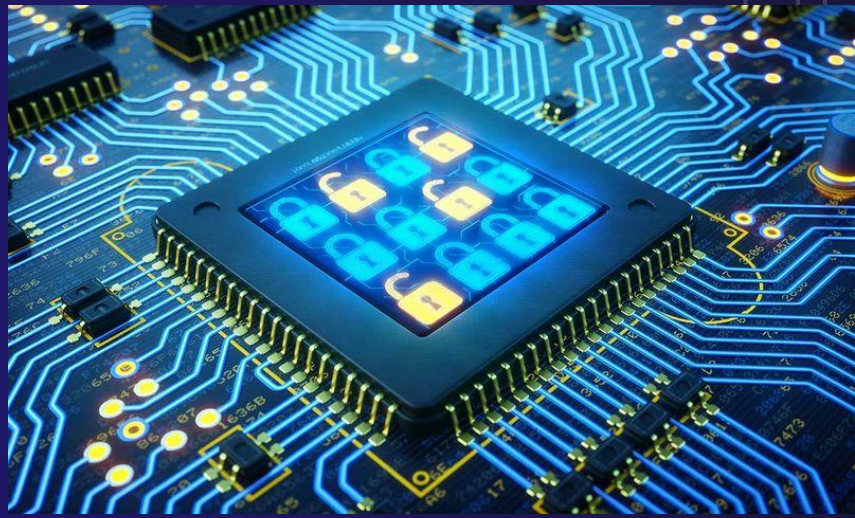
کارگاه آموزشی (تئوری و عملی)

فارنزیک سخت‌افزاری سامانه‌های صنعتی

بیست و دومین کنفرانس بین المللی انجمن رمز ایران- دانشگاه شهید بهشتی



معرفی کانال و صفحه GitHub کارگاه

ot-forensic.github.io

فهرست مطالب کارگاه آموزشی (تئوری و عملی)

- راه‌اندازی شبیه‌ساز OT

گام ۱: تکنیک‌های جمع‌آوری اطلاعات از منابع عمومی (OSINT)

گام ۲: شناسایی رابط‌های عیب‌یابی سخت‌افزاری

گام ۳: شناسایی Baud rate صحیح در ارتباط UART

گام ۴: استخراج مشخصات مهم دستگاه از لاگ‌ها

گام ۵: فعالسازی Boot menu

گام ۶: استخراج Firmware

گام ۷: یافتن رمز عبور root

اهمیت فارنزیک سخت‌افزاری OT

- بازیابی شواهد غیرقابل‌دسترسی از سیستم عامل و شبکه
- تشخیص حملات سایبری طراحی شده برای لایه سخت‌افزار مانند ارسال فریمور جعلی
- کشف بدافزارهای hook شده به Firmware که با بوت دستگاه، فعال می‌شود
- محافظت از زنجیره تأمین (Supply Chain) با شناسایی تراشه‌های تقلبی، ماژول‌های آلوده یا تغییرات غیرمجاز

انواع روش‌های فارنزیک سخت افزاری OT

۱- فارنزیک تهاجمی (Invasive)

- مزیت: دسترسی مستقیم به پایه تراشه‌ها، سیگنال‌ها، لایه‌های داخلی PCB را فراهم می‌کند.
- عیب: ممکن است به سخت‌افزار آسیب بزند یا آن را از کار بیندازد.

