2018 17th ACM/IEEE International Conference on Information Processing in Sensor Networks

# The Signpost Platform for City-Scale Sensing

Joshua Adkins
University of California, Berkeley

Branden Ghena
University of California, Berkeley

Neal Jackson
University of California, Berkeley

Pat Pannuto
University of California, Berkeley

Samuel Rohrer
University of Michigan

Bradford Campbell
University of Virginia

Prabal Dutta
University of California, Berkeley

## ABSTRACT

City-scale sensing holds the promise of enabling a deeper understanding of our urban environments. However, a city-scale deployment requires physical installation, power management, and communications—all challenging tasks standing between a good idea and a realized one. This indicates the need for a platform that enables easy deployment and experimentation for applications operating at city scale. To address these challenges, we present Signpost, a modular, energy-harvesting platform for city-scale sensing. Signpost simplifies deployment by eliminating the need for connection to wired infrastructure and instead harvesting energy from an integrated solar panel. The platform furnishes the key resources necessary to support multiple, pluggable sensor modules while providing fair, safe, and reliable sharing in the face of dynamic energy constraints. We deploy Signpost with several sensor modules, showing the viability of an energy-harvesting, multi-tenant, sensing system, and evaluate its ability to support sensing applications. We believe Signpost reduces the difficulty inherent in city-scale deployments, enables new experimentation, and provides improved insights into urban health.

**Figure 1:** The Signpost platform easily mounts to existing street sign posts, harvests from an integrated 0.1 $m^2$ solar panel, and provides tenant sensor modules with power, communications, processing, storage, time, and location. Signpost is open source, with all hardware and software available online.†

## 1 INTRODUCTION

Today, more than 50% of the world's population live in urban areas, and the U.N. projects that to increase to 66% by 2050 [57]. With increasing population density, there is growing interest in making cities safer, cleaner, healthier, more sustainable, more responsive, and more efficient—in a word, smarter. Supporting this interest are numerous funding opportunities [20, 42, 49], interested cities [15, 16, 32], and active research projects [13, 14, 27, 41], all targeting new technology to enable smarter cities. And for good reason: applications such as pedestrian route planning based on air quality, noise pollution monitoring, and automatic emergency response alerts can all improve the quality of life for a city's inhabitants.

However, we believe that the difficulty of deploying existing smart city technology and applications is impeding progress. Deployments are rooted in single-purpose hardware, necessitating redesigns to support upgraded sensors or revised goals. Moreover, each system requires a re-implementation of standard resources such as power, communications, and storage, taking developer time away from the core application. Deploying sensors is difficult too, with the reliance on energy from wired mains constraining installation locations. These problems limit not only production-ready technology, but also make it particularly challenging to perform short-term, exploratory research, speaking to the need for a platform that will lower the barrier to entry.

To address these challenges, we present Signpost, a modular, energy-harvesting platform enabling deployable city-scale sensing applications. It mounts to pervasive sign posts (Figure 1) and harvests energy from a vertically mounted solar panel. To reduce the burden of developing new applications, Signpost provides commonly required services including power, communications, processing, storage, time, and location. The platform is modular, with eight pluggable slots for sensors, processors, and radios, facilitating modifications and upgrades to the system. To enable shared deployments, Signpost is multi-tenant, supporting multiple applications simultaneously and enforcing isolation between them.

Key to Signpost's deployability is its energy-harvesting, modular architecture. Harvesting energy enables the system to sever ties to wired infrastructure. This in turn opens up an increased selection of deployment locations, allowing for more granular deployments. Harvesting also enables short-term, pop-up deployments to drive application development and experimentation. Support for modularity allows the sensors on the platform to be changed to suit

For questions, email adkins@berkeley.edu

† https://github.com/lab11/signpost

0-7695-6377-5/18/$31.00 ©2018 IEEE
DOI 10.1109/IPSN.2018.00047

application needs. More fundamentally, however, modularity permits Signpost to take advantage of future technology improvements, improving its capabilities over time.

An energy-harvesting, multi-tenant platform faces challenges that do not exist for mains-powered, single-purpose systems. For one, eliminating the connection to mains power limits the energy available for sensing. We assess the expected solar energy throughout the US, finding that a module can expect an average power of at least 120–210 mW for 50% of weeks. We also provide APIs allowing software to adapt to existing energy, reducing functionality in times of famine and opportunistically increasing it when possible. Another challenge is managing and sharing platform resources to support multiple stakeholders with unaligned interests. We explore the hardware and software requirements for measuring usage and enforcing isolation, describing guarantees necessary for sharing Signpost's limited energy budget between applications.

We envision a testbed of Signposts supporting short-term experimentation by many users. Signposts have been deployed on the University of California, Berkeley campus for six months. The ongoing deployment monitors weather, senses TV whitespace spectrum usage, and observes vehicular traffic. We have found Signpost modules are generally easy to create and the software API is simple to implement on commonly used software and hardware platforms such as Arduino and ARM Mbed. To facilitate the creation of new hardware and software to run on Signpost, we have also created desktop development kits capable of emulating deployed behavior. We hope that by providing a platform for city-scale sensing that reduces the barriers to deployable applications, supporting that platform with development tools and accessible interfaces, and working with the community to realize their sensing needs, we can gain deeper insight into the workings of urban areas and enable higher-level applications that impact policy and quality of life throughout a city.

## 2 RELATED WORK

Existing work in urban sensing generally falls into three categories: static deployments of sensing applications, mobile or human-based participatory sensing, and—most similarly to Signpost—deployments of generic sensing infrastructure. The first two categories are particularly insightful as a guide to which services are frequently needed by existing applications, which we summarize in Table 1.

Examples of static deployments include acoustic sensors to monitor, characterize, and localize different sounds [22, 27, 41], particulate sensors to monitor air quality [14], and electromagnetic, radiological, and meteorological sensing to track people [31] and cars [1], measure road conditions [10, 51], monitor wireless traffic [50], locate point sources of radiation [48], and identify severe weather in urban environments [8]. Most deployments are not long-term and are only deployed for the purposes of evaluation. Additionally, almost all of these deployments depend on either mains power or a battery for an energy source, motivated by the desire for rapid prototyping. Many of these deployments use a proof-of-concept node design built mostly with off-the-shelf components, without much consideration for optimized energy consumption [10, 22, 27, 31, 41, 51]. By providing a platform that already handles energy-harvesting, Signpost could provide sustainability

| Deployment | Energy | Network | Processing | Storage | Time | Sync | Location |
|---|---|---|---|---|---|---|---|
| Caraoke [1] | ✓ | ✓ | | | ✓ | | |
| Bouillet et al. [10] | ✓ | ✓ | | | | | |
| AirCloud [14] | ✓ | ✓ | | | | | |
| Girod et al. [22] | ✓ | ✓ | ✓ | | | ✓ | ✓ |
| Lédeczi et al. [27] | ✓ | ✓ | ✓ | | | ✓ | ✓ |
| SenseFlow [31] | ✓ | ✓ | | | | | |
| Argos [50] | ✓ | ✓ | | | ✓ | | |
| SONYC [41] | ✓ | ✓ | ✓ | ✓ | | | |
| Kyun Queue [51] | ✓ | ✓ | | ✓ | ✓ | | |
| Micronet [24] | ✓ | ✓ | | ✓ | | | |
| Seaglass [43] | ✓ | ✓ | | ✓ | ✓ | | ✓ |

**Table 1: Services required by existing applications.** Time is millisecond-accurate as provided by services like NTP, while Sync is microsecond-accurate as provided by GPS. Location is GPS-level accurate coordinates. These represent the minimum services a platform should provide to support existing applications and simplify the creation of new ones. Many of these applications could run on Signpost without significant modifications.

to these deployments. Further, based on reported power numbers, with the exception of the high power Micronet nodes [8, 24], many of these applications and experiments could run on the Signpost platform without significant redesign.

