Kwame Nkrumah University of Science and Technology

College of Engineering

Faculty of Electrical and Computer Engineering

Department of Computer Engineering

Auditorium and Theatre seat Occupancy Monitoring System.

(ATOMS)

By:

Adjei, Akwasi Kingsley

Addo, Ernest Ofosu

This project is submitted to the Computer Engineering Department as partial fulfillment of the requirements for the award of Bachelor of Science degree in Computer Engineering.

May, 2016.

Supervisor: Dipl. -Ing. Benjamin Nii Kommey.

DECLARATION

We, the undersigned, hereby declare that this project report entitled "**Auditorium and Theatre seat Occupancy Monitoring System**" is the result of our own research under the supervision of Dipl. -Ing. Benjamin Kommey, except as cited in the references. The project report has not been accepted for any degree and is not concurrently submitted in candidature for any other degree.

Name: Adjei, Akwasi Kingsley

Index Number: 7767712

Signature: ……………………….

Date: …..……………….………..…

Name: Addo, Ernest Ofosu

Index Number: 7767512

Signature: ……………………….

Date: …..……………….………..…

I hereby declare that I have read through this project report entitled "**Auditorium and Theatre seat Occupancy Monitoring System**" and found that it complies with the partial fulfillment for the award of Bachelor of Science Degree in Computer Engineering.

Supervisor's Name: Dipl. -Ing. Benjamin Kommey

Signature: ……………………………

Date: …………………………………

ACKNOWLEDGEMENT

To our parents and the unsung hero of Berlin.

Your toil did pay off.

ABSTRACT

ATOMS is a dynamic occupancy detector and display system for monitoring a plurality of chairs in seating areas such as theaters and auditoriums. Earlier devices have been made in attempts to efficiently and accurately determine the occupancy of seats. However these devices are based on technologies and theories that are either obsolete, cumbersome or not easily applicable in today's technologically speeding world. Previous systems have relatively older methods like wiring a whole auditorium with electrical cables for power and control of subsystem devices. This culminated in power wastage owing the voltage drop along the lengthy wiring. The lumpy wiring also made troubleshooting of faults in these systems very problematic. Other flaws in older devices which ATOMS addresses are the use of power wasteful redundant seat indicators and the non-portability of these system *et cetera*.

ATOMS poses as a very formidable front in remedying the afore mentioned issues thus realizing the efficiency, accuracy and reduced costs of production and operation that is so vehemently sought for by managements of such enterprises.

ATOMS has two main subsystems; a seat unit and a monitoring base station. The two subsystems although electrically not connected, are able to communicate over Wi-Fi (IEEE 802.11, 2.4GHz, ISM band) to discharge control preferences. The seat unit is composed of a capacitive sensor (chosen and configured specifically to sense a human occupant and not any other object else) and a wireless transceiver (Wi-Fi enabled micro controller unit in this case) affixed to the seat. The capacitive sensor which essentially is a thin circular aluminum foil is connected electrically to the transceiver. This subsystem determines and communicates information concerning the occupancy status of individual seats to the base station.

The second subsystem, which is basically a computing device with Wi-Fi communication capability is sited preferably close to the theatre so it is in the range of the wireless LAN. It processes the received information and displays which seats are occupied and which are vacant. The processing and display of information concerning the vacancy status of seats is done by a desktop application resident at the base station. This application is able to decode the encoded vacancy information and represent it in a graphical form with a layout analogous to the seat layout of the auditorium showing the presence of an occupant or the lack thereof.

Other devices like the gateway device and the repeater may be needed depending on the implementation of the network over which the seat unit and the base station communicate and the size of the auditorium of theatre respectively.

When a patron sits on a seat, the sensor picks up the stimulus from the environment. The sensed stimulus is transferred to the wireless transceiver as an analog signal. Here this continuously changing physical quantity is converted to a digital signal using the microcontroller's analog-to-digital converter. The generated signal is then processed, encoded into a TCP packet and then transmitted over a local area network to the base station. The resident ATOMS software application receives the encoded packet, decodes it and then

changes the status of the chair from '*available'* to '*occupied*'. The base station then updates the occupancy status of the chair on its display and any other connected display devices for observation by the base station administrator and all patrons. The display is also capable of numerically showing the seat availability. Additionally several controls which are effected on seat units can be undertaken from the base station.

Abbreviations and Acronyms

ADC	Analog to Digital Converter
AHB	Advanced High speed Bus
AP	Access Point
APB	Advanced Peripheral Bus
ARM	Advanced RISC Machine
ATOMS	Auditorium and Theatre seat Occupancy Monitoring System
BSSID	Basic Service Set Identifier
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
CSS	Cascading Style Sheets
DTIM	Delivery Traffic Indication Map
GPIO	General Purpose Input/Output
GR_LED	Green LED
HTML	Hyper Text Markup Language
ID	Identifier
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
LAN	Local Area Network
LED	Light Emitting Diode
LPDS	Low Power Deep Sleep
MCU	Microcontroller Unit
O_LED	Orange LED
PWM	Pulse width Modulation
QFN	Quad Flatpack No-Leads
R_LED	Red LED
RAM	Random Access Memory
RX	Reception
SSID	Service Set Identifier
TCP	Transmission Control Protocol
TI	Texas Instruments
TX	Transmission
UART	Universal Asynchronous Receiver/Transmitter
UDP	User Datagram Protocol
USB	Universal Serial Bus
UWB	Ultra wideband
Wi-Fi	Wireless Fidelity
WiMax	Worldwide Interoperability for Microwave Access
WLAN	Wireless LAN
WPA	Wi-Fi Protected Access
XTAL	Crystal clock

List of Figures

Contents

Intentionally Left Blank

1. INTRODUCTION

1.1 Background of Study and Problem Definition

The project associates generally to a system for visually providing information for patron use regarding the occupancy status of the seats and information regarding the seating arrangement of an auditorium, theater or any of the other halls like cinemas.

Patrons of theaters have frequently been unable to find seating for viewing a show easily. Many enterprises permit patrons to enter the seating area after the performance has started. In halls with larger number of seats, the above problem is further worsened as newly entering patrons have to keep standing for longer periods in order to scan the whole auditorium and discern and find a suitable vacant seat. Since this event requires an indispensable amount of time, embarrassment and infuriation on the part of the newly come and already seated clients respectively sets in. These often culminate into squabbles which are often uncalled for.

Furthermore, customers who patronize shows may be in a quest to find a fitting set of chairs with a preferred location to accommodate a family or group of friends compounds the problems as, such chair selections takes time and can only occur with the patron in the hall. After locating a seat, the patron, whiles standing in aisles, goes through the strenuous ritual of interrogating customers in adjacent seats whether the seat is actually empty or has been temporarily vacated. This calls for many conversations which are undesirably distracting and can be very aggravating to seated clients who are attempting to listen to or view the show.

Many hall establishments engage the service of ushers to resolve the above listed problems. These hired or volunteering personnel are tasked to personally seat newly entering patrons at available chairs and direct patrons to restrooms, refreshment stands, etc. However, employment of such additional personnel adds to the cost of operation of such establishments. In addition, the work required for such ushers is often only at certain times during or prior to the performance. Thus, at other times, these ushers do not have any related work to do but most likely have to be on the payroll nonetheless. Such special personnel also add undesired discourse, walking around and hand waving which are also often sources of distraction. Also, such ushers are often in the way of patrons who are walking in and through the seating area or adjacent areas. Since most halls do not have the same layout of seating, and refreshment stands, first time clients find difficulty in locating them. These clients thus wander about the hall premises looking for these facilities or to find an employee to ask. This is undoubtedly frustrating and time wasting. It is also a preferred business practice to keep track of the total number of occupied seats and thereby the total number of sales. Keeping records helps inform newly arrived clients if a show is sold out or there are available spaces. Management of halls are kept abreast with latest information on income generated by a performance (if it is a saleable one). This activity is usually often attempted by ushers by undertaking manual counting, a very error-prone, distracting and tiring event. Lastly, there are often tiffs between ushers and newly come patrons centered on the recommendation of the former and the preference of the latter.

Some developed prior art devices used to keep track of seat occupied or vacant information are basic composed of a system of switches and lengthy wiring, a method not easily portable to some hall settings.

1.2 Statement of Objectives

The ATOMS is aimed at designing a dynamic seat occupancy monitoring system for use in auditoriums, halls, theatres and the like. ATOMS will improve on the existing schemes involved in monitoring the presence of a seat occupant (or the lack thereof) in large public rooms. The system should handle this function efficiently, accurately, neatly and cost effectively.

The wireless scheme of communication integrated with the system should help make use of minimal extra hardware like cables and should be easy to implement. The system operation must be portable to already-built seats and should be portable. It should also be capable of managing the seating in very large halls. Finally, ATOMS should eliminate the need for ushers.

1.3 Organisation of Project Report

This project report is segmented into six parts. The first part presents a brief introduction to the concept and study. Problems associated with operations of theatres and auditoriums are outlined and discussed here as well. Additionally, a summary of the objectives of the study are then spelt out. A brief overview of the entire project report is also described.

The second section reviews existing literature on the design of seat occupancy monitoring systems or and/or other related systems. It also highlights the flaws of these existing inventions.

The complete design of the dynamic seat occupancy monitoring system presented in detail in third part. How seat occupancy is determined and communicated is fully explained and also the channel via which this information exchange is done is briefly examined. The schematics of all building blocks of the system parts are presented. The ATOMS hardware and its underlying software are also designed and implemented.

Section four examines testing and subsequent evaluation of test results of operations of the system level designs developed in the third section.

The ATOMS project is concluded and recommendations for future work are provided in the fifth division of this project report.

All schematics and other diagrams exactly as designed in Microsoft Visio 2013®, EAGLE 7.1.0®, Visual Paradigm CE 12.0® and other development tools are included in the Appendix. References made in this project are included in the final section of this report.

2. LITERATURE REVIEW

There have been several endeavours by engineers to provide solutions to the above stated problems associated with dynamically monitoring seat occupancy in theatres, cinemas and the like. Consequently, some prior art devices have been developed over the past seven decades in attempts to accurately and efficiently keep track of seat occupied or vacant information. Earliest descriptions appeared in papers by the Gauglers in 1941. Examples of these systems are discussed below.

2.1 The Gauglers' System (U.S. Pat. No. 26 751 039 A)

This invention, designed in 1939 by Daniel and Elizabeth Gaugler, relates to a chair for motion picture theaters and principally to a seat having an illuminated signal which is lit when the seat is vacant so as to facilitate the seating of patrons in the dark theater.

The principal object of the invention is to provide an individual signal light for each seat which automatically lights when the patron rises and leaves the seat and which goes off when the seat is occupied. This signal being visible at a distance enables patrons to quickly find suitable seats. The seat indicator which is visible from the rear is mounted in the back of each seat and above the level of the knees of the seated clients. A signal light which is also constructed to be visible from the front of the empty seat and also from the top thereof, a single bulb (embedded within the seat) being employed to illuminate the rear, front and top signals, and the front signal being so constructed as to not interfere with the comfortable seating of the patron.

The invention also provides such an empty seat indicator in which the necessary switch and spring means which cooperate with the swinging seat are mounted on the back of the theater seat, there being no parts which interfere with the movement of the patrons along rows of seats and there being no parts which interfere with the feet of the seated clients or with the cleaning of the floor under the seats. In order not to disturb the patrons when a large number of seats are empty, the system ensures that at the availability of many seats, the empty seat indicators each be rendered inoperative by opening a master switch.

Flaws of the Gauglers' System

- The system has redundant individual seat indicator lighting schemes at the front, top and rear.
- Indicator redundancy results in energy wastage.

- Multiplicity and position of indicators especially the ones affixed on the top side of seats could be distracting and might be harmful to eye health.
- The system can hardly be applied to “already-made” theatre seats without compulsorily dismantling the seats.

2.2 The Helbling and Glass System (U.S. Pat. No. 5 797 126)

The Helbling and Glass system is an automatic theater ticket concierge in which individual ticket booth are in wireless communication with a central post. This arrangement uses the sale of tickets to manage seat occupancy. It displays the view from particular seats as well as the layout of the seats and the interior of the facility. The system utilizes a display panel with touch screen capabilities and displays a scene exemplary of the particular event at the theater.

Flaws of the Helbling and Glass System

- The Helbling and Glass system does not have a provision that indicates seat vacancy per se.
- The system does not conclude vacancy utilizing input directly from a seat occupant (or lack thereof) but rather uses seat availability information accrued from the ticket sales booths.

2.3 The Baron and Abbott System (US. Pat. No. 6 140 921)

Designed and published in October 2000, this indicator and display system provides information regarding occupancy on individual seats and a display providing information relating to seat occupancy and relating to the seating area. The system includes multi-colored lights mounted on the tops of the seats for indicating the occupancy thereof. The lights are controlled by sensors mounted in the seats and manual switches mounted on the seats. The system also includes a display panel located in the lobby or near the theater or auditorium which shows which seats are occupied and which are vacant. The display panel also numerically shows the seat availability and also shows in hologram form the seating arrangement and the location of restrooms, refreshment counter, etc. relative to the seating area as well as aisles and other information of use to the patrons.

Flaws of the Baron and Abbott System

- Installation costs are inordinate due to hologram inclusion and massive electrical wiring for power supply and control.
- There is high possibility of obstruction in the aisle due to lengthy electrical wiring which is likely to become a jumbled mass if proper care is not meted.
- Carpeting used to shroud is likely to be lumpy when they are used to cover or hide the wires.
- Troubleshooting of hitches are difficult due to the lengthy jumbled wires used in the system's implementation.
- The use of multiple indicators per seat isn't energy efficient.

Figure 1: Isometric view of the Baron et al seat.

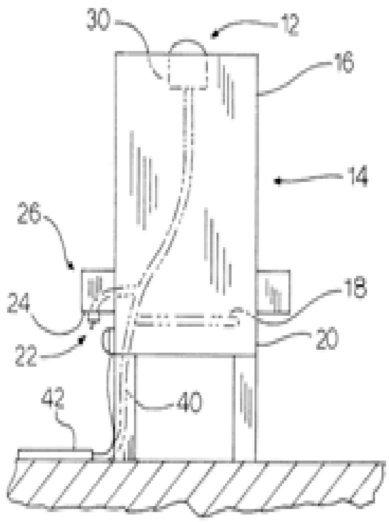


Figure 2: Elements within the Baron et al seat.

3. METHODOLOGY

3.1 Brief Description of the Several Views of the Drawings

The sizes and relative positions of elements in the drawings found in this document are not necessarily drawn to scale. For example, the shapes of various elements and angles are not drawn to scale, and some of these elements are arbitrarily enlarged and positioned to improve drawing legibility. Further, the particular shapes of the elements as drawn are not intended to convey any information regarding the actual shape of the particular elements, and have been selected solely for ease of recognition in the drawings.

3.2 System Communication

The ATOMS project has as its pivot, the objective to accurately, neatly, cost effectively and efficiently gather occupancy data concerning seats in a seating area of an auditorium. This data is then to be processed by a computing device (base station) into an intelligible and graphical form for the benefit of all patrons of such enterprises. As opposed to the older method of wiring up a whole theatre for power supply and control, ATOMS proposes the use of minimal amount of wiring for power and the application of a wireless technology for communication between sensor units that gather and encode data and the computing device via radio waves.

The quality and range of a wireless transmission is dependent on a number of key factors. These include, but are not limited to, the physical environment in which the transmissions take place and the amount of noise or electromagnetic interferers within the vicinity of the transmissions. The following provides some general remarks to consider when evaluating the quality and range of a wireless transmission.

Amongst the assorted wireless technologies at the disposal of today's world like UWB, Zig Bee, WiMAX and Bluetooth, Wi-Fi was chosen to serve this purpose because of the following reasons:

- Wi-Fi is unparalleled in its mobility.
- Wi-Fi provides secure wireless solutions that support the growth and release of a prototype mobile ad hoc wireless network.
- It is a very fast wireless communication technology.
- The Wi-Fi technology has considerable range and channel capacity.
- It is relatively easier and cheaper to implement as compared to the other technologies.
- Wi-Fi is relatively flexible and widely used.

  A Wi-Fi network (based on IEEE 802.11 network protocols) provide wireless connectivity between the sensor unit and the base station over a set-up local area network. This enables seat units to be added or removed from the system without modifying it. Accurate data transmission is ensured since Wi-Fi has built in data-packet building capability and employs CSMA/CA to avoid packet collisions. The topology of the network is a star arrangement. In the above configuration the access point (gateway unit) serves as a controller at the heart, with end devices connected around it. The base unit can double as the access point. The ATOMS project boards are configured such as not to communicate with each other but each with the base station only over the network. Thus Individual sensor units on the network can also be identified and monitored independently by the base station.

