Mobile Gateway for Wireless Sensor Networks utilizing drones

Ioan Deaconu, Andrei Voinescu
Automatic Control and Computers Faculty
University Politehnica of Bucharest,
{ioan.deaconu@cti.pub.ro, andrei.voinescu@cs.pub.ro}

***Abstract*—Mobile Gateway for Wireless Sensor Networks utilizing drones**

Wireless Sensor Networks have good uses in applications running in remote locations, where maintenance is difficult. Wireless sensor network islands often need a connection with a gateway in order to communicate with the outside world, but this cannot be provided in all circumstances. A different approach to connecting wireless sensor networks is to use mobile gateways that can reach the remote locations where sensor devices are running. Our design includes a gateway mounted on an unmanned aircraft that collects data on its flight path.

***Index Terms*—gateway, wireless sensor networks, drone**

I. Introduction

Wireless sensor network applications are on rise, largely thanks to the the internet of things, but developing the correct hardware for an application is not very easy, mainly due to the size of the nodes, that tends to be usually very small. This

In this paper, we will present a brand new powerful and yet low power platform that can be used to easily develop a new application of a wireless sensor network. We designed it as a development platform that lets the user to focus on what he would like to do with the platform and not how to do it.

This platform comes in a very small package, can be programmed using Arduino Studio and has a daughter board extension in order to be fully compatible with all Arduino hardware.

order to collect the data, gateway platforms [6] are

II. Related Work

Research has been done in the area of wireless sensor networks with mobile gateways, but none include the same constraints as in our networks. Where most research focuses on routing packets to a mobile sink, we consider applications where network lifetime and low-maintenance are more important, and as such energy consumption on the nodes must be kept to a minimum (in order to have 3+ years of network lifetime with no maintenance/battery changes). Shifting the energy consumption to the sink is a direct consequence of this decision, under the assumption that there exists a flight path that allows the gateway to collect information from all the nodes.

Two-Tier Data Dissemination[7], Line-Based Dissemination[3] are protocols with mobile sinks for very specific networks. TTDD assumes sensor platforms are arranged in a perfect grid pattern, while LBD establishes a communication pattern over a virtual infrastructure, derived from the position of the nodes. Nodes in both protocols do not have duty-cycles in their activity and have either fixed-pattern positions or have GPS or equivalent equipped. This is very different from the small, energy constrained sensor platforms we wish to use and the mechanisms used by these protocols are not applicable in our system.

Another type of mobile sink system involves proactive routing as a variation to the directed diffusion protocol[4], such as XYLS[2], which disseminates data reports in two geographically-opposite directions, such that it intersects a corresponding data query. This is once again a system which offers a query system to a mobile sink, but cannot offer the collection of all measured data since its last fly-by.

III. System architecture

A. Hardware

Because not all sensors are designed to run at 1v8 up to 3v3, we need to be able to dynamically change the operating voltage of the node. We used TPS62742, a step-down switching DC/DC converter with up to 95

The node can be conneted to an extension daughter board which respects 100

B. Software

The SAMR21 micro-controller is 95

Even thought the Arduino software is well designed, it was not designed with low power approach from the beginning. We will describe some of the problems encountered when trying to create the software stack.

The first problem noticed was that the Arduino Zero board had no sleep functionality implemented. The ideal idle power consumption should have been less than 5uA, tested and measured using a project created in Atmel Studio 7.0. The power consumption of the board was around 350uA. We dig deeper and discovered that the USB device was always initialized, witch accounted for the extra 200 uA. The rest of 150uA came from a default initializations of the pins as input pins, but this only lowered the power consumption to about 60uA. We kept searching for a cause, and discovered that the clock generators are never disabled at start-up, which accounted for about 30uA.

So far we managed to decrease the idle power consumption for the platform from 350uA to about 30uA @ 3v2, but still far from ideal. Surprisingly, lowering the voltage to 1v8 lead to a 3.3uA sleep power consumption and when examining the power trace using a digital oscilloscope, we found that for some reason, a clock remains active, leading which at 3v2 leads to high spikes in power consumption.

