

Hijacking Power and Bandwidth from the Mobile Phone's Audio Interface

Ye-Sheng Kuo, Sonal Verma, Thomas Schmid, and Prabal Dutta
Electrical Engineering and Computer Science Department
University of Michigan
Ann Arbor, Michigan 48109
{samkuo,sonalv,thschmid,prabal}@eecs.umich.edu

ABSTRACT

We endow the mobile phone with a low-cost, open interface that can parasitically power external peripherals, and transfer data to and from them, using analog, digital, and serial signaling, using only the existing headset audio port. This interface, called *HiJack*, allows the mobile phone to easily integrate with a range of external sensors, opening the door to new phone-centric sensing applications. In this paper, we characterize the signaling and power delivery capability of the audio jack, design circuits and software to transfer data and harvest energy, and evaluate the performance of our designs. We also use the mobile phone's audio channel to create a layered communications stack that supports low-level, analog signaling and high-level, multiplexed data communications with external devices. Our design supports a single, bi-directional communications channel at a data rate of 8.82 kbps over a Manchester-encoded serial stream, using just a few discrete components and the hardware peripherals found in almost any microcontroller. Our harvester delivers 7.4 mW to a load with 47% efficiency using components that cost \$2.34 in 10K volume. Integrating the pieces, we present a combined system for delivering data and power over audio, and demonstrate its use by turning an iPhone into an inexpensive oscilloscope, EKG monitor, and soil moisture sensor, all at price points accessible to most consumers in developing regions.

Categories and Subject Descriptors

B.4.2 [HARDWARE]: Input/Output and Data Communications—*Input/Output Devices*; C.3 [COMPUTER-COMMUNICATION NETWORKS]: Special-Purpose and Application-Based Systems

General Terms

Design, Experimentation, Measurement, Performance

Keywords

Mobile phones, Energy harvesting, Phone peripherals, Audio communications, Participatory sensing

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ACM DEV'10, December 17–18, 2010, London, United Kingdom.
Copyright 2010 ACM 978-1-4503-0473-3/10/12 ...\$10.00.



Figure 1: The iPhone headset plug and its pinout. iPhones use a 3.5 mm phono jack/plug to output audio to headphones and receive input from a microphone. The headphone connections are: (1) left earphone (tip), (2) right earphone (ring), (3) common/ground (ring), and (4) microphone (sleeve). The measured impedance of the iPhone headset between the left (or right) earphone and common is 33 Ω . The measured impedance between the microphone and common is approximately 640 Ω .

1. INTRODUCTION

The mobile phone is the most pervasive personal communications and computing platform ever created and yet, among its various analog interfaces, only one is truly open: the headset port, as shown in Figure 1. In this paper, we take a closer look at this common interface and assess its utility for augmenting the mobile phone with a range of phone-powered peripherals. We show that the mobile phone headset port can be used to efficiently power external peripherals and communicate with them, enabling many new phone-centric applications. But why, beyond openness, should we use the headset port? There are many reasons: it is simple, inexpensive, ubiquitous, and documented.

But perhaps even more important is the fact that the headset interface is backward-compatible with most phones in use today, and many recycled ones too, so existing phones could form the basis for many health and communications applications in developing regions. This paper explores the feasibility of this interface for power delivery and the generality of the interface for data transfer. Our goal is to understand the design space, enable phone-powered peripherals, and make the phone the center of a new instrumentation ecosystem for developing regions.

An affordable and ubiquitous plug-and-play interface for attaching sensors could have many applications. It could turn the mobile phone into a scientific data collection instrument, a medical device, or an agricultural tool by connecting low-power sensors to a source of connectivity and power. The mobile phone, with its ever improving computing, communications, and graphics capabilities is an obvious choice. More generally, by leveraging a standard and ubiquitous interface like the headset jack, the phone could also serve as a multimeter or oscilloscope – useful tools that otherwise may be out of the reach of most citizens in developing regions, but well within their grasp as a \$5 phone accessory.

Parameter	Value
Hardware	iPhone 3GS [1]
Software	SignalScope Pro [2]
Function	Signal Generator
Output	Headphones
Type	Tone
Frequency	20 Hz to 24 kHz (5 kHz nom)
Amplitude	0.00 dB
Pan	0.000%
Volume	Maximum

Table 1: Experiment parameters for determining the available power from the iPhone 3GS headset port.

Using the headset port to power and communicate with external peripherals poses several engineering and research challenges. The headset output is a low voltage signal, often even lower than typical transistor threshold voltages. To be useful, it must be converted to a higher voltage using energy harvesting and voltage boosting circuits that can operate with input AC voltages in the 200 mV range. Due to the limited voltage headroom, simple rectification is difficult without substantial power losses, and maximum power point tracking may be required in some cases. Matching the harvesting circuit’s cost, complexity, and conversion efficiency with the ideal audio waveform also presents an iterative co-design problem. Using the audio output to deliver power and data functionality requires exploring these design tradeoffs.

In this paper, we characterize the power available from the audio jack, design a circuit to harvest this power, and evaluate the efficiency of the conversion. We find that the headset can deliver 15.8 mW per channel from the iPhone’s headset port. We present a circuit that can harvest energy from a single channel and an audio signal that when played on the phone can maximize the output power from the harvesting circuit. Our energy harvesting circuit delivers up to 7.4 mW to a load – a 47% power transfer efficiency compared with the output power capability of the headset port – using just \$2.34 in electronic components.

We also demonstrate that a pair of (coded) audio signals can be generated by the phone processor and transmitted to both the energy harvesting circuit (for power transfer) and a microcontroller (for data transfer). Conversely, we show that the microcontroller can generate a coded signal that can be sampled by the mobile phone’s microphone input and decoded by the phone to present a stream of digital data, establishing an 8.82 kbps bi-directional data stream using the hardware resources found in a microcontroller.

Integrating the pieces, we present an oscilloscope, EKG monitor, and soil moisture sensor that runs partly on the phone and partly on an external microcontroller (MCU) powered using the mobile phone’s right audio channel. The two processors communicate using the left audio channel (phone to MCU) and microphone channel (MCU to phone). We also present a single circuit board, measuring just 1.0” x 1.0”, on which application-specific sensors are attached.

2. ENERGY HARVESTING

Our first design goal is to harvest energy from the headset jack of a mobile phone, convert it into a more usable form, and achieve high conversion efficiency in the process. We begin by characterizing the AC waveforms that are available, the output impedance of the headset port, and the range of variables available for manipulation. We then design and evaluate an energy harvesting circuit to convert the available power into a more usable form.

