IJ TECH

School of Engineering Science • Burnaby, BC • V5A 1S6
ensc440-IJtech@sfu.ca

March 9, 2006

Dr. Andrew Rawicz
School of Engineering Science
Simon Fraser University
Burnaby, British Columbia
V5A 1S6

Re: ENSC 440 Design Specifications for a Smart Alarm Clock

Dear Dr. Rawicz,

The enclosed document, Design Specifications for a Smart Alarm Clock, details the design specifications of our product currently in development for the ENSC 440 course.

In an active attempt to remedy the sociological problem of sleep deprivation plaguing over 50 million Americans, we are constructing an alarm clock with a pulse measurement module that awakes individuals at the designated sleeping stage to feel more energetic and less prone to the effects of sleep inertia. An MP3 module allows users to rise to their favourite tunes, while a Bluetooth light control plug simulates the gradually intensifying light of a sunrise.

The accompanying document outlines the design features and testing procedures that the Smart Alarm Clock is to fulfill upon completion. The main components of our device include an Alarm Clock subsystem, an MP3 subsystem, a Pulse Measurement subsystem, and a Light Control subsystem. A detailed approach to the design specification provides an unambiguous design standard for which all team members follow.

Inglewood Jack Technologies Inc. consists of five talented and innovative individuals who study engineering at Simon Fraser University: Albert Su, Christian Le, Herman Leung, William Ng, and Matthew Ng. If you have any questions or concerns, we will be pleased to answer them. We can be contacted via email at ensc440-IJtech@sfu.ca.

Sincerely,

Christian Le
Chief Operations Officer
Inglewood Jack Technologies, Inc.

Enclosure: Design Specifications for a Smart Alarm Clock

Inglewood Jack Technologies, Inc

Design Specifications for a Smart Alarm Clock

Project Team: *Christian Le*
Herman Leung
Matthew Ng
William Ng
Albert Su

Contact Person: Christian Le
ensc440-IJtech@sfu.ca

Submitted to: Dr. Andrew Rawicz – ENSC 440
Steve Whitmore – ENSC 305
School of Engineering Science
Simon Fraser University

Issued Date: March 9, 2006
Revision: 1.0

EXECUTIVE SUMMARY

With the advent of electrical lighting, shift work, social diversion and global competition, modern lifestyle has evolved to sacrifice an average of two hours of sleep a night for more time and productivity in society. Sleep deprivation results in increased vulnerability to accidents, reduced work performance and decreased concentration, yet people in industrialized nations refuse to tradeoff their busy lifestyle for more sleep. As a practical solution to the widespread and serious health problem, the Smart Alarm Clock applies the theories of circadian rhythm and strives to allow its users to wake up feeling more refreshed, sustaining better moods and performing tasks with improved efficiency and accuracy.

Central to the Smart Alarm Clock is a pulse measurement device that monitors user sleep rhythm. Sleeping cycle data are relayed and processed to awake the user at the Rapid Eye Movement stage in the closest proximity to the preset wake time. To promote a pleasant waking experience, an MP3 subsystem and a light control module replaces agonizing alarm sounds with music and light.

To demonstrate the concept of the Smart Alarm Clock, a proof-of-concept model will first be designed and constructed. In the first phase of development, the proof-of-concept model will support full functionalities including the following features:

i. Active sleep cycle monitoring to activate gradually intensifying music and light at the proper sleep stage and time to reduce grogginess.

ii. MP3 storage and playback capabilities for the user to awake to music.

iii. A functional light control unit that enables user to rise to their favorite light or the operation of other electronic devices.

iv. Easy to use alarm clock and menu system.

v. Preliminary compliance to safety and environment standards.

Upon the completion of the proof-of-concept model in April, we will pursue to the development of a production version which adds:

1. Attractive packaging and casing for the alarm clock and light control unit.

2. Additional MP3 capabilities for the user to awake to customized music.

3. Full compliance to safety, environmental and energy efficiency standards.

Table of Contents

List of Figures

List of Tables

Glossary

AC	Alternating Current
ACS	Alarm Clock Subsystem
DAC	Digital to Analog Converter
dB	Decibel
EEPROM	Electrically Erasable Programmable Read Only Memory
FIFO	First In First Out
GPIO	General Purpose Input/Output
I^2C	Inter-Integrated Circuit
ISM	Industrial, Scientific and Medical
LCD	Liquid Crystal Display
LCS	Light Control Subsystem
LD	Light Detector
LE	Light Emitter
LED	Light Emitting Diode
MMC	MultiMedia Card
MPEG	Moving Picture Experts Group
MP3	Moving Picture Experts Group Layer 3
MPS	Music Player Subsystem
NREM	Non Rapid Eye Movement
OSI	Open Systems Interconnect
PMS	Pulse Measurement Subsystem
REM	Rapid Eye Movement
RMS	Root Mean Square
SAC	Smart Alarm Clock
SCI	Serial Control Interface
SD	Secure Digital
SDI	Serial Data Interface
SNR	Signal to Noise Ratio
SPI	Serial Peripheral Interface
SRAM	Static Random Access Memory
SWS	Slow-Wave Sleep
UART	Universal Asynchronous Receiver/Transmitter
USART	Universal Synchronous/Asynchronous Receiver/Transmitter

1. Introduction

The Smart Alarm Clock (SAC) is a device that will help users wake up in the morning with greater ease and feeling more refreshed. This goal is achieved by using a combination of gradually intensifying light and sound as stimuli initiated based on the user's sleep stage. Users can wake up to their favourite music by simply loading MP3 files into the device. The users can also use their existing bedside lamp for the light feature by simply plugging the electrical plug into a subcomponent of the Smart Alarm Clock. Sleep stage is monitored by examining the user's pulse rate for transitional cues. The project will be developed in two phases. Phase one consists of developing a prototype as a "proof of concept" model scheduled for completion in April of 2006. Phase two consists of improving upon the features' effectiveness and usability in preparation for commercial sales.

1.1. Theoretical Background

Sleep stage is divided into two distinct groups: Rapid Eye Movement (REM) sleep and Slow-Wave sleep (SWS). SWS is further broken down into four stages from 1 to 4. REM sleep stage is characterized by the high amount of brainwave activity associated with it, which is very similar to brainwave patterns of an individual who is awake. Stages 1 and 2 of SWS also show similar brainwave patterns as waking. However, Stages 3 and 4 show a much slower brainwave activity. SWS is often collectively lumped together under the term Non-REM (NREM) sleep [15].

A typical night of sleep consists of a periodical rhythm of REM and NREM sleep stages. Each periodic cycle lasts anywhere between 90 to 110 minutes, depending on the time during the night. During the early part of the night, there is more Stage 3 and Stage 4 sleep. As the night progresses, the amount of Stage 3 and Stage 4 sleep reduces, resulting in an alternating pattern between Stage 2 sleep and REM sleep. Figure 1 shows the typical sleep cycle experienced throughout the night.

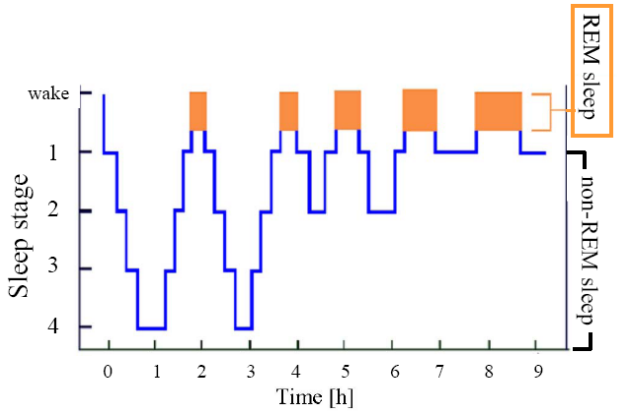


Figure 1: Sleep Cycle throughout a Typical Night of Sleep [12]

Sleep inertia is a physiological state characterised by a decline in motor dexterity and a subjective feeling of grogginess, immediately following an abrupt awakening from deep sleep. The effects of sleep inertia typically last from one minute to several hours. Studies have been conducted to verify the differential effect of sleep inertia based on the sleep stage at which wakening occurs. One such study involved several university students being subjected to two conditions of consecutive nights of REM and NREM sleep stage (Stage 2 in this case) wakening. Assessments on performance were done immediately after awakening and then every three hours thereafter. The results have shown that waking up from Stage 3 or 4 produces the most sleep inertia, whereas Stage 2, REM, and Stage 1 produces reducing amounts of sleep inertia, in that order [4]. Thus, it is ideal to wake the individual during, or shortly after the completion of REM stage.

The autonomic activities associated with the REM and NREM sleep can be used to differentiate the two. REM sleep is characterized by variable heart rate with high bursts whereas SWS shows slow decline in heart rate [15]. Additional studies have shown that significant heart rate variation metrics are observed in different stages of sleep. These metrics compare the variability of the beat-to-beat period of the measured heart beat. SWS and REM shows the greatest observable contrast [5]. By using a pulse rate or heart rate monitor, it is possible to track the variable heart rate pattern associated with REM, and identify when transitions between SWS and REM occur through a threshold level detection of the heart rate variability [12].

Additional considerations must be made when attempting to wake a user up during REM sleep. The importance of REM sleep is still under debate, but a widely held belief is that REM sleep consolidates memories and aids in learning [9]. Thus, it is also ideal to maximize REM sleep whenever possible, by waking up users near the end of the REM sleep stage.

The exposure of light in the morning stimulates the Suprachiasmatic Nucleus (SCN), a group of cells in the hypothalamus that respond to light and dark signals. The SCN controls the circadian biological clock. The SCN responds to light by delaying the release of hormones associated with sleep onset, and releases hormones that raise the body temperature [14]. Thus the use of light will reduce the user's urge to fall back to sleep, and facilitate in the waking process.

1.2. Scope

This document standardizes the design specifications that must be met by the proof of concept model of the Smart Alarm Clock to fulfill the functional requirements as discussed in the *Functional Specifications for a Smart Alarm Clock.* Additional specifications toward the production model will not be stated in this document. An extensive list of tests to be conducted in the development process of the proof of concept model provides the test plan framework for our module and integration engineers.

1.3. Intended Audience

The primary audiences of this document are product developers, integration engineers and quality assurance personnel. The design specification acts as an implementation guideline in order to fulfill all functional requirements for the proof of concept model.

Chief engineers and executives can make use of this document to plan, direct, motivate and control development progress. In carrying out product tests, the design specification provides a comprehensive test plan for test engineers.

2. System Overview

The Smart Alarm Clock is composed of four subsystems working together to provide the overall system functionality. The figure below shows an overview of the various subsystems and their interconnections.

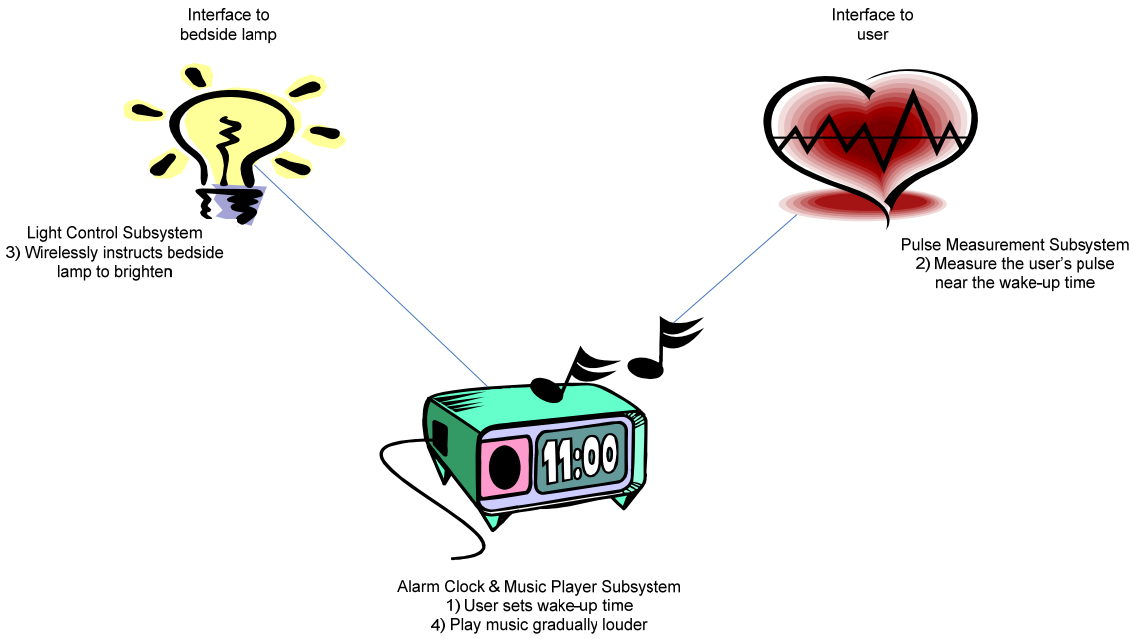


Figure 2: Smart Alarm Clock Modules & Interconnections

2.1. Alarm Clock Subsystem

As the central component, the Alarm Clock subsystem keeps track of the current time and the time when the user needs to wake up. A user interface allows easy setting of the wake-up time, selection of the music to wake up to, and various other user configurations.

This subsystem is responsible for communicating with the other subsystems during the waking process. It takes the user's pulse rate from the Pulse Measurement subsystem and processes the raw data to determine the appropriate time to initiate the light and sound features. Communication with the Light Control subsystem is performed wirelessly to instruct the gradual brightening of an external light source. Similarly, communication with the Music Player subsystem instructs it to play the specified song while gradually increasing the sound volume.

2.2. Music Player Subsystem

This subsystem incorporates MP3 processing and non-volatile storage, allowing the users to load their own songs. Speakers and audio amplifiers are also incorporated to provide support for stereo sound. Digital volume adjustment features are available for automatically increasing the sound volume of the current song being played, as controlled by the Alarm Clock subsystem.

Though there are two subsystems for the alarm clock and music player, these subsystems will be in the same physical unit as shown in Figure 1.

2.3. Pulse Measurement Subsystem

This subsystem is responsible for measuring the user's pulse rate in a non-invasive manner. The measured data is then passed to the Alarm Clock subsystem where processing is done to determine any changes in the pulse rate.

2.4. Light Control Subsystem

This subsystem interfaces to the user's current bedside lamp, which acts as the external light source. It controls the light intensity by varying the amount of AC power that gets passed to the external light source. It communicates with the Alarm Clock subsystem wirelessly.

3. System Hardware Design

The hardware design approach of the SAC is divided into four sections, one for each functional subsystem.

3.1. Alarm Clock Subsystem Design

The Alarm Clock Subsystem (ACS) is the central component of the SAC. It is responsible for keeping track of the current time as well as the wake up time. It provides visual feedback to the user through two Liquid Crystal Displays (LCDs) and accepts user inputs through directional and push buttons. The ACS Subsystem also interfaces with the other subsystems that make up the complete SAC system.

The high level functional block diagram of the ACS is shown in Figure 3.

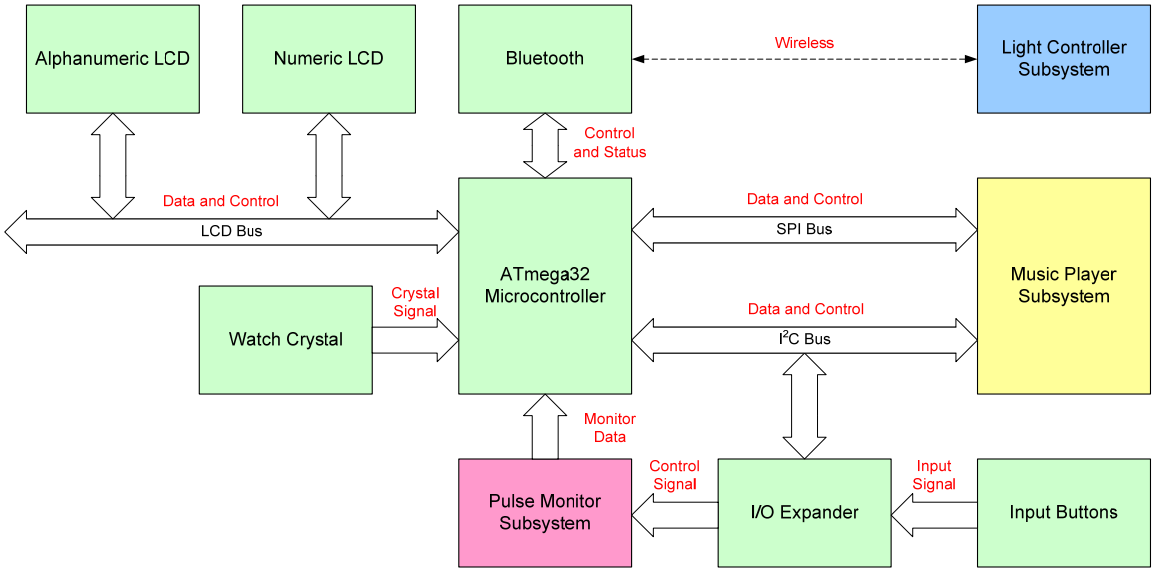


Figure 3: Functional Block Diagram of the Alarm Clock Subsystem

There are several components of interest: the ATmega32 microcontroller, Alphanumeric LCD, Numeric LCD, I/O Expander, Input Buttons, Bluetooth, and Watch Crystal. The ACS also interfaces with the Light Controller Subsystem, Music Player Subsystem, and Pulse Monitor Subsystem. Through the Pulse Monitor Subsystem, the user's pulse rate is measured and closely monitored for the ideal time to wake the user. Once the ideal time is determined, the Light Controller Subsystem is instructed wirelessly via Bluetooth to gradually increase the intensity of the external light source. The Music Player Subsystem is instructed to begin playing songs at an increasing volume level.

3.1.1. ATmega32 Microcontroller

The central processor of the SAC is the ATmega32 microcontroller from Atmel. The microcontroller is responsible for communicating with various subsystems using various bus interface standards as well as general purpose input/output (GPIO) pins. The ATmega32 is an 8-bit microprocessor with 32 KB of Flash memory, 2 KB of internal static random access memory (SRAM) and 1 KB of Electronically Erasable Programmable Read Only Memory (EEPROM). It boosts a maximum throughput of 16 million instructions per second when operated at 16 MHz. External interrupt lines are also available on the microcontroller for interrupt generation, reducing the need to constantly "poll" the line value. This results in more efficient use of the microprocessor's processing time.

This chip features the Serial Peripheral Interface (SPI), Inter-Integrated Circuit (I^2C) interface, and Universal Synchronous/Asynchronous Receiver/Transmitter (USART) interface. Both the SPI and I^2C interfaces allow multiple devices to be connected simultaneously on a single bus, greatly reducing the number of pins used on the microprocessor. Communication over this bus is established in a master-slave fashion. Data lines are pulled-up so that in the event of bus contention, the amount of current flowing through the devices is limited.

The ATmega32 processor was selected based on the amount of features it provided, ease of development and low cost. Cost was also an important discriminator, since mass production of the SAC is intended.

3.1.2. LCM-S02004DSR 20x4 Alphanumeric LCD

To display the user menu, a 20 character by 4 line alphanumeric LCD is used. Due to the amount of text that needs to be displayed at a given time by the user menu, 2 lines will not be sufficient. The LCM-S02004DSR alphanumeric LCD was chosen since it incorporates a dot-matrix LCD driver (which is used to control each alphanumeric character). This allows the microcontroller to change the contents of the LCD using as little as 7 lines.

3.1.3. LCD-S401C71TR Four Digit Numeric LCD

As outlined in the functional specification, the current alarm time must be displayed to the user. This is achieved by using a four digit, 8-segment LCD display. The LCD-S401C71TR in particular is capable of displaying ":" between digits, allowing us to display the time format in "HH:MM" easily.

Figure 4 shows the size of digits on the numeric LCD, measured in millimetre [inches]. The size of these digits can be easily viewed by a person with 20/20 vision over a distance of 20 feet (6.1 meters) [6]. Since users will usually be closer to the alarm than 20 feet, clock visibility will not be an issue.