۲- فارنزیک نیمه‌تهاجمی (Semi-invasive) (هدف این کارگاه آموزشی)

- مزیت: دستگاه بعد از فارنزیک معمولاً سالم و قابل استفاده باقی می‌ماند.
- عیب: بسیار پیچیده و زمانبر

ساخت شبیه‌ساز OT

سخت افزارهای مورد نیاز:

- برد توسعه **NodeMCU**
- مبدل **USB** به **UART**

نرم افزارهای مورد نیاز:

- نرم افزار **Saleae Logic**
- نرم‌افزار **Putty**
- درایور مبدل **USB** به **UART** (**CH340**)

گام ۱: جمع‌آوری اطلاعات از منابع عمومی (OSINT)
بخش تئوری

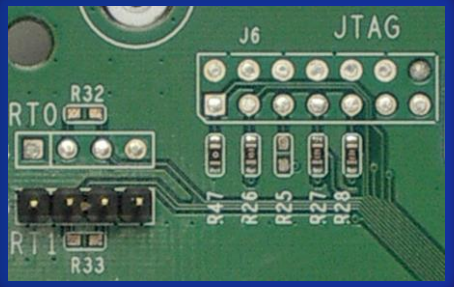
گام ۱: جمع آوری اطلاعات از منابع عمومی(OSINT)

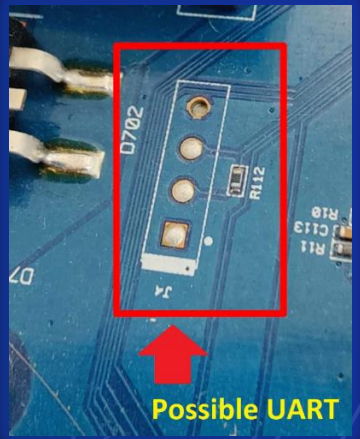
File Name	Document Type
Users Manual-2	Users Manual
Users Manual-1	Users Manual
Test Setup Photos_DFS	Test Setup Photos
Test Setup Photos_NII	Test Setup Photos
Internal Photos	Internal Photos
External Photos	External Photos
Test Report_DFS	Test Report
Test Report_NII-2	Test Report

- OSINT (Open Source Intelligence)
- اهمیت OSINT در فارنزیک سخت‌افزاری OT
- Google Dorking
  filetype:, inurl: , site:, cache:
- معرفی سایت fccid.io
- مطالعه موردی: مبدل سریال به اترنت NPort5210 شرکت Moxa

گام ۲: شناسایی و بررسی رابط‌های عیب‌یابی
بخش تئوری

گام ۲: ضرورت شناسایی رابط‌های عیب‌یابی

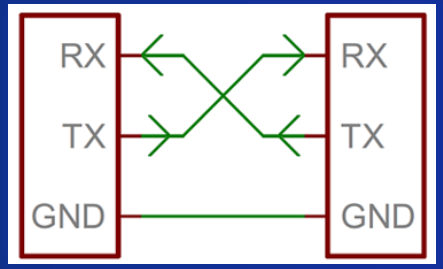




- تعبیه رابط‌های عیب‌یابی توسط سازنده محصول جهت تست و عیب‌یابی
- ضرورت شناسایی رابط‌های عیب‌یابی در فارنزیک سخت‌افزاری
- رابط‌های عیب‌یابی متداول سخت‌افزاری: **UART** و **JTAG**
- امکان خواندن و کپی گرفتن از فریمور یا حتی رجیسترهای داخلی چیپ

گام ۲: معرفی رابط سریال UART و تهدیدات آن

- استفاده از خطوط داده TX (Transmit) و RX (Receive)
- تهدیدات:



Universal Asynchronous Receiver-Transmitter (UART)

تغییر پارامترها یا پیکربندی سیستم

دسترسی غیرمجاز root

تزریق بدافزار یا ایجاد درب پشتی مانا

استخراج اطلاعات حساس (مانند کلیدهای رمزنگاری)