3.3 System Architecture

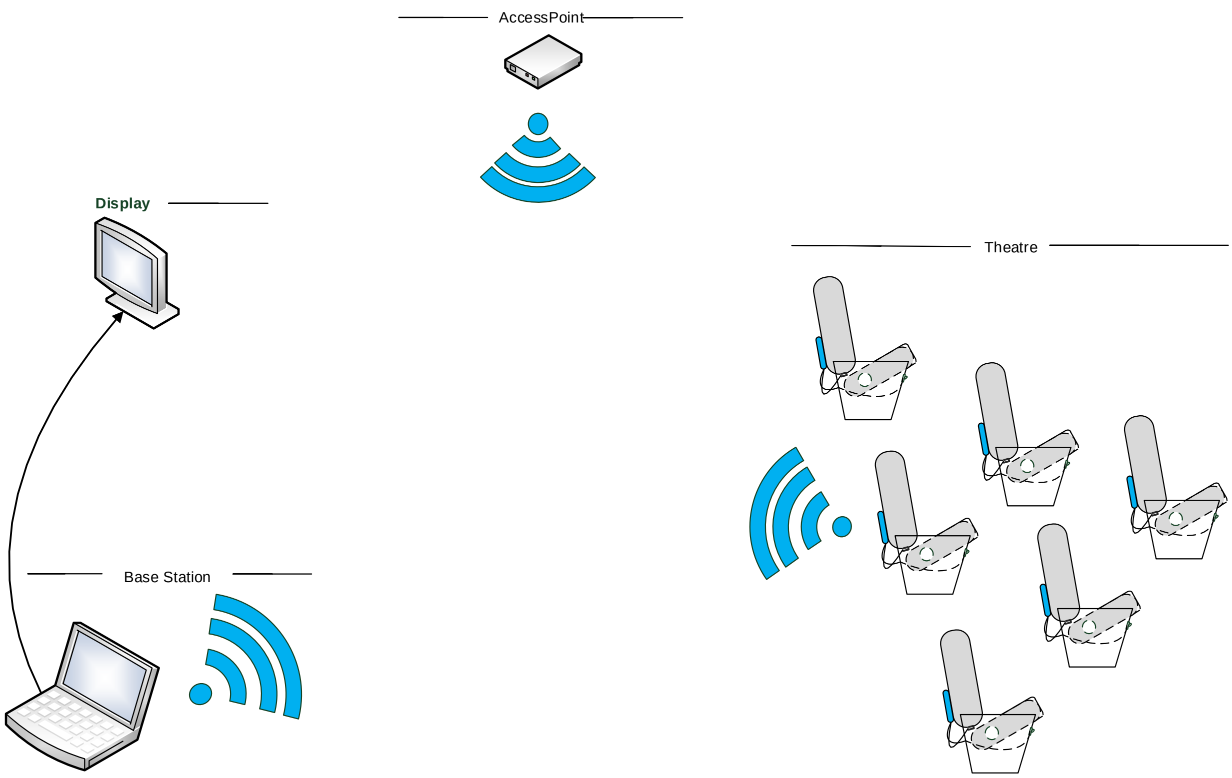


Figure 3: System Architecture.

Figure 3 shows a visual overview of the ATOMS system. The system comprises of basically portable seat units affixed to all the seats in the auditorium or theatre. There is also an access point that act as a gateway device for the wireless local area network over which system communication occurs. It routes all messages sent from the base station to the seat units and from the seat units to the base station. The use of this hardware however depends on the implementation as ATOMS can function without it (that is when the base station doubles as the access point). The system also has as one of its basic components, a base station and lastly a display devices which is used to show the occupancy status of all seats in the auditorium on a layout analogous to the seat layout in the seating area.

3.4 Design and Implementation

ATOMS includes hardware, desktop and embedded wireless software applications based on Texas Instruments' (TI's) CC3200 chip.

3.4.1 Base Station

The base station is basically a computing device. A desktop or laptop computer can be used. However a laptop was used because of its portability. A key requirement of this computing device is that it must possess a Wi-Fi interface with which it can connect to and communicate over the wireless network. If the computer lacks this, a detachable modem can be used for this purpose.

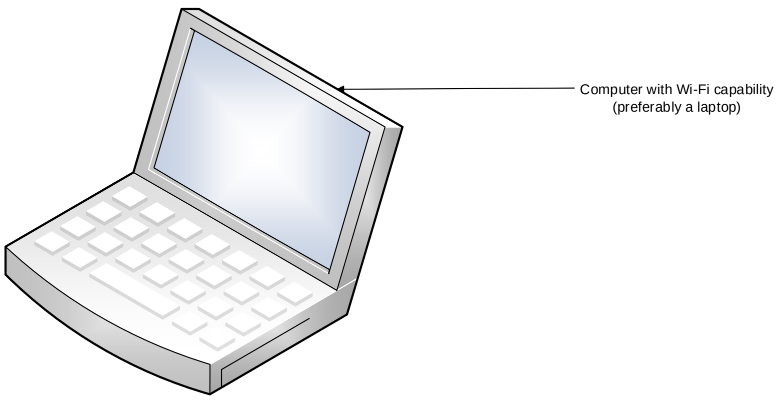


Figure 4: The Base Station.

3.4.2 Seat Unit

The seat unit is composed of two electrically connected sub components. These are a capacitive proximity sensor and an ATOMS microcontroller board. These two components, affixed to the seat, are responsible for sensing a human occupant and then communicating the occupancy status of the chair to the base station for processing and display.

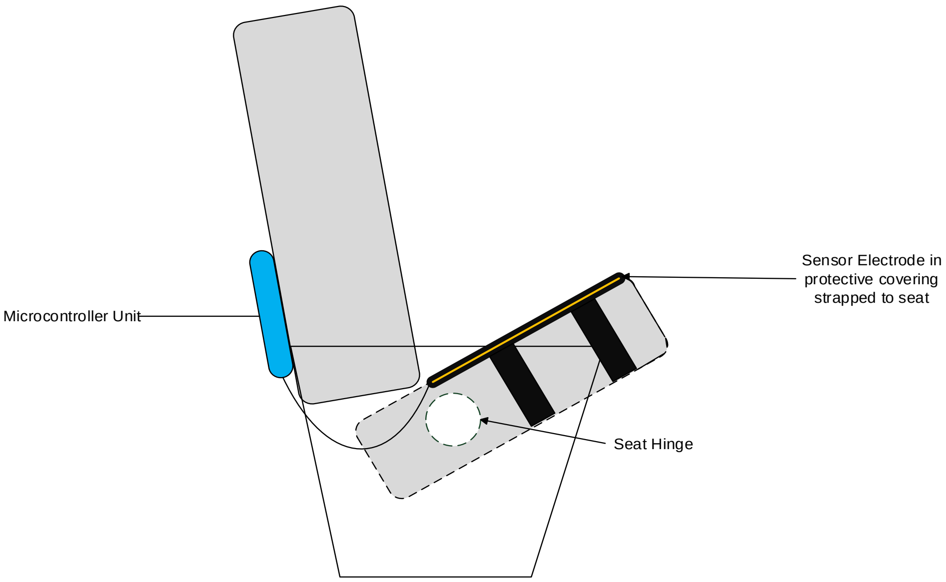


Figure 5: A seat and its seat unit.

3.4.2.1 Capacitive Proximity Sensor

Capacitive proximity sensors provide very reliable proximity detection with low power, low cost and fairly easier design. These sensors operate by producing an electrostatic field as a result of materials or objects that are brought close to them. The idea of a capacitive proximity sensor is a compromise of the traditional two plate capacitor. Capacitance is defined as the amount of electrical charge, which changes with voltage, stored between two non-connected conductive plates. This quantity depends mainly on the dielectric substrate, the size of the plates and the distance between them. The capacitance is thus calculated using the formula

$$C = \frac{A\varepsilon_o\varepsilon_r}{d} = \frac{\pi r^2\varepsilon_o\varepsilon_r}{d}$$

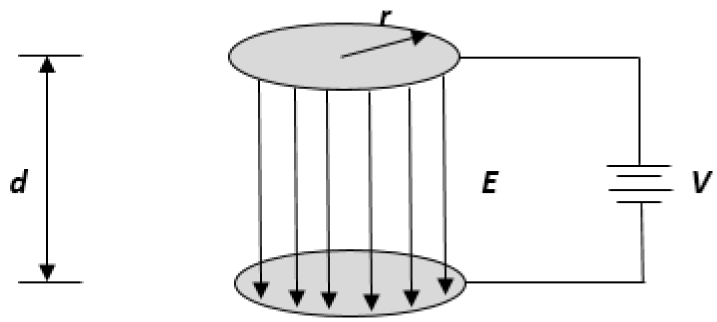


Figure 6: A basic two parallel plate capacitor.

E = Induced electric field.
V = Potential difference applied across the two parallel plates.

A = Overlapping an area.

εr = Relative permittivity or dielectric constant.

ε_O = Permittivity of free space or vacuum = $8.85 \times 10^{-12} F/m$.

The capacitive proximity sensor used in the ATOMS project, unlike the traditional two plate capacitor, has only one electrode, a circular aluminium foil. This foil is used rather than a more known rectangular plate because circular ones reduces fringing effect thus decreasing stray capacitance that appear around the edges of the plates. There is some field outside that plates that curves from one to the other.The other plate of the capacitor is provided by the water in the body of the expected human occupant. The insulator of the capacitor becomes a combined effect of the human skin, muscle, bone, cloth etc.

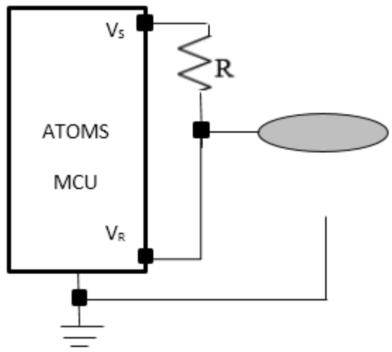


Figure 7: Capacitive proximity sensor setup.

Sensor setup

Two pins of the ATOMS project MCU are setup to be in one of two modes; the stimulus pin, which changes to a new state and waits for the response pin to change to the same state as the stimulus pin. The sensor setup requires a high value resistor (**R**) connected between the stimulus pin (**V_s**) and the sensor electrode. The high value resistor guarantees a sufficient charge/discharge time. The electrode is the sensor component to detect human proximity and is connected to the response pin (**V_R**). This connection is done via a copper wire which taps the wire linking the high value resistor and the foil. The sensor electrode (metal plate) is covered by a piece of non-conducting material (shown in Figure 9), which will be strapped to the seat (as shown in Figure 5).

Sensor operation

In measuring the capacitance, the stimulus pin is set as output and response pin as input. A timer is then started and the stimulus pin is set to logical high (**V_{cc}**) which charges the capacitor until the response pin crosses the threshold logical high (**V_{ht}**). The timer is then briefly paused while the response pin charges completely to **V_{cc}**. The timer is continued when the stimulus pin is set to logical low (**V_{low}**) which discharges the capacitor until the response pin crosses the threshold logical low (**V_{lt}**). The timer determines the time it takes for the response pin to charge to the logical opposite and resets each time $\mathbf{V_s}$ is set to $\mathbf{V_{cc}}$. The change in capacitance can then be deduced from the timer value before each reset occurring at the end of a cycle. The process is the process is then repeated.

The value of the timer is directly proportional to τ, the RC time-constant. Since the resistor value, **R** stays constant, the delay (time constant, τ) depends solely on the capacitance, **C**.

$$\tau = \mathbf{R \times C}$$

When an occupant (the human body is composed of about 70% water and thus a good electrical conductor) comes near the electrode of the sensor, the separation, **d** between the overlapping conductors decrease. This increases the capacitance $\mathbf{C_2}$ between the sensor electrode and ground will increase the total capacitance measured. The RC time constant and thus the capacitor charge/discharge time increases. It is thus possible to deduce the occupancy of a by interpreting the timer value with regard to the predetermined detection threshold.

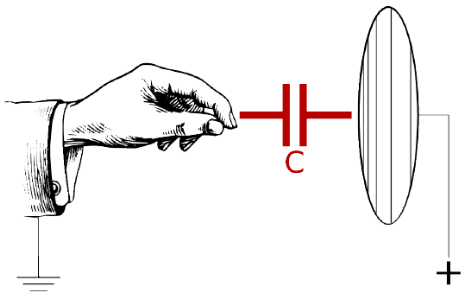


Figure 8: Interaction of the human body with the capacitive electrode.

(Credit: Wimmer, 2011 "Capacitive Sensors for Whole Body Interaction".)

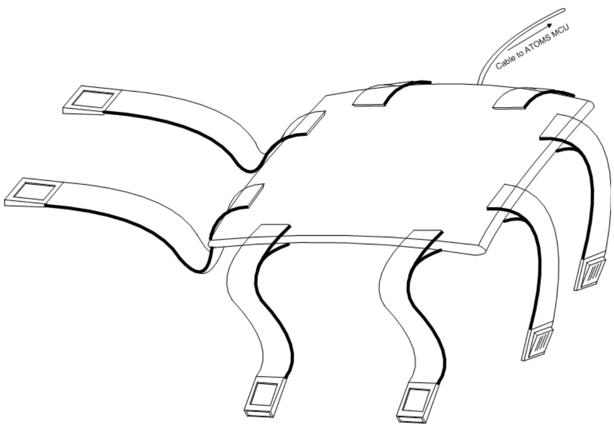


Figure 9: Protective covering for sensor electrode.

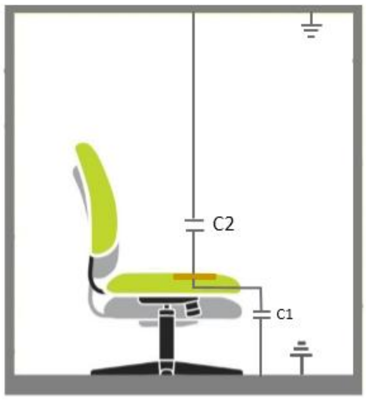


Figure 10: A model of capacitances between sensor electrode and auditorium building.

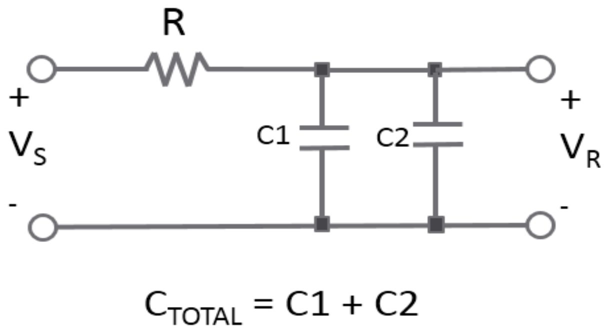


Figure 11: Circuit representation of the capacitive sensor.

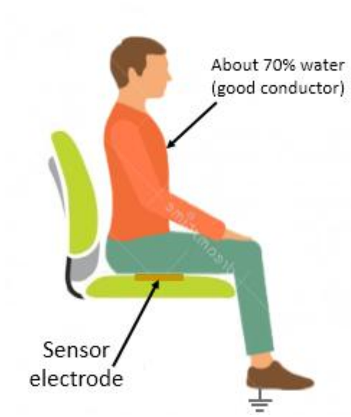


Figure 12: A seat with an occupant.

3.4.2.2 The ATOMS Microcontroller Board

The ATOMS project uses a custom designed microcontroller as a second subcomponent of the seat unit.

Note: For schematics and layout and BOM for the ATOMS Board refer to *ATOMS Board Technical Document* in the appendix.

TI CC3200 Key Features

The *ATOMS Wi-Fi Module* is mounted on the ATOMS Board. This module includes the embedded *TI CC3200 Wi-Fi processor* and external circuitry required for its operation.

The following are some key technical features of the TI CC3200 chip:

- Application Microcontroller Subsystem
  - ARM® Cortex®-M4 Core at 80 MHz
  - 256kB RAM
  - 32-Channel Direct Memory Access
  - UART
  - General-Purpose Timers with 16-Bit PWM Mode
  - Watchdog Timer
  - 4-Channel 12-Bit ADCs
- Clock Source
  - 40.0-MHz Crystal with Internal Oscillator
  - 32.768-kHz Crystal or External RTC Clock
- Package and Operating Temperature
  - 0.5-mm Pitch, 64-Pin, 9-mm × 9-mm QFN
- Wi-Fi Network Processor Subsystem
  - 802.11 b/g/n Radio
  - Simultaneous TCP or UDP Sockets
  - TX Power
    - ✓ 18.0 dBm @ 1 DSSS
    - ✓ 14.5 dBm @ 54 OFDM
  - RX Sensitivity
    - ✓ -95.7 dBm @ 1 DSSS
    - ✓ -74.0 dBm @ 54 OFDM
  - Application Throughput
    - ✓ UDP: 16 Mbps
    - ✓ TCP: 16 Mbps
- Power-Management Subsystem
  - Integrated DC-DC Supports a Wide Range of Supply Voltage:
    - ✓ V_{BAT} Wide-Voltage Mode: 2.1 to 3.6 V
    - ✓ Preregulated 1.85V Mode
  - Advanced Low-Power Modes:
    - ✓ Hibernate: 4 µA
    - ✓ Low-Power Deep Sleep (LPDS): 250 µA
    - ✓ RX Traffic (MCU Active): 59 mA @ 54 OFDM

▪ Ambient Temperature Range: -40°C to 85°C	✓ TX Traffic (MCU Active): 229 mA @ 54 OFDM, Maximum Power ✓ Idle Connected (MCU in LPDS): 825 µA @ DTIM =1

For a full technical description, refer to the TI CC3200 Datasheet (www.ti.com).

Figure 13: Key features of the CC3200.

ATOMS Board Components

The following figure highlights some of the key components of the ATOMS Board.

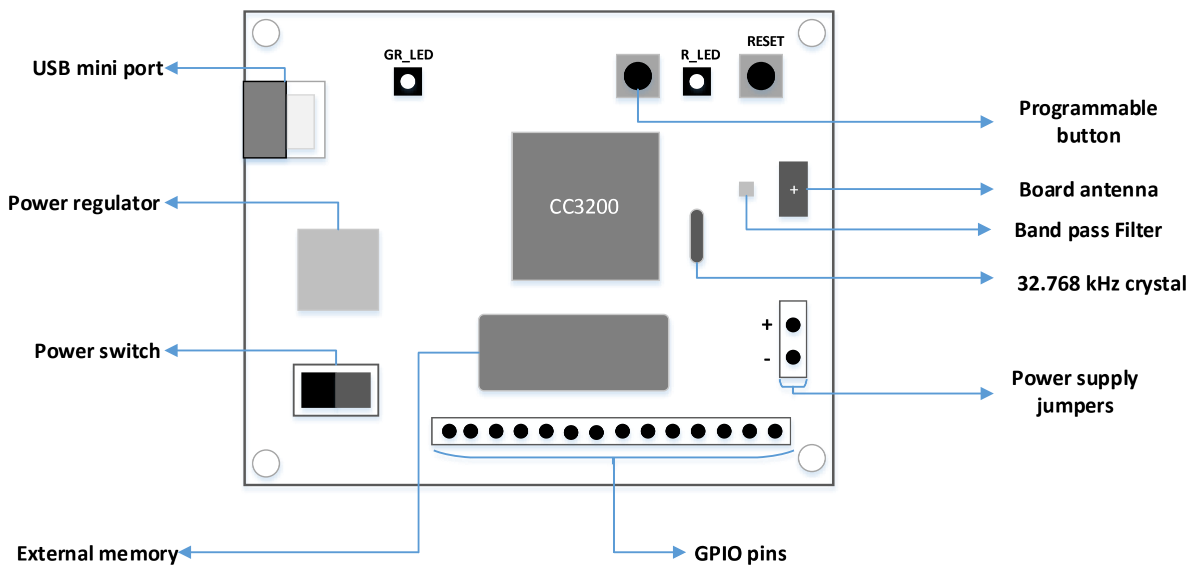


Figure 14: ATOMS Board key components.