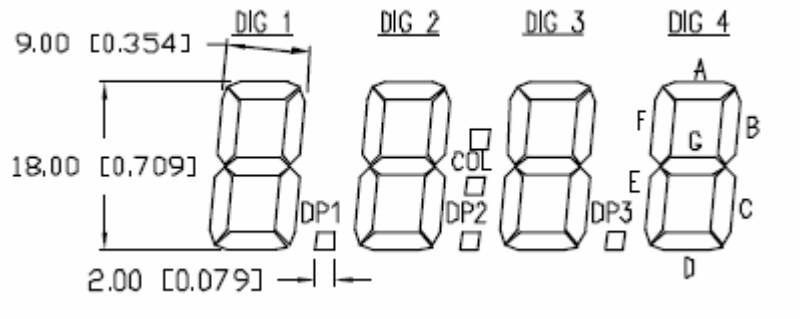


Figure 4: Digit Size on Numeric LCD

3.1.4. ICM7211 LCD Driver

To reduce the number of lines required to drive the numeric LCD, the ICM7211 LCD driver is used. This allows the microcontroller to control the contents of the LCD using as few as 7 lines. Moreover, it will provide the signals necessary to decode and drive each segment for each digit, since each segment needs to be driven by alternating logic level.

3.1.5. EmbeddedBlue eb505 Bluetooth Module

In order to communicate wirelessly with the Light Controller Subsystem, the EmbeddedBlue eb505 manufactured by A7 Engineering allows us to accomplish this task with minimal prototyping. This module uses the Bluetooth standard for wireless communication. The benefits of using Bluetooth for our design are:

1. Provides worldwide specification, small-form factor and low cost.
2. Less susceptible to noise and interference due to the use of frequency hopping spread spectrum.
3. Ability to form small localized network of up to 8 Bluetooth devices coexisting in master-slave relationship. Allows for possibility of "add-on" components to be designed to expand the functionalities of the SAC.

The reason for using the eb505 is because the complexity of working with the Bluetooth standard, from implementing the Bluetooth protocol stack to circuit layout and antenna matching, is removed. Communication simply requires setting up the connection between the two endpoints. Once the connection is established, a serial port link is established through emulation. Data transmission can then proceed as if using a simple Universal Asynchronous Receiver/Transmitter (UART) device.

Figure 5 shows the eb505 module, which can be easily prototyped.

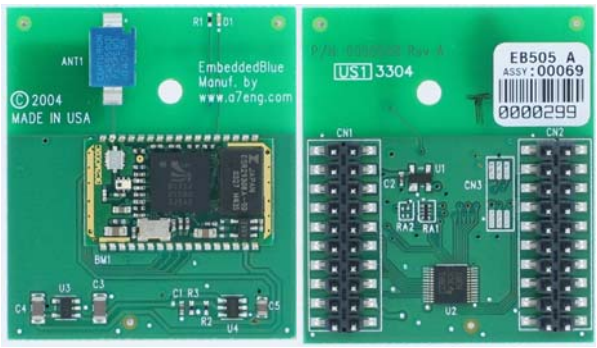


Figure 5: EmbeddedBlue eb505 Module; front view (Left) and back view (Right)

3.1.6. PCF8575 Remote 16-bit I/O Expander

Due to the large number of input signals required to interface with the input buttons, directly interfacing the buttons with the microprocessor is a pin exhausting process. The PCF8575 I/O expander uses the I^2C bus to interface with the microprocessor, reducing the pin usage to two pins for the I^2C interface, and a third pin for interrupt generation. The PCF8575 is a quasi-bidirectional I/O interface, which means that the direction of each I/O port does not need to be individually configured. An interrupt signal can be generated whenever an input signal changes from its previous state, signifying to the microprocessor to perform a read. The interrupt signal is cleared shortly after the read.

3.1.7. Input Buttons

Two types of input buttons are available on the SAC. The ALPS SKQUCAA010 is a 4-direction tactile input button with center push is available for navigating through the user menu. Connection to ground is established whenever the button is pushed in a particular direction, or pressed down. Figure 6 shows the appearance of the directional button.

Figure 6: 4-Direction Input Button

The other type of input button used is the tactile push button. Two of these buttons are available, one for turning off the alarm and one for toggling the external light source.

3.1.8. Seiko Tuning Fork Watch Crystal

To perform accurate time-keeping function, a 32.768 kHz watch crystal is used. The watch crystal can be connected to the microcontroller's TOSC1/TOSC2 pins where internal amplification is done. This amplified clock signal is used to clock an 8-bit counter at a rate of 32.768 kHz / 128 = 256 Hz. Thus, the 8-bit timer will overflow every 1 second allowing accurate time-keeping to take place.

The crystal has a frequency deviation of 5 parts per million, or ±0.16384 Hz. This effectively creates a deviation of ±12.96 seconds over a period of 30 days. This satisfies the functional requirement of no more than ±1 minute per month.

3.1.9. Voltage Translators

Although not shown on the high level block diagrams, voltage translation is necessary between certain hardware components and the microprocessor. This is accomplished by using buffers connected to a different power supply rail. Two supply rails are made available: 5 V and 3.3 V. The 74AHC126 buffer connected to 3.3 V supply is used to translate from 5 V to 3.3 V, since this buffer can tolerate input voltage levels much higher than its supply. The 74AHCT244 buffer is used to translate from 3.3 V to 5 V, since the minimum voltage for high input logic level is 2 V. Please refer to the schematic in the Appendix for further details.

3.1.10. LCD Data and Control Bus

The alphanumeric and numeric LCDs share the same data and control bus. This is aptly termed, the LCD Bus. Sharing the LCD Bus reduces the number of I/O pins required without sacrificing performance. Table 1 and Table 2 show the pin assignments for the alphanumeric and numeric LCD, respectively.

Table 1: Alphanumeric LCD Pin Assignment

Name	Type	Function
DB4~DB7	Input/Output	Data Bus
R/W	Input	Read/Write Select
RS	Input	Register Select
E	Input	Enable

Table 2: Numeric LCD Pin Assignment

Name	Type	Function
B0~B3	Input	Data Input
DS1, DS2	Input	Digit Select
!CS1	Input	Chip Select

The LCD data bus consists of 4 lines which encompass DB4~DB7 and B0~B3. The control bus consists of 2 lines which encompass R/W, RS and DS1, DS2. Dedicated control lines are used to control E and !CS1. In total, eight pins are used on the microprocessor.

Data and control can be placed on the bus at any time. As long as E and !CS1 remain low and high, respectively, there is no risk of data being latched. It is the action of strobing these control lines that result in data and control being latched. Provided that E is held low whenever signals are changing on the LCD data and control bus, there is no risk of the alphanumeric LCD trying to drive the data bus (in response to R/W signal being driven low). PORTA on the microcontroller is reserved for interfacing with the LCDs. Please refer to the Appendix for detailed schematics.

3.1.11. Bluetooth Interconnection

The eb505 module has several pins of interest besides the ones that connect to the microcontroller's USART interface. Table 3 shows the pin assignment for the Bluetooth module.

Table 3: Bluetooth Pin Assignment

Name	Type	Function
TX	Output	Serial transmit line
RX	Input	Serial receive line
Status	Output	Bluetooth connection status
Mode	Input	Command/Data mode toggle
On/Off	Input	Power eb505 up or down

The transmission rate between the local serial port on the TX/RX ranges between 9.6 k and 230.4 kbps. The over-the-air transmission rate ranges between 9.6 k and 115.2 kbps. Two operating modes are possible: command mode and data mode. In command mode, serial commands can be sent to configure the Bluetooth module. Once a connection is

established between two end points, the module automatically switches over to the data mode. In this mode, all data sent along serial port will transmit to the remote device. The Status line can then be used to monitor the connection status.

Please refer to the Appendix for the interconnection between the Bluetooth module and the microcontroller.

3.1.12. I/O Expander Interconnection

The I/O expander uses the I^2C bus for communication with the microcontroller. The I/O ports are read from and written to all together through the I^2C bus. Figure 7 shows the block diagram of the I/O expander while Table 4 summarizes the pins of interest and their functionalities.

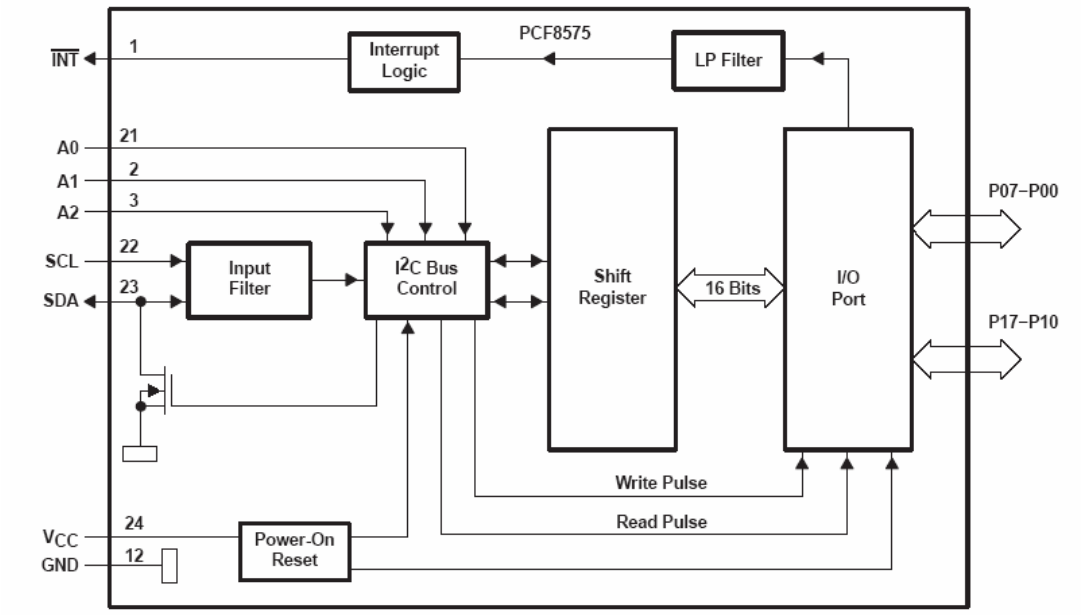


Figure 7: I/O Expander Block Diagram

Table 4: I/O Expander Pin Assignment

Name	Type	Function
!INT	Output	Interrupt line
A0~A2	Input	Device address setting
P00~P17	Input/Output	Input/Output ports
SCL	Input	Serial Clock
SDA	Input/Output	Serial Data

The !INT line is connected to the INT0 pin on the microcontroller, which detects external interrupts. Whenever the input signal changes in logic level, the !INT line will be brought low until data is read from the I/O expander. An interrupt service routine is used to handle this process.

Please refer to the Appendix for detailed schematics of the interconnection between the I/O expander and the microprocessor.

3.1.13. Interconnection with Other Subsystems

The connection between the Music Player Subsystem and the ACS is established through the SPI and I^2C bus. The two bus interfaces are well-established standards so its operations will not be elaborated. The connection between the Light Controller Subsystem is done wirelessly via Bluetooth. Please refer to the section on the Bluetooth communication protocol for further details. The connection between the Pulse Monitoring Subsystem is done through the I/O expander for control signals, while the measured data comes in to the microcontroller through a GPIO pin that accepts external interrupt.

Please refer to the individual sections on each subsystem for details about interfacing with the ACS.

3.1.14. Power Supply

Power is provided to the ACS through an AC to DC converter that converts 120 VAC to 9 VDC. The adapter is Underwriters Laboratory, or UL certified, which reduces the amount of components that needs to be certified for the production model. Two voltage regulators, KA78R05C and KA78RM33 are used to convert the 9 V into 5 V and 3.3 V, respectively. Two supply rails are required because certain components require 5 V supply, whereas others require 3.3 V.

3.1.15. Physical Enclosure

To shield potential electrical shock from the user, while providing a rigid enclosure, an off-the-shelf plastic case will be used for the prototype model. Holes will be made for mounting the numeric and alphanumeric LCD panels, as well as input buttons. As specified in the functional specification, the LCD panels shall be mounted in the front of the enclosure. The directional button will also be mounted in the front. The Alarm Off and Light On/Off button will be placed on the top of the enclosure. Speakers will be mounted on the front of the case and shall be exposed such that maximum air flow can result from the speakers. Adhesive rubber feet will be used to isolate the plastic enclosure from the surface which it rests on. The rubber feet will also prevent the alarm clock from slipping when buttons are being pushed.

The production model will use a rigid plastic injection molded enclosure with no sharp edges or corners. The enclosure will also be flame retardant to act as a safety redundancy.

3.2. Music Player Subsystem Design

From the functional specifications, the SAC must be able to play audio files of Moving Picture Experts Group Layer 3 (MP3) format. The Music Player Subsystem (MPS) is responsible for MP3 processing as well as interfacing with non-volatile storage. The subsystem must also control the volume level, and increase the volume level gradually during the waking process.

The high level functional block diagram of the MPS is shown in Figure 8.

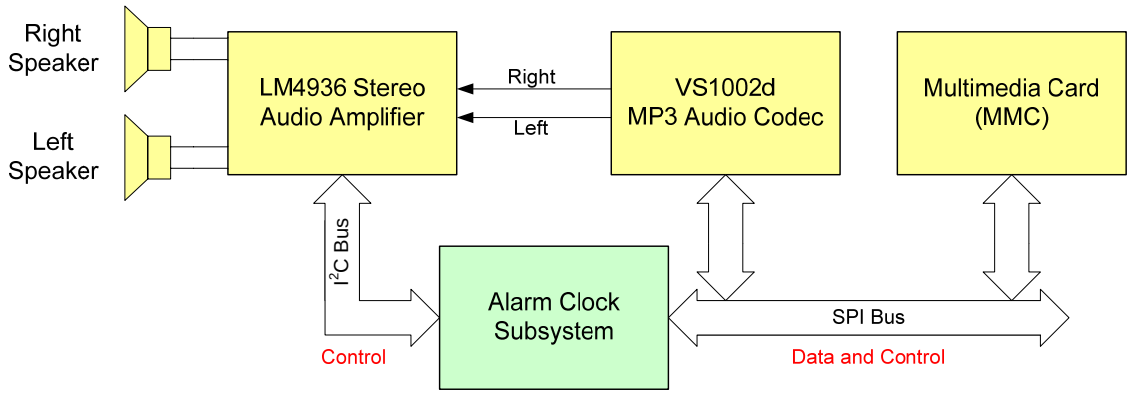


Figure 8: Functional Block Diagram of the Music Player Subsystem

There are three major components in the functional block diagram: the VS1002d MP3 Audio Codec, Multimedia Card (MMC) and LM4936 Stereo Audio Amplifier. The ACS interfaces with the MPS via two data and control buses, namely the I^2C and SPI bus. The ACS acts as the master of these two buses. The microprocessor within the ACS reads data from the MMC and passes it to the MP3 Audio Codec for MP3 decoding. The decoded signal constitutes the Left and Right analog audio signals, which are then passed on to the Stereo Audio Amplifier for amplification. The microprocessor can also control the amount of amplification provided in discrete steps. The amplified signal then drives the two speakers, constituting the MPS.

3.2.1. VS10xx Prototyping Board

The foundation of the MPS is the VS10xx Prototyping Board by VLSI Solution Oy, shown in the figure below.

Figure 9: VS10xx Prototyping Board

Overall, the prototyping board will be responsible for audio performances such as MP3 decoding, audio amplification, digital volume control, and audio output to external speakers. Another responsibility of the board is to properly communicate with the ACS to determine when audio is played.

The prototyping board incorporates an MP3 Audio Codec that can decode MP3 audio files of various sampling rates along with a MMC connector for hot-removal and insertion. Onboard power regulation ensures that all chips connected are operating with the correct voltage level. The board also provides a headphone jack with proper circuitry to drive an earphone. An onboard EEPROM contains the boot code required to allow the board to function as a standalone MP3 player.

This prototyping board was chosen to reduce the amount of components we need to assemble manually, since most of the audio circuitry is available for us. Moreover, header pins are available to easily interface with the microprocessor.

3.2.2. VS1002d MP3 Audio Codec

The prototype board incorporates the VS1002d MP3 Audio Codec shown in Figure 10. The codec is capable of decoding Moving Picture Experts Group (MPEG) 1.0 and 2.0 audio layer III files, with sampling rates of 48 kHz and bitrate up to 224 kbps. The single-chip solution includes a high-quality stereo digital to analog converter (DAC), eliminating the need for a separate DAC, and an earphone amplifier for direct driving of a 30Ω load. Digital volume control allows the external microprocessor to control the volume level.

Figure 10: VS1002d MP3 Audio Codec

Most MP3 decoders that are available on the market require a separate DAC, which increases part count and complexity. Having a single-chip solution eliminates many problems inherent to digital to analog conversion. The chip also contains an on-chip digital signal processor (DSP) which can be used for audio equalizer.

3.2.3. MMC

The non-volatile storage of choice is the Multimedia Card format. MMC is a widely used flash memory that is very compact (about the size of a postage stamp). Although MMC has been superseded by Secure Digital (SD) cards, they are still compatible with new devices supporting SD. The size of MMC ranges from 32MB to 2GB, which roughly translates to storing 8 to 500 songs.

Figure 11 shows our MMC of choice, the Kingston 128MB MMC. The MMC provides non-volatile storage for holding the user's MP3 files, as well as providing external storage for the Alarm Clock subsystem in general. The user can load his or her favorite songs via the MMC by using an external computer with a suitable card reader/writer.

Figure 11: Kingston 128MB MMC

3.2.4. LM4936 Stereo Audio Amplifier

To ensure enough power is available for driving the stereo speakers, the LM4936 Stereo Audio Amplifier is used. It is capable of providing 1.25 W of power into an 8 Ω load, which is the typical impedance for a speaker. The chip provides volume control via the I^2C control interface with the external microprocessor, with 31 adjustable levels. The "click and pop" suppression circuitry on-chip minimizes turn-on and shutdown transients. As specified in the functional specification, the alarm clock must begin at a barely audible sound level. Thus, proper transient suppression is critical in our design, and the LM4936 chip fulfills this requirement.

3.2.5. Speakers

A pair of speakers is used to produce stereo-quality sound. The speakers have an impedance of 8 Ω and a nominal power rating of 1 W. The frequency response of the speakers is between 300 Hz to 20 kHz, which is almost the typical audible range for humans [16]. As mentioned in our functional specification, the speakers must be able to produce a sound pressure level of 80 dbSPL. Since human ears are most sensitive to sounds between 1 kHz to 3 kHz [16], the speakers of choice must also be capable of producing 80 dbSPL between this range.

The speakers of choice are manufactured by KOBITONE. Figure 12 shows the sound pressure level of the speaker in dbSPL versus sound frequency. Except for the roll-off at 3 kHz, the speaker otherwise satisfies the aforementioned criteria.

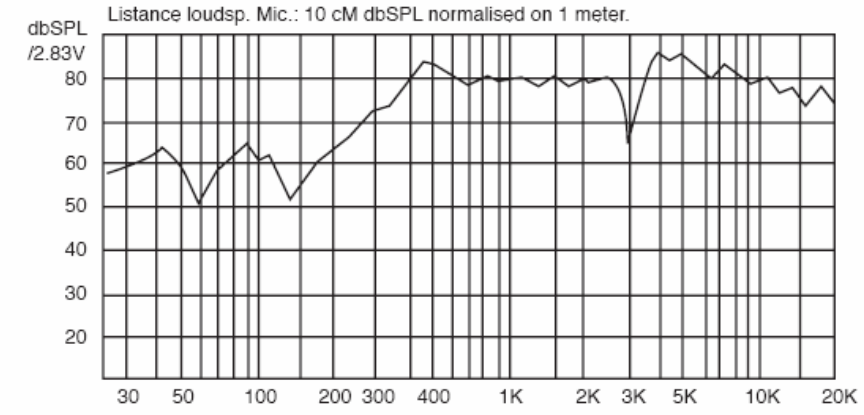


Figure 12: Sound Pressure Level (dbSPL) versus Frequency Response [8]

3.2.6. Interface to the MMC

The MMC has 7 pins. The pin assignments are summarized in Table 5, with the pin assignment on the card shown in Figure 13.

