Building and Environment 100 (2016) 114–126

Contents lists available at ScienceDirect

Building and Environment

journal homepage: www.elsevier.com/locate/buildenv

Open Source Building Science Sensors (OSBSS): A low-cost Arduino-based platform for long-term indoor environmental data collection

Akram Syed Ali, Zachary Zanzinger, Deion Debose, Brent Stephens*

Department of Civil, Architectural and Environmental Engineering, Illinois Institute of Technology, Chicago, IL, USA

ARTICLE INFO

Article history:
Received 13 December 2015
Received in revised form
25 January 2016
Accepted 11 February 2016
Available online 15 February 2016

Keywords:
Open source
Arduino
Sensors
Data collection
Indoor environment

ABSTRACT

Accurate characterization of parameters that influence indoor environments is often limited to the use of proprietary hardware and software, which can adversely affect costs, flexibility, and data integration. Here we describe the Open Source Building Science Sensors (OSBSS) project, which we created to design and develop a suite of inexpensive, open source devices based on the Arduino platform for measuring and recording long-term indoor environmental and building operational data. The goal of OSBSS is to allow for more flexibility in synchronizing a large number of measurements with high spatial and temporal resolution in a cost effective manner for use in research projects and, eventually, in building automation and control. Detailed tutorials with instructions for constructing the data loggers using off-the-shelf electronic components are made available freely online. The project currently includes a variety of sensors and data loggers designed to measure a number of important parameters in buildings, including air and surface temperatures, air relative humidity, human occupancy, light intensity, CO_2 concentrations, and a generic voltage data logger that can log data from a variety of other sensors such as differential pressure sensors. We also describe results from co-location tests with each data logger installed for one week in an educational building alongside their commercial counterparts, which demonstrate excellent performance at substantially lower costs.

1. Introduction

The need for collecting high quality data in buildings is continuing to increase [1], as better information on building performance can improve our understanding of energy use [2], thermal comfort [3], indoor environmental quality (IEQ) [4,5], and even the microbiology of the built environment [6]. Advanced building management systems can collect large amounts of data on building operation [7], but accurate characterizations of many indoor environmental parameters are often limited to the use of proprietary hardware and software, which can adversely affect costs, flexibility, and data integration [8]. As a result, many indoor environmental investigations are unable to collect widespread building performance or occupant activity data, which may negatively impact their effectiveness. Thus, many opportunities exist to reduce the costs of built environment data collection by, for example, moving from proprietary to open source platforms [9].

In response, we launched the Open Source Building Science Sensors (OSBSS) project in 2013 to facilitate more cost-effective data collection for a wide variety of important building environmental and operational parameters. The goal of OSBSS was to build a platform for sensor and data logger development that had the following features:

- Affordable price by utilizing inexpensive, yet high performance electrical components;
- Designed and constructed using open source hardware and software;
- Easy to build, program, and launch within ~2 h for novice users following detailed tutorials online;
- Long-term onboard data storage on SD cards at user-defined intervals;
- Accurate and synchronized time-keeping across all devices;

* Corresponding author. Department of Civil, Architectural and Environmental Engineering, Illinois Institute of Technology, Alumni Memorial Hall Room 212, 3201 South Dearborn Street, Chicago, IL 60616, USA.
E-mail address: brent@iit.edu (B. Stephens).

http://dx.doi.org/10.1016/j.buildenv.2016.02.010

- Measurement accuracy similar to commonly used commercial devices [10]; and
- Low-power draw and long battery life for typical building applications (i.e., approximately 1 year on a single battery for most sensors).

For reasons discussed in Section 2.1, we chose to use the open source Arduino platform as the base hardware for the data loggers and selected a variety of off-the-shelf sensors to build onto the platform to meet all of these goals. Here we describe the development of the OSBSS platform and demonstrate the performance of a number of prototype sensors installed for one week alongside their commercial counterparts in an educational building, including air and surface temperatures, air relative humidity, human occupancy, light intensity, CO_2 concentrations, and a generic voltage data logger that can log data from a variety of other sensors such as differential pressure sensors. Data from the relatively simple field study are meant to demonstrate the utility of the OSBSS platform in real-life applications. Data from many more controlled environment tests are provided in Ali, 2015 [11].

2. The OSBSS platform

The following sections describe important components of the OSBSS platform, including: Arduino hardware and software (Section 2.1); data storage (Section 2.2); power-saving strategies (Section 2.3); an accurate real-time clock (RTC) for time-keeping (Section 2.4); and battery life (Section 2.5).

2.1. Arduino hardware and software

The OSBSS project utilizes the open source Arduino prototyping platform in the development of the base data acquisition hardware (www.arduino.cc). The Arduino platform provides various development circuit boards, some of which utilize Atmel's low-power CMOS 8-bit microcontrollers based on AVR enhanced RISC architecture. These microcontrollers are capable of computing approximately 300,000 lines of program code per second, which is more than sufficient for most of the input and output applications required for the time scales of typical building data collection (e.g., seconds or minutes). Arduino provides an integrated development environment (IDE) that is capable of running on all major operating systems and has support for a simplified C/C++ programming language. Arduino also has a large online community that stimulates engagement in development and enables rapid prototyping and debugging.

Further, a large number of high-grade sensors and devices have custom Arduino libraries and active support from manufacturers for the platform. The Arduino platform has been used successfully in several other similar data collection efforts including embedding wireless sensor networks for temperature and humidity monitoring within concrete structures [12], monitoring human activity and integration via ZigBee and Wi-Fi Networks [13], and balancing envelope and heating system parameters for retrofit analyses using building sensor data [14]. These studies and others have demonstrated the reliability and ease of use of the Arduino platform for data collection, making it a viable choice for developing sensor prototypes for OSBSS.

The Arduino board chosen for the OSBSS prototypes was the Pro Mini, which is a bare-bones version of the more common Arduino Uno board. It is based on the same Atmel ATmega328P microprocessor, although it has extra peripherals removed and provides direct access to the microprocessor. Additionally, the microprocessor on the Pro Mini is a Thin Quad Flat Package (TQFP) instead of the much larger Dual In-line Package (DIP) on the Uno, which is advantageous because it comes in a much smaller footprint, thereby allowing circuits to be more compact and portable. All of the pins on the microprocessor are broken out on the board in a plated through-hole format to provide easy communication access. Headers must be soldered on the Pro Mini to allow interfacing external devices to it. The Pro Mini comes in two variations: 5 V with 16 MHz clock speed or 3.3 V with 8 MHz clock speed. The OSBSS data loggers use the 3.3 V 8 MHz version of the Pro Mini because most sensors we discovered are designed to work only on the lower 3.3 V supply voltage and often cannot tolerate the higher 5 V voltage levels.

2.2. Data storage

All first generation OSBSS data logger prototypes were designed to store time-stamped measurements on microSD cards. The 3.3 V version of the Pro Mini is ideal since microSD card specifications state that supply voltage and logic levels to and from microSD cards should be between 2.7 and 3.6 V. Using microSD cards for data storage provides virtually unlimited storage space because all sensor and time data are stored as plain text in a comma-delimited format and each data point consists of only a few bytes of data. At the time of writing, a typical microSD card is available in 8 GB capacity for less than $5, which can theoretically store billions of measurements. MicroSD cards also allow ease of use of data retrieval through interfacing with most modern computers.

Using William Greiman's SdFat Arduino library [15], the Pro Mini can be programmed to save all data in a *.txt or *.csv file format (or other supported formats). Since we are using delimiter-separated values, all data is typically stored in *.csv format, which can be opened in spreadsheet programs once retrieved. Since there is no way for the Pro Mini to interact with a microSD card directly, a microSD card breakout from SparkFun is used to communicate with the microSD card. During the course of testing, it was found that some spring-based microSD card sockets are known to provide loose connections in some cases where the build-quality is not sufficient, thereby causing frequent microSD card failures or data loss. Friction-based microSD card sockets are recommended for more reliable performance. Retrieving the data from a microSD card is as simple as taking the microSD card out from the socket and plugging it in a computer. Other forms of data storage such as EEPROMs or SPI flash memories were also explored but discarded for the prototypes due to limited memory capacity and additional programming required to extract data from them. However, they may prove to be more useful as backup memory for future wireless data loggers in the event of loss of communication with the gateway.

2.3. Power-saving strategies

The Pro Mini boards are modified to achieve lower current draw first by disconnecting the two on-board LEDs. This is done because the power LED ordinarily remains on at all times, unnecessarily drawing excess current. Further, in some Pro Mini clones, the status LED has a high value series resistor that can interfere with data transfer over the SPI bus. To avoid these issues, the traces leading up to these LEDs are cut with a sharp blade while making sure other components on board were not damaged.

To further reduce the current draw, various sleep modes of the ATmega328P microprocessor were explored. Given that the OSBSS data loggers are designed to take measurements either at user-defined intervals (e.g., once every minute) or on detection of an activity (e.g., proximity or motion), the microprocessor does not need to be active during any periods other than those that involve active data logging. Fortunately, the ATmega328P has several sleep modes that can be used to save power while the processor is not in use. Two of the deepest possible sleep modes in the microprocessor were tested, including ‘standby’ and ‘power-down’ modes. In the power-down mode, most of the internal peripherals, including the main processor clock and timer oscillators, are disabled.

