A Wireless Sensor Network for Smart Roadbeds and Intelligent Transportation Systems

by

Ara N. Knaian

S.B., Electrical Science and Engineering (1999)
Massachusetts Institute of Technology

Submitted to the Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degree of

Master of Engineering in Electrical Engineering and Computer Science

at the
Massachusetts Institute of Technology
June 2000

Author____
Department of Electrical Engineering and Computer Science
May 22, 2000

Certified by____
Joseph Paradiso
Principal Research Scientist, MIT Media Lab
Thesis Supervisor

Accepted by____
Arthur C. Smith
Chairman, Department Committee on Graduate Theses

A Wireless Sensor Network for Smart Roadbeds and Intelligent Transportation Systems

by

Ara N. Knaian

Submitted to the Department of Electrical Engineering and Computer Science on May 26, 2000 in Partial Fulfillment of the Requirements for the Degree of Master of Engineering in Electrical Engineering and Computer Science

Abstract

We have developed a wireless sensor package to instrument roadways for Intelligent Transportation Systems. The sensor package counts passing vehicles, measures the average roadway speed, and detects ice and water on the road. Clusters of sensors can transmit this information in near real-time to wired base stations for use controlling and predicting traffic, and in clearing road hazards.

The sensor package draws a maximum time-averaged current of 17µA from an internal lithium battery, allowing it to operate in the roadbed for at least 10 years without maintenance. The nodes cost well under $30 to manufacture, and can be installed without running wires under the road, facilitating wide deployment. Unlike many other types of traffic sensors, these sensors count vehicles in bumper-to-bumper traffic just as well as in widely separated traffic.

The devices detect vehicles by detecting the perturbations in the Earth's magnetic field caused by the vehicles. They measure this perturbation using an anisotropic magnetoresistive magnetic field sensor. The radio transmitters in the sensor are frequency-agile, and the sensors use a randomized sparse TDMA protocol, which allows several transmit-only devices to share a channel. The sensor package includes a custom-designed, compact, broadband, inexpensive printed circuit microstrip antenna for the 915 MHz U.S. ISM band.

We built a prototype sensor package, and installed it in a pothole in a city street. We used the sensor to monitor the traffic flow rate during free-flowing traffic and a traffic jam.

Thesis Supervisor: Joseph Paradiso
Title: Principal Research Scientist, MIT Media Lab

Table of Contents

Introduction

The congestion level on our nation’s roadways is spiraling out of control. The Texas Transportation Institute estimates that the total cost of congestion to commuters in 1997 was $72 billion, or $750 per commuter. [Schrank99, V-13] The amount of congestion-induced delay experienced by the average commuter in a large city such as Los Angeles or Boston has more than doubled since 1982. [Schrank99, III-9]

Better public transportation is the only sustainable long-term solution to road congestion. However, a nearer term solution is also required. The most obvious is to build new roads, but, in urban areas, this is generally not feasible, due to a lack of suitable land or insufficient funds. [DOT99] There is a major effort underway, the National Intelligent Transportation Systems (ITS) initiative, to use information technology to make better use of our nation’s roads.

One particularly compelling system envisioned by ITS workers is the Automated Traveler Information System. (ATIS) Before embarking on a trip, drivers could consult a web page to obtain accurate trip time estimates for various departure times and modes of transportation. If they chose to drive, a dynamic route guidance system would provide them with turn-by-turn directions based on up-to-the minute information about roadway speeds and congestion levels.

At the very least, this type of system would allow drivers to make better route decisions, to be confident that they were taking the most efficient route, and to plan their activities around traffic delays. More importantly, though, this type of system would make participating in a traffic jam into a conscious, planned action, rather than the consequence of losing a perceived route-planning gamble. Drivers might postpone discretionary trips, consider local or electronic alternatives, consider the use of public transportation, or even decide to move closer to work or to find work closer to home.

One of the largest obstacles to the implementation of this type of system is the shortage of accurate, real-time traffic data. Currently available traffic sensor systems (video, sonar, radar, inductive, magnetic, capacitive, PVDF wire, pneumatic treadle) use significant electrical power, so each sensor must be connected to the power distribution network. For sensors that are installed on electrical poles (video, sonar, radar), this costs a few hundred dollars. For sensors that are installed in the road, (inductive, magnetic, PVDF wire, capacitive, pneumatic treadle) this costs several thousand dollars[i]. Nevertheless, in-road sensors are still popular, because they are very accurate, provide direct information with very little ambiguity, can monitor road conditions (i.e. presence of ice), and do not require a human operator.

. Here, I present the design for a wireless in-road traffic sensor system. The sensors are small, low in cost ($30), and extremely rugged. They count and measure the speed of passing vehicles using magnetic technology. They also measure information about road conditions. Each cluster of sensors transmits data to a receiver mounted on an electrical pole up to 300 meters away, which relays the data to a processing station. Each sensor node consumes so little power that it can operate from a small internal lithium battery for at least 10 years.

Traffic Sensor Technology

Approaches

Traffic sensor systems can be classified into two types. The first type uses sensors, placed at many points throughout the road network, that count and measure the speed of passing vehicles. This data can be used to intelligently control traffic signals, and can be used with a simple queuing model to predict link times on the road network. The second type of system aims to measure link times directly, by tracking the progress of probe vehicles through the road network.

[i] The Massachusetts Highway Department charges $50/foot to dig up the road to install loop sensors, and $200/foot to install electrical conduits under the roadway.

Sensors that Count and Clock

The most common type of traffic sensor is the in-road inductive loop. A coil of wire several meters in diameter is buried under the road and connected to a roadside control box. The control box passes an electrical current through the coil. Passing vehicles change the inductance of the coil. In most units, the control box outputs a digital signal based on a threshold inductance---this signal can then be used to control traffic signals or to count vehicles. Two loops placed a few meters from each other can be used as a speed trap. Recently, some workers have successfully used the analog signals from inductive loops to classify vehicles and identify them later at other inductive loops further down the road. [Sun97] Inductive loops must be physically large, and this large size means that they cannot distinguish vehicles in bumper-to-bumper traffic, because two vehicles may be over the sensor at once. To install an inductive loop traffic sensor, it is necessary to completely tear-up and re-install a section of the road.

Traffic can also be detected using magnetic sensors. Magnetic traffic sensors detect passing vehicles by measuring disturbances in the Earth's magnetic field. [Caruso99] Magnetic traffic sensors are much more compact than inductive loop traffic sensors, and thus are better able to count vehicles in bumper-to-bumper traffic.

A company in the United Kingdom has developed "Intelligent Road Studs" to be placed along lane dividers, as passive reflectors are today. The studs charge an internal battery during the day with a photovoltaic cell, and light up an LED lamp at night. The studs can detect icy, wet, and foggy conditions, and can relay this information using infrared communication, so that studs further down the road can change color to warn drivers of upcoming dangerous conditions. [Graham-Rowe98]

Most commercial traffic reports are made using video cameras: Human operators sitting in a control room watch images from the cameras, identify incidents, and assign speed rankings to various roads. This technology does not require any hardware to be placed in the road. However, the need for human operators makes it cost-effective only for the most-traveled roads. Some workers are using machine vision to count and classify vehicles, with much success under ideal lighting conditions. [Beymer97, 495] However, outdoor machine vision is notorious for problems caused by day-night transitions, shadows, and headlight reflections off wet pavement. [Palen97, 2]

Above-road traffic sensors based on sonar, radar and lidar have all been developed. Under ideal conditions, these sensors are very effective. However, fog, smog, dust, snow, rain, and “roadwash” behind trucks can interfere with their operation. [Palen97, 2]

Systems that use Probe Vehicles

Another type of traffic sensor system aims to measure travel times directly, using probe vehicles. In one project, workers placed GPS tracking devices on a small subset of vehicles in a metropolitan area. By monitoring the progress of the instrumented vehicles through the road network, past link times could be measured, from which future link times could be extrapolated. Unfortunately, this approach requires a statistically significant number of vehicles be instrumented with GPS tracking devices, which is a daunting undertaking.

Another, more practical variation on this theme is the U.S. Wireless Corporation’s RadioCamera system, which can be used to make a map of all active cellular phones within a certain area. Their system uses the difference in arrival time of each telephone’s signal to all of the cellular towers in range to compute the position of each telephone. With this system, every car with an active cellular phone becomes a probe vehicle. [USW00] This technology has the potential to revolutionize the link-time prediction business, although it cannot be used for applications like congestion control and global traffic flow optimization by streetlight control, since it does not measure vehicle presence and does not compute link counts.

Another approach is the use of Automatic Vehicle Identification, a very well developed technology, to track vehicles. Roadside cameras read the license plate numbers of passing vehicles and relay this information to a central server. The server keeps track of the progress of individual vehicles through the road network, constantly updating a database of link times. Unfortunately, this approach suffers from most of the problems of outdoor computer vision described above, although to a lesser degree.

Traffic Sensor System Design Goals

We wanted to design a system that would allow easy, inexpensive monitoring of traffic flow statistics and environmental conditions on every major road. The sensors would have to be small, extremely robust, easily manageable, and installable at a low cost. The system would have to be capable of returning data for processing in near real time. To be installable at a low cost, the sensors could not require any wires to be installed under the road; they had to be internally or parasitically powered for their entire useful life, and had to communicate their state to the system by radio to wired base stations.

Node Architecture

Each node is a compact, self-contained package. It contains magnetic field sensors, a temperature sensor, a radio transmitter, a microcontroller and a lithium battery. (See Figure 1) The node is installed directly into the roadbed, at the center of the lane of interest.

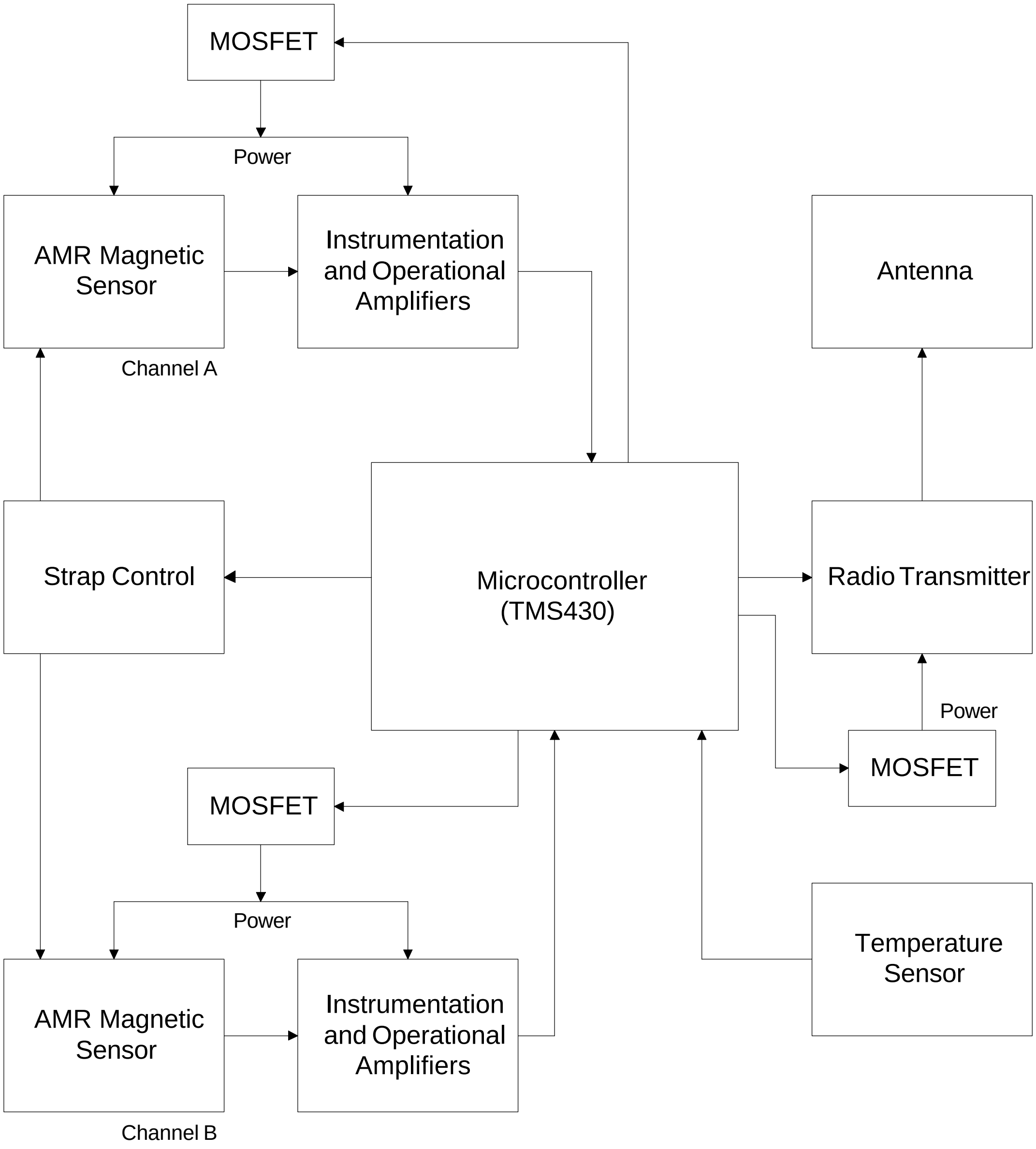


Figure 1: Node Block Diagram

The node samples the magnetic field at its front and back ends, and internally processes this data to count vehicles and compute the average speed of passing vehicles. Once a minute, it transmits a data packet containing its ID, vehicle counts in 10-second bins, average speed, and temperature to the base station. All of these fundamental design choices were motivated by a desire to achieve the other design goals while making the most efficient use of power, to allow for as long a battery life as possible. Table 1 shows the amount of energy used by the node for each operation.