Table 5: MMC Pin Assignment

Pin	Name	Type	Function
1	CS#	Input	Chip Select (active low)
2	Din	Input	Data Input
3	VSS1	Power	GND
4	VDD	Power	VCC
5	CLK	Input	Clock Input
6	VSS2	Power	GND
7	Dout	Output	Data Output

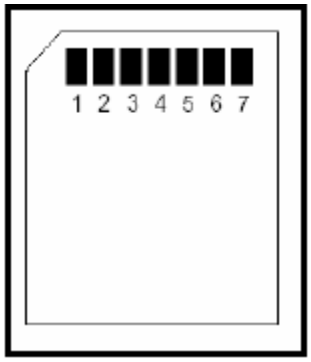


Figure 13: MMC Pin Assignments

MMC has two modes of operation: MMC mode and SPI mode. Many microprocessors have on-chip support for SPI, making it the ideal mode of choice for integration. The MMC is selected by the microcontroller whenever the CS# pin is pulled low. Data is subsequently clocked in and out of the device through the Din and Dout pins, respectively, by clock pulses sent through the CLK pin.

3.2.7. MP3 Decoding

MP3 decoding is achieved by using the VS1002d chip. The block diagram of the chip is shown in Figure 14, and the pins of interest are summarized in Table 6.

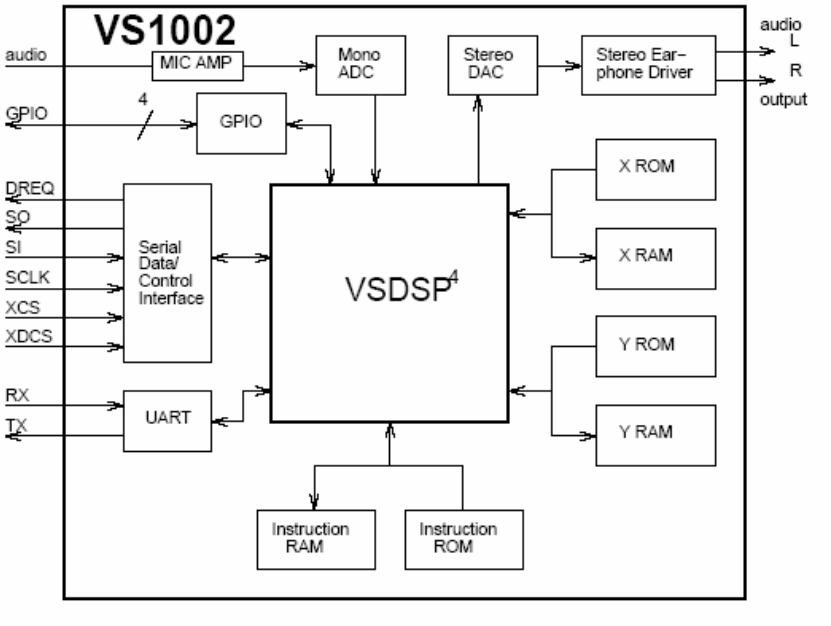


Figure 14: VS1002d Functional Block Diagram

Table 6: VS1002d Pin Assignment

Name	Type	Function
DREQ	Output	Data Request
SO	Output	Slave Output
SI	Input	Slave Input
SCLK	Input	SPI Clock
XCS	Input	Control Chip Select
XDCS	Input	Data Chip Select

The Serial Data/Control Interface allows the chip to be interfaced through the SPI bus. Control signals are exchanged by pulling XCS low and transmitting through SI for control input to the chip and SO for control output from the chip. This is known as the Serial Control Interface (SCI). Similarly, data can be exchanged by pulling XDCS low and transmitting through SI for data input to the chip. This is known as the Serial Data Interface (SDI). Both interfaces use the SCLK signal to control the clocking of the serial bus.

The VS1002d chip contains a First-In First-Out (FIFO) buffer for storing incoming audio data that needs to be decoded. Thus, audio data to the chip can be written in batches to fill the buffer. There is a 32-byte safety area on the buffer. Whenever this safety area is empty, the DREQ line is high to signal to the microprocessor that it can handle at least 32-bytes more audio data. DREQ is low when this safety area is non-empty. The FIFO buffer and DREQ line prevents the need for the microprocessor to constantly poll the decoder chip. Instead, an interrupt can be used.

3.2.8. Audio Amplification

After the MP3 is decoded by the VS1002d chip, the output is an analog signal that is only suitable for directly driving headphones and not speakers. Thus an audio amplifier is required to drive the speakers. This is accomplished by using the LM4936 Stereo Audio Amplifier. Figure 15 shows the functional block diagram while Table 7 shows the pins of interest for this chip.

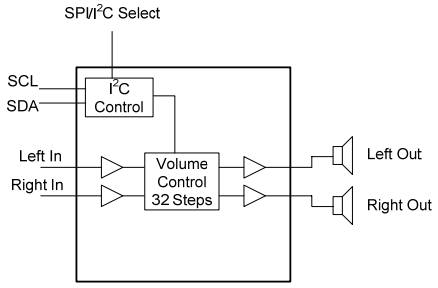


Figure 15: LM4936 Functional Block Diagram

Table 7: LM4936 Pin Assignment

Name	Type	Function
SPI/I^{2}C Select	Input	Select between I^{2}C and SPI
SCL	Input	Serial Clock
SDA	Input	Serial Data
Left In	Analog Input	Left Audio Channel
Right In	Analog Input	Right Audio Channel

The analog audio signal from the MP3 chip arrives through the Left In and Right In pins. The SPI/I^{2}C Select pin is used to select between controlling the device using the I^{2}C or SPI bus. Since the SPI bus is in use by the VS1002d chip and the MMC, for which both reside on the VS10xx prototyping board, it was decided that the I^{2}C bus be used instead. This decision was also due to the heavy amount of data passing through the SPI bus during MP3 playback, and further loading can be avoided by placing the LM4936 on the I^{2}C bus. Control is achieved through sending commands serially on the SDA pin clocked by SCL.

The amount of amplification can be controlled through 32 steps, with two of these steps corresponding to the same level of amplification. These steps are not linear adjustments but correspond to logarithmic scaling (dB) between -90 dB for total silence to 0 dB, or maximum amplification.

3.2.9. Interconnection with Alarm Clock Module

In order to use the VS10xx prototyping board, which contains the VS1002d chip and the MMC, wire modification was required both on the prototype board, as well as the modifying the 40 pin header (JP1) on board. Fortunately jumpers are available on the board to facilitate this process. The jumpers that were removed are summarized in Table 8, while Table 9 shows the connections to the microcontroller. Please refer to the Appendix for the schematic of the VS10xx prototyping board for further details.

Table 8: Summary of Jumpers Removed on VS10xx Prototyping Board

Jumper	Reason
JP6, 7, 16	Prevent the onboard buttons from interfering with the SPI bus.
JP8	Prevent DREQ line from interfering with the SPI bus.
JP10	Removed jumper to prevent loading the boot code on the EEPROM for standalone player operation.
JP15, 17	Remove the connection between the MMC and various GPIO pins on the VS1002d chip.

Table 9: VS10xx Prototype Board Connection to Microcontroller

Jumper#[pin#]	Name	Connection on Microcontroller
JP1[8]	XRST	PB1
JP1[27]	SO	PB6 (MISO)
JP1[28]	CS#	PB0
JP1[29]	SI	PB5 (MOSI)
JP1[31]	SCLK	PB7 (SCK)
JP1[32]	CLK	PB7 (SCK)
JP1[33]	XCS	PB4
JP1[35]	XDCS	PB3
JP1[37]	DREQ	PB2 (INT2)
JP17[2]	DI	PB5 (MOSI)
JP17[4]	DO	PB6 (MISO)

For the connection between the LM4936 Stereo Amplifier and the microcontroller, please refer to the schematic in the Appendix for more details.

3.3. Light Control Subsystem Design

The Light Control Subsystem (LCS) is a physically standalone device that is responsible for controlling the brightness of an external lamp. The LCS communicates wirelessly with the ACS which commands the LCS appropriately during different system states. The LCS then responds by varying the AC power supplied to the external lamp.

The high level functional block diagram of the LCS is shown in Figure 16.

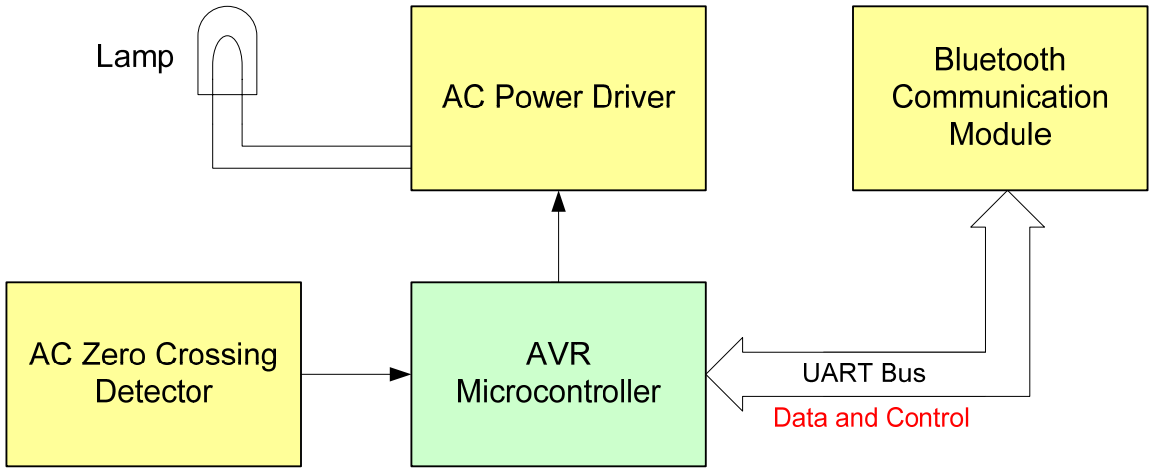


Figure 16: Functional Block Diagram of the Light Control Subsystem

The LCS system composes of four major functional blocks: the AVR Microcontroller, Bluetooth Communication Module, AC Zero Crossing Detector, and AC Power Driver. Since the LCS is a standalone device, an AC/DC Power Supply is included in the system architecture which is not shown in the functional block diagram. The microcontroller is the heart of the system; it handles the communication with the ACS, the monitoring of the AC line, and the AC output voltage of the external lamp. The Bluetooth Communication Module is connected to the microcontroller by the Universal Asynchronous Receiver Transmitter (UART) bus; the AC Zero Crossing Detector and AC Power Driver are optically isolated and interfaced through general I/O pins.

3.3.1. Light Intensity Control

The technique of AC phase angle control is used to control the intensity of the external lamp. This effective technique is widely used and can be implemented using a microcontroller with relatively little resources. By varying the root mean square (RMS) power delivered to the lamp, the intensity of the lamp can be varied. Phase control will be

used to control the conduction angle of the AC sinusoidal; this results in a change in the applied RMS voltage and power.

In order to control the conduction angle, the zero crossings of the AC sinusoidal are identified and used as time references. This is achieved through the AC Zero Crossing Detector. The AC sinusoidal is full-wave rectified to act as an input to an opto-isolator; the output is connected to the microcontroller's external interrupt. At zero crossings, the output will transition from low to high and thus interrupts the microcontroller.

The microcontroller can determine the frequency of the AC sinusoidal using the zero crossing interrupts and an onboard timer. Along with the commands received from the ACS, the turn-on time of the AC Power Driver can be calculated to give the desired conduction angle. The AC Power Driver provides optical isolation between the AC and DC circuits and it acts as an electronic switch that allows the microcontroller to switch on and off the AC power delivered to the external lamp.

3.3.2. Wireless Protocol with Alarm Clock Module

The ACS must be able to wirelessly communicate with the LCS to control the gradual increase in light intensity during the waking process. The Bluetooth communication module in use is the same as the one in the ACS, namely EmbeddedBlue eb505. Please refer to the section under the ACS for further details about interfacing with the microcontroller since the methodology is the same. The Appendix contains the detailed schematic of the interface between the eb505 module and the microcontroller.

Commands such as turning the external light on or off, and gradual increase in light intensity is sent over the air. The format of the communication protocol is summarized later in Section 5.4.

3.3.3. AC/DC Power Supply

To power all of the DC electronics in the LCS, a regulated AC to DC power supply is used. It follows the design of a linear power supply as efficiency is not the main concern. A step-down transformer converts the 120 VAC to 9 VAC; this voltage is then full wave rectified and filtered. To accommodate changes in load and line conditions, a linear voltage regulator is used to provide the 5 VDC needed to power the subsystem.

3.3.4. Physical Enclosure

Due to the high voltages and power present in the LCS, proper electrical safety guidelines must be followed to prevent the risk of fire and electrical shock. The enclosure must present a barrier between foreign objects and the high potential circuits. Furthermore, this must be achieved such that proper ventilation is maintained as the AC/DC Power Supply and AC Power Driver circuits dissipate heat under normal operation.

In the production unit, a rigid plastic injection molded enclosure with no sharp edges and corners will be used for obvious safety reasons. Vents will be designed on three faces of the enclosure to provide adequate ventilation. The plastic used must also be flame retardant and resistant to high temperature that may be produced by the AC Power Driver under heavy load conditions. The enclosure will be small and compact with a 3-plong plug on the back face for connecting to the AC outlet and a 3-plong outlet on the front face for connecting the external lamp. The enclosure is envisioned to be very similar to a "wall wart" with a 3-plong outlet; thus it must be small and compact to avoid blocking other devices from connecting to the other available AC outlet.

3.4. Pulse Measurement Subsystem Design

The Pulse Measurement Subsystem (PMS) is responsible for measuring the user's pulse and sending this information back to the ACS for determining the best sleep stage to wake the user. It must obtain accurate pulse measurements in a non-intrusive manner; the device must be light, compact, and comfortable. To achieve all of these objectives, a semiconductor sensor will be used to determine the pulse rate base on the varying light absorption by the user's body.

The high level functional block diagram of the PMS is shown in Figure 17.

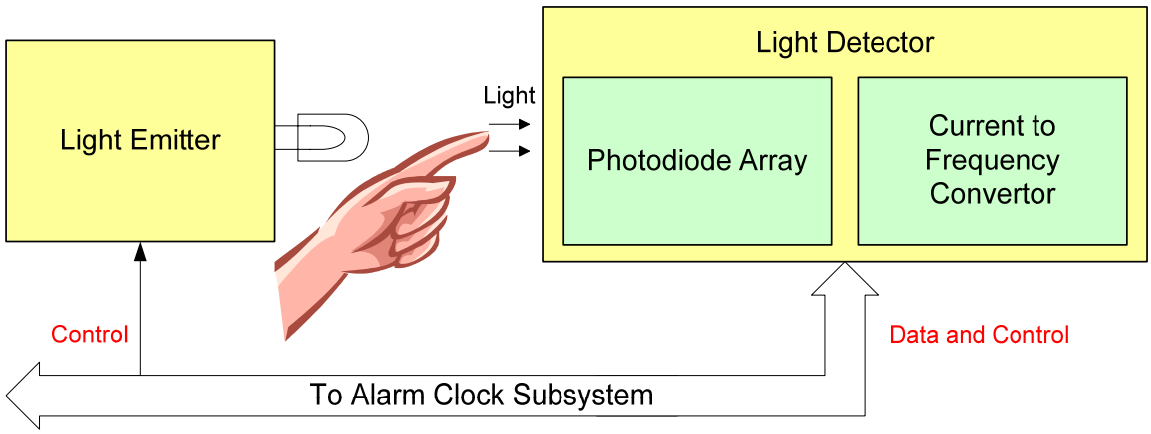


Figure 17: Functional Block Diagram of the Pulse Measurement Subsystem

The PMS composes of 2 major functional blocks: the Light Emitter (LE) and Light Detector (LD). LE is a constant intensity light source with a peak wavelength of 660 nm and LD provides a pulse train output with varying frequency as a function of the incident light intensity. Using a sensor with a digital output avoids the need for sensitive analog circuitry and makes the PMS compact.

The emitter is placed on one side of the user's finger and the detector is placed on the other. The user's bones and tissues absorb light at a constant level, but a variable absorption level is present due to blood flow. Since the light intensity is constant, the

changing absorption level causes different light intensity on the photodiode array, which results in a varying frequency output from the LD. By sampling this frequency output, the user's pulse rate can be determined by the ACS. Since pulse rates have a relatively slow frequency, the output of the LD does not need to be sampled at a high rate. To measure a pulse rate as high as 200 beats per minute as stated in the functional specifications, the Nyquist frequency is 6.67 Hz. Thus, the sampling approach is a viable method to determine the user's pulse rate without a heavy burden on the microcontroller present in the ACS.

3.4.1. Light to Frequency Conversion

To supply the needed light source for sampling the light absorption by the user's body, the LE is a constant current source driving a super bright light emitting diode (LED). The constant current source ensures the LED emits constant intensity light and it gives the ACS ability to turn on/off the LED.

In order to simplify the design and maintain the compactness and light weight of the PMS, a monolithic light to frequency solution is used to implement the LD. The chosen integrated circuit is the TSL230R by Texas Advanced Optoelectronic Solution, it encompasses a photodiode array, a current to frequency converter, and various control logic all within an 8 pin package.

When photons strike the photodiode array of the TSL230R, a current is generated within the photodiodes array. The magnitude of the current is proportional to the amount of incident light on the array. This current is then amplified by the TSL230R and converted into a pulse train. The frequency of the pulse train is dependent on the current which in turn depends on the light intensity striking the sensor.

Aside from the basic functionality, the sensor also has 4 control inputs (S0-S3) and an output enable (OE#), the functions of these pins are shown in Table 10.

Table 10: Terminal Functions of TSL230R

Terminal Name	Description
S0, S1	Photodiode array sensitivity select inputs
S2, S3	Output frequency scaling select inputs
OE#	Output enable (active low)

The photodiode array sensitivity can be selected from power down, 1x, 10x, and 100x modes and the output frequency scaling can be selected from /1, /2, /10, and /100 modes. The output enable pin can be used to place the output pin in a high-impedance state.

3.4.2. Interconnection with Alarm Clock Module

Interfacing the PMS with the ACS requires a total of 6 signals: 5 control lines and an output line. The control lines are used to turn on/off the LE, and select the sensitivity and output frequency scaling of the LD. Since the OE# pin of the TSL230R is always grounded, the output of the LD is always active.

Since the control lines do not require relatively high speed and changes seldom occur, these lines are interfaced to the Alarm Clock Subsystem's microcontroller through the I^2C I/O Expander then voltage translated from 3.3V to 5V. This approach reserves the microcontroller's general I/O pins for other more timing critical applications. However, the output of the LD is connected to the microcontroller's external interrupt pin (INT1) after being voltage translated from 5V to 3.3V. The edges of the LD's pulse train output will interrupt the microcontroller and start a timer to measure the period of the output. This measurement is representative of the incident light on the sensor and can be used to determine the heart rate of the user.

In terms of timing, there are no stringent requirements on the control lines as all the control signals are level sensitive. Furthermore, the frequency output of the LD should present no timing issues for the microcontroller during the period measurement. The only concern is the finite amount of time needed for the LD output to response to the new sensitivity and frequency scaling settings. According to the TSL230R datasheet, the output is valid after 2 periods plus 1 us after a settings change. Thus, a delay must be added before a valid sample can be taken from the PMS when settings are changed through the control lines.

3.4.3. Device Conduit Design

As the physical user interface with the pulse measurement subsystem, the device conduit slips onto the user's finger in a non-intrusive manner. Figure 18 provides a conceptual drawing of the pulse measurement subsystem in use.

