DESIGN AND IMPLEMENTATION OF A MICROCONTROLLER BASED SOLAR ELEVATION TRACKER

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF BACHELOR OF ENGINEERING (HONOURS) IN ELECTRICAL AND COMPUTER ENGINEERING

BY

UMAR M MUSTAPHA

(2001/13935EE)

NOVEMBER, 2005

DECLARATION

I UMAR MOHAMMAD MUSTAPHA hereby declare that this thesis is Original work of mine. All information obtained from published and unpublished work has been acknowledged.

....

Signature

7/12/05

Date

ATTESTATION

This is to that this project "Design and Implementation of a Microcontroller Based Solar Elevation Tracker" was undertaken by "UMAR MOHAMMAD MUSTAPHA", in partial fulfillment of the requirements for the degree of Bachelor Of Engineering (honours) in Electrical and Computer Engineering of the Federal University of Technology, Minna, Niger State.

....

Engineer M. D Abdullahi (HOD) DATE

....

Mr. S N Rumola DATE

....

EXTERNAL EXAMINER

....

SIGNATURE: DATE

ACKNOWLEDGEMENT

This thesis project has benefited from review by Mr. S N Rumola, who gave generously his time and expertise. This thesis could never have been completed without his valuable suggestions and contributions.

DEDICATION

This work is dedicated to Almighty Allah, who has been the source of my inspiration, the source of all knowledge and understanding, without which I would have done nothing. And to my loving mother whose moral and financial contribution can not be overestimated.

ABSTRACT

This project explores the use of a sun tracking device to improve the power output of solar panels. The report discusses the design and implementation of the sun elevation t racking device including the support structure, the use of electric motors, and a metho d of determining the location of the sun. Solar panels help us tap into the immense ene rgy of the sun. However, stationary panels are only optimized for a portion of the day, and only produce a fraction of their maximum power output in the morning and eveni ng. By tracking the sun's elevation, the angle of incidence of the sun on the solar pane ls will be maximized, and the power output from the panels will be near maximum all day. The tracking is accomplished by a microcontroller, two pairs of Mosfet transistor s, a pair of Cadmium Sulfide photoconductive cells, a Phototransistor optocouplers, a Photointerrupter and a DC motor. The tracking portion of the project was successful u nder sunny conditions. The system was able to correctly aim the solar panel's frame at the sun's elevation, even when started from a variety of positions. However, the syste m was only able to perform near optimum tracking, as optimization is not the require ment for this project.

Table of Contents

Nomenclature

Acronym	Description
BJT	Bipolar Junction Transistor
MOSFET	Metal-Oxide-Semiconductor Field Effect Transistor
PWM	Pulse Width Modulate
SET	Solar Elevation Tracker
MCU	Microcontroller unit
ADC	Analog to digital converter
RISC	Reduced Instruction Set Controller
CMOS	Complementary Metal Oxide Semiconductor
ALU	Arithmetic Logic Unit
CISC	Complex Instruction Set controller
SEA	Solar Elevation Angle
IR LED	Infra red Light Emitting Diode
EMF	Electro Motive Force

List of Figures

List of Tables

List Graphs

Chapter One

INTRODUCTION

1.1 General

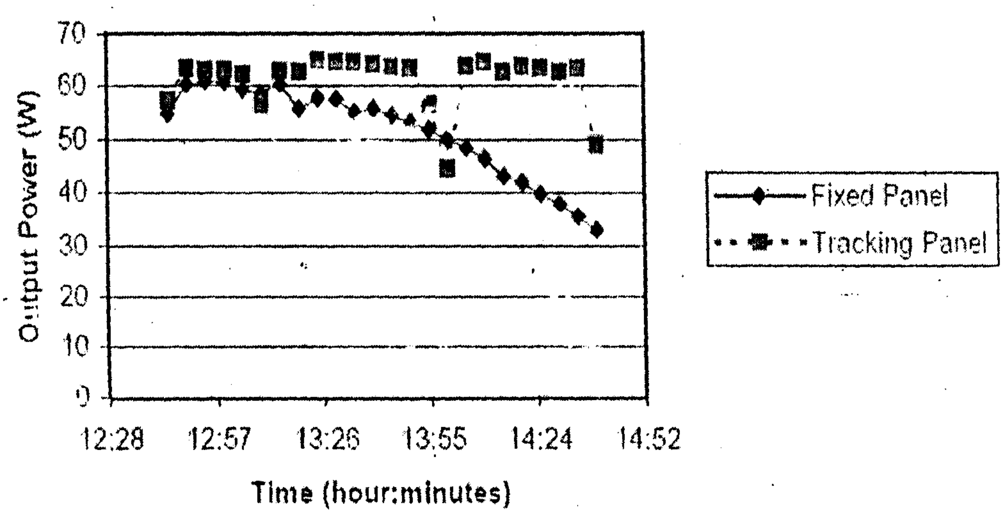
The sun is an ever present source of vast amounts of power. For example, Minna, Nigeria receives an average of 3.98 kWh of solar irradiation energy per day per square meter: a vast supply of power, which is largely untapped. Solar panels can be used to harness some of this power. The energy collected by a solar panel is always dependent in part on the angle of incidence of the sun upon the panel. Traditionally, solar panels are mounted on a fixed position, which leaves them in a sub-optimal orientation for most of the day. The various movements of the earth make the solar irradiation pattern to vary with time of the day and year. This has made it impossible to harness optimum amount of usable energy throughout the day. It became obvious that consistent alignment of the panel must be ensured to always obtain optimum irradiation energy from the sun. For optimum irradiation energy collection, both lateral and elevation alignments must be used. However, because the structure that supports, imparts and automates optimum alignment represents a large portion of the overall cost, this project incorporates only the elevation alignment to achieve a near optimum energy collection. [1]

1.2 Solar Tracking Approaches

Several approaches for tracking were considered, in which any could be either single or double axis tracking:

- Fixed panels (no tracking).
- Tracking the sun at a constant angular velocity over an arc, which approximately following the sun's path.
- Solar illumination feedback
- Use (and possibly calculate) the expected position of the sun in the sky.

Fixed panels are the simplest approach, in any terms. However, based on the panel's characteristics, shown in Graph 2-1, it is expected that moving panels can collect more power. For optimum collection, this approach requires some form of consistent manual tracking.



Graph 1-1: Comparison of power from a fixed and a tracking panel

The most straightforward form of tracking considered was *constant angular velocity tracking*. A reasonable, fixed arc about the horizon would be covered during a fixed period every day. Such an approach would yield better results than a fixed panel, but a more complex system could yield even a better results.

Using *solar illumination feedback* would involve using the observed effects of the sun to attempt to determine its position, and follow its movements. Such a system would perform better than constant angular velocity tracking, as it could track the sun with reasonable accuracy, and would adjust its behaviour for the sun's seasonal changes automatically. A simple implementation of such tracking would require a relatively small increase in system complexity.

A system using *solar position data would look up or calculate the sun's path* for a given day, and would follow that path. Such system could allow for very good tracking accuracy, and could be coupled with solar illumination feedback for even better performance. However, this approach is also the most complex, and for a system as small as this, such control is not necessary.

The single axis (elevation) solar illumination feedback for tracking was chosen for this project. Initially, it was intended to use the power readings from the solar panel itself to guide the system. However, due to complexity, cost and several anticipated problems, such as losing track of the sun in the event of power loss, or misinterpreting a drop in illumination, as it could either be a wisp of cloud or a movement of the sun. Instead, it was decided to use photosensors to give the system more information about its surroundings. To achieve this approach, two photosensors are mounted side-by-side facing away from the panel with an opaque divider between them. The configuration of the pair is shown in Figure 1-1:

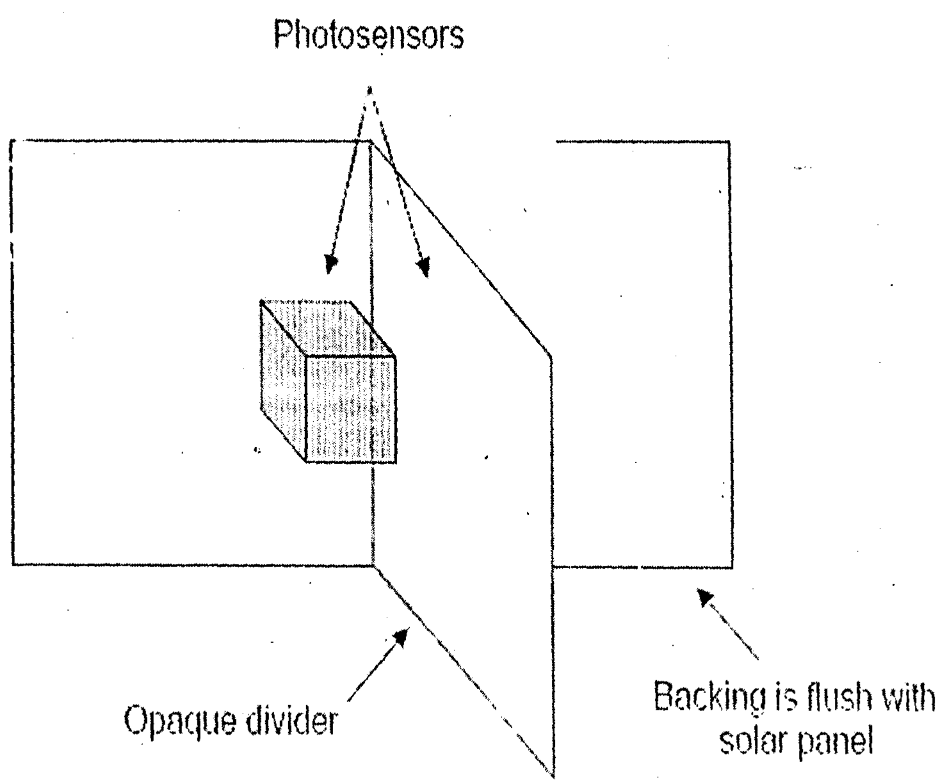


Figure 1.1: Photosensor configuration.

The opaque divider casts a shadow on the photosensor that is away from the sun, causing a greater difference between the intensity of light on the sensors. To align the panel with the sun, the system turns the panel toward the brighter sensor, until the opposite sensor becomes brighter. This procedure is illustrated in Figure 1-2: [1]

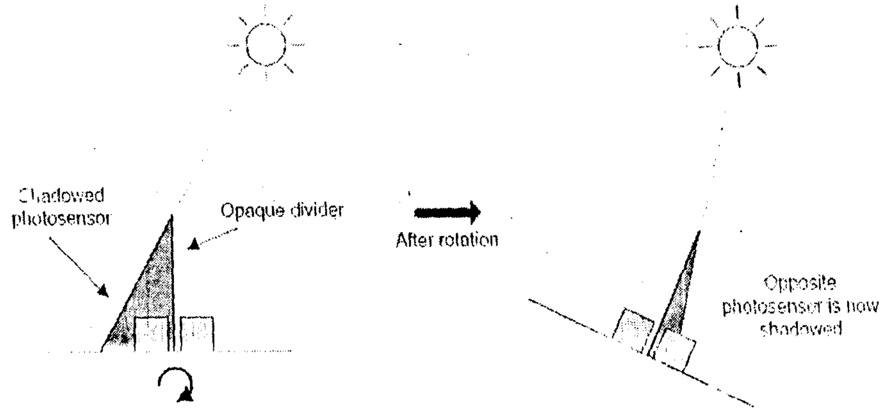


Figure 1.2: Illustration of tracking procedure.

1.3 Motivation

Where utility power such as PHCN is available, a grid-connected solar power system can supply some of the energy needed which could be use in place of batteries. Home, government and business owners are installing solar power systems connected to the utility grid. They do so because they know that the system reduces the amount of electricity they purchase from the utility each month. They also realize that solar power plants consume no fuel and produce no air or water pollution while they silently generate electricity without green house effects. Utilities can build solar power plants much more quickly than they can build conventional power plants because the arrays themselves are easy to install and connect together electrically. Utilities can locate solar power plants where they are most needed in the grid because siting solar cell arrays is much easier than siting a conventional power plant. Moreover, unlike conventional power plants, solar power plants can expand incrementally as demand increases. This system is ideal for remote applications such as communication stations, military installations, and rural villages for agricultural consumption. With all these benefits, solar power generation can be optimized by adding a tracking system to the solar array or panels. This tracking mechanism increases electrical output to about 20-30 percent when compared to a non-tracking(fixed or passive) solar array. [1, 2].

Chapter Two

LITERATURE REVIEW

2.1 Historical Perspective Of Solar Energy Application

The earliest known record of the direct conversion of solar radiation into mechanical power belongs to Auguste Mouchout, a mathematics instructor at the Lyce de Tours. Mouchout began his solar work in 1860 after expressing grave concerns about his country's dependence on coal. By the following year he was granted the first patent for a motor running on solar power and continued to improve his design until about 1880. During this period the inventor laid the foundation for our modern understanding of converting solar radiation into mechanical steam power.

