Digital Programmable Power Supply

John Mashurano[1]

[1]Tianjin University of Technology and Education, School of Electronics Engineering
Tianjin, China 300222
mashurano@aol.com

Abstract: ***These days, majority of electronic devices work on DC power source, so there is a requirement of a reliable and customized power supply. Generally, the requirements are not too varied, but still they require every time a new hardware designing. The idea presented here is to build a Digital Programmable power supply that is flexible enough to meet different customer requirements, with minor software changes and no corresponding hardware change. Hardware issues are discussed, with a goal of developing a generalized power supply that has programmable output voltage and current. It is able to recognize faults and take corrective actions to prevent any permanent damage to the system. The system discussed here is capable of functioning independently by its own without any intervention from the user. The system finds application at remote sites to automatically manage primary (AC) and secondary (Battery) power sources to provide smooth uninterrupted power output even during switchovers between AC and DC power sources. It would also be helpful to insurance companies who expect that the products covered by them be reliable, robust and not prone to be damages.***

Keywords: Embedded system, AT98S52 microcontroller, Compiler, LCD.

1. Introduction

The trend and demand these days are to develop systems that are reliable and intelligent. People need products that are reliable and do not require constant supervision. From the manufacturer point of view, they require a product that is highly reliable and does not damage if the parameters run out of specifications or even on mishandling by the user.

To increase the market base of their products the manufacturer tries to put enough intelligence in the system so that it is capable of functioning by itself. Embedded system's is one such area, which fits well in these situations. The microcontroller based power SUPPLY is developed with a view to make a flexible and an intelligent power supply controlled by a microcontroller of 8051 family and to eliminate the drawbacks or limitations of the existing systems. They are discussed most of the existing systems are hardwired and expensive, which increases the production and procurement cost. The PCBs are larger and complex because of low component integration. Due to hardwired logic the flexibility is low and a particular PCB circumvents to the specified requirements only there is no room for change, as any modification in specification requires redesigning. The system parameters such as battery charging voltage, current are not programmable. They do not have the ability to generate logs or have periodic automatic self-testing techniques. User interface is not that interactive or informative.

2. Embedded system

Embedded System is a combination of hardware and software used to achieve a single specific task. An embedded system is a microcontroller-based, software driven, reliable, real-time control system, autonomous, or human or network interactive, operating on diverse physical variables and in diverse environments and sold into a competitive and cost conscious market An embedded system is a special-purpose computer system designed to perform a dedicated function. Unlike a general-purpose computer, such as a personal computer, an embedded system performs one or a few pre-defined tasks, usually with very specific requirements, and often includes task-specific hardware and mechanical parts not usually found in a general-purpose computer. Since the system is dedicated to specific tasks, design engineers can optimize it, reducing the size and cost of the product. Embedded systems are often mass-produced, benefiting from economies of scale. Physically, embedded systems range from portable devices such as digital watches and MP3 players, to large stationary installations like traffic lights, factory controllers, or the systems controlling nuclear power plants. In terms of complexity embedded systems run from simple, with a single microcontroller chip, to very complex with multiple units, peripherals and networks mounted inside a large chassis or enclosure Mobile phones or handheld computers share some elements with embedded systems, such as the operating systems and microprocessors which power them, but are not truly embedded systems themselves because they tend to be more general purpose, allowing different applications to be loaded and peripherals to be connected.

Examples of Embedded Systems

An embedded system typically has a specialized function with programs stored on ROM. Examples of embedded systems are chips that monitor automobile functions, including engine controls, antilock brakes, air bags, active suspension systems, environmental systems, security systems, and entertainment systems. Everything needed for those functions is custom designed into specific chips. No external operating system is required

3. Characteristics of Embedded System

- An embedded system is any computer system hidden inside a product other than a computer
- There will encounter a number of difficulties when writing embedded system software in addition to those we encounter when we write applications
  1. **Throughput** – Our system may need to handle a lot

of data in a short period of time.