Operation	Energy/Volt (ns-A)
Sampling Magnetic Field	47
Switching Processor to Active Mode	0.8
Performing an A/D Conversion	55
Starting Radio Transmitter	5500
Transmitting one byte	2900
Performing one cycle of processing	0.5
Set/Reset Pulse for sensor	2000
Offset pulses for calibration	240

Table 1: Energy used by node operations

The units of energy/volt, ns-A, are used for convenience, since all system components operate at the same voltage. To determine number of ns-A used by a task, multiply its current draw in µA by its duration is milliseconds. To determine the average current drawn in nA by the system due to a task, multiply the number of ns-A amps used by the task by its frequency is Hz.

. The node needs to sample the magnetic sensor fast enough so that the passing of a vehicle results in a significant, unambiguous excursion from the baseline. At slow speeds, when cars may be traveling bumper-to-bumper, this means getting at least three samples during the engine block. At high speeds, when cars must be separated by several meters, this merely means getting at least three samples during the vehicle. Assuming that there will not be bumper-to-bumper traffic traveling faster than 20 m/s, (about 40 MPH) and that an engine block is at least 60 cm long, to detect vehicles in bumper-to-bumper traffic, we need to take samples at no less than (20 m/s * 3 / 60 cm) 100 Hz. Sampling at this rate allows the node to detect well-separated vehicles of at least 3 m length traveling at up to 100 m/s, (about 200 MPH) which is more than adequate.

For the node to be a component in an ITS system, it is desirable for it to transmit data more frequently than the order-of-magnitude length of a traffic incident. On the other hand, it is useless for it to transmit data more frequently than the typical time taken to traverse the distance between two sensors. The node transmits packets once every 60 seconds. This allows the ITS system to detect incidents as short-lived as multiple light-cycle delays, and only adds 5µA to the node current draw.

The system takes speed measurements once every 10 seconds, as speed measurements are primarily useful to an ITS system for computing lane averages[ii], and the speed in crowded lane cannot change very quickly.

Vehicle Presence Detection and Speed Measurement

The sensor module detects passing vehicles by measuring disturbances in the Earth's magnetic field caused by passing vehicles. Almost all of today's road vehicles (even vehicles with polymer body panels) contain a large mass of steel. The steel has a much higher magnetic permeability than the surrounding air, which concentrates the flux lines of the Earth's magnetic field and increases the magnitude of the B field inside and in the immediate vicinity of the vehicle. This disturbance is detectable as far away as 15 m from the vehicle.

We selected this measurement technology because we felt that it was the most robust vehicle detection system available: it can operate through pavement, does not have any lenses to require cleaning, and is not affected by lighting conditions, temperature, fog, smog, or rain.

To detect vehicles, the system detects excursions of the sampled magnetic field value from a baseline. A passing vehicle generates an excursion first below and then above the baseline, (because it pulls field lines away from the sensor as it approaches and then pulls field lines onto the sensor as it drives over it) providing a simple, unambiguous way to detect multiple end-to-end vehicles.

[ii] This sensor cannot identify vehicles, (say, by license plate number) so data from it alone could not be used to automatically issue citations to speeding motorists. The author takes comfort in this fact.

To measure the speed of a vehicle, the node waits until it detects an excursion from the baseline, and then starts sampling at 2kHz using two sensors, one at the front of the node and one at the back. The waveforms at the outputs of the sensors are identical, except that they are shifted in time and may have slightly different electrical noise characteristics. The sensor waits for the signal from the rear sensor to cross the baseline, and then counts the number of samples until the signal from the forward sensor crosses the baseline. From this count, it can readily compute the speed of the passing vehicle.

Sensing Road Conditions

Since the sensor package is to be installed directly into the roadbed, it can also measure information about road conditions, such as whether the road is covered with snow, ice, or water. One approach to this sensing problem would be to use a capacitive sensor to infer the dielectric or conductive properties of the material above the sensor. This approach, commonly used for fluid detection, [Barlow72] would probably allow for very reliable detection of snow, ice, and water, at a low cost and using a small amount of power. Capacitive sensing at the desired resolution can be inexpensively implemented in a microcontroller-based system with a slow 8-bit A/D converter using synchronous undersampling. [Smith98, 34] One would attach a microcontroller output pin to an electrode near the surface of the sensor package, and attach an adjacent electrode to a microcontroller A/D input through a simple FET amplifier. The microcontroller would repeatedly apply a voltage step to the electrode, wait for a varying number of microseconds, and then sample the A/D input. By sliding the closing time of the A/D sampling gate across the ring-down from the step, the microcontroller can use synchronous demodulation to make the capacitive measurement.

Another approach is to measure the temperature: If the temperature is above freezing, ice cannot be present; similarly, if the temperature is at or below freezing, ice may be present. In addition, the rate of change of the in-road temperature versus the air temperature is determined by the heat capacity of the material above the sensor. We included a temperature sensor in the node to experiment with this approach.

Radio Technology

We considered using a spread-spectrum radio transmitter, because spread-spectrum systems are more immune to jamming and adjacent channel interference, and can be legally operated at higher data rates than narrowband, systems. However, spread spectrum systems need to synchronize to communicate. This typically requires a long time interval (several seconds) where the transmitter is drawing power but not transmitting bits. On top of this, many spread spectrum systems require broadband RF amplifiers, which are much less efficient than narrowband resonant-mode amplifiers, due to resistive losses associated with their lower Q. A typical ISM band spread-spectrum transmitter might consume 500mA for several seconds before becoming available to transmit bits.

On the other hand, simple narrowband FSK radio data transmitters of the same output power turn on within milliseconds, and draw only 10-20mA. Adjacent-channel interference and jamming are very real problems, but can be mitigated by using a frequency-agile narrowband system. Since this application does not require a high data rate, we chose to use narrowband FSK data transceivers.

Detailed Node Design

This section describes the sensor node hardware in detail, and contains an analysis of the node power consumption. Schematics of the node design are attached as Appendix A.

Data Acquisition System

The data acquisition system is designed to use as little energy as possible to take an accurate magnetic field strength measurement. The sensor output is typically sampled at 100 Hz, although this sampling rate is increased to 2 kHz, no more than once every 10 seconds, when measuring the speed of a vehicle.

Each node uses a pair of HMC1021S anisotropic magnetoresistive (AMR) magnetic field sensors from Honeywell to count and measure the speed of passing vehicles. [Honeywell99] One sensor is located at the front of the printed circuit board, and the other at the back. An AMR sensor is made from a thin film of nickel-iron (Permalloy) patterned onto a silicon wafer as a resistive strip. The nickel-iron strips change resistance in the presence of an on-axis magnetic field. The HMC1021 contains a Whetstone bridge, one leg of which is such a strip. When 3.0V is applied to the bridge, a bias current of 3.0mA flows through the bridge, and the on-axis magnetic field strength can be read across the bridge as a voltage of 3.0 mV/gauss.

The output from each sensor is amplified and made single-ended by an INA155 instrumentation amplifier from Burr-Brown. [Burr99] We selected the INA155 because it can operate from a single 2.7V supply and has a fast settling time. The output from the instrumentation amplifier is amplified further by an OP162 from Analog Devices, and then is fed into one of the ADC inputs of the microcontroller. We selected the OP162 because it can also operate from a single 2.7V supply, has a good quiescent current to gain-bandwidth-product ratio, and has rail-to-rail inputs and outputs. This circuit is duplicated for each field measurement channel. The microcontroller can switch power to the sensor and amplifiers for each channel using a Si2301 P-channel enhancement mode MOSFET from Siliconix. [Siliconix98]

The HMC1021 has two internal wire loops (called straps) that can be used to induce magnetic fields inside the sensor. The offset strap creates an on-axis magnetic field, which can be used to put the sensor inside a feedback loop. When stray on-axis magnetic fields greater than 10 gauss are applied to the sensor, some of the magnetic domains of the nickel-iron become misaligned, which causes the sensitivity of the sensor to be sharply reduced, and produces a measurement drift. The Set/Reset strap is used to create a strong cross-axis magnetic field to re-align the domains with the sensor axis, which restores the sensor's sensitivity and accuracy.

Current can be applied to the offset straps of the sensors under microprocessor control, so that the two sensors can be calibrated to give the same reading for the same magnetic field. The offset straps of both sensors are connected in series, so that they carry identical currents. When Q1 is turned on, a current set by R1 is applied to the straps. When Q2 is turned on, a different current, set by R2, is applied to the straps.

Set/Reset pulses can also be applied to the sensors under microprocessor control. Capacitor C7 charges to 3.0V through resistor R10. When Q5 is switched on, capacitors C4 and C5 charge, applying a 2µs long 500mA pulse to each Set/Reset strap. When Q8 is switched on, capacitors C4 and C5 discharge, applying a current pulse of opposite polarity to the straps.

We now calculate the energy/volt in ns-A needed to take a magnetic field sample, and to operate the straps. The bridge has a minimum resistance of 800Ω, which means that the maximum bridge current is (3.00V / 800Ω) 3.75mA. The instrumentation amplifier has a maximum supply current of 2.1mA, and the operational amplifier has a maximum supply current of 1mA. The reference divider has a maximum current draw of [3.0V / (1.1K + 8.2K)] 320µA. The voltage across the op-amp gain-setting resistors has a maximum value of (0.75V * ¾) 0.56V, so the maximum current through the gain setting resistors is [0.56V / (1K + 2K)] 190µA. Therefore, the maximum current drawn by each magnetic channel circuit is (3.75mA + 2.1mA + 1mA + 320µA + 190µA) 7.36mA.

The bridge and amplifiers cannot be turned on and off instantaneously. After a voltage step is applied to their power supply inputs, a finite time interval, called the settling time, must pass before an accurate reading is available at the output. The settling time is a result of the bandwidth limitations of the components in the signal chain.

Each component in the chain has a particular low-pass bandwidth; the one with the lowest bandwidth determines the settling time. In this signal chain, the component with the lowest bandwidth is the instrumentation amplifier, which has a bandwidth of 550 kHz. The sensor has a bandwidth of 5 MHz, the op-amp has a bandwidth of (15 MHz GBW / gain 2) 7.5 MHz, and the ADC SHA input network has a bandwidth of (1 / 2K * 42pF) 11.9 MHz. The instrumentation amplifier has a specified maximum settling time of 5μs to 0.1%. In practice, it is necessary to wait a little longer than 5μs, because one must wait for transients caused by the power up event to settle as well. The actual device leaves the sensors and amplifiers on for a total of 6.4μs. Thus, the sensors and amplifiers use (7.36mA * 6.4μs) 47ns-A to take a magnetic field sample.