Mouchout's initial experiments involved a glass-enclosed iron cauldron: incoming solar radiation passed through the glass cover, and the trapped rays transmitted heat to the water. In the late 1865, he succeeded in using his apparatus to operate a small, conventional steam engine. By the following summer, he enlarged his invention's capacity, refined the reflector, redesigning it as a truncated cone, like a dish with slanted sides, to more accurately focus the sun's rays on the boiler. Mouchout also constructed a tracking mechanism that enables the entire machine to follow the sun's altitude and azimuth, providing uninterrupted solar reception.

William Adams, the deputy registrar for the English Crown in Bombay, India built a large rack of many small mirrors and adjusted each one to reflect sunlight in a specific direction. To track the sun's movement, the entire rack could be rolled around a semicircular track, projecting the concentrated radiation onto a stationary boiler. The rack could be attended by a laborer and had to be moved only three or four times during the day, or more frequently to improve performance. Adam's legacy of producing a powerful and versatile way to harness and convert solar heat survives. Engineers today know this design as the *Power Tower concept*, which is one of the best configurations for large scale, centralized solar plants. As the years wore on, newer methods were designed for collecting power as well as tracking the sun. These included; Engineer Charles Tellier's method of collection without reflection. By 1889 Tellier had increased the efficiency of the collectors by enclosing the top with glass and insulating the bottom. Around 1870, U.S. engineer John Ericsson invented a novel method for collecting solar rays known as the parabolic trough. A parabolic trough is more akin to an oil drum cut in half lengthwise that focuses solar rays in a line across the open side of the reflector. This type of reflector offered many advantages over its circular counterparts: it was comparatively simple, less expensive to construct, and unlike a circular reflector, it only track the sun in a single direction thus eliminating the need for complex tracking machinery. The downside was that the device's generated energy and efficiencies were not as high as with a dish-shaped reflector. The First Commercial Venture was by Aubrey Eneas who began his solar motor experimentation in 1892, and formed the first solar power company (The Solar Motor Co.) in 1900. Though the machine did not become a fixture as Eneas had hoped, the inventor contributed a great deal of scientific and technical data about solar heat conversion and initiated more than his share of public exposure. [3]

2.2 Previous works

The past 25 years have witnessed the emergence of various methods of solar energy collection as a result of the improvement in technology witnessed within this period. Some of our brightest engineers have even produced some exemplary designs during the period.

A shadow method for automatic tracking, which is an automatic method that uses 'back-to-back' semi-cylinders to mask solar irradiation was described and presented for publication at the Solar World Congress 1987 in Hamburg by Sode Shinni Nmadu Rumala. [4]

A time-based solar tracking system was also designed based on single axis tracking, on the equatorial tracking axis to track the sun from east to west daily during sun hour periods. An open loop control mode was adopted using logic control circuit and suitable interface for the stepper motor and other circuitry. [2]

2.3 The Most Rational Source

The afore mentioned solar pioneers were only few of the most notable inventors involved in the development of solar thermal power from 1860 to date. Many others contributed to the more than 50 patents and the scores of books and articles on the subject Solar technology already boasts a century of R&D, requires no toxic fuel and relatively little maintenance, is inexhaustible, and, with adequate financial support, is capable of becoming directly competitive with conventional technologies in many locations. These attributes make solar energy one of the most promising sources for many current and future energy needs. As Frank Shuman declared more than 80 years ago, it is "the most rational source of power." [3]

Chapter Three

DESIGN AND IMPLEMENTATION

3.1 Overview

The solar elevation tracker is essentially a closed loop control system that covers both the fields of electronics and mechanical engineering. The components of the electronic system consist of a MCU logic circuitry, Phototransistor optocouplers as Isolators. DC motor driver (H-Bridge), Photointerrupter used as Incremental Optical Shaft Encoder (Tachometer, Limit Sensor) and Cadmium Sulphide photoconductive cells (photosensors). The DC motor driver is controlled by a PWM signal produced by the MCU. Using data acquisition and processing, the MCU is able to determine the position, and limit range of the solar panels. Figure 3-1 shows a block diagram of the system.

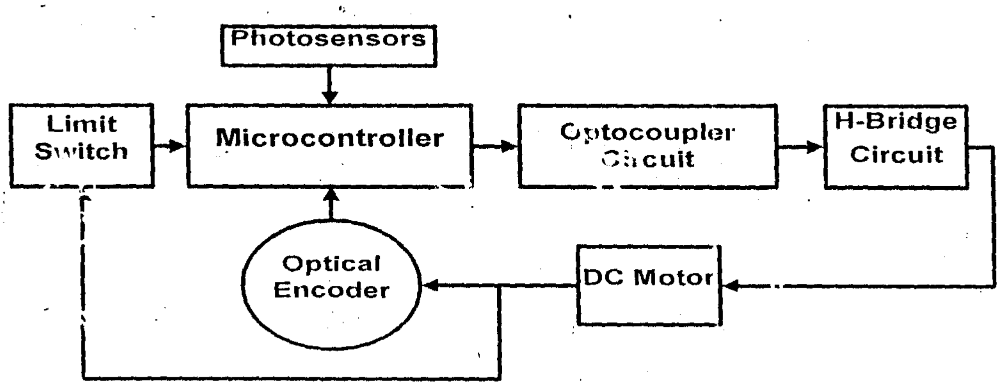


Figure 3-1: Block diagram of the system

3.2 Electrical System Description

3.2.1 Photosensor

Cadmium sulphide (Cds) photoconductive cells are used as photosensors. It is a type of light dependent resistor semiconductor that operates much like a thermistor. In an intrinsic semiconductor, photons (Blue arrow) can promote electron (Black arrow) to the conduction band leaving a hole in the valence band. This increase in carriers leads to a reduction in the resistance. That is, as light hits the Cds it knocks charge carriers loose, and the increased number of carriers decreases the resistance of the device. Cadmium sulphide photoconductive cell has energy gap of 2.42 electron volts (eV) and a wavelength of wavelength 5130 (ė). Figure 3-2 shows the Schematic symbol and energy band of Cadmium sulphide photoconductive cell. [5, 6]

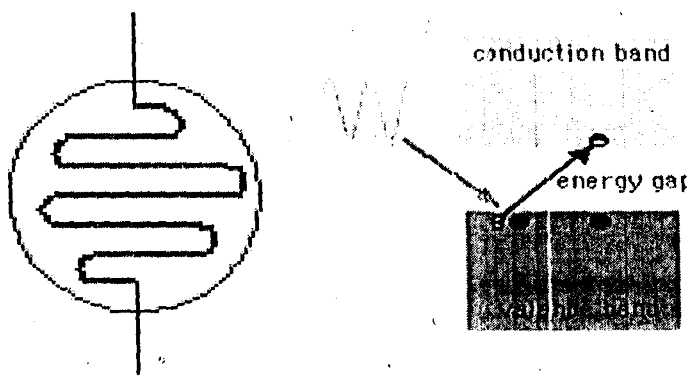


Figure 3-2: Schematic symbol and Energy band of Cds cell

3.2.2 Phototransistor Optocoupler

Phototransistor Optocoupler can be used to provide isolation between components, to avoid ground loop problems, to control floating electronics, and to provide DC shifts.Also applicable to Power supply regulators, Digital logic inputs and Microprocessor inputs. Figure 3-3 shows its shematic symbol and 6-pin dual in-line package.

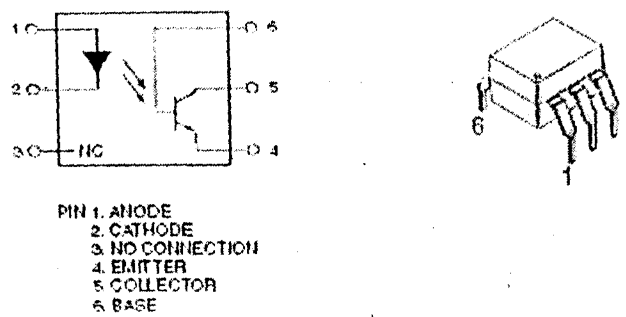


Figure 3-3: Optocoupler schematic symbol and 6pin dual in-line package

The general purpose optocoupler consist of a gallium arsenide infrared (IR) emitting diode spectrally matched to drive a silicon phototransistor. The IR diode section gives up excess energy in the form of photon when electrons and holes combine in the depletion region. This energy is a characteristic of the materials from which the junction is constructed and thus the photon always has the same wavelength. To emit Photons, the junction must be forward biased and the wavelength must also be in the infrared spectrum. The phototransistor section of the optocoupler is an NPN bipolar transistor (BJT) with a large base that does not have a lead. When photons (from IR emitting diode) hit the base they create electron/hole pairs, the electrons are drawn to the collector and the holes are filled with electrons from the emitter. [6,7,10]

3.2.3 H-Bridge

A microprocessor or microcontroller (MCU) cannot drive a motor directly, because it can not supply enough current. However, there must be some interface circuitry to amplify the power to drive the motor from the MCU. This interface circuitry was implemented using a circuit design known as the H-bridge. It is the primary means for driving a motor in the forward and reverse directions. It merely consists of 4 switches A, B, C, D connected in topology of an H, where the motor terminals form the crossbar of the H. Figure 3-4 shows a H-bridge circuit design

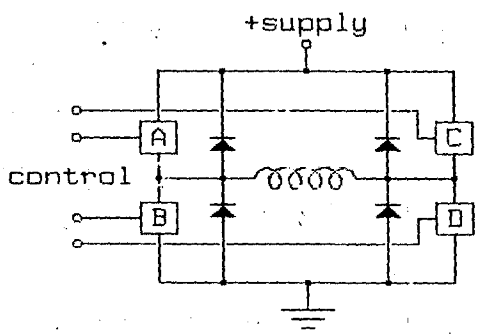


Figure 3-4 H-Bridge circuit

It is worth noting that H-bridges are not only applicable to the control of motors, but also to the control of push-pull solenoids (those with permanent magnet plungers) and many other applications. With 4 switches, the basic H-bridge offers 16 possible operating modes, 7 of which short out the power supply.

The desired operating modes are illustrated in table-3.1.

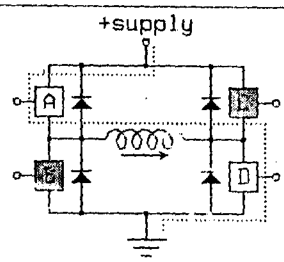
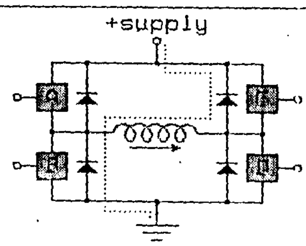
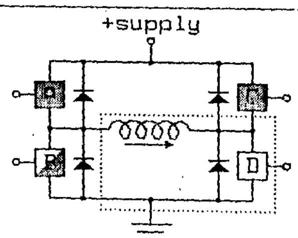
Mode	A	B	C	D	Figure	Comment
1	1 0	0 1	0 1	1 0		(1)Forward (2)Reverse
2	0	0	0	0		Fast decay (coasting)
3	0	X	0	1		Slow decay or. dynamic braking

Table 3-1- H-Bridge operating modes (Close = 1, Open = 0)

MODE 1: Forward and Reverse are the usual operating modes, allowing current to flow from the supply, through the motor winding and onward to ground.

MODE 2: in Fast decay or coasting mode, any current flowing through the motor winding will be working against the full supply voltage, plus two diode drops, so current will decay quickly. This mode provides little or no dynamic braking effect on the motor rotor, so the rotor will coast freely if all motor windings are powered in this mode.

MODE 3: In Slow decay or dynamic braking modes, current may recirculate through the motor winding with minimum resistance. As a result, if current is flowing in a motor winding when one of these modes is entered, the current will decay slowly, and if the motor rotor is turning, it will induce a current that will act as a brake on the rotor.

