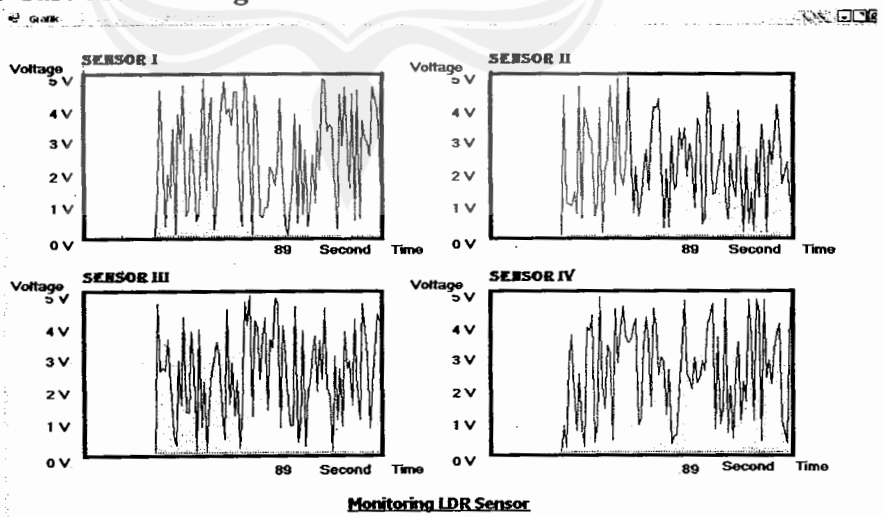
SUNPOSITION	
ID	integer
TIME	time
ASIMUTH	decimal(10)
TILTH	decimal(10)
VALUE	decimal

3.1.1 Tabel SunPosition

Ficld	Tipe Data	NULL	Default	Deskripsi
ID	Int	No		Merupakan no urut dari data yang disimpan
TIME	Time	No		Merupakan waktu pencatatan dari posisi terakhir matahari
AZIMUTH	Decimal	No		Letak azimuth dari matahari
TILTH	Decimal	No		Letak tilth dari matahari
VALUE	Decimal	No		Nilai yang akan disimpan

4. Deskripsi Perancangan Antarmuka

4.1 Use Case : Monitoring LDR Sensor



Gambar 4.1 Rancangan Antarmuka use case LDR Sensor

Deskripsi

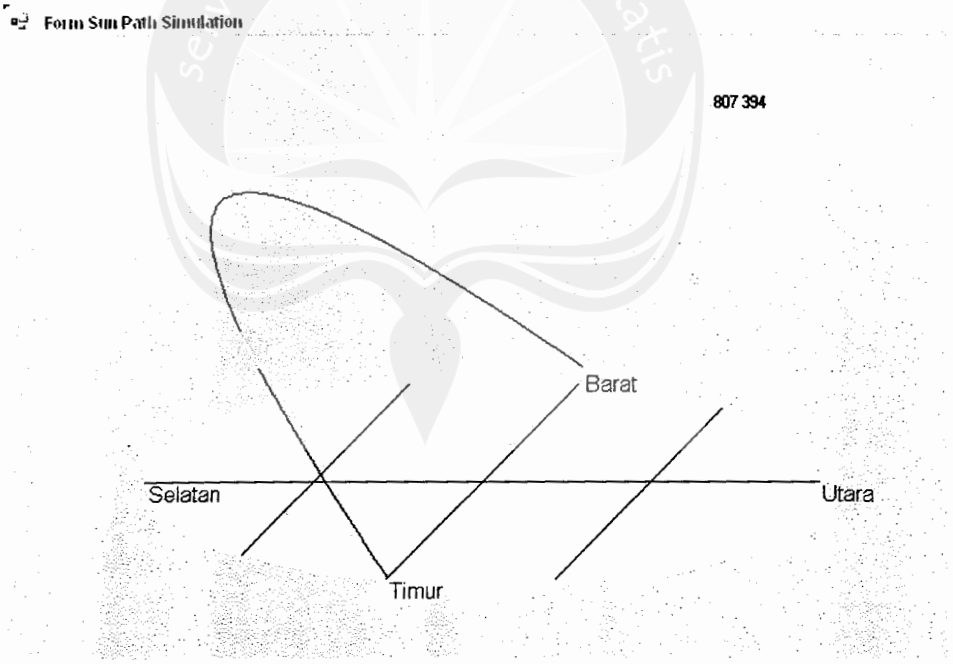
- Rancangan antarmuka ini diimplementasikan pada **Class SunTracker.LDRSensor.MonitoringLDRSensorUI.** Antarmuka ini digunakan pada use case Monitoring LDR Sensor.

Event

- Load Form Monitoring LDR Sensor

```
On_Load MonitoringLDRSensor Form
   Loop :
      GetLDRSensorDatafromSerialPort
      BinaryToDecimal
      Calculate4SensorData
      ShowGraphic
```

4.2 Use Case : Monitoring Motor Stepper



Gambar 4.2 Rancangan Antarmuka Use Case : Monitoring Motor Stepper

Deskripsi

- Rancangan antarmuka ini diimplementasikan pada **Class SunTracker.MotorStepper.MonitoringMotorStepperUI.** Antarmuka ini digunakan pada use case Monitoring Motor Stepper.

Event

- Load Form Monitoring Motor Stepper

```
On_Load MonitoringMotorStepper Form
    Loop :
        GetMotorStepperDatafromSerialPort
        BinaryToDecimal
        CalculateMotorStepperData
        Sql : "Inser into SunPosition value ID =
        "+_id+",TIME = "+_time+",AZIMUTH =
        "+_azimuth+",TILTH = "+_tilth+",VALUE="+_value
        ShowSimluation
```

PDHUPL

PERENCANAAN, DESKRIPSI, DAN HASIL UJI PERANGKAT LUNAK

Pembangunan Aplikasi Otomatisasi Posisi Panel Sel Surya untuk Mendapatkan Energi Cahaya Matahari Optimal (Sun Tracker)

Disusun oleh:

Ign. Hendra Adi Wijaya (03755)

Program Studi Teknik Informatika – Fakultas Teknologi Industri

Universitas Atma Jaya Yogyakarta

DAFTAR PERUBAHAN

Revisi	Deskripsi
A	
B	
C	
D	
E	
F	
G	

INDEX TGL	-	A	B	C	D	E	F	G
Ditulis oleh								
Diperiksa oleh								
Disetujui oleh								

Pendahuluan

1.1 Tujuan

Dokumen PDHUPL ini dibuat untuk menyediakan perencanaan, deskripsi, dan hasil pengujian perangkat lunak SunTracker. Dokumen ini ditujukan untuk pembuat perangkat lunak, dan orang-orang lain yang tertarik untuk mengembangkan perangkat lunak ini lebih lanjut

1.2 Deskripsi Umum Sistem

SunTracker adalah perangkat lunak yang dikembangkan untuk membantu proses monitoring perubahan posisi matahari. Sistem ini dibangun dengan menggunakan bahasa pemrograman C#.NET serta menggunakan SQL Server 2000 sebagai DBMS untuk datadata yang terkait dengan perubahan posisi matahari (*Azimuth* dan *Tilth*).

1.3 Istilah dan Singkatan

Untuk definisi istilah dan singkatan yang digunakan dalam dokumen ini dapat mengacu pada **Apendiks A : Daftar Istilah dan Singkatan**.

1.4 Referensi

Referensi yang digunakan dalam pembuatan dokumen ini adalah :

- Dewi, Findra Kartika Sari. *Perancangan, Deskripisi dan Hasil Uji Perangkat Lunak Fine Generator Of Scheduler*. Universitas Atma Jaya Yogyakarta. 2005.
- Ign. Hendra Adi W. *Spesifikasi Kebutuhan Perangkat Lunak SunTracker*. Universitas Atma Jaya Yogyakarta. 2007.
- Ign. Hendra Adi W. *Deskripsi Perancangan Perangkat Lunak SunTracker*. Universitas Atma Jaya Yogyakarta. 2007.

1.5 Deskripsi Umum Dokumen

Dokumen ini terdiri dari lima bab, yaitu :

1. Bab pertama adalah **Pendahuluan,** yang akan memberikan deskripsi dokumen.
2. Bab kedua adalah **Lingkungan Pengujian Perangkat Lunak**, yang akan menggambarkan lingkungan tempat berjalannya perangkat lunak (perangkat keras dan perangkat lunak), sumber daya manusia, serta prosedur umum pengujian.

3. Bab ketiga adalah **Identifikasi dan Rencana Pengujian**, yang berisi deskripsi umum kelas-kelas dan butir-butir pengujian.
4. Bab keempat adalah **Identifikasi Pengujian**, yang berisi deskripsi rinci kelas-kelas dan butir-butir pengujian.
5. Bab kelima adalah **Hasil Pengujian**, yang berisi langkah-langkah dan hasil pengujian kelas-kelas dan butir-butir pengujian.

2. Lingkungan Pengujian Perangkat Lunak

2..1Perangkat Lunak Pengujian

Perangkat lunak yang digunakan untuk pengujian berupa:

1. Sistem Operasi Windows XP Profesional Edition.
2. SQL Server 2000.
3. Data-data input program seperti data sensor LDR yang telah dibuat sebelumnya.

2..2Perangkat Keras Pengujian

Perangkat keras yang digunakan untuk pengujian berupa:

1. Komputer tempat aplikasi SunTracker berjalan, dengan spesifikasi AMD Athlon 64 2800+, 1 G DDRRAM

2..3Sumber Daya Manusia

Sumber daya manusia yang digunakan untuk pengujian berupa:

1. Pembuat Perangakat Lunak, dengan pengalaman pemrograman berbasis GUI 3 bulan

2..4Prosedur Umum Pengujian

2.4.1 Pengenalan dan Latihan

Pengenalan dan Latihan perangkat lunak SunTracker diharapkan tidak memerlukan waktu lama. SunTracker diharapkan dapat dipelajari langsung dari antamuka bantuan, tanpa melalui pelatihan khusus. Pengguna SunTracker adalah user yang belum/sudah familiar dengan penggunaan program berbasis GUI.

2.4.2 Persiapan Perangkat Keras

Persiapan perangkat keras berupa:

1. Komputer yang terhubung dengan basis data.
2. Keyboard
3. Mouse
4. Rangkaian SunTracker dan juga SunTracker Panel

2.4.3 Persiapan Perangkat Lunak

Persiapan Perangakat Lunak berupa:

1. Instalasi SQL Server 2000
2. Instalasi aplikasi SunTracker

2.4.4 Pelaksanaan

Pelaksanan pengujian akan dilakukan untuk masing-masing use case, basic path dan alternative path. Untuk deskripisi use case dapat mengacu ke Spesifikasi Kebutuhan Perangkat Lunak SunTracker

2.4.5 Pelaporan Hasil

Hasil pengujian akan diserahkan kepada Program Studi Teknik Informatika dan Teknik Industri Fakultas Teknologi Industri Universitas Atma Jaya Yogyakarta.