The majority of work that targets urban sensing uses participatory methods, in which users participate with mobile phones and other handheld devices [11, 12], or vehicles are outfitted with various sensors [23, 28]. These methods use existing mobile resources to collect similar data to static deployments. Many have paired mobile phones with handheld air quality monitors [7, 14, 17, 18], or used phones to directly meter noise pollution [9, 35, 47] or traffic conditions [38, 54, 61]. Similar to participatory sensing methods, vehicular sensor networks monitor air quality, traffic, and road conditions [17, 19, 23, 29, 36], and even detect rogue cellular base stations [43]. These types of deployments often scale very well as the mobility of the devices allows a few sensors to reach a much larger area. However, incentivizing participation can be difficult and coverage can be unpredictable and potentially insufficient.

Finally, several platforms provide generic sensing infrastructure, suitable for many types of smart city applications. CitySense proposes an open, city-scale wireless networking and sensor testbed [40]. It utilizes mains-powered, street pole mounted embedded Linux nodes with 802.11 mesh networking and enables in-situ node programming by end users. Argos, a passive wireless mapping application, builds on a 26 node CitySense deployment [50]. Unfortunately, the CitySense architecture met many logistical challenges that ultimately limited a scaled deployment [59]. The Array of Things project utilizes a network of sensor nodes distributed throughout Chicago to gather environmental data including light, temperature, humidity, and air quality [13]. Like CitySense, Array of Things sensor nodes assume wired power and networking, and thus must be installed in locations where these resources are present. Signpost also provides an open testbed for smart city research. However, through its focus on deployability and modularity, Signpost reaches a different design point than these projects, resulting in a resource-constrained, energy-harvesting, and multi-tenant platform that is more easily deployed, but potentially more challenging to program.

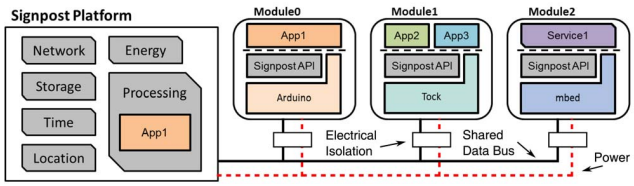



**Figure 2: Signpost platform overview.** Signpost monitors and distributes energy to connected modules and provides shared networking, Linux processing, storage, time, and location services. Modules implement one or more sensing modalities and utilize many possible software stacks, running one or more applications or even providing additional services to the platform. Applications can potentially be distributed across the platform and modules. This platform design supports development and deployment of urban sensing applications.

## 3 PLATFORM OVERVIEW

In the following sections, we present the design, implementation, and evaluation of Signpost, a modular, solar energy-harvesting, sensing platform. In the Signpost platform, sensor hardware connects to a shared backplane via a standard electrical and mechanical interface, enabling modularity. The backplane serves as the module interconnect and has the ability to electrically isolate each module, allowing energy use of any particular module to be limited. To support these sensor modules, the platform harvests solar energy, monitors a shared battery, and distributes metered power. It provides multiple radio interfaces for different communication patterns and shares them among the modules. Other services are implemented as well, including time and location, data storage, and compute offload using a Linux-class co-processor, and these services can be accessed by modules through a standard software API. Resources and modules are orchestrated by a microcontroller-based system controller that oversees the operation of the Signpost platform. All of these components are housed in a waterproof aluminum case that to bolts to a standard street sign post for easy deployment. Figure 2 shows an overview of the platform.

## 4 DESIGN

The Signpost platform's design is guided by four high-level goals:

- **Deployability** is the primary concern of the platform and is key to enabling larger and more frequent deployments, and ultimately wider adoption by the community.
- **Accessibility** reduces burden for developers, thus the platform needs to provide services that meet common application needs.
- **Modularity** allows developers to modify and extend sensing capabilities to support new applications and upgrade modules as technology improves.
- **Multi-tenancy** enables the platform to simultaneously host mutually-untrusting applications created by multiple stakeholders, reducing deployment burden and the cost of experimentation.

### 4.1 Deployability

Deployability is the primary concern for the Signpost platform. Many urban sensing applications require fine-grained sensing, which is not possible for platforms that can only be deployed with easy access to mains power or wired networking. Additionally, to support ad-hoc experimentation, the platform needs to be easily installed, removed, and moved. A deployment made today may not meet the sensing needs of an application tomorrow.

In order to enable deployability, Signpost does not depend on mains power or wired networks. Relying on wired infrastructure would limit Signpost deployments to locations with grid access, such as the top of streetlight poles, and would require costly and time-consuming installation by city utility workers. To support easy physical installation, the platform attaches to existing infrastructure found ubiquitously in urban areas—sign posts.

Making these deployability decisions allows Signpost to better support some applications while restricting others, particularly applications with high power sensors, significant bandwidth needs, or heavy computation. To address these concerns, the platform needs to provide software primitives that enable applications to adapt to available energy and bandwidth. Even if these primitives prove insufficient, we believe that in time most applications will still become possible on Signpost due to the rapid power scaling of embedded hardware. In the last decade alone, best-in-class microcontroller active current has decreased from 220 µA/MHz to 10 µA/MHz [4, 53], radio transmission power has reduced by 3-5x [37, 44], and many sensors have followed similar trajectories. By embracing modularity, hardware can be updated to capitalize on these improvements, with the tradeoff between deployability and resource constraints increasingly favoring the Signpost architecture.

### 4.2 Accessibility

Informed by a review of prior sensing projects in Section 2, Signpost provides several services to support accessibility and reduce the burden for application developers.

*4.2.1 Energy.* Since wired mains power is not an option for Signpost, we turn to batteries and energy harvesting to power the system. Batteries alone may be sufficient for short-term research deployments, but replacement is not scalable for geographically distributed deployments. Instead, a battery would need to store enough energy for the entire deployment duration. Assuming a 1 cm thick Li-ion battery the size of the Signpost solar panel (0.096 $m^2$) yields a storage capacity of 576 Wh [26]. For one year of lifetime, this would result in an average platform power budget of 66 mW.

The expected budget can be improved significantly with the addition of solar energy harvesting. An optimally oriented, 17% efficient solar panel with the same area as Signpost's would generate 2.4 W on average indefinitely in Seattle, a city with notably poor solar conditions [45]. Even with vertical panel placement and suboptimal panel orientation, the addition of energy harvesting yields an increase in energy provided to the platform as we demonstrate in Section 6.2.2, resulting in increased application capabilities.

*4.2.2 Communications.* Signpost needs to support periodic data transmissions, firmware updates, and occasional bulk data uploads. Coverage is needed over a wide area and neither wired network nor WiFi access points can be expected to be accessible for all deployed Signposts. One solution to these problems is cellular radios, especially the machine-to-machine focused LTE Cat-1, LTE-M, or NB-IoT networks. Cellular networks provide high throughput and

good coverage, but also come with costs, both in terms of high power draw and network usage fees.

Alternative solutions include low-power, wide-area networks such as LoRaWAN [34], which provides data transfer at rates of 1-20 kbps with a range of several kilometers and power draw significantly lower than cellular radios. LoRaWAN networks can be deployed by end users, allowing a network to be set up to support a Signpost deployment. However, LoRaWAN predominately supports uplink communications, making firmware updates and other downlink-focused applications more difficult.

Finally, local communication facilitates interactions between a Signpost and any nearby residents or users of the platform. Communication protocols such as Bluetooth Low Energy would enable the platform to interact directly with nearby smartphones.

*4.2.3 Processing.* In nearly any sensing system, data must be processed, batched, transformed, and analyzed, and in the face of energy constraints, local computation is preferable over transferring all data to the cloud. Providing a processing service is not necessarily just about computational capability. A familiar processing environment in which developers can use familiar languages and libraries lowers the barrier to entry for domain scientists.

Many existing urban sensing platforms provide processing by using some variation of a Linux computer as their primary processor [13, 40, 41, 48, 50]. For an energy-constrained system, however, supporting an always-on Linux computer is problematic. Even the lowest power Linux compute modules we survey draw 200-500 mW when active [25]. One compromise is to use a Linux environment not as a core controller, but as a co-processor, employed occasionally to process batched data. This allows developers to use languages and libraries to which they are accustomed, but requires them to split applications between two execution environments.