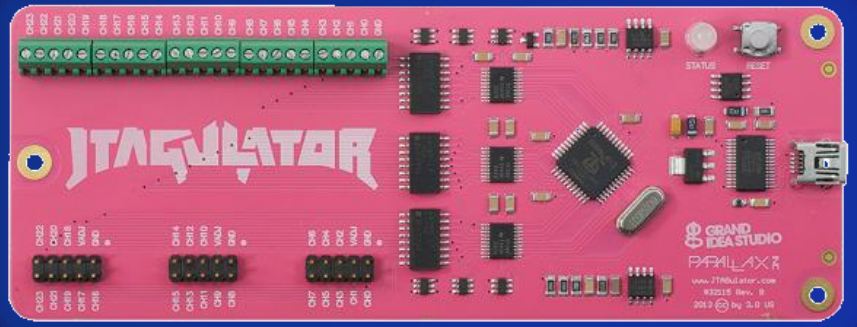
گام ۲: روش‌های شناسایی رابط UART در تجهیزات صنعتی

- شناسایی پین هدرهایی (معمولاً ۲ تا ۴ پین) که با برچسب‌های زیر مشخص شده‌اند:
  TX، RX، GND، Vcc

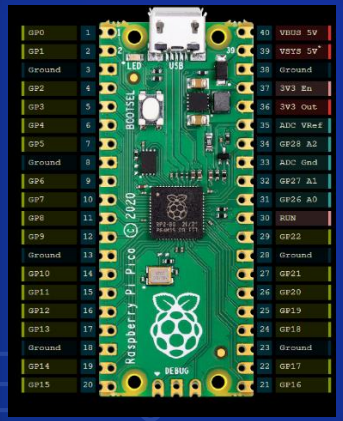
- بررسی برگه مشخصات فنی (**datasheet**)، مستندات یا منابع آنلاین

- استفاده از ابزارهای شناساگر خودکار: **JTAGulator** و **BlueTag**

گام ۲: معرفی ابزارهای JTAGulator و BlueTag



https://github.com/grandideastudio/jtagulator



https://github.com/Aodrulez/blueTag

1. screen /dev/tty.usbserial-AH05OY05 115200 (screen)

```
                                  UU  LLL
 JJJ  TTTTTTT AAAAA  GGGGGGGGGG   UUUU LLL    AAAAA TTTTTTTT 0000000  RRRRRRRRR
 JJJJ TTTTTTT AAAAAA GGGGGGG      UUUU LLL   AAAAAA TTTTTTTT 0000000  RRRRRRRR
 JJJJ  TTTT  AAAAAAA GGG      UUU  UUUU LLL  AAA AAA   TTT  0000 000  RRR RRR
 JJJJ  TTTT  AAA AAA GGG  GGG UUUU UUUU LLL AAA  AAA   TTT  000  000  RRRRRRR
 JJJJ  TTTT  AAA  AA GGGGGGGGG UUUUUUUU LLLLLLLL AAAA  TTT 000000000  RRR RRR
  JJJ  TTTT AAA   AA GGGGGGGGG UUUUUUUU LLLLLLLLL AAA  TTT 000000000  RRR RRR
  JJJ  TT                GGG             AAA                        RR RRR
 JJJ                     GG              AA                           RRR
JJJ                      G               A                            RR

          Welcome to JTAGulator. Press 'H' for available commands.

:h
JTAG Commands:
I   Identify JTAG pinout (IDCODE Scan)
B   Identify JTAG pinout (BYPASS Scan)
D   Get Device ID(s)
T   Test BYPASS (TDI to TDO)

UART Commands:
U   Identify UART pinout
P   UART passthrough

General Commands:
V   Set target I/O voltage (1.2V to 3.3V)
R   Read all channels (input)
W   Write all channels (output)
J   Display version information
H   Display available commands
:j
JTAGulator FW 1.2.2
Designed by Joe Grand, Grand Idea Studio, Inc.
Main: jtagulator.com
Source: github.com/grandideastudio/jtagulator
Support: http://www.parallax.com/support
:
```

