Beaglebone: The Perfect Telemetry Platform?

Matt Ranostay
ELC 2013
Ranostay Industries
http://ranostay.org
mranostay@gmail.com
What is the Beaglebone?

- Low cost AM335x SoC platform
  - $89 from various source (Mouser, Digikey, Amazon)
- 3.3v I/O can interface with many ICs and sensors out of the box, or with cheap 3.3v to 1.8v/5v logic converter
  - I2C
  - SPI
  - GPIO
- 1.8v I/O Analog In (ADC) pins
Introduction to Capes

- Daughter cards that connect to the expansion headers
- Examples of ones that currently exists and provide addition functionality
  - Audio Cape
  - DVI Cape
  - Camera Cape
  - Weather Cape
- Expansion header allows easy breadboard access
Starting a Beaglebone Telemetry Project

• What to design?

• How to design it?
  • Fritzing – Open Source circuit design
  • More hobbyist + breadboard friendly
  • Eagle PCB – Not free but affordable for hobbyists
  • Larger learning curve + schematic first designing.

• Materials and hardware skills required
  • Soldering Iron + Minimal electrical engineering knowledge

• Software skills
  • Any language for reporting
Practical Applications

- Weather Reporting Station
- Radiation Monitoring
- Earthquake Detection Mesh Network
- Home Security System
- Entropy Pool Generation
What kind of data can we report?

- Barometric
- Temperature
- Radiation Exposure (Counts Per Minute)
- Earthquakes
- GPS + Orientation + Compass Heading
- Ambient Light
How can we share data?

- Cosm (Formerly Pachube)
  - Free for typical usage
- Allows reporting of almost any sensor possible via feeds
- Simple JSON or EEML interface
- Handles graphing all datasets and points
  - Allows settings triggers when thresholds are peaked (Twitter or HTTP POST)
def read_bmp085_pressure():
    f = open("/sys/bus/i2c/drivers/bmp085/1-0077/pressure0_input")
    return "%.2f" % (int(f.read().strip()) / 100.0)

....

def start_reporting(pac, w1_serial_id):
    while True:
        if w1_serial_id:
            pac.update([eeml.Data("w1-temp", read_w1_temp(w1_serial_id))])
            bmp085_pressure = read_bmp085_pressure()
            pac.update([eeml.Data("bmp085-temp", read_bmp085_temp())])
            pac.update([eeml.Data("bmp085-pressure", bmp085_pressure)])
            pac.update([eeml.Data("sht21-humidity", read_sht21_humidity())])
            pac.update([eeml.Data("sht21-temp", read_sht21_temp())])
            pac.update([eeml.Data("tsl2250-lux", read tsl2250_lux_value())])
            pac.put()
            sleep(1)

if __name__ == "__main__":
    API_KEY = get_env_value("COSM_KEY")

    API_URL = "#/feeds/%s.xml" % feed

    pac = eeml.Cosm(API_URL, API_KEY, use_https = False)
    start_reporting(pac, w1_serial_id)
Sample COSM Data

<eeml xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://www.eeml.org/xsd/0.5.1" xsi:schemaLocation="http://www.eeml.org/xsd/0.5.1 http://www.eeml.org/xsd/0.5.1/0.5.1.xsd" version="0.5.1">

<environment>

<data id="w1-temp"><current_value>25.00</current_value></data>
<data id="bmp085-pressure"><current_value>1004.00</current_value></data>
<data id="sht21-humidity"><current_value>39.74</current_value></data>
<data id="tsl2250-lux"><current_value>0</current_value></data>
<data id="sht21-temp"><current_value>2.45</current_value></data>
<data id="bmp085-temp"><current_value>25.50</current_value></data>

</environment>

</eeml>
Developing Telemetry Platform

- How are you going to place everything?
  - SMD, THT, etc

- Cost of sensors and how important accuracy and precision are
  - You get what you pay for…