[8, 9, 10]

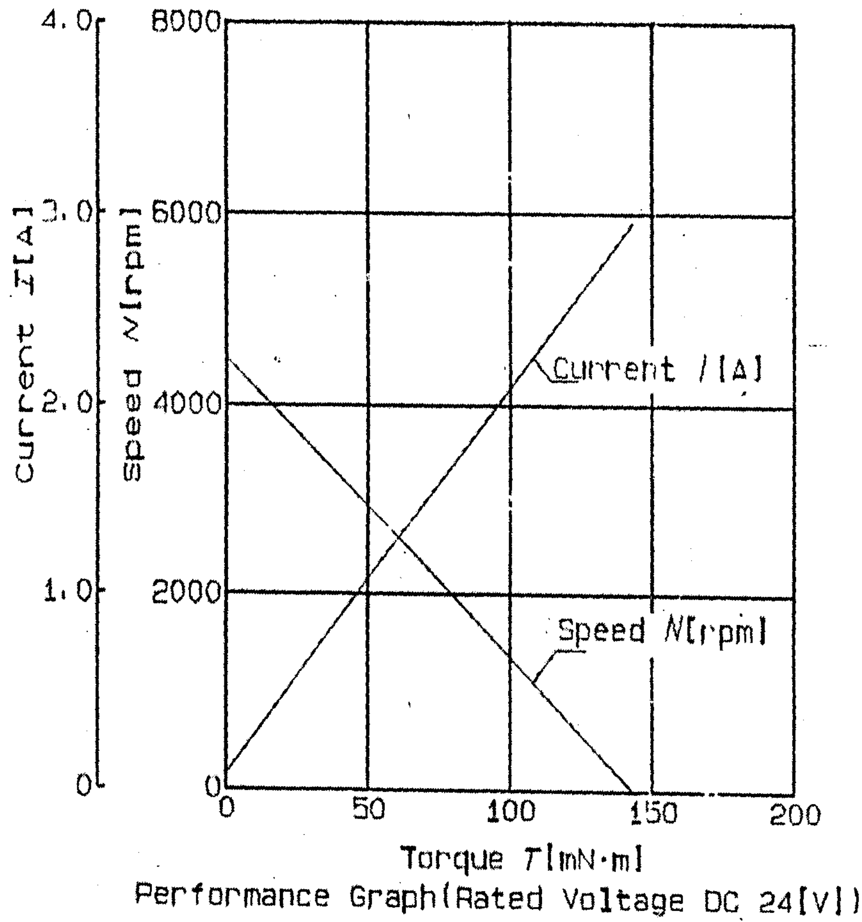
3.2.4 DC Motor

The direct current (DC) motor is one of the first machines devised to convert electrical energy to mechanical power. Its origin can be traced to machines conceived and tested by Michael Faraday, the experimenter who formulated the fundamental concepts of electromagnetism. Most of the world's adjustable speed business is addressed by DC motors. DC motors are utilized in applications where speed control, servo control, and/or positioning needs exist. Several characteristics are important in selecting a DC motor. Listed below, the first two are DC Motor's input ratings that specify its electrical characteristics and the last three are ratings describing the motor's output characteristics:

- Operating Voltage (V)
- Operating Current (I)
- Speed (RPM)

- Torque (Nm)
- Power (V)

The power delivered by a motor is the product of its speed and the torque at which the speed is applied. Graph 3-1 shows relationship between these characteristics.



Graph 3-1- DC Motor Characteristics

[9; 11]

3.2.5 Optical Encoder

Optical encoder is a type of feedback device that gives information on the system's stabilization, speed and position. It is simply used as a digital tachometer for absolute and incremental position encoding. When the motor's shaft is rotated the encoder gives an output signal proportional to distance the (i.e. angle) the shaft is rotated through. It provides a specific address for each shaft position throughout 360 degrees coded onto a disk. The number of tracks on the coded disk may be increased until the desired resolution or accuracy is achieved. A light source passes a beam through the transparent segments onto the encoder's photosensor which outputs a sinusoidal waveform or pulse train. If the output signal is just a sinusoidal waveform, electronic processing can be used to transform the signal into a square pulse train. This is illustrated in figure 3-5.

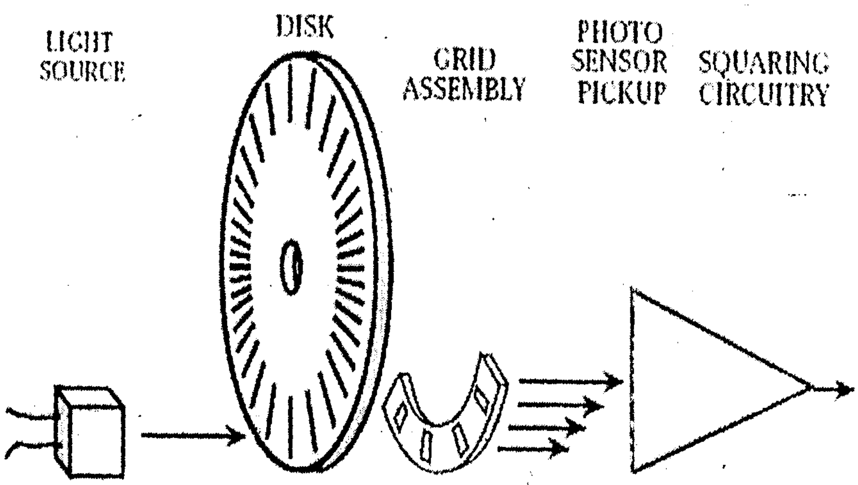


Figure 3-5: Incremental Encoder

In utilizing this device, the following parameters are important:

- **Line count-** This is the number of pulses per revolution. The number of lines is determined by the positional accuracy required in the application.
- **Output signal-** The output from the photosensor can be either a sine or square wave signal.
- **Number of channels-** Either one or two channel outputs can be provided. The two channel version provides a signal relationship to obtain motion direction (i.e. clockwise or counterclockwise rotation).

In addition, a zero index pulse can be provided to assist in determining the "home" position. A typical application using an incremental encoder is as follows: An input signal loads a counter with positioning information. This represents the position the load must be moved to. As the motor accelerates, the pulses emitted from the incremental (digital) encoder come at an increasing rate until a constant run speed is attained. During the run period, the pulses come at a constant rate which can be directly related to motor speed. The counter, in the meanwhile, is counting the encoder pulses and, at a predetermined location, the motor is commanded to slow down. This is to prevent overshooting the desired position.

When the counter is within 1, 2 pulses of the desired position, the motor is commanded to stop with the load is now in position.[5]

3.2 6 Limit Switch

The limit switch provides the system with a maximum and minimum allowable travel range. Zero index pulse signal from the Optical encoder determines either of the limits which after a certain delay, the "home" position will be reset. It is directly connected to the MCU External interrup0 pin (INT1) with that of the Optical encoder connected to External interrup1 pin (INT0).[5]

3.2.7 Microcontroller

A microcontroller (MCU) is a single computer chip (integrated circuit) that executes a user program, normally for the purpose of controlling some device. It is ideal for the types of applications where cost and unit size are very important considerations. Nowadays it is almost always desirable to produce circuits that require the smallest number of integrated circuits, that require the smallest amount of physical space, require the least amount of energy, and cost as little as possible. The type used in this project is an ATmega16 which is a low-power CMOS 8-bit MCU based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega16 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

This particular Atmel MCU family was chosen because its AVR core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers such as the 8051/8052.

The ATmega16 provides the following features: 16K bytes of In-System Programmable Flash Program memory with Read-While-Write capabilities, 512 bytes EEPROM, 1K byte SRAM, 32 general purpose I/O lines, 32 general purpose working registers, a JTAG interface for Boundary-scan, On-chip Debugging support and programming, three flexible Timer/Counters with compare modes, Internal and External Interrupts, a serial programmable USART, a byte oriented Two-wire Serial Interface, an 8-channel, 10-bit ADC with optional differential input stage with programmable gain, a programmable Watchdog Timer with Internal Oscillator, an SPI serial port, and six software selectable power saving modes. Figure 3-6 shows the detailed block diagram of such MCU.

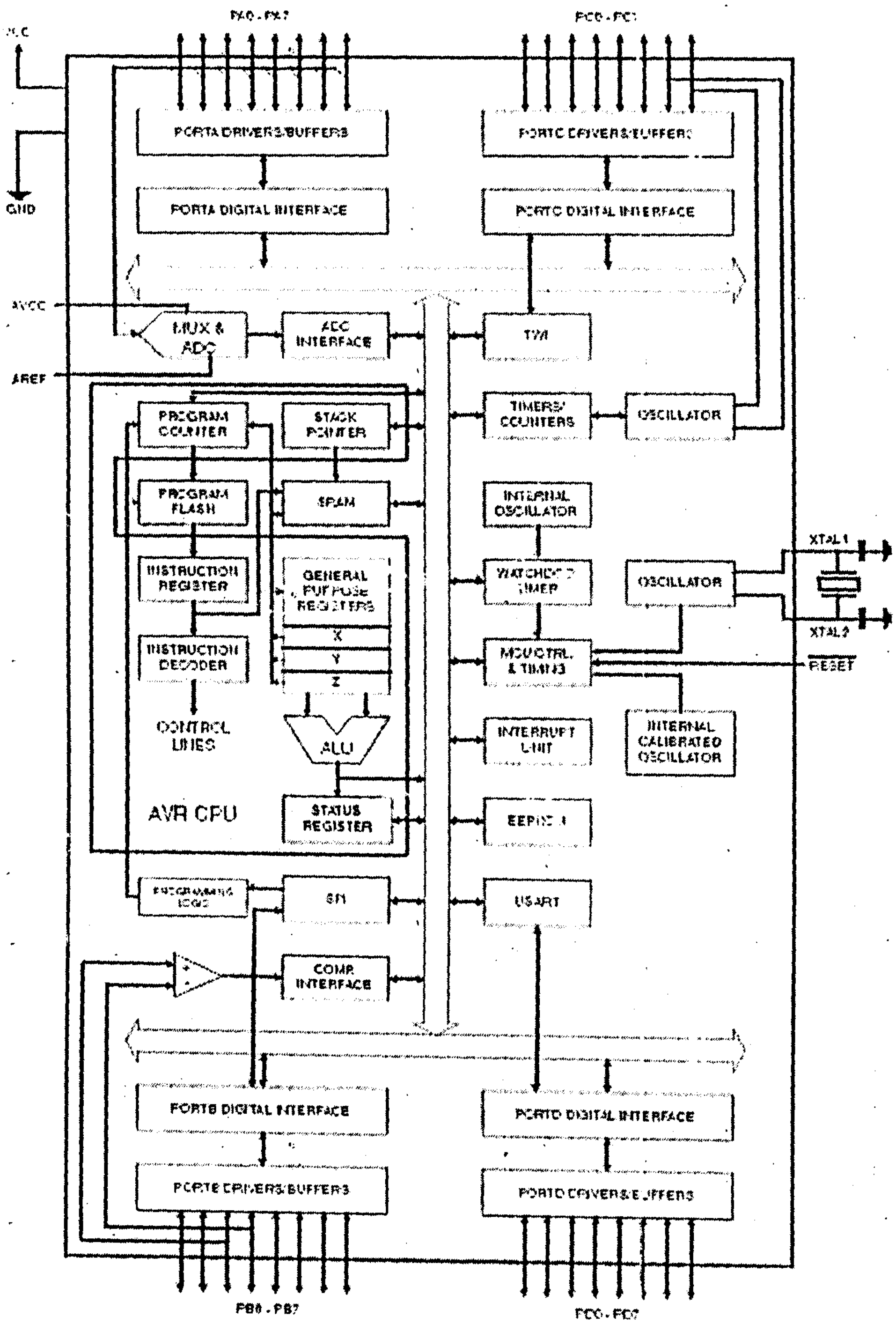


Figure 3-6: ATmega16 Block Diagram

The ATmega16 AVR is supported with a full suite of program and system development tools including: C compilers, macro assemblers, program debugger/ simulators, in-circuit emulators, and evaluation kits [12, 13]

3.3 Electrical System Design

3.3.1 Optocoupled Motor Control Circuit

The Optocoupled motor control circuit is not very complex, it is an integral part of the sun tracking machine. It must be able to interface to the MCU, turn the motor on and off, allow the current through the motor to be reversed, and control the speed of the motor for tracking adjustments. The circuit being used in a power tracking system, must also be able to track with minimum power usage. However, DC motors require a relatively high voltage for the initial start. Lower voltages don't give the motors enough torque, which made it unreliable and very sluggish. In order to solve this problem and conserve power, PWM control signal generated from the MCU was used to pulse the power to the Optocoupled motor control circuit. By adjusting the duty cycle of the PWM signal, various voltage levels can be generated. When the duty cycle is set at 100%, the motor receives all of the voltage being supplied to it, but when the duty cycle is 50%, the effective voltage at the motor is half of the voltage being supplied to it [15]. Thus, the speed of the motors can be varied, while the torque remains constant with minimum power usage. The inputs for the Optocoupled motor control circuit were attached directly to the outputs of the MCU. The Optocoupled motor control circuit is shown in figure 3-7. [11, 12]

