Disclaimer

The opinions expressed and materials shared in this presentation are my own and may not reflect the opinions, policies, or procedures of my employer.
What is Arduino?

- A single board microcontroller platform.
- An open source electronics platform.
Why are we talking about Arduino?

- New, foundational technology appearing in many different form factors.
- No clear direction on forensic acquisition of data on this evolving platform.
Where is Arduino today?
“By 2018, 3D printing will result in the loss of at least $100 billion per year in intellectual property globally.”
Example - Arduino Phone

http://www.instructables.com/id/ArduinoPhone/

http://blog.arduino.cc/2013/08/12/diy-cellphone/

Steve Watson
Examples - ArduSat

ARDUSAT SUCCESSFULLY LAUNCHED IN SPACE - WATCH VIDEO!

Zoe Romano — August 12th, 2013

ArduSat was successfully launched in space last Sunday 4th August and it's now on its way to the International Space Station (ISS):
### Arduino Basics

<table>
<thead>
<tr>
<th>microcontroller</th>
<th>CPU, RAM and ROM on a single chip.</th>
</tr>
</thead>
<tbody>
<tr>
<td>shield</td>
<td>daughter card that sits on top of the Arduino</td>
</tr>
<tr>
<td>sketch</td>
<td>the code or application written in C++ that is uploaded to the Arduino</td>
</tr>
</tbody>
</table>
Is there data to recover?
Where is the data?

Microcontroller  Development Systems  Remote Endpoints
What is the data?

<table>
<thead>
<tr>
<th>Microcontroller</th>
<th>Development Systems</th>
<th>Remote Endpoints</th>
</tr>
</thead>
<tbody>
<tr>
<td>running applications (flash)</td>
<td>.ino (Arduino sketch)</td>
<td>cloud computing updates (Twitter, Facebook, IoT pages)</td>
</tr>
<tr>
<td>NVM, persistent (eeprom)</td>
<td>.elf (intermediate step between c++ and assembly)</td>
<td>control messages (c&amp;c of other microcontroller devices)</td>
</tr>
<tr>
<td>.csv, .txt (asci or hex on SDCARD)</td>
<td>.hex (assembly)</td>
<td>.txt, .csv</td>
</tr>
<tr>
<td>Fuses (single byte hex values)</td>
<td></td>
<td>.json (JSON calls to other applications)</td>
</tr>
</tbody>
</table>
How I approach new devices

1. What is the operating system?
2. What is the storage?
3. What is the connectivity?
4. How is the system updated, installed, accessed?
5. What are the parallels with other systems and devices?
6. What existing documentation and information exists?
Focus on the Arduino
Connection to the target

1. Chip removal (chip-off equivalent)

2. Tethered to another Arduino (computer to computer equivalent)

3. Connect to a programming port on the board (JTAG equivalent)
Connection: Chip-off

Connection: Tethered Arduino

Arduino Tutorial:
Using an Arduino as an AVR ISP (In-System Programmer)
http://arduino.cc/en/Tutorial/ArduinoISP
Connection: JTAG Equivalent

Olimex STK500v2 connected via ICSP to an Arduino UNO
Connection: ICSP

ICSP - In-Circuit Serial Programming
Software used to acquire

1. AVRDUDE - AVR Downloader/Uploader
   
   a. opensource


   c. Included in the Arduino IDE install under install directory ..:/Arduino/hardware/tools/avr/bin/avrdude.exe

2. Atmel AVR Studio
Data to acquire

Flash (32KB)

EEPROM (1KB)
  ● NVM reserved for persistence across uploads

Fuses (1B x 3 reserved)
  ● Ifuse, hfuse, efuse
  ● single byte hex configurations related to clock, bootloader and voltage (see reference slide for more detail)
Example: Read flash memory and dump hex to specific file.

```
avrdude -p m328p -c stk500v2 -P com4 -U flash:r:"[path/to/file/filename.hex]":r
```

- **-p [part number]**
- **-c [programmer]**
- **-P [com port]**
- **-U [memory operation]**
  - Note the :r: and :r to define READ
  - change ‘flash’ to eeprom, lfuse, hfuse and/or efuse to acquire reserved portions.
Software: AVRDUDE

C:\apps\avrdude -p m328p -c stk500v2 -P com4 -U flash:r:"c:/temp/arduino_uno.hex":r

avrdude: AVR device initialized and ready to accept instructions
Reading | ################################################################## | 100% 0.03s

avrdude: Device signature = 0x1e950f

avrdude: reading flash memory:
Reading | ################################################################## | 100% 94.89s

avrdude: writing output file "c:/temp/arduino_uno.hex"

avrdude: safemode: Fuses OK (E:05, H:D6, L:FF)

avrdude done. Thank you.
Software: Atmel Studio

Full walkthrough (screenshots) of an MCU acquisition in the backup slides.
Investigations where data may be needed

1. Supply chain investigations
2. Malware analysis
3. Improvised devices
4. Automation and control systems
5. Medical, fitness
6. Security, access control
7. Drones
8. Cloud
Why does this matter?

1. Investigations and litigation are coming to this new technology area.

2. The principles applied here can be expanded to other embedded technologies.
Thank you!

Steve Watson

forensics@stevewatson.net
steve.watson@intel.com

Twitter @stevewatson
LinkedIn - watsonsteve
Backup Material
slide 5 - Icons made by www.flaticon.com
slide 11 - Icons made by www.flaticon.com
slide 17 - image created with Fritzing
## Reference - Fuses

<table>
<thead>
<tr>
<th>Bit</th>
<th>Low</th>
<th>High</th>
<th>Extended</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td></td>
<td>CKDIV8</td>
<td>Divide clock by 8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RSTDISBL</td>
<td>External reset disable</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>CKOUT</td>
<td>Clock output</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DWEN</td>
<td>debugWIRE Enable</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>SUT1</td>
<td>Select start-up time</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPIEN</td>
<td>Enable Serial programming and Data Downloading</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>SUT0</td>
<td>Select start-up time</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WDTOm</td>
<td>Watchdog Timer Always On</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>CKSEL3</td>
<td>Select Clock Source</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EESAVE</td>
<td>EEPROM memory is preserved through chip erase</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>CKSEL2</td>
<td>Select Clock Source</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BOOTSZ1</td>
<td>Select boot size</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>CKSEL1</td>
<td>Select Clock Source</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BOOTSZ0</td>
<td>Select boot size</td>
</tr>
<tr>
<td>0</td>
<td></td>
<td>CKSEL0</td>
<td>Select Clock Source</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BOOTRST</td>
<td>Select reset vector</td>
</tr>
</tbody>
</table>

AVR Acquisition with Atmel Studio

Begin
Software: Atmel Studio
1. From Atmel Studio main screen, choose ‘Debug’ then ‘Device Programming’.
2. Identify the ‘Tool’, ‘Device’, and ‘Interface’ then click ‘Apply’.
The ISP Clock frequency must be lower than 1/4 of frequency the device is operating on.
An ongoing operation is taking longer than expected.
Details: Modules->readFromMemory
You can stop waiting for the operation, in which case you may need to restart AVR studio, or wait some more.

(The timeout can be set using the "Tools->Options->Debugger->AVR Debugger->AVR Communication Timeout" option.)

[Buttons: Wait 1 more minute, Stop waiting]
Save As dialog box showing the directory 'temp' with the file name 'atmel_atmega328p' selected and the file type 'Eeprom Files (.eep)'.
AVR Acquisition with Atmel Studio

End