2. **Response**–Our system may need to react to events quickly
3. **Testability**–Setting up equipment to test embedded software can be difficult
4. **Debug ability**–Without a screen or a keyboard, finding out what the software is doing wrong (other than not working) is a troublesome problem
5. **Reliability** – embedded systems must be able to handle any situation without human intervention
6. **Memory space** – Memory is limited on embedded systems, and you must make the software and the data fit into whatever memory exists
7. **Program installation** – you will need special tools to get your software into embedded systems
8. **Power consumption** – Portable systems must run on battery power, and the software in these systems must conserve power
9. **Processor hogs** – computing that requires large amounts of CPU time can complicate the response problem
10. **Cost** – Reducing the cost of the hardware is a concern in many embedded system projects; software often operates on hardware that is barely adequate for the job.

- Embedded systems have a microprocessor/ microcontroller and a memory. Some have a serial port or a network connection. They usually do not have keyboards, screens or disk drives.

4. Hardware Design and Software Design

The design of entire system consisted of two part which are hardware and software. The hardware are designed by the rules of embedded system, and the steps of software consisted of Embedded C Compiler

4.1 Hardware Design

Hardware requirements

1. Micro controller
2. Relay
3. LCD
4. Push Buttons
5. Power Supply

4.2 Software design

Software Requirements:

Keil an ARM Company makes C compilers, macro assemblers, real-time kernels, debuggers, simulators, integrated environments, evaluation boards, and emulators for ARM7/ARM9/Cortex-M3, XC16x/C16x/ST10, 251, and 8051 MCU families.

The Keil 8051 Development Tools are designed to solve the complex problems facing embedded software developers Keil development tools for the 8051 Microcontroller Architecture support every level of software developer from the professional applications engineer to the student just learning about embedded software development.
When starting a new project, simply select the microcontroller you use from the Device Database and the µVision IDE sets all compiler, assembler, linker, and memory options for you.

Numerous example programs are included to help you get started with the most popular embedded 8051 devices. The Keil µVision Debugger accurately simulates on-chip peripherals (I²C, CAN, UART, SPI, Interrupts, I/O Ports, A/D Converter, D/A Converter, and PWM Modules) of your 8051 device. Simulation helps you understand hardware configurations and avoids time wasted on setup problems. Additionally, with simulation, you can write and test applications before target hardware is available

1. Embedded C Compiler

Embedded “C” Compiler

- ANSI C - full featured and portable.
- Reliable - mature, field-proven technology.
- Multiple C optimization levels.
- An optimizing assembler.
- Full linker, with overlaying of local variables to minimize RAM usage.
- Comprehensive C library with all source code provided.
- Includes support for 24-bit and 32-bit IEEE floating point and 32-bit long data types.
- Mixed C and assembler programming.
- Unlimited number of source files.
- Listings showing generated assembler.
- Compatible - integrates into the MPLAB IDE, MPLAB ICD.
- Runs on multiple platforms: Windows, Linux, UNIX, Mac OS X, and Solaris.

5. Embedded system tools

5.1 Assembler

An assembler is a computer program for translating assembly language, essentially a mnemonic representation of machine language into object code. A cross assembler (see cross compiler) produces code for one type of processor, but runs on another. The computational step where an assembler is run is known as assembly time. Translating assembly instruction mnemonics into Opcodes, assemblers provide the ability to use symbolic names for memory locations (saving tedious calculations and manually updating addresses when a program is slightly modified), and macro facilities for performing textual substitution typically used to encode common short sequences of instructions to run inline instead of in a subroutine. Assemblers are far simpler to write than compilers for high-level languages.

Assembly language has several benefits

Speed: Assembly language programs are generally the fastest programs around.
Space: Assembly language programs are often the smallest.