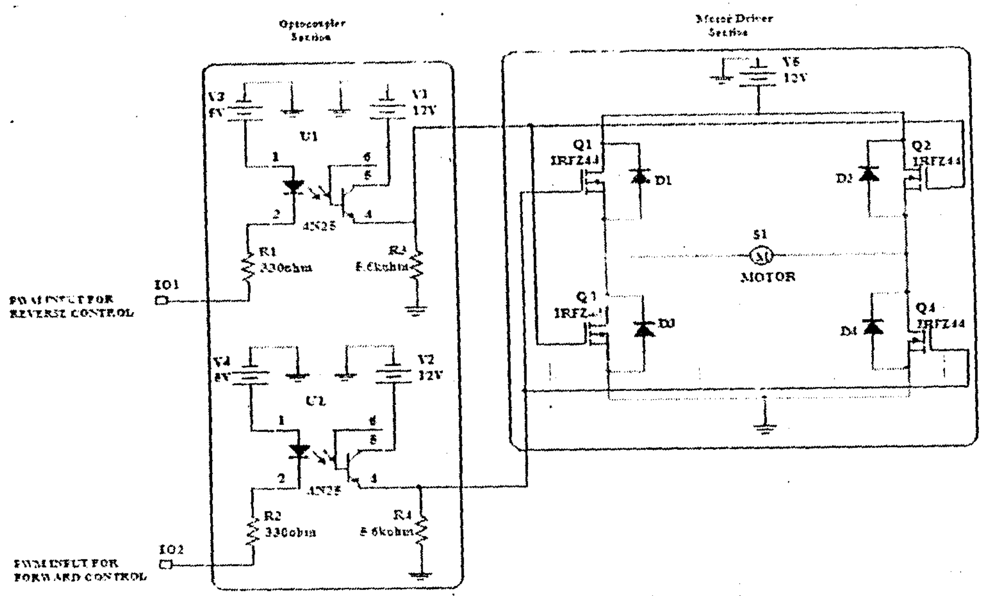


Figure 3-7: Optocoupled Motor Control Circuit

To completely isolate the motor driver circuit, which consist of the H-Bridge, from the MCU, two 4N25 phototransistor Optocouplers were used to detach the two MCU lines from their H-Bridge counterparts. This device uses an infrared diode to turn a phototransistor on and off. When the MCU passes a low level signal, current passes through the photodiode and the phototransistor is essentially a closed switch; its output voltage level will be raised from ground to 12V. When the MCU passes a high level signal, the photodiode is off and the phototransistor is an open circuit; with its output pulled down to ground through the 5.6kOhm resistor. The optocouplers are in an inverting configuration, that is, rather than inputting a signal referenced to 5v, ground was used as the input and was referenced to each of the two signals. [5, 15] Once the signals are inverted through the optocoupler's phototransistor, the correct signals are passed to the H-bridge, with exception that a logical high is 12V rather than 5V. Since the optocouplers use phototransistors that receive IR signals, there is no physical connection between the MCU's circuit and the H-bridge. Hence, no noise from the motor can go to the MCU.

Applying a potential across the driver circuit (motor leads) will cause the motor to spin in one direction, while reversing the polarity on the leads will cause the DC motor to pin in the opposite direction. To control the motor's forward and reverse motion, a simple 4 switch device is required (See Table 3.1) that would ensure protection against a short of the DC supply, that is, switches 1 and 2 or 4 and 3 could never be closed at the same time. To operate the motor in one direction, simply close switches 1 and 4; for reverse operation, open switches 1 and 4 and close switches 2 and 3. To stop the motor completely, simply open all switches. To properly control the motor with high speed switching, a circuit configuration called an H-bridge (See Figure 3.6) was implemented.

MOSFET transistors were implemented as switches in place of BJTs for one reason , to improve the efficiency of the bridge. This is because the BJT transistors (normal transistors), though more linear with better gain have a saturation voltage of approximately 1V across the collector emitter junction when turned on. A H-bridge having power supply of 12V would be consuming 2V (16.6%) across the two transistor required to control just the direction of the current. It is therefore inefficient for an energy collection project to waste 16.7% of its supply just to control current direction. The BJTs also would get quite hot and there is no room for heat sinks. However, IRF44 MOSFETs were used simply because of there low ON resistance ($R_{DS(ON)}$ =0.023Ω). This is the resistance between the Drain and Source when turned on. At 4 amps, this makes the voltage drop per MOSFET to be 0.092V and 0.184V (1.5% of the power sup ply) for the two required to control the direction of the current. This is a definite improvement on the driver's efficiency.

Four protection diodes were used; one per MOSFET to ensure there is never a short from the supply to ground. D1 to D4 route back EMF from the motor back to the power supply. Some MOSFETs (actually most) have these diodes built-in, so they may not be necessary. MOSFETs work by applying a voltage to the Gate. They call this TRANSCONDUCTANCE, which is the rate of change of the drain current with a change of the gate bias. When a positive voltage greater than the Gate threshold voltage is applied, the MOSFET turns on (N Channel only. The P channel works in reverse). One of the most important uses of MOSFETs is to build logic circuits that dissipate very little power [1, 6, 10]. See Appendix B for the DC operating Point Analysis of circuit the circuit in figure 3.7 for both forward and reverse mode.

3.3.2 ATmegma16 MCU

3.3.2.1 Hardware

The electronic system was implemented around an ATmega16 MCU running on 4 MHz Internal RC Oscillator for tracking algorithm to find the Solar Elevation Angle (SEA). The MCU monitors the operating state of the system to ensure proper operation and sequence of events. For memory map showing used registers, see register summary in Appendix E. The following MCU features were used in this project:

- 8-bit timer/counter0 and 16-bit timer/counter1
- Analog to Digital Converter with 10 bit resolution
- Input/Output ports
- Internal RC Oscillator

The 8-bit timer/counter0 and 16-bit timer/counter1 controls the motor's permissible ON/OFF durations, and fast PWM wave generation respectively. The 8 bit Timer/counter0 was configured to overflow anytime its count value reaches 255. This overflow rate was used to provide the *PWM pulse train time* required by the motor just before it saturates (when the generator portion-Back EMF of the motor matches the supply voltage). At saturation point, negligible current will flow into the motor and thus it will produce negligible torque (required to overcome the spinning friction).Using a method of trial and error, the appropriate timing was set to 0.05 sec.

To achieve this timing, the timer/counter0 operating at 4 MHz needed 195 overflows.

The simplified block diagram of the 8-bit Timer/Counter0 is shown in Figure 3-8.

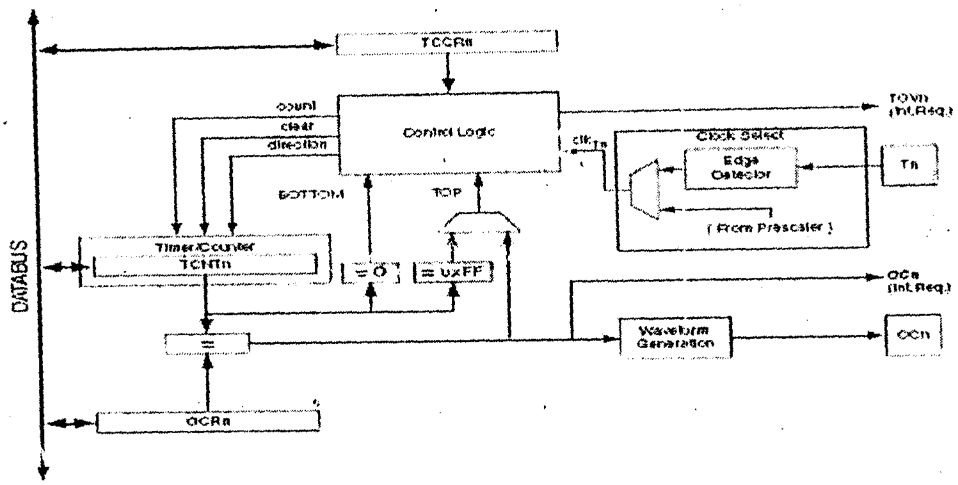


Figure 3-8: 8-bit Timer/Counter0

The 16-bit Timer/Counter1 allows accurate program execution timing, event management, wave generation, and signal timing measurement. The wave generation mode 14 (see Appendix F) was used to generate the fast PWM signal required to drive the motor in either forward or reverse motion. The simplified block diagram of the 16-bit Timer/Counter1 is shown in Figure 3-10.

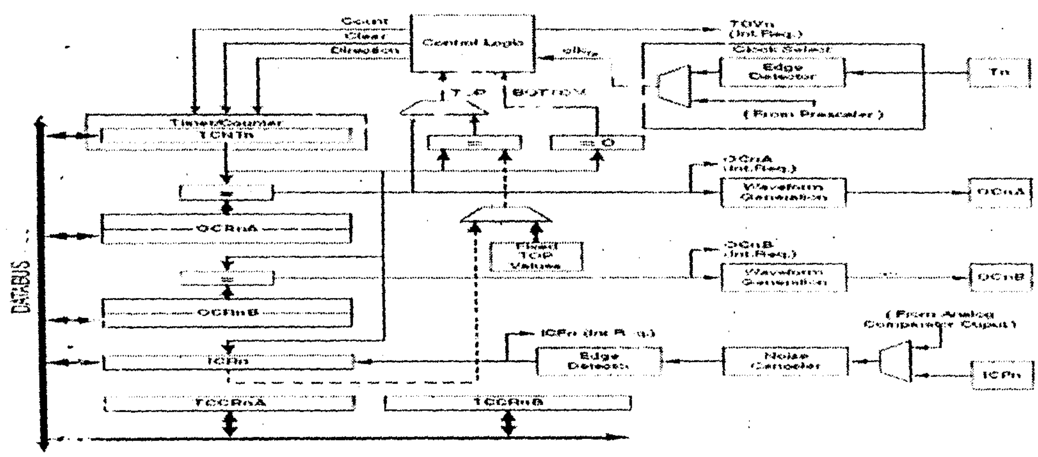


Figure 3.10 16-bit Timer/Counter1

Proper timing of both timers was achieved by configuring there respective prescalers to further scale down the frequency to 3.9 KHz (4MHz/ 1024) values. Bits CS02, CS00 were set and CS01 cleared in the timer0 control register, also bits CS12, CS10 were set and CS11 cleared in the timer1 control registerB. Figure 3-9 shows the prescaler block diagram of timer 0&1 together with the table 3.2 for prescaler clock select bit description.

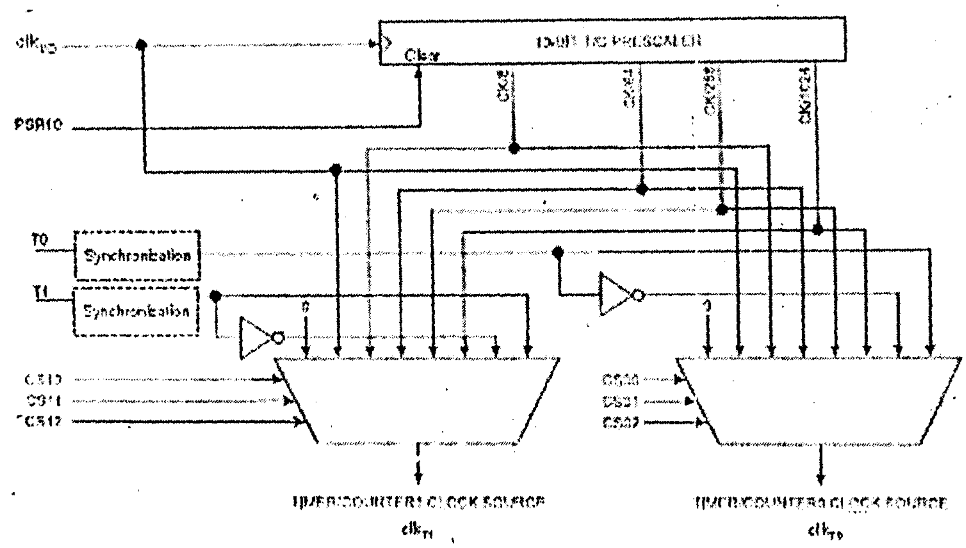


Figure 3-9: Prescaler block diagram of timer 0&1

CS02	CS01	CS00	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	$clk_{I/O}$/(No prescaling)
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/32 (From prescaler)
1	0	0	$clk_{I/O}$/256 (From prescaler)
1	0	1	$clk_{I/O}$/1024 (From prescaler)
1	1	0	External clock source on T0 pin. Clock on falling edge.
1	1	1	External clock source on T0 pin. Clock on rising edge.