3. Identifikasi dan Rencana Pengujian

3.1 Identifikasi Pengujian

Tabel 3.1 Identifikasi Pengujian Use Case : MonitoringLDRSensor

Kelas Uji	Pengujian Use Case MonitoringLDRSensor
Butir Uji	Monitoring LDR Sensor
Identifikasi	
SKPL	UC-SunTracker-01
PDHUPL	AU-01-01
Tingkat Pengujian	Pengujian Unit
Jenis Pengujian	Black Box

Jadwal	Juni 2007

Tabel 3.2 Identifikasi Pengujian Use Case : MonitoringMotor Stepper

Kelas Uji	Pengujian Use Case MonitoringMotorStepper
Butir Uji	Monotirng Motor Stepper
Identifikasi	
SKPL	UC-SunTracker-02
PDHUPL	AU-02-01
Tingkat Pengujian	Pengujian Unit
Jenis Pengujian	Black Box
Jadwal	Juni 2007

3.2 Rencanan Pengujian

3.2.1 Urutan Pelaksanaan Pengujian

Urutan pengujian sesuai dengan nomor identifikasi pengujian yang telah ditentukan pada bab 3.1.

3.2.2 Data Pengujian

Data pengujian meliputi data LDR Sensor yang berupa data perubahan resistansi.

4. Identifikasi Pengujian

4.1 Identifikasi Kelas Pengujian Use Case MonitoringLDRSensor

Kelas pengujian ini meliputi pengujian-pengujian yang melibatkan fungsi antarmuka use case MonitoringLDRSensor dengan aktor user sebagai penggunanya.

4.1.1 Identifikasi Butir Pengujian Monitoring LDR Sensor(AU-01-01)

Butir pengujian ini melakukan pengujian terhadap antarmuka Monitoring LDR Sensor, dengan membuka menu ini dan menghasilkan tampilan berupa grafik perubahan resistansi sensor, dimana data yang ditampilkan berupa data grafik berjalan.

4.2. Identifikasi Kelas Pengujian Use Case MonitoringMotorStepper

Kelas pengujian ini meliputi pengujian-pengujian yang melibatkan fungsi antarmuka use case MonitoringMotorStepper dengan aktor user sebagai penggunanya.

4.2.1 Identifikasi Butir Pengujian Monitoring Motor Stepper (AU-02-01)

Butir pengujian ini melakukan pengujian terhadap antarmuka monitoring motor stepper untuk fungsi menampilkan data simulasi. Dimana ketika menu ini terbuka, maka akan menampilkan posisi terakhir dari matahari, dimana posisi perubahan ini disimulasikan pada posisi siang hari.

5. Hasil Pengujian

5.1 Hasil Pengujian Use Case MonitoringLDRSensor

5.1.1 Hasil Pengujian MonitoringLDRSensor (AU-01-01)

Tabel 5.1 Hasil Pengujian MonitoringLDRSensor(AU-01-01)

Identifikasi	AU-01-01			
Deskripsi	Pengujian terhadap antar muka MonitoringLDRSensor			
Prosedur Pengujian	**Masukan**	**Keluaran yang diharapkan**	**Kriteria Evaluasi Hasil**	**Hasil yang didapat**
Buka menu MontoringLDRSensor	-	Menampilkan data yang berupa grafik, dimana data yang ditampilkan merupakan perubahan data sensor	Menampilkan data grafik sesuai dengan perubahan data yang ada pada sensor LDR	Menampilkan data grafik sesuai dengan perubahan data yang ada pada sensor LDR
Kesimpulan	Handal			

5.2 Hasil Pengujian Use Case MonitoringMotorStepper

5.2.1 Hasil Pengujian MonitoringMotorStepper(AU-02-01)

Tabel 5.2 Hasil Pengujian MonitoringMotorStepper (AU-02-01)

Identifikasi	AU-02-01			
Deskripsi	Pengujian terhadap antar muka MonitoringMotorStepper oleh User			
Prosedur Pengujian	**Masukan**	**Keluaran yang diharapkan**	**Kriteria Evaluasi Hasil**	**Hasil yang didapat**
Buka menu MontoringMotorStepper	-	Menampilkan data yang berupa simulasi perubahan posisi matahari, dimana data yang ditampilkan berdasarkan perubahan data sensor	Menampilkan data pergerakan motor stepper yang disimulasikan yang mewakili perubahan posisi matahari	Menampilkan data pergerakan motor stepper yang disimulasikan yang mewakili perubahan posisi matahari
Kesimpulan	Handal			

Apendiks A : Daftar Istilah dan Singkatan

User adalah operator yang berhak mengakses menu menu yang ada pada SunTracker sistem **SunTracker** adalah merupakan suatu sistem yang terdiri atas perangkat lunak dan keras, dimana pada sistem ini terdapat rangkaian elektronik serta panel yang akan mengukuti pergerakan dari perubahan posisi matahari, dimana perhitungan perubahan ini dilakukan pada rangkaian elektroniknya(mikrokontroler), dan disimulasikan pada PC.

Kode Assembly untuk Mikrokontroler AT89C51

```
;Control ADC0809
SC     BIT P2.7
ALE    BIT P2.6
EOC    BIT P2.5
OE     BIT P2.4

;Address Sensor ADC0809 => LDR
ADDR_A BIT P2.2 ; pin 25 indeks ke 0 ; 1  0  1  = 3 ; 1  1  0  = 6
ADDR_B BIT P2.1 ; pin 24 indeks ke 1 ; C  B  A      ; C  B  A
ADDR_C BIT P2.0 ; pin 23 indeks ke 2 ;

;Data Conversion from ADC0809
MYDATA      EQU P0

       ORG   00H
       SJMP  MAIN

       ORG   23H
       SJMP  SERINT

       ORG   030H
MAIN:  MOV TMOD,#20H
       MOV TH1,#0FdH
       MOV SCON,#50H
       SETB TR1
       MOV R0,#60H
       MOV SP,#10H

       ;Setting ADC0809
       CLR  ALE
       CLR  SC
       SETB EOC
       CLR  OE

       ;Clear data for ULN2803 (Relay Driver)
       CLR P1.1
       CLR P1.2
       CLR P1.3
       CLR P1.4

SETB ES
SETB EA

       MOV DPTR,#DATANYA
       MOV R2,#20
KIRIM : CLR A
       MOVC A,@A+DPTR
       MOV SBUF,A
       JNB TI,$
       CLR TI
       INC DPTR
       DJNZ R2,KIRIM

DEAD: SJMP DEAD
```

```
SERINT:
      JB RI,RCV_ch
      CLR TI
      RETI

RCV_ch:     PUSH PSW
      PUSH ACC
      MOV A,SBUF
      CLR RI
      SUBB A,#32
      SJMP GERAKI


EXIT: MOV SBUF,A
      POP ACC
      POP PSW
      RETI

DATANYA :
      DB 'SELAMAT DATANG...',13,10

;A = gerakan maju motor dc I
;B = gerakan mundur motor dc I
;C = gerakan maju motor dc II
;D = gerakan mundur motor dc II
;E = Permintaan data untuk Sensor I  (Utara)
;F = Permintaan data untuk Sensor II (Selatan)
;G = Permintaan data untuk Sensor III(Barat)
;H = Permintaan data untuk Sensor IV (Timur)
;I = Permintaan data untuk nilai dari Solar Cell

GERAKI    : CJNE A,#'A',GERAKII
          JMP DCI_MAJU
          JMP EXIT
GERAKII   : CJNE A,#'B',GERAKIII
            JMP DCI_MUNDUR
            JMP EXIT
GERAKIII : CJNE A,#'C',GERAKIV
          JMP DCII_MAJU
          JMP EXIT
GERAKIV   : CJNE A,#'D',SENSORI
          JMP DCII_MUNDUR
          JMP EXIT
SENSORI   : CJNE A,#'E',SENSORII
          JMP SENSOR_I
          JMP EXIT
SENSORII : CJNE A,#'F',SENSORIII
          JMP SENSOR_II
          JMP EXIT
SENSORIII: CJNE A,#'G',SENSORIV
          JMP SENSOR_III
          JMP EXIT
SENSORIV : CJNE A,#'H',SOLARCELLI
          JMP SENSOR_IV
          JMP EXIT
SOLARCELLI:CJNE A,#'I',SOLARCELLII
```

```
          JMP SOLAR_CELL_I
          JMP EXIT
SOLARCELLII:CJNE A,#'J',SOLARCELLIII
          JMP SOLAR_CELL_II
          JMP EXIT
SOLARCELLIII:CJNE A,#'K',SOLARCELLIV
          JMP SOLAR_CELL_III
          JMP EXIT
SOLARCELLIV:CJNE A,#'L',EXIT
          JMP SOLAR_CELL_IV
          JMP EXIT

DCI_MAJU :
        SETB P1.1
        ACALL DELAY
        CLR  P1.1
        JMP EXIT
DCI_MUNDUR :
        SETB P1.2
        ACALL DELAY
        CLR  P1.2
        JMP EXIT
DCII_MAJU :
        SETB P1.3
        ACALL DELAY
        CLR  P1.3
          JMP EXIT
DCII_MUNDUR :
        SETB P1.4
        ACALL DELAY
        CLR  P1.4
        JMP EXIT

SENSOR_I :;Setting Address to Channel 0
           CLR ADDR_C
         CLR ADDR_B;; sel ch=0
         CLR ADDR_A
         JMP CONVERSION
SENSOR_II :;Setting Address to Channel 1
           CLR  ADDR_C
         CLR  ADDR_B;; sel ch=1
         SETB ADDR_A
         JMP CONVERSION
SENSOR_III:;Setting Address to Channel 2
           CLR  ADDR_C
         SETB ADDR_B;; sel ch=2
         CLR  ADDR_A
         JMP CONVERSION
SENSOR_IV :;Setting Address to Channel 3
           CLR  ADDR_C
         SETB ADDR_B;; sel ch=3
         SETB ADDR_A
         JMP CONVERSION
SOLAR_CELL_I:;Setting Address to Channel 4
           SETB ADDR_C
         CLR  ADDR_B;; sel ch=4
```

```
        CLR  ADDR_A
        JMP CONVERSION
SOLAR_CELL_II:;Setting Address to Channel 5
          SETB ADDR_C
        CLR  ADDR_B;; sel ch=5
        SETB ADDR_A
        JMP CONVERSION
SOLAR_CELL_III:;Setting Address to Channel 6
          SETB ADDR_C
        SETB ADDR_B;; sel ch=6
        CLR  ADDR_A
        JMP CONVERSION
SOLAR_CELL_IV:;Setting Address to Channel 7
          SETB ADDR_C
        SETB ADDR_B;; sel ch=7
        SETB ADDR_A
        JMP CONVERSION

CONVERSION:
        ACALL ADCDELAY
        SETB ALE ;; latch addr
        ACALL ADCDELAY
        SETB SC ;; start
        ACALL ADCDELAY
        CLR ALE
        CLR SC
        ACALL ADCDELAY ;tunggu sampai konversi selesai
        ACALL ADCDELAY ;tunggu sampai konversi selesai
        SETB OE
        ACALL ADCDELAY
        MOV A, MYDATA
        CLR OE
        JMP EXIT

;Setting Delay for Motor
DELAY  : MOV R0,#1H
DELAY1 : MOV R1,#0AEH
DELAY2 : MOV R2,#0
       DJNZ R2,$
       DJNZ R1,DELAY2
       DJNZ R0,DELAY1
       RET

;Setting Delay for ADC0809
ADCDELAY  : MOV R0,#1H
ADCDELAY1 : MOV R1,#0AEH
ADCDELAY2 : MOV R2,#0
          DJNZ R2,$
          DJNZ R1,ADCDELAY2
          DJNZ R0,ADCDELAY1
          RET
END
```

Features

- Compatible with MCS-51™ Products
- 4K Bytes of In-System Reprogrammable Flash Memory
  - Endurance: 1,000 Write/Erase Cycles
- Fully Static Operation: 0 Hz to 24 MHz
- Three-Level Program Memory Lock
- 128 x 8-Bit Internal RAM
- 32 Programmable I/O Lines
- Two 16-Bit Timer/Counters
- Six Interrupt Sources
- Programmable Serial Channel
- Low Power Idle and Power Down Modes

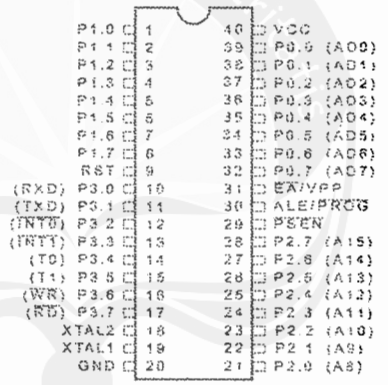
Description

The AT89C51 is a low-power, high-performance CMOS 8-bit microcomputer with 4K bytes of Flash Programmable and Erasable Read Only Memory (PEROM). The device is manufactured using Atmel's high density nonvolatile memory technology and is compatible with the industry standard MCS-51™ instruction set and pinout. The on-chip Flash allows the program memory to be reprogrammed in-system or by a conventional nonvolatile memory programmer. By combining a versatile 8-bit CPU with Flash on a monolithic chip, the Atmel AT89C51 is a powerful microcomputer which provides a highly flexible and cost effective solution to many embedded control applications.

