Automatic Miniblinds System

San Jose State University

Department of Mechanical and Aerospace Engineering

Brian Silva
Jonathan Thein

May 16, 2006

Summary:

The Mechatronics final term project is intended to be an accumulation of the knowledge gained over the course of the semester. Our team of engineers strived to implement a large range of concepts into a simple, practical design. The particular design which peaked our interest was that of an automatic miniblinds system. As seen in Figure 1, our team constructed a miniblinds system that automatically adjusts itself with respect to how much sunlight is present at the face of the window.

Figure 1: Automatic Miniblinds System

Our team started this project by selecting an appropriate stepper motor and driver. The required torque is fairly minimal, therefore allowing us to use a common stepper motor which was provided by our Mechatronics lab instructor. The next step of the project was to develop a method of coupling the motor to the miniblinds. Our initial design called for the motor to be placed inside of the metal housing of the miniblinds. We planned on coupling the motor straight to the internal shaft of the blinds, however the torque needed to rotate this particular shaft was much greater than that needed to rotate the plastic hanging shaft. This is due to the mechanical gears integrated in the miniblinds themselves. Once we discovered the relative difference in torque, we decided to couple the motor to the hanging plastic shaft as seen above. With the design concept in hand, our team then constructed a wooden housing to support and present our automatic miniblinds system.

The next step in implementing our design was to create a circuit which would detect the intensity of sunlight present at the face of the window. Our team used a method similar to the photoresistor lab which we had previously completed. We then integrated this circuit to our microcontroller which processes the information and sends out an appropriate signal to our motor driver. Another simple circuit consisting of transistors and resistors was designed to integrate the microcontroller to the driver. A closer view of the coupling of our system to a common miniblinds system can be seen in Figure 2.

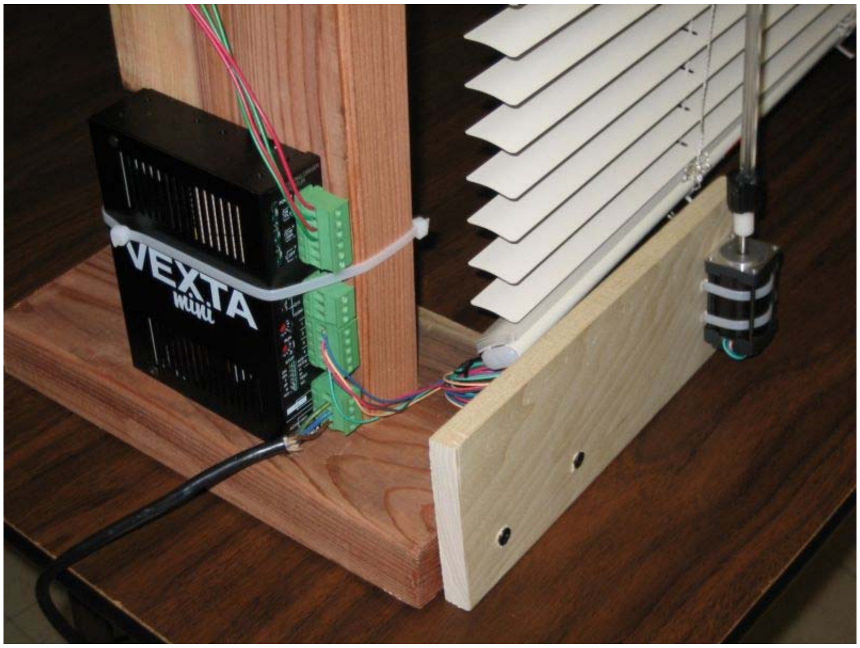


Figure 2: Integration of miniblinds to an automated system

Introduction:

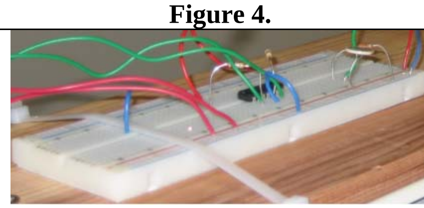
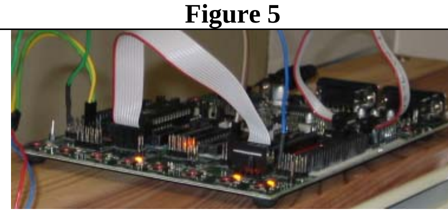
The basic concept of our automated system is that the position of the blinds is dependent on the intensity of light. A microcontroller receives the level of intensity from a photoresistor and sends an appropriate signal to the motor driver, which then steers the stepper motor to one of three desired states.