*4.2.4 Storage.* With low power and low cost flash memory widely available, data storage could be a module-supplied resource. However, we argue it should be centralized on Signpost for two reasons. First, a central data store aids manual data collection (likely over a short-range wireless link). This is useful for collecting high-fidelity data from multiple modules, particularly in the early experimentation phases of a deployment. Second, co-locating the central storage with shared processing resources allows for fast and easy access to batched data.

*4.2.5 Time and Location.* Synchronizing clocks throughout a sensor network deployment is critical to many applications [52]. Providing the capability to synchronize within 100 ns allows a group of Signposts to achieve localization within 30 m for RF signals and less than one meter for audio signals. In addition to just synchronization, the ability to timestamp data and understand the local time of day and year is useful for adapting operation (for example, slowing sampling before night) or predicting available solar harvesting energy. Location also provides automatic installation metadata and enables localization-based applications, such as gunshot detection. Fortunately, all are easily provided by GPS modules, although some care needs to be taken when expecting GPS use in dense city environments where fewer satellites may be in line-of-sight of the receiver. The addition of a stable and low power real-time clock can act as an optimization for a time and location system on a stationary platform by allowing the GPS to be predominantly disabled. This reduces system power draw while maintaining sufficient accuracy for many applications.

## 4.3 Modularity

Modularity enables not only specialization, but it also allows the platform to be upgraded over time, adapting to technology improvements for sensor modules and platform resources alike. Supporting modularity requires standardized electrical and mechanical interfaces to allow sensor modules to be installed and replaced as needed. The electrical interface should be simple but sufficient, including connections to power and an internal communication bus over which modules access platform services. Other signals can be added to support performance, for example a time synchronization signal, but such additions should be kept to a minimum to keep module creation simple.

Regarding mechanical considerations, the interface must allow for a robust connection to the physical platform without significantly limiting sensing capability. Weatherproofing plays an important part in the design of this interface since Signpost will be deployed outdoors, as does physical security since platforms will be unattended for long periods. Additionally, sensor module developers should be able to easily tailor the module enclosure to support the physical and environmental requirements of their sensors.

## 4.4 Multi-tenancy

Finally, Signpost is designed to support multiple stakeholders simultaneously, allowing a single hardware deployment to act as a testbed for multiple applications. Support for multi-tenancy requires fair sharing of resources between applications. For most system services, this reduces to platform software recording usage and implementing some fairness policy.

Sharing energy is a more complex problem and the top priority of a multi-tenant, energy-harvesting system [3]. The power requirements of one application should not limit the capabilities of another. To support this, a platform must first be able to accurately measure and control access to energy. This involves metering not just modules, but also system resources, so that their energy draw may be charged against the application which accessed them.

Second, the platform must use these measurements to implement an energy policy. In the presence of variability, applications need guarantees of energy availability to reason about future processing capabilities. There is one important guarantee: the energy allocated to an application must only decrease in a predictable fashion. It can be spent directly by the application, indirectly by a service the application uses, or taken regularly as a platform tax, but it must not decrease in a manner unpredictable to the application. Particularly, energy should never be taken to support other applications (although it could be given). If energy is harvested by the platform, the allocation of a particular application may increase, but having a minimum known energy to rely on allows applications to plan for future actions. Support for energy isolation has been explored in prior work [3].

Features to support multi-tenancy have an added benefit in supporting overall system reliability. Modules can be isolated from the platform entirely if a hardware or software failure occurs.

## 5 IMPLEMENTATION

The Signpost architecture is shown in Figure 3. The Signpost platform is defined by the Power Module, Control Module, Backplane, and Radio Module. Additional modules connect via a standard electrical and mechanical interface. A full Signpost has six general-purpose module slots, one of which is taken by the Radio Module, leaving five for sensing capabilities. The size of the entire system, including a case, is 42.9 cm high, 30.0 cm wide, and 8.4 cm thick. For comparison, the minimum size of a speed limit sign in the United States is 91 cm by 61 cm [21].

### 5.1 Backplane

The Backplane is the backbone of the Signpost. It has physical and electrical connections for modules, signal routing between modules, and isolation hardware. The Backplane has eight slots in which modules can be connected. Two are special-purpose, corresponding to dedicated signals for the Power Module and Control Module. The remaining six are standard interfaces for modules. The interface provides power at 5 V, access to a shared $I^2C$ bus, two dedicated I/O lines to the Control Module, a Pulse Per Second (PPS) signal for synchronization, and a USB slave connection.

Modules are not required to implement all signals in this interface. However, we expect that most modules will use the $I^2C$ bus and dedicated I/O signals, and that some complex modules will implement USB or PPS support.

All module connections can be individually isolated, along with buffering for $I^2C$ connections. These isolators can be activated by the Control Module and prevent individual modules from negatively impacting the rest of the Signpost. The Backplane also accepts a voltage reference signal from each module and handles translation of voltage levels for all signals except USB, allowing modules to perform I/O at any voltage between 1.65 V and 5 V.

### 5.2 Power Module

The Power Module is responsible for energy harvesting, management, monitoring, and distribution on the Signpost platform. Energy is harvested from a Voltaic Systems 17 W solar panel, a 37 cm by 26 cm panel with an expected 17% efficiency. The solar panel output is monitored by a coulomb counter, and regulated by a maximum power point tracking battery charger. Excess energy is stored in a custom 100 Wh Li-ion battery pack.

System energy is further regulated for consumption before being distributed to the Backplane and modules. Each regulator can provide a constant 1.5 A, and is protected from shorts by a load switch. Each module's power rail is monitored by a coulomb counter that also provides instantaneous current readings, supporting energy accounting.

The Power Module also includes a hardware watchdog that monitors the platform. This further increases Signpost reliability by providing a redundant watchdog in the event of software failures.

### 5.3 Control Module

The Control Module handles system tasks, such as managing the module energy usage, assigning module addresses, and monitoring system faults. It also provides time, location, storage, and processing

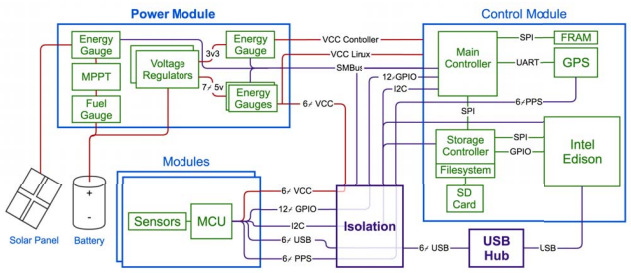



**Figure 3: Signpost architecture.** The Power Module is capable of harvesting energy from a solar panel, storing energy in a battery, supplying power at the correct voltage to modules, and monitoring the energy use of modules. The Control Module provides storage, time and location, and Linux processing services, and also monitors modules with the capability of isolating them from the system if necessary. Finally, there are the modules themselves, with many possible capabilities. This architecture allows for modular and extensible sensing while minimizing deployment complexity.

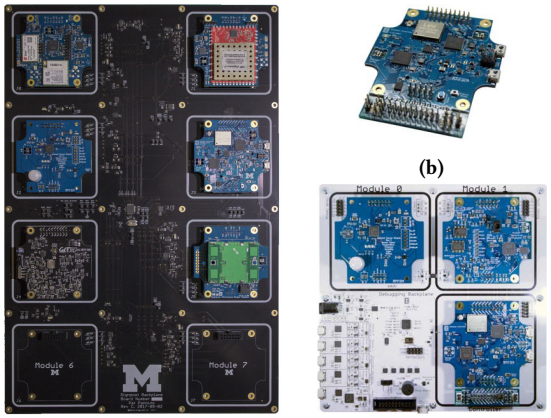
services to the sensor modules. Computation is handled by two ARM Cortex-M4 microcontrollers.

One microcontroller is responsible for isolation, managing the GPS, and accounting for module energy. It can also communicate with sensor modules on the shared $I^2C$ bus and through dedicated per-module I/O signals, sending information such as location and time to the sensor modules in response to Signpost API calls. A globally synchronized Pulse Per Second signal is routed from the GPS to all sensor modules. The second microcontroller is responsible for managing an SD card and providing the storage API to the sensor modules. Each of these subsystems is power gated and can be entirely disabled to save energy.