Capability: You can do things in assembly which are difficult or impossible in High level languages.
Knowledge: Your knowledge of assembly language will help you write better programs, even when using High level languages

6. Block diagram and Descriptions

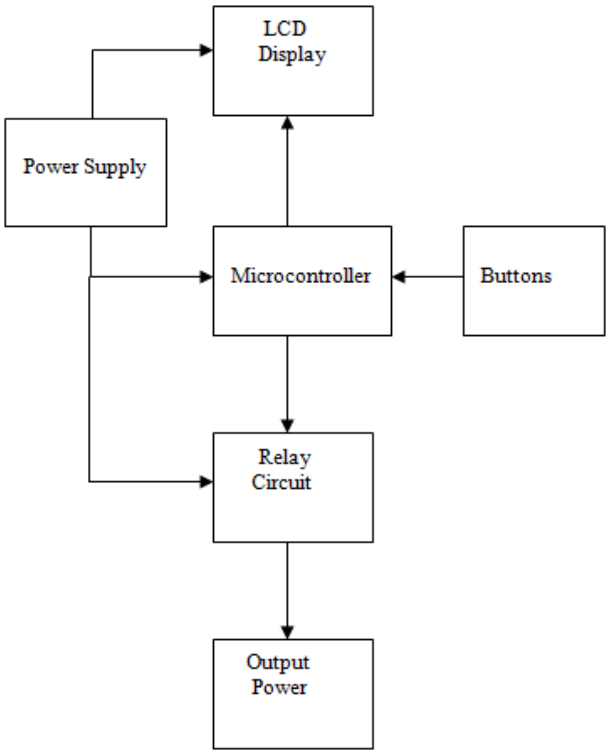


Figure 1: Block diagram of Hardware design

The major heart of this project is at89s52 microcontroller, the reasons why we selected this in this project? The AT89S52 provides the following standard features: 8K bytes of Flash, 256 bytes of RAM, 32 I/O lines, Watchdog timer, two data pointers, three 16-bit timer/counters, a six-vector two-level interrupt architecture, a full duplex serial port, on-chip oscillator, and clock circuitry. In addition, the AT89S52 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 con-tents but freezes the oscillator, disabling all other chip functions until the next interrupt or hardware reset.

7. Introduction to Microcontroller

A microcontroller (or MCU) is a computer-on-a-chip. It is a type of microprocessor emphasizing self-sufficiency and cost-effectiveness, in contrast to a general-purpose microprocessor (the kind used in a PC).

7.1 Typical Microcontroller Architecture and Features

The basic internal designs of microcontrollers are pretty similar. Figure1 shows the block diagram of a typical microcontroller. All components are connected via an internal bus and are all integrated on one chip. The modules are connected to the outside world via I/O pins.

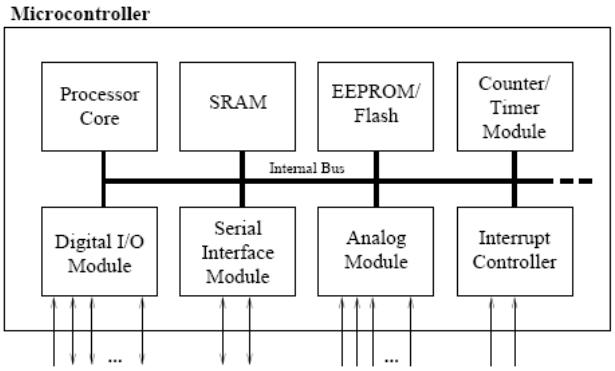


Figure 2: Basic Layout of Microcontroller

8. AT89S52 Microcontroller

Description

The AT89s52 is a low power, high performance CMOS 8-bit micro computer with 8K bytes of flash programmable and erasable read only memory(PEROM).The device is manufactured using Atmel's high density non-volatile memory technology and is compatible with the industry standard 80c51 and 80C52 instruction set and pin out. The on-chip flash allows the program memory to be reprogrammed in-system or by a conventional non-volatile memory programmer. By combining a versatile 8-bit CPU with flash on a monolithic chip, the Atmel AT89s52 Is a powerful microcomputer which provides a highly flexible and cost effective solution to many embedded controls applications. The main advantages of 89s52 over 8051 are

- Software Compatibility
- Program Compatibility
- Rewritability

The 89s52 microcontroller has excellent software compatibility, i.e. the software used can be applicable to any other microcontroller. The program written on this microcontroller can be carried to any base.