The following table describes the components of the ATOMS Board.

Part	Description
ATOMS Wi-Fi Module[1]	Complete Wi-Fi package, including CC3200 network processor and external components. Operates between 1.8 V and 3.0V.
TI CC3200 microcontroller	A low power CMOS 8-bit microcontroller with 128 Kb Flash and 256 Kb SRAM. This microcontroller has subsystems that drive the operations of the ATOMS Board. For

	more details, refer to the TI CC3200 datasheet available from Atmel. Operates between 1.8 V and 3.6 V.
Power Supply	Device is powered by 2 AA batteries (3.0 V unregulated power supply) or via micro USB (V_{BUS} = 5.0 V during flashing).
Battery connector pin (J20)	Connector for a 3.0 V battery pack.
Power ON/OFF switch	Used for manual disconnection of board subcomponents from power supply.
R_LED LED	When illumined, indicates power supply is on.
DC-DC converter (voltage regulator)	Needed to accommodate the differing voltage or current requirements of the system.
O_ LED	When lit, indicates transmission and reception activity.
27 User programmable GPIO pins	GPIOs for easy connection to sensors and actors.
Two programmable buttons	The ATOMS Board provides two user-programmable buttons that connect some of the I/O pins to ground.
Reset button	Used to reset the microcontroller.
32.768 kHz quartz	A 32.768 kHz quartz for the timer/counter oscillator.
40 MHz external oscillator	A 7.3728 MHz quartz crystal as an external oscillator.

[1] is embedded in the TI CC3200 microcontroller

Figure 15: ATOMS MCU components.

Power Supply

The ATOMS Board operates at an average voltage of 3.0 V. However, since the ATOMS Board has a power management system to up-convert and also regulate voltages, a power supply between 1.85 V and 3.6 V is acceptable. The most preferred power supply is a 3.0V power supply provided by a 3.0V rated button cell lithium battery (preferably CR2032 battery). The power management subsystem has in-built reverse voltage protection to prevent the battery from being plugged in the reverse manner. The power management subsystem which receives power from the connected voltage source to the board before disbursing it to the various components of the board. It regulates power to the various components on the board. This makes sure that no component is supplied power more than required which might consequently destroy it. The power management subsystem contains

DC-DC converters to cater for the differing voltage or current needs of components of the system. It is responsible for handling the changing of the various power modes of the board. These power modes and their descriptions are shown in Figure 19 (in the order of increasing power consumption). Caution was taken so that the battery pack and the USB cable were not used at the same time. Personal injuries or explosions could occur and board would have be irreversibly damaged.

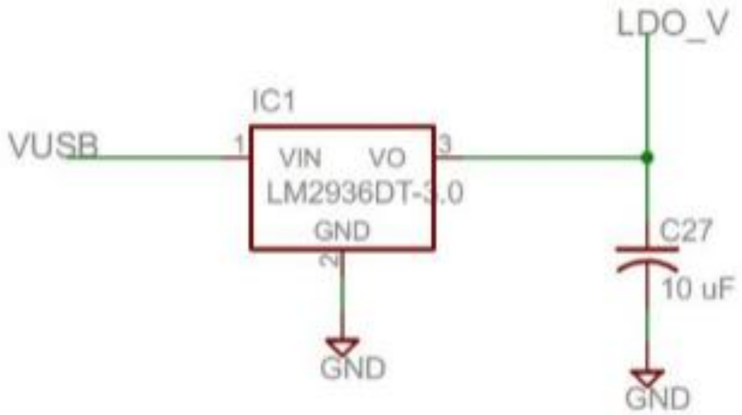


Figure 16: Source voltage regulation circuitry.

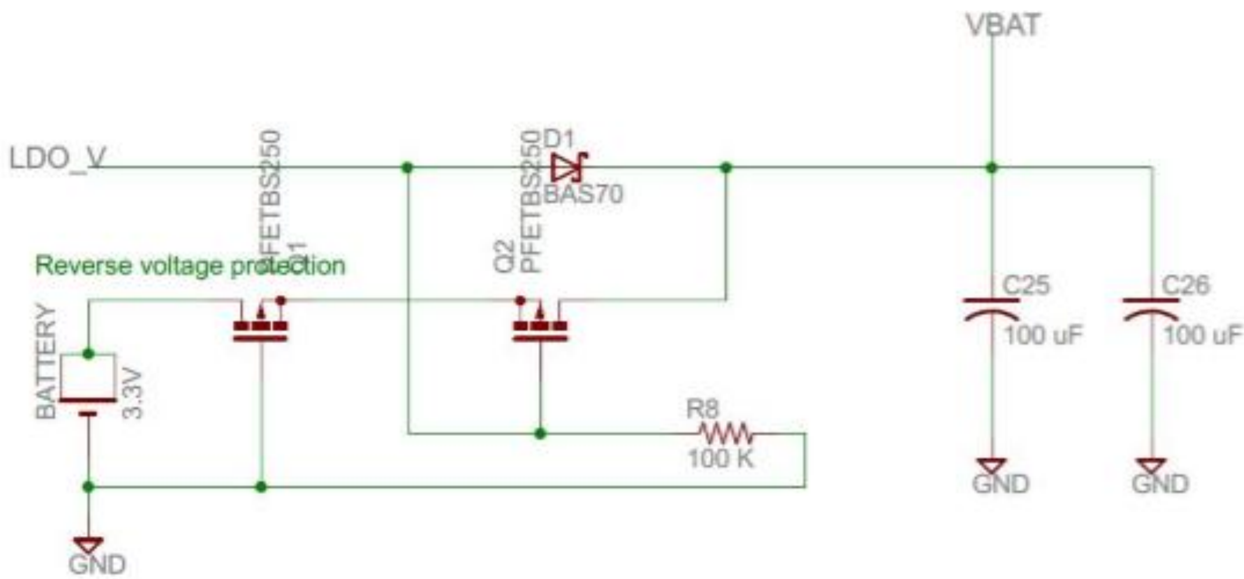
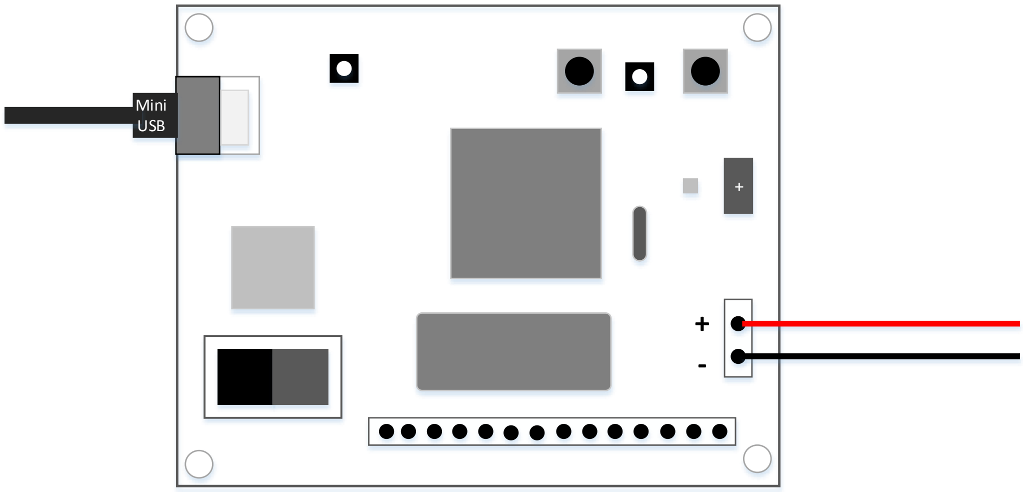


Figure 17: Source voltage connection and protection circuitry.



Warning: USB power should not be used alongside external battery power.
This might cause permanent device damage and personal injuries.

Figure 18: Alternative power supply to ATOMS device.

Low-Power Operating Mode

From a power-management view point, the CC3200 device and hence the ATOMS Board comprises the following two independent subsystems:

• Cortex-M4 application processor subsystem

• Networking subsystem

Each subsystem operates in one of several power states. The Cortex-M4 application processor runs the user application loaded from an external serial flash. The networking subsystem runs preprogrammed TCP/IP and Wi-Fi data link layer functions.

MCU Power Mode	Description
Hibernate mode	This is the lowest power mode in which all digital logic is power-gated. Only a small section of the logic directly powered by the input supply is retained. The (RTC) clock keeps running and the MCU supports wakeup from an external event or from an RTC timer expiry. Wake-up time is longer than LPDS mode at about 15 ms plus the time to load the application from serial flash, which varies according to code size. In this mode, the MCU can be configured to wake up using the RTC timer or external event on a GPIO (GPIO0–GPIO6).

LPDS mode (Low-Power Deep Sleep mode)	State information is lost and only certain MCU-specific register configurations are retained. The MCU can wake up from external events or by using an internal timer. (The wake-up time is less than 3 ms.)Certain parts of memory can be retained while the MCU is in LPDS mode. The amount of memory retained is configurable. Users can choose to preserve code and the MCU-specific setting. The MCU can be configured to wake up using the RTC timer or by an external event on specific GPIOs.
Sleep mode	Here the microcontroller clocks are gated off however the entire state of the device is preserved. This mode offers instant wakeup. The device can be configured to wake up by an internal fast timer or by activity from any GPIO line or peripheral.
Active mode	Here, the microcontroller unit of seat unit executes code at 80MHz state rate.

Figure 19: Application MCU modes.

NWP Mode	Description
Network active mode processing layer 3, 2, and 1	Transmitting or receiving IP protocol packets
Network active mode (processing layer 2 and 1)	Transmitting or receiving MAC management frames; IP processing not required.
Network active listen mode	Special power optimized active mode for receiving beacon frames (no other frames supported)
Network connected Idle	A composite mode that implements 802.11 infrastructure power save operation. The CC3200R network processor automatically goes into LPDS mode between beacons and then wakes to active listen mode to receive a beacon and determine if there is pending traffic at the access point. If not, the network processor returns to LPDS mode and the cycle repeats.
Network LPDS mode	Low-power state between beacons in which the state is retained by the network processor, allowing for a rapid wake up.
Network disabled	

Figure 20: Networking subsystem modes.

On/Off Switch and Power LED

The power switch when slid towards the ON label allows current from the voltage source to the board thus turning it on. When a power supply is provided to the ATOMS Board, the power supply LED (labelled R_LED) is lit. To switch off the board without having to remove the power supply, a sliding switch is slid back to the OFF position.

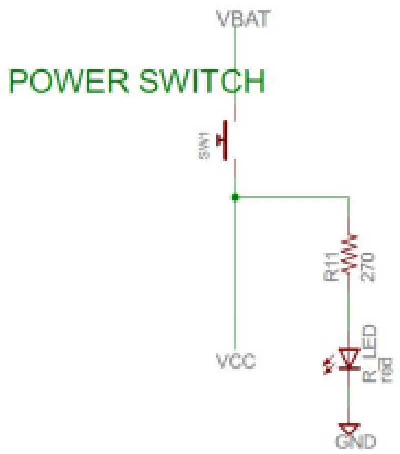


Figure 21: ON/OFF Switch and Power LED schematics.

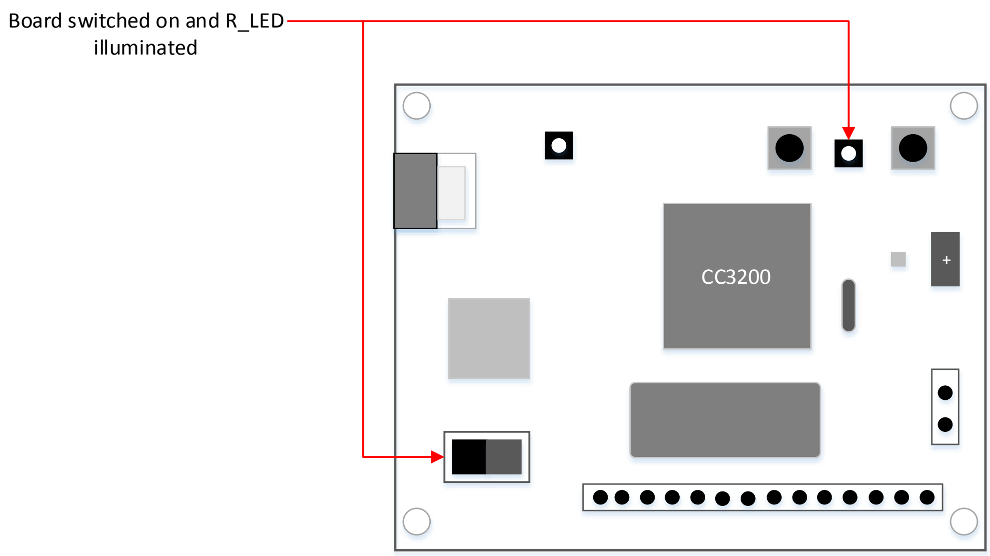


Figure 22: Power switch and indicator.

ATOMS Board Dimensions

The following figure shows the dimensions of the ATOMS Board.

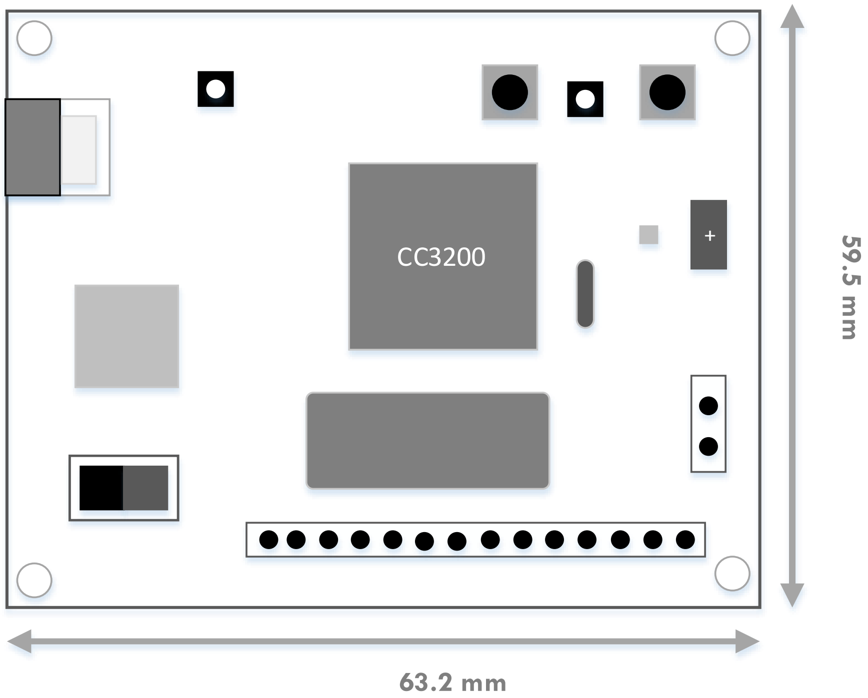


Figure 23: ATOMS Board dimensions.

Block Diagram

A block diagram of the ATOMS Board is provided below. It puts out clearly the various subsystems which coordinate to produce the required result.