A Set/Reset pulse consists of a 2μs-long 500mA pulse, followed by a 2μs 500mA pulse of opposite polarity. Thus, a set/reset operation uses (4μs * 500mA) 2μs-A.

A 0.5 gauss offset pulse draws 25mA. A 0.25 gauss offset pulse draws 12.5mA. A calibration operation requires two of these pulses, each for 6.4μs. Thus, providing the offset fields for a calibration operation uses (6.4μs * 25mA + 6.4μs + 12.5mA) 240ns-A.

Radio Transmitter

The node contains a radio transmitter, which it uses to transmit data to a base station. The radio transmitters used in the prototype nodes are HP-TX transmitter modules from Linx Technologies. [Linx00] The HP-TX is a narrowband, frequency-agile FSK data transmitter. It can hop under microprocessor control between eight frequencies in the 902-928 MHz ISM band[iii], and can transmit data at a maximum rate of 50kB/s. Power to the HP-TX is controlled by the microprocessor using a Siliconix Si2301 P-channel MOSFET.

The transmitter has a maximum specified current draw of 17mA. The transmitter module has an on-board PIC microcontroller, which it uses only for programming its LMX2316 PLL with the correct frequency. If the PLL were programmed directly by the node microcontroller, this part would not be necessary, and the transmitter current draw could be reduced to 11mA. Because it is necessary to wait for the PIC microcontroller's oscillator to start after turning on the transmitter module, the transmitter module has a 12 ms specified maximum time-to-availability. If the node microcontroller were used instead, this delay could be reduced to be less than 500μs.

The transmitter has a maximum data rate of 50kB/s, but we chose to transmit data at 38.4kB/s, so that the data could be received using a standard UART[iv]. At 38.4kb/s, each byte, including its start and stop bits, lasts for 260μs.

The unmodified transmitter uses (17mA * 12ms) 204μs-A to start up, and (17mA * 260μs) 4.4μs-A per transmitted byte. However, since the unmodified transmitter wastes a significant amount of power on start-up, we chose to control the PLL directly with the node microcontroller. Thus, the node uses (11mA * 500μs) 5.5μs-A to start up, and (11mA * 260μs) 2.9μs-A per byte.

It should be noted that with these transmitters, it is necessary to send a garbage character at the start of each packet (commonly a zero-balanced "U") to force the PLL into capture mode. This character will not necessarily be received correctly, but transmitting it still requires power.

Antenna

The node uses a rectangular microstrip antenna to radiate RF from the transmitter. The antenna is fabricated as part of the node printed circuit board.

Microstrip antennas, fabricated on PC boards, are known for their low bandwidth, as the bandwidth of a microstrip antenna is determined by its thickness. A standard square microstrip antenna with a single shorting post, fabricated on .062" FR4 has an impedance bandwidth of about 1%. [Carver81, 20]

However, this application requires a bandwidth of 902-928MHz, and requires PC board fabrication for low cost and a compact form factor. To meet these design goals, we designed an antenna based on a relatively new design

[iii] In an actual large-scale deployment of this technology, one would petition the FCC for a private area of the spectrum at a similar frequency.

[Kapur99], which divides the antenna into two regions with a slot, increasing the impedance bandwidth at the expense of efficiency. Figure 2 shows the antenna dimensions, feed point, and shorting post locations. The antenna is fabricated onto layer one of a .062” four-layer panel of FR4 epoxy glass. The board has 12.5mil between layer one and layer two, 37mil between layer two and layer three, and 12.5 mil between layer three and layer four. Inside a rectangular region that extends 1 cm around the antenna (and underneath the antenna) layer two has no copper, and layer three is the ground plane. Outside that region, the board has a standard layer stack; with a layer two ground plane and a layer three power plane. The antenna is fed through a via (see Figure 2) by a 22-mil wide 50Ω microstrip transmission line on layer four. The antenna ground plane is connected to the system ground plane by another transmission line, which runs next to the feedline.

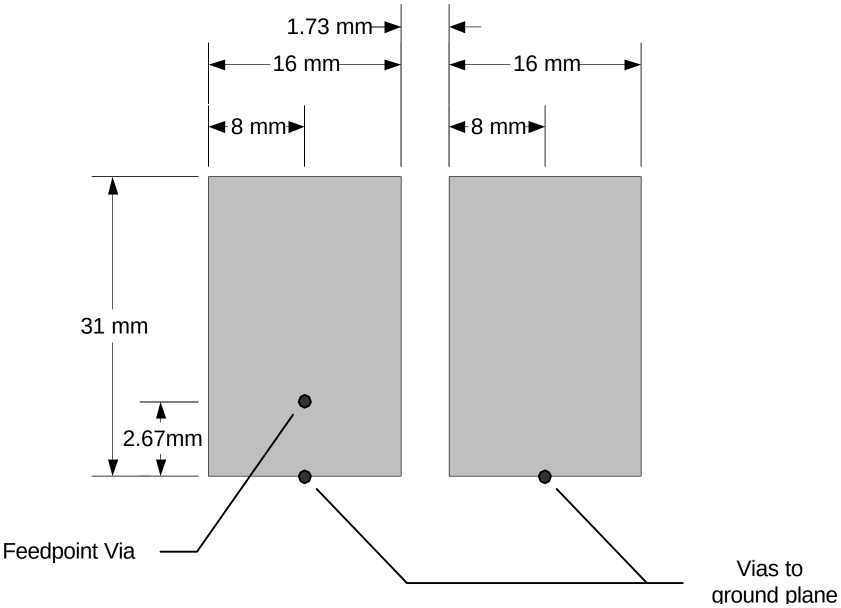


Figure 2: Microstrip Antenna

[iv] One could realize a small improvement in power-per-bit by transmitting data at 50 kb/s and using a receiver UART with a nonstandard bit clock.

Figure 3 and Figure 4 are the Smith Chart and VSWR plot for this antenna, taken with a HP8753D network analyzer. The wide loop on the Smith Chart at the resonant frequency is characteristic of this type of antenna, and verifies the functionality of the Q-spoiling slot. The antenna has a minimum VSWR[v] of 1.18 at 915.44 MHz, and reaches a VSWR of 3.00 at 903.00 MHz and 928.68 MHz. Thus, it has an impedance bandwidth of over 30MHz, roughly coincident with the U.S. 900 MHz ISM band. The antenna appears to have a gain of roughly –9dB relative to a quarter wave whip: however this figure was measured in a lab environment at close range.

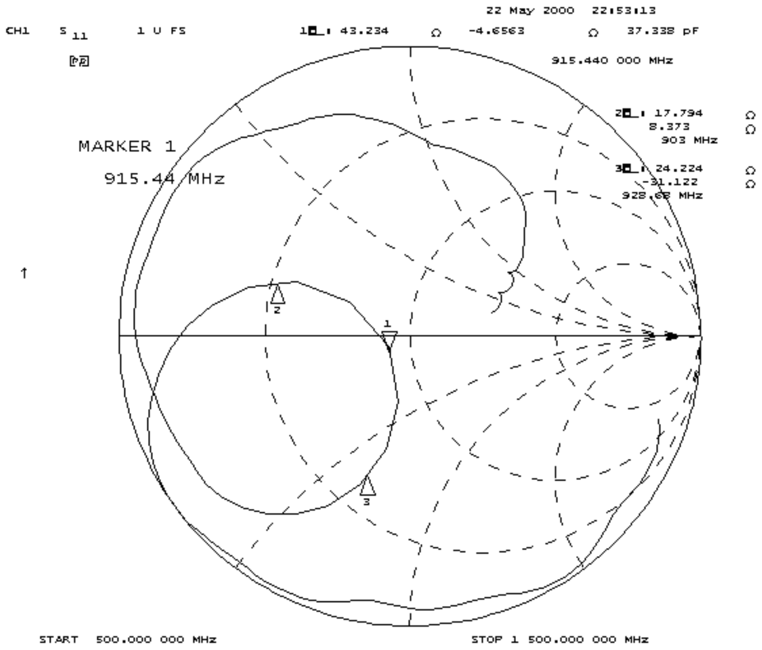


Figure 3: Antenna Impedance Plot

[v] The VSWR (Voltage Standing Wave Ratio) is a measure of the degree of impedance mismatch between a signal source and an antenna. A VSWR of one corresponds to a perfect impedance match. A transmitter-to-antenna VSWR less than two is excellent, and a VSWR less than three is still acceptable.

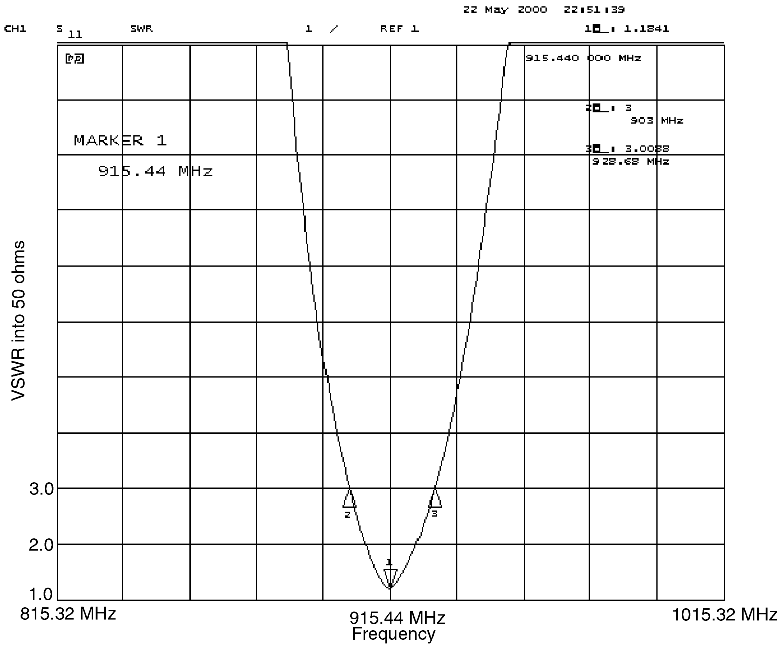


Figure 4: Antenna VSWR Plot

Temperature Sensor

The node contains a LM20 temperature sensor from National Semiconductor. [National99] The LM20 outputs a voltage inversely proportional to temperature. Power to the sensor provided by a port pin of the microprocessor. The output of the sensor is low-pass filtered and fed into one of the microprocessor's ADC inputs.

Microprocessor

Operations on each node are managed by a MSP430P325A mixed-signal microcontroller from Texas Instruments. [Texas00] The MSP430 series of microcontrollers are Von Neumann RISC machines with a versatile clock system. The processor is designed to spend most of its time in a low-power sleep mode, during which is draws only 2.8μA. In the low power mode, the peripherals (timers, counters, ports) continue to operate, deriving their clock from a 32.768 kHz watch crystal. When one of the peripherals throws an interrupt, the processor core and a frequency-locked-loop clock multiplier are activated, and, within 6µs, can execute code at up to 2 MHz. The processor includes a 12-bit successive-approximation A/D converter, which takes 96µs to perform a conversion as configured.

Most of the time, the processor is in LPM3[vi], in which the 32.768 kHz crystal oscillator and peripherals run, but the processor core is switched off. The maximum current draw in this mode is 2.8µA.

It takes 6µs to switch the processor from LPM3 to active mode. This is the maximum startup time of the FLL[vii] clock multiplier. During this time, the FLL may consume its maximum specified power, but the processor core is not running, so the device may draw a maximum of 130µA. Thus, switching the processor to active mode takes (6µs * 130µA) 0.8 ns-A.

The A/D converter requires 96 clock cycles to perform an A/D conversion, 12 of which are used for sampling, and 84 of which are used for the conversion. The A/D converter draws 400µA. During the first 12 cycles, the processor must be in active mode, so it can start the conversion, and turn off the sensors once the sampling gate has closed. During the last 84 cycles, the processor can be in LPM1, in which the processor core is switched off but the FLL continues to operate. The manufacturer only specifies the current draw for the A/D converter with a 1 MHz clock, so we chose to operate it and the system at this frequency. In active mode, the processor draws 500µA, and in LPM1, the processor draws 130µA. To latch a sample, the processor uses (12µs * 500µA + 400µA) 11ns-A. To finish the conversion, the processor uses (84µs * 130µA + 400µA) 44ns-A. Thus, to perform a complete conversion, the processor uses (11 + 44) 55ns-A. It should be noted that it is possible to save a small amount of power by completing processing during the A/D conversion. However, the savings from this optimization are not considered in this analysis for simplicity. To perform a single cycle of processing, the processor uses (1µs * 500µA) 0.5ns-A.