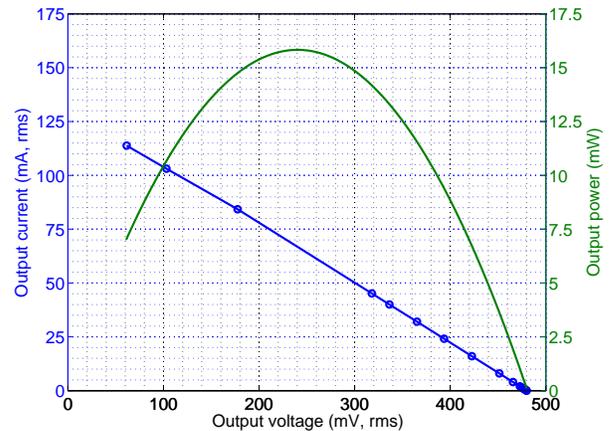


Figure 2: Available power from the iPhone headset jack. The IV curve data show that it is possible to draw 15.8 mW from an iPhone 3GS with an ideally-matched 3.6 Ω load. The iPhone can supply enough to power to operate many low-power electronics. To be useful, however, the current must be rectified, the voltage must be boosted, and the output must be filtered.

2.1 Determining Available Power

We now explore the question of how much power can be harvested from iPhone’s headset port. To do so, we use the Faber Acoustical iPhone SignalScope Pro software [2] to generate a range of audio frequencies, from 20 Hz to 24 kHz, and output them over the iPhone’s audio port. We find that the output power is independent of frequency, so we use a 5 kHz AC tone in our subsequent experiments. Table 1 shows the settings to generate the output.

A load is connected between the right audio channel and common line on the headset and varied from 0 Ω to 15 k Ω , and the output voltage and load current are measured at several points. A linear fit of the data yields the (essentially linear) IV curve shown in Figure 2. From these data, we generate the power transfer curve, which shows that maximum power transfer occurs at 240 mVrms and when delivering 66.0 mArms, for a 3.6 Ω load impedance.

2.2 Exploring the Design Space

We next explore the question of how to efficiently harvest the energy produced from the headset output. The two engineering challenges are to increase the signal amplitude and convert the AC signal into a DC one. Figure 2 shows that the open circuit voltage, V_{oc} , is less than 500 mV and that the maximum power point voltage, V_{mpp} , occurs at 240 mV. These voltages are far below the turn on voltages of switching regulators (typically in the range of 800 mV to 900 mV). They are also below the required startup voltage, after rectification, of ultra-low voltage step-up DC-DC converters, like the Seiko S-882Z [3], which require 300 mV to start.

Rectification losses can be significant in both high-power and low-voltage systems. In our case, for example, to achieve maximum power transfer, an RMS current of 66 mA is required. When rectified using even a low- V_f Schottky diode like the DFSL120L, a 200 mV forward voltage drop occurs (See Fig 1 in reference [4]), meaning that 80% of the power is lost during rectification, and only 20% can be delivered to the load.¹

¹This assumes that only a single rectifier diode is on the path, which would of course reduce the available power by 50%. If two diodes are on the path, as would be the case for a bridge rectifier, the losses would be substantially higher.

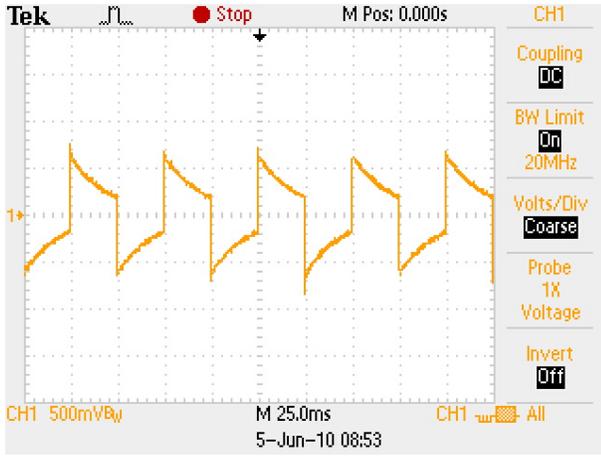


Figure 3: The audio output waveform observed externally when a 20 Hz square wave is transmitted over the audio output channel. The sharp transitions followed by characteristic exponential-decay curves show the output is AC-coupled, eliminating the simple option of driving the audio output with a constant DC voltage.

Synchronous rectification is sometimes used to reduce losses, where a FET switch is used instead of a diode [28]. In low-voltage applications like ours, the problem is generating a sufficiently high gate drive voltage to turn on the FET switch. Given the low voltages involved, this would require several stages of inefficient voltage multiplication, perhaps using a ladder circuit.

We end this section on design alternatives by eliminating two simple, but ultimately unworkable, options: harvesting DC directly from the audio output and harvesting DC from the microphone bias voltage. Figure 3 shows the waveform that is observed on the audio output when a 20 Hz square wave signal is generated on the phone. The characteristic exponential decay curves suggest that the output is AC-coupled, and is therefore high-pass filtered, blocking DC. This eliminates the possibility of simply generating a DC output voltage to power the external devices. Using the microphone bias voltage is also problematic because we plan to use it as the data input channel to the phone, which will be modulated externally by a microcontroller.

2.3 Harvesting Energy Efficiently

To sidestep the two basic engineering challenges – low-supply voltage and need for rectification – we use a step-up microtransformer, followed by FET-based rectification, followed by a blocking Schottky diode, followed by filter capacitors, as shown in Figure 4. One key element of the design, the microtransformer, leverages a recently introduced device for flyback and step-up for energy harvesting applications. These new transformers are small (6 mm x 6 mm x 3.5 mm), have high coupling coefficients (> 0.95), and are available in a range of turns ratios [5]. We use a 1:20 ratio in our energy harvester design.

The stepped-voltage is passed through a FET bridge for rectification. Since the stepped-up voltage is substantially higher than the FET threshold voltage, the FETs are in conduction and offer marginal loss. Another benefit to stepping-up the voltage is a reduction in current flow through the blocking diode, and therefore a reduction in forward voltage drop. However, since the diode is an exponential device, this unfortunately does not result in a substantial decrease in the forward voltage drop, but it does eliminate the

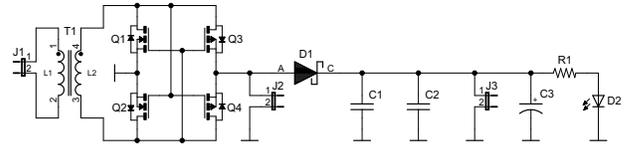


Figure 4: The energy harvesting circuit. A 1:20 microtransformer boosts the input voltage. A FET bridge efficiently rectifies the AC current to DC. A low- V_f Schottky diode provides low-loss blocking to prevent the output filter capacitor from discharging through the FET bridge. An (optional) LED with current-limiting resistor provides a visual power indicator.



Figure 5: Energy harvesting circuit. The transformer, Schottky diode, LED, and resistor are visible on the top side. The FET rectifier and filter capacitors are visible on bottom side.

voltage drop from a second diode in the rectifier. And, since the diode forward voltage drop is only a small fraction of the boosted voltage, this design incurs a small inefficiency compared to direct rectification of the original low-voltage signal.