Finally, the Control Module has an Intel Edison Linux compute module for higher performance processing capabilities. Contrary to common system design, while the Edison is the most capable computer on the Signpost, it is not in control of the system. Instead, the Edison is a coprocessor, capable of batch processing and using languages and libraries that are difficult or impractical to port to embedded microcontrollers. The Intel Edison connects directly to modules over USB, with each module playing the role of a USB slave device. It can also communicate with modules over an internal SPI bus by using one of the Cortex-M4s on the Control Module to forward messages to the shared $I^2C$ bus. The power usage of the Edison is individually monitored, allowing its energy to be attributed to the module utilizing its services.

### 5.4 Radio Module

The Radio Module provides communications services to the Signpost. To handle diverse communication needs, it hosts cellular, LoRa, and BLE radios. An ARM Cortex-M4 microcontroller handles receiving messages through the shared $I^2C$ bus or via USB from the Intel Edison and sending them to the appropriate radio interface. A U-blox SARA-U260 cellular radio is capable of both 2G and 3G operation at up to 7.2 Mb/s. However, it draws up to 2.5 W in its highest throughput modes [56]. A Multitech xDot radio module provides LoRaWAN communications. Sending data through LoRaWAN




(a) (b) (c)

**Figure 4: A populated Backplane (a), Control Module (b) and Development Backplane (c).** The Backplane serves as the Signpost interconnect, while the smaller Development Backplane is the desktop equivalent, enabling easy module and application creation and testing. The Control Module manages Signpost energy and provides services to sensor modules. Existing sensor modules are also shown, with the RF spectrum and radar modules at the top and bottom right of the populated Backplane respectively, and the environmental and audio sensing modules on the top left and top right of the Development Backplane.

is more sustainable from an energy budget standpoint, with the module drawing less than 0.5 W in its highest power state [39]. Finally, the Radio Module includes an nRF51822 BLE SoC. This enables Signpost to send real-time data about the environment to nearby smartphones. Providing three communications interfaces allows Signpost to make decisions about which radio to use based on quality of service, latency, throughput, and energy requirements.

## 5.5 Sensor Modules

Four sensor modules have been created for Signpost and are in use. The existing modules perform ambient environmental sensing (temperature, humidity, pressure, and light), monitor energy in seven audio frequency bins ranging from 63 Hz to 16 kHz, measure RF spectrum usage within 15 MHz to 2.7 GHz, and detect motion within 20 m with a microwave radar. Each was made by a different student, including two undergraduates. All of the sensor modules and the Signpost Backplane are shown in Figure 4.

## 5.6 Module Software

To enable access to the resources on Signpost, we provide APIs for applications that abstract away the specific details of messages sent over the shared $I^2C$ bus and allow module creators to write software at a higher level. Abstract versions of several API calls are listed in Table 2, including calls to allow module applications to POST data, write to an append-only log, be automatically duty-cycled, start processes on the Intel Edison, and send messages to other modules.

All API calls are layered on a minimal intra-Signpost network protocol. The library code is written in C on top of a hardware abstraction layer requiring $I^2C$ master, $I^2C$ slave, and GPIO implementations. We implement the library using the Tock operating system [30] for our own development purposes and have ported the library to the Arduino [5] and ARM Mbed [6] stacks to support a wider array of module designs.

| Service | System Call | Description |
|---|---|---|
| **Init** | `i2c_address = module_init(api_handles)` | Initialize module |
| **Network** | `response = network_post(url, request)` | HTTP POST data to URL |
| | `network_advertise(buf, len)` | Advertise data over BLE |
| | `network_send_bytes(destination, buf, len)` | Send via best available medium |
| **Storage** | `record = storage_write(buf, len)` | Store data |
| **Energy** | `energy_info = energy_query( )` | Request module energy use |
| | `energy_set_warning(threshold, callback)` | Receive energy usage warning |
| | `energy_set_duty_cycle(duty_cycle)` | Request duty cycling of module |
| **Processing** | `processing_call_rpc(path, buf, len, callback)` | Run code on Linux compute |
| **Messaging** | `messaging_subscribe(callback)` | Receive message from a module |
| | `messaging_send(module_id, buf, len)` | Send message to another module |
| **Time** | `time_info = get_time( )` | Request current time and date |
| | `time_info = get_time_of_next_pps( )` | Request time at next PPS edge |
| **Location** | `location_info = get_location( )` | Request location |

**Table 2: Signpost API examples.** Abstract versions of several Signpost API calls for each system service are shown. Providing a high-level API enables easier application development.

Signpost supports multiple views on what it means to be an application. A module may run one or more applications, and an application may be constrained to a single module, include processing code run on the Intel Edison, exist logically across several modules connected by the messaging API, or even across Signposts distributed around a city, connected by wireless communications. An example of the Signpost software model is shown in Figure 2 where one or more applications are running on heterogeneous sensor modules and accessing Signpost services through a common API.

## 5.7 Development

In addition to the full, weatherproofed Signpost platform, a development version of the system aids in creating and testing module hardware and software, as shown in Figure 4. The development Signpost supports two modules and a Control Module. While meant to be wall-powered, it has the same isolation and monitoring hardware as a full Signpost, allowing it to emulate various energy states, track module energy use, and disable modules when they exceed their allocation. Rather than including radios, the development Signpost has a microcontroller that implements the radio API, but sends data over a USB serial connection instead of an RF link. Identical Control Module and Backplane hardware is used on both systems, allowing desktop experimentation with applications that is faithful to deployed system.

# 6 EVALUATION

We evaluate the key claims of the Signpost platform, including deployability, the implications of a deployable design on energy availability, and the ability to support multiple applications. We also benchmark several Signpost services. Finally, given these capabilities and constraints, we explore the types of applications capable of running on Signpost and how they interact with system resources.

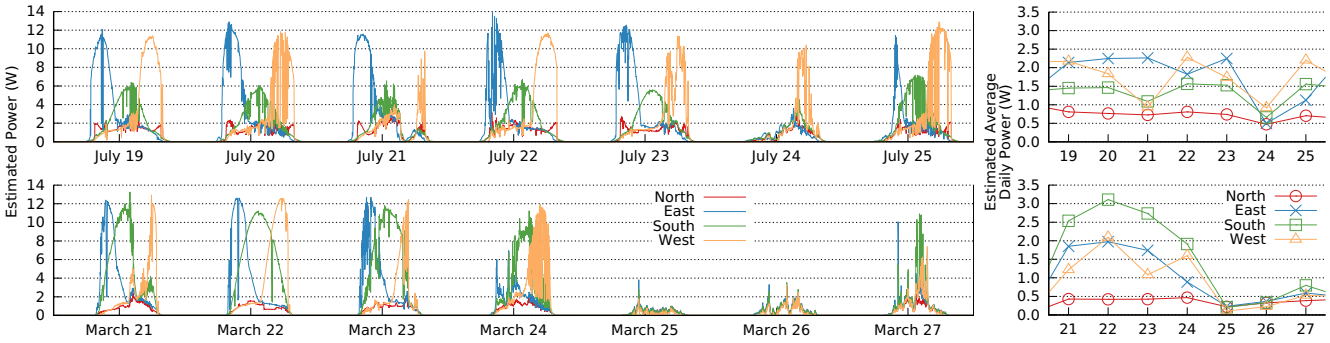



**Figure 5: Solar harvesting in four different cardinal directions and two seasons.** The experiments are run in July 2016 and March 2017 in Ann Arbor, Michigan, with each including periods of both sunny and cloudy days. At left is estimated power generated from solar panels mounted vertically in four cardinal directions captured in 10 second intervals over a week. At right is the average daily power provided by each solar panel. There are large variations in average power both due to direction and daily weather patterns. While some daily variations can be buffered by the battery, Signpost will still experience variability in available energy to which it must adapt.

## 6.1 Deployment Metrics

A primary goal of the Signpost platform is deployability, and over the course of nine months we deploy the platform on over 50 occasions, for varying lengths of time, at several locations. In all of these deployments, we found Signpost to meet our deployability goals in both speed and effort.

