Software Requirements Specification

for

Dec06-04: Geek Binary Alarm Clock

Version 1.1

Prepared by Yesuratnam Thommandru

Iowa State University – Electrical and Computer Engineering

September 28, 2006

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Yesuratnam Thommandru	9/20/06	Started SRS from template. Completed up to 2.2	1.0
Yesuratnam Thommandru	9/22/06	Completed SRS.	1.1

1. Introduction

1.1 Purpose

The purpose of this document is define the software requirements for the PICMicro PIC16877A microcontroller to be placed in the final end product of the Dec06-04: Geek Binary Alarm Clock senior design project version 1.0. The only software involved in this project is the embedded programming of the microcontroller.

1.2 Document Conventions

For this SRS embedded software design standards shall be used. As of yet there are no special sections which needs to be specifically pointed out. Requirements shall exist in a triage similar to the following: mandatory, nice-to-have, optional.

1.3 Intended Audience and Reading Suggestions

This document is intended for any user wishing to understand the underlying concepts in the software developed for the Geek Binary Alarm Clock. Readers will most likely include electrical and computer engineering professors and students as well as any interested embedded software developers. The rest of this SRS shall contain functional requirements for the Geek Binary Alarm Clock and their relationship to its PIC microcontroller. It is suggested that the reader first visit the Dec06-04: Geek Binary Alarm Clock website at http://seniord.ece.iastate.edu/dec0604/ go get an overview of the project.

1.4 Project Scope

The embedded software being specified in this SRS shall be developed in the C programming language using Microchip's MPLab Integrated Development Environment. The purpose of the software to fulfill all functional requirements as specified in the Dec06-04 Problem Statement (see Design Report). The goal is to quickly develop high quality software that can essentially run the Geek Binary Alarm Clock indefinitely. Once the software is developed and finalized it may be "burned" into the PIC16877A microcontroller.

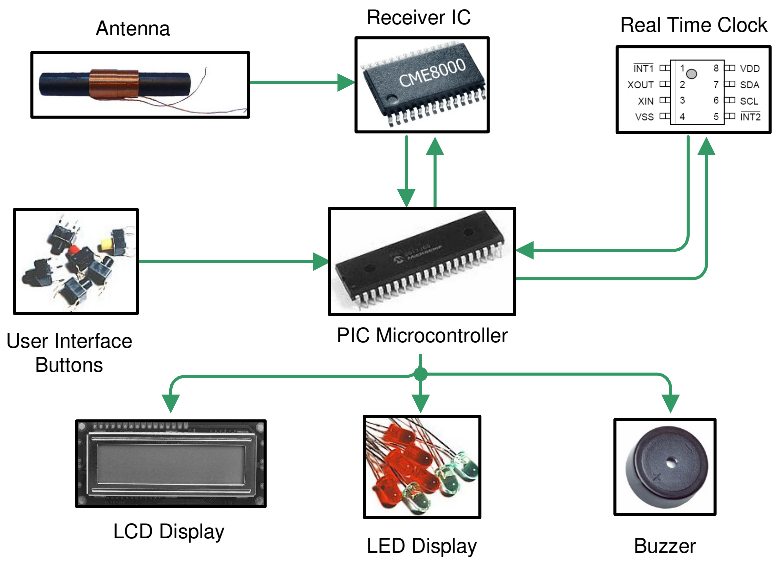
1.5 References

This document shall repeatedly refer to the Dec06-04: Geek Binary Alarm Clock website at http://seniord.ece.iastate.edu/dec0604/. At this website you will be able to view the initial Project Plan, Design Document, Design Review Powerpoint presentation, and Project Poster. This document shall also refer to the senior design course notes at http://seniord.ece.iastate.edu/. Further references can be acquired at http://www.microchip.com/.

2. Overall Description

2.1 Product Perspective

The overall product being specified in this document is the Geek Binary Alarm Clock. More importantly is the fact that this clock contains an embedded system requiring software. This software is the primary concern of this document. The following block diagram shall illustrate the importance of software's place in this project.