Matching the load and source impedances is critical to achieving high power transfer efficiency from a power supply to its load. In this case, the impedance offered by the microtransformer’s primary winding should be matched to the iPhone’s audio output port’s impedance of 3.6 Ω . The transformer’s datasheet states that the primary DC resistance is 200 m Ω and primary inductance is 25 μH , which we also verify empirically. Since the transformer’s DC resistance is small compared to the power supply’s output impedance, we focus on the transformer’s impedance. The impedance, X_L , offered by an inductor is

$$X_L = j\omega L = j2\pi fL.$$

Rearranging to solve for f , the desired excitation frequency, gives

$$f = \frac{X_L}{2\pi L}.$$

Substituting our measured impedance and inductance values gives

$$f = \frac{3.6 \Omega}{2\pi \times 25\mu\text{H}} = 22.9 \text{ kHz}.$$

Note that the target excitation frequency sits just at the edge of what the iPhone is capable of producing. Fortunately, however, we have complete control over the excitation frequency within the audio band, so we can generate a 22 kHz waveform which will achieve near optimal power transfer to the energy harvester circuit.

2.4 Evaluating Performance and Cost

To evaluate the performance and cost of our design, we implement the energy harvesting circuit, as shown in Figure 5. The circuit only requires a footprint of 1.0” x 0.35” (although the current board includes a 0.15” unused area for manufacturing reasons) and several header export lines. The circuit’s small size makes it suitable for embedding inside a small peripheral device or even a head-stick plug, like the one shown in Figure 1.

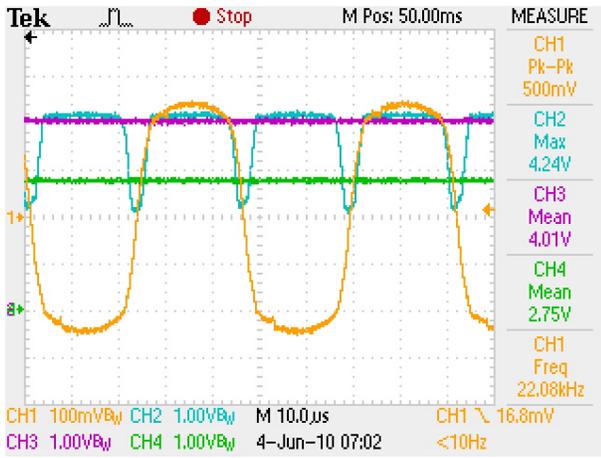


Figure 6: Energy harvesting circuit operation. Channel 1 (orange) shows the filtered audio excitation signal. Channel 2 (cyan) shows the signal after rectification. Channel 3 (magenta) shows the output after passing through a blocking diode and getting filtered. Channel 4 (green) shows the voltage across the LED. Overall, a low amplitude AC signal is efficiently converted into more usable form.

2.4.1 Performance

Figure 6 shows a trace of the circuit in operation. The iPhone generates a 22 kHz, 500 mV peak-to-peak square wave that is band-pass filtered (substantially lower excitation frequencies result in poor power transfer). The RMS value of the signal is 207 mV, meaning that approximately 15 mW is delivered by the phone, or about 90% of peak power. Channel 1 (orange) shows this filtered audio output signal. Channel 2 (cyan) shows a peak 4.24 V signal after rectification using the FET bridge. Channel 3 (magenta) shows the output after passing the rectified signal through (a single) blocking diode, which drops 230 mV at peak current, providing at worst a 94.5% efficiency. Channel 4 (green) shows the voltage across the LED after the signal passes through a 699 Ω resistor.

Figure 7 shows the energy harvester power delivery capability. A load resistance is connected across the harvester’s output terminals and is varied from 0 Ω to 15 k Ω . The output voltage and load current are measured at several points. From these data, we generate the power transfer curve, which shows that maximum power transfer occurs at 3.5 V when delivering 2.11 mA, for a 1.7 k Ω load impedance. The open circuit voltage is 9.95 V and short circuit current is 3.54 mA.

The power supply ripple is less than ± 10 mV as long as the audio signal is present and the load is static. Our design currently does not include voltage regulation for several reasons: (i) it may not be needed in some applications; (ii) it is not needed for our example application; and (iii) the output voltage and supply filtering is often specific to the particular application. These results show we can harvest energy from the phone’s headset port, and convert it into a more useful form, using a simple and inexpensive circuit consisting of a half-dozen components.

2.4.2 Impact on Phone Power

To estimate the impact of energy harvesting on the phone, we modifying an iPod Touch to measure battery current and voltage. We play an audio file that generates the required 22 kHz tone and find that the iPod draws 91 mA at 4.04 V when the harvester is attached and 37 mA at 4.04 V when the harvester is not attached.

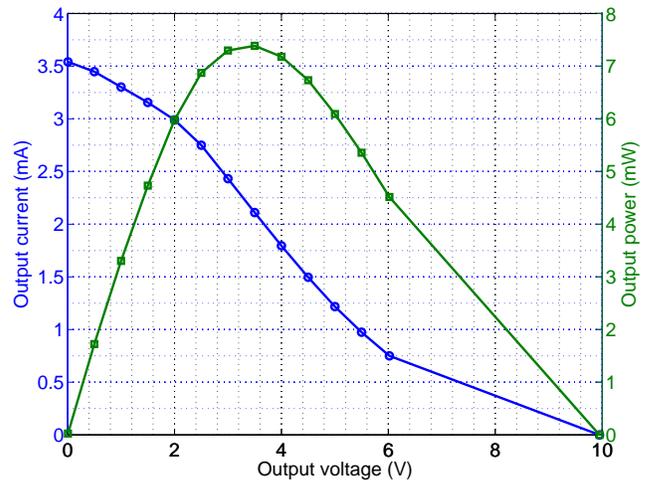


Figure 7: Power delivery capability of the harvester. The harvester delivers 7.4 mW, at 3.5 V and 2.11 mA, for an ideally-matched load of 1.7 k Ω . The power transfer efficiency, η_{eff} , is 47%. Although a higher conversion efficiency may be desirable, the output power, voltage, and current is enough to drive many low-power electronics.

These figures illustrate the substantial inefficiency of this approach compared with directly supplying power to a peripheral. We note, however, that the power output of an iPod is higher than an iPhone, making the numbers appear worse than they are on the iPhone.

2.4.3 Marginal Cost

Table 2 show the cost breakdown of the energy harvester components. The total marginal cost is \$2.34, assuming 10K volumes and list prices. These figures include the audio headset plug and board cost to build an operational energy harvester. The costs reflect list prices and exclude optional parts (J2, J3, R1, C3, and D2).

Ref	Desc	Mfg	Part	Cost (ea)
J1	Plug	Kobiconn	171-7435-EX	\$0.92
T1	XFMR	Coilcraft	LPR6235	\$0.40
Q1, Q2	N-FET	Zetex	ZXM61N03	\$0.14
Q3, Q4	P-FET	Zetex	ZXM61P03	\$0.14
D1	Diode	Diodes	DFLS120L	\$0.16
C1, C2	Cap	TDK	Y5V1A225Z	\$0.02
Fab	PCB	4pcb.com	HIJACK_A	\$0.26

Table 2: Energy harvester cost breakdown including circuit board and headset plug. Total cost at list prices, excluding optional components, is \$2.34 at 10K units.