[vi] Low-Power-Mode Three

[vii] Frequency-Locked-Loop. A frequency-locked-loop is an all-digital variant of a phase-locked-loop.

Battery

A lithium primary battery provides power to the device. The Tadiran TL-4935 lithium thionyl-chloride battery can supply 20µA at 3.6V for 90,000 hours, which is more than 10 years. It has the same diameter as a D cell, but is 1/6 the length. [Tadiran99] The battery is specified over most of the industrial temperature range, from -30°C to +70°C. Over this voltage range, the battery voltage may vary between 3.5V and 3.7V. Since the circuit is designed to operate at 3.0V (to reduce current draw and therefore power consumption) a 0.6V-drop silicon diode should be placed in series with the positive terminal of the battery. The resulting supply voltage range over temperature, 2.9V to 3.1V, is within the specification range of all system components.

Node Power Consumption Analysis

At 100 Hz, the node samples magnetic field with one sensor to count vehicles. This requires switching the processor to active mode, sampling the magnetic field, performing an A/D conversion, and doing 20 cycles of processing. This uses (0.8ns-A + 47ns-A + 55ns-A + 0.5*20ns-A) 113ns-A, adding (113 ns-A * 100 Hz) 11.3µA to the average current drain.

Once every 60 seconds, the node sends back a 30-byte long radio packet. This adds ((5.5µs-A + (2.9µs-A * 30)) * 1/60 Hz) 1.5µA to the system power drain.

At most once every 10 seconds, the sensor operates the magnetic field sensor straps and measure the speed of a vehicle. This requires strapping the sensor, performing a calibration (during which the sensor is sampled twice), then taking 100 samples using both sensors, spaced very closely in time. Each sample requires about 50 cycles of processing. This takes (2000ns-A + 102 * (47 ns-A + 55 ns-A + 50*0.5)) 15000ns-A, which adds (15000ns-A * 1/10Hz) 1.5 µA to the average current draw.

The microcontroller consumes a 2.8µA continuously, to keep the timer running. So, the maximum average node current drain is (11.3µA + 1.5µA +

1.5µA + 2.8µA) 17.1µA. Figure 5 is a breakdown of the node power consumption. The TL-4935 specified above can supply this amount of current for more than 10 years, with 3.9µA to spare.

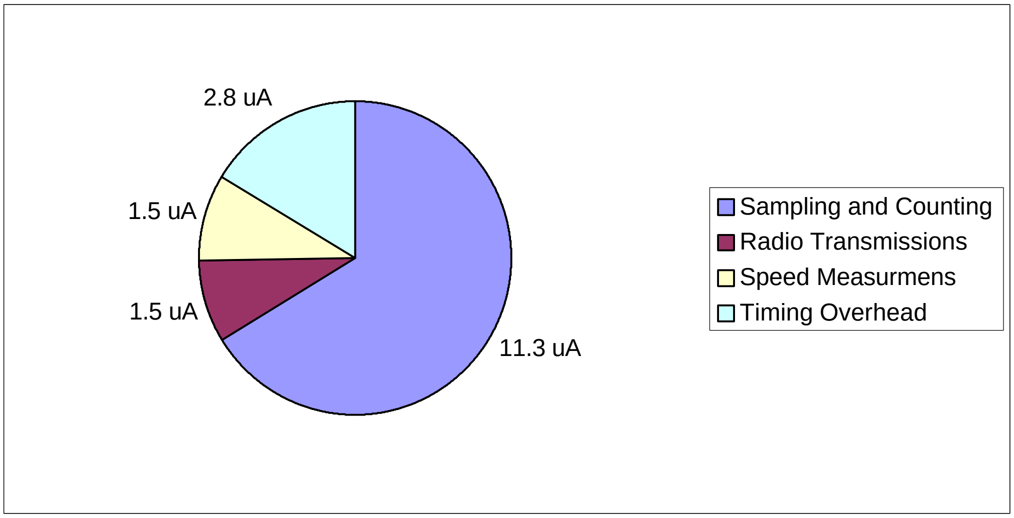


Figure 5: Node Power Consumption Breakdown

Node Cost

One of our design goals was to make the cost of a node low enough to make it economical to instrument every major roadway. Table 2 tabulates the cost per node. Instead of li
\sting the price of a HP-II-TX radio transmitter module in Table 2, we identified the components on it and added them to the bill of materials, since, in a realistic high-volume deployment, the transmitter module design would be integrated with the node design. We did not include the PIC microcontroller on the HP-II-TX transmitter module in Table 2, since it is only used to initialize the PLL, a function that could be performed by the main node microcontroller.

The prices listed per piece are manufacturer's budgetary prices, and standard industry estimates for a 1000-piece manufacturing run, so, the cost estimate given is very conservative. Indeed, if this circuit manufactured in million-piece volumes, most of the components would be combined into a few ASIC's[viii], resulting in substantial savings.

[viii] Application Specific Integrated Circuits

Quantity	Item	Each	Extended
1	TMS430 Microcontroller	$1.00	$1.00
1	32.768 kHz Oscillator	0.50	0.50
1	LMX2316 Digital PLL	2.50	2.50
8	RF Transistors	0.25	2.00
1	12 MHz Oscillator	0.50	0.50
2	AMR Magnetic Sensors	2.40	4.80
2	INA155 Instrumentation Amplifiers	1.00	2.00
2	OP162 Operational Amplifiers	1.50	3.00
7	Siliconix SI2301 Transistors	0.25	1.75
1	PTC Resettable Fuse	0.25	0.25
1	22µF Tantalum Capacitor	0.30	0.30
1	Printed Circuit Board	1.00	1.00
1	TL-4935 Lithium Battery	2.00	2.00
60	Misc. Discrete Components	0.005	0.30
1	Enclosure	1.00	1.00
90	Assembly (per component)	0.05	4.40
		Total	**$27.30**

Table 2: Cost of a node

Radio Transmission Protocol

Each node has a transmitter, but no receiver. This design choice was made primarily to cut the cost and complexity of each node, since the nodes do not have any intrinsic need to receive commands. However, the lack of a receiver complicates the radio protocol, since most standard radio multiplexing techniques require each participant to be able to receive as well as send. The protocol we designed is very simple: each node randomly selects a time slot within a 60-second interval and transmits there. If it detects that a vehicle is present when its randomly selected transmission time arrives, it waits for it to pass before transmitting. Redundancy and sparse use of the channel reduce the probability of collisions to an acceptable level.

The packet format described below calls for 7.8 ms long packets. A 7.8 ms long packet sent every 60 seconds occupies 1/7693 of the channel time.

For a sparse time-multiplexed network with n nodes and s timeslots, the probability of no collisions during a cycle is

$$\text{P(no collisions)} = \frac{C(s,n)}{C(n+s-1,n)}$$

since the numerator is the number of combinations that do not involve a collision, and denominator is the total number of combinations.

With n=10 per base station, there is a 1.2% chance of a collision per minute. The chance of a collision occurring during two consecutive 60-second intervals, assuming good random number generators in the nodes is (1.2% * 1.2%) 0.014%, so the expected time between any two consecutive collisions is about two hours. However, the expected time between collisions that cause data loss is greater than this, because the collisions would have to involve the same node.

To increase the number of nodes per base station, several frequencies can be used, with a frequency deterministically assigned to each node. For example, by using eight frequencies (the limit of the specified transmitter modules) there can be 80 nodes per base station.

Radio Packet Format

The radio transmitter used on the node requires an edge on the digital input data at least every 33ms; otherwise PLL may track out the modulation. So, each transmission requires a one-character preamble to take the PLL from lock to capture. This character may not be transmitted correctly, so it is not protected by the packet's CRC.

PLL Capture Character 'U'	Magic Number 'P'	Node ID (6 Bytes) (Big Endian)	Vehicle Counts 18 10-Second Bins (18 bytes, least to most recent)	Average Speed (1 Byte)	CCITT CRC-16 (2 Bytes) (Big Endian)

Figure 6: Packet Format

Figure 6 is a diagram of the packet sent by the node to the base station. The first character of the packet is a magic number[ix] to identify the system type and software version. The next six bytes are the node ID.[x] Each node has its own unique ID to simplify administration of the network. The last 18 10-second count bins follow, (so that even if one message is lost, the counts are not lost), followed by the average speed. The message concludes with a CCITT CRC-16.

The total message length (including the character to put the PLL into capture mode) is 30 bytes, which takes 7.8ms to transmit at 38,400 kb/s.

Vehicle Detection Algorithm

We developed the vehicle detection algorithm in a parking lot. We programmed a node prototype to stream magnetic field sample values from its radio interface in real time, and interpreted the stream on a laptop with a simple program written in Microsoft Visual Basic[xi]. Using Visual Basic made it easy to view strip charts of the data and processed versions of the data in real time, while also watching the car, and to prototype many algorithms in a short time. In particular, on the laptop, we could code algorithms using floating-point arithmetic and multiplication, with a full debugging environment.

[ix] A magic number is a byte arbitrarily selected by the designer, which is placed inside the packet. The use of a magic number helps to prevent devices being built by others that use the same packet length and checksum algorithm from being incorrectly accepted.

[x] A six byte ID allows for 2^{48} nodes. While this may seem excessive, 50 years of computer science has shown us that you can never have enough unique identifiers, so we chose to pick an identifier size that was several orders of magnitude larger than we anticipate might ever be required.

[xi] Microsoft Visual Basic is a trademark of Microsoft Corporation.

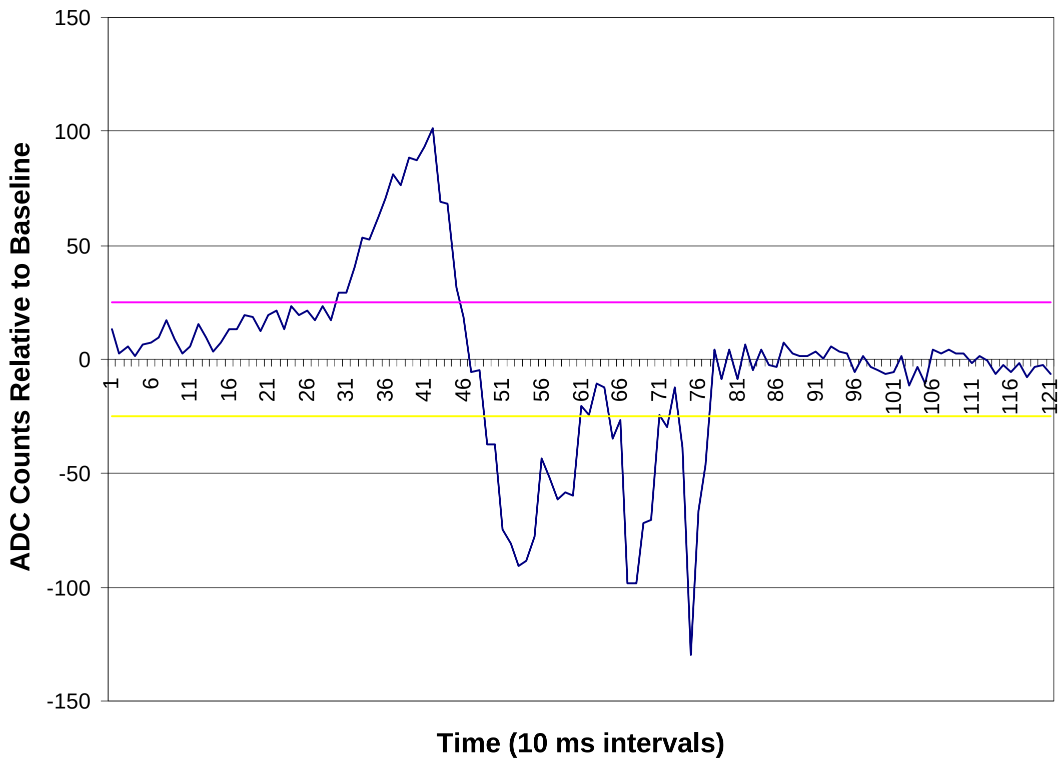


Figure 7: A Compact Car Driving Over the Sensor

Figure 7 shows the signal produced by a 4-door 1999 Volkswagen Golf driving forward over the sensor. The purple and blue horizontal lines are the thresholds used by the counting algorithm.