2.2 Product Features

The following is a brief description of the design objectives of the Dec06-04: Geek Binary Alarm Clock project:

- *Display the current/correct time in 12 or 24 hour format*
- *AM/PM indicator for 12 hour format*
- *Perform basic alarm clock functions (set a time to alarm, snooze, alarm sound)*
- *Self-correcting for daylight saving time (DST)*
- *Able to run during power outages for at least two hours*
- *Styled for the "geeky personality"*
- *Display time in binary code*
- *Option of having a digital display of time as well*

The underlying software in the clock must control all aspects of the project. The information shall be displayed to users via LEDs, an LCD, and buzzer. It is the microcontroller and therefore the software that must process and manage this information.

With the respect to software alone, the requirements are than the microcontroller receive information from the Reciever IC, Real Time Clock, and UI buttons, process that information, convert it to a usable human form (numbers in this case), and output that information to the array of LEDs, the LCD, and buzzer.

2.3 User Classes and Characteristics

The intended user(s) are those who know binary code and fit the geek persona. It can also be used by those who would like to learn binary through a practical application. This should not be used by children due to certain dangers such as small parts and electrical components.

This object is a clock meant to display time primarily in binary with LED's. It is also intended to help individuals learn binary by using the secondary display. This clock can also be used as a standard alarm clock as well. It shall sound at a designated time by the intended user.

With respect to software, no user shall have access once the main program is "burned" into the PIC microcontroller. There is also no software support for the Geek Binary Alarm Clock and as such there will be no user class.

2.4 Operating Environment

The operating environment of the Geek Binary Alarm Clock is limited to indoors. The environment should be able to supply the appropriate power needed through a regular wall outlet. It should not be used in areas that contain a lot of moisture such as the bathroom or pool room. It should not be in an area that it could potentially be dropped in a sink such as a kitchen or laundry room. It should be limited to places such as bedrooms, home offices, and in the work office.

With respect to software the operating environment shall be an embedded system. A primary program shall be written using the C programming language in MPLab IDE and compiled with CCS C Compiler. The hardware environment shall be a PIC16F877A microcontroller. The software must work in tangent with hardware components such as LEDs, LCD, reciever IC, RTC, and buttons and switches. The embedded software "burned" into the microcontroller must be the "glue" that holds the Geek Binary Alarm Clock together.

UPDATE – 9/27/06 – Unfortunately the CCS C compiler will not work for the PIC16F877A because it requires a 14-bit opcode and CCS C compiler only handles microcontroller that have 12-bit opcodes. CCS technical support responded to an email inquiry stating that in order to compile C code for the PIC16F877A the PCM compiler would have to be purchased. Upon learning this fact a new, free compiler was seeked. The result was HI-TECH PICC-Lite, a zero support, free compiler found online. Therefore the compiler that will be used for the Dec06-04 project will be the HI-TECH PICC-Lite compiler.

2.5 Design and Implementation Constraints

The software required for the Geek Binary Alarm Clock falls into the category of embedded systems. As such there are certain design and implementation contraints posed to the software developers. The first contraint is based upon the hardware chosen. A PIC microcontroller is capable of executing only Assembly or C instructions. The second contraint is the development environment. Any C compiler and associated IDE may be used but a specific programmer is necessary to "burn" the program. A PIC compliant programmer must be used such as the following: PICSTART Plus, MPLab ICD, PICkit 1 or PICkit 2, etc.

High quality, error free, and well documented code is desired. This document as well as final documentation shall asist groups in the future who wish to use the Dec06-04 project as a reference.

2.6 User Documentation

Documentation for the software developed for the Dec06-04: Geek Binary Alarm Clock project shall consist of this document and a short design document. An effort will be made to use IEEE standard templates for both documents. All four team members of Dec06-04 shall play a role in the production of software documentation.

A separate user manual has been written by Diana Calhoun for use with the Geek Binary Alarm Clock. This user manual has no reference to software but is a simple tutorial for all users.