Other power-saving techniques include disabling various modules when the microprocessor is in sleep mode such as the Analog-to-Digital Converter (ADC), Two-Wire Interface (TWI), Serial Peripheral Interface (SPI), Watch-Dog Timer (WDT), and Brown-Out Detection (BOD). All of these peripherals can be enabled once the Arduino wakes up from sleep mode, so they can still be used normally. Additionally, internal pull-up resistors on each I/O pin can be either activated or deactivated depending on the application, and each pin can be explicitly set to a defined state to minimize any quiescent current draw by external devices when the logger is in sleep mode. Combined, substantial reductions in current draw were achieved by these strategies (Fig. 1). Cutting traces to the LEDs reduced current draw from 6.41 mA to 0.31 mA in standby mode, while the combination of other power-saving strategies reduced current draw to as little as 0.1 mA in power-down mode.

Finally, to ensure that these low current draws could be practically achieved, external devices were turned off until needed. This includes turning off the microSD card and the sensors when feasible. An NPN bipolar junction transistor (BJT) is used to switch the microSD card off and other devices when not in use (i.e., when the Arduino is in sleep mode), which effectively eliminates any idle current draw. The transistor acts like a switch that can disconnect circuits and reconnect them when needed. It also allows a large amount of current to flow through, which is sufficient for devices that draw a large amount of current on power-up such as the microSD card. Using the transistor to disable the specific sensor on the data logger when it is inactive (i.e., not taking any measurements) also reduces current draw further. Most sensors are already low power draw devices, but disabling them completely provides for maximum battery life. However, care must be taken with sensors that have a warm-up time and are not designed for power-cycling. Extra code to reinitialize the sensors may be necessary after turning them back on.

In the case of the microSD card, power-cycling it at every logging interval does not appear to cause any negative impacts in its performance. This was tested several hundred thousand times over the course of several months to ensure the method works reliably. We should note that one drawback of using this method is that the microSD card requires the data file to be explicitly “closed” before the microSD card power is disabled. This step is necessary to prevent data file corruption, which may lead to data loss. The microSD card also needs to be reinitialized before it can be used every time it is powered on. This will add to the overall time the data logger must be powered on, but in most long-term applications at moderate logging intervals (e.g., 30 or 60 s), the time spent in active mode is likely insignificant as compared to the time spent in sleep mode. Overall, current draw during sleep mode is the largest contributor to the overall power draw [16]; thus, there is a very strong motivation to reduce it as much as possible.

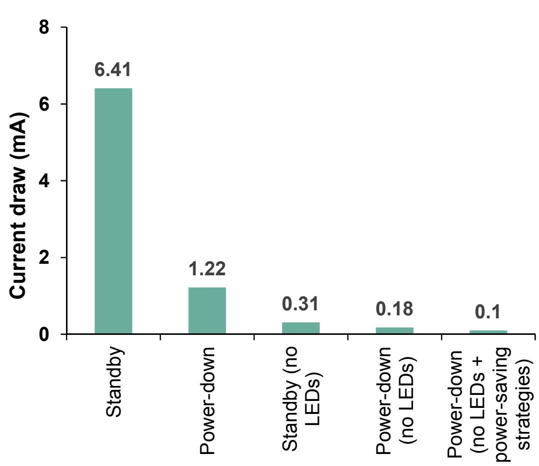

Fig. 1. Current draw during various sleeping states of the Arduino Pro Mini.

2.4. Accurate real-time clock for time-keeping

Two real-time clocks (RTCs) were initially tested for time-keeping, including a Maxim DS1307 and DS3234. The DS1307 was discarded due to several reasons, including the observation of a substantial time drift of several minutes over a few weeks. It also required 5 V to ensure reliable performance, instead of 3.3 V, which is standard in low-power applications. It also lacked alarm features that are essential for low power data logging applications that require defining a specific logging interval. In contrast, the DS3234 has been shown to be an accurate RTC with a temperature-compensated oscillator that precisely keeps time and is known to drift no more than a few seconds per year [17]. This means that the clock is able to calibrate itself for any drift that may occur due to varying temperatures. Further, several online sources have reported that any other drift that occurred with oscillators in clocks were mostly linear [18,19], suggesting that a long-term test can yield results that can be used for self-calibration to account for these drifts, thereby maintaining accuracy over several years without the need to constantly re-sync the time to a central server.

Since putting the microprocessor in power-down mode provides maximum power-savings, all OSBSS data loggers are designed to use this sleep mode when not actively measuring or recording data. However, as noted earlier, when the microprocessor is in power-down mode, most of the internal peripherals are disabled. Consequently, the microprocessor is unable to return from sleep mode by itself and remains in power-down state indefinitely. To overcome this issue, the ATmega328P has several features by which it can detect interrupts from an internal or an external source and wake itself up to resume normal activity. Interrupts are signals that interrupt the current processor activity. They are asynchronous, meaning that they occur outside the regular flow of the program, regardless of what the processor is doing. They may be triggered by an external event (e.g., a change in pin state) or by an internal event (e.g., a programmed timer or a software signal). Once an interrupt is triggered, it gets the highest priority. In effect, it pauses any currently running processor's program to execute an Interrupt Service Routine (ISR). This is a function or a block of code that runs whenever an interrupt is triggered. Once the function is completed, the program returns to what it was doing previously before the interrupt was triggered. In the case of the ATmega328P in power-down mode, interrupts can be set so that when the state of a predefined pin is changed due to an external event, the processor returns from power-down mode to execute the ISR. Because OSBSS data loggers require precise time keeping and are designed to take measurements at a predetermined logging interval, the DS3234 RTC is used to trigger the external interrupt event on the processor.

The RTC has two unique alarm features that can be enabled. These alarms allow the clock to generate an interrupt at any given date or time in the form of a square wave with an adjustable frequency. For example, if 1-min logging intervals are required, the clock can be set to generate an interrupt exactly at the 0 s mark of each minute. Since the RTC was verified to be highly accurate, the date and time when the interrupts occur can be used as a time stamp with any sensor measurement taking place at this time. The microprocessor is set to detect this interrupt generated by the RTC and is then returned from power-down mode. This is done by connecting one dedicated digital pin on the Pro Mini to the INT/SQW pin on the DS3234, which is the pin that generates interrupts. All pin configurations are done in bare AVR or assembly commands instead of the Arduino programming language. This was done since Arduino functions have some overhead in the form of other additional code and logic that run every time a single line of code is executed. Due to this, a block of code written in the Arduino language is executed more slowly than if the same code was written in bare AVR or assembly commands that interface with the microprocessor directly. In normal situations, this does not create any meaningful impact in operation; however, in operations where timing is more critical, using AVR commands are preferred.

2.5. Battery life

The DS3234 RTC maintains date and time using its own backup power: a single CR1225 lithium ion coin cell. Without external power to the clock module, the coin cell can maintain the RTC's time accurately for around 9 years. However, in this time-keeping mode, interfacing with the clock is disabled. Once external power is applied, the clock switches from battery to external power and communication with the clock is enabled. Due to this, the RTC has been kept powered on all the time in all of the prototypes of OSBSS data loggers. The RTC module consumes around 0.089 mA when active. The RTC, coupled with the Arduino Pro Mini in power-down mode (0.1 mA), consumes around 0.195 mA of current when idle. We rely on a typical lithium-ion polymer battery to power the data loggers. These batteries are more reliable than several AA or AAA batteries in series, which have a tendency to discharge unevenly, with one battery discharging faster than others in a set and thus dropping the voltage below minimum requirements sooner than a lithium-ion battery would. Lithium-ion polymer batteries have a mostly linear discharge profile, which is ideal for long-term logging applications. The data logger is setup to sleep, wake up at a specified interval to take measurements, save data to the microSD card, and go back to power-down mode.

To measure the actual current draw of a typical OSBSS data logger, a current draw profile was created by connecting the data logger to a Tektronix TDS1001B oscilloscope. A 1Ω resistor was connected in series to the power supply of the data logger. The probes of the oscilloscope were connected to both leads of the resistor and the voltage across the resistor was captured. Since the resistor is 1Ω, any voltage displayed by the oscilloscope can also be read as current using Ohm's law. The scope was set to trigger when a pulse length greater than 1 ms is detected. The measuring bandwidth of the oscilloscope was lowered to 20 MHz to reduce signal noise. For further reduction of noise, the scope was set to acquire 128 samples and display the average, rather than the raw signal itself. This setup was used to capture a current profile when a measurement takes place every logging interval, as shown in Fig. 2a. The horizontal and vertical divisions are set appropriately as shown.