گام ۲: برقراری ارتباط سریال از طریق UART

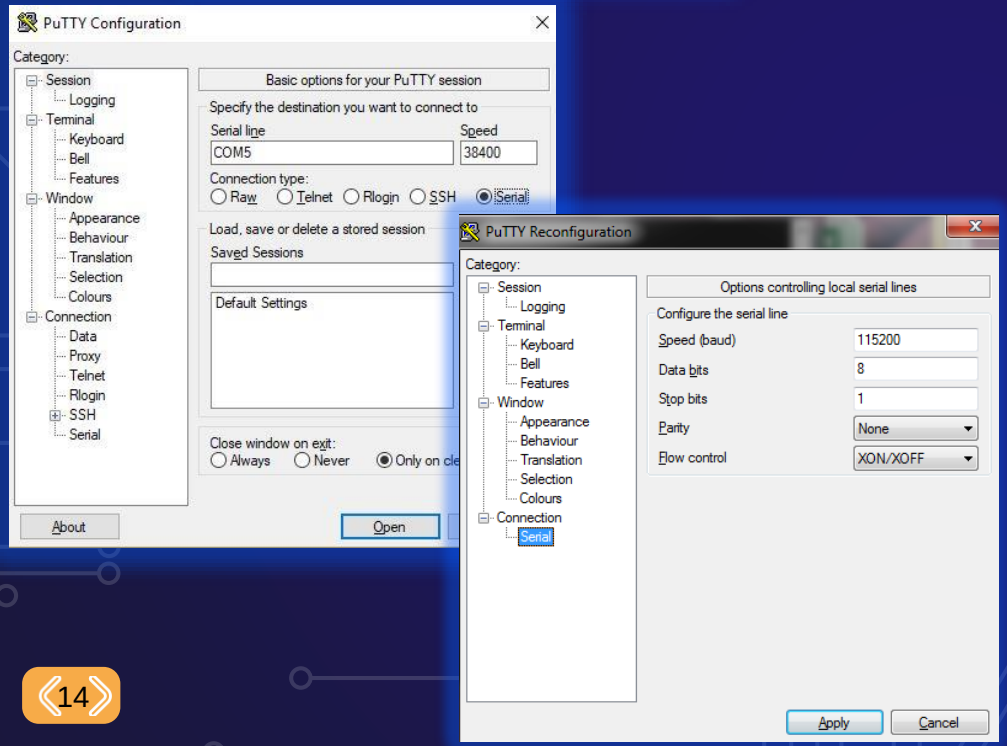
```
screen /dev/ttyUSB0 115200
picocom /dev/ttyUSB0 -b 115200
```

- معرفی نمونه ابزارهای ویندوز و لینوکس

- پارامترهای پیکربندی ارتباط سریال

- **Baud Rate (e.g. 115200)**
- **Data Bits (usually 8)**
- **Stop Bits (typically 1)**
- **Parity (None, Even, or Odd)**
- **Flow control (typically None)**

- سرعت‌های متداول در ارتباط سریال
9600, 19200, 38400, 57600, 115200



گام ۳: شناسایی Baud rate صحیح در ارتباط UART
بخش تئوری

گام ۳: معرفی Logic Analyzer

روش‌های متداول برای شناسایی **Baud rate**:

- استفاده از **OSINT**، حدس بر اساس نوع تراشه یا **Firmware**
- سعی و خطا با نرخ‌های انتقال داده پیش‌فرض رایج:
  9600, 19200, 38400, 57600, 115200
- ایجاد اسکریپتی که به طور خودکار لیستی از نرخ‌های انتقال داده رایج را بررسی کند.
- استفاده از ابزار **Logic Analyzer**

گام ۴: استخراج اطلاعات از لاگ Boot
بخش عملی: پیاده‌سازی در شبیه‌ساز OT

گام ۵: فعالسازی Boot menu
بخش عملی: پیاده‌سازی در شبیه‌ساز OT

گام ۶: استخراج Firmware
بخش تئوری

گام ۶: استخراج و تحلیل Firmware

- استخراج Firmware از حافظه غیرفرار flash که از رابط SPI استفاده می‌کند

```
Spiread <addr> <len>     # SPI flash read from given address and length
```

