|  |
| --- |
| Secure Embedded Computing Systems |
| Hardware Reverse Engineering v02 |
|  |

|  |
| --- |
| Russ Bielawski, Bill Hass  6/26/2017 |

1. Outline
   1. Circuit identification (~45 minutes)

Involves carefully taking apart the assembled component to get to the circuitry and performing circuitry reconnaissance to identify areas of interest.

* + 1. Visual Analysis

Initial lay-of-the-land inspection.

* How are components mounted?

*Through-hole is your friend. BGA is your enemy.*

* Are there barriers or protections in place?

*EMF shielding and robustness coatings can make our job difficult.*

* What are the populated interfaces?

*Things like USB, vehicle connectors, and hidden connectors.*

* Where are interesting areas (depopulated pads, test-points, unsure)?

*Development and debug interfaces are typically removed before production.*

* How do components relate to one another?

*Observe general layout, components will be closest to what they interface with.*

* How is the board powered?

*You will need to power the board. Good starting point for tracing.*

* + 1. Chip Identification

Gather information about each chip on the board for use later. Build a “Bill-of-Materials” (BOM).

* What are each of the chips and what do they do?

*Identify memory, processors, and interface controllers. The more you know the better.*

* + 1. Passive Probing and Tracing

Reverse engineer the circuitry to better understand the board function and zero in on areas of interest. Draw schematics by hand as you develop an understanding during this phase.

* What are the voltage domains?

*Know voltage domains to interface with board later without creating smoke.*

* How are inputs and outputs connected to chips?

*Identify passive circuitry, draw it out, reason about it.*

* What chips are connected?

*Buses between memory and cpu or interfaces and cpu could be MitMd.*

* Where do depopulated pads and test-points connect to?
  + *This can help identify JTAG or serial interfaces and areas to be repopulated.*
  1. Active Probing (1 hour)
     1. Serial Decoding
        + 1. Wire the FTDI device to the board

Read the datasheet for the EXEL UART interface IC. Connect as necessary to A, B, C and D (if time permitting).

* + - * 1. Decode UART

HAVE THE STUDENTS ATTEMPT TO DECODE UART A & D.

* + 1. JTAGing
       1. Setup Bus Pirate and OpenOCD
          1. Solder headers for the JTAG port (TIME PERMITTING)
          2. Upgrade Bus Pirate firmware

Although outdated, I followed this: <https://research.kudelskisecurity.com/2014/05/01/jtag-debugging-made-easy-with-bus-pirate-and-openocd/>

I just used the outdated firmware version BPv3-frimware-v6.1.hex found here: <https://github.com/BusPirate/Bus_Pirate/tree/master/package/BPv3-firmware/old-versions>

* + - * 1. Install OpenOCD (TIME PERMITTING)

TODO: This is for Mac!

sudo port install openocd +buspirate

* + - * 1. Wire the Bus Pirate to the JTAG port

ARM has a pin out for the 20-pin JTAG connection here: <http://infocenter.arm.com/help/topic/com.arm.doc.dui0499d/BEHEIHCE.html>

* + - * 1. Attempt to connect to board using OpenOCD attached to Bus Pirate

First, ensure that you aren’t connected to /dev/ttyXXX with any other program, such as screen or minicom.

Create the simple configuration file from our walkthrough site and name it buspirate-simple.cfg.

Power on the board and wait two seconds. Now, try to attach:

openocd -f buspirate-simple.cfg

* + - * 1. Diagnose the problem (free exploration)

Ideally, the students would have a little bit of time (15-20 min) to explore the problem of why the JTAG port isn’t working.