3. PERIPHERAL INTERFACING

Our second design goal is to enable bi-directional interfacing between the mobile phone and external peripherals. The two constraints for the interfacing channel are: (i) it must operate in the audio frequency range, and (ii) it must be low-cost and energy-efficient. The first constraint reflects the bandwidth limitations of the phone audio channel while the second reflects the cost and power constraints of the space. Within these constraints, we consider three approaches to data transfer: direct encoding in analog voltages, encoding using modulated waveforms, and digitally transmitting serial data as a (Manchester-encoded) UART stream.

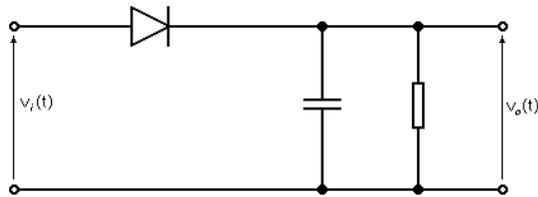


Figure 8: Diode detector circuit. The output voltage, $v_o(t)$ follows the envelope of the input voltage, $v_i(t)$.

3.1 Analog Signaling

Since the audio interface already bi-directionally transfers analog signals, it would seem straightforward to extend this interface to transfer information encoded as an analog voltage. Indeed, for sensors that require a small bias voltage and offer a transient (AC) response, the solution is simple. Such sensors can be connected directly between the microphone and common lines, or perhaps the sensors might require some minor impedance matching, buffering, filtering, or amplification. Sensors that fall into this category are microphones, piezo accelerometers, and magnetic card readers.

Conversely, a simple technique for obtaining a controlled output voltage from the AC-coupled output is to use an *envelope detector* circuit with a fast attack, slow decay response. Such a detector can be constructed from a diode, capacitor, and resistor (and optionally an op amp), as shown in Figure 8. A detector tracks the positive envelope of its driving waveform and responds quickly to increases in the input voltage. Tracking a decrease in the input voltage is a little more complicated since the diode is reverse biased. In such cases, the resistor discharges the capacitor until the capacitor voltage falls below the input voltage, at which time the diode is again forward biased and charges the capacitor.

In our application, the choice of resistor has to balance three factors: output ripple (when the resistor is too small given the driving frequency), tracking delay (when the resistor is too large for the envelope decay), and tracking error (when the resistor is too small and results in a non-trivial voltage drop across the diode). However, these are all particular to the application, so we do not discuss them further.

3.2 Modulated Digital Communications

Analog signaling allows just one input channel and one output channel for peripheral interfacing. In this section, we explore modulated digital communications over the audio port, using a microcontroller (MCU) that is powered by the energy harvesting circuit as one end of the communications link, and the phone’s processor as the other end of the link, with the two ends connected via the headset port. Using a microcontroller affords two benefits. First, it allows us to multiplex the channel. Second, it enables us to connect any (suitably low-power) sensor by using the microcontroller’s ADC, GPIO, I2C, SPI, UART, or other interface.

One source of inspiration for modulated digital communications over the headset port comes from acoustic dial-up computer modems used to connect computers together over the acoustic telephone network. Unfortunately, many high-performance digital signal processing techniques employed in conventional modems are not feasible in our system due to the limited power and computation resources. For example, a typical integrated modulator/demodulator circuit is often not sufficiently low power, drawing tens of milliwatts [6], and so a lower overhead approach is needed.

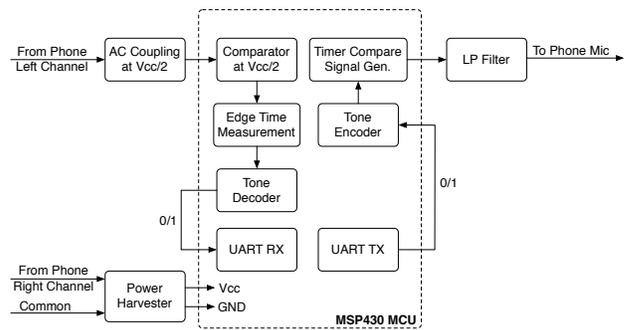


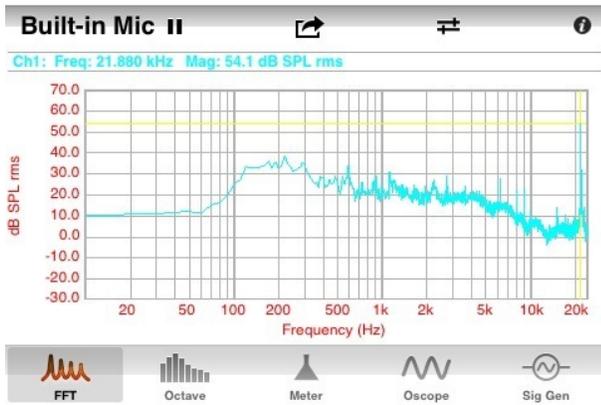
Figure 9: Digital communications system architecture. The phone’s headset port exports four wires: right channel, left channel, microphone, and common. The right channel provides power to the energy harvesting circuit. The left channel provides data output from the phone to the microcontroller. The microphone provides data input from the microcontroller to the phone. Both the phone and the microcontroller implement an FSK modulator/demodulator, although using very different implementation techniques.

The original Bell 202 modem signaling offers such a simple scheme, employing frequency shift keying (FSK) using two tones. A digital zero, or “space,” is represented by a 1200 Hz tone, while a digital one, or “mark,” is represented by a 2200 Hz tone. However, unlike the Bell 202 standard which specifies 1200 baud communications, we use a lower data rate of 300 baud in order to facilitate an implementation on a low-power microcontroller. At the digital level, we use RS-232 signaling to create a virtual UART abstraction over the audio serial bit stream. Since this protocol adds a start and stop bit to every byte, the effective data rate is limited to 30 byte/s.

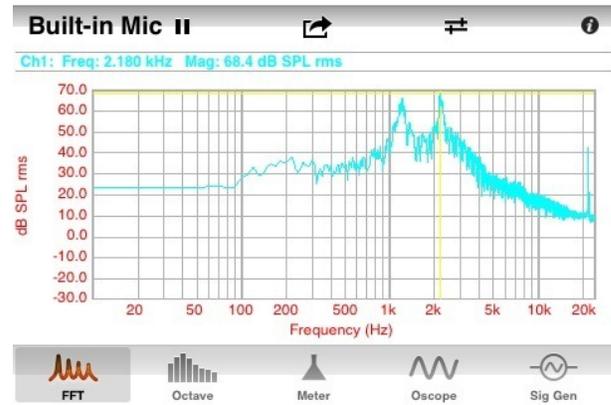
In order to efficiently implement FSK encoding and decoding on a microcontroller, it is useful to make maximum use of the available microcontroller hardware peripherals. One example, for instance, uses several hardware features including timers and USARTs to efficiently modulate and demodulate FSK signals [24]. Figure 9 presents an overview of our implementation of this unconventional, combined hardware and software, approach. There are two key benefits of using this approach. First, the hardware accelerators make it feasible to actually implement the modulation schemes on a low-end microcontroller. Second, the accelerators enable lower power operation than would be possible if the operations were executed in software.