2.7 Assumptions and Dependencies

It is assumed that the software shall be written in the C programming language using MPLab IDE and the CCS C Compiler. It is also assumed that the software is being written for a PIC microcontroller and shall fulfill all functional requirements listed in the Project Plan.

UPDATE – 9/27/06 – As stated in the preceding update the CCS C compiler will not be used in this project. The HI-TECH PICC-Lite compiler will be used as an alternative.

3. System Features

3.1 Display the current time in 12 or 24 hour format

3.1.1 Description and Priority

The software must display the current time in 12 or 24 hour formats in both the LEDs and the LCD. The priority level is high for displaying time but low for switching between formats.

3.1.2 Stimulus/Response Sequences

The microcontroller must periodically check the real time clock and the reciever IC for the current time. In order to switch formats the microcontroller must listen for toa digital switch to be placed on the clock.

3.1.3 Functional Requirements

REQ-1: The software embedded in the PIC micrcontroller must periodically check the reciever IC for the current time and use the real time clock as its reference.

REQ-2: When a digital switch is triggered the software must switch formats between 12 and 24 hours.

3.2 AM/PM indicator for 12 hour format

3.2.1 Description and Priority

There shall be a single LED signal whether the time is AM or PM in 12 hour mode. The on/off signal shall be controlled in software. This is a lower priority requirement.

3.2.2 Stimulus/Response Sequences

When the digital switch is in 12 hour format the software shall determine whether the LED should be on or off.

3.2.3 Functional Requirements

REQ-1: The software must be able to detect the position of the digital switch controlling time format.
REQ-2: The software must detect whether it is AM or PM in 12 hour mode.
REQ-3: Once REQ-2 is determined the software must activate or deactivate the coresponding LED.

3.3 Basic alarm clock functions (set time or alarm, sound alarm, snooze)

3.3.1 Description and Priority

The Geek Binary Alarm Clock should posess all the basic functionality of a standard clock. The user should be able to set the time and alarm. The clock should be able to sound the alarm which the user should be able to silence temporarily using a snooze function. All of these high priority requirements must be taken care of in software.

3.3.2 Stimulus/Response Sequences

There are four buttons: Time, Alarm, Up, and Down. Setting the time and alarm shall be handled by holding down either button and using the Up and Down buttons. Software shall trigger the piezo buzzer when the alarm time is reached. When the alarm sounds the user shall be able to hit any button other than alarm to snooze.

3.3.3 Functional Requirements

REQ-1: The software must be able to detect the position and time depressed of time, alarm, up and down.
REQ-2: The software must trigger the piezo buzzer when the alarm time is reached.
REQ-3: A function in software must suspend the piezo buzzer for a short amount of time (TBD).

3.4 DST, leap year, and auto-update

3.4.1 Description and Priority

The Geek Binary Alarm Clock shall automatically adjust for daylight savings time, leap year, and auto-update. This is a medium priority requirement.

3.4.2 Stimulus/Response Sequences

No human stimulus is involved in this feature. Auto-update shall require stimulus from the reciever IC. DST and leap year must be handled by software by hardcoding certain dates or receiving input from the RTC.

3.4.3 Functional Requirements

REQ-1: The software must be able to process information passed to the microcontroller from the reciever IC and RTC.
REQ-2: The software must decrement or increment the hour variable at the appropriate time for DST.
REQ-3: The software must be able to adjust year variable on leap years.

3.5 Able to run during power outages for at least two hours

3.5.1 Description and Priority

The Geek Binary Alarm Clock shall be able to operate for a minimum of two hours during a power outage as well as shift into sleep mode to preserve time and alarm. This is a high priority feature.

3.5.2 Stimulus/Response Sequences

The stimulus for this feature shall consist of "pulling the plug" on the clock. Once the electrical plug is removed from the socket the software must switch and use the battery for energy.