Table 3-2- clock select bit description

The ATmega16 features a 10-bit successive approximation ADC. The ADC is connected to an 8-channel Analog Multiplexer which allows 8 single-ended voltage inputs constructed from the pins of Port A. The single-ended voltage inputs refer to 0V (GND). The ADC contains a Sample and Hold circuit which ensures that the input voltage to the ADC is held at a constant level during conversion. A block diagram of the ADC is shown in Figure 3.13. [12, 13,14]

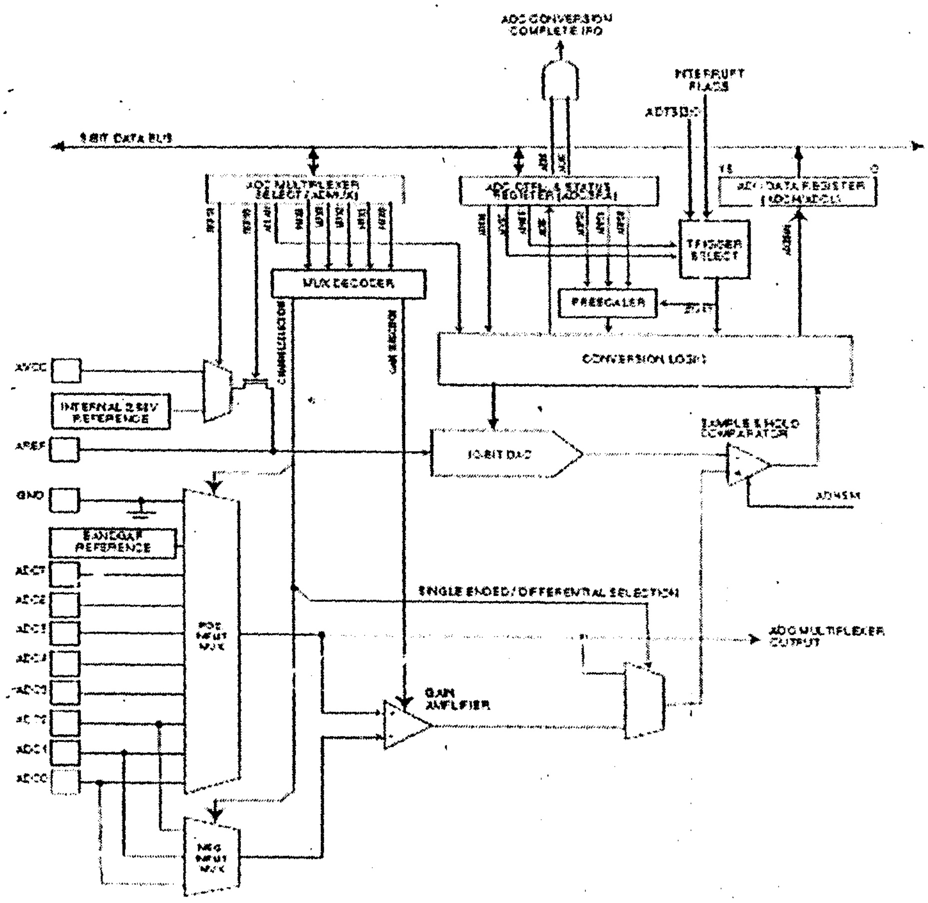


Figure 3.13 Analog to Digital Converter Block Schematic

The ADC has a separate analog supply voltage pin, AVCC. AVCC must not differ more than 0.3 V from VCC. Internal reference voltages of nominally 2.56V or AVCC are provided On-chip. The voltage reference may be externally decoupled at the AREF pin by a capacitor for better noise performance. The ADC AVCC was used as the reference and was directly connected to the MCU's VCC. The ADC input channels, ADC1 (pin A1) and ADC3 (pin A3) were connected to one of each of the two Cds Photosensors lead respectively. The other leads were also connected to the ADC0 (pin A0) and ADC2 (pin A2) respectively. The analog input channel was selected by writing to the MUX bits in ADMUX. The ADC input pins, as well as GND and a fixed band gap voltage reference, were selected as single ended inputs to the ADC. The ADC was enabled by setting the ADC Enable bit, ADEN in ADCSRA. After the completion of every conversion, the ADC generates a 10-bit result which was presented in the ADC Data Registers, ADCH and ADCL. The result was presented left adjusted by setting the ADLAR bit in ADMUX register because no more than 8-bit precision is required. This adjustment allows just the high bytes of the converted signal value contained in ADCH to be read. A single conversion was started by writing a logical one to the ADC Start Conversion bit, ADSC. This bit stays high as long as the conversion is in progress and will be cleared by hardware when the conversion is completed. The ADC has its own interrupt which can be triggered when a conversion completes. For single ended conversion, the result is;

$$ADC = V_{IN} * 1024 / V_{REF}$$

The ADC module contains a prescaler, which generates an acceptable ADC clock frequency from any CPU frequency above 100 kHz. The prescaling is set by the ADPS bits in ADCSRA. The prescaler starts counting from the moment the ADC is switched on by setting the ADEN bit in ADCSRA. The prescaler keeps running for as long as the ADEN bit is set, and is continuously reset when ADEN is low. The ADC prescaler unit is shown in figure 3.14 and its selection table at Table 3-4 .

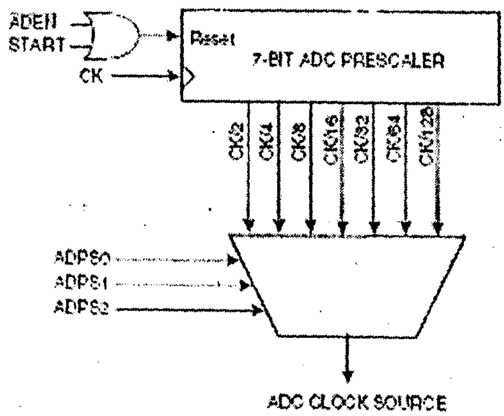


Figure 3.14 ADC prescaler unit

ADPS2	ADPS1	ADPS0	Division Factor
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

Table 3-4- ADC prescaler selection table.

The ADPS2 and ADPS1 were set in order to sample the photosensors at 62.5 KHz (F_{CPU}/64). [12, 13] See Appendix C on how the Cds photocells are connected to the MCU's ADC. The pull-up resistor (internal to the MCU) were activated in order to provide a relative voltage using the potential divider rule. The Cds (LDR) is connected as the base resistor to provide proportionate behaviors with the voltage seen at the ADC pin A1and A3 inputs. Alternatively, pin A0 and A1 provide the groung logic outputs.

Similarly the external interupt pin (PD2-int0 and PD3-int1) pull-ups were activated. Both interupts are triggered on falling edge of the sensed signal. Interupt0 gives the system its current position after and before travel so that both the lower and upper limits are not exceeded. Interup1 provides the reset home position whenever the system restarts.

3.3.2.2 Software

The control program was written in C (rather than assembly language) because the program requires math functions not directly supported in AVR assembler, and speed was not much of an issue. The sensors were sample 62.5K times per second, and the processor was configured to runs at 4MHz, therefore only about 64 cycles were available to compute and process it before the next input comes in.

The main program initializes the system and then goes into an endless wait loop. Everything happens on the loop unless when an interrupt triggers. Timer0 and 1 runs off an internal 4MHz resonator. Other than that, the software architecture is straightforward, just a linear sequence of commands as illustrated in flow chart in Appendix D. For more details see the code listing (with comments in Appendix F). [12, 13, 14, 15]

3.4 Design and Construction of the Mechanical Support

3.4.1 Design of a suitable structure.

A large component of the project was the design of a support structure for the solar panels. The design of the support structure has several stringent constraints:

1. The design must be able to support two solar panels.
2. The support structure must allow movement in one axis. The system must be capable of at least 90 degrees of vertical adjustment.
3. The entire support structure must be balanced about its axis of elevation, in order to keep its power consumption of the motor to a minimum.
4. The structure must be able to resist significant force from wind.

3.4.2 Construction

In order to meet the requirements above for a suitable support structure, a link-mechanism system was interwoven with a worm gear system to create a balanced platform as shown.

3.4.2.1 The Link Mechanism.

The function of a link mechanism is to produce rotating, oscillating, or reciprocating motion from the rotation of a crank or *vice versa*. Stated more specifically linkages may be used to convert:

1. Continuous rotation into continuous rotation, with a constant or variable angular velocity ratio.
2. Continuous rotation into oscillation or reciprocation (or the reverse), with a constant or variable velocity ratio.
3. Oscillation into oscillation, or reciprocation into reciprocation, with a constant or variable velocity ratio.

3.4.2.2 Slider-Crank Mechanism

The four-bar mechanism has some special configurations created by making one or more links infinite in length. The slider-crank (or crank and slider) mechanism below is a four-bar linkage with the slider replacing an infinitely long output link.

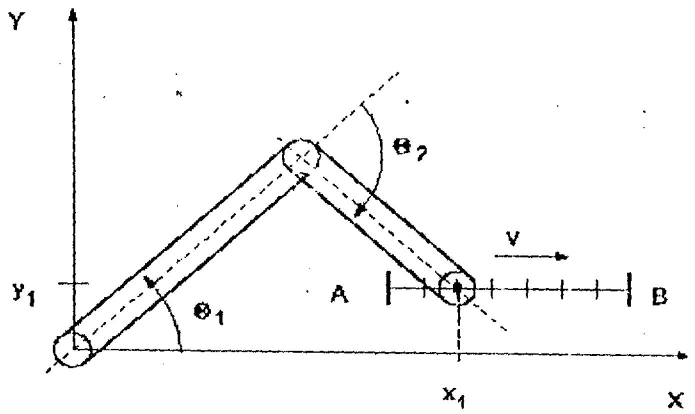


Figure 3.16 Crank and Slider Mechanism

This configuration translates a rotational motion into a translational one. Most mechanisms are driven by motors, and slider-cranks are often used to transform rotary motion into linear motion. To conveniently calculate the total kinetic energy of each link, it is divided into infinitesimal parts and integrated along the length of each link.

The Kinetic energy of a mass m moving with velocity v is $(1/2)mv^2$. Thus the total kinetic energy of a link can be obtained from

$$K = \frac{m}{2\ell} \int_0^{\ell} v^2(s)\, ds$$

Where s is the distance along the link and v(s) is the velocity located a distance s from one end assuming that all the mass is concentrated along a line and is distributed uniformly from one end to the other with linear density m/l. In general, v(s) is of the form;

$$V^2(s) = a + bs + cs^2$$

Then clearly,

$$K = (m/2l)\,[al + bl^2/2 + cl^3/3]$$

3.4.2.3 The Worm Gear

Gears are used in most mechanical devices. They do several important jobs, but most important, they provide a gear reduction in motorized equipment.

The small dc motor spins very fast and can provide enough power for the device, but not enough torque. The motor only produces a small amount of torque at a high speed. With a gear reduction, the output speed can be reduced while the torque is increased.

Another thing the gear does is adjust the direction of rotation. There are a lot of intricac ies in the different types of gears. However the worm gear was employed in this project due to the property discussed below. Worm Gear drives are the smoothest and quietest form of gearing when properly applied and maintained. They were considered for the following requirements:

- HIGH RATIO SPEED REDUCTION
- LIMITED SPACE
- RIGHT ANGLE (NON-INTERSECTING) SHAFTS
- GOOD RESISTANCE TO BACK DRIVING

A Worm gear was used due to its large gear reductions. It is common for worm gears to have reductions of 20:1, and even up to 300:1 or greater. Many worm gears ha ve an interesting property that no other gear set has: the worm can easily turn the gear, but the gear cannot turn the worm. This is because the angle on the worm is so shallow that when the gear tries to spin it, the friction between the gear and the worm holds the worm in place.

This feature was most useful for the structural link system, in which the locking feature acted as a brake for the system when the motor is not turning.

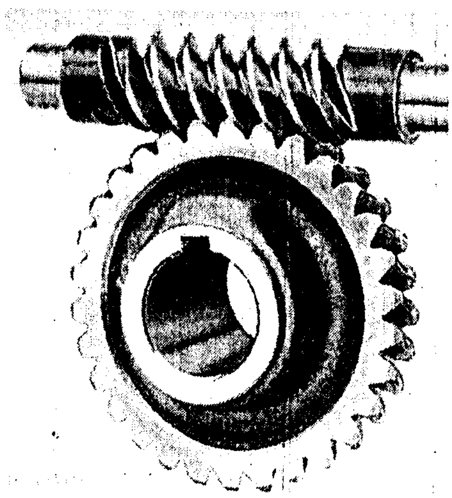


Figure 3.17 Worm gear

3.4.3 The Complete structure

The complete structure was an arrangement of the Link and Gear as shown in Appendix H. The motor is connected to point 'A'. This spins the worm at a high speed. As the worm spins, the rotational motion is translated into a linear motion by the gear. Since the longer bar of the link system is connected to the gear at point 'B', and because the shorter bar is also connected to a fixed point, the linear motion is transferred from 'B' to point 'C' resulting in an oscillatory motion about point C which in turn is translated to point 'D through a crank shaft mechanism. The Solar panels are mounted on a base fixed to the shaft along 'D'. When in operation, point 'D' undergoes an angular displacement of over 90^0. This effectively covers the sun path.

Chapter four

CONSTRUCTION, TESTING AND RESULTS

4.1 Construction tools and materials

The tools and materials as well as instruments used during the testing and the construction of the project are briefly described below:

1. **The simulation:** The circuit diagram was tested on the computer using the Mult isim 2001 software for the simulation analysis of dc operating points, transients and parameter sweep.
2. **The breadboard:** This is a temporary board for circuit testing with tiny sockets that allows for electronic components (i.e. resistors, capacitors, ICs e.t.c) to be plugged in, easily without damaging the component. The breadboard was used for pre-construction testing of circuit and sub-circuits before the Components were soldered on the Vero board.
3. **Analogue / digital multimeter:** These devices (instruments) were used for the Measurement of electrical quantities such as resistance, voltage and current. There were also used to test the circuit sections for continuity.
4. **The Vero board:** This is a perforated board on which electronic components can be inserted and soldered permanently.It was used for the permanent

construction of the project prototype from the circuit diagram.