Using this scheme, the main communication interface for an application running on the microcontroller is the UART receiver and transmitter peripherals, which are familiar to many programmers. The pins of both the receive and transmit unit of the UART are connected to other microcontroller peripherals which provide additional low-level symbol and data processing that occur transparently to the application code.

On the encoder side, the UART transmitter generates the data bits at 300 baud (Channel 2, cyan, top of Figure 10). These bits are fed back into a microcontroller interrupt line. A timer compare unit generates the correct frequency according to the incoming bit from the UART output. The output of the timer compare is a square wave at either 1200 Hz or 2200Hz (Channel 1, orange, bottom of Figure 10). This signal is low-pass filtered (not shown) before it is fed into the microphone port of the phone. This filtering reduces the high frequency components of the square wave that can be clearly seen in Figure 10.



(a) Noise spectrum



(b) Communications spectrum

Figure 11: Audio spectrum utilization. (a) shows the noise spectrum of the system (and the 22 kHz power tone). The sound pressure level is less than 40 dB. (b) shows the audio spectrum with data communications and power transfer. Two peaks are clearly visible at 1200 Hz and 2200 Hz. The 22 kHz tone that provides power is shown coupling back through a -30 dB path. We use the FFT function in the SignalScope Pro software on the iPhone to capture this data.

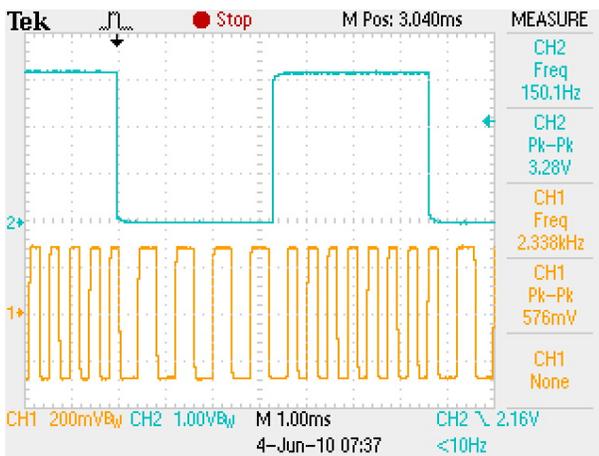


Figure 10: The data stream and its modulated transmission from the microcontroller to the phone. The top trace shows a 300 bps data stream, where a zero-bit is shown from 2.0 ms to 5.3 ms, followed by a one-bit from 5.3 ms to 8.6 ms. The bottom trace shows the FSK-encoded transmission of this data, where the zero-bit is encoded as a 1200 Hz tone and the one-bit is encoded as a 2200 Hz tone. Although the microcontroller-generated signals are square waves, they are low-pass filtered.

The left channel headphone output is a ground-referenced AC signal so we cannot directly process it. Thus we AC couple it to a voltage divider whose midpoint is set to the microcontroller's $V_{cc}/2$. The main idea behind the FSK decoder is to measure the zero-crossing time of the signal, and estimate the instantaneous, cycle-level frequency. We achieve this by comparing the signal to $V_{cc}/2$ in a comparator that is internally connected to a timer capture unit. In software, we calculate the time difference between rising and falling edges using a timer capture interrupt. The result of this decision is then output on a digital IO line that is externally connected back into the receive port of the UART peripheral. The UART receiver is then responsible for decoding the signal at 300 baud, as if it received a normal UART frame.

Implementing the FSK encoder and decoder on the phone is easier as we have a more powerful processor available, allowing for a more robust algorithm – a non-coherent FSK demodulator – on the phone [25]. In non-coherent FSK demodulation, the input is correlated with four different signals: a pair of sine/cosine signals at 1200 Hz, and a pair of sine/cosine signals at 2200 Hz. The correlation value of the two pairs is sampled and added together. Then, the sums of the two pairs are compared to each other to decide which frequency is being received. A state machine processes the stream of ones and zeros to detect the UART start bit before decoding the byte value transmitted in the FSK signal.

The FSK encoder on the phone creates a continuous-phase signal that switches between the two tones according to the transmitted bit. Figure 11 shows the audio spectrum used by the phone. Two peaks are clearly visible at 1200 Hz and 2200 Hz, plus the 22 kHz tone used to power the energy harvester. A test of the communication channel from microcontroller to phone found a bit error rate of 3.0×10^{-4} over a run length of 27,872 bytes.

3.3 Direct Digital Communication

The final option we explore is directly transmitting UART data over the headset port, without first FSK modulating it. However, since the audio channel is AC-coupled, a long sequence of zeros or ones saturates the channel and is difficult to decode. To sidestep this problem, we Manchester encode the UART data stream. Manchester encoding is a balanced 1:2 code that reduces the overall data rate by a factor of two but transmits an equal number of zeros and ones. The encoder works by replacing every 1 in the input stream with a 01, and every 0 with 10. This encoding is self-synchronizing since it guarantees a transition after at most two half-bits.

Figure 12 shows the UART data stream and its corresponding Manchester encoding as seen on the output of the headset port of a phone. We note two things. First, the waveform is not square and is thus band-limited. Second, the edge direction after every other bit in the Manchester encoded data indicates the UART bit. The second observation is key to the synchronization of our decoder. As the UART line is high if there is no communication, the Manchester encoding of the idle UART is a sequence of binary 01. However, the receiver cannot robustly distinguish a long run of 10s from 01s, leading to potential synchronization loss.

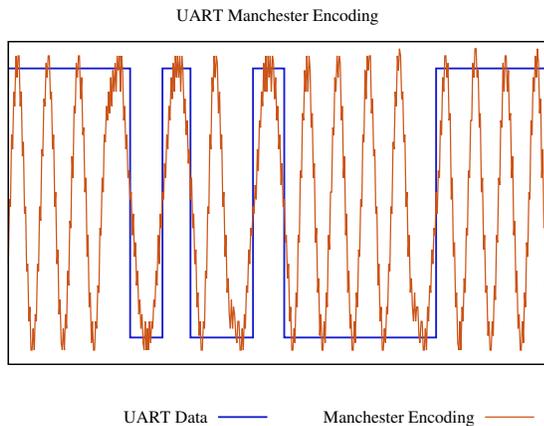


Figure 12: UART data and its equivalent Manchester encoded and low-pass filtered representation. Long runs of zeros and ones are indistinguishable but the key to synchronization is that a UART start bit, encoded as a high-to-low transition, creates a double-baud length positive symbol. The start bit’s high-to-low transition in the UART data is coincident with the falling edge of this double-baud positive symbol. This symbol synchronizes the receiver and transmitter after the link has been idle.

Our design addresses the resynchronization from idle as follows. Once the transmitter sends a UART start bit, the receiver detects a double-baud-length positive symbol, as seen in the first high-to-low transition of the UART data in Figure 12. This symbol synchronizes the receiver to the bit stream as it indicates with the falling zero-crossing that the current UART bit is a 0, and hence the prior sequence of bits were a sequence of 1s.