Event tough we did not reach the goal of 5uA, we still reached a respectable 30uA @ 3v2 and less than 4 uA @1v8. Due to the little time remaining and the need to use the nodes in order to implement and test new features, we decided that for now this is acceptable, and for future revisions, we will come back and find the wild clock source.

Even the run power consumption was not ideal, instead of achieving the promised 70uA/MHz @ 3v2, which would have lead to around 3.5mA power consumption of the CPU, the micro-controller consumed 8mA @ 48MHz. We managed to reduce the power consumption to 5.5mA @ 48MHz, due to clock optimizations presented bellow.

The Peripherals are run at a much lower clock, instead of 48 MHz, we run them at 12 MHz. Also if peripherals are not used, we completely disable them. Due to this, we ran into problems related toSERCOM implementation, a generic module that handle USART, SPI and I2C. It was working on Arduino Zero, because the CPU and the BUS were configured at the same speed, but due to some clock source modifications, the SERCOM did not set the correct speed. Also, there are 6 SERCOMs, and instead of enabling the clock for each one only when it is used, all of them were enabled, which lead to extra power consumption during run time.

We implemented in the platform architecture special modules, designed for low power like a sleep implementation, power management which can dynamical change the running voltage between 1v8 to 3v2. This allows the user to select which voltage is better required for applications. For example, some sensor must be run at exactly 2v8 or other run at 2v5 or higher. This module allows for precise voltage selection, use the sensor, and then switch to the lowest voltage in order to obtain the best power consumption possible.

The RF module is AT86RF233, integrated into the micro-controller. We integrated a module for the RF in the core of the platform, in order to let the user focus on what to do with the platform and not how to do something with it. For example, in case the RF is constantly running in receive mode for more than 5 minutes, than it is recommended to do Fine Tuning of the PLL clock in order to eliminate possible clock skews. Also, the module automatically receives a packet, saves it locally together with RSSI and LQI, which can be later read and used buy the user. The internal buffer is designed for 8 packets of 127KB of data, which amounts for 1 KB of ram. The buffer is cyclic, so in case the buffer is not read, the oldest data is discarded and replaced by a new one. This should not happen very often, because the buffer is large enough to handle all request, even for high bandwidth transfers.

In order to be able to save data locally, the micro-controller has an EEPROM like functionality which allows a 16KB region of flash to emulate flash write endurance. The flash contains pages of 64 bytes, and the EEPROM has an overhead of 4 bytes, which leaves 60 bytes for actual data. Also, for each page, another page must be reserved for further use. This means that out of 16KB used, the total amount of usable space left is (16*1024 / 2) * 60/64 = 7680 bytes. This should be more than enough for normal use. This approach rases the 25k cycles of flash write and erase to at least 100k, with typical values reaching 600k cycles. For the moment, the EEPROM zone is completely erased when programming the node.

For timekeeping when sleeping, RTC functionality was implemented. Besides keeping the time, RTC provides alarm interrupts for a special dates, which can be configured to be triggered every minute, every hour, every day, every month, every year, or only once, ever. Together with another peripheral EventSys, periodic interrupts are provided also, interrupts that can range from once very second up to 128 times per seconds, with increments of power base 2.

Because the software and hardware are not perfect, a watchdog functionality is also implemented, in order to avoid code lock-up or hardware failure due to extreme environment conditions.

IV. Implementation

A. Hardware Details

Figure 1. SparrowDongle connected to AR Parrot Drone 2.0

The hardware components used are the following:

- a SparrowDongle with two micro controllers: The 8-bit ATMega128RFA1 which has an on-chip 2.4GHz wireless transceiver and the ATMega32U4, both from Atmel.
- a SparrowV3.2 with an ATMega128RFA1 micro controller

- a AR Parrot Drone v2.0
- a SAMSUNG Galaxy S4 with android 4.4.2, but a laptop or other mobile phones with android/ios can be used for controlling the drone

Because of the addition of the SparrowDongle, the drone's polyester hull has had to been carved a little in order to accommodate it.