- Sensors interfaces to use
  - I2C, SPI, Analog In, etc, etc

- Pins to use and what other capes you many want to use
DT Overlays/Not Capebus

• Consists of DT fragments that modify the live tree
  • Each fragment can target a DT node to add/remove/override settings

• Geiger Cape for instance has fragments that setup various subsystems
  • PWM – Duty period and percent
  • GPIO + LEDS – Events and status
  • Analog In – Voltage reading across Geiger tube
Cape Manager

- Extends DT overlays to implement “firmware” for each cape design + version
  - Example -> firmware/capes/cape-bone-geiger-00A0.dts

- Allow override for prototype capes (those without an identification EEPROM)

- Doesn’t require rebuilding all of the device tree for changes in one cape
  - Parsing of version information that can toggle features and fix erratas
Cape Manager + DT Overlays

... Continued on next page ...
Cape Manager + DT Overlays Cont.

```c
bone-cape-geiger {
    compatible = "bone-cape-geiger";
    status = "okay";

    pinctrl-names = "default";
    pinctrl-0 = >&bone_geiger_cape_pins>

    pwms = >&chrpwm1 0 500000 0>
    pwm-names = "bone-geiger-cape";

    pwm-frequency = <20000>; /* 20KHz */
    pwm-duty-cycle = <60>;    /* 60% - 500V LND712 */
    event-blink-delay = <30>; /* 30ms */

    gpios = >&gpio4 17 0;    /* pulse */

    vsense-name = "AIN5";    /* analog vsense */
    vsense-scale = <37325>;  /* scaling */
};

... Continued on next page ...
Confirm Cape Manager picked up device

```bash
# cd /sys/devices/ocp.2/bone-cape-geiger.11
# ls
  counter  modalias  run  uevent
  driver  power  subsystem  vsense
```

(start running)

```bash
# echo 1 > run
```

(power LED turns on and the event LED lights up on a “click”)

Display counts:

```bash
# cat counter
4344
```

Display VSENSE (voltage feedback loop) in Millivolts:

```bash
# cat vsense
538004
```
Geiger Counter

- Reason for picking this project
  - Simple + Fun
  - Practical purpose
  - Nature gives a perfect test source since atoms are always decaying
  - Excuse to test out various consumer and scientific items for radioactivity
Geiger Counter Continued

- Data points provided are simple “clicks” in time
- All the magic happens in how you report and display the data
- Remote stations need to take in account for power usage
  - CPUFreq ‘powersave’ governor e.g. `cpufreq-set -g powersave`
  - Adjust sample update rate
  - Offload any data processing upstream off the device.
- Note that AM335x is not low power.. About 2.5 watts on the Beaglebone
Subsystems Used

Beaglebone

DT Overlays + Cape Manager

Pin Control

Weather Cape

Geiger Cape

I2C

One Wire

GPIO/eCAP

PWM
Geiger Counter Reporting Flow

- Radiation Event
- eCAP/GPIO Interrupt Event
- COSM Reporting
- COSM Graphing + Alerts

- GPIO Event - ~1 millisecond
- eCAP Event - ~36 microseconds
### Geiger Cape Pinout

**Table 11. Expansion Header P9 Pinout**

<table>
<thead>
<tr>
<th>SIGNAL NAME</th>
<th>PIN</th>
<th>CONN</th>
<th>PIN</th>
<th>SIGNAL NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>GND</td>
<td>1</td>
<td>2</td>
<td>GND</td>
<td></td>
</tr>
<tr>
<td>VDD_3V3EXP</td>
<td>3</td>
<td>4</td>
<td>VDD_3V3EXP</td>
<td></td>
</tr>
<tr>
<td>VDD_5V</td>
<td>5</td>
<td>6</td>
<td>VDD_5V</td>
<td></td>
</tr>
<tr>
<td>SYS_5V</td>
<td>7</td>
<td>8</td>
<td>SYS_5V</td>
<td></td>
</tr>
<tr>
<td>PWR_BUT*</td>
<td>9</td>
<td>10</td>
<td>A10</td>
<td>SYS_RESETn</td>
</tr>
<tr>
<td>UART4_RXD</td>
<td>T17</td>
<td>11</td>
<td>U18</td>
<td>GPIO1_28</td>
</tr>
<tr>
<td>UART4_TXD</td>
<td>U17</td>
<td>13</td>
<td>U14</td>
<td>EHRPWM1A</td>
</tr>
<tr>
<td>GPIO1_16</td>
<td>R13</td>
<td>15</td>
<td>T14</td>
<td>EHRPWM1B</td>
</tr>
<tr>
<td>I2C1_SCL</td>
<td>A16</td>
<td>17</td>
<td>B16</td>
<td>I2C1_SDA</td>
</tr>
<tr>
<td>I2C2_SCL</td>
<td>D17</td>
<td>19</td>
<td>D18</td>
<td>I2C2_SDA</td>
</tr>
<tr>
<td>UART2_TXD</td>
<td>B17</td>
<td>21</td>
<td>A17</td>
<td>UART2_RXD</td>
</tr>
<tr>
<td>GPIO1_17</td>
<td>V14</td>
<td>23</td>
<td>D15</td>
<td>UART1_TXD</td>
</tr>
<tr>
<td>GPIO3_21</td>
<td>A14</td>
<td>25</td>
<td>D16</td>
<td>UART1_RXD</td>
</tr>
<tr>
<td>GPIO3_19</td>
<td>C13</td>
<td>27</td>
<td>C12</td>
<td>SPI1_CS0</td>
</tr>
<tr>
<td>SPI1_D0</td>
<td>B13</td>
<td>29</td>
<td>D12</td>
<td>SPI1_D1</td>
</tr>
<tr>
<td>SPI1_SCLK</td>
<td>A13</td>
<td>31</td>
<td>VDD_ADC(1.8V)</td>
<td></td>
</tr>
<tr>
<td>AIN4</td>
<td>C8</td>
<td>33</td>
<td>GND_ADC</td>
<td></td>
</tr>
<tr>
<td>AIN5</td>
<td>A5</td>
<td>35</td>
<td>A5</td>
<td>AIN5</td>
</tr>
<tr>
<td>AIN6</td>
<td>B7</td>
<td>37</td>
<td>A7</td>
<td>AIN3</td>
</tr>
<tr>
<td>AIN0</td>
<td>B6</td>
<td>39</td>
<td>C7</td>
<td>AIN1</td>
</tr>
<tr>
<td>CLKOUT2</td>
<td>D14</td>
<td>41</td>
<td>C18</td>
<td>GPIO0_7</td>
</tr>
<tr>
<td>GND</td>
<td>43</td>
<td>44</td>
<td>GND</td>
<td></td>
</tr>
<tr>
<td>GND</td>
<td>45</td>
<td>46</td>
<td>GND</td>
<td></td>
</tr>
</tbody>
</table>

- Green – Ground
- Red – Power Supplies
- Blue – PWM
- Yellow – GPIO/eCAP event
- Purple – I2C
- Black – Analog In
Sensor Selection (Geiger Counter)

- Geiger tubes have various features of quality
  - Dead time
  - Sensitivity
  - Type of radiation it can detect
    - LND-712 was selected since it can detect all three main types of radiation (alpha + gamma + beta). Most expensive tube of its type.
    - SBT-9 (Soviet-Era tube) was a close second but not as sensitive as the LND or available in North America
    - SBM-20 (Soviet surplus) is an good low end choice (gamma + beta only)
- Voltage needed to register counts vary greatly