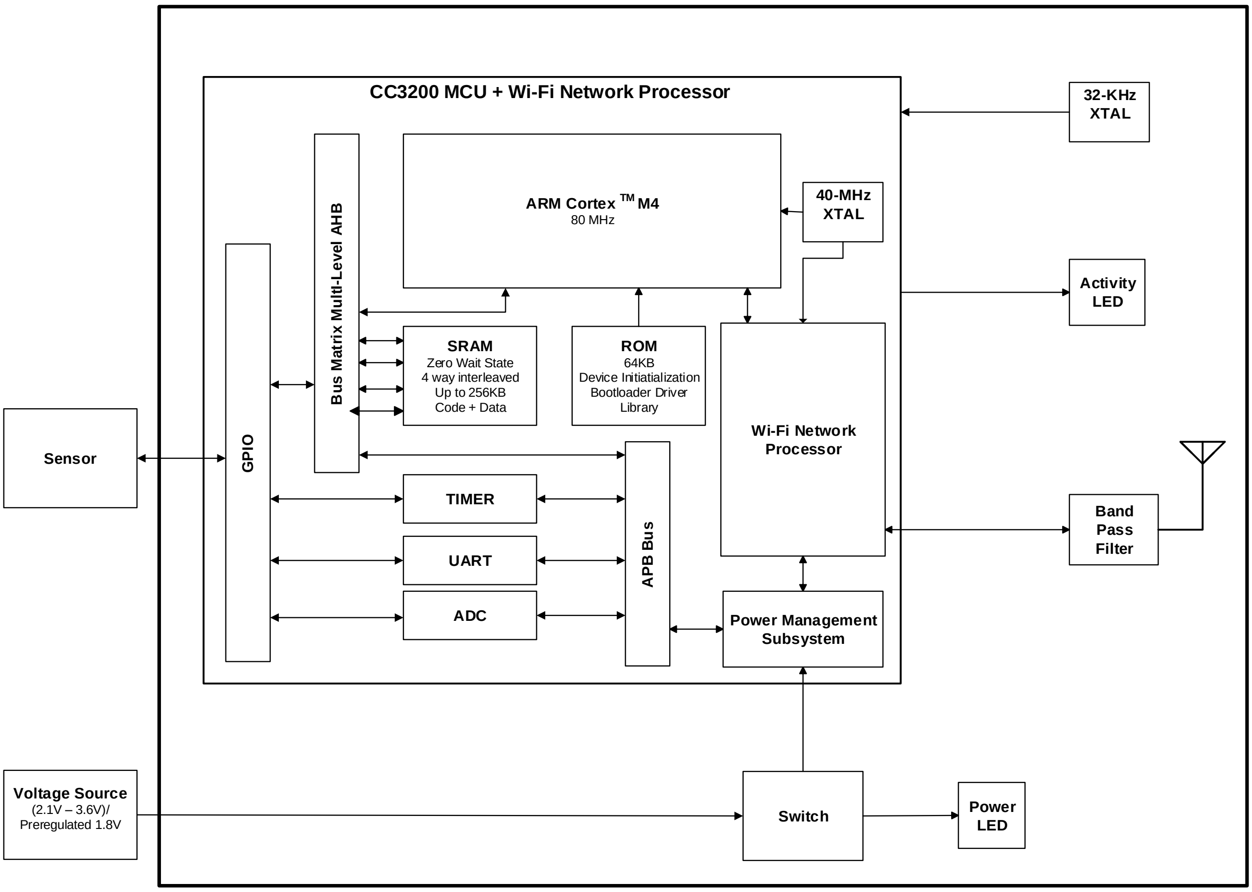


Figure 24: ATOMS Board block diagram.

ATOMS Board Modules

Micro USB Interface

The USB interface of the ATOMS Board can serve two uses. It primarily serves as an interface through which a written embedded C code is loaded unto the TI CC3200 microchip so as desired on-board logic functions can be undertaken. This loading is done via the UART component of the board thus requiring the services of a USB-UART preprocessing device to handle data streams. The V_{BUS} pin of the USB supplies

5V to the board when the C code is being the ATOMS software flashing. This voltage is however regulated to safeguard the ATOMS board.

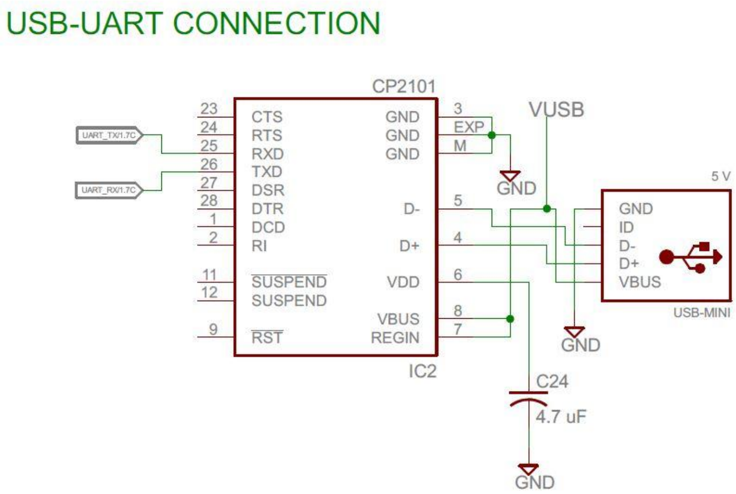


Figure 25: The USB-UART connection schematics.

Digital I/O Module

The ATOMS Board provides a digital I/O module that extends the four external ports of the TI CC3200 microcontroller. In the ATOMS Board, these I/O ports are used as follows:

- PORT1 (General Purpose Input/Output - GPIO)
- PORT2 (LEDs)
- PORT3 (Buttons)
- PORT4 (ADC)

PORT1 (General I/O)

PORT1 has 27 GPIO pins (Pin 0 and pins 3 to 30) and can be used for connecting sensor and actors. Two pins labelled V_{cc} and Ground are allocated to the provision of power supply.

The PORT1 pin connector is shown below:

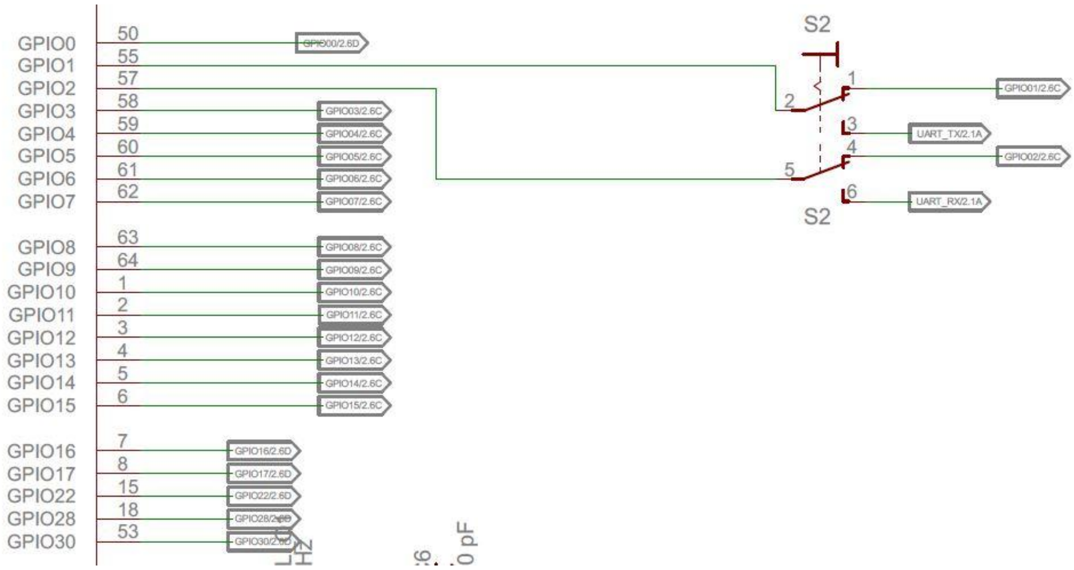


Figure 26: GPIO pins schematics.

PORT2 (LED Block)

PORT2 on the TI CC3200 microcontroller is connected via a 270 Ω resistor array to four activity LEDs and a power indicator LED (R_LED).

The PORT2 LEDs are shown below:

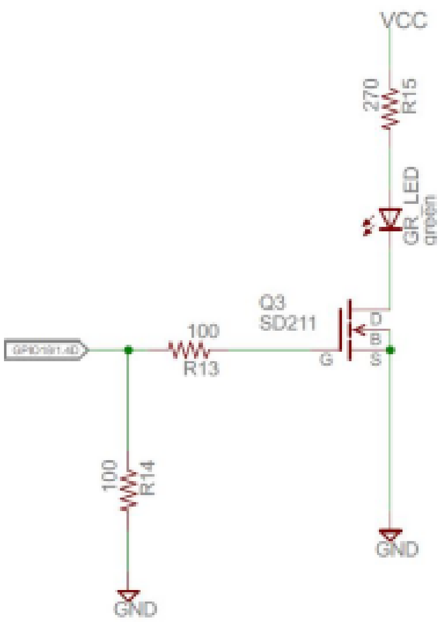


Figure 27: PORT2 green LED schematic.

	Microcontroller LEDs	
LED #	**Colour**	**Signal**
1 (labelled R_LED)	Red	Power supply indicator.
2 (labelled ACT_LED)	Orange	Lit to indicate attempts to connect to access point. Used to also indicate occurrence of other activities such as sending and receipt of packets.
3 (labelled GR_LED)	Green	Lit to indicate normal running of ATOMS device.

Figure 28: Microcontroller LEDs.

PORT3 (Buttons)

PORT3 pins to two buttons. Pin 1 and 2). The buttons on the board are shown below:

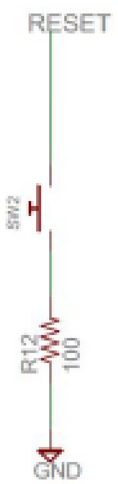


Figure 29: Reset Button Schematic.

Microcontroller	Buttons
Button #	**Signal**
1 (labelled POWER)	This button when pressed powers on the ATOMS devices.
2 (labelled RESET)	It is used to reset the microcontroller.

Figure 30: Microcontroller buttons.

PORT4 (ADC)

PORT4 on the TI CC3200microcontroller is connected to the ADC/IO analog pin connector. The ADC I/O pin connector provides an analog input to the A/D Converter.

Filter and Antenna

As shown in Figure 31, a band pass filter is connected between the output of the TI CC3200's Wi-Fi network processor and the on-board antenna is used to cut out signals with frequencies outside the acceptable band. Thus all signals outside the IEEE 802.11n band of center carrier frequency of 2.4 GHz are discarded whiles those which fall within are allowed to pass and subsequently processed by the network processor. A RAM for use by the processor is also indicated in the diagram.

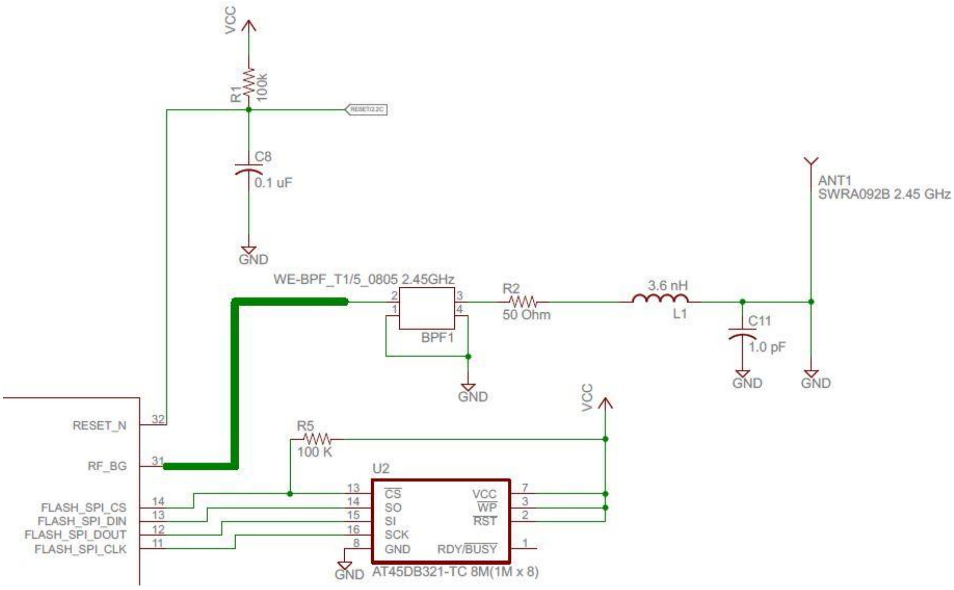


Figure 31: ATOMS on-board pass band filter and antenna.

Schematic of the ATOMS Board

Figures 38 and 39 in the appendix of this document shows the schematics of the ATOMS project (Refer to *ATOMS Board Technical Document*). The schematics were developed using the EAGLE® development tool .These reveal extensively, the various circuitry components of the board and their configuration.

Figure 32 shows the 3D-simulation of the ATOMS Board generated from the board's schematic. This simulation gives an impression of how the board would look after it has been manufactured. Here the layout of all the components are presented. It shows all the GPIOs on the left of the board and the CC3200 microcontroller in the "brain on board", amongst other components. Figures… in the Appendix shows the layout of components on the two layers of the board, the top and bottom layers, and how they are routed together. Two layers are used in order to prevent the board from control heating during operations.

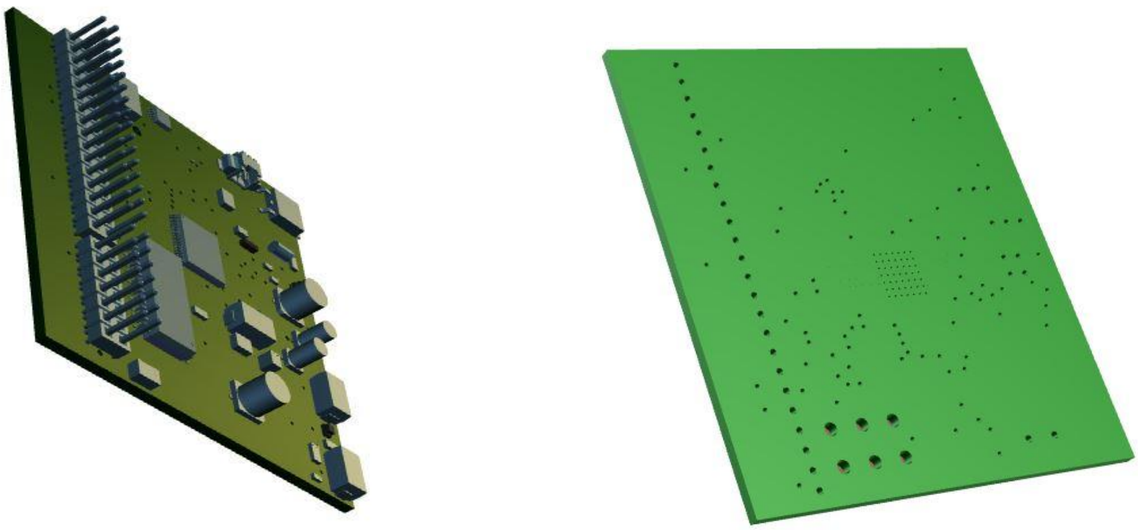


Figure 32: A 3D simulation of the ATOMS Board.

3.4.3 Access Point

The access point is a common piece of wireless equipment. This special wireless station receives radio transmissions from a wireless transmitter on a network and then forward the signals to the rest of the network. Figure 33 shows a standalone access point. The access point functionality can be found in a detached device or in a computer (base station) that has a wireless network adapter along with a special management software or bundled in other devices like cable modems.

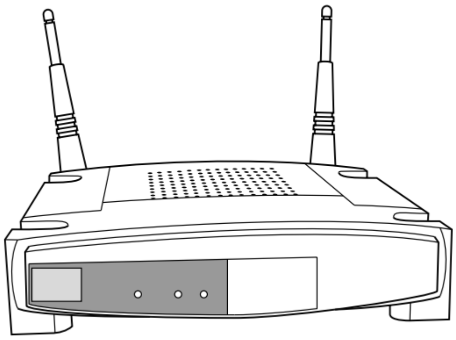


Figure 33: A standalone access point.

3.5 Software

The software involved in the operations of the ATOMS system can be categorized into two distinct parts: the embedded system software and the base station application software.

3.5.1 Embedded System Software

The source code that controls the operations on the ATOMS Board was written using the Embedded C language. It was written in the TI Code Composer Studio® (CCS) IDE version 6.1.0. The source code contains logic that facilitates the harvesting of signals generated as a result of the sensor detecting the presence of an occupant. The occupancy information that these signals contain are then extracted and processed into TCP packet payloads ready to be relayed to the base station.

Power on and activity indicators

The code below controls the behavior of the orange LED as the ATOMS device waits for a connection to the access point. The LED continually blinks with a constant delay of one second till connection is successfully established when it is turned off.

```
    while((!IS_CONNECTED(g_ulStatus)) || (!IS_IP_ACQUIRED(g_ulStatus)))
    {
        // Wait for WLAN Event
#ifndef SL_PLATFORM_MULTI_THREADED
        _SlNonOsMainLoopTask();
#endif
```

```
//Blink orange led to show connection to AP
        MAP_GPIOPinWrite(GPIOA1_BASE,GPIO_PIN_2,GPIO_PIN_2);
             MAP_UtilsDelay(1000000);
             MAP_GPIOPinWrite(GPIOA1_BASE,GPIO_PIN_2,0);
             MAP_UtilsDelay(1000000);

    //turn off orange led after successful connection to AP
    MAP_GPIOPinWrite(GPIOA1_BASE,GPIO_PIN_2,0);
    return SUCCESS;

}
```

The green led

```
//Turn on green led to show application is still running
      MAP_GPIOPinWrite(GPIOA1_BASE,GPIO_PIN_3,GPIO_PIN_3);
```

GPIO interrupt generation and handling

The hardware supports 8 bits of interrupt priority hence a macro to generate an interrupt priority mask based on this number of bits. The code below shows the setting up of interrupts of some GPIO pins on PORT 1. The *SetupA1_Int()*function is used to achieve that. The code is written such as to trigger interrupt when any of the stated experiences a rising edge voltage change. This is done using the *GPIOIntTypeSet()*function. The priority level of these pins are set to LEVEL 2 using *the MAP_IntPrioritySet()* function. The *GPIOIntClear()* and *IntPendClear()* functions are used to initially clear all pending interrupts present on the pins and ports respectively. The interrupt utility is enabled for ports and GPIO pins by employing the *IntEnable()* and *GPIOIntEnable()* functions respectively.

```
void SetupA1_Int(long targetPin)
{
      switch(targetPin)
      {
            case 0:
                  GPIOIntTypeSet(GPIOA1_BASE,GPIO_PIN_6,GPIO_RISING_EDGE);
                  GPIOIntTypeSet(GPIOA1_BASE,GPIO_PIN_5,GPIO_RISING_EDGE);
                  MAP_IntPrioritySet(INT_GPIOA1, INT_PRIORITY_LVL_2);
                  GPIOIntRegister(GPIOA1_BASE,A1_IntHandler);
                  GPIOIntClear(GPIOA1_BASE,GPIO_PIN_6);
                  GPIOIntClear(GPIOA1_BASE,GPIO_PIN_5);
                  IntPendClear(INT_GPIOA1);
                  IntEnable(INT_GPIOA1);
                  GPIOIntEnable(GPIOA1_BASE,GPIO_INT_PIN_6);
                  GPIOIntEnable(GPIOA1_BASE,GPIO_INT_PIN_5);
                  break;

            case GPIO_INT_PIN_6:
                  GPIOIntTypeSet(GPIOA1_BASE,GPIO_PIN_6,GPIO_RISING_EDGE);
```

```
			GPIOIntRegister(GPIOA1_BASE,A1_IntHandler);
			GPIOIntClear(GPIOA1_BASE,GPIO_PIN_6);
			IntEnable(INT_GPIOA1);
			GPIOIntEnable(GPIOA1_BASE,GPIO_INT_PIN_6);
			break;

		case GPIO_INT_PIN_5:
			GPIOIntTypeSet(GPIOA1_BASE,GPIO_PIN_5,GPIO_RISING_EDGE);
			GPIOIntRegister(GPIOA1_BASE,A1_IntHandler);
			GPIOIntClear(GPIOA1_BASE,GPIO_PIN_5);
			IntEnable(INT_GPIOA1);
			GPIOIntEnable(GPIOA1_BASE,GPIO_INT_PIN_5);
			break;

	}}
```