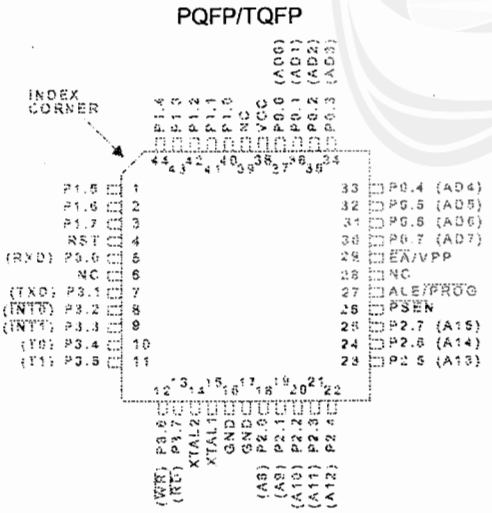
(continued)

Pin Configurations

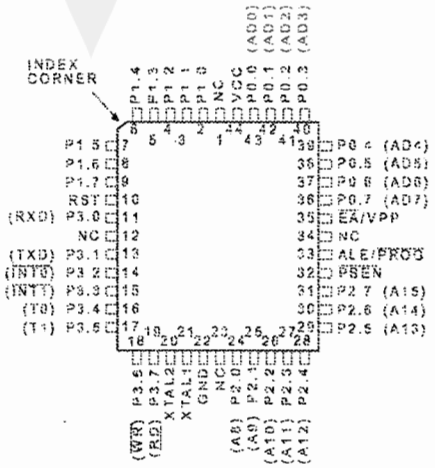
PDIP



PQFP/TQFP



PLCC

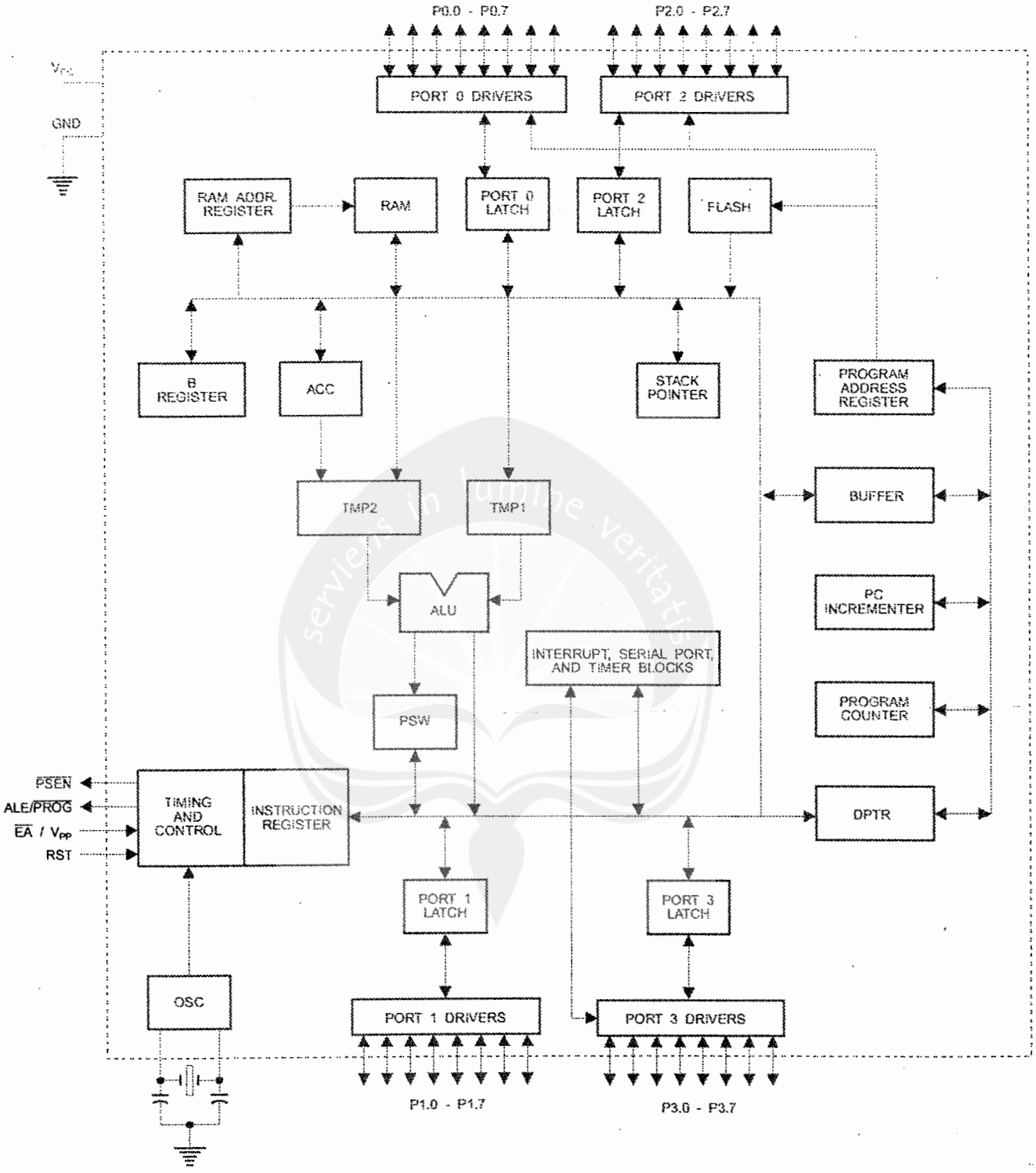


8-Bit Microcontroller with 4K Bytes Flash

AT89C51

0265F-A–12/97

Block Diagram



The AT89C51 provides the following standard features: 4K bytes of Flash, 128 bytes of RAM, 32 I/O lines, two 16-bit timer/counters, a five vector two-level interrupt architecture, a full duplex serial port, on-chip oscillator and clock circuitry. In addition, the AT89C51 is designed with static logic for operation down to zero frequency and supports two software selectable power saving modes. The Idle Mode stops the CPU while allowing the RAM, timer/counters, serial port and interrupt system to continue functioning. The Power Down Mode saves the RAM contents but freezes the oscillator disabling all other chip functions until the next hardware reset.

Pin Description

V_{CC}
Supply voltage.

GND
Ground.

Port 0
Port 0 is an 8-bit open drain bidirectional I/O port. As an output port each pin can sink eight TTL inputs. When 1s are written to port 0 pins, the pins can be used as high-impedance inputs.

Port 0 may also be configured to be the multiplexed low-order address/data bus during accesses to external program and data memory. In this mode P0 has internal pullups.

Port 0 also receives the code bytes during Flash programming, and outputs the code bytes during program verification. External pullups are required during program verification.

Port 1
Port 1 is an 8-bit bidirectional I/O port with internal pullups. The Port 1 output buffers can sink/source four TTL inputs. When 1s are written to Port 1 pins they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 1 pins that are externally being pulled low will source current (I_{IL}) because of the internal pullups.

Port 1 also receives the low-order address bytes during Flash programming and verification.

Port 2
Port 2 is an 8-bit bidirectional I/O port with internal pullups. The Port 2 output buffers can sink/source four TTL inputs. When 1s are written to Port 2 pins they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 2 pins that are externally being pulled low will source current (I_{IL}) because of the internal pullups.

Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that use 16-bit addresses (MOVX @ DPTR). In this application it uses strong internal pullups when emitting 1s. During accesses to external data memory that use 8-bit addresses (MOVX @ RI), Port 2 emits the contents of the P2 Special Function Register.

Port 2 also receives the high-order address bits and some control signals during Flash programming and verification.

Port 3
Port 3 is an 8-bit bidirectional I/O port with internal pullups. The Port 3 output buffers can sink/source four TTL inputs. When 1s are written to Port 3 pins they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 3 pins that are externally being pulled low will source current (I_{IL}) because of the pullups.

Port 3 also serves the functions of various special features of the AT89C51 as listed below:

Port Pin	Alternate Functions
P3.0	RXD (serial input port)
P3.1	TXD (serial output port)
P3.2	$\overline{INT0}$ (external interrupt 0)
P3.3	$\overline{INT1}$ (external interrupt 1)
P3.4	T0 (timer 0 external input)
P3.5	T1 (timer 1 external input)
P3.6	$\overline{WR}$ (external data memory write strobe)
P3.7	$\overline{RD}$ (external data memory read strobe)

Port 3 also receives some control signals for Flash programming and verification.

RST
Reset input. A high on this pin for two machine cycles while the oscillator is running resets the device.

ALE/$\overline{PROG}$
Address Latch Enable output pulse for latching the low byte of the address during accesses to external memory. This pin is also the program pulse input ($\overline{PROG}$) during Flash programming.

In normal operation ALE is emitted at a constant rate of 1/6 the oscillator frequency, and may be used for external timing or clocking purposes. Note, however, that one ALE pulse is skipped during each access to extemal Data Memory.

If desired, ALE operation can be disabled by setting bit 0 of SFR location 8EH. With the bit set, ALE is active only during a MOVX or MOVC instruction. Otherwise, the pin is weakly pulled high. Setting the ALE-disable bit has no effect if the microcontroller is in external execution mode.

$\overline{PSEN}$
Program Store Enable is the read strobe to external program memory.

When the AT89C51 is executing code from external program memory, $\overline{PSEN}$ is activated twice each machine cycle, except that two $\overline{PSEN}$ activations are skipped during each access to external data memory.

$\overline{EA}$/V_{PP}
External Access Enable. $\overline{EA}$ must be strapped to GND in order to enable the device to fetch code from external program memory locations starting at 0000H up to FFFFH. Note, however, that if lock bit 1 is programmed, $\overline{EA}$ will be internally latched on reset.

$\overline{EA}$ should be strapped to V_{CC} for internal program executions.

This pin also receives the 12-volt programming enable voltage (V_{PP}) during Flash programming, for parts that require 12-volt V_{PP}.

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

XTAL2
Output from the inverting oscillator amplifier.