Fig. 2b shows a zoomed-in version of the same profile from Fig. 2a. The sensor measurement combined with writing data to the microSD card takes under 200 ms. The current draw varies during this time, from ~13.5 mA during 112.5 ms when the data logger wakes up from sleep mode and takes measurements, to ~20–50 mA in several short bursts when the microSD card opens the file and writes data, to another series of short bursts of ~45–60 mA when the timestamp is written to the microSD card and the file is closed and the LED is blinked to indicate a successful data entry. By dividing the graph in three parts, as shown in Fig. 2b, and taking the average areas under the curve, the battery life can be estimated. With a 2500 mAh lithium-ion polymer battery as the power source and a 1-min logging interval, Equation (1) gives a close approximation of the expected battery life.

$$\begin{aligned} 2,500\ mAh \div \Bigg(\left(\frac{59.81s}{60s} \times 0.19mA \right) + \left(\frac{0.1135s}{60s} \times 13.5mA \right) \\ + \left(\frac{0.04s}{60s} \times 33mA \right) + \left(\frac{0.03s}{60s} \times 45mA \right) \Bigg) \\ = 9,636\ hours\ (\sim 401.5\ days\ or\ \sim 1.1\ years) \end{aligned} \quad (1)$$

Taking the self-discharge rate of the battery [20] as well as impact of battery life due to varying temperature [21,22] into consideration, the actual battery life is more likely to be around one year for a typical OSBSS data logger. Although this estimate can vary with the type of sensor used, this level of battery life is more than adequate for most building applications and is comparable to many commercially available data loggers.

Once the base hardware and software was setup and tested, most of the code was compiled into various libraries for Arduino. This was done so that the power-saving techniques can easily be used with just one line of code referencing the library. The libraries are developed specifically for the ATmega328P processor, which means that any Arduino board based on the ATmega328P processor will be able to use these libraries. These libraries can be viewed and downloaded from the OSBSS GitHub repository at https://github.com/OSBSS. All components are soldered on an Adafruit Perma-proto half-sized solderable breadboard to provide robust connections on a platform that is also easy to construct by those unfamiliar with electronics. Detailed tutorials, technical documentation, and support for all devices are available on http://www.osbss.com.

3. Development of several sensor prototypes

Fig. 3 shows the basic layout of an OSBSS data logger in schematic form. The dotted lines connecting the sensor indicate that the sensor is largely replaceable with others, depending on the specific application. All sensors are powered-down using the NPN transistor to conserve power, unless noted otherwise.

Utilizing this low-power OSBSS platform based on the Arduino Pro Mini (ATmega328P), several prototypes using existing off-the-shelf sensors were developed to construct individual OSBSS data loggers. The data logger prototypes include: (1) temperature and relative humidity, (2) surface or airstream temperature, (3) occupant proximity, (4) CO_2, (5) light intensity, and (6) a generic analog voltage data logger that can be used to record outputs from other commercial sensors such as differential pressure transducers or other CO_2 sensors. Fig. 4a–f shows OSBSS data logger prototypes, each of which include an Arduino Pro Mini, a DS3234 RTC, a microSD card holder, a transistor and the specific sensor. Full details for the development of each sensor are provided in the next sections.

3.1. Temperature and relative humidity

A US Sensor ultra-precision NTC thermistor (part no. PR103J2) and Sensirion SHT15 digital humidity sensor were chosen for temperature and relative humidity (T/RH) measurements, respectively. The thermistor has a reported accuracy of ±0.05 °C and can operate between −55 °C and +80 °C, satisfying almost all relevant indoor and outdoor conditions in most buildings. It has an epoxy coating over the head, which gives a response time of 5–10 s in open air. In most indoor conditions, temperature generally changes slower than this, so this response time is considered adequate. This

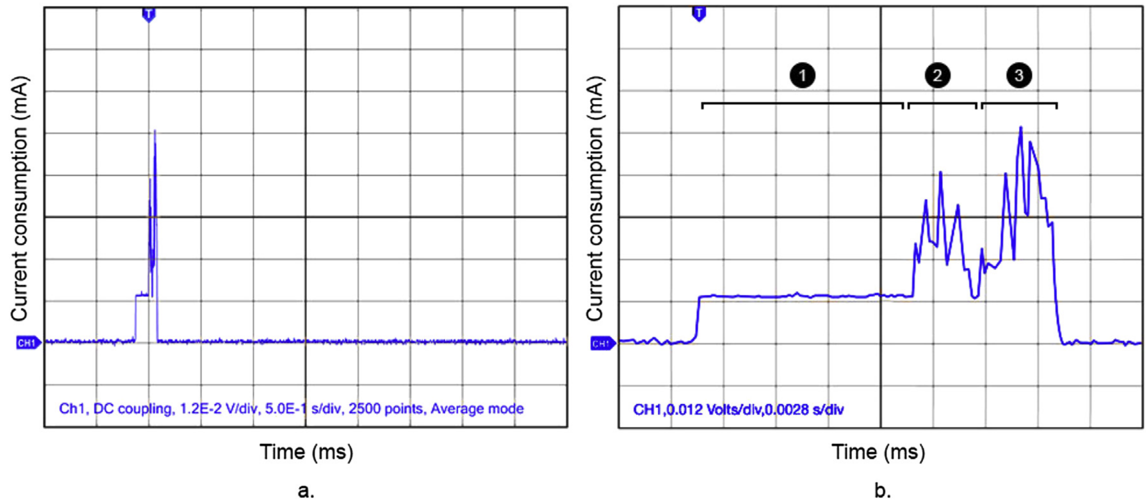

Fig. 2. Current consumption profile of an OSBSS temperature and relative humidity data logger during a measurement: (a) zoomed-out version and (b) zoomed-in version.

particular thermistor has a negative temperature coefficient, which means its resistance varies inversely with temperature. This value is measured as a voltage reading over a 10,000Ω resistor and converted to the actual resistance of the thermistor using Ohm's Law. The resistance value of the thermistor is then converted to a temperature reading based on a customized Steinhart-Hart equation derived from the resistance versus temperature data for the specific thermistor, which is provided by the manufacturer.

Using a resistance-temperature calibration equation for calculating an unknown temperature from the resistance of NTC thermistors has been proven to yield good approximations of temperatures with differences within a few milliKelvin [23]. Since the thermistor is a non-polarized analog device, it can be connected to one of the analog input pins on the Arduino. Using the analog-to-digital convertor (ADC) on the Arduino provides 10-bit readings of analog voltage. This means that any voltage reading between 0 and 3.3 V can be correlated to a value between 0 and 2^{10}, or 1024. Using a higher resolution external ADC would provide better readings, however, since the thermistor has an accuracy of ± 0.05 °C in optimal conditions, the level of precision from the 10-bit ADC is sufficient. To increase the accuracy of the ADC readings and reduce the impact of electrical noise by surrounding components, several consecutive samples can be taken and averaged.

Sensirion's SHT15 was chosen for relative humidity measurements due to its known reliability and long-term stability. The sensor element can also self-calibrate over time after being exposed to relatively polluted or extreme environmental conditions, ensuring an accurate relative humidity reading over an extended period of time. The same sensor chip is also available with pins instead and is labeled as the SHT75, which is the same one used in the commercially available Onset HOBO U12 temperature and relative humidity loggers. The SHT15 sensor can also perform temperature measurements; however, since the OSBSS T/RH data logger utilizes the more accurate US sensor thermistor for temperature readings, the SHT15 temperature readings are not used and the temperature compensation for relative humidity calculations is done based on the thermistor temperature readings alone. This provides slightly higher accuracy relative humidity readings; however, this requires the thermistor to be in close proximity to the SHT15 sensor, to ensure the temperature readings correlate to the actual temperature in and around the sensor chip.

3.2. Air stream/surface temperature

The OSBSS T/RH data logger setup can also be used to measure temperatures on surfaces or in fluids (e.g., air or water) with small modifications to the hardware setup. The humidity sensor is removed and the thermistor is soldered on a long wire of adequate length. The leads of the thermistor are protected against shorting each other by covering them both with heat shrink tubing. The

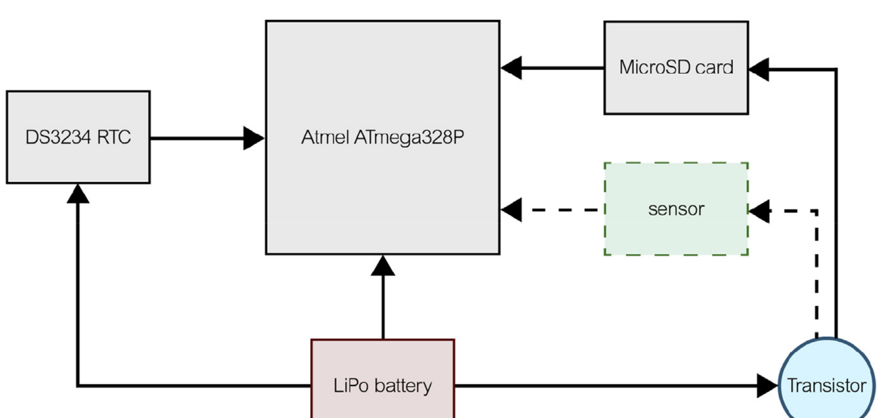

Fig. 3. Basic layout of an OSBSS data logger.