The *A1_*`IntHandler()` function is responsible for handling interrupts captured at GPIO pins. It is defined as follows (*defined for just two GPIO pin cases here for documentation purposes*):

```
void A1_IntHandler()
{
	switch(GPIOIntStatus(GPIOA1_BASE,true))
	{
		case GPIO_INT_PIN_6:
			MAP_GPIOIntClear(GPIOA1_BASE,GPIO_PIN_6);
			SetupA1_LowInt(GPIO_INT_PIN_6);
			MAP_GPIOPinWrite(GPIOA1_BASE,GPIO_PIN_6,GPIO_PIN_6);
			break;

		case GPIO_INT_PIN_5:
			MAP_GPIOIntClear(GPIOA1_BASE,GPIO_PIN_5);
			SetupA1_LowInt(GPIO_INT_PIN_5);
			MAP_GPIOPinWrite(GPIOA1_BASE,GPIO_PIN_2,GPIO_PIN_2);
			TcpSend("2\n","ON\n");
			break;
	}}
```

Watchdog timer property configuration and interrupt handling

The watchdog timer is a safety feature, which resets the processor if the program becomes stuck in infinite loop. Its main function is to protect the system against malfunctions

```
#define WD_PERIOD_MS                 5000
#define MAP_SysCtlClockGet           80000000 //80MHz i.e frequency of ARM Cortex
#define WD_MILLISECONDS_TO_TICKS(ms)    ((MAP_SysCtlClockGet / 1000) * (ms))
```

The routine that feed to watchdog timer is defined as

```
static void WatchdogAck()
{
  // Acknowledge watchdog by clearing interrupt
  //
       MAP_WatchdogIntClear(WDT_BASE);
}

//Watchdog timer interrupt Handler
void WatchdogIntHandler(void)
{
// Exit from handler to allow immediate system reset
   return;
}
```

Access Point property configuration

Every access point has certain properties. These include SSID, BSSID, security key, security type *et cetera.* In the ATOMS Board embedded C source code, values for these are supplied. With these properties given, the ATOMS device would be able to locate the access point and connect successfully to it when the application is executed

```
#define SSID_NAME           "Ing  X"    // AP SSID
#define SECURITY_TYPE       SL_SEC_TYPE_WPA// Security type (OPEN or WEP or WPA)
#define SECURITY_KEY        "silverblade"        // Password of the secured AP
#define SSID_LEN_MAX        32 //maximum SSID length
#define BSSID_LEN_MAX       6 //maximum BSSID length
```

Client-Server Relationship

With Client-Server relationship, the client requests that some action be performed, and the server performs the action and responds to the client. In the ATOMS setting, there is a two way communication between the microcontroller board and the base station. Thus to establish this the socket-based communication over the network, a TCP client and server are setup on the ATOMS device. The popular Transmission Control Protocol (*which uses stream sockets*) was chosen for use as opposed to the Use Datagram Protocol, UDP (*which uses datagram sockets*) because unlike UDP, TCP provides a connection-oriented service. That is, with stream sockets, a device establishes a connection with another. While the connection is in place, data flows through the devices in continuous streams. It also guarantees that packets arrive in the order in which they are sent thus it's reliable and require relatively less error-checking.

A suitable port is chosen whose number is defined as *SERVER_PORT*. Although port numbers can range as wide as between 0 and 65535, a large port number is chosen. This is because some operating systems reserve port numbers below 1024 for system services. The port number (PORT_NUM) of the client program on the base station is then defined for use in connection. The size of the buffer (*BUF_SIZE*) which temporarily hold yet to be transmitted data or just received data, is defined too.

```
#define PORT_NUM            12000
#define SERVER_PORT              18000
#define BUF_SIZE            12
```

The server setup function, *TcpServer*, opens a TCP server side socket in listen mode and listens indefinitely (or *blocks*) for an attempt by a client to connect. If a socket connection is established then the function will try to read TCP packets from the connected client. The function will wait for an incoming connection till one is established in a process called *binding the server to the port*. The client requests to connect to the port. After receiving from connected TCP client, the connected socket is closed. Its definition is as follows:

```
//****************************************************************************
//! \param[in] port number on which the server will be listening on
//
//****************************************************************************
void TcpServer(unsigned short usPort)
{
    SlSockAddrIn_t  sAddr;
    SlSockAddrIn_t  sLocalAddr;
    int             iCounter;
    int             iAddrSize;
    int             iSockID;
    int             iStatus;
    int             iNewSockID;
    int             iTestBufLen;

    // filling the buffer
    for ( iCounter=0 ; iCounter<BUF_SIZE ; iCounter++)
    {
        G_payload[iCounter] = (char)(iCounter % 10);
    }

    iTestBufLen  = BUF_SIZE;

    //filling the TCP server socket address
    sLocalAddr.sin_family = SL_AF_INET;
    sLocalAddr.sin_port = sl_Htons((unsigned short)usPort);
    sLocalAddr.sin_addr.s_addr = 0;

    // creating a TCP socket
    iSockID = sl_Socket(SL_AF_INET,SL_SOCK_STREAM, 0);

    iAddrSize = sizeof(SlSockAddrIn_t);

    // binding the TCP socket to the TCP server address
    iStatus = sl_Bind(iSockID, (SlSockAddr_t *)&sLocalAddr, iAddrSize);

    // putting the socket for listening to the incoming TCP connection
    iStatus = sl_Listen(iSockID, 0);
```

```
    // setting socket option to make the socket as non blocking
    //Continually listen for connections on active server port
    while(1)
    {
       iNewSockID = SL_EAGAIN;

       // waiting for an incoming TCP connection
       while( iNewSockID < 0 )
              {
                     // accepts a connection from a TCP client, if there is any
                     // otherwise returns SL_EAGAIN
                     iNewSockID = sl_Accept(iSockID, ( struct SlSockAddr_t
*)&sAddr,(SlSocklen_t*)&iAddrSize);
              }

              iStatus = sl_Recv(iNewSockID, G_payload, iTestBufLen, 0);


       // close the connected socket after receiving from connected TCP client
              iStatus = sl_Close(iNewSockID);

              Report(G_payload);
```

The *TcpClient()* when called opens a TCP client side socket and attempts to connect to the base station server program waiting on port *PORT_NUM*. using the IP address, *G_destinationIp*, of the base station given it by the AP.

```
void TcpClient()
{
       unsigned short usPort = PORT_NUM;    //filling the TCP server socket
address
    G_sAddr.sin_family = SL_AF_INET;
    G_sAddr.sin_port = sl_Htons((unsigned short)usPort);
    G_sAddr.sin_addr.s_addr = sl_Htonl((unsigned int)G_destinationIp);
    G_iAddrSize = sizeof(SlSockAddrIn_t);

}
```

AP connection establishment

In the establishment of a connection to the wireless LAN access point, the *WLanConnect()* plays the main role. The function connects to the required AP (*SSID_NAME*) with the security parameters specified in the form of macros at the top of the *main.c* file.

```
static long WlanConnect()
{    SlSecParams_t secParams = {0};
    long lRetVal = 0;
```

```
    secParams.Key = (signed char*)SECURITY_KEY;
    secParams.KeyLen = strlen(SECURITY_KEY);
    secParams.Type = SECURITY_TYPE;

    lRetVal = sl_WlanConnect((signed char*)SSID_NAME, strlen(SSID_NAME), 0,
&secParams, 0);
    ASSERT_ON_ERROR(lRetVal);


    /* Wait till device is connected to the access point */
    while((!IS_CONNECTED(g_ulStatus)) || (!IS_IP_ACQUIRED(g_ulStatus)))
    {
      // Wait for WLAN Event
#ifndef SL_PLATFORM_MULTI_THREADED
        _SlNonOsMainLoopTask();
#endif

        //Blink orange led to show connection to AP
        MAP_GPIOPinWrite(GPIOA1_BASE,GPIO_PIN_2,GPIO_PIN_2);
             MAP_UtilsDelay(1000000);
             MAP_GPIOPinWrite(GPIOA1_BASE,GPIO_PIN_2,0);
             MAP_UtilsDelay(1000000);
    }

    //turn of red orange after successful connection to AP
    MAP_GPIOPinWrite(GPIOA1_BASE,GPIO_PIN_2,0);
    return SUCCESS;
}
```

TCP packet anatomy

The payload field of the TCP packet is 16 bytes long. The length of the payload is divided into two major parts. The first 8 byte contains information concerning the board unit ID. The part tells of the occupancy information.

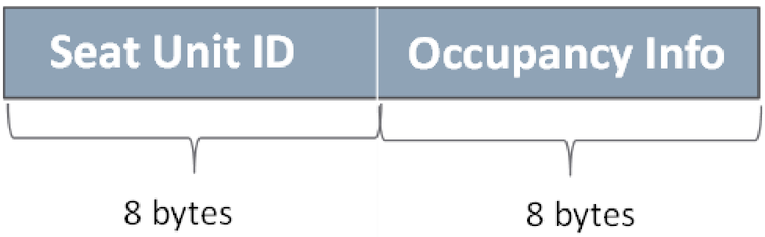


Figure 34: Packet structure.

Sending TCP packets to Base Station Application.

The `TcpSend()` function is used to send the constructed packets from the ATOMS device client to the TCP server application on the Base Station. A call is made to the interrupt handlers which are responsible for capturing interrupts whenever they are triggered.

```
void TcpSend(char seat_no[8],char message[8])
{
	int		iSockID;
	int		iStatus;
	short		sTestBufLen;
	char			payload[24];
	//unsigned int	counter;

	sTestBufLen  = 24;

	// fill payload with data in specific order
	strcat(payload,UNIT_NO);
	strcat(payload,seat_no);
	strcat(payload,message);

	// creating a TCP socket
	iSockID = sl_Socket(SL_AF_INET,SL_SOCK_STREAM, 0);


	// connecting to TCP server
	iStatus = sl_Connect(iSockID, ( SlSockAddr_t *)&G_sAddr, G_iAddrSize);

	// send packet to the TCP server
	iStatus = sl_Send(iSockID, payload, sTestBufLen, 0 );

	Report("Sent %s packets successfully\n\r",payload);
	//memset(G_payload,'\0',12);

	//closing the socket after sending packet
	iStatus = sl_Close(iSockID);

	MAP_GPIOPinWrite(GPIOA1_BASE,GPIO_PIN_2,0);}
```

Receiving TCP packets from Base Station

TCP packets sent from the Base Station client is received by the ATOMS Board server. Refer to the `TcpServer()` function

Temporary vacation of seats

This is a feature that allows seated clients to briefly evacuate their seats without them been viewed as vacant and subsequently taken by newly come patrons. The former might want to visit the washroom, grab some snack or do something elsewhere. Generically when an occupant vacates a seat, an interrupt is triggered and a packet indicating vacancy is transmitted. However when the temporary vacation button is pressed before evacuation, a timer is started and the transmission is delayed. This vacation is timed for a predetermined amount of time after which, the packet is sent and the seat is declared available for taking.

Generation of interrupt is shown in code below

```
void A2_IntHandler()
{
	switch(GPIOIntStatus(GPIOA2_BASE,true))
	{
		case GPIO_INT_PIN_6:
			MAP_GPIOIntClear(GPIOA2_BASE,GPIO_PIN_6);
			G_vacated = 1;
			Timer_IF_Start(g_ulBase, TIMER_A, 10000); //time for 10 secs
			break;
	}}
```

After the time elapses, a “vacancy” TCP packet is then assembled and then sent to the base station. The code below illustrates this.

```
void Temp_Vacate_IntHandler(void)
{

    // Clear the timer interrupt.
    Timer_IF_InterruptClear(g_ulBase);

    G_vacated = 0;
    g_ulTimerInts ++;
    MAP_GPIOPinWrite(GPIOA1_BASE,GPIO_PIN_2,GPIO_PIN_2);
       MAP_UtilsDelay(1000000);
       MAP_GPIOPinWrite(GPIOA1_BASE,GPIO_PIN_2,0);

       G_vacateFlag = 1;
}
```

Hibernation of seat units

All ATOMS devices affixed to seats in the seating area can be remotely hibernated by issuing a command from the base station. When the ‘hibernate’ button on the base station application interface is pressed, a TCP packet with a payload `‘sleep’` is assembled and multicast over the wireless LAN. This is received by the server on each board and then processed. The boards are consequently switched to hibernate power mode. In this mode, almost shutting down all processes running on the device. The state of these processes are however saved for recall. This is used to deep sleep after a working day of an enterprise. It saves device power.

However, the ATOMS devices cannot be ‘woken up’ remotely in a similar way as the hibernation is done. This is because in the hibernate mode, the network processor is put in a state where communication with base station is not possible as there is less power for reception and processing of sent packets. Nonetheless a device can restored, when the sensor picks up a stimulus.

The server side check for a *'sleep'* payload and a subsequent call to the 'hibernate' function is shown below:

```
if(!strcmp(G_payload,"sleep"))
             {
                    HIBEntrePreamble();
                    MAP_PRCMOCRRegisterWrite(0,1);
                    MAP_PRCMHibernateWakeupSourceEnable(PRCM_HIB_SLOW_CLK_CTR);
                    MAP_PRCMHibernateIntervalSet(330);
                    MAP_PRCMHibernateEnter();//hibernate function
             }
```

Main Function

The main running program is shown below. In this function, the board is initialized by a call to the *BoardInit()*. The *PinMuxConfig()* function configures the pin multiplexing settings for the peripherals exercised. With the device assumed to be configured in station mode (its default state) already, the declaration

```
lRetVal = sl_Start(0, 0, 0);
```

is made to start device. Thereafter, attempts to establish connection with the wireless LAN AP is initiated. A call to the *TCPClient()* function initiates TCP connection global variables. The interrupts on ports A1 and A2 are activated here. To show the application is still running, the green LED is turned on. The ATOMS device's TCP server is then started after which the green LED is turned off. This *main()* function will run for the rest of the application till an interrupt is received, after which it will resume running.

```
void main()
{
    long lRetVal = -1;

    // Board Initialization
    BoardInit();


    // Configure the pinmux settings for the peripherals exercised
    PinMuxConfig();


    // Configuring UART
    InitTerm();


      // Set up the watchdog interrupt handler.
      WDT_IF_Init(WatchdogIntHandler,WD_MILLISECONDS_TO_TICKS(WD_PERIOD_MS));
```

```
      // Display banner
    DisplayBanner(APPLICATION_NAME);


    // Initialize the application variables
    InitializeAppVariables();


    // This starts the simplelink device
    lRetVal = sl_Start(0, 0, 0);
    if (lRetVal < 0)
    {
        LOOP_FOREVER();
    }

    // Connecting to WLAN AP - Set with static parameters defined at common.h
    // After this call we will be connected and have IP address
    lRetVal = -1;

    while(lRetVal < 0)
    {
      lRetVal = WlanConnect();

    }

    //Initiate TCP connection global variables
    TcpClient();


    // Activate interrupts on ports A1 an A2 respectively
    SetupA1_Int(0);
    SetupA2_Int(0);


      // Base address for timer
      g_ulBase = TIMERA0_BASE;


      // Configuring the timer
      Timer_IF_Init(PRCM_TIMERA0, g_ulBase, TIMER_CFG_ONE_SHOT_UP, TIMER_A, 0);


      // Setup the interrupt for the timer timeout.
      Timer_IF_IntSetup(g_ulBase, TIMER_A, Temp_Vacate_IntHandler);


    //Turn on green led to show application is still running
      MAP_GPIOPinWrite(GPIOA1_BASE,GPIO_PIN_3,GPIO_PIN_3);

      // Main system loop -- Note actual server loop in function TCPServer
    while(1)
    {
      WatchdogAck();
      TcpServer(SERVER_PORT);
    }
}
```

3.5.2 Base Station Application Software

The base station application software is responsible for correspondence with all individual ATOMS devices affixed to seats in the auditorium. This application thus controls the reception and subsequent processing of data packets sent from the seat units. The application is written as a standalone desktop application and installed on the base station. It is so written such as to liaise easily with the wireless interfaces of these computers on which it runs. Computers running on the windows operating system were considered for use as the stations in this this study The Windows versions that are compatible with the software includes all versions from Windows XP to the latest Windows 10. The application runs on a local server thus can operate autonomously without internet connection.

The software was written using Python, JavaScript, HTML5, and CSS3. Integrated development environments used were PyCharm version 4.0.6 and Bracket Sprint version 1.3.