Oscillator Characteristics

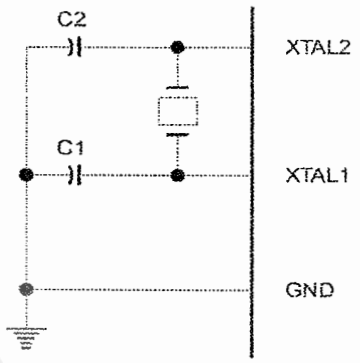
XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier which can be configured for use as an on-chip oscillator, as shown in Figure 1. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven as shown in Figure 2. There are no requirements on the duty cycle of the external clock signal, since the input to the internal clocking circuitry is through a divide-by-two flip-flop, but minimum and maximum voltage high and low time specifications must be observed.

Idle Mode

In idle mode, the CPU puts itself to sleep while all the on-chip peripherals remain active. The mode is invoked by software. The content of the on-chip RAM and all the special functions registers remain unchanged during this mode. The idle mode can be terminated by any enabled interrupt or by a hardware reset.

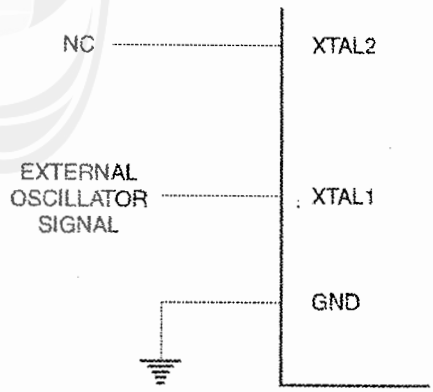
It should be noted that when idle is terminated by a hard ware reset, the device normally resumes program execution, from where it left off, up to two machine cycles before the internal reset algorithm takes control. On-chip hardware inhibits access to internal RAM in this event, but access to the port pins is not inhibited. To eliminate the possibility of an unexpected write to a port pin when Idle is terminated by reset, the instruction following the one that invokes Idle should not be one that writes to a port pin or to external memory.

Figure 1. Oscillator Connections



Note: C1, C2 = 30 pF ± 10 pF for Crystals
= 40 pF ± 10 pF for Ceramic Resonators

Figure 2. External Clock Drive Configuration



Status of External Pins During Idle and Power Down Modes

Mode	Program Memory	ALE	$\overline{PSEN}$	PORT0	PORT1	PORT2	PORT3
Idle	Internal	1	1	Data	Data	Data	Data
Idle	External	1	1	Float	Data	Address	Data
Power Down	Internal	0	0	Data	Data	Data	Data
Power Down	External	0	0	Float	Data	Data	Data

Power Down Mode

In the power down mode the oscillator is stopped, and the instruction that invokes power down is the last instruction executed. The on-chip RAM and Special Function Registers retain their values until the power down mode is terminated. The only exit from power down is a hardware reset. Reset redefines the SFRs but does not change the on-chip RAM. The reset should not be activated before V_{CC} is restored to its normal operating level and must be held active long enough to allow the oscillator to restart and stabilize.

Program Memory Lock Bits

On the chip are three lock bits which can be left unprogrammed (U) or can be programmed (P) to obtain the additional features listed in the table below:

When lock bit 1 is programmed, the logic level at the $\overline{EA}$ pin is sampled and latched during reset. If the device is powered up without a reset, the latch initializes to a random value, and holds that value until reset is activated. It is necessary that the latched value of $\overline{EA}$ be in agreement with the current logic level at that pin in order for the device to function properly.

Lock Bit Protection Modes

Program Lock Bits				Protection Type
	LB1	LB2	LB3	
1	U	U	U	No program lock features.
2	P	U	U	MOVC instructions executed from external program memory are disabled from fetching code bytes from internal memory, $\overline{EA}$ is sampled and latched on reset, and further programming of the Flash is disabled.
3	P	P	U	Same as mode 2, also verify is disabled.
4	P	P	P	Same as mode 3, also external execution is disabled.

Programming the Flash

The AT89C51 is normally shipped with the on-chip Flash memory array in the erased state (that is, contents = FFH) and ready to be programmed. The programming interface accepts either a high-voltage (12-volt) or a low-voltage (V_{CC}) program enable signal. The low voltage programming mode provides a convenient way to program the AT89C51 inside the user's system, while the high-voltage programming mode is compatible with conventional third party Flash or EPROM programmers.

The AT89C51 is shipped with either the high-voltage or low-voltage programming mode enabled. The respective top-side marking and device signature codes are listed in the following table.

	V_{PP} = 12V	V_{PP} = 5V
Top-Side Mark	AT89C51 xxxx yyww	AT89C51 xxxx-5 yyww
Signature	(030H)=1EH (031H)=51H (032H)=FFH	(030H)=1EH (031H)=51H (032H)=05H

The AT89C51 code memory array is programmed byte-by-byte in either programming mode. *To program any non-blank byte in the on-chip Flash Memory, the entire memory must be erased using the Chip Erase Mode.*

Programming Algorithm: Before programming the AT89C51, the address, data and control signals should be set up according to the Flash programming mode table and Figures 3 and 4. To program the AT89C51, take the following steps.

1. Input the desired memory location on the address lines.
2. Input the appropriate data byte on the data lines.
3. Activate the correct combination of control signals.
4. Raise $\overline{EA}/V_{PP}$ to 12V for the high-voltage programming mode.
5. Pulse ALE/$\overline{PROG}$ once to program a byte in the Flash array or the lock bits. The byte-write cycle is self-timed and typically takes no more than 1.5 ms. Repeat steps 1 through 5, changing the address and data for the entire array or until the end of the object file is reached.

$\overline{Data}$ Polling: The AT89C51 features $\overline{Data}$ Polling to indicate the end of a write cycle. During a write cycle, an attempted read of the last byte written will result in the complement of the written datum on PO.7. Once the write cycle has been completed, true data are valid on all outputs, and the next cycle may begin. $\overline{Data}$ Polling may begin any time after a write cycle has been initiated.

Ready/$\overline{Busy}$: The progress of byte programming can also be monitored by the RDY/$\overline{BSY}$ output signal. P3.4 is pulled low after ALE goes high during programming to indicate BUSY. P3.4 is pulled high again when programming is done to indicate READY.

Program Verify: If lock bits LB1 and LB2 have not been programmed, the programmed code data can be read back via the address and data lines for verification. The lock bits cannot be verified directly. Verification of the lock bits is achieved by observing that their features are enabled.

Chip Erase: The entire Flash array is erased electrically by using the proper combination of control signals and by holding ALE/$\overline{PROG}$ low for 10 ms. The code array is written with all "1"s. The chip erase operation must be executed before the code memory can be re-programmed.

Reading the Signature Bytes: The signature bytes are read by the same procedure as a normal verification of locations 030H,

031H, and 032H, except that P3.6 and P3.7 must be pulled to a logic low. The values returned are as follows.

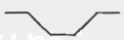
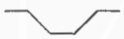
(030H) = 1EH indicates manufactured by Atmel
(031H) = 51H indicates 89C51
(032H) = FFH indicates 12V programming
(032H) = 05H indicates 5V programming

Programming Interface

Every code byte in the Flash array can be written and the entire array can be erased by using the appropriate combination of control signals. The write operation cycle is self-timed and once initiated, will automatically time itself to completion.

All major programming vendors offer worldwide support for the Atmel microcontroller series. Please contact your local programming vendor for the appropriate software revision.

Flash Programming Modes

Mode		RST	$\overline{PSEN}$	ALE/$\overline{PROG}$	$\overline{EA}$/V_{PP}	P2.6	P2.7	P3.6	P3.7
Write Code Data		H	L		H/12V	L	H	H	H
Read Code Data		H	L	H	H	L	L	H	H
Write Lock	Bit - 1	H	L		H/12V	H	H	H	H
	Bit - 2	H	L		H/12V	H	H	L	L
	Bit - 3	H	L		H/12V	H	L	H	L
Chip Erase		H	L	 (1)	H/12V	H	L	L	L
Read Signature Byte		H	L	H	H	L	L	L	L

Note: 1. Chip Erase requires a 10-ms PROG pulse.

Figure 3. Programming the Flash

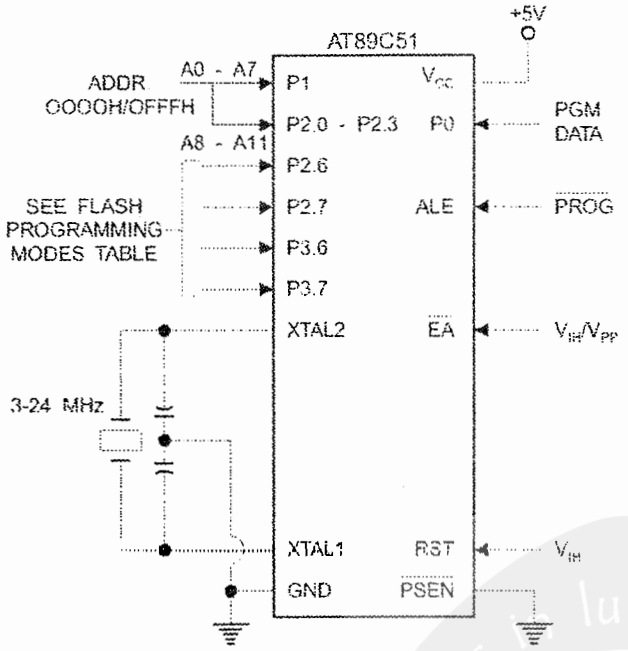
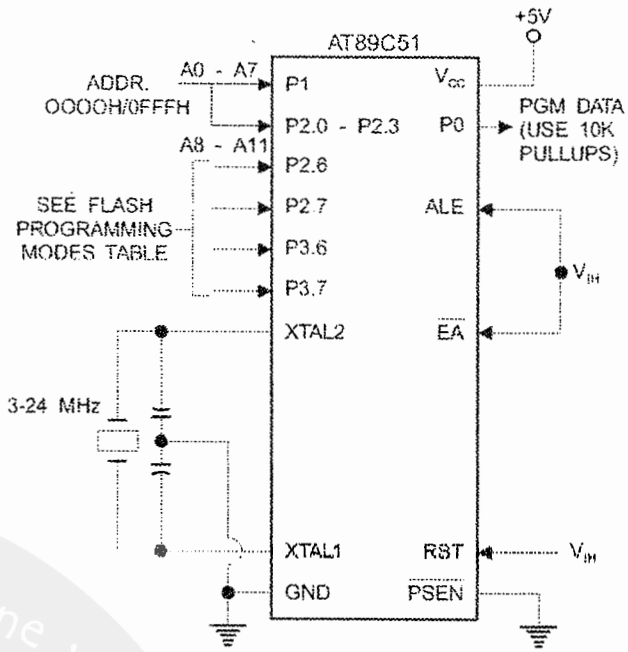