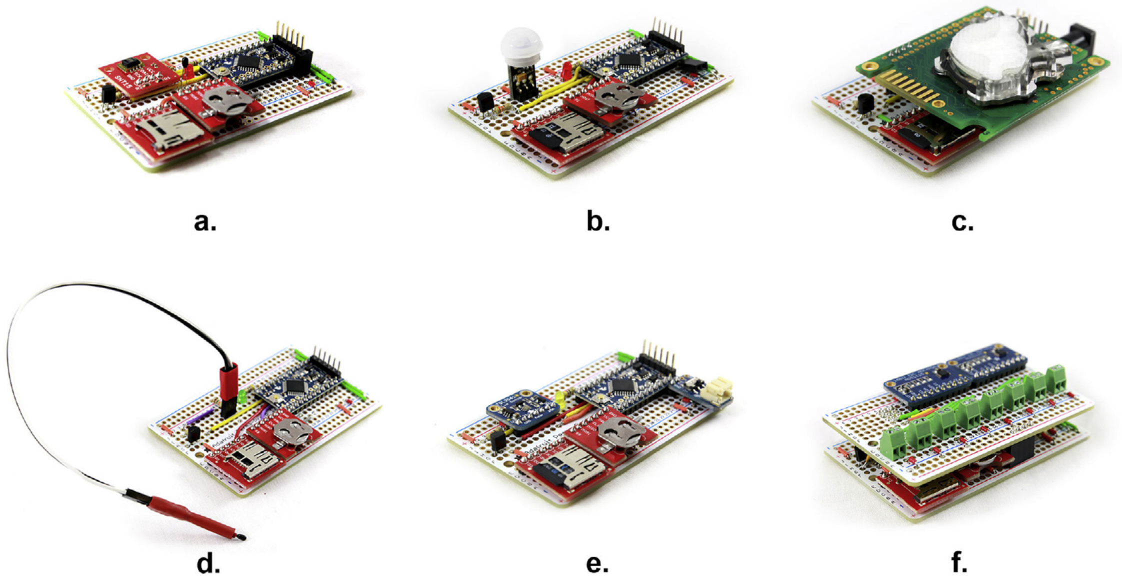

Fig. 4. OSBSS data logger prototypes: (a) temperature and relative humidity; (b) occupant motion/proximity; (c) CO_2 concentration; (d) air stream/surface temperature; (e) light intensity; and (f) generic voltage data logger.

thermistor head and insulated leads can optionally be covered with a small layer of hot glue making the thermistor water-resistant for use in liquids. However, applying a large quantity of hot glue may increase the response time of the thermistor. This data logger can also be used for non-invasive temperature measurements in air streams, such as in HVAC systems where access to the data logger needs to be kept outside but the measurement needs to be kept inside the system.

3.3. Light intensity

The TAOS TSL2561 luminosity sensor was chosen for light intensity measurements. This comes on a breakout board by Adafruit. It has a digital two-wire interface and the address of the sensor can be set manually. The TSL2561 is a digital light sensor with a relatively wide sensing range of 0.1–40,000 lux. Typical low cost photoresistors or photocells are variable resistors made of high resistance semiconductor, usually cadmium sulfide. They exhibit photoconductive properties, in which the resistance of the photocell decreases with increasing incident light intensity, similar to the working of a thermistor. These inexpensive cells are found in many consumer items such as camera light meters, clock radios, alarm devices, night-lights, clocks, and street lamps. While these cells are good for applications that require light and dark switching, they are not accurate for measuring exact light intensity. Various photoresistors also react differently to photons in different wavelength bands of light and in different temperatures.

In contrast, the TSL2561 provides a more precise solution allowing for lux calculations in a wide range of operating environmental conditions. The sensor also contains both infrared and full-spectrum diodes, allowing a much wider analysis of light intensity over a wider range of wavelengths, which approximates the response of the human eye. The sensor contains all circuitry on board to provide a digital output that can be processed easily. The sensor also has programmable and automatic gain settings to adapt to extremely bright or dark conditions. It has a high dynamic range and automatically rejects 50/60 Hz lighting ripples from common artificial light sources. The sensor also has low current draw while active, allowing for extended battery powered operations.

3.4. Occupant proximity/motion

Measurements of human proximity near surfaces can be used to assess human occupancy and activity in a room [6]. From preliminary testing, the Parallax PIR mini sensor was chosen for motion detection and occupancy measurements. The Parallax PIR mini sensor can detect motion up to 12 feet (3.65 m) away. It has a 100° detection angle horizontally and vertically. It has a wide operating supply voltage range of 3–15 V. This sensor was also chosen because of its low current consumption of around 35 μA at 3.3 V. The Parallax PIR mini sensor has a compact form factor that allows easy integration with the OSBSS platform. Communication with the sensor is by means of the OUT pin. When the sensor detects motion, it provides a "high" output on the OUT pin, which can be read as logic 1. When motion is no longer detected, the sensor outputs a "low" signal, or logic 0. Since the output by the sensor has only two logic states, it can be connected to any digital pin on the Arduino, which can be configured to read the state of the OUT pin. Based on the value read by the digital pin, data can be written to the microSD card. Each data point is stored with a timestamp and either 1 or 0, with 1 indicating motion and 0 indicating the absence of motion.

Since the proximity data logger is dependent on motion in a room to save data rather than a logging interval, the RTC can be switched off when not in use to save power. In this case, the RTC no longer generates interrupts at a specified interval, and data is only saved to the microSD card when motion is detected. The PIR module however, needs to remain active all the time. To detect this state of motion in a room, a pin-change interrupt is configured on the digital pin connected to the OUT pin of the PIR module. The PIR module has a factory-set timeout period of approximately 2 s after motion, or lack thereof, is detected. Since the RTC is disabled in this setup with only the Arduino (in sleep mode) and the PIR module enabled, the idle current draw of the proximity data logger is lower than that of other OSBSS data loggers that depend on the RTC to be enabled to generate an interrupt at a specific logging interval. The PIR module consumes around 35 μA (0.035 mA) at all times, bringing the total idle current draw of the PIR data logger to ~0.13 mA. Because the occupancy logger does not record at user-defined intervals, its battery life will be influenced by how active the space is. In low or moderately active buildings, we estimate the battery life to be at least one year given its low idle current draw.

3.5. CO_2 concentration

Carbon dioxide (CO_2) concentrations within an indoor environment can provide accurate estimates of occupancy [24] and/or ventilation rates [25] using mass balance approaches, provided that other information is known. After testing several CO_2 sensors, a SenseAir K-30 1% CO_2 sensor was chosen for CO_2 concentration measurements. The K-30 CO_2 sensor is a low-cost, maintenance-free sensor based on non-dispersive infrared (NDIR) technology. Since infrared energy passes through an atmospheric sampling chamber without deformation, it is widely used in gas sensor modules. The K-30 sensor has an optical filter that is designed to eliminate all light except the wavelength that CO_2 gas molecules can absorb (4.26 μm). It has a measurement range of 0–10,000 ppm with an accuracy of ±30 ppm, which is better than many commercially available products commonly used in buildings [26]. It has a warm-up time of around 1 min, which is relatively rapid compared to commercial alternatives such as the PP systems SBA-5 CO_2 gas analyzer, which has a warm up time of around 20 min. It has a wide operating supply voltage range of 5–14 VDC.

Two linear signal outputs provide CO_2 concentrations at 10-bit resolution up to 2000 ppm, each configured separately. The OSBSS CO_2 data logger uses a Two-Wire Interface (TWI) for communication with the CO_2 module. TWI is a protocol that allows microcontrollers to communicate with several low-speed peripherals such as many environmental sensors. Since the K-30 module has a hardware implementation of TWI, Arduino boards can be easily interfaced with the sensor. The sensor's default address can be manually changed in the software provided by CO2Meter, Inc. called GasLab. Having different addresses can allow multiple K-30 sensors to be connected to one Arduino board. The K-30 sensor automatically takes CO_2 measurements every 2 s and saves this value on its internal memory in 4 bytes, which can then be retrieved and translated by the Pro Mini when desired. Requesting measurement values from the sensor more often than 2 s is not recommended as it can result in lockups due to overflow of the TWI buffer. Checksum errors can happen occasionally with the sensor dropping values. This is because the sensor is designed to give the highest priority towards taking measurements. When the sensor takes a measurement, other peripherals are either disabled or set to low priority to reduce electrical noise or avoid potential failures. Due to this, any communication made to the sensor while it is taking a measurement may give checksum errors. Adequate timing delays are also necessary for ensuring the TWI bus does not lock up. However, these are small tradeoffs in return for high accuracy measurements for a very affordable price.

