A Bluetooth-Smart Insulating Container for Cold-Chain Logistics

Pai H. Chou
Dept. of EECS
UC Irvine
Irvine, CA USA
phchou@uci.edu

Cheng-Ting Lee, Zan-Ya Peng, Jo-Ping Li, Tong Kun Lai,
Chun-Min Chang, Cheng-Hsun Yang, Yi-Lin Chen
Dept. of Computer Science
National Tsing Hua University
Hsinchu City, Taiwan

Chin-Chung Nien, Li-Huei Chen,
Li-Yen Lai, Jen-Chao Lu, Shang-Chih Hung
Service Systems Technology Center
Industrial Technology Research Institute
Hsinchu City, Taiwan

***Abstract*—This paper describes an intelligent insulating shipping container (IISC) and the supporting backend for cold-chain logistics. Each IISC can monitor the interior temperature of the container during shipping of temperature-sensitive items and alerts the deliverer wirelessly via a Bluetooth-Smart (also called Bluetooth 4.0 Low Energy, BLE) connection on their smartmobile without infrastructure or dongle. The smartmobile also serves as a BLE-tag reader, user interface, and communication uplink for shipment tracking. BLE enables the IISCs to be wakable on demand within a short latency while lasting for six months to a year on each battery charge. When integrated into cold-chain logistics, our IISCs are expected to not only reduce the amount of goods spoiled during shipping but also improve fuel efficiency of fleets by enabling effective use of passive cooling elements.**

***Keywords*-Bluetooth Smart, insulating shipping container, temperature monitoring, smartmobile, cold-chain logistics**

I. INTRODUCTION

Thermally insulated shipping containers (ISC) are an important technology for the transport and delivery of temperature-sensitive items. Examples range from frozen food and ice cream to medical items such as serum and donated organs. Vehicles equipped with refrigerators or compressors are available but incur high capital cost and suffer from poor fuel efficiency. Cold packs can save fuel and cost, but they alone cannot guarantee proper maintenance of temperature, especially if the container must be opened several times along delivery stops. The ability to monitor the interior temperature of the ISCs and alert the deliverer for remedial action such as putting in another cold pack is necessary, and wirelessly connectivity is highly desirable, since it means eliminating the extra step of plugging and unplugging, tripping hazard, rusting, or worn-out connectors.

Wireless technologies may be passive or active. Passive ones such as RFID allow the tags to operate without batteries, since power is emitted by the RFID reader. However, RFID *readers* can be relatively expensive and cannot support smarter protocols in case of obstruction of RF signals by other containers. Active RF protocols can support multi-hop relaying, but idle listening can consume much power. Although average idle power can be reduced by duty cycling the receiver, doing so may still incur longer latency than desired when waking up a node.

To address these problems, we propose our intelligent insulating shipping container (IISC) design and the supporting system for cold-chain logistics. We choose Bluetooth Smart, also called Bluetooth 4.0 Low Energy Technology (BLE), to enable direct connectivity between a smartmobile and a large number of IISCs without infrastructure or a dongle. BLE not only is low power but also enables a smartmobile to serve as a tag reader, a friendly graphical user interface, and a wireless Internet uplink. This not only is convenient for the deliverer but also saves the cost of the tag reader.

We embed a BLE-enabled sensor node into each container to monitor its interior temperature using a thermocouple at a user-settable rate and to alert the deliverer wirelessly on the smartmobile upon detecting abnormal temperature. Each node logs its temperature history throughout shipping. The smartmobile also serves as the signature interface for the recipient upon delivery and as the wireless uplink for real-time update of the delivery status. A novel feature is the use of barcode or QR-code to assist with container identification and pairing with the smartmobile. To overcome the problems of obstruction, we also enhance BLE with the multi-hop extensions to BLE for relaying data. These containers are supported by a logistics backend that maintains the configurations of the containers and the status of the payload.

II. RELATED WORK

This section reviews existing containers including ISCs and temperature-monitoring technologies for shipping temperature-sensitive goods.