Program compatibility is the major advantage in 89s52. The program can be used in any other advanced microcontroller. The program can be reloaded and changed for nearly 1000 times.

Features

- Compatible with MCS-51® Products
- 8K Bytes of In-System Programmable (ISP) Flash Memory
  - 4.0V to 5.5V Operating Range
  - Fully Static Operation: 0 Hz to 33 MHz
  - Three-level Program Memory Lock
  - 256 x 8-bit Internal RAM
  - 32 Programmable I/O Lines

- Three 16-bit Timer/Counters
- Eight Interrupt Sources
- Full Duplex UART Serial Channel
- Low-power Idle and Power-down Modes
- Interrupt Recovery from Power-down Mode
- Watchdog Timer
- Dual Data Pointer
- Power-off Flag

89S52 Processor Architecture

The AT89s52 provides the following standard features: 8K bytes of Flash, 256 bytes of RAM, 32 I/O lines, three 16-bit timer/counters, a six-vector two-level interrupt architecture, a full-duplex serial port, on-chip oscillator, and clock circuitry. In addition, the AT89s52 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.

9. LCD Character 2 x 16

The LCD Module can easily be used with an 8051 microcontroller such as the AT89s52. The LCD Module comes with a 16 pin connector. This can be plugged into connector 16 pin. The pins on the 16 pin connector of the LCD Module are defined below.

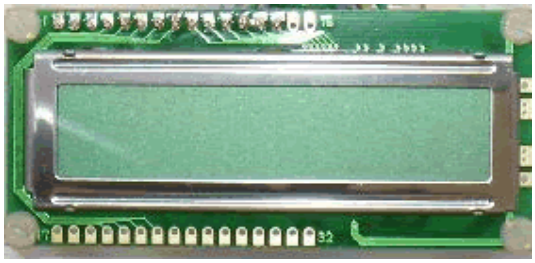


Figure 3: LCD Character 2 x 16 Module

LCD Character Background

The LCD Character standard requires 3 control lines as well as either 4 or 8 I/O lines for the data bus. The user may select whether the LCD is to operate with a 4-bit data bus or an 8-bit data bus. If a 4-bit data bus is used the LCD will require a total of 7 data lines (3 control lines plus the 4 lines for the data bus). If an 8-bit data bus is used the LCD will require a total of 11 data lines (3 control lines plus the 8 lines for the data bus).

The three control lines are referred to as EN, RS, and RW.

The EN line is called "Enable." This control line is used to tell the LCD that you are sending it data. To send data to the LCD, your program should make sure this line is low (0) and then set the other two control lines and/or put data on the data bus. When the other lines are completely ready, bring EN high (1) and wait for the minimum amount of time required by the LCD datasheet (this varies from LCD to LCD), and end by bringing it low (0) again.

The RS line is the "Register Select" line. When RS is low (0), the data is to be treated as a command or special instruction (such as clear screen, position cursor, etc.). When RS is high (1), the data being sent is text data which should be displayed on the screen. For example, to display the letter "T" on the screen you would set RS high.

The RW line is the "Read/Write" control line. When RW is low (0), the information on the data bus is being written to the LCD. When RW is high (1), the program is effectively querying (or reading) the LCD. Only one instruction ("Get LCD status") is a read command. All others are write commands--so RW will almost always be low.

Finally, the data bus consists of 4 or 8 lines (depending on the mode of operation selected by the user). In the case of an 8-bit data bus, the lines are referred to as DB0, DB1, DB2, DB3, DB4, DB5, DB6, and DB7.