The counting algorithm uses the magnetic field value's deviation from a baseline to drive a state machine, which makes the counting decisions. To advance the state machine from the un-triggered state, the signal must deviate by more than a threshold value from the baseline. To advance the state machine from a partially triggered state to a count state, the signal must deviate by that same threshold on the other side of the baseline. After counting, once the signal comes within another (smaller) threshold, the state machine returns to the un-triggered state. If the state machine is in a half-triggered state, and comes within that threshold smaller for a long time (500ms) the state machine is also reset to the un-triggered state. The system may also return to a partially triggered state directly from a count state, if the signal deviates again to the opposite extreme.

The hysteresis introduced by this detection algorithm debounces vehicle counts in a way that works in fast-moving bumper-to-bumper traffic, and prevents running counts due to baseline drift and noise over the threshold. It also rejects large vehicles in adjacent lanes, since they do not cause a deviation in both directions.

When the sensor is powered up, it uses a 50/50 smearing filter[xii] to take an initial baseline for the first 10 seconds. Thereafter, the node adjusts the baseline using a very long time-constant low-pass filter, with a state-dependent time constant. When the state machine is un-triggered, the baseline is adjusted upwards by 1/10 of a count per sample if the signal is above it, and downwards by 1/10 of a count per sample if the signal is below it. When the state machine is in any other state, the baseline is adjusted by 1/100 of a count per sample. It takes the sensor about a half hour to return valid counts after being physically rotated through 90 degrees, which should only happen at installation. However, this long time constant insures that a vehicle sitting over the sensor at a stoplight is unlikely to change its baseline very much.

Test Results

We built a prototype node, and used it to log the number of vehicles using a road per minute over time. To facilitate rapid concept evaluation and improve testability, we made some slight changes to the design described above when building the prototype node:

1. The node firmware does not compute vehicle speed. (However, we did measure the speed of a screwdriver in bench tests using the streaming data link and an external computer, to validate our concept and algorithms.)
2. The node transmits one data packet once per second, and as such uses a simpler packet format, since there is currently no network. This change

[xii] I.e. baseline=(baseline + value)/2

allowed us to more readily verify that the sensor’s reported counts were correct.

3. The node uses a 3.0V flat lithium battery, whose form factor was more compatible with enclosures that we could readily install on a public street without having to drill holes.
4. We used an unmodified Linx transmitter module.

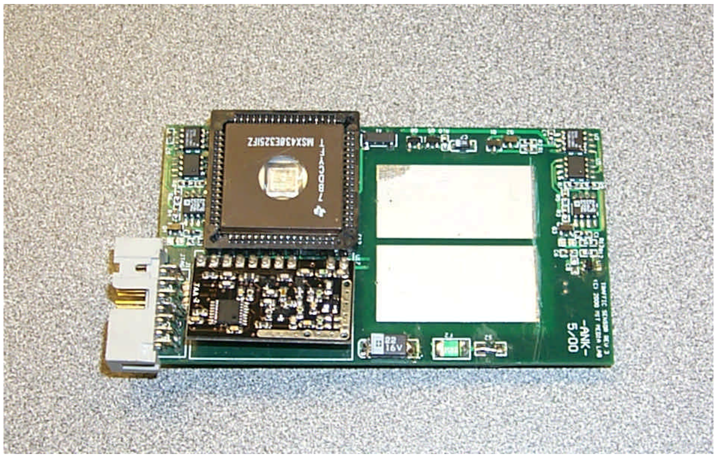


Photo 1: Node Circuit Board

Photo 1 shows the sensor circuit board. The large conductive areas on the top surface of the circuit board are part of the microstrip antenna. The large IC is the TMS430 microcontroller, and the connector is a JTAG header for in-circuit programming. The black circuit board is the HP-TX radio transmitter module. The two magnetic sensors and supporting circuitry are at the far left and right of the board. The strap control circuitry as at the far side of the board, and the PTC fuse and main bypass capacitor are at the near side.

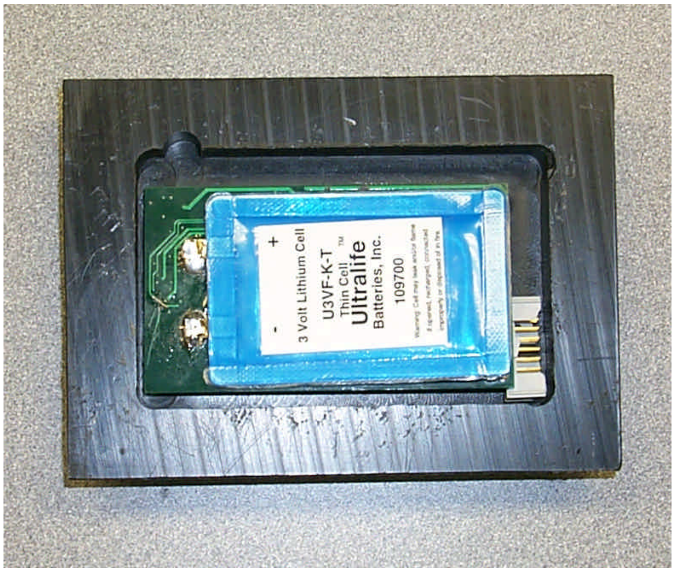


Photo 5: Node Circuit Board Inside Enclosure

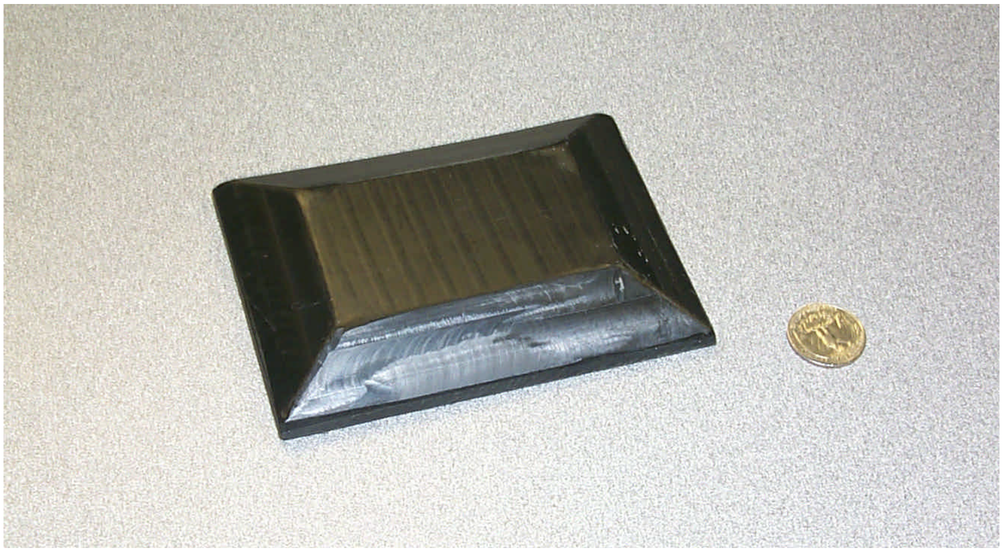


Photo 6: The Completed Node

Photo 5 shows the backside of the node circuit board, placed inside the enclosure. The large blue object is the flat lithium battery, which connects to pads on the circuit board with foil tabs. Figure 6 shows the completed sensor node. The enclosure was machined from a ¾" thick sheet of Dupont Delrin 570.

Photo 7: Vassar Street

We tested the sensor in a pothole (Photo 8) on the eastbound side of Vassar Street, (Photo 7) across from Building 44 on the MIT campus. Vassar Street carries a variety of traffic, from large tanker trucks to subcompact cars. Twenty-four hour public parking is allowed on both sides of the street. Vehicles typically travel at 35-40 MPH, but, during the morning, evening, and lunchtime rush, there is often bumper-to-bumper traffic and multiple light-cycle delays.

Photo 8: The Testing Site

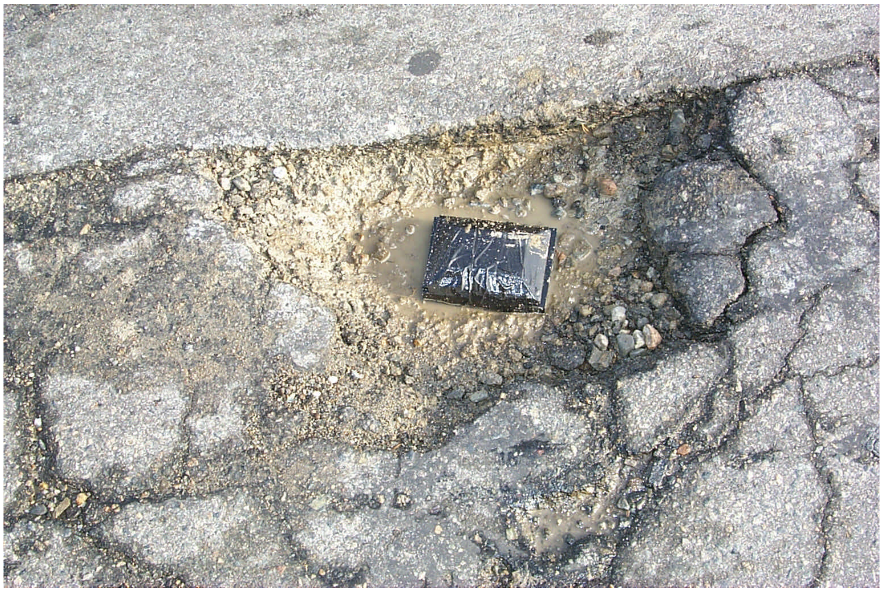


Photo 9: The Installed Node

Photo 9 shows the node inside the pothole. The node is wrapped with black waterproof duct tape to prevent water and dirt from contacting the electronics. The node enclosure was designed for mounting on a flat, rigid surface, but the pothole was filled with deeply embedded irregularly shaped rocks. We attached a ¾” thick sheet of Delrin to the bottom of the node to prevent the rocks from damaging the node electronics when the node was run over by a tire. While installed, the node sustained several direct hits by the tires of very heavy trucks, but was undamaged.

We set up an improvised base station in a parked car to collect data from the node. A laptop computer displayed the current temperature, count, and baseline value reported by the sensor, and logged these values to disk with a timestamp once a minute.

Visual observation confirmed that the sensor worked as expected. Generally, it counted vehicles that drove over it, and did not count vehicles that swerved around the pothole (a small fraction) or that were traveling in other lanes. The sensor double-counted certain types of large trucks, and occasionally (perhaps once every twenty) did not count a vehicle. The sensor performed just as well in bumper-to-bumper traffic as it did in widely separated traffic.

Indoors, the radio and antenna worked very well, and we were able to receive data from the node with almost no packet loss, from hundreds of feet away. Outdoors, radio performance was unremarkable, and we were only able to receive about one in three packets without error, even when the receiver was much closer to the node than it had been indoors. Our building's construction severely attenuates radio waves above 100 MHz, so we suspect that the problems we experienced during outdoor operation were caused by interference from other users of the 900 MHz ISM band. However, this interference did not prevent us from getting accurate data, because the packets were sent redundantly and contained checksums.

Figure 5 is a graph of vehicles per unit time passing the sensor during morning rush hour on May 22, 2000, from 6:20 AM to 8:30 AM. The X-axis demarcates 10-minute intervals, which ended at the marked time. The Y-axis shows the number of vehicles detected by the sensor during that 10-minute interval. During the period from 6:00 to 8:00, traffic moved smoothly, increasing in rate as 6:30 and 8:00 approached. Around 8:00, a traffic jam developed, decreasing the flow rate along the road. This traffic jam continued past 9:00, when we stopped taking data.