A. Temperature Maintenance

Temperature can be maintained by passive refrigerant packs or by active temperature control. Refrigerant packs can contain gel or blue-ice commonly used for maintaining the temperature range inside the ISCs. They are frozen prior to shipping and consume no power during shipping. However, for longer trips or precise temperature requirements, an active device such a refrigerator, freezer, or thermoelectric cooler (TEC) may be used. They require a power source provided by the shipping vehicle, a battery, or a solar panel. TECs contain no moving parts and can be used for both heating and cooling by controlling the power polarity.

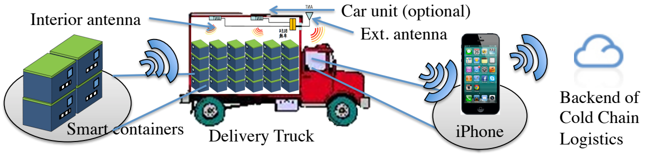

Figure 1: Major subsystems of the cold-chain logistics system

B. *Temperature Monitor*

Some ISCs, especially the active temperature-controlled ones, are built with sensors to monitor their interior temperature. The AcuTemp AX27L mobile refrigerator/freezer [1] provides two temperature set-points of +4°C or −22°C. It can be powered by 115 V or 230 V AC, by two 21-Ah gel cell batteries, or by the car lighter outlet at 12 VDC. It can operate for up to five days entirely on battery power. The wireless temperature sensor (WTS) [2] monitors the air temperature in the vehicular refrigerators via Bluetooth and consumes about 3-65 mA. Cold-Trace [3] includes not only temperature sensors via Bluetooth to a PDA in a truck but also cellular data (GPRS) upload with geocoordinate stamps and fleet management to a logistics backend. However, its power consumption is not published. Low-power temperature monitors with alerts to smartphones have been proposed based on an extension of the IEEE 802.15.4 beacons to improve battery lifetime similar to the concept of connection intervals in BLE [4]. However, it requires a dongle and its logistic backend is not described.

Our proposed IISC supports per-container temperature monitoring. It can ensure temperature requirements are met during shipping. Furthermore, our IISC's use of BLE enables it to scale to a large number of nodes while adapting to a variety of vehicles. Our IISC consumes significantly lower power than existing solutions do. It also can be charged by wireless power for convenience.

III. SYSTEM OVERVIEW

This section presents an overview of the subsystems of the cold-chain logistics system, followed by a description of the operating scenarios.

A. *Subsystems of Cold-Chain Logistics*

Fig. 1 shows the four major subsystems of the cold-chain logistics system: the node on the IISC, the optional car-relay unit, the smartmobile, and the backend. This subsection describes their required behavior; the details of their designs will be presented in later sections.

1) Node on the IISC: The *node* on the IISC is a battery-powered BLE node with a temperature sensor and non-volatile memory for data logging. It can connect to different systems over time, including a smartmobile, a PC, a car unit, or other IISCs. The node, battery, and inductive-charging coil are embedded in a recessed area on the outside of the container, and a hole for the thermocouple can be inserted to sense the interior temperature of the container.

2) Car-Relay Unit: The car-relay unit, or just *car unit* for short, is an optional relay between the nodes on the IISCs in the cargo and the driver. A relay is necessary in case the IISCs are shipped in an all-metal enclosure (as opposed to canvas), making it impossible for the RF signal to penetrate. The car unit relays data between the IISCs and the smartmobile while also capturing the cargo-interior temperature. The car unit is unnecessary if the IISCs are within the RF range of the smartmobile in the driver cabin.

3) Smartmobile: A *smartmobile* is a general term for a smartphone or a tablet and contains Wi-Fi, built-in camera(s), cellular data, GPS, and BLE. It serves as the GUI for the packer and the driver. During shipping, the smartmobile collects and uploads data from the IISCs to the backend server via its Internet connection with a geocoordinate stamp. It also renders alerts from the IISCs as they occur.

4) Backend: The logistics backend provides services including container asset management, location tracking, and query of temperature history, inventory, packing lists, and delivery history of the goods. The backend maintains the mapping between identifiers and the MAC addresses of the BLE transceiver of the IISCs so that the packer can associate the packing list with the IISC and subsequently track the delivery history. It also maintains the status of the IISCs during shipping and delivery. It logs the smartmobile's location, the IISCs' temperatures, and alerts from the IISCs. The data are kept for post-processing, analysis, or targeted tracking of specific IISCs or delivery trucks.