3.5.3 Functional Requirements

REQ-1: The software must be able to detect that electricity from a wall socket has been cut off.
REQ-2: The software must switch to the battery and maintain all present operations.
REQ-3: The software must deactivate LEDs, LCD, and any other energy consuming device and move into sleep mode.
REQ-4: The software must keep track of time and alarm during a power outage.

3.6 Display time in binary code or standard digital format

3.5.1 Description and Priority

The Geek Binary Alarm Clock shall be able to display the current time is both binary and standard formats. This is the highest priority feature of this project.

3.5.2 Stimulus/Response Sequences

There is no human stimulus for this feature. The entire sequence shall exist in software only. The only stimulus the software may receive is a signal to turn the LCD off.

3.5.3 Functional Requirements

REQ-1: The software must divide the current time into segments.
REQ-2: The software must convert each digit into a usable format for the LEDs or LCD. This will most likely mean hardcoding.
REQ-3: The software activate and deactivate the appropriate LEDs and LCD cells.
REQ-4: The software must detect a digital switch to turn the LCD off.

4. External Interface Requirements

4.1 User Interfaces

N/A

4.2 Hardware Interfaces

Once the software is written a programmer will be necessary to "burn" the software. The programmer is a hardware device called the PICSTART Plus. The device onto which the software will be "burned" is a Microchip PIC16F877A microcontroller. This is the primary hardware interface for the software in this project. The rest of the hardware involved in the Geek Binary Alarm Clock is described in detail in the Dec06-04 Design Report.

4.3 Software Interfaces

Software shall be written in C using MPLab IDE and compiled using the CCS C Compiler. Software interfaces will only exist prior to placement of the primary program onto the microcontroller. The embedded system has no interaction with any other software systems.

UPDATE – 9/27/06 – As stated above the CCS C compiler will not be used in this project. The HI-TECH PICC-Lite compiler will be used instead.

4.4 Communications Interfaces

N/A

5. Other Nonfunctional Requirements

5.1 Performance Requirements

A minimalistic approach shall also be used to preserve limited amount of memory. The size of the program written must not exceed memory size and must execute reasonably quickly.
Speed can be assessed from a human perspective: all functions must work fast enough that "lag" is never detected. Individual LEDs and LCD cells must be updated quickly. The software must never make the clock fall behind standard time.There shall be no software maintenance or evolution for this project.

5.2 Safety Requirements

N/A

5.3 Security Requirements

N/A

6. Other Requirements

The software for the Dec06-04: Geek Binary Alarm Clock should be developed quickly, cheaply, and effeciently. It should also contain no errors, be dependable, be easy to read, be well documented, and reusable. Future teams may wish to use this software as a reference.
The software does not have to be scalable or extensible. This software is not intended to grow in size or have future versions.

Appendix A: Glossary

Binary – a number system using a base of 2 consisting of on/off, high/low, or ones and zeros used by almost all computer systems.

C programming language – a powerful, efficient, low-level language developed in the 1970s for Unix OS now used for systems software and general applications.

Daylight saving time (DST) – the time is which clocks are set exactly one hour ahead of standard time in order to provide more daylight during late spring, summer, and fall.

Fourteen possible calendar years – There are only fourteen possible calendar years supported that include all of the leap years and DST.

Geek - slang – a term to describe a person with good computer skills, an interest in technology, and firm knowledge of the sciences…usually accompanied with an almost complete social ineptitude.

Light-emitting diodes (LED) – a type of diode that emits light when it is subjected to a flow of current. LEDs may have different colors depending on the material used.

Liquid-crystal displays (LCD) - two thin sheets of plastic filled with individual cells of ionic liquid crystal capable of being manipulated by a current.

PCB (printed circuit board) – a thin plastic board onto which electronic components such as resistors and capacitors are soldered.

12 hour format – the standard hourly display of analog and digital clocks which a separate indication for AM or PM. e.g. 12:34 pm

24 hour format – (a.k.a. military time, universal time) – the hourly display of clocks without a separate indication for AM or PM that increments hours upon reaching noon based on 24 hours. e.g. 23:45 equals 11:45 pm.