Assisted Discovery of On-Chip Debug Interfaces

Joe Grand (@joegrand)
Introduction

• On-chip debug interfaces are a well-known attack vector
  – Used as a stepping stone to further an attack
  – Can provide chip-level control of a target device
  – Extract program code or data
  – Modify memory contents
  – Affect device operation on-the-fly

• Inconvenient for vendor to remove functionality
  – Would prevent capability for legitimate personnel
  – Obfuscated or password protected instead
Introduction 2

- Identifying OCD interfaces can sometimes be difficult and/or time consuming

← http://spritesmods.com/?art=hddhack
Goals

- Create an easy-to-use tool to simplify the process
- Attract non-HW folks to HW hacking
Identifying Interfaces: External

• **Accessible to the outside world**
  – Intended for engineers or manufacturers
  – Device programming or final system test

• **Usually hidden or protected**
  – Underneath batteries
  – Behind stickers/covers

• **May be a proprietary/non-standard connector**
Identifying Interfaces: Internal

- Test points or unpopulated pads
- Silkscreen markings or notation
- Easy-to-access locations
Identifying Interfaces: Internal 2

• Familiar target or based on common pinout
  – Often single- or double-row footprint
  – JTAG: www.jtagtest.com/pinouts/

← www.blackhat.com/html/bh-us-10/bh-us-10-archives.html#Jack
→ www.nostarch.com/xboxfree
Identifying Interfaces: Internal 3

- Can use PCB/design heuristics
  - Traces of similar function are grouped together (bus)
  - Array of pull-up/pull-down resistors (to set static state of pins)
  - Test points usually placed on important/interesting signals

Identifying Interfaces: Internal 4

- Might be covered by soldermask

← Linksys WRT54G2 v1.3
→ http://elinux.org/File:Peekjtag3.png
PCB

Target I/F (24 channels)

Input protection

Level translation

Status

Propeller

USB

Op-Amp/DAC

*** 2x5 headers compatible w/ Bus Pirate probes, http://dangerousprototypes.com/docs/Bus_Pirate
Propeller/Core

• Completely custom, ground up, open source
• 8 parallel 32-bit processors (cogs)
• Code in Spin, ASM, or C

*** INFORMATION: www.parallax.com/propeller/
*** DISCUSSION FORUMS: http://forums.parallax.com
*** OBJECT EXCHANGE: http://obex.parallax.com
Propeller/Core 2

- Clock: DC to 128MHz (80MHz recommended)
- Global (hub) memory: 32KB RAM, 32KB ROM
- Cog memory: 2KB RAM each
- GPIO: 32 @ 40mA sink/source per pin
- Program code loaded from external EEPROM on power-up
Propeller/Core 3

- Standard development using Propeller Tool & Parallax Serial Terminal (Windows)
- Programmable via serial interface (usually in conjunction w/ USB-to-serial IC)
USB Interface

- Allows for Propeller programming & UI
- Powers JTAGulator from bus (5V)
- **FT232RL USB-to-Serial UART**
  - Entire USB protocol handled on-chip
  - Host will recognize as a virtual serial port (Windows, OS X, Linux)
- **MIC2025 Power Distribution Switch**
  - Internal current limiting, thermal shutdown
  - Let the FT232 enumerate first (@ < 100mA), then enable system load
Adjustable Target Voltage (VADJ)

- **PWM from Propeller**
  - Duty cycle corresponds to output voltage
  - Look-up table in 0.1V increments (1.2V–3.3V)

- **AD8655 Low Noise, Precision CMOS Amplifier**
  - Single supply, rail-to-rail
  - Voltage follower configuration
  - ~150mA output current @ Vo = 1.2V–3.3V
Level Translation

- Allows 3.3V signals from Propeller to be converted to VADJ
- Prevents potential damage due to over-voltage on target device's unknown connections
- **TXS0108E Bidirectional Voltage-Level Translator**
  - Designed for both open drain and push–pull interfaces
  - Internal pull–up resistors (40kΩ when driving low, 4kΩ when high)
  - Automatic signal direction detection
  - High–Z outputs when OE low → will not interfere with target when not in use
Input Protection

• Prevent high voltages/spikes on unknown pins from damaging JTAGulator

• Diode limiter clamps input if needed

• Vf must be < 0.5V to protect TXS0108Es
On-Chip Debug Interfaces

- JTAG
- UART
JTAG

- **Industry-standard interface (IEEE 1149.1)**
  - Created for chip- and system-level testing
  - Defines low-level functionality of finite state machine/Test Access Port (TAP)

- **Provides a direct interface to hardware**
  - Can "hijack" all pins on the device (Boundary scan/test)
  - Can access other devices connected to target chip
  - Programming/debug interface (access to Flash, RAM)
  - Vendor-defined functions/test modes might be available
JTAG: Architecture