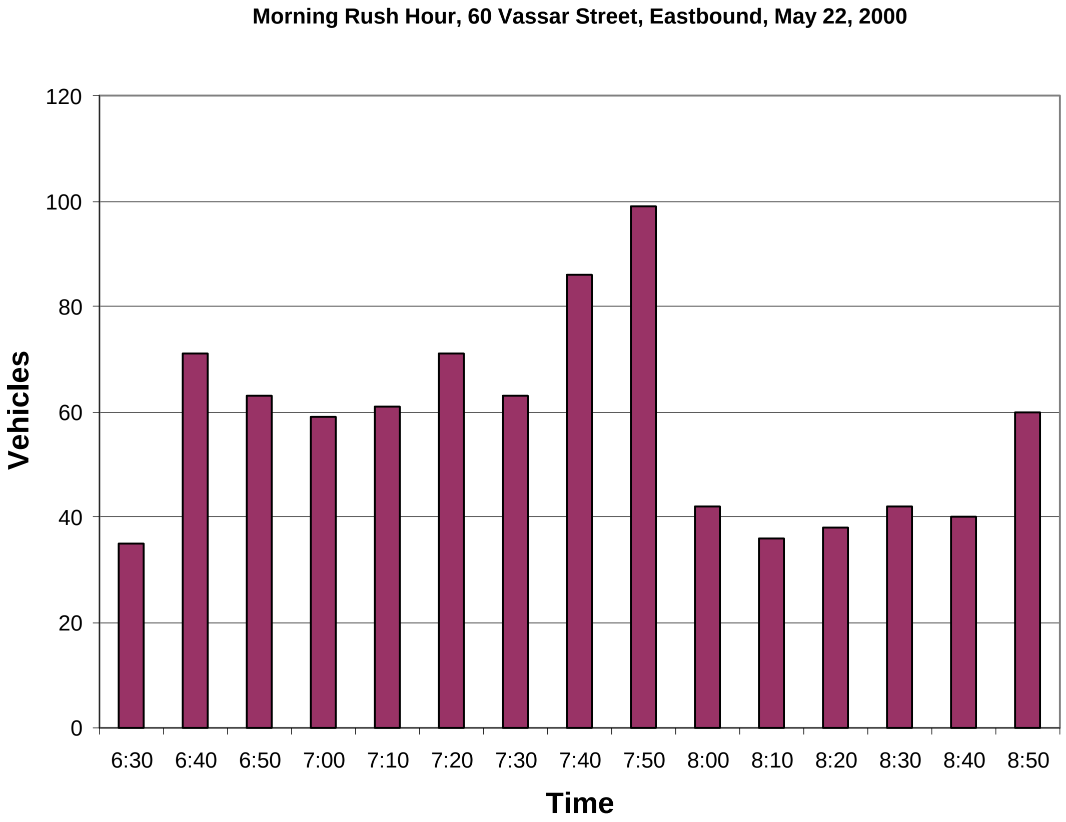


Figure 8: Morning Rush Hour; Vehicle Counts per 10-Minute Interval

Conclusion

The in-road wireless traffic sensor appears to be a viable technology for Intelligent Transportation Systems. Wireless traffic sensors can be installed for a much lower cost than almost any similar technology, and provide accurate traffic data that can be used to forecast road speeds and usage in real time.

Some aspects of this technology need to be more fully developed before it can be commercially deployed: In particular, radio performance needs to be characterized and improved, ultimately enough so that the node can transmit through thin layers of salt water, snow, and dirt. Making use of more efficient channel coding with forward error correction, like Trellis Coding, would increase radio performance, and result in lower radio power consumption. Battery performance also needs to be validated over the incredible range of temperatures experienced on a roadway. The capacitive road-condition sensing and vehicle speed detection subsystems need to be fully implemented and tested. The simple vehicle detection algorithm that we used performed well, but a more complicated algorithm might perform better. Finally, power consumption could be reduced by a factor of four by using components specified for 1.8V operation, and using a switched-capacitor voltage regulator during periods of high current draw.

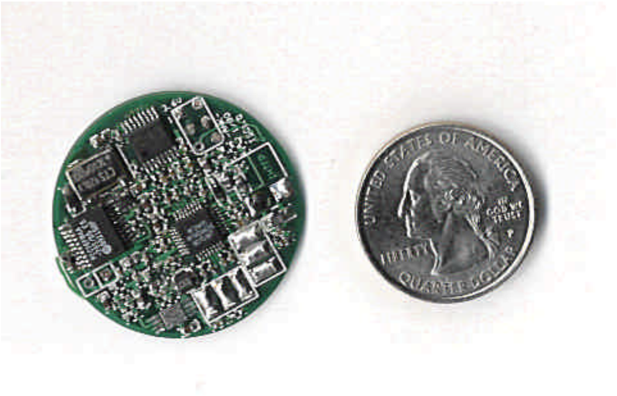


Photo 10: A Miniature Digital Radio Transceiver

This technology could also benefit from miniaturization. As part of an earlier prototype of a sensor node, we built a complete radio data transceiver that was only slightly larger than a quarter. (Photo 10) Ultimately, one could build complete sensor node as a chip, with an antenna printed on the top surface of the package, and battery printed on the bottom surface. If a higher radio frequency was used (to allow a smaller antenna) and power consumption was reduced by a factor of four (to allow a smaller battery), these nodes could be about the size of a dime. Even with no optimization over the current design (except packaging and antenna substrate) they could be the size of a silver dollar. They would cost only a few dollars to produce, and could be installed by dropping them into cooling road tar, of the type used to patch cracks after the spring thaw.

Acknowledgments

This research was conducted at the MIT Media Lab in the Responsive Environments Group, which is directed by Dr. Joseph Paradiso. This research was funded by the Car Consortium, CC++, which is directed by Betty Lou McLanahan.

Thanks to Pat Turner, whose sardonic insights into various vendors' invoicing practices always put a smile on my face. Thanks to Neil Gershenfeld, for allowing me to use his lab when I needed RF test equipment or fine-pitch SMT assembly equipment.

Thanks to Ari Adler and Ari Benbesat, the other two members of the "Ari, Ari, and Ara" office, for their mechanical and electronic advice, and for the gummy bears and food alerts. Thanks to Ashwini Deshpande, for using the soldering skills she learned at NASA to solder together some radio boards with 0402 discrete components. Thanks to Matt Reynolds for his advice about matters RF and analog, and to Yael McGuire, for an RF layout lesson. Thanks to Matt Hancher, for scoring me some PIC16C873's when Digi-Key was listing a 12-week lead time, and to Holly Gates, for giving me some capacitors during a world-wide shortage. Thanks to Linda Kiley, for doing the design calculations for the sensor node housings, machining two prototypes, and for bringing me coffee while I was taking data on Vassar Street.

Finally, thanks to my parents: Thanks to my mother, for teaching me to look at the world scientifically and to ask questions. Thanks to my father, for teaching me how to keep complexity under control, how to write computer programs, and how to solder.

References

[Caruso99] M. Caruso, L. Withanawasam, "Vehicle Detection and Compass Applications using AMR Magnetic Sensors," Honeywell Solid State Electronics Center, 12001 State Highway 55, Plymouth, MN

[Schrank99] D. Schrank, T. Lomax, "The 1999 Annual Mobility Report," Texas Transportation Institute, Texas A&M University System

[DOT99] “National ITS Architecture Documents: Executive Summary; U.S Department of Transportation,” EDL #5388, U.S. Government Printing Office

[USW00] U.S. Wireless Corporation web page, http://www.uswcorp.com

[Tadiran99] TL-4935 Data Sheet, Tadiran U.S. Battery Division, 2 Seaview Blvd., Port Washington, NY

[Palen97] J. Palen, “The Need For Survaillance in Intelligent Transportation Systems, Part II,” Intellimotion, Volume 6, Number 2, 1997, California PATH

[Beymer97] D. Beymer, P. McLauchlan, B. Coifman, J. Malik, “A Real-Time Computer Vision System for Measuring Traffic Parameters, In Proc. Computer Vision and Pattern Recognition, 1997, pp. 495-501, Association for Computing Machinery

[Sun99] C.Sun, “Intelligent Surveillance Using Inductive Vehicle Sensors,” Intellimotion, Volume 8, Number 3, March 1999, California PATH

[Kapur99] R. Kapur, G. Kumar, “Hybrid-coupled shorted rectangular microstrip antennas,” IEEE Electronics Letters, Volume 35, Number 18, September 1999, pp. 1502-1502
[Carver81] K. Carver, J. Mink, “Microstrip Antenna Technology,” IEEE Transactions on Antennas and Propagation, Volume AP-29, Number 1, pp. 2-24, January 1981

[Honeywell99] “HMC1021 Data Sheet Rev A,” Honeywell Solid State Electronics Center, 12001 State Highway 55, Plymouth, MN

[Burr99] “INA155 Data Sheet Rev B,” PDS1529-B, Burr-Brown Corporation, International Airport Industrial Park, 6730 S. Tuscon Blvd., Tuscon AZ

[Siliconix98] “SI2101DS Data Sheet Rev D,” 70627, Vishay Siliconix, 2201 Laurelwood Road, Santa Clara, CA

[Linx00] “HP-II Series Transmitter Module Design Guide,” Linx Technologies, 575 S.E. Ashley Place, Grants Pass, OR

[National99] “LM20 2.4V, 10µA, SC70 micro SMD Temperature Sensor,” DS100908, National Semiconductor Corporation, Santa Clara, CA

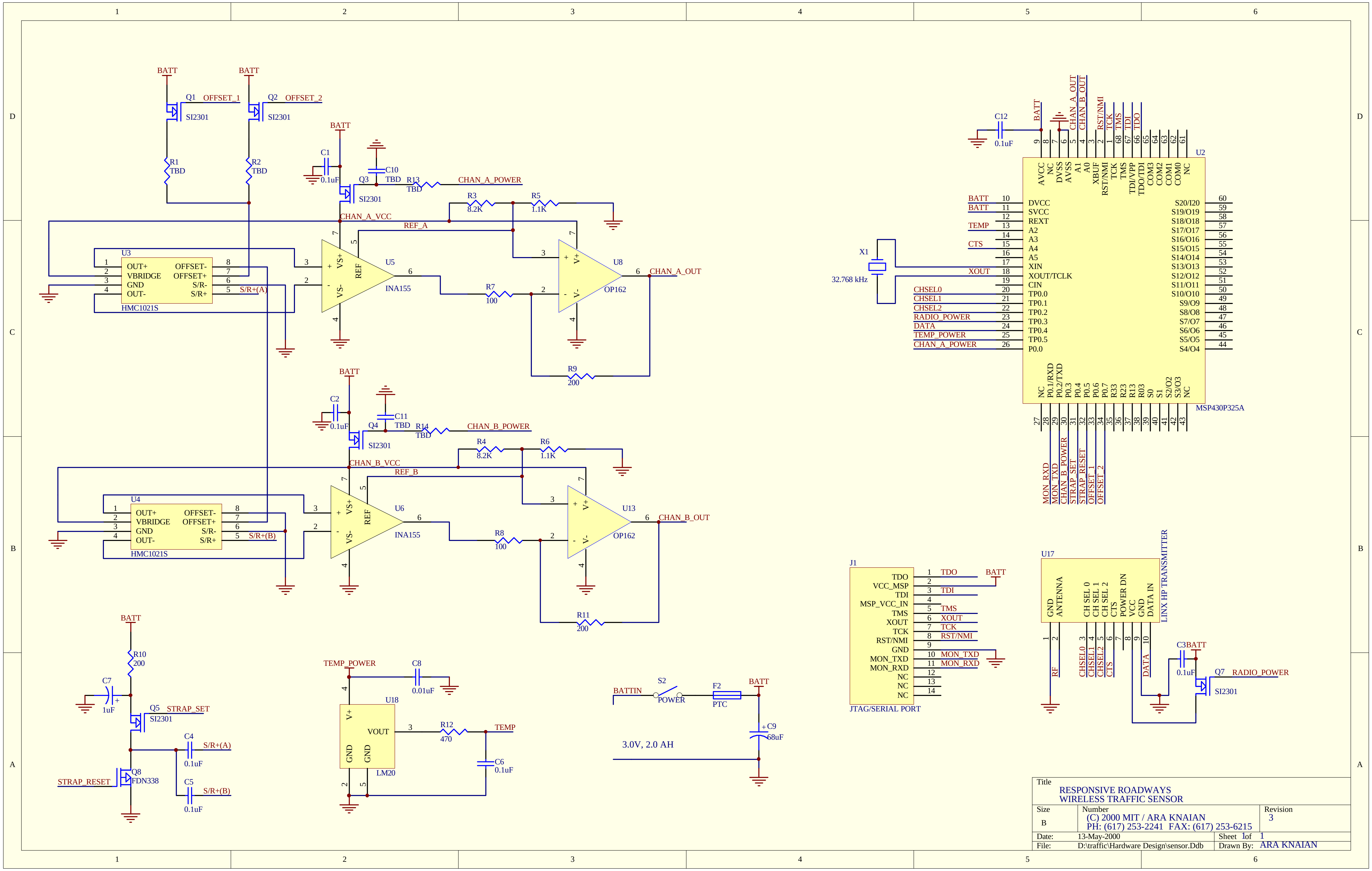
[Texas00] “MSP430P325A Data Sheet,” SLAS219B, Texas Instruments, P.O. Box 655303, Dallas, TX