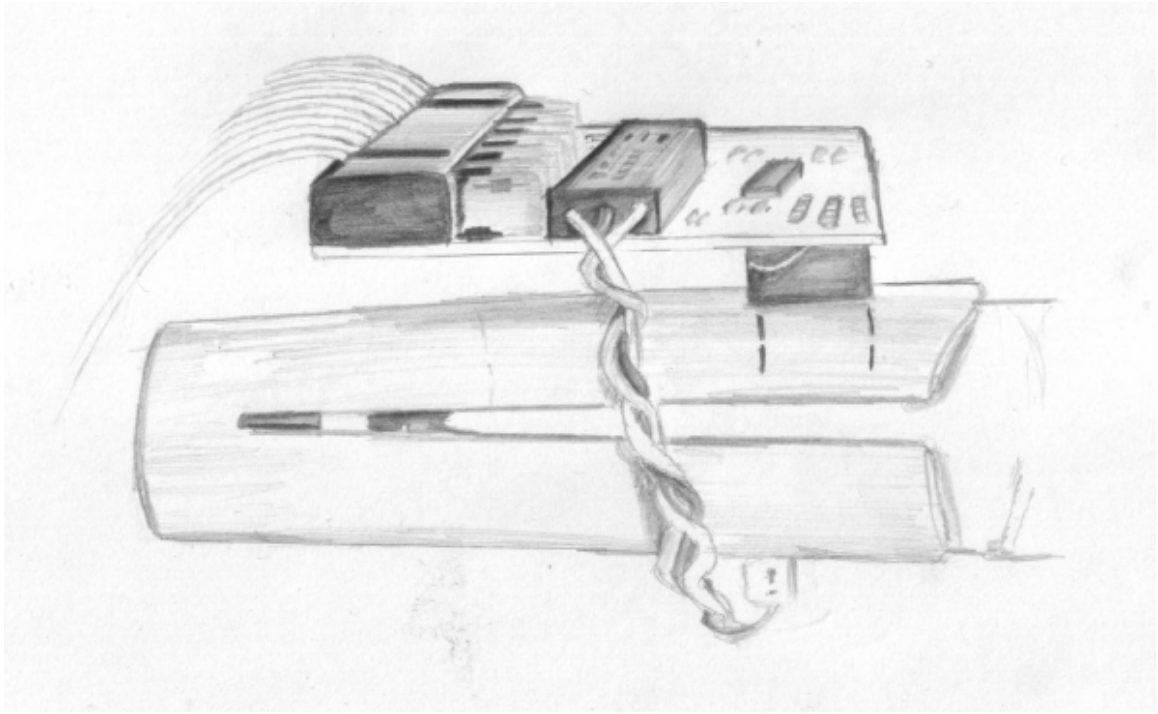


Figure 18: Pulse Measurement Subsystem in Use

A plastic casing with the light emitter mounted on top and the light detector attached to the bottom in the orientation shown slips through the user's finger. Two side openings are designed to provide a secured fit and allow for ventilation. A thin layer of polyethylene foam sheeting lined in the inner circumference of the plastic casing serves to improve comfort and prevent external light from interfering with the pulse measurement process. A detailed orthographic drawing of the conduit is shown in Appendix G.

4. Firmware Design

Firmware design serves as the glue logic that allows higher level software to communicate with the hardware components. Topics covered in the firmware design include basic functions and configurations applicable to a particular hardware.

4.1. Microcontroller

At a low level, the microcontroller communicates with the various hardware components using standardized interfaces such as I^2C, SPI, USART and GPIO. Appropriate pin configurations are performed at the register level for each interface. Low level functions are written to handle the proper signal timing as outlined on the datasheet of each hardware component. For brevity such details will not be elaborated.

Code of all level, including both firmware and upper level software, are developed using C programming language. Atmel provides an Integrated Development Environment (IDE) known as AVR Studio. C code is compiled using GNU C Compiler (GCC), an open source compiler. The IDE allows compiled code to be loaded onto the microcontroller through the RS232 serial interface using a development kit. Code is stored in the non-volatile Flash and EEPROM memory on the ATmega32 chip. The consequence of using embedded systems is that highly-efficient code be developed that is both fast and small.

4.1.1. Alphanumeric LCD

The alphanumeric LCD supports a wide range of instructions. Instructions are sent from the microprocessor through the LCD data and control bus as mentioned in the hardware section. Once the data and control values are successfully latched by the alphanumeric LCD, the instruction begins execution. Commands have different execution time and during this time no other instructions can be issued. The device can be polled to see if the next instruction can be sent, or the maximum execution time can pass before sending the next instruction.

Figure 19 shows the list of instructions that can be performed by the alphanumeric LCD. Setup and initialization of the LCD is done once. The Function Set instruction can set the LCD to operate in 4-bit data bus width mode, reducing the total number of lines needed on the microcontroller at the cost of writing the desired function to the bus twice. Once the desired initialization settings are set, characters can be written to the screen through the Write Data to RAM instruction, which writes to the desired DDRAM address specified by the Set DDRAM Address instruction. Figure 20 shows a list of supported characters that can be written.

Instruction	Instruction Code										Description Instruction Code	Execution time (fsoc=270kHz)
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		
Clear Display	0	0	0	0	0	0	0	0	0	1	Write "20H" to DDRAM. and set DDRAM address to "00H" from AC.	1.53ms
Return Home	0	0	0	0	0	0	0	0	1	X	Set DDRAM address to "00H" from AC and return cursor to its original position if shifted. The contents of DDRAM are not changed.	1.53ms
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	SH	Assign cursor moving direction and make shift of entire display enable.	39μs
Display ON/OFF Control	0	0	0	0	0	0	1	D	C	B	Set display(D), cursor(C), and blinking of cursor(B) on/off control bit.	39μs
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	X	X	Set cursor moving and display shift control bit, and the direction, without changing DDRAM data.	39μs
Function Set	0	0	0	0	1	DL	N	F	X	X	Set interface data length (DL : 4-bit/8-bit), numbers of display line (N : 1-line/2-line), display font type(F : 5 X 8 dots/ 5 X 11 dots)	39μs
Set CGRAM Address	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	Set CGRAM address in address counter.	39μs
Set DDRAM Address	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Set DDRAM address in address counter.	39μs
Read Busy Flag and Address	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Whether during internal operation or not can be known by reading BF. The contents of address counter can also be read.	0μs
Write Data to RAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0	Write data into internal RAM (DDRAM/CGRAM).	43μs
Read Data from RAM	1	1	D7	D6	D5	D4	D3	D2	D1	D0	Read data from internal RAM (DDRAM/CGRAM).	43μs

Figure 19: List of Alphanumeric LCD Instructions and Execution Time [10]

Figure 20: List of Characters Supported by the Alphanumeric LCD [10]

User-defined characters can be loaded onto the LCD for use through the Write Data to RAM instruction and the Set CGRAM Address instruction. This allows us to define special characters for use.

4.1.2. Numeric LCD Driver

The numeric LCD driver latches the bits on the LCD data bus whenever !CS1 pin transitions from low to high. The four bit latched data corresponds to an encoding of the numeric digit that is to be displayed. Two types of encoding schemes are available, as Hexadecimal or Code B. Figure 21 shows the list of characters available for display on the numeric LCD screen and the corresponding binary encoding.

BINARY				HEXADECIMAL ICM7211(M) ICM7212(M)	CODE B ICM7211A(M) ICM7212A(M)
B3	B2	B1	B0		
0	0	0	0	0	0
0	0	0	1	1	1
0	0	1	0	2	2
0	0	1	1	3	3
0	1	0	0	4	4
0	1	0	1	5	5
0	1	1	0	6	6
0	1	1	1	7	7
1	0	0	0	8	8
1	0	0	1	9	9
1	0	1	0	A	-
1	0	1	1	b	E
1	1	0	0	C	H
1	1	0	1	d	L
1	1	1	0	E	P
1	1	1	1	F	(Blank)

Figure 21: List of Numeric Characters Available on Seven-Segment Display [13]

4.1.3. Bluetooth Module

Commands sent to the Bluetooth module is achieved by writing American Standard Code for Information Interchange (ASCII) characters. The module initially starts up in "command mode". The prompt character (>) is provided by the module whenever it is able to accept a command or data. Command strings with parameters are then written to the module as ASCII characters followed by the carriage return (<CR>) character. Commands are acknowledged as correct by returning "ACK<CR>" character string, or incorrect by returning "NAK<CR>Err *number*<CR>" where *number* is a numeric value that corresponds to the error. Following the acknowledgement, ">" is issued again. Table 11 shows a list of common commands for the Bluetooth module along with the proper syntax.

Table 11: Bluetooth Module Common Commands and Syntax

Name	Syntax	Function
Connect	**con** *address*<CR>	Establishes a connection to the other Bluetooth device, where *address* is the 48-bit address in hexadecimal.
Disconnect	**dis**<CR>	Closes connection with the other Bluetooth device.
List Visible Device	**lst** visible<CR>	Returns a list of all the devices that is currently in range and visible.

Once a connection is established to the other device, the module enters "data mode". Characters can be sent to the other device when the prompt ">" character is available. Commands and responses as defined in the Bluetooth Communication Protocol Section in Section 5.4 can be sent through this established link.

4.2. MP3 Audio Codec

The SCI on the MP3 Audio Codec allows control information to be read from or written to the device. The serial bus protocol for SCI consists of an instruction byte, address byte, and one 16-bit data word. Data bits are clocked in by the device at the rising edge of the serial clock. Data bits are also sent most significant bit (MSB) first.

The instruction byte can either be Read Instruction (0x3) or Write Instruction (0x2). The address byte specifies the register address of interest to read from or write to. The registers of interest on the MP3 Audio Codec are summarized in Table 12. The size of each register is 16 bits.

Table 12: Registers of Interest on MP3 Audio Codec

Register Name	Address	Type	Description
MODE	**0x0**	rw	Mode control
DECODE_TIME	**0x8**	r	Decode time in seconds
VOL	**0xB**	rw	Volume control

The contents of interest on the MODE register are summarized in Table 13. The chip can be held in soft reset, or can be powered up or down via the SCI by configuring the appropriate bits in MODE register. The SDI can also be configured to function as MSB first or MSB last, with either rising or falling edge trigger.

Table 13: Register Contents of Interest for MODE

Register[bit]	Function	Value	Description
MODE[2]	Soft reset	0 1	No Reset Reset
MODE[4]	Power down	0 1	Power On Power Off
MODE[8]	DCLK active edge	0 1	Rising Edge Falling Edge
MODE[9]	SDI bit order	0 1	MSB First MSB Last

The DECODE_TIME register contains the current decoded time in seconds for the particular file being played. This value can be read from and updated to the LCD screen for the purpose of keeping track of the song's play time in the production model.

The VOL register allows the analog volume to change in steps of 0.5 dB attenuation from the maximum volume. Both left and right analog channels can be individually configured for different attenuation level. A range of 256 steps can be configured which provides 128 dB attenuation. The VOL register value is set to:

*(Left Channel Volume in 0.5 dB steps) * (256) + (Right Channel Volume in 0.5 dB steps)*

4.3. MMC

The MMC communicates with the microcontroller using the SPI interface. The MMC is selected when CS# is pulled low, and data to the MMC is transmitted on Din pin while data from the MMC is transmitted on Dout pin. Commands sent to the MMC are all six bytes long and are transmitted MSB first. A start bit of value 0 and a transmission bit of value 1 is transmitted, followed by 6 bits of command, 32 bits of argument, 7 bits of Cyclic Redundancy Check (CRC) and an end bit of value 1. Figure 22 illustrates the command format of the MMC, while Figure 23 shows the most commonly used commands.

Bit position	47	46	[45:40]	[39:8]	[7:1]	0
Width (bit)	1	1	6	32	7	1
Value	0	1	X	X	X	1
Description	Start bit	Transmission bit	Command index	Argument	CRC7	End bit

Figure 22: MMC Command Format

CMD INDEX	ARGUMENT	RESPONSE	ABBREVIATION	COMMAND DESCRIPTION
CMD0	None	R1	GO_IDLE_STATE	Resets the MultiMediaCard
CMD1	None	R1	SEND_OP_COND	Activates the card Initialization process
CMD13	None	R2	SEND_STATUS	Asks the selected card to send its status register
CMD16	[31:0]block length	R1	SET_BLOCKLEN	Selects a block length (in bytes) for all following block commands (read and write).
CMD17	[31:0]data address	R1	READ_SINGLE_BLOCK	Reads a block of size selected by the SET_BLOCKLEN command
CMD24	[31:0]data address	R1	WRITE_BLOCK	Writes a block of the size selected by the SET_BLOCKLEN command
CMD32	[31:0]data address	R1	TAG_SECTOR_START	Sets the address of the first sector of the erase group
CMD33	[31:0]data address	R1	TAG_SECTOR_END	Sets the address of the last sector in a continuous range within the selected erase group, or the address of a single sector to be selected for erase.
CMD34	[31:0]data address	R1	UNTAG_SECTOR	Removes one previously selected sector from the erase selection
CMD38	[31:0]don't care	R1b	ERASE	Erases all previously selected sectors
CMD59	[31:1]don't care [0:0]CRC option	R1	CRC_ON_OFF	Turns the CRC option on or off. A '1' in the CRC option bit will turn the option on. A '0' will turn it off.

Figure 23: Most Commonly Used MMC Commands

Commands are followed by a response from the MMC, which can either be of type R1 or type R2 response as illustrated above. Depending on the type of command executed, the data blocks to read from or write to the MMC immediately follows the response. Figure 24 shows the transfer flow of the MMC protocol.

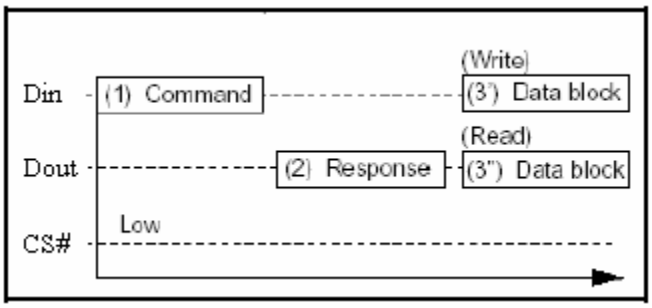


Figure 24: MMC Transfer Flow

SET_BLOCKLEN function can be used to set the length of blocks to read from or write to the MMC. Since the File Allocation Table (FAT) file system will be used on the MMC, block lengths are set to 512 bytes. The data address argument allows the start of the 512 byte block to be specified.

4.4. Audio Amplifier

The audio amplifier is connected with the microcontroller via the I^2C bus. Data is transmitted MSB first. When the device detects its own address on the serial data line, it accepts the next byte of data and writes it to one of two registers, Mode Control Register and Volume Control Register. The register contents and the bit contents of the audio amplifier are summarized in Table 14 and Table 15, respectively.

Table 14: Register Contents on the Audio Amplifier

	B7	B6	B5	B4	B3	B2	B1	B0
Mode Control Register	0	0	0	HP Control	Gain Select	Mode	Mute	Shutdown
Volume Control Register	1	0	0	V4	V3	V2	V1	V0

Table 15: Bit Contents on the Audio Amplifier

	Value	Description
HP Control	0 1	Configures the amplifier output stage
Gain Select	0 1	Internal Gain External Gain
Mode	0 1	Fixed Volume Adjustable Volume
Mute	0 1	Mute Off Mute On
Shutdown	0 1	Device Shutdown Device Active
V4~V0	00000~11111	Attenuation level

4.5. I/O Expander

The I/O Expander is connected with the microcontroller via the I^2C bus. Data is transmitted MSB first. When the device detects its own address on the serial data line, a write or read to the I/O Expander's ports occur. During a write, it accepts the next 2 bytes of data and writes it the output registers. On a read, it sends the status of the I/O Expander's ports to the microcontroller in the next 2 bytes. The definition of the interface is summarized in Table 16.

Table 16: I/O Expander Interface Definition

	B7	B6	B5	B4	B3	B2	B1	B0
P0 I/O Data Bus	P07	P06	P05	P04	P03	P02	P01	P00
P1 I/O Data Bus	P17	P16	P15	P14	P13	P12	P11	P10

5. Communication Protocols

One of the purposes of the proof-of-concept model for the Smart Alarm Clock is to demonstrate the possibility for wireless communication between the Alarm Clock Subsystem and the Light Control Subsystem in order to promote improved usage flexibility. A piconet representation of the operation is shown in Figure 25.

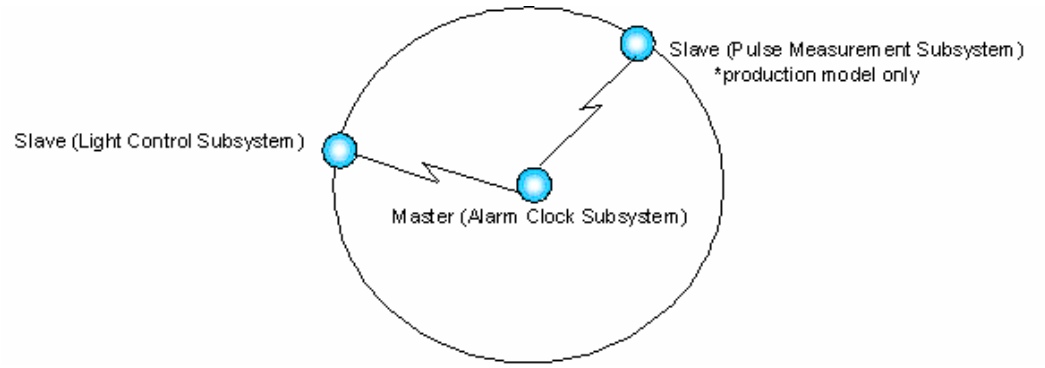


Figure 25: Point to Multipoint Piconet

To standardize data transmission between the Alarm Clock Subsystem and the Light Control Subsystem for the control of light brightness, we establish a wireless communication protocol.

5.1. Wireless Communication Requirements

A need for the Smart Alarm Clock to properly operate in as wide range an environment as possible motivates us to choose a mean of wireless communication technology transmitting through the license-free Industrial, Scientific and Medical (ISM) band. The communication method must be robust against other users and noise polluters of the shared spectrum.

Respective to power requirements, the communication technology is to consume minimal radio power in both the operation and standby states. A small bandwidth is required for the signal carrier, as data transmission primarily serves to adjust light brightness for the Light Control Subsystem with the proof-of-concept model. In addition, a short reception range in the order of meters is needed due to the nature of the operation.

5.2. Bluetooth Implementation

Bluetooth is chosen as a low cost, low power, short range transmission solution to our wireless communication needs. Operating at 2.4 GHz in the globally available, license-free ISM band, the Bluetooth devices communicates using the scheme of Frequency Hopping Spread Spectrum (FHSS) whereby devices continuously tune their radio from

one channel to another in order to reduce the influence of noise in the data transmission process. The eb505 module we utilize supports a sufficient maximum bidirectional data transfer rate of 230.4 kbps, with a coverage range of 10 meters.

Essential to the discussion of our wireless communication protocol design is the Bluetooth standard communication protocol stack. Figure 26 shows the Bluetooth stack alongside the Open Systems Interconnect (OSI) standard reference model to illustrate the division of responsibility amongst the different stack parts.

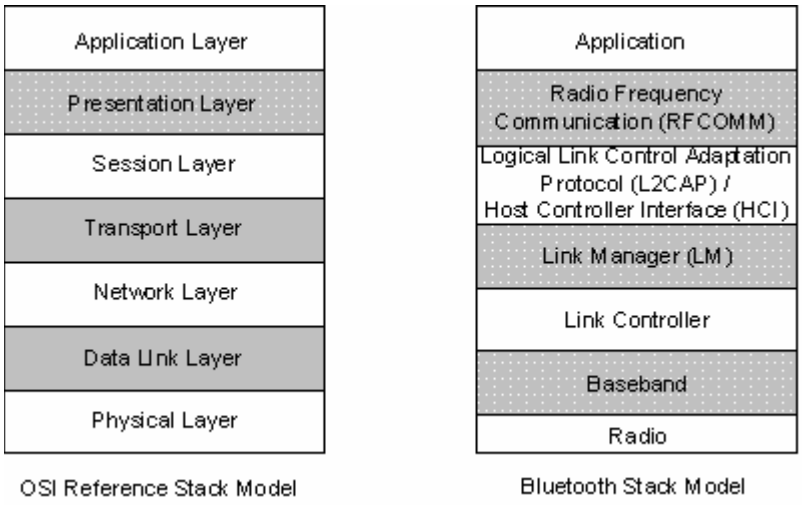


Figure 26: OSI Reference and Bluetooth Stack Models

Our current focus is on our implementation of wireless communication protocol between the Alarm Clock Subsystem and the Light Control Subsystem, which communicates to each other at the application layer. The eb505 module abstracts the device searching and paging procedures involving the Logical Link Control Adaptation Protocol (L2CAP) and the Radio Frequency Communication (RFCOMM) into the EmbeddedBlue command set in which we will employ to establish connection from the Alarm Clock Subsystem and the Light Control Subsystem.