An Example Hardware Configuration

As we've mentioned, the LCD requires either 8 or 11 I/O lines to communicate with. For the sake of this tutorial, we are going to use an 8-bit data bus--so we'll be using 11 of the 8051's I/O pins to interface with the LCD.

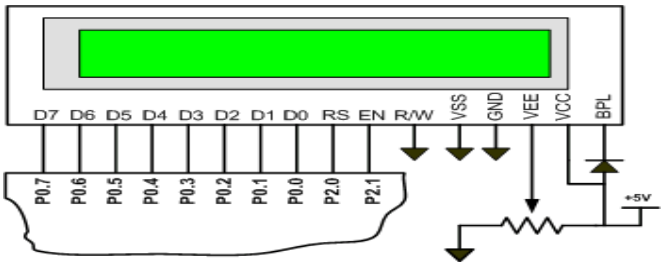
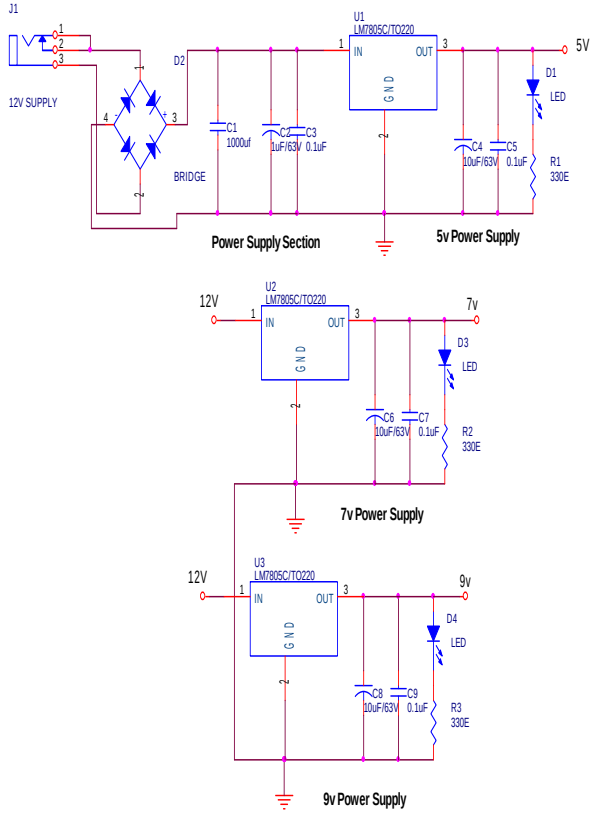


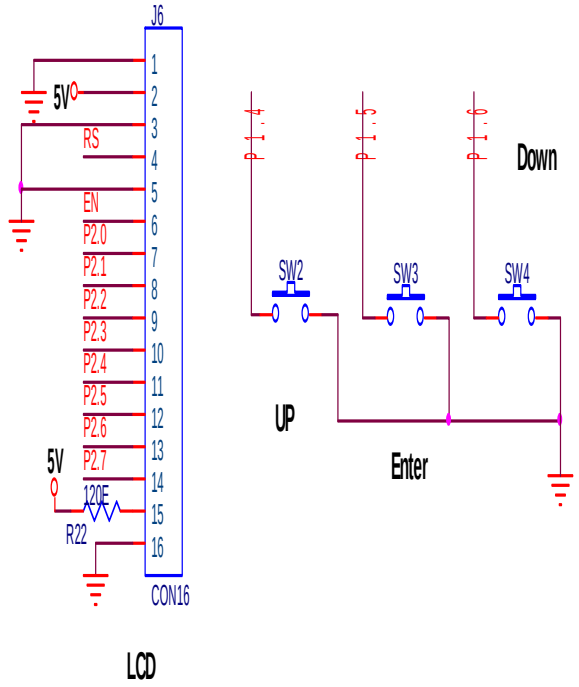
Figure 4: Pin connection LCD Character to microcontroller