Specifically, we find that two students can deploy a single Signpost in less than five minutes. In a specific case, it took less than 90 minutes to walk and deploy twelve Signposts across a portion of the UC Berkeley campus. Although we take no precautions, the deployments have experienced no vandalism or theft, even with Signposts placed near a popular concert venue in an area with relatively high property crime. We believe that this indicates the platform is unobtrusive and blends in with other city infrastructure. Approval for these deployments, while sometimes slow for bureaucratic reasons, has been simple due to the non-destructive, attachment method. While this level of deployability comes at the cost of energy availability, a system with these properties greatly facilitates ad-hoc experiments and highly-granular long term sensing applications.

## 6.2 Signpost Energy

This focus on deployability makes energy availability a fundamental challenge for Signpost. We investigate the overhead of multi-tenancy and expectations for how much energy Signpost can harvest.

*6.2.1 Platform Overhead.* While supporting city-scale sensing is the purpose of Signpost, not all energy goes directly to applications. In particular, multi-tenancy and platform services each incur overhead. These costs can be primarily attributed to the static power of the regulation and monitoring hardware, which have a total quiescent power draw of 13.2 mW. The components for module isolation draw an additional 1 mW, as do the microcontrollers on the Control Module, on average.

Additionally, the over-sized charging and regulation circuitry has a lower efficiency than similar circuitry designed to match the requirements of a single-purpose sensor. We measure the battery charging efficiency to be 85% at a wide range of power inputs, and the regulator efficiency to be 89% at all but the lowest power draws. Across the platform, this totals to 76% efficiency and a base power draw of 16 mW, less than 2% of the 50th percentile average power budget and 6-18% of the 95th percentile budget. We believe this is an acceptable overhead for the advantages of multi-tenancy.

Services provided by the Control Module, such as storage and location, are power gated when not in use and do not contribute to the static power of the platform. If applications request these services, their energy is attributed to the sensor modules using them. We find the Intel Edison Linux module draws 15–24 mW in sleep mode, the GPS chip draws 40 mW when tracking satellites, and the Radio Module sleeps at less than 1 mW. The SD card is enabled on demand, and therefore has no idle power draw.

*6.2.2 Harvesting.* A key enabler of deployability is the shift to a solar energy-harvesting power source. To further increase deployability, it is preferable to make no assumptions about solar panel positioning, and therefore expect the panel to be deployed vertically facing an arbitrary direction. We evaluate the expected energy availability under these constraints in different locations, solar panel directions, and times of year.

We start this evaluation by deploying four solar panels on sign posts in Ann Arbor, Michigan, with one panel pointing in each cardinal direction. A building is located to the south of the posts and a small hill directly west. For each panel, we record the open-circuit voltage and short-circuit current at ten second intervals and estimate the power output of the panels by assuming an 80% fill factor. Figure 5 shows the output of this experiment for one week in July 2016 and one week in March 2017. We present both the instantaneous output of each solar panel and the daily averages.

This experiment shows that the power availability of a Signpost is highly variable, ranging from over 3.08 W for the south facing panel on March, 22nd to only 219 mW for the north facing panel on March, 25th. We find that the direction, season, and degree of cloud cover all contribute to this variability. While some of the variability can be buffered by the battery, variability will inevitably be experienced by applications running on Signpost.

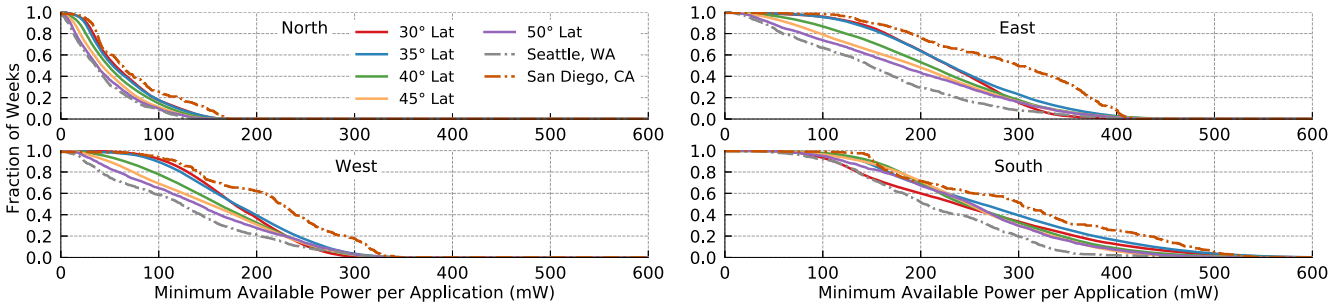



**Figure 6: Fraction of weeks when an application can expect a minimum power income at different latitudes and cardinal directions.** To evaluate how much power a Signpost application can expect under varying deployment conditions, we model the solar harvesting potential of a vertical Signpost facing the four cardinal directions across the United States. We use a standard solar model that accounts for both direct and diffuse light [33] along with hourly irradiance data from the NREL MTS2 2005 dataset [60]. We group these locations by latitude, and also plot distributions for Seattle, Washington and San Diego, California, where local weather patterns create poor and near-ideal solar harvesting conditions, respectively. The per application expected minimum power is calculated by subtracting the static power draw (16 mW) from the weekly average harvested power, dividing among an expected five applications, and multiplying by the regulator efficiency (76%). We find that orientation generally has a stronger influence on harvested energy than latitude or climate.

To more broadly determine the expected power budget for a sensing application running on Signpost, we create an energy availability model that predicts the average weekly power available to Signposts at different geographic locations in the United States throughout the year. The model is based on hourly direct and diffuse light measurements at 1,500 locations around the United States from the NREL MTS2 dataset [60], and these measurements are converted into expected power output using a standard harvesting model for tilted solar panels which takes into account solar panel direction, angle, and the harvestable portion of diffuse light [33]. We compare our model with the experimental data shown in Figure 5 and find the model strictly underestimates our experimental results by an average 3.3% on sunny days and 22% on cloudy days. We believe this error primarily can be attributed to diffuse light collection for north-facing solar panels, a scenario that is not well studied in solar modeling literature.

The results of this model are displayed in Figure 6 as the fraction of weeks at which an application will have a minimum available power. To generate this plot, we group the weekly average power data by latitude, subtract the platform overhead and regulator efficiency losses discussed in Section 6.2.1, then divide by an expected five applications (assuming one for each available module slot on Signpost). In addition to showing data for each latitude, we also plot energy available in Seattle, Washington and San Diego, California, which are particularly poor and ideal solar energy harvesting locations, respectively, in the United States. We see that the 95th percentile of available weekly average power ranges from 3.84 mW per application for a north facing Signpost in Seattle, WA to 147 mW per application for a south facing Signpost in San Diego, CA.

We conclude that, in general, the direction at which Signpost is placed impacts available energy more than the latitude of the platform. This creates a tradeoff between deployability and energy availability. While it is possible to entirely ignore orientation when deploying Signposts, this comes at the cost of expected energy for some of the deployed systems. Putting in care to avoid facing north when possible may be a sufficient compromise.

One aspect which is not included in the prior evaluations is potential shading from nearby obstructions. This is a particularly real concern in urban areas where buildings are expected to obstruct direct sunlight for portions of each day. The amount of shade a Signpost can expect is, however, particularly deployment-specific and difficult to predict in a general fashion. For example, due to its vertical orientation, even with a building directly to its east a west facing Signpost can expect to harvest most of its predicted clear-sky energy. In our deployments, we have found that Signposts deployed under moderate, continuous shade (under a tree in this case) see harvested energy similar to a north facing, clear-sky Signpost.

## 6.3 Managing Multi-tenancy

Signpost expects to host not just a single application, but several. Here, we evaluate how the system responds to multiple demands to its resources simultaneously.

*6.3.1 Energy Isolation.* The primary resource that must be shared between all applications is energy. On Signpost, we virtualize stored energy, making it appear to each application that they have independent batteries. Stored energy in the battery is split into a "virtual allocation" for each application. A virtual allocation is guaranteed to never deplete except when predictably spent. For example, it will never be taken to support another application's needs. This allows programs to plan and make decisions based on available energy that are independent of the actions and needs of others.