- تحلیل Firmware استخراج‌شده: استخراج stringها، استفاده از ابزاری مانند Binwalk برای استخراج فایل‌های فشرده در Firmware و نرم‌افزارهای Ghidra و IDA Pro برای مهندسی معکوس و تحلیل عمیق کد

```
binwalk -e firmware.bin  # Extracts recognized files and compressed data from the firmware.

strings -n 6 firmware.bin # Finds all strings of 6+ characters inside the firmware.
```

- مقایسه هش Firmware سالم با Firmware آلوده: به‌منظور شناسایی هرگونه تغییر غیرمجاز مانند تزریق کد، ایجاد درب پشتی، تغییر در تنظیمات یا اسکریپت‌های راه‌اندازی

گام ۷: یافتن رمز عبور root
بخش عملی: پیاده‌سازی در شبیه‌ساز OT

جمع‌بندی و توصیه‌های امنیتی

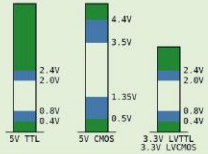
- دریافت نسخه‌های معتبر و به‌روز **Firmware** فقط از منابع رسمی سازنده
- تطبیق **Firmware hash** با مقدار **hash** اعلام شده در وب‌سایت رسمی سازنده
- محدود کردن و کنترل دقیق فرآیند به روزرسانی **Firmware** از طریق کانال‌های امن و توسط افراد مجاز
- ثبت و لاگ‌گیری دقیق تمام تغییرات مانند به روزرسانی **Firmware**
- رمز نگاری **Firmware** و استفاده از امضای دیجیتال برای **Firmware**

Hardware Forensics Cheat Sheet

By Azimi

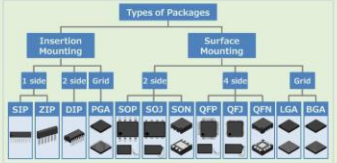
PCB Reconnaissance

Visually inspect PCB for: chip markings, test points, silk screen labels, manufacturer logos, part numbers, debug interfaces (JTAG, UART, SWD)
Power analysis: Identify GND and Vcc
Helpful Tools: Multimeter (continuity & voltage)
Common voltage levels:



Common IC packages

List chips and their functions by looking up datasheets
Helpful online tools: Chipdb, Findchip
Trace hidden or unlabeled pins
Techniques: Continuity Tracing



Common Debugging Interfaces

Look for headers or pads labeled
Use a multimeter to trace lines from microcontroller pins to potential test points

UART	SPI	I2C	JTAG	SWD
TX	SCK	SCK	TCK	SCK
RX	MOSI	SDA	TDI	SDIO
	MISO		TDO	
	CS		TMS	

TX: Transmit data RX: Receive data Optional pins: RTS: Request to Send, CTS: Clear to Send, DTR: Data Terminal Ready	SCK: Serial Clock MOSI: Master Out Slave In MISO: Master In Slave Out CS: Chip Select
SCK: Serial Clock Line SDA: Serial Data Line	TCK: Test Clock TDI: Test Data In TDO: Test Data Out TMS: Test Mode Select TRST (optional)

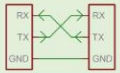
Logic Analyzer

Capture and analyze digital signals from multiple lines to understand data communication
Decodes standard communication protocols (e.g., UART, SPI, I2C, JTAG, SWD)
Typically supports 8–32 channels
Popular Tools: Saleae Logic , Sigrok/Open Logic Sniffer
Helps identify baud rates by:

$$B_d \approx \frac{1}{T_{length}}$$

UART

Universal asynchronous receiver / transmitter
A simple serial protocol for debugging
Common USB-to-UART convertors: FTDI, CP2102, CH340
Common serial Monitor Software: PuTTY, Tera Term, Picocom
Note: gibberish in terminal output — it's often a baud mismatch
Standard baud rate: 4800, 9600, 19200, 38400, 57600, 115200, 230400, 460800, 921600
UART possible capabilities: unauthorized root access, modify system configuration, install backdoor, extract sensitive information (e.g: passwords, encryption keys)
Low Level Security Feature bypass tip: Do not connect the converter TX pin