Since the K-30 module requires a minimum power supply voltage of 5 V and has an average current consumption of 150 mA (which also increases briefly to 300 mA when taking a measurement), we recommend that this sensor remain powered on all the time for data collection. We have chosen not to implement power-cycling for the CO_2 sensor because the sensor requires a 1-min warm-up time that must be met for accurate and stable readings. Having the sensor powered on for that long before every reading would negate any power savings achieved from turning the sensor off, particularly for higher time resolution applications (although this may be appropriate for longer interval sampling applications). Power-cycling the sensor can also affect its durability and lifespan. We recommend powering the CO_2 data logger with a dedicated power supply plugged into an outlet for longer-term data collection, as the battery life in always-on mode with a 2500 mAh lithium-ion polymer battery is approximately one day. However, this is not an uncommon limitation for many commercial CO_2 sensors.

3.6. Generic voltage data logger

Generic data loggers are ubiquitous tools that can be used to store and transmit data from a wide variety of sensors. Data loggers read output from sensors in units of voltage or current, which are correlated to actual values of the unit of measure according to the sensor. Ubiquitous data loggers that can both store data locally and transmit data wirelessly are an important component of any robust sensor network. To convert voltage outputs of sensors into tangible data that can be analyzed, an Analog-to-Digital Converter (ADC) is used. The OSBSS generic data logger uses the Texas Instruments ADS1115 16-bit ADC, which also comes on a breakout board by Adafruit. It performs data transfer over a TWI compatible serial interface and the address of the sensor can be manually set so that up to four ADS1115s can be connected together to allow a total of sixteen single-ended channel inputs, or eight differential channels. It has an onboard low-drift voltage reference and oscillator to maintain accurate readings. The data sample rate can is programmable from 8 up to 860 samples per second. It can read signals as low as ±256 mV, and has a programmable gain amplifier of up to 16× to boost up small signals to full range. The ADC is capable of operating from a single power supply ranging from 2.0 to 5.5 V. Regular spring terminals or screw terminals can be used for allowing sensors with analog output to connect to the generic data logger. When taking voltage measurements, an average of 10–20 consecutive samples can be taken to reduce noise from other internal components.

4. Testing and validation of OSBSS data loggers in an educational building

The performance and accuracy of each of the OSBSS data loggers was first tested alongside several commercial sensors and data loggers in controlled laboratory conditions [11], followed by co-location tests in the field. The commercial equivalents of each sensor chosen included: (1) Onset HOBO U12-012 for temperature and relative humidity; (2) Onset HOBO UX90-005 for proximity/motion detection; (3) PP systems SBA-5 and Telaire 7000 series air quality monitor, both connected to an Onset HOBO U12-012 for CO_2 concentrations; (4) TMC20-HD connected to an Onset HOBO U12-012 for air stream temperatures; (5) Onset HOBO U12-012 for light intensity; and (6) Veris PXUXX055 connected to an Onset HOBO–U12-012 for differential pressure measurements (also recorded by the OSBSS generic voltage data logger). These commercial devices were chosen for comparison because they are widely used in both industry and academic research [3,4]. Co-location of both OSBSS loggers and their commercial equivalents was done in two locations in an educational building on the main campus of Illinois Institute of Technology in Chicago, IL, including a graduate student lab in the core of the building and a faculty office on the perimeter. These two rooms were selected to collect data from two very different environments, as described below. Data collection occurred at 1-min intervals over a period of 7 days for both commercial and OSBSS data loggers. Fig. 5 shows a plan view indicating the placement of the sensors in the lab and office spaces.

In the lab space, the T/RH, light, and CO_2 sensors were placed on top of shelves to ensure that data collection was not affected by human activity within immediate proximity to the sensors. The motion sensors were installed on a wall facing the main door to capture the daily occupancy/activity of the room. There were no exterior windows in the lab. The air temperature sensors were

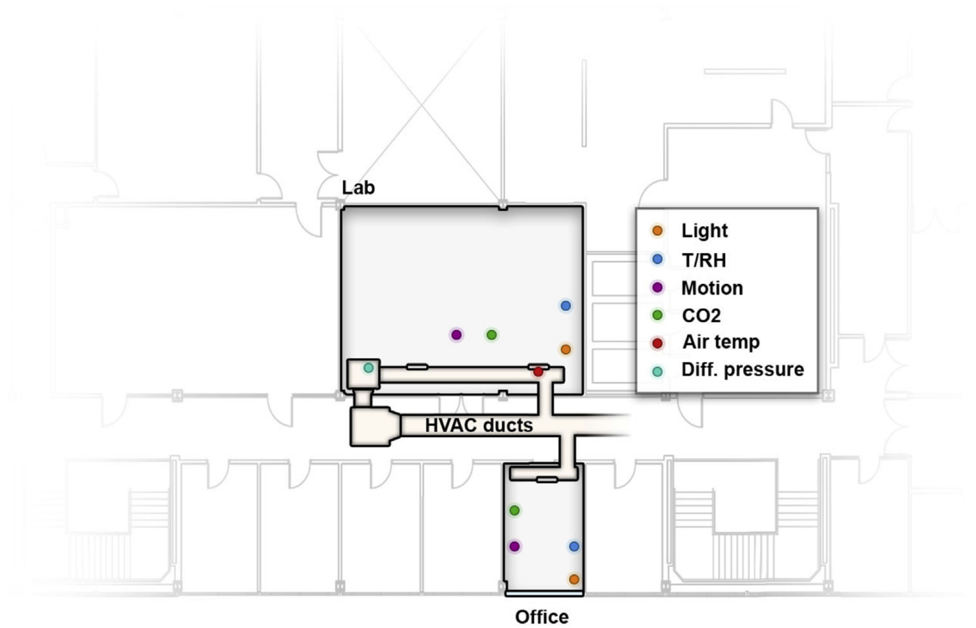

Fig. 5. Placement of OSBSS data loggers for co-location and validation testing.

placed inside the supply register of the HVAC duct that runs near the ceiling within the room. The generic data loggers logging differential pressure were placed inside a large supply/return diffuser. All sensors were placed at a height of 1.5 m, with the exception of the air temperature and generic data logger, which were installed just outside the HVAC ducts overhead with the temperature sensor and pressure tap placed inside the duct. The OSBSS generic data logger and the Onset HOBO U12 were both connected to the output

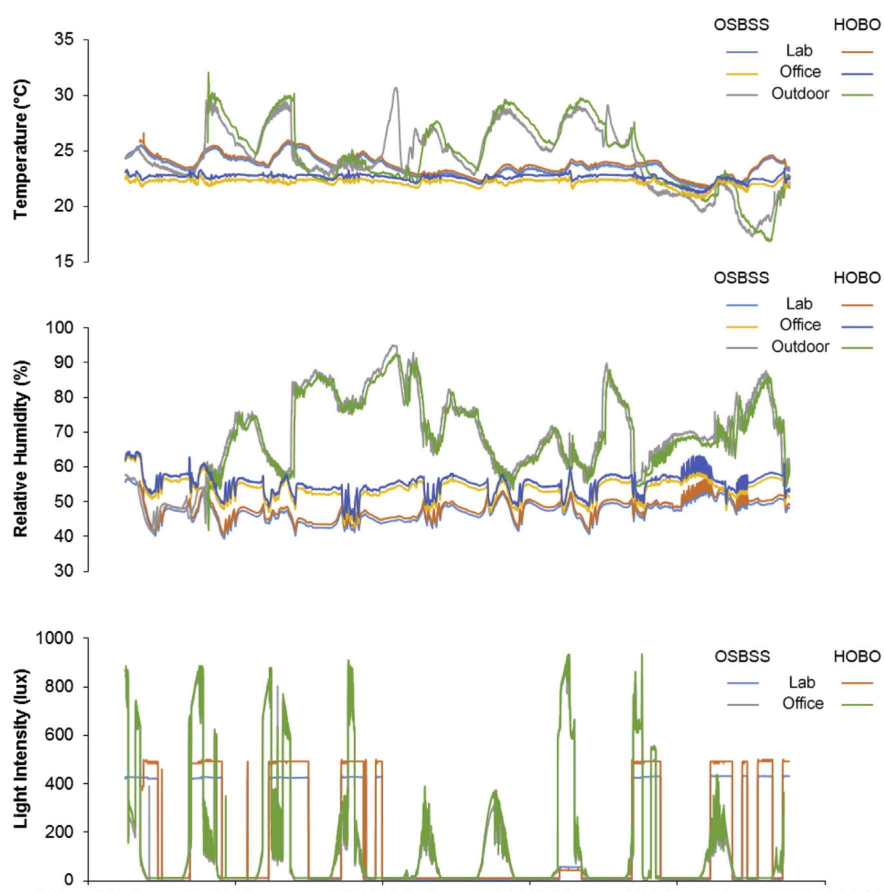

Fig. 6. Temperature, relative humidity and light intensity measurements in a laboratory and office space using OSBSS data loggers and their commercial counterparts.