The first state is referred to as our ground state. The ground state is achieved when ambient light is present at the face of the window. At this particular time, the miniblinds are turned to a fully open position as seen in figure 1. The second state is referred to as our bright state. This takes place when sunlight is shinning directly on the window. Once the intensity of our sensor rises past a predetermined value, the blinds turn downward to a quarter-open state. The final state of our system takes place when the intensity of light drops below a certain value. This particular value is modeled to simulate darkness, and therefore the blinds rotate upward to a fully closed position. This state is referred to as the night state. The levels of intensity were determined experimentally and then slightly recalibrated so that the light in the room in which our team would be presenting would simulate ambient sunlight.

The motivation that drove our team of engineers to complete this project was that this practical design can actually be implemented and used as a tool to indirectly reduce power consumption in a household. With electricity rates in California being priced at a premium, effectively minimizing the use of lighting as well as air-conditioning units can make a huge impact on ones electricity bill. The system that we have designed maximizes natural lighting, which in turn reduces the need for turning on lights during daylight hours. Our system also adjusts the position of the miniblinds to block out light when the intensity is high. This can drastically cool a temporarily unattended room which might otherwise require the use of an air-conditioner or fan to reach a comfortable level.

DESCRIPTION OF PROJECT:

The project consists of five major subsystems: the microcontroller, motor driver, motor, photo-resistor sensor circuit, and the mini-blind.

Stepper Motor with an attached coupler connecting shaft of the motor with the shaft of the mini-blinds.	 **Figure 3.**
Driver for the Stepper Motor	 **Figure 4.**
Circuit used for photo-resistance reading	 **Figure 5**
Microcontroller Atmega16L on programmer's development board STK 500	 **Figure 6.**

Mini-Blind System: Mini-blinds mounted on a wooden frame	 **Figure 7.**

The following is a block diagram representation of how all of the above parts are interacting with each other.

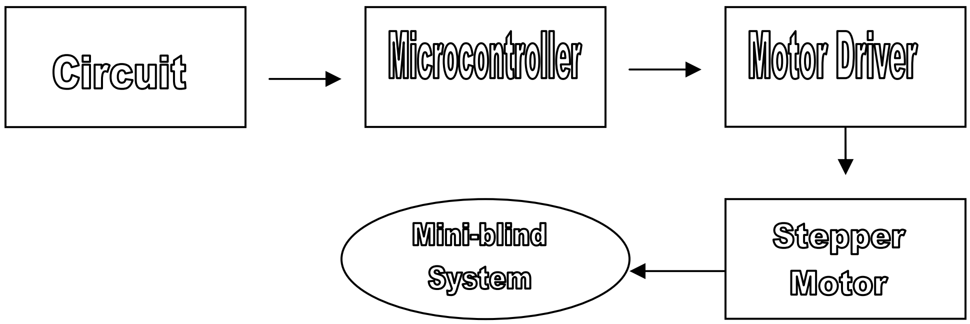


Figure 8. Outline of the major subsystems and how they are integrated to work with each other.

Circuit:

The overall circuit was constructed and looks similar to that of the following figure.

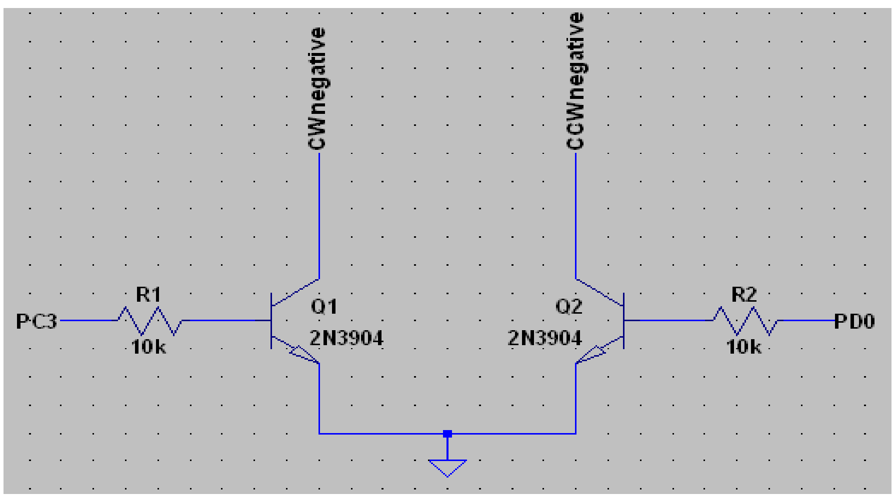


Figure 9. Circuit constructed to control the stepper motor with pins labeled for the microcontroller.