B. *Workflow*

The workflow of the cold-chain logistics is divided into the following phases.

1) Configuration: This is to write the factory-default settings into the nonvolatile memory of the node of a new IISC. First, a BLE-enabled computing device (which may be a PC) scans all BLE nodes wirelessly to obtain the MAC addresses and creates a spreadsheet. Second, the user interface loads and displays the spreadsheet. When all data have been reviewed, the configuration tool generates the GS1 code, sets it as the device name for the node, writes the mapping of the MAC address and GS1 code to the backend server, and prints the barcode for each IISC. Then, our smartmobile app can be used to scan each barcode, which causes the IISC with that ID to light up its LED to identify itself so that the operator can put the barcode inside a protective, dirt-resistant, scratch-resistant clear-plastic envelope and stick it on that IISC.

2) Initialization: This is done before the (empty) IISC is dispatched. The operator uses a smartmobile's camera to scan the barcode on the outside of the IISC to obtain the identifier, which it uses to query the backend server to obtain the necessary information (BLE device name and passcode)

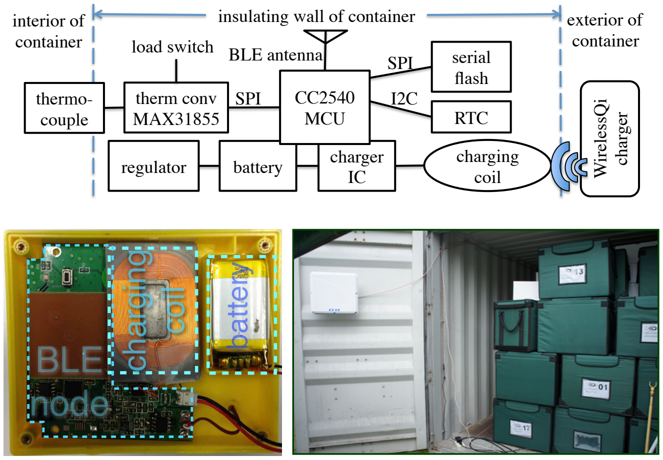

Figure 2: (a) Hardware block diagram of the proposed IISC. (b) BLE node on the IISC. (c) stacked IISCs in cargo.

to pair with the IISC. The smartmobile app also sets the real-time clock, sensing interval, and temperature threshold.

3) Packing: The packer uses a smartmobile to ensure all information is up to date in the IISC and the backend. The packer scans the barcode of each packing list, scans the barcode for the IISC to use, packs the items, and sends the packing data to the backend.

4) Transport: Each node senses and logs data locally in its nonvolatile memory at the specified rate, usually on the order of once every tens of minutes. When not sampling and logging, each node remains in sleep mode but can be waken on demand by either the driver's smartmobile or the car unit to retrieve the logged data. The smartmobile relays the temperature data with the geocoordinates to the backend.

5) Locating: At each stop, the delivery person locates the IISC whose contents are destined for the current location by using the smartmobile to wake up the IISC and blink its indicator light.

6) Delivery: The deliverer hands the goods to the recipient and collects a signature on the smartmobile. Optionally, the deliverer can show the temperature time-history plot for proper temperature maintenance during transport. The signature, time and location stamps, and the packing list identifier are transmitted by the smartmobile to the backend.

IV. HARDWARE

We designed custom hardware boards for the BLE-based temperature-sensing node for the IISC and the car unit.

A. *BLE Node on the IISC*

Fig. 2a shows the block diagram of the BLE node. It uses the TI CC2540 BLE microcontroller unit (MCU) with 256 KB program flash and 8 KB on-chip SRAM [5]. The peripherals include a 4 Mbit serial flash (S25FL016K), real-time clock (RTC), and MAX31855 as the cold-junction compensated thermocouple-to-digital converter. Fig. 2b shows a photo of the node on the IISC. It is powered by a 700 mAh rechargeable lithium-polymer battery pack that can be charged by an inductive coil based on the Qi wireless charging standard [6]. The remaining battery level can be determined using a charge counter IC. The RTC can be powered by a coin-cell battery.

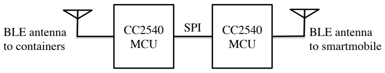

Figure 3: Block diagram of the car unit.