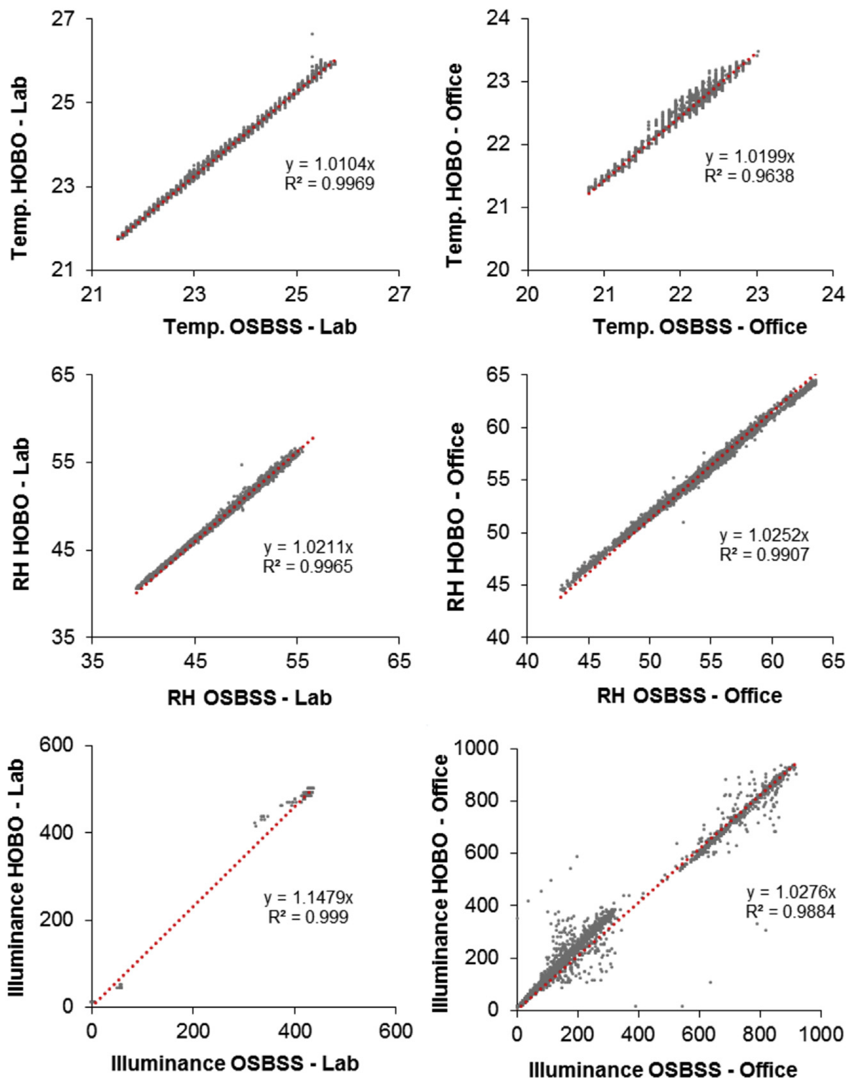

Fig. 7. Correlations between OSBSS temperature, relative humidity, and illuminance data loggers and their commercial counterparts.

of the same pressure transducer in order to compare the analog to digital conversion by each logger.

In the office, the T/RH, light, and CO_2 sensors were placed on top of shelves and the motion sensor was placed on a faculty member's desk. The light sensors were placed near the windows to capture a combination of natural and artificial light throughout the day (the office contained a large exterior south-facing window). The air temperature and differential pressure inside the HVAC duct in the office was not measured, since it shares the same supply duct branch as the lab.

Finally, an OSBSS T/RH data logger and an Onset HOBO U12 were both placed outside under cover to log outdoor environmental conditions at the same interval as the indoor loggers. All OSBSS data loggers were placed immediately adjacent to their commercial equivalents to ensure the same environmental conditions were being measured. Additionally, it was found during initial testing that the added thermal mass from enclosure of the Onset HOBO U12 data logger caused a small lag in temperature and relative humidity readings relative to the bare OSBSS logger without an enclosure. Thus, for the co-location tests, both the OSBSS and Onset HOBO U12 data loggers were operated without enclosures.

5. Results

Temperature, relative humidity, and light intensity data from the OSBSS data loggers and their commercial counterparts installed in both the lab and office space are illustrated in Fig. 6. Linear correlations between these same measurements are shown in Fig. 7.

OSBSS and HOBO data loggers revealed strong correlations for both temperature and relative humidity readings in both the lab and office spaces throughout the duration of the tests ($R^2 > 0.96$ and slopes between 1.01 and 1.03 for all four comparisons). The maximum deviation was only 0.1–0.2 °C for temperature and 0.5% for relative humidity. Fig. 6 also demonstrates that OSBSS data loggers were able to detect relatively rapid changes in temperature and relative humidity that occurred when the HVAC system was operating in the cooling mode.

Light intensity measurements were also strongly correlated between OSBSS and HOBO data loggers in both locations ($R^2 > 0.98$ for both), although slopes varied between locations (i.e., 1.15 in the lab and 1.03 in the office). For both loggers, light intensity measurements varied substantially based on the location of the data logger. The loggers placed in the lab had a repeated on and off

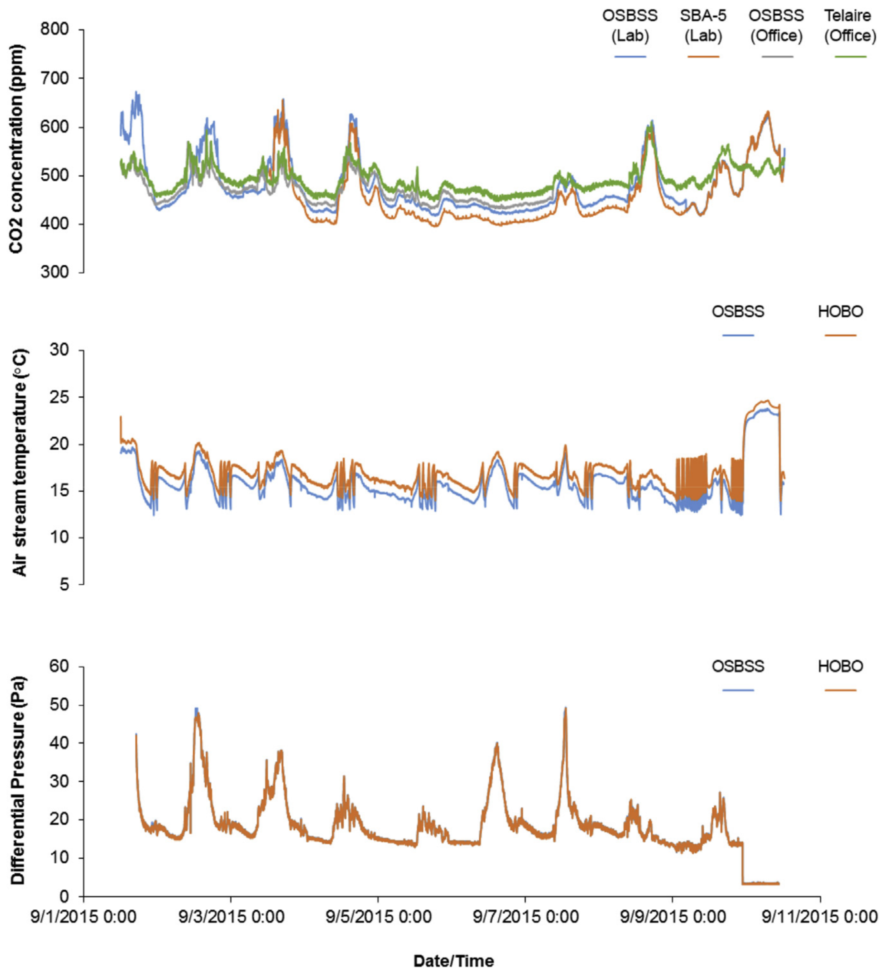

Fig. 8. CO_2 concentrations measured in a laboratory and office space and air stream temperatures and differential pressures measured in the supply ductwork of the variable air volume (VAV) HVAC system serving the same spaces made using both OSBSS data loggers and their commercial counterparts.

profile, which is indicative of the presence of artificial lighting and the lack of exterior windows. A constant high value of 400–500 lux indicated times when the room was occupied and lit, while a constant low value of approximately 0 lux indicated an absence of activity. The loggers placed in the office however showed a much more variable profile due to the presence of large windows that cover approximately 70% of the south wall. Daily light profiles followed a bell-shaped trend with highs of almost 900 lux, showing the presence of sunlight in the room. Those days with a maximum of ~400 lux inside the office room indicated overcast skies outside.

In all cases, the HOBO data logger showed higher lux values (and more so in the artificially lit lab space). To further explore these differences, spot measurements of light intensity were performed using two desk lamps each with 14W compact fluorescent bulbs with a color temperature of 5000 K. The desk lamps were placed at a fixed height of 30 cm and both sensors were tested alongside several commercial light meters, including a Tondaj LX 1010B and Tondaj LX1330B. These measurements showed that HOBO's light sensor always produced higher values, while the TSL2561 sensors used in OSBSS data logger produced lower values that were also more similar to the Tondaj readings. Comparing the datasheet for both OSBSS and HOBO light loggers, the spectral response of the TSL2561 is more sensitive than the light sensor on the HOBO. Regardless, both demonstrate the ability to make useful light intensity measurements in these two spaces.