Use Case Diagram

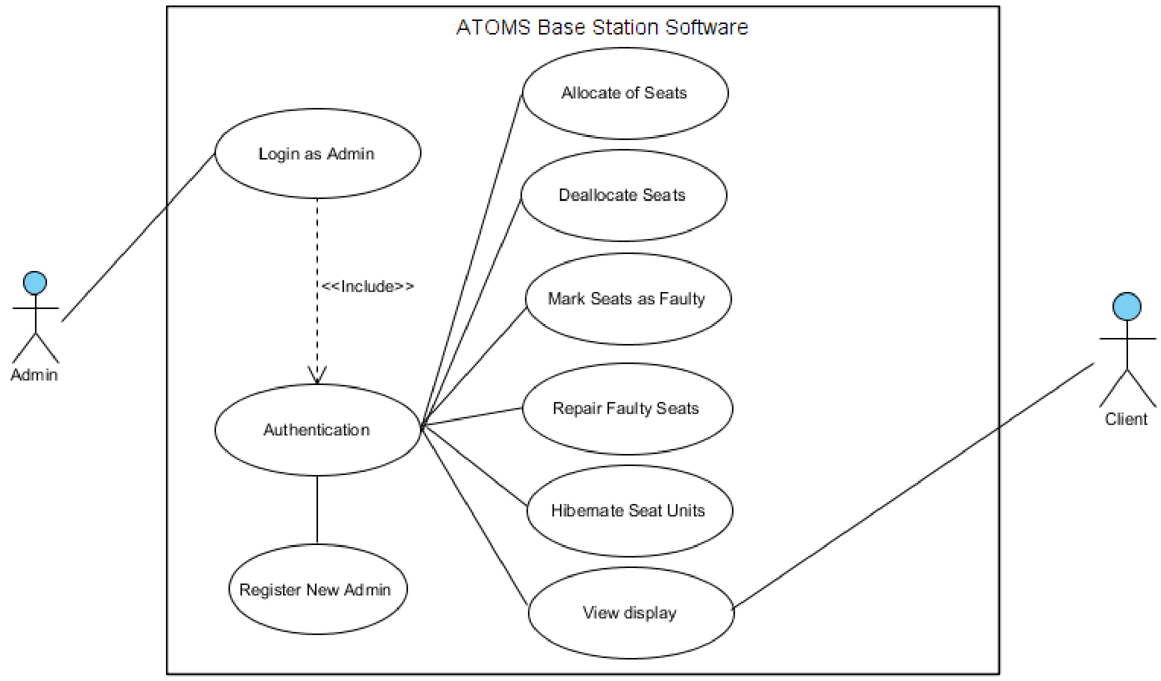


Figure 35: Use Case Diagram for the base station application software.

Figure 35 shows the use case diagram for the base station application software. The diagram gives a visual representation of the relationships between the actors (internal or external entities that interact with the system) and the use cases that documents the system's intended behavior. The ATOMS system has two actors: an administrator charged with the task of controlling the system from the base station and a client who patronizes the services of the auditorium. From the diagram in the above figure, an administrator has the ability to reserve seats in the seating area from the application software. He can also de-allocate reserved seats. When any seat in the seating area is damaged, that seat can be tagged as faulty so as to duly inform clients of state of the seat. After such chairs are repaired, the administrator the removes can then fault tag associated with a seat. The administrator can remotely hibernate all individual or specifically named ATOMS devices affixed to the seats.

Log in and authentication

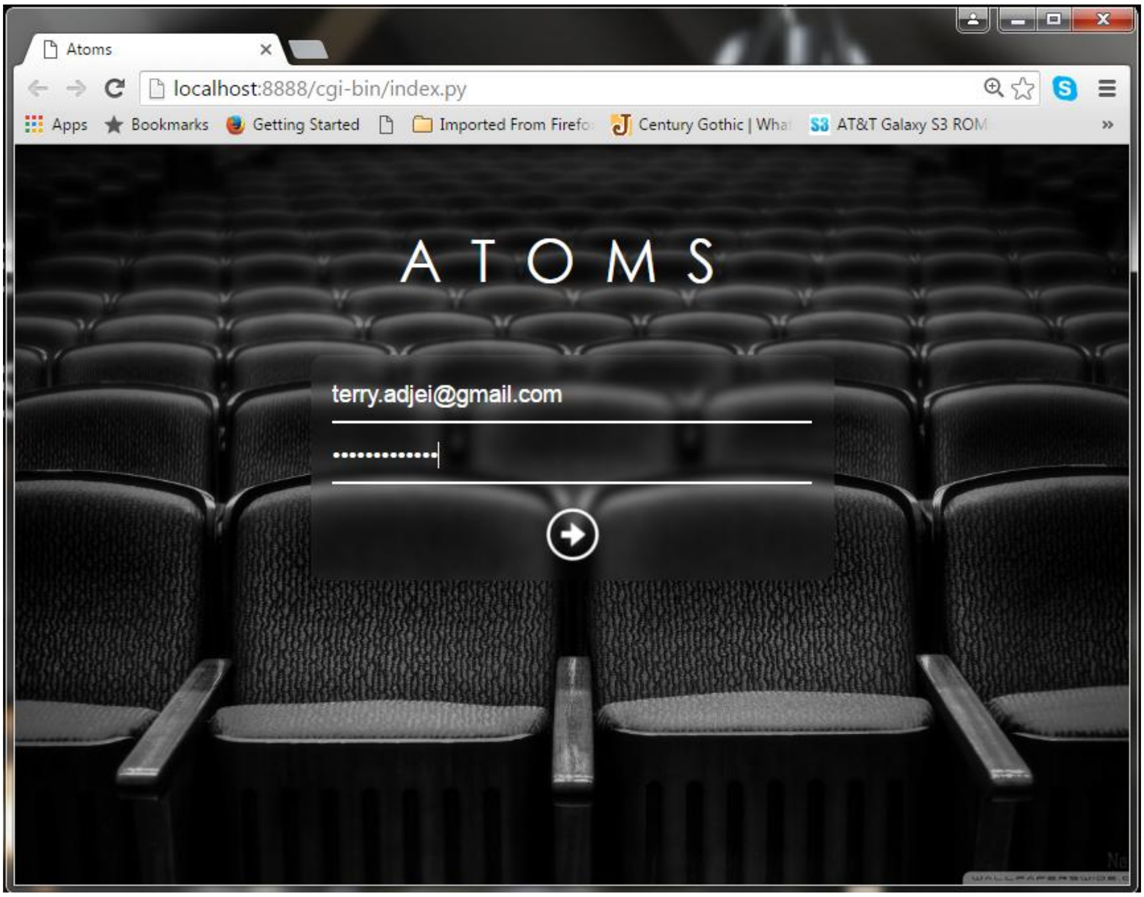


Figure 36: A view of the login page of the ATOMS software.

The application, when started, opens up in the default browser or any other selected browser on the base station. The browser pops up the landing page of the ATOMS application. This page doubles as the log in portal page of the application thus a user is required to log in with his credentials in order to be granted access to his account and use the application. An administrator's account allows the him/her to be authenticated by the application and to receive authorization to access resources provided by the system; however, authentication does not infer authorization. To sign in to an account, a user is required to authenticate oneself with registered email and matching password for the purposes of accounting, security, logging, and resource management. Once the administrator has logged on with the correct credentials, the ATOMS system uses an identifier, rather than the username, through a process known as *identity correlation*. He is then given access into the system and is directed to the main page which trespassers cannot see.

The ATOMS application software is a multi-user system because it makes provision for multiple users to identify themselves before using the system. Each administrator's (user's) account on a multi-user system typically has a *home directory*, in which all files relating solely to that user's undertakings are stored. Access to these files is denied from other administrators. These accounts contain user profile, which contains elementary information given by the account bearer. The files stored in the home directory have file system approvals which are inspected by the application to determine which users are granted access to read or execute a file, or to store a new file in that directory.

Main application page

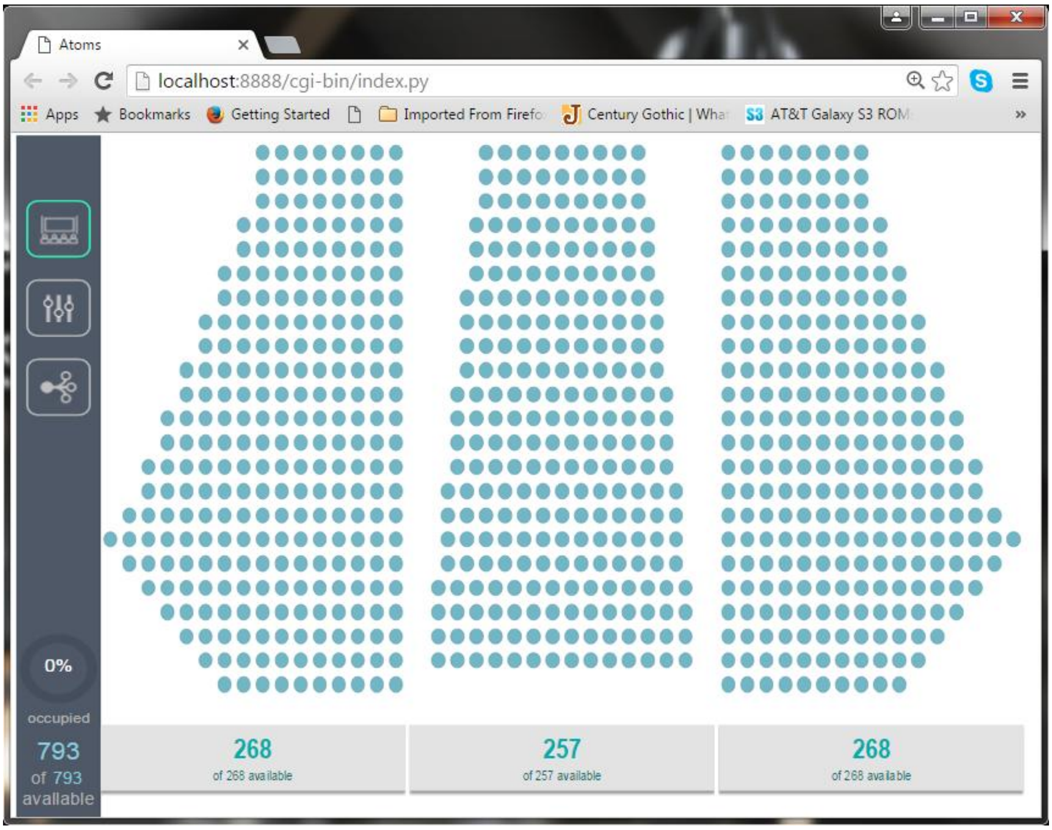


Figure 37: A view of the main page of the ATOMS Base station application on startup.

The figure above highlights the startup view of the main page of the base station application. There is a left bar housing three buttons involved in the major actor-system interactions. These buttons are “*Hall view*”, ‘*Options*’ and ‘*Allocate*’. The lower section of the bar gives numerical information concerning the total available seats in the hall. This section additionally expresses the number of available seats as a percentage of the number of available seats. The section of the page right of the button bar shows a corresponding representation of seats arrangement in the auditorium. Below each column of seats, the number of available seats for that column is provided.

Seats representation and colour coding

In the application interface as shown, the chairs in the seating area are represented by circles. The colour a seat bears at any instant tells the administrator or patron of its occupancy status. The colour scheme employs three colours: light cadet blue, reddish orange and black. The colours, how they are presented on the application interface and their description are provided in the figure below.

	Colour code	
Symbol	**Colour**	**Description**
	Light cadet blue	Indicates that a seat is vacant
	Red	Indicates that a seat is occupied, reserved or temporarily vacated
	Black	Indicates that a seat is faulty and requires immediate repair.

Figure 38: Colour coding scheme.

Functional Buttons

“Hall view” button

This is the normally active button in the left aligned button bar; that is it the button whose functionality is automatically activated when an administrator logs in. When the *“Hall view”* button is pressed, all extended panels are closed and the view reverts to the startup state where the administrator can have an overview of the hall.

‘Options’ button

This is the second button in the button bar. The *'Options'* button when pressed, its border colour changed from gray to turquoise and then a pane extends right of the button bar. This *'Options'* pane contains a log out button and faulty seat, seat hibernation and administrator registration sub panes.

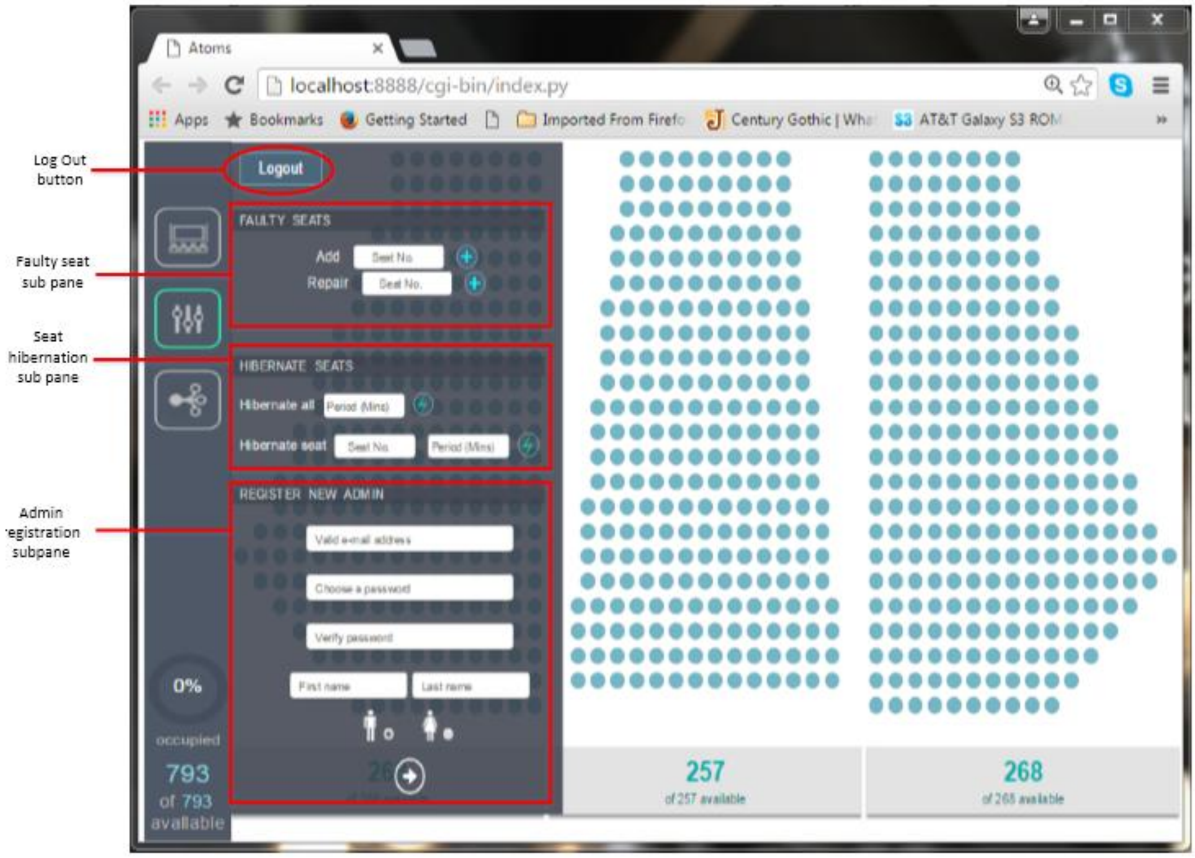


Figure 39: A view of the 'Options' pane.

“Allocation” button

The allocation button is used when the administrator intends to reserve seats for some clients. An allocation pane containing allocation controls is slid to the right of the button bar when the button is pressed. The pane also gives a user the opportunity to de-allocate reserved seats. The allocation pane also makes available to the administrator, a list of vacant se seats in the auditorium. Additionally a list of allocated seats.

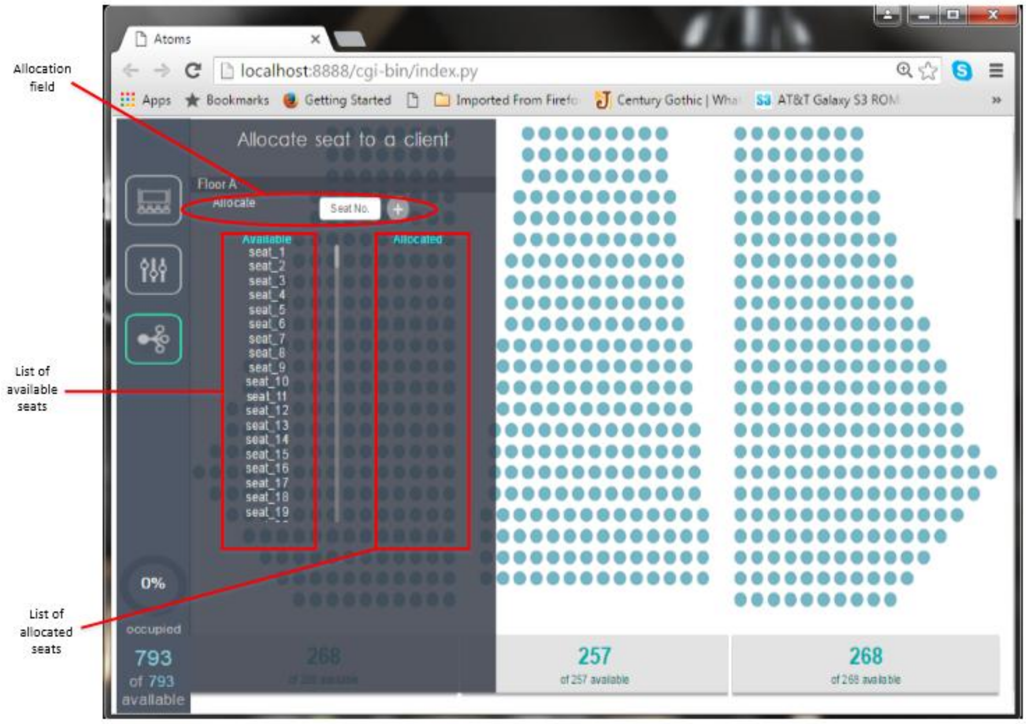


Figure 40: A view of the allocation pane.