Figure 4. Verifying the Flash



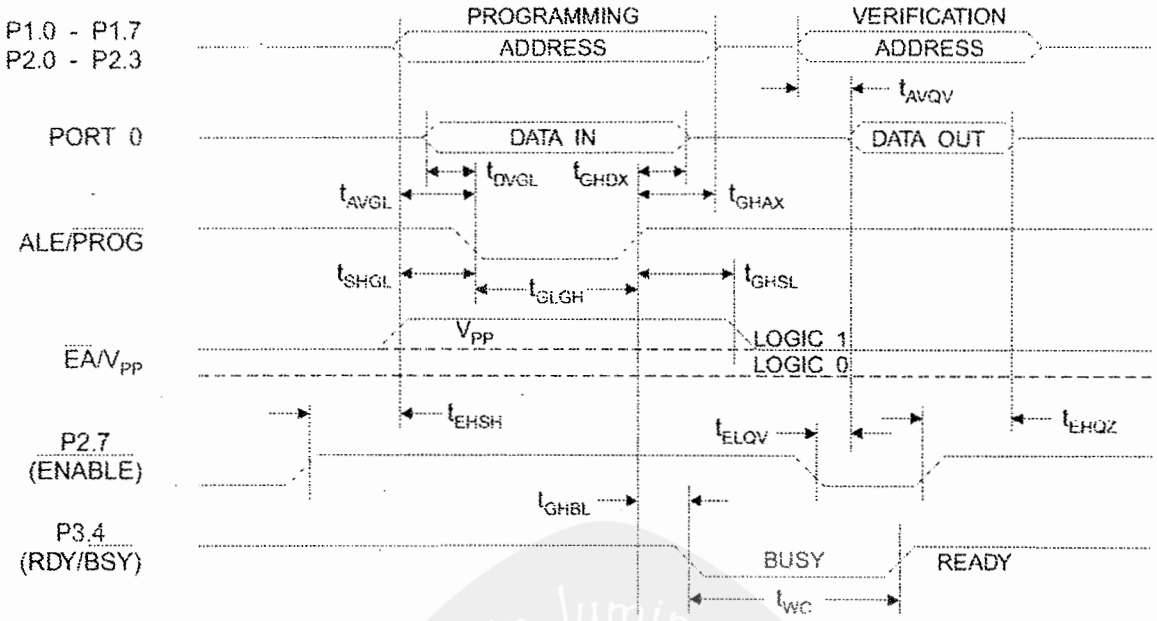
Flash Programming and Verification Characteristics

T_A = 0°C to 70°C, V_{CC} = 5.0 ± 10%

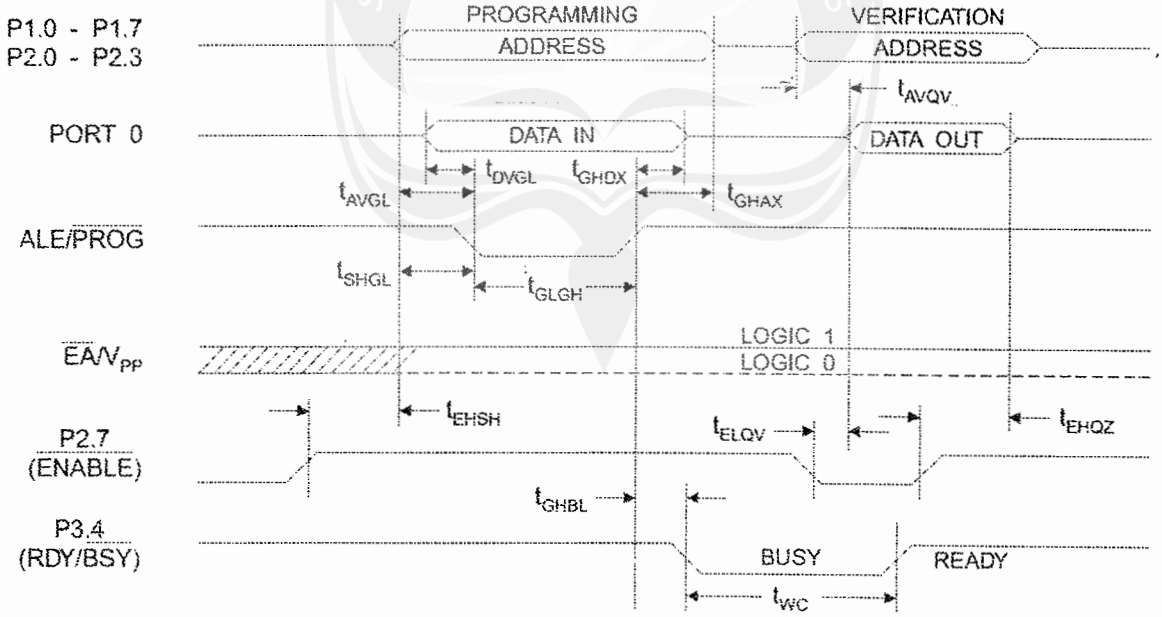
Symbol	Parameter	Min	Max	Units
V_{PP}(1)	Programming Enable Voltage	11.5	12.5	V
I_{PP}(1)	Programming Enable Current		1.0	mA
$1/t_{CLCL}$	Oscillator Frequency	3	24	MHz
t_{AVGL}	Address Setup to $\overline{PROG}$ Low	$48t_{CLCL}$		
t_{GHAX}	Address Hold After $\overline{PROG}$	$48t_{CLCL}$		
t_{DVGL}	Data Setup to $\overline{PROG}$ Low	$48t_{CLCL}$		
t_{GHDX}	Data Hold After $\overline{PROG}$	$48t_{CLCL}$		
t_{EHSH}	P2.7 ($\overline{ENABLE}$) High to V_{PP}	$48t_{CLCL}$		
t_{SHGL}	V_{PP} Setup to $\overline{PROG}$ Low	10		µs
t_{GHSL}(1)	V_{PP} Hold After $\overline{PROG}$	10		µs
t_{GLGH}	$\overline{PROG}$ Width	1	110	µs
t_{AVQV}	Address to Data Valid		$48t_{CLCL}$	
t_{ELQV}	$\overline{ENABLE}$ Low to Data Valid		$48t_{CLCL}$	
t_{EHQZ}	Data Float After $\overline{ENABLE}$	0	$48t_{CLCL}$	
t_{GHBL}	$\overline{PROG}$ High to $\overline{BUSY}$ Low		1.0	µs
t_{WC}	Byte Write Cycle Time		2.0	ms

Note: 1. Only used in 12-volt programming mode.

Flash Programming and Verification Waveforms - High Voltage Mode (V_{PP} = 12V)



Flash Programming and Verification Waveforms - Low Voltage Mode (V_{PP} = 5V)



Absolute Maximum Ratings*

Operating Temperature	-55°C to +125°C
Storage Temperature	-65°C to +150°C
Voltage on Any Pin with Respect to Ground	-1.0V to +7.0V
Maximum Operating Voltage	6.6V
DC Output Current	15.0 mA

*NOTICE: Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

DC Characteristics

T_A = -40°C to 85°C, V_{CC} = 5.0V ± 20% (unless otherwise noted)

Symbol	Parameter	Condition	Min	Max	Units
V_{IL}	Input Low Voltage	(Except $\overline{EA}$)	-0.5	0.2 V_{CC} - 0.1	V
V_{IL1}	Input Low Voltage ($\overline{EA}$)		-0.5	0.2 V_{CC} - 0.3	V
V_{IH}	Input High Voltage	(Except XTAL1, RST)	0.2 V_{CC} + 0.9	V_{CC} + 0.5	V
V_{IH1}	Input High Voltage	(XTAL1, RST)	0.7 V_{CC}	V_{CC} + 0.5	V
V_{OL}	Output Low Voltage(1) (Ports 1,2,3)	I_{OL} = 1.6 mA		0.45	V
V_{OL1}	Output Low Voltage(1) (Port 0, ALE, $\overline{PSEN}$)	I_{OL} = 3.2 mA		0.45	V
V_{OH}	Output High Voltage (Ports 1,2,3, ALE, $\overline{PSEN}$)	I_{OH} = -60 µA, V_{CC} = 5V ± 10%	2.4		V
		I_{OH} = -25 µA	0.75 V_{CC}		V
		I_{OH} = -10 µA	0.9 V_{CC}		V
V_{OH1}	Output High Voltage (Port 0 in External Bus Mode)	I_{OH} = -800 µA, V_{CC} = 5V ± 10%	2.4		V
		I_{OH} = -300 µA	0.75 V_{CC}		V
		I_{OH} = -80 µA	0.9 V_{CC}		V
I_{IL}	Logical 0 Input Current (Ports 1,2,3)	V_{IN} = 0.45V		-50	µA
I_{TL}	Logical 1 to 0 Transition Current (Ports 1,2,3)	V_{IN} = 2V, VCC = 5V ± 10%		-650	µA
I_{LI}	Input Leakage Current (Port 0, $\overline{EA}$)	0.45 < V_{IN} < V_{CC}		±10	µA
RRST	Reset Pulldown Resistor		50	300	KΩ
C_{IO}	Pin Capacitance	Test Freq. = 1 MHz, T_A = 25°C		10	pF
I_{CC}	Power Supply Current	Active Mode, 12 MHz		20	mA
		Idle Mode, 12 MHz		5	mA
	Power Down Mode(2)	V_{CC} = 6V		100	µA
		V_{CC} = 3V		40	µA

Notes: 1. Under steady state (non-transient) conditions, I_{OL} must be externally limited as follows:
Maximum I_{OL} per port pin: 10 mA
Maximum I_{OL} per 8-bit port: Port 0: 26 mA
Ports 1, 2, 3: 15 mA
Maximum total I_{OL} for all output pins: 71 mA
If I_{OL} exceeds the test condition, V_{OL} may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test conditions.

2. Minimum V_{CC} for Power Down is 2V.

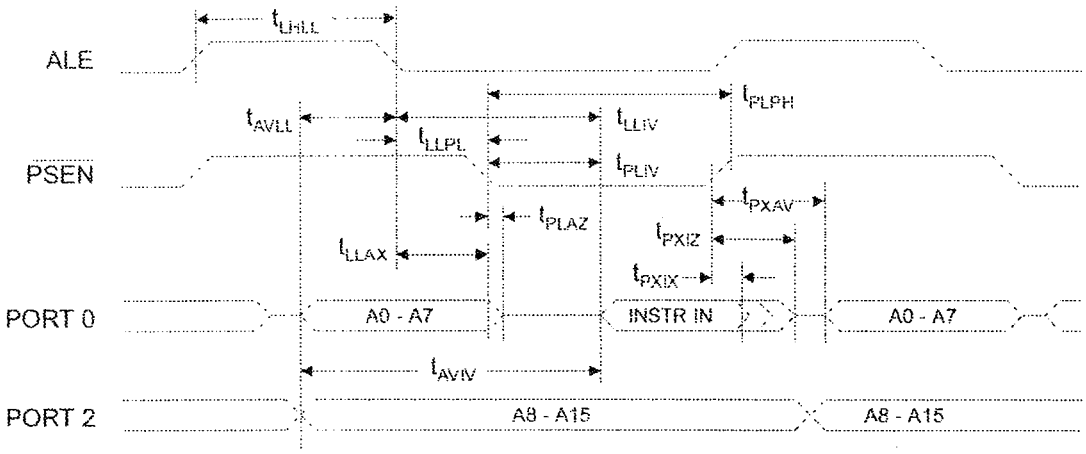
AC Characteristics

(Under Operating Conditions; Load Capacitance for Port 0, ALE/$\overline{PROG}$, and $\overline{PSEN}$ = 100 pF; Load Capacitance for all other outputs = 80 pF)