Along with the circuit includes the sensor portion which is ultimately the photo-resistor. The circuit is represented below:

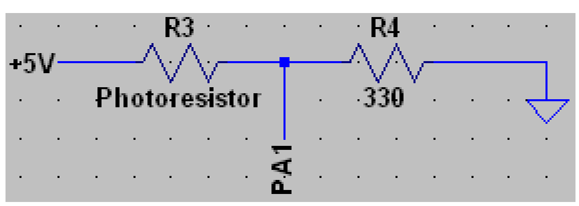
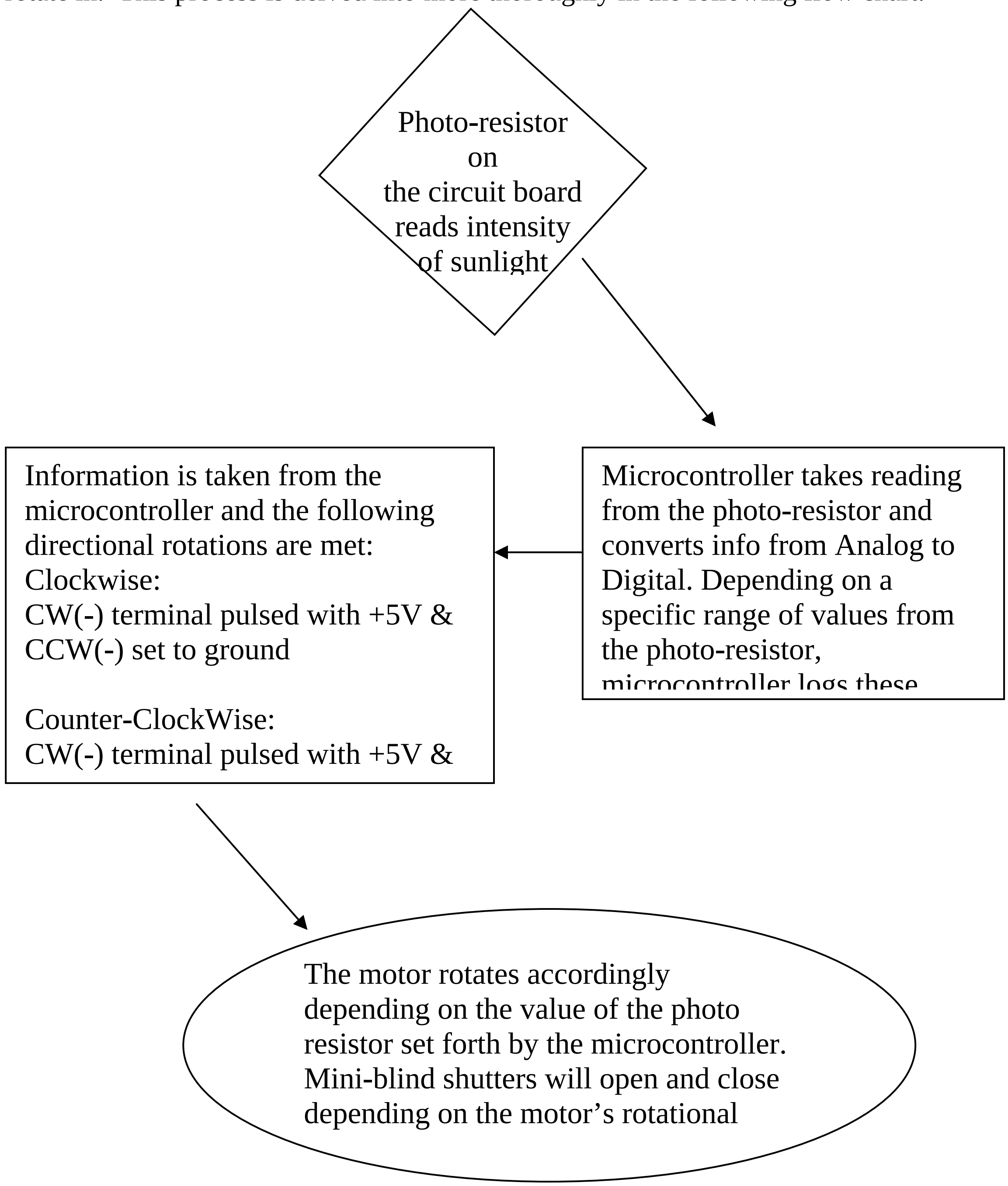