A step-down DC-DC converter is needed for a lithium-polymer battery to power the node, since the on-chip LDO (low drop-out) linear regulator of the MCU has a lower voltage range. However, most DC-DC converters have a quiescent current of around 1 mA, even though the MCU draws 1 μA when sleeping. We chose a DC-DC converter with low quiescent current to minimize sleep power.

We use a thermocouple instead of a temperature sensor. A thermocouple consists of a hot and a cold junction that produce a voltage difference. The MAX31855 is used as the cold-junction compensated thermocouple-to-digital converter [7]. It has a 0.25°C resolution and ±2°C accuracy in the range of -200°C to $+700$°C. It repeatedly converts the temperature, and the last converted value can be read over its digital interface (SPI). To save power, we put a load switch [8] to power-manage this component, although it needs about 1-2 seconds to stabilize after powering up.

B. *Car Unit*

The car unit acts as a bridge between the IISCs in the metal cage and the driver. As shown in Fig. 3, it contains two sets of BLE-and-antenna interfaces: one on the cargo side and one on the driver side. The two MCUs are connected via SPI and housed in the same enclosure mounted inside the cargo. The cargo-side MCU acts as a BLE master while the smartmobile-side one acts as BLE slave. The car unit can be powered either by battery or by the truck. One antenna covers the cargo side and the other on the driver side.

V. SOFTWARE

This section describes the software organization for the subsystems, including the node firmware, mobile app, configuration tool, and backend server.

A. *Node Firmware*

1) Firmware Organization: We adopt the firmware organization recommended by TI for CC2540. The hardware-abstraction layer (HAL) maps resource names such that the same source code can be compiled without editing for chip-specific or board-specific assignments. The operating system abstraction layer (OSAL) is runtime layer above the drivers and below the stack and user code. It supports messages, tasks, timing, memory, and power management, and is required by TI's BLE stack. Above that, the BLE profiles define the messaging protocols for specific application classes. Our application code is organized as tasks invoked by the OSAL event loop.

2) Thermocouple Code: The code for thermocouple first initializes the MAX31855 thermocouple converter. In every iteration of the sampling loop, our program reads data and error code from the MAX31855. Although the MCU can call the temperature compensation function for more accurate alert reporting, an alternative may offload this feature to the smartmobile or the backend.

3) Communication and Networking: Our firmware includes a layer responsible for maintaining state of communication and networking. A BLE device includes six possible link-layer states: *standby* (unconnected), *advertising* (broadcasting or slave availability), *scanning* (master looking for slaves), *initiating* (master pairing with slave), *master*, and *slave*. A master-slave pair agrees on a *connection interval*, which can range from 7.5 ms to 4.0 s.

We implemented a multi-hop extension on top of BLE for supporting multi-hop networking in case some IISCs are obstructed by others and are not directly reachable wirelessly. Our source-routing algorithm uses a version of breadth-first search (BFS) adapted to a task-event structure. Unlike Bluetooth BR/EDR, a BLE node cannot be a master and a slave at the same time, so when the parent node tells a child to find its neighbors, the child needs to change from slave to master. Once the child finishes finding neighbors, it changes from master back to slave before it can be reconnected with the parent node for returning the information.

4) Data Logger: We lay out the external flash memory for data logging. The first block (64 KB) stores the ID followed by 3 bytes (19 bits) of the queue pointer, which indicates where the data should be written next. The rest of the flash memory is used for logging the temperature data with the time stamp. The RTC value is 6 bytes long, including the year, month, day, hour, minute, and second. The temperature data is 4 bytes. Given the total capacity of 4 Mbits, our external flash can store about 50,000 time-temperature records. The queue pointer wraps around after reaching the end of the flash memory. The temperature data with the time stamp are stored in a 300-byte RAM buffer first and flushed to flash when full. The delivery phase is when we flush the queue pointer back to the flash. This logging scheme avoids excessive wearing of certain flash sectors. The end-to-end latency of sending 20 bytes data over BLE to write to the flash takes 503 ms, whereas reading 20 bytes data from the flash of container over BLE takes 445 ms.

5) Time Synchronization: In loading stage, the user synchronizes the node's RTC to the time on the smartmobile, which is assumed to be accurate based on the cellular service provider's clock. Time synchronization is performed after the IISC receives data including the temperature threshold value, current time, and the sensing interval over BLE protocol. The node acknowledges successful setting.