10. Circuit for Digital Programmable Power Supply

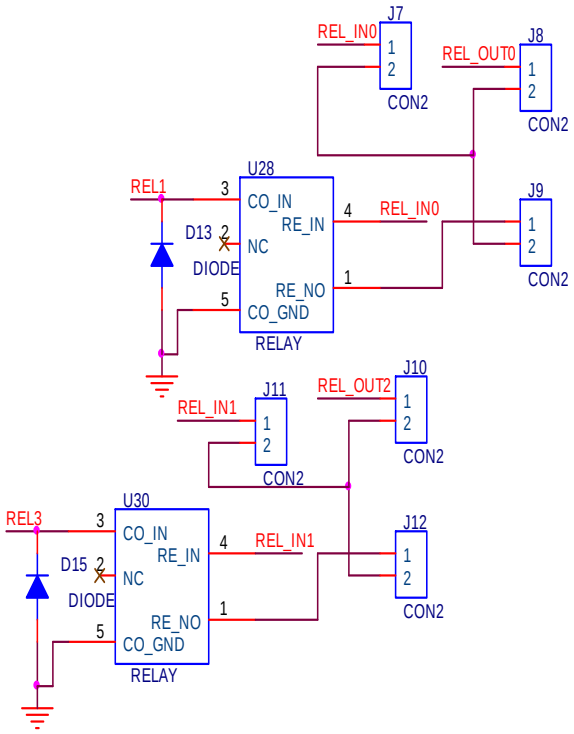
Power Supply Section



LCD and Button Interface



Relay Circuit



10. 1 Operation

The circuit consist four button connected to the microcontroller, where by three of them is used to select the voltage output and one button used as reset button by pressing the button circuit provide different outputs starting with low voltage, Buttons represent SW2, SW3, SW4 will give the information accordingly to microcontroller by sending appropriate commands and which OUTPUT is displayed on the LCD display as well by measuring the output (Display voltage = measured voltage) then select voltage according to the device requirement.

11. Operating system

Embedded systems often have no operating system, or a specialized embedded operating system (often a real-time operating system), or the programmer is assigned to port one of these to the new system.

Built- in self- test

Most embedded systems have some degree or amount of built-in self-test.

There are several basic types.

1. Testing the computer.
2. Test of peripherals.
3. Tests of power.
4. Communication tests.
5. Cabling tests.
6. Rigging tests.
7. Consumables test.
8. Operational test.