5. **Wires and connectors:** Wires were used during the testing stage of the project on the breadboard to connect the components together as well as the different sub-units of the circuit, they were also used during the soldering of components on the Vero board. Copper wire was used.
6. **IC Sockets:** This is a device used to hold ICs in position; the IC socket as first soldered on the Vero board before the IC chip was fixed on it to protect the IC from the heat of the soldering iron.
7. **Wire cutters / strippers:** These tools were used to cut wires to the desired size required size before use, as well as to strip off insulation wire in other to expose the conductor for proper and neat soldering.
8. **Soldering Iron:** This is a low power heating element typically in range of about 40 Watts. It provides the heat needed to melt the lead, so that it can be used for the connection of the components permanently on the Vero board. It is usually connected to the AC mains.
9. **Soldering lead:** This is a metal (lead) wire of low melting point. It is used to electrically connect components and wires in fixed position on the Vero board.
10. **Lead sucker:** This is used to suck up excess molten lead from 'the Vero board to prevent short circuit (bridging) or undesirable electrical connections.

4.2 Construction Details

The circuit was laid-out on the bread board to observe its operational response and ensure that it is in line with required objectives. Then it was dismantled.The circuit was finally constructed on the Vero board. The components were inserted into the holes on the board properly to ensure that it is out on the other side of the board where the copper tracks are. All components and connecting wires were inserted in place before soldering. The MCU chip is very sensitive to heat and so was protected by the use of IC socket. The socket was first soldered on the board before inserting the IC.

4.3 Construction Precaution

1. All soldered joints (points) were tested for continuity so as to avoid unnecessary open circuits.
2. All the excess leads were removed to avoid bridges (short circuits) on the boards.
3. Polarities of the electrolytic capacitors and LEDS were properly checked to be correctly positioned before connecting (soldering) on the Vero board.
4. ICs were mounted on 1C sockets to avoid overheating them during soldering by soldering the 1C socket first on the Vero board.
5. Excessive heating of the components was avoided so that they do not burn by making the soldering process to a component very brief.

4.4 Testing and results

The circuit was initially constructed on a breadboard and found to be working properly. It was then dismantled and the components transferred to the Vero board and the connections were made as appropriate. Continuity test was carried out with a multimeter.

The motor controller circuit was tested next to ensure that it could rotate in the clockwise, anticlockwise as well as stop positions with minimal noise by replicating the action of the microcontroller.After the whole system units (electrical and mechanical) had been coupled, the solar elevation tracker was then tested.The first test on the solar tracker was carried out with torchlight. When light was directed more on one side, the system was found to adjust to a position that balanced the light on both sensors. The Light was then transferred to the other side to ensure that the system could work both ways. Desired results were obtained with minimal hysteresis.When the light was shown directly, the system was relatively stable. The same tests were also carried using sunlight. The sensors were shaded individually and desired results obtained.

Chapter Five

CONCLUSION

From the testing and results obtained, it can be seen that the tracker satisfactorily tracks the sun during the preset periods, resets after, and "hibernates" during night time and resumes at a preset period. The system designed improves the output and thus efficiency of an Amorphous Silicon photovoltaic cell.

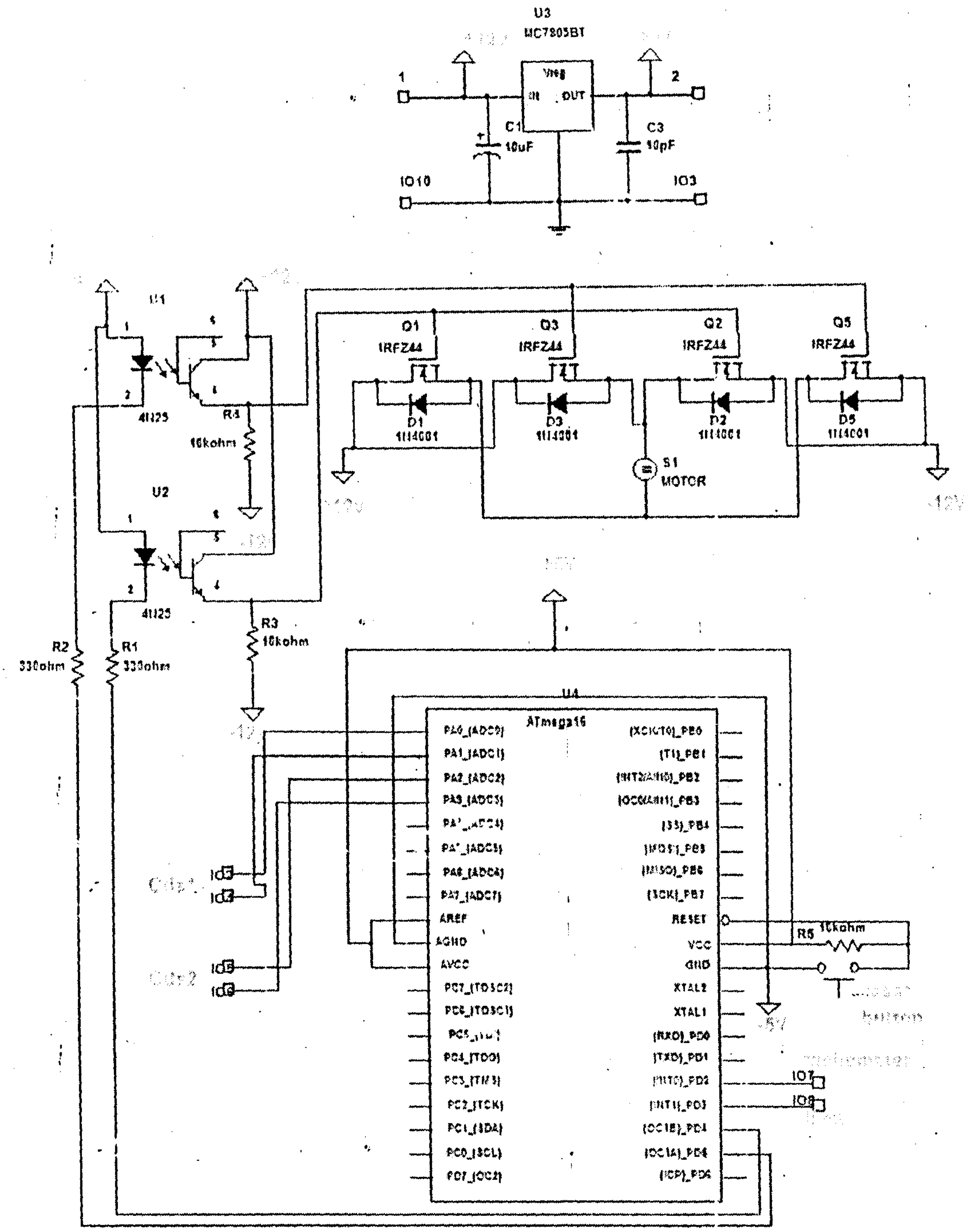
The tracker is an option when a higher cell output is required, rather than investing in the increase of more cells, which is by far more expensive.

RECOMMENDATION

In the design and construction of a Solar Elevation Tracker, the system satisfactorily tracked the elevation of the sun. However due to seasonal changes the system could be expanded to accommodate both elevation as well as lateral tracking. The system could also be modified to source its own power from the one generated. Then it would be completely self - driven.

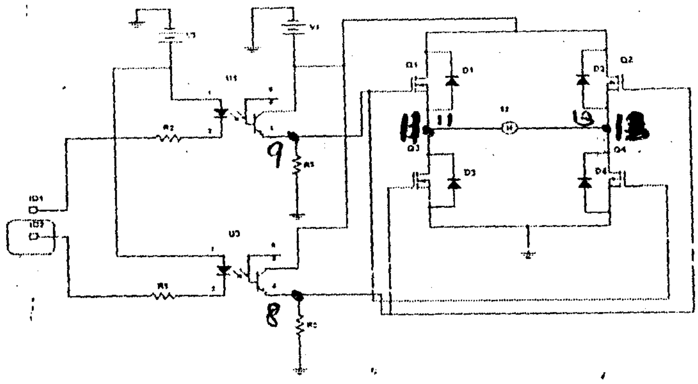
APPENDIX A

Complete circuit diagram



APPENDIX B

(A) Forward Mode

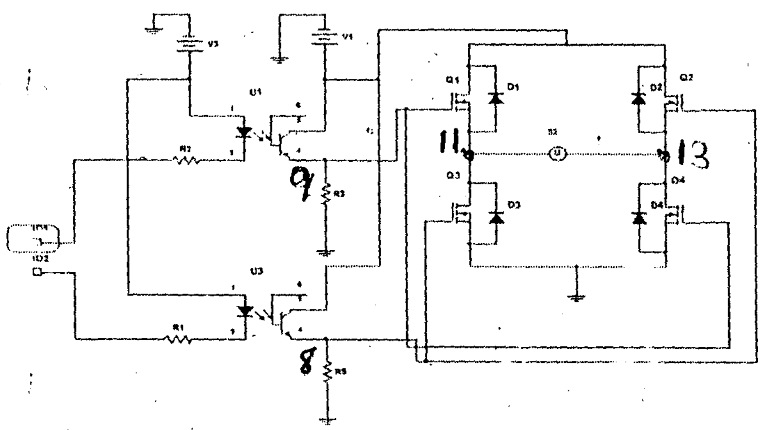


Optocoupled Motor Driver

Operating Point for FORWARD mode

DC Operating Point		
9	2.66674µ	
13	53.04879m	
11	6.85850	
8	10.77832	

(B) Reverse Mode



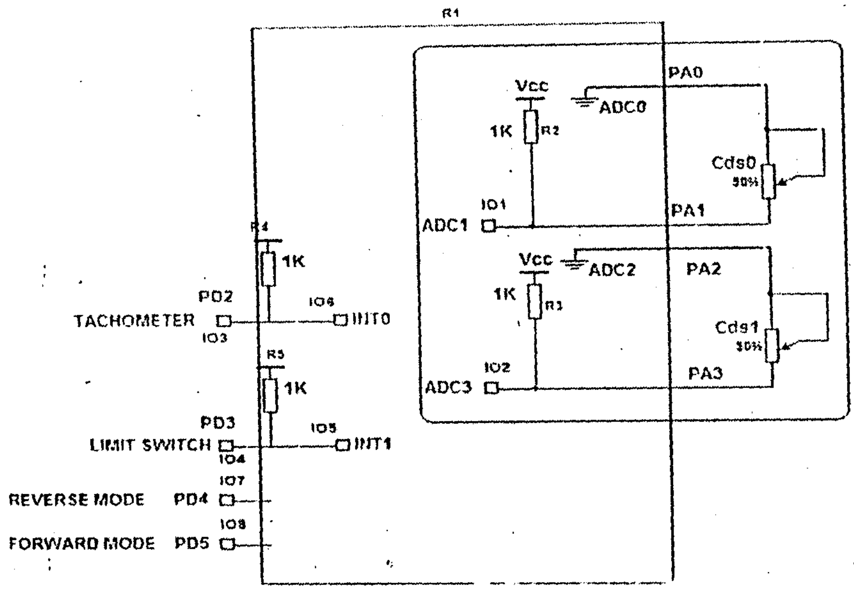
Optocoupled Motor driver

Operating Point

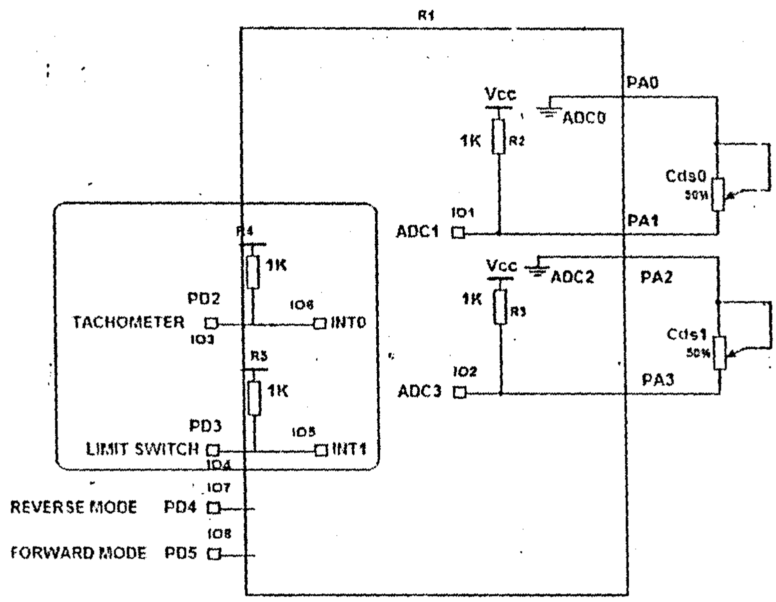
DC Operating Point		
13	6.85850	
11	53.04879m	
8	2.66674µ	
9	10.77832	

APPENDIX C

(A) Solar Intensity Detector (Cds)