JTAG

Joint Test Action Group
Full control over CPU and memory
No standard connector for JTAG. Often 6, 10, 14, or 20-pin headers
JTAG possible capabilities:
Memory dumping (flash/NAND/NOR/eMMC),
Reverse engineering firmware,
Modify firmware to introduce backdoors,
Bypass security mechanisms.
Note: JTAG may be disabled or fused off in production devices
Many vendors obfuscate or remap JTAG pins
Hardware: Bus Pirate, JTAGulator, SEGGER J-Link, JTAG Programmer for Xilinx
Software: OpenOCD, UrJTAG, Flashrom, SEGGER tools
Bypass hardware security locks: Some boards have jumpers or resistors when configured or bridged, can activate JTAG access.

Useful Commands

```
ls usb                                  # List connected USB devices
ls -lart/dev                            # Show newly added device files (sorted by time)
dmesg | grep ttyUSB                     # Check if USB-to-Serial adapter was detected
ls /dev/ttyUSB*                         # List active serial ports
sudo picocom -b 115200 /dev/ttyUSB0     # Connect to UART
xxd -r -ps name.bin                     # Convert hex dump back to binary
strings -n <min_len> file               # Extract ASCII strings from binary
hexdump -C x.dat | less                 # View file in hex + ASCII format
openocd -f interface/<interface_type>/<interface_cfg> -f target/<target_cfg>   # Run OpenOCD with custom interface and target
/usr/share/openocd/scripts/interface/<interface_type>/<interface_cfg>          # Interface configs
/usr/share/openocd/scripts/target/<target_cfg>         # Target configs
telnet localhost 4444                   #Connect to OpenOCD
dump_image <filename> <address> <size>  # Dump firmware
```

Hardware Debug Pinout Identification Tool

JTAGulator: Identify JTAG pinouts tools when possible permutation is high
JTAG scans: IDcode scan (faster without TDI pin), Bypass scan (slower with TDI pin), Combined scan
Identify UART pins and baud rate, Identify SWD, GPIO test
Supports OpenOCD and UART passthrough mode
Voltage range: 1.2V–3.3V
BlueTag (2024): A lightweight, Raspberry Pi Pico-based alternative to JTAGulator that detects JTAG and SWD pinouts (does not support UART and operates with 3.3V).

Firmware Dump & Analysis

Step 1: Dump Firmware from Flash

```
Spiread <addr> <len>    # SPI flash read from given address and length
Nand read <flash-offset> <Len> [dst_addr]   # Read from NAND
```

Step 2: Identify File System in Dump

* SquashFS – Read-only, compressed FS
* UBIFS – Used with raw NAND flash
* JFFS2 – Older NAND/flash file system

Step3: Analyzing & Reversing the Dumped Firmware

```
binwalk dump.bin            # Analyze firmware file structure
binwalk -e dump.bin         # Extract files automatically
binwalk -E dump.bin         # Entropy analysis of binary
strings dump.bin | less     # Search for readable strings
hexdump -C dump.bin | less  # Raw hex+ASCII view
```

Reverse Engineering Tools: IDA pro, Ghidra

Step 4: Compare Dumps (Identify Manipulation)

```
diff -u original.bin modified.bin   # Quick binary diff
```

با تشکر از
شرکت‌کنندگان کارگاه فارنزیک سخت‌افزاری سامانه‌های صنعتی

- آدرس GitHub کارگاه آموزشی: ot-forensic.github.io
- کانال کارگاه آموزشی فارنزیک سیستم‌های صنعتی در پیام‌رسان بله
- کانال هوش تهدید صنعتی در پیام‌رسان بله