Transiently Powered Wireless Embedded Systems

Kasım Sinan Yıldırım
Embedded Software Group, Delft University of Technology

December 9, 2016
4TU Embedded Systems Student Event
Battery-less Computers

Przemysław Pawełczak
Kasım Sinan Yıldırım
Amjad Yousef Majid

Michel Jansen
Koen Schaper
IoT – Wireless Embedded Systems

Sensing

Sensor

Computation

Microcontroller

Communication

Transceiver

Battery

Power

Transiently Powered Wireless Embedded Systems
Powering IoT

• Powering cyber-physical systems is a challenge
  — By 2025: >100 billion IoT devices
  — sustainable operation
  — large-scale deployment
• Batteries
  — increase weight, cost of the hardware
  — replenishment is generally impractical
  — ecological footprint
• Transfer of electromagnetic energy
  — from a power source to receiver devices over the air
  — wireless power transfer
Wireless Power Transfer (WPT) - I

- **Non-radiative techniques**
  - either inductive or magnetic resonant coupling
    - varying **magnetic flux** induces current
  - transfer power over **short** distances
Wireless Power Transfer (WPT) - II

- Radiative techniques
  - use the electric field of the electromagnetic waves
    - radio frequency (RF) waves as an energy delivery medium
  - transfer power over longer distances
  - provision of energy to many receivers simultaneously
    - broadcast nature
  - low complexity, size and cost for the energy receiver hardware
  - suitability for mobility
  - charge low-power embedded devices
    - RFID (Radio Frequency Identification) tags
RFID System Overview

Antenna

IC

Tiny memory

Energy Harvester

Backscatter Communication

- Transmitting data requires too much energy for a battery-free device
- Modulate the reflections of an incident RF signal
  - use the signal sent by the reader to communicate data back
  - not by generating radio waves

http://ieeexplore.ieee.org/xpls/icp.jsp?arnumber=4447347
RF-Powered Computing

• A new class of low-power battery-less embedded systems
  – Transiently Powered Computers (TPCs)

• CRFIDs (Computational RFIDs)
  – RFID technology as a foundation
  – Allow sensing, computation and communication without batteries
    • Charge a super capacitor using harvested rf energy
  – Equipped with a backscatter radio
    • simple circuitry for the receiver
    • allows communication to come almost for free

A CRFID platform: WISP - Wireless Identification and Sensing Platform
(University of Washington)

Ultimate goal: replacing existing battery-powered wireless sensor networks
WISP Hardware - Overview

RFID reader

Antenna

- Antenna gets RF signal
- Maximize power transfer to Power Harvester
- RF signal rectified into DC voltage
- Charge Supercapacitor
  - Get energy from supercapacitor
  - Processes bits
  - Polls sensors to gather data
- Transistor that changes antenna impedance for 'backscatter'

- Power Management
- Impedance Matching
- Demodulator
- Modulator
- Flash Memory
- TI MPS430 Microcontroller
- Temperature Sensor

Sensors and Peripherals

WISPCam: Battery-less Camera

- WISPCam - University of Washington

WISPCam captures a 160x120 low resolution image for face detection

Ambient Backscatter

- Traditional backscatter communication, (e.g. in RFID)
  - a device communicates by modulating its reflections of an incident RF signal - not by generating radio waves

- Ambient backscatter
  - Communicate using ambient RF signals as the only source of power
    - Ambient RF from TV and cellular communications

Vincent Liu et al. “Ambient Backscatter: Wireless Communication Out of Thin Air“, ISIGCOMM, August 2013
### Future...

<table>
<thead>
<tr>
<th></th>
<th>Active</th>
<th>Passive</th>
<th>Ideal</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Power Source</strong></td>
<td>Battery-powered</td>
<td>RF-powered (battery-free)</td>
<td>RF-powered (battery-free)</td>
</tr>
<tr>
<td><strong>Physical Operating Range</strong></td>
<td>Unlimited</td>
<td>Requires proximity to RF power source</td>
<td>Unlimited</td>
</tr>
<tr>
<td><strong>Lifespan</strong></td>
<td>Months to years</td>
<td>No fundamental limitation</td>
<td>No fundamental limitation</td>
</tr>
</tbody>
</table>

**Not Yet**
Programming Challenges
WISP tags vs Battery-powered Nodes - I

- Continuously **varying** voltage level
  - WSNs: stable voltage levels in the short term (battery-powered)
  - WISP: **fluctuating** input voltage

1Benjamin Ransford et al., "Mementos: system support for long-running computation on RFID-scale devices." Acm Sigplan Notices 47.4 (2012): 159-170.
WISP tags vs Battery-powered Nodes - II

• Frequent loss of computation state
  – frequently "die" due to power loss
    • need to save the computation state into the non-volatile memory
    • recover when they harvested sufficient energy to start up
    • saving computational state to non-volatile memory is also energy consuming

WISP tags vs Battery-powered Nodes - III

• The classical motto
  – “compute instead of communicate whenever possible”
  – No longer valid for the WISP platform
    • backscatter communication comes almost for free

• Intermittent power
  – lightweight methods in terms of computation are desirable
A Simple Program: Fibonacci

```
function fibonacci(n){
    a= 1;
    b= 1;
    while(i<=n){
        c = a + b;
        a = b;
        b = c;
    }
    return b;
}

main {
    i = 0;
    while(TRUE){
        fibonacci(i++);
    }
}
```

1, 1, 2, 3, 5, 8, 13, 21, 34, 55,..
TPCs – How?

• How to **design** programs under the effect of frequent power interruptions?
  – How to ensure
    • **Consistency** of the non-volatile memory?
    • **Correctness** of the program?
  
• How to determine **when** and **what** to save in non-volatile memory
  – Energy consuming

---

1Chain: Tasks and Channels for Reliable Intermittent Programs
Alexei Colin, Brandon Lucia, OOPSLA 2016
function fibonacci(n){
    a = 1;
    b = 1;
    
    while(i<=n){
        c = a + b;
        a = b;
        b = c;
    }
    
    return b;
}

main {
    i = 0;
    while(TRUE){
        fibonacci(i+++);
    }
}

1,1,2,3    1,1,2     1,1   1,1   1,1

Reboot
Reboot
Reboot
How to Design Programs

Save computational state into **non-volatile** memory.
**Recover state** when there is sufficient energy and continue.
The program can be interrupted at any time!
Trade-offs

The more frequent you save the computational state, the more overhead you introduce!

The less frequent you save the computational state, the more redundant computations you need to do!

How to ensure consistency of the non-volatile memory? When and what to save?
Exercise: Buble Sort

Implement **Power-Aware** Bubble Sort on array $A$ in non-volatile memory

**Which** variables to store in non-volatile memory?

**When** to update the variables in non-volatile memory?
Conclusions

• **RF-Based TPCs** are emerging
• There are lots of *research opportunities* in this domain
  – Communication Protocols
    • Physical layer
    • MAC layer
    • Routing, Synchronization
  – Programming Platforms
  – Operating Systems
  – Safety, security
  – Many more…
Thank You!