- **Synchronous serial interface**
  - $\rightarrow$ TDI = Data In (to target device)
  - $\leftarrow$ TDO = Data Out (from target device)
  - $\rightarrow$ TMS = Test Mode Select
  - $\rightarrow$ TCK = Test Clock
  - $\rightarrow$ /TRST = Test Reset (optional for async reset)

- **Test Access Port (TAP) w/ Shift Registers**
  - Instruction ($\geq 2$ bit wide)
  - Data
    - Bypass (1 bit)
    - Boundary Scan (variable)
    - Device ID (32 bit) (optional)
JTAG: TAP Controller

- State transitions occur on rising edge of TCK based on current state and value of TMS.
- TAP provides 4 major operations: Reset, Run-Test, Scan DR, Scan IR.
- Can move to Reset state from any other state with TMS high for 5x TCK.
- 3 primary steps in Scan: Capture, Shift, Update.
- Data held in "shadow" latch until Update state.
JTAG: Protection

• Implementation specific
• Security fuse physically blown prior to release
  – Could be repaired w/ silicon die attack
• Password required to enable functionality
  – Ex.: Flash erased after n attempts (so perform n−1), then reset and continue
• May allow BYPASS, but prevent higher level functionality
  – Ex.: TI MSP430
JTAG: HW Tools

- **RIFF Box**
  - www.jtagbox.com

- **H-JTAG**
  - www.hjtag.com/en/

- **SEGGER J-Link**
  - www.segger.com/debug-probes.html

- **Bus Blaster (open source)**
  - http://dangerousprototypes.com/docs/Bus_Blaster

- **Wiggler or compatible (parallel port)**
JTAG: SW Tools

- OpenOCD (Open On-Chip Debugger)

- UrJTAG (Universal JTAG Library)
  - www.urjtag.org
IDCODE Scan

- 32-bit Device ID (if available) is in the DR on TAP reset or IC power-up
  - Otherwise, TAP will reset to BYPASS (LSB = 0)
  - Can simply enter Shift-DR state and clock out on TDO
  - TDI not required/used during IDCODE acquisition
BYPASS Scan

- In BYPASS, data shifted into TDI is received on TDO delayed by one clock cycle
BYPASS Scan 2

- Can determine how many devices (if any) are in the chain via "blind interrogation"
  - Force device(s) into BYPASS (IR of all 1s)
  - Send 1s to fill DRs
  - Send a 0 and count until it is output on TDO
UART

- **Universal Asynchronous Receiver/Transmitter**
  - No external clock needed
  - Data bits sent LSB first (D0)
  - NRZ (Non-Return-To-Zero) coding
  - Transfer speed (bits/second) = 1 / bit width

*** Start bit + Data bits + Parity (optional) + Stop bit(s)***
UART 2

- **Asynchronous serial interface**
  - TXD = Transmit data (to target device)
  - RXD = Receive data (from target device)
  - DTR, DSR, RTS, CTS, RI, DCD = Control signals
    (uncommon for modern implementations)

- Many embedded systems use UART as debug output/console/root shell
UART Scan

- 8 data bits, no parity, 1 stop bit (8N1)
- Baud rates stored in look-up table
  - 75, 110, 150, 300, 900, 1200, 1800, 2400, 3600, 4800, 7200, 9600, 14400, 19200, 28800, 31250, 38400, 57600, 76800, 115200, 153600, 230400, 250000, 307200
UART Scan 3
Possible Limitations

- No OCD interface exists
- OCD interface is physically disconnected
  - Cut traces, missing jumpers/0 ohm resistors
- OCD interface isn't being properly enabled
  - System requires other pin settings
  - Non-standard configuration
  - Password protected
- Strong pull resistors on target prevent JTAGulator from setting/receiving proper logic levels
- Could cause target to behave abnormally due to "fuzzing" unknown pins

*** Additional reverse engineering will be necessary
Future Work

• **Support for other interfaces**
  - TI Spy-Bi-Wire, ARM Serial Wire Debug, Microchip ICSP, Atmel AVR ISP, Freescale BDM, LPC Bus, Flash memory (SPI NOR/eMMC NAND)

• **Level-shifting module?**
  - Target voltage > 5V for industrial/SCADA equipment

• **Logic analyzer?**
  - Interface w/ sigrok
Get It

• www.jtagulator.com
  *** Schematics, source code, BOM, block diagram, Gerber plots, photos, videos, other documentation

• www.parallax.com
  *** Assembled units, accessories

• http://oshpark.com/profiles/joegrand
  *** Bare boards
Demonstration