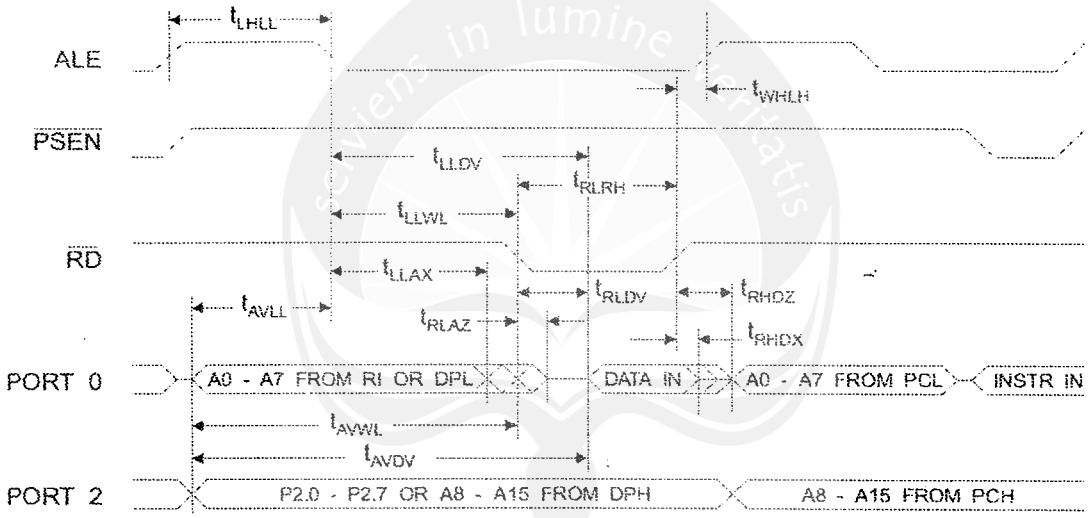
External Program and Data Memory Characteristics

Symbol	Parameter	12 MHz Oscillator		16 to 24 MHz Oscillator		Units
		Min	Max	Min	Max	
$1/t_{CLCL}$	Oscillator Frequency			0	24	MHz
t_{LHLL}	ALE Pulse Width	127		$2t_{CLCL}$-40		ns
t_{AVLL}	Address Valid to ALE Low	43		t_{CLCL}-13		ns
t_{LLAX}	Address Hold After ALE Low	48		t_{CLCL}-20		ns
t_{LLIV}	ALE Low to Valid Instruction In		233		$4t_{CLCL}$-65	ns
t_{LLPL}	ALE Low to $\overline{PSEN}$ Low	43		t_{CLCL}-13		ns
t_{PLPH}	$\overline{PSEN}$ Pulse Width	205		$3t_{CLCL}$-20		ns
t_{PLIV}	$\overline{PSEN}$ Low to Valid Instruction In		145		$3t_{CLCL}$-45	ns
t_{PXIX}	Input Instruction Hold After $\overline{PSEN}$	0		0		ns
t_{PXIZ}	Input Instruction Float After $\overline{PSEN}$		59		t_{CLCL}-10	ns
t_{PXAV}	$\overline{PSEN}$ to Address Valid	75		t_{CLCL}-8		ns
t_{AVIV}	Address to Valid Instruction In		312		$5t_{CLCL}$-55	ns
t_{PLAZ}	$\overline{PSEN}$ Low to Address Float		10		10	ns
t_{RLRH}	$\overline{RD}$ Pulse Width	400		$6t_{CLCL}$-100		ns
t_{WLWH}	$\overline{WR}$ Pulse Width	400		$6t_{CLCL}$-100		ns
t_{RLDV}	$\overline{RD}$ Low to Valid Data In		252		$5t_{CLCL}$-90	ns
t_{RHDX}	Data Hold After $\overline{RD}$	0		0		ns
t_{RHDZ}	Data Float After $\overline{RD}$		97		$2t_{CLCL}$-28	ns
t_{LLDV}	ALE Low to Valid Data In		517		$8t_{CLCL}$-150	ns
t_{AVDV}	Address to Valid Data In		585		$9t_{CLCL}$-165	ns
t_{LLWL}	ALE Low to $\overline{RD}$ or $\overline{WR}$ Low	200	300	$3t_{CLCL}$-50	$3t_{CLCL}$+50	ns
t_{AVWL}	Address to $\overline{RD}$ or $\overline{WR}$ Low	203		$4t_{CLCL}$-75		ns
t_{QVWX}	Data Valid to $\overline{WR}$ Transition	23		t_{CLCL}-20		ns
t_{QVWH}	Data Valid to $\overline{WR}$ High	433		$7t_{CLCL}$-120		ns
t_{WHQX}	Data Hold After $\overline{WR}$	33		t_{CLCL}-20		ns
t_{RLAZ}	$\overline{RD}$ Low to Address Float		0		0	ns
t_{WHLH}	$\overline{RD}$ or $\overline{WR}$ High to ALE High	43	123	t_{CLCL}-20	t_{CLCL}+25	ns

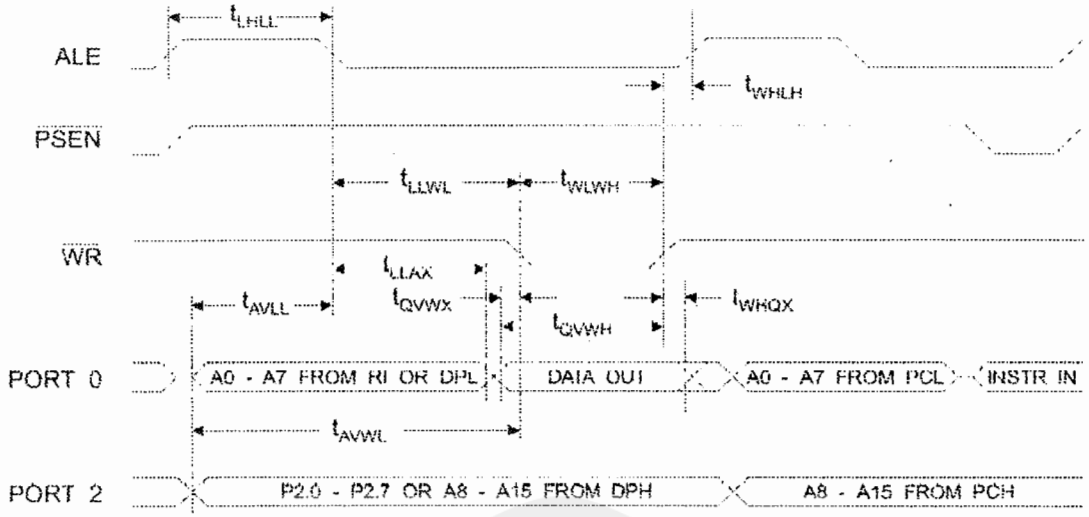
External Program Memory Read Cycle



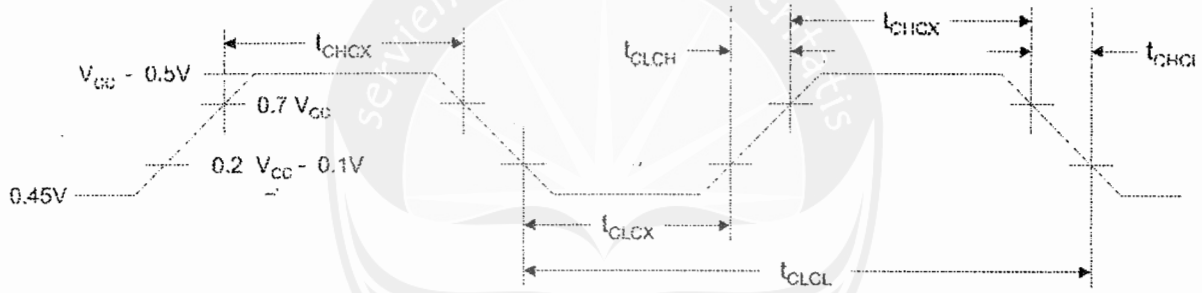
External Data Memory Read Cycle



External Data Memory Write Cycle



External Clock Drive Waveforms



External Clock Drive

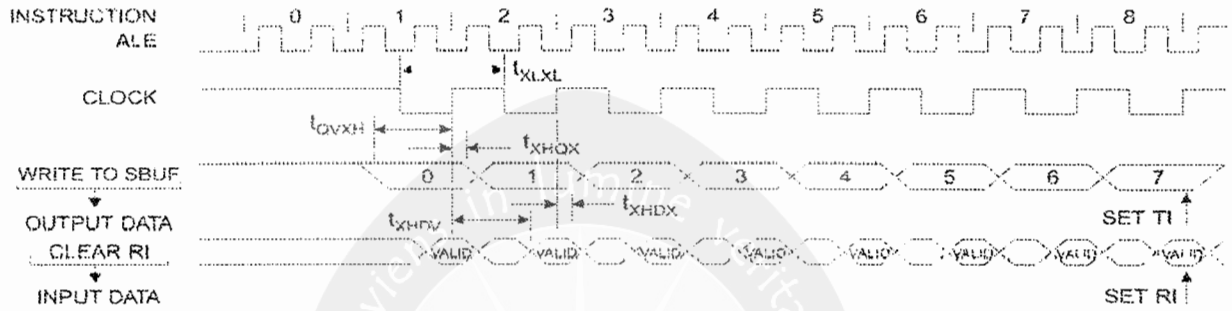
Symbol	Parameter	Min	Max	Units
$1/t_{CLCL}$	Oscillator Frequency	0	24	MHz
t_{CLCL}	Clock Period	41.6		ns
t_{CHCX}	High Time	15		ns
t_{CLCX}	Low Time	15		ns
t_{CLCH}	Rise Time		20	ns
t_{CHCL}	Fall Time		20	ns

Serial Port Timing: Shift Register Mode Test Conditions

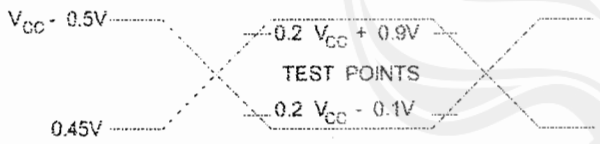
(V_{CC} = 5.0 V ± 20%; Load Capacitance = 80 pF)

Symbol	Parameter	12 MHz Osc		Variable Oscillator		Units
		Min	Max	Min	Max	
t_{XLXL}	Serial Port Clock Cycle Time	1.0		$12t_{CLCL}$		µs
t_{QVXH}	Output Data Setup to Clock Rising Edge	700		$10t_{CLCL}$-133		ns
t_{XHQX}	Output Data Hold After Clock Rising Edge	50		$2t_{CLCL}$-117		ns
t_{XHDX}	Input Data Hold After Clock Rising Edge	0		0		ns
t_{XHDV}	Clock Rising Edge to Input Data Valid		700		$10t_{CLCL}$-133	ns

Shift Register Mode Timing Waveforms

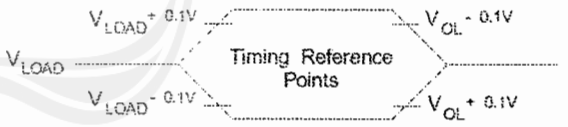


AC Testing Input/Output Waveforms[1]



Note: 1. AC Inputs during testing are driven at V_{CC} - 0.5V for a logic 1 and 0.45V for a logic 0. Timing measurements are made at V_{IH} min. for a logic 1 and V_{IL} max. for a logic 0.

Float Waveforms[1]



Note: 1. For timing purposes, a port pin is no longer floating when a 100 mV change from load voltage occurs. A port pin begins to float when 100 mV change from the loaded V_{OH}/V_{OL} level occurs.