[Graham-Rowe98] D. Graham-Rowe, “Danger, Hazard ahead! The studs that mark out road lanes at night are getting smarter,” New Scientist, Volume 160, Number 2160, November 14, 1998

[Barlow98] W. Barlow, E. Campbell, R. Crockett, “Sensing of Moisture Content in Soil,” U.S. Patent #3,803,570

[Smith99] J. Smith, “Electric Field Imaging,” Doctoral Thesis, Program in Media Arts and Sciences, Massachusetts Institute of Technology, Cambridge, MA

Appendix A: Sensor Schematic


BATT
Q1 OFFSET_1 SI2301
Q2 OFFSET_2 SI2301
R1 TBD
R2 TBD
C1 0.1uF
C10 TBD
Q3 SI2301
R13 TBD CHAN_A_POWER
R3 8.2K
R5 1.1K
CHAN_A_VCC
REF_A
U3 HMC1021S
OUT+ VBRIDGE GND OUT- OFFSET- OFFSET+ S/R- S/R+
S/R+(A)
U5 INA155
VS+ VS- REF
R7 100
U8 OP162
V+ V-
CHAN_A_OUT
R9 200
C2 0.1uF
C11 TBD
Q4 SI2301
R14 TBD CHAN_B_POWER
R4 8.2K
R6 1.1K
CHAN_B_VCC
REF_B
U4 HMC1021S
S/R+(B)
U6 INA155
R8 100
U13 OP162
CHAN_B_OUT
R11 200
R10 200
C7 1uF
Q5 SI2301 STRAP_SET
C4 0.1uF S/R+(A)
C5 0.1uF S/R+(B)
Q8 FDN338 STRAP_RESET
TEMP_POWER
C8 0.01uF
U18 LM20
V+ VOUT GND GND
R12 470 TEMP
C6 0.1uF
BATTIN
S2 POWER
F2 PTC
C9 68uF
3.0V, 2.0 AH
C12 0.1uF
X1 32.768 kHz
U2 MSP430P325A
AVCC NC DVSS AVSS A1 A0 XBUF RST/NMI TCK TMS TDI/VPP TDO/TDI COM3 COM2 COM1 COM0 NC
CHAN_A_OUT CHAN_B_OUT RST/NMI TCK TMS TDI TDO
DVCC SVCC REXT A2 A3 A4 A5 XIN XOUT/TCLK CIN TP0.0 TP0.1 TP0.2 TP0.3 TP0.4 TP0.5 P0.0
TEMP CTS XOUT CHSEL0 CHSEL1 CHSEL2 RADIO_POWER DATA TEMP_POWER CHAN_A_POWER
S20/I20 S19/O19 S18/O18 S17/O17 S16/O16 S15/O15 S14/O14 S13/O13 S12/O12 S11/O11 S10/O10 S9/O9 S8/O8 S7/O7 S6/O6 S5/O5 S4/O4
NC P0.1/RXD P0.2/TXD P0.3 P0.4 P0.5 P0.6 P0.7 R33 R23 R13 R03 S0 S1 S2/O2 S3/O3 NC
MON_RXD MON_TXD CHAN_B_POWER STRAP_SET STRAP_RESET OFFSET_1 OFFSET_2
J1 JTAG/SERIAL PORT
TDO VCC_MSP TDI MSP_VCC_IN TMS XOUT TCK RST/NMI GND MON_TXD MON_RXD NC NC NC
U17 LINX HP TRANSMITTER
GND ANTENNA CH SEL 0 CH SEL 1 CH SEL 2 CTS POWER DN VCC GND DATA IN
RF CHSEL0 CHSEL1 CHSEL2 CTS DATA
C3 0.1uF
Q7 SI2301 RADIO_POWER

Title
RESPONSIVE ROADWAYS
WIRELESS TRAFFIC SENSOR

Size	Number	Revision
B	(C) 2000 MIT / ARA KNAIAN PH: (617) 253-2241 FAX: (617) 253-6215	3

Date: 13-May-2000 Sheet 1of 1
File: D:\traffic\Hardware Design\sensor.Ddb Drawn By: ARA KNAIAN