On an energy-harvesting platform, an additional question arises in how to distribute incoming energy. A fair model distributes energy equally between applications, but there must be a maximum allocation for each. If an application stores the energy it is given but does not use it, its allocation would eventually expand to the entire capacity of the battery. Instead, we define a maximum capacity for each virtual allocation. Harvested energy is then divided between applications that are below maximum capacity. This adds variability to the amount of energy an application receives based on the actions of other applications running on the platform. However, this variability is no worse than the variability inherent to energy

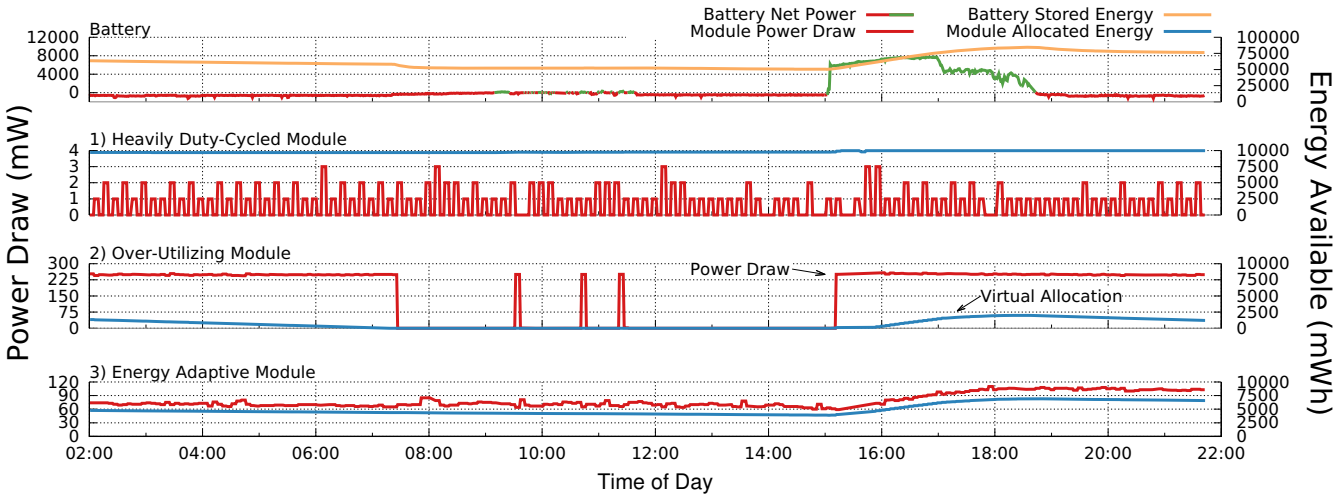



**Figure 7: Energy isolation on Signpost.** Energy allocation and five-minute average power draw are displayed for three simultaneously running applications and the platform as a whole. Each application employs a different strategy for energy use. The first is only active for a brief period every ten minutes, achieving a low average power, and storing up an allocation of energy. The second continuously runs, exhausting its budget, and is disabled by the platform, to be enabled later when energy is available again. The third adapts its actions based on the available energy, running continuously without depleting its allocation. Signpost is capable of balancing the needs of these three applications simultaneously, assigning each a "virtual allocation" of energy it draws from without affecting the operation of the others.

harvesting systems in the first place. Policy choices and support for energy isolation are discussed further in another work [3].

Figure 7 demonstrates energy sharing in practice. Three modules are installed on one Signpost, each running a single application and given virtual allocations with a maximum capacity of 10,000 mWh. Data is shown for a 20 hour period, from night to night. The deployed Signpost has a building directly to the east, only allowing it to harvest later in the day. Displayed are the five-minute average power draws for each application and the net power into the battery. Energy allocations are also reported every five minutes for each module and the battery.

Each application has a different strategy for energy use. The first heavily duty-cycles itself and is active for only a brief period every ten minutes. This results in an average power draw of less than 4 mW, and consequently its virtual allocation stays near or at maximum capacity the entire time. The second application continuously draws 250 mW, an amount that cannot be sustained while the Signpost is receiving no direct sunlight. It eventually exhausts its allocation and is disabled by the platform. Later in the day, when energy is being harvested, it is allocated a portion of incoming energy and resumes operation. The third application adapts to the amount of energy available to it, remaining in continuous operation. Its power draw increases when the solar panel receives direct light, corresponding to an increase in sampling rate in the application. As this experiment demonstrates, Signpost is able to isolate the energy needs of applications from each other.

*6.3.2 Internal Communication.* The Signpost design includes a single, shared, multi-master $I^2C$ network for internal communication, such as requests to platform services. When multiple applications are running simultaneously, this bus can be a source of contention. While the Signpost design expects only a modest utilization of the shared $I^2C$ bus, in practice sensing events can often be correlated and traffic can be bursty. Theoretically the listen-before-talk requirement of $I^2C$ should make the bus achieve nearly 100% reception rates even in these scenarios, however we observe that this feature is not implemented in all TWI/$I^2C$ peripherals. Assuming no carrier sense capability, the $I^2C$ bus resembles the original unslotted ALOHAnet [2], and the target utilization rate should be kept to the 20% proposed by ALOHA. This corresponds to a total traffic of 80 kbps on a 400 kHz $I^2C$ bus, which we believe is sufficient for most sensing applications. Applications that require higher throughput can make use of the optional USB bus.

## 6.4 Microbenchmarks

Several services are important to benchmark due to their impact on the range and performance of Signpost applications.

*6.4.1 Communication Policy.* Signpost provides multiple wireless interfaces. These have an advantage in supporting various communication policies that determine how data should be be transmitted based on quality of service needs and the current energy state of the platform. One simple policy is to primarily use the lower power LoRaWAN radio for data transmission unless the message queue gets too full, which could occur when applications have large amounts of data to transfer or in poor radio conditions when LoRaWAN bandwidth is limited. When the queue gets too full, the cellular radio is activated and all queued messages are transferred quickly. In Figure 8, we demonstrate an example of this policy. Poor communication conditions are emulated by removing the LoRaWAN radio antenna, causing messages to be queued until the cellular radio is activated to dispatch them, resulting in briefly increased power draw.

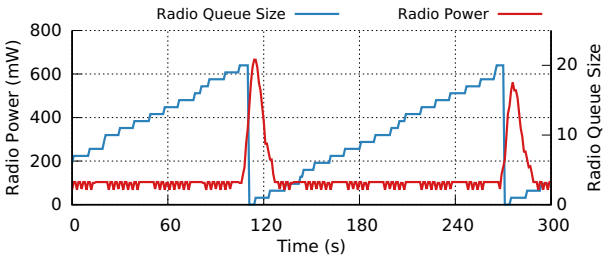



**Figure 8: Communication policy in practice.** The power draw of the Radio Module is shown along with the number of messages queued to be sent. The communication policy is set to automatically transfer data over a cellular connection if the queue reaches twenty messages, as can be seen by the increased power draw. This policy allows the platform to adapt to both increased application requests and poor network conditions by utilizing high-power resources.

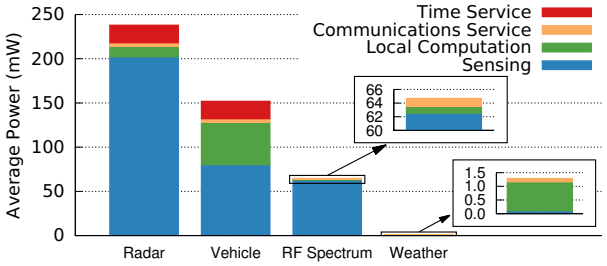



**Figure 9: Resource usage of example applications.** We break apart the major components of usage for example applications into sensing cost, local computation, and network and time service requests. Heavily duty-cycled applications such as the weather monitoring app have nearly inconsequential average power. Applications performing constant sensing with tight timing requirements both draw a higher total power and remit a greater share platform power draw. Applications like spectrum sensing can achieve moderate average power draw even with high instantaneous sensing power using duty cycling. Dynamically adjusting duty cycling allows spectrum sensing to adapt to energy availability.