Similar to Fig. 6, Fig. 8 shows CO_2 concentrations in the same office and lab spaces, as well as air stream temperatures and differential pressures in the supply duct connected to the variable air volume (VAV) HVAC system that served both spaces, measured again with both OSBSS and commercial counterpart sensors and loggers.

Fig. 8 again clearly shows that both OSBSS and their commercial counterpart sensors and data loggers detected typical phenomena in buildings at relatively high time-resolution, including varying CO_2 concentrations with varying occupancy and ventilation conditions (between ~400 and ~700 ppm, depending on the sensor and location), varying air stream temperatures in the supply duct (between ~12 and ~24 °C) as the system cycled between cooling and fan-only periods, and variations in differential pressure between the supply duct and the room (between ~13 and ~50 Pa) as the VAV system varied airflow rates (even shutting off towards the end of testing). Fig. 9 shows linear correlations between each of these measurements.

Again, the OSBSS data loggers showed strong correlations with their commercial counterparts. Correlations between differential pressure measurements recorded by both the OSBSS and HOBO voltage data loggers had an $R^2 = 0.99$ and slope = 0.99 across the measured range of 0–50 Pa. Correlations between air stream temperatures measured and recorded by both OSBSS and HOBO loggers had an $R^2 = 0.98$ and slope = 1.06. The air stream temperatures were actually offset by ~1 °C. However, subsequent measurements with the OSBSS and HOBO TMC20-HD temperature

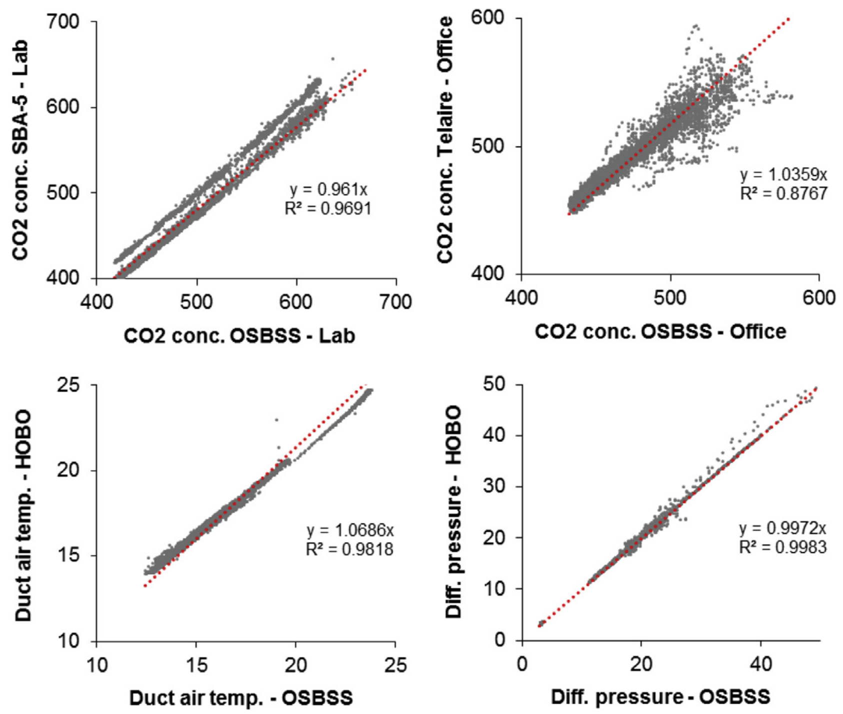

Fig. 9. Correlations between OSBSS CO_2, air temperature in the duct, and differential pressure and their commercial counterparts.

probes immersed in an ice bath demonstrated that the TMC20-HD sensor read approximately 1 °C while the OSBSS probe read approximately 0 °C.

For CO_2 concentrations, the OSBSS CO_2 data logger showed strong correlations with the Telaire sensor, albeit with some scatter ($R^2 = 0.87$ and slope = 1.03), while correlations were even stronger with the more expensive SBA-5 sensors ($R^2 = 0.96$ and slope = 0.96). Interestingly, comparisons between the SBA-5 and the OSBSS logger with the K-30 CO_2 sensor reveals two distinct trend lines, each with a stronger correlation coefficient than the overall value. The reason for these two lines is that the K-30 sensor has built-in algorithms to perform automatic calibration to background values (i.e., 400 ppm), in which it periodically checks the minimum recorded CO_2 concentration over the course of several days and compares to the baseline value. If this value appears to have drifted, the sensor self-calibrates its lowest observed

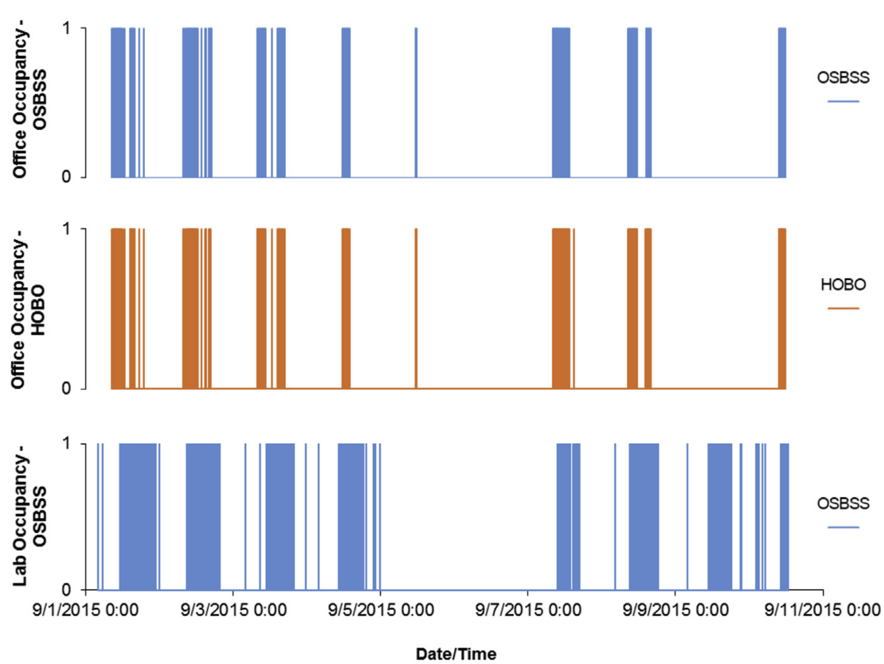

Fig. 10. Proximity-based occupancy measurements made in an office using both an OSBSS prototype and an Onset HOBO UX90, as well as occupancy measurements made with only the OSBSS prototype in a laboratory.

Table 1
Cost comparison of OSBSS data loggers versus commercial counterparts (in USD as of 2015).

Measure	Comparable commercial counterpart		OSBSS versions	
	Make/model	Unit cost	Average total cost per unit	Estimated savings per unit
Air temperature and relative humidity	Onset HOBO U12-012	\$140	\$100	\$40 (29%)
Surface or air stream temperature	Onset HOBO UX120-006 and TMC x-HD	\$175	\$60	\$115 (66%)
Light Intensity	Onset HOBO U12	\$140	\$70	\$70 (50%)
CO_2	Extech CO210	\$300	\$150	\$150 (50%)
	Telaire + U12 data logger	\$600	\$150	\$450 (75%)
	PP SBA-5 + U12 data logger	\$2100	\$150	\$1950 (92%)
Generic voltage logger	Onset HOBO UX120-006	\$140	\$70	\$70 (50%)
Proximity sensor	Onset HOBO UX90-005	\$205	\$70	\$135 (66%)

concentration down to this value. Fig. 8 shows that this automatic calibration occurred early on September 9, 2015, which then brought concentrations even closer in line with the SBA-5 and thus resulted in the second linear relationship shown in Fig. 9. This feature of the K-30 sensor is quite useful for long-term recording of CO_2 concentrations, as the sensor can largely operate free of maintenance. Moreover, repeated testing of several K-30 sensors in the lab has shown stable performance and high repeatability compared to commercial devices.

Finally, Fig. 10 shows profiles of occupancy measurements made with both the OSBSS proximity logger and the HOBO UX90 occupancy logger in the office, as well as an additional profile of proximity occupancy measurements in the lab made only with the OSBSS logger (we only had one UX90 available for comparison and we only installed it in the office).

Occupancy measurements again demonstrated that both OSBSS and UX90 sensors output very similar results. In all profiles, the dense bars indicate a large amount of occupant activity (typically during daytime periods), while a zero value indicates no activity (typically during nighttime and weekend periods). The lab space, which has as many as 10–12 occupants at peak times, was clearly shown to be more active/occupied than the single-occupancy office. These occupancy measurements also correlate well with CO_2 concentrations and light intensity levels in Figs. 6 and 8.