B. Mobile App

We have developed a mobile app as the primary user interface for most stages of the cold-chain logistics. The app exposes different subsets of the functions depending on the role of the person who is logged in to the system.

1) Packer: If the user is logged in as a packer, then the app presents a button to download the packing lists from the backend server. It also shows buttons for the smartmobile to scan the barcode of each IISC, the options of setting the real-time clock, a dialog to set the sensing interval and temperature threshold values on the node, or to use default values. After completing each packing list, the mobile app uploads the packing data to the backend server.

2) Driver and Deliverer: If the user is logged in as a driver and deliverer, then the app asks the user to input the license plate number (LPN) of the truck associated with the packing lists. Depending on whether the optional car unit is used, the smartmobile establishes a connection with the car unit or with the IISCs. During transporting, the smartmobile or car unit periodically queries the IISCs for the last sampled temperature and time stamp and attaches its geocoordinates stamp before logging to its local database while also sending a copy to the backend server. Abnormal temperature alerts are also rendered on the app by beeping and vibration.

At the destination, the app sorts the packing lists based on the closest geocoordinates. The app shows a button to send an LED-blink command to help locate the IISC. The app provides a signature area for the customer to acknowledge receipt with an option to plot the temperature time-history. The signature image is uploaded to the backend server for delivery confirmation.

3) Barcode Scanning: We use ZBar SDK [9] for scanning barcode. The SDK is an open-source software suite for reading bar codes from various sources, such as video streams, image files, and raw intensity sensors. It supports many popular *symbologies* (i.e., types of bar codes), including EAN-13/UPC-A, UPC-E, EAN-8, Code 128, Code 39, Interleaved 2 of 5, and QR Code. Each IISC has a barcode that encodes its ID. The smartmobile scans and decodes the barcode for the ID, requests the server to look up the node name, and pairs with the node.

C. Configuration Tool

The configuration tool writes the factory-default settings and establishes mapping between the node and the container for the IISC. The configuration tool is organized into the communication module and the barcode processor.

1) Peripheral Scanning: The configure tool is written in Python and runs on a PC. It connects to the BLE transceiver using the `pySerial` package. It creates a thread responsible for receiving packets from the transceiver. It sends a `GAP_DeviceDiscoveryRequest` command to the transceiver

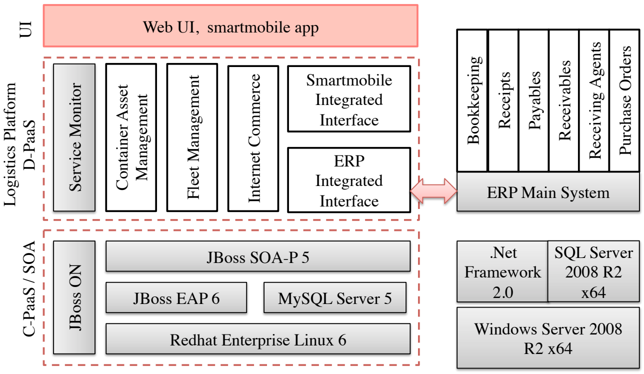

Figure 4: Organization of Backend Server Platform.

to discover peripherals, and the scanned peripherals reply with the address type and address in the packet. Then, the tool displays the list of the scanned nodes.

2) Node-Container Mapping with Barcode: The configuration tool provides an interface for the user to fill in the data needed to generate a unique ID for each container to be configured for factory-default settings. The tool saves the mapping of all peripheral nodes' MAC addresses and containers' GS1-128 code (composed of the company prefix, asset type, check digit, and serial number (GRAI)) to a spreadsheet. Later, the configuration tool can load the spreadsheet and generate the GS1-128 code using these four fields. The mappings of the IDs of all peripherals nodes to the containers are saved on the backend server.

The configuration tool uses Python's `win32print` to print the barcode labels on a label printer. To help the user locate the node corresponding to a barcode, a smartmobile can scan the barcode using the built-in camera, and the smartmobile will send an LED-flashing command to the corresponding node. The user can then paste the printed barcode label on the correct container.