*6.4.2 Synchronization.* Some applications require coordination between multiple modules on a single Signpost or between multiple Signposts, requiring tight synchronization [52]. On Signpost, a PPS signal is routed to each of the sensor modules from the GPS to provide this synchronization. We find the timing difference across Signposts to be 75 ns in the average case with a 95th percentile metric of 97 ns. We observe little skew in the signal from Control Module to sensor modules (less than 6 ns) and almost no variation from module to module. We expect this synchronization precision to suffice for many applications, providing sufficient resolution for RF localization on the order of tens of meters and sub-meter audio localization.

## 6.5 Applications

Applications run on sensor modules and have access to system resources through physical connections and software APIs. We design several applications (and sensor modules) and deploy them on the

```c
uint8_t send_buf[DATA_SIZE];

void send_samples (void) {
  // Add a timestamp to the data
  time_t time = get_time();
  memcpy(send_buf, time, sizeof(time_t));

  // Send data over network, allowing Signpost to decide how
  network_send_bytes(send_buf, DATA_SIZE);
}

int main (void) {
  // Initialize the module with Signpost
  api_t* handles[] = NULL; // provides no services
  module_init(handles);

  // Collect audio data with an ADC, placing it into send_buf
  adc_continuous_sample(SAMPLE_RATE, &data_ready_callback);

  // Send samples every ten seconds
  timer_every(10000, &send_samples);
}
```

**Figure 10: Example module software.** This software snippet from the vehicular sensing application collects averaged volume data for ten seconds and transmits it using the network API. Timestamps for the collected data are requested from the time API and appended to the data before transmitting it. Access to the Signpost APIs makes applications easier to create.

Berkeley campus for several months. While applications written by users will be different, these examples can inform the types of applications that are possible on Signpost. We describe our applications, the platform resources they use, and some example results. Figure 9 shows the power drawn by different components of the applications, broken down into draw by sensors, local processors, and the communications and time services.

*6.5.1 Weather Monitoring.* The weather monitoring application uses the environmental sensing module to sample temperature, pressure, and humidity every ten minutes, sending it to the cloud via the Signpost network API. After the data reaches the cloud, it is posted to Weather Underground to help support their goal of distributed weather sensing. The application achieves very low power operation even without implementing sleep mode by using the energy API to power off the sensor module between samples.

*6.5.2 Vehicle Counting.* The vehicle counting application runs on the audio sensing module, which provides the volume of audio in seven frequency bins collected up to 100 times per second. This module should in principle allow high-level event recognition (e.g. vehicle detection), without capturing recognizable human speech. The application records these volumes, averages them over a second, and every transmits the results every ten seconds to the cloud using the network API. To properly identify vehicle movement, the application must know the precise time at which a volume sample is taken, so the time API is used to timestamp each batch. The code for this application is shown in Figure 10, and the average power draw and resource usage are shown in Figure 9. The requirement for precise timing information results in the application being charged for a portion of the GPS power. Additionally, the local processor must stay active with a relatively high clock frequency to continually sample and process incoming audio volume data. Once the

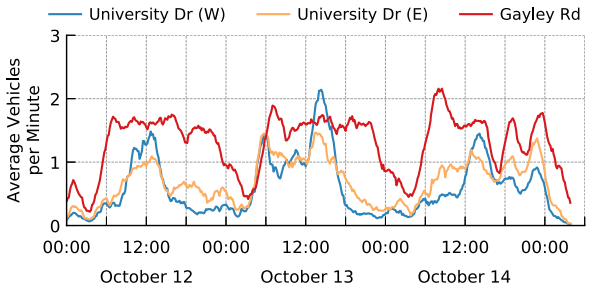



**Figure 11: Vehicle counting application.** Several days of processed audio data are collected in October 2017 for the vehicle counting application. Prominent peaks across several audio frequency bands are used to detect vehicles. We plot estimated vehicles per minute averaged over a one hour time window. The Signposts on University Drive are close, but do not have completely redundant traffic paths. We note that Gayley Road sees traffic much later into the night because it is a through street that routes around campus. Interestingly, all the Signposts experience traffic until around midnight on October 14th, and after further examination, this was due to a concert at a nearby venue. Clear peaks in traffic can be seen before and after the concert, which started at 20:00.

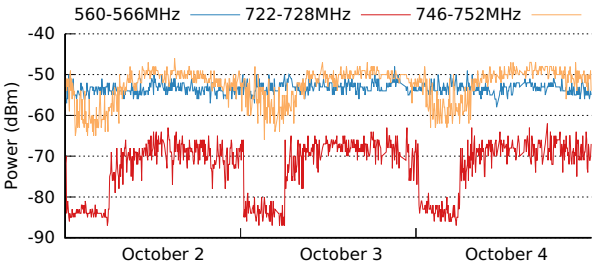



**Figure 12: RF spectrum sensing application.** A sample of RF spectrum data from October 2017 in three frequency bands corresponding to a local TV station (560 MHz), AT&T owned spectrum (722 MHz), and Verizon owned spectrum (746 MHz). Distributed and fined-grained spectrum sensing could help to build better models of RF propagation and inform policy around the reuse of underutilized spectrum. The two higher frequency bands are particularly interesting due to their cyclic nature.

data are in the cloud, it is further processed to look for peaks that are indicative of a moving car. An example of the output of this processing is shown in Figure 11.

*6.5.3 RF Spectrum Sensing.* The white space sensing application runs on the RF spectrum module and periodically samples the energy on each of the TV white space channels (every 6 MHz from 470-830 MHz). For thirty seconds, the spectrum analyzer reads the energy on these channels and computes the min, max, mean, and standard deviation for each channel. The application then sends this data with the Signpost network API and uses the energy API to power off. While the duration for power off is currently set to three minutes, it could be adapted to available energy without significantly degrading the utility of the application.

Three days of this data are shown for several interesting channels in Figure 12. While our RF spectrum module does not yet meet the FCC requirements for a white space utilization sensor, collecting distributed RF spectrum data can be used to inform RF propagation models and inform policy about the reuse of underutilized spectrum.

## 7 DISCUSSION

Signpost is under active development. Here we discuss on-going issues along with future work for the platform.

### 7.1 Cost

Currently Signpost costs roughly $2,000 to produce in quantities of ten, including all parts and labor for a Signpost platform and a typical set of sensor modules. We expect the price to drop significantly at higher quantities, and we plan to explore optimizations to further reduce cost. For context, including labor, a street sign and post costs $250, a solar powered electric speed limit sign costs $3,000, yearly maintenance costs for a stop light are $8,000, and a new stop light costs over $250,000 [46, 55, 58]. This puts Signpost on par with other city infrastructure.

### 7.2 Community Building

To realize the benefits of Signpost modularity, domain experts must be motivated to leverage the platform. To achieve developer buy-in, we believe we need to create a suite of tools that help people develop and test both hardware and software at their desk, then allow them to deploy it on existing Signposts. We have started with the Development Backplane described in Section 5, and we plan to continue to grow the ecosystem around it. Additionally, we have ported the Signpost software API to platforms such as Arduino and Mbed which are more accessible to non-experts. All of our hardware and software is open source to encourage the creation of a community around the Signpost platform. Software, hardware and documentation for Signpost can be found at github.com/lab11/signpost.

### 7.3 Security and Privacy

The pervasive deployment of sensors throughout a city creates significant privacy concerns. While our sensors cannot collect personally identifiable information, the platform could enable the collection of this data if care is not taken. This problem must be addressed through both policy and practice. Policy should dictate that modules are incapable of collecting private information and that applications do not attempt to collect or transmit sensitive data. In practice, this requires some manual oversight into the module creation and deployment process and that all software updates to Signpost be authenticated to ensure they originate from the proper source. Additionally, the data collected from Signposts must be authenticated to ensure its validity, especially if it will directly influence city infrastructure. However, the authentication of large numbers of low-power sensors in a collaborative deployment is an area of active research.

## 8 CONCLUSIONS

In this paper, we introduce Signpost, a solar energy-harvesting modular platform designed to enable city-scale deployments. By providing energy, communications, storage, processing, time, and location services, Signpost allows developers to focus on the sensing application they care about rather than the engineering details of making it deployable. The platform is designed with adaptivity in mind, giving applications the tools to adjust to varying energy and communications availability.