6. Discussion

Results from this case study deployment of the OSBSS network of devices alongside commercial counterparts clearly demonstrate that the suite of instruments can be used to detect a large number of relevant occupancy and building environmental and operational parameters with high accuracy. The main utility of the OSBSS platform is that it is open source, customizable, and inexpensive, and thus we encourage its use to enable making a larger number of built environment measurements with high spatial and temporal resolution in a more cost effective manner. To illustrate potential cost savings relative to commercial products, Table 1 lists the per-unit parts costs of these OSBSS data loggers compared to some of their commercial counterparts. Costs are provided in USD and are current as of 2015. Importantly, these cost estimates (a) assume that parts are purchased individually from the same retailers we used and not purchased in bulk, and (b) do not include the time value of labor to construct the OSBSS loggers from component parts (which, in our experience, was up to 2 h per logger for a novice user).

Most of the OSBSS data loggers have an expected cost savings of between 50 and 75% when compared to their commercial alternatives. This, coupled with the fact that the OSBSS devices are built on an open-source platform that allows for future upgrades in hardware and software as well as custom setups tailored for particular applications, clearly demonstrates the utility of the OSBSS platform of devices.

7. Limitations and future developments

We should note that the OSBSS platform has some limitations that we continue to improve upon. For one, the loggers require one to manually assemble all the parts on the board, solder the circuit, download the libraries, program the logger, and finally deploy it in the field. Thus, constructing each logger can be time consuming and, more importantly, there are many potential sources of error. Debugging these circuits can be relatively difficult and time consuming in the event of a problem. We continue to test the functionality and readability of the tutorials, but we are also working on designing printed circuit board (PCB) versions that will allow for more rapid sensor builds for more advanced users and limit some of the time and effort required for soldering. We are also working on building a graphical user interface (GUI) that would allow the end user to launch all sensors without the need of any assembly or coding. Additionally, through the nature of being open source, we have often found errors in libraries or code downloaded from the Internet. We continue to find and fix these problems, but these issues can limit the stability of some loggers. Finally, we have also recently encountered issues with power-cycling with newer microSD cards that users should be aware of.

We are also working on a number of other new additions to the platform, including: the addition of stop and start logging buttons; improving the ability to input sensor metadata; adding geospatial location functions; improving RTC time syncing with the PC and handling of daylight savings time; exploring other onboard storage options (e.g., flash memory); adding new sensors (e.g., surface proximity, particulate matter concentrations, binary door and window opening, linear window openings, air velocity, solar radiation flux, plug load power draw, and others); and adding wireless transmission capabilities.

In fact, we have very recently successfully demonstrated early wireless prototypes. Early tests of wireless data transmission using HopeRF's RFM69 transceivers show promising results when comparing reliability, cost, power draw, and distance of transmission with other wireless protocols such as ZigBEE (IEEE 802.15.4) or Wi-Fi (IEEE 802.11). RFM69 transceivers include both a transmitter and a receiver that can operate over a wide frequency range with very low power draw when they are in sleep mode. They are widely used in applications such as wireless sensor networks, building automation, industrial monitoring and control, and automated meter reading. While wireless transmission on the OSBSS platform is still in the beginning stages of development, initial results are promising. The addition of this feature to the OSBSS platform will not only improve functionality but may also serve to reduce costs, as some onboard parts can be removed from the existing platform that will no longer be necessary, such as the microSD card reader/writer for long-term data storage. Storage of data in this case would instead occur only on a central server, which would consist of an OSBSS device that acts as a receiver for all other nodes in a building. On-board storage can also be used for data redundancy, in case of errors on the receiving side.

Acknowledgement

This work was funded by a grant from the Alfred P. Sloan Foundation's Program on the Microbiology of the Built Environment (Grant No. 2013-10-07). We would like to extend a special thanks to those who have helped the OSBSS team, including Boyang Dong, Joseph Chee Poh Huan, and Torkan Fazli. We would also like to thank the OSBSS Advisory Board members for their valuable input throughout the development and testing processes.

References

[1] O. Guerra-Santin, C.A. Tweed, In-use monitoring of buildings: An overview of data collection methods, Energy Build. 93 (Apr. 2015) 189–207.
[2] T. Babaei, H. Abdi, C.P. Lim, S. Nahavandi, A study and a directory of energy consumption data sets of buildings, Energy Build. 94 (May 2015) 91–99.
[3] J. Langevin, J. Wen, P.L. Gurian, Simulating the human-building interaction: Development and validation of an agent-based model of office occupant behaviors, Build. Environ. 88 (Jun. 2015) 27–45.
[4] T. Ramos, S. Dedesko, J.A. Siegel, J.A. Gilbert, B. Stephens, Spatial and Temporal Variations in Indoor Environmental Conditions, Human Occupancy, and Operational Characteristics in a New Hospital Building, PLoS ONE 10 (3) (Mar. 2015) p. e0118207.
[5] D. Heinzerling, S. Schiavon, T. Webster, E. Arens, Indoor environmental quality assessment models: A literature review and a proposed weighting and classification scheme, Build. Environ. 70 (Dec. 2013) 210–222.
[6] T. Ramos, B. Stephens, Tools to improve built environment data collection for indoor microbial ecology investigations, Build. Environ. 81 (Nov. 2014) 243–257.
[7] F. Xiao, C. Fan, Data mining in building automation system for improving building operational performance, Energy Build. 75 (Jun. 2014) 109–118.
[8] W.L. Tse, W.L. Chan, A distributed sensor network for measurement of human thermal comfort feelings, Sens. Actuators Phys. 144 (2) (Jun. 2008) 394–402.
[9] W.-S. Jang, W.M. Healy, M.J. Skibniewski, Wireless sensor networks as part of a web-based building environmental monitoring system, Autom. Constr. 17 (6) (Aug. 2008) 729–736.
[10] International Organization for Standardization, BS EN ISO 7726:2001 Ergonomics of the Thermal Environment. Instruments for Measuring Physical Quantities, 2001st ed., International Organization for Standardization, 2001.
[11] A.S. Ali, Open Source Building Science Sensors (OSBSS): a Low-cost Arduino-based Platform for Long-term Data Collection in Indoor Environments, Master's thesis, Illinois Institute of Technology, Chicago, IL, 2015.
[12] N. Barroca, L.M. Borges, F.J. Velez, F. Monteiro, M. Górski, J. Castro-Gomes, Wireless sensor networks for temperature and humidity monitoring within concrete structures, Constr. Build. Mater. 40 (Mar. 2013) 1156–1166.
[13] K.-Y. Lian, S.-J. Hsiao, W.-T. Sung, Intelligent multi-sensor control system based on innovative technology integration via ZigBee and Wi-Fi networks, J. Netw. Comput. Appl. 36 (2) (Mar. 2013) 756–767.
[14] Z. Nagy, D. Rossi, C. Hersberger, S.D. Irigoyen, C. Miller, A. Schlueter, Balancing envelope and heating system parameters for zero emissions retrofit using building sensor data, Appl. Energy 131 (Oct. 2014) 56–66.
[15] "greiman/SdFat," GitHub. [Online]. Available: https://github.com/greiman/SdFat. [Accessed: 11-Dec-2015].
[16] A. M. Holberg and A. Saetre, Innovative Techniques for Extremely Low Power Consumption with 8-bit Microcontrollers, Atmel Corporation, Feb-2006.
[17] Benchmarks: RealTime Clocks - DS3231/PCF8563/MCP79400/DS1307, SwitchDoc Labs.
[18] M. Jiménez, R. Palomera, I. Couvertier, Introduction to Embedded Systems: Using Microcontrollers and the MSP430, Springer Science & Business Media, 2013.
[19] James A. Barnes, The Measurement of Linear Frequency Drift in Oscillators, in: Proc. 15th Annu. PTTI Meet. 1983, 1983.
[20] M. Broussely, P. Biensan, F. Bonhomme, P. Blanchard, S. Herreyre, K. Nechev, R.J. Staniewicz, Main aging mechanisms in Li ion batteries, J. Power Sources 146 (1–2) (Aug. 2005) 90–96.
[21] J. Yi, U.S. Kim, C.B. Shin, T. Han, S. Park, Modeling the temperature dependence of the discharge behavior of a lithium-ion battery in low environmental temperature, J. P*ower Sources* 244 (Dec. 2013) 143–148.
[22] G. Zhu, K. Wen, W. Lv, X. Zhou, Y. Liang, F. Yang, Z. Chen, M. Zou, J. Li, Y. Zhang, W. He, Materials insights into low-temperature performances of lithium-ion batteries, J. P*ower Sources* 300 (Dec. 2015) 29–40.
[23] D. Ilić, J. Butorac, L. Ferković, Temperature measurements by means of NTC resistors and a two-parameter approximation curve, Measurement 41 (3) (Apr. 2008) 294–299.
[24] S. Wang, J. Burnett, H. Chong, Experimental Validation of CO2-Based Occupancy Detection for Demand-Controlled Ventilation, Indoor Built Environ. 8 (6) (Nov. 1999) 377–391.
[25] A.K. Persily, Evaluating building IAQ and ventilation with indoor carbon dioxide, ASHRAE Trans. 103 (2) (1997) 1–12.
[26] "Product Testing Report: Wall Mounted Carbon Dioxide (CO2) Transmitters - Iowa Energy Center", June 2009.