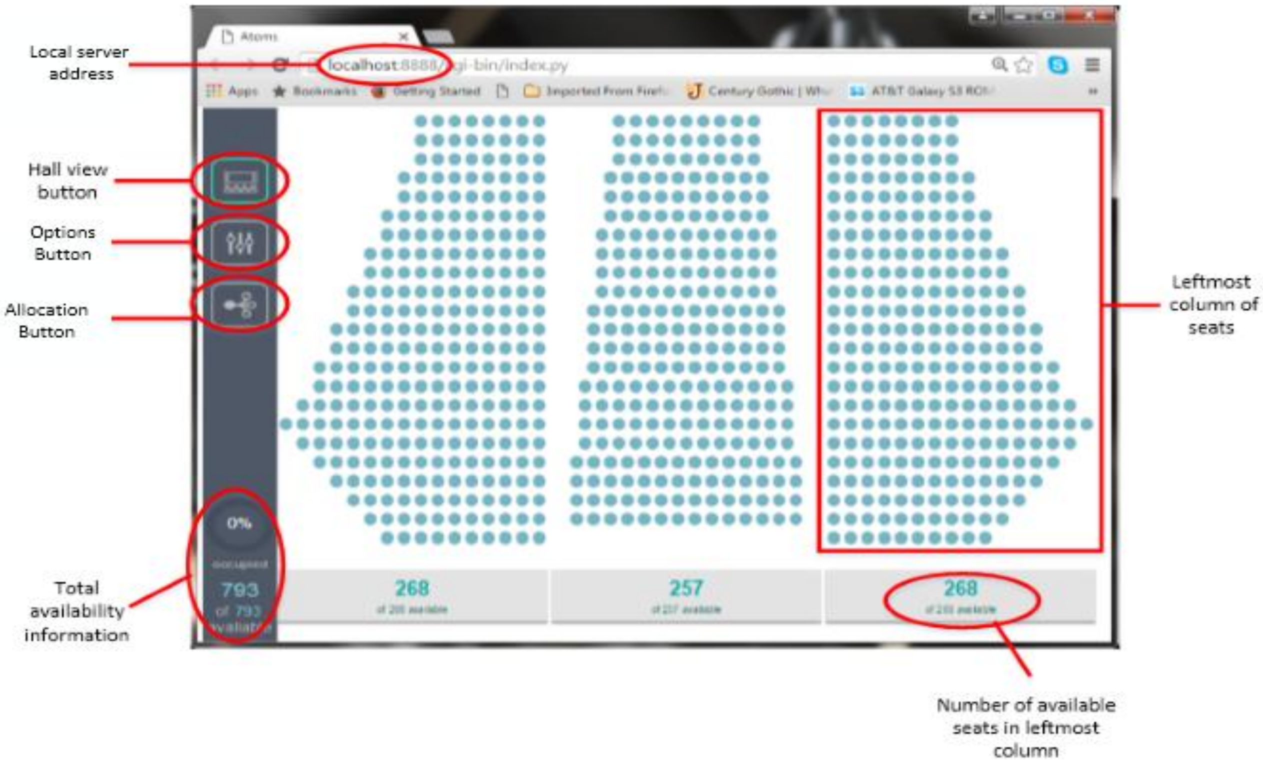


Figure 41: Some major parts of the application interface.

Tagging faulty seats

When seats in the auditorium break down, it is imperative that the administrator tags these seats as faulty. This is to aid the administrator in undertaking actions such as reservation of seats as it is improper that he allocates a faulty seat to a client. It also helps newly come clients to also know seats that are not in good shape for use and thus avoid them accordingly. This saves them a lot of time and energy. Tagging of faulty seats is done using the *"faulty seats"* controls found on the *'Options'* pane. The sub pane has a field tagged *"Add"* which is meant to be filled with the seat number of the faulty seat. After a seat number is provided, the 'plus' labelled button on the right hand side of the field is pressed to tag the faulty seat. When tagged as faulty, the colour of the circle representing that seat on the application interface changes from light cadet blue to black. Instances of tagged faulty seats can be found on Figure 46.

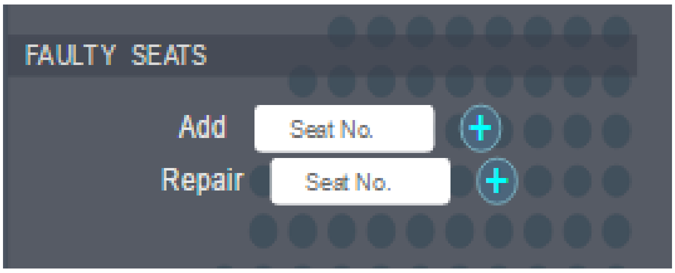


Figure 42: "Faulty seats" controls.

Removing "Faulty seat" tag

"Faulty seat" tags are removed from associated seats when such seats get fixed. Here, the sat number is supplied in the *"Repair"* field in the *"faulty seats"* sub pane and then the 'plus' tagged button is pressed (Refer to Figure 42). The colour of the seat is reverted from black to its original light cadet blue colour. It should however be noted that until the seat is fixed and the fault tag is subsequently removed, any interaction of occupants with the seat units generate no change on the application's interface.

Hibernating seat units

The hibernation feature of the ATOMS system is basically geared towards power conservation. This functionality enables administrators of the system to remotely put in a dormant state, all or particular ATOMS devices affixed to seats in the seating area. Hence, the devices are stopped from undertaking any regular activities unless they are informed to resume active work. However, in the hibernate state, the state of the devices' processes prior to entering the hibernate state are stored for continuation after 'wake-up' (Refer to Figure 19).

To hibernate all seat units in the hall, the *"Hibernate all"* control as shown in the figure below if used. The field next to the label is filled with the period of time intended for the hibernation in minutes and then the button with turquoise thunderbolt label located right of the field is pressed. UDP packets containing this command are assembled, encrypted and the multicast to all seat units in the seating area. All ATOMS devices on the network upon receiving these packets, decode them, extract the information and act on the command by entering the lowest power mode of the devices. However to hibernate a particular seat, the *"Hibernate seat"* control is used where the seat number and the hibernation period (also in minutes) are specified and the "thunderbolt button" is pressed. Here a TCP packet is assembled, encrypted and sent over the WLAN to the specified seat. The seat receives the packet decrypt it, extracts the information and acts on the hibernation command.

The hibernated seat(s) is (are) woken up when an occupant sit on the seat or when the specified sleep time elapses.

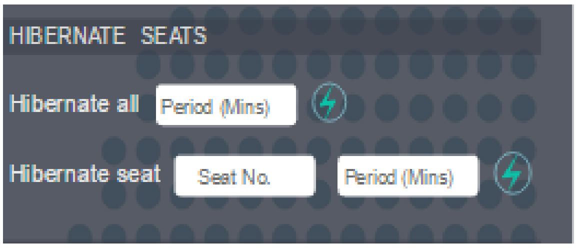


Figure 43: "Hibernate seats" controls.

Registering a new Administrator

New administrators can be registered and given accounts with credentials which they can use to also access the system and operate the system. The registration is the sole prerogative of existing registered administrators. This is done on the *"New administrator registration" sub pane.* The registration sub pane has five mandatory fields and a gender checkbox. These fields take the new administrator's email address, password, retyped password for checking correctness; his or her first name and the last name respectively. After all the fields are filled and the appropriate gender ticked, the forward pointing arrow button is clicked to save the credentials. These credentials and personal information is hashed, stored in a script file and stored in a created home directory linked to the newly created account.

Figure 44: "New administrator registration" sub pane.

Allocating seats

In allocating seats, the system overseer has two options. The first method is by inserting the seat number into the field next to the '*Allocate'* label and then pressing the 'plus' labelled button right of the allocation field. An alternative is scrolling through the available seat on the left of the scroll bar shown in the figure below, clicking on the identifier of the seat intended to be allocated and the pressing the "plus button". When any of the above is done, the seat identifier is moved to the allocated column and the colour of seat on the application interface is changed from light cadet blue to reddish orange indicating that the seat is taken to the knowledge of newly come clients.

De-allocating allocated seats

When clients for whom seats in an auditorium were allocated arrive, it is expedient that the seats are de-allocated to allow the system recognize changes in occupancy status thereafter. This is also done on the "*Allocation*" sub panel. In order to achieve this, the allocated seat that an administrator wishes to de- allocated is identified in the *'Allocated'* column left to the scroll bar (shown in figure below). The button to the left of the seat identifier, bearing a dark orange minus sign, is clicked. The seat is then moved from *'Allocated'* column to the '*Available*' column and the colour of the seat on the floor plan of the hall on the software interface is changed from reddish orange to light cadet blue.

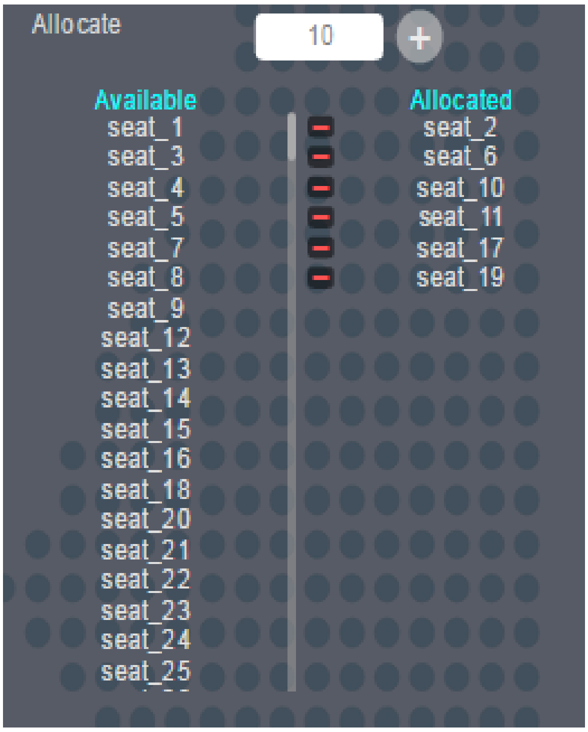


Figure 45: 'Allocation' sub panel.

Log out

When an administrator completes a system control session, he/she can log out of the system using the *"Log Out"* button found on the *'Options'* panel.

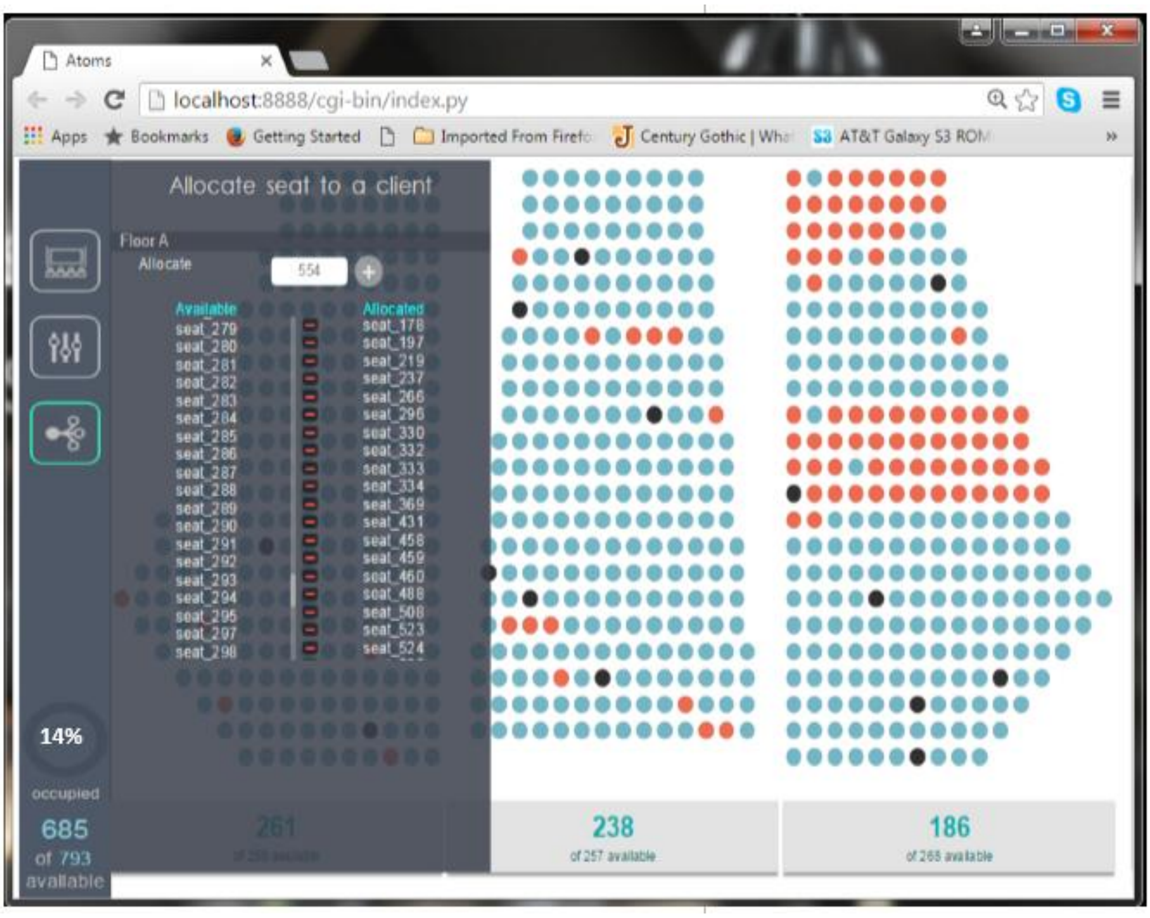


Figure 46: A view of the application interface showing some seats with their status.

3.6 ATOMS device packaging

The proposed ATOMS device package is shown in the figure below. The figure highlights the front, right, rear and top views of this casing. The front of the casing displays boldly "ATOMS" and houses the power button in its upper center half. There reset button is found on the lower right corner of the front of the device casing. The three indicators of the ATOMS device are also found on the front side. The right edge view of the device shows three female ports which receive the stimulus and response pins from the capacitive proximity sensor; and "temporary vacate" pin. The rear view shows the CR2032 battery holder. The edges of the rear side has removable pad beneath which screws can be found.

The package is arrayed in navy blue, deep sky blue, and light gray; which are the theme colours of the ATOMS system.

Figure 47: Device packaging.

Figure 48: Dismantled ATOMS device.

Figure 49: Powering the ATOMS device.

4. TEST AND EVALUATION

4.1 Test

A mini-seating area was set up and the system was set up. The test was carried out for three hours interspersed with random occupying, “*permanent*” and “*temporary*” vacation; allocation, de-allocation, singular and mass hibernation, fault tagging and “*repairing*” of seats. Several arbitrary log in and log out instances by various administrators; and registering of new administrators also took place. Thus all aspects of the system was tested.

4.2 Evaluation

From the theoretical and experimental results, we have estimated that this loading mode capacitive sensing technique can clearly distinguish between an occupied and available seat. The relationship between available and occupied is approximately a double ratio one in favour of the latter. After the three hours of testing, the voltage of the battery was measured and recorded as 2.99V. It was estimated that a battery could last a year when used to power device and thus the device is power efficient. Base station – display interaction was a hundred percent proficient, however the base station and seat unit communication experienced four percent packet failure. This was attributed to factors such as interference from other networks, interference from other wireless-technology-based networks such as Bluetooth, ZigBee and UWB. This could also be alluded to radio range issues.

5. CONCLUSIONS AND RECOMMENDATIONS

5.1 Conclusion

We have proposed and tested a system for dynamically monitoring seat occupancy in auditoriums, halls and the like. The ATOMS system can handle this task efficiently, accurately and cost effectively. Wireless communication adapted into this system to replace the massive cabling make easy seat occupancy monitoring neat and easy to achieve. The designed system is highly practical and adaptable for installation in any auditorium setting. The system is very portable to cater for already built seats. The system also enables hall establishments operate independent of ushers.

5.2 Recommendations

1. Base stations should be diversified. Portable devices such as mobile phones, and tablets should be considered.
2. The CC3200 microchip allows for checking of power supply level. Future versions of the system should be able to remotely inform the administrator of devices with low battery sources.
3. The base station- ATOMS device communication should be made impregnable to interference from other networks or other wireless technologies.
4. The system could be prospectively applied as a ticketing concierge.

6. REFERENCES AND APPENDIX

6.1 References

1. Gaugler and Gaugler, 1941 "*Chair for motion picture theatres*".
2. Blinkilde and Sandi, 1975 "*Occupant sensor seat switch*".
3. Helbling, 1996 "*Charitable contribution centralized system and apparatus*".
4. Helbling and Glass, 1996 "*Automatic theatre ticket concierge*".
5. Gabriel, Gabriely and Corthout, 1996 "*The dielectric properties of biological tissues".*
6. Zeidler, Mickeler, Petri and Meyer, 1997 "*Device for detecting seat occupancy in a motor vehicle*".
7. Barr, 1999 "*Programming embedded systems in C and C++*".
8. Baron and Abbott, 2000 "*Seat occupancy indicator and seating area display system*".
9. Pont, 2002 "*Embedded C*"
10. University of Texas, 2002 "*Embedded C programming*".
11. Briere and Hurley, 2005 *"Wireless hacks and mods for dummies".*
12. VonDoenhoff and Shetty, 2008 "*Automatic airplane seat location mapping*".
13. Schroeder, Sadasiva and Nelson, 2008 "*An Analysis on the role of water content and state on effective permittivity using mixing formulas*
14. Wimmer, 2011 *"Capacitive sensors for whole body interaction".*
15. Grunfeld, 2011 "*seat occupancy detection and display system*".
16. Hefferon, 2011 "*Dynamic occupancy monitoring*".
17. Atmel Corporation, 2012 "*Proximity design guide*".
18. Hefferon, 2013 "*Dynamic occupancy monitoring*".
19. Zeeman, Booysen, Ruggeri and Lagan´a, 2013 *"Capacitive seat sensors for multiple occupancy detection using a low-cost setup".*
20. Stewart "*Wireless LAN Profile".*
21. Texas Instruments, 2014 "*Primer to Wi-Fi® provisioning for IoT applications*".
22. Texas Instruments, 2014 "*A link to internet of things*".
23. Texas Instrument, 2015 *"CC3200 programmers' guide".*
24. Texas Instrument, 2015 *"CC3200 datasheet".*
25. Texas Instrument, 2015 *"CC3200 user guide".*
26. Siemens, 2015 "*Sensor Basics 3"*.
27. Aguilar, Roelofsz and Meijer *"Capacitive human-detection systems with auto-calibration".*
28. Albesa and Gasulla "*Seat occupancy and belt detection in removable seats via inductive coupling*".