Appendix B: Firmware

```
;*******************************************************************
;*                                                                 *
;*           RESPONSIVE ROADWAYS TRAFFIC SENSOR FIRMWARE          *
;*                                                                 *
;*          COPYRIGHT(C) 2000  ARA KNAIAN   MIT MEDIA LAB          *
;*                                                                 *
;*******************************************************************
;tsc.asm Revision 01
;

;*** Set this variable to '1' for the use on the Simulator ***
;SIM       .set   0           ;1 = Simulator
                             ;0 = STK/EVK

;RAM_orig  .set   00230h      ; Free Memory startadress

;          .if SIM=0
;USER_END  .set   003FFh      ; RAM endadress-31 STK/EVK
;counts    .set   0FFFFh      ; counts for STK/EVK
;          .else
;USER_END  .set   0FFFFh      ; RAM endadress-31 Simulator
;counts    .set   0FFh        ; Counts for Simulator
;          .endif



; Memory Map

RAM_orig   .set   00240h      ; Start of on-chip RAM (data)
SP_orig    .set   003DEh      ; User Stack
ROM_orig   .set   0C000h      ; Start of ROM area (code)
USER_END   .set   0FFFFh      ; End of ROM area, end of interrupt vector table




;--- Control register definitions

IE1             .equ    0h
IE2             .equ    01h
IFG1            .equ    02h
IFG2            .equ    03h
ME1             .equ    04h
ME2             .equ    05h

P0IN            .equ    010h
P0OUT           .equ    011h
P0DIR           .equ    012h
P0IFG           .equ    013h
P0IES           .equ    014h
P0IE            .equ    015h

P0IN0           .set    001h

LCDM            .equ    030h
LCD1            .equ    031h

BTME            .set    080h     ; BT module enable
BTIE            .set    080h     ; BT intrpt enable
BTIFG           .equ    080h     ; BT intrpt flag
P0IE0           .set    004h     ; P0.0 intrpt enable
P0IFG0          .set    004h     ; P0.0 intrpt flag
P0DIR0          .set    001h     ; P0.0 direction
P0IES0          .set    001h     ; P0.0 edge select
BTCTL           .equ    040h     ; BT control
TCCTL           .equ    042h     ; Address of Timer/Counter control register
TCPLD           .equ    043h     ; Address of Timer/Counter pre-load register
TCDAT           .equ    044h     ; Address of Timer/Counter

SCFI0           .equ    050h
SCFI1           .equ    051h
```

```
SCFQCTL         .equ    052h

WDTCTL          .equ    0120h
WDTHold         .equ    80h
WDT_wrkey       .equ    05A00h

GIE             .equ    08h

TPD            .equ   04Eh
TPE            .equ   04Fh


; Hardware Definitions for the ADC

AIN             .equ    0110h ; Input Register (For Digital Inputs)
AEN             .equ    0112h ; 0: Analog Input   1: Digital Input
ACTL            .equ    0114h   ; ADC Control Register
ADAT            .equ    0118h   ; ADC Data Register (12 or 14 bit)

ADIFG           .equ    04h   ; ADC End-of-Conversion Bit (IFG2.2)
ADIE           .equ   02h     ; ADC Interrupt Enable Bit

CS      .equ   01h            ; Conversion Start
VREF    .equ   02h            ; 0: Ext. Reference   1: SVCC On
A0      .equ   00h            ; Input A0
A1      .equ   04h            ; Input A1
A2      .equ   08h            ; Input A2
CSA0    .equ   00h            ; Current Source to A0
CSA1    .equ   40h            ; Current Source to A1
CSOFF   .equ   100h           ; Current Source Off
CSON    .equ   000h           ; Current Source On
RNGA    .equ   00h            ; Range Select (0 ... 0.25SVCC)
RNGB    .equ   200h           ; Range Select (0.25 ... 0.50SVCC)
RNGC    .equ   400h           ; Range Select (0.50 ... 0.75SVCC)
RNGD    .equ   600h           ; Range Select (0.75 ... SVCC)
RNGAUTO .equ   800h           ; 1: Range Selected Automatically
PD      .equ   1000h          ; 1: ADC Powered Down

; Address Definitions

MON_TIMER_VECTOR .equ   03E2h

; Defines

strap_cycles           .equ   100
radio_cycles           .equ   100
radio_spin_time               .equ   20000
upper_trigger_offset   .equ   25
lower_trigger_offset   .equ   25
upper_noise_offset     .equ   10
lower_noise_offset     .equ   10

magic_1                .equ   'P'
magic_2                .equ   'F'

zero_timeout           .equ   50

not_triggered          .equ   0
waiting_for_dip               .equ   1
waiting_for_rise       .equ   2
triggered_on_dip       .equ   3
triggered_on_rise      .equ   4

        ;; Port 0

CHAN_A_POWER   .equ   00000001b
CHAN_B_POWER    .equ    00001000b
STRAP_SET       .equ    00010000b
STRAP_RESET    .equ   00100000b
OFFSET_1       .equ   01000000b
OFFSET_2       .equ   10000000b
```

```
        ;; Port TC

CHSEL0         .equ   00000001b
CHSEL1         .equ   00000010b
CHSEL2         .equ   00000100b
RADIO_POWER    .equ   00001000b
DATA           .equ   00010000b
TEMP_POWER     .equ   00100000b

        ;; Analog Inputs

CHAN_A_OUT     .equ   A1
CHAN_B_OUT     .equ   A0
TEMP_OUT       .equ   A2

;************************************************************************
; Reset : Initialize processor
;************************************************************************

          .sect "MAIN",ROM_orig
RESET
          MOV    #SP_orig,SP                  ; initialize stackpointer
          MOV    #(WDTHold+WDT_wrkey),&WDTCTL ; Stop Watchdog Timer

        ;; Setup Basic Timer 0
        ;; This is the main system timebase.  It generates interrupts at 128 Hz.

        ;; Input Select = ACLK
        ;; Hold = 0   (?)
        ;; Input Divide = 1
        ;; LCD Frequency = ACLK/256
        ;; Interrupt Frequency = Fclk2/256

        MOV.B #00011111b, BTCTL

        ;; Initialize port pins

        BIS.B  #11111001b,&P0DIR      ; Setup Port 0 direction
        BIC.W  #00FFh, &AEN           ; All Port A pins are analog inputs
        BIS.B  #00111111b, &TPE              ; All Timer/Port TP pins are outputs

        ;; Set default values for port pins  -- everything powered down

        BIS.B  #(CHAN_A_POWER|CHAN_B_POWER|STRAP_SET|STRAP_RESET|OFFSET_1|OFFSET_2),
&P0OUT
        MOV.B  #RADIO_POWER, &TPD

        ;; Initalize Variables

        MOV.W   #1000, baseline_counter
        MOV.W   #0, vehicle_count
        MOV.B   #not_triggered, trigger_state
        MOV.W   #zero_timeout, zerocount
        MOV.W   #2000, baseline
        MOV.W   #2000, baseline_acc
        MOV.B   #strap_cycles, strap_counter
        MOV.W   #radio_cycles, radio_counter


        ;--- Enable Interrupts

        MOV.B   #0h,IE1
        MOV.B   #10000000b, IE2       ; Enable Basic Timer Interrupt
        CLR.B   IFG1
        CLR.B   IFG2

        ;--- Global Interrupt Enable

        BIS.W   #0000000000001000b, SR
```

```
;******************************************************************************
; Main Loop
;******************************************************************************
mainloop nop
        jmp mainloop

;******************************************************************************
; Basic Timer 0 Interrupt Service Routine
;******************************************************************************
timer_handler:
        ;;  First, we acquire data from the sensor.
        ;;  The A/D conversion starts by holding the sampling gate
        ;;  open for 12 cycles.  However, the passband of the RC filter that
        ;;  causes the settling time of the ADC SHA cap is much larger than
        ;;  the passband of the instrumentation amplifier---so we only need
        ;;  to make sure that a valid signal is present at the sampling gate
        ;;  when it is closed.

        ;;  This code leaves the sensor and amplifiers on for
        ;;  6.4 us.  The sampling gate closes just as the
        ;;  sensor is being turned off, but the system is
        ;;  synchronous, so this is fine.  You can see the sampling
        ;;  gate opening on the scope, since nothing is driving the
        ;;  sensor then.  The sampling gate closes 12 us later.

        ;;  Start A/D Conversion
        MOV #RNGA+CSOFF+CHAN_A_OUT+VREF+CS,&ACTL
        NOP
        NOP
        ;;  Power up sensors
        BIC.B   #(CHAN_A_POWER),&P0OUT
        NOP
        NOP
        NOP
        NOP                   ;  extra
        ;;  Power down sensors
        BIS.B   #(CHAN_A_POWER),&P0OUT

        ;; Spin wait for conversion to finish (change!)
waiting BIT.B   #ADIFG, &IFG2
        JZ      waiting
        ;;  Reset the flag to zero so it can trip next time
        BIC.B   #ADIFG, &IFG2

        MOV     &ADAT, R9

        call    #vehicle_detector

        ; Strap the sensor every 100 cycles

        DEC.B strap_counter
        JZ strap

        ; Run the radio every 500 cycles

        DEC.W radio_counter
        JZ radio

finished:

        RETI

strap:  MOV.B #strap_cycles, strap_counter

        BIC.B #(STRAP_RESET), &P0OUT
        ; Wait 1 us
        BIC.B #(STRAP_SET), &P0OUT
        ; Wait 5 us
        NOP
        NOP
        NOP
        BIS.B #(STRAP_SET), &P0OUT
        ; Wait 20 us
        MOV.B   #5, R6
dels:   DEC.B   R6
        JNZ dels
        BIS.B #(STRAP_RESET), &P0OUT

        JMP finished

radio:   MOV.W  #radio_cycles, radio_counter

        ;; power up radio and temp sensor

        MOV.B  #(5|DATA|TEMP_POWER), &TPD    ;Set radio for 915.37 MHz

        ;; wait 20 ms

        MOV.W   #radio_spin_time, R9
radio_spin:
        DEC.W   R9
        JNZ     radio_spin

        ;; get the temperature

        MOV.W   #RNGAUTO+CSOFF+12+VREF+CS,&ACTL

        NOP
        NOP
        NOP
        NOP
        NOP
        NOP

        ;; spin until conversion finishes
waiting4:
        BIT.B   #ADIFG, &IFG2
        JZ      waiting4
        ;;  Reset the flag to zero so it can trip next time
        BIC.B   #ADIFG, &IFG2

        MOV     &ADAT, temperature

        ; send the packet, protected with a checksum

        MOV.B   #0, checksum

        MOV.B   #'U', R8
        call    #tx_byte
        MOV.B   #'U', R8
        call    #tx_byte
        MOV.B   #'U', R8
        call    #tx_byte
        MOV.B   #'U', R8
        call    #tx_byte
        MOV.B   #magic_1, R8
        ADD.B   R8, checksum
        call    #tx_byte
        MOV.B   #magic_2, R8
        ADD.B   R8, checksum
        call    #tx_byte


        MOV.W   vehicle_count, R9
        SWPB    R9
        MOV.B   R9, R8
        ADD.B   R8, checksum
        call    #tx_byte
        SWPB    R9
        MOV.B   R9, R8
        ADD.B   R8, checksum
         call   #tx_byte
        MOV.W  temperature, R9
        SWPB   R9
        MOV.B  R9, R8
        ADD.B  R8, checksum
        call   #tx_byte
        SWPB   R9
        MOV.B  R9, R8
        ADD.B  R8, checksum
         call  #tx_byte


        MOV.W  baseline, R9
        SWPB   R9
        MOV.B  R9, R8
        ADD.B  R8, checksum
        call   #tx_byte
        SWPB   R9
        MOV.B  R9, R8
        ADD.B  R8, checksum
         call  #tx_byte

        MOV.B  checksum, R8
        call   #tx_byte

        ; power down radio and temp sensor

        MOV.B  #RADIO_POWER, &TPD

        ; done

        JMP    finished




; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; Transmit byte function (tx_byte)
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; This function assumes that the DATA_OUT pin is already in the idle line (high)
; state as an output, and that the transmitter is already powered up and the
; transmitting frequency is selected.
; It takes the byte in R8 and transmits it at 38.4 kb/s using NRZ serial.

tx_byte:
        PUSH.W R8
        PUSH.W R9
        PUSH.W R10

        ; Initialize the bit counter
        MOV.B  #8, R10

        ; Send start bit
        BIC.B  #DATA, &TPD

        MOV.B  #5,R9
wait12: DEC.B  R9
        JNZ    wait12
        NOP
        NOP
        NOP

        ; Send the data, MSB first

next_bit:
        RRC.B  R8
        JC     bit_high
bit_low:
        BIC.B  #DATA, &TPD
        NOP
        JMP     bit_count
bit_high:
        BIS.B   #DATA, &TPD
        NOP
        NOP
        NOP
bit_count:
        MOV.B   #4,R9
wait13: DEC.B   R9
        JNZ     wait13

        DEC.B   R10
        JNZ     next_bit

        ; Send the (high) stop bit

        NOP
        NOP
        NOP
        BIS.B   #DATA, &TPD

        MOV.B   #6,R9
wait14: DEC.B   R9
        JNZ     wait14
        NOP

        POP.W   R10
        POP.W   R9
        POP.W   R8
        RET

; Memory Allocation

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; Vechicle Detector
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;; This routine handles all the details of counting vehicles.
;; It takes an ADC value in R9.  It keeps track of the current
;; count in the vehicle_count variable.  External software can zero or
;; read this variable as required.

vehicle_detector:

        ;; check to see if we are taking an initial baseline
        CMP.W   #0, baseline_counter
        JNZ     integrate_baseline

        ;; adjust the baseline
        call    #adjust_baseline

        ;; compute trigger points
        MOV.W   baseline, trigger_upper
        ADD.W   #upper_trigger_offset, trigger_upper
        MOV.W   baseline, trigger_lower
        SUB.W   #lower_trigger_offset, trigger_lower
        MOV.W   baseline, noise_upper
        ADD.W   #upper_noise_offset, noise_upper
        MOV.W   baseline, noise_lower
        SUB.W   #lower_noise_offset, noise_lower

        ;; This implements a two-sided baseline deviation trigger with hysteresis.
        ;; Passing one trigger threshold followed by the other causes reset

        CMP.W   R9, trigger_upper            ; is trigger_set_upper < adc_value?
        JL      upper_zone
        CMP.W   trigger_lower, R9            ; is adc_value < trigger_set_lower?
        JL      lower_zone

        CMP.W   R9, noise_upper                         ; outside noise band?
        JL      dead_zone
        CMP.W   noise_lower, R9
        JL      dead_zone

        CMP.B   #not_triggered, trigger_state ; if inside the noise band, check state
        JEQ     done

        DEC.W   zerocount                     ; if triggered, run a timeout clock
        JZ      timeout_expired

done:   RET

dead_zone:
        MOV.W   #zero_timeout, zerocount
        JMP     done

timeout_expired:
        MOV.B   #not_triggered, trigger_state ; time has expired --- no car --- reset
state to untriggered
        MOV.W   #zero_timeout, zerocount
        JMP done

lower_zone:
        MOV.W   #zero_timeout, zerocount
        CMP.B   #not_triggered, trigger_state ; we're in the lower zone: check state to
see what to do
        JEQ     prime_lower_zone
        CMP.B   #triggered_on_rise, trigger_state
        JEQ     prime_lower_zone
        CMP.B   #waiting_for_dip, trigger_state
        JEQ     count_lower_zone
        JMP done
prime_lower_zone:
        MOV.B   #waiting_for_rise, trigger_state
        JMP     done
count_lower_zone:
        MOV.B   #triggered_on_dip, trigger_state
        INC.W   vehicle_count
        JMP     done

upper_zone:
        MOV.W   #zero_timeout, zerocount
        CMP.B   #not_triggered, trigger_state ; we're in the lower zone: check state to
see what to do
        JEQ     prime_upper_zone
        CMP.B   #triggered_on_dip, trigger_state
        JEQ     prime_upper_zone
        CMP.B   #waiting_for_rise, trigger_state
        JEQ     count_upper_zone
        JMP done
prime_upper_zone:
        MOV.B   #waiting_for_dip, trigger_state
        JMP     done
count_upper_zone:
        MOV.B   #triggered_on_rise, trigger_state
        INC.W   vehicle_count
        JMP     done

integrate_baseline:
        RRA.W   baseline                      ; this is a simple smearing filter: baseline
= (baseline + sample) / 2
        RRA.W   R9
        ADD.W   R9, baseline
        DEC.W   baseline_counter
        JMP done

; adusts the baseline based on state, so that it never gets too far off due to temp
shifts
adjust_baseline:
        CMP.B   #not_triggered, trigger_state
        JEQ     fast_adjust
```

```
slow_adjust:
        CMP.W  R9, baseline           ; is baseline < adc_value?
        JL     slow_adjust_less
slow_adjust_ge:
        SUB.W  #2, baseline_acc
        JMP    adjust
slow_adjust_less:
        ADD.W  #2, baseline_acc
        JMP    adjust


fast_adjust:
        CMP.W  R9, baseline           ; is baseline < adc_value?
        JL     fast_adjust_less
fast_adjust_ge:
        SUB.W  #20, baseline_acc
        JMP    adjust
fast_adjust_less:
        ADD.W  #20, baseline_acc


adjust: CMP.W  #1000, baseline_acc    ; is baseline_acc < 1000
        JL     adjust_less
        CMP.W  #3000, baseline_acc,   ; is baseline_acc > 3000
        JGE    adjust_greater
        JMP    done_adjust

adjust_greater:
        INC.W  baseline
        MOV.W  #2000, baseline_acc
        JMP    done_adjust

adjust_less:
        DEC.W  baseline
        MOV.W  #2000, baseline_acc
done_adjust:
        RET


; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; Variable Declarations
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *


        .sect  "Data", RAM_orig
        .data

        ;;  Words

        .bss   baseline_counter, 2, 0200h
        .bss   radio_counter, 2
        .bss   trigger_upper,2
        .bss   trigger_lower,2
        .bss   zerocount, 2
        .bss   noise_upper, 2
        .bss   noise_lower, 2
        .bss   baseline, 2
        .bss   baseline_acc, 2
        .bss   temperature, 2
        .bss   vehicle_count, 2

        ;; Bytes

        .bss   strap_counter, 1
        .bss   trigger_state, 1
        .bss   checksum, 1

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; Interrupt vectors
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

```
.sect   "Int_Vect",USER_END-31
.word   RESET
.word   timer_handler  ;  Basic Timer Interrupt
.word   RESET
.word   RESET
.word   RESET
.word   RESET
.word   RESET
.word   RESET
.word   RESET
.word   RESET
.word   RESET
.word   RESET
.word   RESET
.word   RESET
.word   RESET
.word   RESET             ; POR, ext. Reset, Watchdog

.end
```