B. Software Implementation

The parrot drone, with a Linux based operating system, allow the software to be organized in different modules. The modules can be modified individually to add more features to the system.

The SparrowDongle gateway dumps every data received on the serial. It is always in a listen for data state. When it receives the data, it sends back an ack to let the SparrowV3.2 node know that it can begin sending the entire data to the mobile gateway.

The SparrowV3.2 node is sending periodically a small data packet to check if a gateway is available. When it receives the ack for the packet it starts sending the stored data to the gateway. The data sent can vary, from sensor readings, to debugging informations in order to check the state of the Wireless Sensor Network.

The data gathered by the gateway is saved into different files the AR Parrot Drone's internal memory. The file also contains informations about the node identification tag and time of the transfer. The data can be accessed at any time by any device connected to the drone's wireless network via FTP.

All the collected data is processed on the drone. It awaits a socket connection in order to start sending informations regarding the state of the connected nodes.

The nodes can be programmed to determine the signal strength of the surrounding nodes. This data is sent to the drone in order to determine the approximate location of the nodes[5].

V. Results

A. Performance

B. Hardware

C. Software configurations

VI. Future work

The versatility of the platform makes it ideal to be deployed in a wide range of applications, especially those of which environment is dynamic.

For instance, some possible applications in which the drone can be deployed are [1]:

- Autonomous collection of data data by adding a gps receiver to the drone and setting a path of way points to follow
- Debugging a wireless sensor network by checking which node is working, the connection logs and the physical state of the device
- Search and rescue operations, especially when going skiing in a avalanche prone environment by wearing a wireless sensor.
- Creating a small wireless sensor network for a limited time with small costs and large battery life
- Treasure hunt where the sensors can hold the clues that lead to the location of the treasure

VII. Conclusion

The paper presented an innovative mobile gateway platform for wireless sensor networks that offers the ability to interface sensor networks in a novel way.

The platform is easy-to-use and customize to meet the different needs of applications. The system is able to collect data from a Wireless Sensor Network island and offers a potential drone operator realtime data concerning nodes in proximity. The system uses the Parrot AR drone and is decoupled from the piloting module, allowing different piloting schemes to be used with it (i.e. the drone can be piloted anywhere, with or without an operator and it will collect data automatically).

References

[1] T. Arampatzis, J. Lygeros, and S. Manesis. A survey of applications of wireless sensors and wireless sensor networks. In *Intelligent Control, 2005. Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation*, pages 719–724. IEEE, 2005.

[2] S. M. Das, H. Pucha, and Y. C. Hu. Performance comparison of scalable location services for geographic ad hoc routing. In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 2, pages 1228–1239. IEEE, 2005.

[3] E. B. Hamida and G. Chelius. A line-based data dissemination protocol for wireless sensor networks with mobile sink. In *Communications, 2008. ICC'08. IEEE International Conference on*, pages 2201–2205. IEEE, 2008.

[4] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: a scalable and robust communication paradigm for sensor networks. In *Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 56–67. ACM, 2000.

[5] C. Savarese, J. M. Rabaey, and J. Beutel. Location in distributed ad-hoc wireless sensor networks. In *Acoustics, Speech, and Signal Processing, 2001. Proceedings.(ICASSP'01). 2001 IEEE International Conference on*, volume 4, pages 2037–2040. IEEE, 2001.

[6] L. Steenkamp, S. Kaplan, and R. H. Wilkinson. Wireless sensor network gateway. In *AFRICON, 2009. AFRICON'09.*, pages 1–6. IEEE, 2009.

[7] F. Ye, H. Luo, J. Cheng, S. Lu, and L. Zhang. A two-tier data dissemination model for large-scale wireless sensor networks. In *Proceedings of the 8th annual international conference on Mobile computing and networking*, pages 148–159. ACM, 2002.