D. *Backend Server*

The backend is a service-oriented architecture (SOA) that balances performance, flexibility, and scalability. The SOA middleware integrates different modules to provide a diverse range of services while maximizing reuse. Fig. 4 shows the block diagram of this backend server platform. All modules provide services in the form of JavaEE5-compliant web services. MySQL is used for data storage and different web services exchange data through JBoss SOA-P middleware for facilitating service composition while increasing visibility of service operation. In addition, the use of JBoss ON enables the entire system consisting of multiple servers to maximize utilization of CPU, memory, disk space to be managed more effectively and minimize downtime.

VI. EVALUATION

We evaluate our IISC design by several metrics: wireless data rate, power consumption and battery life, RF performance, and temperature accuracy.

Table I: Data throughput over connection interval and number of nodes.

Connection Interval (ms)	20-40		20-125	20-1000
#simult. conn. nodes	1	4	8	10
total conn. latency	< 2 s	< 2 s	< 6 s	< 42 s
aggr. throughput (KB/s)	1.5	**6.6**	3.3	0.6

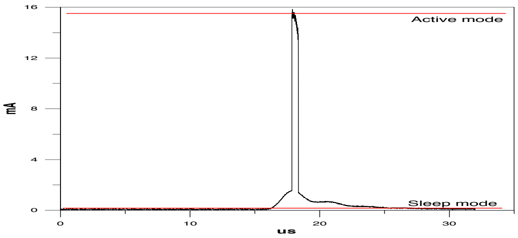

Figure 5: Current consumption of our IISC.

A. *Performance*

1) Data Rate of BLE: In BLE, the length of the connection interval and the number of simultaneously connected (paired) nodes can impact the available bandwidth that can be shared among the nodes. In our case, the nodes transmit data upstream (i.e., towards the smartmobile). Table I shows the aggregate data throughput and latency over different lengths of connection interval and number of simultaneously connected nodes. In our implementation, up to 10 nodes can be simultaneously connected, but the aggregate throughput is maximized (6.6 KB/s) when four nodes are paired with a connection interval of 20-40 ms. This implies an upper bound of 675 samples per second (e.g., 675 containers at once per second, 6750 containers at once per 10 seconds, etc). Since temperature does not need to be sampled too often, our implementation can scale to a large number of containers.

B. *Battery Life*

Fig. 5 shows the measured current consumption of 15.5 mA in active mode and 0.15 mA in sleep mode. Table II expresses the result of Fig. 5 in terms of battery lifetime. Compared to WTS [2], which is based on classic Bluetooth and lasts for a few days, ours is expected to last for over six months on the same battery. Multi-hop about consumes 4.5 μJ per hop.

Table II: Battery life of IISC.

Status	WTS [2]		Ours	
	Current	**Batt. Life**	**Current**	**Batt. Life**
Idle	6.895 mA	4.23 days	0.15 mA	195 days
Read temp./5 sec	8.65- 24.65 mA	1.18- 3.37 days	0.1508 mA	193 days
Read temp./10 sec			0.1504 mA	194 days

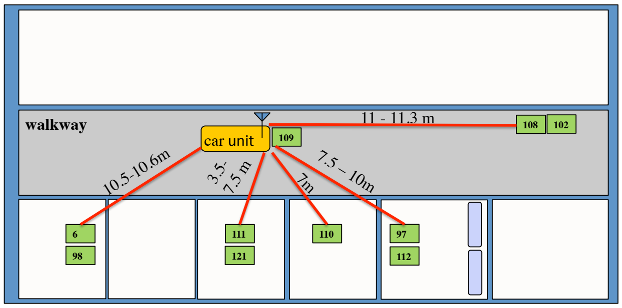

Figure 6: RF test layout.

Table III: RF Performance

Node#	dist (m)	RSSI	success	Node#	dist (m)	RSSI	success
121	3.5	−67	93%	97	10	−75	91%
109	0	−77	65%	6	10.6	−78	68%
110	7	−79	97%	102	11	−78	89%
111	3.5	−76	53%	112	7.5	−83	70%
108	11.3	−75	99%	98	10.5	−81	94%

C. *RF Performance*

Fig. 6 shows the test layout inside a cargo container. The green rectangles with numbers are the IISCs, placed in different partitions, while the car unit is located near the center at the ceiling level. Table III shows the RF performance of our IISC with the car unit. It shows that the RSSI value is in the range of −67 to −83 dBm, well above the noise floor.