Ordering Information

Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
12	5V ± 20%	AT89C51-12AC	44A	Commercial
		AT89C51-12JC	44J	(0°C to 70°C)
		AT89C51-12PC	40P6	
		AT89C51-12QC	44Q	
		AT89C51-12AI	44A	Industrial
		AT89C51-12JI	44J	(-40°C to 85°C)
		AT89C51-12PI	40P6	
		AT89C51-12QI	44Q	
		AT89C51-12AA	44A	Automotive
		AT89C51-12JA	44J	(-40°C to 105°C)
		AT89C51-12PA	40P6	
		AT89C51-12QA	44Q	
16	5V ± 20%	AT89C51-16AC	44A	Commercial
		AT89C51-16JC	44J	(0°C to 70°C)
		AT89C51-16PC	40P6	
		AT89C51-16QC	44Q	
		AT89C51-16AI	44A	Industrial
		AT89C51-16JI	44J	(-40°C to 85°C)
		AT89C51-16PI	40P6	
		AT89C51-16QI	44Q	
		AT89C51-16AA	44A	Automotive
		AT89C51-16JA	44J	(-40°C to 105°C)
		AT89C51-16PA	40P6	
		AT89C51-16QA	44Q	
20	5V ± 20%	AT89C51-20AC	44A	Commercial
		AT89C51-20JC	44J	(0°C to 70°C)
		AT89C51-20PC	40P6	
		AT89C51-20QC	44Q	
		AT89C51-20AI	44A	Industrial
		AT89C51-20JI	44J	(-40°C to 85°C)
		AT89C51-20PI	40P6	
		AT89C51-20QI	44Q	

Ordering Information

Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
24	5V ± 20%	AT89C51-24AC	44A	Commercial
		AT89C51-24JC	44J	(0°C to 70°C)
		AT89C51-24PC	44P6	
		AT89C51-24QC	44Q	
		AT89C51-24AI	44A	Industrial
		AT89C51-24JI	44J	(-40°C to 85°C)
		AT89C51-24PI	44P6	
		AT89C51-24QI	44Q	

Package Type	
44A	44 Lead, Thin Plastic Gull Wing Quad Flatpack (TQFP)
44J	44 Lead, Plastic J-Leaded Chip Carrier (PLCC)
40P6	40 Lead, 0.600" Wide, Plastic Dual Inline Package (PDIP)
44Q	44 Lead, Plastic Gull Wing Quad Flatpack (PQFP)

PAMUKKALE ÜNİVERSİTESİ MÜHENDİSLİK FAKÜLTESİ YIL : 2006
PAMUKKALE UNIVERSITY ENGINEERING COLLEGE CİLT : 12
MÜHENDİSLİK BİLİMLERİ DERGİSİ SAYI : 2
JOURNAL OF ENGINEERING SCIENCES SAYFA : 243-248

MICROCONTROLLER BASED SOLAR-TRACKING SYSTEM AND ITS IMPLEMENTATION

Okan BİNGÖL*, Ahmet ALTINTAŞ, Yusuf ÖNER*****
*Süleyman Demirel Univ., Faculty of Tec. Education, Department of Elec. and Computer Education, Isparta
**Dumlupınar Univ., Faculty of Tec. Education, Department of Elec. Education, Simav/Kütahya
***Pamukkale Univ., Faculty of Eng., Dep., of Elec. & Elec. Engineering, Kınıklı/Denizli

Geliş Tarihi : 23.06.2005

ABSTRACT

In this paper, a new micro-controller based solar-tracking system is proposed, implemented and tested. The scheme presented here can be operated as independent of the geographical location of the site of setting up. The system checks the position of the sun and controls the movement of a solar panel so that radiation of the sun comes normally to the surface of the solar panel. The developed-tracking system tracks the sun both in the azimuth as well as in the elevation plane. PC based system monitoring facility is also included in the design.

Key Words : Solar energy, solar-tracking system, Solar collector

MIKRODENETLEYICI TABANLI GÜNEŞ TAKIP SISTEMI VE UYGULAMASI

ÖZET

Bu çalışmada, mikrodenetleyici tabanlı bir güneş takip sistemi önerilip gerçeklenmiş ve test edilmiştir. Sunulan tasarım, sistemin kurulmuş olduğu coğrafik bölgeden bağımsız olarak çalışabilmektedir. Sistem, güneş ışığının güneş panelleri yüzeyine dik gelmesini sağlamak için güneşin konumunu test edip güneş panellerinin hareketini kontrol eder. Geliştirilen güneş takip sistemi güneşi, yükselti ve açıklık düzlemlerinin her ikisinde de izleyebilmektedir. Tasarıma, bilgisayar tabanlı sistem gözetleme birimi de ilave edilmiştir.

Anahtar Kelimeler : Güneş enerjisi, Güneş takip sistemi, Güneş kollektörü

1. INTRODUCTION

Currently, many alternative energy sources appear to be technically feasible. One of them is solar energy (Kreider and Kreith, 1981). The panels are the fundamental solar-energy conversion component. Conventional solar panels, fixed with a certain angle, limits their area of exposure from the sun during the course of the day. Therefore, the average solar energy is not always maximized. Solar-tracking systems are essential for many applications such as thermal energy storage systems and solar energy based power generation systems in order to improve system performance (Saxena,. and Dutta,. 1990; Koyuncu and Balasubramanian, 1991; Harakawa and Tujiinoto, 2001). The change in sun's position is monitored, and the system always keeps that the plane of the panel is normal to the direction of the sun. By doing so, maximum irradiation and thermal energy would be taken from the sun.

The elevation angle of the sun remains almost invariant in a month and varies little (latitude ± 10°) in a year. Therefore, a single axis position control scheme may be sufficient for the collection of solar energy in some applications (Konar and Mandal, 1991. Yeong-Chau, et al., 2001. Wilamowski and

Xiangli, 2002). Efficient collection of maximum solar irradiation on a flat panel requires adjustments of two parameters of the energy collecting surface namely the angle of azimuth, ψ and the angle of tilt, α, of the surface to be illuminated in Figure 1 (Davies, 1993; Macagnan, et al., 1994).

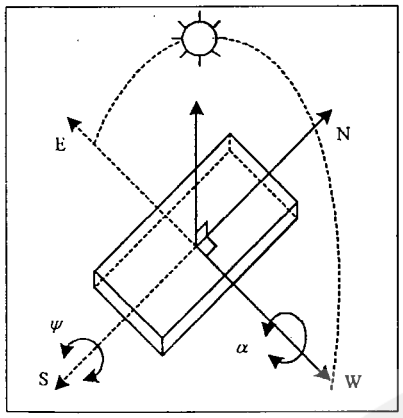


Figure 1. Two axis position control of the solar panel

This paper deals with controlling the solar panel at two axis (or two angles) by using LDRs as sensors, stepper motors as actuators (SM1, SM2) and micro-controller as a controller. In order to keep the design as simple and cheap we have chosen PIC16C71 as a micro-controller-unit (MCU). In addition to this, to observe position of the solar panel, PC based system monitoring facility is included in the design. Block diagram of the proposed solar-tracking system is given in Figure 2.

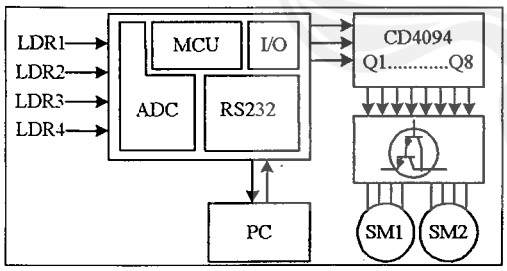


Figure 2. Block diagram of the proposed solar-tracking system

2. DESIGN OF THE SOLAR-TRACKING SYSTEM

The tracker control system contains a control board, a control program, a power supply board, one motor-interface board and a set of sensors. The main idea of design of the solar-tracking system is to sense the sun light by using four light dependent resistors (LDRs). Each LDR is fixed inside the hallow cylindrical tubes. A pair of them, controlling the angle of azimuth, are positioned East-West direction and the two of them, controlling the angle of tilt, are positioned South-North direction. The LDR assembly is fixed onto the flat-solar panel, shown in Figure 3. The tubes are making a degree of 45° with the plane of panel; so, the angle between the tubes is 90°. The differential signals of each pair of LDRs representing the angular error of the solar panel are employed to re-position the panel in such a way that the angular errors are minimized.

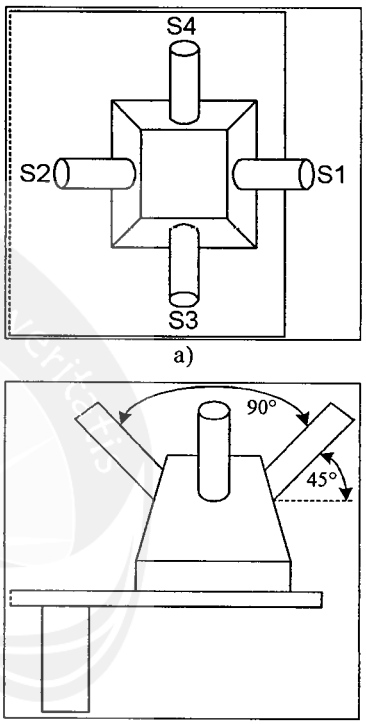


Figure 3. The LDRs assembly, a) Top view, b) Front view

A micro-controller system with PIC16C71 used as the controller of the position control scheme offers up to 10MHz clock frequency, four ADC channels with 8-bit, 1Kx14 EPROM memory and thirteen I/O ports (Anon., 1998a). Since the apparent speed of the sun is very slow, the panel will also move very slowly. Therefore, a crystal with a frequency of 4MHz is used as a clock signal generator for MCU. The signals, taken from voltage divider consisting of resistors and LDRs (S1, S2, S3, S4), are applied to I/O port lines of MCU (RA0, RA1, RA2, RA3) respectively. These analog signals are converted to digital signals and compared with each others (S1-S2, S3-S4).

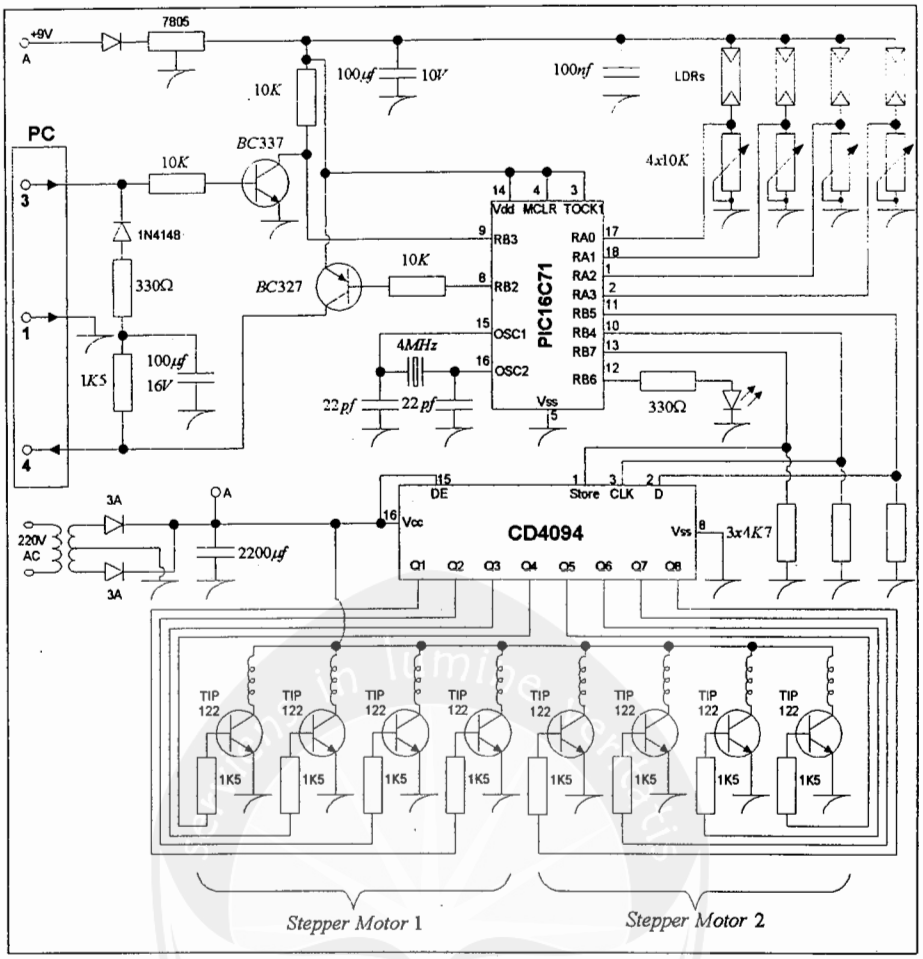


Figure 4. Schematic representation of the solar-tracking system

If the difference between S1 and S2 (or S3 and S4), error signal, is bigger than a certain value (tolerance), MCU generates driving signals for stepper motors. If the error signals are smaller than or equal to the value of tolerance, MCU generates no signal; which means that the solar panel is facing the sun and the light intensities falling on the four LDRs are equal or slightly different. Schematic representation of the solar-tracking system is given in Figure 4.