Before the alarm clock and the Light Control Subsystems can communicate in a bidirectional manner, the alarm clock must first establish a Bluetooth link with the Light Control Subsystem.

The device discovery procedure is shown in Figure 27.

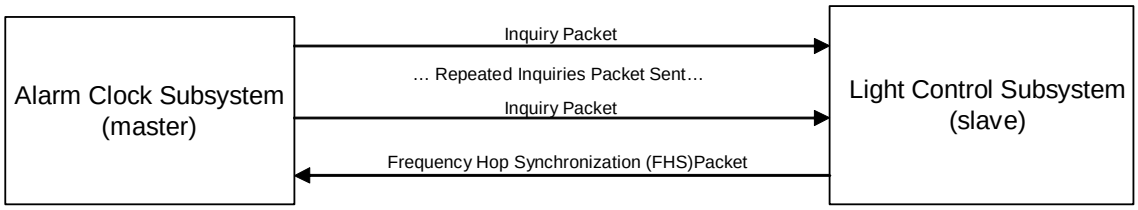


Figure 27: Device Discovery Procedures

As shown in Figure 27, the Alarm Clock Subsystem sends a series of inquiries packets until the Light Control Subsystem response with a Frequency Hop Synchronization (FHS) packet required for the Alarm Clock Subsystem to recognize the existence of the Light Control Subsystem. The eb505 radio module abstracts the synchronization into a sequence of command sending to be discussed in details in Section 5.3.

Once the two devices make recognition, a connection between the Alarm Clock Subsystem and the Light Control Subsystem can be established. Figure 28 illustrates a high level overview of the baseband paging procedures in setting up a link between the two modules in consideration.

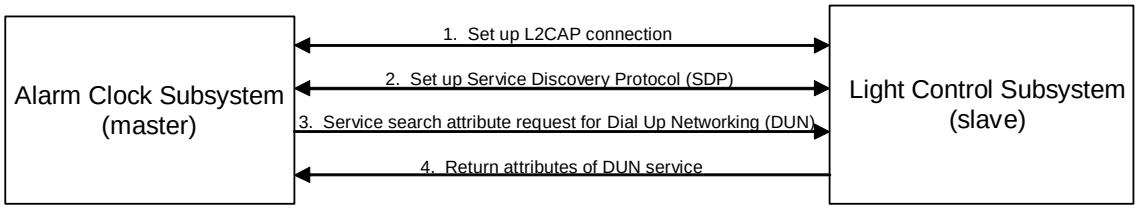


Figure 28: Baseband Paging Procedures

In Section 5.3, we explain the EmbeddedBlue command set used for linking the two master and the slave devices in greater details. A connection for data transmission is formally configured once the Baseband Paging Procedures completes. The Alarm Clock Subsystem can then command the Light Control Subsystem through the Bluetooth link.

5.3. EmbeddedBlue Command Set

The EmbeddedBlue command set provides abstraction to low level device discovery and baseband paging procedures through visible ASCII characters. Each command is issued directly from our embedded application program. A terminal application can also issue the EmbeddedBlue command set between Bluetooth enabled devices through a Bluetooth connection.

In the following subsections, we illustrate the basic structure of the available commands as issued by the eb505 radio module through a terminal application to focus on the

command set's functionalities. The commands are applied in the same manner when used in an embedded application program.

5.3.1. Search Local Device

Before the Alarm Clock Subsystem can connect to the Light Control Subsystem, the 48-bit IEEE address of the target device must be determined. This is done using the EmbeddedBlue command *lst visible*. The command returns a list of addresses of devices that are available for connection.

5.3.2. Check Device Mode

We can check if the Light Control Subsystem is functioning properly before a Bluetooth link is established. If the Light Control Subsystem experiences a fatal error and can not continue to function, it will set its connectable mode off and not accept connection from other Bluetooth devices. The command to prevent other Bluetooth devices to establish a connection is *set connectable off*. In a similar manner, when the Light Control Subsystem is ready for control by the smart alarm module, the Light Control Subsystem should execute *set connectable on*.

5.3.3. Establish Bluetooth Connection

At the point in which the Alarm Clock Subsystem acknowledges the light control device exists and is in a state of proper operation, a connection can be established between the two modules. To that end, we execute the command *con address timeout.* The *address* is the address of the device for which we wish to establish connection to, as returned from *lst visible*. The *timeout* parameter serves as an optional parameter for which abort a connection trial if the specified seconds of time have elapsed and no acknowledgement is obtained from the slave device.

5.4. Bluetooth Communication Protocol

With the Bluetooth linkage in place between the smart alarm module and the Light Control Subsystem, the smart alarm module can now transmit commands to the Light Control Subsystem and govern the light intensity. Our choice of the eb505 Bluetooth radio device is motivated in part because of the integrated UART.

The UART takes bytes of the data from the master transmission device and transfer the individual bits representing the byte. A decoder on the slave transmission device reassembles the bits into bytes. Thus commands and responses between two communication devices transmit small packets of data efficiently while eliminating the need to develop encode and decode functions in a memory limited embedded system. We deploy the full capabilities of the available UART technology with our design of

communication protocol between the smart alarm module and the Light Control Subsystem that maximizes data transfer efficiency and minimizes data transfer size.

A transmission cycle between the smart alarm module and the Light Control Subsystem for a change in light intensity begins with a five byte command. Figure 29 shows the command structure.

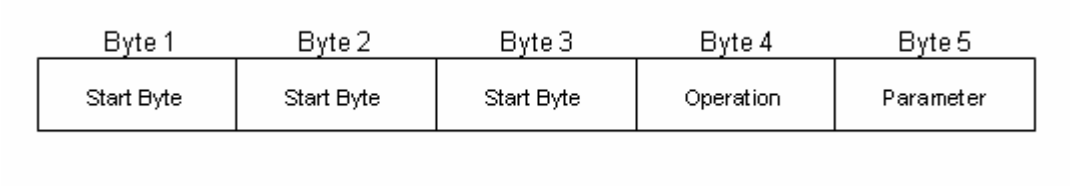


Figure 29: Command Structure

As a mean to detect transmission error, the first two start byte will always be the ASCII representation of the capital letter U, corresponding to 01010101 in binary. The alternation between 0 and 1 tests the data transfer accuracy of the Bluetooth link. If the Light Control Subsystem receives a data packet that does not start with the alternating sequence of 0 and 1, then a transfer failure message will be replied. As for the third byte, the upper four bits are 0101 with the same purpose as the first two bytes. The lower four bits is the binary representation of either 0001 if no parameter exists with the operation or 0010 if the parameter byte is meaningful. Likewise with the first two bytes, if either the upper four bits of the third byte fails to be 0101 or the lower four bits of the third byte represent neither 0001 nor 0010, then the Light Control Subsystem will reply with a transfer failure message.

Assuming the Light Control Subsystem receives the start byte successfully, the fourth byte becomes the operation byte specifying an instruction for which the light control unit is to accomplish. Table 17 tabulates possible valid values and the corresponding instructions.

Table 17: Operation Byte

Operation Byte	Instruction
00000000	Turn off light
00000001	Turn on light to full brightness
00000010	Turn on light gradually
00000011	Pause light at current brightness

The instruction turn off light requests the light control unit to turn off the light immediately. Two modes are available to turning the light on, either to turn on light to full brightness without delay and regardless of current brightness, or to turn on light

gradually at a rate provided in the parameter byte. The pause light at current brightness command will stop any turn on light gradually command in effect.

Serving to control the rate of which to increase light intensity, the parameter byte determines the amount of time required in minutes to adjust the brightness of the light from 0 to 100 percent intensity. The value of the parameter byte is an 8-bit unsigned value, ranging from 0 to 255 minutes.

If the five bytes command does not transmit in completion within a period of ten seconds, the Light Control Subsystem will assume command transfer failure due to timeout and response with a failure message.

As the final procedure towards the transmission cycle, the Light Control Subsystem replies to the command from the Alarm Clock Subsystem with a one byte message. Table 18 lists the possible reply message byte value and the meaning of these values.

Table 18: Reply Byte

Reply Byte	Reply Message
00000000	Transfer Successful
00000001	Transfer Failed due to Interference or Interrupted Bluetooth Linkage (bad start byte or command)
00000010	Transfer Failed due to Timeout
00000011	Transfer Failed due to Too Many or Too Few Bytes

With the acknowledgement from the light control unit to the alarm clock unit, the transmission cycle is complete.

6. Software Design

Once the hardware design and firmware support is in place, the high level software design can be approached. The software design section composes of three sections:

1. Modular description of abstracted hardware devices.
2. High level overview of the states and transitions that will occur.
3. Algorithms pertaining to the functionality of the subsystem, and the overall system functionality.

6.1. Modular Description of Hardware Components

This section describes the various hardware components that have been abstracted as software modules. Software module allows the system to initialize and call functions relating to these modules without worrying about the underlying firmware and hardware details.

6.1.1. LCD Module

This module abstracts both the numeric and alphanumeric LCD. Functions are available to update the contents on these LCDs.

6.1.2. Real Time Clock (RTC) Module

The RTC module is responsible for keeping accurate track of the current time. It abstracts away the watch crystal and provides functions to update the system time at fixed periods.

6.1.3. I^2C Module

This module abstracts away the details of the I^2C bus by providing functions for transmitting and receiving both data and commands along the bus.

6.1.4. SPI Module

This module abstracts away the details of the SPI bus by providing functions for transmitting and receiving both data and commands along the bus.

6.1.5. MMC Module

The MMC module allows the system to read from and write to the MMC in blocks of data. MP3 files can be read off the MMC via this module and passed on to the MP3 module.

6.1.6. MP3 Module

The MP3 module abstracts away the details of the VS1002d MP3 decoder. It allows the system to play MP3 songs by working in conjunction with the MMC module.

6.1.7. Input/Output (IO) Module

This module deals with the interface between the IO expander hardware. It will handle any user input that enters the system through the IO expander as well as provide output signals to configure other hardware devices.

6.1.8. Amplifier (AMP) Module

The AMP module abstracts the detail of the audio amplifier. It provides functions to increase the volume level in a linear fashion.

6.1.9. Bluetooth (BT) Module

This module deals with the wireless Bluetooth connection between the Alarm Clock Subsystem and the Light Control Subsystem. Details relating to the communication protocol are hidden from the system.

6.1.10. Pulse Monitor (PULSE) Module

The PULSE module abstracts away the details of the pulse monitor subsystem. It contains the necessary functions to take pulse measurement samples at regular intervals and process the raw data to translate it into a numerical value.

6.2. System State Design

The SAC system design can be best modeled as a finite state machine (FSM). This approach allows us to define certain states that the system will be in depending on past history. The state transition diagram is shown in Figure 30.

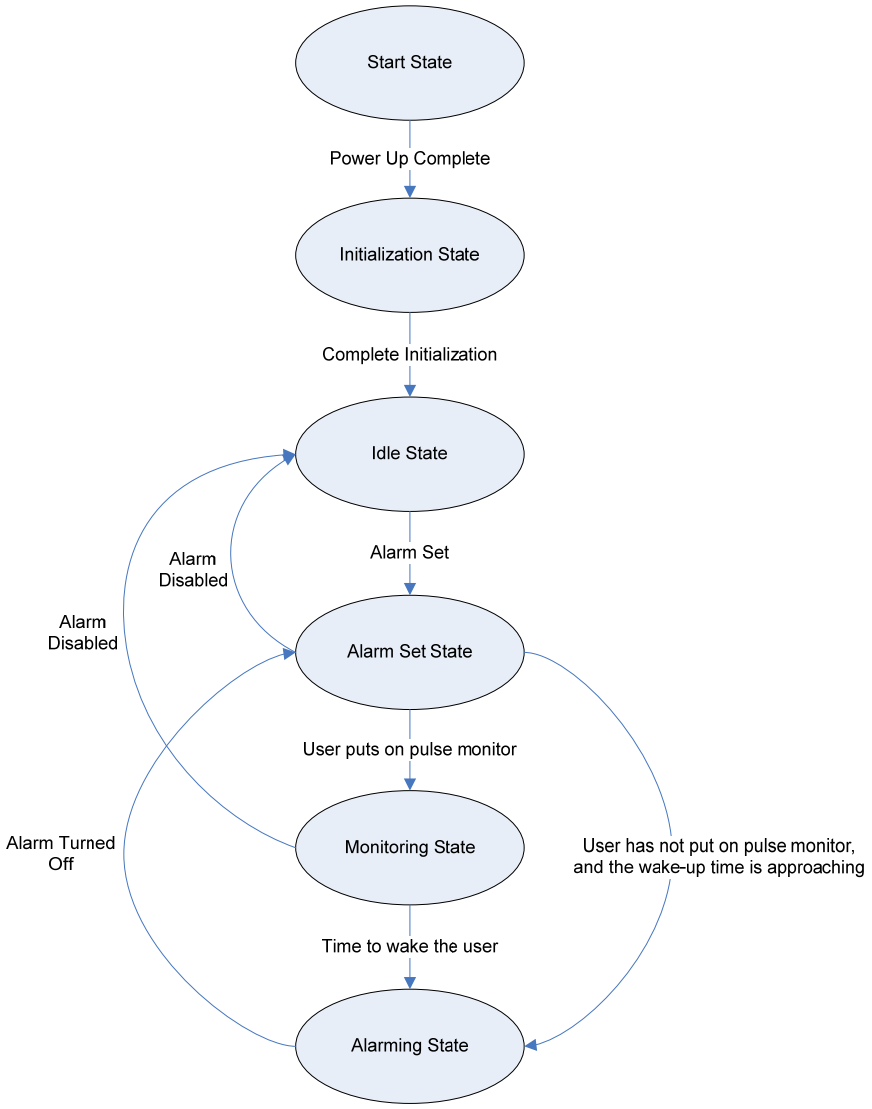


Figure 30: State Transition Diagram for the Smart Alarm Clock

The system is composed of 6 different states. Depending on which state the system is in, different functions need to simultaneously execute.

6.2.1. Start State

This is the state the system begins in shortly after power is applied. The system will remain in this state for as long as it takes to power up the rest of the associated modules. Certain hardware components require some time to pass once power is applied to the system. Timing related hardware such as crystal resonators require some time for the crystal to stabilize. Once power up is complete, the system moves to the Initialization State. Table 19 shows the state transition table for the Start State.

Table 19: Start State Transition

Next State	Condition
Initialization State	After power up is complete.

6.2.2. Initialization State

In this state, the system initializes the various hardware components. The microprocessor will configure various pins and the functionalities associated with those pins such as signal direction and output logic level. Any timers and counters that need to be initialized will also happen here. All other hardware modules shall be initialized and configured during this state and transition to Idle State occurs shortly after initialization completion. Table 20 shows the state transition table for the Initialization State.

Table 20: Initialization State Transition

Next State	Condition
Idle State	After initialization is complete.

6.2.3. Idle State

After the system has powered up and all modules have been initialized, the system enters the Idle State. The system also re-enters this state whenever the alarm has been disabled. This state, along with the Alarm Set State, will constitute the states where the system spends most of the time in. In the Idle State, the wake-up time has not been set. The functions that must be performed in this state include:

1. Updating the current time using the RTC module.
2. Updating the numeric LCD to reflect current time using the LCD module.
3. Handling input commands using the IO module.
4. Processing inputs that traverse the user menu and updating the alphanumeric LCD display using the LCD module.
5. Processing inputs that require turning on the external light via the Light Controller Subsystem using the BT module.

Once the user sets the desired wake-up time or enables the alarm the system will move to the Alarm Set State. Table 21 shows the state transition table for the Idle State.

Table 21: Idle State Transition

Next State	Condition
Alarm Set State	When the user enables the alarm, or sets a wake-up time which automatically enables the alarm.

6.2.4. Alarm Set State

The system enters the Alarm Set State whenever the alarm has been set. Like the Idle State, this is the other state that the system will spend most of the time in. Functionally, this state behaves in a similar fashion as the Idle State (1~5), with the exception of introducing a few other tasks:

6. Determining whether or not the user's finger is inserted in to the Pulse Monitor by using the PULSE module.
7. Determining if the wake-up time is approaching.

Once the user has put on the pulse monitor, the next state will transition to the Monitoring State. If, for any reason, the user fails to put on the pulse monitor, the system will transition to the Alarming State directly once the wake-up time is sufficiently close. If the user happens to disable the alarm during this process, the system moves to the Idle State. Table 22 shows the state transition table for the Alarm Set State.

Table 22: Alarm Set State Transition

Next State	Condition
Monitoring State	When the user puts on the pulse monitor.
Alarming State	When the wake-up time is within 15 minutes and the user has not put on the pulse monitor.
Idle State	When the user disables the alarm.

6.2.5. Monitoring State

In this state, the system actively monitors the user's pulse rate and performs algorithms to see when to wake the user up. The system is very computationally active in this state, more so than previous states. Thus, functions that execute during this state must be able to share the microprocessor's resources efficiently. This state maintains the same functionalities as the Idle State (1~5), with the addition of:

6. Calibrating the Pulse Monitor and determining the best setting for taking samples using the PULSE module.
7. Taking pulse samples at a fixed rate that exceeds twice the Nyquist sampling frequency, through the PULSE module.
8. Converting the raw sample data into numeric pulse value.
9. Performing read and write access to the MMC for storing the user's pulse information using the MMC module.
10. Executing the algorithm for determining the opportune time to wake the user by examining the samples taken.

The system will transition to the Alarming State on two conditions. The first condition occurs when the user enters Rapid Eye Movement (REM) sleep stage during the monitoring window. The second condition occurs when the user has not entered REM sleep stage, but the wake-up time is within 15 minutes. The system also transition back to Idle State when the alarm is disabled. Table 23 shows the state transition table for the Monitoring State.

Table 23: Monitoring State Transition

Next State	Condition
Alarming State	When the wake-up time is within 15 minutes, or when the system detects the user entering REM sleep during the monitoring window.
Idle State	When the user disables the alarm.

6.2.6. Alarming State

In the alarming state, the system attempts to wake the user by controlling the external light source as well as playing the desired music as per the functional requirement. The system may arrive to the Alarming State under one of two circumstances:

i. The system transitioned from the Monitoring State and the user has just entered REM sleep stage during the monitoring window.
ii. The system transitioned from the Alarm Set State, or from the Monitoring State, under the condition that the wake-up time is within 15 minutes.

Depending on which arrival circumstance occurred, the system will wake the user by using two different methods.

Circumstance i:

a. Instruct the Light Control Subsystem to turn on the external light source to full brightness at the wake-up time, which is at least 15 minutes away.
b. Instruct the Music Player Subsystem to start playing the selected MP3 song either immediate after the REM stage (through continual monitoring of the pulse rate) or after 90% of the average REM sleep cycle duration has passed based on previous samples, whichever comes first.
c. Increase the volume starting from the user defined volume level up to 60 dB as per the functional requirement, in 15 minutes.
d. Repeat part (c) until the user turns off the alarm.

e. If the wake-up time is exceeded, instruct the Light Control Subsystem to turn the external light source to maximum brightness and instruct the Music Player Subsystem to start playing the selected alarm sound at 80 dB loudness.
f. Wait until the user turns off the alarm.

Circumstance ii:

a. Instruct the Light Control Subsystem to turn on the external light source to full brightness in 15 minutes.
b. Instruct the Music Player Subsystem to start playing the selected MP3 song.
c. Increase the volume starting from the user defined volume level up to 60 dB as per the functional requirement, in 15 minutes.
d. Repeat part (c) until the user turns off the alarm.
e. If the wake-up time is exceeded, instruct the Light Control Subsystem to turn the external light source to maximum brightness and instruct the Music Player Subsystem to start playing the selected alarm sound at 80 dB loudness.
f. Wait until the user turns off the alarm.