Once again, we use microcontroller peripherals to enable an efficient, low-power implementation. We measure the zero-crossing times of the Manchester encoded signal using a comparator at $V_{cc}/2$ driving the input line of a timer capture unit sensitive on both rising and falling edges. The edge-to-edge time is captured and compared to the baud rate, and then a state machine decodes and outputs the received UART bit.

A similar approach can be used for the decoder running on the phone. However, instead of using hardware peripherals, we implement the comparator and timer capture in software. The comparator simply implements a *less than* test which is used to slice the incoming audio samples into ones and zeros. To emulate a timer unit, we exploit the fact that the audio samples are taken by the phone at precisely 44.1kHz. Therefore, the time between successive samples is 22.68 μs . By counting the number of samples between a pair of zero crossings, we estimate the symbol width and thus can decode the Manchester encoded UART signal using the same approach as on the microcontroller (but in software).

Two factors limit the achievable baud rate using this technique. The first is that the phone’s audio sampling rate at 44.1 kHz, resulting in a maximum UART baud rate of 22.05 kbaud. The second is the clock source used on the microcontroller. Commonly used 32 kHz crystals are neither fast enough, nor offer clean baud rate divisors, to generate reliable Manchester encoding and decoding at 22.05 kbaud. And faster, digitally-controlled RC oscillators are not stable enough over longer periods of time. In addition, a higher frequency RC oscillator can prevent the microcontroller from entering sleep modes, thus increasing average power. Therefore, it is important to balance clock speed and data rate.

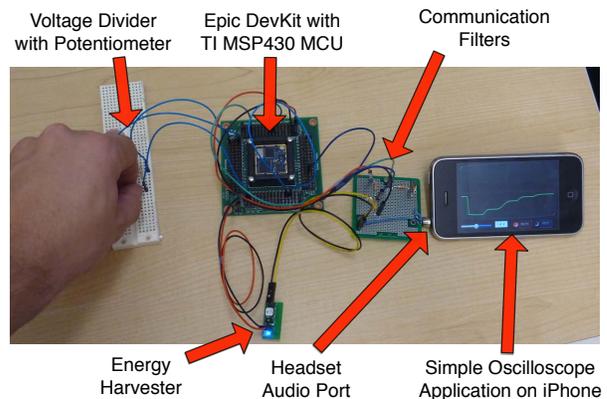


Figure 13: Prototype phone-centric oscilloscope application in operation. The system consists of four distinct subsystems which are shown working together: (i) iPhone; (ii) communication filters; (iii) energy harvester; (iv) microcontroller with potentiometer simulating a resistive sensor. The blue LED, located at the bottom of the energy harvester, is turned on and clearly visible in this image. As the potentiometer is turned, the iPhone-based oscilloscope displays the changing voltages.

We test several baud rate combinations to determine the limits of our current combination of iPhone 3G and the Texas Instruments MSP430F1611. We find that at 7,350 baud, there are no transmission errors after transmitting more than 30,000 bytes. At 8,820 baud, we find 2 byte errors after 65,000 transmitted bytes, indicating a *byte error* rate of less than 3.1×10^{-5} . Finally, at 11,025 baud, we find more than 30 *dropped* bytes out of 250 transmitted bytes – an unacceptable error rate. Based on these results, we conclude that our current implementation can sustain bi-directional communications reliably at 8,820 baud, or approximately 1 kbyte/s.

4. APPLICATIONS

Our final design goal is to illustrate that the various different subsystems – energy harvesting, data input, and data output – can all be combined into an integrated and functional application. To explore the design space, we create four applications. The first, a simple oscilloscope, demonstrates the proof of concept and allows us to microbenchmark power, code, and memory microbenchmarks. We then integrate the various HiJack subsystems into a basic platform that exposes key power, communications, analog, and digital interfaces. Finally, we build two applications – an EKG monitor and a soil moisture sensor – using this basic platform. Our results demonstrate the generality and extensibility of the HiJack in supporting novel applications.

4.1 Oscilloscope

Figure 13 shows a prototype handheld oscilloscope built around the energy harvesting and peripheral interfacing primitives described in the preceding sections. This system illustrates a canonical handheld instrument that uses the phone’s display for visualization, and the microcontroller’s ADC to measure an external signal. The system includes an iPhone, a breakout board populated with transmit and receive communication filters, the prototype energy harvester board, the Epic Mote [21] (which includes a TI MSP430F1611 [7]) in a breakout board, and a breadboard with a potentiometer. The potentiometer approximates an external analog sensor, but a digital or serial sensor could have easily been used.

State	LCD	Audio	HiJack	Power
iPod Standby	off	off	discon.	< 3 mW
iPod Home Scrn.	on	off	discon.	280 mW
Oscilloscope	off	on	discon.	280 mW
Oscilloscope	off	on	attached	480 mW

Table 3: Power draw breakdown of the oscilloscope application running on an iPod Touch 2G. The baseline standby power is less than 3 mW. The active baseline power (screen on, no apps running) is 280 mW. Running the Oscilloscope application without HiJack connected (screen off) draws 200 mW. Attaching HiJack adds an additional 200 mW.

4.1.1 Phone Software

At the heart of this system lies an application running on the phone that is based on the *aurioTouch* sample app [8]. Our version generates a 22 kHz tone on the right audio channel to power the energy harvesting circuit. The left audio channel sends a Manchester-encoded data stream to the microcontroller as described in Section 3.3. The phone’s microphone input line receives a similar, Manchester-encoded, data stream generated by the microcontroller. The phone application generates the raw, low-level data stream and is capable of decoding a stream encoded in a similar manner.

The iPhone application’s user interface, also shown in Figure 13, provides visual feedback of the voltage measured by the microcontroller ADC input. The graph displays a scrolling historical view of measured data while the text box shows the last received measurement. The data are transmitted from the microcontroller at regular sample intervals. A slider control on the application allows the user to change the sampling rate from about 1/10 Hz to 1 kHz, with the upper limit given by the 8.82 kbps encoding of the data stream of the audio channel. This slider setting is sent over the left channel to the microcontroller, which decodes the signal and changes its sample rate accordingly. We have verified that our application runs on the iPhone 3G, 3GS, 4G, iPod Touch 2G, and iPad. We hypothesize that it could be ported to other phones with rudimentary programming support as well, including Android, Windows, and Nokia.

4.1.2 Impact on Phone Power

We modify an iPod Touch 2G to measure the voltage and current of the battery while running our oscilloscope application. Table 3 summarizes the energy breakdown. We find that generating the 22 kHz tone and Manchester encoding and decoding on the phone draw about as much power as the LCD screen. The power draw of the harvester is again comparable to the energy used by the LCD screen. Thus, given the typical battery size of 1200 to 1400 mAh found on several generations of iPhones, and at a nominal 3.7 V, we estimate a battery life of 10 to 11 hours of continuous operation if the screen is turned off. However, we do not envision this as the typical usage model. Rather, because of its small size, we envision a user will carry a HiJack sensor in their pocket. When the user then wishes to take a measurement, she takes it out, plugs the sensor into her phone, and runs the measurement application. This is the typical usage model employed by participatory sensing applications.