In order to drive two-stepper motors, an 8-bit shift register CD4094 is used (Anon., 1998b). Data is shifted serially through to shift register on positive transition of the clock signals generated by MCU. Four-output signals (Q1, Q2, Q3, Q4) drive the first-stepper motor and the rest of the signal (Q5, Q6, Q7, Q8) drive the second-stepper motor. In this system, eight darlington transistors (TIP122) are used to drive the SMs. The control commands generated by MCU for driving of SMs and positioning of the solar panel are given in Table 1. As can be seen from the Table 1, when SM2 is driven, four-driving signals for SM1 is set (1111XXXX), and visa-versa.

The software is developed in an assembly programming language. It is compatible for PIC16C71 and is a machine level language. The software consists of a main module and a few subroutines. Simplified flowchart of the assembly program is given in Figure 5, where *e* denotes error (tolerance).

Table 1. The control Commands for Stepper Motors

STEPPER MOTOR 1		STEPPER MOTOR 2	
Q1-Q2-Q3-Q4-Q5-Q6-Q7-Q8			
Forward	Reverse	Forward	Reverse
0001-1111	1000-1111	1111-0001	1111-1000
0010-1111	0100-1111	1111-0010	1111-0100
0100-1111	0010-1111	1111-0100	1111-0010
1000-1111	0001-1111	1111-1000	1111-0001

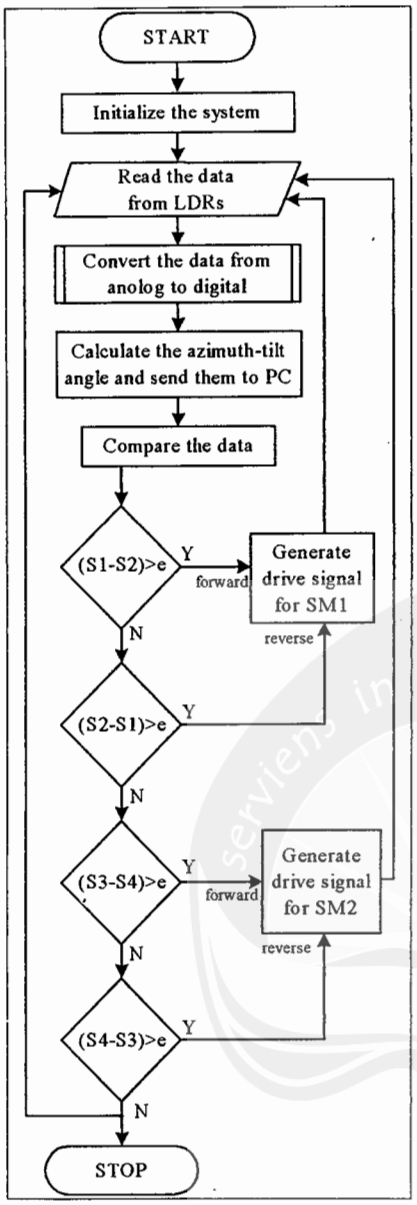


Figure 5. Simplified flowchart of the assembly program

The stepper motors requires a +9V supply and four control pulses at its terminals. One step angle of the motor is equal to 1.8°. If the difference between S1 and S2 is grater than the error, the first stepper motor should rotate forward; if the difference between S2 and S1 is grater than the error, it should rotate reverse direction. Same procedure is applied for the second stepper motor. In order to prevent the flat panel hitting to body, the panel is limited as vertically with a degree of 50° (Figure 6). Thus, the azimuthal drives can rotate on a total of degree of 260° during day times.

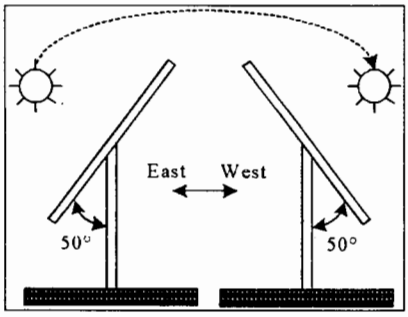


Figure 6. Limits for movement of the solar panel

In general applications, the panel should be rotated towards East direction in order to make it ready for operation on the next day. The proposed system sends no signals during night times by sensing low or none sunshine intensities and stays as pointing at west direction after the sun set. The micro-controller unit is also skipped to sleep mode and consumes low energy. During the sun rise, the LDR senses the sun-light automatically and the panel is moved towards East direction in a short time; so, there is no need any extra circuitry and software to do this. A photograph of the proposed solar-tracker-system body used in experimental studies is given in Figure 7.

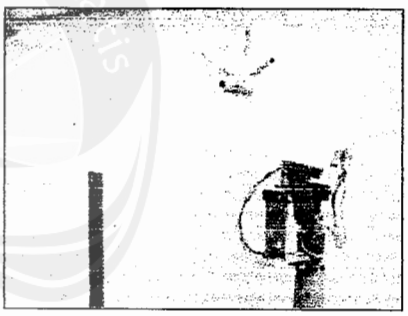


Figure 7. A photograph of the proposed solar-tracker-system body

3. EXPERIMENTAL RESULTS

The solar tracking system presented here is tested experimentally. The experimental study is realized at Gazi University in Ankara, Turkey. Two solar collector panels, one of which is stationary and the other is rotary, are employed in the test. The stationary panel is tilted at a fixed elevation angle and is oriented in some azimuth angle. The LDRs assembly is mounted on the rotary one. Temperature of the panels versus time is measured with a minute-interval of 11.75' in the period 7:48 to 17 : 24. So, a total of measurement time is 9:36'. In this interval, a

total of 50 data were captured from the stationary panel and the rotary panel with approximately an angle interval of 5.2°. The measurement results are given in Figure 8. The differences between the temperature of rotary panel and that of stationary one are also given as graphically in Figure 8.

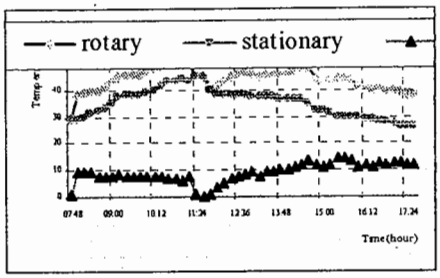


Figure 8. The measurement results

According to the results: while the sun rises in the morning, the temperature of each panel also increases linearly and the difference is almost constant until about 11 o'clock because of being illuminated from the sun with different azimuth and tilt angle (ψ, α). After this point, the temperature of panels decreases until about 11:30 due to weather condition. The temperature, then, increases to the normal condition again, and then keeps decreasing slightly afternoon. The temperature of average value of the stationary panel between the hours of 7 : 48 and 17 : 24 is measured as 34.9 °C and that of the rotary panel is 43.88 °C. It is very obvious that there is approximately a distinction of 9 °C between rotary and stationary panel. This result shows and verifies that the rotary panel containing solar tracking system takes more light density than the stationary panel.

System monitoring software in PC is developed in Pascal® program. Thus, LDRs' information depending on light intensities, the angles of azimuth and elevation, and position of the panel can be simultaneously monitored. The window screen of the system monitoring is given in Figure 9.

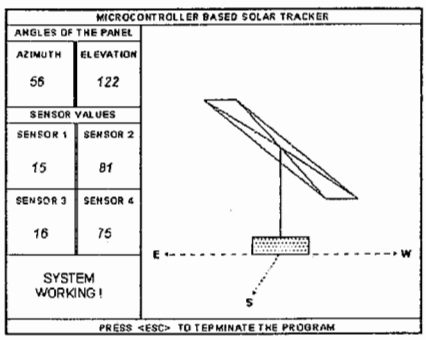


Figure 9. The window screen of the system monitoring

4. CONCLUSIONS

In order to collect the greatest amount of energy from the sun, solar panels must be aligned orthogonally to the sun. For this purpose, a new solar tracking technique based on micro-controller was implemented and tested in this study. The tracking system presented has the following advantages: The tracking system is not constrained by the geographical location of installation of the solar panel since it is designed for searching the maximum solar irradiance in the whole azimuth and tilt angle (except hardware limitations) during day times; namely, the angle of elevation does not need to be adjusted periodically. The operator interference is minimal because of not needing to be adjusted. The tracker provides also PC based system monitoring facility. Since the tracking system is controlled completely by MCU; the PC, used for monitoring the panel only, may not be employed. A drawback of the tracker is being effected by temporal variations in the atmospheric refractions caused by rain, cloud, fog, etc. Thus, the system may give an erroneous detection in the direction of the sun, and lead to wrong positioning of the solar panel.

5. REFERENCES

Anoymous, 1998a. High performance micro-controllers, Data Sheet, Microchip Technology Inc. 1.

Anoymous, 1998b. National Semicon. Data Sheet, pp:1024-28.

Davies, 1993. P.A.. Sun-Tracking Mechanism Using Equatorial and Ecliptic Axes. Solar Energy, v.50, 6:487-489.

Harakawa, T., Tujimoto, T. 2001. A proposal of efficiency improvement with solar power generation system. Industrial Electronics Society, 2001. IECON '01.The 27th Annual Conference of the IEEE, 1:523-8.

Konar, A., Mandal,. A. K. 1991. Microprocessor Based Automatic Sun Tracker. IEE Proc.-A. Vol. 138, 4:237-1.

Koyuncu, B., Balasubramanian, K. 1991. A micro-processor controlled automatic sun tracker. IEEE Trans. on Consumer Elects., Vol. 37, 4 : 913-7.

Kreider, J. F., Kreith, F. 1981. Solar Energy Handbook, McGraw-Hill Book Company, pp : 1-9.

Macagnan, M.H., Lorenzo, E. and Jimenez, C., 1994. Solar Radiation in Madrid. International Journal of Solar Energy, V. 16, 1 : 1-14.

Saxena, A. K., Dutta, V. 1990. A Versatile Microprocessor Based Controller for Solar Tracking. Photovoltaic Specialists Conference, Conference Record of the Twenty First IEEE, 2 : 1105-9.

Wilamowski, B. M., Xiangli, L. 2002. Fuzzy System Based Maximum Power Point Tracking for PV System. Industrial Electronics Society, IEEE 2002 28th Annual Conference of the IECON 02, 4 : 3280-4.

Yeong-Chau K., Tsorng-Juu L., Jiann-Fuh, C. 2001. Novel Maximum-Power-Point-Tracking Controller for Photovoltaic Energy Conversion System. Industrial Electronics, IEEE Trans., Vol. 48, 3 : 594-1.