The goal for both situations is to provide the user as much REM sleep as possible. The functions that will be performed during this stage are the same as the Monitoring State (1~10), with the addition of:

11. Instructing the Music Player Subsystem to start playing the selected MP3 song.
12. Increasing the volume level by using the AMP module.

After the user turns the alarm off or enough time has passed beyond the wake-up time, the system will transition to the Alarm Set State. Table 24 shows the state transition table for the Alarming State.

Table 24: Alarming State Transition

Next State	Condition
Alarm Set State	After user turns off the alarm or the alarm has been on for sufficiently long period after the wake-up time.

6.3. Algorithms

This section describes the various algorithms that are in use by the SAC.

6.3.1. Real Time Clock Generation

The main functionality of the alarm clock is to keep track of the current time as accurately as possible. To do this in an accurate manner, an external 32.768 kHz watch crystal is used. Timer2 on the microcontroller will increment its count by 1 very 128 cycles, and will overflow every 256 counts. This effectively results in the timer overflowing once every 1 second:

32.768 kHz/128 = 256 Hz / 256 counts/overflow = 1 overflow/second

An interrupt service routine will be generated every second to update the seconds, minutes and hours of the day. The flow chart of the real time clock algorithm is shown in Figure 31. When the Timer2 interrupt occurs on the timer overflow, the following algorithm is executed.

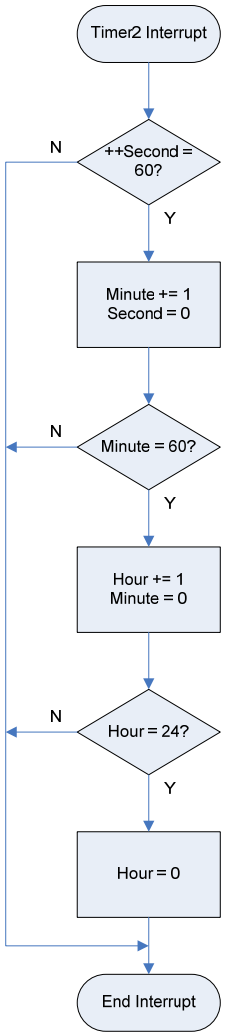


Figure 31: Real Time Clock Algorithm

6.3.2. Input Handling

The I/O expander allows the microprocessor to interface with numerous input and output signals at a great reduction of actual pins used on the microprocessor. Through the I^2C bus, the values on each port of the I/O expander can be read in a serial fashion. An interrupt signal is generated by the I/O expander whenever input signals change. The interrupt is used to instruct the microprocessor to read the input values.

Figure 32 shows the algorithm for handling input commands. The flow chart on the left hand side of the diagram deals with interrupt handling, while the flow chart on the right performs actual processing of the command. New commands are pushed on a command queue and during IO_Process() the command is popped off the queue. Commands are only executed when the input went from Off to On.

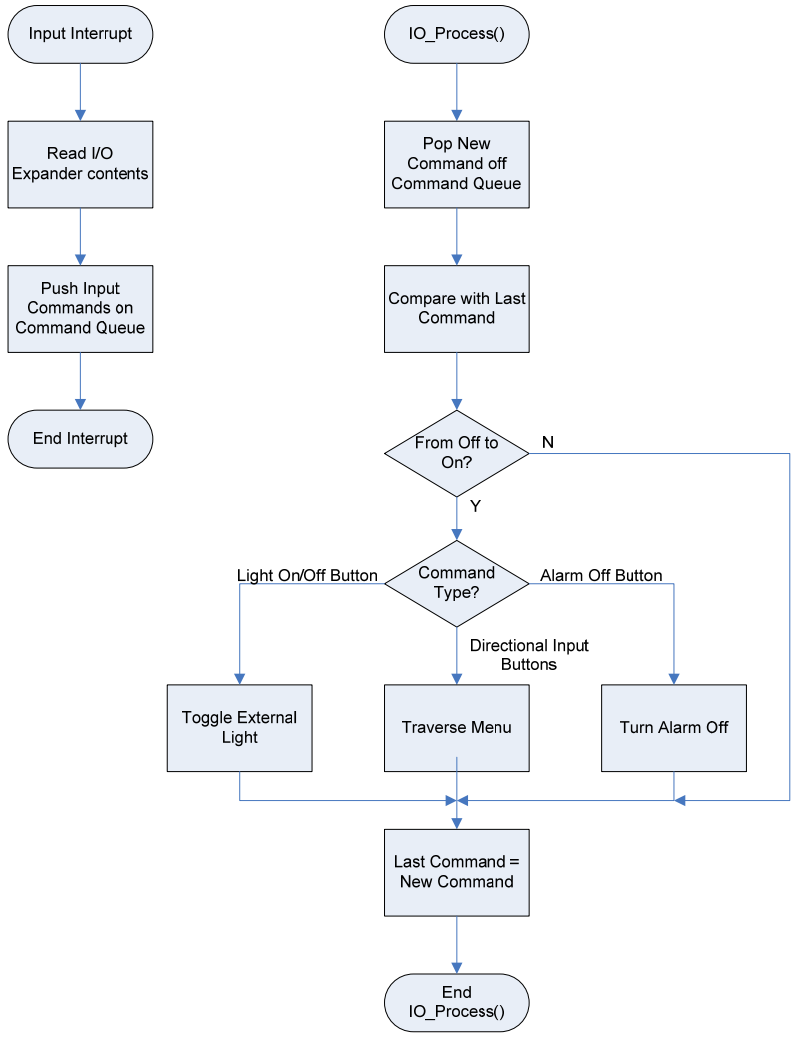


Figure 32: Input Handling Algorithm

6.3.3. Reading MP3 Files from MMC

MP3 files are read from the MMC in blocks of 512 bytes, which is the size of one File Allocation Table (FAT) sector. In order to read a file efficiently from the MMC, it is necessary to build a table that links together the memory addresses of these clusters beforehand so the file can be read continually.

Figure 33 shows the MMC Read File algorithm. Every time the function is called, 512 bytes of data is read back at a time and stored locally in Random Access Memory (RAM). A chain of cluster is built during the first file reading process. When all sectors of a cluster have been exhausted, the first sector in the next linked cluster is read. This function is meant to be called numerous times, where file reading is performed until exhausting all the clusters and sectors for that particular file.

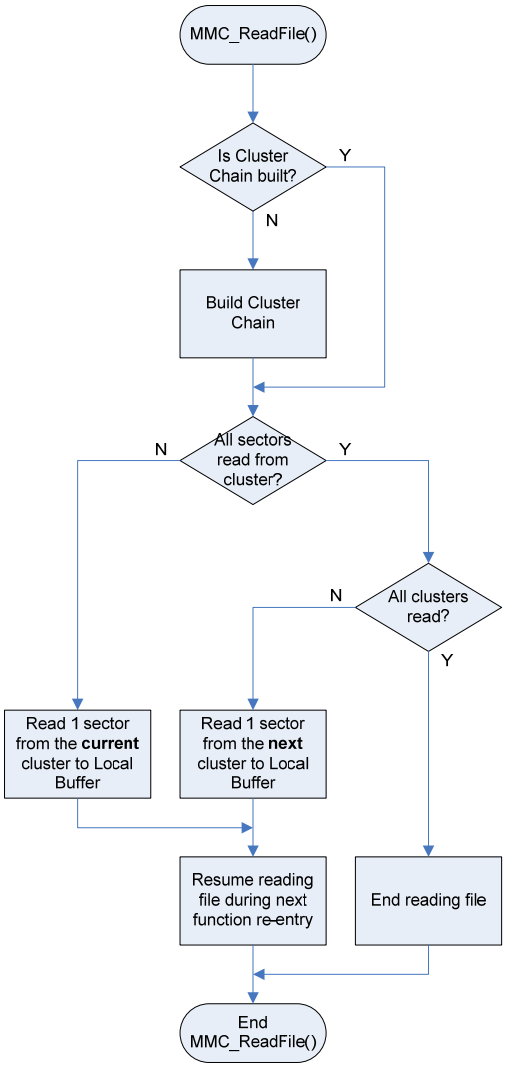


Figure 33: MMC Read File Algorithm

6.3.4. Playing MP3 Files

MP3 decoding takes place when valid audio data is written to the VS1002d chip through the SDI. The DREQ line remains high whenever the chip can accept more audio data. The easiest way to ensure that there is sufficient audio data in the VS1002d chip's buffer for decoding is to continuously poll the DREQ line. During the Alarming State, many functions will be executing, and must share the microprocessor's resources. Thus, the algorithm for playing MP3 files must be kept short along with the other functions that may be executing so no single function hogs all the microprocessor's resources.

Figure 34 shows the algorithm for playing an MP3 song. A Write_Count is initialized to keep track of how many times 32 bytes of data have been written to the VS1002d chip per function call. To prevent the function from monopolizing the processor, once a threshold limit has been exceeded this function will return and yield the processor for other functions. MP3 files are read from the MMC in blocks of data, as mentioned in a previous section and will not be elaborated in the flow chart.

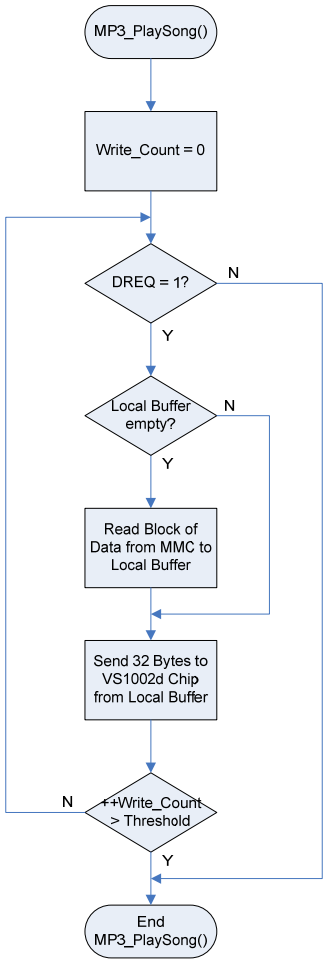


Figure 34: MP3 Playing Algorithm

6.3.5. Increasing Volume Level

The algorithm for increasing volume level is straightforward. The time required to increase from the user-defined minimum volume level to the level required to generate 60 dB is fixed at 15 minutes. Thus, depending on the user-defined minimum volume level, the rate of change will be different.

900 seconds / (Volume Level Required for 60 dB – User Defined Volume Level)
= Interval in seconds between each step increase

The function responsible for increasing the volume level will check against the current time (in seconds) to see when it is time to increase the volume level by 1 step.

6.3.6. Acquiring Pulse Samples

The light absorption by the user's body is continuously monitored by the Pulse Meter Subsystem (PMS), this information is then sampled by the Alarm Clock Subsystem (ACS) to determine a pulse rate. The technique to achieve this goal is by measuring the period of the pulse train output from the PMS at a fixed sampling rate using the 16-bit timer on the ACS's microcontroller.

Clocking the microcontroller at 8MHz and using the /8 timer prescaler, the pulse train period can be measured with a resolution of 1us. According to the functional specifications, the PMS must measure a maximum pulse rate of 200 beats per minute (BPM). This translates to maximum frequency of 3.33Hz, giving a minimum sampling rate of 6.67Hz according to Nyquist. For our design, a sampling rate of 32Hz is chosen such that the same 16-bit timer can be used for setting the sampling rate and measuring the pulse rate period. A constant is loaded into the timer and it is allowed to run freely, on each timer overflow the same constant is reloaded to set the sampling rate. On each timer overflow the external interrupt will be enabled. External interrupts occur on a low to high transition of the pulse train from the PMS. The timer count for the next two external interrupt pulses from the PMS will be captured, and then the external interrupts are disabled until the next timer overflow. By taking the difference between the counts of the first and second external interrupt, the period of the pulse train can be measured.

The accuracy of the pulse rate period is limited on the upper end by timer overflow and on the lower end by the finite cycles needed to jump and return from the Interrupt Service Routine (ISR). To obtain more accurate measurements, the ACS is given the ability to change the sensitivity and scale the frequency output of the PMS through the control lines. In general, the most accuracy is achieved by measuring the longest pulse rate period which doesn't cause the timer to overflow.

6.3.7. Determining Pulse Rate

In order to determine the pulse rate, the system must be able to detect the positive and the negative peaks in the data samples. Since the incident light on the PMS can vary base on the environment and the user's movements, changes in the DC component of the pulse rate period measurements occur throughout the night. Figure 35 shows an example of the sampled pulse rate periods, the data is plotted in Microsoft Excel.

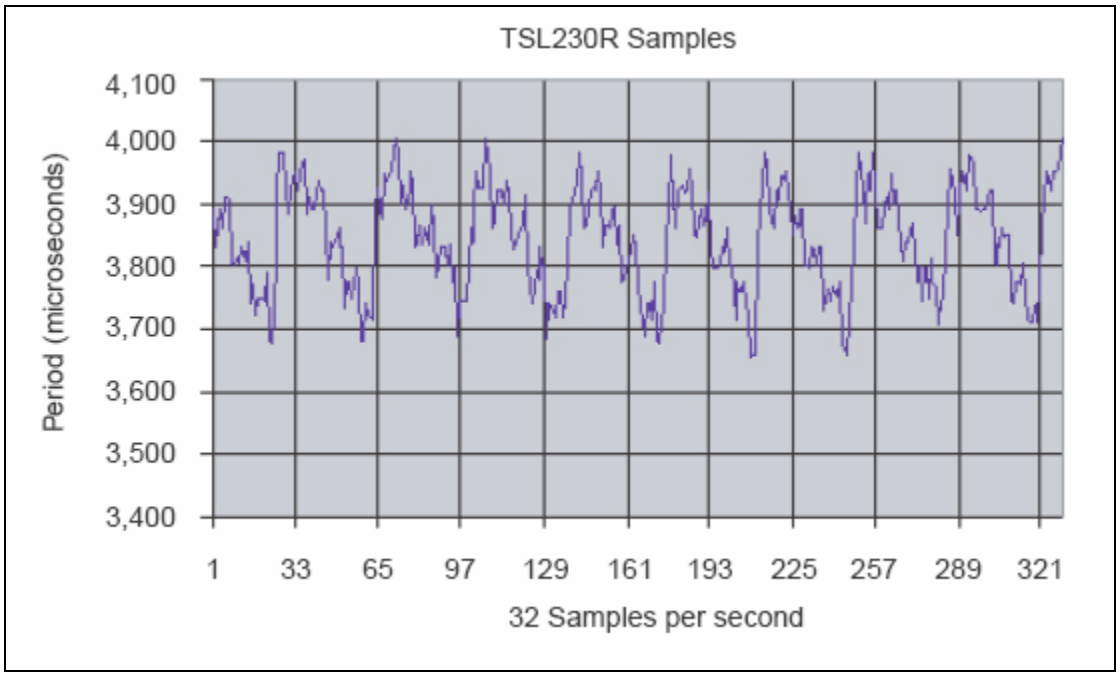


Figure 35: Excel chart of imported samples over time [2]

As can be seen, large amounts of jitter exists in the sampled data. To prevent cases in which a maximum is missed due to the downward shifting of the DC component, leaky peak detectors are implemented in software to detect the peaks. A similar method is used for the detecting the minimums as well.

The rate of leakages for the peak detectors are critical for our purposes; high leakage rate results in false recognition of peaks, low leakage rate leads to missed peaks. To implement the leaky peak detectors, rate of leakages are determined by the DC mean between the detected maximum and minimum. Under the correct conditions, he maximum is then reduced by a proportional factor of the DC mean. Likewise, the minimum is increased by a proportional factor of the DC mean under certain conditions.

Figure 36 shows the algorithm that aids the system in determining the pulse rate by determining the samples that represent the peaks, and finding the number of counts between peaks.

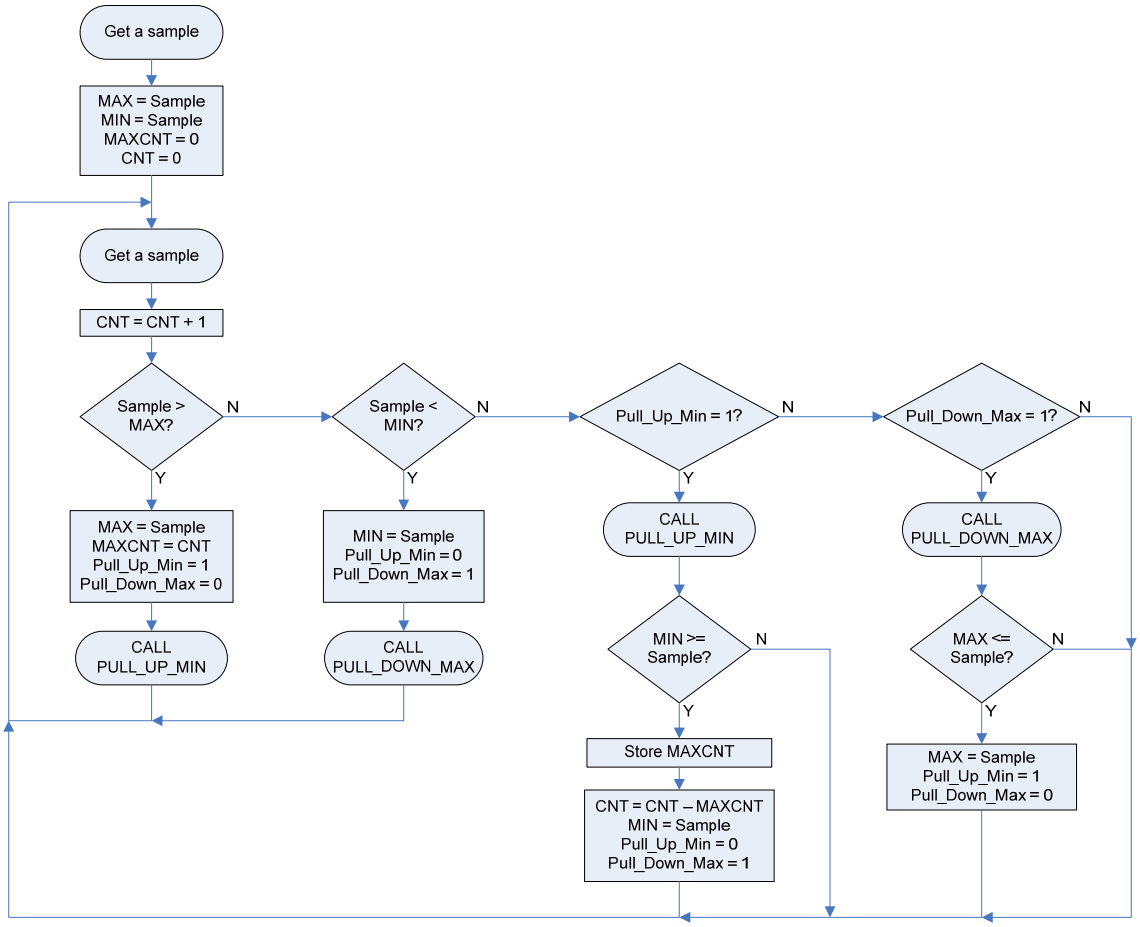


Figure 36: Pulse Measurement Algorithm

Each new pulse rate sample will be compared against the leaking maximum (MAX) and the leaking minimum (MIN) and the variables are updated. MAX and MIN carry information about the previous behaviour of the waveform and they are updated base on the current sample. When the Pull_Up_Min flag is set and a sample falls below MIN, we can be assured the last maximum is indeed the highest peak in the vicinity because the samples are heading towards a falling trend. By using this peak as the new reference point on the time axis, the pulse rate can be determined by finding the next peak in the waveform. A close analogy of our algorithm is a low pass filter; jitters are filtered and only the envelop of the waveform is kept.

6.3.8. Determining Appropriate Waking Stage

To determine the appropriate waking stage, enough pulse rate data must be acquired. This is usually done by monitoring the user's pulse rate throughout the night. The measured pulse rate can have statistical metrics calculated, such as mean, variance and standard deviation, within a sliding window. These statistical metrics will be an indicator of heart

rate variability and can be used to determine whether the user is in REM or non-REM sleep.