By making the Signpost platform widely available, we hope to begin a new era of urban sensing. We envision a future where city-scale experimentation is simple and city-scale deployments are pervasive. This in turn will open new areas of research exploring energy constrained, geographically distributed applications, encouraging the development of more capable sensors, and providing a deeper understanding of our increasingly urban world.

## 9 ACKNOWLEDGMENTS

This work was supported in part by the CONIX Research Center, one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA, and in part by Terraswarm, an SRC program sponsored by MARCO and DARPA. Additionally, this material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under grant numbers DGE-1256260 and DGE-1106400, NSF/Intel CPS Security under grant 1505684, and generous gifts from Intel.

We would also like to thank our anonymous reviewers for their insightful feedback, our shepherd Neal Patwari, and the many collaborators that helped make this work possible. Specifically Amit Levy and the Tock development team for their support in using an under-development operating system, Anthony Rowe, Craig Hesling, and Artur Balanuta for LoRaWAN gateway hardware and support, William Huang for designing the environmental sensing module, Yifan Hao for designing the radar module, Theo Miller for porting the software library to Arduino, Justin Hsieh for work on the communication protocol, Ken Lutz for helping with deployment logistics, and Noah Klugman for help with Signpost assembly.

## REFERENCES

[1] Abari et al. 2015. Caraoke: An e-toll transponder network for smart cities *(SIGCOMM'15)*.
[2] Abramson 1970. THE ALOHA SYSTEM: Another Alternative for Computer Communications *(AFIPS'70 (Fall))*.
[3] Adkins et al. 2017. Energy Isolation Required for Multi-tenant Energy Harvesting Platforms *(ENSsys'17)*.
[4] Ambiq Micro. Apollo2 MCU Datasheet. (2017).
[5] Arduino. Arduino website. arduino.cc. (2017).
[6] Arm. Mbed OS developer website. os.mbed.com. (2017).
[7] Bales et al. 2012. Citisense: Mobile air quality sensing for individuals and communities design and deployment of the citisense mobile air-quality system *(PervasiveHealth'12)*.
[8] Basara et al. 2009. Overview of the Oklahoma City Micronet *(8URBAN)*.
[9] Bilandzic et al. 2008. Laermometer: A mobile noise mapping application *(NordiCHI'08)*.
[10] Bouillet et al. 2013. Fusing Traffic Sensor Data for Real-time Road Conditions *(SENSEMINE'13)*.
[11] Burke et al. 2006. Participatory sensing *(WSW'06)*.
[12] Campbell et al. 2006. People-centric urban sensing *(WICON'06)*.
[13] Catlett et al. 2017. Array of things: a scientific research instrument in the public way: platform design and early lessons learned *(SCOPE'17)*.
[14] Cheng et al. 2014. AirCloud: a cloud-based air-quality monitoring system for everyone *(SenSys'14)*.
[15] City of Columbus. Smart Columbus. (2017). columbus.gov/smartcolumbus.
[16] City of Denver. Denver – A Smart City. (2017). denvergov.org/content/denvergov/en/transportation-mobility/smart-city.html.
[17] Devarakonda et al. 2013. Real-time air quality monitoring through mobile sensing in metropolitan areas *(SIGKDD'15)*.
[18] Dutta et al. 2009. Common sense: Participatory urban sensing using a network of handheld air quality monitors *(SenSys'09)*.
[19] Eriksson et al. 2008. The pothole patrol: using a mobile sensor network for road surface monitoring *(MobiSys'08)*.
[20] European Commission. Using EU funding mechanism for Smart Cities. (2013).
[21] Federal Highway Administration. Manual on uniform traffic control devices. (2009).
[22] Girod et al. 2006. The Design and Implementation of a Self-calibrating Distributed Acoustic Sensing Platform *(SenSys'06)*.
[23] Hull et al. 2006. CarTel: a distributed mobile sensor computing system *(SenSys'06)*.
[24] Illston et al. 2009. Design and deployment of traffic signal stations within the Oklahoma City Micronet *(8URBAN)*.
[25] Intel 2015. *Intel Edison Compute Module*. Rev. 004.
[26] Jeong et al., Prospective materials and applications for Li secondary batteries. *Energy & Environmental Science* 4, 6 (2011).
[27] Ledeczi et al. 2005. Multiple simultaneous acoustic source localization in urban terrain *(IPSN'05)*.
[28] Lee et al. 2006. Efficient data harvesting in mobile sensor platforms *(PerCom'06)*.
[29] Lee et al., Mobeyes: smart mobs for urban monitoring with a vehicular sensor network. *IEEE Wireless Communications* 13, 5 (2006).
[30] Levy et al. 2017. Multiprogramming a 64 kB Computer Safely and Efficiently. In *SOSP'17*.
[31] Li et al. 2015. SenseFlow: An Experimental Study of People Tracking *(RealWSN'15)*.
[32] Li et al. 2015. The development of smart cities in China *(CUPUM'15)*.
[33] Liu et al., The long-term average performance of flat-plate solar-energy collectors: With design data for the U.S., its outlying possessions and Canada. *Solar Energy* (1963).
[34] LoRa Alliance. LoRa Alliance. lora-alliance.org. (2017).
[35] Maisonneuve et al., NoiseTube: Measuring and mapping noise pollution with mobile phones *(ITEE'09)*.
[36] Mathur et al. 2010. Parknet: drive-by sensing of road-side parking statistics *(MobiSys'10)*.
[37] Microchip. AT86RF230. (2006).
[38] Mohan et al. 2008. Nericell: rich monitoring of road and traffic conditions using mobile smartphones *(SenSys'08)*.
[39] Multitech 2017. *MultiConnect xDot*.
[40] Murty et al. 2008. Citysense: An urban-scale wireless sensor network and testbed *(HST'08)*.
[41] Mydlarz et al., The implementation of low-cost urban acoustic monitoring devices. *Applied Acoustics* 117 (2017).
[42] National Science Foundation. NSF commits more than $60 million to Smart Cities Initiative. (2016).
[43] Ney et al., SeaGlass: Enabling City-Wide IMSI-Catcher Detection *(PoPETs'17)*.
[44] Nordic Semiconductor. NRF52840 Specification. (2017).
[45] NREL. Solar Maps. http://www.nrel.gov/gis/solar.html. (July 2016).
[46] Opiela et al. Maintaining Traffic Sign Retroreflectivity: Impacts on State and Local Agencies. (2007).
[47] Rana et al. 2010. Ear-phone: An end-to-end participatory urban noise mapping system *(IPSN'10)*.
[48] Rao et al. 2008. Identification of low-level point radiation sources using a sensor network *(IPSN'08)*.
[49] Review. Smart City Development in China. (2014).
[50] Rose et al. 2010. Mapping the Urban Wireless Landscape with Argos *(SenSys'10)*.
[51] Sen et al. 2012. Kyun Queue: A Sensor Network System to Monitor Road Traffic Queues *(SenSys'12)*.
[52] Sundararaman et al., Clock synchronization for wireless sensor networks: a survey. *Ad hoc networks* (2005).
[53] Texas Instruments. MSP430F20xx Data Sheet. (2005).
[54] Thiagarajan et al. 2009. VTrack: accurate, energy-aware road traffic delay estimation using mobile phones *(SenSys'09)*.
[55] Traffic Safety Warehouse. Radar Speed Control Sign. trafficsafetywarehouse.com/Radar-Speed-Signs/products/69/. (2017).
[56] U-blox 2016. *SARA-U2 Series Datasheet*. Rev. R14.
[57] United Nations Department of Economic and Social Affairs. World Urbanization Prospects: The 2014 Revision. (2014).
[58] Washington State Department of Transportation. Traffic Signals. wsdot.wa.gov/Operations/Traffic/signals.htm. (2017).
[59] Welsh. Personal communication. (9 April 2016).
[60] Wilcox 2007. *National Solar Radiation Database 1991-2005 Update: User's Manual*. Technical Report. National Renewable Energy Laboratory.
[61] Work et al., Mobile Millennium Demonstration-Participatory Traffic Estimation Using Mobile Phones. (2009).