6.2 Appendix

6.2.1 ATOMS Board Technical Document

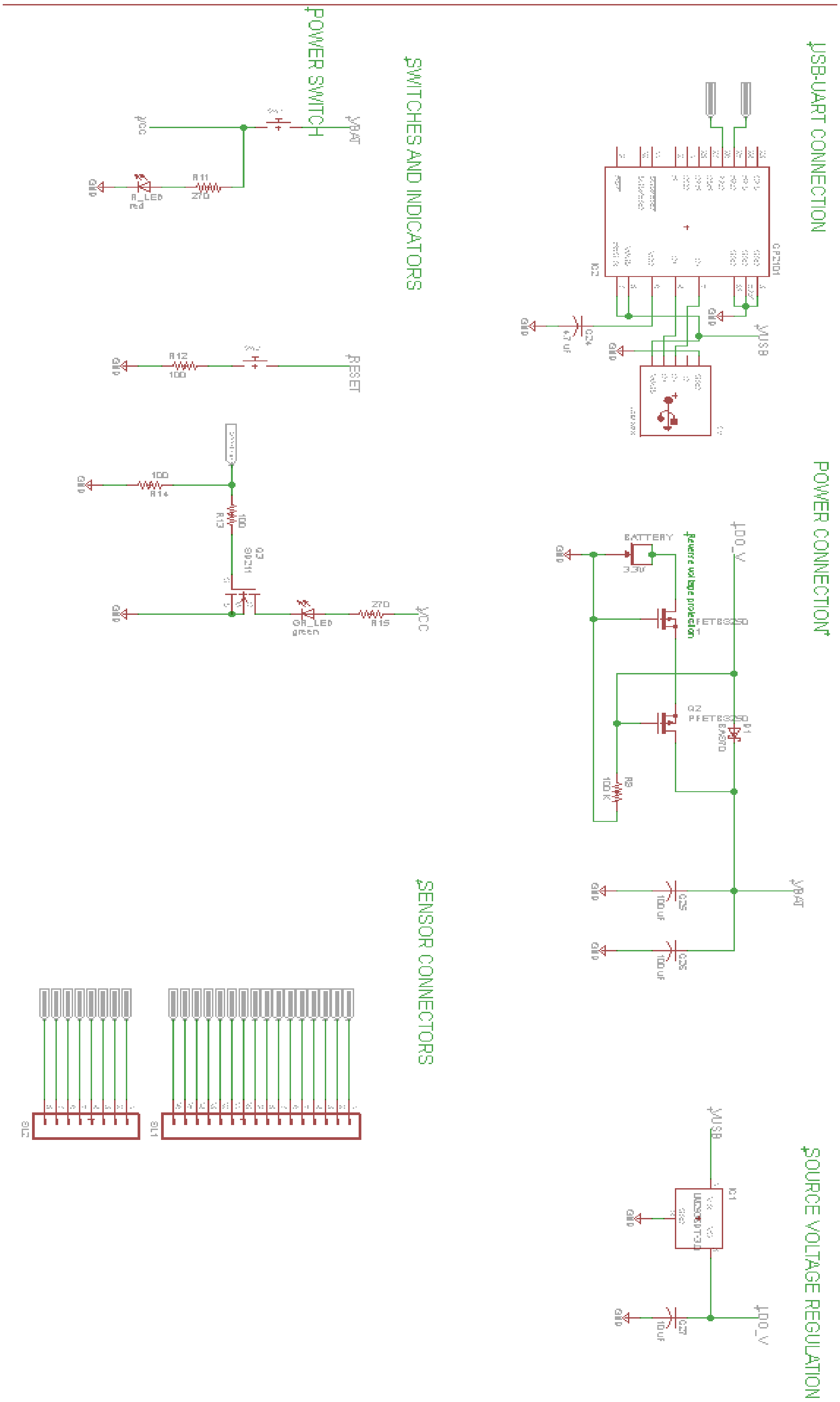


Figure 50: Schematic of the ATOMS project board.

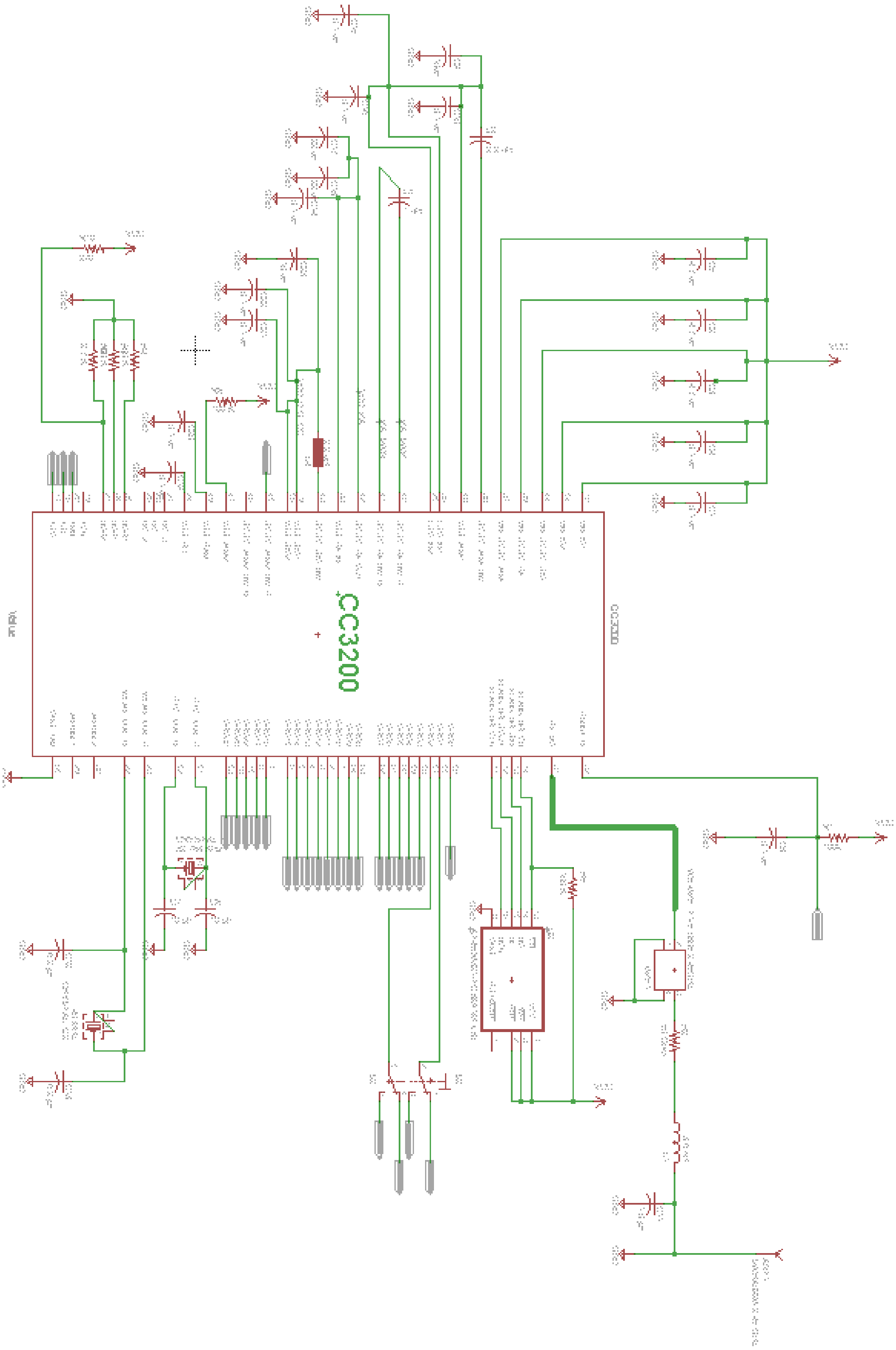


Figure 51: Schematic of the ATOMS project board.

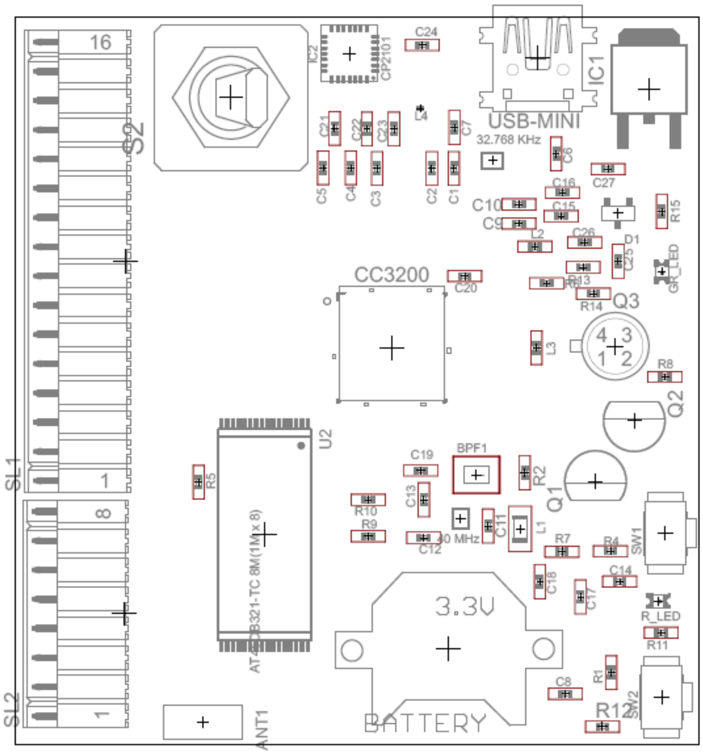


Figure 52: The ATOMS project board without pads.

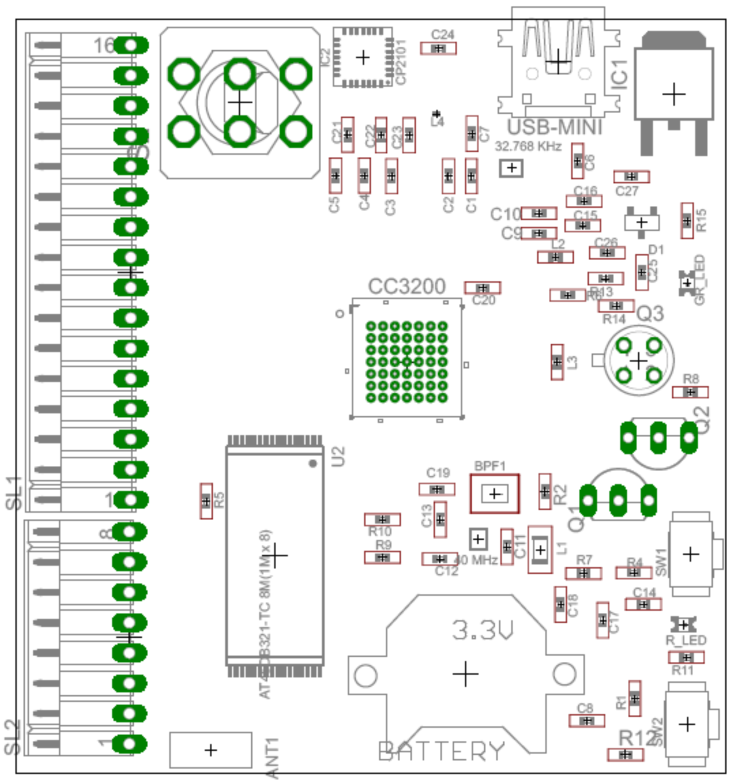


Figure 53: The ATOMS project board with pads.

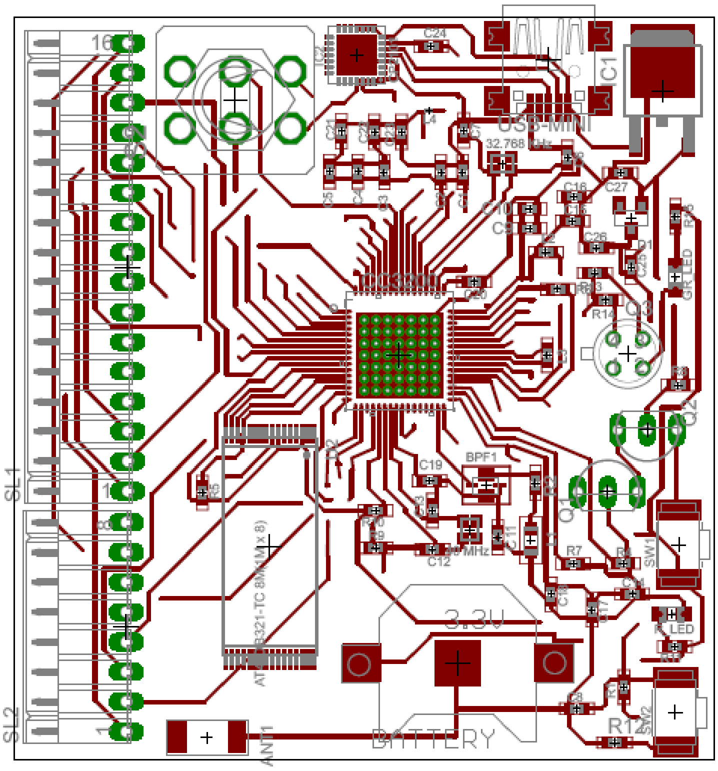


Figure 54: Routed top layer of the ATOMS project board.

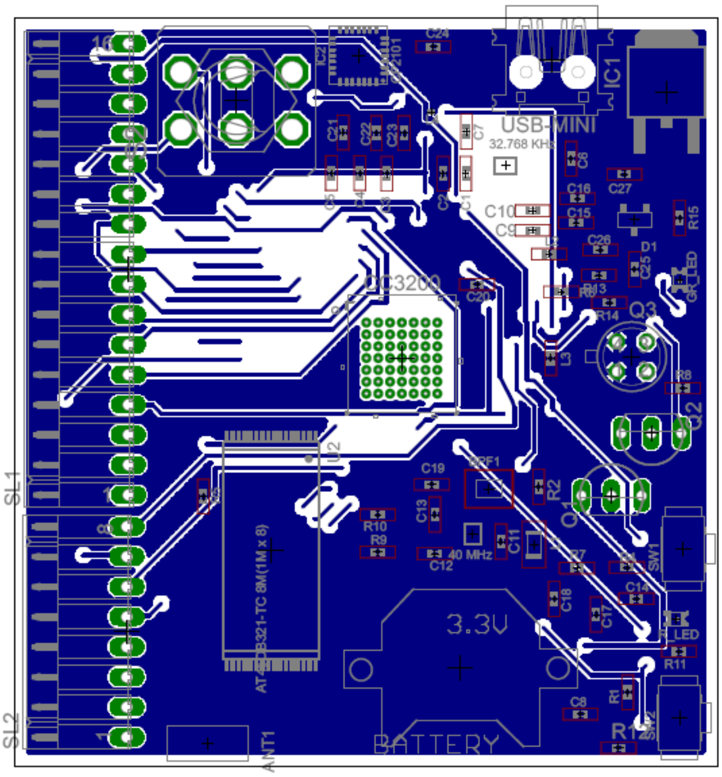


Figure 55: Routed bottom layer of the ATOMS project board.

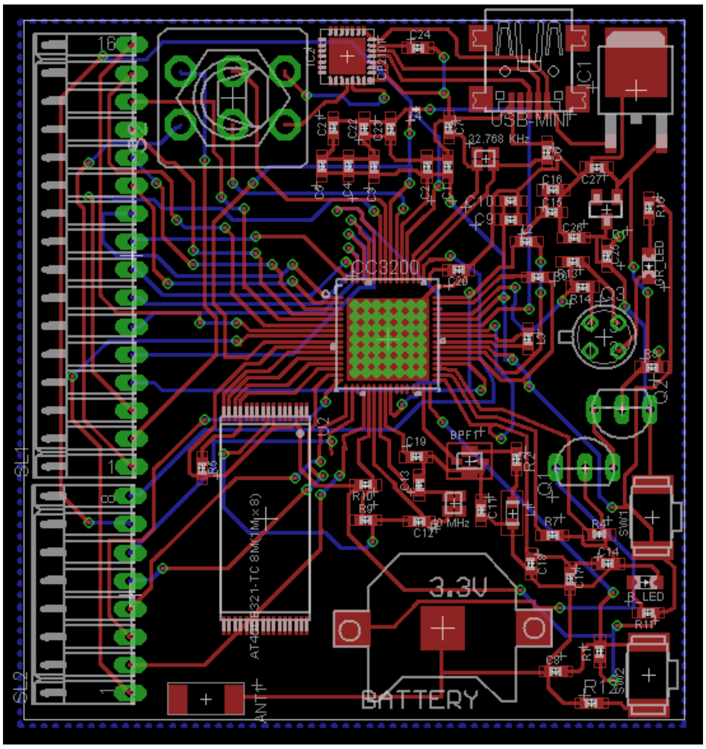


Figure 56: Routed top and bottom layer of the ATOMS project board.

ATOMS

Figure 57: Desktop software application logo.