During NREM stage, both the mean pulse rate and variance are lower than during REM stage. By measuring the user's pulse rate from the time the user falls asleep to the time the user enters short wave sleep, an upper and lower boundary is established.

When the monitoring window as set by the user is approaching, the SAC begins analyzing the user's current pulse rate and compares it to the statistical values collected throughout the night. When the mean pulse rate and variance is sufficiently higher than the lower bound, the user will be consider as being in REM sleep and appropriate waking procedures will take place.

7. User Interface Design

The user interface with the SAC is menu driven, where the alphanumeric screen displays the menu functions that are available. The directional buttons allow the user to scroll through these selections until the desired one is highlighted. The centre push function on the directional button could then be used to select the highlighted function. Table 25 below shows the various menus shown on the alphanumeric screen.

Table 25: SAC Menus

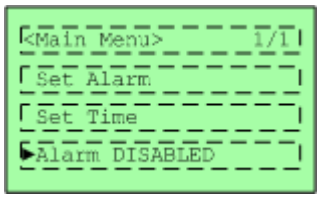
Menu	Description
Root Menu	The main menu of the SAC. Allows the user to enter other menus.
Alarm Menu	Contains alarm setup functions.
Clock Menu	Contains clock display functions.

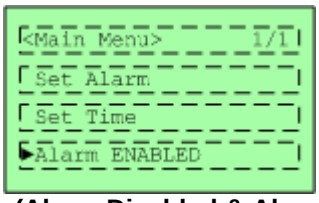
7.1. Initialization

When the SAC is initially powered, the application software will run an initialization routine, which will initialize both the numeric and alphanumeric LCD screens. The numeric screen will then show “12:00” and the alphanumeric LCD screen will show the root menu.

7.2. Root Menu

The Root Menu screen shows the main menu of the SAC as well as displaying whether the alarm is enabled. Two screenshots are shown below in Figures 37 & 38. An arrow points to the first function on the list, and can be moved up and down by the directional buttons. Table 26 also shows the functions available in this menu.





Figures 37 & 38: Root Menu Screenshots (Alarm Disabled & Alarm Enabled)

Table 26: Root Menu Functions

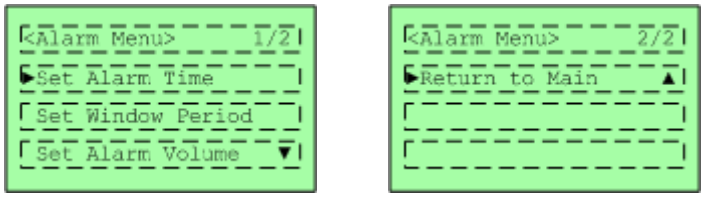
Function	Description
Set Alarm	Takes the user to the Alarm menu.
Set Time	Takes the user to the Clock menu.
Alarm Enabled/Disabled	Turns on or off the waking alarm process.

As stated, the Root Menu is shown upon power-up of the SAC. In addition, to meet one of the functional requirements, the Root Menu is displayed whenever the Alarm Clock subsystem receives no button input for two minutes.

An exception to displaying the Root Menu after two minutes without user input occurs while music is being played during the wake-up process, the alphanumeric screen will continue to show the name of the song being played until the alarm is turned off. Thus, no menu will be displayed.

7.3. Alarm Menu

The Alarm Menu allows the user to edit any functions relating to the alarm applications. Figures 39 & 40 show the menu screens and Table 27 describes the corresponding functions.



Figures 39 & 40: Alarm Menu Screenshots

Table 27: Alarm Menu Functions

Function	Description
Set Alarm Time	Adjusts the time that the alarm is triggered.
Set Window Period	Adjusts the duration of the monitoring window period.
Set Alarm Volume	Adjusts the initial volume that music plays when alarm is triggered.
Return to Main	Returns the user to the main menu.

When the "Set Alarm Time" function is selected, the alphanumeric screen will show the alarm time, as seen in Figure 41. An arrow initially points to the hour value. The user then uses the up and down directional buttons to select the desired value. Pressing the

right directional button moves the arrow to the minute value, and pressing it once more completes the task. The alphanumeric screen returns to the Alarm Menu.

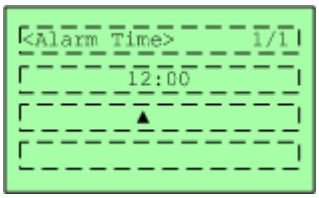
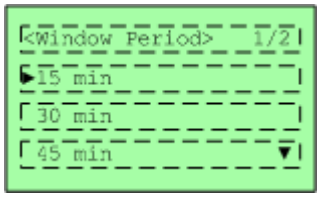
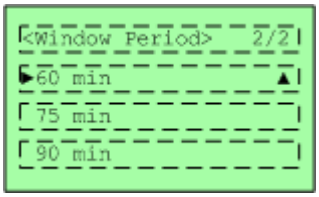


Figure 41: Set Alarm Time Screen

The "Set Window Period" function, seen in Figures 42 & 43, edits the maximum amount of time that the wake-up process can perform before the alarm deadline. This is when the music begins to play at its initial volume and the external light illuminates at its initial intensity.

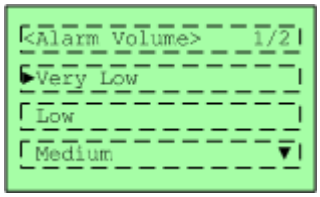


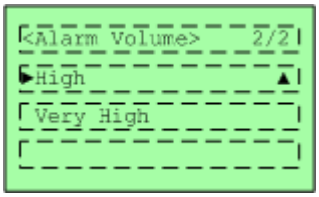


Figures 42 & 43: Set Window Period Screenshots

The maximum period allowed is set at 90 minutes, an approximation to the average length of a sleep cycle. By keeping the maximum period at 90 minutes, it prevents any possibility of the user waking up significantly earlier than the desired deadline. Waking up too early may urge the user to go back to sleep, ruining the general objective of the SAC.

The "Set Initial Volume" function is shown below in Figures 44 & 45. Users with sensitive hearing can set a music volume that starts off very quietly, while "deep sleepers" can set the initial volume at a higher value. Using the up and down directional buttons, the user can change the volume intensity.



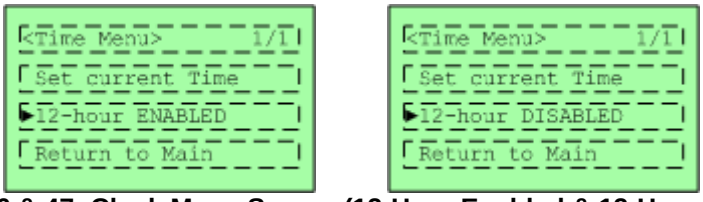


Figures 44 & 45: Set Alarm Volume Screenshots

As the volume is changed, a music or sound effect is played in the background to indicate how loud the actual value is. In addition, volume levels are described in simple English terms instead of volume decibel values for ease of understanding.

7.4. Clock Menu

The clock menu allows the user to change the current time and how it is displayed on the numeric screen. Figures 46 & 47 show two screenshots of this menu screen and Table 28 describes the available options.



Figures 46 & 47: Clock Menu Screen (12-Hour Enabled & 12-Hour Disabled)

Table 28: Clock Menu Functions

Function	Description
Set Current Time	Adjusts the current time.
12-hour Enabled/Disabled	Enables or disables 12-hour clock format.
Return to Main	Returns the user to the main menu.

The "Set Current Time" function takes the user to a screen similar to when he/she sets the alarm time shown in Figure 41. After the time is adjusted, this current time is updated on the numeric screen. The time can also be toggled to either 12-hour or 24-hour format as seen in the figure above.

8. Test Plan

This test plan will describe the guidelines towards testing all the subsystem designs independently, then together as a whole system.

8.1. Hardware Test Plan

8.1.1. General System Testing

Component Connection Verification

This test ensures that the components are properly placed and connected.

1. Verify that all chips and chip sockets are properly mounted on the circuit board.
2. Verify that component placements on board are the same as the schematics.
3. Verify that all the interconnection cables are connected properly.

To pass this test all these conditions must be satisfied.

Power Supply Verification

This test verifies that the power regulation circuit is properly constructed and provides the desired voltage level.

1. Connect the DC adapter to the adapter jack.
2. Verify that output voltages of 5V and 3.3V are measured on the two supply rails.
3. Verify the voltage supervisor resets the microcontroller when the supply rails fall below the monitored thresholds.

To pass this test, the proper supply voltage levels are met.

System Initialization Test

This test ensures all subsystems initialize to an expected state when the user connect all components to power sources.

1. Connect all subsystems to power sources
2. Verify that all subsystems power up in a proper manner and be ready for further operation.

To pass this test, the second condition above must be satisfied.

System Stress Test

This test verifies that all subsystems are able to handle repeated lost and regain of power.

1. Connect and disconnect all subsystems to power in two second intervals for five times.
2. Verify that all subsystems power up in a proper manner and be ready for further operation afterwards.

To pass this test, the second condition above must be satisfied.

Input Buttons

Recall that there are two types of input buttons are available on the SAC: a 4-direction tactile input button with center push and a tactile push button.

The basic functionality test for the input button includes testing for conductivity when the button is pressed, or in the case of the directional button, moved up, down, left or right. In addition to testing this buttons for basic functionality, these buttons shall also be able to sustain frequent usage. The test engineer shall perform the following:

1. Press the 4-direction tactile button up, down, left, and right 5 times for each direction.
2. Press the 4-direction tactile button inwards 5 times.
3. Press the tactile push button 10 times.

In order to pass this test, all buttons have to function properly without mechanical failure.

8.1.2. Alarm Clock Subsystem Verification

Seiko Tuning Fork Watch Crystal

This watch crystal is used as an external clock by the microcontroller. To test that this crystal is fully functional, the test engineer shall perform the following:

1. Connect the watch crystal to the microcontroller's TOSC1 and TOSC2 pins where internal amplification is done.
    a. Probe these two pins using an oscilloscope to check that the watch crystal is operating at a frequency of 32.768 kHz.

For this test to pass, the watch crystal's frequency needs to be exactly 32.768 kHz, taken into account the error uncertainty given by the oscilloscope.

Alphanumeric LCD

The LCD is tested to see if all the characters can be displayed properly with all the pixels functioning. The test engineer shall perform the following:

1. Initialise the alphanumeric LCD to 4-bit operating mode as outlined on the device datasheet.
2. Completely fill the LCD screen (20 characters by 4 lines) with characters 1-9, a-z, and A-Z.
3. Check for significant dead pixels.
4. Execute clear screen instruction to clear the screen contents.
5. Repeat steps 2 and 3.

The LCD should be initialised properly and show the correct alphabetical and numeric characters. The LCD screen should properly clear and be written to again.

Four Digit Numeric LCD

The LCD is tested to see if all the numbers can be displayed properly with all the segments functioning. The test engineer shall perform the following:

1. When the power is connected, ensure that a colon is displayed at all times.
2. Display each number (0-9) on each 7-segment LCD display to ensure that all digits are displayed correctly, and check for any dead segments.

In order to pass this test, the proper number should be written to each digit.

LCD Bus

This test checks whether or not the alphanumeric and numeric LCD can function co-operatively. The test engineer shall perform the following:

1. Initialise the numeric LCD to “12:00”.
2. Initialise the alphanumeric LCD.
3. Write a few lines of characters to the alphanumeric LCD.
4. Verify that the numeric LCD contents remain the same.
5. Update the numeric LCD contents to “12:01”.
6. Verify that the alphanumeric LCD contents remain the same.

To pass this test, neither LCD contents should change when they are not supposed to.

I/O Expander

This test will check if the correct input port values can be read by the microcontroller. The test engineer shall perform the following:

1. Configure all ports on the I/O expander as input by writing 0xFFFF to the chip registers.
2. Short the first input pin to ground, and verify that the port values are correct (0xFFFE).
3. Repeat step 2 by subsequently shorting the next input pin until all pins are shorted to ground, and verifying the port values during each iteration.

To pass this test, the proper input values shall be read.

Bluetooth Interconnection

This interconnection enables the Alarm clock subsystem to wirelessly communicate to the light control subsystem.

The test engineer shall perform the following:

1. Ensure that the Bluetooth module powers up correctly.
2. Using another Bluetooth enabled device ensure that the Bluetooth connection status is active.
3. Toggle between Command and Data modes to ensure both modes are accessible and functional.
4. Transmit and Receive data.

When transmitting and receiving data, the test engineer needs to ensure that no data is lost. No data loss can occur for this test to pass.

8.1.3. Music Player Subsystem Verification

The MPS is able to play audio files of MP3 format. MPS is responsible for MP3 processing as well as interfacing with non-volatile storage. MPS must also control the volume level.

VS1002d MP3 Audio Codec

After connecting all DVDD pins, xRESET, all AVDD pins and TEST0 to 3 volts, connect DGND and all AGND pins to ground.

The test engineer will perform the following to test that the hardware initializes correctly:

1. Measure voltage of RCAP, make sure it's approximately 1.3V.
2. Hardware reset the board (xRESET is turned to 1), and see that DREQ should turn to 0 after approximately 4096 cycles.
3. After another 6000 clock cycles, DREQ should turn to 1.

To pass this test, no deviance from the expected result should occur.

MMC

MMC has two modes of operation: MMC mode and SPI mode. The microprocessor will use the SPI mode.

The test engineer shall perform the following:

1. Verify that the MMC can enter SPI mode.
2. Write a block of sample data to the MMC.
3. Read the block of sample data written to the MMC.

Verify that the data read is identical to the data written.

MP3 Playback

In order to test that MP3 decoding is functioning correctly, the test engineer shall perform the following:

1. Upload one MP3 file to the MMC.
2. Play the MP3 file to ensure that decoding functions correctly.
3. Stop the file playing, and start playing it again.
4. Play the song until completion, and make sure the song can be repeatedly played.
5. Connect the decoder module to speakers, and ensure that the sound is outputted clearly.

8.1.4. Light Control Subsystem Verification

The LCS is a physically standalone device that is responsible for controlling the brightness of an external lamp. The LCS communicates wirelessly with the ACS via Bluetooth technology, the same device used in ACS.

To test this module, the test engineer shall perform the following:

1. Instruct the LCS using the specified Bluetooth
2. Turn off the light completely.
3. Gradually increase the light to full intensity.

8.1.5. Pulse Meter Subsystem Verification

The PMS is responsible for measuring the user's pulse and sending this information back to the ACS for determining the best sleep stage to wake the user.

To test this subsystem, the test engineer shall perform the following:

1. Wear this pulse measurement device and record up to 3 minutes of pulse data.
2. Simultaneously to step 1, wear another pulse measurement device that is retail (end-product) and record up to 3 minutes of pulse data.

For this test to pass, ensure that both devices output approximately the same pulse rate (maximum 2% margin of error).

8.2. Software Test Plan

The functionality of the user interface will be tested to verify that all menus act as per the functional specification. After loading the application onto SAC's prototype board, a set of test cases will ensure that all requirements written in the functional specifications are satisfied.

When testing the software aspect of this system, the test engineer will perform the following high level tasks:

1. Confirm the time is accurately displayed.
2. Explore the menus to ensure that:
    a. All menu options meet the requirements according to the functional specifications.
    b. Navigation is user-friendly and functional.
3. Set the alarm time, and verify that the alarm will go off at the correct time. The alarm will also play songs continuously until the alarm is turned off.

4. Set start level and time for the gradually increasing light intensity and volume.
    a. Ensure that the light intensity gradually starts increasing at the correct level and time.
    b. Ensure that the alarm volume gradually starts increasing at the correct level and time.
5. Compare the pulse measurement device to another type of pulse measurement and REM detection system, ensuring that the SAC is determining the transition into, or out of REM sleep stage correctly.
6. Confirm the SAC system wakes the user during, or shortly after, the REM sleep stage on a consistent basis.

8.3. Proof of Concept Experiments

The proof of concept experiment serves to verify in a quantitative manner the degree to which the Smart Alarm Clock improves performance and efficiency with regards to daily tasks. The experiment will also examine the degree to which the idea of waking people up using gradually increasing sound and light is practical.

When conducting this experiment, the test engineer shall perform the following:

1. Make up a short exam with questions requiring the individuals to be mentally alert to answer correctly.
2. Acquire up to 20 random individuals for testing.
3. While all these individuals are sleeping, set the test up such that half the individuals under the experiment wake up to a loud and obnoxious sound during their non-REM stage of the sleep cycle.
4. The other half the individuals use our device and wake up during their REM stage in the sleep cycle.
5. Request all individuals to write the exam that was prepared in step 1.
6. Record all exam scores, and ensure that the individuals who were awaken during the REM stage score significantly higher than the individuals who were awaken during the non-REM stage. Repeat this test as necessary, while modifying the exam each iteration.

9. Conclusion

The *Design Specifications for a Smart Alarm Clock* documentation provides a preliminary design standard towards the implementation of the Smart Alarm Clock proof of concept model. In accordance to the *Functional Specifications for a Smart Alarm Clock*, the design approach for the development of the Smart Alarm Clock fulfills all functional requirements described in the functional specification proposal. Our discussion of the product design is first divided into hardware, software and firmware design, and subdivided into the subsystems that contribute to the Smart Alarm Clock.

In the process of developing the proof of concept model for the Smart Alarm Clock, we execute the hardware, firmware, interface, software and system tests as described in this document. Further, a set of proof of concept experiments verifies our product concepts in a scientific and quantitative manner. At the end of the proof of concept development phase currently scheduled for April 2006, we hope that the Smart Alarm Clock proof of concept model will serve as an active tool to reduce the psychological and physical impacts that the sociological problem of reduced hours of sleep have on the mass population.

10. References

[1] A7 Engineering, Inc. 2005. *EmbeddedBlue 505 User Manual.* Poway, California.

[2] Bachiochi, J. 2004. Light-to-Frequency Conversion. *Circuit Cellar*. 173 & 174.

[3] Bray J. and C.F. Sturman. 2001. *Bluetooth: Connect without Cables*. Upper Saddle River, New Jersey: Prentice Hall PTR.

[4] Cavallero, C., Versache F. 2003. *Stage at Awakening, Sleep Inertia, and Performance.* Italy: University of Trieste.

[5] Clifford, G. D., and Tarassenko, L. 2004. *Segmenting cardiac-related data using sleep stages increases separation between normal subjects and apnoeic patients.* United Kingdom: IOP Publishing Ltd.

[6] Eulenberg, A. 2006. Eye Charts. 1 Mar. 2006 <http://www.i-see.org/eyecharts.html>.

[7] Inglewood Jack Technologies. 2006. *Functional Specification for a Smart Alarm Clock*. Burnaby, British Columbia.

[8] Kobitone Audio Company. 2005. *KT-400175 Data Sheet*.

[9] Lippman, A. 2003. *The Roles of NREM and REM Sleep On Memory Consolidation.* 9 Mar. 2006 <http://serendip.brynmawr.edu/bb/neuro/neuro03/web2/alippman.html>.

[10] Lumex, Inc. 2002. *LCD General Information*. Palatine, Illinois.

[11] Lumex, Inc. 2002. *LCM-S02004DSR Uncontrolled Document*. Palatine, Illinois.

[12] Masao Yaso, Atsuo Nuruki, Sei-ichi Tsujimura, and Kazutomo Yunokuchi. *Detection of REM sleep by heart rate*. Japan: Kagoshima University.

[13] Maxim Integrated Products. 1993. *Four Digit Display Decoder/Drivers*. Sunnyvale, California.

[14] National Sleep Foundation. *Sleep Drive and Your Internal Body Clock.* 9 Mar. 2006 <http://www.sleepfoundation.org/hottopics/index.php?secid=18&id=267>.

[15] Rosenzweig, M.R, Breedlove, S. and Leiman, A.L. *Biological Psychology*. 3rd ed. Massachusetts: Sunderland.