4.1.3 Microcontroller Software

The microcontroller draws a mere 0.7 mA at 2.8 V while running the full application including communications encoder and decoder, UART, and ADC to sample the sensor. This is less than 2 mW, leaving over 5 mW for more sophisticated sensing and signal conditioning electronics.

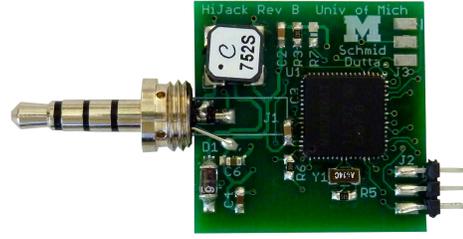


Figure 14: The HiJack base platform, with a 1” x 1” footprint, offers power (>5 mW), analog (2x 12-bit), digital (1x GPIO), and serial (1x I2C and 1x UART) interfaces, exported via connectors, and all multiplexed over the headset port. This board provides the functionality needed to build a variety of external sensor interfaces for the mobile phone.

Module	Size (bytes)
Total RAM	109
Timers, Alarms	1,280
HiJack Application	808
Others (UART, Arbiters, Init, etc)	2,378
Total ROM	4,466

Table 4: Code and memory usage of the MSP430 application. With a small Timer optimization, the total ROM usage can be brought down to 3316 bytes.

4.2 Phone Accessory Platform

One of our motivations for harvesting energy, rather than directly powering a peripheral with an external battery, is to reduce the peripheral’s form factor. While the prototype oscilloscope application is large due to the use of multiple prototype boards and breakout boards, the active components of the system can be integrated into a much smaller footprint circuit board. For example, the two largest components – a transformer and a TI MSP430F1611 microcontroller – measure a mere 6 mm x 6 mm and 9 mm x 9 mm, respectively, allowing us to easily integrate the components onto a double-sided board measuring just 1.0” x 1.0”, as Figure 14 shows.

Table 4 shows that we use 109 bytes of RAM, and with some small code optimization, less than 4 kB of ROM. Thus, in the next version of HiJack, we plan to use a newer TI MSP430² allowing us to incorporate all the components on a circuit board that requires only about a 1.0” x 0.20” footprint, which is small enough to place in a pocket, allowing anyone to carry a range of phone-powered peripheral devices with them.

By providing the most common analog, digital, and serial interfaces, this base board makes it easy to integrate many low-power sensors directly into nearly any mobile phone with a headset connector and rudimentary programming facilities.

One component that is missing from this platform is a voltage regulator. Unsure about the specific power requirements of peripheral devices, this design allows the sensors to regulate voltage as needed. In hindsight, this design choice complicates the interface between the base platform and peripherals since the base board power is unregulated. Subsequent designs (described below) correct this oversight by integrating the regulator directly into the HiJack base board.

²F2121 or F2122, depending on the application requirements, 16 mm² or 25 mm², \$1.30 or \$1.80 in 10k units, respectively.



Figure 15: Application-specific HiJack sensor boards. (a) A 3-lead, phone-powered EKG signal conditioning board that interfaces to HiJack, draws less than $400 \mu\text{A}$, and provides amplification, filtering, and baseline drift correction. (b) An ultralow-cost, passive soil moisture sensor that interfaces to HiJack using a resistive voltage divider. These sensor boards illustrate the breadth of different sensing applications that can be built using HiJack, showing HiJack’s versatility in meeting a range of sensing tasks.

4.3 Phone-Powered EKG Monitor

Hospitals, clinics, and urgent care facilities in developed regions have access to a battery of medical instruments unavailable to health care professionals in developing regions. One example of such an instrument is the EKG monitor. EKG signals are used to diagnose a wide range of medical conditions but they are often unavailable in all but the most advanced hospitals in developing regions. We argue, in this section, that this need not be the case. Rather, any hospital, village clinic, or doctor can have access to advanced health care instruments for little more than the cost of a mobile phone.

We illustrate our claim by designing a low-cost, low-power EKG monitor that uses HiJack for power and communications, and the mobile phone for visualizing the EKG waveform, as shown in Figure 15(a). The EKG monitor attaches to the square-inch form-factor HiJack base node, and extends the PCB to provide space for connectors to attach EKG leads in a 3-lead configuration. The EKG board draws less than $400 \mu\text{A}$, amplifies the input signal, passes it through a notch filter, and corrects for baseline drift. The mobile phone visualizes the EKG and can optionally transmit the data to the cloud for storage or real-time remote display (including on another mobile phone or even a Facebook application). The EKG board led to an iteration in the HiJack base board whereby the base board now provides regulated power (and a wider electrical interface which is visible as perimeter pins in Figure 15(a)).

4.4 Soil Moisture Sensor

An inexpensive soil moisture sensor could enable farmers in developing regions to better understand conditions on and under the ground, and this may lead to better crop management and higher crop yields [20]. The key challenge is in devising a sensor system that can be deployed at modest scale in developing regions. This requires inexpensive sensing, signal conditioning, and visualization.

To address this need, we design a low-cost soil moisture sensor as our final example of a HiJack phone peripheral. The sensor, shown in Figure 15(b), uses the HiJack phone accessory platform presented in Section 4.2 augmented with a simple breakout board that holds the top half of a voltage divider. The soil moisture sensor itself is built from two nails, gypsum, and a pair of wires. The nails are suspended in the gypsum which is then cured. One wire is attached to each to nail. The series circuit formed by wire, nail, gypsum, nail, and wire form the bottom half of a voltage divider. The sensor is buried underground. When the surrounding earth is damp or wet, water seeps into the gypsum and changes its conductivity [9]. These changes can be measured by the HiJack board and visualized using the mobile phone.

5. RELATED WORK

Energy harvesting has received considerable research interest in recent years. Harvester designs suited to radio frequency [27], piezo vibration [26], outdoor solar [23] and indoor lighting [22] have been explored. Many of the proposed designs involve either diode-based voltage doublers which have low efficiency at small input voltages or integrated circuits which have high upfront design and mask costs, making both of these approaches ill-suited to the target domain.

Recently, industry has begun shipping integrated circuits designed for energy harvesting applications. Linear Technology offers the LTC3108 [10] but the device requires a DC input voltage. In this paper, we showed that direct rectification leads to high losses, and so this chip is not ideal for the target domain. In addition, the chip requires several external components whose aggregate cost is higher than the cost point we are able to achieve with our discrete implementation. Another chip, the LTC3588 [11], can operate directly from an AC input, but it requires at least 2.7 V, as well as four capacitors and an inductor, again increasing cost.

Others have proposed using the iPhone’s audio jack as a communication interface. The Square Card Reader [12] is a credit card reader that attaches to a phone’s audio jack. Based on our own tear down and discussions with Square engineers, we find that the reader is a passive device, and that the voltage induced by a card swipe on the magnetic head is directly applied to the microphone input. Thus, the device does not actively harvest energy or transmit digitally-coded information. Rather, it is representative of the simplest kind of analog sensor that can be attached to a mobile phone.