Figure 10. Photo-resistor sensor system to detect intensity of lighting indoors.

The circuit sends back to the microcontroller analog data which can allow either pin PC3 to be pulsed while PD0 is grounded, or PC3 to be pulsed while PD0 is active high. This determines which direction the driver motor will direct the stepper motor to rotate in. This process is delved into more thoroughly in the following flow chart.



CONCLUSION:

The overall outcome of our project was very satisfying because it worked. We initially set our goals to have the operation of the mini-blinds to take two states: Ambient lighting and Dark lighting. When the room is consumed with ambient lighting, the microcontroller would rotate clockwise to the fully open position; in the presence of dark lighting, the microcontroller would rotate the blinds counterclockwise. These tasks proved to be very simple so we extended the challenge to program the microcontroller to take on one additional state: Bright lighting (when there is direct sunlight). This task was very difficult because we had to appropriately time how many rotations between each state it would take for the microcontroller to rotate from one state to the next. The torque required to rotate the mini-blinds was negligible since having the shaft spin freely resulted in approximately 20 complete rotations. Our assumption of the stepper motor being able to provide a sufficient amount of torque was correct.

There were several daunting tasks that consumed a lot of our time, such as finding the appropriate coupler to connect the shaft of the stepper motor with the shaft of the mini-blind. If we had a little more time, we would have integrated the stepper motor into the rail of the mini-blinds so that the shafts would not need to be coupled together as shown in the previous pictures. The original gear box on the rail of the blinds was an inconvenience; due to time constraints, our group was not able to dissect it in time to utilize it. From this project, we learned how to operate the stepper motor and further our ability to program a microcontroller using the Atmega16L. We were unable to use the Atmega128 microcontroller due to firmware issues which prevented us to be able to synchronize it with the computers in the lab or with our personal desktops.

For future improvements, we would like to have had the motor directly integrated into the main frame of the mini-blinds in order to eliminate the use of the mini-blind’s shaft. Another interesting improvement that future groups may want to look into would be to operate the mini-blinds wirelessly through a handheld device.

APPENDIX:

Code used to program the Atmega16L:

```
/*****************************************************
This program was produced by the
CodeWizardAVR V1.24.6 Standard
Automatic Program Generator
© Copyright 1998-2005 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com
e-mail:office@hpinfotech.com

Project : Automatic MiniBlind System
Date    : 5/16/2006
Author  : Jonathan Thein & Brian Silva


Chip type           : ATmega16L
Program type        : Application
Clock frequency     : 3.680000 MHz
Memory model        : Small
External SRAM size  : 0
Data Stack size     : 256
*****************************************************/

#include <mega16.h>
#include <delay.h>

#define ADC_VREF_TYPE 0xE0

// Read the 8 most significant bits
// of the AD conversion result
unsigned char read_adc(unsigned char adc_input)
{
ADMUX=adc_input|ADC_VREF_TYPE;
// Start the AD conversion
ADCSRA|=0x40;
// Wait for the AD conversion to complete
while ((ADCSRA & 0x10)==0);
ADCSRA|=0x10;
return ADCH;
}

// Declare your global variables here
int DIR = 0;
int OPEN = 1;
void main(void)
{
```

```
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTA=0x00;
DDRA=0x00;

// Port B initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out
// State7=1 State6=1 State5=1 State4=1 State3=1 State2=1 State1=1 State0=1
PORTB=0x00;
DDRB=0x08;

// Port C initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTC=0x00;
DDRC=0x01;

// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTD=0xFF;
DDRD=0xFF;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: 3.594 kHz
// Mode: Normal top=FFh
// OC0 output: Toggle on compare match
//TCCR0=0x15;     //Prescaler 1024
//TCCR0=0x14;     //Prescaler 256
TCCR0=0x13;     //Prescaler 64
//TCCR0=0x12;     //Prescaler 8
//TCCR0=0x11;     //Prescaler no prescale
//TCCR0=0x00;
TCNT0=0x00;
OCR0=0xFF;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer 1 Stopped
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
```

```
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer 2 Stopped
// Mode: Normal top=FFh
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=0x00;
MCUCSR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// ADC initialization
// ADC Clock frequency: 115.000 kHz
// ADC Voltage Reference: Int., cap. on AREF
// ADC Auto Trigger Source: None
// Only the 8 most significant bits of
// the AD conversion result are used
ADMUX=ADC_VREF_TYPE;
ADCSRA=0x85;

while (1)
    {

    unsigned char adcpin;
    unsigned char reading;

    /*************************************
      ADC Conversion for Photo Resistor
    *************************************/
    adcpin = PORTA & 0x01;       //PIN1 from port A
    reading = read_adc(adcpin);  //Conversion to Digital
    PORTD = reading;             //Output to LEDs
```

```
if (reading <= 0xFF && reading >= 0xD0) //Dark
{
   PORTC &= ~0x01;    //COUNTER CLOCKWISE
   if (OPEN == 1)     //If currently open
   {
        //spin CCW for certain amount
        OPEN = 0;      //blinds are not open
        TCCR0=0x13;       //Send pulse to Stepper driver
        delay_ms(15000);     //Let stepper spin for 15 seconds
        TCCR0=0x00;          //Stop pulse
        DIR = 2;             //Direction set to 2 for OPEN condition

   }
   else //If not open, then dont change
   {
       TCCR0=0x00;   //ignore (leave blinds open)
   }
}

else if (reading <= 0xD1 && reading >= 0xB0 )  //Ambient Light
{
    //OPEN Blinds
    if (DIR == 2)
    {
        //CW
        PORTC |= 0x01;     //CLOCKWISE
        TCCR0=0x13;        //Send Pulse to stepper driver
        delay_ms(18000);   //Let stepper spin for 18 seconds
        TCCR0=0x00;        //Stop pulse
        DIR = 0;        //Set Direction to 0 for OPEN condition
        OPEN = 1;      //blinds are open

    }
    else if(DIR == 1)
    {
        //CCW
        PORTC &= ~0x01;     //COUNTER CLOCKWISE
        TCCR0=0x13;         //Send pulse to stepper driver
        delay_ms(18000);    //Let stepper spin for 18 seconds
        TCCR0=0x00;         //Stop Pulse
        DIR = 0;         //Set Direction to 0 for OPEN condition
        OPEN = 1;      //blinds are open

    }
    else //ALREADY OPEN
    {
        OPEN = 1;      //Reset to open
        TCCR0=0x00;    //ignore (leave Blinds open)
    }
}

else if (reading <= 0xAF && reading >= 0x00)  //Bright Light
{
     PORTC |= 0x01;   //CLOCKWISE
     if (OPEN == 1)     //If Blinds are open
```

```
    {
      //spin CW for certain amount
      OPEN = 0;        //Blinds not open
      TCCR0=0x13;      //Send pulse to stepper driver
      delay_ms(15000); //Let stepper spin for 15 seconds
      TCCR0=0x00;      //Stop Pulse
      DIR = 1;         //Set direction to 1 for OPEN condition


    }
    else //Blinds are closed
    {
       TCCR0=0x00;   //ignore (leave blinds closed)
    }

  }
  }
```

References:

Furman B. J., "ME 120 Laboratory Report Guidelines" (2006). Retrieved May 10, 2006 from: http://www.engr.sjsu.edu/bjfurman/courses/ME120/me120pdf/ME120labreportguide.pdf

Furman B. J., "ME Photoresistor, LED, and Transistor Lab" (2006). Retrieved April 20, 2006 from: http://www.engr.sjsu.edu/bjfurman/courses/ME106/ME106pdf/photoresistor-atmel.pdf

Vexta Stepper Drivers Manuel (2006). Retrieved April 25, 2006 from: http://www.vexta.com