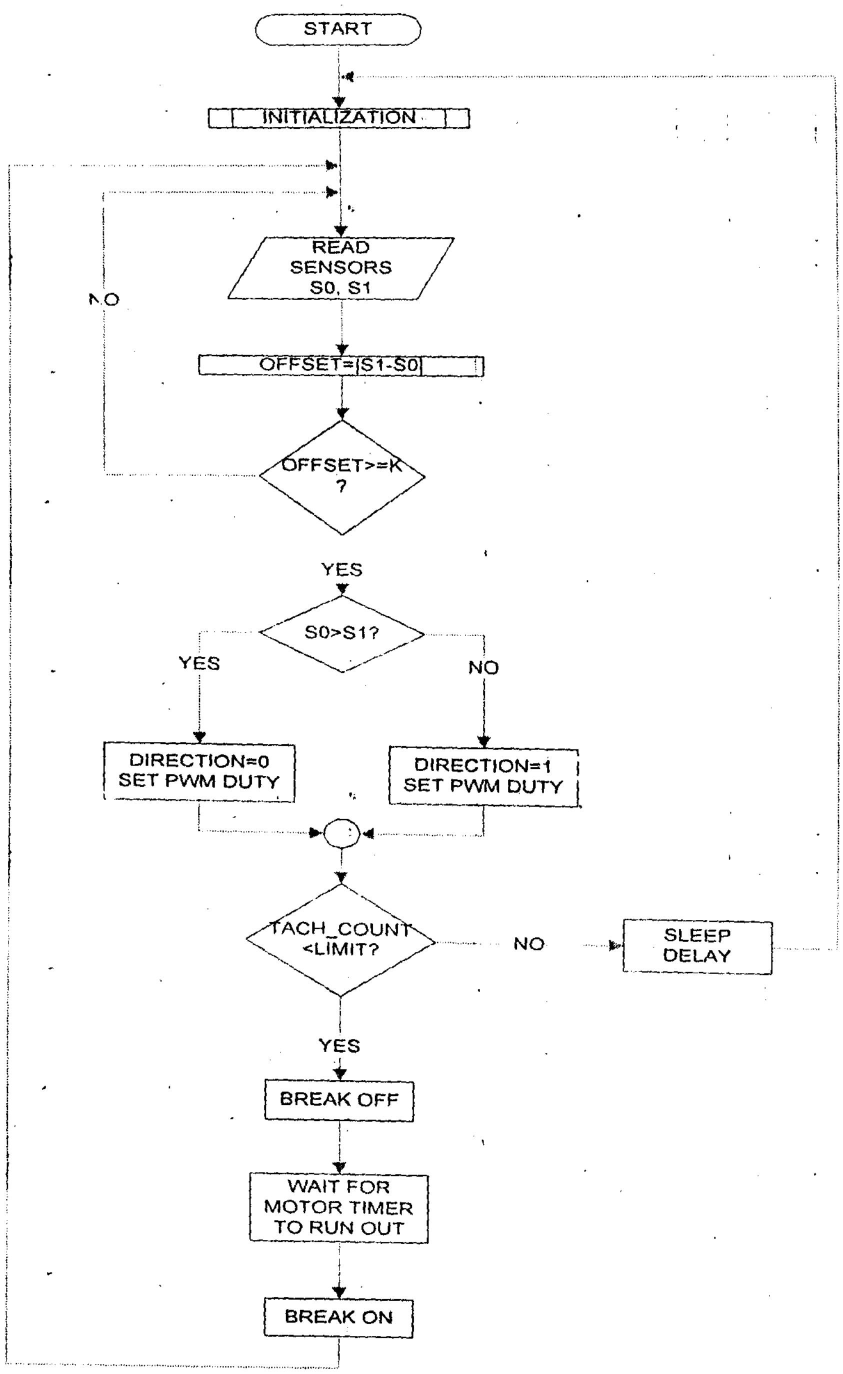


(B) Limit Sensors



APPENDIX D

Flow Chart



APPENDIX E

MCU Registers Summary

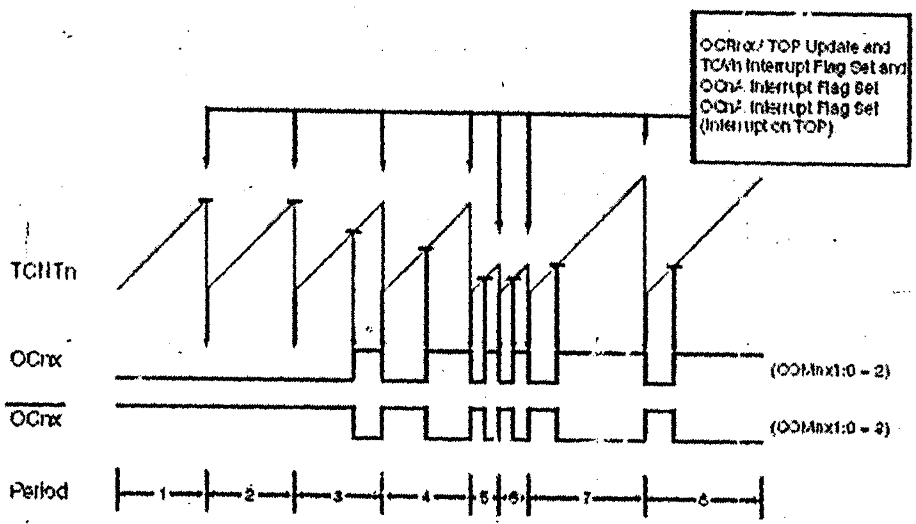
Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
$3F ($5F)	SREG	I	T	H	S	V	N	Z	C	7
$3E ($5E)	SPH	–	–		–	–	SP10	SP9	SP8	10
$3D ($5D)	SPL	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	10
$3C ($5C)	OCR0	Timer/Counter0 Output Compare Register								80
$3B ($5B)	GICR	INT1	INT0	INT2	–	–	–	IVSEL	IVCE	45, 65
$3A ($5A)	GIFR	INTF1	INTF0	INTF2	–	–	–	–	–	66
$39 ($59)	TIMSK	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0	80, 109, 125
$38 ($58)	TIFR	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	OCF0	TOV0	80, 110, 127
$37 ($57)	SPMCR	SPMIE	RWWSB	–	RWWSRE	BLBSET	PGWRT	PGERS	SPMEN	245
$36 ($56)	TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE	174
$35 ($55)	MCUCR	SM2	SE	SM1	SM0	ISC11	ISC10	ISC01	ISC00	30, 64
$34 ($54)	MCUCSR	JTD	ISC2	–	JTRF	WDRF	BORF	EXTRF	PORF	38, 65, 225
$33 ($53)	TCCR0	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00	77
$32 ($52)	TCNT0	Timer/Counter0 (8 Bits)								79
$31[1] ($51)[1]	OSCCAL	Oscillator Calibration Register								28
	OCDR	On-Chip Debug Register								221
$30 ($50)	SFIOR	ADTS2	ADTS1	ADTS0	ADHSM	ACME	PUD	PSR2	PSR10	54,82,128,195,215
$2F ($4F)	TCCR1A	COM1A1	COM1A0	COM1B1	COM1B0	FOC1A	FOC1B	WGM11	WGM10	104
$2E ($4E)	TCCR1B	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	107
$2D ($4D)	TCNT1H	Timer/Counter1 – Counter Register High Byte								108
$2C ($4C)	TCNT1L	Timer/Counter1 – Counter Register Low Byte								108
$2B ($4B)	OCR1AH	Timer/Counter1 – Output Compare Register A High Byte								108
$24 ($44)	OCR1AL	Timer/Counter1 – Output Compare Register A Low Byte								108
$29 ($49)	OCR1BH	Timer/Counter1 – Output Compare Register B High Byte								108
$28 ($48)	OCR1BL	Timer/Counter1 – Output Compare Register B Low Byte								108
$27 ($47)	ICR1H	Timer/Counter1 – Input Capture Register High Byte								109
$26 ($46)	ICR1L	Timer/Counter1 – Input Capture Register Low Byte								109
$25 ($45)	TCCR2	FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20	121
$24 ($44)	TCNT2	Timer/Counter2 (8 Bits)								123
$23 ($43)	OCR2	Timer/Counter2 Output Compare Register								123
$22 ($42)	ASSR	–	–	–	–	AS2	TCN2UB	OCR2UB	TCR2UB	124
$21 ($41)	WDTCR	–	–	–	WDTOE	WDE	WDP2	WDP1	WDP0	40
$20[2] ($40)[2]	UBRRH	URSEL	–	–	–	UBRR[11:8]				161
	UCSRC	URSEL	UMSEL	UPM1	UPM0	USBS	UCSZ1	UCSZ0	UCPOL	159
$1F ($3F)	EEARH	–	–	–	–	–	–	–	EEAR8	17
$1E ($3E)	EEARL	EEPROM Address Register Low Byte								17
$1D ($3D)	EEDR	EEPROM Data Register								17
$1C ($3C)	EECR	–	–	–	–	EERIE	EEMWE	EEWE	EERE	17
$1B ($3B)	PORTA	PORTA7	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	62
$1A ($3A)	DDRA	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	62
$19 ($39)	PINA	PINA7	PINA6	PINA5	PINA4	PINA3	PINA2	PINA1	PINA0	62
$18 ($38)	PORTB	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	62
$17 ($37)	DDRB	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	62
$16 ($36)	PINB	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	63
$15 ($35)	PORTC	PORTC7	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0	63
$14 ($34)	DDRC	DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0	63
$13 ($33)	PINC	PINC7	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0	63
$12 ($32)	PORTD	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	63
$11 ($31)	DDRD	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	63
$10 ($30)	PIND	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	63
$0F ($2F)	SPDR	SPI Data Register								135
$0E ($2E)	SPSR	SPIF	WCOL	–	–	–	–	–	SPI2X	134
$0D ($2D)	SPCR	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0	133
$0C ($2C)	UDR	USART I/O Data Register								156
$0B ($2B)	UCSRA	RXC	TXC	UDRE	FE	DOR	PE	U2X	MPCM	157
$0A ($2A)	UCSRB	RXCIE	TXCIE	UDRIE	RXEN	TXEN	UCSZ2	RXB8	TXB8	158
$09 ($29)	UBRRL	USART Baud Rate Register Low Byte								161
$08 ($28)	ACSR	ACD	ACBG	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0	195
$07 ($27)	ADMUX	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0	211
$06 ($26)	ADCSRA	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	213
$05 ($25)	ADCH	ADC Data Register High Byte								214
$04 ($24)	ADCL	ADC Data Register Low Byte								214
$03 ($23)	TWDR	Two-wire Serial Interface Data Register								176

APPENDIX F

Waveform Generation Mode Bit Description

Mode	WGM13	WGM12 (CTC1)	WGM11 (PWM11)	WGM10 (PWM10)	Timer/Counter Mode of Operation	TOP	Update of OCR1X	TOV1 Flag Set on
0	0	0	0	0	Normal	0xFFFF	Immediate	MAX
1	0	0	0	1	PWM, Phase Correct, 8-bit	0x00FF	TOP	BOTTOM
2	0	0	1	0	PWM, Phase Correct, 9-bit	0x01FF	TOP	BOTTOM
3	0	0	1	1	PWM, Phase Correct, 10-bit	0x03FF	TOP	BOTTOM
4	0	1	0	0	CTC	OCR1A	Immediate	MAX
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	TOP	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	TOP	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	TOP	TOP
8	1	0	0	0	PWM, Phase and Frequency Correct	ICR1	BOTTOM	BOTTOM
9	1	0	0	1	PWM, Phase and Frequency Correct	OCR1A	BOTTOM	BOTTOM
10	1	0	1	0	PWM, Phase Correct	ICR1	TOP	BOTTOM
11	1	0	1	1	PWM, Phase Correct	OCR1A	TOP	BOTTOM
12	1	1	0	0	CTC	ICR1	Immediate	MAX
13	1	1	0	1	Reserved	–	–	–
14	1	1	1	0	Fast PWM	ICR1	TOP	TOP
15	1	1	1	1	Fast PWM	OCR1A	TOP	TOP