  - LND-712 – 500 volts
  - SBT-9 and SBM-20 (and most Soviet tubes) – 300 to 400 volts
Geiger Counter Circuit

- Tube is a large capacitor (in size only, and is only ~3 picofarads) that gets charged up, and when hit by ionizing radiation causes an “overcharge”. Excess charge has to go somewhere and that causes a “click”

- Logic level and pulse shifter design
  - GPIO version extends pulse to ~1 millisecond
  - eCAP version only level shifts
Safety (Geiger Counter)

- High voltage is sourced to the tube. Current boost converter can provide about 600V
  - Although almost no current is flowing. It still can be dangerous…
  - Analog-in pin used for feedback loop to measure voltage
  - Keep HV traces short as possible

- MOSFET can be asserted ON at 100% which is very bad
  - Never trust expansion header pin states…
  - Don’t trust software…

- Various ways of solving this in hardware
  - I2C GPIO expander to toggle states
  - Op-amp + low-pass filter to trigger board reset (SYS_RESETN pin to GND)
ELC Demo Setup (Geiger Counter)

- Geiger Cape (Prototype Rev 8) + LND-712
- 1.8” TFT Adafruit SPI interface display
- Beaglebone Rev A6
Weather + Radiation Station

- Geiger Cape (Prototype Rev 8) + SBT-9
- CircuitCo Weather Cape
- Beaglebone Rev A6
Weather + Radiation Station Continued

My Console

Weatherstation + Geiger Cape

- bmp085-pressure: 1012.52 Millibar
- bmp085-temp: 23.40 Celsius
- sht21-humidity: 8 CPM
- sht21-temp: 47.46 %
- tsl2250-lux: 24.61 Celsius
- wt-temp: 10 Lux
- wt-temp: 23.12 Celsius
PCB Design

- Geiger Cape Prototype Rev 10
Lessons Learned

- Know what functionality your device has and use it
- Avoid bitbanging interfaces that already available
- Microcontroller Unit ≠ Microprocessor Unit treat it as such
- Watch initial states
- Example of a bad states would be a PWM pin that is shared with a GPIO that gets asserted on…
- Watch GPIO states on pins…
- Cape Manager + DT Overlays will not save you every-time…
Lessons Learned Continued

- Watch the logic voltage level
- Watch the pin muxing…
- Really easy to conflict with another cape.
  - DT Overlays + Cape Manager holds the hope for the future
  - Possible damage if you aren’t careful
- Test on breadboard first
- Analog In (ADC) safety. Absolute 1.8v LIMIT!
  - Invest in some 1.8V Zener diodes, especially if you use sensors that may output higher voltage
  - For THT applications two standard rectifier diodes will give you a equivalent 1.4V Zener diode
- Show Demo! No magic smoke, no whammy!
Questions?
References

- Beaglebone SRM
  - http://beagleboard.org/static/beaglebone/latest/Docs/Hardware/BONE_SRMPDF

- Cadsoft Eagle PCB
  - http://www.cadsoftusa.com

- Adafruit Eagle Library (Beaglebone Cape Part)
  - https://github.com/adafruit/Adafruit-Eagle-Library

- Fritzing
  - http://fritzing.org

- Fritzing Parts (Beaglebone Cape Part)
  - https://github.com/ohporter/fritzing-parts
References Continued

- Cupertino (Indoors) Radiation + Weather
  - https://cosm.com/feeds/73056

- Slides + Demo Source Code
  - https://github.com/mranostay/beaglebone-telemetry-presentation
  - https://github.com/mranostay/cosm-analog

- Thanks to other Geiger Cape team members
  - Dimitris Sapountzakis (Hardware)
  - Pantelis Antoniou (Software + DT Overlay patchset)
  - Koen Kooi (Design Advice)
References Continued

- DT Overlays + Capebus patchset branch
  - [https://github.com/koenkooi/linux/tree/3.8-for-panto-rebase](https://github.com/koenkooi/linux/tree/3.8-for-panto-rebase)

- LND-712 End Window Alpha + Beta + Gamma detector datasheet

- SBT-9 Soviet-era Alpha + Beta + Gamma tube
  - [http://gstube.com/data/3004/](http://gstube.com/data/3004/)  (Russian)