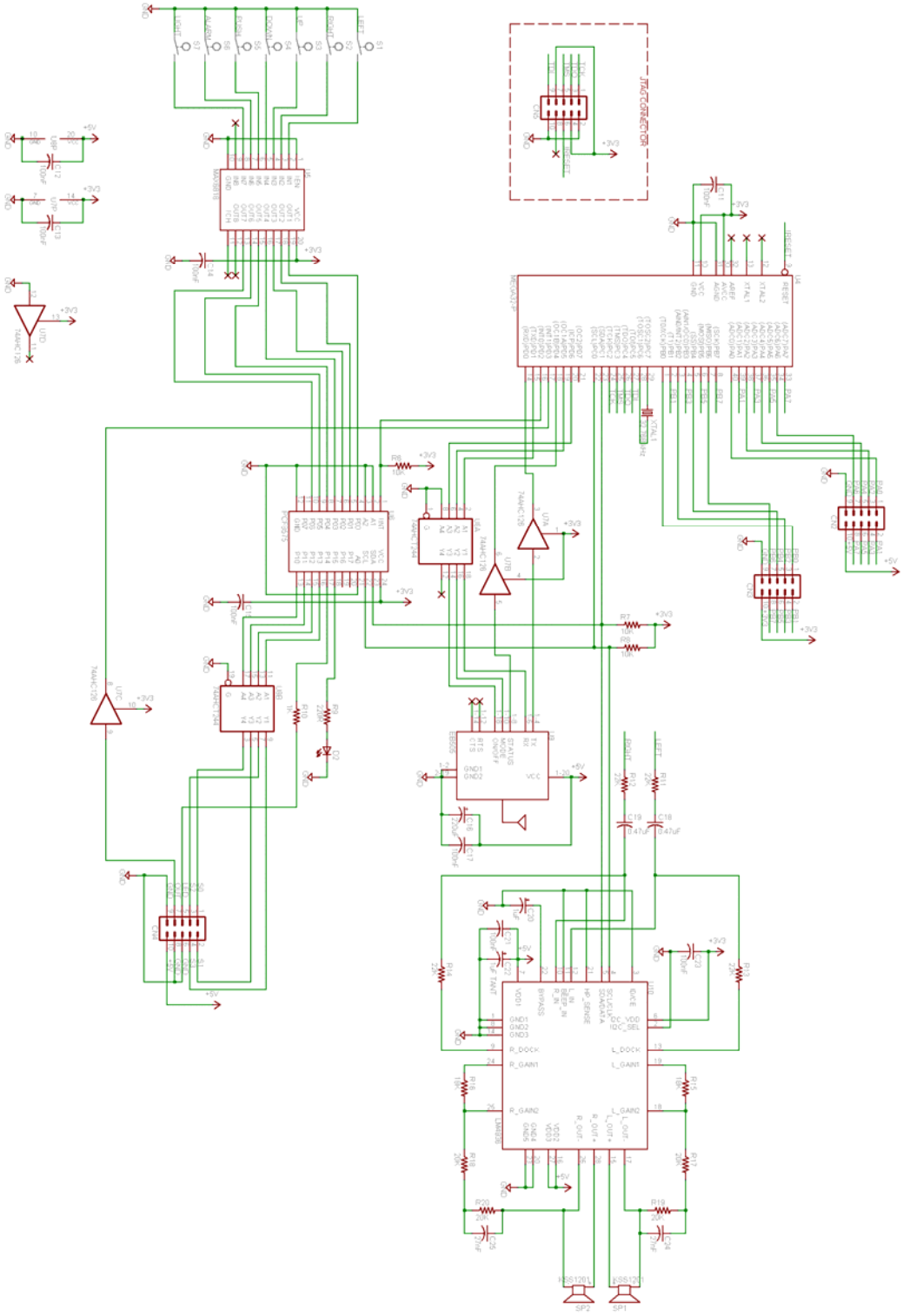
[16] Samsung Electronics Co., Ltd. 2000. Noise Guards. 1 Mar. 2006 <http://www.samsung.com/Products/HardDiskDrive/whitepapers/WhitePaper_02.htm>.

[17] VLSI Solution Oy. 2005. *VS1002d MP3 Audio Code*, Ver. 1.0. Tampere, Finland.

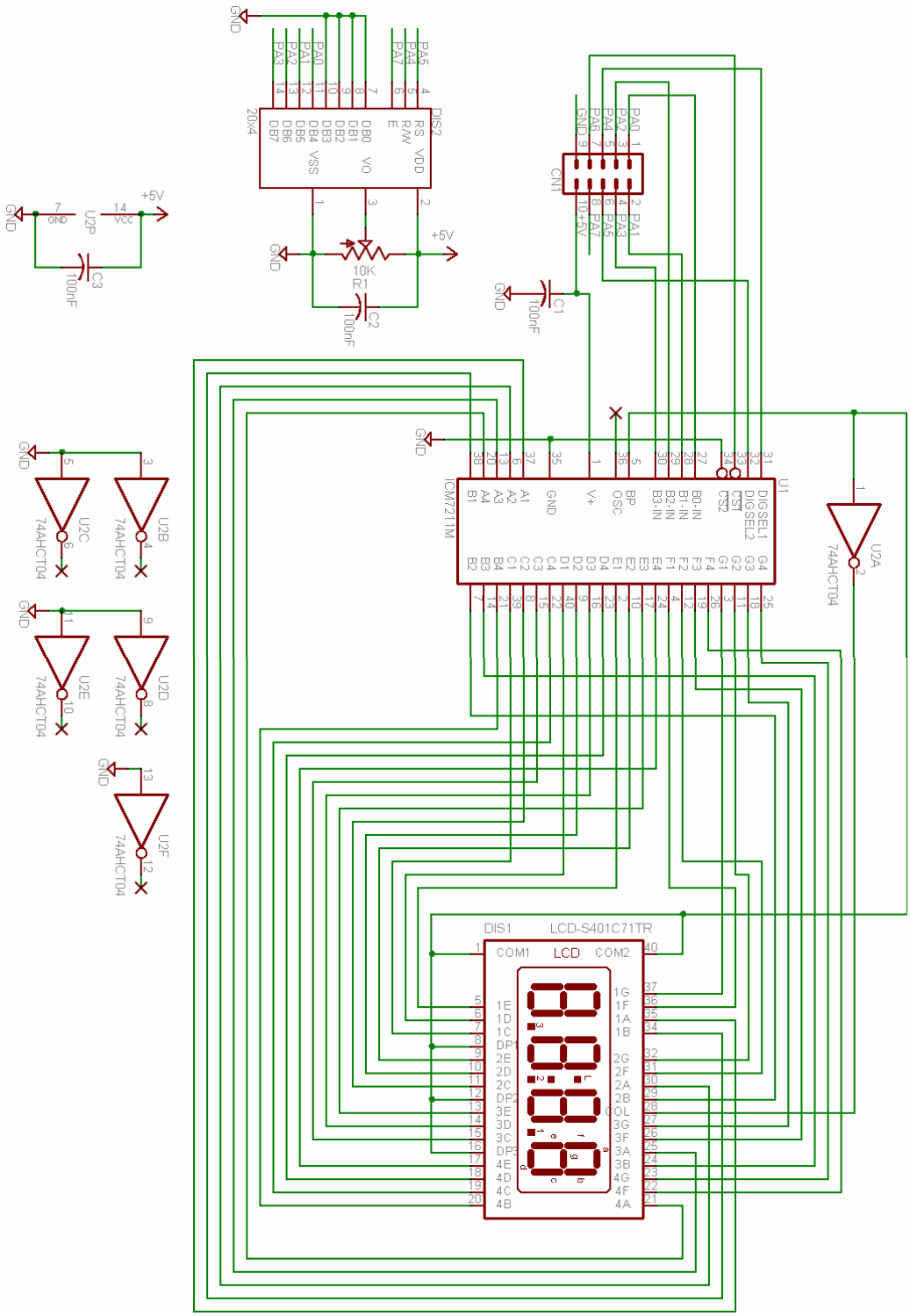
[18] VLSI Solution Oy. 2005. *VS10xx Application Notes*, Ver. 0.73. Tampere, Finland.

[19] VLSI Solution Oy. 2005. *VS10xx Standalone Player*, Rev. 1.01. Tampere, Finland.

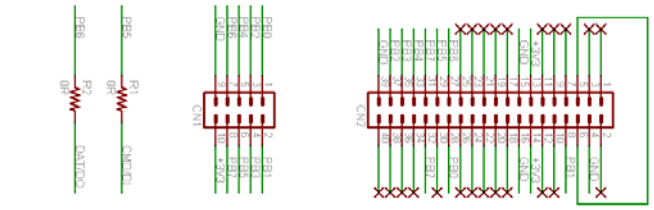
Appendix A – Alarm Clock Module Schematic

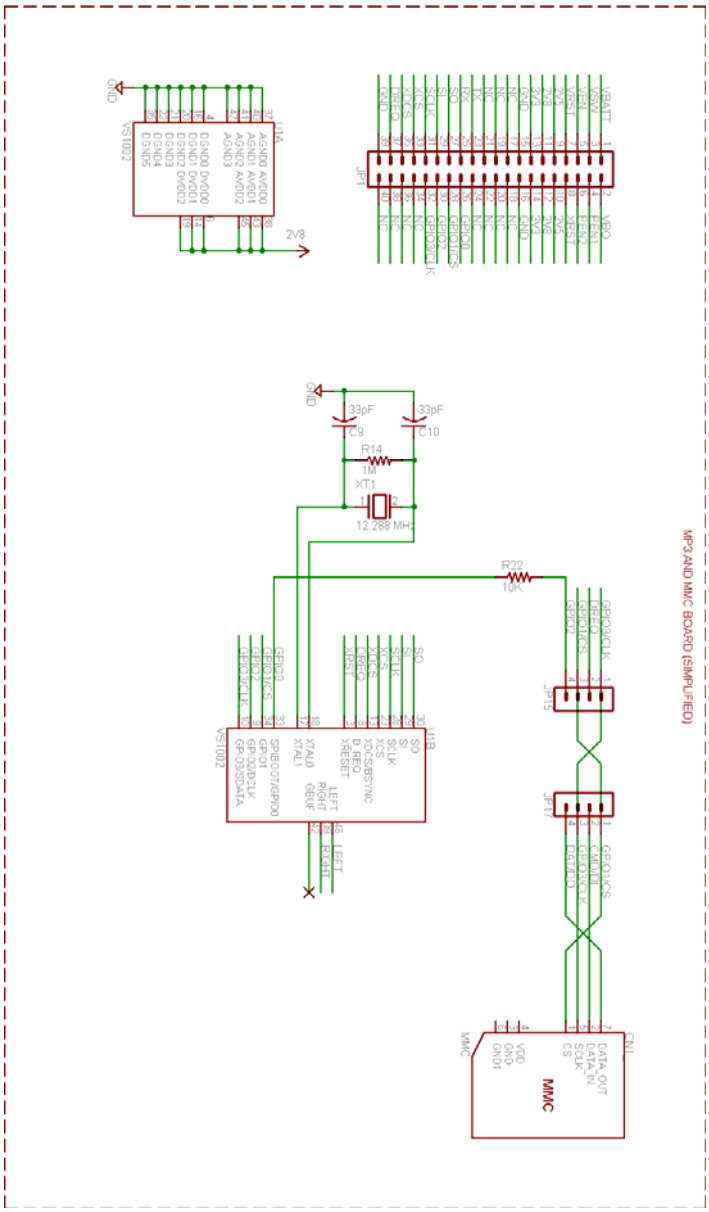


Appendix B – LCD Schematic

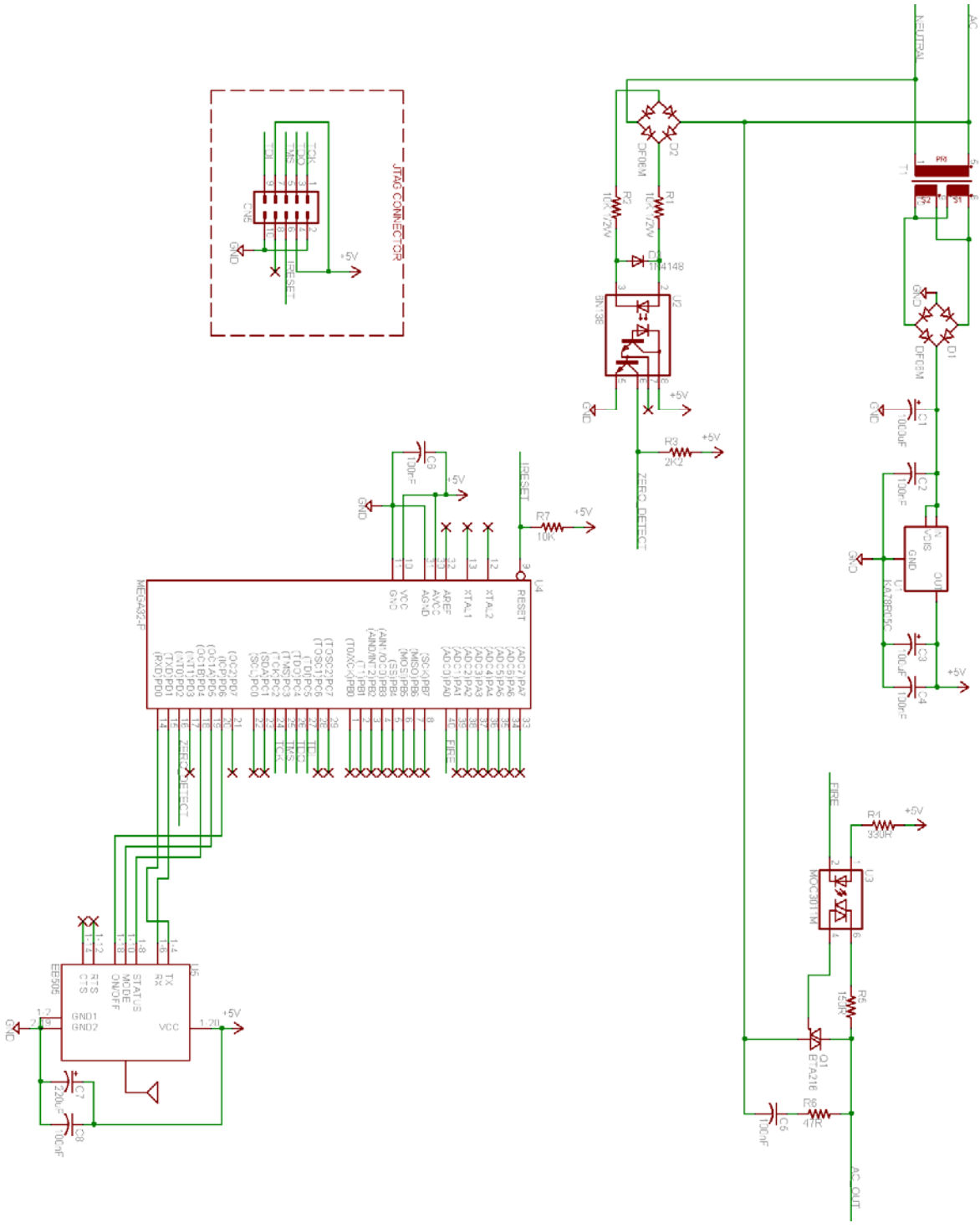


Appendix C – MP3 Module Schematic

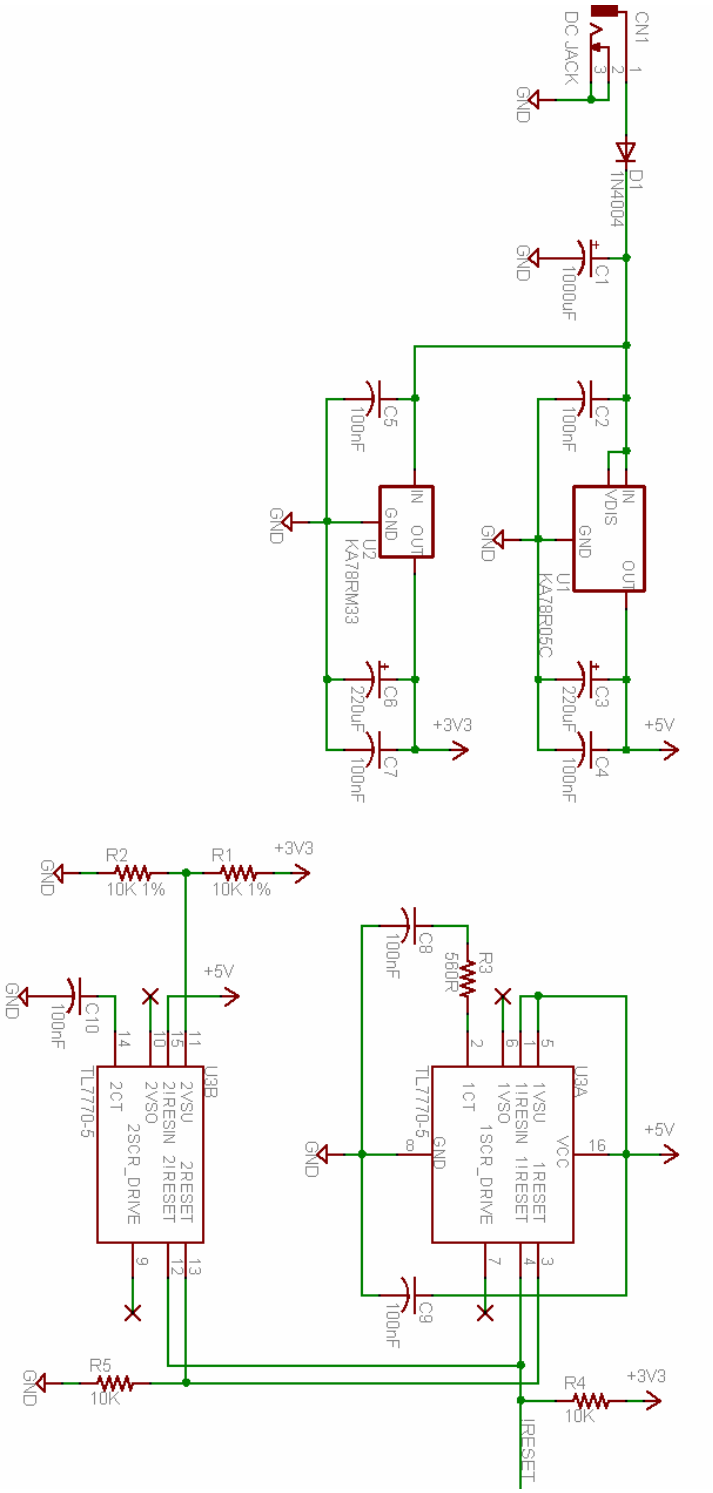




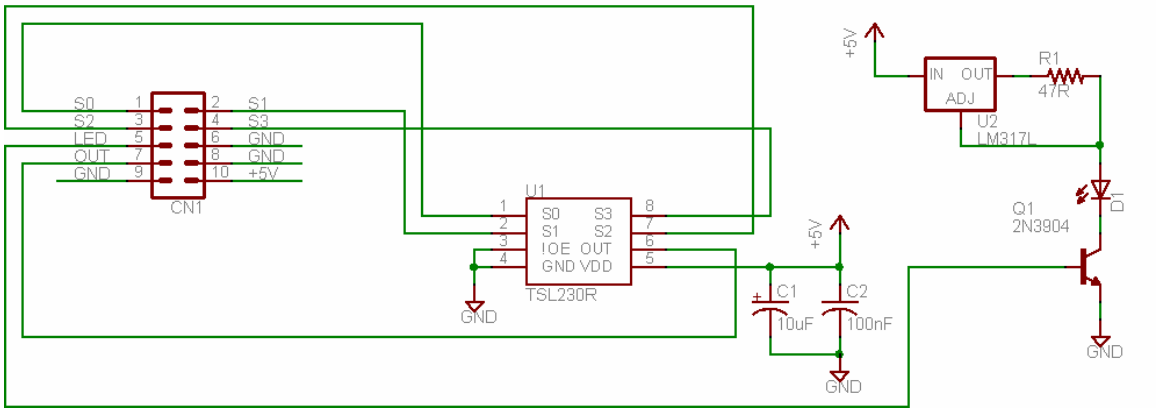
Appendix D – Light Dimmer Module Schematic



Appendix E – Power Regulation Schematic



Appendix F – Pulse Meter Module Schematic



Appendix G – Pulse Meter Conduit Design