Fast PWM timing diagram



APPENDIX G

```
#include <stdio.h>
#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/signal.h>
#include <avr/pgmspace.h>
#include <avr/sleep.h>

#include <ctype.h>
#include <inttypes.h>
#include <string.h>

#define BAUDRATE        9600
#define BAUD_REG        ((uint16_t)((F_CPU / (16.0 * (BAUDRATE))) + 0.5) - 1)
        // if above .5 mark, round up; replace 16 with 8 for double
#define BAUD_H    ((uint8_t)(0xFF&(BAUD_REG>>8)))
#define BAUD_L          ((uint8_t)(0xFF&BAUD_REG))

#define init_motor_period 300           // initial PWM signal period
#define init_motor_duty 150                     // initial PWM duty-cycle period
#define     MOTOR_LIMIT_H 2500
#define     MOTOR_LIMIT_L 100

#define     SID_PORT   PORTA
```

```
#define     SID_PIN       PINA
#define     SID_DDR       DDRA

#define     SW_PORT       PORTB
#define     SW_PIN        PINB
#define     SW_DDR        DDRB

#define     LED_PORT  PORTC
#define     LED_PIN       PINC
#define     LED_DDR       DDRC


int               tach;
uint8_t           limit, direction;    // 1 = left, 0 = right
int         magnitude;
uint8_t           SID[2];


/*** delay ***************************************************************
*******
*     rough delay; 65k loops, 4 instr each, +over head: 65536*4 = 262144,
*     round up tp 300000 clks (time: 300000/F_CPU seconds)
**********************************************************************
********/
void delay(void) {
uint8_t i, j;
for(i=0; i<255; i++) {
```

```
for(j=0; j<255; j++) {
asm volatile("nop"::);
}
}
}

// Timer 0 Interrupt triggered by Overflow
SIGNAL(SIG_OVERFLOW0) {
if (motor_timer > 0) { motor_timer--; }
}
/*** SIG_INTERRUPT0 *****************************************************
**********
 *      tachometer
 *****************************************************************************
********/
SIGNAL (SIG_INTERRUPT0) {
// a hole has passed on the encoder; increment or decrement the counter
//      based on the direction
if(direction == 0)
tach--;
else if(direction == 1)
tach++;

if(tach >= MOTOR_LIMIT_H)
limit=1;
else if(tach <= MOTOR_LIMIT_L)
limit=0;
```

```
else
limit=2;
}
/*** SIG_INTERRUPT1 ****************************************************
**********
 *   limit
**********************************************************************
********/
SIGNAL (SIG_INTERRUPT1) {
if(limit == 2)
limit = 0;
else
limit = 2;
tach = 0;
}
/*** getSID ************************************************************
*******
 *   read current position of panels solar intensity detector
**********************************************************************
********/
void getSID(void) {
ADCSRA |= _BV(ADEN);                 // enable ADC



SID[0] = 0;
ADMUX |= _BV(MUX0);                  // select PA1 as input for ADC
ADMUX &= ~_BV(MUX1);                 //      "
```

```
ADCSRA |= _BV(ADSC);
        // start conversion

loop_until_bit_is_clear(ADCSRA, ADSC);
        // wait for conversion to finish


SID[0] += ADCH;


SID[1] = 0;

ADMUX |= _BV(MUX1) | _BV(MUX0);
        // select PA3 as input for ADC


ADCSRA |= _BV(ADSC);
        // start conversion

loop_until_bit_is_clear(ADCSRA, ADSC);
        // wait for conversion to finish


SID[1] += ADCH;


ADCSRA &= ~_BV(ADEN);            // disable ADC



// excess

if(SID[0] > SID[1]) {

if(SID[1] == 0)

SID[1]++;

magnitude = SID[0] / SID[1];

direction = 0;
```

```
}
else if(SID[1] > SID[0]) {
if(SID[0] == 0)
SID[0]++;
magnitude = SID[1] / SID[0];
direction = 1;
}
else {
magnitude = 0;


}
}
/*** setSpin ************************************************************
*******
 *       set speed of motor (PWM)
 *****************************************************************************
********/
void setSpin(void) {


if(magnitude < 2) {   // replace # with a hysterisis value
motor_break();
else if((direction == 1) && (limit!=1)) {
PORTD |= _BV(PD4);
PORTD &= ~_BV(PD5);
motor_controller(magnitude);
}
else if((direction == 0) && (limit!=0)) {
```

```
PORTD |= _BV(PD5);
PORTD &= ~_BV(PD4);
motor_controller(magnitude);
}
else
PORTD |= _BV(PD4) | _BV(PD5);
}
}



/*
 * MOTOR FUNCTION: motor_controller(char vel, char dir)
 */
void motor_controller(int magnitude) {

// SPEED
motor_PWM_duty = magnitude*motor_PWM_period/100;
motor_updateDuty();

// TIME
motor_timer = 2; // 0.008 sec interval, 122Hz

// Wait until timer runs out
while (motor_timer > 0) {}
```

```
// break ON
motor_break();
}
void motor_break() {

PORTD |= _BV(PD4) | _BV(PD5);
}



/*
 * MOTOR FUNCTION: motor_updateDuty()
 */
void motor_updateDuty() {
char lowByte;
char highByte;

lowByte = (char)(motor_PWM_duty);
highByte = (char)(motor_PWM_duty >> 8);

OCR1AH = highByte;
OCR1AL = lowByte;
}

/*
 * MOTOR FUNCTION: motor_updatePeriod()
 */
```

```
}

void motor_updatePeriod() {

char lowByte;

char highByte;

lowByte = (char)(motor_PWM_period);

highByte = (char)(motor_PWM_period >> 8);

ICR1H = highByte;

ICR1L = lowByte;

}

/*** SIG_UART_RECV ***************************************************
*
*       interrupt on receive byte; for now just echo if any char received
*********************************************************************************/

SIGNAL (SIG_UART_RECV) {

uint8_t temp;

temp = UDR;                    // read

UDR = temp;              // write (just echo what the user types)

}

/*** uart_tx ***************************************************************
*
*       Transmit given data via UART
*********************************************************************************/

uint8_t uart_tx(uint8_t uart_tx) {
```

```
while(!(UCSRA & _BV(UDRE)));        // wait for empty tx buffer


UDR = uart_tx;                                    // put data in buffer, init send


return 0;

}

/*** uart_rx ***************************************************************
******

 *      Transmit given data via UART

 **************************************************************************
********/

uint8_t uart_rx(void) {

while(!(UCSRA & _BV(RXC)));         // wait for full rx buffer


return UDR;                                   /* return the new c */

}


/*** init ******************************************************************
******

 *      init all vars and ports

 **************************************************************************
********/

void init(void) {

SID_DDR             = 0b00000101;
        // sensors (1,3) are inputs, sensor grounds (0,2) outputs

SID_PORT    = 0b00001010;       // pull-ups active (1+3), grounding pins low


/*
```

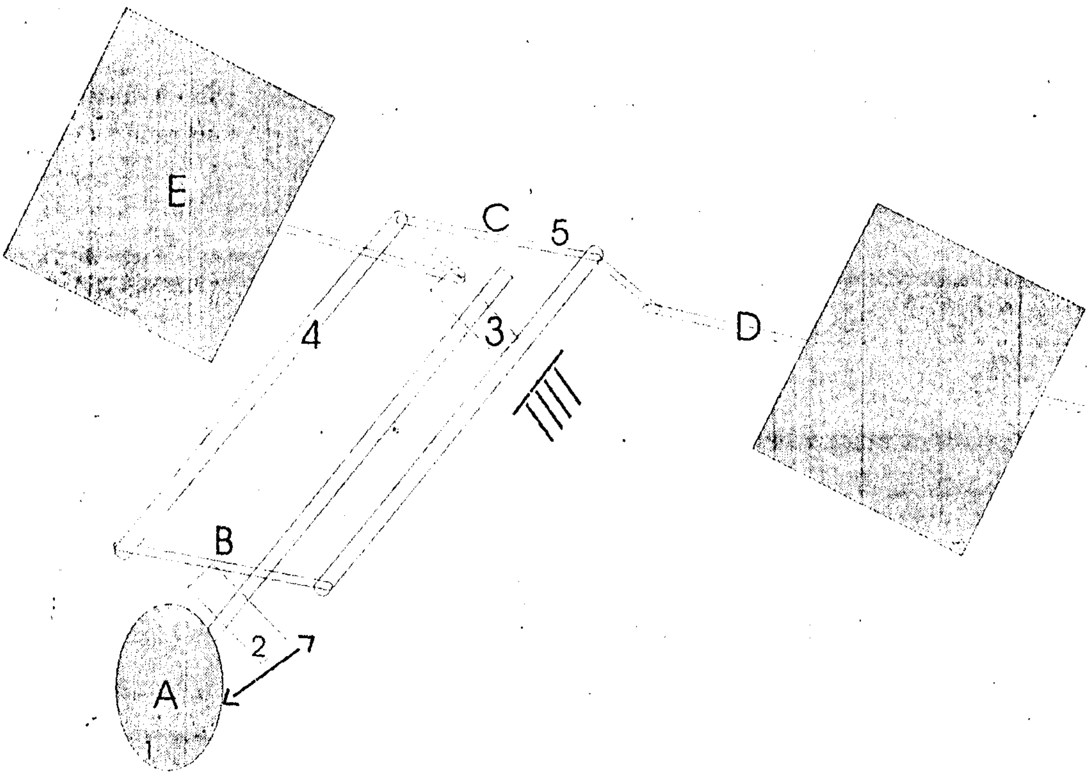
```
PORTD               pullups     direction
0 i    RXD          0           0
1 o    TXD          0           1
2 i    INT0   1           0                 opto-interrupter (rotations)
3 i    INT1   1           0                 opto-interrupter (limit)
4 o    OC1B 0             1                 opto-coupler0 (spin 0)
5 o    OC1A 0             1                 opto-coupler1 (spin 1)
6 i    ICP          1           0
7 i    OC2          1           0
*/

PORTD = 0b11001100;
DDRD = 0b00110010;
TCCR0 = 0b00000100;
TCCR1A = 0b10000010;
TCCR1B = 0b00011100;
TIMSK = 0b00000001;

// set baud rate: UBRR = (F_CPU/(16*BAUDRATE)) - 1
UBRRH = BAUD_H;
UBRRL = BAUD_L;
UCSRB = _BV(RXEN) | _BV(TXEN) | _BV(RXCIE);  // enable tx, rx //and rx int

// setup ext. interrupts 0 and 1
MCUCR |= _BV(ISC11) | _BV(ISC01);   // falling edge
GICR |= _BV(INT1);                          // enable ext int 1
```

```
// setup ADC
ADCSRA = _BV(ADPS2) | _BV(ADPS1);        // set ADC clock source division (64)
ADMUX = _BV(ADLAR);                                // ADC -> ext ref voltage

}

sei();

// Global Variables
motor_PWM_period = init_motor_period;
motor_updatePeriod();
motor_PWM_duty = init_motor_duty;
motor_updateDuty();
motor_break();
motor_timer = 0;

limit = 2;
direction = 0;
magnitude = 5;
setSpin();
while(limit != 0) ;
setSpin();

GICR &= ~_BV(INT1);            // disable ext int 1 (for calibration only)
GICR |= _BV(INT0);                        // enable ext int 0 (tach)

}
```

```
limit = 2;
direction = 1;
magnitude = 5;
tach = 0;
setSpin();
while(tach < MOTOR_LIMIT_L) ;
}
int main(void) {
init();
fdevopen(uart_tx, uart_rx, 0);
printf_P(PSTR("\n\r\n\r\n\rHello World!\n\r\n\r"));
delay();
// loop forever
   for(;;) {
getSID();
setSpin();
}
```

APPENDIX H

Structure of Mechanical support:



1. Motor
2. Gear
3. Worm
4. Links
5. Crank & Shaft

References

The following sources are research papers on the topic of photovoltaic energy systems, MPP tracking and solar tracking which directly relate to our project.

[1] Robert Carlson, Daniel Felnhofer, Paul Rondeau, "Design and Construction of a Maximum Power Tracking System for a Solar Panel". Final year Project, Faculty of Engineering of the University of Manitoba.

[2] Abdulkadir Alkali, "Time base (Constant velocity) Solar Tracker Using Stepper Motor". Final year Project, Department of Electrical Engineering, School of Engineering, A.T.B.U. Bauchi.

[3] Charles Smith, "Revisiting Solar Power's Past". Retrieved March 5, 2005 from *http://www.solarenergy.com/info-history.html.*

[4] Sode-Shinni Nmadu Rumala, " Solar Tracking by the Use of Back to Back Semi-Cylindrical Shades", Solar Energy Journal, Vol. 37, No. 3, 1986, pp. 245-247.

[5] Massachusetts Institute Of Technology,"Introduction to Electronics-Spring 2002". Retrieved March 8, 2005 from http://web.mit.edu/6.071/www/lecture24.pdf

[6] Kasa, N., T. Iida, and G. Majumdar, 2002. "Robust control for maximum power point tracking in photovoltaic power system". Proceedings of the Power Conversion Conference - Osaka, Japan 2002. Vol.2: 827-832.

[7] Bishop, Robert H., Dorf, Richard C. (2001). "Modern Control Systems Ninth Edition". Upper Saddle River, New Jersey, Prentice-Hall Inc.

[8] Chen, Chi-Tsong (1999)."Linear System Theory and Design Third Edition". New York, New York, Oxford University Press.

[9] Carnegie Mellon University,"DC Motor Control Systems". Retrieved April 2, 2005 from http://www.engin.umich.edu/group/ctm/PID/PID.html.

[10] Harakawa, T. and T. Tujimoto, 2001. "A proposal of efficiency improvement with solar power generation system". The 27th Annual Conference of the IEEE Industrial Electronics Society, 2001 (IECON '01). Vol.1: 523-528.

[11] NISCA," DC Motor Datasheets ". Retrieved April 24, 2005. From http://www.nisca.co.jp/pdf/dc_specs.pdf.