D. *Temperature Sensor*

Fig. 7 shows time-history plots of temperature measured by three thermocouples, including two from our IISC and one from the Testo445 as the gold reference. All sensors are put in the containers for about 10 minutes before sampling starts at once very 5 seconds. Fig. 7a shows freezing temperature results whereas Fig. 7b shows room temperature. As can be seen, our thermocouples are indeed just about −2°C lower than the Testo 445's in both frozen and room temperatures, but ours shows better stability than the Testo 445. One other difference may be that our thermocouples are bare metals whereas the Testo 445's are protected by plastic. The most important consideration is power: our thermocouple consumes no more than 4 mA, whereas the Testo 445 consumes 2040 mA.

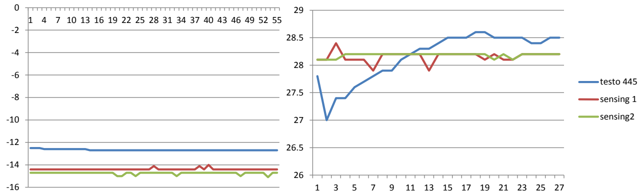

Figure 7: Temperature measurement in IISC.

VII. CONCLUSIONS

We have presented the design of an intelligent insulating shipping container and a backend server in the cloud for supporting all stages of logistics operation. Our smart container is the first to use Bluetooth Smart (BLE) protocol to transmit its interior temperature to a smartmobile carried by the deliverer for real-time monitoring. BLE, in conjunction with our low-power circuit design, enables our system to consume significantly lower power than previous designs do. Moreover, our containers can wirelessly connect to smartmobile devices without requiring a dongle or infrastructure support. We also take advantage of smartmobiles' built-in features for alerts and data and location upload. Moreover, we facilitate one-step secure pairing with a container by scanning a barcode using the smartmobile's built-in camera.

Acknowledgments

This project was supported by the Ministry of Economic Affairs (Taiwan) through subcontract SSTC-102-65-02.

REFERENCES

[1] AcuTemp, "Cold chain shipping containers, temperature sensitive insulated boxes, thermal management solutions," http://www.acutemp.com, 2013.

[2] Q. Shan and D. Brown, "Wireless temperature sensor using Bluetooth," *Proc. IWWAN, London, UK*, 2005. [Online]. Available: http://www.ctr.kcl.ac.uk/IWWAN2005/papers/22.pdf

[3] Y. URSA, P. Perez, and A. Meissner, "Cold-Trace: a mobile-based traceability solution rendering fleet management more effective," in *Exploiting the Knowledge Economy: Issues, Applications, Case Studies*, P. Cunningham and M. Cunningham, Eds. Barcelona, Spain: IOS Press, 2006.

[4] S. Zöller, A. Reinhardt, H. Guckes, D. Schuller, and R. Steinmetz, "On the integration of wireless sensor networks and smartphones in the logistics domain," *Proceedings of the 10th GI/ITG KuVS Fachgespräch Drahtlose Sensornetze*, pp. 49–52, 2011.

[5] Texas Instruments, "CC2540 2.4GHz Bluetooth Low Energy System-on-Chip," http://www.ti.com/product/cc2540, 2013.

[6] Wireless Power Consortium, "Qi Wireless Power Consortium – creating a standard for wireless charging," http://www.wirelesspowerconsortium.com/, 2013.

[7] Maxim Integrated, "MAX31855 cold-junction compensated thermocouple-to-digital converter," http://datasheets.maximintegrated.com/en/ds/MAX31855.pdf, 2012.

[8] Texas Instruments, "Single channel, ultra-low resistance load switch," http://www.ti.com/lit/ds/symlink/tps22965.pdf, 2013.

[9] Jeff Brown, "ZBar iPhone App user guide," http://zbar.sourceforge.net/iphone/userguide, 2011.

[10] AcuTemp, "VXC-2 operations manual - VaxiCool$^{®}$ mobile refrigerator/freezer, Model VXC-2," http://www.usamma.amedd.army.mil/assets/docs/VXC-2%20Rev4%20Jan07.pdf, June 2006.