Another device, iData [13], plugs into an iPhone headset port and claims to communicate using FSK modulation at 1200 baud, equivalent to the Bell 202 standard. No public details are available about how the device is powered. Prological provides a development kit called *homemade for integration* (H4I) [14] to which licensed developers get access to an SDK, and plans for the necessary external hardware components. H4I claims speeds of up to 19.2 kbaud, but external peripherals still need their own source of power. Serial output on the Android platform has also been demonstrated [15]. Our work is distinct as it provides both communications and power to a device directly from the audio jack at the same time.

There are many ways to interface external peripherals with a phone. Bluetooth and WiFi are the most common ways of connecting wireless peripherals, like headsets, keyboards, car stereos, heart rate monitors, or even cars [16], to a phone. While these standards are common on modern phones, they both require an external power source for the peripherals.

Additional interfaces on some phones include microSD [17] and USB On-The-Go (OTG) [18]. Although the microSD card slot is designed specifically for small peripherals, and it includes support for SDIO mode with SPI support, the port is small and often not easily accessible. Thus, connecting peripherals through the SDIO interface can be difficult. And, some devices, like the iPhone do not offer a microSD card slot. However, microSD can be a viable interface, as recently announced microSD WiFi cards illustrate [19]. While USB host capabilities seem like the ideal interface, only a few phones provide OTG capability. And, even when phones offer OTG, the standard only requires 5 V at 8 mA (40 mW), less than three times the power that the headset can deliver on each channel. OTG also requires an external USB slave. On the other hand, USB OTG can supply power very efficiently and in regulated form, which accrues to the lifetime of the mobile phone and simplicity of a peripheral's power supply design. Therefore, as USB OTG gains more widespread adoption on mobile handsets, it may well emerge as the preferred method for interfacing peripherals.

6. CONCLUSION

We show that it is possible to adapt the ubiquitous mobile phone headset port to provide power and data, allowing the phone to interface with a range of external peripherals easily and economically. With the basics of powering external devices and communicating with them now in place, it will soon be possible to plug a cubic-inch sensor into almost any mobile phone, power the device, read its output back, process the data, and present or transmit a result. By leveraging the widely used headset interface for power and data, HiJack enables plug-and-play peripherals whose charging cycles converge with those of the mobile phone. A parasitically-powered expansion interface for the mobile phone will enable many new applications for developing regions that are infeasible with current approaches. The mobile phone has already become *the* personal computer. Tomorrow it could become *the* oscilloscope, portable EKG monitor, and soil moisture sensor as well. This work takes a small step toward that future by enabling a new class of universal, low-cost interface options for the mobile phone.

7. ACKNOWLEDGMENTS

Special thanks to Eric Brewer for originally suggesting the mobile phone's audio interface could be used to power external sensors, engineers at Square, Inc. for details on how their card reader works, and the anonymous reviewers for their helpful feedback. This material is supported in part by National Science Foundation Award #0964120 ("CNS-NeTS"). Additional NSF support was provided under Grant #1019343 to the Computing Research Association for the CIFellows Project.

8. REFERENCES

- [1] <http://www.apple.com/iphone/iphone-3gs/>.
- [2] http://www.faberacoustical.com/products/iphone/signalscope_pro/.
- [3] <http://www.sii-ic.com/en/product1.jsp?subcatID=3&productID=1788>.
- [4] <http://www.diodes.com/datasheets/ds30444.pdf>.
- [5] <http://www.coilcraft.com/lpr6235.cfm>.
- [6] http://www.onsemi.com/pub_link/Collateral/MC1496-D.PDF.
- [7] <http://focus.ti.com/docs/prod/folders/print/msp430f1611.html>.
- [8] <http://developer.apple.com/iphone/library/samplecode/aurioTouch/Introduction/Intro.html>.
- [9] <http://www.cheapvegetablegardener.com/2009/11/how-to-make-cheap-soil-moisture-sensor-2.html>.
- [10] <http://www.linear.com/pc/productDetail.jsp?navId=H0,C1,C1003,C1042,C1031,C1060,P90287>.
- [11] <http://www.linear.com/pc/productDetail.jsp?navId=H0,C1,C1003,C1799,P90393>.
- [12] <http://www.squareup.com>.
- [13] <http://www.alexwinston.com/idata>.
- [14] <http://www.progical.com/>.
- [15] http://robots-everywhere.com/re_wiki/index.php?title=Serial_on_Android_using_the_audio_port.
- [16] <http://www.devtoaster.com/products/rev/index.html>.
- [17] <http://www.sdcard.org/developers/tech/sdcard>.
- [18] <http://www.usb.org/developers/onthego/>.
- [19] C. Davies. KDDI WiFi b/g microSD cards demonstrated. <http://www.slashgear.com/kddi-wifi-bg-microsd-cards-demonstrated-2350125/>.
- [20] P. Dutta and L. Subramanian. Human-enabled microscopic environmental mobile sensing and feedback. In *AI-D'10: Proceedings of the AAAI Spring Symposium on Artificial Intelligence for Development*, Mar. 2010.
- [21] P. Dutta, J. Taneja, J. Jeong, X. Jiang, and D. E. Culler. A building block approach to sensor networks. In *Proceedings of the 6th International Conference on Embedded Networked Sensor Systems, SenSys 2007, Raleigh, NC, USA, November 5-7, 2008*, pages 267–280.
- [22] J. Gummesson, S. S. Clark, K. Fu, and D. Ganesan. On the limits of effective hybrid micro-energy harvesting on mobile crfid sensors. In *MobiSys'10: Proceedings of the 8th Annual International Conference on Mobile Systems, Applications and Services*, June 2010.
- [23] X. Jiang, J. Polastre, and D. Culler. Perpetual environmentally powered sensor networks. In *IPSN'05: The Fourth International Conference on Information Processing in Sensor Networks: Special track on Platform Tools and Design Methods for Network Embedded Sensors*, Apr. 2005.
- [24] W. Lutsch. MSP430 embedded soft-modem demo. <http://focus.ti.com/lit/an/slaa204/slaa204.pdf>, 2004.
- [25] J. Proakis and M. Salehi. Communication systems engineering. 2002.
- [26] Y. K. Ramadass and A. P. Chandrakasan. An efficient piezoelectric energy harvesting interface circuit using a bias-flip rectifier and shared inductor. *IEEE Journal of Solid-State Circuits*, 45(1):189–204, Jan. 2010.
- [27] A. P. Sample, D. J. Yeager, P. S. Powlledge, A. V. Mamishev, and J. R. Smith. Design of an RFID-based battery-free programmable sensing platform. *IEEE Transactions on Inst. and Meas.*, 57(11):2608–2615, Nov. 2008.
- [28] M. Seeman, S. Sanders, and J. Rabaey. An ultra-low-power power management IC for wireless sensor nodes. In *IEEE Custom Integrated Circuits Conference*, 2007.