Program

```
#include "AT89x52.h"
#include "stdio.h"

#define uint unsigned int
#define uchr unsigned char

#define DATA P0

sbit RS=P1^0;
sbit EN=P1^1;

void Delay(uint tick)
{
while(tick--);
}

void LCD_DAT(uchr dat)
{
RS=1;
EN=1;
DATA=dat;
EN=0;
Delay(0xF0);
}

void LCD_CMD(uchr cmd)
{
RS=0;
EN=1;
DATA=cmd;
EN=0;
Delay(0xF0);
}

void LCD_Goto(uchr row,uchr col)
{
if(row==1) LCD_CMD(0x80+col);
else if(row==2) LCD_CMD(0xc0+col);
}

void LCD_CLR(void)
{
LCD_CMD(0X01);
Delay(0xFF);
}

void LCD_INIT(void)
{
LCD_CMD(0X38);
LCD_CMD(0X38);
LCD_CMD(0X0E);
LCD_CMD(0X06);
LCD_CLR();
Delay(0x100);
}

void LCD_Display(uchr *ptr,uchr row,uchr col)
{
if(row==1) LCD_CMD(0x01);
while(*ptr!='\0')
{
   LCD_Goto(row,col);
   LCD_DAT(*ptr);
   ptr++; col++;
}
}

void main(void)
{
unsigned char inc;
P0=P1=P2=P3=0xFF;
LCD_INIT();
Delay(0xFFFF);
LCD_Display(" Programmable ",1,1);
LCD_Display(" Power Supply ",2,1);
Delay(0xFFFF);
Delay(0xFFFF);
Delay(0xFFFF);
while(1)
{
     LCD_Display("Please enter ur",1,1);
  LCD_Display(" voltage range ",2,1);

  while(P2_5==1)
  {
     if(P2_7==0) inc++;
     if(P2_6==0) inc--;
     Delay(0x9fff);
     }
  if(inc==1)
  {
    LCD_Display("Now O/P Voltage",1,1);
    LCD_Display("5V",2,7);
    P2_0=P2_2=P2_3=P2_4=1; P2_1=0;
  }
  else if(inc==2)
  {
    LCD_Display("Now O/P Voltage",1,1);
    LCD_Display("6V",2,7);
    P2_1=P2_2=P2_3=P2_4=1; P2_0=0;
  }
  else if(inc==3)
  {
    LCD_Display("Now O/P Voltage",1,1);
    LCD_Display("9V",2,7);
    P2_0=P2_1=P2_2=P2_4=1; P2_3=0;
  }
  else if(inc==4)
  {
    LCD_Display("Now O/P Voltage",1,1);
    LCD_Display("10V",2,6);
    P2_4=P2_0=P2_3=P2_1=1; P2_2=0;
  }
  else if(inc==5)
  {
    LCD_Display("Now O/P Voltage",1,1);
    LCD_Display("12V",2,6);
    P2_0=P2_2=P2_1=P2_3=1; P2_4=0;
```

```
    }
    else
    {
     LCD_Display("Over Range",1,3);
     LCD_Display("Select Again",2,2);
     inc=0;
    }
    Delay(0xFFFF);
    Delay(0xFFFF);
    Delay(0xFFFF);
     }
}
```

12. Limitations

The system's reaction time is 250ms so in some cases additional protective hardware is required.

* The system output voltage is not programmable.
* The system cannot locate own hardware faults.
* To ensure proper functioning of the system one has to make ensure that the DC voltage is kept greater than battery voltage, so that it can be charged properly.

13. Future Scope

The future work can be done on the development of software that makes the system self diagnostic, detecting damage of components in itself. E.g., some controllers are there which generate an interrupt if the output of a pin does not match the sent value. The separate controller portion can be designed so that it can be put on to any supply and after programming various parameters in the controller any power supply could be controlled.

14. Acknowledgments

I take this opportunity to thank Dr. Wang Liqiang from School of Electronics Tianjin University of technology and education for his valuable guidance and for providing all the necessary support.

Second I would to thank Mr. Mohammed Gouse from Hades InfoTech Chennai India, where in 2008 I did my project and training on embedded system technology.

References

[1] AT89S52 Datasheet (PDF) - ATMEL Corporation - 8-bit Microcontroller with 8K Bytes In-System Programmable Flash - http://www.alldatasheet.com/datasheet pdf/pdf/82390/ATMEL/AT89S52.html
[2] Device Database http://www.keil.com/dd/chip/3411.htm
[3] Embedded systems, Architecture, programming and
[4] Design second Edition McGraw HILL by Raj Kamal
[5] C programming for embedded systems- Kirk Zurell
[6] http://en.wikipedia.org/wiki/Microcontroller- "Microcontroller".
[7] http://www.howstuffworks.com/microcontroller.htm - "How microcontroller work"
[8] Embedded Microcomputer system-jonathan w.Valvano

Author Profile

John Mashurano; I received Full Technician Certificate (FTC) in Civil Engineering from Mbeya Institute of Science and Technology Tanzania, in 2005. B.E in Electronics and Communication engineering from St.Joseph college of Engineering and Technology Tanzania in 2008 and Currently pursuing M.E. degrees in Signal and Information processing from Tianjin University of Technology and Education Tianjin China ,My research interest includes Information systems and Data Communication Networks (Routing and Switching), I worked in the field of information systems, and E- banking systems with different companies , SCI/NCR Tanzania as Field/customer support Engineer and National Microfinance bank Tanzania (NMB) as ATMs Systems Engineer.